UNIVERSITY OF MACEDONIA

Machine learning-based mood classification via standardized questionnaires

by

Michaela Balkoudi

A thesis submitted in partial fulfillment for the
Master 's degree:
Artificial Intelligence and Data Analytics

in the
Department of Applied Informatics

Supervising Professor: Dimitrios Hristu-Varsakelis

27/6/2023

Declaration of Authorship

I, Michaela Balkoudi, declare that this thesis titled, "Machine learning-based mood classification via standardized questionnaires" and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed: _____

Date: _____

Knowledge is power. Information is liberating. Education is the premise of progress, in every society, in every family.

Kofi Annan

UNIVERSITY OF MACEDONIA

Abstract

Department of Applied Informatics

Master 's Degree

by Michaela Balkoudi

In recent years, there has been an increase in awareness of mental health issues and it is widely accepted that their early detection is essential to preventing social consequences. The use of questionnaires is a common medical technique for promptly detecting mental health concerns. Some scientists have proposed further automating the diagnosis of one mental condition by utilizing a questionnaire that diagnoses another condition. However, more research and study are required in order to prove the effectiveness of this further automation of the diagnosis of mental disorders and make it practical. This thesis investigates two questions. First whether a standardized memory questionnaire known as the PRMQ (Prospective and Recall Memory Questionnaire) along with a few demographic and general health-related questions, may be used to diagnose depression. Second, we try to investigate the reverse, that is whether memory-related disorders may be diagnosed in patients by using a common questionnaire that makes a diagnosis of depression called the ZUNG Depression Questionnaire (SDS), coupled with the same demographic questions and health-related questions used in the first investigation. The selection of these two mental illnesses is not arbitrary; rather, it is based on their usual co-occurrence and the link that has been found between them. Both questions will be investigating via machine learning techniques. More specifically, question is approached in two ways: as a regression and as a classification task. For each such task, suitable machine learning models are applied and compared in order to find the one with the best performance. The memory-related classification task will turn out to be an imbalanced classification problem, hence appropriate methods, such as resampling during training and cost-sensitive algorithms, are used to resolve it. Our results show that we can diagnose depression through the memory questionnaire, coupled with some demographic questions and health-related questions with an accuracy of approximately 79%. The diagnosis of memory-related issues via the Zung depression questionnaire could not be achieved with adequate accuracy. This does not necessarily imply that we can not diagnose memory-related issues from a depression questionnaire, but more research is needed to improve performance.

Acknowledgements

I would like to extend my gratitude and appreciation to all those who made significant contributions to the completion of my Master's thesis. This work would not have been accomplished without their assistance, guidance, and encouragement. First and foremost, I want to express my heartfelt gratitude to Dimitrios Hristu-Varsakelis, my thesis supervisor, for his indispensable support throughout my research. His knowledge, understanding, and continuous availability have been instrumental in shaping and refining my argument. I am sincerely grateful for his unwavering support and the countless hours they dedicated to reading and providing valuable criticism. I also want to acknowledge the professors of the MSc in Artificial Intelligence and Data Analytics program in the Department of Applied Informatics at the University of Macedonia. Their valuable ideas, information, and expertise shared throughout my academic journey have greatly influenced my research and intellectual growth. I am deeply grateful to the volunteers who generously offered their time and assistance to this study. Their willingness to share their experiences and insights significantly enhanced the results of my research. My appreciation extends to my family and friends for their consistent support, love, and patience as I pursued my education. Their unwavering encouragement and belief in my abilities have been a constant source of inspiration. I am indebted to my friends for their friendship, stimulating conversations, and assistance with this study project. Their valuable advice, inspiration, and willingness to help have been truly priceless. Furthermore, I would like to express my thanks to the Department of Applied Informatics at the University of Macedonia for their assistance and provision of resources. The availability of libraries, research facilities, and academic resources has greatly facilitated the completion of this thesis. Last but not least, I would like to convey my sincere gratitude to all the authors, researchers, and academics whose works and research I have cited in my thesis. Their contributions have provided the framework for my research and have deepened my understanding of the topic. In conclusion, I want to express my sincere gratitude to everyone who contributed to the completion of my Master's thesis. Their guidance, encouragement, and support have been essential to my academic and personal development.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Mental health is receiving a growing amount of attention in recent years, in light of the global, structural threats including public health emergencies, war, social and economic inequality, changes in cultural and societal attitudes, changes in the "modern" way of life and the climate crisis. A clinically significant disturbance in a person's cognition, emotional regulation, or behavior is an indication of mental disorder. Mental disorders come in a wide variety of forms. They are also termed mental health conditions, a broader notion that encompasses mental illnesses, psychosocial disabilities, and (other) mental states linked to considerable distress, functional impairment, or risk of self-harm.

The broadest assessment of global mental health since the turn of the century was recently published by the World Health Organization [1]. Nearly a billion people worldwide, including 14% of teenagers, lived with a mental illness in 2019, with anxiety and depressive disorders being the most prevalent [2]. More than one death out of every 100 is a result of suicide, and 58% of suicides happen before the age of 50. One in every six years of life is spent with a disability, with mental diseases being the primary reason. Most often owing to physical illnesses that can be prevented, people with severe mental health disorders pass away 10 to 20 years earlier than the general population. Depression is frequently brought on by childhood sexual abuse and bullying victimization. Due to the COVID-19 pandemic, there were significantly more people in 2020 who suffered from anxiety and depressive illnesses. According to preliminary estimates, major depressive disorders and anxiety disorders have each increased by 28% and 26% in just one year [3].

The process of identifying mental health issues entails numerous steps and is not one that can be completed quickly. Generally, the diagnosis will typically start with a detailed interview that is jam-packed with inquiries regarding symptoms, medical history and physical examination. More specifically, mental illness is typically diagnosed based on the individual's self-report,

which necessitates the use of questionnaires created to identify particular emotional or social interaction patterns [4]. Additionally, psychological tests and diagnostic tools are offered and are used to determine whether a person has mental health issues. Serious societal repercussions from mental illness necessitate novel prevention and intervention. The process of early mental health detection is crucial. Many people with mental illness or emotional disorders should be able to heal with the right care and therapy [5]. However, oftentimes people who are facing mental or emotional challenges are unsure as to whether their particular situation warrants/requires professional help. This may result in unnecessary visits to mental health professionals or, worse, in mental health conditions that are not diagnosed and treated in a timely manner. It is therefore necessary to have a medical method that makes a quick and reliable assessment of mental health in order to easily detect mental health issues.

A common method for making a preliminary diagnosis of mental health conditions is the use of questionnaires, which are easily accessible to the general public and easy to complete. The use of questionnaires gives people the opportunity to find out from home if they have a possible mental health issue, so that they then seek treatment with a professional. Thus, early treatment can prove to be a lifesaver for their treatment and prevent serious health consequences or even death. On the other hand, by using questionnaires, people can also easily ascertain that they do not have any problem with their mental health and can avoid scheduling an appointment with a doctor, at further expense and inconvenience. The questionnaires also benefit the health care system by not conducting unnecessary tests to diagnose mental health issues, which is especially important at this time when the healthcare system is burdened by the pandemic and medical staff is under pressure. In general, it can be said that the use of questionnaires to diagnose mental disorders helps to save resources in the healthcare system, which can be allocated to other areas in need.

The automation of diagnosis using questionnaires has given some scientists the idea of further automating the diagnosis of one mental disorder by using a questionnaire that diagnoses another mental disorder. This idea, however, needs further study in order to prove its usefulness and to enable it to be put into practice. Along these lines, this thesis undertakes two related investigations. The first will explore whether a common memory questionnaire that diagnoses memory-related disorders called PRMQ (Prospective and Recall Memory Questionnaire) together with some demographic questions, can diagnose depression. The second investigation will study the reverse, i.e. whether a common questionnaire that makes a diagnosis of depression called SDS (Zung Self-Rating Depression Scale) questionnaire together with the same demographic questions, can be used to diagnose memory-related disorders in patients. The choice of these two mental disorders was not random but rather based on the high frequency of these two disorders and the link which has been shown to exist between the two of them which will be further analysed below.

1.0.1 Depression and the link with Memory loss

As previously mentioned, according to the WHO, approximately one billion people suffer from mental disorders [6] and over 300 million people experience depression around the world [7]. Despite the existence of efficient therapies, the primary issue is to offer precise and early risk detection. Self-reporting surveys are used by medical professionals to further their diagnoses, although such surveys have several drawbacks, such as social stigma and lack of awareness. In the literature, there are many tools proposed for evaluating the severity of depression and screening MDD, or recognizing susceptibility to MDD. The Zung Self-rating Depression Scale (SDS) and Beck Depression Inventory (BDI), among others, are two of those tools. The Beck Depression Inventory [8] is a 21-question multiple-choice self-report inventory, which is a psychometric test for evaluating the severity of depression. The SDS and BDI are useful for diagnosing MDD, although they have drawbacks, e.g. they can be affected by cultural adaptation, and respondents can "cheat" and give answers which they feel "expected" of them.

Apart from the fact that depression is one of the most widespread mental disorders worldwide, it has a significant impact on one's health and quality of life. Suicidal thoughts are more common in those with depression. Every year, almost 800,000 individuals die by suicide. Its signs and symptoms, which include melancholy and mood swings, have been found to impair memory, higher-order thinking, and decision-making [9]. However depression is a complex diagnosis that has a wide range of effects on how the brain functions, including memory and cognition. People with depression claim to have trouble remembering specific memories. This implies that depression can have an impact on both declarative and autobiographical memories, among other memory categories [10]. Major depressive disorder (MDD) [11] is characterized by memory issues [12]. Researchers have also identified a link between depression and several forms of memory loss, including short-term memory impairment and dementia-related memory loss [13]. The International Neuropsychiatric Disease Journal reports that some antidepressants may also have an impact on memory [13].

The link between depression and memory has been the subject of much research. A 2013 study [14] indicated that participants' performance on memory tasks suffered while depressed. Depression, it was concluded, might impair memory. In [15] the authors found that those with depression had poorer executive function, memory and attention compared to controls. These cognitive deficits are "a key hallmark of depression" according to the authors. Working memory deficiencies were also linked to depressive thoughts, [16]. Moreover, the work in [17] showed that depression can also interfere with autobiographical memory, which includes the events that helped shape one 's identity. According to [17], sadness can result in a bias toward remembering more memories that are unfavorable to them and less capacity to recollect memories that are favorable to you. Self-reported memory issues can be impacted by depressive

symptoms. In [18] the authors found that those with more depression symptoms self-reported memory issues at higher rates than those with fewer symptoms.

The effects of depression on cognitive function, such as working memory and long-term memory, can persist even after you have reached remission, according to [19]. Working memory is the capacity to store and use knowledge temporarily, as when we need to recall a phone number or follow straightforward instructions. According to recent studies [20], people who are depressed usually have little trouble remembering "good" memories. People with depression memory loss typically find it easier to recall unpleasant memories than pleasant ones, compared to non-depressed people.

1.0.2 Memory loss

When one has both memory loss and depression, they should not automatically conclude that depression is to blame, especially if the memory problems affect their ability to carry out daily tasks. Other than depression, a number of factors can impair memory [12], including Alzheimer's disease or other forms of dementia, head injuries, Parkinson's disease, multiple sclerosis, a brain tumor, infections, lack of sleep, medications, age-related memory loss, stroke, thyroid issues, alcoholism or other substance abuse, vitamin B12 deficiency, mild cognitive impairment (MCI), and others. Finding the cause of memory problems is the first step in receiving the best treatment. A doctor could advise one to take certain memory tests if they feel that memory loss is a concern. Additional testing may also be advised including a brain magnetic resonance imaging (MRI) scan to look for brain damage or blood tests to check for signs of infection.

Testing for cognitive problems can be performed in a number of ways. A typical cognitive examination can consist of [12]:

- Account of medical history: The subjects' physical and mental health background is examined by a doctor. The participant can be questioned about any illnesses that run in their family as well as their current medication intake.
- Physical exam or diagnostic tests: A physician will take subjects' blood pressure, listen to their heart and lungs and take urine or blood samples for lab analysis.
- Cognitive test: This test measures the subjects' capacity for memory, thought, and problem-solving. While some cognitive tests are quick, others are more involved and time-consuming. Some physicians administer computerized cognitive testing.
- Neurological exam: The subjects' speech, eye movement, reflexes and coordination are evaluated. An imaging test of the brain may also be performed.

Other possible causes of cognitive issues can be ruled out by the subjects' physician. A neurologist or a mental health specialist, such as a psychologist, psychiatrist, or therapist, may be recommended, depending on the results of the examination. Even a slight memory loss might be a sign of a more serious issue, therefore anyone experiencing it should always consult a doctor. If the doctor suspects that depression is to blame, asking about memory-boosting techniques and getting a mental therapy referral are options.

1.0.3 Machine Learning on Questionnaire data for Mental health disorders

Machine learning (ML) methods can be used to probe a subjects ' mental health and evaluate their emotional state. Because this thesis deals with questionnaire data, similar works in the literature on machine learning models applied to questionnaire data for predicting mental health disorders are discussed next.

A recent mental health study, [21], applied machine learning based on questionnaire data to create an effective selective screening of mild cognitive impairment (MCI) which is one of the first indications of dementia among elderly populations. In that study, decision trees were applied to target population groups more susceptible to suffering from mild cognitive impairment and were used for cost-effective selective screening of the disease. At first, the study gathered a variety of demographic and lifestyle features along with information about patient medications. Detection of possible cases of MCI was made by the usage of the Short Portable Mental Status Questionnaire (SPMSQ) [22] and the Mini-Mental State Examination (MMSE). Finally, machine learning technique was used to classify people at risk of MCI.

In [23] ML algorithms were applied on data coming from a Mental Disorder Questionnaire (MDQ) to screen Mental Health. Two different types of questionnaires were used in this study. The first Self Reporting Questionnaire-15 (SRQ-15) contained 15 questions about general mental disorders with a Yes/No response option. Five questions were included for each of the five main mental health problems indicated in the second Self Reporting Questionnaire-25 (SRQ-25), for a total of 25 items. Supervised Machine Learning was employed to label the training data set. Finally, the results were compared with manual testing using appropriate algorithms. Specifically, Logistic Regression was implemented on MHS (Mental Health Screening). On MMDS (Multiple Mental Disorder Screening), Decision Tree Classifier, SVM (Linear Kernel), SVM (RBF-Radial Basis Function), and Naive Byes were applied for a prediction about mental health through Screening Questionnaires. For MHS, the Accuracy of Logistic Regression was more than 90%. For MMDS, the Accuracy of SVM linear (78%) was the highest compared to other algorithms.

In [24] the authors described the use of cognitive behavioral performance data integrated with Machine Learning to create a novel and objective diagnostic tool for anxiety and depression differentiation. A high degree of anxiety, a high level of depression, a high level of anxiety and depression, and the controls, were the four major symptom categories that subclinical participants were assigned questionnaires. Six different cognitive behavioral tasks were used to test the participants' cognitive activities in order to access different biases. A Random Forest algorithm was used to assess the data after that, and the model assigned individuals strictly based on their combined cognitive performance. That study confirmed the efficacy of the combination of the cognitive-behavioral test battery analyzed by the ML algorithm as a diagnostic tool, capable of distinguishing between anxiety and depression, based on cognitive-behavioral performance patterns and no self-report measures, in order to support differential diagnostic decisions.

In [25] studied data of adult patients who underwent diagnosis of Attention deficit hyperactivity disorder (ADHD) over the past few years. For that experiment, a hybrid approach made up of an ML model and a knowledge-based model was utilized with clinical data and surveys. The authors demonstrated a 95% accuracy rate and their approach has been used in a clinical setting. They suggested utilizing an extreme learning machine (ELM) to automatically diagnose ADHD, as ELM performed better than SVM. In [26] machine learning was used on questionnaire data to predict types of psychiatric disorders in child mental health clinics in London and Dhaka. In that paper, a computerized algorithm forecast child psychiatric diagnoses based on the symptom and impact scores obtained from questionnaires completed by parents, teachers, and the children themselves.

It is worth noting that some studies produced conflicting results and deficiencies, such as they apply different machine learning tools to assess the depression status and the selection of the calibration questionnaires. Additionally, there were no studies that looked specifically at the evaluation procedures and findings of the depression status in the Chinese soldier population, particularly in Chinese recruits. Hence, a study presented in [27], has been made to assess the predictive and diagnostic capability of three machine learning methods (decision trees, neural networks, support vector machines (SVM)) that evaluate the depression status, based on the BDI. The results could be useful for providing more tools for the evaluation of the depression status, and for estimating how much subjects are prone to that disease.

1.0.4 Contribution and impact

This thesis investigates whether

- (M from D): responses to a questionnaire designed to diagnose depression can be used to identify memory-related disorders
- (D from M): responses to a separate questionnaire used to detect memory disorders, can be used to also detect depression

Based on the outcomes, we will determine whether the results of one test can be used to accurately predict the results of the other and vice versa. If either question can be answered positively, then the link between memory disorders diagnosis and depression diagnosis using questionnaires will be strengthened and the reliability of diagnosis of mental disorders through questionnaires will be further enhanced. This will also result in a step forward on the automated diagnosis on mental health conditions. By means of this automation people who do not need further treatment will be spared from unnecessary trips to the doctor, and avoid unnecessary expense, disruption, stress and inconvenience. The health system will also benefit by avoiding unnecessary examinations, which equates to savings in resources, i.e. medical staff and equipment. At the same time, people who may not be aware they need help may be "nudged" towards seeking it, and therefore be given early treatment that can be life-saving, as late diagnosis can often lead to irreversible results.

If the results are unclear we will conclude that we cannot diagnose depression from the memory disorders questionnaire or we cannot diagnose memory disorders from the depression questionnaire or neither of the two questionnaires that diagnose one disease can diagnose the other. This does not necessarily mean that we cannot diagnose the one mental disorder from the other, but it does mean that further research is needed to be done on the questionnaires and it guides and triggers new efforts to rebuild the memory and depression questionnaires.

1.0.5 Methodology

We will analyze responses collected from an online questionnaire created by a team of mental health doctors at Papageorgiou Hospital. This questionnaire has two sub-questionnaires built in, one dedicated to memory and one to depression. From the answers of the memory questionnaire, a memory numerical score and a memory "class" are generated. Similarly, from the answers to the depression questionnaire, a depression numerical score and a depression "class" are generated. Thus, the two aforementioned questions: (M from D) and (D from M) are converted into four prediction tasks. More specifically, the (M from D) questions corresponds to either a regression task that predicts the memory score or a classification task that predicts memory class (having memory disorders or not). Similarly, the (D from M) question corresponds to either a regression task, predicting the depression score, or a classification task

predicting depression class (having depression or not). For the prediction of each of the two tasks the latest machine learning models, techniques and statistical analysis are employed.

1.0.6 Thesis Outline

In Chapter 2, we will describe the structure of the study, the data which was collected and the machine learning models to be used. We also detail the way in which the target variables for memory disorders and depression diagnosis were created, and how these were used to formulate a regression problem and a classification problem for each. Finally, the statistical methods applied to select the variables used as predictor variables in each of the four prediction tasks created are presented. In Chapter 3, we summarize the basic theory behind each machine learning algorithm, model and technique used in this study. In Chapter 4, we present performance metrics and general results from each machine learning technique used. Chapter 5 focuses on the memory classification problem, which belongs to a special class of classification problems called imbalanced classification. We present the specific methodologies applied, as well as comments and results from the various tests carried out.

Chapter 2

Data Collection and the Problem Statement

2.1 Data Description

In this thesis, the data consists of 3340 responses to an online questionnaire along with the response times to each question. This data comes from an inter-clinical research called "Memory And Depression Study: MANDY" under the auspices of the First University Psychiatry and Neurology Clinic of Papageorgiou Hospital at Thessaloniki. Study and data collection protocol approved by the Papageorgiou Hospital of Thessaloniki Scientific Council on June 16, 2021. Approval nr. 136 (Ref. nr. 19728/15-06-2021). In this research, an online questionnaire (Appendix C) was created, which is a version of the PRMQ (Prospective and Retrospective Memory Questionnaire) containing questions about memory (Appendix C) in conjunction with the SDS questionnaire (Zung Self-Rating Depression Scale) (Appendix C) containing also questions about depression and some demographic questions and other health related questions. For the sake of simplicity in the questionnaire, the questions were coded. The full questionnaire is given in the Appendix C. It contains 20 questions about depression and mood coded as Q1, Q2,...,Q20, 16 questions related to memory coded as Mem01, Mem02,..., Mem16 and 5 questions for electronic participation consent coded as Cons01, Cons02,...,Cons05. Moreover, it has 19 questions which relate to demographic and basic health information, coded as Dem00, Dem01,..., Dem18. Finally, it has two self-assessment questions about memory and mood coded as SEM01, SEM02. After removing the 5 consent participation features Cons01-Cons05, participant reference codes, and URLs, dates of entry of responses and the responses to the questions coded as Dem10 and Dem17, the dataset contains 62 features which after encoding are 131 in number.

The online questionnaire is used to assess in a timely manner whether people are experiencing problems with their memory or mood (e.g., Alzheimer's disease, mild cognitive impairment, Parkinson's disease, lewy type dementia, subjective memory disorders, depression, etc.). In practice, after taking approximately 15 minutes to complete the questionnaire, participants can find out free of charge if they are having trouble with their memory or mood and may possibly need help from a medical professional. The participants of the study were all adults, residents of the Region of Central Macedonia and Thrace, or able to travel to the examination site of the Memory Mental Functions Clinic of Papageorgiou Hospital at Thessaloniki.

2.2 Coding responses from PRMQ and SDS questionnaires

It is known that before applying any statistical method or machine learning model to the data, we should perform some data preprocessing to transform raw data to well-formed data sets in order to proceed with analytics. In this data set, the data come from questionnaires, so it is necessary to pay special attention when coding the data. The questionnaire data are in a text format that computers can not recognize, therefore we should convert them to numbers. First, the responses to questions about memory and depression from the two questionnaires were encoded into numbers, as they were measured on the Likert scale. As it is known, Likert scale or Likert-type questions are often used to rank the level of agreement with a statement on a scale from 1) Strongly disagree to 5) Strongly agree, however they have a wide range of uses and can also measure items such as frequency, quality importance and satisfaction. Individual Likert-type questions are generally considered ordinal data, since the items have clear rank order, although they do not follow an even distribution.

In this thesis, memory responses measure frequency and have thus been coded according to the mental health professionals' instructions in the following way: Very often=4, Often=3, Sometimes=2, Seldom=1, Never=0. The responses to questions about depression also measure frequency, but they were divided according to codification into two categories. In the first category, depression data was coded in the following way: Always=4, Often=3, Sometimes or Seldom=2, At all=1. In the second category depression data was coded in the following way: At all=4, Sometimes or Seldom=3, Often=2, Always=1.

2.3 Creating memory score and class

After coding answers to the memory questions: Mem01, Mem02,..., Mem16 into numbers, we summed up all the coded answers by column and the result was the "PRMQ" variable, which will be a memory score, a numeric target variable that helped to identify memory disorders.

According to mental health professionals the value 42 is an acceptable cut-off for the numeric memory score, in order to create a binary variable "PRMQ class", a dichotomous memory score, that takes the value 0 when "PRMQ" variable is less or equal to 42 and 1 otherwise. When a subject 's PRMQ class is 1 and (the numeric memory score is greater than 42), it means that the subject may have a memory disorder and needs the help of a mental health professional, otherwise the person likely does not need treatment about memory disorders.

2.4 Creating depression score and class

A similar process was applied to the depression answers. More specifically, the answers to the depression and mood questions: Q1, Q2,...,Q20 were also coded into numbers as described above. Then, they were summed and the total was our "Zung depression" variable, which is a numeric depression score, the numeric target variable that helps detect depression problems. According to professionals the value 44, is an appropriate cut-off for the depression score. Thus, we created a binary variable "Zung depression class", which is a dichotomous depression score that takes the value 0 when Zung depression score is less or equal to 44 versus 1, otherwise. In particular, when the Zung depression score is 1 (the numeric depression score is greater than 44) then there is an indication for some depression disorders and the subject may likely benefit by consulting a mental health professional. When the numeric depression score is less or equal to 44 (the Zung depression class is 0), the person likely does not need treatment for depression.

2.5 The four prediction problems

2.5.1 Problem 1 (D-from-M score): Depression score via memory

For the first question called "Depression via memory", we investigated whether the responses to the PRMQ questionnaire (Mem01,Mem02,...,Mem16) that diagnoses memory related issues together with the demographic and health related responses (Dem00,Dem01,...,Dem18 excluding Dem10 and Dem17), two two self-assessment questions about memory and mood-depression (SEM01, SEM02) and the response time data can be effective for diagnosing depression. This task is essentially a regression problem for predicting the numeric Zung depression score variable, using as predictors the aforementioned data.

2.5.2 Problem 2 (D-from-M class): Depression class via memory

The first question called "Depression through memory" was also investigated by the classification task that predicts the depression class or Zung depression class variable. This classification

task used as predictors the responses to the PRMQ questionnaire (Mem01, Mem02.., Mem016) that diagnoses memory related disorders together with the demographic and health related responses (Dem00,Dem01,...,Dem18 without Dem10 and Dem17), two self-assessment questions about memory and mood-depression (SEM01, SEM02) and the response time data. The one of our aims was to classify a person into two states. If the person was classified to the class 1, it means the person needs mental health professional help about depression. Otherwise, if the person was classified to the class 0, it means the person does not need treatment for depression. The proportions of the two classes of Zung depression class variable in our data set are presented in the Table 2.1:

Zung class	**percentage**
class 0: Without Zung depression	55.57%
class 1: With Zung depression	44.43%

Table 2.1: The proportion of the two classes of the Zung depression class variable

From the proportion of the two classes of the Zung depression class, we notice that the one class is more common than the other, with the first class (people without Zung depression) accounting for the majority of the observations and the second class (people with Zung depression) representing a smaller proportion of the total observations. Although, the difference in proportions between the two classes is not extremely large, as the difference between 55.57% and 44.43% is 11.14%. This suggests that the classes are relatively balanced, and there is not a severe class imbalance issue.

2.5.3 Problem 3 (M-from-D score): Memory score via depression

For the second question called "Memory through depression", we investigated whether the responses to the SDS questionnaire (Q1,..,Q20) that diagnoses depression and evaluates mood, together with the demographic and health related responses (Dem00,Dem01,...,Dem18 without Dem10 and Dem17), two two self-assessment questions about memory and mood-depression (SEM01, SEM02) and the response time data can be effective for diagnosing memory disorders. This is essentially a regression task for predicting the numeric memory score (or PRMQ score), using as predictors the aforementioned data.

2.5.4 Problem 4 (M-from-D class): Memory class via depression

The second question called "Memory through depression" was also investigated by trying to solve the classification problem that predicts the memory class (or PRMQ class variable). For the solution of this problem we used used as predictors the responses to the SDS questionnaire

(Q1,..,Q20) that diagnoses depression and evaluates mood, together with the demographic and health related responses (Dem00,Dem01,...,Dem18 excluding Dem10 and Dem17), two self-assessment questions about memory and mood-depression (SEM01, SEM02) and the response time data. The one of the aims of this thesis is to classify a person into two memory-related classes (class 0: without memory disorders, class 1: with memory disorders). If it is predicted that a person belongs to the class 1, it means the person needs mental health professional help for memory disorders otherwise it means the person does not mental health professional help. The proportion of the two classes of the PRMQ class variable in the whole data set is presented in the Table 2.2.

PRMQ class	**percentage**
class 1: Without PRMQ disorder	95.27%
class 0: With PRMQ disorder	4.73%

Table 2.2: The proportion of the two classes of the PRMQ class variable

From the proportion of the two classes of PRMQ memory class variable means we notice that the class 0 is much more common than the other, e.g accounting for the vast majority of the observations and the class 1 representing a very small proportion of the total observations. The large difference in proportions between the two classes suggests that there is a class imbalance issue. In such cases, the machine learning model used to predict the dicthotomous class variable might be biased towards the majority class, leading to poor performance in predicting the minority class. To address this issue, resampling techniques such as oversampling the minority class or undersampling the majority class were employed and cost-sensitive learning algorithms.

2.6 Feature Selection

One method to reduce the number of input features of a machine learning model is Feature selection. Reducing the number of input features can be advantageous, because it can help reduce the model 's cost and sometimes can increas the model 's performance. A large number of features can reduce the model's training speed and may require expansive system memory. Moreover, in some cases the model 's performance can be negatively affected if irrelevant features are included in the training set.

A commonly used technique for feature selection is a class of methods based on statistics that evaluates the association between each input feature and the target variable by applying statistical methods, and chooses those input features which have statistical significant effect on

the target variable. These methods can be fast and compelling, however the choice of statistical method used to investigate the relationship between each task 's target variable (numeric and binary) and each candidate predictor, depends on their data types. In this thesis, we will consider the statistical methods listed below. Each is described in detail in Appendix A.

Numerical target variable (PRMQ memory score variable or Zung depression score variable) with

- numerical input feature: Pearson Correlation (A.1.1)
- ordinal input feature: Spearman Rank Correlation Coefficient (A.1.2)
- binary input feature: Point biserial correlation (A.1.3)
- nominal input feature: Kruskal–Wallis H Non Parametric Hypothesis Test (A.1.4)

Binary target variable (PRMQ memory class variable or Zung depression class variable) with

- numerical input feature: Point biserial correlation (A.3.1)
- ordinal input feature: Mann-Whitney U test (A.3)
- binary input feature: Fisher 's Exact Test (A.2.1)
- nominal input feature: Chi-square test of independence (A.2.2)

2.7 Modeling process for each problem 1-4

In order to make predictions about memory score or class (M-from-D score or M-from-D class), we first exclude all features (Mem01,Mem02,...,Mem16) of the dataset that correspond to the *PRMQ memory questionnaire*. Respectively, in order to make predictions about depression score or class (D-from-M score and D-from-M class), we first exclude all features of the dataset that correspond to the SDS depression-mood questionnaire (Q1,Q2,...,Q20). After the exclusion of the aforementioned features, we follow the following procedure:

1. Define the initial input features from the questionnaire data. The initial features will be all the questions from the online questionnaire except questions about registration dates, consent in replying questions, the two demographic questions (Dem 10 and Dem17) and the respective aforementioned questions depending on the task.

2. Apply Feature Selection process by performing the appropriate statistical hypothesis test between the target variable and every input feature. The choice of the appropriate hypothesis test is based on the data type of both target variable and input feature.

3. Encode nominal categorical input features by one-hot encoding and ordinal categorical features by ordinal encoding.

4. Standardize the numeric input features, e.g. $X_{new} = \frac{X_i - X_{mean,i}}{X_{SD,i}}$

5. Use the selected features to train a variety of ML models on Problems 1-4 and identify the best model for each task.

where X_i is the i-predictor, $X_{mean,i}$ is the sample mean value of the i-predictor and $X_{SD,i}$ is the sample standard deviation of the i-predictor.

Chapter 3

Machine Learning Models

The four problems presented in chapter 1 are approached either as a classification or as a regression task. For each task, suitable machine learning models are applied and compared in order to find the one with the best performance. For the classification problem, the following models are used: Logistic regression, Gaussian Naive Bayes classifier, Support Vector Machines, K-nearest neighbor (KNN) classifier, Decision Tree classifier, Linear Discriminant Analysis and ensemble methods like boosting, bagging, stacking and voting. Regarding the boosting technique, we use AdaBoost classifier, CatBoost classifier, Gradient boosting classifier. Regarding the bagging technique, we use Bagging classifier, Random Forest classifier, Extra Trees classifier. It should be noted that the memory-related classification task is an imbalanced classification problem, hence appropriate methods, such as resampling during training and cost-sensitive algorithms, will be used to resolve it.

For the regression problem, the following models are used: Linear regression, Lasso regression (and the corresponding cross validated version called Lasso CV), Elastic net linear regression (and the corresponding cross validated version called Elastic net CV), Ridge linear regression (and the corresponding cross validated version called Ridge CV), Support Vector Regression, K-nearest neighbor (KNN) regression and the same ensemble methods applied in classification and also averaging. Regarding the boosting technique, we use AdaBoost regression, CatBoost Regression, LightGBM regression. Regarding bagging technique, we use Random Forest regression, Bagging regression, Extra Trees Regression.

3.1 Logistic Regression

Logistic regression [28] is used to model categorical outcomes or the probability that an event will occur based on a number of categorical or numerical predictor features.

Let the functions:

$$Y = f(X) \quad or \quad f : X \to Y \quad or \quad P(Y|X)$$

be such that

- Y takes on discrete values
- $X = (X_1, X_2, ..., X_n)$ is a vector of discrete or continuous random variables

for notational simplicity, only the situation when Y is a binary variable is taken into consideration.

Logistic regression considers a parametric form for the distribution $P(Y|X)$ and estimates its parameters from training data. The parametric model listed below is taken into account if Y is binary:

$$P(Y = 1|X) = \frac{1}{1 + exp\{-(b_0 + \sum_{i=1}^{n} b_i X_i)\}} \tag{3.1}$$

$$= \frac{exp\{b_0 + \sum_{i=1}^{n} b_i X_i\}}{1 + exp\{b_0 + \sum_{i=1}^{n} b_i X_i\}} \tag{3.2}$$

and

$$P(Y = 0|X) = 1 - P(Y = 1|X) \tag{3.3}$$

$$= \frac{1}{1 + exp\{b_0 + \sum_{i=1}^{n} b_i X_i\}} \tag{3.4}$$

where $b_i,\ i = 1, ..., n$ are the coefficients, also known as the model parameters or regression coefficients and represent the relationship between the predictor variables (features) and the probability of a binary outcome. The coefficients determine the impact of each predictor variable on the *log-odds* of the outcome.

Thus, the distribution $P(Y = 1|X)$ is of the form $Y = \frac{1}{1+exp(X)}$ which results in a simple linear expression for classification. For the classification of a given X, the idea is to find the y_j value that maximizes $P(Y = y_j|X)$. In practice, the label $Y = 1$ is assigned when:

$$\frac{P(Y = 1|X)}{P(Y = 0|X)} > 1 \tag{3.5}$$

using the equations (2.2) and (2.4), it turns out that

$$exp\left\{b_0 + \sum_{i=1}^{n} b_i X_i\right\} > 1 \tag{3.6}$$

After applying the natural log on both sides, the following linear classification rule arises

$$Y = \begin{cases} 1 & if \quad (b_0 + \sum_{i=1}^{n} b_i X_i) > 0 \\ 0 & otherwise. \end{cases}$$

3.1.1 Learning Logistic Regression

It is necessary to estimate the n+1 parameters, $B = (b_0, b_1, b_2, ..., b_n)$, in order to specify the logistic regression model. The estimation of the parameters is based on the concept of finding the suitable values that maximize the probability of the observed data "given" the training data set.

For the estimate of these parameters, many methods are used. One technique is based on finding the parameter values that maximize the conditional data likelihood or maximizing the conditional data log likelihood. The conditional data likelihood is defined as the probability of the observed Y values in the training data, conditioned on their corresponding X values.

If Y^l is the observed data of Y in the l^{th} training sample and $L(B)$ is the conditional data log likelihood, then L(B) is defined as follows:

$$L(B) = log \prod_{l} P(Y^l | X^l, B). \tag{3.7}$$

More simply, let Y be a random binary variable taking on the values $\{0, 1\}$ and

$$P(Y = y_l) = p^{y_l}(1-p)^{(1-y_l)}.$$

It is easy to see that $P(Y = 1) = p$ and $P(Y = 0) = 1 - p$.

Then, the likelihood is defined as follows:

$$\prod_{l} P(Y = y_l) = p^{y_1+y_2+...+y_N}(1-p)^{N-(y_1+y_2+...+y_N)}.$$

The log likelihood is equal to:

$$\begin{aligned} L =& log \prod_l P(Y = y_l) = (y_1 + y_2 + ... + y_N)logp \\ &+ (N - (y_1 + y_2 + ... + y_N))log(1 - p) \\ &\sum_l \{y_l log(P(Y = 1)) + (1 - y_l)log(P(Y = 0))\} \end{aligned}$$

which also can be written as:

$$\begin{aligned} L(B) =& \sum_l \{Y^l logP(Y^l = 1|X^l, B) \\ &+ (1 - Y^l)logP(Y^l = 0|X^l, B)\} \\ &= \sum_l Y^l \{log \frac{P(Y^l = 1|X^l, B)}{P(Y^l = 0|X^l, B)} \\ &+ logP(Y^l = 0|X^l, B)\} = \\ &\sum_l \{Y^l (b_0 + \sum_{i=1}^{n} b_i X_i^l) \\ &- log(1 + exp(b_0 + \sum_{i=1}^{n} b_i X_i^l))\} \end{aligned}$$

where X_i^l is the value of X_i for the l^{th} training sample.

Unfortunately, there is no closed-form solution that maximizes $L(B)$ with respect to B.One can proceed with the gradient ascent method that optimizes the weights B by calculating $\frac{\partial L(B)}{\partial b_i}$ for each b_i, where the partial derivative is equal to:

$$\begin{aligned} \frac{\partial L(B)}{\partial b_i} =& \sum_l (Y^l X_i^l - X_i^l \frac{exp(b_0 + \sum_{i=1}^{n} b_i X_i^l)}{1 + exp(b_0 + \sum_{i=1}^{n} b_i X_i^l)}) \\ =& \sum_l X_i^l (Y^l - P(Y^l = 1|X^l, B)). \end{aligned}$$

The rule for updating the weights b_i according to gradient ascent technique is given by

$$\begin{aligned} b_i(t+1) = b_i(t) + \eta \frac{\partial L(B)}{\partial b_i} = \\ \\ = b_i(t) + \eta \sum_l X_i^l (Y^l - P(Y^l = 1|X^l, B)) \end{aligned}$$

where η is the learning rate.

There is an algorithm that describes a method for estimating the parameters of logistic regression for a two-class issue using the gradient ascent methodology where the term $P(Y = 1|X)$ is calculated as:

$$P(Y = 1|X) = \frac{exp(b_0 + \sum_{i=1}^{n} b_i X_i)}{1 + exp(b_0 + \sum_{i=1}^{n} b_i X_i)} = \frac{1}{1 + exp\{-(b_0 + \sum_{i=1}^{n} b_i X_i)\}}$$

The above-mentioned algorithm uses two convergent criteria. The first criterion uses a predefined number of iterations. The second one is a measure, referred to as prospective parameter change (PPC), which is defined as the highest relative change in the parameters suggested by the parameter-change vector estimated for the next iteration. PPC is defined as follows:

$$\frac{|b_i(t+1) - b_i(t)|}{b_i(t) + \epsilon}$$

where

- $\epsilon > 0$ is a small positive constant.
- $b_i(t)$ is the current value of the parameter b_i.
- $b_i(t+1)$ is the prospective value of this parameter after adding the change vector computed for the next iteration.

When PPC is closer to zero, the repetition process stops.

In case of binary classification, $R = (b_0 + \sum_{i=1}^{n} b_i X_i)$ is calculated. Then

$$Y = \begin{cases} 1, & if \;\; R \geq 0 \\ 0, & otherwise. \end{cases}$$

3.2 Decision tree classifiers

A decision tree classifier [29] is a popular supervised machine learning algorithm utilized for classification tasks. It constructs a tree-shaped model that represents a series of decisions and their potential outcomes, enabling predictions and classification of new instances based on their features.

Initially, the decision tree begins with a single node representing the entire dataset. It then recursively divides the data based on different attributes or features, generating decision nodes and branches. The splits are determined by selecting the features that offer the most information gain or effectively separate the classes.

At each decision node, the tree assesses a specific feature and its potential values to determine the subsequent node to follow. This process continues until a stopping criterion is met, such as reaching a maximum depth, a minimum number of instances within a node, or when all instances in a node belong to the same class.

The leaf nodes within the decision tree signify the final predicted class or outcome. Each path from the root node to a leaf node corresponds to a specific combination of attribute conditions that lead to a particular class assignment. During the classification phase, a new instance traverses the decision tree by evaluating the attribute conditions at each node, ultimately arriving at a leaf node and assigning the appropriate class label.

Decision trees are renowned for their ease of understanding and interpretation, and they can handle both numerical and categorical data. However, they are susceptible to overfitting when the tree becomes overly complex and might struggle to generalize well to unseen data. Techniques such as pruning, ensemble methods (e.g., random forests), or regularization can be employed to address these concerns and enhance the performance of the decision tree classifier.

3.2.1 Decision tree algorithm

The decision tree algorithm is a method for constructing a tree-like structure that helps in predicting the class or value of a given dataset. This algorithm follows a set of steps to create the decision tree:

1. Start with the root node, representing the entire dataset.
2. Determine the best attribute to split the data based on an attribute selection measure (e.g., Information Gain or Gini Index). In a decision tree, an attribute refers to a feature or characteristic of the data that is used to make decisions and perform splits in the tree. Attributes are the properties or variables associated with the instances or examples in the dataset.
3. Divide the dataset into subsets based on the potential values of the chosen attribute.
4. Create a decision tree node that contains the selected attribute.
5. Recursively create new decision trees using the subsets of data generated in step 3. Repeat this process until further splitting is not possible, and the last node is referred to as a leaf node.

In a classification tree, the goal is to assign each observation to the most frequently occurring class in its corresponding region. The classification error rate measures the proportion of training samples in a region that do not belong to the most frequent class.

The Gini index is another measure used in classification trees. It quantifies the total variance across all classes in a region, taking into account the proportion of training samples from each class.

These measures guide the splitting process and help in determining the structure of the decision tree. By evaluating attribute values and following the branches, the algorithm can predict the class or value for new instances based on their attributes.

Overall, the decision tree algorithm provides an interpretable and intuitive way to make predictions by organizing data into a tree structure based on attribute values and their relationships.

3.3 Decision Trees for regression

In the context of regression trees [30], attributes are not explicitly defined as in classification trees. Instead, the focus is on partitioning the predictor space into non-overlapping and distinct regions (denoted as $R_1, R_2, ..., R_i$). These regions represent different subsets of feasible values for the predictor variables $X_1, X_2, ..., X_n$.

The process of building a regression tree involves the following steps:

1. Partitioning the predictor space: The predictor space is divided into regions $R_1, R_2, ..., R_i$ in a way that minimizes the Residual Sum of Squares (RSS). Each region corresponds to a subset of samples in the training set. The goal is to find the partitioning that minimizes the sum of squared differences between the actual response values (y_j) and the predicted response values ($\hat{y}_{R_i}$) for each sample j in region R_i. This helps in creating distinct regions that capture the variability in the data.

2. Predicting the response within each region: Once the regions $R_1, R_2, ..., R_i$ are defined, the same prediction is assigned to all the samples within a specific region. This prediction is typically the average of the dependent feature (response) values for the training samples in that region. The average value serves as the prediction for any new sample that falls within that region.

By dividing the predictor space into distinct regions and assigning predictions to each region, a regression tree is constructed. The resulting tree can capture complex relationships between predictor variables and the response variable, allowing for nonlinear modeling and providing interpretable predictions within each region. The main objective in building a regression tree is to find the optimal partitioning of the predictor space that minimizes the RSS. This ensures that the predictions within each region are as accurate as possible given the training data.

The Residual Sum of Squares (RSS) is defined in the following mathematical formula:

$$\sum_{i=1}^{I} \sum_{j \in R_i} (y_j - \hat{y}_{R_i})^2$$

where $(\hat{y}_{R_i})$ is the mean response of the training sets in the i-th box.

3.4 Support Vector Machines for classification (SVC)

In its initial design, the support vector machine (SVM) was intended for binary classification. The SVM [31] seeks to build a hyperplane in a given p-dimensional feature space, where p is the number of features, distinctly separating the data points with different labels. Finding a plane with the maximum possible distance between data points from both classes is necessary. In case of binary classification, the primary goal of SVM is to find the optimal hyperplane, $f(w, x) = wx + b$, that separates the two classes with labels $y \in \{1, +1\}$ in a given data set with input features $x \in \mathbb{R}^p$.

The learning procedure for SVM is achieved by solving the following constrained optimization task:

$$\begin{aligned} & min \frac{1}{p} w^T w + C \sum_{i=1}^{p} \xi_i \\ & s.t \;\; y_i^{'}(wx + b) \geq 1 - \xi_i \\ & \xi_i \geq 0, i = 1, ..., p \end{aligned}$$

where

- w is a vector of weights
- $w^T w$ is the Manhattan norm
- ξ is a cost function
- C is the penalty parameter (a hyper-parameter)

Assuming that K is the kernel function and that ϕ is the projection function for the vector x_i into the high-dimensional space. The input is mapped to the feature space via this function. The following is how these two functions are related:

$$K(x_i, x_j) = \phi(x_i)^T \phi(x_j)$$

Following is the corresponding optimization problem without constraints:

$$min\frac{1}{p}w^T w + C\sum_{i=1}^{p} max(0, 1 - y_i^{'}(w_i x_i + b)) \tag{3.8}$$

where $wx + b$ is the predictor function.

The above equation's goal is known as the L1-SVM primal form problem with the standard hinge loss. The drawback of the L1-SVM is that it is not differentiable. However, the L2-SVM (the variant of L1-SVM) is differentiable and produces more stable results. L2-SVM is defined as follows:

$$min\frac{1}{p}||w||_2^2 + C\sum_{i=1}^{p} max(0, 1 - y_i^{'}(w_i x_i + b))^2 \tag{3.9}$$

Despite the SVM's initial focus on linear relationships, different kernels, such as polynomial or radial, may be used thanks to the kernel function. The kernel to be used must be chosen before an SVM method is applied.

As a result, SVM may effectively be applied to non-linear classification in addition to linear classification by using a technique known as the kernel trick, which implicitly maps inputs into high-dimensional feature spaces. In practice, it draws margins between the classes, and those margins are created in a way that maximizes the distance between the classes and the margin. The maximization of this distance minimizes the classification error.

3.5 Support Vector Machine for Regression (SVR)

By introducing a different loss function that is updated to include a distance measure, SVMs may also be used to solve regression tasks. In this case, the term "Support Vector Regression" is used (SVR). In practice, SVR [32, 33] is capable of nonlinear mapping the input x onto an m-dimensional feature space and then building a linear model inside the feature space. For each instance, there should be a smaller difference than the epsilon between the function to be learned and the output variable.

Let the task of approximation the set of data

$$(x_1, y_1), ..., (x_l, y_l), \quad x \in \mathbf{R}^N, \quad y \in \mathbf{R}$$

with a linear function: $f(x, \alpha) = (w \cdot x) + b$.

Minimizing the following empirical risk yields the optimal regression function:

$$R_{emp}(w, b) = \frac{1}{l} \sum |y_i - f(x_i, \alpha)|_\epsilon$$

With the most general loss function with ϵ-insensitive zone described as

$$|y - f(x, \alpha)| = \begin{cases} \epsilon \quad if \quad |y - f(x, \alpha)| \leq \epsilon \\ |y - f(x, \alpha)|, \quad otherwise \end{cases}$$

Finding a function $f(x, \alpha)$ that is both as flat as feasible and has at most ϵ deviation from the actual observed targets y_i is the objective. This is equivalent to minimizing the functional:

$$\phi(w, \xi_i^*, \xi_i) = \frac{1}{2}||w|| + C \sum (\xi_i + \xi_i^*)$$

where C is a pre-specified value, ϵ is an insensitive loss function and ξ and ξ^* are slack variables representing upper and lower constraints on the outputs of the system as follows:

$$\begin{aligned} &y_i - ((wx_i) + b) \leq \epsilon + \xi_i, \quad i = 1, 2, ..., l \\ &((wx_i) + b) \leq \epsilon + \xi_i^*, \quad i = 1, 2, ..., l \\ &\xi_i \geq 0 \quad and \quad \xi_i^* \geq 0, \quad i = 1, 2, ..., l \end{aligned}$$

3.6 K-nearest Neighbor (KNN)

The supervised learning technique K-nearest neighbors (KNN) [34] may be applied to classification and regression tasks. By calculating the distance between the test data and all of the training points, the KNN algorithm tries to predict the proper label for the test data. The K points that are closest to the test data are then chosen.

In case of classification, a new instance is classified by the classes of its k-nearest neighbors. A new instance is classified to the class which is most common amongst its k-nearest neighbors or otherwise it is classified by a majority vote of its neighbors.

In the case of regression, the predicted value to the new instance is the mean value of the K selected training points.

Distance functions

Some distance functions commonly used in the KNN algorithm for finding nearest neighbours are defined below:

- Euclidian

$$\sqrt{\sum_{i=1}^{k}(x_i - y_i)^2}$$

- Manhattan

$$\sum_{i=1}^{k}|x_i - y_i|$$

- Minkowski

$$(\sum_{i=1}^{k}(|x_i - y_i|)^q)^{\frac{1}{q}}$$

The previous three distance metrics are only applicable for continuous-valued variables. If there are categorical variables, the Hamming distance should be preferred which is defined as follows:

- Hamming

$$D_H = \sum_{i=1}^{k}|x_i - y_i|$$
$$x = y \Rightarrow D = 0$$
$$x \neq y \Rightarrow D = 1$$

The Hamming distance measures the number of cases where corresponding symbols are not the same in two strings of equal length.

3.6.1 KNN Algorithm

1. Initialize K value

2. For each example in the data
    - Compute the distance between the query example and the current example from the data.
    - Put the distance and the index of the example to an ordered collection
3. Sort the ordered collection of distances and indices in ascending order by the distances
4. Choose the first K elements from the sorted collection
5. Get the labels of the selected K elements
6. In case of regression, the predicted value is the mean of the K labels
7. In case of classification, the predicted label is the mode of the K labels

3.6.2 Choosing the right value for K

Selecting the optimal number for K is aided by the primary inspection of the data. Generally, a large K value often improves accuracy since it reduces the overall noise. As K is increased, the predictions resulting from averaging or majority voting become more stable and prediction accuracy is more likely to increase up to a certain point. However, the drawback is that the boundaries in the feature space start to blur. It is important to keep in mind that if a majority vote is used among labels, K usually has an odd number chosen as a tiebreaker.

Cross-validation is a usual approach to retrospectively choosing a good K value through the use of an independent data set for the validation of K value. For most data sets, the best K is 10 or more.

3.7 Naive Bayes

A simple probabilistic classification technique built on the Bayes theorem is called Naive Bayes (NB) [35, 36]. The idea of independence between each attribute value on a given class and the values of the other attributes is where the word “naive” originates.

Let

- $X = (X_1, X_2, ..., X_n)$ be a random variable
- $A_1, A_2, ..., A_n$ be the attributes of X associated with the n components $X_1, X_2, ..., X_n$ respectively.

- $T = x = (X_1 = x_1, X_2 = x_2, ..., X_n = x_n)$ be the set of training samples drawn from the population of X
- There are c classes: $C = y_1, y_2, ..., y_c$
- Each and every sample has a certain class labels, i.e. $Y = y_j \in C$

The classifier 's objective is the prediction of the class label Y when a sample X is given. NB computes $P(Y = y_j|x)$ for each class y_j, $j = 1, 2, ..., c$ to predict the class label of a given x. In practice, the predicted class label of x is that class whose probability has the highest value.

More formally, NB use the following equation to find the class label (C_{MAP}) of x:

$$\begin{aligned} C_{MAP} &= argmax_{y_j \in C} P(Y = y_j|x) \\ &= argmax_{y_j \in C} \frac{P(x|Y = y_j)P(Y = y_j)}{\sum_j P(Y = y_j)} \\ &= argmax_{y_j \in C} P(x|Y = y_j)P(Y = y_j) \\ &= argmax_{y_j \in C} P((x_1, x_2, ..., x_n)|Y = y_j)P(Y = y_j) \\ &= argmax_{y_j \in C} \prod_i^n p(x_i|Y = y_j)P(Y = y_j) \end{aligned}$$

3.7.1 Learning NB

In the process of learning NB, the training set is used to estimate the probabilities

$$P(x_1|Y = y_j), P(x_2|Y = y_j), ..., P(x_n|Y = y_j)$$

for each class y_j with $j = 1, 2, ..., c$.

The probability of $Y = y_j, \forall j \in \{1, 2, ..., c\}$ is simply estimated by the frequencies of $Y = y_j$ in the training set. Thus,

$$P(Y = y_j) = \frac{N(y = y_j, T)}{N},$$

where

- $N(y = y_j, T)$ is the number of sample of class y_j in T
- N is the number of training samples in T

For the estimation of $P(X_i = x_k|Y = y_j)$, there are two cases:

1. If X_i has discrete observations or A_i is categorical:

$$P(X_i = x_k|Y = y_j) = \frac{N(X_i = x_k, Y = y_j, T)}{N(Y = y_j, T)},$$

where

- $N(X_i = x_k, Y = y_j, T)$ is the number of samples of class y_j in T that have the value x_k for attribute A_i.

2. If A_i is continuous-valued:

- NB is called Gaussian NB and
- it is extended to numerical attributes by assuming a Gaussian distribution ($N(\mu_j, \sigma_j^2)$) for the values, i.e.:

$$P(X_i = x_k|Y = y_j) = \frac{1}{\sqrt{2\pi}\sigma_j} exp(-\frac{(x_k - \mu_j)^2}{2\sigma_j^2})$$

Thus, μ_j and σ_j, the mean and standard deviation of values of attribute A_i for training samples of class y_j should be estimated from the training set.

3.8 Linear Discriminant Analysis for classification

A linear classification and dimension reduction model is linear discriminant analysis (LDA) [37, 38]. It is a method for categorizing data, reducing its dimensions, and visualizing it. It is most frequently applied to feature extraction problems involving pattern classification.

LDA assumes that the class conditional densities $p(x|Y = k) = f_k(x)$ are multivariate Gaussian distributions for each of the K classes, i.e. $P(x|Y = k) \sim N(\mu_k, \Sigma_k)$. Therefore, the densities are defined as follows:

$$f_k(x) = \frac{1}{(2\pi)^{\rho/2}|\Sigma_k|^{1/2}} exp\{-\frac{1}{2}(x - \mu_k)^T \Sigma_k^{-1}(x - \mu_k)\}$$

In this case an unlabeled datapoint is assigned to the class k where the posterior probability $p(Y = k|x)$ is maximized.

Consideration of the two class classification is instructive. We would classify the datapoint x as $\underset{k}{argmax}(p(Y = k|x)) = \underset{k}{argmax}(logp(Y = k|x))$ if the multivariate Gaussian distributions for classes 1 and 2 are $f_1(x)$ and $f_2(x)$, respectively, with the corresponding prior probability of π_1 and π_2.

Following the application of Bayes' rule, we may now choose the class k with the highest $\underset{k}{argmax}(log(\pi_k) + log(f_k(x)))$ and eliminate the independent of k denominator. By choosing the class k with the highest $\delta_k(x)$, we arrive at the discriminant function $\delta_k(x)$ below for each class k (again emphasizing that the common 2π constant can also be removed from the maximization):

$$\delta_k(x) = log(\pi_k) - \frac{1}{2}log(|\Sigma_k|) - \frac{1}{2}(x-\mu_k)^T\Sigma_k^{-1}(x-\mu_k)$$

where the quantity $(x-\mu_k)^T\sigma_k^1(x-\mu_k)$ called (squared) Mahalanobis distance metric measures the distance between a point and a distribution (squared) Mahalanobis distance metric.

Additionally, the covariance matrices of the different classes are assumed to be equal by LDA: $\Sigma_1 = \Sigma_2 = ... = \Sigma_k = \Sigma$. When the discriminant functions are precisely identical between two classes, we may analyze the decision boundary between them:

$$\delta_1(x) = \delta_2(x)$$

$$log(\pi_1) - \frac{1}{2}log(|\Sigma|) - \frac{1}{2}(x-\mu_1)^T\Sigma^{-1}(x-\mu_1) = log(\pi_2) - \frac{1}{2}log(|\Sigma|) - \frac{1}{2}(x-\mu_2)^T\Sigma^{-1}(x-\mu_2)$$
$$0 = log(\frac{\pi_1}{\pi_2}) - \frac{1}{2}(\mu_1+\mu_2)^T\Sigma^{-1}(\mu_1-\mu_2) + x^T\Sigma^{-1}(\mu_1-\mu_2)$$

The result from the previous equation discriminates between the data from class 1 and the data from class 2 and it is linear in x. This is where the term "linear discriminant" analysis comes from.

3.9 Multiple linear regression

Multiple linear regression (MLR) [39] is a statistical method that uses a number of explanatory (independent) factors to predict the result of a response (dependent) variable. The objective of MLR is to model the linear relationship between the explanatory variables and the response variable.

In particular, MLR assumes the following relationship between a response y and some exploratory variables $x_1, ..., x_{k-1}$:

$$y = b_0 + b_1x_1 + ... + b_{k-1}x_{k-1} + \epsilon, \tag{3.10}$$

where ϵ is the model 's error term also known as the residuals.

The measurements are carried out n times resulting in n values of y for n sets of x_j.

$$y_i = b_0 + b_1 x_{i1} + ... + b_{k-1} x_{ik-1} + \epsilon_i, \quad i = 1, ..., n, \tag{3.11}$$

where

- x_{ij} is the $i - th$ observation of x_j
- The ϵ_i are not observed directly.

After the addition of the following parameters, the previous equations can be described equivalently in the matrix form

$$x_{10} = x_{20} = ... = x_{n0} = 1. \tag{3.12}$$

Thus,

$$Y = X\beta + \epsilon, \tag{3.13}$$

where

$$Y = [y_i]_n, \quad X = [x_{ij}]_{nxk}, \quad B = [b_j]_k, \quad \epsilon = [\epsilon_i]_n \tag{3.14}$$

The coordinates $b_0, b_1, ..., b_{k-1}$ of the vector B are unknown. MLR aims to estimate the vector B according to the multivariate observations, because the coefficients $b_0, b_1, ..., b_{k-1}$ are not known.

$$[X, Y] = \begin{bmatrix} x_{10} & x_{11} & \dots & x_{1k-1} & y_1 \\ x_{20} & x_{21} & \dots & x_{2k-1} & y_2 \\ \dots & \dots & \dots & \dots & \dots \\ x_{n0} & x_{n1} & \dots & x_{nk-1} & y_n \end{bmatrix} \tag{3.15}$$

The ordinary least squares (OLS) estimator is usually applied to solve this problem where

$$\sum_{i=1}^{n} (y_i - \sum_{j=0}^{k-1} b_j x_{ij})^2 \to min. \tag{3.16}$$

The OLS estimates of the unknown coordinates $b_0, b_1, ..., b_{k-1}$ solve the previous minimization problem and can also be represented as follows:

$$\hat{B} = [\hat{b}_j]_k. \tag{3.17}$$

In case of $detX^TX > 0$, the OLS estimates can be estimated from the next formula:

$$\hat{B} = (X^TX)^{-1}X^TY. \tag{3.18}$$

Let $\hat{Y} := X\hat{B}$, this equation can be rewritten in the coordinate form in the following way:

$$\hat{y}_i = \hat{b}_0 + \hat{b}_1 x_{i1} + ... + \hat{b}_{k-1} x_{ik-1}, \;\; i = 1, ..., n. \tag{3.19}$$

Thus, the predicted response value that corresponds to the predictor values $x_1, ..., x_{k-1}$ is $\hat{y}$.

The residual sum of squares (RSS) calculates the difference between the estimating model and the data as follows:

$$RSS := \sum_{i=1}^{n} (\hat{y}_i - y_i)^2. \tag{3.20}$$

R^2 is the coefficient of determination and is defined as follows:

$$R^2 := 1 - \frac{\sum_{i=1}^{n}(\hat{y}_i - y_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2} \in [0, 1] \tag{3.21}$$

It is a goodness of fit measure for linear regression models. The closer it is to 1, the better the regression model fits the data.

3.9.1 Data standardization

Linear regression analysis usually uses data standardization.

By denoting,

$$\hat{x}_j = \frac{1}{n}\sum_{i=1}^{n} x_{ij}, \;\; \hat{y} = \frac{1}{n}\sum_{i=1}^{n} y_i, \;\; S_y^2 = \sum_{i=1}^{n}(y_i - \bar{y})^2, \;\; S_j^2 = \sum_{i=1}^{n}(x_{ij} - \bar{x}_j)^2, \;\; j = 1, ..., k-1 \tag{3.22}$$

Centered and normalized variables for the original sample are got

$$v_i := \frac{y_i - \bar{y}}{S_y}, \quad w_{ij} = \frac{x_{ij} - \bar{x}_j}{S_j}, \quad i = 1, .., n, \quad j = 1, ..., k-1. \tag{3.23}$$

Let,

$$V = [v_i]_n, \quad W = [w_{ij}]_{nx(k-1)}, \tag{3.24}$$

In case $detW^TW > 0$, the following formula can be used to find the OLS estimates for the standardized model

$$\hat{B} = (W^TW)^{-1}W^TV. \tag{3.25}$$

For the following reasons, standardized data is helpful for linear regression:

- When standardized data are utilized, the solution is independent of the measuring scale.
- Instead of the absolute value x_j, the predictor input most likely depends on the relative value w_j.

3.10 Ridge regression and the LASSO

The matrix X^TX frequently approaches the singular form. Multicollinearity is the name given to this phenomenon in MLR. Although the OLS estimates can still be estimated, they are unlikely to have "good" statistical properties. Because of this, even little modifications to the data, such as dropping or adding a few observations, can cause significant variations in the estimators of the model's coefficients.

The variance inflation factors $VIF_j, j = 1, ..., k$, can be used to identify multicollinearity in the regression data. The regression matrix X is said to have a high multicollinearity if $VIF_j > 5$ or $VIF_j > 10$ for at least one j.

The estimators of the coefficients are regularized by the regression analysis techniques Ridge and LASSO (Least Absolute Shrinkage and Selection Operator). Therefore, they could be helpful to overcome the drawbacks of OLS estimators.

Prediction accuracy

In case the variables are almost linearly related and the sample size is significantly larger than the number of predictors (k), e.g. $(k << n)$, it's likely that the simple OLS estimator would provide accurate findings. The OLS estimates will often have low accuracy and considerable variation when n is not much higher than k. If $k > n$, the OLS process will not provide a unique result, and the estimator's variance will be infinite. Regularization techniques often allow for the reduction of estimate variance at the expense of slight bias introduction. As a result, prediction accuracy improves.

Model interpretability

Regression models frequently have a large number of predictors (k), but in practice, not all of them have a significant impact on the outcome. It would be simpler to comprehend the model if the factors that don't truly effect the answer were taken out. The LASSO and Ridge estimators provide an alternative to "subset selection" approaches in this regard, which reduce the number of predictors in the regression model. These regularization techniques result in linear models with coefficient estimates that are close to zero or equal to zero.

3.11 Ridge regression

The estimate of an unknown vector B by the Ridge regression based on standardized observations $\{W, V\}$ (defined in the equation 3.24) is equal to

$$\tilde{B}_\lambda := (W^T W + \lambda \mathbb{I})^{-1} W^T V, \tag{3.26}$$

where $\mathbb{I}$ is the identity matrix and $\lambda > 0$ is the parameter of regularization.

The coordinate form of the Ridge estimator is shown below:

$$\tilde{B}_\lambda = [\tilde{\beta}_j(\lambda)]_{k-1}$$

By including the "ridge" parameter λ to the diagonal elements of the matrix $W^T W$, it is possible to solve the issue of the ill-conditioned matrix $W^T W$. The ill-conditioned matrix $W^T W$ becomes the well-conditioned matrix $(W^T W + \lambda \mathbb{I})$ using this technique. As a result, the usual issues that occur when reversing the ill-conditioned matrix are resolved. Nevertheless, compared to the OLS estimate, the Ridge estimate $\tilde{B}_\lambda$ is biased.

It is worth noting that the Ridge estimate $\tilde{B}_\lambda$ solves the following equivalent minimization problems:

- min$(||V - WB||^2 + \lambda ||B||^2)$
- For all $\lambda > 0$, there is a $t(\lambda) > 0$ such that
  min$(||V - WB||_2)$ subject to $||B||_2 \leq t(\lambda)$.

Therefore, the Ridge estimate may be seen as an OLS estimate after adding a penalty placed on the coefficient vector.

3.12 Lasso

The LASSO estimate $\tilde{B}_\lambda$ solves the following equivalent minimization problems based on standardized observations $\{W, V\}$.

- min($||V - WB||^2 + \lambda||B||_1$)
- For all $\lambda > 0$, there is a $t(\lambda) > 0$ such that
  min($||V - WB||^2$) subject to $||B||_1 \leq t(\lambda)$,

where $||B||_1 = \sum_{j=1}^{k-1} |\beta_j|$.

Lasso penalizes the coefficient vector β_j, $j = 1, ..., k-1$, in a slightly different way from Ridge. Ridge multiplies the λ parameter by the l_2-norm of the vector $(\beta_1, ..., \beta_{k-1})$, whereas LASSO uses the l_1-norm.

In terms of model interpretability, one benefit of the LASSO approach is that it produces a model, as opposed to the Ridge regression, where certain coefficient estimates are precisely equal to zero when λ is large. The LASSO regularization also does variable selection, making the model easier to understand.

It is important to note that different λ values in the Ridge regularization lead to various $\tilde{B}_\lambda$ vectors. Therefore, picking the right λ value is important. The following is a presentation of the cross-validation approach that may be used for this purpose.

3.12.1 Cross-validation for choosing the λ value for LASSO and Ridge

When a collection of candidate values for the λ parameter is provided, the approach that may be used to choose a "proper" value is called cross-validation. The train set and the test set are two subsets of the initial data that are separated before the algorithm is applied. The train set is used to estimate coefficients after that. The validation of these estimations is then carried out using the test data.

The term "proper" value refers to selecting a λ that yields the greatest degree of accuracy in forecasting the response values. The model tends to explain the data noise when λ is too small, which can lead to overfitting. On the other hand, if the method is unable to capture the underlying relationship, overly large λ values likely to result in underfitting. In both situations, the computation of the error using the test data yields a high value.

The algorithm is explained in some more detail below. The first step is to randomly partition the initial data set into Q blocks of the same size. While the remaining $Q-1$ blocks form the train set, one of the blocks serves as the test set. The typical range for Q is 5 to 10. Following that, the regression coefficients are calculated for each value in the grid of values $\lambda = [\lambda_s]$. These regression coefficients are used to compute the residual sum of squares presented below.

$$RSS^q_{\lambda_s} = \sum_{i=1}^{n}(y_i - \sum_{j=0}^{k-1}\hat{b}_j(q,\lambda_s)x_{ij})^2$$

where $q = 1, ..., Q$ is the block's index that has been chosen as the test set. Often the average of these RSS values over all blocks is computed.

$$MSE_{\lambda_s} = \frac{1}{Q}\sum_{q=1}^{Q} RSS^q_{\lambda_s}.$$

Afterwards, λ takes the value of λ_s that corresponds to the minimum MSE_{λ_s}.

The "one-standard-error rule" is used in a second widely used approach. The standard error of the mean is calculated for each MSE_{λ_s}. Then, the largest λ_s for which the MSE_{λ_s} is within one standard error of the minimum MSE value is chosen. From this approach, a "more regularized" model is obtained whereas the MSE is increased by not more than one standard error.

3.13 LassoCV

A regularization technique to reduce overfitting in a regression model is LASSO. By applying the L_1 regularization, it reduces large coefficients. LassoCV is a Lasso regression model with built-in cross-validation of the α parameter. When training the model, an iterative process is applied to automatically tune alpha parameter in order to choose the optimal value for a Lasso model using cross-validation, resulting in finding the optimal model for the data.

3.14 RidgeCV

RidgeCV stands for ridge regression with built-in cross-validation of the α parameter. In practice, an iterative practice is applied to automatically tune the α paramter using cross-validation during the implementation of a ridge regression model. This technique operates similarly to grid-search cross-validation, with the exception that Generalized Cross-Validation, a kind of effective LeaveOne-Out cross-validation, is carried out by default.

3.15 Ridge classifier

When compared to Logistic Regression with a l_2 penalty, the Ridge Classifier operates differently. Cross entropy is not the loss function for the Ridge Classifier. The following is how Ridge Classifier develops a classifier using the Ridge Regression Model:

For sake of simplicity, let's consider the binary classification task:

1. Convert target variable into +1 or -1 based on the class in which it belongs to.
2. Create a Ridge regression model to predict the target variable. MSE plus l_2 penalty forms the loss function.
3. If the decision function-based Ridge regression's prediction value is greater than 0, it will predict a positive class; otherwise, it will predict a negative class.

3.16 Elastic-Net

The Elastic-Net [40] is a regularised linear regression model that linearly combines both l_1 and L_2 penalties of the Lasso and Ridge regression methods respectively. With this combination, it is possible to learn a sparse model, similar to Lasso, in which just a small percentage of the weights are non-zero while yet preserving Ridge's regularization properties. The $l1\_ratio$ parameter is used to control the convex combination of l_1 and l_2.

When there are several features correlated with one another, elastic-net is helpful. Lasso will probably choose one of these at random, but elastic-net will probably choose both at once. Elastic-Net can inherit part of Ridge's stability under rotation as a result of the trade-off between Lasso and Ridge.

The objective function that is minimised by Elastic-Net is given below:

$$\underset{w}{argmin}\frac{1}{2n}||Xw-y||_2^2+\alpha\rho||w||_1+\frac{\alpha(1-\rho)}{2}||w||_2^2$$

where α and ρ are the parameters parameters that determine the trade-off between L_1 and L_2 regularization, w is the weight vector in the cost function formula and n is the the sample size.

3.17 Elastic-NetCV

ElasticNetCV is an Elastic-Net model with built-in cross-validation of the α parameter. When training the model, an iterative process is applied to select the optimal parameter α among multiple candidate alpha values for an Elastic-Net model using cross-validation. The usage of ElasticNetCV, as opposed to Grid-Search cross-validation in the Elastic-Net class, will avoid redundant computation and allow for the automated tuning of the value of alpha. To find also the optimal value of ρ in complement, standard Grid-Search cross-validation can be used.

3.18 Ensemble methods

A particular model fitting methodology or statistical learning can be improved by the use of ensemble techniques [41]. The fundamental principle of ensemble methods is to linearly mix different model fitting techniques rather than relying just on one fit..

Let's simplify things by assuming that the task of function estimation has the goal of estimating a real-valued function.:

$$g : \mathbb{R}^d \to \mathbb{R}$$

using the data $(X_1, Y_1), ..., (X_n, Y_n)$ where X is a d-dimensional predictor variable and Y is an 1-dimensional response. It is conceivable to generalize to other $g(\cdot)$ functions and different data types. The function $\hat{g}(\cdot)$ can be estimated if a base process is stated and some input data are provided.

When changing the input data, we may execute a base process several times. The initial concept behind ensemble techniques was to employ reweighted original data to generate various estimates $\hat{g}_2(\cdot)$, $\hat{g}_3(\cdot)$ etc. based on various reweighted input data. The individual function estimates $\hat{g}_k(\cdot)$ may then be combined linearly to provide an ensemble-based function estimate $g_{ens}(\cdot)$:

$$g_{ens}() = \sum_{k=1}^{M} c_k \hat{g}_k(),$$

Two basic techniques of ensemble learning known as Bagging and Boosting will be presented below.

3.18.1 Bagging

Bagging which stands for Bootstrap aggregating is an ensemble learning technique that enhances the predictive performance and accuracy of machine learning algorithms. It lowers the variance of a prediction model and is employed to handle bias-variance trade-offs. Bagging is used for regression and classification tasks and especially for decision tree methods. It usually helps to address the problem of over fitting.

Let the regression or classification task. The data is given in the form of pairs (X_i, Y_i), $i = 1, ..., n$, where $X_i \in \mathbb{R}^d$ is the d-dimensional predictor variable and Y_i is the response. For regression: $Y_i \in \mathbb{R}$ and for classification with J classes: $Y_i \in \{0, 1, ..., J-1\}$

Typically, the target function of interest is:

- For regression: $E[Y|X = x]$
- For classification is the multivariate function: $P[Y = j|X = x]$, $(j = 0, ..., J-1)$

When a certain base process is used, the function estimator $\hat{g}(\cdot)$ is the outcome:

$$\hat{g}(\cdot) = h_n((X_1, Y_1), ..., (X_n, Y_n))(\cdot) : \mathbb{R}^d \to \mathbb{R},$$

where the estimator is defined as a function of the data by the function $h_n(\cdot)$.

3.18.2 Bagging algorithm

1. Build a bootstrap sample $(X_1^*, Y_1^*), ..., (X_n^*, Y_n^*)$ from the data $(X_1, Y_1), ..., (X_n, ..., Y_n)$, by drawing n times at random with replacement.
   (The initial sample size and the size of the bootstrap samples are equal.)

2. Calculate the bootstrapped estimator $\hat{g}^*(\cdot)$ based on plug-in principle or in either case, estimate a base model using the bootstrap sample.:
   $\hat{g}^*(\cdot) = h_n((X_1^*, Y_1^*), ..., (X_n^*, Y_n^*))(\cdot)$.
   (Typically, bagging uses the same based models.)

3. Steps 1 and 2 should be repeated M (M is frequently set to 50 or 100) times to get M bootstrapped estimators: $\hat{g}^{*1}(\cdot)$,$\hat{g}^{*2}(\cdot)$,..., $\hat{g}^{*M}(\cdot)$.
   (A base model is simply calculated from each bootstrap sample and provides M score functions.)

4. By aggregating these functions into the final score, the bagged estimator is obtained as follows:

- For regression: $\hat{g}_{bag}(\cdot) = \frac{1}{M}\sum_{k=1}^{M}\hat{g}^{*k}(\cdot)$
- For classification: $\hat{g}_{bag}(\cdot) = maxvoting(\hat{g}^{*1}, \hat{g}^{*2}, ..., \hat{g}^{*M})$

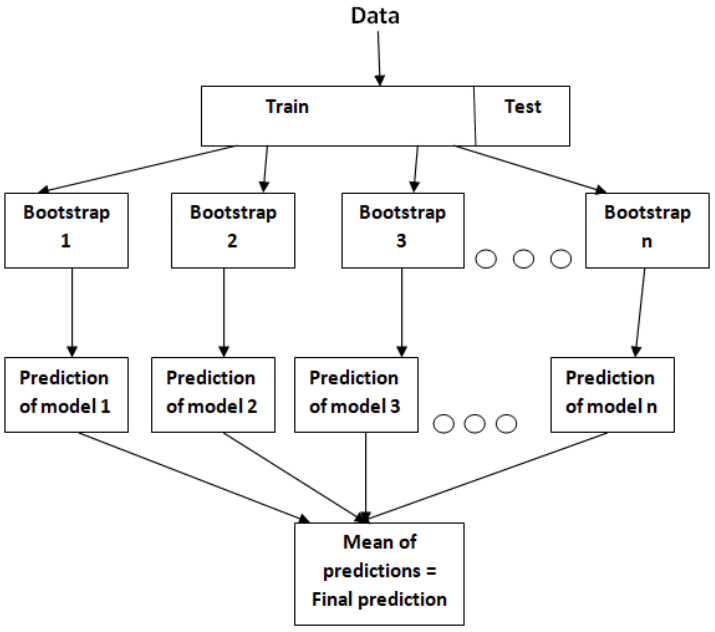

FIGURE 3.1: Bagging algorithm for regression

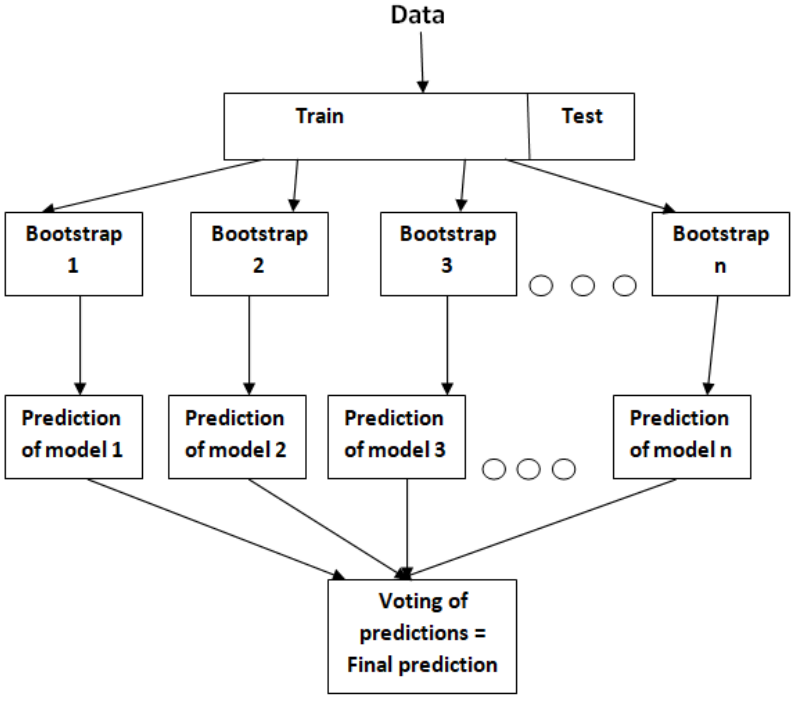

FIGURE 3.2: Bagging algorithm for classification

Bagging simply explained

- From the initial data set, M bootstrap samples of equal size are built by sampling observations with replacement.
- On each of these bootstrap samples, a base model is fitted.
- Each model is independently learnt in parallel from every training set.
- Combining all of the models' predictions results in the final predictions.

3.18.3 Bagging models

3.18.3.1 Random Forest

Random forest [42] is a supervised learning algorithm. This method applies to both classification and regression problems and is proven to be an efficient non-linear tool. It is a combination of tree predictors such that the trees are constructed independently of each other. Each tree makes predictions and the final classification decision is obtained by a majority vote law on all the classification trees. The methodology is described in more detail below.

A classification tree is defined iteratively by a division criterion (node) obtained from one of the variables, x_j, which results in the construction of two subsets in the training sample. The observations i satisfying the condition $x_j^i < T$ where the algorithm defines the real number T are contained in the first subset. Whereas the observations i that satisfy $T \leq x_j^i$ are contained in the second subset. The algorithm automatically builds the choices of the variable and of the threshold T to minimize a heterogeneity criterion. More specifically, the aim is to create two subsets that are the most homogeneous as possible in term of their values of Y. The tree stops growing when all the subsets resulting have homogeneous values of Y.

However, a single classification tree is frequently not a very accurate classifier, despite its intuitive form. Therefore, a Random Forest is an improvement of this classification method by aggregating several under efficient classification trees using a bagging procedure. According to the aforementioned, the algorithm at each step randomly chooses several observations and several variables and builds a classification tree from this new data set.

At the end, the random forest makes the classification decision using a majority vote law on all the classification trees. Also, the probability of each class can be estimated by calculating the proportion of each decision on all the classification trees. It is worth noting that the higher the number of trees in the forest results in higher accuracy in the random forest classifier, because

the generalization ability of this algorithm is good.

3.18.3.2 Random Forest classifier algorithm

Random forest classifier algorithm intuition can be divided into two stages. In the first stage, k features are selected randomly out of total m features and the random forest is constructed. In the first stage, the following procedure is applied:

1. k features from a total of m features where $k < m$ are randomly chosen.
2. Calculation of the node d using the best split point among the k features.
3. Splitting the node into daughter nodes by the best split.
4. Repetition of steps 1 to 3 until l number of nodes has been reached.
5. Repeat steps 1 to 4 for n number of times to construct n number of trees and build forest.

In the second stage, the trained random forest algorithm is used to make predictions.

1. The test features are taken and the rules of each randomly constructed decision tree are used to predict the outcome and then the predicted outcome is stored.
2. Calculation of the votes for each predicted target.
3. The high voted predicted target is considered as the final prediction from the random forest.

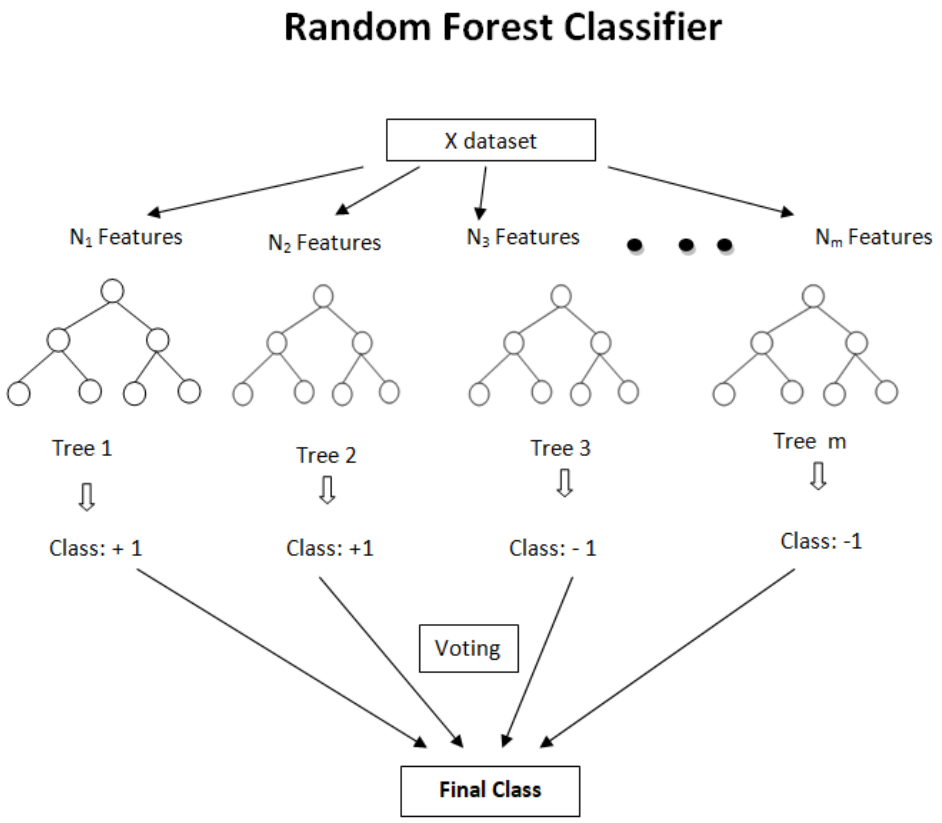

FIGURE 3.3: Random Forest Classifier

3.18.3.3 Random Forest regression

The Random Forest (RF) ensemble technique [43] generates hundreds or even thousands of decision trees, each of which functions as a regression function on its own, through "bagging or bootstrap aggregation." The average of all decision tree outputs constitutes the Random Forest regression's ultimate result.

Let Y be the output scalar, $X = \{x_1, x_2, ..., x_m\}$ be the input vector having m features, and S_n be the training set with n observations defined the following expression:

$$S_n = (X_1, Y_1), (X_2, Y_2), ..., (X_n, Y_n), X \in \mathbf{R}^m, Y \in R$$

To build a collection of q prediction trees $\hat{h}(X, S_n^{\Theta 1}), ..., \hat{h}(X, S_n^{\Theta q})$, the boosting algorithm chooses a number of bootstrap samples $(S_n^{\Theta_1}, ..., S_n^{\Theta_q})$ from S_n and applies a CART algorithm's extension to these samples. The training process ends with the creation of a prediction function of the form $\hat{h}(X, S_n)$ over S_n.

The nonparametric statistical model known as CART, or Classification and Regression Tree, is based on the construction of a decision tree. It is important to note that neither CART fixes the tree structure nor makes any assumptions about the class densities beforehand. Although the growth of the tree depends on the complexity of the input data throughout the learning procedure. Nodes and leaf nodes are the components of every decision tree. Making the optimal split among all variables is necessary at the decision tree's initial stage. This splitting process starts at the root and proceeds through each node, applying its own split function to each new input sample. This is repeated up to the terminal node, commonly known as the tree leaf. The tree stops growing after a predetermined number of levels have been reached or when a node has less observations than a predetermined threshold.

The random forest regression model, an extension of the CART technique, can yield more accurate predictions. Several de-correlated decision trees are constructed during the RF training phase. The forest is referred regarded as "random" because each tree in RF is produced using a randomized subset of predictors. X is an input vector, the family of random vectors, Θ_k, is a set of independent random vectors, and the basic classifier utilized by RF is a L tree-structured classifier: $h(X, \Theta_k)$, where $k = 1, ..., L$.

According to each tree, RF creates q outputs, $\hat{Y}_1 = \hat{h}(X, S_n^{\Theta 1})$, $\hat{Y}_2 = \hat{h}(X, S_n^{\Theta 2})$,...,$\hat{Y}_q = \hat{h}(X, S_n^{\Theta q})$. Then, by averaging all of the trees' outputs, the aggregation is completed. As a result, the estimation of the output, $\hat{Y}$, may be obtained by

$$\hat{Y} = \frac{1}{q} \sum_{l=1}^{q} \hat{Y}_l = \frac{1}{q} \sum_{l=1}^{q} \hat{h}(X, S_n^{\Theta l}) \tag{3.27}$$

where $\hat{Y}_l$ is the output of $l - th$ tree and $l = \dots 1, 2, ..., q$.

The following figure shows the methodology for utilizing RF regression for prediction:

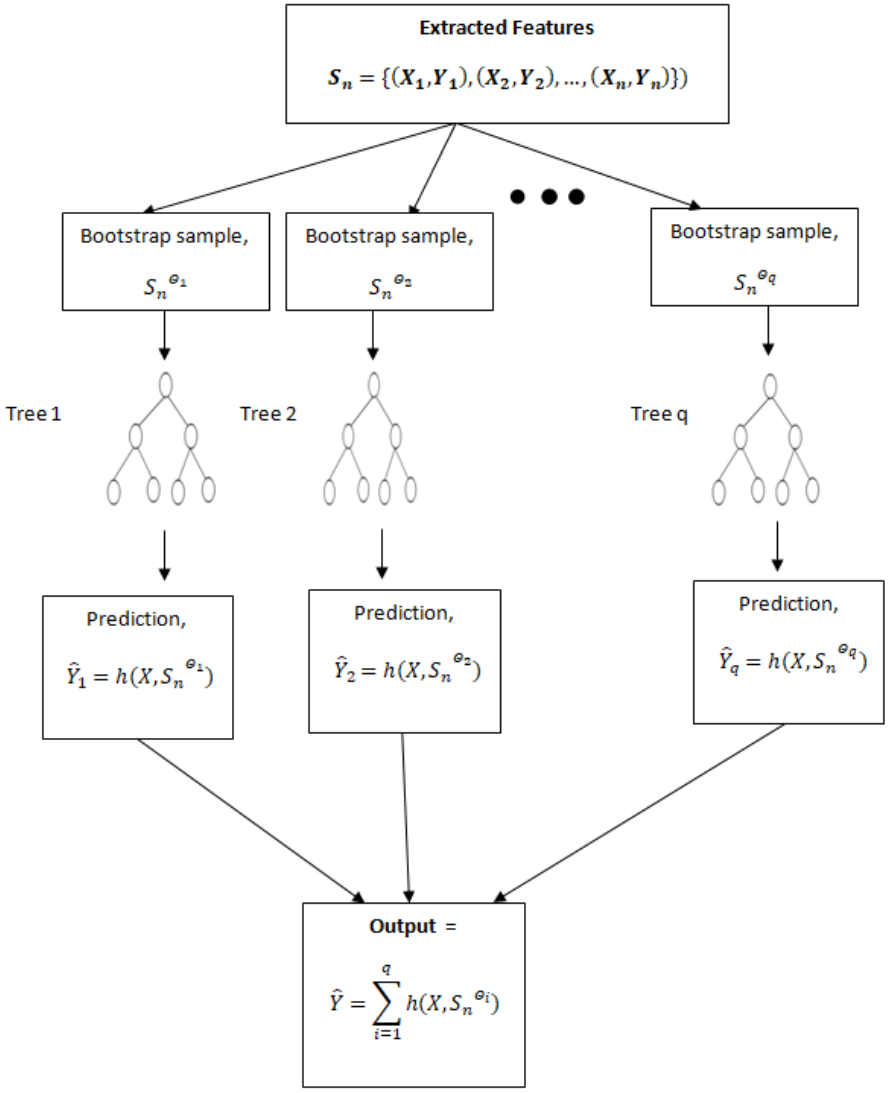

FIGURE 3.4: Random Forest for regression

3.18.4 Boosting

A machine learning ensemble approach called "boosting" [44] can turn poor learners into strong ones. The main idea behind boosting is to sequentially fit a group of weak learners. Therefore, boosting is a sequential process in which each new model attempts to correct the errors in the preceding model. In practice, it weights the data used to train the weak learners such that each successive learner pays greater weight to or only fits observations that the prior learners failed to accurately predict.

Simply put, the boosting technique prioritizes observations that were incorrectly predicted by

model n during the training of model $n + 1$, as opposed to the bagging procedure, which randomly selects training samples from the whole population. Each model is therefore dependent on the one before it. Additionally, with boosting, as opposed to bagging, where all learners' weights are identical, each learner might have a varied significance or weight in the final result.

Training process of a Boosting model

The training approach varies depending on the Boosting algorithm being used (such as Adaboost, XGBoost, or LightGBM), but in general, it goes like this:

1. All of the data samples are first given identical weights. On these samples, an individual model is fitted.
2. Prediction error is calculated for every sample.
3. To make those samples more significant for the fitting of the subsequent individual model, the weights of those samples with a larger error are raised.
4. The relevance or weight of any particular model is determined by how well it predicts the response. High weight or high significance in the final score will be given to a model with a very excellent predictive performance.
5. The posterior model's input is weighted data, and steps 2) and 3) are repeated.
6. Step four is repeated until either a set number of models have been trained or the error does not go over a predetermined level (threshold).

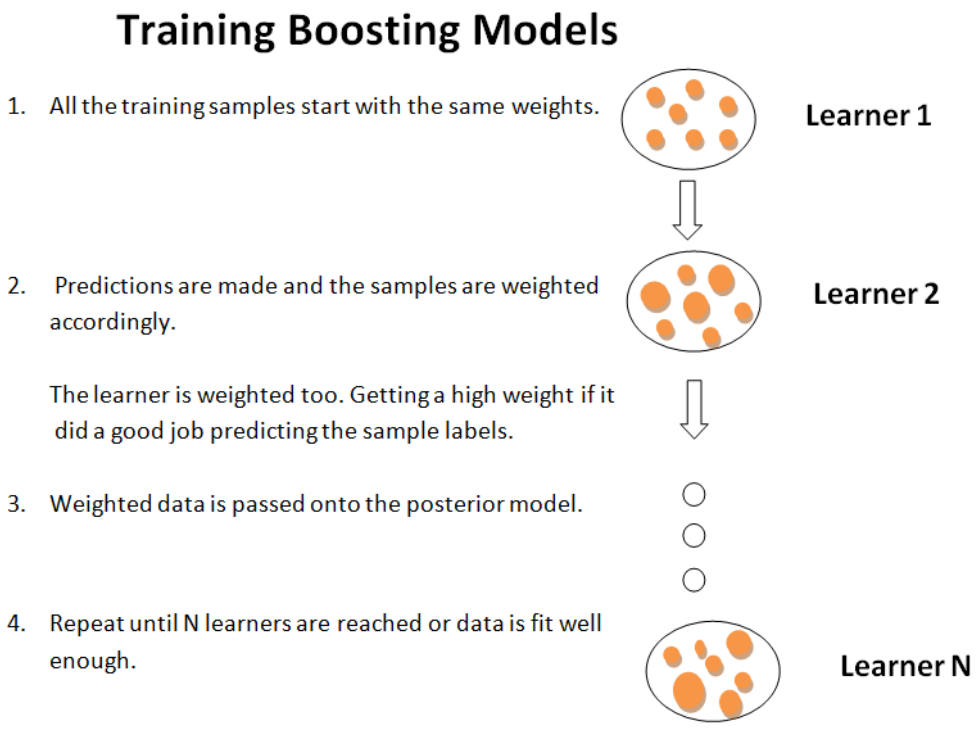

FIGURE 3.5: Boosting Model

3.18.4.1 A more mathematical approach to Boosting

Boosting techniques are characterized as sequential ensemble algorithms, as opposed to bagging, which is referred to as a parallel ensemble approach. In Boosting the weights c_k in $\hat{g}_{ens}(\cdot) = \sum_{k=1}^{M} c_k \hat{g}_k(\cdot)$ are dependent on the former estimated functions $\hat{g}_1, ..., \hat{g}_{k-1}$.

Breiman was the first to put the boosting method in a novel light, saying that it may be viewed as a nonparametric optimization procedure in function space. The use of boosting to problems other than classification, such as regression and survival analysis, is made possible by this approach.

3.18.4.2 Boosting as functional gradient descent

Boosting algorithms can be viewed as functional gradient descent methods rather than ensemble methods. The objective is to estimate a function $g : \mathbb{R}^d \to \mathbb{R}$ that minimizes an expected

loss

$$E[\rho(Y, g(X))], \quad \rho(\cdot, \cdot) : \mathbb{R} \times \mathbb{R} \to \mathbb{R}^+ \tag{3.28}$$

for some given data (X_i, Y_i), $(i = 1, ...n)$.

Here, we take into account both situations in which the univariate answer Y is continuous (regression problem) or discrete (classification problem), as boosting may be helpful in either situation.

As we'll see in the section below, boosting algorithms are seeking a "small" empirical risk,

$$n^{-1} \sum_{i=1}^{n} \rho(Y_i, g(X_i)) \tag{3.29}$$

by choosing a g in the linear hull of some function class, i.e. $g(\cdot) = \sum_k c_k g_k(\cdot)$ with $g_k(\cdot)$'s from a function class like trees.

The table below lists the most prevalent loss functions for binary classification and regression.

boosting	loss function	population minimizer for (3.28)
L_2Boost	$\rho(y, g) = (y - g)^2$	$g(x) = E[Y\|X = x]$
LogitBoost	$\rho(y, g) = log_2(1 + exp(-2(y - 1)g))$	$g(x) = 0.5logit(P[Y = 1\|X = x])$
Adaboost	$\rho(y, g) = exp(-2(y - 1)g)$	$g(x) = 0.5logit(P[Y = 1\|X = x])$

TABLE 3.1: The squared error, binomial negative log-likelihood and exponential loss functions and their population minimizers; $logit(p) = log(p/(1p))$.

While the exponential loss and the log-likelihood are exclusively utilized for binary classification, the squared error loss and the log-likelihood are mostly employed in regression.

3.18.4.3 The generic boosting algorithm

A constrained minimization of the empirical risk $n^{-1} \sum_{i=1}^{n} \rho(Y_i, g(X_i))$ is followed in order to estimate the function $g(\cdot)$, which minimizes an expected loss defined in the equation (3.28). By using a technique known as functional gradient descent to minimize the empirical risk, the constraint is introduced algorithmically and implicitly. In short, the solver of the minimization of the empirical risk is required to satisfy a "smoothness" constraint as regards a linear expansion of "simple" fits from a real-valued base procedure function estimate.

Generic functional gradient descent

1. **Initialization:** Given the data $\{(X_i, Y_i)i = 1, ..., n\}$, the basic approach is used to produce the function estimate:

$$\hat{F}_1(\cdot) = \hat{g}(\cdot),$$

where $\hat{g} = \hat{g}_{X,Y} = h_n((X_1, Y_1), ..., (X_n, Y_n))$ is a function of the initial data.
Set m=1.

2. **Projecting gradient to learner:** Calculate the negative gradient vector

$$U_i = -\frac{\partial \rho(Y_i, g)}{\partial g}|_{g=\hat{F}_m(X_i)}, \quad i = 1, 2, ..., n,$$

evaluated at the current $\hat{F}_m(\cdot)$. Afterwards, apply the basic technique on the gradient vector

$$\hat{g}_{m+1}(\cdot),$$

where $\hat{g}_{m+1} = \hat{g}_{X,U} = h_n((X_1, U_1), ..., (X_n, U_n))$ is a function of the original predictor variables and the current negative gradient vector as pseudoresponse.

3. (line search). Find the best step-size defined as follows using a one-dimensional numerical search.

$$\hat{s}_{m+1} = argmin_s \sum_{i=1}^{n} \rho(Y_i, \hat{F}_m(X_i) + s\hat{g}_{m+1}(X_i)).$$

Update,

$$\hat{F}_{m+1}(\cdot) = \hat{F}_m(\cdot) + \hat{s}_{m+1}\hat{g}_{m+1}(\cdot).$$

4. (iteration). Increase m by one and repeat Steps 2 and 3 until a stopping iteration M is achieved.

M is the boosting tuning parameter and is the number of algorithm replications. As the parameter value rises, the estimator becomes more convex.

3.18.5 Boosting models

3.18.5.1 AdaBoost for classification

AdaBoost [45] is one of the first implementations of boosting algorithm. It is a machine learning technique than can be applied to binary classification tasks. A decision stamp, or a decision

tree with one level, serves as the basic model in this boosting approach. AdaBoost 's objective is to create a classifier from a collection of weak decision stamp classifiers. For the purpose of training the following model, C_{m+1}, observations that the previous model C_m misclassified are given greater weights. The next model's goal is to fix the wrong predictions made by the prior model. The final prediction is a weighted combination of the predictions from each weak learner, with each weak learner's weight corresponding to the quality of its predictive performance on the training data.

3.18.5.2 AdaBoost classification algorithm

Let a binary classification problem with the observed data $\{X_i, Y_i\}_{i=1}^N$, where $Y_i \in \{0, 1\}$. The AdaBoost applies the following algorithm to formulate a model which accurately predicts Y:

Initialize the observation weights $w_i = \frac{1}{N}$, $i = 1, 2, .., N$.

For m=1 to M repeat steps (1)-(4):

1. Fit a classifier $C_m(x)$ to the training data using weights w_i.
2. Computed weighted error of newest classifier:

$$err_m = \frac{\sum_{i=1}^N w_i I(y_i \neq C_m(x_i))}{\sum_{i=1}^N w_i}$$

3. Compute $\alpha_m = log[(1 - err_m)/err_m]$
4. Update weights for $i = 1, ..., N$:
   $w_i \leftarrow w_i * exp[\alpha_m * I(y_i \neq C_m(X_i))]$
   and normalize to w_i to sum to 1.

   This scheme increases the weights of observations poorly predicted by C_m.

Output $C(x) = sign[\sum_{m=1}^M a_m C_m(x)]$

3.18.5.3 AdaBoost for regression

Given AdaBoost's success with classification problems, the method was expanded to include regression [46] issues using the AdaBoost.R technique [47], which turns the regression issue into a binary classification problem. Although, there are two drawbacks to this extension.

Firstly, the number of Boosting iterations is linearly increased when each example in the regression sample is expanded into multiple classification examples. Secondly, the iteration error function changes, and even this function varies among samples from the same iteration. Then, a number of approaches have been put out to convert the regression problem into a classification problem, with the AdaBoost.RT approach being emphasized owing to its performance and simplicity of implementation when compared to AdaBoost. R.

AdaBoost.RT Algorithm

Inputs: Sequence of m examples

$(x_1, y_1), (x_2, y_2), ..., (x_m, y_m)$ where $y \in \mathbf{R}$

Weak Learner Algorithm

Integer T (Number of algorithm iterations)

Threshold ϕ $(0 < \phi < 1)$ (Demarcate correct samples)

Initialize: $D_t(i) = \frac{1}{m} \forall i$ (Probability Distribution Function)

While $t \leq T$ do

Train the Weak learner, providing D_t;

Build the regression model: $f_t(x) \rightarrow y$

Compute the Absolute Relative Error (ARE) for each training sample

$$ARE(i) = \frac{f_t(x_i) - y_i}{y_i}$$

Compute the $f_t(x)$ error rate as follows:

$$\epsilon_t = \sum_{i:ARE_t(i)>\phi} D_t(i)$$

Set $\beta_t = \epsilon_t^n, \quad n = 1, 2, 3$ (Linear, square or cubic)

Update the distribution D_t

$$D_{t+1}(i) = \frac{D_t(i)}{Z_t} * \begin{cases} \beta_t \text{ if } ARE_t(i) \leq \phi \\ 1, otherwise \end{cases}$$

end

Output: Calculate the Strong Learner

$$SL(x) = \frac{\sum_{t=1}^{T} log(\frac{1}{\beta_t}) f_t(x)}{\sum_{t=1}^{T} log(\frac{1}{\beta_t})}$$

Gradient boosting machine

Let the function estimate problem in the classical supervised learning context. The objective of supervised learning, given a training dataset $(x_i, y_i)_{i=1}^N$, where the $x = (x_1, ..., x_N)$ are the explanatory input variables and y are the corresponding values of the response variable, is to minimize a certain loss function $\psi(y, f)$ while rebuilding the unknown functional dependence x $\xrightarrow{f}$ y using the approximation $\hat{f}(x)$, where:

$$\hat{f}(x) = y,$$
$$\hat{f}(x) = \underset{f(x)}{argmin}\psi(y, f(x))$$

An ensemble of weak prediction models, frequently tree-structured models, can be combined using the machine learning technique known as gradient boosting [48] to generate a prediction model. It results from combining Gradient Descent with Boosting and its objective is to minimize this loss function by adding weak learners using gradient descent.

The main distinction between boosting approaches and traditional machine-learning methods is that optimization takes place in the function space. In other words, the function estimate $\hat{f}$ is parameterized in the additive functional form:

$$\hat{f}(x) = \hat{f}^M(x) = \sum_{i=0}^{M} \hat{f}_i(x)$$

where M is the number of iterations, $\hat{f}_0$ is the initial guess and $\{\hat{f}_i\}_{i=1}^M$ are the function increments also known as boosts.

A similar method of parameterizing the family of functions may be used to make the functional approach practical in reality. In order to distinguish the parameterized "base-learner" functions from the overall ensemble function estimates $hatf(x)$, they will be denoted as $h(x, \theta)$. It is worth noting that there are different families of baselearners such as decision trees or splines.

The "greedy stagewise" approach of function incrementing with the base-learners can be now formulated. For this objective at each iteration, the optimal step size, ρ, should be specified. The following is the definition of the optimization rule for the function estimate at the $t - th$ iteration:

$$\hat{f}_t \leftarrow \hat{f}_{t-1} + \rho_t h(x, \theta_t)$$

with

$$(\rho_t, \theta_t) = \underset{\rho,\theta}{argmin} \sum_{i=1}^{N} \psi(y_i, \hat{f}_{t-1}) + \rho h(x_i, \theta)$$

Gradient boost algorithm

Both the loss function and the base learner models may be arbitrarily specified on request. It can be challenging to find the solution to the parameter estimates in practice, given a specific loss function $\psi(y, f)$ and/or a custom base-learner $h(x, \theta)$. In order to address this, it was suggested to select a new function $h(x, \theta_t)$ that is most parallel to the negative gradient $\{g_t(x_i)\}_{i=1}^N$ along the observed data:

$$g_t(x) = E_y[\frac{\partial \psi(y, f(x))}{\partial f(x)}|x]_{f(x)=\hat{f}^{t-1}(x)}$$

The new function increment that is most correlated with $-g_t(x)$ can be simply selected, as opposed to searching for the general solution for the boost increment in the function space. As a result, a potentially challenging optimization task can be replaced with the traditional least-squares minimization problem:

$$(\rho_t, \theta_t) = \underset{\rho,\theta}{argmin} \sum_{i=1}^{N} [-g_t(x_i) + \rho h(x_i, \theta)]^2$$

Finally, the complete form of the gradient boosting algorithm, as initially suggested by Friedman (2001) is given below. It is worth noting that the choices of $\psi(y, f)$ and $h(x, \theta)$ affect the exact form of the algorithm.

Gradient Boost Algorithm

Inputs:

- Input data $(x_i, y_i)_{i=1}^N$
- Number of iterations M
- Choice of the loss-function $\psi(y, f)$
- Choice of the base-learner model $h(x, \theta)$

Algorithm:

1. Intialize $\hat{f}_0$ with a constant
2. for t=1 to M do
3. Compute the negative gradient $g_t(x)$
4. Fit a new base-learner function $h(x, \theta_t)$

5. Find the best gradient descent step-size ρ_t:

$$\rho_t = argmin_\rho \sum_{i=1}^{N} \psi[y_i, \hat{f}_{t-1}(x_i) + \rho h(x_i, \theta_t)]$$

6. Update the function estimate:

$$\hat{f}_t \leftarrow \hat{f}_{t-1} + \rho_t h(x, \theta_t)$$

7. End for

Gradient boosting is a boosting-like algorithm used for regression and classification. When the target column is continuous, we use Gradient Boosting Regressor whereas when it is a classification problem, we use Gradient Boosting Classifier. The only difference between the two is the loss function. Generally, there are several distributions from which the response variable y may come. Naturally, this results in the definition of several loss functions. Regression, classification, and time-to-event analysis tasks are only a few of the response families for which specific boosting algorithms have been developed. We can organize the most popular loss-functions in the manners described below, depending on the family of the response variable y:

1. Continuous response, y $\in \mathbf{R}$:
    - Gaussian L_2 loss function
    - Laplace L_1 loss function
    - Huber loss function, δ specified
    - Quantile loss function, α specified
2. Categorical response, $y \in \{0, 1\}$:
    - Binomial loss function
    - Adaboost loss function

3.18.5.4 XGboost for regression

Generally, a boosting algorithm creates a weak learner at each step and incorporates it into the overall model. If the weak learner for each step is based on the gradient direction of the loss function, the boosting algorithm is called Gradient Boosting Machines (GBM). XGBoost algorithm [49] which stands for Extreme Gradient Boosting is a scalable machine learning system for tree boosting. It is composed of a sequence of decision trees using the gradient descent

technique to minimize the errors of weak estimators.

In comparison with other gradient boosting algorithms, XGBoost uses a more regularized model formalization, e.g. its objective functions consists of a training loss and a regularization term in order to control the over-fitting resulting in a model with a better performance.

Objective function

For a better understanding of this boosting algorithm, let the following data set:

$$D = (x_i, y_i) : i = 1, ..., n, \;\; x_i \in \mathbf{R}^m, \;\; y_i \in \mathbf{R}$$

composed of n observations with m features each and with a corresponding variable y.

Let's define $\hat{y}_i$ as the outcome of an ensemble which is defined in the following generalised model:

$$\hat{y}_i = \phi(x_i) = \sum_{k=1}^{K} f_k(x_i)$$

where f_k is a regression tree and $f_k(x_i)$ denotes the score assigned by the k-th tree to the i-th observation in data.

The regularized objective function that follows should be minimized in order to learn the functions f_k:

$$L(\phi) = \sum_{i=1}^{n} l(\hat{y}_i, y_i) + \sum_{k=1}^{K} \Omega(f_k),$$

where l denotes the loss function and Ω represents the regularization term.

The penalty term Ω is added in the model in the following way to reduce the model 's complexity and avoid over-fitting:

$$\Omega(f_k) = \gamma T + \frac{1}{2}\lambda ||\omega^2||$$

where

- γ is a parameter that controls penalty for the number of leaves T
- λ is a parameter that controls the magnitude of leaf weights w

Iterative method

The objective function is minimized by an iterative method. In this algorithm, every tree is trained once at a time. The objective function minimized in j-th iteration to learn the j-th tree, f_j, is the following one:

$$L^j = \sum_{i=1}^{n} l(y_i, \hat{y}_i^{(j-1)} + f_j(x_i)) + \Omega(f_j)$$

Using the Taylor expansion, this function may be made simpler. After the tree split from a particular node, a formula for loss reduction may be derived:

$$G = \frac{1}{2}[\frac{(\sum_{i \in I_L} g_i)^2}{\sum_{i \in I_L} h_i + \lambda} + \frac{(\sum_{i \in I_R} g_i)^2}{\sum_{i \in I_R} h_i + \lambda} + \frac{(\sum_{i \in I_I} g_i)^2}{\sum_{i \in I_I} h_i + \lambda}] - \gamma$$

where

- I denotes a subset of the available observations in the current node
- I_L is the subset of the available observations in the left nodes after the split
- I_R is the subset of the available observations in the right nodes after the split

The definitions of the functions g_i and h_i are given as follows:

- $g_i = \partial_{\hat{y}^{(j-1)}} l(y_i, \hat{y}_i^{(j-1)})$ is the first order gradient statistics on the loss function
- $h_i = \partial^2_{\hat{y}^{(j-1)}} l(y_i, \hat{y}_i^{(j-1)})$ is the second order gradient statistics on the loss function

The formula L_{split} can be used to determine the optimal split at any given node. The only two parameters on which the formula depends are the regularization parameter γ and the loss function. It is clear that any loss function that may provide the first and second-order gradients can be optimized using this algorithm.

Three factors explain why XGBoost outperforms other tree boosting techniques:

- The establishment of the regularised loss function
- A provided constant, η, can be used to scale down the weights of each new tree, reducing the effect of a single tree on the final score.
- Comparable to random forests in operation is column-sampling

3.18.5.5 LightGBM

In 2017 a novel Gradient Boosting Decision Tree (GBDT) algorithm LightGBM [50] was developed, which has been employed in a variety of data mining applications, including classification, regression, and ordering. Two innovative techniques are included in the LightGBM algorithm, namely gradient-based one-side sampling and exclusive feature bundling.

LightGBM aims to find an approximation $\hat{f}(x)$ to a particular function $f^*(x)$ that minimizes the expected value of a particular loss function $L(y, f(x))$ given the supervised training set $\{(x_i, y_i)\}_{i=1}^n$ as follows:

$$\hat{f} = \underset{f}{argmin} E_{y,X} L(y, f(x)) \tag{3.30}$$

To approximate the final model which is defined in the following equation, LightGBM integrates a number of T regression trees $\sum_{t=1}^{T} f_t(X)$.

$$f_T(X) = \sum_{t=1}^{T} f_t(X) \tag{3.31}$$

Regression trees can be written as $w_{q(x)}$, $q \in \{1, 2, \ldots, J\}$, where J is the number of leaves, q represents the tree's decision-making rules and w is a vector that represents the sample weight of leaf nodes. As a result, at step t, LightGBM would be trained in an additive way as follows:

$$\Gamma_t = \sum_{i=1}^{n} L(y_i, F_{t-1}(x_i) + f_t(x_i)) \tag{3.32}$$

Newton's approach is used in LightGBM to rapidly approximate the objective function. To make the formulation more simple, the constant term in the equation (2.31) can be removed in the following way:

$$\Gamma_t \cong \sum_{i=1}^{n} (g_i f_t(x_i) + \frac{1}{2} h_i f_t^2(x_i)) \tag{3.33}$$

Where

- g_i is the first order gradient statistic of the loss function
- h_i is the second order gradient statistic of the loss function

The equation (2.32) can be transformed in the following way:

$$\Gamma_t = \sum_{j=1}^{J} ((\sum_{i \in I_j} g_i) w_j + \frac{1}{2} ((\sum_{i \in I_j} h_i + \lambda) w_j^2))$$

Where I_j is the sample set of leaf j.

The optimal leaf weight scores of each leaf node w_j^* for a given tree structure $q(x)$ and the extreme value of Γ_k could be solved in the following way:

$$w_j^* = -\frac{\sum_{i \in I_j} g_i}{\sum_{i \in I_j} h_i + \lambda}$$

$$\Gamma_T^* = -\frac{1}{2} \sum_{j=1}^{J} \frac{(\sum_{i \in I_j} g_i)^2}{\sum_{i \in I_j} h_i + \lambda}$$

where Γ_T^* could be considered as the scoring function to gauge how well the tree structure q is constructed. At the end, after adding the split, the objective function is defined as follows:

$$G = \frac{1}{2}(\frac{(\sum_{i \in I_L} g_i)^2}{\sum_{i \in I_L} h_i + \lambda} + \frac{(\sum_{i \in I_R} g_i)^2}{\sum_{i \in I_R} h_i + \lambda} - \frac{(\sum_{i \in I_I} g_i)^2}{\sum_{i \in I_I} h_i + \lambda})$$

Where

- I_L is the sample set of the left branch
- I_R is the sample set of the right branch

LightGBM processes large-scale data and features more efficiently than other algorithms because in contrast to traditional GBDT-based techniques like XGBoost and GBDT growing trees horizontally, it grows the tree vertically.

3.18.5.6 CatBoost classifier

CatBoost [51, 52] stands for Categorical Boosting. It belongs to depth-wise gradient boosting algorithms and was released as an open source machine learning library by Yandex in 2017. One-hot encoding and label encoding are the two most used methods of handling categorical data in machine learning. This algorithm's ability to handle the various categories of data in the form as they come, without the need to pre-process them, is the reason for its name. Instead of the need of pre-process, the algorithm simply needs the specification of the set of categorical features by the usage of the parameter $cat\_featuresparameter$.

CatBoost presents two crucial algorithmic innovations: the implementation of ordered boosting, a permutation-driven alternative to the traditional approach, and a cutting-edge technique for processing categorical features. These advances are the primary components of CatBoost.

A modified version of the gradient boosting method is the ordered boosting approach. Prediction shifting was considered to be an issue with gradient boosting algorithms. A special kind of target leakage causes predictions shifting. In order to fight the prediction shift brought on by a special kind of target leakage found in all current implementations of gradient boosting algorithms, both methods are built in such a way that they employ random permutations of the training data.

Gradient boosted decision trees are used by Catboost. To develop a balanced tree, it makes use of oblivious decision trees. The oblivious trees are easier to fit and more CPU-efficient to implement than classic trees. For each level of the tree, the left and right splits are created using the same features. During the training, numerous decision trees will be constructed. Decision trees created in subsequent iteration will have a minimized loss than trees created in earlier iterations. In this manner, the iteration will go on until the loss function does not significantly decrease. The following figure shows how the Catboost algorithm 's working order.

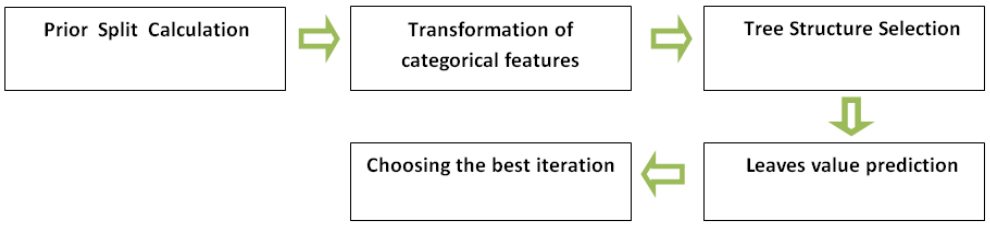

FIGURE 3.6: The workflow of a Catboost algorithm

Some information about the algorithm's steps for the building stages for a single tree is provided below:

Prior Split Calculation

Quantization will be used in the prior split calculation step to form the buckets from the data. To determine the potential ways to split data into buckets, quantization is carried out for each numerical feature. The tree structure is chosen using the information that is generated. The initial settings provide the quantization technique and the number of buckets. The algorithm 's starting parameters define the quantization method and the number of buckets.

Transformation of categorical features

Prior to choosing each split in the tree, categorical characteristics are converted to numerical ones by permuting the buckets at random and choosing the acceptable integer values for

the labels. Various statistics on combinations of categorical features and combinations of categorical and numerical features are used for this.

Tree structure selection

The greedy behaviour of the tree structure selection is such that a higher priority is given to the features having the ability to split the tree. For substitution in each leaf, features are chosen in order along with their splits. Candidates are chosen based on data obtained from the preliminary split computation and the conversion of categorical features to numerical features. The algorithm 's starting parameters include the tree depth as well as further rules for selecting the structure.

he following describes how a feature-split pair is selected for a leaf:

1. Candidate feature-split pairs that are possible be assigned to a leaf as the split are aggregated into a list.
2. If all of the candidates from step 1 have been assigned to the leaf, each object has a number of penalty functions computed for it.
3. Selected is the split that carries the lowest penalty.

The leaf is given the resultant value. For each following leaf, this process is repeated. It is worth noting that the number of leaves must correspond to the depth of the tree.

A random permutation of classification objects is carried out before the construction of each new tree. The structure of the following tree is chosen using a measure that indicates the direction for further improving the function.For each object the value is computed sequentially. The calculation uses the permutation created before the tree construction. In the order they were arranged before to the operation, the data for the objects are utilised.

3.19 Boosting vs Bagging

The main difference between Boosting and Bagging [53] is that in the latter the weak learners are fitted in parallel utilizing randomness, while in boosting the learners are trained in a sequential manner in arrange to be able to perform the data weighting or filtering described above.

Similarities between Bagging and Boosting

- Both ensemble approaches start with 1 learner and take N learners..
- Using random sampling, both generate various training data sets.
- In both cases, the final decision is made by either utilizing the majority vote of the learners in classification or taking the average of the N learners in regression.

Differences between Bagging and Boosting

Boosting [54]:

- A way of combining predictions that belong to the different types.
- Sequential
- Boosting aims to lessen bias.
- Increases variance (large ensemble can cause over-fitting)
- High dependency between ensemble elements
- Models are weighted based on how well they perform.
- Performance of previously constructed models has an impact on newly constructed models.
- Every new subset contains the elements that were poorly predicted by previous models.
- Apply boosting if the model is stable and simple (high bias).

Bagging:[54]

- The simplest method of merging forecasts of the same type.
- Parallel
- Bagging attempts to lower variance and address the overfitting issue.
- The bias does not change much
- Attempts to minimize correlation between ensemble elements.
- Every model is given equal weight..

- Each model is built independently.
- From the complete training dataset, several training data subsets are randomly selected with replacement.
- Apply bagging in case the model is unstable (high variance).

In the present thesis, the following Boosting algorithms: Gradient Boosting Machines (GBM), Extreme Gradient Boosting (XGBoost), Light Gradient Boosting Machines (LightGBM), Categorical Boosting (CatBoost) and AdaBoost are implemented and briefly presented. Also, the following Bagging algorithms: Random Forests, Extremely Randomized Trees ang Bagging are implemented and briefly presented.

3.20 Extra Trees Classifier

Extra Trees Classifier [55], also known as Extremely Randomized Trees Classifier, is a form of ensemble learning method. To provide its classification result, it combines the output from multiple de-correlated decision trees that have been gathered in a forest. Its only major conceptual difference from a Random Forest Classifier is how the decision trees in the forest are built. There are two main differences in the construction of the tree: bootstrap method is not used and rather than using the optimal splits, nodes are divided based on random splits among a randomly chosen subset of the features at each node. Therefore, randomness in Extra Trees originates from the random splitting of all observations, not from bootstrapping of the data.

In more detail, the original training sample is used to build each decision tree in the Extra Trees Forest. Afterwards, each decision tree is given a random sample of k features from the feature-set at each test node, from which it must choose the best feature to split the data according to certain mathematical criterion for example the Gini Index. There are several de-correlated decision trees produced as a result of this random sampling of features.

When the aforementioned forest structure is used, Gini index (if it is used to build the forest) is calculated for each feature to perform feature selection. The process of feature selection involves ranking each feature according to its Gini importance and selecting the top k features.

Extremely randomized trees have even more randomization in the computation of splits. A random subset of candidate features is employed, much as in random forests, but instead of searching for the thresholds that are the most discriminative, the best of these randomly produced thresholds is chosen as the splitting criteria. This frequently permits reduction in the variance of the model at the cost of a little increase in bias.

3.21 Stacking

Stacking is an ensemble learning technique [56] defined as the procedure that combines several estimators via a meta-model (meta-regressor or meta-classifier) aiming to reduce their biases. Predictions from each estimator (base-level model) are stacked together and the meta-model gets them as input to calculate the final prediction or otherwise the meta-model is fitted on the outputs of the base level model as features.

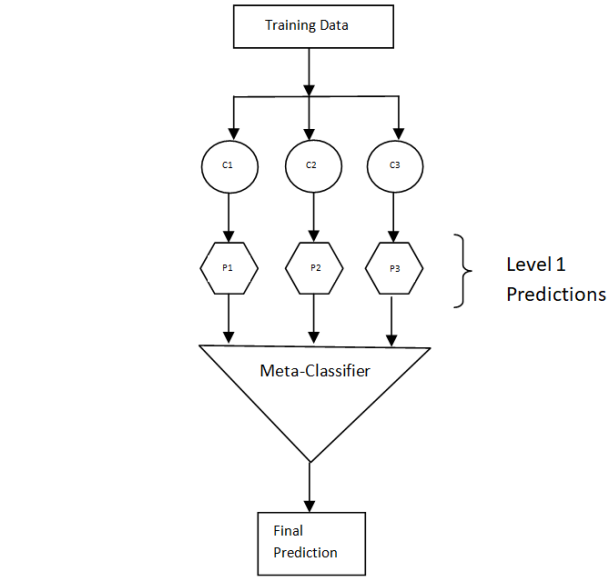

FIGURE 3.7: Stacking Model

The algorithm for Stacking follows the next steps:

1. Divide the data into a training and validation set,
2. Split the training set into K folds, for instance 10,
3. Fit a base model (e.g. Decision Tree) on 9 folds and make predictions on the 10-th fold,
4. Repeat until predictions are made for each fold,
5. Repeat step 3 – 6 for other base models (e.g. SVM),
6. Utilize predictions from the test set as features to the meta-model,
    - For regression: the values passed to the meta-model are numeric.
    - For classification: the values passed to the meta-model are probabilities or class labels.
7. The meta-model makes the final predictions on the test set.

Voting Classifier

Voting Classifier is an ensemble classifier that uses a number of classifiers to make predictions. In particular, multiple classifiers are trained, then the aggregation of their findings are passed into a single model called Voting classifier which predicts the output class based on combined majority of voting for each output class. Twod ifferent voting methods are supported by Voting Classifier:

- **Hard Voting:** The class with the highest majority of votes is the predicted output. In other words, the class which has the highest probability to be predicted by each of the classifiers is the predicted output.
- **Soft Voting:** For each class, the average of the probabilities given to this class by each classifier is calculated. The predicted output is the class with the highest probability averaged by each classifier.

3.22 Voting regression

Voting Regressor [57] is an ensemble meta-estimator that comprises several machine learning models and averages their individual predictions across all models to form a final prediction. This method is useful for a set of well performing models to compensate for their individual weaknesses in order to build a single model that can better generalize.

3.23 Theory for choosing hyperparmaters for the models

3.23.1 Hyperparameter tuning

Selecting the best set of hyperparameters for a learning algorithm is known as hyperparameter tuning [58]. A model argument known as a hyperparameter is one whose value is predetermined before learning process begins since it is unknown and cannot be learnt from the data. Although, the model's performance is impacted by the values given to these arguments. As a result, since the model's hyperparameters' optimum values are unknown, tuning them is crucial. Distinct hyperparameter settings result in different models that may perform differently in practice.

In case the model has several hyperparameters, the goal is to identify the ideal combination of hyperparameter values by scanning a space with multiple dimensions. Because of this, hyperparameter tuning may be an exceptionally complex and time-expensive task. There are two

simple techniques to tune the hyperparameters: Grid search and Random search. oth of these techniques include cross validation method which will be presented in more detail below. Also, the two aforementioned techniques will be described in further detail after the presentation of the cross-validation technique.

3.23.1.1 Cross validation

A statistical technique known as cross validation compares and evaluates learning algorithms by splitting data into two segments: one for learning or training a model and the other for model evaluation. There are three methods of cross validation:

- **K-fold method:** samples of equal sizes resulting from the shuffling of the original data set. One of the k samples is saved to be utilized as a testing sample for the model. The model is trained using the remaining samples. Then, with each of the remaining k - 1 samples, this process is repeated k times. The cross validation procedure has an error value for each run. The average error is then calculated from all the k runs. As a result, the variance is decreased, but this method's disadvantage is its running time.
- **Holdout method**: The sample for the holdout technique is divided into two segments, one for training and the other for testing. The training set 's size is typically bigger than the testing set 's one due to the size of this split. The model is fitted using the larger training sample and the model is tested using the test set. This cross validation technique is the simplest, because it is just a single cross validation. The holdout method has a faster running time than the K-fold method which is its advantage.
- **Leave one out method**: With a very large number for k, often equal to the size of the sample universe, this approach is similar to the k-fold method. With the exception of one, which is left out for testing, the model is trained on every value in the dataset. Although resource and time expensive, this method decreases the variance.

3.23.1.2 Grid search

Grid search is the simplest hyperparameter tuning approach. It is an exhaustive technique that examines all the candidate combinations of values for the hyperparameters. As a result, it actually finds the domain's optimum point. It is worth noting that checks every point in the grid by k-fold cross-validation, so it requires k training iterations. Therefore, it takes a lot of time to test every possible combination of the space, which is occasionally unavailable. As a result, its main disadvantage is its running time. Although, Grid search may be costly and complex, it is quite effective in finding the best combination of values of the hyperparameters.

In the proposed master thesis, every combination of hyperparameters used in the predictive models is selected by applying grid search which uses 10-cross validation.

3.24 Evaluation techniques used for the machine learning models

Evaluation of a machine learning model's performance on a test set

A common technique to evaluate the performance of a machine learning model on unseen data is to use a subset of data called test set. It is a separate subset of data used to test the model after completing the training. It is worth noting that the training set and the test set are mutually exclusive sets. In this master thesis, the data is apportioned into training and test sets, with an 80-20 split.

K cross validation for evaluating a machine learning model

A machine learning model's evaluation might be challenging. As previously mentioned, the data is randomly divided into a training set and a test set in order to assess the model's performance. The training set is used to train the model. By calculating various model performance metrics on the test set, the model's performance is assessed. The performance measurements obtained on one test set may be considerably different from the performance metrics derived on another test set, making this approach not very reliable. Additionally, in case a model is evaluated for different "hyperparameters", there is still a risk of overfitting on the test set, since the estimator's parameters can be tuned until the estimator performs optimally. In this manner, the test set's knowledge might "leak" into the model, and evaluation measures will no longer reflect generalization performance.

Another section of the data set can be also held out as a so-called "validation set" which will aid in dealing with the previous problems. Thus, the data set is split into three subsets: training set, validation set and test set. In particular, training set is used to train the model, and the test set is used for final evaluation when it seems that the experiment was successful.

Although, when the available data is divided into three sets, the size of the training set utilized to learn the model is significantly decreased and the results may depend on a particular random selection for the pair of train-set and validation-set. A method known as cross-validation (CV) has been suggested as a solution to this problem. The validation set is no longer necessary for performing the CV, but the test set still be held out for final evaluation. The k-fold CV applied on the training set is one common approach. First, this method divides the training set into k

folds or segments of equal size, making sure that each fold is utilized as a "validation" set at some point. k iterations of training and validation are then completed. In each $i-th$ iteration, where $i \in \{1, 2, ..., k\}$:

- A different fold of the data is firstly held-out
- The model is trained on the remaining k-1 folds
- The fitted model is validated on the held-out fold, i.e. it is used as a test set to calculate a performance metric.

After the completion of k-cross validation procedure, k values of a performance metric have been calculated. The average of these k values computed in the loop is the performance measure reported by k-fold cross-validation. Although this method could be computationally expensive, it doesn't waste as much data as the case when fixing an arbitrary validation set.

In summary, cross-validation is only utilized on training data while evaluating a model, and a test set is maintained separate for the final assessment. This method's goal is to determine whether the model that will be used can be generalized. Therefore, in this master thesis, each metric that evaluates the performance of the model was calculated in two ways. Firstly, it was computed using the 10-cross validation technique on the training set and secondly it was simply computed on the test set.

3.24.1 Binary classification

Classification matrix

A classification model's or "classifier's" performance on a test set where the real values that correspond to the model's predictions are known is frequently evaluated using a confusion matrix [59], which is a $N \times N$ table where N is the number of target classes.

For a binary classification problem, a 2×2 matrix is defined as follows with four count values:

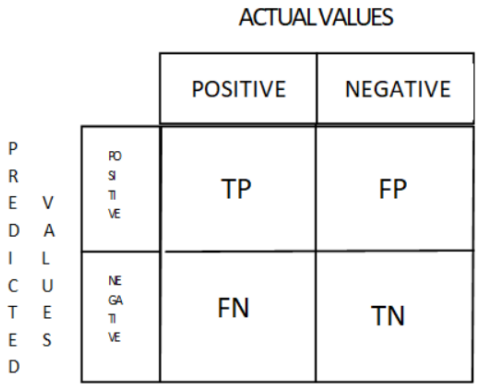

Figure 3.8: Confusion matrix for classification

Where

- There are two target classes: positive and negative
- The actual values of the target variable are represented by the columns
- The predicted values of the target variable by the ML model are represented by rows
- True Positive (TP):
  - The predicted value equals to the actual value
  - The predicted value is positive and the actual value is positive
- True Negative (TN):
  - The predicted value equals the actual value
  - The predicted value is negative and the actual value is negative
- False Positive (FP)
  - The predicted value is incorrectly predicted
  - The predicted value is positive and the actual value is negative
- False Negative (FN)
  - The predicted value is incorrectly predicted
  - The predicted value is negative and the actual value is positive

Accuracy

The accuracy metric [60] is the percentage of correct predictions to the total of all predictions and is defined as follows:

$$Accuracy = \frac{\text{Correct Predictions}}{\text{Total predictions}} = \frac{TP + TN}{TP + TN + FP + FT}$$

Sensitivity or Recall or True Positive Rate (TPR)

The percentage of data points that truly belong to the "positive" category and are accurately predicted as such in relation to all actual "positive" data points is known as Sensitivity, Recall, or True Positive Rate (TPR) [60]. It demonstrates how well the positive class was predicted. The recall metric, in other words, measures the model's capability to identify positive data points. As a result, it exclusively focuses on the predictions made for the positive data points. How the negative data points are classified has no bearing on this.

It is defined as follows:

$$sensitivity = \frac{TP}{TP + FN}$$

More positive data points are detected when recall is higher.

Specificity or True Negative rate (TNR)

The percentage of data points that actually belong to the negative class and are correctly predicted as negative when compared to all actual negative data points is known as Specificity or True Negative rate (TNR) [60]. It indicates how well the negative class is predicted.

It is defined as follows:

$$specificity = \frac{TN}{TN + FP}$$

More negative data points are detected when specificity is higher.

Precision

The precision [60] is the ratio of the number of data points that were correctly predicted as positive to the total number of the data points that were correctly or wrongly predicted as positive. In other words, the precision measures how well the model predicts a data point to be positive.

Precision is defined as follows:

$$precision = \frac{TP}{TP + FP}$$

F1-score

A harmonic mean of precision and recall is the F1-score. The F1-score [60] reflects the balance between the precision and the recall, which gives a combined idea of these two measurements. The range for F1-score is [0,1]. F1 increases as the classifier's predictive performance improves.

F1-score is defined as follows:

$$F1 - score = 2 * \frac{precision * recall}{precision + recall} = \frac{TP}{TP + \frac{1}{2}(FP + FN)}$$

AUC-ROC score

The name AUC [61] stands for "area under the curve". In this case, the curve is the Receiver Operating Characteristics (ROC) curve which is a coordinate schema analysis method. Sensitivity vs. 1-Specificity are plotted on this graph for various probabilities thresholds of model predictions.

In other words, at various classification thresholds, the ROC curve depicts the relationship between false positive rate and true positive rate. This single curve therefore represents a summary of the information included in the cumulative distribution functions of the scores of the two classes. As the choice of the classification threshold varies, the (ROC) curve may be considered as a full representation of the performance of the classifier. Therefore, a common assessment statistic for model performance is the AUC.

AUC 's range is the interval $[0, 1]$. The performance of the model improves as AUC value increases. An excellent model has an AUC that is very near to 1, which indicates that it has strong separability. Good predictive performance is often indicated by an AUC value > 0.8. An AUC near to 0 indicates a poor model, which has the worst separability metric.

In practice, this implies that the results are reversed, i.e., negative points are classified as positive ones and positive points are classified as negative ones. If the AUC is 0.5, the model is unable to distinguish between different classes or otherwise it does not have any class separation ability. The probability that the model would rank a randomly selected positive instance higher than a randomly selected negative instance is another way to interpret AUC.

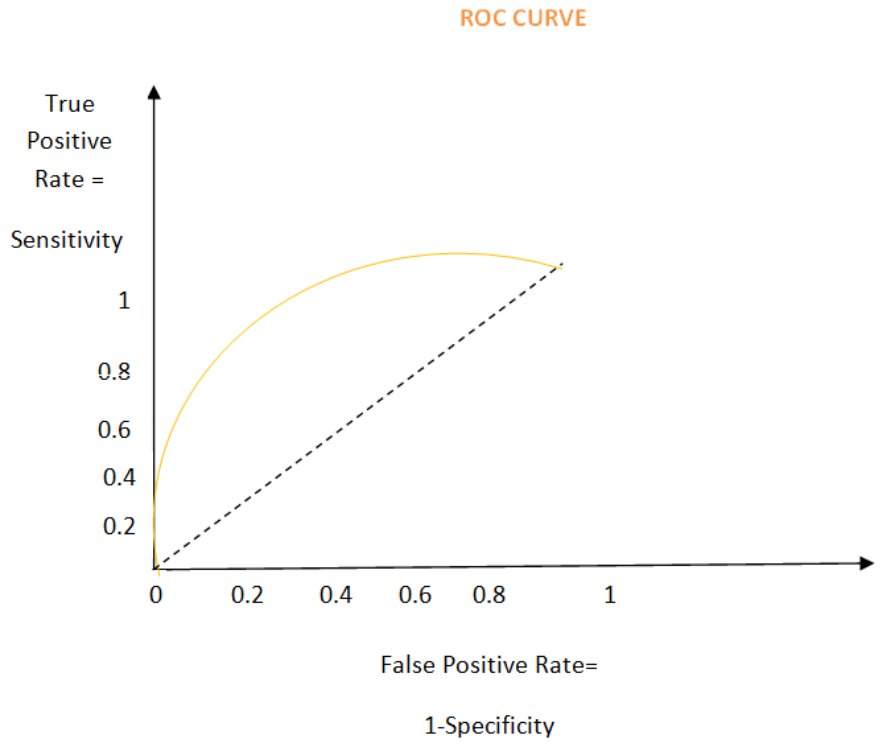

FIGURE 3.9: ROC curve

Balanced Accuracy

Balanced Accuracy is the arithmetic mean of Sensitivity(TPR) and Specificity (TNR). It is particularly useful for imbalanced data sets.

Balanced Accuracy is defined as follows:

$$\text{Balanced Accuracy} = \frac{Sensitivity + Specificity}{2}$$

3.24.2 Regression task

Mean Absolute Error (MAE)

The absolute differences between the actual and predicted values are averaged using then Mean Absolute Error (MAE). In MAE, each error is given the same weight. The prediction result is

more accurate the smaller the MAE value. The following equation provides its definition:

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|$$

where n is the number of observations, y_i and $\hat{y}_i$ are the actual and predicted values, respectively.

Mean Squarred Error (MSE)

A performance indicator that measures how well a model fits the target is the mean squared error, or MSE [62]. The average of all squared differences between the actual value and the predicted one defines the mean squared error as follows:

$$MSE = \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{n}$$

where n denotes the number of observations, y_i is the actual true value of the i-th data, $\hat{y}_i$ is the predicted value of the i-th data. A high MSE value indicates poor model performance, whereas an MSE of 0 indicates a flawless model that accurately predicts the target.

The average squared differences between the actual values and the predicted values is what is referred to as MSE. As a result of squaring the differences, the unit of MSE is different from the unit of the y-values, which makes MSE interpretation somewhat hard. Another measure of performance is the root mean squared error (RMSE), which is the square root of the MSE. RMSE in contrast to MSE takes on the same unit as that of the target values.

Root Mean-Square Error (RMSE)

It is common practice to measure the difference between values predicted by the model and the actual values using the Root Mean-Square Error (RMSE) [63]. It is fairly comparable to the MAE, but it penalizes higher absolute values by assigning them greater weight than the MAE. The difference between MAE and RMSE increases as the variation in the individual errors increases. Following is the definition of RMSE:

$$RMSE = \sqrt{\frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{n}}$$

The squared value of the prediction errors is used in RMSE. This aids in determining the influence of outliers. In regression, the RMSE is often employed as the loss function and a decreased RMSE value enhances the model's performance. The accuracy of predictions can be measured since the higher the error, the larger the RMSE. In other words, more precise predictions are

indicated by a prediction model with a lower RMSE value.

Goodness-of-fit (R^2)

Another indicator of how well a model's predicted values match actual values is the R^2 coefficient. A model's optimum R^2 value is 1, which means it can fully explain all of the variability of the target. It is defined in the following way:

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y}_i)^2}$$

where $\bar{y}$ stands for the average of the response variables. The MAE and RMSE values are almost zero when the predicted values are close to the actual values. Conversely, R^2 around 1 implies an excellent match between actual and predicted values.

*Mean Arctangent Absolute Percentage Error (MAAPE)

The normal MAPE [63] a severe drawback, it creates infinite or undefined values when the actual values are zero or very near to zero. Mean Arctangent Absolute Percentage Error (MAAPE) overcomes this drawback by transforming the normal MAPE using the inverse tangent function.

In practice, this transformation via arctangent function resolves two issues. Firstly, a more balanced penalty for small and large errors is retained. And secondly, when the actual values approach zero, the bounded range of the arctangent function assures that undefined or infinite errors may be avoided.

As a result, MAAPE is an indicator of prediction accuracy that enhances the quality of measurements of actual values that are zero or nearly zero. MAAPE is defined in the following way:

$$MAAPE = \frac{1}{n}\sum_{i=1}^{n} AAPE_i = \frac{1}{n}\sum_{i=1}^{n} arctan(|\frac{y_i - \hat{y}_i}{y_i}|)$$

It is worth noting that the unbounded range of MAPE $[0, \infty]$ has been turn into the bounded range $[0, \pi/2]$ for MAAPE. It is also noted that more accurate predictions are indicated by a prediction model with a lower MAAPE value.

Symmetric Mean Absolute Percentage Error (SMAPE)

Symmetric mean absolute percentage error (SMAPE) [64] is an accuracy metric based on relative or percentage errors. The absolute error is divided by the size of the actual value to form the relative error. Additionally, SMAPE is a relative error metric that is scale independent and fixes MAPE's disadvantages.

Although highly intuitive, MAPE degenerates into positive infinity the moment any of the actual values is zero. MAPE may readily explode towards infinity, even for very tiny actual values. When any y_i is 0, SMAPE does not fall apart, in contrast to the slightly simpler MAPE measure. SMAPE is defined in the following way:

$$\frac{100\%}{n} \sum_{i=1}^{n} \frac{|y_i - \hat{y}_i|}{(|y_i| + |\hat{y}_i|)/2}$$

SMAPE has a lower bound and an upper bound, unlike the MAPE metric. It may be used to compare prediction results across datasets because it is scale-independent. The SMAPE is consistently between 0% and 200%, with 0% denoting zero error. When SMAPE numbers are small enough, especially when under 30%, they can be interpreted much like MAPE values.

SMAPE has the drawback that the error value will be close to 100% if the actual value or predicted value equals 0. A forecast's accuracy increases with decreasing SMAPE value .

Summarization of the of the regreesion metrics

The table below summarises the useful information for each of the previously presented metric:

Metric	**Scale dependency**	**Formula**	**Value range**	**Aim to**
RMSE	Dependent	$\sqrt{\frac{\sum_{i=1}^{n}(y_i-\hat{y}_i)^2}{n}}$	$[0, \infty]$	Decrease
MAE	Dependent	$MAE = \frac{1}{n}\sum_{i=1}^{n}\|y_i - \hat{y}_i\|$	$[0, \infty]$	Decrease
SMAPE	Independent	$\frac{100\%}{n}\sum_{i=1}^{n}\frac{\|y_i-\hat{y}_i\|}{(\|y_i\|+\|\hat{y}_i\|)/2}$	[0,200]	Decrease
MAAPE	Independent	$\frac{1}{n}\sum_{i=1}^{n} arctan(\|\frac{y_i-\hat{y}_i}{y_i}\|)$	$[0, \pi/2]$	Decrease
MSE	Dependent	$MSE = \frac{\sum_{i=1}^{n}(y_i-\hat{y}_i)^2}{n}$	$[0, \infty]$	Decrease
R^2	Independent	$1 - \frac{\sum_{i=1}^{n}(y_i-\hat{y}_i)^2}{\sum_{i=1}^{n}(y_i-\bar{y}_i)^2}$	$[-\infty, 1]$	Increase

Chapter 4

Results

Chapter 4 first presents the results of the statistical analysis applied to each of the four regression and classification problems specified in Section 2.5 in order to select the predictors that are statistically significant for the prediction of the target variable. In particular, we focus on feature selection techniques to identify the most important predictors that are statistically significant in predicting the target variable of each problem. For each of the four problems, we start with a large set of candidate predictors and aim to identify the subset of predictors that have the strongest association with the target variable. To accomplish this, we employ appropriate statistical tests that evaluate the predictive power of each candidate predictor. These methods help us to determine the relevance and importance of each predictor in the context of the overall model. By applying feature selection techniques, we can reduce the number of predictors used in our model while maintaining a high level of predictive accuracy. This approach is particularly useful when dealing with high-dimensional datasets that contain a large number of potential predictors, as it helps to avoid overfitting and improve model interpretability. Overall, our goal in this section is to identify the most significant predictors that contribute to the prediction of the target variable in each of the four problems, providing insights that can be used to better understand and address this important mental health issue.

Chapter 4 also presents the results of the machine learning models that were developed and tested to predict the target variable for each of the following three problems: Problem: 1 D-from-M score, Problem 2: D-from-M class, Problem 3: M-from-D score. We utilized a range of machine learning algorithms, to develop and test our models. For each problem, we used a combination of data pre-processing, feature engineering, and model selection techniques to develop the most accurate and robust machine learning model possible. We also utilized cross-validation techniques to ensure that our models were not overfitting to the training data. The performance of each model was evaluated using a variety of metrics, such as mean squared error, mean absolute error, R-squared, accuracy, precision, recall, and F1 score. The results are

presented in a clear and concise manner, making it easy for the reader to understand the effectiveness of each model in predicting the target variable. By presenting these results, we aim to contribute to the growing body of literature on machine learning in various fields and provide insights into the effective use of these techniques for problem-solving.

4.1 Feature Selection

In this section, we present the results of the Feature selection for each of the four regression and classification problems specified in Section 2.5. More specifically, for each problem the results are organized and presented in four tables corresponding to the four different data types (i.e., ordinal, numerical, binary and nominal) of the features/predictors of the dataset. In each table, rows correspond to features of the dataset, the second column refers to the statistical test used to test if the feature of the row has a statistically significant effect at 5% significance level on the target variable of the problem, the third column contains the result of the test, i.e. the ✓symbol if the feature of the row has a statistically significant effect on the target variable, or the ✗ if there is no statistically significant effect. The numbers in the fourth column corresponds either to the effect size of the statistical test performed or to the correlation between the target variable and the feature of the corresponding row. It is worth noting that we will later use as predictors for each problem only the features with the ✓symbol.

4.1.1 Problem 1 (D-from-M score): Depression score through memory

This section presents the outcomes of the statistical analysis conducted on the depression score and various independent features or questionnaire attributes that were candidate predictors of the depression score. The results of these evaluations have been divided into four sections based on the types of independent variables analyzed. Notably, only the features that demonstrated statistical significance to the depression score, as indicated by the 3 symbol, were utilized as predictors for the prediction of the depression score

4.1.1.1 Ordinal features vs. Depression (ZUNG) score

Feature	Statistical test	statistically significant	Kendall Rank correlation
Which of the following best describes your education?	Spearman's rank-order test	✓	-0.09026
How good do you think your memory is? [Mnemonic Ability]	Spearman's rank-order test	✓	-0.25208
In the last two weeks what was your average mood? [Mood]	Spearman's rank-order test	✓	-0.51415
Do you decide to do something in the next few minutes and then forget to do it?	Spearman's rank-order test	✓	0.27637
Having trouble recognising a place you've visited before?	Spearman's rank-order test	✓	0.23804
Do you find it difficult to do something you intended to do even if it is in front of you, such as taking a pill or turning off the stove?	Spearman's rank-order test	✓	0.26063
Are you forgetting something you were told a few minutes ago?	Spearman's rank-order test	✓	0.27546
Do you forget appointments if someone doesn't remind you or if you haven't written them down in a calendar?	Spearman's rank-order test	✓	0.21854
Do you have trouble recognising a character on TV from one scene to the next?	Spearman's rank-order test	✓	0.18621
Forget to buy something you were planning to get, like a gift, even though you've seen the store?	Spearman's rank-order test	✓	0.26403
Do you find it difficult to recall things that have happened to you in the last few days?	Spearman's rank-order test	✓	0.28094
Do you repeat the same story to the same person in different circumstances?	Spearman's rank-order test	✓	0.24980
Do you forget to take something with you before you leave a room or before you go out even if it is right there in front of you?	Spearman's rank-order test	✓	0.28948
Do you lose things you just put somewhere, like a magazine or your glasses?	Spearman's rank-order test	✓	0.25645
Do you forget to mention something or give something to someone who asked you?	Spearman's rank-order test	✓	0.29283
Are you looking at something without realizing you saw it a few minutes ago?	Spearman's rank-order test	✓	0.31735
If you try to contact a friend or relative who is absent at the time, do you forget to try again later?	Spearman's rank-order test	✓	0.26598
Do you forget what you saw on TV the day before?	Spearman's rank-order test	✓	0.24387
Are you forgetting to mention something you ? intended to mention just a few minutes ago?	Spearman's rank-order test	✓	0.31349

Table 4.1: Results of the Feature selection on the ordinal features that are candidate to predict depression (ZUNG) score.

The above results from the statistical analysis reveals compelling evidence that each of the 19 candidate ordinal features, specifically designed to predict the Depression (ZUNG) score, demonstrates a noteworthy and statistically significant influence on the score. Notably, among these ordinal features, the comprehensive set of 16 questions from the (PRMQ) questionnaire is included. This observation holds significant implications, as it underscores the preliminary indication that the incorporation of memory-related questions within the predictive model has the potential to make a valuable contribution to accurately predicting the depression (Zung) score. This finding suggests that memory-related factors may play a substantial role in understanding and evaluating an individual's depressive tendencies as measured by the Zung scale. Further investigation and analysis are warranted to explore the precise nature and extent of the relationship between these memory-related questions and the prediction of depression score. More information about the statistical tests applied are given in Appendix A.1.2

4.1.1.2 Nominal features vs. Depression (ZUNG) score

Feature	Statistical test	Statistically significant	Effect size
Which statement best describes the quality of your sleep?	The Kruskal-Wallis H test	✓	0.14168
Where do you live?	The Kruskal-Wallis H test	✓	0.01023
Do you smoke?	The Kruskal-Wallis H test	✓	0.01111

TABLE 4.2: Results of the feature selection on the nominal features that are candidate to predict depression (ZUNG) score

The findings indicate that the three potential nominal features, designed to forecast the depression (ZUNG) score, exhibit a notable statistical impact on it. As a result, all three of these features will be incorporated into the machine learning model to estimate the depression (ZUNG) score. This choice is motivated by the belief that integrating these significant nominal features can improve the model's ability to predict the depression (ZUNG) score. By including these particular features, our objective is to leverage their influence and enhance the precision of the predictions produced by the machine learning model. More information about the statistical tests applied are given in Appendix A.1.4

4.1.1.3 Binary features vs. Depression (ZUNG) score

Feature	Statistical test	Statistically significant	Point Biserial
Do you think you have memory disorders that affect your daily life?	Welch's t-Test	✓	0.31146
Have you ever visited a memory clinic, psychologist or psychiatrist?	Welch's t-Test	✓	0.18863
Is there a family history of memory disorders (mother, father, grandfather, grandmother, brother or sister)?	Welch's t-Test	✗	0.02391
Currently, you have a known disease that affects memory (e.g., depression, Alzheimer's disease, mild cognitive impairment, Alzheimer's disease, Multiple Sclerosis, etc.)?	Welch's t-Test	✓	0.28281
Do you think you experience mood disorders (e.g. depression) that affect your daily life?	Welch's t-Test	✓	0.57886
Do you suffer from a medical condition?If so, which one? [I do not suffer from any pathological problem. I am perfectly healthy.]	Welch's t-Test	✓	-0.13568
Do you suffer from a medical condition? If so, which one? [Blood Pressure]	Welch's t-Test	✓	-0.0414
Do you suffer from a medical condition?If so, which one? [Chronic Atrial Fibrillation or Other Cardiac Problem]	Welch's t-Test	✗	0.01255
Do you suffer from a medical condition? If so, which one? [Stroke]	Welch's t-Test	✗	-0.00168
Do you suffer from a medical condition? If so, which one? [High Cholesterol]	Welch's t-Test	✗	-0.00904
Do you suffer from a medical condition? If so, which one? [Diagnosed Anxiety]	Welch's t-Test	✓	0.22535
Have you ever suffered a head injury that resulted in hospitalisation?	Welch's t-Test	✓	0.03718
Do you suffer from hypothyroidism?	Welch's t-Test	✓	0.05848
Do you suffer from Diabetes mellitus?	Welch's t-Test	✗	0.02601
What gender are you?	Welch's t-Test	✓	0.24985
Do you exercise more than 3 hours a week?	Welch's t-Test	✓	-0.20194
Have you been a confirmed case/infected with coronavirus (COVID-19) in the past?	Welch's t-Test	✗	-0.00944

Table 4.3: Results of the Feature selection on the binary features that are candidate to predict depression (ZUNG) score

From the above results, we can see that among the 17 candidate binary features to forecast the depression (ZUNG) score, an impressive 11 of them manifest a statistically significant influence in predicting the aforementioned score. It is worth noting, however, that certain binary features concerning a family history of memory disorders, chronic atrial fibrillation or cardiac problems, stroke, high cholesterol, diabetes mellitus, and confirmed cases of COVID-19 in the past do not demonstrate a statistically significant effect on the prediction process. This finding suggests that these particular binary features may not carry substantial predictive power in relation to the depression (ZUNG) score and, consequently, their inclusion in the model may not significantly contribute to accurately estimating the depression (ZUNG) score. More information about the statistical tests applied are given in Appendix A.1.3

4.1.1.4 Numeric features vs. Depression (ZUNG) score

Feature	Statistical test	Statistically significant	Pearson correlation coefficient (r)
How many glasses of alcohol do you consume per week? [Beer - 500ml glass of 5% alcohol]	Pearson correlation test	✗	0.66948
How many glasses of alcohol do you consume per week? [Cider (330ml) 4.5% alcohol]	Pearson correlation test	✓	0.000024
How many glasses of alcohol do you consume per week? [Wine - Medium glass of Chardonnay (175ml) 12.5% alcohol]	Pearson correlation test	✓	0.00008
How many glasses of alcohol do you consume per week? [Tsipouro, Ouzo or raki (40ml) 40% alcohol]	Pearson correlation test	✗	0.30333
How many glasses of alcohol do you consume per week? [White Drink (vodka, tequila, gin, etc.) (40ml) 40% alcohol]	Pearson correlation test	✓	0.00378
How many glasses of alcohol do you consume per week? [Dark Drink (rum, whisky etc.) (40ml) 40% alcohol]	Pearson correlation test	✗	0.05632
How many glasses of alcohol do you consume per week? [Sweet liqueur (40ml) 17% alcohol]	Pearson correlation test	✓	0.000001
Fill in your age.	Pearson correlation test	✓	$6.92547 * 10^{-37}$
Response time to question: Dem01	Pearson correlation test	✗	0.4264
Response time to question: Dem02	Pearson correlation test	✗	0.86086
Response time to question: Dem03	Pearson correlation test	✗	0.12006
Response time to question: Dem04	Pearson correlation test	✗	0.86356
Response time to question: Dem05	Pearson correlation test	✗	0.08223
Response time to question: Dem06	Pearson correlation test	✗	0.49328
Response time to question: Dem07	Pearson correlation test	✗	0.64739
Response time to question: Dem08	Pearson correlation test	✗	0.13909
Response time to question: Dem09	Pearson correlation test	✗	0.43733
Response time to question: Dem10	Pearson correlation test	✓	0.00262
Response time to question: Dem11	Pearson correlation test	✓	0.00784
Response time to question: Dem12	Pearson correlation test	✗	0.69373
Response time to question: Dem13	Pearson correlation test	✓	0.01364
Response time to question: Dem14	Pearson correlation test	✗	0.17091
Response time to question: Dem15	Pearson correlation test	✓	0.00006
Response time to question: Dem16	Pearson correlation test	✗	0.4732
Response time to question: Dem17	Pearson correlation test	✗	0.30811
Response time to question: Dem18	Pearson correlation test	✗	0.74661
Response time to question: SEM02	Pearson correlation test	✗	0.84552
Response time to question: SEM01	Pearson correlation test	✗	0.29125
Response time to question: Mem01	Pearson correlation test	✓	0.00599
Response time to question: Mem02	Pearson correlation test	✗	0.31806
Response time to question: Mem03	Pearson correlation test	✗	0.09978
Response time to question: Mem04	Pearson correlation test	✗	0.91829
Response time to question: Mem05	Pearson correlation test	✗	0.08709
Response time to question: Mem06	Pearson correlation test	✗	0.34058
Response time to question: Mem07	Pearson correlation test	✗	0.35084
Response time to question: Mem08	Pearson correlation test	✗	0.57598
Response time to question: Mem09	Pearson correlation test	✗	0.34009
Response time to question: Mem10	Pearson correlation test	✓	0.02817
Response time to question: Mem11	Pearson correlation test	✗	0.05928
Response time to question: Mem12	Pearson correlation test	✗	0.44828
Response time to question: Mem13	Pearson correlation test	✗	0.79531
Response time to question: Mem14	Pearson correlation test	✗	0.25996
Response time to question: Mem15	Pearson correlation test	✗	0.11301
Response time to question: Mem16	Pearson correlation test	✗	0.62853

Table 4.4: Results of the feature selection on the numeric features that are candidate to predict depression (ZUNG) score

Based on the results presented above, it is evident that out of the 44 candidate numerical features considered for predicting the depression (ZUNG) score, a total of 11 features demonstrate

a statistically significant effect on the score. Notably, among these 11 significant features, four of them pertain to the quantity of alcohol consumption per week, specifically focusing on cider, wine, white drink, and sweet liqueuer. Additionally, other significant features include age, response times to two questions belonging to PRMQ questionnaire: Mem01, Mem10 and response times to questions related to some pathological diseases, a history of head injury resulting in hospitalization, information regarding diabetes mellitus, and the quantity of some kind of alcohol consumption per week. The identified findings emphasize the notable importance of these specific features in effectively predicting the depression (ZUNG) score. Consequently, based on their statistical significance, we have made a deliberate decision to include these features as predictors in the machine learning model that aims to forecast the depression (ZUNG) score. By incorporating these influential features into the model, we aim to leverage their predictive power and enhance the accuracy of the predictions generated for the depression (ZUNG) score. This strategic choice underscores our commitment to utilizing the most relevant and statistically significant factors to optimize the performance of the machine learning model in predicting the depression (ZUNG) score. More information about the statistical tests applied are given in Appendix A.1.1

To sum up, we initially started with a total of 83 candidate input features to predict Depression (ZUNG) score. Among them, there were 17 binary features, 19 ordinal features (16 of them were the questions from the PRMQ questionnaire), 3 were nominal data and 44 were numeric features (36 of them were response time variables). After applying feature selection, we ended up with 44 input features to predict Depression (ZUNG) score. Among them, there are 11 binary features, 19 are ordinal features, 3 are nominal features and 11 are numeric features.

4.1.2 Problem 2 (D-from-M class): Depression class through memory

Within this section, we present the findings derived from the statistical analysis performed on the depression class variable and the independent features or questionnaire attributes that were evaluated as potential predictors of this variable. The outcomes of these assessments have been thoughtfully categorized into four sections, based on the distinct types of independent variables investigated. It is worth highlighting that only those particular features which exhibited a statistically significant association with the depression class variable, as signified by the 3 symbol, were ultimately employed as predictors in the prediction of the depression class variable.

4.1.2.1 Ordinal features vs. Depression (ZUNG) class

Feature	Statistical test	statistically significant	Effect size
Which of the following best describes your education?	Mann-Whitney U test	✓	0.07728
How good do you think your memory is? [Mnemonic Ability]	Mann-Whitney U test	✓	0.26572
In the last two weeks what was your average mood? [Mood]	Mann-Whitney U test	✓	0.54599
Do you decide to do something in the next few minutes and then forget to do it?	Mann-Whitney U test	✓	-0.26667
Having trouble recognising a place you've visited before?	Mann-Whitney U test	✓	-0.20636
Do you find it difficult to do something you intended to do even if it is in front of you, such as taking a pill or turning off the stove?	Mann-Whitney U test	✓	-0.23962
Are you forgetting something you were told a few minutes ago?	Mann-Whitney U test	✓	-0.28504
Do you forget appointments if someone doesn't remind you or if you haven't written them down in a calendar?	Mann-Whitney U test	✓	-0.21442
Do you have trouble recognising a character on TV from one scene to the next?	Mann-Whitney U test	✓	-0.14792
Forget to buy something you were planning to get, like a gift, even though you've seen the store?	Mann-Whitney U test	✓	-0.23745
Do you find it difficult to recall things that have happened to you in the last few days?	Mann-Whitney U test	✓	-0.27041
Do you repeat the same story to the same person in different circumstances?	Mann-Whitney U test	✓	-0.23944
Do you forget to take something with you before you leave a room or before you go out even if it is right there in front of you?	Mann-Whitney U test	✓	-0.27583
Do you lose things you just put somewhere, like a magazine or your glasses?	Mann-Whitney U test	✓	-0.24946
Do you forget to mention something or give something to someone who asked you?	Mann-Whitney U test	✓	-0.27621
Are you looking at something without realizing you saw it a few minutes ago?	Mann-Whitney U test	✓	-0.29716
If you try to contact a friend or relative who is absent at the time, do you forget to try again later?	Mann-Whitney U test	✓	-0.26033
Do you forget what you saw on TV the day before?	Mann-Whitney U test	✓	-0.23018
Are you forgetting to mention something you intended to mention just a few minutes ago?	Mann-Whitney U test	✓	-0.29548

Table 4.5: Results of the feature selection on the ordinal features that are candidate to predict depression (ZUNG) class

The statistical analysis results presented above provide compelling evidence that each of the 19 candidate ordinal features to predict the Depression (ZUNG) class variable, exerts a significant and noteworthy influence on this variable. Notably, among these ordinal features, the inclusive set of 16 questions derived from the (PRMQ) questionnaire is encompassed. This noteworthy observation holds substantial implications, as it highlights the initial indication that the inclusion of memory-related questions within the predictive model holds the potential to contribute significantly to accurately predicting the depression (Zung) class variable. This finding suggests that memory-related factors may play a substantial role in understanding and evaluating an individual's propensity towards depression as measured by the Zung scale. Further investigation and analysis are necessary to delve deeper into the precise nature and extent of the relationship between these memory-related questions and the prediction of

the depression class variable. More information about the statistical tests applied are given in Appendix A.3

4.1.2.2 Nominal features vs. Depression (ZUNG) class

Feature	Statistical test	Statistically significant	Cramer's V
Which statement best describes the quality of your sleep?	χ^2 test	✓	0.31191
Where do you live?	χ^2 test	✓	0.07445
Do you smoke?	χ^2 test	✓	0.10267

Table 4.6: Results of the feature selection on the nominal features that are candidate to predict depression (ZUNG) class

The observation of the above results is that the three potential nominal features intended to predict the depression (Zung) class variable demonstrate a statistically significant influence on it. Therefore, all three of these features will be integrated into the machine learning model to predict the depression (Zung) class variable. This decision is based on the understanding that incorporating these significant nominal features can enhance the model's predictive capabilities for the depression (Zung) class variable. By including these specific features, we aim to capitalize on their impact and optimize the accuracy of the predictions generated by the machine learning model. It is noteworthy that similar results were observed for predicting the corresponding numerical depression score. More information about the statistical tests applied are given in Appendix A.2.2

4.1.2.3 Binary features vs. Depression (ZUNG) class

Feature	Statistical test	Statistically significant	ϕ
Do you think you have memory disorders that affect your daily life?	Fisher's Exact Test	✓	0.25588
Have you ever visited a memory clinic, psychologist or psychiatrist?	Fisher's Exact Test	✓	0.15143
Is there a family history of memory disorders (mother, father, grandfather, grandmother, brother or sister)?	Fisher's Exact Test	✗	0.01292
Currently, you have a known disease that affects memory (e.g., depression, Alzheimer's disease, mild cognitive impairment, Alzheimer's disease, Multiple Sclerosis, etc.)?	Fisher's Exact Test	✓	0.22659
Do you think you experience mood disorders (e.g. depression) that affect your daily life?	Fisher's Exact Test	✓	0.51646
Do you suffer from a medical condition?If so, which one? [I do not suffer from any pathological problem. I am perfectly healthy.]	Fisher's Exact Test	✓	-0.09616
Do you suffer from a medical condition? If so, which one? [Blood Pressure]	Fisher's Exact Test Fisher's Exact Test	✓	-0.03982
Do you suffer from a medical condition?If so, which one? [Chronic Atrial Fibrillation or Other Cardiac Problem]	Fisher's Exact Test	✗	0.00926
Do you suffer from a medical condition? If so, which one? [Stroke]	Fisher's Exact Test	✗	-0.01638
Do you suffer from a medical condition? If so, which one? [High Cholesterol]	Fisher's Exact Test	✗	-0.00852
Do you suffer from a medical condition? If so, which one? [Diagnosed Anxiety]	Fisher's Exact Test	✓	0.17329
Have you ever suffered a head injury that resulted in hospitalisation?	Fisher's Exact Test	✗	0.01571
Do you suffer from hypothyroidism?	Fisher's Exact Test	✓	0.04469
Do you suffer from Diabetes mellitus?	Fisher's Exact Test	✗	0.02783
What gender are you?	Fisher's Exact Test	✓	0.20773
Do you exercise more than 3 hours a week?	Fisher's Exact Test	✓	-0.16015
Have you been a confirmed case/infected with coronavirus (COVID-19) in the past?	Fisher's Exact Test	✗	-0.01355

TABLE 4.7: Results of the feature selection on the binary features that are candidate to predict depression (ZUNG) class

The aforementioned results reveal that among the 17 candidate binary features specifically selected for predicting the depression (ZUNG) class variable, an impressive 11 of them exhibit a statistically significant influence on predicting the class variable. However, it is important to note that certain binary features related to family history of memory disorders, a head injury resulting in hospitalization, chronic atrial fibrillation or cardiac problems, stroke, high cholesterol, diabetes mellitus, and confirmed cases of COVID-19 in the past do not demonstrate a statistically significant effect on the prediction process.

Interestingly, it is worth mentioning that 10 out of the 11 features mentioned above, which lack statistical significance for the depression (ZUNG) class variable, are also not statistically significant on the corresponding depression numerical score, e.g. the depression (ZUNG) score,

except for the question related to a head injury resulting in hospitalization. This finding implies that these specific binary features may not possess substantial predictive power in relation to the depression (ZUNG) class variable, and therefore, their inclusion in the model may not significantly contribute to accurately estimating the depression (ZUNG) class variable. More information about the statistical tests applied are given in Appendix A.2.1

4.1.2.4 Numerical features vs. Depression (ZUNG) class

Feature	Statistical test	Statistically significant	Point Biserial
How many glasses of alcohol do you consume per week? [Beer - 500ml glass of 5% alcohol]	Welch's t-Test	✗	-0.00485
How many glasses of alcohol do you consume per week? [Cider (330ml) 4.5% alcohol]	Welch's t-Test	✓	0.04788
How many glasses of alcohol do you consume per week? [Wine - Medium glass of Chardonnay (175ml) 12.5% alcohol]	Welch's t-Test	✓	-0.05853
How many glasses of alcohol do you consume per week? [Tsipouro, Ouzo or raki (40ml) 40% alcohol]	Welch's t-Test	✗	-0.02299
How many glasses of alcohol do you consume per week? [White Drink (vodka, tequila, gin, etc.) (40ml) 40% alcohol]	Welch's t-Test	✓	0.03424
How many glasses of alcohol do you consume per week? [Dark Drink (rum, whisky etc.) (40ml) 40% alcohol]	Welch's t-Test	✗	0.0194
How many glasses of alcohol do you consume per week? [Sweet liqueur (40ml) 17% alcohol]	Welch's t-Test	✓	0.06525
Fill in your age.	Welch's t-Test	✓	-0.16439
Response time to question: Dem01	Welch's t-Test	✗	-0.00715
Response time to question: Dem02	Welch's t-Test	✗	0.01005
Response time to question: Dem03	Welch's t-Test	✗	-0.01761
Response time to question: Dem04	Welch's t-Test	✗	-0.01978
Response time to question: Dem05	Welch's t-Test	✓	-0.04869
Response time to question: Dem06	Welch's t-Test	✗	-0.01116
Response time to question: Dem07	Welch's t-Test	✗	0.00888
Response time to question: Dem08	Welch's t-Test	✗	0.00768
Response time to question: Dem09	Welch's t-Test	✗	-0.0052
Response time to question: Dem10	Welch's t-Test	✗	0.0322
Response time to question: Dem11	Welch's t-Test	✗	0.024
Response time to question: Dem12	Welch's t-Test	✗	-0.01486
Response time to question: Dem13	Welch's t-Test	✓	-0.04072
Response time to question: Dem14	Welch's t-Test	✗	-0.02197
Response time to question: Dem15	Welch's t-Test	✓	-0.07079
Response time to question: Dem16	Welch's t-Test	✗	0.00282
Response time to question: Dem17	Welch's t-Test	✓	-0.03799
Response time to question: Dem18	Welch's t-Test	✗	-0.00458
Response time to question: SEM02	Welch's t-Test	✗	-0.01017
Response time to question: SEM01	Welch's t-Test	✓	-0.03576
Response time to question: Mem01	Welch's t-Test	✓	-0.04025
Response time to question: Mem02	Welch's t-Test	✗	-0.0017
Response time to question: Mem03	Welch's t-Test	✓	-0.04029
Response time to question: Mem04	Welch's t-Test	✗	0.00302
Response time to question: Mem05	Welch's t-Test	✓	-0.03622
Response time to question: Mem06	Welch's t-Test	✗	0.00408
Response time to question: Mem07	Welch's t-Test	✗	0.00506
Response time to question: Mem08	Welch's t-Test	✗	0.00058
Response time to question: Mem09	Welch's t-Test	✗	-0.02969
Response time to question: Mem10	Welch's t-Test	✓	-0.04894
Response time to question: Mem11	Welch's t-Test	✗	-0.02389
Response time to question: Mem12	Welch's t-Test	✗	0.00781
Response time to question: Mem13	Welch's t-Test	✗	-0.00778
Response time to question: Mem14	Welch's t-Test	✗	-0.01634
Response time to question: Mem15	Welch's t-Test	✗	-0.0045
Response time to question: Mem16	Welch's t-Test	✗	0.00013

Table 4.8: Results of the feature selection on the numerical features that are candidate to predict Depression (ZUNG) class

Based on the aforementioned findings, it becomes evident that out of the 44 candidate numerical features evaluated for predicting the depression (ZUNG) class variable, a total of 14 features exhibit a statistically significant effect on the class variable. Among these significant features, four of them specifically pertain to the quantity of alcohol consumed per week, focusing on cider, wine, white drink, and sweet liqueuer. Additionally, other significant features include age, response times to four questions from the PRMQ questionnaire (Mem01, Mem03, Mem05, Mem10), response times to questions related to memory disorders affecting daily life, a history of head injury resulting in hospitalization, information regarding diabetes mellitus, sleep quality, average mood in the past two weeks, and the quantity of certain types of alcohol consumed per week.

These identified findings highlight the notable importance of these specific features in effectively predicting the depression (ZUNG) class variable. Consequently, based on their statistical significance, a conscious decision has been made to incorporate these features as predictors in the machine learning model aimed at forecasting the depression (ZUNG) class variable. By including these influential features in the model, we aim to leverage their predictive power and enhance the accuracy of the predictions generated for the Zung class variable. This strategic choice underscores our commitment to utilizing the most relevant and statistically significant factors to optimize the performance of the machine learning model in predicting the depression class variable. More information about the statistical tests applied are given in Appendix A.3.1

To sum up, we initially started with 83 candidate input features to predict Depression (ZUNG) class variable. Among them, there were 17 binary features, 19 ordinal features (16 of them were the questions from the PRMQ Memory questionnaire), 3 were nominal features and 44 were numeric features (36 of them were response time variables). After applying feature selection, we ended up with 46 input features to predict Depression (ZUNG) class. Among them, there are 10 binary features, 19 are ordinal features, 3 are nominal features and 14 are numeric features.

4.1.3 Problem 3 (M-from-D score): Memory score through depression

In this section, we provide an overview of the statistical analysis conducted between the memory (PRMQ) score and various independent features or questionnaire attributes that were considered as potential predictors of this score. The results obtained from these evaluations have been categorized into four sections based on the types of independent variables examined. It

is important to note that only the features demonstrating statistical significance to the memory (PRMQ) score, indicated by the 3 symbol, were used as predictors in the prediction of the memory (PRMQ) score.

4.1.3.1 Ordinal features vs. Memory (PRMQ) score

Feature	Statistical test	statistically significant	Kendall Rank correlation
Which of the following best describes your education?	Spearman's rank-order test	✓	-0.09678
How good do you think your memory is? [Mnemonic Ability]	Spearman's rank-order test	✓	-0.4394
In the last two weeks what was your average mood? [Mood]	Spearman's rank-order test	✓	-0.25476
I feel discouraged or sad.	Spearman's rank-order test	✓	0.25021
I cry easily or feel ready to cry	Spearman's rank-order test	✓	0.18013
I have trouble sleeping at night.	Spearman's rank-order test	✓	0.20277
I notice that I am losing weight	Spearman's rank-order test	✓	0.06550
I have constipation problems	Spearman's rank-order test	✓	0.11628
I have palpitations.	Spearman's rank-order test	✓	0.15646
I get tired for no particular reason	Spearman's rank-order test	✓	0.31003
I feel anxious and I can't calm down.	Spearman's rank-order test	✓	0.24636
I have more nervousness than before.	Spearman's rank-order test	✓	0.19964
I feel it would be better for others if I died.	Spearman's rank-order test	✓	0.20053
In the morning I feel better than at any time of the day.	Spearman's rank-order test	✓	0.08311
I eat the same amount of food as before.	Spearman's rank-order test	✓	0.168
I'm still interested in sex.	Spearman's rank-order test	✓	0.17312
My mind is as clear as before.	Spearman's rank-order test	✓	0.41230
It's easy for me to do the things I used to do before.	Spearman's rank-order test	✓	0.33673
I am optimistic about my future.	Spearman's rank-order test	✓	0.22291
I make decisions as easily as before.	Spearman's rank-order test	✓	0.31271
I feel useful and necessary.	Spearman's rank-order test	✓	0.25999
My life is quite full.	Spearman's rank-order test	✓	0.19059
I still enjoy the things I used to do.	Spearman's rank-order test	✓	0.28157

Table 4.9: Results of the feature selection on the ordinal features that are candidate to predict memory (PRMQ) score

The results obtained from the statistical analysis provide compelling evidence that each of the 23 candidate ordinal features to predict the memory (PRMQ) score, exerts a noteworthy and statistically significant influence on the score. Notably, among these ordinal features, the comprehensive set of 20 questions from the (Zung) depression questionnaire is included. This observation carries significant implications as it suggests an initial indication that incorporating depression-related questions within the predictive model can contribute significantly to accurately predicting the memory (PRMQ) score. This finding implies that depression-related factors may have a substantial impact on understanding and assessing an individual's memory-related issues as measured by the PRMQ scale. Further investigation and analysis are necessary to delve deeper into the precise nature and extent of the relationship between these depression-related questions and the prediction of the memory (PRMQ) score. More information about the statistical tests applied are given in Appendix A.1.2

4.1.3.2 Nominal features vs. memory (PRMQ) score

Feature	Statistical test	Statistically significant	Effect size
Which statement best describes the quality of your sleep?	The Kruskal-Wallis H test	✓	0.04131
Where do you live?	The Kruskal-Wallis H test	✓	0.00635
Do you smoke?	The Kruskal-Wallis H test	✓	0.00621

TABLE 4.10: Results of the Feature selection on the nominal features that are candidate to predict memory (PRMQ) score

Based on the findings above, it is evident that the three potential nominal features aimed at predicting the memory (PRMQ) score exert a statistically significant influence on it. As a result, all three of these features will be incorporated into the machine learning model to predict the memory (PRMQ) score. This decision is grounded in the understanding that integrating these significant nominal features can enhance the model's predictive capabilities for the memory (PRMQ) score. By including these specific features, our objective is to leverage their impact and maximize the accuracy of the predictions generated by the machine learning model. More information about the statistical tests applied are given in Appendix A.1.4

4.1.3.3 Binary features vs. Memory (PRMQ) score

Feature	Statistical test	Statistically significant	Point Biserial
Do you think you have memory disorders that affect your daily life?	Welch's t-Test	✓	0.50319
Have you ever visited a memory clinic, psychologist or psychiatrist?	Welch's t-Test	✓	0.10827
Is there a family history of memory disorders (mother, father, grandfather, grandmother, brother or sister)?	Welch's t-Test	✗	0.01601
Do you think you experience mood disorders (e.g. depression) that affect your daily life?	Welch's t-Test	✓	0.27229 0.27229
Do you suffer from a medical condition?If so, which one? [I do not suffer from any pathological problem. I am perfectly healthy.]	Welch's t-Test	✓	-0.05827
Do you suffer from a medical condition? If so, which one? [Blood Pressure]	Welch's t-Test Welch's t-Test	✗	-0.01748 -0.01748
Do you suffer from a medical condition?If so, which one? [Chronic Atrial Fibrillation or Other Cardiac Problem]	Welch's t-Test	✗	0.02859
Do you suffer from a medical condition? If so, which one? [Stroke]	Welch's t-Test	✓	0.0462
Do you suffer from a medical condition? If so, which one? [High Cholesterol]	Welch's t-Test	✗	-0.02547
Do you suffer from a medical condition? If so, which one? [Diagnosed Depression]	Welch's t-Test	✓	0.14316
Do you suffer from a medical condition? If so, which one? [Diagnosed Anxiety]	Welch's t-Test	✓	0.09416
Have you ever suffered a head injury that resulted in hospitalisation?	Welch's t-Test	✗	0.03319
Do you suffer from hypothyroidism?	Welch's t-Test	✗	0.00727
Do you suffer from Diabetes mellitus?	Welch's t-Test	✓	0.04409
What gender are you?	Welch's t-Test	✓	0.15137
Do you exercise more than 3 hours a week?	Welch's t-Test	✓	-0.10735
Have you been a confirmed case/infected with coronavirus (COVID-19) in the past?	Welch's t-Test	✗	0.01845

Table 4.11: Results of the feature selection on the binary features that are candidate to predict memory (PRMQ) score

Based on the results above, it is evident that among the 17 candidate binary features intended to forecast the memory (PRMQ) score, a remarkable 10 of them exhibit a statistically significant influence on predicting the score. However, it is important to note that certain binary features, such as family history of memory disorders, blood pressure, chronic atrial fibrillation or cardiac problems, high cholesterol, hypothyroidism, a head injury resulting in hospitalization, and confirmed cases of COVID-19 in the past, do not demonstrate a statistically significant effect on the prediction process. This finding suggests that these specific binary features may lack substantial predictive power in relation to the memory (PRMQ) score. Consequently, their inclusion in the model may not significantly contribute to accurately estimating the memory (PRMQ) score. More information about the statistical tests applied are given in Appendix A.1.3

4.1.3.4 Numeric features vs. Memory (PRMQ) score

Feature	Statistical test	Statistically significant	Pearson correlation coefficient (r)
How many glasses of alcohol do you consume per week? [Beer - 500ml glass of 5% alcohol]	Pearson correlation test	✓	0.00014
How many glasses of alcohol do you consume per week? [Cider (330ml) 4.5% alcohol]	Pearson correlation test	✓	0.00742
How many glasses of alcohol do you consume per week? [Wine - Medium glass of Chardonnay (175ml) 12.5% alcohol]	Pearson correlation test	✗	0.79497
How many glasses of alcohol do you consume per week? [Tsipouro, Ouzo or raki (40ml) 40% alcohol]	Pearson correlation test	✗	0.12893
How many glasses of alcohol do you consume per week? [White Drink (vodka, tequila, gin, etc.) (40ml) 40% alcohol]	Pearson correlation test	✓	0.00062
How many glasses of alcohol do you consume per week? [Dark Drink (rum, whisky etc.) (40ml) 40% alcohol]	Pearson correlation test	✓	0.03738
How many glasses of alcohol do you consume per week? [Sweet liqueur (40ml) 17% alcohol]	Pearson correlation test	✓	0.023035
Fill in your age.	Pearson correlation test	✓	0.000001
Response time to question: Dem01	Pearson correlation test	✗	0.13742
Response time to question: Dem02	Pearson correlation test	✗	0.08863
Response time to question: Dem03	Pearson correlation test	✗	0.55117
Response time to question: Dem04	Pearson correlation test	✗	0.32206
Response time to question: Dem05	Pearson correlation test	✗	0.07314
Response time to question: Dem06	Pearson correlation test	✗	0.41044
Response time to question: Dem07	Pearson correlation test	✗	0.14201
Response time to question: Dem09	Pearson correlation test	✗	0.32659
Response time to question: Dem10	Pearson correlation test	✓	0.00115
Response time to question: Dem11	Pearson correlation test	✓	0.003
Response time to question: Dem12	Pearson correlation test	✗	0.86254
Response time to question: Dem13	Pearson correlation test	✗	0.21532
Response time to question: Dem14	Pearson correlation test	✗	0.13478
Response time to question: Dem15	Pearson correlation test	✓	0.01669
Response time to question: Dem16	Pearson correlation test	✗	0.55146
Response time to question: Dem17	Pearson correlation test	✗	0.31763
Response time to question: Dem18	Pearson correlation test	✗	0.10507
Response time to question: SEM02	Pearson correlation test	✗	0.0564
Response time to question: SEM01	Pearson correlation test	✗	0.35285
Response time to question: Q1	Pearson correlation test	✗	0.49245
Response time to question: Q2	Pearson correlation test	✓	0.00029
Response time to question: Q3	Pearson correlation test	✗	0.826423
Response time to question: Q4	Pearson correlation test	✗	0.32475
Response time to question: Q5	Pearson correlation test	✗	0.90466
Response time to question: Q6	Pearson correlation test	✗	0.38294
Response time to question: Q7	Pearson correlation test	✗	0.52583
Response time to question: Q8	Pearson correlation test	✗	0.95166
Response time to question: Q9	Pearson correlation test	✗	0.30671
Response time to question: Q10	Pearson correlation test	✗	0.17766
Response time to question: Q11	Pearson correlation test	✓	0.00026
Response time to question: Q12	Pearson correlation test	✗	0.07442
Response time to question: Q13	Pearson correlation test	✗	0.79835
Response time to question: Q14	Pearson correlation test	✗	0.68669
Response time to question: Q15	Pearson correlation test	✗	0.87902
Response time to question: Q16	Pearson correlation test	✗	0.99184
Response time to question: Q17	Pearson correlation test	✗	0.31473
Response time to question: Q18	Pearson correlation test	✗	0.42116
Response time to question: Q19	Pearson correlation test	✗	0.09864
Response time to question: Q20	Pearson correlation test	✗	0.20743

Table 4.12: Results of the Feature selection on the numeric features that are candidate to predict memory (PRMQ) score

According to the presented results, it is clear that among the 47 numerical features considered for predicting the memory score (PRMQ), 11 features have a significant impact on the score. Notably, out of these 11 features, 5 of them are related to the amount of alcohol consumed per week, specifically focusing on beer, cider, white drink, dark drink, and sweet liqueur. Other significant features include age, response times to two questions from the Zung depression questionnaire: Q2 (I feel better in the morning than at any time of the day) and Q11 (My mind is as clear as before), response times to questions about certain diseases, a history of hospitalization due to head injury, and the quantity of some type of alcohol consumed per week. These findings highlight the importance of these specific features in accurately predicting the memory score (PRMQ). Therefore, based on their statistical significance, we have decided to include these features as predictors in our machine learning model that aims to forecast the memory score (PRMQ). By incorporating these influential features into the model, we intend to utilize their predictive power and improve the accuracy of the predictions for the memory score (PRMQ). This strategic decision emphasizes our dedication to using the most relevant and statistically significant factors to optimize the performance of the machine learning model in predicting the memory score (PRMQ). More information about the statistical tests applied are given in Appendix A.1.1

To sum up, we initially started with 90 candidate input features to predict Memory (PRMQ) score. Among them, there were 17 binary features, 23 ordinal features (20 of them were the questions from the depression questionnaire), 3 were nominal data and 47 were numeric features (39 of them were response time variables). After applying Feature selection, we endep up with 47 input features (these are the features that have statistically significant effect at 5% significance level on the Memory (PRMQ) score) to predict Memory (PRMQ) score. Among them, there are 10 binary features, 23 are ordinal features, 3 are nominal features and 11 are numeric features.

4.1.4 Problem 4 (M-from-D class): Memory class through depression

This section presents a summary of the statistical analysis performed to assess the relationship between the memory (PRMQ) class variable and different independent features or questionnaire attributes that were considered as potential predictors to the class variable. The obtained results have been divided into four sections based on the types of independent variables investigated. It is crucial to mention that only the features showing statistical significance to the memory (PRMQ) class variable, indicated by the 3 symbol, were utilized as predictors in predicting the memory (PRMQ) class variable.

4.1.4.1 Ordinal features vs. Memory (PRMQ) class

Feature	Statistical test	statistically significant	Effect size
Which of the following best describes your education?	Mann-Whitney U test	✓	0.07285
How good do you think your memory is? [Mnemonic Ability]	Mann-Whitney U test	✓	0.24976
In the last two weeks what was your average mood? [Mood]	Mann-Whitney U test	✓	0.04163
I feel discouraged or sad.	Mann-Whitney U test	✓	0.09366
I cry easily or feel ready to cry	Mann-Whitney U test	✓	-0.15609
I have trouble sleeping at night.	Mann-Whitney U test	✓	0.06244
I notice that I am losing weight	Mann-Whitney U test	✓	0.18732
I have constipation problems	Mann-Whitney U test	✓	-0.27057
I have palpitations.	Mann-Whitney U test	✓	-0.27057
I get tired for no particular reason	Mann-Whitney U test	✓	-0.14569
I feel anxious and I can't calm down.	Mann-Whitney U test	✓	0.05203
I have more nervousness than before.	Mann-Whitney U test	✓	0.12488
I feel it would be better for others if I died.	Mann-Whitney U test	✓	0
In the morning I feel better than at any time of the day.	Mann-Whitney U test	✓	-0.21854
I eat the same amount of food as before.	Mann-Whitney U test	✓	0.04163
I'm still interested in sex.	Mann-Whitney U test	✓	-0.27057
My mind is as clear as before	Mann-Whitney U test	✓	-0.1665
It's easy for me to do the things I used to do before.	Mann-Whitney U test	✓	0.07285
I am optimistic about my future.	Mann-Whitney U test	✓	0.02081
I make decisions as easily as before.	Mann-Whitney U test	✓	-0.09366
I feel useful and necessary.	Mann-Whitney U test	✓	-0.07285
My life is quite full.	Mann-Whitney U test	✓	0.12488
I still enjoy the things I used to do.	Mann-Whitney U test	✓	0.03122

Table 4.13: Results of the Feature selection on the ordinal features that are candidate to predict Memory (PRMQ) class

From the above results, it is worth mentioning that all the 23 candidate ordinal features for predicting the memory (PRMQ) class variable has a statistically significant effect on the class variable. Notably, among these ordinal features, the comprehensive set of 20 questions from the Zung depression questionnaire is included. This observation is particularly important as it suggests that incorporating questions related to depression in the predictive model can greatly contribute to accurately predicting the memory (PRMQ) class variable. This finding suggests that depression-related factors may play a significant role in understanding and assessing an individual's memory-related issues as measured by the PRMQ scale. However, further investigation and analysis are needed to gain a deeper understanding of the precise nature and extent of the relationship between these depression-related questions and the prediction of the memory (PRMQ) class variable. It is worth mentioning that similar observations were noticed for the corresponding numerical depression (Zung) score, thus the same ordinal predictors were used to predict the depression (Zung) score as well as with the depression (Zung) class variable. More information about the statistical tests applied are given in Appendix A.3

4.1.4.2 Nominal features vs. Memory (PRMQ) class

Feature	Statistical test	Statistically significant	Cramer's V
Which statement best describes the quality of your sleep?	χ^2 test	✓	0.08762
Where do you live?	χ^2 test	✗	0.05199
Do you smoke?	χ^2 test	✓	0.04845

Table 4.14: Results of the feature selection on the nominal features that are candidate to predict memory (PRMQ) class

Based on the aforementioned findings, it is observed that two out of the three potential nominal features, excluding the question "Where do you live?", intended to forecast the memory (PRMQ) class variable, have a statistically significant influence on it. Consequently, these two features will be integrated into the machine learning model to predict the memory (PRMQ) class variable. This decision is grounded in the understanding that incorporating these significant nominal features can enhance the model's ability to predict the memory (PRMQ) class variable. By including these specific features, our goal is to leverage their impact and optimize the accuracy of the predictions generated by the machine learning model. It is important to note that for predicting the corresponding numerical depression score, we utilized all three candidate nominal features. More information about the statistical tests applied are given in Appendix A.2.2

4.1.4.3 Binary features vs. Memory (PRMQ) class

Feature	Statistical test	Statistically significant	ϕ
Do you think you have memory disorders that affect your daily life?	Fisher's Exact Test	✓	0.18649
Have you ever visited a memory clinic, psychologist or psychiatrist?	Fisher's Exact Test	✓	0.05619
Is there a family history of memory disorders (mother, father, grandfather, grandmother, brother or sister)?	Fisher's Exact Test	✗	-0.02373
Do you think you experience mood disorders (e.g. depression) that affect your daily life?	Fisher's Exact Test	✓	0.13208
Do you suffer from a medical condition?If so, which one? [I do not suffer from any pathological problem. I am perfectly healthy.]	Fisher's Exact Test	✗	-0.0138
Do you suffer from a medical condition? If so, which one? [Blood Pressure]	Fisher's Exact Test	✗	-0.01275
Do you suffer from a medical condition?If so, which one? [Chronic Atrial Fibrillation or Other Cardiac Problem]	Fisher's Exact Test	✗	0.00846
Do you suffer from a medical condition? If so, which one? [Stroke]	Fisher's Exact Test	✗	0.02929
Do you suffer from a medical condition? If so, which one? [High Cholesterol]	Fisher's Exact Test	✗	-0.02681
Do you suffer from a medical condition? If so, which one? [Diagnosed Depression]	Fisher's Exact Test	✓	0.1176
Do you suffer from a medical condition? If so, which one? [Diagnosed Anxiety]	Fisher's Exact Test	✓	0.07164
Have you ever suffered a head injury that resulted in hospitalisation?	Fisher's Exact Test	✗	-0.00574
Do you suffer from hypothyroidism?	Fisher's Exact Test	✗	-0.01489
Do you suffer from Diabetes mellitus?	Fisher's Exact Test	✗	0.03511
What gender are you?	Fisher's Exact Test	✓	0.08482
Do you exercise more than 3 hours a week?	Fisher's Exact Test	✓	-0.05125
Have you been a confirmed case/infected with coronavirus (COVID-19) in the past?	Fisher's Exact Test	✗	0.0287

TABLE 4.15: Results of the feature selection on the binary features that are candidate to predict memory (PRMQ) class

The above-mentioned findings indicate that out of the 17 chosen binary features candidate for predicting the memory (PRMQ) class variable, 7 of them demonstrate a statistically significant impact on predicting the class variable. These 7 binary features encompass questions regarding the following factors: exercising more than 3 hours a week, gender, diagnosed anxiety, diagnosed depression, visit to a memory clinic, psychologist or psychiatrist, personal assessment of memory problems affecting daily life, and personal assessment of mood disorders (such as depression) affecting daily life. This discovery suggests that these particular binary features possess considerable predictive strength in relation to the memory (PRMQ) class variable. Therefore, incorporating them into the model is likely to significantly contribute to accurately estimating the memory (PRMQ) class variable. More information about the statistical tests applied are given in Appendix A.2.1.

4.1.4.4 Numeric features vs. Memory (PRMQ) class

Feature	Statistical test	Statistically significant	Point Biserial
How many glasses of alcohol do you consume per week? [Beer - 500ml glass of 5% alcohol]	Welch's t-Test	✗	0.02517
How many glasses of alcohol do you consume per week? [Cider (330ml) 4.5% alcohol]	Welch's t-Test	✓	0.03559
How many glasses of alcohol do you consume per week? [Wine - Medium glass of Chardonnay (175ml) 12.5% alcohol]	Welch's t-Test	✗	-0.0175
How many glasses of alcohol do you consume per week? [Tsipouro, Ouzo or raki (40ml) 40% alcohol]	Welch's t-Test	✗	0.00149
How many glasses of alcohol do you consume per week? [White Drink (vodka, tequila, gin, etc.) (40ml) 40% alcohol]	Welch's t-Test	✗	0.01913
How many glasses of alcohol do you consume per week? [Dark Drink (rum, whisky etc.) (40ml) 40% alcohol]	Welch's t-Test	✗	0.01372
How many glasses of alcohol do you consume per week? [Sweet liqueur (40ml) 17% alcohol]	Welch's t-Test	✗	0.01321
Fill in your age.	Welch's t-Test	✓	-0.04489
Response time to question: Dem01	Welch's t-Test	✗	0.01992
Response time to question: Dem02	Welch's t-Test	✗	-0.00091
Response time to question: Dem03	Welch's t-Test	✗	0.00973
Response time to question: Dem04	Welch's t-Test	✗	0.00159
Response time to question: Dem05	Welch's t-Test	✗	-0.01379
Response time to question: Dem06	Welch's t-Test	✗	-0.00503
Response time to question: Dem07	Welch's t-Test	✗	-0.00669
Response time to question: Dem09	Welch's t-Test	✗	-0.01408
Response time to question: Dem10	Welch's t-Test	✗	0.0065
Response time to question: Dem11	Welch's t-Test	✗	0.02244
Response time to question: Dem12	Welch's t-Test	✗	-0.01111
Response time to question: Dem13	Welch's t-Test	✗	-0.00254
Response time to question: Dem14	Welch's t-Test	✗	-0.01057
Response time to question: Dem15	Welch's t-Test	✗	-0.0203
Response time to question: Dem16	Welch's t-Test	✗	-0.01864
Response time to question: Dem17	Welch's t-Test	✗	-0.01032
Response time to question: Dem18	Welch's t-Test	✗	-0.00434
Response time to question: SEM02	Welch's t-Test	✗	0.00457
Response time to question: SEM01	Welch's t-Test	✗	-0.02264
Response time to question: Q1	Welch's t-Test	✗	-0.01393
Response time to question: Q2	Welch's t-Test	✓	-0.04295
Response time to question: Q3	Welch's t-Test	✗	-0.00991
Response time to question: Q4	Welch's t-Test	✗	-0.00804
Response time to question: Q5	Welch's t-Test	✗	-0.01755
Response time to question: Q6	Welch's t-Test	✗	-0.01917
Response time to question: Q7	Welch's t-Test	✗	-0.01038
Response time to question: Q8	Welch's t-Test	✗	-0.00349
Response time to question: Q9	Welch's t-Test	✗	-0.01144
Response time to question: Q10	Welch's t-Test	✗	-0.01353
Response time to question: Q11	Welch's t-Test	✓	-0.05948
Response time to question: Q12	Welch's t-Test	✗	-0.01865
Response time to question: Q13	Welch's t-Test	✗	-0.01299
Response time to question: Q14	Welch's t-Test	✗	-0.00967
Response time to question: Q15	Welch's t-Test	✗	-0.0205
Response time to question: Q16	Welch's t-Test	✗	-0.02177
Response time to question: Q17	Welch's t-Test	✗	-0.00749
Response time to question: Q18	Welch's t-Test	✗	0.01059
Response time to question: Q19	Welch's t-Test	✗	-0.00206
Response time to question: Q20	Welch's t-Test	✗	-0.01552

Table 4.16: Results of the feature selection on the numeric features that are candidate to predict memory (PRMQ) class

Based on the obtained results, it is evident that among the 47 numerical features examined for predicting the memory (PRMQ) class variable, four features have a significant effect on the class variable. Notably, one of these four features is associated with the weekly consumption of cider. The other three statistically significant features encompass age, as well as response times to two questions from the Zung depression questionnaire: Q2 (I feel better in the morning than at any time of the day) and Q11 (My mind is as clear as before). It is worth noting that these two response time variables also exhibited statistical significance in relation to the corresponding memory score.

These findings underscore the importance of these specific features in accurately predicting the memory (PRMQ) class variable. Consequently, based on their statistical significance, we have made the decision to incorporate these features as predictors in our machine learning model, which aims to forecast the memory (PRMQ) class variable. By integrating these influential features into the model, our intention is to leverage their predictive power and enhance the accuracy of the predictions for the memory (PRMQ) class variable. This strategic choice emphasizes our commitment to utilizing the most pertinent and statistically significant factors in order to optimize the performance of the machine learning model in predicting the memory (PRMQ) class variable. More information about the statistical tests applied are given in A.3.1

To sum up, we initially started with 90 candidate input features to predict Memory (PRMQ) class. Among them, there were 17 binary features, 23 ordinal features (20 of them were the questions from the depression questionnaire), 3 were nominal data and 47 were numeric features (39 of them were response time variables. After applying Feature selection, we ended up with 36 input features to predict PRMQ class variable. Among them, there are 7 binary features, 23 are ordinal features, 2 are nominal features and 4 are numeric features.

4.2 Results from the Machine Learning models

In this section, we present an overview of the performance measures derived from an evaluation of various machine learning models for addressing Problem 1: D-from-M score, Problem 2: D-from-M class, and Problem 3:M-from-D score. We analyzed the effectiveness of these models to assess their suitability for solving the specific problems at hand. Moreover, we scrutinized the performance metrics of each model. By considering the performance metrics, we were able to identify the top-performing models that exhibited exceptional performance across all evaluated measures. We believe that this examination of performance measures and the selection of the best models will offer valuable insights for researchers, practitioners, and decision-makers in choosing the most suitable machine learning approaches for addressing similar problems in various domains.

4.2.1 Results for the Problem 2 (D-from-M class)

After the training of each classification model designed to predict depression (ZUNG) class problem 2: D-from-M class from Section 2.5.2, we illustrate the performance measure of each classification model calculated via 10-cross validation technique as well as calculated on the test set, itself, in the following tables.

Model	Recall (CV)	Recall (test)	Accuracy (CV)	Accuracy (test)	Precision (CV)	Precision (test)
LightGBM classifier	0.77605	0.79125	0.80576	0.79341	0.78468	0.75563
Gradient Boosting classifier	0.78249	0.78451	0.81062	0.79042	0.78937	0.75405
Bagging classifier	0.7686	0.77778	0.80277	0.79042	0.78319	0.75738
Voting Classifier (2)	0.782241	0.78788	0.80651	0.79042	0.78299	0.75241
CatBoost classifier	0.75626	0.76094	0.79753	0.78892	0.78207	0.76351
AdaBoost classifier	0.79019	0.77778	0.81139	0.78743	0.78622	0.75244
XGBoost classifier	0.77901	0.78115	0.80914	0.78743	0.78859	0.75081
Logistic Regression	0.78496	0.78115	0.80989	0.78743	0.78719	0.75081
Voting Classifier (1)	0.71689	0.72727	0.79604	0.78593	0.80278	0.77698
LDA classifier	0.78845	0.80135	0.80801	0.78593	0.78116	0.73913
Ridge classifier	0.79	0.80135	0.80837	0.78593	0.78068	0.73913
Voting Classifier (3)	0.71689	0.72727	0.79604	0.78593	0.80278	0.77698
Voting Classifier (4)	0.78876	0.80135	0.80426	0.78593	0.7743	0.73913
Random Forest classifier	0.78291	0.79125	0.80501	0.78443	0.77893	0.74133
Voting Classifier (0)	0.78031	0.78451	0.80688	0.78293	0.78371	0.74204
Voting Classifier (5)	0.78817	0.79125	0.80651	0.77994	0.77879	0.73438
Stacking Classifier (1)	0.81109	0.78115	0.81101	0.77994	0.77371	0.73885
SVM Classifier	0.78828	0.79125	0.80204	0.77695	0.77101	0.72981
Extra Trees classifier	0.78223	0.78788	0.79154	0.77246	0.75631	0.72446
Decision tree classifier	0.6731	0.64983	0.76421	0.76048	0.77669	0.7751
KNN classifier	0.72689	0.72727	0.78293	0.75449	0.77289	0.72241
Stacking Classifier (2)	0.74652	0.75758	0.7687	0.75299	0.73674	0.70755
GaussianNB classifier	0.75711	0.78115	0.75972	0.73653	0.71873	0.67639

Table 4.17: Performance measures calculated via 10-cross validation as well as on the test set for each ML model used to predict Depression (ZUNG) class.

Model	F1 (CV)	F1 (test)	AUC-ROC (CV)	AUC-ROC (test)	Balanced accuracy (CV)	Balanced accuracy (test)
LightGBM classifier	0.77949	0.77303	0.80308	0.79319	0.80308	0.79319
Gradient Boosting classifier	0.78506	0.76898	0.80796	0.78983	0.80796	0.78983
Bagging classifier	0.77496	0.76744	0.80003	0.78916	0.80003	0.78916
Voting Classifier (2)	0.78168	0.76974	0.80459	0.79017	0.80459	0.79017
CatBoost classifier	0.76798	0.76223	0.79379	0.78613	0.79379	0.78613
AdaBoost classifier	0.78739	0.7649	0.80966	0.78646	0.80966	0.78646
XGBoost classifier	0.78288	0.76568	0.80638	0.78679	0.80638	0.78679
Logistic Regression	0.78515	0.76568	0.80806	0.78679	0.80806	0.78679
Voting Classifier (1)	0.75658	0.7513	0.78857	0.78008	0.78857	0.78008
LDA classifier	0.78378	0.76898	0.80655	0.78747	0.80655	0.78747
Ridge classifier	0.78442	0.76898	0.80697	0.78747	0.806970	0.78747
Voting classifier (3)	0.75658	0.7513	0.78857	0.78008	0.78857	0.78008
Voting Classifier (4)	0.78065	0.76898	0.80285	0.78747	0.80285	0.78747
Random Forest classifier	0.77993	0.76547	0.80296	0.78511	0.80296	0.78511
Voting Classifier (0)	0.78117	0.76268	0.80449	0.78309	0.80449	0.78309
Voting Classifier (5)	0.78249	0.76175	0.80494	0.78107	0.80494	0.78107
Stacking Classifier (1)	0.79129	0.75941	0.81143	0.78006	0.81143	0.78006
SVM classifier	0.77855	0.75929	0.80081	0.77837	0.80081	0.77837
Extra Trees classifier	0.76809	0.75484	0.79083	0.77399	0.79083	0.77399
Decision tree classifier	0.71525	0.70696	0.75704	0.74944	0.75704	0.74944
KNN classifier	0.74835	0.72483	0.77747	0.75178	0.77747	0.75178
Stacking Classifier (2)	0.74092	0.73171	0.76633	0.75345	0.76633	0.75345
GaussianNB classifier	0.73647	0.725	0.75984	0.74098	0.75984	0.74098

TABLE 4.18: Performance measures calculated via 10-cross validation as well as on the test set for each ML model used to predict Depression (ZUNG) class

Based on the provided Table 4.17 of performance measures for different ML models used to predict the Depression (ZUNG) class, here are some notices:

1. Recall (CV) represents the ability of the model to correctly identify positive cases during cross-validation. The Stacking Classifier (1) has the highest recall value in cross-validation with 0.81109, indicating its effectiveness in capturing positive instances. It is followed by the AdaBoost classifier and Ridge classifier with recall values of 0.79019 and 0.79, respectively.

2. Recall (test) measures the model's ability to correctly identify positive cases on the test set. The Ridge classifier, LDA classifier and Voting classifier (4) have the highest recall value on the test set with 0.80135.

3. Accuracy (CV) represents the overall correctness of the model's predictions during cross-validation. The Stacking Classifier (1) achieves the highest accuracy in cross-validation with a value of 0.81101. Other models with relatively high accuracy in cross-validation include the AdaBoost classifier, XGBoost classifier, and Logistic Regression.

4. Accuracy (test) measures the overall correctness of the model's predictions on the test set. The LightGBM classifier exhibits the highest accuracy on the test set with a value of 0.79341. Other models with notable accuracy on the test set include the Gradient Boosting classifier, Bagging classifier, and Voting classifier (2).

5. Precision (CV) indicates the model's ability to correctly identify positive cases out of the predicted positive instances during cross-validation. The Voting classifier (1) and Voting classifier (3) achieve the highest precision value in cross-validation with 0.80278, followed by the Gradient Boosting classifier and XGBoost classifier.

6. Precision (test) measures the model's ability to correctly identify positive cases out of the predicted positive instances on the test set. The Voting classifier (1) and Voting classifier (3) demonstrates the highest precision value on the test set with 0.77698, followed by the CatBoost classifier and Bagging classifier.

These notices highlight the performance of different ML models in predicting the Depression (ZUNG) class based on the provided performance measures. It's important to consider the specific requirements and priorities of the project while selecting a suitable model.

Also, we notice that

7. Overfitting: Overfitting occurs when a model performs well on the training data but fails to generalize to unseen data. One way to identify overfitting is by comparing the performance measures between cross-validation and the test set. If there is a significant drop in performance on the test set compared to cross-validation, it suggests overfitting. In the provided table, models like the Decision tree classifier and GaussianNB classifier show a noticeable drop in performance on the test set, indicating potential overfitting.

8. Underfitting: Underfitting happens when a model is too simple and fails to capture the underlying patterns in the data. This can be identified by low performance measures on both cross-validation and the test set. However, none of the models in the provided table exhibit clear signs of underfitting based on the performance measures provided.

9. Cross-Validation Metrics: Cross-validation is a technique used to assess the model's performance by splitting the data into multiple subsets for training and evaluation. The performance measures calculated via cross-validation (CV) provide insights into the model's average performance across different subsets. Models like the Stacking Classifier (1), AdaBoost classifier, and Ridge classifier demonstrate relatively high cross-validation recall, accuracy, and precision values, indicating their effectiveness in capturing patterns in the data.

10. Test Set Metrics: Test set metrics represent the model's performance on unseen data, which is crucial for evaluating its generalization capability. Models like the LightGBM classifier, Gradient Boosting classifier, and CatBoost classifier exhibit high recall, accuracy, and precision values on the test set, suggesting their ability to generalize well to new instances.

Based on the provided Table 4.18 of performance measures for different ML models used to predict the Depression (ZUNG) class, here are some notices:

11. F1 Score (CV and Test):
    - The model with the highest F1 score in cross-validation is the Stacking Classifier (1) with a score of 0.79129.
    - The model with the highest F1 score on the test set is the LightGBM classifier with a score of 0.77303.
    - The Stacking Classifier (1) performs well in cross-validation, while the LightGBM classifier performs well on the test set.
12. AUC-ROC Score (CV and Test):
    - The model with the highest AUC-ROC score in cross-validation is the Stacking Classifier (1) with a score of 0.81143.
    - The model with the highest AUC-ROC score on the test set is the LightGBM classifier with a score of 0.79319.
    - The Stacking Classifier (1) has the highest AUC-ROC score in cross-validation, while the LightGBM classifier performs well on the test set.
13. Balanced Accuracy (CV and Test):
    - The Stacking Classifier (1) has the highest Balanced Accuracy with a score 0.81143 in cross-validation.
    - The LightGBM classifier has the highest Balanced Accuracy on the test with a score of 0.79319.
    - The Stacking Classifier (1) has the highest Balanced Accuracy score in cross-validation, while the LightGBM classifier performs well on the test set.
14. The Stacking Classifier (1) stands out with the highest scores in terms of F1 score, AUC-ROC, and balanced accuracy in cross-validation. However, the LightGBM classifier also perform well on the test set. It's important to note that the model's performance on the test set should be considered for evaluating its generalization capabilities.

Based on the table 4.18, we can make the following observations regarding overfitting and underfitting:

15. Overfitting:

- The LightGBM classifier, Gradient Boosting classifier, and AdaBoost classifier show relatively small differences between CV and test set metrics across different performance measures, indicating less overfitting.
- On the other hand, the Decision tree classifier and KNN classifier demonstrate significant drops in performance on the test set compared to the CV metrics, suggesting potential overfitting.

16. Underfitting:

    - The Decision tree classifier and KNN classifier exhibit lower performance on both CV and test set metrics, indicating potential underfitting.

17. Overall, models such as LightGBM classifier, Gradient Boosting classifier, and AdaBoost classifier appear to generalize well, while the Decision tree classifier and KNN classifier may suffer from overfitting or underfitting. It's important to note that these conclusions are based solely on the provided performance measures, and additional analysis considering the dataset, model complexity, and available data size would provide a more comprehensive evaluation of overfitting and underfitting.

From these observations, we see that the LightGBM classifier has the best performance in 4 of the 12 calculated metrics and the Adaboost classifier has also the best performance in 4 of the 12 metrics. So we propose either the LightGBM classifier or the AdaBoost classifier to predict depression (ZUNG) class. The hyperparameters for these two models are given in Tables 4.17 and 4.18.

Model	**Hyperparameters**
LightGBM classifier	reg_alpha=7.11 reg_lambda=0.009 num_leaves=141 min_child_samples=37 lambda_l1=9.83 feature_fraction=0.69 bagging_freq=2 max_depth=14 learning_rate=0.005 colsample_bytree=0.1 n_estimators=1182

Table 4.19: The hyperparameters of the LightGBM model used to predict depression (ZUNG) class.

Model	Hyperparameters
AdaBoost classifier	n_estimators=100 learning_rate=0.1

TABLE 4.20: The hyperparameters of the AdaBoost classifier to predict depression (ZUNG) class.

To ensure optimal performance and fine-tuned configuration of the tested machine learning models, we employed the grid search method to select the most suitable hyperparameters. Grid search is a systematic approach that exhaustively explores different combinations of hyperparameters within predefined ranges. For each machine learning model utilized in solving the problem, we carefully designed a specific grid of hyperparameters, capturing various settings that could significantly impact the model's performance. This grid, outlining the ranges and values considered for each hyperparameter, is provided in the appendix for reference. The details of the specific grids used for each machine learning model tested to predict depression (ZUNG) class are given in detail in the appendix B.1, allowing for transparency and reproducibility of the experimentation process.

4.2.2 Results for the Problem 1 (D-from-M score)

After the training of each regression model designed to predict depression (ZUNG) score (Problem 1: D-from-M score from Section 2.5.1), we illustrate each regression model 's performance measure calculated via 10-cross validation technique as well as on the test set, itself, in the following tables.

Model	RMSE (CV)	RMSE (test)	MAE (CV)	MAE (test)	SMAPE (CV)	SMAPE (test)
Second averaged regression model	2.36381	5.52242	4.43816	4.41056	10.477	10.23
Weighted voting Regression (1)	2.36251	5.52503	4.43499	4.40782	10.478	10.22
Weighted voting Regression (2)	2.36225	5.52543	4.4334	4.40772	10.473	10.22
Stacking Regression (1)	2.36382	5.53396	4.43424	4.39927	10.468	10.20
Stacking Regression (2)	2.36283	5.53479	4.43178	4.40688	10.465	10.22
First averaged model	2.36933	5.53647	4.45349	4.40652	10.511	10.23
Weighted voting Regression (3)	2.35886	5.53651	4.41478	4.42447	10.421	10.27
ElasticCV Regression	2.37389	5.53681	4.47549	4.43039	10.577	10.28
Lasso Regression	2.36774	5.53721	4.45704	4.42446	10.534	10.27
Ridge Regression	2.36888	5.54314	4.46167	4.44022	10.547	10.30
RidgeCV Regression	2.37582	5.54363	4.48377	4.44069	10.597	10.30
Bagging Regressor	2.37133	5.54377	4.46786	4.43629	10.566	10.32
Linear Regression	2.37132	5.54378	4.46789	4.43627	10.567	10.31
Elastic Net Regression	2.36885	5.54411	4.45915	4.43788	10.543	10.31
Stacking Regression (3)	2.36132	5.54983	4.42331	4.42862	10.442	10.28
XGBoost Regressor	2.36575	5.59442	4.44029	4.44516	10.479	10.31
Stacking Regression (4)	2.36942	5.6076	4.44208	4.47558	10.479	10.40
CatBoost Regressor	2.3666	5.62218	4.42629	4.49914	10.448	10.45
Gradient Boosting Regressor	2.37496	5.62467	4.46392	4.48009	10.527	10.40
LightGBM Regressor	2.38827	5.64626	4.52356	4.51751	10.666	10.49
AdaBoost Regressor	2.39611	5.68707	4.52466	4.53743	10.663	10.52
Stacking Regression (5)	2.39781	5.6974	4.5593	4.57823	10.740	10.66
Extra Trees Regressor	2.42036	5.80048	4.64361	4.64903	10.935	10.79
Random Forest Regressor	2.42402	5.80359	4.63193	4.62952	10.913	10.74
LassoCV Regression	2.54028	6.34258	5.13647	5.05154	12.124	11.68
KNN Regressor	2.56693	6.63388	5.2242	5.28204	12.284	12.16
Support Vectors Regression	2.71955	7.04617	5.7712	5.54169	13.607	12.94

Table 4.21: Performance measure calculated via 10-cross validation as well as on the test set for each ML model used to predict depression (ZUNG) score.

Model	MAAPE (CV)	MAAPE (test)	MSE (CV)	MSE (test)	$\mathbf{R^2(CV)}$	$\mathbf{R^2(test)}$
Second averaged regression model	0.11851	0.10223	31.32754	30.49715	0.65273	0.64526
Weighted voting Regression (1)	0.10584	0.10217	31.2669	30.5259	0.65324	0.64492
Weighted voting Regression (2)	0.10579	0.10216	31.25239	30.53039	0.65339	0.64487
Stacking Regression (1)	0.10575	0.10195	31.33838	30.62469	0.65245	0.64377
Stacking Regression (2)	0.105720	0.10214	31.28102	30.63387	0.65305	0.64367
First averaged regression model	0.17798	0.10201	31.63005	30.65254	0.64946	0.64345
Weighted voting Regression (3)	0.10532	0.10263	31.07722	30.65297	0.65543	0.64345
ElasticCV Regression	0.10683	0.10275	31.8145	30.65624	0.64677	0.64341
Lasso Regression	0.10638	0.10262	31.55525	30.66068	0.65009	0.64336
Ridge Regression	0.10653	0.10297	31.61174	30.72643	0.64947	0.64259
RidgeCV Regression	0.10701	0.10298	31.91612	30.7318	0.64557	0.64253
Bagging Regressor	0.10661	0.10291	31.74873	30.73345	0.64793	0.64251
Linear Regression	0.10662	0.10292	31.74872	30.73355	0.64795	0.6425
Elastic Net Regressor	0.1064	0.10295	31.61824	30.73714	0.64939	0.64247
Stacking Regression (3)	0.10548	0.10269	31.21237	30.80056	0.65392	0.64173
XGBoost Regressor	0.10583	0.10314	31.42103	31.29751	0.65156	0.63595
Stacking Regression (4)	0.10586	0.10363	31.63646	31.44519	0.64926	0.63423
CatBoost Regressor	0.10559	0.1042	31.48061	31.60892	0.65094	0.63233
Gradient Boosting Regressor	0.10634	0.10397	31.93868	31.63693	0.64579	0.63199
LightGBM Regressor	0.1077	0.10471	32.65037	31.88029	0.63786	0.62917
AdaBoost Regressor	0.10798	0.10502	33.04641	32.34281	0.63367	0.62379
Stacking Regression (5)	0.10816	0.10658	33.21549	32.46041	0.63185	0.62242
Extra Trees Regressor	0.11079	0.10785	34.41742	33.64551	0.61859	0.60864
Random Forest Regressor	0.11035	0.10725	34.63442	33.68174	0.61605	0.60821
LassoCV Regressor	0.12313	0.11735	41.71499	40.22827	0.5374	0.53207
KNN Regressor	0.12297	0.12088	43.53819	44.00838	0.51786	0.48809
Support Vector Regressor	0.13559	0.12671	54.90445	49.64851	0.39282	0.42249

Table 4.22: Performance measure calculated via 10-cross validation as well as on the test set for each ML model used to predict depression (ZUNG) score.

Based on the Table 4.21 of performance measures for different ML models used to predict depression (ZUNG) score, we can make the following comparisons:

1. RMSE (Root Mean Squared Error):
    - The model with the lowest RMSE (CV) score is “Weighted voting Regression (3)” with a value of 2.35886 and then follows “Stacking Regression (3)” with value of 2.36132.
    - The model with the lowest RMSE (test) score is “Second averaged regression” model with a value of 5.52242. Then, follows the “Weighted voting Regression (1)” with a value of 5.52503.
2. MAE (Mean Absolute Error):

- The model with the lowest MAE (CV) scores is “Weighted voting Regression (3)” with a value of 4.41478. Then, follows the “Stacking Regression (3)” with a value of 4.42331.
- The model with the lowest MAE (test) scores is the “First averaged model” with a value of 4.40652. Then, follows the “Stacking Regression (2)” with a value of 4.40688.

3. SMAPE (Symmetric Mean Absolute Percentage Error):
    - The models with the lowest SMAPE (CV) scores are ”Weighted voting Regression (3)” and ”Stacking Regression (3)” with values of 10.421 and 10.442, respectively.
    - The model with the lowest SMAPE (test) scores is ”Stacking Regression (1)” with a value 10.20.
4. Based on these comparisons, ”Weighted voting Regression (3)” and ”Stacking Regression (3)” consistently perform well across multiple evaluation metrics. It’s important to note that these comparisons are based on the provided data, and further analysis or experimentation may be required to validate the performance of these models in different scenarios or datasets.

We can analyze the provided data from the Table 4.21 to identify potential cases of overfitting, underfitting, and to assess model performance using cross-validation and test set metrics. Here are some observations:

5. Overfitting:
    - In the given data, if a model has significantly lower performance metrics (RMSE, MAE, SMAPE) on the test set compared to the cross-validation set, it indicates possible overfitting.
    - For instance, “Extra Trees Regressor” and “Random Forest Regressor” have higher RMSE (test), MAE (test), and SMAPE (test) values compared to their cross-validation counterparts, suggesting overfitting.
6. Underfitting:
    - Underfitting occurs when a model fails to capture the underlying patterns in the data, resulting in poor performance on both the training and test data.
    - If a model has relatively high performance metrics (RMSE, MAE, SMAPE) on both the cross-validation and test sets, it may indicate underfitting.
    - However, based on the provided data, there aren’t clear cases of severe underfitting. Some models like “Support Vectors Regression” show relatively higher errors compared to others, but it is not evident that they are consistently underfitting.

7. Cross-validation vs. Test Set Metrics:
    - Cross-validation metrics are calculated by repeatedly training and evaluating the model on different subsets of the data, providing an estimate of the model's generalization performance.
    - Test set metrics are calculated by evaluating the model on a separate, unseen dataset, which helps assess the model's performance on new, independent data.
    - Generally, if the cross-validation metrics and test set metrics are close, it indicates that the model is performing well and generalizing to new data.
    - Models like "Second averaged regression model," "Weighted voting Regression (1-3)," and "Stacking Regression (1-3)" show similar performance between cross-validation and test set metrics, suggesting good generalization.
8. It's important to note that the provided observations are based on the given data, and further analysis or experimentation may be necessary to draw definitive conclusions about the presence of overfitting, underfitting, and model performance.

Based on the provided performance measures (MAAPE, MSE, R2) from Table 4.22 for each ML model, we can make comparisons between the models. Here are some observations:

1. The best-performing models in terms of MAAPE (CV) are the Weighted Voting Regression model (3), Stacking Regression model (3) and Stacking Regression model (2) with MAAPE values ranging from 0.10532 to 0.10572. These models have the lowest average percentage error on the cross-validated data.
2. The model with the lowest MAAPE (test) is Stacking Regression (1), with a value of 0.10195. This indicates that it has the lowest average percentage error on the test set.
3. When considering MSE (CV) and MSE (test), the models with the lowest values are the Second Averaged Regression Model and the Weighted Voting Regression models (1, 2, and 3). These models have relatively similar performance in terms of mean squared error.
4. In terms of R2 (CV) and R2 (test), the models with the highest values are the Weighted Voting Regression models (1, 2, and 3). These models have a higher coefficient of determination, indicating better goodness of fit.
5. The ElasticCV Regression, Lasso Regression, Ridge Regression, RidgeCV Regression, Bagging Regressor, and Linear Regression have similar performance across various measures, suggesting comparable performance among these models.
6. The worst-performing models, based on the provided measures, are the Support Vector Regressor, KNN Regressor, and LassoCV Regressor. These models have higher MAAPE, MSE, and lower R2 values compared to other models in the table.

7. Overall, the Weighted Voting Regression models (1, 2, and 3) and the Second Averaged Regression Model demonstrate strong performance across multiple evaluation metrics, indicating their effectiveness in predicting the depression (ZUNG) score.

Also, we can examine the provided Table 4.22 to analyze overfitting, underfitting, and the performance of models using cross-validation and test set metrics. Here are some observations:

8. overfitting: One way to identify overfitting is by comparing the performance of the model on the training set (cross-validation) and the test set. If the model has significantly better performance on the training set than on the test set, it indicates overfitting.

   In the provided table, the Second Averaged Regression Model has a lower MAAPE (CV) value of 0.11851 compared to its MAAPE (test) value of 0.10223. This suggests a potential overfitting issue, as the model performs better on the training data than on the unseen test data.

9. Underfitting: Underfitting occurs when a model fails to capture the underlying patterns in the data and performs poorly on both the training and test sets. It is characterized by high errors and low R^2 values.

   - None of the models in the provided table exhibit significant underfitting based on their performance measures. However, models with higher MAAPE, MSE, and lower R^2 values, such as the Support Vector Regressor, KNN Regressor, and LassoCV Regressor, may indicate a potential underfitting problem.

10. Cross-validation metrics:

    - The Weighted Voting Regression models (1, 2, and 3) consistently exhibit lower MAAPE (CV) values, indicating better performance on cross-validated data.
    - Models like the Support Vector Regressor, KNN Regressor, and LassoCV Regressor have higher MAAPE (CV) values, suggesting poorer performance on cross-validated data.

11. Test set metrics:

    - The Stacking Regression (1) has the lowest MAAPE (test) value of 0.10194, indicating better performance on unseen data.
    - Other models, such as the Weighted Voting Regression models, Stacking Regression models, and various individual regression models, also demonstrate comparable performance based on their MAAPE (test) values.

12. In summary, while there may be some indications of overfitting in the Second Averaged Regression Model, most of the models in the provided table show reasonably consistent

performance between cross-validation and test set metrics. The Weighted Voting Regression models generally perform well in terms of both cross-validation and test set metrics, suggesting their potential effectiveness in predicting the Depression (ZUNG) class.

From these observations, we notice that Weighted voting Regression (3) model has the best performance in 6 out of 12 calculated metrics. Thus, we propose Weighted voting Regression (3) model for modeling the (D-from-M score) or predicting depression (ZUNG) score.

The elements necessary to specify the Weighted Voting Regression (3) model are the regressors and the weights used for voting. These elements are given in the Table 4.23.

Model	**Meta-regressor**	**Regressor**
Weighted Voting Regression model (3)	Linear Regression	Lasso Regression
		Elastic Net Regression
		Gradient Boosting Regressor

Table 4.23: The regressors of the Weigthed voting Regression (3) models used to predict depression (ZUNG) score.

To ensure optimal performance and fine-tuned configuration of the tested machine learning models, we employed the grid search method to select the most suitable hyperparameters. The details of the specific grids used for each machine learning model and the final values of the hyperparameters tested to predict depression (ZUNG) score are given in detail in the Appendix B.2, allowing for transparency and reproducibility of the experimentation process.

4.2.3 Results for the Problem 3 (M-from-D score)

After the training of each regression model designed to predict memory (PRMQ) score (Problem 3: M-from-D score from Section 2.5.3), we illustrate each regression model 's performance measure calculated via 10-cross validation as well as on the test set, itself, in the Tables 4.24 and 4.25.

Model	RMSE (CV)	RMSE (test)	MAE (CV)	MAE (test)	SMAPE (CV)	SMAPE (test)
Stacking Regression (1)	2.76952	7.6284	6.01643	5.94725	32.715	31.49
Second averaged regression model	2.76647	7.63245	6.00143	5.94761	32.480	31.38
Stacking Regression (2)	2.76979	7.63671	6.02034	5.95041	32.711	31.48
Stacking Regression (3)	2.76382	7.64591	5.97894	5.95803	32.470	31.49
Weighted voting Regression (1)	2.76503	7.65511	5.98887	5.96416	32.460	31.49
Stacking Regression (4)	2.77409	7.6557	6.01612	5.95753	32.537	31.44
Weighted voting Regression (2)	2.7701	7.66308	6.00426	5.97249	32.523	31.58
Gradient Boosting Regressor	2.772975	7.673872	6.008919	5.968707	32.471	31.48
Weighted voting Regression (3)	2.76894	7.67823	5.99801	5.97118	32.458	31.57
First averaged regression model	2.77142	7.68263	6.01708	5.99696	32.621	31.59
Weighted voting Regression (4)	2.76774	7.70115	5.99473	5.9911	32.416	31.61
Weighted voting Regression (5)	2.77831	7.72062	6.05807	6.02474	32.945	31.74
Lasso Regression	2.78108	7.73114	6.07049	6.02569	33.012	31.76
CatBoost Regressor	2.77519	7.73381	6.03281	5.98823	32.589	31.55
Elastic Net Regression	2.78176	7.74264	6.073	6.03274	32.993	31.77
ElasticCV Regression	2.78136	7.74273	6.07504	6.03707	33.011	31.79
Ridge Regression	2.78283	7.74585	6.08815	6.06421	33.212	32.00
RidgeCV Regression	2.78236	7.74898	6.08527	6.06097	33.116	31.94
XGBoost Regression	2.79319	7.75851	6.09199	6.04819	33.038	31.97
Stacking Regression (5)	2.79201	7.76291	6.07588	6.03889	32.779	31.80
Linear Regression	2.786291	7.764213	6.099420	6.093982	33.356	32.23
Stacking Regression (6)	2.795286	7.781608	6.087085	6.057157	32.831	31.93
AdaBoost Regressor	2.80075	7.79922	6.12796	6.06138	33.122	32.04
LightGBM Regressor	2.78897	7.79986	6.0721	6.07985	32.900	31.89
Stacking Regression (7)	2.833005	7.831471	6.279176	6.071750	33.665	32.05
Bagging Regression	2.79748	7.8367	6.15047	6.09202	33.211	32.15
Random Forest Regressor	2.807405	7.852295	6.165974	6.125943	33.308	32.29
Extra Trees Regressor	2.80344	7.87736	6.12641	6.11362	32.953	32.12
Stacking Regression (8)	2.83205	8.06624	6.29262	6.30448	33.782	32.72
LassoCV Regreiion	2.85828	8.31645	6.40931	6.48687	34.359	33.81
KNN Regressor	2.989342	8.779788	6.960747	6.778069	36.796	35.47
Support Vector Regression	3.072019	9.265469	7.311703	7.028870	40.793	38.93

Table 4.24: Performance measures calculated via 10-cross validation as well as on the test set for each ML model used to predict memory (PRMQ) score.

Model	MAAPE (CV)	MAAPE (test)	MSE (CV)	MSE (test)	$\mathbf{R^2(CV)}$	$\mathbf{R^2(test)}$
Stacking Regression (1)	0.32632	0.31369	59.12159	58.19254	0.52203	0.55923
Second averaged regression model	0.33381	0.31482	58.83056	58.25423	0.52434	0.55877
Stacking Regression (2)	0.32693	0.31432	59.14397	58.31941	0.52189	0.55827
Stacking Regression (3)	0.32415	0.31394	58.63854	58.45997	0.52597	0.55721
Weighted voting Regression (1)	0.32541	0.31556	58.72352	58.6007	0.52527	0.55614
Stacking Regression (4)	0.32529	0.31446	59.46623	58.60974	0.51884	0.55607
Weighted voting Regression (2)	0.32501	0.31513	59.13659	58.72286	0.52174	0.55522
Gradient Boosting Regressor	.3253	0.31569	59.41539	58.88832	0.51959	0.55396
Weighted voting Regression (3)	0.32463	0.31506	59.05106	58.95521	0.52248	0.55346
First averaged regression model	0.32847	0.31656	59.26518	59.02281	0.52079	0.55294
Weighted voting Regression (4)	0.32519	0.31694	58.94953	59.30777	0.52335	0.55079
Weighted voting Regression (5)	0.32874	0.31698	59.87041	59.60793	0.51617	0.54851
Lasso Regression	0.329215	0.316695	60.104103	59.770573	0.514271	0.547279
CatBoost Regressor	0.3268	0.31637	59.55989	59.81187	0.51836	0.54697
Elastic Net Regression	0.32949	0.31726	60.16141	59.94848	0.51382	0.54593
ElasticCV Regression	0.32941	0.31744	60.12697	59.94987	0.51189	0.54592
Ridge Regression	0.33025	0.31836	60.27181	59.99813	0.51295	0.54556
RidgeCV Regression	0.33011	0.31836	60.20294	60.04674	0.51121	0.54519
XGBoost Regressor	0.32847	0.31755	61.16058	60.19451	0.50522	0.54407
Stacking Regression (5)	0.32755	0.31786	61.00985	60.26281	0.50611	0.54355
Linear Regression	0.33034	0.31948	60.58236	60.28299	0.51048	0.54339
Stacking Regression (6)	0.32803	0.31911	61.30837	60.55342	0.50392	0.54135
AdaBoost Regressor	0.32768	0.31611	61.76975	60.82784	0.49991	0.53927
LightGBM Regressor	0.32779	0.31858	60.83011	60.83778	0.50832	0.53919
Stacking Regression (7)	0.333674	0.319137	64.709229	61.331943	0.476334	0.535453
Bagging Regression	0.33381	0.32218	61.4673	61.41392	0.50265	0.53483
Random Forest Regressor	0.33374	0.32294	62.33373	61.65854	0.49539	0.53298
Extra Trees Regressor	0.33099	0.32215	62.00698	62.05284	0.4985	0.52999
Stacking Regression (8)	0.33445	0.32672	64.58796	65.06427	0.47747	0.50718
LassoCV Regression	0.34664	0.34067	67.04736	69.16337	0.45562	0.47614
KNN Regressor	0.35768	0.34268	80.11971	77.08467	0.35222	0.41614
Support Vector Regression	0.37229	0.34778	89.29185	85.84892	0.27543	0.34975

TABLE 4.25: Performance measures calculated via 10-cross validation as well as on the test set for each ML model used to predict memory (PRMQ) score.

From the Table 4.24, we compared the models tested to predict memory (PRMQ) score based on their performance measures and we noticed the following:

1. RMSE (CV) and RMSE (test):

    - The models with lower RMSE (CV) and RMSE (test) values generally indicate better performance in terms of predicting the memory (PRMQ) score.
    - The models "Stacking Regression (3)", "Stacking Regression (1)", and "Second averaged regression model" have the lowest RMSE (CV) and RMSE (test) values, suggesting better predictive accuracy.

2. MAE (CV) and MAE (test):

    - MAE (CV) and MAE (test) values indicate better performance in terms of capturing the average absolute difference between predicted and actual depression scores.
    - The models “Stacking Regression (3)”, “Stacking Regression (1)”, and “Second averaged regression model” again have the lowest MAE (CV) and MAE (test) values, indicating better accuracy.

3. SMAPE (CV) and SMAPE (test):

    - Lower SMAPE (CV) and SMAPE (test) values indicate better performance in terms of the symmetric mean absolute percentage error.
    - The models “Second averaged regression model”, “Stacking Regression (1)”, and “Stacking Regression (4)” have the lowest SMAPE (CV) and SMAPE (test) values, suggesting better performance.

4. Based on these comparisons, the models ”Stacking Regression (3)”, ”Stacking Regression (1)”, and ”Second averaged regression model” consistently perform well across multiple metrics. They have relatively lower RMSE, MAE, and SMAPE values both on cross-validation and the test set. These models show promise in accurately predicting the memory (PRMQ) score.

From Table 4.24, we analyzed the provided data and make observations regarding overfitting, underfitting, and the performance of the models using cross-validation metrics and test set metrics and we noticed the following:

5. RMSE (CV) and RMSE (test):

    - Lower RMSE (CV) and RMSE (test) values indicate better model performance in terms of Root Mean Squared Error.
    - Models with similar RMSE values for both CV and the test set are less likely to suffer from overfitting or underfitting.
    - The models “Stacking Regression (1)”, “Second averaged regression model”, and “Stacking Regression (2)” have relatively low RMSE values for both CV and the test set, suggesting consistent performance.

6. MAE (CV) and MAE (test):

    - Lower MAE (CV) and MAE (test) values indicate better model performance in terms of Mean Absolute Error.
    - Models with similar MAE values for both CV and the test set are less likely to suffer from overfitting or underfitting.

- The models "Stacking Regression (3)", "Weighted voting Regression (1)", and "Stacking Regression (4)" have relatively low MAE values for both CV and the test set, indicating consistent performance.

7. SMAPE (CV) and SMAPE (test):
   - Lower SMAPE (CV) and SMAPE (test) values indicate better model performance in terms of Symmetric Mean Absolute Percentage Error.
   - Models with similar SMAPE values for both CV and the test set are less likely to suffer from overfitting or underfitting.
   - The models "Stacking Regression (1)", "Second averaged regression model", and "Stacking Regression (2)" have relatively low SMAPE values for both CV and the test set, suggesting consistent performance.
8. Based on these observations, the models "Stacking Regression (1)", "Second averaged regression model", and "Stacking Regression (2)" exhibit consistent performance across multiple metrics. They have relatively low RMSE, MAE, and SMAPE values for both CV and the test set. These models are less likely to suffer from overfitting or underfitting and provide reliable predictions for the memory (PRMQ) score.
9. On the other hand, models with significant discrepancies between CV and test set metrics or relatively higher errors, such as "KNN Regressor", "Support Vector Regression", and "LassoCV Regression", may indicate overfitting or underfitting issues. These models should be further evaluated and potentially fine-tuned to improve their performance.

Based on the performance measures provided in the Table 4.25, we can make comparisons between the different ML models used to predict the memory (PRMQ) score. Here are some observations:

1. Stacking Regression (1), Stacking Regression (2), Stacking Regression (3), Stacking Regression (4), Stacking Regression (5), and Stacking Regression (6) are different variations of stacking regression models. They generally perform well compared to other models in terms of MAAPE, MSE, and R^2 scores.
2. The Second averaged regression model and the First averaged regression model are two different ensemble models that involve averaging the predictions of multiple regression models. Both models show similar performance, but the First averaged regression model has slightly better results in terms of MAAPE and MSE.
3. Weighted voting Regression (1), Weighted voting Regression (2), Weighted voting Regression (3), Weighted voting Regression (4), and Weighted voting Regression (5) are

different variations of weighted voting regression models. They perform reasonably well, with similar performance across the different versions.

4. The Gradient Boosting Regressor, CatBoost Regressor, AdaBoost Regressor, LightGBM Regressor, Bagging Regression, Random Forest Regressor, and Extra Trees Regressor are ensemble models that utilize boosting or bagging techniques. They generally perform better than linear models like Linear Regression, Lasso Regression, Elastic Net Regression, Ridge Regression, and their variants in terms of MAAPE, MSE, and R^2 scores.

5. The XGBoost Regressor shows good performance in terms of R2 score but has slightly higher MAAPE and MSE compared to other ensemble models.

6. LassoCV Regression and Lasso Regression show similar performance, but LassoCV Regression performs slightly better in terms of MAAPE, MSE, and R^2 scores.

7. KNN Regressor, Support Vector Regression, and ElasticCV Regression have relatively lower performance compared to other models. They have higher MAAPE and MSE and lower R^2 scores, indicating that they may not be the best models for predicting the memory score in this case.

8. Overall, the stacking regression models, ensemble models utilizing boosting or bagging techniques, and some of the weighted voting regression models tend to perform better than other models in terms of MAAPE, MSE, and R^2 scores. It's important to note that the specific performance of each model may vary depending on the dataset and problem at hand, so it's recommended to consider these results as a starting point and conduct further experimentation and evaluation to choose the most suitable model.

We analyzed the data from the table 4.25 to identify overfitting, underfitting, and evaluate the performance of the models using cross-validation metrics and test set metrics. Here are some observations:

9. Overfitting: Overfitting occurs when a model performs well on the training data but poorly on unseen data. Signs of overfitting include significantly better performance on the training set compared to the test set or cross-validation results. Models that potentially exhibit overfitting:

    - Stacking Regression (8): This model shows a high MAAPE and MSE on the test set compared to cross-validation, indicating possible overfitting.
    - LassoCV Regression: Similar to Stacking Regression (8), this model has higher MAAPE and MSE on the test set, suggesting overfitting.

10. Underfitting: Underfitting happens when a model fails to capture the underlying patterns in the data, resulting in poor performance on both the training and test sets. Signs of underfitting include consistently low performance metrics. Models that potentially exhibit underfitting:

    - KNN Regressor: This model shows relatively high MAAPE and MSE on both the test set and cross-validation, indicating underfitting.
    - Support Vector Regression: Similar to KNN Regressor, this model has consistently higher MAAPE and MSE scores on both test set and cross-validation, suggesting underfitting.

11. Cross-validation metrics vs. Test set metrics: Cross-validation is a technique used to estimate the performance of a model on unseen data by splitting the training data into multiple folds. The test set metrics, on the other hand, provide the actual performance of the model on completely unseen data.

    - In general, the cross-validation metrics and test set metrics are relatively close for most models, indicating that the models are performing consistently on unseen data.
    - However, some models show slight variations between the cross-validation metrics and test set metrics. For example, models like Stacking Regression (1), Stacking Regression (2), Stacking Regression (3), Second averaged regression model, and Gradient Boosting Regressor have slightly better performance on the test set compared to cross-validation, suggesting good generalization capabilities.

From the previous observations, we see that Stacking Regression (1) has the best performance in 5 of the 12 calculated metrics and Stacking Regression (3) has also the best performance in 5 of the 12 metrics. So we propose a Stacking Regression model for modeling the memory-related regression task. We choose the Stacking regression (1) model to predict Memory (PRMQ) score as it has the best performance according to the R^2 metric calculated on the test set.

The elements necessary to specify the Stacking Regression (1) model are the meta-regressor and the regressors. In this case the meta-regressor is a Lasso Regression model. The regressors and their hyperparameters are given in the Table 4.26.

Model	**Meta-regressor**	**Regressor**
Stacking Regression (1)	Linear Regression	Lasso Regression
		ElasticNet Regression
		XGBoost Regression

TABLE 4.26: The necessary elements to specify the Stacking Regression (1) model used to predict the memory (PRMQ) score.

To ensure optimal performance and fine-tuned configuration of the tested machine learning models, we employed the grid search method to select the most suitable hyperparameters. The details of the specific grids used for each machine learning model and the final values of the hyperparameters tested to predict memory (PRMQ) score are given in detail in the Appendix B.3, allowing for transparency and reproducibility of the experimentation process.

Chapter 5

Imbalanced classification and the memory-related classification task

Chapter 5 discusses the problem of imbalanced classification in machine learning. Several approaches have been proposed to solve this problem, such as algorithm-level techniques, data-level strategies, and hybrid-level techniques. The chapter also describes the inadequacy of traditional performance metrics, such as accuracy, for imbalanced datasets and presents appropriate metrics, such as precision, recall, F1-measure, and area under the ROC curve (AUC). Moreover, Chapter 5 presents the methodology and results for solving the memory-related problem classification problem (M-from-D class). The main objective is to diagnose memory-related problems through questions used to diagnose depression, and the task is an imbalanced binary classification problem. In the methodology section, two approaches were applied: data level methods and Cost-Sensitive learning. At the end of the chapter, the results of the M-from-D class are given.

5.1 Imbalanced classification

Classification can be divided into two sub-categories. The first one is termed *complete classification* and is used with balanced datasets. In machine learning, a balanced dataset refers to a dataset where the number of data points for each class or category is approximately equal. For example, in a binary classification problem where we are trying to predict whether an email is spam or not, a balanced dataset would mean having roughly the same number of spam and non-spam emails in the dataset. A balanced dataset is desirable because it helps prevent bias in the model towards any particular class and ensures that the model is equally capable of predicting both classes. However, in some real-world scenarios, such as medical diagnosis, it's common to have imbalanced datasets where one class significantly outnumbers the other. In

such cases, specific techniques can be used to balance the dataset during the training process. The second is called *partial classification* and used on imbalanced datasets [65], which is a basic issue in machine learning and has drawn a lot of attention [66, 67]. In the binary imbalanced datasets, there are more samples belonging to one class than there are of the other. The first class is therefore referred to as the majority class, and the second as the minority class. As a result, there are two types of classes in such datasets: majority and minority. The imbalance ratio (IR) [65],

$$IR = \frac{\text{number of majority instances}}{\text{number of minority instances}}$$

is a statistical indicator of the distribution of instances in imbalanced binary datasets. There are three groups of imbalanced datasets, as determined by the value of their IR: Datasets with low imbalance (IR is between 1.5 and 3), datasets with medium imbalance (IR is between 3 and 9), and datasets with high imbalance (IR is higher than 9)

Several methods have been put forth in an effort to solve the binary imbalanced classification problem. We categorize the methods using: The algorithm level, the data level, and the hybrid level. This categorization has been done on the basis of the way the methods handle the class imbalance. First, algorithm-level [66, 68] techniques modify the existing classification algorithms to give more importance to the minority class during the training process. Second, data-level strategies [69, 67], such as undersampling and oversampling provide a processing step to lessen the effect of skewed class distribution. Last but not least, algorithm level and data-level techniques like resampling and cost-sensitive learning are combined when the hybrid-level techniques [70, 71] are applied.

5.2 Evaluation metrics for the imbalanced classification task

For imbalanced datasets, the traditional performance metrics used to assess classifier performance e.g. accuracy, are inappropriate. This comes as a result of their sensitivity to class skews and significant bias toward the majority class. For instance, the accuracy metric can be misleading when the dataset is imbalanced. For example, in a dataset with only 1% of minority instances and 99% of majority instances, the accuracy is 99% if all majority instances are correctly classified. Nevertheless, an enormous cost can arise from the misclassification of the 1% minority instances and, for a medical diagnosis the seemingly high 99% accuracy might be disastrous. As a result, alternate metrics are required to evaluate the effectiveness of classifiers with imbalanced datasets.

We can use the confusion matrix, to extract directly certain measures [65]. They can evaluate independently how well the majority and minority classes are classified. We can also combine some others to evaluate the performance of a classifier. Below is a description of some performance metrics which are appropriate for imbalanced classification:

- **Precision:** is the percentage of the minority instances which were correctly classified relative to all instances that were classified to the minority class.
- **Recall:** is the percentage of minority instances correctly classified as belonging to the minority class.
- **F1 measure:** is the harmonic mean of precision and recall. The increase of the accuracy and recall causes a proportional increase in the F1 metric. When F1 is high, the model is more effective in predicting the minority class.
- **Area under the ROC curve (AUC):** A ROC (Receiver Operating Characteristic) curve displays a binary classifier system's performance in a visual format by changing its discrimination threshold. It graphs the true positive rate (TPR) against the false positive rate (FPR) using varying threshold values. The true positive rate is the proportion of correct positive predictions to the total number of positive samples in the data, while the false positive rate is the proportion of incorrect positive predictions to the total number of negative samples.

  The area under the ROC curve (AUC) is used to compare the performance of two classifiers. If classifier C1's area is larger than classifier C2's area, then C1's performance will be better than C2's. To calculate the AUC, first sort the data in descending order based on the predicted probability of the positive class. Then, initialize the true_positives and false_positives variables to 0, as well as the area variable. Proceed by looping through each sample in the sorted data, incrementing true_positives or false_positives depending on whether the sample is a true positive or false positive. Calculate the true positive rate (TPR) and false positive rate (FPR) using the updated true_positives and false_positives values. Next, calculate the area of the trapezoid formed by the current point and the previous point on the ROC curve using a specific formula. Add the area of the trapezoid to the area variable. Finally, set the previous_FPR and previous_TPR variables to the current FPR and TPR values, respectively, and return the area as the AUC.

5.3 Cross validation with resampling

If a resampling process is applied to an imbalanced dataset, cross-validation is a delicate task. If cross-validation is applied on already upsampled data, the performance metrics are not necessarily valid for new data. On the other hand if the resampling process is applied before cross validation, the same data points may end up being in the training and validation sets. As a result, the values of those data points will be perfectly predicted by a complicated enough model when predictions are made on the validation set, inflating the values of the accuracy and recall metrics.

When k cross-validating is applied in combination with resampling [72], the following steps guarantee the generalisation of the results:

- At the beginning of each cross-validation loop, choose randomly a sample from the data set, keep it separated and do not use it for anything related to resampling, model training, or feature selection.
  - The excluded sample is called the validation set.
  - The data set without the excluded sample is called the training set.
- Apply a method of resampling on the training set and take resampled data (oversampled minority class + the majority class, or undersampled majority class + the minority class).
- Use the resampled data to build and train the model.
- Use the excluded sample (validation set) for validation, i.e., testing the performance of the trained model to "unknown" data.
- Repeat the above steps k times.

5.4 Techniques for imbalanced classification

Data-level resampling techniques are a class of techniques used to address class imbalance in machine learning. These techniques involve modifying the original dataset by either oversampling the minority class, undersampling the majority class, or both. The goal of these techniques is to create a balanced dataset that can improve the performance of the machine learning model.

5.4.1 Data-level methods

Data-level methods [65] resample the data to lessen the effect of the imbalance. They are divided into three categories: oversampling, undersampling and hybrid approaches.

5.4.2 Oversampling methods

Oversampling is used to expand a dataset that is unevenly distributed by replicating a few minority instances. The following techniques can be used for this duplication.

5.4.2.1 Random Oversampling

In random oversampling, the minority class is oversampled by randomly duplicating some of its samples until its size matches the size of the majority class. This is done to balance the class distribution and avoid bias towards the majority class during model training. However, duplicating samples can lead to overfitting, where the model may become too focused on the minority class and not generalize well to new data.

5.4.2.2 Synthetic minority oversampling technique (SMOTE)

SMOTE is a data-generating synthetic approach. For each minority instance, x_i, SMOTE generates a synthetic example x_{new} as shown in the Equation (5.1). Initially, SMOTE finds the K-nearest neighbors $(y_j, j \in \{1, ..., K\})$ of x_i. Then, it finds one of K-nearest neighbors, y_i, at random. Finally, SMOTE creates a new sample by interpolating between the two samples. The interpolation is done by selecting a random point along the line that connects the two samples and using this point to create the new synthetic sample. The previous process is done by applying the following equation, where δ is a number chosen at random in the interval $[0, 1]$.

$$x_{new} = x_i + (y_i - x_i) * \delta \tag{5.1}$$

We recognize that x_{new} is a point on the segment that connects x_i and y_i. SMOTE has the benefit of improving the generalization performance of the model by increasing the diversity of the data. SMOTE produces artificial samples that don't exactly replicate existing ones, but instead, create new instances within the feature space. Nonetheless, SMOTE may cause overfitting when the newly created samples resemble the existing ones too closely. As a result, it is crucial to cautiously fine-tune the SMOTE algorithm's parameters, such as the oversampling ratio and the number of nearest neighbors, to achieve a harmonious balance between oversampling and generalization.

5.4.2.3 Adaptive Synthetic Sampling Approach (ADASYN)

ADASYN [69] is an adaptive synthetic sampling approach for imbalanced learning. The main concept behind ADASYN is the usage of a weighted distribution for different minority class instances depending on their level of learning difficulty, e.g., for minority class instances that are more challenging to learn, more synthetic data is produced than for minority examples that are simpler to learn. The number of synthetic instances that may be produced for each minority instance is determined by ADASYN, using a distribution function termed *density*. This function measures the density of the minority class in the feature space. The algorithm then calculates the difference between the density of each sample in the minority class and the density of its k nearest neighbors. This difference is used to determine the degree of synthetic sample generation, with samples in sparser regions of the feature space receiving more synthetic samples. ADASYN has an edge over SMOTE in that it can create synthetic samples that are more varied and inclusive. This is due to its capability of generating samples in underrepresented regions of the minority class. Furthermore, ADASYN can efficiently cope with datasets with varying degrees of imbalance, as it modifies to the level of imbalance present in the data. Nevertheless, ADASYN, like other oversampling methods, has the potential of overfitting the model if the newly generated samples are too much alike to the existing ones. Hence, it is crucial to assess the model's performance meticulously by employing techniques like cross-validation.

5.4.3 Undersampling methods

Undersampling methods reduce the data size by deleting some majority instances with the objective of equalizing the number of instances of each class [44]. There are several approaches for this kind of undersampling, that differ in the way of selection of majority instances that will be deleted.

5.4.3.1 Random undersampling

Random undersampling is the inverse of random oversampling. The aim of this strategy is to reduce the number of examples in the majority class in the modified data by randomly selecting and removing samples from it.

5.4.3.2 Tomek Links

Tomek Links is an undersampling technique [73] that is a modification of the Condensed Nearest Neighbors (CNN) created by Tomek (1976). It involves identifying pairs of examples from

different classes that are closest to each other and then removing the example from the majority class in that pair. The Tomek Links technique, as opposed to the CNN approach which chooses at random samples from the majority class with their k nearest neighbors, employs a rule that defines a *Tomek Link* to choose the pair of observations. In particular, let's say, a and b fulfill all the following properties:

- Observation b is nearest to observation a.
- Observation a is nearest to observation b.
- The observations a and b are of distinct classes. This means that a and b are members of the minority and majority classes, respectively, or vice versa.

It may be stated mathematically as follows: Let $d(x_i, x_j)$ be the Euclidean distance between samples x_i and x_j, where x_i stands for the minority class sample and x_j for the majority class sample. If there is no sample x_k satisfying the following conditions:

1. $d(x_i, x_k) < d(x_i, x_j)$, or
2. $d(x_j, x_k) < d(x_i, x_j)$

then the pair of (x_i, x_j) is a *Tomek Link*.

This process continues until there are no more such pairs with a *Tomek Link* left in the dataset. The removal of these examples creates a cleaner and more easily separable dataset, thereby improving the performance of machine learning models on imbalanced datasets.

5.4.3.3 Edited Nearest Neighbor (ENN)

The ENN approach [74] finds each observation's K-nearest neighbor first, afterwards it checks whether or not the observation's class and its k-nearest neighbor's majority class are the same. The observation and its K-nearest neighbor are removed from the dataset if the majority class of the observation's neighbor is different from that of the observation.

The following will describe how the ENN algorithm works:

- Determine K, the number of nearest neighbors, given a dataset containing N observations.
- In the dataset, locate the observation's K-nearest neighbor and then return the majority class from K-nearest neighbor.

- The observation and its K-nearest neighbor are removed from the dataset if the class of the observation and the majority class from the observation's K-nearest neighbor vary.
- Till the desired proportion of each class is reached, repeat steps 2 and 3 as necessary.

Compared to Tomek Links, ENN approach is more powerful. When the class of the observation and the majority class from the observation's K-nearest neighbor vary, ENN drops the observation and its K-nearest neighbor, rather than just eliminating observation and its 1-nearest neighbor belonging to different data classes. As a result, ENN is anticipated to provide more in-depth data cleaning than Tomek Links.

5.4.4 Hybrid data-level

Hybrid data-level techniques are a class of techniques used to address class imbalance in machine learning that combine oversampling and undersampling techniques. The goal of these techniques is to create a balanced dataset while preserving as much information as possible. A brief presentation of hybrid data-level techniques used in this thesis are given below.

5.4.4.1 SMOTE-Tomek Links

The SMOTE-Tomek Links [75] algorithm is a hybrid data-level resampling technique used to address class imbalance in machine learning. The algorithm combines two techniques: Synthetic Minority Over-sampling Technique (SMOTE) and Tomek Links undersampling. SMOTE is used to oversample the minority class by creating synthetic samples that are generated by interpolating between existing minority class samples. This helps to increase the representation of the minority class in the dataset. Tomek Links, on the other hand, are pairs of samples from different classes that are close to each other but represent borderline cases. Tomek Links undersampling removes samples that belong to Tomek Links, which can help to reduce the overlap between the minority and majority classes

The SMOTE-Tomek Links algorithm works in the following way:

1. Apply SMOTE to oversample the minority class, generating synthetic samples.
2. Identify Tomek Links between the minority and majority classes.
3. Remove the samples from both classes that belong to Tomek Links.
4. Evaluate the performance of the machine learning model on the balanced dataset.

The combination of SMOTE and Tomek Links undersampling can improve the performance of the machine learning model by creating a more separable dataset that is less susceptible to noise and overlapping data points. This can result in a more accurate and reliable model that is better able to handle class imbalance.

5.4.4.2 SMOTE-ENN Method

SMOTE-ENN Method [74] combines SMOTE's ability to produce synthetic instances of the minority class data and the ability of ENN to exclude some observations from both classes that are determined to have a different class from the observation's class and the majority class of its K-nearest neighbors. The following describes how SMOTE-ENN works:

1. (SMOTE begins) Choose at random sample from the minority class.
2. Calculate the distance between the sample data and its k nearest neighbors.
3. Multiply the distance with a random number between 0 and 1, afterwards add the outcome to the minority class as a synthetic sample.
4. Till the needed size of the minority class is reached, repeat steps 2-3. (SMOTE ends)
5. (ENN begins) Determine K, the number of nearest neighbors.
6. In the dataset, locate the observation's K-nearest neighbors and then return the majority class from K-nearest neighbors.
7. The observation and its K-nearest neighbors are removed from the dataset if the class of the observation and the majority class from the observation's K-nearest neighbors differ.
8. Repeat steps 6 and 7 until the needed size of each class is reached. (ENN ends)

5.4.4.3 Cost-Sensitive Learning

Most machine learning algorithms assume that all misclassification errors made by a model are "equal" independently whether they involve minority or majority class targets. This is often not the case for imbalanced classification problems, where missing a positive (or minority) class case is worse than incorrectly classifying a sample from the negative (majority) class. Cost-sensitive learning [76, 77] is a subfield of machine learning that takes into account the costs of prediction errors (and potentially other costs) when training. Many machine learning algorithms can be updated to be cost-sensitive, where the model is penalized more for misclassification errors from one class compared to the other.

The scikit-learn library in Python provides this capability for a range of algorithms via the "class_weight" attribute specified when defining the model. A weighting can be specified that is inversely proportional to the class distribution. For example, if the class distribution was 0.99 to 0.01 for the majority and minority classes respectively, then the "class_weight" argument could be defined as a dictionary that defines a penalty of 0.01 for errors made for the majority class and a penalty of 0.99 for errors made with the minority class.

5.5 Solving Problem 4 (M-from-D class): Memory class through depression

One of our state goals in this thesis is to diagnose memory-related problems through questions that relate to depression, along with demographics and general health questions. From the available data, we see that there is an uneven distribution between those who have problems with their memory and those who do not. Specifically, approximately only 5% of subjects who responded to the questionnaires were diagnosed with a memory problem. Therefore, the task of predicting memory-related conditions through questions that are used to diagnose depression is an *imbalanced binary classification task*.

To solve the M-from-D problem, we attempted all of methods mentioned in Section 5.4. In each case, we evaluated the resulting models with the metrics described in Section 5.2 ,i.e. Precision, Recall, F1-Measure and AUC. More specifically, after randomly splitting the data set into a training set and a test set with a ratio of 80% to 20% in a manner that preserved the distribution of the memory classes (majority class 95%: without memory-related problems, minority class 5%: with memory-related problems), we followed each of two approaches to solve the binary imbalanced classification problem.

In the first approach, we applied, in turn, 14 classifiers: Logistic Regression, Decision Tree, Random Forest, Linear Discriminant Analysis, Support Vector, K-Nearest Neighbors, Bagging, AdaBoost, Gaussian Naive Bayes, Gradient Boosting, XGBoost, CatBoost and Extra Trees. Each classifier was trained on five different training sets:

1. Original training set
2. Training set arising from the application of SMOTE resampling on the original training set.
3. Training set arising from the application of ADASYN resampling on the original training set

4. Training set arising from the application of Smote & Tomek Links resampling on the original training set.

5. Training set arising from the application of Smote & ENN resampling on the original training set.

All trained models were evaluated on the same original test set which was independent of the resampling technique used during the training phase.

In the second approach, we applied Cost-Sensitive learning. In particular, different weights were given to the majority (subjects with no memory-related problems indicated) and minority (subjects with a memory-related problem) classes during the training phase of each classification algorithm, in order to take into account the skewed distribution of the classes. The idea was to penalize to a higher degree the misclassification made by the minority class by increasing its class weight, while simultaneously decreasing the corresponding weight for the majority class. After the class weights were set, they were multiplied by the class loss, and the minority class had a much higher error than the majority class. In practice, we used some of the sklearn modeling libraries and some boosting based libraries in Python which have an in-built parameter "class_weight". This parameter does exactly the job of assigning different weights to different classes. The "class_weight" setting's default value is None, in this case both classes have equal weights. To assign different weights to the two different classes, we set $class\_weight =$"balanced" or assign a dictionary with manual weights for each class to the $class\_weight$ parameter. In case, $class\_weight =$"balanced", the model automatically assigns the class weights to be inversely proportional to their corresponding frequencies, i.e.

$$w_j = \frac{n\_samples}{n\_classes * n\_samples\_j}$$

where w_j is the weight for the j-th class, $n\_samples$ is the total number of samples or rows in the dataset, $n\_classes$ is the total number of unique classes in the target (i.e. 2 in case of binary classification) and $n\_samples\_j$ is the total number of rows of the respective class.

In this thesis, we applied Cost-Sensitive Logistic Regression, Cost-Sensitive Decision trees, Cost-Sensitive Support Vector Classifier, Cost-Sensitive Random Forest Classifier, Cost-Sensitive Ridge Classifier, Cost-Sensitive XGBoost Classifier and Cost-Sensitive Extra Trees Classifier. For each model, we executed 3 trials by changing the class weight parameter. In the first trial, we used the $class\_weight =$"balanced", in the second we assigned manually weights depending on the proportion of the two classes in the training set and in the third trial we performed a grid search to choose the best class weights among a predefined grid for candidate class-weights. Each model was trained on the same training set and evaluated on the same test set.

5.6 Results for Problem 4: (M-from-D class)

Although approximately 100 trials were conducted to model the imbalanced binary classification problem of diagnosing memory-related problems, the results obtained were not satisfactory enough. The range of values of each evaluation metric calculated on the test set by applying the approaches described in Section 5.4 is given below: Precision: [0.11789,1], Recall: [0.03125, 0.9375], F1-score: [0.05882,0.46154], AUC-ROC: [0.514839,0.91799]. We notice that the F1-score values are not high, which we expected because F1 is the harmonic mean of precision and recall for which it was observed from the trials that when one had a very high value the other had a very low value. It is worth noting that we wanted a high Recall in order to pay attention on predicting well the people with memory problems (Positive Class). So, we decided to choose the model that gave a Recall $>$ 0.5 with the maximum Precision. The model was the Extra Trees classifier with SMOTE & ENN resampling technique on the training set which gave the following metric values on the test set: Precision: 0.307692, Recall: 0.62500, F1-score: 0.412371, AUC-ROC: 0.890625.

Chapter 6

Conclusions

This thesis focused on the growing prominence of mental health issues worldwide, driven by global challenges like public health emergencies, social inequality, and the climate crisis. Mental disorders had a significant impact on individuals and society, with anxiety and depressive disorders being the most prevalent. The COVID-19 pandemic further worsened mental health, leading to a rise in anxiety and depression cases. Identifying mental health issues involves various methods such as interviews, questionnaires, psychological tests, and diagnostic tools. Early detection is crucial for timely and appropriate care, and questionnaires had emerged as accessible tools for individuals to assess their mental health and seek help. They enable early treatment, potentially saving lives and preventing severe consequences.

The thesis explored the use of questionnaires for diagnosing mental health conditions, specifically focusing on the relationship between depression and memory disorders. Depression affected the quality of life and was often associated with memory impairments. Understanding the link between depression and memory loss is crucial for accurate diagnosis and effective treatment. Memory problems could arise from various causes, including depression, but it was essential to rule out other underlying conditions that contributed to cognitive impairments. Seeking medical evaluation and undergoing cognitive tests helped determine the cause of memory issues and guided appropriate treatment. By utilizing questionnaires like the PRMQ for memory and the SDS questionnaire for depression, the thesis investigated whether these tools could diagnose depression and detect memory-related disorders, respectively. Machine learning techniques were also explored as a way to predict mental health disorders using questionnaire data. Previous studies had shown promise in using machine learning algorithms to identify mental health conditions. These approaches provided valuable insights and potential for automated diagnosis using questionnaire data from one mental health disorder to predict another one.

In practice, the thesis discussed four prediction problems based on the questionnaire data:

- Problem 1: “D-from-M score” focused on predicting the depression score using the responses to the memory PRMQ questionnaire and other demographic and health-related data.
- Problem 2: “D-from-M class” involved predicting the depression class (with or without depression) using the responses the memory PRMQ questionnaire and other demographic and health-related data.
- Problem 3: “M-from-D score” aimed to predict the memory score using the responses to the depression SDS questionnaire and other demographic and health-related data.
- Problem 4: “M-from-D class” focused on predicting the memory class (with or without memory disorders) using the responses to the depression (SDS) questionnaire and other demographic and health-related data.

To address these prediction problems, various statistical methods for feature selection were employed, depending on the data types and target variables. These methods included Pearson Correlation, Spearman Rank Correlation Coefficient, Point Biserial Correlation, Kruskal-Wallis H Non-Parametric Hypothesis Test, Mann-Whitney U Test, Fisher’s Exact Test, and Chi-Square Test of Independence. The modeling process for each problem involved training machine learning models using the selected features and the corresponding target variables. The thesis discussed the steps and methodologies employed for each prediction problem and highlighted the proportions of the different classes in the target variables to provide insights into class imbalance issues. Below we present the results of the machine learning applications for each problem, the final model-models chosen to model it and their performance results.

6.1 Results for the Problem 1: D-from-M score

Based on the provided data and analysis of the performance measures for different ML models used to predict the depression (ZUNG) score, the “Weighted Voting Regression (3)” model stood out. The “Weighted Voting Regression (3)” model consistently demonstrates strong performance across multiple evaluation metrics for predicting depression (ZUNG) score. It achieves the lowest RMSE (CV) and MAE (CV) scores, indicating its accuracy in capturing the underlying patterns in the data. Additionally, it performs well in terms of SMAPE (CV), further highlighting its effectiveness. Moreover, it shows competitive performance in terms of RMSE (test), MAE (test), and SMAPE (CV). Below we give detailed information about the performance metrics of the ”Stacking Regression (3)” model:

- RMSE (CV) and RMSE (test): The model achieves an RMSE (CV) value of 2.35886, indicating that, on average, its predictions deviate from the actual values by approximately 2.36 units during cross-validation. On the test set, the RMSE value is 5.53651, suggesting that the model's predictions deviate by around 5.54 units from the true values.
- MAE (CV) and MAE (test): The model achieves an MAE (CV) value of 4.41478 during cross-validation, meaning that, on average, its predictions have an absolute difference of approximately 4.41 units compared to the true values. On the test set, the MAE value is 4.42447, indicating an average absolute difference of around 4.42 units between the model's predictions and the actual values. The MAE metric provides insights into the magnitude of the model's errors.
- SMAPE (CV) and SMAPE (test): The model achieves an SMAPE (CV) value of 10.421 during cross-validation, suggesting that, on average, the model's predictions have a relative error of approximately 10.42% compared to the true values. On the test set, the SMAPE value is 10.27, indicating a relative error of around 10.27%. The SMAPE metric provides a measure of the percentage error in the predictions.
- MAAPE (CV) and MAAPE (test): The model achieves an MAAPE (CV) value of 0.10532 during cross-validation, indicating that, on average, the model's predictions have an absolute percentage difference of approximately 10.53% compared to the true values. On the test set, the MAAPE value is 0.10263, suggesting an average absolute percentage difference of around 10.26% between the model's predictions and the actual values. The MAAPE metric provides a measure of the percentage error in the predictions.
- MSE (CV) and MSE (test): The model achieves an MSE (CV) value of 31.0772 during cross-validation, indicating the average squared difference between the predicted and true values. On the test set, the MSE value is 30.65297, suggesting a similar average squared difference between the model's predictions and the actual values. The MSE metric provides insights into the magnitude of the errors made by the model.
- R^2 (CV) and R^2 (test): The model achieves an R^2 (CV) value of 0.65543 during cross-validation, indicating that approximately 65.54% of the variance in the target variable can be explained by the model's predictions. On the test set, the R^2 value is 0.64345, suggesting that around 64.35% of the variance in the target variable is explained by the model's predictions. The R^2 metric provides a measure of how well the model fits the data and captures the underlying patterns.

Overall, the "Weighted Voting Regression (3)" model performs reasonably well based on the provided metrics, including low errors and high coefficient of determination (R^2). It achieves a relatively low MAAPE, indicating that its predictions have a relatively small absolute percentage difference from the true values. The MSE values are also moderate, indicating that the

model's predictions have a reasonable squared difference from the actual values. Additionally, the R^2 values indicate that a significant portion of the variance in the target variable is captured by the model's predictions. It achieves relatively low RMSE, MAE, and SMAPE values during cross-validation and on the test set, indicating its ability to make accurate predictions on both seen and unseen data. Thus, it demonstrates good generalization ability to unseen data with similar performance between cross-validation and test set metrics. Therefore, it is a strong candidate for predicting depression (ZUNG) score and presents a promising approach for solving the problem: D-from-M score. It's also worth noting that these comments are based solely on the provided data and metrics. Further analysis, comparison with other models, and consideration of additional factors such as interpretability, computational efficiency, and domain-specific requirements would be necessary to make a more comprehensive assessment of the model's practical utility.

6.2 Problem 2: D-from-M class

Based on the machine learning analysis of Problem 2: D-from-M class, it is evident that the LightGBM classifier and AdaBoost classifier consistently outperform other models in predicting the Depression (ZUNG) class. These models demonstrate strong performance across multiple metrics, including recall, accuracy, precision, F1 score, AUC-ROC score, and balanced accuracy. Also, models like the LightGBM classifier, Gradient Boosting classifier, and AdaBoost classifier exhibit better generalization capabilities, indicating their ability to perform well on unseen data in contrast to the presence of potential overfitting or underfitting in other models. For instance, the Decision tree classifier and KNN classifier may be prone to overfitting or underfitting, as suggested by noticeable differences between cross-validation and test set performance. Below, there are some analytical notes on the classification performance of the LightGBM classifier and AdaBoost classifier for the prediction of the Depression (ZUNG) class:

- Recall (CV) and Recall (test): The LightGBM classifier achieves a recall (CV) value of 0.77605 during cross-validation, indicating that it correctly identifies approximately 77.61% of the positive instances in the dataset. On the test set, the recall value is 0.79125, suggesting that the model correctly identifies around 79.13% of the positive instances. Similarly, the AdaBoost classifier achieves a recall (CV) value of 0.79019 during cross-validation and a recall (test) value of 0.77778 on the test set. These values indicate that both models have a relatively high ability to identify positive instances correctly. Both of them have recall values of around 0.78-0.79 in cross-validation and test set, indicating their ability to capture most positive instances.

- Accuracy (CV) and Accuracy (test): The LightGBM classifier achieves an accuracy (CV) value of 0.80576 during cross-validation, indicating that it correctly classifies approximately 80.58% of the instances in the dataset. On the test set, the accuracy value is 0.79341, suggesting that the model correctly classifies around 79.34% of the instances. Similarly, the AdaBoost classifier achieves an accuracy (CV) value of 0.81139 during cross-validation and an accuracy (test) value of 0.78743 on the test set. These values indicate that both models have a relatively high overall classification accuracy or overall correctness in predictions (Accuracy).

- Precision (CV) and Precision (test): The LightGBM classifier achieves a precision (CV) value of 0.78468 during cross-validation, indicating that approximately 78.47% of the instances predicted as positive are actually true positives. On the test set, the precision value is 0.75563, suggesting that around 75.56% of the instances predicted as positive are true positives. Similarly, the AdaBoost classifier achieves a precision (CV) value of 0.78622 during cross-validation and a precision (test) value of 0.75244 on the test set. These values indicate that both models have a relatively good precision, meaning they have a relatively low rate of false positives. Thus, when it comes to correctly identifying positive cases out of all predicted positive instances (Precision), both classifiers demonstrate good precision scores.

- F1 (CV) and F1 (test): The F1 score is a measure that combines precision and recall, providing an overall assessment of the model's ability to balance between correctly identifying positive instances and minimizing false positives. The LightGBM classifier achieves an F1 score (CV) of 0.77949 during cross-validation, indicating a good balance between precision and recall. On the test set, the F1 score is 0.77303, suggesting a similar balance in performance. The AdaBoost classifier also performs well with an F1 score (CV) of 0.78739 during cross-validation and an F1 score (test) of 0.7649 on the test set. These values indicate that both models achieve a relatively good balance between precision and recall.

- AUC-ROC (CV) and AUC-ROC (test): The Area Under the Receiver Operating Characteristic curve (AUC-ROC) is a popular metric for evaluating the overall performance of a classification model across different probability thresholds. The LightGBM classifier achieves an AUC-ROC score (CV) of 0.80308 during cross-validation, indicating a high ability to distinguish between positive and negative instances. On the test set, the AUC-ROC score is 0.79319, suggesting that the model maintains good performance on unseen data. Similarly, the AdaBoost classifier achieves an AUC-ROC score (CV) of 0.80966 during cross-validation and an AUC-ROC score (test) of 0.78646 on the test set. These values indicate that both models have a relatively good ability to discriminate between positive and negative instances.

- Balanced accuracy (CV) and Balanced accuracy (test): The balanced accuracy takes into account the imbalance between the number of positive and negative instances in the dataset and provides a fair assessment of the model's performance. The LightGBM classifier achieves a balanced accuracy (CV) of 0.80308 during cross-validation, indicating a good overall accuracy considering the class imbalance. On the test set, the balanced accuracy is 0.79319, suggesting that the model maintains similar performance on unseen data. The AdaBoost classifier also performs well with a balanced accuracy (CV) of 0.80966 during cross-validation and a balanced accuracy (test) of 0.78646 on the test set. These values indicate that both models have a relatively good overall accuracy, accounting for the class imbalance in the dataset.

Overall, both the LightGBM classifier and the AdaBoost classifier show promising performance in predicting the Depression (ZUNG) class based on the provided metrics. They achieve relatively high recall values, indicating their ability to correctly identify positive instances. Additionally, they demonstrate good accuracy values, reflecting their overall classification performance. Furthermore, the precision values suggest that both models have a low rate of false positives, meaning they are conservative in predicting positive instances. They achieve good F1 scores, indicating a balanced trade-off between precision and recall. Additionally, they perform well in terms of AUC-ROC, showcasing their ability to distinguish between positive and negative instances. Furthermore, the balanced accuracy that both models have a good overall accuracy, accounting for the class imbalance in the dataset. Also, they show consistent performance across different evaluation metrics, indicating their reliability and effectiveness in classifying instances.

Based on these observations, it is recommended to consider either the LightGBM classifier or the AdaBoost classifier for predicting the Depression (ZUNG) class. The LightGBM classifier performs slightly better in terms of precision on the test set, indicating its ability to minimize false positives. On the other hand, the AdaBoost classifier has a slightly higher recall on the test set, suggesting it captures a higher proportion of positive instances. The final choice may depend on specific requirements, such as the importance of precision or recall, and the need for higher accuracy or generalization to new data. It is recommended to further evaluate and compare these classifiers on additional datasets to gain more insights into their performance. The final decision should take into account other factors such as computational requirements, interpretability, and specific project requirements. It's important to note that these comments are based solely on the provided data and metrics. Further analysis, comparison with other models, and consideration of additional factors such as computational efficiency, interpretability, and specific requirements of the classification task would be necessary to make a more comprehensive assessment of the models' practical utility.

6.3 Problem 3: M-from-D score

Based on the provided data and analysis of the performance measures for different ML models used to predict the memory (PRMQ) score, it can be concluded that the Stacking Regression (1) model is the most suitable for predicting the memory (PRMQ) score compared to the other models tested. This conclusion is based on the following factors:

1. Performance across multiple metrics: Stacking Regression (1) consistently performs well across various evaluation metrics, including MAAPE, MSE, and R^2 scores. It has the best performance in 5 out of the 12 calculated metrics, indicating its overall effectiveness.
2. Generalization capability: Stacking Regression (1) exhibits good generalization capabilities as its performance on the test set is slightly better or comparable to its performance on cross-validation. This suggests that the model can effectively generalize to unseen data.
3. Comparison with other models: Stacking Regression (1) outperforms other models, including different variations of stacking regression models, ensemble models, and weighted voting regression models, in terms of MAAPE, MSE, and R^2 scores.
4. Stability and reliability: Stacking Regression (1) consistently demonstrates good performance across different metrics, indicating stability and reliability in predicting the memory (PRMQ) score.
5. Therefore, based on its overall performance, generalization capability, and comparison with other models, the Stacking Regression (1) model is recommended for accurately predicting the memory (PRMQ) score.

Below we present detailed comments on the performance of the Stacking Regression (1) model:

- RMSE (CV) and RMSE (test): The Stacking Regression (1) model achieved an RMSE of 2.76952 during cross-validation, indicating that it can predict the target variable with an average deviation of approximately 2.76952 units from the actual values. On the test set, the RMSE was slightly higher at 7.6284, suggesting that the model's predictions had a slightly higher average deviation of approximately 7.6284 units from the actual values. Overall, the model performed well in terms of RMSE, but there is a larger deviation on the unseen test data.
- MAE (CV) and MAE (test): The Stacking Regression (1) model achieved an MAE of 6.01643 during cross-validation, indicating that, on average, its predictions had an absolute deviation of approximately 6.01643 units from the actual values. On the test set,

the MAE improved slightly to 5.94725, suggesting that the model's predictions had a slightly lower absolute deviation of approximately 5.94725 units from the actual values. The model's MAE values indicate reasonably accurate predictions, but there is still room for improvement.

- SMAPE (CV) and SMAPE (test): The Stacking Regression (1) model achieved an SMAPE of 32.715% during cross-validation, indicating that, on average, its predictions had a percentage error of approximately 32.715% compared to the actual values. On the test set, the SMAPE improved slightly to 31.49%, suggesting a slightly lower percentage error of approximately 31.49%. The model's SMAPE values indicate that the predictions had a moderate level of error, but further improvements are desirable.

- MSE (CV) and MSE (test): The Stacking Regression (1) model achieved an MSE of 59.12159 during cross-validation, indicating that, on average, the squared differences between its predictions and the actual values amounted to approximately 59.12159. On the test set, the MSE was slightly lower at 58.19254, indicating a slightly smaller average squared difference of approximately 58.19254. The model's MSE values suggest that there is still room for improvement to reduce the errors between predictions and actual values.

- R^2 (CV) and R^2 (test): The Stacking Regression (1) model achieved an R^2 of 0.52203 during cross-validation, indicating that it can explain approximately 52.203% of the variance in the target variable based on the independent variables. On the test set, the R^2 improved slightly to 0.55923, suggesting an increased ability to explain approximately 55.923% of the variance. These R^2 values indicate that the Stacking Regression model captures a moderate amount of the target variable's variability, but there is potential for further improvement in explaining the variance.

Overall, the Stacking Regression (1) model shows promise in making predictions for memory (PRMQ) score, as indicated by the performance metrics. However, there is room for improvement, particularly in reducing the average deviation, absolute deviation, and percentage error between the predicted values and the actual values. Additionally, enhancing the model's ability to explain more variance in the target variable would be beneficial.

6.4 Problem 4: M-from-D class

Although approximately 100 trials were conducted to model the imbalanced binary classification problem of diagnosing memory-related problems, the results obtained were not satisfactory enough. The range of values of each evaluation metric calculated on the test set by applying the approaches described in Section 5.4 is as follows:

- Precision: Ranging from 0.11789 to 1
- Recall: Ranging from 0.03125 to 0.9375
- F1-score: Ranging from 0.05882 to 0.46154
- AUC-ROC: Ranging from 0.514839 to 0.91799

Based on these results, it can be observed that the overall performance of the models in diagnosing memory-related problems was not satisfactory, as indicated by the relatively low values for precision, recall, and F1-score. Specifically for the F1-score, it was observed for its values that were not high, which was expected because the F1-score is the harmonic mean of precision and recall. It was noticed from the trials that when one had a very high value, the other had a very low value, indicating a trade-off between precision and recall. It is worth noting that we wanted a high Recall in order to pay attention to predict well the people with memory problems (Positive Class). Among the models considered, the Extra Trees classifier with SMOTE & ENN resampling technique was chosen as the best option. This model achieved the following performance metric values on the test set:

- Precision: 0.307692. Precision measures the accuracy of positive predictions made by the model. In this case, the model achieved a precision of almost 0.31, indicating that almost 31% of the positive predictions were correct and a significant proportion (69%) of the positive predictions made by the model were incorrect. This indicates a potential issue with the model's ability to accurately classify positive samples.
- Recall: 0.625. It measures the proportion of actual positive samples that were correctly identified by the model. In this case, the model achieved a recall of 0.625, indicating that 62.5% of the positive samples were detected.
- F1-score: 0.412371. F1-score is the harmonic mean of precision and recall, providing a balanced measure of a model's performance. The achieved F1-score of 0.412371 indicates a moderate trade-off between precision and recall, highlighting the need for improvement in both areas.
- AUC-ROC: 0.890625. AUC-ROC (Area Under the Receiver Operating Characteristic Curve) is a metric that assesses the overall performance of a classification model across various classification thresholds. Despite the moderate precision, recall, and F1-score, the AUC-ROC value of 0.890625 indicates that the model exhibits relatively good discriminatory ability in distinguishing between positive and negative samples across different classification thresholds.

In summary, the performance of the Extra Trees classifier with SMOTE & ENN resampling technique in diagnosing memory-related problems through depression questionnaire data, as well as demographic and other health-related data, was not very satisfactory. While it showed relatively better results in terms of the Recall compared to other models considered, its classification performance on the imbalanced dataset was sub-optimal, as indicated by the low precision and F1-score. Therefore, further optimization of the Extra Trees classifier with SMOTE & ENN resampling technique is necessary. To improve the classifier's performance in an imbalanced task, several approaches can be explored. These include parameter tuning, considering additional features, applying different resampling techniques to address class imbalance, or incorporating additional data sources to enhance the model's predictive capabilities. Although feature selection, parameter tuning, and various resampling techniques have been applied in the current study, focusing on data-level approaches such as data collection or incorporating more data sources with less class imbalance may yield better results. Regular monitoring and updating of the model with new data are crucial to ensure adaptability and continued performance improvement. Additionally, conducting thorough analysis and domain-specific investigations into potential sources of bias or limitations can provide valuable insights for refining the model.

6.5 Summary results for predicting memory disorders and future work

In conclusion, the results obtained from predicting memory disorders based on responses to the depression SDS questionnaire were not satisfactory enough. However, we remain optimistic about the potential to achieve better outcomes in future studies. It is widely recognized that individuals with depression often experience concurrent issues with their memory, and we firmly believe that this relationship can be effectively verified through the application of a machine learning approach. One of the main challenges encountered in this study was the presence of class imbalance within the memory disorder category. This imbalance likely contributed to the suboptimal performance of the predictive model. Addressing this class imbalance should be a primary focus in future research endeavors to improve the accuracy and reliability of our predictions.

To overcome this limitation, we propose exploring various strategies to rectify the class imbalance issue. This may involve actively seeking a larger sample size of individuals with memory disorders to participate in the study. By including a more diverse and representative range of participants, we can mitigate the impact of class imbalance and enhance the model's ability to accurately identify and classify memory-related problems associated with depression. Moreover, further refinement of the machine learning approach is essential. This can include

exploring alternative algorithms, optimizing model parameters, and incorporating additional relevant features or data sources to strengthen the predictive capabilities. By employing a comprehensive and systematic approach, we can unlock the full potential of machine learning in predicting memory disorders.

In terms of future work, we propose a replication of the study using alternative questionnaires for depression and memory assessment, excluding the PRMQ questionnaire for memory disorders assessment and the depression SDS questionnaire for depression assessment. This replication will allow for a more comprehensive evaluation of the relationship between depression and memory, aiming to enhance the validity and generalizability of the findings. By incorporating a diverse set of validated questionnaires, we can explore various aspects of depression and memory functioning, gaining a more nuanced understanding of their interconnectedness. The replication study holds significant potential in advancing our understanding of the complex relationship between depression and memory. Through the utilization of machine learning techniques and a wider range of questionnaires, the predictive model can be refined, potentially revealing novel insights into the specific cognitive mechanisms underlying memory impairments in individuals with depression.

In conclusion, our proposed replication study, utilizing different questionnaires for depression and memory assessment, aims to improve the results and validate the link between depression and memory problems. Addressing class imbalance and expanding the sample to include more individuals with memory difficulties will likely lead to enhanced performance in predicting memory-related issues associated with depression. This replication study, combined with the exploration of alternative questionnaires, has the potential to deepen our understanding of the intricate relationship between depression and memory, ultimately benefiting clinical assessment, intervention, and support for individuals affected by these conditions.

6.6 Summary results for predicting depression and future work

This study has unveiled a remarkable predictive approach that can significantly contribute to the field of depression diagnosis and bring forth substantial insights into the link between depression and mental health assessment in general. With the utilization of the responses data to the PRMQ memory questionnaire, alongside additional demographic and health-related questions, we can train a ML model (“Weighted Voting Regression (3)”) that achieves a 0.79 accuracy rate in predicting depression. This remarkable accuracy rate of 0.79 demonstrates the potential of this predictive model in identifying individuals at risk of depression. The implications of such accurate predictions are significant. Early detection of depression allows for timely interventions, support, and appropriate treatment, leading to improved outcomes and

enhanced mental well-being. Additionally, the findings highlight the link between depression and memory while providing insights into the intricate relationship between these two factors. The following are some of the key insights gained from the findings:

(a) The PRMQ memory questionnaire, designed to assess memory problems, has proven to be a powerful tool in identifying individuals at risk of depression. By analyzing responses to specific memory-related questions, researchers have gained a deeper understanding of how memory disturbances intertwine with depressive symptoms. The questionnaire's remarkable accuracy rate signifies its potential as a reliable screening method, facilitating early intervention and support for those vulnerable to depression.

(b) Moreover, by incorporating demographic and other health-related questions alongside the PRMQ memory questionnaire, researchers have unraveled additional layers of complexity in the depression-memory link. Factors such as age, gender, education, and medical history have been found to influence the relationship between memory and depression. These findings shed light on the multifaceted nature of depressive disorders, emphasizing the importance of considering various contextual variables in mental health assessments.

The enhanced understanding of the connection between depression and memory offered by this research carries significant implications for both clinical practice and future research endeavors. Clinicians can utilize the PRMQ memory questionnaire, combined with demographic and health-related inquiries, to augment their diagnostic accuracy and tailor treatment plans more effectively. Additionally, researchers can delve deeper into the underlying mechanisms driving the depression-memory association, opening new avenues for therapeutic interventions and prevention strategies.

In conclusion, the remarkable 0.79 accuracy achieved in predicting depression through the PRMQ memory questionnaire, along with demographic and health-related questions, marks a pivotal advancement in mental health assessment. This model holds great potential in identifying individuals at risk of depression, facilitating early interventions, and improving mental health outcomes. Moreover, these findings not only strengthen the link between depression and memory but also provide invaluable insights into the complex interplay between these factors. By harnessing this knowledge, we can strive towards more targeted interventions, improved patient outcomes, and a better understanding of the intricate nature of mental health.

For future work, we suggest the continued refinement and application of this predictive model,

so it can hold promise for advancing the field of depression diagnosis. By leveraging the insights gained from the predictive performance of the model, we can inform the development of targeted interventions, personalized treatment plans, and preventative strategies, ultimately improving the lives of individuals affected by depression. Furthermore, we recommend conducting a replication of the study by employing different questionnaires to evaluate depression and memory, excluding SDS and PRMQ for these specific purposes. This approach would enable an examination of the performance of machine learning models when applied to datasets derived from diverse sources. The outcomes of these models may yield comparable results, which would further solidify the reliability of the conclusions drawn. Conversely, the alternative questionnaires may produce even more favorable outcomes, indicating the effectiveness of utilizing them for training machine learning models in the prediction of depression. Conversely, if the results prove to be inferior, it would underscore the necessity of selecting PRMQ and SDS as the most suitable questionnaires for our study.

Appendices

Appendix A

Statistical tests for Feature Selection

A.1 Statistical tests for applying feature selection on problems that predict a numeric target variable

A.1.1 Numeric target variable and numeric input variable: Pearson Correlation

Pearson correlation is a type of correlation used only for numerical variables. It is a metric that measures the extent of linear correlation between two numerical variables. Pearson correlation coefficient ranges between -1 and 1.

- A number closer to 0 means weaker correlation. When the value is exactly 0, there is no correlation.
- A number closer to 1 means stronger positive correlation
- A number closer to -1 means stronger negative correlation

Pearson correlation formula:

$$r = \frac{\sum_{i=1}^{N}(x_i - mean(x))(y_i - mean(y))}{\sqrt{\sum_{i=1}^{N}(x_i - mean(x))^2 \sum_{i=1}^{N}(y_i - mean(y))^2}}$$

The $p - value$ for testing the significance of the correlation between x and y variables can be computed as follows:

1. By calculating the t statistic in the following way:

$$t = \frac{r}{\sqrt{1 - r^2}}\sqrt{n - 2}$$

Then, the corresponding p-value is determined using t distribution table for $df = n - 2$, where n is the number of observations in x and y variables.

In case the p-value is less than 5%, the linear correlation between x and y is statistically significant.

2. Or by using the Pearson 's correlation table for the degrees of freedom: $df = n - 2$.

A.1.2 Numeric target variable and ordinal input variable: Spearman Rank Correlation Coefficient

The Spearman's rank-order correlation is the non-parametric analog of the Pearson 's correlation. As a nonparametric correlation measurement, it can also be used with ordinal data. Spearman's correlation coefficient, r_s, determines the strength and direction of the monotonic relationship between two variables rather than the strength and direction of the linear relationship between your two variables. It can be considered as a statistical test of independence, because the Spearman rank correlation coefficient is used as a hypothesis test to determine if there is a relation between two variables. In the proposed master thesis, it will be used to evaluate the relationship between a numeric variable and an ordinal variable.

As a correlation measurement between two variables, Spearman's correlation coefficient satisfies the following:

1. It ranges between -1 and $+1$.
2. X and Y variables are positively correlated, if the value of the correlation coefficient is positive. They are perfectly positively correlated, if the correlation coefficient is equal to $+1$.
3. X and Y variables are negatively correlated, if the value of the correlation coefficient is negative. They are perfectly negatively correlated, if the correlation coefficient is equal to -1.
4. X and Y are not correlated, if the correlation coefficient is close to zero.

Mathematical Formula:

Let $(X_1, X_2, \ldots, X_n)$ and $(Y_1, Y_2, \ldots, Y_n)$ be two samples of size n of two variables X and Y. We define R_{X_i} as the rank of X_i for $i = 1, 2, \ldots, n$ compared to the other values of the X

sample. More specifically, $R_{X_i} = 1$, if X_i is the smallest value of X, $R_{X_i} = 2$, if X_i is the second smallest value, etc., until $R_{X_i} = n$, if X_i is the largest value of X. In the same way, R_{Y_i} symbolizes the rank of Y_i , for $i = 1, 2, \ldots, n$.

The Spearman rank correlation coefficient is computed as follows:

$$r_s = 1 - \frac{6 \sum_{i=1}^{n} d_i^2}{n(n^2 - 1)}$$

where $d_i = R_{X_i} - R_{Y_i}$

Hypothesis Testing:

H_0: X and Y are mutually independent.

H_1: There is either a positive or a negative correlation between X and Y.

Decision rule rejects H_0 at the significance level α if

$$r_s > t_{n-1,1-\frac{\alpha}{2}} \quad \text{or} \quad r_s < t_{n,\frac{\alpha}{2}}$$

where t is the critical value of the test given by the Spearman table.

A.1.3 Numeric target variable and binary input variable: Point biserial correlation

The relationship between a continuous variable and a binary variable can be assessed by a correlation coefficient which is called point biserial correlation and symbolized as r_{pb}. The point biserial correlation cannot be used to analyze categorical variables with more than two categories. The Point-biserial is a correlation coefficient, so it ranges between -1 and $+1$.

- -1 implies that two variables are perfectly negatively correlated.
- 0 implies that two variables are not correlated.
- 1 implies that two variables are perfectly positively correlated.

Mathematical formula:

$$r_{pb} = \frac{\bar{Y}_1 - \bar{Y}_0}{\bar{s}_y} \sqrt{\frac{n_1 n_0}{n(n-1)}},$$

where

- Y_0 is the mean of the numerical observations coded as 0 (first category of the binary variable).
- Y_1 is the mean of the numerical observations coded as 1 (second category of the binary variable).
- n_0 and n_1 are number of observations coded 0 and 1 respectively.
- $n = n_0 + n_1$ is total number of observations.
- $\bar{s}_y$ is the standard deviation of all the numeric observations

$$\bar{s}_y = \frac{\sum_{i=1}^{N} y_i^2 - \frac{(\sum_{i=1}^{n} y_i)^2}{n}}{n-1}$$

Hypothesis Testing:

H_0: Continuous variable and binary variable are not associated

H_1: Continuous variable and binary variable are associated

Or equivalently,

H_0: Mean values of the numerical observations coded as 0 and 1 are equal

H_1: Mean values of the numerical observations coded as 0 and 1 are not equal

Determining the significance of the test:

The point biserial and the independent samples t-test are very similar. In fact, the p-value which determines the significance of a point biserial correlation coefficient is exactly the same as the p-value computed by an independent samples t-test given that the two tests are done on the same data. Thus, testing if a value of r_{pb} differs significantly from zero is equivalent to testing if the difference between the means of the two groups is statistically significant.

This in practice means that an independent groups t test with $(n-2)$ degrees of freedom (df) or a one-way analysis of variance (ANOVA) with two levels can be used to test whether r_{pb} is nonzero.

We can see below how the t-statistic for comparing two independent groups and r_{pb} are related.

$$t = \sqrt{n-2}\frac{r_{pb}}{\sqrt{1-r_{pb}^2}}$$

A.1.4 Numeric target variable and nominal input variable: Kruskal–Wallis H Non Parametric Hypothesis Test

The Kruskal-Wallis H test is a rank-based nonparametric (distribution free) test that we use to investigate if there are statistically significant differences on a continuous or ordinal dependent variable by a categorical independent variable (with two or more groups). It can be characterized as a nonparametric version of the one-way analysis of variance (one-way ANOVA) which should meet specific assumptions to be applied. One of these assumptions is that the dependent variable follows normal distribution and another assumption is that the variance of dependent variable in the different groups should be approximately the same. However, in most cases these assumptions do not hold in practice. Kruskal-Wallis Test does not have these assumptions. Thus, we can apply the Kruskal-Wallis test either to continuous or ordinal dependent variables.

Hypothesis Testing:

H_0 : Population medians are equal

H_1 : Population medians are not equal

Or equivalently,

H_0 : There are not statistically significant differences between the groups of the independent categorical variable on the continuous or ordinal dependent variable.

H_1 : There are statistically significant differences between the groups of the independent categorical variable on the continuous or ordinal dependent variable.

Determining the significance of the test:

In order to apply Kruskal-Wallis test, the observations from the k different groups should be gathered into one sample. Then, the observations in the combined sample should be ranked from lowest to highest value.

The test statistic for the Kruskal Wallis test is defined as follows:

$$H = \frac{12}{n(n+1)} \sum_{i=1}^{c} \frac{T_i^2}{n_i} - 3(n+1)$$

Where

- n=total number of observations from all samples.
- c=number of samples.
- $T_i =$ sum of ranks in the i sample.
- $n_i =$ size of the i-th sample.

Then, the value of the statistic H should be compared to the χ^2 distribution with $c-1$ degrees of freedom, where c is the number of groups. In case the critical χ^2 value is less than the H statistic, the null hypothesis is rejected. On the other hand, there is enough evidence to accept the null hypothesis.

A.2 Statistical tests for applying feature selection on problems that predict a binary target variable or a class

A.2.1 Binary target variable and binary input variable: Fisher 's Exact Test

Fisher's exact test is a statistical hypothesis test used to determine if there is an association between two binary categorical variables. Thus, data can be represented in a $2x2$ contingency table. In other words, it is a statistical test to investigate whether there is a dependence between the two variables or equivalently if there are differences in the proportions of one variable depending on the value of the other variable. Fisher's Exact test is more suitable for samples with small number of observations (regularly under 1.000 records), since it does not dependent on distributional assumptions.

Hypothesis Testing:

$$H_0: \text{ The two binary variables are not associated}$$
$$H_1: \text{ The two binary variables are associated}$$

There is no formula for the test statistic of Fisher's exact test instead of other statistical tests. Indeed, hypergeometric distribution is used to calculate directly the p-value of Fisher's exact test.

Methodology to perform Fisher's exact test:

Firstly, we create a $2x2$ contingency table with the cell frequencies represented by $f_{1,1}$, $f_{1,2}$, $f_{2,1}$ and $f_{2,2}$:

	category 1	category 2
group 1	$f_{1,1}$	$f_{1,2}$
group 2	$f_{2,1}$	$f_{2,2}$

Secondly, we calculate the marginal totals and the grand total n:

	category 1	category 2	
group 1	$f_{1,1}$	$f_{1,2}$	$f_{1,1} + f_{1,2}$
group 2	$f_{2,1}$	$f_{2,2}$	$f_{2,1} + f_{2,2}$
	$f_{1,1} + f_{2,1}$	$f_{1,2} + f_{2,2}$	$f_{1,1} + f_{1,2} + f_{2,1} + f_{2,2} = n$

It has been proven that under the assumption of the null hypothesis and if the values of the marginal totals are fixed, the probability of observing the previous table is described by the hypergeometric distribution described bellow:

$$\frac{\binom{f_{1,1}+f_{1,2}}{f_{1,1}}\binom{f_{2,1}+f_{2,2}}{f_{2,1}}}{\binom{n}{f_{1,1}+f_{2,1}}} = \frac{(f_{1,1}+f_{1,2})!(f_{2,1}+f_{2,2})!(f_{1,1}+f_{2,1})!(f_{1,2}+f_{2,2})!}{f_{1,1}!f_{1,2}!f_{2,1}!f_{2,2}!n!}$$

It is observed that in case the marginal totals are known, there is only one degree of freedom, because only a frequency value, for example $f_{1,1}$, is sufficient to find all the other frequencies. Therefore, the previous probability depends only on $f_{1,1}$.

In order to calculate the p-value for a Fisher's exact test, all possible tables of non-negative integers with the same row and column totals as the original table should be found and afterwards the probability of each such table should be computed. After, in case we should perform a two tailed test, all the probabilities of every table that has a probability lower than or the same as that of the observed table should all be summed together.

If we perform a one-tailed test, only the tables that are more extreme than the observed table should be accounted but only in one direction:

- In case $f_{1,1} < f_{1,2}$, the probabilities of all tables that have a value lower than or equal to $f_{1,1}$ in the upper-left corner should be summed, including the observed table.
- In case $f_{1,1} > f_{1,2}$, the probabilities of all tables that have a value greater than or equal to $f_{1,1}$ in the upper-left corner should be summed, including the observed table.
- In case $f_{1,1} = f_{1,2}$, we can choose both of the previous options. Whatever the choice, the result will be the same.

A.2.2 Binary target variable and Nominal input variable: Chi-square test of independence

The Chi-square test of independence is a statistical test applied when we need to investigate if two categorical variables are likely to be related or not. The assumptions for the Chi-square test are the following ones:

1. Two categorical variables (Two or more categories for each variable.)
2. Independence of observations.
3. Relatively large sample size.
    - Each cell should have expected frequencies at least 1.
    - Expected frequencies should be at least 5 for the majority (80%) of the cells.

Hypothesis Testing:

$$\begin{aligned} H_0: &\quad \text{The two variables are independent.} \\ H_1: &\quad \text{The two variables are not independent.} \end{aligned}$$

Determining the significance of the test:

Formula for the test statistic:

$$\chi^2 = \sum_{i=1}^{R} \sum_{j=1}^{C} \frac{(o_{ij} - e_{ij})^2}{e_{ij}}$$

where

- o_{ij} is the observed cell count in the i^{th} row and j^{th} column of the table
- e_{ij} is $X^2$the expected cell count in the i^{th} row and j^{th} column of the table, computed as

$$e_{ij} = \frac{row\ \ i\ \ total * col\ \ j\ \ total}{grand\ \ total}$$

  The quantity $(o_{ij} - e_{ij})$ is sometimes referred to as the residual of cell (i, j), denoted as r_{ij}.

The χ^2 statistic is tested against the critical value from the X^2 distribution table with degrees of freedom $df = (R-1)(C-1)$.

In case, χ^2 value $\geq$ critical χ^2 value, the null hypothesis is rejected.

A.3 Binary target variable and ordinal input variable: Mann-Whitney U test

The Mann-Whitney U test is a nonparametric statistical hypothesis test used to determine if the differences between two independent groups of a dependent variable which is either ordinal or continuous, but not normally distributed, are statistically significant or not. In other words, the test assess the association between a continuous or ordinal dependent variable and a binary independent variable. Mann-Whitney U test has the following assumptions:

1. The dependent variable should be either ordinal or continuous.
2. The independent variable should have two categorical and independent groups.
3. Independent observations.

Hypothesis Testing:

H_0 : The probability that a randomly drawn observation from one group is larger than a randomly drawn observation from the other is equal to 0.5

H_1 : The probability that a randomly drawn observation from one group is larger than a randomly drawn observation from the other is not equal to 0.5

Mann-Whitney U-statistic:

The computation of Mann-Whitney U-statistic depends on the type of method:

- Direct method:

  Firstly, we should order separately the two data sets in ascending order. Then, we compute the statistic U_1 by summing up the number of times each observation in sample A is exceeded by an observation in sample B. It is practical to denote the sample with the fewer observations as A, but not necessary. Afterwards, U_2 is calculated by $(n_A n_B - U_1)$. The direct method is only really suitable for samples with small sizes.

- Indirect method:

For large sample sizes, the quantities U_1 and U_2 are easier to be computed by the indirect method.

$$U_1 = S_A - \frac{n_A(n_A+1)}{2}$$
$$U_2 = S_B - \frac{n_B(n_B+1)}{2}$$

Where

- U_1 and U_2 are the two alternative U statistics
- n_A is the size of sample A
- n_B is the size of sample B,
- S_A and S_B are the sums of ranks for each sample after gathering the data of the two samples into a single sample

In both cases the test statistic U equals to the minimum of U_1 and U_2. The maximum value of U_1 and U_2 is symbolized as U'. Rather than computing both U_1 and U_2 from the previous formula, U_2 can be calculated as follows: $U_2 = n_A n_B - U_1$

Determining the significance of the statistic U:

The critical value for the significance of U can be obtained by the published tables of U (e.g. given by Siegel (1956)). In case the computed U statistic is less than the corresponding critical value, it is statistically significant. Also, the exact p-value can be easily obtained using a software.

Normal approximation:

In case the number of observations in sample A or the number of observations in sample B are more than twenty, we can use the normal approximation. Normal approximation was also used in the previous years when ties presented as published tables were only accurate for data without ties. (The term tie is utilized in association with rank order statistics. Tied data are observations having the same value that forbids the assignment of unique rank numbers to the data. In order to solve this problem, tied data are assigned to the average of their hypothetical ranks).

Untied data:

$$z = \frac{S_A - n_A(N+1)/2}{\sqrt{(n_A n_B (N+1))/12}}$$

Tied data:

$$z = \frac{S_A - n_A(N+1)/2}{\sqrt{\frac{n_A n_B}{N(N-1)} \sum R^2 - \frac{(N+1)^2 n_A n_B}{4(N-1)}}}$$

Where

- S_A is the sum of ranks for sample A,
- n_A is the size of sample A,
- n_B is the size of sample B,
- $N = n_A + n_B$ the total number of observations,
- ΣR^2 is the sum of all the squared ranks.

Finally, z is compared to the standard normal deviation Z in order to determine if the null hypothesis will be rejected or not.

Nowadays accurate tests are also provided by synchronous software for tied data.

A.3.1 Binary target variable and Numerical input variable: Point biserial correlation

In order to test the association between a binary target variable and a numerical input variable, we use the Point biserial correlation of Section A.1.3.

Appendix B

Machine Learning models

B.1 Hyperparameters of classification models for the Problem 2 (D-from-M class)

Table B.1 contains the hyperparameters of each ML model used to predict depression (ZUNG) class (or to solve Problem 2 (D-from-M class)) its combination of hypereparmeters and the grid used by grid search to determine the combination of the used hypereparameters.

Model	Hyperparameters	Grid
Gradient boosting	n_estimators=545	[100,1000, step 5]
	learning_rate=0.02	[0.01,0.1, step 0.01]
	min_samples_leaf=3	[3,20, step 1]
	max_depth=3	[1,9, step 1]
	max_features=0.15	[0.1,0.5, step 0.05]
AdaBoost classifier	n_estimators=100	[5, 10, 15, 20, 25, 50, 75, 100]
	learning_rate=0.1	[0.001, 0.01, 0.1, 1.]
CatBoost classifier	n_estimators=867	[500,1000, step 10]
	max_depth=10	[3,10, step 1]
	learning_rate=0.013	[0.01,0.02, step 0.001]
	max_bin=13	[10,20, step 1]
	l2_leaf_reg=0	[0,10, step 1]
	thread_count=3	[0,10, step 1]
Logistic Regression	C=0.1	[0.01,0.1,1.0,10,100]
	solver=liblinear	['liblinear']
	penalty=l1	['l1', 'l2']
Random Forest classifier	n_estimators=590	[50,1000, step 10]
	max_depth=24	[4,50, step 1]
	min_samples_leaf=1	[1,60, step 1]
	min_samples_split=30	[1,100, step 1]
LightGBM classifier	reg_alpha=7.11	[7,7.9, step 0.01]
	reg_lambda=0.009	[0.0001,0.001, step 0.0001]
	num_leaves=141	[100,200, step 1]
	min_child_samples=37	[1,50, step 1]
	lambda_l1=9.83	[9,10, step 0.01]
	feature_fraction=0.69	[0,1, step 0.01]
	bagging_freq=2	[1,5.step 1]
	max_depth=14	[1,20, step 1]
	learning_rate=0.005	[0,009, step 0.001]
	colsample_bytree=0.1	[0.1,0.9, step 0.1]
	n_estimators=1182	[1100,1200, step 1]
LDA classifier	solver='lsqr'	['svd', 'lsqr', 'eigen']
	tol=0.0001	[0.0001,0.001,0.01,0.1,1]
Ridge classifier	alpha=0.9	[0.1, 1, step 0.1]
Bagging classifier	n_estimators=223	[2,300, step 1]
	max_samples=143	[1,200, step 1]
Extra Trees classifierlassifier	n_estimators=186	[100,200, step 1]
	max_depth=9	[2,10, step 1]
Decision tree classifier	max_depth=2	[1,20, step 1]
	min_samples_split=362	[300,450, step 1]
	max_leaf_nodes=124	[100,200, step 1]
	criterion='entropy'	['gini', 'entropy']
KNN classifier	metric='minkowski'	['minkowski']
	p=1	[1,5, step 1]
	n_neighbors=11	[1,30, step 1]
XGBoost classifier	n_estimators=713	[500,1000, step 1]
	max_depth=10	[3,10, step 1]
	learning_rate=0.02	[0.01,0.1, step 1]
	min_child_weight=44	[20,100, step 1]
	subsample=0.99	[0,1, step 0.01]
	colsample_bytree=0.95	[0,1, step 0.01]
	colsample_bylevel=0.36	[0,1, step 0.01]
	gamma=7	[0,20, step 1]
	eta=0.99	[0,1, step 0.01]
	reg_lambda=0.67	[0,1, step 0.01]
	reg_alpha= 0.72	[0,1, step 0.01]
SVM classifier	C=0.37	[0.1,0.5, step 0.01]
	kernel='linear'	['rbf', 'linear', 'poly']
	gamma=0.009	[0.001,0.01, step 0.001]
GaussianNB classifier	var_smoothing=0.000074	[0.00007,0.00008, step 0.000001]

Table B.1: Hyperparameters of ML classifiers predicting depression (ZUNG) class

In the following tables, we give the parameters of each Voting Classifier model used to predict depression (ZUNG) class.

Model	Voting	Classifier	weights
Voting Classifier (1)	Hard	Gradient boosting	(0,2,2)
		Catboosting	
		AdaBoost	

Table B.2: Definition of Voting Classifier (1) predicting depression (ZUNG) class

Voting Classifier (2) and Voting Classifier (3) are as Voting Classifier 1, the only differences being the weights. In particular,

- Voting Classifier (1): Weights=(0,2,2)
- Voting Classifier (2): Weights=(1,1,1)
- Voting Classifier (3): Weights=(0,1,1)

Voting Classifier (0) is as Voting Classifier (1), the differences are in the weights and the way of voting. In particular,

- Voting Classifier (1): Weights=(0,2,2) and voting=Hard
- Voting Classifier (0): Weights=(2,0,1) and voting=Soft

Model	Voting	Classifier	weights
Voting Classifier (4)	Soft	Extra Trees Classifier	(1,1,1)
		LDA Classifier	
		Random Forest classifier	

Table B.3: Definition of Voting Classifier (4) predicting depression (ZUNG) class

Voting Classifier (5) is as Voting Classifier (4), the only difference is in the way of voting. In particular,

- Voting Classifier (4): Voting=Soft
- Voting Classifier (5): Voting=Hard

Model	Voting	Classifier	weights
Voting classifier (5)	Soft	Gradient boosting	(2,0,1)
		Catboosting	
		AdaBoost	

Table B.4: Definition of Voting Classifier (5) predicting depression (ZUNG) class

The following tables define the Stacking classifiers used to predict depression (ZUNG) class.

Model	**Meta-classifier**	**Classifier**
Stacking Classifier (1)	Gradient Boosting	Catboosting
		AdaBoost
		Logistic Regression

Table B.5: Definition of Stacking Classifier (1) predicting depression (ZUNG) class

Model	**Meta-classifier**	**Classifier**
Stacking Classifier (2)	Extra Trees classifier	Ridge Classifier
		LDA Classifier
		Random Forest classifier

Table B.6: Definition of Stacking Classifier (2) predicting depression (ZUNG) class

B.2 Hyperparameters of regression models for the Problem 1 (D-from-M score)

Table B.7 contains the hyperparameters of each ML model used to predict depression (ZUNG) score (or to solve Problem 1 (D-from-M score)) its combination of hypereparmeters and the grid used by grid search to determine the combination of the used hypereparameters.

Model	Hyperparameters	Grid
Lasso Regression	$alpha = 0.03$	[0,1, step 0.01]
Ridge Regression	$alpha = 100$	[0.00001, 0.0001, 0.001, 0.01, 0.1, 1, 10, 100]
	$fit\_intercept = True$	[True, False]
	$normalize = False$	[True, False]
	$solver$='cholesky'	['svd', 'cholesky', 'lsqr', 'sag']
Elastic Net Regression	$alpha = 0.01$	[0.00001, 0.0001, 0.001, 0.01, 0.1, 1, 10, 100]
	$l1\_ratio = 0.8$	[0.2,0.4,0.6,0.8]
Support Vector Regression	$C = 1100$	[700,1200, step 100]
	$epsilon = 0.25$	[0.1,0.3, step 0.05]
	$kernel$='rbf'	['rbf']
Random Forest Regressor	$bootstrap = True$	[True, False]
	$max\_features$ ='auto'	["auto", "log2", "sqrt"]
	$n\_estimators$ =100	[10,50,100]
XGBoost Regressor	$colsample\_bytree = 0.7$	[0.1,0.3,0.5,0.7]
	$learning\_rate$ =0.03	[0.03, 0.05, 0.07]
	$max\_depth = 5$	[5, 6, 7]
	$min\_child\_weight = 4$	[3,4,5]
	$n\_estimators = 500$	[400,500,600]
	$nthread = 4$	[4]
	$objective$ ='reg:linear'	['reg:linear']
	$silent = 1$	[1]
	$subsample = 0.7$	[0.1,0.3,0.5,0.7]
Extra-trees Regressor	$max\_features = 50$	[50,400, step 50]
	$min\_samples\_leaf = 20$	[20,50, step 5]
	$min\_samples\_split = 15$	[15,35, step 5]
	$n\_estimators = 125$	[50,125, step 25]
Bagging Regressor	$base\_estimator = LinearRegression$	[None, LinearRegression(), KNNRegressor()]
	$bootstrap = False$	[True,False]
	$bootstrap\_features = False$	[True,False]
	$max\_features = 1.0$	[0.5,0.6,0.7,0.8,0.9,1]
	$max\_samples = 1.0$	[0.5,0.6,0.7,0.8,0.9,1]
	$n\_estimators = 20$	[20,50,100]
Gradient boosting Regressor	$n\_estimators = 353$	[300, 400, step 1]
	$learning\_rate = 0.05$	[0.01,0.09, step 0.01]
	$max\_depth = 2$	[1,20, step 1]
	$min\_samples_split = 13$	[2,20, step 1]
	$min\_samples\_leaf = 9$	[2,20, step 1]
	$max\_features = 30$	[10,50, step 1]
CatBoost Regressor	$depth = 6$	[6,7,8,9]
	$iterations = 100$	[30, 50, 100]
	$learning\_rate = 0.1$	[0.001,0.01,0.1,0.2]
	$l2\_leaf\_reg = 10$	[1,3,5,10,100]
LightGBM Regressor	$num\_leaves = 79$	[50,100, step 1]
	$max\_depth = 5$	[5,6,7,8,9]
	$learning\_rate = 0.03$	[0.01,0.03,0.05,0.07,0.09]
	$n\_estimators = 181$	[150,200, step 1]
	$min\_child\_weight = 1.008$	[1,1.01, step 0.001]
	$reg\_alpha = 1.6$	[1.2,1.8, step 0.2]
	$reg\_lambda = 0.99$	[0.8,1, step 0.01]
	$subsample = 0.71$	[0.6,0.8, step 0.01]
AdaBoost Regressor	base_estimator=DecisionTreeRegressor()	[None, DecisionTreeRegressor(), KNNRegressor(), LinearRegression()]
	n_estimators=300	[100,350, step 50]
	learning_rate=2	[0.5, 0.8, 1.0, 2.0]
KNN Regressor	n_neighbors=5	[1,20, step 1]

Table B.7: Hyperparameters of ML regressors predicting depression (ZUNG) score

In the following tables, we define every Voting Regressor model used to predict depression (ZUNG) score.

Model	Regressor	weights
Weighted voting Regression (1)	Lasso Regression XGBoost Regressor Ridge Regressor	(3,1,1)

Table B.8: Definition of Weighted voting Regression (1) predicting depression (ZUNG) score

Model	Regressor	weights
Weighted voting Regression (2)	Lasso Regression ElasticNet Regression XGBoost Regressor Ridge Regression	(4,1,2,2)

Table B.9: Definition of Weighted voting Regression (2) predicting depression (ZUNG) score

Model	Regressor	weights
Weighted voting Regression (3)	CatBoost Regression Gradient Boosting Regressor Lasso Regression	(1,1,2)

Table B.10: Definition of Weighted voting Regression (3) predicting depression (ZUNG) score

The following tables define the Stacking Regressors used to predict depression (ZUNG) score.

Model	Meta-regressor	Regressor
Stacking Regression (1)	Linear Regression	Lasso Regression
		Elastic Net Regression
		XGBoost Regressor

Table B.11: Definition of Stacking Regression (1) predicting depression (ZUNG) score

Model	Meta-regressor	Regressor
Stacking Regression (2)	Lasso Regression	Elastic Net Regression
		XGBoost Regressor
		Ridge Regression

Table B.12: Definition of Stacking Regression (2) predicting depression (ZUNG) score

Model	Meta-regressor	Regressor
Stacking Regression (3)	Linear Regression	Lasso Regression
		Elastic Net Regression
		Gradient Boosting Regressor

Table B.13: Definition of Stacking Regression (3) predicting depression (ZUNG) score

Model	Meta-regressor	Regressor
Stacking Regression (4)	Lasso Regression	LightGBM Regressor
		XGBoost Regressor
		CatBoost Regressor

Table B.14: Definition of Stacking Regression (4) predicting depression (ZUNG) score

Model	Meta-regressor	Regressor
Stacking Regression (5)	CatBoost Regressor	Gradient Boosting Regressor
		Elastic Net Regression
		XGBoost Regressor

Table B.15: Definition of Stacking Regression (5) predicting depression (ZUNG) score

The First averaged model: The First averaged model averages the predictions of the models: Lasso Regression, Elastic Net Regression,XGBoost Regression, Ridge Regression,Linear Regression, Random Forest Regressor, Support Vector Machines Regressor used to predict depression (ZUNG) score.

The Second averaged model: The Second averaged model averages the predictions of all the models used to predict the depression (ZUNG) score except for the stacking regression models and the weighted voting regression models.

B.3 Hyperparameters of regression models for the Problem 3 (M-from-D score)

Table B.16 contains the hyperparameters of each ML model used to predict memory (PRMQ) score (or to solve Problem 3 (M-from-D score)) its combination of hypereparmeters and the grid used by grid search to determine the combination of the used hypereparameters.

Model	Hyperparameters	Grid
Gradient Boosting Regressor	$n\_estimators = 585$	[500,700, step 5]
	$learning\_rate = 0.01234$	[0.0123,0.0124, step 0.00001]
	$max\_depth = 5$	[1,20, step 1]
	$min\_samples\_split = 4$	[2,20, step 1]
	$min\_samples\_leaf = 10$	[2,20, step 1]
	$max\_features = 11$	[2,20, step 1]
CatBoost Regressor	$border\_count = 100$	
	$depth = 6$	[6,7,8,9]
	$iterations = 100$	[30, 50, 100]
	$l2\_leaf\_reg = 10$	[3,1,5,10,100]
	$learning\_rate = 0.1$	[0.001,0.01,0.1,0.2]
LightGBM Regressor	$boosting\_type =$ "$dart$"	["gbdt", "dart", "goss"]
	$num\_leaves = 65$	[17,75, step 10]
	$max_depth = 2$	[2,10, step 1]]
	$learning\_rate = 0.55$	[0.5,1,0.05]
	$n\_estimators = 100$	[100,1000, step 100]
AdaBoost Regression	$base\_estimator = DecisionTreeRegressor()$	[None, DecisionTreeRegressor(), KNNRegressor(), LinearRegression()]
	$learning\_rate = 0.8$	[0.5, 0.8, 1.0, 2.0]
	$n\_estimators = 300$	[100, 350, step 50]
XGBoost Regression	$colsample\_bytree = 0.7$	[0.5,0.6,0.7,0.8,0.9]
	$learning\_rate = 0.03$	[0.03, 0.04,0.05,0.06, 0.07]
	$max\_depth = 5$	[5, 6, 7]
	$min\_child\_weight = 4$	[1,2,3,4,5]
	$n\_estimators = 500$	[400,500,600,700,800,900,1000]
	$nthread = 4$	[1,2,3,4,5]
	$objective =$ '$reg : linear$'	['reg:linear']
	$silent = 1$	[1,2,3]
	$subsample = 0.7$	[0.5,0.6,0.7,0.8,0.9]
Elastic Net Regression	$alpha = 0.1$	[0.1,0.2,0.3,0.4,0.5]
	$l1\_ratio = 0.8$	[0,1, step 0.01]
Bagging Regression	$base\_estimator = None$	[None, LinearRegression(), KNNRegressor()]
	$bootstrap = True$	[True, False]
	$bootstrap\_features = False$	[True, False]
	$max\_features = 1.0$	[0.5,1.0, step 0.1]
	$max\_samples = 0.5$	[0.5,1.0, step 0.1]
	$n\_estimators = 100$	[20,50,100]
Lasso Regression	$alpha = 0.07$	[0,1, step 0.01]
Random Forest Regressor	bootstrap=True	[True, False]
	$max\_features =$ "$auto$"	["auto", "log2", "sqrt"]
	$n\_estimators = 100$	[10,50,100]
Ridge Regression	$alpha = 100$	[0.00001, 0.0001, 0.001, 0.01, 0.1, 1, 10, 100]
	$fit\_intercept = True$	[True, False]
	$normalize = False$	[True, False]
	$solver =$ "svd"	['svd', 'cholesky', 'lsqr', 'sag']
Extra Trees Regressor	$max\_features = 50$	[40,50,60,70,80,90,100]
	$min\_samples\_leaf = 20$	[10,20,30,40,50]
	$min\_samples\_split = 20$	[10,20,30,40,50]
	$n\_estimators = 125$	[100,125,150,175,200]
KNN Regressor	$n\_neighbors = 4$	[1,20, step 1]
Support Vector Regression	$C = 1100$	[700,1200, step 100]
	$epsilon = 0.25$	[0.1,0.3, step 0.05]
	$kernel =$ "rbf"	["rbf"]

Table B.16: Hyperparameters of ML regressors predicting memory (PRMQ) score

In the following tables, we give the parameters os each Voting Regressor model used to predict memory (PRMQ) score.

Model	Regressor	weights
Weighted voting Regression (1)	CatBoost Regressor	[1,4,4]
	Gradient Boosting Regressor	
	Lasso Regressor	

Table B.17: Definition of Weighted voting Regression (1) predicting memory (PRMQ) score

Model	Regressor	weights
Weighted voting Regression (2)	Gradient Boosting Regressor	[2,2,2,4]
	CatBoost Regressor	
	LightGBM Regressor	
	AdaBoost Regressor	

Table B.18: Definition of Weighted voting Regression (2) predicting memory (PRMQ) score

Model	Regressor	weights
Weighted voting Regression (3)	Gradient Boosting Regressor	[3,1,2]
	CatBoost Regressor	
	AdaBoost Regressor	

Table B.19: Definition of Weighted voting Regression (3) predicting memory (PRMQ) score

Model	Regressor	weights
Weighted voting Regression (4)	Gradient Boosting Regressor	[2,1]
	CatBoost Regressor	

Table B.20: Definition of Weighted voting Regression (4) predicting memory (PRMQ) score

Model	Regressor	weights
Weighted voting Regression (5)	Lasso Regression	[4,2,1]
	XGBoost Regressor	
	Ridge Regression	

Table B.21: Definition of Weighted voting Regression (5) predicting memory (PRMQ) score

The following tables define the Stacking regressors used to predict mwmory (PRMQ) score.

Model	Meta-regressor	Regressor
Stacking Regression (1)	Linear Regression	Lasso Regression
		ElasticNet Regression
		XGBoost Regression

Table B.22: Definition of Stacking Regression (1) predicting memory (PRMQ) score

Model	Meta-regressor	Regressor
Stacking Regression (2)	Lasso Regression	XGBoost Regression
		ElasticNet Regression
		Ridge Regression

Table B.23: Definition of Stacking Regression (2) predicting memory (PRMQ) score

Model	Meta-regressor	Regressor
Stacking Regression (3)	Linear Regression	Lasso Regression
		ElasticNet Regression
		Gradient Boosting Regression

Table B.24: Definition of Stacking Regression (3) predicting memory (PRMQ) score

Model	Meta-regressor	Regressor
Stacking Regression (4)	Lasso Regression	LightGBM Regression
		XGBoost Regression
		CatBoosting Regression

Table B.25: Definition of Stacking Regression (4) predicting memory (PRMQ) score

Model	Meta-regressor	Regressor
Stacking Regression (5)	XGBoost Regression	CatBoosting Regression
		LightGBM Regression
		AdaBoost Regression

Table B.26: Definition of Stacking Regression (5) predicting memory (PRMQ) score

Model	Meta-regressor	Regressor
Stacking Regression (6)	XGBoost Regression	CatBoosting Regression
		LightGBM Regression
		AdaBoost Regression

Table B.27: Definition of Stacking Regression (6) predicting memory (PRMQ) score

Model	Meta-regressor	Regressor
Stacking Regression (7)	CatBoosting Regression	Gradient Boosting Regression
		LightGBM Regression
		AdaBoost Regression

Table B.28: Definition of Stacking Regression (7) predicting memory (PRMQ) score

Model	Meta-regressor	Regressor
Stacking Regression (8)	CatBoosting Regression	Gradient Boosting Regression
		ElasticNet Regression
		XGBoost Regression

Table B.29: Definition of Stacking Regression (8) predicting memory (PRMQ) score

Below we define the two averaged models used to predict memory (PRMQ) score:

The First averaged model: The First averaged model averages the predictions of the models: Lasso Regression, Elastic Net Regression,XGBoost Regression, Ridge Regression,Linear Regression, Random Forest Regressor, Support Vector Machines Regressor

The Second averaged model: The Second averaged model averages the predictions of all the models used to predict the Memory (PRMQ) score except for the stacking regression models and the weighted voting regression models.

B.4 Hyperparameters of regression models for the Problem 4 (M-from-D class)

The grid for the grid search to choose the best set of hyperparameters for each classifier predicting memory-related disorders to solve Problem 4 (M-from-D class) is given in the following table.

Model	Hyperparameters	Grid
Gradient boosting	n_estimators	[10, 100, 1000]
	learning_rate	[0.001, 0.01, 0.1]
	subsample	[0.5, 0.7, 1.0]
	max_depth	[3, 7, 9]
AdaBoost classifier	n_estimators	[5, 10, 15, 20, 25, 50, 75, 100]
	learning_rate	[0.001, 0.01, 0.1, 1.]
Catboosting	depth	[4,5,6,7,8,9,10]
	learning_rate	[0.01,0.02,0.03,0.04]
	iterations	[10,20,30,40,50,60,70,80,90,100]
Logistic Regression	C	[0,10, step 0.5]
Random Forest classifier	n_estimator	[2,10,30,50,100]
	max_depth	[5,16, step 2]
	min_samples_leaf	[1,2,5]
	min_samples_split	[2,5,10,15,20,50,100]
LDA Classifier	solver	['svd', 'lsqr', 'eigen']
Ridge Classifier	alpha	[0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0]
Bagging classifier	n_estimators	[100, 300, 500, 800, 1200]
	max_samples	[1, 2, 5, 10, 13]
	max_features	[5, 10, 25, 50, 100]
Extra Trees Classifier	n_estimators	[1,50, step 2]
	max_features	[1,50, step 2]
Decision Tree Classifier	max_depth	[5,16, step 2]
	min_samples_leaf	[1,2,5]
	min_samples_split	[2,5,10,15,20,50,100]
KNN classifier	leaf_size	[1,50, step 1]
	p	[1,2]
	n_neighbors	[1,30, step 1]
XGBoost Classifier	n_estimators	[60,220, step 40]
	max_depth	[2,10, step 1]
	learning_rate	[0.01,0.05,0.1]
GaussianNB Classifier	var_smoothing=0.000074	[0,0.0001, step 0.00001]
SVM classifier	C	[50, 10, 1.0, 0.1, 0.01]
	kernel	['rbf','sigmoid', 'poly']
	gamma	['scale']

Table B.30: Hyperparameters of ML classifiers predicting memory (PRMQ) class

Appendix C

Questionnaire and Time Features

Questionnaire and Time Features

1. Response ID
2. Registration date
3. Last page
4. Original language
5. Seed
6. Start date
7. Last action date
8. Reference URL
9. I agree to participate in the above study and wish to receive the results of the online PRMQ (memory) test and the Zung Test for depression at the end of the study (Cons03)
10. I confirm that I am over 18 years of age. (Cons05)
11. My participation is voluntary and I am free to withdraw at any time and without explanation. (Cons01)
12. I understand that this result is for informational purposes only and cannot replace a clinical examination by a psychiatrist/neurologist/psychologist or other health care professional. (Cons04)
13. I confirm that I have read and understood the information sheet for the above study and have had the opportunity to ask questions. (Cons02)
14. Participant Code. Your code is made up of the first letter of your maiden name, the first letter of your first name and your year of birth. (Dem00)
15. Fill in your age. (Dem01)
16. What gender are you? (Dem02)
17. Where do you live? (Dem03)
18. Which of the following best describes your education? (Dem04)
19. Do you think you have memory disorders that affect your daily life? (Dem05)

20. Have you ever visited a memory clinic, psychologist or psychiatrist? (Dem06)
21. Is there a family history of memory disorders (mother, father, grandfather, grandmother, brother or sister)? (Dem07)
22. Do you currently suffer from a known disease that affects memory (e.g., depression, Alzheimer's disease, mild cognitive impairment, n. Parkinson's disease, Multiple Sclerosis, etc.)? (Dem08)
23. Do you think you experience mood disorders (e.g. depression) that affect your daily life? (Dem09)
24. Do you suffer from a medical condition?If so, which one? [I do not suffer from any pathological problem. I am perfectly healthy] (Dem10)
25. Do you suffer from any pathological disease? [Blood Pressure] (Dem10)
26. Do you suffer from any pathological disease? If so, which one? [Chronic Atrial Fibrillation or Other Cardiac Problem] (Dem10)
27. Do you suffer from any pathological disease? [Stroke] (Dem10)
28. Do you suffer from a medical condition? [High Cholesterol] (Dem10)
29. Do you have a medical condition? [Diagnosed Depression] (Dem10)
30. Do you suffer from a medical condition? [Diagnosed Anxiety] (Dem10)
31. Do you suffer from a medical condition? [Other] (Dem10)
32. Have you ever suffered a head injury that resulted in hospitalization? (Dem11)
33. Do you suffer from hypothyroidism? (Dem12)
34. Do you suffer from diabetes mellitus? (Dem13)
35. Do you smoke? (Dem14)
36. How many glasses of alcohol do you consume per week? [Beer - 500ml glass of 5% alcohol] (Dem15)
37. How many glasses of alcohol do you consume per week? [Cider (330ml) 4.5% alcohol] (Dem15)
38. How many glasses of alcohol do you consume per week? [Wine - Medium glass of Chardonnay (175ml) 12.5% alcohol] (Dem15)
39. How many glasses of alcohol do you consume per week? [Tsipouro, Ouzo or raki (40ml) 40% alcohol] (Dem15)
40. How many glasses of alcohol do you consume per week? [White Drink (vodka, tequila, gin, etc.) (40ml) 40% alcohol] (Dem15)
41. How many glasses of alcohol do you consume per week? [Dark Drink (rum, whisky, etc.) (40ml) 40% alcohol] (Dem15)
42. How many glasses of alcohol do you consume per week? [Sweet liqueur (40ml) 17% alcohol] (Dem15)
43. Do you exercise more than 3 hours per week? (Dem16)

44. Which statement best describes your sleep quality [Other] (Dem17)

45. In the past, have you been a confirmed case/infected with coronavirus (COVID-19)? (Dem18)

46. To what extent do you think your memory is good? [Mnemonic Ability] (SEM02)

47. In the last two weeks what has been your average mood? [Mood] (SEM01)

PRMQ questionnaire

48. Do you decide to do something within the next few minutes and then forget to do it? (Mem01)

49. Do you have difficulty recognizing a place you have visited before? (Mem02)

50. Do you find it difficult to do something you intended to do even though it is right in front of you, such as taking a pill or turning off the stove? (Mem03)

51. Do you forget something you were told a few minutes ago? (Mem04)

52. Do you forget appointments if you are not reminded by someone or if you have not marked them in a diary? (Mem05)

53. Do you have trouble recognizing a character on TV from one scene to another? (Mem06)

54. Do you forget to buy something you were planning to get, like a gift, even though you've seen the store? (Mem07)

55. Do you have trouble recalling things that have happened to you in the last few days? (Mem08)

56. Do you repeat the same story to the same person on different occasions? (Mem09)

57. Do you forget to take something with you before you leave a room or before you go out even if it's right there in front of you? (Mem10)

58. Do you lose things you just put somewhere, like a magazine or your glasses? (Mem11)

59. Do you forget to mention something or give something to someone who asked you? (Mem12)

60. Do you look at something without realizing you saw it a few minutes ago? (Mem13)

61. If you try to contact a friend or relative who is absent at the time do you forget to try again later? (Mem14)

62. Do you forget what you saw on TV the day before? (Mem15)

63. Do you forget to mention something you were going to mention just a few minutes ago? (Mem16)

Zung questionnaire

64. Feeling discouraged or sad. (Q1)

65. I feel better in the morning than at any time of the day. (Q2)

66. I cry easily or feel ready to cry (Q3)

67. I have trouble sleeping at night. (Q4)
68. I eat the same amount of food as before. (Q5)
69. I am still interested in sex. (Q6)
70. I notice that I am losing weight (Q7)
71. I have constipation problems (Q8)
72. I have palpitations. (Q9)
73. I get tired for no particular reason (Q10)
74. My mind is as clear as before. (Q11)
75. It is easy for me to do the things I used to do before. (Q12)
76. I feel restless and cannot calm down. (Q13)
77. I have optimism about my future. (Q14)
78. I have more nervousness than before. (Q15)
79. I make decisions as easily as before. (Q16)
80. I feel useful and needed. (Q17)
81. My life is quite “full”. (Q18)
82. I feel it would be better for others if I died. (Q19)
83. I still enjoy the things I used to do. (Q20)

Time features

84. Total Time to complete the questionnaire: Total Time
85. Response time to question: Cons03
86. Response time to question: Cons05
87. Response time to question: Cons01
88. Response time to question: Cons04
89. Response time to question: Cons02
90. Response time to question: Dem00
91. Response time to question: Dem01
92. Response time to question: Dem02
93. Response time to question: Dem03
94. Response time to question: Dem04
95. Response time to question: Dem05
96. Response time to question: Dem06
97. Response time to question: Dem07
98. Response time to question: Dem08
99. Response time to question: Dem09
100. Response time to question: Dem10
101. Response time to question: Dem11
102. Response time to question: Dem12
103. Response time to question: Dem13
104. Response time to question: Dem14
105. Response time to question: Dem15
106. Response time to question: Dem16
107. Response time to question: Dem17
108. Response time to question: Dem18
109. Response time to question: SEM02
110. Response time to question: SEM01
111. Response time to question: Mem01
112. Response time to question: Mem02
113. Response time to question: Mem03

Short code	Question number
Cons01	11
Cons02	13
Cons03	9
Cons04	12
Cons05	10
Dem00	14
Dem01	15
Dem02	16
Dem03	17
Dem04	18
Dem05	19
Dem06	20
Dem07	21
Dem08	22
Dem09	23
Dem10	24-31
Dem11	32
Dem12	33
Dem13	34
Dem14	35
Dem15	36-42
Dem16	43
Dem17	44
Dem18	45
Sem01	47
Sem02	46
Mem01	48
Mem02	49
Mem03	50
Mem04	51
Mem05	52
Mem06	53
Mem07	54
Mem08	55
Mem09	56
Mem10	57
Mem11	58
Mem12	59
Mem13	60
Mem14	61
Mem15	62
Mem16	63
Q1	64
Q2	65
Q3	66
Q4	67
Q5	68
Q6	69
Q7	70
Q8	71
Q9	72
Q10	73
Q11	74
Q12	75
Q13	76
Q14	77
Q15	78
Q16	79
Q17	80
Q18	81
Q19	82
Q20	83

Table C.1: For the sake of simplicity, the same questionnaire is presented by showing only the number of the question and its abbreviation if it exists

Bibliography

[1] W. H. Organization, "Who highlights urgent need to transform mental health and mental health care." Available at https://www.who.int/news/item/17-06-2022-who-highlights-urgent-need-to-transform-mental-health-and-mental-health-care.

[2] U. of Washington, "Institute of health metrics and evaluation — explore results from the 2019 global burden of disease (gbd) study." Available at https://vizhub.healthdata.org/gbd-results/.

[3] W. H. Organization, "Mental health and covid-19: Early evidence of the pandemic's impact: Scientific brief, 2 march 2022." Available at https://www.who.int/publications/i/item/WHO-2019-nCoV-SciBrief-Mentalhealth-2022.1.

[4] M. Hamilton, "Development of a rating scale for primary depressive illness," *British journal of social and clinical psychology*, vol. 6, no. 4, pp. 278–296, 1967.

[5] F. Edition *et al.*, "Diagnostic and statistical manual of mental disorders," *Am Psychiatric Assoc*, vol. 21, no. 21, pp. 591–643, 2013.

[6] W. H. Organization, "World mental health day: An opportunity to kick-start a massive scale-up in investment in mental health.." Available at https://www.who.int/news/item/27-08-2020-world-mental-health-day-an-opportunity-to-kick-start-a-massive-scale-up-in-investment-in-mental-health.

[7] A. Wu, L. March, X. Zheng, J. Huang, X. Wang, J. Zhao, F. M. Blyth, E. Smith, R. Buchbinder, and D. Hoy, "Global low back pain prevalence and years lived with disability from 1990 to 2017: estimates from the global burden of disease study 2017," *Annals of translational medicine*, vol. 8, no. 6, 2020.

[8] J. Upton, *Beck Depression Inventory (BDI)*, pp. 178–179. New York, NY: Springer New York, 2013.

[9] S. P. C. H. Arif., "Depression." Available at https://www.ncbi.nlm.nih.gov/books/NBK430847/, Accessed on 2022-07-18.

[10] N. A. Hubbard, J. L. Hutchison, M. Turner, J. Montroy, R. P. Bowles, and B. Rypma, "Depressive thoughts limit working memory capacity in dysphoria," *Cognition and Emotion*, vol. 30, no. 2, pp. 193–209, 2016. PMID: 25562416.

[11] PsychCentral, "What is major depressive disorder?." Available at https://psychcentral.com/depression/major-depressive-disorder#symptoms, Accessed on 2021-03-25.

[12] PsychCentral, "Can depression cause memory loss?." Available at https://psychcentral.com/depression/depression-and-memory-loss, Acceessed on 2021-08-24.

[13] A. K. Zawn Villines, "Are depression and memory loss connected?." Available at https://www.medicalnewstoday.com/articles/depression-and-memory-loss, Accessed on 2020-07-22.

[14] D. J. Shelton and C. B. Kirwan, "A possible negative influence of depression on the ability to overcome memory interference," *Behavioural brain research*, vol. 256, pp. 20–26, 2013.

[15] P. L. Rock, J. Roiser, W. J. Riedel, and A. Blackwell, "Cognitive impairment in depression: a systematic review and meta-analysis," *Psychological medicine*, vol. 44, no. 10, pp. 2029–2040, 2014.

[16] N. A. Hubbard, J. L. Hutchison, M. Turner, J. Montroy, R. P. Bowles, and B. Rypma, "Depressive thoughts limit working memory capacity in dysphoria," *Cognition and Emotion*, vol. 30, no. 2, pp. 193–209, 2016.

[17] C. A. Kohler, A. F. Carvalho, G. S. Alves, R. S. McIntyre, T. N. Hyphantis, and M. Cammarota, "Autobiographical memory disturbances in depression: a novel therapeutic target?," *Neural plasticity*, 2015.

[18] S. Schweizer, R. A. Kievit, T. Emery, R. N. Henson, *et al.*, "Symptoms of depression in a large healthy population cohort are related to subjective memory complaints and memory performance in negative contexts," *Psychological medicine*, vol. 48, no. 1, pp. 104–114, 2018.

[19] M. Semkovska, L. Quinlivan, T. O'Grady, R. Johnson, A. Collins, J. O'Connor, H. Knittle, E. Ahern, and T. Gload, "Cognitive function following a major depressive episode: a systematic review and meta-analysis," *The Lancet Psychiatry*, vol. 6, no. 10, pp. 851–861, 2019.

[20] D. G. Dillon and D. A. Pizzagalli, "Mechanisms of memory disruption in depression," *Trends in neurosciences*, vol. 41, no. 3, pp. 137–149, 2018.

[21] F. J. Muñoz-Almaraz, M. T. Climent, M. D. Guerrero, L. Moreno, and J. Pardo, "A machine learning approach to design an efficient selective screening of mild cognitive impairment," *JoVE (Journal of Visualized Experiments)*, no. 155, p. e59649, 2020.

[22] E. Pfeiffer, "A short portable mental status questionnaire for the assessment of organic brain deficit in elderly patients †," *Journal of the American Geriatrics Society*, vol. 23, 1975.

[23] M. Karunakaran, J. Balusamy, and K. Selvaraj, "Machine learning models based mental health detection," in *2022 Third International Conference on Intelligent Computing Instrumentation and Control Technologies (ICICICT)*, pp. 835–842, IEEE, 2022.

[24] T. Richter, B. Fishbain, E. Fruchter, G. Richter-Levin, and H. Okon-Singer, "Machine learning-based diagnosis support system for differentiating between clinical anxiety and depression disorders," *Journal of Psychiatric Research*, vol. 141, pp. 199–205, 2021.

[25] I. Tachmazidis, T. Chen, M. Adamou, and G. Antoniou, "A hybrid ai approach for supporting clinical diagnosis of attention deficit hyperactivity disorder (adhd) in adults," *Health Information Science and Systems*, vol. 9, no. 1, pp. 1–8, 2021.

[26] R. Goodman, D. Renfrew, and M. Mullick, "Predicting type of psychiatric disorder from strengths and difficulties questionnaire (sdq) scores in child mental health clinics in london and dhaka," *European child & adolescent psychiatry*, vol. 9, no. 2, pp. 129–134, 2000.

[27] M. Zhao and Z. Feng, "Machine learning methods to evaluate the depression status of chinese recruits: A diagnostic study," *Neuropsychiatric disease and treatment*, vol. 16, p. 2743, 2020.

[28] T. Bhowmik, "Inteligencia artificial: Naive bayes vs logistic regression: Theory, implementation and experimental validation," *Inteligencia Artificial*, vol. 18, pp. 14–30, 12 2015.

[29] G. James, D. Witten, T. Hastie, and R. Tibshirani, *Tree-Based Methods*, pp. 303–335. New York, NY: Springer New York, 2013.

[30] A. Ibrahim, R. Raheem, M. Muhammed, R. Abdulaziz, and S. Ganiyu, "Comparison of the catboost classifier with other machine learning methods," *International Journal of Advanced Computer Science and Applications*, vol. 11, p. 11, 12 2020.

[31] A. F. M. Agarap, "On breast cancer detection: an application of machine learning algorithms on the wisconsin diagnostic dataset," in *Proceedings of the 2nd international conference on machine learning and soft computing*, pp. 5–9, 2018.

[32] S. Rajasekaran, S. Gayathri, and T.-L. Lee, "Support vector regression methodology for storm surge predictions," *Ocean Engineering*, vol. 35, no. 16, pp. 1578–1587, 2008.

[33] W. K. Ho, B.-S. Tang, and S. W. Wong, "Predicting property prices with machine learning algorithms," *Journal of Property Research*, vol. 38, no. 1, pp. 48–70, 2021.

[34] E. Y. Boateng, J. Otoo, and D. A. Abaye, "Basic tenets of classification algorithms k-nearest-neighbor, support vector machine, random forest and neural network: a review," *Journal of Data Analysis and Information Processing*, vol. 8, no. 4, pp. 341–357, 2020.

[35] T. K. Bhowmik, "Naive bayes vs logistic regression: theory, implementation and experimental validation," *Inteligencia Artificial. Revista Iberoamericana de Inteligencia Artificial*, vol. 18, no. 56, pp. 14–30, 2015.

[36] K. M. Al-Aidaroos, A. A. Bakar, and Z. Othman, "Naive bayes variants in classification learning," in *2010 international conference on information retrieval & knowledge management (CAMP)*, pp. 276–281, IEEE, 2010.

[37] N. Hoernle, "Introduction to data science advanced section 6: Topics in supervised classification." Available at `https://harvard-iacs.github.io/2018-CS109A/a-sections/a-section-6/presentation/asection6.pdf`.

[38] Z. Shayan, N. M. G. Mezerji, L. Shayan, and P. Naseri, "Prediction of depression in cancer patients with different classification criteria, linear discriminant analysis versus logistic regression," *Global journal of health science*, vol. 8, no. 7, p. 41, 2016.

[39] L. Melkumova and S. Y. Shatskikh, "Comparing ridge and lasso estimators for data analysis," *Procedia engineering*, vol. 201, pp. 746–755, 2017.

[40] H. Zou and T. Hastie, "Regression shrinkage and selection via the elastic net, with applications to microarrays," *JR Stat Soc Ser B*, vol. 67, pp. 301–20, 2003.

[41] P. Bühlmann, "Bagging, boosting and ensemble methods." Available at `https://stat.ethz.ch/Manuscripts/buhlmann/handbook-cs-rev2010.pdf`.

[42] A. Ruiz and N. Villa, "Storms prediction: Logistic regression vs random forest for unbalanced data," *arXiv preprint arXiv:0804.0650*, 2008.

[43] Y. Li, C. Zou, M. Berecibar, E. Nanini-Maury, J. C.-W. Chan, P. Van den Bossche, J. Van Mierlo, and N. Omar, "Random forest regression for online capacity estimation of lithium-ion batteries," *Applied energy*, vol. 232, pp. 197–210, 2018.

[44] Y. Pandya, "Ensemble methods in machine learning." Available at `https://medium.com/analytics-vidhya/ensemble-methods-in-machine-learning-31084c3740be`, Accessed on 2021-02-14.

[45] B. Ozen, "Introduction to boosting methodology & adaboost algorithm." Available at `https://burakozen.medium.com/introduction-to-boosting-m`

ethodology-adaboost-algorithm-a9a3e8be6bc3, Accessed on 2021-01-21.

[46] G. Ridgeway, D. Madigan, and T. S. Richardson, "Boosting methodology for regression problems," in *Seventh International Workshop on Artificial Intelligence and Statistics*, PMLR, 1999.

[47] J. C. Figueroa-García, E. R. López-Santana, and J. I. Rodriguez-Molano, *Applied Computer Sciences in Engineering: 5th Workshop on Engineering Applications, WEA 2018, Medellín, Colombia, October 17-19, 2018, Proceedings, Part I*, vol. 915. Springer, 2018.

[48] A. Natekin and A. Knoll, "Gradient boosting machines, a tutorial," *Frontiers in neurorobotics*, vol. 7, p. 21, 2013.

[49] B. Pan, "Application of xgboost algorithm in hourly pm2. 5 concentration prediction," in *IOP conference series: earth and environmental science*, vol. 113, p. 012127, IOP publishing, 2018.

[50] X. Sun, M. Liu, and Z. Sima, "A novel cryptocurrency price trend forecasting model based on lightgbm," *Finance Research Letters*, vol. 32, p. 101084, 2020.

[51] L. Prokhorenkova, G. Gusev, A. Vorobev, A. V. Dorogush, and A. Gulin, "Catboost: unbiased boosting with categorical features," *Advances in neural information processing systems*, vol. 31, 2018.

[52] A. A. Ibrahim, R. L. Ridwan, M. M. Muhammed, R. O. Abdulaziz, and G. A. Saheed, "Comparison of the catboost classifier with other machine learning methods," *International Journal of Advanced Computer Science and Applications*, vol. 11, no. 11, 2020.

[53] A. R. Rout, "Bagging vs boosting in machine learning." Available at https://www.geeksforgeeks.org/bagging-vs-boosting-in-machine-learning/, Accessed on 2022-06-01.

[54] Roger Grosse, Chris Maddison, Juhan Bae, Silviu Pitis, "CSC 311: Introduction to machine learning." Available at https://www.cs.toronto.edu/rgrosse/courses/csc311f20/slides/lec06.pdf, Accessed on Fall, 2020.

[55] A. Gupta, "Ml — extra tree classifier for feature selection." Available at https://www.geeksforgeeks.org/ml-extra-tree-classifier-for-feature-selection/ Accessed on 2020-06-01.

[56] N. L. 2022, "A comprehensive guide to ensemble learning: What exactly do you need to know." Available at https://neptune.ai/blog/ensemble-learning-guide, Accessed on 2020-01-27.

[57] R. Natras, B. Soja, and M. Schmidt, "Ensemble machine learning of random forest, adaboost and xgboost for vertical total electron content forecasting," *Remote Sensing*, vol. 14, no. 15, p. 3547, 2022.

[58] G. MALATO, "Hyperparameter tuning. grid search and random search." Available at https://www.yourdatateacher.com/2021/05/19/hyperparameter-tuning-grid-search-and-random-search/, Accessed on 2021-05-19.

[59] A. Bhandari, "Everything you should know about confusion matrix for machine learning." Available at https://www.analyticsvidhya.com/blog/2020/04/confusion-matrix-machine-learning/#::text=A%20Confusion%20matrix%20is%20an,by%20the%20machine%20learning%20model., Accessed on 2017-04-17.

[60] . Explorium, "Top 10 evaluation metrics for classification models." Available at https://www.explorium.ai/blog/top-10-evaluation-metrics-for-classification-models/, Accessed on 2023-01-26.

[61] . . . G. L. E.-L. S. P. Ltd., "Understanding roc (receiver operating characteristic) curve — what is roc?." Available at https://www.mygreatlearning.com/blog/roc-curve/, Accessed on 2022-10-31.

[62] I. Inada, "Comprehensive guide on mean squared error (mse)." Available at https://www.skytowner.com/explore/comprehensiveguideonmeansquarederror, Accessed on 2023-03-05.

[63] . A. Inc., "Understand advanced metrics." Available at https://help.anaplan.com/685ff9b2-6370-46ba-af10-679405937113-Understand-advanced-metrics.

[64] F. F. Rivera, T. F. Pena, and J. C. Cabaleiro, *Euro-Par 2017: Parallel Processing: 23rd International Conference on Parallel and Distributed Computing, Santiago de Compostela, Spain, August 28–September 1, 2017, Proceedings*, vol. 10417. Springer, 2017.

[65] A. Mahani and A. R. B. Ali, "Classification problem in imbalanced datasets," *Recent Trends in Computational Intelligence*, pp. 1–23, 2019.

[66] H. He and E. A. Garcia, "Learning from imbalanced data," *IEEE Transactions on Knowledge and Data Engineering*, vol. 21, no. 9, pp. 1263–1284, 2009.

[67] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "Smote: synthetic minority over-sampling technique," *Journal of artificial intelligence research*, vol. 16, pp. 321–357, 2002.

[68] F. Feng, K.-C. Li, J. Shen, Q. Zhou, and X. Yang, "Using cost-sensitive learning and feature selection algorithms to improve the performance of imbalanced classification," *IEEE Access*, vol. 8, pp. 69979–69996, 2020.

[69] H. He, Y. Bai, E. A. Garcia, and S. Li, "Adasyn: Adaptive synthetic sampling approach for imbalanced learning," in *2008 IEEE international joint conference on neural networks (IEEE world congress on computational intelligence)*, pp. 1322–1328, IEEE, 2008.

[70] N. V. Chawla, A. Lazarevic, L. O. Hall, and K. W. Bowyer, "Smoteboost: Improving prediction of the minority class in boosting," in *Knowledge Discovery in Databases: PKDD 2003: 7th European Conference on Principles and Practice of Knowledge Discovery in Databases, Cavtat-Dubrovnik, Croatia, September 22-26, 2003. Proceedings 7*, pp. 107–119, Springer, 2003.

[71] C. Seiffert, T. M. Khoshgoftaar, J. Van Hulse, and A. Napolitano, "Rusboost: A hybrid approach to alleviating class imbalance," *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, vol. 40, no. 1, pp. 185–197, 2009.

[72] M. ALTINI, "Dealing with imbalanced data: Undersampling, oversampling and proper cross-validation." Available at `https://www.marcoaltini.com/blog/dealing-with-imbalanced-data-undersampling-oversampling-and-proper-cross-validation`, Accessed on 2015-08-17.

[73] R. A. A. Viadinugroho, "Imbalanced classification in python: Smote-tomek links method." Available at `https://towardsdatascience.com/imbalanced-classification-in-python-smote-tomek-links-method-6e48dfe69bbc`, Accessed on 2021-04-18.

[74] R. A. A. Viadinugroho, "Imbalanced classification in python: Smote-enn method." Available at `https://towardsdatascience.com/imbalanced-classification-in-python-smote-enn-method-db5db06b8d50`, Accessed on 2021-05-31.

[75] E. F. Swana, W. Doorsamy, and P. Bokoro, "Tomek link and smote approaches for machine fault classification with an imbalanced dataset," *Sensors*, vol. 22, no. 9, p. 3246, 2022.

[76] J. Brownlee, "Imbalanced classification with python (7-day mini-course)." Available at `https://machinelearningmastery.com/imbalanced-classification-with-python-7-day-mini-course/`, Accessed on 2020-01-20.

[77] K. Singh, "How to improve class imbalance using class weights in machine learning." Available at `https://www.analyticsvidhya.com/blog/2020/10/improve-class-imbalance-class-weights/`, Accessed on 2020-10-06.

[78] S. Li and X. Zhang, "Research on orthopedic auxiliary classification and prediction model based on xgboost algorithm," *Neural Computing and Applications*, vol. 32, no. 7, pp. 1971–1979, 2020.