A Comparative Study of Various Machine Learning Classification Algorithms

Paraschopoulos Kyriakos

Supervisor Professor:

Konstantinos E. Psannis

MSc in Department of Applied Informatics
University of Macedonia,

February 2020

Abstract

Machine learning has the ability to learn from data and provide data driven insights, decisions, and predictions. Due to huge amount of data which is generated every day, it is very essential to use machine learning tecnhiques. The basic objective of this thesis is to present a brief introduction of various and most used classification algorithms. At this study the *Logistic Regression*, *Naïve Bayes*, *k-Nearest Neighbors* and *Support Vector Machine* algorithms are described and implemented. Additionally, a set of evaluation metrics is applied to the classifiers and gives useful insights about the predictive ability of each algorithm.

Keywords: Machine Learning, Classification algorithms, Logistic Regression, Naïve Bayes, k-Nearest Neighbors, Support Vector Machine

Table of Contents

Chapter 1

Introduction

The amount of data generated and stored in databases is already enormous and it keeps on growing very fast. Recent technological revolutions such as social media enable us to generate data much faster than ever before. While the amount of data is increasing rapidly, the interest in data mining is becoming an increasingly important tool to transform these data into information. Data Mining refers to the non-trivial extraction of implicit, previously unknown and potentially useful information from data in databases.

1.1 Big Data

Big data is a term that describes the large volume of data, both structured and unstructured, whose size or type is beyond the ability of traditional relational databases to capture, manage and process efficiently. Big data can be analyzed for insights that lead to better decisions and strategic business moves. Big data can be characterized along three important dimensions: *volume*, *velocity*, and *variety*.

Data volume measures the amount of generated and stored data. As data volume increases, the value of different data records will decrease as well as the size of data which determines if it can be considered as big data or not. **Data velocity** measures the speed of data creation where the data is generated and processed. **Data variety** is a measure of the the data quality and the variety of the data representation. The quality of captured data as well formats can vary greatly from

structured, numeric data in traditional databases to unstructured text documents, emails, videos, audios. [2]

1.2 Data Science, AI & Machine Learning

Artificial intelligence, Machine learning, and Data Science are interdisciplinary fields that are all related to each other, using key skills of a wide range of fields. Figure 1.1 represents the relationship between Artificial Intelligence, Machine Learning, and Data Science.

1.2.1 Data Science

Data science refers to the process of extraction of useful insights from data. It merges different techniques from various fields of computer science, mathematics and statistical models in order to extract insights in automated ways. Many companies are currently using data science, applying algorithms on their large amount of data, to build recommendation engines or predict user behaviors.

Examples of data science can be any recommendation engines that are able to recommend the activity of a particular user or any fraud alert models that detect fraudulent credit card transactions or predict revenue for the next quarter.

1.2.2 Artificial intelligence

Artificial intelligence (AI) refers to the process of making machines able to simulate the human brain function, to understand data, learn from the data, and make decisions based on patterns hidden in the data. AI is defined as a collection of mathematical algorithms that leads to computers' understanding of relationships between different types and pieces of data.

The term artificial intelligence appeared for the first time in print in 1955. It became super popular recently because of the advancements of computer power. [3] Nowadays, the unlimited processing power, as well as the great imporvement of algorithm implementations have managed to make the possibilities of AI seem endless. AI applications are being used in every day life; facial recognition, automated

driving, sorting mail can be some indicative examples. In many case machines have exceeded human abilities and efficiency. Our prediction for the future is that the AI applications will become faster, smarter, and more convenient in terms of implementation and use. [1]

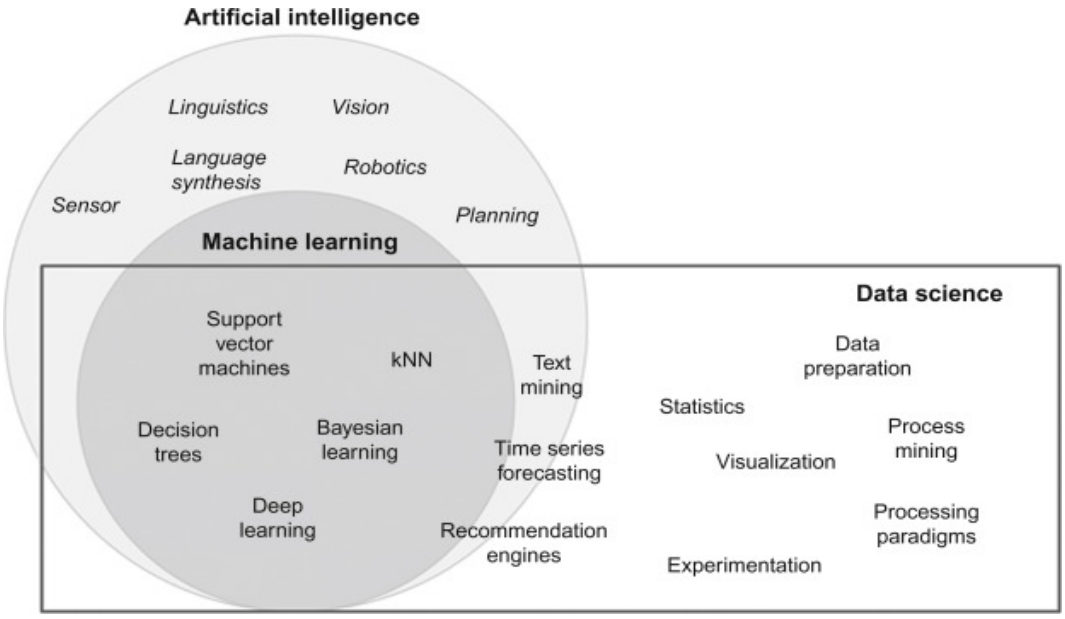

Figure 1.1: Artificial intelligence, machine learning, and data science

1.2.3 Machine Learning

In 1959, Arthur Samuel defined machine learning as a "Field of study that gives computers the ability to learn without being explicitly programmed". [4] Tom M. Mitchell provided a widely more formal definition of the algorithms studied in the machine learning field: "A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P if its performance at tasks in T, as measured by P, improves with experience E." [5]

Machine learning (ML) is a sub-field of artificial intelligence that provides a computer system the ability to automatically learn and improve from experience without being explicitly programmed. It focuses on the development of computer programs, using statistical methods and algoritmhs, to learn from the provided data, gather insights and make predictions on previously unanalyzed data based on the gathered information.

Types of Machine Learning Algorithms

Machine learning is sub-categorized to three types. The algorithms can be organized based on the type of input available for training and the desired outcome of the algorithm.

The types of learning algorithms are the following:

- Supervised learning.
- Unsupervised learning.
- Reinforcement learning.

Supervised Learning

In supervised learning, algorithms learn from labeled data. A training dataset is provided and the learning algorithms make predictions for any new input associating the learned patterns to the unlabeled new data. The supervised learning is divided into 2 categories **Classification** and **Regression**.

Classification

Classification is a technique used to identify to which category new observations belong. Classification algorithms are used when the outputs are discreted or categorical.

Regression

Regression is a statistical approach to find the relationship between a dependent variable and one or more independent variables. Regression models are used to predict numerical or continuous variables based on previous observed data from the trained dataset.

Unsupervised Learning

The unsupervised learning analyzes, sorts and categorizes large amount of unstructured data. It is used when the provided data is not classified or labeled. The goal

of unsupervised learning is to explore interesting hidden patterns from unlabeled data.

Reinforcement Learning

The Reinforcement learning algorithms constitute a learning method that interacts with its environment in order to make decisions that will optimize a given of expected reward over a period of time. Reinforcement learning consists of the agent, the decision maker, the environment the agent interacts with, and the actions that the agent can do. It is mostly used for robotics and navigation. [6]

1.2.4 Traditional Programming versus Machine Learning

On the one hand, in traditional programming, a programmer implements a code using data as input and the code runs on a computer producing an output. On the other hand, in machine learning, the input as well as the output (training data) are fed to an algorithm to produce the model of the program. [1]

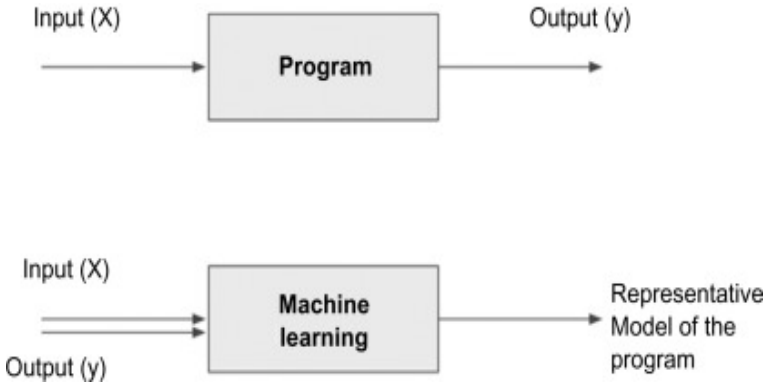

Figure 1.2: Traditional program and machine learning

Chapter 2

Dataset

2.1 Data Exploration

The word “data” comes from the plural Latin word *datum* which means “(thing) given”, the neuter past participle of dare "to give". In the 20th century datum was being used in singular and data in plural. [7]

A dataset is a collection of data. It makes reference to a database table, where each column of the table represents a specific variable and each row corresponds to the observation of each member.

There are three different dataset splits that are used for training, tuning and testing of Machine Learning models: *training set*, *validation set* and *test set*. [9]

2.1.1 Training set

The training set is the actual collection of data used to train the machine learning model by matching the input with the expected output. The training set represents the 60% of the original data set.

2.1.2 Validation set

The validation set is a set of examples used to tune the hyperparameters of the classifier to train the machine learning model with the optimal process. It is a way to evaluate how well the model has been trained. The validation set represents the

20% of the original data set.

2.1.3 Test set

The test set is a set of examples that is applied at the final step when the model is fully trained and it is used to evaluate the performance of the classifier. After assessing the final model on the test set, no more tunings are allowed. The test set represents the 20% of the original data set. [10]

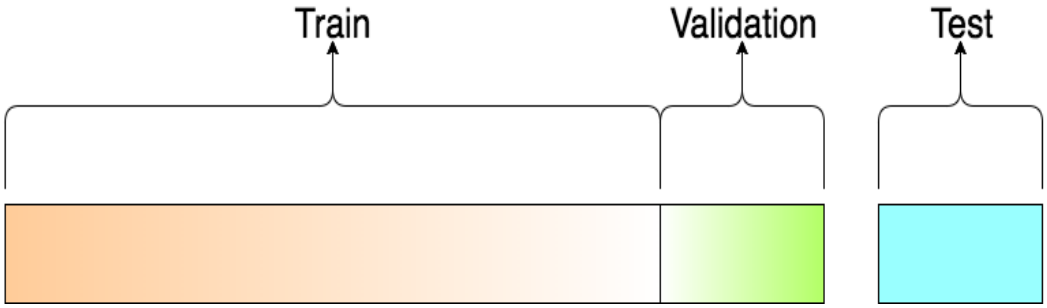

Figure 2.1: A visualisation of the datset splits

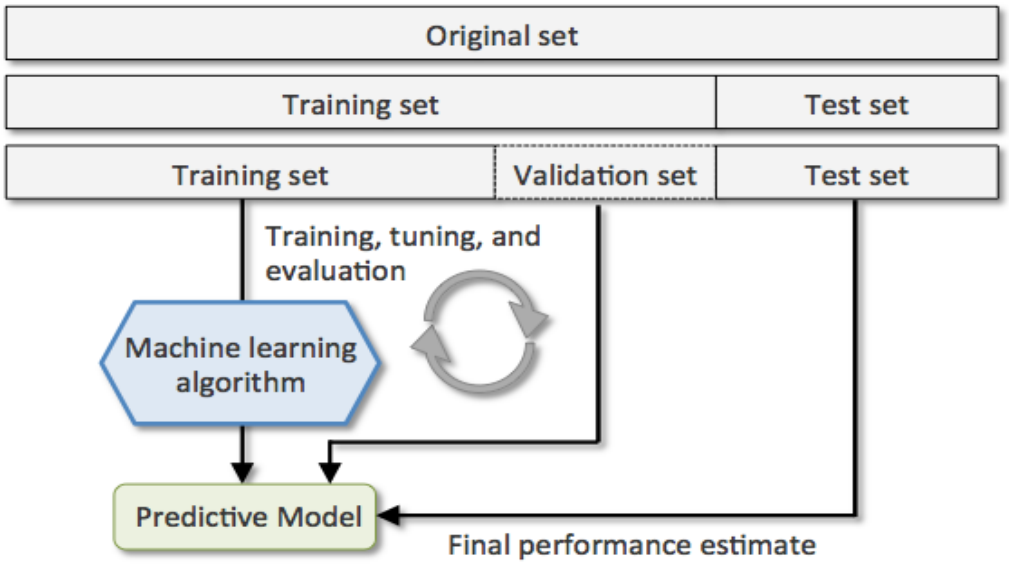

Figure 2.2: Dataset graph cicle [8]

2.1.4 Overfitting & Underfitting

Overfitting happens when a model learns the detail and noise patterns that are present in the training data and it occurs when the performance of the model on new data and on test data is negative. On the contrary, underfitting occurs when a model is too simple and fails to capture the patterns both in training and test data. [12]

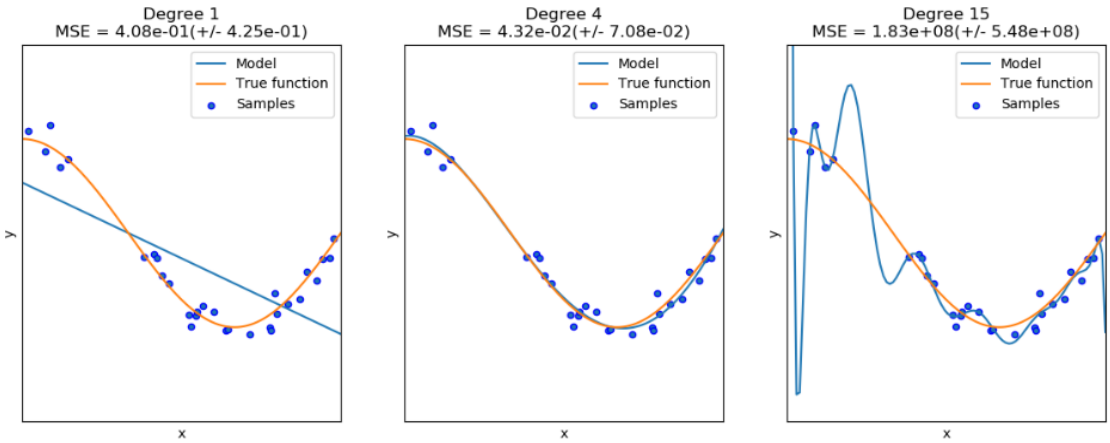

Figure 2.3: Overfitting Vs Underfitting [13]

2.2 Wine dataset

The Wine dataset from UCI Machine Learning Repository will be used to perform the experiments of the classification algorithms. Information regarding the dataset is also described below. [11]

2.2.1 Dataset Information

The data of this wine dataset are the results of a chemical analysis of wines grown in Italy but derived from three different cultivars. The analysis determined the quantities of 13 constituents found in each of the three types of wines.

There are 13 numeric, predictive attributes:

1. Alcohol
2. Malic acid
3. Ash
4. Alcalinity of ash
5. Magnesium
6. Total phenols
7. Flavanoids
8. Nonflavanoid phenols
9. Proanthocyanins
10. Color intensity
11. Hue
12. OD280/OD315 of diluted wines
13. Proline

2.2.2 Dataset analysis

First, we have to import the libraries needed for the analysis of the dataset.

```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
%matplotlib inline
```

Load the dataset and review the first 5 rows of your data using the head() function.

```
dataset = pd.read_csv('data/Winewithheader.csv')
display(dataset.head(n=5))
```

Check if any of these columns have missing information.

```
dataset.isnull().any()
```

```
class                 False
Alcohol               False
Malic_acid            False
Ash                   False
Alcalinity_of_ash     False
Magnesium             False
Total_phenols         False
Flavanoids            False
Nonflavanoid_phenols  False
Proanthocyanins       False
Color_intensity       False
Hue                   False
OD280/OD315           False
Proline               False
dtype: bool
```

Figure 2.4: Missing information per column

Get Additional information of the dataset.

```
dataset.info()
```

```
dataset.isnull().any()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 178 entries, 0 to 177
Data columns (total 14 columns):
class                   178 non-null int64
Alcohol                 178 non-null float64
Malic_acid              178 non-null float64
Ash                     178 non-null float64
Alcalinity_of_ash       178 non-null float64
Magnesium               178 non-null int64
Total_phenols           178 non-null float64
Flavanoids              178 non-null float64
Nonflavanoid_phenols    178 non-null float64
Proanthocyanins         178 non-null float64
Color_intensity         178 non-null float64
Hue                     178 non-null float64
OD280/OD315             178 non-null float64
Proline                 178 non-null int64
dtypes: float64(11), int64(3)
memory usage: 19.6 KB
```

Figure 2.5: Dataset information

2.2.3 Dimensions of the dataset

Using the shape property from Pandas library will represent the dimensionality of the dataset.

```
dataset.info()
```

The dataset consists of 178 row and 14 columns.

2.2.4 Class distribution

A dataset with imbalanced class distributions (with more observations for one class than another) may cause problems in classification. If the imbalance level is high, some techniques will be applied in the data preparation step to solve the class imbalance problem.

Using the Pandas library will count the observations per class in the dataset.

```
dataset.groupby('class').size()
```

From the output, it turns out that the class 1 consists of 59 observations, class 2 of 71 and class 3 of 48.

Chapter 3

Evaluation Metrics

Classification algorithms are widely used in every day life. The optimization of the trained model is crucial and we use evaluation metrics to choose the optimal generative classifier. In this thesis four main error indicators will be used to estimate the quality of trained classification models. *Accuracy*, *Precision*, *Recall* and *f1-score* are the evaluation metrics which will be analyzed and implemented.

The evaluation metrics are categorized in three types; threshold, probability and ranking metric. These metrics are used to evaluate the classifier with different objective. The threshold and ranking metric are used to measure the performance of classifiers. [40]

3.1 Review of Evaluation Metrics

In classification problems, the evaluation metrics are applied in the training and testing steps. In the training step, the evaluation metric is used to measure the performance of classifier for the optimal algorithm selection. In the testing step, the evaluation metric is used to measure the effectiveness of the final classifier on unseen data. [40]

3.1.1 Confusion Matrix

The confusion matrix, known as error matrix, is a table which visualizes the performance of an algorithm. It is used for binary or multi-class classification prob-

lems. [41]

In table 3.1 a binary classification problem is depicted. Each cell in the table has a specific name.

	Actual Positive Class	**Actual Negative Class**
Predicted Positive Class	True Positive (tp)	False Positive (fp)
Predicted Negative Class	False Negative (fn)	True Negative (tn)

Table 3.1: Confusion Matrix for Binary Classification and the Corresponding Array Representation. [42]

The row of the table represents the predicted (positive or negative) class, while the column represents the actual (positive or negative) class.

- **True Positive (tp):** When the *actual* class of data and the *predicted* are 1 (true).
- **True Negative (tn):** When the *actual* class of data and the *predicted* are 0 (false).
- **False Positive (fp):** When the *actual* class of data is 0 (false) and the *predicted* is 1 (true).
- **False Negative (fn):** When the *actual* class of data is 1 (true) and the *predicted* is 0 (false).

The optimal scenario is when there are 0 false positives and 0 false negatives.

3.1.2 Accuracy

The Accuracy metric is the most used evaluation metric number for both binary and multi-class classification problems. Accuracy reflects the number of correct predictions made by the model over the total number of instances evaluated.

$$\text{Accuracy (acc)} = \frac{tp + tn}{tp + fp + tn + fn}$$

Accuracy can be considered as a valid evaluation metric when the target variable classes are well balanced and not when the target variable classes correspond mostly to one class. [40]

3.1.3 Error Rate

Error rate is the percentage of incorrect predictions and it measures the ratio of incorrect predictions over the total number of instances. [40]

$$\text{Error Rate (err)} = \frac{fp + fn}{tp + fp + tn + fn}$$

3.1.4 Precision

The Precision metric is used to measure what proportion of predicted positives is actual positive or correctly predicted as positive at the trained model. [42]

$$\text{Precision (p)} = \frac{tp}{tp + fp}$$

3.1.5 Recall

The Recall metric is used to measure the proportion of positives that are correctly classified and the number of correct positives divided by the number of all samples that have been identified as positive. [42]

$$\text{Recall (r)} = \frac{tp}{tp + tn}$$

3.1.6 F1-score

The f1-score metric is used to measure how precise a classifier is, by finding the balance between precision and recall. It represents the harmonic mean of the *Precision* and the *Recall* scores. The range of f1-score is between 0 and 1.

$$\text{F-Measure (FM)} = \frac{2 * p * r}{p + r}$$

If the *Precision score* is low, the f1-score is also low and if the *Recall score* is low the f1-score is low as well.

Chapter 4

Logistic Regression

Regression is an old technique dated back to the Victorian era (from the 1830s to the early 1900s). Francis Galton described a biological phenomenon of children's height comparing against their parents' height. He observed that the height of children of tall parents tended to be slightly shorter than themselves and for short parents, the height of their children was slightly higher than themselves. [14]

Logistic regression, like any statistic technique, is used to find the best fitting model to describe the relationship between the dependent variable (outcome) and a set of independent (predictor or explanatory) variables. Logistic Regression is used for binary classification and it essentially predicts the probability that an item belongs to one of two categories, it provides a binary outcome (1 / 0, Yes / No, True / False) given a set of independent variables. Since probabilities are always between 0 and 1, we can threshold them at 0.5 to construct a classification algorithm. Logistic regression actually behaves identically with the linear regression. The only difference is that you need to threshold the outputs to produce class predictions. In other words, logistic regression is a special case of linear regression when the outcome variable is categorical; it predicts the probability of occurrence of an event by fitting data to a logit function.

In cases where dependent variables can have more than two outcomes, like class 1/class 2/class 3, such scenarios are classified as **multinomial logistic regression**, and they are used in the same way to predict the outcome.

4.1 Linear Regression Vs. Logistic Regression

Linear regression gives you a continuous output in contrast to logistic regression which provides a discrete output. Linear regression is estimated using Ordinary Least Squares (OLS) while logistic regression is estimated using Maximum Likelihood Estimation (MLE) approach.

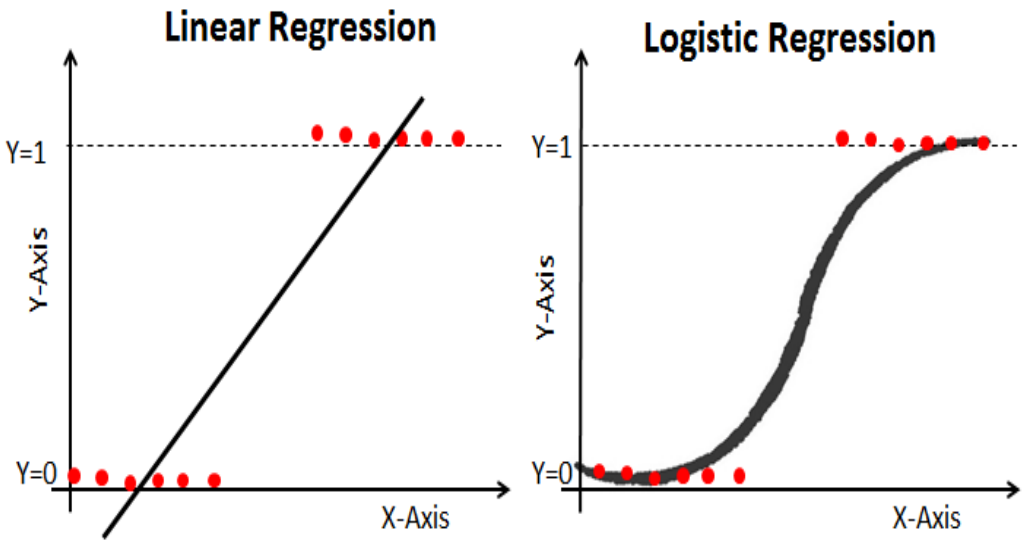

Figure 4.1: Linear Regression VS Logistic Regression Graph| Image: Data Camp

- **Linear Regression** predictions are continuous (numbers in a range).
- **Logistic Regression** predictions are discrete (only specific values or categories are allowed). We can also view probability scores underlying the model's classifications.

Maximum Likelihood Estimation Vs. Ordinary Least Square method

The Maximum Likelihood Estimation (MLE) is a "likelihood" maximization method that determines values for the parameters of a model, which are most likely to produce the observed data. In statistics, MLE sets the mean and variance as parameters in determining the specific parametric values for a given model.

The Ordinary Least Square (OLS) also called as the linear least squares, determines the unknown parameters located in a linear regression model. In statistics, OLS is a method of analysis that estimates the relationship between one or more independent variables and a dependent variable minimizing the sum of the squares of the differences between the observed dependent variable in the given dataset and those predicted by the linear function. [15]

4.2 Basics of Logistic Regression

Logistic model: Sigmoid Function

The logistic regression model finds the correct decision boundary for one of the two categories in the data set. The line in the graphic represents the logistic function shifted and squeezed to fit the data.

The sigmoid function, also called logistic function, has a characteristic "S"-shaped curve or sigmoid curve. The name Sigmoid comes from the Greek letter Sigma, and when graphed, it appears as a sloping “S” across the Y-axis. A logistic curve starts with slow, linear growth, followed by exponential growth, which then slows again to a stablerate. A simple logistic function is defined by the following formula:

$$f(x) = \frac{1}{1 + e^{-x}}$$

f(x) = output between 0 and 1 (probability estimate)

z = input to the function (your algorithm’s prediction e.g. mx + b)

e = base of natural log

Decision boundary

The sigmoid function can take any real-valued number between 0 and 1. If the output of the sigmoid function is more than 0.5, we can classify the outcome as 1 or YES, and if it is less than 0.5, we can classify it as 0 or NO. So, by definition, this model predicts probabilities and then we use this probability to predict classes. [16]

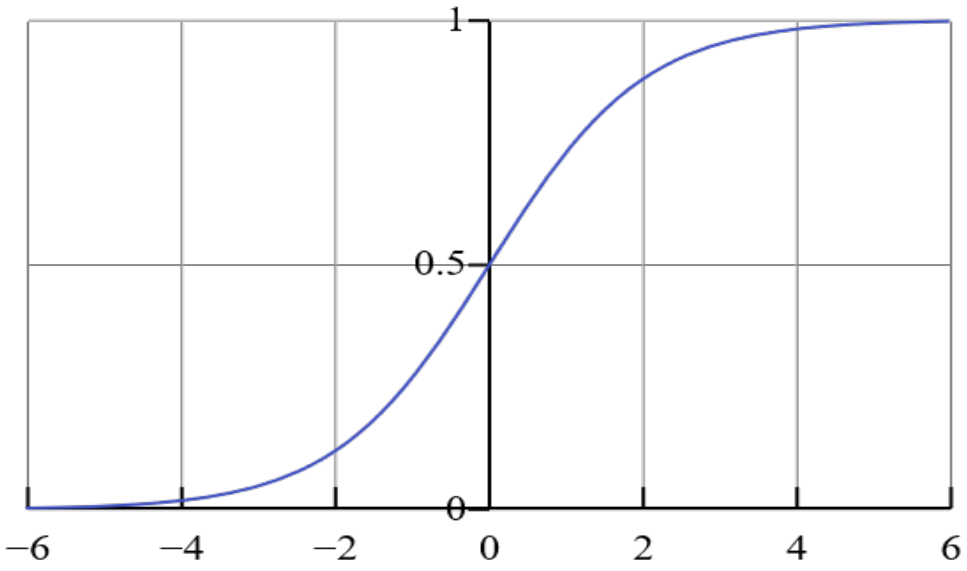

Figure 4.2: Sigmoid Function Graph

$$f(x) \geq 0.5, class = 1$$

$$f(x) < 0.5, class = 0$$

$$0 \leq h_\theta(x) \leq 1$$

4.3 Types of Logistic Regression

- **Binary Logistic Regression:** It has only two possible outcomes (yes/no).
- **Multinomial Logistic Regression:** The target variable has three or more nominal categories such as predicting the type of Wine.
- **Ordinal logistic regression:** The target variable has three or more ordinal categories such as predicting the quality of wine by rating it from 1 to 5.

4.4 Applications of Logistic Regression

Spam Detection

Spam detection or spam filtering are used to detect unsolicited and unwanted email and prevent those messages from getting to a user's inbox. Spam detection is a **binary classification** problem where we are given an email and we need to classify whether or not it can be considered as spam. If the email is considered as spam, we label it 1; if it is not spam, we label it 0. In order to apply Logistic Regression to the spam detection problem, the following features of the email are extracted:

- Sender of the email
- Number of typos in the email
- Occurrence of words/phrases like "offer", "prize", "free gift", etc.

The resulting feature vector is then used to train a Logistic classifier which emits a score in the range 0 to 1. If the score is more than 0.5, we label the email as spam. Otherwise, we don't label it as spam. [17]

Credit Card Fraud

The increased usage of credit cards over the last few years has eventually led to the problem of credit card fraud. Identifying or detecting fraudulent behavior in credit card transaction system is crucial. Credit card fraud is defined as an unauthorized account activity by a person to whom the account was not intended. Thus, it is necessary to develop more sophisticated and effective techniques in order to predict, detect, and avoid undesirable fraud.

The Credit Card Fraud Detection problem is of significant importance to the banking industry in order to reduce the amount of loss due to frauds. When a credit card transaction happens, the bank systems collects and analyzes data such as the date of the transaction, the amount, the place, the type of purchase, etc. Based on these factors, a Logistic Regression model is developed of whether or not the transaction is a fraud. [18]

Tumour Prediction

A Logistic Regression classifier may be used to identify whether a tumour is malignant or if it is benign based on some characteristic features. Several medical imaging techniques are used to extract various features of tumours. For instance, the size of the tumour, the affected body area, etc. These features are then fed to train a Logistic Regression classifier to identify if the tumour is malignant or if it is benign. [19]

Marketing

Logistic regression can also be used to analyze the marketing effectiveness, pricing and promotions on the sales of a product. For instance, when a company wants to know if the funds invested in the marketing of a particular brand has given them substantial return on the investment using the predictive insights exported from the model. [20]

4.5 Logistic Regression Implementation

4.5.1 LR model

Implementation of Logistic Regression using scikit-learn library:

```
# Load dataset
dataset = pd.read_csv('/content/data/WineWithHeader.csv')


# Retrieve rows from dataset
X = dataset.iloc[:,1:]
y = dataset.iloc[:,0]


# Feature Scaling
slc= StandardScaler()
X = slc.fit_transform(X)


# Spliting data
```

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.2,
    random_state = 0)

# Define the model
model = LogisticRegression()

# Train the model
model.fit(X_train, y_train)

# Predict
prediction = model.predict(X_test)
```

4.5.2 Confusion Matrix

Corresponding confusion matrix of LR classifier:

```
cm = confusion_matrix(y_test, prediction)

print("Confusion Matrix:\n", cm)

fig, ax = plot_confusion_matrix(conf_mat=cm)

plt.ylabel("True Values")
plt.xlabel("Predicted Values")
plt.title("Confusion Matrix Visualization")
plt.show()
```

Output of the above block of code:

```
Confusion Matrix:
    [14 0  0]
    [0  16  0]
    [0  0   6]
```

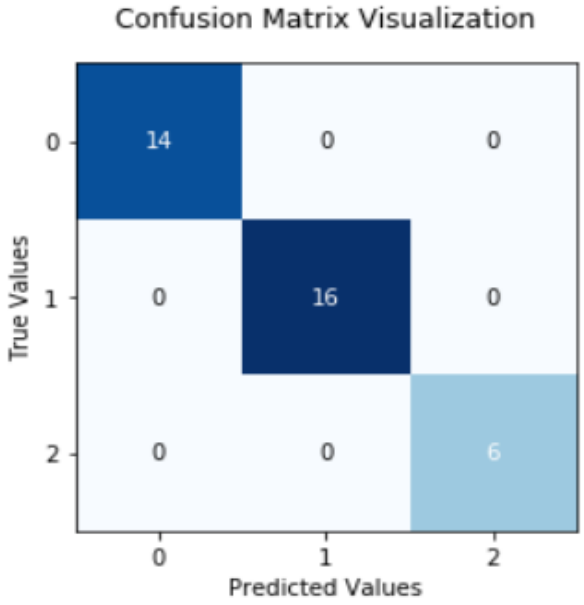

Figure 4.3: Visualization of Confusion Matrix

4.5.3 Evaluation Metrics

Accuracy

```
# Accuracy
accuracy = accuracy_score(y_test, prediction)
print("Accuracy: ", accuracy)
```

Accuracy output:

```
Accuracy: 1.0
```

Precision, Recall, f1-score

```
# Precision, Recall, f1-score
metrics = metrics.classification_report(y_test,prediction, digits=3)
print(metrics)
```

Here is a summary of the precision, recall and f1-score for our three classes:

	precision	recall	f1-score	support
1	1.000	1.000	1.000	14
2	1.000	1.000	1.000	16
3	1.000	1.000	1.000	6
accuracy			1.000	36
macro avg	1.000	1.000	1.000	36
weighted avg	1.000	1.000	1.000	36

Figure 4.4: Classification Metrics Report

The *support* column lists the number of samples for each class.

Chapter 5

Naïve Bayes

Naïve Bayes is an algorithm that learns the probability of an object with certain features that belongs to a particular group/class.

The Naïve Bayesian algorithm is built on the Bayes' theorem, named after Reverend Thomas Bayes. "Essay Towards Solving a Problem in the Doctrine of Chances" (1763) describes Bayes' work which is published posthumously. Naïve Bayes algorithm has been studied since 1960. It was introduced and stil remains a popular method for text categorization, classifying documents to one or other category.

5.1 Basics of Naïve Bayes

Naïve Bayes is a subset of the Bayesian decision theory. It is called "naïve" because it makes the assumption that the occurrence of a certain feature is independent of the occurrence of other features. It is a probabilistic classifier and it is primarily used for text classification which involves high dimensional training data sets.

The Naïve Bayesian algorithm tries to predict class labels by best approximating the probabilistic relationship between the independence attributes and the class label. [21]

5.2 Bayes' theorem

The Bayes' theorem is one of the most influential and important concepts in statistics and probability theory. The theorem describes a way of finding a probability when we know certain other probabilities. Bayes' theorem finds the probability of a given event occurring the probability of another event that has already occurred. [22]

The Formula For Bayes' theorem is

$$P(A \mid B) = \frac{P(A \cap B)}{P(B)} = \frac{P(A) * P(B \mid A}{P(B)}$$

where:

P(A) and P(B) : Probabilities of the occurrence of event and respectively

P(A | B) : The probability of occurrence of event A given the event B is true

P(B | A) : The probability of occurrence of event B given the event A is true

Using Bayes theorem, we can find the probability of A happening, given that B has occurred. Here, B is the evidence and A is the hypothesis. The assumption made here is that the predictors/features are independent. The presence of one particular feature does not affect the other. Hence it is called naïve.

5.2.1 Bayes' Theorem Explained

Bayes' theorem count on the *prior* probability in order to generate *posterior* probabilities. **Prior probability** is the probability of an event without any other data, while, **posterior** probability is the revised probability of an event occurring after collecting new data. Posterior probability is calculated by updating the prior probability by using Bayes' theorem.

5.2.2 Naïve Bayes classifier

There are multiple classifiers of the Naïve Bayes algorithm depending on the distribution of $P(x_i|y)$. Three of the commonly used classifiers are:

Gaussian Naïve Bayes classifier

In Gaussian Naïve Bayes, continuous data associated with each feature are assumed to be distributed according to a Gaussian distribution. A Gaussian distribution is also called Normal distribution. When plotted, it gives a bell shaped curve which is symmetric about the mean of the feature values as shown in figure 5.1:

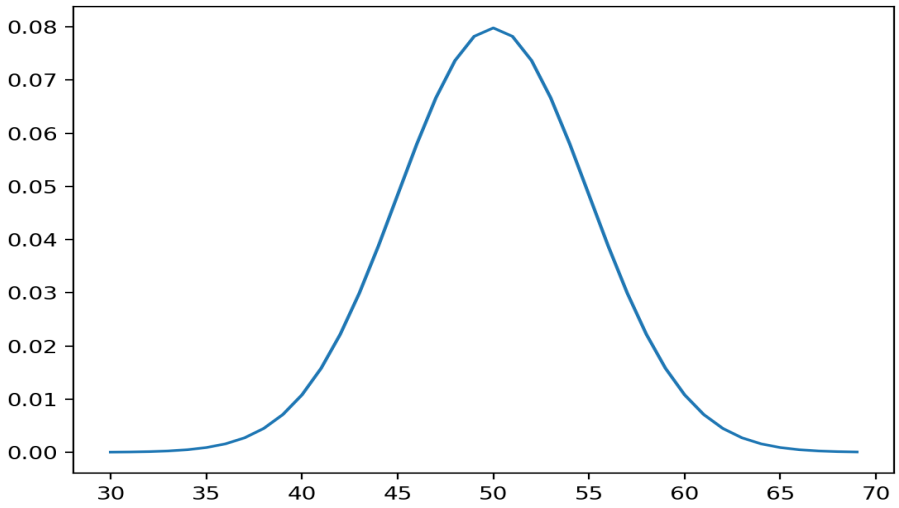

Figure 5.1: Normal Distribution

It is used in classification and the likelihood of the features is assumed to be Gaussian, hence, conditional probability is given by:

$$P(x_i|y) = \frac{1}{\sqrt{2\pi\sigma_y^2}} exp\left(-\frac{(x_i - \mu_y)^2}{2\sigma_y^2}\right)$$

The parameters σ_y and μ_y are estimated using maximum likelihood. [23]

Multinomial Naïve Bayes classifier

The Multinomial Naïve Bayes algorithm is used when the data is distributed multinomially, the feature vectors represent the frequencies with which certain events have been generated by a multinomial distribution. This model is mostly used for document classification problems. [24]

$$\theta_{y_i} = \frac{N_{y_i} + \alpha}{N_y + \alpha n}$$

n : the number of features, in text classification, the size of the vocabulary.
θ_{y_i} : the probability of feature i appearing in a sample belonging to class y.
N_{y_i} : the number of times that feature i appears in a sample belonging to class y.

Bernoulli Naïve Bayes

The Bernoulli Naïve Bayes algorithm is used when the feature vectors are independent booleans (binary variables) describing inputs. The algorithm is similar to the Multinomial Naïve Bayes except for the predictors which are boolean variables. Like the multinomial model, this model is mostly used for document classification tasks. [25] [26]

5.3 Advantages & Disadvantages of Naïve Bayes Algorithm

Advantages:

- Prediction on a new data point is very fast.
- It requires a small amount of training data to estimate the test data; it can achieve better results than other classifiers because it has a low propensity to overfit.
- It is easy to understand and implement

Disadvantages:

- The main imitation of Naïve Bayes is the assumption of independent predictors. Naïve Bayes implicitly assumes that the features to be independent which is hardly true in real life applications.

- If the categorical variable has a category (in test data set), which was not observed in training data set, then the model will assign a 0 (zero) probability and will be unable to make a prediction, "Zero Conditional Probability Problem.". There are techniques to solve this such as the "Laplacian Correction". [27]

5.4 Applications of Naïve Bayes

The Naïve Bayes algorithm is used in multiple applications such as:

Text classification/ Spam Filtering/ Sentiment Analysis

Naïve Bayes classifiers are mostly used in text classification (due to their better results in multi-class problems, a text document belongs to one or more classes). As a result, it is widely used in spam filtering, in recognizing spam emails from legitimate emails as well as, as Sentiment Analysis, to analyze the tone of tweets, comments, and reviews to identify positive and negative customer sentiments.

Recommendation System

Naïve Bayes Classifier in combination with algorithms like Collaborative Filtering is used to make a Recommendation System that uses machine learning and data mining techniques to filter unseen information and predict whether a user would like a given resource or not.

Real-time Prediction

As Naïve Bayes is fast, it can be used for making predictions in real time.

Multi-class Prediction

This algorithm can predict the posterior probability of multiple classes of the target variable. [27]

5.5 Naïve Bayes Implementation

5.5.1 NB model

Implementation of Naïve Bayes using scikit-learn library:

```
# Load dataset
dataset = pd.read_csv('/content/data/WineWithHeader.csv')

# Retrieve rows from dataset
X = dataset.iloc[:,1:]
y = dataset.iloc[:,0]

# Feature Scaling
slc= StandardScaler()
X = slc.fit_transform(X)

# Spliting data
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.2,
    random_state = 0)

# Define the model
model = GaussianNB()

# Train the model
model.fit(X_train, y_train)

# Predict
prediction = model.predict(X_test)
```

5.5.2 Confusion Matrix

Corresponding confusion matrix of NB classifier:

```
cm = confusion_matrix(y_test, prediction)
```

```
print("Confusion Matrix:\n", cm)

fig, ax = plot_confusion_matrix(conf_mat=cm)

plt.ylabel("True Values")
plt.xlabel("Predicted Values")
plt.title("Confusion Matrix Visualization")
plt.show()
```

Output of the above block of code:

```
Confusion Matrix:
    [14 0  0]
    [2  13  1]
    [0  0  6]
```

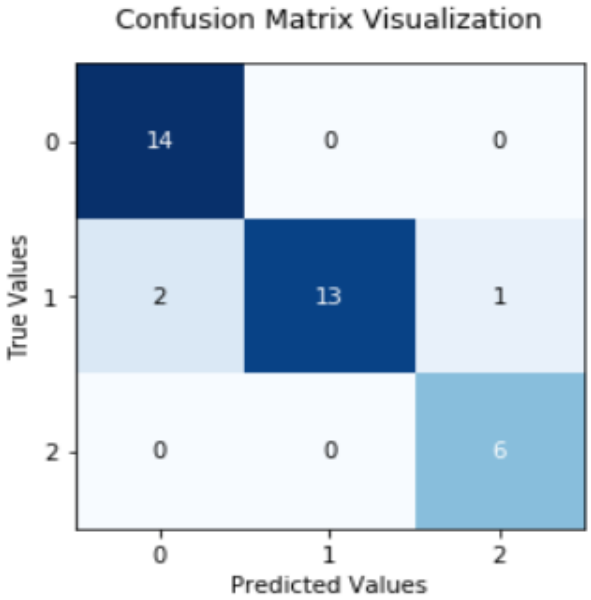

Figure 5.2: Visualization of Confusion Matrix

5.5.3 Evaluation Metrics

Accuracy

```
# Accuracy
accuracy = accuracy_score(y_test, prediction)
print("Accuracy: ", accuracy)
```

Accuracy output:

```
Accuracy: 0.9166666666666666
```

Precision, Recall, f1-score

```
# Precision, Recall, f1-score
metrics = metrics.classification_report(y_test,prediction, digits=3)
print(metrics)
```

Here is a summary of the precision, recall and f1-score for our three classes:

	precision	recall	f1-score	support
1	0.875	1.000	0.933	14
2	1.000	0.812	0.897	16
3	0.857	1.000	0.923	6
accuracy			0.917	36
macro avg	0.911	0.938	0.918	36
weighted avg	0.928	0.917	0.915	36

Figure 5.3: Classification Metrics Report

The *support* column lists the number of samples for each class.

Chapter 6

k-Nearest Neighbors

K-Nearest Neighbors (k-NN) algorithm is a type of supervised machine learning algorithm which can be used for both classification as well as regression predictive problems, nevertheless, it is mainly used for classification problems. The k-NN is defined as one of the simplest classification methods used in data mining. It is used when there are examples which have to be classified based on the class of their nearest neighbours. [28] K-NN classifier belongs to the category of lazy learners; it does not have a specialized training phase and uses all the data for training during classification. It is also called Example-Based Classification or Case-Based Classification because the classification is based directly on the training examples. [29]

6.1 Basics of k-Nearest Neighbors

K-NN algorithm uses the 'feature similarity' for the prediction of new data point values which means that the new data point will be assigned a value based on how closely it matches the points in the training set. Any record in a dataset is visualized as a point in an n-dimensional space, where n is the number of attributes. The k-NN algorithm tries to find the nearest training data point from an unseen test data point in multi-dimensional space. It calculates the distance between other training records to determine the class label of unknown record. The figure 6.1 represents how the k-NN works.

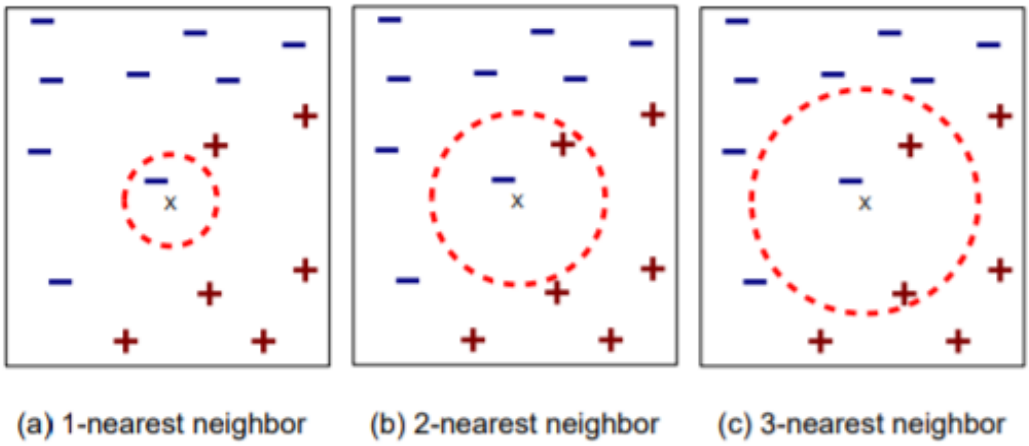

Figure 6.1: k-Nearest Neighbors [30]

1. Load the dataset.
2. Initialize k to your chosen number of neighbors.
3. For each point in the test data:
   (a) Calculate the distance between test data and each row of the training data with the help of any method such as: Euclidean, Manhattan or Hamming distance. The most commonly used method to calculate distance is Euclidean.
   (b) Add the distance and the index of the example in ascending order.
4. Pick the first top k rows from the sorted array entries from the sorted collection.
5. Assign a class to the test point based on most frequent class of these rows.

6.1.1 Distance

Distance metrics is a method to find distance between a new data point and existing training dataset. 3 distance metrics are explained below in detail.

Euclidean distance

The distance between two points X (x1, x2) and Y (y1, y2) in two-dimensional space is calculated by Euclidean distance as:

$$d = \sqrt{(x_1 - y_1)^2 + (x_2 - y_2)^2}$$

The distance d can be generalized for datasets with n attributes, where X is $(x_1, x_2, ..., x_n)$ and Y is $(y_1, y_2, ..., y_n)$, as:

$$d = \sqrt{(x_1 - y_1)^2 + (x_2 - y_2)^2 + ... + (x_n - y_n)}$$

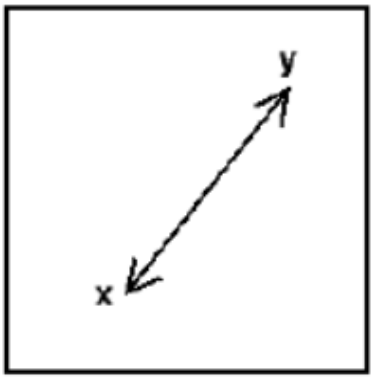

Figure 6.2: Euclidean distance

Euclidean distance implementation in python:

```
def euclidean_distance(x,y):
    return sqrt(sum(pow(a-b,2) for a, b in zip(x, y)))
```

Manhattan distance

The Manhattan distance function calculates the distance between two data points measured if a grid-like path is followed. The Manhattan distance between two items is the sum of their absolute differences.

The formula for this distance between a point $x = (x_1, x_2, .., x_n)$ and a point $y = (y_1, y_2, .., y_n)$ is:

$$d(x, y) = ||x - y|| = \sum_{n=1}^{d} |x_i - y_i|$$

Where n is the number of variables, and x_i and y_i are the values of the variables, at points x and y respectively. [34]

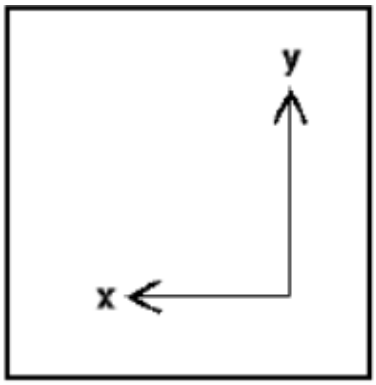

Figure 6.3: Manhattan distance

Manhattan distance is also known as Taxicab Geometry, City Block Distance etc. Implementation of the Manhattan distance:

```
def distancesum (arr, n):
    # sorting the array.
    arr.sort()
    # for each point, finding the distance.
    res = 0
    sum = 0
    for i in range(n):
        res += (arr[i] * i - sum)
        sum += arr[i]
    return res
def totaldistancesum( x , y , n ):
    return distancesum(x, n) + distancesum(y, n)
```

Euclidean distance vs Manhattan distance

Taxicab geometry versus Euclidean distance: In taxicab geometry (Manhattan distance), the red, yellow, and blue paths all have the same shortest path length of 12. In Euclidean geometry, the green line has length $6\sqrt{2} \approx 8.49$ and is the unique shortest path.

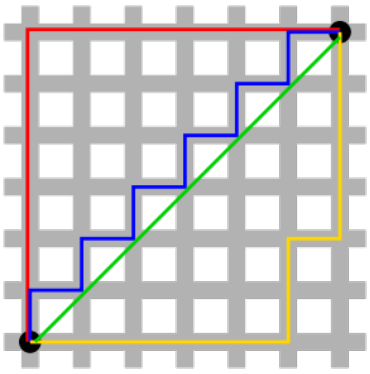

Figure 6.4: Source — Taxicab geometry Wikipedia

Hamming Distance

The Hamming distance is a metric for comparing two binary data strings of equal length where the distance is the number of bit positions at which the corresponding symbols are different. In other words, it measures the minimum number of substitutions required to change one string into the other.
The Hamming distance between two strings, x and y is denoted as $HamD_{(x,y)}$. [36]

$$HamD_{(x,y)} = \sum_{i=1}^{n} 1x_i \neq y_i$$

It is used in telecommunications as an error detection or error correction, to count the number of flipped bits in a fixed-length binary word as an estimation of error, and therefore it is sometimes called the signal distance. It is also applied in the coding theory to compare equal length data words, as well as in the document classification and image classification. [35]

The function **hammingDistance()** which is implemented in Python, computes the Hamming distance between two strings of equal length:

```
def hammingDistance(s1, s2) -> int:
    "Return the Hamming distance between equal-length sequences."
    if len(s1) != len(s2):
        raise ValueError("Undefined for sequences of unequal length.")
    return sum(el1 != el2 for el1, el2 in zip(s1, s2))
```

6.1.2 Choosing the right value for k

The k in the k-NN algorithm specifies the number of close training record(s) that need to be considered when making the prediction for an unlabeled test record. To select the right k that fits our data, we have to run the k-NN algorithm several times with different values of k and choose the one with the more accurate predictions on new data.

- When k=1, our predictions become less stable. The model tries to find the first nearest record and adopts the class label of the first nearest training record as the predicted target class value.
- As we increase the value of k, our predictions become more stable due to the majority class of the nearest training records, and thus, more it is likely to make more accurate predictions (up to a certain point). Eventually, we begin to witness an increasing number of errors. It is at this point that we know we have pushed the value of k too far.
- In cases where the class of the target record is evaluated by voting, k is usually assigned an odd number for a two-class problem. [31]

6.2 Advantages & Disadvantages of k-Nearest Neighbors

Advantages

Below is a list of the advantages of k-NN machine learning algorithm:

- **k-NN is intuitive, simple and very easy to implement.** There are only two parameters required to implement the k-NN, the value of k and the distance function (e.g. Euclidean or Manhattan etc.)
- **No Training Step.** k-NN does not explicitly build any model. It stores the training dataset and learns from it only at the time of making real time predictions. This makes the k-NN algorithm much faster than other algorithms that require training.
- **Can be used both for Classification and Regression.** The algorithm is versatile. It can be used for classification as well as for regression.

Disadvantages

Even though k-NN has several advantages, it has a few limitations. Below are listed some cons of k-NN:

- **It does not work well with large datasets.** In large datasets, the cost of calculating the distance between the new point and each existing point is huge. Therefore, while a dataset grows, efficiency or speed of algorithm declines very fast.
- **It does not work well with high dimensions.** The k-NN algorithm works better with small number of input since with large number of dimensions, it becomes difficult for the algorithm to calculate the distance in each dimension.
- **k-NN needs homogeneous features.** Before applying the k-NN algorithm to any dataset, feature scaling (standardization and normalization) needs to be applied.

- **Sensitive to outliers.** k-NN algorithm is very sensitive to outliers.

6.3 Applications of k-Nearest Neighbors

The k-NN algorithm has a wide variety of applications in classification as well as in regression. Some of the applications are mentioned below:

6.3.1 Text classification

The k-NN algorithm is one of the most popular algorithms for text categorization or text mining. The algorithm determines the class of the given text based on the document that is closer to it as well as the categories of the k documents. [32]

6.3.2 Finance

The k-NN algorithm could be used in financial modeling as a data mining technique and a process of discovering useful patterns and correlations. Some applications are used to discover uncovered market trends, planning investment strategies etc. [33]

6.4 k-Nearest Neighbors Implementation

6.4.1 k-NN model

Implementation of k-Nearest Neighbors using scikit-learn library:

```
# Load dataset
dataset = pd.read_csv('/content/data/WineWithHeader.csv')


# Retrieve rows from dataset
X = dataset.iloc[:,1:]
y = dataset.iloc[:,0]


# Feature Scaling
slc= StandardScaler()
```

```
X = slc.fit_transform(X)

# Spliting data
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.2,
    random_state = 0)

# Define the model
model = KNeighborsClassifier()

# Train the model
model.fit(X_train, y_train)

# Predict
prediction = model.predict(X_test)
```

6.4.2 Confusion Matrix

Corresponding confusion matrix of LR classifier:

```
cm = confusion_matrix(y_test, prediction)

print("Confusion Matrix:\n", cm)

fig, ax = plot_confusion_matrix(conf_mat=cm)

plt.ylabel("True Values")
plt.xlabel("Predicted Values")
plt.title("Confusion Matrix Visualization")
plt.show()
```

Output of the above block of code:

```
Confusion Matrix:
```

```
[14 0  0]
[0  15 1]
[0  0  6]
```

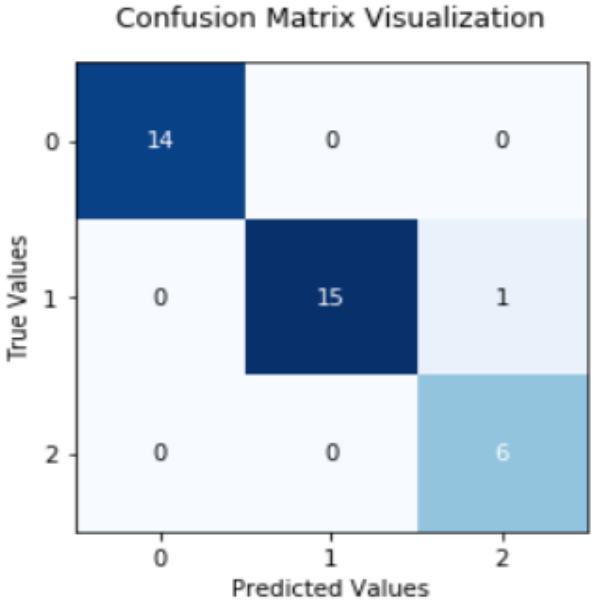

Figure 6.5: Visualization of Confusion Matrix

6.4.3 Evaluation Metrics

Accuracy

```
# Accuracy
accuracy = accuracy_score(y_test, prediction)
print("Accuracy: ", accuracy)
```

Accuracy output:

```
Accuracy: 0.9722222222222222
```

Precision, Recall, f1-score

```
# Precision, Recall, f1-score
metrics = metrics.classification_report(y_test,prediction, digits=3)
print(metrics)
```

Here is a summary of the precision, recall and f1-score for our three classes:

	precision	recall	f1-score	support
1	1.000	1.000	1.000	14
2	1.000	0.938	0.968	16
3	0.857	1.000	0.923	6
accuracy			0.972	36
macro avg	0.952	0.979	0.964	36
weighted avg	0.976	0.972	0.973	36

Figure 6.6: Classification Metrics Report

The *support* column lists the number of samples for each class.

Chapter 7

Support Vector Machine

Support Vector Machine (SVM) is a supervised learning algorithm which can be used for both classification and regression. Support Vector Machine first showed up in 1992, introduced by Boser, Guyon and Vapnik in COLT-92.

7.1 Basics of SVM

SVM algorithm is suitable for binary classification tasks. Additionally, SVM tries to find the hyper plane in a N-dimensional feature space, in order to produce classifiers, where N is the number of features.

7.2 Margin Maximization

The separation of data could be achieved with a lot of hyper planes. Despite this fact, only one hyper plane has the maximum margin.

$(w.x + b) = 1$ is defined for positive class and $(w, x + b) = -1$ for negative class. In figure 7.1 the region between the two hyper planes is called margin band and given by $\frac{2}{\|w\|^2}$.
The hyper plane $(w.x + b) = 0$ separates the positive and negative examples using the output of a decision function. If the output is greater than 1, it is classified as positive and if the output is -1 it is classified as negative.

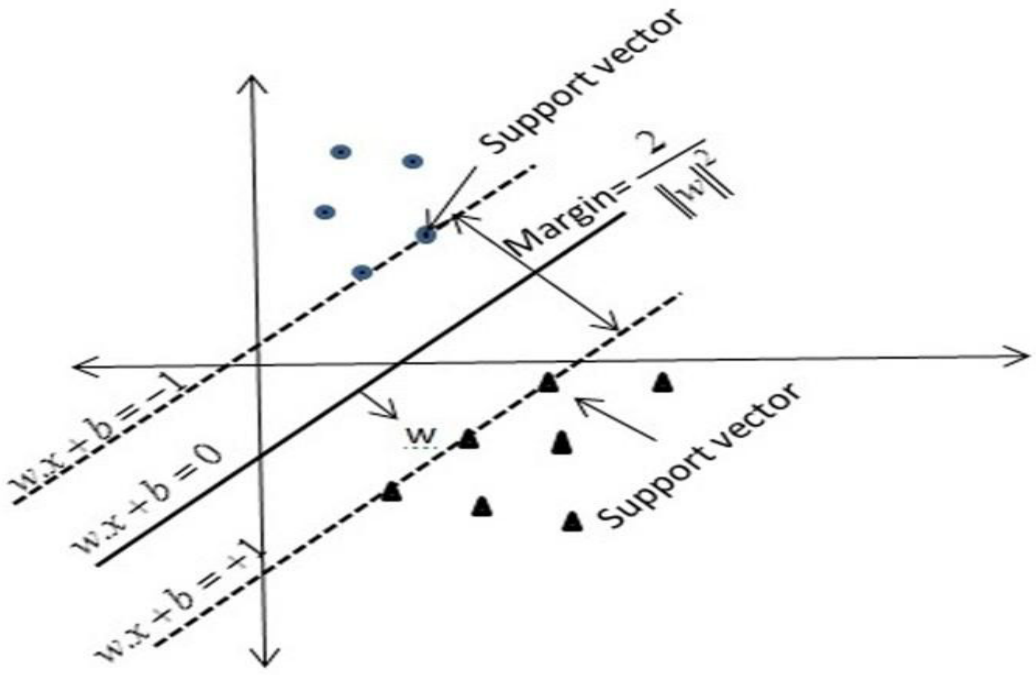

Figure 7.1: K nearest neighbor [37]

$$f(x) = sign(w, x + b), \text{where } sign(x) : \begin{cases} -1, & x < 0 \\ 0, & x = 0 \\ 1, & x > 0 \end{cases}$$

The optimization for the calculation of w and b can thus be expressed by:

$$min \frac{1}{2} \parallel w \parallel + C \sum_{i=1}^{m} \xi_i$$

$$u_i(w.x_i + b) \geq 1, \forall i = 1, 2, 3, ..., m$$

$$\xi_i \geq 0$$

The parameter C, "Capacity", is a tuning parameter which weights classification errors.

7.3 Advantages & Disadvantages of SVM

Advantages

The advantages can be summarized as follows [38]:

- SVM works well when there is a clear margin of separation between classes.
- SVM is more effective in high dimensional spaces.
- SVM is effective in cases where the number of dimensions is greater than the number of samples.
- SVM is relatively memory efficient, because it uses only a subset of training points in the decision function.

Disadvantages

Here are the disadvantages of SVM [38]:

- SVM is not suitable for large data sets because the required training time is higher.
- SVM does not perform very well when the data set has a lot of noise, due to overlappings in target classes.

7.4 Applications of SVM

The SVM algorithm is used in various real-life applications. [39]

- **Text and hypertext categorization.** Using training data to classify text documents into predefined categories based on their content.
- **Handwriting recognition**. SVM can be used to recognize handwritten characters.
- **Image classification**. The task of image classificaiton can also be performed using SVM.

7.5 Support Vector Machine Implementation

7.5.1 SVM model

Implementation of Support Vector Machine using scikit-learn library:

```
# Load dataset
dataset = pd.read_csv('/content/data/WineWithHeader.csv')


# Retrieve rows from dataset
X = dataset.iloc[:,1:]
y = dataset.iloc[:,0]


# Feature Scaling
slc= StandardScaler()
X = slc.fit_transform(X)


# Spliting data
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.2,
    random_state = 0)


# Define the model
model = SVC()


# Train the model
model.fit(X_train, y_train)


# Predict
prediction = model.predict(X_test)
```

7.5.2 Confusion Matrix

Corresponding confusion matrix of SVM classifier:

```
cm = confusion_matrix(y_test, prediction)
```

```
print("Confusion Matrix:\n", cm)

fig, ax = plot_confusion_matrix(conf_mat=cm)

plt.ylabel("True Values")
plt.xlabel("Predicted Values")
plt.title("Confusion Matrix Visualization")
plt.show()
```

Output of the above block of code:

```
Confusion Matrix:
    [14 0  0]
    [2  13  1]
    [0  0   6]
```

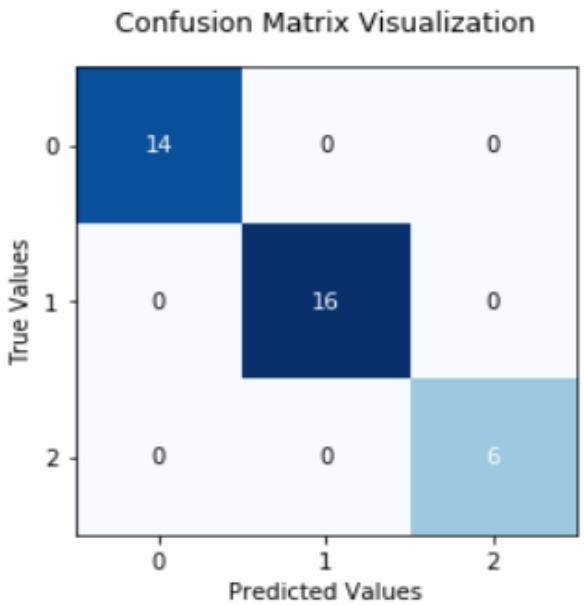

Figure 7.2: Visualization of Confusion Matrix

7.5.3 Evaluation Metrics

Accuracy

```
# Accuracy
accuracy = accuracy_score(y_test, prediction)
print("Accuracy: ", accuracy)
```

Accuracy output:

```
Accuracy: 1.0
```

Precision, Recall, f1-score

```
# Precision, Recall, f1-score
metrics = metrics.classification_report(y_test,prediction, digits=3)
print(metrics)
```

Here is a summary of the precision, recall and f1-score for our three classes:

	precision	recall	f1-score	support
1	1.000	1.000	1.000	14
2	1.000	1.000	1.000	16
3	1.000	1.000	1.000	6
accuracy			1.000	36
macro avg	1.000	1.000	1.000	36
weighted avg	1.000	1.000	1.000	36

Figure 7.3: Classification Metrics Report

The *support* column lists the number of samples for each class.

Conclusion

Overall, this paper demonstrates an overview of various classification algorithms along with very useful insights exported by evaluation metrics. In the current thesis, fundamental meanings such as Big Data, Machine Learning, Data Science and AI have been defined. The thesis primarily focused on a descriptive analysis of various classification algorithms which were finally presented with a python implementation. Each algorithm classifies a wine dataset and applies some metrics to evaluate the performance of every of them, using the same dataset. Each algorithm has advantages and disadvantages which are applicable depending on the input data.

References

[1] Vijay Kotu, Bala Deshpande. "Data Science Concepts and Practice."

[2] Stephen Kaisler, Frank Armour, J. Alberto Espinosa, William Money. "Big Data: Issues and Challenges Moving Forward." 2013 46th Hawaii International Conference on System Sciences

[3] McCarthy J, Minsky ML, Rochester N, Shannon CE. "A proposal for the dartmouth summer research project on artificial intelligence" August 31, 1955. AI Magazine 2006;27:12

[4] Samuel, Arthur (1959). "Some Studies in Machine Learning Using the Game of Checkers". IBM Journal of Research and Development. 3 (3): 210–229. CiteSeerX 10.1.1.368.2254. doi:10.1147/rd.33.0210.

[5] Mitchell, T. (1997). "Machine Learning." McGraw Hill. p. 2. ISBN 978-0-07-042807-2.

[6] Vincent François-Lavet, Peter Henderson, Riashat Islam, Marc G. Bellemare and Joelle Pineau (2018). "An Introduction to Deep Reinforcement Learning." Foundations and Trends in Machine Learning: Vol. 11, No. 3-4. DOI: 10.1561/2200000071.

[7] data (n.) `https://www.etymonline.com/word/data`

[8] Shan-Hung Wu & DataLab. "Cross Validation & Ensembling." `http://www.cs.nthu.edu.tw/~shwu/courses/ml/labs/08_CV_Ensembling/08_CV_Ensembling.html`

[9] Training, validation, and test sets. https://en.wikipedia.org/wiki/Training,_validation,_and_test_sets

[10] About Train, Validation and Test Sets in Machine Learning. https://towardsdatascience.com/train-validation-and-test-sets-72cb40cba9e7

[11] Wine Data Set. UCI, Machine Learning Repository. http://archive.ics.uci.edu/ml/datasets/Wine

[12] Shaeke Salman and Xiuwen Liu. "Overfitting Mechanism and Avoidance in Deep Neural Networks." arXiv:1901.06566v1 [cs.LG] 19 Jan 2019

[13] Underfitting vs. Overfitting. https://scikit-learn.org/stable/auto_examples/model_selection/plot_underfitting_overfitting.html

[14] Galton, Francis. "Kinship and Correlation (reprinted 1989)". Statistical Science. 4 (2): 80–86. doi:10.1214/ss/1177012581

[15] Xianghong Luo. "A Comparison of Three Estimation Methods In Linear Regression Analysis". Advances in Computer Science Research, volume 71. 4th International Conference on Machinery, Materials and Information Technology Applications (ICMMITA 2016)

[16] Park, Hyeoun-Ae. "An Introduction to Logistic Regression: From Basic Concepts to Interpretation with Particular Attention to Nursing Domain ". J Korean Acad Nurs Vol.43 No.2 April 2013 http://dx.doi.org/10.4040/jkan.2013.43.2.154

[17] W.A. Awad and S.M. ELseuofi. "MACHINE LEARNING METHODS FOR SPAM E-MAIL CLASSIFICATION". International Journal of Computer Science & Information Technology (IJCSIT), Vol 3, No 1, Feb 2011

[18] Raghavendra Patidar, Lokesh Sharma. "Credit Card Fraud Detection Using Neural Network". International Journal of Soft Computing and Engineering (IJSCE) ISSN: 2231-2307, Volume-1, Issue-NCAI2011, June 2011

[19] Xiaobo Zhou Kuang-Yu Liu Stephen T.C. Wong. "Cancer classification and prediction using logistic regressionwith Bayesian gene selection". Journal of Biomedical Informatics 37 (2004) 249–259

[20] Adarsh Anand, Gunjan Bansal. "Predicting Customer's Satisfaction (Dissatisfaction) Using Logistic Regression". International Journal of Mathematical, Engineering and Management Sciences Vol. 1, No. 2, 77–88, 2016

[21] Rish. "An empirical study of the naive Bayes classifier". T.J. Watson Research Cente

[22] Jean Dezert, Albena Tchamova, Deqiang Han. "Total Belief Theorem and Generalized Bayes' Theorem." 21st International Conference on Information Fusion (Fusion 2018), Jul 2018, Cambridge, United Kingdom. ffhal-01876332

[23] B. M. Gayathr, C. P. Sumathi. "An Automated Technique using Gaussian Naïve Bayes Classifier to Classify Breast Cancer." International Journal of Computer Applications (0975 – 8887) Volume 148 – No.6, August 2016

[24] Jason D. M. Rennie, Lawrence Shih, Jaime Teevan, David R. Karger. "Tackling the Poor Assumptions of Naive Bayes Text Classifiers." Artificial Intelligence Laboratory; Massachusetts Institute of Technology; Cambridge, MA 02139

[25] Andrew McCallum, Kamal Nigam. "A Comparison of Event Models for Naive Bayes Text Classification"

[26] Vangelis Metsis, Ion Androutsopoulos, Georgios Paliouras. "Spam Filtering with Naive Bayes – Which Naive Bayes?"

[27] Pouria Kaviani, Mrs. Sunita Dhotre. "Short Survey on Naive Bayes Algorithm." International Journal of Advance Engineering and Research Development Volume 4, Issue 11, November -2017

[28] Altman, N. S.. "An Introduction to Kernel and Nearest Neighbor Nonparametric Regression." The American Statistician. 46 (3): 175–185 (1992).

[29] E. McKenna and B. Smyth. "Competence-guided editing methods for lazy learning." In W. Horn, editor, ECAI 2000, Proceedings of the 14th European Conference on Artificial Intelligence, pages 60–64. IOS Press, 2000.

[30] Rashmi Agrawal. "K-Nearest Neighbor for Uncertain Data." International Journal of Computer Applications (0975 – 8887) Volume 105 – No. 11, November 2014

[31] Peterson, L. k-Nearest neighbors. Scholarpedia. (2009). Retrieved from `http://www.scholarpedia.org/article/K-nearest_neighbor`.

[32] Lijun Wang, Xiqing Zhao. "Improved KNN classification algorithms research in text categorization"

[33] Sadegh Bafandeh Imandoust And Mohammad Bolandraftar. "Application of K-Nearest Neighbor (KNN) Approach for Predicting Economic Events: Theoretical Background." S B Imandoust et al. Int. Journal of Engineering Research and Applications Vol. 3, Issue 5, Sep-Oct 2013, pp.605-610

[34] Mrs .Mahananda D. Malkauthekar. "Analysis of Euclidean Distance and Manhattan Distance Measure in Face Recognition"

[35] Norouzi, M., Fleet, D.J., Salakhutdinov R.R.: "Hamming distance metric learning." In: Advances in Neural Information Processing Systems (NIPS). 061–1069 (2012)

[36] V. B. Surya Prasath, Haneen Arafat Abu Alfeilate, Ahmad B. A. Hassanate, Omar Lasassmehe, Ahmad S.Tarawnehf, Mahmoud Bashir Alhasanatg,h, Hamzeh S. Eyal Salmane. "Effects of Distance Measure Choice on KNN Classifier Performance - A Review."

[37] Krupal S. Parikh, Trupti P. Shah. "Support Vector Machine – a Large Margin Classifier to Diagnose Skin Illnesses." 3rd International Conference on Innovations in Automation and Mechatronics Engineering ICIAME 2016

[38] Laura Auria and Rouslan A. Moro. "Support Vector Machines (SVM) as a Technique for Solvency Analysis."

[39] Vikramaditya Jakkula. "Tutorial on Support Vector Machine (SVM)."

[40] Hossin, M. and Sulaiman, M.N. "A Review on Evaluation Metrics for Data Classification Evaluations." International Journal of Data Mining & Knowledge Management Process (IJDKP) Vol.5, No.2, March 2015

[41] Confusion matrix `https://en.wikipedia.org/wiki/Confusion_matrix`

[42] Jesse Davis, Mark Goadrich. "The Relationship Between Precision-Recall and ROC Curves"