UNIVERSITY OF MACEDONIA
DEPARTMENT OF APPLIED INFORMATICS
MSC IN ARTIFICIAL INTELLIGENCE AND DATA ANALYTICS

DYNAMIC / ADAPTIVE DETERMINATION OF PARAMETER k IN k-NN CLASSIFIER

A dissertation
by

Merkourios Papanikolaou

Thessaloniki, October 2022

DYNAMIC / ADAPTIVE DETERMINATION OF PARAMETER k IN kNN CLASSIFIER

Merkourios Papanikolaou

Dipl. Civil Engineer, MEng, MSc AUTH, 2016

Dissertation

Submitted in partial fulfilment of the requirements for
THE MSC IN ARTIFICIAL INTELLIGENCE AND DATA ANALYTICS

Supervisor Professor
Georgios Evangelidis

Approved by the three-member Examination Committee on 14/10/2022

Georgios Evangelidis	Georgia Koloniari	Konstantinos Diamantaras
.....	.....	.....

Merkourios Papanikolaou

.....

Abstract

K – Nearest Neighbor (k-NN) classifier is one of them most widely used supervised learning algorithms. Its popularity is mainly due to its simplicity, effectiveness, ease of implementation and ability to add new data in the training set at any time. However, one of the major drawbacks of this algorithm is the fact that its performance mainly depends on the parameter value k, i.e. the number of nearest neighbors that the algorithm examines in order to classify the unlabeled instance. In most cases, it is a fixed value, independent of data distribution. The most frequently used technique for the "best" k determination is the cross validation as there is no general rule for choosing the k value due to its dependency on the training dataset. A large k value results in a noise tolerant classifier, as its search area is large. On the other hand, a small k value results in a noise sensitive classifier, as the search area is limited. So, selecting a fixed k value throughout the dataset does not take into account its special features, like data distribution, class separation, imbalanced classes, sparse and dense neighborhoods and noisy subspaces.

In recent years, a lot of research have been conducted in order to tackle the above-mentioned disadvantage. The research has led to various approaches of k-NN classifier, which mainly combine it with various other techniques for k value determination. In the present study, a thorough literature review is conducted in order to summarize all the achievements made to date in this field. This procedure led to a pool of twenty-eight (28) publications, covering a time period from 1986 till 2020 (with median value 2009). These studies are presented in this work, describing the techniques used for dynamic k determination. For each study, several indicators are recorded, namely the technique used for k selection, the level of k selection, the number of datasets used for experiments, whether statistical tests were conducted or not, the total number of citations each research has received as well as the average citations per year.

Apart from the above, a new alternative version of k-NN algorithm is proposed. The proposed algorithm is an extension of a previous work, found in the literature. The approach is a k-free k-NN variation, in the sense that the user does not select the parameter, but it is selected dynamically, depending on the area where each unlabeled data point lies. The algorithm falls into the group of the studies that exploit prototype and clustering techniques in order to represent the initial dataset. Through a recursive process, homogenous clusters are created, each of which are represented by a representative.

Moreover, a new term is introduced, namely the Sphere of Influnce (SoI), an index which indicates the size of each created cluster. This index, in combination with the indicator depth (d), provides useful information about the subspace that each representative lies. Finally, heuristics are proposed in order to exploit the information provided from SoI and d and convert it in a k value, unique for every unlabeled instance.

Extensive experiments were conducted on thirty (30) datasets for all proposed heuristics. Some of these datasets contained artificial noise in order to test the proposed algorithm in real life situations. The experiments showed a very competitive performance – in terms of accuracy – of the proposed algorithm in comparison with some commonly used k-NN variations. Moreover, Wilcoxon statistical test was used to find statistically significant differences.

Keywords: supervised machine learning, classification, k-nearest neighbor (k-NN), Dynamic k value determination, heuristics

Table of contents

List of Figures (if any)

List of tables (if any)

1 Introduction

1.1 Problem – Importance of the topic

Nowadays, supervised machine learning is one of the most frequently used technique, exploited among a wide variety of industries, for example epidemiology – especially with the recent COVID-19 outbreak - (Alballa & Al-Turaiki, 2021), sales forecasting (Rohaan et al., 2022), stocks market (Sakhare & Sagar Imambi, 2019), Architecture, Engineer and Construction (AEC) (Kifokeris & Xenidis, 2019), etc. Its use is mainly focused on making insightful predictions, which, in turn, will be the cornerstone of the so-called data driven decision making. The utmost importance of these algorithms, as they affect serious decisions, bring them to the forefront of the research community, which in turn tries to make them more and more accurate.

Supervised machine learning can split into two main subbranches; Classification and regression. Both techniques rely on the idea of "learning from examples". A brief description of these two terms is following.

1.1.1 Regression

Regression is mainly a statistical process. During this process, a relationship model is structured between the dependent and the independent variables. The output of a regression model is a real or a continuous value. For example, the prediction of a stock price given the price of the previous days is a regression problem, as the output variable is a real number.

On the contrary, predicting whether a stock price will increase, or decrease is not a regression problem, as the output variable will be categorical (yes / no). Among the most common regression algorithms are linear regression, polynomial regression, and support vector regression.

1.1.2 Classification

On the other hand, classification algorithms or classifiers, as they are called, also build a model, based on observed values, whose output is categorical value. More specifically, classifiers attempt to recognize, understand, and group objects or data instances into preset categories. Using a set of instances - training/test dataset - for which the category or class label is known, they attempt to build/train a model that can successfully predict the class label of unlabeled instances. For example, predicting whether

an incoming e-mail is spam or not is a classification problem, as the output variable will be categorical (yes / no).

In turn, classifiers are divided into two main categories; eager and lazy ones (Bulut & Amasyali, 2017). Eager learners build a generalized model based on the training instances that is used for making predictions, without storing each instance in memory. On the other hand, lazy classifiers usually store in memory all the instances and conduct a local search. In other words, the instances are both the training dataset and the model at the same time.

1.2 K-NN Classifier

One of the widely used classification algorithms is k-Nearest Neighbors (k-NN) (Cover & Hart, 1967). It belongs to the lazy classifiers category. Its popularity is due to (i) its simplicity, as there is no model to train, (ii) its effectiveness, as the asymptotic classification error of 1-NN is bounded by twice the Bayesian error rate, (iii) its ease of implementation, as it is mainly based on distance computations, and, (iv) its ability to add new training data at any time as there is not an already trained model, an extremely useful characteristic, especially, for data streams (Bulut & Amasyali, 2017; Mullick et al., 2018).

Specifically, k-NN predicts the class label of an unknown instance by conducting a local search among its k nearest neighbors and then applying a majority voting; the unknown instance is labelled with the class of the majority of the k nearest neighbors. Usually, local search is based on Euclidean distance. Other, frequently used, distance metrics are Minkowski, Cosine similarity, Manhattan, Jaccard and Hamming.

Based on the above, one can easily conclude that a major drawback of k-NN, and generally of all lazy learners, is the high demand of both storage space, as they store all instances in memory, and computational resources, especially in case of high dimensional datasets. Figure 1-1 depicts an additional drawback of k-NN; the fact that its performance is highly dependent on value k, which determines the extent of the neighborhood that the search is taking place (Johansson, Boström, et al., 2008; Ougiaroglou et al., 2020). Even a slight variation in k value may affects the classification outcome.

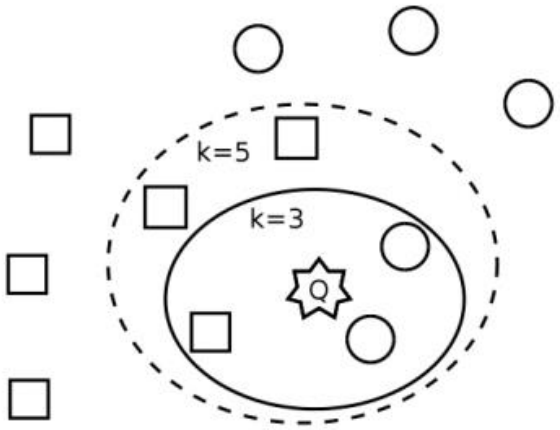


Figure 1-1: Classification process using k-NN algorithm with k=3 and k=5

In recent years, a lot of research have been conducted in order to tackle the above-mentioned disadvantages. In practice, k is treated as a hyper-parameter and, consequently, the most frequently used technique for the "best" k determination is the cross validation as there is no general rule for choosing the k value due to its dependency on the training dataset. However, the k that is finally determined with the use of cross validation is a unique and fixed value for the whole dataset without taking into account the specific and unique features that each dataset may have as well as its distribution. For example, in cases either of classes that are not well separated or of noisy instances, a large k value may be more suitable in order to examine an extensive subspace (neighborhood). A large k value results in a noise tolerant classifier, as its search area is large. On the contrary, in case of distinct classes, a large k value may result both in higher computational cost and in accuracy deterioration. In such cases, a small k value may be more appropriate. A small k value results in a noise sensitive classifier, as the search area is limited.

However, the problem becomes more and more complex in cases of real-life datasets that may simultaneously contain both well and not-well separated classes, imbalanced classes, sparse and dense neighborhoods, and noisy sub-spaces (Ougiaroglou et al., 2020). Based on the above, one should consider that a globally defined fixed k value is not appropriate for a dataset. Instead, one should take into account the special features of the dataset and the subspace that each instance lies into and try to dynamically determine a local k value for each instance to be classified. Thus, the research has led to various

approaches of k-NN classifier, which mainly combine it with various other techniques for k value determination.

1.3 Aim – Objectives

This study has dual research objective. On the one hand, a thorough literature review is conducted in order to gather and compare (according to comparison axes that will be defined later in this study) the existing k-NN implementations and the related alternative versions. Worth noting that the dissertation is not restricted only to the highly cited state-of-art publications but is extended to almost all the existing approaches for dynamic k value determining found in the literature.

On the other hand, a new alternative version of k-NN algorithm is proposed, based on some findings of the literature review. Under this alternative, parameter "k" – a key parameter of k-NN classifier which mainly determines algorithm's performance – will not be a constant anymore but it will be adapted dynamically, exploiting heuristics, and taking into account local variation within each dataset. This modified k-NN version aims at outperforming the original k-NN algorithm.

1.4 Contribution

The contribution of this study is highly connected with its objectives. Thus, the first contribution is that, through the thorough literature review, the research community will be given the opportunity to have a clear picture of the achievements made to date as far as far as the dynamic k value determination is concerned and, probably, inspiring and guiding their future efforts. Moreover, in the context of this study, both the most influencing and the most completed approaches will be highlighted.

On the other hand, another approach of dynamic k-NN algorithm is proposed and tested with experiments. This contributes to the researchers' efforts transitioning from a "static" to a "dynamic" k-NN algorithm, which will consider the majority of the unique features each dataset may have.

1.5 Structure of the study

Studies related to the topic of the dissertation, as well as the methodology used for the literature review are presented in Chapter 2. The Chapter 3 discusses the proposed alternative version of k-NN. The same chapter also includes the theoretical background

behind it as well as the experiments conducted for testing it. Finally, Chapter 4 discusses the findings and concludes to useful remarks.

2 Literature review – Theoretical background

2.1 Introduction – Methodology

As mentioned before, the first research objective of this study is to gather together all the proposed approaches for dynamic k value determination when applying k-NN classifier. It is worth mentioning that the content of this chapter have been published in (Papanikolaou et al., 2021). The adopted research methodology consisted of the following steps.

The first step included a thorough literature search for potential references in academic bibliographic databases and search engines that are connected with scientific publications (e.g., Springer Link, Mendeley, Science Direct, Google Scholar, Scopus, Wiley Online, Emerald Insight, etc.).

The second step included the one-by-one examination of the references included in the documents collected in step one.

The final step involved the in-depth study of the most relevant with the research objective references.

The successful execution of the first two steps led to the collection of about sixty five (65) publications. The in-depth study of these sources (step three) led to a pool of twenty eight (28) publications that either propose a new methodological framework for dynamic k determination or refer to the need of a more adaptive k value by proposing, for example, alternatives where the classification process is independent of k. Seven out of twenty eight publications refer to global k value (in other words, they treat k selection as a hyper-parameter tuning process, choosing one k value for the whole dataset) while the rest twenty one two examine it locally (i.e., for each test instance or for each distinct sub-space of the whole dataset).

Then, the publications extracted from step three are grouped according to the methodological framework they are based on. In total, six (6) groups were identified, namely mapping to genetic algorithms, neural networks, prototypes or clustering, heuristics, probabilistic and the group “other”, which includes any other approach that does not fit into the previous categories.

Apart from the above, for each study were recorded the number of data sets used for experimenting and testing, whether or not statistical tests were conducted as well as the number of citations and the average citations per year. The last two indexes will help to

highlight the most influential approaches. All the above are summarized in Table 2-1, which presents the findings of this chapter.

All the above-mentioned process for literature review was conducted between December 2020 and February 2021.

2.2 Global approaches

Kardan et al. (2013) propose an approach that is based on a genetic algorithm, namely Biogeography based optimization (BBO). This algorithm uses a multi-part chromosome for the simultaneous optimization of feature selection, feature weighting and the k value. The proposed algorithm was compared with six evolutionary and fourteen non-evolutionary genetic algorithms, on 10 different datasets. After conducting experiments, the BBO seems to outperform all the compared algorithms, in terms of accuracy rate, Kohen's Kappa and reduction rate, across all datasets.

A heuristic based k-NN variation is presented by Ferrer-Troyano et al. (2001), where the k value is selected automatically, without any user intervention. The heuristic is based on the idea that the algorithm will search for that k value that correctly classifies the majority of training instances. The proposed approach was compared, in terms of accuracy, to conventional k-NN, with $k \in [1, 51]$ (only odd numbers) on 25 datasets. In thirteen out of twenty cases, the proposed algorithm outperformed the widely used 1-NN. Moreover, in five out of the thirteen above mentioned cases, the difference was statistically significant.

Despite the fact that the approach presented by Nock et al., (2003) does not propose a different procedure for k selection, the described heuristic makes the performance of conventional k-NN less dependent from the selected k value. Briefly, the idea behind this approach is that (for a given k) if x votes for y, then y also votes for x, even if x does not belong to y′s k nearest neighbors. The proposed algorithm was compared, in terms of accuracy, to 1-NN and 1-tNN, over 29 datasets. The results demonstrated that the proposed algorithm outperformed the rest ones without statistical significance.

Holmes & Adams (2002) presented an analytic probabilistic k-NN variation which deals with the uncertainty on k using a prior distribution on it. The proposed approach was compared, in terms of classification error rate on a variety of datasets, to conventional k -NN algorithm, demonstrating competitive performance.

The approach presented by Hamerly & Speegle (2012) does not propose a varied procedure for k selection but a faster cross-validation technique in order to examine a larger

amount of k values within the same running time, reducing the time complexity by O(K∗), where K∗ is the maximum k value. The proposed technique was tested on 3 datasets demonstrating its contribution in running time reduction.

A quite older empirical approach is presented Enas & Choi (1986). The authors argue that the optimal k value is dependent on the dimensions, the size and the structure of the sample size. Moreover, they propose some equations for k computation in function with the difference between sample proportions and the difference between con-variance matrices. No experimental tests are reported.

A general discussion about the k selection in problems with binary imbalanced class is made in another study (Hand & Vinciotti, 2003). Specifivally, in this paper the authors cite some proposed equations for k approximation. The most popular of the are k $\approx n^{\frac{2}{8}}$ or k $\approx n^{\frac{3}{8}}$

2.3 Local Approaches

2.3.1 Genetic Algorithms Approaches

An approach utilizing genetic algorithms is presented by Johansson, König, et al. (2008). In this work, authors propose a genetic alternative of conventional k-NN, namely G-k-NN, which is used for the optimization of the k value. Specifically, this technique builds decision trees that split the search space into distinct regions. Then, a fixed, unique and optimized k value is assigned to each region. In turn, this k value is assigned to each unlabeled instance (test instance), according to the position it lies in input space and the region it belongs to. The search for the k nearest neighbours can be performed either locally - within the limits of the boundaries of each region - or globally - across all regions - or in a mixed way - both globally and locally. The three alternatives of the proposed algorithm - local G-k-NN, global G-k-NN and mixed G-k-NN - were compared, in terms of accuracy, to the classic k-NN for k = 5, k = 11 and k = 11, on 27 different datasets. As far as the statistical significance is concerned, global G-k-NN significantly outperforms the three versions of conventional k-NN. However, this is not the case for local G-k-NN.

2.3.2 Neural Networks Approaches

A quite interesting approach is presented by Mullick et al. (2018). The authors of this study present two alternative versions of classic k-NN; the Adaptive k-NN (Ada-k-

NN) and the the Adaptive k-NN2 (Ada-k-NN2). Both methods utilize the data density and distribution in order to find an appropriate k value for each unlabeled instance. On the one hand, Ada-k-NN uses artificial neural networks in order to learn this suitable k. On the other hand, Ada-k-NN2 is a simplified version in the light of using a heuristic as indicator of the local density around the unlabelled instance. The heuristic is based on the idea of searching for that k that correctly classifies the majority of test instance's neighbors. Two sets of experiments were conducted. The first set was oriented in comparing the algorithms with eight other classifiers on 17 datasets. The datasets were divided into two categories; the small/medium ones and the large ones. The second set was oriented in comparing the proposed algorithms in the presence of imbalanced classes. As far as the first set, the results revealed that Ada-k-NN2 outperformed the other classifiers in terms of average accuracy. It, also, achieved the best Average Rank for Small and Medium-Scale datasets. Respectively, Ada-k-NN attained the fifth place. In case of large datasets, Ada-k-NN2 attained the second place in terms of average accuracy and the first place in terms of Average Rank for Large-Scale dataset. On the contrary, Ada-k-NN attained the sixth and eighth place, respectively, which shows that it suffers from scalability. As far as the second set is concerned, Ada-k-NN2 in combinations with GIHS - a simple weighting scheme - outperformed the rest classifiers.

2.3.3 Prototypes and Clustering based Approaches

An interesting approach is presented Garcia-Pedrajas et al. (2017) where the search space is divided into sub-spaces. Each sub-space constitutes a neighbor which, in turn, is represented by a prototype. Each prototype is assigned with a k value, which is considered to be the optimum within the neighbor borders. A greedy approach is adopted in order to find these optimum k values; for each prototype the local performance is tested, with k varying within an interval [kmin, kmax], with most common values kmin = 1 and kmax = 100. Eventually, the k value with the highest performance is assigned to the corresponding prototype. Then, the classification process for each unlabeled (test) instance is conducted using the k value of its nearest prototype (training) instance. Given that the proposed approach can be adopted in almost every k-NN variation, the authors conducted experiments using the standard k-NN, the adaptive k-NN and the symmetrical k-NN in combination with the proposed algorithm for k selection versus the same k-NN variations in combination with ten-fold cross validation for k selection. In both cases, the interval

was set to kmin = 1 and kmax = 100. The experiments were conducted using 80 datasets for regular problems and 65 datasets for imbalanced problems. The algorithms were compared in terms of accuracy and Kohen's Kappa (for regular datasets) and G-mean and auROC (for imbalanced datasets). The experiments results showed that the proposed algorithm performed better both for regular and imbalanced datasets.

An approach, based on clustering, is demonstrated by Bulut & Amasyali (2017), namely the One Nearest Cluster (1NC) approach. According to this approach, the user pre-defines the number of clusters (l) and picks the M closest samples around the unlabelled instance. All the distances among the test instance and the M closest samples are calculated and, according to the distance, all the M samples are laid on an one-dimensional axis, from the closest to the farthest. Then, clustering is applied in order to split the M samples into l clusters. Finally, the majority voting technique is applied within the closest cluster. That way, the approximation of k can be expressed as $k \approx \frac{M}{l}$. This procedure is repeated for every test instance. The proposed algorithm (with $k \approx \frac{15}{3}$) was compared with 1-NN and 5-NN, in terms of accuracy, on 36 datasets. Moreover, a T-test is performed in order to check statistically the pairs 1NC - 1-NN and 1NC - 5-NN. 1NC outperformed nine times, 1-NN ten times and 5-NN seventeen times. In terms of averaged accuracy over the 36 datasets, 1NC demonstrated the highest accuracy. The statistical test showed that 1NC outperforms 1-NN significantly. The authors argue that M and l should have small integer numbers because this leads not only to reduced computational time but also to the same results in comparison with larger values (e.g. $\frac{10}{2}$ gives the same outcome with $\frac{100}{20}$).

One more prototype and clustering based k-NN variation is presented by Guo et al. (2003). The fundamental idea behind this approach is that some representative instances will represent the whole training set. After the selection of representatives, a procedure based on the training set's local features which is beyond the scope of this study, the k value is chosen automatically - without user intervention - for each representative; k value equals to the number of instances covered by each representative. In other words, the local search for applying the majority voting technique, is conducted within the limits of the local neighborhood of each representative instance. The proposed algorithm was compared, in terms of accuracy, with C5.0 and k-NN over 6 datasets. Due to the fact that its average classification accuracy was the highest one (85, 15%) comparing to C5.0 (81,

35%) and k-NN (80,90% for k = 1, 83,52% for k = 3 and 83,67% for k = 5), its performance is considered satisfactory.

The concept of representing the whole training set with a few representative instances (prototypes) is farther exploited in the study of Ougiaroglou et al. (Ougiaroglou et al., 2020), who use the idea of homogeneous clusters. Specifically, in this approach, namely Subspace Homogeneity based Dynamic k-NN classifier (shd-k-NN), the authors apply repetitively the k-means clustering procedure until all created clusters are homogeneous. This repetitive procedure produces a kind of tree of clusters. Each leaf node represents a homogeneous cluster whose depth (d) provides information about the region where the unclassified instance lies. When an unlabeled instance needs to be classified, the algorithm finds the nearest homogeneous cluster's centroid and its corresponding d value. Then, the authors suggest five heuristics based on which the k is calculated as a function of d and apply the majority voting technique. Based on the above, it is easily concluded that this k-NN variation is independent of the "best" or the "optimum" k selection as it is dynamically selected for each instance in an automated way, without any user intervention. The proposed algorithm (using all different heuristics) was compared to some common used k-NN variations; the "best" k-NN (k was estimated using 5-fold cross validation), 1-NN, 5-NN, 10-NN,$\sqrt{N}$-NN and $\sqrt{\frac{N}{2}}$-NN, where N represents the number of instances included in training set. The comparison was made in terms of accuracy on 14 original datasets and 19 variations of them with random "noise" addition. Generally speaking, all heuristics demonstrate not only competitive performance to the "best" k-NN but also outperform it in some cases. Comparing the shd-k-NN to conventional k-NN (with fixed k), the experiments showed that the proposed algorithm's performance is better in almost all cases.

2.3.4 Heuristic based Approaches

The exploitation of heuristics is a quite common technique for conventional k-NN improvement. One such case is presented by Ougiaroglou et al. (2007). According to this study, the authors apply incremental computation of nearest neighbors in R-Trees. However, the nearest neighbor search breaks if some criteria induced by a heuristic are satisfied. Thus, each unlabeled instance is classified by a non-fixed k. The most important property of the incremental search is that the neighbors are discovered in their order of their distance from the query instance. This allows the discovery of the (k + 1)-th nearest

neighbor if we have already discovered the previous k nearest neighbors. Three different heuristics for k-NN early break, independently of the selected k value. This way, conventional k-NN preserves its simple concept and implementation while, simultaneously, reduces the required computational cost. Briefly, the three proposed heuristics for k-NN early break are the following:

(a) The majority voting technique breaks when a predefined threshold for the dominant class is met.

(b) The majority voting technique breaks when the remaining votes are less than the difference k - (current votes for the majority class). For example, it's pointless to continue searching in case of a binary classification problem, with k = 9 and after searching 7 neighbors, the dominant class has already 5 votes.

(c) The majority voting technique breaks when a predefined number of consecutive neighbours is met which all vote for the dominant class.

The aforementioned heuristics were tested on 2 datasets in terms of accuracy and computational cost. As far as the accuracy is concerned, the second heuristic outperforms the rest. As far as the the computational cost is concerned, the first heuristic outperforms the rest.

Another, quite similar approach, using a heuristic, is presented by Sun & Huang (2010). According to this k-NN variation, a different k value is assigned to each instance of the training set. This k value is equal to the minimum number of neighbours needed to be included in the majority voting procedure in order the instance in question be classified correctly. Then, for each unlabeled instance, the proposed algorithm finds its nearest neighbor and its assigned k value. The k-NN algorithm is the applied with this k value. The above-mentioned approach was compared, in terms of accuracy, to conventional k-NN, with $k \in [1, 9]$ and Ada-k-NN (Mullick et al., 2018) on 15 datasets. Its performance is considered competitive as it is classified second on six datasets and third on one dataset. Moreover, the authors support that the Ada-k-NN overall outperforms the rest conventional k-NN tested algorithms.

In an older study, Baoli et al. (2004) have developed a k-NN variation, oriented in text categorization. The proposed algorithm does not suggest a new procedure for k selection but develops a k-NN variation less dependent on the k value. Specifically, the authors suggest that the k should be proportional to the number of training instances that belong to the category the test instance is going to be classified. In order to classify a test

instance in a category with more training points, the algorithm should use a larger k value in comparison with classifying the same test instance in another category with less training points. The experiments conducted on 2 different text datasets revealed that the proposed variation is less sensitive to k selection. As a consequence, it can effectively deal with class imbalance.

2.3.5 Probabilistic Approaches

The concept of "spheres of confidence" is presented by Johansson, König, et al.(2008). According to this study, the authors exploit the notion of Laplace estimator $P_{\text{ClassA}} = \frac{\text{k+1}}{\text{N+C}}$, where, k is the number of training instance belonging to class A, N is the total number of training instances belonging within the sphere, and C is the total number of classes.

The cornerstone of this approach is that each training instance is surrounded by a sphere of confidence that includes other training instances. In order to construct the sphere of confidence, the algorithm searches the nearest neighbors one by one, starting from the closest one. The procedure breaks either when the Laplace estimator's value starts decreasing or when all training instances are examined. The first case is called "eager construction" and the generated sphere of confidence includes all the points examined by the algorithm until the break. The second case is called "total construction" and as sphere of confidence is selected that with the highest Laplace estimator value.

When an unlabeled instance needs to be classified, the algorithm applies the majority voting technique either among all instances within the spheres covering the test instance (instance aggregation) or among all spheres covering the test instance (sphere aggregation), considering each sphere as one class (the class of the majority instances within the sphere).

The above-mentioned approach was compared, in terms of accuracy and auROC, to the conventional k-NN with fixed values k = 5, 11, 17 on 18 datasets. Briefly, the results showed that the proposed algorithm significantly outperformed the rest ones, demonstrating better performance on thirteen out of eighteen datasets.

The concept of exploiting the local data distribution is also utilized by the approach presented by Bhattacharya et al. (2014, 2015). The authors, in both studies, suggest the construction of a hypersphere around each unlabeled point in order to capture the distribution of the training instances around it. Moreover, they introduce the concept of

hubness weight, i.e., the probability a point belongs to the specific test point's neighbourhood. Combining the above-mentioned notions, a unique k value is assigned to each unlabelled point.

As referred in (Bhattacharya et al., 2014), the proposed algorithm was compared in terms of accuracy, both to the conventional k-NN with fixed values k = 1, 3, 5, 7, $[\sqrt{N}]$ and to other k-NN variations like these presented in (Johansson, Boström, et al., 2008) and in (Tomašev et al., 2014). The comparison was made on 15 datasets. The results showed that the proposed algorithm mainly outperformed the competitors and thus can be considered as an effective k-NN variation.

In the research presented by Zhang & Li (2013) the authors suggest that the neighborhood size should be versatile in order to handle the imbalanced classification problem. Thus, for a given k, they propose that the examined neighborhood should be increased until $\frac{k}{2}$ instances of the rare class are included.

The above-mentioned approach was compared, in terms of auROC and Convex Hull analysis, to some others k-NN variations (ENN and CCW-k-NN) and techniques for handling imbalanced datasets (SMOTE re-sampling and MetaCost) on 12 datasets. Briefly, the experiments showed that the proposed approach demonstrated quite competitive performance as it outperformed both the k-NN variations and the widely used techniques for handling the imbalanced classification problem.

Song et al. (2007) argue that parameter k is not sufficient enough to decide the neighborhood's size. Thus, they introduce a new parameter, namely I, which is measuring the number of informative instances within the selected neighborhood. A training point is considered to be informative if it is close to the test instance and far enough from instances from other classes. According to the same authors, k and I selection is made using cross validation techniques.

The proposed approach was compared, in terms of error rate, to some k-NN variations and other popular classifiers, like Support Vector Machines (SVM). The comparison was made on a variety of datasets, including text categorization and object recognition. The results showed that the proposed algorithm is less sensitive to k selection (comparing to conventional k-NN algorithm) while it demonstrated competitive performance compared to widely used classifiers.

Another probabilistic approach is presented by Ghosh (2007). According to this approach, a unique k value is selected for each unlabeled instance based on the class distribution around it.

2.3.6 Other Approaches

An approach for dynamic k estimation is presented by Zhong et al (2017). Briefly, according to this approach, an iterative algorithm outputs an interval [kmin, kmax], where k is searched dynamically. Then, the algorithm creates variation tendency curves, according to the proportion of correctly classified instances for each k value within the aforementioned interval. The final k value is selected based on three criteria that are dependent on the shape of the variation tendency curves. The proposed algorithm is compared, in terms of precision, recall and F-score, to the conventional k-NN with fixed values k = 1, 5, 7, $\sqrt{N}$, where N =sample size, on a Facebook dataset. The approach in question outperformed the rest ones, in terms of recall and F-score, while it had the second best performance in terms of precision.

Another approach where k is selected for each data point is presented by Anava & Levy) (2016). Adopting a weighting scheme, the authors tried to minimize the distance between the generated prediction and the ground truth, optimizing at the same time the number of the selected nearest neighbors for each unique unlabeled instance. The proposed algorithm was compared, in terms of standard error metric, to the conventional k-NN and the Nadaraya-Watson estimator on 8 datasets. According to the results, the proposed approach outperformed the rest ones on seven out of eight datasets while on three out of seven datasets, the outperformance was statistically significant.

S. Zhang et al. (2014) present an analytic optimization framework, namely Graph Sparse k-NN (GS-k-NN), in order to learn a different (optimum) k value for each unlabeled (test) instance, utilizing the already known data distribution. The proposed approach was compared, in terms of accuracy and root mean square error (RMSE), to conventional k-NN (with fixed k = 5) and another k-NN variation (L-k-NN) on 12 datasets. The experiments conducted for classification, regression and missing values imputation. In summary, the GS-k-NN is considered by the authors a quite competitive approach as it outperformed the rest algorithms, demonstrating higher accuracy and lower RMSE.

2.4 Conclusions

Based on Table 2-1, the aforementioned approaches cover the time period from 1986 till 2020 (with median value 2009). An interesting observation is that the last three years (from 2018 till 2020) only two publications have been recorded.

The majority of the approaches uses probabilistic frameworks in order to dynamically select the proper k value. However, these approaches are mostly the older ones, with median year of publication 2007. As far as the most recently published works (within the last five years), the majority of them utilize the prototype and clustering technique (three out of six publications), following the "other" (two out of six publications) and neural networks (one out of six publications) technique.

As far as the level of analysis for k value selection, either the search space is divided in distinct subspaces (level of analysis: the specific region or prototype), each of which is assigned with a unique k value (and, consequently, each unlabelled instance is assigned with the k value of the subspace it belongs to) or each test instance is assigned with a unique k value (level of analysis: the test instance itself).

The median number of datasets, used for testing the aforementioned techniques, is twelve (12), something that is considered insufficient to reliably evaluate the performance. This study supports that one should use a wide variety of datasets, like datasets containing noise, imbalaced datasets, real life datasets, etc., in order to evaluate in a holistic manner each approach's performance.

In any case, this should be supplemented with the appropriate statistical tests in order to ascertain if performance superiority is significant. Unfortunately, only in twelve out of twenty-eight cases, at least one statistical test was conducted.

The last two columns of the summarizing table deal with the number of citations received by each paper in an attempt to identify the most influential papers. In order to achieve this, it is considered appropriate not to just report the total number of citations each paper has received so far, but, also to compute a normalized index, i.e., the average citations per year. All the needed information was retrieved from Google Scholar in June 2021. The median number of citations is 36.5 while the median number of the average citations per year is 4.035. Nine (9) out of twenty eight (28) papers have received more than one hundred citations, while six (6) out of twenty eight (28) papers have more than ten (10) citations per year on average).

The three most influential papers - both in terms of the total number of citations and the average citations per year - are (Guo et al., 2003) (621 / 34.5), (Wang et al., 2007) (322 / 23) and (Song et al., 2007) (270 / 19.28). Unfortunately, one should note that none of the above-mentioned papers conduct statistical tests in order to ascertain the performance superiority, especially, in combination with the insufficient number of datasets used for testing (6, 5 and 12).

The author of the study in question feels the need to distinguish (Garcia-Pedrajas et al., 2017) as a complete approach, which could be adopted in almost every k-NN variation. The proposed algorithm, a prototype-based variation, was tested using 80 datasets for regular and 65 for imbalanced problems in terms of three different metrics (Kohen's Kappa, G-mean and auROC). The algorithm's outperformance was significant, based on Wilcoxon tests that were conducted for all different situations (regular and imbalanced datasets) and metrics. In light of all the above, the specific approach have received seventy-one (71) citations within only four (4) years (17.75 citations per year).

All the above makes it clear that a wide variety of approaches have been proposed in the literature, in an effort to build an even more effective k-NN algorithm as it is now evident that a fixed k value is not efficient enough to lead to optimal performance. This is due to the special features that every sub-space of each dataset has, e.g., each subspace within a dataset does not have same density or noise. However, researchers who try to overcome one of the main k-NN disadvantages, namely, the dependency on the k selection, by creating variations of the standard algorithm, should have in mind not to alleviate its advantages, which were mentioned in the first section. In other words, one should strike a balance between achieving higher classification rate (due to a "better" k selection) and keeping the algorithm relatively simple.

Table 2-1: Summary table of proposed approaches

A/A	paper	year	approach	level of k selection	#datasets	statistical tests	#citations	average citations per year
1	(Kardan et al., 2013)	2013	Genetic Algorithm	Global	10	No	9	1.12
2	(Johansson, König, et al., 2008)	2008	Genetic Algorithm	Region	27	Yes	3	0.23

3	(Mullick et al., 2018)	2018	Neural Networks / Heuristic	Region	17	Yes	39	13
4	(Garcia-Pedrajas et al., 2017)	2017	Prototypes & Clustering	Prototype	145	Yes	71	17.75
5	(Bulut & Amasyali, 2017)	2017	Prototypes & Clustering	Prototype	36	Yes	26	6.5
6	(Guo et al., 2003)	2003	Prototypes & Clustering	Prototype	6	No	**621**	**34.5**
7	(Ougiaroglou et al., 2020)	2020	Prototypes & Clustering	Prototype	33	No	0	0
8	(Ferrer-Troyano et al., 2001)	2001	Heuristics	Global	25	Yes	5	0.25
9	(Nock et al., 2003)	2003	Heuristics	Global	29	Yes	35	1.94
10	(Sun & Huang, 2010)	2010	Heuristics	Test Instance	15	No	103	9.36
11	(Baoli et al., 2004)	2004	Heuristics	Test Instance	2	No	142	8.35
12	(Ougiaroglou et al., 2007)	2007	Heuristics	Test Instance	2	No	57	4.07
13	(Holmes & Adams, 2002)	2002	Probabilistic	Global	6	No	172	9.05
14	(Johansson, Boström, et al., 2008)	2008	Probabilistic	Test Instance	18	Yes	6	0.46
15	(Bhattacharya et al., 2014)	2014	Probabilistic	Test Instance	15	Yes	18	2.57
16	(Bhattacharya et al., 2015)	2015	Probabilistic	Test Instance	15	No	9	1.5
17	(X. Zhang & Li, 2013)	2013	Probabilistic	Test Instance	12	Yes	28	3.5
18	(Song et al., 2007)	2007	Probabilistic	Test Instance	12	No	270	19.28
19	(Dhurandhar & Dobra, 2013)	2013	Probabilistic	Test Instance	5	No	38	1.63
20	(Ghosh, 2007)	2007	Probabilistic	Region	14	Yes	31	2.21
21	(Ghosh, 2006)	2006	Probabilistic	Region	11	Yes	123	8.2
22	(Wang et al., 2007)	2007	Probabilistic	Region	5	No	322	23

23	(Hamerly & Speegle, 2012)	2012	Other	Global	3	No	9	1
24	(Enas & Choi, 1986)	1986	Other	Global	0	No	120	4.8
25	(Hand & Vinciotti, 2003)	2003	Other	Global	0	No	100	5.55
26	(Anava & Levy, 2016)	2016	Other	Test Instance	8	Yes	50	10
27	(X. Zhang & Li, 2013)	2014	Other	Test Instance	12	No	19	2.71
28	(Zhong et al., 2017)	2017	Other	Region	1	No	6	4

3 Methodology

3.1 Introduction

Based on the previous chapter, an obvious trend for improving k-NN classifier has been demonstrated. This trend stems from the finding that a unique and constant k value for the whole dataset is not adequate enough to take into consideration all the special features a dataset may have, such as noisy subspaces, dense and sparse neighborhoods and imbalanced classes.

In this direction, in the context of this chapter, an alternative version of k-NN classifier will be presented. The approach extends the modified k-NN classifier, namely Rhd-k-NN, presented by Ougiaroglou et al. (2020), by, firstly, introducing the concept of sphere of influence and, secondly, expanding the list of the applied heuristcs. The author of this study strongly supports heuristics' worth in the sense of preserving the simplicity of the original algorithm, a major k-NN's advantage.

Based on the above, in this chapter, the study of Ougiaroglou et al. (2020) will be briefly presented as well as the concept of sphere of influence and the applied heuristics. Finally, the chapter will end with the presentation of the conducted experiments results.

3.2 rhd-k-NN

rhd-k-NN is an alternative k-NN version, presented by Ougiaroglou et al. (2020), falling into the group of local, prototype and clustering, approaches of modified k-NN algorithms.

The first step of this approach is a procedure called Structure of Homogeneous Clusters (SHC). This process, a recursively application of k-Means clustering algorithm, is applied until all created clusters are homogenous. The total number of representatives represents the original dataset. This process aims at reducing the entropy, a measure of disorder, of the original dataset. This step is visually presented in Figure 3-1.

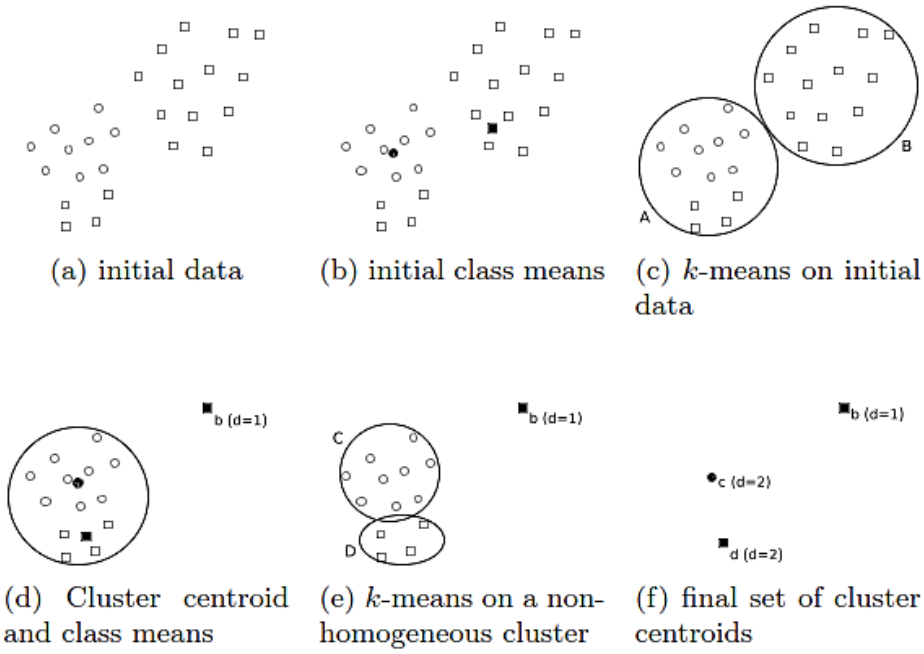
The second step is the determination of depth (d) for each representative. As depth is defined the number of repetitions needed to call k-Means algorithm in order to form a homogenous cluster. For example, if the k-Means algorithm is called three repeatedly times to create a homogenous cluster, the depth of cluster's representative equals to three (d = 3). Greater depth means greater entropy for the specific subspace. In turn, the greater the entropy, the greater k is needed to classify correctly.

The final step of this approach is the definition of k as a function of depth (d) for each representative. Specifically, for each test instance that needs to be classified, the one nearest representative is found as well as its depth. Then, the k value that will be used by k-NN – in order to classify the test instance in question – is computed, as a function of depth as previously mentioned, utilizing the following heuristics:

- $k = d$
- $k = 2^d$
- $k = d^2$
- $k = [d \times (d + 1) / 2]$
- $k = \lfloor e^{\sqrt{d}} \rfloor$

The experiments were conducted on a total of thirty-three datasets; the fourteen of them were standard benchmarking datasets while the rest nineteen were modified version of the original ones, with some noise addition (10% or 30%). The test was conducted in terms of accuracy against some common used k-NN variations; the ”best” k-NN (k was estimated using 5-fold cross validation), 1-NN, 5-NN, 10-NN, $\sqrt{N}$-NN and $\sqrt{\frac{N}{2}}$-NN, where N represents the number of instances included in training set.

The proposed algorithm demonstrated competitive performance, as it achieved higher accuracy than that of the best-k-NN in eighteen datasets while in two datasets the performance was similar. As far as the proposed heuristics are concerned, the $k = [d \times (d + 1) / 2]$ heuristic seemed to be the approach with the highest accuracy. The heuristic $k = d$ performed well in datasets without noise while the rest ones demonstrated similar performance with the best-k-NN.



(a) initial data (b) initial class means (c) k-means on initial data

(d) Cluster centroid and class means (e) k-means on a non-homogeneous cluster (f) final set of cluster centroids

Figure 3-1: The process of creating homogenous clusters through SHC [Source: (Ougiaroglou et al., 2020))

3.3 Proposed Algorithm and Heuristics

The proposed algorithm, in the context of this research, is based on the first two steps of rhd-k-NN. In other words, the algorithm initiates calling the procedure SHC in order to create homogenous clusters. This procedure runs once for every dataset, keeping a low computational cost (Ougiaroglou et al., 2020). Next, and after the completion of representative identification, follows the calculation of d value that corresponds to each representative. As one can see, these steps are identical with rhd-k-NN.

However, the point at which this research differentiates, is step three; specifically, this research extends the previous one, by introducing and studying a new concept, namely the Sphere of Influence (SoI). This index is straightly connected with the size of the created homogenous cluster, providing additional information for the original datasets. This index can be utilized either solely or supplementary to depth. For example, a homogenous cluster with large d value and small SoI value corresponds to dense, not well separated neighborhoods that maybe contain noise. That's why it took a large number of repetitions in order to create homogenous clusters (information deduced from index d) and the generated clusters are small (information deducted from index SoI). On the other hand, small d value and large SoI corresponds to well separated neighborhoods.

Specifically, the following two definitions of SoI were used during the experiments.

$$SoI = 1.25 \times Average\ Distance$$

$$SoI = \frac{Average\ Distance + Max\ Diastance}{2}$$

Where:

Average Distance: The average distance between the representative and the data points within the cluster.

Max Distance: The distance between the representative and the most distant point within the cluster.

The concept behind the second definition of SoI is to take into consideration the max distance within the dataset but counterbalancing it with the average distance. That's why an outlier could increase unpredictably the max distance, without providing useful information for k value calculation. On the contrary, such a great value of SoI, would result to large k values which in turn would increase the computational cost without increasing the accuracy, respectively.

The question that now arises is how to connect SoI with k value calculation. The answer is by exploiting heuristics, in order to keep the algorithm simple. All experiments, testing the proposed algorithm and the corresponding heuristics, were executed twice; the first execution used the first definition of SoI while the second execution used the second definition of SoI. In cases that SoI = 0 (that means that the whole cluster consists of only one data point – which is identical with the representative), the corresponding k value is calculated using the best performed heuristic in rhd-k-NN, namely k = [d x (d + 1) / 2].

In detail, the proposed heuristics are the following

1. k = [d x (d + 1) / 2], if SoI = 0
   k = SoI, in any other case
2. k = [d x (d + 1) / 2], if SoI = 0
   $k = SoI^2$, if $1 \leq SoI \leq 9$
   $k = 9^2$, in any other case
3. k = [d x (d + 1) / 2], if SoI = 0
   $k = 2^{SoI}$, if $1 \leq SoI \leq 9$
   $k = 2^9$, in any other case

4. k = [d x (d + 1) / 2], if SoI = 0

   k = [SoI x (SoI + 1) / 2], if $1 \leq SoI \leq 9$

   k = [9 x (9 + 1) / 2], in any other case

5. k = [d x (d + 1) / 2], if SoI = 0

   k = $\lfloor e^{\sqrt{SoI}} \rfloor$, if $1 \leq SoI \leq 9$

   k = $\lfloor e^{\sqrt{9}} \rfloor$, in any other case

6. k = [d x (d + 1) / 2], if SoI = 0

   k = SoI + d, in any other case

7. k = [d x (d + 1) / 2], if SoI = 0

   k = $\frac{SoI + d}{2}$, in any other case

8. k = [d x (d + 1) / 2], if SoI = 0

   k = d, if SoI = 1

   k = [d x (d + 1) / 8] x SoI, in any other case

9. k = [d x (d + 1) / 2], if SoI = 0

   k = d, if SoI = 1

   k = $\frac{SoI \text{ x } d}{3}$, in any other case

10. k = 1, if SoI = 1

    k = [d x (d + 1) / 2], in any other case

The attributes of the original datasets used for experiment conducting are presented in Table 3-1:

Table 3-1: Datasets attributes [Source: (Ougiaroglou et al., 2020)]

Dataset	**Size**	**Attributes**	**Classes**
Balance (bl)	625	4	3
Banana (bn)	5300	2	2
Ecoli (ecl)	336	7	8
Iris	120	4	3
Letter Recognition (lir)	20000	16	26
Landsat Satellite (ls)	6435	36	6
Magic G. Telescope (mgt)	19020	10	2
Pen-Digits (pd)	10992	16	10
Phoneme (ph)	5404	5	2
Pima (pm)	615	8	2
Shuttle (sh)	58000	9	7
Twonorm (tn)	7400	20	2
Texture (txr)	5500	40	11
Yeast (ys)	1484	8	10

Apart from the original datasets, experiments were conducted on some modified versions of them. These versions contain some noise in order to test the proposed algorithm in real – life situations. The selected datasets were those that Ougiaroglou et al. (2020) conducted their own experiments in order to have comparable results. The "noisy" versions are denoted using the suffix 10 or 30, containing 10% or 30% random uniform noise, respectively.

3.4 Experiments Results

Table 3-2 and Table 3-3 present the distribution of data points according to the total number of spheres that they belong. For each data set, the column containing the maximum number of points is highlighted with bold. One can see that the distribution is more or less similar, independently of the chosen definition of SoI, while the majority of data points belonging either to one or to none sphere. This is not the case for dataset Shuttle (sh). This fact is highly expected, given the special attributes of this dataset. Specifically, the dataset has nine classes while the 80% of the data points belong to the class 1. The rest classes (2 – 9) are mostly outliers. Given the fact that they are outliers, they increase the maximum distance within the created homogeneous clusters. This, in turn, increase each representative's SoI, respectively.

Table 3-2: Data points distribution in spheres if SoI = 1.25 x Average Distance

		0 spheres	**1 sphere**	**2 spheres**	**3 spheres**	**4 spheres**	**5 spheres**	**6+ spheres**
1	bl	30	**74**	21	0	0	0	0
2	bl10	**67**	55	3	0	0	0	0
3	bl30	**73**	49	3	0	0	0	0
4	bn	455	**577**	28	0	0	0	0
5	ecl	**31**	22	13	1	0	0	0
6	ecl10	28	**30**	7	2	0	0	0
7	ecl30	**51**	16	0	0	0	0	0
8	iris	**15**	**15**	0	0	0	0	0
9	lir	992	**2091**	721	147	38	6	5
10	ls	210	**409**	215	172	84	62	135
11	ls10	245	**337**	216	171	90	69	159
12	ls30	**398**	340	192	103	78	49	127
13	mgt	697	**1037**	917	589	316	158	90
14	mgt10	832	**1095**	908	518	268	114	69

15	pd	404	**1105**	503	157	22	7	0
16	pd10	754	**825**	431	138	43	6	1
17	pd30	**1062**	809	242	66	19	0	0
18	ph	328	**581**	148	21	2	0	0
19	ph10	425	**516**	122	16	1	0	0
21	pm	31	**50**	31	27	11	3	0
22	pm10	36	**44**	39	22	10	2	0
23	pm30	46	**49**	38	9	6	4	1
24	sh	2629	**7826**	1062	82	0	0	0
25	tn	106	66	59	54	62	58	**1075**
26	tn10	111	68	58	61	59	47	**1076**
27	tn30	135	84	56	65	65	60	**1015**
28	txr	226	**595**	200	62	10	6	1
29	txr10	379	**437**	193	73	9	5	4
30	txr30	**549**	387	129	27	7	1	0
31	ys	**131**	102	54	4	3	2	0

Table 3-3: Data points distribution in spheres if SoI = (Average + Max) / 2

	0 spheres	**1 sphere**	**2 spheres**	**3 spheres**	**4 spheres**	**5 spheres**	**6+ spheres**
bl	39	**72**	14	0	0	0	0
bl10	**87**	35	3	0	0	0	0
bl30	**95**	29	1	0	0	0	0
bn	290	**631**	138	1	0	0	0
ecl	27	**28**	8	3	1	0	0
ecl10	**33**	27	4	3	0	0	0
ecl30	**57**	9	1	0	0	0	0
iris	11	**17**	2	0	0	0	0
lir	961	**1994**	783	210	44	5	3
ls	204	**365**	227	186	94	77	134
ls10	314	**322**	240	167	108	58	78
ls30	**567**	382	156	80	53	24	25
mgt	522	**718**	711	635	487	334	397
mgt10	854	**1132**	901	462	274	116	65
pd	168	685	**775**	348	147	68	7
pd10	725	**816**	426	172	44	13	2
pd30	**1274**	723	172	26	3	0	0
ph	224	**429**	232	113	67	13	2
ph10	378	**462**	164	57	17	2	0
pm	42	**43**	30	16	15	5	2
pm10	49	**50**	30	21	3	0	0
pm30	**56**	52	32	11	2	0	0
sh	7	182	29	833	524	984	**9040**
tn	106	80	71	75	83	60	**1005**
tn10	211	135	107	77	74	56	**820**
tn30	305	151	118	84	73	55	**694**

txr	120	**602**	230	109	25	10	4
txr10	**441**	418	183	44	12	2	0
txr30	**726**	313	51	9	1	0	0
ys	**150**	102	37	5	2	0	0

Table 3-4 presents the experiment results for the ten tested heuristics, which were presented earlier in this chapter, when SoI is defined as 1.25 x Average Distance (for brevity, first case). Respectively, Table 3-5 present the experiment results when SoI is defined as $\frac{Average\ Distance+Max\ Distance}{2}$ (for brevity, second case). All results are expressed in terms of accuracy (%). In each table and for every dataset, the heuristic(s) demonstrating the highest performance are highlighted with bold. Moreover, Table 3-6 presents the percentage change between Table 3-4 and Table 3-5. Finally, Table 3-7 presents the accuracy results, obtained for the same datasets, using either some commonly used k-NN versions or the heuristics tested in (Ougiaroglou et al., 2020).

Beginning from Table 3-6, one can see that the results between the two approaches are quite similar. Specifically, there are percentage changes more than 5% only in two cases (datasets); in case of bl30 dataset – the "noisy" version of Balance dataset - for heuristcs 1-5 and heuristic 10 and in case of ys dataset for heuristic 1-5 and heuristic 7. The average performance – in terms of accuracy – is 86,94% in the first case and 87,27% in the second case. Based on this, one can support that the SoI definition that take into consideration both average and max distance, slightly outperforms the definition which take into consideration only the average distance.

Examining the overall performance of each of the proposing heuristics, it is clear that heuristic 6 outperforms the rest ones, independently from the definition of SoI. The average performance in the first case is 88,85%, being the sole heuristic achieving the highest accuracy in fourteen out of thirty cases. Quite similar is the second case, as well. Specifically, heuristic 6 achieved 88,91% accuracy, being the sole heuristic achieving the highest accuracy in twelve out of thirty cases. Compared to best k-NN (overall accuracy 89,88%) and the heuristic k = [d x (d + 1) / 2] (overall accuracy 89,24%), heuristic 6, even if it slightly underperforms, it demonstrates very competitive results. In fact, heuristic 6 outperforms in comparison with heuristic k = [d x (d + 1) / 2] in twelve datasets while there is also one tie. Moreover, heuristic 6 outperforms in comparison with best k-NN in four datasets, while there are also two ties.

In order to statistically compare the two pairs of algorithms, the Wilcoxon Statistical Test was performed, with significance level of 0.05. Wilcoxon test is recommended as a powerful and easily implemented procedure to compare two classifiers (Demšar & Janez, 2006). The same procedure is, also, followed by others researches in order to compare their proposed k-NN variations (Garcia-Pedrajas et al., 2017; Garcia et al., 2012; Mullick et al., 2018; Ougiaroglou & Evangelidis, 2016) The results are presented in Table 3-8 and Table 3-9. Specifically, the comparison between heuristic 6 and best k-NN, showed that the null hypothesis should be rejected, i.e. the mean accuracy between the 2 classifiers are not equal. On the other hand, the comparison between heuristic 6 and the heuristic k = [d x (d + 1) / 2], highlighted by (Ougiaroglou et al., 2020) as the most highly performed variation, showed that the null hypothesis failed to be rejected. This means that there is not sufficient evidence to say that the mean accuracy is not equal between the two groups.

Heuristic 9 (88,44% / 88,71%), heuristic 8 (88,36% / 88,71%) and heuristic 10 (86,85% / 87,06%) follow in the ranking. Wilcoxon tests were also performed to compare heuristic 6 against heuristics 8, 9 and 10, respectively. The test showed that the null hypothesis failed to be rejected in comparisons of heuristic 6 and heuristic 8 and 9. This was not the case in comparison of heuristic 6 with heuristic 10, where the null hypothesis was rejected. All results are presented in Table 3-10, Table 3-11 and Table 3-12.

Despite the fact that heuristic 10 proved to underperform, in comparison with heuristic 6, it seems to be a very interesting and simple approach. That's why it is considered as an improvement of the commonly used 1-NN classifier. Specifically, heuristic 10 utilize the 1-NN classifier when SoI = 1, i.e. in cases where the unlabeled test instance belongs only in one sphere of influence. In this case, the classification is conducted by finding the one nearest neighbor. In rest cases, the classification is made utilizing the heuristic k = [d x (d + 1) / 2]. The improvement in 1-NN classifier is obvious as it increases the accuracy from 79,54% to 87,06%. Generally speaking, 1-NN approach, even if it is a simple and easily implemented approach, proves to be insufficient to meet the demands and the complexity of real – life datasets.

Table 3-4: Experiments results for all heuristic with SoI = 1.25 x Average Distance

DATASET	Heuristc 1	Heuristc 2	Heuristc 3	Heuristc 4	Heuristc 5	Heuristc 6	Heuristc 7	Heuristc 8	Heuristc 9	Heuristc 10
bl	82,400%	83,200%	83,200%	82,400%	83,200%	**88,000%**	84,000%	86,400%	87,200%	84,000%
bl10	80,800%	80,800%	82,400%	80,800%	82,400%	**88,800%**	84,800%	88,000%	88,000%	81,600%
bl30	70,400%	70,400%	68,000%	70,400%	68,000%	**81,600%**	74,400%	79,200%	79,200%	72,000%
bn	89,906%	89,811%	89,906%	89,906%	89,906%	90,094%	**90,283%**	90,094%	90,094%	89,811%
ecl	**89,552%**	**89,552%**	88,060%	86,567%	88,060%	88,060%	88,060%	88,060%	88,060%	**89,552%**
ecl10	86,567%	86,567%	88,060%	85,075%	88,060%	**91,045%**	86,567%	89,552%	89,552%	86,567%
ecl30	**88,060%**	**88,060%**	86,567%	88,060%	86,567%	**88,060%**	86,567%	86,567%	86,567%	88,060%
iris	**96,667%**	**96,667%**	**96,667%**	**96,667%**	**96,667%**	**96,667%**	**96,667%**	**96,667%**	**96,667%**	**96,667%**
lir	94,725%	94,675%	94,175%	**94,850%**	94,250%	94,575%	94,275%	94,675%	94,600%	94,825%
ls	90,676%	**91,531%**	90,210%	91,375%	90,676%	90,987%	91,064%	90,987%	91,142%	91,298%
ls10	85,703%	87,024%	87,102%	87,102%	87,801%	**89,899%**	88,967%	89,433%	89,510%	87,413%
ls30	75,447%	81,197%	81,041%	79,176%	80,342%	**87,723%**	81,585%	87,413%	86,791%	82,517%
mgt	80,547%	81,178%	82,019%	81,493%	81,966%	**83,044%**	82,729%	82,624%	82,860%	80,862%
mgt10	78,049%	79,364%	80,047%	78,601%	79,784%	**82,466%**	81,677%	82,387%	82,413%	79,495%
pd	98,908%	98,908%	98,908%	**98,954%**	98,908%	98,817%	**98,954%**	98,772%	98,908%	98,772%

pd10	94,177%	95,541%	95,632%	95,223%	95,632%	**98,590%**	97,179%	98,499%	98,499%	95,587%
pd30	86,852%	89,263%	88,444%	88,808%	88,353%	**97,361%**	90,901%	96,906%	96,497%	89,854%
ph	84,537%	84,722%	84,074%	**85,370%**	84,074%	82,500%	83,426%	82,870%	82,870%	84,815%
ph10	81,111%	81,296%	82,315%	81,296%	**82,500%**	**82,500%**	82,315%	81,759%	81,944%	81,019%
pm	73,203%	72,549%	73,203%	73,856%	73,856%	**76,471%**	74,510%	73,203%	75,163%	73,203%
pm10	75,817%	75,817%	77,124%	73,856%	76,471%	**77,778%**	75,163%	72,549%	75,817%	74,510%
pm30	66,667%	66,013%	64,706%	64,706%	65,359%	71,895%	67,320%	**73,203%**	72,549%	69,935%
sh	**99,879%**	99,871%	99,845%	99,871%	99,845%	99,810%	99,836%	99,819%	99,819%	99,862%
tn	97,770%	97,905%	98,041%	97,905%	97,905%	97,905%	97,973%	**98,108%**	97,905%	98,041%
tn10	95,676%	96,351%	96,959%	96,554%	96,622%	97,568%	97,432%	97,635%	**97,770%**	96,757%
tn30	89,797%	94,459%	94,392%	93,446%	90,608%	95,676%	91,892%	**96,959%**	96,014%	95,946%
txr	**98,182%**	**98,182%**	**98,182%**	**98,182%**	**98,182%**	**98,182%**	**98,182%**	**98,182%**	**98,182%**	**98,182%**
txr10	93,455%	94,636%	94,455%	94,636%	94,455%	97,182%	94,727%	**97,273%**	97,182%	94,909%
txr30	85,636%	88,364%	87,636%	87,727%	87,545%	**95,545%**	87,455%	93,818%	93,545%	89,182%
ys	55,743%	56,419%	53,716%	56,757%	53,716%	56,757%	54,054%	59,122%	57,770%	**60,135%**
Average	85,564%	86,344%	86,169%	85,987%	86,057%	**88,852%**	86,765%	88,358%	88,436%	86,846%

Table 3-5: Experiments results for all heuristic with SoI = (Average Distance + Max Distance) / 2

DATASET	Heuristc 1	Heuristc 2	Heuristc 3	Heuristc 4	Heuristc 5	Heuristc 6	Heuristc 7	Heuristc 8	Heuristc 9	Heuristc 10
bl	81,600%	81,600%	82,400%	81,600%	82,400%	86,400%	83,200%	**88,000%**	87,200%	84,000%
bl10	84,000%	84,000%	84,800%	84,000%	84,800%	**88,800%**	86,400%	88,000%	88,000%	84,800%
bl30	76,000%	76,000%	74,400%	76,000%	74,400%	**80,800%**	77,600%	80,000%	80,000%	76,800%
bn	90,000%	89,811%	89,717%	89,906%	89,717%	**90,472%**	90,094%	90,000%	90,094%	89,811%
ecl	89,552%	89,552%	89,552%	88,060%	89,552%	88,060%	**91,045%**	88,060%	88,060%	89,552%
ecl10	88,060%	88,060%	88,060%	86,567%	88,060%	**91,045%**	88,060%	88,060%	88,060%	88,060%
ecl30	86,567%	86,567%	88,060%	86,567%	88,060%	**89,552%**	88,060%	88,060%	88,060%	86,567%
iris	**96,667%**	**96,667%**	**96,667%**	**96,667%**	**96,667%**	**96,667%**	**96,667%**	**96,667%**	**96,667%**	**96,667%**
lir	94,575%	94,600%	94,000%	**94,700%**	94,150%	94,575%	94,175%	94,675%	94,525%	**94,700%**
ls	90,443%	91,220%	90,054%	91,064%	90,443%	90,831%	91,220%	90,831%	90,987%	**91,298%**
ls10	86,558%	88,112%	87,801%	88,112%	88,034%	**90,132%**	88,811%	89,588%	**90,132%**	88,034%
ls30	75,758%	79,720%	80,264%	78,089%	79,254%	**87,646%**	81,818%	87,257%	86,558%	80,886%
mgt	81,204%	81,940%	82,150%	81,914%	82,229%	**82,992%**	82,886%	82,597%	82,676%	81,335%
mgt10	77,813%	78,996%	79,679%	78,891%	79,416%	82,597%	81,598%	**82,624%**	82,571%	79,232%
pd	**98,954%**	98,726%	98,681%	**98,954%**	98,772%	98,817%	98,863%	98,726%	98,863%	98,726%

pd10	94,677%	95,496%	95,450%	95,314%	95,450%	**98,681%**	97,407%	98,635%	98,681%	95,587%
pd30	89,399%	90,855%	90,309%	90,446%	90,309%	**97,680%**	92,448%	97,043%	96,906%	91,083%
ph	84,630%	84,722%	84,167%	**85,093%**	84,352%	82,500%	83,704%	82,778%	82,963%	84,352%
ph10	80,926%	80,926%	82,222%	80,926%	82,222%	**83,148%**	82,778%	82,685%	82,500%	80,648%
pm	71,895%	72,549%	72,549%	73,856%	72,549%	**75,817%**	73,203%	74,510%	75,163%	72,549%
pm10	75,163%	74,510%	74,510%	73,203%	74,510%	**76,471%**	74,510%	75,163%	**76,471%**	74,510%
pm30	65,359%	66,667%	66,013%	64,052%	64,706%	72,549%	67,320%	**74,510%**	73,856%	68,627%
sh	99,862%	99,595%	99,396%	99,733%	99,784%	99,784%	**99,862%**	99,707%	99,776%	99,802%
tn	97,838%	97,770%	97,973%	97,838%	97,973%	97,635%	**98,108%**	97,770%	97,770%	97,770%
tn10	94,865%	96,284%	96,216%	96,284%	95,946%	**97,770%**	97,297%	97,635%	**97,770%**	96,622%
tn30	87,568%	92,162%	91,892%	91,351%	87,838%	94,730%	89,527%	**96,892%**	95,608%	95,270%
txr	98,273%	98,273%	98,364%	**98,636%**	98,364%	98,000%	98,364%	98,273%	98,273%	98,091%
txr10	93,636%	95,364%	94,909%	95,273%	94,909%	97,455%	95,091%	**97,545%**	97,455%	95,455%
txr30	88,818%	89,818%	89,091%	89,545%	89,091%	**96,364%**	89,545%	95,273%	95,273%	89,909%
ys	59,122%	59,459%	58,108%	59,797%	57,770%	59,459%	58,446%	59,797%	60,473%	**61,149%**
Average	85,993%	86,667%	86,582%	86,415%	86,391%	**88,914%**	87,270%	88,712%	88,713%	87,063%

Table 3-6: Accuracy percentage change between the two definitions for SoI

DATASET	Heuristc 1	Heuristc 2	Heuristc 3	Heuristc 4	Heuristc 5	Heuristc 6	Heuristc 7	Heuristc 8	Heuristc 9	Heuristc 10
bl	-0,97%	-1,92%	-0,96%	-0,97%	-0,96%	-1,82%	-0,95%	1,85%	0,00%	0,00%
bl10	3,96%	3,96%	2,91%	3,96%	2,91%	0,00%	1,89%	0,00%	0,00%	3,92%
bl30	**7,95%**	**7,95%**	**9,41%**	**7,95%**	**9,41%**	-0,98%	4,30%	1,01%	1,01%	**6,67%**
bn	0,10%	0,00%	-0,21%	0,00%	-0,21%	0,42%	-0,21%	-0,10%	0,00%	0,00%
ecl	0,00%	0,00%	1,69%	1,72%	1,69%	0,00%	3,39%	0,00%	0,00%	0,00%
ecl10	1,72%	1,72%	0,00%	1,75%	0,00%	0,00%	1,72%	-1,67%	-1,67%	1,72%
ecl30	-1,70%	-1,70%	1,72%	-1,70%	1,72%	1,69%	1,72%	1,72%	1,72%	-1,70%
iris	0,00%	0,00%	0,00%	0,00%	0,00%	0,00%	0,00%	0,00%	0,00%	0,00%
lir	-0,16%	-0,08%	-0,19%	-0,16%	-0,11%	0,00%	-0,11%	0,00%	-0,08%	-0,13%
ls	-0,26%	-0,34%	-0,17%	-0,34%	-0,26%	-0,17%	0,17%	-0,17%	-0,17%	0,00%
ls10	1,00%	1,25%	0,80%	1,16%	0,27%	0,26%	-0,18%	0,17%	0,69%	0,71%
ls30	0,41%	-1,82%	-0,96%	-1,37%	-1,35%	-0,09%	0,29%	-0,18%	-0,27%	-1,98%
mgt	0,82%	0,94%	0,16%	0,52%	0,32%	-0,06%	0,19%	-0,03%	-0,22%	0,58%
mgt10	-0,30%	-0,46%	-0,46%	0,37%	-0,46%	0,16%	-0,10%	0,29%	0,19%	-0,33%
pd	0,05%	-0,18%	-0,23%	0,00%	-0,14%	0,00%	-0,09%	-0,05%	-0,05%	-0,05%

pd10	0,53%	-0,05%	-0,19%	0,10%	-0,19%	0,09%	0,23%	0,14%	0,18%	0,00%
pd30	2,93%	1,78%	2,11%	1,84%	2,21%	0,33%	1,70%	0,14%	0,42%	1,37%
ph	0,11%	0,00%	0,11%	-0,32%	0,33%	0,00%	0,33%	-0,11%	0,11%	-0,55%
ph10	-0,23%	-0,46%	-0,11%	-0,46%	-0,34%	0,79%	0,56%	1,13%	0,68%	-0,46%
pm	-1,79%	0,00%	-0,89%	0,00%	-1,77%	-0,86%	-1,75%	1,79%	0,00%	-0,89%
pm10	-0,86%	-1,72%	-3,39%	-0,88%	-2,56%	-1,68%	-0,87%	3,60%	0,86%	0,00%
pm30	-1,96%	0,99%	2,02%	-1,01%	-1,00%	0,91%	0,00%	1,79%	1,80%	-1,87%
sh	-0,02%	-0,28%	-0,45%	-0,14%	-0,06%	-0,03%	0,03%	-0,11%	-0,04%	-0,06%
tn	0,07%	-0,14%	-0,07%	-0,07%	0,07%	-0,28%	0,14%	-0,34%	-0,14%	-0,28%
tn10	-0,85%	-0,07%	-0,77%	-0,28%	-0,70%	0,21%	-0,14%	0,00%	0,00%	-0,14%
tn30	-2,48%	-2,43%	-2,65%	-2,24%	-3,06%	-0,99%	-2,57%	-0,07%	-0,42%	-0,70%
txr	0,09%	0,09%	0,19%	0,46%	0,19%	-0,19%	0,19%	0,09%	0,09%	-0,09%
txr10	0,19%	0,77%	0,48%	0,67%	0,48%	0,28%	0,38%	0,28%	0,28%	0,58%
txr30	3,72%	1,65%	1,66%	2,07%	1,77%	0,86%	2,39%	1,55%	1,85%	0,82%
ys	**6,06%**	**5,39%**	**8,18%**	**5,36%**	**7,55%**	4,76%	**8,13%**	1,14%	4,68%	1,69%
Average	0,61%	0,50%	0,66%	0,60%	0,53%	0,12%	0,69%	0,46%	0,38%	0,29%

Table 3-7: Experiments results for commonly used k-NN versions and heuristic tested in (Ougiaroglou et al., 2020) [Source: (Ougiaroglou et al., 2020)]

DATASET	Best k-NN	1-NN	5-NN	10-NN	RoT $k = \sqrt{N}$	RoT $k = \sqrt{\frac{N}{2}}$	k = d	$k = 2^d$	$k = d^2$	k = [d× (d + 1)]/2	$k = [e^{\sqrt{d}}]$
bl	89,60%	79,20%	86,40%	89,60%	89,60%	89,60%	84,80%	88,80%	89,60%	**90,40%**	89,60%
bl10	88,80%	68,80%	86,40%	87,20%	89,60%	89,60%	88,00%	**91,20%**	89,60%	90,40%	87,20%
bl30	**88,00%**	59,20%	69,60%	79,20%	83,20%	83,20%	73,60%	83,20%	87,20%	82,40%	76,80%
bn	**90,66%**	87,26%	89,81%	90,19%	90,57%	90,47%	89,81%	88,68%	90,38%	90,28%	90,47%
ecl	91,05%	83,58%	89,55%	**92,54%**	86,57%	**92,54%**	88,06%	88,06%	86,57%	86,57%	89,55%
ecl10	89,55%	77,61%	88,06%	**92,54%**	88,06%	89,55%	86,57%	91,05%	**92,54%**	91,05%	91,05%
ecl30	85,08%	61,19%	82,09%	85,08%	85,08%	85,08%	76,12%	85,08%	**88,06%**	**88,06%**	85,08%
iris	93,33%	90,00%	93,33%	93,33%	93,33%	93,33%	93,33%	93,33%	93,33%	**96,67%**	93,33%
lir	**95,78%**	95,70%	95,40%	95,03%	81,05%	84,20%	95,65%	95,28%	93,83%	95,05%	95,43%
ls	**91,53%**	89,98%	**91,53%**	91,14%	86,09%	87,34%	91,22%	90,68%	89,59%	90,99%	90,83%
ls10	**91,30%**	81,66%	90,52%	90,99%	86,09%	87,10%	90,75%	90,37%	89,90%	89,98%	90,75%
ls30	88,27%	61,23%	82,21%	89,04%	86,25%	86,79%	82,98%	88,58%	**89,36%**	88,81%	87,72%
mgt	**83,57%**	80,13%	82,97%	**83,57%**	80,97%	81,65%	83,36%	78,97%	80,86%	81,86%	83,39%
mgt10	**83,10%**	73,32%	80,97%	82,76%	81,13%	81,55%	82,89%	78,68%	80,81%	81,84%	82,99%

pd	99,05%	99,05%	**99,09%**	98,73%	95,04%	96,13%	**99,09%**	98,59%	98,32%	98,54%	98,91%
pd10	98,86%	89,40%	**98,95%**	98,73%	95,13%	96,36%	98,91%	98,73%	98,36%	98,64%	98,91%
pd30	**98,68%**	69,75%	94,18%	98,45%	94,90%	96,36%	92,58%	98,04%	98,23%	98,41%	97,59%
ph	**88,70%**	**88,70%**	86,94%	86,30%	82,04%	83,52%	85,74%	78,98%	81,30%	81,94%	84,44%
ph10	**86,67%**	81,39%	86,02%	86,02%	81,76%	83,89%	84,35%	78,80%	81,11%	81,48%	84,26%
pm	76,47%	77,12%	78,43%	81,05%	75,16%	77,78%	80,39%	72,55%	74,51%	73,86%	**82,35%**
pm10	78,43%	72,55%	76,47%	78,43%	75,16%	78,43%	75,82%	77,12%	77,78%	**79,09%**	77,78%
pm30	73,86%	64,05%	64,71%	69,94%	71,90%	71,24%	71,24%	73,86%	71,90%	73,20%	**76,47%**
sh	**99,96%**	**99,96%**	99,91%	99,86%	99,44%	99,36%	99,89%	99,72%	99,67%	99,83%	99,85%
tn	97,97%	95,54%	97,70%	97,77%	97,77%	98,04%	97,77%	**98,11%**	97,70%	97,91%	98,04%
tn10	97,77%	84,39%	95,47%	97,10%	97,57%	97,50%	97,10%	**97,91%**	97,43%	97,70%	97,70%
tn30	**97,77%**	67,23%	79,19%	86,08%	97,43%	97,57%	86,69%	**97,77%**	97,03%	97,57%	94,73%
txr	**98,73%**	**98,73%**	98,09%	97,82%	94,73%	95,82%	98,36%	98,09%	97,55%	98,00%	98,09%
txr10	97,91%	90,55%	97,91%	97,82%	94,55%	95,55%	**98,09%**	97,73%	97,46%	97,64%	97,73%
txr30	**97,55%**	69,73%	93,55%	**97,55%**	94,82%	94,91%	91,09%	96,55%	97,00%	**97,55%**	96,64%
ys	58,45%	49,32%	57,10%	59,12%	61,15%	60,14%	55,41%	61,15%	60,14%	**61,49%**	57,43%
Average	**89,88%**	79,54%	87,09%	89,10%	87,20%	88,15%	87,32%	88,52%	88,90%	89,24%	89,17%

Table 3-8: Wilcoxon test between heuristic 6 and best k-NN

Wilcoxon signed-rank test / Two-tailed test:	
V	297
V (standardized)	2,583
Expected value	189
Variance (V)	1732,500
p-value (Two-tailed)	0,010
alpha	0,05

Test interpretation:
H0: The two samples follow the same distribution.
Ha: The distributions of the two samples are different.
As the computed p-value is lower than the significance level alpha=0,05, one should reject the null hypothesis H0, and accept the alternative hypothesis, Ha.

Table 3-9: Wilcoxon test between heuristic 6 and heuristic k = [dx(d+1)/2]

Wilcoxon signed-rank test / Two-tailed test:	
V	236
V (standardized)	0,740
Expected value	203
Variance (V)	1928,375
p-value (Two-tailed)	0,459
alpha	0,05

Test interpretation:
H0: The two samples follow the same distribution.
Ha: The distributions of the two samples are different.
As the computed p-value is greater than the significance level alpha=0,05, one cannot reject the null hypothesis H0.

Table 3-10: Wilcoxon test between heuristic 6 and heuristic 8

Wilcoxon signed-rank test / Two-tailed test:	
V	307
V (standardized)	1,925
Expected value	217,500
Variance (V)	2138,625
p-value (Two-tailed)	0,054
alpha	0,05

Test interpretation:
H0: The two samples follow the same distribution.
Ha: The distributions of the two samples are different.

As the computed p-value is greater than the significance level alpha=0,05, one cannot reject the null hypothesis H0.

Table 3-11: Wilcoxon test between heuristic 6 and heuristic 9

Wilcoxon signed-rank test / Two-tailed test:	
V	292
V (standardized)	1,600
Expected value	217,500
Variance (V)	2138,625
p-value (Two-tailed)	0,110
alpha	0,05

Test interpretation:
H0: The two samples follow the same distribution.
Ha: The distributions of the two samples are different.
As the computed p-value is greater than the significance level alpha=0,05, one cannot reject the null hypothesis H0.

Table 3-12: Wilcoxon test between heuristic 6 and heuristic 10

Wilcoxon signed-rank test / Two-tailed test:	
V	365
V (standardized)	3,179
Expected value	217,500
Variance (V)	2138,625
p-value (Two-tailed)	0,001
alpha	0,05

Test interpretation:
H0: The two samples follow the same distribution.
Ha: The distributions of the two samples are different.
As the computed p-value is lower than the significance level alpha=0,05, one should reject the null hypothesis H0, and accept the alternative hypothesis, Ha.

4 Conclusion

4.1 Summary and conclusions

As mentioned in chapter 2, this study aims at two goals. On the one hand, it attempts to record both the trend and the progress accomplished so far about k-NN variations – focusing on dynamic k value determination. This way, researchers will have the opportunity to form a clear picture of the achievements made to date, probably inspiring their future efforts. On the other hand, it attempts to develop a k-NN alternative, which will dynamically determine the k value, i.e., the number of neighbors that take part in the majority vote process, based on which the classification is made, without any user intervention.

In the above-mentioned direction, a thorough literature review was conducted. The research resulted in a pool of 28 publications, covering a time period between 1986 and 2020, with median value 2009. These studies, presented in the present work, revealed a clear tendency to improve the conventional k-NN algorithm in order to overcome one of its most important disadvantages; the performance's dependency on k value selection. This tendency comes in contrast to the prevailing (in practice) technique of choosing a fixed "best" k using cross-validation. The studies were divided into six groups, according to the approach they follow for k determination, namely genetic algorithms, neural networks, prototypes and clustering, heuristic based, probabilistic and the group other. The majority of the older publications exploited probabilistic approaches while the tendency changed over the years, with prototypes and clustering becoming the prevailing approach. Apart from this, the level of k selection for each study was recorded, namely global, region, prototype and test instance, the number of datasets used for experiments, whether statistical tests were conducted or not, the total number of citations each research has received as well as the average citations per year. The last two indexes helped to distinguish the three most influential studies out of the twenty-eight. In summary, in most cases, and after conducting experiments on standard benchmarking datasets, the developed variations outperformed the conventional k-NN algorithm, with a fixed k value throughout the dataset.

In addition to the literature review, a k-NN variation is proposed in the present study. Specifically, it is a k-free k-NN classifier, in the sense that the user does not select the parameter, but it is selected depending on the area where each unlabeled data point lies.

The algorithm initiates with a pre-processing step, the SHC algorithm. This algorithm takes as input the whole dataset and divides it into homogeneous clusters. Each homogeneous cluster is represented by a unique representative. Moreover, the depth for each representative is recorded, i.e. the number of recursions needed to call the k-Means clustering procedure in order to create the homogenous cluster. This study introduces a new term, namely Sphere of Influence (SoI). This number represents the size of the created homogenous cluster. Combined with depth, this index provides useful information about the subspace where an unlabeled test instance lies. Several proposed heuristics exploit this information in order to dynamically determine the appropriate k value for each test instance that needs to be classified.

All the proposed heuristics were tested on several datasets. Some of these datasets contained artificial noise in order to test the classifier in real life conditions. The experimental results revealed a very competitive performance. In fact, the proposed algorithm achieved higher accuracy in four datasets against the conventional k-NN classifier that selects the k value with a cross-validation procedure while there were, also, two ties. Moreover, the proposed algorithm achieved higher accuracy in comparison with heuristic k = [d x (d + 1) / 2], that was highlighted by Ougiaroglou et al (2020) as the most highly performed heuristic.

The above-mentioned results were, also, tested using the Wilcoxon statistical test. Besides, the literature review proved that statical tests are a necessary tool to compare two classifiers, in addition to experiment conducting. The test in question is a commonly used statistical procedure to compare two classifiers. Based on this, the difference between “best k” k-NN and the proposed algorithm is statistically significant. On the other hand, this was not the case in the comparison of the proposed algorithm and the heuristic [d × (d + 1)] / 2.

4.2 Future extensions

As far as the future extensions of this study are concerned, the efforts will be concentrated in developing new metrics that will take into consideration even more features about the subspace that each representative lies. For example, such metrics could be the number of instances (of the initial dataset) that each representative represents, the standard deviation of distances within each homogenous cluster or the number of classes that exist in the spere of influence of each representative. This way, new heuristics will be

created. These heuristics will consolidate all the above-mentioned information and will result in a dynamic k determination procedure, without any user interference. This classifier, an even more improved version of dynamic k determination k-NN, aspires to outperform the conventional k-NN.

5 References

Alballa, N., & Al-Turaiki, I. (2021). Machine learning approaches in COVID-19 diagnosis, mortality, and severity risk prediction: A review. *Informatics in Medicine Unlocked*, *24*, 100564. https://doi.org/10.1016/j.imu.2021.100564

Anava, O., & Levy, K. Y. (2016). k∗-Nearest neighbors: From global to local. *Advances in Neural Information Processing Systems*, 4923–4931.

Baoli, L., Qin, L., & Shiwen, Y. (2004). An adaptive k -nearest neighbor text categorization strategy. *ACM Transactions on Asian Language Information Processing*, *3*(4), 215–226. https://doi.org/10.1145/1039621.1039623

Bhattacharya, G., Ghosh, K., & Chowdhury, A. S. (2015). A probabilistic framework for dynamic k estimation in kNN classifiers with certainty factor. *2015 Eighth International Conference on Advances in Pattern Recognition (ICAPR)*, 1–5. https://doi.org/10.1109/ICAPR.2015.7050683

Bhattacharya, G., Ghosh, K., & Chowdhury, A. S. (2014). Test Point Specific k Estimation for kNN Classifier. *2014 22nd International Conference on Pattern Recognition*, 1478–1483. https://doi.org/10.1109/ICPR.2014.263

Bulut, F., & Amasyali, M. F. (2017). Locally adaptive k parameter selection for nearest neighbor classifier: one nearest cluster. *Pattern Analysis and Applications*, *20*(2), 415–425. https://doi.org/10.1007/s10044-015-0504-0

Cover, T., & Hart, P. (1967). Nearest neighbor pattern classification. *IEEE Transactions on Information Theory*, *13*(1), 21–27. https://doi.org/10.1109/TIT.1967.1053964

Demšar, & Janez. (2006). Statistical Comparisons of Classifiers over Multiple Data Sets. *The Journal of Machine Learning Research*, *7*, 1–30. http://delivery.acm.org/10.1145/1250000/1248548/7-1-demsar.pdf?ip=143.107.252.18&id=1248548&acc=PUBLIC&key=344E943C9DC262BB.0DBCED839AA5AFE8.4D4702B0C3E38B35.4D4702B0C3E38B35&CFID=453198389&CFTOKEN=69587539&__acm__=1415480611_4fa88c3d7309c3e5b407ff62d2808

Dhurandhar, A., & Dobra, A. (2013). Probabilistic characterization of nearest neighbor classifier. *International Journal of Machine Learning and Cybernetics*, *4*(4), 259–272. https://doi.org/10.1007/s13042-012-0091-y

Enas, G. G., & Choi, S. C. (1986). Choice Of The Smoothing Parameter And Efficiency

Of K-Nearest Neighbor Classification. In *Statistical Methods of Discrimination and Classification* (Vol. 12, Issue 2, pp. 235–244). Elsevier. https://doi.org/10.1016/B978-0-08-034000-5.50011-3

Ferrer-Troyano, F. J., Aguilar-Ruiz, J. S., & Riquelme, J. C. (2001). Non-parametric Nearest Neighbor with Local Adaptation. In P. Brazdil & A. Jorge (Eds.), *Progress in Artificial Intelligence. EPIA 2001. Lecture Notes in Computer Science* (Vol. 2258, Issue June 2014, pp. 22–29). Springer, Berlin, Heidelberg. https://doi.org/10.1007/3-540-45329-6_6

Garcia-Pedrajas, N., Romero del Castillo, J. A., & Cerruela-Garcia, G. (2017). A Proposal for Local k Values for k -Nearest Neighbor Rule. *IEEE Transactions on Neural Networks and Learning Systems*, *28*(2), 470–475. https://doi.org/10.1109/TNNLS.2015.2506821

Garcia, S., Derrac, J., Cano, J. R., & Herrera, F. (2012). Prototype Selection for Nearest Neighbor Classification: Taxonomy and Empirical Study. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *34*(3), 417–435. https://doi.org/10.1109/TPAMI.2011.142

Ghosh, A. K. (2006). On optimum choice of k in nearest neighbor classification. *Computational Statistics & Data Analysis*, *50*(11), 3113–3123. https://doi.org/10.1016/j.csda.2005.06.007

Ghosh, A. K. (2007). On Nearest Neighbor Classification Using Adaptive Choice of k. *Journal of Computational and Graphical Statistics*, *16*(2), 482–502. https://doi.org/10.1198/106186007X208380

Guo, G., Wang, H., Bell, D., Bi, Y., & Greer, K. (2003). KNN Model-Based Approach in Classification. In R. Meersman, Z. Tar, & D. C. Schmidt (Eds.), *On The Move to Meaningful Internet Systems 2003: CoopIS, DOA, and ODBASE* (Vol. 2888, Issue January, pp. 986–996). Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-540-39964-3_62

Hamerly, G., & Speegle, G. (2012). Efficient Model Selection for Large-Scale Nearest-Neighbor Data Mining. In L. . MacKinnon (Ed.), *Data Security and Security Data. BNCOD 2010. Lecture Notes in Computer Science* (Vol. 6121, Issue January 2010, pp. 37–54). https://doi.org/10.1007/978-3-642-25704-9_6

Hand, D. J., & Vinciotti, V. (2003). Choosing k for two-class nearest neighbour classifiers with unbalanced classes. *Pattern Recognition Letters*, *24*(9–10), 1555–1562.

https://doi.org/10.1016/S0167-8655(02)00394-X

Holmes, C. C., & Adams, N. M. (2002). A probabilistic nearest neighbour method for statistical pattern recognition. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, *64*(2), 295–306. https://doi.org/10.1111/1467-9868.00338

Johansson, U., Boström, H., & König, R. (2008). Extending nearest neighbor classification with spheres of confidence. *Proceedings of the 21th International Florida Artificial Intelligence Research Society Conference, FLAIRS-21, February 2014*, 282–287.

Johansson, U., König, R., & Niklasson, L. (2008). Evolving a locally optimized instance based learner. *Proceedings of the 2008 International Conference on Data Mining, DMIN 2008*, 124–129.

Kardan, A. A., Kavian, A., & Esmaeili, A. (2013). Simultaneous feature selection and feature weighting with K selection for KNN classification using BBO algorithm. *The 5th Conference on Information and Knowledge Technology*, 349–354. https://doi.org/10.1109/IKT.2013.6620092

Kifokeris, D., & Xenidis, Y. (2019). Risk source-based constructability appraisal using supervised machine learning. *Automation in Construction*, *104*(May), 341–359. https://doi.org/10.1016/j.autcon.2019.04.012

Mullick, S. S., Datta, S., & Das, S. (2018). Adaptive Learning-Based k-Nearest Neighbor Classifiers With Resilience to Class Imbalance. *IEEE Transactions on Neural Networks and Learning Systems*, *29*(11), 1–13. https://doi.org/10.1109/TNNLS.2018.2812279

Nock, R., Sebban, M., & Bernard, D. (2003). A Simple Locally Adaptive Nearest Neighbor Rule With Application To Pollution Forecasting. *International Journal of Pattern Recognition and Artificial Intelligence*, *17*(08), 1369–1382. https://doi.org/10.1142/S0218001403002952

Ougiaroglou, S., & Evangelidis, G. (2016). RHC: a non-parametric cluster-based data reduction for efficient $$k$$ k -NN classification. *Pattern Analysis and Applications*, *19*(1), 93–109. https://doi.org/10.1007/s10044-014-0393-7

Ougiaroglou, S., Evangelidis, G., & Diamantaras, K. I. (2020). Dynamic k-NN Classification Based on Region Homogeneity. In J. Darmont, B. Novikov, & R. Wrembel (Eds.), *Communications in Computer and Information Science: Vol. 1259 CCIS* (pp. 27–37). Springer, Cham. https://doi.org/10.1007/978-3-030-54623-6_3

Ougiaroglou, S., Nanopoulos, A., Papadopoulos, A. N., Manolopoulos, Y., & Welzer-

Druzovec, T. (2007). Adaptive k-Nearest-Neighbor Classification Using a Dynamic Number of Nearest Neighbors. In *Advances in Databases and Information Systems* (pp. 66–82). Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-540-75185-4_7

Papanikolaou, M., Evangelidis, G., & Ougiaroglou, S. (2021). Dynamic k determination in k-NN classifier: A literature review. *IISA 2021 - 12th International Conference on Information, Intelligence, Systems and Applications*. https://doi.org/10.1109/IISA52424.2021.9555525

Rohaan, D., Topan, E., & Groothuis-Oudshoorn, C. G. M. (2022). Using supervised machine learning for B2B sales forecasting: A case study of spare parts sales forecasting at an after-sales service provider. *Expert Systems with Applications*, *188*(April 2021), 115925. https://doi.org/10.1016/j.eswa.2021.115925

Sakhare, N. N., & Sagar Imambi, S. (2019). Performance analysis of regression based machine learning techniques for prediction of stock market movement. *International Journal of Recent Technology and Engineering*, *7*(6), 655–662.

Song, Y., Huang, J., Zhou, D., Zha, H., & Giles, C. L. (2007). IKNN: Informative K-Nearest Neighbor Pattern Classification. In *Knowledge Discovery in Databases: PKDD 2007* (pp. 248–264). Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-540-74976-9_25

Sun, S., & Huang, R. (2010). An adaptive k-nearest neighbor algorithm. *Proceedings - 2010 7th International Conference on Fuzzy Systems and Knowledge Discovery, FSKD 2010*, *1*(Fskd), 91–94. https://doi.org/10.1109/FSKD.2010.5569740

Tomašev, N., Radovanović, M., Mladenić, D., & Ivanović, M. (2014). Hubness-based fuzzy measures for high-dimensional k-nearest neighbor classification. *International Journal of Machine Learning and Cybernetics*, *5*(3), 445–458. https://doi.org/10.1007/s13042-012-0137-1

Wang, J., Neskovic, P., & Cooper, L. N. (2007). Improving nearest neighbor rule with a simple adaptive distance measure. *Pattern Recognition Letters*, *28*(2), 207–213. https://doi.org/10.1016/j.patrec.2006.07.002

Zhang, S., Zong, M., Sun, K., Liu, Y., & Cheng, D. (2014). Efficient kNN Algorithm Based on Graph Sparse Reconstruction. In X. Luo, J. X. Yu, & Z. Li (Eds.), *Advanced Data Mining and Applications. ADMA 2014. Lecture Notes in Computer Science* (Vol. 8933, pp. 356–369). Springer, Cham. https://doi.org/10.1007/978-3-319-14717-8_28

Zhang, X., & Li, Y. (2013). A Positive-biased Nearest Neighbour Algorithm for Imbalanced Classification. In J. Pei, V. S. Tseng, L. Cao, H. Motoda, & G. Xu (Eds.), *Advances in Knowledge Discovery and Data Mining. PAKDD 2013* (Vol. 7819, pp. 293–304). Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-642-37456-2_25

Zhong, X.-F., Guo, S.-Z., Gao, L., Shan, H., & Zheng, J.-H. (2017). An Improved k-NN Classification with Dynamic k. *Proceedings of the 9th International Conference on Machine Learning and Computing*, 211–216. https://doi.org/10.1145/3055635.3056604