Masaryk University
Faculty of Informatics

Ensembles for anomaly detection

Bachelor's Thesis

Tomáš Krutý

Brno, Spring 2018

Masaryk University
Faculty of Informatics

Ensembles for anomaly detection

Bachelor's Thesis

Tomáš Krutý

Brno, Spring 2018

This is where a copy of the official signed thesis assignment and a copy of the Statement of an Author is located in the printed version of the document.

Declaration

Hereby I declare that this paper is my original authorial work, which I have worked out on my own. All sources, references, and literature used or excerpted during elaboration of this work are properly cited and listed in complete reference to the due source.

Tomáš Krutý

Advisor: doc. RNDr. Lubomír Popelínský, Ph.D.

Acknowledgements

I appreciate the honest and accurate feedback received from doc. RNDr. Lubomír Popelínský, Ph.D., who helped me tremendously with this thesis. The findings, figures and tables all were made much better because of doc. Popelinsky's meaningful comments; his untiring effort was essential in shaping this work.

Additionally, I'd like to express my gratitude to my family and friends for being extremely supportive throughout the development of this thesis.

Abstract

This thesis presents the main principles of ensemble learning and research of previous work done on the subject of ensemble methods for anomaly detection. Next the thesis presents various ways to create ensemble methods for anomaly detection using the local outlier factor and class outlier: distance-based methods as base learners for these ensembles and provides implementation of selected ensembles in R language. Finally the thesis evaluates the selected ensembles using generated and real world datasets and shows the full results of this evaluation.

Keywords

ensembles, anomaly detection, machine learning

Contents

List of Tables

List of Figures

Introduction

The definition states that an anomaly is: "an observation which deviates so much from the other observations as to arouse suspicions that it was generated by a different mechanism." (Hawkins [1], 1980) In literature an anomaly can also be called an outlier, in this thesis the words outlier and anomaly are used interchangeably and have the same meaning. This thesis mostly looks at the local outlier factor and the class outlier: distance based method of detecting outliers and ensembles created based on those methods. The mentioned methods will be further discussed and analyzed in chapters 3, 4 and 5.

In general, detecting anomalies in data is significant mainly for two reasons. The first reason is to remove anomalies from data. If the intention is to classify the data or to predict based on the data, removing the faulty readings is critical for producing an accurate output. The second reason is that anomalies may carry valuable information, for example in fraud detection or to detect only the interesting observations in datasets.

The main problem of outlier detection algorithms is that every algorithm provides accurate results on a particular dataset but it will be, to a certain extent, inaccurate on most real-world heterogeneous data. It is stated in literature that use of ensembles should improve the accuracy of the result and minimize a need to have data in an exact format required by a specific algorithm [2]. Ensembles for anomaly detection are meta-algorithms that use several different algorithms, or the same algorithm with different thresholds defining what an outlier is, on same dataset. Alternatively the same algorithm can be used on selected subsets of the dataset, to gain the expected quality. The ensemble is also used to compare and combine the outputs from various algorithms and to provide a meaningful conclusion.

The most challenging task of creating an ensemble is combining the outputs from multiple algorithms. Most often the results from different algorithms are not directly comparable and must be normalized prior to the comparison. The normalization and combination functions of the outputs will be further discussed in later chapters.

The aim of this thesis is to present existing approaches to designing ensembles for anomaly detection as well as to present the main

principles of ensemble learning algorithms over-all. Next the thesis presents multiple ways of creating ensembles based on local outlier factor and class outlier: distance based methods and further analyzes and provides implementation in R language for two particular ensembles, one based on local outlier factor and one on class outlier: distance based method. Last but not least the objective is to evaluate these methods using various configurations of the ensemble and compare to non-ensemble methods.

The thesis consists of five chapter. The 1^{st} chapter describes the main principles of ensemble learning, their advantages and drawbacks as well as the motivation behind using ensemble methods. In addition, it also discusses the categorization of ensemble methods by component type as well as component independence. The 2^{nd} chapter details existing ensembles for anomaly detection, the research questions about ensembles for anomaly detection and the challenges posed. The 3^{rd} chapter reviews possible ways of creating an ensemble method for detecting anomalies based on a local outlier factor. Chapter 3 presents the implementation of a selected ensemble method based on local outlier factor. Chapter 4 is devoted to the class outlier and the class outlier: distance based method, further possible ways of creating ensemble methods based on class outlier: distance based method and implementation of a selected ensemble in R language. Finally chapter 5 provides the evaluation of those ensemble methods and comparison to non-ensemble methods. The 5^{th} chapter is dedicated to the conclusion and findings from the evaluation.

1 Principles of ensemble learning

The ensemble paradigm in machine learning is a method of combining multiple different algorithms (or different instances of an algorithm) to perform a task in order to obtain more robust results [2]. This chapter provides basic principles of the ensemble paradigm.

1.1 Motivation

Ensembles in machine learning have gained a lot of attention and are being widely used for tasks as classification, clustering or regression [3, 4, 5, 6, 7]. The use of ensembles in anomaly detection has also been gaining significant attention in the past few years [2, 8, 9, 10, 11, 12, 13, 14, 15, 16], but is not nearly as widely researched and formalized, as will be discussed in next chapter.

The attention to ensemble learning is well deserved; it proposes a simple concept that promises more robust and data independent results. Ensemble learning is a method, or a meta-algorithm. The main idea behind ensembles is to execute multiple base learners and combine their outputs. Base learners can consist of different algorithms, different instances of the same algorithm with different parameters, or the same algorithm executed on different subsets of data. The task of deciding on the type and number of base learners is left for the user, or occasionally, the architect of an ensemble.

1.2 Advantages

As was already mentioned, the main goal of ensemble learning is to provide more robust results independent from specific data types. This goal is achieved by using multiple base learners. Many learning algorithms suffer from being inconsistent as they are based on specific types of data, meaning that they can work very well on one dataset but poorly on another. This can be specific to data size, density, dimensions and so on. Ensemble learning tackles this challenge by using multiple learning algorithms. With this approach, even if there is an algorithm

with low precision on one given dataset present in the ensemble, the other base learners will most likely outvote it.

In addition to reducing the dependency, according to Dietterich as he stated in [17], there are three main challenges in classical approaches that can be, to a certain extent, solved by ensemble learning.

The statistical problem; this problem can occur when a learning algorithm calculates the same probability for multiple outputs. In this case the algorithm must choose one of them, and the chosen output may not be the best one. If an ensemble would be used in this case, this problem could be avoided since it is less likely, although not impossible, that the majority of base learners would choose the same faulty output and, once again, they would be outvoted.

The computational problem; this problem may appear when the dataset is too large and the algorithm struggles to find the best output, e.g. getting stuck in local minima while using the gradient descent method. The risk of this happening can be reduced, just as in the statistical problem, by the voting of multiple base learners.

The last problem proposed by Dietterich that can be solved by ensembles is the so-called representational problem. This problem results from the inability to extract an acceptable approximation from the dataset. By taking a vote from these approximations, the ensemble may be able to come up with a more precise approximation.

1.3 Drawbacks

From what was already said in this chapter, it may seem that ensembles will always out perform regular learning algorithms, and the question may arise about why not use ensembles everywhere that we use single learning algorithms now? Although it is possible to substitute every single learner by an ensemble, it is not advised to do so, as it will not always provide superior results and can even invalidate some needed features that are provided by single learners. As is the case with most methods in machine learning, one cannot simply have the best solution for a problem. The best algorithm or method chosen to solve a problem is found experimentally, therefore it is neither precise nor correct to assume that ensemble methods will always outperform singular learners, keep in mind the fact that, just as with regular learners, the

user has to choose what type of ensemble to use, and every type of ensemble has its own advantages and drawbacks. The different types of ensembles and how they are built will be covered later in this chapter, that being said, there are multiple specific disadvantages to ensembles in general.

As Holder mentioned in his class on Artificial Intelligence in [18], one of the disadvantages of ensembles is that it is *harder to understand* the complex reasons of why the ensemble choose the particular output, it is not that trivial to trace back the steps and calculate it as would be with single learners. Another disadvantage that affects some of the ensembles is that since an ensemble executes multiple learners, the *computational speed and space required* may greatly increase. Naturally the process of building the ensembles also introduces new challenges such as what base learners to use, and *how to combine the outputs of the base learners*. Especially when using different algorithms as base learners, as their outputs may not be directly comparable, thus the ensemble must either normalise the outputs or combine them in a way that is not affected by the actual values of the output, for example by voting. Despite the fact that the latter challenges affect the architect of the ensemble, and not users directly, they should be kept in mind when talking about ensemble learning.

1.4 Categorization of ensemble methods

As Charu Aggarwal proposed in [2] ensemble methods can be categorised either by component independence, or component type. This section will discuss this categorisation and will provide examples of specific ensembles for every category.

1.4.1 By component independence

The categorisation by component independence divides ensemble methods into two categories, the sequential ensembles and the independent ensembles.

In sequential ensembles, the base learners are applied sequentially, meaning that the future application of learners will be dependent on previous applications. The main idea behind sequential ensembles

is that the previous application of a base learner adapts the data for a finer execution of later applications. The simplest example of a sequential ensemble is Adaboost [19], matter of fact boosting in general is a sequential ensemble method. In boosting a different subset of data is being used for each learner, where every learner uses a more difficult data set then the previous one. Basically, the points that were, for example, incorrectly classified in previous application are more likely to be chosen for the next applications.

The next, and last, category in categorisation by component independence is the independent ensemble. The idea is that the base learners are executed independently of each other, and their outputs are combined after all of the base learners have been applied. This category also introduces the challenge of combining the outputs of different base learners, which, as was already mentioned, may not be a trivial task. An example of an independent ensemble is bootstrap aggregating or bagging for short [6]. Bagging uses an idea analogous to boosting, to be more explicit; boosting is a special form of bagging. In bagging, a subset of data, called bags, is chosen randomly for each application of a base learner. Because of the arbitrary creation of the bags this ensemble is, in fact, independent.

1.4.2 By component type

The categorisation by component type also divides ensembles into two categories, model-centred and data-centred ensembles.

In model-centred ensembles, as the name suggests, the base learners are built using different models on the same data. An example that explains a model-centred ensemble is a cross validation ensemble. This method is recognised as an ensemble due to its use of cross validation to determine which algorithm will be used to perform the given task. It must train and test different base learners and decide which one is the most accurate.

Data-centred ensembles are opposite to model-centred, these ensembles build the same model, but multiple times, each time with a different dataset. An example of a data-centred ensemble is the random forest method [5]. The random forest method randomly selects observations and features to build multiple decision trees. Since the selection process is random the data will most probably be different

for each decision tree while the model and the decision tree remain the same. Of course, bagging or boosting are also satisfactory examples of data-centred ensembles.

2 Ensembles for anomaly detection

In contrast to ensemble methods for tasks such as clustering, classification or regression, ensembles for anomaly detection is not a well-researched and formalised area of machine learning. Given the successes of ensemble methods in other tasks, many researchers have been gaining an interest in ensembles for anomaly detection. This chapter will focus on work that has already been achieved in the field of anomaly detection using ensemble methods. To be more exact, this chapter will cover research questions posed by Arthur Zimek in [8] and responses to them. New methods and suggestions for building ensembles and novel ensemble methods for anomaly detection. For the purposes of this thesis, this chapter will only cover findings from the year 2013 onward. A well written research of related work prior to year 2013 can be found in [20].

2.1 Research Questions

Arthur Zimek posed serious research questions in [8]. The insufficient answers to these questions have limited the understanding of ensemble methods for anomaly detection thus rendering further research unnecessarily difficult.

The first issue he addressed is the lack of evaluation methods of outlier detection algorithms in general. He stated that there is no systematic approach in evaluating outlier detection methods. He then described how these methods can be evaluated, e.g. by using the Area Under Curve (AUC), this evaluation method plots true positive rate against a false positive rate and calculates the area under this curve. Of course one has to know what observations are outliers in order to know the true and false positive rates, and this ground truth is rarely available in anomaly detection. This and other related challenges were undertaken in an extensive experimental study [21].

Zimek also concentrates on the matter of diversity in ensemble anomalies and mentions that absence of diversity may result in less accurate outputs compared to single detectors. This concern was later supported and shortly discussed in [12, 22] and partially studied in [23]. In [24] Zimek proposes a solution to the lack of diversity by adding

controlled noise to data. In this paper Zimek further proves that this diversity induction widens the gap between inliers and outliers, therefore effectively removing the complications that occur when an outlier is adjacent to inliers.

Lastly Zimek talks about the scarcity of an effective method of choosing appropriate ensemble members. Charu Aggarwal provides a discussion and heuristics on choosing outlier detection algorithms [25], however an effective method of choosing the members cannot be found in the literature so far.

2.2 Challenges and resolutions

Charu Aggarwal presented general information about ensembles for anomaly detection vital for future research in [2]. As discussed in the previous chapter he specified categorisation of ensemble methods. Besides categorisation, he argues about the possibility of reusing concepts and knowledge gained from known ensemble methods for different machine learning tasks, such as clustering or classification and gives examples of how could these concepts be reused in anomaly detection tasks. Aggarwal also claims that boosting is not fit to be applicable to unsupervised anomaly detection, yet the CARE ensemble method analyzed in [15] applies the boosting approach to sequentially remove anomalies from the original dataset to form a better model. Finally he examines the potential problems that may arise from defining the combination function of an ensemble, namely the normalization and combination problems. With the deeper understanding he gained while working on [26] to formally define ensemble methods for outlier detection, he proposed two new combination functions based on the theory discussed in this paper.

The problem of combination functions was further discussed in [27]. Additionally a novel combination function was proposed named inverse cluster weighted averaging. The main idea supporting inverse cluster weighted averaging is clustering the vectors of outlier scores returned by base learners and assigning weights to the base learners that are inversely proportional to the cluster sizes.

As briefly mentioned above, a significant obstacle for researchers was the fact that the formalization of ensemble methods for anomaly

detection is scarce. Aggarwal examined the theoretical foundations of these methods and their application [26]. He also analyzed the similarity with the theory behind classification methods and came to the conclusion that even though anomaly detection has an unsupervised nature, the theory underlying anomaly detection is more similar to a supervised classification problem rather than to unsupervised problems such as clustering.

The majority of ensemble methods provide a more robust solution but increase computational complexity. The increased computational time of ensembles could be a potential issue given the volume of data being analyzed daily. A method presented in [22] addresses both of these challenges. The paper shows that the sub-sampling method proves to be more robust and can improve the performance in comparison to the base learner while needing less computational time than most ensembles and is even comparable to the computational time of a base learner for small number of base learners.

2.3 Novel ensemble methods for anomaly detection

Because of the increasing interest in ensemble methods for anomaly detection, plenty of novel ensemble methods have emerged in the literature [9, 10, 11, 12, 13, 14, 15, 16].

Every novel ensemble method experimentally proves to be superior to the base learners. The methods are, for the most part, independent except the CARE approach [15], which is a sequential ensemble based on boosting, as mentioned above. Further, a new paradigm for outlier detection appeared that tries to get the best of both worlds, the supervised and unsupervised outlier detection. This method takes advantage of prior knowledge available about some anomalies in datasets. The semi-supervised method, as Micenkova calles it, combines the information from unlabeled data and supervision of some labeled data. This method is introduced in [10]. And finally a rather interesting method based on neural networks is proposed in [12], this method uses auto-encoders to reconstruct the data and uses the reconstruction error as the outlier score. As the article [12] states a big problem of neural networks is their sensitivity to noise and requirement for large datasets to work robustly, which causes them to be slow.

The proposed algorithm in [12] randomly varies on the connectivity architecture of the autoencoder and combine this method with an adaptive sampling method to obtain efficiency and effectiveness.

3 Local outlier factor ensemble

The local outlier factor, or LOF for short, was introduced in 2000 in [28] and made a truly groundbreaking contribution to the discipline of anomaly detection. Local outlier factor is understood to be the first method that assigned each observation a degree to which the given observation is anomalous. Prior to the LOF method whether an observation is an outlier or not was viewed as a binary property. Breuning showed in [28] that for many situations it may be more meaningful to assign each observation an outlier factor rather than a binary value. Local outlier factor is a density-based method, meaning that this method is capable of detecting outliers based on their surroundings. If an observation lies in a fairly dense area it's local outlier factor will be close to one, whereas when an observation belongs to a more sparse area it's local outlier factor will be larger. The size of the surrounding taken into account when computing the local outlier factor is denoted by a value of k given as an input for the LOF function.

The local outlier factor is driven by three main concepts that the function uses to calculate the outlier factor. Those are: *core distance*, *reachability distance* and *local reachability distance*. Core distance of an observation is the distance between the observation and it's k^{th} nearest neighbor. The reachability distance between observation $o1$ and $o2$ is the maximum of core distance of $o1$ and the actual distance between $o1$ and $o2$. Finally the local reachability distance is inversely proportional the the average reachability distance of it's k nearest neighbors. The local outlier factor is computed as follows.

$$LOF_k(o) = \sum_{o' \in KNN(o)} localReachabilityDistance(o') * \sum_{o' \in KNN(o)} reachabilityDistance(o', o) \tag{3.1}$$

The local outlier factor implementation used in this thesis to create an ensemble was implemented in R language in data mining with R package, officially called DMwR, developed by Luis Torgo. The function, called *lofactor*, is described in [29].

3.1 Creating the ensemble

There are a large number of ways to create an ensemble from a local outlier factor. The fact that the algorithm that computes the local outlier factor uses an integer value of k as an input, makes it possible to create a so-called parameter tuning ensemble. As the name suggests, parameter tuning ensembles are ensembles that execute multiple instances of the same algorithm each time with a different combinations of parameters. This allows the ensemble then to either combine the outputs of each instance or choose the best combination of inputs. As expected, to choose the best combination of inputs, the algorithm needs some sort of feedback on how the instance with given parameters performed, this is possible with classification tasks where usually the dataset is split into training and testing data, but very hard with anomaly detection because of the scarcity of ground truth in real world datasets. So the only reasonable way to create a parameter tuning ensemble for anomaly detection is through combining the individual outputs. As it is stated in Chapter 1 of this thesis, the process of combining outputs of base learners can be a challenging task. During the work on this thesis, the following experiments in combining functions have been conducted.

3.2 Choosing the combination function

The easiest and most straightforward combination function is the *simple mean*. The simple mean combination function sums all results for each observation and divides the result by the number of base learners. The next combination function that was analyzed for this thesis is the *weighted average*. The weighted average combination function assigned each base learner a weight. The weight was assigned based on the proximity of base learner's k value to the optimal value of k. This introduces the problem that an optimal value of k is highly dependent on data and choosing an optimal k is also a very problematic task. Nonetheless, Breunig recommends a method how one can choose the k value in [28]. Breuning also recommends in his paper when choosing the value for k to choose a minimum and maximum value of k. Again depending on the dataset, where the minimum value should

be the size of the smallest cluster in a given dataset, and similarly the maximum value of k, should be the size of the largest cluster in the dataset. Once the algorithm has the minimum and maximum values of k, it can iterate over all the possible values and return the maximum factor for each observation. Breuning's method can be also thought of as a combination function, this combination function is known as the *maximum* combination function. And the last combination function that was tested is *voting*. The ensemble that uses voting as its combination function also needs the number, or percentage, of outliers as input. Each base learner will vote for observations it believes to be the most anomalous. Those votes are summed and returned as an output of the ensemble. For the purposes of this thesis only the simple mean combination function will be thoroughly analyzed, leaving the reminder of the combination functions for future work.

3.3 Analyzed ensemble

As mentioned in the previous section, the ensemble based on the local outlier factor analyzed in this thesis uses the simple mean combination function and Luis Torgo's implementation of the local outlier factor function that can be found in the Data mining with R package. Below is presented a code snippet that computes this ensemble. This implementation takes a sequence of integers that hold the k values used as input for individual base learners and a location of the file carrying the observation for which the local outlier factor will be calculated as input.

It is clearly seen that the sequence of k values determines the number of base learners as well as their configuration. The number of k values is the number of base learners the ensemble will use, and their values are the actual configurations of those base learners. This raises the problem of how to choose these properties of the ensemble. This thesis uses the experimental approach to this problem. In later chapter the thesis will discuss on the evaluation of the presented ensembles and will provide the ensemble configuration for each result.

The file containing the observation has to contain only numeric features for each observation, since this is a requirement by the *lofactor* function. It is easily shown that this requirement does not affect the

generality of the ensemble. If an enumeration type T, would appear in an observation, there exists an injective function f with signature $f : T \to \mathbb{N}$, since an enumeration type must have only a finite number of possible values. This effectively shows that it is possible to represent an enumeration type with numeric values. Thanks to the finality of the enumeration possible values, this representation is also distinctive.

If an attribute that carries text information exists in an observation, this attribute can be broken down into a sequence of characters. This reduces the problem to representing a single character with a numerical value, and each individual character can be represented by a numerical value with an arbitrary character encoding standard. Any other type would be some kind of a combination of the two former data types and numerical data types, however it would be highly irregular to find such data structures in data that one would aim to detect anomalies in.

```
lof.ensemble.mean <- function(Ks, data) {
  data <- read.csv(file = data, header = TRUE, sep = ",")
  outlier.scores <- vector(mode = "numeric", length = nrow(data))
  for(k in Ks) {
    score <- lofactor(data, k)
    outlier.scores <- cbind.data.frame(score, outlier.scores)
  }
  outlier.score <- simple_mean(outlier.scores)
  outlier.score <- cbind.data.frame(outlier.score, data)
  return(outlier.score)
}
```

4 Class outliers: distance-based ensembles

The class outliers: distance-based, or CODB, method to detect anomalies was introduced in 2007 by Nabil M. Hewahi and Matoz K. Saad in [30]. This method takes a different approach of detecting anomalies than the local outlier factor. While local outlier factor detects the degree of outlyingness of an observation based on its closest neighborhood, the class outlier: distance-based method calculates the outlier score based on classes of observations in the analyzed point's closest proximity. This way the algorithm takes advantage of the information about classes of individual observations. This allows the method to detect not only the observation with low local density as outliers, but also observations that lie in areas with a high concentration of different classes than the observation being analyzed. This approach can be used in real world scenarios for example for finding a politician with left-wing views but belonging to a right-wing party.

4.1 Class outliers

As the name implies, class outliers are those observations that may not seem to be anomalous without the class label, but start to raise suspicions once the class label is taken into account. One of the first papers that laid down the concept of mining for class outliers [31] defined the process of detecting anomalies as finding those observations that arose suspicions, taking into account the class labels, when given a set of observations with class labels. The figure 4.1 shows two class outliers. The two classes in figure 4.1 are distinguished by color. As is easily spotted, there is one observation with red class in the blue cluster and one with blue class in the red cluster. This example shows the importance of class outlier detection, as the observations that reside in clusters occupied by different classes are easily determined as anomalous just by looking at them, an anomaly detection method that does not take the class labels into consideration wouldn't be able to determine those observations as anomalous, they lie deep within a cluster and they are in a considerably dense neighborhood, yet when looking at their class labels it's obvious those observations are most probably outliers.

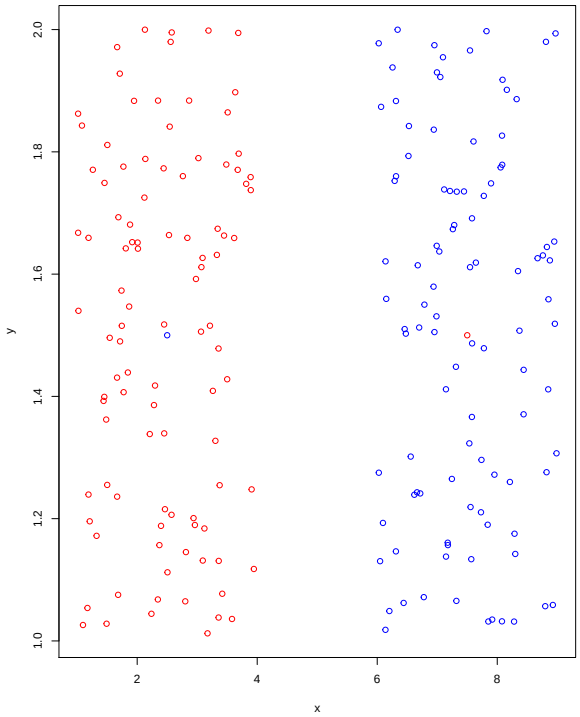

Figure 4.1: Scatter plot showing two class outliers

4.2 Class outlier: distance-based

The class outlier: distance-based method uses three concepts to calculate a so-called class outlier factor. The method assigns this factor to every observation. The three concepts are the *probability of the class label*, *deviation*, and *k-distance*. The probability of a class label, or PCL, is the ratio of number of observation with same class labels as the class label of the analyzed observation, to the size of observation's closest neighborhood. The size of closest neighborhood, or k, is given as an input to this method. The deviation is sum of distances between the analyzed observation and all other observations with same class as the analyzed observation. Intuitively, how much the observation deviates from a subset of observations with the same class. And lastly the k-distance is sum of distances between the observation and all of its k nearest neighbors. The class outlier factor is then computed as follows.

$$COF = k * \text{PCL} + (\alpha/\text{deviation}) + (\beta * \text{k-distance}) \tag{4.1}$$

Where α and β are the factors that control the importance of deviation and k-distance. Alternatively in 2009 an enhanced method of class outlier: distance based method was introduced in [32] called eCODB. This method does not use the α and β factors, instead the enhanced method uses a *min-max normalization* function for both deviation and k-distance.

$$COF = k * \text{PCL} + norm(\text{deviation}) + norm(\text{k-distance}) \tag{4.2}$$

Where the min-max normalization function normalizes the values and maps them to an interval between 0 and 1.

$$norm(x) = (x - min_x)/(max_x - min_x) \tag{4.3}$$

4.3 Creating the ensemble

The ensemble used in this thesis is going to use the enhanced method of class outlier: distance based approach to detect class anomalies. Similarly to the local outlier factor ensemble, the ensemble to detect class outliers will also belong to the parameter tuning ensemble family

as the class outlier: distance based method also takes value k as input, where k is the size of the closest neighborhood. This being said this ensemble can use any of the combination functions described in the previous chapter, but once again only the simple mean combination function will be analyzed for the purposes of this thesis. As was the case with the local outlier factor ensemble the number of base learners as well as their configuration was decided experimentally and will be discussed later in the thesis.

The input parameters for the class outlier: distance-based ensemble are a list of k values that determines the number of base learners as well as their configuration and the path to a data source, where all of the observations, except the class attribute, are allowed to have only numeric features, but as was already shown, this does not affect the generality of the model. The class attribute has to be present for each observation and the only limitation set to the type of class label is that there has to exist a comparison method for equality as it is used to calculate the probability of class label which counts the occurrences of same labels to the class of observation being analyzed.

```
CODB.ensemble.mean <- function(Ks, data) {
  data <- read.csv(file = data, header = FALSE, sep = ",")
  outlier.scores <- vector(mode = "numeric", length = nrow(data))
  for(k in Ks) {
    score <- CODB(k, data)
    outlier.scores <- cbind.data.frame(score$cof, outlier.scores)
  }
  outlier.score <- rowMeans(outlier.scores)
  outlier.score <- cbind.data.frame(outlier.score, data)
  return(outlier.score)
}
```

5 Evaluation of analyzed methods

Evaluation of outlier detection methods is not a trivial task mainly because of the lack of datasets with ground truth of which observations are outliers and which aren't. Most of real world applications of anomaly detection uses unsupervised anomaly detection, and from the nature of unsupervised learning it is hard to determine with certainty whether the observations detected as being anomalous by an algorithm are in fact anomalies. Along with the problem of finding a suitable dataset there is the problem of defining the proper method of evaluation. This chapter will discuss the most common methods for evaluation and provide the results obtained when evaluating the presented ensembles in former chapters as well as comparing them to non-ensemble methods for detecting anomalies. The methods explained in this chapter were used to analyze common outlier detection methods in an extensive experimental study done by Zimek [21].

5.1 Precision at n

The *precision at n* evaluation method is perhaps the simplest method to measure the performance of outlier detection methods. The requirement of this method is that the number of outliers is known in advance. The precision at n is defined as the proportion of correctly identified observations as outliers in the top n ranks. When obtaining the precision at n, one would run the method that is being analyzed on a desired dataset and count the number of observations that the method correctly detected as outliers. The outlier detection method must, however assign an outlier score to each observation. Because of this limitation, the method cannot be used to evaluate all outlier detection methods. After obtaining the number of accurately determined outliers, the method divides this number by n, returning a percentage of *precision at n*.

$$P@n(n, O) = \frac{|\{o \in O | rank(o) \leq n\}|}{n} \tag{5.1}$$

Where n is the number of outliers in a given dataset, O is a set of the ground truth outliers and function $rank(0)$ determines the order of

the observation based on its outlier factor, the more anomalous an observation is the lower it's rank will be.

To make the comparison of the results calculated on dataset with different outlier rates easier, Zimek suggested the adjustment for chance in [21] And proposed the following formula for $n \leq |O|$.

$$Adjusted\ P@n(n, O, N) = \frac{P@n - |O|/N}{1 - |O|/N} \tag{5.2}$$

Where N is the size of the dataset used. For larger n than $|O|$, the maximum $|O|/n$ must be used instead of the value 1.

5.2 Average precision

The precision at n evaluation method is highly dependent on the selection of n parameter. The n parameter can be affect greatly the results of precision at n evaluation method. For those reasons Zimek proposes an aggregate function over a wide range of possible choices of n in [21]. The method that calculates this aggregate results is called the *average precision*.

$$AP(O) = \frac{1}{|O|} * \sum_{o \in O} P@rank(o) \tag{5.3}$$

This method iterates over all possible ranks of outliers from ground truth and sums the precisions at those ranks.

As was the case with precision at n, the paper also suggests the adjustment for chance for the average precision method.

$$Adjusted\ AP(O, N) = \frac{AP - |O|/N}{1 - |O|/N} \tag{5.4}$$

5.3 Area under curve

The most commonly used method to evaluate an outlier detection algorithm is based on the receiver operating characteristic curve. The receiver operating characteristic curve is, simply put, the plot of true positive rate against the false positive rate. The algorithm of plotting

the receiver operating characteristic curve iterates over all observations, sorted by the outlier factor assigned to each observation by the analyzed outlier detection method from the most anomalous observation to the least anomalous ones. For each observation it calculates the true positive rate against the false positive rate of the observations already seen by the algorithm. Now that the receiver operating characteristic curve is plotted, the evaluation method calculates the area under this curve to obtain the result of the evaluation. Because of the fact that true positive rate and false positive rate only obtains values between 0 and 1, the area under such curve will also always be between 0 and 1. 1 being the best evaluation an algorithm can achieve, meaning it sorted all the outliers prior to all the inliers. An example of a receiver operating characteristic curve is shown in figure 5.1. If the outlier detection method would classify all of the observation correctly, only the true positive rate will increase until it reaches the value of 1 and the receiver operating characteristic curve would be identical to second curve in 5.1. It is clearly seen that the area under the latter curve shown in 5.1 is equal to 1, thus satisfying the statement that area under the receiver characteristic curve is equal to 1 only when the analyzed anomaly detection algorithm classified all outliers correctly.

5.4 Evaluation of local outlier factor ensemble

To evaluate local outlier factor ensemble, this thesis uses a subset of data used in Zimek's experimental outlier detection methods evaluation study. The data was chosen because of the ground truth that is present in this data and the fact that most other real world datasets lack this ground truth. For the purposes of this thesis only the area under receiver operating characteristic curve method was used as an evaluation measure as it is the most popular evaluation measure used in literature and makes evaluation of multiple methods on different datasets easier. The area under curve does not require any adjustment for chance and can be directly used for competitive evaluation of algorithms [21]. The thesis then compares the ensemble method with various configurations to the local outlier factor method of detecting anomalies.

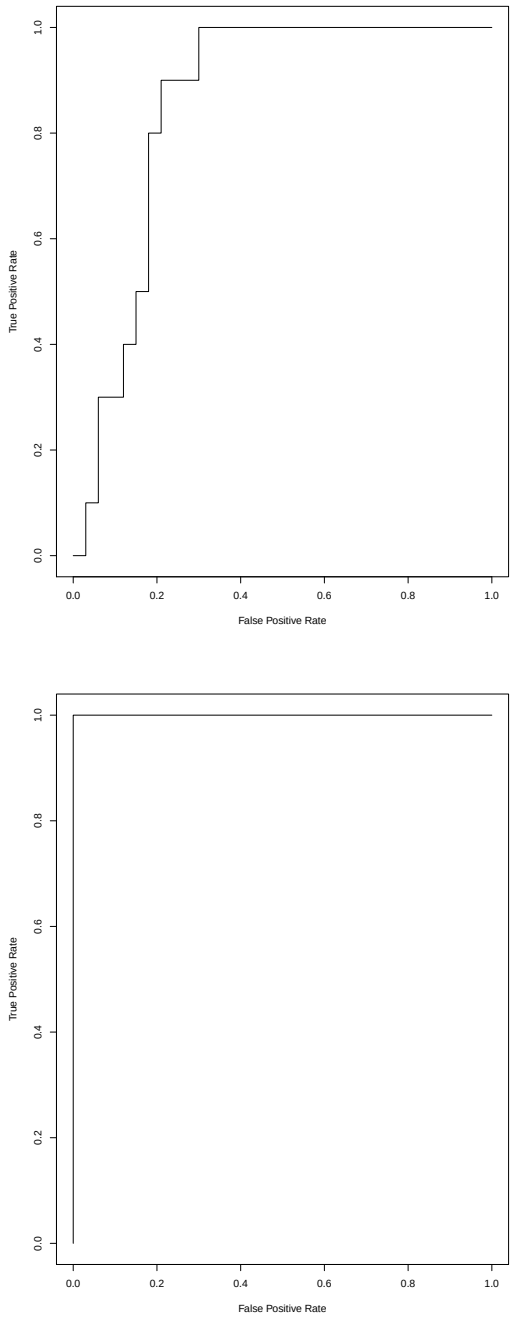

Figure 5.1: Receiver operating characteristic curve

The results show that the ensemble in general performs better than the non-ensemble, however the difference is not significant. When comparing an ensemble that iterates over all tested k values, the results are somewhat similar, for some datasets the non-ensemble methods even outperformed the ensemble methods. Keeping in mind the possible value range of the area under curve being between 0 and 1, the differences were on average somewhere between 0.001 to 0.05. In conclusion, it is safe to say that the ensemble methods performed slightly better than non-ensemble methods, but the two methods performed similarly and the precision of the ensemble methods should not and cannot be taken as the biggest advantage over the non-ensemble methods.

The most interesting finding while evaluating the methods is that the ensemble methods were much more robust that the non-ensemble methods. Meaning that in the non-ensemble local outlier factor the result was very much dependent on the k value supplied. For example, while evaluation the methods on the Wisconsin Diagnostic Breast Cancer dataset, the results of non-ensemble methods were ranging from 0.477 to 0.959, while the ensemble methods were only ranging from 0.829 to 0.959, see A. This is because while the non-ensemble methods solely rely on one detection algorithm, that may perform poorly, the ensemble methods have more base learners and one algorithm that performs poorly can still be corrected to a certain extent by others in the final output. The complete results of this evaluation can be found at github.com/tomkru/bachelor_code/blob/master/LOF_results.

5.5 Evaluation of class outliers: distance-based ensemble

For evaluation of the class outliers: distance-based the thesis could not use the data used to evaluate the local outlier factor ensemble since the data used to evaluate the local outlier factor ensemble did not contain the class attribute. For this method the thesis used data that was used to evaluate the RF-OEX ensemble method to detect class outleirs [16]. Like the evaluation methods used in local outlier factor ensemble, it also couldn't be used for evaluation the class outlier: distance-based ensemble because in some of the datasets it is not clear which observations are anomalous and which aren't, the ground truth is not present

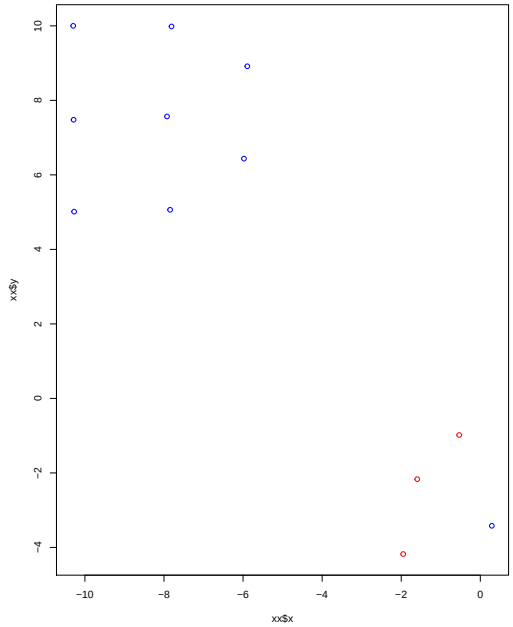

Figure 5.2: Positives Negatives Positive dataset

in those datasets. The evaluation of the class outlier: distance-based ensemble will consist of comparing the ensemble method to the non-ensemble eCODB method and interpreting why the outlier detection algorithm detected certain observations as outliers.

On the given dataset both the ensemble and the non-ensemble methods performed exceptionally well. Most of the ensemble as well as the non-ensemble methods detect the most anomalous observations as certain outliers. The ground truth is missing from those datasets, but manual analysis of the data provides more awareness. For instance in the data in the PositivesNegativesPositive dataset shown in 5.2, it is obvious that the most bottom right observation with the blue class is the most anomalous point with respect to the class label. Remarkably, for the non-ensemble methods for some k values, to be exact larger than 6, the non-ensemble methods returned the observation with red class as outliers, see 5.1. The ensemble methods were much more robust when it came to configuration of the ensemble. This supports the result from the previous chapter. The outcomes of non-ensemble methods com-

pared to the ensemble methods are fairly similar when using rather small values of k, smaller than 6. In conclusion the evaluation of the class outlier: distance-based ensemble only confirmed what was found while testing the local outlier factor ensemble. The ensembles, while they bring a slight improvement to the non-ensemble methods, the improvement is not great enough to bring much difference to the final outliers found by the algorithms. The true improvement that ensemble methods introduce is that the user of the outlier detection method can spend considerably less time configuring the outlier detection model in order to achieve the same results as if the best possible k value would be used. The complete results of this evaluation can be found at github.com/tomkru/bachelor_code/tree/master/CODB_Results.

Table 5.1: eCODB Ensemble vs Non-Ensemble method for Positives Negatives Positive dataset

COF for k = 2,3,4,5,6,7,8	COF for k = 8	x	y	class
4.77179611675106	7.85496555510782	-10.2952744172085	10.0004917621102	p
4.88058120668359	7.92806599967713	-10.2723232914546	5.01253279586995	p
4.77405451916639	7.76150989098437	-10.2860218691297	7.47928279288922	p
4.76894499676176	7.85340985416669	-7.80922289509702	9.98372483717126	p
4.77750800406319	7.76046206012108	-7.92289705720816	7.56794629196642	p
4.87081295181143	7.92219436530188	-7.84491129482959	5.06296770608633	p
4.95196025256673	7.79150913307359	-5.89274381316187	8.91493397386242	p
4.98474395737	7.81021073577631	-5.97709821252976	6.4361117908871	p
2.10803676179255	4.86693975563246	0.291803518271949	-3.41908428203293	p
3.52140293424621	3.63717684676111	-0.533355542211379	-0.982783020702809	n
3.38906125071904	3.72047617926564	-1.59463482314756	-2.1673327337902	n
3.41877487017057	3.97624705347108	-1.94887523530788	-4.17721756007278	n

Conclusion

This thesis introduced the main principles of ensemble learning. It explains how ensemble learners work in general and how the knowledge of ensemble learners gained in the past can be applied to known outlier detection methods. The thesis also talks about some of the advantages of ensemble learning, for instance robustness and data independence, and disadvantages such as higher computational complexity due to ensemble methods using numerous base learners. The thesis also shows one of the ways that ensemble methods can be classified. In addition the thesis provides a thorough research of previous work and accomplishments in the study of ensembles for anomaly detection. Next, the thesis proposes a number of ways to build ensemble methods from local outlier factor and class outlier: distance-based methods as base learners and presents implementation and analyses of two selected ensemble methods, one that uses the local outlier factor for base learners and one that uses class outlier: distance-based method, and also explains how the two mentioned outlier detection methods work. Finally, the thesis presents multiple ways of evaluating outlier detection methods and reviews the difficulty of evaluation due to the scarcity of ground truth in most real life data sets and provides the full evaluation results of both selected ensembles.

Future work on the subject of ensembles for anomaly detection should include the creation of a testing environment to automate the process of evaluating the ensemble methods for anomaly detection as this has not been addressed in the literature to date. Implementation of the remaining ensemble methods suggested in this thesis will also be a subject of future work. Furthermore a standardized package of the implementations should be created and submitted to a known and accessible repository such as the comprehensive R archive network. Finally an extensive study on the evaluation of known ensemble methods should be undertaken.

Bibliography

1. HAWKINS, Douglas M. *Identification of outliers*. Springer, 1980.
2. AGGARWAL, Charu C. Outlier Ensembles: Position Paper. *SIGKDD Explor. Newsl.* 2013, vol. 14, no. 2, pp. 49–58. ISSN 1931-0145. Available from DOI: 10.1145/2481244.2481252.
3. BICKEL, Steffen; SCHEFFER, Tobias. Multi-view clustering. In: *ICDM*. 2004, vol. 4, pp. 19–26.
4. MULLER, Emmanuel; GUNNEMANN, Stephan; FARBER, Ines; SEIDL, Thomas. Discovering multiple clustering solutions: Grouping objects in different views of the data. In: *Data Engineering (ICDE), 2012 IEEE 28th International Conference on*. 2012, pp. 1207–1210.
5. BREIMAN, Leo. Random forests. *Machine learning*. 2001, vol. 45, no. 1, pp. 5–32.
6. BREIMAN, Leo. Bagging predictors. *Machine learning*. 1996, vol. 24, no. 2, pp. 123–140.
7. FREUND, Yoav; SCHAPIRE, Robert E. A desicion-theoretic generalization of on-line learning and an application to boosting. In: *European conference on computational learning theory*. 1995, pp. 23–37.
8. ZIMEK, Arthur; CAMPELLO, Ricardo J.G.B.; SANDER, Jörg. Ensembles for Unsupervised Outlier Detection: Challenges and Research Questions a Position Paper. *SIGKDD Explor. Newsl.* 2014, vol. 15, no. 1, pp. 11–22. ISSN 1931-0145. Available from DOI: 10.1145/2594473.2594476.
9. RAYANA, Shebuti; AKOGLU, Leman. Less is More: Building Selective Anomaly Ensembles. *ACM Trans. Knowl. Discov. Data*. 2016, vol. 10, no. 4, pp. 42:1–42:33. ISSN 1556-4681. Available from DOI: 10.1145/2890508.
10. MICENKOVÁ, Barbora; MCWILLIAMS, Brian; ASSENT, Ira. Learning Outlier Ensembles: The Best of Both Worlds—Supervised and Unsupervised. In: *Proceedings of the ACM SIGKDD 2014 Workshop on Outlier Detection and Description under Data Diversity (ODD2). New York, NY, USA*. 2014, pp. 51–54.

11. SALEHI, Mahsa; LECKIE, Christopher A; MOSHTAGHI, Masud; VAITHIANATHAN, Tharshan. A relevance weighted ensemble model for anomaly detection in switching data streams. In: *Pacific-Asia Conference on Knowledge Discovery and Data Mining*. 2014, pp. 461–473.

12. CHEN, Jinghui; SATHE, Saket; AGGARWAL, Charu; TURAGA, Deepak. Outlier Detection with Autoencoder Ensembles. In: *Proceedings of the 2017 SIAM International Conference on Data Mining*, pp. 90–98. Available from DOI: 10.1137/1.9781611974973.11.

13. BOSMAN, Hedde H.W.J.; IACCA, Giovanni; TEJADA, Arturo; WÖRTCHE, Heinrich J.; LIOTTA, Antonio. Ensembles of incremental learners to detect anomalies in ad hoc sensor networks. *Ad Hoc Networks*. 2015, vol. 35, pp. 14–36. ISSN 1570-8705. Available from DOI: https://doi.org/10.1016/j.adhoc.2015.07.013. Special Issue on Big Data Inspired Data Sensing, Processing and Networking Technologies.

14. BANDARAGODA, T. R.; TING, K. M.; ALBRECHT, D.; LIU, F. T.; WELLS, J. R. Efficient Anomaly Detection by Isolation Using Nearest Neighbour Ensemble. In: *2014 IEEE International Conference on Data Mining Workshop*. 2014, pp. 698–705. ISSN 2375-9232. Available from DOI: 10.1109/ICDMW.2014.70.

15. RAYANA, S.; ZHONG, W.; AKOGLU, L. Sequential Ensemble Learning for Outlier Detection: A Bias-Variance Perspective. In: *2016 IEEE 16th International Conference on Data Mining (ICDM)*. 2016, pp. 1167–1172. Available from DOI: 10.1109/ICDM.2016.0154.

16. NEZVALOVÁ, Leona; POPELÍNSKÝ, Luboš; TORGO, Luis; VACULÍK, Karel. Class-Based Outlier Detection: Staying Zombies or Awaiting for Resurrection? In: *International Symposium on Intelligent Data Analysis*. 2015, pp. 193–204.

17. DIETTERICH, Thomas G et al. Ensemble learning. *The Handbook of Brain Theory and Neural Networks*. 2002.

18. HOLDER, Larry. *Lecture notes in Artificial Intelligence*. Washington State University, 2006.

19. FREUND, Yoav; SCHAPIRE, Robert; ABE, Naoki. A short introduction to boosting. *Journal-Japanese Society For Artificial Intelligence*. 1999, vol. 14, no. 771-780, pp. 1612.

20. PEKARČÍKOVÁ, Zuzana. *Detekce odlehlých bodů v klasifikovaných datech [online]*. 2013. Master thesis. Masaryk university, Faculty of informatics, Brno. Supervised by Lubomír POPELÍNSKÝ.

21. CAMPOS, Guilherme O.; ZIMEK, Arthur; SANDER, Jörg; CAMPELLO, Ricardo J. G. B.; MICENKOVÁ, Barbora; SCHUBERT, Erich; ASSENT, Ira; HOULE, Michael E. On the evaluation of unsupervised outlier detection: measures, datasets, and an empirical study. *Data Mining and Knowledge Discovery*. 2016, vol. 30, no. 4, pp. 891–927. ISSN 1573-756X. Available from DOI: 10.1007/s10618-015-0444-8.

22. ZIMEK, Arthur; GAUDET, Matthew; CAMPELLO, Ricardo J.G.B.; SANDER, Jörg. Subsampling for Efficient and Effective Unsupervised Outlier Detection Ensembles. In: *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. Chicago, Illinois, USA: ACM, 2013, pp. 428–436. KDD '13. ISBN 978-1-4503-2174-7. Available from DOI: 10.1145/2487575.2487676.

23. SLUBAN, Borut; LAVRAČ, Nada. Relating ensemble diversity and performance: A study in class noise detection. *Neurocomputing*. 2015, vol. 160, pp. 120–131. ISSN 0925-2312. Available from DOI: https://doi.org/10.1016/j.neucom.2014.10.086.

24. ZIMEK, Arthur; CAMPELLO, Ricardo J. G. B.; SANDER, Jörg. Data Perturbation for Outlier Detection Ensembles. In: *Proceedings of the 26th International Conference on Scientific and Statistical Database Management*. Aalborg, Denmark: ACM, 2014, 13:1–13:12. SSDBM '14. ISBN 978-1-4503-2722-0. Available from DOI: 10.1145/2618243.2618257.

25. CHARU C. AGGARWAL, Saket Sathe. *Outlier Ensembles: An Introduction*. Springer, 2017. ISBN 978-3-319-54765-7.

26. AGGARWAL, Charu C.; SATHE, Saket. Theoretical Foundations and Algorithms for Outlier Ensembles. *SIGKDD Explor. Newsl.* 2015, vol. 17, no. 1, pp. 24–47. ISSN 1931-0145. Available from DOI: 10.1145/2830544.2830549.

27. CHIANG, Alvin; DAVID, Esther; LEE, Yuh-Jye; LESHEM, Guy; YEH, Yi-Ren. A study on anomaly detection ensembles. *Journal of Applied Logic*. 2017, vol. 21, no. Supplement C, pp. 1–13. ISSN 1570-8683.

Available from DOI: https://doi.org/10.1016/j.jal.2016.12.002.

28. BREUNIG, Markus M.; KRIEGEL, Hans-Peter; NG, Raymond T.; SANDER, Jörg. LOF: Identifying Density-based Local Outliers. *SIGMOD Rec.* 2000, vol. 29, no. 2, pp. 93–104. ISSN 0163-5808. Available from DOI: 10.1145/335191.335388.
29. TORGO, L. *Data Mining with R: Learning with Case Studies, Second Edition*. CRC Press, 2016. Chapman & Hall/CRC Data Mining and Knowledge Discovery Series. ISBN 9781315399096. Available also from: https://books.google.cz/books?id=2aedDQAAQBAJ.
30. HEWAHI, Nabil M; SAAD, Motaz K. Class outliers mining: Distance-based approach. *International Journal of Intelligent Systems and Technologies*. 2007, vol. 2, pp. 5.
31. HE, Zengyou; XU, Xiaofei; HUANG, Joshua Zhexue; DENG, Shengchun. Mining class outliers: concepts, algorithms and applications in CRM. *Expert Systems with Applications*. 2004, vol. 27, no. 4, pp. 681–697. ISSN 0957-4174. Available from DOI: https://doi.org/10.1016/j.eswa.2004.07.002.
32. SAAD, Motaz K; HEWAHI, Nabil M. A comparative study of outlier mining and class outlier mining. *Computer Science Letters*. 2009, vol. 1, no. 1.

A LOF Results

Table A.1: Ensemble LOF for k values: 10,11,12,13,14

k values = 10,11,12,13,14	LOF Results
Glass_withoutdupl_norm.arff	0.8677506775067757
Lymphography_withoutdupl_norm_idf.arff	0.5046948356807507
Shuttle_withoutdupl_v09.arff	0.9496923076923087
Waveform_withoutdupl_v07.arff	0.6839066706551074
WBC_withoutdupl_v09.arff	0.7154929577464776
WDBC_withoutdupl_v04.arff	0.9591036414565798

Table A.2: Ensemble LOF for k values: 3,4,5,6,7,8,9,10,11,12

k values = 3,4,5,6,7,8,9,10,11,12	LOF Results
Glass_withoutdupl_norm.arff	0.817344173441735
Lymphography_withoutdupl_norm_idf.arff	0.4976525821596239
Shuttle_withoutdupl_v09.arff	0.9350000000000006
Waveform_withoutdupl_v07.arff	0.6377325755309586
WBC_withoutdupl_v09.arff	0.64225352112676
WDBC_withoutdupl_v04.arff	0.9179271708683447

Table A.3: Ensemble LOF for k values: 3,4,5,6,7

k values = 3,4,5,6,7	LOF Results
Glass_withoutdupl_norm.arff	0.6818428184281824
Lymphography_withoutdupl_norm_idf.arff	0.4788732394366192
Shuttle_withoutdupl_v09.arff	0.7321538461538486
Waveform_withoutdupl_v07.arff	0.6001286269817562
WBC_withoutdupl_v09.arff	0.5610328638497643
WDBC_withoutdupl_v04.arff	0.848739495798317

Table A.4: Ensemble LOF for k values: 3,4,5,6,7,8

k values = 3,4,5,6,7,8	LOF Results
Glass_withoutdupl_norm.arff	0.7132791327913269
Lymphography_withoutdupl_norm_idf.arff	0.48591549295774594
Shuttle_withoutdupl_v09.arff	0.832615384615384
Waveform_withoutdupl_v07.arff	0.610062817828306
WBC_withoutdupl_v09.arff	0.569483568075116
WDBC_withoutdupl_v04.arff	0.863585434173667

Table A.5: Ensemble LOF for k values: 3,4,5,6,7,8,9

k values = 3,4,5,6,7,8,9	LOF Results
Glass_withoutdupl_norm.arff	0.7582655826558267
Lymphography_withoutdupl_norm_idf.arff	0.4894366197183093
Shuttle_withoutdupl_v09.arff	0.8787692307692311
Waveform_withoutdupl_v07.arff	0.618378701764881
WBC_withoutdupl_v09.arff	0.5863849765258207
WDBC_withoutdupl_v04.arff	0.8787114845938351

Table A.6: Ensemble LOF for k values: 3,4,5,6,7,8,9,10

k values = 3,4,5,6,7,8,9,10	LOF Results
Glass_withoutdupl_norm.arff	0.7853658536585373
Lymphography_withoutdupl_norm_idf.arff	0.4929577464788727
Shuttle_withoutdupl_v09.arff	0.9064615384615391
Waveform_withoutdupl_v07.arff	0.6256117259946141
WBC_withoutdupl_v09.arff	0.6061032863849757
WDBC_withoutdupl_v04.arff	0.8921568627450955

Table A.7: Ensemble LOF for k values: 4,5,6,7,8,9,10

k values = 4,5,6,7,8,9,10	LOF Results
Glass_withoutdupl_norm.arff	0.7891598915989166
Lymphography_withoutdupl_norm_idf.arff	0.4988262910798117
Shuttle_withoutdupl_v09.arff	0.8986153846153851
Waveform_withoutdupl_v07.arff	0.633368232126823
WBC_withoutdupl_v09.arff	0.618779342723004
WDBC_withoutdupl_v04.arff	0.9005602240896333

Table A.8: Ensemble LOF for k values: 5,6,7,8,9

k values = 5,6,7,8,9	LOF Results
Glass_withoutdupl_norm.arff	0.7701897018970196
Lymphography_withoutdupl_norm_idf.arff	0.4988262910798117
Shuttle_withoutdupl_v09.arff	0.8850000000000001
Waveform_withoutdupl_v07.arff	0.6346485192940422
WBC_withoutdupl_v09.arff	0.6112676056338017
WDBC_withoutdupl_v04.arff	0.8974789915966361

Table A.9: Ensemble LOF for k values: 6,7,8,9

k values = 6,7,8,9	LOF Results
Glass_withoutdupl_norm.arff	0.7891598915989166
Lymphography_withoutdupl_norm_idf.arff	0.5023474178403751
Shuttle_withoutdupl_v09.arff	0.9026923076923078
Waveform_withoutdupl_v07.arff	0.6424319473526763
WBC_withoutdupl_v09.arff	0.6187793427230039
WDBC_withoutdupl_v04.arff	0.8952380952380927

Table A.10: Ensemble LOF for k values: 3,4,5,6

k values = 3,4,5,6	LOF Results
Glass_withoutdupl_norm.arff	0.6509485094850931
Lymphography_withoutdupl_norm_idf.arff	0.4753521126760558
Shuttle_withoutdupl_v09.arff	0.7296923076923094
Waveform_withoutdupl_v07.arff	0.5895094226742501
WBC_withoutdupl_v09.arff	0.5483568075117364
WDBC_withoutdupl_v04.arff	0.829131652661062

Table A.11: Ensemble LOF for k values: 4,5,6,7,8

k values = 4,5,6,7,8	LOF Results
Glass_withoutdupl_norm.arff	0.7197831978319776
Lymphography_withoutdupl_norm_idf.arff	0.49061032863849713
Shuttle_withoutdupl_v09.arff	0.8100000000000008
Waveform_withoutdupl_v07.arff	0.6182859706850085
WBC_withoutdupl_v09.arff	0.5830985915492948
WDBC_withoutdupl_v04.arff	0.8775910364145634

Table A.12: Ensemble LOF for k values: 10,11,12

k values = 10,11,12	LOF Results
Glass_withoutdupl_norm.arff	0.8639566395663963
Lymphography_withoutdupl_norm_idf.arff	0.5046948356807507
Shuttle_withoutdupl_v09.arff	0.9526153846153858
Waveform_withoutdupl_v07.arff	0.6758719712832781
WBC_withoutdupl_v09.arff	0.6999999999999991
WDBC_withoutdupl_v04.arff	0.946778711484591

Table A.13: Ensemble LOF for k values: 9,10,11,12

k values = 9,10,11,12	LOF Results
Glass_withoutdupl_norm.arff	0.8563685636856375
Lymphography_withoutdupl_norm_idf.arff	0.5046948356807507
Shuttle_withoutdupl_v09.arff	0.9507692307692319
Waveform_withoutdupl_v07.arff	0.672955429255161
WBC_withoutdupl_v09.arff	0.6840375586854452
WDBC_withoutdupl_v04.arff	0.9414565826330504

Table A.14: Ensemble LOF for k values: 6,7,8,9,10,11,12

k values = 6,7,8,9,10,11,12	LOF Results
Glass_withoutdupl_norm.arff	0.8357723577235779
Lymphography_withoutdupl_norm_idf.arff	0.5023474178403751
Shuttle_withoutdupl_v09.arff	0.9384615384615393
Waveform_withoutdupl_v07.arff	0.6583876757403415
WBC_withoutdupl_v09.arff	0.6586854460093892
WDBC_withoutdupl_v04.arff	0.9257703081232466

Table A.15: LOF for k value: 12

k value = 12	LOF Results
Glass_withoutdupl_norm.arff	0.8682926829268299
Lymphography_withoutdupl_norm_idf.arff	0.5046948356807507
Shuttle_withoutdupl_v09.arff	0.9486153846153856
Waveform_withoutdupl_v07.arff	0.6825545916841234
WBC_withoutdupl_v09.arff	0.7305164319248812
WDBC_withoutdupl_v04.arff	0.9585434173669439

Table A.16: LOF for k value: 11

k value = 11	LOF Results
Glass_withoutdupl_norm.arff	0.8644986449864506
Lymphography_withoutdupl_norm_idf.arff	0.5046948356807507
Shuttle_withoutdupl_v09.arff	0.9551538461538474
Waveform_withoutdupl_v07.arff	0.6731079868381682
WBC_withoutdupl_v09.arff	0.699999999999999
WDBC_withoutdupl_v04.arff	0.9406162464985967

Table A.17: LOF for k value: 10

k value = 10	LOF Results
Glass_withoutdupl_norm.arff	0.8563685636856375
Lymphography_withoutdupl_norm_idf.arff	0.5035211267605628
Shuttle_withoutdupl_v09.arff	0.9533846153846165
Waveform_withoutdupl_v07.arff	0.6699042775949782
WBC_withoutdupl_v09.arff	0.6699530516431919
WDBC_withoutdupl_v04.arff	0.9322128851540589

Table A.18: LOF for k value: 9

k value = 9	LOF Results
Glass_withoutdupl_norm.arff	0.8395663956639573
Lymphography_withoutdupl_norm_idf.arff	0.5023474178403751
Shuttle_withoutdupl_v09.arff	0.9443076923076934
Waveform_withoutdupl_v07.arff	0.6615524977565163
WBC_withoutdupl_v09.arff	0.6436619718309851
WDBC_withoutdupl_v04.arff	0.9022408963585408

Table A.19: LOF for k value: 8

k value = 8	LOF Results
Glass_withoutdupl_norm.arff	0.8092140921409221
Lymphography_withoutdupl_norm_idf.arff	0.5023474178403751
Shuttle_withoutdupl_v09.arff	0.9343076923076932
Waveform_withoutdupl_v07.arff	0.6539066706551063
WBC_withoutdupl_v09.arff	0.5957746478873236
WDBC_withoutdupl_v04.arff	0.8845938375350115

Table A.20: LOF for k value: 7

k value = 7	LOF Results
Glass_withoutdupl_norm.arff	0.7181571815718144
Lymphography_withoutdupl_norm_idf.arff	0.4976525821596239
Shuttle_withoutdupl_v09.arff	0.7352307692307748
Waveform_withoutdupl_v07.arff	0.6343075082261441
WBC_withoutdupl_v09.arff	0.5967136150234738
WDBC_withoutdupl_v04.arff	0.8885154061624625

Table A.21: LOF for k value: 6

k value = 6	LOF Results
Glass_withoutdupl_norm.arff	0.6249322493224937
Lymphography_withoutdupl_norm_idf.arff	0.49178403755868494
Shuttle_withoutdupl_v09.arff	0.6776923076923089
Waveform_withoutdupl_v07.arff	0.6135148070595152
WBC_withoutdupl_v09.arff	0.6131455399061018
WDBC_withoutdupl_v04.arff	0.8705882352941152

Table A.22: LOF for k value: 5

k value = 5	LOF Results
Glass_withoutdupl_norm.arff	0.639024390243901
Lymphography_withoutdupl_norm_idf.arff	0.4812206572769948
Shuttle_withoutdupl_v09.arff	0.6370769230769266
Waveform_withoutdupl_v07.arff	0.597559078671857
WBC_withoutdupl_v09.arff	0.5760563380281676
WDBC_withoutdupl_v04.arff	0.871428571428569

Table A.23: LOF for k value: 4

k value = 4	LOF Results
Glass_withoutdupl_norm.arff	0.6834688346883453
Lymphography_withoutdupl_norm_idf.arff	0.46830985915492906
Shuttle_withoutdupl_v09.arff	0.6403846153846194
Waveform_withoutdupl_v07.arff	0.5766856117259916
WBC_withoutdupl_v09.arff	0.4877934272300473
WDBC_withoutdupl_v04.arff	0.7840336134453757

Table A.24: LOF for k value: 3

k value = 3	LOF Results
Glass_withoutdupl_norm.arff	0.6390243902439021
Lymphography_withoutdupl_norm_idf.arff	0.4483568075117366
Shuttle_withoutdupl_v09.arff	0.70638461538461
Waveform_withoutdupl_v07.arff	0.5576697577026647
WBC_withoutdupl_v09.arff	0.5366197183098592
WDBC_withoutdupl_v04.arff	0.4767507002801105