

# A Cost-Sensitive, Semi-Supervised, and Active Learning Approach for Priority Outlier Investigation

by

Xinran Song

Department of Statistical Science  
Duke University

Date: \_\_\_\_\_

Approved: \_\_\_\_\_

\_\_\_\_\_  
Jian Pei, Supervisor

\_\_\_\_\_  
Simon Mak

\_\_\_\_\_  
Yue Jiang

A thesis submitted in partial fulfillment of the  
requirements for the degree of Master of Science  
in the Department of Statistical Science  
in the Graduate School of  
Duke University

2023

ABSTRACT

A Cost-Sensitive, Semi-Supervised, and Active Learning  
Approach for Priority Outlier Investigation

by

Xinran Song

Department of Statistical Science  
Duke University

Date: \_\_\_\_\_

Approved:

\_\_\_\_\_  
Jian Pei, Supervisor

\_\_\_\_\_  
Simon Mak

\_\_\_\_\_  
Yue Jiang

An abstract of a thesis submitted in partial fulfillment of the  
requirements for the degree of Master of Science  
in the Department of Statistical Science  
in the Graduate School of  
Duke University

2023

Copyright © 2023 by Xinran Song  
All rights reserved

# Abstract

This master's thesis presents a novel approach to address the problem of balancing the cost of investigating suspected cases with the potential gain of detecting an outlier, particularly in the context of fraud detection. The proposed approach is a cost-sensitive, semi-supervised, and active-learning priority outlier investigation model, which aims to identify the top- $k$  unlabeled cases that maximize the overall expected gain.

The proposed approach is developed based on a comprehensive review of related work in cost-sensitive and active learning in outlier detection. We formulate the problem as a maximization function that utilizes kernel density estimation to calculate the probability of unlabeled cases. To improve the model's accuracy and efficiency, we employ graph representation, which takes into account the similarities and relationships among cases. Furthermore, we utilize the neighborhood of cases for efficient kernel density estimation. The performance of the proposed approach is evaluated using both synthetic data and a real-world credit card fraud detection dataset.

The contributions of this thesis include the development of effective and efficient outlier investigation strategies with practical applications in various domains, particularly in the context of fraud detection. The proposed approach offers a promising solution to the challenge of balancing the cost of investigating suspected cases with the potential gain of detecting an outlier.

# Contents

<b>Abstract</b>	<b>iv</b>
<b>List of Tables</b>	<b>vii</b>
<b>List of Figures</b>	<b>viii</b>
<b>Acknowledgements</b>	<b>ix</b>
<b>1 Introduction</b>	<b>1</b>
<b>2 Related Works</b>	<b>4</b>
<b>3 Model Formulation</b>	<b>10</b>
3.1 Model Outline . . . . .	10
3.2 Kernel Density Estimation (KDE) . . . . .	11
3.3 Model Characteristics . . . . .	12
3.4 Technical Challenges . . . . .	13
<b>4 Algorithms</b>	<b>14</b>
4.1 Baseline Algorithm . . . . .	14
4.2 Neighborhood-based Algorithm . . . . .	16
4.2.1 Graph Representation . . . . .	18
4.2.2 Radius Determination . . . . .	19
<b>5 Application in Credit Card Fraud Detection</b>	<b>22</b>
5.1 Synthetic Data . . . . .	22
5.2 Case Study: Credit Card Fraud Detection . . . . .	25
5.2.1 Dataset Description . . . . .	25
5.2.2 Results . . . . .	25

<b>6 Conclusion</b>	<b>28</b>
<b>A Appendix</b>	<b>30</b>
<b>Bibliography</b>	<b>33</b>

# List of Tables

5.1	Cumulative gain with different methods . . . . .	24
5.2	Results of anomaly detection models on credit card dataset. . . . .	26
5.3	Cumulative gain using the credit card transaction data. . . . .	27
A.1	Local Outlier Factor outliers (ranked) . . . . .	30
A.2	One-Class SVM outliers (ranked) . . . . .	31
A.3	Isolation Forest outliers (ranked) . . . . .	31
A.4	Results of Neighborhood-based Algorithm . . . . .	32

# List of Figures

2.1	Example of a kd tree. (a) The kd tree decomposition for a region containing seven data points. (b) The kd tree for the region of (a). . .	8
4.1	Graph representation in a two dimensional space: blue nodes are labeled cases, black ones are unlabeled. . . . .	18
5.1	Synthetic data and outlier detection results using LOF, SVM, Isolation Forest. . . . .	23
5.2	Confusion matrices for LOF, SVM, and Isolation Forest models. . . .	26

# Acknowledgements

I would like to first express my sincere gratitude to my advisor, Professor Jian Pei, for his invaluable guidance and support throughout my master's thesis journey. Professor Pei's deep insights, constant encouragement, and patience have been instrumental in shaping my research work and enhancing my academic skills. His clear and constructive feedback on drafts of my thesis was incredibly helpful, and he was always available to discuss difficult concepts or provide guidance on my research.

I would like to thank Professor Simon Mak and Professor Yue Jiang for agreeing to serve as committee members for my master's thesis. I appreciate their willingness to dedicate their time and expertise to review and evaluate my research work, and for providing valuable feedback and suggestions to help me improve the quality of my work. Their support and guidance were integral to the successful completion of my thesis, and I am grateful for their contributions to my academic journey.

I am grateful to the faculty and staff of the Department of Statistical Science for their continuous support and assistance during my studies. I would like to extend my thanks to my classmates and friends for their encouragement and inspiration.

Lastly, I am deeply grateful to my family for their unwavering love, encouragement, and support throughout my academic journey. Their support has been immeasurable, providing me with a stable foundation from which to pursue my goals.

# Chapter 1

## Introduction

Outlier detection is a critical problem in many domains, and has been extensively studied in machine learning, data mining, and information theory. However, balancing the cost of investigating a suspected outlier with the potential gain of detecting an actual outlier is a major challenge, especially in the context of fraud detection where the financial implications can be significant. Outlier investigation is an important aspect of many machine learning applications, as it involves identifying and investigating instances in the dataset that deviate significantly from the norm.

To address the challenges of investigating outliers in datasets, it is essential to create a cost-sensitive model. While outlier investigation has the potential to yield significant benefits, it can also be a costly endeavor. Investigating each outlier requires a significant investment of time, effort, and resources, particularly in large datasets. Such investigations may sometimes result in false positives, leading to a waste of valuable resources. Labeling cost is also a critical factor to consider, as labeling data points - such as identifying instances of healthcare insurance fraud or fraudulent transactions - is frequently a manual process that requires specialized domain knowledge and time. Additionally, the cost plays a significant role in planning the outlier investigation strategy because of class imbalance, which is natural and common of outlier detection datasets. This will be discussed in detail in Chapter 2.

One practical application of outlier detection is fraud detection, where the financial implications of false positives or false negatives can be significant. Investigating each suspected case requires a significant investment of time, effort, and resources, particularly in large datasets. To address this challenge, it is essential to create a cost-sensitive model that can identify a ranked list of outliers with the highest expected overall gain while minimizing

investigation costs.

To demonstrate the importance of investigating outliers in a cost-sensitive and sequential manner, consider the following example: two cases, A and B, with amounts of \$10 and \$60, respectively, are initially assigned scores of 0.8 and 0.6 by the outlier detection algorithm. Investigating case A first increases the probability of case B being an outlier from 0.6 to 0.61, while investigating case B first increases the probability of case A being an outlier from 0.8 to 0.9. Despite the algorithm ranking case A higher than B, when considering expected gains, the order is flipped, and B becomes the priority case to investigate first. Therefore, deciding which case to investigate initially depends on weighing the potential gain and cost associated with each transaction, as well as the impact of investigating one transaction on the outlier scores of other transactions.

To tackle the challenge of balancing the cost of investigating a suspected case with the potential gain of detecting an outlier, this master's thesis proposes a novel approach - a cost-sensitive, semi-supervised, and active-learning priority outlier investigation model. The main objective of this model is to identify a ranked list of outliers that maximizes the overall expected gain while minimizing investigation costs. Specifically, the model achieves this by adaptively ranking top- $k$  unlabeled cases with the highest expected overall gain. This is done by updating the estimated probability of other cases each time a case is identified and investigated, according to the label of the investigated case, using active learning. Existing methods for estimating outlier probability include Kernel density estimation (KDE), which is a non-parametric technique for estimating the probability density function of a random variable. In our model, we use KDE to estimate the probability density of the normal instances and the probability density of the outliers. By incorporating semi-supervised and active learning techniques, the model can efficiently and effectively detect outliers with minimal investigation costs.

This thesis first provides a comprehensive review of related work in cost-sensitive and active learning in outlier detection, as well as graph representation in outlier detection. The model formulation is then described in detail, with a focus on constructing a kernel

density estimation-based approach. The algorithm employed in the model is also presented, along with an explanation of how the graph-based algorithm enhances the efficiency of the baseline algorithm through the use of graph representation and the neighborhood of cases for kernel density estimation. Finally, the performance of the model is evaluated using both synthetic data and a real-world credit card fraud detection dataset.

The contributions of this thesis lie in the development of effective and efficient outlier investigation strategies with practical applications in various domains, particularly in the context of fraud detection. The proposed approach can help to balance the cost of investigation with the potential gain of detecting an actual outlier, thereby providing a valuable tool for machine learning applications involving outlier detection or anomaly detection.

# Chapter 2

## Related Works

Outlier detection has been extensively researched in various domains, including computer vision, data mining, and anomaly detection. While many existing methods aim to identify outliers from a given dataset, few studies have focused on the challenge of actively selecting the next object to label for efficient outlier detection. Additionally, it is crucial to consider the labeling cost and potential gain of labeling each sample. For example, investigating outliers with low gain may be less important than investigating normal points with high gain. In this section, we will review the literature on active learning and outlier detection, including the use of kernel density estimation for identifying outliers. We will place particular emphasis on methods that incorporate cost and gain considerations and examine recent advances in graph representation-based outlier detection.

### **Cost-sensitive Learning for Outlier Investigation**

Several studies have emphasized the importance of constructing a cost-sensitive approach for outlier investigation, particularly due to class imbalance. Ling et al. [LS08] define cost-sensitive learning as a type of learning in data mining that takes misclassification costs and other types of cost into consideration to minimize total cost. Elkan and Zadrozny [Elk01, ZE01] publish the theory of cost-sensitive learning, which describes how the misclassification cost plays an essential role in various cost-sensitive learning algorithms. Class imbalance often affects the performance of cost-sensitive classifiers, and the natural class distribution is generally favored unless misclassification costs are significantly unequal [LZ06].

To address the issue of cost and gain in outlier investigation, several studies have been conducted. Active learning-based outlier detection studies have investigated the effect of labeling cost on performance. Krishnamurthy and Kardel [KAH<sup>+</sup>17] design an active learning

algorithm for cost-sensitive multiclass classification that predicts the smallest cost. Chandola et al. [CBK09] show that the gain of labeling also affects the performance of outlier detection methods. To handle both concept drift and class imbalance issues, Zhang et al. [NZL<sup>+</sup>19] propose an incremental ensemble learning method that includes an imbalance-reversed bagging method to improve true positive rates while maintaining a low false positive rate, and a dynamic cost-sensitive weighting scheme for component classifiers.

Considering both labeling cost and gain is crucial for efficient and effective outlier investigation. Incorporating cost-sensitive learning into the selection of the next object to label can significantly reduce the labeling cost and achieve high gain while maintaining high detection accuracy.

## Active Learning for Outlier Detection

Active learning has been extensively studied in the context of outlier detection with one-class classifiers, showing great potential for improvement. In [TEB18], a comprehensive overview of existing methods is provided, along with guidelines for selecting active learning methods for outlier detection with one-class classifiers. This work categorizes various methods, proposes ways to evaluate active learning results, and conducts extensive experiments for a broad variety of scenarios.

Several outlier detection methods using active learning have been proposed based on different scenarios. For example, in [TB19], SubSVDD, a semi-supervised classifier, is proposed, which learns decision boundaries in low-dimensional projections of the data. SubSVDD de-constructs the outlier classification so that users can comprehend and interpret results more easily. For active learning, SubSVDD features a new update mechanism that adjusts decision boundaries based on user feedback, taking into account that outliers may only occur in some of the low-dimensional projections. In [PMVZ18], an active learning method is proposed that can be added to existing deep learning models for unsupervised anomaly detection. A new layer is introduced that can be easily attached to any deep learning model, and results are reported on synthetic and real anomaly detection datasets.

In [PM04], a novel active-learning scenario is introduced where a user wants to identify useful anomalies that are subjectively categorized by the user. A mixture model fit to the data is assumed, and a detailed empirical analysis demonstrates the ability to quickly identify an anomaly set containing a few tens of points in a dataset of hundreds of thousands. In [AZL06], a simple reduction of outlier detection to classification is presented, via a procedure that involves applying classification to a labeled data set containing artificially generated examples that play the role of potential outliers. Once the task has been reduced to classification, a selective sampling mechanism based on active learning is invoked to the reduced classification problem.

These studies demonstrate the effectiveness of active learning in outlier detection and highlight the potential of combining active learning with other outlier detection methods. Our work is inspired by these studies and proposes a new approach to selecting the next object to label for efficient outlier detection by employing kernel density estimation.

## **Kernel Density Estimation**

Kernel density estimation (KDE) is a well-known non-parametric density estimation method that has been widely employed in outlier detection. KDE estimates the probability density function of a random variable by kernel smoothing, which involves convolving the observed data with a smoothing kernel. In [Sco04], Scott gives a comprehensive overview of KDE and its applications, including outlier detection. He discusses the advantages and limitations of KDE compared to other density estimation techniques and provides guidelines for selecting an appropriate bandwidth parameter for KDE. The effectiveness of KDE in outlier investigation has been demonstrated in previous studies. Motivated by this, we propose a novel approach to outlier investigation by combining active learning with KDE.

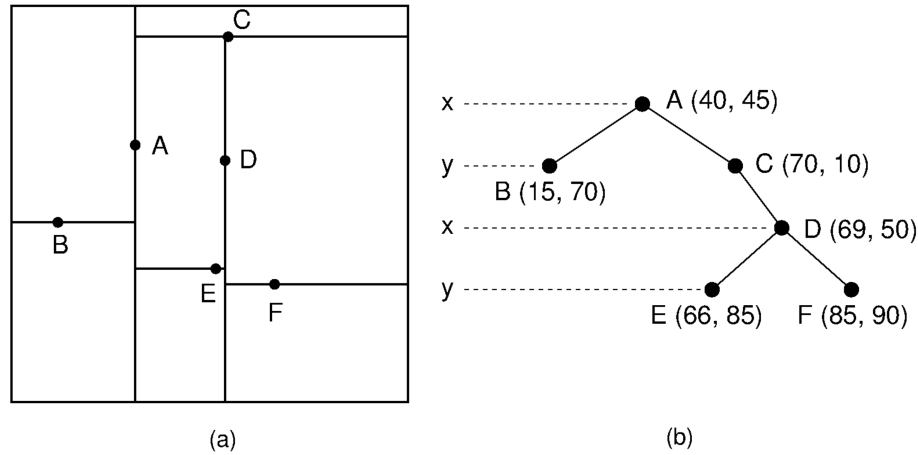
## **Graph Representation for Outlier Investigation**

Recently, graph-based approaches have emerged as a promising method for outlier investigation. In their review paper, Akoglu et al. [ATK15] classified various graph-based anomaly

detection methods based on their techniques and applications. Graph-based approaches are important for several reasons. Firstly, data objects are often inter-dependent, making it necessary to account for related objects when detecting anomalies. Secondly, graphs naturally represent inter-dependencies and facilitate the representation of complex datasets. Thirdly, anomalies can exhibit relational behavior, and finally, graphs are more robust tools for detecting anomalies as fraudsters may not have a global view of the entire network they are operating in.

Several related studies have used graphs for outlier investigation. Hautamaki et al. [HKF04] presented an Outlier Detection using Indegree Number (ODIN) algorithm that utilizes a k-nearest neighbor graph. Chakrabarti [Cha04] proposed a novel way to group nodes using information-theoretic principles to choose both the number of groups and the mapping from nodes to groups. Their algorithm is parameter-free and scales practically linearly with the problem size. Furthermore, they proposed novel algorithms that use this node group structure to find outliers and compute distances between groups. Wang et al. [WLG19] proposed Virtual Outlier Score (VOS), which constructs a similarity graph using the top-k similar neighbors of each object and introduces a virtual node coupled with a collection of virtual edges to generate a k-virtual graph. A tailored Markov random walk process is then performed on the strongly connected virtual graph under the principle that a potential outlier should receive more weight to be visited. After reaching equilibrium, the stationary distribution vector is utilized to deduce the virtual outlier score.

In this study, we have implemented a neighborhood-based algorithm using graph representation, and utilized KD-Tree [FBF77] (short for k-dimensional tree) for determining the neighborhood. KD-Tree is a space-partitioning data structure used for organizing points in a k-dimensional space, making it an essential tool for various applications, such as searches involving a multidimensional search key, including range searches and nearest neighbor searches, as well as creating point clouds. It is worth noting that k-d trees are a special case of binary space partitioning trees, which are commonly used in computer science for efficient spatial data organization and searching. Figure 2.1 is an example of a KD-Tree.



**Figure 2.1:** Example of a kd tree. (a) The kd tree decomposition for a region containing seven data points. (b) The kd tree for the region of (a).

## Outlier Detection Methods

In this thesis, we compared the gain of our outlier priority investigation model with three outlier detection methods, Local Outlier Factor, Support Vector Machines, and Isolation Forest. Local Outlier Factor (LOF)[BKNS00] is a popular algorithm for outlier detection that measures the local density around a data point and compares it to the density of its neighbors. If a point has a much lower density than its neighbors, it is likely to be an outlier. LOF is particularly useful for detecting outliers in high-dimensional data sets where traditional distance-based methods may fail. Support Vector Machines (SVM)[CV95] are another powerful tool for outlier detection. SVMs are a type of machine learning algorithm that can be used for both classification and regression tasks, but they can also be adapted for outlier detection. The basic idea behind SVM-based outlier detection is to train a model on a data set and then identify points that fall far from the decision boundary. SVMs are particularly useful for detecting outliers in complex, nonlinear data sets. Isolation Forest[LTZ08] is a relatively new algorithm for outlier detection that uses a series of randomized decision trees to isolate outliers. The basic idea behind Isolation Forest is to randomly select a feature and then split the data set along a random value of that

feature. The algorithm then repeats this process recursively, creating a tree-like structure that isolates outliers in the leaves of the tree. Isolation Forest is particularly useful for detecting outliers in high-dimensional data sets and can be very fast and efficient.

Our work integrates and extends several areas of research, including cost-sensitive learning, active learning, and outlier detection. We propose a novel method for efficient outlier investigation by combining these approaches, which selects the next object to label based on the expected gain after confirmation. Our approach employs kernel density estimation to estimate the probability of an unlabeled object being an outlier, allowing us to prioritize labeling of objects with higher outlier probability and gain. This method significantly reduces the number of labeled objects required for accurate outlier detection, making our approach both efficient and effective.

# Chapter 3

## Model Formulation

### 3.1 Model Outline

Consider a set of objects  $\mathcal{D} = \{x_1, \dots, x_n\} \subset \mathbb{R}^M$ , where  $\mathbb{R}$  is the set of real numbers. Within  $\mathcal{D}$  we may have a subset  $\mathcal{D}^\ell$  of objects that are known as outliers or inliers. We also call  $\mathcal{D}^\ell$  the set of labeled objects. For each case  $x_i \in \mathcal{D}^\ell$ , the indicator variable  $y_i$  represents if the object is an outlier ( $y_i = 1$  if  $x_i$  is an outlier). In  $\mathcal{D}^\ell$ , the subset containing all the outliers is called  $\mathcal{D}_{\text{out}}^\ell$ , and the subset containing all the inliers is called  $\mathcal{D}_{\text{in}}^\ell$ .

Denote by  $\mathcal{D}^* = \mathcal{D} \setminus \mathcal{D}^\ell$  the set of objects that are not confirmed as outliers or not. We also call  $\mathcal{D}^*$  the set of unlabeled objects. For an object  $x_i \in \mathcal{D}^*$ , we can obtain a confirmation on whether  $x_i$  is an outlier or not by paying a cost  $l(x_i)$ . If an object  $x_i$  is confirmed as an outlier, we can gain  $gain(x_i)$ , such as collecting the fraudulent amount from a fraud.

Before asking for a confirmation on an object  $x_i$ , we can obtain for free an outlier score  $F(x_i) \in [0, 1]$  that indicates the outlyingness of  $x_i$ . The larger the outlier score, the more outlying the object.

Let  $P^{\text{in}}(x_i)$  and  $P^{\text{out}}(x_i)$  be the probability that  $x_i$  is an inlier and an outlier, respectively. That is,  $P^{\text{in}}(x_i) + P^{\text{out}}(x_i) = 1$ .

Based on the above setting, for any object  $x_i \in \mathcal{D}^*$ , if we request the confirmation of whether  $x_i$  is an outlier, the expected gain is

$$g(x_i) = P^{\text{out}}(x_i | \mathcal{D}_{\text{in}}^\ell) \sum_{x_j \in \mathcal{D}^*} [(P^{\text{out}}(x_j | \mathcal{D}_{\text{out}}^\ell \cup \{x_i\}) - P^{\text{out}}(x_j | \mathcal{D}_{\text{out}}^\ell)) \cdot gain(x_j)] - l(x_i) \quad (3.1)$$

where  $P^{\text{out}}(x_j | \mathcal{D}_{\text{in}}^\ell) = 1 - P^{\text{in}}(x_j | \mathcal{D}_{\text{in}}^\ell)$  is the probability of the point  $x_j$  being an outlier given the known inliers from the labeled set  $\mathcal{D}_{\text{in}}^\ell$ .  $P^{\text{out}}(x_j | \mathcal{D}_{\text{out}}^\ell)$  is the probability of the point

$x_j$  being an outlier given the known outliers from the labeled set  $\mathcal{D}_{\text{out}}^\ell$ ,  $P^{\text{out}}(x_j|\mathcal{D}_{\text{out}}^\ell \cup \{x_i\})$  the probability of  $x_j$  after labeling  $x_i$  as an outlier given  $\mathcal{D}_{\text{out}}^\ell$ .

Our goal is to pick the next object  $x_i$  that can maximize the expected gain after confirmation. That is, we solve the following maximization problem.

$$\max_{x_i \in \mathcal{D}^*} \{g(x_i)\} \quad (3.2)$$

## 3.2 Kernel Density Estimation (KDE)

In our model, one issue is how we can estimate  $P^{\text{out}}(x_i)$  and the conditional probabilities.

We employ kernel density estimation, a non-parametric density estimation method [Par62][Ros56].

The kernel density estimator is defined as

$$\hat{P}(x) = \frac{1}{nh} \sum_{i=1}^n K\left(\frac{X_i - x}{h}\right), \quad (3.3)$$

where  $K(x)$  is a kernel function that is generally a smooth, symmetric function, such as a Gaussian, and  $h > 0$  is the smoothing bandwidth that controls the amount of smoothing.

Using Gaussian kernel  $K(x) = \frac{1}{\sqrt{2\pi}} e^{-\frac{x^2}{2}}$ ,  $P^{\text{out}}(x_j|\mathcal{D}^\ell)$  is thus estimated to be

$$\hat{P}^{\text{out}}(x_j|\mathcal{D}_{\text{out}}^\ell) = \frac{1}{\sqrt{2\pi}Nh} \sum_{i=1}^N \exp\left\{-\frac{\|x_j - y_i^{\text{out}}\|^2}{2h^2}\right\},$$

where  $N$  is the total number of outliers in the labeled set  $\mathcal{D}^\ell$ , and  $y_i^{\text{out}}$  are the labeled outliers in  $\mathcal{D}^\ell$ .

Similarly,  $P^{\text{in}}(x_i)$  is estimated to be

$$\hat{P}^{\text{in}}(x_j|\mathcal{D}_{\text{in}}^\ell) = \frac{1}{\sqrt{2\pi}Mh} \sum_{i=1}^M \exp\left\{-\frac{\|x_j - y_i^{\text{in}}\|^2}{2h^2}\right\},$$

where  $M$  is the total number of inliers in the labeled set  $\mathcal{D}^\ell$ , and  $y_i^{\text{in}}$  are the labeled inliers in  $\mathcal{D}^\ell$ .

### 3.3 Model Characteristics

This priority outlier investigation model is a cost-sensitive, semi-supervised and active learning approach in the following ways.

- *Cost-sensitive.* Taking into account of the investigation cost and gain, this model could maximize the expected gain of investigating outliers. This priority outlier investigation strategy is sensitive to the cost and gain of each query. Intuitively, if an inlier has a relatively large gain, it will be prioritized for investigation. On the other hand, an outlier with small gain and high labeling cost will be likely to rank low in our strategy. Our model thus could maximize benefit from a global perspective by scheduling queries in a cost-sensitive strategy.
- *Semi-supervised.* In many situations where the labeling process is expensive and time-consuming, obtaining large number of labeled cases is challenging and unrealistic. In our model, we construct a semi-supervised learning model to avoid this problem. We use a small number of labeled cases and a large number of unlabeled cases,  $|\mathcal{D}^*| \gg |\mathcal{D}^\ell|$ . We initially use the labeled cases for predicting the outlying probability of unlabeled cases. Then, we include investigated unlabeled cases and update our model case by case for subsequent learning.
- *Active learning.* Active learning is a special case of machine learning in which a learning algorithm can interactively query a user (or some other information source) to label new data points with the desired outputs [Set09]. In the proposed model, each time a potential outlier is identified for investigation, it is investigated and labeled. The newly-labeled case will then be included in our model for learning. When a case is labeled and included, the probability of the rest unlabeled cases being inlier or outlier will be updated and the prediction process improves through active learning.

## 3.4 Technical Challenges

Our model faces several technical challenges that need to be addressed to ensure optimal performance. The first challenge concerns the kernel density estimation method upon which our model relies. Specifically, we need to pay close attention to bandwidth selection and computational cost. KDE is highly sensitive to the bandwidth parameter, denoted as  $h$ . If  $h$  is too large, the probability density function (PDF) will be over-smoothed, and outliers will be less visible. Conversely, if  $h$  is too small, the PDF will be excessively noisy, and outliers may be falsely identified.

The second challenge we face is computational cost, which can become a significant issue when dealing with large and high-dimensional datasets. The calculation of pairwise Euclidean distances required to estimate the density of cases is computationally expensive, and thus we need to explore alternative solutions. One such solution is to estimate the local density of points using an error bound, which is discussed in detail in Chapter 4.

Another challenge is determining the radius of the neighborhood for each unlabeled case. Determining an appropriate neighborhood radius is crucial for ensuring that our model captures the underlying patterns in the data accurately. This issue is addressed in Chapter 4.2.2, where we discuss how to determine the radius according to distance and gain of cases.

Our model must also address the challenge of imbalanced datasets, which is a prevalent issue in many real-world applications. Imbalanced datasets can cause biased density estimates, leading to a skewed model output. To overcome this challenge, we utilize a cost-sensitive approach, where the cost function is an essential component. Determining an appropriate cost function is a non-trivial problem, and it requires careful consideration of the dataset characteristics.

# Chapter 4

## Algorithms

### 4.1 Baseline Algorithm

Based on our model, we first design a brute-force baseline algorithm for the priority outlier investigation. In the initialization part, we calculate the estimates of outlier probability and inlier probability using KDE. Then, we calculate expected gain of each unlabeled case by summing up the marginal gain of all the unlabeled cases after which is investigated. Specifically, suppose  $\mathcal{D}^\ell$  has  $M$  inliers and  $N$  outliers.  $\mathcal{D}^*$  has  $p$  points. The difference in probability of being an outlier before and after  $x_i$  is investigated and labeled is:

$$\begin{aligned}\hat{P}^{out}(x_j | \mathcal{D}^\ell \cup \{x_i\}) - \hat{P}^{out}(x_j | \mathcal{D}^\ell) &= \frac{1}{(N+1)h} \left( \sum_{k=1}^N K\left(\frac{x_k - x_j}{h}\right) + K\left(\frac{x_i - x_j}{h}\right) \right) \\ &\quad - \frac{1}{Nh} \sum_{k=1}^N K\left(\frac{x_k - x_j}{h}\right) \\ &= \frac{1}{N+1} \left( \frac{1}{h} K\left(\frac{x_j - x_i}{h}\right) - \hat{P}^{out}(x_j | \mathcal{D}^\ell) \right).\end{aligned}\tag{4.1}$$

After the overall expected gain is calculated for each point, we can find the outlier that maximizes expected gain to investigate and label it. Then we can update the labeled set and the estimated probability accordingly. The updated probability after  $x^{(m)}$  is labeled is:

$$\hat{P}_{new}^{out}(x_i | \mathcal{D}^\ell) \leftarrow \frac{N}{N+1} \hat{P}^{out}(x_i | \mathcal{D}^\ell) + \frac{1}{N+1} K\left(\frac{x_i - x^{(m)}}{h}\right).$$

We can repeat this process until a top- $k$  list of priority outliers are investigated. The detailed algorithm can be seen in Algorithm 1.

This brute-force baseline algorithm is a simple and straightforward realization of our model. It has a clear and interpretable logic and can be used for understanding the relationships of the variables in our model. However, its time complexity is  $\mathcal{O}(p^2)$ , where  $p$  is the

---

**Algorithm 1:** Baseline Algorithm

---

**Input:**  $\mathcal{D}^\ell$ ,  $\mathcal{D}^*$ ,  $l(x)$ ,  $\text{gain}(x)$ ,  $k$ ,  $\sigma$

**Output:**  $\{x^{(1)}, \dots, x^{(k)}\}$

```
/* Initialization. */
1 for i = 1 to p do
2   calculate  $\hat{P}^{in}(x_i | \mathcal{D}_{in}^\ell) = \frac{1}{M} \sum_{j=1}^M \delta_\sigma(y_j - x_i)$  // y are the inliers in
   labeled set
3   calculate  $\hat{P}^{out}(x_i | \mathcal{D}_{out}^\ell) = \frac{1}{N} \sum_{j=1}^N \delta_\sigma(y_j - x_i)$  // y are the outliers
   in labeled set
/* Calculate expected gain. */
4 for i = 1 to p do
5   for j = 1 to p do
6     calculate  $\delta_\sigma(x_j - x_i)$  // calculate pairwise kernel, x are
     cases in the unlabeled set
7   calculate  $A(x_i) = \sum_{j=1}^p (\delta_\sigma(x_j - x_i) \cdot \text{gain}(x_j))$ 
8   calculate  $B_i = \sum_{j=1}^p (\hat{P}^{out}(x_j | \mathcal{D}^\ell) \cdot \text{gain}(x_j))$ 
9   calculate  $S(x_i) = A(x_i) - B_i$ 
10  calculate  $g(x_i) = S(x_i) \cdot (1 - \hat{P}^{in}(x_i)) - l(x_i)$ 
/* Find the next outlier and update the overall process. */
11 for m = 1 to k do
12   $x^{(m)} = \arg \max_{x_i} g(x_i)$ 
13  for j = 1 to p do
14    calculate  $\delta_\sigma(x_j - x^{(m)})$ 
15   $B_i \leftarrow \frac{N}{N+1} B_i + \frac{1}{N+1} \sum_{j=1, j \neq i}^p \delta_\sigma(x_j - x^{(m)}) \cdot \text{gain}(x_j)$ 
16   $S(x_i) \leftarrow A(x_i) - B_i$ 
17   $N \leftarrow N + 1$ ,  $\mathcal{D}^\ell \leftarrow \mathcal{D}^\ell \cup \{x^{(m)}\}$ ,  $\mathcal{D}^* \leftarrow \mathcal{D}^* \setminus \{x^{(m)}\}$ 
```

---

number of unlabeled cases, which can be inefficient and time-consuming for large datasets with a large number of unlabeled cases. To improve the efficiency of our algorithm, we have designed a neighborhood-based approach that reduces the time complexity to  $\mathcal{O}(c \cdot p)$ , where  $c$  is the number of cases in the neighborhood of each unlabeled case. This approach is discussed in detail in the following section.

## 4.2 Neighborhood-based Algorithm

To improve our baseline algorithm, we use a local kernel density estimation which estimates the probability density function at a specific point in the data instead of a kernel density estimation in our algorithm. To achieve this, we form a graph for our model and use an error bound to determine a neighborhood for every unlabeled case.

There are three advantages to this approach. First, by determining a neighborhood for each case, we can obtain a better representation of local behavior. Since local KDE only considers the data in a neighborhood, it can capture more subtle variations and offer better adaptation to local data density, especially for data that exhibits strong heterogeneity. Second, local KDE is more computationally efficient for a large dataset. This is because it is computationally expensive to calculate the pairwise Euclidean distances to estimate density and update density across the entire data set in Algorithm 1. By reducing the size of the computational problem, local KDE can be more computationally efficient and faster to implement since it only requires computation within a small neighborhood of each point. Third, local KDE allows for more flexibility for our model in choosing the error bound for neighborhood of the cases. The error bound of the neighborhood is determined by the distance and gain of specific cases, allowing for a more precise estimation of the probability density function.

---

**Algorithm 2:** Neighborhood-based Algorithm

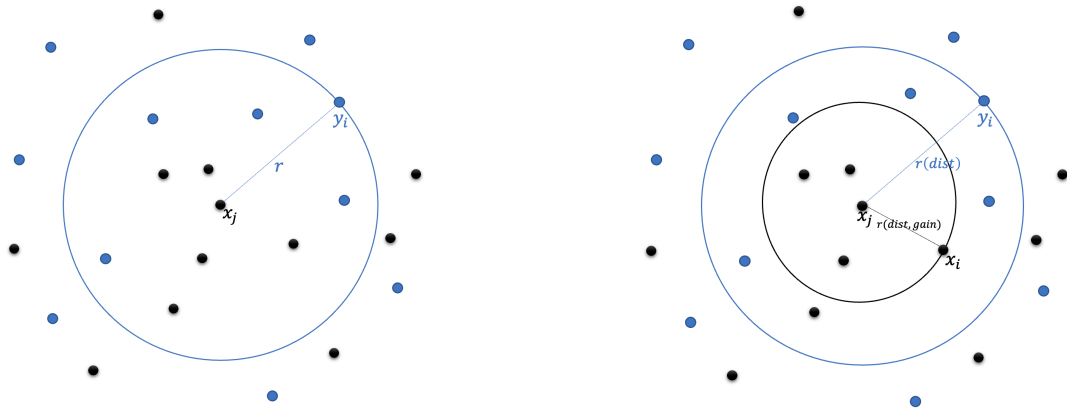
---

**Input:**  $\mathcal{D}^\ell$ ,  $\mathcal{D}^*$ ,  $l(x)$ ,  $\text{gain}(x)$ ,  $k$ ,  $\sigma$

**Output:**  $\{x^{(1)}, \dots, x^{(k)}\}$

```
/* Initialization. */
1 for  $i = 1$  to  $p$  do
2   calculate  $\hat{P}^{in}(x_i | \mathcal{D}_{in}^\ell) = \frac{1}{M} \sum_{j=1}^M \delta_\sigma(y_j - x_i)$  //  $y$  are the inliers in
   labeled set
3   calculate  $\hat{P}^{out}(x_i | \mathcal{D}_{out}^\ell) = \frac{1}{N} \sum_{j=1}^N \delta_\sigma(y_j - x_i)$  //  $y$  are the outliers
   in labeled set
/* Calculate expected gain. */
4 for  $i = 1$  to  $p$  do
5   for  $j = 1$  to  $N_{x_i}$  do
6     calculate  $\delta_\sigma(x_j - x_i)$  // calculate kernel of neighborhood to  $x_i$ 
7     calculate  $A(x_i) = \sum_{j=1}^p (\delta_\sigma(x_j - x_i) \cdot \text{gain}(x_j))$ 
8     calculate  $B_i = \sum_{j=1}^p (\hat{P}^{out}(x_j | \mathcal{D}^\ell) \cdot \text{gain}(x_j))$ 
9     calculate  $S(x_i) = A(x_i) - B_i$ 
10    calculate  $g(x_i) = S(x_i) \cdot (1 - \hat{P}^{in}(x_i)) - l(x_i)$ 
/* Find the next outlier and update its neighborhood. */
11 for  $m = 1$  to  $k$  do
12    $x^{(m)} = \arg \max_{x_i} g(x_i)$ 
13   for  $j = 1$  to  $N_{x^{(m)}}$  do
14     calculate  $\delta_\sigma(x_j - x^{(m)})$ 
15      $B_i \leftarrow \frac{N_{x^{(m)}}}{N_{x^{(m)}}+1} B_i + \frac{1}{N_{x^{(m)}}+1} \sum_{j=1, j \neq i}^p \delta_\sigma(x_j - x^{(m)}) \cdot \text{gain}(x_j)$ 
16      $S(x_i) \leftarrow A(x_i) - B_i$ 
17      $N \leftarrow N + 1$ ,  $\mathcal{D}^\ell \leftarrow \mathcal{D}^\ell \cup \{x^{(m)}\}$ ,  $\mathcal{D}^* \leftarrow \mathcal{D}^* \setminus \{x^{(m)}\}$ 
```

---



**Figure 4.1:** Graph representation in a two dimensional space: blue nodes are labeled cases, black ones are unlabeled.

## 4.2.1 Graph Representation

As shown in Figure 4.1, we can represent our algorithm by using a graph, where the nodes of the graph correspond to the data points and the edges represent the local neighborhood relationships between the points. The graph representation allows us to visualize the updating process and better understand how the error bound affects the estimated probability density function.

First, in the initialization step, we construct a graph which represents the relationships of cases in  $D^*$  and  $D^\ell$  as shown in the left panel of Figure 4.1. The blue nodes represent labeled cases in  $D^\ell$ , and black nodes represent unlabeled cases in  $D^*$ . After determining the radius  $r$ , which is discussed in detail in the following section, we calculate  $\hat{P}^{in}(x_i|\mathcal{D}_{in}^\ell)$  and  $\hat{P}^{out}(x_i|\mathcal{D}_{out}^\ell)$  by summing the kernel functions evaluated at the distances between the point and its neighbors within radius  $r$ .

In implementation of the initialization step, function ‘build\_graph’ is designed to build a graph structure based on the input unlabeled data. The function takes four arguments: ‘unlabeled\_dat’, ‘gain’, ‘radius’, and ‘min\_gain’. ‘unlabeled\_dat’ represents the input dataset, ‘gain’ is the column name that represents the gain of each data point, ‘radius’ is the radius for the neighbor search, and ‘min\_gain’ is the minimum gain required for a point to be considered as a neighbor. The function builds a KD-Tree [FBF77] based on the data to

optimize the neighbor search. For each data point, the function queries the KD-Tree to find its neighbors within the given radius. It then filters out neighbors with small gain and stores the remaining neighbors as a list for that data point in the graph dictionary. The graph dictionary contains two keys for each data point: ‘num\_neighbors’ and ‘neighbors’, where ‘num\_neighbors’ is the number of neighbors for that data point, and neighbors is a list of indices representing the indices of the neighboring data points. The function returns the graph dictionary as the output.

Then, we construct a graph that represents the relationships among all the cases in the unlabeled set, which are represented by black nodes as shown in the right panel of Figure 4.1. For each unlabeled case  $x_j$ , there is a corresponding neighborhood  $\mathcal{N}_{x_j}$  which includes  $m_{x_j}$  unlabeled cases that has relatively large gain and small distances to  $x_j$ . Thus, for  $x_j$ , the gain in our model formula 3.1 is

$$\begin{aligned} g(x_j) &= \hat{P}^{in}(x_j | \mathcal{D}_{in}^\ell) \sum_{x_i \in \mathcal{N}_{x_j}} [(\hat{P}^{out}(x_i | \mathcal{D}_{out}^\ell \cup \{x_j\}) - \hat{P}^{out}(x_i | \mathcal{D}_{out}^\ell)) \cdot gain(x_i)] - l(x_j) \\ &= \hat{P}^{in}(x_j | \mathcal{D}_{in}^\ell) \sum_{x_i \in \mathcal{N}_{x_j}} \left[ \frac{1}{m_{x_j} + 1} \left( \frac{1}{h} K\left(\frac{x_j - x_i}{h}\right) - \hat{P}^{out}(x_i | \mathcal{D}_{out}^\ell) \right) \cdot gain(x_i) \right] - l(x_j). \end{aligned} \tag{4.2}$$

In the updating process, the updated probability after  $x^{(m)}$  is labeled is:

$$\hat{P}_{new}^{out}(x_i | \mathcal{D}^\ell) \leftarrow \frac{N_{x^{(m)}}}{N_{x^{(m)}} + 1} \hat{P}^{out}(x_i | \mathcal{D}^\ell) + \frac{1}{N_{x^{(m)}} + 1} K\left(\frac{x_i - x^{(m)}}{h}\right).$$

## 4.2.2 Radius Determination

### Radius Based Only On Distance

We use an error bound  $\varepsilon$  to discard the cases that are far away or with relatively small gain when we calculate the estimated probability.

**Theorem 4.2.1.** *When the radius  $r$  satisfies the condition that*

$$r \geq \sqrt{-2h^2 \log \frac{\varepsilon N h}{n^2}},$$

*the outlier probability can be estimated by the cases in the neighborhood determined by  $r$ .*

*Proof.* As discussed in previous chapter, the formula for calculating outlier density in the initialization step can be rewritten as

$$\begin{aligned}\hat{P}^{\text{out}}(x_j | \mathcal{D}_{\text{out}}^\ell) &= \frac{1}{Nh} \sum_{i=1}^N K\left(\frac{x_j - y_i^{\text{out}}}{h}\right) \\ &= \frac{1}{Nh} \left( \sum_{y_i^{\text{out}} \in N(x_j, r)} K\left(\frac{x_j - y_i^{\text{out}}}{h}\right) + \sum_{y_i^{\text{out}} \notin N(x_j, r)} K\left(\frac{x_j - y_i^{\text{out}}}{h}\right) \right).\end{aligned}\tag{4.3}$$

If  $r \geq \sqrt{-2h^2 \log \frac{\varepsilon Nh}{n^2}}$ , we have

$$\begin{aligned}\Delta &= \hat{P}^{\text{out}}(x_j | \mathcal{D}_{\text{out}}^\ell) - \hat{P}^{\text{out}'}(x_j | \mathcal{D}_{\text{out}}^\ell) \\ &= \frac{n}{Nh} \sum_{y_i^{\text{out}} \notin N(x_j, r)} K\left(\frac{x_j - y_i^{\text{out}}}{h}\right) \\ &= \frac{n}{Nh} \sum_{y_i^{\text{out}} \notin N(x_j, r)} \exp\left\{-\frac{\|x_j - y_i^{\text{out}}\|^2}{2h^2}\right\} \\ &\leq \frac{n}{Nh} \cdot n \cdot e^{-\frac{r^2}{2h^2}} \\ &\leq \varepsilon.\end{aligned}\tag{4.4}$$

Thus, we can use  $\hat{P}^{\text{out}'}(x_j | \mathcal{D}_{\text{out}}^\ell) = \frac{1}{Nh} \left( \sum_{y_i^{\text{out}} \in N(x_j, r)} K\left(\frac{x_j - y_i^{\text{out}}}{h}\right) \right)$  as an estimate of  $\hat{P}^{\text{out}}(x_j | \mathcal{D}_{\text{out}}^\ell)$ .

□

## Radius Based on Distance and Gain

When considering the relationships among unlabeled cases, specifically, calculating and updating the density of an unlabeled case by its surrounding unlabeled cases, we wish to exclude points which are far away or have little gain. Thus, the radius of this neighborhood is determined by the distance and gain.

**Theorem 4.2.2.** *When the radius  $r$  satisfies the condition that*

$$r \geq \sqrt{-2h^2 \log \frac{\varepsilon Nh}{n^2 \cdot \text{gain}(x_j)}},$$

the marginal gain of  $x_j$  can be estimated by the cases in the neighborhood determined by  $r$  with error  $\leq \varepsilon$ .

*Proof.* The marginal gain can be rewritten as

$$\begin{aligned} \text{MarginalGain} &= (\hat{P}^{\text{out}}(x_i | \mathcal{D}_{\text{out}}^\ell \cup \{x_j\}) - \hat{P}^{\text{out}}(x_i | \mathcal{D}_{\text{out}}^\ell)) \cdot \text{gain}(x_i) \\ &= \frac{1}{N+1} \left( \frac{1}{h} K\left(\frac{x_j - x_i}{h}\right) - \hat{P}^{\text{out}}(x_j | \mathcal{D}^\ell) \right) \cdot \text{gain}(x_j). \end{aligned} \quad (4.5)$$

The part we wish to estimate by the neighborhood is  $\hat{P}^{\text{out}}(x_j | \mathcal{D}^\ell) \cdot \text{gain}(x_j)$ . If  $r \geq \sqrt{-2h^2 \log \frac{\varepsilon N h}{n^2 \cdot \text{gain}(x_j)}}$ , similar to before, we have

$$\begin{aligned} \Delta &= (\hat{P}^{\text{out}}(x_j | \mathcal{D}_{\text{out}}^\ell) - \hat{P}^{\text{out}'}(x_j | \mathcal{D}_{\text{out}}^\ell)) \cdot \text{gain}(x_j) \\ &= \left( \frac{n}{Nh} \sum_{y_i^{\text{out}} \notin N(x_j, r)} K\left(\frac{x_j - y_i^{\text{out}}}{h}\right) \right) \cdot \text{gain}(x_j) \\ &= \frac{n}{Nh} \sum_{y_i^{\text{out}} \notin N(x_j, r)} \exp\left\{-\frac{\|x_j - y_i^{\text{out}}\|^2}{2h^2}\right\} \cdot \text{gain}(x_j) \\ &\leq \frac{n}{Nh} \cdot n \cdot e^{-\frac{r^2}{2h^2}} \cdot \text{gain}(x_j) \\ &\leq \varepsilon. \end{aligned} \quad (4.6)$$

□

# Chapter 5

## Application in Credit Card Fraud Detection

### 5.1 Synthetic Data

In order to evaluate the performance of our outlier detection models, we generated a synthetic dataset with both normal and outlier observations. We used the NumPy library [HMvdW20] to generate 900 observations from a normal distribution with mean  $\boldsymbol{\mu} = [0, 0]$  and standard deviation  $\boldsymbol{\Sigma} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$ . Additionally, we generated 100 outlier observations

from a normal distribution with mean  $\boldsymbol{\mu} = [4, 4]$  and standard deviation  $\boldsymbol{\Sigma} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$ .

$$normal \sim \mathcal{N}\left(\begin{bmatrix} 0 & 0 \end{bmatrix}, \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}\right), outliers \sim \mathcal{N}\left(\begin{bmatrix} 4 & 4 \end{bmatrix}, \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}\right).$$

We concatenated the normal and outlier observations to create a dataset of 1000 observations.

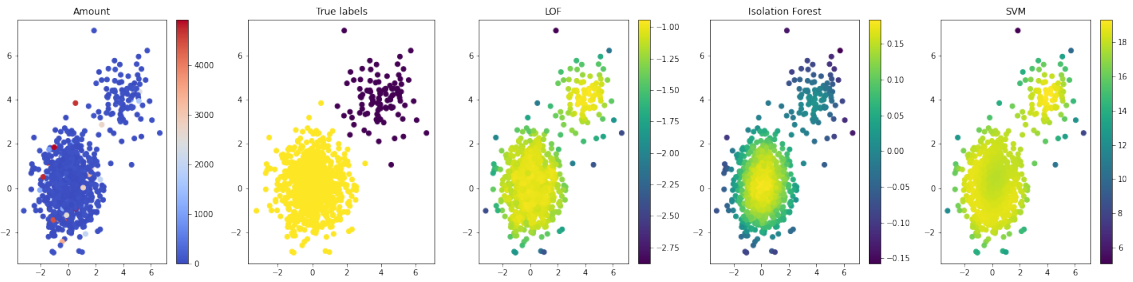
To further add variability to the data and simulate real-world scenarios, an “Amount” column is added to the feature matrix X. The “Amount” column is created by randomly choosing values from a set of 950 numbers that are linearly spaced between 0 and 100, and 50 numbers that are linearly spaced between 1000 and 5000. The resulting feature matrix X and label vector y are combined to create a Pandas DataFrame with the columns “feature\_1”, “feature\_2”, “Amount”, and “label”.

To evaluate the performance of our outlier detection models, we used three popular algorithms: Local Outlier Factor (LOF) [BKNS00], One-Class Support Vector Machines

(SVM) [CV95], and Isolation Forest [LTZ08]. We used the implementation of these algorithms provided by the scikit-learn library [PVG<sup>+</sup>11].

After fitting the outlier detection models to the synthetic dataset, we made predictions for the data and added the scores to the DataFrame. We then used Matplotlib [Hun07] to visualize the results.

Figure 5.1 shows the scatter plots of the synthetic dataset with different color schemes for each column of interest. The first column, “Amount” shows the distribution of the random values added to the dataset. The second column shows the true labels of the observations, where the red points represent outliers and blue points represent normal observations. The remaining columns show the scores generated by the LOF, SVM, and Isolation Forest algorithms, respectively.



**Figure 5.1:** Synthetic data and outlier detection results using LOF, SVM, Isolation Forest.

We use the synthetic dataset to evaluate the performance of our algorithm. To achieve this, we randomly split the dataset into two parts: a labeled set representing 20% of the data, and an unlabeled set representing the remaining 80%. In order to compare the gain of our priority outlier investigation approach with the three outlier detection methods, we assume that the other three methods prioritize cases that are more likely to be outliers. Specifically, we rank the outliers according to the outlier score of the three methods. The ranking of cases using the three methods can be seen in Appendix A.3, A.1, and A.2, respectively.

Table 5.1 presents a comparison of the amount and cumulative gain of the ranked

**Table 5.1:** Cumulative gain with different methods

Rank	LOF			SVM			Isolation Forest			Our Approach		
	ID	Amt	Gain	ID	Amt	Gain	ID	Amt	Gain	ID	Amt	Gain
1	580	20.55	0.00	550	87.78	0.00	975	15.17	15.17	951	2387.76	2387.76
2	239	39.62	0.00	964	33.19	33.19	580	20.55	15.17	945	1163.27	3551.02
3	104	4673.47	0.00	932	4.85	38.04	906	42.99	58.17	946	2632.65	6183.67
4	955	70.28	70.28	580	20.55	38.04	940	59.54	117.70	690	4020.41	6183.67
5	530	42.89	70.28	962	68.18	106.22	937	43.62	161.33	170	3938.78	6183.67
6	985	57.53	127.82	131	85.56	106.22	962	68.18	229.50	490	3612.24	6183.67
7	550	87.78	127.82	985	57.53	163.75	964	33.19	262.70	514	3857.14	6183.67
8	131	85.56	127.82	955	70.28	234.04	967	77.03	339.73	758	3204.08	6183.67
9	967	77.03	204.85	967	77.03	311.06	978	76.19	415.91	357	4183.67	6183.67
10	978	76.19	281.03	978	76.19	387.25	955	70.28	486.20	316	3530.61	6183.67

cases using different outlier detection methods. Our approach outperforms the other three popular outlier detection methods (LOF, SVM, and Isolation Forest) in terms of cumulative gain. Our approach also identifies cases with a higher amount, which is an advantage over the other methods since such cases could potentially involve higher risk. Additionally, our approach ensures that a relatively high gain is achieved at the very first few cases, which is a significant advantage since it can provide more funds for future investigation costs.

Furthermore, we can see that our approach is effective in identifying outliers that are missed by other methods. For example, case ID 951 has the highest amount and gain among all cases, and it is only identified by our approach. Similarly, cases with IDs 945, 946, 690, 170, 514, 758, 357, and 316 have significantly higher gains compared to their corresponding amounts, indicating that they are potentially lucrative outliers that should be investigated. Our approach successfully identifies all these cases while the other methods fail to do so.

In conclusion, the results from Table 5.1 demonstrate that our approach has several advantages over existing popular outlier detection methods. It identifies cases with a higher amount, achieves a relatively high gain at the very first few cases, and is effective in identifying outliers that are missed by other methods. These advantages make our approach more cost-effective and efficient in detecting outliers, which is essential in many real-world applications.

## 5.2 Case Study: Credit Card Fraud Detection

### 5.2.1 Dataset Description

Credit card fraud is a major concern for financial institutions and cardholders. The publicly-available credit card fraud detection dataset [Kag16] from Kaggle is used to evaluate our model.

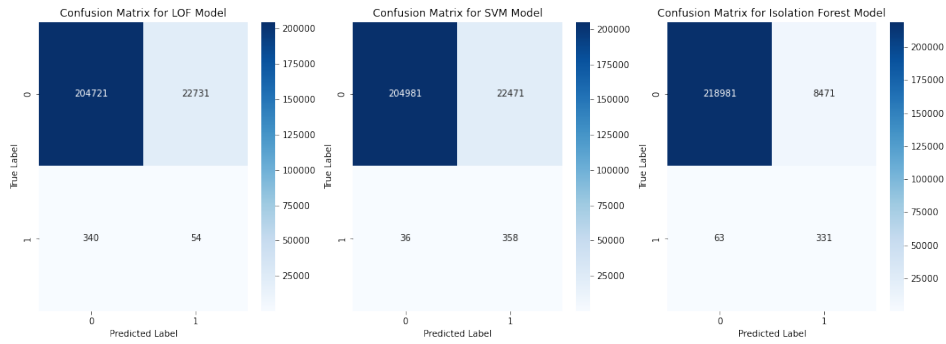
The dataset includes a total of 284,807 transactions, out of which only 492 are fraudulent. This makes the dataset highly imbalanced, with fraudulent transactions accounting for only 0.172% of the total transactions. The dataset contains transactions that were made over a two-day period in September 2013 by European cardholders. The transactions involve payments made with credit cards, and the dataset includes transactions that are both online and in-person.

Each transaction in the dataset is described by 28 anonymized features, which include both numerical and categorical variables. Some of the features include the transaction amount, the time of the transaction, the type of transaction, and various other technical features. The dataset was collected by aggregating transactions made by several European credit cardholders.

### 5.2.2 Results

We first evaluated the performance of three anomaly detection models, LOF, SVM, and Isolation Forest, on the credit card fraud detection dataset. The Isolation Forest model outperformed the other two models in terms of accuracy, precision, recall, and F1-score, achieving an accuracy of 0.962545 and an F1-score of 0.071988. The LOF and SVM models had significantly lower accuracy and F1-scores, with LOF having the lowest scores overall. The precision and recall scores varied among the models, with SVM having the highest precision and recall but with much lower accuracy and F1-score compared to Isolation Forest. The results are summarized in Table 5.2 and visualized in Figure 5.2.

Then we apply our approach to investigate the credit card transaction data. For outlier



**Figure 5.2:** Confusion matrices for LOF, SVM, and Isolation Forest models.

**Table 5.2:** Results of anomaly detection models on credit card dataset.

Model	Accuracy	Precision	Recall	F1-Score
LOF	0.898743	0.002370	0.137056	0.004659
SVM	0.901218	0.015682	0.908629	0.030832
Isolation Forest	0.962545	0.037605	0.840102	0.071988

cases that are identified, the gain is the transaction amount. For cases that are normal, the gain is set to 0. Since we assume the labeling cost is constant for all cases, we utilized the cumulative gain of all the ranked cases for comparison purposes.

Table 5.3 presents the comparison of the amount and cumulative gain of the ranked cases using various outlier detection methods with the credit card transaction data. The results obtained from this real dataset align with the findings from the synthetic data.

Due to the imbalanced nature of the dataset, and with 28 dimensions, the accuracy of LOF and SVM in detecting outliers is unsatisfactory. The false positives from these methods may lead to the loss of labeling cost with zero gain. Isolation Forest, on the other hand, has a relatively higher accuracy in identifying outliers, but the outliers ranked by its outlying score have very small transaction amounts, which may be less than the labeling cost in real-world scenarios.

Our algorithm offers an advantage in such a situation. It not only examines cases with

a large amount, indicating a potentially high risk, but also prioritizes potential outliers with a large amount. By prioritizing cases with a high amount, our algorithm can identify potentially lucrative outliers, leading to higher gains and better risk management. Thus, our approach can provide a more effective and efficient method for outlier investigation in real-world applications.

**Table 5.3:** Cumulative gain using the credit card transaction data.

Rank	LOF		SVM		Isolation Forest		Our Approach	
	Amt	Gain	Amt	Gain	Amt	Gain	Amt	Gain
1	2.24	0	370.9	0	25691.16	0	25691.16	0
2	35.55	0	500	0	1.00	1.00	19656.53	0
3	0.76	0	25691.16	0	1.00	2.00	18910	0
4	1.79	0	303.8	0	1.00	3.00	1504.93	1504.93
5	1.00	0	8360	0	0.01	3.01	1721.84	1504.93
6	43.5	0	3804.63	0	1.00	4.01	1591.26	1504.93
7	0.92	0	10199.44	0	2.28	6.29	1588	1504.93
8	1.00	0	102	0	1.00	7.29	1587.46	1504.93
9	2.99	0	7367	0	1.00	8.29	1402.16	2907.09
10	0.76	0	4248.34	0	1.63	9.92	1096.99	4004.08

# Chapter 6

## Conclusion

This master’s thesis proposes a novel approach to outlier detection that contribute to the development of effective and efficient outlier investigation strategies with practical applications in various domains, particularly in the context of fraud detection. The main contribution of this work is a cost-sensitive, semi-supervised, and active-learning priority outlier investigation model that can rank outliers and efficiently identify top- $k$  unlabeled cases with the highest expected overall gain, while minimizing investigation costs. By incorporating semi-supervised and active learning techniques, the model can efficiently and effectively detect outliers with minimal investigation costs.

In addition, this thesis also makes use of graph representation to improve the efficiency and accuracy of the proposed algorithm. Specifically, the neighborhood of cases is used for kernel density estimation, which improves the efficiency of the algorithm. Moreover, graph representation is employed to improve the accuracy of the model by taking into account the similarities and relationships among cases.

While this thesis has made significant contributions to the field of outlier detection, there are also some limitations. For example, the proposed approach relies on a kernel density estimation-based approach, which may not be optimal for all datasets. Additionally, the model assumes that the cost of investigating a suspected case is fixed, which may not be the case in all real-world scenarios.

In future work, it would be beneficial to explore the use of other outlier detection methods within the proposed algorithm. For instance, Local Outlier Factor (LOF), Isolation Forest, and Support Vector Machines (SVM) are popular techniques that have shown promising results in various outlier detection tasks. Integrating these methods into the current approach could enhance the accuracy and robustness of the proposed model, par-

ticularly in scenarios where the underlying data distribution is complex or non-linear.

Additionally, another potential avenue for future research is to investigate the use of graph neural networks (GNNs) for outlier detection. GNNs are a powerful class of deep learning models that can effectively capture the structural information and dependencies among data points in graph-structured data. By incorporating GNNs into the current framework, it may be possible to further improve the efficiency and effectiveness of outlier detection by leveraging the underlying graph structure of the data.

# Appendix A

## Appendix

Table A.1, A.2 and A.3 present the ranked outliers detected from LOF, SVM and Isolation Forest using synthetic data. The outliers are ranked based on the score assigned by the respective method. The higher the score, the more likely it is that the case is an outlier. The ‘Label’ column indicates the ground truth labels.

Table A.4 shows the results of our neighborhood-based algorithm for synthetic data. It is noticeable that the ‘Amount’ of the ranked outliers from our approach is much higher than those of the previous three methods.

**Table A.1:** Local Outlier Factor outliers (ranked)

ID	Feature 1	Feature 2	Amount	Label	LOF
580	-2.50	2.29	20.55	1	-1.98
239	3.08	1.12	39.62	1	-2.04
104	0.52	3.85	4673.47	1	-2.08
955	5.73	6.23	70.28	-1	-2.12
530	0.77	-2.85	42.89	1	-2.15
985	4.59	1.06	57.53	-1	-2.36
550	1.00	-2.90	87.78	1	-2.38
131	-3.24	-1.02	85.56	1	-2.43
967	6.64	2.50	77.03	-1	-2.43
978	1.86	7.14	76.19	-1	-2.88

**Table A.2:** One-Class SVM outliers (ranked)

ID	Feature 1	Feature 2	Amount	Label	SVM
550	1.00	-2.90	87.78	1	13.36
964	5.10	5.96	33.19	-1	13.26
932	5.30	2.34	4.85	-1	13.03
580	-2.50	2.29	20.55	1	12.63
962	6.07	3.08	68.18	-1	12.36
131	-3.24	-1.02	85.56	1	11.94
985	4.59	1.06	57.53	-1	10.61
955	5.73	6.23	70.28	-1	9.98
967	6.64	2.50	77.03	-1	8.09
978	1.86	7.14	76.19	-1	5.04

**Table A.3:** Isolation Forest outliers (ranked)

ID	Feature 1	Feature 2	Amount	Label	IForest
975	2.23	5.26	15.17	-1	-0.09
580	-2.50	2.29	20.55	1	-0.10
906	5.61	4.90	42.99	-1	-0.10
940	4.96	5.62	59.54	-1	-0.11
937	5.31	5.40	43.62	-1	-0.11
962	6.07	3.08	68.18	-1	-0.11
964	5.10	5.96	33.19	-1	-0.12
967	6.64	2.50	77.03	-1	-0.13
978	1.86	7.14	76.19	-1	-0.14
955	5.73	6.23	70.28	-1	-0.16

**Table A.4:** Results of Neighborhood-based Algorithm

ID	Feature 1	Feature 2	Amount	label	LOF	SVM	IForest
<b>951</b>	5.173125	4.369642	2387.755102	-1.0	-1.044008	17.822125	-0.044221
<b>945</b>	4.546284	4.006422	1163.265306	-1.0	-0.977455	19.051091	0.006692
<b>946</b>	3.563614	3.890390	2632.653061	-1.0	-0.948273	19.024309	0.011826
<b>690</b>	0.677926	-0.487911	4020.408163	1.0	-1.027873	18.479167	0.162060
<b>170</b>	-0.822220	0.243687	3938.775510	1.0	-1.012133	18.497634	0.155133
<b>490</b>	0.785800	0.425458	3612.244898	1.0	-0.969414	18.010126	0.163016
<b>514</b>	0.613518	-1.022793	3857.142857	1.0	-1.041445	18.722789	0.138781
<b>758</b>	1.513450	0.630812	3204.081633	1.0	-0.986748	18.261032	0.137700
<b>357</b>	-0.037635	1.103302	4183.673469	1.0	-0.990388	18.174699	0.164421
<b>316</b>	-0.158008	-0.426881	3530.612245	1.0	-1.012321	18.261214	0.173073

# Bibliography

- [AM13] Mohiuddin Ahmed and Abdun Naser Mahmood. A novel approach for outlier detection and clustering improvement. In *2013 IEEE 8th Conference on Industrial Electronics and Applications (ICIEA)*, pages 577–582, 2013.
- [ATK15] Leman Akoglu, Hanghang Tong, and Danai Koutra. Graph based anomaly detection and description: A survey. *Data Mining and Knowledge Discovery*, 29(3):626–688, 2015.
- [AZL06] Naoki Abe, Bianca Zadrozny, and John Langford. Outlier detection by active learning. In *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '06*, page 504–509, New York, NY, USA, 2006. Association for Computing Machinery.
- [BKNS00] Markus M Breunig, Hans-Peter Kriegel, Raymond T Ng, and Jörg Sander. Lof: identifying density-based local outliers. In *Proceedings of the 2000 ACM SIGMOD international conference on Management of data*, pages 93–104, 2000.
- [CBK09] Varun Chandola, Arindam Banerjee, and Vipin Kumar. Anomaly detection: A survey. *ACM Computing Surveys*, 41(3):1–58, 2009.
- [Cha04] Deepayan Chakrabarti. Autopart: Parameter-free graph partitioning and outlier detection. In Jean-Francois Boulicaut, Floriana Esposito, Fosca Gianotti, and Dino Pedreschi, editors, *Knowledge Discovery in Databases: PKDD 2004*, volume 3202 of *Lecture Notes in Computer Science*, Berlin, Heidelberg, 2004. Springer.
- [CMA19] Anshika Chaudhary, Himangi Mittal, and Anuja Arora. Anomaly detection using graph neural networks. In *2019 International Conference on Machine Learning, Big Data, Cloud and Parallel Computing (COMITCon)*, pages 346–350, 2019.

- [CMC19] Raghavendra Chalapathy, Vinayakumar R Menon, and Sanjay Chawla. Deep learning for anomaly detection: A survey. *arXiv preprint arXiv:1904.03404*, 2019.
- [CV95] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine learning*, 20(3):273–297, 1995.
- [Elk01] Charles Elkan. The foundations of cost-sensitive learning. *Proceedings of the 17th International Joint Conference on Artificial Intelligence*, pages 973–978, 2001.
- [FBF77] Jerome H Friedman, Jon L Bentley, and Richard A Finkel. An algorithm for finding nearest neighbors. *IEEE Transactions on Computers*, C-26(11):1097–1101, 1977.
- [HKF04] V. Hautamaki, I. Karkkainen, and P. Franti. Outlier detection using k-nearest neighbour graph. In *Proceedings of the 17th International Conference on Pattern Recognition, 2004. ICPR 2004.*, volume 3, pages 430–433 Vol.3, 2004.
- [HMvdW20] Charles R Harris, K Jarrod Millman, and Stéfan J van der Walt. Array programming with numpy. *Nature*, 585(7825):357–362, 2020.
- [Hun07] J. D. Hunter. Matplotlib: A 2d graphics environment. *Computing in Science & Engineering*, 9(3):90–95, 2007.
- [Kag16] Kaggle. Credit card fraud detection. <https://www.kaggle.com/mlg-ulb/creditcardfraud>, 2016. Accessed on March 6th, 2023.
- [KAH<sup>+</sup>17] Akshay Krishnamurthy, Alekh Agarwal, Tzu-Kuo Huang, Hal Daumé, III, and John Langford. Active learning for cost-sensitive classification. In Doina Precup and Yee Whye Teh, editors, *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 1915–1924. PMLR, 06–11 Aug 2017.

- [KN00] Erik M Knorr and Raymond T Ng. Distance-based outliers: algorithms and applications. *The VLDB Journal*, 8(3-4):237–253, 2000.
- [LS08] Charles X Ling and Victor S Sheng. Cost-sensitive learning and the class imbalance problem. *Encyclopedia of machine learning*, 2011:231–235, 2008.
- [LTZ08] Fei Tony Liu, Kai Ming Ting, and Zhi-Hua Zhou. Isolation forest. In *Proceedings of the 2008 Eighth IEEE International Conference on Data Mining*, pages 413–422. IEEE, 2008.
- [LZ06] Xu-ying Liu and Zhi-hua Zhou. The influence of class imbalance on cost-sensitive learning: An empirical study. In *Sixth International Conference on Data Mining (ICDM’06)*, pages 970–974, 2006.
- [NZL<sup>+</sup>19] Wing W. Y. Ng, Jianjun Zhang, Chun Sing Lai, Witold Pedrycz, Loi Lei Lai, and Xizhao Wang. Cost-sensitive weighting and imbalance-reversed bagging for streaming imbalanced and concept drifting in electricity pricing classification. *IEEE Transactions on Industrial Informatics*, 15(3):1588–1597, 2019.
- [Par62] Emanuel Parzen. On Estimation of a Probability Density Function and Mode. *The Annals of Mathematical Statistics*, 33(3):1065 – 1076, 1962.
- [PM04] Dan Pelleg and Andrew W Moore. Active learning for anomaly and rare-category detection. In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 426–435, 2004.
- [PMVZ18] Tiago Pimentel, Marianne Monteiro, Adriano Veloso, and Nivio Ziviani. Deep active learning for anomaly detection. *arXiv preprint arXiv:1802.03971*, page 8, 2018.
- [PSCH21] Guansong Pang, Chunhua Shen, Longbing Cao, and Anton Van Den Hengel. Deep learning for anomaly detection: A review. *ACM Comput. Surv.*, 54(2), mar 2021.

- [PVG<sup>+</sup>11] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, and Edouard Duchesnay. Scikit-learn: Machine learning in python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.
- [Ros56] Murray Rosenblatt. Remarks on Some Nonparametric Estimates of a Density Function. *The Annals of Mathematical Statistics*, 27(3):832 – 837, 1956.
- [Sco04] David W Scott. Multivariate density estimation: theory, practice, and visualization. *John Wiley & Sons*, 10(1):139–142, 2004.
- [Set09] Burr Settles. Active learning literature survey. 2009.
- [SS17] Ozan Sener and Silvio Savarese. Active learning for convolutional neural networks: A core-set approach, 2017.
- [TB19] Holger Trittenbach and Klemens Böhm. One-class active learning for outlier detection with multiple subspaces. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management, CIKM '19*, page 811–820, New York, NY, USA, 2019. Association for Computing Machinery.
- [TD04] David M.J. Tax and Robert P.W. Duin. Support Vector Data Description. *Machine Learning*, 54(1):45–66, January 2004.
- [TEB18] Holger Trittenbach, Adrian Englhardt, and Klemens Böhm. An overview and a benchmark of active learning for one-class classification. *CoRR*, abs/1808.04759, 2018.
- [TSHT03] Gokhan Tur, R.E. Schapire, and Dilek Hakkani-Tur. Active learning for spoken language understanding. volume 1, pages I–276, 05 2003.

- [WLG19] Chao Wang, Zhen Liu, Hui Gao, and Yan Fu. Vos: A new outlier detection model using virtual graph. *Knowledge-Based Systems*, 185:104907, 2019.
- [ZE01] Bianca Zadrozny and Charles Elkan. Learning and making decisions when costs and probabilities are both unknown. In *Proceedings of the Seventh International Conference on Knowledge Discovery and Data Mining*, pages 204–213, 2001.