University of Cincinnati

Date: 1/15/2015

I, Vineet Joshi, hereby submit this original work as part of the requirements for the degree of Doctor of Philosophy in Computer Science & Engineering.

It is entitled: Unsupervised Anomaly Detection in Numerical Datasets

Student's name: Vineet Joshi

This work and its defense approved by:

Committee chair: Raj Bhatnagar, Ph.D.

Committee member: Prabir Bhattacharya, Ph.D.

Committee member: Karen Davis, Ph.D.

Committee member: Anil Jegga, D.V.M., M.Res.

Committee member: Mario Medvedovic, Ph.D.
Unsupervised Anomaly Detection in Numerical Datasets

A dissertation submitted to the Division of Research and Advanced Studies of the University of Cincinnati in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

in the Department of Electrical Engineering and Computing Systems of the College of Engineering and Applied Science

January 2015

by

Vineet Joshi
MS (CS), Univ. Cincinnati

Dissertation Advisor and Committee Chair: Dr. Raj Bhatnagar
Abstract

Anomaly detection is an important problem in data mining with diverse applications such as financial fraud detection, medical diagnosis and computer systems intrusion detection. Anomalies are data points that are substantially different from the rest of the population. These generally represent valuable information about the system for which the analyst is interested in detecting anomalies accurately and efficiently, and then taking appropriate actions in response. There are scenarios where tremendous impact can be made by detecting anomalies in a timely and accurate manner, e.g. early detection of spurious credit card transactions can prevent financial damages to a credit card holder as well as the banking institution that issued the credit card. Similarly, abnormal readings by a sensor monitoring an industrial plant can help detect system faults and avert damages. All these applications have led to an interest in finding efficient methods for detecting anomalies. Anomaly detection continues to be an active research area within data mining.

In this dissertation we investigate various aspects of anomaly detection problem. To determine anomalies in a dataset, a concrete definition of anomalous behavior is required. There is no single universally applicable definition of anomalies because each definition presents perspective of an anomalous behavior which may not necessarily apply across diverse
datasets. In this work we investigate a new definition of anomalous behavior. We compare this definition with an existing definition of outlier-ness and demonstrate the effectiveness of the new definition.

We further present a refinement of the metric of outlier-ness that we have mentioned above. We discovered that the metric initially proposed can be altered to yield a new metric of outlier-ness that accentuates the difference in the outlier-ness scores of strong outliers as compared to the non-anomalous datapoints. We compare this updated metric with the metric we first presented, and also with an established metric of outlier-ness.

As the number of attributes increases, the distances between the nearest and the farthest data points tend to converge resulting in distance concentration. Thus the anomalies reported by most definitions of anomalous behavior tend to lose meaning with increasing numbers of attributes. It has been suggested that in such datasets, the anomalies are located in smaller subspaces of attributes. Hence, anomalies should be searched in subspaces of the attributes, instead of the complete attribute space. However the number of subspaces increases very rapidly as the number of attributes increases. The number of possible subspaces for a given set of attributes in the dataset is a combinatorial number. This makes, an exhaustive search through all possible subspaces infeasible. In this dissertation, after presenting a novel definition of anomalous behavior, we present an efficient method of exploring the possible subspaces arising from the attributes of a dataset.
The subspaces of attributes in any dataset can be arranged in a lattice. The anomalous behavior of data points as we traverse this lattice conveys meaningful information about the structure of the data. In the fourth problem that we address, we present a method that investigates the anomalous behavior of data points across the different subspaces in the lattice in which the same point displays anomalous behavior. Further, our method also computes the contiguous regions of the subspace lattice where the same data point demonstrates anomalous behavior.
Dedicated to the loving memory of my brother Puneet
Acknowledgements

I want to thank my advisor Dr. Raj Bhatnagar for his support and guidance throughout my dissertation work. I consider it a privilege that I got a chance to work with him.

I want to thank Dr. Mario Medvedovic for employing me as a research assistant in his lab. His financial support has been a blessing for me.

I have found great support from my parents throughout the course of my studies. My tauji Dr. S. K. Joshi has been an inspiration for me. His example is an ideal that I have strived for. A special thanks to my wife Divya for all her strength and support during my research work.

I want to thank my friends for their constant support and understanding over the course of my doctoral work. My friend Krishnendu has been very supportive over the previous few years. I want to thank him for his advice and opinion over the many years. I want to thank Amitabh, Ravi, Abhik, Zhen, Jagadeesh and Dippy for their camaraderie. I want to thank Mukta and Prateek for their help and support over the many years.
## Contents

Abstract

1 Introduction

1.1 Cohesiveness-Based Outlier Factor

1.2 Improved Cohesiveness-Based Outlier Factor

1.3 eSelect: Effective Subspace Selection for Detection of Anomalies

1.4 Outlier Analysis over Lattice of Contiguous Subspaces

1.5 Dissertation Outline

2 Cohesiveness-Based Outlier Factor (CBOF)

2.1 Background

2.2 Related Work

2.2.1 Clustering-based Methods

2.2.2 Density-based Methods

2.3 Preliminary Concepts

2.3.1 k-Nearest Neighbors
2.3.2 Sum of Distances Between Nearest Neighbors Including P ........................................... 16
2.3.3 Sum of Distances Between Nearest Neighbors Excluding P ....................................... 17
2.4 Cohesiveness-based Outlier Factor (CBOF) ................................................................. 18
  2.4.1 Algorithm Specification ................................................................. 21
  2.4.2 Scaling of CBOF ................................................................................... 22
  2.4.3 Effect of Dataset Shape on CBOF ......................................................... 28
  2.4.4 Comparison with LOF ........................................................................ 28
  2.4.5 Complexity Analysis ........................................................................... 30
2.5 Experimental Evaluation ......................................................................................... 31
  2.5.1 Synthetic Dataset .................................................................................. 31
  2.5.2 Real Datasets ......................................................................................... 32
    Design of Experiment: .................................................................................. 33
    Results for Wisconsin Breast Cancer Dataset: ........................................... 38
    Results for Spam Dataset: ......................................................................... 40
    Results for Thyroid Disease ANN Training Dataset: ................................ 43
2.6 Conclusion ............................................................................................................ 46

3 Improved Cohesiveness-Based Outlier Factor (iCBOF) ........................................... 48
  3.1 Background ..................................................................................................... 48
  3.2 Related Work .................................................................................................. 51
  3.3 Our Approach .................................................................................................. 52
4 eSelect: Effective Subspace Sampling for Outlier Detection in High Dimensional Data

4.1 Background ........................................ 64
4.2 Related Work ...................................... 66
4.3 Our Approach ..................................... 69
  4.3.1 Computational Complexity ...................... 74
4.4 Experimental Evaluation ........................... 75
  4.4.1 Results for Musk dataset ....................... 75
  4.4.2 Results for Spam dataset ....................... 80
  4.4.3 Results for Wisconsin Breast Cancer dataset .... 83
4.5 Conclusion ......................................... 87

5 Outlier Analysis Using Lattice of Contiguous Subspaces 90

5.1 Related Work .................................... 95
5.2 Our Approach .................................... 98
5.2.1 Subspace Lattice Exploration .......................... 98
5.2.2 Summarization of Subspaces for Outliers .......... 102
5.2.3 Determination of Contiguous Subspaces ............ 103
5.3 Results ......................................................... 110
5.3.1 Iris Dataset ................................................. 110
5.3.2 Seeds Dataset ............................................ 113
5.4 Conclusion ...................................................... 116

6 Conclusions and Future Directions 119
6.1 Future Work .................................................. 122
6.1.1 Metric of Outlier-ness .................................. 122
6.1.2 Outliers in High-Dimensional Data .................. 123
6.1.3 Outliers in Subspace Lattice ......................... 123
List of Tables

2.1 Avg. *AUC* values for *CBOF* for Wisconsin Breast Cancer Dataset ........................................ 40

2.2 Avg. *AUC* values for *CBOF* for Spam Dataset .................. 43

2.3 Avg. *AUC* values for *CBOF* for Thyroid Disease ANN Training Dataset ................................. 45

3.1 Avg. *AUC* values for *iCBOF* for Sonar dataset ............... 55

3.2 Avg. *AUC* values for *iCBOF* for Email Spam dataset ....... 57

3.3 Avg. *AUC* values for *iCBOF* for Wisconsin Breast Cancer dataset ........................................... 58

3.4 Avg. *AUC* values for *iCBOF* for Musk dataset ............... 59

3.5 Avg. *AUC* values for *iCBOF* for Thyroid Disease ANN Training dataset ................................. 60

4.1 Avg. *AUC* values for *eSelect* for Musk dataset .............. 79

4.2 Avg. *AUC* values for *eSelect* for Email Spam dataset ...... 83

4.3 Avg. *AUC* values for *eSelect* for Wisconsin Breast Cancer dataset ........................................... 85
5.1 Outlier Analysis on Iris Dataset using Z-Dist . . . . . . . . . 112
5.2 Outlier Analysis on Iris Dataset using LOF . . . . . . . . . 113
5.3 Outlier Analysis on Seeds Dataset using Z-Dist . . . . . . . . 115
5.4 Outlier Analysis on Seeds Dataset using LOF . . . . . . . . . 116
List of Figures

2.1 k-Nearest Neighborhood .............................................. 14
2.2 Sum of distances between all pairs of points in \( k \)-neighborhood
    of \( P \) including \( P \) .................................................. 16
2.3 Sum of distances between all pairs of points in \( k \)-neighborhood
    of \( P \) excluding \( P \) .................................................. 17
2.4 Sum of distances between all points in neighborhood for \( P' \)
    including \( P' \) .......................................................... 19
2.5 Sum of distances between all points in neighborhood for \( P' \)
    excluding \( P' \) .......................................................... 20
2.6 Square Dataset Within Range (-2, 2) ............................... 23
2.7 Square Dataset Within Range (-5, 5) ............................... 24
2.8 Circular Dataset Within Range (-10, 10) ........................... 25
2.9 Circular Dataset Within Range (-50, 50) ........................... 26
2.10 Linear Dataset Within Range (1, 10) ............................... 27
2.11 CBOF vs LOF Case 1 .................................................... 29
2.12 CBOF vs LOF Case 2 .................................................... 29
2.13 CBOF and LOF Scores - Compact Dataset ......................... 32
2.14 CBOF and LOF Scores - Spread-Out Dataset ............... 33
2.15 2-Class Dataset ............................................. 34
2.16 Curated 2-Class Dataset .................................... 34
2.17 Wisconsin Breast Cancer Dataset, \( k = 5 \) ................. 37
2.18 Wisconsin Breast Cancer Dataset, \( k = 10 \) ............... 38
2.19 Wisconsin Breast Cancer Dataset, \( k = 15 \) ............... 39
2.20 Wisconsin Breast Cancer Dataset, \( k = 20 \) ............... 39
2.21 Spam Dataset, \( k = 5 \) ...................................... 40
2.22 Spam Dataset, \( k = 10 \) ...................................... 41
2.23 Spam Dataset, \( k = 15 \) ...................................... 42
2.24 Spam Dataset, \( k = 20 \) ...................................... 42
2.25 Thyroid Disease ANN Training Dataset, \( k = 5 \) ........ 43
2.26 Thyroid Disease ANN Training Dataset, \( k = 10 \) ........ 44
2.27 Thyroid Disease ANN Training Dataset, \( k = 15 \) ........ 44
2.28 Thyroid Disease ANN Training Dataset, \( k = 20 \) ........ 45

3.1 CBOF Score Computation for an Example Dataset ........... 49
3.2 CBOF Score Computation for an Example Dataset ........... 50
3.3 Sonar dataset, \( k = 20 \) ....................................... 55
3.4 Spam dataset, \( k = 15 \) ....................................... 56
3.5 Wisconsin Breast Cancer dataset, \( k = 15 \) ................. 57
3.6 Musk dataset, \( k = 5 \) ....................................... 58
3.7 Thyroid Disease ANN Training dataset, \( k = 5 \) ........... 60
4.1 Musk Dataset, $k = 5$, $attrib = 1$ ........................................ 76
4.2 Musk Dataset, $k = 10$, $attrib = 1$ ............................... 77
4.3 Musk Dataset, $k = 15$, $attrib = 1$ ............................... 77
4.4 Musk Dataset, $k = 20$, $attrib = 1$ ............................... 78
4.5 Spam Dataset, $k = 5$, $attrib = 1$ ................................. 80
4.6 Spam Dataset, $k = 10$, $attrib = 1$ ............................... 81
4.7 Spam Dataset, $k = 15$, $attrib = 1$ ............................... 82
4.8 Spam Dataset, $k = 10$, $attrib = 1$ ............................... 82
4.9 Wisconsin Breast Cancer Dataset, $k = 5$, $attrib = 1$ ...... 84
4.10 Wisconsin Breast Cancer Dataset, $k = 10$, $attrib = 1$ ... 84
4.11 Wisconsin Breast Cancer Dataset, $k = 15$, $attrib = 1$ ... 85
4.12 Wisconsin Breast Cancer Dataset, $k = 20$, $attrib = 1$ ... 86

5.1 Subspace Lattice with a Contiguous Region of Subspaces ... 91
Anomaly detection is an important problem in data mining with diverse applications such as financial fraud detection, medical diagnosis and computer system intrusion detection. Anomalies are data points that are substantially different from the rest of the population. These generally represent valuable information about the system in which the stakeholders would like to efficiently and accurately detect anomalies and then take appropriate action in response. There are scenarios where tremendous impact can be made by detecting anomalies in a timely and accurate manner. E.g. early detection of spurious credit card transactions can prevent financial damages to a credit card holder as well as the banking institution that issued the credit card. Similarly, abnormal readings by a sensor monitoring an industrial plant can help detect system faults and avert damages. All these applications have led to an interest in finding efficient methods for detecting anomalies. Anomaly detection continues to be an active research
area within data mining. In this thesis we have worked on various aspects of anomaly detection.

In our research we have addressed different aspects of anomaly detection problem. These are:

1. A new metric of outlier-ness called $CBOF$.

2. Refinement called $iCBOF$ of the metric of outlier-ness based on our work mentioned in point 1.

3. A method called $eSelect$ to identify outliers in very high dimensional data.

4. Analysis of outlier-ness over contiguous subspaces in the subspace lattice of a dataset.

1.1 Cohesiveness-Based Outlier Factor

An unambiguous definition of anomalous behavior is required in order to be able to detect anomalies in a data set. While various definitions of anomalous behavior have been proposed, there is no universally accepted definition of an anomaly. This is because different anomaly detection tasks require a definition of anomalous behavior that satisfies the specific requirements of the task. This is analogous to the problem of data clustering, where there is no single universal definition of data clusters. Different def-
Definitions of data clustering have arisen to accommodate the varying requirements of different data clustering tasks.

Irrespective of the prevalence of different definitions of anomalies, an anomaly is a data point that is markedly different from the non-anomalous data points. Thus we need a description of a non-anomalous data point, so that we can declare all the data points that significantly deviate from our idea of ‘normal’ data points as anomalies. In our work we consider normal data points as the ones that reside in regions of similar density. By this, we mean that ‘normal’ data points are located at about the same distance from their neighbors. Then we define anomalies as data points that are placed unusually far from the data points in their neighborhoods.

To numerically measure the outlier-ness of a data point, we compute the sum of the distances between all pairs of data points in the given point’s neighborhood, including the given point. We then compute the fraction of this sum that is contributed by the distances involving the given data point. This number provides the degree of outlier-ness of the given point. This numerical figure is called the Cohesiveness-Based Outlier Factor (CBOF). Anomalous data points will have a higher value of this number than the non-anomalous points. Analyst can then use a threshold and filter out all points with scores higher than the threshold as outliers.
1.2 Improved Cohesiveness-Based Outlier Factor

While CBOF is an effective measure of outlier-ness, there is potential to further refine the outlier-ness information that can be derived by investigating the distances between any given data point and its neighbors. The goal of this enhancement is to further accentuate the difference in the outlier-ness scores of strong outliers when compared with non-anomalous data points. We term this refinement the Improved Cohesive-ness Based Outlier Factor (iCBOF). The method involves accumulation of the CBOF scores of a given datapoint as we expand outwards from the datapoint to its successively distant neighbors.

1.3 eSelect: Effective Subspace Selection for Detection of Anomalies

In high dimensional data sets, search for anomalies using all the attributes in the data set at the same time is counterproductive. This is because of the curse of dimensionality[7] wherein the distance between the closest and the farthest data points tend to converge to the same value[3]. Consequently, the methods that use all the complete attribute space in the dataset for detecting anomalies lose algorithmic effectiveness[3]. It has been sug-
gested that in such high dimensional datasets, the anomalies exist in low-dimensional projections (i.e. a subspace) of the attributes[2].

However, the number of subspaces possible for a given set of attributes is a combinatorial number and increases very rapidly. Consequently, an exhaustive search through all the possible subspaces of a large dataset is not possible.

Similar issues related to very large search space are encountered by some other data mining problems also, such as the association rule mining. However in such problems, the search space can be pruned because of the upward or downward closed properties of the search criteria. In such domains the search is guided by such a property. Consequently, large portions of the search space get pruned. No such property exists for the problem of search of anomalies through subspace lattice. The presence or absence of anomalies in a given subspace provides little information about the presence of anomalies in a subspace derived by removing or adding an attribute.

In this dissertation we present an effective subspace sampling method that picks the low dimensional subspaces in preference over high dimensional subspace. We base this heuristic on the hypothesis that outliers are more likely to be discovered in low-dimensional subspaces that in the high-dimensional subspaces. This hypothesis is based on the experiences with the curse of dimensionality wherein high dimensional subspaces are not very useful for anomaly detection.
The proposed method eSelect uses the heuristic proposed by Aggarwal et al.[3] which provides a bound on the number of attributes that need to be reasonably investigated in order to determine anomalies. However, that method of searching for anomalies is specific to a particular definition of anomalies. eSelect can be combined with any definition of anomalies and will support the investigation of anomalies using that definition over the attribute subspaces.

1.4 Outlier Analysis over Lattice of Contiguous Subspaces

The subspaces derived from a set of attributes can be arranged in a lattice. As we traverse through this subspace lattice, the subspaces in which a data point displays outlying behavior conveys insights about the dataset. We discover the contiguous region of the subspace lattice where the same data point displays outlying behavior. We also report the topmost and the bottom-most subspaces in this contiguous region for a given outlier.

1.5 Dissertation Outline

Remainder of this dissertation consists of the following chapters. Chapter 2 discusses the work related to Cohesiveness-Based Outlier Factor (CBOF), which is a novel metric of outlier-ness we have proposed. Chapter 3 presents
an refinement over the novel method CBOF. The new method is called *Improved Cohesiveness-Based Outlier Factor (iCBOF)*. Chapter 4 discusses the work related to *eSelect*, the method to explore high dimensional datasets for anomalies. Chapter 5 discusses the investigation of regions of subspace lattice in which the same data point displays outlying behavior.
Chapter 2

Cohesiveness-Based Outlier Factor (CBOF)

In this chapter we will discuss a new metric of outlier-ness called the Cohesiveness-Based Outlier Factor (CBOF). This novel definition represents a numerical score that captures the extent to which a given data point is abnormally distant from the other data points within its neighborhood. We consider that non-anomalous data is arranged more or less with a certain regularity, whereas anomalies are the data points that are unusually separated from the data points in their neighborhood. Cohesiveness-Based Outlier Factor is a numerical figure between 0 and 1 that captures this property.

Anomalous data points will have higher CBOF value than non-anomalous data points. The higher the CBOF value of a data point, the greater is its the outlier-ness. Instead of generating a binary label signifying whether any data point is or is not an outlier, CBOF specifies the degree of outlier-
ness for the data point in the dataset. The analyst can then apply thresholds on these CBOF values and designate all points with value higher than the cutoff as outliers.

2.1 Background

Data mining research literature includes various definitions of outliers. There is no single universally applicable definition of anomalous data records that encompass the requirements of various anomaly detection tasks.

Hawkins has defined outliers as[13]:

“An outlier is an observation which deviates so much from the other observations as to arouse suspicions that it was generated by a different mechanism.”

While this definition conveys the intuition behind the meaning of outliers, it leaves many details open to interpretation. Details of the property that Hawkins refers to as *deviation* and the mechanism of measuring this deviation vary from one anomaly detection task to another. This has led to a range of definitions of what constitutes the outliers. Each definition offers a unique perspective on the meaning of anomalous records for a specific anomaly detection task.

This situation is analogous to the case with clustering algorithms, where there is a plethora of algorithms, and each algorithm captures a specific
meaning of what a cluster is in a specific data mining task. A given clustering algorithm is adept at determining specific types of clusters. Hence the appropriate cluster algorithm for a specific clustering task depends on how the analyst expects the data points to cluster together. The analyst can choose the appropriate clustering algorithm for a given data set based on his/her expertise in the domain, and experience about how the data points relate to each other.

Similarly for anomaly detection tasks, the analyst can choose the appropriate definition of outlier-ness based on the knowledge about the data set, and about the way the anomalous data points stand-out from the normal data points. With this in view, various researchers have proposed different definitions of what constitutes an outlying behavior.

This chapter discusses one such novel definition of outlier-ness. The new definition, which can be compared to existing definitions of anomalies, brings a new perspective and captures the kinds of anomalies that would evade detection with other state-of-the-art method, namely LOF.

2.2 Related Work

Statistical approaches to anomaly detection have been one of the earliest methods of detecting anomalies and these techniques can be traced back to the nineteenth century[12]. In these methods, a statistical model of the data is created based on a probability distribution. The analyst selects a
probability distribution to model the data, and then determines the parameters for the model[1]. It is assumed that the data points that conform to the model are normal records whereas the data points that deviate from the model are anomalous. Effectiveness of these methods depends on the applicability of the assumed model for a given data set. If the assumed model does not represent actual data, the inferences about anomalous data will be spurious.

It is difficult to know a-priori the appropriate statistical model to be used for a dataset and the analyst needs to call upon his/her experience with the data sets from a given domain to choose the correct model. Also, once a model has been assumed, determining the correct parameters to fit a data set needs to be carefully done to avoid over-fitting[1]. Further, these methods suffer from the obscurity of interpretability of the results. Specifically, the analyst would want to be able to understand why a given data point in the dataset is an outlier. Finding such insights from statistical methods is not straightforward and does not contribute to analyst’s goal of determining the underlying causes of anomalous behavior[1].

Because of the difficulties associated with choosing a statistical model and efficiently determining the parameters for large high dimensional datasets, data mining based methods for anomaly detection have also been explored. Many of these methods are based on proximity between data points. These can be broadly classified into the following[1]:

- Clustering-based methods
• Distance-based methods

• Density-based methods

2.2.1 Clustering-based Methods

Clustering-based methods use the distances to neighbors as the basis for outlier detection. Knorr et al. proposed that points for which greater than a fraction $f$ of data points exist at distances $D$ or greater be considered outliers, where $f$ and $D$ are user-defender parameters[15]. Ramaswamy et al. proposed the use of distances to $k$-nearest neighbors for a user-specified value of $k$[19]. The points were ranked on the basis of this distance and top few were designated as outliers. The threshold in this second method is on the distance rank instead of the distance values[1]. Further computation of distances between all pairs of data points is computationally intensive unless indexing techniques are used[1]. The method proposed by Knorr et al. included a cell-based data-space partitioning method that is applicable for low-dimensional data and facilitates effective pruning[15].

CBOF is somewhat comparable to these distance-based methods because, like these other distance-based methods, CBOF also requires the calculation of distances between pairs of points. However, CBOF offers a novel use of these distances in evaluating the outlier-ness of data points.
2.2.2 Density-based Methods

Density-based methods consider the density of data in the locality of any given data point in question. The representative technique in this category is the Local Outlier Factor (LOF) proposed by Breunig et al. [9]. This technique considers the density of data around a data point and compares this to the density around other data points in its neighborhood. The method identifies the points that have unexpectedly low density in their neighborhoods as anomalies. This method has an advantage over global methods of anomaly detection that it can identify anomalies that are embedded within dense regions of the data. Such anomalies are ignored by global anomaly detection methods which consider the anomalous-ness over the whole data set. This method considers the outlier-ness of a data point within its locality and hence can detect anomalies that appear anomalous within their neighborhood [9].

CBOF can also be compared conceptually to density-based methods because implicitly this novel definition identifies the points that are quite far-removed from their neighbors and hence are located in low-density regions of the data. However, while LOF actively and explicitly considers the density around a given data point while considering its outlier-ness, for CBOF lower density around anomalies appears as a consequence of outlier-ness.
2.3 Preliminary Concepts

In this section we will describe a few preliminary concepts before we move on to describe Cohesiveness Based Outlier Factor.

2.3.1 k-Nearest Neighbors

Consider the section of 2-dimensional dataset as described in figure 2.1.

In this figure the point \( P \) is the data point whose \( k \)-nearest neighborhood is under investigation. To make the discussion concrete, let us say that we consider the case with \( k \) as the value 5. In other words, we are investigating the 5-nearest neighborhood of \( P \).

Breunig et al. have formally defined the \( k \)-distance of an object \( P \) as follows[9]:

**Definition 2.1.** For any positive integer \( k \), the \( k \)-distance of an object \( P \) in a dataset \( D \), is the distance \( d(P, O) \) between \( P \) and an object \( O \in D - \{ P \} \)
such that for at least $k$ objects $O'$, $d(P,O') \leq d(P,O)$, and for at most $(k - 1)$ objects $O' \in D - \{P\}$, $d(P,O') < d(P,O)$.

This definition essentially means that k-distance for a point is the least radius of the hyper-sphere centered at the point which contains at least $k$ data points. Breunig et al. proceed to define the $k$–distance neighborhood of $P$ as follows[9]:

**Definition 2.2.** $k$–distance neighborhood of a point $P$ contains every object whose distance from $P$ is not greater than $k$–distance of $P$.

Essentially, the k-distance neighborhood includes the ‘$k$’ nearest points from $P$. In case multiple data points tie for the $k^{th}$ place, all such points are considered to be the k-distance neighbors. The definition by Tan et al. captures the meaning of k-nearest neighbors more intuitively, while it does not make an attempt to address the case of multiple points in a tie[21]:

**Definition 2.3.** The $k$–nearest neighbors of a point $P$ refer to the $k$ points that are closes to $P$.

With these definitions in view, the dotted boundary in figure 2.1 represents the $k$-nearest neighborhood for data point $P$. For convenience of discussion, we have avoided the pathological case where multiple data points lie on the boundary.
2.3.2 Sum of Distances Between Nearest Neighbors Including P

The new measure of outlier-ness we are presenting in this dissertation depends on the computation of sum of distances between all pairs of points in the \(k\)-nearest neighborhood of the point \(P\), including the point \(P\). Specifically, for the section of data displayed in figure 2.2, we compute:

\[
\text{Define } S \leftarrow \{k \text{-nearest neighborhood of } P\} \cup \{P\}
\]

\[
n_1 = \sum_{p,q \in S} d(p, q)
\]

In other words, the figure \(n_1\) represents the sum of distances between all pairs of data points in the \(k\)-nearest neighborhood of the point \(P\) including the point \(P\) itself.
Figure 2.3: Sum of distances between all pairs of points in $k$-neighborhood of $P$ excluding $P$

2.3.3 Sum of Distances Between Nearest Neighbors Excluding $P$

The difference between this quantity and the one discussed above is that in this case we exclude the point $P$ from the set of data points within the $k$-nearest neighborhood. Consider the figure 2.3 to illustrate this.

This time we conduct the same computation, except that we remove the point $P$ from the set of points for which all-pairs distances are being computed:

$$
\text{Define } S' \leftarrow \{k \text{ – nearest neighborhood of } P\}
$$

$$
n_2 = \sum_{p,q \in S'} d(p,q)
$$
2.4 Cohesiveness-based Outlier Factor \((CBOF)\)

\(CBOF\) is a new metric of anomalous behavior that represents the extent to which the data point in question contributes to the sum of all-pairs distances between the data points. The premise of this metric of outlier-ness is that anomalous points will increase the sum of all-pairs distance in their neighborhood by a large fraction, as compared to the non-anomalous data points.

In other words, the non-anomalous data points have distances to their \(k\)-nearest neighbors that are very similar to the distance between the neighbors. We call this property of non-anomalous data points their cohesiveness, and say that non-anomalous data points have high cohesiveness.

On the other hand, the sum of distances from anomalous data points to other data points in the neighborhood has a higher proportion with respect to the distances between the neighbors of the anomalous data point. We say that non-anomalous data points have low cohesiveness.

The value of \(CBOF\), which we will describe next, captures a numerical measure of cohesiveness. A higher value of \(CBOF\) indicates a higher outlier-ness and hence lower cohesiveness. \(CBOF\) is computed as:

\[
CBOF = \left\{ 1 - \frac{n_2}{n_1} \right\}
\]

By definition, the value of \(CBOF\) is bounded by 0 from below and by 1 from above. Also, a value closer to 1 indicates higher outlier-ness whereas
a value closer to 0 indicates greater normalcy in the data point.

The figures and above describe the computation of \( CBOF \) value for a point \( P \) which falls within about equally-spaced neighbors. It is likely that such a point is non-anomalous. On the other hand, consider the case of data point \( P' \) as described in figure 2.4.

For such a point \( P' \), it is evident from the figure that the \( k \)-nearest neighbors are much closer to each other than to the point \( P' \). Consequently, the neighbors have much higher cohesion among each other than with \( P' \). For the same dataset, the computation of sum of distances only within the neighbors looks like the figure 2.5.

As is evident from the figures 2.5 and 2.4, the neighbors of \( P' \) have much higher cohesion among themselves than with the point \( P' \) included. Consequently, \( P' \) contributes a greater proportion of the sum of distances to all pairs of data points in figure 2.4 than the point \( P \) does in figure 2.4.
$CBOF$ is the numerical figure that captures and quantifies this effect.

When anomaly detection is performed over a high-dimensional dataset, a common practice is to look for anomalies in specific low-dimensional subspaces[3]. The outlier scores computed in these subspaces are then aggregated to make predictions about the outliers in the complete dataset. In such a scenario, it would help to have the outlier scores to stand-out in comparison to the outlier scores for non-anomalous points. Hence, if required, the $CBOF$ values can be scaled up to achieve this effect. The scaled-up values, represented by $CBOF'$ can be determined as:

$$CBOF' = (CBOF + f)^x$$

Here $f$ is a positive quantity and $x$ is a positive integer. Examples of values for these can be 0.5 and 2 respectively, and these are the values we have used for our experimental evaluation. It should be noted that $CBOF'$
is just a scaled-up version of \( CBOF \). Further, we emphasize the higher \( CBOF \) scores over larger neighborhoods by scaling the outlier scores by multiplying with \( n_1 \).

### 2.4.1 Algorithm Specification

Algorithm 2.1 provides the formal specification of the algorithm to compute \( CBOF \).

**Algorithm 2.1: CBOF Score Computation**

```plaintext
Input:

- \( k \): the number of nearest neighbors to consider
- \( D \): the dataset containing \( n \) records
- \( f \): the fraction to be used in scaling \( CBOF \)
- \( x \): the exponent to be used in scaling \( CBOF \)

Output: \( C[1..n] \)

1. for each point \( P \in D \) do
2.     Compute \( N_P \), the \( k \)-nearest neighborhood of \( P \)
3.     Generate set \( N'_P \leftarrow \{N_P \cup P\} \)
4.     Compute \( n_1 \), the sum of distances between all pairs of points in \( N'_P \)
5.     Compute \( n_2 \), the sum of distances between all pairs of points in \( N_P \)
6.     Compute \( CBOF \leftarrow \left(1 - \frac{n_2}{n_1}\right) \)
7.     Compute \( CBOF' \leftarrow (CBOF + f)^x \)
8.     Set \( C[P] \leftarrow CBOF' \times n_1 \)
9. end
```

Next we will discuss various aspects of this definition of anomalous behavior.
2.4.2 Scaling of CBOF

The CBOF values are immune to the scale of the data. CBOF only captures the ratio of the distances between the different data points in the neighborhood of any given point. Hence, the CBOF scores remain unaltered as a result of scaling the dataset. The CBOF scores in the following representative datasets illustrate this further.

These examples consist of datasets in two shapes, a square and a circle. For each shape, we created two datasets, with one dataset being the scaled version of the other. The figures also include the CBOF scores for that dataset. As is evident from the figures, both the square datasets have the same CBOF scores for all points. Similarly, for both the circle-shaped datasets, the scores of the points are the same.

Figure 2.6 represents a two-dimensional dataset containing four points. The values of the two dimensions are within the ranges of $-2$ and $2$. The datapoints are labeled with their corresponding CBOF scores. For calculating the CBOF scores, all the points in the dataset have been considered as neighbors. The four datapoints have been placed at each of the corners of a square. Hence the dataset is symmetric with respect to the data points and we expect the CBOF scores to be the same for all the points. This bears out in the CBOF scores computed for this datasets.

The figure 2.7 is derived from the earlier dataset by increasing the range of the two dimensions from $(-2, 2)$ to $(-5, 5)$. Again, the four data points are located at the four corners of a square. Hence the dataset is just a
Figure 2.6: Square Dataset Within Range (-2, 2)
Figure 2.7: Square Dataset Within Range (-5, 5)
Figure 2.8: Circular Dataset Within Range (-10, 10)

The data points are labeled with their $CBOF$ scores as before. As expected, the $CBOF$ scores for this scaled dataset are the same as for the previous dataset.

Figure 2.8 represents a two-dimensional dataset consisting of 10 data-points arranged in a circle. The circle is centered at $(0, 0)$ and has radius of 10. Hence, the two dimensions span a range of $(-10, 10)$. As with the square-shaped dataset above, we have considered all the points in the dataset as neighbors for calculating the $CBOF$ scores. Again, as the data-points are symmetrically arranged, we expect the $CBOF$ scores to be the same. This is borne out by the $CBOF$ values for the data-points which
Figure 2.9: Circular Dataset Within Range (-50, 50)

label the points in the figure.

Figure 2.9 represents a scaled version of the same two-dimensional dataset, with the two dimensions spanning the range \((-50, 50)\). Again, all the points in the dataset have been considered as neighbors while computing the \(CBOF\) values. The labels for the data points represent the computed \(CBOF\) values. Again, we observe that the \(CBOF\) scores for the datapoints in this scaled dataset are the same as the \(CBOF\) values in the original un-scaled dataset.
Figure 2.10: Linear Dataset Within Range (1, 10)
2.4.3 Effect of Dataset Shape on $CBOF$

The datasets discussed so far have been chosen to be symmetric and consequently all the data points exhibit the same $CBOF$ values. However, if the dataset has some points that are somewhat removed from the rest of the population, we would expect it to have a $CBOF$ score that reflects this property. Let us consider a dataset consisting of points laid-out along a straight line.

In such a dataset, the points along the edges are somewhat separated from the rest of the population. On the other hand, the points in the middle of the line very much belong within the population. We expect the $CBOF$ scores of the data point to reflect this ground truth.

Figure 2.10 depicts the linear dataset along with the $CBOF$ scores of the various points.

2.4.4 Comparison with $LOF$

There is no universal definition of outliers that can identify all possible anomalies. Each definition of anomalous behavior is based on an implicit assumption about what anomalous behavior means. Local Outlier Factor ($LOF$) is an established definition of anomalous behavior that focuses on varying density of anomalous points to identify outliers. It compares the density around a data-point under investigation with the density around its neighbors. If the density around a data point is unusually low as com-
pared to the density around its neighbors, the data-point is designated as an outlier. However, there can be instances where a data point is unusually different from the neighbors and still does not get flagged by LOF.

CBOF offers a new perspective into anomalous behavior, and can identify outliers that would not be detected by other methods of outlier detection such as LOF.

For example, let us consider two example datasets described in 2.11 and 2.12. In these figures, $p$ and $p'$ are the data points for which the outlier-ness is being investigated. In figure 2.11, the region $C$ represents the neighborhood of $p$ and regions $A$ and $B$ represent the neighborhoods of the
neighbors of \( p \). Similarly, in figure 2.12, the region \( C' \) represents the neighborhood of points \( p' \) and regions \( A' \) and \( B' \) represent the neighborhoods of the neighbors of \( p' \).

In these two datasets, the relative density of region \( C \) with respect to the regions \( A \) and \( B \) is the same as the relative density of region \( C' \) with respect to the regions \( A' \) and \( B' \). Consequently, the \( LOF \) score for \( p \) and \( p' \) in the two datasets would be comparable, and hence \( LOF \) will not distinguish between the outlier-ness of \( p \) and \( p' \).

However \( CBOF \) would recognize that the neighbors of \( p \) in the region \( C \) in figure 2.11 are evenly spread out. In figure 2.12, the neighbors of \( p' \) in the region \( C' \) are more compactly located. Hence, in this figure, the point \( p' \) contributes a greater proportion to the sum of distance between all points in the neighborhood. Correspondingly, the \( CBOF \) score of \( p' \) in figure 2.12 would be higher than the \( CBOF \) score of point \( p \) in figure 2.11.

Hence, in this example, \( CBOF \) will be more effective than \( LOF \) in identifying the anomalous behavior of point \( p \) as compared to that of point \( p' \).

### 2.4.5 Complexity Analysis

A fundamental operation in the computation of anomaly scores is the computation of distances between the data points. This has complexity of \( O(n^2) \) where \( n \) is the number of data points in the dataset where no indexing technique has been used.

Further, if \( k \) is the number of neighbors we are considering for comput-
ing the $CBOF$ scores, the time complexity of computing the $CBOF$ for any
given data point would be $O(k^2)$. However, $k$ is typically a small number
compared to $n$, the number of data points in the dataset. Also, $k$ is a fixed
number and does not vary as the size $n$ of the dataset scales. Hence the
computational complexity of the complete computation will be determined
by $O(n^2)$.

2.5 Experimental Evaluation

We performed experimental evaluation on synthetic datasets as well as a
few real datasets.

2.5.1 Synthetic Dataset

We first discuss the result of our experiment with the synthetic datasets that
represent the scenario described in figures 2.11 and 2.12.

Fig. 2.13 displays the $CBOF$ and LOF scores for a ‘compact’ dataset
where the neighbors of the point $P$ are located close together. Fig. 2.14 dis-
plays the scores for the dataset derived from compact dataset by spreading-
out the neighbors of the point $P$ while retaining the distances among the
neighbors. Hence in the compact as well as the spread-out datasets the
density among the neighbors is the same. For both $CBOF$ and LOF, the
value of $k$ used for both datasets was 4.

$CBOF$ scores of $P$ for the compact and spread-out datasets are 0.45 and
respectively whereas the LOF scores are 1.83 in the two cases. As expected, the LOF scores do not discriminate between the two cases whereas CBOF scores do bring-out the difference between them and assign higher score to $P$ in the compact dataset where $P$ is more ‘left-out’ with respect to its neighbors than the spread-out dataset.

Thus we see that as we claimed earlier, CBOF is able to differentiate between the compact and spread-out datasets, whereas LOF assigns the same outlier scores to $P$ in the two cases.

### 2.5.2 Real Datasets

We have used the following datasets from the UCI Machine Learning repository [5] to evaluate our method: Email Spam dataset, Sonar dataset, and Thyroid disease dataset.
Design of Experiment:

It is difficult to obtain datasets with known anomalies for use in evaluating anomaly detection methods. For example the UCI Machine Learning repository contains many datasets for use in classification and clustering etc, but none especially directed towards anomaly detection tasks. A commonly used approach is to use the datasets intended for use with classification tasks for anomaly detection too where records belonging to the smaller class are designated as outliers [2][14].

However, the anomaly detection task is fundamentally different from the classification task because the number of anomalies is very small as compared to the non-anomalous points. This renders the datasets intended for classification algorithm testing unsuitable for anomaly detection tasks.

For instance, consider the use of 2-class dataset depicted in fig. 2.15 for
evaluating an outlier detection algorithm, where records belonging to class B have been a-priori designated as outliers. We should not expect an outlier detection algorithm to report all the records from class B as outliers. The records belonging to class B might be outliers with respect to the records of class A, they appear non-anomalous with respect to other data points of class B. Consequently methods such as CBOF and LOF which examine the neighborhood of a data point while evaluating its outlier-ness will not flag them as outliers and thus fail to meet our expectations with such a dataset.

Hence a perfectly functional outlier detection algorithm will not report the records in class B as outliers, contrary to our a-priori expectation about
This problem arises because we are using an unsuitable dataset for the experiment. An appropriate evaluation dataset should have anomalies that are rare and isolated instead of being numerous and belonging to a well-defined class. With such a dataset we can reasonably expect the outlier detection method to detect the outliers and can then evaluate the methods on their efficacy.

Such a dataset can be obtained if we select a small number of data-points from class $B$ along with all the points of class $A$ and discard the un-selected records from class $B$. In this curated dataset (fig. 2.16), the records from class $B$ will appear rare and isolated, and we can expect the outlier-detection algorithm to detect them. We can expect the outlier detection algorithm to:

- Report all retained records of class $B$ as outliers, and
- Not report any record of class $A$ as an outlier.

If the outlier detection algorithm falls short on any of these goals with the curated dataset, it will be only because of the inherent limitations of the algorithm instead of the unsuitability of the dataset used for evaluation.

With this view, we created the curated datasets from Email Spam, Sonar and ANN Thyroid Disease datasets taken from the UCI Machine Learning Repository.
**Wisconsin Breast Cancer Dataset** : This dataset consists of 569 records, split into two classes of benign and malignant instances. The count of records for benign category is 357 while that for malignant category is 212. We created the curated dataset out of this by selecting 10 records from the malignant category and embedding them within the records for benign category. Use a uniformly distributed random number generator for this. Further, we repeated this process 10 times to generate 10 such curated datasets. Thus the final datasets consisted of 367 records.

**Spam Dataset** : This dataset consists of 4601 records, out of which 2788 records are for non-spam emails and 1813 records are for spam emails. We used a uniformly distributed random number generator to choose 10 records from the spam records and appended them to the set of non-spam records. Further, this was repeated 10 times to generate 10 different curated datasets.

**Thyroid Disease ANN Training Dataset** : The Thyroid Disease ANN training dataset consists of 3772 records belonging to three distinct classes. We chose to designate the class 3 as the ‘normal’ class. We then again used the uniformly distributed random number generator to choose 10 records from class 1 and appended them to the records of class 3 to generate the curated dataset. Again, this was repeated 10 times to generate 10 different curated datasets.

In all these cases we knew the labels for true outliers a-priori by the
Figure 2.17: Wisconsin Breast Cancer Dataset, $k = 5$

virtue of having created the curated dataset. The outlier scores were computed using both CBOF and LOF. The ROC curves were plotted for both cases and the corresponding area under curve (AUC) was computed using the 
\textit{pROC} package of the \textit{R} programming environment [20].

We conducted the experiment with all curated datasets obtained from the three real datasets mentioned earlier. The values of $k$ used were 5, 10, 15 and 20. To provide an overview of the results of all experiments, we report the average \textit{AUC} values for all of curated dataset for these values of $k$. In the interest of saving space we display the \textit{ROC} curves for only a few experiments.
Results for Wisconsin Breast Cancer Dataset:

Figure 2.17 displays the ROC curve for \( k = 5 \) for one curated Spam dataset. The AUC values for CBOF and LOF are 0.994 and 0.810 respectively.

Figure 2.18 displays the ROC curve for \( k = 10 \) for a different curated Spam dataset. The AUC values for CBOF and LOF are 0.965 and 0.892 respectively.

Figure 2.19 displays the ROC curve for \( k = 15 \) for another curated Spam dataset. The AUC values for CBOF and LOF are 0.962 and 0.883 respectively.

Figure 2.20 displays the ROC curve for \( k = 20 \) for another curated Spam dataset. The AUC values for CBOF and LOF are 0.983 and 0.907 respectively.

Table 2.1 represents the AUC scores obtained for various values of \( k \)
Figure 2.19: Wisconsin Breast Cancer Dataset, \( k = 15 \)

Figure 2.20: Wisconsin Breast Cancer Dataset, \( k = 20 \)
Table 2.1: Avg. *AUC* values for *CBOF* for Wisconsin Breast Cancer Dataset

<table>
<thead>
<tr>
<th>k</th>
<th>CBOF</th>
<th>LOF</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>0.979356</td>
<td>0.890589</td>
</tr>
<tr>
<td>10</td>
<td>0.979468</td>
<td>0.950477</td>
</tr>
<tr>
<td>15</td>
<td>0.977339</td>
<td>0.957031</td>
</tr>
<tr>
<td>20</td>
<td>0.975658</td>
<td>0.944314</td>
</tr>
</tbody>
</table>

Figure 2.21: Spam Dataset, *k* = 5

and averaged over all the curated datasets.

As is evident from these results, CBOF performs quite well in comparison to LOF for the Wisconsin Breast Cancer dataset.

**Results for Spam Dataset:**

Fig. 2.21 displays the *ROC* curve for *k* = 5 for one curated Spam dataset. For this experiment, the *AUC* values for *CBOF* and *LOF* are 0.902 and 0.628 respectively.
Fig. 2.22: Spam Dataset, $k = 10$

Fig. 2.22 displays the ROC curve for $k = 10$ for a different curated Spam dataset. For this experiment, the AUC values for CBOF and LOF are 0.910 and 0.637 respectively.

Fig. 2.23 displays the ROC curve for $k = 15$ for another curated Spam dataset. For this experiment, the AUC values for CBOF and LOF are 0.905 and 0.640 respectively.

Fig. 2.24 displays the ROC curve for $k = 20$ for yet another curated Spam dataset. For this experiment, the AUC values for CBOF and LOF are 0.910 and 0.657 respectively.

Table 2.2 represents the AUC scores obtained for various values of $k$ and averaged over all the curated datasets.

As is evident from these results, CBOF performs quite well in comparison to LOF for the Spam dataset too.
Figure 2.23: Spam Dataset, $k = 15$

Figure 2.24: Spam Dataset, $k = 20$
Table 2.2: Avg. AUC values for CBOF for Spam Dataset

<table>
<thead>
<tr>
<th>k</th>
<th>CBOF</th>
<th>LOF</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>0.805341</td>
<td>0.644301</td>
</tr>
<tr>
<td>10</td>
<td>0.803433</td>
<td>0.649344</td>
</tr>
<tr>
<td>15</td>
<td>0.803077</td>
<td>0.657794</td>
</tr>
<tr>
<td>20</td>
<td>0.803537</td>
<td>0.656969</td>
</tr>
</tbody>
</table>

Figure 2.25: Thyroid Disease ANN Training Dataset, $k = 5$

Results for Thyroid Disease ANN Training Dataset:

Figure 2.25 displays the ROC curve for $k = 5$ for one curated Thyroid Disease ANN training dataset. The AUC values for CBOF and LOF are 0.987 and 0.830 respectively.

Figure 2.26 displays the ROC curve for $k = 10$ for a different curated Thyroid Disease ANN training dataset. The AUC values for CBOF and LOF are 0.944 and 0.930 respectively.

Figure 2.27 displays the ROC curve for $k = 15$ for another curated
Figure 2.26: Thyroid Disease ANN Training Dataset, $k = 10$

Figure 2.27: Thyroid Disease ANN Training Dataset, $k = 15$
Figure 2.28: Thyroid Disease ANN Training Dataset, $k = 20$

Table 2.3: Avg. AUC values for CBOF for Thyroid Disease ANN Training Dataset

<table>
<thead>
<tr>
<th>$k$</th>
<th>CBOF</th>
<th>LOF</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>0.971809</td>
<td>0.931527</td>
</tr>
<tr>
<td>10</td>
<td>0.968501</td>
<td>0.973119</td>
</tr>
<tr>
<td>15</td>
<td>0.964157</td>
<td>0.980938</td>
</tr>
<tr>
<td>20</td>
<td>0.959146</td>
<td>0.981663</td>
</tr>
</tbody>
</table>

Thyroid Disease ANN training dataset. The AUC values for CBOF and LOF are 0.987 and 0.995 respectively.

Figure 2.28 displays the ROC curve for $k = 20$ for yet another curated Thyroid Disease ANN training dataset. The AUC values for CBOF and LOF are 0.951 and 0.985 respectively.

Table 2.3 represents the AUC values obtained for various values of $k$ and averaged over all curated datasets.

For the Thyroid Disease dataset, both CBOF and LOF perform quite well
and attain high $AUC$ values. CBOF edges out LOF for value 5 of $k$, while the reverse occurs for values 5, 10 and 15 of $k$. Overall, CBOF and LOF perform equally well for this dataset.

2.6 Conclusion

In this chapter we have presented a new metric to define outlier-ness of a data point. Some approaches consider distances from cluster centers as measures of outlier-ness [22] and some other methods assume a statistical model and consider deviations from the model as a measure of outlier-ness [6]. Many other methods use distance computations among the data points as the basis for detecting anomalies, such as the Z-score of distances to k-nearest neighbors [11].

$CBOF$, the novel metric described in this chapter captures the intent behind Hawkin's definition in a unique way by examining the distances within k-nearest neighborhood of a datapoint. Our extensive experiments demonstrate that CBOF captures facets of outlier-ness that elude $LOF$. This is exemplified by the results of experiments with the Spam dataset in Fig. 2.21 and table 2.2, and the results with Wisconsin Breast Cancer dataset in Fig. 2.17 and table 2.1.

However, experiments with the Thyroid Disease ANN training dataset depicted in fig 2.28 and table 2.3 demonstrate that there can be instances where outliers are capably identified by $LOF$ too, where it marginally
edges out $CBOF$. This observation, that perhaps there is no single universal metric of outlier-ness and outcomes from multiple definitions should be combined to produce a final decision about outliers, as has been proposed by Lazarevic et al. [17]. An anomaly detection scheme that uses both, the $CBOF$ and $LOF$ metrics to indicate the outlier-ness of a data point promises to be a more robust scheme. That is, a data point marked as an outlier based on either metric should be considered an anomaly.

The $LOF$ method was proposed to bring-out anomalies residing in low-density pockets within a high-density neighborhood. So if we expect a-priori that the dataset consists of such anomalies, $LOF$ would be a good metric of outlier-ness to be applied on such a dataset.

However, $CBOF$ will further differentiate the outlier-ness in neighborhoods that don’t necessarily contain variable density pockets, but have certain data points that are more geometrically isolated from their neighbors within the neighborhood. Hence if the dataset does not necessarily have variable-density regions, or if we don’t want to base our anomaly detection on only variable density but also on relative isolation of data points within their localities, $CBOF$ would be the appropriate technique to apply.

Nevertheless, $CBOF$ provides a unique perspective on what constitutes outlying behavior and in many scenarios is more capable than $LOF$ in detecting anomalies.
Chapter 3

Improved Cohesiveness-Based Outlier Factor (iCBOF)

In this chapter we will present an improvement over $CBOF$, the metric of outlier-ness we have presented in chapter 2. While $CBOF$ performs quite well compared to alternate anomaly detection methods, it is possible to further emphasize the discriminating nature inherent in $CBOF$ and thus obtain and even better metric of outlier-ness. We will describe such a method in this chapter.

3.1 Background

As discussed in chapter 2, unsupervised anomaly detection is an active area of research in data mining. Further, in that chapter we described a novel metric of outlier-ness called $CBOF$ and demonstrated its effectiveness. We
discussed scenarios in datasets where other outlier detection techniques such as LOF do not discriminate between the varying outlier-ness among datapoints. In the cases depicted in that chapter, CBOF successfully assigns outlier scores which discriminate between the outlier-ness among datapoints which is missed by LOF. Through experiments on real as well as synthetic datasets we provided evidence for effectiveness of CBOF.

However, this discriminatory property of CBOF can be further enhanced. In this chapter we will explore this idea and demonstrate the efficacy of this improvement through extensive experiments on real datasets.

Consider the scenarios described in figures 3.1 and 3.2 and assume that the value $k$ of the number of nearest neighbors used is 5. CBOF will assign higher outlier score to point $P$ in figure 3.2 than to the similarly labeled point in figure 3.1. Our goal is to increase the contrast between the outlier scores in these two cases.
We propose that instead of considering all the $k$ nearest neighbors in computing $CBOF$ values, we incrementally compute the $CBOF$ for 2 nearest neighbors, 3 nearest neighbors up to $k$ nearest neighbors and we add up these intermediate $CBOF$ values.

Let us discuss the case of 2 nearest neighbors. In figures 3.1 as well as 3.2, the points labeled $I$ and $II$ are the 2-nearest neighbors. Let us denote the $CBOF$ score of $P$ for this neighborhood as $CBOF_2$. In general, the $CBOF$ score of $P$ considering a neighborhood of $k$ nearest points can be denoted by $CBOF_k$. Clearly $CBOF_2$ of $P$ in figure 3.2 is greater than $CBOF_2$ of $P$ in figure 3.1.

Similarly, for all $i$ such that $2 \leq i \leq k$, the $CBOF_i$ of $P$ in figure 3.2 will be greater than $CBOF_i$ of $P$ in figure 3.1. If we accumulate all these $CBOF_i$ for various values of $i$, the accumulated $CBOF$ for $P$ in the two figures will have much stronger contrast than the vanilla $CBOF$ of $P$ in

Figure 3.2: $CBOF$ Score Computation for an Example Dataset
these two figures. We denote this improved $CBOF$ by the term $iCBOF$.

3.2 Related Work

Unsupervised anomaly detection in numerical datasets has been investigated in the data mining research community for a long time and the work by Knorr et al.[15] is an early representative work of this type. A drawback of this method is that it requires the user to input two parameters, the fraction $p$ of datapoints as well as $D$, a distance value and all points for which a fraction $p$ of the complete set of datapoints lies at distance $D$ or greater are declared as outliers. It is difficult for an analyst to come up with appropriate values of these parameters.

Ramaswamy et al. proposed an alternate anomaly detection method in which they ranked the datapoints with respect to their distances to the nearest neighbors and then selected the top $n$ datapoints as anomalies, where $n$ is an analysis parameter[19]. $Local \ Outlier \ Factor (LOF)$ is an outlier-detection method proposed by Breunig et al.[9] that compares the data density around a point with the density around neighbors and uses this to compute outlier scores. This has been mentioned in chapter 2. In that chapter we also proposed a novel outlier-ness metric called $CBOF$ and compared it with $LOF$. 
3.3 Our Approach

We will now formalize the method to compute $iCBOF$ that we introduced in an earlier section. In the computation of $iCBOF$, we will leverage the computation of $CBOF$ as described in algorithm 2.1 in chapter 2. Algorithm 3.1 describes the precise steps in the computation.

**Algorithm 3.1: iCBOF Score Computation**

**Input:**

- $k$: the number of nearest neighbors to consider
- $D$: the dataset containing $n$ records

**Output:** $C[1..n]$

1. for each point $P \in D$ do
2.   Compute $N_P$, the $k$-nearest neighborhood of $P$
3.   Sort $N_P$ in increasing distances from $P$
4.   Initialize $iCBOF \leftarrow 0$
5.   for $j \leftarrow 2$ to $k$ do
6.     $CBOF_j \leftarrow \{\text{Compute CBOF for point } P \text{ using } j \text{ nearest neighbors}\}$
7.     $iCBOF \leftarrow iCBOF + CBOF_j$
8.   end
9.   Set $C[P] \leftarrow iCBOF$
10. end

3.3.1 Computational Complexity

The computational complexity of computation of $iCBOF$ for any given value of $k$ is the same as the computational complexity of computing $CBOF$ for the same value of $k$. This is because the distances between any given pair of points, say $P_1$ and $P_2$, that was used for any iteration of $j$ in the
algorithm 3.1 can be reused to in the next iteration for \((j + 1)\). Hence, the distance between any two points \(P_1\) and \(P_2\) needs to be computed exactly once, which is the same as with computation of \(CBOF\).

Hence, the time complexity of \(iCBOF\) computation is \(O(n^2)\), following from the discussion in section 2.4.5.

### 3.4 Experimental Evaluation

We performed extensive experimental evaluation on multiple real datasets from the UCI Machine Learning Repository:

- Email Spam dataset
- Wisconsin Breast Cancer dataset
- Sonar dataset
- Musk dataset
- Thyroid Disease ANN training dataset

The Email Spam, Sonar and Thyroid Disease ANN training datasets have already been described in the previous chapter. We will briefly summarize the remaining datasets now.
Sonar dataset: This dataset consists of 208 records out of which 111 records belong to mines and 97 belong to rocks. Again a uniformly distributed random number generator was used to select 10 records from the class of rocks. These were appended to the class of mines to create the curated dataset. Again, this step was repeated 10 times to generate 10 different curated datasets.

Musk dataset: This dataset consists of 476 molecular conformations that were adjudged by humans as being musk molecules or not. The category of musk molecules consists of 207 instances while that for non-musk molecules consists of 269 instances. Again, we created a set of 10 curated datasets by using uniformly distributed random number generator. We selected 10 records from the set of non-musk records and embedded them within the records for musk molecules.

3.4.1 Results for Sonar dataset

Fig. 3.3 displays the ROC curve for $k = 20$ for one curated Sonar dataset. As is evident from this figure, $iCBOF$ out-performs $CBOF$ as well as $LOF$, as the $AUC$ values for the three are 0.66757, 0.62973 and 0.60991 respectively for this curated dataset.

Table 3.1 represents the $AUC$ scores obtained for various values of $k$ and averaged over all the curated datasets created from the Sonar dataset. For this dataset, $iCBOF$ slightly improves upon the results for $CBOF$ for
Figure 3.3: Sonar dataset, \( k = 20 \)

Table 3.1: Avg. AUC values for \( iCBOF \) for Sonar dataset

<table>
<thead>
<tr>
<th>( k )</th>
<th>( iCBOF )</th>
<th>( CBOF )</th>
<th>( LOF )</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>0.775946</td>
<td>0.771351</td>
<td>0.832613</td>
</tr>
<tr>
<td>10</td>
<td>0.718649</td>
<td>0.676937</td>
<td>0.708468</td>
</tr>
<tr>
<td>15</td>
<td>0.683604</td>
<td>0.649729</td>
<td>0.653514</td>
</tr>
<tr>
<td>20</td>
<td>0.664054</td>
<td>0.634505</td>
<td>0.629009</td>
</tr>
</tbody>
</table>
all values of $k$. However, for $k = 5$, LOF outperforms both $iCBOF$ and $CBOF$. For all other values for $k$, both $iCBOF$ and $CBOF$ perform better than LOF.

### 3.4.2 Results for Email Spam dataset

Fig. 3.4 displays the ROC curve for $k = 15$ for one curated Email Spam dataset. For this dataset, $iCBOF$ again outperforms $CBOF$, but only marginally. However, both, $iCBOF$ and $CBOF$ outperform LOF. The $AUC$ values for the three are 0.683, 0.675 and 0.643 for this specific curated dataset.

Table 3.2 represents the $AUC$ scores obtained for various values of $k$ and averaged over all the curated datasets created from the Email Spam dataset. For this dataset, $iCBOF$ slightly improves upon the results for
Table 3.2: Avg. AUC values for iCBOF for Email Spam dataset

<table>
<thead>
<tr>
<th>$k$</th>
<th>iCBOF</th>
<th>CBOF</th>
<th>LOF</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>0.806463</td>
<td>0.805341</td>
<td>0.644301</td>
</tr>
<tr>
<td>10</td>
<td>0.804749</td>
<td>0.803433</td>
<td>0.649344</td>
</tr>
<tr>
<td>15</td>
<td>0.803555</td>
<td>0.803077</td>
<td>0.657794</td>
</tr>
<tr>
<td>20</td>
<td>0.804164</td>
<td>0.803537</td>
<td>0.656969</td>
</tr>
</tbody>
</table>

Figure 3.5: Wisconsin Breast Cancer dataset, $k = 15$

CBOF for all values of $k$. Further, for all values of $k$, both iCBOF and CBOF outperform LOF.

3.4.3 Results for Wisconsin Breast Cancer dataset

Fig. 3.5 displays the ROC curve for $k = 15$ for one curated Wisconsin Breast Cancer dataset. For this dataset, iCBOF outperforms CBOF as well as LOF. The AUC values for the three are 0.964, 0.956 and 0.947 for this specific curated dataset.
Table 3.3: Avg. $AUC$ values for $iCBOF$ for Wisconsin Breast Cancer dataset

<table>
<thead>
<tr>
<th>$k$</th>
<th>$iCBOF$</th>
<th>$CBOF$</th>
<th>$LOF$</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>0.978683</td>
<td>0.979356</td>
<td>0.890588</td>
</tr>
<tr>
<td>10</td>
<td>0.979692</td>
<td>0.979468</td>
<td>0.950476</td>
</tr>
<tr>
<td>15</td>
<td>0.979244</td>
<td>0.977339</td>
<td>0.957031</td>
</tr>
<tr>
<td>20</td>
<td>0.977759</td>
<td>0.975658</td>
<td>0.944314</td>
</tr>
</tbody>
</table>

Figure 3.6: Musk dataset, $k = 5$

Table 3.3 represents the $AUC$ scores obtained for various values of $k$ and averaged over all the curated datasets created from the Wisconsin Breast Cancer dataset. For this dataset, $iCBOF$ outperform $CBOF$ as well as $LOF$. 
Table 3.4: Avg. AUC values for iCBOF for Musk dataset

<table>
<thead>
<tr>
<th>$k$</th>
<th>iCBOF</th>
<th>CBOF</th>
<th>LOF</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>0.899275</td>
<td>0.894734</td>
<td>0.906498</td>
</tr>
<tr>
<td>10</td>
<td>0.8743</td>
<td>0.8543</td>
<td>0.918599</td>
</tr>
<tr>
<td>15</td>
<td>0.849855</td>
<td>0.814541</td>
<td>0.912609</td>
</tr>
<tr>
<td>20</td>
<td>0.830048</td>
<td>0.781787</td>
<td>0.902995</td>
</tr>
</tbody>
</table>

3.4.4 Results for Musk dataset

Fig. 3.6 displays the ROC curve for $k = 15$ for one curated Musk dataset. For this dataset, iCBOF outperforms CBOF as well as LOF. The AUC values for the three are 0.920, 0.906 and 0.880 for this specific curated dataset.

Table 3.4 represents the AUC scores obtained for various values of $k$ and averaged over all the curated datasets created from the Musk dataset. For this dataset, iCBOF outperform CBOF for all the values. However, LOF performs better than iCBOF as well as CBOF for all these cases. This observation underscores our earlier conclusion from chapter 2 that there is no single universal definition of outlier-ness that will out-perform all other definitions.

3.4.5 Results for Thyroid Disease ANN Training dataset

Fig. 3.7 displays the ROC curve for $k = 5$ for one curated Thyroid Disease ANN Training dataset. For this dataset, iCBOF again outperforms CBOF as well as LOF. The AUC values for the three are 0.990, 0.989 and 0.918.
Table 3.5: Avg. AUC values for iCBOF for Thyroid Disease ANN Training dataset

<table>
<thead>
<tr>
<th>(k)</th>
<th>(iCBOF)</th>
<th>(CBOF)</th>
<th>(LOF)</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>0.971316</td>
<td>0.971809</td>
<td>0.931527</td>
</tr>
<tr>
<td>10</td>
<td>0.970459</td>
<td>0.968501</td>
<td>0.973119</td>
</tr>
<tr>
<td>15</td>
<td>0.967873</td>
<td>0.964157</td>
<td>0.980838</td>
</tr>
<tr>
<td>20</td>
<td>0.964702</td>
<td>0.959146</td>
<td>0.981663</td>
</tr>
</tbody>
</table>

respectively. However, the performance of \(iCBOF\) and \(CBOF\) is almost the same.

Table 3.5 represents the AUC scores obtained for various values of \(k\) and averaged over all the curated datasets created from the Thyroid Disease ANN Training dataset. For this dataset, \(iCBOF\) outperforms \(CBOF\). However, \(LOF\) performs better than \(iCBOF\) as well as \(CBOF\).
3.5 Conclusion

In this chapter we have presented $iCBOF$, an improvement over $CBOF$ the novel metric of outlier-ness that we presented in an earlier chapter. Extensive experiments with multiple real datasets demonstrate the efficacy of the new method. The $ROC$ curves for $iCBOF$, while providing better $AUC$ than $CBOF$, closely follow the shape of $ROC$ curves for $CBOF$. This is somewhat expected observation because $iCBOF$ follows the similar heuristic as $CBOF$. Further, $iCBOF$ excels in situations where $CBOF$ excels, and is bettered by $LOF$ in situations where $LOF$ out-performs $CBOF$ too.

The observation that there are datasets where $LOF$ outperforms $iCBOF$ and $CBOF$, such as the Thyroid Disease ANN Training dataset described here underscore the general conclusion that there is not single universal definition of outlier-ness that captures all possible scenario of anomalous behavior. Different definitions suit different data situation and, as has been observed in established literature for anomaly detection, the best strategy is to combine the results from different metrics of outlier-ness[17].

Analogous to $CBOF$, $iCBOF$ will be suitable for datasets that don’t have variable density regions that identify outliers. It will also be applicable for datasets where we don’t want to depend only on density variation for detecting outliers, because we have demonstrated that there are certain outliers that exist in neighborhoods with similar densities, and yet are
geometrically isolated-enough from neighbors to be identified as outliers. For such data analysis scenarios, both $iCBOF$ and $CBOF$ will be better methods than $LOF$. 
Chapter 4

eSelect: Effective Subspace Sampling for Outlier Detection in High Dimensional Data

Anomaly detection is an important data mining task and is used for many applications such as detection of credit card fraud, medical diagnosis and computer system intrusion detection. Many interesting real-world data sets are high dimensional. Detection of anomalies in such high dimensional datasets is hampered due to the curse of dimensionality. Hence searching for anomalous data points using all attributes is neither efficient nor fruitful. In such datasets the anomalies are hidden in smaller subspaces of attributes. However the number of subspaces possible from a given attribute set increases in a combinatorial fashion. Consequently an exhaus-
tive search through all possible subspaces for anomalies is not computationally feasible. A lot of research has focused on efficiently determining subspaces within which outliers must be searched. In this chapter we present a method for exploring the subspaces in a very high dimensional data set in an efficient and organized way to detect anomalies within them while avoiding an exhaustive search over all possible subspaces. The new method is called eSelect. Through extensive experimentation we compare eSelect to a well-established subspace selection method and demonstrate that our newly presented method attains marked improvements.

4.1 Background

Anomaly detection is an important problem within data mining with diverse applications such as credit card fraud detection, computer system intrusion detection and medical diagnosis [10]. There have been many approaches to anomaly detection including the statistical, distance based, information theory based etc[10].

The increase in the sizes and dimensionality of datasets has introduced new challenges about detecting anomalies. The existing methods of anomaly detection do not perform well when directly applied to high dimensional datasets because of the curse of dimensionality[3]. In this chapter we present a new technique for effectively searching for anomalies in high-dimensional datasets. Through extensive experiments we demonstrate that our tech-
nique performs better than other methods such as the subspace sampling method proposed by Lazarevic[17].

Our method builds upon the foundations laid by Aggarwal et al.[3] and Lazarevic et al.[17]. Our specific contributions in this chapter are:

1. A subspace sampling algorithm that selects subspaces that are effective in discovering anomalies.

2. Not require the analyst to provide input parameters that are difficult to know \textit{a-priori}.

3. Allow the analyst to adapt the computational overhead according to the resources available, instead of simply failing to terminate in the face of very high-dimensional data.

Our method improves upon the method proposed by Aggarwal et al.[2] by not requiring the analyst to supply a parameter $\phi$ that specifies the number of \textit{equi-depth} portions into which each attribute should be divided. It is difficult to know the correct value for this parameter \textit{apriori} and hence eliminating the need for such a parameter is one of the goals of our method.

Further, we also aim at not basing our method on arbitrary parameters that, which reducing the attribute search space, cannot be universally justified. For example, Nguyen et al.[18] proposed an anomaly detection method in which they have implicitly assumed the value of $\phi$ from Aggarwal's work to be 10. Choosing a high value of this parameter helps reduce the search space. However in cases where this is not a suitable value, the
search space is in-justifiably pruned. The goal of our method is to not prune the search space without justification, and instead provide the analyst with parameters that they can use to control the computational overhead to suite their computational resources.

Hence our contribution in this chapter is an anomaly detection method for high-dimensional data that eliminates these shortcomings.

4.2 Related Work

It has been observed that anomaly detection techniques developed for low dimensional datasets lose effectiveness with high dimensional datasets because of the curse of dimensionality[3]. Statistical techniques were among the earliest attempts at detecting anomalies in datasets[10]. These techniques depend on the assumption of a statistical model to represent the complete dataset. They identify outliers as datapoints that don’t conform to this assumed model for the complete dataset. These statistical techniques don’t scale to high dimensional datasets because of the computational complexity associated with determining the correct statistical model and choosing its parameters to effectively model the data[10]. The proximity-based methods also tend to lose meaning because the concept of distance loses meaning in high-dimensional datasets[3].

Aggarwal et al. were among the early researchers to explore the problem of anomaly detection in very high dimensional datasets. They observed
that in very high dimensional dataset all points seem almost equally distant with no clear outliers. Instead, the real outliers are embedded in subspaces formed by subsets of attributes. Hence the use of subspaces for anomaly detection has been proposed, analogous to the use of subspaces for data clustering[3].

However, the number of subspaces available for a given set of attributes is a combinatorial number and increases at a very rapid rate as the number of attributes increases. Consequently, an exhaustive search through all possible subspaces in a high-dimensional dataset is not computationally feasible. Aggarwal et al. have termed this situation as searching for needle within a haystack when the number of haystacks itself is exponential[1].

To tackle this problem of exploring a very large number of subspaces, Lazarevic et al. proposed the method of subspace sampling[17]. Here the idea is to randomly sample subspaces of sizes varying from \( \lfloor d/2 \rfloor \) to \((d - 1)\) where \(d\) represents the total number of attributes in the dataset. It has been noted that unlike association rule mining, the outlier-ness of a datapoint is not upward or downward closed with respect to the set of subspaces. Hence, the exploration of outlier-ness of a datapoint in a given subspaces does not provide any guidance about its outlier-ness in any other subspace in the complete lattice of subspaces[1]. The subspace sampling method proposed by Lazarevic et al. implicitly acknowledges this and resorts to randomly sampling the subspaces to search for outliers[1].

Aggarwal and Yu have investigated the question of the number of at-
tributes of a dataset that need to be examined in order to determine the anomalies embedded in the subspaces of attributes[3][2]. They argued that for an input parameter $\phi$, the number of attributes that need to be examined is $\lfloor \log_{\phi} (N/s^2 + 1) \rfloor$. Here the value of parameter $\phi$ needs to be judiciously chosen. This parameter represents the number of portions into which each attribute is divided. These portions are not equal-sized. Instead, the sizes of portions are chosen to each contain the same number of data points. The authors use the term *equi-depth* to describe these ranges. These regions across different attributes divide the complete data space into small cubes. The proposed method then uses evolutionary algorithms to find the cubes that have abnormally low density of data. Choosing too large a value of $\phi$ is counterproductive because it would cause most data cubes to contain too small a number of data points to yield a meaningful conclusion about outliers within them. Choosing a smaller value would increase the number of attributes that would need to be included in the search for sparsely populated data cubes and hence would increase the search space. Choice of the correct value would depend on the analyst’s insights about the dataset.

Nguyen et al. implicitly choose the value of $\phi$ as 10 and explore all the subspaces of sizes from 1 to $\lfloor \log_{10} N \rfloor$ where $N$ represents the number of records in the dataset[18]. From the context of Aggarwal’s proposal, this means that each attribute is always divided into 10 equi-distant regions. However, Nguyen does not provide a basis for this choice, and it seems
somewhat arbitrary. Further, for large datasets, this value can cause a very large number of subspaces to be explored and the algorithm does not provide any method to control the computational overhead. For sufficiently large datasets, the method may fail to terminate in a reasonable length of time, and the analyst does not have any control over this.

These related works provide a platform for the method we present in this chapter. We are presenting a new method of sampling subspaces based on the theoretical consideration provided by Aggarwal[3]. Our subspace sampling method performs better than the Lazarevic’s method of subspace sampling.

The new method is based on the hypothesis that low-dimensional subspaces are more useful than high dimensional subspaces in the search for outliers. Therefore the computational resource should be preferably spent exploring low-dimensional subspaces as compared to the high-dimensional subspaces as compared to the high-dimensional subspaces. The results of experiments conducted thoroughly vindicate our faith in this hypothesis and the new method performs substantially better than Lazarevic’s subspace sampling method.

4.3 Our Approach

We start from Aggarwal’s conclusion that the number of attributes that need to be examined to detect anomalies is \( \lceil \log_{\phi} \left( N/s^2 + 1 \right) \rceil \), where \( \phi \) is an in-
put parameter. However, we don’t depend on the analyst to supply the appropriate value of $\phi$. Unlike Nguyen’s method[18] we set it to 2, which results in the maximum number of attributes to be searched. This avoids arbitrarily restrictions on the search space, as would happen if we choose any higher arbitrary value of $\phi$. Hence we use $\lceil \log_2 N \rceil$ as the maximum size of the subspaces that need to be searched for anomalies. Then we sample subspaces from the set of subspaces of sizes from 1 and $\lceil \log_2 N \rceil$.

Further, we use the heuristic that outliers are more likely to be discovered in low-dimensional subspaces rather than in high-dimensional subspaces in the subspace lattice. Consequently, when we sample the subspaces of sizes between 1 and $\lceil \log_2 N \rceil$, we choose a greater fraction of low dimensional subspaces than the high-dimensional subspaces.

In this chapter we focus on the exploration of subspaces for detection of outliers within them. Our method can accommodate any technique of outlier detection that generates outlier scores. Like Lazarevic’s proposal, we collect the outlier scores for all records across various sample subspaces. Then we use the cumulative sum approach described by Lazarevic et al.[17] to draw conclusions about the outliers detected in various subspaces.

Our method of searching for outliers in different subspaces does not depend on any specific definition of outlier-ness. We only require that the definition of outlier-ness generate outlier scores that signify the extent of outlying behavior demonstrated by records in the dataset. For our discussion in this chapter we have used the \textit{Local Outlier Factor} (LOF)[9] as
the definition of outliers. Any other definition generating the outlier scores such as the z-score of distances to the nearest neighbors[11] would also serve our purpose and we have used this definition in one instance to support this point.

Once the outlier scores have been computed, the analyst can apply thresholds to the score and select the records whose cumulative outlier scores exceed the threshold. This step is similar to the application of thresholds to the $LOF$ scores to identify the outliers.

To control the computational complexity, we introduce two user defined parameters: $c$, which is a number between 1 and $\lceil \log_2 N \rceil$. While choosing subspaces and computing outliers within them, we terminate the process when all subspaces of sizes 1 to $c$ have been already selected. We will use $n$ as a short notation to represent the quantity $\lceil \log_2 N \rceil$ in our discussion from now onwards.

Let $D$ denote the complete dimensionality of the dataset, and $N$ denote the number of records in the dataset.

It should be noted that the exhaustive search through all the subspaces of sizes between 1 and $\lceil \log_2 N \rceil$ can also be computationally intensive. This step can be scaled down to suit the computational resources that the analyst is able to offer. We choose the input parameter $c$ such that instead of exploring all the subspaces of sizes between 1 and $n$, we terminate when all subspaces of sizes from 1 to $c$ have been explored.

It should be noted that the number of subspaces with 1 attribute will
Algorithm 4.1: Cumulative Outlier Score Computation

Input:

$DS$: data set

$N$: number of records in the dataset $D$

Output:

$C[1..n]$: cumulative outlier scores of each record in the dataset.

1. initialize $n \leftarrow \lceil \log_2 N \rceil$
2. allocate array $M[1..n]$
3. allocate array $P[1..n]$
4. allocate array $C[1..n]$
5. for $i \leftarrow 1$ to $n$ do
   6. initialize $M[i] \leftarrow \binom{i}{2}$
   7. initialize $P[i] \leftarrow 0$
   8. initialize $C[i] \leftarrow 0$
6. end
7. while true do
   8. continue_flag $\leftarrow$ false
   9. for $i \leftarrow 1$ to $n$ do
      10. if $P[i] < M[i]$ then
          11. continue_flag $\leftarrow$ true
          12. break
     13. end
   14. end
   15. if continue_flag is true then
     16. $r \leftarrow \{\text{uniformly distributed random number between 1 and } n\}$
     17. while $P[r] == M[r]$ do
     18.       $r \leftarrow (r + 1)$
     19. end
     20. Choose a subspace $S_r$ of size $r$ that
     21. has not been chosen before.
     22. $P[r] \leftarrow P[r] + 1$
     23. Compute LOF outlier scores of all records
     24. in the dataset in subspace $S_r$.
     25. Add outlier scores for each record $[i]$ to its cumulative outlier score in $C[i]$.
   26. end
   27. else
     28. break
   29. end
30. end
be smaller than the number of subspaces with 2 attributes, which in turn will be smaller than the number of subspaces with 3 attributes and so on. In a general case, if \( D \) represents the dimensionality of the dataset, then till \( D/2 \) attributes, the number of subspaces will increase as the number of attributes is increased. Consequently, when we choose subspaces by first selecting the number of attributes based on the uniformly distributed random number, we will end up exhausting all subspaces of size 1 before we exhaust all subspaces of size 2. Similarly, we will use up all subspaces of size 2 before we use up all subspaces of size 3, and so on. Hence, we will end up exhausting all subspaces of size \((d - 1)\) before we exhaust all subspaces of size \( d \).

We can use the input parameter \( 1 \leq c \leq n \) that we described before to indicate that we intend to stop the computation once all subspaces of size up to \( c \) have been explored. Hence the analyst can use this parameter to terminate the search for outliers if the complete search through all subspaces of size up to \( n \) is more than the computational resources available.

We would like to observe that restricting the size of subspaces searched to \( n \) is only a heuristic to make the problem of searching through all possible subspaces computationally tractable. However, it does not absolutely guarantee that all outliers have been found. In fact, such a guarantee is not possible without performing exhaustive search through the complete subspace lattice because anomalous behavior in the lattice does not demonstrate upward or downward closed properties[1].
It is possible to further restrict the computational resources required by introducing another input parameter $f$ which represents the fraction of subspaces of sizes from 1 to $c$ that needs to be fully explored.

Putting these restrictions in place reduces the thorough-ness of the subspace search process with a possibility that some outliers escape detection, or at least don’t score as high as they should in the cumulative outlier-scores. On the other hand, they provide the analyst with control over the computational resources that the analyst will consume. This is a trade-off that the analyst has to make while analyzing very high dimensional datasets.

### 4.3.1 Computational Complexity

The number of subspaces of size from 1 to $n$ is

\[ O \left( n \times \left( \frac{D}{n} \right) \right) \]

where $D$ is the total number of dimensions in the complete dataset. Hence, the worst case computational complexity of the process of exploring subspaces also is the same. This represents only the complexity of the search through the subspaces. Complexity of the specific definition of anomalous behavior will vary from one method to another.
4.4 Experimental Evaluation

We have compared our method of subspace sampling with the Lazarevic’s method[17]. We have used the following datasets from the UCI Machine Learning repository[5] for comparison:

- Musk dataset
- Spam dataset
- Wisconsin Breast Cancer (Diagnostic) dataset

These datasets comprise of two classes each and are intended to be used for classification tasks. We created the curated datasets for each of these, as explained in the previous chapters. For each of these datasets, we obtained 10 curated datasets, which were obtained by selecting 10 records from one class and embedding these records within the records belonging to the other class.

4.4.1 Results for Musk dataset

Fig. 4.1 demonstrates the results obtained for a curated dataset obtained from the musk dataset. The definition of outlier-ness chosen to search for outliers in the Local Outlier Factor (LOF)[9]. The number of nearest neighbors chosen for LOF is 5. The subspaces for eSelect were chosen per the algorithm mentioned earlier till all subspaces up-to 1 attributes were
selected. Following this, an equal number of subspaces were chosen for the subspace sampling method suggested by Lazarevic. As is evident from the figure, the \( AUC \) for \( eSelect \) is 0.857 whereas the \( AUC \) for Lazarevic’s method is 0.673. This shows that the subspaces picked by \( eSelect \) are more effective in identifying outliers and are a better choice than the subspaces picked up by Lazarevic’s method.

Fig. 4.2 demonstrates the comparison for a different curated dataset for Musk dataset when the number of nearest neighbors chosen was 10, and subspaces were chosen till all the subspaces of size 1 were consumed. The definition of outliers used is again \( LOF \). The \( AUC \) for \( eSelect \) is 0.829, whereas the \( AUC \) for Lazarevic’s method is 0.735.
Figure 4.2: Musk Dataset, $k = 10$, $attrib = 1$

Figure 4.3: Musk Dataset, $k = 15$, $attrib = 1$
Fig. 4.3 demonstrates the comparison for another curated dataset for Musk dataset when the number of nearest neighbors chosen was 15, and subspaces were chosen till all the subspaces of size 1 were consumed. The definition of outliers used is again $LOF$. The $AUC$ for $eSelect$ is 0.835, whereas the $AUC$ for Lazarevic’s method is 0.561.

Fig. 4.4 demonstrates the comparison for yet another curated dataset for Musk dataset when the number of nearest neighbors chosen was 20, and subspaces were chosen till all the subspaces of size 1 were consumed. The definition of outliers used is again $LOF$. The $AUC$ for $eSelect$ is 0.879, whereas the $AUC$ for Lazarevic’s method is 0.584.

Comparing the results in figures 4.1, 4.2, 4.3 and 4.4 provide a useful
Table 4.1: Avg. $AUC$ values for $eSelect$ for Musk dataset

<table>
<thead>
<tr>
<th>$k$</th>
<th>$eSelect$</th>
<th>Lazarevic's Method</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>0.894251</td>
<td>0.861159</td>
</tr>
<tr>
<td>10</td>
<td>0.891401</td>
<td>0.849517</td>
</tr>
<tr>
<td>15</td>
<td>0.863575</td>
<td>0.710531</td>
</tr>
<tr>
<td>20</td>
<td>0.805217</td>
<td>0.630242</td>
</tr>
</tbody>
</table>

Insight into the nature of subspaces selected by $eSelect$. This is that $eSelect$ narrows-down onto the most meaningful subspaces for outlier detection much more effectively than Lazarevic's method.

It should be noted that the results in figures 4.1, 4.2, 4.3 and 4.4 can be compared despite using different values of $k$ in the different cases. This is because $eSelect$ and Lazarevic's method are using the same definition of anomalies, and hence any differences between the two methods within the same figure are only because of the choices made about the subspaces. So, while the absolute values of $AUC$ in the two experiments depend also on the value of $k$, the difference in $AUC$ values for the two methods for an experiment is only due to the quality of subspaces analyzed.

We summarize the average $AUC$ values for the different curated datasets obtained from Must dataset for different values of $k$ in table 4.1. As this table demonstrates, $eSelect$ consistently out-performs the Lazarevic’s method of subspace selection for the Musk dataset, for various values of $k$, the number of nearest neighbors.
4.4.2 Results for Spam dataset

Figure 4.5 demonstrates the results for one specific curated dataset obtained from the Email Spam dataset. The definition of outlier-ness chosen for this experiment is again LOF. The value of $k$ chosen for nearest neighbors is 5. Subspaces were sampled till all subspaces consisting of 1 attribute got selected. The $AUC$ for $eSelect$ is 0.967 whereas the $AUC$ for Lazarevic’s method is 0.633.

Figure 4.6 demonstrates the results for another curated dataset obtained from the Email Spam dataset. The definition of outlier-ness chosen for this experiment is again LOF. The value of $k$ chosen for nearest neighbors is 10. Subspaces were sampled till all subspaces consisting of 1
attribute got selected. The $AUC$ for $eSelect$ is 0.905 whereas the $AUC$ for Lazarevic's method is 0.512.

Figure 4.7 shows the results for another curated dataset for $k = 15$ and subspaces sampled till all subspaces with one attribute got picked. Again, the metric of outlier-ness selected is $LOF$. The $AUC$ for $eSelect$ is 0.898 whereas the $AUC$ for Lazarevic's method is 0.611.

Figure 4.8 demonstrates the results for the same curated dataset and $k = 20$ using $LOF$, again selecting subspaces will all subspaces of size 1 got picked. The $AUC$ for $eSelect$ for this experiment is 0.849 whereas the $AUC$ for Lazarevic's method is 0.564.

Table 4.2 summarizes the averages $AUC$ values for the different curated
Figure 4.7: Spam Dataset, $k = 15$, $attrib = 1$

Figure 4.8: Spam Dataset, $k = 10$, $attrib = 1$
datasets obtained from the Email Spam dataset, for different values of $k$. Again, it is evident that $eSelect$ consistently outperforms the Lazarevic’s method in all these cases.

### 4.4.3 Results for Wisconsin Breast Cancer dataset

Figure 4.9 depicts the results one specific curated dataset for the Wisconsin breast cancer diagnostic dataset. $LOF$ was used as the definition of outlier-ness. The value of $k$ used for k-nearest neighbors is 5. Subspaces were sampled till all subspaces consisting of one attribute got picked up. The $AUC$ of $eSelect$ method is 0.950 whereas the $AUC$ for Lazarevic’s method is 0.891.

Figure 4.10 depicts the results for another curated dataset for the Wisconsin breast cancer dataset with $k = 10$. The definition of outlier-ness used for $LOF$. The $AUC$ for $eSelect$ is 0.942 whereas the $AUC$ of Lazarevic’s method is 0.885.

Figure 4.11 depicts the results of a different curated dataset for the Wisconsin breast cancer dataset with $k = 15$. The definition of outlier-ness used was $LOF$. The $AUC$ for $eSelect$ is 0.992 whereas the $AUC$ for
Figure 4.9: Wisconsin Breast Cancer Dataset, $k = 5$, $attrib = 1$

Figure 4.10: Wisconsin Breast Cancer Dataset, $k = 10$, $attrib = 1$
Lazarevic’s method is 0.975.

Finally, figure 4.12 depicts the results of yet another curated dataset for the Wisconsin breast cancer dataset with $k = 20$. The definition of outlier-ness used was \textit{LOF}. The \textit{AUC} for \textit{eSelect} is 0.998 whereas the \textit{AUC} for Lazarevic’s method is 0.909.

Table 4.3 summarizes the averaged \textit{AUC} values for the different cu-

**Table 4.3: Avg. \textit{AUC} values for \textit{eSelect} for Wisconsin Breast Cancer dataset**

<table>
<thead>
<tr>
<th>$k$</th>
<th>\textit{eSelect}</th>
<th>Lazarevic’s Method</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>0.941092</td>
<td>0.939608</td>
</tr>
<tr>
<td>10</td>
<td>0.949692</td>
<td>0.927619</td>
</tr>
<tr>
<td>15</td>
<td>0.975602</td>
<td>0.975434</td>
</tr>
<tr>
<td>20</td>
<td>0.962969</td>
<td>0.947115</td>
</tr>
</tbody>
</table>

85
Figure 4.12: Wisconsin Breast Cancer Dataset, $k = 20$, $attrib = 1$

rated datasets for Wisconsin Breast Cancer dataset for various values of $k$. Again, the table provides evidence that $eSelect$ performs better than the Lazarevic’s subspace selection method.

Thus we observe that the Wisconsin breast-cancer dataset has very cleanly separable outliers and $eSelect$ as well as Lazarevic’s method perform very well. However, $eSelect$ does marginally out-perform Lazarevic’s method on all the experiments described here.

However for the Spam dataset the outliers are not very cleanly separable from the normal data records, and that is why the $AUC$ for anomaly detection is not as high as it is for the Wisconsin breast-cancer dataset. Irrespective of this, $eSelect$ is much more effective than Lazarevic’s method.
in picking subspaces that contain outliers and its $AUC$ is consistently better than the $AUC$ for Lazarevic’s method.

The Musk dataset also does not have very cleanly separable outliers and for this dataset also the $AUC$ is not as high as it is with the Wisconsin breast-cancer dataset. Nevertheless, for this dataset also, $eSelect$ consistently out-performs the Lazarevic’s method.

4.5 Conclusion

Detection of anomalies in high-dimensional data poses unique problems. The techniques developed for low-dimensional data are not very effective when applied to the dataset with very large number of dimensions[3]. This occurs due to the curse of dimensionality[7]. It has been observed that as the number of dimensions increases, the distances between the nearest data points approach the distances between the farthest data points[8]. Consequently, the proximity based methods become ineffective in the face of high-dimensional datasets. Further, it has been observed that many attributes don’t contribute much towards recognition of outliers. Instead, many of these attributes contain randomly distributed combinations[14] and can be considered as noise[24].

It has been suggested that in such high dimensional data the outliers are actually hidden in the low-dimensional projections of the dataset[3]. However, the number of low-dimensional projects, also called the subspace,
is a combinatorial number with respect to the number of attributes and its exhaustive exploration is not possible.

We have presented in this chapter a subspace sampling based method called $eSelect$ which is an effective method for sifting through the search space consisting of all possible subspaces in the subspace lattice of the dataset. The new method out-performs the currently well-established methods of sampling the subspaces.

The newly proposed method is founded upon the theoretical arguments provided by Aggarwal et al. [3]. We use these arguments to narrow down the subspaces that need to be explored in the search of outliers. Hence, unlike Lazarevic’s method that samples even very large subspaces, we select from subspaces containing much smaller number of attributes.

As stated earlier, our method is based on the hypothesis that low dimensional subspaces are more useful in the search for outliers as compared to high dimensional subspaces. Hence our method preferably sifts through the low dimensional subspaces over high dimensional subspaces. Validity of this assumption is the fundamental reason behind the improvements demonstrated by $eSelect$. Our dependence on this hypothesis is vindicated by the results of our experiments.

Further, unlike Aggarwal’s method that requires the analyst to input a parameter to represent the number of equi-depth ranges into which the attributes need to be split, we place no such requirement upon the analyst. Unlike the method proposed by Nguyen et al. [18] which uses the arbitrary
value of 10 as the number of regions into which the attributes need to be split, we use the value of 2 which does not unwarrantedly restrict the number of attributes in the subspaces that need to be explored. Further, unlike Nguyen’s method, we don’t exhaustively search through all the subspaces up to a certain size. Instead, we sample for subspaces among all the available subspaces up to a maximum number of attributes, as described in the description of our method. This allows us to scale our method to arbitrarily large datasets.

Finally, our method includes parameters that specify the cardinality of the subspaces that need to be completely explored as a heuristic to further control the execution of computation so that it does not exceed the computation resources available to the analyst. We aim that the search for anomalies should gracefully degrade in the face of insufficient resources as the number of dimensions grows. The search should not instead simply fail to terminate in a reasonable period of time.

Hence our method can operate on dataset with huge number of records and attributes and produce results with accuracy governed by the computational resources committed by the analyst, instead of simply failing to accommodate such huge datasets.
Chapter 5

Outlier Analysis Using Lattice of Contiguous Subspaces

In the previous chapters we have discussed two different aspects of anomaly detection, namely, a new metric of outlier-ness and discovery of anomalies in very high dimensional datasets by examining individual subspaces. Another important facet of anomaly detection is the identification of inter-relationships among subspaces within which the anomalies are detected. An analyst may get important clues about the underlying processes that lead to outlying data points from an analysis of the set of attributes that, in their subspace, reveal specific outliers.

As an example, consider a dataset $D$ with four attributes, 1, 2, 3 and 4. Fig. 5.1 displays the subspace lattice for such a dataset. In this figure, each node represents a subspace. Let us say that we checked for outliers in all these subspaces and found that some data point $P$ is identified as
Figure 5.1: Subspace Lattice with a Contiguous Region of Subspaces

anomalous in the subspaces 1, 1.2 and 1.2.3, and is not seen as outlier in the remaining subspaces. Note that these subspaces are connected to each other through paths that only include these subspaces in the lattice and are encapsulated in the dotted region. This is a contiguous region of the subspace lattice within which the same point \( P \) has displayed anomalous behavior. Such contiguous regions can provide important insights into the processes that give rise to outliers.

For example, in the above mentioned dataset, an analyst can get a hint that there is something peculiar about attribute 1 because even in presence of attribute 2 in subspace 1.2 or both 2 and 3 in subspace 1.2.3, the point \( P \) continues to be an anomaly. At the same time, when only the attributes 1 and 3 are present in subspace 1.3, the point comes across as normal even though \( P \) was anomalous in the subspace above and below the subspace 1.3. Further the data point also comes across as normal in
subspaces containing only attribute 2 or attribute 3.

We believe that valuable insights can be gained not only by knowing about subspace 1 which is the topmost subspace in this contiguous region, but also by knowing the bottom-most subspaces such as 1.2.3 of such a region and also the other lattice nodes that lie inside the region. When we have a larger lattice with more subspaces below 1.2.3, such as 1.2.3.4 in Fig. 5.1, we can infer the peculiarity of 1.2.3 because it is the lower-most subspace in which $P$ was anomalous. All such subspaces identify the sets of attributes and their specific combinations under which $P$ turns out to be anomalous.

We call subspaces such as 1.3 in which $P$ appears normal but which lies between subspaces such as 1 (top) and 1.2.3 (bottom) where $P$ again becomes an anomaly as the absent subspaces.

Taken together, such topmost, bottom-most and absent subspaces along with the aggregate of subspaces inside the contiguous lattice region can provide hints about the source attributes due to which anomalies are arising in a domain. We believe that having such insights in addition to knowing the list of anomalous data points is important for discovery of the sources of outliers in datasets.

Another use of the subspace information is for a better description of the outliers. Consider an example of a group of students who are enrolled in a set of courses, say $English(E)$, $Biology(B)$, $Physics(P)$, $Chemistry(C)$, and $Mathematics(M)$. Let us say we want to find anomalous students
among the group, and also justify our choice.

A student may have a GPA of 3.0 in all the courses taken together and may not be anomalous in the complete data space. But in subspaces C, E, M, E.M, C.M and C.E.M, (s)he may stand out as an outlier because of his low GPA. He may also be anomalous in P, B, and B.P because of his high GPA. So there are two regions in the lattice where (s)he stands out as an anomaly. The first region has C and E as the topmost subspaces, C.E.M as its bottom-most subspace, and C.E as the absent subspace. The second region has P and B as topmost subspaces and B.P as the bottom-most subspace. There is no absent subspace in this second region. This student is not an anomaly in supersets of subspace C.E.M, or in the supersets of subspace B.P, because his anomalous nature in any of the superset subspaces is neutralized due to the values of other attributes (and also depending on the criterion used to call a point an outlier). Knowledge of the regions provides an insight of anomalous nature of student’s performance even though in the complete data space he does not show up as an anomaly.

The top subspaces of a region of anomaly show the smallest combinations of attributes that make a data point an anomaly and in smaller spaces the data point is not an anomaly. For the case of a 2 – D space, it is possible that a point may be within normal range for its x – value individually and also for its y – value individually. But the combination (x, y) makes it an anomaly.

Similarly, the bottom-most subspaces of a lattice region for an anomaly
tell us the largest subspaces in which a point remains an anomaly and addition of other attributes dilutes its anomalous nature. The additional attributes may either bring in noise or contributions that are counter to a point remaining an anomaly. The absent subspaces provide an insight about the mutually neutralizing effect of certain attributes of peculiarities of the underlying datasets.

Such insights are very useful to explain the anomalousness of data points. It is useful to know the topmost and the bottom-most nodes of a region of anomaly in a lattice because of the insights provided by them. This is one of the differences in our work presented here from other approaches [16] in which only the topmost subspace is reported for some of the outlier points.

A few more examples where this information may be useful are as follows. In a database of voting records a congressman may have voted 90% of times with his party but may be an anomaly in all the subspace of bills involving immigration and agriculture issues. In an insurance company’s database of patients, every one may have been reimbursed the same total amount but some patients may be anomalous because of their expenses in all subspaces involving emergency room visits, medical assertive device purchases, and cardiac tests. The bottom-most and the topmost subspaces in these examples provide important information about why the anomalous cases are arising in the dataset.

In this chapter we provide a methodology to identify such regions of
subspaces in lattices and also for identifying their topmost, bottom-most, and absent subspaces.

5.1 Related Work

Anomaly or outlier detection has been a topic of investigation in the field of statistics for quite some time [13]. Hawkins’ widely quoted definition of an outlier, as also mentioned in earlier chapters is “An outlier is an observation that deviates so much from other observations as to arouse suspicion that it was generated by a different mechanism”. The canonical approach is to assume, with some justification, a statistical model for the data, and then to detect data points that violate the assumed model [10]. This approach works with data that has a relatively small number of attributes because with increasing number of attributes, it becomes progressively difficult and computationally intensive to justify a choice of model and also to test the data points against this model. Hence the data mining research is also exploring non-statistical methods where this requirement is eliminated.

The research on outlier detection can be divided into two broad categories. The first category of research is about developing efficient methods for finding outliers. Examples of this kind of research include finding new distance metrics, new definitions of outliers and better algorithms of searching for outliers in dataset. Discussions from chapters 2 and 3 are representative of such approaches. The second category of research involves
trying to understand or explain the anomalies detected using the methods in the first category.

Our work in this chapter provides a framework to address the issue of trying to look for the reasons for observed outliers. In other words, in this chapter we address a somewhat different problem than the one tackled in the research discussed in previous chapters. The methodology we present here can utilize any statistical criterion of identifying outliers in the context of subspace, and use it to provide details about the subspace structure in the lattice of subspaces within which the outliers are being generated. Hence our work belongs to the second broad direction of research on outlier detection that involves understanding and explaining some insights about the detected outliers. A representative work of this type was by Knorr et al[16] where the goal was to find ‘intensional knowledge’ about outliers by determining the minimal subspaces in which anomalous behavior was detected along with the detected outliers. The authors claimed that these minimal subspaces, called ‘strongest outlying subspaces‘ provide a kind of explanation of the outlying behavior.

While our work belongs to the same category of research as that of Knorr mentioned above, we aim at not only finding the minimal subspaces, but the whole multiple disjoint contiguous subspace regions of subspace lattice where the same data point exhibits anomalous behavior. Further we also consider those exception subspaces that lie between the topmost and the bottom-most subspaces of such regions, and in which the data point
appears normal. Hence we provide a more comprehensive treatment of the problem of finding insights about the possible causal factors of outlier behavior of data points.

Keller et al. have recently proposed a data pre-processing method wherein the goal is to determine the subspaces where interesting outliers may be found [14]. The analyst can then specifically apply outlier detection algorithms in these subspaces. However this method is limited to working with pre-selected subspaces and does not look at the relationships between various contiguous subspaces in which interesting outliers can be found. Zhang et al. have proposed an algorithm called HOS-Miner (High-dimensional Outlier Subspace Miner)[23]. The goal of this algorithm is to detect some subspaces within which a given data point is an outlier. However, this method again did not look at the lattice structure of the subspaces to extract deeper insights from them. Further, this method compares distances across different subspaces without normalizing these distances with respect to the number of attributes in the subspace. This yields closure properties that allow pruning of the set of subspaces to be searched. Further, as noted by Aggarwal[1], while this pruning can make the method efficient, it can miss out the large chunks of the subspace lattice in which the outlier-containing subspaces lie between two subspaces that themselves don’t have outliers.
5.2 Our Approach

We provide a methodology to determine the set of subspaces in which the same data point exhibits outlying behavior. We analyze such a set of subspaces to provide hints about why the outliers might be occurring. Our methodology is independent of specific distance metrics and definitions of outliers. It builds upon the existing outlier detection algorithms. Specifically:

- We determine the set of contiguous subspaces within which the same data point exhibits outlying behavior.
- Summarize the collected information for each outlier and identify the subspaces in which a given outlier is detected.
- Finally, analyze the subspaces computed in previous stem and report the contiguous subspace regions, topmost subspaces, bottom-most subspaces and absent subspaces.

We briefly explain the three phases now.

5.2.1 Subspace Lattice Exploration

As explained earlier, our methods builds upon existing techniques and definitions of outliers and similarity measures. Hence we can pick any existing method and utilize it for further outlier analysis. For the sake of discussion in this chapter, we have adopted the following two definitions as examples:
A - Z-Scores in the context of ‘k’ nearest neighbors.

B - \textit{LocalOutlierFactor}(LOF)[9]

Knorr and Ng[16] proposed one of the earliest definitions of distance based outliers as:

\textbf{Definition 5.1.} An object $O$ in a dataset $T$ is a $DB(p, D)$ \textit{outlier} if at least fraction ‘$p$’ of the objects in $T$ lies at greater than distance $D$ from $O$.

Angiulli and Pizutti[4] presented a related and widely used definition of outliers, described by Nguyen[18] as:

\textbf{Definition 5.2.} The dissimilarity of a point $p$ with respect to its $k$ nearest neighbors is known by its cumulative neighborhood distance, defined as:

$$F_{out}(p) = \sum_{m \in kNN_p} D(p, m)$$

We adapt a variation of these two definitions as our definition of outliers as:

\textbf{Definition 5.3.} Outliers are the data points whose $z$-scores of distances from their $k$-nearest neighbors, which a subspace, are greater than some threshold $t$.

In most of our tests, we have used value of 3.0 for $t$. Let us call this definition of outliers ‘$Z$-Dist’. Which this definition, the only other parameter that needs to be specified for identifying outliers is the value of ‘$k$’,...
i.e. the number of nearest neighbors to consider. This definition is not a new contribution. It has been used in the field of geographic datasets[11].

Furthermore, we also demonstrate our analysis using another definition of outliers to exemplify that our proposed methodology can plug-in any definition of outliers. This second definition of outliers is the widely used Local Outlier Factor proposed by Breunig et al.[9].

The pseudo-code for algorithm to explore the outliers in subspace lattice is in the Algorithm 5.1.

**Algorithm 5.1: Lattice Exploration Using Z-Dist**

```
Input:
  k: number of nearest neighbors
  L: subspace Lattice
  DS: data set

Output: Subspace lattice L containing the detected outliers in each subspace

1 for each subspace S ∈ L do
2   knn_dist ← allocate array of length |DS|
3   i ← 1
4   for each data point P ∈ DS do
5     knn_dist[i] ← avg. dist. to k nearest neighborhood of P
6     i ← i + 1
7   end
8   z_score ← calculate z-score of distances in knn_dist
9   Select points P whose z-score is greater than 3 and record them in jS
10  end
11 return L
```

This procedure performs the following functions:

- Walks through the subspaces in the lattice
- Computes the outliers in each subspace
- Stores the outliers detected in a given subspace within the data structure representing the subspace.

If the number of subspaces is $m$ and number of data points is $n$, the time complexity of this procedure is $O(mn^2)$. This is because we iterate through each subspace and calculate distance matrix for the $n$ data points.

A similar procedure to detect outliers using LOF is depicted in Algorithm 5.2.

**Algorithm 5.2: Lattice Exploration Using LOF**

**Input:**

- $k$: number of nearest neighbors
- $l$: number of nearest neighbors
- $L$: subspace Lattice
- $DS$: data set

**Output:** Subspace lattice $L$ containing the detected outliers in each subspace

```plaintext
1 for each subspace $S \in L$ do
2     lof_score ← allocate array of length $|DS|$  
3         i ← 1
4     for each data point $P \in DS$ do
5         score ← calculate LOF score of $P$ using $k$ nearest neighbors
6         lof_score[i] ← score
7         i ← i + 1
8     end
9     Select points $P$ whose $lof-score$ is among $l$ highest and record them in $S$
10 end
11 return $L$
```

The information about outliers contained within each subspace is then seeded into the next step below.
In a general case, finding LOF scores for \( n \) data points has time complexity \( O(n^2) \)[9]. Since we repeat this calculation for \( m \) subspaces, the time complexity of the procedure above is \( O(mn^2) \).

### 5.2.2 Summarization of Subspaces for Outliers

Algorithm 5.1 collects information about the outliers identified within the various subspaces in subspace lattice. Using this, we next find out the subspaces within which a given outlier is identified. The method is specified in Algorithm 5.3

**Algorithm 5.3: Subspace Summarization for Outliers**

**Input:**

- \( L \): subspace lattice containing the outliers identified in each subspace of the lattice
- \( DS \): data set

**Output:** Set \( S_{out} \) of outliers, with each outlier data structure containing the set of subspaces within which this outlier has been identified.

```
1 for each subspace \( S \in L \) do
2     \( S_{out} \) ← allocate new set of outliers
3     for each subspace \( s \in L \) do
4         for each outlier \( o \in s \) do
5             if \( o \in S_{out} \) then
6                 o.subspace.append(s)
7             end
8         end
9     else
10        o ← allocate new outlier object
11        o.subspace.set.append(s)
12        \( S_{out}.add(o) \)
13     end
14 end
15 return \( S_{out} \)
```
let us suppose that on average, the count of outliers in each subspace is \( q \). The average time complexity of algorithm 5.3 is \( O(mq) \) where, as before, \( m \) is the number of subspaces. Consider a case in which no outlier was found in any subspace. In this case we will execute the outer for-loop \( m \) times. Hence the lower bound of the time complexity is \( O(m) \). Similarly, consider the other extreme case where all \( n \) points get flagged as outlier. This is not a realistic scenario; however this is the extreme case that the algorithm may theoretically encounter. The time complexity in this case has the bound \( O(mn) \).

The set of outliers obtained in the second phase is fed into the third phase.

### 5.2.3 Determination of Contiguous Subspaces

The goal of this phase is to determine the set of contiguous subspaces for each identified outlier. A subspace lattice can be modeled analogous to a graph as follows:

**Definition 5.4.** A lattice \( L \) is a tuple:

\[
L = (S, E, \phi, S_{last})
\]

where:

- \( S \) is a set of independent sets of subspaces
- \( E \) is a set of edges between subspaces in \( S \)
φ is the NULL subspace that contains no attributes

\( s_{\text{last}} \) is the last subspace that contains all attributes

Note that as \( S \) is a set of independent sets, the edges in \( E \) can only consist of subspaces in two different independent sets in \( S \).

Further, if two subspaces \( p \) and \( q \) are connected by an edge, then \( p \) has exactly one more (or less) attribute than \( q \).

**Definition 5.5.** If an edge exists between two subspaces \( p \) and \( q \) in \( L \), then if \( p \) has one more attribute that \( q \), it is called a child of \( q \). Otherwise \( p \) is called a parent of \( q \).

**Definition 5.6.** A path between two subspaces \( p \) and \( q \) consists of the sequence of subspaces \( (s_1, s_2, ..., s_{n-1}, s_n) \) such that:

\[ 1 \leq i < n \text{ there is an edge between } s_i \text{ and } s_{i+1}, \text{ and } s_1 = p \text{ and } s_n = q \]

The subspaces \( s_1, s_2, ..., s_{n-2}, s_{n-1} \) in the path from \( p \) to \( q \) are called intermediary subspaces.

We now define a set of contiguous subspaces.

Informally, a contiguous region of a subspace lattice is the set of subspaces such that we can start from any subspace in this region and reach any other subspace in this same region without needing to visit a subspace outside the region. A formal definition of contiguous subspaces is as follows.
Definition 5.7. Two subspaces $p$ and $q$ are said to cohabit a region $R$ if:

- Either $p$ and $q$ are connected by an edge.
- Or, there exists a path between $p$ and $q$ with intermediary nodes $s_1, s_2, s_{n-1}, s_n$ such that for $1 \leq i < n$, $s_i$ and $s_{i+1}$ also cohabit the regions $R$.

Definition 5.8. The ‘cohabit’ relationship as described above defines an equivalence relation on subspaces in $S$ that can be used to divide $S$ into disjoint sets. Each such disjoint set is called a ‘contiguous region’ in the given subspace set $S$.

Definition 5.9. For a contiguous subspace lattice region $R$, all nodes $p$ such that no parent(child) of $p$ exists within $R$ are called ‘topmost nodes’ (‘bottom-most nodes) in region $R$.

Definition 5.10. Subspace $q$ is descendant (ancestor) of subspace $p$ if:

- Either $q$ is child (parent) subspace of $p$
- Or $q$ is descendant (ancestor) of a child (parent) subspace of $p$.

The above definition recursively defines the concept of descendant subspaces in subspace lattice. It is analogous to the recursive definition of binary trees.

The next step is to identify contiguous regions among the set of subspaces in which the same outlier has been identified. The procedure described in algorithm 5.4 picks up a subspace in which this given outlier has
been identified. Then it keeps exploring those child and parent nodes of this subspace in which the same outlier is identified. Each completion of this step yields one contiguous region of subspaces in which the outlier is identified. This step is continued till all the subspaces in which the given outlier is identified have been explored, potentially generating more than one contiguous region. The procedure repeats this computation for all the outliers that have been identified.

**Algorithm 5.4: Contiguous Region Computation**

**Input:**

- $S_{out}$: set of outliers identified. Outliers include the subspaces in which they were identified.

**Output:** Collection of contiguous regions of subspaces.

1. for each outlier $o \in S_{out}$ do
2.   $C \leftarrow$ allocate a collection of sets of subspaces
3.   while subspaces unexplored is not empty do
4.     $R \leftarrow$ allocate new subspace set
5.     $s \leftarrow$ get a subspace from subspaces unexplored
6.     list subspace $\leftarrow$ allocate a new list of subspaces
7.     Add $s$ to list subspace
8.     while list subspace is not empty do
9.       $s_{tmp} \leftarrow$ get a subspace from list subspace
10.      Add $s_{tmp}$ to $R$
11.     Add to list subspace all parents of $s_{tmp}$ that exist in subspaces unexplored and have not yet been explored, and remove them from subspaces unexplored
12.     Add to list subspace all children of $s_{tmp}$ that exist in subspaces unexplored and have not yet been explored, and remove them from subspaces unexplored
13.   end
14.   add $R$ to $C$
15. end
16. end
17. return $C$

Let us denote the number of outliers by $q$ and the number of subspaces
by $m$. In the worst case, all $q$ data points will be outliers in all the $m$ subspaces. Hence the worst case complexity of the above procedure is $O mq$. However, in real datasets it is unlikely that a data point will appear anomalous in all subspaces and this can happen only for pathological cases in very simple datasets. Hence this worst case is extremely unlikely to occur. The average behavior depends on the average occurrence of outliers. Since this is difficult to quantify a-priori what the average number of outliers and their distribution of various subspaces can be, the complexity algorithms in average case is difficult to specify because of lack of information about the average input data.

Next, we determine the topmost subspaces in any given contiguous region. Procedure 5.5 iterates through all the subspaces in a given contiguous region and identifies and returns those subspaces whose parents don’t exist in this contiguous region.

### Algorithm 5.5: Computation of Topmost Subspaces

**Input:**

$R$: A contiguous region of subspaces

**Output:** Set $S_{\text{out}}$ of outliers, with each outlier data structure containing the set of subspaces within which this outlier has been identified.

1. $S \leftarrow \text{allocate a set of subspaces}$
2. **for each subspace** $s \in R$ **do**
3.   **if** $R$ does not contain any parent of $s$ **then**
4.   
5.   **end**
6. **end**
7. **return** $S$

The worst case complexity for this procedure is $Oq$, where we assume
that all $q$ subspaces in the lattice belong to a single contiguous region.

A procedure analogous to algorithm 5.5 can be written for determining the bottom-most subspaces in a contiguous region by considering the child subspaces instead of the parent subspaces of the subspace $s$ in the \textit{for} loop.

Next, we determine the absent subspaces for a given contiguous region. The procedure starts from the topmost subspaces and places them in a queue. For each subspace in the queue, it determines whether the bottom-most subspaces are descendants of this subspace, and also whether this subspace exists in the region. If this subspace does not exist in the region, it is considered absent subspace. If it does exist in the region, its child nodes are placed in the queue for similar examination.

In the worst case, we will end up iterating through all the subspaces in the lattice. Hence the worst case complexity is $O(m)$ where $m$ is the number of subspaces in the lattice.

As as result of execution of these algorithms, we obtain the following:

- The set of contiguous regions within which a given data point displays anomalous behavior
- The topmost and bottom-most subspaces in the aforementioned contiguous regions.
- The subspaces between the topmost and bottom-most subspaces within which the data points appear normal.
Algorithm 5.6: Computation of Absent Subspaces

Input:

- $R$: A contiguous region of subspaces
- $S_{\text{topmost}}$: topmost subspaces of $R$
- $S_{\text{bottom-most}}$: bottom-most subspaces of $R$

Output: Set of subspaces that exist between $S_{\text{topmost}}$ and $S_{\text{bottom-most}}$ in the contiguous subspaces lattice, but do not lie within the contiguous region $R$.

1. $\text{list}_{\text{absent-subspaces}} \leftarrow \text{allocate list of subspaces}$
2. $\text{list}_{\text{subspaces}} \leftarrow \text{allocate list of subspaces}$
3. Add all subspaces in $S_{\text{topmost}}$ to $\text{list}_{\text{subspaces}}$
4. Add all subspaces in $S_{\text{topmost}}$ to $\text{list}_{\text{subspaces}}$
5. while $\text{list}_{\text{subspaces}}$ is not empty do
6.     $s \leftarrow \text{get a subspace from list}_{\text{subspaces}}$
7.     if $S_{\text{bottom-most}}$ contains descendants of $s$ then
8.         Add children of $s$ to $\text{list}_{\text{subspaces}}$
9.         if $R$ does not contain $s$ then
10.            $\text{Add } s \text{ to list}_{\text{absent-subspaces}}$
11.         end
12.     end
13. end
14. return $\text{list}_{\text{absent-subspaces}}$
This information is instrumental in providing insights into the anomalous behavior of data points.

5.3 Results

We executed the procedure mentioned in the previous section on the following datasets obtained from the UCI Machine Learning Repository[5]:

- Iris dataset
- Seeds dataset

We used the two definitions of outliers mentioned before as drivers in our analysis of outliers over various subspaces.

5.3.1 Iris Dataset

Let us first consider the analysis of Iris dataset using $ZDist$ that we defined earlier. Iris is well-studied dataset with 150 data points and 4 attributes that represent properties of the Iris flower. The attributes are:

- Sepal length
- Sepal width
- Petal length
- Petal width
All measurements are in centimeters. We represent these attributes with numerals 1, 2, 3 and 4 respectively. We represent a subspace as a dot-separated sequence of attributes that constitute this subspace. For example, the sequence 1.3.4 represents the subspace comprising of attributes 1, 3 and 4. While the order of these attributes does not matter, we list them in increasing numerical order.

If we consider the whole attribute space, only three data points, namely 132, 118 and 119 are identified as outlier using $Z_{Dist}$, with a neighborhood of 30 points. However, when we search through subspaces, 18 data points including the above mentioned 3 data points are identified as outliers across various subspaces.

Further, we gain interesting insights such as, data point 123 is identified as outlier in the contiguous region of subspaces \{1, 3, 1.3, 3.4, 1.2.3, 1.3.4\}. The topmost subspaces of this region are 1 and 3. The bottom-most subspaces are \{1.2.3\} and \{1.3.4\}. Interestingly, the subspaces 1.2, 1.4 and 2.3 that lie on the subspace lattice along the paths from the topmost subspaces to bottom-most subspaces do not identify the data point 123 as an outlier. This indicates a peculiarity in these subspaces because the same data point appears anomalous in the subspaces immediately above and below these subspaces, but not within them. Note also that this data point is not identified as an outlier in the complete attribute space. It is anomalous only within specific subspaces.
Table 5.1: Outlier Analysis on Iris Dataset using Z-Dist

<table>
<thead>
<tr>
<th>Data Point</th>
<th>Is Outlier in Complete Attribute Space</th>
<th>Contiguous Regions</th>
<th>Topmost Subspaces</th>
<th>Bottommost Subspaces</th>
<th>Absent Subspaces</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Yes</td>
<td>{1, 1.2, 1.3, 1.4, 2.3, 2.4, 1.2.3, 1.2.4, 1.3.4, 1.2.3.4}</td>
<td>1, 2.3, 2.4</td>
<td>1.2.3.4</td>
<td>2.3.4</td>
</tr>
<tr>
<td>132</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Yes</td>
<td>{1, 3, 1.2, 1.3, 2.3, 2.4, 3.4, 1.2.3, 1.2.4, 1.3.4, 2.3.4, 1.2.3.4}</td>
<td>1, 3, 2.4</td>
<td>1.2.3.4</td>
<td>1.4</td>
</tr>
<tr>
<td>118</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Yes</td>
<td>{1, 3, 1.2, 1.3, 2.3, 3.4, 1.2.3, 1.2.4, 1.3.4, 2.3.4, 1.2.3.4}</td>
<td>1, 3</td>
<td>1.2.3.4</td>
<td>1.4</td>
</tr>
<tr>
<td>119</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>106</td>
<td>No</td>
<td>{3}</td>
<td>3</td>
<td>3</td>
<td>None</td>
</tr>
</tbody>
</table>
Table 5.2: Outlier Analysis on Iris Dataset using LOF

<table>
<thead>
<tr>
<th>Data Point</th>
<th>Is Outlier in Complete Attribute Space</th>
<th>Contiguous Regions</th>
<th>Topmost Subspaces</th>
<th>Bottom-most Subspaces</th>
<th>Absent Subspaces</th>
</tr>
</thead>
<tbody>
<tr>
<td>99</td>
<td>Yes</td>
<td>{1.3, 1.4, 2.3, 3.4, 1.2.3, 1.3.4, 2.3.4}</td>
<td>1.3, 1.4, 2.3, 3.4</td>
<td>1.2.3.4</td>
<td>1.2.4</td>
</tr>
<tr>
<td>23</td>
<td>Yes</td>
<td>{2.3, 3.4, 1.2.3, 2.3.4, 1.2.3.4}</td>
<td>2.3, 3.4</td>
<td>1.2.3.4</td>
<td>1.3.4</td>
</tr>
<tr>
<td>42</td>
<td>Yes</td>
<td>{3, 2.3, 2.4, 1.2.3, 1.2.4, 2.3.4, 1.2.3.4}</td>
<td>3, 2.4</td>
<td>1.2.3.4</td>
<td>1.3, 3.4</td>
</tr>
</tbody>
</table>

5.3.2 Seeds Dataset

The seeds dataset consists of 210 data points and 7 attributes. The lattice generated from this dataset consists of 127 subspaces. Being a larger lattice, this yields richer set of information than the Iris dataset.

We first discuss the analysis of this dataset using our methodology with Z-Dist. When we look for outliers in all subspaces, we identify 4 outliers. However, on searching across various subspaces, 23 outlier points were identified, including the aforementioned 4 data points. As an example,
data point 109 is identified as an outlier in contiguous region

\{4, 7, 3.4, 3.7, 4.5, 4.7, 3.4.5, 3.4, 7, 4, 5.7, 3.4.5.7\}

of the subspace lattice. The topmost subspaces in this region are 4 and 7. The bottom-most subspace in this region is 3.4.5.7. Subspaces 5.7 and 3.5.7 lie between the aforementioned topmost and bottom-most subspaces but are absent from the contiguous region. Also, the size of this contiguous region is 10.

Since the subspace lattice of seeds dataset is much larger than that of Iris dataset, the contiguous regions determined for a few outliers are very large. We show a few representative outliers in table 5.3. We have also recorded the sizes of contiguous regions of subspaces. To save space, instead of showing huge contiguous regions of subspaces, we only mention a few subspaces from this region. The size of region indicates that there are a greater number of subspaces in the region than those displayed in the column for contiguous region.

Finally, we discuss the analysis of this dataset using LOF as the definition of outliers. When we consider all the attributes, 10 outliers were identified. However, on searching across various subspaces, 89 outliers including the aforementioned 10 data points were identified as outliers. Data point 125 is an example of the identified outliers in the contiguous region \{2.4, 4.5, 2.3.4, 2.4.5, 4.5.7, 2.3.4.5, 2.4.5.7, 3.4.5.7, 2.3.4.5.7\}. The top-
Table 5.3: Outlier Analysis on Seeds Dataset using Z-Dist

<table>
<thead>
<tr>
<th>Data Point</th>
<th>Is Outlier in Complete Attribute Space</th>
<th>Contiguous Regions</th>
<th>Topmost Subspaces</th>
<th>Bottom-most Subspaces</th>
<th>Absent Subspaces</th>
<th>Region Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>78</td>
<td>No</td>
<td>{1, 2, 4, 1.2, 1.3, …, 2.3.4.5.7, 1.2.3.4.5.7}</td>
<td>1, 2, 4</td>
<td>1.2.3.4.5.7</td>
<td>4.5, 4.7, 3.4.5, 3.4.7</td>
<td>50</td>
</tr>
<tr>
<td>114</td>
<td>Yes</td>
<td>{1.6, 2.6, …, 2.3.4.5.6.7, 1.2.3.4.5.6.7}</td>
<td>1.6, 2.6, 4.5.6, 4.6.7, 5.6.7</td>
<td>1.2.3.4.5.6.7</td>
<td>Empty</td>
<td>56</td>
</tr>
<tr>
<td>115</td>
<td>No</td>
<td>{1, 2, …, 2.3.4.5.7, 1.2.3.4.5.7}</td>
<td>1, 2, 4, 5</td>
<td>1.2.3.4.5.7</td>
<td>4.7, 5.7, 3.4.7, 3.5.7, 4.5.7, 3.4.5.7</td>
<td>54</td>
</tr>
</tbody>
</table>

most subspaces for this regions are 2.4 and 4.5. The bottom-most subspace is 2.3.4.5.7. The subspaces 2.4.7, 3.4.5 and 2.3.4.7 lie between the topmost and bottom-most subspaces, but are absent from the contiguous region. The size of this contiguous region is 9. We show a few more representative results in table 5.4.
Table 5.4: Outlier Analysis on Seeds Dataset using LOF

<table>
<thead>
<tr>
<th>Data Point</th>
<th>Is Outlier in Complete Attribute Space</th>
<th>Contiguous Regions</th>
<th>Topmost Subspaces</th>
<th>Bottom-most Subspaces</th>
<th>Absent Subspaces</th>
<th>Region Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>150</td>
<td>No</td>
<td>{1, 1.2, 1.3.4.5, 1.2.3.4.5}</td>
<td>1, 2.4, 3.5</td>
<td>1.2.3.4.5</td>
<td>2.3.5, 2.4.5, 3.4.5, 2.3.4.5</td>
<td>10</td>
</tr>
<tr>
<td>84</td>
<td>No</td>
<td>{1, 1.3, 1.4.5, 1.3.4.5}</td>
<td>1</td>
<td>1.3.7, 1.3.4.5</td>
<td>Empty</td>
<td>10</td>
</tr>
<tr>
<td>84</td>
<td>No</td>
<td>{4.6, 6.7, 4.5.6.7, 3.4.5.6.7}</td>
<td>4.6, 6.7</td>
<td>3.4.5.6.7</td>
<td>4.6.7</td>
<td>11</td>
</tr>
<tr>
<td>175</td>
<td>No</td>
<td>{1, 1.3, 1.2.4.5, 1.2.3.4.5}</td>
<td>1, 3.5</td>
<td>1.2.3.4.5</td>
<td>1.2, 1.2.3, 1.4.5, 2.3.5, 3.4.5, 1.3.4.5</td>
<td>13</td>
</tr>
</tbody>
</table>

5.4 Conclusion

In this chapter we have presented a methodology that aims at providing greater insights into outlier behavior instead of just reporting the outliers and the subspaces with which they are identified. We compute:

- **Contiguous regions** of the subspace lattice within which the same data point is identified as outlier.

116
• *Topmost and bottom-most subspaces* in this region. These subspaces are important points of inflection along the subspace lattice where specific data points either start appearing as anomalies, or stop being anomalies.

• Subspaces that lie within the topmost and bottom-most subspaces mentioned above, and in which the data point in question does not appear anomalous. These subspaces introduce attributes under the influence of which the data point again appears normal. We term these subspaces the *absent subspaces*.

Taken together, all this information provides deeper insight to the analyst about causal factors (in terms of responsible attributes) of anomalies. This can help to either mitigate the reasons that give rise to anomalies, or anticipate anomalies and prepare to reduce their impact.

We believe that the aggregate of this information is more useful that only a listing of outliers and the subspaces in which they are identified. There may be cases in which a data point may be an outlier in a large number of subspaces and a comprehensible summarization of all these subspaces is given by specifying the topmost, the bottom-most and the absent subspaces. Our methodology can work in conjunction with any definition of outliers. We have demonstrated the insights gained from our methodology and its operability with multiple definitions of outliers by showing examples of results of analysis over the Iris and seeds datasets, using $Z - Dist$.
and $LOF$ as two competing definitions of outliers.
Chapter 6

Conclusions and Future Directions

Anomaly detection is an important problem in data mining and has tremendous applications. Anomalies are unusual data points that provide actionable information to an analyst, and in this respect they are different from noise[1]. While, like noise, outliers are the records that don’t belong with the rest of data, the analyst wants to detect them and make use of them instead of removing them from the data in the data-cleaning step.

However, it is not possible to work with a single definition of outlier-ness and have it apply universally to all situations in the data-analysis task. An analogy is that with the distance-metric. There are various distance metrics such as the Euclidean distance or the cosine similarity, and many others, and each of them serve important role in different data analysis scenarios. Similarly different definitions of outlier-ness have been proposed
In this dissertation we have proposed a new definition of outlier-ness and have demonstrated its efficacy over the current state-of-the-art methods such as the Local Outlier Factor (LOF)\cite{9}. The new definition of outlier-ness is called the Cohesiveness-Based Outlier Factor (CBOF). We have discussed the data-scenarios in which CBOF will out-perform LOF and have demonstrated this using synthetic dataset situations that were designed to exemplify our hypothesis.

Further, we applied the new method on multiple real datasets and demonstrated the effectiveness of CBOF over LOF. However, we also analyzed datasets where LOF performs quite well. This emphasizes the acknowledgement in contemporary literature that there is no single universal definition of outlier-ness that will necessarily work better than all alternatives and will detect all possible outliers.

We then proceeded to refine the CBOF metric to come up with the improvement called Improved Cohesiveness-Based Outlier Factor (iCBOF). Again, we thoroughly experimented with multiple real datasets and compared the performance of this method with CBOF and LOF. We demonstrated that both iCBOF and iCBOF offer improvements over LOF in specific data situations.

We then handled the case of detection of outliers in very high dimensional data. Most real-world datasets are very high-dimensional, e.g. the market-basket data from any grocery chain would consist of a large num-
ber of attributes for each item in the grocery store[21]. Consequently, it is important that data analysis methods should be able to handle such large and high-dimensional datasets. However, it is known that data-analysis methods don’t work well with such high-dimensional datasets and this phenomenon has been termed the *curse of dimensionality*[7]. It has been proposed by Aggarwal[2][1] that in such high dimensional data, interesting anomalies are located in low-dimensional subspaces of the complete attribute space. Consequently, the search for anomalies should be done in such small subspaces. However, the count of possible subspaces is a combinatorial function of the number of attributes in the dataset. Consequently, an exhaustive traversal of all possible subspaces is not computationally feasible.

In our third research problem, we have proposed a random subspace sampling based approach to work with such high-dimensional datasets. We improved upon an existing random subspace sampling method proposed by Lazarevic et al.[17] by prefering to select the low-dimensional subspaces among the available set of subspaces. We demonstrated the effectiveness of the new method called *eSelect* by extensive experiments on multiple real datasets.

We concluded our dissertation by proposing a method to conduct an organized traversal of the subspace lattice and studying the behavior of outliers over this lattice. The new method determines not only the top-most, but also the bottom-most subspaces in which the same record appears
anomalous. The new method further discovers the contiguous regions of the lattice where the same record retains its anomalous-ness. Finally, the new method also discovers subspaces within such a contiguous region in which the record does not appear anomalous. Such a systematic exploration of anomalous-ness over the subspace lattice reveals the structure in the data that is not otherwise apparent.

6.1 Future Work

The different threads of research on anomaly detection that we explored in this dissertation can be further extended. We will summarize some directions that we find worth exploring in future work.

6.1.1 Metric of Outlier-ness

In chapters 2 and 3 we proposed two new metrics of outlier-ness. These metrics work with Euclidean distances, and expect the data to be numerical. However, many datasets in real-world can also be categorical or ordinal. Distance measures for such datasets will be different from the distances measures for numerical datasets, and consequently the distance-based anomaly detection methods such as \(iCBOF, CBOF\) and \(LOF\) will need to be modified for such datasets. The effective-ness of these methods for such non-numerical datasets remains to be investigated. Upon further examination, it might be possible to come up with better methods of outlier
detection with are better suited for such datasets.

6.1.2 Outliers in High-Dimensional Data

Contemporary research literature acknowledges the difficulty in detecting outliers in high-dimensional data. Further, it is known that in such high-dimensional data, interesting outliers are embedded within low-dimensional subspaces. However, an extensive analysis about whether we need to at all look at high-dimensional subspaces remains to be seen. It is possible that the search space of random subspace sampling proposed in our work is already too large and a subspaces search over an even more restricted set of subspaces comprised of even lower number of attributes is very effective in real-world datasets. While an argument appealing to intuition has been made to advocate about also exploring high dimensional subspaces, an actual analysis on real-world data to demonstrate the presence of outliers in high-dimensional subspaces that eluded detection in lower dimensional subspaces can be done.

6.1.3 Outliers in Subspace Lattice

In our work we have proposed a method to explore the subspace lattice of a dataset in order to study how outlier-ness of datapoints behaves across the subspaces in the lattice. However, because of the combinatorial increase in the number of subspaces with increase in the number of attributes, it is
not possible to make such an exploration in an exhaustive and deterministic manner. Hence, probabilistic methods need to be explored for such datasets, and this is can be an interesting idea to explore to extend the solution proposed in this dissertation for high-dimensional subspaces.
Bibliography


