Unsupervised Spectral Ranking for Anomaly Detection

by

Ke Nian

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Master of Mathematics
in
Computer Science

Waterloo, Ontario, Canada, 2014

Abstract

Anomaly detection is the problem of finding deviations from expected normal patterns. A wide variety of applications, such as fraud detection for credit cards and insurance, medical image monitoring, network intrusion detection, and military surveillance, can be viewed as anomaly detection. For anomaly detection, obtaining accurate labels, especially labels for anomalous cases, is costly and time consuming, if not practically infeasible. This makes supervised anomaly detection less desirable in the domain of anomaly detection. In this thesis, we propose a novel unsupervised spectral ranking method for anomaly detection (SRA). Based on the 1st non-principal eigenvectors from Laplacian matrices, the proposed SRA can generate anomaly ranking either with respect to a single majority class or with respect to multiple majority classes. The ranking type is based on whether the percentage of the smaller class instances (positive or negative) is larger than the expected upper bound of the anomaly ratio. We justify the proposed spectral ranking by establishing a connection between the unsupervised support vector machine optimization and the spectral Laplacian optimization problem. Using both synthetic and real data sets, we show that our proposed SRA is a meaningful and effective alternative to the state-of-art unsupervised anomaly ranking methods. In addition, we show that, in certain scenarios, unsupervised SRA method surpasses the state-of-art unsupervised anomaly ranking methods in terms of performance and robustness of parameter tuning. Finally, we demonstrate that choosing appropriate similarity measures remains crucial in applying our proposed SRA algorithm.

Acknowledgements

I would like to thank my supervisors Prof. Yuying Li and Prof. Thomas F. Coleman for their patience and guidance with my work.

I thank my thesis reader Prof. Peter Forsyth and Prof. Justin Wan, for taking the time to review the thesis and give me valuable comments.

Thanks also go to my friends and lab mates from Scientific Computing group of University of Waterloo, Adiya Tayal, Eddie Cheung, HaoFan Zhang, Kai Ma, Ken Chan, Parsiad Azimzadeh and Swathi Amarala, for engaging and memorable discussions.

Dedication

Dedicated to my parents, Fushun Nian and Xuemei Wang, for their support and encouragement all along the years.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Anomaly Detection and Associated Challenges

Anomaly detection, at the abstract level, is to detect data instances in a given data set that do not conform to the expected normal patterns. The anomalies in data sets often translate to valuable knowledge such as clues to fight against malicious attack or behaviors, indication of malfunctions and so on. Anomalies in credit card transactions and insurance claims are often the fraudulent cases. Anomalies in the readings of electronic sensors often indicate malfunction of devices. Thus, successful detection of anomalies can lead to significant gains in business efficiency.

Traditionally, anomaly detection (e.g., fraud detection, malfunction detection, network intrusion detection) relies heavily on expert inspection which is often costly, time consuming, and highly inefficient. Data mining and machine learning techniques have been recognized as the key to speed up the detection of anomalous cases and greatly reduce economic losses. Based on the type of input data used, data mining and machine learning techniques used for anomaly detection can be classified into three categories:

- Supervised learning algorithms train models based on labeled data, i.e., input data instances whose output targets are already known. The objective of the supervised learning is to learn a function that map the input data instances to the output targets. In the cases of anomaly detection, output targets are often the binary class labels: 'normal' versus 'abnormal'. The learnt function is used to produce prediction of the output targets for previously unknown data instances. Typical supervised learning tasks include classification problems where the output targets are in discrete format and regression problems where the output targets are real number.

- Unsupervised learning algorithms train models based on unlabeled data, i.e., input data instances whose output targets are unknown. Unlike supervised learning algorithms which seek to map input data instances to output targets, unsupervised learning algorithms often aim at exploring and discovering the structure of the data sets. One of the typical unsupervised learning tasks is clustering, which tries to group the data instances so that data instances belonging to the same group are more similar to each other than data instances in other groups.
- Semi-supervised learning algorithms train models based on both labeled data and unlabeled data to generate appropriate discriminative functions that map input data instances to output targets. They are often used in the cases where output targets are known for only a small proportion of the input data instances.

The most straightforward approach to deal with anomaly detection is to formulate it as a supervised rare class classification problem which aims at inferring a discriminant function from labeled training data with highly unbalanced class distributions. However, several factors make this approach ineffective in practice.

- Obtaining labeled data for training in the context of anomaly detection is often difficult. Considering fraud detection as an example, obtaining clearly fraudulent and non-fraudulent samples is very costly, if not impossible.
- Crimes or malicious actions from human beings are often the major causes of anomalous instances in many data sets (e.g., fraud detection, network intrusion detection). There is a tendency for those anomalous instances to mimic the normal cases. Therefore the distinction between abnormal cases and normal cases is often blurred and imprecise. This fuzziness poses difficulties in defining accurate discriminant functions with satisfying predictive power.
- The presence of noises in the normal data instances, which often act similarly as anomalies, also introduces difficulty in deducing clear decision rules within the data sets.
- In many applications domains, credibility of labeled data is also an issue. Again, we use fraud detection as an example. Even if one ignores human investigative costs, it is quite common to find fraud investigators to differ in their claim assessments. This raises issues of data (label) trustworthiness.

- For many applications, both abnormal and normal patterns can evolve. Since obtaining labels takes time, this nature often leads to poor performance of using currently learnt discriminant functions to predict future anomalies.

The difficulties in obtaining labels, the credibility of human judgement, and the nature of evolving patterns means, in practice, unsupervised learning can be more feasible and useful than supervised and semi-supervised learning. Therefore, developing an efficient unsupervised anomaly detection technique can be of great use in practice. Several anomaly detection methods have already been proposed in the literatures. See, e.g., [42, 9, 26]. However, there are three additional challenging aspects, which are not well addressed in the previous work, that motivate us in developing a new unsupervised anomaly detection method:

- The nature of anomalies in different application domains are often different. Sometimes, anomalies correspond to outliers which are the isolated data instances far away from the normal data instances. We will refer to outliers as "point anomalies" in later sections. Sometimes, anomalies form small clusters. Sometimes, both scenarios exist. In certain cases, anomalies are actually embedded in certain contexts. These contextual anomalies can only be detected after we define the proper context. For example, a male with a height of 180 cm will not be regarded as an anomaly unless we provide the context that he is only 8 years old. Defining such context can be extremely hard. Although many effective anomaly detection techniques have been proposed, most of them have explicit assumptions of the nature of anomalies. In the cases where no domain knowledge exists, choosing the right detecting algorithms and tuning the parameter for the detection algorithms can be challenging .

- The types of input data can also have significant impact on the effectiveness of the anomaly detection. Many of existing methods are only effective with numerical data and lack the support for mixing data types or categorical data.

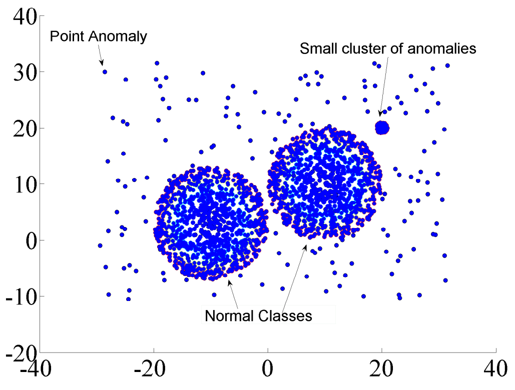
- In certain scenarios, only the pairwise distances or similarities between data instances are given as input [12]. In such cases, algorithms that need to access the original data instances, e.g. most of statistical methods, can not be used.

These additional challenges motivate us to develop an approach that can deal with different types of anomalies and different types of data simultaneously. In addition, we want to develop a method that use only the pairwise distances and similarities as input. Furthermore, we want to develop an approach whose learning outputs can be useful in deducing potential context. Thus, human judgement can be included to see whether the detected abnormal patterns is meaningful or not. In the unsupervised setting, this feature is crucial because validation of results in unsupervised setting is challenging.

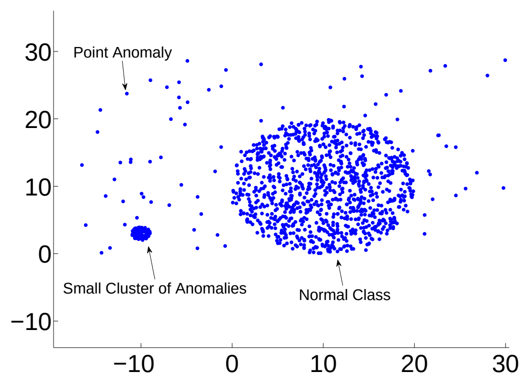
The presence of noises in normal data instances and the tendency of the malicious data instances to mimic normal patterns pose extra difficulties in defining clear discriminant functions between normal data instances and anomalies. Therefore, providing a ranking which indicates the relative likelihood of being abnormal is much more sensible and useful than providing binary classifications. Furthermore, for many anomaly detection problems, the cost associated with misclassifying an anomaly as a normal case is much greater than the cost associated with misclassifying a normal case as an anomaly. For example, the cost of providing loans to defaulters greatly exceeds that of denying loads to a non-defaulters. Similarly, the cost of predicting cancerous patients as being healthy greatly exceeds that of providing false alarms. Thus the cost-benefit analysis can be crucial in many application domains. With an anomaly ranking, an analyst can either analyze top few highly ranked instances or use a cut-off threshold to generate binary classifications. The analysts can conduct cost-benefit analysis by trying different cut-off thresholds to select the best output for their problems. This is the major advantage of providing anomaly ranking. Therefore, in this thesis, we focus on developing a ranking method which assigns an anomaly score to each data instance in a given data set.

1.2 Existing Anomaly Ranking Methods

In this section, we briefly survey related supervised, semi-supervised and unsupervised anomaly ranking methods and discuss their merits and disadvantages. Generally, there are three ways to classify the anomaly detection methods. One is to classify them based on whether they can cope with multiple patterns for normal cases or not. The difference between multi-class anomaly detection and one-class anomaly detection is illustrated in Figure 2.1. A second way to classify them is based on whether the labels have been used or not. Supervised and semi-supervised algorithms require labeled training data while unsupervised algorithms do not. However, the supervised and semi-supervised algorithms can produce the prediction for the new data instances which are not used in the training process while most of unsupervised algorithms typically do not. Finally, based on the basic ideas behind the algorithms, standard anomaly detection methods can also be classified as classification based approaches, clustering based approaches, nearest neighbor based approaches and statistical approaches see, e.g., [25] for a detailed discussion. Here, we only introduce several methods that can generate ranking scores. The discussion here are



(a) multi-class anomaly detection



(b) one-class anomaly detection

Figure 1.1: Illustration of multi-class anomaly detection and one-class anomaly detection

mainly drawn from [25] and [12].

Classification based approaches are supervised or semi-supervised learning. The supervised or semi-supervised classification methods require accurate labels and a few of them can be modified to generate ranking instead of binary predictions efficiently. RankSVM [27] can be applied to a bi-class rare class prediction problem. Unfortunately solving a kernel RankSVM problem is computationally prohibitive for large data mining problems. Using Support Vector Machines (SVMs) ranking loss function, a rare class based nonlinear kernel classification method, RankRC, is proposed recently in [50, 49]. RankRC has gained significant computational advantage over RankSVM while not sacrificing performance and effectiveness. One-class classification based approaches, such as One-Class Support Vector Machine (OC-SVM)[42], can be generalized to the unsupervised setting. For anomaly detection, the one-class classification algorithms assume that all training instances are normal cases and it will learn a discriminative boundary around the training data instances. Any test instances that do not fall within the boundary is regarded as anomalies. In the case of OC-SVM, the algorithm fits a hypersphere with smallest radius in a feature space that encloses most of the data instances. The feature space is usually a high dimensional derived feature space implicitly defined by a kernel function. A corresponding anomaly ranking score can be generated based on the distance of each data instance to the center of the hypersphere.

There are many clustering based anomaly detection algorithms. Most of the clustering based anomaly detection techniques output binary predictions. But there are also several clustering based ranking methods. The Clustering-Based Local Outlier Factor (CBLOF) [26] is one of the examples. The score from CBLOF is computed based on the size of the cluster to which the data instance belongs and the distance between the data instances to the cluster centroid. The implicit assumption is that normal cases belong to large and dense clusters while anomalies belong to small clusters. The main merit of clustering based ranking methods is that they do not require labels and they can handle both one-class anomaly detection and multi-class anomaly detection. The major disadvantage is that they often have implicit assumptions of the nature of anomalies within the data sets.

The nearest neighbor based approaches are also applied frequently in the context of anomaly detection. There are two categories: distance based methods and density based methods. The most representative distance based nearest neighbor ranking approach [12] can be summarized as follows: the anomaly score of a data instance is defined as its distance to its kth nearest neighbor. Many variations of this idea exist [21, 29]. There are also variations to deal with categorical features. The most representative density based nearest neighbor ranking approach is the Local Outlier Factor (LOF) [9]. The LOF score of a specific data instance is the ratio between the average local density of k nearest neighbors of the data instance and the local density of the data instance itself. Many variations of LOF such as Connectivity-based Outlier Factor (COF) [47] exist. One of the merits of the nearest neighbor based methods is that they only need a distance or similarity metric so they can handle data sets with both numerical and categorical attributes well and they do not need to access the original data instances. Another merit is that, with proper parameter k, they can deal with both multi-class anomaly detection cases and one-class anomaly detection cases. In addition, they do not have explicit assumption of the nature of anomalies. Anomalies can be data instances in small clusters or outliers. The major disadvantage is that tuning the parameter k is often quite challenging.

The last category of methods consists of the statistical algorithms. The basic idea is to estimate a probability density function from the sample data and regard the cases occur with a high probability as normal cases and the cases occur with low probability as anomalies. Here, the sample data is assumed to come from a specific probability model such as multi-variate Gaussian distribution or mixture of Gaussian distribution [33]. The parameters of the specific probability model are then estimated. We can also estimate non-parametric models using methods such as Parzen Window Estimation [55] and use the estimated probability density function to generate anomaly ranking. The major disadvantage of the parametric statistical approach is that if the assumption regarding the underlying statistical model is wrong, the performance of the ranking can be extremely poor. Another disadvantage of the statistical approaches is that it will be hard to apply them to data sets with categorical attributes.

1.3 Contribution

In this thesis, we focus on unsupervised learning in the context of anomaly detection. We develop a generic anomaly ranking approach that can be adapted to dealing with different types of anomalies and also different types of input data. Specifically we consider both the case in which anomaly is assessed with respect to a majority class, as well as the cases where anomaly is assessed with more than one major classes. To quantify patterns, we assume that a similarity measure is given from which patterns can be assessed. Each similarity measure yields a view on data patterns. We then use spectral analysis to identify the first couple of bi-modal non-principal eigenvectors of the corresponding Laplacian matrices. The anomaly ranking is generated based on the non-principal eigenvectors. The main contributions of the thesis are as follows:

- Motivated by the connection between the unsupervised support vector machine optimization and normalized spectral clustering optimization [24], we propose the unsupervised Spectral Ranking for Anomaly Detection (SRA) algorithm which generates ranking score directly from the non-principal bi-modal eigenvectors of a normalized Laplacian matrix. Alternative justification will also be briefly discussed.
- Following a detailed analysis on a real auto fraud detection data set, we demonstrate the potential usage of SRA on auto insurance fraud detection. Through identifying significant attributes which separate identified clusters, we also show that our proposed method leads to potentially useful rules for identifying fraud.
- We demonstrate the effectiveness of the unsupervised SRA for anomaly detection using both synthetic data sets and a few real data sets from UCI machine learning repository [3].

Presentation of the following thesis is organized as follows. In Chapter 2, we review the formulation of the spectral optimization and SVM optimization and discuss the similarity measures that can be used to cope with different types of input data. In Chapter 3, we introduce our SRA algorithm and justify our algorithm by establishing a connection between an unsupervised SVM optimization formulation and the spectral optimization. The experimental results for both synthetic data sets and real data sets are shown in Chapter 4. Concluding remarks and discussion of future work are given in Chapter 5.

Chapter 2

Spectral Clustering and Support Vector Machine

In this chapter, we briefly review the techniques of spectral clustering and support vector machine (SVM). We will also introduce several similarity matrices and kernel matrices that can be used by spectral clustering algorithms and SVM algorithms.

2.1 Spectral Clustering

Spectral clustering is a class of recent popular clustering algorithms based on eigenvalue computation. Such methods often outperform traditional k-means and hierarchical clustering algorithms [25]. Since the main objective of clustering is to separate data in a meaningful and reasonable way, the simplest and most straightforward way to conduct clustering is to partition the data set into different groups so that the overall similarity between different groups is minimized. If we represent a given data set as a graph, then the problem of finding appropriate clustering can be modeled as graph partition problems. Spectral clustering techniques, which have many variations, can be viewed as approximate solutions to graph partition problems which are often NP-hard problems. In this section, we will introduce the spectral clustering techniques and the corresponding graph partition intuition behind it. The discussions here are mainly drawn from [53] [45] and [36].

2.1.1 Graph Construction

The first step of spectral clustering is to construct a meaningful graph to represent the structure of the data set. Assume that we have a graph $G = (V, E)$, with vertices $V = \{v_1, v_2, ..., v_n\}$ where each vertex corresponds to a data instance in the given data set. The graph is usually represented by an adjacency matrix W. Typically, the graph is assumed to be undirected and the adjacency matrix W represents all pairwise edge weights. There are basically three ways to construct the adjacency matrix W and all of which rely on a meaningful similarity matrix S where S_{ij} is the similarity between data instances i and data instance j:

1. **The ϵ-neighborhood graph**: Let ϵ be a given threshhold. Here we connect two vertices v_i and v_j if the $S_{ij} \geq \epsilon$. That is we assign $W_{ij} = 1$ if $S_{ij} \geq \epsilon$. The constructed graph is an undirected unweighted graph. We can also assign weight to the edges where we assign $W_{ij} = S_{ij}$ if $S_{ij} \geq \epsilon$.

2. **The k-nearest neighbors graph**: Here we connect each vertex v_i to its k-nearest neighbors where the nearest neighbors are determined based on the similarity matrix. If we construct the graph this way, the graph will be a directed graph since the adjacency matrix W will not be symmetric. There are two ways to make the graph undirected. One is to connect vertex v_i and v_j if vertex v_j is one of the k-nearest neighbors of v_i or if vertex v_i is one of the k-nearest neighbors of v_j. The other is to connect vertex v_i and v_j only when v_j is one of the k-nearest neighbors of v_i and v_i is one of the k-nearest neighbors of v_j. Hence $W_{ij} = S_{ij}$, if v_j and v_i is connected. Otherwise, $W_{ij} = 0$. In the unweighted version, $W_{ij} = 1$, if v_j and v_i is connected.

3. **Fully connected graph**: Here we directly use the similarity matrix S as the adjacency matrix W, i.e., $W = S$. The constructed graph is an undirected weighted graph.

2.1.2 Graph Partition

After the graph construction process, we have an adjacency matrix W to describe the graph $G = (V, E)$ which captures the structure of the given data set. The next step to get meaningful clustering of the data instance is to conduct a graph partition. The goal of the graph partition is to separate the graphs into subgraphs so that an objective function, which can be viewed as a cost function, is minimized.

The simplest objective function is the minimum k-cut objective. The goal here is to minimize the summation of edge weights connecting different partitions. Let $A_1, \ldots, A_k$ be k disjoint subsets of vertex V and $\bigcup_{i=1}^{k} A_i = V$. For a minimum k-cut, we want to find $A_1, \ldots, A_k$ to minimize:

$$Cut(A_1, \cdots, A_k) := \frac{1}{2}\sum_{i=1}^{k} W(A_i, \overline{A_i})$$

where $\overline{A_i}$ is the complement of A_i and $W(A_i, \overline{A_i}) = \sum_{i \in A_i, j \in \overline{A_i}} W_{ij}$.

In practice, the solution to the minimum cut problem will often give us unsatisfying results because often one of the partitions in the solution consists of only a few data instances. Consequently, two variants, ratiocut and normalized cut, are proposed:

$$RatioCut(A_1, \ldots, A_k) := \frac{1}{2}\sum_{i=1}^{k} \frac{W(A_i, \overline{A_i})}{|A_i|}$$

$$Ncut(A_1, \ldots, A_k) := \frac{1}{2}\sum_{i=1}^{k} \frac{W(A_i, \overline{A_i})}{vol(A_i)}$$

where $|A_i|$ is the cardinality of A_i, $vol(A_i) = \sum_{i \in A_i} d_i$, and $d_i = \sum_{j=1}^{n} W_{ij}$.

The intuition behind RatioCut and Ncut is that by adding balancing factors $|A_i|$ and $vol(A_i)$ respectively, we can, to a certain extent, alleviate the problem of separating a single outlier from the rest of the data instances and therefore produce more reasonable and meaningful partitions.

2.1.3 Spectral Approximation

For the ratio cut and normalized cut variations, computing the exact solution to the graph partition problems is extremely costly because the corresponding constrained optimization problems are NP-hard integer programming problems. Therefore certain relaxations are needed in order to get approximate solutions efficiently. In this subsection, we use the normalized cut and ratio cut as examples to illustrate how to relax the integer programming problems to continuous optimization problems such that the approximate solution can be efficiently computed.

Firstly, we introduce the Laplacian matrices which are the key to the approximation process. Again, suppose we can use an undirected graph $G = (V, E)$ with vertex $V =$

$\{v_1, v_2, ..., v_n\}$ to describe the relationship between each data instance. The graph can be expressed by an (weighted) adjacency matrix W where W_{ij} is the edge weight between the vertex v_i and v_j. Furthermore, we define the degree of a vertex as the summation of the edge weights connecting vertex v_i to the other vertices $d_i = \sum_{j=1}^{n} W_{ij}$. The degree matrix $D = \text{diag}(d_1, d_2, ..., d_n)$ is then a diagonal matrix with the degree of each vertex on its diagonal. With the degree matrix and the (weighted) adjacency matrix W, we can define Laplacian matrices in three different ways:

- **unnormalized Laplacian matrix:** [53] $L = D - W$
- **normalized Laplacian matrix [45]:** $L_{rw} = I - D^{-1}W$
- **symmetric normalized Laplacian matrix [36]:** $L_{sym} = I - D^{-1/2}WD^{-1/2}$

The following properties hold for L, L_{rw} and, L_{sym}. These properties will be used to explain the ideas behind the spectral clustering methods and also our Spectral Ranking for Anomaly Detection method. Detailed proofs can be found in the tutorial [53].

Properties of Laplacian Matrices

Property 1 Suppose $L \in R^{n\times n}$, then for every vector $f \in R^n$,

$$f^T L f = \tfrac{1}{2}\sum_{i,j=1}^{n} W_{ij}(f_i - f_j)^2$$

Property 2 Suppose $L_{sym} \in R^{n\times n}$, then for every vector $f \in R^n$,

$$f^T L_{sym} f = \tfrac{1}{2}\sum_{i,j=1}^{n} W_{ij}\left(\tfrac{f_i}{\sqrt{d_i}} - \tfrac{f_j}{\sqrt{d_j}}\right)^2$$

Property 3 L and L_{sym} are symmetric positive semi-definite matrices. L_{rm} is a positive semi-definite matrix.

Property 4 L, L_{rw} and L_{sym} have n non-negative, real-valued eigenvalues $0 = \lambda_0 \leq \lambda_1 \leq ... \leq \lambda_{n-1}$. We refer to the eigenvector associated with eigenvalue $\lambda_0 = 0$ as principal eigenvector and the eigenvector associated with smallest nonzero eigenvalue as 1st non-principal eigenvector of the Laplacian matrices.

Property 5 The smallest eigenvalue of L is $\lambda_{min} = 0$ corresponding to the vector $e = [1, \ldots, 1]^T$ as eigenvector.

Property 6 Zero is an eigenvalue of L_{rw} corresponding to the vector $e = [1, \dots, 1]^T$ as the eigenvector. Zero is an eigenvalue of L_{sym} with $D^{1/2}e$ as the eigenvector.

Property 7 The eigenvalues of L_{rw} and L_{sym} are the same. If u is an eigenvector of L_{rw}, then $g = D^{1/2}u$ is an eigenvector of L_{sym}

Property 8 Let $G = (V, E)$ be an undirected graph with the adjacency matrix W. Then the multiplicity k of the eigenvalue 0 for L, L_{rm} and L_{sym} equals the number of connected components in the graph. When there is only one connected component in the graph, L, L_{rm} and L_{sym} are irreducible.

RatioCut and Unnormalized Spectral Clustering

Here we firstly consider the 2-way ratio cut problem. Suppose we have $f = (f_1, \dots, f_n)$ where

$$f_i = \begin{cases} \sqrt{|\overline{A}|/|A|}, & \text{if } v_i \in A \\ -\sqrt{|A|/|\overline{A}|}, & \text{if } v_i \in \overline{A} \end{cases} \tag{2.1}$$

From Property 1, we know

$$f^T L f = \tfrac{1}{2} \sum_{i,j=1}^{n} W_{ij}(f_i - f_j)^2$$

Let $n_1 = |A|, n_2 = |\overline{A}|$ and $n_1 + n_2 = |V| = n$, following (2.1), we would have:

$$\begin{aligned} f^T L f &= \frac{1}{2} \sum_{i,j=1}^{n} W_{ij}(f_i - f_j)^2 \\ &= \frac{1}{2} \sum_{i \in A, j \in \overline{A}} W_{ij} \left(\sqrt{\frac{n_2}{n_1}} + \sqrt{\frac{n_1}{n_2}} \right)^2 \\ &= (n_1 + n_2) \frac{1}{2} \sum_{i \in A, j \in \overline{A}} W_{ij} \left(\sqrt{\frac{n_2}{n_1(n_1 + n_2)}} + \sqrt{\frac{n_1}{n_2(n_1 + n_2)}} \right)^2 \\ &= (n_1 + n_2) \frac{1}{2} \sum_{i \in A, j \in \overline{A}} W_{ij} \left(\frac{1}{n_1} + \frac{1}{n_2} \right) \\ &= |V| \cdot \frac{1}{2} \left(\frac{W(A, \overline{A})}{|A|} + \frac{W(A, \overline{A})}{|\overline{A}|} \right) \\ &= |V| \cdot \mathit{RatioCut}(A, \overline{A}) \end{aligned}$$

Note that with definition of (2.1) we have $\|f\|_2^2 = n_1 + n_2 = |V|$ and $\sum_{i=1}^{n} f_i = 0$. Therefore, the corresponding constrained optimization problem of RatioCut is:

$$\begin{aligned} &\min_{A \subset V} f^T L f \\ \text{subject to } \; & f \perp e \; , \; \|f\|_2 = \sqrt{n} \\ & f_i \text{ is defined by equation (2.1)} \end{aligned} \tag{2.2}$$

This discrete minimization problem (2.2) is an NP-hard problem. However, if we relax the constraint that requires the solution to be discrete, we have the following relaxation,

$$\min_f f^T L f \quad \text{subject to } f \perp e, \|f\|_2 = \sqrt{n} \tag{2.3}$$

Since the eigenvector associated with the principal eigenvalue $\lambda = 0$ is $e = [1, \ldots, 1]^T$, by the Rayleigh-Ritz theorem [53], the solution to (2.3) is the eigenvector of L that corresponds to the smallest nonzero eigenvalue of Laplacian matrix.

Similarly, for k-way RatioCut problem, suppose we have an n-by-k matrix H where its element is defined as

$$h_{ij} = \begin{cases} \dfrac{1}{\sqrt{|A_j|}}, & \text{if } v_i \in A_j \\ 0, & \text{if } v_i \notin A_j \end{cases} \tag{2.4}$$

Then $RatioCut(A_1, \cdots, A_k) = \frac{1}{2} \sum_{i=1}^{k} \frac{W(A_i, \overline{A_i})}{|A_i|} = Trace(H^T L H)$. Note that with definition of (2.4) we have $H^T H = I$. Therefore, the corresponding minimization problem is equivalent to

$$\begin{aligned} &\min_{A_1, \ldots, A_k} Trace(H^T L H) \\ \text{subject to } \; & H^T H = I \\ & H \text{ is defined by equation (2.4)} \end{aligned}$$

Again this problem is an NP-hard problem because of the discrete value requirement. The relaxed minimization problem is then

$$\min_H Trace(H^T L H) \quad \text{subject to } H^T H = I \tag{2.5}$$

The solution to (2.5) is a matrix H that contains the first k eigenvectors of L, corresponding to the smallest k eigenvalue, as its columns. In unnormalized spectral clustering algorithms [53], the solution matrix H is used. Eventually, using standard k-means or hierarchical clustering algorithm, the rows of the H matrix are converted from the real valued approximate solution to the discrete partition indicator. Note that a bound on the error between relaxed problems, (2.5) and (2.3), and the exact RatioCut problems has yet to be found. However, in practice, solving the relaxed problems can often give us satisfying results.

Ncut and Normalized Spectral Clustering

For the 2-way normalized cut problem, we can define a vector u, where

$$u_i = \begin{cases} \sqrt{vol(\overline{A})/vol(A)}, & \text{if } v_i \in A \\ -\sqrt{vol(A)/vol(\overline{A})}, & \text{if } v_i \in \overline{A} \end{cases} \tag{2.6}$$

Note that with definition (2.6), we have $u^T Du = vol(V)$ and $Du \perp e$. It can be shown that the minimization of $Ncut(A, \overline{A})$ is equivalent to:

$$\begin{aligned} & \min_{A \subset V} u^T Lu \\ \text{subject to } & Du \perp e, u^T Du = vol(V) \\ & u_i \text{ is defined by equation (2.6)} \end{aligned}$$

Again, the original discrete optimization problem is NP-hard, we can relax the constraint that requires the solution u to be discrete. Let $g = D^{\frac{1}{2}}u$ and relax the discreteness constraint, the corresponding relaxed problem is:

$$\min_{g} g^T L_{sym} g \quad \text{subject to } g \perp D^{1/2}e, \|g\|_2^2 = vol(V) \tag{2.7}$$

The solution to equation (2.7) is the 1st non-principal eigenvector of L_{sym}. For arbitrary k way normalized cut, let H be a matrix where its elements is defined as

$$H_{ij} = \begin{cases} \frac{1}{vol(A_j)}, \text{ if } v_i \in A_j \\ 0, \text{ if } v_i \notin A_j \end{cases} \tag{2.8}$$

The minimization of $Ncut(A_1, ..., A_k)$ is equivalent to

$$\begin{aligned} &\min_{A_1,\ldots,A_k} Trace(H^T L H) \\ \text{subject to } & H^T D H = I \\ & H \text{ is defined by equation (2.8)} \end{aligned}$$

Relaxing the discreteness constraint and substituting $T = D^{\frac{1}{2}}H$, we get the relaxed approximation problem

$$\min_T Trace(T^T L_{sym} T) \quad \text{subject to } T^T T = I \tag{2.9}$$

Similarly, the matrix T with the first k eigenvectors of L_{sym} corresponding to the smallest k eigenvalues as its columns is the solution to (2.9). The solution matrix T is used in normalized spectral clustering algorithms [36] where the rows of T matrix are converted from the real valued approximate solution to the discrete partition indicator, using a s-tandard k-means or hierarchical clustering method. From Property 8, we know that, if g is an eigenvector of L_{sym}, then $u = D^{-1/2}g$ is an eigenvector of L_{rw}. Consequently, there is another variant of normalized spectral clustering algorithm [45] that is based on the eigenvectors of L_{rw}. Again there is no theoretical proof of how close the solutions of the relaxed problems (2.9) and (2.7) are to the optimal solutions of the original Ncut problems.

2.2 Support Vector Machine

One of the major justifications of our proposed SRA algorithms is based on connecting unsupervised support vector machine and normalized spectral clustering. In this section, we review the supervised support vector machines (SVMs), unsupervised SVMs, the kernel trick and the associated primal and dual optimization formulation. The discussions here are mainly draw from [17], [43], [52] and [25].

2.2.1 Supervised SVM

The supervised linear SVM tries to learn a linear discriminant hyperplane that separates the two classes in a given training data set. Suppose, we have n training data instances in the given data set which is represented as $D = \{(x_1, y_1), \ldots, (x_n, y_n)\}$, where $x_i \in \chi \subseteq \Re^d$ is the input data instance and $y_i \in \{-1, +1\}$ is the corresponding output target. The linear supervised SVM seeks a linear discriminant function represented as $f(x) = w^T x + b$, such that we can obtain the class label as $\widehat{y_i} = \text{sign}(f(x_i))$ and the geometric margin, which is defined as the minimum distance between data instances and the separating hyperplane $w^T x + b = 0$, is maximized. The resulting hyperplane separates the feature space into two half spaces corresponding to the two classes.

Linearly Separable Data Sets

If the data set is linearly separable, there are infinite separating hyperplanes that can separate the two classes. The SVM learns the separating hyperplane that maximizes the geometric margin between the two classes since it minimizes the bound of the generalization error, irrespective of the dimensionality [52].

Given the separating hyperplane, $w^T x + b = 0$, for a given point x_i with associated label y_i, we can project it orthogonally to the hyperplane. The projection is represented by x_0. Let γ_i be the distance between x_i and hyperplane. Then we have

$$x_i = \begin{cases} x_0 + \gamma_i \dfrac{w}{\|w\|_2}, & \text{if } w^T x_i + b \geq 0 \\ x_0 - \gamma_i \dfrac{w}{\|w\|_2}, & \text{if } w^T x_i + b < 0 \end{cases}$$

Since we know that x_0 is on the hyperplane which means $w^T x_0 + b = 0$

$$w^T x_i + b = \begin{cases} \gamma_i \|w\|_2, & \text{if } w^T x_i + b \geq 0 \\ -\gamma_i \|w\|_2, & \text{if } w^T x_i + b < 0 \end{cases}$$

Thus γ_i can be recovered by multiply y_i with $\frac{(w^T x_i + b)}{\|w\|_2}$:

$$\gamma_i = \frac{y_i(w^T x_i + b)}{\|w\|_2}$$

The functional margin of the data instance i is defined as:

$$\widehat{\gamma_i} = y_i(w^T x_i + b)$$

Note that if we rescale hyperplane associated with (w, b) to be $(\lambda w, \lambda b)$, the functional margin is rescaled but the hyperplane remain unchanged. Therefore, we do not directly optimize over the functional margin. Fixing the minimum function margin at 1, we minimize the geometric margin. The geometric margin is defined as:

$$\widetilde{\gamma} = \min_{i,\ldots,n} \gamma_i = \min_{i,\ldots,n} \frac{y_i(w^T x_i + b)}{\|w\|_2} = \frac{1}{\|w\|_2}$$

Thus, we can see that maximizing the geometric margin is equivalent to maximizing the minimum distance between data instances and the hyperplane. We can also notice that maximizing the geometric margin is equivalent to minimizing the norm of weight vector $\|w\|_2$. Consequently, the associated primal optimization problem is:

$$\begin{aligned} \min_{w,b} \quad & \frac{1}{2}\|w\|_2^2 \\ \text{subject to} \quad & y_i(w^T x_i + b) \geq 1, i = 1, \dots, n \end{aligned}$$

This is a strictly convex quadratic programming problem.

Linearly Inseparable Data Sets

In practice, the data sets are often linearly inseparable. To cope with this problem, penalized slack variables, ε_i, are introduced to allow the margin constraints to be violated. The associated optimization problem is then:

$$\begin{aligned} \min_{w,b} \quad & \frac{1}{2}\|w\|_2^2 + C\sum_{i=1}^{n} \varepsilon_i \\ \text{subject to} \quad & y_i(w^T x_i + b) \geq 1 - \varepsilon_i, i = 1, \dots, n \\ & \varepsilon_i \geq 0, i = 1, \dots, n \end{aligned} \tag{2.10}$$

Here, C is a penalty parameter which acts as a balancing factor between the margin maximization and margin violations. Note that we can also assign different penalties to different data instances. The associated optimization problem is:

$$\begin{aligned} \min_{w,b} \quad & \frac{1}{2}\|w\|_2^2 + \sum_{i=1}^{n} C_i\varepsilon_i \\ \text{subject to} \quad & y_i(w^T x_i + b) \geq 1 - \varepsilon_i, i = 1, \dots, n \\ & \varepsilon_i \geq 0, i = 1, \dots, n \end{aligned} \tag{2.11}$$

The above formulations are often referred to as C-SVM in the academic literature.

For C-SVM, little guidance is available for choosing this penalty parameter C. Therefore, cross-validation, which sometimes can be costly and inefficient, are needed in order to tune this parameter. The ν-SVM was proposed by Schölkopf et al. [44] as a modification of problem (2.10) which replaces C by a parameter ν. The parameter $0 < \nu \leq 1$ has been shown to be linked to the generalization error [44]. The primal problem for the ν-SVM is:

$$\begin{aligned}
\min_{w,b,\varepsilon,p} \quad & \frac{1}{2}\|w\|_2^2 + \frac{1}{n}\sum_{i=1}^{n}\varepsilon_i - \nu p \\
\text{subject to} \quad & y_i(w^T x_i + b) \geq p - \varepsilon_i, i = 1, \ldots, n \\
& \varepsilon_i \geq 0, i = 1, \ldots, n \\
& p \geq 0
\end{aligned} \tag{2.12}$$

Assume that $\varepsilon_i = 0, i = 1, \ldots, n$, and the optimal separating hyperplane computed from (2.12) is $w_*^T x + b_* = 0$. Then the minimum functional margin is p^* and minimum distance between the data instances and the hyperplane $w_*^T x + b_* = 0$ is $\frac{p^*}{\|w_*\|_2}$. Note that, in ν-SVM formulation (2.12), p is also an unknown variable that needs to be optimized.

Dual Formulation

Problem (2.10) and (2.12) can be transformed into their corresponding Lagrange dual problems. The optimization problems (2.10) and (2.12) are convex quadratic programming problems. Thus, the optimal solutions can also be computed from the dual problems. The dual problem leads to the usage of the kernel trick and provides more insights into the SVM formulation.

The Lagrangian of the problem (2.10) is:

$$L(w, b, \varepsilon, \alpha, \eta) = \frac{1}{2}w^T w + C\sum_{i=1}^{n}\varepsilon_i - \sum_{i=1}^{n}\alpha_i(y_i(w^T x_i + b) - 1 + \varepsilon_i) - \sum_{i=1}^{n}\eta_i\varepsilon_i \tag{2.13}$$

with $\alpha_i \geq 0$ and $\eta_i \geq 0$. The first order optimality conditions implies:

$$\begin{aligned}
\frac{\partial L}{\partial w} =& w - \sum_{i=1}^{n} y_i\alpha_i x_i = 0 \\
\frac{\partial L}{\partial \varepsilon_i} =& C - \alpha_i - \eta_i = 0 \\
\frac{\partial L}{\partial b} =& \sum_{i=1}^{n} y_i\alpha_i = 0
\end{aligned} \tag{2.14}$$

Substituting the relations in (2.14) back to (2.13), we get the dual objective function of the dual problem:

$$L(w, b, \varepsilon, \alpha, \eta) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} y_i y_j \alpha_i \alpha_j x_i^T x_j$$

The associated dual optimization formulation of (2.10) is then:

$$\begin{aligned} \min_{\alpha} \quad & \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} y_i y_j \alpha_i \alpha_j x_i^T x_j - \sum_{i=1}^{n} \alpha_i \\ \text{subject to} \quad & \sum_{i=1}^{n} \alpha_i y_i = 0 \\ & 0 \le \alpha_i \le C, i = 1, \dots, n \end{aligned} \tag{2.15}$$

The first constraint $\sum_{i=1}^{n} \alpha_i y_i = 0$ comes directly from the relations (2.14). The relation $\frac{\partial L}{\partial \varepsilon_i} = C - \alpha_i - \eta_i = 0$ and dual feasibility $\eta \ge 0, \alpha \ge 0$ imply $0 \le \alpha_i \le C$, which is the second constraint.

From the Karush-Kuhn-Tucker (KKT) conditions, the optimal solution computed from (2.15) $(w_*, b_*, \varepsilon^*, \alpha^*)$ must satisfy:

$$\begin{aligned} \alpha_i^*(y_i(w_* x_i + b_*) - 1 + \varepsilon_i^*) = 0, \;\; i = 1, \dots, n \\ \varepsilon_i^*(\alpha_i^* - C) = 0, \;\; i = 1, \dots, n \end{aligned} \tag{2.16}$$

The equation $\varepsilon_i^*(\alpha_i^* - C) = 0$ from (2.16) implies that when the slack variable $\varepsilon_i^* \neq 0$, we mush have $\alpha_i^* = C$. These points are the margin violations. In addition, $0 < \alpha_i^* < C$ implies $\varepsilon_i^* = 0$ and $y_i(w_* x_i + b_*) = 1$. These points are the points that lie on the margin hyperplanes $w_* x + b_* = \pm 1$. The points with $\alpha_i^* = 0$ are the points lying beyond the margin hyperplanes in the corresponding half space, i.e, $y_i(w_*^T x_i + b_*) > 1$. Points with non-zero α_i are referred to as support vectors.

With the optimal α^* computed directly from (2.15), the optimal weight vector can then be computed as: $w_* = \sum_{i=1}^{n} y_i \alpha_i^* x_i$. The value of b_* can be determined from any support vector with $0 < \alpha_i^* < C$ by solving $y_i(w_*^T x_i + b_*) = 1$.

Similarly, the dual problem of ν-SVM is [44]:

$$\begin{aligned} \min_{\alpha} \quad & \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} y_i y_j \alpha_i \alpha_j x_i^T x_j \\ \text{subject to} \quad & 0 \leq \alpha_i \leq \frac{1}{n}, i = 1, \ldots, n \\ & \sum_{i=1}^{n} \alpha_i y_i = 0 \\ & \sum_{i=1}^{n} \alpha_i \geq \nu \end{aligned} \tag{2.17}$$

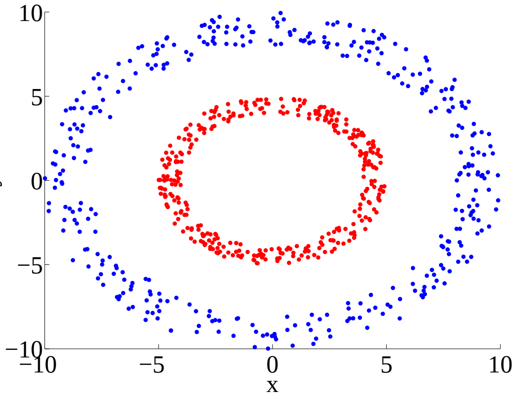
Let the optimal solution from problem (2.17) be α^*. We denote the set of support vectors as $SV = \{x_i | \alpha_i^* \neq 0\}$ and set of margin violations as $\overline{S} = \{x_i | \varepsilon_i \neq 0\}$. It can be shown that the parameter ν is an upper bound on the fraction of margin violations and a lower bound of the fraction of support vectors. Detailed proof can be found in [44]. Therefore, compared the penalty parameter C in C-SVM, the penalty parameter ν in ν-SVM is more meaningful, which facilitates tuning of the tuning of parameter.

2.2.2 Kernel Trick

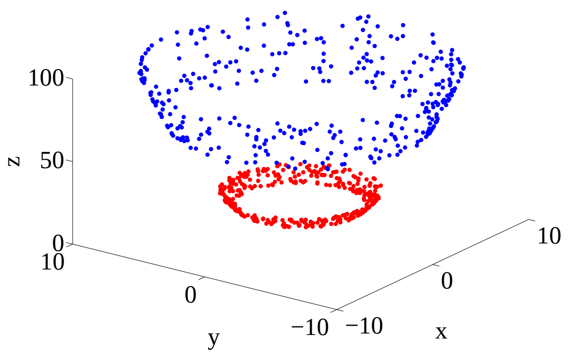
In the previous discussion, we focus on the linear SVMs where we assume that the decision boundary is linear in the original feature space. In practise, we often encounter cases where a decision boundary needs to be non-linear in the original space as illustrated in Figure 2.1 (a). However, by mapping to a derived feature space with a higher dimensionality, a linear separating hyperplane that successfully separates the two classes can possibly be defined. Consider the example in Figure 2.1 (a), with its two concentric rings. We can add a third dimension $z = x^2 + y^2$ to the original data set. The resulting data set in $\Re^3$ is linearly separable by a hyperplane. Thus, provided that we work in this $\Re^3$ space, we can train a linear SVM classifier that successfully finds a good linear decision boundary. Unfortunately, in most of real life cases, explicitly finding a derived high dimensional feature space is not as easy as this example. Thus, kernels are employed. Kernels allow us to implicitly work in a high dimensional derived feature space. Formally, we define a feature map as below:

Definition 1. A feature map is a function $\theta : \chi \subseteq \Re^d \to \mathcal{H} \subseteq \Re^{d'}$ that maps the input space χ to a derived feature space $\mathcal{H}$:

$$x = (x_1, \ldots, x_d) \longmapsto \theta(x) = (\theta_1(x), \ldots, \theta_{d'}(x))$$



(a) Non-linear in $\Re^2(x,y)$



(b) Mapping to $\Re^3(x,y,x^2+y^2)$

Figure 2.1: Illustration of Kernel Trick

The feature map enables us to learn a linear SVM in a new space where the separating hyperplane is $w^T\theta(x)+b=0$. The decision function $f(x)=w^T\theta(x)+b$, when mapping back to the original space, is usually non-linear.

Definition 2. A kernel is a function $\mathcal{K}:\chi\times\chi\rightarrow\Re$ that for all $x,\widehat{x}\in\chi$:

$$\mathcal{K}(x,\widehat{x})=\theta(x)^T\theta(\widehat{x})$$

A kernel function implicitly compute the inner product in a derived space. Thus for any algorithm that only relies on the inner product, e.g SVM, the kernel can be applied to provide the inner product in a derived space. This is known as the 'kernel trick'.

Definition 3. Given a kernel function $\mathcal{K}:\chi\times\chi\rightarrow\Re$ and input data set $X=\{x_1,\ldots,x_n\}$ where $x_i\in\chi$ the matrix

$$K=\begin{pmatrix}\mathcal{K}(x_1,x_1) & \mathcal{K}(x_1,x_2) & \ldots & \mathcal{K}(x_1,x_n)\\ \mathcal{K}(x_2,x_1) & \ddots & \ldots & \vdots\\ \vdots & \vdots & \ddots & \vdots\\ \mathcal{K}(x_n,x_1) & \mathcal{K}(x_n,x_2) & \ldots & \mathcal{K}(x_n,x_n)\end{pmatrix}$$

is called the kernel matrix for the input data X.

We can use the following condition to test whether a specific function $\mathcal{K}(x,\widehat{x})$ is a valid kernel function.

Theorem 1 (Mercer's Condition)**.** A symmetric function $\mathcal{K} : \chi \times \chi \to \Re$ can be expressed as an inner product

$$\mathcal{K}(x, \widehat{x}) = \theta(x)^T \theta(\widehat{x})$$

for some feature map θ if and only if

$$\int \mathcal{K}(x, \widehat{x}) g(x) g(\widehat{x}) dx d\widehat{x} \geq 0 \text{ for any function } g(x) \in L_2(\chi).$$

where $L_2(\chi)$ denotes the space of square-integrable function or, equivalently, the corresponding kernel matrix

$$K = \begin{pmatrix} \mathcal{K}(x_1, x_1) & \mathcal{K}(x_1, x_2) & \dots & \mathcal{K}(x_1, x_n) \\ \mathcal{K}(x_2, x_1) & \ddots & \dots & \vdots \\ \vdots & \vdots & \ddots & \vdots \\ \mathcal{K}(x_n, x_1) & \mathcal{K}(x_n, x_2) & \dots & \mathcal{K}(x_n, x_n) \end{pmatrix}$$

is positive semi-definite for any collection of $\{x_1, x_2, \dots, x_n\}$ where $x_i \in \chi$.

Consider dual problems for linear C-SVM (2.15) and ν-SVM (2.17) in the derived feature space $\mathcal{H}$. Given a kernel function $\mathcal{K}(x, \widehat{x})$ or the kernel matrix K for the train data set, we can utilize the 'kernel trick'. The resulting problem for C-SVM is:

$$\begin{aligned} \min_{\alpha} \quad & \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} y_i y_j \alpha_i \alpha_j \mathcal{K}(x_i, x_j) - \sum_{i=1}^{n} \alpha_i \\ \text{subject to} \quad & \sum_{i=1}^{n} \alpha_i y_i = 0 \\ & 0 \leq \alpha_i \leq C, \ i = 1, \dots, n \end{aligned} \tag{2.18}$$

The resulting problem for ν-SVM is then:

$$\begin{aligned} \min_{\alpha} \quad & \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} y_i y_j \alpha_i \alpha_j \mathcal{K}(x_i, x_j) \\ \text{subject to} \quad & 0 \leq \alpha_i \leq \frac{1}{n}, i = 1, \dots, n \\ & \sum_{i=1}^{n} \alpha_i y_i = 0 \\ & \sum_{i=1}^{n} \alpha_i \geq \nu \end{aligned} \tag{2.19}$$

By solving these problems, we are learning a linear SVM in a new space implicitly defined by the kernel function $\mathcal{K}(x, \widehat{x})$, which usually corresponds to a non-linear decision boundary in the original space. Although usually SVMs work with kernels that satisfy Mercer's condition, indefinite kernels, which are the kernels that violate Mercer's condition, are also widely used [32, 23]. In subsequent sections, when we say a kernel is a valid kernel we mean that the kernel satisfies Mercer's condition.

2.2.3 Unsupervised SVM

In the previous discussion, we have reviewed the standard supervised SVMs which seek to learn a discriminant hyperplane that separates the two classes in a given training data set while maximizing the geometric margin. The idea can be generalized to the unsupervised setting where the objective becomes determining the optimal labels based on a margin maximization C-SVM. The basic idea is to optimally choose the labels so that the corresponding margin maximization optimal hypothesis yields the minimum C-SVM objective [14]. Thus we solve the following problem:

$$\min_{y_i \in \{+1,-1\}} \left\{ \begin{array}{cc} \max_\alpha & e^T\alpha - \frac{1}{2}\alpha^T YKY\alpha \\ \text{subject to} & y^T\alpha = 0 \\ & 0 \leq \alpha_i \leq C, i = 1, \ldots, n \end{array} \right\} \tag{2.20}$$

where K is the kernel matrix for the input data set and $Y = \text{diag}(y_1, \ldots, y_n)$. Now let us introduce a new variable z as:

$$z_i = \alpha_i y_i, \quad i = 1, ..., n$$

For any $\alpha_i \neq 0$,we have $y_i = \text{sign}(z_i), i = 1, ..., n$. Therefore, we have

$$\alpha^T YKY\alpha = z^T Kz, e^T\alpha = e^T|z| \text{ and } y^T\alpha = e^T z.$$

where $|z| = (\ |z_1|, \ldots, |z_n|\ )$. Thus problem (2.20) can be further be rewritten as:

$$\min_{y_i \in \{+1,-1\}} \left\{ \begin{array}{rl} \max\limits_z & e^T|z| - \dfrac{1}{2}z^T Kz \\ \text{subject to} & e^T z = 0 \\ & |z_i| \leq C, \quad i = 1, \ldots, n \end{array} \right\} \tag{2.21}$$

Note that finding the global optimal solutions of the problem (2.20) and (2.21) is NP-hard. In order to solve them, relaxation and approximation are needed.

2.3 Similarity Matrices and Kernel Matrices

In this subsection, we review several similarity and kernel matrices that can be used by spectral clustering algorithms and SVM algorithms. We will also briefly discuss how to deal with numerical data, categorical data and mixing data types. Note that kernel matrices can, sometimes, be viewed as special similarity matrices. For example, for a given Gaussian kernel K, the value of entry K_{ij} is inversely related to the distance between the data instance i and j.

Numerical data is quite common in real life. Table 2.1 summarizes several commonly used kernel functions for numerical data sets. Gaussian kernel is a popular kernel for kernel

Table 2.1: Commonly Used Kernel Function for Numerical Data Sets

Kernel Type	$\mathcal{K}(x, \widehat{x})$
Linear	$x^T \widehat{x}$
P-polynomial	$(x^T \widehat{x} + b)^p$
Gaussian	$e^{-\frac{\|x-\widehat{x}\|_2^2}{2\sigma^2}}$
Spline	$\Pi_{i=1}^d (1 + x_i\widehat{x}_i + x_i\widehat{x}_i \min(x_i, \widehat{x}_i) - \frac{x_i+\widehat{x}_i}{2} \min(x_i, \widehat{x}_i)^2 + \frac{\min(x_i,\widehat{x}_i)^3}{3})$
Wavelet	$\frac{\theta}{\|x-\widehat{x}\|_2} \sin \frac{\|x-\widehat{x}\|_2}{\theta}$

based algorithms, especially when we do not have expert knowledge of the data sets. It is also the main similarity matrix that is used in spectral clustering techniques.

In practice, data sets can also contain large portion of categorical and ordinal data. A common practice is to preprocess data sets to numerical forms which includes treating a direct numerical representation of such data as numerical data, or expanding categorical features into a set of binary indicators. However, these practices often lead to poor pattern recognitions because the structures of the data sets are often not captured appropriately. This crucial understanding is often missing in practical applications as most existing machine learning algorithms solely focus on numerical values.

For categorical data sets, appropriate similarity measures can be applied to explore the structures of the data sets, see, e.g., [5] and references therein. In subsequent discussion, we focus on the simplest similarity measure, overlapping similarity and its derived similarity measures which can be applied directly to categorical data. These similarity measures are defined based on the set of possible values for each specific categorical feature where match and mismatch in categorical values form an intuitive basis for comparing patterns within the data sets. In the subsequent discussion, we assume that the data set comes from sampling of random d-dimensional categorical vector D, with the ith feature D_i having $|D_i|$ distinct values, $i = 1, 2, \cdots, d$.

Overlapping Similarity

Given two d dimensional categorical feature vector x and $\widehat{x}$, the Hamming distance is defined as the number of features for which the nominal values of x and $\widehat{x}$ do not match divided by the total number of features:

$$d^H(x, \widehat{x}) = \frac{\sum_{i=1}^{d} \delta(x_i, \widehat{x}_i)}{d}$$

where

$$\delta(x_i, \widehat{x}_i) = \begin{cases} 1, & x_i \neq \widehat{x}_i \\ 0, & x_i = \widehat{x}_i \end{cases}$$

Consequently overlapping similarity, which is also called as Jaccard index, is given by

$$s^O(x, \widehat{x}) = 1 - d^H(x, \widehat{x})$$

It has been proven that overlapping similarity (Jaccard Index) is a valid kernel function. Detailed proof can be found in [22]. For data instance x and $\widehat{x}$ with numerical features, a standard Gaussian kernel has the form

$$\mathcal{K}(x, \widehat{x}) = e^{-\frac{\|x-\widehat{x}\|_2^2}{2\sigma^2}}$$

where $\sigma > 0$ is a constant bandwidth. We can derive a Gaussian kernel for categorical data from the overlapping similarity.

$$\mathcal{K}^{GH}(x, \widehat{x}) = e^{-\frac{d^H(x,\widehat{x})}{2\sigma^2}} \tag{2.22}$$

The fact that the associated Gaussian kernel is still a valid kernel function can be proven by the following theorem [28]:

Theorem 2. Let χ be a nonempty set and $f : (\chi \times \chi) \to \Re$ be a function. The function $exp(-tf(x, \widehat{x}))$ is positive semi-definite for $t > 0$ if and only if f is negative semi-definite.

Proof. Detailed proof can be found in chapter 3, Theorem 2.2 [16]. □

Note that, since overlapping similarity (Jaccard Index) is a valid kernel function (positive semi-definite), it can be easily seen that $-s^O(x, \widehat{x})$ is negative semi-definite. Therefore, following Theorem 2, we see that

$$\mathcal{K}^{GH}(x, \widehat{x}) = e^{-\frac{d^H(x,\widehat{x})}{2\sigma^2}} = e^{\frac{-1}{2\sigma^2}} \left(e^{-\frac{-s^O(x,\widehat{x})}{2\sigma^2}} \right)$$

is positive semi-definite, which satisfies the Mercer's condition and is a valid kernel function.

Adaptive Gaussian Kernel

In the Gaussian kernel with the Hamming distance (2.22), a single bandwidth σ is applied to every data instance. In [18] the authors argue that performance of spectral clustering on categorical data sets can be improved using an adaptive bandwidth. In addition, a weighted Hamming distance

$$d^{WH}(x, \widehat{x}) = \sum_{i=1}^{d} \frac{\delta(x_i, \widehat{x}_i)}{|\mathcal{D}_i|}$$

is proposed in [18], where $|\mathcal{D}_i|$ is the number of distinct values in that dimension. Corresponding to the weighted Hamming distance, an adaptive kernel

$$\mathcal{K}^{WH}(x, \widehat{x}) = e^{-\frac{d^{WH}(x,\widehat{x})}{2\sigma^2(x,\widehat{x})}}$$

can be defined, where $\sigma(x, \widehat{x})$ is a data driven adaptive bandwidth uniquely determined by the k nearest neighbors of data instances x and $\widehat{x}$. Detailed discussion on the data driven adaptive bandwidth can be found in [18]. In the subsequent section, we use β to denote the neighborhood size parameter of this kernel in order to avoid confusion.

Hamming Distance Kernel

Hamming distance kernel [15] is proposed as a variant of the string kernel [31]. The basic idea is to explicitly create a high dimensional feature space, with each dimension representing a possible feature nominal value combination. The kernel function $\mathcal{K}(x, y)$ equals to the inner product of two vectors in the high dimensional feature space mapped from x and y. Because the resulting kernel function is explicitly defined based on the inner product in the high dimensional space, the Hamming distance kernel is also a valid kernel satisfying the Mercer's condition.

Formally, let $\mathcal{D}_i$ be the domain of the ith feature. Let $\mathcal{D}^n$ be the cross product over all feature domains. Thus, for each $q \in \mathcal{D}^n$, given a categorical instance $x = (x_1, ..., x_n)$, $x_i \in \mathcal{D}_i$, we define an explicit mapping: $\theta_q(x) = \tau^{d^H(q,x)}$, where $\tau \in (0, 1)$. Note that $\theta_q(x)$ is just one dimension of the derived feature space. Each $q \in \mathcal{D}^n$ represents one dimension in the derived feature space and an explicit mapping is defined to map any instance x into this dimension, i.e., $\theta_q(x)$. Thus the Hamming distance kernel between instances x and $\widehat{x}$ is:

$$\mathcal{K}^H(x, \widehat{x}) = \sum_{q \in \mathcal{D}^n} \theta_q(x)\theta_q(\widehat{x}) \tag{2.23}$$

Directly computing the Hamming distance kernel has an exponential computational complexity. Fortunately a dynamic programming technique can be applied, which allows this kernel to be computed efficiently following the recursive procedure below [15]:

$$\begin{aligned}
&\mathcal{K}^0(x, \widehat{x}) = 1 \\
&\mathcal{K}^j(x, \widehat{x}) = (\tau^2(|\mathcal{D}_j| - 1 - \delta(x_j, \widehat{x}_j)) + (2\tau - 1)\delta(x_j, \widehat{x}_j) + 1)\mathcal{K}^{j-1}(x, \widehat{x}), \quad j = 1, \cdots, d
\end{aligned}$$

and $\mathcal{K}^H(x, \widehat{x}) = \mathcal{K}^d(x, \widehat{x})$. While in evaluating similarity between a pair of instance x and $\widehat{x}$, Hamming distance directly compares nominal values of x and $\widehat{x}$ to determine the number of different values. In contrast, a Hamming distance kernel assesses the similarity between x and $\widehat{x}$ in referencing to all possible nominal value combinations of the categorial attribute. Consequently, a Hamming distance kernel can capture more information than the simple Hamming distance (overlapping similarity).

It is also common that data sets contain both numerical attributes and categorical attributes. There are two common approaches to obtain similarity and/or kernel matrices for data sets with mixing data types:

1. Convert categorical attributes to numerical attributes (binarization) or numerical attribute to categorical attributes (discretization). Then compute the kernel or similarity matrices designed for pure numerical data or categorical data.

2. Compute the kernel or similarity matrices K_1 designed for pure numerical data using only numerical attributes and compute the kernel or similarity matrices K_2 designed for pure categorical data using only numerical attribute. The final similarity or kernel matrix is the $K = \alpha K_1 + \beta K_2$. In the cases of supervised learning such as SVM, the coefficient α and β can be determined by cross validation [46].

Chapter 3

Spectral Ranking for Anomaly Detection

In this chapter, we propose the Spectral Ranking for Anomaly Detection (SRA) algorithm. In Section 3.1, we review some insights for the 1st non-principal eigenvector of L_{sym}, which is the eigenvector corresponding to the smallest nonzero eigenvalue. Based on these insights for the 1st non-principal eigenvector of L_{sym}, the SRA algorithm is proposed in Section 3.2.

3.1 Understanding the 1st Non-principal Eigenvector

3.1.1 Degree of Support Perspective

Nonconvex Relaxation of the Unsupervised SVM: Recall that the unsupervised SVM can be formulated as:

$$\min_{y_i \in \{+1,-1\}} \left\{ \begin{array}{cc} \max_\alpha & e^T\alpha - \frac{1}{2}\alpha^T Y K Y \alpha \\ \text{subject to} & y^T\alpha = 0 \\ & 0 \le \alpha_i \le C, i = 1, \ldots, n \end{array} \right\} \tag{3.1}$$

The inner optimization problem can be rewritten as:

$$\begin{aligned} \max_z \ & e^T|z| - \frac{1}{2}z^T K z \\ \text{subject to } & e^T z = 0 \\ & |z_i| \le C, \quad i = 1, \ldots, n \end{aligned} \tag{3.2}$$

where $z_i = \alpha_i y_i, \quad i = 1, ..., n$. We note that $e^T|z|$ is convex and (3.2) has many local maximizers. Consider the following nonconvex quadratic programming problem:

$$\begin{aligned} \min_z \quad & -\frac{1}{2} z^T K z \\ \text{subject to} \quad & e^T z = 0 \\ & |z_i| \leq C, \;\; i = 1, \ldots, n \end{aligned} \tag{3.3}$$

Assume that K is positive definite in the space $\{z : e^T z = 0\}$. Then the local minimizers of (3.3) are at the boundary of $|z_i| \leq C, \;\; i = 1, \ldots, n$. Theorem 3 suggests that problem (3.3) can be considered as a nonconvex relaxation of problem (3.1).

Theorem 3. Suppose that K is symmetric positive definite. Let(α^*,y^*) be a solution to the unsupervised SVM (3.1). Assume that the solution z^* to (3.3) is a local maximizer of (3.2) and satisfies $e^T|z^*| = e^T\alpha^*$. Then $a_z^* = |z^*|$ and $y_z^* = \text{sign}(z^*)$ solves the unsupervised SVM (3.1).

A proof of Theorem 3 is shown in the Appendix A. Note that with Theorem 3, we can see that (3.3) can be viewed as a nonconvex relaxation of the unsupervised support vector machine. However, finding the global optimal solution of (3.3) remains a NP-hard problem.

Connecting the Unsupervised SVM with the Spectral Optimization: Consider the normalized spectral clustering formulation for approximating the 2-way normalized cut:

$$\begin{aligned} \min_g \quad & g^T L_{sym} g \\ \text{subject to} \quad & g \perp D^{1/2} e \\ & \|g\|_2^2 = vol(V) \end{aligned} \tag{3.4}$$

This formulation can be rewritten as:

$$\begin{aligned} \min_g \quad & -g^T D^{-1/2} W D^{-1/2} g \\ \text{subject to} \quad & g^T (D^{1/2} e) = 0 \\ & g^T g = vol(V) \end{aligned}$$

Let $\overline{z} = D^{\frac{1}{2}} g$, we can further rewrite the formulation (3.4) as:

$$\begin{aligned} \min_{\overline{z}} \quad & -\overline{z}^T D^{-1} W D^{-1} \overline{z} \\ \text{subject to} \quad & e^T \overline{z} = 0 \\ & \overline{z}^T D^{-1} \overline{z} = vol(V) \end{aligned}$$

If we assume W is positive definite in the space $\{\overline{z} : (D^{\frac{1}{2}}e)^T\overline{z} = 0\}$, we can replace the second equality constraint with inequality constraint since the ellipsoidal constraint should be active at a solution. Let $\overline{K} = D^{-1}WD^{-1}$. Then (3.4) can the be rewritten as

$$\begin{aligned} \min_{\overline{z}} \quad & -\overline{z}^T\overline{K}\overline{z} \\ \text{subject to} \quad & e^T\overline{z} = 0 \\ & \overline{z}^T D^{-1}\overline{z} \leq vol(V) \end{aligned} \tag{3.5}$$

Problem (3.5) can be approximated by the nonconvex problem below

$$\begin{aligned} \min_{z} \quad & -z^T\overline{K}z \\ \text{subject to} \quad & e^Tz = 0 \\ & |z_i| \leq \overline{C_i}, \quad i = 1, \ldots, n \end{aligned} \tag{3.6}$$

where $\overline{C_i} = \sqrt{d_i} \cdot vol(V)$ and d_i is the degree of the data instance i. We can now observe that problem (3.3) and problem (3.6) have the similar form. Since (3.5) is an approximation to the normalized 2-cut optimization problem, this suggests that the normalized spectral optimization problem can be regarded as an approximation to the unsupervised SVM problem with the kernel $\overline{K} = D^{-1}KD^{-1}$ and $\overline{C_i} = \sqrt{d_i} \cdot vol(V)$.

We note that in connecting the spectral optimization with the unsupervised SVM (3.1), the following assumptions are made:

- The solution z^* to (3.3) is a local maximizer of (3.2) and satisfies $e^T|z^*| = e^T\alpha^*$ where (α^*, y^*) solves the unsupervised SVM (3.1).
- The matrix W is positive definite in the space $\{z : (D^{\frac{1}{2}}e)^Tz = 0\}$.
- The rectangular constraint in (3.3) is approximated by the ellipsoidal constraint in (3.5).

Note that the optimal separating hypothesis from the unsupervised SVM has the form

$$f(x) = \left(\sum_{j=1}^{n} y_j |z_j| \mathcal{K}(x, x_j) + b \right)$$

where $y_j = \text{sign}(z_j)$ and $|z_j|$ denotes the measure of the strength of support from the jth data point on the two classes separation decision.

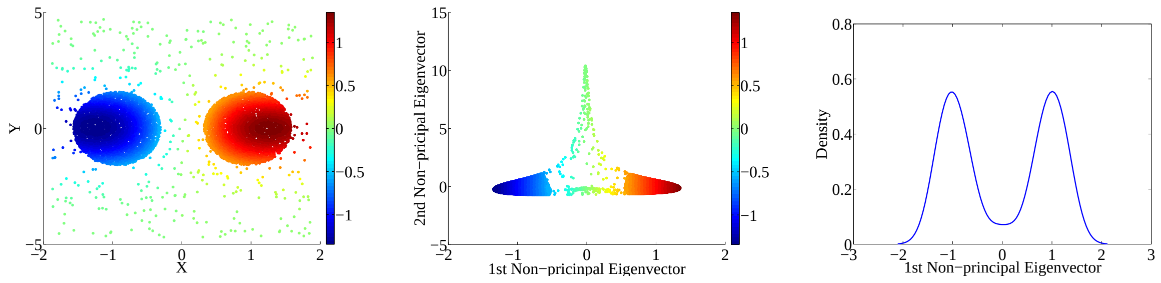

(a) Synthetic Data (b) Visualization of $\overline{z_1} = D^{1/2}g_1$ and $\overline{z_2} = D^{1/2}g_2$ (c) Density Distribution of $\overline{z_1} = D^{1/2}g_1$

Figure 3.1: Interpretation of 1st Non-principal Eigenvector For Synthetic Data 1

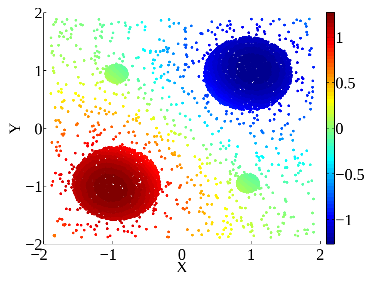
Above discussion suggests that the spectral clustering optimization (3.5) can be viewed as an approximation to the unsupervised SVM. Therefore, the variable $\overline{z_1} = D^{\frac{1}{2}}g_1$, where g_1 is the 1st non-principal eigenvector of L_{sym}, approximates the optimal solution for unsupervised SVM and $|(\overline{z_1})_j|$ approximates the strength of support from the jth data point on the two classes separation decision.

We note that, due to the use of the ellipsoidal constraint rather than rectangular constraint and other approximations, $\overline{z}$ is different from the vector of support computed from the exact unsupervised SVM. In particular, the eigenvector is likely to have no zero components, i.e., every data point offers support to some degree in the two classes separation.

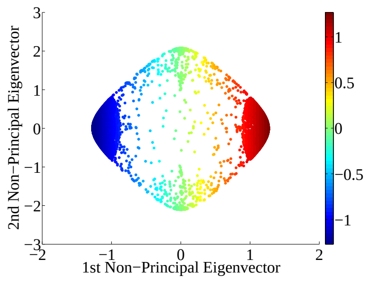
With the degree of support interpretation, supposing we have two major clusters, we can expect anomalies, no matter whether they are point anomalies or anomalies belonging to small clusters, will offer less support in separating the two major clusters. Therefore, for an abnormal data instance i and a data instance j belonging to the major clusters, it is highly likely that $|(\overline{z_1})_i|$ will be smaller than $|(\overline{z_1})_j|$.

Here we present two synthetic data sets in $\Re^2$, which are shown in Figure 3.1 (two large clusters with point anomalies) and 3.2 (two large clusters with small clusters and point anomalies), to illustrate this insight. We first use a Gaussian kernel with the bandwidth $\sigma = 1$ as the similarity matrix S. The graph constructed is a fully connected graph where the adjacency matrix W equals to the similarity matrix S.

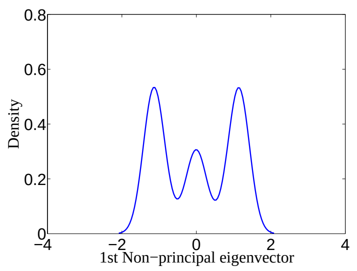
Subplot (a) demonstrates the original 2-D data while subplot (b) demonstrates points in the space of the $\overline{z_1} = D^{1/2}g_1$ and $\overline{z_2} = D^{1/2}g_2$. To depict the meaning of the value of the first non-principal eigenvector, we assign a unique color to each value with the darkest red color representing the largest positive value and darkest blue representing the most



(a) Synthetic Data



(b) Visualization of $\overline{z_1} = D^{1/2}g_1$ and $\overline{z_2} = D^{1/2}g_2$



(c) Density Distribution of $\overline{z_1} = D^{1/2}g_1$

Figure 3.2: Interpretation of 1st Non-principal Eigenvector For Synthetic Data 2

negative value. Each point in the original plot (a) then inherits its designated color value of the corresponding component of the first non-principal eigenvector. The color map is shown at the right side of the subplot (a) and (b) in Figure 3.1 and Figure 3.2. The density distribution of the vector $\overline{z_1} = D^{1/2}g_1$ is shown in subplot (c). We point out that, for these two examples, the vector $\overline{z_2} = D^{1/2}g_2$ is just used for visualization. No information in the vector $\overline{z_2} = D^{1/2}g_2$ is used.

From Figure 3.1 (a), it is easy to see that the corresponding strength of support $|(\overline{z_1})_i|$ of point anomalies, colored in yellow and green, is the smallest. Since the data points in the light blue and light red regions are closer to the other cluster, they offer less information in defining the cluster. Thus the strength of support of those data instances $|(\overline{z_1})_i|$ is smaller than that of the other data points, colored as dark blue and dark red. Also from the density distribution shown in Figure 3.1 (c), we note that there are two major peaks and one valley. The two peaks correspond to the two major clusters and the valley corresponds to the point anomalies.

In Figure 3.2, we show the example with two major clusters, two small clusters for anomalies, together with point anomalies. It is clear that the strength of support $|(\overline{z_1})_i|$ of point anomalies and small clusters for anomalies is the smallest. We can also see, from the darkness of the red and blue colors, that the edges of the major clusters have relatively smaller strength of support $|(\overline{z_1})_i|$ than the cores of the major clusters. We can also observe from Figure 3.2 (c) that there are two major peaks, one small peak and two valleys. The two peaks correspond to the two major clusters, the small peak corresponds to the two small clusters, and the two valleys correspond to the point anomalies. Thus, we can see that $|\overline{z_1}|$ provides a reasonable basis for detecting the point anomalies and also anomalies belonging to small clusters when two main clusters exist.

3.1.2 Random Walk Perspective

Degree of support is not the only view for the 1st non-principal eigenvector of L_{sym}. Wang and Davidson [54] has proposed a similar measure from a random walk perspective. The following discussion of the alternative random walk perspective is mainly drawn from [54].

We start with the similar graph construction discussed in Section 2.1.1 to obtain an (weighted) adjacency matrix W which captures the structure of the given data set. Similarly, we define a degree matrix D with $d_i = \sum_{j=1}^{n} W_{ij}$ on its diagonal. The data set is represented as a graph $G = (V, E)$ with $|V| = n$. A random walk of length t on a graph $G(V, E)$ starting from an initial node r can be represented by a sequence of state random variable $X^0, \ldots, X^t$, where $X^i = j$ indicates that the random walk starting from the initial node r visiting node j at step i. We assume that the random walk is a memoryless Markov process where the next state depends entirely on the current state. Mathematically, the memoryless Markov property implies:

$$P(X^{t+1} = i | X^t = j, X^{t-1} = k, \ldots, X^0 = r) = P(X^{t+1} = i | X^t = j)$$

Therefore a transition matrix $P = WD^{-1}$ can be defined where the entries in P represents the probability of transition from one node to another node in graph $G(V, E)$ at any step t, i.e,

$$p_{ij} = P(X^{t+1} = i | X^t = j), \forall i, j, 1 \leq i, j \leq n$$

We assume that $P(X^{t+1} = i | X^t = j)$ is the same at any step t. The Markov random walk described above can be characterized by the stationary distribution π. The stationary distribution π is a time invariant measure that has been used widely in applications such as anomaly detection [35] and pagerank [38]. Given a graph $G = (V, E)$ and the associated transition matrix P, there exists a unique probability distribution $\pi = (\pi_1, \ldots, \pi_n)^T$ that satisfies the following equation:

$$\pi_j = \sum_{i=1}^{n} \pi_i p_{ij}, \quad \forall j, \quad 1 \leq j \leq n$$

This unique probability distribution is referred to as stationary distribution.

It can be shown that the eigenvector $\overline{z}_0$ corresponding to the largest eigenvalue (which is always 1) of matrix P equals the stationary distribution. For a point anomaly i since it is far from the rest of the data instances, the probability that a random walk in the graph visits the node i from any other node j at any step t, p_{ij}, is smaller. Thus, for a point anomaly i, the corresponding π_i is likely to be smaller than other data instances that belong to certain clusters. The stationary distribution is used in [35] to detect point anomalies. We refer to the the eigenvector $\overline{z}_0$ as the principal eigenvector of matrix P.

The limitation of using stationary distribution π for anomaly detection is that it is based on a global random walk. Therefore, it is only useful to detect point anomalies which are the data instances that are less likely to be visited no matter where the random walk starts. Wang and Davidson [54] propose to use non-principal eigenvectors from the transition matrix P for anomaly detection. Suppose we have two disjoint subset S^+ and S^- where $S^+ \bigcup S^- = V$ and $S^+ \bigcap S^- = \emptyset$, we can define an indicator variable as follows: If node i is not visited at step t in a random walk, we set Y_i^t to 0. If node i is visited at step t and the random walk starts from nodes in S^+, we set Y_i^t to 1. If node i is visited at step t and the random walk starts from nodes in S^-, we set Y_i^t to -1.

$$Y_i^t = \begin{cases} 1, & X^t = i, X^0 \in S^+ \\ -1, & X^t = i, X^0 \in S^- \\ 0, & \text{otherwise} \end{cases} \tag{3.7}$$

Consequently, the expectation $E(Y_i^t) = P(X^t = i, X^0 \in S^+) - P(X^t = i, X^0 \in S^-)$ measures the difference in the likelihood the of node i being visited from S^+ and S^-. For example, the expectation $E(Y_i^t) = P(X^t = i, X^0 \in S^+) - P(X^t = i, X^0 \in S^-)$ being closer to 1 suggests that node i would be visited from set S^+ more likely. The expectation $E(Y_i^t) = P(X^t = i, X^0 \in S^+) - P(X^t = i, X^0 \in S^-)$ being closer to 0 suggests that node i would be visited from both sets almost equally likely.

Wang and Davidson [54] suggest that S^+ and S^- can be regarded as two cluster based contexts and those data instances that can be visited from both sets almost equally likely are the contextual anomalies. The random walk in $G = (V, E)$ with respect to contexts is called a contextual random walk. Thus the expectation $E(Y_i^t)$ can be viewed as a score for a contextual anomaly. However, the $E(Y_i^t)$ is not time invariant and will change as time step t increase. Since generating anomaly ranking score requires a time invariant measure, Wang and Davidson [54] define a time invariant measure which is based on $E(Y_i^t)$.

Definition 4 (Stationary Expectation [54])**.** Given a graph $G = (V, E)$, its two disjoint subset S^+ and S^- where $S^+ \bigcup S^- = V$ and $S^+ \bigcap S^- = \emptyset$, and its transition matrix P, we say that the expectation $E(Y_i^t)$, which is represented as μ_i, is stationary if the following condition holds:

$$\mu_i = C \sum_{j=1}^{n} \mu_j p_{ij} \text{ , where } C \text{ is a constant.} \tag{3.8}$$

We refer to $\mu = \{\mu_1, ..., \mu_n\}$ as the stationary expectation of the contextual random walk with respect to S^+ and S^-.

Suppose $P(X^0 = i) = w_i$ is the probability that the random walk starting from a specific node i. We have:

$$\begin{aligned} P(X^1 = i, X^0 \in S^+) &= \sum_{j \in S^+} P(X^0 = j) P(X^1 = i | X^0 = j) = \sum_{j \in S^+} w_j p_{ij} \\ P(X^1 = i, X^0 \in S^-) &= \sum_{j \in S^-} P(X^0 = j) P(X^1 = i | X^0 = j) = \sum_{j \in S^-} w_j p_{ij} \end{aligned} \tag{3.9}$$

Therefore,

$$E(Y_i^1) = \sum_{j \in S^+} w_j p_{ij} - \sum_{j \in S^-} w_j p_{ij} \tag{3.10}$$

Similarly, we can see that

$$P(X^{t+1} = i, X^0 \in S^+) = \sum_{j=1}^{n} P(X^t = j, X^0 \in S^+) P(X^{t+1} = i | X^t = j)$$

and

$$P(X^{t+1} = i, X^0 \in S^-) = \sum_{j=1}^{n} P(X^t = j, X^0 \in S^-) P(X^{t+1} = i | X^t = j)$$

Therefore,

$$\begin{aligned} E(Y_i^{t+1}) &= \sum_{j=1}^{n} P(X^t = j, X^0 \in S^+) P(X^{t+1} = i | X^t = j) \\ &\quad - \sum_{j=1}^{n} P(X^t = j, X^0 \in S^-) P(X^{t+1} = i | X^t = j) \\ &= \sum_{j=1}^{n} \left[P(X^t = j, X^0 \in S^+) - P(X^t = j, X^0 \in S^-) \right] P(X^{t+1} = i | X^t = j) \\ &= \sum_{j=1}^{n} E(Y_j^t) p_{ij} \end{aligned} \tag{3.11}$$

From (3.9) and (3.11), it is clear that $E(Y_i^t)$ is not time invariant and the value of $E(Y_i^t)$ depends on the initial random walk probability distribution $w = \{w_1, \ldots, w_n\}$. However, supposing that at a time step t, the expectation is $E(Y_i^t) = \mu_i$ where μ_i satisfies (3.8), we notice from (3.8) and (3.11):

$$E(Y_i^{t+1}) = \sum_{j=1}^{n} E(Y_j^t) p_{ij} = \sum_{j=1}^{n} \mu_j p_{ij} = \frac{\mu_i}{C} = \frac{E(Y_i^t)}{C} \tag{3.12}$$

From (3.12), we can see that, if at a time step t, $E(Y_i^t)$ satisfies (3.8), the relative order of expectation $E(Y_i^t)$ will not change as the time step t increase. Hence, the time invariant measure μ can be used as the ranking score to detect contextual anomalies with respect to S^+ and S^-.

Supposing that $\overline{z}$ is an eigenvector of P that corresponds to an eigenvalue less than 1, we can divide the given data set into two contexts:

$$S^+ = \{i : \overline{z}_i \geq 0\},\ S^- = \{i : \overline{z}_i < 0\} \tag{3.13}$$

It can be shown that $\sum_{i=1}^{n} \overline{z}_i = 0$. Let $\overline{\mu} = \{\overline{\mu}_1, ..., \overline{\mu}_n\}$, where $\overline{\mu}_i = \frac{\overline{z}_i}{\sum_{i=1}^{n} |\overline{z}_i|}$, be the the normalized eigenvector. Assume that the random walk starts with:

$$P(X^0 = i, X^0 \in S^+) = \begin{cases} \overline{\mu}_i, & \text{if } i \in S^+ \\ 0, & \text{if } i \in S^- \end{cases} \tag{3.14}$$

$$P(X^0 = i, X^0 \in S^-) = \begin{cases} 0, & \text{if } i \in S^+ \\ -\overline{\mu}_i, & \text{if } i \in S^- \end{cases} \tag{3.15}$$

In other words,

$$P(X^0 = i) = \begin{cases} \overline{\mu}_i, & \text{if } i \in S^+ \\ -\overline{\mu}_i, & \text{if } i \in S^- \end{cases} \tag{3.16}$$

Given the contexts defined in (3.13), we notice from (3.10) that if the initial probability distribution $P(X^0 = i) = w_i$ satisfies (3.16), then:

$$E(Y_i^1) = \sum_{j \in S^+} w_j p_{ij} - \sum_{j \in S^-} w_j p_{ij} = \sum_{j=1}^{n} \overline{\mu}_j p_{ij} = \widetilde{\lambda} \overline{\mu}_i \tag{3.17}$$

From (3.17), we can see $E(Y_i^1)$ satisfies (3.8). Therefore, the normalized eigenvector $\overline{\mu} = \{\overline{\mu}_1, ..., \overline{\mu}_n\}$ is a valid stationary expectation under the assumption that the random walk starts with probability (3.16). Hence, Wang and Davidson [54] suggest that the smaller the $|\overline{\mu_i}|$ is, the more likely the ith data instance is a contextual anomaly with respect to the two contexts defined in (3.13).

We notice, from the above discussion, the reasonability of using the normalized eigenvector $\overline{\mu} = \{\overline{\mu}_1, ..., \overline{\mu}_n\}$ to generate anomaly ranking depends on the reasonability of using (3.16) as the initial random walk probability. However, Wang and Davidson [54] did not provide any justification or explanation on why the assumption that initial random walk probability satisfies (3.16) is reasonable and important. This is the major drawback of this random walk justification.

Note that if we use the same adjacency matrix W to construct L_{sym} and P, then the following relationship holds: Supposing $\overline{z}$ is an eigenvector of $P = WD^{-1}$ and g is an eigenvector of $L_{sym} = I - D^{-1/2}WD^{-1/2}$ with the eigenvalue $\overline{\lambda}$, we have

$$L_{sym}g = (I - D^{-1/2}WD^{-1/2})g = \overline{\lambda}g$$

This is the same as

$$D^{-1/2}WD^{-1/2}g = (1 - \overline{\lambda})g$$

or

$$D^{-1/2}(WD^{-1})D^{1/2}g = (1 - \overline{\lambda})g$$

Consequently,

$$(WD^{-1})(D^{1/2}g) = (1 - \overline{\lambda})(D^{1/2}g)$$

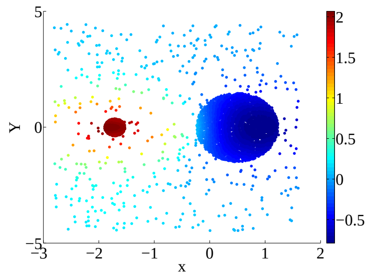
From above we can conclude that g is an eigenvector of $L_{sym} = I - D^{-1/2}WD^{-1/2}$ with eigenvalue $\overline{\lambda}$ if and only if $\overline{z} = (D^{1/2}g)$ is an eigenvector of $P = WD^{-1}$ with eigenvalue $1 - \overline{\lambda}$. We refer to the eigenvector $\overline{z_1} = D^{\frac{1}{2}}g_1$ as the 1st non-principal eigenvector and $\overline{z_2} = D^{\frac{1}{2}}g_2$ as the 2nd non-principal eigenvector of matrix P in latter sections.

Following this random walk perspective from [54], we can see $|(\overline{z_1})_i|$ provides a reasonable basis for measuring the difference in the likelihood of a data instance i being visited by a random walk starting from the cluster based contexts $S^+ = \{i : (\overline{z_1})_i \geq 0\}$ and $S^- = \{i : (\overline{z_1})_i < 0\}$ respectively. The contextual anomalies are those data instances that can be visited from S^+ and S^- almost equally likely. In other words, these contextual anomalies do not belong strongly to either S^+ and S^-.

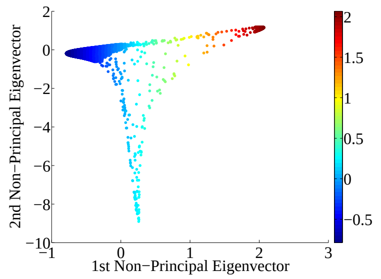
Recall the previous discussion on degree of support perspective in Section 3.1.1, the entry $|(\overline{z_1})_j|$ of the vector $\overline{z_1} = D^{\frac{1}{2}}g_1$, where g_1 is the 1st non-principal eigenvector of L_{sym}, approximates the strength of support from the jth data point on the two classes separation decision. Using the degree of support perspective, we can see that these contextual anomalies, since they do not belong strongly to either S^+ and S^-, they offer less information in defining the two major classes S^+ and S^-. Thus $|(\overline{z_1})_j|$ for the contextual anomalies will be small. In cases where two major clusters exist, the contextual anomalies are actually the point anomalies or anomalies belonging to small clusters since they lie in between the major clusters.

3.2 A Spectral Ranking for Anomaly Detection

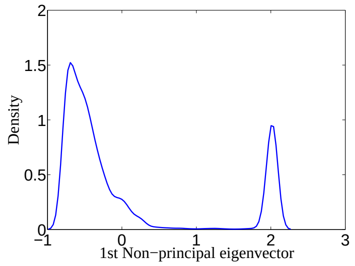
From the degree of support perspective in Section 3.1.1, the vector $\overline{z_1} = D^{\frac{1}{2}} g_1$ approximates the vector of strength of support in the optimal two classes separation. From the random



(a) Synthetic Data



(b) Visualization of $\overline{z_1} = D^{1/2} g_1$ and $\overline{z_2} = D^{1/2} g_2$



(c) Kernel Density Estimation

Figure 3.3: Interpretation of 1st Non-principal Eigenvector For Synthetic Data 3

walk perspective in Section 3.1.2, the vector $\overline{z_1} = D^{\frac{1}{2}} g_1$ can be used to detect the contextual anomalies with regard to the two contexts $S^+ = \{i : (\overline{z_1})_j \geq 0\}$ and $S^- = \{i : (\overline{z_1})_j < 0\}$. Both perspective suggest that, if we define the anomaly score as $f = \max(|\overline{z}|) - |\overline{z}|$, we are able to rank data instances according to their likelihood of being anomalies when two major clusters for normal data instances exist.

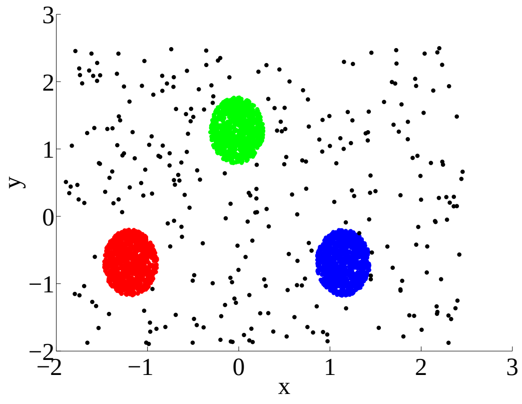
However, there are also cases where there is only one major cluster for the normal data instances together with point anomalies and/or anomalies which form up small clusters. In these scenarios, as illustrated in Figure 3.3, $f = \overline{z}$ or $f = -\overline{z}$ can be used as the anomaly score. The small clusters for anomalies will be ranked as the most abnormal. The point anomalies are ranked as the second most abnormal. The major cluster is ranked as the least abnormal. This ranking is reasonable since the small clusters are the more important targets in these cases.

Based on above analysis, we now formally propose the Spectral Ranking for Anomaly Detection (SRA) algorithm which is summarized in Algorithm 1. SRA automatically decides whether to provide anomaly ranking score f with respect to one majority pattern (cluster) or with respect to the two ($\pm$) major patterns (clusters). The decision is made based on class cardinality as follows: Let $C^+ = \{i : (\overline{z_1})_i \geq 0\}$ and $C^- = \{i : (\overline{z_1})_i < 0\}$ denote the sets of data instances that correspond to non-negative and negative values in $\overline{z}_1$ respectively. We denote the total number of data instances as N. Assume an upper bound on the anomaly ratio is given as R_U. If $\min\{\frac{|C_+|}{N}, \frac{|C_-|}{N}\} \geq R_U$, SRA outputs ranking with respect to multiple patterns. Otherwise, SRA outputs ranking with respect to a single majority class. In other words, if the 1st non-principal eigenvector represents two relatively balanced classes ($|C_+| \approx |C_-|$), SRA outputs ranking with respect to multiple patterns. Otherwise, SRA outputs anomaly ranking score with respect to one major pattern.

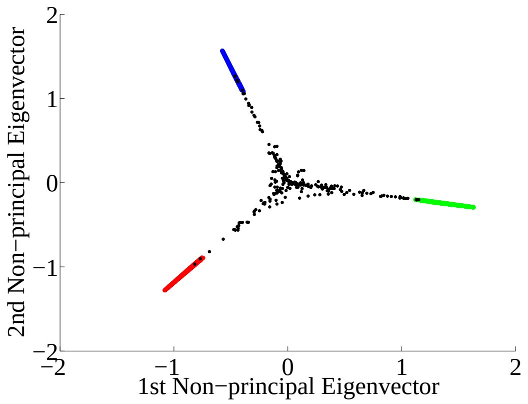
Algorithm 1: Spectral Ranking for Anomaly Detection

```
Input: $W$: An $m$-by-$m$ similarity matrix $W$.
       $R_U$: Upper bound of the ratio of anomaly
Output: $f \in \Re^n$: A score vector where larger value implies being more abnormal
        $Mflag$ : A flag indicating ranking with respect to two patterns or a
        single pattern
begin
    Form Laplacian $L_{sym} = I - D^{-1/2} W D^{-1/2}$ ;
    Compute $\overline{z}_1 = D^{1/2} g_1$ where $g_1$ is the 1st non-principal eigenvector for $L_{sym}$ ;
    Let $C^+ = \{i : (\overline{z_1})_i \geq 0\}$ and $C^- = \{i : (\overline{z_1})_i < 0\}$;
    if $\min\{\frac{|C_+|}{N}, \frac{|C_-|}{N}\} \geq R_U$ then
        $Mflag = 1,\ f = \max(|\overline{z}_1|) - |\overline{z}_1|$ ;
    else if $|C_+| > |C_-|$ then
        $Mflag = 0,\ f = -\overline{z}_1$ ;
    else
        $Mflag = 0,\ f = \overline{z}_1$ ;
    end
end
```

In above discussion, the proposed SRA algorithm mainly focuses on cases where two patterns (clusters) for normal data instances exist and cases where only one pattern (cluster) exists. Note that for cases where more than two patterns (clusters) for normal data instances exist, Algorithm 1 can still be used. This is because that in those cases the 1st non-principal eigenvector from the transition matrix P often separates one main cluster from the other main clusters. Therefore, both the contextual random walk justification and degree of support justification still can be applied. Here we illustrate this fact with one synthetic data set in $\Re^2$, which is shown in Figure 3.4. In Figure 3.4 (a), there are three main clusters, which are marked in red, green and dark blue respectively, together with point anomalies, marked in black. We use a Gaussian kernel with the bandwidth $\sigma = 1$ as the adjacency matrix W. Visualization of the vector $\overline{z_1} = D^{1/2}g_1$ and $\overline{z_2} = D^{1/2}g_2$ is shown in Figure 3.4 (b). For each data instance i, we have marked the corresponding point $((\overline{z_1})_i,(\overline{z_2})_i)$ in Figure 3.4 (b) using the same color as it is in Figure 3.4 (a). As shown in Figure 3.4 (b), $|(\overline{z_1})_i|$ for most of the point anomalies is still much smaller than the data instances that belong to the three major clusters. Therefore a reasonable result can still be achieved using Algorithm 1.



(a) Synthetic Data



(b) Visualization of $\overline{z_1} = D^{1/2}g_1$ and $\overline{z_2} = D^{1/2}g_2$

Figure 3.4: Interpretation of 1st Non-principal Eigenvector For Synthetic Data 4

A second way to cope with more than two main patterns is to utilize additional non-principal eigenvectors. Note that with the random walk perspective, every non-principal eigenvector provides a way to divide the data sets into two contexts, so in cases where more than two patterns (clusters) for normal data instances exist, other non-principal eigenvectors can also be used. For instance, let us suppose that the 2nd non-principal eigenvector of P is also used. The same procedure in Algorithm 1, but with $\overline{z_2} = D^{1/2}g_2$ instead of $\overline{z_1} = D^{1/2}g_1$, can be used to compute another anomaly score denoted as f_2. The final anomaly score for a data instance i can be the summation of $(f_1)_i$ computed from the 1st non-principal eigenvector and $(f_2)_i$ computed from the 2nd non-principal eigenvector, i.e, $f_i = (f_1)_i + (f_2)_i$. More sophisticated way to combine the multiple anomaly score vectors computed from different non-principal eigenvectors can be used [30]. We leave it as a future study.

A third way to cope with many main patterns is to apply the Algorithm 1 iteratively with the 1st non-principal eigenvector. In other words, we further segment both or one of the subgroups $C^+ = \{i : (\overline{z_1})_i \geq 0\}$ and $C^- = \{i : (\overline{z_1})_i < 0\}$. For example, $C^- = \{i : (\overline{z_1})_i < 0\}$ of the synthetic data 4, as shown in Figure 3.4, consists of two main clusters together with point anomalies. We can apply Algorithm 1 only with the data instances in $C^- = \{i : (\overline{z_1})_i < 0\}$ again. This will help us capture more anomalous cases. The challenge here is how to combine scores to generate a single ranking. We also leave this approach as a future study.

One of the difficulties of the clustering algorithm is to decide the number of clusters in a given data set. However, since the process of spectral clustering can also be regarded as a dimension reduction process [4], the non-principal eigenvectors can be viewed as the low dimensional embedding of the high dimensional data set. Therefore, data instances belonging to clusters tend to show up as clusters in the non-principal eigenvectors. Visualization of a couple of non-principal eigenvectors can often be used to get a sense of the number of major clusters in the given data set and decide how many non-principal eigenvectors are needed or which subgroups need further segmentation.

We observe that, with the degree of support perspective, SRA can capture both small clusters for anomalies and point anomalies. Other standard methods usually target one cases instead of both cases. In addition, we note that SRA can also distinguish edges of the main clusters from the cores of the main clusters. Finally, SRA can cope with both cases where there are multiple patterns (clusters) for normal data instances and cases where only one pattern (cluster) exists. Other methods often lack the simultaneous support for both multi-class anomaly detection and one-class anomaly detection.

Although we focus on the unsupervised learning perspective which does not emphasize on generating prediction for new data instances, we observe that, with our SRA algorithm, there is a natural way to generate out of sample predictions. This feature is often missing in other unsupervised anomaly detection methods. However, prediction capability can be useful when new instances arrive after a training model has been computed, particularly if training set is very large, since we do not need to train a new model to incorporate the new data instances. Detailed discussion on how to generate out of sample prediction with the proposed SRA Algorithm is given in Appendix B.

Chapter 4

Experimental Results

In this chapter, we demonstrate effectiveness of the SRA algorithm using both synthetic data sets and real data sets. Particularly, a case study on an automobile insurance fraud detection data set is presented in Section 4.4. Evaluation of the SRA algorithm on synthetic data sets and other real data sets is presented in Section 4.3 and Section 4.5 respectively. These results are evaluated using the Receiver Operating Characteristic (ROC) curves. A brief discussion on the ROC curves is given in Section 4.1. For the purpose of comparisons, we include the performance of two popular unsupervised anomaly detection methods, one-class SVM (OC-SVM) and Local Outlier Factor (LOF), and one supervised learning method, Random Forest (RF). A brief review of the OC-SVM, LOF and RF is given in Section 4.2.

4.1 Receiver Operating Characteristic Curves

Evaluation metrics play an important role in the model selection process. Various evaluation metrics exist for the learning quality of algorithms that produce binary outputs (e.g., 'normal' versus 'abnormal'). The most common evaluation metric is the classification error rate. Let $D = \{(x_1, y_1), \ldots, (x_n, y_n)\}$ be a set of data instances, where x_i is the input data and $y_i \in \{+1, -1\}$ is the output target. Assume that a discriminative function $f(x)$ is used. The classification error rate is defined as:

$$Error = \frac{1}{n} \sum_{i=1}^{n} \delta(f(x_i), y_i) \tag{4.1}$$

where

$$\delta(f(x_i), y_i) = \begin{cases} 1, & f(x_i) \neq y_i \\ 0, & f(x_i) = y_i \end{cases}$$

For anomaly detection, the data set is usually highly unbalanced, i.e., most of the data instances are normal cases. Therefore, the classification error rate is not an appropriate measure since it has a bias toward normal cases. Furthermore, we are more interested in the successful classification of anomalies than normal cases. For example, suppose we have two classifiers for a fraud detection problem where 95% of the data instances are normal cases and 5% of the data instances are fraudulent cases. The first method classifies all the data instances to be normal. The second one classifies 10% of the data instances as abnormal, which include the all the fraudulent cases, and rest of the data instances as normal. The classification error for both classifiers is the the same: 95%, but the second classifier is clearly better for the fraud detection purpose.

Therefore, other measures must be used to evaluate performance of an anomaly detection method. The Receiver Operating Characteristic (ROC) curve is a class skew independent measure [41, 6, 34] that can be used to illustrate the performance of the binary classification as the discriminative threshold varies. Before we discuss the ROC curve, we first introduce several terminologies that are used to describe the ROC curve.

Suppose we have a classifier and we want to evaluate the performance of the classifier on a data set $D = \{(x_1, y_1), \ldots, (x_n, y_n)\}$, following terms can be defined:

- **Positive Data Instances**: The class of data instances that is of interest. In anomaly detection, the positive data instances are anomalies. We denote the number of positive data instances as P.
- **Negative Data Instances**: The data instances that are not positive data instances. In anomaly detection, the negative data instances are normal cases. We denote the number of negative data instances as N.
- **True Positives (TP)**: The positive data instances that are correctly labeled by the classifier. We denote the number of true positives as TP.
- **False Positives (FP)**: The negative data instances that are incorrectly labeled by the classifier. We denote the number of false positives as FP.
- **True Negatives (TN)**: The negative data instances that are correctly labeled by the classifier. We denote the number of true negatives as TN.

Table 4.1: Confusion Matrix

		Predicted Labels		
		Positive	Negative	Total
True Labels	Positive	TP	FN	P
	Negative	FP	TN	N

- **False Negatives (FN)**: The positive data instances that are incorrectly labeled by the classifier. We denote the number of false negatives as FN.

The confusion matrix, as shown in Table 4.1, summarizes the relationship between these terms.

Many anomaly detection algorithms, e.g., our proposed SRA algorithms, LOF, OC-SVM, intrinsically output a numerical ranking score and a predicted label can be obtained by thresholding this ranking score. Each threshold value generates a confusion matrix with different quantities of true positives and false positives. The ROC graph is obtained by plotting the true positive rate ($\frac{TP}{P}$) against the false positive rate ($\frac{FP}{N}$) as the threshold level varies. The true positive rate (also known as sensitivity) is one criterion and the false positive rate (or $1 -$ specificity) is the second criterion.

An increase in the true positive rate, which can be viewed as benefit, often occurs at the cost of an increase in the false positive rate. ROC curves (or true-positive and false-positive frontiers) allow us to visualize the trade-off between true positive rate and false positive rate for different thresholds. Thus it does not depend on a priori knowledge to combine the two objectives into one. The area under the ROC curve (AUC) yields the probability that the generated ranking places a positive class sample above a negative class sample when the positive sample is randomly drawn from the positive class and the negative class sample is drawn randomly from the negative class respectively [19]. Thus, the ROC curve and AUC can be used as criteria to measure how well an anomaly detection ranking algorithm performs. In the subsequent section, we will use AUC and ROC curves to compare different methods. The positive class will be the rarest class, which can be viewed as the anomalies, within the data sets.

4.2 Existing Methods for Comparisons

In this section, we briefly discuss OC-SVM, LOF, and supervised RF and show how to use them for anomaly detection.

One-Class Support Vector Machine (OC-SVM): One-Class Support Vector Machine (OC-SVM) is proposed by Schölkopf et al. [42] for estimating the region where most of the data instances occur in a feature space. The feature space usually is the derived feature space implicitly defined by a specific kernel function.

Let $\theta : \chi \subseteq \Re^d \rightarrow \mathcal{H} \subseteq \Re^{d'}$ be a feature map corresponding to a kernel function $\mathcal{K}(x, \widehat{x})$. Given the unlabelled training data set $X = \{x_1, \ldots, x_n\}$, where $x_i \in \chi \subseteq \Re^d$, the OC-SVM tries to separate $\theta(X) = \{\theta(x_1), \ldots, \theta(x_n)\}$ from the origin in $\mathcal{H} \subseteq \Re^{d'}$ using a hyperplane $w^T\theta(x) = p$ where the distance between the origin and the hyperplane, $\frac{p}{\|w\|_2}$, is maximized and most of the data instances are in the half space $w^T\theta(x) \geq p$. Penalized slack variables, ξ_i, are introduced to allow some data instances to be in the wrong half space, $w^T\theta(x) < p$. The corresponding primal optimization problem is as follows:

$$\begin{aligned} \min_{w,p,\xi} \quad & \frac{1}{2}\|w\|_2^2 + \frac{1}{\nu n}\sum_{i=1}^{n}\xi_i - p \\ \text{subject to} \quad & w^T\theta(x_i) \geq p - \xi_i, i = 1, \ldots, n \\ & \xi_i \geq 0, i = 1, \ldots, n \end{aligned} \tag{4.2}$$

where $0 < \nu \leq 1$ is a parameter that controls the tradeoff between maximizing the distance between origin and the the hyperplane, $w^T\theta(x) = p$, and containing most of the data in the normal region, $w^T\theta(x) \geq p$, created by the hyperplane. We denote this penalty parameter as ν_{svm} in latter discussion. It can be shown [43], under certain condition, solving (4.2) is equivalent to finding a hypersphere with the smallest radius in the derived feature space $\mathcal{H}$ that encloses the training data set.

Given the optimal separating hyperplane $w_*^T\theta(x) = p^*$, the associated anomaly ranking score of a data instance x is then:

$$\widetilde{f}(x) = w_*^T\theta(x) - p^*$$

More detailed discussion of OC-SVM can be found in Appendix C. For results reported in this thesis, we use the implementation of OC-SVM from [13].

Local Outlier Factor: Local Outlier Factor (LOF) was proposed by Breunig et al. [9]. For any given data instance, the LOF score is equal to the ratio of average local density of the k nearest neighbors of a data instance and the local density of the data instance itself.

Formally, let x be a data instance and $Kdistance(x)$ be the distance between data instance x and its kth nearest neighbor. The set of k nearest neighbors of x is denoted as $N_k(x)$. Let x be another data instance and $distance(x, \widehat{x})$ be the distance between x and $\widehat{x}$. The reachability distance between x and $\widehat{x}$ is defined as:

$$Rdistance_k(x, \widehat{x}) = \max\{Kdistance(\widehat{x}), distance(x, \widehat{x})\}$$

The local reachability density is defined as:

$$lrd(x) = \frac{|N_k(x)|}{\sum_{\widehat{x} \in N_k(x)} Rdistance_k(x, \widehat{x})}$$

where $|N_k(x)|$ is cardinality of $N_k(x)$. The LOF score of a data instance x is then defined as:

$$LOF_k(x) = \frac{\sum_{\widehat{x} \in N_k(x)} \frac{lrd(\widehat{x})}{lrd(x)}}{|N_k(x)|} = \frac{\sum_{\widehat{x} \in N_k(x)} lrd(\widehat{x})}{|N_k(x)| lrd(x)} \tag{4.3}$$

In other words, the LOF score of a data instance x is the average local reachability density of the its neighbors divided by its own local reachability density. We denote the neighbor size parameter for LOF as k_{lof} in latter discussion. For results reported for LOF, we use the implementation comes from [48].

Classification And Regression Tree (CART) and Random Forest: Tree based methods use decision trees as predictive model to map input data to output targets [25]. Classification And Regression Tree (CART) proposed by Breiman et al. [8] is one representative tree based method. CART tries to divide the feature space into a set of regions. For a specific region, CART computes a constant as the prediction of all the data instances that reside in the region. In other words, data instances inside the same region has the same prediction.

Formally, let $D = \{(x_1, y_1), \ldots, (x_n, y_n)\}$ be a set of training data instances, where $x_i \in \chi$ is the input data and y_i is the corresponding output target. We denote $(x_i)_j$ as the value of the jth attribute for data instance i. Suppose CART has already divided the feature space χ into a set of regions $R = \{r_1, \ldots, r_m\}$. Then the decision function is:

$$f(x) = \sum_{i=1}^{m} C_i I(x \in r_i) \tag{4.4}$$

where $I(x \in r_m)$ is an indicator function that returns 1 if $x \in r_m$ is true and 0 otherwise. For a regression problem, C_m is the average of the training output targets of all $x_i \in r_m$. For a classification problem, C_m is the majority of the training output targets of all $x_i \in r_m$.

Greedy methods are adopted by CART to divide the feature space into different regions. Taking a regression problem on a real valued training data set, i.e., $y_i \in \Re$, as an example. We start with the full feature space $\Re^d$ and we want to find an attribute j and a split point s to split the space into two regions:

$$r_1(j,s) = \{x_i | (x_i)_j \leq s\} \text{ and } r_2(j,s) = \{x_i | (x_i)_j > s\}$$

so that the sum of squared error is minimized:

$$\min_{j,s} \left(\sum_{x_i \in r_1(j,s)} (y_i - C_1) + \sum_{x_i \in r_2(j,s)} (y_i - C_2) \right)$$

where C_1 is the average of the training output targets of all $x_i \in r_1(j,s)$ and C_2 is the average of the training output targets of all $x_i \in r_2(j,s)$. After we find the best partition strategy for the whole data set, we have two regions. We repeat the same procedure on each of the two regions (r_1,r_2). This process is repeated on all the resulting regions until certain stopping criteria is reached or there is only one data instance in the resulting region. Typical stopping criterion is the maximum depth of the tree. Another common stopping criterion is that we require each resulting region to have at least n_{min} training data instances. For a classification problem, the similar greedy process is adopted where we use weighted sum of some impurity measures to replace the sum of squared errors. Detailed discussion of how to deduce classicification tree is given in Appendix D

Random Forest (RF) is proposed by Breiman [7] to improve the performance of tree based methods. The idea is that, instead of building up a single CART decision tree, we build a series of CART decision trees and combine the predictions from each tree to form the final predictions. It has become a popular supervised machine learning technique due to its predicting power, simplicity and robustness. Formally, let n be the number of training data instances and d be the number of attributes that can be used for prediction. The procedure of random forest can be summarized as below:

1. Suppose that we want to build B trees. Let S be the bootstrap sample of size n_s and m be the number of attributes for building the trees. Then for each tree:
   (a) Create a bootstrap sample S of size n_s from the training data.
   (b) Randomly select m attribute where $m < d$ as predicting attributes.
   (c) Build a CART decision tree using the bootstrap sample S and the randomly selected m attributes

2. Output the ensemble of trees $\{T_1, \dots, T_B\}$.

3. Let the prediction of a data instance x from T_i be $T_i(x)$. For a regression problem, the final prediction from random forest is $f_{RF}(x) = \frac{1}{B}\sum_{i=1}^{B} T_i(x)$. For a Classification problem, the final prediction from random forest $f_{RF}(x)$ is the majority of $\{T_1(x), \dots, T_B(x)\}$.

In the following section, CART is used to deduce rules for clusters and random forest is used to compare the performance of unsupervised learning and supervised learning.

4.3 Synthetic Data Sets

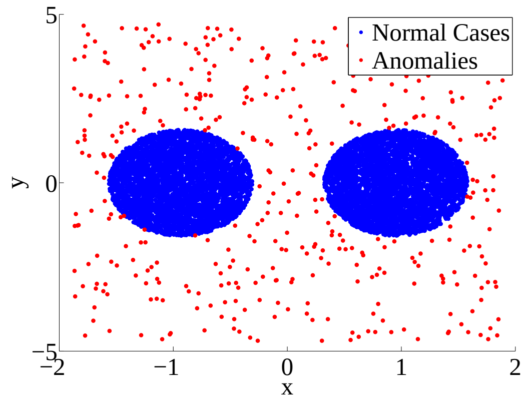
In this section, we demonstrate effectiveness of the proposed SRA algorithm on several synthetic data sets. All the synthetic data sets are real valued. To be consistent, the penalty parameter ν_{svm} for OC-SVM is always set to 0.1 and the neighborhood parameter k_{lof} for LOF is always set as to 200. We use Gaussian kernels as similarity matrices. The same kernel matrix is used by both OC-SVM and the proposed SRA algorithm. The graph

Table 4.2: Summary of the Synthetic Data

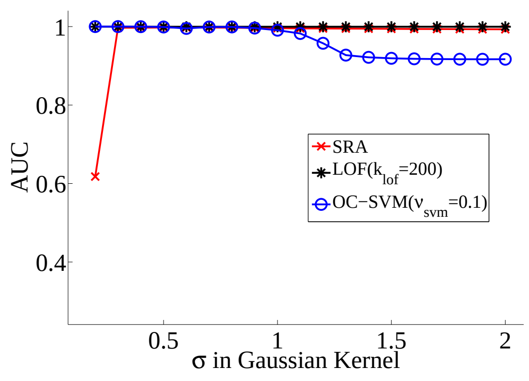
Synthetic Data	No. of clusters for normal cases	No. of normal cases	No. of point anomalies	No. of anomalies belonging to small clusters
1	2	8000	500	0
2	2	8000	2000	1000
3	1	4000	1000	500
4	3	3000	300	0

constructed for each data set is a fully connected graph; the (weighted) adjacency matrix W equals the similarity matrix S. The distance metric used by LOF is the Euclidean distance in the original feature space. In other words, k nearest neighbors are determined based on the Euclidean distance. For SRA, we fix the upper bound of anomaly ratio R_U to 20 %. The four data sets have been used to illustrate the performance of SRA algorithm in Chapter 3. Now we compare performance of SRA with other methods based on these data sets. Table 4.2 presents details regarding these data sets.

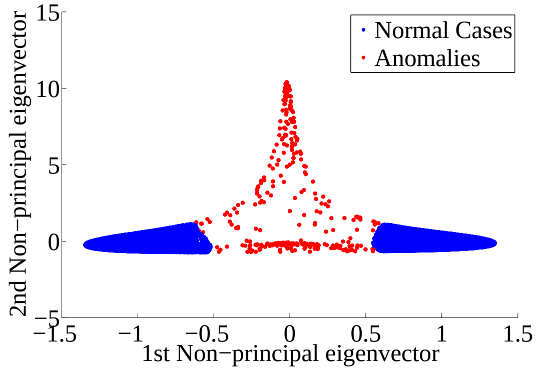
The first synthetic data set, as shown in Figure 4.1 (a), consists of two large and dense clusters, marked in blue, together with 500 point anomalies, marked in red. Each of the clusters consists of 4000 data instances. Visualization of $\overline{z_1} = D^{1/2}g_1$ and $\overline{z_2} = D^{1/2}g_2$ is



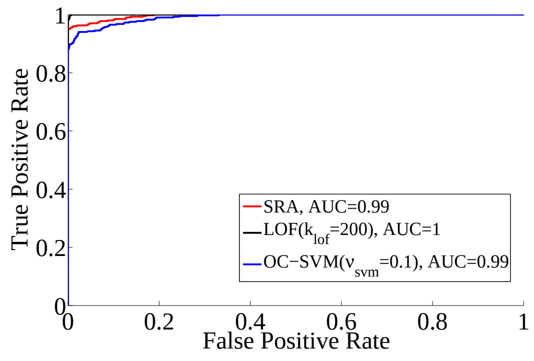
(a) Synthetic Data



(b) Influence of Bandwidth σ in Gaussian Kernel



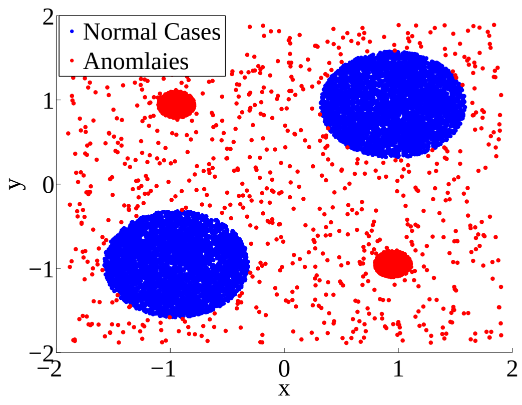
(c) Visualization of $\overline{z_1} = D^{1/2}g_1$ and $\overline{z_2} = D^{1/2}g_2$ ($\sigma = 1$)



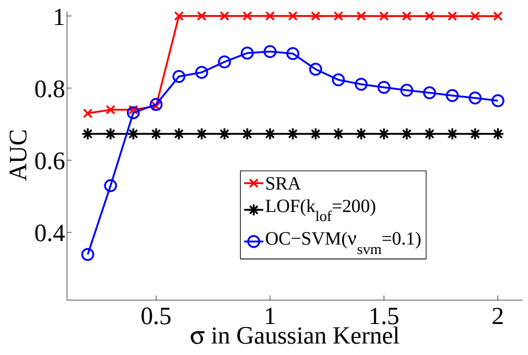
(d) ROC Curves ($\sigma = 1$)

Figure 4.1: Comparing SRA, LOF, OC-SVM on Synthetic Data 1

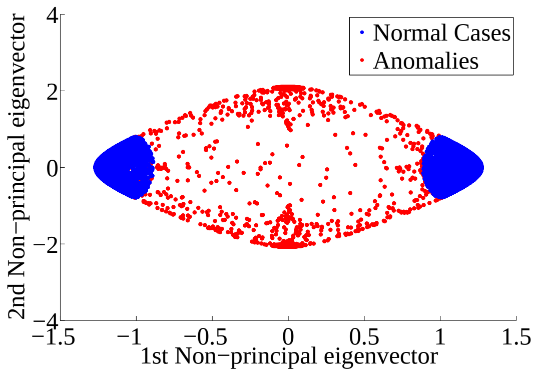
shown in Figure 4.1 (c). Note that, for this data set, the vector $\overline{z_2} = D^{1/2}g_2$ is just used for visualization. No information in the vector $\overline{z_2} = D^{1/2}g_2$ is used. From Figure 4.1 (c), we can see that $|(\overline{z_1})_i|$ for the point anomalies is smaller than the data instances belonging to the two clusters. For this data set, LOF, OC-SVM, and the proposed SRA algorithm all perform reasonably well. ROC curves are shown in Figure 4.1 (d) in which we use the Gaussian kernel with bandwidth $\sigma = 1$ as the kernel and similarity matrix for OC-SVM and SRA. Using LOF, the AUC is close to 1. Using OC-SVM and SRA, the AUC is close to 0.99. The influence of the bandwidth σ in Gaussian kernel on the performance of the three methods is shown in Figure 4.1 (b). Note that we directly use Euclidean distance in original feature space for LOF, so there is no influence of the bandwidth σ on the performance of LOF. For this data set, SRA and OC-SVM both perform stably with respect to the change of bandwidth.



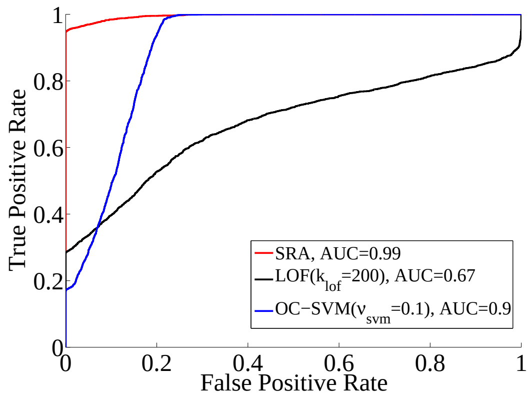
(a) Synthetic Data



(b) Influence of Bandwidth σ in Gaussian Kernel



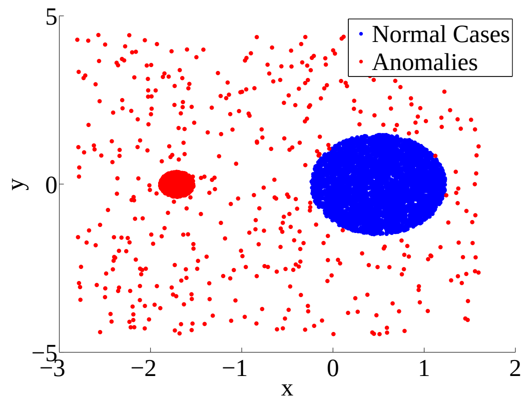
(c) Visualization of $\overline{z_1} = D^{1/2}g_1$ and $\overline{z_2} = D^{1/2}g_2$ $(\sigma = 1)$



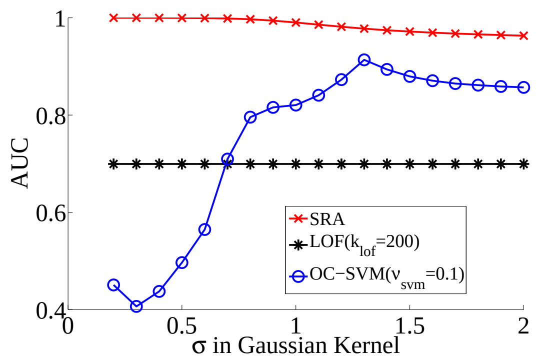
(d) ROC Curves $(\sigma = 1)$

Figure 4.2: Comparing SRA, LOF, OC-SVM on Synthetic Data 2

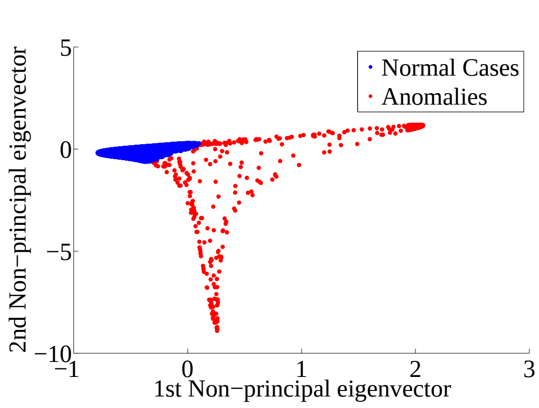
The second synthetic data set, as shown in Figure 4.2 (a), consists of two large and dense clusters, marked in blue, together with 1000 point anomalies and two small clusters of anomalies, marked in red. Each of the major clusters consists of 4000 data instances and each of the small clusters consists of 1000 data instances. Visualization of $\overline{z_1} = D^{1/2}g_1$ and $\overline{z_2} = D^{1/2}g_2$ is shown in Figure 4.2 (c). Again, for this data set, the vector $\overline{z_2} = D^{1/2}g_2$ is just used for visualization. ROC curves are shown in Figure 4.2 (d). Using LOF, AUC is 0.67. Using SRA, AUC is 0.99. Using OC-SVM, AUC is 0.9. The influence of the



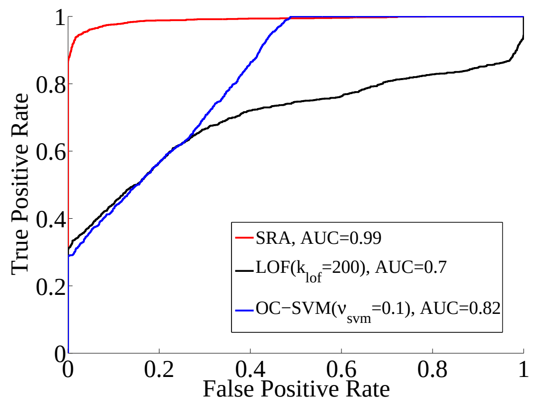
(a) Synthetic Data



(b) Influence of Bandwidth σ in Gaussian Kernel



(c) Visualization of $\overline{z_1} = D^{1/2}g_1$ and $\overline{z_2} = D^{1/2}g_2$ ($\sigma = 1$)

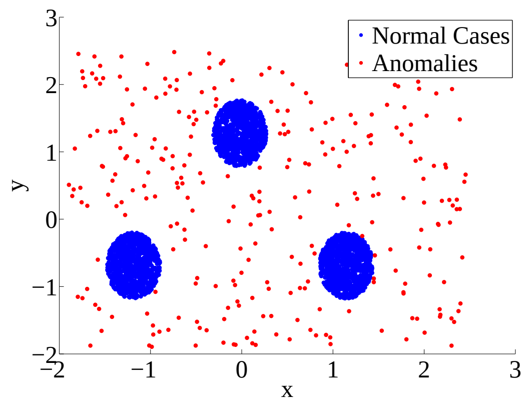


(d) ROC Curves ($\sigma = 1$)

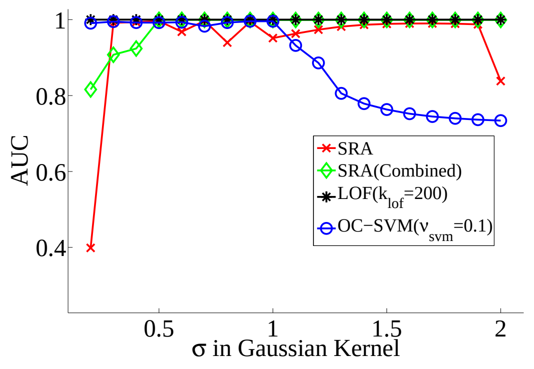
Figure 4.3: Comparing SRA, LOF, OC-SVM on Synthetic Data 3

bandwidth σ in Gaussian kernel on the performance of the three methods is shown in Figure 4.2 (b). For this data set, the performance of SRA is more stable than that of OC-SVM with respect to the change of bandwidth. However, when the bandwidth of Gaussian kernel decreases, the performance of SRA and OC-SVM will decrease. From Figure 4.2 (b) and (d), we can see SRA dominates the other two methods on this synthetic data set.

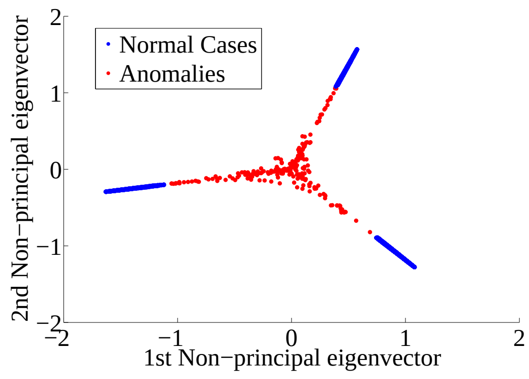
The third synthetic data set, as shown in Figure 4.3 (a), consists of one large and dense clusters, marked in blue, together with 500 point anomalies and one small cluster of anomalies, marked in red. Visualization of $\overline{z_1} = D^{1/2}g_1$ and $\overline{z_2} = D^{1/2}g_2$ is shown in Figure 4.3 (c) and the vector $\overline{z_2} = D^{1/2}g_2$ is just used for visualization. Note that, for this data set,



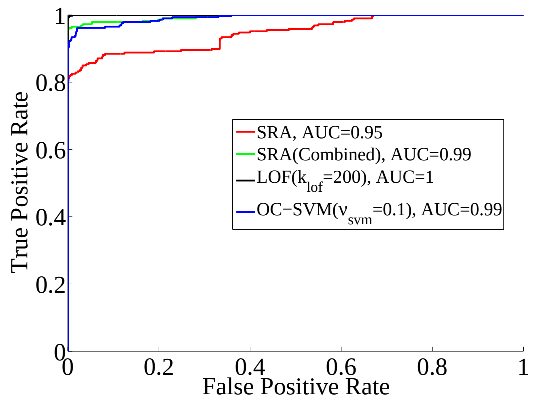
(a) Synthetic Data



(b) Influence of Bandwidth σ in Gaussian Kernel



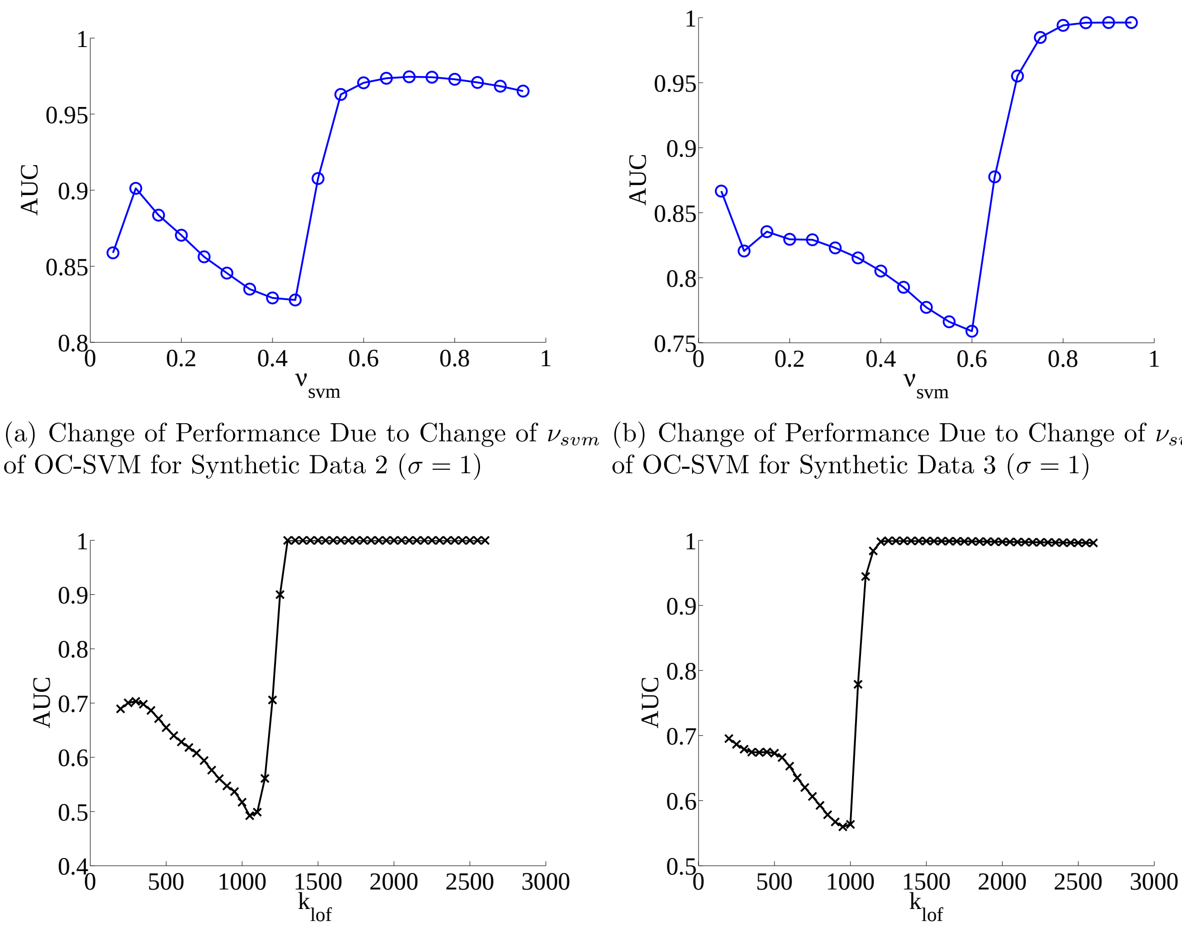
(c) Visualization of $\overline{z_1} = D^{1/2}g_1$ and $\overline{z_2} = D^{1/2}g_2$ ($\sigma = 1$)



(d) ROC Curves ($\sigma = 1$)

Figure 4.4: Comparing SRA, LOF, OC-SVM on Synthetic Data 4

$\min\{\frac{|C_+|}{N}, \frac{|C_-|}{N}\} \leq R_U = 20\%$, where $C^+ = \{i : (\overline{z_1})_i \geq 0\}$ and $C^- = \{i : (\overline{z_1})_i < 0\}$, holds for all the similarity matrices experimented with. Therefore, SRA always output ranking score with regard to one major pattern (Mflag=0 cases in Algorithm 1). Figure 4.3 (d) demonstrates ROC curves of the three methods when Gaussian kernel with bandwidth $\sigma = 1$ is used in OC-SVM and SRA. Using LOF, AUC is 0.7. Using SRA, AUC is 0.99. Using OC-SVM, AUC is 0.82. The influence of the bandwidth σ in Gaussian kernel on the performance of the three methods is shown in Figure 4.3 (b). From Figure 4.3 (b), we can see that the performance of SRA is stabler than that of OC-SVM on this data set. From



(a) Change of Performance Due to Change of ν_{svm} of OC-SVM for Synthetic Data 2 ($\sigma = 1$)

(b) Change of Performance Due to Change of ν_{svm} of OC-SVM for Synthetic Data 3 ($\sigma = 1$)

(c) Change of Performance Due to Change of k_{lof} of LOF for Synthetic Data 2

(d) Change of Performance Due to Change of k_{lof} of LOF for Synthetic Data 3

Figure 4.5: Change of Performance Due to Change of Parameter

Figure 4.3 (b) and (d), we can see that SRA also dominates the other two methods on this synthetic data set

The fourth synthetic data set, as shown in Figure 4.4 (a), consists of three large and dense clusters, marked in blue, together with 300 point anomalies. From Figure 4.4 (d), in which Gaussian kernel with bandwidth $\sigma = 1$ is used for OC-SVM and SRA, we can see that, for this data set, LOF, OC-SVM, and our SRA algorithm all perform reasonably well. The influence of the bandwidth σ in Gaussian kernel on the performance of the three methods is shown in Figure 4.4 (b). Note that for the previous three data sets, only the 1st non-principal eigenvector $\overline{z_1} = D^{1/2}g_1$ is used to generate the SRA anomaly score. However, in this example, we also include the anomaly score computed from the 2nd non-principal eigenvector $\overline{z_2} = D^{1/2}g_2$ and combine it with anomaly score computed the from the 1st non-principal eigenvector $\overline{z_1} = D^{1/2}g_1$ using a simple summation. This corresponds to SRA (Combined), marked in green, in Figure 4.4 (b) and (d). For this data set, we can see that combining the SRA anomaly scores computed from 1st and 2nd non-principal eigenvector together is better than using only SRA anomaly score computed from the 1st non-principal eigenvector.

In the experimental results in this section, we have fixed the penalty parameter ν_{svm} for OC-SVM to be 0.1 and the neighborhood parameter k_{lof} for LOF to be 200. We point out that by tuning the parameter, higher AUC can be achieved with OC-SVM and LOF on the second and third synthetic data sets. The influence of the parameter ν_{svm} on the performance of OC-SVM and k_{lof} on the performance of LOF is illustrated in Figure 4.5. The bandwidth of the Gaussian kernel for OC-SVM is fixed to be $\sigma = 1$. Note that, in practice, it is often hard to tune the parameters for LOF and OC-SVM since we usually do not have labels to evaluate the results. Given a similarity matrix, SRA either output anomaly score vector with regard to a single pattern or with regard to multiple patterns, the choice of which is controlled by parameter R_U. As we can see from previous examples, SRA only requires a rough estimation of the upper bound of the anomaly ratio R_U in order to be effective. Therefore, compared with LOF and OC-SVM, SRA, to some extent, is less susceptible to the problems associated with parameter tuning.

In this section, we have demonstrated with some synthetic examples that SRA can be one effective alternative to the state-of-the-art anomaly detection method. Furthermore, we show that SRA, to some extent, alleviates the problem of parameter tuning. In addition, SRA can sometimes outperform LOF and OC-SVM.

4.4 Case Study: Insurance Fraud Detection

Fighting against insurance fraud is a challenging problem. According to [51, 20], 21% $\sim$ 36% auto-insurance claims contain elements of suspected fraud but only less than 3% of suspected fraud is prosecuted. Traditionally insurance fraud detection relies heavily on auditing and expert inspection. Since manually detecting fraudulant cases is costly and inefficient, investigative resource can be severely constrained. Data mining and machine learning techniques have recently gained the interest from both the insurers and policy holders. In addition, due to the need to detect fraud prior to the claim payment, data mining analytics is increasingly recognized as a key in fighting against fraud. There is a great demand for efficient predicative methods which maximize the true positive detection rate, minimize the false positive rate, and are able to quickly identify new and emerging fraud schemes.

The existing literature on auto insurance fraud detection typically formulates the problem as a supervised learning task, see, e.g., [39, 40] and references therein. The literature on auto insurance *unsupervised* fraud detection is extremely sparse. To the best of our knowledge, there are clustering analysis [11] (based on self-organizing feature map) and PRIDIT analysis [10, 2, 1] (based on RIDIT Score and Principal Component Analysis). However, studies in [11, 10, 2, 1] are all conducted using a single Personal Injury Protection (PIP) data set, which is provided by Automobile Insurance Bureau (AIB) from Massachusetts [10]. This data set has been preprocessed by auditors and fraud detection inspectors so that all the features are the red flags selected by domain experts. Furthermore, this data set consists of predictive variables which have been constructed such that the members from the fraudulent class tend to score lower than the members from the non-fraudulent class. Consequently methods developed for data sets under these assumptions are not truly unsupervised learning methods and are susceptible to pitfalls which come from potential untrustworthiness in human assessments. Furthermore, insurance claim data often consists of numerical, ordinal, categorical, and text data. Consequently, it will be hard to apply methods proposed in [11, 10, 2, 1] without preprocessing from experts since they can only deal with numerical features.

To illustrate the effectiveness of the proposed SRA algorithm on insurance fraud detection, we apply it for an auto insurance claim data set used in [39]. This is the only publicly available auto insurance fraud data set that we can find from the academic literature. This data set consists of 15420 claim instances from January 1994 to December 1996. It has a 6% fraudulent cases and 94% legitimate cases, with an average of 430 claims per month. In addition, the data set consists of 31 attributes, all of which can be considered as categorical or ordinal, including base policy, fault, vehicle category, vehicle price (6 nominal values), month of the accident, manufacturer of the car, accidental area, gender of policy holder, and others. Intuitively, anomaly in this case should be assessed with respect to nominal value combinations. Consequently the overlapping similarity and Hamming distance based kernels are reasonable similarities to use.

In [39], this data set is used to assess achievable prediction quality from the supervised learning. Since labels for auto insurance claims are generally not available at the detection time, we now apply our proposed *unsupervised* SRA to the claim data set. In other words, the labels are used here only for performance evaluation.

When applying standard unsupervised fraud detection methods on this data set, we encounter two major challenges which have been briefly mentioned in the previous sections. Firstly, most of the attributes in this dataset are categorical or ordinal. Secondly, unlike common anomaly detection problems, the claim data forms multiple patterns. Consequently one-class anomaly detection method is less likely to be effective.

We train a supervised Random Forest (RF) on the full dataset. The training accuracy of RF is used as an upper bound for evaluations of unsupervised learning methods. For all computational results reported subsequently, the number of trees built is 500 and the number of attributes used for building each tree is 6. The number of trees that classify the data instance as fraud is used as the ranking score. What we have observed is that the $|C_{+}| \approx |C_{-}|$ for all the similarities we have experimented with, thus we always output the SRA ranking score with regard to multiple patterns (mFLAG=1). Computational results from SRA are all based on the 1st non-principal eigenvector only. We use the distance in the derived feature space, implicitly defined by a kernel function, as the distance metric for LOF. Formally, let $\mathcal{K}(x, y)$ be a kernel function that uniquely define a feature map $\theta : \chi \subseteq \Re^d \to \mathcal{H} \subseteq \Re^{d'}$. The Euclidean distance between two data instance x and $\widehat{x}$ in the derived feature space $\mathcal{H}$ is then:

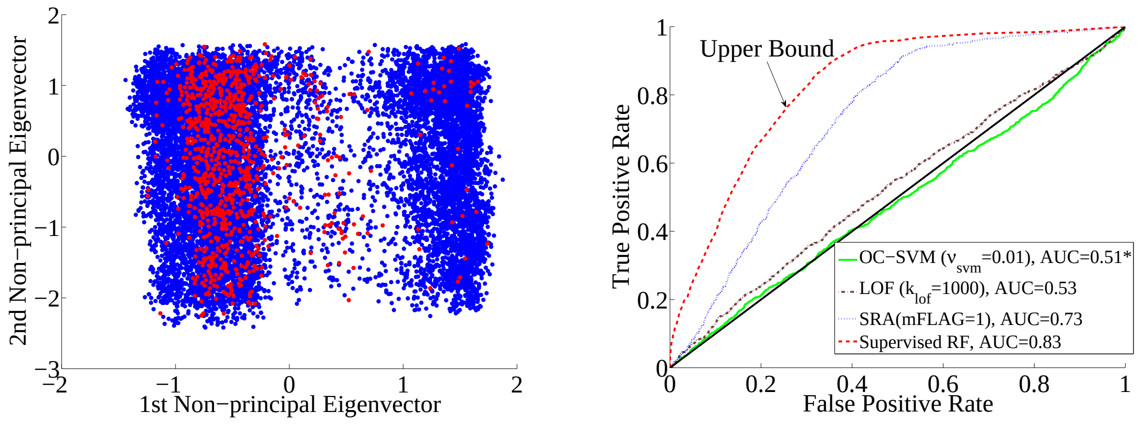
$$\begin{aligned} d(x, y) &= \|\theta(x) - \theta(\widehat{x})\|_2 \\ &= \sqrt{\theta(x)^T\theta(x) + \theta(\widehat{x})^T\theta(\widehat{x}) - 2 \cdot \theta(x)^T\theta(\widehat{x})} \\ &= \sqrt{\mathcal{K}(x, x) + \mathcal{K}(\widehat{x}, \widehat{x}) - 2 \cdot \mathcal{K}(x, \widehat{x})} \end{aligned} \tag{4.5}$$

Table 4.3 summarizes AUCs from different methods on different similarities. We note that some of the results here have been included in our recent paper [37]. In the following sections, the fraudulent cases are marked in red while the legal cases are marked in blue.

Table 4.3: Summary of Results for the Fraud Detection Data Set

Automobile Fraud Detection									
Method			OS	AGK				HDK	
				β				τ	
				10	100	1000	3000	0.5	0.8
LOF	k_{lof}	10	0.53	0.5	0.52	0.58	0.64	0.52	0.53
		100	0.51	0.51*	0.54	0.58	0.67	0.51	0.52
		500	0.53	0.52*	0.55	0.59	0.68	0.51	0.51
		1000	0.53	0.52*	0.53	0.59	0.69	0.5	0.5
		3000	0.5	0.58*	0.55	0.58	0.69	0.54*	0.55*
OC-SVM	ν_{svm}	0.01	0.51*	0.53*	0.51*	0.54	0.59	0.51*	0.52*
		0.05	0.51*	0.53*	0.51*	0.55	0.59	0.52*	0.53*
		0.1	0.51*	0.54*	0.51*	0.55	0.59	0.53*	0.54*
SRA	mFLAG	1	**0.73**	**0.74**	**0.74**	**0.66**	**0.74**	**0.74**	**0.74**

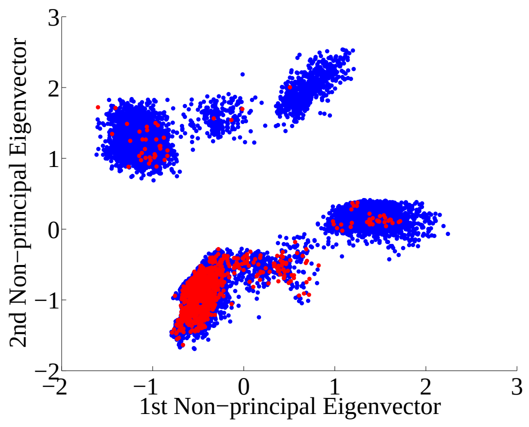
For entries marked by *, AUC reported is one minus the actual AUC (which is below 0.5).
OS:Overlapping Similarity, AGK: Adaptive Gaussian Kernel
HDK:Hamming Distance Kernel



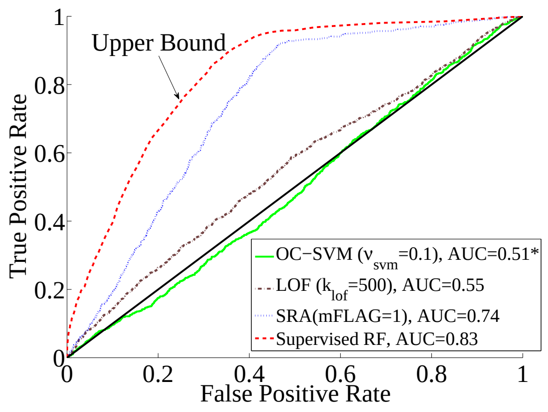
(a) Visualization of $\overline{z_1} = D^{1/2}g_1$ and $\overline{z_2} = D^{1/2}g_2$

(b) ROC Curves

Figure 4.6: Comparisons Based on the Overlapping Similarity



(a) Visualization of $z_1 = D^{1/2}g^{(1)}$ and $z_2 = D^{1/2}g^{(2)}$



(b) ROC Curves

Figure 4.7: Comparisons Based on an Adaptive Gaussian Kernel with $\beta = 100$

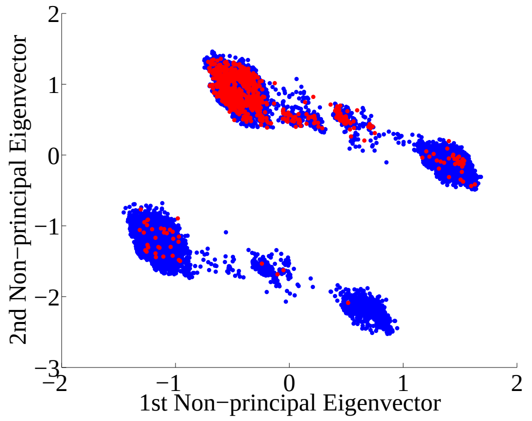
4.4.1 Comparisons with LOF, OC-SVM and (supervised) RF

We first compare the performance of SRA with that of LOF and OC-SVM using overlapping similarity, as well as (supervised) RF. We note that, in contrast to SRA, which only requires a rough estimation of R_U, LOF requires tuning of the parameter k_{lof} and OC-SVM requires tuning of parameter μ_{svm}. This can present more challenges for unsupervised learning since there is no mechanism to decide how to choose their values.

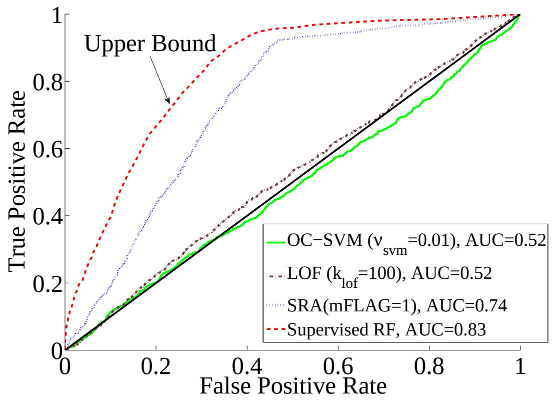
Figure 4.6 (b) presents ROC curves for RF (supervised), SRA, LOF and OC-SVM, where latter three used overlapping similarity. For LOF and OC-SVM, they correspond to the optimal AUC among different parameters we have experimented with.

Figure 4.7 (b) shows ROC curves for SRA, LOF, OC-SVM achieved with the adaptive Gaussian Kernel with the weighted Hamming distance and neighborhood size $\beta = 100$. We observe that AUC for each of SRA, LOF, and OC-SVM is improved. We conjecture that the improvement comes from the fact that the weighted Hamming distance is a better distance measures than Hamming distance, since information on the number of distinct value in each feature is also included in the similarity measure. From Figure 4.7 (a), we also observe that, using the adaptive Gaussian kernel, clusters are also more distinct comparing to the overlapping similarity.

Hamming distance kernel is defined based on the overlapping similarity measure. Figure 4.8 compares performance of SRA, LOF and OC-SVM using a Hamming distance kernel.



(a) Visualization of $\overline{z_1} = D^{1/2}g_1$ and $\overline{z_2} = D^{1/2}g_2$



(b) ROC Curves

Figure 4.8: Comparisons Based on a Hamming Distance Kernel with $\tau = 0.8$

We observe a slight change in cluster structures, but SRA achieves a similar 0.74 AUC.

4.4.2 Understanding and Validating Detected Abnormal Patterns

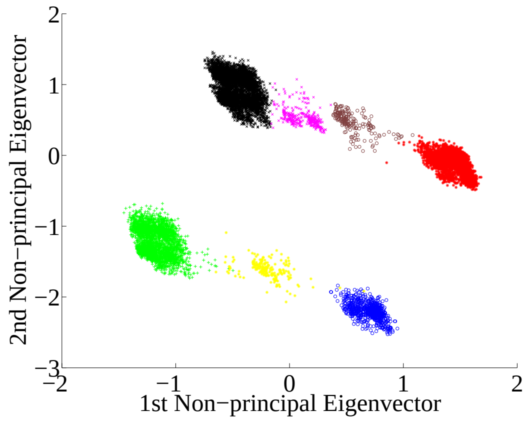


Figure 4.9: Clusters With Color

Cluster	# I	# L	# F	r_f
1 (blue)	723	722	1	0.0014
2 (green)	3261	3228	33	0.0101
3 (red)	4266	4231	35	0.0082
4 (black)	6423	5622	761	0.1185
5 (yellow)	206	203	3	0.0146
6 (purple)	340	294	46	0.1353
7 (brown)	201	157	44	0.2189

# I: No. of Instance, # L: No. of Legal
# F: No. of Fraud, r_f: Fraud Ratio

Table 4.4: Summary Information for Clusters

In order to understand whether the ranking of suspiciousness for this auto fraud detection problem is reasonable, we investigate the significant attributes under which highly

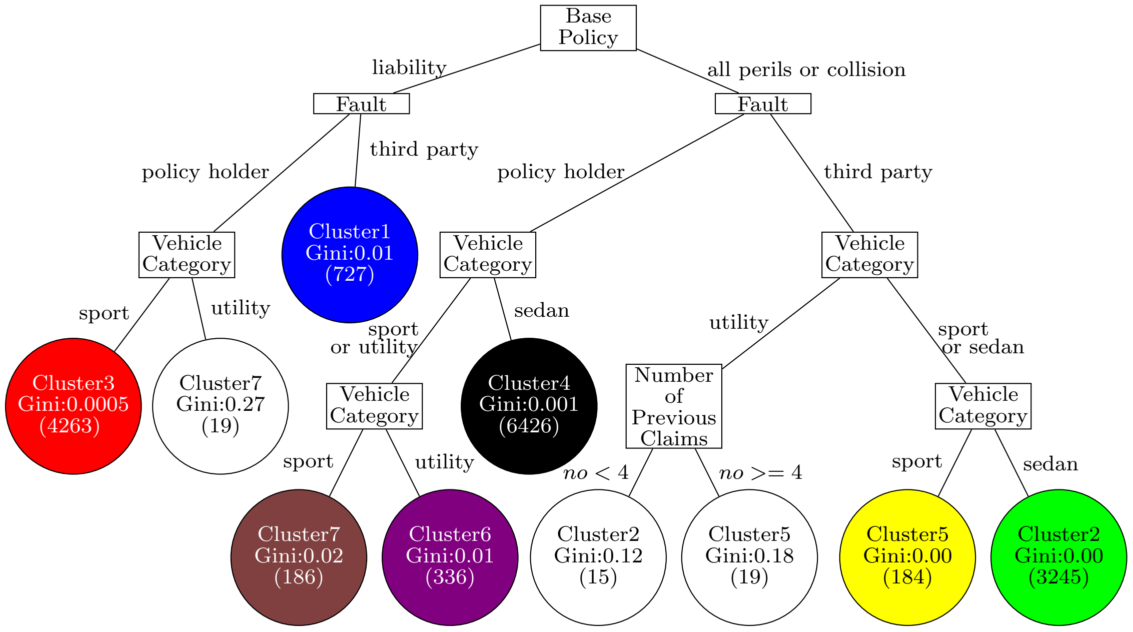


Figure 4.10: Decision Tree

ranked claims are different from the majority. If deviation from the majority is based on attributes which can lead to suspicion, further investigation of these cases are deemed necessary. As an example here, we investigate clusters identified using the Hamming distance kernel with $\tau = 0.8$. Figure 4.9 is identical to Figure 4.8 (a), except that different clusters formed by the 1st and 2nd non-principal eigenvectors are displayed with different colors. The fraud ratio of each cluster is shown in Table 4.4. It can be seen from the plot and the table that most of the fraudulent cases (92%) reside in the region colored in black, brown and purple. They are also regions that have relatively high anomaly score from SRA. Such observation leads us to investigate what each cluster represents, especially the ones with high fraud ratio.

For this purpose, we build a standard CART (classification and regression tree) to deduct rules for each cluster. The training labels for CART are the cluster labels that we have identified from the 1st and 2nd non-principal eigenvectors and training data is the whole data set. We can discover several rules from Figure 4.10 for the clusters, colored in black, brown and purple, which have relatively high rank from SRA:

1. If the insurance policy is for collision or all perils and it is the policy holder who causes the accident (policy holder at fault), the corresponding claim belongs to the

cluster with high fraud ratio (colored in black, brown and purple).

2. If the insurance policy is for liability and/or it is **not** the policy holder who causes the accident (third party at fault), the corresponding claim belongs to the other clusters (colored in blue, red, yellow and green).

3. Following Rule 1, if the policy holder drives a sport car, the corresponding claim belongs to cluster 7 (colored in brown). If the policy holder drive a utility car, the corresponding claim belongs to cluster 6 (colored in purple). Otherwise, the corresponding claim belongs to cluster 4 (colored in black).

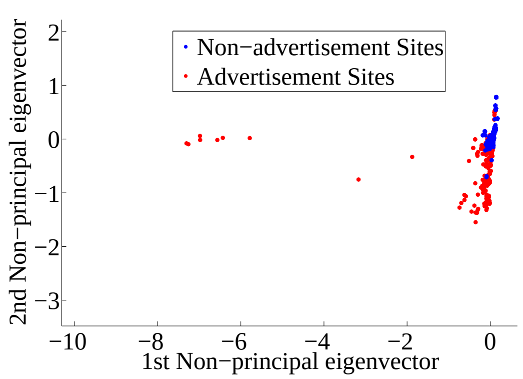
Intuitively, even without the ranking function, the rules deducted for cluster 4, 6 and 7 indeed seem to be suspicious to alert auditors to carry on further investigation. What is more, from training supervised random forest in the previous section, we have learnt that the most important 3 attributes in classify fraudulent cases against legal cases are **base policy**, **car types** , and **fault**. As you can see, the three attributes are actually the attributes used in defining the clusters identified by SRA. Thus, we conclude that ranking from SRA is actually meaningful and reasonable for this auto fraud insurance detection data set.

4.5 Additional Real Data Sets

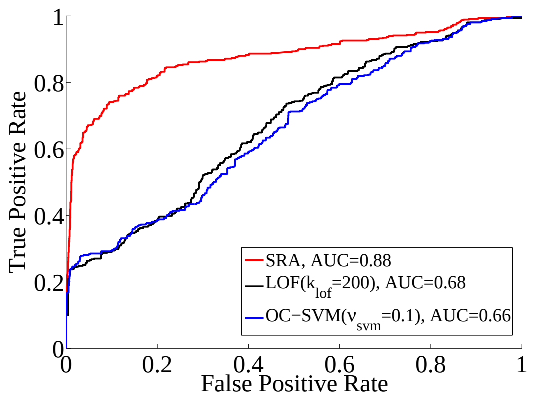
In this section, we demonstrate effectiveness of the proposed SRA algorithm on several other real data sets from the UCI machine learning repository [3]. Again, the penalty parameter ν_{svm} for OC-SVM is set to 0.1 and the neighborhood parameter k_{lof} for LOF is set to 200. The upper bound of anomaly ratio R_U, for SRA algorithm, is set to 30%. The graph constructed for SRA is a fully connected graph. For real valued data sets, Gaussian kernel is used for similarity (kernel) matrices for OC-SVM and SRA. Euclidean distance in the original feature space is used as the distance metric for LOF. For data sets with categorical attributes, Hamming distance kernel discussed in Section 2.3 is used. The Euclidean distance in the derived feature space implicitly defined by the kernel function is used as the distance metric for LOF.

The first data set is the internet advertisement data set which consists of 3279 data instances. A total of 2821 data instances correspond to non-advertisement web sites and 458 data instances correspond to advertisement web sites. For each data instance, there are 1558 real valued attributes. We treat the 458 data instances that correspond to advertisement web sites as anomalies. Note that for this high dimensional data sets, to use

Gaussian kernel as the similarity and kernel matrices for OC-SVM and SRA, we need to set the bandwidth σ to be approximately proportional to the number of attributes d. The visualization of the eigenvectors can be found in Figure 4.11 (a) where we use the Gaussian kernel with bandwidth $\sigma = 60$. For this data set, the vector $\overline{z_2} = D^{1/2}g_2$ is just used for



(a) Visualization of $\overline{z_1} = D^{1/2}g_1$ and $\overline{z_2} = D^{1/2}g_2$ ($\sigma = 60$)

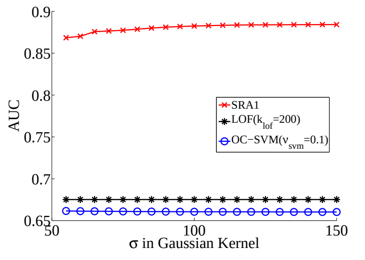


(b) ROC Curves ($\sigma = 60$)

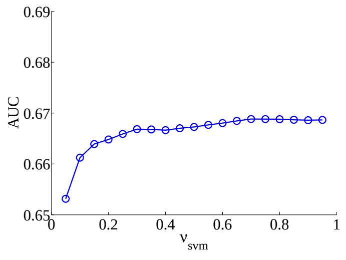
Figure 4.11: Comparing SRA, LOF, OC-SVM on Advertisement Data Set

visualization. For this data set, SRA dominates OC-SVM and LOF. A demonstration of ROC curves is shown in Figure 4.11 (b). SRA achieves AUC=0.88 while LOF achieves AUC=0.68 and OC-SVM achieves AUC=0.66. The changes of performance due to changes of parameters in Gaussian kernel, LOF ,OC-SVM are shown in 4.12. From 4.12 (b) and (c), we observe that, for this data set, tuning the parameters for LOF and OC-SVM will not significantly improve the results.

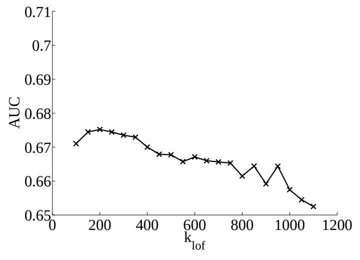
The second data set is the satellite image data set which consists of 6435 data instances. For each data instance, there are 36 real valued attributes. There are 7 classes within the data set and each of them corresponds to a specific type of landscapes. We treat the data instances belonging to Class 4 as anomalies since the number of data instances belonging to Class 4 is the smallest. The rest of the data instances are treated as normal cases. In this example, the anomaly score computed the from the 2nd non-principal eigenvector is also included. We combine it, using a simple summation, with the anomaly score from the 1st non-principal eigenvector. This corresponds to SRA (Combined), marked in green, in Figure 4.13 (b) and 4.14 (a). Visualization of eigenvectors and ROC curves are shown in Figure 4.13, where a Gaussian kernel with bandwidth $\sigma = 10$ is used for OC-SVM



(a) Influence of Bandwidth σ in Gaussian Kernel

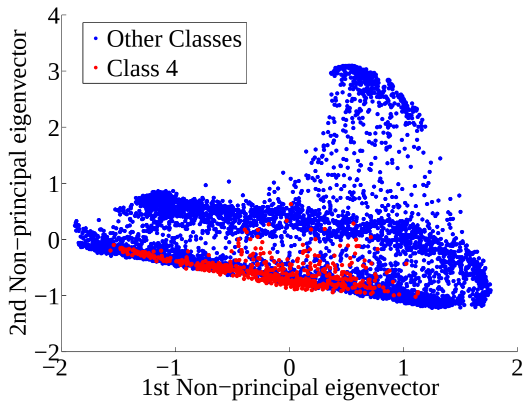


(b) Influence of ν_{svm} for OC-SVM ($\sigma = 60$)

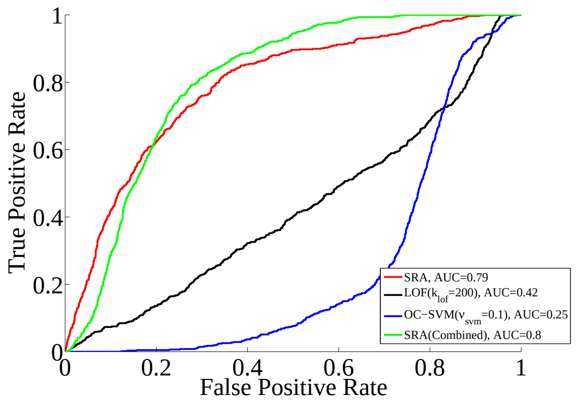


(c) Influence of k_{lof} for LOF

Figure 4.12: Change of Performance Due to Change of Parameter (Advertisement Data)

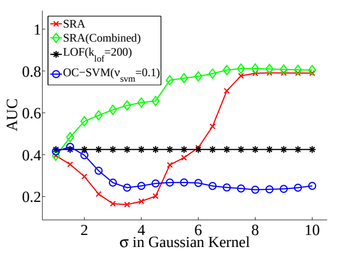


(a) Visualization of $\overline{z_1} = D^{1/2}g_1$ and $\overline{z_2} = D^{1/2}g_2$ ($\sigma = 10$)

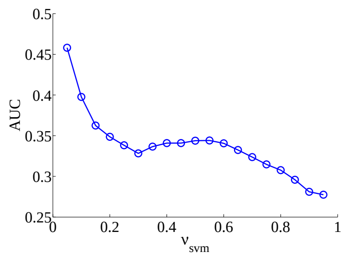


(b) ROC Curves ($\sigma = 10$)

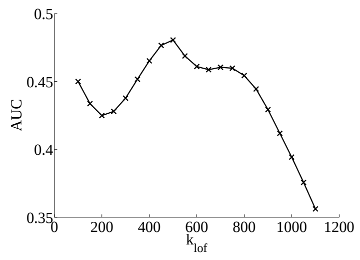
Figure 4.13: Comparing SRA, LOF, OC-SVM on Satellite Data Set



(a) Influence of Bandwidth σ in Gaussian Kernel



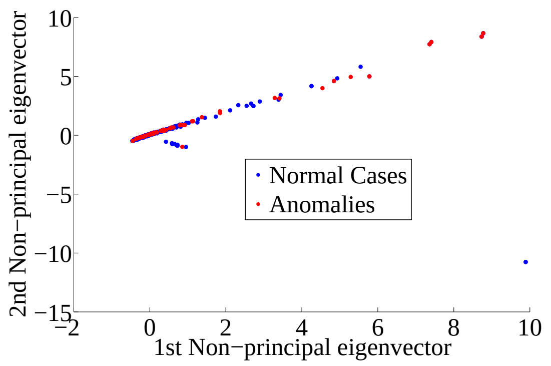
(b) Influence of ν_{svm} for OC-SVM ($\sigma = 10$)



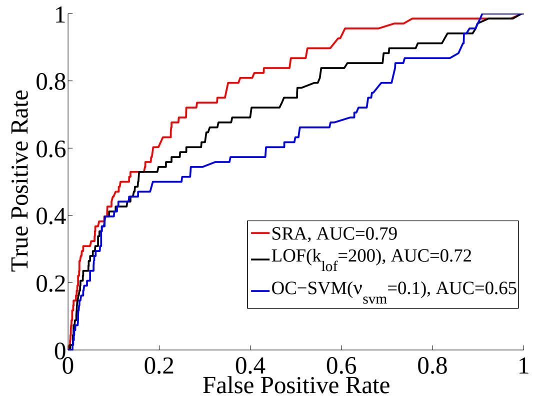
(c) Influence of k_{lof} for LOF

Figure 4.14: Change of Performance Due to Change of Parameter (Satellite Data Set)

and SRA. We note that SRA still dominates LOF and OC-SVM on this data set and tuning the parameter for LOF and OC-SVM does not significantly improve the results. We also notice that for OC-SVM, AUC is actually below 0.3. If we reverse the ranking, we achieve AUC=0.75. However, reversing the ranking contradicts the assumption of OC-SVM. In addition, from Figure 4.13 (a), we can see that SRA also performs poorly when the bandwidth for the Gaussian kernel is small.



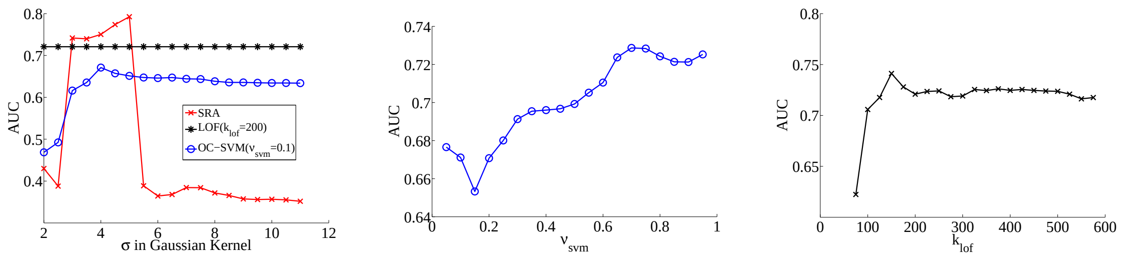
(a) Visualization of $\overline{z_1} = D^{1/2}g_1$ and $\overline{z_2} = D^{1/2}g_2$ ($\sigma = 5$)



(b) ROC Curves ($\sigma = 5$)

Figure 4.15: Comparing SRA, LOF, OC-SVM on Solar Data Set

The third data set is the solar flare data set which consists of 1389 data instances. Each data instance corresponds to a specific day and the output target is the number of times



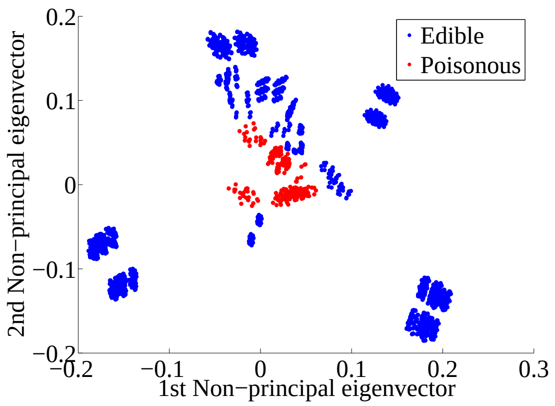
(a) Influence of Bandwidth σ in Gaussian Kernel (b) Influence of ν_{svm} for OC-SVM ($\sigma = 5$) (c) Influence of k_{lof} for LOF

Figure 4.16: Change of Performance Due to Change of Parameter (Solar Data Set)

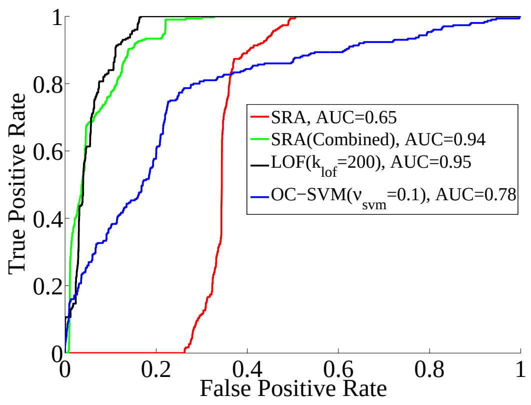
the solar flares occur within that day. There are 32 real valued attributes for this data set. Among the 1389 days recorded, solar flares only occur in 68 days. We treat the days when solar flares occur as anomalies. For this data set, the best AUC for SRA is 0.8 , the best AUC for LOF is 0.75 and the best AUC for OC-SVM is 0.72. When the bandwidth of the Gaussian kernel is within the range of [3,5], SRA dominates the LOF and OC-SVM as shown in Figure 4.15 (b) and Figure 4.16 (a). However, we also observe that SRA only performs well when the bandwidth of the Gaussian kernel is within the range of [3,5].

From the previous two examples, we notice the choice of similarity matrix can play an important role in successfully applying SRA for anomaly detection. How to find the best similarity matrices, which capture the important structure within the data sets, is challenging and worthy of future investigation.

Lastly we consider the mushroom data which consists of 8124 instances originally. There are two classes in this data set: edible and poisonous classes, with 4208 edible data instances and 3916 poisonous data instances. We create an anomaly detection problem by including all 4208 edible data instances as normal cases and randomly selecting 300 poisonous data instances as anomalies. All attributes in this data set are categorical. We use Hamming distance kernel ($\tau = 0.8$) for OC-SVM and SRA. Visualization of eigenvectors is shown in Figure 4.17 (a). Using the 1st non-principal eigenvector only, the SRA ranking score only yields AUC=0.65. However, from visualization of the eigenvectors, we can see clearly that more than two major patterns exist. Therefore, if we also include the anomaly score computed the from the 2nd non-principal eigenvector and combine it with anomaly score computed the from the 1st non-principal eigenvector using a simple summation, AUC=0.94 is achieved. LOF can also perform reasonably well with AUC close to 0.95 on this data sets as long as the k_{lof} is large than 150. OC-SVM can achieve close to 0.9 AUC if we tune

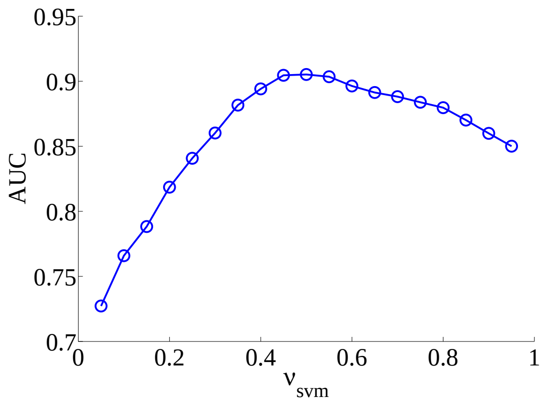


(a) Visualization of $\overline{z_1} = D^{1/2}g_1$ and $\overline{z_2} = D^{1/2}g_2$

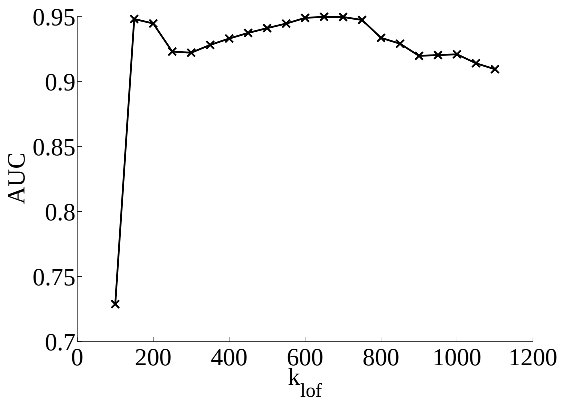


(b) ROC Curves

Figure 4.17: Comparing SRA, LOF, OC-SVM on Mushroom Data Set



(a) Influence of ν_{svm} for OC-SVM)



(b) Influence of k_{lof} for LOF

Figure 4.18: Change of Performance Due to Change of Parameter (Mushroom Data Set)

the penalty parameter ν_{svm}. The influence of k_{lof} on LOF and ν_{svm} on OC-SVM is shown in Figure 4.18.

In this section, with four real data sets, we further demonstrate the effectiveness of the proposed SRA for anomaly detection problems. We can see that SRA is an effective alternative to LOF and OC-SVM. In addition, in certain cases, SRA outperforms OC-SVM and LOF.

Chapter 5

Conclusion

In this thesis, we propose a spectral ranking method for anomaly detection (SRA). We observe that the spectral optimization can be interpreted as a relaxation of the unsupervised SVM; consequently the absolute value of a non-principal eigenvector is a measure of strength of support in a two classes separation. Alternative random walk perspective provides a different motivation for the proposed SRA algorithm. Based on this information in the eigenvector, in the cases where more than one major patterns for normal cases exist, a data instance is more likely to be an anomaly if the absolute value of its corresponding element in the non-principal eigenvector is smaller. Furthermore, we allow a choice of the reference in the assessment of anomaly ranking. If the minority cluster class does not have a sufficient mass, one can choose to assess anomaly likelihood with respect to a single majority class and ranking is generated suitably with this view. Otherwise, anomaly is assessed with two main patterns. Other anomaly detection methods often lack the simultaneous detection for both one-class anomaly and multi-class anomaly. Furthermore, with a meaningful similarity measure, SRA method is able to deal with data sets with different types of attributes. This feature is also often missing for many other anomaly detection methods.

As an illustration of the proposed SRA, we consider the challenging auto fraud insurance detection problem based on a real claim data set. Since obtaining labels is time consuming, costly, and error prone in real applications, we model the problem as unsupervised learning. For the given claim dataset, SRA generates ranking without labels. The given labels are used in performance evaluation only. Since data attributes are categorical, we assess anomaly in nominal value combinations which lead to suspiciousness of the claim. We choose the Hamming distance and Hamming distance based kernels in generating spectral ranking for this data set. SRA yields an impressive 0.74 AUC, which is close to 0.83 AUC generated by the supervised RF.

We also compare our SRA method with two state-of-the-art anomaly detection methods, OC-SVM and LOF, using several synthetic data sets and several real data sets. We show that SRA can be an effective alternative to OC-SVM and LOF and can sometimes outperform OC-SVM and LOF.

5.1 Future Work

Here we list several directions for extending the proposed SRA algorithm in the future:

1. In this thesis, we mainly utilize the 1st non-principal eigenvector. We demonstrate several examples where, by combining the SRA anomaly scores from the 1st and 2nd non-principal eigenvectors using a simple summation, better performance can be achieved. How to best combine the SRA anomaly scores from multiple eigenvectors can be one of the future work.

2. An alternative to cope with cases where more than two major patterns exist is to iteratively apply the SRA with the 1st non-principal eigenvector. In other words, we further segment both or one of the subgroups $C^+ = \{i : (\overline{z_1})_i \geq 0\}$ and $C^- = \{i : (\overline{z_1})_i < 0\}$. This is another direction to extend our proposed SRA algorithm.

3. As we have mentioned in Section 3.1, the principal eigenvector of the transition matrix P is the stationary distribution which can be used for detecting point anomalies. How to combine the anomaly scores from SRA with the stationary distribution π is also one of the interesting direction to extend the SRA algorithm.

4. How to construct appropriate similarity matrices based on the given data sets for SRA and spectral clustering is also another future direction to explore.

Appendix A

Proof of Theorem 3

Recall that the unsupervised SVM formulation is :

$$\min_{y_i \in \{+1,-1\}} \left\{ \begin{array}{cc} \max_\alpha & e^T\alpha - \frac{1}{2}\alpha^T YKY\alpha \\ \text{subject to} & y^T\alpha = 0 \\ & 0 \le \alpha_i \le C, i = 1, \ldots, n \end{array} \right\} \tag{A.1}$$

The inner optimization problem of (A.1) can be rewritten as:

$$\begin{aligned} \max_z \quad & e^T|z| - \frac{1}{2}z^T K z \\ \text{subject to} \quad & e^T z = 0 \\ & |z_i| \le C, \quad i = 1, \ldots, n \end{aligned} \tag{A.2}$$

where $z_i = \alpha_i y_i, \quad i = 1, ..., n$. We note that $e^T|z|$ is convex and (A.2) has many local maximizers. The nonconvex relaxation of the unsupervised SVM is :

$$\begin{aligned} \min_z \quad & -\frac{1}{2}z^T K z \\ \text{subject to} \quad & e^T z = 0 \\ & |z_i| \le C, \quad i = 1, \ldots, n \end{aligned} \tag{A.3}$$

Theorem 3. Suppose that K is symmetric positive definite. Let(α^*,y^*) be a solution to the unsupervised SVM (A.1). Assume that the solution z^* to (A.3) is a local maximizer of (A.2) and satisfies $e^T|z^*| = e^T\alpha^*$. Then $a_z^* = |z^*|$ and $y_z^* = \text{sign}(z^*)$ solves the unsupervised SVM (A.1).

Proof. Assume that (α^*, y^*) solves the unsupervised SVM (A.1). Let $Y^* = \text{diag}(y^*)$ and $Y_z^* = \text{diag}(y_z^*)$. Then $Y^*\alpha^*$ is a local maximizers of (A.2) and has the smallest objective function value among all local maximizers.

It is clear that $Y^*\alpha^*$ is a feasible point for the relaxation problem (A.3). Since z^* solves (A.3), we have

$$-\frac{1}{2}\alpha_z^{*T} Y_z^* K Y_z^* \alpha_z^* \leq -\frac{1}{2}\alpha^{*T} Y^* K Y^* \alpha^*$$

From the assumption $e^T|z^*| = e^T\alpha^*$, we have

$$-\frac{1}{2}\alpha_z^{*T} Y_z^* K Y_z^* \alpha_z^* + e^T\alpha_z^* \leq -\frac{1}{2}\alpha^{*T} Y^* K Y^* \alpha^* + e^T\alpha^*$$

From above and the assumption that z^* is a local maximizer of (A.2), we conclude that $\alpha_z^* = |z^*|$ and $y_z^* = \text{sign}(z^*)$ solves the unsupervised SVM (A.1). This completes the proof. □

Appendix B

Out of Sample Prediction

In previous discussion, we have encountered four different matrices:

- unnormalized Laplacian matrix : $L = D - W$
- normalized Laplacian matrix: $L_{rw} = I - D^{-1}W$
- symmetric normalized Laplacian matrix: $L_{sym} = I - D^{-1/2}WD^{-1/2}$
- transition matrix: $P = WD^{-1}$

Note that if u is an eigenvector of L_{rw} with eigenvalue $\overline{\lambda}$ then $g = D^{1/2}u$ is an eigenvector of L_{sym} with eigenvalue $\overline{\lambda}$. Similarly, if g is an eigenvector of L_{sym} with eigenvalue $\overline{\lambda}$ then $\overline{z} = D^{1/2}g$ is an eigenvector of P with eigenvalue $1 - \overline{\lambda}$, where

$$\overline{z} = D^{1/2}g = Du \ , \ g = D^{1/2}u \tag{B.1}$$

SRA algorithm is based on $\overline{z_1} = D^{1/2}g_1$, which is the eigenvector for P associated with the second largest eigenvalue. Let u_1 be the eigenvector for L_{rw} associated with the smallest nonzero eigenvalue λ_1, i.e,

$$L_{rw}u_1 = \lambda_1 u_1 \tag{B.2}$$

Thus for the jth element $(u_1)_j$ of u_1, we have:

$$\lambda_1(u_1)_j = (u_1)_j - \sum_{i=1}^{n} \frac{1}{d_j} W_{ij}(u_1)_i \Leftrightarrow (1 - \lambda_1)(u_1)_j = \frac{1}{d_j} \sum_{i=1}^{n} W_{ij}(u_1)_i \tag{B.3}$$

From $\overline{z} = D^{1/2}g = Du$, for the the jth element of $\overline{z}_1$, we have:

$$(\overline{z_1})_j = d_j(u_1)_j = \frac{\sum_{i=1}^{n} W_{ij}(u_1)_i}{1-\lambda_1} = \frac{\sum_{i=1}^{n} W_{ij}(u_1)_i}{\widehat{\lambda_1}} \tag{B.4}$$

where $\widehat{\lambda_1} = 1 - \lambda_1$ is the second largest eigenvalue of $P = WD^{-1}$. From (B.4), we can see that the jth element of $\overline{z}_1$ can be viewed as the weighted summation of all the elements of u_1 scaled by the constant $\widehat{\lambda_1}$. The weights are the weights of edges connecting the jth data instance with all the data instances.

Suppose matrix W is positive semi-definite and corresponds to a kernel function $\mathcal{K}(x, y)$, which implicitly computes the inner product $\phi(x)^T\phi(y)$ with a high dimensional feature mapping $\phi(\cdot)$. Consider a specific feature space $\mathcal{H} = \{\phi(x_i)\}$ and we have a training set $X = \{v_1, \ldots, v_n\}$ and an out of sample testing point v_y for prediction. Matrix W is the kernel matrix for training data X. Then we want to find $\alpha = \{\alpha_1, \alpha_2, ..., \alpha_n\}$ such that:

$$\sum_{i=1}^{n} \alpha_i \phi(v_i) = \phi(X)\alpha = \phi(v_y) \tag{B.5}$$

where $\phi(X) = [\phi(v_1), \ldots, \phi(v_n)]$. Hence,

$$\phi(X)^T\phi(X)\alpha = \phi(X)^T\phi(v_y)$$

Consequently,

$$W\alpha = k_y \tag{B.6}$$

where $k_y = \phi(X)^T\phi(v_y)$. More explicitly, we have

$$W = \begin{bmatrix} W_{11} & \ldots & W_{1n} \\ \vdots & \ddots & \vdots \\ W_{n1} & \ldots & W_{nn} \end{bmatrix} \text{ and } k_y = [W_{1y}, \ldots, W_{ny}]^T$$

Suppose $\overline{z}_1$ is the eigenvector associated with the second largest eigenvector of $P = WD^{-1}$. Then $\overline{z}_y$ of the data instance v_y can be predicted by the weighted score of the training instances:

$$\overline{z}_y = \alpha^T \overline{z}_1 = \overline{z}_1^T \alpha = \overline{z}_1^T (W^{-1}k_y)$$

Since $WD^{-1}\overline{z}_1 = \widehat{\lambda}_1 \overline{z}_1$, we have

$$\overline{z}_1^T = \frac{\overline{z}_1^T D^{-1} W}{\widehat{\lambda}_1}$$

Thus

$$\overline{z}_y = \overline{z}_1^T \alpha = \overline{z}_1^T (W^{-1} k_y) = \frac{\overline{z}_1^T D^{-1} k_y}{\widehat{\lambda}_1}$$

From $\overline{z}_1 = Du_1$ and $D^{-1} W u_1 = \widehat{\lambda}_1 u_1$, we have:

$$\overline{z}_y = \overline{z}_1^T \alpha = \overline{z}_1^T (W^{-1} k_y) = \frac{u_1^T W}{\widehat{\lambda}_1} (W^{-1} k_y) = \frac{u_1^T k_y}{\widehat{\lambda}_1} = \frac{\sum_{i=1}^{n} W_{iy} (u_1)_j}{\widehat{\lambda}_1} \tag{B.7}$$

We notice that (B.4) is similar to (B.7). In other words, the prediction of the $\overline{z}_y$ is again represented as a weighted summation of all the elements of u_1 scaled by the constant $\widehat{\lambda}_1$. Thus, the prediction of the anomaly score is set to $f_y = \max(|\overline{z}_1|) - |\overline{z}_y|$ or $f_y = \pm z_y$, depending on whether to output a score with respect to one pattern or not. The prediction procedure discussed above can also be applied to other non-principal eigenvectors. Therefore, the prediction procedure still works in the cases where we want to combine the anomaly scores from more than one non-principal eigenvectors.

Appendix C

One-Class Support Vector Machine (OC-SVM)

One-Class Support Vector Machine (OC-SVM) is proposed by Schölkopf et al. [42] for estimating the region where most of the data instances occur in a feature space. The feature space usually is the derived feature space implicitly defined by a specific kernel function. The OC-SVM is based on the ν-SVM proposed by Schölkopf et al. [44].

The OC-SVM tries to separate the unlabelled training data set $X = \{x_1, \ldots, x_n\}$, where $x_i \in \chi \subseteq \Re^d$, from the origin in $\Re^d$ using a hyperplane $w^T x = p$ where the distance between the origin and the hyperplane, $\frac{p}{\|w\|_2}$, is maximized and most of the data instances are in the half space $w^T x \geq p$. Penalized slack variables, ξ_i, are introduced to allow some data instances to be in the wrong half space $w^T x < p$ as shown in Figure C.1. The corresponding primal optimization problem is as follows:

$$\begin{aligned} \min_{w,p,\xi} \quad & \frac{1}{2}\|w\|_2^2 + \frac{1}{\nu n}\sum_{i=1}^{n} \xi_i - p \\ \text{subject to} \quad & w^T x_i \geq p - \xi_i, i = 1, \ldots, n \\ & \xi_i \geq 0, i = 1, \ldots, n \end{aligned} \tag{C.1}$$

where $0 < \nu \leq 1$ is a parameter that controls the tradeoff between maximizing the distance between origin and the the hyperplane, $w^T x = p$, and containing most of the data in the normal region, $w^T x \geq p$, created by the hyperplane as shown in Figure C.1.

Problem (C.1) is a strictly convex quadratic programming problem. The solution can

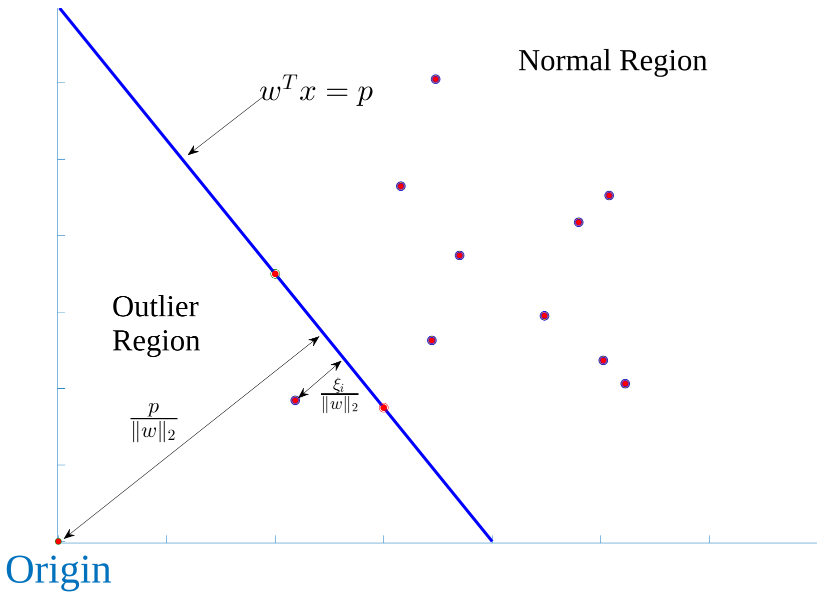


Figure C.1: Illustration of OC-SVM

be obtained by its dual problem [42]:

$$\begin{aligned}
\min_{\alpha} \quad & \sum_{i=1}^{n}\sum_{j=1}^{n} \alpha_i \alpha_j x_i^T x_j \\
\text{subject to} \quad & 0 \le \alpha_i \le \frac{1}{\nu n}, i = 1, \ldots, n \\
& \sum_{i=1}^{n} \alpha_i = 1
\end{aligned} \tag{C.2}$$

Let the optimal solution from problem (C.2) be $\widehat{\alpha}^*$. We denote the set of support vectors as $\widehat{SV} = \{x_i | \widehat{\alpha}_i^* \neq 0\}$ and set of outliers as $\widehat{O} = \{x_i | \xi_i \neq 0\}$. It can be shown [43] that ν is an upper bound on the fraction of outliers and a lower bound on the fraction of support vectors. Detailed proof and explanation are shown in [43].

Let $\mathcal{K}(x, y)$ be a kernel function corresponding to a feature map $\theta : \chi \subseteq \Re^d \rightarrow \mathcal{H} \subseteq \Re^{d'}$.

The OC-SVM in the derived feature space $\mathcal{H}$ is then:

$$\begin{aligned} \min_{\alpha} \quad & \sum_{i=1}^{n}\sum_{j=1}^{n} \alpha_i \alpha_j \mathcal{K}(x_i, x_j) \\ \text{subject to} \quad & 0 \leq \alpha_i \leq \frac{1}{\nu n}, i = 1, \ldots, n \\ & \sum_{i=1}^{n} \alpha_i = 1 \end{aligned} \tag{C.3}$$

After computing the optimal $\widetilde{\alpha}^*$ for problem (C.3), p^* can be recovered from any x_i that satisfies $0 < \widetilde{\alpha}_i^* < \frac{1}{\nu n}$:

$$p^* = \sum_{j=1}^{n} \widetilde{\alpha}_j^* \mathcal{K}(x_j, x_i)$$

The binary decision function for anomaly detection is then:

$$f(x) = \text{sign}\left(\sum_{i=1}^{n} \widetilde{\alpha}_i^* \mathcal{K}(x, x_i) - p^* \right) \tag{C.4}$$

It can be shown [43] that finding the hyperplane $w^T \phi(x) = p$ that separates the training data set from the origin in a derived feature space, where the distance between the origin and the hyperplane, $\frac{p}{\|w\|_2}$, is maximized, is equivalent to finding a hypersphere with the smallest radius in the derived feature space $\mathcal{H}$ that encloses the training data set. Formally, let R be the radius of the hypershpere. We can put most of the data instances into a small hypershpere with its radius minimized. The optimization problem is then:

$$\begin{aligned} \min_{R, \xi, c} \quad & R^2 + \frac{1}{\nu n} \sum_{i}^{n} \xi_i \\ \text{subject to} \quad & \|\phi(x_i) - c\|_2^2 \leq R^2 + \xi_i, i = 1, \ldots, n \\ & \xi_i \geq 0, i = 1, \ldots, n \end{aligned} \tag{C.5}$$

where $c \in \mathcal{H}$ is the center of the hypersphere in a derived space $\mathcal{H}$. This leads to its dual problem:

$$\begin{aligned} \min_{\alpha} \quad & \sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_i\alpha_j\mathcal{K}(x_i,x_j)-\sum_{i=1}^{n}\alpha_i\mathcal{K}(x_i,x_i) \\ \text{subject to} \quad & 0\le\alpha_i\le\frac{1}{\nu n}, i=1,\dots,n \\ & \sum_{i=1}^{n}\alpha_i=1 \end{aligned} \tag{C.6}$$

After obtaining the optimal solution $\widetilde{\alpha}^*$ of (C.6), the center of the hypersphere c is then given by $c=\sum_{i=1}^{n}\widetilde{\alpha}_i^*\phi(x_i)$ [42]. The corresponding decision function is then:

$$f_2(x)=\operatorname{sign}\left(R^2-\sum_{i=1}^{n}\sum_{j=1}^{n}\widetilde{\alpha}_i^*\widetilde{\alpha}_j^*\mathcal{K}(x_i,x_j)+2\sum_{i}^{n}\widetilde{\alpha}_i^*\mathcal{K}(x_i,x)-\mathcal{K}(x,x)\right) \tag{C.7}$$

where R can be recovered by solving $f_2(x_i)=0$ with any x_i where $0<\alpha_i<\frac{1}{\nu n}$. The square of the distance $\|\phi(x)-c\|_2^2$ is then:

$$d(x)=\sum_{i=1}^{n}\sum_{j=1}^{n}\widetilde{\alpha}_i^*\widetilde{\alpha}_j^*\mathcal{K}(x_i,x_j)-2\sum_{i}^{n}\widetilde{\alpha}_i^*\mathcal{K}(x_i,x)+\mathcal{K}(x,x) \tag{C.8}$$

Note that for many kernel functions, such as Gaussian kernel, $\mathcal{K}(x_i,x_i)$ is a constant for $1\le i\le n$. In addition, the second constraint in (C.3) and (C.6) requires that $\sum_{i=1}^{n}\alpha_i=1$. Therefore, under the assumption that $\mathcal{K}(x_i,x_i)$ is a constant, (C.3) is equivalent to (C.6) since

$$\sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_i\alpha_j\mathcal{K}(x_i,x_j)-\sum_{i=1}^{n}\alpha_i\mathcal{K}(x_i,x_i)$$

is equivalent to

$$\sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_i\alpha_j\mathcal{K}(x_i,x_j)$$

when $\mathcal{K}(x_i,x_i)$ is a constant and $\sum_{i=1}^{n}\alpha_i=1$.

The second formulation (C.6) of OC-SVM enables us to generate ranking scores using (C.8) instead of binary predictions. The square of distance $\|\phi(x)-c\|_2^2$, between $\phi(x)$ and c, can be used to compute the anomaly ranking score of the data instance x. The predicted label can be obtained by thresholding the function $d(x)$ instead of using the sign of $R^2 - d(x)$. It can be shown that [43], under the condition that $\mathcal{K}(x_i, x_i)$ is a constant for $1 \leq i \leq n$, decision function (C.4) is equivalent to decision function (C.7). Therefore thresholding the function $d(x)$ is equivalent to thresholding

$$\widetilde{f}(x) = w_*^T \theta(x) - p^* = \sum_{i=1}^{n} \widetilde{\alpha}_i^* \mathcal{K}(x, x_i) - p^*$$

Appendix D

Deduction of Classification Trees

Recall that, to deduce a regression tree, we start with the full feature space $\Re^d$ and we want to find an attribute j and a split point s to split the space into two regions:

$$r_1(j,s) = \{x_i | (x_i)_j \leq s\} \text{ and } r_2(j,s) = \{x_i | (x_i)_j > s\}$$

so that the sum of squared error is minimized:

$$\min_{j,s} \left(\sum_{x_i \in r_1(j,s)} (y_i - C_1) + \sum_{x_i \in r_2(j,s)} (y_i - C_2) \right)$$

where C_1 is the average of the training output targets of all $x_i \in r_1(j,s)$ and C_2 is the average of the training output targets of all $x_i \in r_2(j,s)$. After we find the best partition strategy for the whole data set, we have two regions. The same procedure is adopted to further partition all the resulting regions until certain stopping criteria are reached

For a classification problem, we use weighted sum of certain impurity measures to replace the sum of squared errors. Let n_m be the number of training data instances inside the region r_m. Suppose that the output target y_i can take k possible values, i.e, $y_i \in F = \{t_1, \ldots, t_k\}$. Let h_m be the value of output target y_i that occurs most frequently inside region r_m and p_{jm} be the fraction of output targets that takes the value t_j inside region r_m. Three impurity measures can then be used for a specific region r_m:

- Misclassification error:

$$Error_m = \frac{1}{n_m} \sum_{x_i \in r_m} \delta(h_m, y_i) \ , \ \delta(h_m, y_i) = \begin{cases} 1, & h_m \neq y_i \\ 0, & h_m = y_i \end{cases}$$

- Gini Index:

$$Gini_m = 1 - \sum_{i=1}^{k} p_{im}^2$$

- Information Entropy:

$$Info_m = -\sum_{i=1}^{k} p_{im} \log p_{im}$$

CART uses Gini index as the impurity measure while other tree based methods such as C4.5 and ID3 use information entropy [25]. These impurity measures evaluate the homogeneity within different regions. For example, given a region where all the training data instances inside the region have the same value for the output targets, all three impurity measures mentioned above for this region would be 0. Ideally, we want to partition the space into different regions where the impurity measures for each of the region is 0. Thus, these measures provide criteria for partitioning the feature space. Taking a classification problem on a real valued training data set, i.e., $x_i \in \Re^d$, as an example. Let the total number of data instances be n and n_m be the number of training data instances inside the region r_m. We start with the full feature space $\Re^d$ and we want to find an attribute j and a split point s to split the space into two regions:

$$r_1(j, s) = \{x_i | (x_i)_j \leq s\} \text{ and } r_2(j, s) = \{x_i | (x_i)_j > s\}$$

where the weighted sum of a specific impurity measure of $r_1(j, s)$ and $r_2(j, s)$ is minimized:

$$\min_{j,s} \left(\frac{n_1}{n} Error_1 + \frac{n_2}{n} Error_2 \right) \quad \text{(Misclssisification Error)}$$

or

$$\min_{j,s} \left(\frac{n_1}{n} Gini_1 + \frac{n_2}{n} Gini_2 \right) \quad \text{(Gini Index)}$$

or

$$\min_{j,s} \left(\frac{n_1}{n} Info_1 + \frac{n_2}{n} Info_2 \right) \quad \text{(Information Entropy)}$$

Again, we repeat the same procedure to further partition all the resulting regions until certain stopping criteria are reached or the impurity measure is already 0 for the region we want to split.

References

[1] Ai, J., Brockett, P. L., and Golden, L. L. (2009). Assessing consumer fraud risk in insurance claims: An unsupervised learning technique using discrete and continuous predictor variables. *North American Actuarial Journal*, 13(4):438–458.

[2] Ai, J., Brockett, P. L., Golden, L. L., and Guillén, M. (2012). A robust unsupervised method for fraud rate estimation. *Journal of Risk and Insurance*.

[3] Asuncion, A. and Newman, D. (2007). UCI machine learning repository.

[4] Belkin, M. and Niyogi, P. (2001). Laplacian eigenmaps and spectral techniques for embedding and clustering. In *NIPS*, volume 14, pages 585–591.

[5] Boriah, S., Chandola, V., and Kumar, V. (2008). Similarity measures for categorical data: A comparative evaluation. *red*, 30(2):3.

[6] Bradley, A. P. (1997). The use of the area under the roc curve in the evaluation of machine learning algorithms. *Pattern Recognition*, 30:1145–1159.

[7] Breiman, L. (2001). Random forests. *Machine learning*, 45(1):5–32.

[8] Breiman, L., Friedman, J., Stone, C. J., and Olshen, R. A. (1984). *Classification and regression trees*. CRC press.

[9] Breunig, M. M., Kriegel, H.-P., Ng, R. T., and Sander, J. (2000). LOF: identifying density-based local outliers. In *ACM Sigmod Record*, volume 29, pages 93–104. ACM.

[10] Brockett, P. L., Derrig, R. A., Golden, L. L., Levine, A., and Alpert, M. (2002). Fraud classification using principal component analysis of ridits. *Journal of Risk and Insurance*, 69(3):341–371.

[11] Brockett, P. L., Xia, X., and Derrig, R. A. (1998). Using Kohonen's self-organizing feature map to uncover automobile bodily injury claims fraud. *Journal of Risk and Insurance*, pages 245–274.

[12] Chandola, V., Banerjee, A., and Kumar, V. (2009). Anomaly detection: A survey. *ACM Computing Surveys*, 41(3):15.

[13] Chang, C.-C. and Lin, C.-J. (2011). LIBSVM: a library for support vector machines. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 2(3):27.

[14] Chapelle, O., Sindhwani, V., and Keerthi, S. S. (2008). Optimization techniques for semi-supervised support vector machines. *The Journal of Machine Learning Research*, 9:203–233.

[15] Couto, J. (2005). Kernel k-means for categorical data. In *Advances in Intelligent Data Analysis VI*, pages 46–56. Springer.

[16] Cowling, M. G. (1983). Harmonic analysis on semigroups. *Annals of Mathematics*, pages 267–283.

[17] Cristianini, N. and Shawe-Taylor, J. (2000). *An introduction to support vector machines and other kernel-based learning methods*. Cambridge university press.

[18] David, G. and Averbuch, A. (2012). Spectralcat: Categorical spectral clustering of numerical and nominal data. *Pattern Recognition*, 45(1):416–433.

[19] DeLong, E. R., DeLong, D. M., and Clarke-Pearson, D. L. (1988). Comparing the Areas under Two or More Correlated Receiver Operating Characteristic Curves: A Nonparametric Approach. *Biometrics*, 44(3):837–845.

[20] Derrig, R. A. (2002). Insurance fraud. *Journal of Risk and Insurance*, 69(3):271–287.

[21] Eskin, E., Arnold, A., Prerau, M., Portnoy, L., and Stolfo, S. (2002). A geometric framework for unsupervised anomaly detection. In *Applications of Data mining in Computer Security*, pages 77–101. Springer.

[22] Gower, J. C. (1971). A general coefficient of similarity and some of its properties. *Biometrics*, pages 857–871.

[23] Haasdonk, B. (2005). Feature space interpretation of svms with indefinite kernels. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 27(4):482–492.

[24] Haofan, Z., Ke, N., Thomas, F. C., and Yuying, L. Spectral ranking and unsupervised feature selection for point, collective and contextual anomaly detection. *under preparation.*

[25] Hastie, T., Tibshirani, R., Friedman, J., Hastie, T., Friedman, J., and Tibshirani, R. (2009). *The elements of statistical learning*, volume 2. Springer.

[26] He, Z., Xu, X., and Deng, S. (2003). Discovering cluster-based local outliers. *Pattern Recognition Letters*, 24(9):1641–1650.

[27] Herbrich, R., Graepel, T., and Obermayer, K. (1999). Large margin rank boundaries for ordinal regression. *Advances in Neural Information Processing Systems*, pages 115–132.

[28] Jayasumana, S., Hartley, R., Salzmann, M., Li, H., and Harandi, M. (2013). Kernel methods on the Riemannian manifold of symmetric positive definite matrices. In *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on*, pages 73–80. IEEE.

[29] Knorr, E. M., Ng, R. T., and Tucakov, V. (2000). Distance-based outliers: algorithms and applications. *The International Journal on Very Large Data Bases*, 8(3-4):237–253.

[30] Kriegel, H.-P., Kröger, P., Schubert, E., and Zimek, A. (2011). Interpreting and unifying outlier scores. In *SDM*, pages 13–24. SIAM.

[31] Lodhi, H., Saunders, C., Shawe-Taylor, J., Cristianini, N., and Watkins, C. (2002). Text classification using string kernels. *The Journal of Machine Learning Research*, 2:419–444.

[32] Luss, R. and d'Aspremont, A. (2008). Support vector machine classification with indefinite kernels. In *Advances in Neural Information Processing Systems*, pages 953–960.

[33] Markou, M. and Singh, S. (2003). Novelty detection: a review-part 1: statistical approaches. *Signal Processing*, 83(12):2481–2497.

[34] Metz, C. E. (1978). Basic principles of ROC analysis. *Seminars in nuclear medicine*, 8(4):283–298.

[35] Moonesinghe, H. and Tan, P.-N. (2006). Outlier detection using random walks. In *Tools with Artificial Intelligence, 2006. ICTAI'06. 18th IEEE International Conference on*, pages 532–539. IEEE.

[36] Ng, A. Y., Jordan, M. I., Weiss, Y., et al. (2002). On spectral clustering: Analysis and an algorithm. *Advances in Neural Information Processing Systems*, 2:849–856.

[37] Nian, K., Zhang, H., Tayal, A., Coleman, T. F., and Li, Y. (2014). Auto insurnace fraud detection using unsupervised spectral ranking for anomaly. *Journal of Risk and Insurance*, submitted.

[38] Page, L., Brin, S., Motwani, R., and Winograd, T. (1999). The pagerank citation ranking: Bringing order to the web.

[39] Phua, C., Alahakoon, D., and Lee, V. (2004). Minority report in fraud detection: classification of skewed data. *ACM SIGKDD Explorations Newsletter*, 6(1):50–59.

[40] Phua, C., Lee, V., Smith, K., and Gayler, R. (2010). A comprehensive survey of data mining-based fraud detection research. *arXiv preprint arXiv:1009.6119*.

[41] Provost, F. J., Fawcett, T., and Kohavi, R. (1998). The case against accuracy estimation for comparing induction algorithms. In *ICML*, volume 98, pages 445–453.

[42] Schölkopf, B., Platt, J. C., Shawe-Taylor, J., Smola, A. J., and Williamson, R. C. (2001). Estimating the support of a high-dimensional distribution. *Neural Computation*, 13(7):1443–1471.

[43] Scholkopf, B. and Smola, A. J. (2001). *Learning with kernels: support vector machines, regularization, optimization, and beyond*. MIT press.

[44] Schölkopf, B., Smola, A. J., Williamson, R. C., and Bartlett, P. L. (2000). New support vector algorithms. *Neural computation*, 12(5):1207–1245.

[45] Shi, J. and Malik, J. (2000). Normalized cuts and image segmentation. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 22(8):888–905.

[46] Smits, G. F. and Jordaan, E. M. (2002). Improved svm regression using mixtures of kernels. In *Neural Networks, 2002. IJCNN'02. Proceedings of the 2002 International Joint Conference on*, volume 3, pages 2785–2790. IEEE.

[47] Tang, J., Chen, Z., Fu, A. W.-C., and Cheung, D. W. (2002). Enhancing effectiveness of outlier detections for low density patterns. In *Advances in Knowledge Discovery and Data Mining*, pages 535–548. Springer.

[48] Tax, D. (2014). DDtools, the data description toolbox for matlab. version 2.1.1.

[49] Tayal, A., Coleman, T. F., and Li, Y. (2013a). Bounding the difference between rankrc and ranksvm and an application to multi-level rare class kernel ranking. *Journal on Machine Learning Research*, submitted.

[50] Tayal, A., Coleman, T. F., and Li, Y. (2013b). Rankrc: Large-scale nonlinear rare class ranking. *IEEE Transactions on Knowkedge and Data Engineering*, submitted.

[51] Tennyson, S. and Salsas-Forn, P. (2002). Claims auditing in automobile insurance: fraud detection and deterrence objectives. *Journal of Risk and Insurance*, 69(3):289–308.

[52] Vapnik, V. N. and Vapnik, V. (1998). *Statistical learning theory*, volume 2. Wiley New York.

[53] Von Luxburg, U. (2007). A tutorial on spectral clustering. *Statistics and Computing*, 17(4):395–416.

[54] Wang, X. and Davidson, I. (2009). Discovering contexts and contextual outliers using random walks in graphs. In *Data Mining, 2009. ICDM'09. Ninth IEEE International Conference on*, pages 1034–1039. IEEE.

[55] Yeung, D.-Y. and Chow, C. (2002). Parzen-window network intrusion detectors. In *Pattern Recognition, 2002. Proceedings. 16th International Conference on*, volume 4, pages 385–388. IEEE.