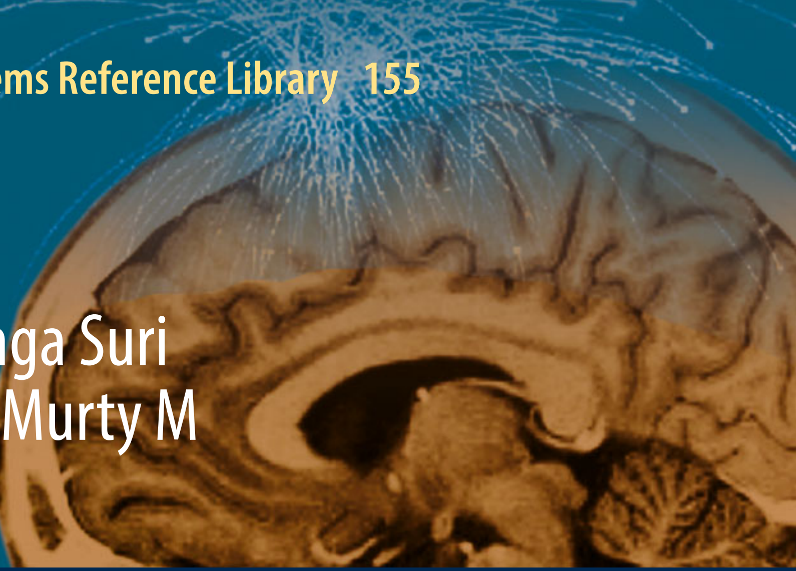
Intelligent Systems Reference Library 155



N. N. R. Ranga Suri
Narasimha Murty M
G. Athithan

Outlier Detection: Techniques and Applications

A Data Mining Perspective

Springer

Intelligent Systems Reference Library

Volume 155

N. N. R. Ranga Suri · Narasimha Murty M
G. Athithan

Outlier Detection: Techniques and Applications

A Data Mining Perspective

Springer

N. N. R. Ranga Suri
Centre for Artificial Intelligence
and Robotics (CAIR)
Bangalore, India

G. Athithan
Defence Research and Development
Organization (DRDO)
New Delhi, India

Narasimha Murty M
Department of Computer Science
and Automation
Indian Institute of Science (IISc)
Bangalore, India

ISSN 1868-4394 ISSN 1868-4408 (electronic)
Intelligent Systems Reference Library
ISBN 978-3-030-05125-9 ISBN 978-3-030-05127-3 (eBook)
https://doi.org/10.1007/978-3-030-05127-3

Library of Congress Control Number: 2018962773

Dedicated to
Our Beloved Family Members
for continuously motivating and supporting
us in writing this book

N. N. R. Ranga Suri
Narasimha Murty M
G. Athithan

Preface

The topic of identifying objects that stand out from the norm in a data set is an exciting one in the field of data mining. Detecting such objects, known as outliers, is important for many applications such as fraud detection, network analysis, etc. Given the significance of outlier detection in real life scenarios, we chose to focus on this challenging research topic in this book.

Contents of this book are mainly derived from the research work carried out towards Ph.D. thesis of the first author at Indian Institute of Science (IISc), Bangalore, under the supervision of the co-authors. The book itself is the outcome of the recommendation made by Prof. Lakhmi C. Jain, University of Canberra, suggesting us to produce a monograph based on our understanding of this research area.

The first part of the book touches upon the significant aspects of the problem that are useful in carrying out further research on related issues. We introduce the concept of outliers through some theoretical ideas, along with our understanding of its practical significance. We discuss various methodologies for outlier detection and present our experience specific to some problem settings.

The second part deals with the problem of outlier detection in networks, highlighting the need for analyzing various kinds of interaction networks that enable exchange of information among people. In this connection, we explain the concept of anomalous sub-graphs in a communication network/graph. We furnish our research results obtained using various algorithmic schemes along with the details of a few recent techniques for detection of anomalous sub-graphs.

Thus, the objective behind producing this book is to shed light on various issues that researchers working in outlier detection generally face, and share some interesting developments and references to the latest results in this area. We hope that the book would be useful to researchers working in data mining as a guidepost.

Bengaluru, India
September 2018

N. N. R. Ranga Suri
Narasimha Murty M
G. Athithan

Acknowledgements

The authors would like to express their sincere thanks to Prof. Lakhmi C. Jain, University of Canberra, for his constructive suggestions on the research work carried out by the first author and recommending the same to be published as a monograph under the ISRL book series by Springer-Verlag. The authors acknowledge the excellent support received from the Springer publication team in materializing this book though the manuscript submission got delayed due to unforeseen reasons. The authors are thankful to researchers who have provided various online resources such as data mining tools and benchmark data sets. They also thank the anonymous reviewers of this work for their valuable comments and suggestions.

The first author sincerely thanks his employer Centre for Artificial Intelligence and Robotics (CAIR), DRDO, Bangalore for supporting this research work and allowing him to produce this book. He is thankful to his Director and colleagues for their continued encouragement in pursuing this research work. He acknowledges the conducive academic environment provided by Indian Institute of Science (IISc), Bangalore, for carrying out this research work. He also thanks his research supervisors Prof. M. N. Murty M and Dr. G. Athithan for their valuable guidance during his Ph.D. course at IISc.

He is deeply indebted to his parents for motivating him to take up research-oriented career. He likes to thank his wife, kids, and all the members of his extended family for their timely encouragement during this work. He believes that he could contribute towards this monograph due to the divine blessings of Lord Rama. Finally, he thanks everyone who helped him directly or indirectly in this effort.

The second author would like to thank Mr. Govind Sharma, Mr. Sharad Nandanwar, Mr. Sambaran Bandyopadhyay, and Dr. Ramasuri Narayanam for motivating him to work on social networks and information networks.

The third author thanks the first author for taking into account all the supervisory advice that was directed at him during his Ph.D. dissertation. He is thankful to his employer, namely, the DRDO for allowing him and the first author to spare their time towards preparation of the manuscript. Finally, he thanks his family and Providence for endowing him with an ecosystem that was conducive to the process of bringing out this book.

Contents

Acronyms

AAAI	Association for the Advancement of Artificial Intelligence
ANMI	Average Normalized Mutual Information
ANN	Approximate Nearest Neighbor
AS	Autonomous Systems
AUC	Area Under Curve
AVF	Attribute Value Frequency
BGP	Border Gateway Protocol
BIRCH	Balanced Iterative Reducing and Clustering using Hierarchies
CBLOF	Cluster-Based Local Outlier Factor
DBLP	DataBase and Logic Programming (Website at University of Trier)
DBSCAN	Density-Based Spatial Clustering of Applications with Noise
DDoS	Distributed Denial of Service
DSD	Dense Subgraph Discovery
EDPL	Egonet Density Power Law
EER	Equal Error Rate
FPR	False Positive Rate
FSD	Frequent Subgraph Discovery
IG	Information Gain
IGFS	Information Gain-based Feature Selection
IQR	Inter-Quartile Range
IS	Information System
KDD	Knowledge Discovery in Databases
KD-tree	K-Dimensional tree
LBSN	Location-Based Social Network
LOCI	LOcal Correlation Integral
LOF	Local Outlier Factor
LRD	Local Reachability Density
LS	Likely Set
MDL	Minimum Description Length
MI	Mutual Information

ML	Machine Learning
MMR	Min-Min-Roughness
mRMR	minimum Redundancy and Maximum Relevance
NMF	Non-negative Matrix Factorization
NMI	Normalized Mutual Information
PRC	Precision-Recall Curve
RKM	Rough K-Means
ROAD	Ranking-based Outlier Analysis and Detection
ROC	Receiver Operating Characteristic
ROCK	RObust Clustering using linKs
SAR	Significant Anomalous Region
SDPL	Subgraph Density Power Law
SNAP	Stanford Network Analysis Platform
SVM	Support Vector Machines
TPR	True Positive Rate
UCI	University of California, Irvine
VDM	Value Difference Metric

List of Figures

List of Tables

Part I
Techniques for Outlier Detection

The emphasis here is on exploring the practical significance of the outlier detection problem in real life applications and understanding the intricacies involved in dealing with this problem. Accordingly, initial chapters deliberate on various research issues associated with outlier detection along with a review of the current literature available in this regard.

In subsequent chapters, some of the key research issues are dealt with presenting various techniques developed addressing these issues. Each chapter presents the theoretical background related to the specific issue being addressed, followed by the description of a few prominent algorithms developed in this regard. Empirical study of these algorithms using benchmark data sets is also furnished towards the end of every chapter to provide a deeper understanding of the outlier discovery process involved.

Chapter 1
Introduction

Abstract This chapter introduces the notion of outlier and its essential properties. It presents a characterization of outliers based on their size, diversity and role in a given context. It also provides a theoretical perspective on the significance of outliers along with the practical need for their detection.

1.1 What is an Outlier?

An object in a data set is usually called an *outlier* if

- It deviates from the normal/known behavior of the data,
- It assumes values that are far away from the expected/average values, or
- It is not connected/similar to any other object in terms of its characteristics.

So, an outlier generally exhibits some abnormality or some kind of out of the way behavior. Even intuitively, such an outlier behavior can play an important role in the decision making process, leading to the following.

1. It can make an individual assume a leadership role and influence others to follow. For example, lateral thinking is one such case.
2. It may play a disturbing role in summarizing the overall behavior of a community of people/entities.

Thus, an outlier can play either a positive role or a negative role based on the application under consideration.

From a theoretical perspective, it is possible to introduce various definitions for outliers based on distinct observed characteristics of the data objects. One such definition states that outlier is an object that deviates so much from other objects as to arouse suspicion that it is generated by a different mechanism. This definition basically captures the severity of deviation of an object leading to an impression that such an object is altogether different from the others.

N. N. R Ranga Suri et al., *Outlier Detection: Techniques and Applications*,
Intelligent Systems Reference Library 155,
https://doi.org/10.1007/978-3-030-05127-3_1

In general, the notion of outliers refers to data objects that deviate significantly from the normal data. The degree of interestingness of such objects can be derived based on a suitable measure of their deviation. The objects that are interesting enough to the analyst are of significance in the analysis process.

1.2 Characterization of Outliers

As stated above, the deviating behavior of outliers can be attributed to various observed properties of the data. Each such property can lead to a specific way of characterization of outliers as explained below.

1.2.1 Size-Based

It is possible to have outliers of varied sizes based on the number of objects involved. Therefore, in an abstract sense, an outlier may quantitatively correspond to one of the following scenarios.

- a single object, or
- a small subset of objects, or
- a community/group of objects, or
- an entire network/graph

Figure 1.1 illustrates the case of a singleton outlier where only one object is involved, in a 2-dimensional feature space. Referring to the data distribution, one can notice the presence of two prominent clusters and an isolated object that is far away from the clusters. Therefore, this object turns out to be an outlier.

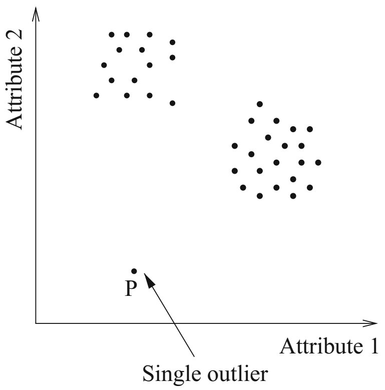


Fig. 1.1 A single outlier in a data set

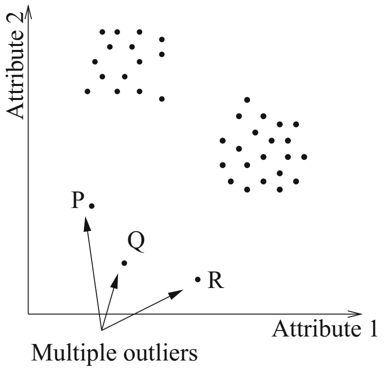


Fig. 1.2 Multiple outliers in a data set

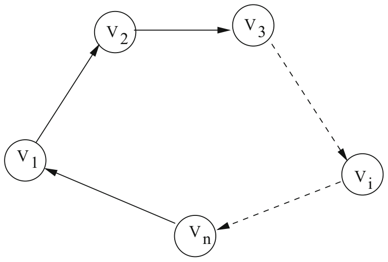


Fig. 1.3 A sample outlier community

It is possible to have more than one outlier in a data set as depicted in Fig. 1.2. There are three isolated objects that are outliers in this case, in addition to the clusters noted above. So, it is possible for a subset of the data to be outliers.

According to network theory, the degree distribution of social networks typically follows a power-law as they belong to the class of scale-free networks. As a result, the number of nodes with degree k will be proportional to $k^{-\alpha}$ and typically α is found to be in the range [2, 3]. Further, large communities in a network are also expected to satisfy the power law degree distribution.

Now, consider a community made up of a collection of n nodes that are of the same degree as in the community shown in Fig. 1.3. More specifically, every node in this example has degree '2'. For a sufficiently large value of n, such a community turns out to be an outlier as the degree distribution of nodes in this community does not satisfy the power law. Similar situation may arise for an entire network, making it an outlier. Thus, it is possible to have an entire community in a social network to be characterized as outlier.

1.2.2 Diversity-Based

Based on the definition of outlier given in Sect. 1.1, one can characterize the same using the notion of diversity as given below.

Table 1.1 A sample 2-dimensional data set

Sl. No.	1	2	3	4	5	6	7	8	9	10
Object	(1, 1)	(2, 2)	(2, 1)	(6, 1)	(6, 2)	(7, 2)	(6, 6)	(6, 7)	(7, 6)	(10, 19)

1. In the case of normally distributed collection of data, it could be an object that is positioned away from the mean based on the variance/covariance structure. For example, in a uni-variate normal distribution, the values in the range $[\mu - 3\sigma, \mu + 3\sigma]$ account for 99.73%. So, any object/value falling outside this range may qualify to be an outlier as it is away from the mean.
2. In data clustering, outliers can impact the resulting grouping structure in an unfavorable manner. For example, consider a two-dimensional data set having 10 objects as shown in Table 1.1.
   Referring to this data, one can visualize 3 clusters each having three objects. Let the initial cluster representatives/centroids for 3 clusters be (1, 1), (6, 7) and (10, 19) respectively. In a typical k-means iteration, one assigns each of the remaining objects (seven of them here) to a cluster based on similarity between the object and the centroid of the cluster. Thus, object assignment to clusters results in the following grouping of objects.

   - C_1: {(1, 1), (2, 2), (2, 1), (6, 1)} and its mean is (2.75, 1.25).
   - C_2: {(6, 7), (6, 2), (7, 2), (6, 6), (7, 6)} with (6.4, 4.6) as its mean.
   - C_3: {(10, 19)} with (10, 19) as its mean/centroid.

   On the other hand, if one ignores the outlier object (10, 19), one would have got the following clustering of the objects by selecting (1, 1), (6, 1) and (6, 7) as initial cluster centroids.

   - C_1: {(1, 1), (2, 2), (2, 1)}
   - C_2: {(6, 1), (6, 2), (7, 2)}
   - C_3: {(6, 7), (6, 6), (7, 6)}

   Looking at these two sets of clustering results, the latter one is intuitively more appealing than the one obtained earlier where the outlier formed a singleton cluster. *So outliers can affect the performance of clustering algorithms*. Here an outlier is an object that is in a very sparse region. Thus, diversity here is characterized by sparsity of the data.
3. An outlier web site can affect the ranking of web pages produced by a search engine. If a search engine ranks pages on fast-loading sites significantly higher than pages on slow-loading sites, then a page on a site which is significantly fast-loading compared to others can get a better ranking. So, here also an outlier site can adversely affect the rankings of a search engine.

4. In the field of *information retrieval*, it is common to remove stop words or frequently used words before indexing of the documents, with the intent of reducing the space and time requirements. For example, in a collection of English documents, words like *the*, *of*, *is* are too frequent and occur in every document. So, *frequent words* will not be useful for discrimination among the documents. Typically, the number of frequent words is small and the number of *rare words* is very large. So, it is good to remove rare words also from the index for reducing the dimensionality of the representation vector. Thus, in this context both frequent and rare words are like outliers.
   This is reflected in the weighing process used to indicate the importance or otherwise of each of the words in the document collection. The most popular weighing method for document indexing is the TF-IDF scheme. For a collection of n documents with a vocabulary size of l, the weighing components are characterized as follows:

   a. *Term Frequency (TF)*: This indicates the frequency of a word $w_i,\ i \in \{1, 2, \ldots, l\}$ in a document $d_j,\ j \in \{1, 2, \ldots, n\}$, denoted as tf_{ij}. It gives more importance to the frequent words over the rare words.
   b. *Inverse Document Frequency (IDF)*: If a word w_i occurs in n_i of the n documents, then $IDF_i = log(\frac{n_i}{n})$. This term annihilates the frequent words and gives more importance to rare words. If a frequent word occurs in all the n documents, then its IDF value is $log(\frac{n}{n}) = 0$.

   Finally, the weight of a word w_i in a document d_j is given by $tf_{ij} * IDF_i$, a product of the two giving larger weights to medium frequency words. This is intuitively appealing as the medium frequency words form the backbone for effective communication and are important in discriminating documents as characterized by the TF-IDF scheme. Here, diversity is characterized in terms of the frequency of the words occurring in a document. Thus, words that are either too frequent or too rare are treated as outliers.

Thus, in many such applications outliers can have diverse end effects requiring suitable characterization of the diversity noticed so as to manage the same in an effective manner.

1.2.3 Role-Based

It is commonly perceived that outliers have an undesirable or negative impact on the data processing activity. Some such examples are presented in the previous subsection. However, there are several situations in which the outliers can play a positive role, as narrated below.

- *Out of the box thinking* is useful for coming up with novel research contributions. Such a thinking pattern is of outlying nature due to its uniqueness.

- In a community of the social elements, it is possible to have a person/entity that is too central or influential. Such a person turns out to be an outlier. However, for the purpose of promoting a product, the influential individual can play a pivotal role as he can positively influence a large section of the persons in his community.
- In classification based on support vector machines (SVMs), the classifier learns the support vectors which form a subset of the training set of patterns. Here, learning is meant for adjusting the weight vector of the linear discriminant function associated with the SVM. All the other patterns (non-support vectors) are discarded or ignored in this process. Typically, support vectors are either boundary patterns or outliers. Thus, outliers are useful for making the classifier learn the patterns by using the subset data.

1.3 Significance of Outliers

It is important to distinguish outliers from random noise typically seen in real life data, as both these kinds of objects are characterized by their deviation from normal data. Further, the issue of what constitutes a sufficient deviation for an object to be considered as an outlier is often a subjective judgment based on the interest of the analyst. While the noise in itself may not be interesting to the analyst, its identification and removal continues to be an important data mining task. Thus, both noise and outlier detection problems are equally important for effective data analysis. In this context, the best way to find outliers and distinguish them from noise is to use the feedback from previously known outlier examples of interest.

In unsupervised scenarios, noise represents the semantic boundary between normal data and true outliers. Thus, it is modeled as a weak form of outliers. From a theoretical perspective, supervised detection algorithms are expected to be more effective, as the characteristics of known examples can be used to sharpen the search process towards more relevant outliers. Therefore, the crux of the problem is that outliers need to be unusual in an interesting way, and the supervision process guides what one might find interesting. While the unsupervised methods can be used either for noise removal or outlier detection, the supervised methods are more suited for application-specic outlier detection.

1.3.1 A Theoretical Perspective

It is understood that many real world applications are faced with the problem of identifying *complex rare events*, which are characterized by contextual combinations of specific actions, often performed by multiple agents. This basically involves two important data exploratory tasks: (a) discovering previously unknown event types, and (b) detecting instances of these event types in the data.

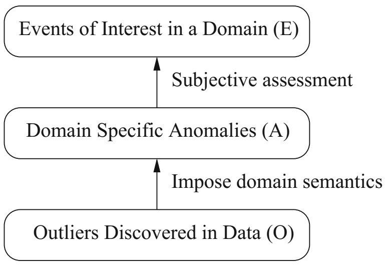

Events of Interest in a Domain (E)
Subjective assessment
Domain Specific Anomalies (A)
Impose domain semantics
Outliers Discovered in Data (O)

Fig. 1.4 Transformation from Outliers to Events of Interest

Towards the discovery of complex rare events, a three-level abstraction of such events is envisaged, with domain specific assessment being the essential mechanism of transformation between these levels, as shown in Fig. 1.4. This is because events of interest in an application are characterized not by their rarity but rather by their inexplicability in terms of applicable real-world behaviors that are known a priori. In such application scenarios, outlier detection techniques act as useful starting points for a larger discovery process. Further, it is argued that the associated explanation plays key role in inferring the underlying intent and identifying the actions linked to a particular event of interest. For example, an event could be benign (simply a coincidence) or malicious depending on the explanation and the context.

According to this perspective, one can think of the following levels of abstraction regarding different types of events.

- *Outliers*: Unusual data objects that are statistically rare but may or may not result from the same generative processes as other data objects.
- *Anomalies*: Data objects that have resulted from a distinct generative process that may or may not appear in the data as outliers.
- *Events of interest*: Anomalies that are caused by some type of behavior in the real world that is of concern for the application.

It is important to note that outliers are always defined with respect to some baseline population. Basically, it is a relative assessment of objects within a set. Further, an outlier must be described not only in terms of its likelihood, but also in terms an analyst can understand for determining whether that outlier is significant in a given context. Thus, the set of anomalies identified (A) specific to an application domain turns out to be a subset of the outliers discovered in the corresponding data (O), i.e. $A \subset O$. Likewise, determining the events of interest from the set of anomalies requires subjective assessment based on the domain knowledge.

At present, domain specific assessment of outliers can only be done manually due to practical limitations of the methods established so far. Adding automated assessment capabilities will enable systems to more accurately distinguish events of interest from other anomalies and enable human analysts to focus on follow-up investigations for those situations most demanding and worthy of their attention.

1.3.2 The Practical Requirement

As defined above, outliers represent valuable information about the evolving nature of the underlying physical phenomenon. They need to be seen as early indicators of the potential change in the behavior of the system. If the perceived change through the detection of outliers happens to be unfavorable (malicious), then one should keep a mitigation plan handy. It is for ensuring this preparedness by the system to cope up with the changes happening over time, outlier detection becomes an essential component of its functionality. It is a kind of self health check of the system and triggers necessary corrective action in time to avoid any plausible adverse effects of such a change. It is more so, when the system is designed to serve real life applications with low tolerance towards unintended effects.

Systems dealing with financial transactions, citizen health, and critical infrastructure need to perform continuous check of their status for ensuring graceful degradation in case of any inconsistent or emergency situations. Such situations can be predicted through early detection of outlier/anomalous behavior of the system employing necessary algorithmic schemes as per the nature of the end application. Therefore, exploring various computational aspects of the outlier detection problem becomes essential in order to determine a suitable method for a given application context.

1.4 Summary

A gentle introduction to the notion of outliers is provided in this chapter in the context of data mining. Outliers are fundamentally identified by their deviation with respect to normal data. Such data objects can be characterized in multiple ways, such as the number of objects involved (size), the different types of their nature (diversity), the specific actions they result in (role).

The significance of outliers is brought out in a theoretical sense as well as its requirement from a practical perspective. Basically, the foundation for progressing towards the detection of outliers in data is laid in this chapter.

1.5 Bibliographic Notes

The field of data mining has been growing rapidly due to its widespread use in several real life applications [5]. The concept of outliers in data is an abstract theory and its interpretation relies on the context. Among various definitions proposed for outliers, the one given in [6] is a popular one. The practical significance of the outlier phenomenon in several real life scenarios was nicely narrated in [4]. Though the deviating nature of objects makes them outliers at large, the issue of what constitutes a sufficient deviation is often a subjective judgment, as pointed out in [1]. An introduction to the notion of anomaly along with a review of various techniques for anomaly detection was provided in [4].

As brought out in [9], several real world applications are faced with the problem of identifying *complex rare events*, which are characterized by contextual combinations of specific actions. In this regard, a three-level abstraction was proposed on the types of events that one may encounter in real life. The key link between these levels happens to be the domain specific explanation regarding the events.

As explained in diversity-based characterization of outliers, word frequency in documents plays a major role for applications related to information retrieval [8]. In this direction, a novel method for identifying semantically deviating outlier documents was proposed recently [10]. Similarly, sparsity of data impacts the outcome of data clustering process, more so when the data consists of outlier objects [7]. One more topic referred to in this chapter is the support vector machines (SVMs), as described in [3].

References

1. Aggarwal, C.C.: Outlier Analysis. Spinger, New York, USA (2013)
2. Chandola, V., Banerjee, A., Kumar, V.: Anomaly detection: a survey. ACM Comput. Surv. **41**(3) (2009)
3. Cortes, C., Vapnik, V.N.: Support vector networks. Mach. Learn. **20**(3), 273–297 (1995)
4. Gladwell, M.: Outliers: The Story of Success. Allen Lane - Penguin Books, Great Britain (2008)
5. Han, J., Kamber, M., Pei, J.: Data Mining: Concepts and Techniques, 3rd edn. Morgan Kaufmann (2011)
6. Hawkins, D.: Identification of Outliers. Chapman and Hall, London (1980)
7. Jain, A.K.: Data clustering: 50 years beyond K-means. Pattern Recognit. Lett. **31**, 651–666 (2010)
8. Manning, C.D., Raghavan, P., Schutze, H.: Introduction to Information Retrieval. Cambridge University Press (2008)
9. Senator, T.E., Goldberg, H.G., Memory, A.: Distinguishing the unexplainable from the merely unusual: adding explanations to outliers to discover and detect significant complex rare events. In: KDD Workshop on Outlier Detection and Description. ACM, Chicago, IL, USA (2013)
10. Zhuang, H., Wang, C., Tao, F., Kaplan, L.M., Han, J.: Identifying semantically deviating outlier documents. In: Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, EMNLP, pp. 2748–2757. Copenhagen, Denmark (2017)

Chapter 2
Outlier Detection

Abstract This chapter deals with the task of detecting outliers in data from the data mining perspective. It suggests a formal approach for outlier detection highlighting various frequently encountered computational aspects connected with this task. An overview of this book with chapter-wise organization is also furnished here giving an idea of the coverage of the material on this research domain.

2.1 Introduction

Traditionally, the purpose of knowledge discovery and data mining process is to extract novel and valid patterns from the data. Such patterns are expected to be understandable and useful to the person handling the discovery task. They can be represented as association rules, classification rules, cluster descriptions, sequential patterns, time series, contingency tables, etc. They are expected to cover a majority of the objects in the data. On the other hand, outliers are the minority set of objects in the data that deviate significantly from the rest of the data. This deviation may be perceivable directly at times or may need some computational model to ascertain the same. Irrespective of the means of understanding the deviating behavior of data objects, their presence in the data has some serious practical implication. Therefore, outlier mining/detection aims at discovering such objects in data.

It is not possible to impose a crisp mathematical definition on outliers, as their detection is more of a subjective decision rather than a routine assessment. Accordingly, one should look for a capable detection method matching with the semantics of the specific application context. A simple method for determining outliers is to produce the *box plot* of the data graphically. A box plot depicts numerical data through their distribution in different quartiles, as shown in Fig. 2.1. Such a plot is also known as *box-and-whisker diagram*, as it can have lines extending vertically from the boxes (whiskers) indicating the possible variability outside the upper and lower quartiles.

The space between distinct parts of the box reflects the spread of values in the data. Through a visual examination of the data using the box plot, it is possible to

N. N. R Ranga Suri et al., *Outlier Detection: Techniques and Applications*,
Intelligent Systems Reference Library 155,
https://doi.org/10.1007/978-3-030-05127-3_2

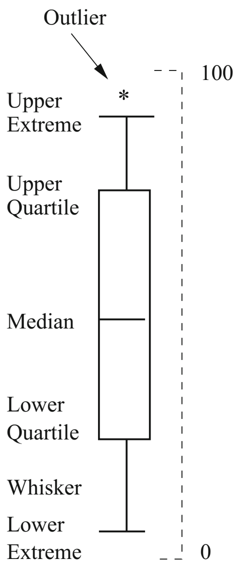


Fig. 2.1 A sample box plot indicating a possible outlier

determine the inter-quartile range (IQR) and other statistical estimators. Also, box plots can be drawn either horizontally or vertically depending on the representation needed.

In a box plot, the bottom and top portions correspond to first and third quartiles respectively, and the dividing band inside the box is the second quartile (the median). The ends of the whiskers may point to different possibilities among the following.

- The minimum and maximum values in the data
- One standard deviation above and below the mean of the data
- The 9th percentile and the 91st percentile
- The smallest value within $1.5IQR$ of the lower quartile, the highest value within $1.5IQR$ of the upper quartile.

Based on the above description of box plots, any data item not included within the value range covered by the whiskers is shown with a special symbol ('*') to signify it as a possible outlier. One such sample data point is shown in Fig. 2.1 at the top of the plot.

2.1.1 Problem Definition

Accurate and efficient detection of outliers present in a data set is ever challenging as there is no universal definition ascribed to outliers. As a result, it is up to the person dealing with this problem to consider a fitting definition and develop an acceptable method addressing the requirements. Given a multidimensional data set, an outlier may be deviating with respect to one of the given dimensions. It is also possible that a data object can be an outlier in the joint distribution of two or more attributes, not necessarily being an outlier when individual attributes are considered. For example, a person might have a low income but have large debt!

Given the above understanding, the outlier detection problem can be broadly defined as: given a set of data objects, find a specific number of objects that are considerably dissimilar, exceptional and inconsistent with respect to the remaining data. This definition basically captures distinct ways of an object displaying unusual characteristics.

Many data mining algorithms try to minimize the influence of outliers or eliminate them altogether. However, this could result in loss of important hidden information as per the well known fact that one person's noise could be another person's signal. Thus, the outliers themselves may be of particular interest, as in the case of fraud detection, where they may indicate some fraudulent activity. Besides fraud detection in financial applications, niche marketing and network anomaly detection are other prominent applications of outlier detection, making it an important data-mining task. Depending on the application domain, outlier detection has been variously referred to as novelty detection, chance discovery, exception mining, anomaly detection in networks, etc. A connected field of research is activity monitoring with the purpose of detecting illegal access. This task consists of monitoring an on-line data source in the search for unusual behavior.

Over the years, outlier detection has emerged as a significant data mining task due to the general perception that outliers represent evolving novel patterns in data. With widespread use of data mining techniques in diverse application domains, a slew of methods exist today for outlier detection catering to various application-specific requirements. Therefore, picking a competent method becomes a challenge in its own right as it decides the course of the detection process as well as the ultimate outcome.

At an abstract level, outlier detection methods can be classified into two main categories: *parametric* (distribution/model based) and *non-parametric* (distance based). A parametric or statistical approach assumes a distribution or probability model for the given data and identifies outliers with respect to this model using discordance tests. However, it is found that model-based detection of outliers has various practical limitations. To counter the limitations associated with the statistical methods, non-parametric methods such as distance-based ones are introduced. Thus, it is possible to avoid the expensive task of fitting a model for the data in this regard.

The surge in the number of research efforts towards outlier detection can be attributed primarily to its relevance in numerous real life applications ranging from simple relational data analysis to great scientific discoveries dealing with huge astronomical data. Other significant applications are connected with biological data analysis and fraud detection in financial data, etc.

Similarly, anomaly detection in networks has been an emerging data mining application involving various outlier detection techniques. Anomalies in a network signify irregularities in several real life applications such as extremely cross-disciplinary authors in an author-paper network, intrusions in a computer network, and several others. In addition to revealing suspicious, illegal and/or dangerous behavior, anomaly detection is useful for spotting rare events. Thus, anomaly detection in networks turns out to be a key data mining task in varied application contexts dealing with network data.

2.1.2 *Significance of Outlier Detection*

Outlier detection is interesting to many researchers as a data cleaning task for exploring the set of normal objects as the candidate set for frequent item mining and class profile building. Of late, it has been gaining more prominence as a potential data discovery task for anomaly detection, novelty detection, etc.

Prior research in this regard talks about the practical significance of the outlier phenomenon in real life scenarios. Similarly, the growing number of research works being published on this research topic is a candid indicator of the importance associated with this task in the field of data mining. In this regard, a few of the recent applications involving outlier detection methods are listed below.

- Identifying misinformation and misbehavior on the web
- Anomaly detection from system logs in complex computer systems
- Finding irregularities in electrical consumption on smart grids
- Modeling antisocial behavior in online discussion communities
- Characterizing a malignant tumor in medical images for diagnosis
- Predicting illegal insider trading in stock market data
- Anomaly detection in heterogeneous information networks
- Evaluating the potential of insider threats in organizations
- Detecting traffic anomalies in urban interstate systems
- Illegal reseller identification by Internet marketplaces like Amazon.

A vast majority of the methods developed for outlier detection have been designed to work with the data described using real valued attributes. However, almost all real world data sets contain a mixture of both nominal (categorical) and real valued attributes. The nominal attributes are typically ignored or incorrectly modeled by existing approaches. Non-availability of a well defined proximity metric on categorical data is other major concern. Therefore, developing efficient methods for outlier detection in categorical data has been a challenging task with several potential applications in real world data mining systems.

Outlier detection is typically viewed as a binary classification task. It may be divided into three main categories: supervised learning (when examples of both normal and outlier classes are made available), one-class classification (where examples of only the normal class are present) and unsupervised learning (when no labeled examples are available). The third category, dealing with unsupervised detection of outliers in data, is generally ill-posed. However, an unsupervised learning method like clustering comes handy to identify objects exhibiting deviating characteristics. Further, fraud detection and anomaly detection are important applications where unsupervised learning is more natural and popular. So, it looks more promising to develop methods based on unsupervised learning for detecting outliers in data.

Most of the latest work connected to outlier detection is due to the thrust emerging from various novel graph mining applications such as anomaly detection in network data. However, many of the existing anomaly detection approaches focus on link and node behavior anomalies. Only a few recent methods explore the network in search of subgraph anomalies of a specific type. Hence, it is required to develop a generic method discovering arbitrary anomalous subgraphs in any input graph. This is another significant research direction with potential applications in diverse fields.

There has been a tremendous growth in the presence of network data sets recording the interaction and/or transactions among a set of entities such as personal communications, on-line social network interactions, web traffic between servers and hosts, and router traffic among autonomous systems, etc. A characteristic feature of these activity networks is the time dependent change in their structure. These temporal dynamics are critical to model and predict the network changes over time. Although some prior work on the analysis of dynamic networks exist, they don't address the issue of uncovering the behavioral patterns of nodes in the graph and modeling how these patterns change over time.

On the other hand, it is observed that unusual behavior often propagates along the network during its evolution and persists in time. Hence, a naive adaptation of existing methods to dynamic networks would require searching in the combined exponential space of possible subgraphs and quadratic space of time intervals, which would be prohibitive on most real world network instances given their growing sizes. This emphasizes the need for developing competent methods for discovering anomalies/unusual behavior in dynamic networks.

2.2 Important Computational Aspects

Motivated by the plethora of applications requiring the outlier detection capability, some of the significant research gaps along with the associated challenges in this field of research are examined below. This exploration is aimed at highlighting the practical significance of various computational aspects that one may come across typically when dealing with the outlier detection problem. Thus, the idea here is to point out what one should be concerned with in order to develop efficient techniques for outlier detection. The subsequent chapters of this book delve on these computational aspects individually and also discuss some of the related algorithmic developments.

2.2.1 Impact of the Type of Data Attributes

The data one encounters with regard to outlier detection could be either *categorical* or *numerical*. Several algorithms that are popularly used on numerical data can't be used on categorical data. For example, the popular *k-means* clustering algorithm can't be used on categorical data. Also, k-means algorithm is not the right candidate for clustering high-dimensional data. Because, in the k-means algorithm a pattern is assigned to that cluster whose centroid is closest to the pattern.

As stated above, a majority of the outlier detection methods mainly deal with numerical data. However, the data pertaining to several real life applications tend to be categorical in nature with high dimensionality. Therefore, it is necessary to devise acceptable techniques to process data described using categorical attributes. In the context of outlier detection, the effectiveness of the detection algorithm strongly depends on the interpretation or one's perception about outliers in the sense of categorical attributes. So, it becomes imperative to have an intuitive definition for characterizing outliers and develop the detection algorithm accordingly. When dealing with categorical data, one can try to find out objects that are deviating with respect to a single attribute or in the joint distribution of multiple attributes.

2.2.2 Unavailability of Labeled Training Data

It is known that any machine learning activity primarily belongs to one of the two learning modes: *supervised* and *unsupervised*. The exact nature of the learning task is decided based on the availability of labeled data. When the data is provided with necessary class labels for the training purpose, the supervised learning techniques are applicable. On the other hand, the unsupervised techniques do not require labels and they try to distinguish between the objects belonging to distinct groups (classes) based on their intrinsic characteristics.

As discussed in the previous section, outlier detection is generally considered as an unsupervised learning task due to lack of awareness on the kind of outliers present in the data. In such a situation, data clustering turns out to be the obvious choice to explore the inherent structure of the data. More specifically, clustering algorithms belonging to the *k-means family* are preferred due to their computational efficiency. Such a clustering structure is expected to show clear demarcation between outliers and normal objects in its grouping process. Additionally, clustering-based methods can detect data objects that deviate in the joint distribution of two or more attributes. This kind of scenario is typically noticed when dealing with categorical data.

2.2.3 Crisp Labeling Versus Relative Ranking of Outliers

A detection method aimed at finding out a certain category of objects in the data can produce its result in two possible ways. It can either label the objects as crisply belonging to the desired category or arrange the objects based on their relative belongingness to the desired category. Of course, the latter option depends on the ability of the detection method in computing the requisite measure as a continuous valued one. Applying a satisfactory threshold on this measure of belongingness, one can always produce crisp result if required.

In the context of outlier detection, it is more meaningful to rank the data objects based on their degree of deviation rather than making a binary decision on whether or not an object is an outlier. Consequently, one can produce a ranking of the data objects based on their relative deviation captured by means of the clustering structure. Alternatively, ranking of objects can be attempted based on their outlier score determined by the frequencies of attribute values. In case of categorical data, this approach seems more natural for identifying the objects that deviate with respect to one or more attributes. Also, one can consider generating multiple rankings of the objects based on different deviation measures and finally fuse all of them to produce a *likely set* of outliers in the data.

2.2.4 Exploring Outliers in High Dimensional Spaces

Data pertaining to several real life applications, such as the biological data and market-basket data, are comprised of prohibitively large number of features. One naturally looks for a small subset of features that preserves the properties of the data without loss of any important information. This calls for the process of *feature selection* so as to reduce the impact of the high dimensionality on the learning task such as data clustering. It is important to note that clustering algorithms depend on proximity between pairs of patterns in order to divide them into groups. When a pattern X is represented as a high-dimensional vector, it is difficult to distinguish between its nearest and farthest neighbors due to the *curse of dimensionality*. So dimensionality reduction is essential before applying several of the clustering algorithms.

From the outlier detection point of view, it is experimentally found that one can develop optimal algorithms by looking for outliers in subspaces of the data. This is because the objects deviating with respect to a subset of features get buried in the high dimensional spaces thereby hindering their detection process. Feature selection is thus aimed at curbing the undesirable effects of data dimensionality on the outlier detection process.

Employing feature selection as a means for dimensionality reduction requires formulation of acceptable measures for assessing the relevance and redundancy of an attribute/feature towards the outlier detection task. As it is learnt that the unsupervised learning paradigm is better suited for outlier detection, one needs to follow a similar approach for feature selection as well. Prior work on dealing with categorical data suggests to employ information theoretic measures defined based on the concept of discrete random variable. It basically indicates that one can consider Mutual Information (MI) based characterization of the features for filtering them objectively. Ultimately, the low dimensional representation is expected to produce superior outlier detection performance over the full dimensional representation.

2.2.5 Dealing with Uncertainty in the Learning Process

Unsupervised learning tasks employing k-means algorithm typically produce non-overlapping, clearly separated ('crisp') clusters with bivalent memberships: an object either belongs to or doesn't belong to a cluster. However, many real life applications are characterized by representations involving overlapping clusters. Mining outliers in data is one such application, as it has to deal with uncertainty regarding the membership of outlier objects to one of the normal groups of objects. In this context, a soft computing approach based on *rough sets* happens to be a better choice.

A rough clustering algorithm works by incorporating the lower and upper approximation properties of rough sets. In this process, objects assigned to the lower approximation of a cluster belong certainly to that cluster. However, cluster membership of the objects assigned to the upper approximation is uncertain, hence they belong to at least two clusters. Thus, the uncertainty regarding the crisp membership of objects can be resolved by using rough clustering principles. In a theoretical sense, the members of upper approximation of a cluster tend to have high chance of getting detected as outliers.

Given the above discussion, it is important to look at both hard and soft clustering algorithms for outlier detection. For example, the hard k-means clustering algorithm and its soft versions like rough k-means (RKM) algorithm are worth exploring. The RKM algorithm basically involves two crucial computational steps: (i) determining the object assignment to various clusters, and (ii) modified centroid computation. Similarly, other soft computing approaches based on *fuzzy sets* and *genetic algorithms* are also attractive choices for carrying out outlier detection.

2.2.6 Detecting Anomalies in Network/Graph Data

Data connected with a number of real life applications is networked in nature, such as computer communication networks, social networks of friends, the citation networks of documents, hyper-linked networks of web pages. Exploring rare connections and emerging scenarios in such data is important from a practical perspective. Such an analysis is intended to unearth possible anomalous activities by the entities in the network. In this connection, anomaly/outlier detection in graphs turns out to be an important pattern discovery activity. The crux of this process lies in the way the anomalous behavior of the network entities is characterized as a graph pattern and also the specific method required for detecting such patterns. Though building necessary algorithmic methods for this purpose looks non-trivial from graph mining point of view, one can extend the concepts and techniques developed for outlier detection in multi-dimensional vector data to the case of network data. Thus, this activity assumes significance as an emerging application area for exploiting various established outlier detection techniques.

A critical look at the related literature reveals that a slew of methods deal with the detection of anomalous nodes and/or anomalous edges. Only a few methods look for anomalous subgraphs, that too based on some assumptions and constraints on the detection logic. Such methods lack generality and universal applicability by their design and hence leave scope for more innovation in this regard. Therefore, it is important to have a generic method for detecting arbitrary subgraphs of the input graph as possible anomalies. In this context, it is noted that detecting communities in networks helps in better understanding the underlying connection semantics of large networks. Among various methods available for community detection, the Non-negative Matrix Factorization (NMF) technique is an interesting one due to its powerful interpretability and close relationship to the clustering methods. Communities or subgraphs produced by this technique are expected to be better candidates for detecting various types of anomalies in graph data.

2.2.7 Characterizing Anomalies in Dynamic Networks

In today's world, people are actively engaging themselves over the on-line *social networks* as means of communication regarding their recent achievements/activities, and also for expressing their opinions on some trending topics of interest. A key property associated with such networks is that they evolve over time due to addition/deletion of individuals in the network. Likewise, many communication networks are dynamic in nature with time varying characteristics and other observed phenomena. It becomes imperative to handle such networks for objective analysis and discover various evolving graph patterns possibly detecting various types of anomalies present.

Thus, the aim here is to characterize the time varying nature of the anomalies present in dynamic networks that reflect the changing real life scenarios. An important activity in this regard is to uncover the temporal behavioral patterns of the nodes in a graph. This requires exploring various kinds of temporal anomalies that could be seen with dynamic networks and also come up with a formal categorization of them with the required semantic foundation. Also, one should validate these theoretical ideas with empirical findings using qualified benchmark network data sets.

2.3 Organization of this Book

This book essentially suggests some elegant methods for various computational aspects concerned with the outlier detection task in the field of data mining. Accordingly, it comprises nine chapters in addition to introduction to outliers and outlier detection, as outlined below.

- Chapter 3—*Research Issues in Outlier Detection*: This chapter gives an overview of the outlier detection problem. A taxonomy of various methods available for outlier detection is presented giving an abstract view of the solution space. Then, it enlists various research issues identified in this study along with some references to the relevant work. It also talks about some recent trends in this regard.

- Chapter 4—*Computational Preliminaries*: This chapter is meant for providing an outline of various definitions, methodologies and best practices related to outlier detection, specific to the work reported in this book. To start with, it gives a quick view of various proximity measures defined over categorical data along with the mathematical notation followed in this work. Then, a summary of the basic concepts connected to matrix algebra is provided as relevant for the outlier discovery task. Subsequently, a few fundamental ideas related to Information Theory are furnished with emphasis on the mutual information measure defined on discrete random variables. Next, the pre-processing steps to prepare both multi dimensional data sets and graph/network data sets for outlier detection are furnished. Finally, various measures for characterizing the performance of a binary classifier are covered, with emphasis on outlier detector.

- Chapter 5—*Outliers in Categorical Data*: This chapter deals with the development of algorithms for efficient detection of outliers in categorical data. At the beginning, it introduces the outlier detection problem in the categorical data scenario highlighting various computational challenges posed by the same. As outlier detection is best attempted using clustering based techniques, various algorithms for clustering categorical data are discussed. Then, the computational model developed based on ranking schemes is explained with necessary theoretical foundations. Subsequently, some of the methods for outlier detection in categorical data are discussed. Finally, an experimental understanding of these methods on benchmark data sets is provided along with sensitivity study of the ranking-based algorithm.

- Chapter 6—*Outliers in High Dimensional Data*: This chapter deliberates on the development of an unsupervised feature selection algorithm addressing the dimensionality aspect of the data. It first provides a brief introduction to the issues relating to outlier detection in high dimensional data. It then talks about various methods for feature selection using Mutual Information (MI) based measures. Then, it covers the measures developed for assessing the relevance and redundancy of categorical features and a method for feature selection employing these measures aimed at outlier detection. Finally, this chapter demonstrates the impact of feature selection on the outlier detection performance over some benchmark data sets.

- Chapter 7—*Outlier Detection using Rough Sets*: This chapter points out the issue of uncertainty in the outlier detection process by developing a possible soft computing approach based on rough sets. First, it introduces the concept of rough sets as per the set theoretic formulation. Then, it describes the rough k-means algorithm developed based on the rough clustering principles. Subsequently, a few rough-sets based algorithms for clustering categorical data are put forward. This includes the rough k-modes method, along with an experimentation over benchmark categorical data sets.

- Chapter 8—*Mining Anomalies in Graph Data*: This chapter discusses useful applications of the outlier detection principles towards graph mining problems, such as anomaly detection in network/graph data. The necessity for analyzing network data is brought out at the beginning of this chapter. Then, the basics of NMF technique are presented signifying its relevance for exploring communities present in a network/graph. This chapter highlights the role of community detection in network data as a more effective way towards anomaly detection. It describes an algorithm for finding subgraph anomalies that are arbitrary subgraphs of the input graph. Details of an experimental study using this algorithm along with the results obtained are furnished towards the end.

- Chapter 9—*Analyzing Dynamic Networks*: This chapter highlights the need for analyzing dynamic networks and suggests that understanding community evolution is a means of achieving this. It then talks about modeling of dynamic networks using Statistical and Game theoretic perspectives as the basis. Subsequently, it presents some recent applications of dynamic network analysis techniques in varied application contexts.

- Chapter 10—*Detecting Anomalies in Dynamic Networks*: This chapter deals with the problem of anomaly detection in evolving networks by emphasizing its importance in several contemporary applications. It describes a method of characterizing the outlier dynamics in evolving networks/graphs along with a categorization of them. An experimental study of this method is furnished bringing out the empirical findings. Finally, this chapter gives details of a few recent applications of dynamic network anomaly detection.

- Chapter 11—*Summary and Future Work*: A summary of the important aspects discussed in this book is presented at the beginning of this chapter. It then brings out a discussion on some key aspects of the outlier discovery task. Finally, it suggests some possible research directions for future work connected with the specific research problems described across various chapters of this book.

2.4 Summary

A few insights on the outlier detection problem from a theoretical perspective are presented in this chapter. The significance of dealing with this problem for various practical applications is highlighted giving a list of active real life examples requiring outlier detection capability. Then, some of the important computational aspects associated with outlier detection task are discussed in brief. Finally, chapter-wise organization of this book is produced for giving an overview of the material covered within.

2.5 Bibliographic Notes

Fundamental principles of data mining were presented in [38] giving details of various tasks involved. A formal definition for outlier detection problem was given in [15] based on the deviating nature of the outliers with respect to rest of the data. This task has gained significance in the data mining context as one person's noise could be another person's signal [22]. A simple technique for identifying outliers in univariate data is to use box-plots [18] with appropriate threshold values. Depending on the target application, outlier detection finds applicability in various forms such as novelty detection [27], chance discovery [28], exception mining [37], anomaly detection in networks [8, 29], search for unusual behavior [13], etc. A comprehensive discussion on anomalies, novelty detection, and one-class classification was put forward in [5]. Similarly, a support vectors based method for novelty detection was discussed in [35]. Likewise, a method was proposed for outlier detection with globally optimal exemplar-based Gaussian Mixture Model (GMM) [42].

Outlier detection finds relevance in numerous real life applications such as scientific discoveries dealing with huge astronomical data [10]. A methodology for identifying misinformation and misbehavior on the web was proposed [24]. In a related effort, a technique for modeling the antisocial behavior in online discussion communities was brought out [11]. Similarly, a query-based method for outlier detection was proposed to deal with heterogeneous information networks [23].

Some of the research efforts [2, 4, 9, 20, 40, 41] reflect the importance associated with the outlier detection problem in the field of data mining. For example, anomaly detection in network data is useful for spotting rare events [3, 7, 31], hyper-linked networks of web pages [21], etc. Thus, outlier detection gained prominence as a potential data discovery task [8, 17].

In high-dimensional spaces, it is difficult to distinguish between the nearest and the farthest neighbors of a data object [6]. Therefore, design of optimal algorithms by looking for outliers in subspaces of the data [1] should be considered. In the context of categorical data, there exist information theoretic methods for carrying out feature selection [12, 32]. As the nature of outlier objects is not known a priori, developing methods based on unsupervised learning is more practical for outlier detection [16]. Detection methods based on data clustering [19, 34] are capable of separating out the outliers from the normal data. Also, it is more meaningful to rank the data objects based on their degree of deviation rather than making a binary decision on whether or not an object is an outlier [30].

There prevails uncertainty in the outlier detection process through clustering. This is because several real life applications are characterized by representations involving overlapping clusters [26]. In such a situation, soft computing approaches are the natural choices due to their popularity in different applications. Recently, a soft computing approach based on fuzzy sets is designed addressing the event-triggered fault detection problem for discrete time fuzzy systems in network environments [36]. Likewise, there exists a genetic algorithm that performs simultaneous outlier detection and feature selection on the data [39]. Thus, it is possible to find many such novel data mining methods employing different soft computing techniques to handle the uncertainty present in the outlier detection process in a pragmatic manner.

Detecting communities in networks helps in better understanding the underlying characteristics of large networks [14]. Due to some computational advantages, the NMF technique [25] happens to be an attractive choice for detecting the communities present in a network. In case of dynamic network analysis, a naive adaptation of existing methods would be prohibitive computationally on most real world network instances [29]. Therefore, exploring the behavioral patterns of the nodes in a network/graph and modeling how these patterns change over time is the key for effective analysis [33].

References

1. Aggarwal, C.C., Yu, P.S.: Outlier detection for high dimensional data. In: ACM SIGMOD International Conference on Management of Data, pp. 37–46. Santa Barbara, USA (2001)
2. Aggarwal, C.C.: Outlier Analysis. Spinger, New York, USA (2013)
3. Akoglu, L., McGlohon, M., Faloutsos, C.: Oddball: spotting anomalies in weighted graphs. In: 14th Pacific-Asia conference on Advances in Knowledge Discovery and Data Mining (PAKDD), Hyderabad, India, pp. 410–421 (2010)
4. Albanese, A., Pal, S.K., Petrosino, A.: Rough sets, kernel set, and spatio-temporal outlier detection. IEEE Trans. Knowl. Data Eng. **26**(1), 194–207 (2014)
5. Bartkowiak, A.M.: Anomaly, novelty, one-class classification: a comprehensive introduction. Int. J. Comput. Inf. Syst. Ind. Manag. Appl. **3**, 61–71 (2011)
6. Beyer, K.S., Goldstein, J., Ramakrishnan, R., Shaft, U.: When is nearest neighbor meaningful? In: 7th International Conference on Database Theory, ICDT. Lecture Notes in Computer Science, vol. 1540, pp. 217–235. Springer, Jerusalem, Israel (1999)
7. Chakrabarti, D.: Autopart: Parameter-free graph partitioning and outlier detection. In: PKDD, pp. 112–124 (2004)
8. Chandola, V., Banerjee, A., Kumar, V.: Anomaly detection: a survey. ACM Comput. Surv. **41**(3) (2009)
9. Chandola, V., Banerjee, A., Kumar, V.: Anomaly detection for discrete sequences: a survey. IEEE Trans. Knowl. Data Eng. (TKDE) **24**(5), 823–839 (2012)
10. Chaudhary, A., Szalay, A.S., Szalay, E.S., Moore, A.W.: Very fast outlier detection in large multidimensional data sets. In: ACM SIGMOD Workshop in Research Issues in Data Mining and Knowledge Discovery, pp. 45–52 (2002)
11. Cheng, J., Danescu-Niculescu-Mizil, C., Leskovec, J.: Antisocial behavior in online discussion communities. In: Proceedings of the Ninth International Conference on Web and Social Media ICWSM, Oxford, UK, pp. 61–70 (2015)
12. Estevez, P.A., Tesmer, M., Perez, C.A., Zurada, J.M.: Normalized mutual information feature selection. IEEE Trans. Neural Netw. **20**(2), 189–201 (2009)
13. Fawcett, T., Provost, F.: Activity monitoring: noticing interesting changes in behavior. In: 5th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD), pp. 53–62 (1999)

14. Girvan, M., Newman, M.E.J.: Community structure in social and biological networks. PNAS **99**(12), 7821–7826 (2002)
15. Han, J., Kamber, M., Pei, J.: Data Mining: Concepts and Techniques, 3rd edn. Morgan Kaufmann (2011)
16. Hido, S., Tsuboi, Y., Kashima, H., Sugiyama, M., Kanamori, T.: Statistical outlier detection using direct density ratio estimation. Knowl. Inf. Syst. **26**(2), 309–336 (2011)
17. Hodge, V.J., Austin, J.: A survey of outlier detection methodologies. Artif. Intell. Rev. **22**, 85–126 (2004)
18. Hubert, E.V.M.: An adjusted boxplot for skewed distributions. Comput. Stat. Data Anal. **52**, 5186–5201 (2008)
19. Jain, A.K., Murty, M.N., Flynn, P.J.: Data clustering: a review. ACM Comput. Surv. **31**(3), 264–323 (1999)
20. Keller, F., Muller, E., Bohm, K.: Hics: High contrast subspaces for density-based outlier ranking. In: 28th International Conference on Data Engineering (ICDE), pp. 1037–1048. IEEE (2012)
21. Kim, M., Leskovec, J.: Latent multi-group memebership graph model. In: 29th International Conference on Machine Learning (ICML), Edinburgh, Scotland, UK (2012)
22. Knorr, E., Ng, R., Tucakov, V.: Distance-based outliers: algorithms and applications. VLDB J. Very Large Databases **8**(3–4), 237–253 (2000)
23. Kuck, J., Zhuang, H., Yan, X., Cam, H., Han, J.: Query-based outlier detection in heterogeneous information networks. In: Proceedings of the 18th International Conference on Extending Database Technology, EDBT, Brussels, Belgium, pp. 325–336 (2015)
24. Kumar, S., Jiang, M., Jung, T., Luo, R.J., Leskovec, J.: MIS2: misinformation and misbehavior mining on the web. In: Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining, WSDM, Marina Del Rey, CA, USA, pp. 799–800 (2018)
25. Lee, D.D., Seung, H.S.: Learning the parts of objects by non-negative matrix factorization. Nature (1999)
26. Lingras, P., Peters, G.: Applying rough set concepts to clustering. In: Rough Sets: Selected Methods and Applications in Management and Engineering, pp. 23–38. Springer Verlag, London (2012)
27. Markou, M., Singh, S.: Novelty detection: a review, Part 1: statistical approaches. Signal Process. **83**(12), 2481–2497 (2003)
28. McBurney, P., Ohsawa, Y.: Chance Discovery. Springer (2003)
29. Mongiovi, M., Bogdanov, P., Ranca, R., Singh, A.K., Papalexakis, E.E., Faloutsos, C.: Netspot: spotting significant anomalous regions on dynamic networks. In: SDM, Austin, Texas (2013)
30. Muller, E., Assent, I., Steinhausen, U., Seidl, T.: Outrank: ranking outliers in high dimensional data. In: IEEE ICDE Workshop, Cancun, Mexico, pp. 600–603 (2008)
31. Noble, C.C., Cook, D.J.: Graph-based anomaly detection. In: SIGKDD, Washington, DC, USA, pp. 631–636 (2003)
32. Peng, H., Long, F., Ding, C.: Feature selection based on mutual information: criteria of max-dependency, max-relevance, and min-redundancy. IEEE Trans. Pattern Anal. Mach. Intell. **27**(8), 1226–1238 (2005)
33. Rossi, R.A., Neville, J., Gallagher, B., Henderson, K.: Modeling dynamic behavior in large evolving graphs. In: WSDM, Rome, Italy (2013)
34. Sato-Ilic, M., Jain, L.C.: Asymmetric clustering based on self-similarity. In: 3rd International Conference on Intelligent Information Hiding and Multimedia Signal Processing, pp. 361–364 (2007)
35. Scholkpof, B., Williamson, R., Smola, A., Taylor, J.S., Platt, J.: Support vector method for novelty detection. In: Advances in Neural Information Processing Systems (NIPS), pp. 582–588. MIT Press (1999)
36. Su, X., Xia, F., Wu, L., Chen, C.L.P.: Event-triggered fault detector and controller coordinated design of fuzzy systems. IEEE Tran. Fuzzy Syst. (2017)
37. Suzuki, E., Zytkow, J.: Unified algorithm for undirected discovery of exception rules. In: PKDD, pp. 169–180 (2000)
38. Tan, P.N., Steinbach, M., Kumar, V.: Introduction to Data Mining. Addison-Wesley (2006)

39. Tolvi, J.: Genetic algorithms for outlier detection and variable selection in linear regression models. Soft Comput. **8**(8), 527–533 (2004)
40. Wu, Q., Ma, S.: Detecting outliers in sliding window over categorical data streams. In: 8th International Conference on Fuzzy Systems and Knowledge Discovery (FSKD), pp. 1663–1667. IEEE (2011)
41. Wu, S., Wang, S.: Information-theoretic outlier detection for large-scale categorical data. IEEE Trans. Knowl. Data Eng. (TKDE) **25**(3), 589–602 (2013)
42. Yang, X., Latecki, L.J., Pokrajac, D.: Outlier detection with globally optimal exemplar-based GMM. In: SIAM International Conference on Data Mining (SDM), pp. 145–154 (2008)

Chapter 3
Research Issues in Outlier Detection

Abstract This chapter provides an overview of the outlier detection problem and brings out various research issues connected with this problem. It presents a detailed survey of the available literature on this problem with respect to the research issues identified. Basically, this chapter sets the direction for the material presented in the rest of the chapters of this book.

3.1 Introduction

Recent developments in the field of data mining have led to the outlier detection process mature as one of the important tasks in data analysis. Due to its significance in various contemporary applications, outlier detection (also known as outlier mining) continues to be an active research problem.

As presented in the previous chapters, outliers are data objects that are significantly different from the rest of the data. Outlier detection or outlier mining refers to the process of identifying such rare objects in a given data set. Although rare objects are known to be fewer in number, their significance is high compared to other objects, making their detection an important task. The general requirement of this task is to identify and remove the contaminating effect of the outlying objects in the data and as such to purify the data for further processing. However, with the evolving perspective on the role played by outliers, it turned out to be an interesting discovery task.

The presence of an outlier in a data set shows up itself explicitly in some form or the other when the data is visualized in a suitable manner. For example, data objects P and Q in Fig. 3.1a are two deviating objects as evident from a visual examination of the data depicted. However, in the case of objects P_1 and P_2 shown in Fig. 3.1b, identifying them as outliers requires additional effort due to their position in relation to the rest of the data. Therefore, depending on the relative position and grouping of outliers in data, specialized techniques may be required for their characterization leading to their detection.

© Springer Nature Switzerland AG 2019
N. N. R Ranga Suri et al., *Outlier Detection: Techniques and Applications*,
Intelligent Systems Reference Library 155,
https://doi.org/10.1007/978-3-030-05127-3_3

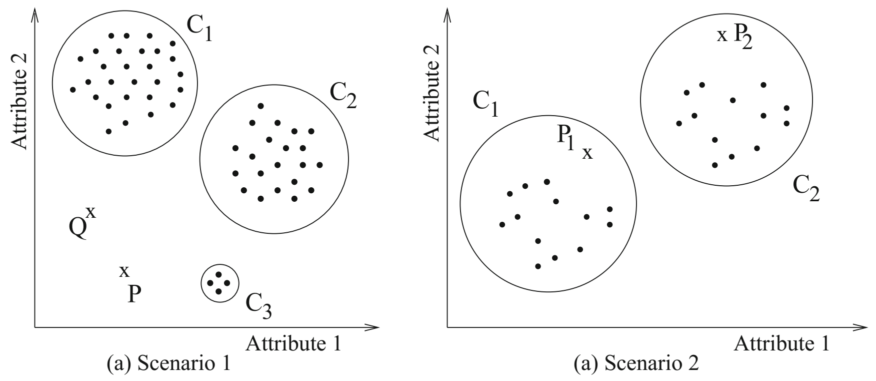


Fig. 3.1 Outliers in data: two sample scenarios

A number of algorithmic techniques exist addressing the outlier detection problem. A chronological view of them indicates that the initial work in this regard is based on the statistical methods. According to these methods, outliers are observations that appear to be statistically inconsistent with the remainder of the data. Typically, statistical methods are parametric in nature and they assume a known underlying distribution or statistical model of the data. As a result, outliers turn out to be the objects that have low probability of belonging to the underlying model. However, these approaches are found to be ineffective even for moderately high dimensional data due to the difficulty in finding a right model. Thus, the statistical methods are known to be infeasible from a practical perspective.

Basically, there are two essential considerations for selecting an acceptable methodology for outlier detection: (i) selecting an algorithm that can accurately model the data distribution and (ii) selecting a suitable neighborhood of interest for characterizing an outlier. Once these aspects are addressed in a right manner, the actual detection process can be carried out seamlessly.

A taxonomy of various outlier detection methods divides them two broad groups: parametric and non-parametric techniques. It focuses mainly on statistical methods for univariate and multivariate outlier detection. Looking from a different perspective, these methods can be divided into following three categories based on the kind of data modeling techniques employed by them.

- *Statistical methods*: These methods are the earliest ones used for outlier detection. Many of the statistical techniques are based on single dimensional or at best univariate model of the data, whereas, most of the recent applications require modeling multi-dimensional data for achieving outlier detection capability.
- *Neural network-based methods*: These methods are generally non-parametric and generalize well to unseen patterns and are capable of learning complex class boundaries. However, the data set has to be traversed numerous times to allow the network to converge and model the data correctly.

- *Machine learning-based methods*: Most statistical and neural methods require computation of vector distances over the data and have no mechanism for processing categorical data. Decision trees are used to detect outliers in categorical data as they do not require any prior knowledge of the data. Another machine learning technique exploited for outlier detection is rule-based systems which are similar to decision trees.

In addition to the above categories of techniques, some methods employ hybrid techniques also for achieving the outlier detection capability.

Though various such studies on outlier detection talk about some of the important aspects of the outlier detection process, it is pertinent to have a comprehensive view of various research issues of interest in this connection. One may notice that these issues have the potential to drive the current research work in this area. Thus, it is important to understand these issues in detail and develop techniques addressing the challenges posed by each one of them.

3.2 Important Research Issues

Outlier detection has emerged as an important data analysis task for various applications in diverse domains. As a result, choosing an appropriate detection method meeting the application requirements turns out to be a non-trivial task. Also, various applications deal with varying types of data in terms of the nature, dimensionality and size. Typically, the requirements of the detection method are driven by the nature of the end application. Thus, researchers dealing with these applications are often faced with new technological challenges in finding the right solutions.

In view of the above, a thorough exploration of the problem space identifies the following *research issues* as the significant ones concerning to outlier detection. These issues correspond to three key directions of exploration such as algorithm related, data dependent and application specific aspects of outlier detection, as listed below.

1. Algorithm related issues
   - Method of detection
   - Learning scheme involved
2. Data dependent issues
   - Type of data attributes
   - Size and dimensionality
3. Application specific issues
   - Nature of the application
   - Mode of analysis

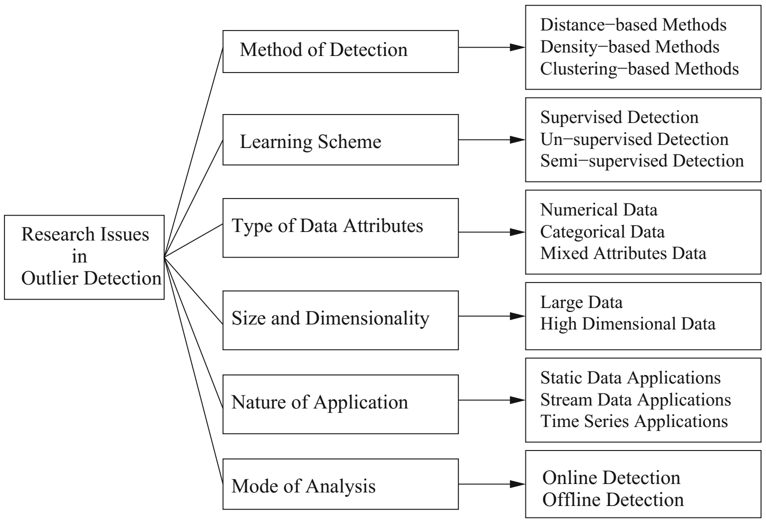


Fig. 3.2 Research issues in outlier detection

One can visualize a number of possible research directions corresponding to each one of the issues listed above. A sample set of them, though not exhaustive, are shown in Fig. 3.2 for brevity. A detailed discussion on these research issues is presented in subsequent sections.

Not withstanding the issues listed above, one may come across more such issues in future as this research domain evolves and the variety of applications widens. At present, this exploration is meant to give a reasonable understanding of the problem to enable further research in this area.

3.2.1 Method of Detection

A number of algorithmic methods are devised for detecting outliers in data over the time. However, no single method stands for universal applicability due to variability in the application context. Hence, depending on the specific application requirement a suitable detection strategy is considered among the available options.

As per the taxonomy mentioned in the previous section, outlier detection techniques can be broadly divided into parametric and non-parametric varieties, as shown in Fig. 3.3. The statistics-based methods that assume a model for the data belong to the parametric variety. Typically, one has to model the data using a statistical distribution, and data objects are determined to be outliers depending on how they appear in relation to the postulated model. On the other hand, most of the non-parametric methods rely on a well-defined notion of distance to measure the separation between a pair of data objects. The non-parametric variety includes distance-based, density-based,

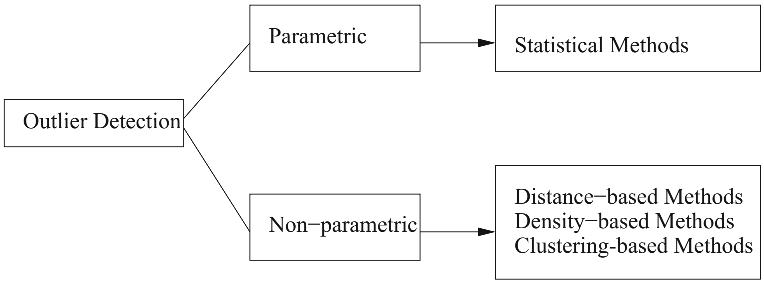


Fig. 3.3 A taxonomy of the outlier detection methods

and clustering-based methods. As already understood, statistical methods have practical limitations due to which their usage is restricted to certain instances, whereas, much of the current research is focused towards non-parametric methods driven by rapid developments taking place in the area of data mining.

3.2.1.1 Distance-Based Methods

In order to overcome the problems associated with statistical methods, distance-based methods are developed based on data mining principles. According to an early method of this variety, a data object is treated as outlier if no more than k objects in the data set are at a distance of d or less from that object. This task is achieved using a simple nested-loop method requiring $O(rN^2)$ computations in the worst case. Here, r is the dimension of the data and N is the number of objects in the data set. This way of finding outliers does not need prior knowledge of data distribution that the statistical methods do. However, this algorithm requires the user to specify a distance threshold d and a value for the parameter k to determine outliers.

A number of improvements over the above method are available making use of different criteria for measuring the distance in order to qualify an object as outlier. The objective in this regard is to reduce the cost of nested-loop computation using applicable simplification as may be the case.

1. *Distance to the* kth *nearest neighbor*: Intuitively, this distance is indicative of how much of an outlier a data object is. For example, referring to Fig. 3.1a, object P is a much stronger outlier than Q. Essentially, one can rank each object based on its distance to its kth nearest neighbor, and then consider a desired number of top ranked outliers from this ranked list.
2. *Average distance to the k-nearest-neighbors*: Similar to the above refinement, one can produce a ranked list of objects using this criteria for distance measurement and find outliers of desired number.
3. *Sum of the distances to its k-nearest-neighbors*: The sum computed for each object is assigned as its weight. Data objects with high weights are considered as outliers.

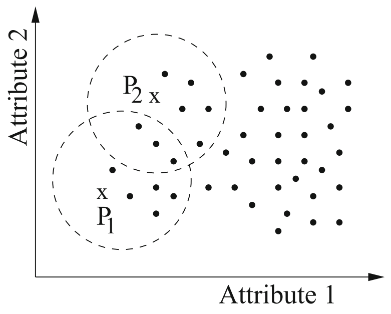


Fig. 3.4 Differences among various distance-based outlier definitions

All the above techniques are based on identifying the nearest neighbors of a data object so as to determine the objects with only a few neighbors as outliers. Though the underlying principle is the same, the set of outliers generated using these techniques could be considerably distinct. This observation is depicted in Fig. 3.4, in which both the objects P_1 and P_2 have 8 nearest neighbors each shown inside the dashed circles. However, they may not get the same outlier rank due to the differences in the detection strategies.

The advantages of distance-based methods are that no explicit distribution needs to be defined to determine unusualness, and that they can be applied to any feature space for which a distance measure can be defined. Thus, distance-based methods find applicability in several contexts as the de-facto methods for outlier detection. However, these methods may not exhibit optimal results when the input data has varying local densities.

3.2.1.2 Density-Based Methods

Density-based approaches for outlier detection are developed to deal with the local density problems of the distance-based methods. In a typical density-based framework, there are two parameters that define the notion of *density* of an object.

- A parameter *MinPts* specifying the minimum number of objects, and
- A parameter specifying the volume under consideration.

These two parameters determine a density threshold for the algorithm to operate. To detect density-based outliers, it is necessary to compare the densities of different sets of objects in the data.

Local Outlier Factor (LOF) is a density-based outlier detection approach employing Local Reachability Density (LRD) of data objects as the density measure. The LRD of an object p, denoted as $LRD_{MinPts}(p)$, is given by the inverse of the average reachability distance based on the *MinPts*-nearest neighbors of that object. Thus, the LOF of an object p is computed as the average of the ratio of the LRD of its nearest neighbors ($N_{MinPts}(p)$) and that of the object, as given below.

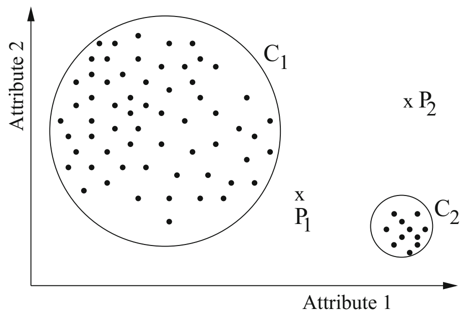


Fig. 3.5 Illustration of density-based outliers

$$LOF_{MinPts}(p) = \frac{\sum_{q \in N_{MinPts}(p)} \frac{LRD_{MinPts}(q)}{LRD_{MinPts}(p)}}{|N_{MinPts}(p)|} \tag{3.1}$$

LOF method determines the likelihood of a data object being a local outlier. Though LOF method does not suffer from the local density problem, selecting *MinPts* for determining the nearest neighbors of an object is non-trivial.

Fig. 3.5 gives an example scenario involving some density-based outliers. Using the nearest neighbor based methods, object P_1 may not be detected as outlier as it has some objects in its proximity, while P_2 gets detected as outlier as all its neighbors are far away. However, the LOF-based detection finds both P_1 and P_2 as outliers as both of them get high LOF values compared to rest of the objects.

The computation of LOF values for all the objects in a data set requires a large number of k-nearest-neighbors searches and can be computationally expensive. To improve upon this aspect, many variants of the LOF method are available with certain modifications to the object density characterization.

1. *Using micro-clusters*: A micro-cluster $MC(n, c, r)$ is a summarized representation of a group of n data objects which are likely to belong to the same cluster. Here, c is the mean/center and r is the radius of the cluster.
2. *Multi-granularity DEviation Factor (MDEF)*: Any data object whose MDEF value deviates much from the local averages can be considered as outlier. Let the r-neighborhood of an object P_i be the set of objects within distance r of P_i. Then, the MDEF at radius r of an object P_i is the relative deviation of its local neighborhood density from the average local neighborhood density in its r-neighborhood. Thus, an object whose neighborhood density matches the average local neighborhood density will have an MDEF of 0. In contrast, outliers will have MDEFs far from 0. The MDEF measure copes with local density variations in the feature space and can detect both isolated outliers as well as outlying clusters, such as cluster C_2 shown in Fig. 3.5.
3. *Local distribution based*: This approach is also meant for finding local outliers from the local distribution of the data of interest. One can extract three local features namely local-average-distance, local-density, and local-asymmetry-degree.

Local outliers are detected from the viewpoint of local distribution characterized by these features. Though it is similar to LOF, the use of local-asymmetry-degree improves over the discriminating power of LOF in detecting outliers.

3.2.1.3 Clustering-Based Methods

Clustering-based approaches for outlier detection are motivated by the imbalance in the distribution of outliers versus normal objects in a data set. This imbalance prevails in the data as outliers typically correspond to only a small fraction of the total data. Addressing this requirement, clustering-based outlier detection tends to consider clusters of small sizes as clustered outliers. The advantage of clustering-based approaches is that they don't have to be supervised. Moreover, these techniques can be used in an incremental mode, i.e., after establishing the clusters, new data objects can be inserted into the system and tested for outliers.

Though many of the clustering algorithms are capable of finding outliers in data, they cannot provide an assessment of the degree of deviation/novelty score of the objects. Therefore, it is required to extend the basic clustering techniques to handle the outlier detection task by incorporating suitable scoring schemes for measuring the deviation of objects. As per the definition, every cluster-based outlier should be local to some specific cluster in the clustering structure. For example, referring to Fig. 3.1b, object P_1 is local to cluster C_1 and P_2 is local to C_2. Thus, the objects P_1 and P_2 turn out to be cluster-based local outliers.

Given below are some of the clustering-based techniques for detecting local outliers and outlying clusters in data.

1. *Modified k-means clustering*: This is a two-phase method that identifies small clusters as outliers using minimum spanning trees. However, a measure for identifying the degree of each object being an outlier is not available with this method. Furthermore, the procedure for distinguishing small clusters from the rest is not incorporated with this algorithm.
2. *Cluster-Based Local Outlier Factor (CBLOF)*: This method determines local outliers by establishing clustering structure of the data using a one pass clustering technique. The CBLOF of an object is measured by considering both the size of the cluster that the object belongs to and the distance between the object and its closest big cluster.
3. *Single linkage hierarchical clustering*: This method works based on the self-consistent outlier reduction approach. This method can deal with data sets comprising diluting clusters, i.e., clusters that possess scattered objects in their periphery.
4. *Quantization and implied distance metrics*: This is a linear time method for outlier detection that includes a direct quantization procedure and discovers outlying clusters explicitly. It requires only two sequential scans for analyzing disk resident data sets, as it does not perform object level comparisons and only stores the cell counts.

5. *Density-based subspace clustering*: This technique defines outliers with respect to maximal and non-redundant subspace clusters. Basically, subspace clustering aims at finding clusters in any subspace of a high-dimensional feature space and hence may result in some overlapping clusters in different subspaces.
6. *Local density based spatial clustering*: This method gives importance to the local data behavior and detects outliers using the LOF-based computation. It defines a new measure called Cluster-Based Outlier Factor (CBOF), which captures the essence of cluster-based outliers. The higher the CBOF of a cluster is, the more abnormal that cluster would be.

3.2.2 Learning Scheme Involved

Outlier detection algorithms of both predictive (supervised) and direct (unsupervised) varieties exist. Looking at various forms of the outlier detection problem in varied application settings, it is possible to group various methodologies into three broad categories based on the type of learning scheme involved.

1. *Determine outliers with no prior knowledge of the data*: This is essentially a learning approach analogous to *unsupervised learning* such as clustering. This approach considers the data as a static distribution and flags the most remote points as potential outliers.
2. *Model both normality and abnormality*: This approach is analogous to *supervised learning* and requires pre-labeled data corresponding to both normal and abnormal classes.
3. *Model only normality or in a few cases model abnormality*: This approach is generally known as novelty detection. It is analogous to a *semi-supervised detection* as the normal class is taught, and the algorithm learns to recognize abnormality on its own.

In general, the kind of outlier detection algorithm required could be determined based on the nature of the data. Additionally, the method of detection also impacts the type of learning technique to be considered. While supervised measures determine similarity based on class information, data-driven measures determine similarity based on the data distribution. In principle, both these ideas can be combined to evolve hybrid methods.

3.2.2.1 Supervised Learning Techniques

Neural networks and decision trees are two common forms of predictive outlier analysis. As already known, supervised neural networks process numerical data and decision tree-based methods deal with categorical data.

Supervised learning methods typically build a prediction model for rare objects based on labeled data, and use it to classify each object as being an outlier or a normal object. These approaches face the problem of imbalanced class distribution in data, which in turn may affect the performance of the underlying classification model. Major disadvantages with supervised techniques include the necessity to have labeled data and the inability to detect new classes of rare objects.

3.2.2.2 Direct Techniques

Direct techniques (unsupervised), which include statistical, proximity (distance), density and clustering based techniques, are employed when labeled training sets are unavailable. Although typically more complex than predictive techniques, direct methods are less constrained, as the discovery process is not dependent upon any pre-defined models.

Unsupervised learning methods detect outliers characterized by their deviation from the normal data based on a suitable measure. The success of these methods depends on the choice of proximity measure, feature selection and weighting, etc. These methods suffer from a possible high rate of false positives, as previously unseen data tend to get recognized as outliers.

3.2.3 Type of Data Attributes

A set of data objects may be described using attributes that are purely numerical or a mix of numerical and categorical varieties. Detecting outliers in numeric data is a fairly well addressed problem, while detecting outliers in categorical data is actively being pursued due to various computational challenges posed by such data.

The fundamental issue in this regard is that various values of a categorical attribute cannot be ordered. Due to this limitation, one cannot define a requisite proximity measure over categorical data. Therefore, the only possibility is to consider the occurrence frequencies of distinct attribute values in a data set in defining such a measure. Proximity measures thus defined are known as data driven measures as they incorporate data specific details in their definition. Recent research in this direction has resulted in defining various data driven similarity measures. These measures are useful in designing various algorithms for outlier detection in categorical data.

Let $D = \{X_1, X_2, \ldots, X_n\}$ be a data set having n data objects described using m categorical attributes $\{f_i, f_2, \ldots, f_m\}$. The following is a brief on some of the techniques for detecting outliers in categorical data.

1. A Greedy method determines outliers by computing the entropy of the data, denoted as $E(D)$, based on the entropy values corresponding to each feature f_j. According to this method, a data object with maximum contribution to the entropy value gets labeled as the first outlier. It goes on identifying the outliers in successive iterations. Hence, this method requires a number of scans over the data set in order to discover a desired number of outliers.

2. Addressing the above issue, a method working based on Attribute Value Frequencies (AVF) of data computes the scores of objects $AVFScore(X_i)$ using the frequency counts. As per this method, objects with low scores are considered as outliers. This implies that objects with rare attribute values turn out to be outliers.

Many such methods exist based on diverse algorithmic strategies such as distance-based detection, clustering-based discovery, etc. However, more innovative efforts are being reported recently giving scope for developing effective and efficient algorithms in this regard.

3.2.4 Size and Dimensionality of Data

Most applications today deal with large volumes of high dimensional data making it a computationally challenging task. To cope with this real life requirement, many of the techniques evolved for outlier detection are extended to deal with large data.

1. ScaN with prIoritized Flushing (SNIF) technique detects outliers in data by processing arbitrarily large data sets through prioritized flushing. This technique associates each object with a priority, and, whenever the memory becomes full, flushes the objects with the lowest priorities. Priorities are assigned based on the likelihood that an object will be an outlier or not, with relatively less number of neighbors. The objective is to discover all the outliers with I/O overhead linear in the data size. It completes mining outliers by scanning the data at most twice, and sometime, even once.
2. Recursive Binning and Re-Projection (RBRP) technique partitions the data into multiple bins such that objects that are close to each other are likely to be assigned to the same bin. This is achieved by applying a recursive procedure similar to divisive hierarchical clustering. It then scans through each bin sequentially to find the approximate nearest neighbors of a data object. To facilitate faster convergence of scanning, objects in each bin are organized as per their order in the projection along the principal component of the objects in the bin. The objective is to achieve near-linear time performance by limiting the number of distance comparisons required in determining the nearest neighbors of an object so as to find out distance based outliers.

More such efforts on detecting outliers in high dimensional data are under progress for developing efficient techniques catering to various practical requirements.

3.2.4.1 Indexing Structures for Large Data

A frequently used idea across various outlier detection methods is to estimate the density of nearest neighbors of an object so that objects in low density regions get detected as outliers. For efficiently determining the nearest neighbors, spatial indexing structures such as kd-tree and X-tree are of immense use.

- *kd-tree*: This data structure is useful for several applications, such as searches involving a multidimensional search key. A *kd*-tree uses only splitting planes that are perpendicular to one of the coordinate system axes. For example, in a 3-dimensional *kd*-tree (also known as 3*d*-tree), the root cell is divided into eight leaf cells through a 3-level splitting. Every node of a typical *kd*-tree, from the root to the leaves, stores a point. As a consequence, each splitting plane must go through one of the points in the tree.
- *X-tree (eXtended node tree)*: This is a spatial access method that supports efficient query processing for high dimensional data. An X-tree consists of three different types of nodes: data nodes, directory nodes, and super-nodes. A data node contains rectilinear Minimum Bounding Rectangles (MBRs) together with pointers to the actual data objects, and the directory nodes contain MBRs together with pointers to subMBRs. The purpose of a super-node is to avoid splits in the directory that would result in an inefficient directory structure. The alternative to using larger node sizes are highly overlapping directory nodes which would require to access most of the child nodes during the search process.

For low-dimensional data, use of indexing structures can work extremely well and potentially scales as $Nlog(N)$ if the index tree can find an example's nearest neighbors in $log(N)$ time. However, index structures break down as the dimensionality increases. Therefore, sequential scanning over the index tree is a possibility in such situations.

3.2.4.2 Dealing with Dimensionality

Typically, the non-parametric algorithms attempt to detect outliers in data by computing the inter-object distances in full dimensional space. However, it is known that in very high dimensional spaces the concept of similarity may not be meaningful anymore as the data are very sparse. As a result, proximity based detection may flag every data object as a potential outlier. Also, many methods tend to miss out some outliers that are embedded in subspaces, as they cannot be seen when working with the entire feature space for detection.

The following is a short description on some of the early efforts on dealing with the dimensionality aspect towards outlier detection in data.

1. *Low-dimensional projections*: This method considers a data object as outlier, if it is present in a local region of abnormally low density in some lower dimensional projection space.
2. *Evolutionary search*: This technique works almost like a brute-force implementation over the search space in terms of finding projections with very negative sparsity coefficients, but at a much lower computational cost.
3. *Multiple projections*: The idea is to project the data onto $[0,1]$ interval with Hilbert space filling curves such that each successive projection improves the estimate of an example's outlier score in the full-dimensional space.

4. *Feature bagging*: This approach considers an ensemble of outlier detection algorithms with every algorithm using a small subset of features that are randomly selected from the original feature set. As a result, each detector identifies different outliers and also assigns an outlier score to every data object. These scores are then combined to find the overall outliers in the data. It is worth noting here that the problem of combining outlier detection algorithms however differs from that of classifier ensembles.

3.2.5 Nature of the Application

The specific nature of the end application requiring outlier detection capability is another influential aspect that restricts the set of feasible techniques for consideration. For the applications requiring real time response, the detection algorithms must be as quick as possible. In case of applications that are evolving in nature, the algorithms must account for this characteristic. In cases where accuracy is of primary concern, the same has to be ensured in choosing a detection method for outlier mining.

- *Stream data applications*: Many of the recent applications deal with data streams, where data arrive incrementally over time. Detecting outliers in stream data poses additional challenge as one cannot keep the entire data in memory for analysis. A technique dealing with data streams detects distance-based outliers using the *sliding window* model. This model is appropriate as stream monitoring applications are usually interested in analyzing the most recent behavior. The notion of one-time outlier query is introduced in this technique in order to detect outliers in the current window at arbitrary points in time.
- *Disk resident data applications*: For the applications operating on disk resident data, cluster-based outlier detection methods or the methods using quantization with a few scans over the data are preferred.
- *Time-series data applications*: Applications dealing with time-series data are another prominent variety. One such frequently encountered time-series application is analyzing the transactions of a financial market. As this analysis has direct bearing on the economic growth and further monetary policies, it assumes importance. Detecting suspicious objects as outliers is difficult using informal inspection and graphical displays, without due understanding of the application related semantics.

In addition to the above characteristics, some applications require the outlier discovery to happen as an *on-line* activity, so as to initiate necessary remedial action at the earliest. However, other applications may allow the same to happen *offline* taking cues from other related information sources. Accordingly, design of the discovery method should consider these details as appropriate.

3.2.6 Mode of Analysis

Mode of analysis basically refers to the manner in which the outlier detection process is carried out on data, such as *online* and *offline* activity, as shown in Fig 3.2. The requirement for online analysis typically arises in case of applications providing various web-based services such as banking, trading, shopping, ticket booking, etc. The specific requirement from the detection algorithm point of view is that it should complete the designated activity at a speed matching with the application functionality. This means to say that the analysis process should be carried out in a seamless manner, without bothering the consumers of the service. In addition to data analysis, a typical online shopping application has to monitor a number of computing cores and devices involved in serving the requests from thousands of users.

On the other hand, offline analysis has the advantage of dealing with a much involved outlier detection method, as it can only result in some preventive action being initiated to restrict the occurrence of anomalous events in future. It is generally expected to be a diagnosis process for explaining something that has already happened. It is also possible to build a repository of the detected signatures of such anomalies so that the same can be leveraged in defining an effective prevention strategy subsequently.

3.3 Illustration of Outlier Discovery Process

This section is meant for the beginners to this area of research who would want to get a quick understanding of the outlier discovery/mining process. It presents two illustrations of outlier detection using benchmark data sets taken from UCI Machine Learning repository. One data set each corresponding to numerical and categorical types of attributes is considered to give a practical view of analyzing such data for outlier mining.

The general practice in outlier detection is to determine the amount of deviation of each data object using a suitable method and then arrange the data in a ranked list based on the measured deviations. The performance of a detection algorithm can be evaluated by counting the number (q) of actual outliers present among the top p objects of the ranked outlier sequence. Here, p is the total number of known/desired outliers in the data. Then, the accuracy of the detection method is given by the ratio of q to p, i.e. $\frac{q}{p}$.

3.3.1 Experiments on Numerical Data

Sample experimentation over numerical data is carried out using a small data set named Iris, which consists of 150 objects belonging to 3 different classes: *Iris-*

Table 3.1 Sample experimentation on numerical data using Orca method

Dataset	# Objects in data	Class-wise data distribution	# Actual outliers (p)	Top-p objects in ranked list	# Actual outliers detected (q)	Detection accuracy (q/p)
Iris	105	50 (*Setosa*) 50 (*Versicolor*) 5 (*Virginica*)	5	**101**, 42, **104** **103**, 16	3	60.0%

setosa, *Iris-versicolor* and *Iris-virginica*. Each class contains 50 objects described using 4 continuous attributes. To make it suitable for outlier detection and to introduce imbalance in the size, every 10th object of the *Iris-virginica* class is considered, thus making it only 5 objects of this class being present in the modified data set of 105 objects. The small class objects are considered as outliers and the rest 100 objects as normal ones.

The *Orca method*, which determines outliers by measuring the distance of a data object to its nearest neighbors, is considered in this experimentation. Accordingly, performance of this method in detecting outliers is shown in Table 3.1. The indices of the top p objects in the ranked outlier list are shown explicitly. The indices corresponding to the actual/known outliers among these objects are shown in bold face.

Having seen the general methodology, one can attempt to discover outliers using any other detection method in a similar manner.

3.3.2 Experiments on Categorical Data

Two outlier detection methods specifically dealing with categorical attributes, namely AVF and Greedy are considered in this experimentation. Wisconsin breast cancer data set, described using 10 categorical attributes, is utilized in this work. This data consists of 699 objects belonging to two different classes *2 (benign)* and *4 (malignant)*. All the objects with missing attribute values are removed from this data. During data pre-processing for outlier detection, every sixth *malignant* object in the set is retained. As per the general practice, data objects belonging to the smallest class are considered as outliers. Accordingly, there are 39 (8%) *malignant* objects treated as outliers and 444 (92%) *benign* objects considered as normal objects in the processed data.

Observations in this regard are given in Table 3.2 using AVF and Greedy methods. To illustrate the discovery process, number of true outliers predicted by these methods corresponding to various top portions (p values) of the ranked list are shown. One can interpret from this tabulation that for $p = 56$, both these methods could predict

Table 3.2 Observations on categorical data for outlier detection

Sl.No.	Top portion considered (p)	# outliers detected (q)	
		Greedy method	AVF method
1	4	4	4
2	8	7	7
3	16	15	14
4	24	22	21
5	32	27	28
6	40	33	32
7	48	36	36
8	56	**39**	**39**

all the designated outlier objects (malignant objects) in the data. However, for other reported values of p, slight difference can be noted in their performance indicating the variation specific to the methodologies considered.

3.4 Emerging Technology Trends

With the view of presenting the emerging technology trends connected with outlier detection, some of the recent methodologies indicative of these trends are discussed below. The basic idea is to appreciate the research efforts in formulating various forms of the outlier detection problem in varied application domains.

3.4.1 Anomaly Detection in Discrete Sequence Data

A discrete (or symbolic) sequence is an ordered set of events such that the events are symbols taken from a finite alphabet. For example, a gene is a sequence of nucleic acids. In contract to detecting point-based anomalies by assessing the deviation of individual objects, anomaly detection in discrete sequences looks for anomalous subsequences and patterns with anomalous frequency of occurrence. Typical applications of such a problem include detecting anomalies in data pertaining to operating system calls, biological sequences, flight safety domain, etc. A point to be noted here is that the nature of sequence data and the nature of anomalies can differ fundamentally across domains.

Considering various methods developed for anomaly detection dealing with discrete sequence data, a typical classification divides them into the following three distinct categories. These categories correspond to various possible ways of formulating the anomaly detection problem for discrete sequences.

- *Sequence-based*: An entire sequence is considered as anomalous if it is significantly different from normal ones.
- *Contiguous subsequence-based*: A contiguous subsequence within a long sequence is treated as anomalous if it is found to be distinct from other subsequences within the same sequence.
- *Pattern-based formulation*: A given test pattern is anomalous if its frequency of occurrence in a test sequence is significantly different from its expected frequency in a normal sequence.

It is possible to apply these techniques to other related problems such as online anomaly detection and time series anomaly detection.

3.4.2 Fast Memory Efficient Outlier Detection in Data Streams

Outlier detection in streaming data is challenging due to the size of the data to be analyzed and such data cannot be stored in memory for long. The high rate at which stream data are generated increases the complexity of the problem further. The way out in this regard is to design efficient algorithms with limited memory usage to the extent possible.

A fast memory efficient local outlier detection method named as *MiLOF* deals with data streams. It extends the iLOF method to compute Local Outlier Factor (LOF) values of objects in an incremental manner. In contrast to unbounded memory requirement of iLOF algorithm, this method executes within limited memory. It stores only a fraction $m << n$ of the n objects that are present up to time T.

MiLOF method allocates one part of the memory to store b data points, and the remaining part to store k summaries of older data points. For each input object observed, its LOF value is computed and inserted in to working memory. Once the assigned memory limit is reached, the first $\frac{b}{2}$ data points in the memory are replaced with their summary, in the form of a set of k prototype vectors determined using k-means clustering algorithm. This algorithm ensures that the number of data points stored in memory to detect outliers is no more than $m = b + k$.

MiLOF method is further refined using a flexible k-means algorithm leading to higher outlier detection accuracy. The intuition behind the improved clustering algorithm is to reduce the density of outlier regions. This is achieved by keeping the summary of inliers in working memory, thereby assigning higher LOF values to future outliers. This in turn results in better distinguishability of outliers from inliers boosting their detection accuracy.

3.5 Summary

The outlier detection problem is presented in detail and various research issues that are of interest in designing suitable algorithms for this problem are also discussed. An unified view of the problem along with the issues concerned is furnished in this chapter to enable a holistic understanding of this emerging research area. Going by this idea, various important practical aspects of this problem are presented corresponding to each one of the research issues identified. Towards the end, a few current technology trends on outlier detection are presented by discussing some recent methodologies in this regard. More details including the recent developments connected to some of these issues are presented in the subsequent chapters.

3.6 Bibliographic Notes

The initial work on outlier detection was based on statistical methods [8, 16]. However, many of the statistical techniques deal with only single dimensional or at best univariate [8] model of the data. The survey presented in [30] brings out various forms of the outlier detection problem in different application settings. In addition to the anomaly detection scenario, it also discussed the use of outlier detection in medicine, pharmacology, etc. It groups the outlier detection methodologies into three broad categories. Similarly, an introductory material on outlier detection presented a taxonomy of various detection methods, which were divided into two broad categories: parametric and non-parametric techniques [10]. Similarly, a keen exploration of the outlier detection problem resulted in the identification of the core research issues in this regard as explained in [45].

Connected with the non-parametric techniques for outlier detection, a distance-based method was proposed [33] using a simple and intuitive definition for outliers involving a nested-loop algorithm. Addressing high computational cost of this algorithm, a modified formulation [42] used the *distance to the* kth *nearest neighbor* of a data object. To improve the computational efficiency, micro clustering phase of BIRCH algorithm [49] was utilized to quantize the space in near linear time and follow a cell-based approach for the computation. A scheme for measuring the *average distance to the* k*-nearest-neighbors* of a data object was considered [34] subsequently. In other such effort, the algorithm proposed in [5] for detecting outliers in large and high-dimensional data finds the weight of an object as the *sum of the distances to its* k*-nearest-neighbors*. It avoids the distance computation of each pair of objects by using the space-filling curves to linearize the data set. Another nested-loop algorithm, with quadratic time complexity in worst case, was found to give near linear time performance when the data were presented in random order [9]. It is also possible to prune the data in addition to randomizing it.

Referring to density-based methods for outlier detection, the Local Outlier Factor (LOF) [16] method employs Local Reachability Density (LRD) of data objects to measure their density. LOF method determines the likelihood of a data object being a local outlier. The notion of local outliers is based on the same theoretical foundation of the density-based clustering algorithm known as Density Based Spatial Clustering of Applications with Noise (DBSCAN) [22]. Later on, an improvised method was proposed [32] using the concept of micro-clusters introduced in BIRCH algorithm [49]. A cut-plane solution was proposed in this work to deal with overlapping micro-clusters. A cut-plane for two micro-clusters, denoted as $cp(MC_i, MC_j)$, is a hyper-plane that is perpendicular to the line between c_i and c_j and divides the line into exactly two halves. Similarly, a method named as LOcal Correlation Integral (LOCI) [40] works based on a deviation measure called the Multi-granularity DEviation Factor (MDEF). This method provides an automatic data driven cut-off for determining outliers by taking into account the distribution of distances between pairs of objects. Another variant of the LOF method is the local distribution based algorithm [51], which is also meant for finding local outliers. It was claimed that this technique outperforms LOF in identifying meaningful and interesting outliers.

With regard to clustering-based methods, algorithms such as BIRCH [49] and DBSCAN [22] are robust and as such tolerant to outliers but were specifically optimized for clustering large data sets. The computations are performed over tree structured indices and cluster hierarchies. A two-phase algorithm [31], making use of a modified k-means algorithm, clusters the data in order to find outliers. Addressing the drawbacks of the above algorithm, another local outlier detection method was developed [29] using the Cluster-Based Local Outlier Factor (CBLOF). The first step in this method is to establish a clustering structure of the data using a one pass clustering technique known as Squeezer algorithm [29]. An improved single linkage hierarchical clustering-based method given in [3] works based on a self-consistent outlier reduction approach. A linear time algorithm [17] was developed using quantization and implied distance metrics. A density-based subspace clustering method [7] defined outliers with respect to maximal and non-redundant subspace clusters. The subsequent technique is to detect cluster-based outliers using a local density based spatial clustering algorithm [21].

On the issue of the learning method involved, it is understood that outlier detection methods belong to both supervised and unsupervised learning varieties [47]. A neural networks-based technique, known as replicator neural network [27] works based on the observation that a trained neural network will reconstruct some small number of objects poorly, and these objects can be considered as outliers. The outlier factor for ranking the data is measured according to the magnitude of the reconstruction error. A detailed survey on the use of neural networks for outlier detection was presented in [39]. An unsupervised method employing Support Vector Machines (SVMs) based on p-kernels was developed for anomaly detection [38]. Similarly, a semi-supervised method for detecting outliers in data was proposed in [23, 52].

Similarly, the type of data attributes is a significant issue due to which the process of detecting outliers in categorical data gained prominence [19, 24, 30]. As brought out in [14], various categories (values) of a categorical attribute cannot be ordered. Therefore, recent research in this direction resulted in defining some data driven similarity measures [15]. The replicator neural networks based method [27], is one of the early methods for detecting outliers in categorical data. Similarly, cluster-based outlier detection method presented in [29] works based on the notion of local outliers. A greedy method presented in [28] computes the entropy of a categorical data set to determine outliers. On the other hand, AVF algorithm [35] was developed using the attribute value frequencies of a data set.

Regarding the size aspect of data, a technique named as ScaN with prIoritized Flushing (SNIF) [46] detects outliers through prioritized flushing. Some of the methods discussed in [37] for mining massive data sets can be inherited to carry out outlier detection involving large data. Similarly, the recursive binning and re-projection (RBRP) technique [25] is a distance-based outlier detection method dealing with large, high dimensional data sets. For efficiently identifying the nearest neighbors of a data object to determine its density, spatial indexing structures such as kd-tree [11], X-tree [12] are popularly used by researchers. It is stated [1, 13] that in very high dimensional spaces, the data are very sparse and the concept of similarity may not be meaningful anymore.

Addressing the curse of dimensionality, a method [1] was developed using low-dimensional projections to find outliers in the projected space. The technique proposed in [2] deals with high dimensional applications effectively by using an evolutionary search technique. A method using multiple projections onto the interval [0,1] with Hilbert space filling curves was presented in [4]. A feature bagging approach for detecting outliers in very large, high dimensional data [36] was designed by considering an ensemble of outlier detection algorithms.

Finally, the nature of application is yet other significant issue regarding outlier detection. Accordingly, a technique dealing with data streams detects distance-based outliers using the sliding window model [6]. Similarly, a method for continuous angle-based detection of outliers in high-dimensional data streams was proposed [48]. In the category of time-series data applications, financial market data are often affected by either missing or erroneous data objects, or unknown external events which can have a large impact on individual objects [26]. Similarly, a framework for discovering periodic outlier patterns in time-series sequences was presented in [43]. With regard to developing online anomaly detection methods, a solution was described in [50] for monitoring millions of cores in services and billions of Windows devices involved in Microsoft Store application.

For a quick illustration of the outlier discovery process, sample experimentation over benchmark data sets taken from UCI Machine Learning repository [20] is provided in this chapter. In case of numerical data, the Orca method [9] is considered as it is a distance-based method for outlier detection.

One may find numerous recent developments with regard to outlier detection as the active directions to pursue. The survey presented in [18] gives a glimpse of various methods developed for anomaly detection dealing with discrete/symbolic sequence data. A fast memory efficient local outlier detection method *MiLOF* was proposed recently [44] extending the iLOF algorithm [41].

References

1. Aggarwal, C.C., Yu, P.S.: Outlier detection for high dimensional data. In: ACM SIGMOD International Conference on Management of Data, Santa Barbara, USA, pp. 37–46 (2001)
2. Aggarwal, C.C., Yu, P.S.: An effective and efficient algorithm for high-dimensional outlier detection. VLDB J. **14**(2), 211–221 (2005)
3. Almeida, J.A.S., Barbosa, L.M.S., Pais, A.A.C.C., Formosinho, S.J.: Improving hierarchical cluster analysis: a new method with outlier detection and automatic clustering. Chemom. Intell. Lab. Syst. **87**, 208–217 (2007)
4. Angiulli, F., Pizzuti, C.: Fast outlier detection in high dimensional spaces. In: 6th European Conference on the Principles of Data Mining and Knowledge Discovery, pp. 15–26 (2002)
5. Angiulli, F., Pizzuti, C.: Outlier mining in large high-dimensional data sets. IEEE Trans. Knowl. Data Eng. (KDE) **17**, 203–215 (2005)
6. Angiulli, F., Fassetti, F.: Distance-based outlier queries in data streams: the novel task and algorithms. Data Min. Knowl. Discov. **20**(2), 290–324 (2010)
7. Assent, I., Krieger, R., Muller, E., Seidl, T.: Subspace outlier mining in large multimedia databases. In: Dagstuhl Seminar Proceedings on Parallel Universes and Local Patterns (2007)
8. Barnett, V., Lewis, T.: Outliers in Statistical Data. Wiley, New York (1994)
9. Bay, S.D., Schwabacher, M.: Mining distance-based outliers in near linear time with randomization and a simple pruning rule. In: Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 29–38. ACM, Washington, DC, USA (2003)
10. Ben-Gal, I.: Outlier detection. In: Maimon, O., Rockack, L. (eds.) Data Mining and Knowledge Discovery Handbook: A Complete Guide for Practitioners and Researchers, pp. 1–16. Kluwer Academic Publishers (2005)
11. Bentley, J.L.: Multidimensional binary search trees used for associative searching. Commun. ACM **18**(9), 509–517 (1975)
12. Berchtold, S., Keim, D., Kreigel, H.P.: The x-tree: an index structure for high-dimensional data. In: 22nd International Conference on Very Large Databases, pp. 28–39 (1996)
13. Beyer, K.S., Goldstein, J., Ramakrishnan, R., Shaft, U.: When is nearest neighbor meaningful? In: 7th International Conference on Database Theory, ICDT. Lecture Notes in Computer Science, vol. 1540, pp. 217–235. Springer, Jerusalem, Israel (1999)
14. Bock, H.H.: The classical data situation. In: Analysis of Symbolic Data, pp. 139–152. Springer (2002)
15. Boriah, S., Chandola, V., Kumar, V.: Similarity measures for categorical data: a comparative evaluation. In: SIAM International Conference on Data Mining, Atlanta, Georgia, USA, pp. 243–254 (2008)
16. Breunig, M., Kriegel, H., Ng, R., Sander, J.: Lof: Identifying density-based local outliers. In: ACM SIGMOD International Conference on Management of Data, Dallas, Texas, pp. 93–104 (2000)
17. Ceglar, A., Roddick, J.F., Powers, D.M.W.: CURIO: a fast outlier and outlier cluster detection algorithm for large datasets. In: Ong, K.L., Li, W., Gao, J. (eds.) Second International Workshop on Integrating AI and Data Mining, Conferences in Research and Practice in Information Technology, vol. 84. Australian Computer Society Inc., Gold Coast, Australia (2007)
18. Chandola, V., Banerjee, A., Kumar, V.: Anomaly detection for discrete sequences: a survey. IEEE Trans. Knowl. Data Eng. (TKDE) **24**(5), 823–839 (2012)
19. Das, K., Schneider, J.: Detecting anomalous records in categorical datasets. In: ACM KDD, San Jose, California, pp. 220–229 (2007)
20. Dua, D., Efi, K.T.: UCI machine learning repository (2017). http://archive.ics.uci.edu/ml
21. Duan, L., Xu, L., Liu, Y., Lee, J.: Cluster-based outlier detection. Ann. Oper. Res. **168**, 151–168 (2009)
22. Ester, M., Kriegel, H., Sander, J., Xu, X.: A density based algorithm for discovering clusters in large spatial databases. In: ACM KDD, Portland, Oregon, pp. 226–231 (1996)

23. Gao, J., Cheng, H., Tan, P.N.: Semi-supervised outlier detection. In: ACM SIGAC Symposium on Applied Computing, pp. 635–636. ACM Press, New York, USA (2006)
24. Ghoting, A., Otey, M.E., Parthasarathy, S.: LOADED: link-based outlier and anomaly detecting in evolving data sets. In: International Conference on Data Mining, pp. 387–390 (2004)
25. Ghoting, A., Parthasarathy, S., Otey, M.: Fast mining of distance-based outliers in high-dimensional datasets. In: SIAM International Conference on Data Mining (SDM), pp. 608–612. SIAM, Bethesda, MA, USA (2006)
26. Gutierrez, J.M.P., Gregori, J.F.: Clustering techniques applied to outlier detection of financial market series using a moving window filtering algorithm. In: Unpublished Working Paper Series, No. 948, European Central Bank. Frankfurt, Germany (2008)
27. Harkins, S., He, H., Williams, G.J., Baxter, R.A.: Outlier detection using replicator neural networks. In: Kambayashi, Y., Winiwarter, W., Arikawa, M. (eds.) 4th International Conference on Data Warehousing and Knowledge Discovery (DaWak). LNCS, vol. 2454, pp. 170–180. Springer, Aixen-Provence, France (2002)
28. He, Z., Xu, X., Deng, S.: A fast greedy algorithm for outlier mining. In: Proceedings of Pacific Asia Conference on Knowledge Discovery in Databases (PAKDD), Singapore, pp. 567–576 (2006)
29. He, Z., Xu, X., Deng, S.: Discovering cluster-based local outliers. Patten Recognit. Lett. **24**, 1641–1650 (2003)
30. Hodge, V.J., Austin, J.: A survey of outlier detection methodologies. Artif. Intell. Rev. **22**, 85–126 (2004)
31. Jiang, M.F., Tseng, S.S., Su, C.M.: Two-phase clustering process for outliers detection. Pattern Recognit. Lett. **22**(6–7), 691–700 (2001)
32. Jin, W., Tung, A.K.H., Han, J.: Mining top-n local outliers in large databases. In: Proceedings of the seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 293–298. ACM, San Francisco, CA, USA (2001)
33. Knorr, E., Ng, R.: Algorithms for mining distance-based outliers in large data sets. In: 24th International conference on Very Large Databases (VLDB), New York, pp. 392–403 (1998)
34. Knorr, E., Ng, R., Tucakov, V.: Distance-based outliers: algorithms and applications. VLDB J. Very Large Databases **8**(3–4), 237–253 (2000)
35. Koufakou, A., Ortiz, E., Georgiopoulos, M.: A scalable and efficient outlier detection strategy for categorical data. In: Proceedings of IEEE ICTAI, Patras, Greece, pp. 210–217 (2007)
36. Lazarevic, A., Kumar, V.: Feature bagging for outlier detection. In: ACM KDD, Chicago, USA, pp. 157–166 (2005)
37. Leskovec, J., Rajaraman, A., Ullman, J.D.: Mining of Massive Datasets, 2nd edn. Cambridge University Press (2014)
38. Li, K., Teng, G.: Unsupervised SVM based on p-kernels for anomaly detection. IEEE International Conference on Innovative Computing, Information and Control, Beijing, China, pp. 59–62 (2006)
39. Markou, M., Singh, S.: Novelty detection: a review, Part 2: neural network based approaches. Signal Process. **83**(12), 2499–2521 (2003)
40. Papadimitriou, S., Kitagawa, H., Gibbons, P.B., Faloutsos, C.: LOCI: fast outlier detection using the local correlation integral. In: Proceedings of the 19th International Conference on Data Engineering, pp. 315–326. IEEE Computer Society, Bangalore, India (2003)
41. Pokrajac, D., Lazarevic, A., Latecki, L.J.: Incremental local outlier detection for data streams. In: Proceedings of the IEEE Symposium on Computational Intelligence and Data Mining, CIDM, pp. 504–515. IEEE, Honolulu, Hawaii, USA (2007)
42. Ramaswami, S., Rastogi, R., Shim, K.: Efficient algorithms for mining outliers from large data sets. In: ACM SIGMOD International Conference on Management of Data, pp. 427–438. ACM Press, New York (2000)
43. Rasheed, F., Alhajj, R.: A framework for periodic outlier pattern detection in time-series sequences. IEEE Trans. Cybern. **44**(5), 569–582 (2014)
44. Salehi, M., Leckie, C., Bezdek, J.C., Vaithianathan, T., Zhang, X.: Fast memory efficient local outlier detection in data streams. In: 33rd IEEE International Conference on Data Engineering. ICDE, pp. 51–52. IEEE, San Diego, CA, USA (2017)

45. Suri, N.N.R.R., Murty, M., Athithan, G.: Data mining techniques for outlier detection. In: Zhang, Q., Segall, R.S., Cao, M. (eds.) Visual Analytics and Interactive Technologies: Data, Text and Web Mining Applications, Chap. 2, pp. 22–38. IGI Global, New York, USA (2011)
46. Tao, Y., Xiao, X., Zhou, S.: Mining distance-based outliers from large databases in any metric space. In: 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 394–403. ACM Press, Philadelphia, PA, USA (2006)
47. Torgo, L., Ribeiro, R.: Predicting outliers. In: Lavrac, N., Gamberger, D., Todorovski, L., Blockeel, H. (eds.) Principles of Data Mining and Knowledge Discovery. LNAI, vol. 2838, pp. 447–458. Springer (2003)
48. Ye, H., Kitagawa, H., Xiao, J.: Continuous angle-based outlier detection on high-dimensional data streams. In: IDEAS, pp. 162–167. ACM, Yokohama, Japan (2015)
49. Zhang, T., Ramakrishnan, R., Livny, M.: Birch: An efficient data clustering method for very large databases. In: ACM SIGMOD International Conference on Management of Data, pp. 103–114. ACM Press, Montreal, Canada (1996)
50. Zhang, J., Wydrowski, R., Wang, Z., Arrabolu, S.S., Kanazawa, K., Gudalewicz, L., Gao, H., Batoukov, R., Aghajanyan, S., Tran, K.: Mbius: online anomaly detection and diagnosis. In: KDD. El London, UK (2018)
51. Zhang, Y., Yang, S., Wang, Y.: LDBOD: a novel distribution based outlier detector. Pattern Recognit. Lett. **29**, 967–976 (2008)
52. Zhu, X., Goldberg, A.: Introduction to Semi-Supervised Learning. Morgan and Claypool Publishers (2009)

Chapter 4
Computational Preliminaries

Abstract This chapter presents the mathematical notation followed to represent the data and the computational measures defined on the data. Basics of matrix algebra and information theory are furnished as they form the building blocks of the computational model followed here. The standard procedure for preparing data sets of various types to perform outlier detection is also covered. In essence, the objective is to present the computational preliminaries necessary for developing various algorithmic methods for outlier detection in multi-dimensional record data as well as anomaly detection in network/graph data.

4.1 Basic Notation and Definitions

This section covers the mathematical notation used for representing the categorical data, which is one of the key research issues addressed in this book. It also presents certain computational measures defined on categorical data that are frequently used in carrying out an analysis of such data.

4.1.1 Outliers versus Anomalies

As described in Sect. 2.1.2, anomalies in a network signify irregularities in numerous real life applications. In most of the cases, anomaly detection is intended to understand evolving new phenomena that is not seen in the past data. A standard method for detecting anomalies is to create a model of the normal data and compare the future observations against the model. However, as the definition of normality differs across various problem domains, the problem of anomaly detection turns out to be a more challenging and involved process. A generic computational approach is

N. N. R Ranga Suri et al., *Outlier Detection: Techniques and Applications*,
Intelligent Systems Reference Library 155,
https://doi.org/10.1007/978-3-030-05127-3_4

to look for outliers in a given data set. Thus, much of the research related to outlier detection has evolved in the context of anomaly detection.

As mentioned above, outlier detection has several useful implications in various real life applications. Successful detection of outliers helps in better determination of various anomalous scenarios where the underlying system may tend to malfunction or may exhibit some unexpected behavior. Thus, outlier detection in data leads to anomaly identification in practical systems. There may exist subtle differences between an outlier and an anomaly in the way they are perceived in varied application contexts. However, from the data mining point of view, both these kinds of objects display similar characteristics in terms of their deviation from the normality, quantified using an acceptable measure. Hence, no distinction is made between these two types of objects in terms of their detection in a data set.

4.1.2 Representing Data

Data pertaining to many real world applications is described mainly by various categorical (also known as nominal) attributes. Unlike continuous valued attributes, categorical attribute values cannot be naturally mapped on to a scale, making most continuous data analysis techniques inapplicable in dealing with such data. Thus, it is important to develop requisite categorical data analysis techniques by defining necessary distance (dissimilarity) measures on categorical attributes. The notation given below is followed to refer to a categorical data set throughout this book.

Let $D = \{X_1, X_2, X_3, \ldots, X_n\}$ be a set of n data objects, described using m categorical attributes $A_1, A_2, \ldots, A_m$. Each attribute A_r has a domain of values, $DOM(A_r)$. Each data object X_i is represented as $[x_{i,1}, x_{i,2}, \ldots, x_{i,m}]$, where $x_{i,r} \in DOM(A_r)$ for $1 \le r \le m$. For any two data objects X_i and X_j, $X_i = X_j$ if and only if $x_{i,r} = x_{j,r}$ for all $1 \le r \le m$.

For various data mining tasks such as outlier detection, distance (dissimilarity) computation between a pair of data objects is vital. A simple *object-to-object distance* between two categorical objects X_i and X_j can be defined as

$$d(X_i, X_j) = \sum_{r=1}^{m} \delta(x_{i,r}, x_{j,r}) \tag{4.1}$$

where

$$\delta(x_{i,r}, x_{j,r}) = \begin{cases} 0, \text{ if } x_{i,r} = x_{j,r}, \\ 1, \text{ if } x_{i,r} \neq x_{j,r}. \end{cases}$$

It is understood that the above function d, known as *overlap* measure, defines a metric on the set of categorical objects. It is also known that d is a kind of generalized Hamming distance. In case of binary attributes, the overlap measure exactly matches with Hamming distance.

4.1.3 Data-Driven Proximity Measures

The above overlap metric and its variants assume that all the values of a categorical/ nominal attribute are of equal distance from each other, whereas, it is known that various categories (values) of a categorical attribute cannot be ordered. Therefore, it is not possible to represent categorical value pairs with differing degrees of similarities required in defining suitable proximity measure to deal with the real world applications. For example, considering a real life attribute *taste*, it is more meaningful to have the value *sour* closer to *sweet* than to *bitter*. As a result, a real valued distance metric is often preferred over a Boolean one. Hence, it is important to take into account the occurrence frequencies of distinct attribute values in a data set for defining a proximity measure over categorical data. Proximity measures thus defined are known as *data-driven* measures as they incorporate data-specific details in their definition.

Though the problem of defining suitable proximity measures over categorical data is widely addressed, only the recent research in this direction has made a thorough exploration of certain data-driven similarity measures.

Some of the well known data-driven categorical similarity measures are listed below. All these measures have the general formula to compute the similarity between two objects X_i and X_j as

$$S(X_i, X_j) = \sum_{r=1}^{m} \frac{1}{m} S_r(x_{i,r}, x_{j,r})$$

1. *Eskin Measure*:

$$S_r(x_{i,r}, x_{j,r}) = \begin{cases} 1, & \text{if } x_{i,r} = x_{j,r} \\ \frac{n_r^2}{n_r^2+2}, & \text{otherwise} \end{cases} \tag{4.2}$$

   where n_r is the number of values of the categorical attribute A_r.

2. *Inverse Occurrence Frequency (IOF) Measure*:

$$S_r(x_{i,r}, x_{j,r}) = \begin{cases} 1, & \text{if } x_{i,r} = x_{j,r} \\ \frac{1}{1+\lg freq(x_{i,r}) * \lg freq(x_{j,r})}, & \text{otherwise} \end{cases} \tag{4.3}$$

   where $freq(x_{i,r})$ is the number of objects in the data set D with the category value $x_{i,r}$ for the attribute A_r.

3. *Lin's Measure*:

$$S_r(x_{i,r}, x_{j,r}) = \begin{cases} 2 \lg p(x_{i,r}), & \text{if } x_{i,r} = x_{j,r} \\ 2 \lg(p(x_{i,r}) + p(x_{j,r})), & \text{otherwise} \end{cases} \tag{4.4}$$

   where $p(x_{i,r}) = \frac{freq(x_{i,r})}{n}$.

For supervised learning problems involving categorical data with class labels, the Value Difference Metric (VDM) is a suitable one. According to this metric, the distance between any two objects X_i and X_j is given by

$$VDM(X_i, X_j) = \sum_{r=1}^{m} \psi(x_{i,r}, x_{j,r}) \tag{4.5}$$

where

$$\psi(x_{i,r}, x_{j,r}) = \sum_{c_k} (P(c_k|x_{i,r}) - P(c_k|x_{j,r}))^2$$

and

$$P(c_k|x_{i,r}) = \frac{freq_{c_k}(x_{i,r})}{freq(x_{i,r})}$$

where $freq_{c_k}(x_{i,r})$ is the number of objects belonging to class c_k with the category value $x_{i,r}$ for the categorical attribute A_r. Similarly, $freq(x_{i,r})$ is the number of objects in the data set D with the category value $x_{i,r}$ for the attribute A_r.

4.1.4 Dissimilarity Measure for Clustering

Data clustering is a frequently used unsupervised learning technique in many applications such as outlier detection in data. In this context, the dissimilarity measure expressed in Eq. 4.1 above is too simple to be used for clustering-based applications, as it does not apply to the case of dissimilarity between a cluster and a single object or another cluster. For this reason, the dissimilarity measure given below is befitting for clustering tasks as it incorporates the attribute value frequencies in its computation.

1. Let $Z_l = [z_{l,1}, z_{l,2}, \ldots, z_{l,m}]$ be the representative of the lth cluster in the clustering process, obtained using an acceptable cluster initialization method. Then, the *object-to-cluster distance* between a data object X_i and Z_l is given by

$$d(Z_l, X_i) = \sum_{r=1}^{m} \phi(z_{l,r}, x_{i,r}) \tag{4.6}$$

where

$$\phi(z_{l,r}, x_{i,r}) = \begin{cases} 1, & \text{if } z_{l,r} \neq x_{i,r}, \\ 1 - \frac{|c_{l,r}|}{|c_l|}, & \text{otherwise.} \end{cases}$$

where $|c_l|$ is the number of objects in the lth cluster and $|c_{l,r}|$ is the number of objects with category value $x_{i,r}$ for the rth attribute in that cluster. Data-driven measures of this kind determine similarity based on the data distribution.

2. The notion of density in the context of categorical data is generally used to refer to the data objects with same values corresponding to a majority of the attributes as that of the object in question. Accordingly, the *density* of a data object X_i can be computed as

$$density(X_i) = \frac{1}{mn} \sum_{r=1}^{m} freq(x_{i,r}) \tag{4.7}$$

where $freq(x_{i,r})$ denotes the number of objects in a data set D with the value $x_{i,r}$ for the rth attribute. The density of an object is a characteristic measure of its neighborhood. The general assumption is that there are no identical objects in the given data set.

4.2 Basics of Matrix Algebra

A data set D consisting of n data objects each described using m attributes is typically represented as a data matrix as given below.

$$D = \begin{bmatrix} x_{11} & x_{12} & \dots & x_{1m} \\ x_{21} & x_{22} & \dots & x_{2m} \\ \dots & \dots & x_{ij} & \dots \\ x_{n1} & x_{n2} & \dots & x_{nm} \end{bmatrix} \tag{4.8}$$

Here, D turns out to be a rectangular matrix of size $n \times m$. Many of the data manipulation functions such as eigenvector computation and principal component analysis can be carried out based on the data matrix.

The following algebraic properties of matrices are useful for data analysis.

1. The *columnspace* of a matrix A, denoted by $R(A)$ is the span of its columns.
2. The *nullspace* $N(A)$ of a matrix $A \in R^{m \times n}$ is the set of all vectors that equal $\mathbf{0}$ when multiplied by A. The dimensionality of the nullspace is known as *nullity* of the matrix A.

$$N(A) = \{x \in R^n : Ax = 0\} \tag{4.9}$$

3. The *rank* of a matrix $A \in R^{m \times n}$ is the size of the largest subset of columns of A that constitute a linearly independent set, i.e.,

$$rank(A) \leq min(m, n) \tag{4.10}$$

4. Given a square matrix $A \in R^{n \times n}$, λ is said to be an eigenvalue of A and vector $\overrightarrow{x}$ the corresponding eigenvector if $A\overrightarrow{x} = \lambda\overrightarrow{x}$. Solving the following characteristic equation of A gives its eigenvalues.

$$|A - \lambda I| = 0 \tag{4.11}$$

Then, the rank of A is given by the number of non-zero eigenvalues of A.
5. The *trace* of a square matrix is the sum of its diagonal entries.

$$tr(A) = \sum_{i=1}^{n} A_{ii} \tag{4.12}$$

The trace of A can also be determined using the eigenvalues of A as

$$tr(A) = \sum_{i} \lambda_i(A) \tag{4.13}$$

6. The *determinant* of a square matrix is equal to the product of its eigenvalues.

$$det(A) = \prod_{i} \lambda_i(A) \tag{4.14}$$

7. A symmetric matrix A is *positive semi-definite* if for all $x \in R^n$,

$$x^T Ax \geq 0. \tag{4.15}$$

This is denoted as $A \succeq 0$. All eigenvalues of such a matrix turn out to be non-negative.
8. Similarly, a symmetric matrix A is *positive definite* if for all nonzero $x \in R^n$,

$$x^T Ax > 0. \tag{4.16}$$

This is denoted as $A \succ 0$.
9. A square matrix A is said to be *diagonalizable* if $\exists$ S such that

$$A = S\Lambda S^{-1} \tag{4.17}$$

where

- The columns of matrix S correspond to the eigenvectors of matrix A,
- Λ is a diagonal matrix of the eigenvalues of matrix A.

The diagonalization condition holds good only when eigenvalues of matrix A are distinct and thus matrix S becomes invertible.

10. The *Singular Value Decomposition* (SVD) of an mxn matrix A is given by

$$A = U\Sigma V^T \tag{4.18}$$

where

- $U \in R^{m\times m}$, the columns of U are known as the left singular vectors of A.
- $\Sigma \in R^{m\times n}$, is a rectangular diagonal matrix with each element being the square root of an eigenvalue of AA^T or A^TA.
- $V \in R^{n\times n}$, the columns of V are known as the right singular vectors of A.

There are many more interesting and useful properties related to matrix algebra that one may refer to some detailed material on this subject.

4.3 Basics of Information Theory

The number of applications employing *Information Theory* principles for accomplishing the computational objectives is ever increasing. This is due to the ease of understanding and elegant formulation of various quantitative measures defined on the basis of random variable concept.

Let x be a discrete random variable that can take any value from the finite set $\{x_1, x_2, \dots, x_k\}$. Then, the following properties hold good for x.

1. The probability of x assuming value x_j is denoted as

$$p(x_j) = Pr[x = x_j], \quad j = 1, 2, ..., k. \tag{4.19}$$

2. The following two conditions must hold for any random variable x.

$$p(x_j) \geq 0 \text{ and } \sum_{j=1}^{k} p(x_j) = 1. \tag{4.20}$$

3. The expected value or mean (μ) of x is given by

$$E[x] = \sum_{j=1}^{k} x_j p(x_j) \tag{4.21}$$

Here, μ represents the arithmetic average of the values in a large random sample.

4. According to Information Theory, entropy measures the uncertainty associated with a random variable. The entropy $H(x)$ of a discrete random variable x is given by

$$H(x) = -\sum_{j} p(x_j)\log(p(x_j)) \tag{4.22}$$

where $p(x_j)$ is the marginal probability value of the variable x corresponding to its jth value.

5. Similarly, Mutual Information (MI) corresponds to the interdependence between two random variables x and y, defined as

$$I(x, y) = H(x) + H(y) - H(x, y) \tag{4.23}$$

6. Given two discrete random variables x and y, their MI value can be computed as

$$I(x, y) = \sum_i \sum_j p(x_i, y_j) log \frac{p(x_i, y_j)}{p(x_i)p(y_j)} \tag{4.24}$$

where $p(x, y)$ is the joint probability distribution and $p(x)$, $p(y)$ are marginal probability distributions. These values can be determined by counting method.

4.4 Data Set Preparation for Outlier Detection

Researchers dealing with outlier detection task are often faced with the problem of not finding any relevant data sets for evaluating the methods developed by them. This is because there are no known public domain benchmark data sets consisting of outliers naturally. As a result, one has to settle with some alternate means of evaluating their methods following some best practices. This is to say that experimental evaluation of any machine learning method requires benchmark data sets with the desired characteristics. Therefore, preparing the data sets as per the format demanded by the algorithmic implementation is necessary to carry out discovery of outliers in data.

4.4.1 A Critical Look

Most published works on outlier detection resort to the use of data sets from specific applications or just employ synthetic data. This results in the following practical problems in their evaluation of the methods.

- Inability to assess the complexity of the anomaly detection problem based on a standard set of properties of the data,
- Unable to generate new data sets (aside from subsampling) that may differ in controlled ways, required for algorithmic validation.
- As synthetic data sets carry no real world validity to the anomalies, algorithms found to work satisfactorily on such data have no guarantee of displaying similar performance with real world data.

As the goal of outlier detection is to identify data objects that are semantically distinct from the normal objects, it necessiates the following three requirements for the benchmark data sets.

- Normal data objects should be drawn from a real-world generating process.
- Anomalous data objects should also be from a real-world process that is semantically distinct from the process generating the normal objects.
- Many benchmark data sets are needed to ensure generality and prevent over fitting of an anomaly detection algorithm.
- Benchmark data sets should be characterized in terms of well defined and meaningful problem dimensions that can be systematically varied.

To address these issues, one can transform classification data sets into ground-truth benchmark data sets for anomaly detection. In this connection, the following four problem dimensions may be considered for creating families of anomaly detection problems from real world data sets.

- *Point difficulty*: It indicates the level of difficulty in detecting an anomalous data point in a data set. A large value of its distance from the rest of the data implies high chance of it becoming an outlier.
- *Relative frequency*: It is the fraction of the incoming data objects that are (true) anomalies. As per the literature, the relative frequency value is typically found to be in the range $[0.01, 0.1]$.
- *Semantic variation*: It is a measure of the degree to which the anomalies are generated by more than one underlying process. Alternatively, a measure of *clusteredness* can also be employed for this purpose.
- *Feature relevance*: It is important to vary the set of features to manipulate both the power of the relevant features and the number of irrelevant (noise) features.

After defining normal versus anomalous data objects on a real data set, a specific benchmark data set can be generated by choosing suitable values for each one of the above listed problem dimensions.

4.4.2 Preparing Categorical Data Sets

The standard practice to test the effectiveness of an outlier detection algorithm is to employ the procedure consisting of the data preparation steps as shown in Fig. 4.1.

1. The first and simple step is to eliminate all the objects with missing attribute values in the data. In some application contexts, variable imputation is preferred in place of missing values, rather than eliminating such objects.
2. The next step is to identify a minority class(es) (with number of objects around 5–10% of the number of objects in the entire data) in the data set and designate the objects belonging to such minority class(es) as outliers for enabling outlier detection.

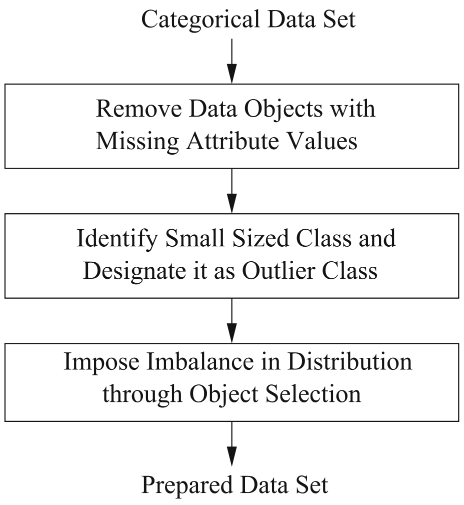

Categorical Data Set
Remove Data Objects with Missing Attribute Values
Identify Small Sized Class and Designate it as Outlier Class
Impose Imbalance in Distribution through Object Selection
Prepared Data Set

Fig. 4.1 Steps in data preparation for outlier detection

3. Another significant preprocessing step is to consider only a subset of the objects belonging to the designated outlier class(s) in order to induce a kind of imbalance in the distribution of normal vs outlier objects in the prepared data set. This is because imbalance in the distribution of objects is a qualifying characteristic of the data sets consisting of outliers naturally.

4.4.3 Preparing Network/Graph Data Sets

Representing network data in the form of a graph is a natural and more meaningful way to deal with such data. In order to apply the outlier detection principles for mining network/graph data, one needs to consider benchmark graph data sets from a suitable repository, such as the Stanford Network Analysis Platform (SNAP). A graph data set can be captured in two fundamental ways: adjacency matrix or adjacency list. More recent representation followed for capturing large graphs is known as *edge file*, which contains a list of $< source, destination >$ node pairs indicating various edges constituting the graph.

Some of the methods directly deal with the edge file of a graph and produce the corresponding results. When it is required to perform some algebraic transformations on the input graph, the edge file needs to be transformed to a matrix representation, typically the adjacency matrix of the graph. Other aspects requiring some attention in this context are the direction of edges, the weight of an edge, and the labels associated with nodes/edges of the given graph. One has to explicitly specify the guiding policy on dealing with self loops and parallel edges (multiple edges between the same pair of nodes) in their implementation of the graph manipulation algorithm. One other issue that is of concern to many in this field is regarding node and edge identifiers used to refer to various nodes and edges of a graph. It is often convenient to refer to the nodes/edges using some continuous valued identifier rather than a random valued one.

4.5 Performance Measurement

Let us consider outlier detection as a classification problem with two classes: outliers (**p**) and normal objects (**n**). Also, consider the hypothesized classes as: outliers (**p***) and normal objects (**n***). If an object is positive and it is classified as positive by the algorithm then it is counted as a true positive. Similarly, other three parts of the two-by-two *confusion matrix* (also called a *contingency table*) shown in Fig. 4.2 can also be described.

As a standard practice, the following mathematical notation is followed throughout this book for measuring the performance of any outlier detection algorithm.

- P = Number of positive (outlier) objects in a data set.
- N = Number of negative (normal) objects in a data set.
- P^* = Number of objects hypothesized as positives (outliers) by the algorithm.
- N^* = Number of objects hypothesized as negatives (normal) by the algorithm.

The performance of a binary classifier can be characterized using the measures furnished in Table 4.1.

As already discussed, outlier detection is a binary classification task with serious class imbalance. This is because only a minority (about 5% of the total objects) of the objects are outliers in a typical data set. In view of this large class skew, the entirety of outlier class distribution is typically ‘swallowed’ by the tail of the normal class distribution as shown in Fig. 4.3. Thus, this skew virtually eliminates the ability to bring out accurate predictions.

Fig. 4.2 Confusion matrix associated with a classifier

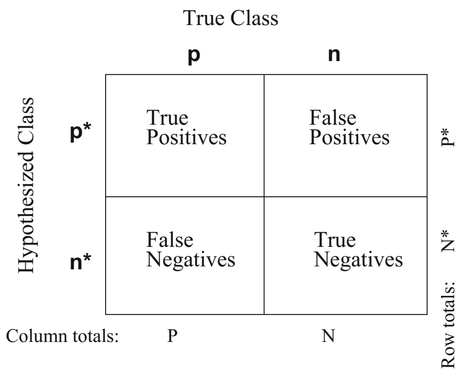


Table 4.1 Performance measures for a binary classifier

1. True Positives (TP)	= Number of actual positive objects (outliers) present among the hypothesized positives
2. False Positives (FP)	$= P^* - TP$
3. False Negatives (FN)	$= P - TP$
4. True Negatives (TN)	$= N - FP$
5. TP Rate (TPR)	$= \frac{TP}{P}$ (also called *recall* or *hit rate*)
6. FP Rate (FPR)	$= \frac{FP}{N}$
7. FN Rate (FNR)	$= \frac{FN}{P}$
8. TN Rate (TNR)	$= \frac{TN}{N}$
9. Precision	$= \frac{TP}{TP+FP}$
10. Recall	$= \frac{TP}{P}$ (same as TPR)
11. Accuracy	$= \frac{TP+TN}{P+N}$
12. F-measure	$= \frac{2}{1/precision+1/recall}$
13. Sensitivity	$= recall$
14. Specificity	$= \frac{TN}{FP+TN}$ $= 1 - FPR$

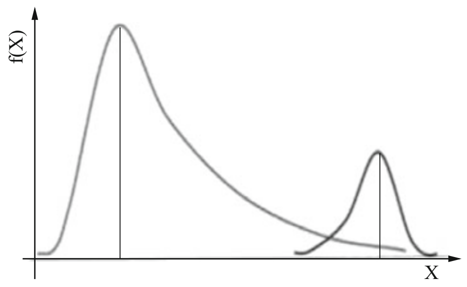


Fig. 4.3 A sample 2-class data distribution with large class skew

So, it is important to reveal the false positive rate also along with the detection rate when reporting the performance of an outlier detection algorithm. A way of doing this is to visualize the performance using the Receiver Operating Characteristic (ROC) curve, which is a non-parametric performance measure. The ROC curve is an interesting measure as it decouples the classifier performance from class skew.

The ROC curve corresponding to a classifier can be generated using the True Positive Rate (TPR) (also known as the Detection rate in this context) and the False Positive Rate (FPR) measures as depicted in Fig. 4.4. An ROC curve depicts relative trade offs between benefits and costs. More specifically, one point in ROC space is better than another if it is to the top-left (TPR is higher, FPR is lower, or both) of the first. The diagonal line $TPR = FPR$ represents the strategy of randomly guessing a class. A classifier that produces a point in the lower right triangle can be negated to result in a point in the upper left triangle.

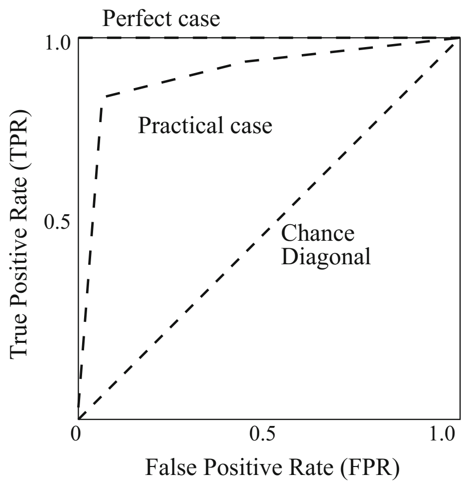


Fig. 4.4 A sample plot with various ROC curves

Other known measures of reporting the performance of a classification algorithm include Area Under the ROC Curve (AUC) and Precision-Recall Curve (PRC), etc. AUC is typically used for quantifying the ROC curve and PRC is employed whenever ROC is found not worthy for some application specific reasons. In such situations, the area under the PRC curve is considered as an evaluation statistic rather than the area under the ROC curve. Another interesting performance measure is the Equal Error Rate (EER). As per the definition, the EER value of a recognition system corresponds to the error value at which the False Positive Rate (FPR) and the False Negative Rate (FNR) of that system are equal.

4.5.1 Comparison of Classifiers

Measuring the generalization ability of a classifier is important to assess if it can perform sufficiently enough for practical use. When there are multiple classifiers available on the same learning task, preference will be given to the one with superior performance. This requires to compare the classifiers for understanding their relative performance levels for the specific learning task.

4.5.1.1 Cross Validation

The general practice when building classifiers is to split the labeled data (of size n objects) into two parts. First part, called as training set, is used for establishing the classifier. Second part, referred to as test/validation set, is used to assess the learning capability. A more practical approach is to carry out k-fold cross-validation by randomly dividing the labeled data into k disjoint subsets of each having $\frac{n}{k}$ objects. Now, learning task is repeated k times, each time holding one of the subsets out for validation purpose and training the classifier with all the remaining data. Accordingly, learning performance is measured by taking the average value over k runs. The recommended size for validation set is $0.1 * n$, which gives it the name 10-fold cross validation.

4.6 Summary

The following salient aspects of the outlier detection problem are discussed here as relevant for bringing out preliminary experimental studies on the same.

- Firstly, the computational notation followed in this book along with some definitions are presented by describing various dissimilarity/ distance measures defined over categorical data.
- Then, the basics of matrix algebra and information theory are furnished as they are the building blocks of the computational model followed here.
- Next, the best practices followed for preparing various benchmark data sets are discussed. This includes both relational data sets as well as graph data sets, required for experimenting with the algorithms covered in this book.
- A Few general observations and some common concerns when dealing with graph data sets are brought out here.
- Finally, the standard procedure for measuring the performance of a binary classifier is presented. The details on generating ROC curves for reporting the results attained using various outlier detection algorithms are also included.

4.7 Bibliographic Notes

Detection of outliers/anomalies assumes significance in several real life applications such as credit card fraud in financial networks, extremely cross-disciplinary authors in an author-paper network, intrusions in a computer network, unexpected planetary movements in astronomical data [4, 6, 15, 18], etc.

Due to various research issues posed by categorical attributes describing a data set, carrying out anomaly/outlier detection on such data turns out to be challenging [2, 5]. In this context, the overlap metric [11] is a natural choice to measure the distance between a pair of categorical data objects. However, a drawback of the overlap measure is that it doesn't distinguish between the values taken by a categorical attribute. Hence, defining data driven similarity/ distance measures based on the frequency distribution of different attribute values will be more useful. From a practical point of view, it is felt that a real valued distance metric is often preferred over a Boolean one [7]. To this effect, the material presented in [3] made a thorough exploration of various data-driven similarity measures pertaining to categorical data.

Subsequently, a new dissimilarity measure suitable for clustering tasks was proposed in [17] by incorporating the attribute value frequencies. Value Difference Metric (VDM) was introduced in [19] to deal with supervised learning problems involving categorical data with class labels.

A method for generating a specific benchmark data set with vital characteristics was presented in [9] for the anomaly/outlier detection problem. This method considers four problem dimensions and assigns suitable values to each one for generating necessary data. As pointed out in [13], outlier detection is a binary classification task with serious class imbalance. To reflect this fact in the data preparation process aimed at outlier detection, it is required to induce imbalance in the distribution of normal vs outlier objects during the data preprocessing stage [12].

Typically, the effectiveness of an outlier detection algorithm can be tested by employing the procedure suggested in [1]. It can also be assessed by visualizing the performance using the Receiver Operating Characteristic (ROC) curve [10], which is a non-parametric performance measure. Alternatively, one can consider Area Under the ROC Curve (AUC) and Precision-Recall Curve (PRC) [8] and also Equal Error Rate (EER) [14] for measuring the performance of classification algorithms. With regard to carrying out anomaly detection on network/graph data, one can consider benchmark graph data sets provided by the Stanford Network Analysis Platform (SNAP) repository [16].

References

1. Aggarwal, C.C., Yu, P.S.: Outlier detection for high dimensional data. In: ACM SIGMOD International Conference on Management of Data, pp. 37–46. Santa Barbara, USA (2001)
2. Bock, H.H.: The classical data situation. In: Analysis of Symbolic Data, pp. 139–152. Springer (2002)
3. Boriah, S., Chandola, V., Kumar, V.: Similarity measures for categorical data: a comparative evaluation. In: SIAM International Conference on Data Mining, Atlanta, Georgia, USA, pp. 243–254 (2008)
4. Chandola, V., Banerjee, A., Kumar, V.: Anomaly detection: a survey. ACM Comput. Surv. **41**(3) (2009)
5. Chandola, V., Boriah, S., Kumar, V.: A framework for exploring categorical data. In: SDM, pp. 187–198. SIAM (2009)
6. Chaudhary, A., Szalay, A.S., Szalay, E.S., Moore, A.W.: Very fast outlier detection in large multidimensional data sets. In: ACM SIGMOD Workshop in Research Issues in Data Mining and Knowledge Discovery, pp. 45–52 (2002)
7. Cheng, V., Li, C.H., Kwok, J., Li, C.K.: Dissimilarity learning for nominal data. Patten Recognit. **37**(7), 1471–1477 (2004)
8. Davis, J., Goadrich, M.: The relationship between precision-recall and ROC curves. In: 23rd International Conference on Machine Learning (ICML), pp. 30–38 (2006)
9. Emmott, A.F., Das, S., Deitterich, T., Fern, A., Wong, W.K.: Systematic construction of anomaly detection benchmarks from real data. In: KDD Workshop on Outlier Detection and Description. ACM, Chicago, IL, USA (2013)
10. Fawcett, T.: An introduction to ROC analysis. Pattern Recognit. Lett. **27**, 861–874 (2006)

11. Gower, J.C., Legendre, P.: Metric and euclidean properties of dissimilarity coefficients. J. Classif. **3**(1), 5–48 (1986)
12. Harkins, S., He, H., Williams, G.J., Baxter, R.A.: Outlier detection using replicator neural networks. In: Kambayashi, Y., Winiwarter, W., Arikawa, M. (eds.) 4th International Conference on Data Warehousing and Knowledge Discovery (DaWak). LNCS, vol. 2454, pp. 170–180. Springer, Aixen-Provence, France (2002)
13. Hido, S., Tsuboi, Y., Kashima, H., Sugiyama, M., Kanamori, T.: Statistical outlier detection using direct density ratio estimation. Knowl. Inf. Syst. **26**(2), 309–336 (2011)
14. Jain, A.K., Duin, R.P.W., Mao, J.: Statistical pattern recognition: a review. IEEE Trans. Pattern Anal Mach. Intell. **22**, 4–37 (2000)
15. Lazarevic, A., Ertoz, L., Kumar, V., Ozgur, A., Srivastava, J.: A comparative study of anomaly detection schemes in network intrusion detection. In: SIAM International Conference on Data Mining (2003)
16. Leskovec, J., Krevl, A.: SNAP datasets: stanford large network dataset collection (2014). http://snap.stanford.edu/data
17. Ng, M.K., Li, M.J., Huang, J.Z., He, Z.: On the impact of dissimilarity measure in k-modes clustering algorithm. IEEE Trans. Pattern Anal. Mach. Intell. **29**(3), 503–507 (2007)
18. Sithirasenan, E., Muthukkumarasamy, V.: Substantiating security threats using group outlier detection techniques. In: IEEE GLOBECOM, pp. 2179–2184 (2008)
19. Stanfill, C., Waltz, D.: Towards memory-based reasoning. Commun. ACM **29**(12), 1213–1228 (1986)

Chapter 5
Outlier Detection in Categorical Data

Abstract This chapter delves on a specific research issue connected with outlier detection problem, namely *type of data attributes*. More specifically, the case of analyzing data described using categorical attributes/features is presented here. It is known that the performance of a detection algorithm directly depends on the way outliers are perceived. Typically, categorical data are processed by considering the occurrence frequencies of various attributes values. Accordingly, the objective here is to characterize the deviating nature of data objects with respect to individual attributes as well as in the joint distribution of two or more attributes. This can be achieved by defining the measure of deviation in terms of the attribute value frequencies. Also, cluster analysis provides valuable insights on the inherent grouping structure of the data that helps in identifying the deviating objects. Based on this understanding, this chapter presents algorithms developed for detection of outliers in categorical data.

5.1 Introduction

As stated earlier, the aim of this study is to identify data objects that deviate significantly from the rest of the data. Detection of such deviating objects or rare objects and exceptions in data is the purpose of outlier analysis. This has numerous applications in various domains such as fraud detection in financial transactions and intrusion detection in computer networks, etc.

The type of attributes describing data is a key issue among the ones identified connected with outlier detection. More specifically, data described using *qualitative (categorical) attributes* is the focus of this study. The importance of dealing with categorical attributes is evident from the observation that data pertaining to several real world applications are described using such attributes. This emphasizes the need for developing suitable algorithms for detecting outliers in categorical data.

N. N. R Ranga Suri et al., *Outlier Detection: Techniques and Applications*, Intelligent Systems Reference Library 155,
https://doi.org/10.1007/978-3-030-05127-3_5

5.1.1 Categorical Attributes

The scale of measure indicates the nature of information within the values assigned to variables/attributes. There are four scales of measurement popularly used: nominal, ordinal, interval and ratio.

Nominal scale differentiates various values based only on their names. Examples of this type are color, direction, language, etc. Mathematical operations permitted on this type of values and their central tendency are listed in Table 5.1.

Ordinal scale allows to rank order the values to produce a sorted sequence. It includes dichotomous values (such as 'true', 'false') and also non-dichotomous data comprising a spectrum of values (such as 'excellent', 'very good', 'good', 'satisfied', 'poor' for expressing one's opinion).

Interval scale allows for measuring the degree of difference between two values. For example, location information of places indicated using Cartesian coordinates can be used to determine which place is closer to a specified place. Mode, median and arithmetic mean are the central tendency measures applicable on this type of data.

Ratio scale measures values by estimating the ratio between a magnitude of a continuous quantity and a basic unit of the same type. All statistical measures can be computed on this type of data as necessary mathematical operations are defined for this data, as indicated in Table 5.1.

A categorical variable/attribute, belonging to the nominal scale, takes one of a limited number of possible values. The central tendency of the values of a categorical variable is given by its mode. Though numeric values may appear corresponding to a categorical variable, they each represent a logically separate concept and cannot be processed as numbers. In computer science, categorical variables are referred to as enumerated types. The term categorical data applies to data sets that contain one or more categorical variables. Categorical data are also known as nominal or qualitative multi-scale data in different application contexts.

A categorical variable that can take on exactly two values is known as dichotomous (binary) variable. The ones with more than two possible values are called as polytomous variables. Regression analysis on categorical variables is accomplished through multinomial logistic regression.

The acceptable values of a categorical (qualitative) attribute are represented by various categories, as illustrated in Table 5.2. The information on the occurrence

Table 5.1 Details on various scales of measure

Scale	Property	Mathematical operations	Central tendency
Nominal	Membership	$=, \neq$	Mode
Ordinal	Comparison	$>, <$	Median
Interval	Difference	$+, -$	Mean
Ratio	Magnitude	x, /	Geometric mean

Table 5.2 Example categorical attributes with sample values

Attribute name	Categories (acceptable values)
Color	Green, red, blue, …
Direction	North, east, west, south
Weather	Sunny, cloudy, raining, …
…	…

frequencies of various categories of a categorical attribute in data is demanded by many data-dependent tasks such as outlier detection.

As already known, the characteristics of a categorical data set can be summarized using the following details.

1. Size of the data set (n)
2. Number of attributes (m)
3. Number of values taken by each attribute ($|DOM(A_r)|, \forall r$)
4. Frequencies of various values of each categorical attribute ($freq(x_{i,r}), \forall i, r$)

It is possible to define any computational method on categorical attributes using the above abstract model.

The simplest way to find similarity between two categorical attributes is to employ the *overlap* measure (Eq. 4.1). This measure treats two categorical attributes as similar if their values are identical, and dissimilar otherwise. A drawback with this measure is that it doesn't distinguish between the different values taken by an attribute. All matches, as well as mis-matches are treated as equal.

Referring to the sample data set shown in Table 5.3, it is obvious that certain combination of the values of the categorical attributes $\{Model, Color\}$ describing the data are more frequent than the others. Bringing in the frequency details into account, it is possible to determine which of the objects are more similar and also the degree of their similarity.

Table 5.3 Frequency distribution of a sample 2-D car data (for illustration)

Model/Color	Black	White	Grey
Maruti-Swift	20	35	45
Tata-Sumo	10	75	15
Honda-City	40	25	35
Toyota-Corolla	35	25	40
Volkswagen-Polo	25	20	55

5.1.2 Challenges with Categorical Data

Though there exist a number of methods for outlier detection in numerical data, only a limited number of them can process the data represented using categorical attributes. Outlier detection in categorical data is an evolving problem due to the computational challenges posed by the categorical attributes/features. The fundamental issue in this regard is the difficulty in defining a proximity measure over the categorical values. This is due to the fact that the various values that a categorical variable can assume are not inherently ordered. As a result, many data mining tasks such as determining the nearest neighbor of a categorical object turn out to be non-trivial. Active research happening in this direction is indicative of the importance associated with this aspect. Depending on the application context, supervised measures determine similarity based on class information, while data-driven measures determine the same based on the data distribution.

Many a time, when dealing with data having categorical attributes, it is assumed that the categorical attributes could be easily mapped into numeric values. However, there are instances of categorical attributes, where mapping to numerical attributes is not a straightforward process, and the results greatly depend on the mapping that is used. Consequently, methods such as those based on distance or density measurements are not acceptable, requiring necessary modification in their formulations.

5.2 Clustering Categorical Data

Clustering is an important data mining task required in numerous real life applications. Due to the difficulties in analyzing categorical data, as discussed in the previous section, performing cluster analysis on such data turns out to be an involved activity. However, as an unsupervised learning task, it helps in understanding the inherent grouping structure of the data for the purpose of outlier detection. In that sense, cluster analysis assumes significance towards detecting outliers in the data.

Prior research in this direction has brought out some interesting algorithms for clustering categorical data. A glimpse of some of these algorithms is provided below to facilitate effective data analysis.

5.2.1 ROCK Algorithm

RObust Clustering using linKs (ROCK) algorithm works based on the notion of links between data objects, rather than applying any metric to measure the similarity. The number of links between a pair of objects is given by the number of common neighbors of the objects. The rationale is that data objects belonging to single cluster will in general have a large number of common neighbors, and consequently more links. Therefore, merging clusters/objects with the most number of links first during clustering process will result in more meaningful clusters.

The neighbors of a data object are those that are considerably similar to it. Measuring similarity could be done using one of the well-known distance metrics (like L_1, L_2) or even non-metric (a distance/similarity function provided by a domain expert).

Consider a data set $D = \{X_1, X_2, \ldots, X_n\}$ consisting of n objects. Let $link(X_i, X_j)$ represent the number of common neighbors between two objects X_i and X_j. Since the idea is to have each cluster with a high degree of connectivity, it is required to maximize the sum of $link(X_q, X_r)$ for data object pairs (X_q, X_r) belonging to a single cluster and at the same time minimize the sum of $link(X_q, X_s)$ for (X_q, X_s) in different clusters. This leads to the following criterion function to be maximized over k clusters.

$$E_t = \sum_{i=1}^{k} n_i * \sum_{X_q, X_r \in C_i} \frac{link(X_q, X_r)}{n_i^{1+2f(\theta)}} \tag{5.1}$$

Here, C_i denotes cluster i having n_i data objects. It is to be noted that the total number of links involving a pair of objects in cluster C_i is divided by the expected total number of links in C_i, to prevent all data objects getting assigned to a single cluster.

Similarly, for a pair of clusters (C_i, C_j), the number of cross links between them $link[C_i, C_j]$ is determined using the expression given below.

$$link[C_i, C_j] = \sum_{p_q \in C_i, p_r \in C_j} link(X_q, X_r) \tag{5.2}$$

Then, the goodness measure $g(C_i, C_j)$ for merging clusters C_i, C_j is computed as

$$g(C_i, C_j) = \frac{link[C_i, C_j]}{(n_i + n_j)^{1+2f(\theta)} - n_i^{1+2f(\theta)} - n_j^{1+2f(\theta)}} \tag{5.3}$$

The denominator in the above equation represents the expected number of cross links or links between pairs of objects each from a different cluster. The pair of clusters for which the goodness measure is maximum is the best pair to be merged at any given step.

According to this method, the larger the number of links between a pair objects, the greater is the likelihood that they belong to the same cluster. Thus, clustering using links injects global knowledge into the clustering process and is thus more robust. This is in contrast to the similarity-based local approach taking into account the characteristics of the data objects in pairs.

ROCK is a hierarchical clustering algorithm that accepts as input the set S of n sampled data objects to be clustered (that are drawn randomly from the original data set), and the desired number of clusters, k. Subsequently, the clusters obtained with the sampled data are used to assign the rest of the data objects to relevant clusters.

ROCK algorithm has a worst-case time complexity of $O(n^2 + nm_m m_a + n^2 log(n))$. Here, m_m is the maximum number of neighbors, m_a is the average number of neighbors and n is the number of data objects. Similarly, the space complexity of this algorithm is $O(min\{n^2, nm_m m_a\})$.

5.2.2 Squeezer Algorithm

Squeezer algorithm performs clustering of categorical data by reading the data objects one by one. It assigns the first object to the initial cluster. Subsequently, further objects are either put into existing clusters or assigned to a new cluster depending on their similarity to the existing clusters. As a result, it doesn't require the number of desired clusters as an input parameter.

Let $D = \{X_1, X_2, \ldots, X_n\}$ be a data set having n data objects described using m categorical attributes $\{A_1, A_2, \ldots, A_m\}$. Given a cluster C_j, the set of values of the attribute A_i with respect to C_j is denoted as $VAL_i(C_j)$. Also, the support of a value a_i of the attribute A_i with respect to cluster C_j, denoted as $Sup(a_i)$, is the number of objects in C_j having the value a_i corresponding to the attribute A_i.

This algorithm determines the summary of every cluster consisting of m element pairs of attribute values and their corresponding supports.

$$Sum(C_j) = \{VS_i | 1 \leq i \leq m\} \text{where } VS_i = \{(a_i, Sup(a_i)) | a_i \in VAL_i(C_j)\}. \quad (5.4)$$

Summary of a cluster is the data structure used to compute the similarity between a data object and a cluster. A similarity threshold s is used to determine whether a data object is to be put into an existing cluster or assigned to a new cluster.

In the worst case, Squeezer algorithm has time complexity of $O(nkpm)$ and space complexity of $O(n + kpm)$. Here, k is the final number of clusters formed and p is the distinct values of a categorical attribute. It is assumed that every attribute has the same number of distinct attribute values, for simplification.

This algorithm is highly efficient for disk resident data sets as it makes only one scan over the data set for clustering the entire data. So, it is suitable for clustering data streams.

5.2.3 k-ANMI Algorithm

The k-ANMI algorithm is a k-means like clustering algorithm for categorical data. It directly optimizes the mutual information sharing based objective function. The goodness of clustering in each step is evaluated using Average Normalized Mutual Information (ANMI) measure, borrowed from cluster ensemble.

In information theory, mutual information is a symmetric measure to quantify the statistical information shared between two distributions. Let A and B be the random variables described by the cluster labeling $\lambda^{(a)}$ and $\lambda^{(b)}$, with $k^{(a)}$ and $k^{(b)}$ groups respectively. Let $I(A, B)$ denote the mutual information between A and B, and $H(A)$ denote the entropy of A. The [0, 1]-normalized mutual information between A and B is given by

$$NMI(A, B) = \frac{2I(A, B)}{H(A) + H(B)} \tag{5.5}$$

Let $n^{(h)}$ be the number of objects in cluster C_h according to $\lambda^{(a)}$, and let n_g be the number of objects in cluster C_g according to $\lambda^{(b)}$. Let $n_g^{(h)}$ be the number of objects in cluster C_h according to $\lambda^{(a)}$ as well as in cluster C_g according to $\lambda^{(b)}$. The [0, 1]-normalized mutual information criteria $\phi^{(NMI)}$ is computed as follows.

$$\phi^{(NMI)}(\lambda^{(a)}, \lambda^{(b)}) = \frac{2}{n} \sum_{h=1}^{k^{(a)}} \sum_{g=1}^{k^{(b)}} n_g^{(h)} log_{k^{(a)}k^{(b)}} \frac{n_g^{(h)} n}{n^{(h)} n_g} \tag{5.6}$$

Therefore, the average normalized mutual information (ANMI) between a set of m labellings, Λ, and a labeling $\bar{\lambda}$ is defined as follows.

$$\phi^{(NMI)}(\Lambda, \bar{\lambda}) = \frac{1}{m} \sum_{q=1}^{m} \phi^{(NMI)}(\bar{\lambda}, \lambda^{(q)}) \tag{5.7}$$

The optimal combined clustering $\lambda^{(k-opt)}$ will be the one that has the maximal average mutual information with all individual partitioning $\lambda^{(q)}$ given that the number of consensus clusters desired is k. Therefore, the optimal clustering is computed as

$$\lambda^{(k-opt)} = arg\ max_{\bar{\lambda}} \sum_{q=1}^{m} \phi^{(NMI)}(\bar{\lambda}, \lambda^{(q)}) \tag{5.8}$$

where $\bar{\lambda}$ goes through all possible k-partitions.

k-ANMI algorithm takes the number of desired clusters (say k) as input and iteratively changes the cluster label of each data object to improve the value of the objective function. For each object, the current label is changed to each of the other $(k - 1)$ possible labels and the ANMI objective is re-computed. If the ANMI value increases, the object's label is changed to the best new value and the algorithm proceeds with next object. A sweep over the data gets completed when all objects are checked for possible label changes. If at least one label change happens in the current sweep, a new sweep is initiated. The algorithm terminates when a local optimum is reached for the objective function.

This algorithm is easy to implement, requiring multiple hash tables as the only major data structure. More precisely, for a clustering task with m attributes and k

clusters, it needs $(k+1)m$ has tables. Through the use of these hash tables, ANMI value could be derived without accessing the original data set, for computational efficiency.

This algorithm has time complexity $O(tnk^2mp^2)$ in worst case. Here, t is the number of iterations the algorithm runs and p is the number of values each attribute has. Similarly, the space complexity is $O(mkp + nm)$. It is important to note that the computational complexity of k-ANMI algorithm is linear in both the number of objects (n) and the number of attributes (m). So, it can be employed for clustering large categorical data sets.

5.2.4 *k-modes Algorithm*

The k-modes algorithm extends the k-means paradigm to categorical domain by employing a suitable dissimilarity measure defined over categorical attributes. It replaces means of clusters with modes, and uses a frequency based method to update modes in the clustering process to minimize the associated cost function.

Compared to k-means method, there are three major modifications incorporated with k-modes method to make it work over categorical data.

1. *Dissimilarity measure*: Let X, Y be two data objects described using m categorical attributes $\{A_1, A_2, \ldots, A_m\}$. The dissimilarity between X and Y can be measured by counting the total mis-matches of the corresponding attribute categories as given in Eq. 4.1. Alternatively, dissimilarity can also be measured by taking into account the frequencies of categories in the data set as follows.

$$d_{\chi^2}(X, Y) = \sum_{j=1}^{m} \frac{(n_{x_j} + n_{y_j})}{n_{x_j} n_{y_j}} \delta(x_j, y_j) \tag{5.9}$$

   Here, n_{x_j}, n_{y_j} are the number of objects in the data set that have categories x_j and y_j for the attribute A_j. Note that $d_{\chi^2}(X, Y)$ is called as *chi-square distance*. It gives more importance to rare categories than frequent ones. So, it is useful in finding out the under-represented object clusters such as fraudulent claims in insurance databases.

2. *Mode of a set*: Let $D = \{X_1, X_2, \ldots, X_n\}$ be a data set of n data objects described using m categorical attributes. A mode of D is a vector $Z = \{z_1, z_2, \ldots, z_m\}$ that minimizes the following.

$$dist(D, Z) = \sum_{i=1}^{n} d(X_i, Z) \tag{5.10}$$

   Here, d is the dissimilarity measure as defined above or any other measure defined over categorical attributes. It is important to note that Z is not necessarily an object in D and also the mode of a data set is not unique.

3. Use of frequency-based method to update the cluster modes iteratively.

Let $\{S_1, S_2, \ldots, S_k\}$ be a partition of the data set, where $S_l \neq \phi$ for $1 \leq l \leq k$, and $\{Z_1, Z_2, \ldots, Z_k\}$ be the modes of $\{S_1, S_2, \ldots, S_k\}$ respectively. Then, the overall cost of the partition is given by

$$E = \sum_{l=1}^{k} \sum_{i=1}^{n} y_{i,l} d(X_i, Z_l) \tag{5.11}$$

Here, $y_{i,l}$ is an element of the partition matrix $\mathbf{Y}_{nxl}$ and d is the dissimilarity function.

Based on the three building blocks detailed above, the k-modes algorithm consists of the following steps for performing clustering of categorical data.

1. Select k initial modes, one for each cluster.
2. Allocate an object to the cluster whose mode is the nearest to it according to d. Update the mode of the cluster after each allocation.
3. After all objects are allocated to clusters, re-compute the dissimilarity of objects against the current modes. If an object is found such that its nearest mode belongs to another cluster rather than its current one, re-allocate the object to that cluster and update the mode of both clusters.
4. Repeat step-3 until no object has changed clusters over the entire data.

The initial modes of the algorithm can be selected either by considering the first k distinct objects from the data set or using any other more involved process. The advantage with this algorithm is its scalability to large data sets.

5.2.4.1 Cluster Initialization Method

A suitable cluster initialization ensures smooth convergence of the k-means family of clustering algorithms. Accordingly, a well known cluster initialization method for categorical data is discussed here. It results in a set of k clusters $\{C_1, C_2, \ldots, C_k\}$ with their representatives (modes) denoted as $\{Z_1, Z_2, \ldots, Z_k\}$.

This initialization method starts with the data object with maximum density (see Eq. 4.7) as the first cluster representative. For selecting the remaining $(k - 1)$ elements of the initial cluster representatives set Z, both the density and the distance between objects (see Eq. 4.1) are utilized as per the steps furnished below.

1. Set the first cluster representative, $Z = \{X_i^1\}$, where $X_i^1 \in D$ is the object with the maximum density.
2. For the second cluster representative, $Z = Z \cup \{X_i^2\}$, where $X_i^2 \in D$ satisfies $d(X_i^2, X_m) * density(X_i^2) = max_{i=1}^{|D|}\{d(X_i, X_m) * density(X_i) | X_m \in Z\}$.
3. Similarly, for the kth cluster representative, $Z = Z \cup \{X_i^k\}$, where $X_i^k \in D$ satisfies $d(X_i^k, X_m) * density(X_i^k) = max_{i=1}^{|D|}\{min_{X_m}\{d(X_i, X_m) * density(X_i)\} | X_i \in D\}$.

5.3 A Ranking-Based Characterization of Outliers

Characterizing the deviating nature of data objects in categorical space is paramount to evolve accurate and efficient algorithms for their detection.

Going by the general practice, it is more meaningful to rank the data objects based on their degree of deviation rather than making a binary decision on whether or not an object is an outlier. Also, in many application domains dealing with large data, it is pertinent to identify the set of most likely outliers, as it gives opportunity for carrying out further analysis on the ranked objects. With this view, the characterization presented here leverages ranking concept for determining the set of most likely outliers in a given data set.

5.3.1 Exploring the Categorical Space

It is known that the performance of any outlier detection algorithm directly depends on the way outliers are perceived. So, a definition for outliers in terms of the attribute value frequencies of the categorical data is considered here. This definition relies on the intuition that an outlier has to display its deviating characteristics from the rest of the data in terms of irregular/peculiar value(s) corresponding to one or more of its attributes. Naturally, the frequency count of such an irregular value happens to be much less than that of the regular values. Thus, infrequent attribute values are regarded as manifestations of the presence of outliers in data.

As per the above approach, a data object X in a categorical data set D turns out to be an *outlier* in the following manner.

- *Type-1 Outlier*: The attribute values describing object X are relatively infrequent.
- *Type-2 Outlier*: The combination of the categorical values describing object X is relatively infrequent, though each one of these values are frequent individually.

These scenarios can be illustrated in a hypothetical manner as shown in Fig. 5.1, for a simple data set described using two categorical attributes. Here, data objects are depicted in categorical space based on the distinct values each attribute has in the data. Referring to this figure, object O_1 turns out to be an outlier of *Type-1* as its value corresponding to Attribute-2 is infrequent. On the other hand, though both the attribute values of object O_2 are frequent individually, their combination is not frequent, making it an outlier of *Type-2*. For object O_3, though it qualifies to be of *Type-2*, it is primarily of *Type-1* due to its infrequent attribute values. Hence, no distinction is made between objects O_1 and O_3 in this methodology.

The methods designed based on frequencies of attribute values are good at detecting outliers of Type-1. However, they fail to detect Type-2 outliers due to their characterizing feature. On the other hand, clustering-based methods are capable of detecting outliers of Type-2, but may fail in dealing with Type-1 as these outliers

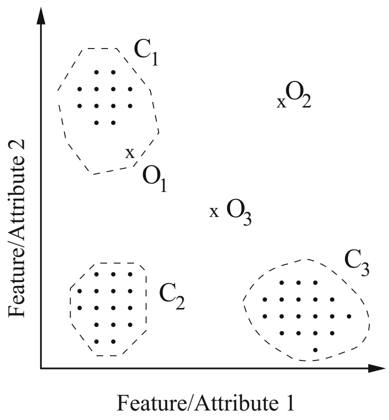


Fig. 5.1 Scenarios of outlier occurrence in categorical space

tend to be part of a valid big cluster due to their proximity to such clusters. For example, object O_1 in Fig. 5.1 may become part of its nearest cluster when clustering is performed. Ideally, an algorithm for outlier detection is expected to identify outliers of both types.

5.3.2 Methodology for Characterization

Consider a data set D comprising n objects described using m categorical attributes. The basic idea is to capture all possible types of outliers considering their occurrence scenarios in the categorical space. Also, such a characterization is to be done based on the frequency counts of the values of categorical attributes describing the data. In this connection, the definition of outliers given above looks more appealing.

In order to determine the most likely set of outliers in the data, a clustering-based methodology is presented here. For the sake of illustration, let us assume a clustering structure of some hypothetical categorical data as shown in Fig. 5.2. To start with, some important definitions constituting this characterization are given below.

1. A cluster C_i is considered as a *big cluster* if it has at least $\alpha\%$ of the number of objects in D. Thus, the set of big clusters BC in a clustering process is given by

$$BC = \{C_i | size(C_i) \geq (\alpha * 0.01 * n)\} \tag{5.12}$$

   where $size(C_i)$ indicates the number objects present in cluster C_i and value of α is determined specific to the data set at hand.
2. The *cluster distance* of an object X_i is measured as the distance between X_i and its closest big cluster as,

$$cdist(X_i) = min_l \ d(Z_l, X_i), \forall C_l \in BC \tag{5.13}$$

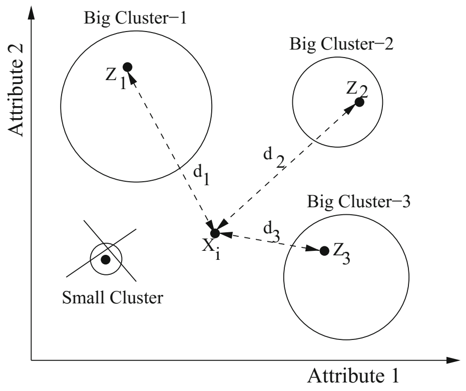


Fig. 5.2 Illustration of cluster distance computation

where Z_l is the representative of cluster C_l and the distance function d refers to the one given in Eq. 4.6.

The computation of cluster distance of an object X_i in a clustering scenario, involving big as well as small clusters, is depicted in Fig. 5.2. In this case, the big cluster with its mode represented by Z_3 turns out to be the nearest big cluster to object X_i, as distance d_3 happens to be the smallest of the distances $\{d_1, d_2, d_3\}$.

The following two aspects define the necessary conceptual details required to determine the most likely outliers in a data set.

1. As per Eq. 4.7, the density of a data object is measured based on the frequencies of its attributes values. One can note that the density of an outlier will be generally low, as the attribute values of such an object tend to be less frequent compared to the values describing normal ones. Thus, density can be looked at as a measure of the relative frequency of the values of a data object. The less the density is, the more likely the object gets detected as outlier. Based on this understanding, the data objects can be arranged in ascending order of their density values, with the least density object displaying the most deviating characteristics. Such a sequence of objects is called as *frequency-based ranking* of data.
2. Similarly, it is possible to identify small clusters possibly having outlier objects that tend to locate away from big clusters, as shown in Eq. 5.12. Therefore, the distance between a data object and its closest big cluster is an indicative measure of its deviating behavior. The more this distance is, the more likely the object gets detected as outlier. Based on this rationale, the *clustering-based ranking* of data objects is determined. More precisely, clustering-based ranking is obtained by arranging the objects in decreasing order of their cluster distance values.

According to the above two ranking schemes, each object $X \in D$ gets its corresponding ranks based on its density and its cluster distance values. Given that, outliers of Type-1 can be detected using the frequency-based ranking, and Type-2 using the clustering-based ranking. Now, a consolidated set of outliers of both types, designed as the *likely set* (LS), can be determined using these two ranking schemes.

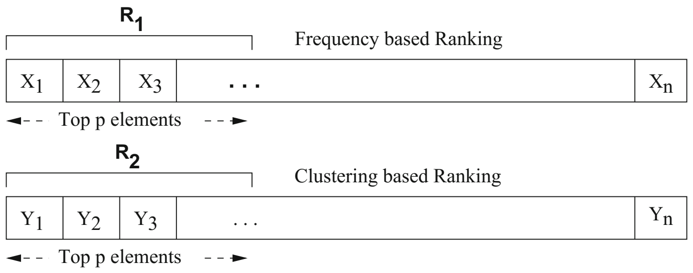


Fig. 5.3 Determining the likely set of outliers in data

To determine p most significant outliers in data, consider the set of indices of p top ranked objects in the frequency-based ranking as R_1. Similarly, the set of indices of p top ranked objects in the clustering-based ranking is denoted as R_2, as shown in Fig. 5.3. Now, the likely set of outliers is constructed as, $LS = R_1 \cup R_2$. Though there are several possible ways of fusing two rank lists, simple set union can produce the likely set satisfactorily.

5.4 Algorithms for Detecting Outliers

As stated earlier, most of the methods for outlier detection are designed to deal with data sets described by numerical attributes, or ordinal attributes that can be directly mapped into acceptable numerical values. Only a limited number of the approaches can process categorical data. This is due to the challenges posed by the categorical attributes in defining requisite measures/metrics for exploring the data. However, with emphasis on this aspect of the outlier detection problem, there are some interesting methods employing different detection strategies. Some of the well known approaches among them for outlier detection in categorical data are discussed below.

5.4.1 Greedy Algorithm

Conventional approaches for exploratory data mining applications do not handle categorical data in a satisfactory manner. To address this gap, the problem of outlier detection in categorical data is defined as an optimization problem in terms of finding a sub-set of k objects such that the expected entropy of the resultant data set after the removal of this sub-set is minimized.

However, an exhaustive search through all possible solutions with k outliers for the one with the minimum objective value is costly since there are (n, k) possible solutions with n objects and k outliers. So, a fast greedy algorithm for mining outliers under the same optimization model is presented here.

For a discrete random variable X with $S(X)$ being the set of values that it can take, $p(x)$ being the probability distribution function, the entropy of X can be computed using Eq. 4.22.

Given a data set D of n objects $\{X_1, X_2, \ldots, X_n\}$, where each object is described using m categorical attributes $\{f_1, f_2, \ldots, f_m\}$, then the entropy of D can be computed as

$$E(D) = E(f_1) + E(f_2) + E(f_3) + \cdots + E(f_m) \tag{5.14}$$

Here, $E(f_j)$ indicates the entropy value (as in Eq. 4.22) corresponding to the attribute/feature f_j of the data set D. The above expression holds good with the assumption of independent attributes.

Given an integer k, the idea is to determine a subset $D^{'} \subset D$ with size k, in such a way that the entropy of the resulting set $(D - D^{'})$ is minimized. According to this idea, a data object with maximum contribution to the entropy value gets labeled as the first outlier. It goes on identifying the outliers in successive iterations at the rate of one outlier per iteration. Hence, this algorithm requires a number of scans over the data set resulting in a time complexity of $O(nkm)$.

5.4.2 AVF Algorithm

Attribute Value Frequency (AVF) algorithm is a fast and scalable outlier detection strategy for categorical data. It is developed based on the intuition that outliers are those objects that are infrequent in the data set. Moreover, an ideal outlier object in a categorical data set is one whose each and every attribute value is extremely irregular (or infrequent). The infrequent-ness of an attribute value can be determined by counting the number of times it appears in the data set.

Considering the above discussion, the AVF method works based on the frequency counts of the attribute values. Given a data set with n data objects and m categorical attributes, this method computes the frequency-based score of an object X_i as

$$AVFScore(X_i) = \frac{1}{m} \sum_{j=1}^{m} Freq(x_{ij}) \tag{5.15}$$

where $Freq(x_{ij})$ is the number of times the jth attribute/feature value of the object X_i appears in the data set.

As per this method, objects with low AVF-scores are considered as outliers. Once the scores of all the data objects are calculated, one can designate the k objects with the smallest AVF-scores as outliers.

The computational complexity of this algorithm is $O(nm)$. It scales linearly with the number of data objects and attributes, and works with a single scan over the data set. It does not need to create or search through different combinations of attribute values.

5.4.3 ROAD Algorithm

As per the characterization of outliers presented in the previous section, the computational flow of a ranking-based algorithm for unsupervised detection of outliers is shown in Fig. 5.4. Basically, this algorithm performs its computations in two phases. In the first phase, it computes the object density values and also explores a clustering of the data. Using the resultant clustering structure, the set of big clusters is identified in order to determine the distance between various data objects and their corresponding nearest big clusters. In the second phase, frequency-based rank and clustering-based rank of each data object are determined. Subsequently, a unified set of the most likely outliers is determined using these two individual rankings. Thus, it is named as Ranking-based Outlier Analysis and Detection (ROAD) algorithm.

Clustering categorical data is an important computational step of the ROAD algorithm, as it is meant for identifying Type-2 outliers which are relatively more difficult to detect when compared to Type-1 outliers. This is because, Type-2 outliers display their deviation in the joint distribution of two or more attributes. Accordingly, the

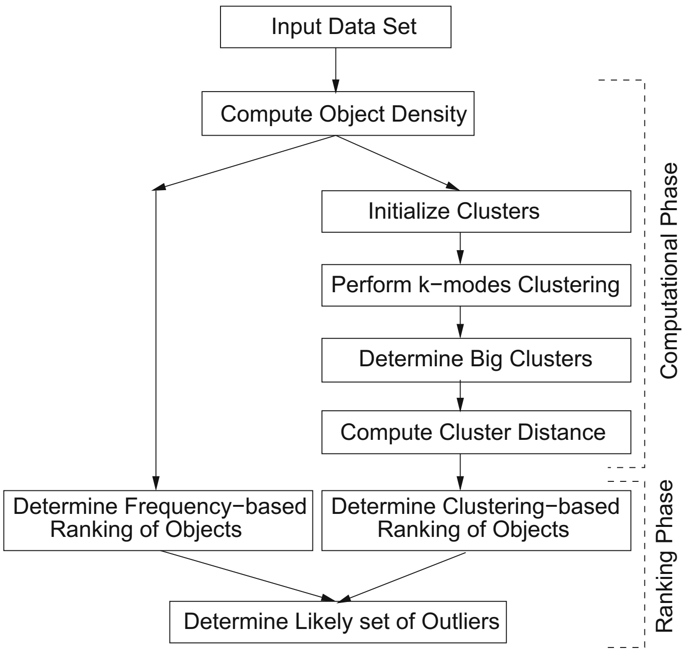

Input Data Set
Compute Object Density
Initialize Clusters
Perform k−modes Clustering
Determine Big Clusters
Compute Cluster Distance
Determine Frequency−based Ranking of Objects
Determine Clustering−based Ranking of Objects
Determine Likely set of Outliers
Computational Phase
Ranking Phase

Fig. 5.4 Computational flow of ROAD algorithm

effectiveness of the specific clustering method employed for this purpose determines the detection performance.

In practice, one can use any established method like ROCK algorithm for clustering categorical data. However, it is recommended to use the *k-modes* method due to its efficiency as a member of the *k-means* family. Moreover, the cluster initialization method presented in Sect. 5.2.4.1 makes it more effective. Hence, using the k-modes algorithm along with the categorical dissimilarity measure given in Eq. 4.6 is expected to exhibit better results.

The data processing steps constituting the ROAD algorithm are furnished below. The initial five steps constitute the first phase known as 'computational phase' as it primarily computes various quantities on the basis of the input data. The later three steps correspond to the second phase called as 'ranking phase', which determines the rank of each data object corresponding to each ranking scheme based on the quantities computed in the previous phase.

1. Compute $density(X_i)$ of each data object $X_i \in D$ using Eq. 4.7.
2. Determine the initial set $\{Z_1, Z_2, \ldots, Z_k\}$ of k cluster representatives.
3. Perform k-modes clustering on D with the distance measure in Eq. 4.6.
4. Determine the set of big clusters BC as given in Eq. 5.12.
5. For each object X_i, determine its cluster distance $cdist(X_i)$ as per Eq. 5.13.
6. Determine the frequency-based rank $freq\_rank(X_i)$ of each object $X_i \in D$.
7. Determine the clustering-based rank $clust\_rank(X_i)$ of each object $X_i \in D$.
8. Construct the likely set LS using the two ranked sequences, for a given p value.
9. Output the the set LS of most likely outliers identified.

Considering various simplifications on the algorithmic steps, the computational complexity of ROAD algorithm turns out to be $O(nm + nlog(n))$. It is important to note that this complexity is not affected by the number of outliers to be found.

5.5 Experiments on Benchmark Data

Experimental study of the outlier detection algorithms on benchmark data includes six frequently used categorical data sets taken from UCI ML Repository. Each data set consists of labeled instances belonging to two different classes. As per the standard procedure explained in Sect. 4.2.2, objects with missing attribute values are removed and objects belonging to small sized class in every data set are considered as outliers. Though these designated outliers are not outliers in real sense, they are considered so for experimental purpose. In order to impose imbalance in the number of objects belonging to normal and outlier categories, only a selected sub-set of the objects belonging to outlier class are considered, by taking every fifth object of the designated outlier class. Table 5.4 summarizes the benchmark categorical data sets considered in this investigation.

Table 5.4 Details of benchmark categorical data sets

Name of the data set	Dimension	Outlier class label	# Outlier objects	# Normal objects	# Total objects
Chess (End-Game)	36	Nowin	305	1669	1974
Tic-Tac-Toe	9	Negative	66	626	692
Breast Cancer (W)	10	4 (malignant)	47	444	491
Congressional Votes	16	Republican	21	124	145
Breast Cancer	9	Recurrence-events	16	196	212
Mushroom	22	p (poisonous)	783	4208	4991

As the algorithms considered for this study work in unsupervised learning mode, they do not require labeled data. However, class labels are used to assess their performance in mining outliers. A comparative view of the performance of these methods is also presented here.

As described in Sect. 5.1, a categorical attribute A_i can take a set of discrete values (categories) represented by $DOM\,(A_i)$. The cardinality of the attribute domains of 9 categorical attributes describing the Breast Cancer data set are shown in Table 5.5 for a better understanding.

Table 5.5 Categorical attributes describing Breast Cancer data

Sl.No.	Attribute name	Arity
1	Age	9
2	Menopause	3
3	Tumor-size	12
4	Inv-nodes	13
5	Node-caps	2
6	Deg-malig	3
7	Breast	2
8	Breast-quad	5
9	Irradiat	2

5.5.1 Performance Measurement

To measure the detection accuracy of ROAD algorithm, the number of actual outliers (as per the designated outlier class labels) present among the elements of the likely set is considered. Following this measure, the results obtained on the benchmark data sets are reported in Table 5.6 under the column labeled 'Combined'. For a comparative view, performance of the AVF and the Greedy algorithms on the same data is indicated here. The best performance obtained corresponding to each data set is shown in bold font.

From this experimentation, one can understand that the ROAD algorithm exhibited better results over the other two methods. This is due to the additional capability of this algorithm in detecting outliers belonging to *Type-2*. To understand this better, the number of Type-1 and Type-2 outliers detected are shown separately in Table 5.6. The number in brackets under the column labeled 'Type-2' indicates the value of the parameter k using which these results are obtained. A fixed value 5 is used for the other parameter α of the algorithm.

As described in Sect. 4.5, experimental results of outlier detection algorithm are typically reported using ROC curves. For increasing values of top portions (p) of the ranked outlier list, TPR and FPR values are determined to produce corresponding ROC curves of all the three algorithms. The curves thus generated on the benchmark data sets are shown in Fig. 5.5 for a quick understanding of their overall performance.

5.5.2 Sensitivity Study

The sensitivity of ROAD algorithm is explored here with respect to its parameters, namely k and α. The impact of these parameters on the performance of the algorithm

Table 5.6 Performance comparison on benchmark data sets

Name of the data set	# Outliers present (p value)	# Outliers detected (among the top p objects)				
		ROAD algorithm			AVF algorithm	Greedy algorithm
		Type-1	Type-2	Combined		
Chess (End-Game)	305	78	115 (8)	**126**	78	90
Tic-Tac-Toe	66	17	30 (14)	**37**	17	8
Breast Cancer (W)	47	37	32 (3)	**42**	37	41
Congressional Votes	21	16	2 (2)	**18**	16	16
Breast Cancer	16	4	5 (3)	**5**	4	3
Mushroom	783	454	132 (8)	**575**	454	340

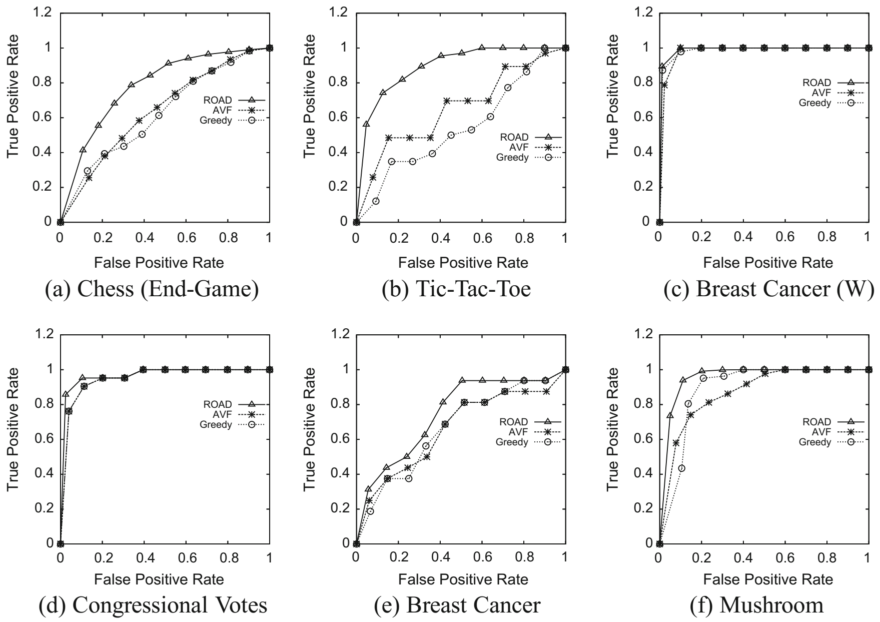


Fig. 5.5 Performance comparison using benchmark data sets

is measured using the Equal Error Rate (EER) measure, as described in Sect. 4.5. In this study, sensitivity of the algorithm is explored using Tic-Tac-Toe data shown in Table 5.4. The effect of varying the number of clusters k (keeping the value of α fixed at various values 5, 7 and 9) is as shown in Fig. 5.6a. This figure indicates that with increasing k value, the performance of ROAD algorithm has improved, as characterized by low EER values. However, arbitrarily high values of k may result in more number of small sized clusters, or the clustering step may not converge.

The observations made in a similar study with respect to parameter α, for various fixed values of k, are shown in Fig. 5.6b. According to this figure, with the increase in the value of α, the number of big clusters has reduced and this has negative impact on the performance, more so for high values of k. Moreover, with high values of α, there may not be any big clusters left out reducing the number of Type-2 outliers detected to zero. On the other hand, a very low value of α is not desirable as it cannot differentiate even the known small clusters from big ones. It is important to note that the parameters k and α are not independent. Thus, the set of allowable values for α depends on the value assigned to k.

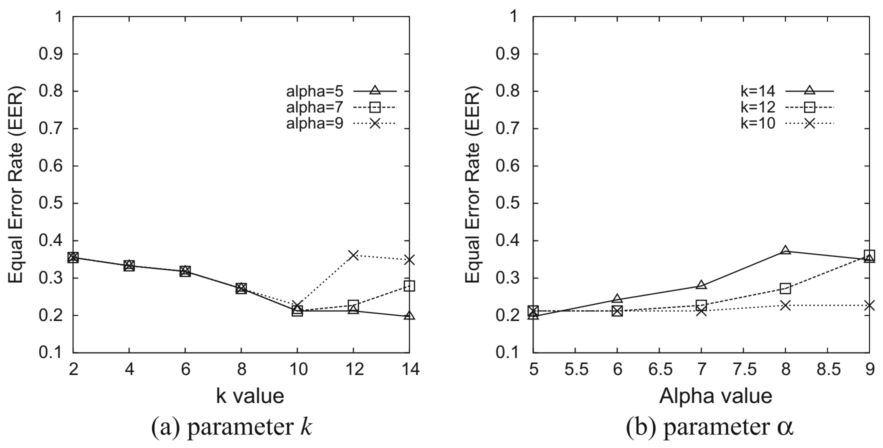


Fig. 5.6 Sensitivity analysis on Tic-Tac-Toe data set

5.6 Current Research Trends

Having seen numerous methods for outlier detection in categorical data, it is time to look at the current research trends based on some recent research efforts towards this problem.

5.6.1 Outlier Detection in Categorical Data Using Holoentropy

An information theoretic outlier detection method based on the concept of *weighted holoentropy* combines the entropy and the total correlation measures with attribute weighting for identifying exceptional objects. The rationale behind this method is to assign weights to the individual attributes so as to give more importance to those attributes with small entropy values for increasing the impact of removing an outlier candidate that is outstanding on those attributes. Hence, the weight computation for the attribute entropy employs a reverse sigmoid function of the entropy itself.

Based on the above description, a novel definition for outliers turns out to be the set of objects with greatest decrease in the weighted holoentropy value when deleted from the original data set. Similarly, the outlier factor computation is also modified to include weighted holoentropy by defining approximate differential holoentropy. As a result, the outlier factor (OF) of an object X_i is computed as

$$OF(X_i) = \sum_{r=1}^{m} OF(x_{i,r}) = \sum_{r=1}^{m} \begin{cases} 0, & \text{if} freq(x_{i,r}) = 1, \\ w_{\psi}(A_i).\delta[freq(x_{i,r})], & \text{else.} \end{cases} \tag{5.16}$$

This method, named as Information Theory Based Step by Step (ITB-SS), determines an anomaly candidate set (AS) and an upper bound on the number of likely outliers (UO). In order to determine the final set of outliers, only the objects in the AS set are examined thereby reducing the time complexity of the algorithm to $O(om(UO))$, where o is the number of outliers to be identified and m is the data dimensionality. Through an empirical study, it is found that the average UO is about $0.21n$, where n the number of objects in the data. However, the number of objects UO to be examined increases significantly for large data sets.

5.6.2 Detecting Outliers in Categorical Data Streams

Most of the techniques detecting anomalies in data streams are either distance-based or cluster-based. Both these varieties require distance computation to achieve their intended purpose. Due to the known limitations of categorical attributes, these methods lack the ability to handle such data in a proper manner. To effectively solve this problem, the concept of mining closed frequent patterns is employed. Based on this idea, the method designed for identifying outliers in categorical data streams is named as Detecting Outliers in Sliding Window over Categorical Data Streams (DOSW_CDStream).

In the context of data streams, *support* of an item set X turns out to be the ratio of the number of categorical data that contain X to the width of the sliding window w. An item set X can be considered as *closed frequent pattern* over the current sliding window, if its support is greater than the minimum support and there is no super-set of X with same support value.

The width of the sliding window is determined by two sliding end points. In this model, old data objects are thrown out as the new data arrive. The current sliding window consisting of w data objects is denoted as

$$SW_{n-w+1} = \{T_{n-w+1}, T_{n-w+2}, \ldots, T_n\} \tag{5.17}$$

Here, n indicates the current time. In order to determine outliers, this method introduced a formula named as Weighted Closed Frequent Pattern Outlier Factor (WCF-POF). For an arbitrary categorical data object T_i, its outlier factor is measured as

$$WCFPOF(T_i) = \frac{\sum_{P \subseteq T_i, P \in CFPS(DS,minsup)} sup(P) * (|P|/|T_i|)}{|CFPS(DS, minsup)|} \tag{5.18}$$

where, $CFPS(DS, minsup)$ is the set of closed frequent patterns in the current window, $|P|$ is the length of closed frequent pattern P and $minsup$ is the minimum support specified by the user. A smaller value of this outlier factor indicates a greater possibility of the corresponding object being an outlier. This method first determines all the closed patterns and then computes their outlier factors. The summary details are stored in an index structure, which can be updated by a decay function. Finally, top-k outliers are determined based on a ranking over the outlier factor values. The computational cost of this method is noted as $O(CFPS + w * N + 2w + (2w * log(2w)))$.

5.7 Summary

An important research issue concerning the outlier detection problem, namely dealing with data described using categorical attributes is studied here. A majority of the outlier detection methods mainly deal with numerical data. However, data pertaining to several real life applications tend to be categorical in nature with high dimensionality. So, it is necessary to have relevant techniques to process such data.

Addressing this requirement, some important research efforts that have gone in this direction are discussed in this chapter.

- Issues and challenges in dealing with categorical data analysis are highlighted.
- Emphasizing on the role of clustering as an unsupervised learning task, a few algorithms for clustering categorical data are presented.
- A ranking-based characterization of outliers in categorical space is presented along with the associated methodology.
- A sample set of algorithms for detecting outliers in categorical data are presented along with a discussion on the detection strategies employed by each method.
- Experimental study using benchmark categorical data sets is presented to facilitate a practical view of the outlier detection problem.

Towards the end, a brief discussion on a few current research trends regarding the outlier detection problem is presented.

5.8 Bibliographic Notes

Outlier detection problem has numerous applications such as fraud detection in financial transactions and intrusion detection in computer networks, etc [6, 18]. Various research issues are involved in outlier detection and so various methods were developed employing varied detection strategies [7, 17, 23, 27, 30, 33]. Clustering-based methods [21, 29] try to identify various groups of objects based on their intrinsic similarity in order to isolate objects with deviating characteristics. The LOF method [4] is a frequently used one for detecting local outliers in numerical data.

As stated at the beginning, the theme of this chapter is to delve on the methods dealing with categorical attributes. As brought out in [3], the characteristics of a categorical data set can be summarized using a simplified representation. The main computational challenge in this regard is that various values taken by a categorical variable are not inherently ordered [2]. Thus, evolving necessary methods to deal with categorical data is a non-trivial task.

Similar to LOF, a method named as Cluster Based Local Outlier Factor (CBLOF) was proposed [15] for detecting local outliers in categorical data. Subsequently, an approach of comparing against marginal distributions of attribute subsets was proposed [9] in an effort to identify anomalies in categorical data in an unsupervised manner. Another such effort is the Greedy algorithm [14], based on the concept of entropy of a data set. In order to detect p outliers in a data set, this algorithm requires p scans over the data, which is computationally expensive for large data sets. Addressing this issue, the AVF algorithm [22] was proposed using the attribute value frequencies. Each data object is assigned a frequency score and objects with low scores are considered as outliers. Though this algorithm can identify objects with infrequent attribute values as outliers, it doesn't explore the case of outliers arising due to infrequent combination of attributes with frequent values.

A distance-based method for outlier detection in high-dimensional categorical data was proposed in [23] using the notion of common neighbor of a pair of objects. According to this method, the distance between two objects X_i and X_j is defined as

$$dist(X_i, X_j, \theta) = 1 - \log_2 |CNS(X_i, X_j, \theta)| / \log_2 |D|$$

where

$$CNS(X_i, X_j, \theta) = NS(X_i, \theta) \cap NS(X_j, \theta).$$

Here, θ is a user defined threshold, $NS(X_i, \theta)$ is the set of neighbors of X_i. Though this method captures dissimilarity between objects driven by the global data distribution, its time complexity turns out to be quadratic in the number of objects. A genetic approach for outlier detection in projected space was proposed [1], which is applicable for both numeric and categorical data. Another recent effort is the SCF (Squares of the Complement of the Frequency) algorithm proposed in [29] using the attribute value frequencies like the AVF algorithm.

Given the unsupervised nature of the outlier detection problem, clustering-based methods are natural choices. Robust Clustering using linKs (ROCK) algorithm [13] works based on the notion of links between data objects, rather than applying any metric to measure the similarity. Squeezer algorithm [32] performs clustering of categorical data by reading the data objects one by one. The k-ANMI algorithm [16] is a k-means like clustering algorithm that directly optimizes the mutual information sharing based objective function.

It is known that the performance of any outlier detection algorithm directly depends on the way outliers are perceived [8]. Therefore, a customized definition meeting the specific requirements of an application is necessary to proceed with the detection of outliers. As brought out in [24], it is more meaningful to rank the data objects based on their degree of deviation. Accordingly, a ranking-based formalism, named as ROAD algorithm, for outlier detection in categorical data was presented in [26, 28] using data clustering as the basis. The objective was to capture all possible types of outliers considering their occurrence scenarios in the categorical space. In this connection, the k-modes algorithm [19] was employed for clustering due to its efficiency as a member of the *k-means* family [20] of algorithms. Moreover, employing the cluster initialization method [5] along with the categorical dissimilarity measure [25] made it more effective.

Experimental study of these methods can be carried out on various benchmark categorical data sets available at UCI ML Repository [10]. Typically, results of outlier detection algorithm are reported using ROC curves [12]. Likewise, the impact of the algorithmic parameters on its performance can be measured using the Equal Error Rate (EER) measure [21].

Some of the recent methods for dealing with categorical data are also presented in this chapter. An information theoretic outlier detection algorithm was proposed recently [31] by defining the concept of weighted holoentropy. Similarly, an algorithm for identifying outliers in categorical data streams was formulated in [30] based on the concept of mining closed frequent patterns.

References

1. Bandhyopadhyay, S., Santra, S.: A genetic approach for efficient outlier detection in projected space. Pattern Recognit. **41**, 1338–1349 (2008)
2. Bock, H.H.: The classical data situation. In: Analysis of Symbolic Data, pp. 139–152. Springer (2002)
3. Boriah, S., Chandola, V., Kumar, V.: Similarity measures for categorical data: a comparative evaluation. In: SIAM International Conference on Data Mining, Atlanta, Georgia, USA, pp. 243–254 (2008)
4. Breunig, M., Kriegel, H., Ng, R., Sander, J.: Lof: identifying density-based local outliers. In: ACM SIGMOD International Conference on Management of Data, Dallas, Texas, pp. 93–104 (2000)
5. Cao, F., Liang, J., Bai, L.: A new initialization method for categorical data clustering. Expert. Syst. Appl. **36**, 10223–10228 (2009)
6. Chandola, V., Banerjee, A., Kumar, V.: Anomaly detection: a survey. ACM Comput. Surv. **41**(3) (2009)
7. Chandola, V., Banerjee, A., Kumar, V.: Anomaly detection for discrete sequences: a survey. IEEE Trans. Knowl. Data Eng. (TKDE) **24**(5), 823–839 (2012)
8. Cui, Z., Ramanna, S., Peters, J.F., Pal, S.K.: Cognitive informatics and computational intelligence: theory and applications. Fundam. Inform. **124**(1–2), v–viii (2013)
9. Das, K., Schneider, J.: Detecting anomalous records in categorical datasets, San Jose, California. In: ACM KDD, pp. 220–229 (2007)
10. Dua, D., Efi, K.T.: UCI machine learning repository (2017). http://archive.ics.uci.edu/ml
11. Duan, L., Xu, L., Liu, Y., Lee, J.: Cluster-based outlier detection. Ann. Oper. Res. **168**, 151–168 (2009)
12. Fawcett, T.: An introduction to ROC analysis. Pattern Recognit. Lett. **27**, 861–874 (2006)
13. Guha, S., Rastogi, R., Kyuseok, S.: ROCK: A robust clustering algorithm for categorical attributes. In: International Conference on Data Engineering (ICDE), Sydney, Australia, pp. 512–521 (1999)
14. He, Z., Xu, X., Deng, S.: A fast greedy algorithm for outlier mining. In: Proceedings of Pacific Asia Conference on Knowledge Discovery in Databases (PAKDD), Singapore, pp. 567–576 (2006)
15. He, Z., Xu, X., Deng, S.: Discovering cluster-based local outliers. Pattern Recognit. Lett. **24**, 1641–1650 (2003)
16. He, Z., Xu, X., Deng, S.: k-ANMI: a mutual information based clustering algorithm for categorical data. Inf. Fusion **9**, 223–233 (2008)

17. Hido, S., Tsuboi, Y., Kashima, H., Sugiyama, M., Kanamori, T.: Statistical outlier detection using direct density ratio estimation. Knowl. Inf. Syst. **26**(2), 309–336 (2011)
18. Hodge, V.J., Austin, J.: A survey of outlier detection methodologies. Artif. Intell. Rev. **22**, 85–126 (2004)
19. Huang, Z.: A fast clustering algorithm to cluster very large categorical data sets in data mining. In: SIGMOD Data Mining and Knowledge Discovery Workshop, pp. 1–8 (1997)
20. Jain, A.K.: Data clustering: 50 years beyond K-means. Pattern Recognit. Lett. **31**, 651–666 (2010)
21. Jain, A.K., Duin, R.P.W., Mao, J.: Statistical pattern recognition: a review. IEEE Trans. Pattern Anal. Mach. Intell. **22**, 4–37 (2000)
22. Koufakou, A., Ortiz, E., Georgiopoulos, M.: A scalable and efficient outlier detection strategy for categorical data. In: Proceedings of IEEE ICTAI, Patras, Greece, pp. 210–217 (2007)
23. Li, S., Lee, R., Lang, S.D.: Mining distance-based outliers from categorical data. In: IEEE ICDM Workshop, Omaha, Nebraska, pp. 225–230 (2007)
24. Muller, E., Assent, I., Steinhausen, U., Seidl, T.: Outrank: ranking outliers in high dimensional data. In: IEEE ICDE Workshop, Cancun, Mexico, pp. 600–603 (2008)
25. Ng, M.K., Li, M.J., Huang, J.Z., He, Z.: On the impact of dissimilarity measure in k-modes clustering algorithm. IEEE Trans. Pattern Anal. Mach. Intell. **29**(3), 503–507 (2007)
26. Suri, N.N.R.R., Murty, M., Athithan, G.: An algorithm for mining outliers in categorical data through ranking. In: 12th International Conference on Hybrid Intelligent Systems (HIS), pp. 247–252. IEEE Xplore, Pune, India (2012)
27. Suri, N.N.R.R., Murty, M., Athithan, G.: Data mining techniques for outlier detection. In: Zhang, Q., Segall, R.S., Cao, M. (eds.) Visual Analytics and Interactive Technologies: Data, Text and Web Mining Applications, Chap. 2, pp. 22–38. IGI Global, New York, USA (2011)
28. Suri, N.N.R.R., Murty, M., Athithan, G.: A ranking-based algorithm for detection of outliers in categorical data. Int. J. Hybrid Intell. Syst. (IJHIS) **11**(1), 1–11 (2014)
29. Taha, A., Hegazy, O.M.: A proposed outliers identification algorithm for categorical data sets. In: 7th International Conference on Informatics and Systems (INFOS), Cairo, Egypt, pp. 1–5 (2010)
30. Wu, Q., Ma, S.: Detecting outliers in sliding window over categorical data streams. In: 8th International Conference on Fuzzy Systems and Knowledge Discovery (FSKD), pp. 1663–1667. IEEE (2011)
31. Wu, S., Wang, S.: Information-theoretic outlier detection for large-scale categorical data. IEEE Trans. Knowl. Data Eng. (TKDE) **25**(3), 589–602 (2013)
32. Zengyou, H., Xiaofei, X., Shengchun, D.: Squeezer: an efficient algorithm for clustering categorical data. J. Comput. Sci. Technol. **17**(5), 611–624 (2002)
33. Zhang, K., Hutter, M., Jin, H.: A new local distance-based outlier detection approach for scattered real-world data. In: PAKDD, Bangkok, Thailand, pp. 813–822 (2009)

Chapter 6
Outliers in High Dimensional Data

Abstract This chapter addresses one of the research issues connected with the outlier detection problem, namely dimensionality of the data. More specifically, the focus is on detecting outliers embedded in subspaces of high dimensional categorical data. To this effect, some algorithms for unsupervised selection of feature subsets in categorical data domain are furnished here. A detailed discussion on devising necessary measures for assessing the relevance and redundancy of categorical attributes/features is presented. Experimental study of these algorithms on benchmark categorical data sets explores the efficacy of these algorithms towards outlier detection.

6.1 Introduction

Data pertaining to many real life applications like market-basket data, biological data and computer network data tend to be high dimensional in nature. Such data are typically described by a number of categorical features, in addition to numeric attributes. This requires outlier detection methods that can scale well with the data dimensionality. Data sparseness in high dimensional space makes the task of outlier detection more challenging as the notion of distance between objects can not be measured in a meaningful manner.

Given the above practical scenario, the way out is to develop requisite algorithms by looking for outliers in subspaces of the data. As depicted in Fig. 6.1, different views of the same set of data in various subspaces reveal interesting observations. More precisely, note the fact that the designated outlier objects *A* and *B* are separable from the rest of the data only in some views. The rationale behind this phenomena is that these outliers are embedded in subspaces of the data and so they can be seen only in some specific views of the data. When projecting the data to such low dimensional views, the specific feature subset selected for representing the data determines how well the outliers can be detected. So, the key aspect here is to construct the desired low dimensional representations of data by selecting the features that are of relevance to the outlier detection task.

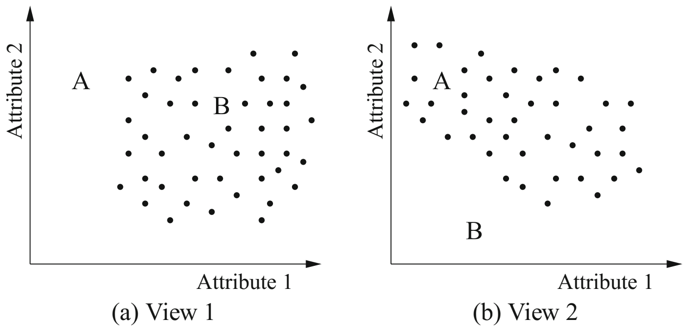


Fig. 6.1 Various views of outliers with different selected feature subsets

Also, high dimensionality of data makes the outlier detection more challenging due to the presence of noisy and irrelevant features that degrade the performance of the detection process. In such a situation, the suggested course of action is to resort to feature subset selection by employing necessary measures for determining the usefulness of the selected features. The following aspects of the outlier detection problem highlight the need for feature selection and suggest the nature of solution required for this task.

- It is known that outlier detection is a class imbalanced data problem. Due to the skewness in the data distribution, majority of the objects belong to the normal class and only a few objects turn out to be outliers. Due to this, looking for outliers in high dimensional data becomes a challenge.
- Outlier detection is generally considered as an unsupervised learning problem due to lack of prior knowledge about the nature of various outlier objects. Moreover, unlabeled data are available in abundance, while obtaining labeled data is expensive in most of the applications.
- Most often it is the case that the input data pertaining to several real life applications consists of categorical attributes. So, necessary measures in the categorical space need to be devised for carrying out feature selection on such data.

This motivates the need for an efficient unsupervised method for selecting features of relevance to outlier detection when dealing with high dimensional categorical data.

With increasing number of attributes/features (m) describing the data, one needs to explore an exponential number ($2^m - 1$) of possible subspaces. This makes the task of feature selection practically impossible due to prohibitively high computational effort involved. Addressing this issue, various sequential-search-based approximation schemes are evolved such as the best individual features and the sequential forward search.

In pattern recognition applications, optimal characterization condition often means minimal classification error. This condition usually requires the maximal statistical dependency of the target class on the data distribution in the selected subspace. This can be realized by selecting the features with the highest relevance to the target class. Also, it is observed that the combinations of individually good features do not necessarily produce good learning performance. In this context, it is noted that *the m best features are not the best m features*. This is due to the redundancy present among the subset of features selected. To handle this situation, it is essential to select features with minimal redundancy while ensuring their relevance to the learning task.

6.2 Pattern Analysis in High-Dimensional Spaces

One of the early studies that tried to explain the problems associated with high dimensional data is the *curse of dimensionality*. In simple terms, as the dimensionality increases the domain of the pattern vector space becomes so huge that a finite collection of data points drawn from this space provides only a sparse representation. Using such a small data set for training a classifier or a clustering model often poses a challenge to the machine learning community. An associated issue in training classifiers is the *peaking phenomenon*, which means the performance of many classifiers peaks for a certain number of features and then decreases as more features are added. It is known that the number of training patterns required per class is 5 to 10 times the dimensionality of the data, as per an empirical observation.

Another important result is associated with the behavior of distances between pairs of objects in high-dimensional spaces. As the dimensionality keeps increasing, the following property holds for any object X in the high-dimensional pattern space.

$$\frac{d(X, X_n)}{d(X, X_f)} \rightarrow 1 \tag{6.1}$$

where X_n is the nearest neighbor of X, X_f is the farthest neighbor of X, and d is the distance function. Further, it is known that L_1 norm is more effective compared to the popular Euclidean or L_2 norm. Prior work is available on exploring the role of fractional norms. For example, the p fractional distance between any two objects X_i and X_j in an m-dimensional space is given by

$$d_p(X_i, X_j) = \left[\sum_{k=1}^{m} |x_{ik} - x_{jk}|^p \right]^{\frac{1}{p}} \quad \text{where} \quad 0 < p < 1 \tag{6.2}$$

6.3 Feature Selection Using Mutual Information

In pattern recognition, each data object is represented using a set of features/attributes. The objective is to select features that enable better discrimination among the objects belonging to distinct classes. Thus, feature selection is the process of identifying a subset of the given features according to some criteria for removing irrelevant, redundant or noisy features. This is an active research problem in data mining as it speeds up the learning process through dimensionality reduction and improves the learning accuracy by removing the noisy features.

More formally, given a labeled data set D with n data objects described using m features $\{f_1, f_2, \ldots, f_m\}$, and the target classification variable c, the feature selection problem is to find a subspace of k features (with $k < m$) that optimally characterizes the classification (c). There are basically two approaches for feature selection, namely wrapper methods and filter methods.

1. A *wrapper method* for feature selection convolves with a classifier aiming to minimize the classification error of the particular classifier. Usually, wrapper methods yield high classification accuracy for a particular classifier with high computational complexity and less generalization of the selected features on other classifiers.
2. On the other hand, a *filter method* selects features by testing whether some preset conditions about the features and the target class are satisfied. In practice, filter approaches have much lower complexity than wrapper methods. The features thus selected are expected to yield comparable classification errors for different classifiers.

A wrapper method in turn can be realized in two distinct ways: forward feature selection and backward feature elimination. Depending on the specific application requirement, an acceptable choice can be made between filter and wrapper methods for feature selection. However, for the purpose of outlier detection, wrapper methods cannot be considered due to the unsupervised nature of learning involved.

Referring to Information theory, Mutual Information (MI) is a measure of the interdependence between two random variables, as computed using the expression given in Eq. 4.24. MI-based methods exist for measuring the relevance and redundancy among features for feature selection. In the case of labeled data, computing the joint MI between a multi-dimensional feature vector $(\{f_i, f_j\})$ and the class variable (c) is impractical. In such a situation, it is feasible to compute only the pair-wise MI values such as $I(c; f_i)$ and $I(f_i; f_j)$. The idea here is that MI measures the information content of a given feature with regard to the learning task at hand. Based on this idea, various MI-based methods are developed for feature selection to accomplish varied learning tasks.

A deeper look at MI philosophy reveals that this has two distinguishing properties in comparison with other dependency measures: (i) the capacity of measuring any kind of relationship between two random variables and (ii) its in-variance under space transformations, which are invertible and differentiable such as translations, rotations, and any transformation that preserves the order of the original elements of the variables. Thus, MI turns out to be an attractive option for feature selection, as each feature is considered to be a random variable.

6.3.1 mRMR Algorithm

The minimal-Redundancy-Maximal-Relevance (mRMR) criterion based method is aimed at selecting good features according to the maximal statistical dependency criterion (Max-Dependency) based on Mutual Information. As Max-Dependency criterion is hard to implement, an alternative is to select features based on maximal relevance criterion (Max-Relevance). It is likely that features selected according to Max-Relevance could have high redundancy (dependency) among them. To combat this problem, minimal redundancy (Min-Redundancy) condition is included in this method to select mutually exclusive features. Thus, this algorithm uses a series of intuitive measures of relevance and redundancy to select promising features for both continuous and discrete data sets.

According to this algorithm, feature selection is performed in an incremental manner by selecting one feature at a time. Having selected a set S_{k-1} of $(k-1)$ features, the one that maximizes the following expression is selected next as the kth feature.

$$G = I(f_i; c) - \frac{1}{k-1} \sum_{f_s \in S_{k-1}} I(f_i; f_s) \tag{6.3}$$

The computational complexity of this incremental search method is $O(|S|.m)$. Here, m is the number of features/attributes describing the data set D. Two significant aspects of the mRMR feature selection method are as follows.

1. Through a theoretical analysis, it is shown that mRMR method is equivalent to Max-Dependency for first-order feature selection.
2. A two-stage algorithm combining mRMR with other feature selection methods (such as wrappers) shows that the space of candidate features selected by mRMR is more characterizing.

As the mRMR method takes the difference between the relevance term and the redundancy term (given in Eq. 6.3), it may so happen that a redundant feature having relatively large relevance gets selected as one of the top features. Though imposing a greater penalty on the redundancy term would lessen this problem, it cannot be avoided completely. An alternative approach is to examine the relevance and the redundancy of a feature independently and establish its utility for a specific learning task. This is the basic idea followed in the design of the feature selection method named as NMIFS-OD, presented in the following section, for accomplishing outlier detection.

6.3.2 NMIFS Algorithm

The mutual information between two features f_i and f_j can be written in terms of their entropies and their conditional entropies as given below.

$$I(f_i; f_j) = H(f_i) - H(f_i|f_j) = H(f_j) - H(f_j|f_i) \tag{6.4}$$

From the above formulation, MI can take values in the range $0 \le I(f_i, f_s) \le min\{H(f_i), H(f_s)\}$. This implies that the MI between two random variables is bounded above by the minimum of their entropies. As the entropy of a feature could vary greatly, so is the MI value. To restrict the MI value to the range [0, 1], it should be normalized with their corresponding entropy. This compensates for the MI bias towards multivalued features.

Therefore, the normalized mutual information (NMI) between two features f_i and f_j is given by

$$NMI(f_i, f_j) = \frac{I(f_i, f_j)}{min\{H(f_i), H(f_j)\}} \tag{6.5}$$

Improving over the mRMR method for feature selection, the redundancy between the ith feature and the subset of already selected features $S = \{f_j\}$ with $j = 1, \ldots, |S|$, is measured by taking the average of normalized MI computed pair-wise, as shown below (Eq. 6.6). Accordingly, the criterion used by this algorithm is to select the feature that maximizes the following measure.

$$G = I(c; f_i) - \frac{1}{|S|} \sum_{f_j \in S} NMI(f_i; f_j) \tag{6.6}$$

The second term on the right hand side is a kind of correlation measure, that is symmetric and takes values in [0, 1]. A value '1' suggests that feature f_i is highly correlated to all features in the selected subset S.

Based on the above selection criterion, the Normalized Mutual Information Feature Selection (NMIFS) algorithm is as follows.

1. Initialization: Set $F = \{f_i | i = 1, \ldots, N\}$ initial set of N features, and $S = \{\phi\}$, empty set.
2. Calculate the MI with respect to the classes: Calculate $I(f_i; c)$, for each $f_i \in F$.
3. Select the first feature: Find $\hat{f_i} = max_{i=1,\ldots,N}\{I(f_i; c)\}$.
   Set $F \leftarrow F \setminus \{\hat{f_i}\}$; set $S \leftarrow \{\hat{f_i}\}$.
4. Greedy Selection: Repeat until $|S| = k$.
   - Calculate the MI between features: Calculate $I(f_i; f_j)$ for all pairs (f_i, f_j) with $f_i \in F$ and $f_j \in S$, if it is not available.
   - Select next feature: Select feature $f_i \in F$ that maximizes the measure in Eq. 6.6. Set $F \leftarrow F \setminus \{f_i\}$; set $S \leftarrow \{f_i\}$.
5. Output the set S containing the selected features.

The main computation involved with this algorithm is the concurrent sorting of each pair of features, which is required to estimate the NMI values. Thus, the computational complexity of NMIFS algorithm is known to be $O(nlog(n))$.

6.4 Unsupervised Feature Selection for Outlier Detection

Based on the theoretical foundations outlined in the previous sections, it is clear that mutual information based characterization is an interesting approach for feature selection. Given the specific requirements of the outlier detection problem, an unsupervised feature selection method based on mutual information is discussed here to handle high dimensional categorical data.

This method essentially uses NMI for characterizing the redundancy among various features. This is basically a filtering method that establishes the utility of a categorical feature in accomplishing the outlier detection task through feature-wise entropy computation. Features thus selected are expected to highlight the deviations characterizing the outliers with minimum redundancy among them.

Let the input be an m-dimensional data set D consisting of n data objects with the descriptive features $F = \{f_1, f_2, \ldots, f_m\}$. The objective is to determine a feature subset $F_s \subset F$ with as low redundancy as possible among the selected features, so as to detect the outliers present in the input data in an efficient manner. Accordingly, the outlier detection process involves two major computational tasks as shown in Fig. 6.2.

The first task is to carry out feature selection on the input data to determine a relevant subset of features F_s. As a result, a low-dimensional version of the input data is produced by dropping the irrelevant features. The next task is to employ a suitable outlier detection algorithm on this low-dimensional categorical data to establish the utility of the selected features. This can be accomplished by picking a requisite algorithm from the available methods.

With the above overview, the details on the relevance assessment and redundancy computation performed by this method are presented below.

6.4.1 *Relevance Assessment*

A way of assessing the relevance of a categorical feature/attribute for outlier detection is to measure its entropy value. The rationale behind this process is that a feature contributing more to the detection of outliers (relevance) tends to have less entropy value (more skewed distribution of the values that it takes) as opposed to a feature primarily characterizing the normal objects. Though a similar situation may arise

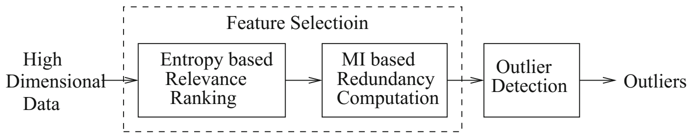


Fig. 6.2 Outlier detection through unsupervised feature selection

when the data set is having an attribute value with lower count without having any outliers, one can assume that it doesn't occur naturally. Hence, the following relationship can be seen between the relevance $Rel(f_i)$ of a feature f_i and its entropy value $H(f_i)$.

$$Rel(f_i) \propto \frac{1}{H(f_i)} \tag{6.7}$$

where the entropy $H(f_i)$ of the feature f_i is given by Eq. 4.22. From this understanding, it is clear that the most relevant features for outlier detection are the ones with least entropy values.

6.4.2 Redundancy Computation

Based on the mutual information measure, this method has the following mathematical expressions devised for computing the redundancy of a single feature as well as a subset of features.

1. Redundancy of a feature f_i with respect to another feature f_j is given by the normalized mutual information (NMI) between these two features.

$$R(f_i, f_j) = NMI(f_i, f_j) \tag{6.8}$$

   where $NMI(f_i, f_j)$ is determined using Eq. 6.5 as both f_i and f_j are categorical attributes/features.
2. The average redundancy of a feature f_i with respect to a set of features F_s is computed as the average of pair-wise redundancy values of f_i with respect to each feature $f_j \in F_s$ as

$$AR(f_i, F_s) = \frac{1}{|F_s|} \sum_{\forall f_j \in F_s} R(f_i, f_j) \tag{6.9}$$

3. The average redundancy of a set of features F_s is determined by taking the average of the average redundancy values of every feature $f_i \in F_s$ with respect to the subset $F_s \setminus \{f_i\}$ of features as given below.

$$AAR(F_s) = \frac{1}{|F_s|} \sum_{\forall f_i \in F_s} AR(f_i, F_s \setminus \{f_i\}) \tag{6.10}$$

6.4.3 NMIFS-OD Algorithm

Given a data set D consisting of n objects described with a set F of m categorical attributes/ features, the objective is to determine a subset $F_s \subset F$ of the features such that the selected features are relevant to the outlier detection task and have less redundancy among them. Towards this objective, the Normalized Mutual Information-based Feature Selection for Outlier Detection (NMIFS-OD) algorithm attempts to perform feature selection on categorical data employing the relevance assessment and the redundancy computation described above.

The NMIFS-OD algorithm deals with the relevance check and the redundancy evaluation of a feature independently. Within these two steps, relevance assessment is given priority over redundancy evaluation. Accordingly, the feature selection process progresses in an incremental manner till the average redundancy among the selected subset F_s of features is within the set threshold or all the input features are exhausted. The computational steps involved in this algorithm are listed below.

1. Compute entropy $H(f_i)$, $\forall f_i \in F$ using Eq. 4.22.
2. Compute the average redundancy $AAR(F)$ using Eq. 6.10.
3. Assign a value for redundancy threshold: $thresh \leftarrow AAR(F)$ (suggested).
4. Arrange features $F' = \{f'_1, f'_2, ..., f'_m\}$ in ascending order of entropy values.
5. Initialize selected list with first feature $F_s \leftarrow \{f'_1\}$.
6. Repeat the following steps for $i = \{2, \dots, m\}$
    - If $(H(f'_i) \neq 0)$ then compute $AR(f'_i, F_s)$ using Eq. 6.9.
    - If $(AR(f'_i, F_s) \leq thresh)$ then set $F_s \leftarrow F_s \cup \{f'_i\}$
7. Output the set F_s containing the selected features.

This algorithm produces a subset F_s of k features (where $k < m$) with less redundancy by applying a threshold (*thresh*) on the allowed maximum redundancy among the selected features. Hence, determining an appropriate value for this parameter in turn determines the qualifying features. The average redundancy of all the features of the input data, i.e. $AAR(F)$ is considered as the reference value for threshold selection, as this is a reasonable indicator of the intrinsic characteristic of the data. Not withstanding this suggested value, one may consider a satisfactory value based on the specific application requirement.

With applicable simplification to the above computational steps, the time complexity of NMIFS-OD algorithm turns out to be linear in terms of n. Thus, this algorithm is feasible for feature selection in large data sets. The following list indicates some of the noteworthy properties of this algorithm.

- *Decoupling relevance from redundancy*: Unlike other algorithms, the relevance check and the redundancy evaluation of a candidate feature are carried out independently.
- *Relevance takes priority over redundancy*: Determining the relevance of a feature is attempted before evaluating its redundancy with emphasis on the intended learning task, i.e., outlier detection in this case.
- *Deterministic relevance/redundancy computation*: Unlike the case of continuous valued features, entropy and mutual information computations on discrete features don't require any density estimation. To that extent, the computations involved with this algorithm are deterministic in nature.

6.5 Experimentation on Benchmark Data

Experimental study is carried out to explore the performance of the algorithms discussed in the previous section for feature selection. More specifically, this study is meant for demonstrating the impact of the unsupervised feature selection on outlier detection using the resultant low dimensional data. So, data sets with reasonably high dimensionality are needed for this investigation. Accordingly, six real life categorical data sets from UCI ML Repository are chosen as furnished in Table 6.1.

Following the standard procedure for data preparation, processed data sets are obtained as shown in Table 6.2. To induce class imbalance in the data, only a subset (every 5th object) of the designated outlier class objects are considered across all data sets. Lymphography data are the only exception to this object subset selection as this data set has only six objects corresponding to the designated outlier class(es). In case of Promoters Genes data and Splice-Junction data, the instance name field (second column) is removed in pre-processing, resulting in 57 and 60 features respectively. As the feature selection algorithms considered here work in unsupervised learning mode, they don't demand labeled data. However, class labels are used to measure their performance in detecting outliers using the low dimensional data.

Table 6.1 Details of high dimensional categorical data sets

Data set name	# Features	# Objects	# Classes
Mushroom	22	8124	2
Chess (KR/KP)	36	3196	2
Splice-junction	61	3190	3
Lymphography	18	148	4
Promoters genes	58	106	2
SPECT heart	22	80	2

Table 6.2 Details of normal and outlier objects in data sets

Data set name	Normal class	# Normal objects	Outlier class	# Outlier objects
Mushroom	e	4208	p	783
Chess (KR/KP)	Won	1669	Nowin	305
Splice-Junction	EI,IE	1535	N	331
Lymphography	2,3	142	1,4	6
Promoters genes	+	53	−	10
SPECT heart	0	40	1	8

6.5.1 Outcome of Feature Selection

The initial part of this study is to perform feature selection using NMIFS-OS algorithm on the benchmark data. Accordingly, Fig. 6.3 shows the ranked sequence of features in ascending order of their relevance, with the most relevant feature being ranked first. Each feature in this ranked sequence is examined for its possible inclusion in the selected set as per the redundancy criterion of the algorithm. All such selected features in the ranked sequence are marked with a '*' symbol on the corresponding impulse representation.

It is important to note that corresponding to two data sets shown in Fig. 6.3, the first feature in the relevance rank list is not selected as its entropy value is '0'. This is because this particular attribute/feature is having a single value throughout the data set. It is dropped by the feature selection algorithm as such a feature is anyway not a good candidate for the discrimination of outliers from the normal objects. Also, the details on the average redundancy values and the values assigned for redundancy threshold (*thresh*) corresponding to various data sets are furnished in Table 6.3.

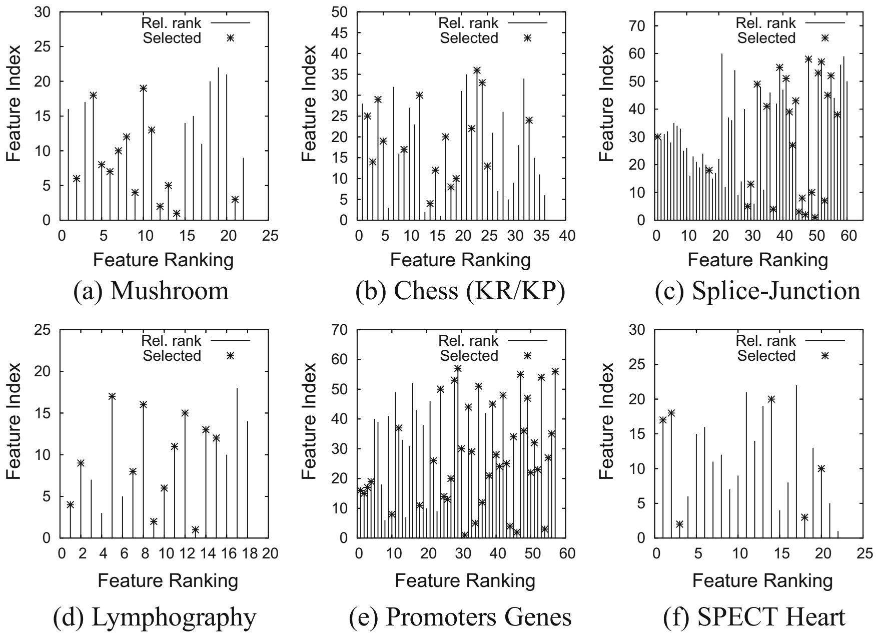


Fig. 6.3 Relevance based ranking of features with their selection status

Table 6.3 Redundancy threshold values assigned for various data sets

Data set name	# Features (original)	Average redundancy	Theshold value assigned	# Features (selected)
Mushroom	22	0.236	0.236	13
Chess (KR/KP)	36	0.072	0.03	16
Splice-junction	60	0.01	0.008	24
Lymphography	18	0.087	0.1	12
Promoters genes	58	0.072	0.09	41
SPECT heart	22	0.103	0.1	6

6.5.2 *Outlier Detection Performance*

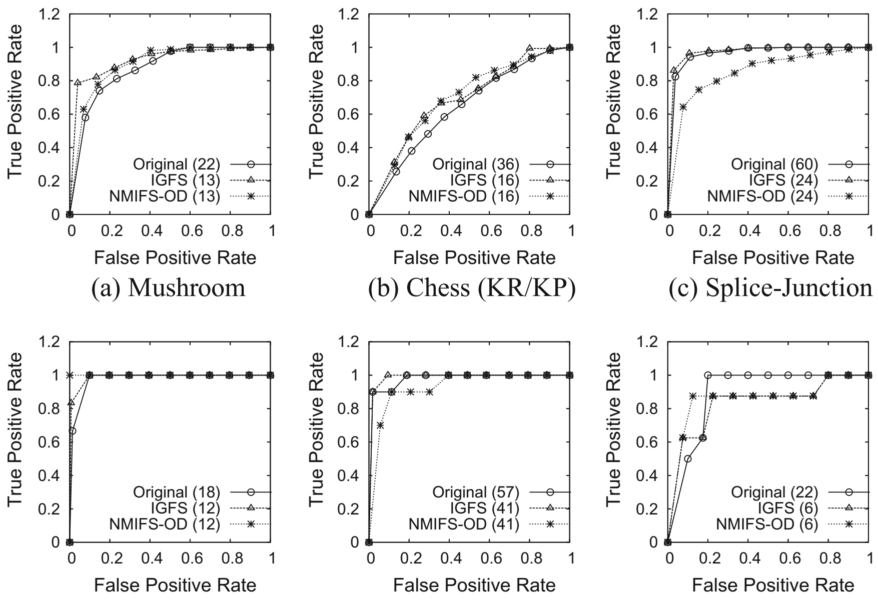
After performing feature selection on each one of the data sets, the resultant low dimensional data sets are subjected to outlier detection. At this stage, it is possible to employ any detection method working with categorical data. One such method discussed in Sect. 5.4, namely AVF method is considered in this experimentation.

For the purpose of comparing the performance of NMIFS-OD algorithm towards feature selection, the Information Gain-based Feature Selection (IGFS) method is considered. Information Gain (IG) based method is known to be one of the best performing methods for feature selection in class imbalanced problems like outlier detection. IG basically measures the difference between the entropy of the class labels (c) and the conditional entropy of the class labels given a feature f_i.

$$IG(c, f_i) = H(c) - H(c|f_i) \tag{6.11}$$

The entropy value is measured using the expression given in Eq. 4.22. IG is maximal for equal probable classes, and uncertainty is minimal. Since the IG-based selection is performed in supervised setting, it is considered as the baseline for comparison in this experimentation.

As per the general practice, results of the outlier detection process are reported using ROC curves. The ROC curves obtained using AVF method on the low dimensional data as well as on the original data are furnished in Fig. 6.4. Low dimensional data obtained using both the feature selection methods, namely NMIFS-OD and IGFS are utilized here with the same number of selected features. The exact number of features selected corresponding to each data set is indicated in the legend in parenthesis. Thus, the effectiveness of NMIFS-OD algorithm is evident from the superior or comparable performance obtained with the reduced set of features vis-a-via the full set.



(a) Mushroom (b) Chess (KR/KP) (c) Splice-Junction

(d) Lymphography (e) Promoters Genes (f) SPECT Heart

Fig. 6.4 Effect of feature selection on outlier detection

6.5.3 Effect of Number of Features Selected

An interesting aspect to explore is the effect of number of features selected on the outlier detection process. To facilitate this study, a micro array gene expression data set with reasonably high dimensionality (namely Lung cancer data) is considered. This data is discretized (as 3-states) and made available after pre-processing, as per the details furnished in Table 6.4.

Experimentation on this data set is carried out using NMIFS-OD algorithm by varying the redundancy threshold parameter to produce different sized selected sets of features. The outlier detection results using AVF algorithm on the low dimensional Lung cancer data corresponding to various cardinalities of the selected feature subsets are shown in Fig. 6.5. Referring to this figure, it is clear that more outliers are detected corresponding to feature subset cardinalities between 40 and 250 than with the original feature set of 325, and the highest is obtained with 41 features. This study suggests that a suitable low-dimensional representation of the data can yield improved outlier detection performance.

Table 6.4 Details of lung cancer data set

# Objects	Dimension	# Classes	Designated normal class	Designated outlier class
73	325	7	7 and 4	3

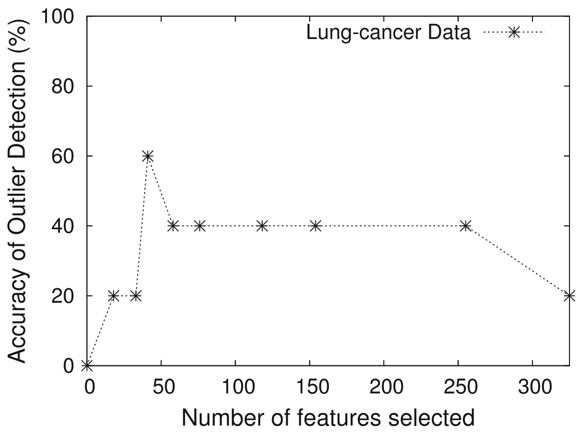


Fig. 6.5 Effect of the number of features selected

6.6 Summary

Dimensionality of the data is an important research issue related to outlier detection. This issue can be handled through the process of feature selection to reduce the effective dimensionality. Salient aspects of the material presented here are as follows:

- Feature selection is a means of combating the high dimensionality of the data, as dimensionality impacts the outlier detection performance.
- Mathematical formulation devised based on Information Theoretic principles is an interesting approach for measuring relevance and redundancy of a feature.
- A few of the algorithms for feature selection employing mutual information are discussed bringing out their computational properties.
- Selection of relevant features aimed at outlier detection can be carried out by employing an unsupervised feature selection method.
- The case of high dimensional categorical data can be handled by evaluating redundancy among features using Mutual Information (MI) based computation.
- The NMIFS-OD algorithm is a feature filtering method to select relevant features one-by-one in an incremental manner.
- The usefulness of feature selection for outlier detection is experimentally shown using benchmark categorical data sets.

6.7 Bibliographic Notes

High dimensional data has several problems associated with it leading to the curse of dimensionality [2]. A related issue in training classifiers is the peaking phenomenon [11, 13]. As the dimensionality keeps increasing, the notion of nearest neighbor loses its significance [3]. The work on determining the inter-object distances in high dimensional space explored the role of fractional norms [12].

Addressing high dimensionality of data, one can develop requisite algorithms by looking for outliers in subspaces of the data [1]. With increasing number of attributes/features describing the data, one needs to explore an exponential number of possible subspaces [22]. The solution for this problem is to perform feature subset selection [9, 28] employing necessary measures for determining the usefulness of the selected features.

There are a number of efforts made in the recent past for performing outlier detection in high dimensional data [20]. A near-linear time approximation algorithm for angle-based outlier detection in high-dimensional data was proposed in [29]. Similarly, a ranking-based method for detecting outliers in high dimensional data was proposed in [23]. A method for detecting outlying subspaces corresponding to high-dimensional data was proposed in [38]. The computational method based on random projection technique [35] preserves the distances from various data objects to their k-nearest neighbors while projecting to a low-dimensional space. Another approach for dealing with high-dimensional spaces applies eigenspace regularization on the training set in order to extract a relevant set of features for outlier detection [24]. Subsequently, a feature selection method applying a non-linear transformation in a feature-wise manner using kernel-based independence measures was developed [37]. Similarly, many methods were proposed [14, 17, 22, 25] for detecting outliers based on the subspace clustering concepts. However, most of these methods work primarily with numerical data.

Outlier detection is a known class imbalanced data problem [26] by nature. Similarly, for dealing with imbalanced text classification problems, it was suggested to carry out feature selection to combat the class imbalance problem [5]. However, feature selection for outlier detection should take into account the specific computational aspects of this problem.

A wrapper-based feature selection method was proposed in [15] for building classification models on binary class imbalanced data. Subsequently, a comparison of the methods evolved for imbalanced data classification problems employing various features selection metrics was presented in [30, 36]. Similarly, an approach employing feature bagging technique was proposed [19] by combining the results obtained from multiple instances of the outlier detection algorithm applied using different subsets of the features set.

Applying the information theory principles [18], entropy based formulations for varied pattern recognition tasks were developed [31]. Similarly, various mutual information based techniques were proposed for feature selection [7, 27, 34] in the recent days. However, all these methods are applicable mainly for supervised learning problems. The Laplacian score-based method proposed in [10] and the feature similarity based method presented in [21] perform feature subset selection in unsupervised manner. Similarly, another feature selection method was proposed recently for dealing with the problems having multi-cluster data [4]. Though these methods can be employed for unsupervised learning tasks, their computation deals with only numerical data and their performance on class imbalanced data needs to be explored.

Addressing the gaps associated with the above techniques, an unsupervised method for feature selection named NMIFS-OD was proposed [32, 33] by employing mutual information. Efficacy of this method towards outlier detection was demonstrated using AVF algorithm [16] over the low dimensional data produced corresponding to various benchmark categorical data sets taken from UCI ML Repository [6]. Information Gain (IG) based method, known to be one of the best performing methods for feature selection in class imbalanced problems [36], was considered as the baseline for the comparative study. Typically, outlier detection results are reported using the ROC curves [8], as the recommended method. So, the same was followed for producing the results of the NMIFS-OD method as well.

References

1. Aggarwal, C.C., Yu, P.S.: Outlier detection for high dimensional data. In: ACM SIGMOD International Conference on Management of Data, pp. 37–46. Santa Barbara, USA (2001)
2. Bellman, R.E.: Dynamic Programming. Courier Dover Publications, New York (2003)
3. Beyer, K.S., Goldstein, J., Ramakrishnan, R., Shaft, U.: When is nearest neighbor meaningful? In: 7th International Conference on Database Theory. ICDT, Lecture Notes in Computer Science, vol. 1540, pp. 217–235. Springer, Jerusalem, Israel (1999)
4. Cai, D., Zhang, C., He, X.: Unsupervised feature selection for multi-cluster data. In: 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 333–342. Washington DC, USA (2010)
5. Chawla, N., Japkowicz, N., Kotcz, A.: Editorial: special issue on learning from imabalanced data sets. ACM SIGKDD Explor. Newslett. **6**(1), 1–6 (2004)
6. Dua, D., Efi, K.T.: UCI machine learning repository (2017). http://archive.ics.uci.edu/ml
7. Estevez, P.A., Tesmer, M., Perez, C.A., Zurada, J.M.: Normalized mutual information feature selection. IEEE Trans. Neural Netw. **20**(2), 189–201 (2009)
8. Fawcett, T.: An introduction to ROC analysis. Pattern Recogn. Lett. **27**, 861–874 (2006)
9. Forman, G.: An extensive empirical study of feature selection metrics for text classification. J. Mach. Learn. Res. **3**, 1289–1305 (2003)
10. He, X., Cai, D., Niyogi, P.: Laplacian score for feature selection. In: Advances in Neural Information Processing Systems 18. Vancouver, Canada (2005)
11. Hughes, G.F.: Number of pattern classifier design samples per class (corresp.). IEEE Trans. Inf. Theory **15**(5), 615–618 (1969)
12. Kaban, A.: Fractional norm regularization: learning with very few relevant features. IEEE Trans. NNLS **24**(6), 953–63 (2013)
13. Kanal, L., Chandrasekaran, B.: On dimensionality and sample size in statistical pattern classification. Pattern Recogn. **3**, 225234 (1971)
14. Keller, F., Muller, E., Bohm, K.: Hics: high contrast subspaces for density-based outlier ranking. In: 28th International Conference on Data Engineering (ICDE), pp. 1037–1048. IEEE (2012)

15. Khoshgoftaar, T.M., Gao, K.: Feature selection with imbalanced data for software defect prediction. In: International Conference on Machine Learning and Applications, pp. 235–240 (2009)
16. Koufakou, A., Ortiz, E., Georgiopoulos, M.: A scalable and efficient outlier detection strategy for categorical data. In: Proceedings of IEEE ICTAI, pp. 210–217. Patras, Greece (2007)
17. Kriegel, H.P., Kroger, P., Schubert, E., Zimek, A.: Outlier detection in arbitrarily oriented subspaces. In: 12th International Conference on Data Mining (ICDM), pp. 379–388. IEEE Computer Society, Brussels, Belgium (2012)
18. Kullback, S.: Information Theory and Statistics. Dover, New York (1997)
19. Lazarevic, A., Kumar, V.: Feature bagging for outlier detection. In: ACM KDD, pp. 157–166. Chicago, USA (2005)
20. Liu, H., Li, X., Li, J., Zhang, S.: Efficient outlier detection for high dimensional data. IEEE Trans. Syst. Man Cybern. Syst. (online)
21. Mitra, P., Murthy, C.A., Pal, S.K.: Unsupervised feature selection using feature similarity. IEEE Trans. Pattern Anal. Mach. Intell. **24**, 301–312 (2002)
22. Muller, E., Assent, I., Iglesias, P., Mulle, Y., Bohm, K.: Outlier ranking via subspace analysis in multiple views of the data. In: 12th International Conference on Data Mining (ICDM), pp. 529–538. IEEE Computer Society, Brussels, Belgium (2012)
23. Muller, E., Assent, I., Steinhausen, U., Seidl, T.: Outrank: Ranking outliers in high dimensional data. In: IEEE ICDE 2008 Workshops: 3rd International Workshop on Self-managing Database Systems (SMDB), pp. 600–603. Cancun, Mexico (2008)
24. Nguyen, H.V., Gopalakrishnan, V.: Feature extraction for outlier detection in high-dimensional spaces. JMLR WCP Feature Sel. Data Min. **10**, 66–75 (2010)
25. Nguyen, H.V., Muller, E., Vreeken, J., Keller, F., Bohm, K.: CMI: an information-theoretic contrast measure for enhancing subspace cluster and outlier detection. In: SIAM International Conference on Data Mining (SDM), pp. 1–9. Asustin, Texas (2013)
26. Own, H.S., AAl, N.A.A., Abraham, A.: A new weighted rough set framework for imbalance class distribution. In: International Conference of Soft Computing and Pattern Recognition, pp. 29–34. Paris, France (2010)
27. Peng, H., Long, F., Ding, C.: Feature selection based on mutual information: criteria of max-dependency, max-relevance, and min-redundancy. IEEE Trans. Pattern Anal. Mach. Intell. **27**(8), 1226–1238 (2005)
28. Peters, J.F., Ramanna, S.: Feature selection: near set approach. In: MCD, pp. 57–71 (2007)
29. Pham, N., Pagh, R.: A near-linear time approximation algorithm for angle-based outlier detection in high-dimensional data. In: 18th ACM International Conference on Knowledge Discovery and Data Mining (SIGKDD), pp. 877–885. ACM, Beijing, China (2012)
30. Sarkar, B.K., Sana, S.S., Chaudhuri, K.S.: A combined approach to tackle imbalanced data sets. Int. J. Hybrid Intell. Syst. (IJHIS) **9**(4) (2012)
31. Subudhi, B.N., Nanda, P.K., Ghosh, A.: Entropy based region selection for moving object detection. Pattern Recogn. Lett. **32**, 2097–2108 (2011)
32. Suri, N.N.R.R., Murty, M.N., Athithan, G.: Unsupervised feature selection for outlier detection in categorical data using mutual information. In: 12th International Conference on Hybrid Intelligent Systems (HIS), pp. 253–258. IEEE Xplore, Pune, India (2012)
33. Suri, N.N.R.R., Murty, M.N., Athithan, G.: Detecting outliers in high dimensional categorical data through feature selection. J. Netw. Innovative Comput. (JNIC) **1**(1), 23–32 (2013)
34. Suzuki, T., Sugiyama, M.: Sufficient dimension reduction via squared-loss mutual information estimation. JMLR Workshop Conf. Proc. **9**, 804–811 (2010)
35. deVries, T., Chawla, S., Houle, M.E.: Finding local anomalies in very high dimensional space. In: IEEE ICDM, pp. 128–137 (2010)
36. Wasikowski, M., Chen, X.: Combating the small sample class imbalance problem using feature selection. IEEE Trans. Knowl. Data Eng. (TKDE) **22**(10), 1388–1400 (2010)
37. Yamada, M., Jitkrittum, W., Sigal, L., E.P.Xing, Sugiyama, M.: High-dimensional feature selection by feature-wise kernelized lasso (2012). arXiv:1202.0515v2 [stat.ML]
38. Zhang, J., Wang, H.: Detecting outlying subspaces for high-dimensional data: the new task, algorithms, and performance. Knowl. Inf. Syst. **10**(3), 333–355 (2006)

Chapter 7
Outlier Detection Using Rough Sets

Abstract Clustering-based methods for outlier detection are preferred in many contemporary applications due to the abundance of methods available for data clustering. However, the uncertainty regarding the cluster membership of an outlier object needs to be handled appropriately during the clustering process. Addressing this issue, this chapter delves on soft computing methodologies based on rough sets for clustering data involving outliers. In specific, the case of data comprising categorical attributes is looked at in detail for carrying out outlier detection through clustering by employing rough sets. Experimental observations on benchmark data sets indicate that soft computing techniques indeed produce promising results for outlier detection over their counterparts.

7.1 Introduction

Outlier detection is an important data mining task with various detection methods developed employing varied algorithmic strategies. Among them, clustering-based methods are frequently used due to the availability of a number of algorithms addressing various aspects of the process. Also, these algorithms are good at identifying the inherent clusters (groups) in the data, thereby enabling the detection of outliers more effectively.

When dealing with the outlier detection problem, there may be uncertainty prevailing in crisp labeling of a data object as a normal one or an outlier, as the exact semantics of the notion of outlier is context dependent. Hence, data clustering involving outliers has to deal with the ambiguity regarding the membership of an outlier object to one of the normal groups (or clusters) of objects, as depicted in Fig. 7.1. In such a situation, it may be difficult to differentiate distinct objects, and so one may find it convenient to consider granules for its handling. Granulation is a computing paradigm that is abstracted from natural phenomena. The structure of granulation can often be defined employing the soft computing approaches like rough sets, fuzzy sets or their combination. Due to this, soft computing approaches are the preferred choices for performing various computing tasks involving uncertainty/vagueness.

N. N. R Ranga Suri et al., *Outlier Detection: Techniques and Applications*,
Intelligent Systems Reference Library 155,
https://doi.org/10.1007/978-3-030-05127-3_7

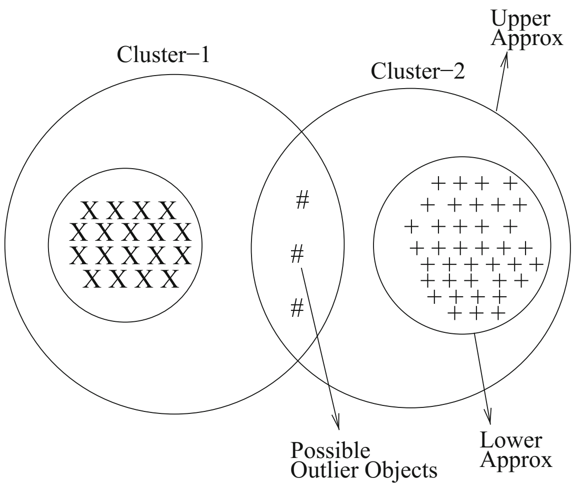


Fig. 7.1 Uncertainty in clustering of data involving outliers

Therefore, rough sets based approach is considered for outlier detection through data clustering.

One can notice considerable progress in applying rough sets theory for performing various computing tasks over numerical attributes. However, it is still challenging to deal with categorical data due to non-availability of requisite measures defined on such data. So, the objective is to look for some means of addressing this problem, emphasizing the need for rough sets based clustering methods for detecting outliers in categorical data.

7.2 Rough Set Theory

The purpose of developing Rough set theory is to deal with vagueness in the data. This theory relies on the assumption that with every object of the universe there is a certain amount of associated information expressed by means of some attributes. Objects with the same description are considered as indiscernible. While fuzzy sets deal with such data using a partial membership function, rough sets express the same by the boundary region of a set.

A rough set is a set of objects which cannot be with certainty classified as members of the set or its complement using the available knowledge. Thus, associated with every rough set (C), there is a pair of precise sets known as *lower approximation* ($\underline{C}$) and *upper approximation* ($\overline{C}$) of the set. The basic idea is to separate discernible objects from indiscernible ones and to assign them to lower and upper approximations of the set respectively. The main advantage of rough set theory in data analysis is that it does not need any preliminary or additional information about data, like probability distributions in statistics, a grade of membership in fuzzy set theory.

The concept of rough sets can be defined quite generally by means of topological operations *interior* and *closure*, called *approximations*. Consider a set of objects U called the *universe* and an *indiscernibility relation* $R \subseteq U \times U$, representing the lack of knowledge about elements of U. Assume that R is an equivalence relation and X is a subset of U. Now, the set X can be characterized with respect to R as follows.

- The *lower approximation* of X w.r.t. R is the set of all objects, which can be for *certain* classified as X.
- The upper approximation of X w.r.t. R is the set of all objects which can be *possibly* classified as X.
- The *boundary region* of X is the set of all objects, which can be classified neither as X nor as not-X.

The set X is *crisp* (with respect to R) if its boundary region is empty, and X is *rough* otherwise. The equivalence classes of R, called *granules*, represent elementary portion of knowledge that one can perceive due to R. Then, the approximations of a set X can be expressed in terms of these granules of knowledge as given below. These approximations are depicted graphically in Fig. 7.2.

- R-lower approximation of X: Union of all granules which are entirely included in the set. $R_*(X) = \cup_{x \in U}\{R(x) : R(x) \subseteq X\}$
- R-upper approximation of X: Union of all granules which have non-empty intersection with the set. $R^*(X) = \cup_{x \in U}\{R(x) : R(x) \cap X \neq \emptyset\}$

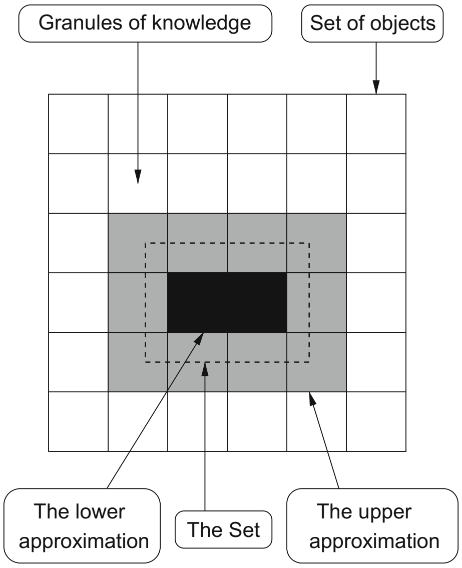


Fig. 7.2 Lower and upper approximations of a rough set

- R-boundary region of X: The difference between the upper and the lower approximation. $RN_R(X) = R^*(X) - R_*(X)$

Two major characteristics of rough set theory are handling uncertainty and providing a mathematical framework for granular computing. Granular computing (also known as perception-based computing) is a nature inspired computing paradigm that performs computation and operations on information granules, i.e. clump of similar objects or data points. Granular computing involves two main aspects: (i) formation, representation, and interpretation of granules (algorithmic aspect), and (ii) problem solving using granules (semantic aspect). According to this philosophy, the lower approximation of a rough set contains granules that completely belong to the set, and the upper approximation contains granules that partially or completely belong to the set.

Over the time, rough set theory along with granular computation has evolved as an attractive soft computing tool providing a stronger framework to achieve tractability and robustness, with close resemblance to human like decision making. The main advantage with this theory is its ability in clearly distinguishing between vagueness and uncertainty. Vagueness is the property of sets and can be described by approximations, whereas uncertainty is the property of elements of a set and can be expressed by the rough membership function. Though rough set theory has an overlap with several other theories dealing with imperfect knowledge (such as evidence theory, fuzzy sets), it is regarded as an independent and complementary discipline.

7.2.1 Characterizing Outliers Using Rough Sets

According to rough sets, a data table (also known as *information system*) can be represented as a quadruple $IS = (U, A, V, f)$, where

- $U = \{X_1, X_2, \dots, X_n\}$ is non-empty finite set of objects (universe).
- $A = \{A_1, A_2, \dots, A_m\}$ is a non-empty finite set of attributes.
- V is the union of attribute domains, i.e., $V = \cup_{i=1}^{m} DOM(A_i)$, where $DOM(A_i)$ denotes the domain of attribute A_i.
- $f : U \times A \rightarrow V$ is an information function such that for any $A_i \in A$ and $X_i \in U,\ f(X_i, A_j) \in DOM(A_j)$.

It is possible to define the concept of outliers using the above representation. Given an Information System (IS), for a data object $X_i \in U$, if X_i has some characteristics that differ greatly from those of other objects in U, in terms of the attributes in A, X_i is called an outlier in U with respect to the IS.

It is known that distance-based approaches for determining unusualness do not require explicit distribution. So, the following two distance measures are provided to facilitate detection of outliers. These measures presented in Sect. 4.1 are redefined here to incorporate rough set theory principles.

1. Given an information system $IS = (U, A, V, f)$ representing a categorical data set, the overlap measure (given in Eq. 4.1) to compute the distance between any two objects $X_i, X_j \in U$ using rough sets can be computed as

$$\Delta(X_i, X_j) = |\{A_r \in A : x_{i,r} \neq x_{j,r}\}| \tag{7.1}$$

   where $\Delta : U \times U \rightarrow [0, \infty]$ is a function, and $x_{i,r}$ and $x_{j,r}$ are the values of objects X_i and X_j respectively corresponding to attribute A_r. In fact, the pair-wise distances between objects can be obtained from the *discernibility matrix* of the information system. If details regarding the attribute weight are also available, the weighting factor can be incorporated with the overlap measure. However, it is assumed that all attribute values are of equal distance from each other, and thus cannot represent value pairs with differing degrees of similarities.

2. Similarly, the Value Difference Metric (VDM) over nominal/categorical attributes (given in Eq. 4.5) can be modified as given below to produce rough sets-based version. Given a decision table $DT = (U, A \cup B, V, f)$, where A is the set of condition attributes and B is the set of decision attributes, the VDM value between any two objects $X_i, X_j \in U$ is computed as follows

$$VDM_R(X_i, X_j) = \sum_{A_r \in A} d_{A_r}(x_{i,r}, x_{j,r}) \tag{7.2}$$

   where

$$d_{A_r}(x_{i,r}, x_{j,r}) = \sum_{E \in U/Ind(B)} \left(\frac{|[x_{i,r}] \bigcap E|}{|[x_{i,r}]|} - \frac{|[x_{j,r}] \bigcap E|}{|[x_{j,r}]|}\right)^2$$

   where $[x_{i,r}]$ is an equivalence class of $IND(A_r)$ that contains object X_i. For any $E \in U/IND(B)$, all elements in E have the same decision attribute values. $|[x_{i,r}]|$ is the number of all objects in U that have the value $x_{i,r}$ on attribute A_r. This implies that $[x_{i,r}] \bigcap E$ is the set of all objects in U that have the value $x_{i,r}$ on attribute A_r and belong to decision category E.

Though measures like the ones presented above are intended to facilitate distance-based outlier detection, clustering-based methods are found to be more prevalent. So, it is important to understand the mechanism of employing rough sets for data clustering.

7.3 Rough Clustering

The primary objective of rough clustering is to improve the data clustering process in terms of assigning objects to various clusters in a meaningful manner (soft membership as opposed to crisp assignment). It is expected to handle the uncertainty in dealing with objects that fall on the boundary of two or more clusters. This requirement gains significance when there are outliers present in the data, as such objects display this kind of phenomenon more prominently.

The objective in this regard can be met by introducing lower and upper approximation of rough sets during the clustering process, as depicted in Fig. 7.1. The upper and lower approximations of a cluster thus associated are required to satisfy the following fundamental properties of rough sets.

- A data object can be a member of one lower approximation at most.
- A data object that is a member of the lower approximation of a cluster is also member of the upper approximation of the same cluster.
- A data object that does not belong to any lower approximation is a member of at least two upper approximations of two different clusters.

7.3.1 RKM Algorithm

In contrast to rough sets theory relying on the classic set theory, rough clustering is inspired by intervals. The rough k-means (RKM) algorithm is an effort in this direction by suitably modifying two important aspects of the traditional k-means clustering method: (a) calculation of the centroids and (b) assigning data objects to clusters. The initial version of the RKM algorithm underwent certain enhancements for improving its efficiency. According to the improvised version of this algorithm, the core clustering steps are carried out as given below.

1. *Computing centroids of rough clusters*: Let $Z_j = \{z_{j,1}, \dots, z_{j,r}, \dots, z_{j,m}\}$ be the centroid of a rough cluster C_j. Then, Z_j is given by

   - If $(\underline{C_j} \neq \emptyset$ and $\overline{C_j} - \underline{C_j} = \emptyset)$ then $z_{j,r} = \frac{\sum_{X_i \in \underline{C_j}} x_{i,r}}{|\underline{C_j}|}$.
   - If $(\underline{C_j} = \emptyset$ and $\overline{C_j} - \underline{C_j} \neq \emptyset)$ then $z_{j,r} = \frac{\sum_{X_i \in (\overline{C_j} - \underline{C_j})} x_{i,r}}{|\overline{C_j} - \underline{C_j}|}$.
   - else $z_{j,r} = w_{low} \frac{\sum_{X_i \in \underline{C_j}} x_{i,r}}{|\underline{C_j}|} + w_{up} \frac{\sum_{X_i \in (\overline{C_j} - \underline{C_j})} x_{i,r}}{|\overline{C_j} - \underline{C_j}|}$.

   Here, $0 \leq w_{low} \leq 1$ and $0 \leq w_{up} \leq 1$ are the weights assigned to the lower approximation and the upper approximation of the rough cluster respectively, s.t. $w_{low} + w_{up} = 1$.

2. *Assigning objects to rough clusters*: Let $d(Z_j, X_i)$ be the distance between a data object X_i and the centroid Z_j of a rough cluster C_j.

   - Determine the nearest centroid Z_j, s.t. $d(Z_j, X_i) = min_{1 \leq j \leq k} d(Z_j, X_i)$ (see Fig. 7.3).
   - Determine the centroids Z_l's of the clusters that are also close to X_i. Let $T = \{l : d(Z_l, X_i)/d(Z_j, X_i) \leq \varepsilon \ and \ l \neq j\}$. Then,

   $$If\ T \neq \emptyset\ then\ [X_i \in \overline{C_j}\ and\ X_i \in \overline{C_l}, \forall l \in T]\ else\ [X_i \in \overline{C_j}\ and\ X_i \in \underline{C_j}].$$

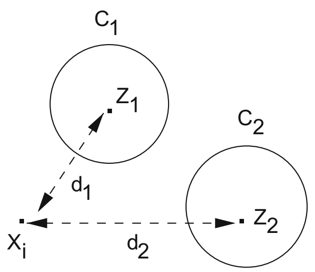


Fig. 7.3 Determining the nearest cluster centroid

Here, $\varepsilon \geq 1$ is a roughness parameter that determines the set T of near clusters to object X_i. If T is empty, then it means that the data object under consideration is close to only one of the rough clusters.

Numerous methods for discovering local outliers are produced with necessary refinements to the RKM algorithm. One of them works based on the idea that the importance of the degree of concentration of the objects in the area where they reside towards the centroid computation is different. Another method tries to identify objects that belong to one or more clusters by virtue of their position in the problem space, using evidential clustering principles.

7.4 Rough Clustering for Categorical Data

A majority of the rough clustering methods work with data described using numerical attributes. Therefore, it is pertinent to explore rough sets based mechanism for carrying out clustering on categorical data. As already mentioned, dealing with categorical data poses certain challenges in devising necessary computational measures needed to build any data processing algorithm. In view of this, it is mandatory to develop necessary algorithms by employing the occurrence frequencies of various values of the categorical attributes describing the data. Some of the algorithms developed based on this philosophy are presented here to perform rough clustering on categorical data.

For the purpose of describing the algorithms, consider a categorical data set $D = \{X_1, X_2, X_3, \ldots, X_n\}$ consisting of n objects and m categorical attributes. Each data object is represented as a multi-dimensional vector $X_i = [x_{i,1}, x_{i,2}, \ldots, x_{i,m}]$.

7.4.1 MMR Algorithm

The Min-Min-Roughness (MMR) algorithm is a rough sets based hierarchical clustering algorithm for categorical data. It is a top-down approach for clustering by iteratively dividing the set of objects D with the goal of achieving better clustering crispness. It determines the clustering attribute by Min-Roughness (MR) criterion as per the theoretical foundation given below.

As presented in Sect. 7.2.1, any categorical information system (IS) in which each row representing a data object can be expressed using $IS = (U, A, V, f)$. Considering a subset of attributes $B \subseteq A$, one can define an indiscernible relation $Ind(B) = \{(X_i, X_j) \in U \times U | \forall A_r \in B, x_{i,r} = x_{j,r}\}$, where $x_{i,r}$ and $x_{j,r}$ indicate the values of objects X_i and X_j corresponding to the attribute A_r. This indiscernible relation $Ind(B)$ divides the universe of objects U into a family of disjoint blocks called as equivalence classes. It should be noted that two objects belonging to the same equivalence class are indiscernible with respect to the attribute subset B.

For any subset of objects Y of the universe U, the accuracy of approximation of Y with respect to a subset of attributes $B \subseteq A$ is given by

$$\alpha_B(Y) = \frac{|\underline{B}(Y)|}{|\bar{B}(Y)|}. \tag{7.3}$$

Given an attribute $A_r \in A$, $V(A_r)$ refers to the set of values of A_r. Let Y be an equivalence class of $U|Ind(A_r)$, which means Y be a subset of objects having one specific value of $V(A_r)$. The roughness of Y with respect to $A_q \in A$, where $A_q \neq A_r$, is measured as follows.

$$R_{\{A_q\}}(Y) = 1 - \alpha_{\{A_q\}}(Y) = 1 - \frac{|\underline{A}_q(Y)|}{|\bar{A}_q(Y)|}. \tag{7.4}$$

Then, the mean roughness of attribute A_r with respect to attribute A_q, where $A_q \neq A_r$, is given by

$$Rough_{A_q}(A_r) = \frac{\sum_{Y \in U|Ind\{A_r\}} R_{\{A_q\}}(Y)}{|V(A_r)|} \tag{7.5}$$

Now, Min-Roughness (MR) of an attribute A_r is the minimum of the mean roughness measured with respect to all the attributes of the data.

$$MR(A_r) = min_{A_q \in A \wedge A_q \neq A_r}\{Rough_{A_q}(A_r)\} \tag{7.6}$$

The MR-criterion determines the best crispness possible with each attribute, and the algorithm determines the best candidate for split among all attributes. This algorithm takes the number of clusters (k) as input and terminates the iterative process when the number of clusters reaches this value.

The MMR algorithm working on the basis of above mathematical formulation involving rough sets has the following advantages.

1. It can handle uncertainty in data while performing clustering.
2. It is robust as it produces stable results with only one input (i.e., # clusters).
3. It can also handle large categorical data sets for clustering.

Typically, hierarchical clustering on data involves two important steps: (i) search for the partitioning attribute, (ii) selecting a leaf node for further clustering. MMR algorithm accomplishes this by employing the concept of roughness for partitioning and choosing the leaf node with more objects for further splitting. However, the following are the disadvantages associated with this clustering algorithm.

1. The roughness criterion used for partitioning cannot reflect the discernibility power to the boundary objects.
2. The leaf node selection mechanism of MMR algorithm may possibly generate undesirable clustering results.

7.4.2 RKModes Algorithm

Given a categorical data set D with n objects, the objective is to produce k rough clusters $\{C_1, C_2, \ldots, C_k\}$ represented by their modes $\{Z_1, Z_2, \ldots, Z_k\}$ respectively. The Rough k-modes (RKModes) algorithm is designed to meet this objective by modifying the basic k-modes method working with categorical data. The RKModes method identifies the inherent grouping structure of the data in accordance to the rough clustering principles.

The flow of computation constituting this algorithm is shown in Fig. 7.4, indicating an iterative process. Also, the requisite symbolic notation for describing this algorithm is furnished in Table 7.1 confining to rough set theory.

Clustering with categorical data using rough sets requires certain computational measures suitably defined on such data to perform the following steps.

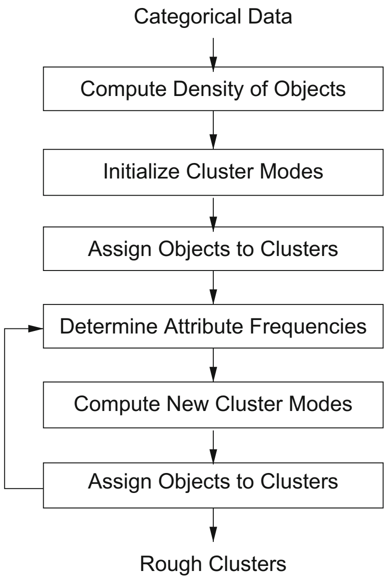

Categorical Data
Compute Density of Objects
Initialize Cluster Modes
Assign Objects to Clusters
Determine Attribute Frequencies
Compute New Cluster Modes
Assign Objects to Clusters
Rough Clusters

Fig. 7.4 Flow of computation with RKModes algorithm

Table 7.1 Symbolic notation used in RKModes algorithm

Symbol	Meaning
n	Number of data objects in D
m	Dimensionality of the data
k	Number of clusters desired
X_i	ith object in the data set D
C_j	jth cluster in the clustering process
Z_j	Representative/mode of jth cluster
$\overline{C_j}$	Upper approximation of cluster C_j
$\underline{C_j}$	Lower approximation of cluster C_j
ε	Parameter determining cluster assignment
w_{low}	Weight of the lower approximation
w_{up}	Weight of the upper approximation

- *Dissimilarity measure over categorical data*: This is required to measure the distance between a data object X_i and a cluster C_j (with its mode Z_j) for assigning data objects to their nearest rough clusters. This is accomplished by using the distance measure given in Eq. 4.6 that works with categorical data.
- *Determining new cluster centroids (modes)*: The procedure suggested in Sect. 7.3.1 is modified by incorporating the frequency counts of various categorical attribute values of the objects assigned to distinct approximations of the rough clusters during the iterative process.

Similar to other algorithms belonging to the k-means family, RKModes method also has to deal with the issue of cluster initialization. In this regard, the cluster initialization method described in Sect. 5.2.4.1 is considered. This method works with categorical data by computing the density of various data objects as given in Eq. 4.7.

In addition to the symbolic notation given in Table 7.1, let $freq_j^{low}(x_{i,r})$ and $freq_j^{up}(x_{i,r})$ denote the number of objects in the lower and upper approximations of j^{th} cluster respectively with the value $x_{i,r}$ for the attribute A_r. These representations are required to formulate the RKModes algorithm consisting of the following computational steps.

1. Compute $density(X_i)$ of each data object $X_i \in D$.
2. Determine the initial set of k cluster modes $Z = \{Z_1, Z_2, \ldots, Z_k\}$.
3. Assign data objects to rough clusters.
    - Compute cluster-to-object distance $d(Z_l, X_i)$ as per Eq. 4.6.
    - Determine object assignment as given in Sect. 7.3.1.
4. Count cluster-wise attribute value frequencies $freq_j^{low}(x_{i,r})$ and $freq_j^{up}(x_{i,r})$.
5. Compute the new mode $Z_j^* \leftarrow X_i$ of each cluster C_j, s.t. $max_{X_i \in \underline{C_j}} density_j(X_i)$, where $density_j(X_i)$ is the density of data object X_i w.r.t. cluster C_j given by

- if ($\underline{C_j} \neq \emptyset\ and\ (\overline{C_j} - \underline{C_j}) = \emptyset$) then $density_j(X_i) = \frac{1}{m}\sum_{r=1}^{m}(\frac{freq_j^{low}(x_{i,r})}{|\underline{C_j}|})$.
- if ($\underline{C_j} = \emptyset\ and\ \overline{C_j} \neq \emptyset$) then $density_j(X_i) = \frac{1}{m}\sum_{r=1}^{m}(\frac{freq_j^{up}(x_{i,r})}{|\overline{C_j}|})$.
- else $density_j(X_i) = \frac{1}{m}\sum_{r=1}^{m}(w_{low}\frac{freq_j^{low}(x_{i,r})}{|\underline{C_j}|} + w_{up}\frac{freq_j^{up}(x_{i,r}) - freq_j^{low}(x_{i,r})}{|\overline{C_j} - \underline{C_j}|})$.
  where $w_{low} + w_{up} = 1$.

6. Repeat steps 3–5 until convergence (cluster assignment doesn't change).
7. Output k rough clusters with their final modes.

The iterative process in the above algorithm converges either after repeating for a predefined number of times or if the cluster assignment of objects doesn't change much over successive iterations. It is important to note that there is no explicit overall objective function computed by this algorithm. It only tries to maximize the density values of the modes of the clusters until convergence. Also, depending on the specific values set for the two parameters w_{up} and w_{low} of the algorithm, the outcome of the clustering process may vary significantly. Applying necessary simplification on the algorithmic steps, the computational complexity of the RKModes method is found to be $O(nm)$.

7.5 Experiments on Benchmark Data

As stated at the beginning of this chapter, the grand objective is to detect outliers in categorical data by employing clustering-based methods. This in turn is expected to exploit the benefits offered by rough clustering. In line with this objective, an empirical study is attempted here in two parts: (i) clustering of categorical data using RKModes algorithm, and (ii) determine outliers from the clustering output using an acceptable outlier detection method.

This experimental study is carried out by employing six frequently used categorical data sets taken from UCI ML Repository, as per the details furnished in Table 7.2. To enable outlier detection on these data sets, the standard data preparation procedure is applied resulting in the processed data.

7.5.1 Outcome of Data Clustering

The utility of a clustering algorithm can be evaluated using a well known measure named as *overall purity of clusters*. The *purity* of a cluster C_j is given by p_j/n_j, where p_j is the number of objects with class label that dominates cluster C_j and n_j is the number of objects in that cluster. Then, the overall purity is given by $\sum_{j=1}^{k} p_j/n$, where n is the total number of objects. As per this measure, a higher value of overall purity indicates a better clustering result, with perfect clustering yielding a value of 1.0 (100%).

Table 7.2 Details of benchmark categorical data sets

Name of data set	Dimension	Outlier class	# Normal objects	# Outlier objects
Chess (KR/KP)	36	Nowin	1669	305
Tic-Tac-Toe	9	Negative	626	66
Breast Cancer (W)	10	4(malignant)	444	47
Congressional Votes	16	Republican	124	21
Breast Cancer	9	Recurrence-events	196	16
Mushroom	22	Poisonous	4208	783

As shown in Fig. 7.1, some of the data objects get assigned to the boundary regions of the clusters during rough clustering. Such objects cannot be considered in the inference process due to the undecidability regarding their cluster membership. Hence, the objects that belong to a cluster with certainty (lower approximation) only determine its purity. Accordingly, the overall purity of rough clustering is given by $\sum_{j=1}^{k} p_j / \sum_{j=1}^{k} n_j$, where n_j is the number of objects in the lower approximation of cluster C_j.

To enable this investigation, two of the roughness parameters of the RKModes algorithm are set to fixed values as: $w_{low} = 0.7$ and $w_{up} = 0.3$. Similarly, the other roughness parameter ε needed for assigning objects to rough clusters is also set to a fixed value 1.1. Clustering results obtained with only two categorical data sets are presented here for brevity. The results corresponding to Mushroom data in terms of the purity of each cluster, are furnished in Table 7.3. Similarly, clustering results of Breast Cancer (Wisconsin) data are shown in Table 7.4.

For a comparative view of the performance, two more rough clustering algorithms namely the Min-Min-Roughness (MMR) method and the Maximum-Total-Mean-Distribution-Precision (MTMDP) method are included in this experimentation. As already discussed, the MMR method produces rough set-based hierarchical clustering of data by determining the clustering attribute using the MR (Min-Roughness) criterion. Similarly, the MTMDP method also performs hierarchical clustering of data based on probabilistic rough set theory, with significantly better clustering results over MMR method.

The overall purity values obtained with rough clustering algorithms along with that of the standard k-modes algorithm are shown in Table 7.5. The empirical observations indicate superior performance of the RKModes algorithm over standard k-modes method. One can attribute this performance improvement to the use of rough sets for clustering, as the core computations are similar in both the cases.

Table 7.3 Clustering results on Mushroom data (with $k = 20$)

Cluster number	# Objects	Class-wise distribution		Purity (%)
		Poisonous	Edible	
1	602	0	602	100
2	614	614	0	100
3	288	0	288	100
4	1296	1296	0	100
5	768	0	768	100
6	56	8	48	85.7
7	101	5	96	95
8	192	0	192	100
9	48	0	48	100
10	288	288	0	100
11	192	0	192	100
12	664	0	664	100
13	36	36	0	100
14	610	610	0	100
15	258	184	74	71.3
16	225	255	0	100
17	16	0	16	100
18	32	32	0	100
19	648	0	648	100
20	104	72	32	69.2

Table 7.4 Clustering results on breast cancer data (with $k = 2$)

Cluster number	# Objects	Class-wise distribution		Purity (%)
		Benign	Malignant	
1	490	448	42	91.4
2	68	0	68	100

Table 7.5 Performance comparison of clustering algorithms

Data set name	# clusters (k)	% Overall purity obtained			
		RKModes	MTMDP	MMR	k-modes
Breast cancer (W)	2	**92.5**	88	79	85.12
Mushroom	20	**98.3**	98	84	93.45

Table 7.6 Outlier detection results using Rough ROAD framework

Data set	# Outliers	# Outliers detected	
		Rough ROAD	ROAD
Chess (KR/KP)	305	**131**	126
Tic-Tac-Toe	66	**38**	37
Breast Cancer (W)	47	**44**	42
Congressional Votes	21	**19**	18
Breast Cancer	16	**6**	5
Mushroom	783	**609**	575

7.5.2 *Outlier Detection Results*

Detection of outliers based on the rough clustering output is carried out by employing the Ranking-based Outlier Analysis and Detection (ROAD) algorithm. Of the two ranking schemes constituting this algorithm, one of them involves a clustering step using the standard k-modes method. In place of this method, the current study uses the RKModes method with the ROAD framework, hereafter referred to as 'Rough ROAD'. Outlier detection results obtained on benchmark data sets using this framework are furnished in Table 7.6.

For a comparative view, the performance of the detection algorithm in both the settings (with and without rough clustering) is presented using the ROC curves, as shown in Fig. 7.5. This assessment indicates that detecting outliers based on rough clustering has indeed better performance across all benchmark data sets considered in this study.

7.5.2.1 Sensitivity Analysis

There are three roughness parameters associated with the RKModes algorithm. Varying the value of parameter 'ε' has its impact on the rough characteristics of the algorithm, thereby on the overall outlier detection performance. To illustrate this impact, further experimentation is carried out on Mushroom data. Fig. 7.6 shows the number of outliers detected corresponding to different values of 'ε'. The other roughness parameters are set to fixed values ($w_{low} = 0.7$ *and* w_{up} = 0.3). Similarly, the parameter specific to ROAD framework is set as $\alpha = 2$.

In continuation of this investigation, the impact of the other roughness parameters, namely w_{low} and w_{up} is explored. As per the ROAD framework, the clustering step is responsible for detection of Type-2 outliers. Hence, the impact is demonstrated by furnishing the number of Type-2 outliers detected corresponding to various values of w_{low} on Breast Cancer (Wisconsin) data, as shown in Fig. 7.7. This study indicates that a higher value of weight assigned to lower approximation of rough clusters has

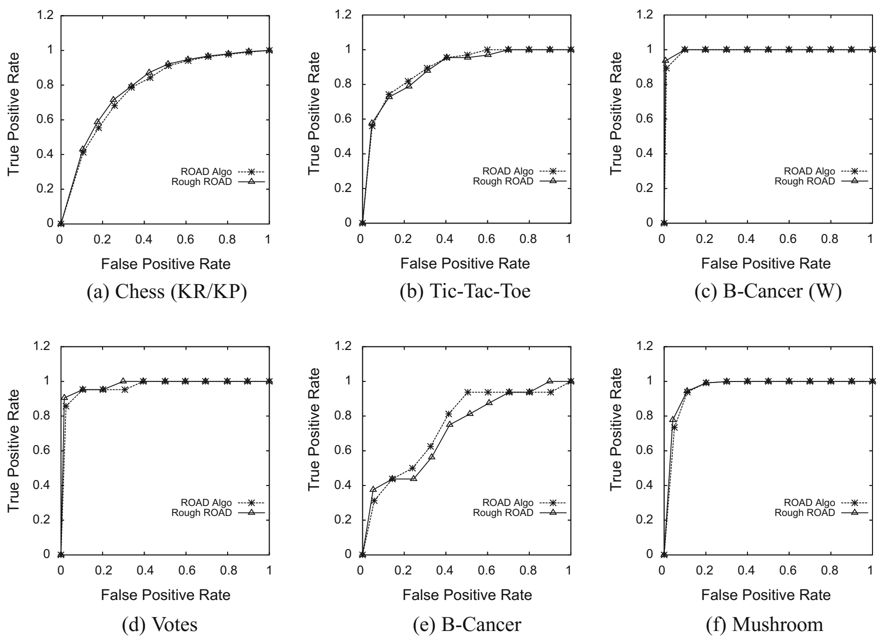
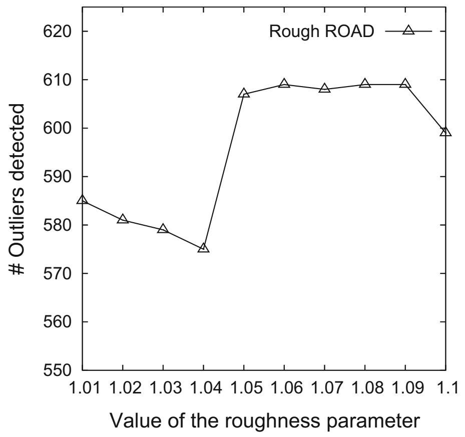


Fig. 7.5 Performance comparison of the rough ROAD framework

Fig. 7.6 Outlier detection performance on Mushroom data with varying ε



enabled detection of more Type-2 outliers. However, this behavior may not be consistent across various data sets as it can vary depending on the intrinsic characteristics of the data.

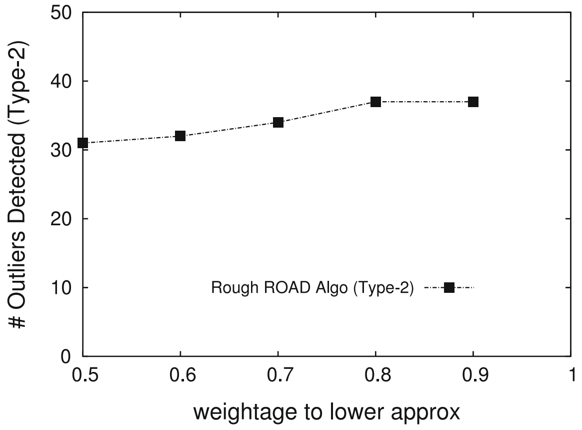


Fig. 7.7 Number of Type-2 outliers detected with varying w_{low}

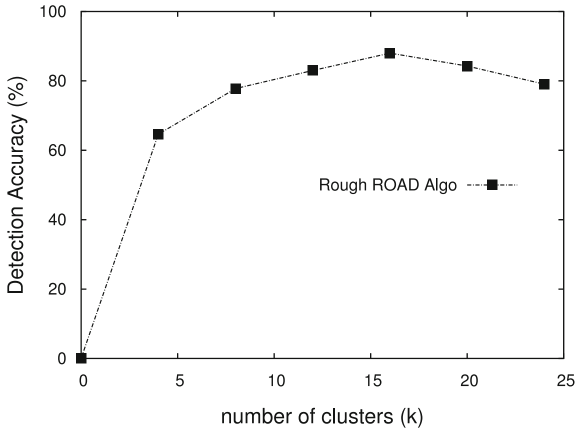


Fig. 7.8 Outlier detection accuracy with varying k value

7.5.2.2 Effect of Number of Clusters

It is known that the performance of the k-means family of algorithms is severely affected by the value assigned to k, namely the number of clusters. So, the performance of RKModes algorithm for outlier detection (with Rough ROAD framework) corresponding to different values of k on Mushroom data set is shown in Fig. 7.8. Similar to sensitivity analysis, the number of Type-2 outliers detected in this process is indicated in Fig. 7.9 for a better understanding of the impact of varying k value. This experimentation suggests that the performance peaks for certain k value and decreases later on as the clusters get degenerated.

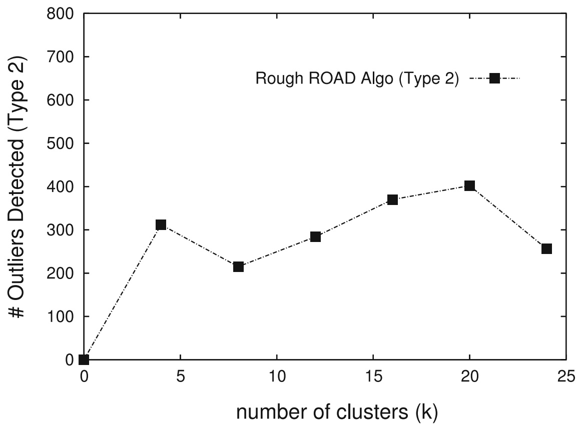


Fig. 7.9 Number of Type-2 outliers detected with varying k value

7.6 Summary

Among various methods for detecting outliers in data, clustering-based ones are more frequently used. The uncertainty prevailing in clustering data involving outliers is addressed by considering a soft computing approach based on rough sets theory. In this connection, some of the recent methods developed based on this philosophy are presented. The RKModes algorithm works based on the standard k-modes algorithm by incorporating rough sets principles for clustering categorical data. The superior clustering performance of the RKModes algorithm in turn leading to improved outlier detection result (using ROAD framework) is evident from the experimental study carried out over benchmark data sets. The sensitivity analysis included as a part of this study indicates the impact of various roughness parameters on the algorithm performance.

7.7 Bibliographic Notes

To deal with uncertainty regarding the class membership of data objects, one can consider the concept of granulation defined based on soft computing approaches like rough sets, fuzzy sets [15]. So, soft computing approaches have become preferred choices when it is required to handle uncertainty/vagueness [1, 7, 12, 16]. Rough set theory [18] was basically developed to deal with vagueness in the data. For the purpose of data analysis, rough set theory does not need any preliminary information about data, like probability distributions in statistics, a grade of membership in fuzzy set theory [20].

As per rough sets theory, an information system can be represented as a quadruple $IS = (U, A, V, f)$. Based on this model, the notion of sequence-based outlier was conceptualized in [5] along with an expression for computing the outlier factor of such an object with deviating behavior. The definition of discernibility matrices was proposed in [21]. Given an information system, it is possible to calculate the corresponding discernibility matrix indicating the pair-wise distances among the data objects.

The idea of developing a rough clustering methodology was initially thought of in the context of web mining [9]. The requirement was to deal with bad/ incomplete data arising in web browsing and web logging domain due to a variety of reasons inherent to that domain. As a result, RKM algorithm was developed for carrying out rough clustering [11] by modifying the k-means clustering algorithm [4]. Subsequently, various refinements of the RKM algorithm were brought out [10, 19]. In connection with rough clustering, the fundamental properties of rough sets to be satisfied by the upper and lower approximations of a cluster were given in [19].

A method of discovering local outliers based on rough clustering was proposed [14] by refining RKM algorithm. The refinement was based on the idea that the importance of the degree of concentration of the objects in the area where they reside towards the centroid computation is different. As per this idea, the density of each data object is computed using the Gauss kernel function. Then, the cluster centroids are calculated using the density-based weight values of the objects. Thus, this method performs a clustering of the given data. Subsequently, the outlier score of each object in a certain cluster is computed to determine various cluster-based local outliers. However, the weight computation using the Gauss kernel function is fit only for numeric data.

Another enhancement of RKM algorithm was proposed [6] for outlier detection based on the evidential clustering principles [13]. Evidential clustering identifies objects that belong to one or more clusters by virtue of their position in the problem space. As this algorithm also works with data described using numerical attributes only, there is scope for developing a suitable method to detect outliers in categorical data.

Motivated by this requirement, the RKModes algorithm [22, 24] was developed based on the k-modes methodology [3] by incorporating the rough clustering principles. In an earlier effort, the Min-Min-Roughness (MMR) algorithm [17] was developed to produce rough set-based hierarchical clustering of data. This method determines the clustering attribute (for splitting the clusters) using the Min-Roughness criterion. Similarly, Maximum-Total-Mean-Distribution-Precision (MTMDP) algorithm [8] also performs hierarchical clustering of data based on probabilistic rough set theory, with significantly better clustering results over MMR method.

To explore the effectiveness of the rough clustering method towards outlier detection, the ROAD algorithm [23] was considered for determining outliers in categorical data based on the clustering results produced by the RKModes algorithm [24]. An experimental study was carried out in this regard on benchmark categorical data sets taken from UCI ML Repository [2] and the results obtained were encouraging.

References

1. Asharaf, S., Murty, M.N., Shevade, S.K.: Rough set based incremental clustering of interval data. Pattern Recogn. Lett. **27**, 515–519 (2006)
2. Dua, D., Efi, K.T.: UCI machine learning repository (2017). URL http://archive.ics.uci.edu/ml
3. Huang, Z.: A fast clustering algorithm to cluster very large categorical data sets in data mining. In: SIGMOD Data Mining and Knowledge Discovery Workshop, pp. 1–8 (1997)
4. Jain, A.K.: Data clustering: 50 years beyond K-means. Pattern Recogn. Lett. **31**, 651–666 (2010)
5. Jiang, F., Sui, Y., Cao, C.: Some issues about outlier detection in rough set theory. Expert Syst. Appl. **36**, 4680–4687 (2009)
6. Joshi, M., Lingras, P.: Enhancing rough clustering with outlier detection based on evidential clustering. In: RSFDGrC. LNCS, vol. 8170, pp. 127–137. Springer, Berlin (2013)
7. Kaiiali, M., Wankar, R., Rao, C.R., Agarwal, A.: A rough set based PCM for authorizing grid resources. In: 10th International Conference on Intelligent Systems Design and Applications (ISDA), pp. 391–396. Cairo, Egypt (2010)
8. Li, M., Deng, S., Wang, L., Feng, S., Fan, J.: Heirarchical clustering algorithm for categorical data using a probabilistic rough set model. Knowl. Based Syst. **65**, 60–71 (2014)
9. Lingras, P.: Rough set clustering for web mining. In: IEEE FUZZ, pp. 1039–1044 (2002)
10. Lingras, P., Peters, G.: Applying rough set concepts to clustering. In: Rough Sets: Selected Methods and Applications in Management and Engineering, pp. 23–38. Springer, London (2012)
11. Lingras, P., West, C.: Interval set clustering of web users with rough k-means. J. Intell. Inf. Syst. **23**(1), 5–16 (2004)
12. Maji, P., Pal, S.K.: Fuzzy-rough sets for information measures and selection of relevant genes from microarray data. IEEE Trans. Syst. Man Cybern. Part B **40**(3), 741–752 (2010)
13. Masson, M., Denoeux, T.: ECM: an evidential version of the fuzzy c-means algorithm. Pattern Recogn. **41**, 1384–1397 (2008)
14. Mi, H.: Discovering local outlier based on rough clustering. In: 3rd International Workshop on Intelligent Systems and Applications (ISA), pp. 1–4. IEEE (2011)
15. Nguyen, H.S., Pal, S.K., Skowron, A.: Rough sets and fuzzy sets in natural computing. Theor. Comput. Sci. **412**(42), 5816–5819 (2011)
16. Pal, S.K., Ghosh, A.: Guest editorial: soft computing data mining. Inf. Sci. **163**, 1–3 (2004)
17. Parmar, D., Wu, T., Blackhurst, J.: Mmr: an algorithm for clustering categorical data using rough set theory. Data Knowl. Eng. **63**, 879–893 (2007)
18. Pawlak, Z.: Rough sets. Int. J. Comput. Inf. Sci. **11**, 341–356 (1982)
19. Peters, G.: Some refinements of rough k-means clustering. Pattern Recogn. **39**, 1481–1491 (2006)
20. Skowron, A., Jankowski, A., Swiniarski, R.W.: 30 years of rough sets and future perspectives. RSFDGrC. LNCS, vol. 8170, pp. 1–10. Springer, Halifax, Canada (2013)
21. Skowron, A., Rauszer, C.: The discernibility matrices and functions in information systems. In: Slowinski, R. (ed.) Intelligent Decision Support—Handbook of Applications and Advances of the Rough Sets Theory, pp. 331–362 (1992)
22. Suri, N.N.R.R., Murty, M.N., Athithan, G.: A rough clustering algorithm for mining outliers in categorical data. In: Engel, P.M. (ed.) 5th International Conference on Pattern Recognition and Machine Intelligence (PReMI). LNCS, vol. 8251, pp. 170–175. Springer, Berlin (2013)
23. Suri, N.N.R.R., Murty, M.N., Athithan, G.: A ranking-based algorithm for detection of outliers in categorical data. Int. J. Hybrid Intell. Syst. (IJHIS) **11**(1), 1–11 (2014)
24. Suri, N.N.R.R., Murty, M.N., Athithan, G.: Detecting outliers in categorical data through rough clusteringa. Nat. Comput. **15**, 385–394 (2016)

Part II
Applications of Outlier Detection in Graph Data Mining

This part addresses some of the emerging applications of outlier detection principles in analyzing graph/network data. In particular, the idea is to detect various subgraphs of the input graph/network data that display deviating characteristics when compared to other parts of the input graph. Anomalous subgraphs of this kind are of great interest for analyzing various kinds of network data such as IP-based computer networks, biological networks, social interactions over the web, etc. With increasing number of real life applications dealing with network data, the anomaly detection task is becoming more significant. Accordingly, the kinds of challenges posed by these applications are also on the rise. To cope up with this ever increasing demand for developing innovative techniques to deal with varied application scenarios, it becomes imperative to continuously devise novel methods in a pragmatic manner.

In line with the above deliberations, some emerging application scenarios along with suitable techniques for anomaly detection are presented in this part. In specific, this part is divided into three contributory chapters discussing different algorithmic methods for analyzing both static and dynamic network applications.

Chapter 8
Mining Anomalies in Graph Data

Abstract Mining graph data is an important data mining task due to its significance in network analysis and several other contemporary applications. With this backdrop, this chapter explores the potential applications of outlier detection principles in graph/network data mining for anomaly detection. One of the focus areas is to detect arbitrary subgraphs of the input graph exhibiting deviating characteristics. In this direction, graph mining methods developed based on latest algorithmic techniques for detecting various kinds of anomalous subgraphs are explored here. It also includes an experimental study involving benchmark graph data sets to demonstrate the process of anomaly detection in network/graph data.

8.1 Introduction

Graph mining is one of the most explored and continuously evolving research areas in the domain of data mining. Graph mining is popular across a slew of applications, as graph-based representation is the primitive structure to model several real world applications in a natural way. Such a representation reduces the complexity involved in understanding a larger problem and thus helps in finding an acceptable solution in a seamless manner.

Among various established graph mining tasks, the significant ones are frequent subgraph discovery, identification of dense subgraphs, graph pattern matching, community/group detection.

1. *Subgraph Discovery*: Subgraph discovery is a well explored task in the graph mining area that is further divided into various specialized tasks depending on the features of the subgraphs to be discovered. Two related tasks in this regard are *frequent subgraph discovery* and *dense subgraph discovery*.

    - Frequent subgraph discovery (FSD) is defined as the process of finding out the subgraphs with multiple instances within a given graph or a set of graphs. The complexity of this task is evident from the diversity of approaches employed in developing various algorithms for this purpose. These algorithms are exten-

N. N. R Ranga Suri et al., *Outlier Detection: Techniques and Applications*,
Intelligent Systems Reference Library 155,
https://doi.org/10.1007/978-3-030-05127-3_8

sively used in various application domains such as computational biology (for discovering motif structure of protein interactions and finding residue packing patterns from protein structures) and in chemical carcinogenesis. These algorithms are found to be useful in security threat assessment and analysis of information networks as well.

- Similarly, dense subgraph discovery (DSD) looks for components of a graph with high edge density in comparison to all other subgraphs of the graph. Dense regions in a graph have a number of useful characteristics such as having small diameters and being robust, that make them attractive in several complex applications modeled using graph-based representation.

2. *Graph Pattern Matching*: Graph matching is the process of finding the occurrences of a given pattern (also known as query graph) in a large reference graph. The size of the pattern graph with respect to the reference graph is expected to be small. Labeled graphs simplify the process of graph matching as the initial condition for a vertex match would be the label match between them, which in turn would prune out many unfit candidates. This is followed by an assessment of the structural similarity, which is the key process for finding a match in unlabeled graphs. Different algorithmic methods exist for establishing structural similarity on graph data trying to find a match, if one exists.
3. *Community Detection in Graphs*: Detecting communities in graph data is an important graph mining task in the context of social networks. According to Network theory, communities in a graph represent groups of nodes with dense connectivity within each group. Numerous methods are developed for this purpose based on various algorithmic approaches such as network modularity, centrality measures, eigenvectors, resistor networks, and simulated annealing.

8.1.1 Anomalies in Graph Data

An emerging research problem related to graph mining is to discover anomalies in graph data. The objective is to identify the subgraphs displaying anomalous characteristics in the given graph. From the analysis perspective, various graph anomaly patterns may be of interest in different application contexts as listed below.

1. For example, a *star* structured subgraph may indicate a telemarketer, or a port scanner scenario in computer networks. These are some typical scenarios that an analyst may be interested in graph data. However, in a large graph, it may not be possible to find a subgraph with exact star connectivity. Hence, any subgraph with almost a similar connectivity structure is a good enough pattern of interest. Such a pattern is named as *near star* subgraph as shown in Fig. 8.1a.
2. Similarly, a *clique* structured subgraph may indicate a close group of individuals in a social network. Consequently, a *near clique* subgraph as shown in Fig. 8.1b may be another pattern of interest.

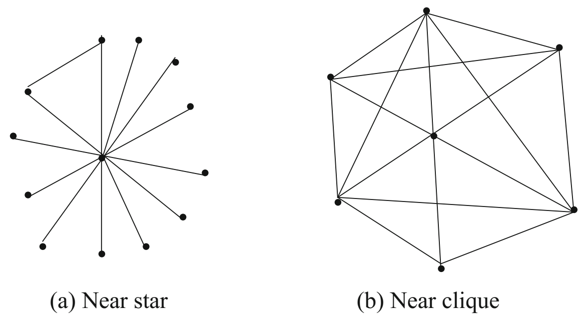


Fig. 8.1 Sub-graph anomalies of interest in graph data

3. Likewise, a hierarchical structure among a set of nodes in a communication network forms a *tree* like pattern. In this connection, a *near tree* subgraph turns out to be of interest.

Thus, there may exist a number of interesting subgraph connectivity patterns in graph representation of real life data. Anomaly detection in graphs is aimed at detecting various such interesting subgraph patterns with implicit semantics, making it an important graph mining task.

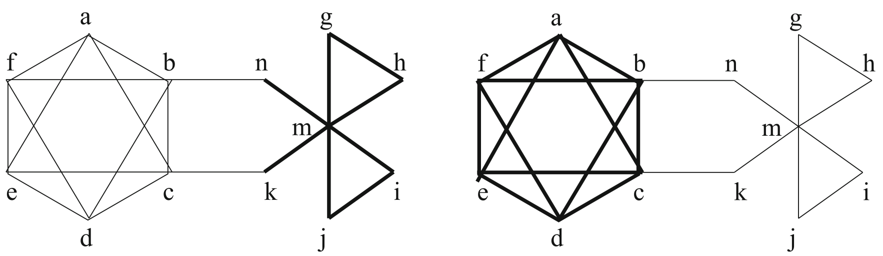
8.2 Analyzing Graph/Network Data

Graph/Network data is ubiquitous with real life applications due to the proliferation of Internet and networking technologies. As a result, one would need the state-of-the-art techniques for analyzing such data to draw meaningful observations on the respective application process and to improve it further. In this connection, it becomes imperative to have efficient algorithmic schemes so as to keep pace with the rapidly changing technologies. As networks are generally modeled using graph-based representation, the discussion here makes use of these terms ‘network’ and ‘graph’ as synonyms.

There are a slew of methods to perform graph data analysis catering to varied application contexts. Typical analysis process looks at individual nodes, edges, and their connectivity structure and also substructure level view depending on the need. Some of the well known analysis methods on graph data are presented below for brevity.

8.2.1 Neighborhood-Based Analysis

Exploring the neighborhood is a means of understanding the inter-connectivity present in graph/network data and drawing cues for further analysis of the data.



(a) Egonet of node *m* (bold lines) (b) An arbitrary subgraph (bold lines)

Fig. 8.2 Examples of (anomalous) subgraphs of different types

The collection of links/edges incident on a node in a graph happens to be the atomic entity for consideration towards characterizing the neighborhood.

When attempting to carry out neighborhood based analysis on graph data, the notion of *egonet* (short form of *ego-network*) appears handy. Formally, the ego-network of a node (*ego*) is defined as the induced subgraph of its 1-step neighbors, as shown in Fig. 8.2a.

Basically, egonet encompasses the nodes that are directly connected (first-level) to the node under consideration along with the edges incident among them. Expanding further, one can explore the 2-step (second-level) neighbors of a node which are incident on its 1-step neighbors. Likewise, it is possible to establish 'k-step neighborhood' of a node considering all its neighbors up to k-steps away and all the connections among these nodes. Given that real-world graphs have small diameter, a small value of k is good enough to reveal the required information about the network structure. Thus, the multi-level neighborhood view acts as first hand analysis on graph data emphasizing the inherent connectivity/link structure.

As per the 'OddBall' method for anomaly detection in graphs, a number of features (such as number of nodes, one or more eigenvalues, number of triangles, etc.) can be used to characterize the neighborhood (egonet) of a node. Thus, every node becomes a point in the feature space enabling multi-dimensional analysis on the same. However, the idea is to select features that will lead to patterns of normal behavior (power laws) that most nodes obey, expect for a few anomalous nodes. Informally, a node (ego) of a given network is anomalous if its neighborhood significantly differs from those of the others.

Among various observed patterns characterizing normal behavior of real graphs, the important one is defined based on the number of nodes (N_i) and the number of edges (E_i) present in an egonet G_i. This pattern essentially implies that there exists the following power law relationship between N_i and E_i, named as the Egonet Density Power Law (EDPL).

$$E_i \propto N_i^{\alpha}, \quad 1 \leq \alpha \leq 2. \tag{8.1}$$

Typical value of the exponent α is found to be in the range [1.10, 1.66] for real data. A scatter plot of E_i versus N_i can be produced by establishing the least squares (LS) fit on the median values corresponding to each bucket of points given by the logarithmic binning on the N_i axis. A point on the line with slope '2' indicates a clique subgraph. Similarly, a point on the line with slope '1' represents a star subgraph. Consequently, the elementary anomalous subgraphs such as *near cliques* and *near stars* can be identified by subjecting the egonets of the nodes to satisfy the EDPL power law. It is possible to measure the amount of deviation from the power law relationship to assess the significance of each such anomalous subgraph.

In addition to these two varieties of anomalous subgraphs, the 'OddBall' method caters to the detection of other graph anomalies such as *heavy vicinity* and *dominant edges* in weighted graphs. However, the scope of this chapter is limited to the discussion on simple undirected graphs without weights, mutiple edges, etc.

8.2.2 Community Detection for Analysis

Accordingly to Network theory, community structure exploration is considered as the fundamental analysis technique that sheds light on the internal structure of a large-scale network. In this connection, communities in a network/graph represent densely connected groups of vertices, with only sparser connections between groups. An illustration of communities present in a sample network is shown in Fig. 8.3.

The ability to detect such groups is significant in several real life applications. For example, groups within the world-wide-web might correspond to sets of web pages on related topics. Similarly, if a metabolic network is divided into groups, it could

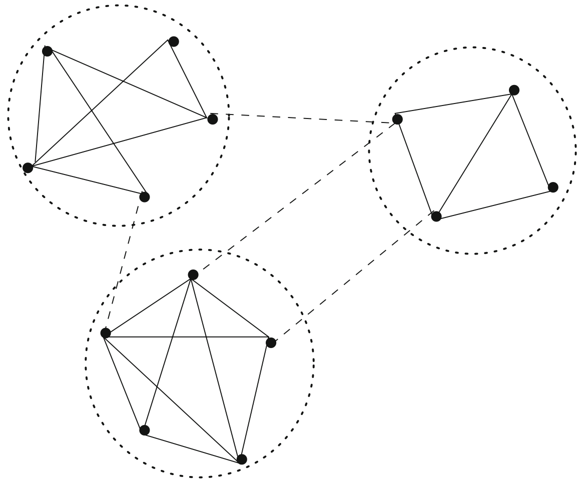


Fig. 8.3 Typical communities in a sample graph/network data

provide evidence for a modular view of the dynamics of the network. Thus, finding tightly knit groups in a network can convey useful information for its analysis.

There are two main research directions for discovering groups in networks: (i) graph partitioning, and (ii) block modeling, hierarchical clustering, or community structure detection. In principle, both these lines of thought are solving the same problem, albeit by somewhat different means. The community structure problem differs from graph partitioning in that the sizes of the communities are not normally known in advance.

It is assumed that the network of interest divides naturally into subgroups and the analyst's job is to find these groups. The number and size of the groups are thus determined by the network itself. One has to admit the possibility that no good division of the network exists and an outcome is considered interesting based on the information that it provides about the network topology.

A good division of a network into communities is the one in which there are fewer than expected edges across communities. True community structure in a network can be quantified by the *modularity* measure. Modularity of a network is the number of edges falling within groups minus the expected number in an equivalent random graph/network. The modularity can be either positive or negative, with positive values indicating a possible community structure. Thus, one can search for community structure in networks having (large) positive modularity, if it exists.

In addition to the modularity based approach for community detection, there exist various other methods employing different techniques such as minimum-cut, hierarchical clustering, Girvan-Newman approach, statistical inference, etc.

8.3 Basics of NMF Technique

The Non-negative Matrix Factorization (NMF) technique is inspired by psychological and physiological evidence for 'parts-based representations of objects' in the human brain. Based on this philosophy, it is designed to produce factors of a matrix representing a set of objects as $G \approx WH$.

While other factorization methods like Principal Component Analysis (PCA) and Vector Quantization (VQ) learn holistic objects, the NMF technique learns localized features that correspond to parts of the object. The non-negativity constraints lead to parts-based representation by allowing only additive combinations to form a whole object.

Given a set of n objects represented using m attributes, the data matrix G will be of $n \times m$ size. This matrix is then approximately factored into an $n \times r$ matrix W and an $r \times m$ matrix H. Usually the rank r of the factorization is chosen so that $(n + m)r < nm$, resulting in a compressed version of the original data matrix. One can understand from this factorization that each object vector is approximated by a linear combination of the columns of W, with the weights given by H matrix. Therefore, W can be regarded as containing a basis that is optimized for the linear approximation of the given data. Good approximation can be achieved if the basis vectors discover structure that is latent in the data. Each column of H is called an encoding, and is in one-to-one correspondence with an object in the input set. In this manner, the NMF technique can be applied for statistical analysis of multivariate data.

An implementation of the NMF method consists of iterative update rules for W and H matrices as given below.

$$W_{ik} \leftarrow W_{ik} \sum_{j} \frac{V_{ij}}{(WH)_{ij}} H_{kj} \text{ and } H_{kj} \leftarrow H_{kj} \sum_{i} W_{ik} \frac{V_{ij}}{(WH)_{ij}} \tag{8.2}$$

The iterative approximation process converges to a local maximum of the following objective function ensuring non-negativity constraints. It is guaranteed to converge because this function is convex in H given W.

$$F = \sum_{i=1}^{n} \sum_{j=1}^{m} [G_{ij} log(WH)_{ij} - (WH)_{ij}] \tag{8.3}$$

The above objective function measures the likelihood of generating the objects in G from the basis W and encoding H. The exact form of the objective function is not as crucial as the non-negativity constraints for the success of the NMF algorithm in learning parts. However, the update rules given above for W and H are specific to this objective function. For example, a squared error objective function leads to a set of update rules different from the ones given above.

NMF technique has gained practical significance due to its powerful interpretability and close relationship with the clustering methods. Sparseness of the data is another important aspect managed effectively by the NMF technique.

8.3.1 NMF Method for Community Detection

There exist numerous methods for detecting communities in graph representation of network data based on various algorithmic strategies. However, the one employing NMF technique has caught the attention of researchers due to its computational advantages. Some of the striking features of NMF that make it suitable for the community detection task are as follows.

- Intuitively, it makes more sense for each node of graph G to be associated with some small subset of a large number of communities, rather than just one community or all the communities.
- The NMF representation contains both a basis and encoding that are naturally sparse, which is crucial for a parts-based representation.

Based on the above discussion, it is pertinent to consider NMF technique as the most applicable one for community detection. Given an input graph with N nodes, its adjacency matrix (G) captures the pair-wise connectivity among all the nodes of the graph. In order to discover the communities present in this graph, its adjacency matrix can be factorized as $G \approx WH$, by subjecting it to the NMF procedure. To do this, it is required to assign a desired value for the number of communities (indicated by k) to be detected. This basically produces a mapping W of the nodes in the graph to various communities $\{G_1, G_2, ..., G_k\}$ with varying degree of membership as determined by the factorization process.

$$\begin{bmatrix} g_{11} & g_{12} & \cdots & g_{1N} \\ g_{21} & g_{22} & \cdots & g_{2N} \\ \cdots & \cdots & g_{ij} & \cdots \\ g_{N1} & g_{N2} & \cdots & g_{NN} \end{bmatrix} \approx \begin{bmatrix} w_{11} & w_{12} & \cdots & w_{1k} \\ w_{21} & w_{22} & \cdots & w_{2k} \\ \cdots & \cdots & w_{ij} & \cdots \\ w_{N1} & w_{N2} & \cdots & w_{Nk} \end{bmatrix} \times \begin{bmatrix} h_{11} & h_{12} & \ldots & h_{1N} \\ h_{21} & h_{22} & \ldots & h_{2N} \\ \ldots & \ldots & h_{ij} & \ldots \\ h_{k1} & h_{k2} & \ldots & h_{kN} \end{bmatrix} \tag{8.4}$$

Here, a value $w_{ij} \in W$ (with $0 \le w_{ij} \le 1$) indicates the degree of membership of the node n_i in G to the community G_j. Each column of W matrix corresponds to one community with varying membership values of the nodes in G.

Applying an acceptable threshold (say t) on the degree of membership of various nodes in G, crisp communities (subgraphs) with binary membership values w_{ij}^b can be determined.

$$w_{ij}^b = \begin{cases} 1, & \text{if } w_{ij} \ge t, \\ 0, & \text{otherwise.} \end{cases} \tag{8.5}$$

Identifying crisp communities is useful in applications requiring to divide the nodes of the graph into disjoint groups.

8.4 Mining Anomalous Sub-graphs via Community Detection

Referring to the earlier discussion, it is understood that the power law relationship-based approach is applicable to identify anomalous subgraphs that are egonets structurally. However, there may be some anomalous subgraphs which are arbitrary subgraphs of a given graph. For example, an arbitrary subgraph shown in Fig. 8.2b qualifies to be an anomalous subgraph (near clique) that is not an egonet structurally. Therefore, a generic approach to identify all possible anomalous subgraphs is essential. In this direction, the methodology presented here is intended to detect *near clique* and *near tree* types of anomalies. In contrast to a near star anomaly, a near tree anomaly represents a rooted hierarchical structure, which is not an egonet.

In order to detect near tree type of anomalies, the set of connected subgraphs is required. As listing out all possible connected subgraphs of a large graph leads to combinatorial explosion, the same is achieved through community detection as explained in the previous section. Then, each one of these subgraphs can be subjected to the power law relationship based anomaly detection mechanism.

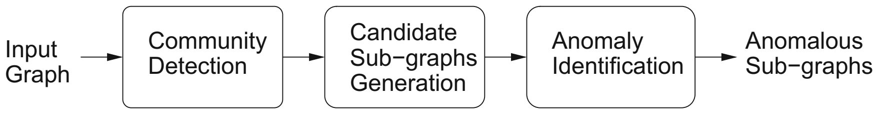


Fig. 8.4 Mining anomalous subgraphs through community detection

The process flow involved with this approach for mining anomalous subgraphs through community detection is shown in Fig. 8.4. There are three building blocks in this method as described below.

1. *Find out the communities/subgraphs using the NMF procedure*:
   Communities present in network/graph data can be detected by performing the NMF procedure on the adjacency matrix G, employing a user defined value for the number of communities (k). This is followed by membership thresholding to produce crisp communities. The resulting subgraphs are basically induced subgraphs corresponding to the nodes contained by the crisp communities.
2. *Apply the subgraph constraints to produce candidate set of subgraphs*:
   Let N_i be the number of nodes and E_i be the number of edges present in a community/subgraph G_i. The following two constraints are defined to prune the set of crisp subgraphs thereby producing the candidate set of l $(0 \leq l \leq k)$ subgraphs/communities.

   - *Sub-graph size constraint*: The minimum number of nodes required in a subgraph for it to be considered as non-trivial is set as '3', i.e., $N_i \geq 3$.
   - *Sub-graph connectivity constraint*: The minimum number of edges needed in a subgraph with N_i nodes for it to be connected is given by $E_i \geq (N_i - 1)$.

   The first constraint is a filtering step on the number of nodes in a subgraph indicating the necessary condition for considering the same in further processing. Notwithstanding the minimum value suggested above, one can choose a higher value for this purpose depending on the requirement. Similarly, the second constraint ensures that the subgraph under consideration is a connected one (a *tree* at least). In case, if a candidate subgraph contains any unconnected nodes (pendant vertices), the same can be deleted as long as the size constraint is satisfied post deletion.
3. *Determine the ranked list of anomalous subgraphs based on SDPL relationship*:
   Anomalous characteristics of subgraphs can be assessed by resorting to the power law relationship. In the context of mining anomalous arbitrary subgraphs, this relationship is named as Sub-graph Density Power Law (SDPL). By imposing the SDPL relationship on the candidate subgraphs, a scatter plot of E_i versus N_i on the *log-log* scale can be produced. Each one of these subgraphs becomes a 2-dimensional point in this scatter plot, and the least squares line fitting can be

done over these points. From this plot, the anomaly score of a subgraph G_i can be computed by measuring its distance to the fitting line as given below.

$$out\_score(G_i) = \frac{max(E_i, CN_i^{\alpha})}{min(E_i, CN_i^{\alpha})} log(|E_i - CN_i^{\alpha}| + 1) \tag{8.6}$$

A ranked sequence of anomalous subgraphs can be produced by arranging them in decreasing order of their score values.

8.4.1 ADG-NMF Algorithm

Given a graph with N nodes and E edges represented by its adjacency matrix G, the objective is to bring out a ranked list of anomalous subgraphs present in this graph using the methodology shown in Fig. 8.4. The corresponding algorithm, named as Anomaly Detection in Graphs using NMF technique (ADG-NMF), comprises the following computational steps.

1. Factorize the adjacency matrix using NMF procedure as $G \approx WH$.
2. Determine k communities $\{G_1, G_2, ..., G_k\}$ using W matrix.
3. Apply the membership threshold t on W to generate crisp communities.
4. Apply the subgraph constraints to determine the candidate set.
5. Produce E_i versus N_i scatter plot along with least squares fitting line.
6. Measure the deviation of each candidate subgraph G_i using Eq. 8.6.
7. Obtain a ranked sequence of the subgraphs as per their outlier scores.

The ADG-NMF algorithm attempts to analyze graph/network data towards detecting anomalies that are arbitrary subgraphs of the input graph G. With the application of two subgraph constraints presented in the previous section, the resultant communities are expected to be of reasonable size and connectivity.

8.5 Experimentation on Benchmark Data

An experimental study is carried out using ADG-NMF algorithm over benchmark graph data sets taken from the SNAP repository. As the basic task is to detect the communities determined by the connectivity structure, the graphs employed in this experimentation are undirected and without any weights on the edges. Details of these data sets are furnished in Table 8.1. These graph data sets are suitably prepared for anomaly detection.

The graph of routers comprising the Internet can be organized into subgraphs called Autonomous Systems (AS). Each AS exchanges traffic flows with some neighbors (peers). It is possible to construct a communication network of who-talks-to-whom from the BGP (Border Gateway Protocol) logs. This data set is collected from

Table 8.1 Experimental setting for anomaly detection in graph data sets

Data set name	# Nodes (N)	# Edges (E)	Parameter values		Observations	
			# Comm. (k)	Membership threshold (t)	# Candidate subgraphs (l)	Highest score
Facebook-107	1,034	53,498	100	0.01	100	4.1549
AS-2000	6,474	13,895	500	0.001	50	2.6012

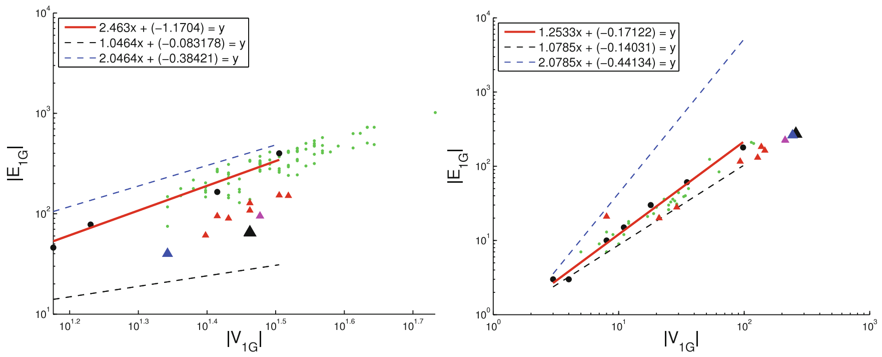
University of Oregon route views project. The experiments in this chapter are carried out on 'as20000102' graph from the Autonomous Systems collection, denoted as 'AS-2000' data set hereafter. This graph data consists of 1,323 self loops and the same are removed as part of data preparation, resulting in only 12,572 edges in the processed data.

Similarly, Facebook data set consists of *social circles* (or friends lists) of various users of this social network. The social network graph of the node with id '107' is considered in this investigation, as it is the largest graph in terms of the number of nodes among the graphs provided. It is referred to as 'Facebook-107' data set in this experimentation. No pre-processing is required on this graph, as it does not contain any self loops or parallel edges.

8.5.1 Anomaly Detection Results

To start with, 'Facebook-107' graph is subjected to ADG-NMF algorithm for subgraph anomaly detection resulting in the scatter plot as shown in Fig. 8.5a. This plot represents the anomalous subgraphs in the form of triangle points, with the area of each triangle point representing the amount of deviation of the corresponding subgraph, as measured using Eq. 8.6. Triangle points below the fitting curve (solid line) indicate near tree anomalies, while the ones above the fitting curve indicate near clique anomalies. This is because of the high edge density of the subgraphs corresponding to the triangle points above the reference line. Similar investigation is carried out on AS-2000 data set producing the results as shown in Fig. 8.5b.

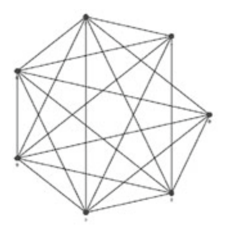
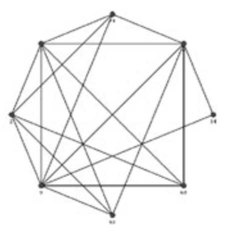
The ADG-NMF algorithm involves two parameters in its computation, namely the number of communities/subgraphs (k) and the threshold on the degree of node membership (t). Accordingly, Table 8.1 provides the values assigned to the parameters in this experimentation. It also indicates the number of candidate subgraphs (l) surviving after applying the subgraph size and connectivity constraints on the crisp communities. Also shown is the highest anomaly score obtained corresponding to each graph data set.



(a) Facebook-107 graph (b) Autonomous Systems graph

Fig. 8.5 Scattor plots indicating anomalous subgraphs in graph data

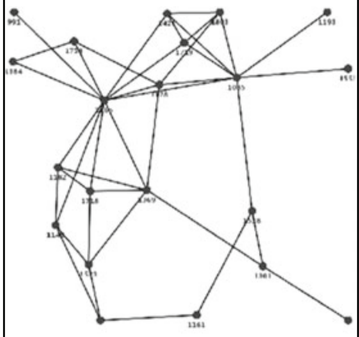
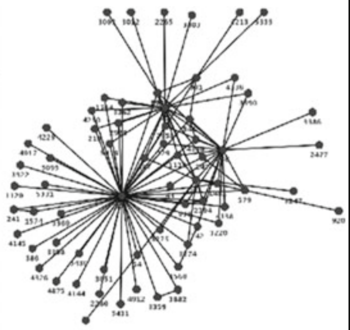
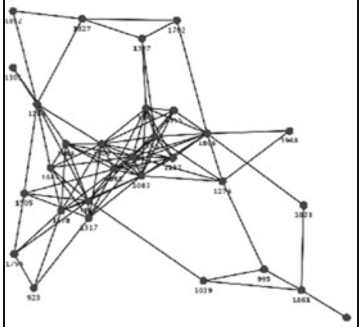
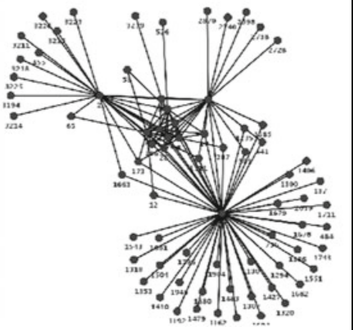
Table 8.2 Egonet type anomalous subgraphs detected in AS-2000 graph

	
(330): nodes=7, edges=20	(439): nodes=8, edges=19

Based on the discussion above, some of the near clique variety of anomalous subgraphs detected in AS-2000 data set are shown in Table 8.2. Referring to the figures embedded in this table, one can understand that these subgraphs are of egonet type in their connectivity structure, resembling the near clique pattern.

Apart from this, some of the near tree variety of subgraphs found with both the data sets are furnished in Table 8.3. The number shown within parenthesis corresponding to each subgraph represents its identifier as per the community detection output. These near tree anomalies are arbitrary subgraphs of the respective input graphs that the ADG-NMF algorithm could find, in addition to the ones having egonet type structure.

Table 8.3 Arbitrary anomalous subgraphs found with benchmark data

Facebook-107 graph	**Autonomous Systems graph**
 (10): nodes=22, edges=40	 (14): nodes=93, edges=116
 (78): nodes=30, edges=95	 (237): nodes=128, edges=131

8.6 Recent Applications

There are numerous real life applications built based on the anomaly/outlier detection principles. Many of them leverage the methodologies developed for data represented using graphs as most of the real world data are inherently linked. Connectivity analysis based techniques offer modularity, thereby leading to the development of scalable algorithms capable of dealing with large data, which is a requirement in most of the application contexts.

8.6.1 Detecting Suspicious Following Behavior in SNs

In a Social Network (SN) of type who-follows-whom, like Twitter, the number of followers indicate the significance associated with a node/individual, leading to potential benefits. As a result, every node tends to attract more followers to boost its in-degree. In the business world, some companies abuse this social phenomenon by creating fake accounts that act as *zombie followers*, i.e. accounts who mindlessly follow others, resulting in a distorted social scenario.

Addressing this problem, a recent method detects the behavior that the zombie followers must have to achieve their business advantage. According to this method, the behavior of the zombie followers exhibits the following properties:

- *Synchronized*: They often follow the very same accounts and
- *Abnormal*: Their behavior pattern is very different from the majority of follower nodes.

Given a directed graph of N nodes in the node set U, find a set of zombie followers U_{zom}. $\mathbf{p}(u)$ represents the feature vector of the node u. Similarly, $F(u)$ denotes the set of node u's followees and $d(u) = |F(u)|$ represents u's out-degree.

- The synchronicity of node u is defined as the average similarity of the feature vectors between each pair of u's followees (v, v'):

$$sync(u) = \frac{\sum_{(v,v') \in F(u) \times F(u)} \mathbf{p}(v).\mathbf{p}(v')}{d(u) \times d(u)}$$

- The normality of node u is defined as the average similarity of the feature vectors between each pair of u's followees and other nodes (v, v'):

$$norm(u) = \frac{\sum_{(v,v') \in F(u) \times U} \mathbf{p}(v).\mathbf{p}(v')}{d(u) \times N}$$

As per the above formalism, the outlier nodes (zombie followers) are those nodes with largest synchronicity scores and smallest normality scores. One can test this method using any large real world who-follows-whom network data, such as Twitter.

8.6.2 Spotting Misbehavior in LBSNs

Location Based Social Networks (LBSNs) are a new class of social networks that allow users to voluntarily share their location, through *check-ins*. The location information can enable a number of novel services to the users. Some service providers of LBSNs offer monetary incentives for users to adopt their usage. As a result, people are tempted to exploit the underlying system and generate false information with fake check-ins.

Addressing the problem of identifying such anomalies in LBSNs, a methodology works based on tensor decomposition technique. It is known that an n-mode tensor is a generalization of a matrix (2-mode tensor) in n dimensions. According to this method, the spatio-temporal information associated with the problem is modeled as a 3-mode (user, venue, time) tensor $\underline{\mathbf{T}}$. Hence, $\underline{\mathbf{T}}(i, j, k) = 1$ iff user i was at venue j at time k. Otherwise, $\underline{\mathbf{T}}(i, j, k) = 0$. Discretizing time in bins of one day, this value can indicate the number of check-ins user i made at venue j on day k.

Employing the Canonical Polyadic (CP) decomposition on the tensor produces a sum of F components as

$$\underline{\mathbf{T}} \approx \sum_{f=1}^{F} \mathbf{a}_f \circ \mathbf{b}_f \circ \mathbf{c}_f \tag{8.7}$$

where $\mathbf{a}_f \circ \mathbf{b}_f \circ \mathbf{c}_f(i, j, k) = \mathbf{a}_f(i)\mathbf{b}_f(j)\mathbf{c}_f(k)$. This means, each component of the decomposition is a rank one tensor corresponding to one of the three modes of the tensor $\underline{\mathbf{T}}$: **a** corresponds to the users, **b** corresponds to the venues, and **c** corresponds to the days.

As a result of this decomposition, one can expect near bipartite cores (in three modes) of people who check-in at certain places for a certain period of time. Due to the time dimension included in this modeling, it is possible to detect certain anomalous check-in patterns such as:

- The temporal evolution of a group of users and venue being almost perfectly periodic, resembling the behavior of a bot.
- A user that has checked-in to a (small) number of venues many times, with an intention of gaining more incentives.

In summary, the above described method is generic in the sense that it does not target any specific misbehavior. It attempts to detect interesting and potentially misbehaving patterns in LBSNs by formulating this problem as a tensor decomposition task.

8.6.3 Anomaly Detection in Attributed Graphs

Graphs in which vertices are associated with attributes and events are known as attributed graphs. For example, in a social network, the vertices indicating various users could be annotated by the level of income he/she is earning. A probabilistic method named as '*g*Anomaly' models generative processes of vertex attributes and divides the graph into regions that are governed by background and anomaly process. According to this method, an anomaly is a connected subgraph whose distribution of attributes significantly differs from the rest of the graph. This method attempts to find connected substructures exhibiting aberrant distribution of attributes. It basically employs a two-component mixture model on an undirected vertex attributed graph $G = (V, E, A)$. Here, V is the set of vertices, E is the set of edges and A is a function that maps a vertex to an attribute value. For ease of understanding, it is assumed that there is only one attribute taking binary values $A = \{1(black), 0(white)\}$. Anomaly detection is carried out by assigning each vertex to one of the mixture components.

Let $V^{(0)}$ be the set of majority (background) vertices, and $V^{(1)}$ the set of anomaly vertices. $V = V^{(0)} \bigcup V^{(1)}$, and $V^{(0)} \bigcap V^{(1)} = \emptyset$. Given a vertex v_i, it belongs to class $V^{(k)}, k = \{0, 1\}$ with probability $\theta_i^{(k)}$. Let P be a mixture model interpreting the overall distribution for a vertex, then

$$P(v_i) = \sum_{k=0}^{1} \theta_i^{(k)} P^{(k)}(v_i) \tag{8.8}$$

where $\{\theta_i^{(0)}, \theta_i^{(1)}\}$ is the vertex-dependent mixture weights and $\theta_i^{(0)} + \theta_i^{(1)} = 1$. $P^{(0)}$ is the background model, and $P^{(1)}$ is the anomaly model.

To determine the best component model to describe each vertex, the model is fitted with the observed data and the total likelihood is computed. Simply maximizing the likelihood over-looks the network structure. Hence, a network regularization method is employed to smoothen the connectivity in each mixture component. Depending on the application and the information propagation mechanism, different variations of the network regularizer can be considered.

To account for the real-world networks scenarios, the gAnomaly method comprises an iterative procedure for anomaly detection. In each iteration, the two-component mixture model is applied on the current graph, and the vertices are classified into anomaly and background. Only the anomaly-induced subgraph is fed into the next iteration. This process continues until an anomaly with desirable size and attribute distribution is found. It is interesting to note that gAnomaly method is extensible to multiple attribute graphs and also to multiple levels of anomalies. Through experimentation, it is noticed that the performance of gAnomaly depends on the structure and pattern distribution within the network.

8.6.4 Detecting Internet Routing Anomalies

Border Gateway Protocol (BGP) handles exchange of external routing information across the autonomous systems (AS) on the Internet. A network prefix represents a set of consecutive IP addresses pertaining to a network. The process of originating prefixes of a network by an unauthorized network is known as *prefix hijacking*, which is a serious security concern relating to network service availability as well as data privacy. This kind of an attack could be intentional or unintentional, possible due to non-availability of authentication mechanism in BGP protocol.

Multiple origin AS (MOAS) events occurs when the same prefix appears to belong to at least two different ASes. In contrast to these small scale anomalies, large scale anomalies represent large routing deviation between ASes and links.

'BGPGraph' approach detects these routing anomalies by employing information theory and a hierarchical graph visualization scheme. This approach works based on AS graph built from AS connectivity structure. It maps the node/edge weights to visualization space and displays the AS graph in a hierarchy of coarse to fine details. Entropy measure is considered to compute the information content of the graph that is currently visualized. It basically defines a global anomaly metric based on entropy measurement as given in Eq. 8.9 for estimating the anomalous behavior of the network at different points in time, thereby identifying the deviating points for further examination.

$$A = w_e * A_e + w_m * A_m \tag{8.9}$$

Here, A_e is a metric to determine large scale anomalies, while A_m captures small scale anomalies. w_e, w_m are the weights used to establish the combined impact of all possible anomalies.

Using the above metric, bar plots indicating the anomaly score of the BGP activity can be generated over a specified period of time. On determining the significant anomalous regions using this plot, the analyst can examine the network visualization to determine exact details leading to anomalous behavior.

The analytical potential of this approach can be tested on BGP routing data connected with RIPE project. This data comprises a list of 12 known anomalous BGP events such as the AS-9121 incident that occurred due to BGP router misconfiguration on 24th December 2004. This method can point out some unknown anomalies also due to its embedded ability to detect small as well as large scale routing events. Through visualization of the BGP graph, it is possible to identify the ASes responsible for the detected anomalies.

8.6.5 Anti-sparse Representation for Detecting DoS Attacks

A new outlier detection method based on anti-sparse representation (also known as l_∞-norm minimization), which is an useful technique for vector quantization and control engineering applications. It deals with the non-smooth objective function by employing a new smoothing approach with descent methods. Anti-sparse representation transforms the input signal to spread evenly over the representation elements. Due to this favorable property, it is considered as a natural scheme for binarization. Binarization embeds anti-sparse input vector to Hamming space binary vector to have computational gain. Embedding maps query vectors to the target vector in order to perform fast and efficient comparison.

A binarization scheme with anti-sparse signal can be created as: $b(\mathbf{k}) = sign(\mathbf{x})$, where $\mathbf{x}$ is the anti-sparse representation of relevant input vector $\mathbf{k}$. A promising use of anti-sparse representation is for applications based on approximate nearest neighbor (ANN) search.

Distance-based outlier detection is known to be a computationally efficient approach that determines an outlier depending on the distance to its nearest neighbors. The idea here is to identify approximate nearest neighbor for a query vector (**q**) and determine whether it is an outlier or not. Using anti-sparse based ANN search, the similarity between query and target can be computed as maximum inner product search (MIPS) between two binary vectors, as defined below.

$$NN(b(\mathbf{q})) = arg\ max_{\mathbf{y} \in R^m} b(\mathbf{q})^T b(\mathbf{y}). \tag{8.10}$$

Input vectors satisfying $NN(b(\mathbf{q})) < s$ are considered as outliers, where s is the similarity threshold. It is important to note that similarity between input vectors is computed here, instead of distance measure.

Denial-of-Service (DoS) attacks are simple and powerful attempts to make a network resource inaccessible. As the behavior of such an attack doesn't fit the expected nature of the network, DoS attacks can be considered as outliers/anomalies. Therefore, anti-sparse representation based outlier detection approach described above is applicable here for detecting DoS attacks on a network. Considering m and n to represent the dimension of the input vector (number of features) and the length of binarized anti sparse signal respectively, the detection performance can be measured against the under determinacy value ($\rho = \frac{m}{n}$) of the representation.

One can test this method using KDDCup-99 data set for detecting various known types of DoS attacks such as 'back', 'land', 'pod', 'teardrop', etc. An experiment using a subset of the data containing 6000 normal and 1000 outliers (DoS attacks) resulted in detection accuracy of above 98% with $n = 4m$ (i.e. $\rho = \frac{1}{4}$).

8.6.6 Network Anomaly Detection by IP Flow Graph Analysis

Detecting network anomalies at the earliest and resolving them quickly is one of the main challenges to the network administrator. This challenge is due to increased complexity of network management with the network growth. More specifically, anomalies generated by DoS attacks is expected to increase as a result of increasing cyber threats to IP networks.

An automated network anomaly detection method models the network flows using graph representation. Analyzing flow graph is useful to identify potential misuse of network resources, bottlenecks and policy violations. Detecting anomalies of this kind based on network traffic features looks for variations in the number of source/destination IP addresses and source/destination ports that a network device uses. These variations can be measured using Tsallis entropy for detecting anomalous network flows.

According to this method, all the network flows collected in the interval t is represented by a directed graph $G = (D, F)$. Here, D represents the network devices/IP-addresses, while the set F depicts the flows sent and received by devices. An edge (d_i, d_j) represents a set of flows grouped by the source address and the destination address. The total number of devices (vertices of the flow graph) is given by

$n_D = |D|$. Then, the probabilities of the four selected flow properties of the grouped flows in each edge are computed.

Given the probabilities $p_1, p_2, ..., p_n$, with $0 \leq p_i \leq 1$ the Tsallis or non-extensive entropy is defined as following.

$$H_q(p_1, p_2, ..., p_n) = \frac{1}{q-1}(1 - \sum_{i=1}^{N} p_i^q) \tag{8.11}$$

In the above equation, q is the entropic parameter. When $q > 1$, higher probabilities contribute more than the lower one to entropy result. The opposite situation occurs when $q < 1$. For anomaly detection, the parameter q allows the network administrator to configure the sensitivity of the anomaly detector.

Due to the large number of source ports involved in a typical DDoS attack, the entropy of this flow property turns out to be high compared to the other ones. Through flow graph analysis, it is possible to depict the data such as number of bytes, packets, and flows sent and received by each device/IP address, in order to identify misuse of network resources and bottlenecks.

Through experimental study on real life network flow data collected using NFDUMP tool (with NetFlow v9 interface), it is found that larger values of entropic parameter (like $q \geq 1.5$) do not give much gain for anomaly detection. On the other hand, when q is less than zero the sensitivity of the detector increases significantly. This method can also help to rank and present the most network resources requested by various network entities for necessary planning to ensure continued services to the users with quality.

8.7 Summary

The problem of detecting anomalous entities in network/graph data as a potential approach for effective analysis is addressed in this chapter. By applying outlier detection principles in graph data mining, it is possible to characterize the deviating nature of nodes/edges/subgraphs present in a graph. The salient aspects of the material presented in this chapter are as follows.

- Various data mining techniques on network/graph data such as discovering frequent subgraphs, identifying dense components, establishing graph pattern match and community detection are discussed.
- Graph analysis techniques based on ego-network notion and community detection are presented in detail.
- The Non-negative Matrix Factorization (NMF) technique as a way of determining the decomposition of a data matrix is presented. Suitability of this technique for discovering communities in a social graph is also looked at.
- A recent method, named as ADG-NMF, for mining anomalous subgraphs in graph data through community detection using the NMF technique is discussed. This

method attempts to find subgraph anomalies by subjecting the detected communities to the EDPL power law constraint and also measures their anomaly scores.
- ADG-NMF method can detect any arbitrary subgraph with anomalous characteristics, covering both near clique and near tree types of patterns present in the input graph.
- Experimental results over benchmark graph data sets indicate that there exist various types of subgraph anomalies in real life social graphs and their detection is important for effective graph analysis for deriving the required semantics inherent to the data.
- Highlighting this aspect, some of the methods developed in the recent past by employing anomaly-based analysis of network/graph data addressing various real life computing requirements are also presented in this chapter.

8.8 Bibliographic Notes

Graph mining is one of the most explored and continuously evolving research areas in the domain of data mining [1, 5]. Graph mining is popular across a slew of applications in real world [13, 20, 33].

Discovering frequent subgraphs and dense subgraphs are two important data mining tasks on network/graph data. A comparative survey of algorithms for frequent subgraph discovery was brought out in [15], by proposing a taxonomy of these algorithms based on three essential factors that connect the algorithms and their solutions to the application requirements. Similarly, one can find dense components in real-world graphs as they are often sparse globally and dense locally [28]. Dense components can be seen in several real world networks such as social networks, world wide web, biological systems, etc. [7, 17]. Accordingly, one may come across different formulations of the dense subgraph detection problem. It is known that clique, quasi-clique and k-densest subgraph detection formulations are NP-hard [28]. Addressing this issue, an algorithm for finding dense subgraphs using spectral graph theory principles was proposed recently [31]. This algorithm first determines primitive dense elements and then merges them iteratively to form higher-level dense components resulting in cliques/near-cliques.

Graph pattern matching and community detection are the other important graph mining tasks. An approach for multi-labeled graph matching was presented in [14] for finding the matches of a labeled query graph. Communities in a network/graph represent groups of nodes with close connectivity within them. A comprehensive exposition of the process of community detection in social networks can be found in [8].

In addition to above graph analysis tasks, anomaly detection in network/graph data is an emerging process with numerous applications. Anomalies in a graph can be found at different granularity such as anomalous nodes/edges/subgraphs. There are some research efforts made in the recent past for detecting various types of anomalies in graph data. The method proposed in [23] is an early one using the minimum description length (MDL) principle. The main idea of this method was that subgraphs containing many common substructures are generally less anomalous than subgraphs with a few common substructures. Thus, it is qualified mainly for applications involving many common substructures such as the graphs describing the atomic structure of various chemical compounds. An algorithm for finding node groups and anomalous edges based on information theoretic principles was proposed [4]. Similarly, the method proposed in [26] is meant for anomalous link (edge) discovery defining a novel edge significance measure.

A recent work on anomaly detection in graph data [2], makes use of the notion of ego-network (*egonet*) of a node (*ego*) for graph analysis. This method, named as 'OddBall', presents the Egonet Density Power Law (EDPL) relationship satisfied by various ego-networks of a social graph. This relationship basically captures the edge density of the egonets. Any variation from the expectation indicates two possibilities: the edge density is either too high or very low. Then, it determines the amount of deviation corresponding to each egonet as its distance to the least squares fitting line. Thus, this method could detect subgraph anomalies of various types such as near-clique, near-star, heavy vicinity, dominant edge, etc.

The Non-negative Matrix Factorization (NMF) technique [18] is a promising approach for finding non-negative factors of a data matrix. It was employed in several social network applications in different contexts [30], due to its powerful interpretability and close relationship with data clustering methods. In this direction, the method proposed in [35] is a recent one employing NMF technique for detecting communities in graph/network data. In subsequent efforts, a similar approach was devised to identify various connected subgraphs of an input graph by applying an acceptable threshold on the membership values produced by the NMF-based decomposition [32]. The crisp communities thus identified were subject to the power law relationship (described above) with the view of determining their deviating characteristics. Thus, this method (named as ADG-NMF) attempted to discover anomalous subgraphs in network/graph data.

Experimental observations reported in this chapter are obtained using benchmark graph data sets taken from the SNAP repository [19]. Among the network/graph data sets considered here, Facebook data is a popular one with published results on discovering social circles [22].

Many recent applications can be found based on the idea of anomaly detection in graph/network data. A method was proposed recently [12] to study and detect the behavior that the zombie followers must have to achieve their business advantage. A novel methodology was proposed recently [25] employing tensor decomposition technique to detect anomalies in LBSNs. As part of this method, Canonical Polyadic (CP) decomposition [10] was applied on the tensor to produce a sum of

F components. A probabilistic method named as gAnomaly [21], models generative processes of vertex attributes in order to identifying graph-based anomalies. 'BGPGraph' [24] approach was proposed for detecting various routing anomalies employing information theory and a hierarchical graph visualization scheme. This approach was demonstrated on BGP routing data connected with RIPE project [27], using a list of 12 known anomalous BGP events [16]. A new outlier detection method based on anti-sparse representation was proposed in [34], which is useful for vector quantization and control engineering applications. Similarly, an automated network anomaly detection method was proposed [3] by modeling the network flows using graph representation.

Referring to the recent works on detecting anomalies in graph data, an unsupervised algorithm was proposed based on graph kernel concept [36]. Similarly, one more approach is the query-based evolutionary graph cuboid outlier detection [6]. In other such effort, a local learning based approach was developed for mining outlier subgraphs from network data sets [9]. From the fraud detection point of view, a graph-based detection technique in the face of camouflage was present recently [11]. Similarly, a study of various interesting graph patterns and anomalies present in real world graphs along with their applications was furnished in a very recent publication [29].

References

1. Aggarwal, C.C., Wang, H. (eds.): Managing and Mining Graph Data. Advances in Database Systems, vol. 40. Springer (2010)
2. Akoglu, L., McGlohon, M., Faloutsos, C.: Oddball: Spotting anomalies in weighted graphs. In: 14th Pacific-Asia Conference on Advances in Knowledge Discovery and Data Mining (PAKDD), pp. 410–421. Hyderabad, India (2010)
3. Amaral, A.A., Mendes, L.S., Pena, E.H.M., Zarpelão, B.B., Jr., Proença, M.L.: Network anomaly detection by ip flow graph analysis: a ddos attack case study. In: 32nd International Conference of the Chilean Computer Science Society. SCCC, pp. 90–94. IEEE Computer Society, Temuco, Cautin, Chile (2013)
4. Chakrabarti, D.: Autopart: Parameter-free graph partitioning and outlier detection. In: PKDD, pp. 112–124 (2004)
5. Cook, D.J., Holder, L.B.: Mining Graph Data. Wiley, USA (2006)
6. Dalmia, A., Gupta, M., Varma, V.: Query-based evolutionary graph cuboid outlier detection. In: 16th International Conference on Data Mining Workshops (ICDMW), pp. 85–92. IEEE Computer Society, Barcelona, Spain (2016)
7. Dourisboure, Y., Geraci, F., Pellegrini, M.: Extraction and classification of dense communities in the web. In: WWW, pp. 461–470. Banff, Alberta, Canada (2007)
8. Girvan, M., Newman, M.E.J.: Community structure in social and biological networks. PNAS **99**(12), 7821–7826 (2002)
9. Gupta, M., Mallya, A., Roy, S., Cho, J.H.D., Han, J.: Local learning for mining outlier subgraphs from network datasets. In: SDM (2014)
10. Harshman, R.A.: Foundations of the parafac procedure: models and conditions for an explanatory multimodal factor analysis (1970)
11. Hooi, B., Shin, K., Song, H.A., Beutel, A., Shah, N.: Graph-based fraud detection in the face of camouflage. ACM Trans. Knowl. Discov. Data **11**(4), Article ID 44, 1–26 (2017)

12. Jiang, M., Cui, P., Beutel, A., Faloutsos, C., Yang, S.: Detecting suspicious following behavior in multimillion-node social networks. In: WWW (Companion). ACM (2014)
13. Koutra, D., Faloutsos, C.: Individual and Collective Graph Mining: Principles, Algorithms, and Applications. Synthesis Lectures on Data Mining and Knowledge Discovery. Morgan & Claypool Publishers (2017)
14. Krishna, V., Suri, N.N.R.R., Athithan, G.: Mugram: a multi-labelled graph matching. In: International Conference on Recent Advances in Computing and Software Systems, pp. 19–26. IEEE Xplore, Chennai, India (2012)
15. Krishna, V., Suri, N.N.R.R., Athithan, G.: A comparative survey of algorithms for frequent subgraph discovery. Curr. Sci. **100**(2), 190–198 (2011)
16. Lad, M., Massey, D., Zhang, L.: Visualizing internet routing changes. IEEE Trans. Vis. Comput. Grapics **12**(6), 1450–1460 (2006)
17. Lee, V.E., Ruan, N., Jin, R., Aggarwal, C.: A survey of algorithms for dense subgraph discovery. In: Managing and Mining Graph Data, pp. 303–336. Springer (2010)
18. Lee, D.D., Seung, H.S.: Learning the parts of objects by non-negative matrix factorization. Nature (1999)
19. Leskovec, J., Krevl, A.: SNAP Datasets: Stanford large network dataset collection (2014). http://snap.stanford.edu/data
20. Leskovec, J.: Large-scale graph representation learning. In: IEEE International Conference on Big Data, p. 4. Boston, MA, USA (2017)
21. Li, N., Sun, H., Chipman, K., George, J., Yan, X.: A probablistic approach to uncovering attributed graph anomalies. In: SDM (2014)
22. McAuley, J., Leskovec, J.: Learning to discover social circles in ego networks. In: Advances in Neural Information Processing Systems (NIPS), pp. 548–556. Nevada, USA (2012)
23. Noble, C.C., Cook, D.J.: Graph-based anomaly detection. In: SIGKDD, pp. 631–636. Washington, DC, USA (2003)
24. Papadopoulos, S., Moustakas, K., Drosou, A., Tzovaras, D.: Border gateway protocol graph: detecting and visualizing internet routing anomalies. IET Inf. Secur. **10**(3), 125–133 (2016)
25. Papalexakis, E., Pelechrinis, K., Faloutsos, C.: Spotting misbehaviors in location-based social networks using tensors. In: WWW(Companion). ACM (2014)
26. Rattigan, M.J., Jensen, D.: The case for anomalous link discovery. SIGKDD Explor. **7**(2), 41–47 (2006)
27. Routing information service project (RIS). https://www.ripe.net
28. Sariyuce, A.E., Seshadhri, C., Pinar, A., Catalyurek, U.V.: Finding the heirarchy of dense subgraphs using nucleus decompositions. In: WWW, pp. 927–937. ACM, Florence, Italy (2015)
29. Shin, K., Eliassi-Rad, T., Faloutsos, C.: Patterns and anomalies in k-cores of real-world graphs with applications. Knowl. Inf. Syst. **54**(3), 677–710 (2018)
30. Snasel, V., Horak, Z., Kocibova, J., Abraham, A.: Reducing social network dimensions using matrix factorization methods. In: Advances in Social Network Analysis and Mining, pp. 348–351. Athens, Greece (2009)
31. Suri, N.N.R.R., Krishna, V., Kumar, K.R.P., Rakshit, S.: Detecting hotspots in network data based on spectral graph theory. In: Second International Conference on Research in Computational Intelligence and Communication Networks (ICRCICN), pp. 45–50. IEEE Xplore (2016)
32. Suri, N.N.R.R., Murty, M.N., Athithan, G.: Mining anomalous sub-graphs in graph data using non-negative matrix factorization. In: P. Maji (ed.) 5th International Conference on Pattern Recognition and Machine Intelligence (PReMI), LNCS, vol. 8251, pp. 88–93. Springer, Berlin, Heidelberg (2013)
33. Swarnkar, T., Mitra, P.: Graph based unsupervised feature selection for microarray data. In: IEEE International Conference on Bioinformatics and Biomedicine (BIBM) Workshops, pp. 750–751. Philadelphia, USA (2012)

34. Vural, M., Jung, P., Stanczak, S.: A new outlier detection method based on anti-sparse representations. In: 25th Signal Processing and Communications Applications Conference. SIU, pp. 1–4. IEEE, Antalya, Turkey (2017)
35. Wang, F., Li, T., Wang, X., Zhu, S., Ding, C.: Community discovery using nonnegative matrix factorization. Data Min. Knowl. Discov. **22**(3), 493–521 (2011)
36. Zhang, L., Wang, H., Li, C., Shao, Y., Ye, Q.: Unsupervised anomaly detection algorithm of graph data based on graph kernel. In: 4th International Conference on Cyber Security and Cloud Computing. CSCloud, pp. 58–63. IEEE, New York, NY, USA (2017)

Chapter 9
Analyzing Dynamic Networks

Abstract This chapter highlights the importance of analyzing dynamic networks and gives details of various applications requiring this capability. It also furnishes a few recent algorithmic approaches for analyzing such networks in a pragmatic manner.

9.1 Introduction

Many real world networks evolve over time indicating their dynamic nature to cope up with the changing real life scenarios. As network evolution happens distinctly in different application domains, exploring varying semantics of the evolution process forms the crux of dynamic network analysis. Evolution in graphs/networks happens by way of adding new edges and/or deleting old ones over time. Various methodologies exist for understanding the application-specific insights of evolution in graphs/networks. As stated earlier, the discussion here makes use of these terms 'network' and 'graph' as synonyms.

According to literature, analysis of evolving networks can be broadly divided into two major categories.

- *Maintenance methods*: These methods maintain/update the results of the data analysis process continuously and incrementally over time. Otherwise, the results will become stale over time as the network evolves.
- *Evolution analysis methods*: These methods are meant to model the change, rather than maintaining the results, by directly quantifying and understanding the changes occurring in the network.

One more categorization of these methods can be thought of based on how fast the network evolution takes place.

- *Slowly evolving networks*: These are the networks that evolve slowly over time. Snapshot analysis can be very effective for such networks with the possibility of performing offline analysis directly. Given network snapshots at times t_1 and t_2, the

N. N. R Ranga Suri et al., *Outlier Detection: Techniques and Applications*,
Intelligent Systems Reference Library 155,
https://doi.org/10.1007/978-3-030-05127-3_9

results of the data analysis algorithm are adjusted at each snapshot by incrementally adjusting the results from the previous snapshot.
- *Streaming networks*: These are the networks that are created by transient interactions, typically represented as graph streams, requiring real-time analysis capability due to the inability to hold the entire graph on the disk.

From the analysis point of view, detecting the communities present in a network is interesting, especially when they are deemed to be produced by a generative methodology such as data clustering (unsupervised model). This is because the temporal variation of the generative behavior of the network can help in understanding the overall network evolution. Thus, the community detection task can be seen as a *bridge* between the two modes (first categorization) of analysis.

As a matter of fact, certain evolutionary clustering methods can perform clustering and evolution analysis simultaneously. Tightly integrated methods of this kind are useful in understanding the nature of the underlying evolution. In general, most of the methods dealing with popular data mining tasks such as clustering, classification and outlier detection on dynamic networks enable evolution analysis in a manner.

- *Clustering*: One of the earliest methods for evolutionary clustering is an on-line algorithm that performs a trade-off between the cost of maintaining the clusters accurately and the cost of deviating from the historical data. Similarly, yet another method for evolutionary clustering uses a probabilistic mixture model to characterize the clusters in a heterogeneous network. It employs a smoothness criterion to determine soft clusters over successive snapshots. It also explores various properties of clusters, such as consistency and clustering quality, to characterize the nature of the evolution of the clusters.
- *Classification*: Addressing the node classification problem in evolving networks, a random walk-based approach uses the fraction of nodes visited belonging to each class to predict the class label of remaining nodes. A dynamic inverted index is maintained to perform the random walk efficiently.

9.2 Understanding Graph Evolution

The changes taking place in the graph connectivity structures due to evolution can be characterized by measuring some graph properties such as centrality, community behavior, minimum description length (MDL), shortest paths, etc. The evolutionary behavior can be modeled using the methods that rely upon the key laws satisfied by a wide variety of social networks. Most evolution analysis laws are derived analytically from the notion of *preferential attachment*. According to this notion, the likelihood of receiving new edges by a node increases with its degree. If $\pi(k)$ is the probability that a node attaches itself to another node with degree k, then this probability $\pi(k)$ is related to k as $\pi(k) \propto k^{\alpha}$, where α is a parameter whose value is dependent on the underlying domain. For example, $\alpha \approx 1$ for scale-free networks, in which only addition of edges happens.

If $n(t)$ is the number of nodes in an evolving network at time t and $e(t)$ is the number of edges, then the network exhibits a densification power law as $e(t) \propto n(t)^{\alpha}$. Here, α takes a value between 1 and 2. As the network densifies, the average distances between the nodes reduce leading to shrinking diameter of the graph, but bounded from below. Densification also results in the emergence of a giant connected component due to the principle of preferential attachment as explained above.

On the phenomenon of group formation in social networks, it is found that the propensity of individuals to join communities depends on the network structure. This is influenced by two factors: (i) the number of friends one has within the community, and (ii) the way they are connected with one another. Other related studies reveal the effect of both extrinsic and intrinsic factors to group formation in social networks.

9.2.1 Community Evolution

Community-based approaches are the natural choices for carrying out evolution analysis due to the ability of clusters (communities) in summarizing the network structure. Based on the properties observed regarding the evolution of small and large communities, it is understood that large groups typically persist longer if they are capable of dynamically altering their membership. A related study says that the evolution of communities is gradual and individuals do not tend to change their *home community* too quickly in most cases.

One of the challenges in community-based analysis of dynamic networks is establishing a proper match of the communities over various snapshots in time. To address this issue, soft clustering methods are introduced that assign probabilities of membership of each node to different clusters. The basic of idea of this method is to perform clustering on temporal snapshots of the network by maintaining continuity among the clusters so as to compare the clusters found in the next snapshot with their counterparts in the current snapshot. This work suggests that temporal community structure analysis can reveal global and local structural properties of networks. In the matching process, the clusters/communities that do not match the ones in the previous snapshot are named as *evolutionary community outliers*.

9.3 Modeling Dynamic Networks

Networks provide a useful and effective way of leveraging large-scale event data for interacting with agent-based models. Nodes are analogous to agents representing individuals in a network. There are two foundational network data representation methods: *dense representation* using data matrices, and *sparse representation* using edge/link lists. Social networks are typically represented using *edge lists* due to their sparse connectivity attributing them the scale-free network behavior.

Various statistics can be used to characterize the networks at different levels of granularity. At the overall network level, these statistics include density, link count, isolate node count, component count, clustering coefficient, average distance, diameter, etc. At the node level, they include various centrality measures based on degree, betweenness, eigenvector, closeness, etc.

Based on the node statistics and the type of network being analyzed, it is possible to identify the key or influential entities in a network. The behavior of these key entities can be characterized by considering various known patterns such as brokers, bridges, structural holes in networks. In addition, it is possible to match the influential entities to these known behaviors and network features using various centrality measures as given below.

1. *Degree centrality* of a node can be useful for identifying resource-rich individuals in a network.
2. High *betweenness centrality* of a node signifies its ability to connect disparate groups.
3. *Eigenvector centrality* indicates agents with strong social capital due to their connections to high degree central nodes.
4. *Closeness centrality* testifies nodes being able to utilize several varied resources very quickly.

In addition to dealing with simple nodes and edges, modern network analysis often finds it useful to associate descriptive attributes/characterizations to nodes. This in turn can help the analyst in partitioning the nodes and facilitating a visual examination of networks by coloring the nodes based on their attribute values.

Comparing networks is another useful analysis task to assess how likely one network structure is to appear as the result of certain transformation on another network structure. However, standard statistical approaches are not acceptable for comparison of network structures due to the fact that neither the independence nor identical distribution assumptions hold good. Though *Hamming distance* computed on vector representations of networks is useful for understanding the overall difference among them, it doesn't provide any meaningful structural comparison. Therefore, one has to employ a more sophisticated technique for accomplishing this task.

In general, networks are known to be complex objects and hence it is difficult to analyze them in an objective manner. However, understanding the dynamic evolution of networks has matured as an important interdisciplinary research problem with applications in economics, physics and computer science. Therefore, having a model of *network formation* becomes the key enabler for carrying out efficient network analysis. In this regard, one can look at two promising approaches to model the evolution of social and economic networks: (i) random graph theory and (ii) evolutionary game theory.

Economists interpret network structures as equilibrium phenomena of strategically acting players who create and destroy links according to their own incentives. They employ game theoretical concepts to model this phenomena. On the other hand, physicists consider networks as an outgrowth of complex system analysis aimed at understanding and characterizing the statistical regularities of large stochastic networks.

9.3.1 *Statistical Model of Networks*

This approach is typically based on the principles of random graph theory. Random graph models provide a flexible framework to construct statistical ensembles of networks to model them. Erdos-Renyi graph or Bernoulli graph is an important random graph model built on the assumption that edges are formed independently with probability $p \in [0, 1]$. The advantage with this model is its simplicity in describing a complete random graph with two parameters: the size of population (N) and the edge-success probability (p). However, it lacks correlation across links in its network formation process.

Similar to random graphs, real world networks are observed to have small average path lengths (small world phenomena leading to six degrees of separation), higher clustering coefficients, exhibit homophily (links between nodes of similar kind are more likely) and often satisfy a power law degree distribution.

From the practical point of view, a model of dynamic network formation should capture two important aspects: (i) the volatility of network (links are deleted and formed over time), and (ii) the likelihood that a link is formed or destroyed. Link creation results in the network expanding at a constant rate $\lambda \geq 0$ over time. Similarly, link destruction also happens at a constant rate $\xi \geq 0$. In principle, the intensities of link creation and destruction can be asymmetric across various pairs of nodes. This enables in modeling the formation of directed as well as undirected networks encountered in real life.

In a typical network, it may be the case that the vertices can be classified into groups. In this connection, a prevalent fact in social networks is the phenomenon of *homophily*, which means vertices of similar characteristics are more likely to get connected with each other. Based on this phenomenon, *block-models* categorize the vertices of a network into groups with each group having its own law of network formation. Networks of this type are called as *multi-type* random networks in economic theory.

9.3.2 *Game Theoretic Model of Networks*

The statistical model described above is useful to describe *how* networks form and evolve. However, it cannot explain *why* networks form and evolve in real life. To complete the study of network formation, it is required to understand the forces that drive the nodes to connect to each other. Economic and game theoretic reasoning provides the necessary explanation of this phenomena. According to this reasoning, networks are interpreted as equilibrium outcomes implemented by rational players.

As per game theory, two individuals are connected with each other because of a *payoff* or *utility* derived from these connections. This theory seems more realistic as it allows us to rationalize the observed network structure in terms of *socioeconomic behavior*. It also allows the researchers to understand the mechanism underlying the formation of a network. A pure statistical approach cannot provide this kind of ‘micro-foundation’ of how these observed networks are actually formed.

An important equilibrium notion connected with strategic network formation is the concept of *pairwise stability*. It considers the differential impact of an added link on the utility of the involved parties. A network is pairwise stable if (i) no player has an incentive to delete one of her links and (ii) there does not exist a pair of players who want to form a mutually beneficial link. One practical limitation of pairwise stability is that it only considers single link changes in each updating step. The stability concept can be extended to consider the formation of mutually beneficial links as well as the deletion of more than one link in one period.

9.4 Applications of Dynamic Network Analysis

Due to the ubiquitous nature of dynamic networks, one is often posed with the challenge of analyzing such networks in connection with several contemporary applications. Though the exact method of analysis varies depending on the specific application, the general requirement happens to be understanding the inherent semantics of the network. In this regard, some of the recent applications dealing with dynamic network analysis with focus towards anomaly/outlier detection are presented below to highlight their significance in practice.

9.4.1 Ranking Suspicious Entities in Time-Evolving Tensors

An important research problem in the context of web-services and online social networks is to detect the fraudsters that are trying to attack the systems providing these services. Typically, fraudulent activities take the form of *lockstep* or highly synchronized behavior such as multiple users liking the same set of pages on Facebook. Such a behavior results in *dense blocks* in matrices/tensors corresponding to these activities due to reuse of resources necessary for the fraud to take place. Among various consumers of the online services, it is essential to know which of them are more suspicious for initiating a directed action on such individuals from system security point of view.

A recent approach for scoring entities based on their participation in suspicious dense blocks has brought out a set of axioms that any ideal individual scoring metric should satisfy. It provides a framework to make use of temporal information that generally exists in all the real-world data sets in the form of *timestamps* of various events.

As presented in Sect. 8.1, dense subgraph discovery (DSD) is an important graph mining task as dense subgraphs (dense blocks) in real-world data tend to reflect fraudulent lock-step behavior. A revised form of this task is to detect dense-blocks in multi-aspect data for spotting groups synchronized in multiple aspects, such as IPs, review scores and review keywords. In this context, assessing the anomalousness of individuals is complementary to detecting dense blocks, which correspond to group activities.

The data from a social networking website or a web service can be represented as a k-way timed tensor. Such tensors can be processed by converting the numerical time stamp to interpretable categorical features as given below.

- 0th-*order features*: This is generated by bucketizing the time stamp into number of days, hours, minutes, etc., elapsed from the first observation made. The temporal resolution is chosen depending on the typical level of time variation present in the data.
- 1st-*order features*: This is determined by computing the log-bucketized inter-arrival time between two consecutive timestamps of the same user.
- *Temporal folding features*: To detect fraudsters showing periodic behavior, three temporal folding features of the timestamps are considered: (i) day of the week, (ii) hour of the day and (iii) hour of the week.

Having determined the categorical features to represent the data, the block level suspiciousness score of a subblock γ of a tensor χ can be computed using one of the following three computational methods.

1. Arithmetic(g_{ari}): compute the arithmetic average mass of the block.
2. Geometric(g_{geom}): compute the geometric average mass of the block.
3. Density(g_{susp}): compute the KL-divergence (Kullback Leibler) between the distribution of the mass in the subblock with respect to the distribution of the mass in the tensor.

The approach presented here is based on the δ-contribution of each entity towards the block suspiciousness score. The δ-contribution of an entity i in mode $m(i)$ of a block γ is computed as the difference between the suspiciousness score of block γ before and after removing entity i from the block as given below.

$$\delta_\gamma(i) = g(\gamma) - g(\gamma \setminus i) \tag{9.1}$$

To aggregate the δ-contribution over the entire list of blocks B, one can consider two specific metrics: (i) sum of the δ-contributions and (ii) maximum of the δ-contributions. The second metric assigns each individual/entity the maximum block suspiciousness score out of all the blocks it is part of. This metric is found to be the best performing one on real world data sets.

$$f_\chi(i) = max_{\gamma \in B}(\delta_\gamma(i)) \tag{9.2}$$

In brief, the algorithm for determining *suspicious entities* in a time evolving tensor consists of the following steps.

1. Find M top suspicious dense blocks using one of the computational methods.
2. For every entity i that has occurred at least once in any of these blocks, compute the individual suspiciousness score $\delta_\gamma(i)$ by determining its marginal contribution towards that specific block.
3. Determine the overall marginal contribution $f_\chi(i)$ of each entity i by taking the maximum of its marginal contributions across all M blocks.

9.4.2 Interactive Outlier Analysis over Dynamic Data sets

Distance-based methods for outlier detection are the oldest and most explored ones. However, selecting parameter values suited for detecting outliers matching the user intent is not easy. In this regard, an interactive outlier analysis method named as ONION enables the users to choose applicable parameter values so as to detect outliers as per their requirement. More specifically, this method employs distance-based detection which classifies an object as an outlier if the number of objects that are within a circle (sphere) of radius r centered on the object is less than k. Therefore, choosing satisfactory values for the parameters r and k is critical for this method to be effective.

ONION analyzes data sets in advance and constructs index structures assuming its parameters to be within certain ranges $k \in [k_{min}, k_{max}]$ and $r \in [r_{min}, r_{max}]$. It helps users to choose specific values of the parameters within their dynamic ranges. Let us denote the distance between an object O and its k-th nearest neighbor by D_O^k. Then, the object O gets classified into one of the following categories.

1. *Always Inlier (IL)*: If $D_O^{k_{max}} \leq r_{min}$ then O turns out to be an inlier.
2. *Always Outlier (OL)*: If $D_O^{k_{min}} > r_{max}$ then O is classified as outlier.
3. *Outlier Candidate (OC)*: $D_O^{k_{max}} > r_{min}$ or $D_O^{k_{min}} \leq r_{max}$ then O becomes a candidate outlier.

Given the above classification of objects, indexing of objects in OC category helps in identifying which of them will be outliers/inliers for each selection of the parameters k and r within their respective dynamic ranges. Basically, the ONION framework supports the following three types of outlier analyzes on data.

1. *Comparative Outlier analysis (CO)*: Given a set of objects O_{in} which the user wants to detect as outlier, this framework identifies the set of other objects which also become outliers when objects in O_{in} are outliers.
2. *Outlier-centric Parameter Space Exploration (PSE)*: The idea here is to find the set of parameter value pairs (k, r) which make the set of outliers as close as the user specified set O_{in} of objects.

3. *Outlier Detection (OD)*: For a given pair of parameter values (k, r), the framework determines the set of outliers accordingly.

To support above listed interactive analysis, this framework makes use of three types of index structures named as ONION space (O-space), parameter space (P-space) and data space (D-space). In this manner, the ONION framework enables interactive outlier analysis trying to match with the users intent. However, it assumes the underlying data sets to be static and updates to the data sets are not accounted for in its methodology.

Given the dynamic nature of several real life data sets, it becomes inevitable to deal with such data for outlier analysis. When the data set is dynamic, one needs to deal with insertions, deletions and modifications of objects in the data. Also, such operations may cause changes to outlier/inlier status of existing objects in the data set in addition to the newly inserted or modified objects.

Catering to the above requirement, the method named as Dynamic Interactive Outlier (DIO) analysis works based on the computational model of the ONION framework. As per this method, the P-space index structure offers better performance over O-space and incremental update of D-space is found to be computationally challenging. Thus, the focus of DIO method is on extending the P-space analysis to dynamic data sets. Basically, it is required to update the index structure depending the current status of the objects in the dynamic data set and the same is performed as per the steps given below.

- Find the objects whose state changes depending on inserts, deletes and modifications, namely determine the updated OC set based on the changed status of the objects.
- Refresh P-space as per the revised OC set of objects.

Thus, the DIO method efficiently achieves P-space updation by making use of a grid structure and the extended object table to enable interactive outlier analysis on dynamic data sets.

9.4.3 Multi-level Analysis on Volatile Graphs

A multi-level analysis approach named as metric forensics for mining volatile graphs comprises three distinct components: (a) a suite of graph metrics, (b) a collection of analysis techniques, and (c) a multi-level approach.

According to this method, a *volatile graph* is defined to be a stream of duration-stamped edges in the form: $< v_{src}, v_{dst}, st\_time, duration >$, with potentially infinite number of nodes, and edges may appear and disappear. For example, IP-to-IP communication graphs and physical proximity graphs (such as blue-tooth connections) are of this kind. Similarly, a *snapshot graph* G_i is defined by its vertices V_t and edges E_t, which are active at time t. Due to the high volatility of the data, it is not worthwhile to consider the snapshot graphs in isolation. Hence, a *summary*

Table 9.1 A subset of the global graph metrics

Sl.No	Metric name	Time complexity
1	Number of active vertices	$O(1)$
2	Number of active edges	$O(1)$
3	Average vertex degree	$O(1)$
4	Average edge weight	$O(1)$
5	Maximum vertex degree	$O(N_T)$

graph $G_T = (V_T, E_T)$ is constructed to summarize all snapshot graphs during a time period T by employing a competent strategy.

The purpose of this method is to characterize a given volatile graph and detect interesting events that deviate from the usual behavior. This is achieved by computing and monitoring a suite of graph metrics that are classified into three groups based on their topological granularity: (i) global, (ii) community, and (iii) local. These metrics are of varying levels of complexity and computational intensity. As listed in Table 9.1, the global metrics generally measure high-level properties of the graph, while the local ones include centrality measures of the individual vertices.

The collection of analysis techniques employed here belong to three categories, namely single metric analysis, coupled metric analysis and non-metric analysis.

1. Single metric analysis techniques include log plots, outlier detection techniques such as Local Outlier Factor, etc.
2. On the other hand, coupled metric analysis techniques consider two or more metrics in unison and subject them to correlation analysis, with the metric values not satisfying their typical relationship being tagged as an interesting event.
3. Typical non-metric analysis includes visualization tools and attribute data inspection.

The genesis of this method is to enable multi-level analysis of the input summary graphs in three distinct dimensions: (1) time, (2) topology and (3) analysis automation.

1. The temporal granularity is adjusted by modifying the interval between summary graphs.
2. At the coarsest level of topology, only the global metrics computed on each summary graph are considered for analysis. When an interesting event is discovered at this level, it follows a *drill down* approach to examine the offending graph using more sophisticated analysis techniques at the next (finer) level.

3. Analysis automation deals with assessing the applicability of a particular analysis task at different levels of analysis. Typical analysis tasks include eigen analysis, correlation analysis, fractal dimension analysis, etc.

As part of the multi-level analysis, this method is capable of highlighting four types of interesting phenomena in volatile graphs:

- *Elbow*: In eigen analysis, the special regions in the λ_2 versus λ_1 plot, where λ_1 is stable and λ_2 is changing, or vice-verse, appear like elbow patterns. It occurs when the dominant phenomenon and the secondary phenomenon swap their roles.
- *Broken correlations*: In correlation analysis, the region where λ_1 is stable but λ_2 is changing, λ_1 will not be correlated to any graph metric in that region resulting in broken correlation. Such a scenario signifies the existence of meta-level correlations between broken correlations and elbow patterns.
- *Prolonged spikes*: In fractal dimension analysis, when the fractal dimension of a graph metric suddenly drops down, the corresponding scenario is referred to as prolonged spike. Such a pattern indicates an activity that has low-volume, but persists for a long time.
- *Lightweight stars*: In the vertex level local analysis, certain vertices form very big star-like structures with very low total edge-weights. Such patterns are referred to as lightweight stars.

One can observe the patterns described above in real life data by considering relevant data sets such as the IP traffic of an enterprise network.

9.4.4 Role-Based Analysis of Networks

An important task regarding the analysis of dynamic networks is to understand the roles of the objects involved in the network formation and exploring their evolving patterns. Learning the temporal structural behavior of individual nodes automatically and also identifying unusual patterns are of considerable interest to network analysts.

The general approach for role-based network analysis is to define different roles based on some specific topological properties. The behavioral roles are defined as a combination of similar structural features that are learned from the initial network. Thus, each role represents a distinct structural pattern, and two nodes with similar connectivity pattern belong to the same role.

9.4.4.1 Discovering Structural Roles using NMF

Representation, tracking and analysis of the structural dynamics of large networks is essential for exploring the global network dynamics as well as the local node dynamics. This in turn enables identification of interesting node and network patterns such as stationary and non-stationary roles, spikes/steps in role-memberships, increasing/decreasing role trends. A methodology developed for accomplishing this task, named as ROLE-DYNAMICS, relies upon the following rationale.

- The importance of the learned structural patterns fluctuate and eventually change entirely over time. The frequency of the fluctuations and changes depends completely on the dynamical system. For instance, the structural behavioral dynamics present in the Twitter social network at its inception are most likely different from the structural dynamics observed today.
- The node structural behavioral dynamics are non-stationary, i.e., they change or fluctuate over time. For example, the structure induced by emails for a given user may appear contrasting during office hours from the rest of the day.

Given a sequence of graphs, the ROLE-DYNAMICS method automatically learns a set of representative features, extracts these features from each graph, then discovers behavioral roles, and iteratively extracts these roles from the sequence of node-by-feature matrices over time. Various computational steps constituting this method are furnished below.

1. *Graph representation*: The representative features such as the degree features and the ego-net features are considered in this method. (By recursively combining these features, the regional features capturing the behavioral information are produced.) At time point t, a node by feature matrix V_t of size $n_t \times f$ is extracted from the corresponding graph instance, where n_t is the number of active nodes and f is the number of features. The sequence of node-by-feature matrices for the graph sequence is denoted as $\{\mathbf{V}_t : t = 1, \ldots, t_{max}\}$.
2. *Role discovery*: Structural roles are discovered using the representative graph features by employing the Non-negative Matrix Factorization(NMF) technique in association with Minimum Description Length (MDL) principle. For a non-negative matrix $\mathbf{V}_t \in \Re^{n_t \times f}$ and a positive integer $r < min(n_t, f)$, the factor matrices $\mathbf{G}_t \in \Re^{n_t \times r}$ and $\mathbf{F} \in \Re^{r \times f}$ are found such that the following functional is minimized.
$$f(\mathbf{G}_t, \mathbf{F}) = \frac{1}{2}\|\mathbf{V}_t - \mathbf{G}_t\mathbf{F}\|_F^2$$
The MDL principle is used to automatically select the number of behavioral roles r such that the model complexity (# of bits) and model errors are balanced. This way of detecting roles in a network has two advantages: (i) it doesn't require prior knowledge of the kinds of roles that may exist, and (ii) it appropriately assigns a mixed membership of the roles to each node in the network.
3. *Global network dynamics*: The above formulation of role discovery can be used to analyze network dynamics. An alternative and better approach is to learn a single global set of roles and track them as they change over time. Accordingly, a global set of features $\psi^* = \psi_1 \cup \psi_2 \cup \cdots \cup \psi_{t_{max}}$ is constructed using the set of features ψ_t corresponding to the graph instance at time t. Then, the sequence of node-by-feature matrices $\mathbf{V}_t \in \Re^{n \times f}$ are extracted. A single global node-by-feature matrix $\mathbf{V}_g \in \Re^{(n \times t_{max}) \times f}$ is constructed by stacking the

individual node-by-feature matrices. Employing the NMF technique mentioned above, matrix factorization is carried out as $V_g \approx \xi F_g$. Then, the node-role memberships $\xi = \{\mathbf{G}_t : t = 1, \ldots, t_{max}\}$ are iteratively estimated using F_g and $V_g = \{\mathbf{V}_t : t = 1, \ldots, t_{max}\}$. The matrix G_t indicates the current role membership of each active node at time t.

4. *Local node dynamics*: The structural patterns of individual nodes are tracked using the relative role importance measure given by $\mathbf{x}_t = \mathbf{G}_t^T \mathbf{e}/n_t$. Where, $\mathbf{e}$ is a vector of ones and $\mathbf{G}_t^T$ indicates the transpose of the node-by-role matrix at time t. This measure implies when a node has increasing or decreasing trends of structural behavior, as well as periodicity, or if the node dynamics are relatively stable. In addition to tracking, one can even detect the time at which the dynamics of a node change.

Given a time-series of role memberships for an individual node, one can identify the unique roles for that node and detect when its behavior deviates from its past behavior. Thus, it is possible to detect node anomalies in a dynamic network. Similarly, one can even detect network anomalies if the behavior of the entire network deviates like that of a node.

9.4.4.2 A Probabilistic Approach for Learning Object Roles

A more recent work on role-based analysis of dynamic networks says that defining roles based on topological properties has certain drawbacks as listed below.

- The topological properties in each task are selected subjectively.
- It may not be possible to characterize complex roles using any existing topological property.
- Applicability of the methods developed for static networks towards analyzing dynamic networks.

In view of the above issues, a possible solution is to consider the latent feature representation of object roles. According to this method, objects of similar roles will have close latent feature representations. This method considers the role of an object as a part of its properties that determine to what extent and how the object impacts the other objects in the formation and the dynamics of the network. This method performs the following three tasks on dynamic networks.

- Learning the role of each object at each snapshot of the dynamic network
- Analyzing how object roles evolve over time and
- Predicting the object roles at the $(t + 1)$-th snapshot of the network using the first to the t-th snapshots.

1. *Role learning on static networks*: The probabilistic method extracts the latent feature representation $\Lambda_{n \times r}$ (here n is the number of nodes and r is the number of roles) of object roles as well as the role interaction matrix H from an observable link matrix E of a static network as follows.

$$p(\Lambda, H|E, \sigma^2) = \frac{p(E|\Lambda, H, \sigma^2)p(\Lambda)p(H)}{p(E)} \tag{9.3}$$

In supervised setting, it is assumed that there are several labeled objects at the first time point to guide the learning of each type of object roles and to maximize the margin between different object roles. Let $Y_{m\times c}$ be the label matrix of the first snapshot, where m is the number of the labeled objects. Let $W_{r\times c}$ be the feature coefficient matrix which measures the contribution of each future in Λ to the object role classes. Given E and Y, the latent feature representation Λ of object roles, the interaction matrix H and the coefficient matrix W for learning object role classes on Λ can be obtained by maximizing the following expression (from Eq. 9.3)

$$p(\Lambda, H, W|E, Y) = \frac{p(E|\Lambda, H)p(Y|\Lambda, W)p(\Lambda)p(H)p(W)}{p(E)} \tag{9.4}$$

2. *Role detection and analysis on dynamic networks*: The probabilistic model described above is extended for detecting and analyzing object roles on dynamic networks. Let E^i denote the observed link matrix and the pair of low-rank matrices $D^i = \{\Lambda^i, H^i\}$, which approximates E^i, captures the role of each object and the interaction patterns of these roles at time point i. Given previously extracted feature $D^{1:i-1}$ and the current observable network E^i, one can learn the current latent object roles Λ^i and their interaction pattern H^i by maximizing the expression given below.

$$p(\Lambda^i, H^i, E^i, D^{i-1}) \simeq p(E^i|\Lambda^i, H^i)p(\Lambda^i)p(H^i)p(\Lambda^i|\Omega^i_\Lambda, \Sigma^i_\Lambda)p(H^i|\Omega^i_H, \Sigma^i_H) \tag{9.5}$$

Here, Ω^i_Λ and Ω^i_H are the Gaussian priors, which are the best estimations of Λ^i and H^i based on $D^{1:i-1}$, respectively. $\Sigma^i_\Lambda = diag\{\Sigma^i_{\Lambda_1}, \Sigma^i_{\Lambda_2}, \ldots, \Sigma^i_{\Lambda_r}\}$ and $\Sigma^i_H = diag\{\Sigma^i_{H_1}, \Sigma^i_{H_2}, \ldots, \Sigma^i_{H_r}\}$ are the corresponding variance matrices.

The solutions for the optimization problems given in Eqs. 9.4 and 9.5 can be obtained using the variational Bayesian inference technique. Then, the conditional probability $p(F_j|V_i)$ that object V_i belongs to the role class F_j is estimated and each object is assigned to the class in which the object has the highest probability. The trend of varying $p(F_j|V_i)$ over different time points reveals the dynamic patterns of object roles.

3. *Prediction on dynamic networks*: Given the first t observable snapshots of a dynamic network, the object roles at time point $(t+1)$ can be predicted by maximizing the following expression.

$$p(\Lambda^{t+1}|E^{1:t}) = \sum_{\Lambda^t} p(\Lambda^{t+1}|\Lambda^t)p(\Lambda^t|E^{1:t}) \tag{9.6}$$

Here $E^{1:t}$ are all observable and Λ^t has already been extracted.

9.5 Summary

Exploring the evolution of dynamic graph/network is an important research problem with widespread applications. Analysis of dynamic networks can be attempted at different levels of granularity such as nodes, communities, etc. However, carrying out evolution analysis based on the communities present in a network is more natural as they summarize the network structure effectively. Employing an acceptable model for characterizing the network formation is essential for understanding its dynamics. While the statistical models describe how networks form, game theoretic reasoning provides the necessary explanation for this phenomena as the equilibrium outcomes implemented by rational players.

Some of trending applications of dynamic network analysis in real life are presented in this chapter indicating the specific methodology adopted in each case.

- Detection of fraudsters in connection with web services and online social networks is important for initiating a directed action on them from system security point of view. This can be achieved through dense block identification and ranking of the individuals involved based on their suspicious scores computed using a fitting metric.
- Interactive analysis of outliers in dynamic network data facilitates the analyst to detect the outliers matching with his expectation. This is enabled through customized assignment of values to the parameters of the outlier detection algorithm based on the pre-determined dynamic ranges of values for each one of the parameters.
- IP-to-IP communication graphs and physical proximity graphs such as blue-tooth connections are volatile in nature. Analysis of such volatile graphs can be attempted by constructing summary of the snapshot graphs pertaining to a specific time period. Then, necessary graph metrics can be computed based on the summary to detect any interesting event that deviates from the usual behavior.
- Understanding the roles of the objects involved in the network formation and exploring their evolving patterns is a relatively new approach for dynamic network analysis. This enables the exploration of the global network dynamics as well as the local node behavior. A way of discovering these behavioral roles is to iteratively process the snapshot graphs over time using competent techniques such as NMF.

9.6 Bibliographic Notes

Various methodologies were proposed for understanding the application-specific insights of evolution in graphs/networks [25]. Similarly, a detailed survey of the methods for analyzing evolving networks was brought out in a recent publication [12]. Some of the details regarding representation of networks and carrying out certain computational task over dynamic networks was presented in [21].

From the analysis point of view, detecting the communities present in a network [8] is interesting. Likewise, evolutionary clustering is another interesting mechanism for dynamic network analysis. The on-line clustering algorithm proposed in [6] is one of the earliest methods for evolutionary clustering. Similarly, the method proposed in [9] is a tightly integrated model that can simultaneously perform clustering and evolution analysis. It uses a probabilistic mixture model to characterize the clusters in a heterogeneous network.

Typical data mining tasks such as classification and clustering on the network data enable some or the other form of analysis on the dynamic network. Belonging to the variety of classification methods, a random walk-based approach was proposed [1] for the node classification problem in evolving networks. A soft clustering model that assigns probabilities of membership of each node to different clusters was proposed in [9]. Subsequently, a method for detecting evolutionary community outliers was presented in [10]. It was shown in [22] that large groups typically persist longer and it was observed that the evolution of communities is gradual [26]. A method of identifying key/influential entities in a network using various centrality measures defined on the nodes was presented in [28].

A detailed survey on the evolution of social networks can be found in [18]. An interesting trend in evolving social networks/graphs is densification with shrinking diameters caused by addition of new edges [19]. This phenomena can be explained based on the evolution analysis laws derived analytically from the notion of preferential attachment [2]. In the context of social networks, the study presented in [3] shows that the propensity of individuals to join communities depends on the network structure.

A discussion on modeling the evolution of dynamic networks using stochastic processes and evolutionary game theory was presented in [12]. It highlighted the potential connections between these two seemingly different modeling strategies for network formation and evolution.

There are numerous trending applications of dynamic network analysis in various real life scenarios. An approach for scoring entities based on their participation in suspicious dense blocks was developed [16] by proposing a set of axioms that any ideal individual scoring metric should satisfy. Similarly, an interactive outlier analysis method named as ONION enables the users to choose applicable parameter values so as to detect outliers as per their requirements [5]. A method for carrying out Dynamic Interactive Outlier (DIO) analysis [24] was proposed based on the computational model of the ONION framework to deal with the analysis needs of dynamic data sets. Likewise, a multi-level analysis approach named as metric forensics was proposed [4, 13] for mining the volatile graphs encountered in many real life applications.

An important task regarding the analysis of dynamic networks is to understand the roles of the objects involved in the network formation and exploring their evolving patterns [20]. A non-parametric approach named as ROLE-DYNAMICS was proposed in [23] for representing, tracking and analyzing the structural dynamics of large networks. This method makes use of representative features such as the degree features and the ego-net features, as described in [15]. Based on these representative graph features, the methodology presented in [14] discovers structural roles using the Non-negative Matrix Factorization(NMF) technique [17] with Minimum Description Length (MDL) principle. Another effort on role-based analysis of dynamic networks pointed out that defining roles based on topological properties had certain drawbacks [20]. A novel framework was proposed based on the latent feature representation of object roles. This framework is based on the variational Bayesian inference technique [7] so as to find solution of the optimization problems related role detection leading to efficient analysis of dynamic networks. Similarly, a methodology for learning linguistic descriptors of user roles in online communities was discussed in [27]. Likewise, a more recent technique for role discovery in graphs was proposed employing the global features [11].

References

1. Aggarwal, C., Li, N.: On node classification in dynamic content-based networks. In: SDM, pp. 355–366 (2011)
2. Albert, R., Barabasi, A.L.: Statistical mechanics of complex networks. Rev. Mod. Phys. **74**(1), 47 (2002)
3. Backstrom, L., Huttenlocher, D.P., Klienberg, J.M., Lan, X.: Group formation in large social networks: membership, growth, and evolution. In: KDD, pp. 44–54 (2006)
4. Breunig, M., Kriegel, H., Ng, R., Sander, J.: Lof: identifying density-based local outliers. In: ACM SIGMOD International Conference on Management of Data, Dallas, Texas, pp. 93–104 (2000)
5. Cao, L., Wei, M., Yang, D., Rundensteiner, E.A.: Online outlier exploration over large datasets. In: KDD. ACM (2015)
6. Chakrabarti, D., Kumar, R., Tomkins, A.: Evolutionary clustering. In: KDD, pp. 554–560 (2006)
7. Fox, C., Roberts, S.: A tutorial on variational Bayesian inference. Artif. Intell. Rev., 85–95 (2012)
8. Girvan, M., Newman, M.E.J.: Community structure in social and biological networks. PNAS **99**(12), 7821–7826 (2002)
9. Gupta, M., Aggarwal, C., Han, J., Sun, Y.: Evolutionary clustering and analysis of bibliographic networks. In: ASONAM, pp. 63–70 (2011)
10. Gupta, M., Gao, J., Sun, Y., Han, J.: Integrating community matching and outlier detection for mining evolutionary community outliers. In: KDD, pp. 859–867 (2012)
11. Gupte, P.V., Ravindran, B., Parthasarathy, S.: Role discovery in graphs using global features: algorithms, applications and a novel evaluation strategy. In: 33rd International Conference on Data Engineering (ICDE), pp. 771–782. IEEE Computer Society, San Diego, CA, USA (2017)
12. Hellmann, T., Staudigl, M.: Evolution of social networks. Eur. J. Oper. Res. **234**, 583–596 (2014)
13. Henderson, K., Eliassi-Rad, T., Faloutsos, C., Akoglu, L., Li, L., Maruhashi, K., Prakash, B.A., Tong, H.: Metric forensics: a multi-level approach for mining volatile graphs. In: KDD, pp. 163–172. ACM (2010)
14. Henderson, K., Gallagher, B., Eliassi-Rad, T., Tong, H., Basu, S., Akoglu, L., Koutra, D., Faloutsos, C., Li, L.: Rox: structural role extraction and mining in large graphs. In: KDD. ACM (2012)
15. Henderson, K., Gallagher, B., Li, L., Akoglu, L., Eliassi-Rad, T., Tong, H., Faloutsos, C.: It's who you know: graph mining using recursive structural features. In: KDD, pp. 1–10. ACM (2011)

16. Lamba, H., Hooi, B., Shin, K., Faloutsos, C., Pfeffer, J.: zooRANK: ranking suspicious entities in time-evolving tensors. In: ECML-PKDD. LNCS, pp. 68–84. Springer, Skopje, Macedonia (2017)
17. Lee, D.D., Seung, H.S.: Learning the parts of objects by non-negative matrix factorization. Nature (1999)
18. Leskovec, J., Backstrom, L., Kumar, R., Tomkins, A.: Microscopic evolution of social networks. In: KDD, pp. 462–470 (2008)
19. Leskovec, J., Kleinberg, J.M., Faloutsos, C.: Graphs over time: densification laws, shrinking diameters and possible explanations. In: KDD, pp. 177–187 (2005)
20. Li, K., Guo, S., Du, N., Gao, J., Zhang, A.: Learning, analyzing and predicting object roles on dynamic networks. In: ICDM, pp. 428–437. IEEE (2013)
21. Morgan, G.P., Frankerstein, W., Carley, K.M.: Introduction to dynamic network analysis. In: BRiMS (2015)
22. Palla, G., Barabasi, A., Vicsek, T.: Quantifying social group evolution. Nature **446**(7136), 664–667 (2007)
23. Rossi, R., Neville, J., Gallagher, B., Henderson, K.: Role-dynamics: fast mining of large dynamic networks. In: WWW, pp. 997–1006. ACM (2012)
24. Sakazume, C., Kitagawa, H., Amagasa, T.: DIO: efficient interactive outlier analysis over dynamic datasets. In: Twelfth International Conference on Digital Information Management (ICDIC), pp. 228–235. IEEE, Fukuoka, Japan (2017)
25. Spiliopoulou, M.: Evolution in social networks: a survey. In: Social Network Data Analytics, pp. 149–175. Springer (2011)
26. Tantipathananandh, C., Berger-Wolf, T., Kempe, D.: A framework for community identification in dynamic social networks. In: KDD, pp. 717–726. ACM (2007)
27. Wang, A., Hamilton, W.L., Leskovec, J.: Learning linguistic descriptors of user roles in online communities. In: Proceedings of the First Workshop on NLP and Computational Social Science, Austin, TX, USA, pp. 76–85 (2016)
28. Wasserman, S.: Social Network Analysis: Methods and Applications, vol. 8. Cambridge University Press (1994)

Chapter 10
Detecting Anomalies in Dynamic Networks

Abstract This chapter deals with an important analysis task over dynamic networks, namely exploring the time varying characteristics of anomalies present in such networks. In this direction, a graph mining based framework is considered that takes a sequence of network snapshots as input for analysis. It defines various categories of temporal anomalies typically encountered in such an exploration and characterizes them appropriately to enable their detection. An experimental study of this framework over benchmark graph data sets is presented here to demonstrate the evolving behavior of the anomalies detected as per the categories defined.

10.1 Introduction

Graph based representation of network structure gave rise to the rapid proliferation of research work on social network analysis. Similarly, graph based representation is preferred for modeling the dynamism inherent to many real life applications such as pervasive computing. Thus, graph mining plays an important role in any meaningful analysis of the network data. An emerging research problem related to graph mining is to discover anomalies (also known as outliers) present in the graph representation of network data. Due to the networked nature of the data pertaining to several real life applications such as computer communication networks, social networks of friends, the citation networks of documents, hyper-linked networks of web pages, etc., anomaly/outlier detection in graphs turns out to be an important pattern discovery activity. It provides the basis for realizing certain application specific tasks such as fraud detection in on-line financial transactions, intrusion detection in computer networks, etc.

In case of Internet Protocol (IP) networks comprising several individual entities such as routers and switches, the behavior of the individual entities determine the ensemble behavior of the network. In this context, network anomalies typically refer to circumstances when network operations deviate from normal behavior, resulting in numerous unusual traffic patterns noticeable in such network data. These traffic patterns arise due to different types of network abuse such as denial of service attacks, port scans, and worms; as well as from legitimate activity such as transient changes

N. N. R Ranga Suri et al., *Outlier Detection: Techniques and Applications*,
Intelligent Systems Reference Library 155,
https://doi.org/10.1007/978-3-030-05127-3_10

in customer demand, flash crowds, or occasional high-volume flows. These traffic anomalies pose major security concern to the network administrators and it is often difficult to detect them, more so in evolving networks.

Connected with any dynamic network, the underlying graph representation undergoes time dependent changes in its connectivity structure. This will have a consequence on the local connectivity and thus influences the temporal behavior of various graph entities such as nodes, edges and subgraphs. As a result, a graph entity that qualifies to be an outlier (irregular connectivity pattern) at the current instance may no more continue to be the same at a later point of time due to the changes in connectivity structure of the graph. Additionally, some new irregularities may arise in a different portion of the graph hinting at the presence of some new outliers. If the graph connectivity structure is altered further, some intermediate outliers may cease to exist and further new types of outliers may start appearing in the graph. Thus, it is important to track such evolving scenarios for a subjective analysis of the graph representation towards understanding and characterizing the outlier dynamics. Accordingly, the study of evolving graphs/networks has been an active research problem.

Motivated by the above discussion, a framework for characterizing temporal anomalies in evolving networks is presented in this chapter. This involves identifying various graph outliers and characterizing their dynamics across multiple instances of the graph representation of a dynamic network at various points of time during its evolution. Accordingly, the underlying algorithm detects various temporal outliers in a dynamic graph and produces a semantic categorization of the detected outliers based on different categories of temporal outliers defined.

10.2 Outlier Detection in Graph Data

As brought out earlier, the essential graph mining task here is to detect various anomalies/outliers present in the graph representation of network data. A recent method for discovering different types of anomalies in graphs makes use of the notion of *egonet*. The egonet of a node is defined as the induced subgraph of its 1-step neighbors as shown in Fig. 8.2a. According to this method, the number of nodes N_i and the number of edges E_i of the egonet G_i of node i in a graph $G = (V, E)$ follow a power law relationship. Consequently, two types of anomalous egonets, named as *near cliques* and *near stars*, are determined by measuring the amount of deviation from the power law relationship.

Similarly, a method for spotting significant anomalous regions in dynamic networks determines the anomalous score of a weighted edge in an undirected connected graph based on a statistical measure. The aim of this method is to find contiguous regions having adjacent anomalous edges forming large regions of higher score, named as Significant Anomalous Regions (SAR) in a dynamic network. The anomaly score of such a region is quantified by aggregating the scores of the participating edges. Given an edge-weighted graph $G = (V, E, W)$, the anomaly score of a temporal network region $R = (G', [i, j])$, where $G' = (V', E')$ is a connected subgraph of G, in the time interval $[i, j]$ is given by

$$score_G(R) = \sum_{e \in E'} \sum_{t=i}^{j} w^t(e) \tag{10.1}$$

where $w^t(e)$ is anomaly score associated with an edge e at time point t.

This method is basically an edge-centric approach where the anomalous scores of edges determine the anomalous regions in a graph. Also, computing aggregate score over time may not highlight the temporal characteristics of the primitive outliers such as node/edge outliers in a graph.

10.3 Characterization of Outliers in Evolving Networks

Given the graph representation of network data, there may exist various types of graph outliers in the form of node outliers, edge outliers and subgraph outliers as discussed in the previous section. The process of graph evolution over time results in some interesting observations in terms of these graph outliers. A framework for capturing this phenomenon is presented here, as shown in Fig. 10.1. The rationale behind this setting is to explore various types of outliers present in a graph at various points of time independently and then to characterize them based on their appearance at different points of time during the network evolution.

10.3.1 Categories of Temporal Outliers

The symbolic notation along with some definitions demanded by the methodology for detecting temporal outliers in dynamic networks is furnished here.

Let $G = \{G_1, G_2, \ldots, G_N\}$ be a dynamic graph data set comprising of N instances of an evolving graph corresponding to the analysis time period $[t_1, t_N]$ covering N discrete time points. Similarly, let $\Psi_i = \{\psi_{i,1}, \psi_{i,2}, \ldots, \psi_{i,k}\}$ denote the top-k outliers present in the graph instance G_i at time point t_i and $S_i = \{s_{i,1}, s_{i,2}, \ldots, s_{i,k}\}$ be their outlier scores respectively. Based on this, the necessary terminology is given below.

- *Temporal outlier*: A graph-based outlier that is present in one or more instances of an evolving network/graph.
- *Cumulative outlier score*: The cumulative outlier score of a temporal outlier is computed using its outlier scores at different points of time during the analysis.

The exact computation of the cumulative outlier score depends upon the specific algorithmic logic considered based on the application context. The set of temporal

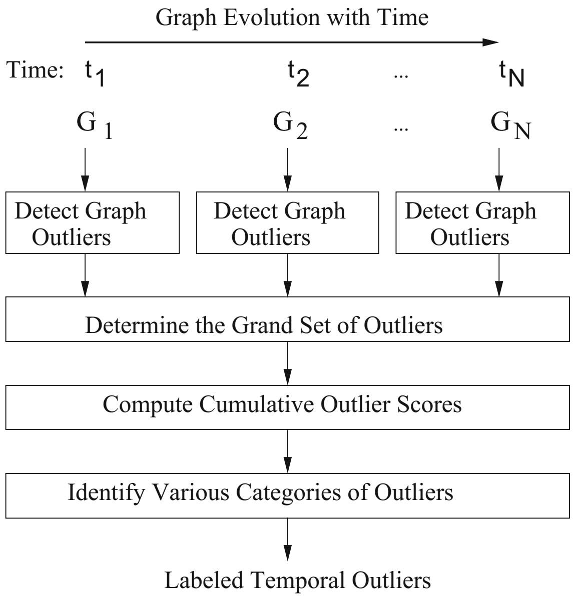

Graph Evolution with Time
Time: t1 t2 ... tN
G 1 G2 ... GN
Detect Graph Outliers
Detect Graph Outliers
Detect Graph Outliers
Determine the Grand Set of Outliers
Compute Cumulative Outlier Scores
Identify Various Categories of Outliers
Labeled Temporal Outliers

Fig. 10.1 Framework for analyzing time varying outliers

outliers detected by the method under consideration is indicated by Ω and $\Phi = \{\phi_1, \phi_2, \dots\}$ represents the cumulative scores of these outliers.

In order to characterize the dynamics of the temporal outliers, a categorization based on their time varying characteristics is brought out first. In this regard, every category of temporal outlier that is noticed during the graph evolution is assigned an explicit label to mark it distinctly depending on its behavior. In order to capture various behavioral patterns of the temporal outliers and to convey the underlying semantics associated with each such pattern, the following five categories of temporal outliers are defined.

1. *Halting outlier*: A temporal outlier O_H that may cease to exist over time during the network evolution.
   $\exists\, t_j \in [t_3, t_N]\ s.t.\ (O_H \in \Psi_p,\ \forall t_p \in [t_1, t_j))\ \wedge\ (O_H \notin \Psi_j)$.
2. *Emerging outlier*: A temporal outlier O_E that starts appearing at some point of time.
   $\exists\, t_i, t_j \in [t_2, t_N]\ s.t.\ (O_E \notin \Psi_p,\ \forall t_p \in [t_1, t_i])\ \wedge\ (O_E \in \Psi_q,\ \forall t_q \in [t_{i+1}, t_j])\ \wedge$ $(t_j > t_{i+1})$.
3. *Alternating outlier*: A temporal outlier O_A that is noticed at different points of time intermittently.
   $\exists\, t_i, t_j, t_k \in [t_1, t_N]\ s.t.\ (O_A \in \Psi_i)\ \wedge\ (O_A \notin \Psi_j)\ \wedge\ (O_A \in \Psi_k)\ \wedge$
   $(O_A \in \Psi_p,\ \forall t_p \in (t_i, t_j))\ \wedge\ (O_A \notin \Psi_q,\ \forall t_q \in (t_j, t_k))\ \wedge\ (t_k > t_j > t_i)$.

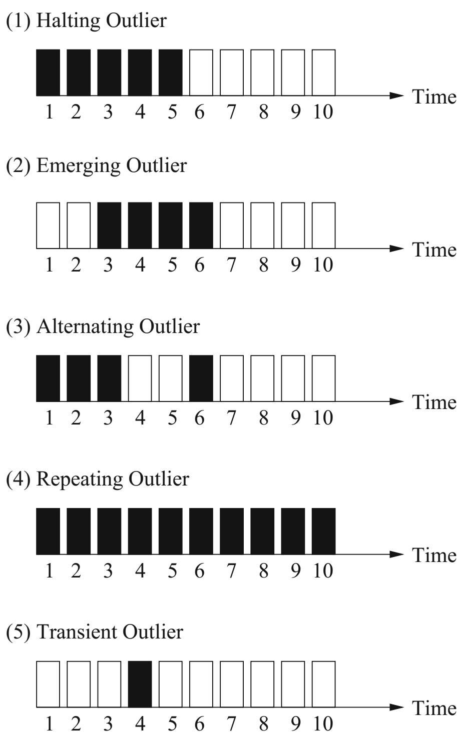


Fig. 10.2 Dynamics of various categories of temporal outliers

4. *Repeating outlier*: A temporal outlier O_R that keeps appearing at all points of time during the network evolution.
   $O_R \in \Psi_i, \forall\, t_i \in [t_1, t_N]$.
5. *Transient outlier*: A temporal outlier O_T that exists only at one point of time.
   $\exists\, t_i \in [t_1, t_N]\; s.t.\; (O_T \in \Psi_i) \;\wedge\; (O_T \notin \Psi_j,\; \forall t_j \in [t_1, t_N]) \;\wedge\; (t_j \neq t_i)$.

The observable dynamics of various categories of temporal outliers with network evolution are depicted in Fig. 10.2. A filled box at a particular time point t_i implies the presence of the corresponding temporal outlier in the graph instance G_i. The time-varying dynamics observed for a reasonably long period of time gives an idea of the presence of various categories of temporal outliers.

It is important to note that a halting outlier is different from a transient outlier, where the former one disappears after being present for more than one time point, while the later one exists precisely at one time point. Also note that the set of top-k outliers Ψ_i present in a graph instance G_i may consist of one or more categories of temporal outliers described above. Thus, anomaly/outlier detection in dynamic networks can be carried out for an effective management of networks, such as ensuring the security of the IP networks.

10.3.2 NetHEART Algorithm

The framework shown in Fig. 10.1 detects various categories of temporal outliers present in a dynamic network. Accordingly, the algorithm concerned is named as *NetHEART* reflecting the names of the categories of outliers being detected. According to this framework, the initial task is to detect the outliers present in a single graph instance. This can be accomplished using the method described in Sect. 10.2 based on power law relationship. The rest of the framework deals with collating the instance specific outliers and characterizing their dynamics across multiple instances of a dynamic graph. Accordingly, the algorithm for detecting various temporal outliers in evolving graphs comprises the following computational steps.

1. For each graph instance G_i perform the following steps.
   a. Construct the corresponding edge list L_i .
   b. Detect the outliers present in graph instance G_i using L_i.
   c. Determine the set of top-k outliers Ψ_i and their scores S_i.
2. Determine the grand set of outliers across all graph instances as

$$\Omega = \Psi_1 \cup \Psi_2 \cup \cdots \cup \Psi_N \tag{10.2}$$

3. For each temporal outlier $O_j \in \Omega$, determine its cumulative outlier score as

$$\phi_j = \frac{1}{N} \sum_{i=1}^{N} \xi_{i,j} \tag{10.3}$$

   where

$$\xi_{i,j} = \begin{cases} s_{i,p} & \text{if } \exists p\ s.t.\ O_j = \psi_{i,p} \\ 0 & \text{otherwise.} \end{cases}$$

4. Determine the set of repeating outliers that exist across all graph instances as

$$\Omega^* = \Psi_1 \cap \Psi_2 \cap \cdots \cap \Psi_N \tag{10.4}$$

5. Similarly, determine all other categories of outliers based on Ψ_i's.
6. For each outlier $O_j \in \Omega$, label it as per its observed dynamics.
7. Arrange the set Ω of temporal outliers in descending order of their scores.

As per Eq. 10.2, the grand set of outliers (Ω) consists of at most $N * k$ elements and at least k elements. Here, N is the number of graph instances considered for analysis. Similarly, Eq. 10.3 determines the cumulative outlier score of a temporal outlier by computing the average of its scores corresponding to various graph instances. Unlike the other such methods, the detection of a temporal outlier is not affected by its cumulative outlier score. However, this score reflects the significance of a temporal outlier over all the graph instances. The computational complexity of NetHEART algorithm is determined by that of the specific method employed for determining the outliers in a single graph instance or snapshot.

10.4 Experimentation on Benchmark Data

An experimental study is carried out considering two real life dynamic graph data sets using NetHEART algorithm. For ease of illustration and simplicity in describing the results, only three graph instances ($N = 3$) are considered corresponding to each one of these data sets.

The DBLP Computer Science Bibliography from the University of Trier contains millions of bibliographic records bundled in a huge XML file. Various subsets of these records concerned with AAAI publications are considered here, denoted as 'DBLP-AAAI' data set, as per the details furnished in Table 10.1.

The graph of routers comprising the Internet can be organized into subgraphs called Autonomous Systems (AS). Each AS exchanges traffic flows with some neighbors (peers). It is possible to construct a communication network of who-talks-to-whom from the BGP (Border Gateway Protocol) logs. The experiments here are carried out on three graph files taken from the Autonomous Systems collection as per the details furnished in Table 10.1, denoted as 'AS-2000' data set hereafter.

10.4.1 Outlier Detection and Characterization

As per the first step of the algorithm, various node-based outliers present in each graph instance are detected. The resulting top-k (=10) outliers corresponding to each instance/snapshot of the 'DBLP-AAAI' graph are shown in Fig. 10.3, indicated by the node IDs in decreasing order of their outlier scores. Subsequently, the grand set consisting of 13 temporal outliers ($|\Omega| = 13$) is determined over all the instances

Table 10.1 Details of dynamic graph data sets used for anomaly detection

Data set name	Instance	# Nodes	# Edges	Graph instance details
DBLP-AAAI	G_1	5,581	25,276	Publications upto 2008
	G_2	6,219	28,622	Publications upto 2010
	G_3	7,551	35,628	Publications upto 2012
AS-2000	G_1	2,107	8,979	as19991231.txt
	G_2	3,570	14,783	as20000101.txt
	G_3	6,474	26,467	as20000102.txt

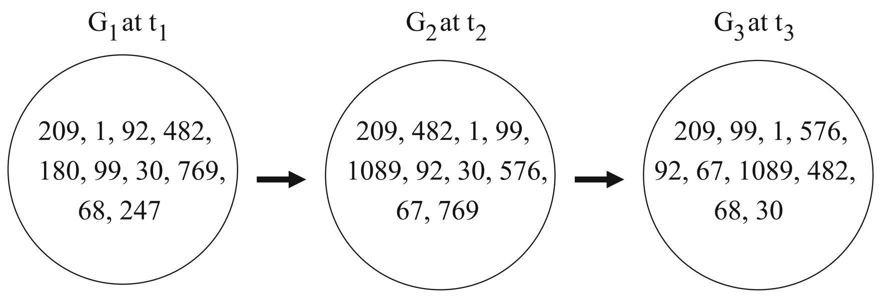


Fig. 10.3 Time varying characteristics of graph outliers in DBLP-AAAI data

Table 10.2 Categories of temporal outliers detected in dynamic graph sets

Graph file	Repeating outliers	Transient outliers	Halting outliers	Emerging outliers	Alternating outliers
DBLP-AAAI	209, 1, 92, 482, 99, 30	G_1:180, 247 G_2: – G_3: –	769	1089, 576, 67	68
AS-2000	4006, 4544 1746, 3563	G_1: 2551, 2041 145, 6467 G_2: 6261, 701 G_3: 5779, 5000 11329, 2828	721, 3847	6113, 668	–

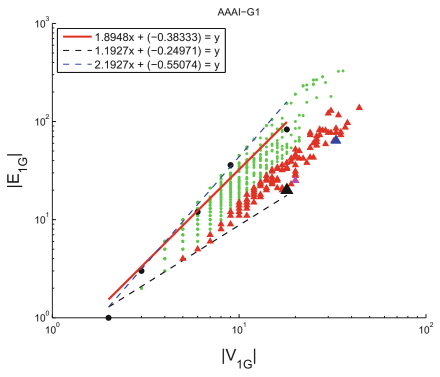
and the cumulative outlier score of each temporal outlier is also computed accordingly. Then, various categories of temporal outliers present in this graph data set are identified as listed in Table 10.2. Finally, a ranked list of labeled temporal outliers is produced as the output of this entire process.

Similarly, the AS-2000 graph data set is also subjected to various computational steps of the algorithm, and different categories of temporal outliers thus detected are furnished in Table 10.2 for completeness.

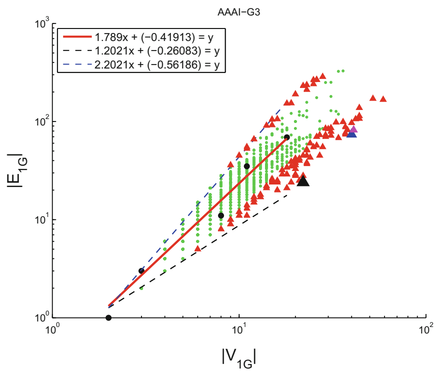
It is important to note that the temporal behavior observed in this experimentation may vary with the number (k) of top ranked outliers considered corresponding to each graph instance and the number of instances (N) considered in each graph data set, as the outlier ranking is basically a relative measure.

10.4.2 Exploring the Outlier Dynamics

As brought out in Sect. 8.2.1, the power law based method captures two types of node-based anomalies, namely the near-star anomalies appearing below the fitting line and the near-clique anomalies above the fitting line. Accordingly, the EDPL



(a) Outliers in G_1



(b) Outliers in G_3

Fig. 10.4 Top-100 outliers in DBLP-AAAI data at different time instances

plots obtained corresponding to different instances of the DBLP-AAAI data set are shown in Fig. 10.4. A keen observation of these plots suggests that as more and more new edges get added during the graph evolution from G_1 to G_3, one can notice the gradual appearance of more near-clique type of anomalies (triangle points marked in the upper portion of the plot) as shown in Fig. 10.4b.

Temporal outliers present in an evolving graph are expected to highlight the inherent semantics of the graph connectivity patterns causing these outliers. To illustrate these semantic aspects, a few of the detected node-based outliers, in the form of the egonet subgraphs, are depicted in Table 10.3. The number of nodes and the number of edges present in each such subgraph are also furnished along side in the same table. It is important to note that with the evolution of the graph over time, the underlying egonets have also undergone corresponding change in their connectivity structure. Though there is not any significant change expected in the node-based anomalous subgraphs individually, their relative position in the top-k outlier rank list may change across various graph instances (refer to Fig. 10.3) due to the modified connectivity structure of the graph during evolution.

10.4.3 Experimentation with More Instances

In order to experiment with NetHEART algorithm over a longer period of evolution ($N > 3$) of a dynamic network, the 'DBLP-AAAI' data set with 5 instances ($N = 5$) is considered. This is the same dynamic network/graph data set considered in the previous section with 2 additional graph instances, denoted as 'AAAI-Long' data set, as shown in Table 10.4.

Table 10.3 Sub-graphs of node outliers in different benchmark graphs

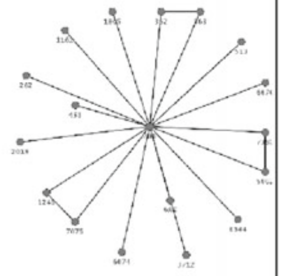
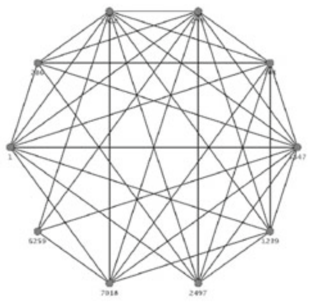
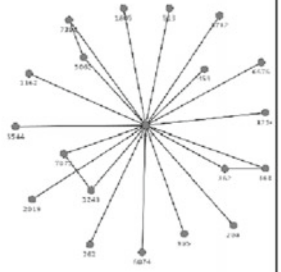
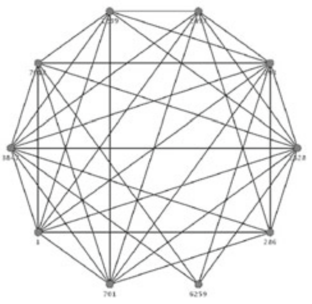
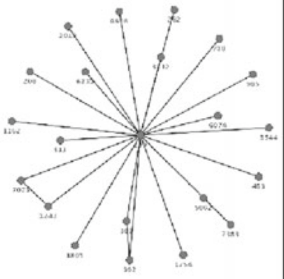
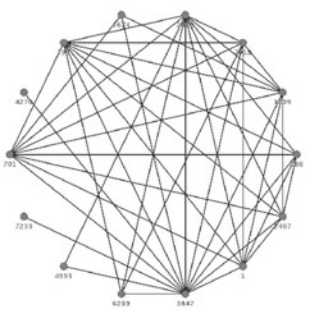
Graph File	DBLP-AAAI Node ID: 209	AS-2000 Node ID: 3847
Egonet in G_1 at t_1	 nodes=18, edges=20	 nodes=10, edges=37
Egonet in G_2 at t_2	 nodes=20, edges=22	 nodes=10, edges=37
Egonet in G_3 at t_3	 nodes=22, edges=24	 nodes=14, edges=44

Table 10.4 Details of AAAI-Long dynamic graph data set

Data set name	Instance	# Nodes	# Edges	Graph instance details
AAAI-Long	G_1	3,478	13,184	Publications upto 2004
	G_2	4,554	19,558	Publications upto 2006
	G_3	5,581	25,276	Publications upto 2008
	G_4	6,219	28,622	Publications upto 2010
	G_5	7,551	35,628	Publications upto 2012

Table 10.5 Categories of temporal outliers identified in AAAI-Long data

Graph file	Repeating outliers	Transient outliers	Halting outliers	Emerging outliers	Alternating outliers
AAAI-Long	209, 482	G_1:1191, 1753, 3505, 2147, 1017	247	180, 30, 92, 769, 1, 99 67	68, 576 1089

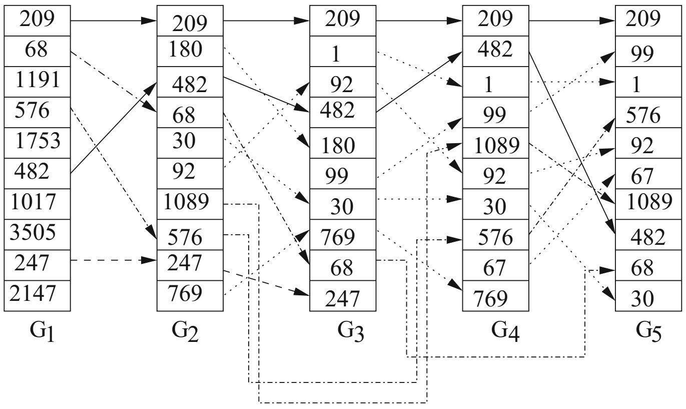


Fig. 10.5 Time varying characteristics of graph outliers in AAAI-Long data

Applying various steps of the algorithm, the grand set consisting of 18 temporal outliers ($|\Omega| = 18$) is determined corresponding to this data set. Various categories of temporal outliers are identified as listed in Table 10.5, and the same are depicted instance-wise in Fig. 10.5. It is important to note that the outlier dynamics observed over a longer period of network evolution is consistent with the categorization presented above.

10.5 Applications of Dynamic Network Anomaly Detection

Some of the recent applications dealing with anomaly detection task on dynamic networks are presented here to highlight the significance of the same in the contemporary research. This is also intended to motivate the researchers to explore further means of detecting temporal outliers addressing various application specific aspects in developing the requisite methodologies.

10.5.1 Anomalous Hotspot Discovery in Graph Streams

Anomalous hot spots show abrupt changes both in the magnitudes and patterns of interaction. Anomalous events may arise in a graph stream in two different ways: (i) significant changes in the magnitudes of the interaction may occur in neighborhood of a node in the network, and (ii) sudden changes in the structural edge patterns at a particular node. Addressing the challenge that changes could occur over different horizons in time, an approach exists for analyzing streams at multiple levels of temporal granularity. The rationale behind this approach is that the relative edge frequencies of different edges will stabilize over a long period of time, unless the frequencies of the edges are counted by providing greater importance to recent edges.

According to this method, the network stream implicitly defines a temporal network $G(t) = (N(t), A(t))$, where $N(t)$ is the set of all distinct nodes in the stream at time t, and $A(t)$ is a sequence of edges corresponding to all edges received so far. This method is designed to analyze stream data automatically at multiple levels of temporal granularity for simultaneous insights over short, medium and long-term horizons. To model the temporal aspect of edge frequencies, it associates a decay factor λ with each edge. This weight regulates the rate at which the importance of an edge decays over time. Various stream statistics are defined using edge weight to capture the temporal characteristics of the data.

Network change monitoring is done by maintaining the mean and standard deviation of various stream related statistics over time. A change is reported as significant, if it is found by any of the detectors corresponding to different values of λ of the multi level analysis.

From implementation point of view, if all nodes need to be tracked for computing anomalous changes at the same time, it is required to periodically store all the statistics for offline analysis. However, to track only a subset of the nodes, on-line analysis using this algorithm seems feasible. An experimental study of this method carried out using Internet Movie Database (IMDB) using four half-life time values ($\lambda = \frac{1}{t}$) reported the observations made during this multi-level analysis methodology discussed above.

10.5.2 Multi-Level Anomalies in Time-Varying Graph Data

Typically, graphs representing social networks consist of labeled nodes (representing individuals/entities) and edges among the nodes (representing interactions). The set of edges changes over time producing a sequence of snapshots of the interaction network. Analyzing networks of this kind involves identifying patterns and aberrations across snapshots that can point out areas of interest to the analyst. Specifically, availability of common node labels across various snapshots of the network should be accounted for more accurate detection of network anomalies. This is due to the inability of various graph models to distinguish isomorphic copies with different node labels.

A multi-scale technique detects anomalies in time-varying graph data using hierarchically connected distributions to detect associated abnormalities at three increasingly fine levels of granularity (namely at graph, subgraph and node levels). This technique introduced Generalized Block Two-level Erdos-Renyi (GBTER) model involving two hierarchical streaming anomaly detectors. This model takes the following input to generate graphs in two stages.

- Expected degree of each node
- Community assignment of the nodes, i.e., a partition of the vertex set into disjoint subsets, $\{C_j\}$, and
- An edge probability p_j for edges within each community C_j.

In the first stage, within community edges are sampled from an Erdos-Renyi model for each community C_j. Then, the second stage is used to generate both inter as well as intra community edges. GBTER model allows greater flexibility and assignment of community membership, size and internal edge density. This model is designed to identify anomalies caused not only in community density, but also changes in the interactions within or between communities. It estimates probability models from observations and new graph data is declared anomalous if it has p-value below the specified threshold. This model works for streaming graph data, as it updates the parameters to include the new graph in the observations.

Two multi-scale anomaly detectors are made available, one which uses the GBTER distribution directly and one which leverages statistics inherent to the GBTER model. Multi-Scale Probability Detector (MSPD) utilizes graph probability as determined by the GBTER model and decomposes it into probabilities of subgraphs and nodes for hierarchical information. On the other hand, Multi-Scale Statistics Detector (MSSD) works bottom up by defining the probability of a node based on the likelihood of its internal and external degree.

Given a graph $G = (V, E)$ with vertices V and edges E, the probability of G as computed by GBTER model is given by

$$P(G) = \prod_{(i,j)\in E} P(i, j) \prod_{(i,j)\notin E} (1 - P(i, j)) \tag{10.5}$$

where $P(i, j)$ is the probability of the edge (i, j) under the GBTER model. Then, the p-value is estimated using Monte-Carlo simulation.

To detect anomalies at different levels, the probability of a graph is decomposed into a product of subgraph probabilities. Hence, given a partition of V into communities, $\{C_i\}$, the probability of G is given by $P(G) = \prod_i P(C_i)$. The probability of a subgraph/community $G^{'} = (V^{'}, E^{'})$ is defined to be $\prod_{i \in V'} P(i)^{\frac{1}{2}}$. Here, the probability of a node i is defined as

$$P(i) = \prod_{j:(i,j) \in E} P(i,j) \prod_{j:(i,j) \notin E} (1 - P(i,j)) \tag{10.6}$$

In case of MSSD, node probability is defined as the joint probability of its degrees, $P(i) = P(d_{in}, d_{ex})$.

Experimental study of the above anomaly detectors is possible by generating labeled graph data with a regular model M_r and a seeded anomaly model M_a. It is empirically found that MSSD has better performance over MSPD due to the reasons explained above.

10.5.3 Streaming Data Analytics for Anomalies in Graphs

Due to complex and diverse nature of the network environments (like biological, communication and financial networks, etc.) it is important to identify threats that are dynamic by taking into account the structural aspects of the networks as well as the relationships among communication events. Analyzing a massive and ever-growing graph will quickly overwhelm the computing resources at hand.

Representing various data sets like telephone call records (CDRs), GPS-based movements and social networks in a graph form allows us to discover structural properties in data that are not evident using traditional data mining techniques. Detecting anomalies in graph-based represented data analyzes the relationships among network entities for structural oddities which turns out to be a rich set of information for the data analyst. However, graph-based approaches are known to be prohibitive due to computational constraints, as they typically perform subgraph isomorphisms. To address this issue, most approaches resort to using some type of heuristic to arrive at an approximate solution. In case of real world scenarios, one should consider the structural aspects of dynamic/streaming graph data.

Graph-based Anomaly Detection(GBAD) approach detects anomalies in data represented as graphs. It discovers the best structures (or normative patterns) in an input graph using Minimum Description Length (MDL) heuristic as

$$M(S, G) = DL(G|S) + DL(S) \tag{10.7}$$

where G is the entire graph, S is the substructure, $DL(G|S)$ is the description length of G after compressing it using S, and $DL(S)$ is the description length of the substructure.

Using a beam search, the GBAD method grows patterns one edge at a time, continuously discovering what substructures best compress the description length of the input graph.

For detecting anomalies, the GBAD method uncovers the relational nature of the data instead of assessing statistical deviation of individual data attributes. It looks for those activities that appear to be legitimate transactions/events, but in fact are structurally different. Therefore, the more anomalous substructure is that which is closer to the normative pattern and appears with lower probability.

Based on the above concept, a methodology named as Pattern Learning and Anomaly Detection in Streams (PLADS) detects anomalies in graph data that is represented in data streams. More precisely, PLADS method takes as input a set of N subgraphs either by partitioning a large static graph, or instances/snapshots of a dynamic graph over time. It should be noted that the size of each subgraph (i.e., number of vertices and edges) is not necessarily the same.

According to this method, N subgraphs are processed in parallel to detect top M normative patterns in each. Then, the best P normative patterns are determined among NM possibilities. Each subgraph discovers anomalous substructures based upon P normative patterns, and then the most anomalous substructures across subgraphs are determined. The oldest subgraph is removed to accommodate the subgraph next one in sequence/time and process in a similar manner.

The PLADS method can be evaluated on VAST data set consisting of the activities of 60 employees at an embassy over the month of January 2008 for detecting insider threat by way of leaking information. The corresponding graph representation of this data has 39,331 vertices and 38,052 edges, with two known anomalous substructures. Further investigation of this method can be carried out on stream data set produced using Linked Stream Benchmark (LSBench) data generator.

10.6 Summary

The task of characterizing temporal anomalies/outliers in evolving network/graph data is addressed in this chapter by considering one such framework for exploring their time varying behavior.

- A categorization of the temporal anomalies (outliers) expected in a dynamic network is presented taking into account the evolving characteristics of these outliers over time.
- NetHEART algorithm for detecting different categories of temporal outliers in a dynamic graph data set is described with all the connected theory and computational steps involved.
- Finally, the experimental observations on various benchmark graph data sets are found to acknowledge the presence of various categories of temporal outliers as envisaged.

10.7 Bibliographic Notes

Graph mining research is actively being pursued due its widespread applicability to social network analysis and several other real life scenarios [15, 24]. Among various analysis tasks over network/graph data, anomaly detection gained a lot of practical significance of late [1, 4, 5, 10, 17, 22, 26].

In case of IP-networks, the behavior of individual entities determine the ensemble behavior of the network [23]. Various types of traffic patterns can be noticed in such networks due to different types of network abuse (anomalies) and it is often difficult to detect them, more so in evolving networks [13, 16, 20]. It is important to understand and characterize the outlier/anomaly dynamics from the point of view of various evolving applications. Thus, the study of evolving graphs/networks has been an active research problem [2, 18, 20]. A consolidated material comprising of various techniques for carrying out outlier detection in temporal data was presented in [8].

The 'OddBall' method proposed in [1] discovers different types of node anomalies in graphs by making use of the notion of *egonet* of a node. Similarly, a method for spotting significant anomalous regions in dynamic networks was proposed [16] recently. In another recent effort, a multi-graph clustering method [19] was proposed exploiting the interactions among different dimensions of a given network. Though the multi-graph scenario is conceptually different from multiple instances of an evolving graph, establishing a relationship between these two problem settings can enable leveraging the methods developed for one setting in effectively solving the other. Similarly, other recent efforts [2, 20] dealing with dynamic graphs are aimed at modeling some specific time varying properties of such graphs as per their end application requirements. A substructure-based network behavior anomaly detection approach, named as weighted frequent subgraphs method, was proposed [9] to detect the anomalies present in large-scale IP networks. According to this method, patterns of abnormal traffic behavior are identified using multivariate time series motif association rules mining procedure.

The NetHEART method [21] introduced a framework that defined various categories of temporal outliers present in a dynamic network in order to characterize their evolving dynamics over time. To experiment with such a framework, it is required to have a suitable real life dynamic data set. Accordingly, the DBLP Computer Science Bibliography consisting of millions of bibliographic records bundled in a huge XML file [12] was considered. Similarly, Autonomous Systems (AS) data set [11] representing the graph of routers comprising the Internet was also considered in the experimental evaluation of the NetHEART method.

Various applications in real life deal with the problem of detecting anomalies in dynamics networks. An automatic approach for analyzing streams at multiple levels of temporal granularity was proposed [25]. Similarly, a multi-scale technique [3] was proposed for anomaly detection in time-varying graph data using hierarchically connected distributions. To address the real world scenarios, one should consider the structural aspects of dynamic/streaming graph data [7]. Graph-based Anomaly

Detection(GBAD) approach [6] detects anomalies in data represented as graphs. Subsequently, a methodology named as PLADS was proposed [7] for detecting anomalies in graph data that is represented in data streams. Experimentation of anomaly detection methods dealing with stream data can be carried out using Linked Stream Benchmark (LSBench) data generator [14].

References

1. Akoglu, L., McGlohon, M., Faloutsos, C.: Oddball: spotting anomalies in weighted graphs. In: 14th Pacific-Asia Conference on Advances in Knowledge Discovery and Data Mining (PAKDD), pp. 410–421. Hyderabad, India (2010)
2. Anagnostopoulos, A., Kumar, R., Mahdian, M., Upfal, E., Vandin, F.: Algorithms on evolving graphs. In: ACM ITCS. Cambridge, Massachussets, USA (2012)
3. Bridges, R.A., Collins, J.P., Ferragut, E.M., Laska, J.A., Sullivan, B.D.: Multi-level anomaly detection in time-varying graph data. In: Proceedings of the 2015 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining, ASONAM, pp. 579–583. ACM, Paris, France (2015)
4. Chakrabarti, D.: Autopart: parameter-free graph partitioning and outlier detection. In: PKDD, pp. 112–124 (2004)
5. Chandola, V., Banerjee, A., Kumar, V.: Anomaly detection: a survey. ACM Comput. Surv. **41**(3) (2009)
6. Eberle, W., Holder, L.: Discovering structural anomalies in graph-based data. In: IEEE ICDM Workshops, pp. 393–398 (2007)
7. Eberle, W., Holder, L.: Streaming data analytics for anomalies in graphs. In: 2015 IEEE International Symposium on Technologies for Homeland Security (HST). IEEE, Waltham, USA, pp. 1–6 (2015)
8. Gupta, M., Gao, J., Aggarwal, C.C., Han, J.: Outlier Detection for Temporal Data. Synthesis Lectures on Data Mining and Knowledge Discovery. Morgan & Claypool Publishers (2014)
9. He, W., Hu, G., Zhou, Y.: Large-scale ip network behavior anomaly detection and identification using substructure-based approach and multivariate time series mining. Telecommun. Syst. **50**(1), 1–13 (2012)
10. Kim, M., Leskovec, J.: Latent multi-group memebership graph model. In: 29th International Conference on Machine Learning (ICML). Edinburgh, Scotland, UK (2012)
11. Leskovec, J., Krevl, A.: SNAP Datasets: stanford large network dataset collection. http://snap.stanford.edu/data (2014)
12. Ley, M.: DBLP—some lessons learned. In: PVLDB, vol. 2, issue 2, pp. 1493–1500 (2009)
13. Li, X., Bian, F., Crovella, M., Diot, C., Govindan, R., Iannaccone, G., Lakhina, A.: Detection and identification of network anomalies using sketch subspaces. In: ACM SIGCOMM Conference on Internet Measurement Conference (IMC), Rio de Janeiro, Brazil, pp. 147–152 (2006)
14. Linked stream benchmark data generator. http://code.google.com/p/lsbench
15. Mitra, S., Bagchi, A.: Modeling temporal variation in social network: an evolutionary web graph approach. In: B. Furht (ed.) Handbook of Social Network Technologies, pp. 169–184. Springer (2010)
16. Mongiovi, M., Bogdanov, P., Ranca, R., Singh, A.K., Papalexakis, E.E., Faloutsos, C.: Netspot: spotting significant anomalous regions on dynamic networks. In: SDM. Austin, Texas (2013)
17. Noble, C.C., Cook, D.J.: Graph-based anomaly detection. In: SIGKDD, Washington, DC, USA, pp. 631–636 (2003)
18. Ohnishi, K., Koppen, M., Yoshida, K.: Evolutionary linkage creation between information sources in P2P networks. Evol. Intell. **5**(4), 245–259 (2012)

19. Papalexakis, E.E., Akoglu, L., Ienco, D.: Do more views of a graph help? community detection and clustering in multi-graphs. In: Fusion. Istanbul, Turkey (2013)
20. Rossi, R.A., Neville, J., Gallagher, B., Henderson, K.: Modeling dynamic behavior in large evolving graphs. In: WSDM. Rome, Italy (2013)
21. Suri, N.N.R.R., Murty, M.N., Athithan, G.: Characterizing temporal anomalies in evolving networks. In: 18th Pacific Asia Conference on Knowledge Discovery and Data Mining (PAKDD), Part-I, vol. LNAI 8443, pp. 422–433. Springer, Switzerland, Tainan, Taiwan (2014)
22. Suri, N.N.R.R., Murty, M.N., Athithan, G.: Data mining techniques for outlier detection. In: Q. Zhang, R.S. Segall, M. Cao (eds.) Visual Analytics and Interactive Technologies: Data, Text and Web Mining Applications, chap. 2, pp. 22–38. IGI Global, New York, USA (2011)
23. Thottan, M., Ji, C.: Anomaly detection in IP networks. IEEE Trans. Signal Process. **51**(8), 2191–2204 (2003)
24. Toahchoodee, M., Ray, I., McConnell, R.M.: Using graph theory to represent a spatio-temporal role-based access control model. Int. J. Next Gener. Comput. **1**(2) (2010)
25. Yu, W., Aggarwal, C.C., Ma, S., Wang, H.: On anomalous hotspot discovery in graph streams. In: ICDM (2013)
26. Zainal, A., Maarof, M.A., Shamsuddin, S.M., Abraham, A.: Ensemble of one-class classifiers for network intrusion detection system. In: The Fourth International Conference on Information Assurance and Security, IEEE Computer Society, Napoli, Italy, pp. 180–185 (2008)

Chapter 11
Directions for Further Work

Abstract A summary of the important technical aspects presented in this book is furnished here for ready reference. It includes a few technically promising directions for future work in this field of research with respect to various emerging applications as well as the developments taking place on the computing front.

11.1 Coverage of the Book

Outlier detection is the process of identifying data objects that deviate from the rest of the data. It is an important data mining task due to its significance in several contemporary applications such as fraud detection and anomaly detection in networks. In this exploration of the relevant work, various research issues that determine the exact properties of the outlier detection process are identified. From the practical point of view, one typically encounters the following problem settings regarding outlier detection, for which possible solutions are presented in this book.

1. As per the general understanding, a majority of the methods deal with mainly numerical data. As the data pertaining to many real life applications tend to be categorical in nature, it is important to have efficient techniques for dealing with such data. It is also essential to address the high dimensionality of the data associated with these applications due to its significance in pattern recognition applications.
2. Mining of outliers in data has to deal with uncertainty regarding the membership of such objects to one of the normal groups of objects. Soft computing approaches based on rough sets seem promising to deal with this uncertainty prevailing in data.
3. In this age of connected world, most of the applications deal with data having inherent links. Detecting anomalies (outliers) in network data is an emerging application area with a majority of the methods detecting mainly anomalous nodes and/or anomalous edges. In this context, it is essential to develop a generic method for detecting anomalous arbitrary subgraphs present in the graph representation of network data. An extension to this idea of subgraph anomalies is

to characterize the time varying nature of the anomalies (outliers) in dynamic networks reflecting the changing real life scenarios.

All the above listed problem settings are discussed in detail in this book and a few solutions corresponding to each problem are also presented for a favorable consideration.

11.1.1 Chapter-wise Summary

A chapter-wise summary of the material presented in this book along with some noteworthy points in this regard is presented below, for a quick assimilation of the same.

- An introductory narration of the concept of outliers in data is presented in Chap. 1 to set the direction for the material covered in this book. Then, a theoretical perspective on characterization of outliers with respect to various facets of the same is presented. Subsequently, a formal definition of outliers is provided by drawing some references to the related work. It also emphasized on the practical significance of the notion of outlier in real life applications.
- Based on the above fundamentals of outliers, Chap. 2 defined the problem of detecting outliers in data and also provided the motivation for pursuing this problem as the subject matter of this book. It also highlighted the significant computational aspects associated with this problem indicating the depth of this research field.
- Then, Chap. 3 presented a higher level abstraction of the outlier detection problem along with an exhaustive survey of various methods developed in the past addressing this problem. It then furnished the key research issues identified as part of this work that determine a suitable detection method for a given application context. Each one of these issues is discussed in detail leveraging the available literature in this regard.
- With the view of facilitating detection of outliers in data, Chap. 4 provided the basic mathematical notation followed throughout this book, along with a few computational measures demanded by the outlier detection process. The general procedure followed for preparing data sets for outlier detection is covered here. Similarly, standard measures for assessing the performance of a binary classifier, such as the outlier detector, are also furnished for completeness.
- Then, Chap. 5 addressed the problem of detecting outliers in categorical data given the importance of the type of data attributes. In that way, categorical data still throws a number of computational challenges due to non-availability of necessary proximity measures dealing with such data. Some of the well known algorithms for clustering categorical data are discussed as unsupervised detection of outliers is the natural way to approach. Subsequent discussion included, a ranking-based scheme for detecting outliers based on the frequency counts of various categorical attribute values and also the inherent clustering structure of the data.

- Then, the dimensionality aspect of the data is considered in Chap. 6 by presenting an unsupervised algorithm for feature selection using Mutual Information based formalism. Objectively assessing the relevance of a particular categorical feature towards detection of outliers and also measuring its redundancy in association with other selected features formed the crux of this effort. Through experimentation on benchmark data, it is found that the subset of features thus selected do produce superior performance in terms of outlier detection.
- Addressing the inherent uncertainty associated with clustering of data involving outliers, a soft computing approach based on rough sets is considered in Chap. 7. This specific algorithm performs clustering of categorical data based on rough clustering principles. This algorithm, paired with the ranking-based scheme described above, has indeed produced better results in terms of the number of outliers detected based on the low dimensional categorical data.
- Having looked at some of the basic algorithmic approaches for outlier detection, the focus then moved towards applying these techniques for anomaly detection in graph/network data. In this direction, Chap. 8 introduced the concept of anomalies in graph data and considered an algorithm for mining anomalous subgraphs through community detection using the NMF technique. This algorithm detects anomalous subgraphs by subjecting the detected communities to the power law constraints and thus capable of detecting any arbitrary subgraph having anomalous characteristics. Some of the recent applications of anomaly detection in graph/network data are also furnished to bring out its significance in contemporary applications.
- Subsequently, Chap. 9 explored the case of analyzing dynamic networks and presented a brief on modeling dynamic networks based on various methodologies. It also highlighted community based understanding of the network evolution process. It then briefly discussed some salient applications of dynamic network analysis in different real life scenarios.
- Finally, the problem of characterizing temporal anomalies/ outliers in evolving networks is addressed in Chap. 10. A framework for exploring the time varying behavior of outliers is considered that defined various categories of temporal anomalies (outliers) based on their observed dynamics over time. This framework deals with dynamic networks with multiple instances for anomaly detection. Towards the end, some of the applications dealing with anomaly detection in dynamic networks are presented.

Thus, this book presented some important technological aspects connected with the outlier detection problem, highlighting its relevance in current day applications.

11.2 Ideas to Probe Further

It is prudent to suggest a few interesting directions to pursue further with the aim of progressing the research related to the problems discussed in this book.

- *Detecting Outliers in Categorical Data*: A ranking-based algorithm for detection of outliers in categorical data is presented in Chap. 5 . In this connection, the following ideas may be interesting to look at.
  - Defining a more relevant cluster proximity measure on categorical attributes so as to get improved outlier detection results, and
  - Establishing a formal relationship between the parameters involved in this algorithm, namely α and k, for enhancing its performance through better understanding of the impact of these parameters.
- *Feature Selection for Categorical Data*: An unsupervised algorithm based on Mutual Information measure is discussed in Chap. 6 for feature selection. The following are the directions to explore further in this context.
  - Developing customized measures of feature relevance and redundancy for outlier detection specific to a target application, and
  - Exploring the suitability of some hybrid filter/wrapper methods for feature selection matching with outlier detection requirements.
- *Outlier Detection using Rough Sets*: An algorithm for clustering categorical data based on rough sets principles is discussed in Chap. 7. In this connection, directions for exploring further work on this aspect are as follows.
  - Improving the performance of the clustering algorithm by exploring the roughness parameters in detail, and
  - Formulating an enhanced outlier detection method based on the set theoretic definition of rough sets.
- *Detecting Anomalies in Graph Data*: A graph mining algorithm is discussed in Chap. 8 for detecting anomalous subgraphs through community detection. In this connection, one may like to explore the following ideas.
  - Improving the candidate subgraphs generation procedure by considering applicable refinements as per the latest developments in this area like different types of random walks, and
  - Employing a more directed anomaly detection methodology so as to detect anomalies specific to a target application.
- *Temporal Anomalies in Dynamic Networks*: A framework for exploring the temporal behavior of anomalies in dynamic networks is discussed in Chap. 10. In this connection, the following seem interesting directions to pursue further.
  - Characterizing the temporal behavior of arbitrary anomalous subgraphs present in dynamic networks.
  - Exploiting the graph invariants in defining various categories of temporal outliers.
  - Combining content/attributes with structure is more important in understanding the dynamic behavior of the networks.

11.3 Some Generic Directions

Having seen some specific directions to consider based on the material covered in various chapters of this book, it is essential to look at some of the generic research directions concerning this important data mining task. The exploration given below narrates certain specific ideas to probe further, driven by the gamut of applications and also the current computing technology trends.

11.3.1 Application Driven

Anomaly detection in network data has numerous applications in varied disciplines, starting from administration of computer networks in enterprise environments to managing various public utility services. These applications highlight several interesting directions to pursue the future work connected with outlier detection.

- *Trustworthy computing systems*: As the computing systems and the application software are becoming complex, it gives scope for bugs and vulnerabilities leading to attacks by the adversaries. In this regard, it is possible to build secure and trustworthy systems by analyzing the system logs periodically, with the aim of detecting anomalies whenever there is any deviation noted from the normal system execution.
- *Medical diagnosis*: Application of anomaly detection techniques for identifying malignant tumors from various types of medical images such as mammogram and CT images is an important data mining task. This task is known to be challenging due to the imbalance between negative and positive samples available for the learning algorithm. So, appropriate techniques need to be employed considering this practical problem associated with this field of application.
- *Insider threat analysis*: It is believed that insider threats are generally triggered by certain precipitating events like employee layoffs, major restructuring, closure of the production plant, etc. Such threats can be detected by monitoring the individuals and their interactions with organizational resources. The basic idea is to capture the temporal evolution of user-system interactions so as to detect statistically significant events.
- *Fraud detection in stock trading*: Financial fraud taking place through illegal stock trading can impact the stability of national and international economies. Therefore, detecting such fraud in securities markets assumes significance. Considering the temporal nature of the securities data, methods for detecting contextual outliers in time series data seem promising for fraud detection.
- *Public utility management*: Some of the real life applications involving network data are vehicular traffic control and power distribution over grid networks. In both these cases network anomaly detection plays an important role for efficient management of these public utilities. For example, identifying traffic congestion

in a road network helps the authorities in ensuring smooth vehicular movement and also routing the emergency services in a hassle-free manner.

Applications described above are only indicative of the possibilities in this regard, but not exhaustive.

11.3.2 Technology Driven

The technical advancements related to data mining, parallel computing and social networks also drive the future course of research in outlier detection, so much as the applications do. In this regard, the following are some of the emerging directions to be looked at.

- *Parallel computing strategies for outlier detection*: The recent advancements in micro-processor technology have resulted in the emergence of Graphics Processing Units (GPUs) and multi-core CPUs as the general purpose computing platforms, dealing with computationally intensive tasks. Using GPUs not only provides a high degree of parallelism, but also hardware acceleration for mathematical functions used in outlier detection. In addition, algorithmic implementation using OpenCL framework enables the same code to run on multi-core CPUs in parallel. As a result, efficient outlier detection algorithms can be designed exploiting the compute power offered by these hardware platforms.
- *Outlier detection using big data framework*: The surge in the development of big data techniques has attracted considerable interest towards the detection of anomalies in data streams. By nature, streaming data is generated continuously by thousands of data sources and sent to the collection system simultaneously in small sizes. As a result, the efficiency of the anomaly detection algorithm matters a lot in this scenario. This highlights the need for employing big data analysis techniques, such as the map-reduce framework, for detecting anomalies in an effective manner.
- *Detecting anomalies in signed social networks*: In the context of social networks, it is noticed that the connections among users that exist in on-line social media sites often exhibit a combination of both positive and negative interactions. Positive relationships indicate trust or friendship, while negative relationships indicate distrust or antagonism. According to the literature, on-line social networks with both positive and negative links are known as *signed social networks*. Anomaly detection in signed social networks is yet other interesting application of outlier detection principles.

With these ideas, this exploration of various research problems connected with outlier detection and related algorithmic solutions is concluded.

11.4 Bibliographic Notes

There are a number of real life applications dealing with anomaly detection employing diverse algorithmic strategies. Some of the recent ones in this regard are discussed here to highlight the practical significance of this data mining task in everyday life scenarios.

The anomaly detection technique developed based on deep neural networks, named as DeepLog [1], was designed to learn the model of log patterns from normal system execution, and detect anomalies when log patterns deviate from the learnt model. It constructs work flows using available system log so that on detecting an anomaly, users can diagnose it and perform root cause analysis effectively.

In the field of medical diagnosis, learning algorithms are typically posed with the problem of imbalance in the number of negative and positive training samples. Addressing this issue, an unsupervised model to characterize a malignant tumor in medical images was envisaged using deep neural networks concepts [14].

An unsupervised learning method was proposed to evaluate the potential of insider threats triggered by various precipitating events [9]. This method leverages a bipartite graph model of the user and system interactions with the intent of capturing correlations between the events and the apparent anomalies.

A prediction-based Contextual Anomaly Detection (CAD) method for complex time series was proposed to detect contextual anomalies in stock market data [3]. More recently, an approach for predicting illegal insider trading from large heterogeneous sources of structured and unstructured data was developed using a deep-learning based approach combined with discrete signal processing on the time series data [5].

A data mining technique to detect traffic system-level anomalies in the urban interstate system was proposed in [4]. This technique was developed based on the concepts of symbolic dynamics to capture the system characteristics. It is capable of extracting salient time series features and discovering spatial and temporal patterns for a traffic system. Similarly, an approach for detecting temporal outliers in vehicle traffic data was presented in [7]. In other recent effort, a two-stage method consisting of a Collaborative Path Inference (CPI) model and a Road Anomaly Test (RAT) model was developed for detecting road traffic anomalies [13].

A graph-based anomaly detection approach for detecting potential irregularities in electrical consumption was developed by modeling the smart grid as a network [8]. The work presented in [15] was aimed at developing an online anomaly detection solution and diagnosis tools for time series data at scale. This solution was integrated into Xpert system intelligence platform of Microsoft Store that monitors millions of cores in services and billions of Windows devices.

Given the plethora of applications of anomaly detection, it is equally important to invest research effort towards developing novel methodologies catering to various computational challenges of the evolving applications. In this regard, the following algorithmic trends shed light on the state of the art connected with anomaly detection techniques.

Regarding the use of sophisticated hardware platforms enabling parallel computing, a morphological algorithm for anomaly detection in hyper-spectral images with a GPU-based implementation was proposed in [11]. Similarly, a map-reduce based big data analysis methodology for network anomaly detection over Hadoop platform was presented in [2]. A comprehensive system was developed based on big data analysis to construct cyber security correlated events with feature selection to anticipate behavior based on various sensors [12]. More recently, a methodology for representation and learning with large-scale graph data was furnished in [6]. Similarly, a ranking mechanism for maximizing spread, trust in signed social networks was proposed [10] based on the information diffusion principle.

References

1. Du, M., Li, F., Zheng, G., Srikumar, V.: Deeplog: anomaly detection and diagnosis from system logs through deep learning. In: CCS, pp. 1285–1298. Dallas, TX, USA (2017)
2. Fontugne, R., Mazel, J., Fukuda, K.: Hashdoop: a mapreduce framework for network anomaly detection. In: IEEE INFOCOM Workshop on Security and Privacy in Big Data, pp. 494–499. IEEE (2014)
3. Golmohammadi, K., Zaiane, O.R.: Time series contextual anomaly detection for detecting market manipulation in stock market. In: DSAA. IEEE (2015)
4. Huang, T., Liu, C., Sharma, A., Sarkar, S.: Traffic system anomaly detection using spatiotemporal pattern networks. Int. J. Progn. Health Manag. **3** (2018)
5. Islam, S.R., Ghafoor, S.K., Eberle, W.: Mining illegal insider trading of stocks: a proactive approach (2018)
6. Leskovec, J.: Large-scale graph representation learning. In: IEEE International Conference on Big Data, p. 4. Boston, MA, USA (2017)
7. Li, X., Li, Z., Han, J., Lee, J.: Temporal outlier detection in vehicle traffic data. In: Proceedings of the 25th International Conference on Data Engineering ICDE, pp. 1319–1322. Shanghai, China (2009)
8. Mookiah, L., Dean, C., Eberle, W.: Graph-based anomaly detection on smart grid data. In: Proceedings of the Thirtieth International Florida Artificial Intelligence Research Society Conference FLAIRS, pp. 306–311 (2017)
9. Moriano, P., Rich, J.P.S., Camp, L.J.: Insider threat event detection in user-system interactions. In: MIST. Dallas, TX, USA (2017)
10. Narayanam, R., Suri, N.N.R.R., Garg, V.K., Murty, M.N.: Ranking mechanisms for maximizing spread, trust in signed social networks. In: ECML-PKDD Workshop: CoLISD (2012)
11. Paz, A., Plaza, A.: A new morphological anomaly detection algorithm for hyperspectral images and its GPU implementation. In: Huang, B., Plaza, A.J., Thiebaut, C. (eds.) Proceedings of SPIE, vol. 8157 (2011)
12. Razaq, A., Tianfield, H., Barrie, P.: A big data analytics based approach to anomaly detection. In: 3rd International Conference on Big Data Computing. Applications and Technologies BDCAT, pp. 187–193. ACM, Shanghai, China (2016)
13. Wang, H., Wen, H., Yi, F., Zhu, H., Sun, L.: Road traffic anomaly detection via collaborative path inference from GPS snippets. Sensors **17**(550), 1–21 (2017)
14. Wei, Q., Ren, Y., Hou, R., Shi, B., Lo, J.Y., Carin, L.: Anomaly detection for medical images based on a one-class classification. In: Proceedings of SPIE Medical Imaging. Houston, Texas, US (2018)
15. Zhang, J., Wydrowski, R., Wang, Z., Arrabolu, S.S., Kanazawa, K., Gudalewicz, L., Gao, H., Batoukov, R., Aghajanyan, S., Tran, K.: Mbius: online anomaly detection and diagnosis. In: KDD. El London, UK (2018)

Appendix A
Online Resources for Outlier Detection

A.1 Open Source Tools/Packages

There are a number of data mining and machine learning libraries made available on the Internet by various working groups of professionals, for academic and non-profit usage. Typically, these libraries vary in terms of the following aspects.

- Coverage of algorithms (data analysis, visualization, predictive models, etc.)
- Implementation language (such as C, C++, Java, Python, etc.)
- Target platform for usage (like Windows, Linux, Mac OS, etc.)
- Mode of usage (such as online browser-based use, download and install, etc.)

Free and open source software tools for data mining are being developed for the past several years. The common objective of these tools is to facilitate the data analysis process in an effective manner and to offer all interested researchers a free alternative to the commercial solutions. This is typically done by employing integrated development environments, and implementation using standard programming languages.

As per the open source philosophy, the source codes of these libraries are also generally available for researchers working in this area. The idea behind this philosophy is that one can extend the functionality of these tools by incorporating algorithms corresponding to recent innovation and maintain the tools from time-to-time. Free and open source software are typically distributed following the GNU general public licensing method.

The objective here is to make young professionals and researchers planning to work in the field of data mining and outlier detection, get familiarized with the availability of various online resources such as different implementations of the machine learning algorithms, benchmark data sets to experiment, research forums to interact, etc. Typically, such resources are required for carrying out advanced research related to data mining. Details on a few of the well known data mining tools and libraries are furnished below for a quick reference. It is important to note that details regarding only a subset of the tools are presented here for brevity.

1. *Waikato Environment for Knowledge Analysis (WEKA)*: This is a collection of various machine learning algorithms for data analysis and predictive modeling, developed using Java programming language at the University of Waikato, New Zealand. This tool comprises necessary graphical user interfaces for easy and convenient use of the analysis functionality. The tool distribution supports multiple operating platforms for its use enabling cross-platform deployment.
   WEKA suite consists of the following functional modules offering a variety of analysis functionality on data. More details regarding this tool can be obtained from: 'https://www.cs.waikato.ac.nz/ml/weka'.
   - Explorer
   - Experimenter
   - KnowledgeFlow
   - Simple CLI
2. *Konstanz Information Miner (KNIME)*: This tool was developed at University of Konstanz as a proprietary product. The objective was to create a modular, highly scalable and open data processing platform for easy integration of different data loading, processing, transformation, analysis and visual exploration modules. This tool has been widely used in pharmaceutical research, customer data analysis, business intelligence and financial analysis.
   KNIME allows users to visually create data flows and selectively execute some of the analysis steps in the sequence. It is developed using Java language based on Eclipse IDE and allows plugins to provide additional functionality. The basic package includes various commonly used methods of statistics, data mining, analysis and text analytics. This tool supports big data analysis framework and also works with Apache Spark library. For more details in this regard, one may refer to: 'https://www.knime.com/knime-software/knime-analytics-platform'.
3. *Environment for DeveLoping KDD-Applications Supported by Index-Structures (ELKI)*: The initial version of this tool distribution contained several algorithms to perform cluster analysis, anomaly detection, along with other data mining tasks. As its name indicates, some spatial index structures are included in this tool with its focus primarily on subspace clustering and correlation clustering methods.
   It was developed at the Ludwig Maximilian University of Munich, Germany, based on various Java interfaces for use in teaching and research purposes. The visualization interface of ELKI uses a scalable graphics library and also supports loss less export of user interface displays to PostScript and PDF file formats. For the purpose of detecting outliers in data, this particular tool offers various algorithms based on different detection strategies such as distance-based outliers, LOF-based local outliers, and many others. Therefore, it can be used as a reference platform for exploring and understanding the research issues related to this problem space. Further details in this regard can be accessed at: 'https://elki-project.github.io'.

4. *Orange Software*: This is a component-based visual programming software package for data visualization, machine learning, and data analysis. The earlier versions of this software include core components written in C++ language and the wrappers using Python scripts. The advanced version makes use of various Python-based scientific computing libraries such as *numpy*, *scipy* and *scikit-learn* to accomplish the intended functionality. The graphical user interface is designed based on the Qt framework, for supporting portability across platforms.
   Orange software basically provides a visual programming front-end for explorative data analysis and interactive data visualization. Interactive analytics helps in better understanding of the analysis process with insights on the inherent structure of the data. This software can be installed on all major computing platforms supporting Python based development. More information regarding this tool can found at: 'https://orange.biolab.si'.
5. *PyTorch Library*: This is a python based open source machine learning library used as a computing platform for developing deep learning based applications. The workflow with this tool is similar to that of the python-based scientific computing library *numpy*. It basically supports two key functionalities

   - Tensor computations with multi-GPU support
   - Computations relating to deep neural networks

   PyTorch provides a framework for building computational graphs allowing the users to perform computations on graph components before the graph is built completely, and even change them during runtime. For more details on this library, one may refer to: 'https://pytorch.org/'.

A.2 Benchmark Data Sets

Benchmark data sets facilitate a systematic evaluation of the machine learning algorithms based on some predefined application settings. These data sets can also be used for establishing a comparative view of various algorithms designed to accomplish a common computational task. Thus, use of appropriate benchmark data sets enables the research community for advancement of the data mining technology in a quantified manner. More specifically, the case of dealing with the outlier detection problem needs special attention in terms of identifying suitable data sets due to the required imbalance in class distribution. To this effect, certain pre-processing steps are performed on the original data, making it ready for outlier detection.

1. *UCI ML repository*: This is a collection of data sets popularly used by the machine learning community for empirical analysis of their algorithms. This repository has divided the data sets under different classifications for easy selection and usage, as per the details given below.

   - Learning task: classification, regression, clustering, etc.
   - Type of attribute: categorical, numerical, mixed.

- Data generation method: uni-variate, multivariate, sequential, time-series, etc.
- Application area: life sciences, physical sciences, computer science and engineering, social sciences, business, game data, etc.

The above data are available at: 'https://archive.ics.uci.edu/ml/datasets.html'.

2. *Stanford Large Network Dataset Collection*: This collection consists of many network data sets divided into 17 broad categories based on the nature of the data captured. The main categories include social networks, networks with ground-truth communities, communication networks, citation networks, collaboration networks, web graphs, autonomous systems, signed networks, temporal networks, etc. These data sets are made available as part of the Stanford Network Analysis Platform (SNAP) platform, which is a general purpose network analysis and graph mining library. These data sets can be accessed at the web page: 'http://snap.stanford.edu/data/index.html'.
3. *Outlier Detection Data Sets (ODDS)*: This collection provides open access to a number of data sets meant for outlier detection, with ground truth as per its availability. These data sets are divided into the following groups based on the specific data they contain and for enabling precise usage.

- Multi-dimensional point data sets
- Time series graph data sets for event detection
- Time series point data sets (Multivariate/Univariate)
- Adversarial/Attack scenario and security data sets
- Crowded scene video data for anomaly detection

The above data are available online at the URL: 'http://odds.cs.stonybrook.edu/'.

Glossary

Anomaly detection The process of identifying data objects that do not conform to the expected behavior of a given collection of data.

Autonomous systems A collection of routers whose prefixes and routing policies are under common administrative control, and they communicate using Border Gateway Protocol.

Bagging Bootstrap aggregating (bagging) is an ensemble method designed to improve the accuracy of the learning algorithms used for classification and regression.

Categorical variable A categorical variable (nominal variable) is one that doesn't have intrinsic ordering of its categories.

Centrality measure In graph theory and network analysis, a centrality measure identifies the most important vertices in a topological sense.

Cluster analysis An exploratory analysis tool that divides the data objects into various groups such that the degree of association between two objects is maximal within a group and minimal otherwise.

Community detection The process of dividing a network into groups of nodes with dense connections internally and sparser connections between groups.

Confusion matrix It contains information about the actual and the predicted classifications done by a classifier.

Curse of dimensionality The phenomena that arise when performing data analysis in high-dimensional spaces (typically hundreds of dimensions).

Data mining The process of identifying valid, novel, potentially useful and ultimately understandable patterns in large data sets.

Decision attribute In a decision tree, it is part of a node, so it performs as a consequent in the decision rules on the paths down the tree to the leaf node.

Dense subgraph A component of the graph with high edge density in comparison to all other sub-graphs of the graph.

Discernibility matrix A matrix in which the classes are indicies and the condition attributes which can be used to discern between the classes in the corresponding row and column are inserted.

Egonet The ego-network of a node is the induced subgraph of its 1-step neighbors in the given graph.

Eigenvalues A special set of scalars associated with a linear system of equations that are sometimes also known as characteristic values, or latent roots.

Ensemble methods The methods that use multiple models to obtain better predictive performance over any of the constituent models.

Entropy The entropy of a random variable is a measure of the uncertainty associated with that random variable.

Evolving networks Networks that change as a function of time, either by adding or removing nodes or links over time.

Example-based outlier mining Given a set of outlier examples, find additional outliers from the data that exhibit similar outlier characteristics.

Feature selection The process of identifying and removing irrelevant and redundant attributes from data that do not contribute to the accuracy of a predictive model or may in fact decrease the accuracy of the model.

Frequent subgraph Discovering subgraphs that occur frequently over the entire set of graphs.

Fuzzy set Any set that allows its members to have different grades of membership (membership function) in the interval [0,1].

Game theory The study of mathematical models of strategic interaction between rational decision-makers.

Genetic algorithm It is a heuristic search and optimization technique inspired by natural evolution.

Granular computing It is a nature inspired computing paradigm that performs computation and operations on information granules.

Graph mining Extracting patterns (subgraphs) of interest from graphs, that describe the underlying data.

Hamming distance Hamming distance between two strings of equal length is the number of positions at which the corresponding characters differ.

High dimensional data Data with large number of features, attributes or characteristics that lead to curse of dimensionality.

Imbalanced data A data set in which one of the two classes has more samples than the other, resulting in skewed distribution of data.

Information theory The mathematical study of the coding of information in the form of sequences of symbols, impulses, etc.

Joint probability A statistical measure that calculates the likelihood of two events occurring together and at the same point in time.

Knowledge discovery An interdisciplinary area focusing on methodologies for extracting useful knowledge from data.

Labeled data A group of data samples that have been tagged with one or more labels.

Likelihood Typically refers to events that have a reasonable probability of occurring, but are not definite or may be influenced by factors not yet observed or measured.

Machine learning This is an application of Artificial Intelligence that provides systems the ability to automatically learn and improve from experience without being explicitly programmed.

Marginal probability The probability of any single event occurring independent of other events.

Matrix algebra The process of performing common algebraic operations (such as addition, subtraction, multiplication, determining rank, etc.) on matrix variables.

Modularity It measures the strength of division of a network into various modules/communities, such that there are dense connections between the nodes within communities but sparse connections across communities.

Mutual information According to information theory, the mutual information of two random variables is a measure of the mutual dependence between them.

Nearest neighbor search A form of proximity search, trying to find the object in a given data set that is closest to a given object.

Network regularization In the field of machine learning, regularization is a process of introducing additional information during the learning phase in order to prevent over fitting.

Non-negative matrix factorization A technique for obtaining low rank representation of matrices with non-negative or positive elements.

Outlier An observation which deviates so much from the other observations as to arouse suspicion that it was generated by a different mechanism.

Outlier detection The process of finding out objects that are considerably dissimilar, exceptional and inconsistent with respect to the majority objects in a data set.

Outlying subspaces The outlying subspace of a data object is the subspace in which the object displays outlier characteristics.

Overlap measure For two multivariate categorical data points, the overlap measure indicates the similarity between them by taking the number of attributes in which they match.

Proximity measure This measures how alike objects are to one another (object similarity) or how unlike they are (object dissimilarity).

Random projection A technique that allows one to substantially reduce the dimensionality of a problem while still retaining a significant degree of the structure of the data set.

ROC curve A receiver operating characteristic (ROC) curve is a 2-dimensional plot showing the performance of a classification model at all classification thresholds.

Rough clustering In contrast to conventional data clustering, the definite membership of some objects cannot be determined when performing rough clustering.

Rough set An approximation of a crisp set in terms of a pair of sets representing the lower and the upper approximation of the original set.

Route views A mechanism that allows Internet users to view global BGP routing information.

Scale-free networks Those information networks that satisfy a power law degree distribution.

Semi-supervised learning A machine learning paradigm that uses a large amount of unlabeled data, together with a small amount of labeled data, to build better classifiers.

Sliding window The basis for how we can turn any time series data set into a supervised learning problem.

Social network It is a web service that allows people with similar interests to come together and share information, photos and videos.

Soft computing A computing method that deals with approximate models and gives solutions to complex real-life problems. It is tolerant of imprecision, uncertainty, partial truth, and approximations.

Sparse data A data set in which a relatively high percentage of the attribute values are unspecified.

Snapshot graph The snapshots of an evolving graph taken periodically form a sequence of snapshot graphs.

Stochastic network A deterministic network that is modeled randomly due to its complexity.

Streaming data The data that is generated continuously by thousands of data sources and sent to the collection system simultaneously, in small sizes.

Subspace The space spanned by a sub-set of features/attributes of a data set.

Traffic patterns Typical traffic patterns over a communication network include traffic volume distribution, application usage, and application popularity.

Unsupervised learning A machine learning paradigm that draws inferences from data sets without class labels.

Volatile graph A stream of duration-stamped edges with potentially infinite number of nodes, and edges may appear and disappear over time.

Web service It is a standardized medium to propagate communication between the client and server applications on the World Wide Web.

Index

N. N. R Ranga Suri et al., *Outlier Detection: Techniques and Applications*,
Intelligent Systems Reference Library 155,
https://doi.org/10.1007/978-3-030-05127-3