Annual Progress Seminar - I Report
on

Social Network Analytics: Determining latent characteristics in data

Submitted
in partial fulfillment of
the requirements of the degree of

Doctor of Philosophy (Computer Engineering)

by

Pranav Nerurkar
(169070001)
2016 - 2017

Under Guidance Of

Dr. S. G. Bhirud (Guide)
Dr. M. M. Chandane (Co - Guide)

Department of Computer Engineering & Information Technology
Veermata Jijabai Technological Institute, Mumbai - 400019
(Autonomous Institute Affiliated to University of Mumbai)
2016 - 2017

CERTIFICATE

This is to certify that **Mr. Pranav. A. Nerurkar (169070001)** , a student of **Doctor of Philosophy (Computer Engineering)**, has completed the Annual Progress Seminar held in September 2017 for the research work entitled, **Social Network Analytics: Determining latent characteristics in data** to our satisfaction.

Dr. S. G. Bhirud
Ph. D. Supervisor (CMPN)

Dr. V. B. Nikam
Head, Department of Computer Engineering
& Information Technology

Date:

Place:

CERTIFICATE

The Annual Progress Seminar Report for the research work entitled , **Social Network Analytics: Determining latent characteristics in data** submitted by **Mr. Pranav. A. Nerurkar (169070001)**, is found to be satisfactory and is approved for the degree of **Doctor of Philosophy (Computer Engineering)** from Veermata Jijabai Technological Institute affiliated to University of Mumbai.

Supervisor:	Co-Supervisor:	Examiner 1:	Examiner 2:
Dr.S.G. Bhirud	Dr.M.M. Chandane	_____	_____
Guide	Co-Guide	_____	_____

Declaration of the Student

I declare that this written submission represents my ideas in my own words and where others' ideas or words have been included, I have adequately cited and referenced the original sources. I also declare that I have adhered to all principles

of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea / data / fact / source in my submission. I understand that any

violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

Pranav Nerurkar
(Roll no : 169070001) Date: _____

Acknowledgements

My sincere thanks goes to Dr. Dhiren Patel, Director of Veermata Jijabai Technological Institute, Mumbai who provided me an opportunity to work as a research scholar and gave me access to the laboratory and research facilities which helped me to conduct this research.

I would like to express my sincere gratitude to my advisor Prof. Dr. S. G. Bhirud and my Co-guide Dr. M. M. Chandane for their continuous support of my Ph.D study and related research, for their patience, motivation, and immense knowledge. Their guidance helped me in all the time of research and writing of this report.

I thank my Head of the Department Dr. V. B. Nikam, for extending his invaluable support to me. I would also thank my institute and the department's faculty members without whom this research work would have been a distant reality.

I also extend my heartfelt thanks to my family and well wishers.

Roll No: 169070001 Pranav. A. Nerurkar
Date:

Abstract

Networks have been studied extensively in the last decade with the purpose of analysing interactions between the participating entities (nodes) and determining the important structural patterns in such interactions. Current research trends have focused on particular category of networks known as "Online social networks". Examples of such networks include social networking sites such as (Twitter, Stackoverflow, Facebook , LinkedIn), Multimedia networks (Flickr), Scientific Collaboration networks (CiteSeer), Sensor networks etc. Networks tend to proliferate more rapidly in the Internet Era because they are no longer constrained by the geographical limitations of a conventional social network. These massive online social networks exhibit phenomena such as diffusion, contagion, influence which are the consequence of interactions seen between the entities in the network. "Latent (hidden) characteristics" is the umbrella term assigned for factors that have an influence on the activities in a social network. The focus of the present investigation is to understand the cause - effect relationship between the latent characteristics and the social network. The analysis to be carried out as a part of this study focuses on the unique issues which arise in the context of the interplay between the structural and data-centric aspects of the network. These issues are related to understanding of the statistical properties of the network, development of techniques for community discovery, clustering of data, node classification, social influence analysis, expert location and link prediction.

This thesis presents an overview of the field of Social network analysis along with the important research areas in this field. The focus of the investigation is on a subset of the research areas mentioned in Chapter 1. The issues that presented in this thesis are in field of social influence analysis, community discovery and clustering of data from a network. The mathematical basis for the various approaches that are popular in the literature are presented. Secondly, simulation of the algorithms is presented on data to highlight their strengths and weaknesses of individual approaches. Based on the simulated results and surveyed literature, conclusions are inferred and future work is proposed.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Networks are used to graphically represent relationship or structure in many complex systems which are seen in natural, technological or social settings. Understanding the process of formation of such networks or studying why certain systems exhibit a particular structure can provide insights into various phenomena that are commonly seen in real life communities such as diffusion, contagion and influence. This have given birth to the scientific study of networks which became a multi disciplinary field spanning physics, computer science as well as social sciences [1] [14]. This is because any system in these fields can be represented in the form of a Network and the theories for studying such Networks have been at the intersection of all these fields.

A Network or graph consists of nodes which are the vertices of the graph and edges. Nodes represent actors or entities and edges depict the relationship between them. An edge connects typically two nodes but if three or more nodes are connected then it is known as a hyperedge and such graphs are called hypergraphs. For example, in a scientific collaboration network, the nodes are the researchers and edges between them denote those researchers that are working on the same research topic. Another common network which is widely used is the 'Internet' which in itself is a massive web graph consisting of web pages (nodes) and hyperlinks (edges). Similarly, any system can be represented as a network e.g citation networks, social networks, networks of protein communities etc [26].

As the Digital transformation of the society gathers pace, there is an increase in the proliferation of online networks as barriers of storage and computation power are lowered. Earlier network science was based on the studies performed on conventional networks which were offline. With the advent of the Internet this has changed and networks started having an online presence. Due to this researchers have received a chance to relook at the conventional social network literature that was developed in the pre-internet age and to validate it in the context of online networks [1] [48]. A classic example of this was the validation of phenomena such as "six-degrees of separation", "small world theory", "preferential attachment" and "shrinking diameters" which were

developed based on studies on offline networks. Thus, the availability of massive data has now given impetus to the verification of the older theories which were based on conventional social networks and at the same time provided a path for a scientific and statistically valid studies in the field of Networks.

1.1 Online Social Networks: Research Issues

The field of online social networks has seen a rapid revival since the first decade of the 21st Century when massive online social networks were created and the barriers such as computation power to analyse them were reduced. A key aspect of many of the online social networks is that they are rich in data, and provide significant opportunities from the perspective of knowledge discovery and data mining. The popular research issues in the domain of Online Networks are listed below:

- **Social Influence Analysis:** Social Influence Analysis answers questions related to modeling the nature of influence of certain actors on the remaining network and its spread. Ranking actors on the basis of their influence is also a part of this domain [1].
- **Community Discovery:** The community detection problem is closely related to that of clustering, and it attempts to determine autonomous regions of the network, which are dense in terms of the linkage behavior [1] [48].
- **Data Clustering:** Online social networks are rich in data as massive amounts of content is shared every second on them. Data Clustering is exploratory data analysis which provides techniques of effective summarization for efficient retrieval and visualization [1] [48] [6] [13].
- **Node Classification:** It is the field related to accurately learning the labels of a node on the basis of latent information present in the attributes or structural properties of the social network [1] [48].
- **Expert Location:** Social Networks can also be used as tools to quantify the area of expertise of a person in a particular domain by learning from the activities performed by him within a context [1] [23].
- **Link Prediction:** Using the links in the social network for deriving interesting information is a key concept in fields like social influence analysis, community discovery etc. But as links are dynamic in nature, to determine and predict future links is an important question [1] [13].

The rest of the chapters provide details of the investigations carried out on a subset of the research topics enumerated above.

Chapter 2

Modeling influence on a Social Network using Interaction Characteristics

2.1 Introduction

The importance of social networks has been realized since the advent of Facebook in 2004 as well as other social networks such as Twitter, Flickr and Instagram. When a user posts a message on social media, other users in the network respond to the content by performing certain actions which create cascading effects within the network and is visible in the form of further activity. The fact that a message from one user prompted certain reactions from other users is an indication of the influence of one user over others in the network. This can be used to enhance the visibility of a product, viral marketing, and spreading awareness or even during disasters to alert citizens, or spreading hatred or rumors.

It is due to this popularity social media analytics has attracted a lot of attention from the research communities in machine learning [1] [18] [48], preference learning [56], recommendation systems, data mining [16], information retrieval [54] etc. Social networks belong to the category of networks called strategic network models where the relation between nodes is determined by the choice of the members involved, not by a random rule. This means that the characteristics of nodes are important in determining whether links shall be formed. This also decides whether the information that is diffused can reach more nodes of the network. Chen *et. al* and Subbian *et. al* have proposed methods for finding influential nodes which use the process of information diffusion or social capital to discover the nodes with high diffusion centrality [7] [47]. Holander *et. al* have argued that use of Eigenvector centrality as a measure is insufficient in deciding the and ranking influential nodes in online networks [24] [42].

Influence maximization problem is a NP hard problem of identifying the subset

of nodes in a social network that would maximize the coverage of information in the graph through diffusion. However the problem of assigning influence scores to every node in the graph is different compared to Influence Maximization since the former is a measurement problem, while the latter is a subset selection problem and both have different objectives [7] [47]. In influence ranking, Klout rank proposed by Rao *et al.* has achieved significant breakthrough in calculating the current influence or rank of the user in the social network by analyzing the interaction characteristics generated by the user on a social network in the preceding 90 days [42]. The mathematical model implemented to calculate the Klout rank uses a feature vector of 3600 attributes. The parameters of the model have been tuned by analyzing 45 billion interactions of various users collected from online networks. Machine learning techniques applied on data to create a predictive model has been investigated by Alexy *et al.* [22] [40]. The model uses interaction characteristics such as number of mentions over time to create a PageRank like score for ranking users by their influence. Behnam *et al.* [29] modeled influence using metrics such as number of followers and their ratio of affection to a particular topic. However previous literature ignored making up decision rules and understanding complex relationships between features that could lead to development of an objective criteria for calculating and comparing influence.

In this paper, attributes that are accumulated by the behavior of a person on the social network are used. The key assumption is that these attributes have a relationship with the influence of the person and can be used to create a feature vector to model the influence. The online characteristics that can be measured by information extracted from an online social network are impressions, clicks, likes, comments, re-shares, followers. Such attributes are difficult to emulate or artificially enhance and hence are appropriate indicators for developing an objective criteria for assessing the influence.

We have conducted experiments on the real world dataset from twitter to validate the effectiveness of the proposed approach.

The highlights of the paper are summarized as follows :

1. A machine learning approach to study the comparison of influencers in the social network using various algorithms and their effectiveness on the data.
2. Analysis of various features and their use in training and building a model.
3. Use of feature selection, feature ranking techniques and their effect on the model and the predictions. Thus obtaining a smaller set of predictors that decide influence within a network.
4. Validation of the model and comparisons to establish which of the techniques has best results.

Section II presents the related work in the field and the introduction to machine learning models and other techniques implemented in the paper. Section III formulates the problem, provides the objective function and then proposed approach. Section IV provides information about the data set, the methodology, experimental design and results. The conclusion is given in Section V.

2.2 Related Work

2.2.1 Twitterrank: finding topic-sensitive influential twitterers

The TwitterRank algorithm was implemented for studying people who are the most influential in the social network. The algorithm is based on the hypothesis that the measure of influence of a person is related to both the topical similarity between twitterers and the link structure. The authors argue that they could find evidence to support the theory which states that more followers a person has in a social network the more is the influence i.e. in-degree of a node is proportional to its influence in the network. The algorithm works on the assumption that there exists homophily within twitter i.e. same interests decide the relationship between the users. However no correlation is presented in the form of a degree distribution to prove existence of homophily [44] [50].

2.2.2 Finding influencers using social capital

The concept of social capital is proposed by the authors. According to them "Social capital" is computed by means of bonding and bridging capital. The bonding capital is defined as the ability to calibrate similar people against each other. The bridging capital is defined as the ability to connect diverse people. A bridging node is like an interdisciplinary researcher who connects different communities of researchers and a bonding node is a good researcher in a domain. It is argued that such nodes create a value throughout the network and this value is shared by them. The hypothesis is that this value is proportional to the influence of the node amongst other nodes it is connected to in the network. The social capital and its distribution amongst the nodes is based on the concept of distance based utility which is seen in strategic network formation process. The distance-based utility concept assumes that all players (nodes) have a utility in the network that is alike. Furthermore it takes into account only the benefits that a node in the network derives from other nodes to which it has indirect links. For the purpose of computing the value derived through the indirect links only links having minimum path length from source to destination are considered. These two features are considered as drawbacks of the distance-based utility in general and the same have been seen in the social capital based approach [47].

2.2.3 Geo-social Influence Spanning Maximization

The Influence maximization problem has been modified to include the physical location of the social users in an attempt to solve the NP- Hard problem of identifying a small set

of users that can influence the maximum number of users in a network. The technique measures influence as a factor of diffusion potential of a node. It is computationally expensive and may yield results that cannot guarantee global optimal solution due to the vast search space. The technique proposed uses greedy algorithm to identify the top "k" seeds that can propagate the influence across the network. The technique also proposes a hybrid indexing structure OIR*- tree which combines the features of an ordered influential user list and R*-tree. The indexing structure doesnt consider the changing in the influence with time. The space complexity of this approach is higher as the influential nodes have to be stored. This approach is not useful for measuring influence of the node in a network [8].

2.2.4 Mining Time-Dependent Influential Users in Facebook Fans Group

"Klout rank" is a popular method of using machine learning techniques to calculate the influence of a user in a network. The klout score is computed based on the statistics calculated about the user from various social platforms and a score is assigned to them. The technique however doesnt consider that influence is time dependent. This means that a user may not have a high average klout score but could be influential at a particular time period. Klout also works on a black box model that doesnt explain what can affect the influence of users in a network. Bias/variance graphs are not presented to indicate whether having 3600 features can lead to overfitting [27] [32].

2.3 Mathematical Model

An online social network is represented as a graph G = (V, E) with nodes (V) represents the number of users and edges (E) denote friendship between two users. In most of the social networks the users send requests to other users to form a relationship. The acceptance of this request is then a undirected link. Usually the friendhsip links in a social networking graph can be both undirected (Facebook) as well as directed (Twitter). The nodes also have properties that are due to the membership of this network. These properties are derived from membership of the network and are varied by the influence of his behavior on the network and the networks influence on his behavior i.e. co-evolution.

The measurement problem of calculating the influence shall become a problem of supervised learning. Supervised learning algorithms are of many types and there is no single algorithm that can work on any given supervised learning problems. Neural networks perform well if the task involves identifying complex set of dependencies and identifying these dependencies. However the best configuration of a neural network has to be determined experimentally.

2.3.1 Calculation of Influence Score

A feature vector x_i is constructed for each user from the set of features extracted for each user from Twitter as shown in Table 1. Then a Given a set of N training examples of the form (x_1, y_1), , (x_n, y_n) such that x_i is the feature vector of the i^{th} example and y_i is its class label. The algorithm creates a function $g : X \to Y$ where X is the input space and Y is the output space.

Table 2.1: Feature vector

Sr. No	**Feature list**	**Sr. No**	**Feature list**
1	Follower count	5	Following count
2	Listed count	6	mentions received
3	retweets received	7	mentions received
4	retweets sent	8	posts

2.3.2 Model Representation

2.3.2.1 Artificial Neural Networks

Artificial neural networks are composed of input layers, hidden layers and output layers. The number of hidden layers ideally should be more as they can compute more non-linear relationships between the inputs. However, it is important to avoid over-fitting while deciding the number of hidden layers and the number of hidden units. The input feature matrix X and the weight matrix of the W_{21} (θ_1) and the result are given to the sigmoid function to decide which neurons of the hidden layer 1 i.e. (a_1) are activated. The output of this hidden layer is the multiplied with the second weight matrix W_{31} (θ_2) and so on till the output layer. The back-propagation algorithm is then used to obtain the adjusted weights. The cost function of the network for the classification problem is provided in Eqn. (1) and the penalty term (regularization) is used to it for preventing over-fitting is provided in Eqn.(2).

2.3.2.2 Cost Function - Artificial Neural Networks

The cost function is that of the standard regularized logistic regression which is generalized for k outputs instead of one output. For our problem the output is a single value of 0/1 depending upon whose influence in a network is more. The cost function shall be minimized using the back propagation algorithm to get the ideal value of the weights.

$$J(\Theta) = \frac{a}{b} \sum_{i=1}^{k} \sum_{k=1}^{k} *[-y_k^i * log((h_\Theta(x^i))_k) - (1 - y_k^{(i)}) * log((h_\Theta(x^i))_k)] \tag{2.1}$$

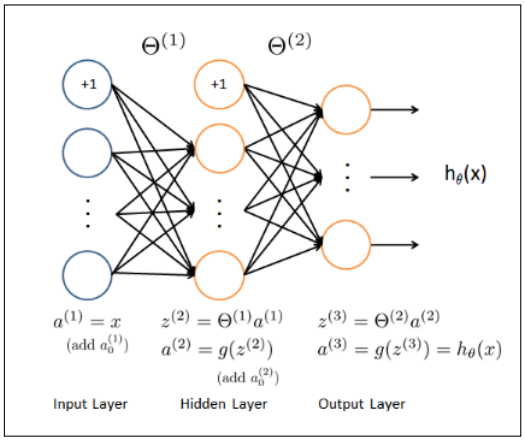


Figure 2.1: Neural Network Architecture with Backpropagation

$$Penalty = \frac{\lambda}{m} * [\sum_{j=1}^{J1}\sum_{k=1}^{K1} *(\Theta_{(j,k))}^{(1))})^2 + \sum_{j=1}^{J2}\sum_{k=1}^{K2} *(\Theta_{(j,k))}^{(2))})^2] \tag{2.2}$$

2.3.2.3 Gradient boosted trees

Gradient boosting is a technique that combines several weak learners into a strong learner using multiple iterations. Gradient boosting is applied with decision trees of fixed size as the first level learners. The decision tree partitions the input space into disjoint regions and predicts a constant value in each region. The General idea for a gradient boosted tree is to fit a model to the data i.e F_1(x) = y . Fit the model to the residuals denoted by $h_{(1)(x)}$ = y - F_1(x). Then obtain a new model F_2(x) = F_1(x) + h_1(x). The value of 'm' or the hyper parameter which gives the number of iterations of the residual correction procedure is obtained by cross validation.

2.3.2.4 Gradient Boosted trees

Model Update rule:

$$F_m(x) = F_{(m-1)}(x)) + \sum_{j=1}^{J_m} \gamma_{jm} I(x \epsilon R_{jm}) \tag{2.3}$$

2.3.3 Influence Ranking Algorithm

The algorithm for comparison of influence between two users whose network features have been collected from Twitter is presented. The parameters of various models are set initially and these values are modified till the ideal values are obtained. The cross validation set is used for this purpose.

Algorithm 1 Influencer Ranking model

```
Set initial seed for random numbers
Set the training control values
Set the tuning grid for parameter search
for each parameter set do do
    for each resampling iteration set do do
        hold out specific samples
        Pre process the data (Center and Scale)
        Fit the model on the remaining samples
        Predict the held out samples
    end for
    Calculate the average performance across held out predictions
end for
Determine the optimal parameter set
Fit the final model to all the training data using optimal parameter set
```

2.4 Experimental Study

2.4.1 Experiment setting

The performance of the Influence Ranking algorithm in Section III-(C) is studied to understand the best set of features to measure influence.

2.4.1.1 Dataset

The dataset comprises a standard, pair-wise preference learning task. Each data point describes two individuals whose identities are anonymized and their future references in the paper are made using labels 'A' and 'B'. For each person, 8 pre-computed, non-negative numeric features based on twitter activity are provided. The binary label represents a human judgement about which one of the two individuals is more influential. The Ground truth labels provided are 0/1 to indicate which of the users is more influential. The test set has 5952 entries for which label has to be predicted.

Table 2.2: Description of the dataset

Training set size	**Test set size**	**Feature vector**	**Classification**
5500	5952	22	Binary

2.4.2 Results

The accuracy and performance on test set was measured for the Three layered ANN trained using correlated and uncorrelated predictors.

Table 2.3: Modeling influence using Multilayered Perceptron with Backpropagation

Sample size	**Hidden units**	**Training Accuracy**	**Training Kappa**
3521	16	72.29	44.55
3521	24	71.29	43.42
3521	32	71.55	43.04

For above experiment correlated features (≥ 0.75) were removed. Cross validation technique used was X-cross fold with X = 5 and 5 times repeat. Training set accuracy was used as selection criteria for evaluation and hidden units were decided as 16. The trained model gave an accuracy of 0.801 on the test set.

Table 2.4: Modeling influence using Multilayered Perceptron with Backpropagation

Sample size	**Hidden units**	**Training Accuracy**	**Training Kappa**
3521	22	73.11	46.19
3521	33	72.96	45.89
3521	44	73.08	46.14

For above experiment correlated features were allowed for training. Cross validation technique used was X-cross fold with X = 5 and 5 times repeat. Training set accuracy was used as selection criteria for evaluation and hidden units were decided as 22. The trained model with 22 hidden nodes gave accuracy of 0.81 on test set.

Modeling influence using Decision trees with gradient boosting was performed. To tune the parameters of the trees such as Number of Trees N_{trees}, Shrinkage, Interaction depth and Minimum observations in nodes, a grid search was conducted. Training set Accuracy was the criteria to select the optimal model. Fig. 2 shows the performance of the algorithm during grid search and the accuracy is seen in Fig. 2 to degrade as the interaction depth increases beyond 5 and the shrinkage is increased beyond 0.1. The final parameter values for the optimal model obtained through grid search were N_{trees} = 110, interaction depth = 5, shrinkage = 0.1 and minimum observations in nodes = 20. The accuracy obtained on the test set was 0.8601.

Random forests (Rf) technique to model influence using the correlated features and removing correlation was performed. Advantage of random forests models are better accuracy but at the cost of interpret-ability of the model. The tuning parameter is Number of trees for these model whose value has to be determined empirically. The final model selected using training set accuracy had classification accuracy of 0.856 on the test set. The model with correlated features gave slightly better result 0.859. Removing correlation doesnt improve accuracy on this dataset for the ANN and Rf.

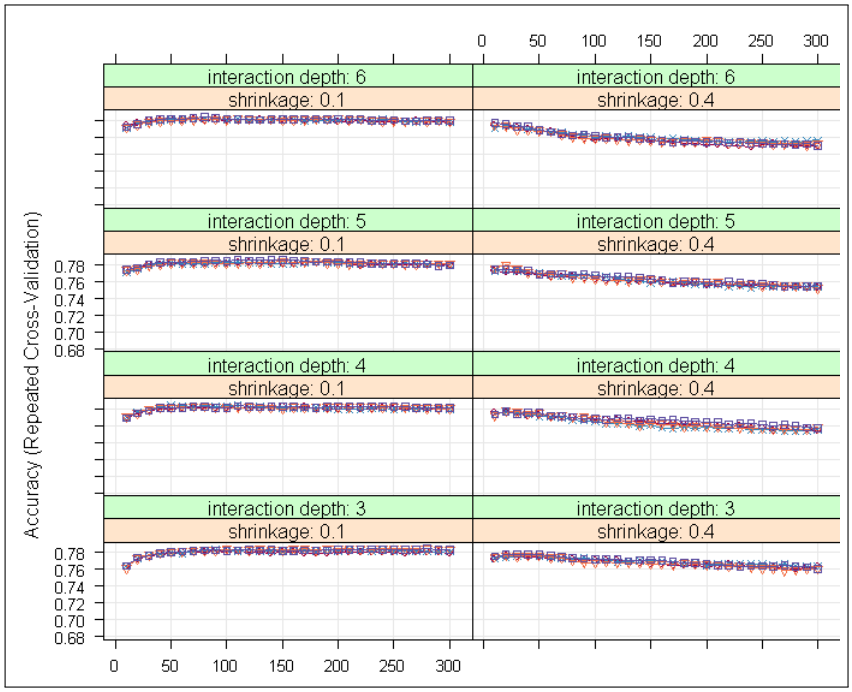


Figure 2.2: Performance of Gradient Boosted Trees

Improving accuracy using ensembling, boosting or bagging at the cost of interpretability is a disadvantage for implementation. Such techniques have improved accuracy but such models obtained had more theoretical importance than practical value as seen in the NetFlix competition. Table V contains the training set and test set accuracy of greedy ensembles of GBM Trees, Random forests and ANN obtained in Section IV-B. The test set accuracy of the models is used as the selection criteria and Extreme Gradient Boosting has the highest accuracy on the test set at 0.87. ROC curves to denote the performance of Ensembled models of ANN, GBM, Rf is shown in Fig 3 and Area under curve is used as the selection criteria to obtain the optimal model. The results are shown for each combination in Fig. 3. The correlation between the predictions of GBM and Rf is 0.88 and so their ensemble was discarded due to highly correlated predictions. The highest accuracy on the test set is seen for the ensemble of GBM and Rf as shown in Table V.

Table 2.5: Ensembled Predictors

Sr. No	Technique	Training Accuracy	Test set Accuracy
1	Boosting	0.7856	0.82
2	Greedy Ensemble (ANN,GBM,RF)	0.86	0.867
3	Greedy Ensemble (GBM,RF)	0.86	0.83
4	Averaging Predictors (GBM,RF)	0.79	0.866
5	Extreme Gradient Boosting	0.79	0.87

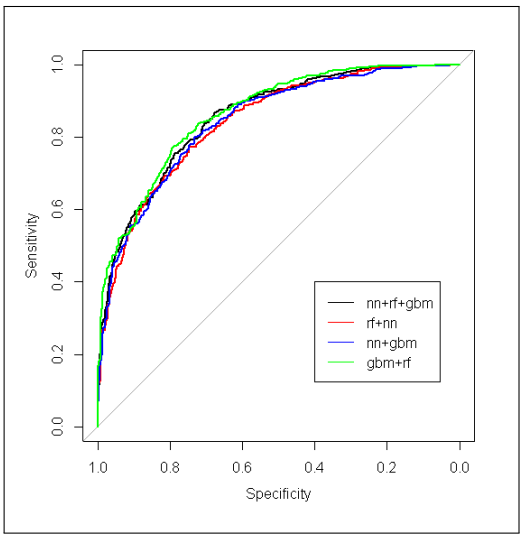


Figure 2.3: Performance of Greedy Ensembles

2.5 Conclusion

The framework for ranking influential nodes in a social network based on characteristics obtained from a nodes interaction on the social network has been presented in this paper. The influence is modeled using machine learning techniques unlike the conventional influence maximization approaches. Features of the individual nodes obtained from its interaction characteristics are used and the network architecture is not

considered. The performance metrics of the algorithms used in the experiments is classification accuracy and using this objective criteria Extreme Gradient Boosted decision tree has shown the highest accuracy. The influence ranking algorithm in this paper is a suitable technique for computation of the influence score of various nodes and this has been validated by the extensive experiments performed on Twitter dataset.

Chapter 3

A Comparative Analysis of Community Detection Algorithms on Social Networks

3.1 Introduction

Networks are used to graphically represent relationship or structure in many complex systems which can be seen in natural, technological or social settings. Understanding the process of formation of such networks or studying why certain systems exhibit a particular structure can provide insights into various phenomena that are commonly seen in real life communities such as diffusion, contagion and influence. These reasons have given birth to the scientific study of networks which became a multi disciplinary field spanning physics, computer science as well as social sciences [1] [14]. One reason for this occurrence is that any system in these fields can be represented in the form of a Network, a second reason is that the theories for studying such Networks have been at the intersection of all these fields [15]. Structurally a network or graph consists of nodes (vertices) and edges. An edge connects typically two nodes but if three or more nodes are connected by a single edge then such an edge is known as a hyperedge and such graphs are called hypergraphs. Other common networks seen in the world are scientific collaboration networks, where the nodes are the researchers and edges between them denote that they are working on the same research topic. An additional example is the Internet which in itself is a massive web graph consisting of web pages (nodes) and hyperlinks (edges). So it is concluded that in general any system can be represented as a network E.g citation networks, social networks, protein communities etc [15].

Most networks of interest demonstrate community structures i.e. nodes (vertices) in them form a dense subgraph. Such subgraphs are referred to as clusters, modules or communities and they exhibit a degree of autonomy in the network[1]. These regions are called autonomous as the nodes in them interact frequently with other nodes in the same clusters than with nodes outside the clusters. Thus, such clusters correspond to

entities that interact regularly to perform a function such as hormones responsible for producing an effect in the body or sensors in a home automation system responsible for detecting human presence in a room etc. Real world networks are so large that it is computationally infeasible to develop techniques for their study so in practise, methods are used that can simplify the structure of these large networks before any useful information can be extracted from them. These methods are known as Community Detection algorithms. There are over 100 algorithms that have proliferated in the network literature [16] [18]. The task of identifying communities is important as it offers insight into how a network is organised. This is because individual communities are functional units of the system and they help in understanding the role of the system.

The vertices in the network can also be classified on the basis of their roles they play with respect to the communities that they are a part of [15]. A central location makes the nodes important for diffusion of information within the network and so such nodes represent figures of importance with respect to that community. Similarly, the nodes that are located at the boundaries of a community might be acting as brokers for passing information to other communities and possibly play an important role in constraining the dynamics of spreading processes that occur on the network [16]. Other important reasons for creating coarse grain descriptions of networks are that using them missing information can be inferred about nodes by referring to the other nodes in its community or false information can be identified such as presence of an attribute that is uncommon in that community [16] [18].

Community detection of graphs is however an ill defined problem due to the absence of a universal definition of the object for detection i.e. "community". This has created multiple definitions for Communities, Methods to detect them and Performance evaluation techniques. Due to this ambiguity there is a diffusion of questionable literature in the domain of Network Science. Scientific opinion has changed its perception about networks. In the classical theory, clusters were viewed as dense sub graphs which exhibit a degree of autonomy in the network due to the presence of high edge density between nodes within the cluster than with nodes outside of it. Thus, the classical view relied more on the degree distributions of the nodes in the graph to determine clusters. This created ideas of strong communities and weak communities in a network that depended on the relation between the internal and external degree of the vertices of the graph [25]. The modern view relies more on calculating the probability of edge formation between nodes i.e. the community should be one in which there is a preferential linking pattern. This definition states that nodes in a community would have a higher probability of linking with each other than with nodes of other communities. Another approach that has developed used the link topology of the network and the trajectory of a random walker. The basis of this theory is that a random walk on the network would be concentrated for a longer interval in a dense sub-graph that corresponds to a community. This is because links moving out of a community would be supposedly lesser than those in it [45].

In this paper, an evaluation of eight different state-of-the-art community detection algorithms available in the "igraph" package is performed. The package is a widely used collection of network analysis tools in R, Python, C and C++ and can be used on undirected and directed, weighted and unweighted graphs with overlapping or non overlapping communities. The graphs under examination are "Egonets" or egocentric networks of individuals obtained from Facebook. A review of the most widely used community detection algorithms is given in Section II followed by Experimental work in Section III and Conclusion in Section IV.

3.2 Community Detection Algorithms

3.2.1 Walktrap - Computing communities in large networks using random walks

The algorithm is based on the intuition that random walks on graphs are trapped into a dense part of the graph and such dense sub-graphs corresponds to communities. Walktrap is a agglomerative, hierarchical clustering algorithm that allows finding community structures at different scales.

The starting point is an initial partition P_1 of the graph with n communities corresponding to the n vertices of the graph i.e. Partition $P_1 = \{\{v\}, v \epsilon V\}$. A distance measure is used to compute vertex similarity between all adjacent vertices. Then the partition modifies itself by repeating the below operations at each step:

- For two adjacent communities C_1 and C_2 in P_k merge into single community $C_3 = C_1 \cup C_2$ and create a new partition $P_{k+1} = (P_k : \{ C_1 , C_2 \}) \cup \{ C_3 \}$ if it satisfies a criteria based on the distance between them, and
- update distance between adjacent communities.

The algorithm at each step obtains a hierarchical data structure of communities called dendrogram. The algorithm computes the communities in time O(mnH) where n = $\mid V \mid$ vertices, m = $\mid E \mid$ edges and H = height of the dendrogram. For real world graphs which are sparse (m = O(log n)) it comes to O(n^2 log n) [39]. The tunable parameters in this is the step size of the random walker t which is used to calculate the probability that the random walkers shall move from a vertex i to a vertex j. This probability is used to calculate the similarity between vertices and create clusters. The drawback of this method is that it is parameter dependent.

3.2.2 Finding community structure in very large networks

The "Fastgreedy" technique implemented in the paper has a running time of O($mHlogn$) on a graph with m = | E | edges, n = | V | vertices and h = height of the dendrogram [10]. In real world graphs which are sparse the computation time is linear O(n log^2 n). The algorithm is based on the greedy optimization of modularity and utilizes shortcuts in the optimization procedure of the original algorithm based on Greedy optimization of modularity [9] and efficient data structures to reduce the time complexity from O(n^2) on sparse graphs to O(n log^2 n). Initially, every vertex belongs to a separate community, and communities are merged iteratively such that each merge is locally optimal. The algorithm stops when the modularity cannot be increased. It has lower time complexity than other techniques and its key drawback is that communities which have nodes and edges below a certain threshold are merged with adjacent communities. The algorithm has detected super communities in graphs that have no underlying clustering structure. It also relies on "modularity optimization" using approximation algorithms to reduce time complexity. However these have produced lower values of modularity than newer versions that have used Simulated Annealing to optimize modularity [41].

3.2.3 Finding and evaluating community structure in networks

The "Edge-betweenness" is a hierarchical decomposition process where edges are removed in the decreasing order of their edge betweenness scores. This is motivated by the fact that edges connecting different communities have a higher probability to occur on the shortest paths between nodes of different communities. However the algorithm has a high running time of O(m^2n). The repeated calculation of edge betweenness after an edge is removed has to be done for the entire graph. This affects the scalability of the algorithm to large graphs. At each iteration of this approach a full dendrogram is built and a measure to determine the optimal cut of the dendrogram can be made using modularity [35] [37].

3.2.4 Near linear time algorithm to detect community structures in large-scale networks

"Label Propagation algorithm" assign an initial unique label at random to the nodes of the network. Each label corresponds to a unique community to which the nodes belongs to. Then a particular node n_1 having 'k' neighbours determines it community affiliation based on the most frequent label amongst its neighbours. The problem however is that subgraphs in the network that are bi-partite or nearly bi-partite in structure lead to oscillations of labels. Hence the label updation step is performed asynchronously. The stopping criteria of the iterative label assigning and re-assigning is till each node has a label to which maximum number of his neighbours belong to. Since the stopping criteria is not a measure to be maximised or minimised the algorithm has no unique solutions in heterogenous graphs with underlying community structure. The method

has linear time complexity as every iteration is finished in O(m) but yields different results based on the initial configuration (which has to be decided randomly), therefore one should run the method a large number of times and then build a consensus labeling, which could be tedious [4].

3.2.5 Fast unfolding of communities in large networks

"multilevel.community" is heuristic algorithm for obtaining communities from graphs by optimising the partition measure of modularity [43]. Modularity is the fraction of edges within a community substracted from expected fraction if edges were distributed at random. It is represented by Eqn 1. Since modularity optimization is NP-hard, an approximation algorithm is used that gives a running time of O($nlogn$).

$$Q = \frac{1}{2m}\sum_{i,j}^{n}[A_{i,j} - \frac{k_i k_j}{2m}]\delta(C_i C_j) \tag{3.1}$$

Where,

- A_{ij} is edge weight between nodes i and j.
- k_i and k_j are degrees of nodes i and j in case of unweighted graphs.
- m is the sum of all edge weights in graphs
- c_i , c_j are communities of nodes.
- δ is 1 if edge exists between c_i , c_j and 0 otherwise;

The algorithm has two phases that are iteratively repeated:
Initially each nodes is assigned to its own unique community. Then for each node, the change in modularity is calculated by removing it from its community and moving it to the community of its neighbours. The change in modularity ΔQ is given by Eqn 2.

$$\Delta Q = [\frac{\sum_{in} + 2K_i, in}{2m} - (\frac{\sum_{tot} + k_i}{2m})^2] - B \tag{3.2}$$

$$B = [\frac{\sum_{in}}{2m} - (\frac{\sum_{tot}}{2m})^2 - (\frac{k_i}{2m})^2] \tag{3.3}$$

Where,

- $\sum_{in}$ is sum of all weights of links inside community to which i is being assigned.
- $\sum_{tot}$ is sum of all weights of links to nodes in community.
- k_i is weight degree of i
- k_i, in is the sum of the weights of links between i and other nodes in the cluster.
- m is the sum of the weights of all links in the network

After calculation of ΔQ for all neighbour nodes of i, it is placed in the appropriate community that achieves local optima. This step is repeated for all nodes sequentially till each is assigned to its suitable community. In the second phase, the clusters formed after above step are treated as a single meta-node. The links within a cluster are treated as self loops to the cluster and the links in between clusters are represented as weighted edges between communities. The first pass is repeated again till all communities are organised in a hierarchy. This technique is most suitable for large graphs due to the low time complexity. However, the effects of the technique on the null benchmark i.e. Erdos Renyi random graphs is not verified. The technique is significant for detecting structure in overlapping clusters but not on graphs having non overlapping clusters.

3.2.6 Statistical mechanics of community detection

"spinglass.community" algorithm approaches the problem of community detection as finding the ground state of an infinite ranged Potts spin glass [36]. In this model, each vertex can be in one of c spin states, and the interactions between the particles (i.e. the edges of the graph) specify which pairs of vertices would prefer to stay in the same spin state and which ones prefer to have different spin states. The model is then simulated for a given number of steps, and the spin states of the particles in the end define the communities. The tunable parameter of this technique is the upper limit for the number of clusters c which make it supervised and not suitable for real world graphs. The algorithm is also non deterministic because of simulations needed. In sparse graphs the computational complexity of the algorithm is $O(n^{3.2})$

3.2.7 Finding community structure in networks using the eigenvectors of matrices

"leading.eigenvector.community" algorithm uses the concept of modularity maximization to obtain optimal partitions of the graph [36]. Due to the NP-hard nature of this problem, the modularity matrix is used for calculating the modularity. The eigenvalues and eigenvectors of the matrix are used for clustering. The largest eigenvalue is used to maximise the modularity of the network. This technique belongs to the spectral clustering techniques. This method has higher time complexity than the fast greedy method. Its computational complexity on sparse graphs is $O(n^2)$.

3.2.8 Maps of random walks on complex networks reveal community structure

"infomap.community" algorithm finds community structure that minimizes the expected description length of a random walker trajectory[36]. The algorithm is based on the Map Equation which yields the description length of an infinite random walk on a graph. The vertices are assigned unique codes and the random walk in the graph is described by the codes of the vertices it visited. Since each vertex has a unique code, the description can be lengthy. The description length can be reduced in a community

structure by following the principles of geographic maps where vertices in different communities can have the same code. The best partition is the one that yields minimum description for the random walk.

The flow based methods provide different partitions than the methods based on structural features of the network like modularity. The results are striking in graphs having directed edges as these constrain the flow in the graph. The Map equation based techniques give importance to flow and are suitable for networks where structural features affect the dynamics of processes in the network like spread of epidemics, scientific collaborations etc. The runtime of the algorithm is $O(E)$

3.3 Experiments

The performance of the algorithms in Section II has been evaluated on the dataset provided in Section III (A)

3.3.1 Dataset

The testing of community detection algorithms is done on real or artificially generated networks where ground truth label for the communities is known as in the case of Zachary's karate club dataset or not known as in the case of GN bechmark or the LFR benchmark. The GN benchmark doesn't have the properties of a real network and so the LFR benchmark is used. In the LFR benchmark the vertex degree and community size are power law distributed as is seen in real world communities [30]. In this paper, in the evaluation of the algorithms "Egonets" of 60 user profiles obtained from Facebook are used. Egonets consist of a focal node "ego" and nodes to whom the ego is connected directly called "alters" along with ties between the alters. Such a network has hierarchical, overlapping communities along with homophilous strong ties.

Table 3.1: Statistics of the Degree Distribution

Min	**I Quad.**	**Median**	**Mean**	**III Quad.**	**Max**
0	6	15	24.93	33	669

Table 3.2: Statistics of the Order of the Egonets

Min	**I Quad.**	**Median**	**Mean**	**III Quad.**	**Max**
45	116.5	219	242	322	670

Table I shows the statistics of the degree distribution obtained from all 60 egonets and Table II shows the summary statistics of number of the nodes of the 60 egonets. Fig 1 shows the histogram of degree distribution which resembles a power law distribution.

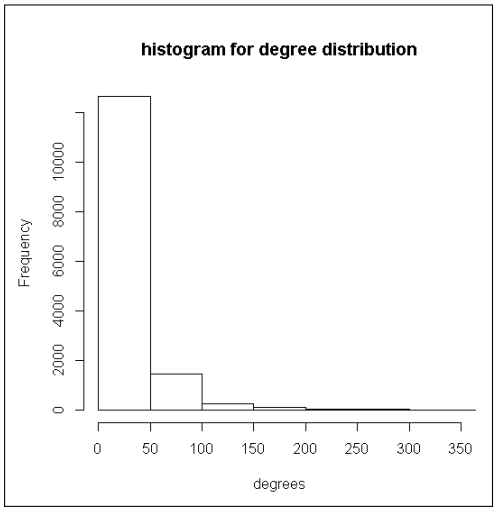


Figure 3.1: Histogram of Degree Distribution

3.3.2 Results

Table III to X show the performance of the eight algorithms on the datasets. The results represent the modularity metric calculated on the optimal community structure generated on the dataset by the algorithms is given.

Table 3.3: Statistics of Performance of walktrap.community on Egonets

Min	**I Quad.**	**Median**	**Mean**	**III Quad.**	**Max**
0.04654	0.38017	0.50352	0.47555	0.58518	0.84149

Table 3.4: Statistics of Performance of fastgreedy.community on Egonets

Min	**I Quad.**	**Median**	**Mean**	**III Quad.**	**Max**
0.2411	0.4220	0.4927	0.4960	0.5813	0.8299

Table 3.5: Statistics of Performance of edge.betweeness.community on Egonets

Min	**I Quad.**	**Median**	**Mean**	**III Quad.**	**Max**
0.1566	0.3600	0.4896	0.4719	0.5921	0.8528

Table 3.6: Statistics of Performance of label.propagation.community on Egonets

Min	**I Quad.**	**Median**	**Mean**	**III Quad.**	**Max**
0.0000	0.3323	0.4824	0.4469	0.5820	0.8482

Table 3.7: Statistics of Performance of multilevel.community on Egonets

Min	**I Quad.**	**Median**	**Mean**	**III Quad.**	**Max**
0.2523	0.4546	0.5264	0.5205	0.6141	0.8557

Table 3.8: Statistics of Performance of spinglass.community on Egonets

Min	**I Quad.**	**Median**	**Mean**	**III Quad.**	**Max**
0.2573	0.4379	0.4988	0.4908	0.5640	0.7188

Table 3.9: Statistics of Performance of leading.eigenvector.community on Egonets

Min	**I Quad.**	**Median**	**Mean**	**III Quad.**	**Max**
0.2372	0.4250	0.5039	0.5050	0.6013	0.8321

Table 3.10: Statistics of Performance of infomap.community on Egonets

Min	**I Quad.**	**Median**	**Mean**	**III Quad.**	**Max**
0.04513	0.41261	0.50660	0.48933	0.60126	0.84931

The highest mean modularity is obtained by the multilevel community detection algorithm. The real world datasets exhibit a community structure which might not be based on modularity. A second evaluation metric is used to calculate the edit distance between the ground truth communities in the dataset and the predicted communities by the algorithms.

Table XI shows the running time in millisecs needed by the algorithms on the dataset. The average nodes in the network were 242 with the minimum size of 45 and maximum of 670. The label propagation algorithm has obtained the lowest running time on the datasets. The highest time complexity is seen in the edge betweenness algorithm and it proves that it may not scale well to large datasets.

3.3.3 Edit Distance metric

Edit distance calculates the minimum number of edit operations needed for transformation of the predicted solution with the actual solution. Each of the following operations cost one edit:

Table 3.11: Running time of Algorithms on Egonets

Name	Min	I Quad.	Median	Mean	III Quad.	Max
EDG	2583	4231	4641	5711	5519	11111
IMaP	0.01	0.05	0.115	0.247	0.360	1.530
MLTL	0.01	0.01	0.01	0.012	0.02	0.05
WLK	0.00	0.01	0.02	0.046	0.052	0.47
FTG	0.00	0.00	0.02	0.08	0.082	1.12
LDG	0.00	0.04	0.09	0.14	0.152	1.96
LBP	0.00	0.00	0.00	0.0048	0.01	0.02
SPG	2.17	7.01	16.7	31.2	36.04	250.45

- Add user to an existing circle.
- Remove user from a circle.
- create a circle with one user.
- delete a circle with one user.

Table 3.12: Edit Distance of Algorithms on Egonets

Sr No.	Name	Edit Distance
1	edge.betweenness	14284
2	multilevel	15162
3	infomap	13988
4	label.propagation	14538
5	spinglass	15256
6	fastgreedy	14736
7	walktrap	14696
8	leading.eigenvector	15304

From comparison of the edit distance metric shown in Table XII the infomap algorithm has obtained the lowest edit cost amongst the algorithms.

3.4 Conclusion

Egocentric networks present a challenge for community detection as the community structure in them is defined by the 'ego' to which the network belongs to. The presence

of ground truth labels to the communities allows one to evaluate the precision of the algorithms effectively. The choice of algorithms therefore can be based on objective criteria like accuracy, running time and computational complexity. From our results, all algorithms with the exception of spinglass and edge betweenness were scalable and could be used on large networks. The partition quality is also an indicator of the accuracy of the algorithm, however real world datasets might not be based on optimum modularity. On this scale, the algorithms had mean modularity was close to each other. This was seen even in the case of algorithms like infomap and label propagation that don't partition based on optimum modularity. Finally, real world datasets exhibit heterogeneity due to which they may contain noise and also they do not have an fixed objective criteria for creation of communities. Therefore the edit distance measure showed high cost which meant that the partitions created by the algorithms had disagreements with the partitions made by the respective users or 'egos'.

Chapter 4

Empirical Analysis of Data Clustering Algorithms

4.1 Introduction

As the Digital transformation of the society gathers pace, there is an increase in proliferation of technologies that simplify the process of recording data efficiently. Low cost sensors, RFIDs , Internet enabled Point of Sales terminals are an example of such data capturing devices that have invaded our lives. The easy availability of such devices and the resultant simplification of operations due to them has generated repositories of data that previously didn't exist. Today, there exist many areas where voluminous amount of data gets generated every second and is processed and stored. Some of the common fields are social networks, sensor networks, cloud storages etc. The availability of data as well as the reduction in cost of computation power have jointly boosted the fields of machine learning, pattern recognition, statistical data analysis and in general data science.

Even though such a volume provides huge opportunities to academia and industry it also represents problems for efficient analysis and retrieval [1] [26]. To mitigate the exponential time and space needed for such operations data is compacted into meaningful summaries i.e. Exploratory Data Analysis [E.D.A.] which shall eliminate the need for storing data. In unsupervised learning literature such summaries are equivalent to "clusters". E.D.A. helps in visualization and promotes better understanding of the data. It utilizes methods that are at the intersection of machine learning, pattern recognition and information retrieval. Cluster analysis is the main task performed in it.

A Cluster in a data is defined objectively using dissimilarity measures such as edit distance, density in a euclidean or non euclidean data space, distance calculated using Minkowski measures, proximity measures or probability distributions. All measures concur that a threshold value should be set for grouping of objects in a cluster and objects which exceed such a threshold are dissimilar and should be separated from the

cluster. Clustering gives a better representation of the data since all objects within a cluster have less variability in their attributes and they can be summarised efficiently. Clustering has found applications in other fields like estimating the missing values in data or identifying outliers in data [23].

Clustering is thus a meta learning approach for getting insights into data and in diverse domains such as Market Research, E-Commerce, Social Network Analysis and Aggregation of Search Results amongst others. Multiple algorithms exist for organizing data into clusters however there is no universal solution to all problems. No consensus exists on the "best" algorithm as each is designed with certain assumptions and has its own biases. These algorithms can be grouped into methodologies such as Partitioning based, hierarchical, density based, grid based, message passing based, neural network based, probabilistic and generative model based. However in terms of complexity it is a NP-hard grouping problem and so existing algorithms rely on approximation techniques or heuristics to reduce the search space in order to find the optimal solution. There is no universally agreed objective criteria for correctness or clustering validity and each of these algorithms has its own drawbacks and successes in solving the challenging problem of unsupervised clustering [6] [48] .

Motivated by these reasons, in this paper a review of the state of the art clustering algorithms is made to highlight their main strengths and weaknesses. Section II covers the theoretical aspects of these algorithms, Section III contains the Experiments performed using these algorithms and Section IV has the conclusion on the results.

4.2 Types of Clustering Algorithms

Various clustering algorithms are found in literature [14] [23] [26] [48] and are broadly categorized into categories on the basis of an algorithm designer's perspective with emphasis on the underlying clustering criteria:

4.2.1 Partition based clustering algorithms

The general principle in these algorithms is that a cluster should contain atleast one object and that each object must belong to exactly one group i.e. hard clustering. The Number of clusters k is pre-specified by the user making this a semi supervised algorithm although many strategies have been suggested to estimate the ideal number of clusters like the empirical method where $k = \sqrt{n}$ where $n = \mid N \mid$ points and Elbow method where the k is fixed as the turning point on the graph of k v/s Avg. distance to centroid. The objective function to be minimised in these k-partitioning algorithms is SSE i.e. Sum of Squared Distance [34].

$$SSE = \sum_{k=1}^{k} \sum_{x_i \epsilon c_k} \|x_i - c_k\|^2 \tag{4.1}$$

where c_k = centroid of the cluster,

Popular k-partitioning algorithms are K-Means which represents a centroid as the arithmetic mean of the objects in the cluster [28]. The algorithm was the most popular in this category even though it had drawbacks as it could not find non convex shaped clusters or handle non numerical attributes in higher dimensions. The time complexity was O(kN) making it suitable for large datasets, however the algorithm could theoretically take infinite iterations to converge and its mean based appraoch was sensitive to noisy data or data with outliers. Algorithms depending on Euclidean Distance measures suffer from the 'Curse of Dimensionality' due to distances being inflated in higher dimensions. K-Means++ by D. Arthur *et. al* 2007 [38] provides an improvement over K-Means by initializing the seed cluster centroids at maximum distance from each other. This technique has provided better results compared to random initializations with multiple repeats. Mini Batch K-Means by D. Sculley *et. al* [11] uses randomly sampled subsets of original data in each iteration of clustering. The approach improves computing time at the cost of slight reduction in accuracy compared to the Original K-Means algorithm.

To overcome the susceptibility to noise in mean based approaches, K-Mediods algorithm represented clusters by objects located near the centroids. Partitioning Around Mediods (PAM) [28] is the most popular approach for mediods based partitioning however it has a computational time complexity of O($k(n-k)^2$) and hence wouldn't scale well to large datasets. Modified version of PAM such as CLARA (PAM with sampling) [38] and CLARANS [11] were proposed. CLARA has a computation time complexity of O($ks^2 + k(n-k)$) and CLARANS has O(n^2) and so both methods wouldn't be applicable to large sets of data.

Kernel K-Means involves converting the points from euclidean space to high dimension kernel space. The kernel choices can be based on Mercer's criteria. R.B.F. or Gaussian is the common choice considered. The advantage of Kernel K-Means over K-Means is in finding non convex clusters albeit at the cost of computational time. BFR algorithm [5] is implemented for detecting clusters in large datasets in a single pass over the data. The algorithm makes a strong assumption that clusters have objects that are normally distributed and due to this it can't find clusters at tilted angles to the axes or clusters of random shapes.

4.2.2 Hierarchical Clustering algorithms

This category of algorithms have a completely unsupervised approach as they do not require the users to specify the number of clusters in advance. The algorithms in this category are of two types: Agglomerative Clustering (AGNES) and Divisive Clustering (DIANA). In Agglomerative, every object is treated as a cluster in the first pass and proximity based measures are used to determine which clusters can be merged in subsequent iterations. In divisive methods, a top down approach is followed in which

all datapoints are merged in a single cluster initially and in subsequent iterations the supercluster is split into subclusters till a stopping criteria is reached. Computation time is the key drawback in such algorithms O(n^3) / O($n^2 logn$) for priority queue implementations and so it is not applicable for large datasets which don't fit in main memory. Distance measures used are Cohesion (d = diameter of a cluster i.e. maximum distance between points in a cluster), Radius (r = maximum distance of a point from a cluster) or Density ($\frac{n}{d}$).

It is difficult to decide a distance measure for the data and hence the optimization objective isn't clear. Hierarchical clustering in general can't handle missing values except for probabilistic hierarchical methods and are sensitive to noise/outliers. The clusters once formed in an iteration cannot be undone in subsequent iterations. The result of clustering is a tree based data structure called dendrogram. In it the leaf nodes are the edges and clustering is indicated by joining leaf nodes by edges. To measure the dissimilarity between clusters several linkage methods are developed, the popular techniques are listed below.

- Maximum linkage clustering: max d (a,b) : $a \in A, b \in B$
- Minimum linkage clustering: min d (a,b) : $a \in A, b \in B$
- Average linkage clustering: $\frac{1}{|A||B|} \sum_{a \in A} \sum_{b \in B} d(a, b)$
- Centroid linkage clustering: $||C_i - C_j||$ centroid $C_i \in A$ and centroid $C_j \in B$.
- Ward's minimum variance method to minimize the within cluster variance.

Popular Algorithms of hierarchical clustering covered in the literature are BIRCH [55], CURE [20], ROCK [20], CHAMELEON [21]. BIRCH is a scalable, hierarchical clustering approach for large datasets. It builds a Clustering Feature tree which is a height balanced tree. Similar to a B+ Tree it has leaf nodes that represent clusters by a tuple: N,LS,SS where N is the number of points in the cluster, LS is the linear sum of N points and SS is the squared sum of N points. This creates compaction as individual data points are stored as summarised information. The non leaf nodes have atmost B entries and leaf nodes have atmost L subclusters that satisfy a threshold value for maximum radius. As new datapoints are assigned to nearest subclusters, the leaf nodes are split to remain under the threshold radius. A single pass is sufficient to cluster the data, however BIRCH is suitable only for numerical data and since each node in the CF Tree is limited by the size, the clusters formed may not be same as natural clusters. The algorithm doesn't perform well against non convex clusters as it uses radius to control the boundary of clusters.

CURE addresses the drawback of non-spherical shaped clusters. A constant number of c well scattered points are chosen to represent the cluster. These points are shrunk

towards the mean of the cluster by a fraction α. Clusters with closest pair of representative points are merged at each iteration. ROCK is for large datasets with categorical attributes. It has a bottom up approach to clustering as initially each data point is a separate cluster. Then pairwise similarity is calculated by Eqn. 2 and a goodness function in Eqn. 3 is used to combine clusters. Calculation of pairwise similarity and merging clusters which are highly similar is done iteratively till a stopping criteria is reached.

$$E_l = \sum_{i=1}^{k} n_i * \sum_{p_q, p_r \in C_k} \frac{link(p_q, p_r)}{n_i^{1+2f(\theta)}} \tag{4.2}$$

$$G(C_i, C_j) = \frac{link(C_i, C_j)}{(n_i + n_j)^{1+2f(\theta)} - (n_i)^{1+2f(\theta)} - (n_j)^{1+2f(\theta)}} \tag{4.3}$$

CHAMELEON constructs a sparse graph from the dataset using K-nearest neighbour. It obtains clusters by partitioning the graph by minimising the edge cut. Individual clusters are merged based on relative inter-connectivity and relative closeness both overcoming disadvantage of BIRCH, CURE, ROCK as they merge based on a single criteria. However it has a high time complexity of O($nm + nlogn + m^2logm$).

4.2.3 Fuzzy clustering

Fuzzy clustering algorithms assign a set of membership coefficients to each element which correspond to a "belongingness" or degree of membership to a cluster i.e. soft clustering. Fuzzy C-Means algorithm by Dunn in 1973 [3] and modified by Bezdek in 1981 [3] minimises the objective function in Eqn. 4 for this purpose, Eqn. 5 defines the degree of belongingness u_{ij}^m and Eqn. 6 defines the centroid C_j of a cluster.

$$\sum_{j=1}^{k} \sum_{x_i \in C_j} u_{ij}^m (x_i - u_j)^2 \tag{4.4}$$

Where,

- u_{ij} is the degree to which an observation x_i belongs to a cluster C_j
- $_j$ is the center of the C_j
- m is the real number $(1 \leq m \leq \infty)$ that defines the level of cluster fuzziness.

$$u_{ij}^m = \frac{1}{\sum_{l=1}^{k} (\frac{|x_i - c_j|}{|x_i - c_k|})^{\frac{2}{m-1}}} \tag{4.5}$$

$$C_j = \frac{\sum_{x \in C_j} u_{ij}^m x}{\sum_{x \in C_j} u_{ij}^m} \tag{4.6}$$

The algorithm minimises intra cluster variance but can converge to a local optimal solution. It depends on initialisation of the seeds and different initializations may lead to different results. The number of clusters k have to be specified in advanced which is another drawback.

4.2.4 Model Based Clustering Algorithms

The traditional clustering algorithms hierarchical and partition based clustering rely on heuristics whereas Model based algorithms assume that the data has been generated from a mixture of multiple probability distributions (Gaussian or multinomial) whose parameters mean, covariance matrix are to be estimated using the Expectation Maximization algorithm. The Bayesian information criteria or the Akaike information criteria can be used for selection of optimal number of clusters. The key drawback of this algorithms is that similar to k-means it also can converge to local optimal solution depending on the initial assignment of the k seeds. The objective function is not convex for these methods. Also, the optimization criteria can theoretically take infinite iterations to converge and a suitable threshold value has to be decided in advance. If the probabilities of the objects don't alter above this threshold then the algorithm can be stopped. Eqn. 6 is the prior probability that denotes the percentage of instances that came from source c. Eqn. 7 gives the mean i.e. expected value of attribute j from source c. Eqn. 8 gives the covariance matrix denoting the covariance of attributes j, k in source c.

$$P(c) = \frac{1}{n}\sum_{i=1}^{n} P(c|\vec{x_i}) \tag{4.7}$$

$$\mu_{c,j} = \sum_{i=1}^{n}\Big(\frac{P(c|\vec{x_i})}{nP(c)}\Big)x_{i,j} \tag{4.8}$$

$$\sum_{c}{}_{j,k} = \sum_{i=1}^{n}\Big(\frac{P(c|\vec{x_i})}{nP(c)}\Big)(x_{i,j} - \mu_{c,j})(x_{i,k} - \mu_{c,k}) \tag{4.9}$$

$$P(c|\vec{x_i}) = \frac{P(\vec{x_i}|c)P(c))}{\sum_{i=1}^{k} P(\vec{x_i}|c)P(c)} \tag{4.10}$$

$$P(\vec{x_i}|c) = \frac{1}{\sqrt{2\pi\sum_c}}\exp\Big(-\frac{1}{2}(\vec{x_i} - \vec{\mu_c})^T\sum_{c}^{-1}(\vec{x_i} - \vec{\mu_c})\Big) \tag{4.11}$$

4.2.5 Grid Based Clustering Algorithm

The key principle of grid based algorithms is to discretize the dataspace into grids and estimate the density by counting the number of points in a grid cell. CLIQUE [26] [46] was developed by Agrawal *et. al* to cluster high dimensional data in subspace. The algorithm identifies dense regions in a grid partition of the dataspace followed by selecting subspaces that contain clusters using Apriori principle. The dense connected

units in all subspaces of interest are chosen and minimal cover for each cluster is created. However in all grid based clustering approaches the quality of clustering depends on the number and width of the grid cells. STING by Wang *et. al* [46] divided the spatial area into rectangular cells at different levels of resolution forming a tree structure. A cell contain statistical parameters like mean, variance, standard deviations. This information is created by going through the data once and take O(n) time to generate clusters. The algorithm doesn't consider the spatial relationship between neighbouring cells before a parent cell is constructed and so all cluster boundaries are horizontal or vertical but no diagonally shaped boundary is detected [46].

4.2.6 Density based clustering algorithms

Cluster is defined as a connected dense component that can grow in any direction till the density continues to be above a threshold. This leads to automatic avoidance of outliers and detection of well separated clusters of arbitrary shapes. Popular methods based on this approach are DBSCAN by Kriegel *et. al* [46], OPTICS [46]. DBSCAN can find non linearly separable clusters and doesn't need the initial value of clusters to proceed. It uses euclidean distance measures to calculate distance between points in space and so is sensitive to curse of dimensionality. The parameters needed by DBSCAN are ϵ which defines the radius of neighbourhood around a point and minimum neighbours $MinPts$ of a point in its ϵ - neighbourhood.

In DBSCAN, pairwise distance is calculated between x_i and other points. For each point in the ϵ - neighbourhood of x_i if the $N_pts \leq MinPts$ then mark it as *core* point. Then for each *core* point create a new cluster or assign it to a cluster if it is not assigned already. Find recursively all its density connected points and assign them to the same cluster as the core point. Iterate through the remaining unvisited points in the dataset. At the end of all iterations, the unassigned points are outliers.

OPTICS is an extension of DBSCAN to address the drawback of detecting clusters in varying densities. It accepts the parameters of DBSCAN ϵ and $MinPts$ in neighbourhood $N_\epsilon(P)$. It additionally defines a new measure for every point known as $Core-Distance_{\epsilon,Minpts}(P) = C$ as in Eqn. 12 and $Reachability-Distance_{\epsilon,Minpts}(o,p) = R$ as in Eqn 13.

$$C = \begin{cases} UNDEFINED; & if|N_{\epsilon(p)}| < MinPts \\ smallestdistancetoN_\epsilon(p); & otherwise \end{cases} \tag{4.12}$$

$$R = \begin{cases} UNDEFINED; & if|N_{\epsilon(p)}| < MinPts \\ max(C, dist(o,p)); & otherwise \end{cases} \tag{4.13}$$

OPTICS produces a cluster ordering with respect to its density based clustering structure.

4.2.7 Affinity Propagation Clustering

This technique is based on concept of message passing between data points [12]. It doesn't require the user to specify the number of clusters in advance. The initial input is a similarity matrix s where $s(i,k)$ denotes suitability of point k to be exemplar or representative of point i. and $s(k,k)$ is higher for points more preferred as exemplars. the value can be set to median to produce moderate number of clusters whereas too high can get many clusters and too low shall give less clusters.

Two kinds of messages are exchanged between the points "responsibility" $r(i,k)$ reflects suitability of point k as an exemplar of i. The availability $a(i,k)$ denotes how appropriate it would be for i to choose k as its exemplar. Initially availability matrix is set to zero and the responsibilities for each point is calculate by the rule:

$$r(i,k) \leftarrow s(i,k) - \max a(i,k') + s(i,k') : \forall k', k' \neq k \tag{4.14}$$

Then availability is updated by the rules:

$$a(i,k) \leftarrow \min \left(0, r(k,k) + \sum_{i' \notin i,k} max(0, r(i'.k)) \right) for(i \neq k) \tag{4.15}$$

$$a(k,k) \leftarrow \sum_{i' \neq k} \max \left(0, r(i',k)\right) \tag{4.16}$$

The value of k such that $a(i,k) + r(i,k)$ is maximum is selected as an exemplar for point i. The iterative updates for availability and responsibility matrices can be terminated after a limited iterations or after changes in the messages fall below a threshold. The time complexity is of the order $\mathrm{O}(N^2T)$, $N = samples$ and $T = iterations$ [17].

4.2.8 Affiliation Graph Model

Community Affiliation graph model is a probabilistic generative model for the network [52] [53]. The assumption is that the network is generated by a model whose parameters have to be estimated. The model for a network with V vertices, C communities, M memberships, p_c probability associated with a community c is represented by tuple $B(V, C, M, p_c)$. Then the edge probability for each pair of node is computed by Eqn. 18

$$P(u,v) = 1 - \prod_{c \in M_u \cap M_v} (1 - p_c) \tag{4.17}$$

$$P(u,v) = 1 - exp(-F_u.F_v^T) \tag{4.18}$$

If $M_u \cap M_v = 0$ then P(u,v) = ε. F_u is the vector that denotes the strengths of association of a node u with each community in the network. The task is to find the matrix of memberships F that maximises the likelihood of generating the graph. The log-likehood of this is Eqn. 19. The Gradient update algorithm is used to find the value of F as shown in Eqn. 20.

$$l(F) = \sum_{u,v \in E} log(1 - \exp(-F_u.F_v^T)) - \sum_{u,v \notin E} (F_u.F_v^T) \tag{4.19}$$

$$\bigtriangledown l(F_u) = \sum_{v \in N(u)} F_v \frac{\exp(-F_u.F_v^T)}{1 - \exp(-F_u.F_v^T)} - \sum_{v \notin N(u)} F_v \tag{4.20}$$

4.2.9 Spectral Clustering

Spectral clustering outperforms k-means or single linkage and also has several advantages such as simple implementation. Spectral clustering can be solved by linear algebra methods [49]. The main tools for spectral clustering are graph laplacian matrices L. The fundamental concept is to partition a given similarity graph such that edges between different groups have low weights and edges within a group have high weights.

$$L = D - A \tag{4.21}$$

$$L_{sym} = D^-1/2 * L * D^-1/2 \tag{4.22}$$

$$L_{rw} = D^-1 * L \tag{4.23}$$

Where,

- D = Degree matrix
- A = Adjacency matrix

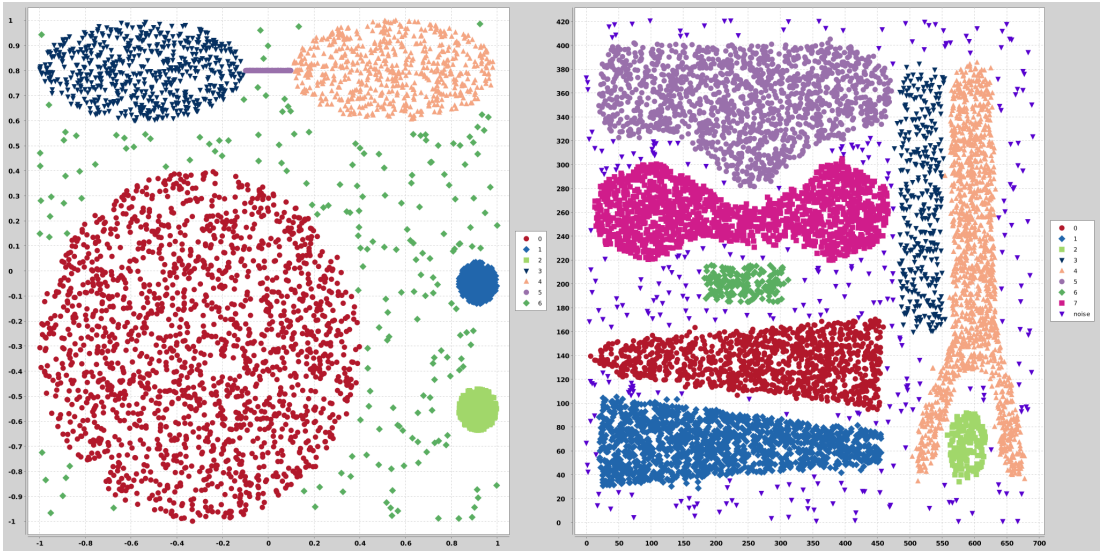
Spectral clustering popular examples are Unnormalised spectral clustering which uses L and Normalised spectral clustering with L_{sym} and L_{rw}. In both methods, the first k eigenvectors $u_1, u_2, ..., u_k$ corresponding to the k smallest eigenvalues are computed to get matrix $U \in R^{n*k}$ which has $u_1, u_2, ..., u_k$ as columns. Then for $y_i \in R^k$ which is the i^{th} row of U, all rows are treated as points and clustered by k-means to get k clusters. The normalised spectral clustering algorithms attempt to minimise inter cluster similarity and maximise intra cluster similarity whereas unnormalised spectral clustering attempts to minimise inter cluster similarity only. For computational simplicity L_{rw} is used in clustering.

4.3 Experiments

Section II has examined custering algorithms from a theoretical point of view and in this section their performance on a clustering benchmark dataset is provided for an empirical evaluation.

4.3.1 Dataset

The Datasets selected are CURE-T2-4K and CLUTO-T8-8K which are publicly available artificially generated benchmarks by ClueMiner. The clusters can be identified by visualization but performance of clustering algorithms produces different results.



(a) CURE-T2-4K (b) CLUTO-T8-8K

Figure 4.1: Artificial Benchmark Datasets for Clustering

Table 4.1: description of Dataset

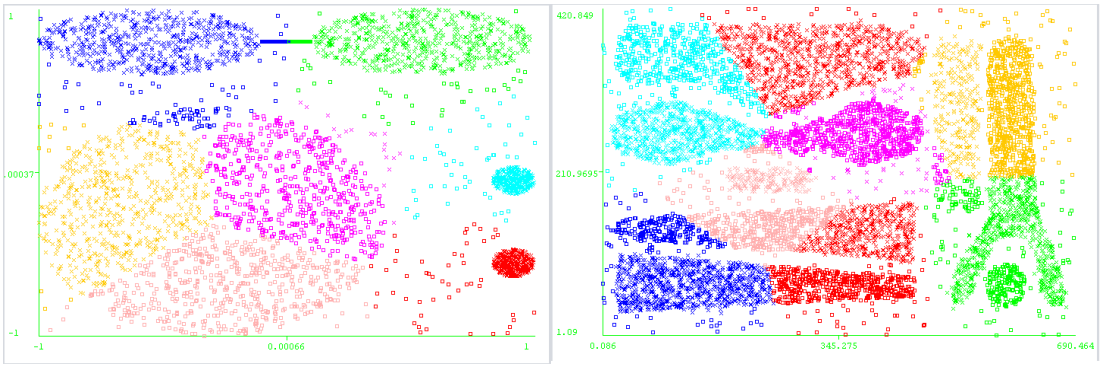
Name	Instances	Attributes	Classes
CURE-T2-4K	4200	3	7
CLUTO-T8-8K	8000	3	9

4.3.2 Experimental Results

4.3.2.1 Partition based clustering Algorithm

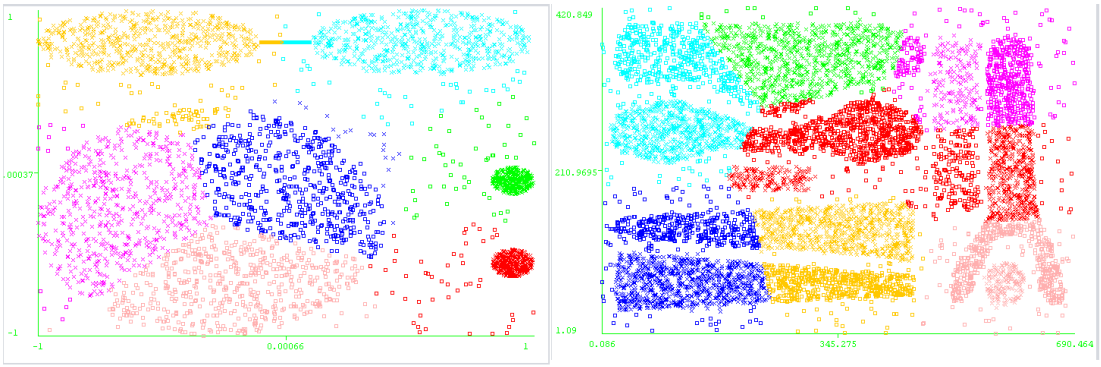
Results of K-Means in Fig 2, K-Means++ in Fig 3 and Kernel K-Means with RBF kernel in Fig 4 applied on the datasets shows the detection of clusters symmetric along

the axes is better than irregular shaped clusters.



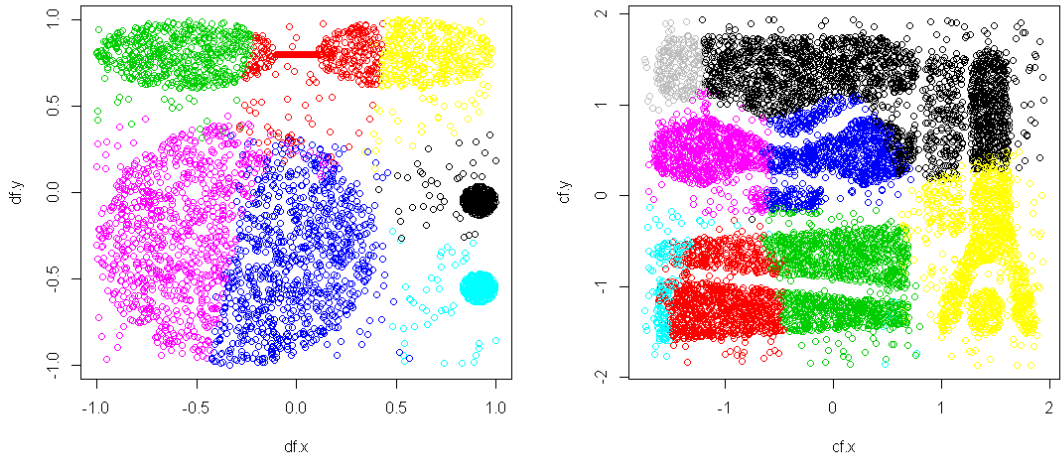
(a) CURE-T2-4K (b) CLUTO-T8-8K

Figure 4.2: Clustering based on K-Means



(a) CURE-T2-4K (b) CLUTO-T8-8K

Figure 4.3: Clustering based on K-Means++

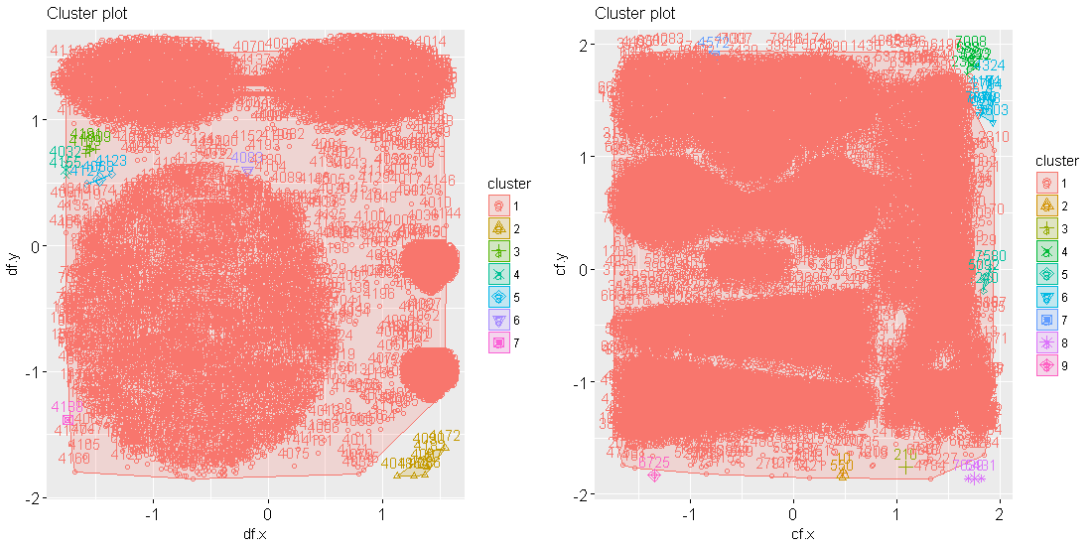


(a) CURE-T2-4K (b) CLUTO-T8-8K

Figure 4.4: Clustering based on Kernel K-Means

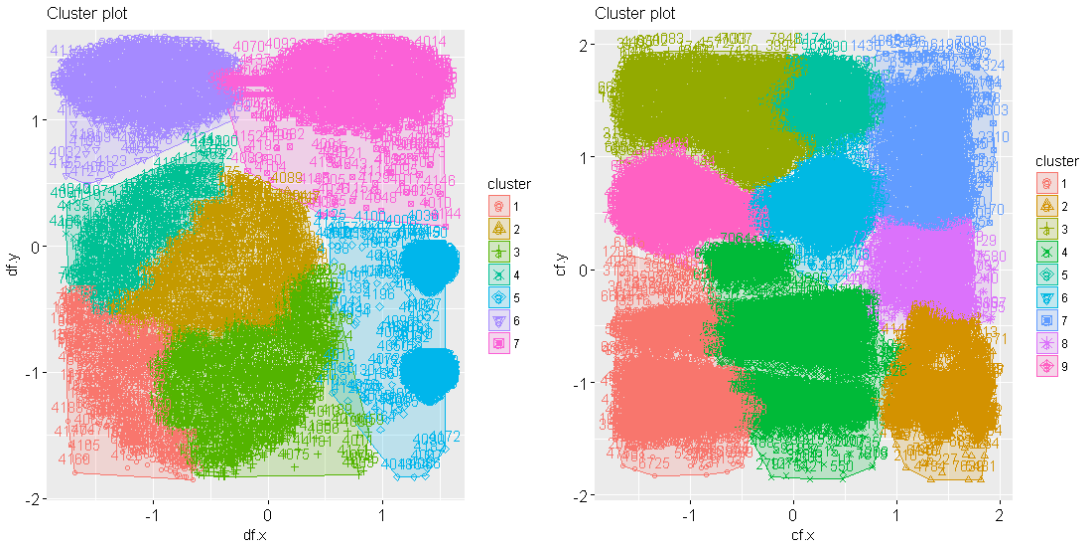
4.3.2.2 Hierarchical clustering Algorithms

Hierarchical clustering algorithms based on different cluster agglomeration methods were applied on the datasets. The methods were not able to separate the noise from the dataset. Another drawback of these techniques was inability to handle noise / outliers. The results show the merging of noisy data with the clusters. The single linkage method has resulted in a super cluster for both datasets as shown in Fig 5. In Fig 6. Average linkage method could not identify clusters of irregular shape. Agglomeration based on complete linkage has resulted in compact clusters as seen in Fig 7. Agglomeration based on centroid has been susceptible to noise as seen in Fig 8. as points not belonging to any cluster were included in clusters. Agglomeration based on Ward's minimum variance criteria too included noise points in the clusters and failed to identify irregular shaped clusters.



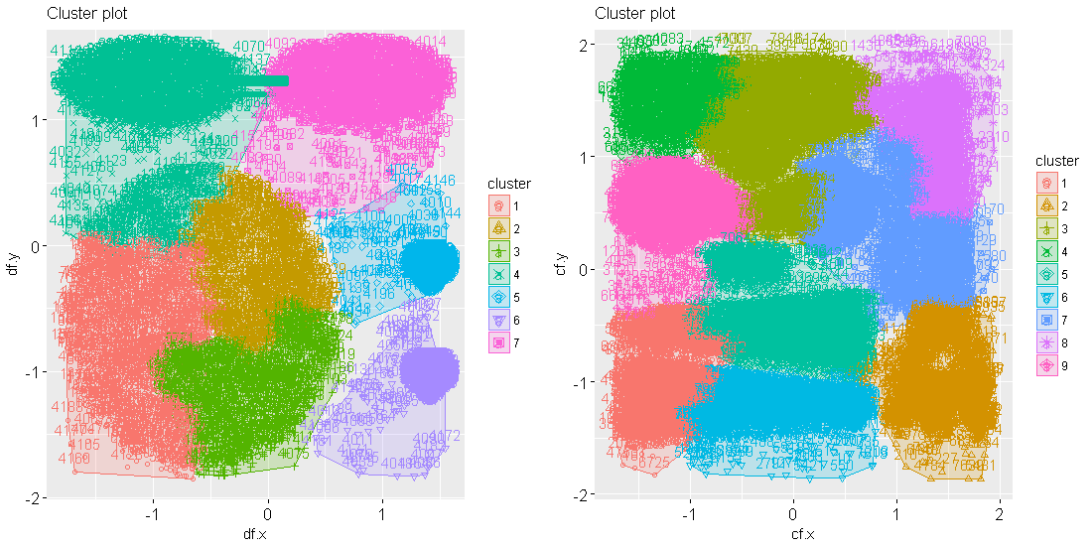
(a) CURE-T2-4K (b) CLUTO-T8-8K

Figure 4.5: Hierarchical Clustering based on Single linkage



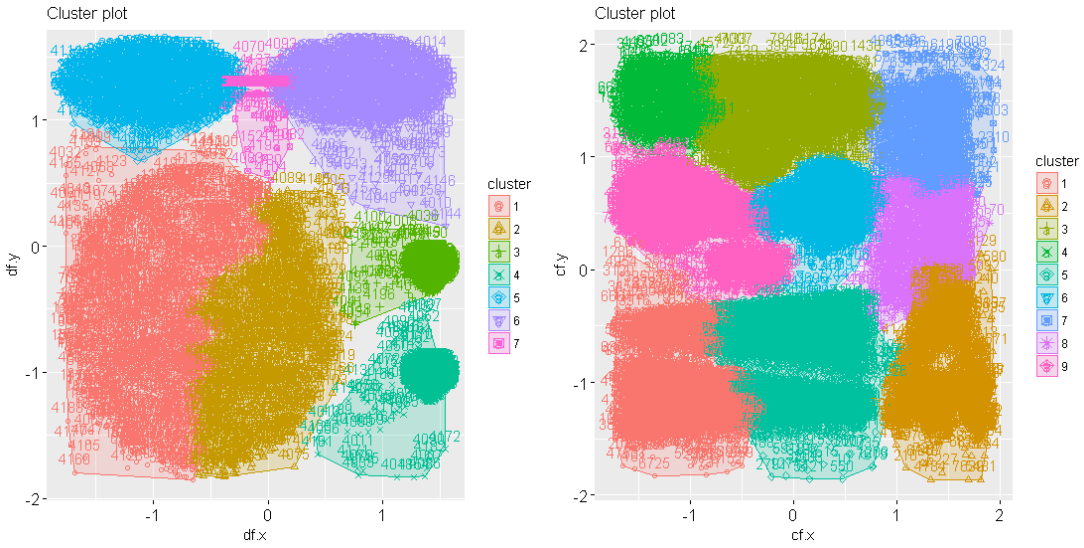
(a) CURE-T2-4K (b) CLUTO-T8-8K

Figure 4.6: Hierarchical Clustering based on Average linkage



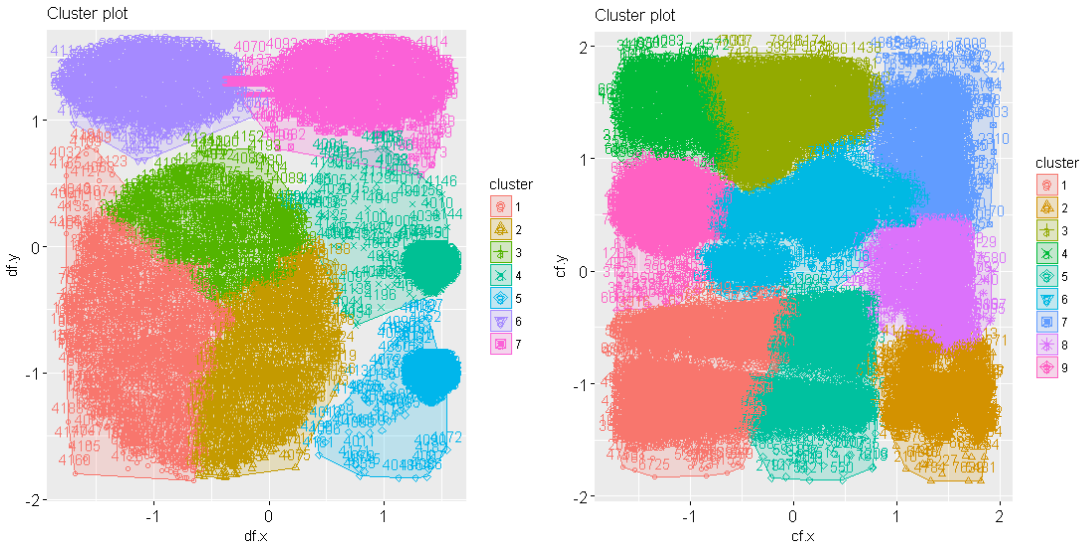
(a) CURE-T2-4K (b) CLUTO-T8-8K

Figure 4.7: Hierarchical Clustering based on Complete linkage



(a) CURE-T2-4K (b) CLUTO-T8-8K

Figure 4.8: Hierarchical Clustering based on Centroid linkage

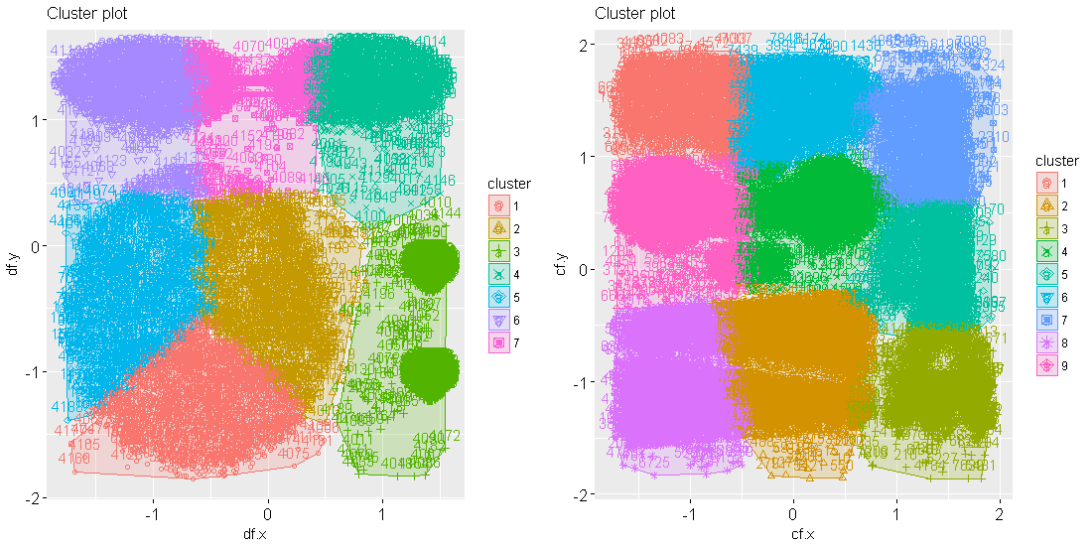


(a) CURE-T2-4K (b) CLUTO-T8-8K

Figure 4.9: Hierarchical Clustering based on Wards minimum variance method

4.3.2.3 Fuzzy clustering Algorithm

The number of clusters have to be specified in advance and the clustering overlap is controlled by the fuzzifier parameter m. Random initialization was done with multiple repeats to avoid convergence to local minima. The approach was sensitive to noise and clusters were symmetric around the axes. Irregular shaped clusters were not detected.

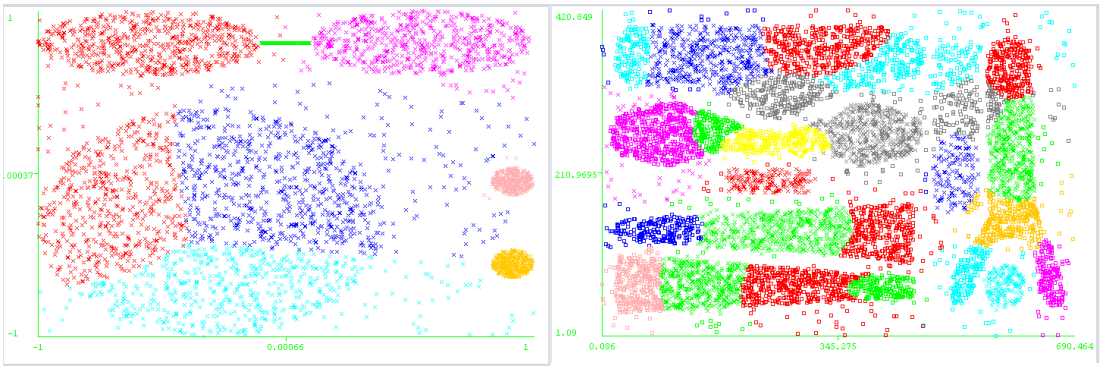


(a) CURE-T2-4K (b) CLUTO-T8-8K

Figure 4.10: Fuzzy C-Means Clustering

4.3.2.4 Model based clustering Algorithm

Performance of the Model based clustering algorithm was evaluated on the datasets with the final model configuration selected by 10-cross fold cross validation. The initial seeds were randomly assigned 12 times to avoid local optima.



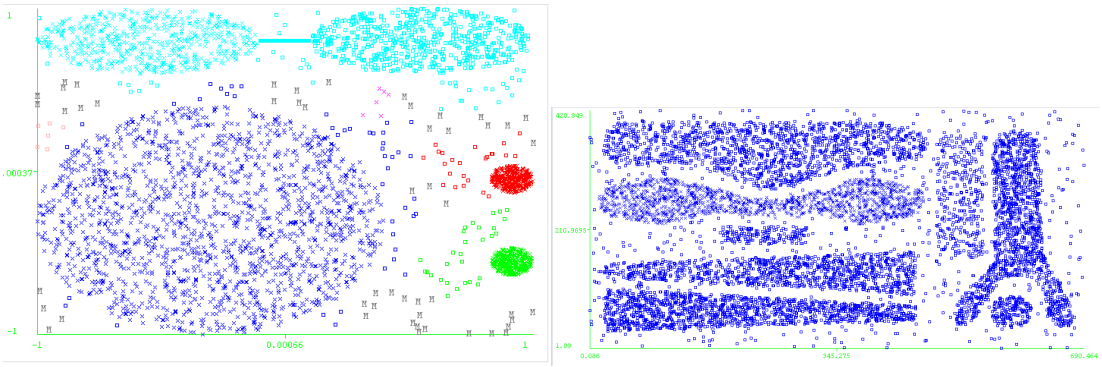
(a) CURE-T2-4K (b) CLUTO-T8-8K

Figure 4.11: Clustering result of Model based clustering Algorithm

The mixture model based clustering has detected 8 clusters in CURE-T2-4K and 24 clusters in CLUTO-T8-8K as seen in Fig. 2 and incorrectly clustered instances were 50.23% and 60.66% respectively. A disadvantage of this approach was specification of the number of clusters initially.

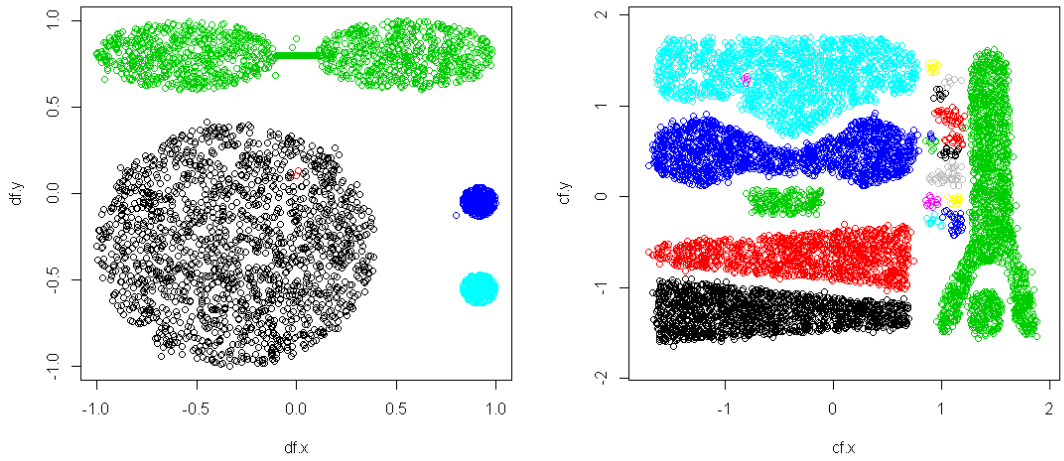
4.3.2.5 Density based Clustering Algorithms

DBSCAN is parameter dependent and detected clusters of irregular shapes. It handled noise effectively. The key drawback is finding the right values of ε and $MinPts$ for a particular dataset. The clusters having irregular densities and not well separated were merged in both datasets to give super clusters. OPTICS wasn't parameter sensitive and could identify clusters effectively.



(a) CURE-T2-4K (b) CLUTO-T8-8K

Figure 4.12: Clustering result of DBSCAN Algorithm

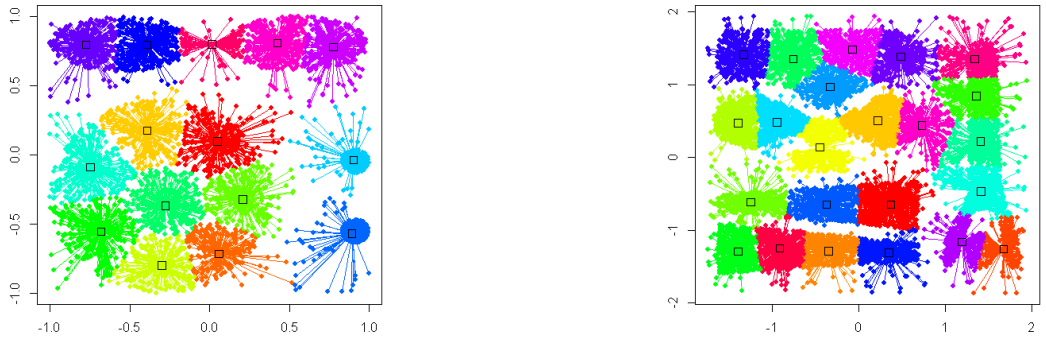


(a) CURE-T2-4K (b) CLUTO-T8-8K

Figure 4.13: Clustering result of OPTICS Algorithm

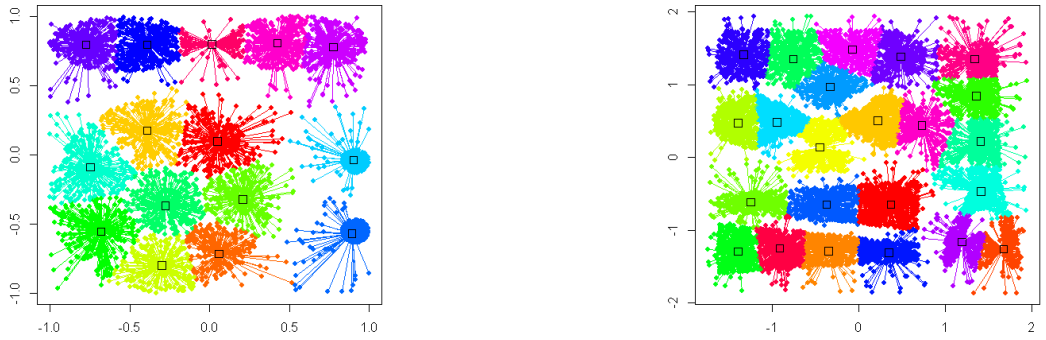
4.3.2.6 Affinity Propagation Clustering

Advantage of Affinity propagation clustering is that number of clusters need not be specified in advance. The tunable parameter q is input preference which can be interpreted as the tendency of a data sample to become an exemplar. Keeping the value minimum result in small number of clusters. In unsupervised setting, affinity propagation resulted in large number of clusters even with $q = 0$.



(a) CURE-T2-4K (b) CLUTO-T8-8K

Figure 4.14: Unsupervised Affinity Propagation Algorithm



(a) CURE-T2-4K (b) CLUTO-T8-8K

Figure 4.15: Supervised Affinity Propagation Algorithm

4.3.2.7 Spectral Clustering

The time complexity of spectral clustering is high O(n^3) and so results on CLUTO-T8-8K could not be evaluated. On CURE-T2-4K dataset the algorithm could not identify clusters connected by a dense region. Noise points too were included in the clusters.

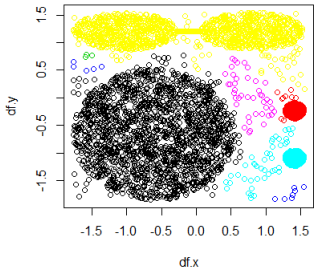


Figure 4.16: Spectral Clustering

4.3.3 Summary of Results

Table 4.2: Evaluation of clustering algorithms

Name	Time Complexity	Parameters	Detect Asym Clusters
KMeans	O(nkd)	1	No
KMeans++	O(nkd)	1	No
Kernel KMeans	O(nkd)	1	No
Hierarchical clustering	O(n^2logn)	1	No
Fuzzy CMeans	O(n)	2	No
Model based	O(knp)	2	No
DBSCAN	O($nlogn$)	2	Yes
OPTICS	O($nlogn$)	2	Yes
Affinity propagation	O(n^2T)	2	No
Spectral clustering	O(n^3)	2	No

4.4 Conclusion

Cluster detection poses a challenge to algorithms especially when underlying model for formation of community structure is not available. This is the case in most real world situations and hence there is ambiguity regarding defining the term "cluster". Ideally the approach to clustering should not require user interference, however all the current

clustering algorithms require parameter tuning and this could result in models that overfit the data and don't generalize well. The algorithms could not identify clusters in the benchmark datasets and had drawbacks like sensitivity to noise and outliers, high time and computational complexity and failure to detect clusters which were not well separated or of arbitrary shapes and densities.

Chapter 5

Community Detection using Node Attributes: A Non-Negative Matrix Factorization Approach.

5.1 Introduction

Exploratory Data Analysis [E.D.A.] is a domain on the intersection of fields such as machine learning, pattern recognition and information retrieval. The key goal of this domain is to generate effective summarizations, visualizations, information discovery and retrievals from data with a goal of reducing the exponential costs involved in its storage. The main task performed in E.D.A. is clustering analysis. Cluster analysis is a type of unsupervised learning as cluster labels are not provided apriori or rather are implicit in the data itself. The term "Cluster" doesn't have a standard definition and hence there is subjectivity in the deciding what forms a "Cluster". This has led to proliferation in the literature of clustering algorithms. Distance based definition of clusters have been explored and have created a family of techniques such as partition based clustering [26] , hierarchical clustering [48], mixture model based clustering [48], fuzzy clustering [18] amongst others. In contrast to a line of previous work a second definition of clusters was proposed based on density, this created popular algorithms such as DBSCAN, OPTICS [46].

A related field is Community Detection which involves identification of latent groups of entities in data. These groups correspond to autonomous regions in the network which are known to have a higher degree of homogeneity within its members than with members of other groupings in the same network. In Network sciences such sub groups are called communities and these are identified using network topology. A vast area of literature has uncovered several state of the art community detection algorithms that

aim to find such communities in un-directed as well as directed graphs [14] . This literature is based on concepts related to Information Theory or the trajectory of Random walks on graphs or the Map Equation [16] [56]. Apart from this, community detection also developed a concept called modularity and a new family of algorithms were developed that detected communities in graphs by optimizing modularity in a greedy manner [14] [18].

Latent Dirichlet Allocation was another concept that emerged and led to a new line of research on detecting communities by utilizing meta-data that is associated with the entities (nodes). This led to novel techniques that utilized the information about network topology along with meta data for obtaining generative models of networks [2] [33]. However even with such methods there were drawbacks such as the limited applicability, as they could not detect overlapping communities. A second drawback is that they assumed soft node-community memberships, which was not appropriate for modeling communities because they did not allow a node to have high membership strength to multiple communities simultaneously. Finally, such methods had a large time complexity and couldn't be scaled to graphs having more than 1000 nodes.

Non negative matrix factorization of the term document matrix was found to be effective in document clustering [51]. NMF was later extended to clustering by aiming to learn the adjacency matrix of a graph. NMF research did not pay attention to the interpretation of latent factors which are used to find out the matrix. This led to development of BIGCLAM which aimed to learn latent factors which the authors argued represented strengths of community affiliations of nodes. BIGCLAM and NMF both used community affiliation knowledge of the nodes so that their membership strengths could be estimated. While most existing literature [53] focuses either on using the meta data or entity annotation for improving the quality of community detection.

The paper is organized as follows. Section II briefly surveys related work. In Section III, the statistical model of the approach is defined, and in Section IV, the parameter fitting procedure is provided in detail. This is followed by describing experimental evaluation in Section V and the conclusion.

5.2 Related Work

Clustering approaches have their own biases in identifying clusters in data but none is considered a universal best fit. For example, the objective function to be minimized in the k-partitioning algorithms is variance or SSE i.e. Sum of Squared Distance. $SSE = \sum_{k=1}^{k} \sum_{x_i \epsilon c_k} \|x_i - c_k\|^2$ where c_k = centroid of the cluster. In this case the clusters are convex but the optimization function converges to a local optima as the objective function is non convex [14]. Hierarchical Clustering algorithms are a category of algorithms having a completely unsupervised approach to clustering. They do not

require the users to specify the number of clusters in advance and are broadly of two types: Agglomerative and divisive. To measure the dissimilarity between clusters obtained in hierarchical clustering, linkage methods were developed with several popular techniques being listed in the literature [14] [18] [48]. But it difficult to decide on a suitable linkage method and at times the selection of a distance measure too isn't clear.

Fuzzy clustering (FCM) minimizes the objective function given as $\sum_{j=1}^{k} \sum_{x_i \in C_j} u_{ij}^{m}(x_i - u_j)^2$ with μ being the fuzzifier and m defining the level of cluster fuzziness. The drawbacks seen in k-means such as non convex objective function, difficulty in detecting non linearly separable clusters etc. are also seen in FCM. The MST clustering algorithm discussed in [19] is known to be capable of detecting clusters with irregular boundaries. Unlike traditional clustering algorithms, the MST clustering algorithm does not assume a spherical shaped clustering structure of the underlying data. If the number of clusters k is given in advance, the simplest way to obtain k clusters is to sort the edges of the minimum spanning tree in descending order of their weights, and remove the edges with the first $(k-1)$ heaviest weights. Undesired clustering structures and an unnecessarily large number of clusters are problems commonly faced by MST based clustering.

In [3] Mixture Model based clustering is discussed which unlike the traditional clustering algorithms doesn't rely on heuristics but assumes that the data has been generated from a mixture of multiple probability distributions (Gaussian or multinomial) whose parameters have to be estimated. This is done using a technique called Expectation Maximization. Subspace clustering [14] [48] is based on key principle which is to discretize the data-space into grids and estimate the density by counting the number of points in a grid cell. Other methods in the literature are Affinity propagation [26] which is based on concept of message passing, Spectral clustering [14] in which the rst k eigenvectors $u_1, u_2, ..., u_k$ corresponding to the k smallest eigenvalues are computed to get matrix $U \in R^{n*k}$ which has $u_1, u_2, ..., u_k$ as columns. Then for $y_i \in R^k$ which is the i^{th} row of U, all rows are treated as points and clustered by k-means to get k clusters. DB-SCAN, OPTICS are based on the concept of density and treat clusters are dense regions connected by less dense regions. However, none of this literature is applicable for clustering in networks.

Community detection is a field that deals with obtaining coarse grained descriptions of large networks as real world graphs are too large to be analyzed efficiently. This is done by utilizing network topology to detect communities of nodes while ignoring node attributes. Topic Link LDA and Block LDA were the first to cluster graphs by jointly modeling links and node attributes. Topic link LDA aims to quantify the effect of topic similarity and community similarity to the formation of a link [2] [33]. Block LDA is a joint model of two components, with one that models links between pairs of entities represented as edges in a graph with a block structure, and the second that models text documents, through shared latent topics. There has also been limited work on com-

bining graph and content information for community discovery leading to techniques such as CESNA and BIGCLAM. CESNA was for statistically modeling the interaction between the network structure and the node attributes. The authors argued that this would lead to more accurate community detection as well as improved robustness in the presence of noise in the network structure [53]. BIGCLAM is another approach that detects both 2-mode as well as cohesive communities which may overlap or be hierarchically nested and is based on affiliation graph models.

To the best our knowledge, in the research no mention could be found of using the attributes of a data point for calculating the latent features on the basis of which communities shall be detected. The work in this paper is based on BIGCLAM framework but the critical difference with existing techniques such as CESNA, AGMFIT and BIGCLAM is that attributes shall be used instead of community affiliations. The intuition here is that attributes are useful in determining the cluster affiliations and as such this intuition is also consistent with the phenomenon of "homophily" that is seen in human networks.

5.3 Mathematical Model

The stochastic generative model for generating communities is presented in this section in which the probability of two entities in data being present in the same community is dependent on the attributes or the annotated text data associated with these nodes. An efficient model fitting procedure is the presented which allows for detecting communities in the network. The current work is based on the assumption that attributes of the data are categorical. The aim is to build upon BIGCLAM, an affiliation model for community detection, however the objective is using attribute information in place of affiliation information for building a bipartite graph which will be partitioned.

Directed Attribute Affiliation Model: BIGCLAM and AGMFIT are build on the Affiliation Graph Model based algorithms which use Maximum Likelihood Estimation to create a AGM from the network. Both however ignore importance of attributed of the nodes being responsible for community creation. In Social networks, Homophily is the tendency to be associated with others who share similar preferences and therefore attribute associated with the entity in a network play an important role in deciding communities. The hypothesis of correlation between attributes and communities is reasonable as its presence is also seen in empirical evidence provided in the literature [53]. Based on this reasoning, a simple conceptual model called Directed Attribute Affiliation Model is formulated. This builds on the family of affiliation network models, but in this work affiliation models are extended to consider attributes.

To represent node and attribute affiliation a bipartite affiliation graph is created where nodes are the bottom layer and attributes to which they belong are shown as the

top layer as seen in Fig.5.1. A directed edge is created between an attribute and a node if the node has the attribute present in it. Such a bipartite graph can be constructed easily if attributes are binary valued. In case the attributes are continuous or categorical, a different mechanism might be needed. In this paper only binary attributes are considered. Cluster affiliation can then be modeled using such a bipartite graph where directed edges are formed between nodes and attributes to denote that those nodes contain that attribute.

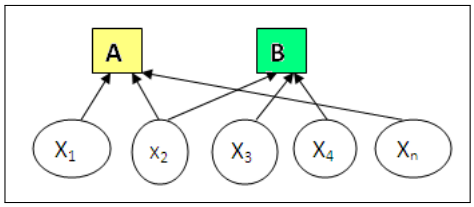


Figure 5.1: Bipartite Attribute Affiliation Graph

A Bipartite Attribute Affiliation Graph is denoted as $B(X, C, M)$, with X as the nodes, C as the attribute value and M denotes the directed edge from X to C if node X has attribute value C. The problem now is to create a set of communities $S = S_1, S_2, ..., S_k$ given $B(X, C, M)$. A parameter p_c is assigned to an attribute value $c \in C$. This is for calculating the probability that a node x_i has the attribute value c. This can also be called the probability that a node x_i belongs to the same community as another x_j having the value of a particular attribute as c. The $P_A(i, j)$ denotes that the nodes i, j belong to the same community A. This can be shown by the below equation.

$$P_A(i,j) = 1 - \prod_{c \in M_i \cap M_j} (1 - p_c) \tag{5.1}$$

Where,

- M_i = node i has membership to attribute value c.
- M_j = node j has membership to attribute value c.

In Eqn. 1, the value of $P_A(i, j)$ is set to ε, following the BIGCLAM procedure the value of ε can be set as $2|E|/|V|(|V| - 1)$ [21].

5.3.1 Calculate the latent weights of the attributes

Every attribute has its own importance or strength in determining the cluster to which the node should belong to, this is denoted here formally as F_{uC}. This is the strength that attribute C has for node u in determining its cluster. Considering this membership strength the Eqn. 1 can be modified as follows:

$$P_A(i,j) = 1 - exp(-F_{uC}.F_{vC}^T) \tag{5.2}$$

F_{uC} is the membership strength of a single attribute, similarly it is assumed that every node i has a attribute membership vector F_i which contains the membership strengths to all attributes in the data. The modified probability that nodes i, j now share a cluster is Eqn 2.

The intuition behind the above formula is simple, Consider a node having attribute values same as the attribute values of another node, in such a case the likelihood of both nodes belonging to a particular community increases. This means that for each attribute a pair of nodes shares we get an independent chance of grouping the nodes. Thus, naturally, the more attributes a pair of nodes shares, the higher the probability of sharing the same community and being connected.

If $M_u \cap M_v = 0$ then $P(u,v) = \varepsilon$ this is done to consider cases where nodes might not share attributes but still are connected. F_u is the vector that denotes the strengths of association of a node u with each attribute community in the network. The task is to find the matrix of memberships F that maximizes the likelihood of generating the graph $G(V,E)$. The log-likelihood of this is Eqn. 5. The Gradient update algorithm is used to find the value of F as shown in Eqn. 6

$$l(F) = \sum_{u,v \in E} log(1 - \exp(-F_u.F_v^T)) - \sum_{u,v \notin E} (F_u.F_v^T) \tag{5.3}$$

$$\bigtriangledown\, l(F_u) = \sum_{v \in N(u)} F_v \frac{\exp(-F_u.F_v^T)}{1 - \exp(-F_u.F_v^T)} - \sum_{v \notin N(u)} F_v \tag{5.4}$$

Decide Community Affiliation: The membership strengths matrix F is computed from above and the next step is to determine a suitable threshold above which it is possible to determine whether the node i belongs to a community. This threshold is δ set at $\sqrt{log(1-\varepsilon)}$ [21]. The initialization isn't done using locally minimal neighborhoods approach of BIG-CLAM [53] as entity annotated attributes are used to get initial values of the membership strengths F_i. The value of $F_{i,k}$ is 0 if attribute k is present and 0 if absent.

Choosing the number of communities: This is done by procedure specified in [53] where the model is trained using an initial value of K. Then we detect K communities on the 80% of node pairs and then evaluate the likelihood on the hold out set. The K with the best hold out likelihood is used.

5.4 Experiments

In this section, evaluation of the performance of the variant of BIG-CLAM and other state-of-the-art community detection methods is done. The data-set used in this work consists of an artificially generated network with node attributes created using the tool described in the work of Christine Largeron *et. al.* [31]. Community detection methods such as Louvain and Fastgreedy technique were not applied as the graph was directed.

5.4.1 Dataset and Evaluation Criteria

The Artificially generated data-set is described below. The comparison metrics used are Variation of information (VI), Normalized mutual information, Split-join distance, Rand index (RI) and Adjusted Rand index (ARI). NMI, RI and ARI are in the range (0 - 1) with higher value indicating better clustering. The split-join distance between partitions A and B is the sum of the projection distance of A from B and the projection distance of B from A and should be low. VI should also ideally have a low value.

Table 5.1: Description of the dataset

Sr. No	**Parameter**	**Value**
1	Node Attributes	3
2	Nodes	200
3	Communities	4
4	Observed homophily	0.74
5	Modularity	0.51
6	Avg. Clustering Coeff	0.33
7	Avg. Degree	5
8	Edges	500
9	Network Type	Directed-Unweighted

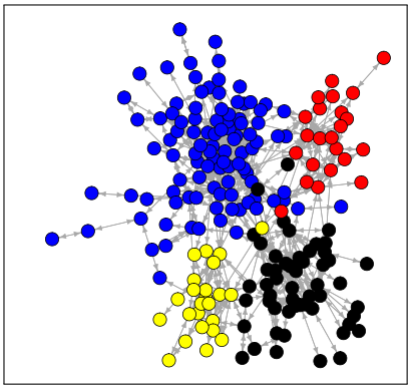


Figure 5.2: Artificially generated Network dataset

5.4.2 Experimental Results

5.4.2.1 InfoMap

"infomap.community" detected multiple communities in the network with few nodes leading to the conclusion that it split large clusters. The clustering is of poor quality as seen in Fig.5.3a as observed in the performance metrics.

Table 5.2: Results

Sr. No	Parameter	Value
1	Execution Time	0.6 secs
2	Modularity	0.567
3	Variation of information	1.72
4	Normalized mutual information	0.565
5	Split-join distance	115
6	Rand index	0.717
7	Adjusted Rand index	0.231
8	Detected Communities	22

5.4.2.2 Leading Eigenvector

"leading.eigenvector.community" has uncovered a community structure with less communities as seen in Fig.5.3b and its performance is good on the metrics.

Table 5.3: Results

Sr. No	Parameter	Value
1	Execution Time	0.53 secs
2	Modularity	0.551
3	Variation of information	0.88
4	Normalized mutual information	0.67
5	Split-join distance	74
6	Rand index	0.795
7	Adjusted Rand index	0.510
8	Detected Communities	6

5.4.2.3 Label Propagation

"label.propagation.community" has detected low number of communities as seen in Fig.5.4a. Performance of this technique is better than other approaches.

Table 5.4: Results

Sr. No	Parameter	Value
1	Execution Time	0.06 secs
2	Modularity	0.545
3	Variation of information	0.597
4	Normalized mutual information	0.778
5	Split-join distance	35
6	Rand index	0.91
7	Adjusted Rand index	0.81
8	Detected Communities	8

5.4.2.4 Walktrap

"walktrap.community" uncovers a community structure with higher modularity as seen in Fig.5.4b. Large number of small communities have been created and so this technique has low values on the performance metrics.

Table 5.5: Results

Sr. No	Parameter	Value
1	Execution Time	0.11 secs
2	Modularity	0.579
3	Variation of information	1.32
4	Normalized mutual information	0.548
5	Split-join distance	76
6	Rand index	0.777
7	Adjusted Rand index	0.446
8	Detected Communities	15

5.4.2.5 Spinglass and Clique Percolation

"spinglass.community" detects high number of communities and has low values on VI, Split join and ARI metrics. Hence clustering quality is low as seen in Fig.5.5a. Clique percolation with $k = 3$ detected community structure given in as seen in Fig.5.5b but 10% of the nodes were unclassified.

Table 5.6: Results

Sr. No	Parameter	Value
1	Execution Time	13.91 secs
2	Modularity	0.35
3	Variation of information	1.12
4	Normalized mutual information	0.638
5	Split-join distance	94
6	Rand index	0.772
7	Adjusted Rand index	0.419
8	Detected Communities	8

5.4.3 Community Structures

The figures below show the community structures uncovered by different approaches.

(a) infomap (b) Leading Eigenvector

Figure 5.3: Community structures in Network

(a) Label propagation (b) Walktrap

Figure 5.4: Community structures in Network

(a) Spinglass (b) Clique Percolation

Figure 5.5: Community structures in Network

5.4.4 Variation of BIGCLAM

With the above experimental results as baselines, the variation of BIGCLAM approach suggested in the paper was applied. The attributes were used to create a Bipartite graph

as given by Jure Lescovec *et. al* [53], however the key difference was that the nodes were now having edges with the attributes. Network topology was thus ignored and BIGCLAM now used the Non negative matrix factorization to calculate the strength of memberships of the nodes to the attributes. These would be used to decide memberships of nodes to communities. As seen in Fig.5.6 the hold out set has highest likelihood at $k = 6$. One advantage of this approach was large communities were formed and so it was inferred that splitting into smaller communities was avoided. As BIGCLAM is highly scalable, the network of 10^5 nodes can be processed efficiently.

Table 5.7: Results

Sr. No	Parameter	Value
1	Execution Time	8.51 secs
2	Maximum likelihood Estimate	-1070
3	Detected Communities	6

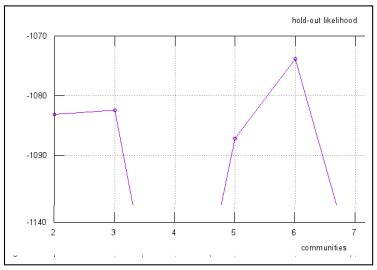


Figure 5.6: Clustering Structure using Variant of BIGCLAM

5.5 Conclusion

Community detection using a joint model of node attributes and network topology is a challenging task as additional information has to be factored while maintaining efficiency criteria. The model implemented in this paper uses Non negative matrix factorization on Bipartite Attribute Affiliation model for community detection. This approach allows for nodes to have high membership strengths simultaneously to various attribute communities. This allows for creating nested, overlapping and hierarchical communities in networks. The intuition behind this technique is that if the number of attributes shared by two nodes are high then the nodes have a higher probability of belonging to a single community. As this approach is relying on the optimization principle of BIGCLAM hence it is possible to state that this method can also be scaled to large networks efficiently.

Chapter 6

Summary

The primary focus of the thesis is to provide a description of Social networks and the evolution of the field of research in this domain from the time of conventional social networks which were created as people associated with a particular work interacted with each other to modern social networks that are established on the basis of virtual interactions. As geography is no longer a barrier and technology is becoming omnipresent such networks have become ubiquitous. The scale of such networks is very high as nodes (entities) involved are in the range of millions to even billion. This creates opportunities for research into areas which were available before. The availability of massive open source datasets, higher computation power and storage capability have made it possible to validate the hypotheses, on networks, proposed during the Pre Internet Age. It has also given researchers the opportunity to venture into fields such social influence analysis, expert prediction, community discovery etc. and create models that are statistically more robust.

6.1 Future Work

The past research in area of social network data clustering has focused on structural analysis (link structure) of social networks. Recent research argues that the inclusion of content information to the structural analysis can yield valuable insights about the underlying social network. This is an objective of future work proposed for community discovery algorithms. Evolutionary algorithms have an advantage over non evolutionary algorithms as they can traverse the search space more effectively. The comparison study of non evolutionary algorithms and evolutionary algorithms is the future work proposed. Identification of a suitable objective function (fitness function) for optimization is needed in evolutionary algorithms. The other research areas in the field of social network analytics are related to link prediction, expert discovery and node classification.

Chapter 7

Publications

1. **P Nerurkar**, S Bhirud "Modeling influence on a Social Network using Interaction Characteristics" , International Journal of Computer & Mathematical Sciences, Vol. 6, Issue 8, pp 152 - 160, presented at C-DAC, Mumbai, 2017.

References

[1] Charu C Aggarwal. An introduction to social network data analytics. *Social network data analytics*, pages 1–15, 2011. 1, 2, 3, 14, 25

[2] Ramnath Balasubramanyan and William W Cohen. Block-lda: Jointly modeling entity-annotated text and entity-entity links. In *Proceedings of the 2011 SIAM International Conference on Data Mining*, pages 450–461. SIAM, 2011. 46, 47

[3] James C Bezdek, Robert Ehrlich, and William Full. Fcm: The fuzzy c-means clustering algorithm. *Computers & Geosciences*, 10(2-3):191–203, 1984. 29, 47

[4] Vincent D Blondel, Jean-Loup Guillaume, Renaud Lambiotte, and Etienne Lefebvre. Fast unfolding of communities in large networks. *Journal of statistical mechanics: theory and experiment*, 2008(10):P10008, 2008. 18

[5] Paul S Bradley, Usama M Fayyad, Cory Reina, et al. Scaling clustering algorithms to large databases. In *KDD*, pages 9–15, 1998. 27

[6] Olivier Chapelle, Bernhard Scholkopf, and Alexander Zien. Semi-supervised learning (chapelle, o. et al., eds.; 2006)[book reviews]. *IEEE Transactions on Neural Networks*, 20(3):542–542, 2009. 2, 26

[7] Wei Chen, Chi Wang, and Yajun Wang. Scalable influence maximization for prevalent viral marketing in large-scale social networks. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1029–1038. ACM, 2010. 3, 4

[8] Wenlong Chen, Shaoyin Cheng, Xing He, and Fan Jiang. Influencerank: An efficient social influence measurement for millions of users in microblog. In *Cloud and Green Computing (CGC), 2012 Second International Conference on*, pages 563–570. IEEE, 2012. 6

[9] Aaron Clauset, Mark EJ Newman, and Cristopher Moore. Finding community structure in very large networks. *Physical review E*, 70(6):066111, 2004. 17

[10] Aaron Clauset, Cristopher Moore, and Mark EJ Newman. Hierarchical structure and the prediction of missing links in networks. *arXiv preprint arXiv:0811.0484*, 2008. 17

[11] Inderjit S Dhillon, Yuqiang Guan, and Brian Kulis. Kernel k-means: spectral clustering and normalized cuts. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 551–556. ACM, 2004. 27

[12] Martin Ester, Hans-Peter Kriegel, Jörg Sander, Xiaowei Xu, et al. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Kdd*, volume 96, pages 226–231, 1996. 32

[13] Vladimir Estivill-Castro. Why so many clustering algorithms: a position paper. *ACM SIGKDD explorations newsletter*, 4(1):65–75, 2002. 2

[14] Adil Fahad, Najlaa Alshatri, Zahir Tari, Abdullah Alamri, Ibrahim Khalil, Albert Y Zomaya, Sebti Foufou, and Abdelaziz Bouras. A survey of clustering algorithms for big data: Taxonomy and empirical analysis. *IEEE transactions on emerging topics in computing*, 2(3):267–279, 2014. 1, 14, 26, 46, 47

[15] Hossein Fani and Ebrahim Bagheri. Community detection in social networks. *Encyclopedia with Semantic Computing and Robotic Intelligence*, 1(01):1630001, 2017. 14, 15

[16] Santo Fortunato and Darko Hric. Community detection in networks: A user guide. *Physics Reports*, 659:1–44, 2016. 3, 15, 46

[17] Brendan J Frey and Delbert Dueck. Clustering by passing messages between data points. *science*, 315(5814):972–976, 2007. 32

[18] Michelle Girvan and Mark EJ Newman. Community structure in social and biological networks. *Proceedings of the national academy of sciences*, 99(12):7821–7826, 2002. 3, 15, 45, 46, 47

[19] Oleksandr Grygorash, Yan Zhou, and Zach Jorgensen. Minimum spanning tree based clustering algorithms. In *Tools with Artificial Intelligence, 2006. ICTAI'06. 18th IEEE International Conference on*, pages 73–81. IEEE, 2006. 47

[20] Sudipto Guha, Rajeev Rastogi, and Kyuseok Shim. Rock: A robust clustering algorithm for categorical attributes. In *Data Engineering, 1999. Proceedings., 15th International Conference on*, pages 512–521. IEEE, 1999. 28

[21] Sudipto Guha, Rajeev Rastogi, and Kyuseok Shim. Cure: an efficient clustering algorithm for large databases. *Information Systems*, 26(1):35–58, 2001. 28

[22] Behnam Hajian and Tony White. Modelling influence in a social network: Metrics and evaluation. In *Privacy, Security, Risk and Trust (PASSAT) and 2011 IEEE Third Inernational Conference on Social Computing (SocialCom), 2011 IEEE Third International Conference on*, pages 497–500. IEEE, 2011. 4

[23] Jiawei Han, Jian Pei, and Micheline Kamber. *Data mining: concepts and techniques*. Elsevier, 2011. 2, 26

[24] Prantik Howlader and KS Sudeep. Degree centrality, eigenvector centrality and the relation between them in twitter. In *Recent Trends in Electronics, Information & Communication Technology (RTEICT), IEEE International Conference on*, pages 678–682. IEEE, 2016. 3

[25] Yanqing Hu, Hongbin Chen, Peng Zhang, Menghui Li, Zengru Di, and Ying Fan. Comparative definition of community and corresponding identifying algorithm. *Physical Review E*, 78(2):026121, 2008. 15

[26] Anil K Jain. Data clustering: 50 years beyond k-means. *Pattern recognition letters*, 31(8):651–666, 2010. 1, 25, 26, 30, 45, 47

[27] Li-Jen Kao, Yo-Ping Huang, and Frode Eika Sandnes. Mining time-dependent influential users in facebook fans group. In *Systems, Man, and Cybernetics (SMC), 2016 IEEE International Conference on*, pages 000718–000723. IEEE, 2016. 6

[28] Leonard Kaufman and Peter J Rousseeuw. *Finding groups in data: an introduction to cluster analysis*, volume 344. John Wiley & Sons, 2009. 27

[29] Alexy Khrabrov and George Cybenko. Discovering influence in communication networks using dynamic graph analysis. In *Social Computing (SocialCom), 2010 IEEE Second International Conference on*, pages 288–294. IEEE, 2010. 4

[30] Andrea Lancichinetti, Santo Fortunato, and Filippo Radicchi. Benchmark graphs for testing community detection algorithms. *Physical review E*, 78(4):046110, 2008. 20

[31] Christine Largeron, Pierre-Nicolas Mougel, Reihaneh Rabbany, and Osmar R Zaïane. Generating attributed networks with communities. *PloS one*, 10(4): e0122777, 2015. 51

[32] Jianxin Li, Timos Sellis, J Shane Culpepper, Zhenying He, Chengfei Liu, and Junhu Wang. Geo-social influence spanning maximization. *IEEE Transactions on Knowledge and Data Engineering*, 2017. 6

[33] Yan Liu, Alexandru Niculescu-Mizil, and Wojciech Gryc. Topic-link lda: joint models of topic and author community. In *proceedings of the 26th annual international conference on machine learning*, pages 665–672. ACM, 2009. 46, 47

[34] James MacQueen et al. Some methods for classification and analysis of multivariate observations. In *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, volume 1, pages 281–297. Oakland, CA, USA., 1967. 26

[35] Mark EJ Newman. Fast algorithm for detecting community structure in networks. *Physical review E*, 69(6):066133, 2004. 17

[36] Mark EJ Newman. Finding community structure in networks using the eigenvectors of matrices. *Physical review E*, 74(3):036104, 2006. 19

[37] Mark EJ Newman and Michelle Girvan. Finding and evaluating community structure in networks. *Physical review E*, 69(2):026113, 2004. 17

[38] Raymond T. Ng and Jiawei Han. Clarans: A method for clustering objects for spatial data mining. *IEEE transactions on knowledge and data engineering*, 14(5): 1003–1016, 2002. 27

[39] Pascal Pons and Matthieu Latapy. Computing communities in large networks using random walks. *J. Graph Algorithms Appl.*, 10(2):191–218, 2006. 16

[40] Xueming Qian, He Feng, Guoshuai Zhao, and Tao Mei. Personalized recommendation combining user interest and social circle. *IEEE transactions on knowledge and data engineering*, 26(7):1763–1777, 2014. 4

[41] Usha Nandini Raghavan, Réka Albert, and Soundar Kumara. Near linear time algorithm to detect community structures in large-scale networks. *Physical review E*, 76(3):036106, 2007. 17

[42] Adithya Rao, Nemanja Spasojevic, Zhisheng Li, and Trevor DSouza. Klout score: Measuring influence across multiple social networks. In *Big Data (Big Data), 2015 IEEE International Conference on*, pages 2282–2289. IEEE, 2015. 3, 4

[43] Jörg Reichardt and Stefan Bornholdt. Statistical mechanics of community detection. *Physical Review E*, 74(1):016110, 2006. 18

[44] Fabián Riquelme and Pablo González-Cantergiani. Measuring user influence on twitter: A survey. *Information processing & management*, 52(5):949–975, 2016. 5

[45] Martin Rosvall and Carl T Bergstrom. Maps of random walks on complex networks reveal community structure. *Proceedings of the National Academy of Sciences*, 105 (4):1118–1123, 2008. 15

[46] Gholamhosein Sheikholeslami, Surojit Chatterjee, and Aidong Zhang. Wavecluster: A multi-resolution clustering approach for very large spatial databases. In *VLDB*, volume 98, pages 428–439, 1998. 30, 31, 45

[47] Karthik Subbian, Dhruv Sharma, Zhen Wen, and Jaideep Srivastava. Finding influencers in networks using social capital. *Social Network Analysis and Mining*, 4(1):219, 2014. 3, 4, 5

[48] John W Tukey. Exploratory data analysis. 1977. 1, 2, 3, 26, 45, 47

[49] Ulrike Von Luxburg. A tutorial on spectral clustering. *Statistics and computing*, 17(4):395–416, 2007. 33

[50] Jianshu Weng, Ee-Peng Lim, Jing Jiang, and Qi He. Twitterrank: finding topic-sensitive influential twitterers. In *Proceedings of the third ACM international conference on Web search and data mining*, pages 261–270. ACM, 2010. 5

[51] Wei Xu, Xin Liu, and Yihong Gong. Document clustering based on non-negative matrix factorization. In *Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval*, pages 267–273. ACM, 2003. 46

[52] Jaewon Yang and Jure Leskovec. Community-affiliation graph model for overlapping network community detection. In *Data Mining (ICDM), 2012 IEEE 12th International Conference on*, pages 1170–1175. IEEE, 2012. 32

[53] Jaewon Yang and Jure Leskovec. Overlapping community detection at scale: a nonnegative matrix factorization approach. In *Proceedings of the sixth ACM international conference on Web search and data mining*, pages 587–596. ACM, 2013. 32, 46, 48, 50, 56

[54] Xiwang Yang, Harald Steck, and Yong Liu. Circle-based recommendation in online social networks. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1267–1275. ACM, 2012. 3

[55] Tian Zhang, Raghu Ramakrishnan, and Miron Livny. Birch: an efficient data clustering method for very large databases. In *ACM Sigmod Record*, volume 25, pages 103–114. ACM, 1996. 28

[56] Jianke Zhu, Hao Ma, Chun Chen, and Jiajun Bu. Social recommendation using low-rank semidefinite program. In *AAAI*, pages 158–163, 2011. 3, 46