Quantitative Prediction of Twitter Message Dissemination: A Machine Learning Approach

Farhad Sarabchi

Master of Science Thesis

Pattern Recognition Lab
Department of Media and Knowledge Engineering
Faculty of Electronic and Engineering, Mathematics and Computer Science
Delft University of Technology

TU Delft Delft University of Technology

Intelligent Systems (INSY)

Quantitative Prediction of Twitter Message Dissemination: A Machine Learning Approach

Master of Science Thesis

For the degree of Master of Science in
Pattern Recognition Lab
at Department of Media and Knowledge Engineering
at Delft University of Technology

Farhad Sarabchi

Faculty of Electronic and Engineering, Mathematics and Computer Science
Delft University of Technology
Delft, The Netherlands

Delft University of Technology
Department of
Media and Knowledge Engineering

The undersigned hereby certify that they have read and recommend to the Faculty of Electronic and Engineering, Mathematics and Computer Science for acceptance a thesis entitled

Quantitative Prediction of Twitter Message Dissemination: A Machine Learning Approach

by

Farhad Sarabchi

in partial fulfillment of the requirements for the degree of

Master of Science.

Dated: _

Supervisor: _____

Dr. D.M.J. Tax

Readers: _____

Prof.dr.ir. M.J.T. Reinders

_____

Dr. E.A. Hendriks

Dr.ir. S.E. Verwer

Master Thesis
Quantitative Prediction of Twitter Message Dissemination: A Machine Learning Approach

Farhad Sarabchi

Pattern Recognition Lab
Media and Knowledge Engineering
Delft University of Technology

Abstract

Predicting the popularity of contents in social networks is quite important for several applications such as viral marketing, news propagation and personalization. In this work, we developed an statistical learning approach to predict the popularity of tweets in the twitter social network. We extracted several user-based, tweet-based and network-based features from each tweet and adopted several classifiers to predict the popularity of tweets. We model this problem with a binary classification problem where popular tweets are considered as the positive and non-popular tweets are considered as the negative class. Popularity is defined by a threshold which indicates how many time a tweet is retweeted. We defined several popularity thresholds and examined the performance of different classifiers based on different threshold values. Our experimental results show that there is no global best classifier for the problem of popularity prediction in twitter but depending on the dataset, popularity threshold and our interest, we can adopt an optimal classifier with a proper set of features for this task.

Categories and Subject Descriptors:
Social Network analysis
Machine Learning

Key words: Twitter, Popularity Prediction, Social Networks, Classification, Feature Extraction, Microblogging

Table of Contents

1 Introduction **1**

1-1 Motivation 2

1-2 Research Questions and Contributions to this Work 4

1-3 Structure of this Thesis 5

2 Related Work **7**

2-1 Introduction 7

2-2 Information Dissemination in Social Networks 7

2-3 Predicting the Popularity of Content in Social Networks 8

2-3-1 Popularity Prediction in Twitter 9

2-3-2 Popularity Predictions and Recommendations 12

2-3-3 Popularity Predictions and Influences 12

2-4 Social Network Prediction Applications 15

2-5 Summary 16

3 Predictive Model **17**

3-1 Introduction 17

3-2 Prediction Challenges 18

3-3 Model Architecture 18
3-4 Classification Methods 19
3-4-1 Linear and Quadratic discriminant Classifiers 21
3-4-2 Naive Bayes Classifier 22
3-4-3 Distance-based Classifiers 23
3-4-4 Support Vector Machine (SVM) 24
3-5 Features 26
3-5-1 Tweet Features 27
3-5-2 User Features 27
3-5-3 Network Features 29
3-5-4 Combination of Features 31
3-6 Summary 32

4 Experimental Results **33**
4-1 Introduction 33
4-2 Dataset 34
4-2-1 Twitter structure 34
4-2-2 Dataset Collection 39
4-2-3 Relational Database Creation 40
4-2-4 Specification of Datasets 40
4-2-5 Splitting Methods of Datasets 42
4-3 Implementation of Classifier 45
4-3-1 Evaluation Metrics 45
4-3-2 Classifier Parameters Setup 50
4-4 Summary 61

5 Conclusions and Future Works **63**
5-1 Conclusions 63
5-2 Future Work 65

Acknowledgement

First of all I would like to thank my family for their kind and great support to do my master study in TU Delft. I would also like to thank my student counselor, John Stals, for his great help and advices; my supervisor, Dr. D.M.J. Tax, for his kind support; Drs. D.E. Butterman-Dorey, for helping me to edit my thesis; and Dr.Christian Doerr for his technical advices.

Chapter 1

Introduction

Due to great success of the Online Social Network (OSN), a large number of people are now utilizing OSN services in order to gain active collaboration, participation and interaction within their communities with other users. Twitter, the largest microblogging online service, has gained significant attention in the past few years. Users share and discuss everything in this social network.

Microblogging is a content-oriented concept in which people can interact with others both known and unknown. Twitter, which is a successful microblogging social network, has gained enormous popularity in recent years. As of March 2013, twitter has over 1 billion users and an average of 500 million tweets per day [1].

In twitter you are restricted to writing messages of no more than 140 characters these are then turned into short messages. These small messages create substantial information dissemination in the network and make twitter a successful social network for content dissemination.

The dissemination of a tweet in the network depends on different factors. One of the factors that contributes considerably to the propagation of the posts is users. Not all users can equally influence the propagation of tweets. *Influential users* are, however, the users whose contents propagate more successfully. Influential users are quite important to the analyzing and managing of propagation in tweeter social networks.

[1] http://expandedramblings.com/index.php/march-2013-by-the-numbers-a-few-amazing-twitter-stats

In this project we have developed a learning-based approach to twitter to discover why and how some particular tweets become popular. We shall further investigate this network to see to what extent we can predict the popularity of tweets.

1-1 Motivation

In twitter more than 19% of the tweets are about organizations or product brands, less than 20% of which are shown to have significant sentiment (Yu and Kak, 2012). Predicting the tweets which are likely to stimulate users' interests can improve the sale and marketing of different products and brands. Online advertisements could use such predicted messages to efficiently target the locations of networks which are visited the most. Moreover successful predictions can also increase user satisfaction by providing them with more attractive contents. Media companies could learn how to effectively generate buzzes for new films and shows. In political campaigning, groups could learn who they should target in order to successfully spread their message .

Predicting the popularity of content in twitter is also quite important for several other purposes such as viral marketing, popular news detection, personalized message recommendation and trend analysis. Users with many connections can suffer from information overload. It is quite important to filter information flow for the end users and to provide them with important tweets. Popularity prediction is also helpful in personalizing the content and finding the right tweets for end users. On the other hand, understanding how and why a tweet becomes popular, can help to gain a better insight into how the information is dispersed over the network. In the case of marketing, predicting popular tweets is quite useful for determining what are the trending topics and products.

In this work we developed an automatic learning-based approach to predict the popularity of content in tweeter. Automatic prediction with machine power has much lower costs compared to human-based work (Bothos et al., 2010). Furthermore, automatic approaches can scale to very large datasets which would be impossible to manage with human-based classifications.

The problem of popularity prediction in twitter has been studied in some previous works. In most of the recent works the popularity of tweets is defined as the number of retweets since retweeting is potentially the most effective way to disseminate messages due to its viral nature. Such popularity can

also be measured by the number of replies made to each tweet as well as the number of times that a tweet is favored by other users. We shall, however, measure the popularity of tweets in terms of the retweet count for the following reasons: 1) People are more likely to retweet a tweet rather than favor it. A tweet (eg. bad news) can be tweeted many times without getting any favorites. We therefore think that the favorite count is not a good indication of the popularity of tweets. 2) The reply count is not a good measure of the popularity of tweets either because not all tweets are *conversational* tweets. A tweet can trigger lots of replies while getting very few retweets and not spreading widely in the network.

Previous works on popularity predictions can be classified to either approaches that rely on the content of message (Hong et al., 2011b; Suh et al., 2010; Tsur and Rappoport, 2012a) or approaches that rely on the social characteristics of the network (Artzi et al., 2012). There are also some studies that try to predict the popularity of content by analyzing the *influential* users (DeRue and Ashford, 2010).

Hong et al. (2011b) and Suh et al. (2010) approach the popularity prediction problem by extracting features from *content* and *metadata* of the messages and trying to predict which messages will attract high numbers of retweets. In similar recent work, Zhang et al. (2014) developed a similar approach while considering different weight values for different features. Tsur and Rappoport (2012b) developed another feature-based approach which predicts the popularity of the contents in twitter by means of a linear regression-based approach.

Artzi et al. (2012) developed a model that is able to predict the likelihood that a tweet will be retweeted. They used a combination of tweet features as well as features from the entire network for the prediction task.

In a very recent work, Zaman et al. (2013) developed a Bayesian network-based approach which measured the popularity of tweets based on only the retweet times and the network structure of retweets in a five minute time window after tweeting. In our experiments, however, we will show that more than 40% of the retweets will occur in the first five minutes after tweeting which is already a significant number of retweets. In this work, we shall consider the retweet counts in different window sizes as different features and study the influence of each feature on the performance of the prediction task. Our approach also differs from previous studies in the sense that we do a binary classification task to see if a tweet becomes popular or not, In the work Zaman et al. (2013) they tried, by contrast to predict the retweeting activity over the course of time.

The previous learning-based works are mainly based on trial and error which trains classifiers based on a limited set of features in a specific dataset. None of the above learning-based studies analyze the predictors to see why they can predict the popularity of content, nor can they prove that their approach is generalized enough to be applied to any dataset.

In this work, however, we shall do an *in-depth* analysis on the different features that we extract from tweets. We not only introduce some new features that improve the performance of the prediction, but we also perform an in-depth analysis on each single feature, on different classifiers, to gain insight into the contributions and limitations of each single feature. Furthermore, we developed our experiments under different conditions (such as extreme data imbalance data, different time periods and different thresholds for the popularity definition) to generalize the model as much as possible.

1-2 Research Questions and Contributions to this Work

From a high level point of view we are interested in seeing how the information is disseminated in the twitter social network. This task is mainly done by predicting which tweets do actually become popular within the network.

The popularity prediction problem is to some extent similar to the recommender systems problem (Yu and Kak, 2012). With the problem of recommendation, the system tries to predict which content would be interesting for a user based on his past history. In contrast, our problem focuses on predicting popular content and information dissemination regardless of the interest of individual users. While the problem of popularity prediction is different from recommendations, it can also be useful for recommender systems in the sense that popular content can be recommended to users. There have also been studies which have showen that combining popularity-based recommender systems with personalized recommendations can provide the best recommendations (Jonnalagedda and Gauch, 2013).

In this work we developed different statistical learning-based classifiers that predict whether a tweet will be popular or not.

We extracted different tweet-based and user-based features and studied the contributions and limitations of each single feature in detail. We conducted our experiments with a set of *generative* and *discriminative* classifiers and explored the advantages and limitations of each classifier. The experiments were carried out on different datasets under various conditions to make sure that the models are generalized enough.

More specifically, we are interested in the following research questions:

- Can we predict whether a tweet will be popular or not and, if so, to what extent can we predict the popularity of tweets.
- What kind of features can be extracted from a tweet to predict its popularity.
- What are the most informative features for popularity predictions.
- What are the contributions and limitations of each individual feature.
- Which machine-learning approaches can best model the popularity prediction problem.
- For what kinds of tweets is the learning-based approach successful and for what kinds of tweets does the model fails to predict such popularity.

1-3 Structure of this Thesis

This thesis is organized as follows: in Chapter 2 we discuss the work related to our research. The popularity prediction problem in twitter and our approach is explained in Chapter 3. In Chapter 4 we explain, in some detail, the datasets and experiments that we have conducted. Finally we draw some conclusions and discuss the possible future directions in chapter 5.

Chapter 2

Related Work

2-1 Introduction

The problem of popularity prediction in social networks has always been widely studied. This problem has not only been studied in conjunction with twitter, but also in connection with other social networks. In this chapter we provide an overview of the existing approaches to popularity prediction in social networks, we discuss the related work and we elaborate on the advantages and limitations of existing methods.

2-2 Information Dissemination in Social Networks

A growing line of research has been followed on information dissemination through social networks. These studies propose that network cascades can play an important role as mediums for the dissemination of various information. These studies tend to be based on the idea that the information is spread by various infection mechanisms (Granovetter, 1978; Kempe et al., 2003).

Under the same category, Kempe et al. (2003) studied a combinatorial optimization problem sometimes known as the *influence maximization problem*. The problem involves finding a small set of seed nodes in social networks to target initial activation so that the largest expected spread of information

can be yielded . However, the exact computation of information cascades is an NP-hard problem (Chen et al., 2010).

Information diffusion has been studied in several online social networks, such as Flicker (Cha et al., 2008) and Digg (Lerman and Galstyan, 2008). The information propagation problem has also been studied in the Twitter social network. In recent work Galuba and Aberer (2010) characterize and model the propagation of URLs in Twitter. They exploit content popularity, user influence and the rate of propagation to model the propagation of URLs in the network on the basis of *linear threshold* models. Yang and Counts (2010) studied information diffusion networks on twitter through mentioned network. They generated a novel model to capture the three general properties of information diffusion: speed, scale, and range. Romero et al. (2011) performed an experimental study on twitter to explore how different types of information actually spread over the network.

2-3 Predicting the Popularity of Content in Social Networks

Due to the advent of web 2.0, user-generated content has increased dramatically. There are various types of contents that can be generated by users, such as comments and reviews on photos, movies and products. Most of these web 2.0 services connect the user with other users through social network, thus producing a social graph. For instance, in microblogging services such as Twitter this social graph is called a *follower network*. Any content generated from a user becomes visible to all of his/her followers and each of these contents has the chance to be re-posted by these followers who subsequently disperse the content over the social network. Re-posting, commonly known as retweeting, gives post the chance to become popular.

The problem of popularity prediction in social networks has been widely studied. In this section we explain this problem in different domains. In a recent study Szabo and Huberman (2010) used two content sharing portals Youtube and Digg to demonstrate how by monitoring responses to the stories, they can predict the popularity of such stories with remarkable accuracy. In another study, Lerman and Galstyan (2008) examined the role of social networks in promoting content on Digg. They discovered that patterns of the spread of interest in a story on the network are indicative of how popular the story will become.

In another domain, Leskovec et al. (2006) considered information cascades in the context of large person-to-person recommendation networks and studied the patterns of cascading that arise in large social networks. Watts and Dodds (2007) added other key factors that can determine influence, (i) the interpersonal relationships between ordinary users (ii) the readiness of a society to adopt an innovation. This modern view on influence leads to many marketing strategies, such as collaborative filtering which is a technique used by some recommender systems.

2-3-1 Popularity Prediction in Twitter

Due to the popularity of the twitter microblogging service there have been many studies on twitter. A great amount of work has been done to predict the popularity of tweets in this network.

In this section we first justify why users do retweeting in twitter by reviewing the related literature and then we briefly explain the related work on popularity prediction on twitter social network.

Understanding how users tweet and their motivations for tweeting is potentially important for predicting whether a tweet will be popular or not. In fact discovering what contents users choose to retweet can help to explain why a particular tweet becomes popular. The motivations for the act of retweeting are well explored in the study done by Boyd et al. (2010). They highlighted the mains reasons for retweeting as given by users. They introduced 10 different motivations for retweeting such as commenting on tweets, propagating tweets to new audiences, to inform specific persons or groups and to save tweets for future personal access. Although the focus of their study is not to predict the popularity of tweets, the underlying motivations of retweeting that they found can suggest which features to extract from tweets to predict their popularity.

Another exploratory study has been done by Suh et al. (2010) to find out the factors that lead to retweeting. They extracted three *latent* factors from tweet features using the Principal Component Analysis (PCA) approach and tried to associate it with real features. They then introduced a linear model to find the degree of popularity of retweets. They did not however motivate their choice of linear model for the prediction task nor did they discuss whether PCA is an effective approach for deriving the important factors of retweetability. Moreover, they only performed their experiments on a limited set of data as expressed by themselves. They concluded that content-based features such as hashtag and url greatly contribution to retweetability.

This conclusion was challenged by later studies Petrovic et al. (2011), which showed that content features are not informative enough to predict the popularity of tweets.

In a similar study, Petrovic et al. (2011) performed experimental work to predict whether a tweet will be retweeted or not. They developed an online learning-based algorithm (Crammer et al., 2006) to make the prediction as quickly as possible. They trained a set of local models merely on different subsets of data which are generated based on the time of the day to be able to better exploit the time information of tweets. As in the study of (Suh et al., 2010), they did not motivate their choice of model and not did they examine their model according to different datasets. But they compared the performance of their online-learning approach with human-based predictions and discovered that their method perform as well as human-based predictions.

Zaman et al. (2010) also performed a popularity prediction study based on the *collaborative filtering* approach. Unlike other studies that use features directly extracted from tweets or users, they incorporate implicitly positive and negative feedback into their model. If the *active* follower users retweet a tweet, it is considered as positive feedback and otherwise it will be considered as negative feedback. One drawback of this study though, is that they train the models based on at least one hour of data after a tweet has been published. On the other hand earlier studies [1] show that more than 90% of the retweets take place within the first hour after tweeting. Thus it is not practically worthwhile to train a model based on such a long time interval.

Artzi et al. (2012) developed a discriminative model to predict the likelihood of retweeting a tweet. They extracted several historical and lexical features from the text of the tweet and some social features from the user publishing the tweets. They adopted two different classifiers, the Multiple Additive Regression-Tree (Wu et al., 2008) and the Maximum Entropy classifier, for their prediction task. Their study mainly focused on the content features and thus as a limitation, their work is only adopted for English tweets. Moreover, they did not exploit features, such as hash tag, which are potentially useful for the popularity prediction of tweets.

Zaman et al. (2013) developed a Bayesian-based approach to predict the number of retweets for a given tweet based on its early spreading pattern. They approached the popularity prediction problem by studying the pattern of spread of tweets. They found that the reaction times to the tweets can be well estimated by adopting a log normal distribution. They introduced

[1] http://www.sysomos.com/insidetwitter/engagement/

a Bayesian network to model the evolution of retweets in terms of time. Unlike other studies which are mainly feature-based learning models, this study does not extract any features directly from tweets or the user. In fact the prediction is only based on the early spreading patterns of tweets. They claim that their approach works well when at least 10% of the retweets of a tweet are observed. This is less interesting for us firstly because it is not clear when 10% of tweets are known, and secondly because we are interested in predicting the popular tweets before they get published or very shortly after their publishing.

In a more recent study, Zhang et al. (2014) proposed a feature-weighted model that predicts the popularity of tweets in terms of the number of potential retweets. Despite other works, this work is a multiple classification task in which a tweet will be assigned to one of the four possible classes. The classes are: 0: not retweeted, 1: retweeted less than 10 times, 2: retweeted less than 100 times and 3: retweeted more than 100 times. Their feature extraction model, extracts a set of features from the tweet itself and from the user who published the tweet. Despite the study of (Artzi et al., 2012), their focus is more on the social features. They adopted a Support Vector Machine (SVM) classifier with a Radial Basis Function (RBF) kernel which allows their model to create complex boundaries to distinguish the classes. Their weighted mechanism assigns a weight to each of the features, which is calculated on the basis of the *Information Gain* of each single feature. This mechanism assigns a higher weigh value to the feature which has more information gains, thus making them contribute more to the classification task. The weight values were obtained based on a experimental evaluation on their dataset. Although their approach is reported to outperform the non-weighted approach, the authors have not provided enough evidence that this approach is also optimal for other datasets and under other settings.

A similar study was done by (Hong et al., 2011a) to predict the popularity of tweets. They formulate this task according to two different binary and multiple classification problems. The binary classification task sets out to predict whether a message will be retweeted or not, and the multiple classification task tries to predict the volume of retweets of a tweet after a tweet has been published. They adopted the TF-IDF mechanism to process the content features but they did not clearly specify what kind of classifier they used and which features contribute most to the classification task. Their approach also suffers from a lack of generality since they only tested their method on a limited dataset.

Most of the related work on popularity prediction in twitter was carried out

on a limited dataset with a limited number of settings. To our knowledge there are no studies which examine the contribution of individual features to popularity prediction. In this work we are studying the contribution of each individual feature to predicting the popularity of tweets. We also introduced some new features that have not been used in previous works. We also performed a comprehensive set of experiments on different datasets to consolidate our findings. Our approach and the features are described in Chapters 3 and 4.

2-3-2 Popularity Predictions and Recommendations

The problems of popularity predictions and recommendations are similar in some aspects. Both problems try to identify *influential* contents. While in popularity prediction problems the focus is more on the popularity of content, in recommender systems the focus is more on the user, the goal being to recommend the items to a user which satisfy him the most.

Predicting the popularity of contents can be quite useful in connection with making recommendations. Jonnalagedda and Gauch (2013) showed that a hybrid popular-based and personalized-based recommender system can outperform merely personalized-based recommender systems.

On the other hand, recommender systems can also be helpful for predicting popular contents. Petrovic et al. (2011); Zaman et al. (2010)proposed a method for predicting popular content based on recommender system algorithms. Their approach seeks to predict whether a tweet is retweeted by another user on the basis of collaborative filtering algorithm.

2-3-3 Popularity Predictions and Influences

Due to the high importance placed by users on making content popular, there has been quite some research into the identifying of *influential* users. By studying the behavior of influential users, we can further investigate their contribution to predictions concerning popular contents.

Solis (2012) discussed the importance of *digital influence* and problems with *measuring influence* and finally defined influence. Some of the most frequent questions which attract the audience were; What is influence and what makes someone influential? Who is influential in social networks and why? How can I recognize influence or the capacity to influence? By better understanding how digital influence works, businesses can improve their understanding of the market and deploy social media media to steer positive

conversations. (Solis, 2012) believes that influence as a score is imprecise, however recently many studies have defined metrics as a score which is assigned to people on what they do and say in social networks. We have reviewed some these studies and we noticed that most of them suffer from the same problem.

Previous studies made use of different terminology in their research into matters such as the influence, popularity, important nodes and efficient seed sets. Below are six different titles that are frequently used in the related literature:

- Predicting the popularity of users on Twitter
- Predicting the popularity of Tweets on Twitter
- Predicting the influential users on Twitter
- Predicting the influential tweets on Twitter
- Predicting the influence of Users on Twitter
- Predicting the influence of tweets on Twitter

Each of these definitions have different meanings and implications. The first definition, the popularity of users on Twitter, can translate into the number of followers of users. The second, implies the number of retweets obtained from tweets. But in the last four titles, very general words such as *influence* and *influential users* appear which cannot be translated into one single measurement. There are clear distinctions on why popularity and influence are not the same definition.

In relation to social network analysis (SNA), several metrics exist that indicate the social influence of users in networks. The top three common measures are presented by (Freeman, 1979) as, (i) Degree centrality, which indicates the number of direct/indirect ties of a node to other nodes. (Iyengar et al., 2010) called well-connected users "hubs". (ii)closeness centrality: unlike degree centrality takes into account only immediate ties rather than all the connections and emphasizes the distance of a user to all others in the network by focusing on the distance from each user to all the others.(iii) Betweenness centrality: quantifies the number of times a user acts as a bridge along the shortest path between two other users. In same the area one study indicates that such bridges that connect two unconnected parts of the network are influential[2].

[2]http://www.fastcompany.com/27701/virus-marketing

Hinz et al. (2011) put forward the notion of degree and betweenness centrality to find the best seed set of influential users dependent on social links. They found that hubs and bridges are more likely to participate in successful seeding strategy in viral marketing campaigns.

Cha and Gummadi (2010) presented an empirical analysis of the influence of the twitter, they compared three different measures of influence: *in-degree* which counted the number of followers of a user, The more followers a user has, the more influential the user is. *Retweet*, which involves counting the number of retweets, belongs to the post of one person, so the more retweet post a user gets the more influential he is. *Mentions* counts the number of mentions containing a person's name, so the more replies a user receives, the more influential he is.

Bakshy et al. (2011) narrowed down the definition of influence to the ability of the user to post URLs which diffuse through the Twitter follower graph. They studied only the users who post URLs and called them "seed" content. They quantified the influence of a given post by the number of users who repost the URL. They fitted a model which predicted influence using individual attributes and past activity to examine the utility of such a model for targeting users. The size of the diffusion is more directly associated with diffusion and the dissemination of information.

Li et al. (2013), like in other works, defined influence as a successful diffusion of information and they correctly mention that information/influence propagation and information/influence diffusion and information cascades are the same concept and that is a concept that is used frequently in their work.

Cha and Gummadi (2010) compared three measures and discovered that the number of retweets and the number of mentions are correlated while the number of friends are not correlated and so their hypothesis is that the number of followers of users may not be a good influence measure.

Kwak et al. (2010) compared different influence measures in terms of both the rewteets and the follower network. The various authors ranked users by the number of followers and PageRank and found two rankings that were similar. They found a gap in the influence calculated on the follower network versus the retweet network which is inferred from the number of followers and the popularity of the tweets.

Weng et al. (2010) did not define influence very clearly, they mentioned that an influential twitter is one with certain authority within the social network. They implemented topic sensitive pagerank to overcome the problem of iden-

tifying the interest of twitters which affects the way twitters influence one another. So they took into account both link structure and topical similarity among twitters.

In another study, Katz and Lazarsfeld (1955) introduced *Opinion leadership* in a two-step flow theory, where opinion leaders receive information from society through the news media and send it to less informed people. Rogers (1995) relies on the idea of two-step flow theory in developing his ideas on the influence of Opinion Leaders in the diffusion of innovation. Opinion Leaders typically have greater exposure to the mass media, more social experience, greater viewers and followers and are more innovative. Wejnert (2002) mentions terms of benefit Vs cost, which means that successful adoption of innovation is the benefit and indirect/direct cost which you pay to increase the benefit is the cost of innovation. An example would be the need to buy a new kind of fertilizer to use innovative seeds.

Trusov et al. (2010) had a different idea, they measured influence based on network activity by studying log-in data in social networks and showed how the posting attitude of one user has an effect on their networks members who were at top-level, or those who are connected by direct invitation and at other levels those who were friends of friends. They evaluated whether content from within members at this top level changes to a log-in frequency and length of stay on site, and concluded that such changes are evidence of influence of top level on the reset of members.

So as we can see, influence is not a black and white concept. In OSN, what people are influenced by are different persons to the person and there is not a single way of measuring influence in OSN.

2-4 Social Network Prediction Applications

Predictive models analyze past information to assess how likely it is than an event will occur in the future. Although human experts could have greater accuracy they are not scalable and do not work properly in cases when events have very low or high probability and they are definitely more expensive compared to the computer-based approach (Bothos et al., 2010).

Different studies have focused on the applications of micro-blogging services in different fields. For instance, Bollen et al. (2010); Sprenger and Welpe (2010) studied the applications of micro-blogging in the stock market. They investigated whether collective intelligence from micro-blogging information

can predict events in the stock market. Asur and Huberman (2010) made a linear regression model to predict box-office revenues before the release of movies using Twitter data. Jansen et al. (2009) believe that micro-blogging could be used as part of online word-of-mouth marketing and services like Twitter and could play an important role in marketing. Micro-blogging has become an important platform for information publishing and dissemination. In recent years, the adoption and use of micro-blogging in an emergency has received a great deal of attention. For example, Culotta (2010) studied the feasibility of detecting influenza outbreaks by analyzing micro-blogging data.

A number of studies discussed the use of micro-blogging as a communication information sharing resource in the event of various crises, involving for instance violence and natural disasters. Sakaki et al. (2010) use a real-time characteristics of Twitter and people's actions and posting on Twitter during catastrophe to investigate the real-time interaction during events such as earthquakes on twitter and they proposed an algorithm to monitor tweets and to detect a target event.

2-5 Summary

In this chapter we discussed various studies related to our work. Due to the importance of predicting popular content in social networks there have been quite a number of studies on how and why content gets popular in social networks. We explored different studies which are based on information propagation and popularity predictions in different social networks. Twitter itself has also been the subject of much related work in this area. We discussed the advantages and limitations of the existing studies in this area and motivate our approach to popularity predictions in twitter social networks.

Chapter 3

Predictive Model

3-1 Introduction

In this chapter we describe our proposed learning method for predicting the popularity of tweets in twitter social networks. We shall model the problem of popularity prediction as a *binary* classification problem. To formulate the binary classification problem we need to define what exactly constitutes popular and unpopular content. As we mentioned in Chapter 1, there are different approaches to defining popularity. Our proposed definition of popularity is based on the works of (Hong et al., 2011a; Zhang et al., 2014) in which the popularity is defined as the number of retweets that a tweet will get. In the present work we will consider different thresholds to define the popular tweets and perform different experiments in various setups in order to also obtain the best possible threshold for our classification problem.

The rest of this chapter is organized as follows. We review the prediction challenges and explain why it is important to predict popular contents in twitter. The overall architecture of our proposed learning-based method is then described. The next section introduces the different types of classifiers that we used and determines their usefulness for our classification task. In Section 3.4 we describe in detail the features that we extracted for our classification task and how we combine these features. In the next chapter we experimentally examine our proposed learning method.

3-2 Prediction Challenges

The ability to predict popular contents in social networks is quite important for *adopting* and *personalizing* the huge amount of information for users. Successful prediction can provide the most relevant contents to users and improve user's experience with social media (Yu and Kak, 2012). Furthermore, early prediction of viral information is quite useful for marketing, trend analysis and popular news detection. Automatic prediction of popular content is not however a simple problem. This problem is even more challenging for twitter social networks due to limitation placed on the size of a tweet message. Moreover, the imbalanced nature of the data, i.e., the huge difference between the number of tweets which get popular with those which do not, makes the problem even more challenging. In fact we need to find the tweets which are likely to become popular in a large pool of tweets which are very unlikely to become popular. Finding the features that are able to distinguish popular tweets from those which are not, is quite important. Another challenging issue of the popularity prediction problem is the ability to predict the popular tweets as soon as possible. In other words we, are in practice limited to using only the information which is available shortly after has been published a tweet. In the next section we explain how we approach this problem which motivating our decision.

3-3 Model Architecture

Our proposed approach to popularity prediction is based on a feature-based classification model in which we extract a set of features from tweets and classify them as popular/unpopular classes. We are in fact interested to see to what extent we can predict the popularity of tweets in the twitter social network. In other words our research interest reduces to a binary classification problem in which a tweet will be assigned to a popular (positive) or unpopular (negative) class. As we mentioned in Chapter 1, there are several research question that can be raised to tackle this problem. In this section we introduce the overall architecture of our system, which comprises different components. Each component needs an in-depth analysis to be able to drive the optimal model for this problem and give answers to the research questions posed in Chapter 1.

Figure 3-1 illustrates the overall architecture of our proposed model. The data collection method is illustrated in the left side of the figure . Detailed

information about the data collection is described in Chapter 4. The model then extracts several features from tweets and different machine learning approaches are used to train classifiers. The classifier is then used to predict whether a tweet will be popular or not.

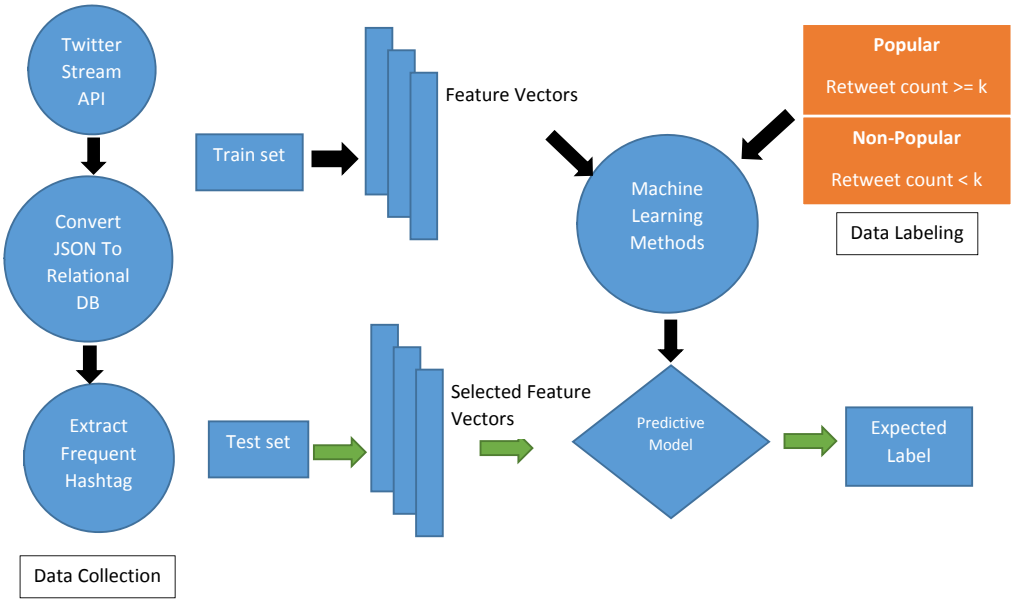

Figure 3-1: Schematic representation of model architecture

The two important decision for learning-based systems are the choice of classifier and the features that are extracted from the data. In the following two section we describe our choices in more detail and motivate our approach. We introduce the different discriminative and generative classifiers which are potentially suitable for our classification problem and the features that will be extracted from the twitter environment. In the next Chapter we experimentally examine the suitability of the classifiers that are introduced in this Chapter and find the optimal model for our problem which is both accurate and generalizable.

3-4 Classification Methods

The task of automatic classification of data can be carried out with the help of several different classifiers. The machine learning community has intro-

duced several feature-based classifiers which are suitable for different applications (Theodoridis and Koutroumbas, 2008). Depending on the features and nature of the data, several classifiers can be used for a prediction task. In statistical machine learning, a classifier can be either *generative* or *discriminative*. A generative classifier tries to predict a probabilistic distribution for each class of data and assign an unknown sample to the class with highest likelihood. On the other hand, discriminative approaches try to depict a curve which best discriminates the data points in different classes.

Depending on the nature of the data, features and desired performance and complexity different models can be trained. In this section we shall describe the classifiers that we used and the reasons for using them. In the next chapter we shall experimentally show the performance of each method and introduce the optimal model for our problem.

Generative classifiers work by learning the class conditional probability, that is,$f_c(\mathbf{x}) = Pr(X = \mathbf{x}|C = c)$ for each class c. In this formula let the feature vector be $\mathbf{x}$ and the class labels be C. Assume the prior probability for class c is denoted as π_c , $\sum_{c=1}^{C} \pi_c = 1$. In order to estimate $f_c(\mathbf{x})$, we will first make some assumptions about its form. First assume that $f_c(\mathbf{x})$ is normal or Gaussian with mean μ_c and covariance σ_c^2.

In particular, the following estimates are used, where n is the total number of training observations, and n_c is the number of training observations in the c_{th} class. μ_c is simply the average of all the training observations from the c_{th} class and σ^2 can be seen as weighted the average of the sample variances for each of the C classes.

$$\hat{\mu_c} = \frac{1}{n_c} \sum\nolimits_{i:C_i=c} X_i \tag{3-1}$$

$$\hat{\sigma^2} = \frac{1}{n - C} \sum_{c \in C} \sum_{i:C_i=c} (X_i - \hat{\mu}_c)^2 \tag{3-2}$$

π_c is usually estimated simply by the empirical frequency of the training set equation 3-3.

$$\hat{\pi_c} = \frac{\text{Number of samples in class c}}{\text{Total number of samples}} = \frac{n_c}{N} \tag{3-3}$$

The class conditional probability $f(\mathbf{x})$ can then be defined as a Gaussian distribution as follows: 3-4.

$$f_c(\mathbf{x}) = \frac{1}{(2\pi)^{p/2}|\sum_c|^{1/2}} e^{-\frac{1}{2}(\mathbf{x}-\mu_c)^T \Sigma_c^{-1}(\mathbf{x}-\mu_c)} \tag{3-4}$$

p is the dimension and $\sum_c$ is the covariance matrix. The vector X and the mean vector μ_c are both column vectors. For QDC this is the density of X conditioned on the each class C or class $C = c$ denoted by $f_c(\mathbf{x})$. According to the Bayes rule, what we need is to compute the posterior probability equation which defined as follows:

$$Pr(C = c|X = \mathbf{x}) = \frac{f_c(\mathbf{x})\pi_c}{\sum_{i=1}^{C} f_i(\mathbf{x})\pi_i} \tag{3-5}$$

given the posterior probabilities the class label for a given sample can be decided based on the following decision rule

$$class(\mathbf{x}) = \arg\max Pr(C = c|X = \mathbf{x}) \tag{3-6}$$

3-4-1 Linear and Quadratic discriminant Classifiers

Linear and quadratic discriminant classifiers are two simple yet effective generative classifiers that have been widely used in different applications such as for text classification (Aggarwal and Zhai, 2012) and face recognition (Lee et al., 2010). To our knowledge this classifier has never been used for the task of popularity prediction in social networks. Due to their simplicity and low time complexity we adopted different linear and quadratic classifiers (such as LDA and QDA) to examine their suitability for our prediction problem.

The purpose of discriminant analysis is to assign labels to one of several groups or classes assuming that the measurements from each class are normally distributed and different classes have the same covariance matrix, Σ. Quadratic Discriminant Analysis, on the other hand, set outs to find the

quadratic combination of features and is more complex than linear discriminant analysis. Unlike LDA, QDC does not make the assumption that different classes have the same covariance matrix Σ. Instead, QDC makes the assumption that each class C has its own covariance matrix Σ_c.

A major problem associated with LDA and even more with QDA is that a large number of parameters have to be estimated in the case of high-dimensional datasets[1]. But most of the datasets in our problem are low-dimensional (around 20 features), which makes the use of these two classifier less of a problem in terms of complexity.

3-4-2 Naive Bayes Classifier

A Naive Bayes classifier is a simple generative classifier based on the application of the Bayes' theorem with strong assumpt-ions that the features are highly independent. In other words, a Naive Bayes classifier assumes that the presence or absence of a particular feature is unrelated to the presence or absence of any other feature, given the class variable. Despite their naive design and apparently oversimplified assumptions, Naive Bayes' classifiers have worked quite well in many complex real-world situations such as for text classification (Frank and Bouckaert, 2006), spam detection (Freeman, 2013), sentiment classification (Narayanan et al., 2013) and with opinion mining (Fouzia Sayeedunnissa et al., 2013).

The Naive Bayes model works very well in the problems in which the features are independent. In our tweet classification problem as you will see later in this chapter, most of the feature are independent and the Naive Bayes classifier is potentially a proper classifier for that. The Bayes' classifier calculates the probability of an object belonging to each of the classes. Given a class label C for a tweet (popular or non-popular) a tweet which is represented by a feature vector x ($x_1, ...x_f$). From the Bayes' rule we can calculate class posterior probability $P(c|X)$ as follows:

$$P(c|X) = P(c \mid x_1, \ldots, x_f) = \frac{P(C)P(x_1, \ldots x_f \mid c)}{P(x_1, \ldots, x_f)} \tag{3-7}$$

Using the naive independence assumption we can write:

[1] http://www.hua.edu.vn/khoa/fita/wp-content/uploads/2014/01/Some-Linear-Classifiers-for-High-Dimensional-Data.pdf

$$P(x_i|c, x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_f) = P(x_i|c) \tag{3-8}$$

for all $i = 1...f$ using this equation the class posterior can be written as:

$$P(c \mid x_1, \ldots, x_f) = \frac{P(c) \prod_{i=1}^{n} P(x_i \mid c)}{P(x_1, \ldots, x_f)} \tag{3-9}$$

Since $P(x_1, ..., x_f)$ is constant for all classes, we can use the following classification

$$\hat{c} = \arg\max_t P(c) \prod_{i=1}^{n} P(x_i \mid c) \tag{3-10}$$

the class with highest posterior probability would be decided as the class label for a given sample.

3-4-3 Distance-based Classifiers

Due to their simplicity, we examined two distance-based classifier namely K-Nearest Neighbour (K-NN) and Nearest Mean classifiers. The K-Nearest Neighbour (K-NN) classifier is another popular and simple classifier which is potentially suitable for our problem. k-NN is a type of instance-based learning, or lazy learning, in which the function is only approximated locally and all computation is deferred until classification. The k-NN algorithm is among the simplest of all machine learning algorithms.

K-nearest neighbour (K-NN) algorithm is a discriminative classification algorithm that assigns query data to the class to which most of its k-nearest neighbours belong. A Euclidean distance measure is used to find the k-nearest neighbours from the sample pattern from a set of known classifications(Witten and Frank, 2005). A drawback of the basic "majority voting" classification occurs when the class distribution is skewed. Frequently class tends to dominate the prediction of the new example, because this tends to be common among the k nearest neighbours due to their large number (Coomans and Massart, 1982).

The nearest Mean Classifier is a classification model that assigns to observations the label of the class of training samples whose mean is closest to the observation. This classifier works in a similar way to the nearest neighbour classifier. In this classifier, instead of storing each training sample, the mean of each class is stored as a class. Using Euclidean distance and objects are assigned to groups with the nearest mean.

Nearest mean classifiers are less sensitive to imbalance data because the mean of classes do not depend on the number of samples in each class.

3-4-4 Support Vector Machine (SVM)

SVM is a discriminative based classifier which has been successfully applied to many problems such as text classification (Aggarwal and Zhai, 2012), image processing and face recognition (Heisele et al., 2001; Jafri and Arabnia, 2009), Spam detection (Wang) and many more problems in social media. This classifier has also been adopted in the tweet popularity prediction problem (Zhang et al., 2014). However, as we mentioned earlier in the previous chapter, this work has not employed some of the features such as content features, that we introduced in this work. SVM tries to discover a hyperplane which discriminates classes and it is not necessary to estimate what is the class density $P(X|C)$ or what is the posterior probability value $P(C|X)$. Suppose we are given a training set $(\mathbf{x}_i, y_i), i = 1, ..., n$ in which $\mathbf{x}_i = (x_{i1}, ..., x_{in})$ is a n-dimensional sample and $y_i \in \{1, -1\}$ is the corresponding label. The task of a support vector classifier is to find a linear discriminant function $g(\mathbf{x}) = \mathbf{w}^T\mathbf{x} + \mathbf{w}_0$, $\mathbf{w}$ has to be chosen to satisfy $\mathbf{w}^T\mathbf{x}_i + \mathbf{w}_0 \geq +1$ for all points in class $y_i = +1$. Similarly it must satisfy $\mathbf{w}^T\mathbf{x}_i + \mathbf{w}_0 \leq -1$ for those in class $y_i = -1$. Therefore we seek a solution which is such that the following condition holds.

$$y_i(\mathbf{w}^T\mathbf{x}_i + \mathbf{w}_0) \geq 1 \; i = 1, ..., n \tag{3-11}$$

The optimal linear function is obtained by minimizing the following quadratic programming problem:

$$\min \tfrac{1}{2}w^T w - \textstyle\sum_{i=1}^{n} \alpha_i(y_i(w^T\mathbf{x}_i + w_0) - 1) \tag{3-12}$$

where $\alpha_i, i = 1, ..., n; \alpha \geq 0$ are Lagrange multipliers ,subject to $\alpha_i \geq 0$,for all n

Minimizing the norm makes the margin maximum. At the optimum of this new objective function, the partial derivative of the objective function with respect to w and b must be zero, which leads to the following solution:

$$w = \sum_{i=1}^{n} \alpha_i y_i \mathbf{x}_i \tag{3-13}$$

where $\alpha_i, i = 1, ..., n; \alpha \geq 0$ are Lagrange multipliers.

$$\max \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j \mathbf{x}_i^T \mathbf{x}_j \tag{3-14}$$

this expression is known as the dual optimization problem and it has to be maximized with the following constraints

$$\alpha \geq 0, \sum_{i=1}^{n} \alpha_i y_i = 0 \tag{3-15}$$

This optimization problem is a constrained quadratic programming task because of the $\alpha_i \alpha_j$ term. To be able to linearly separate data, the feature space should be typically mapped to a higher dimensional space. Functions that correspond to inner products in some spaces are known as kernel functions.

The kernel function $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ takes two samples from input space and maps it to a real number indicating their similarity. For all $\mathbf{x}_i, \mathbf{x_j} \in \mathcal{X}$, the kernel function satisfies

$$k(\mathbf{x}_i, \mathbf{x}_j) = \langle w(\mathbf{x}_i), w(\mathbf{x}_j) \rangle \tag{3-16}$$

Where w is an explicit mapping from input space χ and $< a, b >$ indicate the inner product to a and b to a dot product feature space w. Where w is a Hilbert space [2]. This inner product can be replaced by another kernel function. There are plenty of kernel functions, which are each equivalent to an inner product after some transformation that we can use. The following are the four most popular kernel functions:

[2] A Hilbert space is a complete linear space equipped with an inner product operation

Table 3-1: SVM kenels

Name	Definition
linear	$K(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{x}_i^T \mathbf{x}_j$
Radial Basis	$K(\mathbf{x}_i, \mathbf{x}_j) = e^{-\gamma \|\mathbf{x}_i - \mathbf{x}_j\|^2}, \gamma > 0$
polynomial	$K(\mathbf{x}_i, \mathbf{x}_j) = (\gamma \mathbf{x}_i^T \mathbf{x}_j + c_0)^d, \gamma > 0$
Sigmoid	$K(\mathbf{x}_i, \mathbf{x}_j) = \tanh(\gamma \mathbf{x}_i^T \mathbf{x}_j + c_0)$

where
d is Parameter degree of a kernel function.
γ is Parameter γ of a kernel function .
c0 is Parameter coef0 of a kernel function .

The RBF Polynomial Sigmoid are more flexible and both have additional parameters (γ) that must be set by the user.

In cases where data is non-separable the training feature vector can adhere to three categories. I) vectors can fall outside the margin and can be correctly classified. II) vectors can fall inside the margin and be correctly classified. III) vectors can be misclassified. These three categories can be dealt with under a single type of constraint :

$$y_i(w^T \mathbf{x}_i + w_0) \geq 1 - \xi_i \tag{3-17}$$

The first category of the data corresponds to $\xi_i = 0$, the second to $0 < \xi_i \leq 1$ and the third to $\xi_i > 0$. The goal is to make the margin as large as possible while at the same time making the number of points with $\xi_i > 0$ as small as possible. So equation 3-18 can be written as follows:

$$\min \tfrac{1}{2} w^T w - \textstyle\sum_{i=1}^{n} \alpha_i (y_i (w^T \mathbf{x}_i + w_0) - 1) + C_{svm} \sum_{i=1}^{n} \xi_i \tag{3-18}$$

where C_{svm} is a penalty parameter for the errors on the training set. In this work we are using package which is implemented by the LIBSVM library for support a vector machine (Dimitriadou et al., 2010)

3-5 Features

A critical factor when developing a prediction model is to represent samples with a good set of features. Good features should be *informative* and should

have *discriminative* power. That means that the features should be able to discriminate between the tweets that become popular and those which do not. In our proposed model we have extracted features from three sources of information: the features of the tweet, the user who posts the tweet and the follower network of connected users (i.e. followers and followees). The features can be either discrete which means that they can have a value from a set of defined values, or they can be continuous which means that the features have a continuous value.

Most of the features that we extracted for this work are independent. The tweet features such as date and time for example, do not depend on th user who publish the tweet. Nevertheless some features such as follower count and friend count has some correlation with each other. Figure 3-2 shows the correlation between the features that we extracted in this work.

In this section we describe the features that we extracted and different approaches are adapted to combine such features.

3-5-1 Tweet Features

Tweet features are the features that are extracted from the tweets themselves. We extracted several features from tweets. Table 3-2 lists the features that we extracted from tweets, together with their description and their type (i.e. continuous or discrete).

Table 3-2: Features extracted from tweets

Feature	Type	Description
Date	discrete	Day of the week when the tweet was posted
Time	discrete	Hour of posting
URL	discrete	Tweet containing URL or not
Hashtag	discrete	Tweet containing hashtag or not

3-5-2 User Features

One of the most important factors that contributes to the popularity of any tweet is the user who posted the tweet. Extracting features from the user can significantly help the classifier to predict the popular posts. We extracted several discrete and continuous features from users, all of which are listed in Table 3-4.

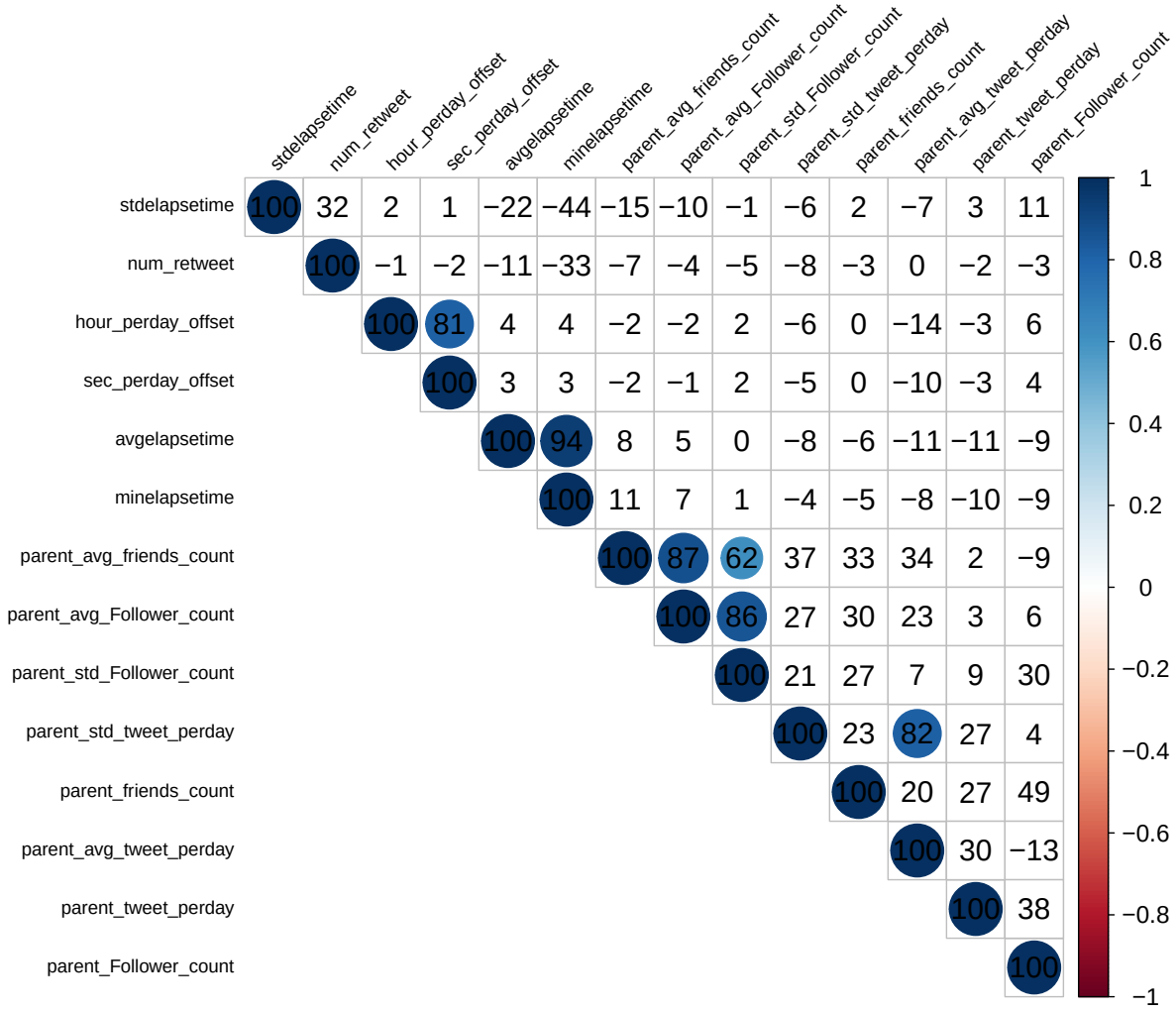

Figure 3-2: The correlation between different features.

Table 3-3: Features extracted from users

Feature	Type	Description
parent Follower count	continuous	number of followers of the user
parent friends count	continuous	number of friends of the user
parent tweet perday	continuous	number of tweets per/day each user has done

3-5-3 Network Features

In addition to tweet and user features, we also extracted some additional features from the network of the user who posted the tweets see (Figure 3-3). In this schema, node one is the original tweeter, who has 13 followers, and node two has seven followers, these followers are called "*followers of follower*". These features helps to better exploit the information in the user's network which can potentially contribute to predicting the popularity of the tweets.

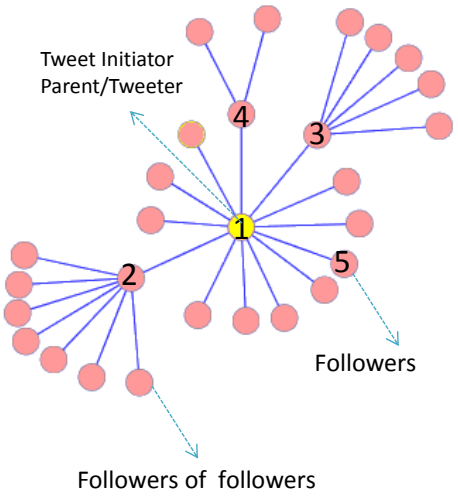

Figure 3-3: Tweet publisher and its follower network.

Avg/Std # of followers of followers

These two features are constructed on the basis of the network of the users. The network of the user who posts the tweet has an important role in the propagation and popularity of tweets because the tweets are mainly propagated through the network of users. In fact the tweets of the users who have

a larger network have higher chance to be exposed and therefore retweeted. To build these features, we calculate the average and standard deviation of the number of followers of user's followers which is indicated by $\bar{fof}$. More specifically, suppose that user u has n followers and $|fo_i(u)|$ indicates the number of followers of i^{th} follower of u. The average and standard deviation of number of followers of followers is then defined as:

$$\bar{fof}(u) = \frac{\sum_{i=1}^{n} |fo_i(u)|}{n} \tag{3-19}$$

$$\sigma_{fof}(u) = \sqrt[2]{\frac{\sum_{i=1}^{n} |fo_i(u) - \bar{fof}(u)|^2}{n}} \tag{3-20}$$

Avg/Std # of fRiends of followers

These two features are constructed in a very similar way to the avg/std number of followers of followers. The only difference is that instead of having followers of followers, the number of friends of each follower will be taken into account which is indicated by $\bar{rof}$. More specifically, suppose that user u has n followers and $|fr_i(u)|$ indicates the number of friends of i^{th} follower of u. These two features are calculated using the following equations:

$$\bar{rof}(u) = \frac{\sum_{i=1}^{n} |fr_i(u)|}{n} \tag{3-21}$$

$$\sigma_{rof}(u) = \sqrt[2]{\frac{\sum_{i=1}^{n} |fr_i(u) - \bar{rof}(u)|^2}{n}} \tag{3-22}$$

Avg/Std Tweets per day of followers

As the name of these two features suggests, they are constructed by averaging/standard deviating over the number of tweets per day of all the followers of the user. These two features are calculated using the following equations:

$$\bar{tof}(u) = \frac{\sum_{i=1}^{n} |t_i(u)|}{n} \tag{3-23}$$

$$\sigma_{tof}(u) = \sqrt[2]{\frac{\sum_{i=1}^{n} |t_i(u) - \bar{tof}(u)|^2}{n}} \tag{3-24}$$

where $t_i(u)$ indicates the tweets per day of the i^{th} follower of user u.

Table 3-4: Features extracted from users

Feature	Type	Description
$\bar{fof}$	continuous	avg # of followers of followers of the user
$\sigma_{\bar{fof}}$	continuous	std # of followers of followers of the user
$\bar{rof}$	continuous	avg # of friends of followers of the user
$\bar{tof}$	continuous	avg # of tweet perday of followers of the user
σ_{rof}	continuous	std # of tweet perday of followers of the user

Early Tweet Features

In addition to typical features of tweets, we also extracted a set of features based on the early features of tweets. For extracting theses features we monitor the events that are happening 120 second after the tweet is published. Table X list the feature that we extracted from this elapsed time period.

Table 3-5: Features extract form first 120 Sec of retweet

Feature	Type	Description
Number of retweet	continuous	# of retweet after 120 sec of first retweet
AvgElapseTime	continuous	Average time of retweets in the first t min
StdElapseTime	continuous	Std time of retweets in the first t min

3-5-4 Combination of Features

We have extracted several features in our model. To obtain the optimal classifier it is important to effectively combine the features. In this work we have conducted a *full factorial design* so that the informativeness of each feature can be calculated. Hassan et al. (2006) has shown that factorial experimental design is a viable approach in feature selection. In statistics, a full factorial experiment is an experiment whose design consists of two or more factors, each with discrete possible values or levels and whose experimental units take on all possible combinations of these levels across all such factors. Due to the varying contribution of each of the features in the classification task, we have done some experiments based on the factorial design model to discover what are the most informative features. Furthermore, full factorial design helps us to detect the useless features in our classification task, so leading to the designing a better model. In the next chapter we will experi-

mentally explain our feature selection method and the contribution made by each single feature to our predictive model.

3-6 Summary

In this chapter we explained our predictive model from the theoretical point of view. To build our predictive model, we examined different types of learning approaches and different features. We explained the classification models, as well as the features we also explained how we obtained them. In the next chapter we will experimentally test our approach to different datasets and justify our model by comparing our results with some baselines and previous works.

Chapter 4

Experimental Results

4-1 Introduction

In this chapter we explain in details how the dataset was collected and how the experiment were conducted. We collected four different datasets from twitter and performed different experiments on them to see to what extent we can predict the popularity of tweets in the Twitter social network. We further explain how the data is collected and how they are split for training and testing. We then describe how we setup the classifiers that we considered for this problem and explain in details how can we effectively tune their parameters to be suitable for our prediction task.

This chapter is organized as follows: Section 4-2 describes the datasets that we used for this work and their collection method. We further explain in this section how we transferred the datasets into a relational database and also introduce the splitting strategies that we considered in this work. In section 4-3 we first introduce the evaluation metrics that we used in this work and motivate their choices. We then address the challenge of tuning the right parameters for the classifiers and compare the performance of different classifiers with different configurations. The summary and concluding points are further discussed in section 4-4.

4-2 Dataset

In this section we will describe in more detail which datasets are used and how we collect this data. Twitter is an information exchange network that produces 200 million tweets per day [1]. In this work we did our experiments on a set of static datasets to see to what extent we can address the research questions which we posed in previous chapters. To be able to test our proposed methods, we created four different datasets using *the twitter streaming API*[2]. It is important to note that the twitter APIs are constantly changing and developing Twitter is not a one-off event.

The four datasets are different in terms of the time when they were collected, the size and the topic of the tweets. Having four different dataset allows us to test our methods on different situations to see how well our methods can be generalized. As streaming API is a free service and we do not have an obligation to collect 100% of data so we use steaming API.

We have created the four datasets on different topics. We have created three datasets from the hot topics, each with a different size and time when the data was collected and the other one is a more general dataset which is not necessarily related to the hot topic of the day. We chose to have datasets from both hot and general topics to gauge the performance of our approach in different types of datasets. These datasets are illustrated in Table 4-1.

Datasets	Description	Duration
Steve Jobs Death	Steve Jobs quit from being CEO	4 Days
The US Election	During US election campaign	16 Days
Foxnews Obama Assassination	Fox News Twitter account hacked	4 Days
:)	All tweets contain :)	1 Days

Table 4-1: Four datasets collected over different periods of time

4-2-1 Twitter structure

Twitter is a micro-blogging site which was created in 2006. This service allows users to share information in the form of 140 character messages known as *tweets*. Users have two different networks, friend networks (following) which receive posts from persons in their time-lines, which shows

[1] http://mashable.com/2011/06/30/twitter-200-million/
[2] https://api.twitter.com/

the numbers of users who are influenced by Twitter. Secondly, followers relationships which follow him/her from a directed follower network, all followers will receive posted message in their time-lines .

Users categorized posts by topic by adding # hashtags these content categories help users to search for a subject and this can occur anywhere in a Tweet at the beginning, middle ,or at the end. When hashtag words become popular they are then called *Trending* topics. In order to send your message to a specific user, it is sufficient to mention his/her user name in that post and they will then see the Tweet in their *Mentions* tab .

Tweet @ [account]

In this way, the originator of this tweet can add other users to the post. When the user opens his/her own permanent page, he can see all the posts he/she is mentioned in. We call this *post action* as it concerns direct post. Another use worth mentioning involves rebroadcasting of other persons posts or *(*retweeting). Users can use the retweet button option available under the post or they can mention the RT @username at the beginning of post. Retweets are useful because they allow one to track the flow of information on twitter.

[additional text] RT @[account] : [original tweet]

Every link between a tweet and retweet can be imagined as a directed edge in a graph, if one connects these retweets together one obtains the *retweet network*.

Twitter Message Structure

At first we are going to describing variables inside tweets. Each tweet has one main body containing single field attributes like,(id, text, source, in reply to status id) and complex attributes like (User, Entity, Geo, Place) which contain more attributes inside them. Here we have shown the three most important distinct parts of each Tweet, *Tweet, User, Entity* in JSON list 1

Tweet Body Fields

Each tweet contains several fields which we show in listing 1 where we are going to explain them in detail. However it is important to mention that the

twitter JSON stream is not reliable and as we mentioned earlier the twitter API can be changed during the time. But we can still count retweet count and extract some useful information from the JSON stream.

```
"created_at":"Fri Jul 04 12:37:51 +0000 2014",
"tweetid":485039785208446976,
"text":"RT @JZarif: Iran's Message: We Can Make History",
"truncated":false,
"in_reply_to_status_id":null,
"in_reply_to_status_id_str":null,
"in_reply_to_user_id":null,
"in_reply_to_user_id_str":null,
"in_reply_to_screen_name":null,
"place":null,
"contributors":null,
"retweet_count":274,
"favorite_count":0,
"lang":"en"
"user":{},
"entities":{},
"retweeted_status":{},
"geo":null,
```

Listing 1: Tweet JSON Stream

- *TweetId*: Tweets are identified by long unique integers which increase per tweet throughout the whole twitter domain.
- *retweet Status*: Contains original tweet information. It will appear in the retweet body.
- *create at*: The date when the user became a members of Twitter.
- *tweet created at*: Time when a tweet/retweet/Reply post occurred.
- *parent Tweet Id*: TweetID of post generator shown in retweet attributes.
- *parent User Id*: UserID of post generator shown in retweet attributes of the originator During the data processing we create our own attribute to facilitate the future work.

- *retweet time difference*: Each tweet has a time stamp so by reducing the time stamp between the original tweet and the retweet we can capture the retweet time difference.

User Profile Fields

Each tweet JSON stream contains a user field which contains a user profile. In listing 2 we have shown these fields.

```
"user":
   {
        "userid": 15496407,
        "name": "Jason H. Moore, Ph.D",
        "screen_name": "moorejh",
        "location": "Hanover, NH, USA",
        "description": "Third Century Professor,
        Bioinformatics,Complexity, BigData",
        "url":
        "entities":
        {
            "url":{},
            "description":{}
        },
        "followers_count": 6440,
        "friends_count": 1980,
        "listed_count": 534,
        "created_at": "Sat Jul 19 23:10:24 +0000 2008",
        "favourites_count": 177,
        "utc_offset": -14400,
        "time_zone": "Eastern Time (US & Canada)",
        "statuses_count": 21275,
        "lang": "en",
      }
```

Listing 2: User fields in Tweet JSON stream

- *userId*: Tweeter users have a unique id.

- *friends count* : Indicates number of users the user follows (known as "followings")
- *followers count* : Indicates the number of users that follow the user
- *status count* : Indicates the number of messages that the user has posted so far.
- *lang* : Indicates the language of the posts he has chosen for the messages.

Tweet entity Fields

The tweet entity gives extra information about the tweets themselves. We have shown this in listing 3.

```
"entities":
  {
      "hashtags":[],
      "symbols":[],
      "urls":[],
      "user_mentions":[]
  }
```

Listing 3: Entities fields in Tweet JSON stream

- hashtag : List of hashtags which are used in the tweet text.
- symbols : List of any extra symbols which are used in the tweet text.
- Urls : List of Urls which are mentioned in the tweet text.
- User mentions : List of Users which are mentioned in the tweet text

In section 4-2-2 we will describe in details how these four dataset are collected.

4-2-2 Dataset Collection

In this section we describe in more detail how we collected the four dataset with the twitter APIs. There are three different ways to access twitter data: (I) Twitter Search API (REST API),(II) Twitter Streaming API, (III) Twitter Firehouse. Streaming API gives you the opportunity to access tweets happening in near real-time. With Twitter's Streaming API, users register a set of criteria (keywords, usernames, locations, named places, etc.) and as tweets match the criteria, they are pushed directly to the user. The major drawback of the Streaming API is that Twitter's Steaming API provides only a sample of tweets that are occurring. The actual percentage of total tweets users receive with Twitter's Streaming API varies greatly, depending on the criteria users request and the current traffic. Studies have estimated that by using Twitter's Streaming API users can expect to receive anywhere from 1% of the tweets to over 40% of the tweets in near real-time. However this shortage can be overcame by using Twitter firehouse API which is not a free service guarantees a delivery of 100% of the tweets that match your criteria. Although since we only access to free APIs we used streaming APIs.

The Twitter search API was founded on REST architecture. REST architecture refers to a collection of network design principles that define resources and ways to address and access data. By allowing third-party developers partial access to its API, Twitter allows them to create programs that incorporate Twitter's services. The Search API passes on the relevant results to ad-hoc user queries from a limited corpus of recent tweets [3]. The REST API allows access to the nouns and verbs of Twitter such as User Profile, Time-lines, Tweets, Tweet-locations, Lists, Friends and Followers.

All these different access methods have some input and output. The input is a specific criteria such as keywords, time, hashtags in the case of streaming API and user id, tweet id, location and etc. in case of search API. The output format of all these data access methods will be JSON or XML. JSON is a simple text format that facilitates reading and writing, it is a widely used data-interchange language because its parsing and its generation is easy for machines. In order to be able to extract data from JSON we need to access all the data in the stream file, we develop a Java program to read and parse the JSON input stream and insert data in the relational database. We used Java language to read JSON streams and import them into relational databases.

[3] http://en.wikipedia.org/wiki/Representationalstatetransfer

4-2-3 Relational Database Creation

Now we have streams of tweets and retweets. We have generated informative fields out of tweet streams and generate a relational database. We define six database tables Tweet, Retweet, User, User Follower, User Follower Network, Entity. We have shown the database schema in figure 4-1. First table is Tweet, we extract *original tweet* from stream of tweets. Original tweet means those tweets which have been written for the first time by a user. These tweets do not have *Retweet Status* section in the tweet stream. We define TweetID as a primary key for this table. In retweet table which looks similar to previous one we store the retweets fields. Each retweet connected to its parent tweet by *ParentTweetId*. In this table TweetId is also a primary key.

Next we extract the user profile into *User* table, each user can have several tweets which connected to tweet or retweet table by *UserID*. In *User Follower Network* table, followers of each user are listed, we are going one step further and will reach the information of user-followers and make another table called *User Follower* which is aggregation of all users followers information. In the last table we have gathered information of tweet-entities which contains Hashtags, Urls, User-mentions and its connected to tweet or retweet table by TweetID.

4-2-4 Specification of Datasets

In this section we will explain in more detail the specification of the four datasets and describe the basic statistics about them.

Datasets	# Tweets	# User	# Retweet	Duration
Steve Jobs Death	19800	11155	158133	4 Day
US Election	253680	57267	741064	16 Day
Foxnews Obama Assassination	66245	41873	158133	4 Day
:)	13956	1162	21578	1 Day

Table 4-2: Dataset overall statistical information

More detailed statistics about the four dataset are listed in tables4-2 and 4-3. The following tables give us an insight on how the data are distributed and what are the differences of the datasets in terms of detailed statistics.

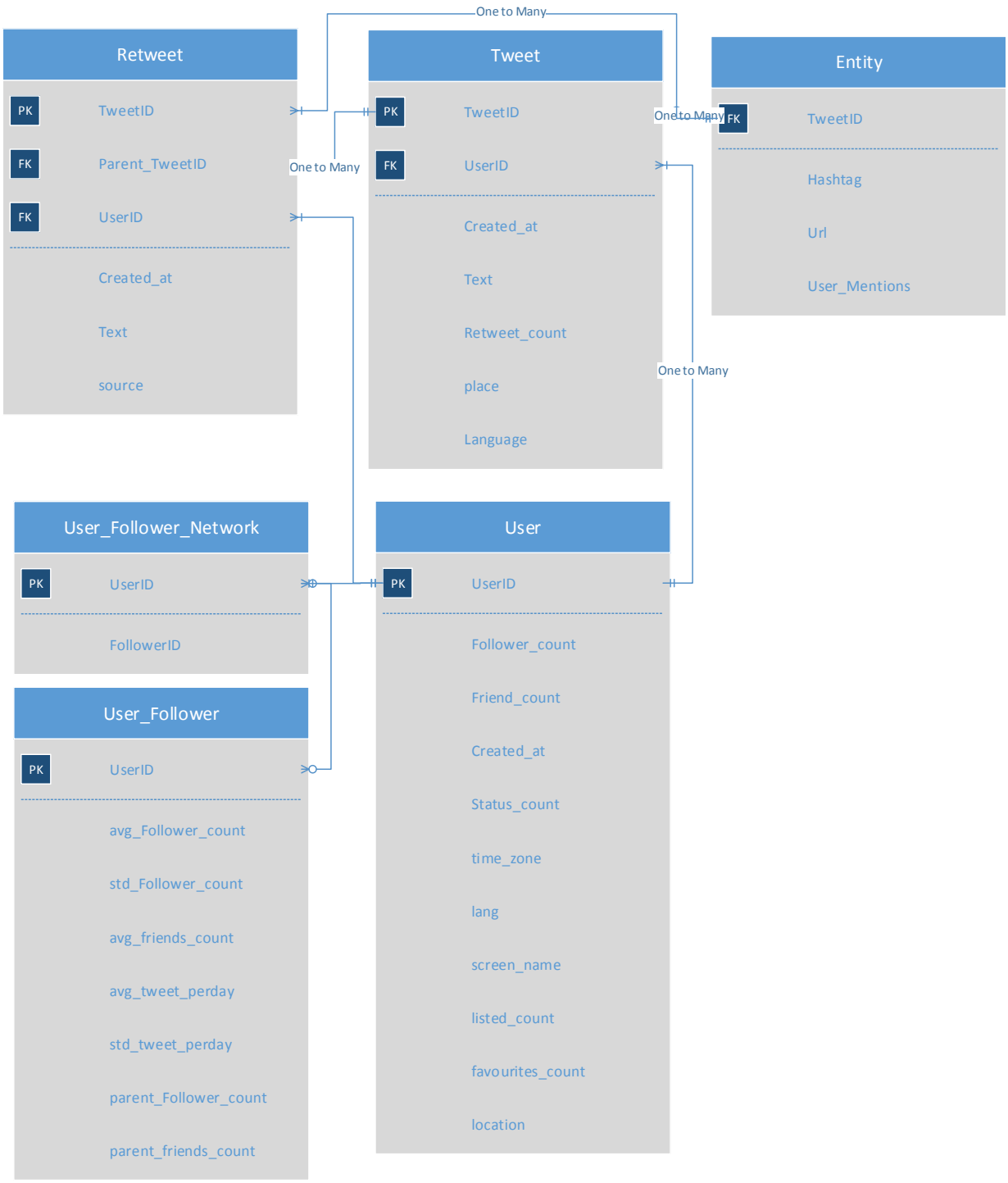

Figure 4-1: Relational database schematic model

Dataset	Fox news		:)		Steve jobs		America Election	
Features	Skewness	kurtosis	Skewness	kurtosis	Skewness	kurtosis	Skewness	kurtosis
$\#fo$	626.85	21.52	778.5	25.384	676.35	20.30	808.1	24.11
$\#fi$	468.03	17.30	244.3	13.405	281.23	13.71	865.1	23.39
$\#tweet/day$	52.60	5.35	13.4	2.928	80.99	7.56	27.2	4.02
$\bar{fof}$	1095.68	25.75	544.6	18.095	3556.01	44.46	2181.7	33.82
$\bar{fof}$	110.89	8.20	72.0	6.979	116.58	8.47	137.4	9.11
$\bar{rof}$	48.27	4.79	46.1	5.112	58.43	5.19	100.3	5.90
$\bar{tof}$	3.83	1.43	1.1	0.706	15.46	2.05	4.0	1.55
σ_{rof}	13.37	1.81	14.0	1.831	17.47	2.52	13.8	1.75
$\#Retweet$	84.26	8.33	262.7	13.123	153.49	11.25	426.8	18.87
$\bar{t\_time}$	1256.30	32.97	854.3	26.706	1092.50	29.48	4644.1	61.47
σ_{t_time}	4.21	2.26	16.9	4.069	7.74	2.87	2.6	1.86
t_h	-0.95	-0.72	-1.1	-0.094	-0.91	-0.33	-1.6	-0.14

Figure 4-2: Skewness and kurtosis of different features

The table 4-2 and 4-3 list skewness and kurtosis of features in four datasets before and after log transformation.

Skewness quantifies how symmetrical the distribution is, a symmetrical distribution has a skewness of zero. An asymmetrical distribution with a long tail to the right (lager value) has positive value and data with long tail to the left has negative value. There is a rule of thumb to indicate skew distribution, if the skewness is greater than 1 (or less than -1) the skewness is substantial and the distribution is far from symmetrical. [4]

Kurtosis quantifies whether the shape of the data distribution matches the Gaussian distribution. A Gaussian distribution has a kurtosis of zero and a flatter distribution has a negative kurtosis and a distribution with sharper peak has positive kurtosis.[5]

We do log transformation to make sure the data is less skew and sharp. As you can see in table 4-3 after log transformation the skewness and kurtosis of data decreased.

4-2-5 Splitting Methods of Datasets

In order to test the performance of our classifier, we need to exactly define how we split the dataset for training and testing and motivate our choices

[4] http://graphpad.com/guides/prism/6/statistics/index.htm?stat_skewness_and_kurtosis.htm
[5] http://en.wikipedia.org/wiki/Kurtosis

Dataset	Fox news		:)		Steve jobs		America Election	
Features	Skewness	kurtosis	Skewness	kurtosis	Skewness	kurtosis	Skewness	kurtosis
$\#fo$	0.5873	0.35	1.062	0.6703	0.27	0.587	0.830	0.313
$\#fi$	1.3348	-0.23	1.839	0.0513	1.48	-0.172	1.841	-0.156
$\#tweet/day$	0.8225	-0.69	1.329	-0.9784	0.49	-0.489	0.743	-0.642
$\bar{fof}$	1.7685	-0.13	0.112	-0.0021	0.57	0.056	1.789	0.067
$\sigma_{\bar{fof}}$	5.4111	-1.11	1.497	-0.7716	3.02	-0.797	3.724	-0.605
$\bar{rof}$	0.0015	-0.19	-0.126	0.0487	-0.30	0.109	-0.053	-0.115
$\bar{tof}$	6.4100	-0.82	30.115	-3.3161	2.35	-0.310	1.302	-0.284
σ_{rof}	9.6730	-1.73	10.504	-1.9932	6.67	-1.273	8.378	-1.472
$\#Retweet$	10.5118	3.14	22.781	4.5129	16.11	3.817	10.495	3.039
$\bar{t\_time}$	2.3174	1.41	0.184	0.9260	0.49	0.734	2.293	1.246
σ_{t_time}	0.7678	1.63	7.965	3.1087	2.85	2.169	-0.436	1.206
t_h	0.4032	-1.38	-0.961	-0.3340	1.52	-1.557	-0.894	-0.773

Figure 4-3: Skewness and kurtosis of **Log-Transformation** of different features

to do so. Data splitting strategies usually are not well defined in previous studies.

We define two different strategies, in order to split data to be able to see whether chronological splitting has any differences (in performance) over other splitting methods on predicting the popularity of tweets. The two splitting methods are as followings:

- Chronological splitting:

  The idea of chronological splitting is to divide the train and test set based on the time of tweets. All the tweets and retweets up to a certain point of time are considered training set and the tweets and retweets in later times are considered as a test set. The motivation of splitting dataset chronologically is based on the fact that in a real popularity prediction scenario we don't know about future tweets but only about tweets that are published until the point of prediction.

  We created different splits on our dataset based on the number of days which are considered as train or test set. Figure 4-4 illustrates our chronological data splitting method. later in this chapter we will show the performance of the classifier on different splits we defined.

- Random splitting:

  Although the idea of chronological splitting seems to be logical, we also split our datasets randomly to see whether or not time-aware splitting

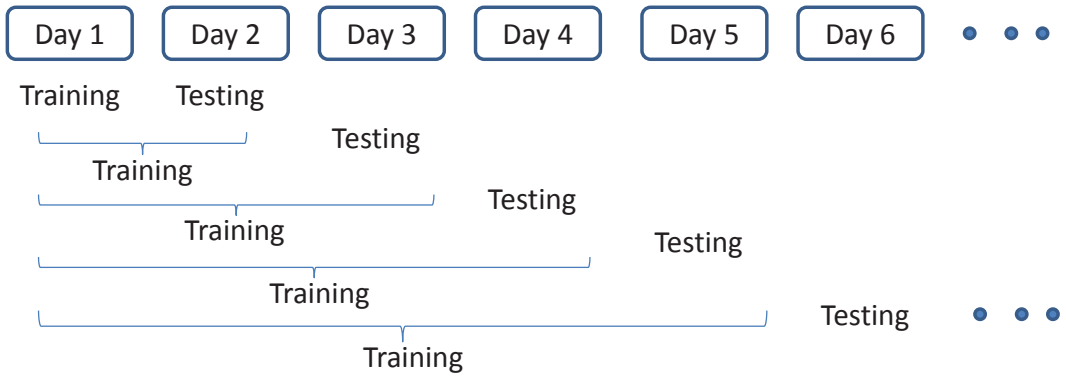

Figure 4-4: Chronological data set splitting

has any influence in the performance of classifying. Further more, random splitting allows us to perform cross-validation on the dataset to make sure the test results are stable among different splits.

In random splitting, depending on weather we want to perform cross-validation or not, we split all tweets into different sets. For each tweet, the number of its tweet would be considered as class label. Figure 4-5 illustrate the random splitting method for train/test set and cross-validation scenarios.

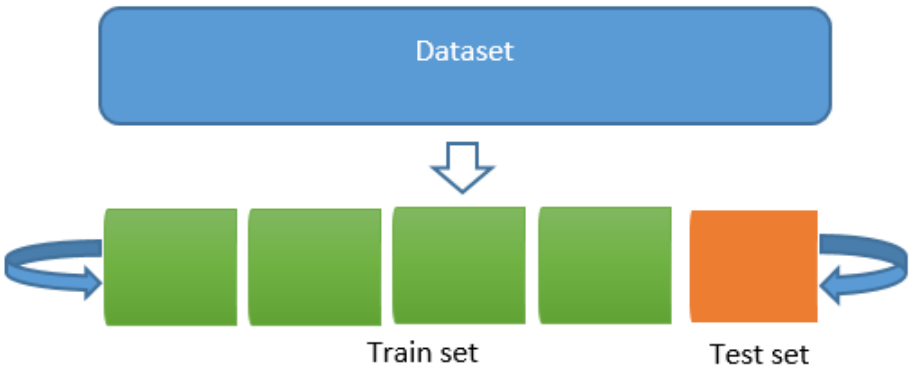

Figure 4-5: Random data set splitting

4-3 Implementation of Classifier

In this section we describe in detail how we performed our experiments, how we built the optimal classifier and how we evaluate them. As we explained in Chapter 3, the intent of our classifiers is to predict whether a newly published tweet would be popular or not. That is whether or not a tweet would be retweeted a certain amount of time which we call it *popularity*. An important decision is to define which kind of tweets should be considered as popular and which not. To allow a flexible definition of popularity we consider different retweet-counts, as a *threshold* for popularity call it *popularity threshold*.

Table 4-3 lists different popularity threshold that we defined and the percentage of tweets that have retweet-counts more than that threshold. As you can see in this table, by having a higher threshold, the percentage of popular tweet (tweets who belongs to the positive class) would be lowered.

This is particularly of interest to us, specially to see whether or not the task of classification would be more difficult, when the number of samples in the positive class (i.e tweets with retweeted more than threshold) are lower.

Popularity Threshold	Steve Job	US Election	Foxnews	:)
5	7.6%	9.07%	6.8%	3.07%
10	3.2%	3.5%	2.9%	1.2%
15	1.8%	2.1%	1.8%	0.72%
20	1.3%	1.5%	1.3%	0.48%
40	0.6%	0.6%	0.5%	0.107%
80	0.2%	0.2%	0.18%	0.042%

Table 4-3: Percentage of popular tweets based on different popularity thresholds

4-3-1 Evaluation Metrics

As we mentioned in the first chapter, our goal is to predict the popularity of tweets. It is necessary to explain what we mean by good prediction and how we can measure such a prediction. The evaluation methods adapted for classification performance play a critical role in design and choosing classifiers, especially when we are faced with an imbalanced data set. Imbalanced data set means having at least one class in minority relative to others. This would be a challenging problem in real world machine learning usage.

To evaluate the performance of our classifiers different metrics can be used. Depending on the goal of problem the different choices for evaluation metric can be made. We first introduce the common evaluation metrics that have been use for binary classification tasks. Examples of these measurements are; Error rate, Recall(Sensitivity, TPR), Precision, Specificity, FPR , F1-measure and Youden Index. Below the definition of common evaluation metrics that can be used for our problem are described:

True Positive Rate(TPR) or *Recall*, assesses to what extent all the examples that needed to be classified are to be considered as positive. If a positive sample is classified as positive, it is counted as a *true positive*.

$$\textit{true positive rate} = \frac{TP}{TP + FN} = Sensitivity \tag{4-1}$$

True Negative Rate(TNR) (Recall -) it is the percentage of negative examples correctly classified within negative class. If the class label of a sample is negative and it is classified as negative, then it is counted as a *true negative*.

$$\textit{true negative rate} = \frac{TN}{TN + FP} = Specificity \tag{4-2}$$

False Positive Rate (FPR) It is the percentage of negative examples misclassified as belonging to the positive class.

$$\textit{false positive rate} = \frac{FP}{FP + TN} = 1 - Specifity \tag{4-3}$$

Precision addresses the question: "Given a positive prediction from the classifier, how likely is it to be correct ?" It is the proportion of positive examples that are actually positive, representing how accurate the learning method is. however in imbalance classes since False Positive Rate would be greater than True Positive Rate then this would affect Precision and it will not be a very useful. as we are more interested to correctly find the positive samples while precision penalizes the classifiers who retrieve high number of TPs and high FPs. Therefore a good calculation metric for us should exploit well both precision and recall evaluation metrics.

$$\textit{Precision} = \frac{TP}{TP + FP} \tag{4-4}$$

F-measure is the harmonic mean of precision and recall. β is a parameter that controls balance between $Precision$ and $Recall$. When $\beta = 1$, F_1 comes to be equivalent to the harmonic mean of $Precision$ and $Recall$, if $\beta > 1$ F become more recall-oriented and if $\beta < 1$ it becomes precision-oriented.

$$F_1 = (\beta^2 + 1)\frac{\text{precision} \cdot \text{recall}}{\beta^2 \text{precision} + \text{recall}} (0 \leq \beta \leq \infty) \tag{4-5}$$

Youden' Index simply is the sum of sensitivity and specificity. Motivation of this measurement is maximizing the sum of $Recall^+$ and $Recall^-$. Conceptually, the Youden metric measures the maximum vertical distance between the ROC curve and the diagonal line.

$$Youden = sensitivity + specificity - 1 \tag{4-6}$$

AUC (Area Under the ROC Curve) This metric measures the area under the Receiver Operating Characteristic (ROC) curve. ROC curve present the trade-off between the true positive rate and false positive rate, as the threshold is varied from $-\infty$ to $+\infty$. The threshold allows the end user to tune a classifier in order to trade-off FPs for FNs or vice versa. The area under curve is often used to summaries a classifiers performance into a single quantity, which represent the performance of a classifier in general and the larger the AUC the better the performance.

This metric is influenced by the confidence of classifiers for positive and negative classes and influenced less by the number of currently classified samples. Since in our problem we are more interested in finding the positive samples, this metric is not the best choice for us.

MAP (Mean Average Precision) This metric measures the area under the *Precision-Recall* (PR) curves. PR curves like the operating characteristic (ROC) curves, are an evaluation tool for binary classification that allows visualization of performance at a range of thresholds. In practice, to calculate

the MAP metric, we should average over all precision results based on different values of threshold Boyd et al. (2013). The thresholds specifies the decision boundary by which a sample is decided to be in the positive or negative class.

Figure 4-6 illustrates the precision-recall curve for different classifiers that we are using in this work. This curve illustrates the changes between precision and recall when the decision boundary of the classifier is changed. The advantage of MAP over other evaluation metrics is that it can find a proper trade-off between precision and recall. As this graph shows, for high values of recall, the precision drops significantly. We therefore define a upper bound threshold for the recall so that the overall MAP does not influenced by low precisions when recall is high. We experimentally found that the threshold of 0.7 is a good upper bound for recall.

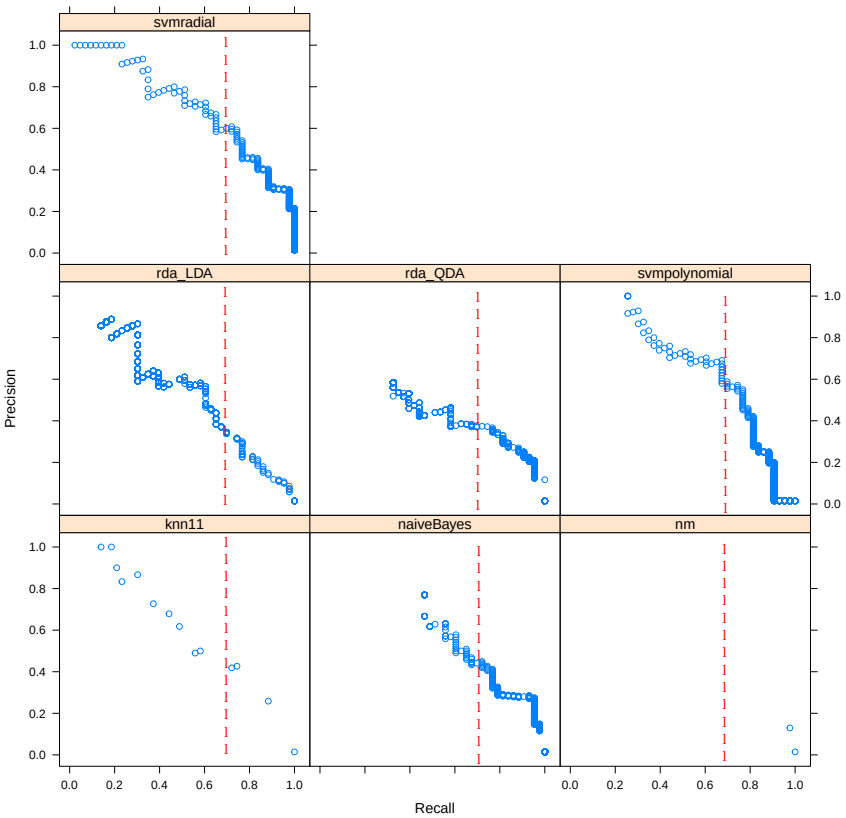

Figure 4-6: Precision-Recall curve for different classifiers we used in this work. The red vertical dashed line specifies the upper bound for the recall

As mentioned earlier the choice of evaluation metric is quite important to properly evaluate the performance of our models. In our problem however, the classes are highly imbalanced, that is, the distribution of position and negative samples are different dramatically. This of course depends on the choice of popularity threshold. In depending on the values of popularity threshold, the distribution of positive and negative classes can be different. Table 4-4 shows the number of positive and negative samples based on two different popularity thresholds: a low threshold value of 5, and a high threshold value of 100.

As you can see in table 4-4, the number of positive samples are much lower compare to the number of negative samples, specially for the hight threshold value. Due to imbalance nature of classes distributions evaluation metrics such as accuracy are not good indicator of classification performance.(Weiss, 2004)

In fact we are interested to find positive samples as much as possible, that is, true positive rate. With accuracy the true negative rate is also taken into account, which is not particularly interesting for us.

	Low Threshold 5		High Threshold 100	
Dataset	# + Sample	# - Sample	# + Sample	# - Sample
Fox News	1520	18280	28	19772
:)	429	13527	6	13950
Steve jobs	4538	61707	82	66163
America Election 2014	23022	230658	465	253215

Table 4-4: The distribution of positive and negative samples in our datasets for the low and high values of popularity threshold

In the case of learning extremely imbalanced data such as our study, the minority class (i.e. positive class) is our interest. In many cases, it is desirable to have a classifier that gives high prediction accuracy in comparison to the minority class ($Accuracy+$), while maintaining reasonable accuracy for the majority class ($Accuracy-$).

Figure 4-7 illustrates the distributions of retweet counts. As this graph shows, there are many tweets which have low number of retweets while there are much lower number of tweets with high number of retweets.

A good strategy to identify a proper evaluation measure should largely depend upon specific application requirements. Choosing appropriate evaluation measure according to different scenarios can help making correct judgment to the classification performance.

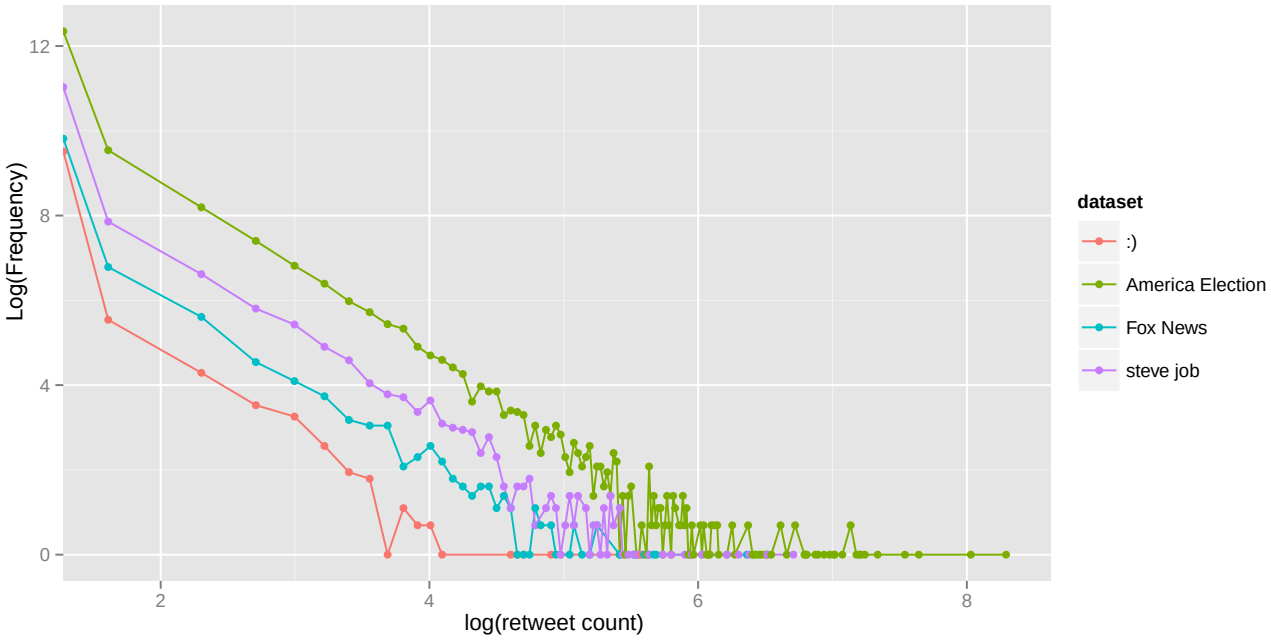

Figure 4-7: Distributions of retweet counts for all dataset

In this thesis our goal is to find all popular tweets (Positive labels). However by maximizing the TPR, FPR will also be increased, therefore we need to use a proper classifier which is considered a trade off and maximizes the accuracy of both positive and negative classes.

4-3-2 Classifier Parameters Setup

In this section we first report the performance of our classifiers based on different parameters (if applicable to a classifier) for each classifier separately, and then we further compare our proposed classifiers using different evaluation metrics and under different data splitting strategies. Before we compare the performance of our proposed classifiers, we will first try to derive the optimal parameters for the classifiers that need to be configured. We later compare the performance of the configured classifiers.

In this section we use all features that we introduced in Chapter 3. Later in this chapter we will investigate the influence of individual features or different combination of features on performance of the classifier.

K-nearest neighbor

For the K-nearest neighbor classifier, the choice of k is quite important. To determine the best possible value for k, we experimentally measure the performance of our K-NN for all datasets. The experiments are done by a 5-fold cross-validation on all datasets and having 20 as the popularity threshold value for positive and negative classes. Later in this section we will report the results based on different popularity threshold and based on different splitting strategies.

In our experiments we used *MAP* metric to evaluate the performance of the classifiers. As we explained earlier, since our data is highly imbalanced, MAP give us the best means of measurement.

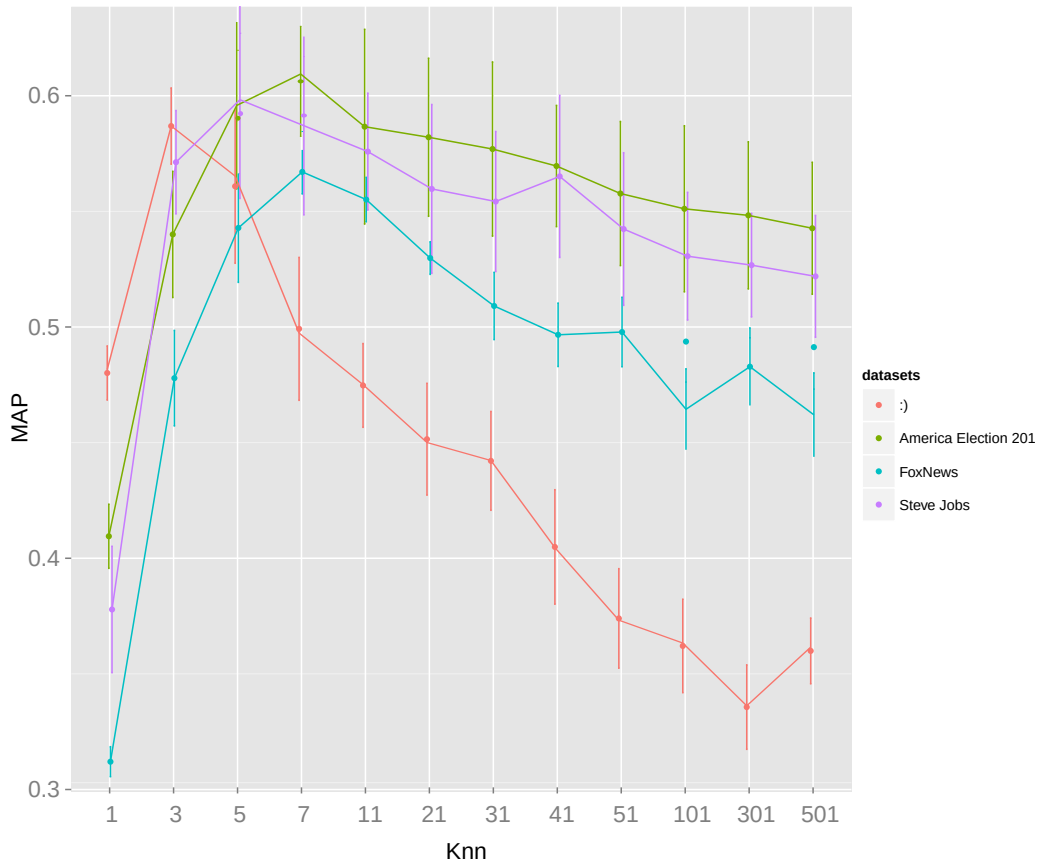

Figure 4-8: The performance of K-NN classifier over different values of k

Figure 4-8 illustrates the performance of our K-NN classifier based on different values of k. The performance of the K-NN classifier is compared with a

baseline which is calculated by classifying all samples to the positive class. As illustrated in figure 4-8, the performance of the K-NN classifier get to its high value when k is 7 for American Election, FoxNews, SteveJobs, dataset while for the smiling dataset the best performance is obtained at k=3.

The difference in the optimal value of k for the smiley dataset compare to other datasets, is most probably due to the fact that there are very few positive samples in this dataset. In fact by increasing the value of k, it become more probable hat decisions are influenced by negative samples. we used the optimal value of k that are obtained in this step to compare K-NN with other classifiers.

SVM

Similar to K-NN, we need to set some parameters for SVM to obtain the optimal SVM for all datasets. In SVM, the choice of kernel, parameter gamma and cost parameter (C) are important. Similarly we tested the performance of our SVM-classifier based on different values of C,$Gamma(\gamma)$ and also based on different kernels. The experiments are setup similarly by having threshold value as 10 and by doing a 5-fold cross-validation on datasets.

In figure 4-9 the performance of our SVM classifier is illustrated based on different combinations of parameters C and Gamma in term of the MAP score. As it can be seen in this graph the performance of the SVM classifier can change dramatically depending on the values of these two parameters. However the pattern og performance change in all dataset is almost the same and combination of $c = 10$ and $Gamma = 0.01$ result the best performance for all datasets. The results in figure 4-9 are based om a linear kernel SVM. In the next subsection we est the performance of our SVM classifier based on different kernels.

The choice of kernels

SVM Classifier can be used based on different kernels. Depending on the nature of data, in various scenarios , different kernels can perform significantly different. The kernels in SVMs transfer the data points into another feature space in which the model can learn better. In this work we have adopted four standard kernels for our SVM classifier. Below the definition of these four kernels are listed.

The parameters and their possible values that are used in table 4-5 are the followings:

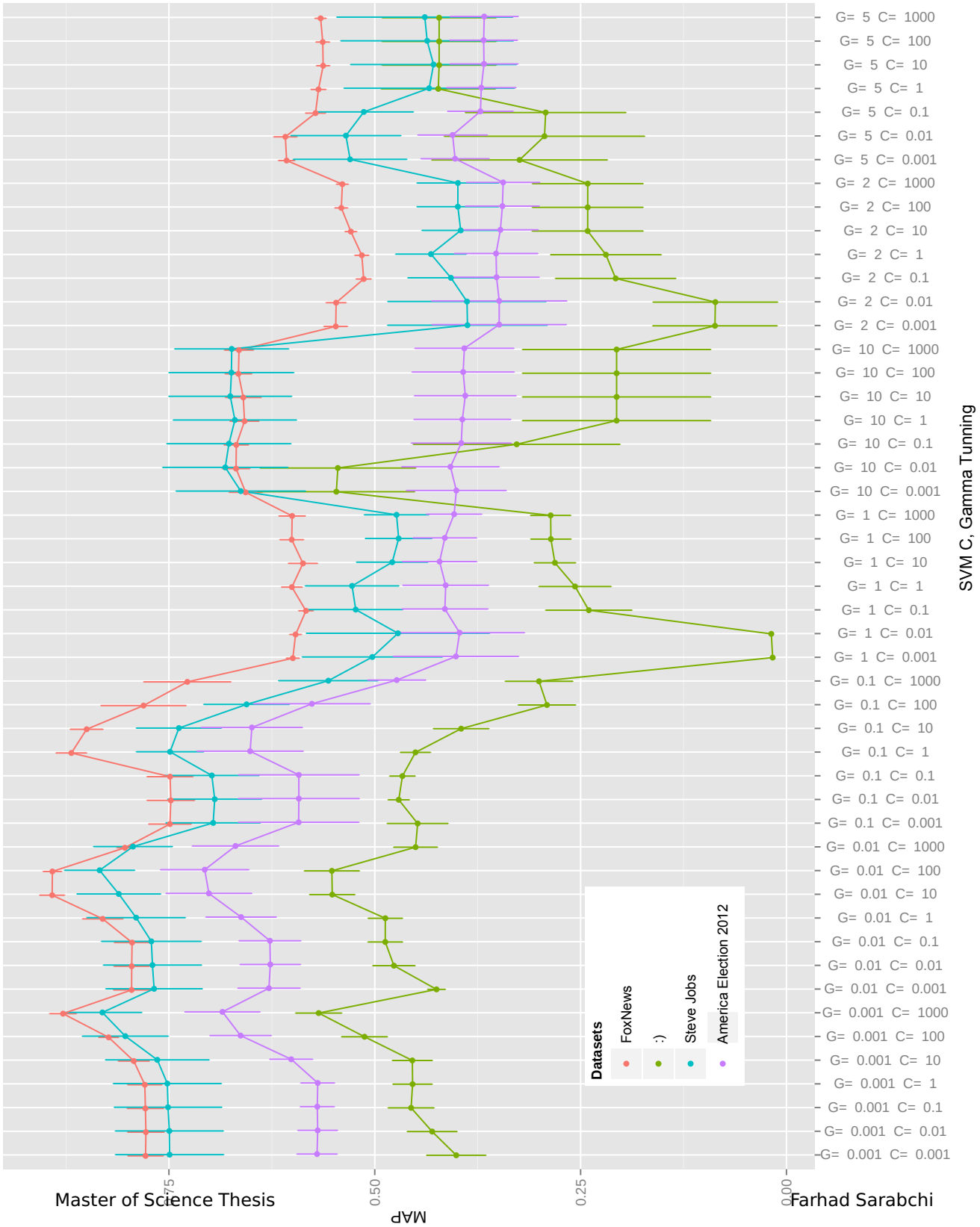

Figure 4-9: Tunning the parameters C and Gamma for the linear SVM classifier

Table 4-5: SVM kenels

Name	Definition
linear	$K(x_i, x_j) = x_i^T x_j$
Radial Basis	$K(x_i, x_j) = e^{-\gamma \|\|x_i - x_j\|\|^2}, \gamma > 0$
polynomial	$K(x_i, x_j) = (\gamma x_i^T x_j + coef0)^{degree}, \gamma > 0$
Sigmoid	$K(x_i, x_j) = \tanh(\gamma x_i^T x_j + coef0)$

- degree: Parameter degree of a kernel function (POLY).
- gamma: Parameter γ of a kernel function (POLY / RBF / SIGMOID).
- coef0: Parameter coef0 of a kernel function (POLY / SIGMOID).
- C-value: Parameter C of a SVM optimization problem (C SVC / EPSSVR / NU SVR).

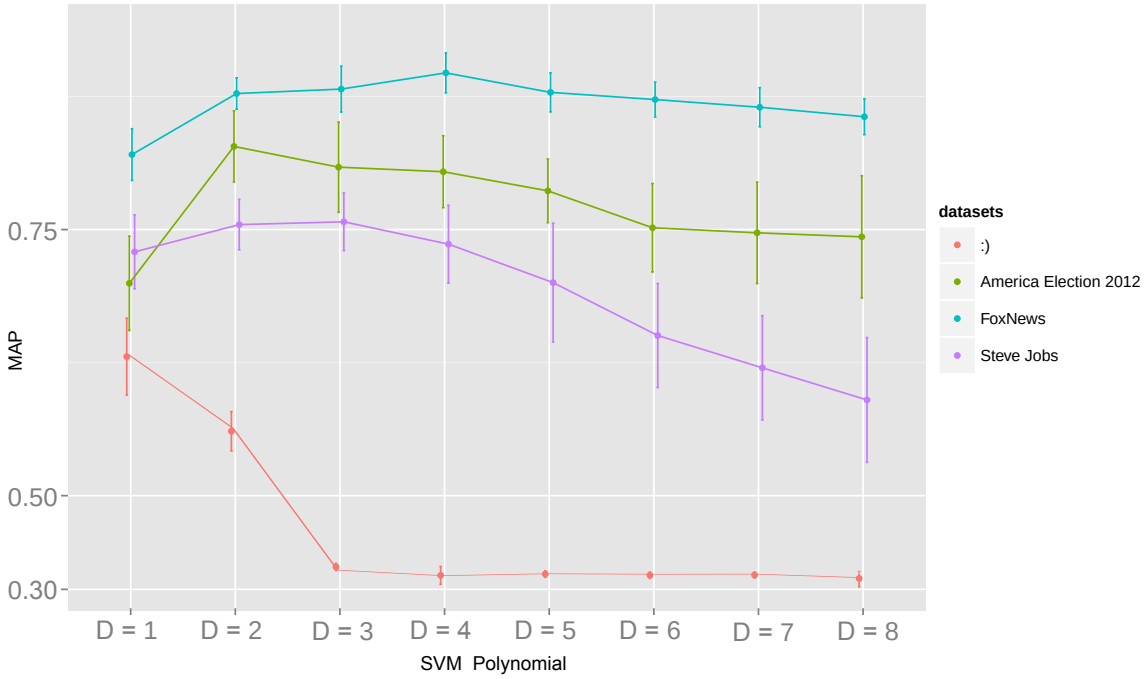

Figure 4-10: The performance of polynomial SVM classifiers based on different degrees of polynomial function.

The polynomial kernel itself can be varied depending on the degree of polynomial function. In figures 4-10 , the performance of our SVM classifier

with polynomial kernel is shown. The horizontal axes represent the degree of polynomial kernel. Based on the results of this graph, $D = 2$ is the best design choice as degree of our polynomial kernel. We further compare this polynomial kernel with other kernels.

Similar to other design choices in SVM, we compare the performance of our SVM classifier based on different kernels on all datasets. Figure 4-11 shows performance of our SVM classifiers based on different kernels in term of MAP. This group suggests that radial-basis kernel perform rather well compared to the other kernels. We therefore consider this kernels the most suitable kernels for our problem.

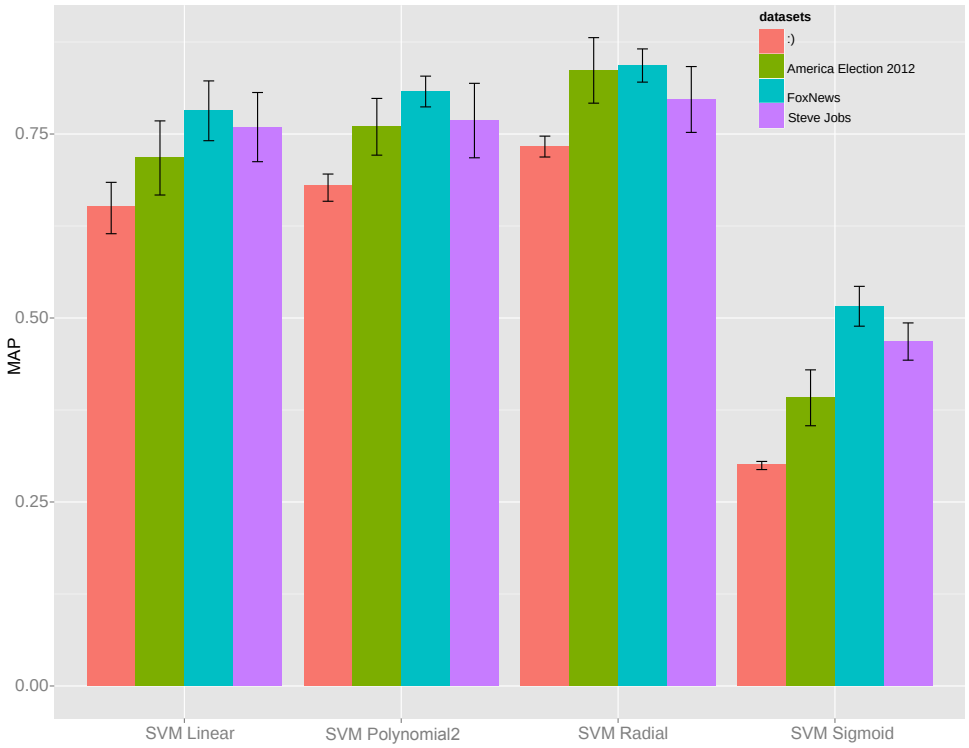

Figure 4-11: The performance of the SVM classifiers based on different kernels

Comparison of different classifiers

In this section we compare the performance of all classifiers that we introduced in Chapter 3 to see which classifiers are the optimal choices in different scenarios. In the previous section the optimal parameters for classifiers

that need parameter setup has been chosen. We now compare the results of all classifiers using same experimental setup. Figure 4-11 illustrates the performance of all classifiers which we have introduced in this work.

The experiments are done based on a 5-fold cross-validation on each dataset. Here we also used the MAP metric to evaluate the performance of different classifiers the popularity threshold is considered as 20 as the default value for threshold. Later we investigate the role of threshold value for different classifiers. The results in this graph shows that in our problem simple classifiers such as LDA perform quite well compare to more advanced classifiers.

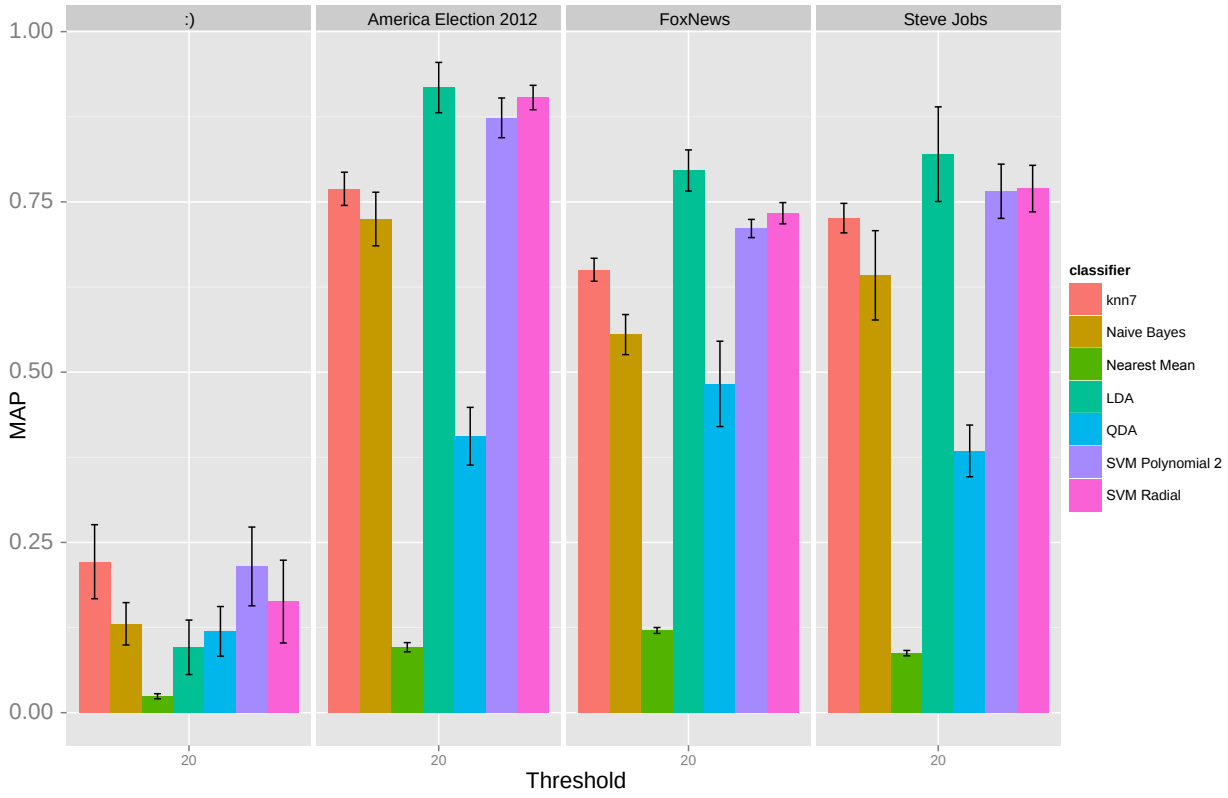

Figure 4-12: Performance of different classifiers based on the popularity threshold of 20.

As the figure shows,the LDA classifier performs better in most cases. the performance of QDA for example is always worse than LDA. This can be due to the fact that QDA tries to learn covariance matrix for each class separately while LDA assumes unique covariance for each class label separately. As the number of positive instances decrease by increasing the threshold value, it

becomes more difficult to predict the covariance of positive class. This can drop the performance of QDA classifier.

Another interesting observation in our results is that the performance of classifiers in all datasets have more or less same pattern. This mean that our classifiers are not dataset-sensitive. The LDA, SVM. and K-NN classifiers always perform he best.

Influence of Threshold

As we discussed earlier in this chapter the choice of popularity threshold is important for designing our experiment. We have further performed additional experiments on all datasets to see how much the results might vary if we consider different threshold values. Figures 4-13, 4-14 illustrate the performance of all classifier compared to each other based on two different threshold values: 5 as a low threshold value and 100 as a high threshold value.

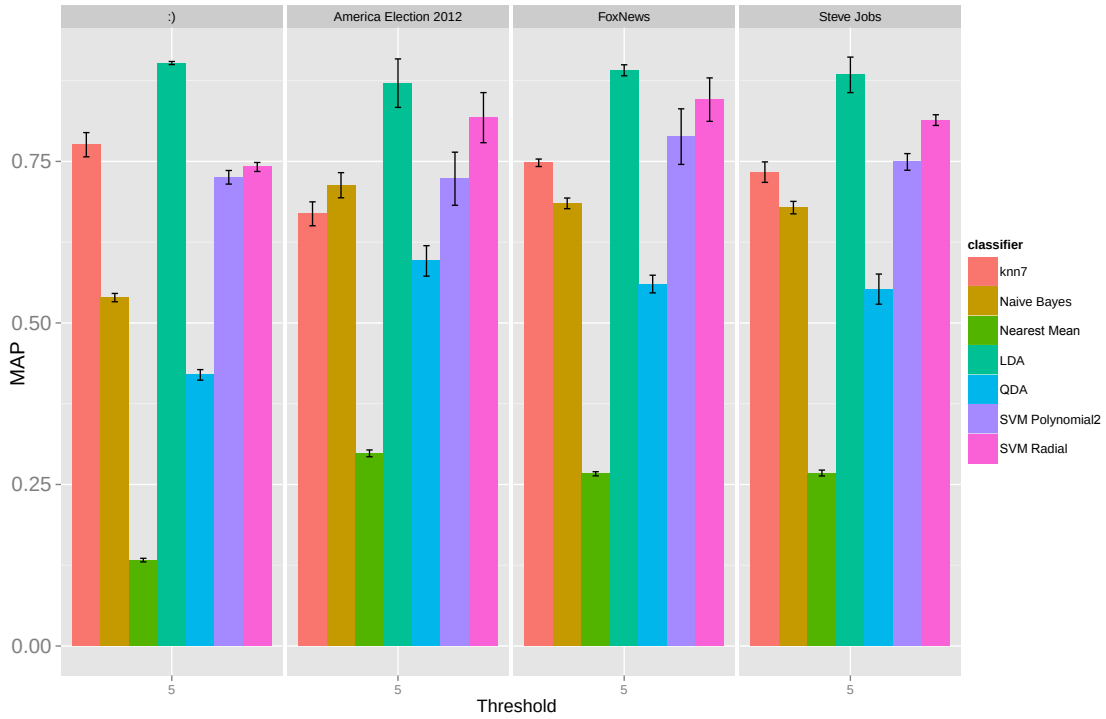

Figure 4-13: Performance of different classifiers based on the popularity threshold of 5.

Not surprisingly the performance of different classifiers are varied for different values of threshold. The graph suggest that some classifiers perform better than others for small values of threshold while others might perform better for higher values of thresholds. This is quite interesting for us because depending on the problem that we are interested in, a different classifier should be chosen. Identifying which classifiers are the most suitable classifier for a particular threshold, helps us to come up with the best design choices for our problem.

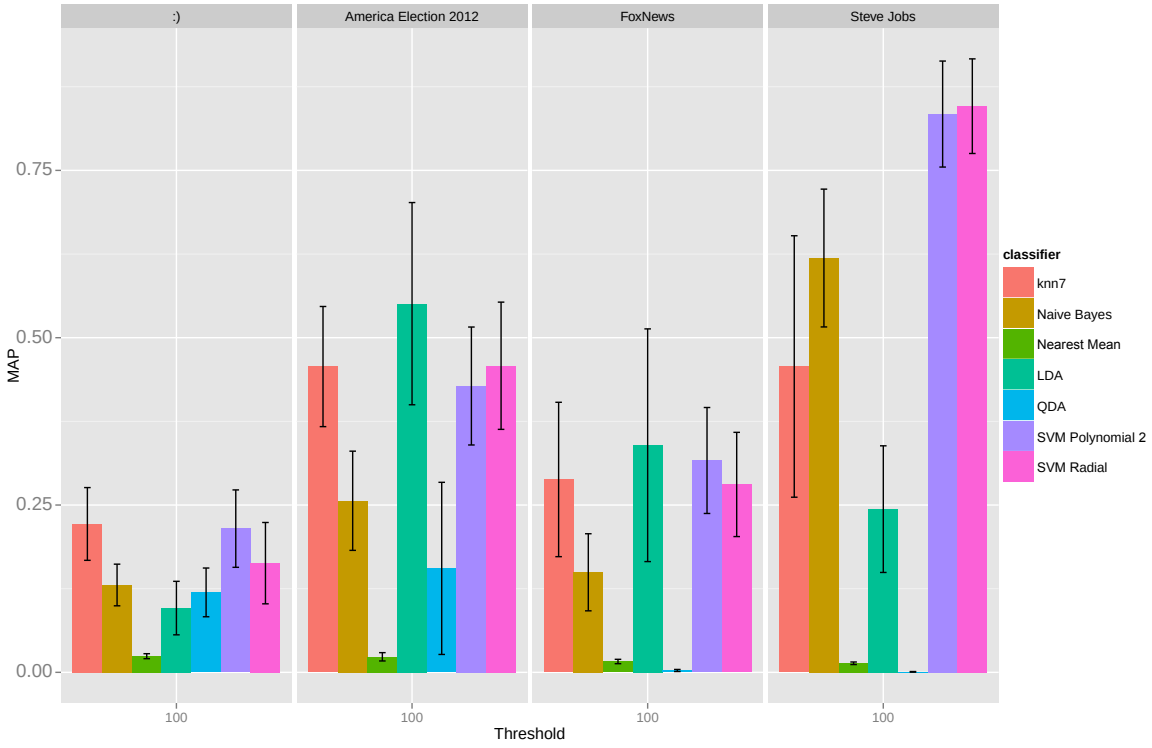

Figure 4-14: Performance of different classifiers based on the popularity threshold of 100.

However an important question here is; why a particular classifier performs better than others in a particular threshold. The answer to this question reflects the fact that different classifiers are suitable in different scenarios. In our case, as illustrated in the two figures, the *QDA* classifier is performing rather well for $T = 5$ while it perform very bad for high threshold value of 100. this is again most probably due to the fact that for high threshold values there are not enough positive samples to derive the right distributions for non-Linear classifier such as QDA. On the other hand linear classifiers such

as LDA and SVM show more stable behavior when the threshold value is changing.

The differences in the results depend on the distribution of data in different datasets. In fact the properties of datasets makes a particular classifier for a dataset the best choice and for an other dataset non-optimal choice.

As we discuss earlier, these result can be explain by fact that the data set is highly imbalanced. For the case of smiley dataset, as Figure 4-7 shows, this dataset has a smaller number of samples having high retweet count. This means classification of samples to positive class is extremely difficult. Therefor its more challenging for classifier to learn the distribution of positive classes when the number of samples is very low.

Generally we can conclude that there is no *universal best* classifier for the task of popularity prediction in twitter social network. It is the task of the designer to chose best classifiers and parameters based on the question of interest and available information.

Influence of Splitting Strategy

As we mentioned earlier in the chapter, the train and test sets can be splitting based on different strategies. In order to see the influence on time in performance of classification we also performed additional experiments on different train and test sets which are spitted chronologically based on time.

In chronological data splitting strategy , all the tweets before a certain time are considered as train and the rest considered as test set.

To be able to compare the performance of the chronological splitting with random splitting, we have split the datasets into a 80% train and 20% test sets. To generate these splits, the tweets are sorted based on tweet time in ascending order and the first 80% of tweets are considered as train set and the rest are considered test set. The size of splits are exactly the same size as the size of train and test sets in our 5-fold cross-validation methods.

In figure 4-15 illustrate the performance of the two splitting strategies on different classifiers based on the MAP score. The experiments are based on having threshold value equal to 5 (low threshold) and 100 (High threshold).

As the results in the above, graph shows the performance of different classifiers are very similar regardless of splitting strategy. In the smiley dataset however, chronological splitting is not a good splitting strategy for high threshold. A more detailed exploration on this dataset revealed that there is

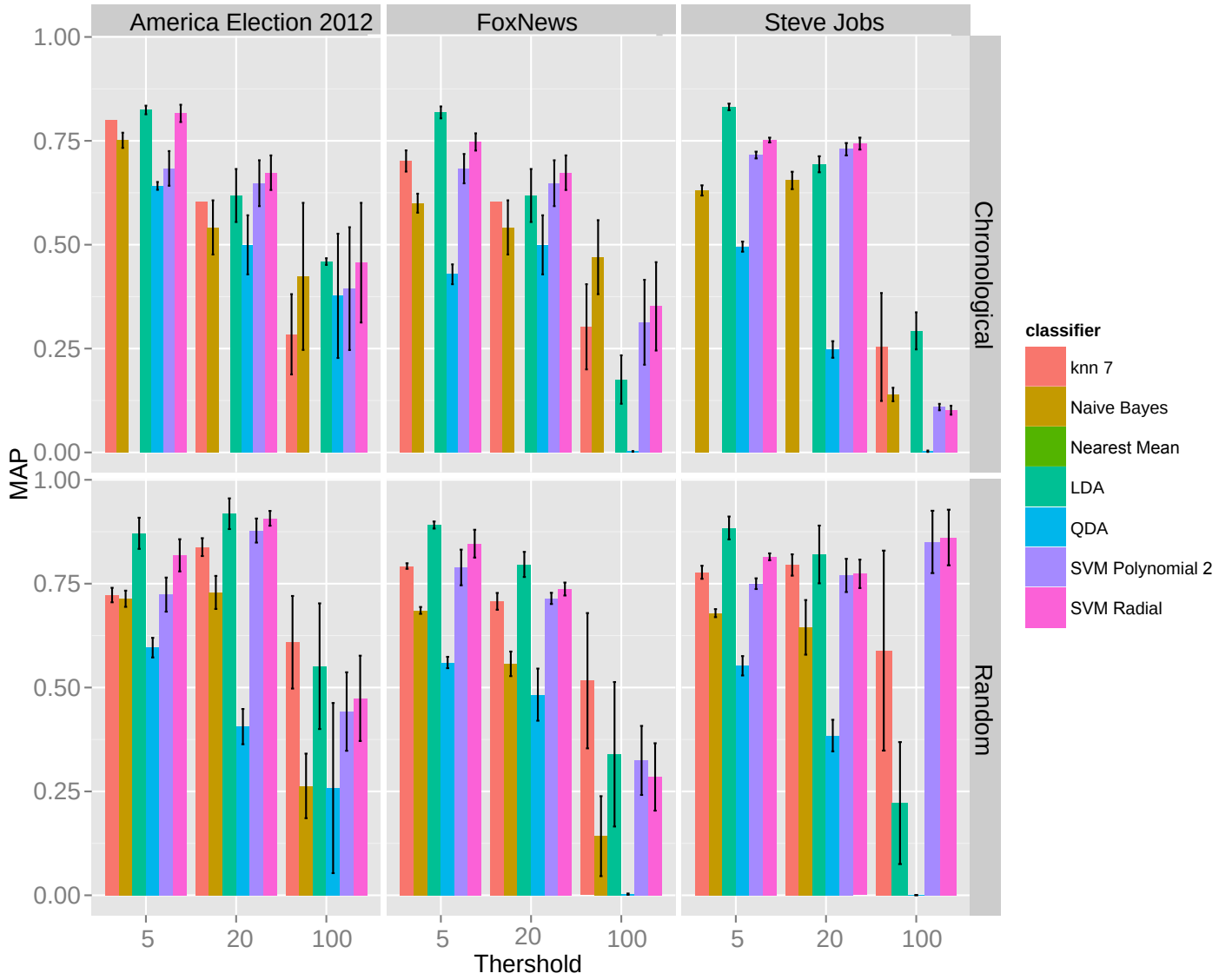

Figure 4-15: The performance of all classifiers on different thresholds with different splitting strategies.

no tweets getting more than 60 retweets and that is most probably because dataset is collected only within one day.

Generally we can conclude that splitting chronological s as good as splitting randomly, this is important because in real-cases, if we want to develop a system which predict the popularity of tweets we can only relay on the information that are generated in the past.

Another remarkable observation in figure 4-15 is that the variance of all classifiers for high threshold is higher than the variance for low threshold. This is most probably due to the fact that the distribution of classes for higher threshold are more unbalance and the results are less stable. Usually in larger dataset, if we have enough samples from each class, the result would be more stable.

4-4 Summary

In chapter we have explained in details the datasets, their collection method and experiments.

We collected four different datasets from twitter having an special topic to make sure that experiments are performed on diverse set of data. We have adapted different classifier and compared their performance on all four datasets. We also come up with a more flexible definition of popularity by defining a threshold on number of retweets.

Our results show that while particular classifier may perform well on a certain threshold, another classifier might perform better for other thresholds. This is not surprising due to the fact that based on different thresholds the distribution of data in positive and negative classes change dramatically and that influence on performance of classifiers.

To show that our conclusions are reliable, we performed all our experiments on all dataset based on a 5-fold cross-validation to make sense that the classifier are not over-trained on a particular test set or on a particular dataset.

Chapter 5

Conclusions and Future Works

5-1 Conclusions

Predicting the popularity of content in social netwroks has attracted several research activities in the past few years. In the case of Twitter social network, predicting the popularity of tweets is quite important for several applications such as viral marketing, personalization and popular news detection.

In this work we proposed a statistical learning approach that extracts different type of features from tweets and tries to predict whether the tweet be a popular tweet or not.

We experimentally tested our approach using four datasets that we collected from twitter. The four datasets were collected using the twitter steaming API. We converted the datasets into a relational database to make it easier to process data and extract features. We extract several user-based and tweet-based features from the body of tweets and the users who published the tweet. Furthermore, we built some additional features from the network of the users which showed to be very informative.

In order to see whether the popularity of tweets can be predicted or not, we developed several statistical classifiers which are trained based on the features that we extracted. The goal of classification task is to predict whether a tweet can be popular or not, that is, whether or not a tweet can get sufficient number of retweets, more than a specific threshold.

We employed several different classifiers and performed a comprehensive set of experiments to find out the optimal classifiers for this prediction problem. We performed different data splitting strategies and tested out approach on different dataset to make sure that our methods is generalizable enough.

The classifiers that we adopted in this work are: SVM, LDA, QDA, Nearest Mean, KNN and Naive Bays. As the choice of parameters in different classifiers are important, we did several experiments in order to find the optimal parameters for the classifiers. We performed all experiment with cross-validation to make sure that the parameters are not optimized for one particular test set.

For the parameter-sensitive classifiers, we found that KNN with small values of K performs better than larger values of K. Also we found that the SVM classifiers perform better when polynomial or radial-basis's kernels are used.

We found that the performance of classifiers are sensitive to the distribution of positive and negative classes. In fact the more imbalance the distribution of classes are, the more challenging is to train the classifiers. To make sure that our classifiers can be fairly evaluated we used the MAP evaluation metric which can better reflect how successful is our classifier to distinguish the positive samples from negative ones.

A key decision to design the classifiers is to specify a popularity threshold value by which the positive and negative classes can be distinguished. In this work, we considered different threshold values to measure how sensitive the classifiers are depending on the popularity threshold. Although the performance of different classifiers with different threshold values are different, still the choice of the right classifier is very important to predict the popularity of tweets. We therefore performed several experiments to measure the performance of different classifiers based on two low and high threshold values. Depending on different threshold values, the performance of our classifiers are slightly different. We found that for lower threshold values the SVM classifiers perform better while for the higher threshold values, which result in more imbalance distributions, simple linear classifiers perform better. This observation can be explained most probably due to the fact that non-linear classifiers need more positive samples to be able to properly learn the distribution of classes.

Furthermore, we did our experiments also based on a chronological data splitting methods, that is, the first few days/hours of a dataset is used as training and the following days/hours are used as test set. We did that to

make sure that our experiments can be modeled with real-case scenarios where we only have access to the past data. Our experimental results show that there is no significant different between the results of random versus chronological splitting methods although for high threshold values the performance of our SVM classifiers is slightly better when the data is split randomly and that is mainly due to the imbalance nature of data.

Our experiment revealed that there is no global best classifier that can always perform good on all datasets and configurations (such as popularity threshold). In fact, depending on the problem and available data, a certain classifier might be the best choice and the other might not. We found that for the scenarios that we have few positive samples, classifiers such as SVM perform better than linear classifiers. In contrast when the dataset is less imbalance, LDA classifier mostly perform better than other classifiers. Depending on the problem and available information the designer can choose the best choice for classifier and its parameters.

5-2 Future Work

Our work can be extended from different point of views for future studies. In this work we examined several classifiers independently. One interesting extension to our work would be to implement fusion and boosting methods to combine all the classifiers and benefit from the advantage of all of them.

Another extension to our work would be to implement some feature engineering methods such as feature extraction to see if more efficient and accurate classifies can be trained. Also techniques such as query expansion can be applied to our problem to exploit additional auxiliary information to improve the performance of our classifiers.

In fact other features from content (content of tweets) can be extracted and used as additional features to improve the performance of classification. Furthermore, more advanced classifiers such as deep neural networks can be employed for this problem to see if they are suitable fo this task or not.

Bibliography

Charu C Aggarwal and ChengXiang Zhai. A survey of text classification algorithms. In *Mining text data*, pages 163–222. Springer, 2012.

Yoav Artzi, Patrick Pantel, and Michael Gamon. Predicting responses to microblog posts. pages 602–606, 2012.

Sitaram Asur and Bernardo A Huberman. Predicting the future with social media. *Computing*, 1(1):492ï£¡499, 2010. URL http://arxiv.org/abs/1003.5699.

Eytan Bakshy, Jake M Hofman, Duncan J Watts, and Winter A Mason. Identifying influencers on twitter. *Communication*, pages 1–10, 2011. URL http://kdpaine.blogs.com/files/twitterinfluencershofmanetal.pdf.

Johan Bollen, Huina Mao, and Xiao-Jun Zeng. Twitter mood predicts the stock market. *Computer*, 2(1):1–8, 2010. URL http://arxiv.org/abs/1010.3003.

E Bothos, D Apostolou, and G Mentzas. Using social media to predict future events with agent-based markets, 2010. URL http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber=5678586.

Danah Boyd, Scott Golder, and Gilad Lotan. *Tweet, Tweet, retweet: Conversational Aspects of retweeting on Twitter*, volume 0, pages 1–10. IEEE, 2010. URL http://www.computer.org/portal/web/csdl/doi/10.1109/HICSS.2010.412.

Kendrick Boyd, Kevin H Eng, and C David Page. Area under the precision-recall curve: Point estimates and confidence intervals. In *Machine Learning and Knowledge Discovery in Databases*, pages 451–466. Springer, 2013.

Meeyoung Cha and Krishna P Gummadi. Measuring user influence in twitter : The million follower fallacy. *Artificial Intelligence*, 146(1):10–17, 2010. URL http://www.aaai.org/ocs/index.php/ICWSM/ICWSM10/paper/download/1538/1826.

Meeyoung Cha, Alan Mislove, Ben Adams, and Krishna P. Gummadi. Characterizing social cascades in flickr. In *Proceedings of the First Workshop on Online Social Networks*, WOSN '08, pages 13–18, New York, NY, USA, 2008. ACM. ISBN 978-1-60558-182-8. doi: 10.1145/1397735.1397739. URL http://doi.acm.org/10.1145/1397735.1397739.

Wei Chen, Chi Wang, and Yajun Wang. Scalable influence maximization for prevalent viral marketing in large-scale social networks. *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, (January):1029–1038, 2010. URL http://portal.acm.org/citation.cfm?doid=1835804.1835934.

D Coomans and DL Massart. Alternative< i> k</i>-nearest neighbour rules in supervised pattern recognition: Part 1. k-nearest neighbour classification by using alternative voting rules. *Analytica Chimica Acta*, 136:15–27, 1982.

Koby Crammer, Ofer Dekel, Joseph Keshet, Shai Shalev-Shwartz, and Yoram Singer. Online passive-aggressive algorithms. *The Journal of Machine Learning Research*, 7:551–585, 2006.

Aron Culotta. Towards detecting influenza epidemics by analyzing twitter messages. *Proceedings of the First Workshop on Social Media Analytics SOMA 10*, 33(2004):115–122, 2010. URL http://portal.acm.org/citation.cfm?doid=1964858.1964874.

D. Scott DeRue and Susan J. Ashford. Who will Lead and Who will Follow? a Social Process of Leadership Identity Construction in Organizations. *Academy of Management Review*, 35(4):627–647, October 2010. ISSN 1930-3807. URL http://amr.aom.org/content/35/4/627.abstract.

Evgenia Dimitriadou, Kurt Hornik, Friedrich Leisch, David Meyer, , and Andreas Weingessel. *e1071: Misc Functions of the Department of Statis-*

tics (e1071), TU Wien, 2010. URL http://CRAN.R-project.org/package=e1071.

S. Fouzia Sayeedunnissa, AdnanRashid Hussain, and MohdAbdul Hameed. Supervised opinion mining of social network data using a bag-of-words approach on the cloud. In Jagdish Chand Bansal, Pramod Singh, Kusum Deep, Millie Pant, and Atulya Nagar, editors, *Proceedings of Seventh International Conference on Bio-Inspired Computing: Theories and Applications (BIC-TA 2012)*, volume 202 of *Advances in Intelligent Systems and Computing*, pages 299–309. Springer India, 2013. ISBN 978-81-322-1040-5. doi: 10.1007/978-81-322-1041-2_26. URL http://dx.doi.org/10.1007/978-81-322-1041-2_26.

Eibe Frank and Remco R Bouckaert. Naive bayes for text classification with unbalanced classes. In *Knowledge Discovery in Databases: PKDD 2006*, pages 503–510. Springer, 2006.

David Mandell Freeman. Using naive bayes to detect spammy names in social networks. In *Proceedings of the 2013 ACM workshop on Artificial intelligence and security*, pages 3–12. ACM, 2013.

L Freeman. Centrality in social networks conceptual clarification. *Social Networks*, 1(3):215–239, 1979. URL http://linkinghub.elsevier.com/retrieve/pii/0378873378900217.

Wojciech Galuba and Karl Aberer. Outtweeting the twitterers - predicting information cascades in microblogs. *Proceedings of the 3rd conference on Online social networks*, 39(12):3âĂŞ3, 2010. URL http://portal.acm.org/citation.cfm?id=1863193.

Mark Goadrich, Louis Oliphant, and Jude Shavlik. Gleaner: Creating ensembles of first-order clauses to improve recall-precision curves. *Machine Learning*, 64(1-3):231–261, 2006.

Mark Granovetter. Threshold Models of Collective Behavior. *American Journal of Sociology*, 83(6):1420–1443, 1978. ISSN 00029602. doi: 10.2307/2778111. URL http://dx.doi.org/10.2307/2778111.

Adnan Hassan, Mohd Shariff Nabi Baksh, Awaluddin M Shaharoun, and Hishamuddin Jamaluddin. Feature selection for spc chart pattern recognition using fractional factorial experimental design. *Intelligent Production Machines and System: 2nd I* IPROMS Virtual International Conference, In: D. T. Pham, EE Eldukhri, and AJ Soroka Ed., Elsevier*, pages 442–447, 2006.

Bernd Heisele, Purdy Ho, and Tomaso Poggio. Face recognition with support vector machines: Global versus component-based approach. In *Computer Vision, 2001. ICCV 2001. Proceedings. Eighth IEEE International Conference on*, volume 2, pages 688–694. IEEE, 2001.

Oliver Hinz, Bernd Skiera, Christian Barrot, and Jan U Becker. Seeding strategies for viral marketing : An empirical comparison seeding strategies for viral marketing : An empirical comparison. *Journal of Marketing*, 75(November):55–71, 2011. URL http://www.marketingpower.com/AboutAMA/Documents/JM_Forthcoming/seeding_strategies_for_viral.pdf.

Liangjie Hong, Ovidiu Dan, and Brian D. Davison. Predicting popular messages in twitter. pages 57–58, 2011a.

Liangjie Hong, Ovidiu Dan, and Brian D. Davison. Predicting popular messages in twitter. In *Proceedings of the 20th International Conference Companion on World Wide Web*, WWW '11, pages 57–58, New York, NY, USA, 2011b. ACM. ISBN 978-1-4503-0637-9. doi: 10.1145/1963192.1963222. URL http://doi.acm.org/10.1145/1963192.1963222.

R Iyengar, C Van Den Bulte, and T W Valente. Opinion leadership and social contagion in new product diffusion. *Marketing Science*, 30(2): 195–212, 2010. URL http://mktsci.journal.informs.org/cgi/doi/10.1287/mksc.1100.0566.

Rabia Jafri and Hamid R Arabnia. A survey of face recognition techniques. 2009.

Bernard J Jansen, Mimi Zhang, Kate Sobel, and Abdur Chowdury. Twitter power : tweets as electronic word of mouth. *Journal of the American Society for Information Science*, 60(11):2169–2188, 2009. URL http://doi.wiley.com/10.1002/asi.21149.

N. Jonnalagedda and S. Gauch. Personalized news recommendation using twitter. In *Web Intelligence (WI) and Intelligent Agent Technologies (IAT), 2013 IEEE/WIC/ACM International Joint Conferences on*, volume 3, pages 21–25, Nov 2013. doi: 10.1109/WI-IAT.2013.144.

E Katz and P F Lazarsfeld. *Personal influence: The part played by people in the flow of mass communications*, volume 21. Free Press, 1955. URL http://www.ncbi.nlm.nih.gov/pubmed/13409025.

David Kempe, Jon Kleinberg, and Tardos Eva. Maximizing the spread of in uence through a social network. *SIGKDD*, 2003.

Haewoon Kwak, Changhyun Lee, Hosung Park, and Sue Moon. What is twitter, a social network or a news media? In *Proceedings of the 19th International Conference on World Wide Web*, WWW '10, pages 591–600, New York, NY, USA, 2010. ACM. ISBN 978-1-60558-799-8. doi: 10.1145/1772690.1772751. URL http://doi.acm.org/10.1145/1772690.1772751.

Chien-Cheng Lee, Shin-Sheng Huang, and Cheng-Yuan Shih. Facial affect recognition using regularized discriminant analysis-based algorithms. *EURASIP journal on advances in signal processing*, 2010:1, 2010.

Kristina Lerman and Aram Galstyan. Analysis of social voting patterns on digg. *CoRR*, abs/0806.1918, 2008.

Jure Leskovec, Ajit Singh, and Jon Kleinberg. *Patterns of Influence in a Recommendation Network*, volume 3918, pages 380–389. Springer, 2006. URL http://www.springerlink.com/content/t85674vqg783/#section=494346&page=1.

Jingxuan Li, Wei Peng, Tao Li, and Tong Sun. Social network user influence dynamics prediction. In Yoshiharu Ishikawa, Jianzhong Li, Wei Wang, Rui Zhang, and Wenjie Zhang, editors, *Web Technologies and Applications*, volume 7808 of *Lecture Notes in ComputerScience*, pages 310–322. Springer Berlin Heidelberg, 2013. ISBN 978-3-642-37400-5. doi: 10.1007/978-3-642-37401-2_32. URL http://dx.doi.org/10.1007/978-3-642-37401-2_32.

Vivek Narayanan, Ishan Arora, and Arjun Bhatia. Fast and accurate sentiment classification using an enhanced naive bayes model. In *Intelligent Data Engineering and Automated Learning–IDEAL 2013*, pages 194–201. Springer, 2013.

Sasa Petrovic, Miles Osborne, and Victor Lavrenko. Rt to win! predicting message propagation in twitter. In *ICWSM*, 2011.

Everett M Rogers. *Diffusion of innovations*, volume 65. Free Press, 1995. URL http://books.google.com/books?hl=en&lr=&id=v1ii4QsB7jIC&pgis=1.

Daniel M. Romero, Brendan Meeder, and Jon Kleinberg. Differences in the mechanics of information diffusion across topics: idioms, political hash-

tags, and complex contagion on twitter. In *Proceedings of the 20th international conference on World wide web*, WWW '11, pages 695–704, New York, NY, USA, 2011. ACM. ISBN 978-1-4503-0632-4. doi: http://doi.acm.org/10.1145/1963405.1963503. URL http://doi.acm.org/10.1145/1963405.1963503.

Takeshi Sakaki, Makoto Okazaki, and Yutaka Matsuo. *Earthquake shakes Twitter users: real-time event detection by social sensors*, pages 851–860. ACM, 2010. URL http://dl.acm.org/citation.cfm?id=1772690.1772777.

Brian Solis. The rise of digital influence. 2012.

Timm O Sprenger and Isabell M Welpe. tweets and trades : The information content of stock microblogs tweets and trades : The information content of stock microblogs abstract. *Social Science Research Network*, 1702854(December):1291–302, 2010. URL http://papers.ssrn.com/sol3/papers.cfm?abstract_id=1702854.

Bongwon Suh Bongwon Suh, Lichan Hong Lichan Hong, P Pirolli, and E H Chi. Want to be retweeted? large scale analytics on factors impacting retweet in twitter network, 2010. URL http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=5590452.

Gabor Szabo and Bernardo A. Huberman. Predicting the popularity of online content. *Commun. ACM*, 53(8):80–88, August 2010. ISSN 0001-0782. doi: 10.1145/1787234.1787254. URL http://doi.acm.org/10.1145/1787234.1787254.

Sergios Theodoridis and Konstantinos Koutroumbas. *Pattern Recognition, Fourth Edition*. Academic Press, 4th edition, 2008. ISBN 1597492728, 9781597492720.

Michael Trusov, Anand V Bodapati, and Randolph E Bucklin. Determining influential users in internet social networks. *Journal of Marketing Research*, XLVII(August):643–658, 2010. URL http://www.atypon-link.com/AMA/doi/abs/10.1509/jmkr.47.4.643.

Oren Tsur and Ari Rappoport. What's in a hashtag?: Content based prediction of the spread of ideas in microblogging communities. In *Proceedings of the Fifth ACM International Conference on Web Search and Data Mining*, WSDM '12, pages 643–652, New York, NY, USA, 2012a. ACM. ISBN 978-1-4503-0747-5. doi: 10.1145/2124295.2124320. URL http://doi.acm.org/10.1145/2124295.2124320.

Oren Tsur and Ari Rappoport. Whats in a hashtag? content based prediction of the spread of ideas in microblogging communities. *Science*, pages 643–652, 2012b. URL http://eprints.pascal-network.org/archive/00009315/.

Qiang Wang. Svm-based spam filter with active and online learning.

Duncan J Watts and Peter Sheridan Dodds. Influentials, networks, and public opinion formation. *Journal of Consumer Research*, 34(4):441–458, 2007. URL http://www.journals.uchicago.edu/doi/abs/10.1086/518527.

Gary M Weiss. Mining with rarity : A unifying framework. *ACM SIGKDD Explorations Newsletter*, 6(1):7–19, 2004. URL http://portal.acm.org/citation.cfm?id=1007734.

Barbara Wejnert. I ntegrating m odels of d iffusion of i nnovations : A conceptual framework. *Annual Review of Sociology*, 28(1):297–326, 2002. URL http://www.annualreviews.org/doi/abs/10.1146/annurev.soc.28.110601.141051.

Jianshu Weng, Ee-peng Lim, and Jing Jiang. Twitterrank : Finding topic-sensitive influential twitterers. *New York*, Paper 504:261–270, 2010. URL http://portal.acm.org/citation.cfm?id=1718520.

Ian H. Witten and Eibe Frank. *Data Mining: Practical Machine Learning Tools and Techniques, Second Edition (Morgan Kaufmann Series in Data Management Systems)*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2005. ISBN 0120884070.

Qiang Wu, Chris JC Burges, Krysta M Svore, and Jianfeng Gao. Ranking, boosting, and model adaptation. 2008.

Jiang Yang and Scott Counts. Predicting the speed, scale, and range of information diffusion in twitter. In *ICWSM*, 2010.

Sheng Yu and Subhash Kak. A survey of prediction using social media. *CoRR*, abs/1203.1647, 2012.

Tauhid Zaman, Emily B. Fox, and Eric T. Bradlow. A bayesian approach for predicting the popularity of tweets. *CoRR*, abs/1304.6777, 2013.

Tauhid R Zaman, Ralf Herbrich, and David Stern. Predicting information spreading in twitter. *Social Science and*, 55(114171):1–4, 2010. URL http://research.microsoft.com/pubs/141866/NIPS10_Twitter_final.pdf.

Yang Zhang, Zhiheng Xu, and Qing Yang. Predicting popularity of messages in twitter using a feature-weighted model. 2014.