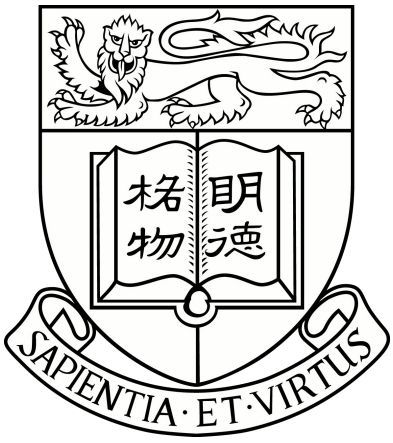

The University of Hong Kong

Faculty of Engineering

Department of Computer Science

COMP7704

Dissertation Title
Social Media Feature Analysis

Submitted in partial fulfillment of the requirements for the admission to the degree of Master of Science in Computer Science

By
Student name: Jiaxin Zhang
HKU Student No.: 3035156341

Supervisor: Dr. Lucas C.K. Hui
Date of submission: 01/08/2015

Abstract

Social media is filled with rumors. They will cause bad damage to the society. Some organizations are responsible for detecting them, preventing people from being fooled. Meanwhile, a part of researchers are devoted to analyze rumor's characteristics and try to build a better automatic rumor detection system. Recently, it becomes a popular research field. This project chooses Sina Weibo, the largest micro-blogging service provider in China, as the research objective. Unlike the previous related work, which are about building classifier with better performance, our work aims to figure out the importance of features towards the credibility of information. We collect the rumor dataset with true and false information and extract a set of features. We train models based on two types of machine learning algorithm and use specific metrics to represent the feature importance. Experiments show that some of the features are quite important while others are irrelevant. Performance of over 80% accuracy on classifying testset supports the conclusion we draw.

Declaration

I declare that this thesis “Social Media Feature Analysis” represents my own work, except where due acknowledgement is made, and that it has not been previously included in a thesis, dissertation or report submitted to this university or to any other institution for a degree, diploma or other qualification.

Acknowledgements

I would like to take this opportunity to show my appreciation to my friends and colleagues who have offered me advice and help.

First and foremost, I would like to express my gratitude to my supervisor, Dr. Lucas C.K. Hui, for his patient guidance and support. He trained me on how to do research oriented work and gave me very useful advice on forming a good research habit. He helped me analyzing my temporal experiment results on each meeting and give his invaluable feedbacks.

I would like to thank Dr. T.W Chim, my second examiner, for listening my presentation carefully and raising important questions. I think his advice also helps me to have a further understanding on this project.

Besides, I want to thank Ms. Miya Liang, who give me lots of advice on designing the experiment scheme. I also want say thanks to my teammate Zilu Fang, who taught me about the data crawling and procecssing. I hope all of them will have a bright future.

Contents

Chapter 1

Introduction

1.1 Motivation

Micro-blogging has become a main approach for people to acquire information outside. In US, people mainly use Twitter as the social tool. But Sina Weibo is most popular social network used in China, with over 500 million registered users, which allows users to broadcast up to 140 characters message with multimedias like images and videos. Those messages can be reposted or commented by the followers of the authors. However, a large amount of messages contain fake information. Once these messages are diffused fast and immoderately, they will cause a huge damage on the stability of the society. These messages are called romors, which need to satisfy two necessary requirement. First, it must be a false message. Second, its repost number should be larger than a specific value. Otherwise, even it is a false message, it can not be concerned as a rumor if nobody receives it.

Due to the property of fast transmission, real-time and no reviewing, Sina Weibo provides the convenience for the creation and transmission of rumor. Whether there is a breaking-news or not, Sina Weibo is filled with various rumors. According to the statistic in this Blue Book[1],

among the 100 hot news, one third of Weibo about them are rumors. March 2012, the rumor stating "Military convoys get into Beijing and something happens" made 16 related websites shutdown. Sina Weibo had to close the comment function[1]. Because of this and other similar rumor incidents, building a rumor automatic detection system on social network is necessary and significant.

1.2 Previous Work

Some researchers have researched on Twitter. Castillo et al.[2] focus on automatic methods for assessing the credibulity of a given set of tweets. They analyze tweets related to "trending" topics and classify them as credible or not credible. Their method is based on supervised learning including SVM, J48 decision tree and Bayes networks. They also do the best-first feature selection and search forward. Qazvinian et al.[3] explore the effectiveness of 3 categories of features including content-based, network-based and microblog-specific memes for correctly identifying rumors. They also show how these features are effective in identifying users who endorse a rumor and further help it to spread. They build different Bayes classifiers as high level features and learning a linear function of these classifiers for retrieval in these two parts.

Recently, there are also some research related to Sina Weibo. Yang et al. collect an extensive set of microblogs from official rumor-busting service provided by Sina Weibo and extract an extensive set of features for training a classifier to automatically detect the rumors from a mixed set of true information and false information. This is the first stydy on rumor analysis and detection on Sina Weibo[4]. Wu et al. stand on a total different viewpoint to study the problem of automatic detection of false rumors on Sina Weibo[5]. Not like the traditional feature-based

[1] http://mil.huanqiu.com/Observation/2012-04/2580647.html

approaches extracting features from the false rumor message such as its author as well as the statistics of its responses to form a flat feature vector, they propose a graph-kernel based hybrid SVM classifier for analyzing the propagation structure of the messages. They also extract semantic feature such as topics and sentiments.

1.3 Contributions

Since we want to build a automatical rumor detection system, we need to crawl rumor dataset and extract features from them. Using these features, a rumor detection model can be trained. However, there are so many features, what feature can best describe the structure inherent in the data? In other words, since we are facing the problem of rumor detection, which feature is most helpful or has the most contribution for us to do accurate classification?

The following are what we have done in this project:

1. We survey the research work related to the rumor detection on Twitter and Sina Weibo, which has been decribed above.
2. We also survey the method to analyze features.
3. We crawl dataset from Sina Weibo.
4. We extract features and train two models. With the help of models, we figure out the importance of features.
5. We evaluate the performance of models we train.

1.4 Report Organization

The rest of this report will be organized using 6 chapters. We first introduce what is feature analysis and its benefit in chapter 2. In the chap-

ter 3, we will discuss the procedure of how we build the dataset. In the chapter 4, the principal theory of algorithm we use will be introduced. In the chapter 5, we will discuss the detailed experiment procedure and we draw conclusions and propose future work in chapter 6.

Chapter 2

Feature Analysis

2.1 Feature

A feature is an individual measurable property of a phenomenon being observed[6]. It is an important part of an observation or a sample for learning about the structure of the problem that is being modeled.

(a) Feature of Weibo (b) Feature of User

Figure 2.1: Features

In this project, some features we are dealing with are circumscribed shown in Fig. 2.1. This is an original rumor-related Weibo along with its author. This message has 869 reposts and 340 comments with 2 likes in the left figure. The author also has some features about the number of

friends, followers and Weibo he or she has. These features are called explicit features, which means we can simply acquire them without further processing. Another kind of feature is called implicit feature that needs to be extracted. We discuss this part in the next section.

2.2 Feature Extraction

Feature extraction is an important step before training a model. It is a process of building derived features inside the data. Feature extraction reduces the useless information while retains the necessary resource to describe a large set of data. By doing this part, we don't need a large amount of memory to store those variables and large computation power to train the model with the risk of overfitting the training samples. The extracted features are expected to contain the related information from the input data.

Castillo et al. extract four types of features including message-based feature, user-based feature, topic-based feature and propagation-based feature[2]. Similarly, Qazvinian et al. use content-based features, network-based features and Twitter Specific memes[3]. For the first one, they present the tweet with lexical patterns and part-of-speech patterns[7]. In the network-based features, they use two types of log-likelihood ratio feature to capture four tpyes of network-based properties. For the Twitter specific memes, they use hashtags and URLs, which has shown the usefulness in [8].

For researching in Sina Weibo, besides the previous feature, Wu et al. also extract the propagation feature towards the transmission path of Weibo[5]. Yang et al. propose two new feature, the location of event and the client program used for posting the microblog.[4] Later in the experiment part, we will discuss the feature we extract and its method.

2.3 Feature Selection

After doing the feature extraction, many features are acquired. However, some features are irrelevant to the problem. There are some features that will be more important than others to the model. Also, some features may be redundant in the context of other features. The focus of feature selection is to select a subset of feature, which can efficiently represent the input data, from the original input feature vector for reducing effects from noise or irrelevant features while still has a good classification or prediction result of the model trained by the chosen features[9]. Thus it is helpful to do feature selection. Feature selection can reduce the number of attributes in the dataset, which is quite similar to the dimensionality reduction. But they are different. For the dimensionality reduction, it reduces the number of features by creating new combinations of attributes such as Principal Component Analysis[10] and Singular Value Decomposition[11], while feature selection only choose the attributes present in the data without changing them.

Feature selection method can be divided into three types including filer methods, wrapper methods and embedded methods.

2.3.1 Filter methods

Filter methods are applied before training a model. These methods need quantitative criteria to measure the relevance of each feature with the output class. Then these features can be ranked by the value of criteria and be selected by ordering. It is helpful in the practical application due to the simplicity computation. Usually, a threshold is needed to decide whether a feature can be retained or not.

Next we will discuss how to measure the relevance of a feature to the output.

Pearson product-moment correlation coefficient

One of the simplest criteria is the Pearson correlation coefficient[12].

$$R(i) = \frac{cov(x_i, Y)}{\sqrt{var(x_i) * var(Y)}} \tag{2.1}$$

where x_i vector is the i_{th} sample. Y is the output class. $cov()$ is the covariance of x_i and Y and $var()$ is the variance of the variable. This method can detect the linear correlation between variable x_i and the target. The value of this coefficient is in the range of -1 to +1, where +1 is total positive correlation, 0 is no correlation and -1 represents the total negative correlation. However, only the numerical or continuous type data can use it.

Mutual Information

Shannon's information theory[13] tolds us the method to quantify the *entropy* by the following equation:

$$H(C) = -\sum_{c=1}^{N_c} P(c) \log(P(c)) \tag{2.2}$$

where $P(c)$ is the probability for the different classes. $c = 1, ...N_c$. This equation represents the uncertainty in the output class. If we know the feature vector $\boldsymbol{f}$, the *conditional entropy* is:

$$H(C|F) = -\sum_{f=1}^{N_f} P(\boldsymbol{f})(\sum_{c=1}^{N_c} P(c|\boldsymbol{f}) \log P(c|\boldsymbol{f})) \tag{2.3}$$

where $P(c|\boldsymbol{f})$ is the conditional probability for class c given the input feature vector $\boldsymbol{f}$

If the input feature vector is of continuous variable, we need to do an integral. The probabilities will be replaced by the probability densities,

like the following equation:

$$H(F) = -\int P(f)\log P(f)df \tag{2.4}$$

In general, the conditional entropy will be less than the initial entropy, except the situation that if and only if the feature is completely independent to the output class. For example, if the joint probability density is equal to the product of the probability density of this feature and the class. Thus comes to the definition of *mutual information*:

$$I(C;F) = H(C) - H(C|F) \tag{2.5}$$

It is a symmetric to the C and F. It can also be expressed by the following equation:

$$I(C;F) = \sum_{c,\boldsymbol{f}} P(c,\boldsymbol{f})\log\frac{P(c,\boldsymbol{f})}{P(c)P(\boldsymbol{f})} \tag{2.6}$$

or by the continuous feature version of mutual information:

$$I(C;F) = \sum_{c}\int P(c,\boldsymbol{f})\log\frac{P(c,\boldsymbol{f})}{P(c)P(\boldsymbol{f})}d\boldsymbol{f} \tag{2.7}$$

We use $\boldsymbol{f}$ rather than one specific feature as the input. Actually, if we want to find out whether one feature has dependence with the output, we can simply calculate the mutual information by the following equation:

$$I(X;Y) = \sum_{y\in Y}\sum_{x\in X} p(x,y)\log\frac{p(x,y)}{p(x)p(y)} \tag{2.8}$$

These equation above show that if X and Y are independent then the value of mutual information would be zero. The dependent feature can decrease the uncertainty by providing information to the class[14].

The filter methods don't rely on learning algorithm so that they are not very computation comsuming. However, these methods may not

perform well when there are redundant features in the input feature vector because filer methods only calculate the relevance between output class and feature one by one. Under this circumstance, the "optimal" subset selected by filter methods may contain many highly-correlated features. However, even if one feature has less information to the output class by its own, but may be informative when combined with other features[15].

2.3.2 Wrapper methods

Wrapper methods is done using the induction algorithm as a black box and use the predictor or classifier performance, usually the accuracy of model, as the objective function to evaluate the feature subset[16]. Suppose we have N features, we will need to evaluate 2^N subset of features, which is a NP-hard problem. If we just evaluate the whole space exhaustively, it will be very computation consuming. To solve this problem, several methods are proposed.

Sequential selection algorithms

These methods start with an empty set and iteratively add one feature into the feature subset. Each time it selects the feature from the remaining features that can give the highest value for the objective function. This process is terminated once the required number of feature is added. On the opposite direction, we can also start with a full set and remove one feature each step which that gives the lowest decrease on the value of objective function. These two method, called Sequential Feature Selection (SFS) and Sequential Backward Selection (SBS), are naive search algorithm because it doesn't take the dependency between features into account.

Pudil et al. do a little improvement on the naive version of SFS, call

Sequential Float Feature Selection (SFFS). They introduce an additional step called backtracking step. After a feature is added into the subset, they try to reach a better performance of the objective function by removing one feature from it[17]. If the performance is getting better, they use the reduced subset to do this trial again until no more performance increase. Then do the feature addition as usual.

On the basis of SFFS, some researchers propose further improvement. Nakariyakul and Casasent add an additional search step called "replacing the weak feature" to check whether removing any feature in the current selected subset and adding a new one at each sequential step can improve the performance of current feature subset[18]. They provide the optimal or quasi-optimal solution for many selected subsets. This solution requires significantly less computation than the optimal feature selection algorithm.

Heuristic search algorithms

When N is relatively large, heuristic search methods such as genetic algorithm[19] can be used to reduce the computation load. It uses the chromosome bits to represent whether a feature is included in the subset or not. If a bit is 1, it means that the corresponding feature is selected while 0 means the corresponding feature is not selected. A typical run of genetic algorithm involves many generations, or we can say the iteration. In each iteration, evaluation of an individual chromosome, which represents a specific subset of feature, involves training a model and compute the accuracy. In genetic algorithm, there are several parameter that can be tuned as followed:

1. Population size

2. Number of generations

3. Probability of crossover

4. Probability of mutation

5. Probability of selection of the highest ranked individual

These concept can be found in this link[1]. Yang and Honavar implement this heuristic search algorithm to do feature selection[20]. To evaluate the individual chromosome, in genetic algorithm called fitness, they use neural network trained by DistAl[21] and evaluate the performance of it by accuracy. Besides, they also calculate the cost of performing the classification. Thus the fitness of chromosome can be represented as the following equation:

$$fitness(x) = accuracy(x) - \frac{cost(x)}{accuracy(x) + 1} + cost_{max} \tag{2.9}$$

where x is a subset of features and $accuracy(x)$ is the test accuracy of the neural-network classifier using the subset feature x. $cost(x)$ is the sum of measurement costs of the feature subset x. By choosing the largest value of $fitness(x)$, its corresponding subset of feature are considered as the optimal selection result.

The obvious drawback of wrapper methods is that for each subset of feature, a model is needed to be trained. So the execution time of algorithms will be very long when the number of sample and feature is large. Also, for genetic algorithm, same subset of feature may be repeatedly evaluated. Overfitting is an another drawback if the model learns the data too well.

2.3.3 Embedded methods

Since we have so many drawbacks to the filter and wrapper methods we discussed above, the embedded methods try to reduce the bad

[1] https://en.wikipedia.org/wiki/Genetic_algorithm

influence to the feature selection. The principle of these methods is that when doing the training process, the task of feature selection is done simultaneously.

The most popular methods involves:

1. l_1-norm or l_2-norm regularization linear model like LASSO or l_1-SVM
2. Decision tree model
3. SVM with Recursive Feature Elimination

Since all of them are used in this project, the concept of them will be discussed later.

2.4 Feature Importance

After feature extraction and feature selection, we roughly acquire the scores allocating for the features. They can also be ranked by their scores. However, the score may be calculated by different method or represented by different coefficient, thus the comparison of value of the score between different method is meaningless. Features importance can provide us a intuitively feeling towards the problem. Meanwhile, by using the features with high score, we can construct new features. In this project, we want to find the most important features that can help us to train a better model for classifying whether a Weibo is a rumor or not.

2.5 Benefit of Feature Analysis

What feature analysis can achieve is that we can use less but more important features to train models rather than input a high dimensional

feature vector. If the data is in a high dimension, the curse of dimensionality will occurs. Data becomes increasingly sparse in the space. The density and distance between each sample increase so that data becomes dissimilar which prevents algorithm from finding the inside structure or relation. For example, some algorithms based on calculating the distance to find similar samples, like k-nearest neighbors algorithm[22] or k-means clustering[23], become less meaningful. Also, anomaly detection may face some problems because all of the distances become numerically similar[24].

Actually, when training models, the computation time and required memory will exponentially increase as the dimention of data increases. Meanwhile, even if the models can be trained, the complexity of model will be too complex, which will cause the *deterministic noise*[25]. The overfitting will occurs because the models try to fit the noise either the stochastic noise or the deterministic noise.

Chapter 3

Data Preparation

3.1 Dataset source

The dataset are acquired from the link provided in this paper[5]. The original dataset are composed of four contents.

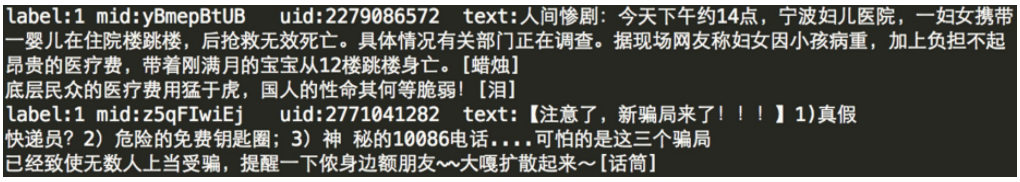

Figure 3.1: Sample in Original Dataset

1. mid: The id of individual Weibo
2. uid: The id of individual user
3. Text
4. label: the class of that Weibo, 1 for rumor and 0 for not rumor.

This dataset only contains these four items. Thus we need to crawl more features from Sina Weibo. There are various API provided by Sina Weibo. Since we have the Id of Weibo, we can use it in Java as followed:

```
import weibo4j.Timeline;
import weibo4j.model.Status;
Timeline tm = new Timeline();
Status status = tm.showStatus(id);
```

We need the id of Weibo as the input. The data are returned as JSON. The class *Status* has many method to resolve JSON files. Thus it is convenient to acquire various items.


```
{
    "id":3496390741401730,
    "created_at":"Mon Oct 01 18:39:35 +0800 2012",
    "original_pic":"http://ww2.sinaimg.cn/large/68c00843jw1dxfw1h5qbhj.jpg",
    "comments_count":11,
    "source":"未通过审核应用",
    "attitudes_count":0,
    "reposts_count":119,
    "pic_urls":[{"thumbnail_pic":"http://ww2.sinaimg.cn/thumbnail/68c00843jw1dxfw1h5qbhj.jpg"}],
    "in_reply_to_status_id":"",
    "text":"完娛17週年 S.H.E 生日祝福 20120928完全娛樂 【在米蘭線的XD】水管: www.you tube.com/watch?v=hjnNpEytivc http://t.cn/zlfEpyZ"
    ...
    "user":{
        "location":"台湾 台北市",
        "created_at":"Sun Jun 13 08:07:33 +0800 2010",
        "verified_type":-1,
        "profile_image_url":"http://tp4.sinaimg.cn/1757415491/50/40025223816/0",
        "province":"71",
        "followers_count":9306,
        "geo_enabled":"False",
        "verified":"False",
        "favourites_count":9,
        "bi_followers_count":32,
        "gender":"f",
        "screen_name":"一首加油的歌給selina",
        "friends_count":152,
        "statuses_count":15748,
        "description":"我是黑不是粉",
        "name":"一首加油的歌給selina",
        "id":1757415491,
        "credit_score":80,
        ...
    },
}
```


Figure 3.2: One sample crawled by Weibo API

3.2 Feature Extraction

In Figure. 3.2 we can see the data are separated into two parts. From line 1 to line 12, those items are about the Weibo. For example, we can see the time of posting at line 3, the number of reposts at line 8, along with the text at line 11, so on and so forth. The rest of the content are about the author of this Weibo including the the number of posts at line 27 and friends count at line 28 and so on. Just for clarifying, in the user part, the *created_at* item represents the registered time of the user.

Then we introduce what feature we extract in this project. The fea-

ture we extract are mainly introduced by previous work. Experiments from related paper have shown that using those features can reach a good performance of model. We consider that those features are selected by domain knowledge[26]. However, the methods for extracting those features are not provided in their theses. Thus we try to use similar methods to extract them and we will discuss the methods in the next section.

First we discuss the implicity features. After that, we integrate them with explicit features in a table.

3.2.1 Implicity Feature

Implicity Feature are mainly extracted from the Weibo text. We have extracted four types of features from the text.

Emotion mark

Chinese text is different from English because there is no space between each character. However, to analyze the emotion of the text, sentimental words is necessary. Thus we first need to split the sentence into several words. To do so, we use a third-party library called *Jieba*[1] written in Python. Three types of cutting word mode are provided. We find the "accurate mode" are suitable for us. Besides, in order to remove some special charactors like "[]" or "#", we use the regular expression to filter out them.

```
人间惨剧：今天下午约14点，宁波妇儿医院，一妇女携带一婴儿在住院楼跳楼，后抢救无效死亡。具体情
况有关部门正在调查。据现场网友称妇女因小孩病重，加上负担不起昂贵的医疗费，带着刚满月的宝宝从1
2楼跳楼身亡。 底层民众的医疗费用猛于虎，国人的性命其何等脆弱!

人间 惨剧 今天下午 约 14 点 宁波 妇儿 医院 一 妇女 携带 一 婴儿 在 住院楼 跳楼 后
抢救无效 死亡 具体情况 有关 部门 正在 调查 据 现场 网友 称 妇女 因 小孩 病重 加上
负担 不起 昂贵 的 医疗费 带 着 刚 满月 的 宝宝 从 12 楼 跳楼 身亡 底层 民众 的 医疗
费用 猛于 虎 国人 的 性命 其 何等 脆弱
```

Figure 3.3: One Example of Word Split

[1] https://github.com/fxsjy/jieba

After we split the sentence into words, we need a dictionary containing emotional words to match. We use the dictionary provided by National Taiwan University called *NTUSD*[2]. This dictionary collects 2800 positive words and 8200 negative words. When we take the splitted sentence to match this dictionary, if the text contain a positive word, we give this text a +1 mark while -1 mark with negative word.

Meanwhile, user also may use emotion icon of emoji to express their feelings. For example, Weibo official emotion icon are represents in text as [good], [Weiwu] or [Nu]. User can also use apple emoji, such as 😂 or 😄. We choose a part of them and manually classify the emotional direction. Identically, if the text has a icon or emoji which is classifying as a positive one, we give this text +1 mark. Otherwise -1 mark. We sum up all the marks this text acquires.

Word Type

Except for the sentimental analysis, we also propose a new set of features by calculating the percentage of the number of different type of word in the whole text, which are consisted of *img*, *real*, *eng* and *other*. Among, *img* is the short for imaginary or function word and *eng* is short for English word. The following table shows the rule of classifying which type each word is.

Real	adjective	verb	noun	numeral	pronoun
Img	exclamation	preposition	conjunction	auxiliary word	adverb

Table 3.1: Word Type

[2] `https://github.com/lackneets/NewsEvents/tree/master/lib/ntusd`

Length

The length of the text. Each word needs 3 bytes to store using utf-8 to encode.

hasURL

We use regular expression to match whether a text contain a url or not. This feature is an binary feature.

```
http[s]?://(?:[\w\d]|[$-_@.&+]|(?:%[\w\d]))+
```

Along with the explicit feature, we extract 22 features as the vector to build our dataset in Table. 3.2

Category	Feature	Description
Text	Emotion	The emotional mark of the text
	HasURL	Whether the message has URLs
	Length	The length of the message
	RealWord	The percentage of real word in the text
	ImgWord	The percentage of imaginary word in the text
	EngWord	The percentage of English word in the text
	OtherWord	The percentage of other word in the text
User	RegPostTime	Time span between registration and posting
	Verified	Whether the user is verified by Sina Weibo
	VerifiedType	Type of user based on verified information
	VerifiedKind	Similar to VerifiedType but more general
	Gender	Female or male
	GeoEnabled	Whether user enables the locating function
	Province	Where the user was registered
	FollowersCount	The number of people following this user
	FriendsCount	The number of people this user follows
	BiFollowersCount	The number of people mutually following
	StatusesCount	The number of status the user has posted
	FavouritesCount	The number of Weibo the user farourites
Repost	CommentCount	The number of comments the Weibo receives
	ShareCount	The number of repost from this Weibo
	AttributesCount	The number of like this Weibo receives

Table 3.2: Feature Vector and Description

3.3 Data Cleansing

Data cleansing is a process of detecting illegal or inappropriate or missing data in the dataset and correcting them. It is an important part because incorrect data may lead to false conclusions. Sometimes, to let the algorithm can run smoothly without exception, missing value needs to be filled.

The methods of data cleansing is dependent on the specific application. Different types of error needs different methods. In our project, we are mainly facing the following types of errors.

Negative value of *time span*

The value of this feature *time span* is the span from the user registered time to the post time of that Weibo. The number of samples which have this type of error is small so that we can simply remove these samples.

Number of *BiFollowersCount* > *FriendsCount*

Mutual followers means two user follow each other. The number should be less or equal to the *FriendsCount*. Thus we use the value of *FriendsCount* as a substitute for the *BiFollowersCount* dealing with this error.

Value of *StatusesCount* is 0

Since we can crawl the Weibo of this user, thus its *StatusesCount* can never be 0. Thus we simply set this feature as 1.

So far, the dataset has been built with roughly 4800 samples with 2400 rumor labeled as *True* and 2400 not rumor labeled as *False*. Fig. 3.4 is a part of our prepared dataset. In the next chapter, we want to introduce

the theory of algorithms that can help us do analyzing the importance of the features.

createtog	followers	bi_follow	friends_c	statuses_	favourite	share_ct	comment	attitudes	verified	verified_t	verified_	gender	geo_enal	province	hasURL	emotion	length	realwd	imgwd	engwd	otherwd	is_rumor
314	668682	2518	2778	20665	4	266	129	0	1	celebrity	yellowV	1	1	11	1	1	106	39.13	13.04	17.39	30.43	FALSE
750	2377080	33	57	46036	59	695	40	100	0	normal u	normal u	1	0	71	0	-8	396	62.5	19.32	0	18.18	FALSE
840	21307	2484	3000	23431	14	119	37	0	1	celebrity	yellowV	1	1	11	0	2	321	72.06	11.76	0	16.18	FALSE
791	3613	7	9	26	0	1387	475	44	0	normal u	normal u	1	0	100	0	1	309	63.01	17.81	5.48	13.7	FALSE
560	1035876	74	214	7474	162	1234	39	155	0	senior m	master	0	1	31	0	-5	171	57.14	28.57	0	14.29	FALSE
487	404906	20	47	111870	6	1154	195	0	0	normal u	normal u	0	1	44	1	-7	231	53.7	16.67	7.41	22.22	FALSE
632	6157666	399	621	86426	5	461	70	0	1	media	blueV	1	1	11	1	-1	221	50	23.08	7.69	19.23	FALSE
1555	1642154	1312	2008	66390	443	1148	55	143	1	media	blueV	1	1	11	0	-2	336	55.95	22.62	0	21.43	FALSE
1258	1652	483	930	17552	254	825	34	3	0	senior m	master	1	1	31	0	0	69	47.06	17.65	5.88	29.41	FALSE
1024	7311696	207	249	36161	13	236	77	2	1	media	blueV	1	0	81	0	1	324	68.66	11.94	0	19.4	FALSE
268	718	64	209	3090	244	1540	357	0	0	normal u	normal u	1	1	44	0	-2	417	69.57	9.78	2.17	18.48	FALSE
309	1955001	59	179	39926	1	742	58	2	0	normal u	normal u	0	1	11	0	-3	389	59.78	22.83	0	17.39	FALSE
316	7782	162	381	2512	11	183	78	1	1	enterpris	blueV	1	1	31	0	-1	188	63.16	10.53	7.89	18.42	FALSE
1285	412270	1934	2210	2173	5401	1342	260	73	1	celebrity	yellowV	0	1	400	0	-3	416	63.53	12.94	2.35	21.18	FALSE
1209	6199626	201	658	79196	477	1800	241	129	0	normal u	normal u	1	0	11	1	0	358	56.79	14.81	4.94	23.46	FALSE
1181	9146023	293	918	92503	7029	1603	370	271	0	normal u	normal u	0	0	44	0	-1	69	47.06	11.76	0	41.18	FALSE
902	106987	1772	1993	8499	80	197	121	0	0	normal u	normal u	1	1	43	0	1	270	70.18	8.77	0	21.05	FALSE
778	10608	2871	2978	211937	18	452	238	30	0	normal u	normal u	1	1	21	1	5	285	41.11	11.11	4.44	43.33	FALSE
437	1009002	1967	2891	42759	634	908	72	2	0	normal u	normal u	1	0	31	0	0	315	49.35	15.58	2.6	32.47	FALSE
611	203187	22	30	6849	31	907	418	7	0	normal u	normal u	0	1	44	0	-6	348	53.57	16.67	9.52	20.24	FALSE
104	166262	1330	1463	16673	281	590	75	0	0	normal u	normal u	0	1	31	1	1	342	67.16	13.43	5.97	13.43	FALSE
1411	8195313	260	293	100396	26	1391	814	110	1	media	blueV	0	0	11	1	-8	404	63.86	18.07	4.82	13.25	FALSE
291	16545	294	1232	9956	57	342	201	0	1	celebrity	yellowV	1	0	61	0	0	412	61.36	25	0	13.64	FALSE
887	93995	596	1018	3298	587	1779	285	4	1	celebrity	yellowV	0	1	400	0	4	403	62.64	18.68	0	18.68	FALSE
343	4022153	35	68	46209	123	1136	459	16	0	normal u	normal u	0	0	44	0	0	66	76.47	11.76	0	11.76	FALSE
824	1715485	220	474	40667	1	1060	245	81	1	enterpris	blueV	0	0	11	0	1	403	65.48	13.1	1.19	20.24	FALSE

Figure 3.4: Extracted Dataset

Chapter 4

Machine Learning Algorithm

To figure out the importance of each feature in the feature vector, we use model-based methods to evaluate them. The statistics method such as pearson coefficient or chi-square can find the relationship between features and class, but since they are univariate feature methods, not considering the dependence between features, and the ultimate target is to train a model with better performance, it is reasonable to use models to directly tell us the feature importance. In this project, we use models with two representations, which is linear model and decision tree model.

4.1 Logistic Regression

It is a simple algorithm for training linear binary classifiers. Although its name contains "regression", it is a classification algorithm. It can be used when the input data is numeric type and nominal type. Numeric data means its value is continuous and its number is meaningful. Nominal data is consisted of several categories. If they are represented in number, the value is meaningless.

Linear model can be represented as the following equation:

$$z = \theta^T x = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + ... + \theta_n x_n \tag{4.1}$$

where x is the input vector. n equals to the number of feature. θ is the coefficient of features. Then take this value into the *sigmoid* function as:

$$h(z) = h_\theta(x) = \frac{1}{1 + e^{-\theta^T x}} \tag{4.2}$$

The curve of sigmoid function is shown in Figure. 4.1. In this figure, x-axis stands for the z we calculated in Eq. 4.1 and y-axis is $h(z)$. It is simply for doing a projection from z to $h(z)$. Since logistic regression algorithm is used for training a binary classifier, we need a threshold to determine which input can be classified as 1 or 0. From this figure, we can see that if the value of z is larger than 0, then the value of $h(z)$ will be larger than 0.5. Thus, 0.5 is a suitable threshold. If $h(z) > 0.5$ then the result is classified as 1, otherwise 0.

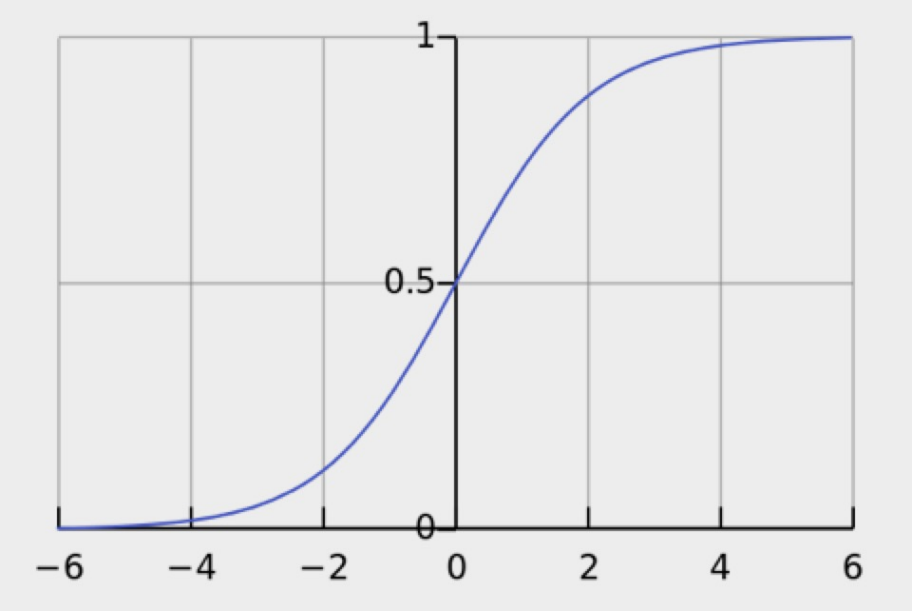

Figure 4.1: Curve of Sigmoid Function

How to use this linear model to interpret the importance of features? The basic idea is using coefficients of the linear model θ in Eq. 4.1. The most important features should have the highest coefficients in the model. If the feature is uncorrelated with the class, its coefficient value should close to zero. So how to train the coefficient θ? In logistic

regression, the idea is gradient descend.

First we need to define cost function J of logistic regression in 4.3

$$J(\theta) = \frac{1}{m}\sum_{i=1}^{m} Cost(h_\theta(x^{(i)}), y^{(i)}) \tag{4.3}$$

where m is the number of sample. $Cost(h_\theta(x^{(i)}), y^{(i)})$ is the error in each sample. It can be represented as Eq. 4.4

$$Cost(h_\theta(x^{(i)}), y^{(i)}) = \begin{cases} -\log\left(h_\theta(x^{(i)})\right) & if\ y^{(i)} = 1 \\ -\log\left(1 - h_\theta(x^{(i)})\right) & if\ y^{(i)} = 0 \end{cases} \tag{4.4}$$

In Eq. 4.5, if a sample is labeled as $y^{(i)} = 1$, we want the cost of the model on this sample is 0, then $-\log\left(h_\theta(x^{(i)})\right)$ should be 0. Also, we know the classification result is $y^{(i)}_{classify} = 1$ when the value of $h_\theta(x^{(i)}) > 0.5$. Thus the value of $-\log\left(h_\theta(x^{(i)})\right)$ is close to 0, which is what we want. The situation when $y^{(i)} = 0$ is similar. So we can do a furtuer simplification into following equation.

$$Cost(h_\theta(x^{(i)}), y^{(i)}) = -y^{(i)}\log\left(h_\theta(x^{(i)})\right) - (1 - y^{(i)})\log(1 - h_\theta(x^{(i)})) \tag{4.5}$$

It is equaled to Eq. 4.4 Thus the cost function of logistic regression is:

$$J(\theta) = \frac{1}{m}[-y^{(i)}\log\left(h_\theta(x^{(i)})\right) - (1 - y^{(i)})\log(1 - h_\theta(x^{(i)}))] \tag{4.6}$$

So far, we can use gradient descend to minimize $J(\theta)$. θ is an (D+1)-dimension vector if the input feature vector is D-dimension. The extra 1 dimension is the bias term for x_0 in Eq. 4.1. To calculate the gredient, we need to calculate partial derivative with respect to each element in θ and minus it in Eq.

$$\theta_j := \theta_j - \alpha\frac{\partial}{\partial\theta_j}J(\theta) \tag{4.7}$$

where j is the j-th coefficient and ":=" means update the value of θ_j using the formula on the right side. What is worth mentioning is that all the coefficient of feature in the vector should be updated simultaneously. α is the learning rate meaning how much distance θ wants to "walk" downward at each iteration. The value is preset by us. If it is too small, the learning procedure may take a long time. If it is too large, the value of $J(\theta)$ may even not convergent.

Interpreting the importance of feature by the value of coefficient is a feasible method. However, to make the result more accurate, we need to do further process.

4.1.1 Reguralized Logistic Regression

Overfitting[1] is a common concept in statistics and machine learning. It means that if our model learns too much from the training dataset, the overfitting may occur. To be more specific, if the model fit the data too well, the model may become more complex, which describes more noise instead of underlying relationship of data. In other word, if the model have too many feature, it will fails to generalize on new samples. To address overfitting, one of a popular method is *regularization*.

If we don't want too many feature to make contribution on the model, we can simply set the coefficient of corresponding feature into 0. However, it is an NP-hard problem to solve because we have 2^N permutations and need to do searching for the best setting. To simplify this problem, we use Eq. 4.8 as an constriant to substitute the NP-hard problem.

$$\sum_{j=1}^{D} \theta_j^2 \leq C \tag{4.8}$$

How to solve this problem of minimizing $J(\theta)$ with a constraint condi-

[1] https://en.wikipedia.org/wiki/Overfitting

tion? The answer is using the *Lagrange Multiplier*[27]. We want to find a lagrange multiplier $\lambda > 0$ and put the constraint condition into the minimization equation of $J(\theta)$ with the help of λ. Thus the Eq. 4.6 would be changed into the following equation:

$$J(\theta) = -\frac{1}{m}\sum_{i=1}^{m}[y^{(i)}\log\left(h_\theta(x^{(i)})\right) + (1-y^{(i)})\log(1-h_\theta(x^{(i)})) + \frac{\lambda}{2m}\sum_{j=1}^{D}\theta_j^2] \tag{4.9}$$

where $\sum_{j=1}^{D}\theta_j^2$ is named as *regularizer*. Thus the Eq. 4.7 becomes:

$$\theta_j := \theta_j - \alpha\frac{\partial}{\partial\theta_j}J(\theta) + \frac{\lambda}{m}\theta_j \tag{4.10}$$

where j starts from 1 to D, not including the bias term. This regularization is call *L2-norm* regularization. Another kind of regularization is called *L1-norm* regularization. The difference between L1 and L2 is that in L1-norm regularization, the regularizer is $\sum_{j=1}^{D}|\theta_j|$. Using L1-norm regularization, the coefficient of weak feature will be 0. Thus, the feature vector may be sparse[28].

4.1.2 Multicollinearity of Features

When there are multiple correlated features, the model trained by logistic regression will be unstable[29]. It means that the coefficient would have a large change even if the value of data has a tiny change. For example, we have a dataset generated from a model $y = x_1 + x_2$ and x_1 and x_2 are highly correlated such as $x_1 \approx x_2$. We manually add some noise ϵ. Then we take this noise-additive dataset into an algorithm. Ideally the trained model would be $y = x_1 + x_2$. But it may be $y = 2x_1$ or $y = -x_1 + 3x_2$ due to the noise. Thus, we need to do a filter-based feature selection to remove the correlation between features before training the model.

Correlation-based Feature Selection

Hall and Mark proposed a method called Correlation-based feature selection (CFS)[30] in 1999. It is a special filter-based feature selection because it ranks a subset of feature rather than individual feature, which means that this method consider the dependence between features.

CFS algorithm is a heuristic for evaluating *merit* of a subset of features. This heuristic is based on:

Good feature subsets contain features highly correlated with the class, yet uncorrelated with each other[31].

The heuristic can be formulized as:

$$Merit_s = \frac{k\overline{r_{cf}}}{\sqrt{k + k(k-1)\overline{r_{ff}}}} \tag{4.11}$$

where $Merit_s$ is the heuristic "merit" of one subset of feature. k is the number of feature in this subset. $\overline{r_{cf}}$ is the average feature-class correlation and $\overline{r_{ff}}$ is the average feature-feature intercorrelation. These two correlation can be calculated as Pearson's correlation coefficient or other coefficient. In Eq. 4.11, the numerator indicates how predictive is this subset of feature while the denominator tells how much redundancy between them. By selecting the subset with large value of *Merit*, this heuristic method can remove the subset composed of highly-correlated features and poor predictors of the class.

4.1.3 Feature Scaling

It is a simple but necessary problem needed to be solved when we use coefficient of feature to indicate the importance. If not, the model will be governed by the feature with large range of value.

Normalization

The general formula is:

$$x' = \frac{x - x_{min}}{x_{max} - x_{min}} \tag{4.12}$$

where x_{min} and x_{max} are the minimum and maximum value in the feature. After doing this processing, the range would be scaled into [0, 1].

Standardization

Feature standardization makes the value of each feature have zero-mean and unit-variance. It is widely used in many machine learning algorithm such as logistic regression and support vector machine. The general formula is:

$$x' = \frac{x - \bar{x}}{\sigma} \tag{4.13}$$

where $\bar{x}$ is the mean of the value of x and σ is its standard deviation.

After the three procedure above, the trained model can be used to indicate the importance of features. However, due to the CFS method, we can only input the feature without correlation. If we want to figure out all the importance of feature in the vector, we need to use other technique, such as the random forest that we will discuss in the next section.

4.2 Random Forest

Random Forest is a ensemble algorithm for classification. The model trained by this algorithm are represented as many decision trees, which is one of the popular algorithm[32]. When we use the test set to evaluate this model, the output are determined by voting. For example, there are 5 trained decision tree in this forest. if the output y of one sample are classified as "A" by three trees and "B" by two trees, then the output

is "A". If the comparison are even, just randomly choose one to be the output class.

To explain this method more convenient, we need to introduce the decision tree algorithm. A decision tree has internal node, branch and leaf. Each internal node is labeled with an input feature. Each leaf represents a value or class. The branch from a internal node are labeled with a specific requirement towards the feature on this node. If a node extends two branches, then the dataset on this node are separated into two parts according to the requirement on the branch.

How to choose the feature on each node and what requirement are needed to split the node? In decision tree, we use *impurity decrease* as the metric. First we want to discuss the coefficient that can represent the *impurity*.

4.2.1 Impurity

Gini Index

Gini index is a measure of statistical dispersion[33]. If the value is zero, means that all the values are the same. It can be calculated by:

$$GINI(S) = 1 - \sum_i {p_i}^2 \tag{4.14}$$

where p_i is the fraction or probability of item labeled with i in the dataset.

Entropy

Entropy is a numeric measure to quantify the concept of uncertainty. If the value of entropy is large, the uncertainty is higher. It has been discussed in chapter 2, with Eq. 2.2.

4.2.2 Information Gain or Impurity Decrease

We have discuss the concept of mutual information in chapter 2. Actually, the information gain is the same as the mutual information from the equation. It expresses the impurity decrease if we know a condition, which is the requirement labeled in the branch. The equation of information gain in the decision tree is:

$$InfoGain = H(S) - \frac{m}{m+n}H(S_1) - \frac{n}{m+n}H(S_2) \tag{4.15}$$

where $H(S)$ is the impurity of the current node. $H(s_1)$ and $H(s_2)$ is the impurity of child node 1 and 2 if it is binary split. m and n is the number of sample in child node 1 and 2. With the help of information gain, we can decide which feature to split and the requirement on the branch by selecting the largest value of information gain.

The discussion above are about one decision tree. To correct its overfitting on training data, random forest was developed by Leo Breiman[34]. Instead of using all the input features to train the decision tree, a random subset of features is available for training one tree. Also, the feature which is selected to split the node is a randomized procedure, rather than a deterministic and simply choosing the largest information gain, just like in the decision tree algorithm we discuss above.

4.2.3 Feature Importance Evaluation

Random forest can provide two straightforward methods for evaluating the importance of feature.

Mean decrease impurity

We have already known if the impurity decrease larger, the more information this feature can provide about the output class. That means

the feature is more important. There are many decision tree in a forest. Each feature has a impurity decrease value on each tree. Sum up these values and take the average, we can get a rank of the features. Due to the randomization, even if the input features have dependence on each other, the model will not be unstable, which can solve this problem in the model trained by logistic regression. For example, there are two feature that are highly correlated. If a node choose one of them to split, the impurity is significantly reduced and this one is considered as an important feature while the other's importance is significantly reduced. In the random forest model, both of these two feature can be selected to split the node in different tree. Thus the bias phenomenon is somewhat reduced. Theoretically, if the number of tree is larger, the bias will be decrease.

However, this method has its own drawback. It is biased towards preferring features with more categories[35] and the next method can somewhat reduce this kind of bias.

Mean decrease accuracy

This method directly measures the impact of each feature on accuracy of the model. The idea is that we use a training set to build a model. Then we use the testset evaluate this model. We record the accuracy of the model. Then we permute the values of each feature in the testset and calculate the new accuracy of the model tested by the permuted testset. We can manually add some noise on the test set to realize the permutation. Then we compare how much the permutation can decrease the accuracy. If the feature is important, the permutation on this feature will cause a large decrease on the accuracy when using test set to evaluate the model. We can use this accuracy decrease to rank the importance of features.

Chapter 5

Experimental Results

In chapter 3, we have prepared the dataset. First we use it to train a linear model using logistic regression.

5.1 Feature Selection in CFS

In the feature vector, there are correlated features. Thus, we need to remove such dependency first. The feature we use in our project are listed in the following table. After selection, only 7 features are left. In

Emotion	HasURL	Length	RealWord
ImgWord	EngWord	OtherWord	RegPostTime
Verified	VerifiedType	VerifiedKind	Gender
GeoEnabled	Province	FollowersCount	FriendsCount
BiFollowersCount	StatusesCount	FavouritesCount	CommentCount
ShareCount	AttributesCount		

Table 5.1: Feature Vector

Emotion	ImgWord	EngWord	VerifiedKind
FollowersCount	BiFollowersCount	ShareCount	

Table 5.2: Feature Vector Left

the left feature, all of them are continuous feature except the "VerifiedKind", which is a nominal feature. To make the logistic regression al-

gorithm can run, we need to transform this nominal feature into several binary features.

In "VerifiedKind" feature, there are four category, which is "normal user", "master", "yellowV" and "blueV". Thus, we create four new features and drop the original "VerifiedKind" feature. So the feature vector becomes

Emotion	ImgWord	EngWord	master
yellowV	blueV	normal user	
FollowersCount	BiFollowersCount	ShareCount	

Table 5.3: Feature Vector for Training

5.2 Training by Logistic Regression

We taking this ten features into logistic regression and do standardization. We choose l2-norm regularization and set $C = 1.5$. After training, the model is given as the following table with intercept or $\theta_0 = -0.36647169$.

Feature	**Coefficient**	**Feature**	**Coefficient**
Emotion	-0.38976887	ImgWord	-0.39248365
EngWord	-0.45859069	master	0.21188558
yellowV	-0.03449316	blueV	-0.30445247
normal user	0.16429209	FollowersCount	-2.44114483
BiFollowersCount	0.36734794	ShareCount	1.92550177

Table 5.4: Coefficient of each feature

When we train this model, we use 70% of data as the training set and use 30% left to evaluate the performance in Table. 5.5. The overall accuracy is 0.781.

	precision	recall	f1-score	support
False	0.81	0.75	0.78	744
True	0.76	0.81	0.78	704
avg/total	0.78	0.78	0.78	1445

Table 5.5: Model Performance trained by Logistic Regression

We can take the absolute value of them and draw a histgram in Fig. 5.1

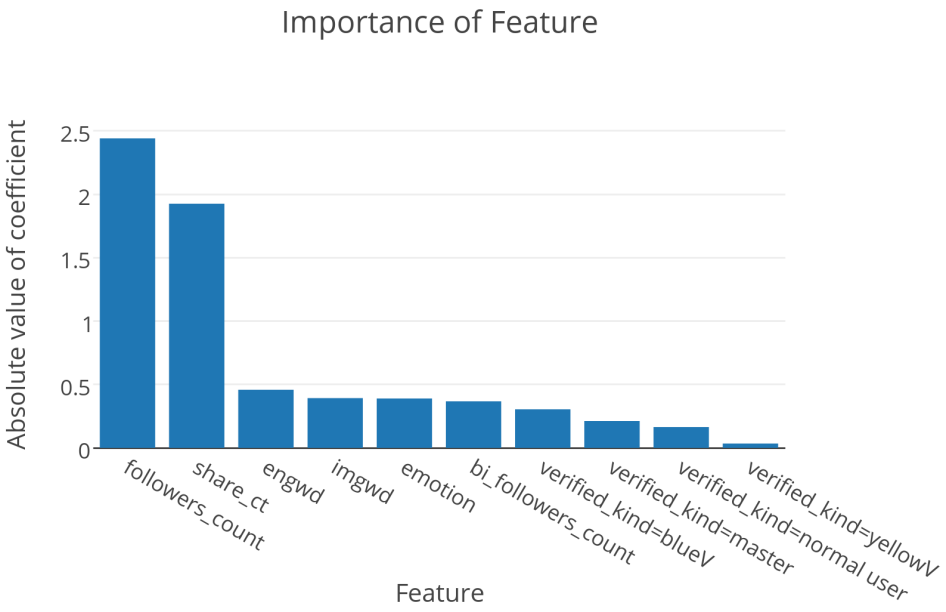

Figure 5.1: Importance of Feature by Logistic Regression

5.3 Training by Random Forest

To support the result from the logistic regression, we train a model using random forest with the same input features. In the algorithm, the number of tree is set as 500. Table 5.6 is the evaluation of the trained model using "30%" testset. The accuracy is 0.817. The importance of each feature is drawn in histgram in Fig. 5.2

	precision	recall	f1-score	support
False	0.82	0.82	0.82	726
True	0.82	0.82	0.82	719
avg/total	0.82	0.82	0.82	1445

Table 5.6: Model Performance trained by Random Forest

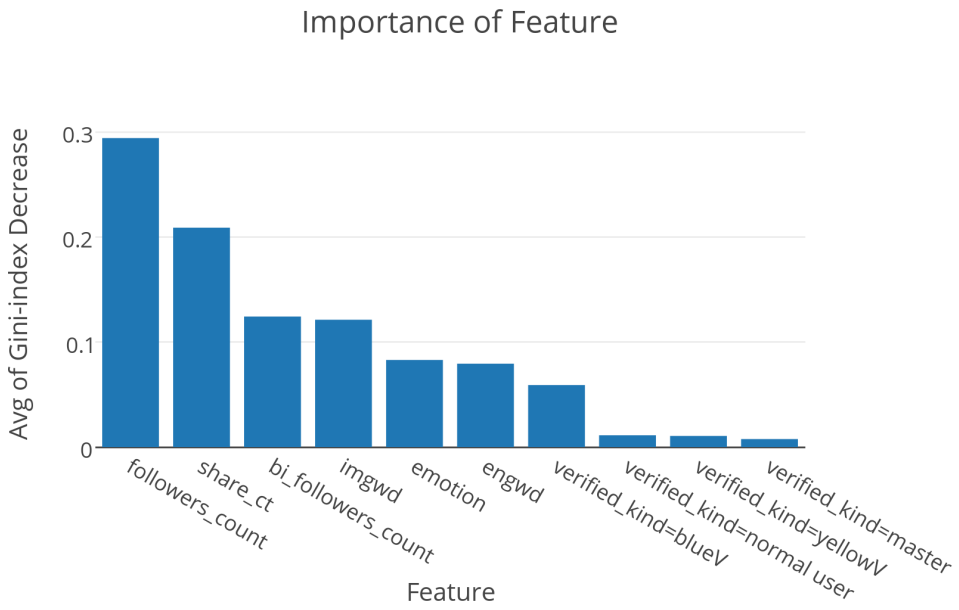

Figure 5.2: Importance of Feature by Random Forest

In Fig. 5.1 and 5.2, "FollowersCount" and "ShareCount" takes the first and second place, means these two features are most important among these input feature.

Random forest can reduce the impact from the correlated input feature while the logistic regression cannot. Thus we design a comparison between these two algorithm using all the features we introduce before the CFS procedure and draw two histgrams in Fig. 5.3 and Fig. 5.4.

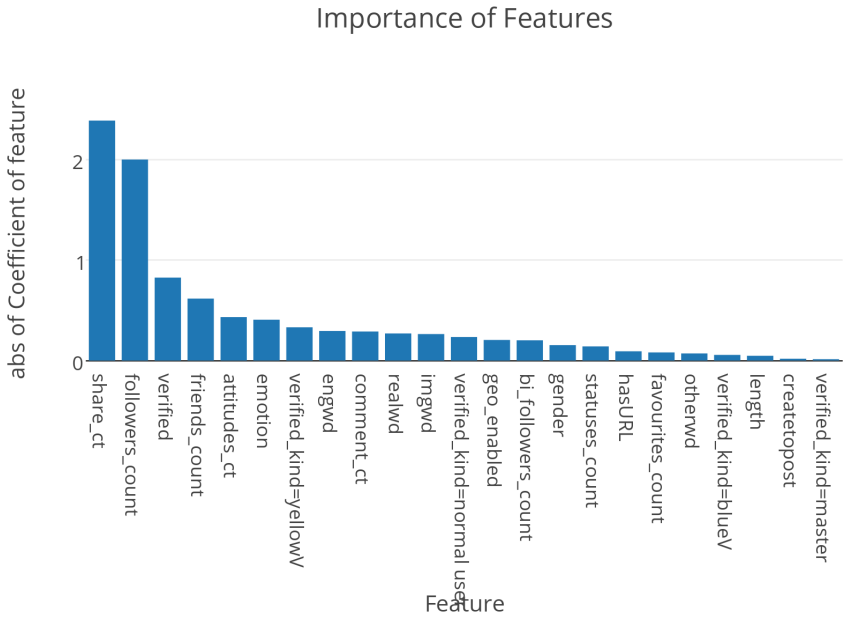

Figure 5.3: All Feature trained by Logistic Regresson

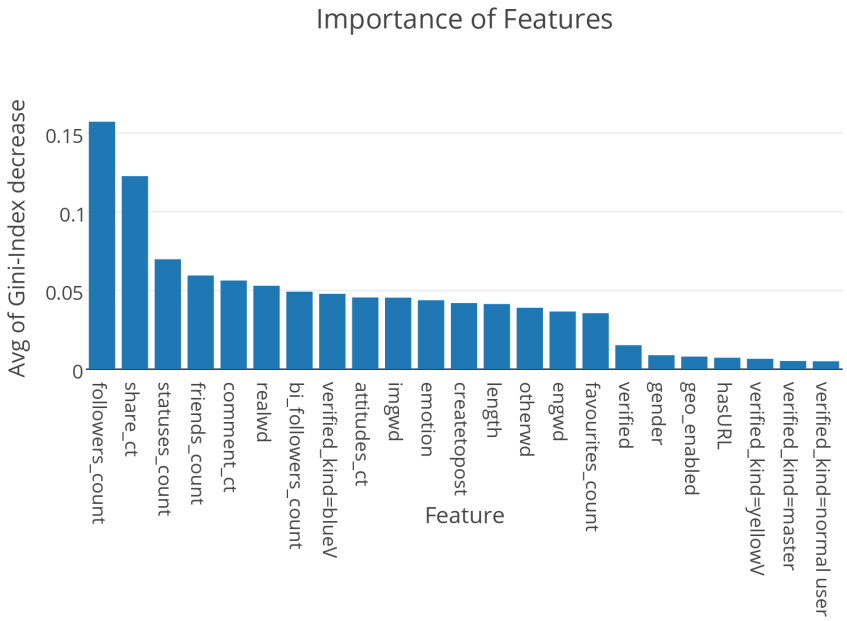

Figure 5.4: All Feature trained by Gini-based Random Forest

From this two figures, most of the features are disordered each other. Take "verified" as the example, which is not be selected by the CFS. It means that this feature is dependent on some other features. Actually it is quite related with the "VerifiedType" and "VerifiedKind" feature. If taking this feature into the logistic regression training procedure, the coefficient of it may be inaccurate. In Fig. 5.3, the coefficient of "verified" is measured as a large value, but actually random forest tells us it should be ranked not at very high pisition in Fig. 5.4.

Even if random forest can reduce the bias from the feature-feature inner-correlation, it has the drawback when dealing with the data including categorical features with different level. It is biased of preferring those features with more categories. For example, we have a nominal type and a numeric type feature. The numeric can be seen as an infinite categories feature compared with the nominal feature. To remove this bias, we run a random forest algorithm to measure the importance of features based on the mean decrease of accuracy when permuting on the data.

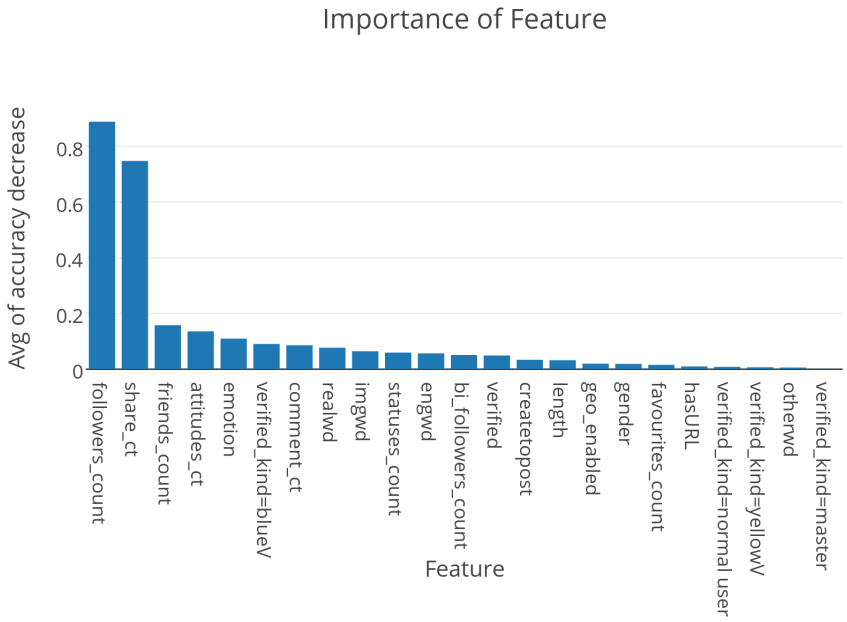

Figure 5.5: All Feature trained by Permutation-based Random Forest

The result is drawn in Fig. 5.5

This figure also indicates the "FollowersCount" and "ShareCount" have strong incluence on the model performance. Permutating these two features decreases model accuracy by over 80% and 78%. We can also see the rank of "BlueV" feature increase. Actually if we find the dataset with "BlueV", we find that the ratio of rumor and not rumor is $1:5$ among 1132 "BlueV" samples. It explains that the "BlueV" feature is relatively importance but its score is limited by the small number of sample.

This method also can work on the situation of correlated input feature because the score of importance is evaluated after the model is trained. We permute only one feature at a time and calculate the score. If the data of correlated feature are permuted one by one, the scores of them are still very close.

Chapter 6

Conclusion

6.1 Conclusions

In this project, we use linear model and tree model to indicate the importance of features. We use CFS to remove the bias from correlation between input features. We also use permutation-based random forest to prevent the bias from different number of category. We also have evaluate the performance of model we train. The good performance also support the correctness of feature importance.

We find that linear model is convenient and feasible to represent the importance of feature by their coefficient. But it is prone to suffer the bias on multiple correlated feature regarding their importance. Thus if we want to use linear model, we need to determine that there is little correlation between input features.

We also find that random forest can reduce the bias of correlation between input features. But it also has a bias on the different level of category on features. We find that permutation-based random forest can work well on this situation and we find that the feature "BlueV" is important using this method.

6.2 Future Work

We think correlation between feature still have bias. We can find more methods to remove it.

We can also extract more feature to analyze using natural language processing technique.

Concerning the dataset, we can also use some outlier detection technique to remove noise or outlier that make our evaluation more accurate.

Actually this project is only related to theoretical research. In the future, we can use the knowledge found in this project to build an application and help us identify the credibility of Weibo in our daily life.

References

[1] X. J. Tang, Chinese Academy of Social Sciences, and Institute for Journalism Communication Research. *Annual report on development of new media in China*. Social Sciences Academic Press(China), 2013.

[2] Carlos Castillo, Marcelo Mendoza, and Barbara Poblete. Information credibility on twitter. In *Proceedings of the 20th international conference on World wide web*, pages 675–684. ACM, 2011.

[3] Vahed Qazvinian, Emily Rosengren, Dragomir R Radev, and Qiaozhu Mei. Rumor has it: Identifying misinformation in microblogs. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 1589–1599. Association for Computational Linguistics, 2011.

[4] Fan Yang, Yang Liu, Xiaohui Yu, and Min Yang. Automatic detection of rumor on sina weibo. In *Proceedings of the ACM SIGKDD Workshop on Mining Data Semantics*, page 13. ACM, 2012.

[5] Ke Wu, Song Yang, and Kenny Q Zhu. False rumors detection on sina weibo by propagation structures. In *IEEE International Conference on Data Engineering, ICDE*, 2015.

[6] Christopher M Bishop. *Pattern recognition and machine learning*. springer, 2006.

[7] Ahmed Hassan, Vahed Qazvinian, and Dragomir Radev. What's with the attitude?: identifying sentences with attitude in online discussions. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 1245–1255. Association for Computational Linguistics, 2010.

[8] Jacob Ratkiewicz, Michael Conover, Mark Meiss, Bruno Gonçalves, Snehal Patil, Alessandro Flammini, and Filippo Menczer. Detecting and tracking the spread of astroturf memes in microblog streams. *arXiv preprint arXiv:1011.3768*, 2010.

[9] Isabelle Guyon and André Elisseeff. An introduction to variable and feature selection. *The Journal of Machine Learning Research*, 3:1157–1182, 2003.

[10] Ian Jolliffe. *Principal component analysis*. Wiley Online Library, 2002.

[11] L De Lathauwer, B De Moor, J Vandewalle, and Blind Source Separation by Higher-Order. Singular value decomposition. In *Proc. EUSIPCO-94, Edinburgh, Scotland, UK*, volume 1, pages 175–178, 1994.

[12] Jacob Benesty, Jingdong Chen, Yiteng Huang, and Israel Cohen. Pearson correlation coefficient. In *Noise reduction in speech processing*, pages 1–4. Springer, 2009.

[13] Claude Elwood Shannon. A mathematical theory of communication. *ACM SIGMOBILE Mobile Computing and Communications Review*, 5(1):3–55, 2001.

[14] Roberto Battiti. Using mutual information for selecting features in supervised neural net learning. *Neural Networks, IEEE Transactions on*, 5(4):537–550, 1994.

[15] Zenglin Xu, Irwin King, Michael Rung-Tsong Lyu, and Rong Jin. Discriminative semi-supervised feature selection via manifold regularization. *Neural Networks, IEEE Transactions on*, 21(7):1033–1047, 2010.

[16] Ron Kohavi and George H John. Wrappers for feature subset selection. *Artificial intelligence*, 97(1):273–324, 1997.

[17] Pavel Pudil, Jana Novovičová, and Josef Kittler. Floating search methods in feature selection. *Pattern recognition letters*, 15(11):1119–1125, 1994.

[18] Songyot Nakariyakul and David P Casasent. An improvement on floating search algorithms for feature subset selection. *Pattern Recognition*, 42(9):1932–1940, 2009.

[19] David E Golberg. Genetic algorithms in search, optimization, and machine learning. *Addion wesley*, 1989, 1989.

[20] Jihoon Yang and Vasant Honavar. Feature subset selection using a genetic algorithm. In *Feature extraction, construction and selection*, pages 117–136. Springer, 1998.

[21] Jihoon Yang, Rajesh Parekh, and V Konavar. Distal: An inter-pattern distance-based constructive learning algorithm. In *Neural Networks Proceedings, 1998. IEEE World Congress on Computational Intelligence. The 1998 IEEE International Joint Conference on*, volume 3, pages 2208–2213. IEEE, 1998.

[22] Naomi S Altman. An introduction to kernel and nearest-neighbor nonparametric regression. *The American Statistician*, 46(3):175–185, 1992.

[23] John A Hartigan and Manchek A Wong. Algorithm as 136: A k-means clustering algorithm. *Applied statistics*, pages 100–108, 1979.

[24] Arthur Zimek, Erich Schubert, and Hans-Peter Kriegel. A survey on unsupervised outlier detection in high-dimensional numerical data. *Statistical Analysis and Data Mining: The ASA Data Science Journal*, 5(5):363–387, 2012.

[25] Yaser S Abu-Mostafa, Malik Magdon-Ismail, and Hsuan-Tien Lin. *Learning from data*. AMLBook, 2012.

[26] Ting Yu, Tony Jan, John Debenham, and Simeon Simoff. Incorporating prior domain knowledge in machine learning: A review. In *AISTA 2004: International Conference on Advances in Intelligence Systems-Theory and Applications in cooperation with IEEE Computer Society*, 2004.

[27] Dimitri P Bertsekas. *Constrained optimization and Lagrange multiplier methods*. Academic press, 2014.

[28] Andrew Y Ng. Feature selection, l 1 vs. l 2 regularization, and rotational invariance. In *Proceedings of the twenty-first international conference on Machine learning*, page 78. ACM, 2004.

[29] Donald E Farrar and Robert R Glauber. Multicollinearity in regression analysis: the problem revisited. *The Review of Economic and Statistics*, pages 92–107, 1967.

[30] Mark A Hall. *Correlation-based feature selection for machine learning*. PhD thesis, The University of Waikato, 1999.

[31] MA Hall. [a]correlation based feature selection for discrete and numeric class machine learning. In [o] *Proc. 17th Int'l. Conf. Machine Learning*, 2000.

[32] S Rasoul Safavian and David Landgrebe. A survey of decision tree classifier methodology. *IEEE transactions on systems, man, and cybernetics*, 21(3):660–674, 1991.

[33] Lidia Ceriani and Paolo Verme. The origins of the gini index: extracts from variabilità e mutabilità (1912) by corrado gini. *The Journal of Economic Inequality*, 10(3):421–443, 2012.

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

[35] Carolin Strobl, Anne-Laure Boulesteix, Achim Zeileis, and Torsten Hothorn. Bias in random forest variable importance measures: Illustrations, sources and a solution. *BMC bioinformatics*, 8(1):25, 2007.