NLP Analysis of Weibo User Behavior and Popularity Prediction

Qixuan Ma
UC San Diego, Mathematics Department
9500 Gilman Dr. # 0112, La Jolla, CA, USA
q5ma@ucsd.edu

Abstract

In this study, we explored several methods of predicting the popularity of a post on Weibo, by treating it as either a classification task or a regression task, and training different models with different sets of hyperparameters obtained from grid search with cross validation. Overall, we found that a pure Term Frequency - Inverse Document Frequency (TF-IDF) model does not achieve significant improvement over a Bag of Words (BoW) model in popularity prediction based on Chinese social media posts.

1 Introduction

Weibo (micro-blog) is a Chinese micro-blogging website. It is one of the most popular sites in China, with a market penetration similar to that of Twitter. In this project, we will attempt to predict user behavior (the amount of likes, comments, and forwards) based on the content of the Weibo post, the user, and the date and time.

2 Data Description

The **weibo_training_data.txt** dataset was provided by Tianchi, a data science platform by Alibaba Cloud, as part of the datasets for the Sina Weibo Interaction-prediction Competition in 2018. In addition to the dataset we plan to use, Tianchi also provided datasets in user action, follower data, and blog data. To keep this project within scope, we have decided to not use them.
The **weibo_training_data.txt** contains 1225088 observations across seven different variables: user id, weibo id, time of posting, forward count, comment count, like count, and the content of the weibo.
we plan to construct three separate regression models to predict forward count (within one week after posting), comment count (within one week after posting), and like count (within one week after posting) respectively, based on the text features of the weibo content, the time the weibo was posted, and the user who made the post.

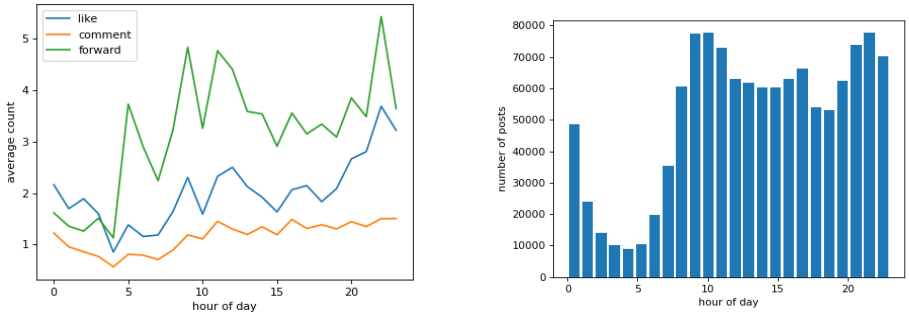
3 Related Work

Social media popularity prediction based on text content has been researched extensively in the past few years. In their report "Predicting Popularity", Zeng et al. explored the relationship between the popularity of the tweet (as measured by amount of likes) and the number of hashtags, links, and mentions present in the tweet [1].

In their paper "Twitter Likes Prediction Using Content and Link based Features", Amjad and Zahra built a prediction model for tweet like count using tweet topic, average follower, and etc [2].
In a study by Daga et al., the authors used a TF-IDF and Doc2Vec approach to perform a classification task on the amount of likes and comments on a tweet.
However, the existing literature for a similar system in the Chinese language is limited. In this study, we will attempt to innovate upon models discribed in existing literature and apply them to Chinese, a language that is not space separated and has more ambiguous word meaning and phrase separation than English and other Germanic and Romance languages. We will use a well developed package, "Jieba" , developed by a team of developers lead by Sun Junyi.

4 Exploratory Analysis

First of all, we wanted to see what temporal relationship exists between the popularity of the weibo (as measured by amount of likes, comments, and forwards) and the time the weibo was posted. we discarded three features, seconds, minutes, and seconds because we do not perceive them to be correlated with the popularity of the weibo. From Figure 1, we can see

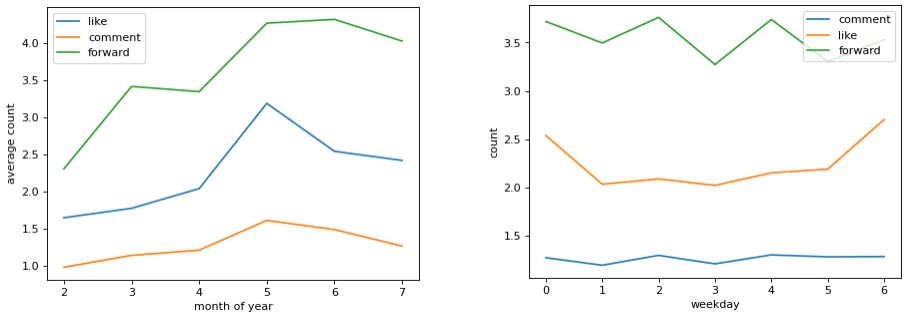


(a) Time of Day Count (b) Time of Day vs Popularity

Figure 1: Time of Day

that there is, despite noisy, some relationship between the popularity of the weibo and the time of day it was posted. We notice a large dip in the popularity of the weibos that are posted during 4am and 5am as users are likely dormant during these periods. We can also see a upwards trend for weibos posted in the evening and at night. All 3 of our metrics (like, comment, forward) peaked for weibos posted between 10pm and 11pm at night, which matches our expectations for user behavior of phone usage before bedtime. In addition, due to the lack of posts made in the morning, weibos from previous night would still retain high popularity.
In addition, posts made in the morning will also on average receive more likes, comments, and forwards according to the plot. We also see a dip in both the popularity and the total number of weibo posted in the afternoon and during dinner hours, which we did not initially expect.
We are pretty confident that using time of day as a feature for the model can improve the accuracy of the model. In stark contrast to the reasonably clear relations revealed by the last set of figures, the features plotted in Figure 2 (month and weekday) shows a less promising temporal relationship between them and popularity of the weibo. In subplot b, we can see that posts made in weekends usually are more popular than posts made in weekdays, which is expected since people obviously spend more time on social media platforms in weekends.

For the Month vs Popularity plot, we do not expect any large deviation from the mean. However, we do see a spike in posts made in May and June. We theorize that this might be



(a) Month vs Popularity (b) Weekday vs Popularity

Figure 2: Other Time-Related Features

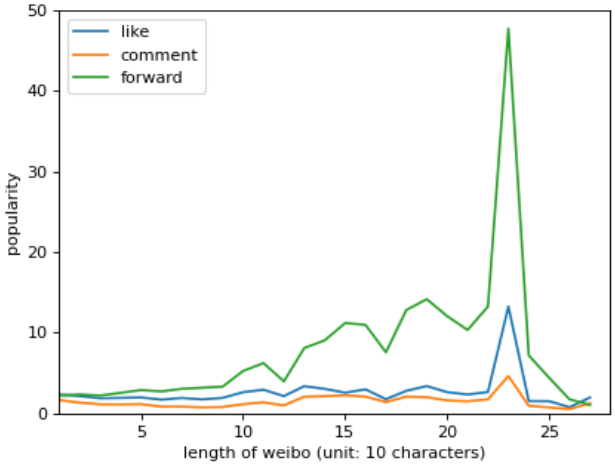


Figure 3: Weibo Length vs Popularity

due to some news event causing wide scale discussion rather than any inherent temporal relationship. According to the reasons stated above, we do not think Month and Weekday features will improve the performance of our model.

From Figure 3, we can see a a clear trend in the relationship of the length of the weibo and the popularity it receives. There is an overall positive correlation between the length of the weibo post and the popularity, which peaks when the weibo is about 230 characters long (the upper limit is 280 characters).

Looking at Figure 4, we can see that whatever trend might exist in there is very high in variance, which would likely result in a lower performance overall. One thing to point out in gathering data about hashtags used (topics included in the weibo), we only counted hashtags as being part of a topic if they appear in pairs of two. In weibo's syntax, a topic has to be wrapped by hashtags on both ends. For instance "#TIL#" would be a valid syntax while "#TIL" would be parsed as just plain text. By doing this, we hope to weed out the possibility of categorizing something as a hashtag enclosed topic while it is just part of an URL address for instance.

Taking a look at Figure 5, we can see that the trend for number of mentions included in the weibo and the popularity it receives have some central tendency toward 7 mentions. But the

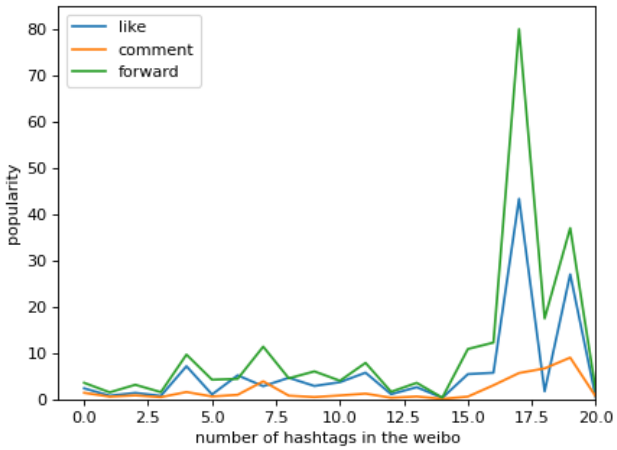


Figure 4: Number of Topics Included vs. Popularity

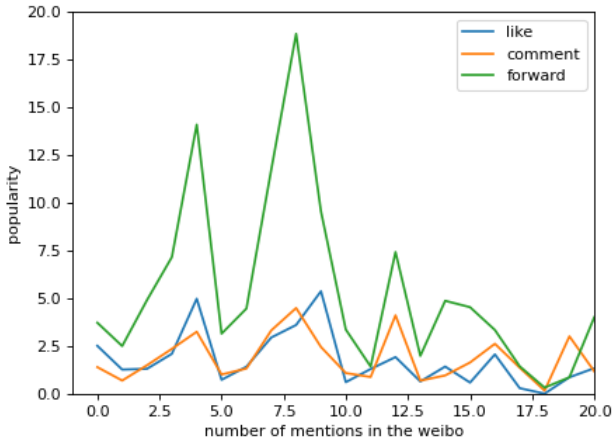


Figure 5: Users Mentioned vs. Popularity

graph looks extremely noisy and does not look like it would improve the accuracy of the model in any meaningful way. Thus, we will leave it out of the model.

5 Models

5.1 Regression Models

For the regression problem, we will attempt to predict a combined score (arithmetic mean) of the 3 categories of popularity metrics, like, forward, and comment. This will give us a balance between we will then train 3 types of regression models on these features: ordinary linear regression, LASSO regression, and ridge regression). Ordinary linear regression is a pretty basic and general model for regression tasks which tends to perform decently well. Ridge regression has the benefit of the ability to assign penalty terms to further improve the accuracy of the model. LASSO regression has the ability to set weights for certain features all the way down to zero, which can hopefully improve the accuracy of the model by ignoring terms that would otherwise be detrimental to the overall performance. To avoid overfitting and underfitting, we will use a 5-fold cross validation for each model with a fixed random state to ensure consistency between models and between runs.
In addition, this model is robust and can be applied to datasets in a much larger scale given enough computing power. One limitation is the implementation of the feature vector. To

be able to work with sklearn models, I had to convert the original scipy sparse matrix to an array of numpy arrays, which significantly increased the space complexity for the model, decreasing the scalability of this method.

5.1.1 Baseline Model

For the baseline model, we will use a simple Bag Of Words (BOW) model with each feature vector composing of the frequency representation of the content of the weibo, time of day of the post, weekday, and length of the post.

5.1.2 Improved Models

For the improved models, we will build a Term Frequency - Inverse Document Frequency (TF-IDF) representation for the aforementioned BoW feature.

6 Performance Metrics

To compare the performance of regression models, we will use mean squared error (MSE), absolute squared error (ASE), and explained variance (R^2) as defined below:

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2$$

$$\text{MAE} = \frac{\sum_{i=1}^{n}|y_i - x_i|}{n}$$

$$\text{R}^2 = 1 - \frac{RSS}{TSS}$$

where

$$\text{RSS} = \sum_{i=1}^{n}(y_i - f(x_i))^2$$

$$\text{TSS} = \sum_{i=1}^{n}(y_i - \bar{y})^2$$

These metrics were chosen because we believe it provides good insights into the performance of the model from several perspectives: how robust is the model (is it susceptible to outliers?), how accurate is the model on average (how far is the predicted trend from the actual data?), and how much does the model correlate with the dataset (how much of the variance is explained by the model?).

7 Results

To tune hyperparameters for the aforementioned models, I used the GridSearchCV method in the sklearn library. I supplied each model a range of values in each of its hyperparameters. The model selection process yielded the following set of hyperparameters:
For Ridge Regression: alpha = 1.0, solver = sparse_cg
For Linear Regression: default, not much parameter options
For LASSO Regression: alpha = 0.1 Looking at Table 1, we can see that overall, the results of our classification models are not the most desirable. The lowest MSE achieved was 15,692.10. The lowest MSE achieved was 11.181. The largest R^2 value (explained variance) is 0.03. In terms of difference between the 3 models used (Linear Regression, Ridge Regression, Lasso Regression), we find that the performance across several different metrics for Linear Regression and Ridge Regression is very similar, while LASSO Regression is able to minimize Mean Absolute Error at the cost of a lower R^2. We also find that using a TF-IDF model as compared to a BoW model does not bring significant improvements to the performance of the models. Due to the noisy nature of non-text features (length, hashtag count, etc), using them in model training also do not provide a significant advantage.

Performance Metrics for Regression Models			
Model	MSE	MAE	EV
LR (BoW)	15,692.80	14.584	**0.030**
LR (BoW with Features)	**15,692.16**	14.643	**0.030**
LR (TF-IDF)	15,808.45	**13.820**	0.023
LR (TF-IDF with Features)	1,5807.852	13.867	0.023
Ridge (BoW)	15,693.17	14.587	**0.030**
Ridge (BoW with Features)	**15,692.10**	14.64	**0.030**
Ridge (TF-IDF)	15,808.37	**13.81**	0.023
Ridge (TF-IDF with Features)	15,807.77	13.857	0.023
Lasso (BoW)	**15,774.24**	11.602	**0.025**
Lasso (BoW with Features)	15,993.90	11.663	**0.025**
Lasso (TF-IDF)	15,994.74	**11.181**	0.010
Lasso (TF-IDF with Features)	15,993.90	11.183	0.010

Table 1: Regression Model Performance

8 Discussion

There are several areas where this experiment can be improved on.

First of all, the data engineering might not be thorough enough. For instance, we did realize the potential issue of skew of the distribution of the outcome variable, but did not take any steps to address it from the beginning. This could be addressed in several ways. We could introduce a normalization method such as Z-score or log scaling to treat the outcome variable. In addition, we could instead use relative popularity instead of absolute popularity. Using percentiles, we can then make sure the data is normally distributed. Both these two methods, however, might still not be enough to address the fact that a great number of samples have the same popularity at 0. The end result would likely end up retaining its skewness.

More over, potential problems could arise from the cross validation we employed. My implementation used a index based 5-fold cross validation, meaning that the dataset is split into 5 sets of continuous samples, which could be problematic in this situation. The dataset was ordered by date, meaning that each "fold" will contain information from a continuous range of dates. By doing this, we are assuming uniformity amongst weibo posts from different time periods, which is not a very good assumption. A better form of performing cross validation might be to shuffle the data prior to splitting into training or testing sets. In addition, we propose splitting the dataset into different strata according to time information and draw an equal amount from each strata to make up the training and testing sets, ensuring an equal representation of samples from all time points in each fold.

In addition, we believe that the performance of our model can be further improved by extracting more features from the text content such as topic and part of speech. We can infer topics by doing clustering analysis on common patterns in hashtag usage and user mentions. The Jieba package also have support for part of speech tagging based on deep learning. We can then assign different weights to differnent part of speech components, which will likely improve our results. Doing so, however, will require significant more time and computing power which we do not possess.

Last but not least, we would like to propose a different model for future studies. Instead of using the absolute popularity or relative popularity as the outcome variable, we can split the weibos into different buckets of popularity status and change our question accordingly. This

way, we can convert this into a multi-class classification problem solvable with multi-layer perceptron, kernel support vector machine, or linear discriminant analysis. Fuzzy logic might also be useful in this circumstance.

9 Appendix

9.1 Code

All code used in for this project can be found in my GitHub repository at https://github.com/Harrison-Q-Ma/WeiboRecommender.

9.2 Data

The dataset used for this project can be found at https://tianchi.aliyun.com/dataset/dataDetail?dataId=51#1.

10 Acknowledgements

I thank Professor Julian McAuley and the CSE 158/258 instruction team for planning and instructing CSE 158, which furthered my knowledge in text mining and recommender systems, making this project possible. I thank Stella Ma for her company and support throughout this quarter. I also thank Sophie Hao, for suggestions and proof reading.

11 Reference

1. Daga, Ishita, et al. "Prediction of Likes and Retweets Using Text Information Retrieval." Procedia Computer Science, vol. 168, 2020, pp. 123–128., https://doi.org/10.1016/j.procs.2020.02.273.
2. Zeng, Belinda, et al. "Predicting Popularity". http://belindazeng.github.io/goingviral/
3. Amjad, Tehmina. Zahra, Hafsa. "Twitter Likes Prediction Using Content and Link based Features." PJCIS (2017), Vol. 2 No. 1 : 1-15. http://173.208.131.244:9060/xmlui/bitstream/handle/123456789/742/Article
4. https://github.com/fxsjy/jieba