A MODEL FOR DETERMINING TWEET POPULARITY VIA PREDICTION METHODS

Farhan Hormasji
Dept. of Computer Science
Michigan State University
hormasji@cse.msu.edu

Bonnie Reiff
Dept. of Computer Science
Michigan State University
reiffbon@cse.msu.edu

ABSTRACT

With the spike in popularity of social media sites over the past 10 years, it is important for some to use these sites to their advantage in order communicate information to a large user-base. This project focuses on the Twitter social networking service and attempts to answer two questions. The first goal of the project is to determine the features available through the public Twitter API that contribute the most to a particular Tweet's popularity. Secondly, the project aims to expand on the knowledge gained from the first objective to determine whether the contributing features can determine whether a Tweet will be "popular" after a designated period of time, where popularity is defined by the number of times a Tweet has been "Favorited." The project uses a stream of Tweets collected using the Twitter Public Stream API. After labeling the data as "popular" or "not popular", the project then uses sequential feature selection and a binary support vector machine (SVM) classifier with k-fold cross validation to achieve the two objectives. The analysis reveals that if you know the number of URLs in a Tweet, the favorite count of a Tweet after an hour, and if the Tweet under observation quoted another Tweet, then you have approximately 97% chance of correctly predicting if a Tweet will be popular or not using our trained SVM model.

Keywords

Twitter, Support Vector Machine, Sequential Feature Selection, K-Fold Cross Validation

1. INTRODUCTION

Motivated by a desire to apply data mining techniques to social media networks, the student team made an initial decision to work within the Twitter application domain because of the easy to interact with, publicly available APIs. Twitter is an online social networking service that allows users to post short messages called "Tweets." In addition to normal text, these Tweets can contain URLs, embedded images or videos, username "tags", and "hashtags" (among other features), where a hashtag is a user-defined content tag that allows the service to better identify Tweets with a specific theme or content.

The networking service offers several ways in which a user can indicate their interest in another user's Tweet. Firstly, a Tweet may be "Retweeted" or "Quoted", where the difference between these two is that a "Retweet" is a re-post of the original Tweet by another user whereas a "Quoted" Tweet is a copy of the original Tweet with additional commentary by another users. Twitter users can also "Favorite" Tweets, which is similarly to a "Like" in the Facebook domain.

This paper focuses on determining what attributes of a Tweet (additionally considering attributes of the user posting the Tweet) best determine its popularity. This type of information is extremely useful to users that are attempting to communicate information outside of their group of followers, as Retweets by followers can result in the followers of followers viewing the original Tweet and so on. For example this may be useful for users generally only followed by a particular group of people (i.e. politicians) or for accounts posting Tweets with the goal of product advertisement. The attributes that we examine are publicly available (given that the requesting user has a Twitter login) via the Twitter Public Stream and Search APIs, which provide a stream of Tweets and the ability to search for Tweets based on ID numbers, respectively. After applying sequential feature selection (SFS) to filter out irrelevant attributes, we apply a binary support vector machine (SVM) classifier to train a model to predict whether a given Tweet will become popular within 24 hours of being posted based on the remaining attributes. The major findings of this paper are that the correct combination of binary and count-type features describing attributes of a Tweet can result in a Correct Detection Rate of over 97% given a correctly trained SVM model. In addition, this paper presents evidence that features such as presence of a popular hashtag are not good predictors of Tweet popularity.

The remainder of this paper will proceed as follows. We will first formalize the research questions that this paper wishes to answer and summarize related work that was found during research in the initial stages of the project. In Section 3, we will provide detailed information on the work that was done to implement the experimental setup. Section 4 provides specifics on the setup of the experiments as well as the results that were obtained for the various datasets used. In the sections following that, we then provide an analysis of our results, summarize the conclusions that we can draw to answer our research questions based on the project work, and provide some thoughts on ways that the project work can be augmented in the future.

2. PROBLEM STATEMENT

Formally, the goal of this project is to answer two related questions: (1) What attributes of a Tweet (including information about the actual Tweet, the user who posted it, and initial reaction information) contribute most heavily to its popularity? and, (2) Can we use a set of attributes of a Tweet to reliably predict a Tweet's popularity? Note that for these two questions, the "popularity" of a Tweet is defined as the number of times the Tweet has been Favorited at the end of a 24 hour period after the Tweet has been published. Data for this problem will consist of one Tweet per record, with the attributes being information about the Tweet and the user who posted the Tweet, as well as generated features summarizing the initial reaction of Twitter users to the post. Each Tweet will be labeled as "popular" or "not popular" according to a threshold, allowing the team to build a model to perform binary classification on any given unlabeled Tweet, where binary classification is defined as the task of predicting a two-class value attribute based on the values of the other attributes in the dataset.

2.1 Related Work

Initial research into prior work that has been done on determining what makes Tweets popular led to many works by Tauhid Zaman, an Assistant Professor of Operations Management at the MIT Sloan School of Management. Zaman's research in this area includes a journal article on the use of a probabilistic model for evolution of Retweets using a Bayesian network approach, forming his predictions based on observations of how often a Tweet is Retweeted and the network/graph structure of the users Retweeting the original post[7]. He also presented a trained probabilistic collaborative filter model for the prediction of future Retweets at the 2010 conference on Neural Information Processing Systems (NIPS). The results of this study were that the most important features for prediction are the identity of the source of the Tweet and the "Retweeter"[8]. As these are not attributes available to us in our data, the goal of our project work will be to augment Zaman's findings with other potential predictors for Tweet popularity.

Much of the other referenced work that was found for the Twitter application domain revolves around sentiment analysis (also known as opinion mining), which is defined as the use of text analysis to extract subjective information from the source material. This type of analysis is outside of the scope of this project.

3. METHODOLOGY

3.1 Data Collection and Preprocessing

Two PHP programs were written for the data collection portion of the project, one to interact with the Public Stream API and one to interact with the Search API. Both of these programs utilize the Twitter library created by Matt Harris, which can be found on Github under the project name "tmhOAuth." After creating an application on Twitter to obtain authentication keys, this Twitter library communicates "GET" and "POST" requests to Twitter and returns the responses to the calling function.

The first step in the creation of the program written to interact with the Public Stream API, called streamTweets, was to determine the Tweet attributes to be output to the resulting CSV file. A Tweet has many associated "fields" containing information about the text, metadata (for example, the geographic location of the Tweet), indications as to whether the Tweet is a Retweet or a Quoted Status, information about the user who posted the Tweet, and information related to the interaction of the authenticating user with the Tweet. The full list of fields that is returned via a streaming request can be found in Twitter's Developers documentation. Initially, deprecated/unused values, redundant information, and all fields related to the interactions of the authenticating user with the Tweet were ignored. Under further review of the data collection files after the first set of runs, more fields with information that was not relevant to the research questions were removed. In its final version, this program outputs features regarding the "entities" of a Tweet (the number of users mentioned/tagged, the number of URLs present, the number of hashtags present, and the number of attached media files), whether the Tweet is a Quoted status, whether the Tweet is in response to another Tweet, whether the Tweet contains sensitive information, and user information. User information includes values such as how many statuses a user has posted, how many followers a user has and how many other users they are following, and whether the user is using a default profile scheme and/or image (among other features).

The constructed streaming request has two important parameters: language and track. The "language" parameter specifies that all returned Tweets should be in English. The "track" parameter defines a comma-separated list of phrases, at least one of which should be contained in each returned Tweet. Data collection for this project was performed using two different methodologies for this parameter. For one dataset, a set of 22 common English words were used to stream as many Tweets as possible. For a second dataset, the parameter was set to 12 "trending" topics (according to the Twitter homepage) at the start time of the data collection.

In order to be added to the list of collected data, a Tweet from the stream must follow two requirements: (1) the user who posted the Tweet must have more than a certain number of followers (where the certain number is defined as 500 for the common word data collection and 2000 for the trending word data collection), and (2) the Tweet may not already by a Retweet. The reasoning behind these two requirements comes from initial data collection experiments. The team found that if Tweets from users with a fewer number of followers than that threshold were allowed, the data would contain too many data points labeled as "not popular". Thus, the team chose to bias the data collection with this filter, meaning that we are inherently considering a low value in this field to be an indicator that the Tweet will not be popular (although the inverse of this statement is not necessarily true). In regards to the second requirement, because our popularity is defined as the Favorite count 20 hours after the original post, we needed to only obtain data that did not already have associated Retweet and Favorite counts at the time of streaming.

For the creation of the CSV file containing Tweet attributes, the streamTweets program is responsible for several preprocessing activities. Firstly, the program removes all commas and newlines before writing the data to the file to prevent any formatting errors when importing the data into a processing utility. Secondly, the program generates binary presence and integer count attributes based on the Tweet fields retrieved by the streaming process. For example, if the Tweet is a Quoted status, rather than use the Quoted status ID as the attribute, the program outputs a "1" to the column for this indication. Lastly, the program parses the hashtags in each Tweet and tracks the count of each of these hashtags in an array. At the end of the data collection, the top 20 hashtags from the streaming session are printed to a file. The binary presence of the top 10 hashtags (case insensitive) are then added to the data collection file as additional attributes, where the presence of a hashtag within a Tweet was determined via the Microsoft Excel SEARCH formula.

The program written to interact with the Search API, called searchTwitter, uses the “statuses/lookup” GET request, which returns Tweet objects for up to 100 Tweets per request, with a rate limit of 180 requests in 15 minutes. The main processing function of the program takes as input a file built by the streamTweets program containing lines of a maximum of 100 Tweet ID strings per line, with a maximum of 15 lines of Tweet ID strings per file. For each returned Tweet, the Retweet and Favorite counts are obtained from the field information and are printed to a CSV file along with the ID and full text of the Tweet. Unfortunately, the Search API consistently only returned values for 70%-80% of the streamed Tweets and returns the Tweets out of the order in which they were searched. The values in the output CSV file from the searchTwitter program were later matched to the output values from the streamTweets program via an Excel VLOOKUP formula and combined into a single data file.

In addition, a BASH shell script program, collectData.sh, was created to coordinate the timing of the streaming and search runs. For each iteration (where the number of iterations is specified in the script), the program generates the time-tagged filenames to be passed as inputs to the streamTweets program, calls the streaming program, and then subsequently calls the searchTwitter program at 10 minutes, 30 minutes, and 1 hour intervals after the completion of the data collection (and appropriately renames the output files from the program). After the completion of the collectData program, a 20 hour search is run at the appropriate time to get longer-term Retweet and Favorite count information.

3.2 *K*-Fold Cross Validation

Due to the limit in the amount of Tweets that could be retrieved from the API at one time, there was a limited amount of data available for analysis. To increase the amount of data available for training and validation of the model. *k*-fold cross validation was used. In *k*-fold cross validation, all the training data is partitioned into k subsamples; �– 1 subsamples are used as training data, and 1 subsample is used as the test data. A *k* of 5 was chosen, because after �= 5, it was experimentally determine that the results did not improve enough to justify the extra computation time.

3.3 Sequential Feature Selection

When we first used the streamTweets API, it resulted in 49 total features. After eliminating some obvious extraneous features (i.e. ID string, date/time stamp, etc.), we were left with 36 features. We aimed to reduce the dimensionality of the problem by eliminating two types of features. First, we want to eliminate features which hurt our correct detection rate. This occurs when the feature’s values have no correlation with the Tweet’s classification. The second type of feature we want to get rid of are features which are not needed as long as another feature is included. In other words, these are features which only help predict classification for a subset of items that another feature already helps predict. We will show examples of both types of features being eliminated in the Results (Section 4.2).

In order to determine which features were needed, we applied Sequential Feature Selection (SFS). SFS minimizes an objective function over all feasible feature subsets. In this case, the objective function is the Error Detection Rate for the SVM classifier. Features are sequentially added to an empty candidate set until the addition of further features does not decrease the Error Detection Rate.

3.4 Binary SVM Classification

Once the data is received from the Twitter API, each Tweet is classified as “popular” or “not popular”. Using *k*-fold cross validation, the classified data was split into a training and test set. The training set and its associated labels/classifications are trained using the Matlab’s SVM Classification toolbox. The resulting model is used to classify the test set. The performance of the model is compared to the actual classification and used to output the Correct Detection Rate, Error Detection Rate, and Confusion Matrix.

4. EXPERIMENTAL EVALUATION

4.1 Experimental Setup

All interactions with the Twitter APIs (streaming and search programs) were run on a Michigan State University engineering compute server. The server is a Dell PowerEdge R620 with 384GB of RAM and 48 Logical Cores running CentOS 6.6. This computer runs PHP 5.3.3 and CURL 7.19.7 and has OpenSSL installed, all of which are required for the streaming and search programs.

The output of the streamTweets program is a 31 feature dataset consisting of 24 raw attributes obtained from the Tweet API and 7 integer-type attributes related to the count of or binary presence of other raw Tweet attributes. The analysis of the presence of each of the top 10 hashtags in the Tweets adds another 10 attributes. The last 8 features are the Retweet and Favorite counts for all of the Tweets after 10 minutes, 30 minutes, 1 hour, and 20 hours.

The final dataset from the collection generated by searching on common English words contained 23,068 Tweets. This dataset, referred to hereon in as the “Common Keyword dataset” has 36 integer-type features (where the features are a subset of those described above generated by removed any String-type attributes). Of these attributes, a subset of 25 were used as predictor attributes and the 20 hour Favorite count was used as the target attribute. In order to classify each Tweet as “popular” or “not popular” a threshold of 10 was used for this value, where the number was determined based on the spread of the obtained data.

A second dataset, hereon in referred to as the “Trending Keyword dataset” was subsequently produced (as explained in Section 3.1) by requesting Tweets related to the trending topics at the beginning of the data collection session. This dataset has the same feature set as Common Keyword dataset.

All prediction-related activities referenced in Sections 3.2 through 3.4 were run in Matlab R2011a on a Dell laptop. In our experiments, training the model takes on average 30 seconds for the large dataset and 10 seconds for the small datasets, and classification of the Tweets in a single partition (from *k*-fold cross validation) takes approximately 5 minutes for the big dataset and 1 minute for the small datasets.

4.2 Experimental Results

Table 1 shows the final features used as predictors for the problem, where the features are a subset of all collected and processed attributes described in the previous sections. The table also shows the type of each attribute. Note that every feature is required to have an integer value based on the use of Matlab as the processing utility.

Table 1: Dataset Explanatory Attributes and Types

Count Type Features	Binary Type Features
url_and_media_count	quoted_status
user_mentions_count	binary_in_reply_to_user_id
hashtag_count	binary_in_reply_to_screen_name
user_statuses_count	binary_in_reply_to_status_id
user_followers_count	possibly_sensitive
user_following_count	user_contributors_enabled
user_list_count	user_geo_enabled
10 min RT	user_protected
10 min FAVE	user_verified
30 min RT	user_default_profile
30 min FAVE	user_default_profile_image
1 hr RT	popular_hashtag 1-10
1 hr FAVE	

Using these features, the resulting confusion matrix for the Common Keyword dataset is shown in Table 2. Recall that this dataset contains 23,068. Based on the results presented in the confusion matrix, we can calculate that this model resulted in a Correct Detection Rate (CDR) of 95.45%.

Table 2: Common Keyword Dataset Confusion Matrix

		Predicted Class	
		Popular	*Not Popular*
Actual Class	*Popular*	1258	87
	Not Popular	961	20762

All remaining experiments were performed on the smaller Trending Keyword dataset with the goal of manipulating the set of predictor features to answer specific questions and to see if better results could be obtained. Using SFS, we first obtained an optimal set of count and binary type features. This set, which represents the features that resulted in the best detection rate, contains url_and_media_count, hashtag_count, user_followers_count, 1 hr FAVE, quoted_status, user_contributors_enabled, user_protected, and user_verified.

The first question posed is whether the count-type features or binary-type features serve as better predictors of the chosen target attribute. Separate models were trained and tested for each category of attribute (using only the optimal features selected by SFS). The resulting confusion matrices are shown below in Table 3 and Table 4. The respective CDRs of these two sets of results are 94.65% and 77.85%, indicating that the set of count features make better predictors than the set of binary features. This is a somewhat intuitive result as count features contain more information (based on the value of the number) as opposed to a binary value.

Table 3: Count Features Confusion Matrix on Trending Keyword Dataset

		Predicted Class	
		Popular	*Not Popular*
Actual Class	*Popular*	188	57
	Not Popular	35	1440

Table 4: Binary Features Confusion Matrix on Trending Keyword Dataset

		Predicted Class	
		Popular	*Not Popular*
Actual Class	*Popular*	1260	166
	Not Popular	215	79

In our next experiment, we ran SFS again on the set of optimal features output by the first iteration of SFS. This algorithm returned two count features and one binary feature, namely url_and_media_count, 1 hr FAVE, and quoted_status. The resulting confusion matrix is shown in Table 5. The correct detection rate for this problem was 97.38%, therefore making this the optimal model over all experiments thus far.

Table 5: Optimized Feature Set Confusion Matrix on Trending Keyword Dataset

		Predicted Class	
		Popular	*Not Popular*
Actual Class	*Popular*	222	23
	Not Popular	22	1453

Lastly, we wished to test whether the use of popular hashtag analysis contributed at all to the correct classification of Tweets as "popular" or "not popular." It was initially hypothesized by the team that this type of analysis would be a useful trait, but SFS had not selected any of the popular hashtag columns as a potentially good feature. In order to implement this experiment, all features other than the popular hashtag columns were removed from the dataset. The results, shown in the confusion matrix in Table 6, produced a CDR of 85.76%. Although this is a fairly high CDR, we can observe that this was not a good prediction model based on the fact that not popular Tweets were correctly predicted.

Table 6: Hashtag Feature Set Confusion Matrix on Trending Keyword Dataset

		Predicted Class	
		Popular	*Not Popular*
Actual Class	*Popular*	0	245
	Not Popular	0	1475

4.3 Discussion

Overall, correctly classifying a Tweet as popular or not popular was dependent on both the dataset and prediction features chosen.

First we will discuss the data collection and resulting Twitter datasets. There are over 320 million users on Twitter. Streaming all Twitter data, even when just searching on popular hashtags, is certain to output more Tweets with low Retweet/Favorite counts than high ones. This affects our training data, as our model is being trained on a lot more unpopular Tweets than popular ones. This is clearly seen in our confusion matrices. Looking at the last confusion matrix for top 10 trending hashtags, for example, we see that even though there is a CDR of 85.76% no Tweets were classified as popular. Ideally we would have closer to a 50/50 split between unpopular and popular data in our training set. The only way to get this would be to get a lot more Twitter data and randomly select popular and unpopular Tweets until that ratio is met.

Now we will analyze the effect of features on classification. Overall, using all features to predict popularity resulted in a high Correct Detection Rate (CDR) of 95.45%. However, it also resulted in a high number of popular Tweets being misclassified as unpopular. In order to try and bring the Error Detection Rate (EDR) down and prevent overtraining, we aimed to reduce the number of features.

First we divided features into 2 categories: those that have counts as values and those that have a binary value. Using the hashtag dataset as an example dataset, we found the CDRs when using the 2 sets of features separately. SFS was applied to each set of features, and the best group of features was used to train the SVM classifier. Using only the best count features resulted in a CDR of 94.65%, and using only the best binary features resulted in a CDR of 77.85%. This shows that in general binary attributes aren't a good predictor of Tweet popularity, while count features are.

Next, the best binary and count features were combined into one feature set, and SFS was run on that set. Three features were chosen from the set: 2 count features (the combined URL and media count and the number of favorites a Tweet has after an hour) and 1 binary feature (whether a Tweet quotes another Tweet or not). Just using these features resulted in the highest CDR, 97.38%. This means that if you know the number of URLs in a Tweet, the favorite count of a Tweet after an hour, and if a Tweet quoted another Tweet, then you have ~97% chance of correctly predicting if a Tweet will be popular or not.

The last experiment run tested to see whether knowing if one of the top 10 trending hashtags was in a Tweet could help predict Tweet popularity. Running the SVM classifier on those binary attributes resulted in a CDR of 85.76%. However, upon further examination of the confusion matrix we see that no Tweets were predicted to be popular. This is probably because a large number of unpopular Tweets had trending hashtags in them. Therefore, we concluded that the presence of a trending hashtag in a Tweet is not a good predictor of Tweet popularity.

5. CONCLUSIONS

Our results led us to several important conclusions regarding which features are good predictors. First, features which have counts of objects are better predictors of popularity than features which are binary. Second, the best feature set for predicting if a Tweet is popular or not is the number of URLs in a Tweet, the favorite count of a Tweet after an hour, and if a Tweet quoted another Tweet. Third, the presence of a hashtag is not a good predictor of a Tweet's popularity.

The team has identified several potential improvements and further directions in which this research can be taken. In terms of improvements upon the current system, one of the areas in which we would most like to improve our analysis is in the use of the hashtags in the Tweets as attributes. Although this was not as important experimentally as we had hoped, we noticed during the hashtag analysis preprocessing that many of the popular hashtags within a single data streaming session were related via a common topic. This indicates that if we group the popular hashtags and identify whether a given Tweet is related to a particular popular topic, rather than identifying whether a given Tweet contains a specific hashtag, we may get getter prediction results from the attribute.

In addition, this project did not utilize any geographical information. This type of information could be incorporated in two ways. In the data collection stage, a list of comma-separated longitude, latitude pairs specifying a set of bounding boxes by which the retrieved Tweets can be filtered can be specified via the "locations" parameter. As a potential extension to the project, the team could use this ability to ask (and subsequently answer) the question of whether attributes contribute different weights to the prediction of popularity in different geographic locations. The team could also use the geographic location of a Tweet, reported in the "coordinates" field of a retrieved post, in order to ask whether the location of a Tweet influences the popularity. However, much more data collection time would be required for both of these tasks as many Tweets are not geo-tagged.

6. ACKNOWLEDGEMENTS

Special thanks to Dr. Pang-Ning Tan for providing documentation on how to interact with the Twitter APIs and a sample program on which both PHP programs in this project were based.

7. REFERENCES

[1] Apoorv Agarwal, Boyi Xie, Ilia Vovsha, Owen Rambow, and Rebecca Passonneau. 2011. Sentiment analysis of Twitter data. In *Proceedings of the Workshop on Languages in Social Media* (LSM '11). Association for Computational Linguistics, Stroudsburg, PA, USA, 30-38.

[2] Arias, Marta, Argimiro Arratia, and Ramon Xuriguera. "Forecasting with Twitter Data." *ACM Transactions on Intelligent Systems and Technology* (2012).

[3] Bao, Yanwei, Changqin Quan, Lijuan Wang, and Fuji Ren. "The Role of Pre-processing in Twitter Sentiment Analysis." *Intelligent Computing Methodologies*. 2014. 615-24.

[4] Null, Christopher. "Researcher Cracks the Code on What Makes a Tweet Popular." *PCWorld*. 31 May 2013.

[5] Ross, Arun. "Feature Selection." Michigan State University CSE 802: *Pattern Recognition and Analysis*. Lecture.

[6] Tan, Pang-Ning. "Data Collection." Michigan State University CSE 891: *Computation Techniques for Large-Scale Data Analysis*. Lecture.

[7] Zaman, Tauhid, Emily B. Fox, and Eric T. Bradlow. "A Bayesian Approach for Predicting the Popularity of Tweets." *Annals of Applied Statistics* 8.3 (2014): 1583-1611.

[8] Zaman, Tauhid R., Ralf Herbrich, Jurgen Van Gael, and David Stern. "Predicting Information Spreading in Twitter." *Computational Social Science and the Wisdom of Crowds Workshop* (2010)