Predicting the Odds of Getting Retweeted

Arun Mahendra
Stanford University
arunmahe@stanford.edu

1. Introduction

Millions of people tweet every day about almost any topic imaginable, but only a small percent of these get retweeted. People retweet for several reasons and psychology of why people retweet can be complex. In general, it is safe to assume that essentially people retweet when they find something interesting and they want to share it with others. The more a tweet gets retweeted the more people know about it. A tweet is more beneficial if it gets retweeted. Knowing the odds of getting retweeted based on the content has useful applications. For example, including the right words, rephrasing or including a relevant hyperlink in the tweet can get the attention of more people and thereby, increase the chances of getting the message passed on further. The goal of my project is to develop a machine learning algorithm to predict the likelihood of a tweet getting retweeted based on the content of the tweet. In this paper I evaluate the performance of supervised classifiers trained on topics derived using Latent Dirichlet Allocation (LDA) [3] model from tweets.

2. Acquiring and Processing Twitter Data

For this project, I downloaded data using Curl and the twitter streaming API. The data is continuously downloaded on to an amazon EC2 machine. One of the criteria for using the Twitter streaming API is that at least one keyword must be tracked and capture tweets containing those words only. In order to get a large set of diverse tweets, I began by tracking most commonly used words like “the”, “a”, “at”, “then”, “what”, “when” “RT” etc. For a targeted study, I also downloaded tweets made by a single user and his/her followers over the course of few weeks.

Once the data is downloaded, the tweet JSON objects are processed to remove fragmented data and extract desired attributes. The processed data is then moved in to a Hadoop-Hive data warehouse. Using Hive simplifies many file manipulation and organization tasks. Large text files can be easily manipulated and transformed simply by running SQL queries and this avoids the need to write custom scripts to traverse large text files to accomplish trivial tasks.

Preprocessing tweet text

1. Only tweets where the user language is set to English is used.
2. Common stop words are removed.
3. Non printable characters are removed.
4. Tweets are lemmatized.
5. Punctuations are used loosely in tweets when compared to other, more organized written texts, like in following example tweet: “I am soo cooooool!!!, please retweet!! ^^”. In the preprocessing all punctuations are removed.
6. Hyperlinks are replaced with TOKEN_URL
7. All text is converted to lower case

3. Method

Unlike large structured texts, twitter data poses unique challenges for developing machine learning algorithms. Tweets in general are very casual form of writing. Spelling errors are abundant in tweets; words are often arbitrarily stretched to emphasize a point of speech or abbreviated to fit into short sentences. As humans we can read and understand the tweets easily but for machine learning it makes the problem more challenging. Tweets make a lot of sense when put

into a context; some of these are global contexts which most people understand, while others make sense only to a select group of people.

In this project, I use LDA to generate topics from retweeted tweets. To illustrate discovering topics from tweets, consider the following set of tweets that was acquired from random users using the twitter streaming API on the night of the 2012 vice-presidential debate. LDA algorithm was run with each tweet as a document and the vocabulary used was created from the same set of documents. The following is the average inferred topic proportion for the retweet dataset.

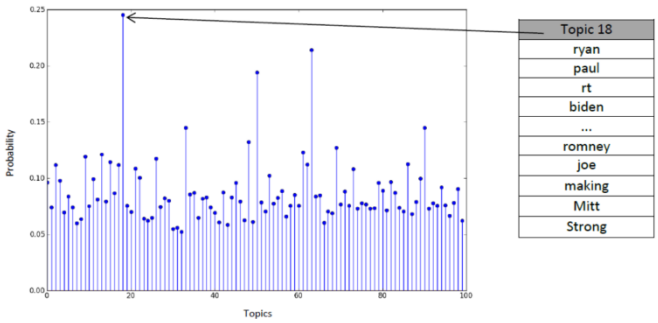


Figure 1 shows the topic distribution on arbitrary set of retweets from random set of users. For a single twitter user, a tweet must be first retweeted by the user's immediate followers before it can be retweeted by anyone else. By discovering and then learning topics that a user's followers often retweet, the algorithm can attempt to predict chances of a new tweet getting retweeted. Using twitter streaming API, only tweets (and retweets) sent out during that time can be captured, tweets sent out by a user previously cannot be acquired.

Twitter users are interconnected to their followers and friends. A user's tweets are received only by their immediate followers, the followers can further retweet them to their followers. A user's followers also receive tweets from other people they are following.

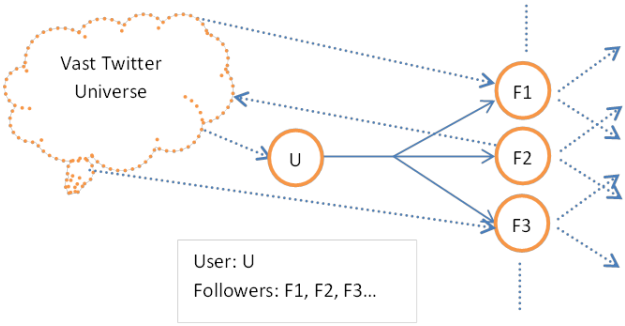


Figure 2: Interconnection of twitter users to their followers and friends

To predict the odds of a tweet getting retweeted by the followers, the learning algorithm should model the retweeting behavior of the followers. This can be accomplished using a supervised learning algorithm.

In order to train a supervised learning algorithm, the tweets must be labeled. I use the following simple metric to assign positive and negative classes to tweets in my dataset:

Let set of tweets $D = \{d_1, d_2, d_3 \ldots . d_m\}$ be the complete set of training examples.

Let the followers of user u be $F_u = \{f_1, f_2 f_3, \ldots f_n\}$

Let the number of times tweet d_i is retweeted by the followers F_u be = P_i

Let the life span of a tweet be

$$L_{d_i} = \max \text{time } d_i - \text{origin time } d_i$$

Then, retweet rate $R_{d_i} = \frac{P_i}{L_{d_i}}$

The retweet rate gives a simple metric to gauge the importance of a tweet to a set of followers. The labels are assigned to each tweet by using the distribution of this rate among a set of followers.

A positive label is given to tweets with high retweet rate. A negative label is given for tweets with a low retweet rate; this means that it has been retweeted only a few times in a long period of time.

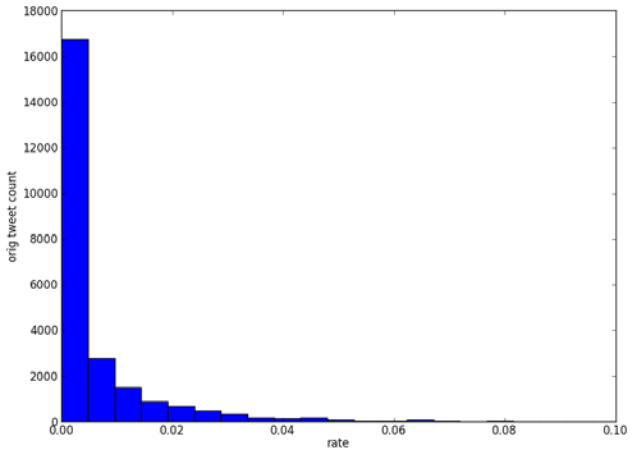


Figure 3: Tweets with retweet rate<0.01 are labeled as negative and tweets with rate>0.03 are labeled as positive

Positive label clearly indicates popularity whereas, in contrast, tweets labeled negative could mean either the users in set U did not want to retweet d_i or simply fewer users in set U actually received tweet d_i and thus resulting in low rate of retweet.

4. Feature Extraction and classification

There are several supervised text classification methods available in the literature such as [1], [2],[4],[6]. In this paper I evaluate two layered classification methods: LDA-Logistic regression and LDA-SVM ♠. I also propose a variation to LDA-SVM method to factor in the labels.

In LDA-SVM method, LDA model is used to generate topics from a text corpus and these topics are provided as input features to train SVM for classification.

To empirically evaluate these models, I use the test dataset containing 6,800 tweets, acquired and processed as described in the previous section. A balanced set (consisting of equal number of positive and negative examples) of M labeled tweets are randomly selected. Out of this, 80% of the data is used for training and 20% for testing.

For a K-topic LDA model, consider a tweet as document d_i in corpus $C = \{d_1, d_2, d_3 \ldots\ldots d_m\}$. The size of the total dataset is M. Vocabulary V is derived from the corpus C. The LDA model is trained on the corpus C and then the variational posterior Dirichlet parameters γ are computed for the training data set. γ_i for each document d_i is used as input features for training linear support vector machine and logistic regression model.

The following figures (4,5) show the performance of the SVM and logistic regression trained on the topics generated by the LDA on the labeled tweets respectively.

Figure 4

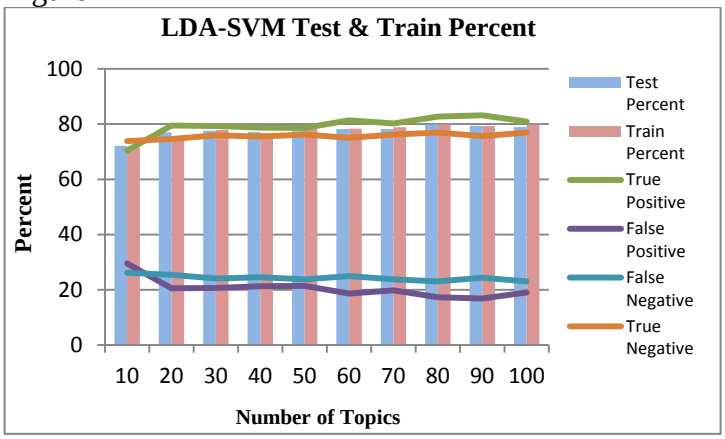
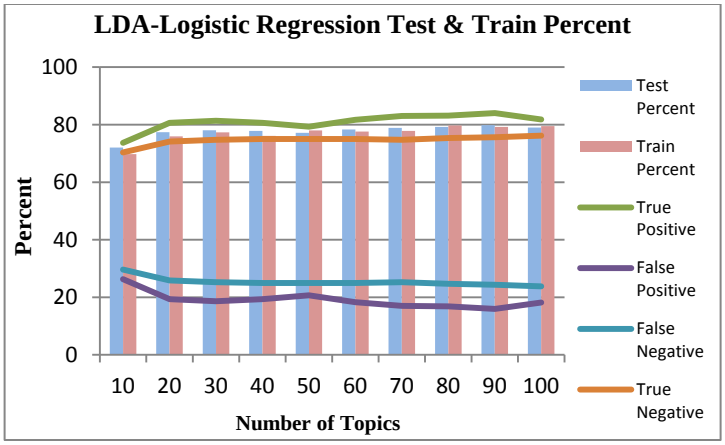


Figure 5



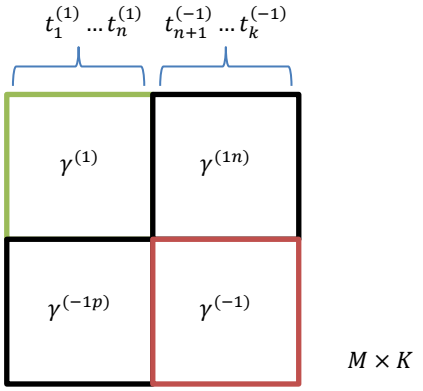
Number of training examples used: 6800

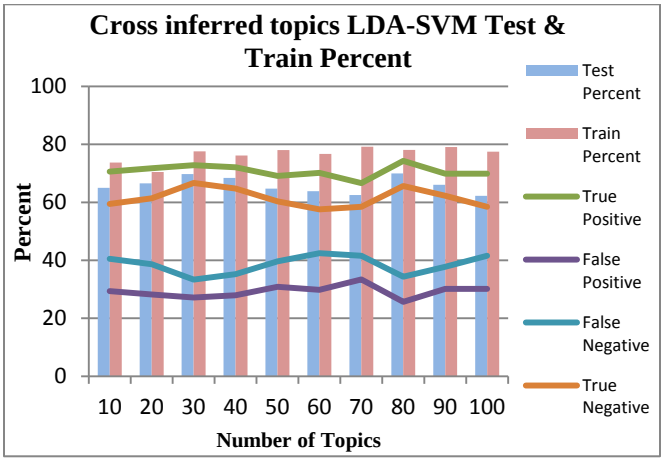
One of the drawbacks of this method, as noted in the literature [2] is that LDA is a generative model and the topics inferred using this unsupervised process may not be the ones that best represent the labels. This means that labels are not factored into the topic generation process.

I propose a variation to this two layered method in order to factor in the labels. For the case of binary classification, the idea is to train two LDA models, one on each corpus: negative corpus and positive corpus separately. Compute two sets of variational posterior Dirichlet parameters $\gamma^{(1)}, \gamma^{(-1)}$ for positive and negative documents respectively. Also, cross compute γ for positive training examples using LDA trained on negative set and vice versa. Let the resulting parameters be called $\gamma^{(1n)}, \gamma^{(-1p)}$. Combining the topics from $\gamma^{(1)}, \gamma^{(-1)}$, the following is defined:

Topic set T = $\{t_1^{(1)}, t_2^{(1)} \ldots \ldots t_n^{(1)}, t_{n+1}^{(-1)} \ldots \ldots t_k^{(-1)}\}$

Topics generated by LDA model using positive and negative datasets are represented by superscript (1) and (-1) respectively. M is the total number of training examples





5. Discussion and Conclusion

LDA-logistic regression and LDA-SVM performed roughly about the same. Their performance peaked at 100 topics with 79% accuracy. The cross inferred topic LDA-SVM performed worse than both these methods on all topic counts on the test sets. On the training set it fit the data almost as good and in some cases better than LDA-logistic and LDA-SVM. The better fitting of training data suggests that cross inferred topic LDA-SVM is over fitting the data and therefore, has high variance. It should be noted that in this study only a small number of tweets were used; cross inferred topic LDA-SVM could possibly perform better on a large training set, this can be explored in future work.

Given the nature of tweeting, a set of randomly acquired tweets can have high density of topics. Accuracy of labeling can have significant effect on the performance of supervised classifiers. The simple labeling metric used in this study can be improved. Twitter data comes with a variety of auxiliary data that includes various user specific information such as friends, follower count, profile description, personalized settings information, geo-location etc. Some of these can be used to create labels or even used as input features for classification.

In order to predict the odds of getting retweeted with greater accuracy, the next phase is to explore non-linear supervised classifiers. This work has laid foundation for future work.

♠ In this project, SVM and logistic regression was implemented in python using scikit-learn [6] and LDA algorithm using gensim [7]

6. References

[1] Perotte, Adler, et al. "Hierarchically supervised latent dirichlet allocation."Neural Information Processing Systems (to appear) (2011).

[2] Blei, David M., and Jon D. McAuliffe. "Supervised topic models." arXiv preprint arXiv:1003.0783 (2010).

[3] Blei, David M., Andrew Y. Ng, and Michael I. Jordan. "Latent dirichlet allocation." the Journal of machine Learning research 3 (2003): 993-1022.

[4] Quercia, Daniele, Harry Askham, and Jon Crowcroft. "TweetLDA: Supervised Topic Classification and Link Prediction in Twitter." Proceedings of the 4th ACM International Conference on Web Science (WebSci). 2012.

[5] Ramage, Daniel, Christopher D. Manning, and Susan Dumais. "Partially labeled topic models for interpretable text mining." Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining. ACM, 2011.

[6] Pedregosa, Fabian, et al. "Scikit-learn: Machine learning in Python." The Journal of Machine Learning Research 12 (2011).

[7] Řehůřek, Radim, and Petr Sojka. "Petr. Software Framework for Topic Modelling with Large Corpora." Proceedings of LREC 2010 workshop New Challenges for NLP Frameworks. 2010.