NLP_Update_Report

Will your blog-post appeal to the world? : An inference and estimation of popularity of a blog post based on analysis of its content

11/24/2012

Manisha Kamal - 108312148
Nilesh Poddar - 108393824
Purvesh Sahoo - 108231506

1. Problem Statement:

We intend to perform binary classification of blog-posts as to whether they are popular or not. The domain that we intend to target is category 'iphone' in reddit.com - subreddit /r/technology.

The motivation lies in the fact that this area has a lot of scope in success prediction but unfortunately, little research has been done in this area. Also, this kind of analysis will help a novice blogger to predict if his blog post will be popular or not. Finally, such analysis can also help in the marketing world, wherein new products are being launched every day, to decide if the product will be a success or not based on the information provided to the users.

The metric to compare against is the score of the post on reddit.com. In general, we sort the pages by 'top' and take the top 500 as the popular posts and bottom 500 as the unpopular ones.

2. Research and Data Collection:

Data collection can be divided in 2 stages:
1. Metadata collection: This involved parsing the json format of the pages, collecting the score, title, domain, url and other fields of the posts.

2. Time-periods:
We have collected data over a specific time period - sorted by week. We realized this resulted in lesser data than required, so switched to month and year. However, it also needs to be mentioned that though sorting by month and year gives us more data than by week, reddit still has the limitation of restricting the number of listings to 1000.

3. Data extraction:

a. Getting a list of blog posts in the subcategory 'iphone' and other domains as explained in the section 'experiments' by sorting it based on score (internally done by reddit when we sort by 'top')

b. Get the content of the top 500 and bottom 500 of the blog posts from the above list.

c. We used the reddit search API and the top API:

The scores and other parameters can be fetched in JSON format by using the APIs of reddit. The posts are sorted based on the score value by reddit internally when we choose sort by 'top'. Since we will get a list of the top posts, we are going to train our model based on the first 500 top posts as popular posts. We will categorize the posts based on this method. The reddit API to get the json for the top entries for a year has the url in the following format:
http://www.reddit.com/r/technology/top.json?sort=top&t=year
We parse the data fetched from the obtained json, and get a list of the top 500 blog posts and the bottom 500 blog posts. We have experimented with varying number of blog posts. Once we have the

blog links, we use the Goose content scraping library to scrape the content. We have also collected a few blog posts manually, if the content scraper failed to scrape the content.

4. Fold Generation:

Once the data was collected, we created the folds for our experiments. We had to make sure that the files in our folds have around the same average word length. This was done by taking an average of the word count in all the files of the collected blog posts. We select only those files for our training and testing data, which have word counts within a specified bracket of the calculated average word length. In doing this, we are ignoring certain files from the originally collected blog posts.

The way we train and test is as follows: Use 4 out of the 5 folds for training the model and then test it on the 5th fold. In the next iteration, we take another 4 folds and test it on the remaining fold. We have 5 such iterations, thus the name 5-fold.

3. Experiments & Results:

This section describes a variety of experiments conducted and why we finalized on the sub-domain "iphone". We list the experiments below in chronological order along with results.

1. **Technology Domain:**
   We initially experimented with the technology domain with 150 posts in the popular class and 150 in unpopular. This gave us ~50% accuracy on an average in n-grams and POS. We thought the reason for such less accuracy was sparsity in data. Following this experiment, we increased our dataset to 400 popular and 400 unpopular blog posts. Increasing data improved accuracy but not as much as we expected it to (average of ~54%). We then resorted to 2 plans - a) Switch to another domain and b) Continue with technology domain but with even more data. The former plan was to analyze if the domain played a role in the accuracy and the latter was to just put our previous intuition to use - more the data, more accurate the results. We increased the amount of tech data to 500 top and 500 bottom. The latter did not give much improvement in accuracy.

Technology (150 posts)

		Popular				Unpopular	
	Accuracy	**P**	**R**	**F**	**P**	**R**	**F**
Unigram	52.66	0.52	0.53	0.52	0.54	0.52	0.53
Bigram	53.84	0.32	0.33	0.32	0.33	0.32	0.33
Trigram	48.03	0.33	0.32	0.33	0.39	0.40	0.40
POS	49.62	0.49	0.73	0.58	0.526	0.258	0.323

Technology (500 posts)

		Popular				Unpopular	
	Accuracy	**P**	**R**	**F**	**P**	**R**	**F**
Unigram	54	0.55	0.57	0.54	0.54	0.51	0.51
Bigram	53.68	0.52	0.51	0.53	0.53	0.52	0.52
Trigram	51.132	0.5	0.49	0.51	0.49	0.50	0.49
POS	50.82	0.5056	0.533	0.5024	0.513	0.494	0.502

2. Different Domain - Politics:

We also analyzed /r/politics as the domain. As an observation from the results we got for the technology domain, we suspected that the POS models would tend to perform better in domains where the syntactic structure of the post would be more relevant. Politics is one such domain, where the syntactic structure of the post could play a part in the eventual popularity of the corresponding post. The posts from the technology domain, according to our analysis, rely more on the content of the post, than the lexical structure of the posts. Going ahead with this assumption, we extracted the top 500 and bottom 500 posts in the politics subreddit. But this experiment also did not give satisfactory results.

3. Drilling down to a particular sub-domain (iphone):

To further our experiments, we limited our domain within technology - to “iphone”. We believed our model was unable to learn much despite abundant data. The reason for this was the diversity of topics within the technology domain and thus we felt that going a few levels deep in the domain would help the model train better. We limited our domain to ‘iphone’ in

/r/technology. We found instant rise in accuracy for the n-gram models and POS. To further refine the accuracy or the model, we used probabilities instead of traditional TF-IDF values and also applied smoothing. We could see there was a drastic increase in accuracy for n-grams and POS. We could see an increase of around 6-7% in accuracy.

IPhone Subdomain

		Popular				Unpopular	
	Accuracy	**P**	**R**	**F**	**P**	**R**	**F**
Unigram	64.3	0.58	0.57	0.64	0.60	0.61	0.60
Bigram	63	0.62	0.77	0.66	0.54	0.52	0.53
Trigram	59.23	0.610	0.70	0.59	0.48	0.50	0.51
POS	48.23	0.47	0.68	0.55	0.46	0.27	0.33
POS + Bigram	57.76	0.5692	0.6035	0.589	0.59	0.55	0.56
POS + Trigram	57	0.55	0.64	0.60	0.59	0.50	0.53

The figures indicate averages.

5. Feature Vectors:

We have used the following feature vectors for our experiments:

- Unigram
- Bigram
- Trigram
- POS
- POS + Bigram
- POS + Trigram

The metric to measure each feature was the tf-idf value. We experimented with various metrics, such as using probabilities instead of tf-idf values, and also applying smoothing techniques. We also removed the stop-words from our data set. We observed that by using probabilities the results were better.

Titles as a feature:

We combined POS and "Title" of the post since the title of a post in certain domains is a major indicator of the popularity of the post. However, it gave us accuracies in the order of 50%. This can be attributed to the fact that popularity is also dependent on quantitative factors which are explained more in the 'Analysis' section below.

4. Analysis:

a. **Sparse data:** With our initial experiment on technology domain with 150 posts, we understood that the data was too less for the model to learn anything. This was proved when the model gave the same class as an output for all the test data. Thus, the classifier could not learn anything because the stop words were not removed and the data was sparse. Thus, the classifier got confused with the data available.

b. **Abundant but unclean data:** Even after increasing the data set to 500 posts for each class in the technology domain, we did not see much improvement in results (as seen from the figures). The problem here was the unevenness in the data size. We took all the posts irrespective of the amount of data it had. For eg, few posts had only around 15 - 20 words in it. Thus with such diverse data it was difficult for the classifier to learn anything with so less content.

c. **Restricted domain and clean data:** We cleaned the data by choosing only those posts which had enough content in them. Thus we learned that the amount of data did not matter much but the quality did. Initially we had 500 posts for each class but after all the filtering, we were left with ~200 posts in each class. Even with such less data, the classifier was able to learn and thus perform better because the data was clean and the diversity of data was reduced to a large extent.

d. **Syntactic Analysis:** With our initial few experiments and results, our intuition led us to conclusion that syntax or the structure does not matter for the domain that we are targeting. If the domain would have been Funny blog posts or Scary posts then the syntactic analysis would give better results for obvious reasons. But, when we restricted the domain and cleaned the data, to our surprise, syntactic analysis did give better results than before. Thus, the classifier is able to learn something about the structure also apart from the content and this leaves us with ample areas to explore for syntactic analysis. The classifier is able to learn some set of tags through which it is able to discriminate between popular and unpopular classes. Exploring what those tags are and other details is a part of our future plan.

e. **Quantitative measure:** Another major factor is quantitative analysis. Training the model quantitatively with respect to the number of likes, re-shares, tweets, google hits etc. will definitely improve the accuracy quite a lot since they are actual measures of popularity. But doing so will not analyze structure or content of the post and it just becomes a quantitative classification. Though they are good indicators, they can be combined with qualitative analysis but cannot be sole factors of judgment. Additionally, finding the number of hits to a particular page requires a good knowledge and use of Google APIs which is to be considered as a future scope to this project. However, quantitative means will not serve as a good indicator for unseen data - extrinsic evaluation of estimating how popular a post is going to

be since it does not have a measure already. It will however help in perfecting our model with seen posts which already have a measure to compare against.

5. Future plan:

a. Bag of words:

We plan to use this feature, to find out the top words in each class of data, i.e popular and unpopular blogs. This will help us understand the features which dominant and responsible for classification of the posts.

b. Normalized scoring:

Since we use the reddit scoring mechanism for a post as the basis of classification between the blog posts, we want to explore the options of having normalized scores across the posts. The scores would be normalized across the number of days the post has been active. Effectively, we would be penalizing a high scored post which has been live for a longer period. We might be able to use this score as a feature to train the model.

c. Analysis:

As mentioned in the analysis section, we believe that the content of the posts and the syntax, both are playing a part in deciding the popularity of the post. We plan to further our analysis as follows:

Content Analysis:

i. PCFG
ii. Bag of Words
iii. Lemmatization of the content
iv. Name-Entity relation

Syntactic Analysis:

i. Figure out the English tags that are occurring most frequently and relation between tags.
ii. Explore if the tense of the post creates a difference.

6. References:

1. http://www.cs.cornell.edu/~cristian/memorability_files/memorability.pdf
2. http://en.wikipedia.org/wiki/Blog
3. https://dl.acm.org/citation.cfm?id=1937087
4. www.fbe.hku.hk/~mchau/papers/Blog_WITS2009.pdf
5. https://github.com/nilesh0489/NLP_Project
6. https://github.com/jiminoc/goose