Capstone Report
Tweet Authorship

Vaishnav Vishal

March 30, 2017

1 Definition

1.1 Project Overview

This project is about tweet authorship. So, I'm going to take a dataset consisting of tweets from Hillary Clinton and President Donald Trump and would predict whether a given tweet is from Trump or Hillary by training a model on this dataset. We would do this using many techniques like supervised machine learning, NLP etc.

Supervised learning is the machine learning task in which the algorithms reason from externally supplied instances to produce general hypothesis, which then make predictions about future instances. It is the task of deriving a function from labeled training data.

Natural language processing (NLP) is a field of computer science, artificial intelligence, and computational linguistics concerned with the interactions between computers and human (natural) languages and, in particular, concerned with programming computers to fruitfully process large natural language corpora. Challenges in natural language processing frequently involve natural language understanding, natural language generation (frequently from formal, machine-readable logical forms), connecting language and machine perception, managing human-computer dialog systems, or some combination thereof.

Tweet encoding can be used to take data from tweets and use NLP on it to know about the behavior of the tweets to analyse similar kind of tweets. Twitter provides many APIs for downloading its tweets by users. Many people use these APIs to analyse the tweets realted to a particular topic.

1.2 Problem Statement

In this project, we will use supervised learning to train a model on a tweet dataset and then given a tweet, we have to predict who tweeted it.

- **Task:** Predicting the tweeter.
- **Performance:** Accuracy - No. of correct predictions.
- **Target function:** A function that gives weights to the terms in the tweets and then tells us who the author is.
- **Target function representation:** A Classification model.

Therefore, I seek to use tf-idf scores and a Naive Bayes classifier to predict whether a tweet is more likely to have been tweeted by @realDonaldTrump or @HillaryClinton.

1.3 Metrics

- **Prediction accuracy.** Accuracy is a common metric for binary classifiers; it takes into account both true positives and true negatives with equal weight.

  $accuracy = \frac{true\ positives + true\ negatives}{dataset\ size}$

  This metric was used when evaluating the classifier because false negatives and false positives both erode the user experience.

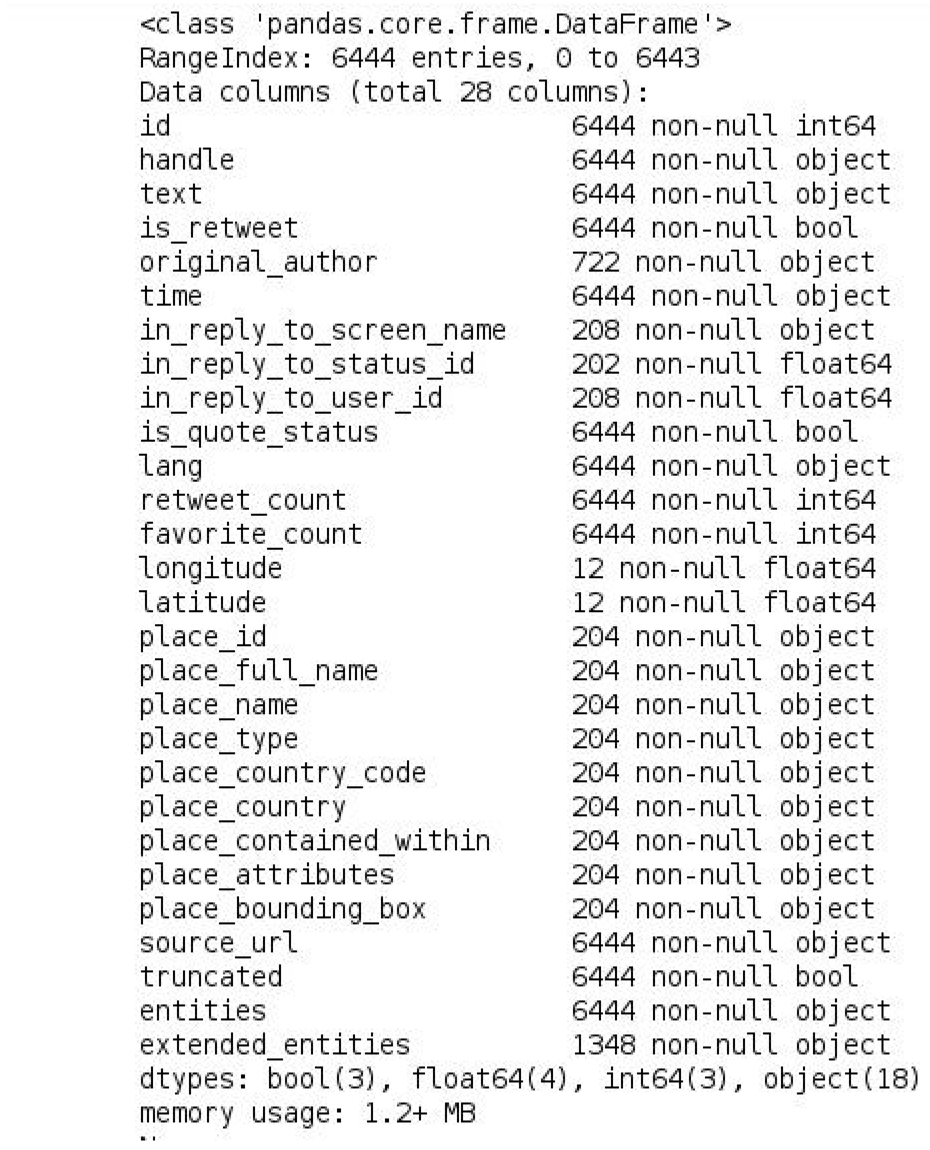
  As this is a classification type of problem, accuracy turns out to be the best evaluation metric to evaluate the performance of the model. I preferred accuracy over the other metrics such as recall, f1-score , precision because the data is well balanced as there are almost equal amount of data for both the classes.

  We can also use the confusion matrix as a good matrix because it also considers both the true positives and true negatives while evaluating the model.

2 Analysis

2.1 Data Exploration

The tweets data contains 6444 tweets from the handles @HillaryClinton and @realDonaldTrump



```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 6444 entries, 0 to 6443
Data columns (total 28 columns):
id                           6444 non-null int64
handle                       6444 non-null object
text                         6444 non-null object
is_retweet                   6444 non-null bool
original_author              722 non-null object
time                         6444 non-null object
in_reply_to_screen_name      208 non-null object
in_reply_to_status_id        202 non-null float64
in_reply_to_user_id          208 non-null float64
is_quote_status              6444 non-null bool
lang                         6444 non-null object
retweet_count                6444 non-null int64
favorite_count               6444 non-null int64
longitude                    12 non-null float64
latitude                     12 non-null float64
place_id                     204 non-null object
place_full_name              204 non-null object
place_name                   204 non-null object
place_type                   204 non-null object
place_country_code           204 non-null object
place_country                204 non-null object
place_contained_within       204 non-null object
place_attributes             204 non-null object
place_bounding_box           204 non-null object
source_url                   6444 non-null object
truncated                    6444 non-null bool
entities                     6444 non-null object
extended_entities            1348 non-null object
dtypes: bool(3), float64(4), int64(3), object(18)
memory usage: 1.2+ MB
```

Figure 1: Info

So, out of 6444 tweets , 3218 are tweeted by Donald Trump and 3226 are tweeted by Hillary Clinton which tells us that the data is not skewed towards any one class.

Figure 1 shows the info about the data.

So, out of all those features we select only 3 features which are 1. Handle, 2. Text and 3. Is Retweet to train our model.

Top Mentions: We then find out about the top mentions of both Hillary and Clinton. We come to know that Trump used mentions more than twice that of Hillary. Trump mentioned someone in 1964 of his tweets in comparision to 716 of Hillary.

Top Hashtags: Similarly, Trump was way ahead in terms of using hashtags than Clinton during their campaign. Trump used a hashtag 1425 times, whereas hillary used it merely 257 times.

2.2 Exploratory Visualization

The things we explored in the previous section can be visualized using the following figures.

2.2.1 Top Mentions

Top Mentions of Trump:

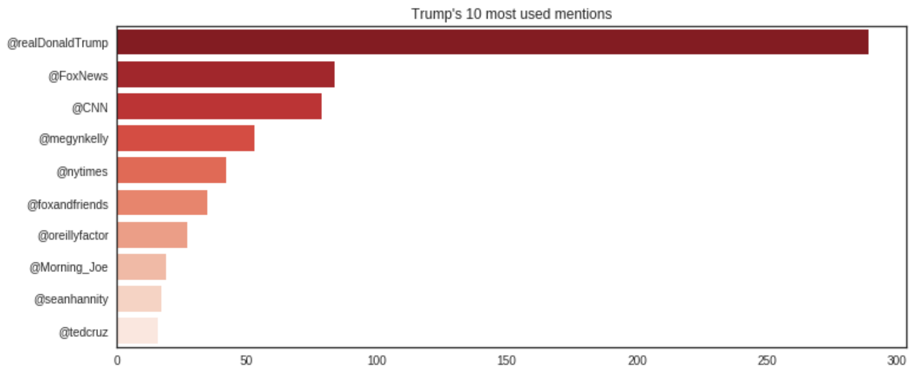


Figure 2: Top Mentions of Trump

Top Mentions of Clinton:

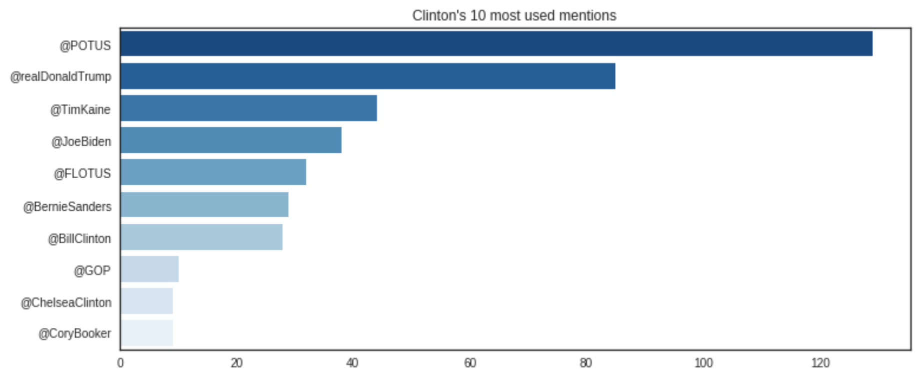


Figure 3: Top Mentions of Clinton

2.2.2 Top Hashtags

Top Hashtags of Trump:

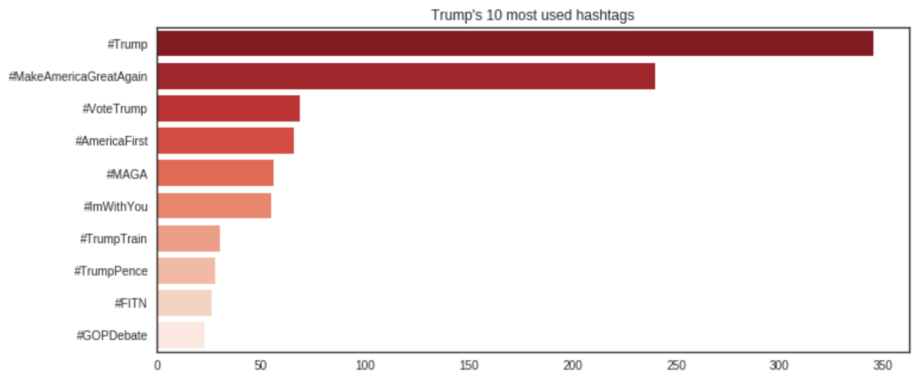


Figure 4: Top Hashtags of Trump

Top Hashtags of Clinton:

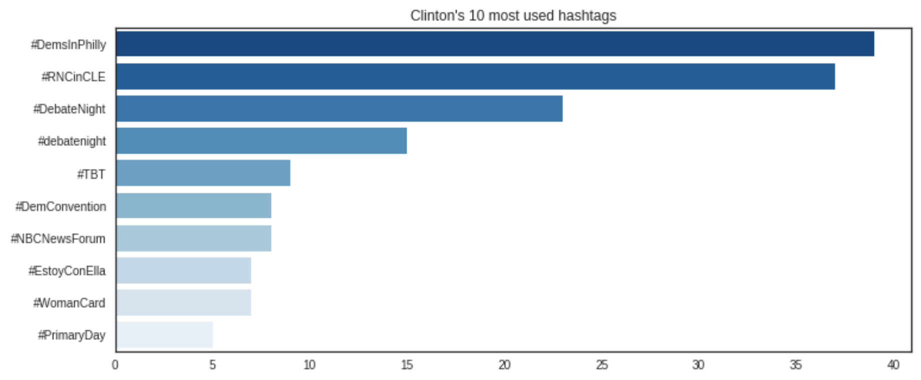


Figure 5: Top Hashtags of Clinton

2.3 Algorithms and Techniques

Algorithm: The Machine Learning algorithm that we will use is Multinomial Naive Bayes

With a multinomial event model, samples (feature vectors) represent the frequencies with which certain events have been generated by a multinomial $(p_1, \ldots, p_n)$ where p_i is the probability that event i occurs (or **K** such multinomials in the multiclass case). A feature vector $\mathbf{x} = (x_1, \ldots, x_n)$ is then a histogram, with x_i counting the number of times event i was observed in a particular instance. This is the event model typically used for document classification, with events representing the occurrence of a word in a single document (bag of words assumtion). The likelihood of observing a histogram **x** is given by:

$$p(\mathbf{x} \mid C_k) = \frac{(\sum_i x_i)!}{\prod_i x_i!} \prod_i {p_{ki}}^{x_i}$$

If a given class and feature value never occur together in the training data, then the frequency-based probability estimate will be zero. This is problematic because it will wipe out all information in the other probabilities when they are multiplied. Therefore, it is often desirable to incorporate a small-sample correction, called pseudocount, in all probability estimates such that no probability is ever set to be exactly zero. This way of regularizing naive Bayes is called Laplace smoothing when the pseudocount is one, and Lidstone smoothing in the general case.

Techniques: For encoding the tweets I will be using TFIDF Vectors.

In information retrieval, tf–idf, short for term frequency–inverse document frequency, is a numerical statistic that is intended to reflect how important a word is to a document in a collection or corpus It is often used as a weighting factor in information retrieval, text mining, and user modeling. The tf-idf value increases proportionally to the number of times a word appears in the document, but is offset by the frequency of the word in the corpus, which helps to adjust for the fact that some words appear more frequently in general. Nowadays, tf-idf is one of the most popular term-weighting schemes. For instance, 83% of text-based recommender systems in the domain of digital libraries use tf-idf.

So, we just create a bag of words using a count vectorizer and then form them into tf-idf vectors and then use the vectors to train the learning model to predict the author.

2.4 Benchmark

To create an initial benchmark for the the classifier, I used Gaussian Naive Bayes Classifier and trained the model.

For Accuracy, my benchmark turned out to be above 84.88% So, my final model must get atleast 85% accuracy to beat the benchmark.

3 Methodology

3.1 Data Preprocessing

No data preprocessing is required as there are no empty gaps which need to be filled in. We use the data from the dataset as it is.

3.2 Implementation

We read every tweet from the dataset and split them into tokens or lemmas.

After that we create a Bag of Words using the CountVectorizer method. We find that the vocabulary is of the size 9016.

We then feed this bag of words into a tf-idf transformer which converts the sparse matrix into tf-idf vectors. Then we fit our training data along with the labels into the Multinomial Naive Bayes model. It turns out that the accuracy on training set is 97

So our pipeline is :
1. Count Vectorizer 2. TF-IDF Transformer 3. Multinomial Naive Bayes

Now we split our data into 2 parts i.e, training and testing set with testing set being 20% of the total data. Then we again fit the training data into our model and test it on test data. The accuracy on test data turns out to be 93.66%.

3.3 Refinement

So I used GridSearchCV to tune the parameters of my model. So, the parameters that I wanted tune were tfidf__use_idf on values True or False and bow$_analyseronvaluessplitintolemmasandsplitintotokens.Ialsoused5foldcrossvalidationwhiletrainingtoimp$

As it turned out that the best accuracy was achieved when I created bag of words using lemmas while using inverse document frequency. The accuracy achieved was 93.66% on the test set. On the other hand if I used tokens to create bag of words without using inverse document frequency, the accuracy turned out to be 92.52%.

So if tfidf__use_idf was set to true and bow$_analyzerwassettosplitintolemmasaccuracywas93.664\%$

So if tfidf__use_idf was set to false and bow$_analyzerwassettosplitintotokensaccuracywas92.528\%$

So if tfidf__use_idf was set to true and bow$_analyzerwassettosplitintotokensaccuracywas93.445\%$

So if tfidf__use_idf was set to false and bow$_analyzerwassettosplitintolemmasaccuracywas92.593\%$

4 Results

4.1 Model Evaluation and Validation

The Final model achieves 93.66% accuracy on the test set. The parameters that we tuned were tfidf__use_idf and bow_analyzer.

Robustness of the model: I trained the model using a stratified K-fold technique to make sure that the model is robust enough on any real data. I created 5 fold cross validation to learn the right weights which will make sure that the model will predict really accurately.

4.2 Justification

The final model performs really well in comparision to the benchmark model. Our benchmark was to achieve atleast 85% and our model achieves nearly 95% accuracy with a standard deviation of 0.01. This clearly beats the benchmark model and is significant enough to solve the problem.

5 Conclusion

5.1 Free-Form Visualization

We extracted the terms with highest tf-idf scores to know about when the probability of a particular author is high and the results are as following.

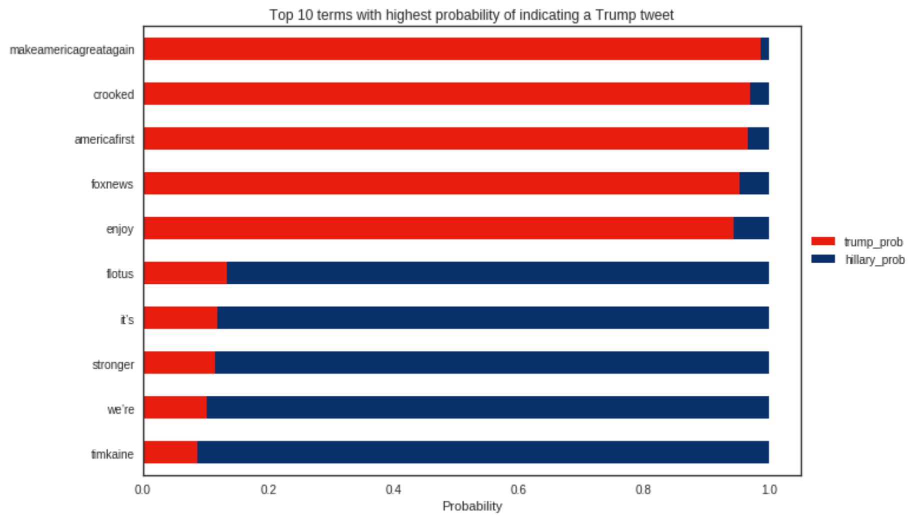


Figure 6: Top 10 terms with highest probability of tweeter being Donald

As we can see that Trump used the phrase make america great again as his motto, so if a tweet contains that phrase it is most likely that the tweet came from Trump. Similarly Hillary used the phrase or hashtag demsinphilly many times so if the phrase is a part of a tweet it is likely to be tweeted by Hillary.

Similarly, Figure 7 shows the terms with highest probability of tweeter being Hillary.

5.2 Reflection

I also tested this on many random tweets that contained the terms used by the candidates and the results were really good. The aspects that I found difficult in this project are 1. I didn't know how and frome where to start the project. So, I took time and then used regular expressions to extract mentions and hashtags of the candidates. 2. While tuning the model, it was already producing good results i.e, more than 90% accuracy. So, there was not much scope for improvement.

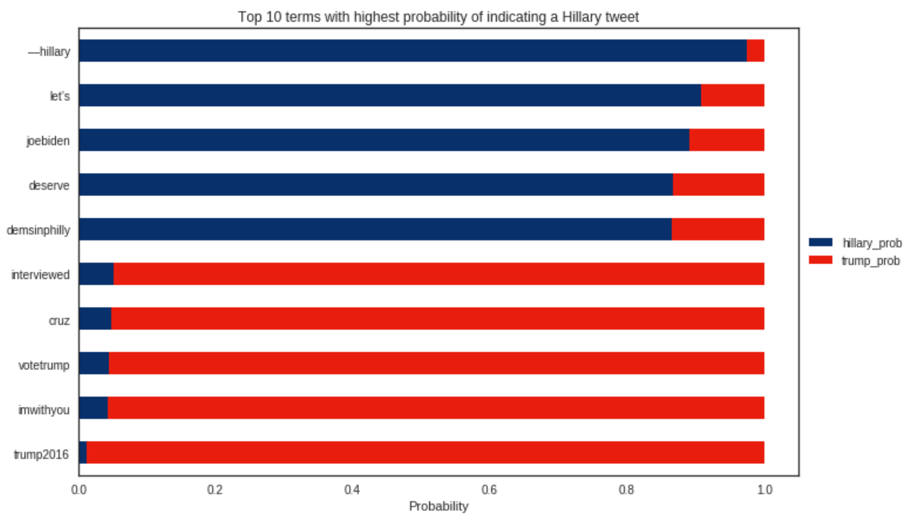


Figure 7: Top 10 terms with highest probability of tweeter being Hillary

5.3 Improvement

We can improve the model, by further improving the parameters like max_features in Tfidf Vectorization. We can also use word2vec instead of Tf-idf or along with tf-idf to improve the accuracy if the model. As it is already near 95%, there is not much scope for improvement.