CS 4780 - Kaggle Writeup

Team name: BabesOf4030
Public Leaderboard score: 0.82500
Team Members:

- Sena Katako: (senakatako)
- Sokhna Fall: (Sokhna Mously Fall)
- Arjun Chattoraj: (Arjun Chattoraj)
- Vegen Soopramanien: (Vegen Soopramanien)

1. Introduction

For the Final Project of CS 4780, we were tasked with developing a Machine Learning model that classifies tweets from @realDonaldTrump, the Twitter account of the 45th President of the United States. This was a binary classification problem with tweets from an Android phone and tweets from an iPhone.

2. The Data

Through Kaggle, we were provided with training and testing datasets. The training dataset had 1089 rows and 18 columns, while the testing had 16 columns. The extra columns in the training data were the obviously the label and the statusSource. The labels are 1 and -1, which correspond to Android and iPhone respectively.

3. Pre-processing

For fields in our data set that were not Boolean, we first looked at the unique values in those features and determined the distribution of the values.

3.1 Dropping columns

Certain variables present in the dataset were deemed useless for a prediction.

The **statusSource** variable was not present in the training dataset, and as such, became an obvious candidate of variables to be dropped. It actually gave away whether a tweet was from an Android or iPhone, and was understandably omitted from the testing data.

Other variables dropped were **replyToSID**, **latitude** and **longitude**. We noticed that only two rows in the training data in the case of latitude and longitude had information while **replyToSID** was all none.

We also dropped the **screenName** column since all the values in this column were @realDonaldTrump. Likewise, we dropped the **id.1** column which was a unique identifier for each tweet from the Twitter API.

3.2 Tokenizing Text

For each tweet, we first tokenize the tweet using the ***nltk*** package and take out all stop words from the tweets. Once tokenization is complete, we got the vector representation of each word from the ***genSim*** package and added the vectors for each tweet. We determine the vector representation of the tweet by dividing the summed up vectors by the length of the tokens. These features are used in building our model.

3.3 Feature Extraction

During our initial exploration of tweets, we noticed certain phrases that were more common with tweets from Trump. Some of these phrases include “Crooked Hillary” and “Thank you.” Hence, while looping through our tweets, we initialize a Boolean to capture if the key phrases pertinent to Trump are present.

Another feature extracted was has_link, which determines if there is a link included in the tweet. The reason is that Trump usually does not post links, but rather posts thoughts and reactions. As such, it seems likely that whenever there is a link, it was probably by some staff member in order to share events or news that could help Trump's image.

Another feature is has_I. As mentioned before, Trump posts reactions. There is a good chance that if the tweet is in the first-person, it is probably by Trump rather than a staff member.

By studying the time of the day Trump’s tweets were usually sent and just by general knowledge, we noticed that Trump posts tweets during the early hours of the day.Using this information, we extract the hour of the day the tweet was posted.

4. <u>Model Selection</u>

4.1 Validation split

We performed a simple train/test split in order to create a validation set so that we can determine the accuracy of our model before actually predicting on the test set. We used a 80:20 split and obtained 871 rows of data for our training set and 218 rows for our validation set.

4.2 Baseline

For our baseline model, we used the Naive Bayes algorithm. This gave us an average precision of 70%.

4.3 Random Forest

We chose to use the Random Forest classifier with 5000 trees and at a depth of 30 because it increases the chances of accurately predicting whether a tweet is by Trump or not Trump. We settled on a depth of 30 because that gave the best accuracy of the different parameters we tried. Using our random forest model and our features, we obtained an average precision of 86% on our validation set.

5. Packages used & citations

All packages used in our file are present in the 1st cell of the IPython Notebook. These are the ones used, with their link:

- Numpy: http://www.numpy.org/
- Matplotlib: https://matplotlib.org/
- Pandas: https://pandas.pydata.org/
- Scikit-learn: https://scikit-learn.org/
- Natural Language Toolkit: https://www.nltk.org/
- Gensim: https://radimrehurek.com/gensim/