Device Classification for Trump's Tweet

Team Name: **weak learner**

Team Members: Rakesh Gunukula (rg749), Ziqing Zhao (zz652), Sreenath Gopalakrishnan(sg2373), Akshay Agarwal (aa2657)

Score on Public Leaderboard: 0.83333

Problem Statement

The goal is to classify the device that Trump used to write each tweet with. We have been provided the tweets from President Trump's twitter from 2015 to 2016.

Data Preprocessing

We have access to President Trump's twitter data (content of the tweets) and associated data regarding context (retweet, time of the day etc.). To ensure the integrity and quality of data used for modeling following checks were performed:

Data Sanity checks:

Basic sanity checks were performed primarily on the fields – text, label, screen name, created and retweet count to ensure tweets were indeed from realDonaldTrump and that there aren't any missing data.

Tweet pre-processing and cleaning:

- Before converting the tweets (unstructured data) into structured feature tables, the following pre-processing steps were done/tested in model iterations.
- Removed stop words (NLTK stop words as reference)
- Removed special characters like exclamation marks, punctuation etc.
- Removed hyperlinks and the associated text
- Performed stemming and lemmatization

The above pre-processing steps were iteratively tested to optimize for model performance. Given that tweet is a short text with a strict (140 or 280) word limit, we hypothesize that punctuations and special characters contain much more information than regular words.

Learning and Hypotheses:

On exploring the data, we discerned few patterns in the training data which can be used for feature extraction and build the model.

1. Few columns such as “favorited”, “truncated” have exact same values for all the data and hence won’t affect the model
2. “Time of the day” plays a role in the output. We can see that Trump on the Android does a lot more tweeting in the morning, while the campaign posts from the iPhone more in the afternoon and early evening. Hence “time” can be used as a feature.
3. Another place we can spot a difference is in Trump’s behavior of “manually retweeting” people by copy-pasting their tweets and then surrounding them with quotation marks. Hence in the tweet quotation marks (“) can be a good feature vector.
4. Tweets from iPhone are generally about conveying some information and hence we can see that it mostly contains a link. Hence https:// can be used as a good feature vector.
5. iPhone tweets contain most of the hashtag (#).
6. Trump’s tweets are much more negative than the one from the white house. It contains a lot of negative words such as crooked Hillary, disgusting etc. Hence sentiment analysis can be a good way to distinguish between the tweets.

Feature Extraction:

Based on our hypotheses, we broadly looked at two sets of features –

Context-driven: Features that describe the context/nature of tweet.

Following extra features were extracted in order to test the following hypotheses –

1. Time of the day
2. Day of the week
3. Number of retweets
4. Length of the tweet

Content-driven: Features that describe the content of the tweet

An exhaustive list of words and symbols derived from the cleaned and tokenized twitter corpus forms the base feature (Count: #). TF-IDF works by penalizing the common words by assigning them lower weights while giving importance to words which are rare in the entire corpus but appear in good numbers in few documents.

1. **TFIDF computation:**

TF = (Number of times term t appears in a document)/(Number of terms in the document) and IDF = log(N/n), where, N is the number of documents and n is the number of documents a term t has appeared in.

TF-IDF = TF*IDF

TFIDF scores were computed using the TfidfVectorizer from sklearn.feature_extraction.text

2. **Word2Vec:**

Alternatively, the genism implementation of Google's word2vec was also tested to come up with features describing the content of the tweet.

Apart from the above list, features were created to test the following hypotheses –

1. Number of mentions present in the tweet
2. Presence of hyperlinks in the tweet
3. Presence of MAKE AMERICA GREAT AGAIN! #Trump2016, CrookedHillary etc.
4. The sentiment of the tweet

While the features in the final version were chosen based on the model performance on validation and test set, we tried shortlisting the features using PCA.

Model Selection

The training dataset was divided into training and validation datasets and accuracy was the metric used to compare the model results on validation dataset. Once feature selection was done, we started out by running the simplest of classification models, logistic regression. We did some oversampling/ undersampling to ensure the classes are balanced in the training dataset. This led to some improvement in accuracy in logistic regression. Later we tried fitting Ridge Classifier, Naïve Bayes and a few ensemble models like random forests, gradient boosting classifier, and AdaBoost classifier. Doing a 10-fold cross-validation and looking at the average accuracy on the validation datasets, we decided to go ahead with gradient boosting classifier with an exponential loss function as it results were better than the rest. All the implementations we tried were from the scikit-learn package in python.

Hyperparameter optimization

Now that we have decided which classifier we will be using, it was time for tuning the parameters to make the model more powerful. Again, we were looking at the results on validation datasets to do hyperparameter tuning. The weak learner in our case was a decision tree and we could have tweaked the number of trees, depth of each tree, minimum no. of samples in each leaf node and a fraction of samples available for each individual base learners to learn from, with replacement. Some of these parameters interact with others and we did a grid search to finalize the hyperparameters.