CS 4780: Machine Learning for Intellectual System

Final Project

Team Mushroom: Alex Wan (fw98), Xinquan (Crysta) Yang (xy368)

December 4, 2018

# 1 Introduction

This is the report on CS 4780 Final project. We got a dataset on Trump messages, and the goal is to identify which device he used for a given Twitter text. In general, we decomposed text messages to feature vectors and used random forest as classification algorithm. The best kaggle score is 82.5% accuracy.

# 2 Feature Engineering

We were given 16 training features, among which [favorited favoriteCount replyToSN truncated replyToSID id replyToUID statusSource screenName retweetCount isRetweet retweeted longitude latitude] are essentially meaningless in terms classification power as they stay the same. Therefore, we focus on three features: text, favoriteCount, retweetCount.

## 2.1 Text Feature: Bag of Words

It is really hard for a classification algorithm to process string sentence. Therefore, we use the established technique "bag of words" to transform a string input to a numeric vectors.

**Prepossess Data** Following established natural language processing methods, we decide to "clean" text messages by taking out two components: 1) punctuation, defined by built-in library *string.punctuation*); 2) stopwords, which are words that too frequently used to mean little in terms of classification, defined by *nltk.stopwords*.

To accomplish the goal above, We built a text_process function from scratch.This function will take a string input and take out the two components defined above. The left word tokens will be returned as a list. We then apply this function to all text messages.

Note that we made models both with and without the method, and compare the result later on.

**Construction** We construct a bag of words vector for each question. To do that, we use function *CountVectorizer* from online library *sklearn.feature_extraction.text*. Essentially, we run through all the text messages in training and testing set, summarizing all the unique words. These words are columns. After that we run through all the text messages again. This time, for each text, we initialize a zero vector same size as the unique vocabulary size, and update the word count of text on corresponding entry.

**Example (without preprocessing)** text 1 = "We have a really fun semester taking ML", text 2 = "We have fun in ML". There are 9 unique words: We, have, a, really, fun, semester, taking, ML, in. Therefore, each bag-of-word vector has 9 dim, each entry corresponding to each unique word type. bow 1 = [1, 1, 1, 1, 1, 1, 1, 1, 0], and note that we have 0 as the last entry since we don't have have 'in' in text 1. Similarly, bow 2 = [1, 1, 0, 0, 1, 0, 0, 1, 1].

We use the result sparse matrix as our input feature matrix.

# 3 Model

## 3.1 Baseline on Dev Set

This problem is a binary classification question. Therefore, we tried the following three binary classification algorithm on development set, and choose the best one to try on test set.

**Construct Development Set** We simply used *train_test_split* from online library *sklearn.model_selection* to split the training data set to "training" and "development set".

**Result**

| | **Naive Bayes** | **SVM** | **Random Forest** |
|---|---|---|---|
| **Accuracy** | 0.8534 | 0.6227 | 0.8461 |

Clearly, Naive Bayes and Random Forest outperform SVM. Because there score aren't that different, we then move on test set with random forest due to its time efficiency.

## 3.2 Random Forest on Test Set

**1) Hyper-parameter Tuning** We used function *GridSearchCV* from online library *sklearn.model_selection* to find the best hyper-parameter. We define *parameter_grid* as {'n_estimators': [100, 500, 1000], 'max_features': ('sqrt', 'log2', None)}. The best parameter was found to be n_estimators = 500, max_features = 'sqrt'. The best parameters were found to be: n_estimators = 1000, max_features = sqrt.
**2) Result: Kaggle Evaluation Score** We used *RandomForestClassifier* from library *sklearn.ensemble*. We feed in the bag-of-words matrix on test set as input matrix. The parameters for random forest are set to be n_estimators = 1000, max_features = sqrt. The predictions generated was then submitted on Kaggle.


# Citation

Scikit-learn: Machine Learning in Python, Pedregosa et al., JMLR 12, pp. 2825-2830, 2011.