**Fall 2019 Reviewer Model Report**

**Introduction to Problem:**

For this semester’s CS 4000 project, my group and I were tasked with building and training a machine learning model which could take a review, or set of reviews, and give a prediction of whether the review was real or fake. The first step, before thinking of building a model, was to gather a reliable dataset. After, we would decide the tools and methods which we could use for building a model. Finally, we divided multiple machine learning techniques among the group and begin applying them. Specifically, I tested Random Forest, Logistic Regression and Support Vector Machine classification techniques (K-Means clustering was also experimented with).

**Collection the Dataset:**

Regarding the dataset, it was decided that in order to receive as much useful experience as possible from the course, and to also gather a reliable dataset, that we would be collecting the reviews ourselves. Also, it was difficult to find a dataset for our model because of its nature. I will go into more depth of why when discussing the classifying algorithm’s results. For our dataset, we decided to gather reviews from several local Indian restaurants in Lubbock. The restaurants were Tikka Shack, Royal Indian Cuisine and India Palace Indian Restaurant.

Starting with the fake reviews, searching for them among the reviews of the restaurants would be extremely difficult. Instead, we wrote our own. In order to do so we each individually researched fake reviews and their characteristics. Many sights had examples of fake reviews and their subtle features, which made it helpful to craft our fake reviews without making them too obviously fake. Some example sites which I used for crafting my fake restaurant reviews are listed here:

* <https://www.marketwatch.com/story/10-secrets-to-uncovering-which-online-reviews-are-fake-2018-09-21>
* <https://www.inc.com/jessica-stillman/heres-how-to-spot-fake-online-reviews-with-90-perc.html>

After making our hand-written fake reviews, we collected our real reviews from the internet. The problem with this approach was how could we tell which reviews were real? After some brief research, we found that it was easier to spot a real review than fake. Certain websites were more credible than others (such as yelp vs google reviews) and some people were more credible than others (such as a person who’s written 200 reviews and has a full profile vs a reviewer with only one to two reviews).

**Feature Analysis:**

After gathering the reviews, and before building my model, I read through the reviews of my group members and made observations on the characteristics of a real or fake review. Since we wouldn’t be able to tell the correlations the model would find between the words/features in the reviews, we wished to find our own features before-hand. Of course, there could be bias because I had previously looked for features of real and fake reviews when doing data collection, but I tried to make my observations as unbiased as possible.

**Possible Real Features:**

* More specific to the reviewer’s experience than fake review
* When the reviewer can’t recall all the details of the meal, this seems like it is a trend with authenticity (such as the user can remember the name of an exact dish or appetizer)
* Has a mix of good, bad and okay comments about the restaurant

**Possible Fake Features:**

* Very broad with little to no specifics
* Overly positive or negative
  + - Too many positive or negative things to say about the restaurant
* Many grammatical errors
* Very bland with negative reviews (it seems when a customer is very angry, they will document their experience more)
* Too much scene-setting or irrelevant detail

**Clustering**

Before testing classification algorithms on our dataset, I wanted to see how the data could be clustered. In other words, correlations between words would be clustered into two categories, as if we were mocking a classification algorithm.

To convert the reviews to a matrix of features, TfidiVectorizer from the scikit-learn library was used. For the clustering algorithm, I used K-Means algorithm. First, I decided the take 10 words which were considered features of each cluster from the model. Then I displayed the accuracy, f1, recall and precision scores. The results were as follows:

**10 words from the second cluster:**

food

chicken

naan

place

like

restaurant

butter

garlic

really

sauce

**10 words from the first cluster:**

indian

food

place

really

restaurant

good

authentic

friends

lubbock

experience

**Three Testing of K-Means Algorithm:**

*Accuracy Rate using K-Means Algorithm*

*Testing Accuracy: 0.6538461538461539*

*Testing F1 score: 0.6523076923076924*

*Testing Recall score: 0.5714285714285714*

*Testing Precision score: 0.7272727272727273*

*﻿ ﻿Accuracy Rate using K-Means Algorithm*

*Testing Accuracy: 0.8076923076923077*

*Testing F1 score: 0.8068376068376069*

*Testing Recall score: 0.7142857142857143*

*Testing Precision score: 0.9090909090909091*

*﻿Accuracy Rate using K-Means Algorithm*

*Testing Accuracy: 0.38461538461538464*

*Testing F1 score: 0.38461538461538464*

*Testing Recall score: 0.35714285714285715*

*Testing Precision score: 0.4166666666666667*

As we can see, the correlations found have no relation to real and fake reviews. The cluster could represent a different correlation, such as positive and negative reviews for example, but it is difficult to identify the correlation found.

**Classification**

For my classification algorithm, I decided to use Doc2Vec from the Gensim library to turn our reviews into number vectors. To clean and tokenize the reviews and the words in the review, before converting to a number vector, I used the Natural Language Toolkit Library. Then for the training, testing, and prediction of the data, scikit-learn was utilized.

*Note, that even though the dataset included additional features, my model was trained with purely the text reviews and real/fake labels (1 for real, 0 for fake).*

Before talking about the results of the process, it is important to mention that the dataset was fairly small. The final dataset totaled around 86 reviews. This led to some inconsistencies in results when the program would be run, and predictions would be made. However, the general ranges will be showed for each method. Also, for simplicity I will use the testing accuracy as my only qualifier for how a model performed, but the F1, recall, and precision score can also be seen with the printed results.

The first algorithm tried on the vector was Support Vector Machine (SVM). The lower accuracies were around 60%. The higher ones around 77%. On average, after about 10 runs of the algorithm, I calculated the average at 73%.

***Higher Scores Average:***

*Accuracy Rate using* ***Support Vector Machine***

***Testing Accuracy: 0.7692307692307693***

*Testing F1 score: 0.7511312217194571*

*Testing Recall score: 1.0*

*Testing Precision score: 0.7*

*﻿*

***Lower Scores Average:***

*Accuracy Rate using* ***Support Vector Machine***

***Testing Accuracy: 0.6153846153846154***

*Testing F1 score: 0.6009615384615385*

*Testing Recall score: 0.7857142857142857*

*Testing Precision score: 0.6111111111111112*

***Average of Average Scores:***

*Accuracy Rate using* ***Support Vector Machine***

***Testing Accuracy: 0.7307692307692307***

*Testing F1 score: 0.702262443438914*

*Testing Recall score: 1.0*

*Testing Precision score: 0.6666666666666666*

The second algorithm tried on the vector was Logistic Regression. The lowest accuracies were found to be around 60%. The higher were around 80-84%. On average I calculated the average at 77%.

***Higher Scores Average:***

*﻿ ﻿Accuracy Rate using* ***Logistic Regression***

***Testing Accuracy: 0.8461538461538461***

*Testing F1 score: 0.8461538461538461*

*Testing Recall score: 0.8571428571428571*

*Testing Precision score: 0.8571428571428571*

***Lower Scores Average:***

*﻿Accuracy Rate using* ***Logistic Regression***

***Testing Accuracy: 0.6923076923076923***

*Testing F1 score: 0.6923076923076923*

*Testing Recall score: 0.6428571428571429*

*Testing Precision score: 0.75*

***Average of Average Scores:***

*﻿Accuracy Rate using* ***Logistic Regression***

***Testing Accuracy: 0.7692307692307693***

*Testing F1 score: 0.7664335664335664*

*Testing Recall score: 0.8571428571428571*

*Testing Precision score: 0.75*

The final classification algorithm tried on the vector was Random Forest Classification. The lower accuracies were around 60%. The higher ones were around 80%. On average I calculated the average at 73%.

***Higher Scores Average:***

*﻿Accuracy Rate using* ***Random Forest Classifier***

***Testing Accuracy: 0.8076923076923077***

*Testing F1 score: 0.8032612548741582*

*Testing Recall score: 0.9285714285714286*

*Testing Precision score: 0.7647058823529411*

***Lower Scores Average:***

*﻿Accuracy Rate using* ***Random Forest Classifier***

***Testing Accuracy: 0.5769230769230769***

*Testing F1 score: 0.5321266968325792*

*Testing Recall score: 0.8571428571428571*

*Testing Precision score: 0.5714285714285714*

***Average of Average Scores:***

*﻿Accuracy Rate using* ***Random Forest Classifier***

***Testing Accuracy: 0.7307692307692307***

*Testing F1 score: 0.7245657568238213*

*Testing Recall score: 0.8571428571428571*

*Testing Precision score: 0.7058823529411765*

With Logistic Regression averaging around 77%, and Support Vector Machine/Random Forest Classification averaging around 73%. It seems Logistic Regression was the best of the three classification algorithms I applied to the dataset. However, in many cases SVM outperformed logistic regression by a good margin, while Logistic Regression had outperformed SVM in just the previous run of the program. What does this mean? Well as previously mentioned the model was trained with only 60 reviews (70% of the 86 total reviews that our dataset contained). I believe with a larger set of reviews, with real and fake labels, the results would be clearer and more consistent with less varying accuracies. Especially given the complexity of text data and NLP. The only difficulty in finding or gathering a dataset larger than our own, is that when the reviews are being gathered, it is very hard to verify a review as either real or fake. The collection of such a large dataset would be complex and costly in time. However, with the dataset we gathered, it is clear that the model definitely found correlations within the text of the reviews to classify them as real or fake with relatively good accuracy.