Jade Selke

Yelp Review Analysis Findings

 Within this project, I intend to answer: Is it possible to analyze reviews from Yelp to determine the sentiment of the review, and to identify fraudulent reviews, based on sentiment analysis, natural language processing, and n-gram processing of the reviews using logistic regression?

**Hypothesis Testing**

 For this project, the following hypotheses are being tested:

Logistic Regression of Reviews for Sentiment

Null Hypothesis: Sentiment of a review cannot be determined from sentiment analysis, natural language processing, and n-gram processing of the reviews using logistic regression.

Alternative Hypothesis: Sentiment of a review can be determined from sentiment analysis, natural language processing, and n-gram processing of the reviews using logistic regression.

Logistic Regression and K-Nearest Neighbors of Reviews for Fraud

Null Hypothesis: Fraudulent reviews and reviewers cannot be identified from sentiment analysis, natural language processing, and n-gram processing of the reviews using logistic regression or k-nearest neighbors algorithms.

Alternative Hypothesis: Fraudulent reviews and reviewers can be identified from sentiment analysis, natural language processing, and n-gram processing of the reviews using logistic regression or k-nearest neighbors algorithms.

**Modelling Set Creation**

 In generating the results of this project, a set of New York City restaurant reviews taken from Yelp (referenced as YelpNYC) are analyzed and further broken into a specific modelling set. This sample set contains an equal number of samples of negative real reviews, positive real reviews, negative fraudulent reviews, and positive fraudulent reviews taken at random from the full YelpNYC set. This new set was created in Python programming language version 3.8.5, and all further analysis is done in Python as well (Python Software Foundation, https://www.python.org/). The full YelpNYC data set contained 359,052 reviews, and 5,958 of those reviews were both negative and fraudulent. This is only 1.7% of the total number of reviews. If analysis was done against the full set, any algorithm that guessed a review was never fraudulent when the review was negative would be correct most of the time. Because this is the smallest sample of the data specifically being checked for, the algorithms use equal random samples of the other review types: positive and real, positive and fraudulent, and negative and real. Based on that, the resample tool from pandas version 1.3.3 was used to generate random sample sets of the other three categories with a sample size of 5,958 (McKinney, 2010). This brings the total modelling set size to 23,832 reviews.

**Modelling Set Preprocessing**

 To build the training and test sets, the modelling set was divided into two data frames, one containing the independent variables, the Review string data and LemmaSummary string data, and another containing the dependent variables, the flags for FakeReview and Sentiment. The numpy library version 1.21.2 is then used to randomly select twenty percent of the population for testing and eighty percent for training (Harris, et al., 2020).

 The LemmaSummary string data is created using removal of stop words and punctuation followed by casing all words to lower case and tagging each word with its part of speech. Next, each string is parsed through a vectorizer to build a library of bigrams, trigrams, and tetragrams used throughout the modelling set of reviews that occur more than five times. This creates a total of 10,241 vectors, which are used as the coefficients for the logistic regression to build from. After the LemmaSummary string set is built and empty sets removed, the final modelling data set size used had a population of 23,832 reviews, evenly distributed, with 18,637 in the training set and 5,195 in the test set.

**Logistic Regression and Model Building**

 Two logistic regression models are built from the training set. The first logistic regression was built to find sentiment predictions based on the review and lemma summary of a user. It was built using a lbfgs algorithm, both because it is the default in the scikit model version 0.23, and due to its optimization in speed (Pedragosa et al., 2011). This algorithm is made to converge with a low number of iterations for smaller data sets. Sag, saga, liblinear, and newton-cg algorithms were also tested with this data set but were found to be slower in computation time, or not as accurate. The number of iterations needed to converge with the lbfgs algorithm was increased from the default of 100 to 1,000.

 The second logistic regression was built to find fraud review predictions based on the sentiment, review, and lemma summary of a user. Here, the SentimentRating is considered an independent variable instead of dependent. The saga algorithm was found to have the highest accuracy in building this model and required 5,000 iterations to reach convergence.

**Model and Results Analysis for Sentiment Predictor Logistic Regression**

 For the sentiment prediction logistic regression, overall accuracy of predicting sentiment based on the review provided by a user was calculated at 92.5% ± 1% and can be seen in Table 1. The software used does not calculate p-values for a logistic regression but using the computations at GraphPad (2021) show a p-value for this accuracy to be roughly 0.0425.

**Table 1**, Classification Report for Sentiment Predictor

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Precision | Recall | F1-Score | Support |
| Negative Sentiment | 0.91 | 0.94 | 0.92 | 2,533 |
| Positive Sentiment | 0.94 | 0.91 | 0.93 | 2,662 |
| Accuracy |  |  | 0.92 | 5,195 |
| Macro Average | 0.92 | 0.93 | 0.92 | 5,195 |
| Weighted Average | 0.93 | 0.91 | 0.91 | 5,195 |

 In using the sentiment predictor model against the test set, generating a full logistic regression equation would be extensive, given 10,739 coefficients used in the vectorizer of the LemmaSummary. However, they are sorted from lowest to highest to give an idea of how the coefficients are used in the sentiment predictor in Table 2 and is created from modelling against the test set. The intercept for the predictor is 0.4225. It is worth noting in this set of coefficients, that the only bigrams used are the positive n-grams and there were no trigrams or tetragrams in the top ten. Also, negative n-grams have a much stronger effect on the overall prediction value than positive n-grams; the highest negative coefficient is 68% higher in magnitude than the strongest positive n-gram.

**Table 2**, Highest and Lowest Coefficients and Associated N-grams in Test Set

|  |  |  |  |
| --- | --- | --- | --- |
| Positive N-gram | Positive Coefficient | Negative N-gram | Negative Coefficient |
| delicious | 1.527 | overcook | -1.866 |
| incredible | 1.529 | disappointment | -1.876 |
| amaze | 1.556 | horrible | -2.029 |
| fantastic | 1.603 | overprice | -2.092 |
| excellent | 1.677 | meh | -2.095 |
| perfect | 1.693 | terrible | -2.320 |
| awesome | 1.693 | bland | -2.375 |
| favorite | 1.740 | mediocre | -2.795 |
| heaven | 1.849 | tasteless | -2.846 |
| well worth | 1.934 | overrate | -2.850 |

 A confusion matrix was generated from the results of using the sentiment predictor against the test set and the confusion matrix result is shown in Figure 1 below. This shows that the logistic regression can accurately predict the sentiment of a review as negative when the review is negative 2,375 times out of 2,607 cases; and predict a positive sentiment when a review is positive 2,430 times out of 2,588 events. In the case of a Type I error, in which the null hypothesis is rejected, or a review is predicted to have positive sentiment when it has a negative sentiment instead of being predicted correctly, happens 232 out of 4,963 events in the test set, or 4.67% of the time. A Type II error, in which a negative sentiment is predicted when a review is positive instead of being correctly predicted happens 158 out of 4,963 events in the test set, or 3.18% of the time.



**Figure 1**, Confusion Matrix for Sentiment Predictor against Test Set

 Finally, an analysis of the precision-recall curve for the test set was done to show the tradeoff between the true positive rate and false positive rate for the sentiment predictor (Brownlee, 2021). This is how the accuracy of 92.5% is generated and helps to show how much more accurate the sentiment predictor is than a no-skill classifier would be at predicting, such as a coinflip. The further the model bows away from the diagonal dotted line, the more accurate or precise the model. The precision-recall curve for the sentiment prediction logistic regression model can be seen in Figure 2. This model has a high degree of accuracy in predicting true false positive or predicting a sentiment is positive.



**Figure 2**, Precision-Recall Curve for Sentiment Predictor

**Model and Results Analysis for Fraud Review Predictor Logistic Regression**

 For the fraud review prediction logistic regression, overall accuracy of predicting fraud based on the review provided by a user was calculated at 59% and can be seen in Table 3. The software used does not calculate p-values for a logistic regression but using the calculators at GraphPad show a p-value for this accuracy to be roughly 0.59.

**Table 3**, Classification Report for Fraud Review Predictor

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Precision | Recall | F1-Score | Support |
| Fraudulent Review | 0.63 | 0.58 | 0.60 | 2,783 |
| Real Review | 0.57 | 0.59 | 0.58 | 2,412 |
| Accuracy |  |  | 0.59 | 5,195 |
| Macro Average | 0.59 | 0.60 | 0.59 | 5,195 |
| Weighted Average | 0.60 | 0.59 | 0.59 | 5,195 |

In using the fraud review predictor model against the test set, the highest and lowest n-gram sets and their coefficients are shown in Table 4. The intercept for the predictor is -0.648. It is worth noting in this set that many more n-grams are used than previously seen, as well as references to the food, such as really sweet and thicker. The scale of the coefficients for the real reviews vs the fraudulent reviews is much closer than seen in the sentiment predictor.

**Table 4**, Highest and Lowest Coefficients and Associated N-grams

|  |  |  |  |
| --- | --- | --- | --- |
| Real Review N-gram | Real Review Coefficient | Fraud Review N-gram | Fraud Review Coefficient |
| restaurant time | 1.536 | amateur | -1.524 |
| friend place | 1.542 | kind service | -1.536 |
| ready wait | 1.543 | anything not | -1.538 |
| cheat | 1.565 | 3rd time | -1.545 |
| thicker | 1.580 | minute make | -1.577 |
| always delicious | 1.593 | exquisite | -1.628 |
| last visit | 1.594 | route | -1.633 |
| bread really | 1.713 | price service | -1.655 |
| find good | 1.759 | really sweet | -1.746 |
| aunt | 1.764 | fold | -1.918 |

A confusion matrix was generated from the results of using the fraud review predictor against the test set and the confusion matrix result is show in Figure 3 below. This shows that the logistic regression is not much more accurate than a coin flip. It predicts a review as fraudulent when the review is fraudulent at 1612 times out of 2,549 cases and predicts a real review when a review is real 1,475 times out of 2,646 events. In the case of a Type I error where a review is predicted to be fraudulent when it is real instead of being predicted correctly happens 937 out of 3,087 events in the test set, or 30.4% of the time. A Type II error, in which a real review is predicted to be fraudulent instead of being correctly predicted happens 1171 out of 3,087 events in the test set, or 37.9% of the time.



**Figure 3**, Confusion Matrix for Fraud Review Predictor against Test Set

Finally, an analysis of the precision-recall curve for the test set was done to show the tradeoff between the true positive rate and false positive rate for the fraud predictor (Brownlee, 2021). This is how the accuracy of 59.5% is generated and helps to show how this model is slightly better at predicting than a no-skill predictor. The precision-recall curve for the fraud review prediction logistic regression model can be seen in Figure 4. This model has about 10% more accuracy than a coinflip in predicting a true false positive or predicting a review is fraudulent.



**Figure 4**, Precision-Recall Curve for Fraud Review Predictor

**Model and Results Analysis for Fraud Review Predictor K-Nearest Neighbors**

In using the model for k-nearest neighbors (KNN) algorithm, the data scientist must select the value for k. To get the best value for k, ranges of k from 2 to 2,500 were run to calculate the best possible values for both accuracy and receiver operating characteristic (ROC). After this was allowed to run, the values 7 and 2003 were the two best values found for k, with near identical values for accuracy and ROC. The scikit package provides a modified KNN algorithm that allows supervised learning, and the number of neighbors was set to 7 for this model.

 In analyzing the final model, accuracy for KNN was found to be only about 53% and can be seen in Table 5. The mean squared error was found to be 1.68, and there isn’t a direct way to access R2 or variance for the KNN algorithm with scikit. According to the calculator at GraphPad, this leads to a p-value of roughly 1 and would not be statistically significant.

**Table 5**, Classification Report for KNN Fraud Review Predictor

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Precision | Recall | F1-Score | Support |
| Fraudulent Review | 0.70 | 0.51 | 0.59 | 3,488 |
| Real Review | 0.36 | 0.56 | 0.44 | 1,707 |
| Accuracy |  |  | 0.53 | 5,195 |
| Macro Average | 0.53 | 0.54 | 0.52 | 5,195 |
| Weighted Average | 0.59 | 0.53 | 0.54 | 5,195 |

 For prediction values, the mean was found to be -0.427, and the standard deviation found at 0.388. The mean was treated as the prediction point for this model. The predictions here cannot be directly associated with any specific n-grams, but the confusion matrix can still be evaluated, seen in Figure 5 below. This shows that the knn model is much more accurate than a coin flip when predicting a review fraudulent and it is fraudulent, but it also gives a high percentage of false negatives. It predicts a review as fraudulent when the review is fraudulent at 1795 times out of 2,549 cases and predicts a real review when a review is real 953 times out of 2,646 events. In the case of a Type I error where a review is predicted to be fraudulent when it is real instead of being predicted correctly happens 1,693 out of 2,748 events in the test set, or 61.6% of the time. A Type II error, in which a fraudulent review is predicted to be real instead of being correctly predicted happens 754 out of 2,748 events in the test set, or 27.4% of the time.



**Figure 5**, Confusion Matrix for KNN Fraud Review Predictor against Test Set

Finally, an analysis of the precision-recall curve for the test set was done to show the tradeoff between the true positive rate and false positive rate for the KNN fraud predictor (Brownlee, 2021). This is how the accuracy of 53.2% is generated and helps to show how this model is not much better at predicting than a no-skill predictor. The precision-recall curve for the fraud review prediction logistic regression model can be seen in Figure 6. This model has about 9% more accuracy than a coinflip in predicting a true false positive or predicting a review is fraudulent.

**Conclusion**

 Based on the analysis of the results, it is safe to reject the first null hypothesis. It is possible to use sentiment analysis, natural language processing, and n-gram processing to build a logistic regression to accurately predict the sentiment of a user to a particular restaurant in New York City based on the review that they provide of the restaurant. The use of two different models for the second hypothesis shows that rejecting the second null hypothesis is a safe conclusion as well. Though neither was especially accurate, the logistic regression fraud detector was more accurate than a no-skill predictor, such as a coin flip. Given that the number of negative reviews that are fraudulent are such a small sample in the entire population, it may be worth building a new model that first separates reviews into positive and negative, and then builds out separate models for fraud on each of those new data sets. This may provide better predictions and new methodologies moving forward. This also builds on the idea of the central limit theorem, in that when an event is predicted that is an outcome that is far less likely to be observed by chance, there should be more confidence that other factors are at play (Wheelan, 2014). To get the highest degree of accuracy and precision in fraud detection, it would be important to build models which are first accurate, and then test them against samples that more closely resemble the population to ensure that they maintain the same level of precision and accuracy moving forward.

References

Brownlee, J. (2021, Jan 13). How to Use ROC Curves and Precision-Recall Curves for Classification in Python. *Machine Learning Mastery*. <https://machinelearningmastery.com/roc-curves-and-precision-recall-curves-for-classification-in-python/>

GraphPad. (2021). Statistical distributions and interpreting P values. <https://www.graphpad.com/quickcalcs/distMenu/>

Harris, C.R. et al. (2020). Array programming with NumPy. *Nature 585, 357–362*. <https://doi.org/10.1038/s41586-020-2649-2>

McKinney, W. (2010) Data structures for statistical computing in python. *Proceedings of the 9th Python in Science Conference, 445*. <https://conference.scipy.org/proceedings/scipy2010/pdfs/mckinney.pdf>

Pedregosa et al. (2011). Scikit-learn: Machine Learning in Python. *JMLR 12, pp. 2825-2830*. <http://jmlr.csail.mit.edu/papers/v12/pedregosa11a.html>

Waskom, M. (2021). seaborn: statistical data visualization. *Journal of Open Source Software, 6 (60), 3021*. <https://doi.org/10.21105/joss.03021>

Wheelan, C. J. (2014). *In Naked Statistics: Stripping the Dread from the Data* (pp. 145-148). W.W. Norton & Company.