Group 3 Project:

NLP Comparison of Models

to Create a Survey Dialog Bot

PROG8420: Programming for Big Data

Jaibir Singh: 8652995

Tanya Grimes: 8721231

Friday, August 14, 2020

**Background**

The goal of this exercise is to determine which classification technique can generate an accurate survey dialog for a restaurant business to better understand their customers. The most efficient model will be used to power an application that captures the sentiment of its user base and ranks them, based on the underlying sentiment dataset. Appropriate responses will be generated by the application based on the response rankings to complete the survey dialog. The classification techniques to be tested are: Random Forest Classification, Logistic Regression and Naïve Bayes Classification.

**Data Source**

The dataset consists of two files: a flat csv file of a subset of Yelp reviews, as taken from Fan, 2019, and a json document of constructed dialog responses, based on the classification result rankings. The data dictionaries for each file can be seen in Appendix 1 and 2, respectively.

**Data Transformation and Cleaning**

**Stars and Text Variables**

The dataset was ultimately stripped of all other columns, except Stars and Text, as those were needed to run the binary analysis in the various classification techniques. Only the extreme star values of 1s and 5s were kept.

**2-Star Observations**

It was noted early in the data analysis that the dataset was dominated by 5-star ratings. 2-star ratings were reclassified as star-1 ratings and combined in attempts to balance out the dataset.

**Funny Variable**

The 5-star were reduced by filtering out all observations that given a ‘Funny’ score greater than 0 to help create further balance.

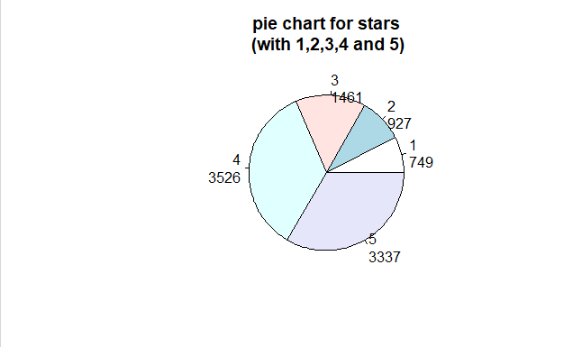
**NLP Text Transformation and Cleaning**

**Clean Text**

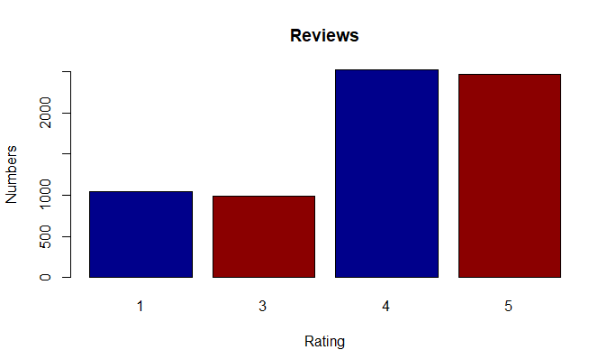
During the model preparation, the Text variable data was transformed. All unnecessary was stripped. The remaining text was tokenized into individual words and the unnecessary stem endings were removed from the stop-words. This was done in bulk for the Text variable, as well as for the individual user input text.

**Data Analysis**

The original data was analysed to determine what type of dimension reduction would be needed to isolate only 1-star and 5-star ratings. The source code can be seen in Appendix 3.



As seen by the pie chart above, the 5-star data for the ‘stars’ variable held a larger number of observations than the 1-star rating. As a result, the 1-star and 2-star ratings were combined, and the ‘funny’ observations were removed from the 5-star data. This increased the ratio from 3:1 to 2:1.



**Model Development**

Three datasets were randomly generated: a training dataset, containing 90%, a validation dataset containing 5% of the data, and a test dataset containing 5% of the data. Random Forest classification, Naïve-Bayes classification, and Logistic Regression models were generated to classify the Yelp text appropriately in a validation dataset. The datasets were re-generated for each model. A confusion matrix and various metrics were then calculated for each model. 1 represents star ratings of 1, while 5 represents star ratings of 5. The original data can be seen in Appendix 4.

**Random Forest Classification Results (RNF)**

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **RNF** |  | Predicted |  |  |  |  | % |  |  | Type 1 Errors | 17 |
|  |  | **1** | **5** |  |  | Accuracy | 87 |  |  | Type 2 Errors | 11 |
| Actual | **1** | 67 | 17 | 84 |  | Sensitivity | 91 |  |  |  |  |
|  | **5** | 11 | 112 | 123 |  | Specificity | 80 |  |  |  |  |
|  |  | 78 | 129 | 207 |  | Precision | 87 |  |  |  |  |

**Naïve Bayes Classification Results (NBC)**

The multinomial classification library in Python was used for this analysis.

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **NBC** |  | Predicted |  |  |  |  | % |  |  | Type 1 Errors | 30 |
|  |  | **1** | **5** |  |  | Accuracy | 83 |  |  | Type 2 Errors | 5 |
| Actual | **1** | 54 | 30 | 84 |  | Sensitivity | 96 |  |  |  |  |
|  | **5** | 5 | 118 | 123 |  | Specificity | 64 |  |  |  |  |
|  |  | 59 | 148 | 207 |  | Precision | 80 |  |  |  |  |

**Logistic Regression Classification Results (LGT)**

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **LGT** |  | Predicted |  |  |  |  | % |  |  | Type 1 Errors | 17 |
|  |  | **1** | **5** |  |  | Accuracy | 87 |  |  | Type 2 Errors | 9 |
| Actual | **1** | 67 | 17 | 84 |  | Sensitivity | 92 |  |  |  |  |
|  | **5** | 9 | 114 | 123 |  | Specificity | 80 |  |  |  |  |
|  |  | 76 | 131 | 207 |  | Precision | 87 |  |  |  |  |

Only the Random Forest Classification Logistic Regression models produced an ideal accuracy over 85%. Both models had very similar results. However, it was noted that while regenerating random the datasets, the Random Forest model would fluctuate between accuracies of 83% and 87%, while the Logistic Regression model remained consistent around 87%. Hence, the Logistic Regression model was selected to generate the NLP survey classifications. The model with the test dataset can be seen below.

**Logistic Regression Classification Results (LGT) with Test Dataset**

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **LGT** |  | Predicted |  |  |  |  | % |  |  | Type 1 Errors | 10 |
|  |  | **1** | **5** |  |  | Accuracy | 90 |  |  | Type 2 Errors | 10 |
| Actual | **1** | 74 | 10 | 84 |  | Sensitivity | 92 |  |  |  |  |
|  | **5** | 10 | 114 | 124 |  | Specificity | 88 |  |  |  |  |
|  |  | 84 | 124 | 208 |  | Precision | 92 |  |  |  |  |

**Observations on NLP Survey Dialog**

The survey dialog consists of three questions. Each user response is run through the classification model and the resulting probability taken from extremes values is ranked from 1 (worst) to 5 (best) to create a stand-alone classification system. The rating is then passed to the bot dialog to respond with an appropriate message. The average score of the survey is also passed to the bot dialog to respond with an appropriate summary message. Various scenarios, grouped by trials, were simulated to see how well the application dialog responded and the results are as follows.

**Trial 1: All positive responses submitted**

This resulted in correct dialog responses for both individual survey questions and the summary.

**Trial 2: All negative responses submitted**

This resulted in correct dialog responses for both individual survey questions and the summary.

**Trial 3: 2 negative responses and 1 positive response submitted**

This resulted in the appropriate dialog responses. The average survey score was in the medium range and the summary response reflected this.

**Trial 4: 2 positive responses and 1 negative response submitted**

This resulted in the appropriate dialog responses. The average survey score was in the higher range and the summary response reflected this.

**Trial 5: All neutral responses submitted**

This resulted in either classifying the results as a mix of high positive or high negative ranks. Sometimes, the average could result in a medium score, but not because the text was being classified as all neutral.

**Trial 6: All unrelated responses submitted**

Responses that have nothing to do with sentimental reviews were used, such as objective headlines. This resulted in all neutral rankings.

**Trial 7: Interesting results from responses submitted**

* Adding the word ‘atmosphere’ or ‘setting’ to a blatantly negative comment resulted in 4 or 5-star rating.
* Some neutral comments, such as ‘It was ok’ Received a 1-star rating.
* Other neutral comments, such as ‘I was full’ or ‘Same as usual’ received a 4 or 5-star ratings.
* Gibberish text received a 4 or 5-star ratings.
* Use of double negatives, such as ‘I didn’t hate this at all’ received a 1 to 2-star rating.

**Discussion on NLP Survey Dialog**

**Biased Dataset**

The Yelp dataset used was heavily dominated by 5-star ratings. Even after some dimensionality reduction, these ratings were still in a larger proportion than 1-star ratings. This can lead to higher chances of a word being categorized as positive. It is possible that words such as ‘atmosphere’ and ‘setting’ were found mostly in 5-star ratings and as a result, aided in classifying the text as positive.

**Words taken out of context**

The misclassification of double negatives suggests that insufficient contextual classification was present. It is possible that this is more of an advanced topic and should be considered at a future date. This also includes gibberish words and possible profanity.

**Lack of a middle tier in the analysis**

This analysis was based solely on a binary classification of 1-star and 5-star. As a result, the model had to decide what to categorize as intermediate responses. Only responses with around 50% probability of being a 1 or a 5 ended up in this category. These were mostly the unrelated text messages noted above. Furthermore, neutral responses related to sentimental reviews were classified as one of the extreme ratings. This is possibly a result of similar words being found in those categories, due to a lack of a category on its own.

**Suggestions for Improvement**

* Extend the Dataset
  + Try the original Yelp dataset and reduce accordingly as necessary.
  + Include dialogs to handle unrelated customer responses
* More Involved Data Cleanup
  + Perform deeper exploratory analysis to reduce the data further for a more balanced dataset.
* Include classification of 3-star ratings
  + Switch to multi-classification techniques to accommodate non-binary data.

**Conclusion**

Although the model used returned a high accuracy reading, further improvements still need to be made when handling user input. This exercise is only the beginning to building a more efficient survey dialog bot.

**References**

Fan, Z. (2019). *NLP for Yelp Reviews*. Kaggle. Retrieved August 8, 2020 from <https://www.kaggle.com/zhenyufan/nlp-for-yelp-reviews?select=yelp.csv>

**APPENDIX 1: Data Dictionary for Yelp Reviews (Fan, 2019)**

|  |  |
| --- | --- |
| **Name** | **Description** |
| Business\_id | Unique Identifier for business |
| Date | Reviews collected between 17 April 05 and 4 Jan 2013 |
| Review\_id | Unique identifier for reviews |
| Stars | Star rating from 1 (worst) to 5 (best) |
| Text | User written sources of text |
| Type | Type of content for the text field |
| User\_id | Unique identifier for users |
| Cool | Grouping of reviews that users flagged as cool |
| Useful | Grouping of reviews that users flagged as useful |
| Funny | Grouping of reviews that users flagged as funny |

**APPENDIX 2: Data Dictionary for Restaurant Business Bot Response Dialog**

|  |  |
| --- | --- |
| **Name** | **Description** |
| Star1 | Bot dialog response for any survey answer that was classified with star rating 1 |
| Star2 | Bot dialog response for any survey answer that was classified with star rating 2 |
| Star3 | Bot dialog response for any survey answer that was classified with star rating 3 |
| Star4 | Bot dialog response for any survey answer that was classified with star rating 4 |
| Star5 | Bot dialog response for any survey answer that was classified with star rating 5 |
| Overall1 | Bot dialog response for overall average survey score that was low |
| Overall2 | Bot dialog response for overall average survey score that was medium |
| Overall3 | Bot dialog response for overall average survey score that was high |

**APPENDIX 3: Data Analysis**

**Creating pie chart of original data, using R**

js2<-table(js$stars)

labels<- paste(names(js2),"\n",js2,sep="")

pie(js2,labels = labels,main="pie chart for stars\n (with 1,2,3,4 and 5)")

**Creating bar plots after each data transformation, using R**

barplot(js2, xlab='Rating',ylab='Numbers',main="Review",

col=c("darkblue","darkred")

,legend=rownames(js2), args.legend = list(x = "topright"))

js1<-table(js$stars[1:1:5])

labels<- paste(names(js1),"\n",js1,sep="")

tv<-pie(js1,labels = labels,main="pie chart for stars\n (with 1,2,3,4 and 5)")

barplot(js1, xlab='Rating',ylab='Numbers',main="Reviews",

col=c("darkblue","darkred")

,legend=rownames(1))

summary(rjs)

rjs1<-table(rjs$stars)

labels<- paste(names(rjs1),"\n",rjs1,sep="")

pie(rjs1,labels = labels,main="pie chart for stars\n (with 1,2,3,4 and 5)")

barplot(rjs1, xlab='Rating',ylab='Numbers',main="Reviews",

col=c("darkblue","darkred")

,legend=rownames(1))

fjs1<-table(fjs$stars)

labels<- paste(names(fjs1),"\n",fjs1,sep="")

pie(fjs1,labels = labels,main="pie chart for stars\n (with 1,2,3,4 and 5)")

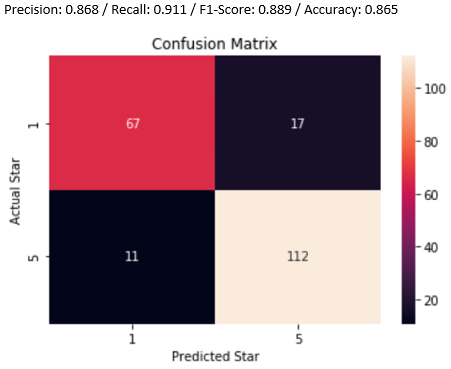
barplot(fjs1, xlab='Rating',ylab='Numbers',main="Reviews",

col=c("darkblue","darkred")

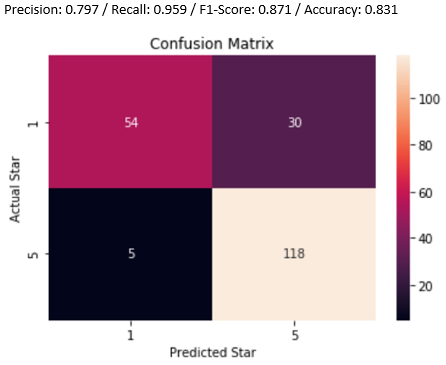
,legend=rownames(1))

**APPENDIX 4: Model Results**

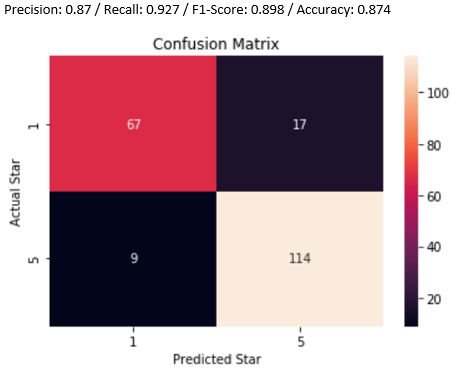
**Generating Random Forest Validation Model Results**



**Generating Naïve Bayes Validation Model Results**



**Generating Logistic Regression Validation Model Results**



**Generating Logistic Regression Test Model Results**

