REVIEW RATING PREDICATION USING NLP

Submitted by:
Suraj Kumar Soni

SME: **Shwetank Mishra**

ACKNOWLEDGMENT

I would like to express my special gratitude to “Flip Robo” team, who has given me this opportunity to deal with a beautiful dataset and it has helped me to improve my analyzation skills. And I want to express my huge gratitude to Shwetank Mishra (SME Flip Robo), he is the person who has helped me to get out of all the difficulties I faced while doing the project.

Introduction

Business Problem Framing

Our customer has a website where users can post various product reviews for technical items. The reviewer will now be required to include stars (ratings) along with the review on their website, which is a new feature they are currently adding. There are only 5 alternatives available, and the ranking is out of 5. 1, 2, 3, 4, and 5 stars, respectively. They are attempting to forecast ratings for past reviews that have not yet received one. Therefore, we must create a programme that can gauge the rating from the review.

Conceptual Background of the Domain Problem

In general, shoppers utilise two straightforward heuristics to determine whether to make a final purchase of a product: ratings and pricing. The total star ratings of the product reviews, however, frequently do not accurately reflect the polarity of the opinions. Due of the possibility of varying customer ratings for a given review, rating prediction becomes a challenging topic. For instance, one person might give a product a 5-star rating and rate it as nice, but another user might write the same comment and only give it a 3-star rating. Additionally, reviews could include anecdotal data, which is not informative and makes forecasting more difficult.

Users may select different ways to express their sentiments. For instance, some users might use the word "good" to describe a merely passable product, while others might use it to describe a top-notch one. In addition to user bias, there is product bias. For example, the opinion word "long" can express a "positive" feeling for a cell phone's battery life but a "negative" feeling for a camera's focus time. We may use different opinion words to review different products, or even the same opinion word to express different sentiment polarities for different products. For the purpose of forecasting review ratings, it is crucial to take into account both the relationships between the review authors and the target products.

Review of Literature

According to the Lackermair, Kailer and Kanmaz (2013), product reviews and ratings represent an important source of information for consumers and are helpful tools in order to support their buying decisions [6]. They also found out that consumers are willing to compare both positive and negative reviews when searching for a specific product. The authors argue that customers need compact and concise information about the products. Therefore, consumers first need to pre-select the potential products matching their requirements. With this aim in mind, consumers use the star ratings as an indicator for selecting products. Later, when a limited number of potentials products have been chosen, reading the associated text review will reveal more details about the products and therefore help consumers making a final decision.

It becomes daunting and time-consuming to compare different products in order to eventually make a choice between them. Therefore, models able to predict the user rating from the text review are critically important (Baccianella, Esuli & Sebastiani, 2009) [7].
Pang, Lee and Vaithyanathan (2002) [9] approach this predictive task as an opinion mining problem enabling to automatically distinguish between positive and negative reviews. In order to determine the reviews polarity, the authors use text classification techniques by training and testing binary classifiers on movie reviews containing 36.6% of negative reviews and 63.4% of positive reviews. On the top of that, they also try to identify appropriate features to enhance the performance of the classifiers.
Dave, Lawrence, and Pennock (2003) [10] also deal with the issue of class imbalance with a majority of positive reviews and show similar results. SVM outperforms Naïve Bayes with an accuracy greater than 85% and the implementation of part-of-speech as well as stemming is also ineffective. However, this work demonstrates that bigrams turn out to be more successful at capturing context than unigrams in the specific situation of their datasets, despite earlier research having produced better results with unigrams.
to capture the weights of such characteristics by minimising the mean square error.

Motivation for the Problem Undertaken

My first project from Flip Robo Technologies under the internship programme was the project. The main drivers behind this were the chance to apply my skill set to a real-world problem and the exposure to data from the actual world.
The data needed for this project must be scraped from an e-commerce site and cleaned up. Its associated star ratings are predicted using features collected from textual evaluations. To do this, the prediction issue is turned into a task requiring multi-class classification, where reviews are assigned to one of five categories based on their star rating. Gaining a general understanding of a text review may enhance the user experience. However, the reason I decided to do this project was because it is relatively a new field of research.

Analytical Problem Framing

Mathematical / Analytical Modelling of the Problem

$$tf - id f_{t,d} = tf_{t,d} * id f_t$$

where:

- $tf_{t,d} = \frac{n_{t,d}}{\sum_k n_{k,d}}$ with $n_{t,d}$ the number of term t contained in a document d, and $\sum_k n_{k,d}$ the total number of terms k in the document d
- $id f_t = \log \frac{N}{df_t}$ with N the total number of documents and df_t the number of documents containing the term t

In order to apply text classification, the unstructured format of text has to be converted into a structured format for the simple reason that it is much easier for computer to deal with umbers

than text. This is mainly achieved by projecting the textual contents into Vector Space Model, where text data is converted into vectors of numbers.
Documents are frequently handled like a Bag-of-Words (BoW) in the field of text categorization, which means that each word is distinct from the other words that are present in the text. They are scrutinised without consideration for grammar or word order. In this model, the classifier is trained using the term-frequency (the frequency with which each word occurs) as a feature. However, the use of the word frequency suggests that all concepts are given equal weight. The word frequency, as its name implies, does nothing more than weight each term according to how frequently it occurs; it does not take the discriminatory potential of terms into consideration. Each word is given a term frequency inverse document frequency in order to handle this issue and penalise words that are used excessively (tf-idf) score which is defined above:

Data Sources and their formats

Data is collected from Amazon.using selenium and saved in CSV file. Around 20000 Reviews are collected for this project.

```
print('No. of Rows :',data.shape[0])
print('No. of Columns :',data.shape[1])
pd.set_option('display.max_columns',None)
data.head()
```

```
No. of Rows : 24652
No. of Columns : 2
```

This is multi-classification problem and Rating is our target feature class to be predicated in this project. There are five different categories in feature target i.e., The rating is out 5 stars and it only has 5 options available 1 star, 2 stars, 3 stars, 4 stars, 5 stars.

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 24652 entries, 0 to 24651
Data columns (total 2 columns):
 #   Column    Non-Null Count  Dtype 
---  ------    --------------  ----- 
 0   Rattings  24652 non-null  int64 
 1   Review    24641 non-null  object
dtypes: int64(1), object(1)
memory usage: 385.3+ KB
```

There are some missing values in product review. The datatype of Product review is object while datatypes of Ratings is int.

Data Pre-processing

The dataset is large and it may contain some data error. In order to reach clean, error free data some data cleaning & data pre-processing performed data.

Missing Value Imputation:

Missing value in product reviews are replace with 'Review Not Available'.

Data is pre-processed using the following techniques:

Convert the text to lowercase
Remove the punctuations, digits and special characters
Tokenize the text, filter out the adjectives used in the review and create a new column in data frame
Remove the stop words
Stemming and Lemmatising
Applying Text Vectorization to convert text into numeric

Data Inputs- Logic- Output Relationships

The dataset consists of 2 features with a label. The features are independent and label is dependent as our label varies the values (text) of our independent variable's changes. Using word cloud, we can see most occurring word for different categories.

Hardware & Software Requirements with Tool Used

Hardware Used -

- Processor — Intel i3 processor with 2.4GHZ
- RAM— 4GB
- GPU — 2GB AMD Radeon Graphics card Software utilised -
- Anaconda – Jupyter Notebook
- Selenium – Web scraping
- Google Colab – for Hyper parameter tuning
- Libraries Used – General library for data wrangling & visualsation

Models Development & Evaluation

Identification Of Possible Problem-Solving Approaches (Methods)

First part of problem solving is to scrap data from amazon which we already done. Second is performing text mining operation to convert textual review in ML algorithm useable form. Third part of problem building machine learning model to predict rating on review. This problem can be solve using classification-based machine learning algorithm like logistics regression. Further Hyperparameter tuning performed to build more accurate model out of best model.

Testing of Identified Approaches (Algorithms)

The different classification algorithm used in this project to build ML model are as below:
Random Forest classifier
Decision Tree classifier

Logistics Regression
AdaBoost Classifier
Gradient Boosting Classifier

Key Metrics for Success in Solving Problem Under Consideration

Precision can be seen as a measure of quality; higher precision means that an algorithm returns more relevant results than irrelevant ones.
Recall is used as a measure of quantity and high recall means that an algorithm returns most of the relevant results.
Accuracy score is used when the True Positives and True negatives are more important.
Accuracy can be used when the class distribution is similar. F1-score is used when the False Negatives and False Positives are crucial. While F1-score is a better metric when there are imbalanced classes.
Cross validation Score: To run cross-validation on multiple metrics and also to return train scores, fit times and score times. Get predictions from each split of cross-validation for diagnostic purposes. Make a scorer from a performance metric or loss function.

Run And Evaluate Selected Models

- Logistics Regression

```
print('Training feature matrix size:',X_train.shape)
print('Training target vector size:',Y_train.shape)
print('Test feature matrix size:',X_test.shape)
print('Test target vector size:',Y_test.shape)
```

```
Training feature matrix size: (17256, 7577)
Training target vector size: (17256, 1)
Test feature matrix size: (7396, 7577)
Test target vector size: (7396, 1)
```

Train-test split is used to split data into training data & testing data. Further best random state is investigated through loop.

```
# Finding best Random state
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, confusion_matrix,classification_report,f1_score
maxAccu=0
maxRS=0
for i in range(50,100):
    X_train,X_test,Y_train,Y_test = train_test_split(X,Y,test_size = 0.3, random_state=i)
    log_reg=LogisticRegression()
    log_reg.fit(X_train,Y_train)
    y_pred=log_reg.predict(X_test)
    acc=accuracy_score(Y_test,y_pred)
    if acc>maxAccu:
        maxAccu=acc
        maxRS=i
print('Best accuracy is', maxAccu ,'on Random_state', maxRS)
```

```
Best accuracy is 0.9057598702001082 on Random_state 69
```

Logistics regression evaluation matrix is shown below:

Logistics Regression Evaluation

Accuracy Score of Logistics Regression : 0.9057598702001082

Confusion matrix of Logistics Regression :

```
[[ 127    0    5   27    1]
 [   0   43    1   23    6]
 [   0    0  315  197   10]
 [   0    0    4 5041   64]
 [   0    0    0  359 1173]]
```

classification Report of Logistics Regression

	precision	recall	f1-score	support
1	1.00	0.79	0.89	160
2	1.00	0.59	0.74	73
3	0.97	0.60	0.74	522
4	0.89	0.99	0.94	5109
5	0.94	0.77	0.84	1532
accuracy			0.91	7396
macro avg	0.96	0.75	0.83	7396
weighted avg	0.91	0.91	0.90	7396

- Decision Tree Classifier

Decision Tree Classifier model is built and evaluation matrix is shown as below:

Decision Tree Classifier Evaluation

Accuracy Score of Decision Tree Classifier : 0.9509194159004868

Confusion matrix of Decision Tree Classifier :

```
[[ 140    0    1   18    1]
 [   1   64    0    8    0]
 [   2    1  456   55    8]
 [   3    5   14 5033   54]
 [   0    1    7  184 1340]]
```

classification Report of Decision Tree Classifier

	precision	recall	f1-score	support
1	0.96	0.88	0.92	160
2	0.90	0.88	0.89	73
3	0.95	0.87	0.91	522
4	0.95	0.99	0.97	5109
5	0.96	0.87	0.91	1532
accuracy			0.95	7396
macro avg	0.94	0.90	0.92	7396
weighted avg	0.95	0.95	0.95	7396

- Random Forest Classifier
-

Random Forest Classifier

Accuracy Score of Random Forest Classifier : 0.9564629529475392

Confusion matrix of Random Forest Classifier :

```
[[ 140    0    0   19    1]
 [   0   64    0    9    0]
 [   0    0  454   66    2]
 [   0    0    0 5081   28]
 [   0    0    0  197 1335]]
```

classification Report of Random Forest Classifier

	precision	recall	f1-score	support
1	1.00	0.88	0.93	160
2	1.00	0.88	0.93	73
3	1.00	0.87	0.93	522
4	0.95	0.99	0.97	5109
5	0.98	0.87	0.92	1532
accuracy			0.96	7396
macro avg	0.98	0.90	0.94	7396
weighted avg	0.96	0.96	0.96	7396

- Ada Boost Classifier

AdaBoost Classifier Evaluation

Accuracy Score of AdaBoost Classifier : 0.7174148188209843

Confusion matrix of AdaBoost Classifier :

```
[[  66    1    0   83   10]
 [   0   12    1   36   24]
 [   6    3   48  442   23]
 [   8    7   48 4909  137]
 [   8   11   11 1231  271]]
```

classification Report of AdaBoost Classifier

	precision	recall	f1-score	support
1	0.75	0.41	0.53	160
2	0.35	0.16	0.22	73
3	0.44	0.09	0.15	522
4	0.73	0.96	0.83	5109
5	0.58	0.18	0.27	1532
accuracy			0.72	7396
macro avg	0.57	0.36	0.40	7396
weighted avg	0.68	0.72	0.65	7396

- Gradient Boosting Classifier

Gradient Boosting Classifier Evaluation

Accuracy Score of Gradient Boosting Classifier : 0.8597890751757706

Confusion matrix of Gradient Boosting Classifier :

```
[[ 139    0    0   20    1]
 [   0   64    0    8    1]
 [   1    1  216  293   11]
 [   2    2    3 5074   28]
 [   1    1    2  662  866]]
```

classification Report of Gradient Boosting Classifier

	precision	recall	f1-score	support
1	0.97	0.87	0.92	160
2	0.94	0.88	0.91	73
3	0.98	0.41	0.58	522
4	0.84	0.99	0.91	5109
5	0.95	0.57	0.71	1532
accuracy			0.86	7396
macro avg	0.94	0.74	0.81	7396
weighted avg	0.88	0.86	0.84	7396

5-fold Cross validation performed over all model. We can see that Random Forest Classifier gives us good Accuracy and maximum f1 score along with best Cross-validation score. Hyperparameter tuning is applied over Random Forest model and used it as final model.

```
[CV 3/5; 12/12] START criterion=entropy, max_features=log2, n_estimators=150....
[CV 3/5; 12/12] END criterion=entropy, max_features=log2, n_estimators=150;, score=0.945 total time=  28.2s
[CV 4/5; 12/12] START criterion=entropy, max_features=log2, n_estimators=150....
[CV 4/5; 12/12] END criterion=entropy, max_features=log2, n_estimators=150;, score=0.946 total time=  27.0s
[CV 5/5; 12/12] START criterion=entropy, max_features=log2, n_estimators=150....
[CV 5/5; 12/12] END criterion=entropy, max_features=log2, n_estimators=150;, score=0.943 total time=  28.3s
GridSearchCV(estimator=RandomForestClassifier(),
             param_grid={'criterion': ['gini', 'entropy'],
                         'max_features': ['auto', 'log2'],
                         'n_estimators': [75, 100, 150]},
             verbose=10)
```

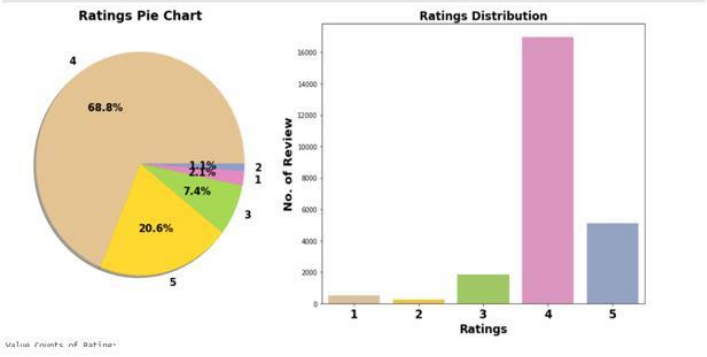
```
GCV.best_params_
```

```
{'criterion': 'entropy', 'max_features': 'auto', 'n_estimators': 75}
```

Final model is built using best parameter in hyper parameters tuning. The corresponding evaluation matrix shown below:

```
Final Random Forest Classifier Model
Accuracy Score :
 0.9571389940508382


Confusion matrix of Random Forest Classifier :
 [[ 140    0    0   19    1]
 [   0   64    0    9    0]
 [   0    0  454   66    2]
 [   0    0    0 5083   26]
 [   0    0    0  194 1338]]


Classification Report of Random Forest Classifier
              precision    recall  f1-score   support

           1       1.00      0.88      0.93       160
           2       1.00      0.88      0.93        73
           3       1.00      0.87      0.93       522
           4       0.95      0.99      0.97      5109
           5       0.98      0.87      0.92      1532

    accuracy                           0.96      7396
   macro avg       0.99      0.90      0.94      7396
weighted avg       0.96      0.96      0.96      7396
```

Visualizations

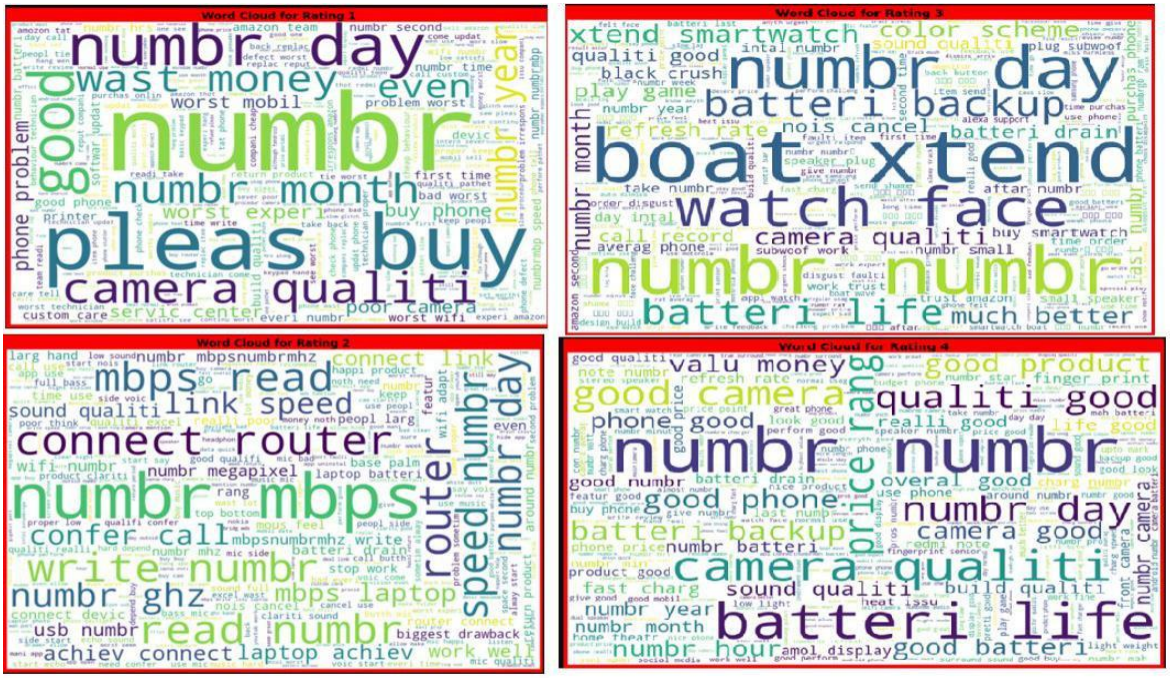


Comment:

1. Around 68% customer given 4- star rating followed by 20% customer given lowest 5-star rating.
2. Average Rating is 4.04

Word Cloud:

Word Cloud is a visualization technique for text data wherein each word is picturized with its importance in the context or its frequency.
The more commonly the term appears within the text being analysed, the larger the word appears in the image generated.The enlarged texts are the greatest number of words used here and small texts are the smaller number of words used



Conclusion

Key Findings and Conclusion of the Study

Algorithm	Accuracy Score	Recall	Precision	F1 Score	CV Score
Logistics Regression	0.9057	0.75	0.96	0.91	0.71
Decision Tree Classifier	0.9509	0.90	0.94	0.95	0.74
Random Forest Classifier (RFC)	0.9564	0.90	0.98	0.96	0.73
Gradient Boosting Classifier	0.8597	0.74	0.94	0.86	0.70
Ada Boost Classifier	0.7174	0.36	0.57	0.72	0.67
Final Model (RFC- Tuned)	0.9571	0.90	0.99	0.96	0.74

Final Model is giving us Accuracy score of 95.7% which is slightly improved compare to earlier Accuracy score of 95.6%.

Learning Outcomes of Data Science

Hands on chance to enhance my web scraping skillset.
In this project we were able to learn various Natural language processing techniques like lemmatization, stemming, removal of Stop words.
This project has demonstrated the importance of sampling effectively, modelling and predicting data.

Limitations of this work and Scope for the Future

More input features can be scrap to build predication model.
There is scope for application of advanced deep learning NLP tool to enhanced text mining operation which eventually help in building more accurate model with good cross validation score.

References use in this project:

- SCIKIT Learn Library Documentation
- Blogs from towardsdatascience, Analytics Vidya, Medium
- Andrew Ng Notes on Machine Learning (GitHub)
- Data Science Projects with Python Second Edition by Packt
- Hands on Machine learning with scikit learn and tensor flow by Aurelien Geron
- Lackermair, G., Kailer, D. & Kanmaz, K. (2013). Importance of online product reviews from a consumer's perspective. Horizon Research Publishing, 1-5. doi:
- 10.13189/aeb.2013.010101
- Baccianella, S., Esuli, A. & Sebastiani, F. (2009). Multi-facet rating of product reviews. Proceedings of the 31st European Conference on Information Retrieval (ECIR), 461-472