FLIP ROBO

Ratings prediction project

Submitted by:

POOJASHREE D S

ACKNOWLEDGMENT

• Sentiment analysis by text mining by Bert Cameron featured in towardsdatascience.

• Ultimate guide to deal with text data by Shubham Jain featured in Analytics vidya

• Visualizing text data using wordcloud by Deepika Singh featured in Pluralsight.

• Review rating prediction: A combined approach by Yereya Berdugo featured in towardsdatascience.

• Guide to text classification with machine learning and NLP featured in medium.

INTRODUCTION

- Business Problem Framing

  Missing ratings from products may result in the decrease in their popularity, most of the online shoppers view a product owing to its rating. The review only comes into the picture when they have an idea of purchase. Rating is what attracts the first response. The aim of this problem is to generate ratings from the review text. There are five classes in ratings 1,2,3,4 and 5, we analyse and process the text data to get the corresponding rating. Ratings prediction and generation is a popular sentiment analysis application. It has wide spread use in e-commerce platforms with missing ratings. The first step is to scrape data off e-commerce websites namely reviews and its corresponding ratings. Then we proceed to Natural Language Processing and Sentiment analysis to predict the rating.

- Conceptual Background of the Domain Problem

  To come up with a satisfactory solution it is necessary to have knowledge of web scraping techniques and the tools associated with BeautifulSoup and Selenium. Scraping of data must be carried out keeping in mind the importance of balanced nature of the target. Knowledge on natural language processing libraries and understanding of text analysis techniques and machine learning concepts is essential.

- ## Review of Literature

  The main concern with any dataset with text as input is its processing and analysis. Thorough processing will result in better extraction of features through vectorization which is vital for building a machine learning model. Various NLP tools and libraries such as NLTK, TextBlob and GenSim are employed for the text processing step. Text vectorization or embeddings is an important step that converts the text data into numerical representations that are required as input for model generation. This can be achieved by many ways including Tf-idf vectorizer. Text visualization through wordcloud gives us a pictorial representation that is favourable and easy to view. The final step is to run multiple models and evaluate their performance to choose the most suitable algorithm for the problem. Algorithms that are popular for text processing are Multinomial Naive Bayes, SVM and deep learning.

- ## Motivation for the Problem Undertaken

  This project will equip us with hands-on experience in text processing and model building. The importance of emojis and emoticons in predicting ratings and using histograms to detect if the length has an influence on target is also gathered from this project. This problem also provides us with a challenge that can be employed towards our growth in the data science and analytics field.

Analytical Problem Framing

- Mathematical/ Analytical Modeling of the Problem

  The first step is to look at the size of the data set. We do this with the help df .shape method. The given data set consists of 23555 entries with 2 columns. As the aim of our problem is classifying the given review into different ratings, we look at the counts of different classes of target by .value_counts() method, it is clear that the data is balanced from the obtained output. The common and rare words across all classes are generated.

- Data Sources and their formats

  Data source: Flipkart and Amazon. The data is scraped from the websites of e-commerce giants flipkart and Amazon. The reviews scraped are of electronics including mobile phones, monitors, headphones, routers, printers, home theatres and laptops using Selenium. The review and ratings are then made into a dataframe and stored in the form of a csv file for processing. The data has one feature column – Review and one target column – Rating. Review comprises text data that needs to be classified into a rating category. There are five classes in the Ratings column as star entries – one, two, three, four and five star ratings.

- Data Preprocessing

  Presence of missing values is detected with the help of .isnull() method. The data consists of one NaN value in the review column, we remove the row associated with it using dropna. It is vital to convert all text data into lower cases to maintain the uniformity of words and to make sure that the model does not differentiate between capital and small letters. Also, we remove unnecessary characters that do not provide any information such as punctuations, numbers and stopwords. Rare and common words in the text data are generated to get a general idea of the review emotion. Common words are generated with the help of Counter(), the top 20 frequent words along with their frequencies are seen in the output. Some of them are product, good, quality, nice, battery and sound. Some of the rare words are compatible, keyboard, existed and prolific. Emojis are useful in detecting the mindset of the consumers, emot library is used to convert emojis to their respective emotion-word. A function convert_emojis is defined for this purpose. The function is then applied on the review column. Tokenization is also an essential step in text processing that helps in computing lengths of text data and vectorization. A new column consisting of the length of the text in each entry is produced to understand the influence of length on target. This is done with the help of histograms with respect to each class in the target. The average length for each class is also computed. The output implies that the length has no particular influence on the target. The wordcloud library assists in visualizing loud words with respect to each category of the target. There are many common frequent words that are present in all classes. Vectorization of text data is achieved through tf-idf vectorizer. Train_test_split is used for training and validating the data set with the help of evaluation metrics.

- ## Hardware and Software Requirements and Tools Used

Pandas and NumPy are the basic libraries imported. Matplotlib.pyplot and seaborn are used for visualization. Wordcloud is the main visualization tool used in case of text data. Libraries nltk, emot and string are used for text exploration and cleaning. Sklearn is used for pre-processing and model building steps. All algorithms for processing are imported. Sklearn.metrics will provide the needed evaluation metrics such as accuracy score, classification report and confusion matrix

Model/s Development and Evaluation

- Identification of possible problem-solving approaches (methods)

The size of the dataset is about twenty thousand and the dataset does not contain many features, it has one column of text data. Hence, both high bias and high variance models can be considered. Naive bayes models are popular when it comes to text processing, we make use of Multinomial Naive Bayes. SVM and trees also perform well on text data, hence we also use SVM and decision tree classifiers.

- Testing of Identified Approaches (Algorithms)

The algorithms used are:

Multinomial

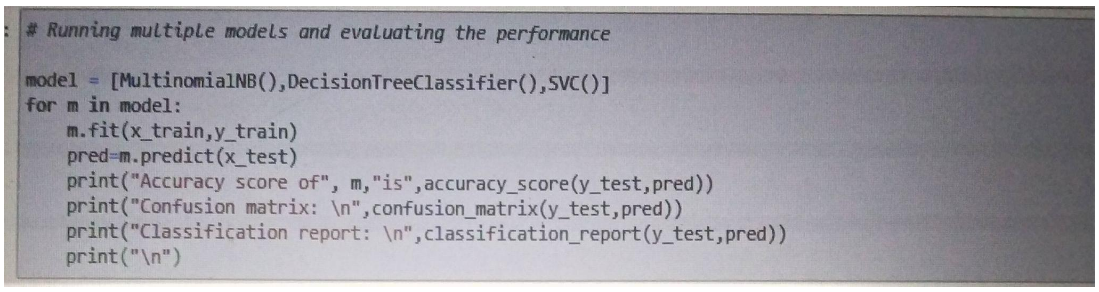
Naive Bayes

Support Vector Machine Classifier

Decision Tree Classifier

- Run and Evaluate selected models

The above algorithms are fitted to x_train and y_train which is generated by train test split method. The performance of the model is validated based on the predictions made for x_test.



```
# Running multiple models and evaluating the performance

model = [MultinomialNB(),DecisionTreeClassifier(),SVC()]
for m in model:
    m.fit(x_train,y_train)
    pred=m.predict(x_test)
    print("Accuracy score of", m,"is",accuracy_score(y_test,pred))
    print("Confusion matrix: \n",confusion_matrix(y_test,pred))
    print("Classification report: \n",classification_report(y_test,pred))
    print("\n")
```

```
Accuracy score of MultinomialNB() is 0.5151772447463383
Confusion matrix:
 [[1126   24   66   67   32]
 [ 471   51   79   61   27]
 [ 236   10  296  271   97]
 [  67    5   87  479  285]
 [  68    3   38  290  475]]
Classification report:
              precision    recall  f1-score   support

           1       0.57      0.86      0.69      1315
           2       0.55      0.07      0.13       689
           3       0.52      0.33      0.40       910
           4       0.41      0.52      0.46       923
           5       0.52      0.54      0.53       874

    accuracy                           0.52      4711
   macro avg       0.51      0.46      0.44      4711
weighted avg       0.52      0.52      0.48      4711



Accuracy score of DecisionTreeClassifier() is 0.4272978136276799
Confusion matrix:
 [[843 190 152  84  46]
 [303 156 124  74  32]
 [178  91 264 258 119]
 [ 69  49 134 406 265]
 [ 89  32  95 314 344]]
Classification report:
              precision    recall  f1-score   support

           1       0.57      0.64      0.60      1315
           2       0.30      0.23      0.26       689
           3       0.34      0.29      0.31       910
           4       0.36      0.44      0.39       923
           5       0.43      0.39      0.41       874

    accuracy                           0.43      4711
   macro avg       0.40      0.40      0.40      4711
weighted avg       0.42      0.43      0.42      4711



Accuracy score of SVC() is 0.5243048185098705
Confusion matrix:
 [[1073   69   92   57   24]
 [ 414  105  103   48   19]
 [ 183   35  346  259   87]
 [  56   15   96  494  262]
 [  61    9   54  298  452]]
Classification report:
              precision    recall  f1-score   support

           1       0.60      0.82      0.69      1315
           2       0.45      0.15      0.23       689
           3       0.50      0.38      0.43       910
           4       0.43      0.54      0.48       923
           5       0.54      0.52      0.53       874

    accuracy                           0.52      4711
   macro avg       0.50      0.48      0.47      4711
weighted avg       0.51      0.52      0.50      4711
```

Multiple models are fitted with the help of a for loop. The evaluation metrics- accuracy score, confusion matrix and classification report are used. The above output indicates that SVM and Multinomial NB have the highest accuracy score. The classification report gives a detailed picture on the performance of all the classes of the target. Gridsearch CV is the best method for hyperparameter tuning, we apply it on SVM to obtain a better version of the algorithm. The best parameters yielded from Gridsearch CV are: {'C': 1, 'gamma': 1, 'kernel': 'rbf'}
The model is fitted and evaluated again with the above parameters. The output of the evaluation metrics can be seen below:

```
Accuracy score of SVC(C=1, gamma=1) is 0.5238802801952877
Confusion matrix:
 [[1072   69   93   57   24]
 [ 414  104  104   48   19]
 [ 183   35  347  259   86]
 [  56   15   97  493  262]
 [  61    9   54  298  452]]
Classification report:
              precision    recall  f1-score   support

           1       0.60      0.82      0.69      1315
           2       0.45      0.15      0.23       689
           3       0.50      0.38      0.43       910
           4       0.43      0.53      0.47       923
           5       0.54      0.52      0.53       874

    accuracy                           0.52      4711
   macro avg       0.50      0.48      0.47      4711
```

```
weighted avg       0.51      0.52      0.50      4711
```

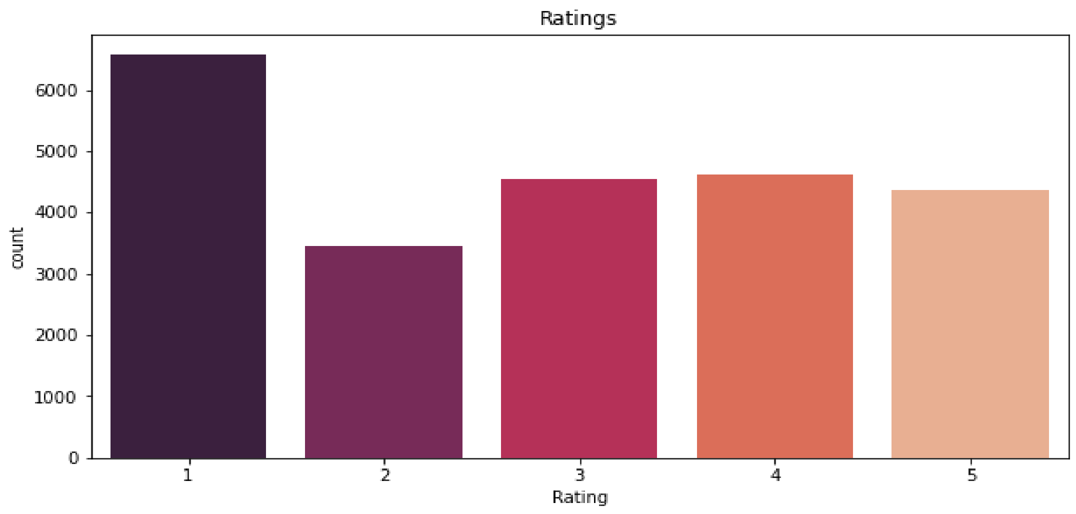
Accuracy score of 0.52 indicates that the performance of the model is average. The recall, precision and f1 score in the classification report shows that the performance in all classes is similar and the model is not biased towards a particular category. Confusion matrix shows the quantity of False positives and False negatives.

- ## Key Metrics for success in solving problem under consideration

The metrics used are accuracy score, classification report and confusion matrix. Accuracy score gives the performance of both classes as a whole. Classification report consisting of precision, recall and f1 score help us in understanding the efficiency of the model with respect to each class. Confusion matrix places a distinction between false positives and false negatives.

- ## Visualizations

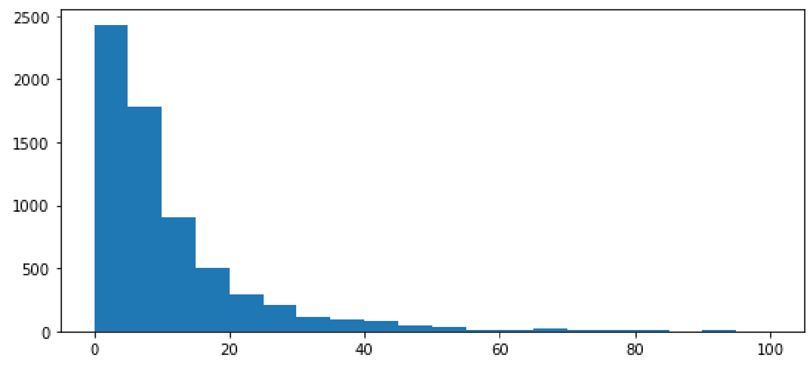
<u>Count plot</u> is used to get a grip of the number of entries in different classes of target.



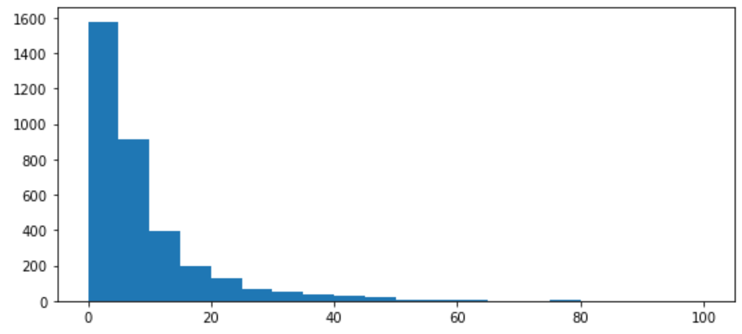
The target is fairly balanced with each class approximately four thousand entries.

<u>Histograms</u> are generated to understand the distribution of word count in different star ratings.

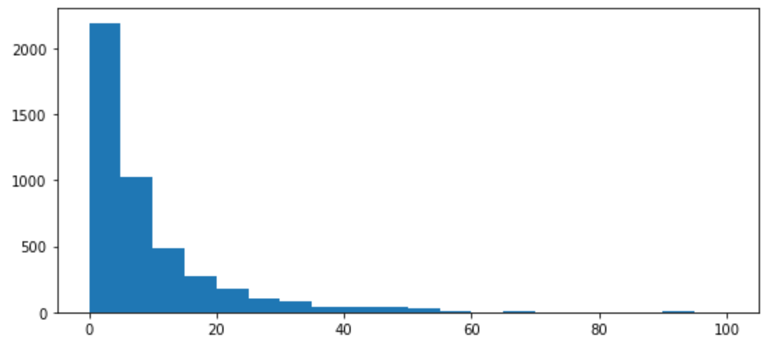
One star:



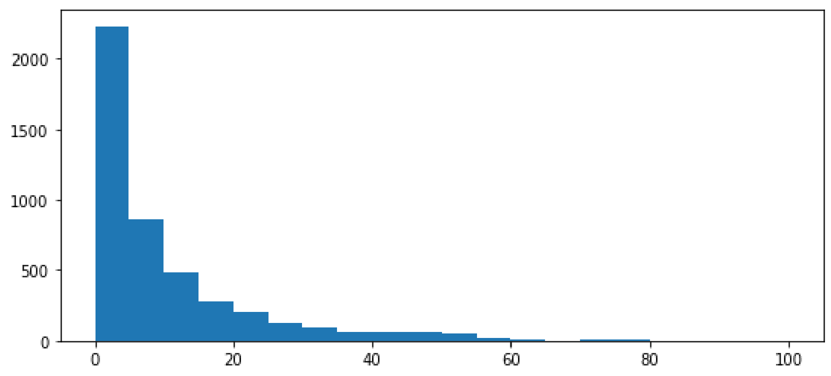
Two star:



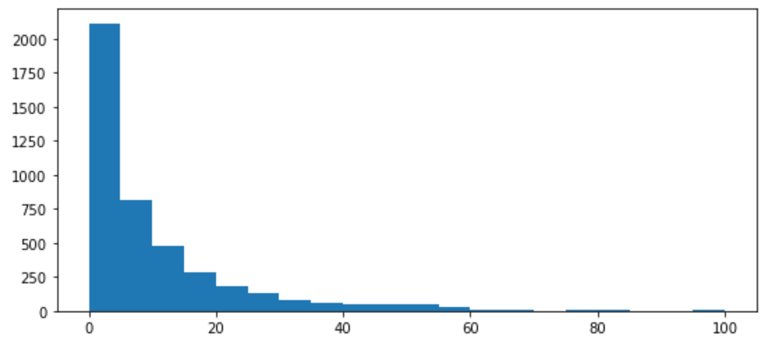
Three star:



Four star:



Five star:



The number of words in in different star ratings are quite similar. The length peaks for the bin 0-20 in all classes. It can be gathered that length of a review has no impact on the rating of the product.

CONCLUSION

The aim of our problem was to successfully classify reviews into their respective star ratings. The dataset is obtained by scraping data off e-commerce giants Amazon and Flipkart using Selenium. It is balanced and contains one feature column – Review and target column - Rating. The data was scraped keeping in mind the importance of a balanced target; therefore, there is no need for resampling the imported data. This can be confirmed by looking into the counts of each star rating.

The common words in the reviews are generated along with their frequencies with the help of Counter() and they are: good, product, quality, sound, battery, nice, price and one. Rare words are also generated and they are: inactivity, fills, manageable, lika, bad, too and cooperated. Histograms are plotted for word counts with respect to every star rating and the average word count for each class is also obtained. The average word count after tokenization ranges from nine to eleven and there is not a significant difference between classes. I.e., the length of a review does not influence the nature or the star category of a rating. Wordclouds help in recognizing the words which appear frequently. Recurrent words in each class of the target column are generated, we see that there are common loud words in all star rating which hints at the performance of the model. Multiple algorithms are fitted and validated with the help of evaluation metrics. The most suitable model for this problem comes out to be SVM classifier. The accuracy score of 0.52 cannot be considered as a good performance. The model is not able to predicted the star ratings well, it is clear from this that the data scraping phase must include much more entries with diversified background to avoid noise. Better results can be achieved by giving more importance to the web scraping phase.