Ratings Prediction Project Presentation

Submitted by
RAJESH KUMAR SINGH

INTRODUCTION

- This is a Machine Learning Project performed on customer reviews. Reviews are processed using common NLP techniques.

- Millions of people use Amazon and Flipkart to buy products. For every product, people can rate and write a review. If a product is good, it gets a positive review and gets a higher star rating, similarly, if a product is bad, it gets a negative review and lower star rating. My aim in this project is to predict star rating automatically based on the product review.

- The range of star rating is 1 to 5. That means if the product review is negative, then it will get low star rating (possibly 1 or 2), if the product is average then it will get medium star rating (possibly 3), and if the product is good, then it will get higher star rating (possibly 4 or 5).

- This task is similar to Sentiment Analysis, but instead of predicting the positive and negative sentiment (sometimes neutral also), here we need to predict the rating.

PROBLEM STATEMENT

- The rise in e-commerce has brought a significant rise in the importance of customer reviews. There are hundreds of review sites online and massive amounts of reviews for every product. Customers have changed their way of shopping and according to a recent survey, 73 percent of customers say that they use rating filters to filter out low rated items in their searches.

- The ability to successfully decide whether a review will be helpful to other customers and thus give the product more exposure is vital to companies that support these reviews, companies like Google, Amazon, Flipkart etc.

- There are two main methods to approach this problem. The first one is based on review text content analysis and uses the principles of natural language process (the NLP method). This method lacks the insights that can be drawn from the relationship between costumers and items. The second one is based on recommender systems, specifically on collaborative filtering and focuses on the reviewer's point of view.

DATA COLLECTION PHASE

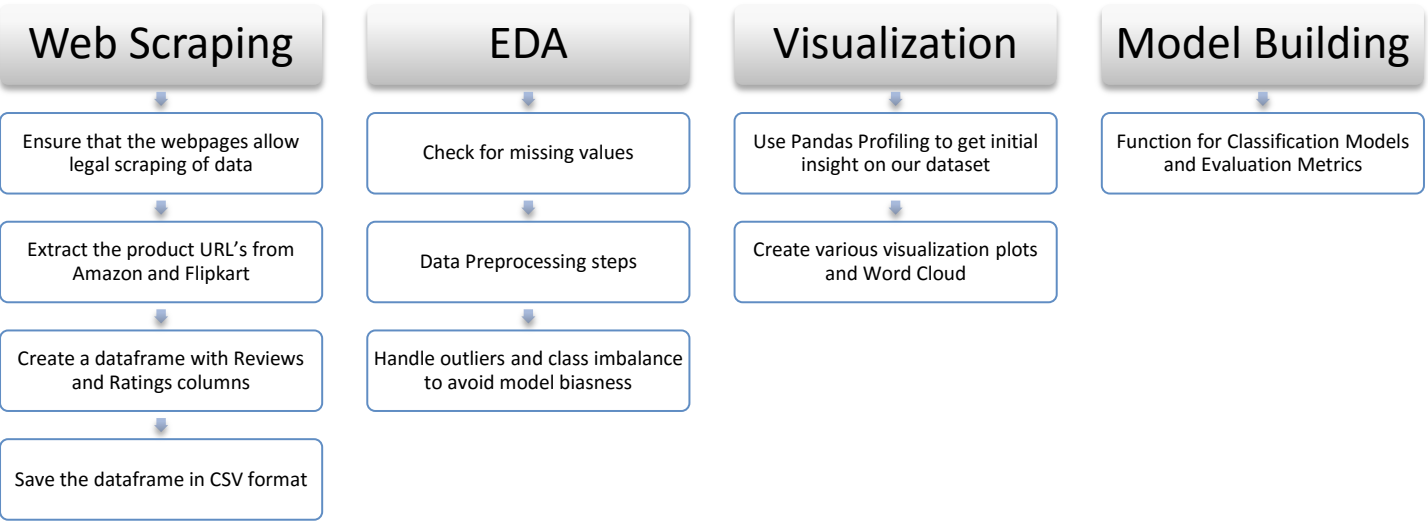
- We have to scrape at least 20000 rows of data. You can scrape more data as well, it's up to you. More the data better the model. In this section you need to scrape the reviews of different laptops, Phones, Headphones, smart watches, Professional Cameras, Printers, monitors, home theatre, router from different e-commerce websites.
- Basically, we need these columns:
    1) reviews of the product.
    2) rating of the product.
- Fetch an equal number of reviews for each rating, for example if you are fetching 10000 reviews then all ratings 1,2,3,4,5 should be 2000. It will balance our data set. Convert all the ratings to their round number as there are only 5 options for rating i.e., 1,2,3,4,5. If a rating is 4.5 convert it 5.

MODEL BUILDING PHASE

- After collecting the data, i built a machine learning model. Before model building do all data pre-processing steps involving NLP. Tried different models with different hyper parameters and select the best model. Follow the complete life cycle of data science. Include all the steps mentioned below:

    1. Data Cleaning
    2. Exploratory Data Analysis
    3. Visualization
    4. Data Pre-processing
    5. Model Building
    6. Model Evaluation
    7. Selecting the Best classification model

PROJECT FLOW



HARDWARE AND SOFTWARE USED

- Hardware Used:

  - ✓ RAM: 4 GB
  - ✓ CPU: INTEL Core i3, 1.99GHz.
  - ✓ GPU: NVIDIA GETFORCE RAM: 4 GB

- Software used.

Programming language : Python.

Distribution : Anaconda Navigator.

Browser based language shell : Jupyter Notebook.

Libraries/Packages specifically being used:

Pandas, NumPy, matplotlib, seaborn, scikit-learn, pandas-profiling, NLTK.

DATA PREPROCESSING

- Importing the necessary libraries/dependencies
- Checking dataset dimensions and null value details
- Taking a look at various label categories using the Unique method
- Performing data cleaning and then visualization steps
- Making Word Clouds for loud words in each label class
- Handling the class imbalance issue manually and fixing it
- Converting text into vectors using the TF-IDF Vectorizer
- Splitting the dataset into train and test to build classification models
- Evaluating the classification models with necessary metrics

Analytical Problem Framing

As per the client's requirement for this rating prediction project I have scraped reviews and ratings from well known e-commerce sites. This is then saved into .csv format. Also I have shared the script for web scraping into the github repository.

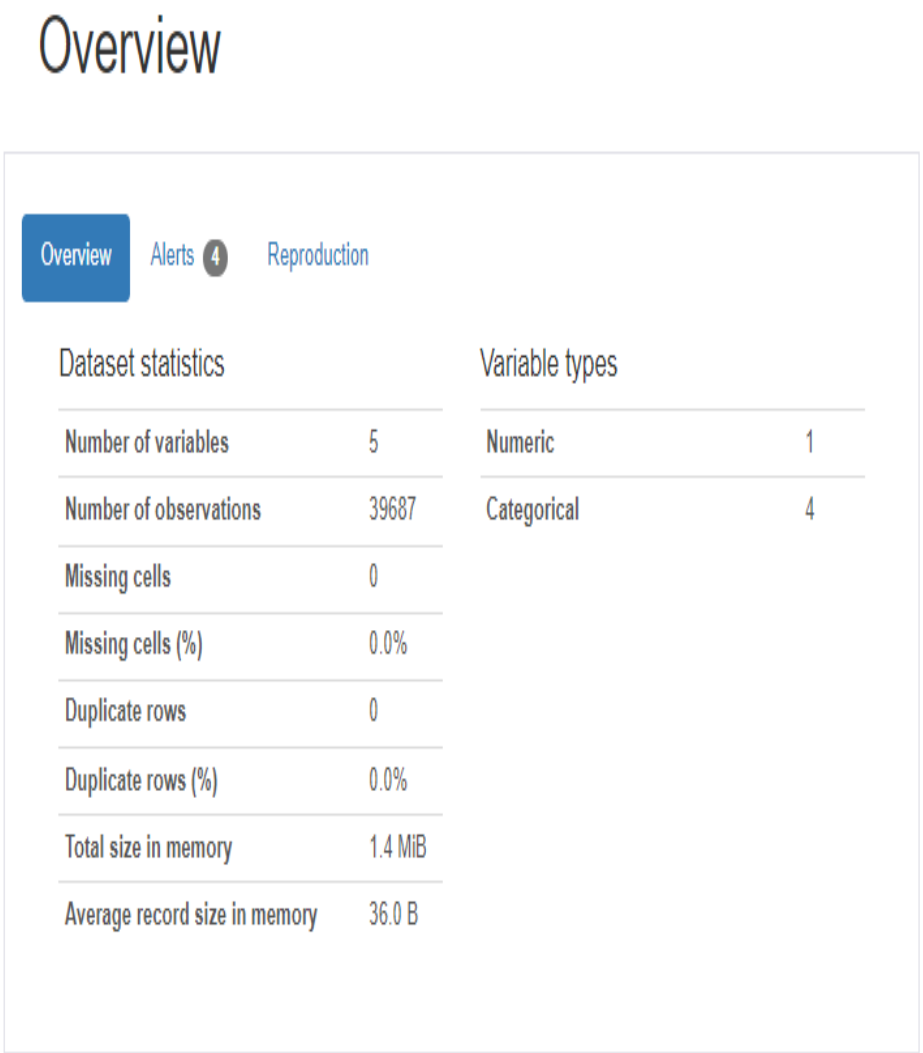
Then loaded this data into a data frame and did some of the important natural language processing steps and gone through several EDA steps to analyse the data. After all the necessary steps I have build a NLP ML model to predict the ratings.

	Unnamed: 0	Review_title	Review_text	Ratings
0	0	Display as claimed FHD is not satifactory	\n I have purchased #Lenovoldeapad3 ...	2.0 out of 5 stars
1	1	Display problem	\n Display problem.. i got delivered...	2.0 out of 5 stars
2	2	Received defective product	\n Received defective product (background noi...	2.0 out of 5 stars
3	3	No lenova product can T buy no support for ...	\n Product warranty is not support and batt...	2.0 out of 5 stars
4	4	Display quality is not good	\n The display quality is not good, the rest ...	3.0 out of 5 stars
...	...	...	...	...
41977	41977	Paisa wasool Product	I am a developer and i need multiple screen fo...	5
41978	41978	Simply awesome	Amazingly super fast delivery!! Got it for a g...	5
41979	41979	Best in the market!	Nice Monitor. Curve is perfect for movies and ...	5
41980	41980	Highly recommended	Perfect for gaming. Has a good display when us...	5
41981	41981	Terrific purchase	excellent product	5

41982 rows × 4 columns

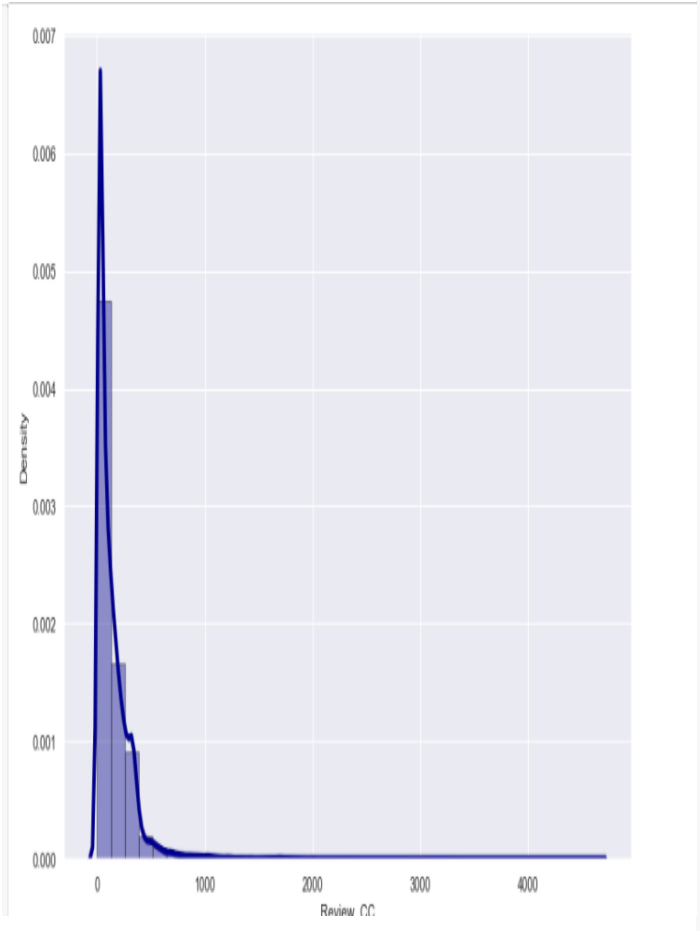
PANDAS PROFILING

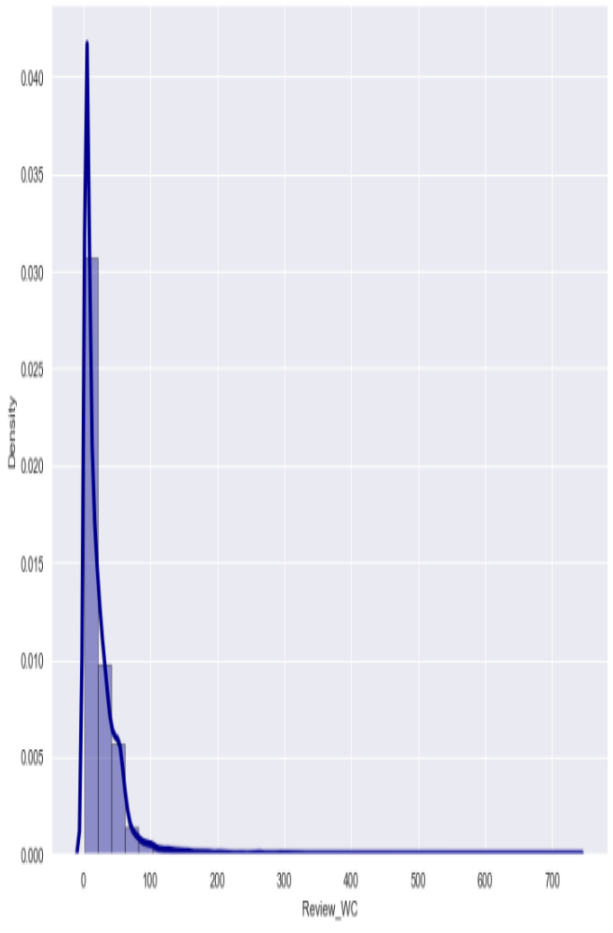
I used the pandas-profiling feature to get an insight on the initial dataset details and check out the application of all the data preprocessing steps on it.



WORD AND CHARACTER COUNT

Created the histogram + distribution plots for Word Counts and Character Counts before and after cleaning the text data. We basically removed all the stop words, punctuations, smiley, special characters, white spaces etc.





Text processing

Text Processing

23]:
```
#Here I am defining a function to replace some of the contracted words to their full form and removi
def decontracted(text):
    text = re.sub(r"won't", "will not", text)
    text = re.sub(r"don’t", "do not", text)
    text = re.sub(r"can't", "can not", text)
    text = re.sub(r"im ", "i am", text)
    text = re.sub(r"yo ", "you ",text)
    text = re.sub(r"doesn’t", "does not",text)
    text = re.sub(r"n\'t", " not", text)
    text = re.sub(r"\'re", " are", text)
    text = re.sub(r"\'s", " is", text)
    text = re.sub(r"\'d", " would", text)
    text = re.sub(r"\'ll", " will", text)
    text = re.sub(r"\'t", " not", text)
    text = re.sub(r"\'ve", " have", text)
    text = re.sub(r"\'m", " am", text)
    text = re.sub(r"<br>", " ", text)
    text = re.sub(r'http\S+', '', text) #removing urls
    return text
```

24]:
```
#Lowercasing
df['Review'] = df['Review'].apply(lambda x : x.lower())

df['Review'] = df['Review'].apply(lambda x : decontracted(x))

#removing punctuations
df['Review'] = df['Review'].str.replace('[^\w\s]','')
df['Review'] = df['Review'].str.replace('\n',' ')
```

Lemmatization:

Lemmatization is the process of grouping together the different inflected forms of a word so they can be analysed as a single item. Lemmatization is similar to stemming but it brings context to the words.

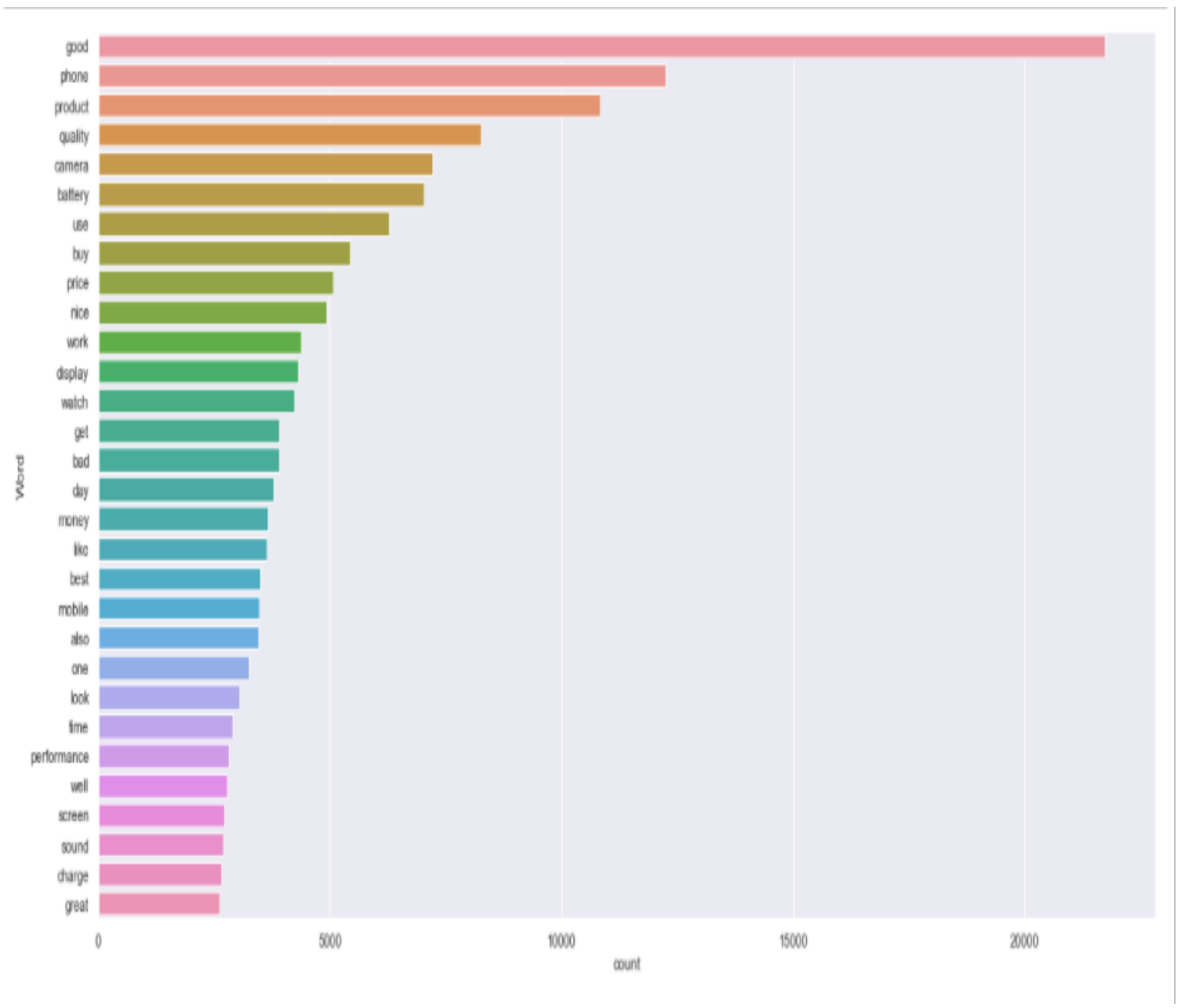
Lemmatization

[30]:
```
#Defining functiom to convert nltk tag to wordnet tags
def nltk_tag_to_wordnet_tag(nltk_tag):
    if nltk_tag.startswith('J'):
        return wordnet.ADJ
    elif nltk_tag.startswith('V'):
        return wordnet.VERB
    elif nltk_tag.startswith('N'):
        return wordnet.NOUN
    elif nltk_tag.startswith('R'):
        return wordnet.ADV
    else:
        return None
```

[31]:
```
#defining function to lemmatize our text
def lemmatize_sentence(sentence):
    #tokenize the sentence & find the pos tag
    nltk_tagged = nltk.pos_tag(nltk.word_tokenize(sentence))
    #tuple of (token, wordnet_tag)
    wordnet_tagged = map(lambda x : (x[0], nltk_tag_to_wordnet_tag(x[1])), nltk_tagged)
    lemmatize_sentence = []
    for word, tag in wordnet_tagged:
        if tag is None:
            lemmatize_sentence.append(word)
        else:
            lemmatize_sentence.append(lemmatizer.lemmatize(word,tag))
    return " ".join(lemmatize_sentence)
```

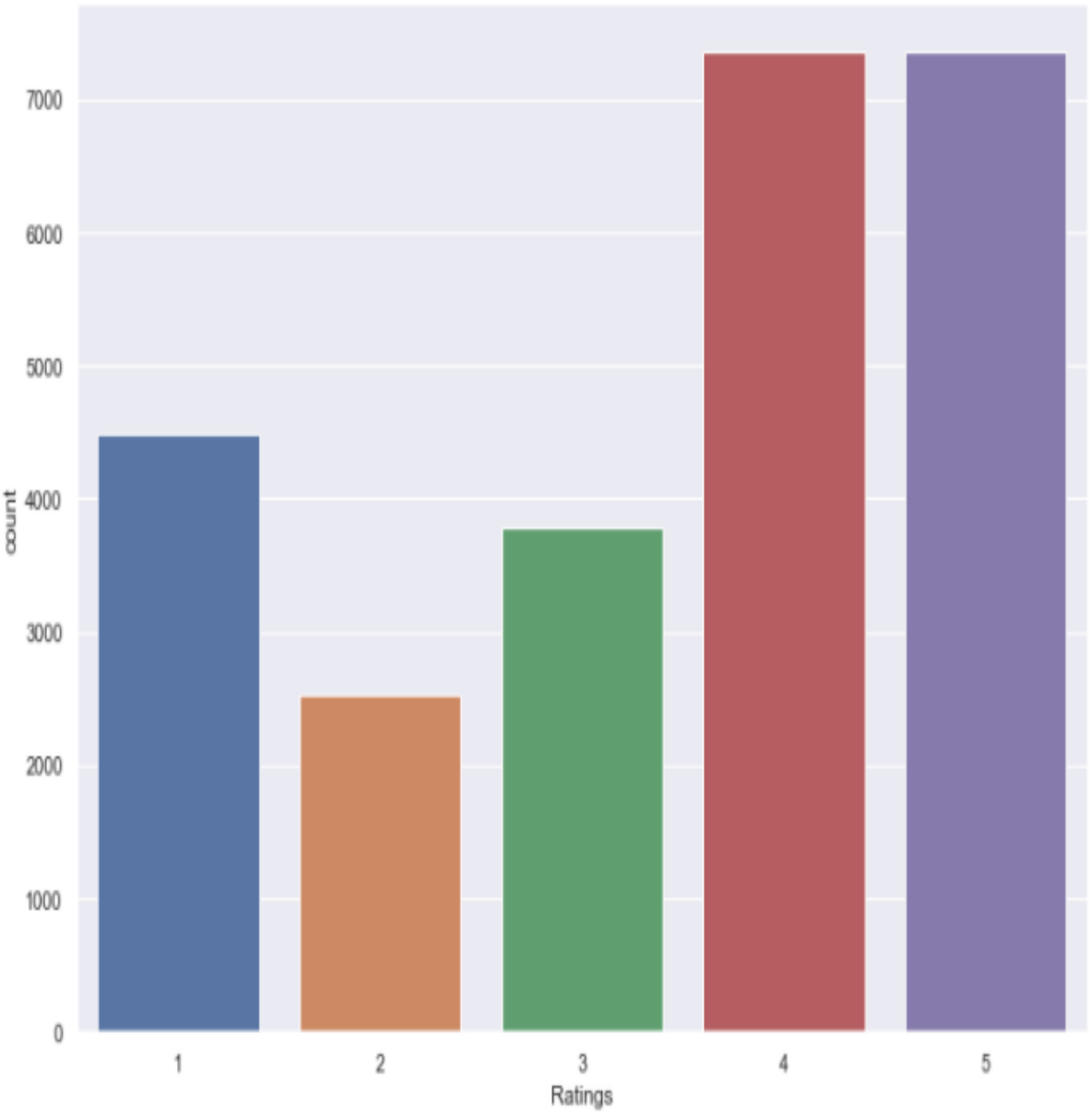
BAR PLOTS

Generated these bar plots for most frequently used words in review summary and least or rarely used words in a review summary by any customer in our dataset.



Count Plots

Generated these count plots before and after handling the data imbalance concern where we notice that the dataframe consisted of different number of rating reviews that needed to be equalized.



WORD CLOUD

Word Cloud as the name suggests is a cloud of words. It is a visualization technique for text data wherein each word is picturized with its importance in the context or its frequency.

MODEL DEVELOPMENT ALGORITHMS

The complete list of algorithms that were used in training and testing the classification model are listed below:

1. Logistic Regression
2. Linear Support Vector Classifier
3. Random Forest Classifier
4. LGBM Classifier
5. XGB Classifier

MODEL CREATION AND EVALUATION

```
In [72]: for model in [lr,svc,bnb,mnb,sgd,rf,xgb]:
             BuiltModel(model)
```

```
****************************LogisticRegression****************************
Accuracy Score: 85.46794168364947
------------------------------------------------
CLASSIFICATION REPORT :
              precision    recall  f1-score   support

           1       0.82      0.91      0.86      1117
           2       0.77      0.63      0.69       647
           3       0.80      0.76      0.78       913
           4       0.85      0.89      0.87      1856
           5       0.94      0.91      0.93      1846

    accuracy                           0.85      6379
   macro avg       0.83      0.82      0.83      6379
weighted avg       0.85      0.85      0.85      6379

Confusion Matrix :
 [[1020   38   36   18    5]
 [ 150  408   48   37    4]
 [  45   60  691  103   14]
```

Great, among all these algorithms 4 are giving good accuracies: LogisticRegression, LinearSVC, SGDClassifier and RandomForestClassifier

FINAL MODEL

Final Model

78]:
```
#training and testing our final model with above parameters
model = LinearSVC(dual = True, intercept_scaling = 2, loss = 'hinge', multi_class = 'ovr', penalty = 'l2')
model.fit(x_train,y_train) #fitting data to model
pred = model.predict(x_test)
accuracy = accuracy_score(y_test,pred)*100

#printing accuracy score
print("Accuracy Score :", accuracy)

#printing Confusion matrix
print(f"\nConfusion Matrix : \n {confusion_matrix(y_test,pred)}\n")

#printing Classification report
print(f"\nCLASSIFICATION REPORT : \n {classification_report(y_test,pred)}")
```

```
Accuracy Score : 85.71876469666093

Confusion Matrix :
 [[1007   49   34   16   11]
 [ 123  438   46   31    9]
 [  56   59  676  102   20]
 [  22   36   69 1649   80]
 [  11    3   17  117 1698]]


CLASSIFICATION REPORT :
              precision    recall  f1-score   support

           1       0.83      0.90      0.86      1117
           2       0.75      0.68      0.71       647
           3       0.80      0.74      0.77       913
           4       0.86      0.89      0.87      1856
           5       0.93      0.92      0.93      1846

    accuracy                           0.86      6379
   macro avg       0.83      0.83      0.83      6379
weighted avg       0.86      0.86      0.86      6379
```

CONCLUSION

- Key findings of the study: In this project I have collected data of reviews and ratings for different products from amazon.in and flipkart.com. Then I have done different text processing for reviews column and chose equal number of text from each rating class to eliminate problem of imbalance. By doing different EDA steps I have analyzed the text. We have checked frequently occurring words in our data as well as rarely occurring words. After all these steps I have built function to train and test different algorithms and using various evaluation metrics I have selected Random Forest Classifier as our final model. Finally by doing hyperparameter tuning we got optimum parameters for our final model. And finally we got improved accuracy score for our final model.
- Limitations of this work and scope for the future work: As we know the content of text in reviews is totally depends on the reviewer and they may rate differently which is totally depends on that particular person. So it is difficult to predict ratings based on the reviews with higher accuracies. Still we can improve our accuracy by fetching more data and by doing extensive hyperparameter tuning.

CONCLUSION

- Areas of improvement:

I. Less time complexity

II. More computational power can be given

III. More accurate reviews can be given

IV. Many more permutations and combinations in hyper parameter tuning can be used to obtain better parameter list.

- Final Remarks: After applying the hyper parameter tuning the best accuracy score obtained was 85.71876469666093% which can be further improved by obtaining more data and working up through other parameter combinations.
- We were able to create a rating prediction model that can be used to identify rating details just by evaluating the comments posted by a customer.

THANK YOU