RATINGS PROJECT

Submitted by:

ANJANA.P

ACKNOWLEDGMENT

I would like to express my appreciation to team Fliprobo for giving such a data for scraping , analysis and also to build a new model, with a full-length description of the project. My mentor Ms.Khushboo Garg has helped me in many stages of this project where I was stuck with problems. I use this opportunity to thank her for helping me at the right time without any delay.

Also, this project made me search for a lot of data's in several webpages and sites, that helped me to rectify my doubts and, I was able to study more about webscraping and NLP

INTRODUCTION

- ## Business Problem Framing

We are assigned with a new project about reviews and ratings in e-commerce sites. 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, 70 percent of customers say that they use rating filters to filter out low rated items in their searches. In this project we have to scrap ratings and reviews of different products from different online sites and with this scraped data we have to do basic Natural Language Processing based on sentiment analysis and build a new model which will predict the rating of the product based on the review of the product.

- ## Conceptual Background of the Domain Problem

We have a client who has a website where people write different reviews for technical products. Now they are adding a new feature to their website i.e. The reviewer will have to add stars(rating) as well with the review. The rating is out 5 stars and it only has 5 options available 1 star, 2 stars, 3 stars, 4 stars, 5 stars. Now they want to predict ratings for the reviews which were written in the past and they don't have a rating. So, we have to build an application which can predict the rating by seeing the review. Here we have to scrape at least 20000 rows of data. First we need to scrape the reviews of different laptops, Phones, Headphones, smart watches, Professional Cameras, Printers, monitors, Home theater, router from different e-commerce websites.
Basically, we need these columns-

1) reviews of the product.

2) rating of the product.

We have to fetch data from different websites, it will help our model to remove the effect of over fitting.

We have to fetch an equal number of reviews for each rating to balance our data set.

- # Model Building

The scraped datas from different e-commerce sites like amazon,flipkart,snapdeal etc are saved as a csv file. After collecting the data, we need to build a machine learning model. Before model building we have to do all data preprocessing steps involving Natural Language Processing steps and then We have to try out different models with different hyper parameters, evaluate their performance, and finally report the best model. We have to include all the steps like-

1. Data Cleaning
2. Exploratory Data Analysis

3. Data Preprocessing
4. Model Building
5. Model Evaluation
6. Selecting the best model

- # Review of Literature

First of all the data is scraped from different e-commerce sites like Amazon, flipkart, snapdeal etc. The main datas that are scraped includes customer reviews and ratings of different products from these above mentioned sites. And then they are saved as a csv file. Then its shape, datatypes, column value counts are all checked to get an outline of the data collected. Presence of Null values are also checked and should be cleared if any. Data visualization is done for more clarification and understanding of the data. Data is cleaned by removing all punctuations and symbols. All the cases are converted to lower and stop words are removed. Word cloud is used to display the most frequent words appeared in the news. Train test split is done and various classification models are performed and the best model which gave maximum accuracy is selected as the final model and is saved as pickle file.

- # Motivation for the Problem Undertaken

Online reviews are crucial to any online business that wants control of its reputation on the internet. Customer reviews have changed and created innovative ways for companies to market to them. Reviews allow companies to understand what their customers think of their products, service as well as themselves. A recent study has revealed that in today's web-based world, virtually everyone is reading online reviews. In fact, 91% of people read them and 84% trust them as much as they would a personal recommendation. The effects of reviews are measurable, too. The average customer is willing to spend 31% more on a retailer that has excellent reviews.

Hope this analysis may help this company to know more about customer opinions through this rating and improve themselves in future

ANALYTICAL PROBLEM FRAMING

- Mathematical/ Analytical Modeling of the Problem

In the describe function we have checked mean, std.deviation, minimum, maximum, 25 percentile, 50 percentile, 75 percentile of each attribute columns.

Mean is the average, median is the central value and mode is the frequency.
Percentile is the value below which the percentage of data falls.

We also use evaluation matrix like confusion matrix, accuracy score classification report, auc-roc. These can be expressed in mathematical formulas as:

Accuracy = TP+TN / TP+FN+FP+TN
Recall (True positive rate TPR) = TP / TP+FN
False negative rate (FNR) = FN / TP+FN
Precision =TP / TP+FP
F1 Score = precision – Recall / Precision +Recall
Specificity = TN / FP+TN

- Data Sources and their formats

FlipRobo technologies have provided this opportunity to scrape datas from different online sites and save them as dataset for detailed analysis and build a model to predict the rating of a product based on reviews from customers. Different e-commerce sites like amazon and flipkart are chosen to scrape reviews and ratings of different products.20000 datas are collected in total with 4000 datas each for 1star,2star,3star,4star and 5star ratings.Then the data scraped was saved as a csv file and loaded in Jupyter notebook first. There are 2 columns ,one for rating and one for review. The data was in object datatypes. Reviews are text data and ratings are numbering from 1 to 5 where 1 is the worst and 5 is the best review. An output column named 'label' is added to it , which is filled with only 0's and 1 value.As we are doing sentiment analysis, it is important to tell our model what is positive sentiment and what is a negative sentiment. if you think you will put 3 in the good review slot. We will denote positive sentiments as 1 and negative sentiments as 0. Our label column returns 1 if the rating is 3 or more else return 0 that will represent the positive and negative sentiment as 1 or 0. We have to test various models and select the more accurate one to predict the ratings from reviews

- Data Scraping And Preprocessing Done

First the important libraries for scraping is imported.Then chrome driver is activated and e-commerce site to scrape data is opened through link.Then product names and search buttons are activated through commands using their x_paths.Empty lists to scrape rating and reviews are created and reviews and ratings of that products are scraped using their x_paths from

various pages.Like wise datas for different products are scraped from amazon and flipkart till 20000 datas are scraped with 4000 each for 1star, 2star, 3star, 4star and 5star.And atlast all these datas are saved as csv file.

For preprocessing also important libraries and csv file is imported. Shape and datatypes of columns are checked. There are 20000 rows and 2 columns in our dataset. They are object datatypes. Unnecessary column like index is dropped as they provide no necessary information for our analysis. Null values are checked and should be cleared if there is any. Extra columns for original length of reviews and cleanlength after removal of stop words is added.Rating column is converted to integer datatype from string datatype .Then a new column named label is added to classify the reviews as good and bad.All the ratings equal to or above 3 are considered as good reviews and is represented as 1 and ratings below 3 are considered as bad reviews and are represented as 0. Review column data is converted to lower case and also all the punctuations and stop words are removed from it.Word cloud is used to display the most frequent words present both in good and bad reviews based on label column.The data needs to be converted into a format that can be interpreted by a machine learning algorithm since these algorithms donot work well with textual data. Hence we need to convert it into a form that will enable the algorithm to discern patterns and meaningful insights from the data. Inorder to achieve this I used Tfidf vectorizer. Then maximum r_state is found and then various classification models like multinomial NB, Randomforest classifier, decision tree classifier,knearest neighbor ,svc etc are found and the best model is saved as pickle file.

- Data Inputs- Logic- Output Relationships

There are two index columns and one is deleted.Shape of the dataset is 20000 rows and 2 columns.All the columns are object datatypes.Datatype of rating column should be changed to integer datatype to add a label column with values 0 and 1.There are no null values in our dataset. All the data in review column are converted to lowercases and also punctuations and stop words are removed from them as part of data cleaning.

1) NLTK library-stopwords are downloaded from nltk library.Stopwords are English words which doesnot add much meaning to a sentence. stopwords are also removed from our dataset .List of stopwords can be found in corpus module.To remove the stopwords from a data first divide the text into words and then remove the word if it exists in the list of stopwords provided by NLTK.

Porter stemmer- stemmers remove morphological affixes from words leaving only the word stem.It extracts the base of a modified word.so the efficiency of any content based spam filter can be significantly improved.
Corpus- it is a language resource consisting of large and structured set of texts.

- Hardware and Software Requirements and Tools Used

I used intel core i3 processor, 4GB RAM and 64 bit operating system as hardware and windows 10, MS excel, MS word and python 3 Jupyter notebook as software for the

completion of this project. In jupyter notebook various libraries are also used. They include selenium, webdriver, time, request, pandas, numpy, matplotlib , seaborn , imblearn ,wordcloud and sklearn.

MODEL/S DEVELOPMENT AND EVALUATION

- Identification of possible problem-solving approaches (methods)
  The major problems we dealt with this dataset are
- Unlike other projects here the datas to be analysed are scraped first from different websites.
- Minimum 20000 datas with 4000 each of 1star,2star,3star,4star,and 5star are to be scraped to balance our dataset. This was really time consuming step.
- Data should be fetched from different websites. If data is from different websites, it will help our model to remove the effect of over fitting.
- 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 to 5.

- Testing of Identified Approaches (Algorithms)

After removing punctuations and stop words Tf-idf vectoriser is used to convert the text into machine learning algorithm format.Tf-Idf is also known as Term Frequency –Inverse document frequency. It gives us a way to associate each word in a document with a number that represents how relevant each word is in that document. With Tf-Idf, instead of representing a term in a document by its raw frequency (number of occurrences) or its relative frequency (term count divided by document length), each term is weighted by dividing the term frequency by the number of documents in the corpus containing the word. The overall effect of this weighting scheme is to avoid a common problem when conducting text analysis: the most frequently used words in a document are often the most frequently used words in all of the documents. In contrast, terms with the highest Tf-Idf scores are the terms in a document that are distinctively frequent in a document, when that document is compared to other documents. I used TfidfVectorizer from the sklearn library to convert the text into a sparse matrix. This matrix represents the Tf-Idf values for all the words present in my data. The training and test data are now represented by the variables x and y.Since I now have the data ready for implementing the machine learning algorithm, I move to the next step which includes fitting my machine learning algorithm on the training data.

- Run and Evaluate selected models

  1)for loop used for scrapping different stars reviews and ratings

```
for url in urls[:]:
    driver.get(url)                                                    # Loading the webpage by url
    time.sleep(2)
    try:
        pop = driver.find_element_by_xpath('//a[@class="a-popover-trigger a-declarative"]/i[2]')     # Button for expanding the s
        pop.click()
        time.sleep(2)

        rev=driver.find_element_by_xpath("//div[@class='a-section a-spacing-base a-text-center']/a") #button for expanding reviev
        rev.click()
        time.sleep(2)

        star5=driver.find_element_by_xpath("//table[@id='histogramTable']/tbody/tr[1]")  #Locating the 5star ratings
        star5.click()

        rating=driver.find_element_by_xpath("//div[@class='a-row a-spacing-micro a-size-base']/span[1]") #scrape rating
        Rating.append(rating.text)  #appending the ratings in Ratings list
        time.sleep(1)

        review=driver.find_element_by_xpath("//a[@class='a-size-base a-link-normal review-title a-color-base review-title-content
        Review.append(review.text)  #appending the review in Review list

        star4=driver.find_element_by_xpath("//table[@id='histogramTable']/tbody/tr[2]")  #Locating the 4star rating
        star4.click()

        rating=driver.find_element_by_xpath("//div[@class='a-row a-spacing-micro a-size-base']/span[1]") #scrape rating
        Rating.append(rating.text)  #appending the ratings in Ratings list
        time.sleep(1)
```

```
        review=driver.find_element_by_xpath("//a[@class='a-size-base a-link-normal review-title a-color-base review-title-content
        Review.append(review.text)   #appending the review in Review list

        star3=driver.find_element_by_xpath("//table[@id='histogramTable']/tbody/tr[3]")  #Locating the 3star rating
        star3.click()

        rating=driver.find_element_by_xpath("//div[@class='a-row a-spacing-micro a-size-base']/span[1]") #scrape rating
        Rating.append(rating.text)  #appending the ratings in Ratings list
        time.sleep(1)

        review=driver.find_element_by_xpath("//a[@class='a-size-base a-link-normal review-title a-color-base review-title-content
        Review.append(review.text)   #appending the review in Review list

        star2=driver.find_element_by_xpath("//table[@id='histogramTable']/tbody/tr[4]")  #Locating the 2star rating
        star2.click()

        rating=driver.find_element_by_xpath("//div[@class='a-row a-spacing-micro a-size-base']/span[1]") #scrape rating
        Rating.append(rating.text)  #appending the ratings in Ratings list
        time.sleep(1)

        review=driver.find_element_by_xpath("//a[@class='a-size-base a-link-normal review-title a-color-base review-title-content
        Review.append(review.text)   #appending the review in Review list
        time.sleep(2)

        star1=driver.find_element_by_xpath("//table[@id='histogramTable']/tbody/tr[5]")  #Locating the 1star rating
        star1.click()

        rating=driver.find_element_by_xpath("//div[@class='a-row a-spacing-micro a-size-base']/span[1]") #scrape rating
        Rating.append(rating.text)  #appending the ratings in Ratings list
        time.sleep(1)
```

2)Maximum accuracy score corresponding to r_state is found using for loop.

```
from sklearn.metrics import accuracy_score
from sklearn.tree import DecisionTreeClassifier
dtc=  DecisionTreeClassifier()

max_score=0
for r_state in range(40,100):
    x_train,x_test,y_train,y_test=train_test_split(x,y,random_state=r_state,test_size=0.22)

    dtc.fit(x_train,y_train)
    y_pred=dtc.predict(x_test)
    acc_scr=accuracy_score(y_test,y_pred)
    if acc_scr>max_score:
        max_score=acc_scr
        final_r_state=r_state
print("max accuracy score corresponding to",final_r_state,"is",max_score)
```

```
max accuracy score corresponding to 94 is 0.49613636363636365
```

I got maximum accuracy score as 0.49 and the corresponding r_state is 94.

3) Multinomial Naïve Bayes-It is a variant Naïve Bayes that follows Multinomial normal distribution and supports continuous data.

```
mnb=MultinomialNB()
mnb.fit(x_train,y_train)
predmnb=mnb.predict(x_test)
print(accuracy_score(y_test,predmnb))
print(confusion_matrix(y_test,predmnb))
print(classification_report(y_test,predmnb))
```

```
0.45886363636363636
[[471  73  75  79 194]
 [131 271 140 116 188]
 [101  44 424 112 198]
 [121  71  70 328 287]
 [132  45  50 154 525]]
              precision    recall  f1-score   support

           1       0.49      0.53      0.51       892
           2       0.54      0.32      0.40       846
           3       0.56      0.48      0.52       879
           4       0.42      0.37      0.39       877
           5       0.38      0.58      0.46       906

    accuracy                           0.46      4400
   macro avg       0.48      0.46      0.46      4400
weighted avg       0.48      0.46      0.46      4400
```

Multinomial NB is giving an accuracy score of 0.45

4)

```
svc=SVC(kernel='rbf')
svc.fit(x_train,y_train)
svc.score(x_train,y_train)
predsvc=svc.predict(x_test)
print(accuracy_score(y_test,predsvc))
print(confusion_matrix(y_test,predsvc))
print(classification_report(y_test,predsvc))
```

```
0.4990909090909091
[[530  21  67  62 212]
 [150 220 145 130 201]
 [ 44  36 470 119 210]
 [ 88  55  79 367 288]
 [ 77  28  50 142 609]]
              precision    recall  f1-score   support

           1       0.60      0.59      0.60       892
           2       0.61      0.26      0.36       846
           3       0.58      0.53      0.56       879
           4       0.45      0.42      0.43       877
           5       0.40      0.67      0.50       906

    accuracy                           0.50      4400
   macro avg       0.53      0.50      0.49      4400
weighted avg       0.53      0.50      0.49      4400
```

5)

```
svc=SVC(kernel='poly')
svc.fit(x_train,y_train)
svc.score(x_train,y_train)
predsvc=svc.predict(x_test)
print(accuracy_score(y_test,predsvc))
print(confusion_matrix(y_test,predsvc))
print(classification_report(y_test,predsvc))
```

```
0.49863636363636366
[[539  23  72  60 198]
 [154 225 151 118 198]
 [ 43  38 472 103 223]
 [ 81  60  83 360 293]
 [ 79  36  57 136 598]]
              precision    recall  f1-score   support

           1       0.60      0.60      0.60       892
           2       0.59      0.27      0.37       846
           3       0.57      0.54      0.55       879
           4       0.46      0.41      0.44       877
           5       0.40      0.66      0.50       906

    accuracy                           0.50      4400
   macro avg       0.52      0.50      0.49      4400
weighted avg       0.52      0.50      0.49      4400
```

6)

```
def svmkernel(ker):
    svc=SVC(kernel=ker)
    svc.fit(x_train,y_train)
svc.score(x_train,y_train)
predsvc=svc.predict(x_test)
print(accuracy_score(y_test,predsvc))
print(confusion_matrix(y_test,predsvc))
print(classification_report(y_test,predsvc))
```

```
0.49863636363636366
[[539  23  72  60 198]
 [154 225 151 118 198]
 [ 43  38 472 103 223]
 [ 81  60  83 360 293]
 [ 79  36  57 136 598]]
              precision    recall  f1-score   support

           1       0.60      0.60      0.60       892
           2       0.59      0.27      0.37       846
           3       0.57      0.54      0.55       879
           4       0.46      0.41      0.44       877
           5       0.40      0.66      0.50       906

    accuracy                           0.50      4400
   macro avg       0.52      0.50      0.49      4400
weighted avg       0.52      0.50      0.49      4400
```

7) Decision tree classifier – A tree structure is constructed that breaks the dataset into smaller subsets eventually resulting in prediction.The root node partitions the data based on most influential feature partitioning.There are two measures for this. They are gini impurity and Entropy.

```
dtc=DecisionTreeClassifier()
dtc.fit(x_train,y_train)
dtc.score(x_train,y_train)
preddtc=dtc.predict(x_test)
print(accuracy_score(y_test,preddtc))
print(confusion_matrix(y_test,preddtc))
print(classification_report(y_test,preddtc))
```

```
0.49613636363636365
[[597  28  71  50 146]
 [169 248 166 108 155]
 [ 62  61 483 113 160]
 [120  95 100 344 218]
 [113  68  75 139 511]]
              precision    recall  f1-score   support

           1       0.56      0.67      0.61       892
           2       0.50      0.29      0.37       846
           3       0.54      0.55      0.54       879
           4       0.46      0.39      0.42       877
           5       0.43      0.56      0.49       906

    accuracy                           0.50      4400
   macro avg       0.50      0.49      0.49      4400
weighted avg       0.50      0.50      0.49      4400
```

8) K Nearest Neighbor-It is found that two neighbors who have identical distance but different labels,the result will depend on the ordering of the training data.

```
knn=KNeighborsClassifier()
knn.fit(x_train,y_train)
knn.score(x_train,y_train)
predknn=knn.predict(x_test)
print(accuracy_score(y_test,predknn))
print(confusion_matrix(y_test,predknn))
print(classification_report(y_test,predknn))
```

```
0.41363636363636364
[[515 134  68  70 105]
 [201 321 129 106  89]
 [118 158 397 114  92]
 [201 134 110 297 135]
 [263  99  82 172 290]]
              precision    recall  f1-score   support

           1       0.40      0.58      0.47       892
           2       0.38      0.38      0.38       846
           3       0.51      0.45      0.48       879
           4       0.39      0.34      0.36       877
           5       0.41      0.32      0.36       906

    accuracy                           0.41      4400
   macro avg       0.42      0.41      0.41      4400
weighted avg       0.42      0.41      0.41      4400
```

9) Random Forest Classifier –Random forest is a meta estimator that fits a number of decision tree classifiers on various sub samples of the dataset and uses averaging to improve the predictive accuracy and control overfitting.

```
#Random forest classifier
from sklearn.ensemble import RandomForestClassifier
rf=RandomForestClassifier(n_estimators=20,random_state=94)
rf.fit(x_train,y_train)
print('score:',rf.score(x_train,y_train))
predrf=rf.predict(x_test)
print('\n')
print("Accuracy score :",accuracy_score(y_test,predrf))
print(confusion_matrix(y_test,predrf))
print(classification_report(y_test,predrf))
```

```
score: 0.5942948717948718


Accuracy score : 0.49727272727272726
[[523  69  70  57 173]
 [118 289 153 120 166]
 [ 47  57 471 112 192]
 [ 99  78  92 365 243]
 [100  49  61 156 540]]
              precision    recall  f1-score   support

           1       0.59      0.59      0.59       892
           2       0.53      0.34      0.42       846
           3       0.56      0.54      0.55       879
           4       0.45      0.42      0.43       877
           5       0.41      0.60      0.49       906

    accuracy                           0.50      4400
   macro avg       0.51      0.50      0.49      4400
weighted avg       0.51      0.50      0.49      4400
```

Random Forest Classifier gives accuracy -0.59

- Key Metrics for success in solving problem under consideration

Using MultinomialNB, Decision Tree classifier,Random Forest classifier knearest neighbour,svc etc Random forest classifier is giving maximum accuracy score of 59%.So its cross validation score is checked.

```
from sklearn.model_selection import cross_val_score
rf=RandomForestClassifier()
score=cross_val_score(rf,x,y,cv=5)
print("score:",score)
print("Mean score:",score.mean())
print("Standard deviation:",score.std())
```

```
score: [0.28875 0.22475 0.42775 0.508   0.492  ]
Mean score: 0.38825000000000004
Standard deviation: 0.11251388803165589
```

After checking cross_val_score The model is saved as pickle file as it is the best model.

Visualizations

```
sns.countplot(d1["Rating"])
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x2b7462bf0a0>
```

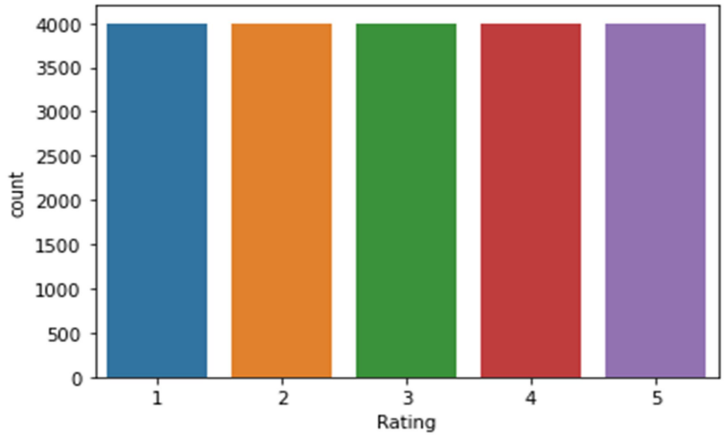

Fig:1 shows the countplot of ratings.

All the ratings have 4000 datas each

```
#Number of characters present in each rating
d1['Review'].str.len().hist(by=d1['Rating'])
```

```
array([[<matplotlib.axes._subplots.AxesSubplot object at 0x000002B7477FD760>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x000002B74782A5E0>],
       [<matplotlib.axes._subplots.AxesSubplot object at 0x000002B747857A30>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x000002B747886E80>],
       [<matplotlib.axes._subplots.AxesSubplot object at 0x000002B7478BE340>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x000002B7478E87F0>]],
      dtype=object)
```

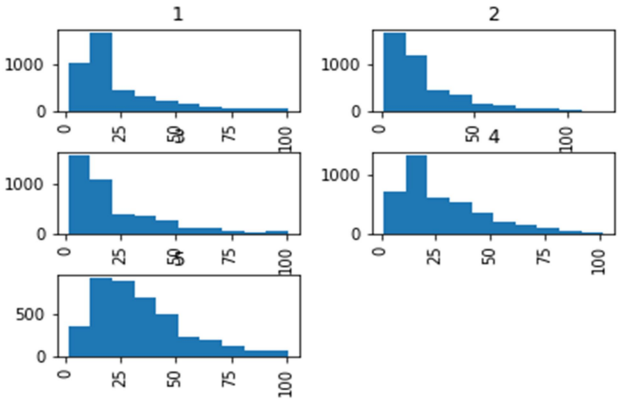

Fig:2 shows the strength of review column with each star rating.

Fig:3 shows the most frequent words appeared in good review comments.

Fig:4 shows the most frequent words appeared in bad review comments

Interpretation of the Results

The reviews after removing punctuations and stopwords are converted to vector form to make it machine readable and then they are splitted to train and test and various classification models are performed to predict the ratings of product based on its review. We tested multinomial naïve baes, decision tree classifier, random forest classifier , knearest neighbor ,svc etc. We get maximum accuracy for random forest classifier and we used it as our selected model.

CONCLUSION

- ## Key Findings and Conclusions of the Study

We used Natural Language Process method to solve this problem. This simple sentiment analysis classifier can be useful in many types of datasets. It can be used in real-world projects and businesses as well. The dataset we used here resembles a real business dataset. 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!. to approach this problem. This project is based on review text content analysis and uses the principles of natural language process (the NLP method).

This is project is a small approach to predict the ratings from reviews.We collected the data samples from various online sites and first of all the data is cleaned and preprocessed. Then various classification models are performed and found that Random forest classifier gives the best accuracy score.

- ## Learning Outcomes of the Study in respect of Data Science

I have tried various models for this dataset. They are multinomial Nb ,KNN and decision tree classifer,svc etc. From this Random forest classifier gives a maximum score of 0.59. cross_val_score of Random forest classifier is checked Since it gives the best score, I suggest this model for my dataset and thus saved random forest classifier as my model.

- ## Limitations of this work and Scope for Future Work

- Data scraping was really a tough task as we need to scrape more than 20000 datas of different products from different online sites.
- Execution of programme was not easy as it took a lot of time for execution.
- There were also a lot of problems during each phase of the project, they were all resolved by searching webpages and also with the help of mentor and datatrained support team.