FLIP ROBO

Ratings Prediction

Submitted by:

Dipak Someshwar

ACKNOWLEDGMENT

I have taken efforts in this project. However, it would not have been possible without the kind support and help of many individuals. We would like to extend my sincere thanks to SME. Sapna Verma.

We are highly indebted to Flip Robo technology for their guidance and constant supervision as well as for providing necessary information regarding the project & also for their support in completing the project.

I thank and appreciations also go to our colleague in developing the project and people who have willingly helped us out with their abilities.

Thanks all.

Dipak Someshwar

INTRODUCTION

- 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.

- This project contains two phase:

1. **Data Collection Phase:**

   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 ecommerce websites.

   Basically, we need these columns

   1) reviews of the product.

   2) rating of the product.

   You can fetch other data as well, if you think data can be useful or can help in the project. It completely depends on your imagination or assumption.

2. Model Building Phase:

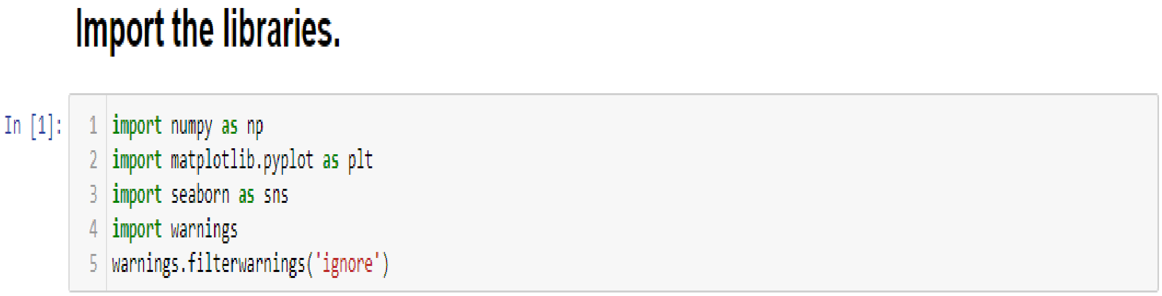
After collecting the data, build a machine learning model. Before model building do all data pre-processing steps involving NLP.

Try different models with different hyper parameters and select the best model.

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

Analytical Problem Framing

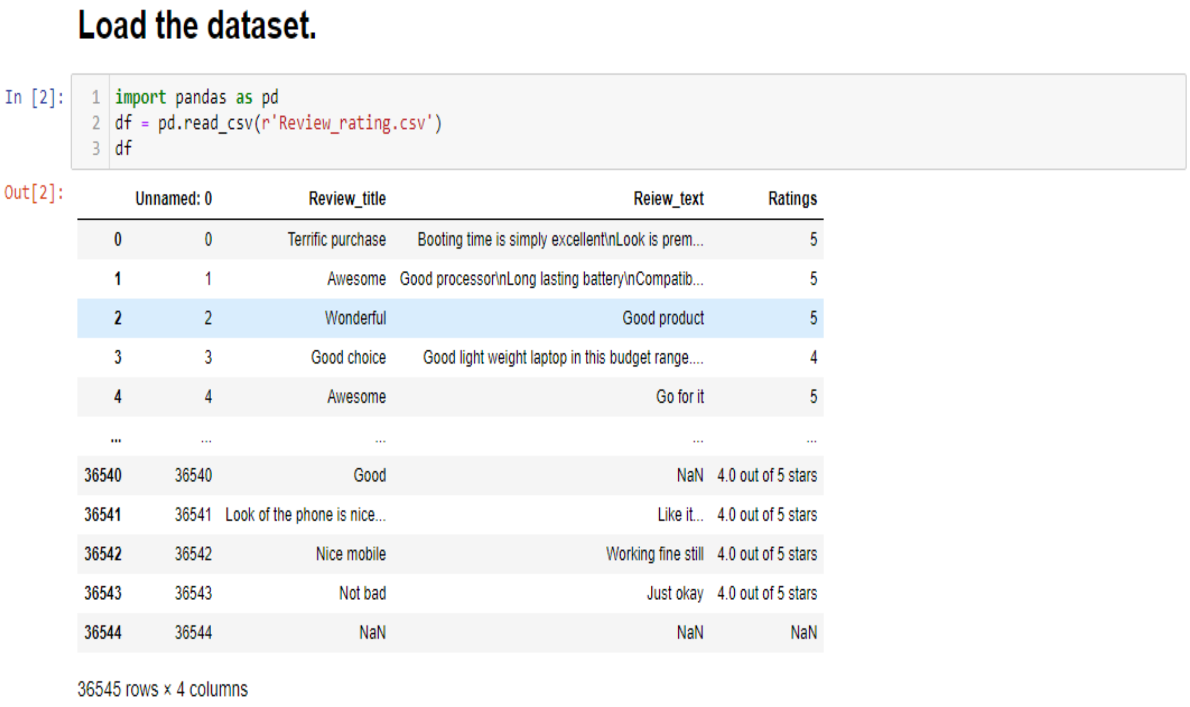
- Import library and load the dataset.



Import the libraries.

In [1]:

```python
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings('ignore')
```



Load the dataset.

In [2]:

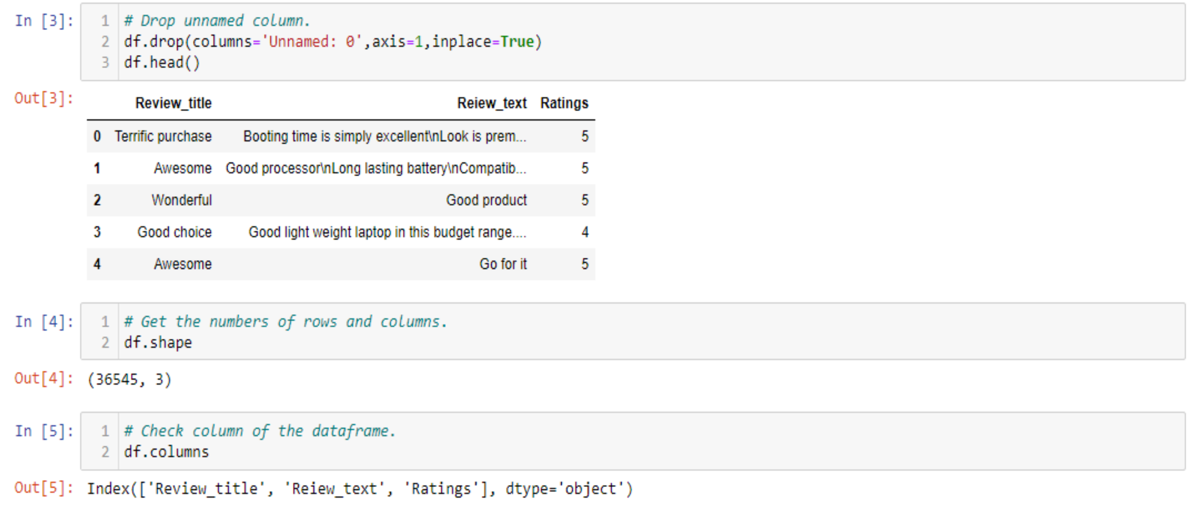
```python
import pandas as pd
df = pd.read_csv(r'Review_rating.csv')
df
```

Out[2]:

	Unnamed: 0	Review_title	Reiew_text	Ratings
0	0	Terrific purchase	Booting time is simply excellent\nLook is prem...	5
1	1	Awesome	Good processor\nLong lasting battery\nCompatib...	5
2	2	Wonderful	Good product	5
3	3	Good choice	Good light weight laptop in this budget range....	4
4	4	Awesome	Go for it	5
...	...	...	...	...
36540	36540	Good	NaN	4.0 out of 5 stars
36541	36541	Look of the phone is nice...	Like it...	4.0 out of 5 stars
36542	36542	Nice mobile	Working fine still	4.0 out of 5 stars
36543	36543	Not bad	Just okay	4.0 out of 5 stars
36544	36544	NaN	NaN	NaN

36545 rows × 4 columns

- Drop unnamed column and display all column name of dataset.



In [3]:
```
# Drop unnamed column.
df.drop(columns='Unnamed: 0',axis=1,inplace=True)
df.head()
```

Out[3]:

	Review_title	Reiew_text	Ratings
0	Terrific purchase	Booting time is simply excellent\nLook is prem...	5
1	Awesome	Good processor\nLong lasting battery\nCompatib...	5
2	Wonderful	Good product	5
3	Good choice	Good light weight laptop in this budget range....	4
4	Awesome	Go for it	5

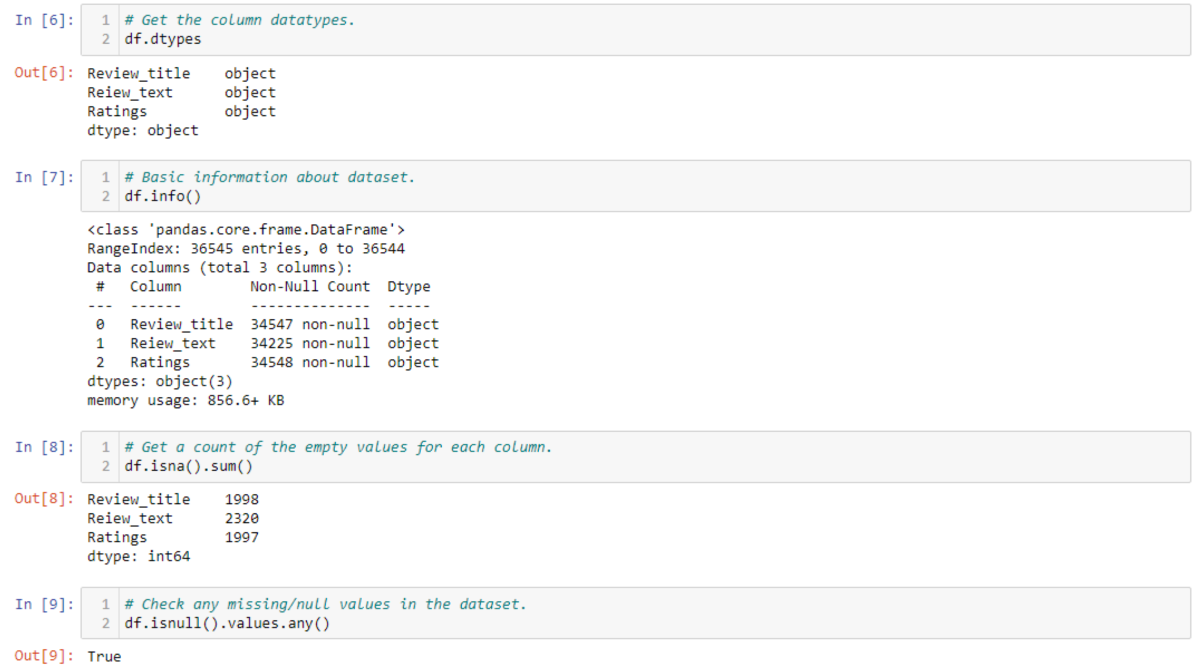
In [4]:
```
# Get the numbers of rows and columns.
df.shape
```

Out[4]:
```
(36545, 3)
```

In [5]:
```
# Check column of the dataframe.
df.columns
```

Out[5]:
```
Index(['Review_title', 'Reiew_text', 'Ratings'], dtype='object')
```

- Display datatypes, basic info and sum of null values.



In [6]:
```
# Get the column datatypes.
df.dtypes
```

Out[6]:
```
Review_title    object
Reiew_text      object
Ratings         object
dtype: object
```

In [7]:
```
# Basic information about dataset.
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 36545 entries, 0 to 36544
Data columns (total 3 columns):
 #   Column        Non-Null Count  Dtype 
---  ------        --------------  ----- 
 0   Review_title  34547 non-null  object
 1   Reiew_text    34225 non-null  object
 2   Ratings       34548 non-null  object
dtypes: object(3)
memory usage: 856.6+ KB
```

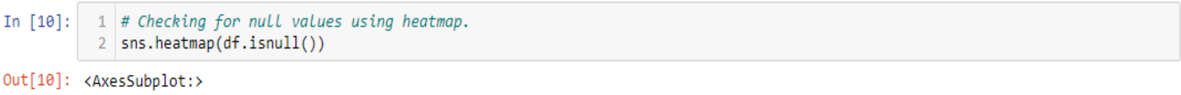
In [8]:
```
# Get a count of the empty values for each column.
df.isna().sum()
```

Out[8]:
```
Review_title    1998
Reiew_text      2320
Ratings         1997
dtype: int64
```

In [9]:
```
# Check any missing/null values in the dataset.
df.isnull().values.any()
```

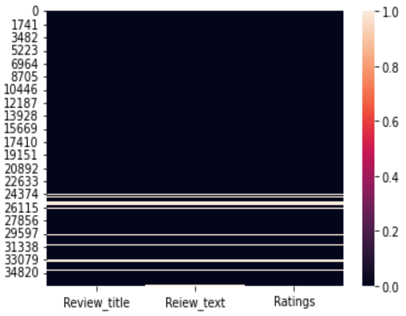
Out[9]:
```
True
```

- Display null values of columns using heatmap.

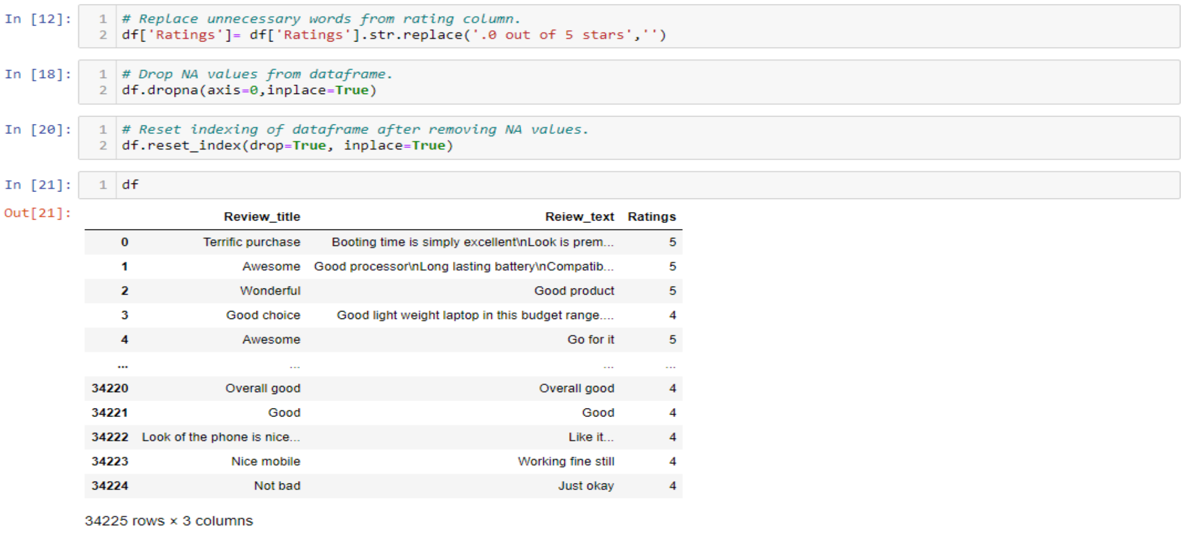


In [10]:
```
# Checking for null values using heatmap.
sns.heatmap(df.isnull())
```

Out[10]:
```
<AxesSubplot:>
```



- Perform some feature engineering tech



In [12]:
```
# Replace unnecessary words from rating column.
df['Ratings']= df['Ratings'].str.replace('.0 out of 5 stars','')
```

In [18]:
```
# Drop NA values from dataframe.
df.dropna(axis=0,inplace=True)
```

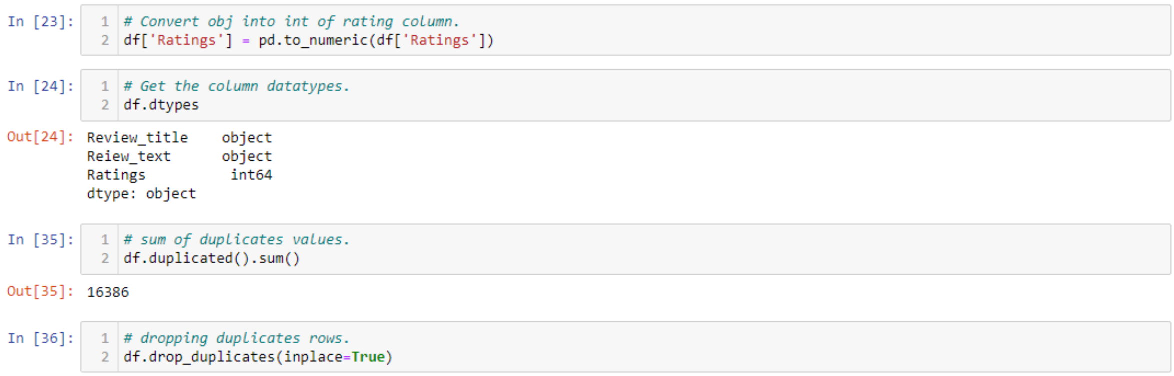
In [20]:
```
# Reset indexing of dataframe after removing NA values.
df.reset_index(drop=True, inplace=True)
```

In [21]:
```
df
```

Out[21]:

	Review_title	Reiew_text	Ratings
0	Terrific purchase	Booting time is simply excellent\nLook is prem...	5
1	Awesome	Good processor\nLong lasting battery\nCompatib...	5
2	Wonderful	Good product	5
3	Good choice	Good light weight laptop in this budget range....	4
4	Awesome	Go for it	5
...	...	...	...
34220	Overall good	Overall good	4
34221	Good	Good	4
34222	Look of the phone is nice...	Like it...	4
34223	Nice mobile	Working fine still	4
34224	Not bad	Just okay	4

34225 rows × 3 columns



```
In [23]: # Convert obj into int of rating column.
         df['Ratings'] = pd.to_numeric(df['Ratings'])
```

```
In [24]: # Get the column datatypes.
         df.dtypes
```

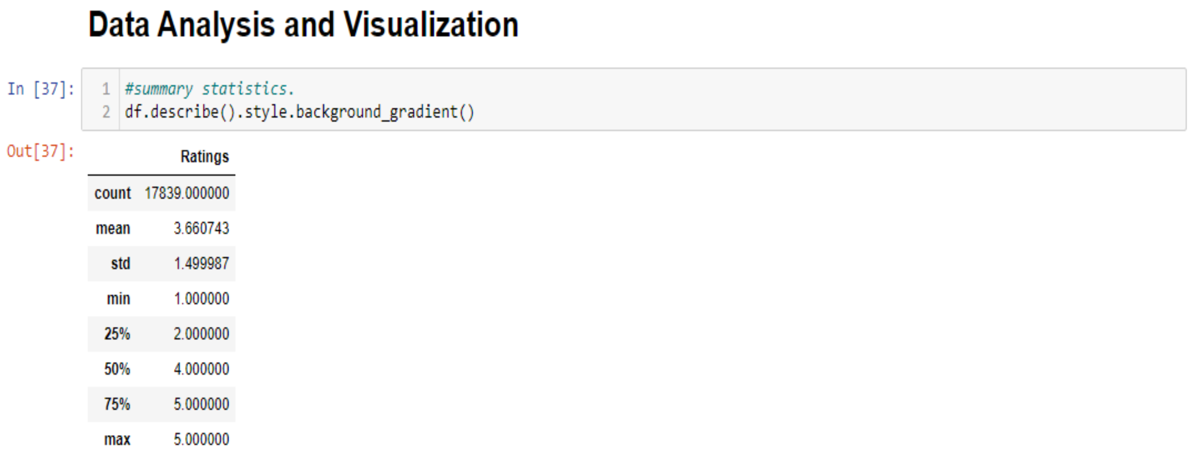
```
Out[24]: Review_title    object
         Reiew_text      object
         Ratings          int64
         dtype: object
```

```
In [35]: # sum of duplicates values.
         df.duplicated().sum()
```

```
Out[35]: 16386
```

```
In [36]: # dropping duplicates rows.
         df.drop_duplicates(inplace=True)
```

- Display statistical summary.



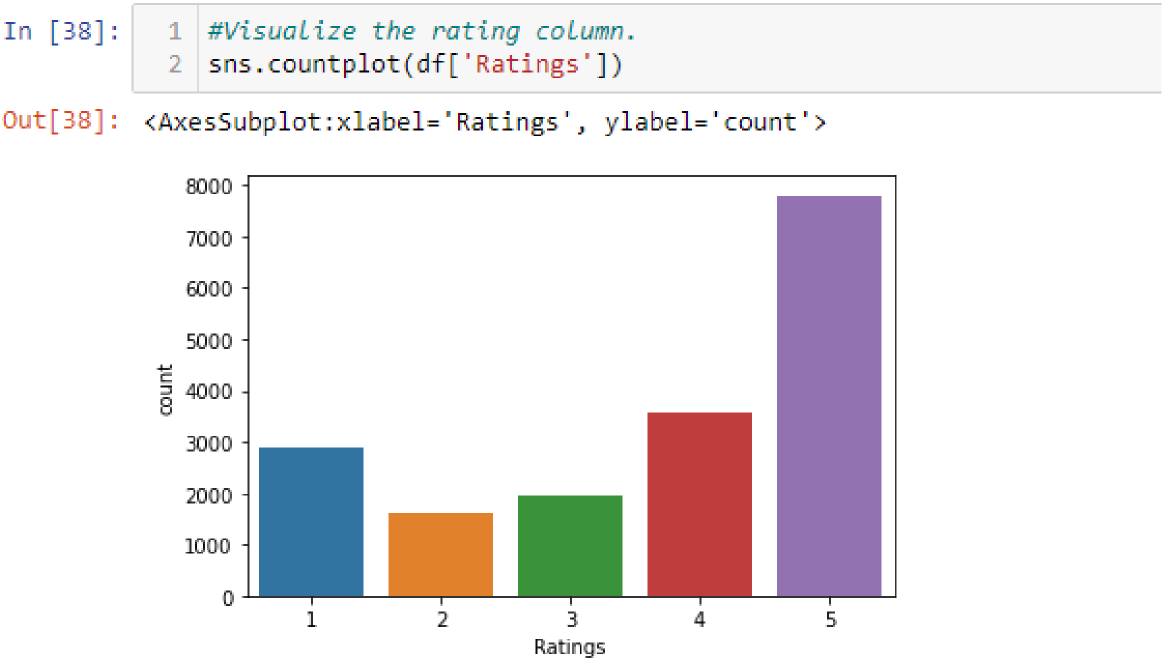
Data Analysis and Visualization

```
In [37]: #summary statistics.
         df.describe().style.background_gradient()
```

Out[37]:

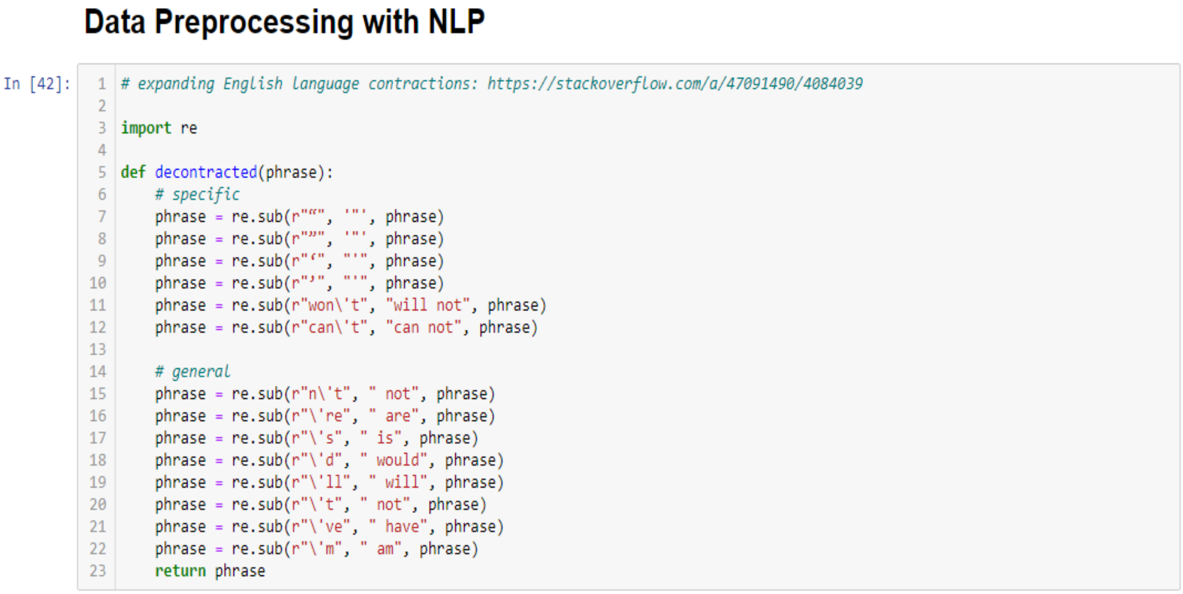
	Ratings
count	17839.000000
mean	3.660743
std	1.499987
min	1.000000
25%	2.000000
50%	4.000000
75%	5.000000
max	5.000000

- Display countplot of ratings column.



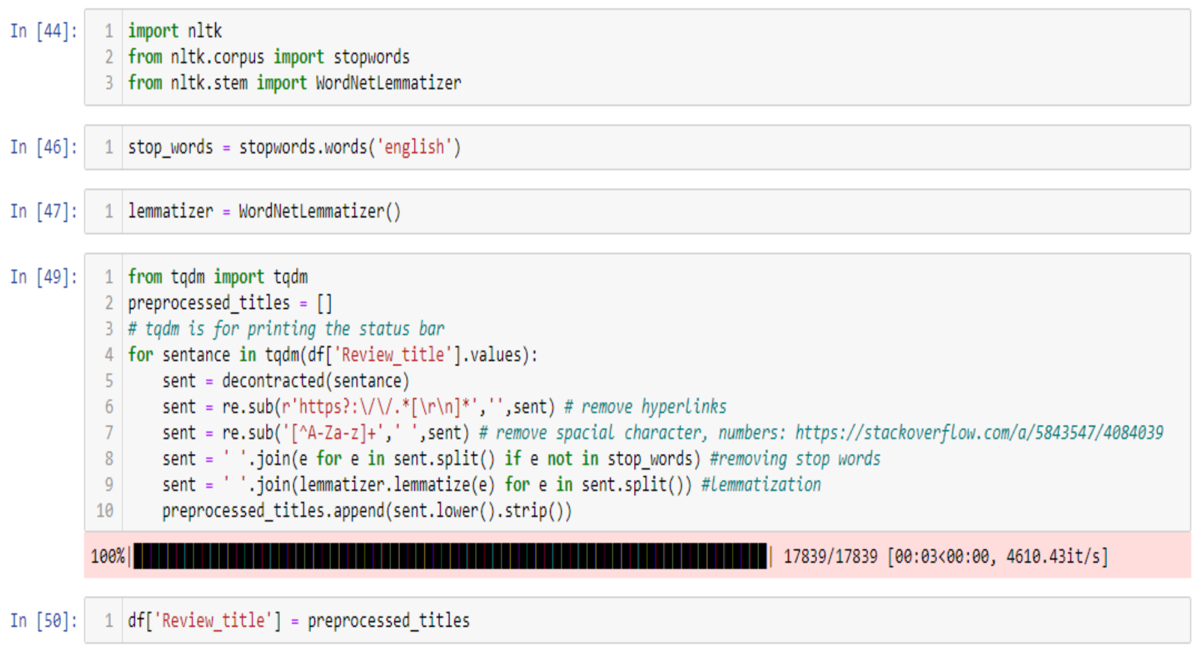
```
In [38]:  #Visualize the rating column.
          sns.countplot(df['Ratings'])

Out[38]: <AxesSubplot:xlabel='Ratings', ylabel='count'>
```

- Data Preprocessing with NLP.



Data Preprocessing with NLP

```
In [42]:  # expanding English language contractions: https://stackoverflow.com/a/47091490/4084039

          import re

          def decontracted(phrase):
              # specific
              phrase = re.sub(r"“", '"', phrase)
              phrase = re.sub(r"”", '"', phrase)
              phrase = re.sub(r"‘", "'", phrase)
              phrase = re.sub(r"’", "'", phrase)
              phrase = re.sub(r"won\'t", "will not", phrase)
              phrase = re.sub(r"can\'t", "can not", phrase)

              # general
              phrase = re.sub(r"n\'t", " not", phrase)
              phrase = re.sub(r"\'re", " are", phrase)
              phrase = re.sub(r"\'s", " is", phrase)
              phrase = re.sub(r"\'d", " would", phrase)
              phrase = re.sub(r"\'ll", " will", phrase)
              phrase = re.sub(r"\'t", " not", phrase)
              phrase = re.sub(r"\'ve", " have", phrase)
              phrase = re.sub(r"\'m", " am", phrase)
              return phrase
```



In [44]:
```
import nltk
from nltk.corpus import stopwords
from nltk.stem import WordNetLemmatizer
```

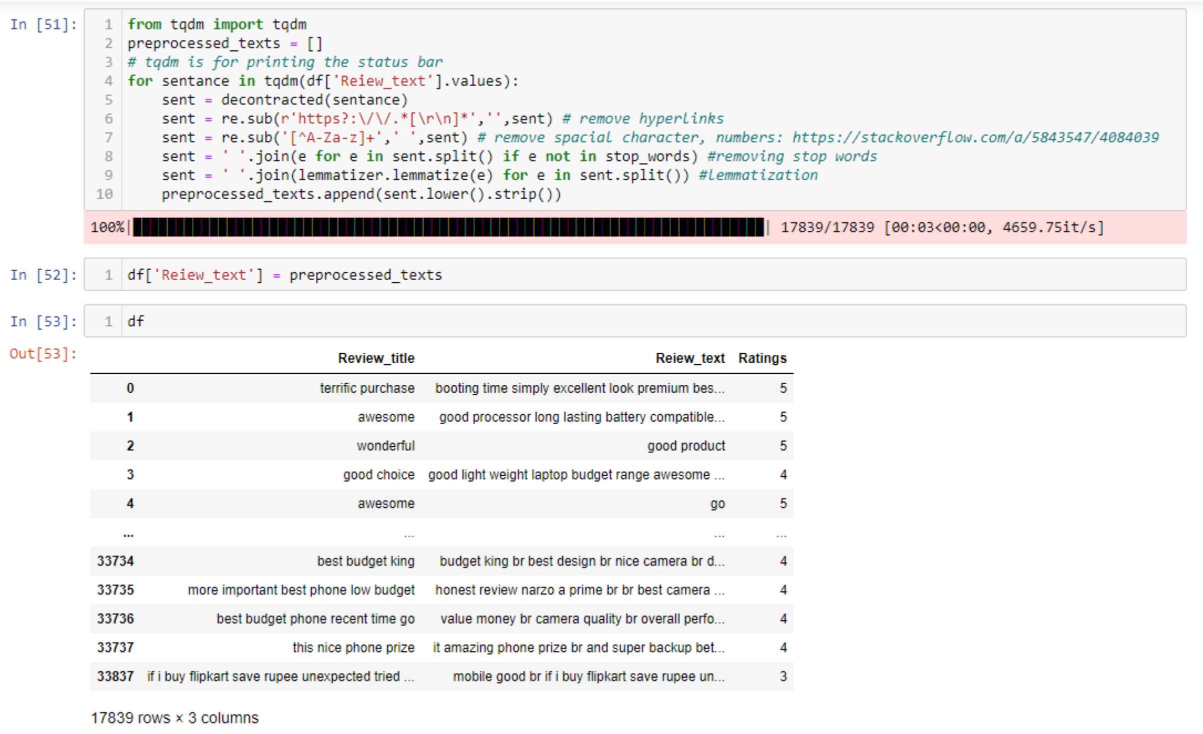
In [46]:
```
stop_words = stopwords.words('english')
```

In [47]:
```
lemmatizer = WordNetLemmatizer()
```

In [49]:
```
from tqdm import tqdm
preprocessed_titles = []
# tqdm is for printing the status bar
for sentance in tqdm(df['Review_title'].values):
    sent = decontracted(sentance)
    sent = re.sub(r'https?:\/\/.*[\r\n]*','',sent) # remove hyperlinks
    sent = re.sub('[^A-Za-z]+',' ',sent) # remove spacial character, numbers: https://stackoverflow.com/a/5843547/4084039
    sent = ' '.join(e for e in sent.split() if e not in stop_words) #removing stop words
    sent = ' '.join(lemmatizer.lemmatize(e) for e in sent.split()) #Lemmatization
    preprocessed_titles.append(sent.lower().strip())
```

```
100%|██████████████████████████████████████████████████| 17839/17839 [00:03<00:00, 4610.43it/s]
```

In [50]:
```
df['Review_title'] = preprocessed_titles
```



In [51]:
```
from tqdm import tqdm
preprocessed_texts = []
# tqdm is for printing the status bar
for sentance in tqdm(df['Reiew_text'].values):
    sent = decontracted(sentance)
    sent = re.sub(r'https?:\/\/.*[\r\n]*','',sent) # remove hyperlinks
    sent = re.sub('[^A-Za-z]+',' ',sent) # remove spacial character, numbers: https://stackoverflow.com/a/5843547/4084039
    sent = ' '.join(e for e in sent.split() if e not in stop_words) #removing stop words
    sent = ' '.join(lemmatizer.lemmatize(e) for e in sent.split()) #Lemmatization
    preprocessed_texts.append(sent.lower().strip())
```

```
100%|██████████████████████████████████████████████████| 17839/17839 [00:03<00:00, 4659.75it/s]
```

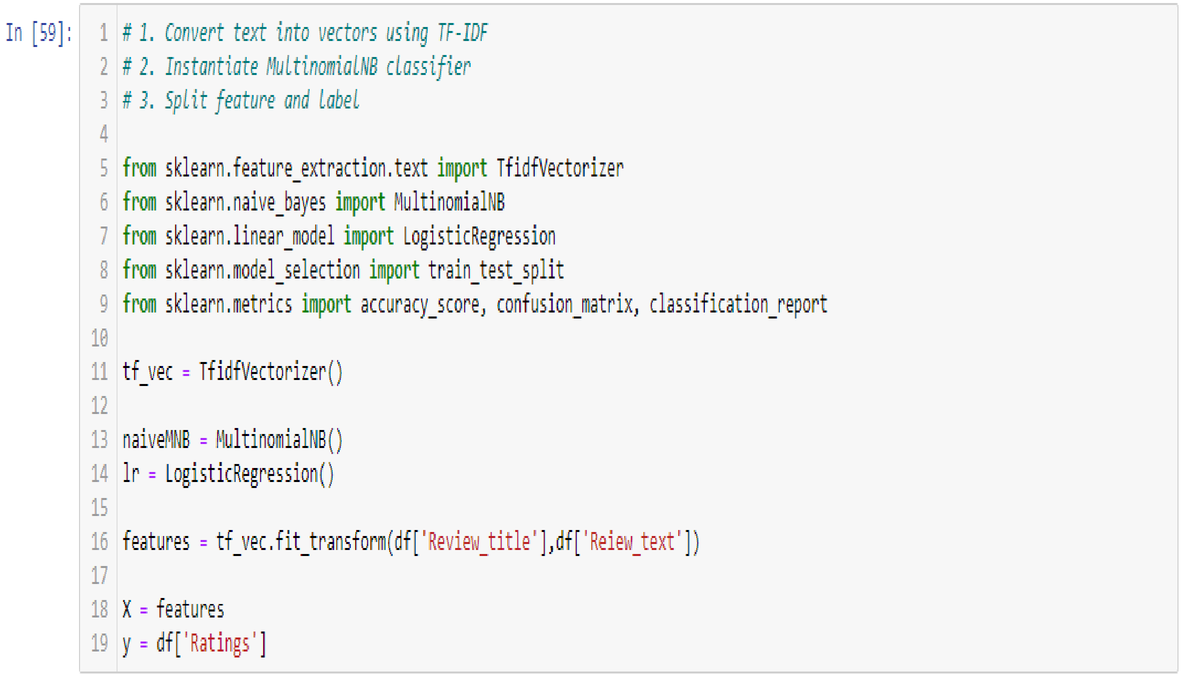
In [52]:
```
df['Reiew_text'] = preprocessed_texts
```

In [53]:
```
df
```

Out[53]:

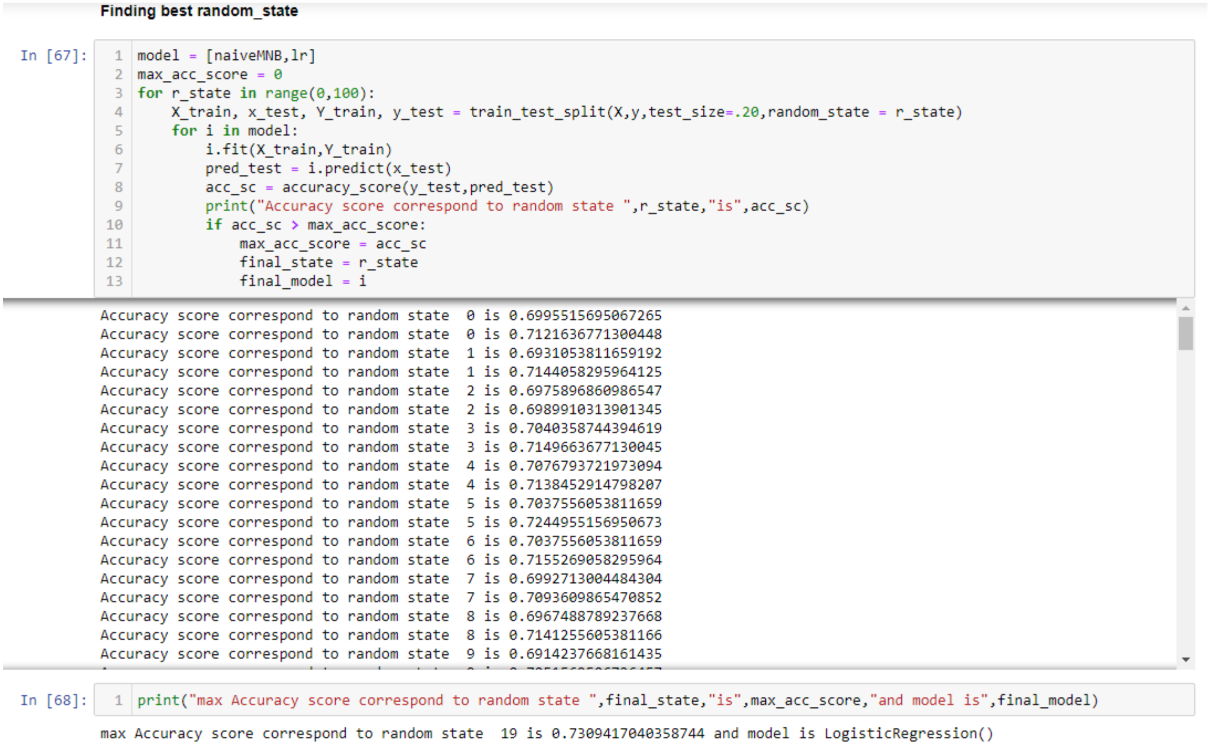
	Review_title	Reiew_text	Ratings
0	terrific purchase	booting time simply excellent look premium bes...	5
1	awesome	good processor long lasting battery compatible...	5
2	wonderful	good product	5
3	good choice	good light weight laptop budget range awesome ...	4
4	awesome	go	5
...	...	...	...
33734	best budget king	budget king br best design br nice camera br d...	4
33735	more important best phone low budget	honest review narzo a prime br br best camera ...	4
33736	best budget phone recent time go	value money br camera quality br overall perfo...	4
33737	this nice phone prize	it amazing phone prize br and super backup bet...	4
33837	if i buy flipkart save rupee unexpected tried ...	mobile good br if i buy flipkart save rupee un...	3

17839 rows × 3 columns



In [59]:
```python
# 1. Convert text into vectors using TF-IDF
# 2. Instantiate MultinomialNB classifier
# 3. Split feature and label

from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.naive_bayes import MultinomialNB
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score, confusion_matrix, classification_report

tf_vec = TfidfVectorizer()

naiveMNB = MultinomialNB()
lr = LogisticRegression()

features = tf_vec.fit_transform(df['Review_title'],df['Reiew_text'])

X = features
y = df['Ratings']
```

- Run and evaluate selected models.



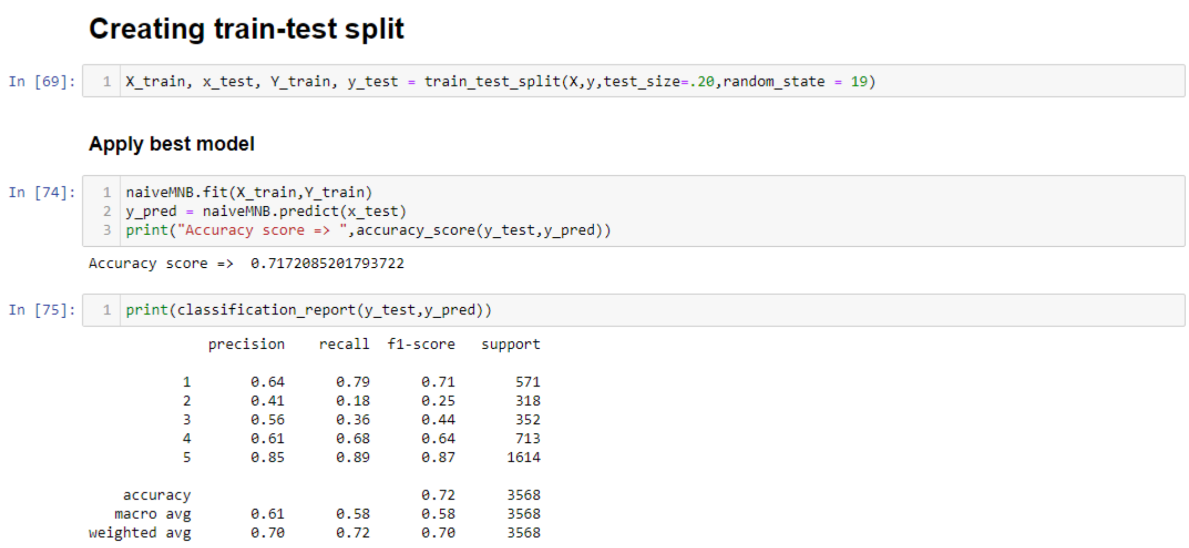
Finding best random_state

In [67]:
```python
model = [naiveMNB,lr]
max_acc_score = 0
for r_state in range(0,100):
    X_train, x_test, Y_train, y_test = train_test_split(X,y,test_size=.20,random_state = r_state)
    for i in model:
        i.fit(X_train,Y_train)
        pred_test = i.predict(x_test)
        acc_sc = accuracy_score(y_test,pred_test)
        print("Accuracy score correspond to random state ",r_state,"is",acc_sc)
        if acc_sc > max_acc_score:
            max_acc_score = acc_sc
            final_state = r_state
            final_model = i
```

```
Accuracy score correspond to random state  0 is 0.6995515695067265
Accuracy score correspond to random state  0 is 0.7121636771300448
Accuracy score correspond to random state  1 is 0.6931053811659192
Accuracy score correspond to random state  1 is 0.7144058295964125
Accuracy score correspond to random state  2 is 0.6975896860986547
Accuracy score correspond to random state  2 is 0.6989910313901345
Accuracy score correspond to random state  3 is 0.7040358744394619
Accuracy score correspond to random state  3 is 0.7149663677130045
Accuracy score correspond to random state  4 is 0.7076793721973094
Accuracy score correspond to random state  4 is 0.7138452914798207
Accuracy score correspond to random state  5 is 0.7037556053811659
Accuracy score correspond to random state  5 is 0.7244955156950673
Accuracy score correspond to random state  6 is 0.7037556053811659
Accuracy score correspond to random state  6 is 0.7155269058295964
Accuracy score correspond to random state  7 is 0.6992713004484304
Accuracy score correspond to random state  7 is 0.7093609865470852
Accuracy score correspond to random state  8 is 0.6967488789237668
Accuracy score correspond to random state  8 is 0.7141255605381166
Accuracy score correspond to random state  9 is 0.6914237668161435
```

In [68]:
```python
print("max Accuracy score correspond to random state ",final_state,"is",max_acc_score,"and model is",final_model)
```

```
max Accuracy score correspond to random state  19 is 0.7309417040358744 and model is LogisticRegression()
```



Creating train-test split

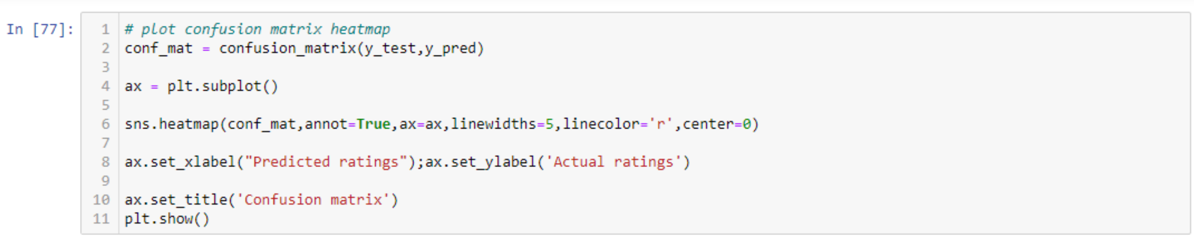
```
In [69]: X_train, x_test, Y_train, y_test = train_test_split(X,y,test_size=.20,random_state = 19)
```

Apply best model

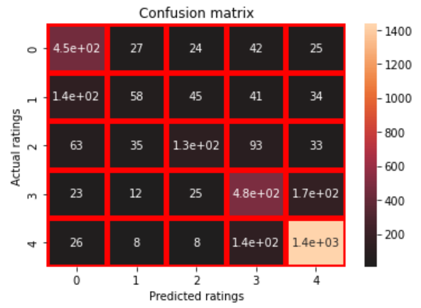
```
In [74]: naiveMNB.fit(X_train,Y_train)
         y_pred = naiveMNB.predict(x_test)
         print("Accuracy score => ",accuracy_score(y_test,y_pred))

Accuracy score =>  0.7172085201793722
```

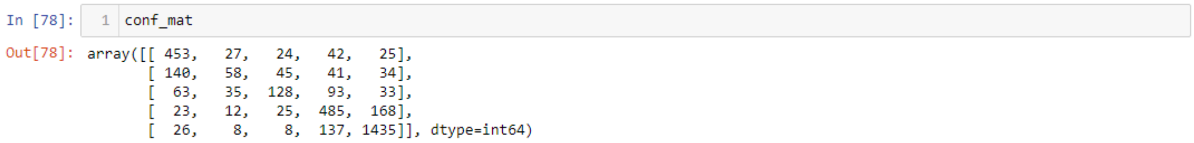
```
In [75]: print(classification_report(y_test,y_pred))

              precision    recall  f1-score   support

           1       0.64      0.79      0.71       571
           2       0.41      0.18      0.25       318
           3       0.56      0.36      0.44       352
           4       0.61      0.68      0.64       713
           5       0.85      0.89      0.87      1614

    accuracy                           0.72      3568
   macro avg       0.61      0.58      0.58      3568
weighted avg       0.70      0.72      0.70      3568
```

- Confusion matrix:



```
In [77]: # plot confusion matrix heatmap
         conf_mat = confusion_matrix(y_test,y_pred)

         ax = plt.subplot()

         sns.heatmap(conf_mat,annot=True,ax=ax,linewidths=5,linecolor='r',center=0)

         ax.set_xlabel("Predicted ratings");ax.set_ylabel('Actual ratings')

         ax.set_title('Confusion matrix')
         plt.show()
```





```
In [78]: conf_mat
Out[78]: array([[ 453,   27,   24,   42,   25],
                [ 140,   58,   45,   41,   34],
                [  63,   35,  128,   93,   33],
                [  23,   12,   25,  485,  168],
                [  26,    8,    8,  137, 1435]], dtype=int64)
```

Hardware and Software Requirements and Tools Used

Language :-	Python
Tool:-	Jupyter Notebook
OS:-	Windows 10
RAM:-	8gb

CONCLUSION:

- This Kernel investigates different models for car price prediction.
- Different types of Machine Learning methods including LogisticRegression, and MultinomialNB in machine learning are compared and analysed for optimal solutions.
- Even though all of those methods achieved desirable results, different models have their own pros and cons.
- The MultinomialNB is probably the best one and has been selected for this problem.
- Finally, the MultinomialNB is the best choice when parameterization is the top priority.