import librearies

In [1]:
```
import os
import numpy as np
import pandas as pd

import seaborn as sns
import matplotlib.pyplot as plt

import nltk
import re
nltk.download('stopwords')
stemmer = nltk.SnowballStemmer("english")
from nltk.corpus import stopwords
import string
stopword=set(stopwords.words('english'))

from nltk.sentiment.vader import SentimentIntensityAnalyzer
from wordcloud import WordCloud, STOPWORDS, ImageColorGenerator


from nltk.stem import PorterStemmer, WordNetLemmatizer, porter

nltk.download('punkt')
nltk.download('vader_lexicon')
```

```
[nltk_data] Downloading package stopwords to
[nltk_data]     C:\Users\Vikas\AppData\Roaming\nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
[nltk_data] Downloading package punkt to
[nltk_data]     C:\Users\Vikas\AppData\Roaming\nltk_data...
[nltk_data]   Package punkt is already up-to-date!
[nltk_data] Downloading package vader_lexicon to
[nltk_data]     C:\Users\Vikas\AppData\Roaming\nltk_data...
[nltk_data]   Package vader_lexicon is already up-to-date!
```

Out[1]:
```
True
```

In [2]:
```
df = pd.read_csv("Filpkart Product Reviews")
```

In [3]:
```
df.head()
```

Out[3]:

	product_name	product_price	Rate	Review	Summary	Sentiment
0	Candes 12 L Room/Personal Air Cooler??????(Whi...	3999	5	super!	great cooler excellent air flow and for this p...	positive
1	Candes 12 L Room/Personal Air Cooler??????(Whi...	3999	5	awesome	best budget 2 fit cooler nice cooling	positive
2	Candes 12 L Room/Personal Air Cooler??????(Whi...	3999	3	fair	the quality is good but the power of air is de...	positive
3	Candes 12 L Room/Personal Air Cooler??????(Whi...	3999	1	useless product	very bad product its a only a fan	negative
4	Candes 12 L Room/Personal Air Cooler??????(Whi...	3999	3	fair	ok ok product	neutral

Basic Information

In [4]:
```
df.shape
```

Out[4]:
```
(205052, 6)
```

In [5]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 205052 entries, 0 to 205051
Data columns (total 6 columns):
 #   Column         Non-Null Count   Dtype 
---  ------         --------------   ----- 
 0   product_name   205052 non-null  object
 1   product_price  205052 non-null  object
 2   Rate           205052 non-null  object
 3   Review         180388 non-null  object
 4   Summary        205041 non-null  object
 5   Sentiment      205052 non-null  object
dtypes: object(6)
memory usage: 9.4+ MB
```

In [6]:
```
df.isnull().sum()/len(df)*100
```

Out[6]:
```
product_name      0.000000
product_price     0.000000
Rate              0.000000
Review           12.028168
Summary           0.005364
Sentiment         0.000000
dtype: float64
```

In [7]:
```
df.describe()
```

Out[7]:

	product_name	product_price	Rate	Review	Summary	Sentiment
count	205052	205052	205052	180388	205041	205052
unique	958	525	8	1324	92923	3
top	cello Pack of 18 Opalware Cello Dazzle Lush Fi...	1299	5	wonderful	good	positive
freq	6005	9150	118765	9016	17430	166581

In [8]:
```
df.columns
```

Out[8]:
```
Index(['product_name', 'product_price', 'Rate', 'Review', 'Summary',
       'Sentiment'],
      dtype='object')
```

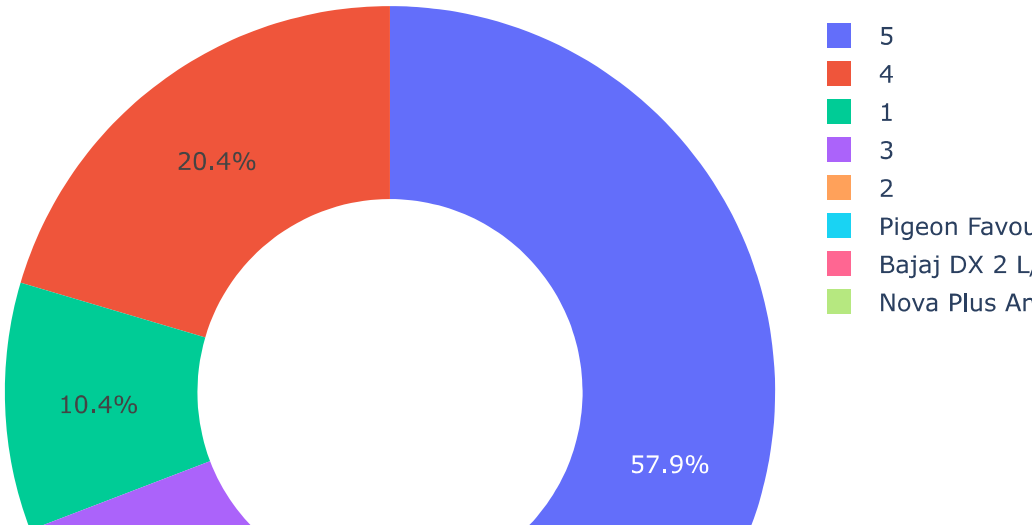
In [9]:
```
def clean(text):
    text = str(text).lower()
    text = re.sub('\[.*?\]', '', text)
    text = re.sub('https?://\S+|www\.\S+', '', text)
    text = re.sub('<.*?>+', '', text)
    text = re.sub('[%s]' % re.escape(string.punctuation), '', text)
    text = re.sub('\n', '', text)
    text = re.sub('\w*\d\w*', '', text)
    text = [word for word in text.split(' ') if word not in stopword]
    text=" ".join(text)
    text = [stemmer.stem(word) for word in text.split(' ')]
    text=" ".join(text)
    return text
df["Review"] = df["Review"].apply(clean)
```

In [10]:
```
df['Rate'].value_counts()
```

Out[10]:
```
5                                                                  118765
4                                                                   41894
1                                                                   21300
3                                                                   16599
2                                                                    6491
Pigeon Favourite Electric Kettle??????(1.5 L, Silver, Black)            1
Bajaj DX 2 L/W Dry Iron                                                 1
Nova Plus Amaze NI 10 1100 W Dry Iron?ÃÂ¿?ÃÂ¿(Grey & Turquoise)        1
Name: Rate, dtype: int64
```

In [11]:
```
rate = df['Rate'].value_counts()
numbers = rate.index
quantity = rate.values

import plotly.express as px
figure = px.pie(df,values = quantity,names = numbers,hole = 0.5)
figure.show()
```



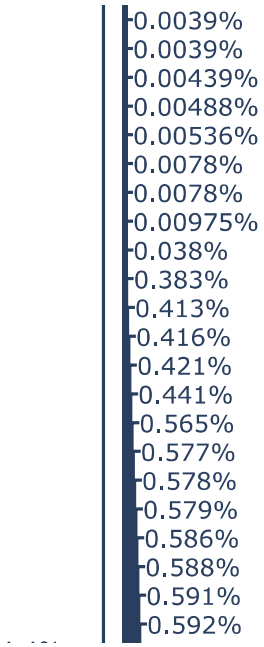
In [12]:
```
frequent_words = ' '.join([text for text in df['Rate']])
from wordcloud import WordCloud
wordcloud = WordCloud(width=800, height=500, random_state=21, max_font_size=11(
 background_color='black',colormap='viridis').generate(frequent_words)
plt.figure(figsize=(10, 7))
plt.imshow(wordcloud, interpolation="bilinear")
plt.axis('off')
plt.show()
```

In [13]:
```
df['Review'].value_counts()
```

Out[13]:
```
nan                                                 24664
wonder                                               9017
specifi                                              8351
good                                                 6476
brilliant                                            5643
                                                    ...
optim small famili                                      1
nice use product                                        1
good product  lowest price ever  flipkart               1
nice see product less rate                              1
product look nice  strong qualiti                       1
Name: Review, Length: 1184, dtype: int64
```

In [14]:
```
review = df['Review'].value_counts()
numbers = review.index
quantity = review.values

import plotly.express as px
figure = px.pie(df,values = quantity,names = numbers,hole = 0.5)
figure.show()
```



In [15]:
```
frequent_words = ' '.join([text for text in df['Review']])
from wordcloud import WordCloud
wordcloud = WordCloud(width=800, height=500, random_state=21, max_font_size=11(
 background_color='black',colormap='viridis').generate(frequent_words)
plt.figure(figsize=(10, 7))
plt.imshow(wordcloud, interpolation="bilinear")
plt.axis('off')
plt.show()
```

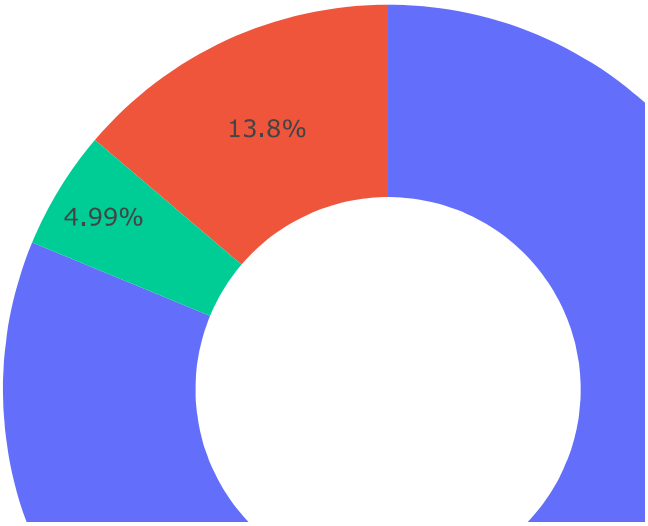
In [ ]:
```
```

In [16]:
```
df['Sentiment'].value_counts()/len(df)*100
```

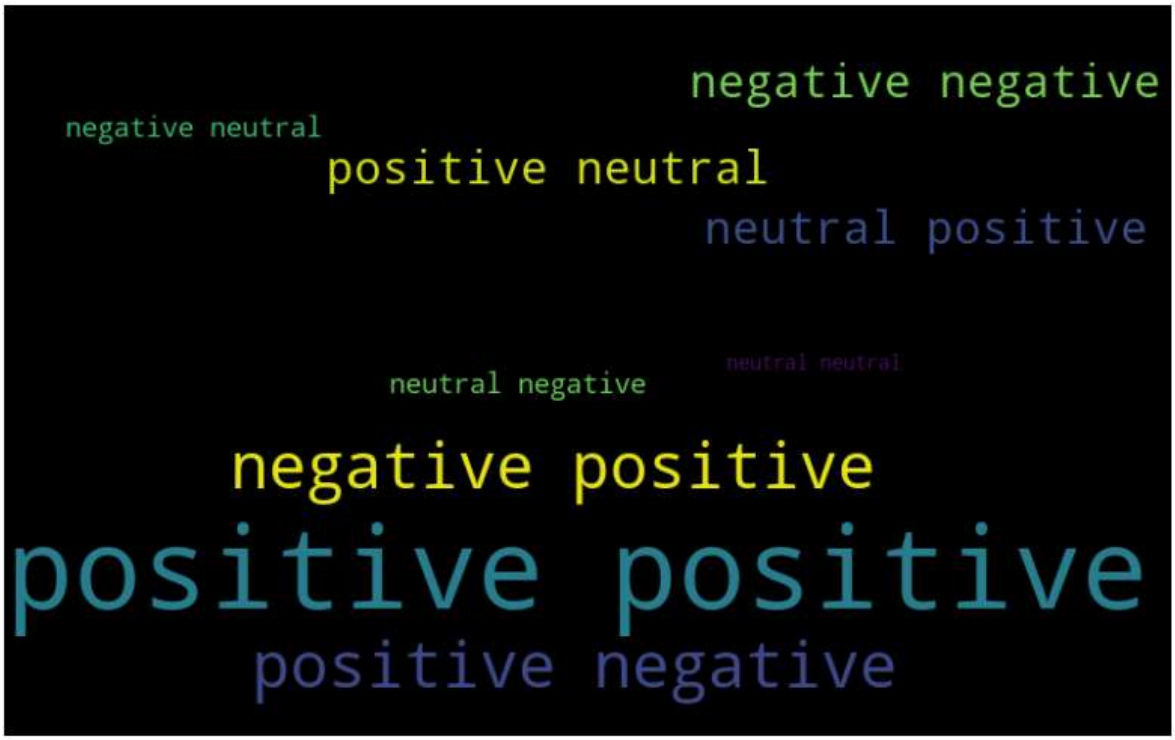
Out[16]:
```
positive    81.238418
negative    13.768215
neutral      4.993368
Name: Sentiment, dtype: float64
```

In [17]:
```
sentiment = df['Sentiment'].value_counts()
numbers = sentiment.index
quantity = sentiment.values

import plotly.express as px
figure = px.pie(df,values = quantity,names = numbers,hole = 0.5)
figure.show()
```



In [18]:
```
frequent_words = ' '.join([text for text in df['Sentiment']])
from wordcloud import WordCloud
wordcloud = WordCloud(width=800, height=500, random_state=21, max_font_size=11(
 background_color='black',colormap='viridis').generate(frequent_words)
plt.figure(figsize=(10, 7))
plt.imshow(wordcloud, interpolation="bilinear")
plt.axis('off')
plt.show()
```



In [19]:
```
df['Sentiment'].value_counts()/len(df)*100
```

Out[19]:
```
positive    81.238418
negative    13.768215
neutral      4.993368
Name: Sentiment, dtype: float64
```

In [20]:
```
positive_sentiments = ' '.join([text for text in df['Review'][df.Sentiment ==
positive_wordcloud = WordCloud(width=800, height=500, random_state=21, max_fon
 background_color='green',colormap='Set2').generate(positive_sentiments)
plt.figure(figsize=(10, 7))
plt.imshow(positive_wordcloud, interpolation="bilinear")
plt.axis('off')
plt.show()
```

In [21]:
```
positive_sentiments = ' '.join([text for text in df['Review'][df.Sentiment ==
positive_wordcloud = WordCloud(width=800, height=500, random_state=21, max_fon
 background_color='yellow',colormap='Set2').generate(positive_sentiments)
plt.figure(figsize=(10, 7))
plt.imshow(positive_wordcloud, interpolation="bilinear")
plt.axis('off')
plt.show()
```

In [22]:
```
positive_sentiments = ' '.join([text for text in df['Review'][df.Sentiment ==
positive_wordcloud = WordCloud(width=800, height=500, random_state=21, max_fon
 background_color='red',colormap='Set2').generate(positive_sentiments)
plt.figure(figsize=(10, 7))
plt.imshow(positive_wordcloud, interpolation="bilinear")
plt.axis('off')
plt.show()
```

In [23]:
```
df.head()
```

Out[23]:

	product_name	product_price	Rate	Review	Summary	Sentiment
0	Candes 12 L Room/Personal Air Cooler??????(Whi...	3999	5	super	great cooler excellent air flow and for this p...	positive
1	Candes 12 L Room/Personal Air Cooler??????(Whi...	3999	5	awesom	best budget 2 fit cooler nice cooling	positive
2	Candes 12 L Room/Personal Air Cooler??????(Whi...	3999	3	fair	the quality is good but the power of air is de...	positive
3	Candes 12 L Room/Personal Air Cooler??????(Whi...	3999	1	useless product	very bad product its a only a fan	negative
4	Candes 12 L Room/Personal Air Cooler??????(Whi...	3999	3	fair	ok ok product	neutral

In [24]:
```
from nltk.sentiment.vader import SentimentIntensityAnalyzer
sia = SentimentIntensityAnalyzer()
```

In [25]:
```
df['Sentiment'].value_counts()
```

Out[25]:
```
positive    166581
negative     28232
neutral      10239
Name: Sentiment, dtype: int64
```

In [26]:
```
sia.polarity_scores(df.loc[0]['Review'])
```

Out[26]:
```
{'neg': 0.0, 'neu': 0.0, 'pos': 1.0, 'compound': 0.5994}
```

In [27]:
```
df.loc[0]['Review']
```

Out[27]:
```
'super'
```

In [28]:
```
df.shape
```

Out[28]:
```
(205052, 6)
```

In [29]:
```
def clean(text):
    text = str(text).lower()
    text = re.sub('\[.*?\]', '', text)
    text = re.sub('https?://\S+|www\.\S+', '', text)
    text = re.sub('<.*?>+', '', text)
    text = re.sub('[%s]' % re.escape(string.punctuation), '', text)
    text = re.sub('\n', '', text)
    text = re.sub('\w*\d\w*', '', text)
    text = [word for word in text.split(' ') if word not in stopword]
    text=" ".join(text)
    text = [stemmer.stem(word) for word in text.split(' ')]
    text=" ".join(text)
    return text
df["Summary"] = df["Summary"].apply(clean)
```

In [30]:
```
df.isnull().sum()
```

Out[30]:
```
product_name     0
product_price    0
Rate             0
Review           0
Summary          0
Sentiment        0
dtype: int64
```

In [31]:
```
df['score']=df['Review'].apply(lambda review : sia.polarity_scores(review) )
df.head(10)
```

Out[31]:

	product_name	product_price	Rate	Review	Summary	Sentiment	score
0	Candes 12 L Room/Personal Air Cooler?????? (Whi...	3999	5	super	great cooler excel air flow price amaz unbelie...	positive	{'neg': 0.0, 'neu': 0.0, 'pos': 1.0, 'compound...
1	Candes 12 L Room/Personal Air Cooler?????? (Whi...	3999	5	awesom	best budget fit cooler nice cool	positive	{'neg': 0.0, 'neu': 1.0, 'pos': 0.0, 'compound...
2	Candes 12 L Room/Personal Air Cooler?????? (Whi...	3999	3	fair	qualiti good power air decent	positive	{'neg': 0.0, 'neu': 0.0, 'pos': 1.0, 'compound...
3	Candes 12 L Room/Personal Air Cooler?????? (Whi...	3999	1	useless product	bad product fan	negative	{'neg': 0.737, 'neu': 0.263, 'pos': 0.0, 'comp...
4	Candes 12 L Room/Personal Air Cooler?????? (Whi...	3999	3	fair	ok ok product	neutral	{'neg': 0.0, 'neu': 0.0, 'pos': 1.0, 'compound...
5	Candes 12 L Room/Personal Air Cooler?????? (Whi...	3999	5	awesom	cooler realli fantast provid good air flow hig...	positive	{'neg': 0.0, 'neu': 1.0, 'pos': 0.0, 'compound...
6	Candes 12 L Room/Personal Air Cooler?????? (Whi...	3999	5	high recommend	good product	positive	{'neg': 0.0, 'neu': 0.286, 'pos': 0.714, 'comp...
7	Candes 12 L Room/Personal Air Cooler?????? (Whi...	3999	3	nice	nice	positive	{'neg': 0.0, 'neu': 0.0, 'pos': 1.0, 'compound...
8	Candes 12 L Room/Personal Air Cooler?????? (Whi...	3999	1	unsatisfactori	bad cooler	negative	{'neg': 0.0, 'neu': 1.0, 'pos': 0.0, 'compound...
9	Candes 12 L Room/Personal Air Cooler?????? (Whi...	3999	4	worth money	good	positive	{'neg': 0.0, 'neu': 0.345, 'pos': 0.655, 'comp...

In [32]:
```
len(df['score'])
```

Out[32]: 205052

In [33]:
```
df['compound'] = df['score'].apply(lambda score_dict : score_dict['compound'])
df.head(10)
```

Out[33]:

	product_name	product_price	Rate	Review	Summary	Sentiment	score	compo
0	Candes 12 L Room/Personal Air Cooler?????? (Whi...	3999	5	super	great cooler excel air flow price amaz unbelie...	positive	{'neg': 0.0, 'neu': 0.0, 'pos': 1.0, 'compound...	0.
1	Candes 12 L Room/Personal Air Cooler?????? (Whi...	3999	5	awesom	best budget fit cooler nice cool	positive	{'neg': 0.0, 'neu': 1.0, 'pos': 0.0, 'compound...	0.
2	Candes 12 L Room/Personal Air Cooler?????? (Whi...	3999	3	fair	qualiti good power air decent	positive	{'neg': 0.0, 'neu': 0.0, 'pos': 1.0, 'compound...	0.
	Candes 12 L							

In [34]:
```
df['comp_score'] = df['compound'].apply(lambda c: 'pos' if c >=0 else 'neg')
df.head(10)
```

Out[34]:

	product_name	product_price	Rate	Review	Summary	Sentiment	score	compoun
0	Candes 12 L Room/Personal Air Cooler?????? (Whi...	3999	5	super	great cooler excel air flow price amaz unbelie...	positive	{'neg': 0.0, 'neu': 0.0, 'pos': 1.0, 'compound...	0.599
1	Candes 12 L Room/Personal Air Cooler?????? (Whi...	3999	5	awesom	best budget fit cooler nice cool	positive	{'neg': 0.0, 'neu': 1.0, 'pos': 0.0, 'compound...	0.000
2	Candes 12 L Room/Personal Air Cooler?????? (Whi...	3999	3	fair	qualiti good power air decent	positive	{'neg': 0.0, 'neu': 0.0, 'pos': 1.0, 'compound...	0.318
3	Candes 12 L Room/Personal Air Cooler?????? (Whi...	3999	1	useless product	bad product fan	negative	{'neg': 0.737, 'neu': 0.263, 'pos': 0.0, 'comp...	-0.421
4	Candes 12 L Room/Personal Air Cooler?????? (Whi...	3999	3	fair	ok ok product	neutral	{'neg': 0.0, 'neu': 0.0, 'pos': 1.0, 'compound...	0.318
5	Candes 12 L Room/Personal Air Cooler?????? (Whi...	3999	5	awesom	cooler realli fantast provid good air flow hig...	positive	{'neg': 0.0, 'neu': 1.0, 'pos': 0.0, 'compound...	0.000
6	Candes 12 L Room/Personal Air Cooler?????? (Whi...	3999	5	high recommend	good product	positive	{'neg': 0.0, 'neu': 0.286, 'pos': 0.714, 'comp...	0.361
7	Candes 12 L Room/Personal Air Cooler?????? (Whi...	3999	3	nice	nice	positive	{'neg': 0.0, 'neu': 0.0, 'pos': 1.0, 'compound...	0.421
8	Candes 12 L Room/Personal Air Cooler?????? (Whi...	3999	1	unsatisfactori	bad cooler	negative	{'neg': 0.0, 'neu': 1.0, 'pos': 0.0, 'compound...	0.000
9	Candes 12 L Room/Personal Air Cooler?????? (Whi...	3999	4	worth money	good	positive	{'neg': 0.0, 'neu': 0.345, 'pos': 0.655, 'comp...	0.226

Preprocessed Method

In [35]:
```
from tqdm import tqdm

preprocessed_reviews = []

for sentence in tqdm(df['Review'].values):
    sentence = re.sub('[^a-zA-Z]',' ',sentence)
    sentence = ' '.join(low.lower() for low in sentence.split() if low.lower()
    preprocessed_reviews.append(sentence.strip())
```

```
100%|████████████████████████████████████████████████████████████████| 205052/205052 [02:35<00:00,
1321.88it/s]
```

In [36]:
```
df['Review'].values
```

Out[36]:
```
array(['super', 'awesom', 'fair', ..., 'nice', 'wow', 'valueformoney'],
      dtype=object)
```

In [37]:
```
preprocessed_reviews
```

Out[37]:
```
['super',
 'awesom',
 'fair',
 'useless product',
 'fair',
 'awesom',
 'high recommend',
 'nice',
 'unsatisfactori',
 'worth money',
 'great product',
 'mindblow purchas',
 'high recommend',
 'brilliant',
 'classi product',
 'must buy',
 'fabul',
 'worth everi penni',
 'super',
 'great product',
 'awesom',
 'worth everi penni',
 'wow',
 'awesom',
 'terrif purchas',
 'excel',
 'worth everi penni',
 'terrif',
 'awesom',
 'simpli awesom',
 'wonder',
 'terrif',
 'expect better product',
 'excel',
 'terrif purchas',
 'wonder',
 'wonder',
 'good qualiti product',
 'awesom',
 'excel',
 'great product',
 'high recommend',
 'awesom',
 'decent product',
 'fabul',
 'satisfactori',
 'good qualiti product',
 'mindblow purchas',
 'wast money',
 'must buy',
 'good best',
 'must buy',
 'job',
 'nice',
 'worth money',
 'pretti good',
 'good cooler',
```

```
 'wonder',
 'wonder',
 'worthless',
 'great product',
 'great product',
 'must buy',
 'good qualiti product',
 'decent product',
 'brilliant',
 'super',
 'worst experi ever',
 'perfect product',
 'wonder',
 'simpli awesom',
 'nice product',
 'worth money',
 'useless product',
 'excel',
 'like assembl one get cheap qualiti',
 'simpli awesom',
 'great product',
 'wonder',
 'wonder',
 'good',
 'good',
 'super',
 'terrif purchas',
 'slight disappoint',
 'good qualiti product',
 'expect better product',
 'horribl',
 'nice',
 'good qualiti product',
 'simpli awesom',
 'mindblow purchas',
 'high recommend',
 'fabul',
 'brilliant',
 'great product',
 'high recommend',
 'must buy',
 'terrif purchas',
 'poor',
 'mindblow purchas',
 'terrif purchas',
 'good qualiti product',
 'wonder',
 'useless product',
 'terrif purchas',
 'simpli awesom',
 'unsatisfactori',
 'realli nice',
 'unsatisfactori',
 'best market',
 'high recommend',
 'classi product',
 'good qualiti product',
```

```
'best market',
'good qualiti product',
'brilliant',
'absolut rubbish',
'terrif',
'terrif purchas',
'okay',
'nice product',
'good qualiti product',
'perfect product',
'must buy',
'perfect product',
'bad qualiti',
'mindblow purchas',
'worth money',
'worst experi ever',
'realli nice',
'fabul',
'high recommend',
'fabul',
'good qualiti product',
'great product',
'realli nice',
'wonder',
'wow',
'good qualiti product',
'okay',
'super',
'nice',
'fabul',
'worthless',
'worth everi penni',
'worth money',
'absolut rubbish',
'super',
'good',
'realli nice',
'okay',
'terrif purchas',
'mindblow purchas',
'high recommend',
'poor',
'mindblow purchas',
'terrif purchas',
'best market',
'excel',
'delight',
'wonder',
'great product',
'worst experi ever',
'terrif',
'valueformoney',
'fabul',
'good choic',
'must buy',
'good qualiti product',
'wonder',
```

```
'good',
'excel',
'poor',
'mindblow purchas',
'classi product',
'wonder',
'worst experi ever',
'wow',
'nice',
'absolut rubbish',
'decent product',
'mindblow purchas',
'hate',
'useless product',
'wonder',
'awesom',
'super',
'good qualiti product',
'fair',
'worth money',
'good choic',
'good',
'simpli awesom',
'valueformoney',
'absolut rubbish',
'unsatisfactori',
'good qualiti product',
'mindblow purchas',
'awesom',
'terrif',
'nice',
'wonder',
'terribl product',
'super',
'best market',
'good choic',
'terrif purchas',
'super',
'good choic',
'best market',
'recommend',
'awesom',
'terrif',
'delight',
'good',
'perfect product',
'perfect product',
'super',
'hate',
'decent product',
'terrif',
'excel',
'terrif',
'simpli awesom',
'brilliant',
'classi product',
'wonder',
```

```
'best market',
'best market',
'best market',
'meet expect',
'wonder',
'classi product',
'wow',
'terrif',
'good',
'fabul',
'simpli awesom',
'delight',
'valueformoney',
'poor',
'nice product',
'terrif',
'valueformoney',
'wonder',
'decent product',
'must buy',
'mindblow purchas',
'super',
'high recommend',
'super',
'excel',
'good',
'horribl',
'fabul',
'perfect product',
'decent product',
'bad disappoint',
'fabul',
'good',
'perfect product',
'terribl product',
'unsatisfactori',
'fabul',
'wow',
'high recommend',
'good qualiti product',
'okay',
'excel',
'mindblow purchas',
'valueformoney',
'worth everi penni',
'brilliant',
'wonder',
'terrif purchas',
'unsatisfactori',
'valueformoney',
'good',
'excel',
'nice',
'wow',
'wonder',
'brilliant',
'best market',
```

```
 'good',
 'worth money',
 'must buy',
 'wonder',
 'utter disappoint',
 'realli nice',
 'mindblow purchas',
 'pretti good',
 'awesom',
 'terribl product',
 'job',
 'excel',
 'decent product',
 'meet expect',
 'absolut rubbish',
 'good',
 'good qualiti product',
 'job',
 'nice',
 'worth money',
 'worth money',
 'perfect product',
 'valueformoney',
 'pretti good',
 'excel',
 'best market',
 'good',
 'excel',
 'super',
 'perfect product',
 'delight',
 'moder',
 'poor',
 'horribl',
 'perfect product',
 'super',
 'worth everi penni',
 'fabul',
 'wonder',
 'best market',
 'wonder',
 'absolut rubbish',
 'mindblow purchas',
 'nice product',
 'simpli awesom',
 'realli nice',
 'delight',
 'super',
 'realli nice',
 'wonder',
 'best market',
 'terrif purchas',
 'fabul',
 'fabul',
 'delight',
 'excel',
 'worth everi penni',
```

```
'brilliant',
'terrif purchas',
'okay',
'excel',
'horribl',
'bad qualiti',
'excel',
'valueformoney',
'good',
'must buy',
'realli nice',
'classi product',
'perfect product',
'perfect product',
'worth everi penni',
'worth everi penni',
'worth money',
'valueformoney',
'wow',
'pretti good',
'fabul',
'good qualiti product',
'worth everi penni',
'nice',
'delight',
'mindblow purchas',
'good',
'best market',
'terribl product',
'must buy',
'fabul',
'classi product',
'terrif purchas',
'good qualiti product',
'best product best buy',
'utter disappoint',
'classi product',
'mindblow purchas',
'pretti good',
'simpli awesom',
'high recommend',
'classi product',
'wow',
'must buy',
'excel',
'worth everi penni',
'super',
'awesom',
'must buy',
'good qualiti product',
'classi product',
'wonder',
'wow',
'classi product',
'good choic',
'fair',
'wonder',
```

```
'fabul',
'good',
'pretti good',
'perfect product',
'nice product',
'good',
'worth everi penni',
'recommend',
'pretti good',
'okay',
'terrif purchas',
'good choic',
'brilliant',
'hate',
'classi product',
'worth everi penni',
'worth money',
'valueformoney',
'high recommend',
'good',
'poor',
'recommend',
'simpli awesom',
'wow',
'good qualiti product',
'worth everi penni',
'wow',
'good',
'worth money',
'terrif',
'must buy',
'terrif purchas',
'super',
'excel',
'realli nice',
'mindblow purchas',
'simpli awesom',
'wonder',
'wow',
'dont wast money',
'wonder',
'great product',
'brilliant',
'excel',
'terrif purchas',
'worth money',
'okay',
'bad qualiti',
'brilliant',
'good',
'worth money',
'valueformoney',
'perfect product',
'perfect product',
'valueformoney',
'best market',
'simpli awesom',
```

```
'wow',
'good',
'worth everi penni',
'great product',
'high recommend',
'nice product',
'wow',
'mindblow purchas',
'excel',
'high recommend',
'brilliant',
'worth everi penni',
'utter disappoint',
'brilliant',
'classi product',
'good',
'best market',
'delight',
'must buy',
'must buy',
'expect better product',
'excel',
'wow',
'good qualiti product',
'must buy',
'classi product',
'excel',
'worth money',
'nice product',
'awesom',
'terrif purchas',
'wonder',
'awesom',
'nice product',
'must buy',
'worth everi penni',
'good qualiti product',
'wonder',
'okay',
'super',
'good choic',
'fabul',
'moder',
'perfect product',
'perfect product',
'good qualiti product',
'valueformoney',
'worth money',
'perfect product',
'bad qualiti',
'good',
'terrif',
'worth everi penni',
'simpli awesom',
'simpli awesom',
'awesom',
'brilliant',
```

```
'product good huge size',
'high recommend',
'horribl',
'great product',
'mindblow purchas',
'fabul',
'simpli awesom',
'high recommend',
'recommend',
'recommend',
'perfect product',
'wonder',
'fair',
'fair',
'excel',
'good choic',
'high recommend',
'super',
'valueformoney',
'must buy',
'simpli awesom',
'great product',
'poor',
'good',
'nice',
'classi product',
'must buy',
'brilliant',
'perfect product',
'utter disappoint',
'terrif purchas',
'terrif',
'great product',
'delight',
'bad qualiti',
'moder',
'realli nice',
'wonder',
'delight',
'awesom',
'dont wast money',
'simpli awesom',
'good',
'utter disappoint',
'awesom',
'pretti good',
'must buy',
'nice',
'pretti good',
'classi product',
'great product',
'worth money',
'fair',
'okay',
'terribl product',
'good',
'terrif purchas',
```

```
 'terrif',
 'awesom',
 'terrif',
 'nice',
 'good choic',
 'best market',
 'super',
 'best market',
 'simpli awesom',
 'terrif purchas',
 'realli nice',
 'good qualiti product',
 'delight',
 'dont wast money',
 'good choic',
 'awesom',
 'valueformoney',
 'good choic',
 'wow',
 'decent product',
 'simpli awesom',
 'good',
 'must buy',
 'perfect product',
 'terrif',
 'great product',
 'nice product',
 'brilliant',
 'valueformoney',
 'best market',
 'nice product',
 'job',
 'wonder',
 'good',
 'great product',
 'terrif purchas',
 'pretti good',
 'worth money',
 'must buy',
 'mindblow purchas',
 'good',
 'best market',
 'simpli awesom',
 'delight',
 'good',
 'awesom',
 'best market',
 'super',
 'valueformoney',
 'realli nice',
 'nice product',
 'fabul',
 'realli nice',
 'classi product',
 'could way better',
 'bad qualiti',
 'wonder',
```

```
'awesom',
'great product',
'hate',
'wonder',
'wonder',
'excel',
'must buy',
'wast money',
'worst experi ever',
'valueformoney',
'classi product',
'wow',
'delight',
'best market',
'must buy',
'wow',
'excel',
'wonder',
'terrif',
'good',
'terrif',
'terrif purchas',
'brilliant',
'good qualiti product',
'decent product',
'good',
'wow',
'wow',
'great product',
'bad qualiti',
'mindblow purchas',
'wonder',
'realli nice',
'classi product',
'good qualiti product',
'mindblow purchas',
'fabul',
'fabul',
'mindblow purchas',
'terrif',
'super',
'excel',
'realli nice',
'terrif purchas',
'simpli awesom',
'good qualiti product',
'fair',
'fair',
'terribl product',
'nice product',
'worthless',
'good choic',
'awesom',
'high recommend',
'wonder',
'wonder',
'valueformoney',
```

```
 'wow',
 'perfect product',
 'high recommend',
 'simpli awesom',
 'simpli awesom',
 'pretti good',
 'pretti good',
 'classi product',
 'fabul',
 'terrif',
 'perfect product',
 'nice',
 'excel',
 'delight',
 'wonder',
 'best market',
 'horribl',
 'super',
 'fabul',
 'worthless',
 'terrif',
 'wonder',
 'worth money',
 'realli nice',
 'nice',
 'good qualiti product',
 'brilliant',
 'nice product',
 'wonder',
 'worth money',
 'job',
 'wow',
 'terribl product',
 'must buy cool product',
 'wonder',
 'great product',
 'must buy',
 'simpli awesom',
 'valueformoney',
 'brilliant',
 'excel',
 'super',
 'okay',
 'hate',
 'excel',
 'valueformoney',
 'worth everi penni',
 'delight',
 'utter disappoint',
 'must buy',
 'great product',
 'wow',
 'mindblow purchas',
 'recommend',
 'worth money',
 'terrif',
 'decent product',
```

```
'good',
'excel',
'mindblow purchas',
'worth money',
'high recommend',
'brilliant',
'best market',
'nice product',
'terrif purchas',
'classi product',
'wonder',
'simpli awesom',
'moder',
'bad qualiti',
'worth money',
'wonder',
'good choic',
'pretti good',
'okay',
'mindblow purchas',
'good qualiti product',
'terrif purchas',
'great product',
'moder',
'nice',
'great product',
'high recommend',
'realli nice',
'good qualiti product',
'worth money',
'delight',
'nice product',
'high recommend',
'high recommend',
'pretti good',
'good qualiti product',
'simpli awesom',
'nice',
'wonder',
'classi product',
'good',
'good qualiti product',
'absolut rubbish',
'wow',
'terrif',
'must buy',
'good',
'must buy',
'wonder',
'hate',
'good qualiti product',
'wast money',
'worth everi penni',
'brilliant',
'terrif purchas',
'good',
'decent product',
```

```
'job',
'mindblow purchas',
'delight',
'wow',
'good qualiti product',
'good qualiti product',
'worth everi penni',
'high recommend',
'terrif',
'super',
'wow',
'terrif',
'super',
'okay',
'best market',
'valueformoney',
'wonder',
'hate',
'simpli awesom',
'worth money desert cooler live name',
'worth everi penni',
'fabul',
'super',
'terrif purchas',
'classi product',
'perfect product',
'cooler ac cooler',
'wonder',
'ok kind coolernot effect hot condit',
'dont wast money',
'worth everi penni',
'terrif purchas',
'nice product',
'nice',
'mindblow purchas',
'wonder',
'valueformoney',
'unsatisfactori',
'utter disappoint',
'brilliant',
'best market',
'mindblow purchas',
'good qualiti product',
'simpli awesom',
'classi product',
'wow',
'awesom',
'wonder',
'brilliant',
'great product',
'wonder',
'awesom cooler',
'cool cooler extrem summer heat',
'worth everi penni',
'terrif purchas',
'wow',
'high recommend',
```

```
 'excel',
 'perfect product',
 'wow',
 'decent product',
 'perfect product',
 'good',
 'simpli awesom',
 'terrif',
 'great product',
 'perfect product',
 'delight',
 'wow',
 'worth everi penni',
 'terrif',
 'okay',
 'nice product',
 'cool well user manual improv',
 'worth everi penni',
 'worth everi penni',
 'simpli awesom',
 'terrif',
 'unsatisfactori',
 'worth money',
 'worth everi penni',
 'fabul',
 'excel',
 'meet expect',
 'best market',
 'decent product',
 'dont wast money',
 'terrif',
 'wow',
 'okay',
 'good',
 'fair',
 'awesom',
 'great product',
 'awesom',
 'terrif purchas',
 'brilliant',
 'worth money',
 'high recommend',
 'worth money',
 'meet expect',
 'pretti good',
 'high recommend',
 'brilliant',
 'high recommend',
 'excel',
 'fair',
 'perfect product',
 'best cooler',
 'great product',
 'wonder',
 'perfect product',
 'valueformoney',
 'brilliant',
```

```
'superexcell product',
'fabul',
'brilliant',
'high recommend',
'terrif',
'awesom',
'wonder',
'nice',
'good',
'worth money',
'mindblow purchas',
'mindblow purchas',
'fantast air cooler afford price',
'delight',
'perfect product',
'simpli awesom',
'terrif purchas',
'awesom',
'ultim cooler price rang',
'great product',
'terrif',
'wow',
'worth everi penni',
'wonder',
'excel',
'brilliant',
'great product',
'decent product',
'wow',
'super',
'perfect product',
'perfect product',
'wow',
'mindblow purchas',
'great product',
'mindblow purchas',
'best market',
'pretti good',
'fair',
'high recommend',
'worth everi penni',
'must buy',
'delight',
'terrif purchas',
'awesom',
'wow',
'fab cooler',
'crompton ozon desert air cooler',
'worth everi penni',
'terrif',
'excel',
'decent product',
'excel',
'worth everi penni',
'brilliant',
'valueformoney',
'best market',
```

```
 'mindblow purchas',
 'high recommend',
 'recommend',
 'product chang',
 'item good delhiveri courier servic worst',
 'terrif',
 'great product',
 'best market',
 'worthless',
 'worth everi penni',
 'pretti good',
 'perfect product',
 'good',
 'mindblow purchas',
 'best market',
 'wonder',
 'delight',
 'must buy',
 'brilliant',
 'horribl',
 'mindblow purchas',
 'absolut rubbish',
 'meet expect',
 'classi product',
 'super',
 'useless product',
 'good qualiti product',
 'great product',
 'terribl product',
 'best market',
 'terrif purchas',
 ...]
```

Feature Extraction

TF-IDF : Term Frequency - Inverse Documents Frequency

In [38]:
```
from sklearn.feature_extraction.text import TfidfVectorizer
tfidf = TfidfVectorizer()
x = tfidf.fit_transform(preprocessed_reviews).toarray()
pd.DataFrame(x).shape
```

Out[38]:
```
(205052, 1041)
```

In [39]:
```
pd.DataFrame(x).head()
```

Out[39]:

	0	1	2	3	4	5	6	7	8	9	...	1031	1032	1033	1034	1035	1036	1037
0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0
1	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0
2	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0
3	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0
4	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0

5 rows × 1041 columns

In [40]:
```
df['Sentiment'] = df['Sentiment'].astype('category')
df['Sentiment'] = df['Sentiment'].cat.codes
```

In [41]:
```
df['Sentiment'].value_counts()
```

Out[41]:
```
2    166581
0     28232
1     10239
Name: Sentiment, dtype: int64
```

In [42]:
```
# Split the data into training and test
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(x, df['Sentiment'], test_s
```

naive_bayes Therom

In [43]:
```
from sklearn.naive_bayes import MultinomialNB
nbmodel = MultinomialNB().fit(x_train, y_train)
```

In [44]:
```
from sklearn.naive_bayes import MultinomialNB
```

In [45]:
```
nbmodel = MultinomialNB().fit(x_train, y_train)
```

In [46]:
```
# fitting the model
nbmodel.fit(x_train, y_train)
```

Out[46]:
```
MultinomialNB
MultinomialNB()
```

In [47]:
```
# predicting the test set result
nbmodel_pred = nbmodel.predict(x_test)
```

In [48]:
```
# Training accuracy
print('Training accuracy  ',nbmodel.score(x_train,y_train))
```

```
Training accuracy   0.8900636586491881
```

Logistic Regression

In [49]:
```
from sklearn.linear_model import LogisticRegression
```

In [50]:
```
lr = LogisticRegression()
```

In [51]:
```
# fitting the model
lr.fit(x_train,y_train)
```

```
C:\ProgramData\anaconda3\lib\site-packages\sklearn\linear_model\_logistic.py:
458: ConvergenceWarning:

lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html (https://sciki
t-learn.org/stable/modules/preprocessing.html)
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regres
sion (https://scikit-learn.org/stable/modules/linear_model.html#logistic-regr
ession)
```

Out[51]:
```
▾ LogisticRegression

LogisticRegression()
```

In [52]:
```
# predicting the test set result
lr_pred = lr.predict(x_test)
```

In [53]:
```
# Training accuracy
print('Training accuracy  ',lr.score(x_train,y_train))
```

```
Training accuracy   0.8954021418957142
```

In [54]:
```
nltk.download('vader_lexicon')
sentiments = SentimentIntensityAnalyzer()
df["Positive"] = [sentiments.polarity_scores(i)["pos"] for i in df["Review"]]
df["Negative"] = [sentiments.polarity_scores(i)["neg"] for i in df["Review"]]
df["Neutral"] = [sentiments.polarity_scores(i)["neu"] for i in df["Review"]]
df = df[["Review", "Positive", "Negative", "Neutral"]]
print(df.head())
```

```
[nltk_data] Downloading package vader_lexicon to
[nltk_data]     C:\Users\Vikas\AppData\Roaming\nltk_data...
[nltk_data]   Package vader_lexicon is already up-to-date!

            Review  Positive  Negative  Neutral
0            super       1.0     0.000    0.000
1           awesom       0.0     0.000    1.000
2             fair       1.0     0.000    0.000
3  useless product       0.0     0.737    0.263
4             fair       1.0     0.000    0.000
```

In [60]:
```
A = sum(df["Positive"])
B = sum(df["Negative"])
C = sum(df["Neutral"])

def sentiment_score(a, b, c):
    if (a>b):
        print("Positive 😊 ")
    elif (b>c):
        print("Negative 😠 ")
    else:
        print("Neutral 🙂 ")
sentiment_score(A,B,C)
```

```
Positive 😊
```

Conclusion

In [ ]:
```
There are positive review 166581,negative review 28232,neutral review 10239

1.most of the review are in favor of positive review.
2.The negative review are midway and neutral review less than negative review.
3.So with the help of the above sentiment analysis implementation, we can conc
4.Sentiment analysis is important because, based on bad reviews, the e-commerc
```