Import libary

In [1]:
```
import pandas as pd
```

In [2]:
```
import numpy as np
```

In [3]:
```
import matplotlib.pyplot as plt
```

In [4]:
```
import seaborn as sns
```

Import dataset

In [5]:
```
df = pd.read_csv('Womens Clothing E-Commerce Reviews.csv')
```

In [6]:
```
df.head()
```

Out[6]:

	Unnamed: 0	Clothing ID	Age	Title	Review Text	Rating	Recommended IND	Positive Feedback Count	Division Name	D
0	0	767	33	NaN	Absolutely wonderful - silky and sexy and comf...	4	1	0	Initmates	
1	1	1080	34	NaN	Love this dress! it's sooo pretty. i happene...	5	1	4	General	
2	2	1077	60	Some major design flaws	I had such high hopes for this dress and reall...	3	0	0	General	
3	3	1049	50	My favorite buy!	I love, love, love this jumpsuit. it's fun, fl...	5	1	0	General Petite	
4	4	847	47	Flattering shirt	This shirt is very flattering to all due to th...	5	1	6	General	

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

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 23486 entries, 0 to 23485
Data columns (total 11 columns):
 #   Column                   Non-Null Count  Dtype 
---  ------                   --------------  ----- 
 0   Unnamed: 0               23486 non-null  int64 
 1   Clothing ID              23486 non-null  int64 
 2   Age                      23486 non-null  int64 
 3   Title                    19676 non-null  object
 4   Review Text              22641 non-null  object
 5   Rating                   23486 non-null  int64 
 6   Recommended IND          23486 non-null  int64 
 7   Positive Feedback Count  23486 non-null  int64 
 8   Division Name            23472 non-null  object
 9   Department Name          23472 non-null  object
 10  Class Name               23472 non-null  object
dtypes: int64(6), object(5)
memory usage: 2.0+ MB
```

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

Out[8]:
```
(23486, 11)
```

In [9]:
```
df.isna().sum()
```

Out[9]:
```
Unnamed: 0                   0
Clothing ID                  0
Age                          0
Title                     3810
Review Text                845
Rating                       0
Recommended IND              0
Positive Feedback Count      0
Division Name               14
Department Name             14
Class Name                  14
dtype: int64
```

In [10]:
```
df[df['Review Text']==""]=np.NAN
```

In [11]:
```
df['Review Text'].fillna("No Review Text",inplace=True)
```

In [12]:
```
df.isna().sum()
```

Out[12]:
```
Unnamed: 0                    0
Clothing ID                   0
Age                           0
Title                      3810
Review Text                   0
Rating                        0
Recommended IND               0
Positive Feedback Count       0
Division Name                14
Department Name              14
Class Name                   14
dtype: int64
```

In [13]:
```
df['Review Text']
```

Out[13]:
```
0        Absolutely wonderful - silky and sexy and comf...
1        Love this dress!  it's sooo pretty.  i happene...
2        I had such high hopes for this dress and reall...
3        I love, love, love this jumpsuit. it's fun, fl...
4        This shirt is very flattering to all due to th...
                               ...
23481    I was very happy to snag this dress at such a ...
23482    It reminds me of maternity clothes. soft, stre...
23483    This fit well, but the top was very see throug...
23484    I bought this dress for a wedding i have this ...
23485    This dress in a lovely platinum is feminine an...
Name: Review Text, Length: 23486, dtype: object
```

Define targer(y) and feature(x)

In [14]:
```
df.columns
```

Out[14]:
```
Index(['Unnamed: 0', 'Clothing ID', 'Age', 'Title', 'Review Text', 'Rating',
       'Recommended IND', 'Positive Feedback Count', 'Division Name',
       'Department Name', 'Class Name'],
      dtype='object')
```

In [15]:
```
x = df['Review Text']
```

In [16]:
```
y = df['Rating']
```

In [17]:
```
df['Rating'].value_counts()
```

Out[17]:
```
5.0    13131
4.0     5077
3.0     2871
2.0     1565
1.0      842
Name: Rating, dtype: int64
```

Train Test split

In [18]:
```
from sklearn.model_selection import train_test_split
```

In [19]:
```
x_train, x_test, y_train,y_test =train_test_split(x, y, train_size = 0.7, stra
```

In [20]:
```
x_train.shape, x_test.shape, y_train.shape, y_test.shape
```

Out[20]:
```
((16440,), (7046,), (16440,), (7046,))
```

Get Feature Text Conversion To Tokens

In [21]:
```
from sklearn.feature_extraction.text import CountVectorizer
```

In [22]:
```
cv = CountVectorizer(lowercase = True, analyzer='word', ngram_range=(2,3), sto
```

In [23]:
```
x_train = cv.fit_transform(x_train)
```

In [24]:
```
cv.get_feature_names_out()
```

Out[24]:
```
array(['10 12', '10 bought', '10 fit', ..., 'yellow color', 'yoga pants',
       'zipper little'], dtype=object)
```

In [25]:
```
x_train.toarray()
```

Out[25]:
```
array([[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, 0]], dtype=int64)
```

In [26]:
```
x_test = cv.fit_transform(x_test)
```

In [27]:
```
cv.get_feature_names_out()
```

Out[27]:
```
array(['10 12', '10 dress', '10 fit', ..., 'years come', 'years old',
       'yoga pants'], dtype=object)
```

In [28]:
```
x_test.toarray()
```

Out[28]:
```
array([[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, 0]], dtype=int64)
```

Get Model Train

In [29]:
```
from sklearn.naive_bayes import MultinomialNB
```

In [30]:
```
model = MultinomialNB()
```

In [31]:
```
model.fit(x_train, y_train)
```

Out[31]:
```
MultinomialNB()
```

Get Model Prediction

In [32]:
```
y_pred = model.predict(x_test)
```

In [33]:
```
y_pred.shape
```

Out[33]:
```
(7046,)
```

In [34]:
```
y_pred
```

Out[34]:
```
array([4., 4., 5., ..., 5., 5., 4.])
```

Get Probability Of Each Predicted Class

In [35]:
```
model.predict_proba(x_test)
```

Out[35]:
```
array([[9.61627846e-02, 1.21587107e-01, 2.61053999e-01, 4.15632757e-01,
        1.05563352e-01],
       [1.52260754e-01, 2.00471990e-02, 2.96168321e-01, 3.28511250e-01,
        2.03012477e-01],
       [3.45951842e-02, 4.19250461e-02, 1.46458042e-01, 2.58410083e-01,
        5.18611644e-01],
       ...,
       [4.40456735e-02, 1.13409468e-02, 2.65711717e-02, 7.56260955e-02,
        8.42416112e-01],
       [7.25608901e-03, 3.03718334e-04, 5.58927047e-03, 4.54625587e-01,
        5.32225335e-01],
       [1.30506112e-01, 7.21920545e-02, 1.13371567e-02, 4.43923937e-01,
        3.42040740e-01]])
```

Get Model Evaluation

In [36]:
```
from sklearn.metrics import confusion_matrix, classification_report
```

In [37]:
```
print(confusion_matrix(y_test,y_pred))
```

```
[[  25   25   45   33  125]
 [  41   55   67   84  223]
 [  76   85  140  181  379]
 [ 174  108  185  364  692]
 [ 440  246  354  681 2218]]
```

In [38]:
```
print(classification_report(y_test,y_pred))
```

```
              precision    recall  f1-score   support

         1.0       0.03      0.10      0.05       253
         2.0       0.11      0.12      0.11       470
         3.0       0.18      0.16      0.17       861
         4.0       0.27      0.24      0.25      1523
         5.0       0.61      0.56      0.59      3939

    accuracy                           0.40      7046
   macro avg       0.24      0.24      0.23      7046
weighted avg       0.43      0.40      0.41      7046
```

Recategories Rating as Poor(0) and Good(1)

In [39]:
```
df['Rating'].value_counts()
```

Out[39]:
```
5.0    13131
4.0     5077
3.0     2871
2.0     1565
1.0      842
Name: Rating, dtype: int64
```

Re-Rating as 1,2,3 as 0 and 4,5 as 1

In [40]:
```
df.replace({'Rating' : { 1 : 0, 2 : 0, 3 :0,4 : 1,5 : 1}}, inplace = True)
```

In [41]:
```
y = df['Rating']
```

In [42]:
```
x = df['Review Text']
```

Train Test Split

In [43]:
```
from sklearn.model_selection import train_test_split
```

In [44]:
```
x_train, x_test, y_train,y_test =train_test_split(x, y, train_size = 0.7, stra
```

In [45]:
```
x_train.shape, x_test.shape, y_train.shape, y_test.shape
```

Out[45]:
```
((16440,), (7046,), (16440,), (7046,))
```

Get Feature Text Conversion to Tokens

In [46]:
```
from sklearn.feature_extraction.text import CountVectorizer
```

In [47]:
```
cv = CountVectorizer(lowercase = True, analyzer='word', ngram_range=(2,3), sto
```

In [48]:
```
x_train = cv.fit_transform(x_train)
```

In [49]:
```
x_test = cv.fit_transform(x_test)
```

Get Model Re-Train

In [50]:
```
from sklearn.naive_bayes import MultinomialNB
```

In [51]:
```
model = MultinomialNB()
```

In [52]:
```
model.fit(x_train, y_train)
```

Out[52]:
```
MultinomialNB()
```

Get Model Prediction

In [53]:
```
y_pred = model.predict(x_test)
```

In [54]:
```
y_pred.shape
```

Out[54]:
```
(7046,)
```

In [55]:
```
y_pred
```

Out[55]:
```
array([1., 0., 1., ..., 1., 1., 1.])
```

Get Model Evaluation

In [56]:
```
from sklearn.metrics import confusion_matrix, classification_report
```

In [57]:
```
print(confusion_matrix(y_test,y_pred))
```

```
[[ 488 1095]
 [1079 4384]]
```

In [58]:
```
print(classification_report(y_test,y_pred))
```

```
              precision    recall  f1-score   support

         0.0       0.31      0.31      0.31      1583
         1.0       0.80      0.80      0.80      5463

    accuracy                           0.69      7046
   macro avg       0.56      0.56      0.56      7046
weighted avg       0.69      0.69      0.69      7046
```