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
```

In [5]:
```
df = pd.read_csv('https://raw.githubusercontent.com/Dhruv9hirud/datasets/main/Womens%20Clothing%20E-Commerce%20Reviews.csv')
```

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

Out[6]:

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

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.isna().sum()
```

Out[8]:
```
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 [9]:
```
df[df['Review Text']==""]=np.NaN
```

In [10]:
```
df['Review Text'].fillna("No Review",inplace=True)
```

In [11]:
```
df.isna().sum()
```

Out[11]:
```
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 [12]:
```
df['Review Text']
```

Out[12]:
```
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
```

In [13]:
```
df.columns
```

Out[13]:
```
Index(['Unnamed: 0', 'Clothing ID', 'Age', 'Title', 'Review Text', 'Rating',
       'Recommended IND', 'Positive Feedback Count', 'Division Name',
       'Department Name', 'Class Name'],
      dtype='object')
```

In [14]:
```
X = df['Review Text']
```

In [15]:
```
y = df['Rating']
```

In [16]:
```
df['Rating'].value_counts()
```

Out[16]:
```
5.0    13131
4.0     5077
3.0     2871
2.0     1565
1.0      842
Name: Rating, dtype: int64
```

In [17]:
```
from sklearn.model_selection import train_test_split
```

In [18]:
```
X_train, X_test, y_train, y_test = train_test_split(X, y, train_size=0.7, stratify=y,random_state=22529)
```

In [19]:
```
from sklearn.feature_extraction.text import CountVectorizer
```

In [20]:
```
cv = CountVectorizer(lowercase = True, analyzer='word', ngram_range=(2,3),stop_words='english',max_features=5000)
```

In [21]:
```
X_train = cv.fit_transform(X_train)
```

In [22]:
```
cv.get_feature_names_out()
```

Out[22]:
```
array(['00 petite', '0p fit', '10 12', ..., 'years old', 'yellow color',
       'yoga pants'], dtype=object)
```

In [23]:
```
X_train.toarray()
```

Out[23]:
```
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 [24]:
```
X_test = cv.fit_transform(X_test)
```

In [25]:
```
cv.get_feature_names_out()
```

Out[25]:
```
array(['10 12', '10 bought', '10 fit', ..., 'years come', 'years old',
       'yoga pants'], dtype=object)
```

In [26]:
```
X_test.toarray()
```

Out[26]:
```
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 [27]:
```
from sklearn.naive_bayes import MultinomialNB
```

In [28]:
```
model = MultinomialNB()
```

In [29]:
```
model.fit(X_train, y_train)
```

Out[29]:
```
▼ MultinomialNB
MultinomialNB()
```

In [30]:
```
y_pred = model.predict(X_test)
```

In [31]:
```
y_pred
```

Out[31]:
```
array([3., 5., 5., ..., 5., 5., 5.])
```

In [32]:
```
model.predict_proba(X_test)
```

Out[32]:
```
array([[2.24971416e-02, 1.08193100e-02, 5.67447201e-01, 1.71827961e-01,
        2.27408387e-01],
       [5.00003189e-03, 9.04432823e-04, 1.07939724e-02, 7.03172752e-02,
        9.12984288e-01],
       [5.10285019e-02, 2.71045644e-03, 3.32973245e-02, 4.32925847e-02,
        8.69671132e-01],
       ...,
       [1.56434488e-03, 5.85597793e-03, 2.39623618e-04, 5.50498737e-02,
        9.37290180e-01],
       [7.17740944e-02, 1.07387794e-01, 4.13796902e-02, 3.57360775e-01,
        4.22117736e-01],
       [2.84309552e-01, 3.35589644e-03, 7.49210615e-02, 2.84713169e-01,
        3.52700321e-01]])
```

In [33]:
```
from sklearn.metrics import confusion_matrix, classification_report
```

In [34]:
```
print(confusion_matrix(y_test, y_pred))
```

```
[[  17   23   31   41  141]
 [  44   63   67   72  224]
 [  81   99  155  148  378]
 [ 158  149  215  282  719]
 [ 395  304  361  631 2248]]
```

In [35]:
```
print(classification_report(y_test,y_pred))
```

```
              precision    recall  f1-score   support

         1.0       0.02      0.07      0.04       253
         2.0       0.10      0.13      0.11       470
         3.0       0.19      0.18      0.18       861
         4.0       0.24      0.19      0.21      1523
         5.0       0.61      0.57      0.59      3939

    accuracy                           0.39      7046
   macro avg       0.23      0.23      0.23      7046
weighted avg       0.42      0.39      0.41      7046
```

In [36]:
```
df['Rating'].value_counts()
```

Out[36]:
```
5.0    13131
4.0     5077
3.0     2871
2.0     1565
1.0      842
Name: Rating, dtype: int64
```

In [37]:
```
df.replace({'Rating': {1:0,2:0,3:0,4:1,5:1}}, inplace=True)
```

In [38]:
```
y=df['Rating']
```

In [39]:
```
X=df['Review Text']
```

In [40]:
```
from sklearn.model_selection import train_test_split
```

In [41]:
```
X_train, X_test, y_train, y_test = train_test_split(X,y,train_size=0.7,random_state=22529)
```

In [42]:
```
from sklearn.feature_extraction.text import CountVectorizer
```

In [43]:
```
cv = CountVectorizer(lowercase=True,analyzer='word',ngram_range=(2,3),stop_words='english',max_features=5000)
```

In [44]:
```
X_train = cv.fit_transform(X_train)
```

In [45]:
```
X_test = cv.fit_transform(X_test)
```

In [46]:
```
from sklearn.naive_bayes import MultinomialNB
```

In [47]:
```
model = MultinomialNB()
```

In [48]:
```
model.fit(X_train, y_train)
```

Out[48]:
```
▼ MultinomialNB
MultinomialNB()
```

In [49]:
```
y_pred = model.predict(X_test)
```

In [50]:
```
y_pred
```

Out[50]:
```
array([0., 1., 1., ..., 1., 1., 0.])
```

In [51]:
```
from sklearn.metrics import confusion_matrix, classification_report
```

In [52]:
```
print(confusion_matrix(y_test,y_pred))
```

```
[[ 449 1139]
 [ 925 4533]]
```

In [53]:
```
print(classification_report(y_test, y_pred))
```

```
              precision    recall  f1-score   support

         0.0       0.33      0.28      0.30      1588
         1.0       0.80      0.83      0.81      5458

    accuracy                           0.71      7046
   macro avg       0.56      0.56      0.56      7046
weighted avg       0.69      0.71      0.70      7046
```

In [ ]:
```

```