Women Cloth Review Prediction With Naive Bayes

DataSet : Womens Clothing E-Commerce Reviews.csv from kaagle.com (https://www.kaggle.com/datasets/yasserhessein/womens-clothing-ecommerce-reviews)

Import Library

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

Import Data

```
df=pd.read_csv("Womens Clothing E-Commerce Reviews.csv")
df.head()
```

	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

```
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
```

```
df.shape
```

```
(23486, 11)
```

```
df.isna().sum()
```

```
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
```

```
df['Review Text']
```

```
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
```

```
df.columns
```

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

```
x=df['Review Text']
y=df['Rating']
```

```
df['Rating'].value_counts()
```

```
Rating
5    13131
4     5077
3     2871
2     1565
1      842
Name: count, dtype: int64
```

Train Test Split

```
Rating=df['Rating']
from sklearn.model_selection import train_test_split
X_train , X_test , y_train , y_test = train_test_split(x,y, test_size = 0.7,stratify = y, random_state = 2529)
```

```
X_train.shape
```

```
(7045,)
```

```
from sklearn.feature_extraction.text import CountVectorizer
cv=CountVectorizer(lowercase=True,analyzer='word',ngram_range=(2,3),stop_words='english',max_features=5000)
X_train=cv.fit_transform(X_train.values.astype('U'))
cv.get_feature_names_out()
X_train.toarray()
```

```
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]])
```

```
X_test=cv.fit_transform(X_test.values.astype('U'))
cv.get_feature_names_out()
X_test.toarray()
```

```
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]])
```

Model

```
from sklearn.naive_bayes import MultinomialNB
model=MultinomialNB()
model.fit(X_train,y_train)
```

```
MultinomialNB
MultinomialNB()
```

```
y_pred=model.predict(X_test)
y_pred.shape
```

```
(16441,)
```

```
model.predict_proba(X_test)
```

```
array([[2.50243939e-01, 1.21534354e-01, 4.79327448e-01, 1.27392454e-02,
        1.36155013e-01],
       [7.07567572e-02, 4.58818848e-02, 2.63088820e-01, 6.12325394e-01,
        7.94714345e-03],
       [2.01162807e-01, 3.38200504e-01, 2.95746505e-01, 9.24481824e-02,
        7.24420016e-02],
       ...,
       [3.91842500e-03, 2.98245742e-03, 3.09715894e-04, 7.72959108e-03,
        9.85059811e-01],
       [1.61530659e-01, 3.51476805e-02, 5.52149762e-02, 5.43909305e-01,
        2.04197379e-01],
       [1.26442837e-01, 4.89792929e-02, 3.28750971e-01, 2.68648730e-01,
        2.27178169e-01]])
```

Evaluation

```
from sklearn.metrics import confusion_matrix,classification_report
print(confusion_matrix(y_test,y_pred))
#print(classification_report(y_test,y_pred))
```

```
[[  37   76   66  130  280]
 [  83  166  182  203  462]
 [ 196  285  342  396  791]
 [ 386  392  457  695 1624]
 [ 885  837  963 1584 4923]]
```

```
from sklearn.metrics import accuracy_score;
from sklearn import metrics
print("Gaussian Naive Bayes model accuracy(in %):",(metrics.accuracy_score(y_test,y_pred )*100).round(2))
```

```
Gaussian Naive Bayes model accuracy(in %): 37.49
```

ReTrain

```
df['Rating'].value_counts()
```

```
Rating
5    13131
4     5077
3     2871
2     1565
1      842
Name: count, dtype: int64
```

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

```
y=df['Rating']
x=df['Review Text']
```

```
from sklearn.model_selection import train_test_split
X_train , X_test , y_train , y_test = train_test_split(x,y, test_size = 0.7,stratify = y, random_state = 2529)
```

```
from sklearn.feature_extraction.text import CountVectorizer
cv=CountVectorizer(lowercase=True,analyzer='word',ngram_range=(2,3),stop_words='english',max_features=5000)
X_train=cv.fit_transform(X_train.values.astype('U'))
X_test=cv.fit_transform(X_test.values.astype('U'))
```

```
model=MultinomialNB()
model.fit(X_train,y_train)
```

```
▾ MultinomialNB
MultinomialNB()
```

```
y_pred=model.predict(X_test)
```

```
y_pred.shape
```

```
(16441,)
```

```
y_pred
```

```
array([0, 0, 1, ..., 1, 1, 0])
```

Improved Model Evalauation

```
from sklearn.metrics import accuracy_score;
from sklearn import metrics
print("Gaussian Naive Bayes model accuracy(in %):",(metrics.accuracy_score(y_test,y_pred )*100).round(2))
```

```
Gaussian Naive Bayes model accuracy(in %): 67.61
```