Clothing Review Project

Objective: The objective of womens clothing e-commerce review is to analyse the rating of clothing based on their review and predict the rating based on their review.

Data Source: Ybifoundation Github

Import library

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

Data Import

```
df=pd.read_csv("https://github.com/YBIFoundation/MachineLearning/raw/main/Dataset/Women%20Clothing%20E-Commerce%20Review.csv")
```

```
df
```

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

Next steps: Generate code with df | View recommended plots | New interactive sheet

Describing Data

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 23486 entries, 0 to 23485
Data columns (total 10 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   Clothing ID        23486 non-null  int64
 1   Age                23486 non-null  int64
 2   Title              19676 non-null  object
 3   Review             22641 non-null  object
 4   Rating             23486 non-null  int64
 5   Recommended        23486 non-null  int64
 6   Positive Feedback  23486 non-null  int64
 7   Division           23472 non-null  object
 8   Department         23472 non-null  object
 9   Category           23472 non-null  object
dtypes: int64(5), object(5)
memory usage: 1.8+ MB
```

```
df.describe()
```

	Clothing ID	Age	Rating	Recommended	Positive Feedback
count	23486.000000	23486.000000	23486.000000	23486.000000	23486.000000
mean	918.118709	43.198544	4.196032	0.822362	2.535936
std	203.298980	12.279544	1.110031	0.382216	5.702202
min	0.000000	18.000000	1.000000	0.000000	0.000000
25%	861.000000	34.000000	4.000000	1.000000	0.000000
50%	936.000000	41.000000	5.000000	1.000000	1.000000
75%	1078.000000	52.000000	5.000000	1.000000	3.000000
max	1205.000000	99.000000	5.000000	1.000000	122.000000

```
df.shape
```

```
(23486, 10)
```

Data Preprocessing

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

```
Clothing ID          0
Age                  0
Title                0
Review               0
Rating               0
Recommended          0
Positive Feedback    0
Division             0
Department           0
Category             0
dtype: int64
```

```
df.dropna()
```

	Clothing ID	Age	Title	Review	Rating	Recommended	Positive Feedback	Division	Department	Category
2	1077	60	Some major design flaws	I had such high hopes for this dress and reall...	3	0	0	General	Dresses	Dresses
3	1049	50	My favorite buy!	I love, love, love this jumpsuit. it's fun, fl...	5	1	0	General Petite	Bottoms	Pants
4	847	47	Flattering shirt	This shirt is very flattering to all due to th...	5	1	6	General	Tops	Blouses
5	1080	49	Not for the very petite	I love tracy reese dresses, but this one is no...	2	0	4	General	Dresses	Dresses
6	858	39	Cagrcoal shimmer fun	I aded this in my basket at hte last mintue to...	5	1	1	General Petite	Tops	Knits
...	...	...	...	...	...	...	...	...	...	...
23481	1104	34	Great dress for many occasions	I was very happy to snag this dress at such a ...	5	1	0	General Petite	Dresses	Dresses
23482	862	48	Wish it was made of cotton	It reminds me of maternity clothes. soft, stre...	3	1	0	General Petite	Tops	Knits
23483	1104	31	Cute, but see through	This fit well, but the top was very see	3	0	1	General Petite	Dresses	Dresses

```
df.dropna(inplace=True)
```

```
df.columns
```

```
Index(['Clothing ID', 'Age', 'Title', 'Review', 'Rating', 'Recommended',
       'Positive Feedback', 'Division', 'Department', 'Category'],
      dtype='object')
```

**Defining X and Y **

```
y=df['Rating']
X=df['Review']
```

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

```
Rating
5    10858
4     4289
3     2464
2     1360
1      691
Name: count, dtype: int64
```

Train test split

```
from sklearn.model_selection import train_test_split

X_train,X_test,y_train,y_test = train_test_split(X,y,random_state=2529,stratify=y)
```

```
X_train.shape,X_test.shape,y_train.shape,y_test.shape
```

```
((14746,), (4916,), (14746,), (4916,))
```

Text Conversion

```
from sklearn.feature_extraction.text import CountVectorizer

cv=CountVectorizer(lowercase=True,analyzer='word',ngram_range=(1,2),stop_words='english',max_features=10000)
```

```
X_train =cv.fit_transform(X_train)
```

```
cv.get_feature_names_out()
```

```
array(['00', '00p', '00p 0p', ..., 'zippers', 'zipping', 'zips'],
      dtype=object)
```

```
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)
```

```
cv.get_feature_names_out()
```

```
array(['00', '00 petite', '00p', ..., 'zippers', 'zipping', 'zone'],
      dtype=object)
```

```
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 train

```
from sklearn.naive_bayes import MultinomialNB
model=MultinomialNB()
```

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

```
▾ MultinomialNB
MultinomialNB()
```

Model Prediction

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

```
y_pred.shape
```

```
(4916,)
```

```
y_pred
```

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

PROBABLITY OF EACH PREDICTED CLASS

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

```
array([[9.99993889e-01, 6.10206241e-06, 2.86834830e-09, 1.02345061e-09,
        4.76838824e-09],
       [9.99686749e-01, 3.12745827e-04, 4.88630525e-07, 1.64625702e-08,
        2.51676028e-10],
       [9.99942381e-01, 5.69211010e-05, 1.32999534e-08, 6.09621620e-07,
        7.53292617e-08],
       ...,
       [8.33285307e-01, 5.82354770e-03, 1.60567359e-01, 1.98224298e-04,
        1.25562130e-04],
       [5.59396818e-01, 4.40363788e-01, 2.34243177e-04, 3.19917057e-06,
        1.95122831e-06],
       [6.12274880e-01, 1.35428099e-01, 5.78748754e-03, 7.02250842e-02,
        1.76284450e-01]])
```

Model Evaluation

```
from sklearn.metrics import classification_report
```

```
print(classification_report(y_test,y_pred))
```

```
              precision    recall  f1-score   support

           1       0.04      0.94      0.07       173
           2       0.11      0.06      0.08       340
           3       0.17      0.00      0.01       616
           4       0.24      0.02      0.04      1072
           5       0.67      0.03      0.05      2715

    accuracy                           0.06      4916
   macro avg       0.25      0.21      0.05      4916
weighted avg       0.45      0.06      0.05      4916
```

Recategorising rating as 0 and 1

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

```
y=df['Rating']
X=df['Review']
```

train test split

```
from sklearn.model_selection import train_test_split
```

```
X_train,X_test,y_train,y_test = train_test_split(X,y,random_state=2529,stratify=y)
```

text conversion to token

```
from sklearn.feature_extraction.text import CountVectorizer
cv=CountVectorizer(lowercase=True,analyzer='word',ngram_range=(1,2),stop_words="english",max_features=10000)
```

```
X_train=cv.fit_transform(X_train)
```

```
X_test=cv.fit_transform(X_test)
```

Model Retrain

```
from sklearn.naive_bayes import MultinomialNB
```

```
model=MultinomialNB()
```

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

```
▾ MultinomialNB
MultinomialNB()
```

Model prediction

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

```
y_pred.shape
```

```
(4916,)
```

```
y_pred
```

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

**EVALUATION **

```
from sklearn.metrics import classification_report
```

```
print (classification_report(y_test,y_pred))
```

```
              precision    recall  f1-score   support

           0       0.23      0.32      0.27      1129
           1       0.77      0.68      0.72      3787

    accuracy                           0.59      4916
   macro avg       0.50      0.50      0.49      4916
weighted avg       0.64      0.59      0.62      4916
```

Explainantion:

By analysing and preprocessing data if any null value is their by droping null value and selecting target variable [y="rating"] and [x="Review"] and training testing a model. and The CountVectorizer from sklearn.feature_extraction.text is used to convert text reviews into numerical data. A Naive Bayes classifier (MultinomialNB) is trained on the training data. Predictions are made on the test data. After recategorizing the ratings, the model shows improved performance with a weighted average accuracy of 59%.

Start coding or generate with AI.

Start coding or generate with AI.