OBJECTIVE

The objective is to develop a machine learning model in Python that predicts the sentiment (positive or negative) of women's clothing reviews based on textual data. This involves preprocessing the text data, extracting features, and training a model to classify review sentiments accurately. The final model will be used to automatically assess the sentiment of new clothing reviews.

DATA SOURCE

Data Source: YBIFoundation/ProjectHub-MachineLearning Women Clothing Commerce Review dataset

Importing library

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

Import data

```
df = pd.read_csv("https://raw.githubusercontent.com/YBIFoundation/ProjectHub-MachineLearni
```

Describe data

```
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

Data visualization

Samples of the data is visualized to better understand how it is structured.

```
df.head()
```

	Clothing ID	Age	Title	Review	Rating	Recommended	Positive Feedback	Division	Depa
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	5	1	4	General	

```
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.shape
```

```
(23486, 10)
```

Data Preprocessing

Calling isna() method along with the sum() method on dataframe df to find the Review columns with no review text for further processing.

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

```
Clothing ID            0
Age                    0
Title               3810
Review               845
Rating                 0
Recommended            0
Positive Feedback      0
Division              14
Department            14
Category              14
dtype: int64
```

Define Target variable(y) and feature variable(x)

```
df[df['Review']==""] = np.NaN
df['Review'].fillna("No review is given", inplace=True)
df.isna().sum()
```

```
Clothing ID            0
Age                    0
Title               3810
Review                 0
Rating                 0
Recommended            0
Positive Feedback      0
Division              14
Department            14
Category              14
dtype: int64
```

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

```
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, Length: 23486, dtype: object
```

```
df.columns
```

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

```
x = df['Review']
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

```
from sklearn.model_selection import train_test_split

x_train, x_test, y_train, y_test = train_test_split(x, y, train_size=0.7, stratify=y, rand
x_train.shape, x_test.shape, y_train.shape, y_test.shape
```

```
((16440,), (7046,), (16440,), (7046,))
```

```
from sklearn.feature_extraction.text import CountVectorizer

cv = CountVectorizer(lowercase=True, analyzer='word', ngram_range=(2, 3), stop_words='engl
x_train = cv.fit_transform(x_train)
cv.get_feature_names_out()
```

```
array(['00 big', '00 fits', '00 petite', ..., 'zipper zip',
       'zippered pockets', 'zippers buttons'], 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 24', '00 petite', '00 sold', ..., 'zipper split',
       'zipper sturdy', 'zippers buttons'], 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 training

Using Multinomial Naïve Bayes algorithm, which is implemented in sci-kit as MultinomialNB

```
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
```

```
(7046,)
```

```
y_pred
```

```
array([1, 2, 5, ..., 4, 3, 1])
```

```
model.predict_proba(x_test)
```

```
array([[0.45433767, 0.08215679, 0.44111678, 0.00904855, 0.01334022],
       [0.08538546, 0.53197259, 0.36096501, 0.00370374, 0.01797319],
       [0.04978519, 0.07954803, 0.11784385, 0.31199241, 0.44083052],
       ...,
       [0.07871327, 0.0343138 , 0.02156397, 0.8608945 , 0.00451447],
       [0.09545745, 0.00239741, 0.84956399, 0.01241549, 0.04016567],
       [0.65456291, 0.01868614, 0.14266667, 0.04591333, 0.13817096]])
```

Model evaluation

```
from sklearn.metrics import confusion_matrix, classification_report

print(confusion_matrix(y_test, y_pred))
```

```
[[  65   41   46   37   64]
 [ 158   75   57   80  100]
 [ 280  176  138  115  152]
 [ 539  297  217  201  269]
 [1237  777  619  533  773]]
```

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

```
              precision    recall  f1-score   support

           1       0.03      0.26      0.05       253
           2       0.05      0.16      0.08       470
           3       0.13      0.16      0.14       861
           4       0.21      0.13      0.16      1523
           5       0.57      0.20      0.29      3939

    accuracy                           0.18      7046
   macro avg       0.20      0.18      0.15      7046
weighted avg       0.38      0.18      0.22      7046
```

```
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']
```

Train Test Split

```
from sklearn.model_selection import train_test_split

x_train, x_test, y_train, y_test = train_test_split(x, y, train_size=0.7, stratify=y, rand
x_train.shape, x_test.shape, y_train.shape, y_test.shape
```

```
((16440,), (7046,), (16440,), (7046,))
```

```
from sklearn.feature_extraction.text import CountVectorizer

cv = CountVectorizer(lowercase=True, analyzer='word', ngram_range=(2, 3), stop_words='engl
x_train = cv.fit_transform(x_train)
x_test = cv.fit_transform(x_test)
```

```
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
```

```
(7046,)
```

```
y_pred
```

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

Model evaluation

```
from sklearn.metrics import confusion_matrix, classification_report

print(confusion_matrix(y_test, y_pred))

print(confusion_matrix(y_test, y_pred))
```

```
[[ 712  871]
 [2643 2820]]
[[ 712  871]
 [2643 2820]]
```

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

```
              precision    recall  f1-score   support
```

```
           0       0.21      0.45      0.29      1583
           1       0.76      0.52      0.62      5463

    accuracy                           0.50      7046
   macro avg       0.49      0.48      0.45      7046
weighted avg       0.64      0.50      0.54      7046
```

Explanation

This project is focused on building a prediction model. At first, the all required libraries and a test dataset are imported. The dataset was evaluated and pre processed to prepare for it for processing, then a portion of it was kept for testing and the rest was used to train the model. The model was used to get some prediction dataset. Finnaly, prediction accuracy was checked against the test dataset, some adjusment were made and the model was re-trained for better accuracy.