```
import pandas as pd

df=pd.read_csv('Reviews.csv')

df
```

```
             Id   ProductId          UserId
ProfileName  \
0             1  B001E4KFG0  A3SGXH7AUHU8GW
delmartian
1             2  B00813GRG4  A1D87F6ZCVE5NK
dll pa
2             3  B000LQOCH0   ABXLMWJIXXAIN  Natalia Corres "Natalia
Corres"
3             4  B000UA0QIQ  A395BORC6FGVXV
Karl
4             5  B006K2ZZ7K  A1UQRSCLF8GW1T    Michael D. Bigham "M.
Wassir"
...         ...         ...             ...
...
568449  568450  B001EO7N10  A28KG5XORO54AY                  Lettie D.
Carter
568450  568451  B003S1WTCU  A3I8AFVPEE8KI5                       R.
Sawyer
568451  568452  B004I613EE  A121AA1GQV751Z                    pksd
"pk_007"
568452  568453  B004I613EE   A3IBEVCTXKNOH          Kathy A. Welch
"katwel"
568453  568454  B001LR2CU2  A3LGQPJCZVL9UC
srfell17

        HelpfulnessNumerator  HelpfulnessDenominator  Score
Time  \
0                          1                       1      5
1303862400
1                          0                       0      1
1346976000
2                          1                       1      4
1219017600
3                          3                       3      2
1307923200
4                          0                       0      5
1350777600
...                      ...                     ...    ...          ..
.
568449                     0                       0      5
1299628800
568450                     0                       0      2
1331251200
568451                     2                       2      5
```

```
1329782400
568452                      1                       1      5
1331596800
568453                      0                       0      5
1338422400

                                    Summary  \
0                    Good Quality Dog Food
1                        Not as Advertised
2                    "Delight" says it all
3                           Cough Medicine
4                              Great taffy
...                                    ...
568449                 Will not do without
568450                       disappointed
568451           Perfect for our maltipoo
568452  Favorite Training and reward treat
568453                       Great Honey

                                                     Text
0       I have bought several of the Vitality canned d...
1       Product arrived labeled as Jumbo Salted Peanut...
2       This is a confection that has been around a fe...
3       If you are looking for the secret ingredient i...
4       Great taffy at a great price.  There was a wid...
...                                                   ...
568449  Great for sesame chicken..this is a good if no...
568450  I'm disappointed with the flavor. The chocolat...
568451  These stars are small, so you can give 10-15 o...
568452  These are the BEST treats for training and rew...
568453  I am very satisfied ,product is as advertised,...

[568454 rows x 10 columns]
```

```
df.shape
```

```
(568454, 10)
```

```
df.columns
```

```
Index(['Id', 'ProductId', 'UserId', 'ProfileName',
'HelpfulnessNumerator',
       'HelpfulnessDenominator', 'Score', 'Time', 'Summary', 'Text'],
      dtype='object')
```

```
df= df[['Text', 'Score']]
df.head()
```

```
                                                Text  Score
0  I have bought several of the Vitality canned d...      5
1  Product arrived labeled as Jumbo Salted Peanut...      1
```

```
2  This is a confection that has been around a fe...      4
3  If you are looking for the secret ingredient i...      2
4  Great taffy at a great price.  There was a wid...      5
```

```
def get_sentiment(score):
    if score >= 4:
        return "Positive"
    elif score <= 2:
        return "Negative"
    else:
        return None

df['Sentiment'] = df['Score'].apply(get_sentiment)
df = df.dropna(subset=['Sentiment'])
```

```
/tmp/ipykernel_7279/524111217.py:9: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation:
https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#
returning-a-view-versus-a-copy
  df['Sentiment'] = df['Score'].apply(get_sentiment)
```

```
import re
import nltk
from nltk.corpus import stopwords
nltk.download('stopwords')

stop_words = set(stopwords.words('english'))

def clean_text(text):
    text = text.lower()
    text = re.sub(r'[^a-z\s]', '', text)
    tokens = text.split()
    tokens = [w for w in tokens if w not in stop_words]
    return ' '.join(tokens)

df['CleanText'] = df['Text'].apply(clean_text)
```

```
[nltk_data] Downloading package stopwords to /home/admin1/nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
```

```
from sklearn.feature_extraction.text import TfidfVectorizer

vectorizer = TfidfVectorizer(max_features=50000)
X = vectorizer.fit_transform(df['CleanText'])
y = df['Sentiment']

from sklearn.model_selection import train_test_split
```

```
X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.2, random_state=42
)

from sklearn.linear_model import LogisticRegression

model = LogisticRegression(max_iter=1000)
model.fit(X_train, y_train)
```

```
LogisticRegression(max_iter=1000)
```

```
from sklearn.metrics import accuracy_score, classification_report

y_pred = model.predict(X_test)

print("Accuracy:", accuracy_score(y_test, y_pred))
print(classification_report(y_test, y_pred))
```

```
Accuracy: 0.937744263666879
              precision    recall  f1-score   support

    Negative       0.87      0.71      0.78     16379
    Positive       0.95      0.98      0.96     88784

    accuracy                           0.94    105163
   macro avg       0.91      0.84      0.87    105163
weighted avg       0.94      0.94      0.94    105163
```

```
reviews_to_test = [
    "I absolutely love this product, great taste!",
    "Terrible experience, I will never buy it again.",
    "Average item — nothing special."
]

# Use the same clean_text function from training
clean_reviews = [clean_text(review) for review in reviews_to_test]

vectors = vectorizer.transform(clean_reviews)

predictions = model.predict(vectors)

for review, pred in zip(reviews_to_test, predictions):
    print(f"Review: {review}")
    print(f"Predicted Sentiment: {pred}")
    print("-" * 50)
```

```
Review: I absolutely love this product, great taste!
Predicted Sentiment: Positive
--------------------------------------------------
```

```
Review: Terrible experience, I will never buy it again.
Predicted Sentiment: Negative
--------------------------------------------------
Review: Average item — nothing special.
Predicted Sentiment: Negative
--------------------------------------------------
```