TWITTER SENTIMENT ANALYSIS BY USING NAÏVE BAYES BASED ON NATURAL LANGUAGE PROCESSING

INPUT

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

from nltk import word_tokenize, WordNetLemmatizer
from nltk.corpus import stopwords
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.model_selection import train_test_split, cross_val_score
from sklearn.naive_bayes import MultinomialNB, ComplementNB
from sklearn.metrics import classification_report, confusion_matrix,
f1_score, precision_score, \
    recall_score

plt.style.use('ggplot')

# DATA EXPLORATION
tweets_df = pd.read_csv('Training.csv', encoding='latin',
                        names=['sentiment', 'id', 'date', 'query', 'user', 'tweet'])
print(tweets_df)

tweets_df.info()

tweets_df = tweets_df.drop(['id'], axis=1)
print(tweets_df)
```

```
sns.heatmap(tweets_df.isnull(), yticklabels='False', cbar='False',
cmap='Blues')

tweets_df.hist(bins=30, figsize=(13, 5), color='r')
plt.show()

tweets_df['length'] = tweets_df['tweet'].apply(len)
print(tweets_df)

tweets_df['length'].plot(bins=80, figsize=(13, 5), kind='hist')
plt.show()

tweets_df.describe()
print(tweets_df.describe())

var1 = tweets_df[tweets_df['length'] == 15]['tweet'].iloc[0]  # shortest
comment
var2 = tweets_df[tweets_df['length'] == 115]['tweet'].iloc[0]  # longest
comment
var3 = tweets_df[tweets_df['length'] == 57]['tweet'].iloc[0]  # average
comment
print("Shortest Comment is = ", var1)
print("Longest Comment is = ", var2)
print("Average comment is = ", var3)

positive_comments = tweets_df[tweets_df['sentiment'] == 0]
print(positive_comments)
print("Positive Comments")
negative_comments = tweets_df[tweets_df['sentiment'] == 4]
print(negative_comments)
print("Negative Comments")


def username_remover(input_text, username):
```

```
    """removes the username handle from the data"""

    r = re.findall(username, input_text)
    for i in r:
        input_text = re.sub(i, '', input_text)
    return input_text


tweets_df['user_removed'] =
np.vectorize(username_remover)(tweets_df['tweet'], "@[\w]*")
print(tweets_df.head())

# PERFORMING DATA CLEANING, TOKENIZATION,
STEMMING AND COUNT VECTORIZATION
# Stop Words: A stop word is a commonly used word (such as “the”,
“a”, “an”, “in”)
# that a search engine has been programmed to ignore,
# both when indexing entries for searching and when retrieving them as
the result of a search query.
nltk.download('stopwords')
stopword = set(stopwords.words('english'))
print(stopword)

urlPattern = r"((http://)[^ ]*|(https://)[^ ]*|( www\.)[^ ]*)"
userPattern = '@[^\s]+'
some = 'amp,today,tomorrow,going,girl'


def process_tweets(tweet):
    # Removing all URls
    tweet = re.sub(urlPattern, '', tweet)
    # Removing all @username.
    tweet = re.sub(userPattern, '', tweet)
    # remove some words
    tweet = re.sub(some, '', tweet)
```

```
    # Remove punctuations
    tweet = tweet.translate(str.maketrans("", "", string.punctuation))
    # tokenizing words
    tokens = word_tokenize(tweet)
    tokens = [w for w in tokens if len(w) > 2]
    # Removing Stop Words
    final_tokens = [w for w in tokens if w not in stopword]
    # reducing a word to its word stem
    wordLemm = WordNetLemmatizer()
    finalwords = []
    for w in final_tokens:
        if len(w) > 1:
            word = wordLemm.lemmatize(w)
            finalwords.append(word)
    return ' '.join(finalwords)


tweets_df['clean_tweet'] =
tweets_df['user_removed'].apply(process_tweets)
print(tweets_df.head())

vectorizer = CountVectorizer(analyzer=process_tweets)
tokenized_tweets = vectorizer.fit_transform(tweets_df['clean_tweet'])
print(tokenized_tweets.shape)
X = tokenized_tweets
y = tweets_df['sentiment']

# TRAIN THE MODAL USING A NAIVE-BAYES CLASSIFIER
print(X.shape)
print(y.shape)

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2)
NB_classifier = MultinomialNB()
NB_classifier.fit(X_train, y_train)
```

```
# EVALUATING THE NAIVE-BAYES CLASSIFIER
PERFORMANCE
# Prediciting the test results
y_predict_test = NB_classifier.predict(X_test)
cm = confusion_matrix(y_test, y_predict_test)

group_names = ['True Pos', 'False Pos', 'False Neg', 'True Neg']

group_counts = ["{0:0.0f}".format(value) for value in cm.flatten()]
group_percentages = ["{0:.2%}".format(value) for value in cm.flatten() /
np.sum(cm)]
labels = [f'{v1}\n{v2}\n{v3}' for v1, v2, v3 in zip(group_names,
group_counts, group_percentages)]
labels = np.asarray(labels).reshape(2, 2)
sns.heatmap(cm, annot=labels, fmt='', cmap='Blues')
plt.show()

# Classification report
print(classification_report(y_test, y_predict_test))

cnb = ComplementNB()
cnb.fit(X_train, y_train)
cross_cnb = cross_val_score(cnb, X, y, n_jobs=-1)
print("Cross Validation score = ", cross_cnb)
print("Train accuracy = {:.2f}%".format(cnb.score(X_train, y_train) *
100))
print("Test accuracy = {:.2f}%".format(cnb.score(X_test, y_test) * 100))
train_acc_cnb = cnb.score(X_train, y_train)
test_acc_cnb = cnb.score(X_test, y_test)

# Predict test data set
y_predict_test = cnb.predict(X_test)

# Calculating Prescision scores, Recall scores and F1
print("Precision score = {:.2f}%".format(precision_score(y_test,
```

```
y_predict_test, average="macro") * 100))
precision_cnb = precision_score(y_test, y_predict_test,
average="macro")
print("Recall score = {:.2f}%".format(recall_score(y_test,
y_predict_test, average="macro") * 100))
recall_cnb = recall_score(y_test, y_predict_test, average="macro")
print("F1 score = {:.2f}%".format(f1_score(y_test, y_predict_test,
average="macro") * 100))
f1_cnb = f1_score(y_test, y_predict_test, average="macro")
```

OUTPUT

C:\Users\Admin\AppData\Local\Programs\Python\Python311\python.exe "G:\M.Tech Course & Syllabus 2020-2022\M.Tech Thesis\M.tech_Thesis_DP\Code\NaiveBayesModelling.py"

	sentiment	...	tweet
0	0	...	@switchfoot http://twitpic.com/2y1zl - Awww, t...
1	0	...	is upset that he can't update his Facebook by ...
2	0	...	@Kenichan I dived many times for the ball. Man...
3	0	...	my whole body feels itchy and like its on fire
4	0	...	@nationwideclass no, it's not behaving at all....
...	...	...	...
1599995	4	...	Just woke up. Having no school is the best fee...
1599996	4	...	TheWDB.com - Very cool to hear old Walt interv...

1599997 4 ... Are you ready for your MoJo Makeover? Ask me f...

1599998 4 ... Happy 38th Birthday to my boo of alll time!!! ...

1599999 4 ... happy #charitytuesday @theNSPCC @SparksCharity...

[1600000 rows x 6 columns]

<class 'pandas.core.frame.DataFrame'>

RangeIndex: 1600000 entries, 0 to 1599999

Data columns (total 6 columns):

#	Column	Non-Null Count	Dtype
0	sentiment	1600000 non-null	int64
1	id	1600000 non-null	int64
2	date	1600000 non-null	object
3	query	1600000 non-null	object
4	user	1600000 non-null	object
5	tweet	1600000 non-null	object

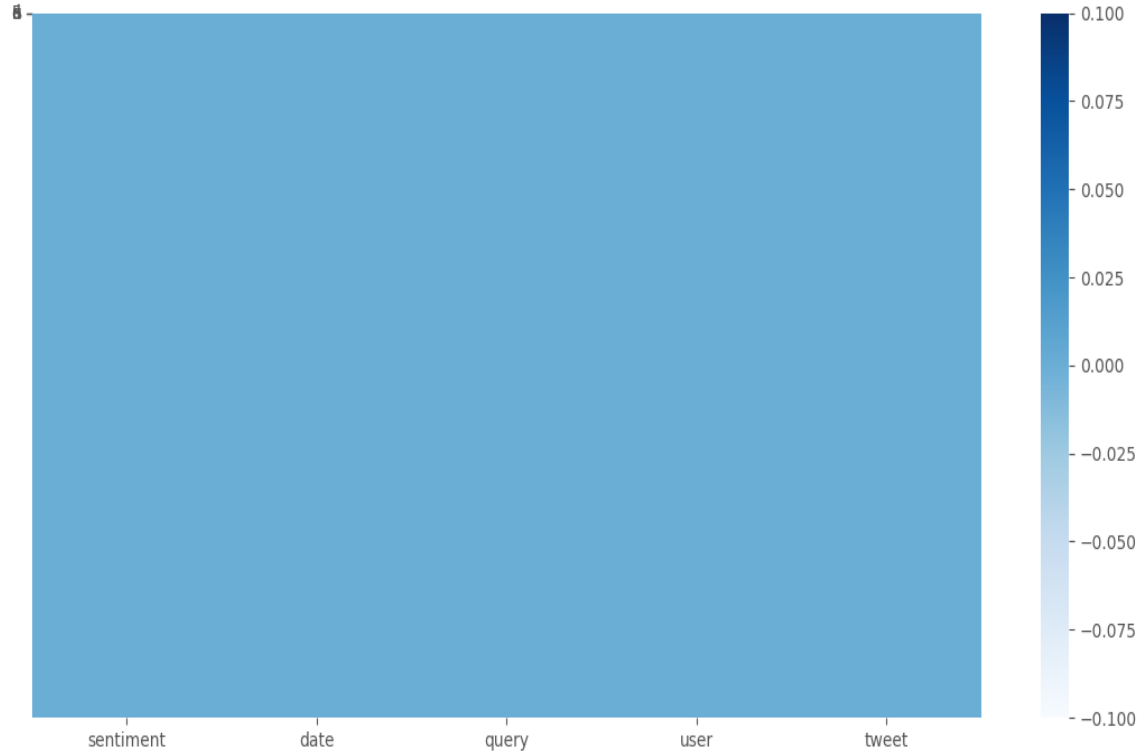
dtypes: int64(2), object(4)

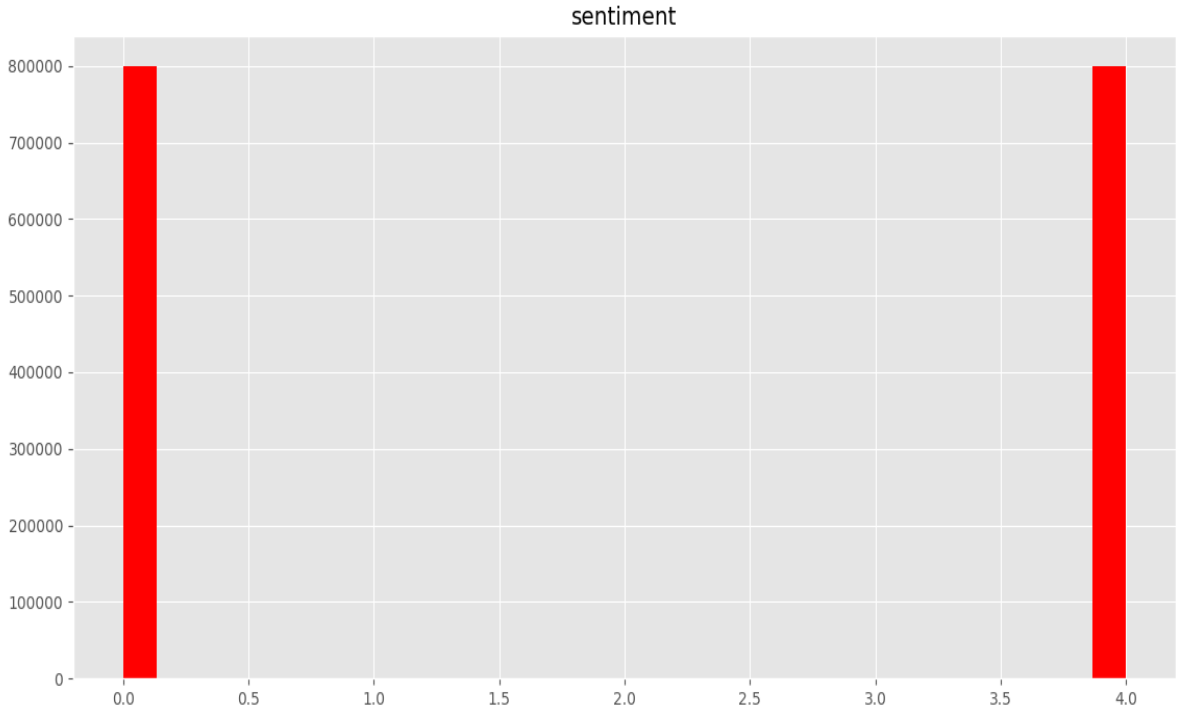
memory usage: 73.2+ MB

sentiment ... tweet

0	0	...	@switchfoot http://twitpic.com/2y1zl - Awww, t...	
1	0	...	is upset that he can't update his Facebook by ...	
2	0	...	@Kenichan I dived many times for the ball. Man...	
3	0	...	my whole body feels itchy and like its on fire	
4	0	...	@nationwideclass no, it's not behaving at all....	
...	...	...	...	
1599995	4	...	Just woke up. Having no school is the best fee...	
1599996	4	...	TheWDB.com - Very cool to hear old Walt interv...	
1599997	4	...	Are you ready for your MoJo Makeover? Ask me f...	
1599998	4	...	Happy 38th Birthday to my boo of alll time!!! ...	
1599999	4	...	happy #charitytuesday @theNSPCC @SparksCharity...	

[1600000 rows x 5 columns]


0
sentiment
date
query
user
tweet
0.100
0.075
0.050
0.025
0.000
−0.025
−0.050
−0.075
−0.100


sentiment
800000
700000
600000
500000
400000
300000
200000
100000
0
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
4.0

sentiment ... length

0 0 ... 115

1 0 ... 111

2 0 ... 89

3 0 ... 47

4 0 ... 111

... ...

1599995 4 ... 56

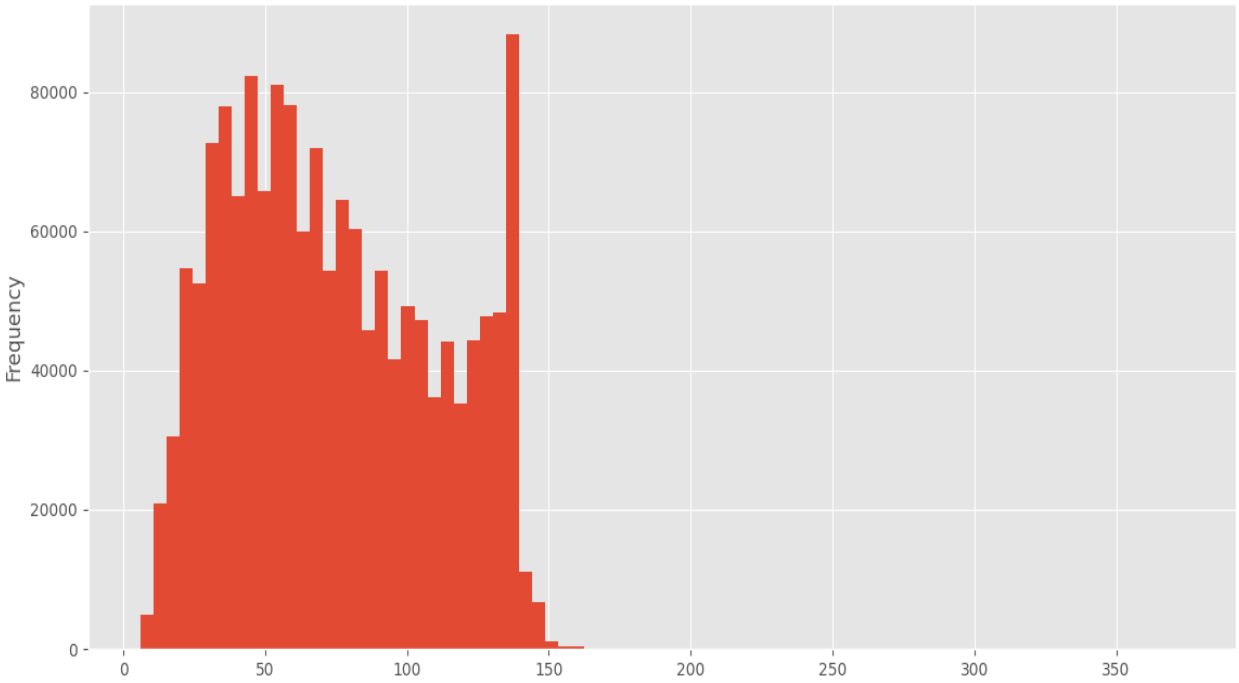
1599996 4 ... 78

1599997 4 ... 57

1599998 4 ... 65

1599999 4 ... 62

[1600000 rows x 6 columns]



	sentiment	length
count	1.600000e+06	1.600000e+06
mean	2.000000e+00	7.409011e+01
std	2.000001e+00	3.644114e+01
min	0.000000e+00	6.000000e+00
25%	0.000000e+00	4.400000e+01
50%	2.000000e+00	6.900000e+01
75%	4.000000e+00	1.040000e+02
max	4.000000e+00	3.740000e+02

Shortest Comment is = almost bedtime

Longest Comment is = @switchfoot http://twitpic.com/2y1zl - Awww, that's a bummer. You shoulda got David Carr of Third Day to do it. ;D

Average comment is = Damn... I don't have any chalk! MY CHALKBOARD IS USELESS

	sentiment	...	length
0	0	...	115
1	0	...	111
2	0	...	89
3	0	...	47
4	0	...	111
...	...	...	...
799995	0	...	63
799996	0	...	15
799997	0	...	29
799998	0	...	93
799999	0	...	82

[800000 rows x 6 columns]

Positive Comments

	sentiment	...	length
800000	4	...	44

800001 4 ... 72

800002 4 ... 137

800003 4 ... 104

800004 4 ... 47

... ...

1599995 4 ... 56

1599996 4 ... 78

1599997 4 ... 57

1599998 4 ... 65

1599999 4 ... 62

[800000 rows x 6 columns]

Negative Comments

	sentiment	...	user_removed
0	0	...	http://twitpic.com/2y1zl - Awww, that's a bum...
1	0	...	is upset that he can't update his Facebook by ...
2	0	...	I dived many times for the ball. Managed to s...
3	0	...	my whole body feels itchy and like its on fire
4	0	...	no, it's not behaving at all. i'm mad. why am...

[5 rows x 7 columns]

[nltk_data] Downloading package stopwords to

[nltk_data] C:\Users\Admin\AppData\Roaming\nltk_data...

[nltk_data] Package stopwords is already up-to-date!

{"shan't", "you've", "you're", 'both', 'll', 'won', 'are', 'did', "don't", 'will', 'been', 'themselves', 't', 'by', 'shan', 'should', 'wouldn', 'myself', 'further', 'why', 'so', 'then', 'himself', 'doing', "won't", 'their', 'before', 'once', 'i', 'she', 'they', 'until', "isn't", 'above', 'than', 'shouldn', "she's", 'these', 'under', 'those', 'after', 'any', 'such', 'him', 'isn', 'same', 'were', 'here', 'ma', 'didn', 'off', 'few', "couldn't", 'haven', 'having', 'o', 'it', "that'll", 'ain', 's', 'how', 'own', 'all', "hadn't", 'am', 'ourselves', "mustn't", 'where', 'my', "needn't", 'aren', 'below', 'as', 'nor', 'to', 'who', 'too', 'mustn', 'them', 'out', 'his', "didn't", 'we', 'have', "hasn't", 'itself', 'being', "should've", 'again', 'over', "doesn't", 'yours', 'what', 'about', 'weren', 'or', 'more', 'yourselves', 'some', 'your', 'when', 'has', 'not', 'an', 'only', 'hers', 'can', 'couldn', 'mightn', 'ours', 'theirs', 'needn', 'wasn', 'hasn', 'y', 'the', 'and', 'for', 'm', 'you', 'other', 'do', 'don', 'is', "you'd", 'me', "shouldn't", 'on', "haven't", 'be', "aren't", 'does', 'a', 'if', 'this', 'at', "weren't", 'during', 'there', 'of', 'each', 've', 'our', 'most', 'very', 'now', "mightn't", 'while', 'had', 'yourself', 'hadn', "wasn't", 'doesn', 'because', 'into', 'down', 're', 'against', 'just', 'he', 'her', 'herself', 'was', "wouldn't", 'whom', 'up', 'with', 'between', "you'll", 'through', 'no', 'd', 'which', 'in', 'but', "it's", 'its', 'from', 'that'}

	sentiment	...	clean_tweet
0	0	...	Awww thats bummer You shoulda got David Carr T...
1	0	...	upset cant update Facebook texting might cry r...
2	0	...	dived many time ball Managed save The rest bound

3 0 ... whole body feel itchy like fire

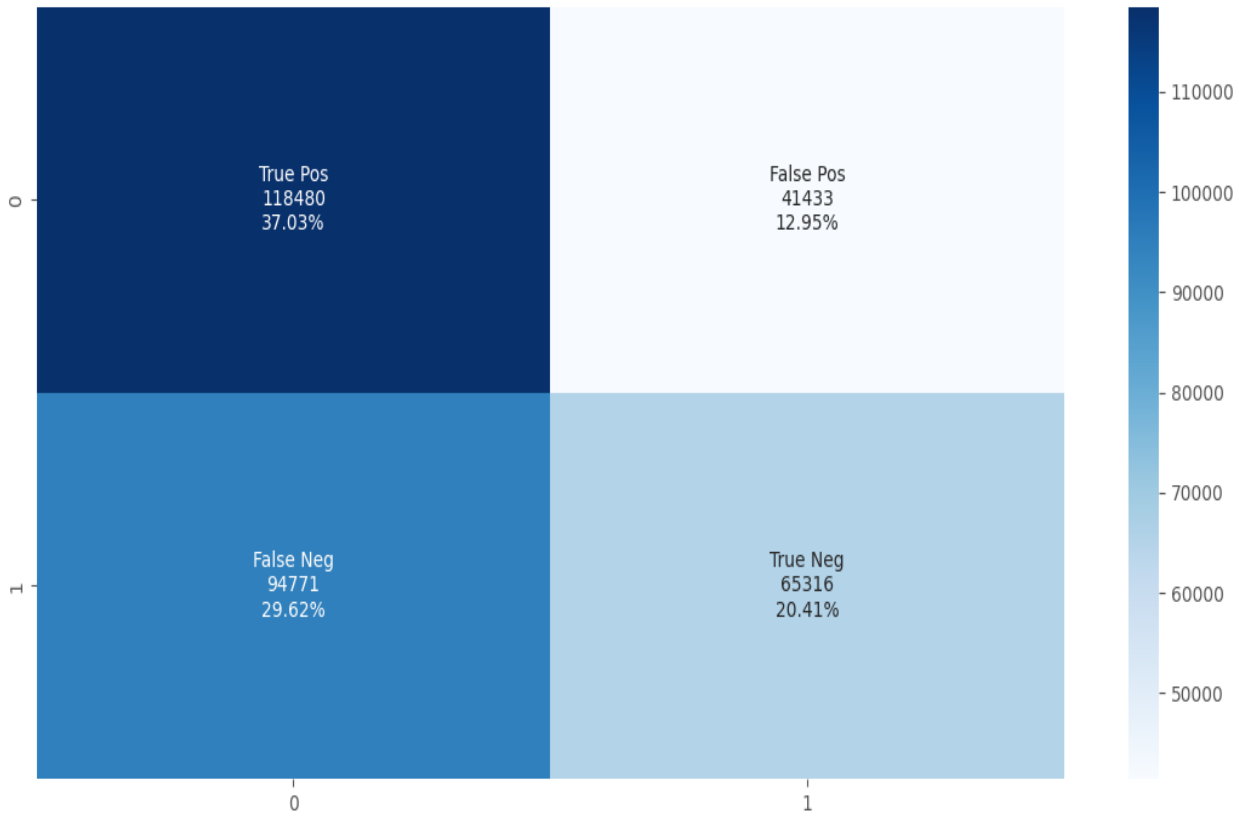
4 0 ... behaving mad cant see

[5 rows x 8 columns]

(1600000, 159)

(1600000, 159)

(1600000,)



	precision	recall	f1-score	support
0	0.56	0.74	0.64	159913
4	0.61	0.41	0.49	160087
accuracy			0.57	320000
macro avg	0.58	0.57	0.56	320000
weighted avg	0.58	0.57	0.56	320000

Cross Validation score = [0.57264687 0.57219062 0.57585937 0.57673125 0.57521563]

Train accuracy = 57.55%

Test accuracy = 57.45%

Precision score = 58.38%

Recall score = 57.45%

F1 score = 56.24%

Process finished with exit code 0

```
              precision    recall  f1-score   support

           0       0.56      0.74      0.64    159913
           4       0.61      0.41      0.49    160087

    accuracy                           0.57    320000
   macro avg       0.58      0.57      0.56    320000
weighted avg       0.58      0.57      0.56    320000

Cross Validation score =  [0.57264687 0.57219062 0.57585937 0.57673125 0.57521563]
```

```
Cross Validation score =  [0.57264687 0.57219062 0.57585937 0.57673125 0.57521563]
Train accuracy = 57.55%
Test accuracy = 57.45%
Precision score = 58.38%
Recall score = 57.45%
F1 score = 56.24%

Process finished with exit code 0
```