Naïve bayes algorithm

Dataset: 'tweet\_dataset.csv'

**Naïve bayes algorithm**

from google.colab import files

uploaded = files.upload()

**To see that uploaded file**

import pandas as pd

import io

sentiment\_data = pd.read\_csv(io.BytesIO(uploaded['tweet\_dataset.csv']), encoding='unicode\_escape')

print(sentiment\_data)

output:

textID text \

0 cb774db0d1 I`d have responded, if I were going

1 549e992a42 Sooo SAD I will miss you here in San Diego!!!

2 088c60f138 my boss is bullying me...

3 9642c003ef what interview! leave me alone

4 358bd9e861 Sons of \*\*\*\*, why couldn`t they put them on t...

... ... ...

27476 4eac33d1c0 wish we could come see u on Denver husband l...

27477 4f4c4fc327 I`ve wondered about rake to. The client has ...

27478 f67aae2310 Yay good for both of you. Enjoy the break - y...

27479 ed167662a5 But it was worth it \*\*\*\*.

27480 6f7127d9d7 All this flirting going on - The ATG smiles...

selected\_text sentiment \

0 I`d have responded, if I were going neutral

1 Sooo SAD negative

2 bullying me negative

3 leave me alone negative

4 Sons of \*\*\*\*, negative

... ... ...

27476 d lost negative

27477 , don`t force negative

27478 Yay good for both of you. positive

27479 But it was worth it \*\*\*\*. positive

27480 All this flirting going on - The ATG smiles. Y... neutral

Time of Tweet Age of User Country Population -2020 \

0 morning 0-20 Afghanistan 38928346

1 noon 21-30 Albania 2877797

2 night 31-45 Algeria 43851044

3 morning 46-60 Andorra 77265

4 noon 60-70 Angola 32866272

... ... ... ... ...

27476 night 31-45 Ghana 31072940

27477 morning 46-60 Greece 10423054

27478 noon 60-70 Grenada 112523

27479 night 70-100 Guatemala 17915568

27480 morning 0-20 Guinea 13132795

Land Area (Km²) Density (P/Km²)

0 652860.0 60

1 27400.0 105

2 2381740.0 18

3 470.0 164

4 1246700.0 26

... ... ...

27476 227540.0 137

27477 128900.0 81

27478 340.0 331

27479 107160.0 167

27480 246000.0 53

[27481 rows x 10 columns]

Applying various function over sentiment\_data variable

sentiment\_data.head()

sentiment\_data.info()

sentiment\_data.isna()

output:

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

RangeIndex: 27481 entries, 0 to 27480

Data columns (total 10 columns):

# Column Non-Null Count Dtype

--- ------ -------------- -----

0 textID 27481 non-null object

1 text 27480 non-null object

2 selected\_text 27481 non-null object

3 sentiment 27481 non-null object

4 Time of Tweet 27481 non-null object

5 Age of User 27481 non-null object

6 Country 27481 non-null object

7 Population -2020 27481 non-null int64

8 Land Area (Km²) 27481 non-null float64

9 Density (P/Km²) 27481 non-null int64

dtypes: float64(1), int64(2), object(7)

memory usage: 2.1+ MB

|  | **textID** | **text** | **selected\_text** | **sentiment** | **Time of Tweet** | **Age of User** | **Country** | **Population -2020** | **Land Area (Km²)** | **Density (P/Km²)** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **0** | False | False | False | False | False | False | False | False | False | False |
| **1** | False | False | False | False | False | False | False | False | False | False |
| **2** | False | False | False | False | False | False | False | False | False | False |
| **3** | False | False | False | False | False | False | False | False | False | False |
| **4** | False | False | False | False | False | False | False | False | False | False |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... |
| **27476** | False | False | False | False | False | False | False | False | False | False |
| **27477** | False | False | False | False | False | False | False | False | False | False |
| **27478** | False | False | False | False | False | False | False | False | False | False |
| **27479** | False | False | False | False | False | False | False | False | False | False |
| **27480** | False | False | False | False | False | False | False | False | False | False |

27481 rows × 10 columns

Selecting input\_data and output\_data

input\_data=sentiment\_data['selected\_text']

output\_data=sentiment\_data['sentiment']

print(input\_data.shape)

print(output\_data.shape)

output: (27481,) //shape of input data

(27481,)// shape of output data

print(input\_data)

output:

0 I`d have responded, if I were going

1 Sooo SAD

2 bullying me

3 leave me alone

4 Sons of \*\*\*\*,

...

27476 d lost

27477 , don`t force

27478 Yay good for both of you.

27479 But it was worth it \*\*\*\*.

27480 All this flirting going on - The ATG smiles. Y...

Name: selected\_text, Length: 27481, dtype: object

**// for training and spliting**

from sklearn.model\_selection import train\_test\_split // for training and spliting

input\_data\_train,input\_data\_test,output\_data\_train, output\_data\_test=train\_test\_split(input\_data,output\_data,test\_size=0.3)

print(input\_data\_train.shape)

print(output\_data\_train.shape)

print(input\_data\_test.shape)

print(output\_data\_test.shape)

**output:**

19236-input\_data\_train.shape

19236-output\_data\_train.shape

8245-input\_data\_test.shape

8245-output\_data\_test.shape

// importing naïve bayes algorithm

from sklearn.feature\_extraction.text import TfidfVectorizer

from sklearn.naive\_bayes import MultinomialNB

from sklearn.pipeline import make\_pipeline

model= make\_pipeline(TfidfVectorizer(),MultinomialNB())

**// fitting the model**

model.fit(input\_data\_train,output\_data\_train)

Predicted\_sentiment=model.predict(input\_data\_test)

from sklearn.metrics import confusion\_matrix

//checking predicted machine learning model

confusion\_matrix(output\_data\_test, Predicted\_sentiment)

output:

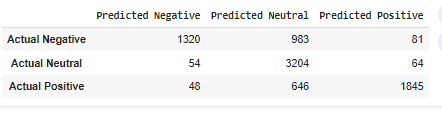
array([[1320, 983, 81],

[ 54, 3204, 64],

[ 48, 646, 1845]])

//printing the confusion matrix with label and index

pd.DataFrame(confusion\_matrix(output\_data\_test,Predicted\_sentiment),columns=['Predicted Negative', 'Predicted Neutral', 'Predicted Positive'],index=['Actual Negative','Actual Neutral','Actual Positive'])



// Printing accuracy

from sklearn.metrics import accuracy\_score

accuracy\_info=accuracy\_score(output\_data\_test,Predicted\_sentiment)

accuracy\_info

output:

0.7724681625227411

// giving our own data for checking

def  Predict\_sentiment(txt,train=input\_data\_train,model=model):

      Pred=model.predict([txt])

     return Pred

//checking

Predict\_sentiment('I love python')

//output

array(['positive'], dtype='<U8')

Predict\_sentiment('I hate python')

array(['negative'], dtype='<U8')

Predict\_sentiment('ok with python')

array(['neutral'], dtype='<U8')