In [1]:
```
#reading the data
import pandas as  pd
sentiment_data= pd.read_csv("tweet_dataset.csv" , encoding = "unicode_esc
```

In [ ]:
```

```

In [2]:
```
sentiment_data.head()
```

Out[2]:

	textID	text	selected_text	sentiment	Time of Tweet	Age of User	Country	Popula -2
0	cb774db0d1	I`d have responded, if I were going	I`d have responded, if I were going	neutral	morning	0-20	Afghanistan	38928
1	549e992a42	Sooo SAD I will miss you here in San Diego!!!	Sooo SAD	negative	noon	21-30	Albania	2877
2	088c60f138	my boss is bullying me...	bullying me	negative	night	31-45	Algeria	43851
3	9642c003ef	what interview! leave me alone	leave me alone	negative	morning	46-60	Andorra	77
4	358bd9e861	Sons of ****, why couldn`t they put them on t...	Sons of ****,	negative	noon	60-70	Angola	32866

In [3]:
```
sentiment_data.info()
```

```
<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
```

check null value

In [4]:
```
sentiment_data.isna().sum()
```

Out[4]:
```
textID              0
text                1
selected_text       0
sentiment           0
Time of Tweet       0
Age of User         0
Country             0
Population -2020    0
Land Area (Km²)     0
Density (P/Km²)     0
dtype: int64
```

input and output datas

In [5]:
```
input_data =sentiment_data["selected_text"]
output_data =sentiment_data["sentiment"]
```

In [6]:
```
print (input_data)
```

```
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
```

In [7]:
```
print (output_data)
```

```
0         neutral
1        negative
2        negative
3        negative
4        negative
           ...
27476    negative
27477    negative
27478    positive
27479    positive
27480     neutral
Name: sentiment, Length: 27481, dtype: object
```

In [8]:
```
print(input_data.shape)
print (output_data.shape)
```

```
(27481,)
(27481,)
```

Model test (allocation datas test,train)

In [9]:
```
from sklearn.model_selection import train_test_split
input_data_train , input_data_test , output_data_train , output_data_test
```

In [10]:
```
print(input_data_train.shape)
print(output_data_train.shape)
print(input_data_test.shape)
print(output_data_test.shape)
```

```
(21984,)
(21984,)
(5497,)
(5497,)
```

In [ ]:

In [11]:
```
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.naive_bayes import MultinomialNB
from sklearn.pipeline import make_pipeline
```

In [12]:
```
model = make_pipeline(TfidfVectorizer(), MultinomialNB())
```

In [13]:
```
model
```

Out[13]:
```
▸ Pipeline
  ▸ TfidfVectorizer
  ▸ MultinomialNB
```

In [ ]:

Model fit (train)

In [14]:
```
model.fit(input_data_train, output_data_train)
```

Out[14]:
```
▸ Pipeline
  ▸ TfidfVectorizer
  ▸ MultinomialNB
```

In [ ]:

In [15]:
```
predicted_sentiment = model.predict(input_data_test)
```

In [ ]:

In [16]:
```
print(input_data_train.shape)
print(output_data_train.shape)
print(input_data_test.shape)
print(output_data_test.shape)
```

```
(21984,)
(21984,)
(5497,)
(5497,)
```

In [ ]:

In [ ]:

In [ ]:

compare predit value and assign value using confusion_matrix

In [17]:
```
from sklearn.metrics import confusion_matrix
```

In [18]:
```
confusion_matrix(output_data_test , predicted_sentiment)
```

Out[18]:
```
array([[ 856,  598,   43],
       [  45, 2181,   55],
       [  32,  419, 1268]])
```

In [19]:
```
pd.DataFrame(confusion_matrix(output_data_test , predicted_sentiment), co
```

Out[19]:

	predited Negative	predited Netural	predited Positive
Actual Negative	856	598	43
Actual Netural	45	2181	55
Actual Positive	32	419	1268

In [ ]:

In [ ]:

accuracy check

In [20]:
```
from sklearn.metrics import accuracy_score
accuracy_info = accuracy_score(output_data_test , predicted_sentiment)
```

In [21]:
```
print(accuracy_info)
```

```
0.7831544478806621
```

In [ ]:

test our own way

In [22]:
```
def predict_sentiment (txt,train =input_data_train , model =model):
    pred = model.predict([txt])
    return pred
```

In [23]:
```
predict_sentiment("I love python")
```

Out[23]:
```
array(['positive'], dtype='<U8')
```

In [24]:
```
predict_sentiment("I hate python")
```
Out[24]:
```
array(['negative'], dtype='<U8')
```

In [25]:
```
predict_sentiment("I love you")
```
Out[25]:
```
array(['positive'], dtype='<U8')
```

In [26]:
```
predict_sentiment("I am ok")
```
Out[26]:
```
array(['neutral'], dtype='<U8')
```

In [27]:
```
predict_sentiment("what is your name")
```
Out[27]:
```
array(['neutral'], dtype='<U8')
```

In [28]:
```
predict_sentiment("Life is beautiful")
```
Out[28]:
```
array(['positive'], dtype='<U8')
```

In [29]:
```
predict_sentiment("beautiful is life")
```
Out[29]:
```
array(['positive'], dtype='<U8')
```

In [30]:
```
predict_sentiment("i python love")
```
Out[30]:
```
array(['positive'], dtype='<U8')
```

In [ ]:

In [ ]:

In [ ]:

In [33]:
```
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.naive_bayes import MultinomialNB
from sklearn.pipeline import make_pipeline

import joblib

model = make_pipeline(TfidfVectorizer(), MultinomialNB())

model.fit(input_data,output_data)

joblib.dump(model, "predict_sentiment_Identifier")
```
Out[33]:
```
['predict_sentiment_Identifier']
```

In [ ]: