ASSIGNMENT 09

NAME: BHOSALE SRUSHTI AMAR

CLASS: TY AIDS-A

ROLL NO.: EN23107008

In [51]:

```
import pandas as pd
import re
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.linear_model import LogisticRegression
from sklearn.svm import LinearSVC
from sklearn.metrics import classification_report, accuracy_score, confusion_m
```

In [38]:

```
df = pd.read_csv("train_data.csv")
```

In [39]:

```
df
```

Out[39]:

	sentence	sentiment
0	awww that s a bummer you shoulda got david car...	0
1	is upset that he can t update his facebook by ...	0
2	i dived many times for the ball managed to sav...	0
3	my whole body feels itchy and like its on fire	0
4	no it s not behaving at all i m mad why am i h...	0
...	...	...
1523970	just woke up having no school is the best feel...	1
1523971	thewdb com very cool to hear old walt interviews	1
1523972	are you ready for your mojo makeover ask me fo...	1
1523973	happy th birthday to my boo of alll time tupac...	1
1523974	happy charitytuesday	1

1523975 rows × 2 columns

In [40]:

```
df.shape
```

Out[40]:

```
(1523975, 2)
```

In [41]:

```
df.isnull().sum()
```

Out[41]:

```
sentence     0
sentiment    0
dtype: int64
```

In [42]:

```
df1 = df.rename(columns={"sentence": "text", "sentiment": "label"})
print(df1.head())
```

```
                                                text  label
0  awww that s a bummer you shoulda got david car...      0
1  is upset that he can t update his facebook by ...      0
2  i dived many times for the ball managed to sav...      0
3     my whole body feels itchy and like its on fire      0
4  no it s not behaving at all i m mad why am i h...      0
```

In [43]:

```
def clean_text(text):
    text = str(text).lower()
    text = re.sub(r"http\S+", "", text)
    text = re.sub(r"@\w+", "", text)
    text = re.sub(r"[^a-z\s]", "", text)
    return text
df1["text"] = df1["text"].apply(clean_text)
```

TRAIN TEST SPLIT

In [44]:

```
X_train, X_test, y_train, y_test = train_test_split(df1["text"],df1["label"],t
```

TF-IDF vectorization

In [45]:

```
vectorizer = TfidfVectorizer(max_features=30000, ngram_range=(1,2))
```

In [46]:

```
X_train_vec = vectorizer.fit_transform(X_train)
X_test_vec = vectorizer.transform(X_test)
```

Logistic Regression

In [47]:

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

In [48]:

```
lr_model.fit(X_train_vec, y_train)
lr_preds = lr_model.predict(X_test_vec)
```

In [49]:

```
def evaluate_model(y_true, y_pred, model_name):
    print(f"===== {model_name} =====")
    print("Accuracy:", accuracy_score(y_true, y_pred))
    print("Classification Report:\n", classification_report(y_true, y_pred))
    print("Confusion Matrix:\n", confusion_matrix(y_true, y_pred))
    print("\n")
```

In [50]:

```
lr_acc= accuracy_score(y_test, lr_preds)
print( "Accuracy",lr_acc)
lr_cr= classification_report(y_test, lr_preds)
print( "Classification Report:\n",lr_cr)
lr_cm= confusion_matrix(y_test, lr_preds)
print( "Confusion Matrix:\n",lr_cm)
```

```
Accuracy 0.8114831280040683
Classification Report:
               precision    recall  f1-score   support

           0       0.82      0.80      0.81    153092
           1       0.80      0.82      0.81    151703

    accuracy                           0.81    304795
   macro avg       0.81      0.81      0.81    304795
weighted avg       0.81      0.81      0.81    304795

Confusion Matrix:
 [[122841  30251]
 [ 27208 124495]]
```

Gradient Boosting model

In [16]:

```
svm = LinearSVC()
```

In [54]:

```
svm.fit(X_train_vec, y_train)
svm_preds = svm.predict(X_test_vec)
```

In [55]:

```
svm_acc= accuracy_score(y_test, svm_preds)
print( "Accuracy",gb_acc)
svm_cr= classification_report(y_test, svm_preds)
print( "Classification Report:\n",svm_cr)
svm_cm= confusion_matrix(y_test, svm_preds)
print( "Confusion Matrix:\n",svm_cm)
```

```
Accuracy 0.7023606030282649
Classification Report:
               precision    recall  f1-score   support

           0       0.82      0.80      0.81    153092
           1       0.80      0.82      0.81    151703

    accuracy                           0.81    304795
   macro avg       0.81      0.81      0.81    304795
weighted avg       0.81      0.81      0.81    304795

Confusion Matrix:
 [[122193  30899]
 [ 26961 124742]]
```

COMPARISON

In [56]:

```
if lr_acc > svm_acc:
    print("Logistic Regression performs better")
elif gb_acc > lr_acc:
    print("Gradient Boosting performs better")
else:
    print("Both models perform equally")
```

```
Logistic Regression performs better
```

```
In [ ]:
```

```
In [ ]:
```

```
In [ ]:
```