3107 – JAWAHAR ENGINEERING COLLEGE

Subject Title - AI 101- Artificial Intelligence
Project Title – Building a Smarter AI Powered Classifier

Team Head – 310721106001: Junia Susheela Shalom

S.No.	NaanMudhalvan ID	Position	Name	Department
		Faculty Mentor	V. Nivaskumar	
1.	au310721106001	Team Head	Junia Susheela Shalom	ECE
2.	au310721205001	Team member	M. Hariragavan	IT
3.	au310721205002	Team Member	S. Muthunivas Pandi	IT

STEP 1: Uploading the CSV file into a Jupyter notebook

>First we import the required libraries.
>Secondly we open the csv file using the code: dataset = pd.read_csv('spam.csv')

In [1]:
```
import numpy as np
import pandas as pd
```

In [2]:
```
dataset = pd.read_csv('spam.csv')
```

In [3]:
```
dataset.sample(5)
```

Out[3]:

	v1	v2	Unnamed: 2	Unnamed: 3	Unnamed: 4
1246	ham	I do know what u mean, is the king of not hav...	NaN	NaN	NaN
2339	ham	Cheers for the message Zogtorius. IåÕve been s...	NaN	NaN	NaN
3340	ham	Still i have not checked it da. . .	NaN	NaN	NaN
1351	ham	Yo theres no class tmrw right?	NaN	NaN	NaN
2997	ham	No b4 Thursday	NaN	NaN	NaN

STEP 2: Cleaning the data and preprocessing

>First we converting text to lowercase.
> Secondly, tokenize the text to individual words.
> Then, we remove stop words and punctuation.
>Last but not least, implement Lemmatization (that that involves grouping together different inflected forms of the same word).
> The above processes are performed using the code shown below:

In [54]:
```
from nltk.stem.porter import PorterStemmer
ps = PorterStemmer()
ps.stem('dancing')
```

Out[54]:
```
'danc'
```

In [56]:
```
def transform_text(text):
    text = text.lower()
    text = nltk.word_tokenize(text)

    y=[]
    for i in text:
        if i.isalnum():
            y.append(i)
    text = y[:]
    y.clear()

    for i in text:
        if i not in stopwords.words('english') and i not in string.punctuation:
            y.append(i)

    text = y[:]
    y.clear()

    for i in text:
        y.append(ps.stem(i))

    return " ".join(y)
```

In [57]:
```
transform_text('I love the lectures on machine learning')
```

Out[57]:
```
'love lectur machin learn'
```

STEP 3: Feature Extraction

> This step involves converting tokenized words to numerical features.
> Here we use the TF-IDF technique to implement the following lines of code:

```
In [18]: print(Y.shape)
         print(Y_train.shape)
         print(Y_test.shape)
```

```
(5572,)
(4457,)
(1115,)
```

```
In [21]: feature_extraction = TfidfVectorizer(min_df = 1, stop_words = 'english', lowercase='True')

         X_train_features = feature_extraction.fit_transform(X_train)
         X_test_features = feature_extraction.transform(X_test)

         Y_train = Y_train.astype('int')
         Y_test = Y_test.astype('int')
```

```
In [22]: print(X_train)
```

```
3075                  Don know. I did't msg him recently.
1787    Do you know why god created gap between your f...
1614                          Thnx dude. u guys out 2nite?
4304                                    Yup i'm free...
3266    44 7732584351, Do you want a New Nokia 3510i c...
                              ...
789     5 Free Top Polyphonic Tones call 087018728737,...
968     What do u want when i come back?.a beautiful n...
1667    Guess who spent all last night phasing in and ...
3321    Eh sorry leh... I din c ur msg. Not sad alread...
1688    Free Top ringtone -sub to weekly ringtone-get ...
Name: Message, Length: 4457, dtype: object
```

STEP 4: Model Selection

> For this project we implement the Naïve Bayes algorithm.
> Naïve Bayes algorithm is a supervised learning algorithm, which is based on Bayes Theorem and used for solving classification problems.
> The model can trained as follows:

```
In [336]: from sklearn.model_selection import train_test_split

In [337]: X_train,X_test,y_train,y_test = train_test_split(X,y,test_size=0.2,random_state=2)

In [338]: from sklearn.naive_bayes import GaussianNB,MultinomialNB,BernoulliNB
          from sklearn.metrics import accuracy_score,confusion_matrix,precision_score

In [339]: gnb = GaussianNB()
          mnb = MultinomialNB()
          bnb = BernoulliNB()

In [340]: gnb.fit(X_train,y_train)
          y_pred1 = gnb.predict(X_test)
          print(accuracy_score(y_test,y_pred1))
          print(confusion_matrix(y_test,y_pred1))
          print(precision_score(y_test,y_pred1))

          0.874274661508704
          [[791 105]
           [ 25 113]]
```

STEP 5: Evaluation and Iterative Improvement of Model

> First, we measure the model's performance using metrics like accuracy, precision, recall, and F1-score, Area under Curve, Confusion Matrix and Mean Square Error.
> Secondly, the model is fine-tuned with hyperparameters to improve its accuracy.
> The above can be implemented by the following lines of code:

In [50]:
```
from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score

print("Accuracy score: {}". format(accuracy_score(y_test, prediction)) )
print("Precision score: {}". format(precision_score(y_test, prediction)) )
print("Recall score: {}". format(recall_score(y_test, prediction)))
print("F1 score: {}". format(f1_score(y_test, prediction)))
```

```
Accuracy score: 0.97847533632287
Precision score: 0.891156462585034
Recall score: 0.9424460431654677
F1 score: 0.9160839160839161
```

In [54]:
```
from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score

print("Accuracy score: {}". format(accuracy_score(y_test, prediction)) )
print("Precision score: {}". format(precision_score(y_test, prediction)) )
print("Recall score: {}". format(recall_score(y_test, prediction)))
print("F1 score: {}". format(f1_score(y_test, prediction)))
```

```
Accuracy score: 0.9865470852017937
Precision score: 0.984375
Recall score: 0.9064748201438849
F1 score: 0.9438202247191011
```