> **NAME: DHANUSHREE K G**

> **SPAM DETECTION**

```
import pandas as pd
import numpy as np
```

LOADING THE DATASET

```
data = pd.read_csv
("https://drive.google.com/file/d/1BnULfJNLiw99SzfXUzdlHUaBYTC9qZ0h/view?usp=sharing",
 encoding= 'unicode_escape')
data.head()
```

	type	text
0	ham	Go until jurong point, crazy.. Available only ...
1	ham	Ok lar... Joking wif u oni...
2	spam	Free entry in 2 a wkly comp to win FA Cup fina...
3	ham	U dun say so early hor... U c already then say...
4	ham	Nah I don't think he goes to usf, he lives aro...

PREPROCESSING OF DATASET

ENCODING

```
from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()
data['spam']= le.fit_transform(data['type'])
data.head()
```

	type	text	spam
0	ham	Go until jurong point, crazy.. Available only ...	0
1	ham	Ok lar... Joking wif u oni...	0
2	spam	Free entry in 2 a wkly comp to win FA Cup fina...	1
3	ham	U dun say so early hor... U c already then say...	0
4	ham	Nah I don't think he goes to usf, he lives aro...	0

```
data.shape
```

```
(116, 3)
```

```
print("Columns in the dataframe:")
data.columns
```

```
Columns in the dataframe:
Index(['type', 'text', 'spam'], dtype='object')
```

```
print("Columns info:")
data.info();
```

```
Columns info:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 116 entries, 0 to 115
Data columns (total 3 columns):
 #   Column  Non-Null Count  Dtype
```

```
---  ------  --------------  -----
 0   type    116 non-null    object
 1   text    116 non-null    object
 2   spam    116 non-null    int64
dtypes: int64(1), object(2)
memory usage: 2.8+ KB
```

TOKENIZATION

```
def tokenizer(txt):
  return txt.split()
```

```
data['text'] = data['text'].apply(tokenizer)
```

```
data.head()
```

	type	text	spam
0	ham	[Go, until, jurong, point,, crazy.., Available...	0
1	ham	[Ok, lar..., Joking, wif, u, oni...]	0
2	spam	[Free, entry, in, 2, a, wkly, comp, to, win, F...	1
3	ham	[U, dun, say, so, early, hor..., U, c, already...	0
4	ham	[Nah, I, don't, think, he, goes, to, usf,, he,...	0

STEMMING

```
from nltk.stem.snowball import SnowballStemmer
```

```
porter = SnowballStemmer("english",ignore_stopwords=False)

def stem_it(txt):
  return [porter.stem(i) for i in txt]


data['text'] = data['text'].apply(stem_it)


data.head()
```

	type	text	spam
0	ham	[go, until, jurong, point,, crazy.., avail, on...	0
1	ham	[ok, lar..., joke, wif, u, oni...]	0
2	spam	[free, entri, in, 2, a, wkli, comp, to, win, f...	1
3	ham	[u, dun, say, so, earli, hor..., u, c, alreadi...	0
4	ham	[nah, i, don't, think, he, goe, to, usf,, he, ...	0

LEMMETIZATION

```
import nltk
nltk.download('wordnet')
```

```
[nltk_data] Downloading package wordnet to /root/nltk_data...
[nltk_data]   Unzipping corpora/wordnet.zip.
True
```

```
from nltk.stem import WordNetLemmatizer
lemmatizer = WordNetLemmatizer()
```

```
def lemmit_it(txt):
  return [lemmatizer.lemmatize(i,pos = "a") for i in txt]


data['text'] = data['text'].apply(lemmit_it)


data.head()
```

	type	text	spam
0	ham	[go, until, jurong, point,, crazy.., avail, on...	0
1	ham	[ok, lar..., joke, wif, u, oni...]	0
2	spam	[free, entri, in, 2, a, wkli, comp, to, win, f...	1
3	ham	[u, dun, say, so, earli, hor..., u, c, alreadi...	0
4	ham	[nah, i, don't, think, he, goe, to, usf,, he, ...	0

STOPWORD REMOVAL

```
import nltk
nltk.download('stopwords')
```

```
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Unzipping corpora/stopwords.zip.
True
```

```
from nltk.corpus import stopwords
stop_words = stopwords.words('english')


def stop_it(txt):
```

```
    r = [i for i in txt if not i in stop_words]
    return r

data['text'] = data['text'].apply(stop_it)


data.head()
```

	type	text	spam
0	ham	[go, jurong, point,, crazy.., avail, onli, bug...	0
1	ham	[ok, lar..., joke, wif, u, oni...]	0
2	spam	[free, entri, 2, wkli, comp, win, fa, cup, fin...	1
3	ham	[u, dun, say, earli, hor..., u, c, alreadi, sa...	0
4	ham	[nah, think, goe, usf,, live, around, though]	0

Transform text data into TDF/ TF-IDF vectors

```
from sklearn.feature_extraction.text import TfidfVectorizer
tfidf = TfidfVectorizer()
y = data['spam'].values
x = tfidf.fit_transform(''.join(i) for i in data['text'])
```

TRAIN TEST SPLIT

```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test = train_test_split(x,y,random_state=1,
                                                 test_size=0.2,shuffle=False)
```

MODEL

```
from sklearn.metrics import accuracy_score
```

Logistic Regression

```
from sklearn.linear_model import LogisticRegression
m1 = LogisticRegression()
m1.fit(x_train,y_train)
y_pred = m1.predict(x_test)
acc1 = accuracy_score(y_pred,y_test)*100
print("Accuracy of Logistic Regression model: ",acc1)
```

```
Accuracy of Logistic Regression model:  87.5
```

LinearSVC

```
from sklearn.svm import LinearSVC
m2 = LinearSVC()
m2.fit(x_train,y_train)
y_pred = m2.predict(x_test)
acc2 = accuracy_score(y_pred,y_test)*100
print("Accuracy of LinearSVC model: ",acc2)
```

```
Accuracy of LinearSVC model:  87.5
```