```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import nltk
from nltk.corpus import stopwords
from nltk.stem.porter import PorterStemmer
import csv

from google.colab import files
uploaded = files.upload()
```

Choose Files No file chosen Upload widget is only available when the cell has been executed in the current browser session. Please rerun this cell to enable.

```
Saving spam[1].csv to spam[1].csv
```

```
from google.colab import drive
drive.mount('/content/drive')
```

```
Mounted at /content/drive
```

```
df=pd.read_csv("spam[1].csv", encoding="latin")
df.head()
```

	v1	v2	Unnamed: 2	Unnamed: 3	Unnamed: 4
0	ham	Go until jurong point, crazy.. Available only ...	NaN	NaN	NaN
1	ham	Ok lar... Joking wif u oni...	NaN	NaN	NaN
2	spam	Free entry in 2 a wkly comp to win FA Cup fina...	NaN	NaN	NaN
3	ham	U dun say so early hor... U c already then say...	NaN	NaN	NaN
4	ham	Nah I don't think he goes to usf, he lives aro...	NaN	NaN	NaN

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5572 entries, 0 to 5571
Data columns (total 5 columns):
 #   Column      Non-Null Count  Dtype 
---  ------      --------------  ----- 
 0   v1          5572 non-null   object
 1   v2          5572 non-null   object
 2   Unnamed: 2  50 non-null     object
 3   Unnamed: 3  12 non-null     object
 4   Unnamed: 4  6 non-null      object
dtypes: object(5)
memory usage: 217.8+ KB
```

```
df.isna().sum()
```

```
v1               0
v2               0
Unnamed: 2    5522
Unnamed: 3    5560
Unnamed: 4    5566
dtype: int64
```

```
df.rename({"v1":"label","v2":"text"},inplace=True,axis=1)
```

```
df.tail()
```

	label	text	Unnamed: 2	Unnamed: 3	Unnamed: 4
5567	spam	This is the 2nd time we have tried 2 contact u...	NaN	NaN	NaN
5568	ham	Will Ì_ b going to esplanade fr home?	NaN	NaN	NaN
5569	ham	Pity, * was in mood for that. So...any other s...	NaN	NaN	NaN
5570	ham	The guy did some bitching but I acted like i'd...	NaN	NaN	NaN
5571	ham	Rofl. Its true to its name	NaN	NaN	NaN

```
from sklearn.datasets import load_iris
iris=load_iris()
x=iris.data
y=iris.target
```

```
from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()
df['label']=le.fit_transform(df['label'])
```

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

```
from imblearn.over_sampling import SMOTE
sm=SMOTE(random_state=2)
x_train_res,y_train_res=sm.fit_resample(x_train,y_train)
```

```
nltk.download("stopwords")
```

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

```
import nltk
from nltk.corpus import stopwords
from nltk.stem import PorterStemmer
```

```
import re
corpus=[]
length=len(df)
```

```
for i in range(0,length):
  text=re.sub("[^a-zA-z0-9]"," ",df["text"][i])
  text=text.lower()
  text=text.split()
  pe=PorterStemmer()
  stopword=stopwords.words("english")
  text=[pe.stem(word) for word in text if not word in set(stopword)]
  text=" ".join(text)
  corpus.append(text)
  corpus
```

```
from sklearn.feature_extraction.text import CountVectorizer
cv=CountVectorizer(max_features=35000)
x=cv.fit_transform(corpus).toarray()
```

```
import pickle
pickle.dump(cv,open('cv1.pkl','wb'))
```

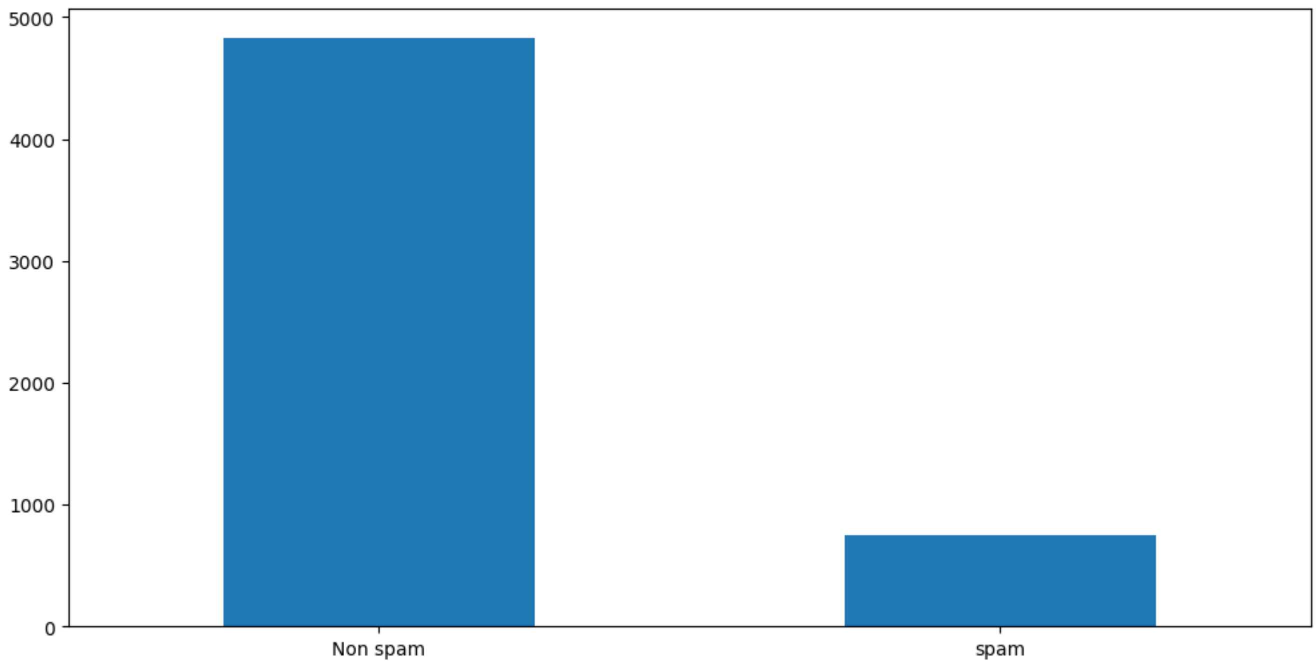
```
df.describe()
```

	label
count	5572.000000
mean	0.134063
std	0.340751
min	0.000000
25%	0.000000
50%	0.000000
75%	0.000000
max	1.000000

```
df.shape
```

```
(5572, 5)
```

```
df["label"].value_counts().plot(kind="bar",figsize=(12,6))
plt.xticks(np.arange(2),('Non spam','spam'),rotation=0);
```



```
from sklearn.tree import DecisionTreeClassifier
model=DecisionTreeClassifier()
model.fit(x_train_res,y_train_res)
```

```
DecisionTreeClassifier
```

```
from sklearn.ensemble import RandomForestClassifier
model1=RandomForestClassifier()
model1.fit(x_train_res,y_train_res)
RandomForestClassifier
RandomForestClassifier()
```

```
from sklearn.naive_bayes import MultinomialNB
model=MultinomialNB()
model.fit(x_train_res,y_train_res)
MultinomialNB
MultinomialNB()
```

```
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense
model=Sequential()
x_train.shape
```

```
model.add(Dense(units=x_train_res.shape[1],activation="relu",kernel_initializer="random_uniform"))
model.add(Dense(units=100,activation="relu",kernel_initializer="random_uniform"))
model.add(Dense(units=100,activation="relu",kernel_initializer="random_uniform"))
model.add(Dense(units=1,activation="sigmoid"))
model.compile(optimizer="adam",loss="binary_crossentropy",metrics=['accuracy'])
generator=model.fit(x_train_res,y_train_res,epochs=10,steps_per_epoch+len(x_train_res)//64)
```

```
y_pred=model.predict(x_test)
y_pred
```

```
y_pr=np.where(y_pred>0.5,1,0)
y_test
```

```
from sklearn.metrics import confusion_matrix,accuracy_score
cm=confusion_matrix(y_test,y_pr)
score=accuracy_score(y_test,y_pr)
```

```
print(cm)
print('Accuracy Score Is:-',score*100)
```

```
def new_review(new_review):
  new_review=new_review
  new_review=re.sub('[^a-zA-Z]','',new_review)
  new_review=new_review.lower()
  new_review=new_review.split()
  ps=PorterStemmer()
  all_stopwords=stopwords.words('english')
  all_stopwords.remove('not')
  new_review=[ps.stem(word) for word in new_review if not word in set(all_stopwords)]
  new_review=''.join(new_review)
  new_corpus=[new_review]
  new_x_test=cv.transform(new_corpus).toarray()
  print(new_x_test)
  new_y_pred=loaded_model.predict(new_x_test)
  print(new_y_pred)
  new_x_pred=np.where(new_y_pred>0.5,1,0)
  return new_y_pred
new_review=new_review(str(input("Enter new review...")))
```

```
from sklearn.metrics import confusion_matrix,accuracy_score,classification_report
cm=confusion_matrix(y_test,y_pred)
score=accuracy_score(y_test,y_pred)
print(cm)
print('Accuracy Score Is Naive Bayes|:-',score*100)
```

```
cm=confusion_matrix(y_test,y_pred)
score=accuracy_score(y_test,y_pred)
print(cm)
print('Accuracy Score Is:-',score*100)
cm1=confusion_matrix(y_test,y_pred1)
score1=accuracy_score(y_test,y_pred1)
print(cm1)
print('Accuracy Score Is:-',score1*100)
```

```
from sklearn.metrics import confusion_matrix,accuracy_score
cm=confusion_matrix(y_test,y_pr)
score=accuracy_score(y_test,y_pr)
print(cm)
print('Accuracy Score Is:-',score*100)
```

```
from sklearn.metrics import confusion_matrix,accuracy_score
cm=confusion_matrix(y_test,y_pr)
score=accuracy_score(y_test,y_pr)
print(cm)
print('Accuracy Score Is:-',score*100)
```

```
model.save('spam.h5')
```

```
from flask import Flask,render_template,request
import pickle
import numpy as np
import re
import nltk
from nltk.corpus import stopwords
from nltk.stem import PorterStemmer
from tensorflow.keras.models import load_model
```

```
loaded_model=load_model('spam.h5')
cv=pickle.load(open('cv1.pkl','rb'))
app=Flask(_name_)
```

```
def home():
  return render_template('home.html')
```

```
def prediction():
  return render_template('spam.html')
def predict():
  if request.method=='POST':
    message=request.form['message']
    data=message
  new_review=str(data)
  print(new_review)
  new_review=re.sub('[^a-zA-Z]','',new_review)
  new_review=new_review.lower()
  new_review+new_review.split()
  ps=PorterStemmer()
  all_stopwords=stopwords.words('english')
  all_stopwords=stopwords.remove('not')
  new_review=[ps.stem(word)for word in new_review if not word in set(all_stopwords)]
  new_review=''.join(new_review)
  new_corpus=[new_review]
  new_x_test=cv.transform(new_corpus).toarray()
  print(new_x_test)
  new_y_pred=loaded_model.predict(new_x_test)
  new_x_pred=np.where(new_y_pred>0.5,1,0)
  print(new_x_pred)
  if new_review[0][0]==1:
    return render_template('result.html',prediction="Spam")
  else:
    return render_template('result.html',prediction="Not a Spam")



if _name_=="_main_":
  port=int(os.environ.get('PORT',5000))
  app.run(debug=False)
```