PRACTICAL – 12

E – MAIL SPAM CLASSIFICATION

Code: - (For Deployment)

spamDetector.py

```python

import pickle
import streamlit as st
#pip install -U pypiwin32
#from win32com.client import Dispatch

#def speak(text):
#    speak=Dispatch(("SAPI.SpVoice"))
#    speak.Speak(text)

model=pickle.load(open("spam.pkl","rb"))
cv=pickle.load(open("vectorizer.pkl", "rb"))

def main():
    st.title("Email Spam Classification :")
    st.subheader("Made By Ujjwal Chauhan")
    msg=st.text_input("Enter a text: ")
    if st.button("Predict"):
            data=[msg]
            vect=cv.transform(data).toarray()
            prediction=model.predict(vect)
            result=prediction[0]
            if result==1:
                st.error("This is a SPAM mail")
#               speak("This is a SPAM mail")
            else:
                st.success("This is NOT a SPAM mail")
#               speak("This is NOT a SPAM mail")
main()
```

Source Code (ipynb)-

->Data Pre-Processing: -

```
import pandas as pd
```

```
import pandas as pd
```

```
data=pd.read_csv("spam.csv", encoding="latin-1")
```

```
data.head(5)
```

	class	message	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

```
data.columns
```

```
Index(['class', 'message', 'Unnamed: 2', 'Unnamed: 3', 'Unnamed: 4'], dtype='object')
```

```
data.drop(['Unnamed: 2', 'Unnamed: 3', 'Unnamed: 4'],axis=1, inplace=True)
```

```
data.columns
```

```
Index(['class', 'message'], dtype='object')
```

```
data.head()
```

	class	message
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...

```
data['class']=data['class'].map({'ham':0,'spam':1})
```

```
data.head()
```

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

->Model Selection: -

```
from sklearn.feature_extraction.text import CountVectorizer
```

```
cv=CountVectorizer()
```

```
x=data['message']
y=data['class']
```

```
x.shape
```

```
(5572,)
```

```
y.shape
```

```
(5572,)
```

```
print()
```

```
x=cv.fit_transform(x)
```

```
x
```

```
<5572x8672 sparse matrix of type '<class 'numpy.int64'>'
        with 73916 stored elements in Compressed Sparse Row format>
```

1.The Cat 2.The Dog 3.The Bird

The Cat Dog Bird

1. 1 1 0 0
2. 1 0 1 0
3. 1 0 0 1

```
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)
```

```
x_train.shape
```

```
(4457, 8672)
```

➔ Naïve Bayes Algorithm: -

```
from sklearn.naive_bayes import MultinomialNB
```

```
model=MultinomialNB()
```

```
model.fit(x_train,y_train)
```

```
▾ MultinomialNB
MultinomialNB()
```

➔ Accuracy: -

```
result=model.score(x_test,y_test)
```

```
result=result*100
```

```
result
```

```
98.65470852017937
```

```
import pickle
```

```
pickle.dump(model, open("spam.pkl","wb"))
```

```
pickle.dump(cv, open("vectorizer.pkl","wb"))
```

```
clf=pickle.load(open("spam.pkl","rb"))
```

```
clf
```

```
▾ MultinomialNB
MultinomialNB()
```

➔ Testing: -

```
msg="You win 100 Dollar"
data=[msg]
vect=cv.transform(data).toarray()
result=model.predict(vect)
print(result)   # 1 for Spam and 0 for NotSpam
```

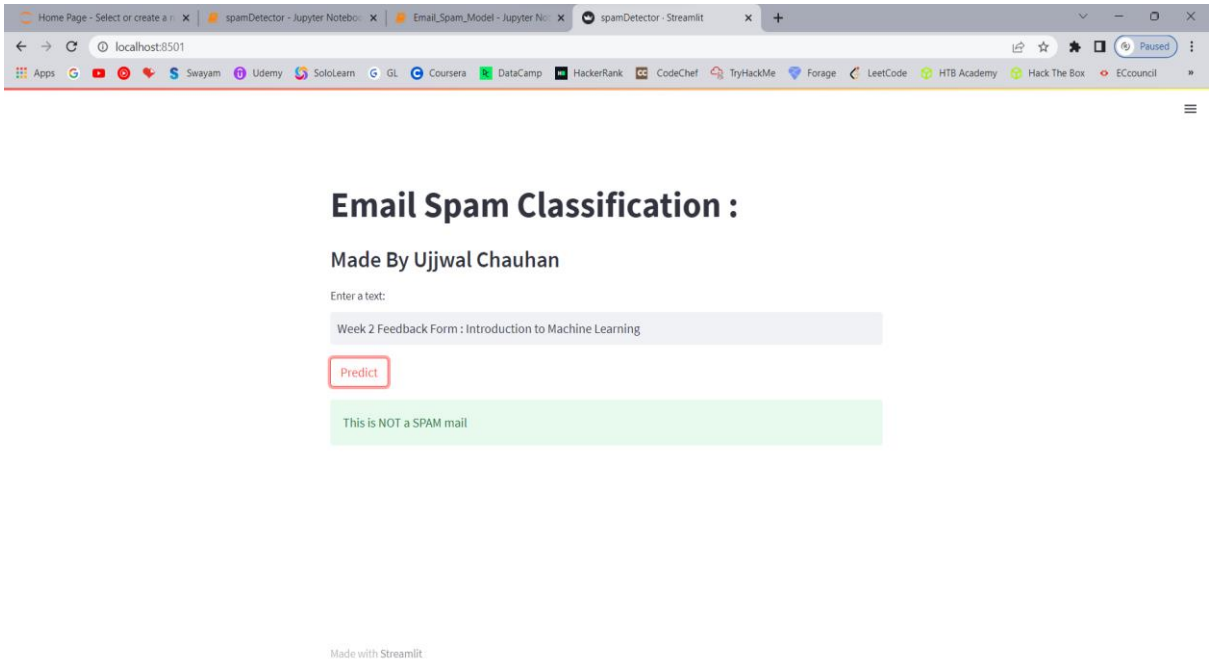
```
[1]
```

```
msg="Hi "
data=[msg]
vect=cv.transform(data).toarray()
result=model.predict(vect)
print(result)
```

```
[0]
```

➔ Result: -

NOT SPAM -



SPAM -