

**ASSIGNMENT-4**  
**PYTHON PROGRAMMING**

Team ID -

**PNT2022TMID19999**

Assignment Date	03 Nov 2022
Student Name	Joshitha BM
Student Roll No	727819TUEC048
Maximum Marks	2 Marks

**#Download the Dataset**

**#Import the library**

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from tensorflow.keras.preprocessing.sequence import pad_sequences
from sklearn.preprocessing import LabelEncoder
from tensorflow.keras.models import Model
from tensorflow.keras.layers import LSTM, Activation, Dense, Dropout, Input, Embedding
from tensorflow.keras.optimizers import RMSprop
from tensorflow.keras.preprocessing.text import Tokenizer
from tensorflow.keras.preprocessing import sequence
from tensorflow.keras.utils import to_categorical
from tensorflow.keras.callbacks import EarlyStopping
%matplotlib inline
```

**#Read Dataset and Preprocessing**

```
df = pd.read_csv('/content/spam.csv', delimiter=',', encoding='latin-1')
df.head()
```

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

	Unnamed: 3	Unnamed: 4		
0	NaN	NaN		
1	NaN	NaN		
2	NaN	NaN		
3	NaN	NaN	4	NaN NaN

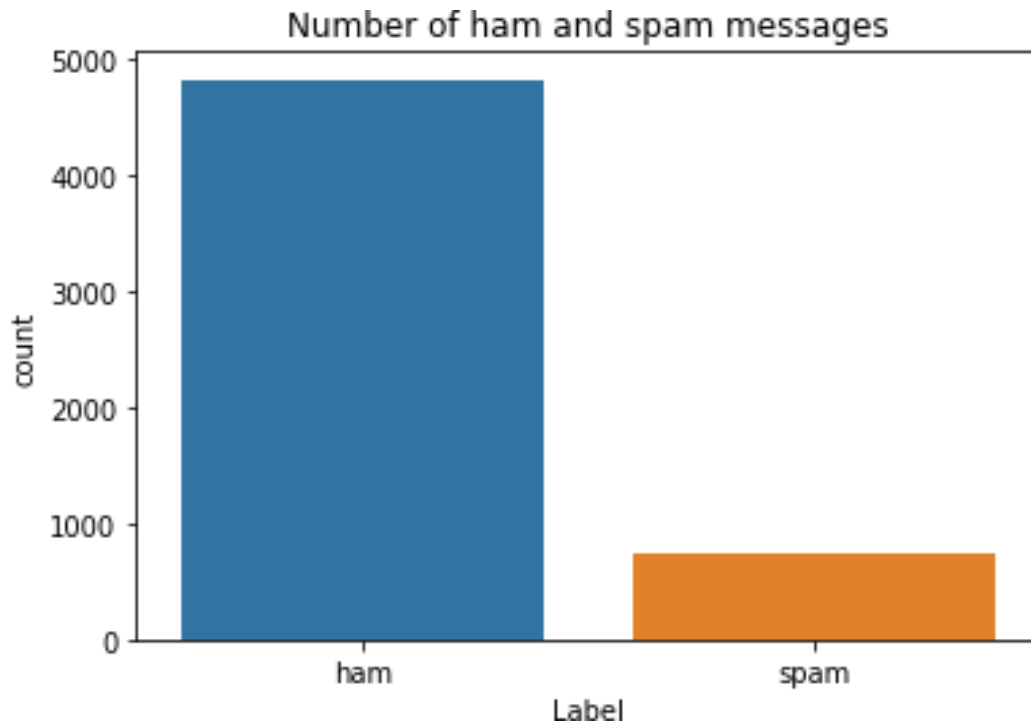
```
df.drop(['Unnamed: 2', 'Unnamed: 3', 'Unnamed: 4'],axis=1,inplace=True)
df.info()
```

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

```
sns.countplot(df.v1) plt.xlabel('Label')
plt.title('Number of ham and spam messages')
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43:
FutureWarning: Pass the following variable as a keyword arg: x. From version
0.12, the only valid positional argument will be `data`, and passing other
arguments without an explicit keyword will result in an error or
misinterpretation. FutureWarning
```

```
Text(0.5, 1.0, 'Number of ham and spam messages')
```



### #Creating input and output vectors.

```
X = df.v2 Y = df.v1 le =
LabelEncoder() Y =
le.fit_transform(Y) Y =
Y.reshape(-1,1)
```

### #Split into training and test data.

```
X_train,X_test,Y_train,Y_test = train_test_split(X,Y,test_size=0.15)
```

### #Processing the data from tensorflow.keras.preprocessing.sequence

```
import pad_sequences
```

```
max_words = 1000 max_len
= 150 tok = Tokenizer(num_words=max_words)
tok.fit_on_texts(X_train) sequences =
tok.texts_to_sequences(X_train) sequences_matrix =
sequence.pad_sequences(sequences,maxlen=max_len)
```

### #Create the model, Add Layers (LSTM, Dense)

```
def RNN(): inputs = Input(name='inputs',shape=[max_len])
layer = Embedding(max_words,50,input_length=max_len)(inputs) layer
= LSTM(64)(layer) layer = Dense(256,name='FC1')(layer) layer
= Activation('relu')(layer) layer =
```

```
Dropout(0.5)(layer)    layer = Dense(1,name='out_layer')(layer)    layer
= Activation('sigmoid')(layer)    model =
Model(inputs=inputs,outputs=layer)    return model #Compile the
model model = RNN() model.summary()
model.compile(loss='binary_crossentropy',optimizer=RMSprop(),metrics=['accuracy'])
Model: "model"
```

Layer (type)	Output Shape	Param #
=====		
inputs (InputLayer)	[(None, 150)]	0
embedding (Embedding)	(None, 150, 50)	50000
lstm (LSTM)	(None, 64)	29440
FC1 (Dense)	(None, 256)	16640
activation (Activation)	(None, 256)	0
dropout (Dropout)	(None, 256)	0
out_layer (Dense)	(None, 1)	257
activation_1 (Activation)	(None, 1)	0
=====		
Total params: 96,337		
Trainable params: 96,337		
Non-trainable params: 0		

## **#Fit The Model**

```
model.fit(sequences_matrix,Y_train,batch_size=128,epochs=10,
validation_split=0.2,callbacks=[EarlyStopping(monitor='val_loss',min_delta=0.
0001)])
Epoch 1/10
30/30 [=====] - 12s 295ms/step - loss: 0.3310 -
accuracy: 0.8759 - val_loss: 0.1603 - val_accuracy: 0.9346 Epoch 2/10
30/30 [=====] - 8s 273ms/step - loss: 0.0854 -
accuracy: 0.9778 - val_loss: 0.0571 - val_accuracy: 0.9831
<keras.callbacks.History at 0x7fbf58e3fbd0>
```

## **#Process the data**

```
test_sequences = tok.texts_to_sequences(X_test) test_sequences_matrix =
sequence.pad_sequences(test_sequences,maxlen=max_len)
```

**#Save the model** model.save('spam.h5') **#Testing the**

**model**

accr

```
= model.evaluate(test_sequences_matrix,Y_test)
```

```
27/27 [=====] - 1s 34ms/step - loss: 0.0363 -  
accuracy: 0.9904
```

```
print('Test set\n Loss: {:.3f}\n Accuracy:  
{:.3f}'.format(accr[0],accr[1]))
```

Test set

Loss:

0.036

Accuracy: 0.990