

**Assignment -4**  
Python Programming

Assignment Date	28 October 2022
Student Name	Mukil varathan.M
Student Roll Number	721419104027
Maximum Marks	2 Marks

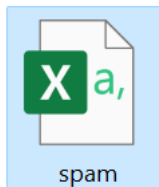
**Question-1:**

Download the dataset

**Output:**

Download the dataset from

<https://www.kaggle.com/code/kredy10/simple-lstm-for-text-classification/data>



**Question-2:**

Import required library

**Output:**

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

### Question 3:

Read dataset and do pre-processing

Output:

```
[2] df = pd.read_csv('spam.csv',delimiter=',',encoding='latin-1')
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

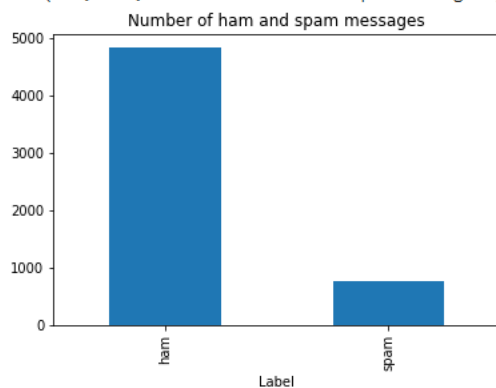
```
df.drop(['Unnamed: 2', 'Unnamed: 3', 'Unnamed: 4'],axis=1,inplace=True)
```

```
df.shape
```

```
(5572, 2)
```

```
df['v1'].value_counts().plot(kind='bar')
plt.xlabel('Label')
plt.title('Number of ham and spam messages')
```

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



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

```
X_train,X_test,Y_train,Y_test = train_test_split(X,Y,test_size=0.20)
```

```
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 = keras.utils.pad_sequences(sequences,maxlen=max_len)
```

#### Question 4:

Create Model

Output:

```
model = Model(inputs=inputs,outputs=layer)
```

#### Question 5:

Add Layers (LSTM, Dense-(Hidden Layers), Output)

Output:

```
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)
```

#### Question 6:

Compile the Model

Output:

```
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
```

### Question 7:

Fit the Model

#### Output:

```
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  
28/28 [=====] - 11s 287ms/step - loss: 0.3337 - accuracy: 0.8654 - val_loss: 0.2007 - val_accuracy: 0.9742  
Epoch 2/10  
28/28 [=====] - 11s 392ms/step - loss: 0.0863 - accuracy: 0.9792 - val_loss: 0.0667 - val_accuracy: 0.9832  
<keras.callbacks.History at 0x7f3ec8f940d0>
```

### Question 8:

Save The Model

#### Output:

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

### Question 9:

Test The Model

#### Output:

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

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

```
35/35 [=====] - 1s 23ms/step - loss: 0.0512 - accuracy: 0.9874  
Test set  
Loss: 0.051  
Accuracy: 0.987
```