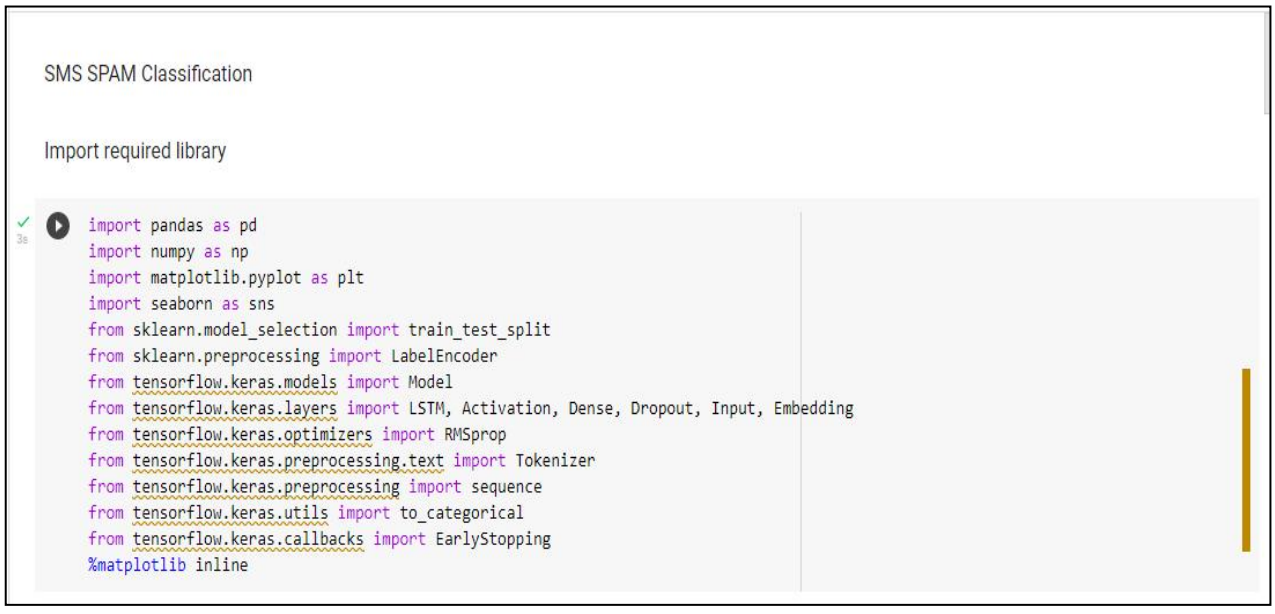
Assignment-4

AssignmentDate	31th October 2022
StudentName	K.Yokhalakshmi
StudentRoll Number	9517201906058
MaximumMarks	2Marks

Problem Statement:

Over recent years, as the popularity of mobile phone devices has increased, Short Message Service (SMS) has grown into a multi-billion dollar industry. At the same time, reduction in the cost of messaging services has resulted in growth in unsolicited commercial advertisements (spams) being sent to mobile phones. Due to Spam SMS, Mobile service providers suffer from some sort of financial problems as well as it reduces
calling time for users. Unfortunately, if the user accesses such Spam SMS they may face
the problem of virus or malware. When SMS arrives at mobile it will disturb mobile user privacy and concentration. It may lead to frustration for the user. So Spam SMS is one of
the major issues in the wireless communication world and it grows day by day.

1. Download the Dataset:-



2. Read dataset and do pre-processing

Read dataset and do pre-processing

```
[3] df = pd.read_csv(r'/content/spam.csv',encoding='latin-1')
    df.head(10)
```

	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
5	spam	FreeMsg Hey there darling it's been 3 week's n...	NaN	NaN	NaN
6	ham	Even my brother is not like to speak with me. ...	NaN	NaN	NaN
7	ham	As per your request 'Melle Melle (Oru Minnamin...	NaN	NaN	NaN
8	spam	WINNER!! As a valued network customer you have...	NaN	NaN	NaN
9	spam	Had your mobile 11 months or more? U R entitle...	NaN	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   v2      5572 non-null   object
dtypes: object(2)
memory usage: 87.2+ KB
```

Process the labels.

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

Split into training and test data.

```
[6] 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 = sequence.pad_sequences(sequences,maxlen=max_len)
```

4.Create Model
5. Add Layers (LSTM, Dense-(Hidden Layers), Output)

Create Model and add Layers

```
def RNN():
    inputs = Input(name='inputs',shape=[max_len])
    layer = Embedding(max_words,50,input_length=max_len)(inputs)
    layer = LSTM(128)(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('tanh')(layer)
    model = Model(inputs=inputs,outputs=layer)
    return model
```

6. Fit the Model
7.Save The Model
8.Test the Model

```
model = RNN()
model.summary()
model.compile(loss='binary_crossentropy',optimizer=RMSprop(),metrics=['accuracy','mse','mae'])
```

```
Model: "model"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 inputs (InputLayer)         [(None, 150)]             0

 embedding (Embedding)       (None, 150, 50)           50000

 lstm (LSTM)                 (None, 128)               91648

 FC1 (Dense)                 (None, 256)               33024

 activation (Activation)     (None, 256)               0

 dropout (Dropout)           (None, 256)               0

 out_layer (Dense)           (None, 1)                 257

 activation_1 (Activation)   (None, 1)                 0

=================================================================
Total params: 174,929
Trainable params: 174,929
Non-trainable params: 0
```

Fit the model

```
[10] 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 [==============================] - 14s 402ms/step - loss: 0.2933 - accuracy: 0.8948 - mse: 0.0800 - mae: 0.1550 - val_loss: 0.1701 - val_accuracy:
Epoch 2/10
28/28 [==============================] - 10s 357ms/step - loss: 5.5787 - accuracy: 0.6264 - mse: 0.3755 - mae: 0.4155 - val_loss: 13.0781 - val_accuracy
<keras.callbacks.History at 0x7f1cfa1e98d0>
```

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

```
[12] accr = model.evaluate(test_sequences_matrix,Y_test)
```

```
35/35 [==============================] - 2s 51ms/step - loss: 13.4439 - accuracy: 0.1184 - mse: 0.8816 - mae: 0.8816
```

```
[13] print('Test set\n  Loss: {:0.3f}\n  Accuracy: {:0.3f}'.format(accr[0],accr[1]))
```

```
Test set
  Loss: 13.444
  Accuracy: 0.118
```

Save the Model

```
[14] model.save(r"C:\Users\ADMIN\Downloads\model_lSTM.h5")
```

Test the Model

```
[15] from tensorflow.keras.models import load_model
     m2 = load_model(r"C:\Users\ADMIN\Downloads\model_lSTM.h5")
```

```
m2.evaluate(test_sequences_matrix,Y_test)
```

```
35/35 [==============================] - 2s 51ms/step - loss: 13.4439 - accuracy: 0.1184 - mse: 0.8816 - mae: 0.8816
[13.443947792053223,
 0.11838565021753311,
 0.8816143274307251,
 0.8816143274307251]
```