

Student's Name	Subreena Bano
Registration Number	960519104067
Assignment Number	04

```
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 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.sequence import pad_sequences
from keras.utils import to_categorical
from keras.callbacks import EarlyStopping
```

▼ READ DATASET AND PRE PROCESSING

```
df = pd.read_csv('/content/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
-	.	U dun sav so earlv hor... U c already then

```
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
```

```
X = df.v2
Y = df.v1
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=0.15)
```

```
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 = pad_sequences(sequences,maxlen=max_len)
```

▼ Create Model and Add Layers (LSTM, Dense-(Hidden Layers), 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)
model = Model(inputs=inputs,outputs=layer)
```

```
model.summary()
```

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
=====
```


▼ Compile the Model

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

▼ Train and Fit the Model

```
model.fit(sequences_matrix,Y_train,batch_size=128,epochs=10,
          validation_split=0.2)
```

```
Epoch 1/10
30/30 [=====] - 12s 286ms/step - loss: 0.3377 - accu:
Epoch 2/10
30/30 [=====] - 9s 301ms/step - loss: 0.0934 - accu:
Epoch 3/10
30/30 [=====] - 10s 327ms/step - loss: 0.0395 - accu:
Epoch 4/10
30/30 [=====] - 9s 317ms/step - loss: 0.0311 - accu:
Epoch 5/10
30/30 [=====] - 9s 294ms/step - loss: 0.0213 - accu:
Epoch 6/10
30/30 [=====] - 9s 305ms/step - loss: 0.0167 - accu:
Epoch 7/10
30/30 [=====] - 9s 316ms/step - loss: 0.0115 - accu:
Epoch 8/10
30/30 [=====] - 9s 286ms/step - loss: 0.0081 - accu:
Epoch 9/10
30/30 [=====] - 9s 310ms/step - loss: 0.0065 - accu:
Epoch 10/10
30/30 [=====] - 10s 346ms/step - loss: 0.0064 - accu:
<keras.callbacks.History at 0x7f03f70fe810>
```



▼ Save The Model

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

▼ Preprocessing the Test Dataset

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

▼ Testing the Model

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

27/27 [=====] - 1s 23ms/step - loss: 0.1346 - accura
```

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

```
Test set
```

```
Loss: 0.135
```

```
Accuracy: 0.982
```