

## **ASSIGNMENT 4**

**MAHENDRA ENGINEERING COLLEGE FOR  
WOMEN**

**NAME:P.AARTHI**

**CLASS:4 YEAR ECE**

**SUBJECT:IBM**

**REGISTER NO:611419106001**

In[37]:

```
#@titleImportLibraries
```

In[38]:

```
import pandas as pd
import numpy as np
import tensorflow as tf
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.callbacks import EarlyStopping
%matplotlib inline
```

In[39]:

```
#@titleLoadthedata
```

In[40]:

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

Out[40]:

	v1	v2	Unnamed: 2	Unnamed: 3	Unnamed: 4
0	ham	Go until jurong point, crazy.. Available only	NaN	NaN	NaN
1	ham	Oklar...Joking wif u oni...	NaN	NaN	NaN
2	spam	Free entry in 2a wkly comp to win FA Cup fina	NaN	NaN	NaN
3	ham	Udun says de arly hor... Ucal ready then say...	NaN	NaN	NaN
4	ham	Nah I don't think he goes to usf, hell vesaro...	NaN	NaN	NaN

In[41]:

```
#@titleDropunnecessarycolumns
```

In[42]:

```
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
```

```
objectdtypes: object(2) -----
```

```
memory usage: 87.2+KB
```

In[43]:

```
#@titleCreateinputandoutputvectorsandprocessthe labels
```

In[44]:

```
X=df.v2Y=df
.v1
le=LabelEncoder()
Y=le.fit_transform(Y)Y=Y.reshape(-1,1)
```

In[45]:

```
#@titleSplitthedatastfortrainingandtest.
```

In[46]:

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

In[47]:

```
#@titleProcesssthe data
```

In[48]:

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

In[49]:

```
#@titleDefinethemodel
```

In[50]:

```
defRNN():
    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)returnmodel
```

In[51]:

```
#@titleCallthefunctionandcompilethemodel
```

In[52]:

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

Model:"model\_1"

Layer(type)	OutputShape	Param#
=====		
inputs(InputLayer)	[(None,150)]	0
embedding_1(Embedding)	(None,150,50)	50000
lstm_1(LSTM)	(None,64)	29440
FC1(Dense)	(None,256)	16640
activation_2(Activation)	(None,256)	0

dropout_1(Dropout)	(None,256)	0
out_layer(Dense)	(None,1)	257
activation_3(Activation)	(None,1)	0

```

=====
Totalparams:96,337
Trainableparams:96,337
Non-trainableparams:0

```

In[53]:

```
#@titleFitthemodel
```

In[54]:

```
model.fit(sequences_matrix,Y_train,batch_size=128,epochs=10,validation_split=0.2,callbacks=[EarlyStopping(monitor='val_loss',
min_delta=0.001)])
```

Epoch1/10

30/30[=====]-10s267ms/step-loss:0.3345-accuracy:0.8730

- val\_loss:0.1491-val\_accuracy:0.9462Epoch2/10

30/30[=====]-8s251ms/step-loss:0.0887-accuracy:0.9794

- val\_loss:0.0625-val\_accuracy:0.9821Out[54]:

<keras.callbacks.Historyat0x7f0a5c167750>

In[55]:

```
#@titleProcesssthetestdata
```

In[56]:

```
test_sequences=tok.texts_to_sequences(X_test)
```

```
test_sequences_matrix=tf.keras.utils.pad_sequences(test_sequences,maxlen=max_len)
```

In[57]:

```
#@titleEvaluatethemodelwiththetest
```

In[58]:

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

27/27[=====]-1s21ms/step-loss:0.0643-accuracy:0.9797

In[59]:

```
print('Test set\nLoss: {:.3f}\nAccuracy: {:.3f}'.format(accr[0],accr[1]))Testset
```

Loss:0.064

Accuracy:0.980