Team ID : PNT2022TMID42525

Date : 27 Oct 2022

ASSIGNMNET_4 :- (Deepan.P.S)

Importing Required Libraries :

```
import numpy as np
import pandas as pd
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 import sequence
from keras.utils import pad_sequences
from keras.utils import to_categorical
from keras.callbacks import EarlyStopping
```

Reading And Preprocessing The Dataset :

```
#reading ds
ds = pd.read_csv('/content/spam.csv', encoding="ISO-8859-1")
ds.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

```
#preprocessing ds
ds.info() #checking datatype
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5572 entries, 0 to 5571
```

```
Data columns (total 5 columns):
 #   Column      Non-Null Count  Dtype 
---  ------      --------------  ----- 
 0   v1          5572 non-null   object
 1   v2          5572 non-null   object
 2   Unnamed: 2  50 non-null     object
 3   Unnamed: 3  12 non-null     object
 4   Unnamed: 4  6 non-null      object
dtypes: object(5)
memory usage: 217.8+ KB
```

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

```
from sklearn.model_selection import train_test_split
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)
```

Creating Model And Adding Layers:

```
#adding layers in model
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_1"
_________________________________________________________________
 Layer (type)                Output Shape              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

=================================================================
Total params: 96,337
Trainable params: 96,337
Non-trainable params: 0
_________________________________________________________________
```

Compiling The Model:

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

Fit The Model:

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

```
Epoch 1/10
30/30 [==============================] - 8s 30ms/step - loss: 0.3282 - accuracy: 0.8730
Epoch 2/10
30/30 [==============================] - 0s 13ms/step - loss: 0.0863 - accuracy: 0.9770
Epoch 3/10
30/30 [==============================] - 0s 14ms/step - loss: 0.0430 - accuracy: 0.9863
Epoch 4/10
30/30 [==============================] - 0s 13ms/step - loss: 0.0331 - accuracy: 0.9900
Epoch 5/10
30/30 [==============================] - 0s 13ms/step - loss: 0.0248 - accuracy: 0.9937
Epoch 6/10
30/30 [==============================] - 0s 14ms/step - loss: 0.0187 - accuracy: 0.9942
Epoch 7/10
30/30 [==============================] - 0s 13ms/step - loss: 0.0128 - accuracy: 0.9963
Epoch 8/10
30/30 [==============================] - 0s 13ms/step - loss: 0.0105 - accuracy: 0.9966
Epoch 9/10
30/30 [==============================] - 0s 13ms/step - loss: 0.0065 - accuracy: 0.9971
Epoch 10/10
30/30 [==============================] - 0s 13ms/step - loss: 0.0061 - accuracy: 0.9984
<keras.callbacks.History at 0x7f3850294ed0>
```

Saving The Model:

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

Testing The Model:

```
#preprocessing test ds
test_sequences = tok.texts_to_sequences(X_test)
test_sequences_matrix = pad_sequences(test_sequences,maxlen=max_len)

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

```
27/27 [==============================] - 0s 6ms/step - loss: 0.1245 - accuracy: 0.9844
```

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

```
Test set
  Loss: 0.125
  Accuracy: 0.984
```