Name:Yuvaprakash K

RollNo:611219106086

Date:02/11/2022

1.Download dataset /content/spam.csv

2.Import library

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
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 Adam
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
```

3.Read the dataset

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

```
          v1                                        v2  Unnamed: 2  Unnamed: 3  Unnamed: 4
```

```
dataset.drop(['Unnamed: 2', 'Unnamed: 3', 'Unnamed: 4'],axis=1,inplace=True)
from wordcloud import WordCloud, STOPWORDS, ImageColorGenerator
A = dataset.v2
B = dataset.v1
le = LabelEncoder()
B = le.fit_transform(B)
B = B.reshape(-1,1)
A_train,A_test,B_train,B_test = train_test_split(A,B,test_size=0.25)
max_words = 1000
max_len = 150
tok = Tokenizer(num_words=max_words)
tok.fit_on_texts(A_train)
sequences = tok.texts_to_sequences(A_train)
sequences_matrix = pad_sequences(sequences,maxlen=max_len)
```

4.Create the Model

```
inputs = Input(shape=[max_len])
layer = Embedding(max_words,50,input_length=max_len)(inputs)
```

5.Add a layer

```
layer = LSTM(128)(layer)
layer = Dense(128)(layer)
layer = Activation('relu')(layer)
layer = Dropout(0.5)(layer)
layer = Dense(1.5)(layer)
layer = Activation('sigmoid')(layer)
model = Model(inputs=inputs,outputs=layer)
model.summary()
Model: "model"
```

```
Model: "model"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 input_1 (InputLayer)        [(None, 150)]             0

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

 lstm (LSTM)                 (None, 128)               91648

 dense (Dense)               (None, 128)               16512
 activation (Activation)     (None, 128)               0

 dropout (Dropout)           (None, 128)               0

 dense_1 (Dense)             (None, 1)                 129

 activation_1 (Activation)   (None, 1)                 0

=================================================================
Total params: 158,289
Trainable params: 158,289
Non-trainable params: 0
_________________________________________________________________
```

6.Compile the model

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

7.Fit the model

```
model.fit(sequences_matrix,B_train,batch_size=20,epochs=15,validation_split=0.2)
```

```
Epoch 1/15
168/168 [==============================] - 35s 190ms/step - loss: 0.1919 - accuracy: 0.9
Epoch 2/15
168/168 [==============================] - 30s 180ms/step - loss: 0.0430 - accuracy: 0.9
Epoch 3/15
168/168 [==============================] - 31s 183ms/step - loss: 0.0225 - accuracy: 0.9
Epoch 4/15
168/168 [==============================] - 30s 181ms/step - loss: 0.0117 - accuracy: 0.9
Epoch 5/15
168/168 [==============================] - 30s 179ms/step - loss: 0.0103 - accuracy: 0.9
Epoch 6/15
168/168 [==============================] - 30s 179ms/step - loss: 0.0071 - accuracy: 0.9
Epoch 7/15
168/168 [==============================] - 30s 179ms/step - loss: 0.0092 - accuracy: 0.9
Epoch 8/15
168/168 [==============================] - 32s 190ms/step - loss: 0.0519 - accuracy: 0.9
Epoch 9/15
168/168 [==============================] - 30s 181ms/step - loss: 0.0081 - accuracy: 0.9
Epoch 10/15
168/168 [==============================] - 30s 181ms/step - loss: 0.0039 - accuracy: 0.9
Epoch 11/15
168/168 [==============================] - 30s 180ms/step - loss: 0.0020 - accuracy: 0.9
Epoch 12/15
168/168 [==============================] - 30s 181ms/step - loss: 0.0019 - accuracy: 0.9
Epoch 13/15
168/168 [==============================] - 31s 182ms/step - loss: 0.0013 - accuracy: 0.9
```

```
Epoch 14/15
168/168 [==============================] - 31s 184ms/step - loss: 0.0016 - accuracy: 0.9
Epoch 15/15
168/168 [==============================] - 32s 189ms/step - loss: 0.0011 - accuracy: 0.9
<keras.callbacks.History at 0x7f729250ced0>
```

8.Save the model

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

9.Test the model

```
test_sequences = tok.texts_to_sequences(A_test)
test_sequences_matrix = pad_sequences(test_sequences,maxlen=max_len)
accuracy1 = model.evaluate(test_sequences_matrix,B_test)
```

```
44/44 [==============================] - 3s 78ms/step - loss: 0.1392 - accuracy: 0.9828
```