Name:Prathap S

RollNo:611219106058

Date:02/11/2022

1.Download dataset /content/spam.csv

2.Import the 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

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

```
data.drop(['Unnamed: 2', 'Unnamed: 3', 'Unnamed: 4'],axis=1,inplace=True)
from wordcloud import WordCloud, STOPWORDS, ImageColorGenerator
A = data.v2
B = data.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.Creating a Model

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

5.Add 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=20,validation_split=0.2)
```

```
Epoch 1/20
168/168 [==============================] - 42s 249ms/step - loss: 0.0018 - accur
Epoch 2/20
168/168 [==============================] - 37s 219ms/step - loss: 0.0016 - accur
Epoch 3/20
168/168 [==============================] - 30s 180ms/step - loss: 0.0014 - accur
Epoch 4/20
168/168 [==============================] - 30s 180ms/step - loss: 0.0015 - accur
Epoch 5/20
168/168 [==============================] - 30s 181ms/step - loss: 0.0017 - accur
Epoch 6/20
168/168 [==============================] - 31s 186ms/step - loss: 0.0017 - accur
Epoch 7/20
168/168 [==============================] - 31s 182ms/step - loss: 0.0149 - accur
Epoch 8/20
168/168 [==============================] - 30s 180ms/step - loss: 0.0180 - accur
Epoch 9/20
168/168 [==============================] - 30s 181ms/step - loss: 0.0079 - accur
Epoch 10/20
168/168 [==============================] - 30s 180ms/step - loss: 0.0018 - accur
Epoch 11/20
168/168 [==============================] - 30s 181ms/step - loss: 0.0026 - accur
Epoch 12/20
168/168 [==============================] - 31s 185ms/step - loss: 0.0015 - accur
Epoch 13/20
168/168 [==============================] - 31s 181ms/step - loss: 0.0018 - accur
Epoch 14/20
168/168 [==============================] - 30s 182ms/step - loss: 0.0017 - accur
Epoch 15/20
168/168 [==============================] - 30s 180ms/step - loss: 0.0016 - accur
Epoch 16/20
168/168 [==============================] - 31s 182ms/step - loss: 0.0017 - accur
Epoch 17/20
168/168 [==============================] - 30s 181ms/step - loss: 0.0016 - accur
Epoch 18/20
168/168 [==============================] - 30s 181ms/step - loss: 0.0014 - accur
Epoch 19/20
168/168 [==============================] - 31s 187ms/step - loss: 0.0012 - accur
Epoch 20/20
168/168 [==============================] - 30s 181ms/step - loss: 0.0017 - accur
<keras.callbacks.History at 0x7f7296b6ae90>
```

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 [==============================] - 4s 81ms/step - loss: 0.0962 - accuracy:
```