Assignmet_4

1. Import required library

```
[9] 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
    import keras
    from keras.utils import to_categorical
    from keras.callbacks import EarlyStopping
    %matplotlib inline
```

2. Read dataset and do pre-processing

```
[10] 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
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

```
[11] 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
```

```
[12] X = df.v2
     Y = df.v1
     le = LabelEncoder()
     Y = le.fit_transform(Y)
     Y = Y.reshape(-1,1)
```

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

```
X_train
```

```
2213    Goodmorning, today i am late for 2hrs. Because...
906     all the lastest from Stereophonics, Marley, Di...
4334    Now u sound like manky scouse boy steve,like! ...
176                           U still going to the mall?
4093                            how are you? I miss you!
                              ...
5495    Good afternoon, my love ... How goes your day ...
650     Thats cool! Sometimes slow and gentle. Sonetim...
2242    Nope wif my sis lor... Aft bathing my dog then...
4010    Please call our customer service representativ...
3306    Set a place for me in your heart and not in yo...
Name: v2, Length: 4736, dtype: object
```

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

3. Create Model

4. Add Layers (LSTM, Dense-(Hidden Layers), Output)

+ Code + Text

```
[16] def RNN():
         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)
         return model
```

5. Compile the Model

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

```
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
_________________________________________________________________
```

6. Fit the Model

```
[18] model.fit(sequences_matrix,Y_train,batch_size=128,epochs=10,
               validation_split=0.2,callbacks=[EarlyStopping(monitor='val_loss',min_delta=0.0001)])
```

```
Epoch 1/10
30/30 [==============================] - 15s 401ms/step - loss: 0.3063 - accuracy: 0.8949 - val_loss: 0.1306 - val_accura
Epoch 2/10
30/30 [==============================] - 8s 271ms/step - loss: 0.0848 - accuracy: 0.9797 - val_loss: 0.0639 - val_accurac
<keras.callbacks.History at 0x7f7338c44f10>
```

6. Fit the Model

```
model.fit(sequences_matrix,Y_train,batch_size=128,epochs=10,
          validation_split=0.2,callbacks=[EarlyStopping(monitor='val_loss',min_delta=0.0001)])
```

```
10
==============================] - 15s 401ms/step - loss: 0.3063 - accuracy: 0.8949 - val_loss: 0.1306 - val_accuracy: 0.9673
10
==============================] - 8s 271ms/step - loss: 0.0848 - accuracy: 0.9797 - val_loss: 0.0639 - val_accuracy: 0.9821
allbacks.History at 0x7f7338c44f10>
```

7. Save The Model

```
[19] model.save('NLP.h5')
```

8. Test The Model

```
[20] test_sequences = tok.texts_to_sequences(X_test)
     test_sequences_matrix = keras.utils.pad_sequences(test_sequences,maxlen=max_len)
```

```
[21] accr = model.evaluate(test_sequences_matrix,Y_test)
```

```
27/27 [==============================] - 1s 47ms/step - loss: 0.0461 - accuracy: 0.9904
```

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

```
Test set
  Loss: 0.046
  Accuracy: 0.990
```