1.Upload dataset

In [1]:

```
!unzip '/content/archive.zip'
```

```
Archive:  /content/archive.zip
  inflating: spam.csv
```

2.Import required library

In [2]:

```
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
from keras.preprocessing import sequence
from keras.utils import pad_sequences
from keras.utils import to_categorical
from keras.callbacks import EarlyStopping
```

3.Read dataset and data preprocessing

In [3]:

```
import chardet
```

In [4]:

```
with open('/content/spam.csv', 'rb') as f:
    enc = chardet.detect(f.read())
```

In [5]:

```
df=pd.read_csv('/content/spam.csv', encoding = enc['encoding'])
df.head()
```

Out[5]:

	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

In [ ]:

```
df.isnull().sum()
```

Out[ ]:

```
v1               0
v2               0
Unnamed: 2    5522
Unnamed: 3    5560
Unnamed: 4    5566
dtype: int64
```

In [ ]:

```
df.info()
df
```

```
<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
```

Out[ ]:

	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**
...	**...**	**...**	**...**	**...**	**...**
5567	**spam**	**This is the 2nd time we have tried 2 contact u...**	**NaN**	**NaN**	**NaN**
5568	**ham**	**Will Ì_ b going to esplanade fr home?**	**NaN**	**NaN**	**NaN**
5569	**ham**	**Pity, * was in mood for that. So...any other s...**	**NaN**	**NaN**	**NaN**
5570	**ham**	**The guy did some bitching but I acted like i'd...**	**NaN**	**NaN**	**NaN**
5571	**ham**	**Rofl. Its true to its name**	**NaN**	**NaN**	**NaN**

5572 rows × 5 columns

In [6]:

```
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
```

In [7]:

```
X = df.v2
Y = df.v1
```

```
le = LabelEncoder()
Y = le.fit_transform(Y)
Y = Y.reshape(-1,1)
```

In [8]:

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

In [9]:

```
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)
```

4.Create model and 5.Add layers(LSTM,Dense-(Hidden layers),Output

In [11]:

```
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"
_________________________________________________________________
 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.Compile the model

In [12]:

```
model.compile(optimizer='adam',loss='mse')
```

7.Fit the model

7.Fit the model

In [13]:

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

```
Epoch 1/10
30/30 [==============================] - 12s 293ms/step - loss: 0.1576 - val_loss: 0.0911
Epoch 2/10
30/30 [==============================] - 8s 274ms/step - loss: 0.0655 - val_loss: 0.0103
Epoch 3/10
30/30 [==============================] - 8s 273ms/step - loss: 0.0146 - val_loss: 0.0072
Epoch 4/10
30/30 [==============================] - 8s 271ms/step - loss: 0.0097 - val_loss: 0.0066
Epoch 5/10
30/30 [==============================] - 8s 272ms/step - loss: 0.0072 - val_loss: 0.0068
Epoch 6/10
30/30 [==============================] - 8s 271ms/step - loss: 0.0053 - val_loss: 0.0063
Epoch 7/10
30/30 [==============================] - 8s 268ms/step - loss: 0.0037 - val_loss: 0.0066
Epoch 8/10
30/30 [==============================] - 8s 270ms/step - loss: 0.0032 - val_loss: 0.0071
Epoch 9/10
30/30 [==============================] - 9s 318ms/step - loss: 0.0028 - val_loss: 0.0070
Epoch 10/10
30/30 [==============================] - 8s 269ms/step - loss: 0.0022 - val_loss: 0.0076
```

Out[13]:

```
<keras.callbacks.History at 0x7f2c02c67f10>
```

8.Save the model

In [14]:

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

9.Test the model

In [16]:

```
test_sequences = tok.texts_to_sequences(X_test)
test_sequences_matrix = pad_sequences(test_sequences,maxlen=max_len)
```

In [17]:

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

```
27/27 [==============================] - 1s 25ms/step - loss: 0.0170
```