Import required library

```
import pandas as pd
import numpy as np
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 to_categorical
from tensorflow.keras.models import Sequential
```

#Read Dataset and do preprocessing

```
df = pd.read_csv('/content/spam.csv',delimiter=',',encoding='latin 1')
df.head()
```

```
 v1 v2 Unnamed: 2  \
0 ham Go until jurong point, crazy.. Available only ... NaN  1 ham Ok lar... Joking wif u oni...

NaN  2 spam Free entry in 2 a wkly comp to win FA Cup fina... NaN  3 ham U dun say so early

hor... U c already then say... NaN  4 ham Nah I don't think he goes to usf, he lives aro... NaN

 Unnamed: 3 Unnamed: 4
0 NaN NaN
1 NaN NaN
2 NaN NaN
3 NaN NaN
4 NaN NaN
```

```
 df.drop(['Unnamed: 2', 'Unnamed: 3', 'Unnamed:
4'],axis=1,inplace=True) #dropping unwanted
 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
```

```
df.groupby(['v1']).size() # Count of Spam and Ham values,
v1
ham 4825
spam 747
dtype: int64
```

```
# Label Encoding target column
X = df.v2
Y = df.v1
le = LabelEncoder()
Y = le.fit_transform(Y)
Y = Y.reshape(-1,1)
```

```
 X_train,X_test,Y_train,Y_test = train_test_split(X,Y,test_size=0.15)  # Test and train split
```

```
 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 = sequence.pad_sequences(sequences,maxlen=max_len)
```

Create Model and Add Layers (LSTM, Dense-(Hidden Layers), Output) *#Create Model*

```
  input = Input(name='InputLayer',shape=[max_len])
#Add Layers (LSTM, Dense-(Hidden Layers), Output)
```

```
 layer = Embedding(max_words,50,input_length=max_len)(inputs)   layer =
LSTM(64)(layer)
 layer = Dense(256,name='FullyConnectedLayer1')(layer)  layer =
Activation('relu')(layer)
 layer = Dropout(0.5)(layer)
 layer = Dense(1,name='OutputLayer')(layer)
 layer = Activation('sigmoid')(layer)
```

```
#Compile The Model
```

```
model = Model(inputs=inputs,outputs=layer)
model.summary()
model.compile(loss='binary_crossentropy',optimizer=RMSprop(),metrics=[ 'accuracy'])
```

```
Model: "model_1"
_________________________________________________________________
Layer (type) Output Shape Param #
=================================================================
InputLayer (InputLayer) [(None, 150)] 0
```

```
 embedding_3 (Embedding) (None, 150, 50) 50000   lstm_3 (LSTM) (None, 64)

29440

 FullyConnectedLayer1 (Dense (None, 256) 16640   )

 activation_6 (Activation) (None, 256) 0   dropout_3 (Dropout) (None, 256) 0

OutputLayer (Dense) (None, 1) 257   activation_7 (Activation) (None, 1) 0

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

The Model

```
 model.fit(sequences_matrix,Y_train,batch_size=128,epochs=30,
validation_split=0.2)
Epoch 1/30
30/30 [==============================] - 8s 255ms/step - loss: 0.3000  - accuracy:
0.8788 - val_loss: 0.5301 - val_accuracy: 0.8481 Epoch 2/30
30/30 [==============================] - 7s 247ms/step - loss: 0.2767  - accuracy:
0.8820 - val_loss: 0.5733 - val_accuracy: 0.8354 Epoch 3/30
30/30 [==============================] - 8s 269ms/step - loss: 0.2672  - accuracy:
0.8881 - val_loss: 0.5753 - val_accuracy: 0.8302 Epoch 4/30
30/30 [==============================] - 9s 294ms/step - loss: 0.2506  - accuracy:
0.8928 - val_loss: 0.6359 - val_accuracy: 0.7806 Epoch 5/30
30/30 [==============================] - 7s 236ms/step - loss: 0.2406  - accuracy:
0.9010 - val_loss: 0.6116 - val_accuracy: 0.8175 Epoch 6/30
30/30 [==============================] - 7s 238ms/step - loss: 0.2278  - accuracy:
0.9055 - val_loss: 0.6498 - val_accuracy: 0.8049 Epoch 7/30
30/30 [==============================] - 7s 237ms/step - loss: 0.2178  - accuracy:
0.9118 - val_loss: 0.6866 - val_accuracy: 0.7806 Epoch 8/30
30/30 [==============================] - 7s 234ms/step - loss: 0.2049  - accuracy:
0.9139 - val_loss: 0.7650 - val_accuracy: 0.8133 Epoch 9/30
30/30 [==============================] - 7s 235ms/step - loss: 0.1999  - accuracy:
0.9213 - val_loss: 0.7879 - val_accuracy: 0.7711 Epoch 10/30
30/30 [==============================] - 7s 237ms/step - loss: 0.1892  - accuracy:
0.9242 - val_loss: 0.8410 - val_accuracy: 0.8080 Epoch 11/30
30/30 [==============================] - 7s 239ms/step - loss: 0.1856  - accuracy:
0.9277 - val_loss: 0.8711 - val_accuracy: 0.8049 Epoch 12/30
30/30 [==============================] - 7s 238ms/step - loss: 0.1710  - accuracy:
0.9359 - val_loss: 0.9013 - val_accuracy: 0.7901 Epoch 13/30
30/30 [==============================] - 7s 237ms/step - loss: 0.1683  - accuracy:
```

```
0.9348 - val_loss: 0.9538 - val_accuracy: 0.8070 Epoch 14/30
30/30 [==============================] - 7s 236ms/step - loss: 0.1686  - accuracy:
0.9382 - val_loss: 0.9887 - val_accuracy: 0.7943 Epoch 15/30
30/30 [==============================] - 7s 238ms/step - loss: 0.1533  - accuracy:
0.9435 - val_loss: 1.0721 - val_accuracy: 0.7932 Epoch 16/30
30/30 [==============================] - 7s 237ms/step - loss: 0.1519  - accuracy:
0.9424 - val_loss: 1.0436 - val_accuracy: 0.7932 Epoch 17/30
          30/30 [==============================] - 7s 237ms/step - loss: 0.1422
- accuracy: 0.9440 - val_loss: 1.0637 - val_accuracy: 0.7648 Epoch 18/30
30/30 [==============================] - 7s 236ms/step - loss: 0.1404  - accuracy:
0.9475 - val_loss: 1.0544 - val_accuracy: 0.7985 Epoch 19/30
30/30 [==============================] - 7s 236ms/step - loss: 0.1307  - accuracy:
0.9514 - val_loss: 1.1296 - val_accuracy: 0.7584 Epoch 20/30
30/30 [==============================] - 7s 236ms/step - loss: 0.1256  - accuracy:
0.9520 - val_loss: 1.2484 - val_accuracy: 0.7711 Epoch 21/30
30/30 [==============================] - 8s 278ms/step - loss: 0.1263  - accuracy:
0.9535 - val_loss: 1.2196 - val_accuracy: 0.7679 Epoch 22/30
30/30 [==============================] - 7s 238ms/step - loss: 0.1145  - accuracy:
0.9591 - val_loss: 1.2910 - val_accuracy: 0.7426 Epoch 23/30
30/30 [==============================] - 7s 237ms/step - loss: 0.1153  - accuracy:
0.9575 - val_loss: 1.3979 - val_accuracy: 0.7542 Epoch 24/30
30/30 [==============================] - 7s 237ms/step - loss: 0.1096  - accuracy:
0.9591 - val_loss: 1.4036 - val_accuracy: 0.7637 Epoch 25/30
30/30 [==============================] - 7s 237ms/step - loss: 0.1031  - accuracy:
0.9622 - val_loss: 1.4102 - val_accuracy: 0.7806 Epoch 26/30
30/30 [==============================] - 7s 239ms/step - loss: 0.1023  - accuracy:
0.9617 - val_loss: 1.4230 - val_accuracy: 0.7447 Epoch 27/30
30/30 [==============================] - 7s 239ms/step - loss: 0.0929  - accuracy:
0.9644 - val_loss: 1.5604 - val_accuracy: 0.7110 Epoch 28/30
30/30 [==============================] - 7s 239ms/step - loss: 0.0926  - accuracy:
0.9654 - val_loss: 1.5457 - val_accuracy: 0.7743 Epoch 29/30
30/30 [==============================] - 7s 237ms/step - loss: 0.0923  - accuracy:
0.9649 - val_loss: 1.5456 - val_accuracy: 0.7437 Epoch 30/30
30/30 [==============================] - 7s 236ms/step - loss: 0.0831  - accuracy:
0.9681 - val_loss: 1.7871 - val_accuracy: 0.7489

<keras.callbacks.History at 0x7f59d6db4f50>
```

#SAVE MODEL

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

#TEST The MODEL

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

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

```
print('Accuracy: {:0.3f}'.format(accuracy[1]))
```

```
27/27 [==============================] - 1s 20ms/step - loss: 1.9870 - accuracy: 
0.7500
Accuracy: 0.750
```

```
y_pred = model.predict(test_sequences_matrix)
print(y_pred[25:40].round(3))
```

```
27/27 [==============================] - 1s 19ms/step [[0.   ]
 [0.998]
 [0.   ]
 [0.   ]
 [0.002]
 [0.   ]
 [0.   ]
 [0.   ]
 [0.   ]
 [0.681]
 [0.001]
 [0.   ]
 [0.   ]
 [0.   ]
 [0.   ]]
```