**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** to\_categorical

**from** keras.callbacks **import** EarlyStopping

**from** keras.models **import** load\_model

**%matplotlib** inline

In [ ]:

In [14]:

df **=** pd**.**read\_csv('spam.csv',delimiter**=**',',encoding**=**'latin-1')

df**.**head()

Out[14]:

|  | **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 |

In [15]:

df**.**drop(['Unnamed: 2', 'Unnamed: 3', 'Unnamed: 4'],axis**=**1,inplace**=True**)

df**.**info()

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 [16]:

*# data distribution*

sns**.**countplot(df**.**v1)

plt**.**xlabel('Label')

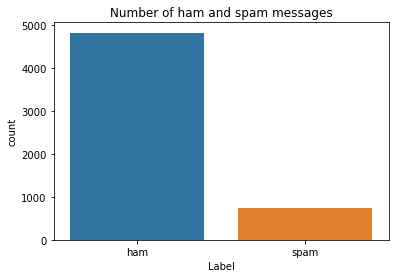
plt**.**title('Number of ham and spam messages')

C:\Users\sathi\anaconda3\lib\site-packages\seaborn\\_decorators.py:36: FutureWarning: Pass the following variable as a keyword arg: x. From version 0.12, the only valid positional argument will be `data`, and passing other arguments without an explicit keyword will result in an error or misinterpretation.

warnings.warn(

Out[16]:

Text(0.5, 1.0, 'Number of ham and spam messages')



In [17]:

*# splitting data into input and output*

X **=** df**.**v2

Y **=** df**.**v1

le **=** LabelEncoder()

Y **=** le**.**fit\_transform(Y)

Y **=** Y**.**reshape(**-**1,1)

In [18]:

*# test and train split*

X\_train,X\_test,Y\_train,Y\_test **=** train\_test\_split(X,Y,test\_size**=**0.15)

In [19]:

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)

In [20]:

*#layers of the model*

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)

In [21]:

model **=** Model(inputs**=**inputs,outputs**=**layer)

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

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

In [22]:

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 [==============================] - 14s 282ms/step - loss: 0.3317 - accuracy: 0.8720 - val\_loss: 0.1777 - val\_accuracy: 0.9768

Epoch 2/10

30/30 [==============================] - 7s 226ms/step - loss: 0.0930 - accuracy: 0.9776 - val\_loss: 0.0651 - val\_accuracy: 0.9842

Out[22]:

In [30]:

*# saving a model*

model**.**save("my\_model")

WARNING:absl:Found untraced functions such as lstm\_cell\_layer\_call\_fn, lstm\_cell\_layer\_call\_and\_return\_conditional\_losses while saving (showing 2 of 2). These functions will not be directly callable after loading.

INFO:tensorflow:Assets written to: my\_model\assets

INFO:tensorflow:Assets written to: my\_model\assets

**loading model**

**model = load\_model('saved\_model/my\_model')**

In [31]:

test\_sequences **=** tok**.**texts\_to\_sequences(X\_test)

test\_sequences\_matrix **=** sequence**.**pad\_sequences(test\_sequences,maxlen**=**max\_len)

In [32]:

accr **=** model**.**evaluate(test\_sequences\_matrix,Y\_test)

27/27 [==============================] - 1s 34ms/step - loss: 0.0686 - accuracy: 0.9844

In [33]:

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

Test set

Loss: 0.069

Accuracy: 0.984

In [34]:

y\_pred **=** model**.**predict(test\_sequences\_matrix)

print(y\_pred[0:10])

27/27 [==============================] - 1s 33ms/step

[[0.0052104 ]

[0.00270706]

[0.01115318]

[0.00448523]

[0.02838335]

[0.00183202]

[0.43712318]

[0.00751833]

[0.00228402]

[0.78971624]]

In [35]:

print(Y\_test[0:10])

[[0]

[0]

[0]

[0]

[0]

[0]

[0]

[0]

[0]

[1]]