**1.Download the dataset**

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

**3.Read Dataset and do preprocessing**

df **=** pd**.**read\_csv('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 |

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

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

*# Count of Spam and Ham values*

df**.**groupby(['v1'])**.**size()

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)

*# Test and train split*

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

*# Tokenisation function*

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)

**4.Create Model and 5. Add Layers (LSTM, Dense-(Hidden Layers), Output)**

*# Creating LSTM model*

inputs **=** Input(name**=**'InputLayer',shape**=**[max\_len])

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)

**6.Compile the model**

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

model**.**summary()

model**.**compile(loss**=**'binary\_crossentropy',optimizer**=**RMSprop(),metrics**=**['accuracy'])

Model: "model\_2"

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

Layer (type) Output Shape Param #

=================================================================

InputLayer (InputLayer) [(None, 150)] 0

embedding\_2 (Embedding) (None, 150, 50) 50000

lstm\_2 (LSTM) (None, 64) 29440

FullyConnectedLayer1 (Dense (None, 256) 16640

)

activation\_4 (Activation) (None, 256) 0

dropout\_2 (Dropout) (None, 256) 0

OutputLayer (Dense) (None, 1) 257

activation\_5 (Activation) (None, 1) 0

=================================================================

Total params: 96,337

Trainable params: 96,337

Non-trainable params: 0

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

**7.Fit the Model**

model**.**fit(sequences\_matrix,Y\_train,batch\_size**=**128,epochs**=**10,

validation\_split**=**0.2)

Epoch 1/10

30/30 [==============================] - ETA: 0s - loss: 0.3374 - accuracy: 0.8712

**8.Save the Model**

model**.**save("model\_1")

WARNING:absl:Function `\_wrapped\_model` contains input name(s) InputLayer with unsupported characters which will be renamed to inputlayer in the SavedModel.

WARNING:absl:Found untraced functions such as lstm\_cell\_1\_layer\_call\_fn, lstm\_cell\_1\_layer\_call\_and\_return\_conditional\_losses while saving (showing 2 of 2). These functions will not be directly callable after loading.

**9.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 36ms/step - loss: 0.1163 - accuracy: 0.9856

Accuracy: 0.986

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

print(y\_pred[25:40]**.**round(3))

27/27 [==============================] - 1s 20ms/step

[[0. ]

[0. ]

[0. ]

[0. ]

[0. ]

[0.002]

[0. ]

[0.024]

[0. ]

[0. ]

[0. ]

[0. ]

[0. ]

[0. ]

[0. ]]

print(Y\_test[25:40])

[[0]

[0]

[0]

[0]

[0]

[0]

[0]

[0]

[0]

[0]

[0]

[0]

[0]

[0]

[0]]