

## ▼ Download Dataset

```
from google.colab import drive
drive.mount('/content/drive')
```

Mounted at /content/drive

## ▼ Import required library

```
import numpy as np
import pandas as pd
import keras
```

```
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
%matplotlib inline
```

## ▼ Read Dataset and do pre-processing

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

## ▼ drop the unnamed values NaN

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

3	ham	U dun say so early hor... U c already then say...	NaN	NaN	NaN
---	-----	---	-----	-----	-----

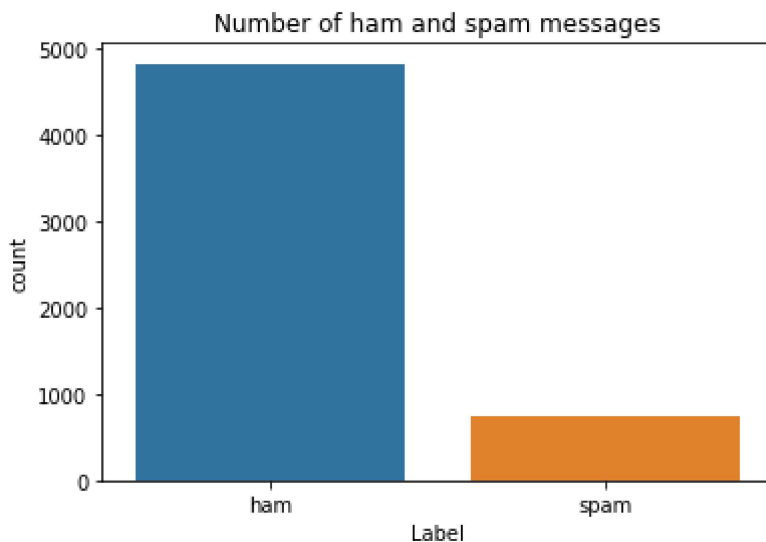
```
df.shape
```

```
(5572, 2)
```

```
sns.countplot(df.v1)
plt.xlabel('Label')
plt.title('Number of ham and spam messages')
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning: Pass the following variables as keyword arguments: {'x': 'Label', 'y': 'count'}. This will ensure compatibility in future versions of Seaborn.
```

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



```
X = df.v2
Y = df.v1
#label encoding for Y
le = LabelEncoder()
Y = le.fit_transform(Y)
Y = Y.reshape(-1,1)
```

```
#split into train and test sets
X_train,X_test,Y_train,Y_test = train_test_split(X,Y,test_size=0.20)
```

```
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)

```

## ▼ Add Layers (LSTM, Dense-(Hidden Layers), Output)

```

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)

```

## ▼ Compile the Model

```

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

```

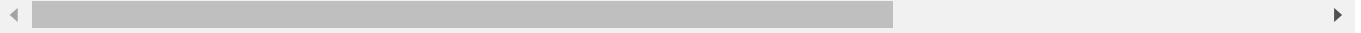
=====
Total params: 96,337
Trainable params: 96,337
Non-trainable params: 0

```

## ▼ 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)])
```

```
Epoch 1/10  
28/28 [=====] - 10s 256ms/step - loss: 0.3376 - accuracy: 0.877  
Epoch 2/10  
28/28 [=====] - 7s 238ms/step - loss: 0.0930 - accuracy: 0.9778  
<keras.callbacks.History at 0x7fc5ff20dcd0>
```



## ▼ Save the Model

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

## ▼ Test the Model

```
#processing test data  
test_sequences = tok.texts_to_sequences(X_test)  
test_sequences_matrix = keras.utils.pad_sequences(test_sequences,maxlen=max_len)
```

```
#evaluation of our model  
accr = model.evaluate(test_sequences_matrix,Y_test)  
print('Test set\n Loss: {:.3f}\n Accuracy: {:.3f}'.format(accr[0],accr[1]))
```

```
35/35 [=====] - 1s 21ms/step - loss: 0.0492 - accuracy: 0.9839  
Test set  
Loss: 0.049  
Accuracy: 0.984
```