

## 1.Dataset Upload

Upload the dataset <https://www.kaggle.com/code/kredy10/simple-lstm-for-textclassification/data>

## 2.Importing Libraries

```
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.a)Reading the Dataset

```
df = pd.read_csv('/content/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 FAfi Cupna...	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

## b)Pre-processing the dataset

```
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

```

```

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)

```

```

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.Creating 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) model =
Model(inputs=inputs,outputs=layer) model.summary()

```

Model: "model"

## 5.Add layers(LSTM,Dense-(Hidden Layers),Output)

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

```
model.compile(loss='binary_crossentropy',optimizer=RMSprop(),metrics=['accuracy'])
```

## 7. Train and Fit the model

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

```
Epoch 1/10
30/30 [=====] - 8s 263ms/step - loss: 0.0060 - accurac
Epoch 2/10
30/30 [=====] - 8s 263ms/step - loss: 0.0036 - accurac
Epoch 3/10
30/30 [=====] - 8s 263ms/step - loss: 0.0572 - accurac
Epoch 4/10
30/30 [=====] - 8s 262ms/step - loss: 0.0038 - accurac Epoch 5/10
30/30 [=====] - 8s 261ms/step - loss: 0.0018 - accurac
Epoch 6/10
30/30 [=====] - 8s 263ms/step - loss: 0.0022 - accurac
Epoch 7/10
30/30 [=====] - 9s 310ms/step - loss: 0.0020 - accurac
Epoch 8/10
30/30 [=====] - 8s 261ms/step - loss: 0.0015 - accurac
Epoch 9/10
30/30 [=====] - 8s 264ms/step - loss: 0.0015 - accurac
Epoch 10/10
30/30 [=====] - 8s 263ms/step - loss: 0.0021 - accurac
<keras.callbacks.History at 0x7f2b60b5f110>
```

## 8. Saving Model

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

Preprocessing the Test Dataset

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

## 9. Testing the Model

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

```
27/27 [=====] - 1s 21ms/step - loss: 0.2618 - accuracy
```

```
print('Test set\n          Loss: {:.3f}\n          Accuracy: {:.3f}'.format(accr[0],accr[1]))
```

Test set

Loss: 0.262

Accuracy: 0.977