

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



Mounted at /content/drive

```
dataset_location = "/content/drive/MyDrive/spam.csv"
```

```
import pandas as pd
import nltk
import re
import numpy as np
from nltk.corpus import stopwords
from nltk.stem.porter import PorterStemmer
from nltk.translate.ribes_score import word_rank_alignment
from numpy.lib.shape_base import split
from sklearn import preprocessing
from sklearn.feature_extraction.text import CountVectorizer
from tensorflow.keras.models import Sequential
from sklearn.model_selection import train_test_split
from keras.layers import LSTM,Dense,Dropout,Input,Embedding,Activation,Flatten
from keras.models import Model
import nltk
```

```
data = pd.read_csv(dataset_location,encoding = "ISO-8859-1")
```

```
data.drop(["Unnamed: 2","Unnamed: 3","Unnamed: 4"],axis = 1,inplace = True)
data.head()
```

	v1	v2
0	ham	Go until jurong point, crazy.. Available only ...
1	ham	Ok lar... Joking wif u oni...
2	spam	Free entry in 2 a wkly comp to win FA Cup fina...
3	ham	U dun say so early hor... U c already then say...
4	ham	Nah I don't think he goes to usf, he lives aro...

```
nltk.download('stopwords',quiet=True)
nltk.download('all',quiet=True)
```

True

```
ps = PorterStemmer()
input = []
```

```
for i in range(0,5572):
    v2 = data['v2'][i]
```

```
#removing punctuation
v2 = re.sub('[^a-zA-Z]', ' ', v2)

#converting to lower case
v2 = v2.lower()

#splitting the sentence
v2 = v2.split()

#removing the stopwords and stemming
v2 = [ps.stem(word) for word in v2 if not word in set(stopwords.words('english'))]

v2 = ' '.join(v2)

input.append(v2)


#creating document term matrix
cv = CountVectorizer(max_features=2000)
x = cv.fit_transform(input).toarray()
x.shape

(5572, 2000)

le = preprocessing.LabelEncoder()

data['v1'] = le.fit_transform(data['v1'])
data['v1'].unique()

array([0, 1])

y = data['v1'].values

y = y.reshape(-1,1)

x_train,x_test,y_train,y_test = train_test_split(x,y,test_size = 0.4)

model = Sequential()

model.add(Dense(1565,activation = "relu"))
model.add(Dense(3000,activation = "relu"))
model.add(Dense(1,activation = "sigmoid"))
model.add(Flatten())

model.compile(optimizer = "adam",loss = "binary_crossentropy", metrics = ["accuracy"])

model.fit(x_train,y_train,epochs = 15)
```

```

Epoch 1/15
105/105 [=====] - 10s 84ms/step - loss: 0.1495 - accuracy: 0
Epoch 2/15
105/105 [=====] - 9s 84ms/step - loss: 0.0131 - accuracy: 0
Epoch 3/15
105/105 [=====] - 8s 78ms/step - loss: 0.0039 - accuracy: 0
Epoch 4/15
105/105 [=====] - 8s 78ms/step - loss: 0.0030 - accuracy: 0
Epoch 5/15
105/105 [=====] - 8s 78ms/step - loss: 0.0027 - accuracy: 0
Epoch 6/15
105/105 [=====] - 8s 78ms/step - loss: 0.0026 - accuracy: 0
Epoch 7/15
105/105 [=====] - 8s 78ms/step - loss: 0.0026 - accuracy: 0
Epoch 8/15
105/105 [=====] - 8s 77ms/step - loss: 0.0026 - accuracy: 0
Epoch 9/15
105/105 [=====] - 8s 78ms/step - loss: 0.0024 - accuracy: 0
Epoch 10/15
105/105 [=====] - 9s 83ms/step - loss: 0.0023 - accuracy: 0
Epoch 11/15
105/105 [=====] - 11s 110ms/step - loss: 0.0024 - accuracy: 0
Epoch 12/15
105/105 [=====] - 9s 85ms/step - loss: 0.0024 - accuracy: 0
Epoch 13/15
105/105 [=====] - 8s 79ms/step - loss: 0.0024 - accuracy: 0
Epoch 14/15
105/105 [=====] - 8s 78ms/step - loss: 0.0022 - accuracy: 0
Epoch 15/15
105/105 [=====] - 8s 78ms/step - loss: 0.0022 - accuracy: 0
<keras.callbacks.History at 0x7f93d185b950>

```

```
model.save("spam-message-classifier.h5")
```

```

ham = "im donee. come pick me up"
spam = "WINNER$$$$ SMS REPLY 'WIN'"
message = re.sub('[^a-zA-Z]', ' ', spam)
message

```

```
'WINNER      SMS REPLY  WIN '
```

```

message = message.split()
message = [ps.stem(word) for word in message if not word in set(stopwords.words('english'))]
message = ' '.join(message)

```

```

message1 = cv.transform([message])
message1

```

```

<1x2000 sparse matrix of type '<class 'numpy.int64'>'
with 4 stored elements in Compressed Sparse Row format>

```

```
TruePredction = model.predict(message1.astype(float))
```

1/1 [=====] - 0s 183ms/step

TruePredction > 0.5

array([[True]])

```
msg = re.sub('[^a-zA-Z]', ' ', ham)
msg
```

'im donee come pick me up'

```
msg = msg.split()
msg = [ps.stem(word) for word in msg if not word in set(stopwords.words('english'))]
msg = ' '.join(msg)
```

msg

'im done come pick'

cv.transform([msg])

<1x2000 sparse matrix of type '<class 'numpy.int64'>'
with 4 stored elements in Compressed Sparse Row format>

FalsePredction = model.predict(cv.transform([msg]))

1/1 [=====] - 0s 100ms/step

FalsePredction > 0.5

array([[False]])

[Colab paid products](#) - [Cancel contracts here](#)

