

```

import pandas as pd
import numpy as np
import matplotlib.pyplot as plt

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 Adam
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

```

```

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)

```

```

from wordcloud import WordCloud, STOPWORDS, ImageColorGenerator

```

```

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=0.2)

```

```

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)

```

```

inputs = Input(shape=[max_len])
layer = Embedding(max_words,50,input_length=max_len)(inputs)
layer = LSTM(128)(layer)
layer = Dense(128)(layer)
layer = Activation('relu')(layer)
layer = Dropout(0.5)(layer)
layer = Dense(1)(layer)
layer = Activation('sigmoid')(layer)
model = Model(inputs=inputs,outputs=layer)

```

```
model.summary()
```

Model: "model"

Layer (type)	Output Shape	Param #
=====		
input_1 (InputLayer)	[(None, 150)]	0
embedding (Embedding)	(None, 150, 50)	50000
lstm (LSTM)	(None, 128)	91648
dense (Dense)	(None, 128)	16512
activation (Activation)	(None, 128)	0
dropout (Dropout)	(None, 128)	0
dense_1 (Dense)	(None, 1)	129
activation_1 (Activation)	(None, 1)	0

```

=====
Total params: 158,289
Trainable params: 158,289
Non-trainable params: 0

```

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

```
history = model.fit(sequences_matrix,Y_train,batch_size=32,epochs=10,
                    validation_split=0.2)
```

```

Epoch 1/10
112/112 [=====] - 31s 248ms/step - loss: 0.2155 - accuracy:
Epoch 2/10
112/112 [=====] - 26s 237ms/step - loss: 0.0468 - accuracy:
Epoch 3/10
112/112 [=====] - 26s 236ms/step - loss: 0.0296 - accuracy:
Epoch 4/10
112/112 [=====] - 27s 243ms/step - loss: 0.0150 - accuracy:
Epoch 5/10
112/112 [=====] - 27s 238ms/step - loss: 0.0083 - accuracy:
Epoch 6/10
112/112 [=====] - 27s 238ms/step - loss: 0.0056 - accuracy:
Epoch 7/10
112/112 [=====] - 27s 238ms/step - loss: 0.0079 - accuracy:

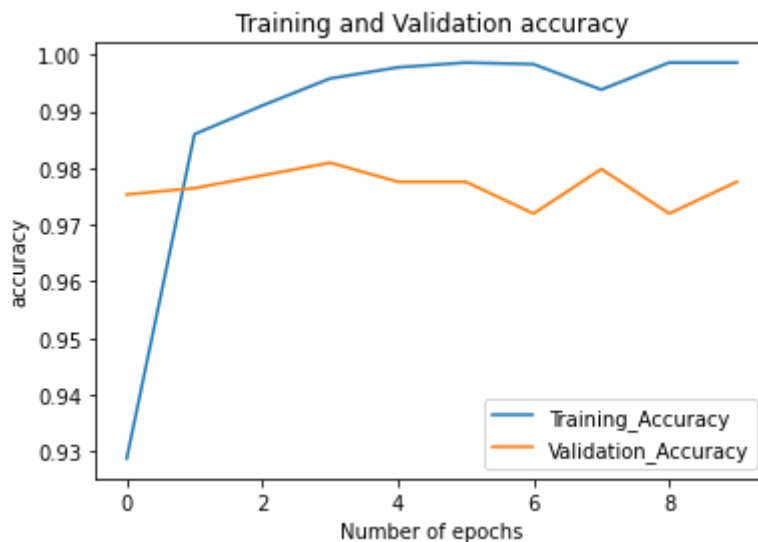
```

```
Epoch 8/10
112/112 [=====] - 26s 236ms/step - loss: 0.0167 - accuracy:
Epoch 9/10
112/112 [=====] - 27s 237ms/step - loss: 0.0055 - accuracy:
Epoch 10/10
112/112 [=====] - 27s 237ms/step - loss: 0.0046 - accuracy:
```



```
metrics = pd.DataFrame(history.history)
metrics.rename(columns = {'loss': 'Training_Loss', 'accuracy': 'Training_Accuracy', 'val_1
def plot_graphs1(var1, var2, string):
    metrics[[var1, var2]].plot()
    plt.title('Training and Validation ' + string)
    plt.xlabel ('Number of epochs')
    plt.ylabel(string)
    plt.legend([var1, var2])
```

```
plot_graphs1('Training_Accuracy', 'Validation_Accuracy', 'accuracy')
```



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

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

```
accuracy1 = model.evaluate(test_sequences_matrix,Y_test)
```

```
35/35 [=====] - 3s 85ms/step - loss: 0.0838 - accuracy: 0.9
```



```
print(' Accuracy: {:.3f}'.format(accuracy1[0],accuracy1[1]))
```

Accuracy: 0.084

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

✓ 0s completed at 8:08 PM

