1) Download and upload the data set into colab

```
!unzip '/content/archive.zip'
```

```
Archive:  /content/archive.zip
  inflating: spam.csv
```

2)Import the required library

```
import numpy as np
import pandas as pd
import nltk
import re
nltk.download('stopwords')
from nltk.corpus import stopwords
from nltk.stem.porter import PorterStemmer
from sklearn.model_selection import train_test_split
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, LSTM
from keras.layers import Embedding
from keras.preprocessing.text import Tokenizer
from keras.preprocessing import sequence
from keras_preprocessing.sequence import pad_sequences
```

```
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Unzipping corpora/stopwords.zip.
```

3) Read Data set and do pre processing

```
df = pd.read_csv('/content/spam.csv', encoding="ISO-8859-1")
df
```

	v1	v2	Unnamed: 2	Unnamed: 3	Unname
0	ham	Go until jurong point, crazy.. Available only ...	NaN	NaN	
1	ham	Ok lar... Joking wif u oni...	NaN	NaN	

```
data = df[['v1', 'v2']]
data
```

	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...
...	...	...
5567	spam	This is the 2nd time we have tried 2 contact u...
5568	ham	Will Ì_ b going to esplanade fr home?
5569	ham	Pity, * was in mood for that. So...any other s...
5570	ham	The guy did some bitching but I acted like i'd...
5571	ham	Rofl. Its true to its name

5572 rows × 2 columns

```
ps = PorterStemmer()

for i in range(0, 5572):
  review = data['v2'][i]
  review = re.sub('[^a-zA-Z]',' ', review)
  review = review.lower()
  review = review.split()
  review = [ps.stem(word) for word in review if word not in set(stopwords.words('english')
  review = ' '.join(review)
  data['v2'][i] = review
```

```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:10: SettingWithCopyWarn
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/u
  # Remove the CWD from sys.path while we load stuff.
```

```
data
```

	v1	v2
0	ham	go jurong point crazi avail bugi n great world...
1	ham	ok lar joke wif u oni
2	spam	free entri wkli comp win fa cup final tkt st m...
3	ham	u dun say earli hor u c alreadi say
4	ham	nah think goe usf live around though
...	...	...
5567	spam	nd time tri contact u u pound prize claim easi...
5568	ham	b go esplanad fr home
5569	ham	piti mood suggest
5570	ham	guy bitch act like interest buy someth els nex...

```
Max = 50000
Max_seq = 250
emb = 100

tokenizer = Tokenizer(num_words = Max)
tokenizer.fit_on_texts(data['v2'].values)
word_index = tokenizer.word_index

x = tokenizer.texts_to_sequences(data['v2'].values)
x = pad_sequences(x, maxlen = Max_seq)

y = pd.get_dummies(data['v1']).values

print(x.shape, y.shape)
```

```
(5572, 250) (5572, 2)
```

```
xtrain,xtest,ytrain,ytest=train_test_split(x,y)
print(xtrain.shape, ytrain.shape)
print(xtest.shape, ytest.shape)
```

```
(4179, 250) (4179, 2)
(1393, 250) (1393, 2)
```

```
xtrain.reshape(4179, 250, 1)
ytrain.reshape(4179, 2, 1)
xtest.reshape(1393, 250, 1)
ytest.reshape(1393, 2, 1)
```

```
array([[[1],
        [0]],

       [[1],
        [0]],
```

```
       [[1],
        [0]],

       ...,

       [[1],
        [0]],

       [[1],
        [0]],

       [[1],
        [0]]], dtype=uint8)
```

4) Create model

```
model = Sequential()
```

5) Add Layers

```
model.add(Embedding(Max, emb, input_length = x.shape[1]))
model.add(LSTM(100))
model.add(Dense(2, activation = 'relu'))
```

6) Compile model

```
model.compile(optimizer='adam',loss='mse',metrics = ['accuracy'])
```

```
model.summary()
```

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 embedding (Embedding)       (None, 250, 100)          5000000

 lstm (LSTM)                 (None, 100)               80400

 dense (Dense)               (None, 2)                 202

=================================================================
Total params: 5,080,602
Trainable params: 5,080,602
Non-trainable params: 0
_________________________________________________________________
```

7) Fit the model

```
model.fit(xtrain,ytrain,epochs=10)
```

```
Epoch 1/10
131/131 [==============================] - 29s 202ms/step - loss: 0.0761 - accuracy:
Epoch 2/10
131/131 [==============================] - 28s 218ms/step - loss: 0.0100 - accuracy:
Epoch 3/10
131/131 [==============================] - 27s 203ms/step - loss: 0.0035 - accuracy:
Epoch 4/10
131/131 [==============================] - 27s 209ms/step - loss: 0.0020 - accuracy:
Epoch 5/10
131/131 [==============================] - 27s 203ms/step - loss: 0.0013 - accuracy:
Epoch 6/10
131/131 [==============================] - 27s 204ms/step - loss: 0.0010 - accuracy:
Epoch 7/10
131/131 [==============================] - 26s 201ms/step - loss: 8.9223e-04 - accur
Epoch 8/10
131/131 [==============================] - 27s 209ms/step - loss: 8.0955e-04 - accur
Epoch 9/10
131/131 [==============================] - 27s 206ms/step - loss: 6.0584e-04 - accur
Epoch 10/10
131/131 [==============================] - 27s 205ms/step - loss: 7.5646e-04 - accur
<keras.callbacks.History at 0x7fb3c2cba450>
```

8) Save the model

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

9) Test the model

```
op = ['ham', 'spam']

def text_processing(text):
  review = re.sub('[^a-zA-Z]',' ', text)
  review = review.lower()
  review = review.split()
  review = [ps.stem(word) for word in review if word not in set(stopwords.words('english')
  review = ' '.join(review)
  return review


# Testing 1
text =  '''Dear candidate,
           Your otp number is 09478'''

text = text_processing(text)
seq = tokenizer.texts_to_sequences([text])
padded = pad_sequences(seq, maxlen = Max_seq)
pred = model.predict(padded)

print(pred, op[np.argmax(pred)])
```

```
1/1 [==============================] - 1s 512ms/step
[[1.0094543 0.       ]] ham
```

```
# Testing 2
text =  '''claim money 50000 for free and enjoy lexury life'''

text = text_processing(text)
seq = tokenizer.texts_to_sequences([text])
padded = pad_sequences(seq, maxlen = Max_seq)
pred = model.predict(padded)

print(pred, op[np.argmax(pred)])
```

```
1/1 [==============================] - 0s 30ms/step
[[0.3601427  0.62205034]] spam
```

```
# Testing 3

text =  '''Check alert!!,
    You have won cash prize.
    steal it away'''

text = text_processing(text)
seq = tokenizer.texts_to_sequences([text])
padded = pad_sequences(seq, maxlen = Max_seq)
pred = model.predict(padded)

print(pred, op[np.argmax(pred)])
```

```
1/1 [==============================] - 0s 29ms/step
[[0.40358758 0.67646027]] spam
```

```
# Testing 4

text =  '''Really do hope the work doesnt get stressful. Have a gr8 day.'''

text = text_processing(text)
seq = tokenizer.texts_to_sequences([text])
padded = pad_sequences(seq, maxlen = Max_seq)
pred = model.predict(padded)

print(pred, op[np.argmax(pred)])
```

```
1/1 [==============================] - 0s 25ms/step
[[1.0080519 0.       ]] ham
```

Colab paid products - Cancel contracts here

✓ 0s completed at 10:51 PM