Team Leader: Ameena Begum

Team members: Hemalatha
Saran
Sutharsan

#Import required 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
%matplotlib inline
```

Read dataset and do pre-processing

```
!pip install -q kaggle

!mkdir ~/.kaggle

!cp kaggle.json ~/.kaggle/

! chmod 600 ~/.kaggle/kaggle.json

! kaggle datasets download -d uciml/sms-spam-collection-dataset

Downloading sms-spam-collection-dataset.zip to /content
   0% 0.00/211k [00:00<?, ?B/s]
 100% 211k/211k [00:00<00:00, 43.1MB/s]

!unzip sms-spam-collection-dataset.zip

Archive:  sms-spam-collection-dataset.zip
inflating: spam.csv

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

```
     v1                                                 v2 Unnamed: 2  \
0   ham  Go until jurong point, crazy.. Available only ...        NaN

1   ham                      Ok lar... Joking wif u oni...        NaN

2   spam  Free entry in 2 a wkly comp to win FA Cup fina...        NaN

3   ham  U dun say so early hor... U c already then say...        NaN
    4   ham  Nah I don't think he goes to usf, he lives aro...
    NaN


  Unnamed: 3 Unnamed: 4
0        NaN        NaN
1        NaN        NaN
2        NaN        NaN
3        NaN        NaN  4         NaN        NaN
```

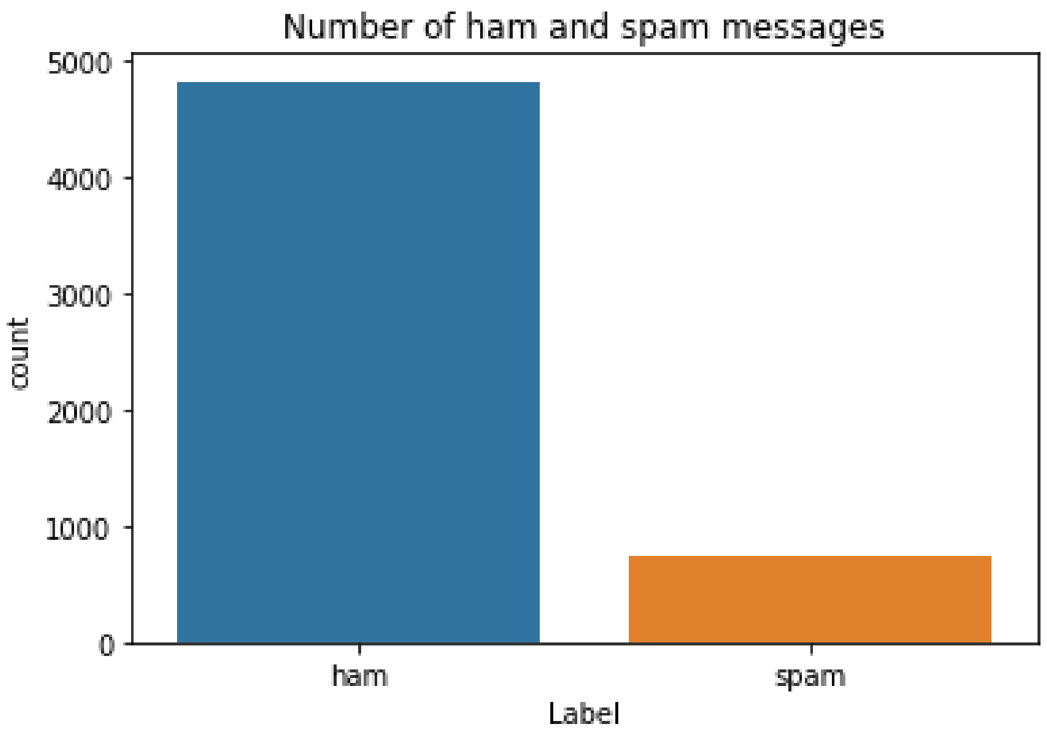
```
df.drop(['Unnamed: 2', 'Unnamed: 3', 'Unnamed:
4'],axis=1,inplace=True)
df.info()
```

```
<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
```

```
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 variable as a keyword arg: x. From
version 0.12, the only valid positional argument will be `data`, and
passing other arguments without an explicit keyword will result in an
error or misinterpretation.   FutureWarning
```

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



```
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)
```

Create Model

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

```
defRNN():
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)
return model
```

Compile the model.

```
model = RNN()
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)
```

```
Epoch 1/10
30/30 [==============================] - 12s 300ms/step - loss: 0.3153
- accuracy: 0.8807 - val_loss: 0.1207 - val_accuracy: 0.9631
Epoch 2/10
30/30 [==============================] - 8s 274ms/step - loss: 0.0783
- accuracy: 0.9797 - val_loss: 0.1235 - val_accuracy: 0.9694
Epoch 3/10
30/30 [==============================] - 9s 304ms/step - loss: 0.0448
- accuracy: 0.9865 - val_loss: 0.0575 - val_accuracy: 0.9842
Epoch 4/10
30/30 [==============================] - 8s 275ms/step - loss: 0.0315
- accuracy: 0.9905 - val_loss: 0.0651 - val_accuracy: 0.9800
Epoch 5/10
```

```
30/30 [==============================] - 8s 276ms/step - loss: 0.0264
- accuracy: 0.9926 - val_loss: 0.0637 - val_accuracy: 0.9842
Epoch 6/10
30/30 [==============================] - 8s 274ms/step - loss: 0.0163
- accuracy: 0.9958 - val_loss: 0.0745 - val_accuracy: 0.9789
Epoch 7/10
30/30 [==============================] - 10s 325ms/step - loss: 0.0135
- accuracy: 0.9960 - val_loss: 0.0807 - val_accuracy: 0.9821
Epoch 8/10
30/30 [==============================] - 8s 277ms/step - loss: 0.0400
- accuracy: 0.9905 - val_loss: 0.1125 - val_accuracy: 0.9800
Epoch 9/10
30/30 [==============================] - 8s 275ms/step - loss: 0.0464
- accuracy: 0.9897 - val_loss: 0.0722 - val_accuracy: 0.9905
Epoch 10/10
30/30 [==============================] - 8s 276ms/step - loss: 0.0081
- accuracy: 0.9982 - val_loss: 0.0725 - val_accuracy: 0.9895
```

```
<keras.callbacks.History at 0x7f5451bc3d10>
```

Save the Model

```
model.save('Trained Model')
```

```
WARNING:absl:Found untraced functions such as lstm_cell_layer_call_fn,
lstm_cell_layer_call_and_return_conditional_losses while saving
(showing 2 of 2). These functions will not be directly callable after
loading.
```

Test Model

```
from keras.models import load_model
model = load_model("Trained Model")
```

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

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

```
27/27 [==============================] - 1s 25ms/step - loss: 0.0413
accuracy: 0.9892
```

```
print('Test set\n  Loss: {:0.3f}\n  Accuracy:
{:0.3f}'.format(accr[0],accr[1]))
Test set
  Loss: 0.041
  Accuracy: 0.989
```