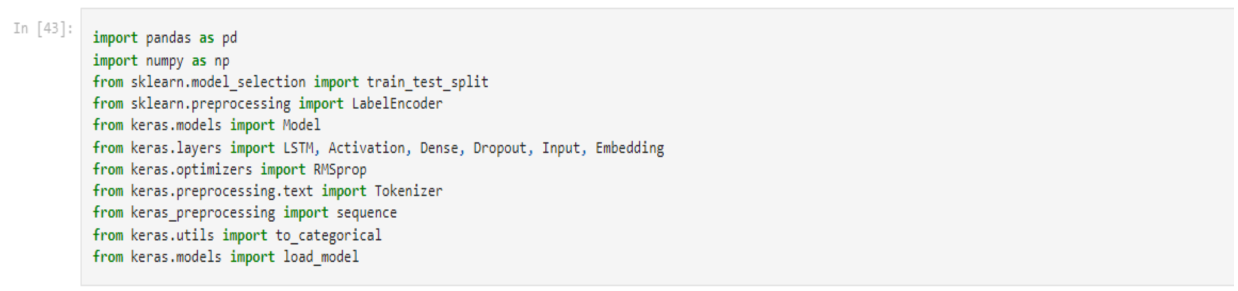
IBM NALAIYA THIRAN

Assignment -4

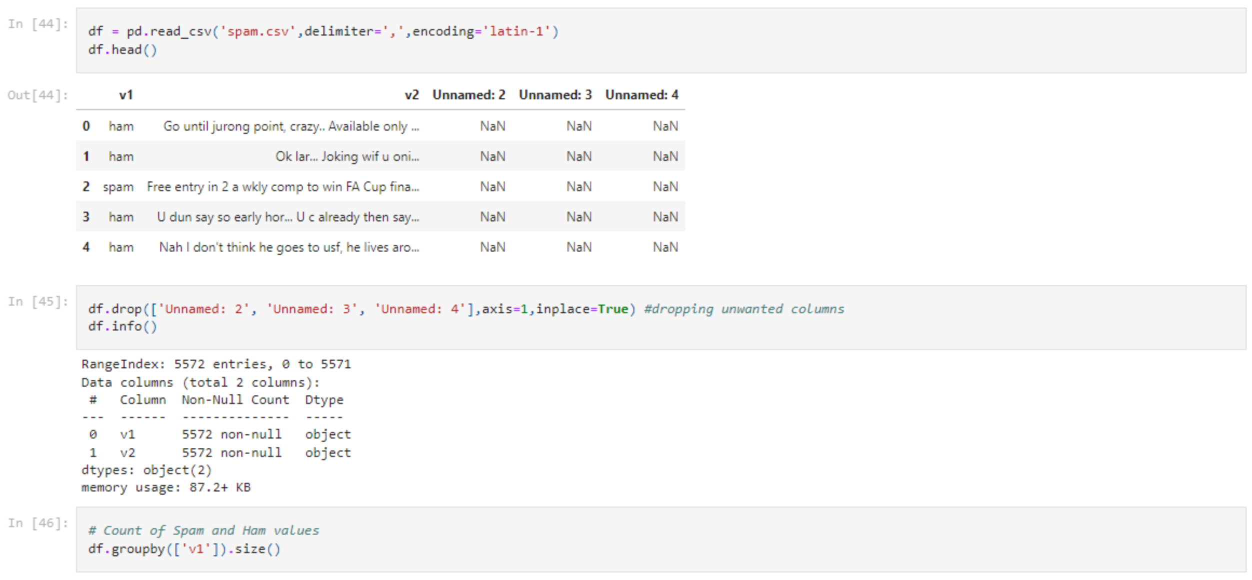
Team ID	PNT2022TMID33620
Project Name	AI based discourse for Banking Industry
Student Name	Siva R
Student Roll Number	922519106150
Maximum Marks	2 Marks

Import required library:



```
In [43]:
import pandas as pd
import numpy as np
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 to_categorical
from keras.models import load_model
```

Read Dataset and do pre-processing:



```
In [44]:
df = pd.read_csv('spam.csv',delimiter=',',encoding='latin-1')
df.head()
```

Out[44]:

	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

```
In [45]:
df.drop(['Unnamed: 2', 'Unnamed: 3', 'Unnamed: 4'],axis=1,inplace=True) #dropping unwanted columns
df.info()
```

```
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
```

```
In [46]:
# Count of Spam and Ham values
df.groupby(['v1']).size()
```

```
Out[46]: v1
         ham     4825
         spam     747
         dtype: int64
```

```
In [47]:
# Label Encoding target column
X = df.v2
Y = df.v1
le = LabelEncoder()
Y = le.fit_transform(Y)
Y = Y.reshape(-1,1)
```

```
In [48]:
# Test and train split
X_train,X_test,Y_train,Y_test = train_test_split(X,Y,test_size=0.15)
```

```
In [49]:
# Tokenisation function
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 = sequence.pad_sequences(sequences,maxlen=max_len)
```

Create Model and Add Layers (LSTM, Dense- (Hidden Layers), Output):

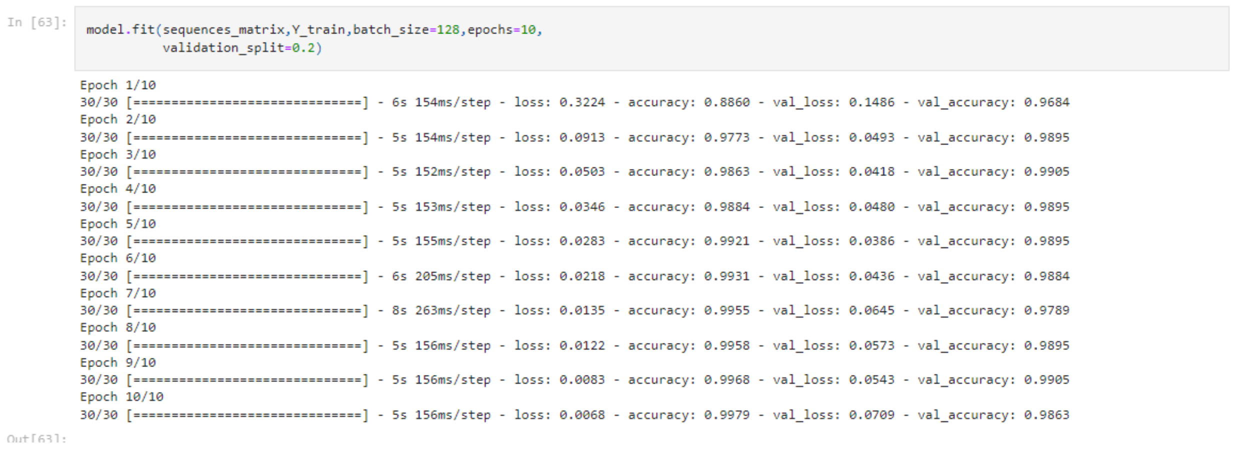
```
In [58]:
# Creating LSTM model
inputs = Input(name='InputLayer',shape=[max_len])
layer = Embedding(max_words,50,input_length=max_len)(inputs)
layer = LSTM(64)(layer)
layer = Dense(256,name='FullyConnectedLayer1')(layer)
layer = Activation('relu')(layer)
layer = Dropout(0.5)(layer)
layer = Dense(1,name='OutputLayer')(layer)
layer = Activation('sigmoid')(layer)
```

Compile the model:

```
In [59]:
model = Model(inputs=inputs,outputs=layer)
model.summary()
model.compile(loss='binary_crossentropy',optimizer=RMSprop(),metrics=['accuracy'])
```

```
Model: "model_2"
_________________________________________________________________
 Layer (type)                Output Shape              Param #   
=================================================================
 InputLayer (InputLayer)     [(None, 150)]             0         
                                                                 
 embedding_5 (Embedding)     (None, 150, 50)           50000     
                                                                 
 lstm_5 (LSTM)               (None, 64)                29440     
                                                                 
 FullyConnectedLayer1 (Dense  (None, 256)              16640     
 )                                                               
                                                                 
 activation_5 (Activation)   (None, 256)               0         
                                                                 
 dropout_3 (Dropout)         (None, 256)               0         
                                                                 
 OutputLayer (Dense)         (None, 1)                 257       
                                                                 
 activation_6 (Activation)   (None, 1)                 0         
                                                                 
=================================================================
Total params: 96,337
Trainable params: 96,337
Non-trainable params: 0
```

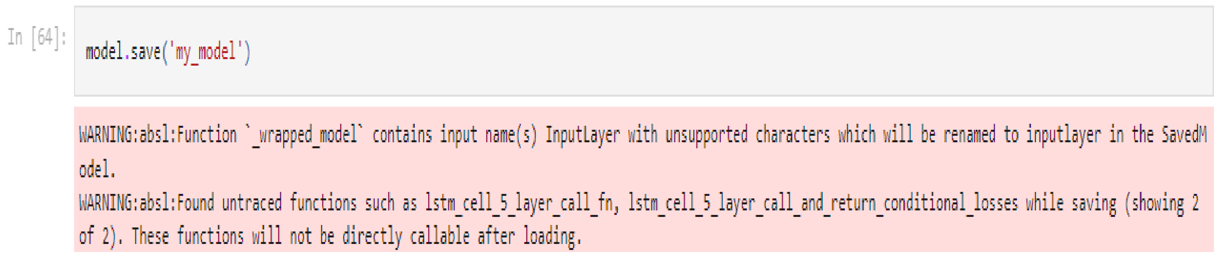
Fit the Model:



```
In [63]: model.fit(sequences_matrix,Y_train,batch_size=128,epochs=10,
                   validation_split=0.2)

Epoch 1/10
30/30 [==============================] - 6s 154ms/step - loss: 0.3224 - accuracy: 0.8860 - val_loss: 0.1486 - val_accuracy: 0.9684
Epoch 2/10
30/30 [==============================] - 5s 154ms/step - loss: 0.0913 - accuracy: 0.9773 - val_loss: 0.0493 - val_accuracy: 0.9895
Epoch 3/10
30/30 [==============================] - 5s 152ms/step - loss: 0.0503 - accuracy: 0.9863 - val_loss: 0.0418 - val_accuracy: 0.9905
Epoch 4/10
30/30 [==============================] - 5s 153ms/step - loss: 0.0346 - accuracy: 0.9884 - val_loss: 0.0480 - val_accuracy: 0.9895
Epoch 5/10
30/30 [==============================] - 5s 155ms/step - loss: 0.0283 - accuracy: 0.9921 - val_loss: 0.0386 - val_accuracy: 0.9895
Epoch 6/10
30/30 [==============================] - 6s 205ms/step - loss: 0.0218 - accuracy: 0.9931 - val_loss: 0.0436 - val_accuracy: 0.9884
Epoch 7/10
30/30 [==============================] - 8s 263ms/step - loss: 0.0135 - accuracy: 0.9955 - val_loss: 0.0645 - val_accuracy: 0.9789
Epoch 8/10
30/30 [==============================] - 5s 156ms/step - loss: 0.0122 - accuracy: 0.9958 - val_loss: 0.0573 - val_accuracy: 0.9895
Epoch 9/10
30/30 [==============================] - 5s 156ms/step - loss: 0.0083 - accuracy: 0.9968 - val_loss: 0.0543 - val_accuracy: 0.9905
Epoch 10/10
30/30 [==============================] - 5s 156ms/step - loss: 0.0068 - accuracy: 0.9979 - val_loss: 0.0709 - val_accuracy: 0.9863
Out[63]:
```

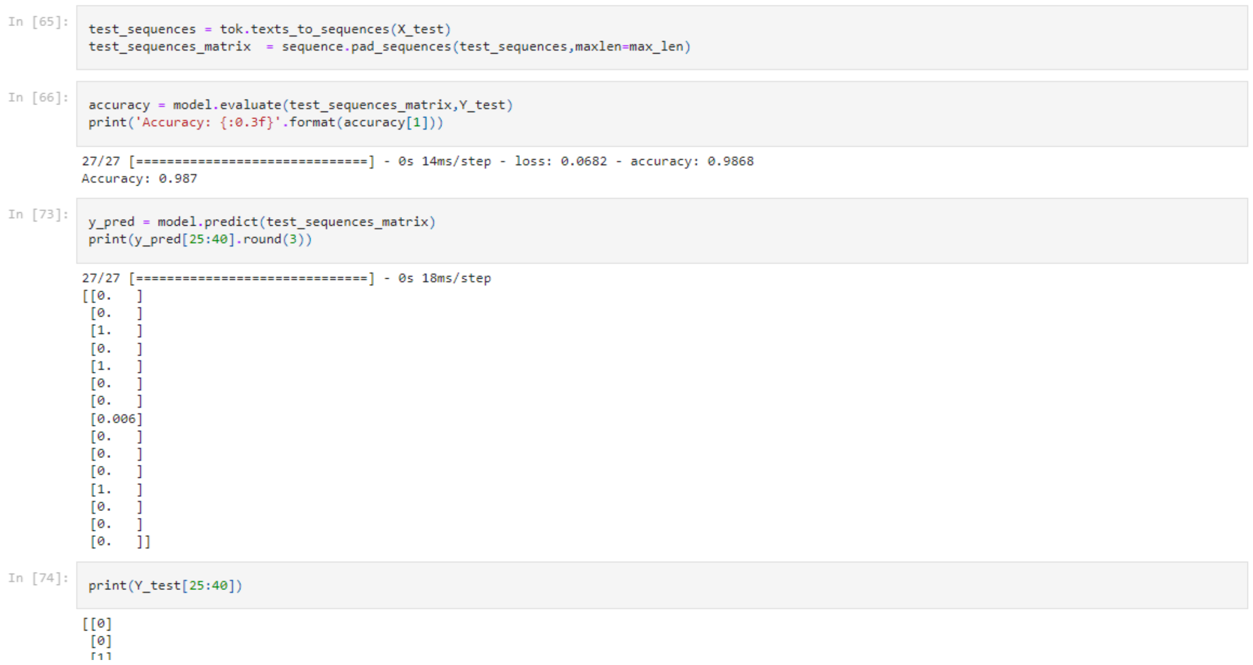
Save the Model:



```
In [64]: model.save('my_model')

WARNING:absl:Function `_wrapped_model` contains input name(s) InputLayer with unsupported characters which will be renamed to inputlayer in the SavedM
odel.
WARNING:absl:Found untraced functions such as lstm_cell_5_layer_call_fn, lstm_cell_5_layer_call_and_return_conditional_losses while saving (showing 2
of 2). These functions will not be directly callable after loading.
```

Test the model:



```
In [65]: test_sequences = tok.texts_to_sequences(X_test)
         test_sequences_matrix  = sequence.pad_sequences(test_sequences,maxlen=max_len)

In [66]: accuracy = model.evaluate(test_sequences_matrix,Y_test)
         print('Accuracy: {:0.3f}'.format(accuracy[1]))

27/27 [==============================] - 0s 14ms/step - loss: 0.0682 - accuracy: 0.9868
Accuracy: 0.987

In [73]: y_pred = model.predict(test_sequences_matrix)
         print(y_pred[25:40].round(3))

27/27 [==============================] - 0s 18ms/step
[[0.   ]
 [0.   ]
 [1.   ]
 [0.   ]
 [1.   ]
 [0.   ]
 [0.   ]
 [0.006]
 [0.   ]
 [0.   ]
 [0.   ]
 [1.   ]
 [0.   ]
 [0.   ]
 [0.   ]]

In [74]: print(Y_test[25:40])

[[0]
 [0]
 [1]
 [1.   ]
 [0.   ]
 [1.   ]
 [0.   ]
 [0.   ]
 [0.006]
 [0.   ]
 [0.   ]
 [0.   ]
 [1.   ]
 [0.   ]
 [0.   ]
 [0.   ]]
```

In [74]:

```
print(Y_test[25:40])
```

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