Assignment -3

Problem Statement :- SMS SPAM Classification

Assignment Date	27 October 2022
Student Name	S.Karthika
Student Roll Number	912419104014
Maximum Marks	2 Marks

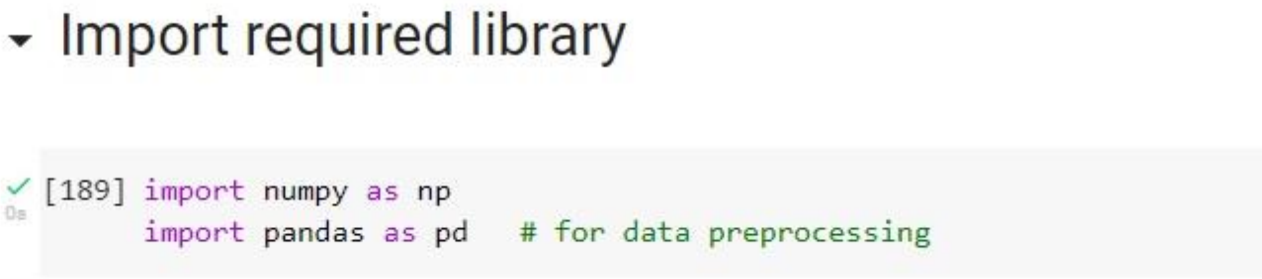
Problem Statement:

Over recent years, as the popularity of mobile phone devices has increased, Short Message Service (SMS) has grown into a multi-billion dollar industry. At the same time, reduction in the cost of messaging services has resulted in growth in unsolicited commercial advertisements (spams) being sent to mobile phones. Due to Spam SMS, Mobile service providers suffer from some sort of financial problems as well as it reduces calling time for users. Unfortunately, if the user accesses such Spam SMS they may face the problem of virus or malware. When SMS arrives at mobile it will disturb mobile user privacy and concentration. It may lead to frustration for the user. So Spam SMS is one of the major issues in the wireless communication world and it grows day by day.

Import required library

Solution:

```
import numpy as np
import pandas as pd   # for data preprocessing
```



Import required library

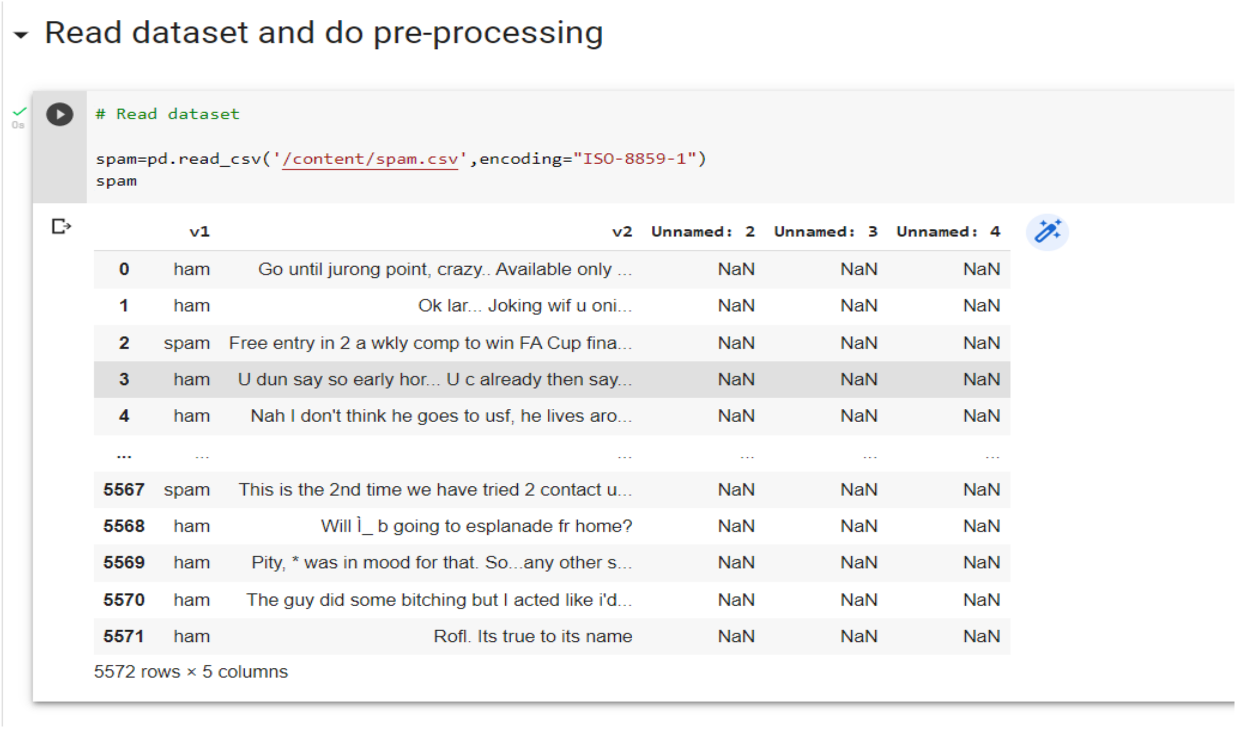
```
[189] import numpy as np
      import pandas as pd   # for data preprocessing
```

Question-1:

Read dataset and do pre-processing

Solution:

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



Read dataset and do pre-processing

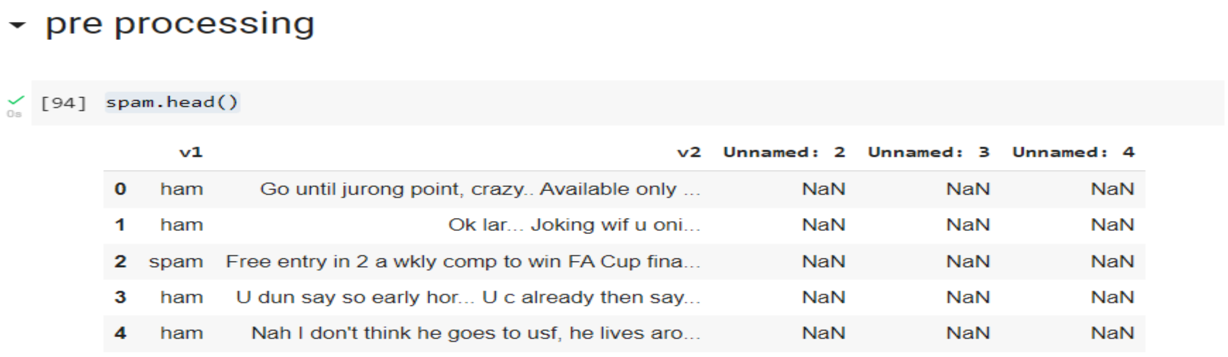
```
# Read dataset

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

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

5572 rows × 5 columns

```
# preprocessing
spam.head()
```

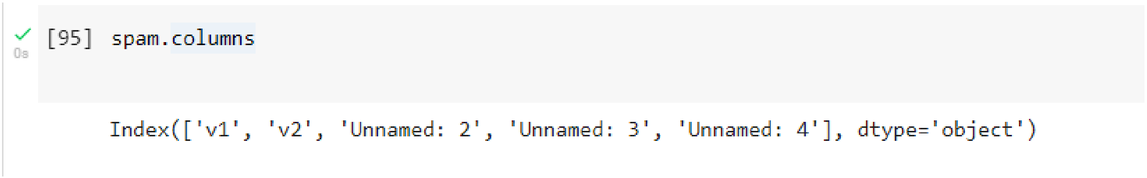


pre processing

```
[94] spam.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

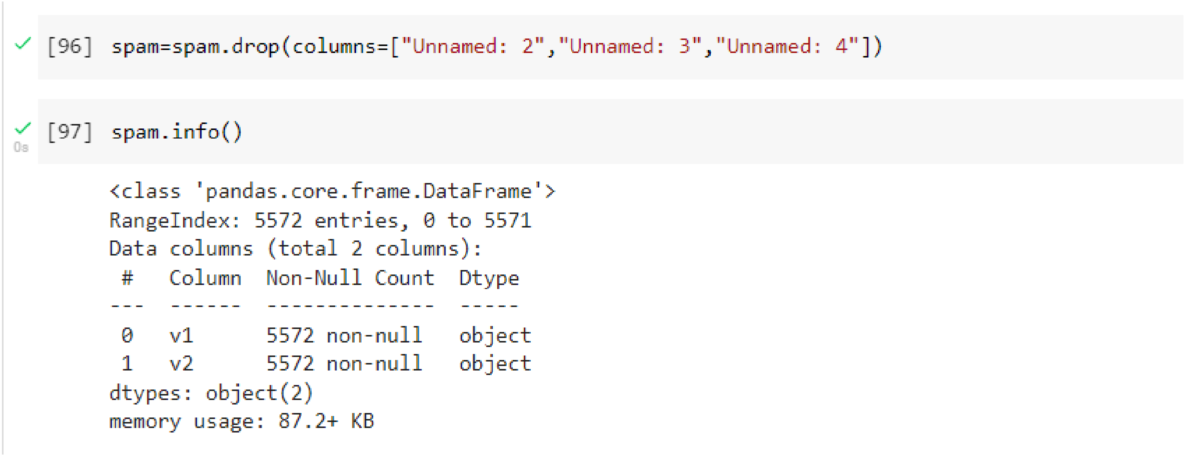
```
spam.columns
```



```
[95] spam.columns

Index(['v1', 'v2', 'Unnamed: 2', 'Unnamed: 3', 'Unnamed: 4'], dtype='object')
```

```
spam=spam.drop(columns=["Unnamed: 2","Unnamed: 3","Unnamed: 4"])
spam.info()
```



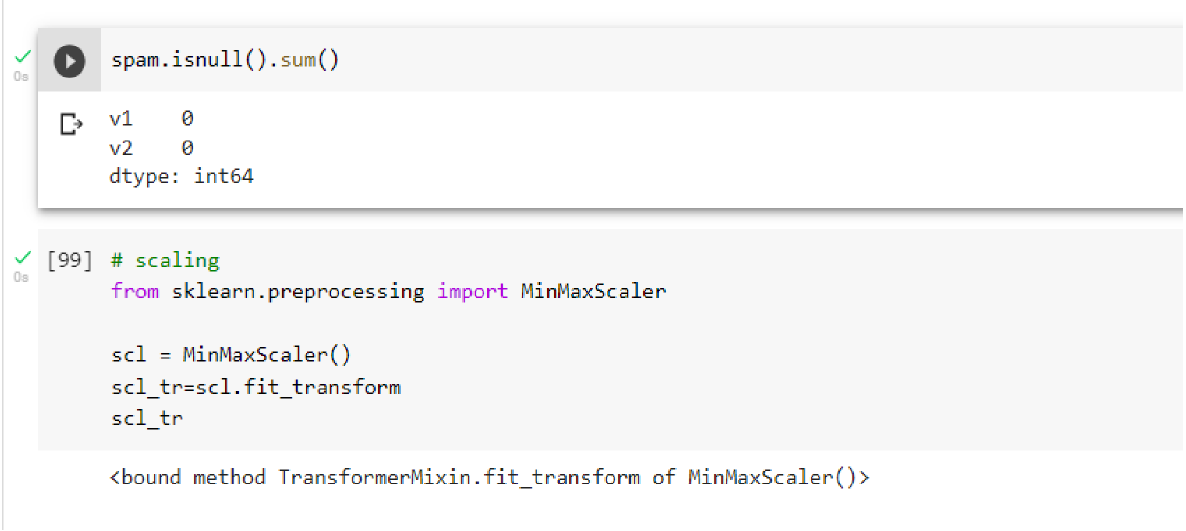
```
[96] spam=spam.drop(columns=["Unnamed: 2","Unnamed: 3","Unnamed: 4"])
```

```
[97] spam.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
```

```
spam.isnull().sum()
# scaling
from sklearn.preprocessing import MinMaxScaler

scl = MinMaxScaler()
scl_tr=scl.fit_transform
scl_tr
```



```
spam.isnull().sum()
```

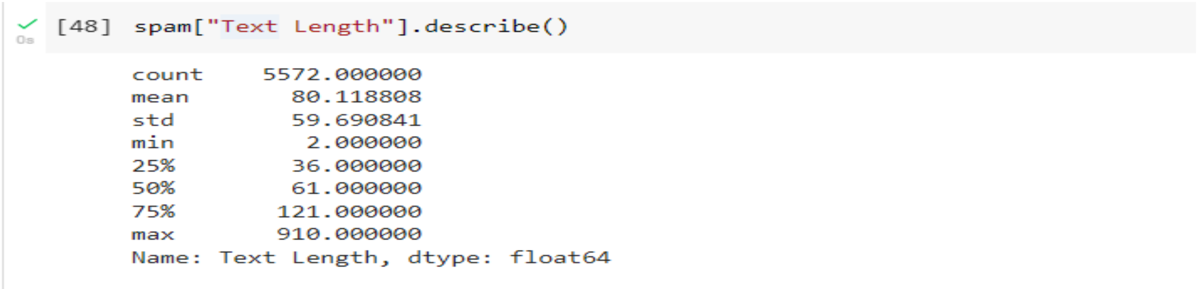
```
v1    0
v2    0
dtype: int64
```

```
[99] # scaling
     from sklearn.preprocessing import MinMaxScaler

     scl = MinMaxScaler()
     scl_tr=scl.fit_transform
     scl_tr
```

```
<bound method TransformerMixin.fit_transform of MinMaxScaler()>
```

```
spam["Text Length"].describe()
```

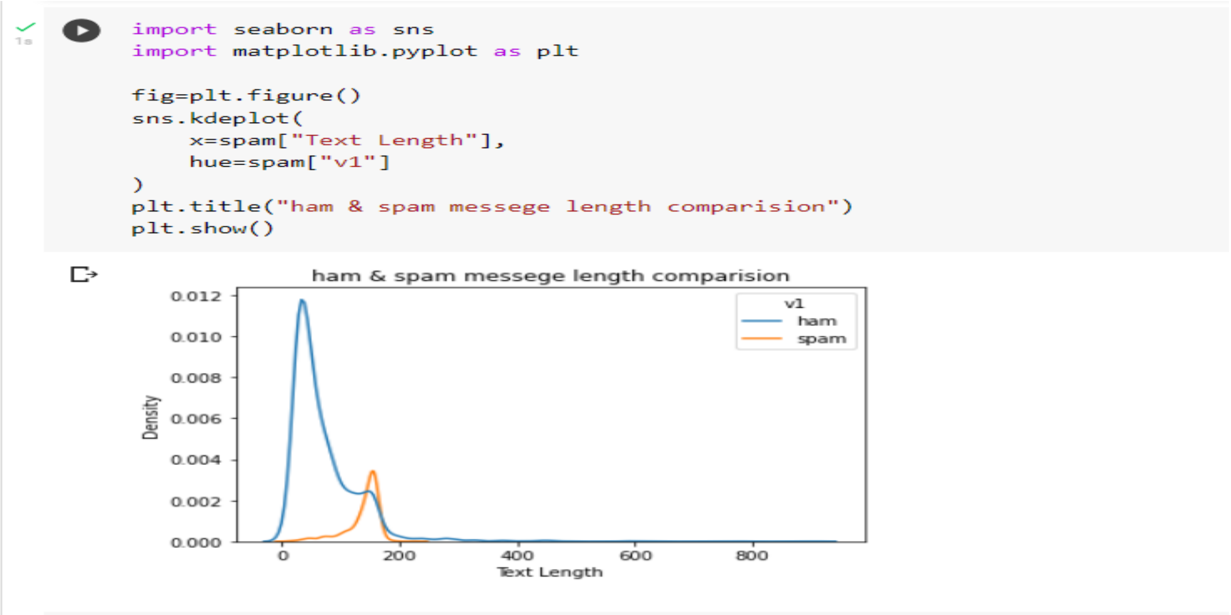


```
[48] spam["Text Length"].describe()
```

```
count    5572.000000
mean       80.118808
std        59.690841
min         2.000000
25%        36.000000
50%        61.000000
75%       121.000000
max       910.000000
Name: Text Length, dtype: float64
```

```
import seaborn as sns
import matplotlib.pyplot as plt

fig=plt.figure()
sns.kdeplot(
 x=spam["Text Length"],
hue=spam["v1"]
)
plt.title("ham & spam messege length comparision")
plt.show()
```



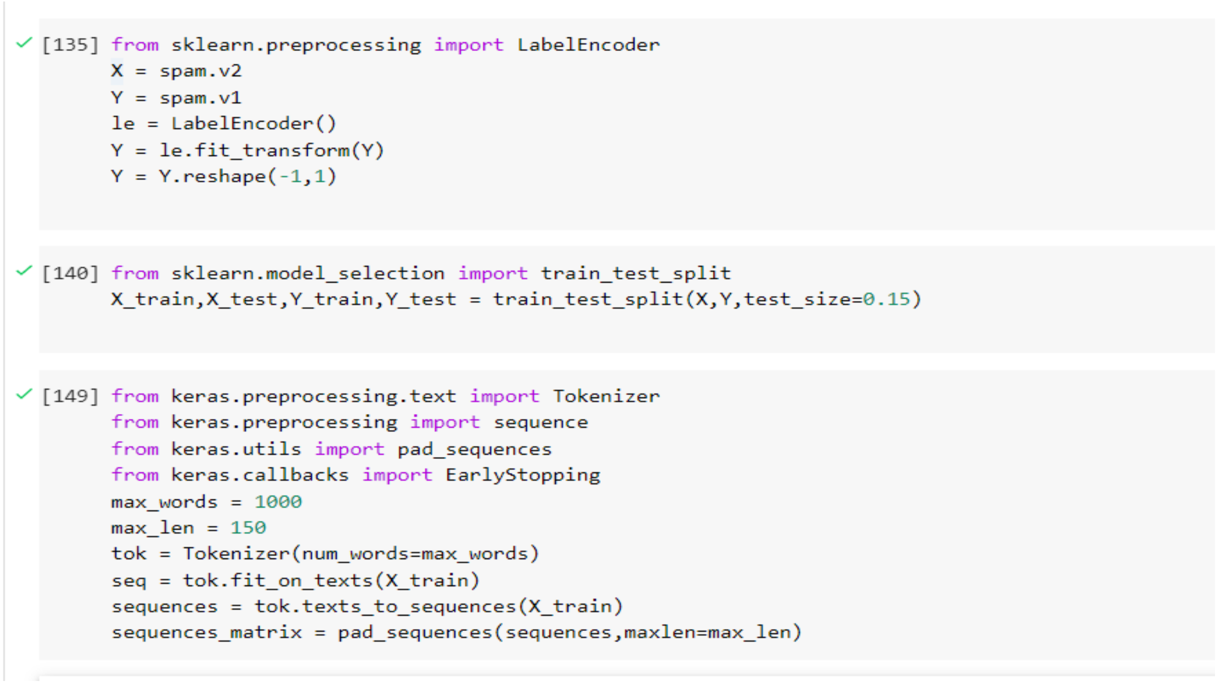
```
import seaborn as sns
import matplotlib.pyplot as plt

fig=plt.figure()
sns.kdeplot(
    x=spam["Text Length"],
    hue=spam["v1"]
)
plt.title("ham & spam messege length comparision")
plt.show()
```

```
from sklearn.preprocessing import LabelEncoder
X = spam.v2
Y = spam.v1
le = LabelEncoder()
Y = le.fit_transform(Y)
Y = Y.reshape(-1,1)

from sklearn.model_selection import train_test_split
X_train,X_test,Y_train,Y_test = train_test_split(X,Y,test_size=0.15)

from keras.preprocessing.text import Tokenizer
from keras.preprocessing import sequence
from keras.utils import pad_sequences
from keras.callbacks import EarlyStopping
max_words = 1000
max_len = 150
tok = Tokenizer(num_words=max_words)
seq = tok.fit_on_texts(X_train)
sequences = tok.texts_to_sequences(X_train)
sequences_matrix = pad_sequences(sequences,maxlen=max_len)
```



```
✓ [135] from sklearn.preprocessing import LabelEncoder
        X = spam.v2
        Y = spam.v1
        le = LabelEncoder()
        Y = le.fit_transform(Y)
        Y = Y.reshape(-1,1)
```

```
✓ [140] from sklearn.model_selection import train_test_split
        X_train,X_test,Y_train,Y_test = train_test_split(X,Y,test_size=0.15)
```

```
✓ [149] from keras.preprocessing.text import Tokenizer
        from keras.preprocessing import sequence
        from keras.utils import pad_sequences
        from keras.callbacks import EarlyStopping
        max_words = 1000
        max_len = 150
        tok = Tokenizer(num_words=max_words)
        seq = tok.fit_on_texts(X_train)
        sequences = tok.texts_to_sequences(X_train)
        sequences_matrix = pad_sequences(sequences,maxlen=max_len)
```

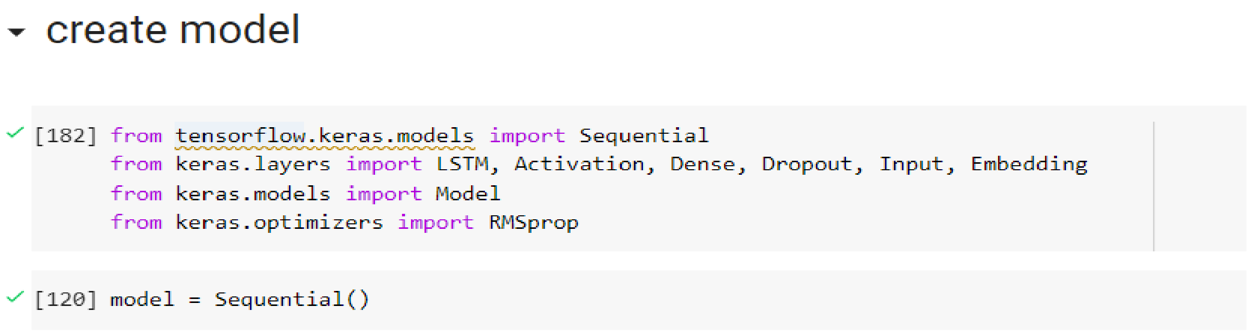
Question-2:

Create Model

Solution:

from tensorflow.keras.models import Sequential
from keras.layers import LSTM, Activation, Dense, Dropout, Input, Embeddir
from keras.models import Model
from keras.optimizers import RMSprop

model = Sequential()



▾ create model

```
✓ [182] from tensorflow.keras.models import Sequential
        from keras.layers import LSTM, Activation, Dense, Dropout, Input, Embedding
        from keras.models import Model
        from keras.optimizers import RMSprop
```

```
✓ [120] model = Sequential()
```

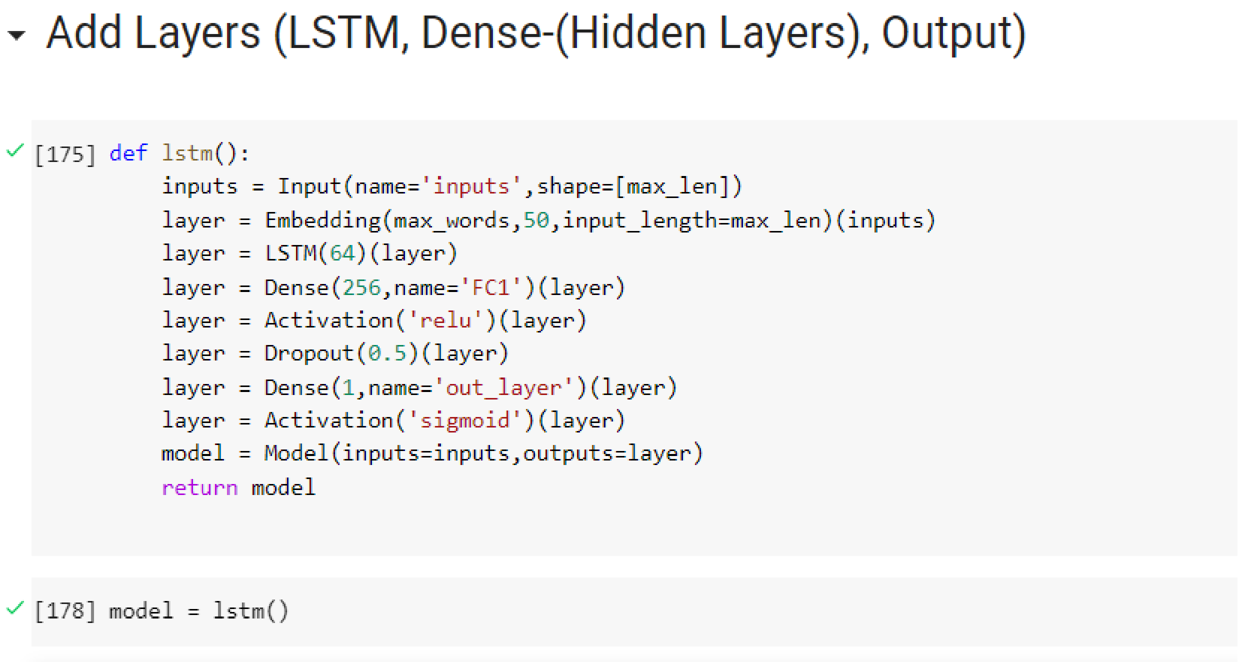
Question-3:

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

Solution:

```
def lstm():
    inputs = Input(name='inputs',shape=[max_len])
    layer = Embedding(max_words,50,input_length=max_len)(inputs)
    layer = LSTM(64)(layer) # LSTM
    layer = Dense(256,name='FC1')(layer) # DENSE
    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) # OUTPUT
    return model

    model = lstm()
```



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

```
[175] def lstm():
          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
```

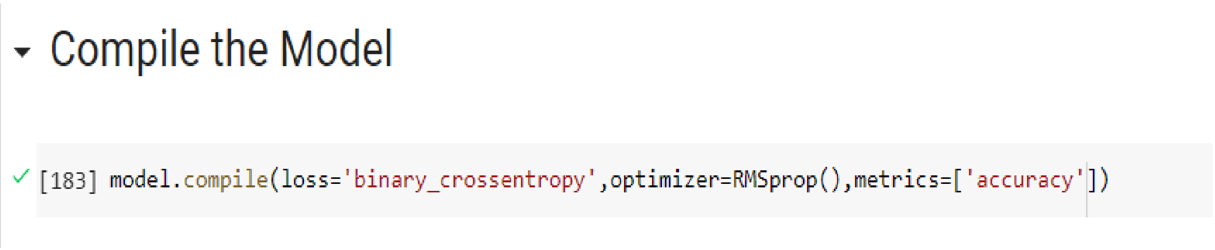
```
[178] model = lstm()
```

Question-4:

Compile The Model

Solution:

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



Compile the Model

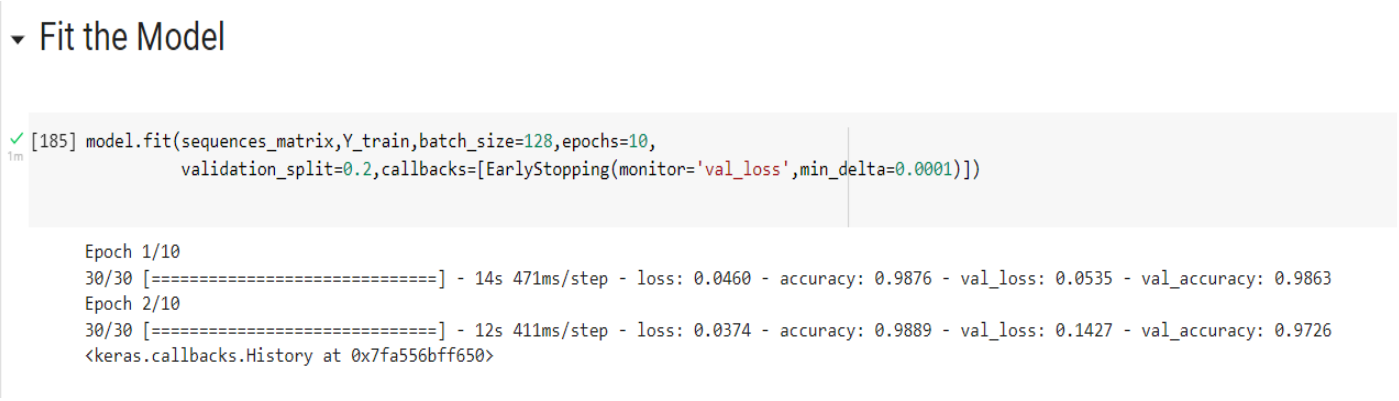
```
[183] model.compile(loss='binary_crossentropy',optimizer=RMSprop(),metrics=['accuracy'])
```

Question-5:

Fit The Model

Solution:

model.fit(sequences_matrix,Y_train,batch_size=128,epochs=10, validation_split=0.2,callbacks=[EarlyStopping(monitor='val_loss', min_delta=0.0001)])



Fit the Model

```
[185] model.fit(sequences_matrix,Y_train,batch_size=128,epochs=10,
              validation_split=0.2,callbacks=[EarlyStopping(monitor='val_loss',min_delta=0.0001)])
```

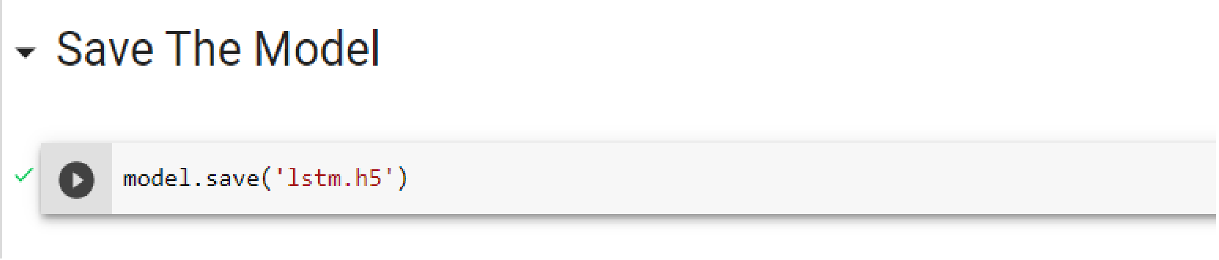
```
Epoch 1/10
30/30 [==============================] - 14s 471ms/step - loss: 0.0460 - accuracy: 0.9876 - val_loss: 0.0535 - val_accuracy: 0.9863
Epoch 2/10
30/30 [==============================] - 12s 411ms/step - loss: 0.0374 - accuracy: 0.9889 - val_loss: 0.1427 - val_accuracy: 0.9726
<keras.callbacks.History at 0x7fa556bff650>
```

Question-6:

Save The Model

Solution:

model.save('lstm.h5')



Save The Model

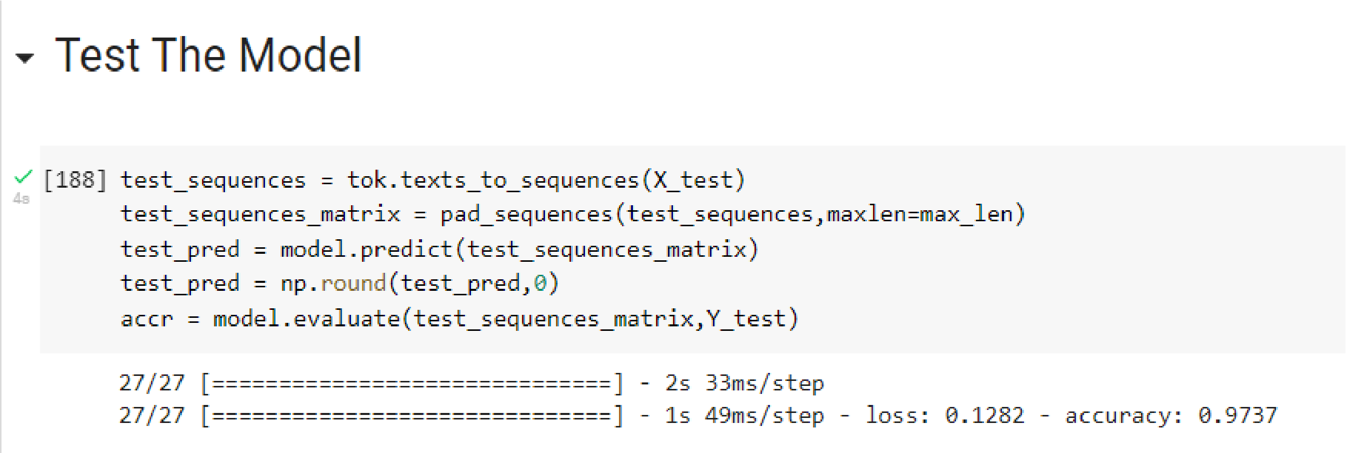
```
model.save('lstm.h5')
```

Question-7:

Test The Model

Solution:

```
test_sequences = tok.texts_to_sequences(X_test)
test_sequences_matrix = pad_sequences(test_sequences,maxlen=max_len)
test_pred = model.predict(test_sequences_matrix)
test_pred = np.round(test_pred,0)
accr = model.evaluate(test_sequences_matrix,Y_test)
```



Test The Model

```
test_sequences = tok.texts_to_sequences(X_test)
test_sequences_matrix = pad_sequences(test_sequences,maxlen=max_len)
test_pred = model.predict(test_sequences_matrix)
test_pred = np.round(test_pred,0)
accr = model.evaluate(test_sequences_matrix,Y_test)
```

```
27/27 [==============================] - 2s 33ms/step
27/27 [==============================] - 1s 49ms/step - loss: 0.1282 - accuracy: 0.9737
```