

Assignment-4

AssignmentDate	17 november2022
StudentName	S. Girija
StudentRollNumber	421319104006
MaximumMarks	2Marks

Question1:

DownloadthedatasetL

ink:

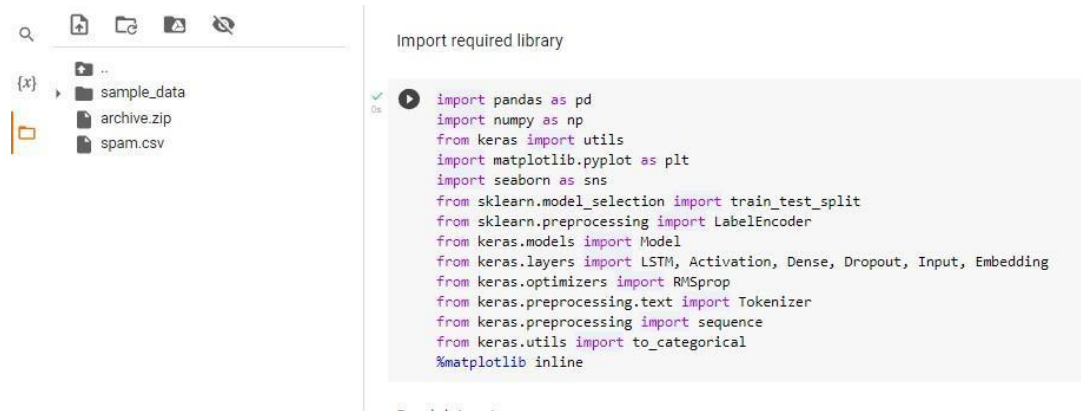
https://drive.google.com/file/d/1Sjqx5H5R86tRp2YZKzzd4_iEfjChZ3ob/view?usp=sharing

Question2:

ImportrequiredlibraryS

olution:

```
import pandas as pd
import numpy as np
from keras import utils
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 to_categorical
```



Question3:

Readdatasetanddopre-

processingSolution:

Readdataset

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

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

Preprocessing

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

```
sns.countplot(df.v1,palette='Set3')pl
t.xlabel('Label')
plt.title('Numberofhamandspammessages')
```

```
X =
df.v2Y=d
f.v1
le=LabelEncoder()Y=
le.fit_transform(Y)Y=
Y.reshape(-1,1)
```

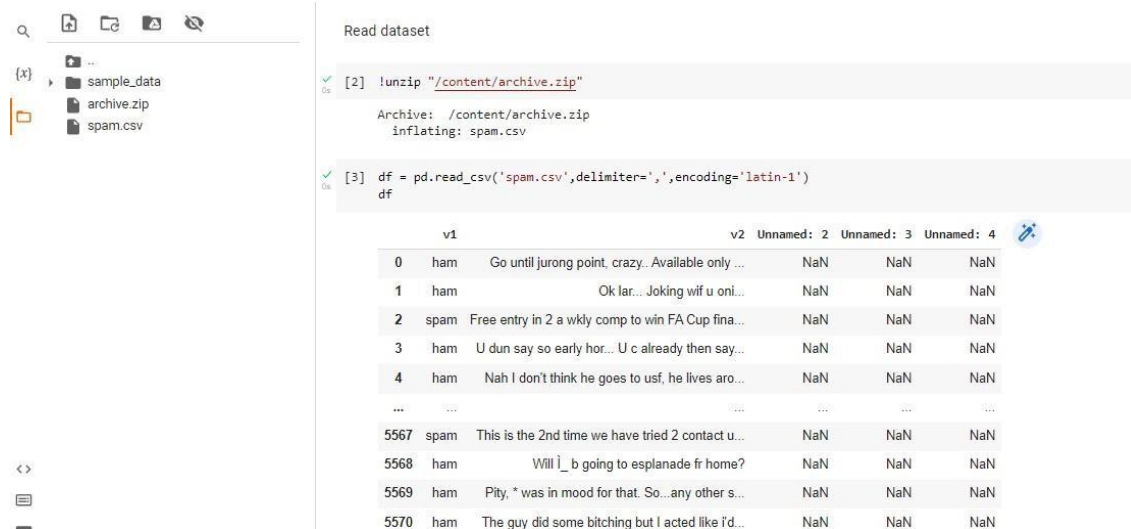
```
X_train,X_test,Y_train,Y_test=
```

```
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=utils.pad_sequences(sequences,maxlen=max_len)
```

sequences_matrix.shapes

sequences_matrix.ndim

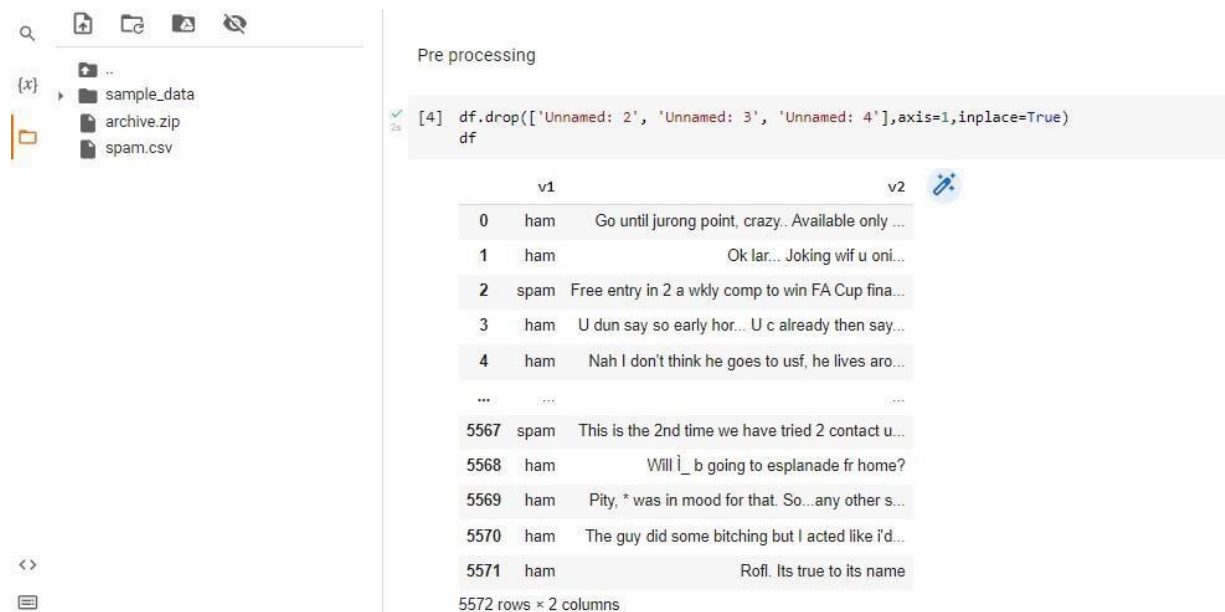
```
sequences_matrix=  
np.reshape(sequences_matrix,(4736,150,1))sequences_matrix.n  
dim
```



The Jupyter Notebook interface shows a file explorer on the left with a folder named 'sample_data' containing 'archive.zip' and 'spam.csv'. The main area is titled 'Read dataset' and contains two code cells. Cell [2] unzips the archive, and cell [3] reads the CSV file into a DataFrame. The DataFrame preview shows columns 'v1' and 'v2', with three unnamed columns containing NaN values.

```
[2] !unzip "/content/archive.zip"  
  
Archive: /content/archive.zip  
  inflating: spam.csv  
  
[3] df = pd.read_csv('spam.csv',delimiter=',',encoding='latin-1')  
df
```

	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 i_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



The Jupyter Notebook interface shows the same file explorer. The main area is titled 'Pre processing' and contains a code cell [4] that drops the unnamed columns from the DataFrame. The DataFrame preview shows only two columns, 'v1' and 'v2', with 5572 rows.

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

	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 i_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 x 2 columns



```
[7] X = df.v2
    Y = df.v1
    le = LabelEncoder()
    Y = le.fit_transform(Y)
    Y = Y.reshape(-1,1)

[8] X_train,X_test,Y_train,Y_test = train_test_split(X,Y,test_size=0.15)

[9] 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 = utils.pad_sequences(sequences,maxlen=max_len)

[10] sequences_matrix.shape
(4736, 150)

[11] sequences_matrix.ndim
2

[12] sequences_matrix = np.reshape(sequences_matrix,(4736,150,1))
    sequences_matrix.ndim
3
```

Question4:

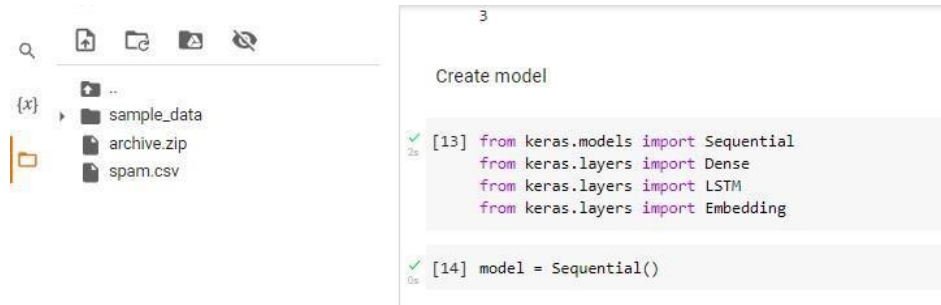
Create

modelSolution

n:

```
from keras.models import Sequential
from keras.layers import Dense
from keras.layers import LSTM
from keras.layers import Embedding
```

```
model=Sequential()
```

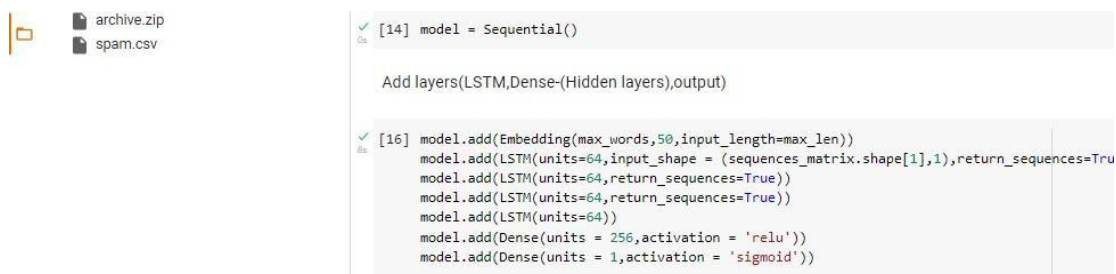


Question5:

Addlayers(LSTM,Dense-

(Hiddenlayers),output)Solution:

```
model.add(Embedding(max_words,50,input_length=max_len))model.add(LSTM(units=64,input_shape =
(sequences_matrix.shape[1],1),return_sequences=True))model.add(LSTM(units=64,return_sequences
s=True))model.add(LSTM(units=64,return_sequences=True))
model.add(LSTM(units=64))model.add(Dense(units
s = 256,activation =
'relu'))model.add(Dense(units=1,activation='sigmoid'))
```

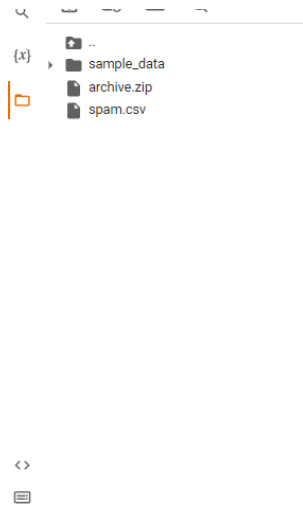


Question6:

Compilethemodel

Solution:

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



Compile the model

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

Model: "sequential"

Layer (type)	Output Shape	Param #
embedding (Embedding)	(None, 150, 50)	50000
lstm (LSTM)	(None, 150, 64)	29440
lstm_1 (LSTM)	(None, 150, 64)	33024
lstm_2 (LSTM)	(None, 150, 64)	33024
lstm_3 (LSTM)	(None, 64)	33024
dense (Dense)	(None, 256)	16640
dense_1 (Dense)	(None, 1)	257

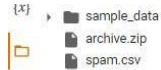
=====
Total params: 195,409
Trainable params: 195,409
Non-trainable params: 0

Question7:

Fitthemodel

Solution:

X=model.fit(sequences_matrix,Y_train,batch_size=128,epochs=5,validation_split=0.2)X



Fit the model

```
[18] X = model.fit(sequences_matrix,Y_train,batch_size=128,epochs=5,validation_split=0.2)  
X
```

```
Epoch 1/5  
30/30 [=====] - 43s 1s/step - loss: 0.4490 - accuracy: 0.8688 - val_loss: 0.4257 - val_accuracy: 0.8513  
Epoch 2/5  
30/30 [=====] - 33s 1s/step - loss: 0.2615 - accuracy: 0.9092 - val_loss: 0.1283 - val_accuracy: 0.9610  
Epoch 3/5  
30/30 [=====] - 35s 1s/step - loss: 0.0724 - accuracy: 0.9794 - val_loss: 0.0896 - val_accuracy: 0.9705  
Epoch 4/5  
30/30 [=====] - 33s 1s/step - loss: 0.0529 - accuracy: 0.9857 - val_loss: 0.0853 - val_accuracy: 0.9778  
Epoch 5/5  
30/30 [=====] - 33s 1s/step - loss: 0.0392 - accuracy: 0.9900 - val_loss: 0.0836 - val_accuracy: 0.9768  
<keras.callbacks.History at 0x7f263f739bd0>
```

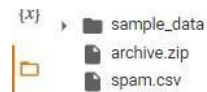
Save the model

Question8:Sav

ethemodelSolu

tion:

model.save



Save the model

```
[19] model.save
```

```
<bound method Model.save of <keras.engine.sequential.Sequential object at 0x7f2643a9c750>>
```

Question9:

Testthemodel

Solution:

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

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

```
l=accr[0]a
=accr[1]
print('Testset\nLoss:{:0.3f}\nAccuracy: {:.3f}'.format(l,a))
```

