Assignment-4 PythonProgramming

AssignmentDate	20October 2022
StudentName	Zaheeb Afnan A
StudentRollNumber	511919104026
MaximumMarks	2Marks

Dataset

Question-1:

Importing the dataset

Solution-1:

from google.colab import drive drive.mount('/content/drive')

Output:

Mountedat/content/drive

Solution-2:

dataset_location = "/content/drive/MyDrive/IBM/spam.csv"

Output:

/content/drive/MyDrive

Question-2

Load the dataset

Solution-1:

import pandas as pd

import nltk

import re

import numpy as np

from nltk.corpus import stopwords

from nltk.stem.porter import PorterStemmer

from nltk.translate.ribes_score import word_rank_alignment

from numpy.lib.shape base import split

from sklearn import preprocessing

from sklearn.feature_extraction.text import CountVectorizer

from tensorflow.keras.models import Sequential

from sklearn.model selection import train test split

from keras.layers import LSTM, Dense, Dropout, Input, Embedding, Activation, Flatten

from keras.models import Model

import nltk

Output:

dataset import successfully

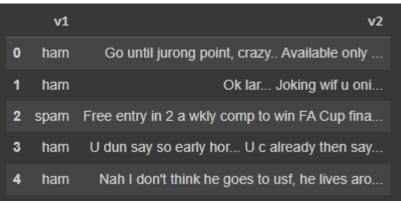
Question-3

Read dataset and do pre-processing

Solution:

```
data = pd.read_csv(dataset_location,encoding = "ISO-8859-1")
data.drop(["Unnamed: 2","Unnamed: 3","Unnamed: 4"],axis = 1,inplace = True)
data.head()
```

Output:



Solution-2:

```
nltk.download('stopwords',quiet=True)
nltk.download('all',quiet=True)
```

Output:

True

Solution-3:

```
ps = PorterStemmer()
input = []
for i in range(0,5572):
 v2 = data['v2'][i]
 #removing punctuation
 v2 = re.sub('[^a-zA-Z]',' ',v2)
 #converting to lower case
 v2 = v2.lower()
 #splitting the sentence
 v2 = v2.split()
#removing the stopwords and stemming
v2 = [ps.stem(word) for word in v2 if not word in set(stopwords.words('english'))]
v2 = ' '.join(v2)
input.append(v2)
#creating document term matrix
cv = CountVectorizer(max features=2000)
x = cv.fit transform(input).toarray()
x.shape
```

Output:

(5572*,* 2000)

```
Solution-4:
```

```
le = preprocessing.LabelEncoder()
data['v1'] = le.fit_transform(data['v1'])
data['v1'].unique()
```

Output:

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

Solution-5:

```
y = data['v1'].values
y = y.reshape(-1,1)
x_train,x_test,y_train,y_test = train_test_split(x,y,test_size = 0.4)
```

Output:

Successfully reshaped

Question-4:

Create model

Solution:

model = Sequential()

Output:

Model created successfully

Question-5:

5. Add layers (LSTM, Dense-(Hidden Layers), Output

Solution:

```
model.add(Dense(1565,activation = "relu"))
model.add(Dense(3000,activation = "relu"))
model.add(Dense(1,activation = "sigmoid"))
model.add(Flatten())
```

Output:

Model added

```
Compile the model
Solution:
       model.compile(optimizer = "adam",loss = "binary_crossentropy", metrics = ["accuracy"])
Output:
       Model compiled
Question-7:
       Fit the model
Solution:
       model.fit(x_train,y_train,epochs = 15)
Output:
       Model fit
Question-8:
       Save the model
Solution:
       model.save("spam-message-classifier.h5")
Output:
       Model saved
Question-9:
       Test the model
Solution:
       ham = "imdonee. come pick me up"
       spam = "WINNER$$$$ SMS REPLY 'WIN'"
       message = re.sub('[^a-zA-Z]',' ',spam)
       message
Output:
       WINNER SMS REPLY WIN
Solution-2:
       message = message.split()
       message = [ps.stem(word) for word in message if not word in
       set(stopwords.words('english')) ]
       message = ' '.join(message)
       message1 = cv.transform([message])
       message1
Output:
       <1x2000 sparse matrix of type '<class 'numpy.int64'>'
              with 4 stored elements in Compressed Sparse Row format>
```

Question-6:

```
Solution-3:
      TruePredction = model.predict(message1.astype(float))
Output:
       1/1 [======] - 0s 13ms/step
Solution-4:
      TruePredction> 0.5
Output:
      array([[ True]])
Solution-5:
      TruePredction> 0.5
Output:
       array([[ True]])
Solution-6:
      msg = re.sub('[^a-zA-Z]',' ',ham)
      msg
Output:
       imdonee come pick me up
Solution-7:
      msg = msg.split()
      msg = [ps.stem(word) for word in msg if not word in set(stopwords.words('english'))]
      msg = ' '.join(msg)
      msg
Output:
      im done come pick
Solution-8:
      cv.transform([msg])
Output:
      <1x2000 sparse matrix of type '<class 'numpy.int64'>' with 4 stored elements in
      Compressed Sparse Row format>
```

##