**Assignment -IV STM for Text Classification**

|  |  |
| --- | --- |
| Assignment Date | 11 November 2022 |
| Student Name | S. Alagammai |
| Student Roll Number | 9517201903007 |
| Maximum Marks | 2 Marks |

**#Import necessary libraries**

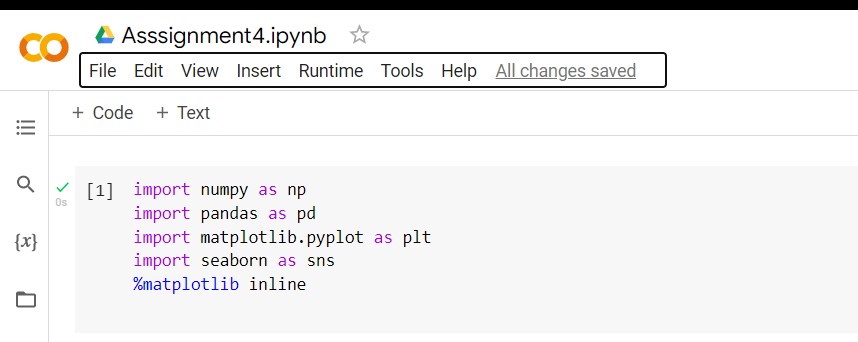
import numpy as np import pandas as pd import matplotlib.pyplot as plt import seaborn as sns %matplotlib inline

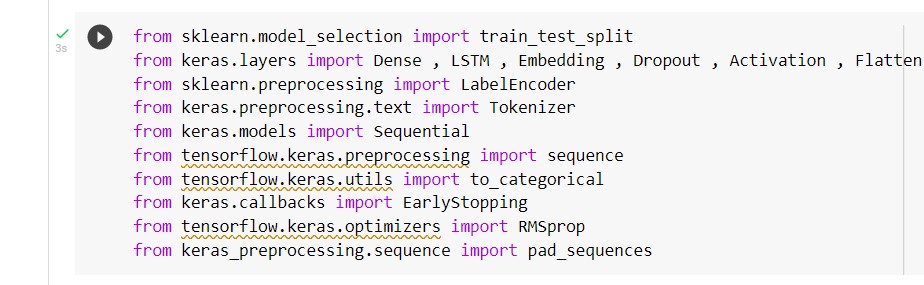
from sklearn.model\_selection import train\_test\_split

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

from sklearn.preprocessing import LabelEncoder from keras.preprocessing.text import Tokenizer from keras.models import Sequential

from tensorflow.keras.preprocessing import sequence from tensorflow.keras.utils import to\_categorical from keras.callbacks import EarlyStopping from tensorflow.keras.optimizers import RMSprop from keras\_preprocessing.sequence import pad\_sequences





# #Read dataset and do pre-processing

data = pd.read\_csv('/content/spam.csv',delimiter=',',encoding='latin-1') data

#Information about dataset data.describe().T data.shape

#Check if there is any missing values data.isnull().sum()

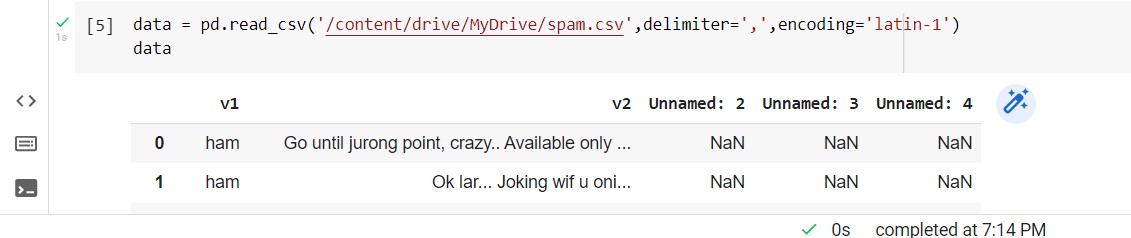
data.drop(['Unnamed: 2', 'Unnamed: 3', 'Unnamed: 4'],axis=1,inplace=True)

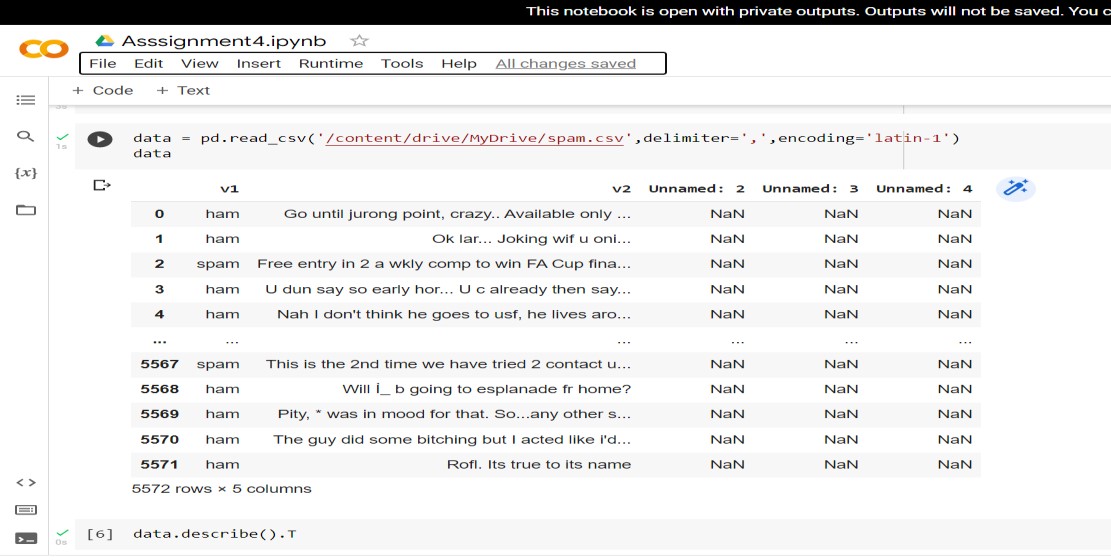
#Visualize the dataset sns.countplot(data.v1)

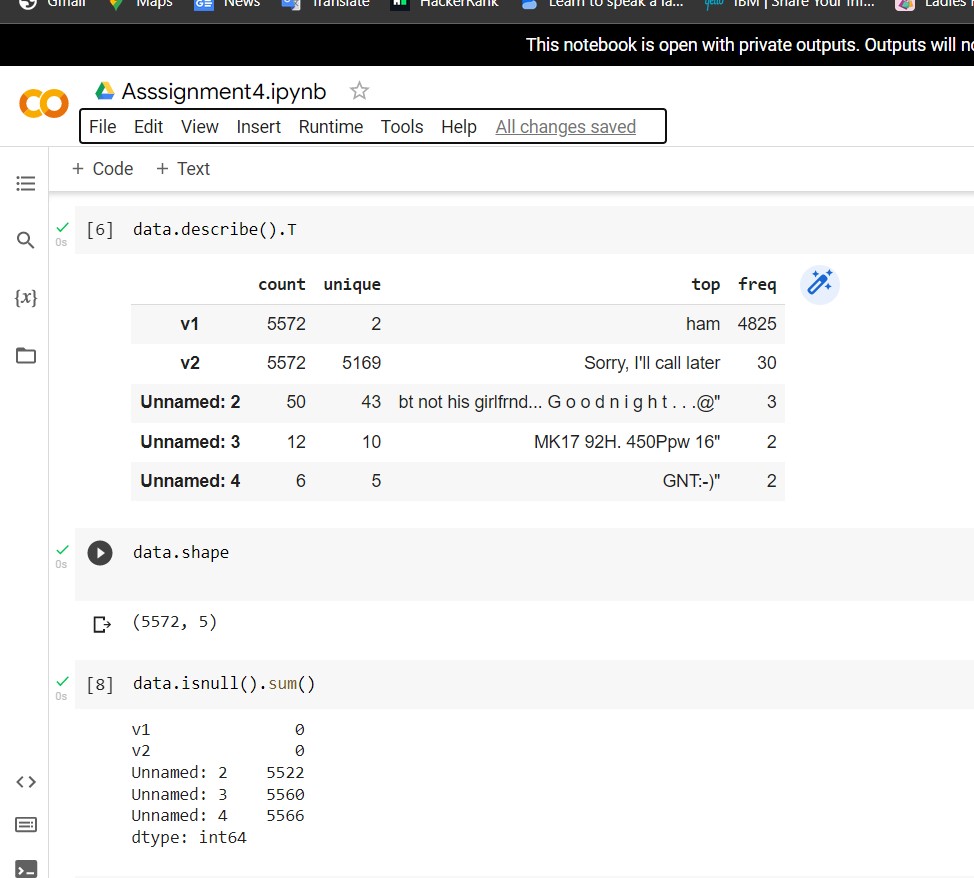
#Preprocess using Label Encoding

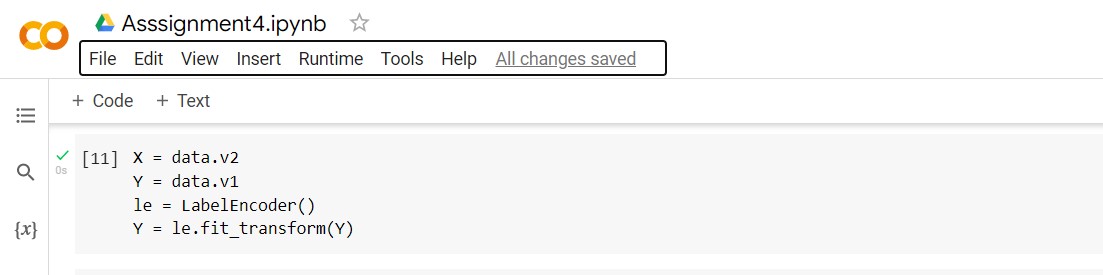
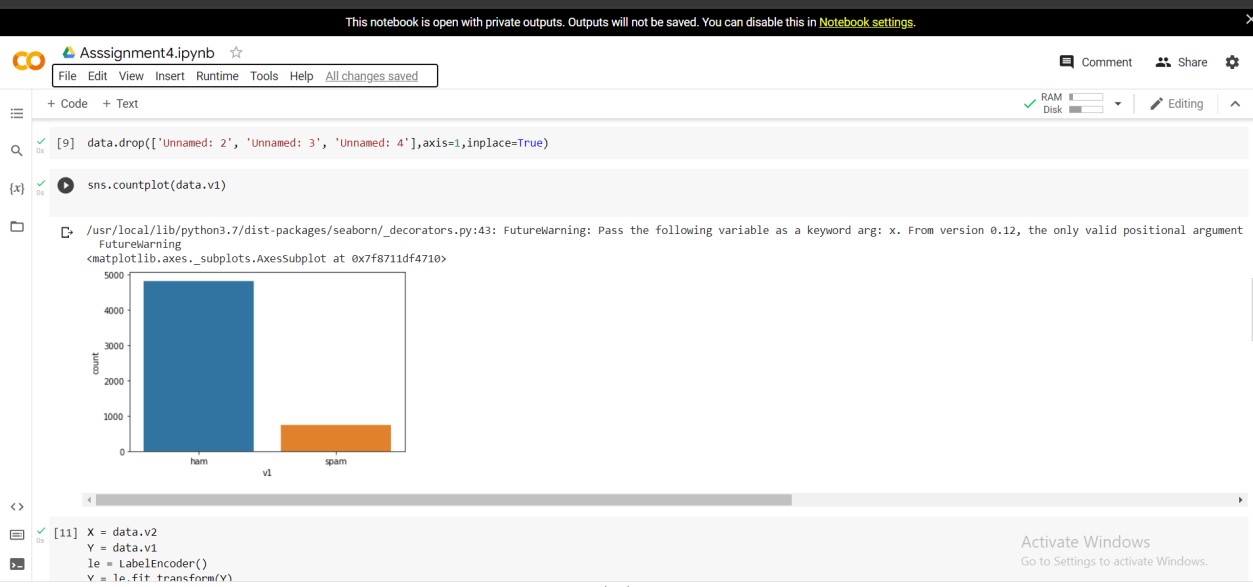
1. = data.v2 Y = data.v1 le = LabelEncoder()
2. = le.fit\_transform(Y)

Y = Y.reshape(-1,1)









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

#Splitting into training and testing data

X\_train,X\_test,Y\_train,Y\_test = train\_test\_split(X,Y,test\_size = 0.2) max\_word = 1000 max\_len = 250

token = Tokenizer(num\_words = max\_word) token.fit\_on\_texts(X\_train)

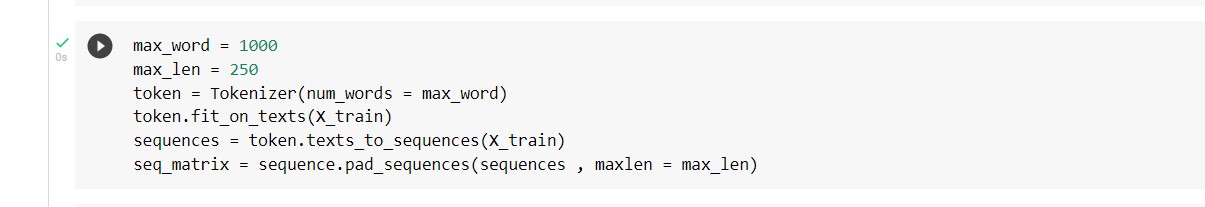
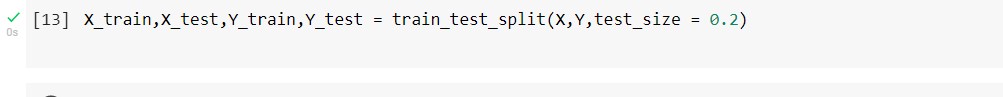
sequences = token.texts\_to\_sequences(X\_train)

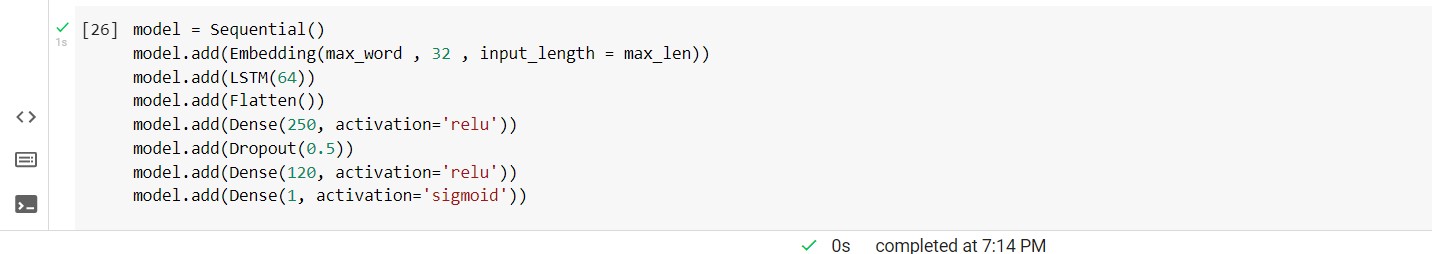
seq\_matrix = sequence.pad\_sequences(sequences , maxlen = max\_len)

#Creating the model model = Sequential() model.add(Embedding(max\_word , 32 , input\_length = max\_len)) model.add(LSTM(64)) model.add(Flatten())

model.add(Dense(250, activation='relu')) model.add(Dropout(0.5))

model.add(Dense(120, activation='relu')) model.add(Dense(1, activation='sigmoid'))

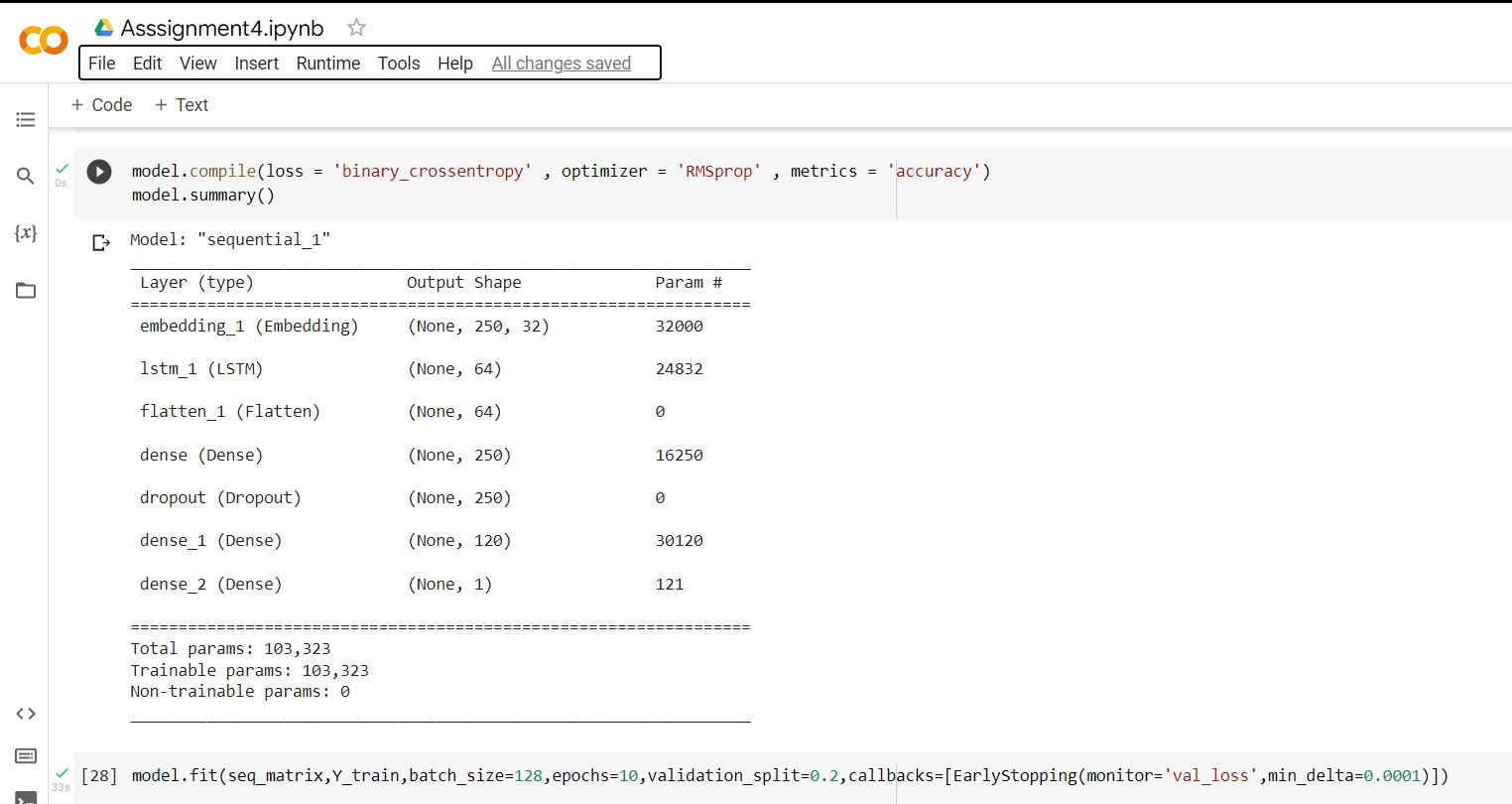




# #compile the model

model.compile(loss = 'binary\_crossentropy' , optimizer = 'RMSprop' , metrics =

'accuracy') model.summary()



# #Fit the model

model.fit(seq\_matrix,Y\_train,batch\_size=128,epochs=10,validation\_split=0.2,c allbacks=[EarlySt

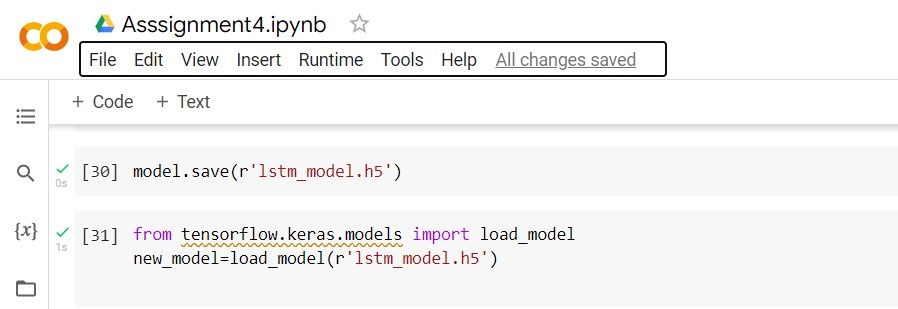
opping(monitor='val\_loss',min\_delta=0.0001)]) test\_seq = token.texts\_to\_sequences(X\_test)

test\_seq\_matrix = sequence.pad\_sequences(test\_seq,maxlen=max\_len)



# #Save the model

model.save(r'lstm\_model.h5')



**#Test the model:** from tensorflow.keras.models import load\_model new\_model=load\_model(r'lstm\_model.h5') new\_model.evaluate(test\_seq\_matrix,Y\_test)

scores = model.evaluate(test\_seq\_matrix, Y\_test, verbose=0) scores

print("Accuracy: %.2f%%" % (scores[1]\*100))

