CHANDIGARH UNIVERSITY
Discover. Learn. Empower.
NAAC GRADE A+
Accredited University

Subject Code: 21CAP722

Final LAB MST

Student Name:Km Rishika Jaiswal **UID: 21MCI1152**
Section/Group: 21MAM-1_B **Semester: 3rd**
Date of Submission: 14/11/2022 **Course: MCA (AIML)**
Subject: Machine Learning Lab

Aim/Overview of the practical:

Task to be done:

Twitter has become an important communication channel in times of emergency.
The ubiquitousness of smartphones enables people to announce an emergency they're observing in real-time. Because of this, more agencies are interested in programatically monitoring Twitter (i.e. disaster relief organizations and news agencies).

The author explicitly uses the word "ABLAZE" but means it metaphorically. This is clear to a human right away, especially with the visual aid. But it's less clear to a machine.

In this competition, you're challenged to build a machine learning model that predicts which Tweets are about real disasters and which one's aren't. You'll have access to a dataset of 10,000 tweets that were hand classified

Solution:

1. **Code for experiment/practical**

```
import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)
# visulization
import matplotlib.pyplot as plt
# %matplotlib inline
```

```
import seaborn as sns
# to detect languages
import langid

pip install langid

df_train=pd.read_csv('train.csv')

df_test=pd.read_csv('test.csv')

df_train.head()

df_test.head()

df=df_train

df.info()

df.shape

df.isna().sum()

df.duplicated().sum()

sns.countplot(df['target'])
plt.title("Countplot for target Labels")

df['target'].value_counts()

min_val_sum = min(df.text,key=len)
print("The minimum text is:\n" ,min_val_sum,"\nAnd his length",len(min_val_sum))

max_val_sum = max(df['text'], key=len)
print("The maximum text is:\n" ,max_val_sum,"\nAnd his length",len(max_val_sum))

ids_langid=df['text'].apply(langid.classify)
langs = ids_langid.apply(lambda tuple: tuple[0])
print("Number of tagged languages (estimated):")
print(len(langs.unique()))
print("Percent of data in English (estimated):")
print((sum(langs=="en")/len(langs))*100)

langs_df = pd.DataFrame(langs)
langs_count = langs_df.text.value_counts()
print(langs_count)

langs_count.plot.bar(figsize=(15,10), fontsize=15)

df['detect']=langs=="en"
```

```
non_english_text=df[df['detect'] == False]
non_english_text.head()

pip install pySpellChecker

pip install contractions

from re import sub
from nltk.tokenize import word_tokenize
from nltk.corpus import stopwords
from nltk.stem import WordNetLemmatizer
from spellchecker import SpellChecker
import contractions as ct
import string
import nltk

df['text'][0:10].values

nltk.download('stopwords')

import nltk
nltk.download('punkt')

spell = SpellChecker()

corpus=[]
for i in df['text'].values:

    text=i

    #expand contraction in text
    text=[ct.fix(i) for i in text.split()]
    text=' '.join(text)

    #removing http and @
    text = sub(r"http\S+|@\S+", "", text)

    #Removing Punctuations
    text = sub("[^a-zA-Z]",' ',text)

        #Lowercasing
    text=text.lower()

    #Tokenization
    text=word_tokenize(text)

    #Spelling Correction
    text = [spell.correction(i) for i in text]
```

```
    #join text
    text = ' '.join(filter(lambda x: x if x is not None else '', text))


    corpus.append(text)

corpus[0:10]
```

```
df['clean_text']=corpus
df.head()

df=df[['clean_text','target']]
df
```

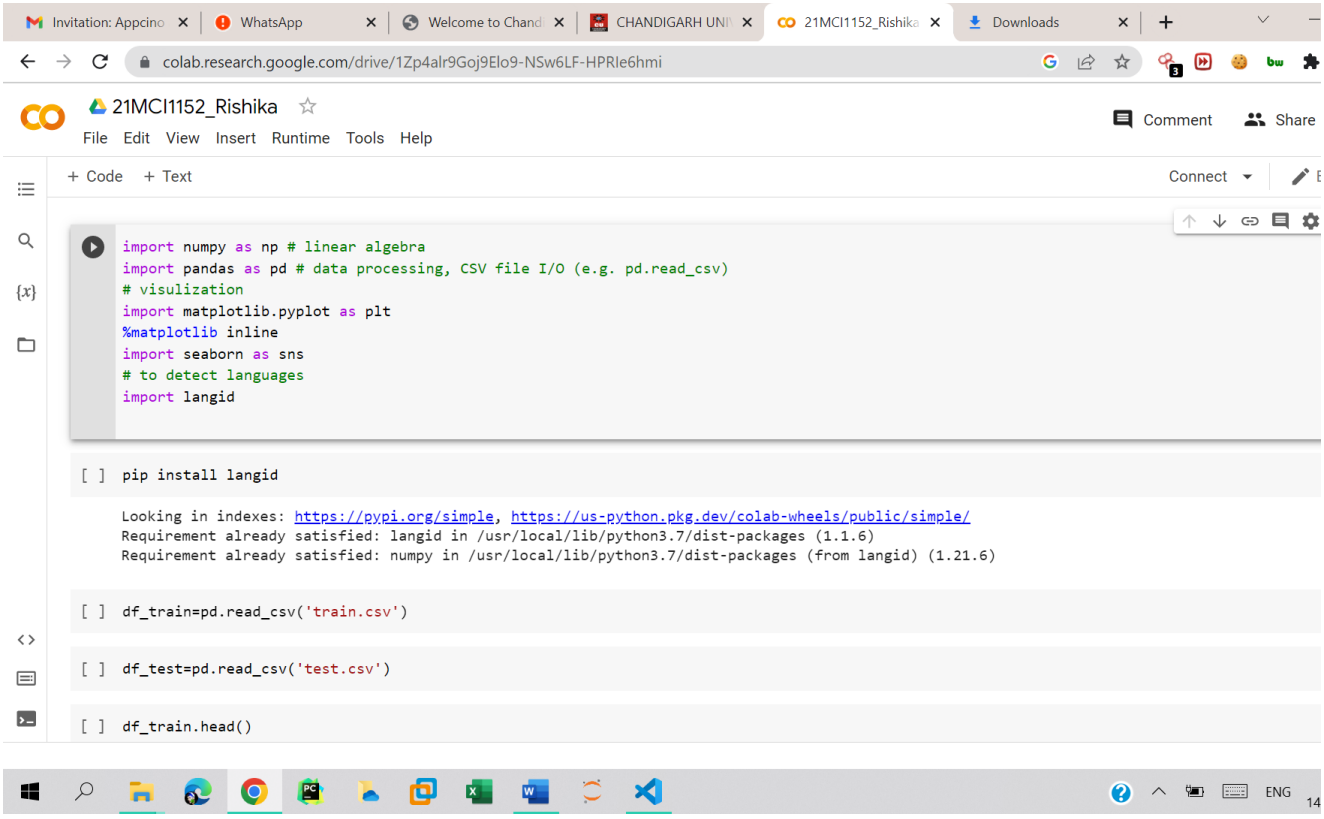
```
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.model_selection import train_test_split
from sklearn.metrics import confusion_matrix, classification_report, accuracy_score
```

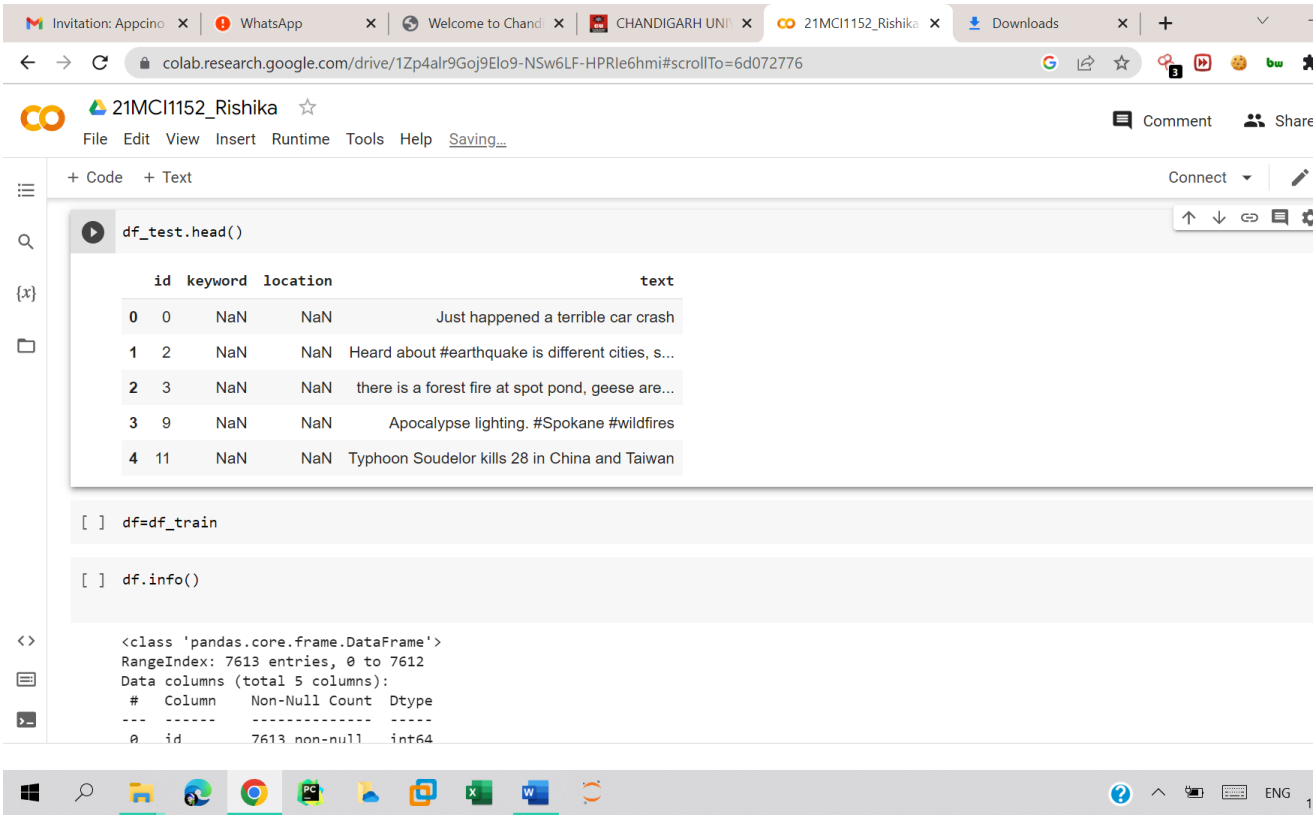
```
tfidf = TfidfVectorizer(stop_words='english')

X = df['clean_text']
y = df['target']

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size= 0.2, random_state= 0)

#apply feature extraxtion on train set
X_train = tfidf.fit_transform(X_train)
X_test = tfidf.transform(X_test)

print(len(y_train))
print(len(y_test))

def classification(model, X, y):

    model.fit(X_train, y_train)
    y_pred = model.predict(X_test)

    print('-Classification Report-\n')
    print(classification_report(y_test, y_pred))

    print('Accuracy= ', accuracy_score(y_test, y_pred)*100, '%\n')
```

```
    print('-Confusion Matrix-\n')
    cm=confusion_matrix(y_test, y_pred)
    sns.heatmap(cm, annot=True, fmt='d')
    plt.xlabel('Predicted')
    plt.ylabel('Truth')

from sklearn.ensemble import RandomForestClassifier

RF=RandomForestClassifier(n_estimators = 100, criterion = 'entropy' , random_state = 0)
classification(RF, X, y)

from sklearn.naive_bayes import MultinomialNB

NB=MultinomialNB()
classification(NB, X, y)

from sklearn.svm import SVC

SVM = SVC(kernel = 'rbf', random_state = 0)
classification(SVM, X, y)

from sklearn.svm import SVC

SVM = SVC(kernel = 'rbf', random_state = 0)
classification(SVM, X, y)

from sklearn.tree import DecisionTreeClassifier

DT= DecisionTreeClassifier(criterion = 'entropy', random_state = 0)
classification(DT, X, y)

from sklearn.ensemble import RandomForestClassifier

RF=RandomForestClassifier(n_estimators = 100, criterion = 'entropy' , random_state = 0)
classification(RF, X, y)

from sklearn.linear_model import LogisticRegression

LR = LogisticRegression()
LR.fit(X_train, y_train)
y_pred= LR.predict(X_test)

print('-Classification Report-\n\n',classification_report(y_test, y_pred))

print('Accuracy= ', accuracy_score(y_test, y_pred)*100, '%\n')

print('-Confusion Matrix-\n')
cm=confusion_matrix(y_test, y_pred)
```

```
sns.heatmap(cm, annot=True, fmt='d')
plt.xlabel('Predicted')
plt.ylabel('Truth')

df_test=df_test[['id','text']]

df_test.head()

x = tfidf.transform(df_test['text'])

df_test['target'] = LR.predict(x)

df_test.head()

submission = df_test[['id','target']]
submission.to_csv('submission.csv', index = False)

submission
```

2. Result/Output:





Invitation: Appcino | WhatsApp | Welcome to Chand | CHANDIGARH UNI | 21MCI1152_Rishika | Downloads

colab.research.google.com/drive/1Zp4alr9Goj9Elo9-NSw6LF-HPRIe6hmi#scrollTo=6d072776

21MCI1152_Rishika

File Edit View Insert Runtime Tools Help Saving...

Comment Share

+ Code + Text

Connect

```
df_test.head()
```

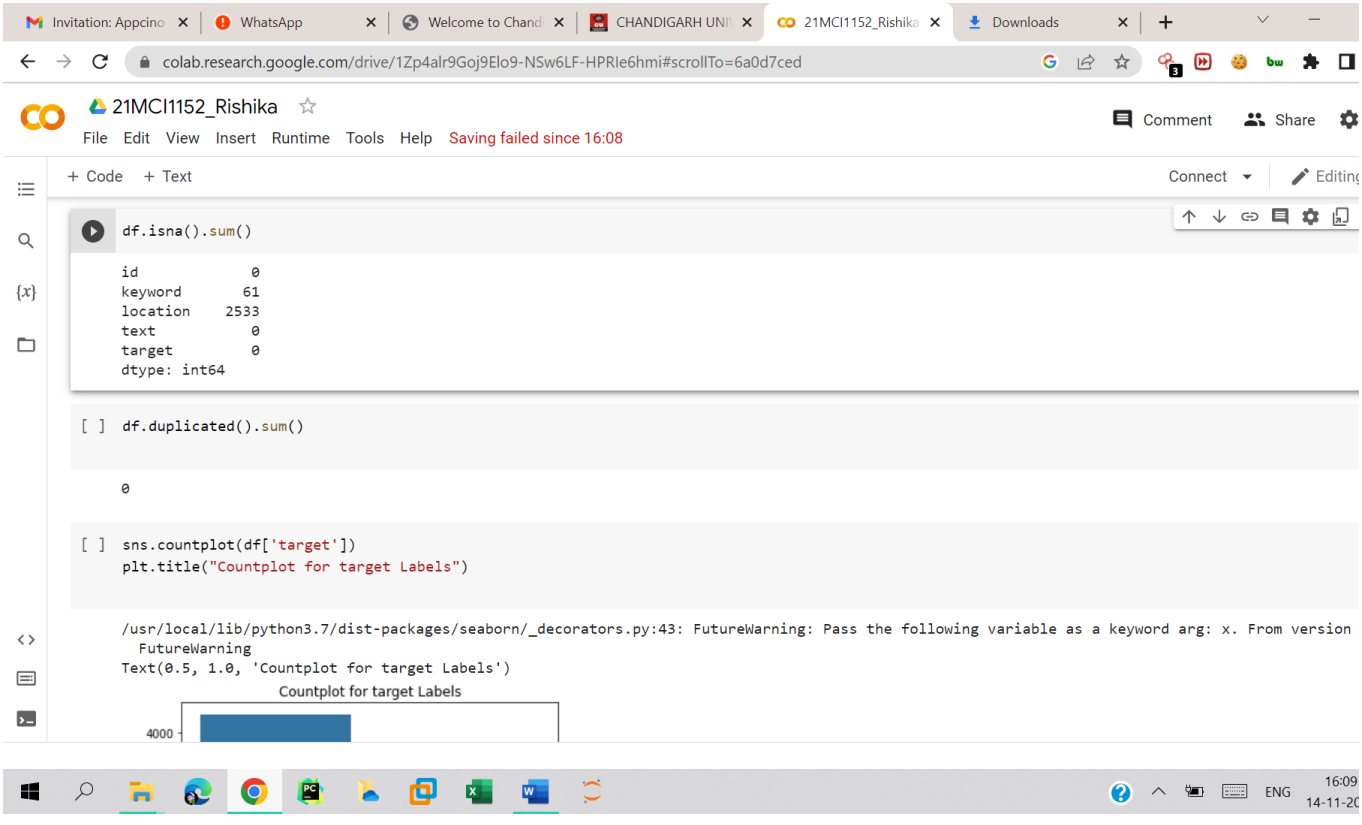
	id	keyword	location	text
0	0	NaN	NaN	Just happened a terrible car crash
1	2	NaN	NaN	Heard about #earthquake is different cities, s...
2	3	NaN	NaN	there is a forest fire at spot pond, geese are...
3	9	NaN	NaN	Apocalypse lighting. #Spokane #wildfires
4	11	NaN	NaN	Typhoon Soudelor kills 28 in China and Taiwan

```
[ ] df=df_train
```

```
[ ] df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 7613 entries, 0 to 7612
Data columns (total 5 columns):
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   id        7613 non-null   int64
```

ENG



Invitation: Appcino | WhatsApp | Welcome to Chand | CHANDIGARH UNI | 21MCI1152_Rishika | Downloads

colab.research.google.com/drive/1Zp4alr9Goj9Elo9-NSw6LF-HPRIe6hmi#scrollTo=6a0d7ced

21MCI1152_Rishika

File Edit View Insert Runtime Tools Help Saving failed since 16:08

Comment Share

+ Code + Text

Connect Editing

```
df.isna().sum()
```

```
id             0
keyword       61
location    2533
text           0
target         0
dtype: int64
```

```
[ ] df.duplicated().sum()
```

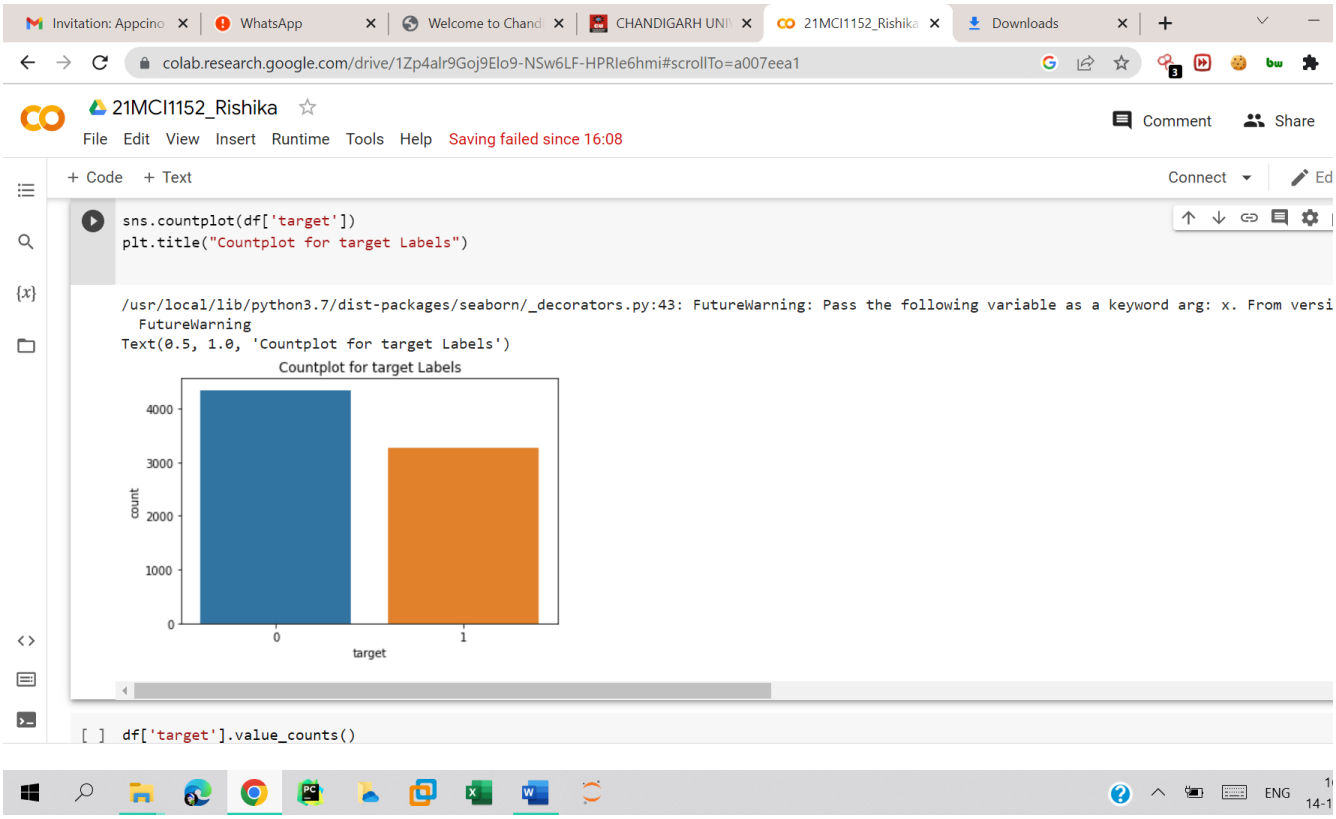
```
0
```

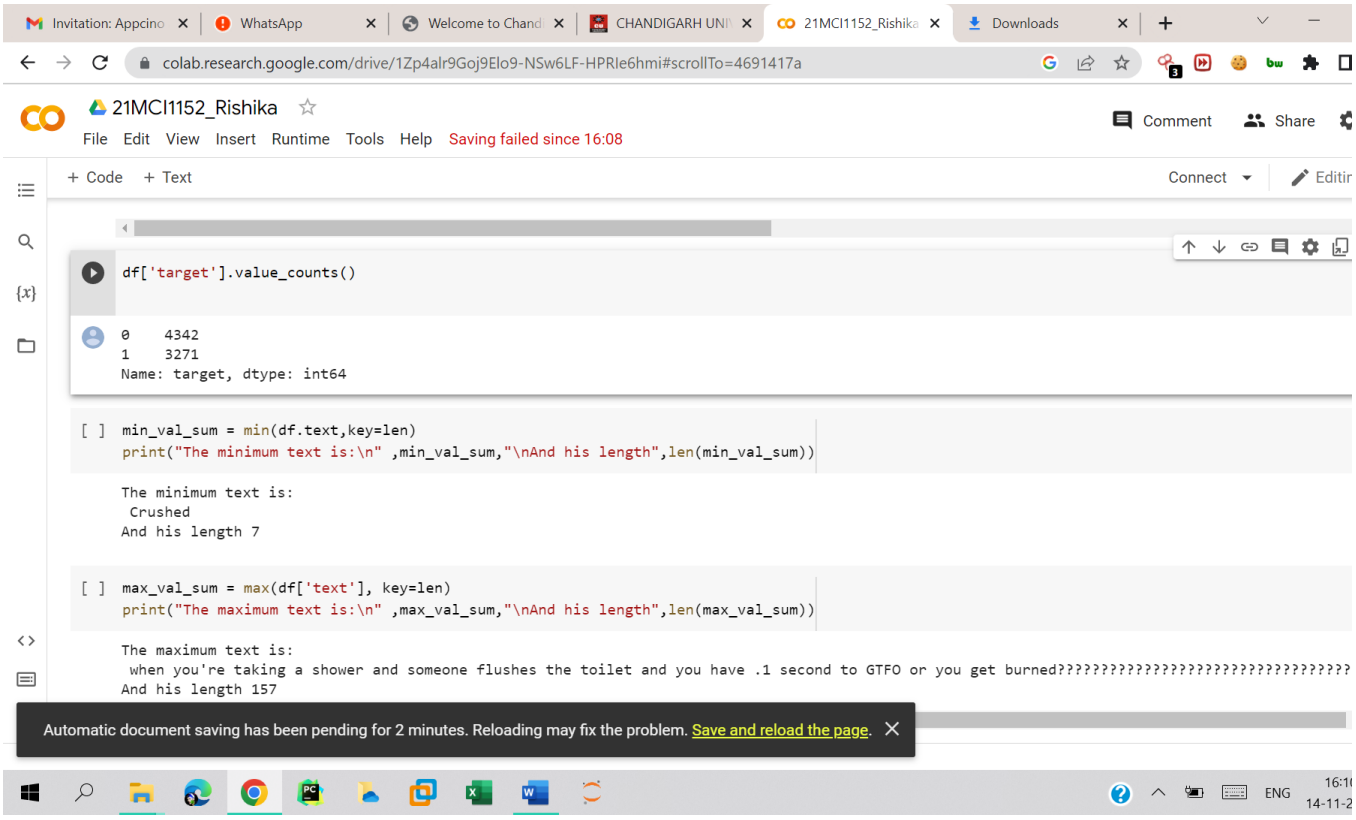
```
[ ] sns.countplot(df['target'])
    plt.title("Countplot for target Labels")
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning: Pass the following variable as a keyword arg: x. From version
  FutureWarning
Text(0.5, 1.0, 'Countplot for target Labels')
```

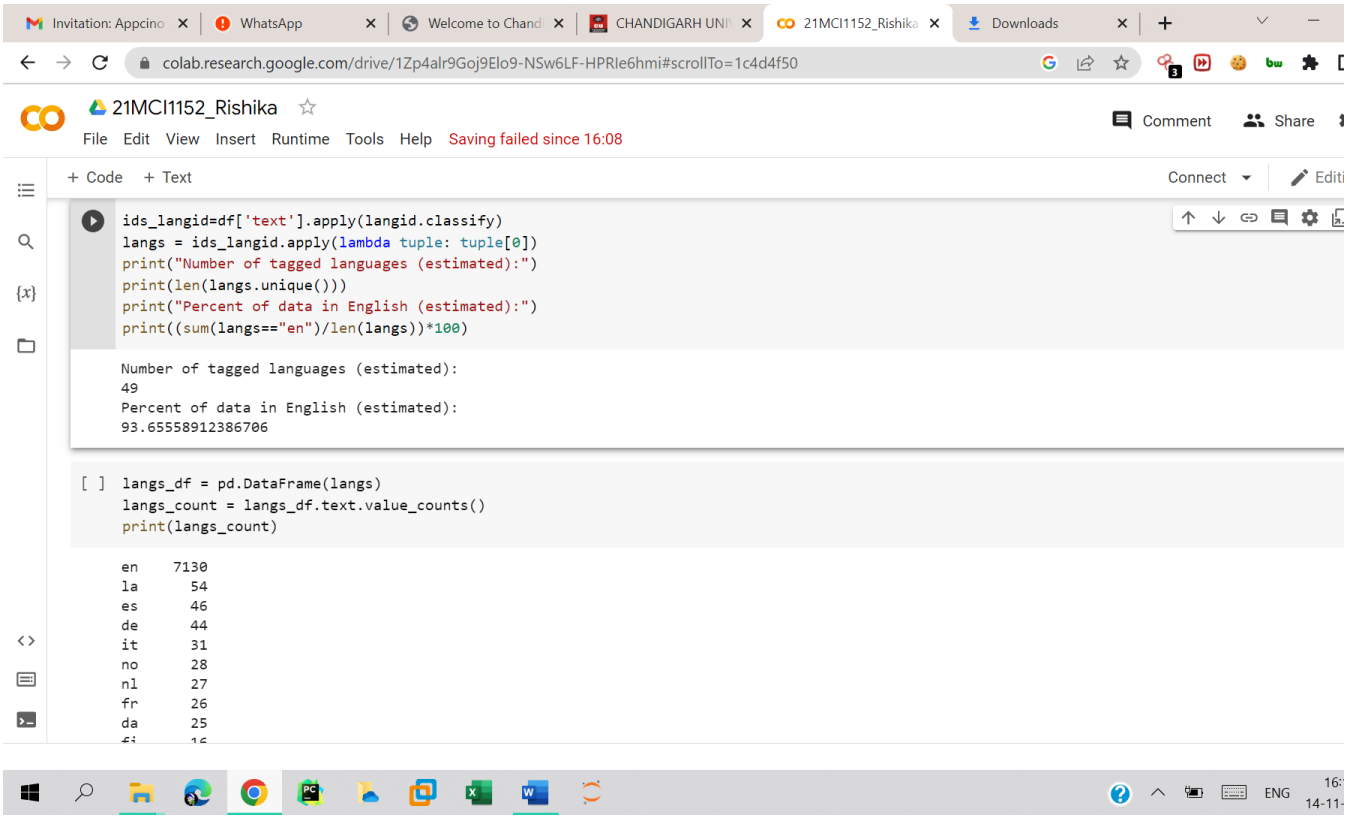

ENG 16:09 14-11-20

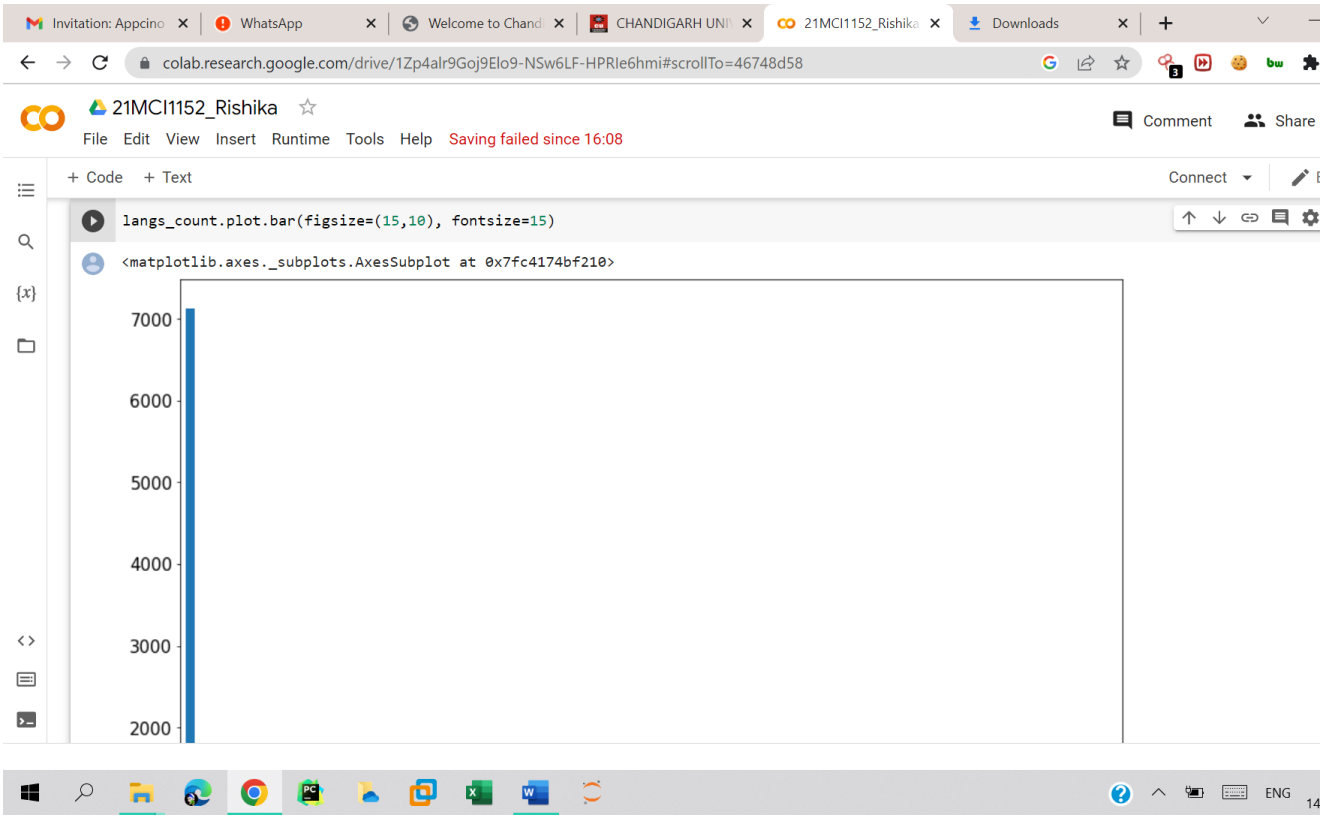
CHANDIGARH UNIVERSITY
Discover. Learn. Empower.
NAAC GRADE A+
Accredited University

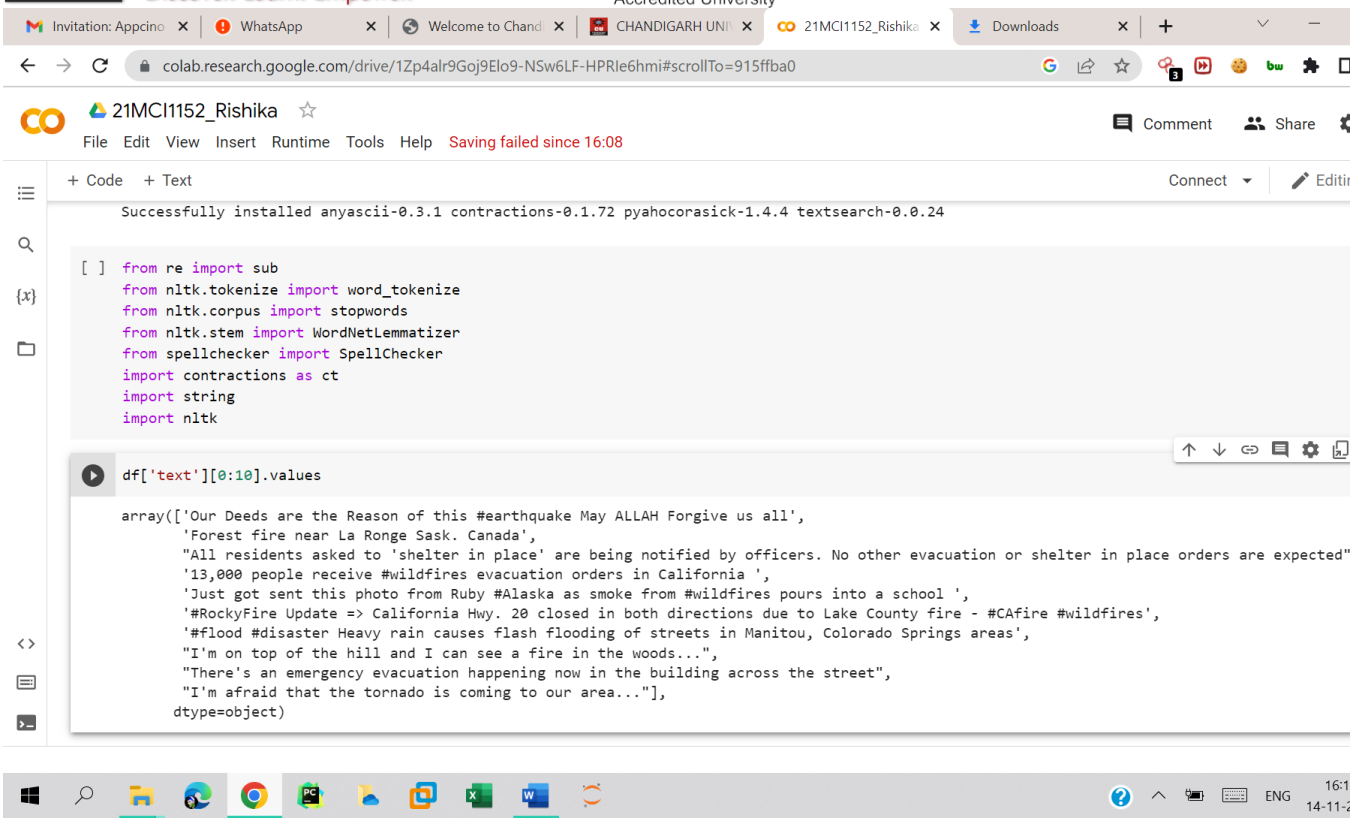


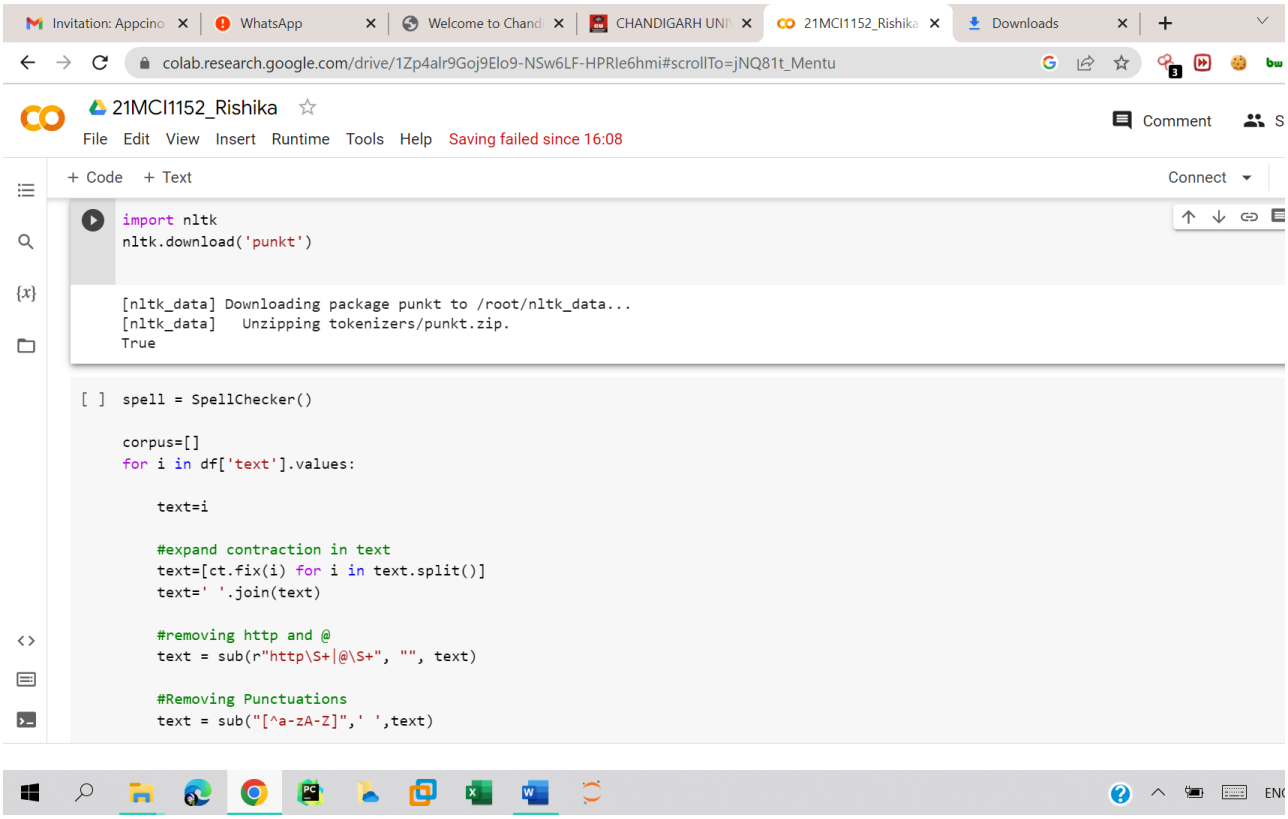


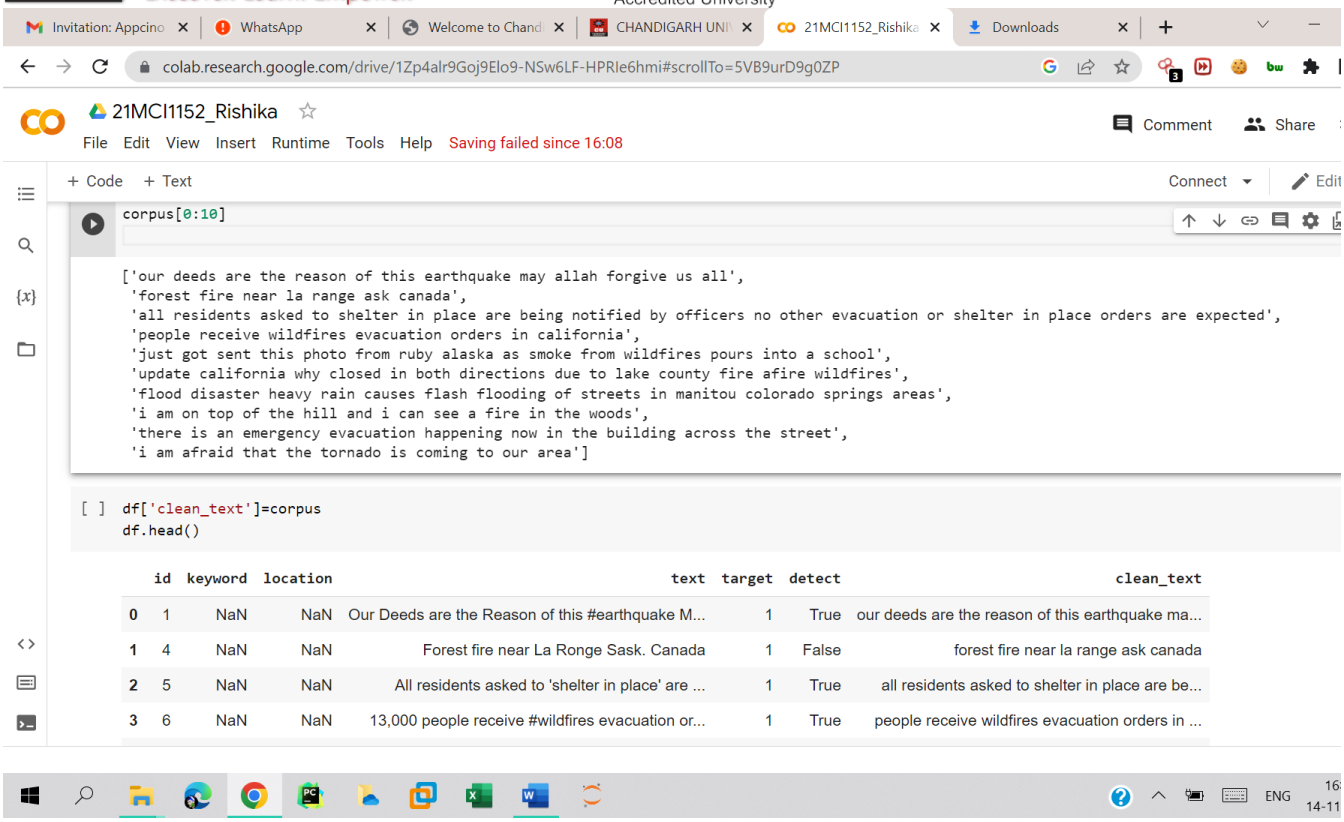
CHANDIGARH UNIVERSITY
Discover. Learn. Empower.
NAAC GRADE A+
Accredited University



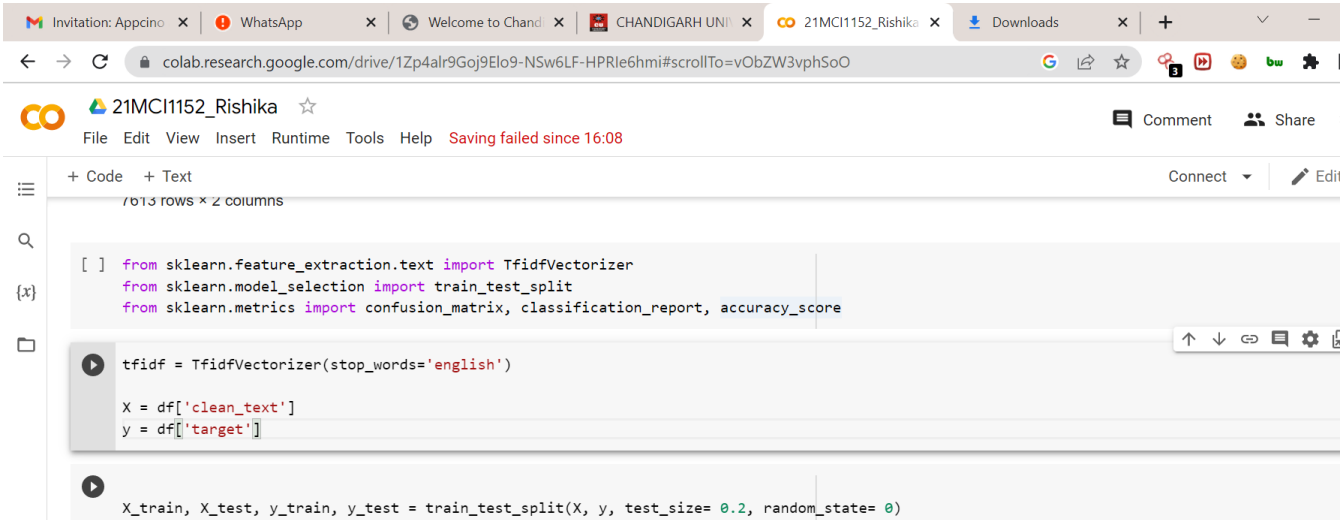








	id	keyword	location	text	target	detect	clean_text
0	1	NaN	NaN	Our Deeds are the Reason of this #earthquake M...	1	True	our deeds are the reason of this earthquake ma...
1	4	NaN	NaN	Forest fire near La Ronge Sask. Canada	1	False	forest fire near la range ask canada
2	5	NaN	NaN	All residents asked to 'shelter in place' are ...	1	True	all residents asked to shelter in place are be...
3	6	NaN	NaN	13,000 people receive #wildfires evacuation or...	1	True	people receive wildfires evacuation orders in ...



CHANDIGARH UNIVERSITY
Discover. Learn. Empower.
NAAC GRADE A+
Accredited University

Invitation: Appcino | WhatsApp | Welcome to Chand | CHANDIGARH UNI | 21MCI1152_Rishika | Downloads

colab.research.google.com/drive/1Zp4alr9Goj9Elo9-NSw6LF-HPRIe6hmi#scrollTo=49c2e69c

21MCI1152_Rishika

File Edit View Insert Runtime Tools Help Saving...

Comment

+ Code + Text

Connect

```
def classification(model, X, y):

    model.fit(X_train, y_train)
    y_pred = model.predict(X_test)

    print('-Classification Report-\n')
    print(classification_report(y_test, y_pred))

    print('Accuracy= ', accuracy_score(y_test, y_pred)*100, '%\n')

    print('-Confusion Matrix-\n')
    cm=confusion_matrix(y_test, y_pred)
    sns.heatmap(cm, annot=True, fmt='d')
    plt.xlabel('Predicted')
    plt.ylabel('Truth')
```

```
from sklearn.ensemble import RandomForestClassifier

RF=RandomForestClassifier(n_estimators = 100, criterion = 'entropy' , random_state = 0)
classification(RF, X, y)
```

```
-Classification Report-

              precision    recall  f1-score   support
```

Invitation: Appcino | WhatsApp | Welcome to Chand | CHANDIGARH UNI | 21MCI1152_Rishika | Downloads

colab.research.google.com/drive/1Zp4alr9Goj9Elo9-NSw6LF-HPRIe6hmi#scrollTo=eLlnut9mkaWW

21MCI1152_Rishika

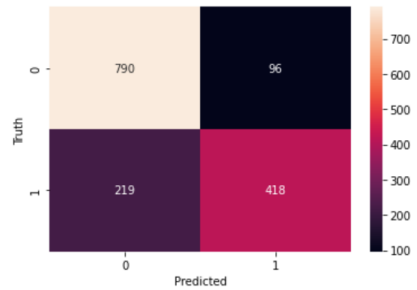
File Edit View Insert Runtime Tools Help Saving failed since 16:08

Comment Share

+ Code + Text

Connect Editin

```
    accuracy                           0.79      1523
   macro avg       0.80      0.77      0.78      1523
weighted avg       0.80      0.79      0.79      1523

Accuracy=  79.31713722915299 %

-Confusion Matrix-
```



Automatic document saving has been pending for 6 minutes. Reloading may fix the problem. Save and reload the page.

ENG 16:14 14-11-2

CHANDIGARH UNIVERSITY
Discover. Learn. Empower.
NAAC GRADE A+
Accredited University

Invitation: Appcino | WhatsApp | Welcome to Chand | CHANDIGARH UNI | 21MCI1152_Rishika | Downloads

colab.research.google.com/drive/1Zp4alr9Goj9Elo9-NSw6LF-HPRIe6hmi#scrollTo=eLlnut9mkaWW

21MCI1152_Rishika

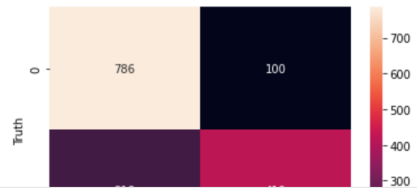
File Edit View Insert Runtime Tools Help Saving failed since 16:08

Comment Share

+ Code + Text Connect

```
NB=MultinomialNB()
classification(NB, X, y)
```

```
-Classification Report-

              precision    recall  f1-score   support

           0       0.78      0.89      0.83       886
           1       0.81      0.66      0.72       637

    accuracy                           0.79      1523
   macro avg       0.80      0.77      0.78      1523
weighted avg       0.79      0.79      0.79      1523

Accuracy=  79.12015758371635 %

-Confusion Matrix-
```



ENG

Invitation: Appcino | WhatsApp | Welcome to Chand | CHANDIGARH UNI | 21MCI1152_Rishika | Downloads

colab.research.google.com/drive/1Zp4alr9Goj9Elo9-NSw6LF-HPRIe6hmi#scrollTo=DFLSLdv7ksWH

21MCI1152_Rishika

File Edit View Insert Runtime Tools Help Saving failed since 16:08

Comment Share

+ Code + Text Connect

```
from sklearn.tree import DecisionTreeClassifier

DT= DecisionTreeClassifier(criterion = 'entropy', random_state = 0)
classification(DT, X, y)
```

```
-Classification Report-

              precision    recall  f1-score   support

           0       0.79      0.80      0.80       886
           1       0.72      0.70      0.71       637

    accuracy                           0.76      1523
   macro avg       0.75      0.75      0.75      1523
weighted avg       0.76      0.76      0.76      1523

Accuracy=  76.03414313854235 %

-Confusion Matrix-
```

ENG

Invitation: Appcino | WhatsApp | Welcome to Chand | CHANDIGARH UNI | 21MCI1152_Rishika | Downloads

colab.research.google.com/drive/1Zp4alr9Goj9Elo9-NSw6LF-HPRIe6hmi#scrollTo=kNYqZPILk99c

21MCI1152_Rishika

File Edit View Insert Runtime Tools Help Saving failed since 16:08

Comment Share

+ Code + Text Connect Editi

Predicted

```
[ ] from sklearn.linear_model import LogisticRegression

    LR = LogisticRegression()
    LR.fit(X_train, y_train)
    y_pred= LR.predict(X_test)

    print('-Classification Report-\n\n',classification_report(y_test, y_pred))
```

```
-Classification Report-

              precision    recall  f1-score   support

           0       0.78      0.91      0.84       886
           1       0.84      0.65      0.73       637

    accuracy                           0.80      1523
   macro avg       0.81      0.78      0.78      1523
weighted avg       0.80      0.80      0.79      1523
```

```
print('Accuracy= ', accuracy_score(y_test, y_pred)*100, '%\n')

print('-Confusion Matrix-\n')
```

VMware Workstation Pro

ENG 16:
14-11-

Invitation: Appcino | WhatsApp | Welcome to Chand | CHANDIGARH UNI | 21MCI1152_Rishika | Downloads

colab.research.google.com/drive/1Zp4alr9Goj9Elo9-NSw6LF-HPRIe6hmi#scrollTo=kNYqZPILk99c

21MCI1152_Rishika

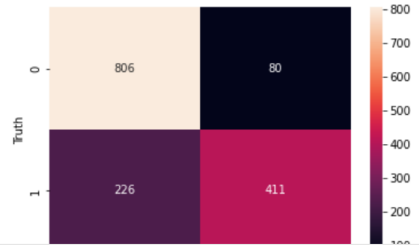
File Edit View Insert Runtime Tools Help Saving failed since 16:08

Comment

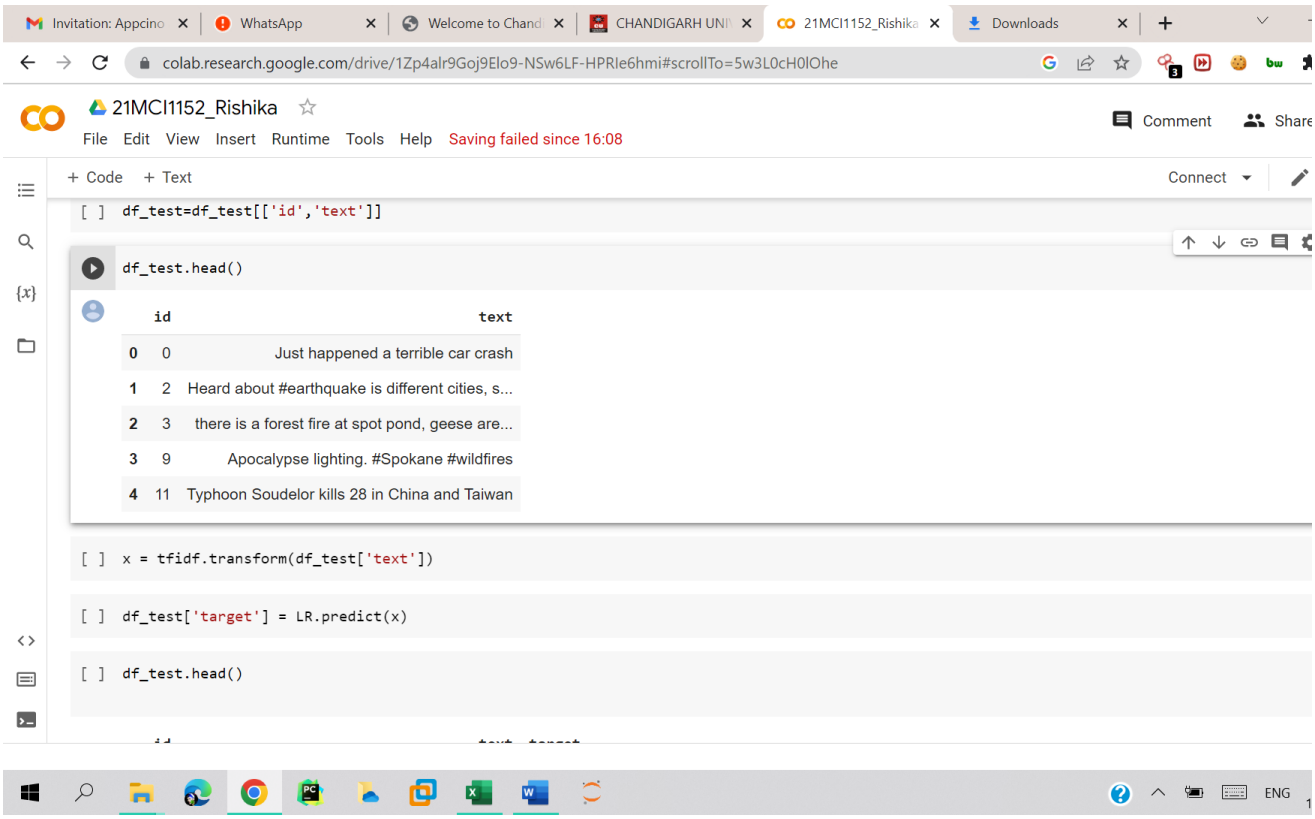
+ Code + Text Connect

```
print('Accuracy= ', accuracy_score(y_test, y_pred)*100, '%\n')

print('-Confusion Matrix-\n')
cm=confusion_matrix(y_test, y_pred)
sns.heatmap(cm, annot=True, fmt='d')
plt.xlabel('Predicted')
plt.ylabel('Truth')
```

```
Accuracy=  79.9080761654629 %

-Confusion Matrix-

Text(33.0, 0.5, 'Truth')
```



EN



Invitation: Appcino | WhatsApp | Welcome to Chand | CHANDIGARH UNI | 21MCI1152_Rishika | Downloads

colab.research.google.com/drive/1Zp4alr9Goj9Elo9-NSw6LF-HPRIe6hmi#scrollTo=5w3L0cH0lOhe

21MCI1152_Rishika

File Edit View Insert Runtime Tools Help Saving failed since 16:08

Comment Share

+ Code + Text

Connect

```
[ ] df_test=df_test[['id','text']]
```

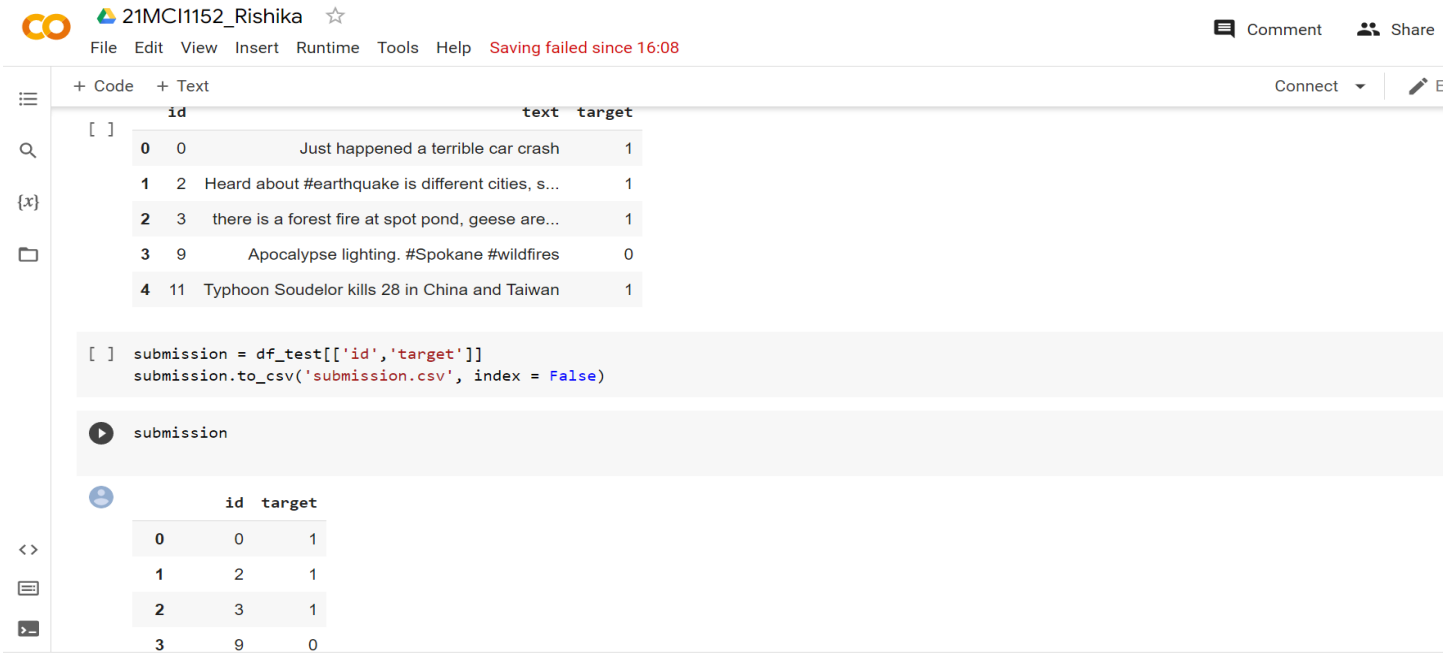
```
df_test.head()
```

	id	text
0	0	Just happened a terrible car crash
1	2	Heard about #earthquake is different cities, s...
2	3	there is a forest fire at spot pond, geese are...
3	9	Apocalypse lighting. #Spokane #wildfires
4	11	Typhoon Soudelor kills 28 in China and Taiwan

```
[ ] x = tfidf.transform(df_test['text'])
```

```
[ ] df_test['target'] = LR.predict(x)
```

```
[ ] df_test.head()
```



21MCI1152_Rishika

File Edit View Insert Runtime Tools Help Saving failed since 16:08

Comment Share

+ Code + Text

Connect

```
[ ]
```

	id	text	target
0	0	Just happened a terrible car crash	1
1	2	Heard about #earthquake is different cities, s...	1
2	3	there is a forest fire at spot pond, geese are...	1
3	9	Apocalypse lighting. #Spokane #wildfires	0
4	11	Typhoon Soudelor kills 28 in China and Taiwan	1

```
[ ] submission = df_test[['id','target']]
    submission.to_csv('submission.csv', index = False)
```

```
submission
```

	id	target
0	0	1
1	2	1
2	3	1
3	9	0

Learning outcomes (What I have learnt):

1. **I have learnt about the implementation and working of Decision Tree.**
2. **I have learnt about how to create Decision Tree Classifier and training on dataset.**
3. **I have learnt how to do prediction on validation data and check accuracy of model.**
4. **I have learnt to plot and visualize Decision Tree Classifier model.**

Evaluation Grid:

Sr. No.	Parameters	Marks Obtained	Maximum Marks
1.	Demonstration and Performance (Pre Lab Quiz)		5
2.	Worksheet		10
3.	Post Lab Quiz		5

UNIVERSITY INSTITUTE of COMPUTING
Asia's Fastest Growing University

NAAC GRADE A+ ACCREDITED UNIVERSITY

Experiment - 3.3

Student Name: Siddhant Rai Jain **UID:** 21MCI1182

Branch: MCA (AIML) **Section/Group:** 21MAM1/B

Semester: III **Date of Performance:** 10/11/22

Subject Name: Deep Learning Lab **Subject Code:** 21CAH-724

1) Task to be done:

Take a NLP data set from any online repository and use LSTM, Bi-directional LSTM and conv-1D layers for any NLP application

1. Load the data

2. Understanding the data format

3. Reprocessing the data

4. Build a model

5. Compile the model

6. Train the model

7. What is the accuracy of the model?

8 How to improve the accuracy of the model?

9. Evaluate the model

10. What results do you get with a model with Bi-directional LSTM and Conv-1D layers

2) Dataset Used:

About Data: Covid Tweets and its Sentiments from Extremely positive to negative.

Data view

UserName	ScreenNam	Location	OriginalTweet	Sentiment
3799	48751	London	@MeNyrbie @Phil_Gahan @Chrisitv https://t.co/iFz9FAn2Pa and https://t.co/xX6ghGFzCC an	Neutral
3800	48752	UK	advice Talk to your neighbours family to exchange phone numbers create contact list with pho	Positive
3801	48753	Vagabond	Coronavirus Australia: Woolworths to give elderly, disabled dedicated shopping hours amid C	Positive
3802	48754		My food stock is not the only one which is empty... PLEASE, don't panic, THERE WILL BE ENOUGH FOOD FOR EVERYONE if you do not take more than you need. Stay calm, stay safe. #COVID19france #COVID_19 #COVID19 #coronavirus #confinement #Confinementotal #ConfinementGeneral https://t.co/zrlG0Z520j	Positive
3803	48755		Me, ready to go at supermarket during the #COVID19 outbreak. Not because I'm paranoid, but because my food stock is litteraly empty. The #coronavirus is a serious thing, but please, don't panic. It causes shortage... #CoronavirusFrance #restezchezvous #StayAtHome #confinement https://t.co/usmuaLq72n	Extremely Negative
3804	48756	ÜT: 36.319	As news of the region□s first confirmed COVID-19 case came out of Sullivan County last week,	Positive
3805	48757	35.926541,	Cashier at grocery store was sharing his insights on #Covid_19 To prove his credibility he comn	Positive

3) Steps for experiment/practical:

Code

```
import pandas as pd

import numpy as np

import os


#@title Reading the data ||   Data: Covid Tweets and its Sentiments from Extremely +ve to -ve

#Loading the positive and negative tweeets

data = pd.read_csv("/content/drive/MyDrive/CU/Sem3/DL Lab/Exp3.1_covid_tweet_text_classfi_ANN/Corona_NLP_train.csv", encoding='latin-1')


print("\nLoaded Data :\n-----------------------------------")

print(data.head())
```

```
# Information on the data
data.info()

# Selecting the features and labels
x = data['OriginalTweet']
y = data['Sentiment']

# Checking the number of labels
print(y.head())
print(y.unique())

#@title Build a label encoder for target variable to convert strings to numeric values.
from sklearn import preprocessing
import tensorflow as tf

label_encoder = preprocessing.LabelEncoder()
y = label_encoder.fit_transform(y)

#Convert target to one-hot encoding vector
y = tf.keras.utils.to_categorical(y, 5)

print("One-hot Encoding Shape : ", y.shape)

#@title Word Encoding -> Tokenization -> Solving OOV Problem  -> Create Sequence
#Preprocess data for covid- tweets
from tensorflow.keras.preprocessing.text import Tokenizer
from tensorflow.keras.preprocessing.sequence import pad_sequences

#Max words in the vocabulary for this dataset
```

```
VOCAB_WORDS=10000
#Max sequence length for word sequences
MAX_SEQUENCE_LENGTH=100

#Create a vocabulary with unique words and IDs
tweets_tokenizer = Tokenizer(num_words=VOCAB_WORDS, oov_token = "<oov>")
tweets_tokenizer.fit_on_texts(x)

print("Total unique tokens found: ", len(tweets_tokenizer.word_index))
print("Example token ID for word \"safe\" :", tweets_tokenizer.word_index.get("safe"))

#Convert sentences to token-ID sequences
sequences = tweets_tokenizer.texts_to_sequences(x)

#Pad all sequences to fixed length
padded = pad_sequences(sequences, maxlen=MAX_SEQUENCE_LENGTH)

print("\nTotal sequences found : ", len(padded))
print("Example Sequence for sentence : ", x[0])
print(padded[0])

#Load the pre-trained embeddings

import numpy as np

#Read pretrained embeddings into a dictionary
glove_dict = {}

#Loading a 50 feature (dimension) embedding with 6 billion words
```

```
with open('/content/drive/MyDrive/CU/Sem3/DL Lab/Exp3.1_covid_tweet_text_classfi_ANN/glove.6B.50d.txt', "r", encoding="utf8") as glove_file:
    for line in glove_file:

        emb_line = line.split()
        emb_token = emb_line[0]
        emb_vector = np.array(emb_line[1:], dtype=np.float32)

        if emb_vector.shape[0] == 50:
            glove_dict[emb_token] = emb_vector

print("Dictionary Size: ", len(glove_dict))
print("\n Sample Dictionary Entry for word \"the\" :\n", glove_dict.get("people"))

#We now associate each token ID in our data set vocabulary to the corresponding embedding in Glove
#If the word is not available, then embedding will be all zeros.

#Matrix with 1 row for each word in the data set vocubulary and 50 features

vocab_len = len(tweets_tokenizer.word_index) + 1

embedding_matrix = np.zeros((vocab_len, 50))

for word, id in tweets_tokenizer.word_index.items():
    try:
        embedding_vector = glove_dict.get(word)
        if embedding_vector is not None:
            embedding_matrix[id] = embedding_vector
    except:
```

```
        pass

print("Size of Embedding matrix :", embedding_matrix.shape)

print("Embedding Vector for word \"safe\" : \n", embedding_matrix[tweets_tokenizer.word_index.get("safe")])


#Split into training and test data

from sklearn.model_selection import train_test_split

X_train, X_test, Y_train, Y_test = train_test_split(padded,y,test_size=0.2)
```

Model Using LSTM

```
#Create a model

from tensorflow import keras

from tensorflow.keras import optimizers

from tensorflow.keras.regularizers import l2

from keras.layers import LSTM,Dense


#Setup Hyper Parameters for building the model

NB_CLASSES=5


model = tf.keras.models.Sequential()


model.add(keras.layers.Embedding(vocab_len,
                                 50,
                                 name="Embedding-Layer",
                                 weights=[embedding_matrix],
                                 input_length=MAX_SEQUENCE_LENGTH,
                                 trainable=True))
```

```
#Add LSTM Layer

model.add(LSTM(250))

model.add(LSTM(250))

model.add(keras.layers.Flatten())


model.add(keras.layers.Dense(NB_CLASSES,

                             name='Output-Layer',

                             activation='softmax'))


model.compile(loss='categorical_crossentropy',

              metrics=['accuracy'])


model.summary()


# Model Training and Accuracy Plotting


#@title Model 1 Training || Bastch_size=256  & Epochs=10

#Make it verbose so we can see the progress

VERBOSE=1


#Setup Hyper Parameters for training

BATCH_SIZE=256

EPOCHS=10

VALIDATION_SPLIT=0.2


print("\nTraining Progress:\n----------------------------------")


history=model.fit(X_train,

          Y_train,
```

```
            batch_size=BATCH_SIZE,
            epochs=EPOCHS,
            verbose=VERBOSE,
            validation_split=VALIDATION_SPLIT)

print("\nEvaluation against Test Dataset :\n-----------------------------------")
model.evaluate(X_test,Y_test)

#@title Model 1 Training || Bastch_size=256  & Epochs=10

#Make it verbose so we can see the progress
VERBOSE=1

#Setup Hyper Parameters for training
BATCH_SIZE=256
EPOCHS=10
VALIDATION_SPLIT=0.2

print("\nTraining Progress:\n-----------------------------------")

history=model.fit(X_train,
            Y_train,
            batch_size=BATCH_SIZE,
            epochs=EPOCHS,
            verbose=VERBOSE,
            validation_split=VALIDATION_SPLIT)

print("\nEvaluation against Test Dataset :\n-----------------------------------")
model.evaluate(X_test,Y_test)
```

```
import matplotlib.pyplot as plt

# Training accuracy and validation accuracy

plt.plot(history.history["accuracy"])

plt.plot(history.history["val_accuracy"])

plt.legend(["Accuracy","Val_accuracy"])


plt.plot(history.history['accuracy'])

plt.title('model accuracy')

plt.ylabel('accuracy')

plt.xlabel('epoch')

plt.legend(['train'], loc='upper left')

plt.show()


plt.plot(history.history['loss'])

plt.title('model loss')

plt.ylabel('loss')

plt.xlabel('epoch')

plt.legend(['train'], loc='upper left')

plt.show()
```

Model Using Bi-Directional LSTM

```
 #@title Model 1

#Create a model

from keras.layers import Dense, SimpleRNN, LSTM, Bidirectional

import tensorflow as tf

#Setup Hyper Parameters for building the model

model = tf.keras.Sequential()
```

```
model.add(tf.keras.layers.Embedding(vocab_len, 50,weights=[embedding_matrix], input_length = MAX
_SEQUENCE_LENGTH))

model.add(tf.keras.layers.SpatialDropout1D(0.4))

model.add(tf.keras.layers.Bidirectional(tf.keras.layers.LSTM(196, dropout=0.05, recurrent_dropou
t=0.2)))

model.add(tf.keras.layers.Dense(5, activation='softmax'))

model.compile(loss = 'categorical_crossentropy', optimizer='adam', metrics = ['accuracy'])

model.summary()


#@title Model 1 Training || Bastch_size=400  & Epochs=10


#Make it verbose so we can see the progress

VERBOSE=1


#Setup Hyper Parameters for training

BATCH_SIZE=400

EPOCHS=10

VALIDATION_SPLIT=0.2


print("\nTraining Progress:\n-----------------------------------")


history=model.fit(X_train,

          Y_train,

          batch_size=BATCH_SIZE,

          epochs=EPOCHS,

          verbose=VERBOSE,

          validation_split=VALIDATION_SPLIT)


print("\nEvaluation against Test Dataset :\n-----------------------------------")

model.evaluate(X_test,Y_test)
```

```
import matplotlib.pyplot as plt

# Training accuracy and validation accuracy

plt.plot(history.history["accuracy"])

plt.plot(history.history["val_accuracy"])

plt.legend(["Accuracy","Val_accuracy"])


plt.plot(history.history['accuracy'])

plt.title('model accuracy')

plt.ylabel('accuracy')

plt.xlabel('epoch')

plt.legend(['train'], loc='upper left')

plt.show()


plt.plot(history.history['loss'])

plt.title('model loss')

plt.ylabel('loss')

plt.xlabel('epoch')

plt.legend(['train'], loc='upper left')

plt.show()
```

#Model Using Conv1D

```
model = tf.keras.Sequential([

    tf.keras.layers.Embedding(vocab_len, 50, weights=[embedding_matrix], input_length=MAX_SEQUEN
CE_LENGTH),

    tf.keras.layers.Conv1D(64, 5, activation='relu'),

    tf.keras.layers.GlobalAveragePooling1D(),

    tf.keras.layers.Dense(24, activation='relu'),

    tf.keras.layers.Dense(5, activation='sigmoid')
```

```
])

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


model.summary()


#@title Model 1 Training || Bastch_size=400  & Epochs=10


#Make it verbose so we can see the progress

VERBOSE=1


#Setup Hyper Parameters for training

BATCH_SIZE=400

EPOCHS=10

VALIDATION_SPLIT=0.2


print("\nTraining Progress:\n-----------------------------------")


history=model.fit(X_train,

          Y_train,

          batch_size=BATCH_SIZE,

          epochs=EPOCHS,

          verbose=VERBOSE,

          validation_split=VALIDATION_SPLIT)


print("\nEvaluation against Test Dataset :\n-----------------------------------")

model.evaluate(X_test,Y_test)


import matplotlib.pyplot as plt
```

```
def plot_graphs(history, string):
  plt.plot(history.history[string])
  plt.plot(history.history['val_'+string])
  plt.xlabel("Epochs")
  plt.ylabel(string)
  plt.legend([string, 'val_'+string])
  plt.show()

plot_graphs(history, "accuracy")
plot_graphs(history, "loss")
```

4) Output

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

```
#Loading the positive and negative tweeets
data = pd.read_csv("/content/drive/MyDrive/CU/Sem3/DL Lab/Exp3.1_covid_tweet_text_classfi_ANN/Corona_NLP_train.csv", encoding='latin-1')

print("\nLoaded Data :\n-------------------------------------")
print(data.head())
```

```
Loaded Data :
-------------------------------------
   UserName  ScreenName   Location     TweetAt  \
0      3799       48751     London  16-03-2020
1      3800       48752         UK  16-03-2020
2      3801       48753  Vagabonds  16-03-2020
3      3802       48754        NaN  16-03-2020
4      3803       48755        NaN  16-03-2020

                                       OriginalTweet           Sentiment
0  @MeNyrbie @Phil_Gahan @Chrisitv https://t.co/i...             Neutral
1  advice Talk to your neighbours family to excha...            Positive
2  Coronavirus Australia: Woolworths to give elde...            Positive
3  My food stock is not the only one which is emp...            Positive
4  Me, ready to go at supermarket during the #COV...  Extremely Negative
```

Info of the data

+ Code + Text

```
[31]  1  # Information on the data
      2  data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 41157 entries, 0 to 41156
Data columns (total 6 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   UserName       41157 non-null  int64
 1   ScreenName     41157 non-null  int64
 2   Location       32567 non-null  object
 3   TweetAt        41157 non-null  object
 4   OriginalTweet  41157 non-null  object
 5   Sentiment      41157 non-null  object
dtypes: int64(2), object(4)
memory usage: 1.9+ MB
```

List of labels

File Edit View Insert Runtime Tools Help All changes saved

- Code + Text

```
[33]  1  # Checking the number of labels
      2  print(y.head())
      3  print(y.unique())
```

```
0              Neutral
1             Positive
2             Positive
3             Positive
4    Extremely Negative
Name: Sentiment, dtype: object
['Neutral' 'Positive' 'Extremely Negative' 'Negative' 'Extremely Positive']
```

Label Encoder for converting labels through one hot encoding

File Edit View Insert Runtime Tools Help All changes saved

Code + Text

```
import tensorflow as tf

label_encoder = preprocessing.LabelEncoder()
y = label_encoder.fit_transform(y)

#Convert target to one-hot encoding vector
y = tf.keras.utils.to_categorical(y, 5)

print("One-hot Encoding Shape : ", y.shape)
```

```
One-hot Encoding Shape :  (41157, 5)
```

After word tokenizing, solving OOV and implementing pre-padding

File Edit View Insert Runtime Tools Help All changes saved

Code + Text

```
[16]
print("\nTotal sequences found : ", len(padded))
print("Example Sequence for sentence : ", x[0])
print(padded[0])
```

```
Total unique tokens found:  85199
Example token ID for word "safe" : 172

Total sequences found :  41157
Example Sequence for sentence :  @MeNyrbie @Phil_Gahan @Chrisitv https://t.co/iFz9FAn2Pa and https://t.co/xX6
[   0    0    0    0    0    0    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0    0    0    0    0    1 7251
    1    1    8    5    6    1    7    8    5    6    1    7    8    5
    6    1]
```

Embedding Matrix

e Edit View Insert Runtime Tools Help All changes saved

ode + Text

```
]
        except:
            pass

print("Size of Embedding matrix :", embedding_matrix.shape)
print("Embedding Vector for word \"safe\" : \n", embedding_matrix[tweets_tokenizer.word_index.get("safe")])
```

```
Size of Embedding matrix : (85200, 50)
Embedding Vector for word "safe" :
 [ 0.82823998 -0.38284999  0.28854001 -0.40331     0.55057001 -0.58244002
 -0.55631     0.2089      1.12779999  0.18073     0.27462    -0.27867001
  0.51273     0.01266     0.061183    1.00370002  0.79961002 -0.0035723
  0.19964001 -0.1948     -0.32393     0.67132002  0.0057699   0.35095999
 -0.018524   -1.29159999 -0.065977    0.36985999  1.44060004  0.12498
  2.7895      0.060375   -0.72425997 -0.35687     0.38117     0.012956
  0.47402999  0.23586001 -0.22132    -0.30540001 -0.058053   -0.20364
  0.91986001  0.32143     0.13541    -0.13328999 -0.82081997  0.082733
  0.60049999  0.55239999]
```

Model 1 Using LSTM

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

```
31  model.summary()
```

```
Model: "sequential_2"
_________________________________________________________________
 Layer (type)                 Output Shape              Param #
=================================================================
 Embedding-Layer (Embedding)  (None, 100, 50)           4260000

 lstm_2 (LSTM)                (None, 256)               314368

 flatten_2 (Flatten)          (None, 256)               0

 Output-Layer (Dense)         (None, 5)                 1285

=================================================================
Total params: 4,575,653
Trainable params: 4,575,653
Non-trainable params: 0
_________________________________________________________________
```

Model 1 Training and Accuracy

21MCI1033_Sanjoy_Sahani_DL_Exp3.3.ipynb ☆

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

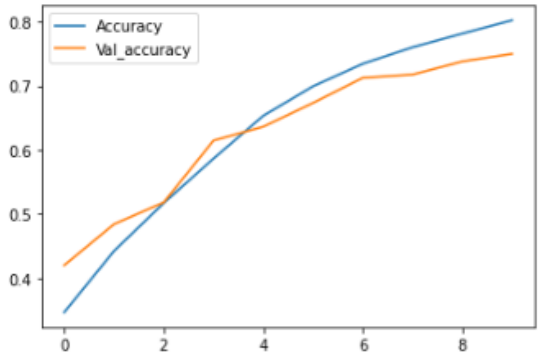
```
[45] Epoch 6/10
     103/103 [==============================] - 144s 1s/step - loss: 0.7885 - accuracy: 0.6993 - val_loss: 0.8430 - val_accuracy: 0.6733
     Epoch 7/10
     103/103 [==============================] - 146s 1s/step - loss: 0.7137 - accuracy: 0.7344 - val_loss: 0.7699 - val_accuracy: 0.7125
     Epoch 8/10
     103/103 [==============================] - 146s 1s/step - loss: 0.6590 - accuracy: 0.7602 - val_loss: 0.7727 - val_accuracy: 0.7175
     Epoch 9/10
     103/103 [==============================] - 148s 1s/step - loss: 0.6098 - accuracy: 0.7815 - val_loss: 0.7174 - val_accuracy: 0.7380
     Epoch 10/10
     103/103 [==============================] - 154s 1s/step - loss: 0.5634 - accuracy: 0.8022 - val_loss: 0.6951 - val_accuracy: 0.7499

     Evaluation against Test Dataset :
     ------------------------------------
     258/258 [==============================] - 30s 116ms/step - loss: 0.6782 - accuracy: 0.7557
     [0.6781731247901917, 0.7557094097137451]
```

Plotting Model 1 Accuracy

File Edit View Insert Runtime Tools Help All changes saved

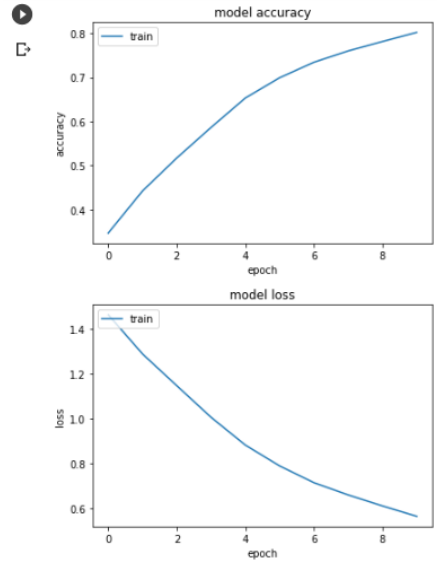
+ Code + Text

```
<matplotlib.legend.Legend at 0x7f33c34e4950>
```



File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text



Bi-Directional LSTM

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

[65]

```
Model: "sequential_3"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 embedding (Embedding)       (None, 100, 50)           4260000

 spatial_dropout1d (SpatialD  (None, 100, 50)          0
 ropout1D)

 bidirectional (Bidirectiona  (None, 392)              387296
 l)

 dense (Dense)               (None, 5)                 1965

=================================================================
Total params: 4,649,261
Trainable params: 4,649,261
Non-trainable params: 0
_________________________________________________________________
```

Model 2 Training

File Edit View Insert Runtime Tools Help All changes saved

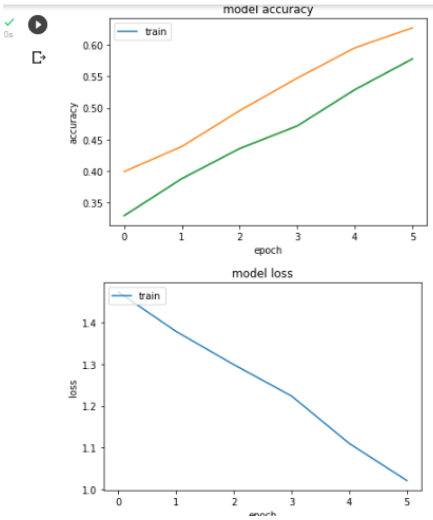
+ Code + Text

```
[67] Training Progress:
     ----------------------------------
     Epoch 1/6
     66/66 [==============================] - 286s 4s/step - loss: 1.4735 - accuracy: 0.3294 - val_loss: 1.3644 - val_accuracy: 0.3992
     Epoch 2/6
     66/66 [==============================] - 282s 4s/step - loss: 1.3789 - accuracy: 0.3881 - val_loss: 1.2828 - val_accuracy: 0.4390
     Epoch 3/6
     66/66 [==============================] - 282s 4s/step - loss: 1.2992 - accuracy: 0.4356 - val_loss: 1.1718 - val_accuracy: 0.4957
     Epoch 4/6
     66/66 [==============================] - 285s 4s/step - loss: 1.2239 - accuracy: 0.4715 - val_loss: 1.0817 - val_accuracy: 0.5473
     Epoch 5/6
     66/66 [==============================] - 280s 4s/step - loss: 1.1099 - accuracy: 0.5290 - val_loss: 0.9775 - val_accuracy: 0.5953
     Epoch 6/6
     66/66 [==============================] - 280s 4s/step - loss: 1.0206 - accuracy: 0.5778 - val_loss: 0.9235 - val_accuracy: 0.6270

     Evaluation against Test Dataset :
     ----------------------------------
     258/258 [==============================] - 29s 113ms/step - loss: 0.9125 - accuracy: 0.6376
     [0.9124816060066223, 0.6376336216926575]
```

Model 2 accuracy

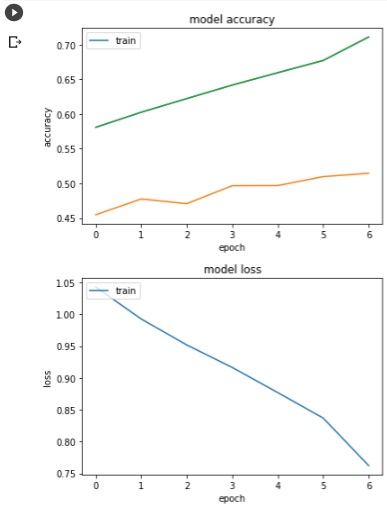
File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text



File Edit View Insert Runtime Tools Help All changes saved

Code + Text



#Conv1D Model

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

```
8  model.compile(loss='binary_crossentropy',optimizer='adam',metrics=['accuracy'])
9
10 model.summary()
```

```
Model: "sequential_6"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 embedding_3 (Embedding)     (None, 100, 50)           4260000

 conv1d_2 (Conv1D)           (None, 96, 64)            16064

 global_average_pooling1d_2  (None, 64)                0
 (GlobalAveragePooling1D)

 dense_5 (Dense)             (None, 24)                1560

 dense_6 (Dense)             (None, 5)                 125

=================================================================
Total params: 4,277,749
Trainable params: 4,277,749
Non-trainable params: 0
_________________________________________________________________
```

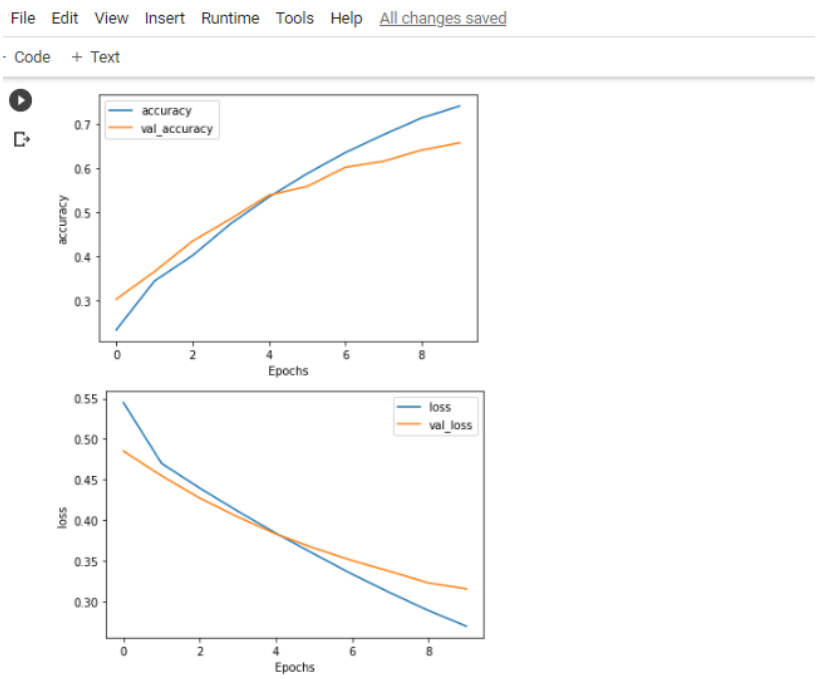
Model 2 Training and Accuracy

File Edit View Insert Runtime Tools Help All changes saved

Code + Text

```
75] 66/66 [==============================] - 10s 154ms/step - loss: 0.3102 - accuracy: 0.6754 - val_loss: 0.3366 - val_accuracy: 0.6155
    Epoch 9/10
    66/66 [==============================] - 10s 154ms/step - loss: 0.2885 - accuracy: 0.7133 - val_loss: 0.3225 - val_accuracy: 0.6407
    Epoch 10/10
    66/66 [==============================] - 10s 155ms/step - loss: 0.2690 - accuracy: 0.7402 - val_loss: 0.3151 - val_accuracy: 0.6571

    Evaluation against Test Dataset :
    ------------------------------------
    258/258 [==============================] - 1s 4ms/step - loss: 0.3103 - accuracy: 0.6656
    [0.31030037999153137, 0.6655733585357666]
```

Model 3 Trainnig and Accuracy



Summary of the worksheet

Among the all the three model of LSTM, Bi-Directional LSTM and Conv1D model I got the accuracy of 75%, 63% and 66%, though I trained Bi-Directional Model with less epochs then rest, otherwise it might show the good accuracy then the rest.

5) Learning outcomes (What I have learnt):

1. Learn the concept and implementation word Encoding, Tokenization, text to Sequence, Padding and Embedding matrix.

2. Learn the concept of NLP and text classification.

3. Learn to implement and train model for text classification using LSTM and Bi-directional LSTM and Conv1D.