National College of Ireland

Configuration Manual

Public perception of the “Digital Rupee” in India

MSc in Fintech

Aditya Kulkarni
Student ID: X20260326

School of Computing
National College of Ireland

Supervisor: Victor del Rosal

National College of Ireland
MSc Project Submission Sheet
School of Computing

Student Name: Aditya Kulkarni

Student ID: X20260326

Program: MSc in Fintech **Year:** 2022

Supervisor: Victor del Rosal

Due Date: 15th AUG 2022

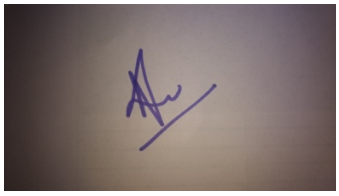
Project Title: Public perception of the "Digital Rupee" in India

Word Count: 518 Words

Page Count: 12 Pages

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project. ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature:



Date: 15th Aug 2022

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST

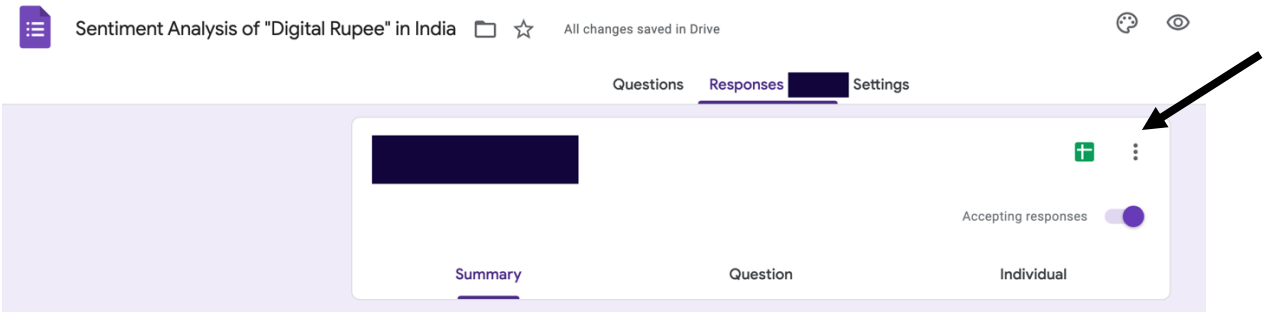
Attach a completed copy of this sheet to each project (including multiple copies)	□
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	□
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	□

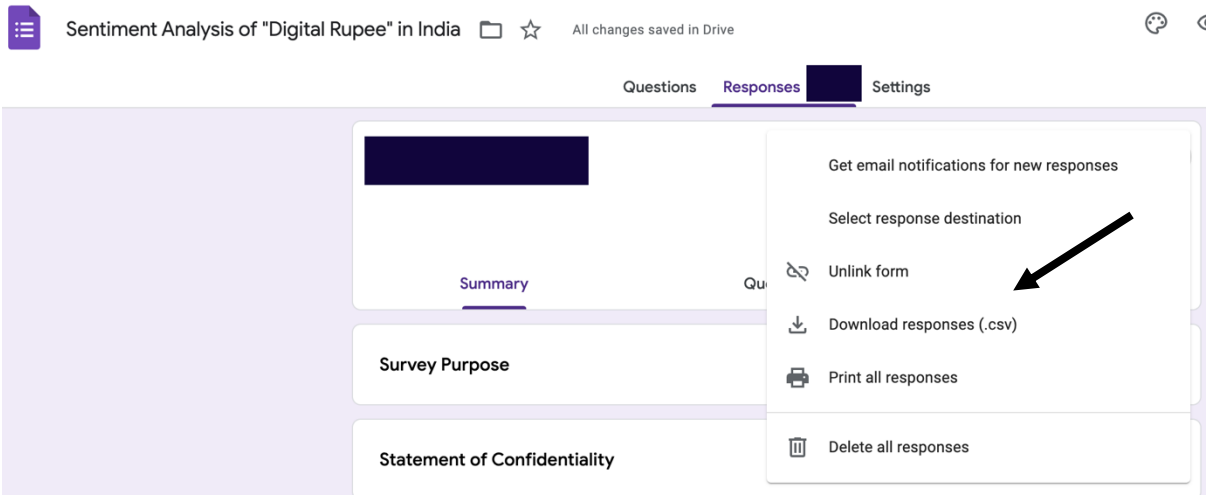
1. Introduction

This configuration manual describes the steps the research has done to perform sentimental analysis from two different data sources. The primary data was collected from Survey and subjected to sentiment analysis using Microsoft Azure Machine Learning and the secondary data was collected from twitter and subjected to sentiment analysis using python in Jupiter Notebook on Anaconda Navigator. **Please zoom into the document for better viewing.*

2. Sentiment Analysis using Microsoft Azure Machine Learning

2.1 Extracting the data from Google Forms

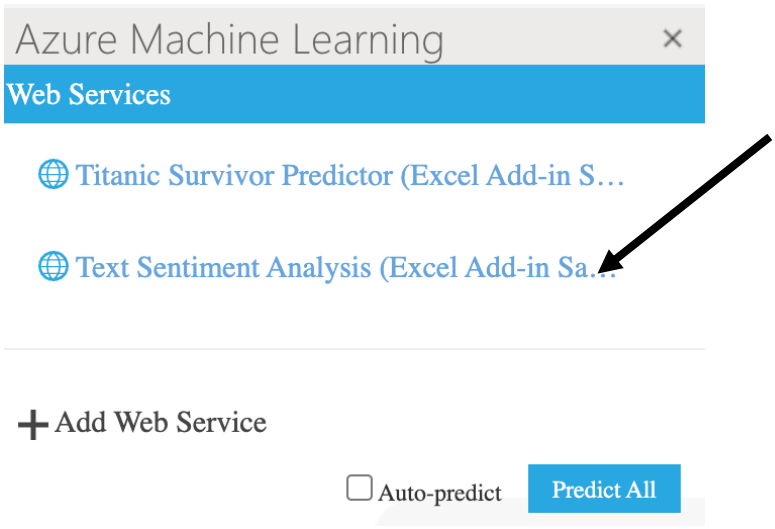




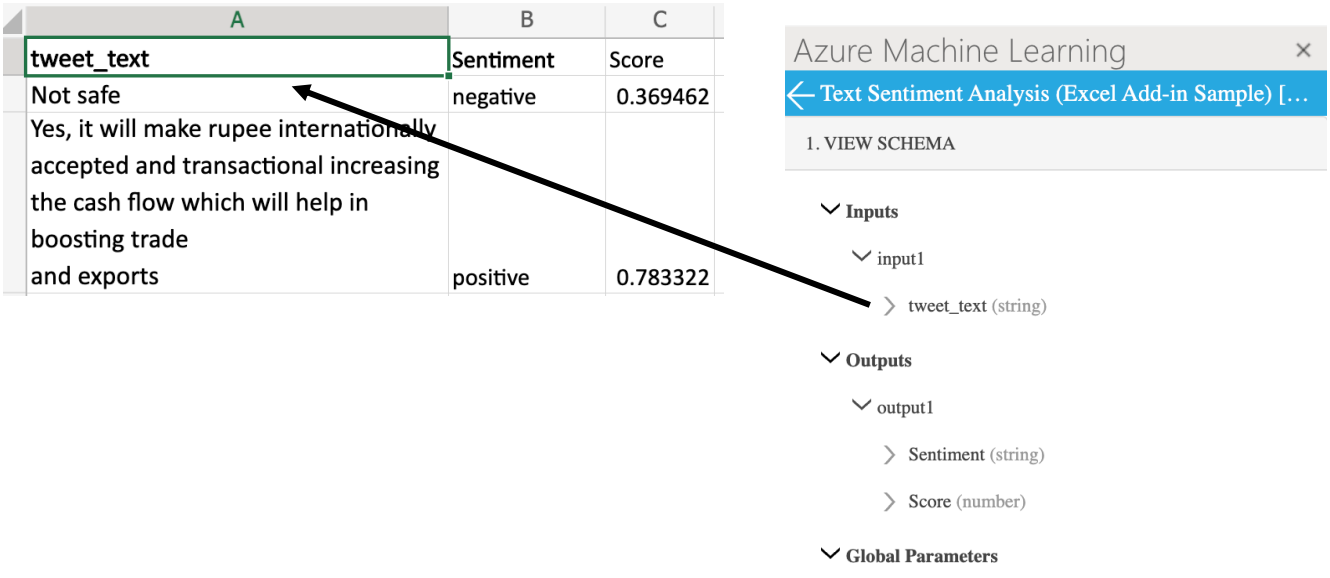
2.2 Adding Microsoft Azure Machine Learning

Add MS Azure Machine Learning via Office Add-ins

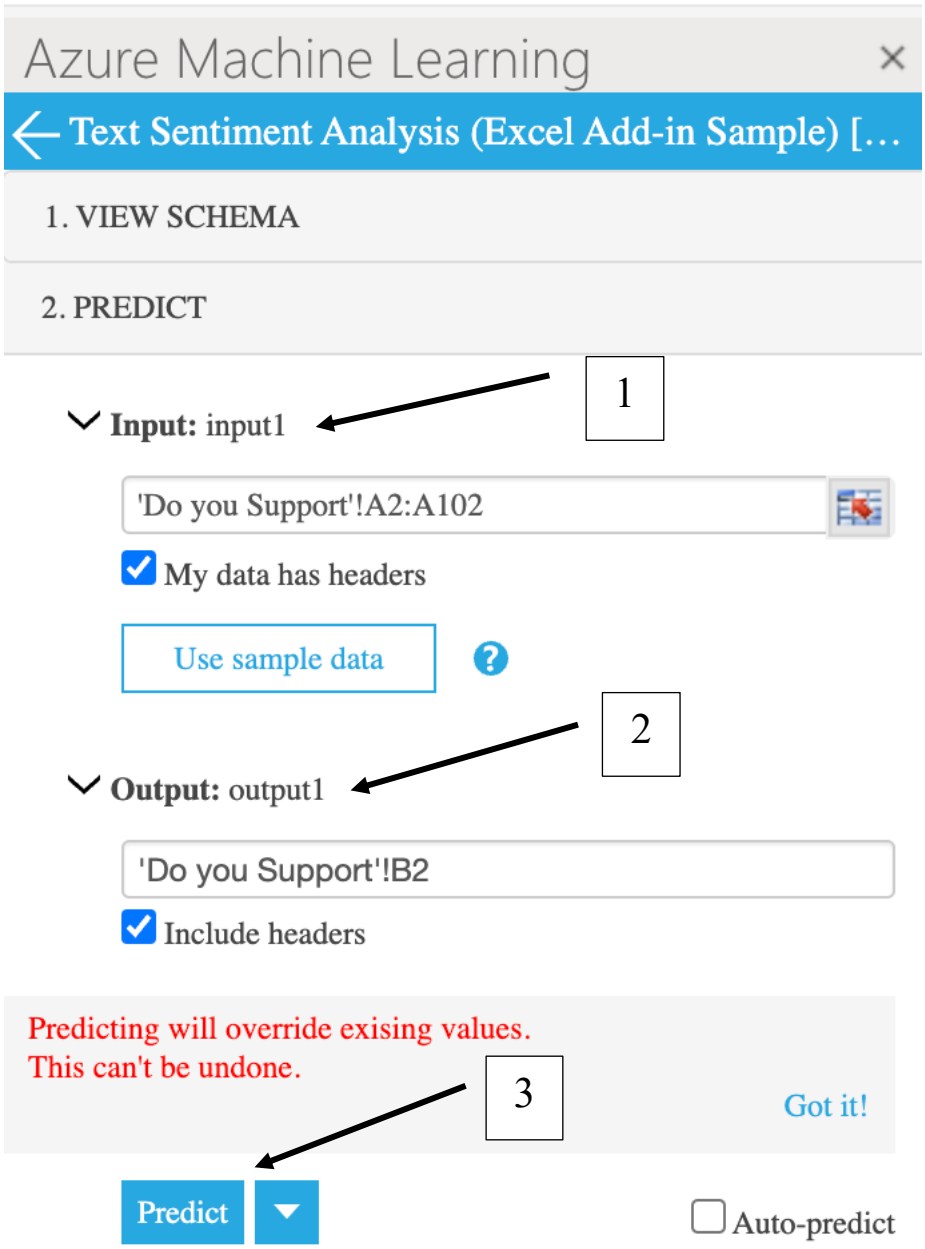
2.2 Configuring MS Azure Machine Learning



Rename Column header to tweet_text



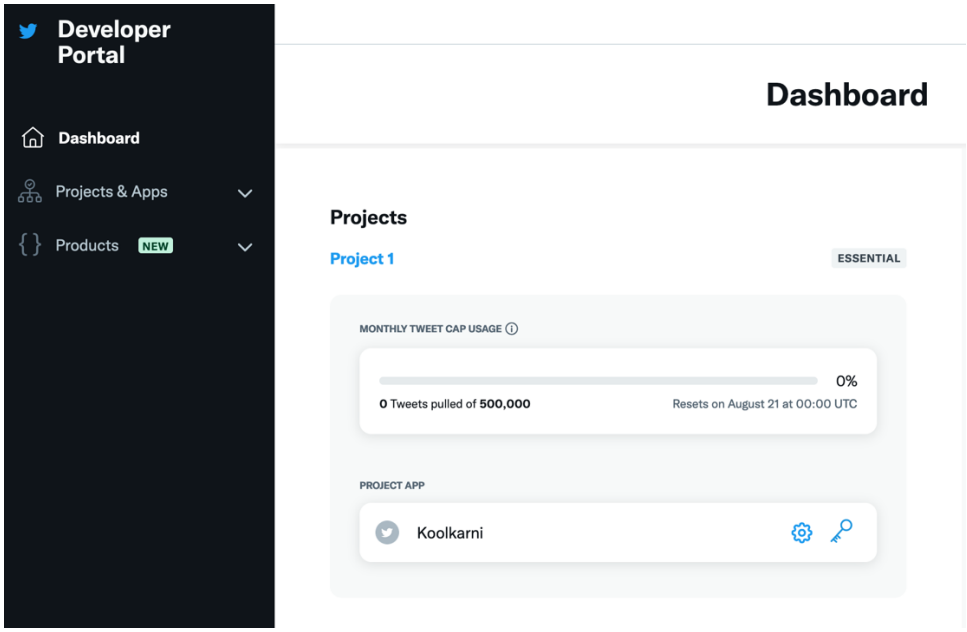
2.3 Perform Sentiment Analysis



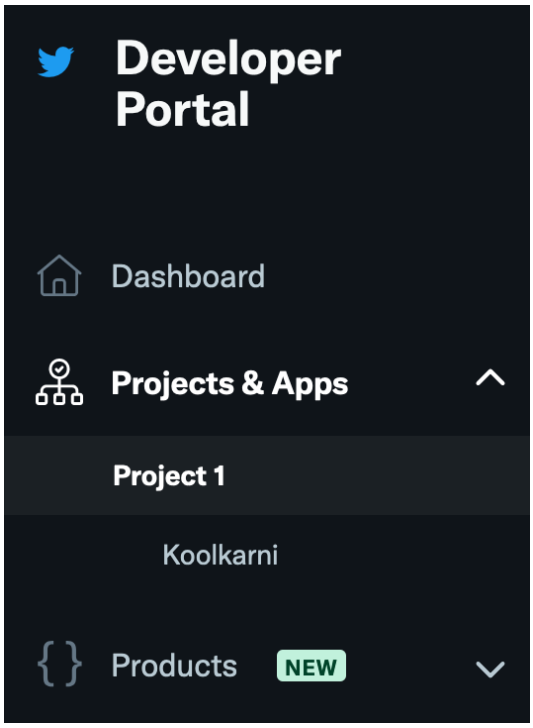
3. Sentiment Analysis using Python

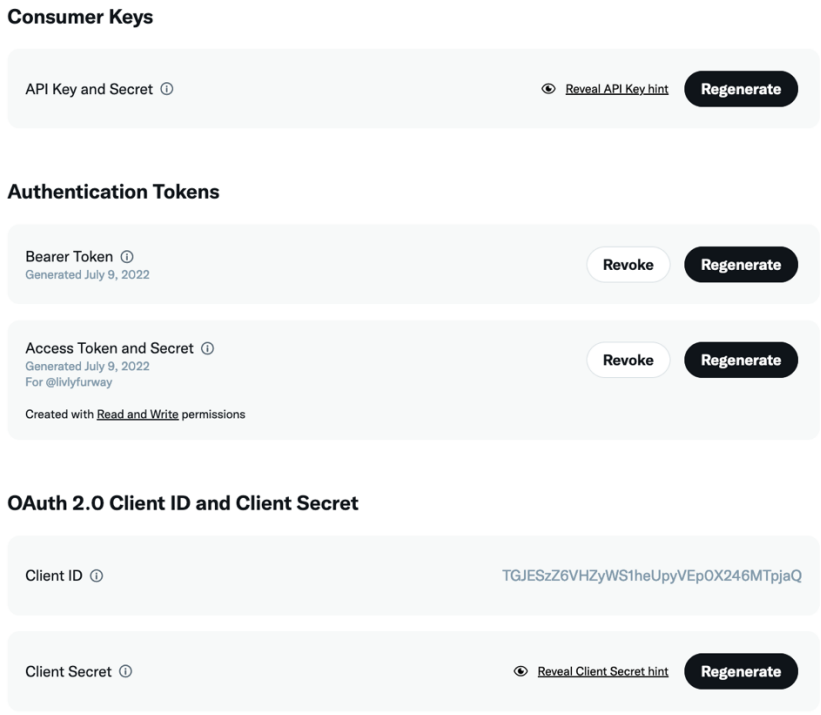
3.1 Signing up for Twitter API

https://developer.twitter.com/en/support/twitter-api/developer-accountb

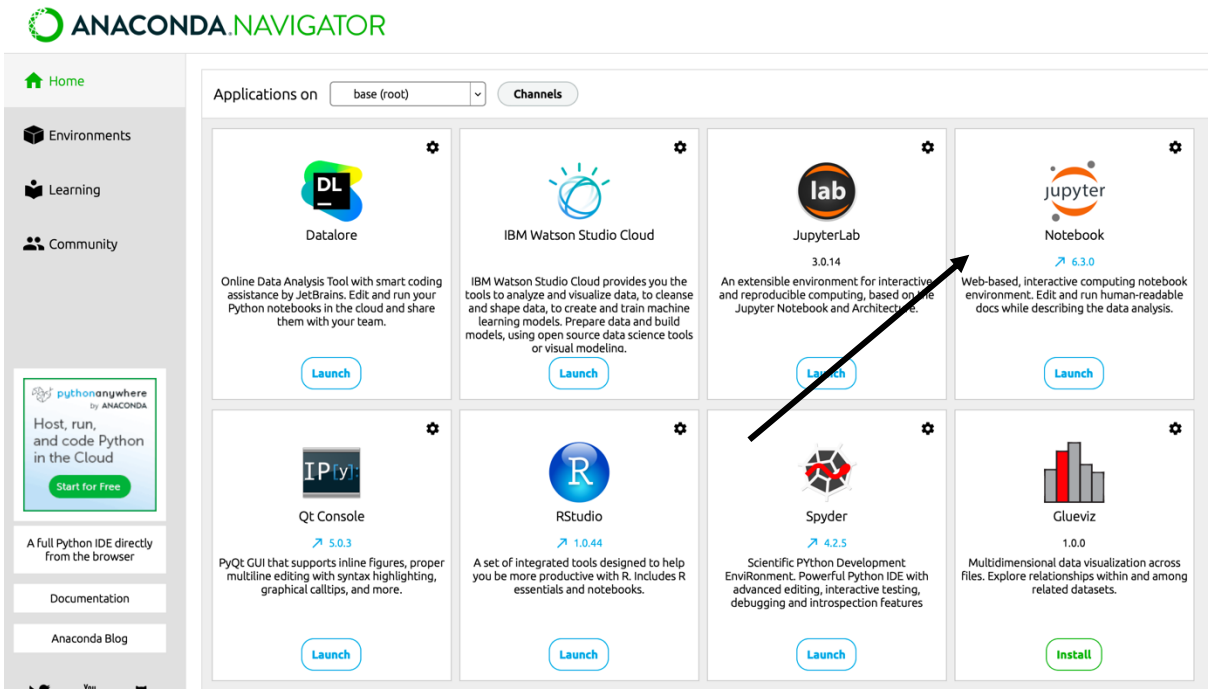


3.2 Creating projects and generating API Keys & Secrets





3.2 Downloading and installing Anaconda Navigator and using Jupiter Notebook please refer: https://docs.anaconda.com/anaconda/install/



3.3 Extracting Data from Twitter

```
import pandas as pd
import tweepy

# function to display data of each tweet
def printtweetdata(n, ith_tweet):
        print()
        print(f"Tweet {n}:")
        print(f"Username:{ith_tweet[0]}")
        print(f"Description:{ith_tweet[1]}")
        print(f"Location:{ith_tweet[2]}")
        print(f"Following Count:{ith_tweet[3]}")
        print(f"Follower Count:{ith_tweet[4]}")
        print(f"Total Tweets:{ith_tweet[5]}")
        print(f"Retweet Count:{ith_tweet[6]}")
        print(f"Tweet Text:{ith_tweet[7]}")
        print(f"Hashtags Used:{ith_tweet[8]}")


# function to perform data extraction
def scrape(words, date_since, numtweet):

        # Creating DataFrame using pandas
        db = pd.DataFrame(columns=['username',
                                   'description',
                                   'location',
                                   'following',
                                   'followers',
                                   'totaltweets',
                                   'retweetcount',
                                   'text',
                                   'hashtags'])

        # We are using .Cursor() to search
        # through twitter for the required tweets.
        # The number of tweets can be
        # restricted using .items(number of tweets)
        tweets = tweepy.Cursor(api.search_tweets,
                               words, lang="en",
                               since_id=date_since,
                               tweet_mode='extended').items(numtweet)
```

```
  # Enter Hashtag and initial date
  print("Enter Twitter HashTag to search for")
  words = input()
  print("Enter Date since The Tweets are required in yyyy-mm--dd")
  date_since = input()

  # number of tweets you want to extract in one run
  numtweet = 100
  scrape(words, date_since, numtweet)
  print('Scraping has completed!')
```

3.4 Saving the data collected from twitter to csv

```
        # .Cursor() returns an iterable object. Each item in
        # the iterator has various attributes
        # that you can access to
        # get information about each tweet
        list_tweets = [tweet for tweet in tweets]

        # Counter to maintain Tweet Count
        i = 1

        # we will iterate over each tweet in the
        # list for extracting information about each tweet
        for tweet in list_tweets:
                username = tweet.user.screen_name
                description = tweet.user.description
                location = tweet.user.location
                following = tweet.user.friends_count
                followers = tweet.user.followers_count
                totaltweets = tweet.user.statuses_count
                retweetcount = tweet.retweet_count
                hashtags = tweet.entities['hashtags']

                # Retweets can be distinguished by
                # a retweeted_status attribute,
                # in case it is an invalid reference,
                # except block will be executed
                try:
                        text = tweet.retweeted_status.full_text
                except AttributeError:
                        text = tweet.full_text
                hashtext = list()
                for j in range(0, len(hashtags)):
                        hashtext.append(hashtags[j]['text'])

                # Here we are appending all the
                # extracted information in the DataFrame
                ith_tweet = [username, description,
                             location, following,
                             followers, totaltweets,
                             retweetcount, text, hashtext]
                db.loc[len(db)] = ith_tweet

                # Function call to print tweet data on screen
                printtweetdata(i, ith_tweet)
                i = i+1
        filename = '#digital money.csv'

        # we will save our database as a CSV file.
        db.to_csv(filename)
```

3.5 Authorization using the keys generated via Twitter API

```
# Enter your own credentials obtained
# from your developer account
consumer_key = "Enter consumer_key"
consumer_secret = "Enter consumer-secret"
access_key = "Enter access_key"
access_secret = "Enter access-secret"


auth = tweepy.OAuthHandler(consumer_key, consumer_secret)
auth.set_access_token(access_key, access_secret)
api = tweepy.API(auth)
```

3.6 Importing the csv as data frames (*structure of data frame is what the data columns defined in sec 3.4)

```
import pandas as pd
df1=pd.read_csv('#digitalrupee.csv')
df2=pd.read_csv('#digital rupee.csv')
df3=pd.read_csv('#digital money.csv')
df4=pd.read_csv('#digitalmoney.csv')
df5=pd.read_csv('#digital currency.csv')
df6=pd.read_csv('#digitalcurrency.csv')
```

3.7 Printing the shape of the data frames (i.e. rows and columns)

```
# Printing the shape of the each dataset
```

```
print("Shape of first data",df1.shape)
print("Shape of second data",df2.shape)
print("Shape of third data",df3.shape)
print("Shape of fourth data",df4.shape)
print("Shape of fifth data",df5.shape)
print("Shape of sixth data",df6.shape)
```

3.8 Merging all the data frames for one data source

```
]: #Merging all the Data Frames

]: frames = [df1, df2, df3 , df4 ,df5 ,df6]

   main_data = pd.concat(frames)

]: print("The final data has a shape of",main_data.shape)

]: print("The name of columns are ",main_data.columns)
```

3.9 Installing all the required NLP Libraries

```
]: !! pip install textblob
```

```
]: !! pip install langdetect
```

```
]: #Importing all libraries for NLP
from textblob import TextBlob
import sys
import tweepy
import matplotlib.pyplot as plt
import pandas as pd
import numpy as np
import os
import nltk
import pycountry
import re
import string
from wordcloud import WordCloud, STOPWORDS
from PIL import Image
from nltk.sentiment.vader import SentimentIntensityAnalyzer
from langdetect import detect
from nltk.stem import SnowballStemmer
from nltk.sentiment.vader import SentimentIntensityAnalyzer
from sklearn.feature_extraction.text import CountVectorizer
```

```
]: #Lexicon Dictionary
import nltk
nltk.download('vader_lexicon')
```

3.10 Sentiment analysis by testing polarity

```
## Sentiment Analysis

def percentage(part,whole):
    return 100 * float(part)/float(whole)
#keyword = input("Please enter keyword or hashtag to search: ")
#noOfTweet = int(input ("Please enter how many tweets to analyze: "))
#tweets = tweepy.Cursor(api.search, q=keyword).items(noOfTweet)
positive = 0
negative = 0
neutral = 0
polarity = 0
tweet_list = []
neutral_list = []
negative_list = []
positive_list = []
for tweet in main_data["text"]:
    tweet_list.append(tweet)
    analysis = TextBlob(tweet)
    score = SentimentIntensityAnalyzer().polarity_scores(tweet)
    neg = score['neg']
    neu = score['neu']
    pos = score['pos']
    comp = score['compound']
    polarity += analysis.sentiment.polarity

    if neg > pos:
        negative_list.append(tweet)
        negative += 1
    elif pos > neg:
        positive_list.append(tweet)
        positive += 1

    elif pos == neg:
        neutral_list.append(tweet)
        neutral += 1

        ## TO TAKE OUT THE PERCENTAGES OF POSITIVE/ NEGATIVE/ NEUTRAL TWEETS
positive = percentage(positive, 327)
negative = percentage(negative, 327)
neutral = percentage(neutral, 327)
polarity = percentage(polarity, 327)
positive = format(positive, '.1f')
negative = format(negative, '.1f')
neutral = format(neutral, '.1f')
```

3.11 Calculating percentage of positive/negative/neutral tweets

```
#Calculating the percentage
def count_values_in_column(data,feature):
    total=data.loc[:,feature].value_counts(dropna=False)
    percentage=round(data.loc[:,feature].value_counts(dropna=False,normalize=True)*100,2)
    return pd.concat([total,percentage],axis=1,keys=['Total','Percentage'])
```

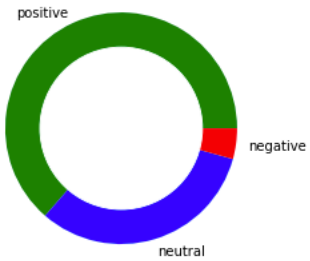
```
#Count_values for sentiment
count_values_in_column(tw_list,"sentiment")
```

	Total	Percentage
positive	208	63.61
neutral	105	32.11
negative	14	4.28

3.12 Creating a pie chart for positive/negative/neutral tweets

```
# create data for Pie Chart
pc = count_values_in_column(tw_list,"sentiment")
names= pc.index
size=pc["Percentage"]

# Create a circle for the center of the plot
my_circle=plt.Circle( (0,0), 0.7, color='white')
plt.pie(size, labels=names, colors=['green','blue','red'])
p=plt.gcf()
p.gca().add_artist(my_circle)
plt.show()
```



3.11 Cleaning the data further for word cloud

```
## Data Cleaning for Word Cloud
main_data["text"] =tweet_list
#tweet_list.drop_duplicates(inplace = True)
tw_list = pd.DataFrame(tweet_list)
tw_list['text'] = tw_list[0]
#Removing RT, Punctuation etc

#Removing RT, Punctuation etc
remove_rt = lambda x: re.sub('RT @\w+: ',"  ",x)
rt = lambda x: re.sub("(@[A-Za-z0-9]+)|([^0-9A-Za-z \t])|(\w+:\/\/\S+)","  ",x)
tw_list["text"] = tw_list.text.map(remove_rt).map(rt)
tw_list["text"] = tw_list.text.str.lower()
tw_list.head(10)
```

3.12 Extracting data for different word cloud.

```
#data extract for different word cloud
tw_list_negative = tw_list[tw_list["sentiment"]=="negative"]
tw_list_positive = tw_list[tw_list["sentiment"]=="positive"]
tw_list_neutral = tw_list[tw_list["sentiment"]=="neutral"]
```

3.13 Creating word cloud *(*Kindly make sure you have an reference image stored on the location – e.g. – "cloud.png"*

```
def create_wordcloud(text):
    mask = np.array(Image.open("cloud.png"))
    stopwords = set(STOPWORDS)
    wc = WordCloud(background_color="white",
                   mask = mask,
                   max_words=3000,
                   stopwords=stopwords,
                   repeat=True)
    wc.generate(str(text))
    wc.to_file("wc.png")
    print("Word Cloud Saved Successfully")
    path="wc.png"
    display(Image.open(path))
```