Assignment 1

Assumptions

- Whenever we are asked for total and the question **does not specify** that consider “only states” we have used **“tt”** variable available in the dataset, which gives the total sum of cases for a particular day.
- Delhi is considered as a state.
- Chandigarh is considered to have code **“ch”**
- The dictionary used for the mapping state codes to their names is shown below.
- {'ap':"Andhra Pradesh",'ar':"Arunachal Pradesh", 'as':"Assam", 'br':"Bihar", 'ct':"Chhattisgarh", 'dl':"Delhi", 'ga':"Goa", 'gj':"Gujarat", 'hp':"Himachal Pradesh", 'hr':"Haryana", 'jh':"Jharkhand", 'ka':"Karnataka",'kl':"Kerala", 'mh':"Maharashtra", 'ml':"Meghalaya", 'mn':"Manipur", 'mp':"Madhya Pradesh", 'mz':"Mizoram", 'nl':"Nagaland", 'or':"Odisha", 'pb':"Punjab", 'rj':"Rajasthan", 'sk':"Sikkim", 'tg':"Telangana",'tn':"Tamil Nadu", 'tr':"Tripura", 'up':"Uttar Pradesh", 'ut':"Uttarakhand", 'wb':"WestBengal"}
- For all subparts of question 1 and 2, the start date should be less than the end date. If the start date is more than the end date, then a statement is printed indicating the same.
- The date input format is “YYYY:MM:DD”. If a particular date is not in the given format then a message is printed indicating what format needs to be used.
- If either of the start date or end date is not present in the data frame then we display the message that the date is out of range, please enter a date in range.

Methodology

1_1.

Question Specific assumptions

➔ Used “tt” variable available in the data set for calculations. “tt” total cases for a particular day, including all the states, union territories and unassigned cases.

Methodology

➔ Validate the date according to the points mentioned in assumptions.(Took help from datetime library)

➔ If the dates are valid, called the Data_frame(json_file_path,start_date,end_date) function.

- ◆ Data_frame(json_file_path,start_date,end_date) function takes the the json file path, start date and end date as parameter.
  - It checks if the start date and end date exist in the json file or not.
    - If not marks a dateflag as false.
  - It converts the json file into a dataframe and return the data for all the dates between start date and end date and the date flag

2018348
Navneet Agarwal

2018378
Aditya SIngh

- ➔ If the dateflag returned from Data_frame is false then we return the flow back to main function.
- ➔ If the dateflag is true then we have data for all the dates between start and end date.
- ➔ We use get_sorted(DataFrame, “tt”) function to extract out the values of “tt” attribute.
    - ◆ get_sorted(df,tt) : function takes the dataframe and a string as inputs. The string represents the the stated code, U.T. code or Total sum code.
        - This function drops creates three temporary dataframes for confirmed , recovered and deceased.
        - The code attribute stores the number of confirmed ,recovered and deceased cases in temporary dataframes for confirmed , recovered and deceased respectively corresponding to each date.
        - These dataframes are then finally merged into one final dataframe.The merging is done on date attribute.
        - The final dataframe contains 4 attributes now : date , Confirmed , Recovered and Deceased.
        - This function returns this data frame.
- ➔ After we have the data frame we can individually sum the confirmed, recovered and deceased case [Done using dataframe inbuilt function], print the output and return the required values.

1_2.

Methodology

- ➔ Validate the date according to the points mentioned in assumptions.(Took help from datetime library)
- ➔ If the dates are valid, called the Data_frame(json_file_path,start_date,end_date) function.
- ➔ If the dateflag returned from Data_frame is false then we return the flow back to main function.
- ➔ If the dateflag is true then we have data for all the dates between start and end date.
- ➔ We use get_sorted(DataFrame, “dl”) function to extract out the values of state Delhi.
- ➔ The final dataframe contains 4 attributes now : date , Confirmed , Recovered and Deceased for state delhi.
- ➔ After we have the data frame we can individually sum the confirmed, recovered and deceased cases and print the output.

1_3.

Methodology

- ➔ Validate the date according to the points mentioned in assumptions.(Took help from datetime library)
- ➔ If the dates are valid, called the Data_frame(json_file_path,start_date,end_date) function.
- ➔ If the dateflag returned from Data_frame is false then we return the flow back to main function.
- ➔ If the dateflag is true then we have data for all the dates between start and end date.
- ➔ We use get_sorted(DataFrame, “dl”) function to extract out the values of state Delhi.
- ➔ The final dataframe contains 4 attributes now : date , Confirmed , Recovered and Deceased for delhi.
- ➔ We use get_sorted(DataFrame, “mh”) function to extract out the values of state Maharashtra.
- ➔ The second final dataframe contains 4 attributes now : date , Confirmed , Recovered and Deceased for delhi.
- ➔ After we have the data frame we can individually sum the confirmed, recovered and deceased cases for delhi and maharashtra separately.
- ➔ Finally we add values for both delhi and maharashtra and return the required values.

1_4.

Question Specific assumptions

- ➔ All the valid states taken into consideration are mentioned in the assumptions above.
- ➔ Instead of printing state code , we have printed state name.

Methodology

- ➔ Validate the date according to the points mentioned in assumptions.(Took help from datetime library)
- ➔ If the dates are valid, called the Data_frame(json_file_path,start_date,end_date) function.
- ➔ If the dateflag returned from Data_frame is false then we return the flow back to main function.
- ➔ If the dateflag is true then we have data for all the dates between start and end date.
- ➔ Now we use get_states( ) function on our dataframe.
  - ◆ get_states(dataframe):
    - Removes all the union territories , “tt” and “un” attributes from dataframe.['jk','an','ld','ch','dn','la','py','tt','un','dd'] were removed.
    - Add a new attribute “tt” which includes the sum of cases of all the valid states.
    - Returns the modified dataframe.

- ➔ Once we have the data frame for all the valid states, we make three different dataframes separately for confirmed, recovered and deceased.
- ➔ We call GetValueMax( ) for confirmed dataframe.
  - ◆ GetValueMax(dataframe):
    - Finds the indexes for maximum cases in the given dataset.
    - If it is a index is unique then returns that list with that index, the dataframe with the name of the states and the dataframe containing the total value of cases for each.
    - Else return the list of indices with max. cases, the the dataframe with the name of the states and the dataframe containing the total value of cases for each.
- ➔ Using the above three list to print the most affected areas for confirmed.
- ➔ Similar procedure applied for recovered and deceased.

1_5.

Question Specific assumptions

- ➔ All the valid states taken into consideration are mentioned in the assumptions above.
- ➔ Instead of printing state code , we have printed state name.

Methodology

- ➔ Validate the date according to the points mentioned in assumptions.(Took help from datetime library)
- ➔ If the dates are valid, called the Data_frame(json_file_path,start_date,end_date) function.
- ➔ If the dateflag returned from Data_frame is false then we return the flow back to main function.
- ➔ If the dateflag is true then we have data for all the dates between start and end date.
- ➔ Now we use get_states( ) function on our dataframe.
- ➔ Once we have the data frame for all the valid states, we make three different dataframes separately for confirmed, recovered and deceased.
- ➔ We call GetValueMin( ) for confimed.
  - ◆ GetValueMin(dataframe):
    - Finds the indexes for Minimum cases in the given dataset.
    - If it is a index is unique then returns that list with that index, the dataframe with the name of the states and the dataframe containing the total value of cases for each.
    - Else return the list of indices with min. cases, the the dataframe with the name of the states and the dataframe containing the total value of cases for each.
- ➔ Using the above three list to print the least affected areas for confirmed.
- ➔ Similar procedure applied for recovered and deceased.

1_6.

Question Specific assumptions

- ➔ HIghestSpike is considered to be the day with most number of cases.
- ➔ Assuming that there is one unique highest spike day for each confirmed ,deceased and recovered case.

Methodology

- ➔ Validate the date according to the points mentioned in assumptions.(Took help from datetime library)
- ➔ If the dates are valid, called the Data_frame(json_file_path,start_date,end_date) function.
- ➔ If the dateflag returned from Data_frame is false then we return the flow back to main function.
- ➔ If the dateflag is true then we have data for all the dates between start and end date.
- ➔ We use get_sorted(DataFrame, “dl”) function to extract out the values of state Delhi.
- ➔ The final dataframe contains 4 attributes now : date , Confirmed , Recovered and Deceased for state delhi.
- ➔ After we have the data frame we can use pandas to find the index of the day with maximum cases separately for confirmed, deceased and recovered.
- ➔ Using these indexes to print the days and count of the highest spike for confirmed, recovered and deceased.

1_7

Question Specific assumptions

- ➔ All the valid states taken into consideration are mentioned in the assumptions above.
- ➔ Instead of printing state code , we have printed state name.

Methodology

- ➔ Validate the date according to the points mentioned in assumptions.(Took help from datetime library)
- ➔ If the dates are valid, called the Data_frame(json_file_path,start_date,end_date) function.
- ➔ If the dateflag returned from Data_frame is false then we return the flow back to main function.
- ➔ If the dateflag is true then we have data for all the dates between start and end date.
- ➔ Now we use get_states( ) function on our dataframe.
- ➔ Once we have the data frame for all the valid states, we make three different dataframes separately for confirmed, recovered and deceased for each state.
- ➔ Individually sum the values of the three dataframes rowise to get total confirmed,recoved and deceased.

- ➔ Finally subtracting the summation of total recovered and deceased from the total confirmed to get the total number of active cases for each state, and printing these values for each state.

2_1

Question Specific assumptions

- ➔ Used “tt” variable available in the data set for calculations. “tt” total cases for a particular day, including all the states, union territories and unassigned cases.

Methodology

- ➔ Validate the date according to the points mentioned in assumptions.(Took help from datetime library)
- ➔ If the dates are valid, called the Data_frame(json_file_path,start_date,end_date) function.
- ➔ If the dateflag returned from Data_frame is false then we return the flow back to main function.
- ➔ If the dateflag is true then we have data for all the dates between start and end date.
- ➔ We use get_sorted(DataFrame, “tt”) function to extract out the values of “tt” attribute.
- ➔ Using the final data frame returned from get_sorted( ),we use cumsum() function on the confirmed recovered and deceased attributes of data frame to modify the count of cases to cumulative count.
- ➔ Finally used the modified final dataframe for the area plot.

2_2

Methodology

- ➔ Validate the date according to the points mentioned in assumptions.(Took help from datetime library)
- ➔ If the dates are valid, called the Data_frame(json_file_path,start_date,end_date) function.
- ➔ If the dateflag returned from Data_frame is false then we return the flow back to main function.
- ➔ If the dateflag is true then we have data for all the dates between start and end date.
- ➔ We use get_sorted(DataFrame, “dl”) function to extract out the values of state Delhi
- ➔ Using the final data frame returned from get_sorted( ),we use cumsum() function on the confirmed recovered and deceased attributes of data frame to modify the count of cases to cumulative count.
- ➔ Finally used the modified final dataframe for the area plot.

2_3

Question Specific assumptions

- ➔ Used “tt” variable available in the data set for calculations. “tt” total cases for a particular day, including all the states, union territories and unassigned cases.

Methodology

- ➔ Validate the date according to the points mentioned in assumptions.(Took help from datetime library)
- ➔ If the dates are valid, called the Data_frame(json_file_path,start_date,end_date) function.
- ➔ If the dateflag returned from Data_frame is false then we return the flow back to the main function.
- ➔ If the dateflag is true then we have data for all the dates between start and end date.
- ➔ We use get_sorted(DataFrame, “tt”) function to extract out the values of “tt” attribute.
- ➔ Using the final data frame returned from get_sorted( ), Found the active cases for each day (temporarily) by subtracting the deceased and recovered from the active case. (done for each date)
- ➔ Used cumulative sum function on the above daily active cases to find the total active cases for every day

Q3.

Assumptions/Method used

- ➔ Used Linear Regression Using Least Squares method
- ➔ Considered only one feature date for the regression lines.
- ➔ Mapped dates to a ordinal values 0,1,2,3, …. With 14-05-2020 being 0.
- ➔ Line equation for simple linear regression
  - ◆ $Y = Slope * X + Intecept$
    - Y is the dependent variable which refers to confirmed/recovered/deceased cases.
    - X is the input/independent variabl which refers to ate in ordinal formal.
  - ◆ Slope= $\mathbf{[\sum^{n}_{i=1}(X_i - X\_mean)(Y_i - Y\_mean)] / [\sum^{n}_{i=1}(X_i - X\_mean)]^2}$
  - ◆ Intercept = Y_Mean - Slope* X_Mean

References:https://towardsdatascience.com/linear-regression-using-least-squares-a4c3456e8570

Methodology

- ➔ Validate the date according to the points mentioned in assumptions.(Took help from datetime library)
- ➔ If the dates are valid, called the Data_frame(json_file_path,start_date,end_date) function.

- ➔ If the dateflag returned from Data_frame is false then we return the flow back to main function.
- ➔ If the dateflag is true then we have data for all the dates between start and end date.
- ➔ We use get_sorted(DataFrame, “dl”) function to extract out the values of state Delhi.
- ➔ Map the date to ordinal values like 0,1,2,3,4…..
- ➔ Call the linear_regression function separately for confirmed ,recovered and deceased cases.
    - ◆ Linear_Regression(X,Y,Title,Date):
        - X is independent variable and Y is dependent variable.
        - Calculate the slop and intercept using the formula above
        - .Return the computed slopes and intercepts.
- ➔ Return the slope and intercept for all three cases.

Outputs

1_1.

confirmed_count: 4110211 recovered_count: 3177666 deceased_count: 70094

1_2.

confirmed_count: 188193 recovered_count: 163785 deceased_count: 4538

1_3.

confirmed_count: 1072055 recovered_count: 800359 deceased_count: 30813

1_4.

Confirmed

Highest affected State is: Maharashtra
Highest affected State count is: 883862

Recovered

Highest affected State is: Maharashtra
Highest affected State count is: 636574

Deceased

Highest affected State is: Maharashtra
Highest affected State count is: 26275

1_5.

Confirmed

Lowest affected State is: Mizoram
Lowest affected State count is: 1062

Recovered

Lowest affected State is: Mizoram
Lowest affected State count is: 713

Deceased

Lowest affected State is: Mizoram
Lowest affected State count is: 0

1_6

Confirmed

Day: 2020-06-23
Count: 3947

Recovered

Day: 2020-06-20
Count: 7725

Deceased

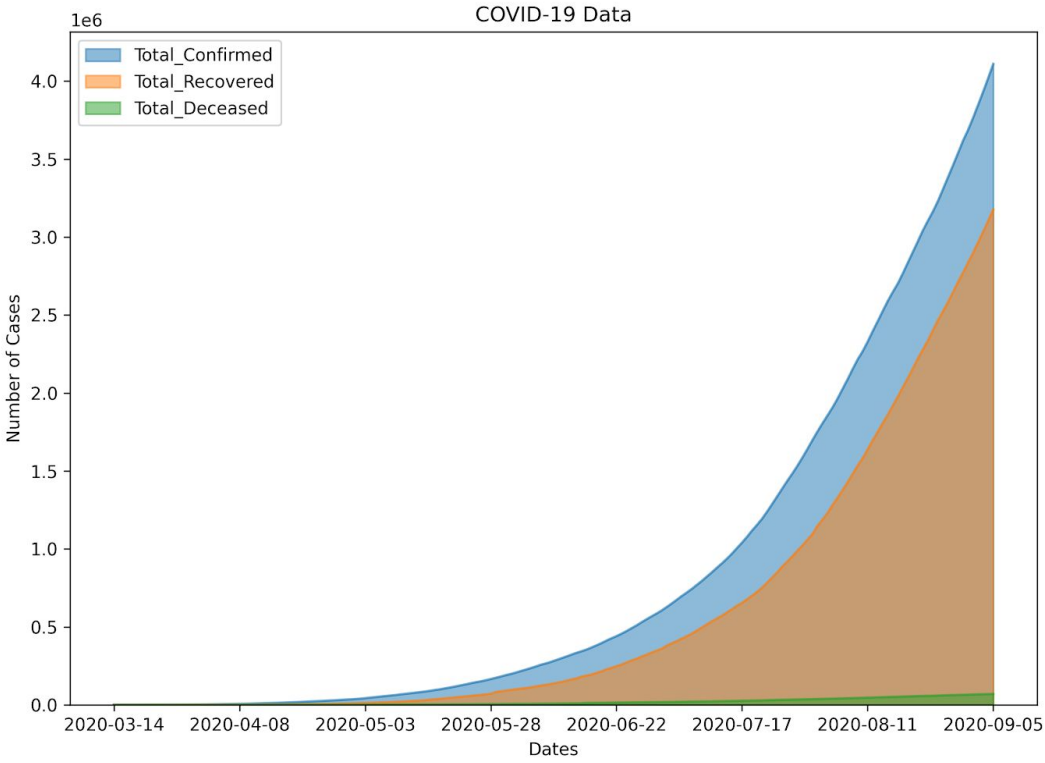
Day: 2020-06-16
Count: 437

1_7

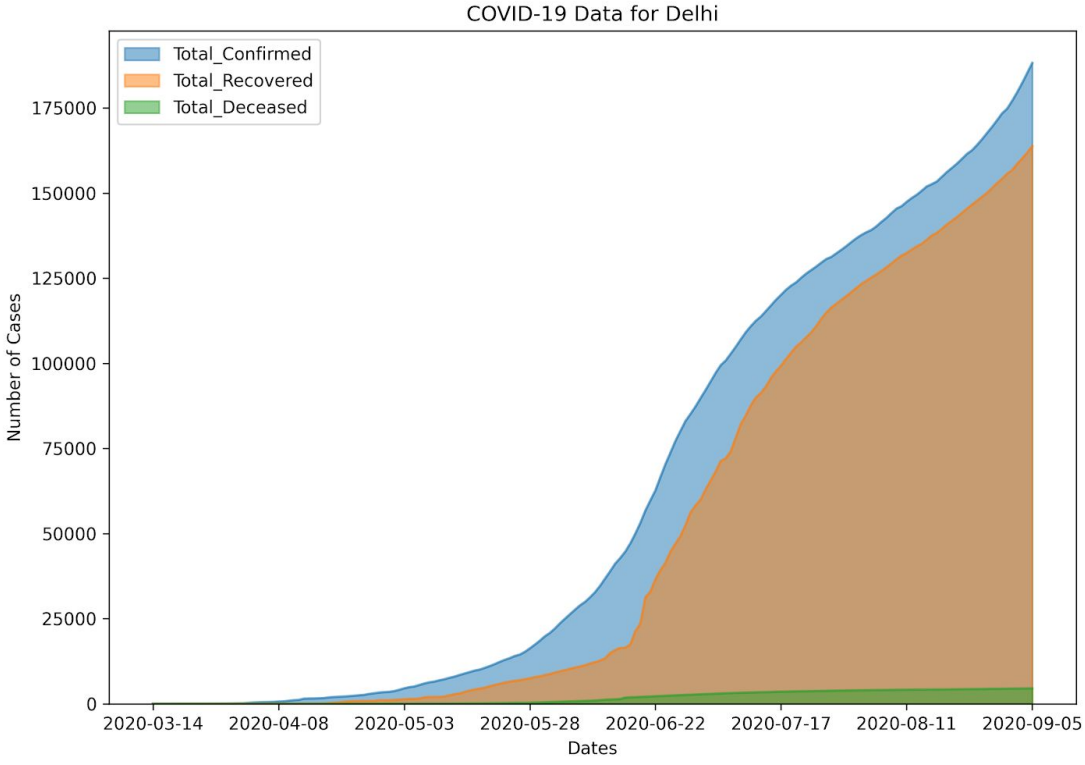
Andhra Pradesh : 100880
Arunachal Pradesh : 1525
Assam : 28404
Bihar : 16735

Chattisgarh : 22320
Delhi : 19870
Goa : 4945
Gujarat : 16266
Himachal Pradesh : 2023
Haryana : 14912
Jharkhand : 14980
Karnataka : 100224
Kerala : 21867
Maharashtra : 221013
Meghalaya : 1374
Manipur : 1872
Madhya Pradesh : 15687
Mizoram : 349
Nagaland : 701
Odisha : 25856
Punjab : 15870
Rajasthan : 14996
Sikkim : 561
Telangana : 32405
Tamil Nadu : 51580
Tripura : 5905
Uttar Pradesh : 59963
Uttarakhand : 7649
WestBengal : 23390

2_1

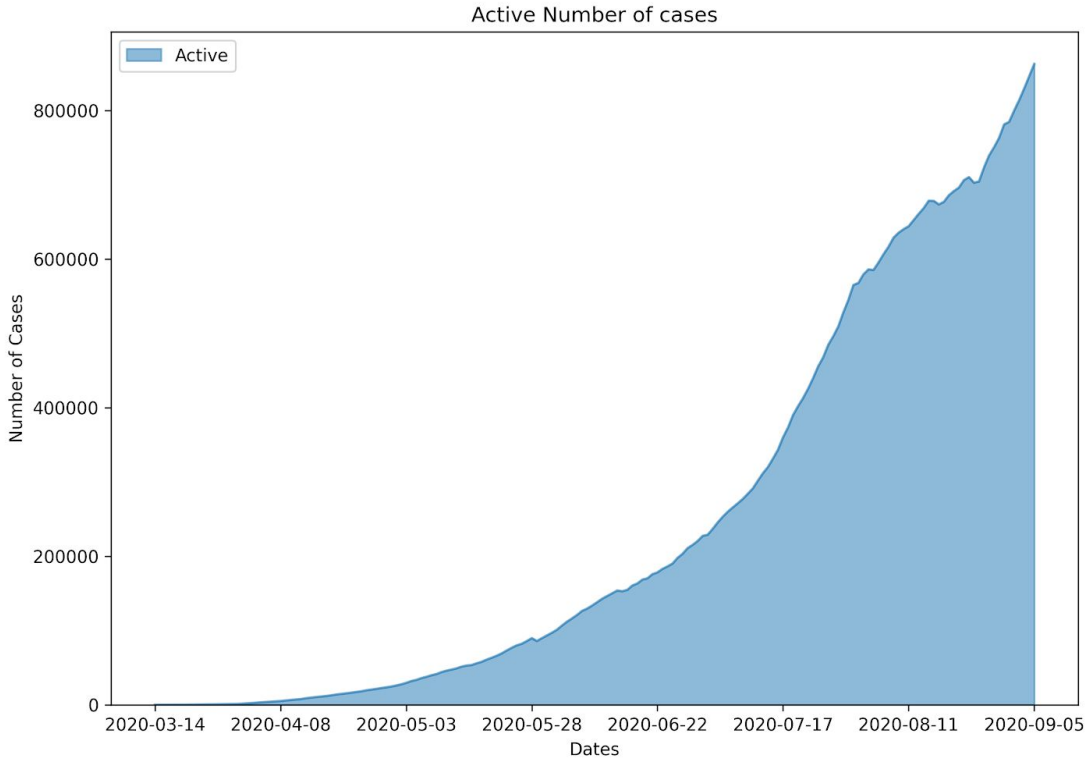


2_2



2018
Navr

2_3



Q3.

Printed on terminal

The slope is 12.2142692053709 and the intercept is 0.5298536209552367 for the Linear Regression line for Confirmed cases
The slope is 12.305528285274049 and the intercept is -146.13713405238843 for the Linear Regression line for Recovered cases
The slope is 0.19023332599603787 and the intercept is 9.138674884437595 for the Linear Regression line for Deceased cases

Function returns this

(0.5298536209552367, 12.2142692053709, -146.13713405238843, 12.305528285274049, 9.138674884437595, 0.19023332599603787)

2018348
Navneet Agarwal

2018378
Aditya SIngh