**Data-Preprocessing-using-R**

In this project we have to apply the pre-processing techniques in the given dataset to prepare the dataset for data analysis.

**Dataset**

The following dataset - [dataset.csv](https://github.com/MunimAhmed/Data-Preprocessing-using-R/blob/main/dataset.csv) contains statistics in arrests per 100,000 residents for assault and murder, in each of the 50 US states, in 1973. Also given is the percentage of the population living in urban areas.

**Overview**

In order to prepare a cleaned dataset, we need to perform the following tasks of data pre-processing using R language:

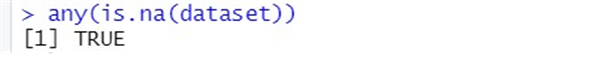
1. Data cleaning:
   * Smooth Noisy Data
   * Handling Missing Data
   * Data Wrangling or Munging
2. Data Integration
3. Data Discretization

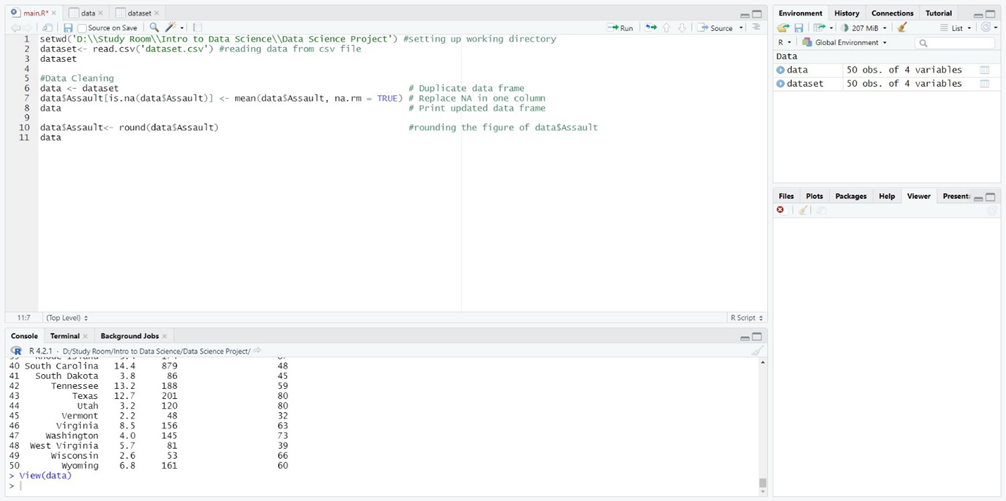
**Project Solution Design:**

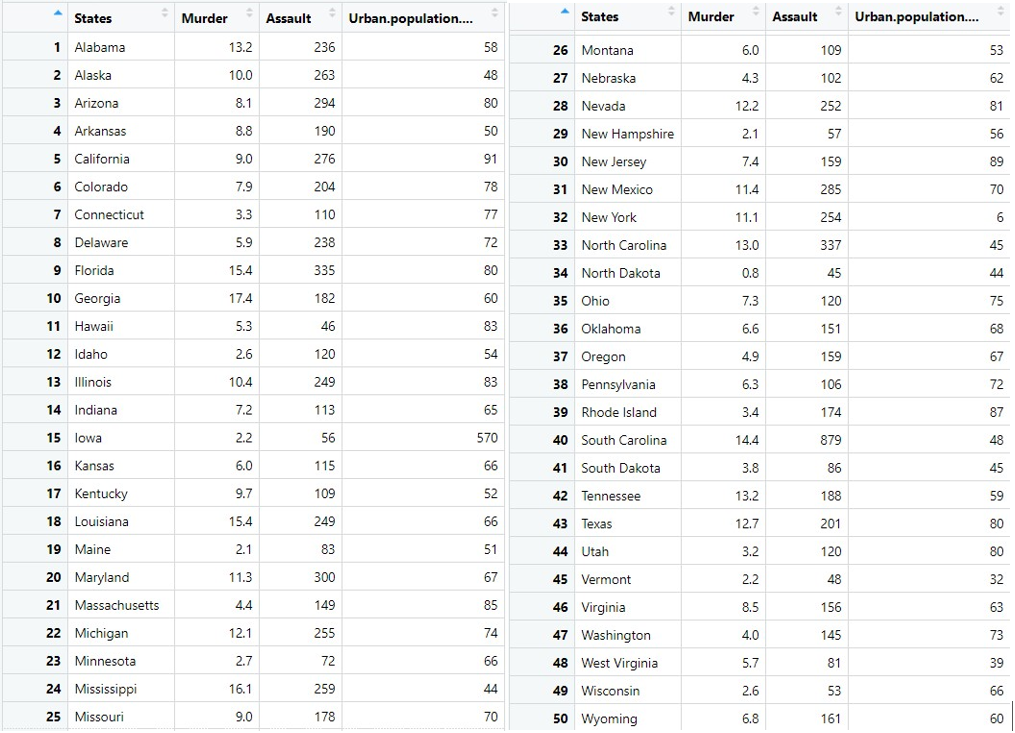
Firstly, the data file was scanned and transformed into a csv(comma separated value) format. Then, Rstudio tool was used and the csv file was read and imported as a dataframe in Rstudio. Next, the data cleaning process started, first missing values were checked and we handled the missing values. Then we started to handle noisy data, we checked the outliers for different variables and handle the outliers with proper methods. Then we started data integration part and add a new column in the dataframe which contains value according to the given condition comparing other variable of the particular row. Lastly, data discretization was performed to categorized and understand the data easily.

**Data Cleaning**

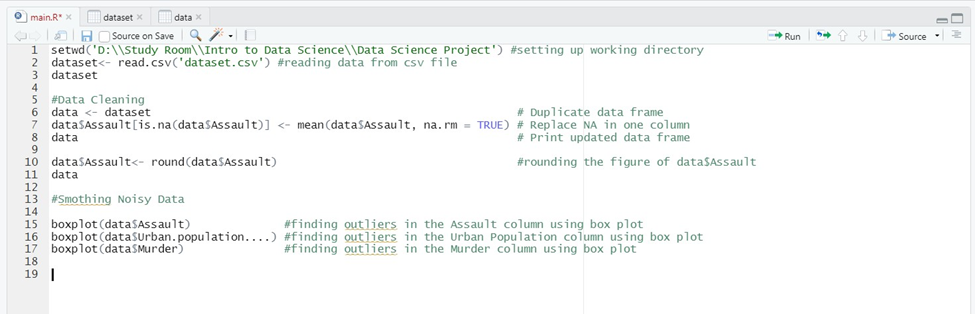
**Handling Missing Data:**

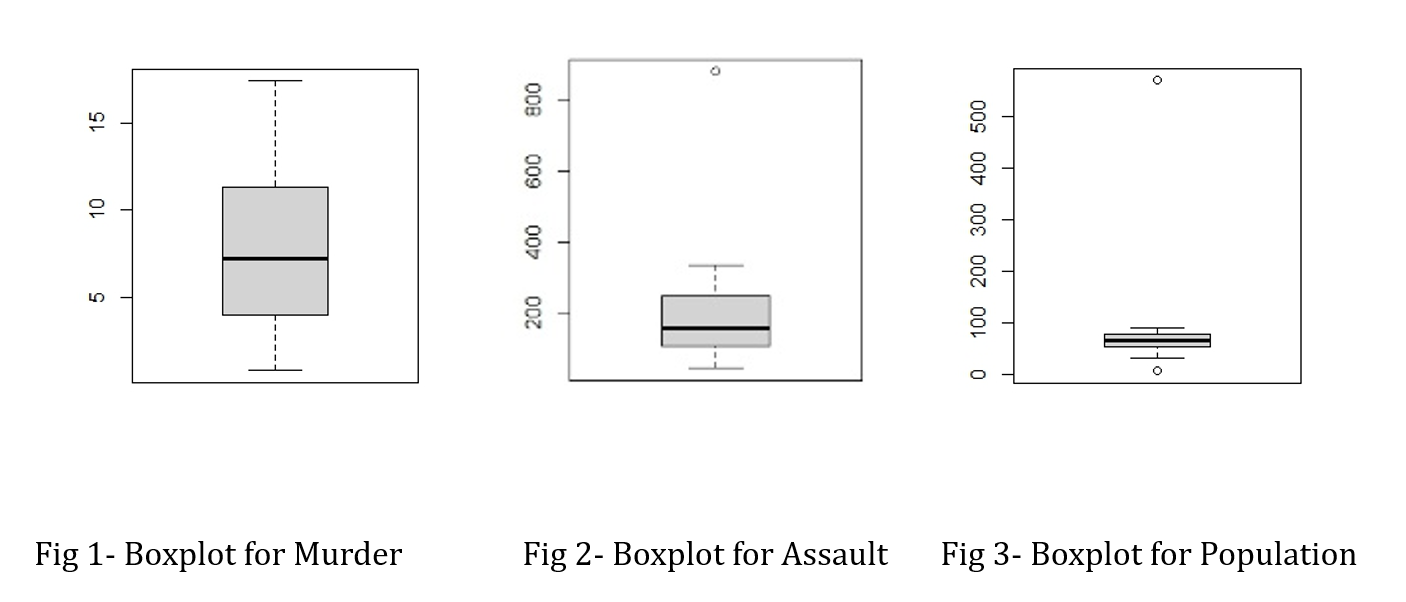
In order to handle missing data at first we check if there are any missing values in the dataframe. [](https://user-images.githubusercontent.com/69957858/210398299-181599cf-584f-47d9-a7b4-19bfa143ed32.png)

The result shows true, that means there are missing values in the data. And then we found the there were missing value in the “Assault” column. In order to handle the missing value we replace it with the Mean value of that column. [](https://user-images.githubusercontent.com/69957858/210398405-c797cf34-65d1-4979-8d1f-58f9918cf991.png) Though it is a part of data transformation, here, we have also round the values of the column “Assault” as the mean value was four decimal numbers, all the numbers converted into four decimal values. So, we used round function to fix it.

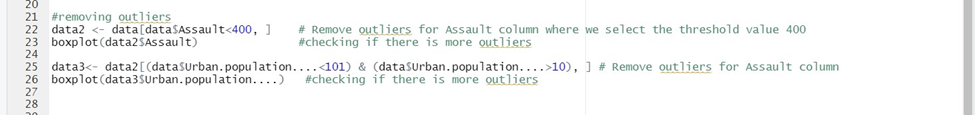
**After Handling Missing Values: (49 rows, 4 variables)** [](https://user-images.githubusercontent.com/69957858/210398780-ce397086-6578-4096-b26d-cf24fb0e6cec.png)

**Smoothing Noisy Data:**

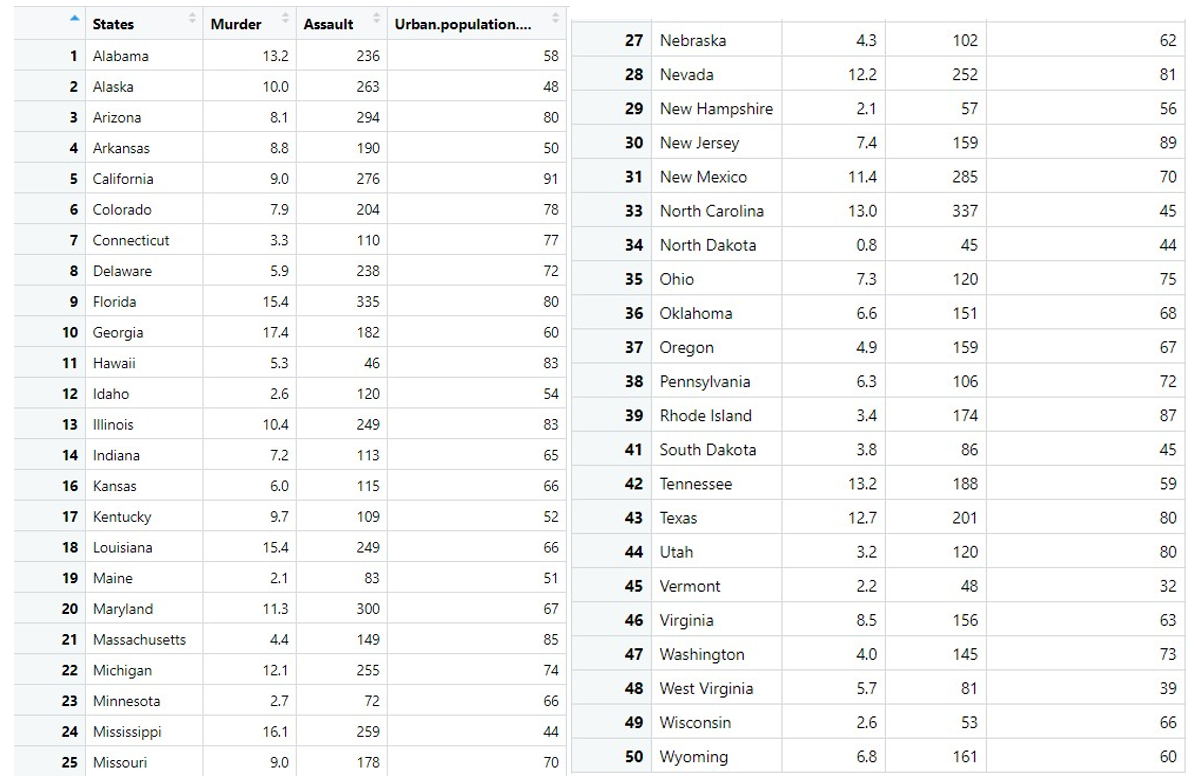
In order to smooth noisy data, we need to find outliers in different variables and then we will remove the noisy data. In order to find the outlier we used boxplot() functions, it helps to detect outliers easily. [](https://user-images.githubusercontent.com/69957858/210399179-babea080-7f45-4edb-afa6-7845bb30015c.png)

After plotting: [](https://user-images.githubusercontent.com/69957858/210399324-aa6b10c4-0a11-46b1-b11a-47b643c2c8bc.png)

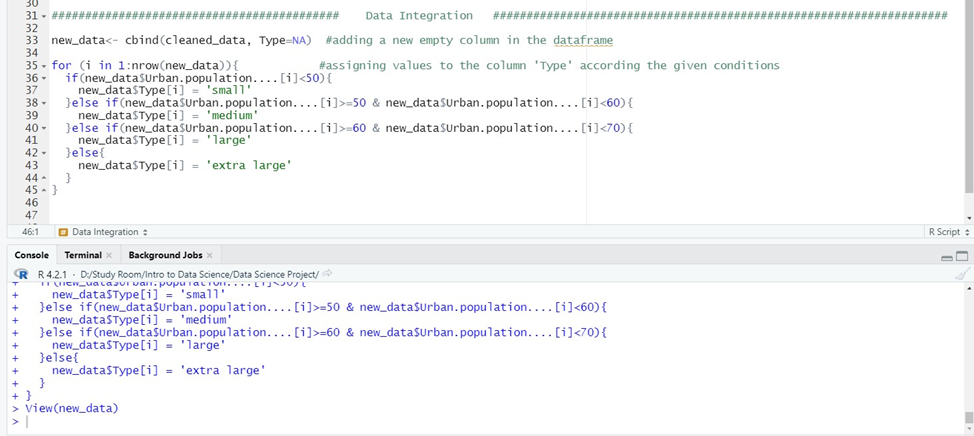
From here, we can see there is no outlier in Murder column but in the Assault column there is one outlier and in the Population column there is 2 outliers. So, we have to handle these noisy data by taking some thresholds for the value and using condition to check the values of each column whether they are between the thresholds value. For population, as it is in percentage, we kept the threshold value between 10 to 100, any value greater than 100 or less than 10 was removed. So, rows that contains population percentage more than 100 or less than 10 was dropped out. For assaults, we kept the threshold values less than 400.

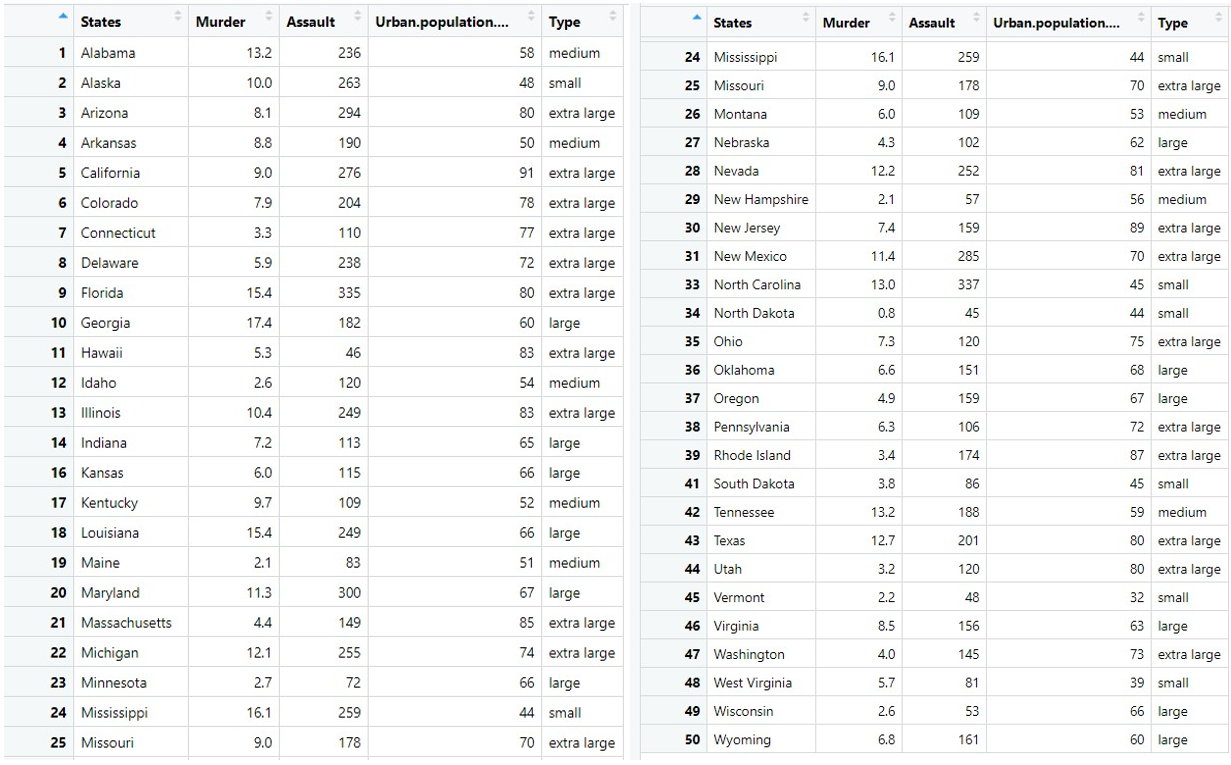
[](https://user-images.githubusercontent.com/69957858/210399411-23aca048-bbbc-4748-bc5b-147870f13170.png)

So, there were no more outliers in any column of the dataframe.

**After Smoothing Noisy Data: (47 rows, 4 variables)** [](https://user-images.githubusercontent.com/69957858/210399625-6bbb3a3e-1b65-441f-9021-d32ef424353e.png) From here we can see that, row 15, 32 and 40 were removed as there were noisy data.

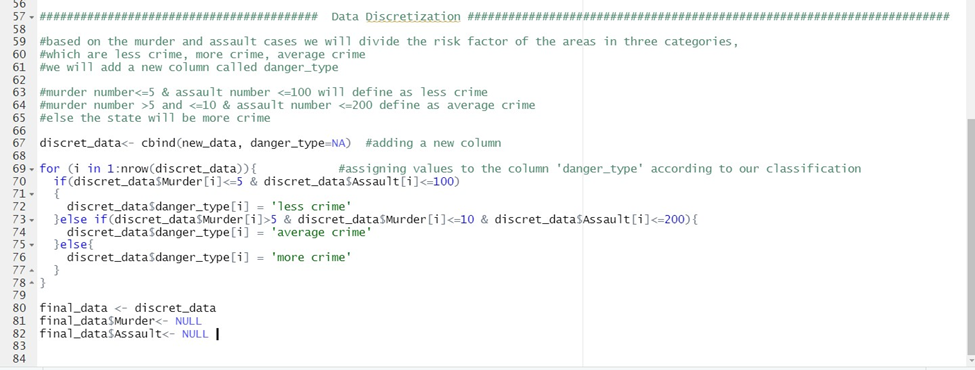
**Data Integration:**

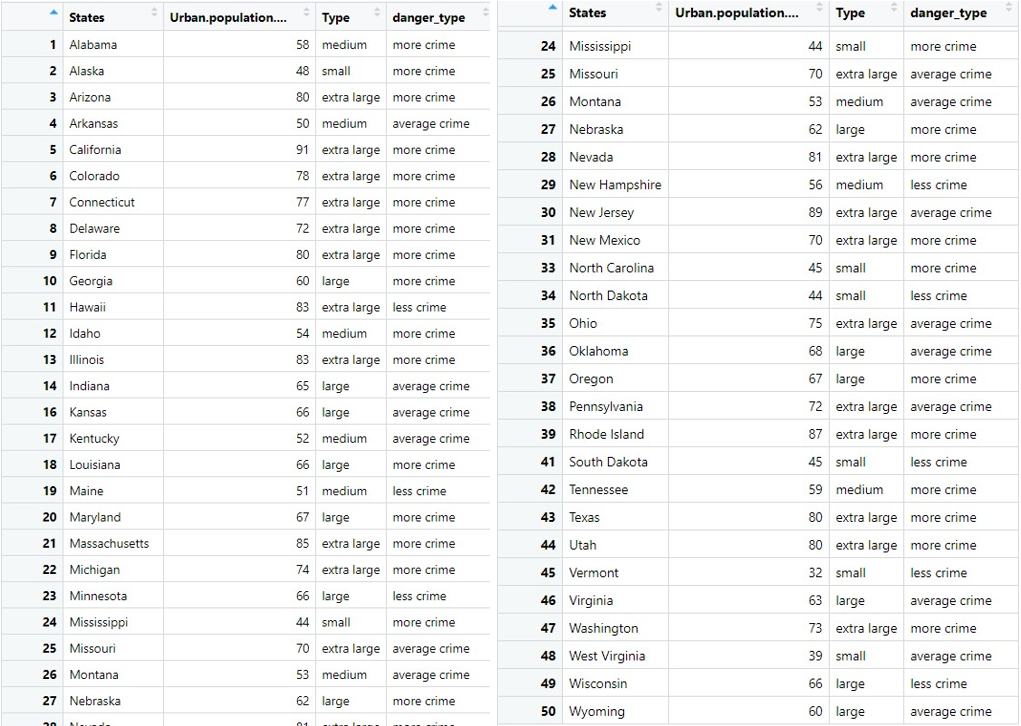
In integration part, we have added a new column name “Type” to our latest cleaned dataframe, at first, we did not set any value for the column. Next, we used the given condition and assigned the values for each row of the Type column. The given condition was, small (<50%), medium (<60%), large (<70%), and extra-large (70% and above) [](https://user-images.githubusercontent.com/69957858/210399866-f8898a50-bdd9-4145-b7c8-47c6f608b095.png)

**After Data Integration: (47 rows, 5 variables)** [](https://user-images.githubusercontent.com/69957858/210400019-55be6560-d6fd-4ea8-a02e-814d0de712c5.png)

**Data Discretization:**

As we can see, all the attributes involved in our dataset are continuous type values in real numbers). However, depending on the model we want to build, we have to discretize the attribute values into binary or categorical types. For this dataset, we want to take the column Murders and Assaults values and categorize them based on the numbers of murder and assault. we will divide the risk factor of the areas in three categories, which are less crime, more crime, average crime. For example: murder number<=5 & assault number <=100 will define as ‘less crime’, murder number >5 and <=10 & assault number <=200 define as ‘average crime’ and the others will be ‘more crime’ area.

[](https://user-images.githubusercontent.com/69957858/210401608-fb166f3a-a121-49d5-bde4-f33d31f5cd18.png)

**After Data Discretization:(47 rows, 4 variables)** [](https://user-images.githubusercontent.com/69957858/210401573-0447ab18-a5b8-449b-ba9e-5e1d20cc592f.png)

**Conclusion**

We now have a clean dataset because all the procedures have been completed. Outliers and noisy data are no longer present. The mean value for the category has been used to fill in the gaps left by the missing data. In order to make it easier to interpret the data, we additionally do discretization and add a new category that deals with the data's range.