|  |  |
| --- | --- |
|  | **AMERICAN INTERNATIONAL UNIVERSITY-BANGLADESH (AIUB)** |





|  |  |
| --- | --- |
| **Course:**  **INTRODUCTION TO DATA SCIENCE** |  |

**Mid Project**

***Submitted by***

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **ID** | **Section** | **Email** |
| SAJID IBNA MAHBUB | 20-42109-1 |  | **20-42109-1@student.aiub.edu** |



**Submitted To**

DR. AKINUL ISLAM JONY

Associate Professor

Faculty of Science and Technology

American International University-Bangladesh (AIUB)

**Project Overview:**

Real-world data is frequently incomplete, noisy, and inconsistent, meaning it needs to be cleaned up before it can be put to the intended use. Data pre-processing is a common term for this. Data preprocessing is a data mining technique used to turn raw data into a format that is both practical and effective. The most important tasks involved in data pre-processing are Data Cleaning, Data Integration, Data Transformation, Data Reduction, and Data Discretization. Data can have irrelevant and missing parts. To handle this part, data cleaning is done. It involves data munging, handling missing data, and smoothing noisy data. After that, to make data as effective and efficient for various data analyses as possible, data from various sources commonly needs Data Integration. Data Integration contains many steps like combining data from various sources into a coherent storage place, detecting and resolving data value conflicts, and addressing redundant data. The Data Transformation is performed to transform data and make it more consistent and readable. The steps of data transformation are smoothing, aggregation, generalization, normalization, and feature construction. After that, Data reduction is a crucial procedure in which a condensed version of a dataset that yields the same or comparable analytical results are obtained. Lastly, the process that is performed is Data Discretization. Data discretization is a method of converting attribute values of continuous data into a finite set of intervals with minimum data loss. These are the things that are combined to give us the final product which is Data Pre-Processing.

The dataset that is provided includes statistics for each assault and murder arrest per 100,000 residents in 1973, of the 50 US states. The percentage of people who live in urban regions is also provided. The dataset consists of the following attributes,

1. Name of the 50 US **States** represented.
2. Number of **Murder** arrests per 100,000.
3. Number of **Assault** arrests per 100,000.
4. Amount of Urban Population in percentage.

In this project, we are required to perform the techniques of Data pre-processing to obtain a clean dataset ready for Data Analysis. This dataset can be used to test a number of hypotheses or the connection between different traits.

**Project Solution Design:**

To reach the solution we first need to understand the data and decide the course of action. As we know, Data Pre-Processing involves five core steps which are Data Cleaning, Data Integration, Data Transformation, Data Reduction, and Data Discretization. To arrive at the conclusion that is a cleaned version of the supplied data set, we will proceed step by step.

Dirty Data

Data Cleaning

Data Integration

Data Transformation

Data Reduction

Data Discretization

Cleaned

Data Set

Figure: Data Pre-Processing steps

The software we’re going to use to manipulate data is RStudio. First, the data will be converted to a supported file pattern like CSV, XML or Excel. In our case, we converted it to excel and then imported the data to RStudio.

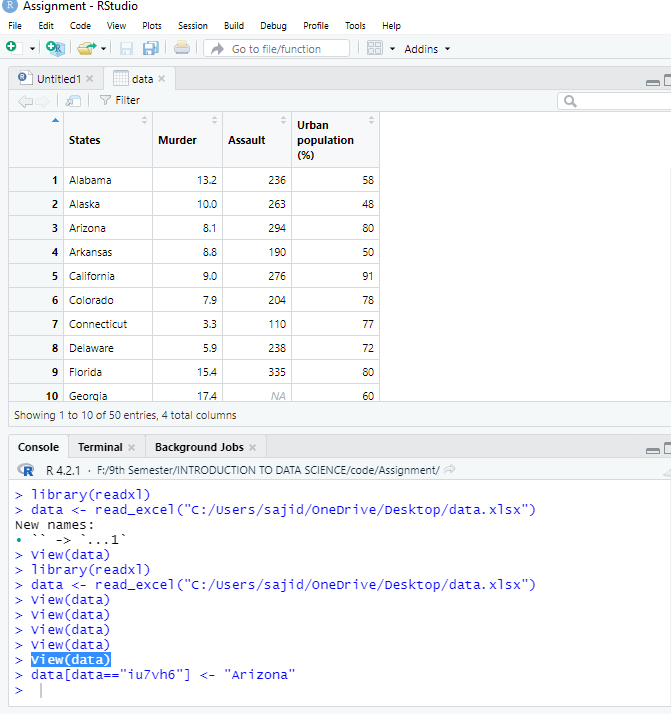
From pdf converted to excel then read the file using the following code:

>library(readxl)

>data <- read\_excel("C:/Users/sajid/OneDrive/Desktop/data.xlsx") [stored the file in data. A structure used : Data Frame]

Then view the data using the following code:

>View(data)



**Data Pre-processing:**

**1)Data Cleaning:**

**a. Data Munging:** The Data set does not involve any data munging steps because all the data are per 100,000 residents.

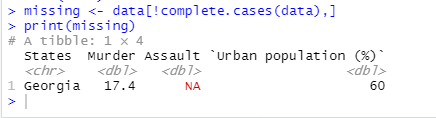
**b. Handling Missing Data:** To handle missing data we first need to search the data set for any value that is not assigned. To do so we write a code which will show us the row which contains the missing value,

Code:

missing <- data[!complete.cases(data),]

print(missing)

Output:

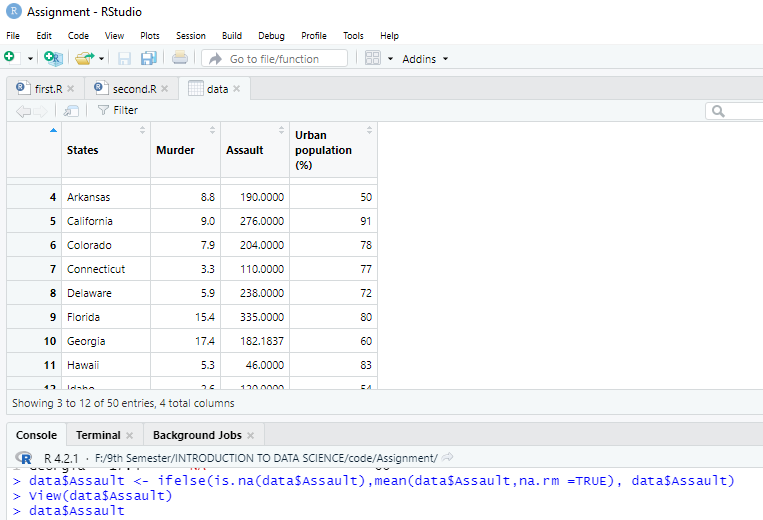


Now we’ve got the only missing data in the whole data set. The next stop is to perform the handling procedure. To do so, We calculate the mean of the column Assault except for the empty data and place it in the place on empty data.

Code:

data$Assault <- ifelse(is.na(data$Assault),mean(data$Assault,na.rm =TRUE), data$Assault)

Output:



Now that the missing data is being handled we move to Smooth Noisy data.

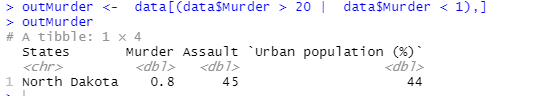
**c. Smooth Noisy Data:**

For this process, we first search for any outliers in the data set. We set specific ranges for each column. Now we run the R codes that are mentioned below,

Code & Output:

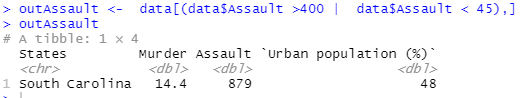
1) outMurder <- data[(data$Murder > 20 | data$Murder < 1),]

outMurder



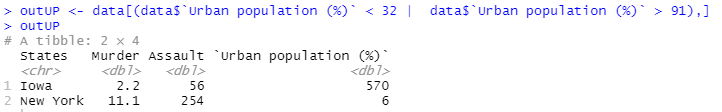
2) outAssault <- data[(data$Assault >400 | data$Assault < 45),]

outAssault



3) outUP <- data[(data$`Urban population (%)` < 32 | data$`Urban population (%)` > 91),]

outUP



From the outputs we can see that the Murder column has one outlier, The Assault column has one outlier and the Urban Population column has two outliers.

The outlier of the Murder column will be handled as a part of Data Transformation. But for this part, we’ll handle the other outliers.

Now first let’s handle the outlier of the Assault column:

To handle it we replaced the value with the second-highest value available in the column.

Code:

data$Assault[data$Assault == 879] <- sort(data$Assault, TRUE)[2]

Output:



Second to handle the outliers of the Urban Population Column:

Here we replace 570 and 6 with 60 as we think it is a error that occurred during datafication.

Code:

data$`Urban population (%)`[data$`Urban population (%)` == 570] <- 57

data$`Urban population (%)`[data$`Urban population (%)` == 6] <- 60

Output:





Then we just to be sure, check whether there are any duplicate data,

Code & Output:



As there is no duplicate data, we move on with out next step.

**2) Data Integration:**

In this phase, we need to integrate new columns based on the condition provided which are applied to the Urban Population Variable.

**Question:** prepare the dataset to integrate a new column (named type) based on the

urban population variable. [Hint: Convert the urban population percentage into types, for example, small (<50%), medium (<60%), large (<70%), and extra-large (70% and above).]

Code:

library(dplyr)

new <- data %>% mutate(Type = case\_when(

(data$`Urban population (%)`) < 50 ~ "Small",

(data$`Urban population (%)`) < 60 ~ "Medium",

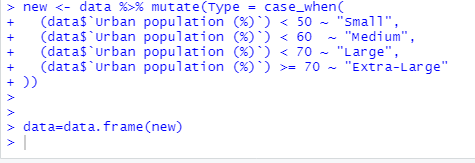
(data$`Urban population (%)`) < 70 ~ "Large",

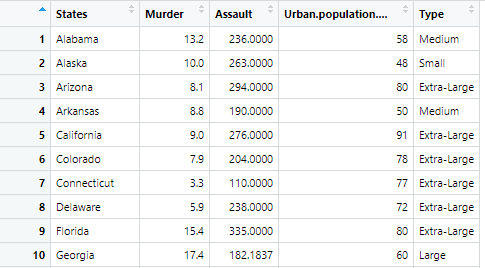
(data$`Urban population (%)`) >= 70 ~ "Extra-Large"

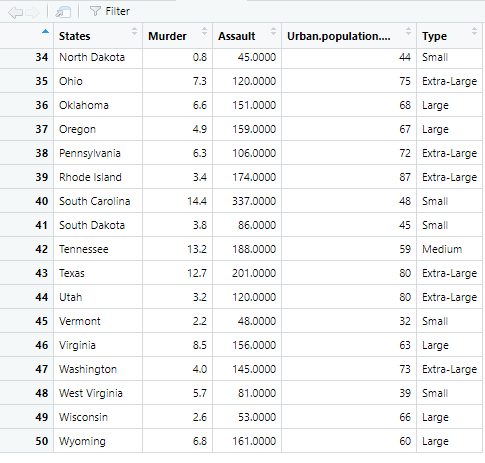
))

data=data.frame(new)

Output:







The integration of the new column is done. We continue the steps,

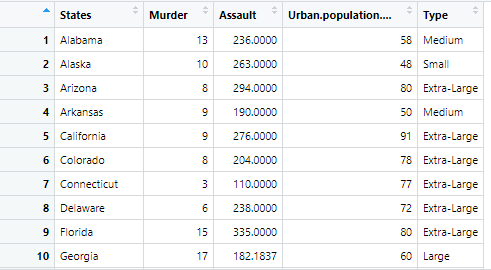
**3) Data Transformation:**

In this step, we can see the Murder Variable contains values that are in decimal. Body count cannot be decimal or fraction. It should be a whole number. Now we transform the values of the variable in whole number,

Code:

data$Murder <- round (data$Murder)

Output:



The Murder column is now Transformed.

We can also transform the Murder and Assault column into an integer.

Code:

data$Murder <- as.integer(data$Murder)

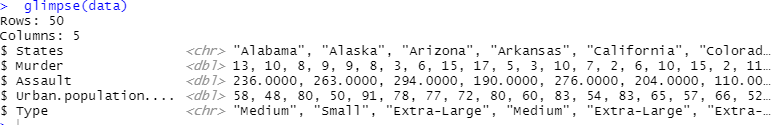
data$Urban.population.... <-as.integer(data$Urban.population....)

Next, we check the types of all the Variables,

Code:

glimpse(data)

Output:



It is seen that the Type variable is of CHARACTER type. So we need to transform it into Factor Structure(Ordinal). Because Factors make data easy to understand and analyze data.

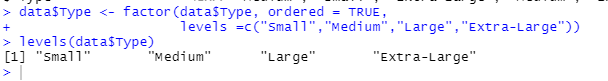
Code:

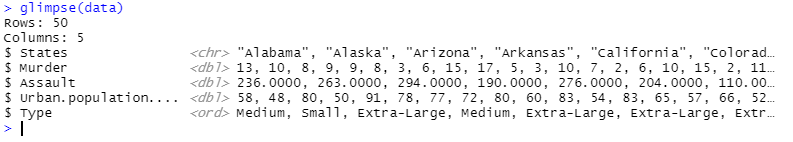
data$Type <- factor(data$Type, ordered = TRUE,

levels =c("Small","Medium","Large","Extra-Large"))

levels(data$Type)

Output:





1. **Data Reduction:**

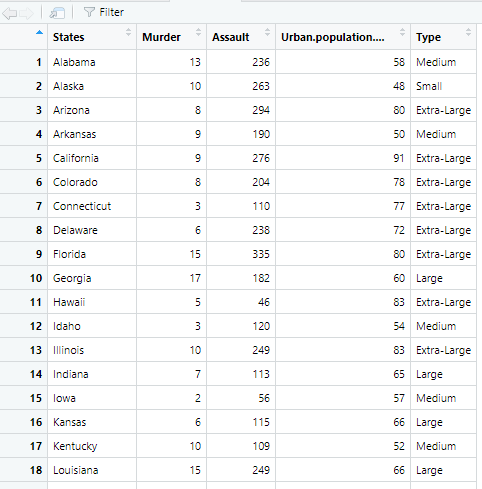
A large dataset will require a substantial quantity of storage space for each additional decimal place for every data point. So reducing the Assault column to one decimal place would result in a more reduced dataset.

Code:

data$Assault = as.numeric(format(round(data$Assault,0)))

Result:

So at the end of the Data Reduction Stage, the Data set looks like the below.



1. **Data Discretization:** Discretization of the data set is not needed in this project.

After the complete pre-processing of the data, this is what it looks like,

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **States** | **Murder** | **Assault** | **Urban.population....** | Type |
| Alabama | 13 | 236 | 58 | Medium |
| Alaska | 10 | 263 | 48 | Small |
| Arizona | 8 | 294 | 80 | Extra-Large |
| Arkansas | 9 | 190 | 50 | Medium |
| California | 9 | 276 | 91 | Extra-Large |
| Colorado | 8 | 204 | 78 | Extra-Large |
| Connecticut | 3 | 110 | 77 | Extra-Large |
| Delaware | 6 | 238 | 72 | Extra-Large |
| Florida | 15 | 335 | 80 | Extra-Large |
| Georgia | 17 | 182 | 60 | Large |
| Hawaii | 5 | 46 | 83 | Extra-Large |
| Idaho | 3 | 120 | 54 | Medium |
| Illinois | 10 | 249 | 83 | Extra-Large |
| Indiana | 7 | 113 | 65 | Large |
| Iowa | 2 | 56 | 57 | Medium |
| Kansas | 6 | 115 | 66 | Large |
| Kentucky | 10 | 109 | 52 | Medium |
| Louisiana | 15 | 249 | 66 | Large |
| Maine | 2 | 83 | 51 | Medium |
| Maryland | 11 | 300 | 67 | Large |
| Massachusetts | 4 | 149 | 85 | Extra-Large |
| Michigan | 12 | 255 | 74 | Extra-Large |
| Minnesota | 3 | 72 | 66 | Large |
| Mississippi | 16 | 259 | 44 | Small |
| Missouri | 9 | 178 | 70 | Extra-Large |
| Montana | 6 | 109 | 53 | Medium |
| Nebraska | 4 | 102 | 62 | Large |
| Nevada | 12 | 252 | 81 | Extra-Large |
| New Hampshire | 2 | 57 | 56 | Medium |
| New Jersey | 7 | 168 | 89 | Extra-Large |
| New Mexico | 11 | 285 | 70 | Extra-Large |
| New York | 11 | 254 | 60 | Large |
| North Carolina | 13 | 337 | 45 | Small |
| North Dakota | 1 | 45 | 44 | Small |
| Ohio | 7 | 120 | 75 | Extra-Large |
| Oklahoma | 7 | 151 | 68 | Large |
| Oregon | 5 | 168 | 67 | Large |
| Pennsylvania | 6 | 106 | 72 | Extra-Large |
| Rhode Island | 3 | 174 | 87 | Extra-Large |
| South Carolina | 14 | 168 | 48 | Small |
| South Dakota | 4 | 86 | 45 | Small |
| Tennessee | 13 | 188 | 59 | Medium |
| Texas | 13 | 201 | 80 | Extra-Large |
| Utah | 3 | 120 | 80 | Extra-Large |
| Vermont | 2 | 48 | 32 | Small |
| Virginia | 8 | 156 | 63 | Large |
| Washington | 4 | 145 | 73 | Extra-Large |
| West Virginia | 6 | 81 | 39 | Small |
| Wisconsin | 3 | 53 | 66 | Large |
| Wyoming | 7 | 161 | 60 | Large |

**Discussion & Conclusion:**

In this Project, we applied Data Pre-Processing techniques on a Data Set by using R Language. We used numerous R language structures and tricks to refine the data step by step. After successfully executing all the data pre-processing techniques, we got a cleaner and better version of the Data set. Though some phases of the techniques were not needed to be executed in this project. We got insights into real-world data and how data is being pre-processed in the industry, Adding more experience to our arsenal. Data pre-preprocessing is one of the most important tasks in making data prepared for analysis. Because with dirty data, it is almost impossible to generate an accurate result. At the end of the preprocessing, we managed to prepare a much better and more meaningful dataset ready for analysis. Still, some more things can be done. Like if we replace the type column with corresponding numbers, we will be able to get a better type of data to analyze. This analysis enriched my knowledge about data preprocessing. The project also made us learn some R language libraries that will help us in the future. Overall doing this project was a good experience.