Department of Computer Science
American International University-Bangladesh
Mid Term Report

Course Name: INTRODUCTION TO DATA SCIENCE

"A report on Data Pre-Processing"

Supervised By:

Dr. Akinul Islam Jony

Associate Professor, Computer Science-AIUB

Submitted By:

Saima Sadia Ratri

ID: 20-43793-2

Section: B

Submission Date: October 24, 2022.

Project Title: Applying Data Pre-processing on a Dataset.

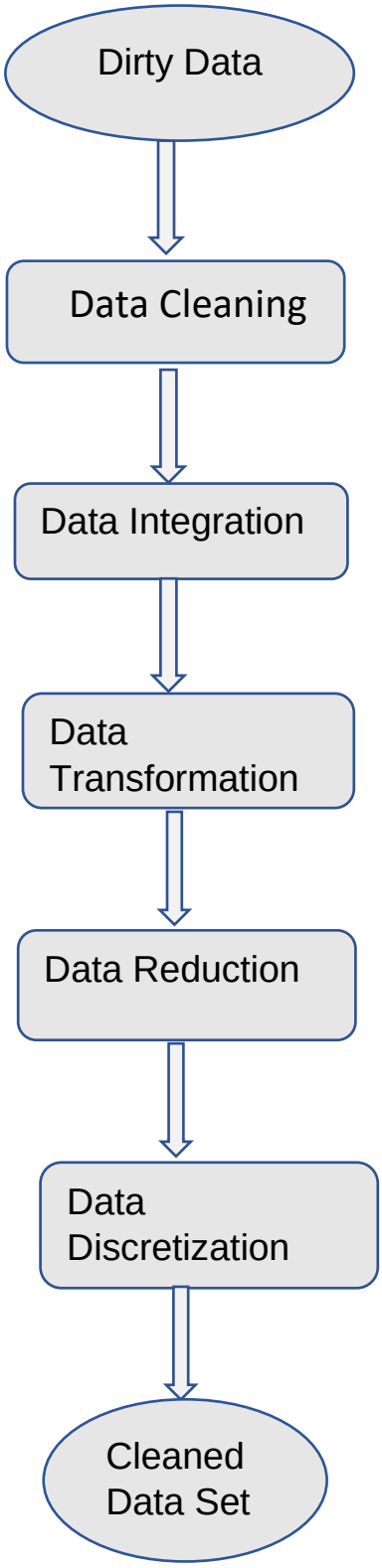
Project Overview:

Raw, real-world data in the form of text, images, video, etc., is messy. Not only may it contain errors and inconsistencies, but it is often incomplete, and doesn't have a regular, uniform design. Machines like to process nice and tidy information – they read data as 1s and 0s. So, calculating structured data, like whole numbers and percentages is easy. However, unstructured data, in the form of text and images must first be cleaned and formatted before analysis. Data preprocessing refers to the steps involved in transforming or encoding data so that it may be easily interpreted by a computer. The algorithm must be able to quickly interpret the data's attributes in order for a model to be accurate and exact in predictions. Due to their various origins, most real-world datasets are particularly prone to missing, inconsistent, and noisy data. Applying data mining algorithms to this noisy data would produce poor results since they would be unable to detect patterns. As a result, data preprocessing is critical for improving overall data quality. There are 4 major stages of data preprocessing – Data Cleaning, Data Integration, Data Transformation, Data Reduction and Data Discretization. Data Cleaning is a step in the data preprocessing process that involves filling in missing values, smoothing noisy data, resolving inconsistencies, and removing outliers. Data Integration is a data preparation phase that combines data from numerous sources into a single larger data storage, such as a data warehouse. Data Transformation is a technique for converting high-quality data into different formats by altering the value, structure, or format of data using techniques such as scaling, normalization, and others. Data transformation includes data cleaning techniques and a data reduction technique to convert the data into the appropriate form. Data transformation includes data cleaning techniques and a data reduction technique to convert the data into the appropriate form. Data transformation is an essential data preprocessing technique that must be performed on the data before data mining to provide patterns that are easier to understand. This a systemic process for data pre-processing.

The following dataset of the reports 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.

FLOWCHART:

In this project, we are required to perform the techniques of Data pre-processing to obtain a clean dataset ready for Data Analysis. Here's a flowchart of the project.



Software Used for This Project:

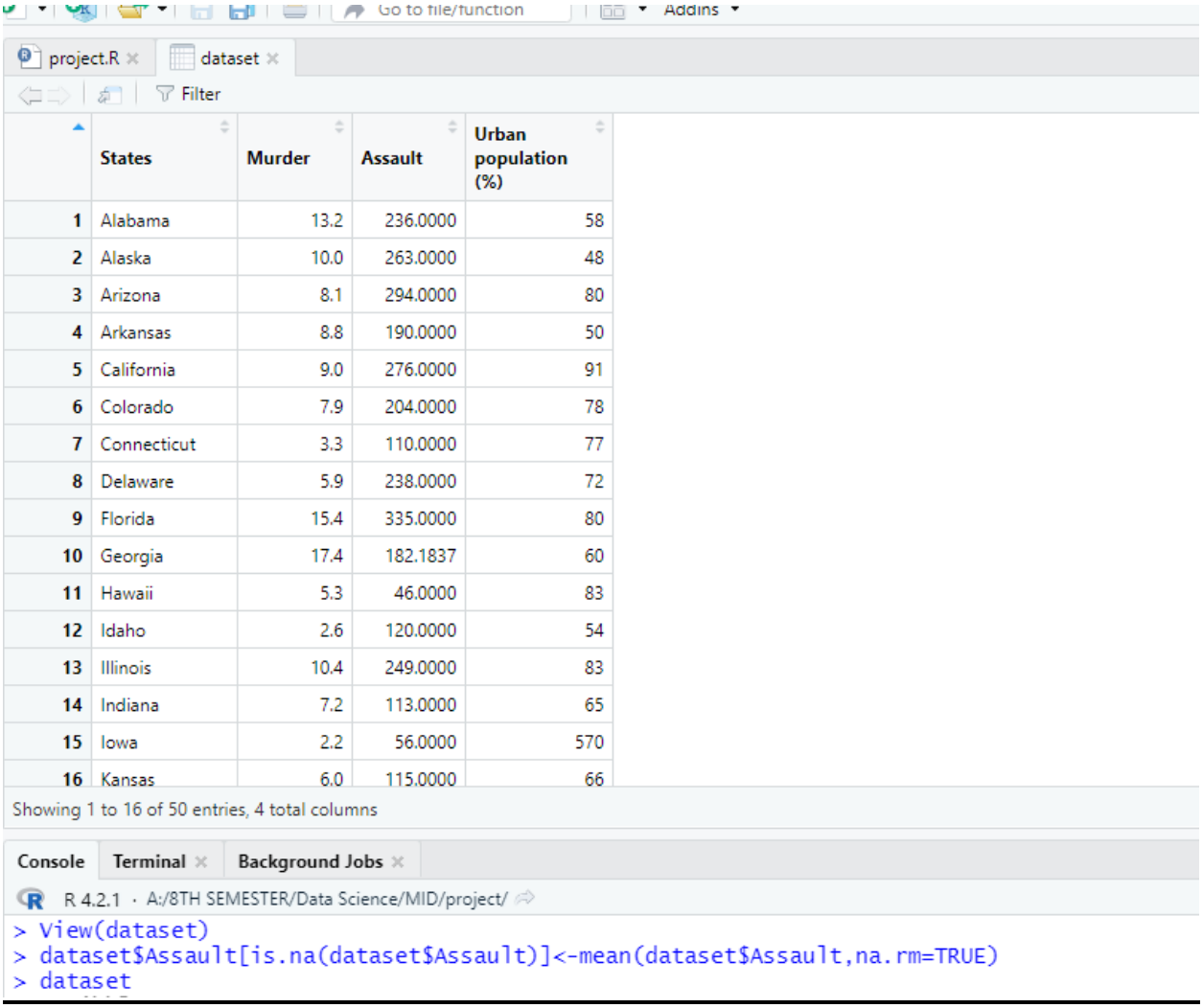
The software we're going to use to shape data is RStudio. RStudio is a powerful and easy way to interact with R programming, considered as Integrated Development Environment (IDE) that provides a one-stop solution for all the statistical computing and graphics. The RStudio is a more advanced version of R that comes with a multi-pane window setup that provides access to all primary things on a single screen (such as source, console, environment & history, files, photos, graphs, etc). Firstly, we converted data set to excel file with the help of a pdf to excel converter tool.

Data Pre-processing:

1. Data Cleaning:
    I. **Data Munging:** Since in this dataset all the data are per 100,000 residents, there are no data munging steps in the data set.
    II. **Handling Missing Data:** To deal with missing data, we must first examine the data collection for any unassigned values. As here Georgia data is missing as we can see. We calculate the mean of the column Assault, except for the empty data, and we add it where the empty data should be.

    Code:

    ```
    dataset$Assault[is.na(dataset$Assault)]<-mean(dataset$Assault,na.rm=TRUE)
    dataset
    ```



project.R | dataset

Filter

	States	Murder	Assault	Urban population (%)
1	Alabama	13.2	236.0000	58
2	Alaska	10.0	263.0000	48
3	Arizona	8.1	294.0000	80
4	Arkansas	8.8	190.0000	50
5	California	9.0	276.0000	91
6	Colorado	7.9	204.0000	78
7	Connecticut	3.3	110.0000	77
8	Delaware	5.9	238.0000	72
9	Florida	15.4	335.0000	80
10	Georgia	17.4	182.1837	60
11	Hawaii	5.3	46.0000	83
12	Idaho	2.6	120.0000	54
13	Illinois	10.4	249.0000	83
14	Indiana	7.2	113.0000	65
15	Iowa	2.2	56.0000	570
16	Kansas	6.0	115.0000	66

Showing 1 to 16 of 50 entries, 4 total columns

Console | Terminal | Background Jobs

R 4.2.1 · A:/8TH SEMESTER/Data Science/MID/project/

```
> View(dataset)
> dataset$Assault[is.na(dataset$Assault)]<-mean(dataset$Assault,na.rm=TRUE)
> dataset
```

Here missing data is placed.

III. Smooth Noisy Data:

Here we first search for any outliers in the data set. Now we run the codes .

Code: outMurder <- dataset[(dataset$Murder > 20 | dataset$Murder < 1),]

outMurder

output:

```
10 Georgia       17.4    182.                    60
# … with 40 more rows
# ℹ Use `print(n = ...)` to see more rows
> outMurder <-  dataset[(dataset$Murder > 20 |  dataset$Murder < 1),]
> outMurder
# A tibble: 1 × 4
  States       Murder Assault `Urban population (%)`
  <chr>         <dbl>   <dbl>                  <dbl>
1 North Dakota    0.8      45                     44
> |
```

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

outAssault

```
# A tibble: 1 × 4
  States         Murder Assault `Urban population (%)`
  <chr>           <dbl>   <dbl>                  <dbl>
1 South Carolina   14.4     879                     48
> |
```

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

outUP

```
> outUP <- dataset[(dataset$ Urban population (%)  < 32 |  dataset$ Urban population (%)  > 91),]
> outUP
# A tibble: 2 × 4
  States   Murder Assault `Urban population (%)`
  <chr>     <dbl>   <dbl>                  <dbl>
1 Iowa        2.2      56                    570
2 New York   11.1     254                      6
> |
```

We can observe from the results that there is one outlier in the Murder column, one in the Assault column, and two in the Urban Population column. The data transformation process will deal with the Murder column's outlier. However, we'll take care of the other outliers for this part. Here we deal with the outlier in the assault column. We dealt with it by changing the value to the second-highest number that was present in the column.

Code & Output:

dataset$Assault[dataset$Assault == 879] <- median(dataset$Assault)

```
  <chr>      <dbl>   <dbl>                  <dbl>
1 Iowa         2.2      56                    570
2 New York    11.1     254                      6
> dataset$Assault[dataset$Assault == 879] <- median(dataset$Assault)
> |
```

40	South Carolina	14.4	159.0000	48

Urban Population:

Code:

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

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

Output:

15	Iowa	2.2	56.0000	57
10	Georgia	17.4	182.1837	60

Data Integration:

During this step, we must implement new columns based on the specified conditions that are used with the Urban Population Variable. In our project we had told to do 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 <- dataset %>% mutate(Type = case_when(

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

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

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

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

))

dataset=dataset.frame(new)

Urban.population....	Type
58	Medium
48	Small
80	Extra-Large
50	Medium
91	Extra-Large
78	Extra-Large
77	Extra-Large
72	Extra-Large
80	Extra-Large
60	Large

Data Transformation:

As we can see murder values are in decimal, it can't be like this. To transfer decimal to a round number we can write

Code: dataset$Murder <- round (dataset$Murder)

	States	Murder
1	Alabama	13
2	Alaska	10
3	Arizona	8
4	Arkansas	9
5	California	9
6	Colorado	8
7	Connecticut	3
8	Delaware	6
9	Florida	15
10	Georgia	17
11	Hawaii	5
12	Idaho	3
13	Illinois	10
14	Indiana	7
15	Iowa	2

Code:

```
dataset$Type <- factor(dataset$Type, ordered = TRUE,
    levels =c("Small","Medium","Large","Extra-Large"))
levels(dataset$Type)
```

Output:

```
> levels(data$Type)
[1] "Small"       "Medium"      "Large"       "Extra-Large"
>
```

Data Reduction: Every additional decimal place for every data point in a big dataset will take a lot of storage space. Therefore, a more condensed dataset would arise by limiting the Assault column to one decimal place.

Code: dataset$Assault = as.numeric(format(round(dataset$Assault,0)))

	States	Murder	Assault
1	Alabama	13	236
2	Alaska	10	263
3	Arizona	8	294
4	Arkansas	9	190
5	California	9	276
6	Colorado	8	204
7	Connecticut	3	110
8	Delaware	6	238
9	Florida	15	335
10	Georgia	17	182
11	Hawaii	5	46
12	Idaho	3	120
13	Illinois	10	249
14	Indiana	7	113
15	Iowa	2	56

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

Though we can discretize type columns with numbers instead of Ordinal factors

Code:

```
dataset$Type <- factor(dataset$Type,
                       levels =c("Small","Medium","Large","Extra-Large"),labels=c(1,2,3,4))
```

Output:

	States	Murder	Assault	Urban.population....	Type
1	Alabama	13	236	58	2
2	Alaska	10	263	48	1
3	Arizona	8	294	80	4
4	Arkansas	9	190	50	2
5	California	9	276	91	4
6	Colorado	8	204	78	4
7	Connecticut	3	110	77	4
8	Delaware	6	238	72	4
9	Florida	15	335	80	4
10	Georgia	17	182	60	3
11	Hawaii	5	46	83	4
12	Idaho	3	120	54	2
13	Illinois	10	249	83	4
14	Indiana	7	113	65	3
15	Iowa	2	56	57	2
16	Kansas	6	115	66	3
17	Kentucky	10	109	52	2
18	Louisiana	15	249	66	3

Discussion & Conclusion:

To data processing we gradually improve the data, we used R language structures and techniques. The data set was made cleaner and better after all the data pre-processing techniques were successfully applied. Even so, not all steps of the techniques had to be used in this project. We learned about actual data and the pre-processing of data in the sector. enhancing our toolbox with greater experience. By pre-processing data, we Improve the accuracy of our dataset. We remove any values that are wrong or missing as a consequence of human error or problems. Consistency had been improved.