**American International University – Bangladesh**

**Department of Computer Science & Engineering**

****

**Project Title: Apply Data Pre-processing on a Dataset**

**Course: Introduction to Data Science**

|  |  |
| --- | --- |
| **Submitted by-** | **Submitted to-** |
| Name: Pritom Debnath  ID: 20-42414-1  Section: B  B.Sc. CSE | Name: Dr. Akinul Islam Jony |

**Project Overview:**

Data pre-processing is a phase in the data analysis process that takes raw data and converts it into a clean format that computers and machine learning can understand and analyse. Raw data in the real world is a jumbled mess. It may include contradictions and inaccuracies. It must be cleaned before it may be used for the intended purpose. The information in this project offers statistics in arrests per 100,000 residents for assault and murder in each of the 50 United States in 1973. The percentage of the people residing in cities is also provided. The dataset, as we can see, is not in clean format. Before it can be used, the dataset must be pre-processed and cleaned.

**Project Solution Design:**

The dataset shows that there is a missing value (null) in the Assault column. As a result, we must deal with the missing value. Because the Assault column's data type is numeric, substituting in the missing value with mean (average) might be an acceptable choice.

In addition, the Urban population (%) column has corrupt data. Because the Urban population (%) column shows the fraction of the population that lives in cities, it cannot be greater than 100 or less than 0. Yet, there is data in Iowa state where the Urban population (%) score is 570, indicating that there may be a larger problem. This issue might be caused by malfunctioning data gathering devices, data input issues, or technological restrictions. To deal with this faulty data, we must smooth it by removing the final digit (s).

We must separate the percentage of the population living in urban areas into Population\_level column in four groups during data pre-processing. Those are less than 50% (small), less than 60% (medium), less than 70% (large) and 70% and above (extra-large)

As Polulation\_level is not an ordered factor variable, that’s why it should be a better choice to add an ordered factor variable in the dataset. So, ordered\_factor\_population column is added.

|  |  |
| --- | --- |
| Polulation\_level | Ordered\_factor\_population |
| Small | 1 |
| Medium | 2 |
| Large | 3 |
| Extra-large | 4 |

So, at the end of data discretization stage, two new column named type will be integrated into the dataset based on above conditions.

**Data pre-processing:**

1. **Importing the Dataset:**

The data is saved in the working directory in the dataset.csv file. To begin pre-processing data in R, we must first import the dataset. Importing the dataset in R code –

|  |
| --- |
|  |

After importing the dataset, the dataset.csv converts into R dataframe and it is stored in dataset variable. After printing the dataset variable,

it looks like this-

|  |
| --- |
| Fig-1: Unprocessed dataset |

1. **Dealing with Missing Values:**

According to the dataset, there is a missing value (NA) in the Assault column. We can replace the missing value with the Assault column's mean value. R code for replacing missing value by the mean-

|  |
| --- |
|  |

Here in the code:

|  |  |
| --- | --- |
| is.na(dataset$Assault) | Returns true for all the cells in the specified column with no values. |
| mean(dataset$Assault, na.rm = TRUE) | Returns the average of the column passed as argument. |
| na.rm= TRUE | Calculates the mean excluding the null value. |

Now, the dataset looks like this-

|  |
| --- |
| Fig:2 – Removed NA value |

1. **Dealing with Data Formats:**

After dealing with null values in the Assault column, we can see that the Assault variable has decimal places in the data. Because we don't want decimal places in the Assault column, we'll round it up. We can round Assault variable by the following R code-

|  |
| --- |
|  |

Here, the argument 0 in the round function means no decimal places. Now, the dataset looks like this-

|  |
| --- |
| Fig.3- Assault column is rounded up |

1. **Smooth Noisy Data:**

We can see that, there is noisy data present in Urban population (%) column which is 570. As this column represents percentage, so it must be between 0 to 100. We need to smooth the noisy data. R code for smoothing this noisy data-

|  |
| --- |
|  |

Here, the fix\_UrbanPopulation(df) function fix the data range (0 to 100) we divide each data by 10 continuously, while it is greater than 100. Now the dataset looks like this-

|  |
| --- |
| Fig.4 – Noise free Urban Population column |

1. **Data Transformation:**

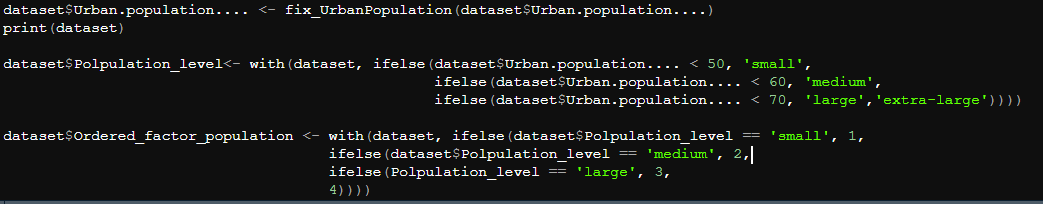
Smoothing, noise removal from data, summarization, generalization, and normalization are all part of the data transformation process. Smoothing, which we studied in IV, will be used in this case (Smooth Noisy Data).

1. **Data Reduction:**

This dataset does not involve any data reduction steps.

1. **Data Discretization and Data Integration:**

We frequently work with data that is gathered through continuous procedures. But there are situations when it's necessary to split up these continuous numbers into smaller chunks. Discrete mapping is the term for this process. As you can see, every attribute in our dataset is of the continuous type. Using logic, we may discretize the data into category kinds and include the column into our dataset. R code for this step-



Here, the with () function take two parameters. One is dataframe, another one is expression. with () function integrates a new column in the dataframe based on the expression. As we have two columns to add for each column we used with () function.

After integrating new column, the dataset looks like this-

|  |
| --- |
| Fig.5 – Adding two rows |

**Discussion & Conclusion:**

At the beginning of the project, we were given a dataset which was totally messy. Null value, noisy data was present in this dataset in Fig.1. After pre-processing the dataset and integrating new column in the dataset, we got totally a clean dataset. The dataset looks like this-

|  |
| --- |
| Fig.6 – Dataset after pre-processing |

Now, we can use this clean, pre-processed dataset for further use.