AMERICAN INTERNATIONAL UNIVERSITY – BANGLADESH

INTRODUCTION TO DATA SCIENCE

SPRING- 2024-2025

Supervise By

DR. Abdus Salam

Assignment: Mid Term Project Report

Group No: 05

Section: E

Submitted By:

NAME	ID
MD. Tashrifull Alam	22-46610-1
Priya Rani Das	22-46358-1
Jeba Shajida	22-46590-1
Sowhanur Rahman Nirob	22-46319-1

TABLE OF CONTENTS

Description of the dataset

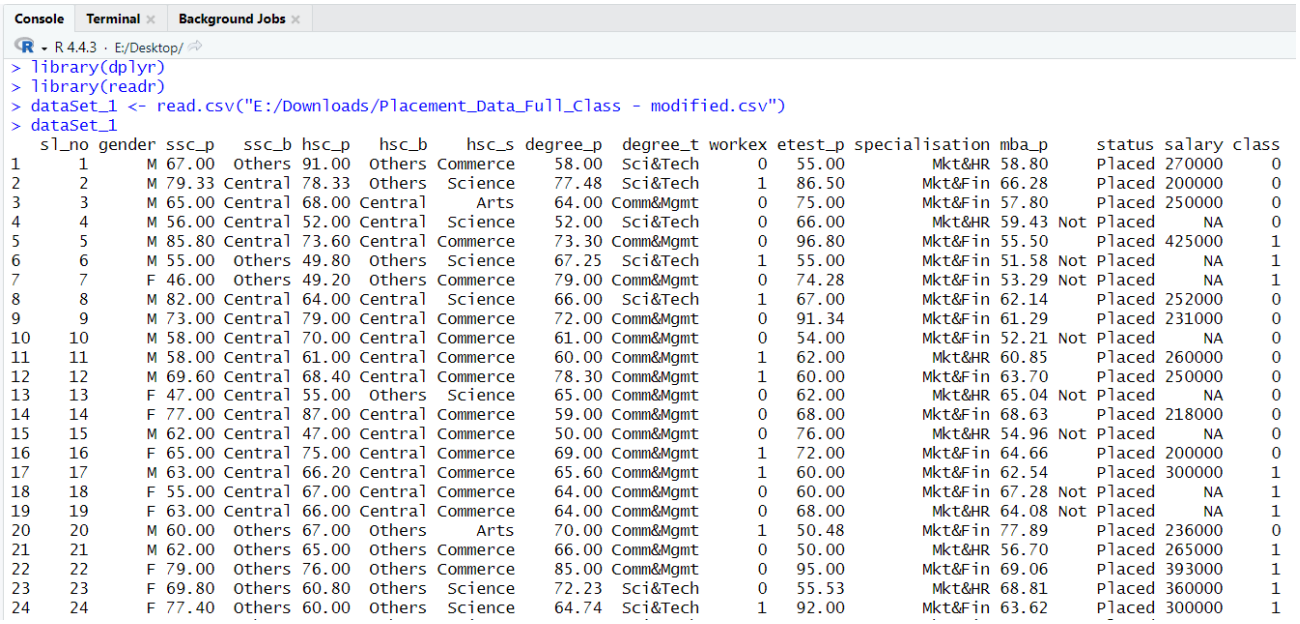
Placement_Data_Full_Class is a synthetic dataset created for binary classification tasks focused on predicting student placement outcomes. It contains information about students enrolled in a campus recruitment program, described by a variety of academic and demographic attributes. These attributes offer a comprehensive profile of each student, including their academic performance in secondary school (ssc_p), higher secondary school (hsc_p), and their area of specialization in higher secondary education (hsc_s). The dataset further includes details of the student's undergraduate degree percentage (degree_p), the type of degree pursued (degree_t), and whether the student had any prior work experience (workex). Additionally, it provides information about whether the student was placed and the salary package offered to them if placed. The target variable, status, indicates the final placement status of the student, where "Placed" denotes successful placement and "Not Placed" indicates otherwise. This dataset combines both numerical and categorical features, making it suitable for exploring and modeling factors that contribute to student employability and placement success in a campus recruitment setting.

Loading the dataset

Code

```
install.packages("dplyr")

library(dplyr)

install.packages("readr")

library(readr)

dataSet_1 <- read.csv("E:/Downloads/Placement_Data_Full_Class -
modified.csv")

dataSet_1

head(dataSet_1, 20)

dataSet_1[1:nrow(dataSet_1), ]
```

Output



```
Console   Terminal   Background Jobs
R 4.4.3 · E:/Desktop/
> library(dplyr)
> library(readr)
> dataSet_1 <- read.csv("E:/Downloads/Placement_Data_Full_Class - modified.csv")
> dataSet_1
   sl_no gender ssc_p    ssc_b hsc_p    hsc_b    hsc_s degree_p  degree_t workex etest_p specialisation mba_p     status salary class
1      1      M 67.00   Others 91.00   Others Commerce    58.00   Sci&Tech      0   55.00         Mkt&HR 58.80     Placed 270000     0
2      2      M 79.33 Central 78.33   Others  Science    77.48   Sci&Tech      1   86.50        Mkt&Fin 66.28     Placed 200000     0
3      3      M 65.00 Central 68.00 Central     Arts    64.00 Comm&Mgmt      0   75.00        Mkt&Fin 57.80     Placed 250000     0
4      4      M 56.00 Central 52.00 Central  Science    52.00   Sci&Tech      0   66.00         Mkt&HR 59.43 Not Placed     NA     0
5      5      M 85.80 Central 73.60 Central Commerce    73.30 Comm&Mgmt      0   96.80        Mkt&Fin 55.50     Placed 425000     1
6      6      M 55.00   Others 49.80   Others  Science    67.25   Sci&Tech      1   55.00        Mkt&Fin 51.58 Not Placed     NA     1
7      7      F 46.00   Others 49.20   Others Commerce    79.00 Comm&Mgmt      0   74.28        Mkt&Fin 53.29 Not Placed     NA     1
8      8      M 82.00 Central 64.00 Central  Science    66.00   Sci&Tech      1   67.00        Mkt&Fin 62.14     Placed 252000     0
9      9      M 73.00 Central 79.00 Central Commerce    72.00 Comm&Mgmt      0   91.34        Mkt&Fin 61.29     Placed 231000     0
10    10      M 58.00 Central 70.00 Central Commerce    61.00 Comm&Mgmt      0   54.00        Mkt&Fin 52.21 Not Placed     NA     0
11    11      M 58.00 Central 61.00 Central Commerce    60.00 Comm&Mgmt      1   62.00         Mkt&HR 60.85     Placed 260000     0
12    12      M 69.60 Central 68.40 Central Commerce    78.30 Comm&Mgmt      1   60.00        Mkt&Fin 63.70     Placed 250000     0
13    13      F 47.00 Central 55.00   Others  Science    65.00 Comm&Mgmt      0   62.00         Mkt&HR 65.04 Not Placed     NA     0
14    14      F 77.00 Central 87.00 Central Commerce    59.00 Comm&Mgmt      0   68.00        Mkt&Fin 68.63     Placed 218000     0
15    15      M 62.00 Central 47.00 Central Commerce    50.00 Comm&Mgmt      0   76.00         Mkt&HR 54.96 Not Placed     NA     0
16    16      F 65.00 Central 75.00 Central Commerce    69.00 Comm&Mgmt      1   72.00        Mkt&Fin 64.66     Placed 200000     0
17    17      M 63.00 Central 66.20 Central Commerce    65.60 Comm&Mgmt      1   60.00        Mkt&Fin 62.54     Placed 300000     1
18    18      F 55.00 Central 67.00 Central Commerce    64.00 Comm&Mgmt      0   60.00        Mkt&Fin 67.28 Not Placed     NA     1
19    19      F 63.00 Central 66.00 Central Commerce    64.00 Comm&Mgmt      0   68.00         Mkt&HR 64.08 Not Placed     NA     1
20    20      M 60.00   Others 67.00   Others     Arts    70.00 Comm&Mgmt      1   50.48        Mkt&Fin 77.89     Placed 236000     0
21    21      M 62.00   Others 65.00   Others Commerce    66.00 Comm&Mgmt      0   50.00         Mkt&HR 56.70     Placed 265000     1
22    22      F 79.00   Others 76.00   Others Commerce    85.00 Comm&Mgmt      0   95.00        Mkt&Fin 69.06     Placed 393000     1
23    23      F 69.80   Others 60.80   Others  Science    72.23   Sci&Tech      0   55.53         Mkt&HR 68.81     Placed 360000     1
24    24      F 77.40   Others 60.00   Others  Science    64.74   Sci&Tech      1   92.00        Mkt&Fin 63.62     Placed 300000     1
```

Description

The dplyr package is installed and loaded to facilitate efficient data manipulation, while the readr package is used for reading structured text data like CSV files. The dataset named **"Placement_Data_Full_Class - modified.csv"** is imported from the specified file path and stored in the object dataset_1. To preview the data, the head() function is used to display the first 20 rows. Following that, the entire dataset is printed by indexing all rows, ensuring the complete contents are visible in the output.

```
no_of_col <- ncol(dataSet_1)
no_of_row <- nrow(dataSet_1)
cat("now of row in the dataset: ", no_of_row)
cat("now of column in the dataset: ", no_of_col)
str(dataSet_1)
```

Output

```
> no_of_col <- ncol(dataSet_1)
> no_of_row <- nrow(dataSet_1)
> cat("now of row in the dataset: ", no_of_row)
now of row in the dataset:  216
> cat("now of column in the dataset: ", no_of_col)
now of column in the dataset:  16
> str(dataSet_1)
'data.frame':   216 obs. of  16 variables:
 $ sl_no         : int  1 2 3 4 5 6 7 8 9 10 ...
 $ gender        : chr  "M" "M" "M" "M" ...
 $ ssc_p         : num  67 79.3 65 56 85.8 ...
 $ ssc_b         : chr  "Others" "Central" "Central" "Central" ...
 $ hsc_p         : num  91 78.3 68 52 73.6 ...
 $ hsc_b         : chr  "Others" "Others" "Central" "Central" ...
 $ hsc_s         : chr  "Commerce" "Science" "Arts" "Science" ...
 $ degree_p      : num  58 77.5 64 52 73.3 ...
 $ degree_t      : chr  "Sci&Tech" "Sci&Tech" "Comm&Mgmt" "Sci&Tech" ...
 $ workex        : int  0 1 0 0 0 1 0 1 0 0 ...
 $ etest_p       : num  55 86.5 75 66 96.8 ...
 $ specialisation: chr  "Mkt&HR" "Mkt&Fin" "Mkt&Fin" "Mkt&HR" ...
 $ mba_p         : num  58.8 66.3 57.8 59.4 55.5 ...
 $ status        : chr  "Placed" "Placed" "Placed" "Not Placed" ...
 $ salary        : int  270000 200000 250000 NA 425000 NA NA 252000 231000 NA ...
 $ class         : int  0 0 0 0 1 1 1 0 0 0 ...
```

Description

This code calculates and prints the number of rows and columns in the dataset dataSet_1 using the nrow and ncol functions. Additionally, the str function provides a detailed overview of the dataset's structure, including the column names, data types, and sample values for each column. The output includes both the dataset's dimensions and a concise summary of its attributes.Here found that 216 rows and 16 column from the dataset.

Extracting Unique Values from Dataset Columns

Code

```
    unique(dataSet_1$gender)
```

Output

```
> unique(dataSet_1$gender)
[1] "M"      "F"      "Female"
```

Description

By applying the unique function to the gender column in the dataset dataSet_1, we retrieve all distinct gender categories present in the data. This allows us to identify the range of gender values recorded. As with other columns, any missing values in this column can be addressed later.

Removing Duplicate Rows from the Dataset

Code

```
remo_dupli_dataset <- dplyr::distinct(dataSet_1)
remo_dupli_dataset

fresh_dataset <- remo_dupli_dataset
cat ("No of row and column after removing duplicate instances: ",
     nrow(remo_dupli_dataset), ncol(remo_dupli_dataset))
```

Output

```
> remo_dupli_dataset <- dplyr::distinct(dataSet_1)
> remo_dupli_dataset
   sl_no gender ssc_p   ssc_b hsc_p   hsc_b    hsc_s degree_p  degree_t workex etest_p specialisation mba_p     status salary class
1      1      M 67.00  Others 91.00  Others Commerce    58.00  Sci&Tech      0   55.00         Mkt&HR 58.80     Placed 270000     0
2      2      M 79.33 Central 78.33  Others  Science    77.48  Sci&Tech      1   86.50        Mkt&Fin 66.28     Placed 200000     0
3      3      M 65.00 Central 68.00 Central     Arts    64.00 Comm&Mgmt      0   75.00        Mkt&Fin 57.80     Placed 250000     0
4      4      M 56.00 Central 52.00 Central  Science    52.00  Sci&Tech      0   66.00         Mkt&HR 59.43 Not Placed     NA     0
5      5      M 85.80 Central 73.60 Central Commerce    73.30 Comm&Mgmt      0   96.80        Mkt&Fin 55.50     Placed 425000     1
6      6      M 55.00  Others 49.80  Others  Science    67.25  Sci&Tech      1   55.00        Mkt&Fin 51.58 Not Placed     NA     1
7      7      F 46.00  Others 49.20  Others Commerce    79.00 Comm&Mgmt      0   74.28        Mkt&Fin 53.29 Not Placed     NA     1
8      8      M 82.00 Central 64.00 Central  Science    66.00  Sci&Tech      1   67.00        Mkt&Fin 62.14     Placed 252000     0
9      9      M 73.00 Central 79.00 Central Commerce    72.00 Comm&Mgmt      0   91.34        Mkt&Fin 61.29     Placed 231000     0
10    10      M 58.00 Central 70.00 Central Commerce    61.00 Comm&Mgmt      0   54.00        Mkt&Fin 52.21 Not Placed     NA     0
11    11      M 58.00 Central 61.00 Central Commerce    60.00 Comm&Mgmt      1   62.00         Mkt&HR 60.85     Placed 260000     0
12    12      M 69.60 Central 68.40 Central Commerce    78.30 Comm&Mgmt      1   60.00        Mkt&Fin 63.70     Placed 250000     0
13    13      F 47.00 Central 55.00  Others  Science    65.00 Comm&Mgmt      0   62.00         Mkt&HR 65.04 Not Placed     NA     0
14    14      F 77.00 Central 87.00 Central Commerce    59.00 Comm&Mgmt      0   68.00        Mkt&Fin 68.63     Placed 218000     0
15    15      M 62.00 Central 47.00 Central Commerce    50.00 Comm&Mgmt      0   76.00         Mkt&HR 54.96 Not Placed     NA     0
50    50      F 50.00  Others 37.00  Others     Arts    52.00    Others      0   65.00         Mkt&HR 56.11 Not Placed     NA     0
51    51      F 75.20 Central 73.20 Central  Science    68.40 Comm&Mgmt      0   65.00         Mkt&HR 62.98     Placed 200000     0
52    52      M 54.40 Central 61.12 Central Commerce    56.20 Comm&Mgmt      0   67.00         Mkt&HR 62.65 Not Placed     NA     0
53    53 Female 40.89  Others 45.83  Others Commerce    53.00 Comm&Mgmt      0   71.20         Mkt&HR 65.49 Not Placed     NA     0
54    54      M 80.00  Others 70.00  Others  Science    72.00  Sci&Tech      0   87.00         Mkt&HR 71.04     Placed 450000     1
55    55      F 74.00 Central 60.00  Others  Science    69.00 Comm&Mgmt      0   78.00         Mkt&HR 65.56     Placed 216000     0
56    56      M 60.40 Central 66.60  Others  Science    65.00 Comm&Mgmt      0   71.00         Mkt&HR 52.71     Placed 220000     0
57    57      M 63.00  Others 71.40  Others Commerce    61.40 Comm&Mgmt      0   68.00        Mkt&Fin 66.88     Placed 240000     0
58    58      M 68.00 Central 76.00 Central Commerce    74.00 Comm&Mgmt      0   80.00        Mkt&Fin 63.59     Placed 360000     1
59    59      M 74.00 Central 62.00  Others  Science    68.00 Comm&Mgmt      0   74.00        Mkt&Fin 57.99     Placed 268000     1
60    60      M 52.60 Central 65.58  Others  Science    72.11  Sci&Tech      0   57.60        Mkt&Fin 56.66     Placed 265000     1
61    61      M 74.00 Central 70.00 Central  Science    72.00 Comm&Mgmt      1   60.00        Mkt&Fin 57.24     Placed 260000     1
62    62      M 84.20 Central 73.40 Central Commerce    66.89 Comm&Mgmt      0   61.60        Mkt&Fin 62.48     Placed 300000     1
 [ reached 'max' / getOption("max.print") -- omitted 154 rows ]
> fresh_dataset <- remo_dupli_dataset
> cat ("No of row and column after removing duplicate instances: ",
+      nrow(remo_dupli_dataset), ncol(remo_dupli_dataset))
No of row and column after removing duplicate instances:  216 16
```

Description

To ensure the uniqueness of each entry, duplicate rows are removed from the dataset dataSet_1 using the distinct function from the dplyr package. The cleaned dataset is stored in remo_dupli_dataset and displayed. This refined dataset is then assigned to a new variable fresh_dataset for further use. Finally, the total number of rows and columns in the updated dataset is printed using the cat function.

Handling Invalid Values

Code

```
deal_invalid_dataset <- fresh_dataset


deal_invalid_dataset$gender <- ifelse(
  toupper(substr(deal_invalid_dataset$gender, 1, 1)) == "M", "M",
  ifelse(
    toupper(substr(deal_invalid_dataset$gender, 1, 1)) == "F", "F",
    "OTHER"
  )
)

unique(deal_invalid_dataset$gender)
print(deal_invalid_dataset)
```

Output

```
> deal_invalid_dataset <- fresh_dataset
> unique(dataSet_1$gender)
[1] "M"      "F"      "Female"
> deal_invalid_dataset$gender <- ifelse(
+   toupper(substr(deal_invalid_dataset$gender, 1, 1)) == "M", "M",
+   ifelse(
+     toupper(substr(deal_invalid_dataset$gender, 1, 1)) == "F", "F",
+     "OTHER"
+   )
+ )
> unique(deal_invalid_dataset$gender)
[1] "M" "F"
> print(deal_invalid_dataset)
   sl_no gender ssc_p    ssc_b hsc_p   hsc_b    hsc_s degree_p  degree_t workex etest_p specialisation mba_p     status salary class
1      1      M 67.00  Others 91.00  Others Commerce    58.00  Sci&Tech      0   55.00         Mkt&HR 58.80     Placed 270000     0
2      2      M 79.33 Central 78.33  Others  Science    77.48  Sci&Tech      1   86.50        Mkt&Fin 66.28     Placed 200000     0
3      3      M 65.00 Central 68.00 Central     Arts    64.00 Comm&Mgmt      0   75.00        Mkt&Fin 57.80     Placed 250000     0
4      4      M 56.00 Central 52.00 Central  Science    52.00  Sci&Tech      0   66.00         Mkt&HR 59.43 Not Placed     NA     0
5      5      M 85.80 Central 73.60 Central Commerce    73.30 Comm&Mgmt      0   96.80        Mkt&Fin 55.50     Placed 425000     1
6      6      M 55.00  Others 49.80  Others  Science    67.25  Sci&Tech      1   55.00        Mkt&Fin 51.58 Not Placed     NA     1
7      7      F 46.00  Others 49.20  Others Commerce    79.00 Comm&Mgmt      0   74.28        Mkt&Fin 53.29 Not Placed     NA     1
8      8      M 82.00 Central 64.00 Central  Science    66.00  Sci&Tech      1   67.00        Mkt&Fin 62.14     Placed 252000     0
51    51      F 75.20 Central 73.20 Central  Science    68.40 Comm&Mgmt      0   65.00         Mkt&HR 62.98     Placed 200000     0
52    52      M 54.40 Central 61.12 Central Commerce    56.20 Comm&Mgmt      0   67.00         Mkt&HR 62.65 Not Placed     NA     0
53    53      F 40.89  Others 45.83  Others Commerce    53.00 Comm&Mgmt      0   71.20         Mkt&HR 65.49 Not Placed     NA     0
54    54      M 80.00  Others 70.00  Others  Science    72.00  Sci&Tech      0   87.00         Mkt&HR 71.04     Placed 450000     1
55    55      F 74.00 Central 60.00  Others  Science    69.00 Comm&Mgmt      0   78.00         Mkt&HR 65.56     Placed 216000     0
56    56      M 60.40 Central 66.60  Others  Science    65.00 Comm&Mgmt      0   71.00         Mkt&HR 52.71     Placed 220000     0
57    57      M 63.00  Others 71.40  Others Commerce    61.40 Comm&Mgmt      0   68.00        Mkt&Fin 66.88     Placed 240000     0
58    58      M 68.00 Central 76.00 Central Commerce    74.00 Comm&Mgmt      0   80.00        Mkt&Fin 63.59     Placed 360000     1
59    59      M 74.00 Central 62.00  Others  Science    68.00 Comm&Mgmt      0   74.00        Mkt&Fin 57.99     Placed 268000     1
60    60      M 52.60 Central 65.58  Others  Science    72.11  Sci&Tech      0   57.60        Mkt&Fin 56.66     Placed 265000     1
61    61      M 74.00 Central 70.00 Central  Science    72.00 Comm&Mgmt      1   60.00        Mkt&Fin 57.24     Placed 260000     1
62    62      M 84.20 Central 73.40 Central Commerce    66.89 Comm&Mgmt      0   61.60        Mkt&Fin 62.48     Placed 300000     1
```

Description

We examined the unique values in various columns of dataSet_1 to identify potential inconsistencies, particularly in categorical fields. A copy of the dataset, deal_invalid_dataset, was created for safe cleaning. We standardized the gender column by categorizing entries starting with "M" as "M", those with "F" as "F", and all others as "OTHER". The changes were verified using the unique () function, and the cleaned dataset was printed for review.

Dealing with Missing Values

Discard instances

Code

```
deal_miss_value_dataset <- fresh_dataset;
colSums(is.na(deal_miss_value_dataset));
which(is.na(deal_miss_value_dataset$ salary))

deal_miss_value_dataset <- na.omit(deal_miss_value_dataset)
deal_miss_value_dataset
colSums(is.na(deal_miss_value_dataset))
```

Output

```
> deal_miss_value_dataset <- fresh_dataset;
> colSums(is.na(deal_miss_value_dataset));
        sl_no        gender         ssc_p         ssc_b         hsc_p         hsc_b         hsc_s       degree_p       degree_t        workex      etest_p specialisation 
            0             0             0             0             0             0             0             0             0             1             0             0 
        mba_p        status        salary         class 
            0             0            67             0 
> which(is.na(deal_miss_value_dataset$ salary))
 [1]   4   6   7  10  13  15  18  19  26  30  32  35  37  42  43  46  47  50  52  53  64  66  69  76  80  83  88  92  94  98 100 101 106 107 110 112 121 131 137 142 145 150 156 159
[45] 160 162 166 168 169 170 171 174 176 180 182 183 185 187 189 190 191 195 199 202 207 209 215
> deal_miss_value_dataset <- na.omit(deal_miss_value_dataset)
> colSums(is.na(deal_miss_value_dataset))
        sl_no        gender         ssc_p         ssc_b         hsc_p         hsc_b         hsc_s       degree_p       degree_t        workex      etest_p specialisation 
            0             0             0             0             0             0             0             0             0             0             0             0 
        mba_p        status        salary         class 
            0             0             0             0 
```

Description

We started by checking for missing values in the dataset using colSums(is.na()), which provided a summary of missing entries in each column. This allowed us to identify how many values were missing per column. Specifically, we pinpointed the rows with missing values in the salary column using the which(is.na()) function. To handle the missing data, we applied the na.omit() function to remove all rows containing any NA values. After cleaning the dataset, we used colSums(is.na()) again to verify that all missing values had been successfully eliminated. The results confirmed that there were no remaining missing values in any column.

Top-Down and Bottom-Up Approach

Code

```
install.packages('tidyr')
library(tidyr)

bottom_up_dataset <- fresh_dataset %>% fill(salary,workex,degree_t, .direction = 'up')
colSums(is.na(bottom_up_dataset))
top_down_dataset <- fresh_dataset %>% fill(salary,workex,degree_t, .direction = 'down')
colSums(is.na(top_down_dataset))
```

Output

```
> library(tidyr)
> bottom_up_dataset <- fresh_dataset %>% fill(salary,workex,degree_t, .direction = 'up')
> colSums(is.na(bottom_up_dataset))
        sl_no        gender         ssc_p         ssc_b         hsc_p         hsc_b         hsc_s       degree_p       degree_t        workex       etest_p specialisation
            0             0             0             0             0             0             0             0             0             0             0              0
        mba_p        status        salary         class
            0             0             0             0
> top_down_dataset <- fresh_dataset %>% fill(salary,workex,degree_t, .direction = 'down')
> colSums(is.na(top_down_dataset))
        sl_no        gender         ssc_p         ssc_b         hsc_p         hsc_b         hsc_s       degree_p       degree_t        workex       etest_p specialisation
            0             0             0             0             0             0             0             0             0             0             0              0
        mba_p        status        salary         class
            0             0             0             0
```

Description

We used two different strategies to handle missing values in the dataset by leveraging the fill() function from the tidyr package. First, in the Bottom-Up approach, we filled missing values in the salary, workex, and degree_t columns by propagating the next non-missing value upwards using .direction = 'up'. We then used colSums(is.na()) to confirm that the missing values were addressed.

Next, we applied the Top-Down approach on the same columns, where missing values were filled by carrying the previous non-missing value downward using .direction = 'down'. Again, we verified the results by checking for any remaining missing values with colSums(is.na()).

Replace by Most Frequent/Average Value

For categorical columns (Mode)

Code

```
deal_miss_value_mode <- fresh_dataset


mode_gender <- names(sort(table(deal_miss_value_mode$gender), decreasing = TRUE))[1]

deal_miss_value_mode$gender[is.na(deal_miss_value_mode$gender)] <- mode_gender

mode_ssc_b <- names(sort(table(deal_miss_value_mode$ssc_b), decreasing = TRUE))[1]

deal_miss_value_mode$ssc_b[is.na(deal_miss_value_mode$ssc_b)] <- mode_ssc_b

mode_hsc_b <- names(sort(table(deal_miss_value_mode$hsc_b), decreasing = TRUE))[1]

deal_miss_value_mode$hsc_b[is.na(deal_miss_value_mode$hsc_b)] <- mode_hsc_b

mode_hsc_s <- names(sort(table(deal_miss_value_mode$hsc_s), decreasing = TRUE))[1]

deal_miss_value_mode$hsc_s[is.na(deal_miss_value_mode$hsc_s)] <- mode_hsc_s

mode_degree_t <- names(sort(table(deal_miss_value_mode$degree_t), decreasing = TRUE))[1]

deal_miss_value_mode$degree_t[is.na(deal_miss_value_mode$degree_t)] <- mode_degree_t

mode_specialisation <- names(sort(table(deal_miss_value_mode$specialisation), decreasing =
TRUE))[1]

deal_miss_value_mode$specialisation[is.na(deal_miss_value_mode$specialisation)] <-
mode_specialisation

mode_status <- names(sort(table(deal_miss_value_mode$status), decreasing = TRUE))[1]

deal_miss_value_mode$status[is.na(deal_miss_value_mode$status)] <- mode_status


deal_miss_value_mode


na_counts <- colSums(is.na(deal_miss_value_mode))

print(na_counts)
```

Output

```
> deal_miss_value_mode <- fresh_dataset
> mode_gender <- names(sort(table(deal_miss_value_mode$gender), decreasing = TRUE))[1]
> deal_miss_value_mode$gender[is.na(deal_miss_value_mode$gender)] <- mode_gender
> mode_ssc_b <- names(sort(table(deal_miss_value_mode$ssc_b), decreasing = TRUE))[1]
> deal_miss_value_mode$ssc_b[is.na(deal_miss_value_mode$ssc_b)] <- mode_ssc_b
> mode_hsc_b <- names(sort(table(deal_miss_value_mode$hsc_b), decreasing = TRUE))[1]
> deal_miss_value_mode$hsc_b[is.na(deal_miss_value_mode$hsc_b)] <- mode_hsc_b
> mode_hsc_s <- names(sort(table(deal_miss_value_mode$hsc_s), decreasing = TRUE))[1]
> deal_miss_value_mode$hsc_s[is.na(deal_miss_value_mode$hsc_s)] <- mode_hsc_s
> mode_degree_t <- names(sort(table(deal_miss_value_mode$degree_t), decreasing = TRUE))[1]
> deal_miss_value_mode$degree_t[is.na(deal_miss_value_mode$degree_t)] <- mode_degree_t
> mode_specialisation <- names(sort(table(deal_miss_value_mode$specialisation), decreasing = TRUE))[1]
> deal_miss_value_mode$specialisation[is.na(deal_miss_value_mode$specialisation)] <- mode_specialisation
> mode_status <- names(sort(table(deal_miss_value_mode$status), decreasing = TRUE))[1]
> deal_miss_value_mode$status[is.na(deal_miss_value_mode$status)] <- mode_status
> na_counts <- colSums(is.na(deal_miss_value_mode))
> print(na_counts)
        sl_no        gender         ssc_p         ssc_b         hsc_p         hsc_b         hsc_s      degree_p      degree_t        workex      etest_p specialisation 
            0             0             0             0             0             0             0             0             0             1             0             0 
        mba_p        status        salary         class 
            0             0            67             0 
```

Description

We addressed missing values in several categorical columns of the fresh_dataset by replacing them with the most frequent value (mode) in each respective column. For each categorical column—gender, ssc_b, hsc_b, hsc_s, degree_t, specialisation, and status—we calculated the mode using the table() and sort() functions. We then replaced any NA values in these columns with the corresponding mode. Finally, we verified that all missing values had been handled by using colSums(is.na()) to summarize the number of remaining missing values across columns.

For numerical columns (mean)

Code

```
deal_miss_value_mean <- fresh_dataset


for(col_name in c("salary", "workex")) {
  if(is.numeric(deal_miss_value_mean[[col_name]])) {


    column_mean <- mean(deal_miss_value_mean[[col_name]], na.rm = TRUE)



    deal_miss_value_mean[[col_name]][is.na(deal_miss_value_mean[[col_name]])] <- column_mean


    deal_miss_value_mean[[col_name]] <- round(deal_miss_value_mean[[col_name]], digits = 0)
  }
}


na_counts <- colSums(is.na(deal_miss_value_mean))
print(na_counts)
```

Output

```
> deal_miss_value_mean <- fresh_dataset
> for(col_name in c("salary", "workex")) {
+   if(is.numeric(deal_miss_value_mean[[col_name]])) {
+
+     column_mean <- mean(deal_miss_value_mean[[col_name]], na.rm = TRUE)
+
+
+     deal_miss_value_mean[[col_name]][is.na(deal_miss_value_mean[[col_name]])] <- column_mean
+
+     deal_miss_value_mean[[col_name]] <- round(deal_miss_value_mean[[col_name]], digits = 0)
+   }
+ }
>
> na_counts <- colSums(is.na(deal_miss_value_mean))
> print(na_counts)
        sl_no        gender         ssc_p         ssc_b         hsc_p         hsc_b         hsc_s      degree_p      degree_t        workex       etest_p
            0             0             0             0             0             0             0             0             0             0             0
specialisation        mba_p        status        salary         class
            0             0             0             0             0
```

Description

We handled missing values in numerical columns—salary and workex—by replacing them with the mean of each respective column. A new dataset, deal_miss_value_mean, was created as a copy of the original fresh_dataset. For each column, we first calculated the mean while excluding missing values (na.rm = TRUE), then replaced the missing entries with this mean value. The updated values were then rounded to the nearest integer for consistency. Finally, we verified that all missing values were addressed by summarizing the remaining NA counts with colSums(is.na()).

Converting Categorical Columns to Numeric Factors

Code

```
dataSet_num <- fresh_dataset
dataSet_num$gender <- factor(dataSet_num$gender, levels =
c("M", "F"), labels = c(1,2));
dataSet_num$ssc_b <-  factor(dataSet_num$ssc_b, levels =
c("Others", "Central"), labels = c(1,2));
dataSet_num$hsc_s <-  factor(dataSet_num$hsc_s, levels =
c("Science", "Commerce","Arts"), labels = c(1,2,3));
dataSet_num$hsc_b <-  factor(dataSet_num$hsc_b, levels =
c("Others", "Central"), labels = c(1,2));
dataSet_num$degree_t <-  factor(dataSet_num$degree_t, levels =
c("Sci&Tech","Comm&Mgmt","Others"), labels = c(1,2,3));
dataSet_num$specialisation <-
factor(dataSet_num$specialisation, levels = c("Mkt&HR",
"Mkt&Fin"), labels = c(1,2));
dataSet_num$status <-  factor(dataSet_num$status, levels =
c("Placed", "Not Placed"), labels = c(1,2));
dataSet_num
```

O*ut*put

```
> dataSet_num <- fresh_dataset
> dataSet_num$gender <- factor(dataSet_num$gender, levels = c("M", "F"), labels = c(1,2));
> dataSet_num$ssc_b <-  factor(dataSet_num$ssc_b, levels = c("Others", "Central"), labels = c(1,2));
> dataSet_num$hsc_s <-  factor(dataSet_num$hsc_s, levels = c("Science", "Commerce","Arts"), labels = c(1,2,3));
> dataSet_num$hsc_b <-  factor(dataSet_num$hsc_b, levels = c("Others", "Central"), labels = c(1,2));
> dataSet_num$degree_t <-  factor(dataSet_num$degree_t, levels = c("Sci&Tech","Comm&Mgmt","Others"), labels = c(1,2,3));
> dataSet_num$specialisation <-  factor(dataSet_num$specialisation, levels = c("Mkt&HR", "Mkt&Fin"), labels = c(1,2));
> dataSet_num$status <-  factor(dataSet_num$status, levels = c("Placed", "Not Placed"), labels = c(1,2));
> dataSet_num
   sl_no gender ssc_p ssc_b hsc_p hsc_b hsc_s degree_p degree_t workex etest_p specialisation mba_p status salary class
1      1      1 67.00     1 91.00     1     2    58.00        1      0   55.00              1 58.80      1 270000     0
2      2      1 79.33     2 78.33     1     1    77.48        1      1   86.50              2 66.28      1 200000     0
3      3      1 65.00     2 68.00     2     3    64.00        2      0   75.00              2 57.80      1 250000     0
4      4      1 56.00     2 52.00     2     1    52.00        1      0   66.00              1 59.43      2     NA     0
5      5      1 85.80     2 73.60     2     2    73.30        2      0   96.80              2 55.50      1 425000     1
6      6      1 55.00     1 49.80     1     1    67.25        1      1   55.00              2 51.58      2     NA     1
7      7      2 46.00     1 49.20     1     2    79.00        2      0   74.28              2 53.29      2     NA     1
8      8      1 82.00     2 64.00     2     1    66.00        1      1   67.00              2 62.14      1 252000     0
9      9      1 73.00     2 79.00     2     2    72.00        2      0   91.34              2 61.29      1 231000     0
10    10      1 58.00     2 70.00     2     2    61.00        2      0   54.00              2 52.21      2     NA     0
11    11      1 58.00     2 61.00     2     2    60.00        2      1   62.00              1 60.85      1 260000     0
12    12      1 69.60     2 68.40     2     2    78.30        2      1   60.00              2 63.70      1 250000     0
13    13      2 47.00     2 55.00     1     1    65.00        2      0   62.00              1 65.04      2     NA     0
14    14      2 77.00     2 87.00     2     2    59.00        2      0   68.00              2 68.63      1 218000     0
```

Description

After handling missing values through various methods, we chose to proceed with the dataset obtained by discarding missing values for further analysis. To prepare this dataset for modeling and numerical analysis, all categorical columns were converted into numeric factors using defined label mappings. Specifically, gender was encoded as 1 (Male) and 2 (Female), ssc_b as 1 (Others) and 2 (Central), hsc_s was categorized into three streams—1 (Science), 2 (Commerce), and 3 (Arts), and hsc_b followed the same encoding as ssc_b. Additionally, degree_t was mapped across three types—1 (Sci&Tech), 2 (Comm&Mgmt), and 3 (Others), specialisation was encoded as 1 (Marketing & HR) and 2 (Marketing & Finance), and finally, status was converted to 1 (Placed) and 2 (Not Placed).

Identifying Outliers

Code

```
detect_outlier <- function(dataframe, columns) {
  for (col in columns) {
    if (is.numeric(dataframe[[col]])) {
      Quantile1 <- quantile(dataframe[[col]], probs = 0.25)
      Quantile3 <- quantile(dataframe[[col]], probs = 0.75)
      IQR <- Quantile3 - Quantile1
      outlier_flags <- dataframe[[col]] > Quantile3 + (IQR * 1.5) |
dataframe[[col]] < Quantile1 - (IQR * 1.5)
      outliers <- dataframe[[col]][outlier_flags]
      if (length(outliers) > 0) {
        cat("Outliers detected in column", col, ":\n")
        print(outliers)
      } else {
        cat("No outliers detected in column", col, "\n")
      }
    } else {
      cat("Column", col, "is not numeric, skipping...\n")
    }
  }
}
detect_outlier <- function(dataframe, columns) {
  for (col in columns) {
    if (is.numeric(dataframe[[col]])) {

      Quantile1 <- quantile(dataframe[[col]], probs = 0.25, na.rm =
TRUE)
      Quantile3 <- quantile(dataframe[[col]], probs = 0.75, na.rm =
TRUE)
      IQR <- Quantile3 - Quantile1


      outlier_flags <- dataframe[[col]] > Quantile3 + (IQR * 1.5) |
        dataframe[[col]] < Quantile1 - (IQR * 1.5)

      outliers <- dataframe[[col]][outlier_flags]
      if (length(outliers) > 0) {
        cat("Outliers detected in column", col, ":\n")
        print(outliers)
      } else {
        cat("No outliers detected in column", col, "\n")
      }
    } else {
      cat("Column", col, "is not numeric, skipping...\n")
    }
  }
}
detect_outlier(fresh_dataset, names(fresh_dataset))
```

Output

```
> detect_outlier(fresh_dataset, names(fresh_dataset))
No outliers detected in column sl_no
Column gender is not numeric, skipping...
Outliers detected in column ssc_p :
[1] 0.76
Column ssc_b is not numeric, skipping...
Outliers detected in column hsc_p :
[1]   97.70   39.00   37.00   40.00 6680.00   92.00   42.16   97.00   42.00
Column hsc_b is not numeric, skipping...
Column hsc_s is not numeric, skipping...
Outliers detected in column degree_p :
[1] 91
Column degree_t is not numeric, skipping...
Outliers detected in column workex :
[1] NA
No outliers detected in column etest_p
Column specialisation is not numeric, skipping...
No outliers detected in column mba_p
Column status is not numeric, skipping...
Outliers detected in column salary :
 [1]     NA 425000     NA     NA     NA     NA     NA     NA 393000     NA     NA     NA     NA 411000     NA     NA     NA     NA     NA     NA 450000     NA
[26]     NA     NA     NA 500000     NA     NA 400000     NA     NA     NA 420000     NA     NA     NA     NA     NA     NA 940000     NA 400000     NA     NA     NA     NA
[51] 400000     NA 690000     NA     NA     NA     NA 500000     NA     NA     NA     NA     NA     NA 500000     NA 650000     NA     NA     NA     NA     NA     NA     NA     NA
[76]     NA     NA     NA     NA     NA 400000     NA
No outliers detected in column class
```

Description

We applied a user-defined function detect_outlier to each column of fresh_dataset. This function uses the Interquartile Range (IQR) method to detect outliers in numeric columns. For each numeric column, it calculates the first (Q1) and third (Q3) quartiles, computes the IQR. The function then reports these values if any are found. If no outliers are present in a column, it explicitly states that. Columns that are not numeric are skipped with a message indicating they are non-numeric.

Removing Outliers

Code

```
remove_outlier <- function(dataframe, columns) {
  for (col in columns) {
    if (is.numeric(dataframe[[col]])) {

      Quantile1 <- quantile(dataframe[[col]], probs
       = 0.25, na.rm = TRUE)
      Quantile3 <- quantile(dataframe[[col]], probs
       = 0.75, na.rm = TRUE)
      IQR <- Quantile3 - Quantile1


      dataframe <- dataframe[!(
        dataframe[[col]] > Quantile3 + (IQR * 1.5) |
          dataframe[[col]] < Quantile1 - (IQR * 1.5)
      ), ]
    }
  }
  return(dataframe)
}
detect_outlier(fresh_dataset, names(fresh_dataset))
without_outlier_data <-
       remove_outlier(fresh_dataset,
       names(fresh_dataset))
without_outlier_data

detect_outlier(without_outlier_data,
       names(without_outlier_data))
```

Output

```
> remove_outlier <- function(dataframe, columns) {
+   for (col in columns) {
+     if (is.numeric(dataframe[[col]])) {
+
+       Quantile1 <- quantile(dataframe[[col]], probs = 0.25, na.rm = TRUE)
+       Quantile3 <- quantile(dataframe[[col]], probs = 0.75, na.rm = TRUE)
+       IQR <- Quantile3 - Quantile1
+
+
+       dataframe <- dataframe[!(
+         dataframe[[col]] > Quantile3 + (IQR * 1.5) |
+           dataframe[[col]] < Quantile1 - (IQR * 1.5)
+       ), ]
+     }
+   }
+   return(dataframe)
+ }
> without_outlier_data <- remove_outlier(fresh_dataset, names(fresh_dataset))
> without_outlier_data
      sl_no gender ssc_p   ssc_b hsc_p   hsc_b    hsc_s degree_p  degree_t workex etest_p specialisation mba_p status salary class
1         1      M 67.00  Others 91.00  Others Commerce    58.00  Sci&Tech      0   55.00         Mkt&HR 58.80 Placed 270000     0
2         2      M 79.33 Central 78.33  Others  Science    77.48  Sci&Tech      1   86.50        Mkt&Fin 66.28 Placed 200000     0
3         3      M 65.00 Central 68.00 Central     Arts    64.00 Comm&Mgmt      0   75.00        Mkt&Fin 57.80 Placed 250000     0
NA       NA   <NA>    NA    <NA>    NA    <NA>     <NA>       NA      <NA>     NA      NA           <NA>    NA   <NA>     NA    NA
NA.1     NA   <NA>    NA    <NA>    NA    <NA>     <NA>       NA      <NA>     NA      NA           <NA>    NA   <NA>     NA    NA
NA.2     NA   <NA>    NA    <NA>    NA    <NA>     <NA>       NA      <NA>     NA      NA           <NA>    NA   <NA>     NA    NA
8         8      M 82.00 Central 64.00 Central  Science    66.00  Sci&Tech      1   67.00        Mkt&Fin 62.14 Placed 252000     0
9         9      M 73.00 Central 79.00 Central Commerce    72.00 Comm&Mgmt      0   91.34        Mkt&Fin 61.29 Placed 231000     0
```

```
> detect_outlier(without_outlier_data, names(without_outlier_data))
Outliers detected in column sl_no :
 [1] NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA
[59] NA NA NA
Column gender is not numeric, skipping...
Outliers detected in column ssc_p :
 [1] NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA
[59] NA NA NA
Column ssc_b is not numeric, skipping...
Outliers detected in column hsc_p :
 [1] NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA
[59] NA NA NA
Column hsc_b is not numeric, skipping...
Column hsc_s is not numeric, skipping...
Outliers detected in column degree_p :
 [1] NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA 83 NA NA NA NA NA NA NA 84 NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA
[59] NA NA NA NA NA
Column degree_t is not numeric, skipping...
Outliers detected in column workex :
 [1] NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA
[59] NA NA NA
Outliers detected in column etest_p :
 [1] NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA
[59] NA NA NA
Column specialisation is not numeric, skipping...
Outliers detected in column mba_p :
 [1] NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA
[59] NA NA NA
Column status is not numeric, skipping...
Outliers detected in column salary :
 [1] NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA
[59] NA NA NA
Outliers detected in column class :
 [1] NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA
[59] NA NA NA
```

Description

We utilized a custom remove_outlier function to eliminate extreme values from all numeric columns in the dataset. This function applies the Interquartile Range (IQR) method to systematically filter out data points that fall outside the typical range, effectively removing anomalies that could impact the accuracy of our analysis. The function iterates over each numeric column, identifies outliers using IQR thresholds, and removes the corresponding rows from the dataset. After the outliers were removed, we re-applied the detect_outlier function to the cleaned dataset to ensure that no further extreme values remained. The output confirms that all numeric columns are now free of outliers.

Normalizing the Dataset

Code

```
normalize_dataset <- fresh_dataset;

min_ssc <- min(normalize_dataset$ssc_p, na.rm = TRUE)
max_ssc <- max(normalize_dataset$ssc_p, na.rm = TRUE)
normalize_dataset$ssc_p <- (normalize_dataset$ssc_p - min_ssc) / (max_ssc - min_ssc);

min_hsc <- min(normalize_dataset$hsc_p, na.rm = TRUE)
max_hsc <- max(normalize_dataset$hsc_p, na.rm = TRUE)
normalize_dataset$hsc_p <- (normalize_dataset$hsc_p - min_hsc) / (max_hsc - min_hsc);


min_degree <- min(normalize_dataset$degree_p, na.rm = TRUE)
max_degree <- max(normalize_dataset$degree_p, na.rm = TRUE)
normalize_dataset$degree_p <- (normalize_dataset$degree_p - min_degree) / (max_degree -
min_degree);

min_etest <- min(normalize_dataset$etest_p, na.rm = TRUE)
```

```
max_etest <- max(normalize_dataset$etest_p, na.rm = TRUE)
normalize_dataset$etest_p <- (normalize_dataset$etest_p - min_etest) / (max_etest - min_etest);

min_mba <- min(normalize_dataset$mba_p , na.rm = TRUE)
max_mba <- max(normalize_dataset$mba_p , na.rm = TRUE)
normalize_dataset$mba_p  <- (normalize_dataset$mba_p  - min_mba) / (max_mba - min_mba);

min_sal <- min(normalize_dataset$salary , na.rm = TRUE)
max_sal <- max(normalize_dataset$salary , na.rm = TRUE)
normalize_dataset$salary  <- (normalize_dataset$salary  - min_sal) / (max_sal - min_sal);

normalize_dataset
```

Output

```
> normalize_dataset <- fresh_dataset;
> min_ssc <- min(normalize_dataset$ssc_p, na.rm = TRUE)
> max_ssc <- max(normalize_dataset$ssc_p, na.rm = TRUE)
> normalize_dataset$ssc_p <- (normalize_dataset$ssc_p - min_ssc) / (max_ssc - min_ssc);
>
> min_hsc <- min(normalize_dataset$hsc_p, na.rm = TRUE)
> max_hsc <- max(normalize_dataset$hsc_p, na.rm = TRUE)
> normalize_dataset$hsc_p <- (normalize_dataset$hsc_p - min_hsc) / (max_hsc - min_hsc);
>
>
> min_degree <- min(normalize_dataset$degree_p, na.rm = TRUE)
> max_degree <- max(normalize_dataset$degree_p, na.rm = TRUE)
> normalize_dataset$degree_p <- (normalize_dataset$degree_p - min_degree) / (max_degree - min_degree);
>
> min_etest <- min(normalize_dataset$etest_p, na.rm = TRUE)
> max_etest <- max(normalize_dataset$etest_p, na.rm = TRUE)
> normalize_dataset$etest_p <- (normalize_dataset$etest_p - min_etest) / (max_etest - min_etest);
>
> min_mba <- min(normalize_dataset$mba_p , na.rm = TRUE)
> max_mba <- max(normalize_dataset$mba_p , na.rm = TRUE)
> normalize_dataset$mba_p  <- (normalize_dataset$mba_p  - min_mba) / (max_mba - min_mba);
>
> min_sal <- min(normalize_dataset$salary , na.rm = TRUE)
> max_sal <- max(normalize_dataset$salary , na.rm = TRUE)
> normalize_dataset$salary  <- (normalize_dataset$salary  - min_sal) / (max_sal - min_sal);
> normalize_dataset
   sl_no gender     ssc_p   ssc_b        hsc_p   hsc_b    hsc_s    degree_p  degree_t workex    etest_p specialisation       mba_p     status      salary class
1      1      M 0.7472924  Others 0.0081288574  Others Commerce 0.195121951  Sci&Tech      0 0.10416667          Mkt&HR 0.284482759     Placed 0.094594595     0
2      2      M 0.8863944 Central 0.0062215866  Others  Science 0.670243902  Sci&Tech      1 0.76041667        Mkt&Fin 0.564842579     Placed 0.000000000     0
3      3      M 0.7247292 Central 0.0046665663 Central     Arts 0.341463415 Comm&Mgmt      0 0.52083333        Mkt&Fin 0.247001499     Placed 0.067567568     0
4      4      M 0.6231949 Central 0.0022580160 Central  Science 0.048780488  Sci&Tech      0 0.33333333         Mkt&HR 0.308095952 Not Placed           NA     0
5      5      M 0.9593863 Central 0.0055095589 Central Commerce 0.568292683 Comm&Mgmt      0 0.97500000        Mkt&Fin 0.160794603     Placed 0.304054054     1
6      6      M 0.6119134  Others 0.0019268403  Others  Science 0.420731707  Sci&Tech      1 0.10416667        Mkt&Fin 0.013868066 Not Placed           NA     1
7      7      F 0.5103791  Others 0.0018365196  Others Commerce 0.707317073 Comm&Mgmt      0 0.50583333        Mkt&Fin 0.077961019 Not Placed           NA     1
8      8      M 0.9165162 Central 0.0040644287 Central  Science 0.390243902  Sci&Tech      1 0.35416667        Mkt&Fin 0.409670165     Placed 0.070270270     0
9      9      M 0.8149819 Central 0.0063224447 Central Commerce 0.536585366 Comm&Mgmt      0 0.86125000        Mkt&Fin 0.377811094     Placed 0.041891892     0
10    10      M 0.6457581 Central 0.0049676351 Central Commerce 0.268292683 Comm&Mgmt      0 0.08333333        Mkt&Fin 0.037481259 Not Placed           NA     0
11    11      M 0.6457581 Central 0.0036128255 Central Commerce 0.243902439 Comm&Mgmt      1 0.25000000         Mkt&HR 0.361319340     Placed 0.081081081     0
12    12      M 0.7766245 Central 0.0047267801 Central Commerce 0.690243902 Comm&Mgmt      1 0.20833333        Mkt&Fin 0.468140930     Placed 0.067567568     0
```

Description

We applied min-max normalization to scale selected numeric columns—ssc_p, hsc_p, degree_p, etest_p, mba_p, and salary—to a common range between 0 and 1. These columns were chosen due to the noticeable differences in their value distributions, particularly salary, which exhibited a wide spread compared to the others. The normalization process was performed by identifying the minimum and maximum values for each column (excluding missing values), and transforming each value using the min-max formula. This scaling technique helps to reduce the influence of features with large numeric ranges, ensuring that all selected variables contribute equally in further analysis or modeling. Other numerical columns were not normalized, as their ranges were relatively consistent and did not pose potential imbalance issues

Descriptive Statistics

Displaying summary of the dataset

Code

```
summary(fresh_dataset);
```

Output

```
> summary(fresh_dataset);
     sl_no       gender      ssc_p           ssc_b       hsc_p           hsc_b       hsc_s     degree_p        degree_t       workex         etest_p     specialisation
 Min.   :  1.00   1:140   Min.   : 0.76   1:100   Min.   :  37.00   1:132   1: 91   Min.   :50.00   1   : 60   Min.   :0.0000   Min.   :50.00   1: 96
 1st Qu.: 54.75   2: 76   1st Qu.:60.36   2:116   1st Qu.:  60.95   2: 84   2:114   1st Qu.:61.00   2   :144   1st Qu.:0.0000   1st Qu.:60.00   2:120
 Median :108.50           Median :67.00           Median :  65.00           3: 11   Median :66.00   3   : 11   Median :0.0000   Median :70.50
 Mean   :108.50           Mean   :66.95           Mean   :  97.06                   Mean   :66.33   NA's:  1   Mean   :0.3442   Mean   :72.02
 3rd Qu.:162.25           3rd Qu.:75.25           3rd Qu.:  73.05                   3rd Qu.:72.00              3rd Qu.:1.0000   3rd Qu.:83.25
 Max.   :216.00           Max.   :89.40           Max.   :6680.00                   Max.   :91.00              Max.   :1.0000   Max.   :98.00
                                                                                                             NA's   :1

     mba_p       status      salary          class
 Min.   :51.21   1:149   Min.   :200000   Min.   :0.0000
 1st Qu.:57.97   2: 67   1st Qu.:240000   1st Qu.:0.0000
 Median :61.95           Median :265000   Median :1.0000
 Mean   :62.26           Mean   :288530   Mean   :0.5556
 3rd Qu.:66.24           3rd Qu.:300000   3rd Qu.:1.0000
 Max.   :77.89           Max.   :940000   Max.   :1.0000
                         NA's   :67
```

Description

By using the summary() function, we obtained a statistical overview of each column in the dataset, including measures like minimum, 1st quartile, median, mean, 3rd quartile, and maximum values. This helps in understanding the distribution, spread, and potential outliers across all features. It provides insights into each attribute's central tendency and variability, ensuring data integrity and highlighting areas that may need further cleaning, transformation, or normalization.

Measure of Central Tendancy

Code

```
calculate_stats <- function(dataset, columns) {
   for (column_name in columns) {
       column_data <- dataset[[column_name]]
       if (is.numeric(column_data)) {
              column_mean <- mean(column_data, na.rm = TRUE)
              column_median <- median(column_data, na.rm = TRUE)
               cat("Mean of column", column_name, "is", column_mean, "\n")
              cat("Median of column", column_name, "is", column_median, "\n")
              cat("\n")
            } else {
              column_mode <- names(sort(table(column_data), decreasing = TRUE))[1]
              cat("Mode of column", column_name, "is", column_mode, "\n")
              cat("\n")
            }
          }
        }
       calculate_stats(fresh_dataset,names(fresh_dataset))
```

Output

```
> calculate_stats(fresh_dataset,columns_to_analyze)
Mean of column ssc_p is 66.95366
Median of column ssc_p is 67
Mode of column ssc_p is 62

Mean of column hsc_p is 97.06403
Median of column hsc_p is 65
Mode of column hsc_p is 63

Mean of column degree_p is 66.33144
Median of column degree_p is 66
Mode of column degree_p is 65

Mean of column etest_p is 72.02139
Median of column etest_p is 70.5
Mode of column etest_p is 60

Mean of column mba_p is 62.26208
Median of column mba_p is 61.95
Mode of column mba_p is 56.7
```

Description

We calculated mean and median for numeric columns (e.g., ssc_p, hsc_p,degree_p) to understand their central tendencies, while mode was calculated for categorical columns (e.g.) to identify the most common categories. These statistics help in better understanding data distributions and ensuring that features contribute meaningfully to analysis and modeling.

Measure of spread

Code

```
columns_to_analyze <- c(
  "gender",
  "ssc_p", "ssc_b",
  "hsc_p","hsc_b" , "degree_p",
  "etest_p", "mba_p"
)

calculate_stats <- function(dataset, columns) {
```

```
       for (col_name in columns) {
              if (is.numeric(dataset[[col_name]])) {
                  column_data <- dataset[[col_name]]
                  column_range <- range(column_data, na.rm = TRUE)
                  column_iqr <- IQR(column_data, na.rm = TRUE)
                  column_sd <- sd(column_data, na.rm = TRUE)
                   column_variance <- var(column_data, na.rm = TRUE)

                   cat("For column", col_name, ":\n")
                   cat("  Range:", column_range[2]- column_range[1], "\n")
                   cat(" IQR:", column_iqr, "\n")
                   cat("  Standard Deviation:", column_sd, "\n")
                  cat(" Variance:", column_variance, "\n")
                  cat("\n")
              }
            }
          }
        calculate_spread(fresh_dataset, columns_to_analyze)
```

Output

```
> calculate_spread(fresh_dataset, columns_to_analyze)
For column ssc_p :
  Range: [ 0.76 , 89.4 ]
  IQR: 14.8925
  Standard Deviation: 11.69634
  Variance: 136.8044

For column hsc_p :
  Range: [ 37 , 6680 ]
  IQR: 12.1
  Standard Deviation: 450.1298
  Variance: 202616.9

For column degree_p :
  Range: [ 50 , 91 ]
  IQR: 11
  Standard Deviation: 7.363667
  Variance: 54.22359

For column etest_p :
  Range: [ 50 , 98 ]
  IQR: 23.25
  Standard Deviation: 13.29606
  Variance: 176.7851

For column mba_p :
  Range: [ 51.21 , 77.89 ]
  IQR: 8.275
  Standard Deviation: 5.824613
  Variance: 33.92611
```

Description

We analyzed the spread for selected numeric columns by calculating key metrics like range, interquartile range (IQR), standard deviation, and variance. These metrics help in understanding data dispersion, identifying potential outliers, and ensuring better model performance. Categorical columns were not analyzed in this process, as spread metrics like range or standard deviation are more meaningful for numeric data and do not apply to categorical variables.

Handling imbalance dataset

Oversampling

Code

```
class_distribution <- table(fresh_dataset$status)
print(class_distribution)
if (class_distribution[1] > class_distribution[2]) {
  majority <- filter(fresh_dataset, status == 1)
  minority <- filter(fresh_dataset, status == 2)
} else {
  majority <- filter(fresh_dataset, status == 2)
  minority <- filter(fresh_dataset, status == 1) }
set.seed(123)
oversampled_minority <- minority %>% sample_n(nrow(majority), replace =
TRUE)
oversampled_data <- bind_rows(majority, oversampled_minority)
table(oversampled_data$status)
oversampled_data
```

```
> class_distribution <- table(fresh_dataset$status)
> print(class_distribution)

  1   2
149  67
> if (class_distribution[1] > class_distribution[2]) {
+   majority <- filter(fresh_dataset, status == 1)
+   minority <- filter(fresh_dataset, status == 2)
+ } else {
+   majority <- filter(fresh_dataset, status == 2)
+   minority <- filter(fresh_dataset, status == 1)
+ }
> set.seed(123)
> oversampled_minority <- minority %>% sample_n(nrow(majority), replace = TRUE)
> oversampled_data <- bind_rows(majority, oversampled_minority)
> table(oversampled_data$status)

  1   2
149 149
> oversampled_data
```

	sl_no	gender	ssc_p	ssc_b	hsc_p	hsc_b	hsc_s	degree_p	degree_t	workex	etest_p	specialisation	mba_p	status	salary	class
1	1	1	67.00	1	91.00	1	2	58.00	1	0	55.00	1	58.80	1	270000	0
2	2	1	79.33	2	78.33	1	1	77.48	1	1	86.50	2	66.28	1	200000	0
3	3	1	65.00	2	68.00	2	3	64.00	2	0	75.00	2	57.80	1	250000	0
4	5	1	85.80	2	73.60	2	2	73.30	2	0	96.80	2	55.50	1	425000	1
5	8	1	82.00	2	64.00	2	1	66.00	1	1	67.00	2	62.14	1	252000	0
6	9	1	73.00	2	79.00	2	2	72.00	2	0	91.34	2	61.29	1	231000	0
7	11	1	58.00	2	61.00	2	2	60.00	2	1	62.00	1	60.85	1	260000	0
8	12	1	69.60	2	68.40	2	2	78.30	2	1	60.00	2	63.70	1	250000	0
9	14	2	77.00	2	87.00	2	2	59.00	2	0	68.00	2	68.63	1	218000	0
10	16	2	65.00	2	75.00	2	2	69.00	2	1	72.00	2	64.66	1	200000	0
11	17	1	63.00	2	66.20	2	2	65.60	2	1	60.00	2	62.54	1	300000	1
12	20	1	60.00	1	67.00	1	3	70.00	2	1	50.48	2	77.89	1	236000	0
13	21	1	62.00	1	65.00	1	2	66.00	2	0	50.00	1	56.70	1	265000	1
14	22	2	79.00	1	76.00	1	2	85.00	2	0	95.00	2	69.06	1	393000	1
15	23	2	69.80	1	60.80	1	1	72.23	1	0	55.53	1	68.81	1	360000	1
16	24	2	77.40	1	60.00	1	1	64.74	1	1	92.00	2	63.62	1	300000	1
17	25	1	76.50	1	97.70	1	1	78.86	1	0	97.40	2	74.01	1	360000	1
18	27	1	71.00	1	79.00	1	2	66.00	2	1	94.00	2	57.55	1	240000	0
19	28	1	63.00	1	67.00	1	2	66.00	2	0	68.00	1	57.69	1	265000	1
20	29	1	76.76	1	76.50	1	2	67.50	2	1	73.35	2	64.15	1	350000	1
21	31	2	64.00	2	73.50	2	2	73.00	2	0	52.00	1	56.70	1	250000	0
22	33	2	61.00	2	81.00	2	2	66.40	2	0	50.89	1	62.21	1	278000	1
23	34	2	87.00	1	65.00	1	1	81.00	2	1	88.00	2	72.78	1	260000	1
24	36	2	69.00	2	78.00	2	2	72.00	2	0	71.00	1	62.74	1	300000	1
25	38	2	79.00	2	76.00	2	1	65.60	1	0	58.00	1	55.47	1	320000	1
26	39	2	73.00	1	58.00	1	1	66.00	2	0	53.70	1	56.86	1	240000	0
27	40	1	81.00	1	68.00	1	1	64.00	1	0	93.00	2	62.56	1	411000	1
28	41	2	78.00	2	77.00	1	2	80.00	2	0	60.00	2	66.72	1	287000	1
29	44	1	87.00	1	87.00	1	2	68.00	2	0	95.00	1	62.90	1	300000	1
30	45	2	77.00	1	73.00	1	2	81.00	2	1	89.00	2	69.70	1	200000	0
31	48	1	63.00	2	60.00	2	2	57.00	2	1	78.00	2	54.55	1	204000	0
32	49	1	63.00	1	62.00	1	2	68.00	2	0	64.00	2	62.46	1	250000	0

Output

Description

To manage the class imbalance in the status column, the majority and minority classes were first determined based on their sample counts. The class with the higher number of records was labeled as the majority, while the one with fewer records was identified as the minority. To balance the dataset, oversampling was performed by replicating the minority class using the sample_n() function with replacement until its size matched that of the majority class. Finally, the bind_rows() function was used to combine both classes into a single, balanced dataset.

Undersampling

Code

```
undersampled_majority <- majority %>% sample_n(nrow(minority), replace = FALSE)

undersampled_data <- bind_rows(undersampled_majority, minority)

table(undersampled_data$status)

undersampled_data

fresh_dataset <- oversampled_data
```

Output

```
> undersampled_majority <- majority %>% sample_n(nrow(minority), replace = FALSE)
> undersampled_data <- bind_rows(undersampled_majority, minority)
> table(undersampled_data$status)

 1  2
67 67
> undersampled_data
   sl_no gender ssc_p ssc_b hsc_p hsc_b hsc_s degree_p degree_t workex etest_p specialisation mba_p status salary class
1     22      2 79.00     1 76.00     1     2    85.00        2      0   95.00              2 69.06      1 393000     1
2    152      1 65.00     2 65.00     2     2    75.00        2      0   83.00              2 58.87      1 270000     1
3    196      1 66.00     2 76.00     2     2    72.00        2      1   84.00              1 58.95      1 275000     1
4    128      2 72.00     1 60.00     1     1    69.00        2      0   55.50              1 58.40      1 250000     0
5    186      2 88.00     2 72.00     2     1    78.00        3      0   82.00              1 71.43      1 252000     0
6    147      1 62.00     2 63.00     1     1    66.00        2      0   85.00              1 55.14      1 233000     0
7    155      1 53.00     2 63.00     1     1    60.00        2      1   70.00              2 53.20      1 250000     0
8    117      1 68.20     2 72.80     2     2    66.60        2      1   96.00              2 70.85      1 300000     1
9     44      1 87.00     1 87.00     1     2    68.00        2      0   95.00              1 62.90      1 300000     1
10    39      2 73.00     1 58.00     1     1    66.00        2      0   53.70              1 56.86      1 240000     0
11    40      1 81.00     1 68.00     1     1    64.00        1      0   93.00              2 62.56      1 411000     1
12    11      1 58.00     2 61.00     2     2    60.00        2      1   62.00              1 60.85      1 260000     0
13    61      1 74.00     2 70.00     2     1    72.00        2      1   60.00              2 57.24      1 260000     1
14   213      1 67.00     1 67.00     1     2    73.00        2      1   59.00              2 69.72      1 295000     1
15   130      1 76.70     2 89.70     1     2    66.00        2      1   90.00              2 68.55      1 250000     0
16    82      1 81.70     1 63.00     1     1    67.00        2      1   86.00              2 70.20      1 300000     1
17    95      1 58.00     2 62.00     2     2    64.00        2      0   53.88              2 54.97      1 260000     1
18     5      1 85.80     2 73.60     2     2    73.30        2      0   96.80              2 55.50      1 425000     1
19   108      1 82.00     1 90.00     1     2    83.00        2      0   80.00              1 73.52      1 200000     0
20   163      1 74.20     2 87.60     1     2    77.25        2      1   75.20              2 66.06      1 285000     1
21    38      2 79.00     2 76.00     2     1    65.60        1      0   58.00              1 55.47      1 320000     1
22    12      1 69.60     2 68.40     2     2    78.30        2      1   60.00              2 63.70      1 250000     0
23    79      1 84.00     1 90.90     1     1    64.50        1      0   86.04              2 59.42      1 270000     1
24   126      2 84.00     2 73.00     2     2    73.00        2      0   75.00              2 73.33      1 350000     1
25   122      2 64.00     2 67.00     1     1    69.60        1      1   55.67              1 71.49      1 250000     0
26    67      1 83.00     1 74.00     1     1    66.00        2      0   68.92              1 58.46      1 275000     1
```

Description

To tackle the class imbalance in the status column, an undersampling technique was used. The majority class was randomly reduced to match the size of the minority class by applying the sample_n() function without replacement. Afterward, the minority class and the newly undersampled majority class were merged using bind_rows(), creating a balanced dataset with equal representation of both classes. This balanced data was then saved into fresh_dataset, providing a stronger foundation for model training and analysis.