AMERICAN INTERNATIONAL UNIVERSITY-BANGLADESH (AIUB)

FACULTY OF SCIENCE & TECHNOLOGY

DEPARTMENT OF COMPUTER SCIENCE

INTRODUCTION TO DATA SCIENCE

FALL 2024-2025

Section:___C___ Group:___10___

MID TERM

PROJECT 1 REPORT

Supervised By

TOHEDUL ISLAM

Submitted By:

NAME	ID
MD. SHOHANUR RAHMAN SHOHAN	22-46013-1
A. F. M. RAFIUL HASSAN	22-47048-1
HORISH DAS PRIYO	21-44816-1

Description of the Dataset:

The dataset contains information on individuals and their loan details, including 14 attributes: "person_age" (age), "person_gender" (gender), "person_education" (education level), "person_income" (annual income), "person_emp_exp" (employment experience in years), "person_home_ownership" (home ownership type), "loan_amnt" (loan amount), "loan_intent" (loan purpose), "loan_int_rate" (loan interest rate), "loan_percent_income" (income percentage for loan repayment), "cb_person_cred_hist_length" (credit history length), "credit_score" (credit score), "previous_loan_defaults_on_file" (previous loan defaults), and "loan_status" (loan approval status). The dataset includes both numerical and categorical data, with some missing values and potential outliers.

Load the Dataset:

Code:

```
install.packages("readxl")

library(readxl)

mainData <- read_excel("D:/MID-Data-Science-Project/Project-
1/Midterm_Dataset_Section(C).xlsx")

print(mainData)
```

Output:

```
R 4.4.1 · D:/MID-Data-Science-Project/Project-1/
> library(readxl)
> mainData <- read_excel("D:/MID-Data-Science-Project/Project-1/Midterm_Dataset_Section(C).xlsx")
> print(mainData)
# A tibble: 201 × 14
   person_age person_gender person_education person_income person_emp_exp person_home_ownership loan_amnt loan_intent
        <dbl> <chr>         <chr>                  <dbl>          <dbl> <chr>                     <dbl> <chr>
 1         21 female        Master                 71948              0 RENT                      35000 PERSONAL
 2         21 female        High School            12282              0 OWN                        1000 EDUCATION
 3         25 female        High School            12438              3 MORTGAGE                   5500 MEDICAL
 4         23 female        Bachelor               79753              0 RENT                      35000 MEDICAL
 5         24 male          Master                 66135              1 RENTT                     35000 MEDICAL
 6         NA female        High School            12951              0 OWN                        2500 VENTURE
 7         22 female        Bachelor                  NA              1 RENT                      35000 EDUCATION
 8         24 NA            High School            95550              5 RENT                      35000 MEDICAL
 9         22 female        NA                    100684              3 RENT                      35000 PERSONAL
10         21 female        High School            12739              0 OWN                        1600 VENTURE
# ℹ 191 more rows
# ℹ 6 more variables: loan_int_rate <dbl>, loan_percent_income <dbl>, cb_person_cred_hist_length <dbl>, credit_score <dbl>,
#   previous_loan_defaults_on_file <chr>, loan_status <dbl>
# ℹ Use `print(n = ...)` to see more rows
>
```

Description:

Load the dataset into mainData.

Structure of the Dataset:

Code:

```
numberOfCol <- ncol(mainData)
numberOfRow <- nrow(mainData)
cat("Number of Column: ", numberOfCol, "\n")
cat("Number of Row: ", numberOfRow, "\n")
str(mainData)
```

Output:

```
> numberOfCol <- ncol(mainData)
> numberOfRow <- nrow(mainData)
> cat("Number of Column: ", numberOfCol, "\n")
Number of Column:  14
> cat("Number of Row: ", numberOfRow, "\n")
Number of Row:  201
> str(mainData)
tibble [201 × 14] (S3: tbl_df/tbl/data.frame)
 $ person_age                   : num [1:201] 21 21 25 23 24 NA 22 24 22 21 ...
 $ person_gender                : chr [1:201] "female" "female" "female" "female" ...
 $ person_education             : chr [1:201] "Master" "High School" "High School" "Bachelor" ...
 $ person_income                : num [1:201] 71948 12282 12438 79753 66135 ...
 $ person_emp_exp               : num [1:201] 0 0 3 0 1 0 1 5 3 0 ...
 $ person_home_ownership        : chr [1:201] "RENT" "OWN" "MORTGAGE" "RENT" ...
 $ loan_amnt                    : num [1:201] 35000 1000 5500 35000 35000 2500 35000 35000 35000 1600 ...
 $ loan_intent                  : chr [1:201] "PERSONAL" "EDUCATION" "MEDICAL" "MEDICAL" ...
 $ loan_int_rate                : num [1:201] 16 11.1 12.9 15.2 14.3 ...
 $ loan_percent_income          : num [1:201] 0.49 NA 0 0.44 0.53 0.19 0.37 0.37 0.35 0.13 ...
 $ cb_person_cred_hist_length   : num [1:201] 3 2 3 2 4 2 3 4 2 3 ...
 $ credit_score                 : num [1:201] 561 504 635 675 586 532 701 585 544 640 ...
 $ previous_loan_defaults_on_file: chr [1:201] "No" "Yes" "No" "No" ...
 $ loan_status                  : num [1:201] 1 0 1 1 1 1 1 1 NA 1 ...
>
```

Description:

The dataset consists of 201 instances and 14 attributes. Each attribute, along with its type and instances, is detailed in the summary.

Remove Duplicate:

Code:

```
install.packages("dplyr")
library(dplyr)
mainData <- distinct(mainData)
```

```
mainData
numberOfCol <- ncol(mainData)
numberOfRow <- nrow(mainData)
cat("Number of Column: ", numberOfCol, "\n")
cat("Number of Row: ", numberOfRow, "\n")
```

Output:

```
> library(dplyr)
> mainData <- distinct(mainData)
> mainData
# A tibble: 200 × 14
   person_age person_gender person_education person_income person_emp_exp person_home_ownership loan_amnt loan_intent
        <dbl> <chr>         <chr>                  <dbl>          <dbl> <chr>                     <dbl> <chr>
 1         21 female        Master                 71948              0 RENT                      35000 PERSONAL
 2         21 female        High School            12282              0 OWN                        1000 EDUCATION
 3         25 female        High School            12438              3 MORTGAGE                   5500 MEDICAL
 4         23 female        Bachelor               79753              0 RENT                      35000 MEDICAL
 5         24 male          Master                 66135              1 RENTT                     35000 MEDICAL
 6         NA female        High School            12951              0 OWN                        2500 VENTURE
 7         22 female        Bachelor                  NA              1 RENT                      35000 EDUCATION
 8         24 NA            High School            95550              5 RENT                      35000 MEDICAL
 9         22 female        NA                    100684              3 RENT                      35000 PERSONAL
10         21 female        High School            12739              0 OWN                        1600 VENTURE
# i 190 more rows
# i 6 more variables: loan_int_rate <dbl>, loan_percent_income <dbl>, cb_person_cred_hist_length <dbl>, credit_score <db
l>,
#   previous_loan_defaults_on_file <chr>, loan_status <dbl>
# i Use `print(n = ...)` to see more rows
> numberOfCol <- ncol(mainData)
> numberOfRow <- nrow(mainData)
> cat("Number of Column: ", numberOfCol, "\n")
Number of Column:  14
> cat("Number of Row: ", numberOfRow, "\n")
Number of Row:  200
>
```

Description:

Using distinct() function we remove duplicate row. After removing duplicate Number of attributes 14 and 201 instances are displayed.

Unique Categories:

Code:

```
unique(mainData$person_age)
unique(mainData$person_emp_exp)
unique(mainData$person_home_ownership)
```

Output:

```
> unique(mainData$person_age)
 [1]  21  25  23  24  NA  22 230  26 350 144
> unique(mainData$person_emp_exp)
 [1]   0   3   1   5   4   2   7   6 125   8 121
> unique(mainData$person_home_ownership)
[1] "RENT"     "OWN"      "MORTGAGE" "RENTT"    "OOWN"     "OTHER"
>
```

Description:

We use unique functions to identify unique attributes. In the person_age and person_emp_exp categories, there are various numeric values and some missing values (NA). We'll address the missing values later. For the person_home_ownership attribute, the unique values are: "RENT", "OWN", "MORTGAGE", "RENTT", "OOWN", and "OTHER". The values "RENTT" and "OOWN" are invalid for this attribute.

Handle Invalid Values:

Code:

```
handleInvaData <- mainData
handleInvaData$person_home_ownership <- ifelse(
  substr(handleInvaData$person_home_ownership, 1, 2) == "OT", "OTHER",
  ifelse(
    substr(handleInvaData$person_home_ownership, 1, 1) == "M", "MORTGAGE",
    ifelse(
      substr(handleInvaData$person_home_ownership, 1, 1) == "R", "RENT",
      ifelse(
        substr(handleInvaData$person_home_ownership, 1, 1) == "O", "OWN",
        NA
      )
    )
  )
)
```

Output:

```
> handleInvaData <- mainData
> handleInvaData$person_home_ownership <- ifelse(
+   substr(handleInvaData$person_home_ownership, 1, 2) == "OT", "OTHER",
+   ifelse(
+     substr(handleInvaData$person_home_ownership, 1, 1) == "M", "MORTGAGE",
+     ifelse(
+       substr(handleInvaData$person_home_ownership, 1, 1) == "R", "RENT",
+       ifelse(
+         substr(handleInvaData$person_home_ownership, 1, 1) == "O", "OWN",
+         NA
+       )
+     )
+   )
+ )
> unique(handleInvaData$person_home_ownership)
[1] "RENT"     "OWN"      "MORTGAGE" "OTHER"
```

Description:

Previously some invalid values were found on the person_home_ownership attribute. We assumed that if the starting character was “OT” means “OTHER”, “M” means “MORTGAGE”, “R” means “RENT”, “O” means “OWN” and ignored further characters of each instance in who attribute. And after replacing invalid values, there’s no extra invalid values found using unique() function.

Annotating Dataset:

Code:

```
annoData <- handleInvaData

annoData$person_gender <- factor(annoData$person_gender, levels = c("female",
"male"), labels = c(0, 1))

annoData$person_gender

annoData$person_education <- factor(annoData$person_education, levels =
c("High School", "Bachelor", "Master", "Associate", "Doctorate"), labels =
c(1, 2, 3, 4, 5))

annoData$person_education

annoData$person_home_ownership <- factor(annoData$person_home_ownership,
levels = c("RENT", "OWN", "MORTGAGE", "OTHER"), labels = c(1, 2, 3, 4))

annoData$person_home_ownership
```

```
annoData$loan_intent <- factor(annoData$loan_intent, levels = c("PERSONAL",
"EDUCATION", "MEDICAL", "VENTURE", "HOMEIMPROVEMENT", "DEBTCONSOLIDATION"),
labels = c(1, 2, 3, 4, 5, 6))

annoData$loan_intent

annoData$previous_loan_defaults_on_file <-
factor(annoData$previous_loan_defaults_on_file, levels = c("No", "Yes"),
labels = c(0,1))

annoData$previous_loan_defaults_on_file

annoData
```

Output:

```
> annoData <- handleInvaData
> annoData$person_gender <- factor(annoData$person_gender, levels = c("female", "male"), labels = c(0, 1))
> annoData$person_gender
  [1] 0    0    0    0    1    0    0    <NA> 0    0    0    0    1    1    0    0    <NA> 0    1    0    1    1    0
 [24] 0    1    0    0    1    0    0    1    1    1    0    0    1    0    1    1    0    1    0    1    1    1    1
 [47] 0    1    0    1    0    1    1    0    1    1    1    1    1    1    0    0    0    0    1    1    1    0    1
 [70] 0    1    1    0    0    1    1    1    0    1    1    1    1    0    1    1    1    1    0    0    0    1    1
 [93] 1    1    0    1    0    1    1    0    1    1    1    0    1    1    0    0    1    1    1    1    0    1    1
[116] 1    1    1    0    0    0    0    1    1    1    0    1    1    1    1    1    1    1    0    0    0    0    1
[139] 0    0    1    1    1    0    1    0    1    0    1    0    1    1    0    1    1    0    1    1    1    1    1
[162] 0    1    0    1    1    1    1    1    1    1    1    1    0    0    1    1    1    1    0    1    1    1    1
[185] 1    0    1    1    <NA> 1    0    1    1    0    0    1    <NA> 0    1    0
Levels: 0 1
> annoData$person_education <- factor(annoData$person_education, levels = c("High School", "Bachelor", "Master", "Assoc
iate", "Doctorate"), labels = c(1, 2, 3, 4, 5))
> annoData$person_education
  [1] 3    1    1    2    3    1    2    1    <NA> 1    1    4    2    3    4    <NA> 2    1    2    3    2    1    3
 [24] 2    2    1    4    1    2    2    1    2    2    1    4    4    2    1    2    1    2    1    2    1    3    2
 [47] 2    2    1    3    2    1    2    4    2    1    4    2    3    4    2    2    1    4    3    4    2    2    1
 [70] 2    1    4    1    4    2    4    1    2    2    1    2    2    2    4    2    3    4    2    2    2    4    4
 [93] 3    1    1    4    2    2    3    4    2    2    1    1    4    1    2    2    4    1    2    3    2    4    1
[116] 1    1    2    3    4    2    4    1    3    2    4    2    4    1    2    2    1    1    1    4    2    3    4
[139] 4    2    2    1    1    5    2    1    4    1    3    4    4    2    1    3    2    4    1    4    4    4    1
[162] 4    2    3    1    2    4    1    2    2    1    4    1    1    2    2    1    1    1    1    4    3    1    4
[185] 2    3    4    2    3    3    4    1    2    4    1    4    4    2    2    2
Levels: 1 2 3 4 5
> annoData$person_home_ownership <- factor(annoData$person_home_ownership, levels = c("RENT", "OWN", "MORTGAGE", "OTHE
R"), labels = c(1, 2, 3, 4))
> annoData$person_home_ownership
  [1] 1 2 3 1 1 2 1 1 1 2 1 2 1 1 1 3 1 1 1 3 1 1 1 2 1 1 3 2 1 4 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1
 [60] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1
[119] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1
[178] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
Levels: 1 2 3 4

> annoData$loan_intent <- factor(annoData$loan_intent, levels = c("PERSONAL", "EDUCATION", "MEDICAL", "VENTURE", "HOMEI
MPROVEMENT", "DEBTCONSOLIDATION"), labels = c(1, 2, 3, 4, 5, 6))
> annoData$loan_intent
  [1] 1 2 3 3 3 4 2 3 1 4 4 5 4 2 2 6 2 3 6 2 4 1 2 1 1 5 4 1 5 2 3 4 5 6 3 2 2 6 5 6 3 4 4 2 5 2 5 1 3 6 6 6 1 4 2 1 4
1 1
 [60] 4 4 4 2 3 2 2 5 5 2 3 3 3 2 2 4 2 4 1 6 4 2 4 6 2 6 2 6 5 6 2 1 4 1 4 3 2 3 4 3 5 5 6 3 3 2 1 1 2 5 6 5 5 5 2 3 6
6 3
[119] 2 3 2 2 2 4 2 6 4 1 6 5 6 4 4 4 2 6 2 6 2 2 3 2 1 6 2 5 4 2 4 4 1 4 2 5 2 2 1 6 3 2 1 1 1 6 6 3 5 2 5 1 1 2 3 5 2
6 4
[178] 2 6 2 2 2 6 2 2 4 1 6 6 5 1 2 4 1 1 2 4 1 2 3
Levels: 1 2 3 4 5 6
> annoData$previous_loan_defaults_on_file <- factor(annoData$previous_loan_defaults_on_file, levels = c("No", "Yes"), l
abels = c(0,1))
> annoData$previous_loan_defaults_on_file
  [1] 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 1 0 0 1 0 0 0 0 0 0 0 1 0 1 0 0 0 0 0 0 0 0 0 1 1 0 1 0 0 0 0 0 0 0 0 1 1
0 1
 [60] 1 1 1 0 0 0 0 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 1 0 0 0 1 0 0 0 0 1 1 0 1 1 1 0 1 1 0 0 1 0 0 0 0 1 1 0 1 0
0 0
[119] 0 0 0 0 0 1 0 0 1 0 0 0 0 0 0 0 0 0 0 1 0 1 0 1 1 0 0 0 0 0 0 0 1 1 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 1 0 0 0 0 0 1
0 1
[178] 1 0 1 0 0 1 0 1 1 0 0 0 0 0 0 0 0 0 0 0 1 0 0
Levels: 0 1
> annoData
# A tibble: 200 × 14
   person_age person_gender person_education person_income person_emp_exp person_home_ownership loan_amnt loan_intent
        <dbl> <fct>         <fct>                  <dbl>          <dbl> <fct>                     <dbl> <fct>
 1         21 0             3                      71948              0 1                         35000 1
 2         21 0             1                      12282              0 2                          1000 2
 3         25 0             1                      12438              3 3                          5500 3
 4         23 0             2                      79753              0 1                         35000 3
 5         24 1             3                      66135              1 1                         35000 3
 6         NA 0             1                      12951              0 2                          2500 4
 7         22 0             2                         NA              1 1                         35000 2
 8         24 NA            1                      95550              5 1                         35000 3
 9         22 0             NA                    100684              3 1                         35000 1
10         21 0             1                      12739              0 2                          1600 4
# ℹ 190 more rows
# ℹ 6 more variables: loan_int_rate <dbl>, loan_percent_income <dbl>, cb_person_cred_hist_length <dbl>, credit_score <db
l>,
#   previous_loan_defaults_on_file <fct>, loan_status <dbl>
# ℹ Use `print(n = ...)` to see more rows
```

Description:

After handle invalid values, handleInvadata is stored into annoData to convert categorical data into neumerical data using factor() function.

For person_gender attribute: "female" → 0 & "male" → 1

For person_education attribute: "High School" → 1, "Bachelor" → 2, "Master" → 3, "Associate" → 4, "Doctorate" →5

For person_home_ownership attribute: "RENT" → 1, "OWN" → 2, "MORTGAGE → 3", "OTHER" → 4

For loan_intent attribute: "PERSONAL" → 1, "EDUCATION" → 2, "MEDICAL" →3, "VENTURE" → 4, "HOMEIMPROVEMENT" → 5, "DEBTCONSOLIDATION" → 6

For previous_loan_defaults_on_file attribute: "No" → 0, "Yes" → 1

Data Statistics:

Code:

```
summary(annoData)
```

Output:

```
> summary(annoData)
   person_age     person_gender person_education person_income     person_emp_exp   person_home_ownership   loan_amnt
 Min.   : 21.00   0   : 77      1   :58          Min.   :  12282   Min.   :  0.00   1:188                 Min.   : 1000
 1st Qu.: 22.00   1   :119      2   :70          1st Qu.:  60342   1st Qu.:  0.00   2:  7                 1st Qu.:10000
 Median : 23.00   NA's:  4      3   :23          Median :  86048   Median :  1.00   3:  4                 Median :25000
 Mean   : 27.42                 4   :46          Mean   : 150236   Mean   :  2.77   4:  1                 Mean   :20493
 3rd Qu.: 25.00                 5   : 1          3rd Qu.: 241074   3rd Qu.:  3.00                         3rd Qu.:28000
 Max.   :350.00                 NA's: 2          Max.   :3138998   Max.   :125.00                         Max.   :35000
 NA's   :4                                       NA's   :4
 loan_intent loan_int_rate   loan_percent_income cb_person_cred_hist_length  credit_score   previous_loan_defaults_on_f
ile
 1:30        Min.   : 5.42   Min.   :0.0000      Min.   :2.00               Min.   :484.0   0:149
 2:56        1st Qu.:10.65   1st Qu.:0.0900      1st Qu.:2.00               1st Qu.:594.8   1: 51
 3:26        Median :11.85   Median :0.2300      Median :3.00               Median :629.0
 4:34        Mean   :12.30   Mean   :0.2284      Mean   :2.99               Mean   :628.2
 5:23        3rd Qu.:14.45   3rd Qu.:0.3400      3rd Qu.:4.00               3rd Qu.:664.2
 6:31        Max.   :20.00   Max.   :0.5300      Max.   :4.00               Max.   :807.0
                             NA's   :1
  loan_status
 Min.   :0.0000
 1st Qu.:0.0000
 Median :1.0000
 Mean   :0.6142
 3rd Qu.:1.0000
 Max.   :1.0000
 NA's   :3
>
```

Description:

After converting categorical data into numerical descriptive Statistics has been shown using summary() function.

Find Missing Values:

Code:

```
numOfMissValue <- colSums(is.na(annoData))
numOfMissValue
```

Output:

```
> numOfMissValue <- colSums(is.na(annoData))
> numOfMissValue
                    person_age                  person_gender               person_education                  person_income
                             4                              4                              2                              4
                person_emp_exp          person_home_ownership                    loan_amnt                    loan_intent
                             0                              0                              0                              0
                 loan_int_rate            loan_percent_income     cb_person_cred_hist_length                   credit_score
                             0                              1                              0                              0
previous_loan_defaults_on_file                    loan_status
                             0                              3
> |
```

Description:

We used colSums() and is.na() to count the number of missing values in each attribute.

Handle Missing Values:

- ## By Discard Instances:

  ### Code:

  ```
  newData1 <- na.omit(annoData)
  numOfMissValue <- colSums(is.na(newData1))
  numOfMissValue
  ```

  ### Output:

  ```
  > newData1 <- na.omit(annoData)
  > numOfMissValue <- colSums(is.na(newData1))
  > numOfMissValue
                    person_age                  person_gender               person_education                  person_income
                             0                              0                              0                              0
                person_emp_exp          person_home_ownership                    loan_amnt                    loan_intent
                             0                              0                              0                              0
                 loan_int_rate            loan_percent_income     cb_person_cred_hist_length                   credit_score
                             0                              0                              0                              0
  previous_loan_defaults_on_file                    loan_status
                             0                              0
  > |
  ```

  Description:

  We removed all instances with null values using the na.omit() function.

- **Replace by Most Frequent/Average Value:**

Code:

```
newData2 <- annoData
newData2$person_gender <- as.numeric(newData2$person_gender)
newData2$person_education <- as.numeric(newData2$person_education)
for (col_name in names(newData2)) {
  if (is.numeric(newData2[[col_name]])) {
    column_mean <- mean(newData2[[col_name]], na.rm = TRUE)
    newData2[[col_name]][is.na(newData2[[col_name]])] <- column_mean
    newData2[[col_name]] <- round(newData2[[col_name]], digits = 0)
  }
}
numOfMissValue <- colSums(is.na(newData2))
numOfMissValue
```

Output:

```
> newData2 <- annoData
> newData2$person_gender <- as.numeric(newData2$person_gender)
> newData2$person_education <- as.numeric(newData2$person_education)
> for (col_name in names(newData2)) {
+   if (is.numeric(newData2[[col_name]])) {
+     column_mean <- mean(newData2[[col_name]], na.rm = TRUE)
+     newData2[[col_name]][is.na(newData2[[col_name]])] <- column_mean
+     newData2[[col_name]] <- round(newData2[[col_name]], digits = 0)
+   }
+ }
> numOfMissValue <- colSums(is.na(newData2))
> numOfMissValue
                    person_age                  person_gender               person_education                  person_income
                             0                              0                              0                              0
                person_emp_exp          person_home_ownership                    loan_amnt                    loan_intent
                             0                              0                              0                              0
                 loan_int_rate            loan_percent_income     cb_person_cred_hist_length                  credit_score
                             0                              0                              0                              0
 previous_loan_defaults_on_file                    loan_status
                             0                              0
```

Description:

After converting the person_gender and person_education column to numerical values, we calculated the average value for each numerical attribute. We then rounded these averages and used them to replace the missing values in each column.

Find Outliers:

Code:

```
find_outliers_iqr <- function(column) {
  Q1 <- quantile(column, 0.25, na.rm = TRUE)
  Q3 <- quantile(column, 0.75, na.rm = TRUE)
  IQR <- Q3 - Q1
  lower_bound <- Q1 - 1.5 * IQR
  upper_bound <- Q3 + 1.5 * IQR
  outliers <- column[column < lower_bound | column > upper_bound]
  return(outliers)
}
outliers_age <- find_outliers_iqr(newData2$person_age)
outliers_income <- find_outliers_iqr(newData2$person_income)
outliers_emp_exp <- find_outliers_iqr(newData2$person_emp_exp)
outliers_loan_int_rate <- find_outliers_iqr(newData2$loan_int_rate)
outliers_loan_percent_income <-
find_outliers_iqr(newData2$loan_percent_income)
outliers_credit_score <- find_outliers_iqr(newData2$credit_score)


outliers_age
outliers_income
outliers_emp_exp
outliers_loan_int_rate
outliers_loan_percent_income
outliers_credit_score
```

Output:

```
> find_outliers_iqr <- function(column) {
+   Q1 <- quantile(column, 0.25, na.rm = TRUE)
+   Q3 <- quantile(column, 0.75, na.rm = TRUE)
+   IQR <- Q3 - Q1
+   lower_bound <- Q1 - 1.5 * IQR
+   upper_bound <- Q3 + 1.5 * IQR
+   outliers <- column[column < lower_bound | column > upper_bound]
+   return(outliers)
+ }
> outliers_age <- find_outliers_iqr(newData2$person_age)
> outliers_income <- find_outliers_iqr(newData2$person_income)
> outliers_emp_exp <- find_outliers_iqr(newData2$person_emp_exp)
> outliers_loan_int_rate <- find_outliers_iqr(newData2$loan_int_rate)
> outliers_loan_percent_income <- find_outliers_iqr(newData2$loan_percent_income)
> outliers_credit_score <- find_outliers_iqr(newData2$credit_score)
>
> outliers_age
[1] 230 350 144 144
> outliers_income
[1] 3138998
> outliers_emp_exp
[1] 125   8 121
> outliers_loan_int_rate
[1] 20  6  5 20  6  6
> outliers_loan_percent_income
[1] 1 1
> outliers_credit_score
[1] 789 484 807
>
```

Description:

We created a custom function to calculate Interquartile Range (IQR) for those attributes has outlier and at the end print those outlier.

Remove Outliers:

Code:

```
cleanData <- newData1

cleanData <- cleanData[!(cleanData$person_age %in% outliers_age), ]

cleanData <- cleanData[!(cleanData$person_income %in% outliers_income), ]

cleanData <- cleanData[!(cleanData$person_emp_exp %in% outliers_emp_exp), ]

cleanData <- cleanData[!(cleanData$loan_int_rate %in%
outliers_loan_int_rate), ]
```

```
cleanData <- cleanData[!(cleanData$loan_percent_income %in%
outliers_loan_percent_income), ]

cleanData <- cleanData[!(cleanData$credit_score %in% outliers_credit_score),
]

cleanData
```

Output:

```
> cleanData <- newData1
> cleanData <- cleanData[!(cleanData$person_age %in% outliers_age), ]
> cleanData <- cleanData[!(cleanData$person_income %in% outliers_income), ]
> cleanData <- cleanData[!(cleanData$person_emp_exp %in% outliers_emp_exp), ]
> cleanData <- cleanData[!(cleanData$loan_int_rate %in% outliers_loan_int_rate), ]
> cleanData <- cleanData[!(cleanData$loan_percent_income %in% outliers_loan_percent_income), ]
> cleanData <- cleanData[!(cleanData$credit_score %in% outliers_credit_score), ]
> cleanData
# A tibble: 178 × 14
   person_age person_gender person_education person_income person_emp_exp person_home_ownership loan_amnt loan_intent loan_int_rate
        <dbl> <fct>         <fct>                  <dbl>          <dbl> <fct>                     <dbl> <fct>              <dbl>
 1         21 0             3                      71948              0 1                         35000 1                   16.0
 2         25 0             1                      12438              3 3                          5500 3                   12.9
 3         23 0             2                      79753              0 1                         35000 3                   15.2
 4         24 1             3                      66135              1 1                         35000 3                   14.3
 5         21 0             1                      12739              0 2                          1600 4                   14.7
 6         22 0             1                     102985              0 1                         35000 4                   10.4
 7         21 0             4                      13113              0 2                          4500 5                    8.63
 8         23 1             2                     114860              3 1                         35000 4                    7.9
 9         23 1             2                     136628              0 1                         35000 6                   18.2
10         24 0             3                      14283              1 3                          1750 2                   11.0
# i 168 more rows
# i 5 more variables: loan_percent_income <dbl>, cb_person_cred_hist_length <dbl>, credit_score <dbl>,
#   previous_loan_defaults_on_file <fct>, loan_status <dbl>
# i Use `print(n = ...)` to see more rows
>
```

Description:

We remove outliers from the newData1(Discard Instances) dataset and store the cleaned data in cleanData. It iteratively filters out rows where the values in specific columns match the outliers identified earlier. After this process, the cleanData dataset contains only rows without the identified outliers.

Normalization:

Code:

```
NorData<- cleanData

min_person_income <- min(NorData$person_income, na.rm = TRUE)

max_person_income <- max(NorData$person_income, na.rm = TRUE)

NorData$person_income <- (NorData$person_income - min_person_income) /
(max_person_income - min_person_income)
```

```
NorData
```

Output:

```
> NorData<- cleanData
> min_person_income <- min(NorData$person_income, na.rm = TRUE)
> max_person_income <- max(NorData$person_income, na.rm = TRUE)
> NorData$person_income <- (NorData$person_income - min_person_income) / (max_person_income - min_person_income)
> NorData
# A tibble: 178 × 14
   person_age person_gender person_education person_income person_emp_exp person_home_ownership loan_amnt loan_intent loan_int_rate
        <dbl> <fct>         <fct>                    <dbl>          <dbl> <fct>                     <dbl> <fct>               <dbl>
 1         21 0             3                        0.170              0 1                        35000 1                    16.0
 2         25 0             1                        0                  3 3                         5500 3                    12.9
 3         23 0             2                        0.193              0 1                        35000 3                    15.2
 4         24 1             3                        0.154              1 1                        35000 3                    14.3
 5         21 0             1                        0.000862           0 2                         1600 4                    14.7
 6         22 0             1                        0.259              0 1                        35000 4                    10.4
 7         21 0             4                        0.00193            0 2                         4500 5                     8.63
 8         23 1             2                        0.293              3 1                        35000 4                     7.9
 9         23 1             2                        0.356              0 1                        35000 6                    18.2
10         24 0             3                        0.00528            1 3                         1750 2                    11.0
# ℹ 168 more rows
# ℹ 5 more variables: loan_percent_income <dbl>, cb_person_cred_hist_length <dbl>, credit_score <dbl>,
#   previous_loan_defaults_on_file <fct>, loan_status <dbl>
# ℹ Use `print(n = ...)` to see more rows
>
```

Description:

After removing outliers, we perform Min-Max normalization on the person_income column in the dataset.

Convert Numerical Attributes to Categorical:

Code:

```
OutputData <- NorData

OutputData$person_gender <- factor(OutputData$person_gender, levels = c(0, 1), labels = c("female", "male"))

OutputData$person_education <- factor(OutputData$person_education, levels = c(1, 2, 3, 4, 5), labels = c("High School", "Bachelor", "Master", "Associate", "Doctorate"))

OutputData$person_home_ownership <- factor(OutputData$person_home_ownership, levels = c(1, 2, 3, 4), labels = c("RENT", "OWN", "MORTGAGE", "OTHER"))

OutputData$loan_intent <- factor(OutputData$loan_intent, levels = c(1, 2, 3, 4, 5, 6), labels = c("PERSONAL", "EDUCATION", "MEDICAL", "VENTURE", "HOMEIMPROVEMENT", "DEBTCONSOLIDATION"))
```

```
OutputData$previous_loan_defaults_on_file <-
factor(OutputData$previous_loan_defaults_on_file, levels = c(0, 1), labels =
c("No", "Yes"))

OutputData
```

Output:

```
> OutputData <- NorData
> OutputData$person_gender <- factor(OutputData$person_gender, levels = c(0, 1), labels = c("female", "male"))
> OutputData$person_education <- factor(OutputData$person_education, levels = c(1, 2, 3, 4, 5), labels = c("High School", "Bachelor", "Mast
er", "Associate", "Doctorate"))
> OutputData$person_home_ownership <- factor(OutputData$person_home_ownership, levels = c(1, 2, 3, 4), labels = c("RENT", "OWN", "MORTGAG
E", "OTHER"))
> OutputData$loan_intent <- factor(OutputData$loan_intent, levels = c(1, 2, 3, 4, 5, 6), labels = c("PERSONAL", "EDUCATION", "MEDICAL", "VE
NTURE", "HOMEIMPROVEMENT", "DEBTCONSOLIDATION"))
> OutputData$previous_loan_defaults_on_file <- factor(OutputData$previous_loan_defaults_on_file, levels = c(0, 1), labels = c("No", "Yes"))
> OutputData
# A tibble: 178 × 14
   person_age person_gender person_education person_income person_emp_exp person_home_ownership loan_amnt loan_intent     loan_int_rate
       <dbl> <fct>         <fct>                    <dbl>          <dbl> <fct>                     <dbl> <fct>                 <dbl>
 1         21 female        Master                0.170                0 RENT                      35000 PERSONAL               16.0
 2         25 female        High School           0                    3 MORTGAGE                   5500 MEDICAL                12.9
 3         23 female        Bachelor              0.193                0 RENT                      35000 MEDICAL                15.2
 4         24 male          Master                0.154                1 RENT                      35000 MEDICAL                14.3
 5         21 female        High School           0.000862             0 OWN                        1600 VENTURE                14.7
 6         22 female        High School           0.259                0 RENT                      35000 VENTURE                10.4
 7         21 female        Associate             0.00193              0 OWN                        4500 HOMEIMPROVEMENT         8.63
 8         23 male          Bachelor              0.293                3 RENT                      35000 VENTURE                 7.9
 9         23 male          Bachelor              0.356                0 RENT                      35000 DEBTCONSOLIDAT…        18.2
10         24 female        Master                0.00528              1 MORTGAGE                   1750 EDUCATION              11.0
# ℹ 168 more rows
# ℹ 5 more variables: loan_percent_income <dbl>, cb_person_cred_hist_length <dbl>, credit_score <dbl>,
#   previous_loan_defaults_on_file <fct>, loan_status <dbl>
# ℹ Use `print(n = ...)` to see more rows
>
```

Description:

After normalization person_gender attribute, person_education, person_home_ownership, loan_intent and previous_loan_defaults_on_file columns converted back to categorical attributes shown using factor() function.

Measures of Central Tendency (Mean, Mode, Median):

Code:

```
mean(OutputData$person_income)
mean(OutputData$loan_amnt)

median(OutputData$person_income)
median(OutputData$loan_amnt)
```

```
Mode <- function(m) {
  uniq_vals <- unique(m)
  uniq_vals[which.max(tabulate(match(m, uniq_vals)))]
}
Mode(m = OutputData$person_income)
Mode(m = OutputData$loan_amnt)
```

Output:

```
> mean(OutputData$person_income)
[1] 0.3534925
> mean(OutputData$loan_amnt)
[1] 20328.93
>
> median(OutputData$person_income)
[1] 0.2084707
> median(OutputData$loan_amnt)
[1] 25000
>
> Mode <- function(m) {
+   uniq_vals <- unique(m)
+   uniq_vals[which.max(tabulate(match(m, uniq_vals)))]
+ }
> Mode(m = OutputData$person_income)
[1] 0.1704625
> Mode(m = OutputData$loan_amnt)
[1] 25000
>
```

Description:

We perform mean, median and mode on person_income and loan_amnt attributes. For mean and median we use built in function and for mode we use custom function.

Measure of Spread (Range, Variance, SD):

Code:

```
rangeValue <- function(r){
  max(r) - min(r)
}
rangeValue(r = OutputData$person_income)
```

```
rangeValue(r = OutputData$loan_amnt)


var(OutputData$person_income)
var(OutputData$loan_amnt)


sd(OutputData$person_income)
sd(OutputData$loan_amnt)
```

Output:

```
> rangeValue <- function(r){
+   max(r) - min(r)
+ }
> rangeValue(r = OutputData$person_income)
[1] 1
> rangeValue(r = OutputData$loan_amnt)
[1] 33600
> var(OutputData$person_income)
[1] 0.09417324
> var(OutputData$loan_amnt)
[1] 107441482
> sd(OutputData$person_income)
[1] 0.3068766
> sd(OutputData$loan_amnt)
[1] 10365.4
>
```

Description:

We perform range, variance and standard deviation on person_income and loan_amnt attributes. For variance and standard deviation, we use built in function and for range we use custom function.

Handle Imbalance data:

Code:

```
library(ROSE)
library(dplyr)
class_distribution <- table(OutputData$loan_status)
class_distribution
if (class_distribution[1] > class_distribution[2]) {
```

```
  majority <- filter(OutputData, loan_status == 0)
  minority <- filter(OutputData, loan_status == 1)
} else {
  majority <- filter(OutputData, loan_status == 1)
  minority <- filter(OutputData, loan_status == 0)
}
set.seed(123)
oversampled_minority <- minority %>% sample_n(nrow(majority), replace = TRUE)
oversampled_data <- bind_rows(majority, oversampled_minority)
table(oversampled_data$loan_status)
```

Output:

```
> if (class_distribution[1] > class_distribution[2]) {
+   majority <- filter(OutputData, loan_status == 0)
+   minority <- filter(OutputData, loan_status == 1)
+ } else {
+   majority <- filter(OutputData, loan_status == 1)
+   minority <- filter(OutputData, loan_status == 0)
+ }
>
> set.seed(123)
> oversampled_minority <- minority %>% sample_n(nrow(majority), replace = TRUE)
> oversampled_data <- bind_rows(majority, oversampled_minority)
>
> table(oversampled_data$loan_status)

  0   1
110 110
>
```

Description:

First, we checked if both the minority and majority classes exist. We defined '0' as the majority class and '1' as the minority class. Then, we oversampled the minority class '1' to balance it with the majority class '0'.