BIN381_Project_Group-F_Preprocessing

Group F

2024-10-25

Data Cleaning

```
# Read 'CustData2.csv' file into data frame 'customers'
customers <- read.csv("CustData2.csv")

# Display structure of the data frame
str(customers)
```

```
## 'data.frame':    191323 obs. of  24 variables:
##  $ Column1                              : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ Last.Name                            : chr  "ALBERT" "ARGUELLO" "TUCKER" "DELL" ...
##  $ First.Name                           : chr  "JESSICA" "ADRIAN" "KEVIN" "JAMES" ...
##  $ Middle.Initial                       : chr  "M" "A" "K" "A" ...
##  $ Title                                : chr  "CORRECTIONAL OFFICER" "POLICE OFFICER" "CORRECTIONAL
##  $ Department.Name                      : chr  "CORRECTIONS & REHABILITATION" "POLICE" "CORRECTIONS &
##  $ Annual.Salary                        : num  54620 65250 62394 37735 64386 ...
##  $ Gross.Pay.Last.Paycheck              : num  2502 3468 4514 1562 6666 ...
##  $ Gross.Year.To.Date                   : num  48025 57932 49968 35470 132851 ...
##  $ Gross.Year.To.Date...FRS.Contribution: num  46617 56223 48501 34433 128949 ...
##  $ year_of_birth                        : int  1976 1964 1942 1977 1949 1950 1946 1978 1949 1951 ...
##  $ marital_status                       : chr  "married" "" "single" "married" ...
##  $ street_address                       : chr  "27 North Sagadahoc Boulevard" "37 West Geneva Street
##  $ postal_code                          : int  60332 55406 34077 72996 67644 83786 52773 37400 71349
##  $ city                                 : chr  "Ede" "Hoofddorp" "Schimmert" "Scheveningen" ...
##  $ State                                : chr  "Gelderland" "Noord" "Limburg" "Zuid" ...
##  $ Province                             : chr  "" "Holland" "" "Holland" ...
##  $ Country_id                           : int  52770 52770 52770 52770 52775 52782 52775 52782 52770
##  $ phone_number                         : chr  "519-236-6123" "327-194-5008" "288-613-9676" "222-269-
##  $ email                                : chr  "Ruddy@company.com" "Ruddy@company.com" "Ruddy@company
##  $ Education                            : chr  "Masters" "Masters" "Masters" "Masters" ...
##  $ Occupation                           : chr  "Prof." "Prof." "Prof." "Prof." ...
##  $ household_size                       : int  2 2 2 2 2 2 2 2 2 2 ...
##  $ yrs_residence                        : int  4 4 4 4 4 4 4 4 4 4 ...
```

Create new attributes from existing ones (Feature Engineering)

```
# Import 'lubridate' package to work with Date types
library(lubridate)
```

```
##
## Attaching package: 'lubridate'

## The following objects are masked from 'package:base':
##
##     date, intersect, setdiff, union
```

```
# Create a new column/attribute that calculates the customers age based
# on 'year of birth'
customers$Age <- as.integer(year(today()) - customers$year_of_birth)

# Create the target attribute that states whether the person is eligible or not
customers$Eligible <- ifelse(customers$Annual.Salary > 50000, 1, 0)

# Display structure of the data frame
str(customers)
```

```
## 'data.frame':    191323 obs. of  26 variables:
##  $ Column1                              : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ Last.Name                            : chr  "ALBERT" "ARGUELLO" "TUCKER" "DELL" ...
##  $ First.Name                           : chr  "JESSICA" "ADRIAN" "KEVIN" "JAMES" ...
##  $ Middle.Initial                       : chr  "M" "A" "K" "A" ...
##  $ Title                                : chr  "CORRECTIONAL OFFICER" "POLICE OFFICER" "CORRECTIONAL
##  $ Department.Name                      : chr  "CORRECTIONS & REHABILITATION" "POLICE" "CORRECTIONS &
##  $ Annual.Salary                        : num  54620 65250 62394 37735 64386 ...
##  $ Gross.Pay.Last.Paycheck              : num  2502 3468 4514 1562 6666 ...
##  $ Gross.Year.To.Date                   : num  48025 57932 49968 35470 132851 ...
##  $ Gross.Year.To.Date...FRS.Contribution: num  46617 56223 48501 34433 128949 ...
##  $ year_of_birth                        : int  1976 1964 1942 1977 1949 1950 1946 1978 1949 1951 ...
##  $ marital_status                       : chr  "married" "" "single" "married" ...
##  $ street_address                       : chr  "27 North Sagadahoc Boulevard" "37 West Geneva Street
##  $ postal_code                          : int  60332 55406 34077 72996 67644 83786 52773 37400 71349
##  $ city                                 : chr  "Ede" "Hoofddorp" "Schimmert" "Scheveningen" ...
##  $ State                                : chr  "Gelderland" "Noord" "Limburg" "Zuid" ...
##  $ Province                             : chr  "" "Holland" "" "Holland" ...
##  $ Country_id                           : int  52770 52770 52770 52770 52775 52782 52775 52782 52770
##  $ phone_number                         : chr  "519-236-6123" "327-194-5008" "288-613-9676" "222-269-
##  $ email                                : chr  "Ruddy@company.com" "Ruddy@company.com" "Ruddy@company
##  $ Education                            : chr  "Masters" "Masters" "Masters" "Masters" ...
##  $ Occupation                           : chr  "Prof." "Prof." "Prof." "Prof." ...
##  $ household_size                       : int  2 2 2 2 2 2 2 2 2 2 ...
##  $ yrs_residence                        : int  4 4 4 4 4 4 4 4 4 4 ...
##  $ Age                                  : int  48 60 82 47 75 74 78 46 75 73 ...
##  $ Eligible                             : num  1 1 1 0 1 1 1 0 1 0 ...
```

Calculate % eligible of baseline

```
numEligibleOriginal <- sum(customers$Eligible == 1, na.rm = TRUE)
totalCustomersOriginal <- length(customers$Eligible)
eligiblePercantageOriginal <- (numEligibleOriginal / totalCustomersOriginal) * 100
cat("Percentage of Eligible Customers in baseline model: ", round(eligiblePercantageOriginal, 2))
```

```
## Percentage of Eligible Customers in baseline model:  64.43
```

Remove irrelevant attributes

```
# Create vector with all columns/attributes that need to be kept
keepColumns <- c("Title", "Department.Name", "Annual.Salary",
                 "Gross.Pay.Last.Paycheck", "Gross.Year.To.Date",
                 "Gross.Year.To.Date...FRS.Contribution",
                 "Age", "marital_status", "Country_id", "Education",
                 "Occupation", "household_size", "yrs_residence", "Eligible")

# Remove irrelevant columns/attributes by keeping relevant ones
customers <- customers[keepColumns]

# Display structure of the data frame
str(customers)
```

```
## 'data.frame':    191323 obs. of  14 variables:
##  $ Title                                : chr  "CORRECTIONAL OFFICER" "POLICE OFFICER" "CORRECTIONAL
##  $ Department.Name                      : chr  "CORRECTIONS & REHABILITATION" "POLICE" "CORRECTIONS &
##  $ Annual.Salary                        : num  54620 65250 62394 37735 64386 ...
##  $ Gross.Pay.Last.Paycheck              : num  2502 3468 4514 1562 6666 ...
##  $ Gross.Year.To.Date                   : num  48025 57932 49968 35470 132851 ...
##  $ Gross.Year.To.Date...FRS.Contribution: num  46617 56223 48501 34433 128949 ...
##  $ Age                                  : int  48 60 82 47 75 74 78 46 75 73 ...
##  $ marital_status                       : chr  "married" "" "single" "married" ...
##  $ Country_id                           : int  52770 52770 52770 52770 52775 52782 52775 52782 52770
##  $ Education                            : chr  "Masters" "Masters" "Masters" "Masters" ...
##  $ Occupation                           : chr  "Prof." "Prof." "Prof." "Prof." ...
##  $ household_size                       : int  2 2 2 2 2 2 2 2 2 2 ...
##  $ yrs_residence                        : int  4 4 4 4 4 4 4 4 4 4 ...
##  $ Eligible                             : num  1 1 1 0 1 1 1 0 1 0 ...
```

Cleaning "marital_status"

```
# Display all of the unique values contained in the
# 'marital_status' column/attribute
unique(customers$marital_status)
```

```
##  [1] "married"  ""         "single"   "divorced" "widow"    "Divorc."
##  [7] "NeverM"   "Married"  "Separ."   "Mabsent"  "Widowed"  "Mar-AF"
```

```
# Count the unique values contained in the 'marital_status' column/attribute
length(unique(customers$marital_status))
```

```
## [1] 12
```

```r
# Replace incorrect values for "marital_status"
for (i in 1:nrow(customers)) {
  if (customers$marital_status[i] == "Married") {
    customers$marital_status[i] <- "married"
  } else if (customers$marital_status[i] == "Mar-AF") {
    customers$marital_status[i] <- "married"
  } else if (customers$marital_status[i] == "NeverM") {
    customers$marital_status[i] <- "single"
  } else if (customers$marital_status[i] == "Mabsent") {
    customers$marital_status[i] <- "single"
  } else if (customers$marital_status[i] == "Divorc.") {
    customers$marital_status[i] <- "divorced"
  } else if (customers$marital_status[i] == "Separ.") {
    customers$marital_status[i] <- "divorced"
  } else if (customers$marital_status[i] == "widow") {
    customers$marital_status[i] <- "widowed"
  } else if (customers$marital_status[i] == "Widowed") {
    customers$marital_status[i] <- "widowed"
  }
}

# Check to see if "marital_status" was cleaned successfully
unique(customers$marital_status)
```

```
## [1] "married"  ""         "single"   "divorced" "widowed"
```

```r
length(unique(customers$marital_status))
```

```
## [1] 5
```

Populating “marital_status”

```r
# Count the number of empty cells
sum(customers$marital_status=="")
```

```
## [1] 60795
```

```r
# Function to calculate mode
get_mode <- function(v) {
  uniq_vals <- unique(v)
  uniq_vals[which.max(tabulate(match(v, uniq_vals)))]
}

# Get mode value from function
mode_value <- get_mode(
  customers$marital_status[!is.na(customers$marital_status) &
  customers$marital_status != ""])

# Fill missing or empty values in "marital_status" column with mode
customers$marital_status[is.na(customers$marital_status) |
```

```
                          customers$marital_status == ""] <- mode_value

# Check if "marital_status" is filled
sum(customers$marital_status=="")
```

```
## [1] 0
```

Missing Values

```
sum(customers$Title=="")
```

```
## [1] 6
```

```
sum(customers$Department.Name=="")
```

```
## [1] 6
```

```
sum(is.na(customers$Annual.Salary))
```

```
## [1] 6
```

```
sum(is.na(customers$Gross.Pay.Last.Paycheck))
```

```
## [1] 6
```

```
sum(is.na(customers$Gross.Year.To.Date))
```

```
## [1] 6
```

```
sum(is.na(customers$Gross.Year.To.Date...FRS.Contribution))
```

```
## [1] 6
```

```
sum(is.na(customers$Age))
```

```
## [1] 0
```

```
sum(customers$marital_status=="")
```

```
## [1] 0
```

```
sum(is.na(customers$Country_id))
```

```
## [1] 0
```

```r
sum(customers$Education=="")
```

```
## [1] 0
```

```r
sum(customers$Occupation=="")
```

```
## [1] 0
```

```r
sum(is.na(customers$household_size))
```

```
## [1] 0
```

```r
sum(is.na(customers$yrs_residence))
```

```
## [1] 0
```

```r
# Remove empty cells for all columns/attributes
customers <- customers[!(is.na(customers$Title) | customers$Title == "" |
                       is.na(customers$Department.Name)  |
                       customers$Department.Name == ""  |
                       is.na(customers$Annual.Salary)  |
                       customers$Annual.Salary == ""  |
                       is.na(customers$Gross.Pay.Last.Paycheck)  |
                       customers$Gross.Pay.Last.Paycheck == ""  |
                       is.na(customers$Gross.Year.To.Date)  |
                       customers$Gross.Year.To.Date == ""  |
                       is.na(customers$Gross.Year.To.Date...FRS.Contribution)  |
                       customers$Gross.Year.To.Date...FRS.Contribution == ""), ]

# Check if there are empty cells left
sum(customers$Title=="")
```

```
## [1] 0
```

```r
sum(customers$Department.Name=="")
```

```
## [1] 0
```

```r
sum(is.na(customers$Annual.Salary))
```

```
## [1] 0
```

```r
sum(is.na(customers$Gross.Pay.Last.Paycheck))
```

```
## [1] 0
```

```r
sum(is.na(customers$Gross.Year.To.Date))
```

```
## [1] 0
```

```r
sum(is.na(customers$Gross.Year.To.Date...FRS.Contribution))
```

```
## [1] 0
```

```r
sum(is.na(customers$Age))
```

```
## [1] 0
```

```r
sum(is.na(customers$Country_id))
```

```
## [1] 0
```

```r
sum(customers$Education=="")
```

```
## [1] 0
```

```r
sum(customers$Occupation=="")
```

```
## [1] 0
```

```r
sum(is.na(customers$household_size))
```

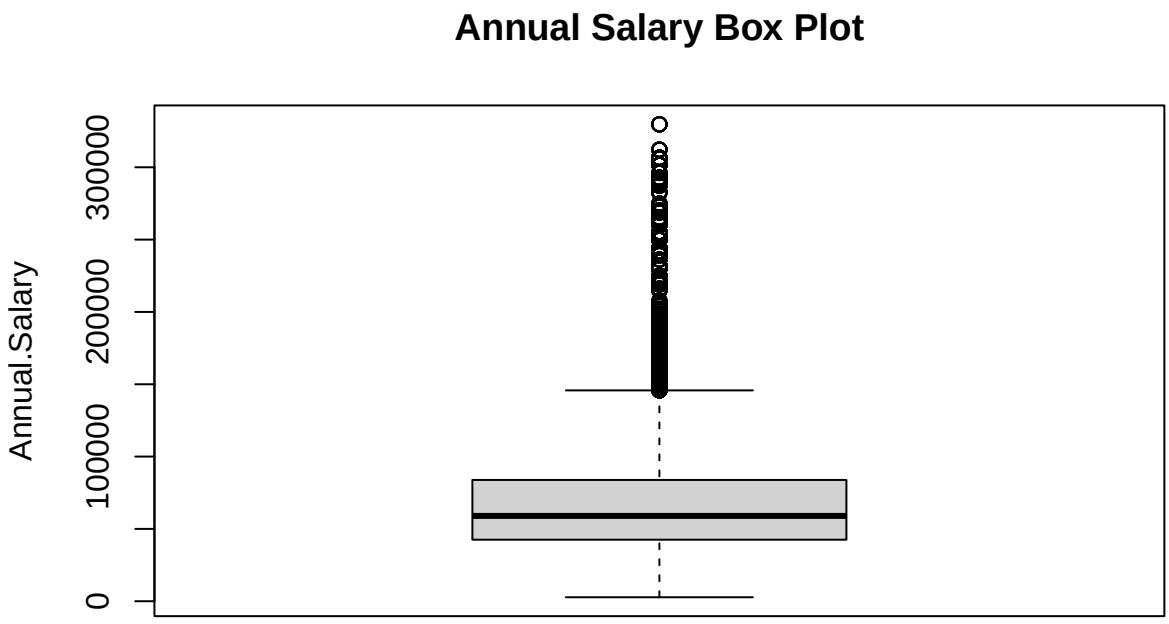
```
## [1] 0
```

```r
sum(is.na(customers$yrs_residence))
```

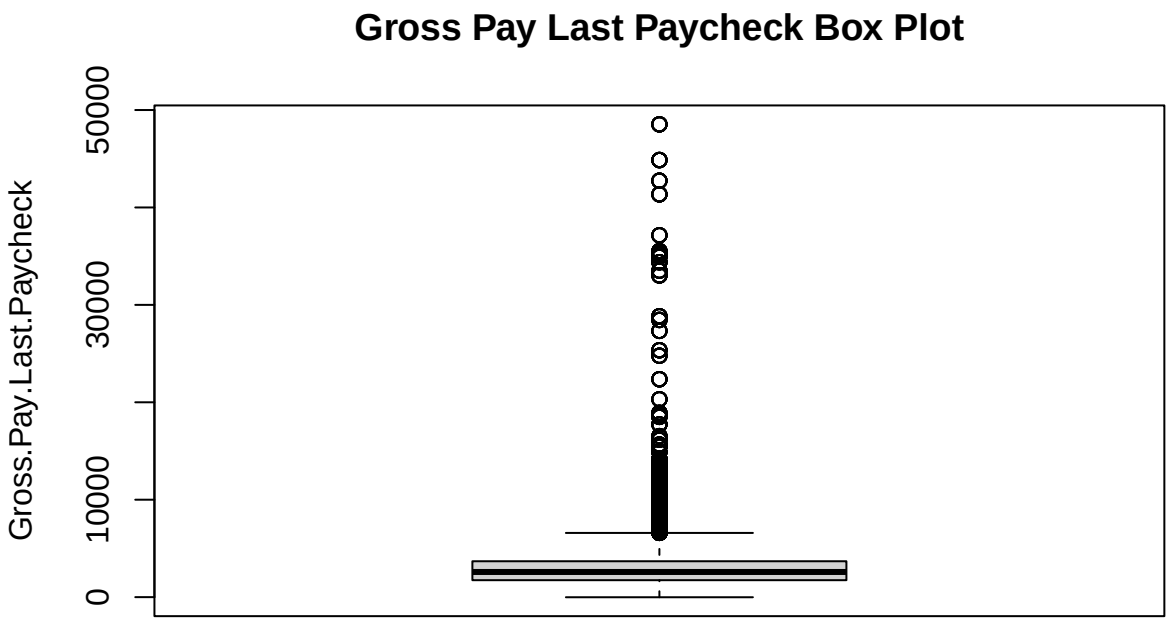
```
## [1] 0
```

Outlier treatment

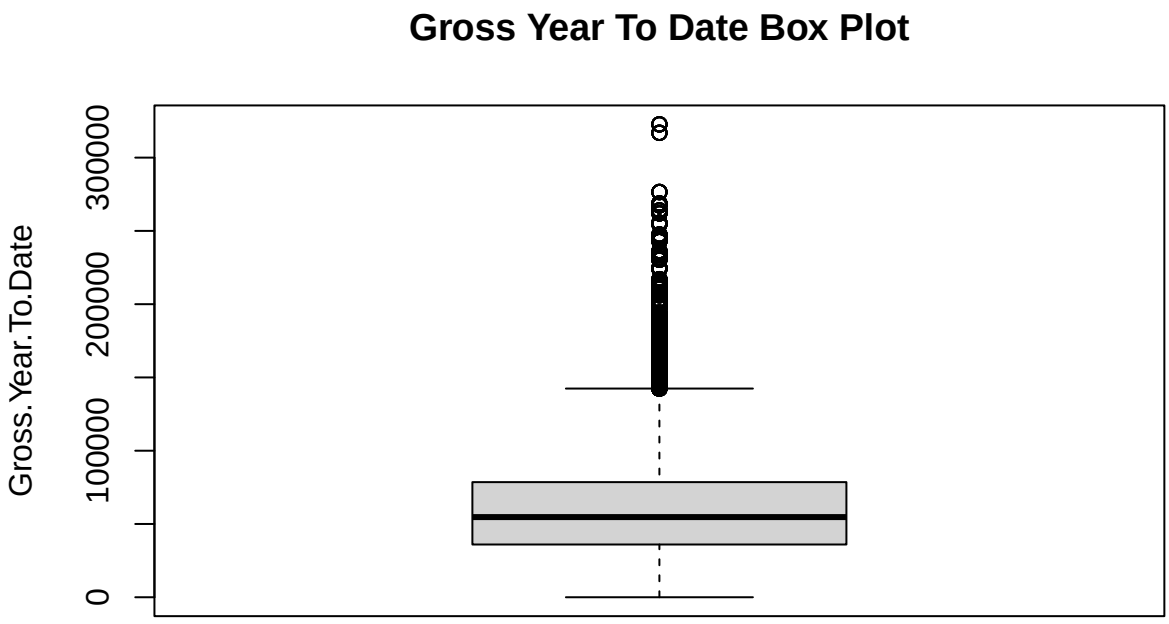
```r
# Display outliers
## Display "Annual.Salary" box plot
boxplot(customers$Annual.Salary,
        main = "Annual Salary Box Plot",
        ylab = "Annual.Salary")
```



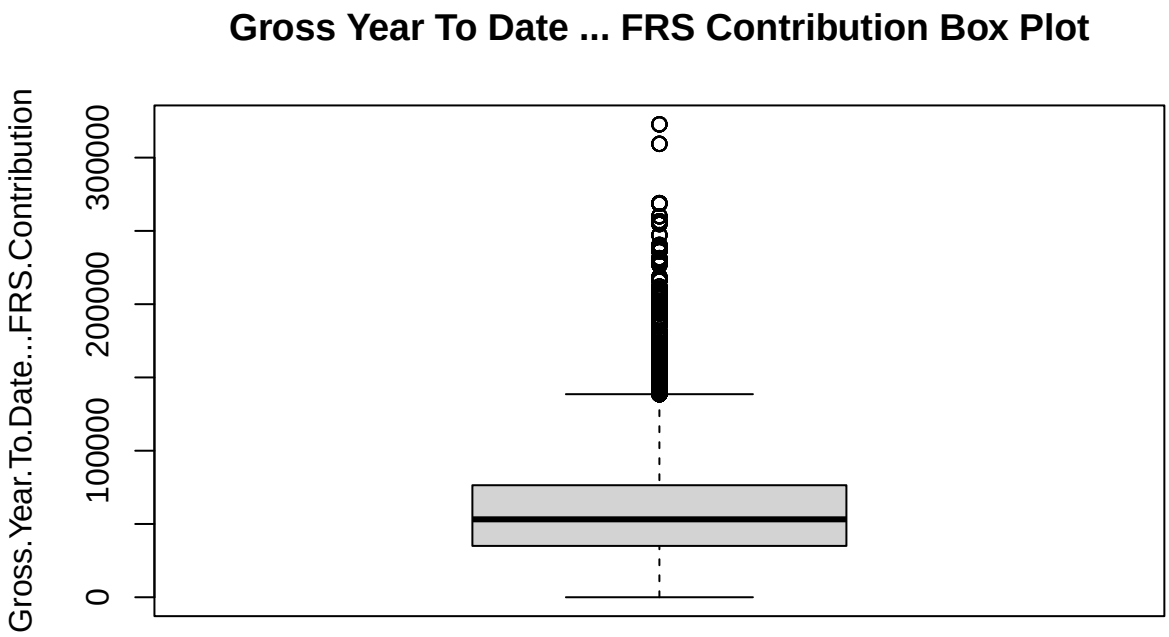

```
## Display "Gross.Pay.Last.Paycheck" box plot
boxplot(customers$Gross.Pay.Last.Paycheck,
        main = "Gross Pay Last Paycheck Box Plot",
        ylab = "Gross.Pay.Last.Paycheck")
```



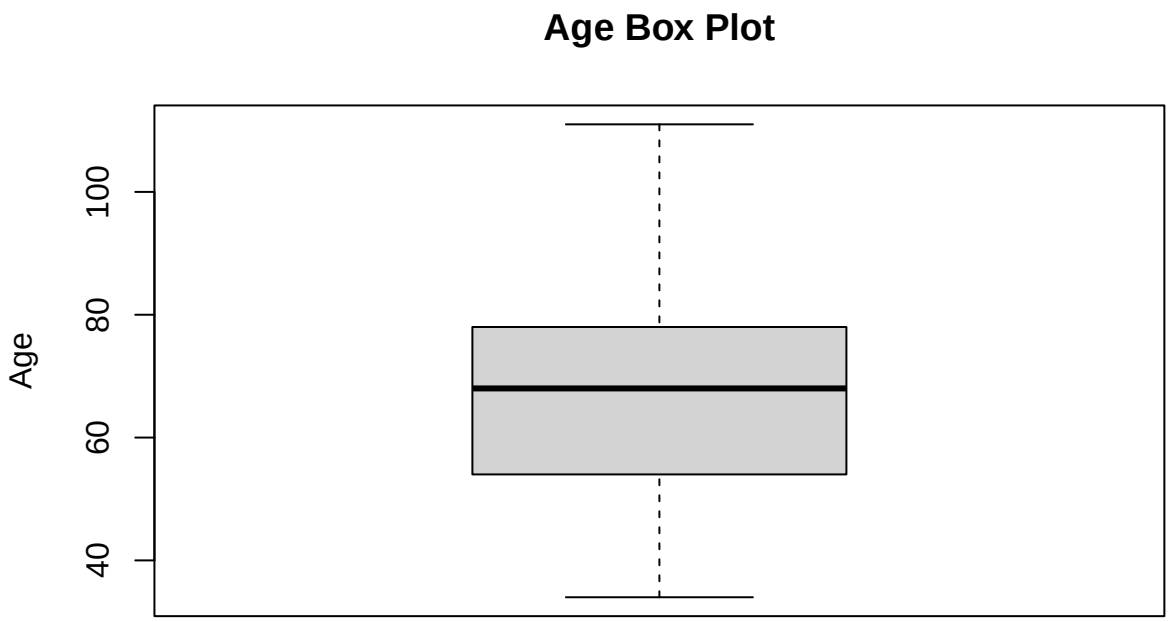

```
## Display "Gross.Year.To.Date" box plot
boxplot(customers$Gross.Year.To.Date,
        main = "Gross Year To Date Box Plot",
        ylab = "Gross.Year.To.Date")
```




```
## Display "Gross.Year.To.Date...FRS.Contribution" box plot
boxplot(customers$Gross.Year.To.Date...FRS.Contribution,
        main = "Gross Year To Date ... FRS Contribution Box Plot",
        ylab = "Gross.Year.To.Date...FRS.Contribution")
```



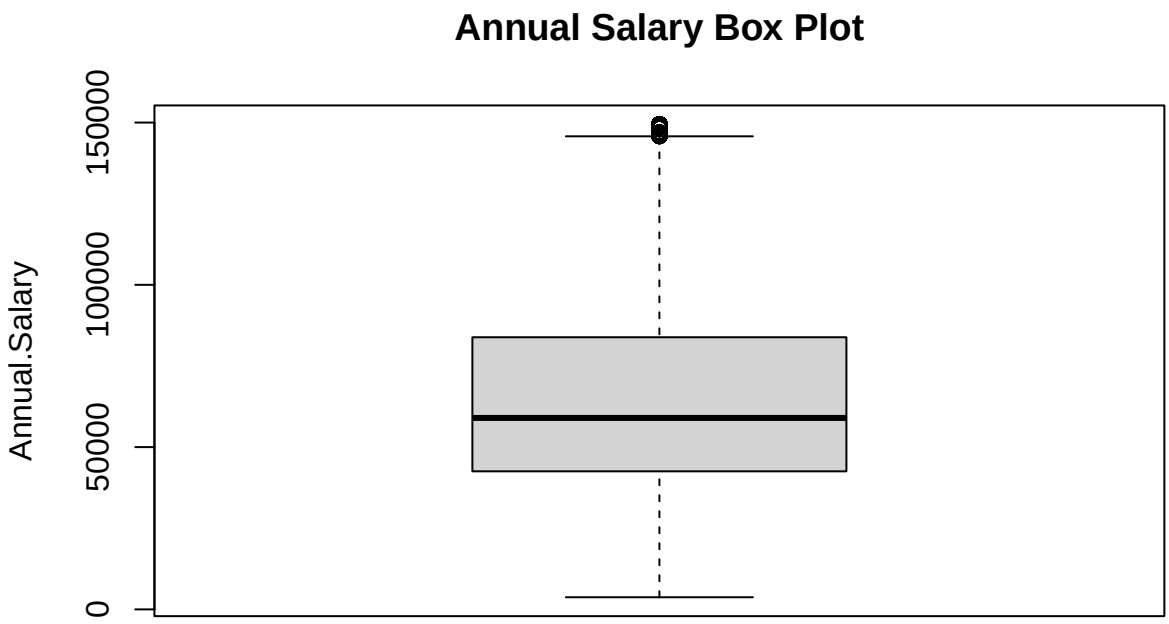

```
boxplot(customers$Age,
        main = "Age Box Plot",
        ylab = "Age")
```



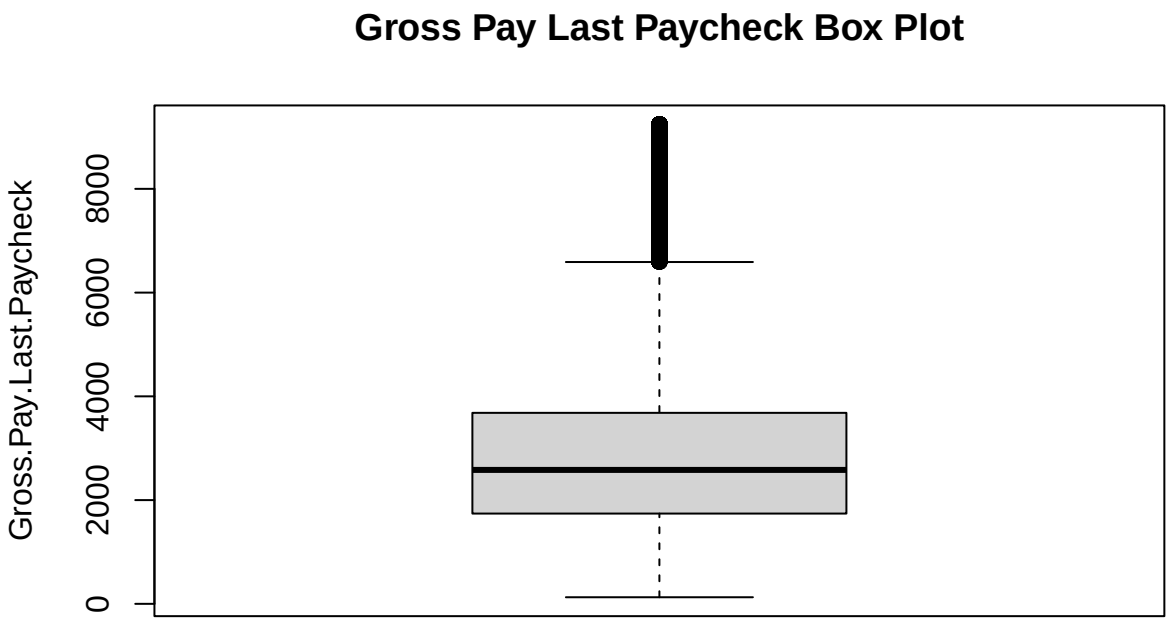

```r
# Capping outliers using the 1st and 99th percentiles
cap_outliers <- function(column) {
  lower_cap <- quantile(column, 0.01)
  upper_cap <- quantile(column, 0.99)
  column[column < lower_cap] <- lower_cap
  column[column > upper_cap] <- upper_cap
  return(column)
}

# Apply capping to the numeric columns
customers$Annual.Salary <- cap_outliers(customers$Annual.Salary)
customers$Gross.Pay.Last.Paycheck <- cap_outliers(
                  customers$Gross.Pay.Last.Paycheck)
customers$Gross.Year.To.Date <- cap_outliers(customers$Gross.Year.To.Date)
customers$Gross.Year.To.Date...FRS.Contribution <- cap_outliers(
                  customers$Gross.Year.To.Date...FRS.Contribution)
```

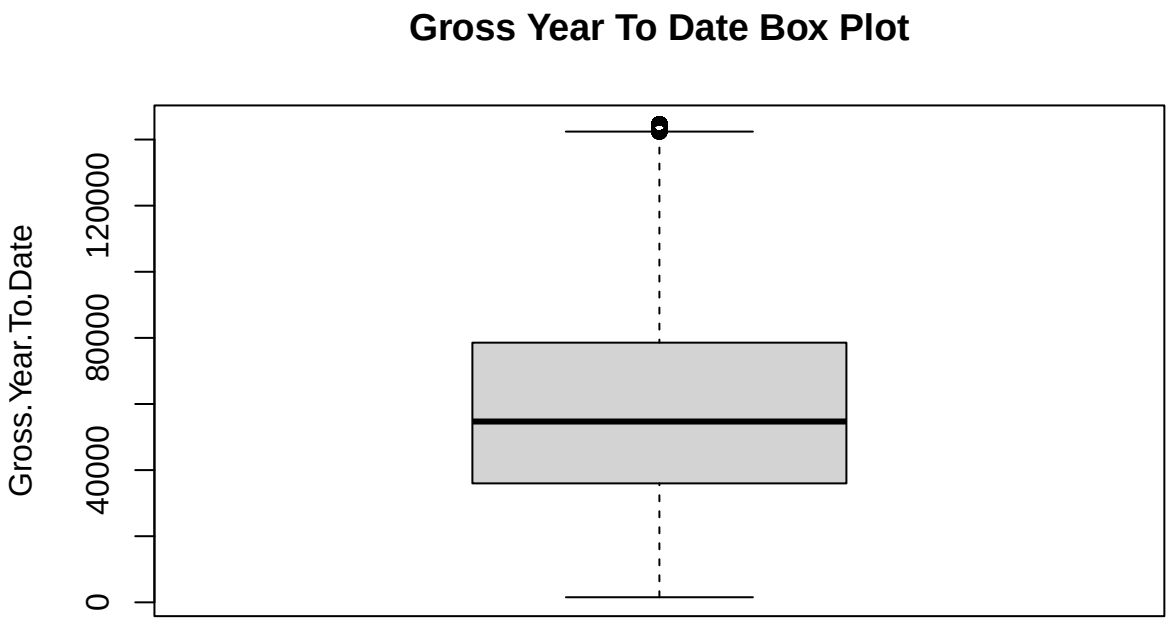
```r
# Check if outliers are fixed
## Display "Annual.Salary" box plot
boxplot(customers$Annual.Salary,
        main = "Annual Salary Box Plot",
        ylab = "Annual.Salary")
```



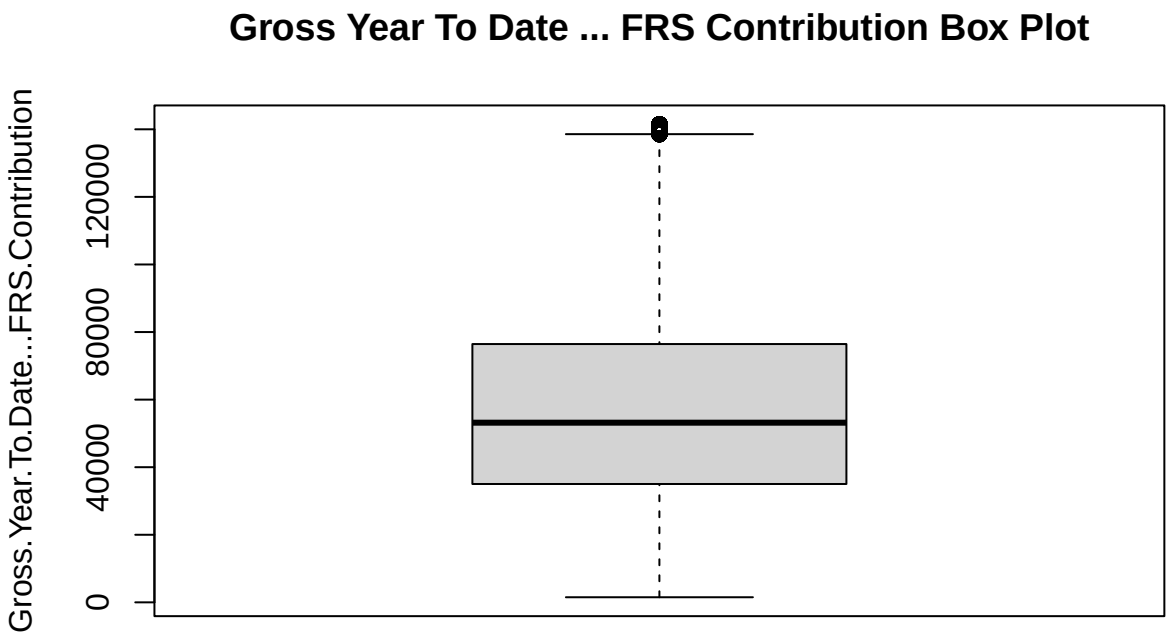

```
## Display "Gross.Pay.Last.Paycheck" box plot
boxplot(customers$Gross.Pay.Last.Paycheck,
        main = "Gross Pay Last Paycheck Box Plot",
        ylab = "Gross.Pay.Last.Paycheck")
```



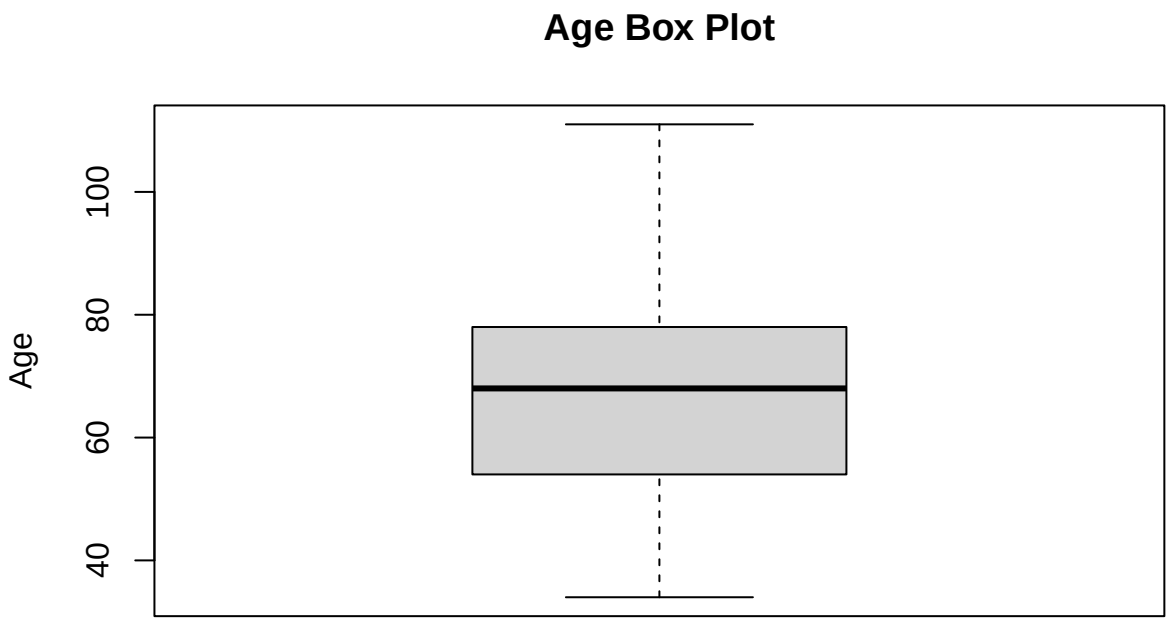

```
## Display "Gross.Year.To.Date" box plot
boxplot(customers$Gross.Year.To.Date,
        main = "Gross Year To Date Box Plot",
        ylab = "Gross.Year.To.Date")
```




```
## Display "Gross.Year.To.Date...FRS.Contribution" box plot
boxplot(customers$Gross.Year.To.Date...FRS.Contribution,
        main = "Gross Year To Date ... FRS Contribution Box Plot",
        ylab = "Gross.Year.To.Date...FRS.Contribution")
```




```
boxplot(customers$Age,
        main = "Age Box Plot",
        ylab = "Age")
```



Data Preprocessing

```
#Assign customers to custData for Aggregation and Tranformation
custData <- customers

#Rename Columns
names(custData)[2] <- 'Department_Name'
names(custData)[3] <- 'Annual_Salary'
names(custData)[4] <- 'Gross_Pay_Last_Paycheck'
names(custData)[5] <- 'Gross_Year_To_Date'
names(custData)[6] <- 'Gross_Year_To_Date_FRS_Contribution'
names(custData)[8] <- 'Marital_Status'
names(custData)[9] <- 'Country_ID'
names(custData)[12] <- 'Household_Size'
names(custData)[13] <- 'Years_Residence'
names(custData)
```

```
##  [1] "Title"                                "Department_Name"
##  [3] "Annual_Salary"                        "Gross_Pay_Last_Paycheck"
##  [5] "Gross_Year_To_Date"                   "Gross_Year_To_Date_FRS_Contribution"
##  [7] "Age"                                  "Marital_Status"
##  [9] "Country_ID"                           "Education"
## [11] "Occupation"                           "Household_Size"
```

```
## [13] "Years_Residence"                    "Eligible"
```

Data transformation

```
#Categorisation
length(unique(custData$Title))
```

```
## [1] 2290
```

```
length(unique(custData$Department_Name))
```

```
## [1] 42
```

```
length(unique(custData$Marital_Status))
```

```
## [1] 4
```

```
length(unique(custData$Education))
```

```
## [1] 3
```

```
length(unique(custData$Occupation))
```

```
## [1] 4
```

```
# Convert categorical variables to factors
custData$Marital_Status <- as.factor(custData$Marital_Status)
custData$Education <- as.factor(custData$Education)
custData$Occupation <- as.factor(custData$Occupation)
str(custData)
```

```
## 'data.frame':    191317 obs. of  14 variables:
##  $ Title                              : chr  "CORRECTIONAL OFFICER" "POLICE OFFICER" "CORRECTIONAL OF
##  $ Department_Name                    : chr  "CORRECTIONS & REHABILITATION" "POLICE" "CORRECTIONS & R
##  $ Annual_Salary                      : num  54620 65250 62394 37735 64386 ...
##  $ Gross_Pay_Last_Paycheck            : num  2502 3468 4514 1562 6666 ...
##  $ Gross_Year_To_Date                 : num  48025 57932 49968 35470 132851 ...
##  $ Gross_Year_To_Date_FRS_Contribution: num  46617 56223 48501 34433 128949 ...
##  $ Age                                : int  48 60 82 47 75 74 78 46 75 73 ...
##  $ Marital_Status                     : Factor w/ 4 levels "divorced","married",..: 2 3 3 2 3 3 2 3 3
##  $ Country_ID                         : int  52770 52770 52770 52770 52775 52782 52775 52782 52770 52
##  $ Education                          : Factor w/ 3 levels "Bach.","HS-grad",..: 3 3 3 3 3 3 3 3 3 3
##  $ Occupation                         : Factor w/ 4 levels "Cleric.","Exec.",..: 3 3 3 3 3 3 3 3 3 3
##  $ Household_Size                     : int  2 2 2 2 2 2 2 2 2 2 ...
##  $ Years_Residence                    : int  4 4 4 4 4 4 4 4 4 4 ...
##  $ Eligible                           : num  1 1 1 0 1 1 1 0 1 0 ...
```

```r
#Bin Salary
summary(custData$Annual_Salary)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    3744   42537   58987   63568   83850  149446
```

```r
custData$Salary_Group <- cut(custData$Annual_Salary,
                             breaks = c(0, 42537, 58987, 83850, Inf),
                             labels = c("Low", "Medium", "High", "Very High"))
table(custData$Salary_Group)
```

```
##
##       Low    Medium      High Very High
##     47814     47874     46540     49089
```

One hot encoding of categorical data

```r
custData <- cbind(custData, model.matrix(~Marital_Status - 1, data = custData))
custData <- cbind(custData, model.matrix(~Education - 1, data = custData))
custData <- cbind(custData, model.matrix(~Occupation - 1, data = custData))
custData <- cbind(custData, model.matrix(~Salary_Group - 1, data = custData))
str(custData)
```

```
## 'data.frame':    191317 obs. of  30 variables:
##  $ Title                              : chr  "CORRECTIONAL OFFICER" "POLICE OFFICER" "CORRECTIONAL OF
##  $ Department_Name                    : chr  "CORRECTIONS & REHABILITATION" "POLICE" "CORRECTIONS & R
##  $ Annual_Salary                      : num  54620 65250 62394 37735 64386 ...
##  $ Gross_Pay_Last_Paycheck            : num  2502 3468 4514 1562 6666 ...
##  $ Gross_Year_To_Date                 : num  48025 57932 49968 35470 132851 ...
##  $ Gross_Year_To_Date_FRS_Contribution: num  46617 56223 48501 34433 128949 ...
##  $ Age                                : int  48 60 82 47 75 74 78 46 75 73 ...
##  $ Marital_Status                     : Factor w/ 4 levels "divorced","married",..: 2 3 3 2 3 3 2 3 3
##  $ Country_ID                         : int  52770 52770 52770 52770 52775 52782 52775 52782 52770 52
##  $ Education                          : Factor w/ 3 levels "Bach.","HS-grad",..: 3 3 3 3 3 3 3 3 3 3
##  $ Occupation                         : Factor w/ 4 levels "Cleric.","Exec.",..: 3 3 3 3 3 3 3 3 3 3
##  $ Household_Size                     : int  2 2 2 2 2 2 2 2 2 2 ...
##  $ Years_Residence                    : int  4 4 4 4 4 4 4 4 4 4 ...
##  $ Eligible                           : num  1 1 1 0 1 1 1 0 1 0 ...
##  $ Salary_Group                       : Factor w/ 4 levels "Low","Medium",..: 2 3 3 1 3 4 4 2 3 1 ..
##  $ Marital_Statusdivorced             : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ Marital_Statusmarried              : num  1 0 0 1 0 0 1 0 0 1 ...
##  $ Marital_Statussingle               : num  0 1 1 0 1 1 0 1 1 0 ...
##  $ Marital_Statuswidowed              : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ EducationBach.                     : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ EducationHS-grad                   : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ EducationMasters                   : num  1 1 1 1 1 1 1 1 1 1 ...
##  $ OccupationCleric.                  : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ OccupationExec.                    : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ OccupationProf.                    : num  1 1 1 1 1 1 1 1 1 1 ...
##  $ OccupationSales                    : num  0 0 0 0 0 0 0 0 0 0 ...
```

```
##  $ Salary_GroupLow                    : num  0 0 0 1 0 0 0 0 0 1 ...
##  $ Salary_GroupMedium                 : num  1 0 0 0 0 0 0 1 0 0 ...
##  $ Salary_GroupHigh                   : num  0 1 1 0 1 0 0 0 1 0 ...
##  $ Salary_GroupVery High              : num  0 0 0 0 0 1 1 0 0 0 ...
```

Frequency encoding

```
#Title Encoding:
Title_Frequency <- table(custData$Title)
Title_Frequency_DF <- data.frame(Title = names(Title_Frequency),
                                 Frequency_Title = as.vector(Title_Frequency))
custData <- merge(custData, Title_Frequency_DF, by = "Title")
head(custData$Frequency_Title)
```

```
## [1] 517 517 517 517 517 517
```

```
#Department Encoding:
Department_Frequency <- table(custData$Department_Name)
Department_Frequency_DF <- data.frame(
            Department_Name = names(Department_Frequency),
            Frequency_Department = as.vector(Department_Frequency))
custData <- merge(custData, Department_Frequency_DF, by = "Department_Name")
head(custData$Frequency_Department)
```

```
## [1] 1602 1602 1602 1602 1602 1602
```

```
#Country_ID Encoding:
Country_ID_Frequency <- table(custData$Country_ID)
Country_ID_Frequency_DF <- data.frame(
            Country_ID = names(Country_ID_Frequency),
            Frequency_Country_ID = as.vector(Country_ID_Frequency))
custData <- merge(custData, Country_ID_Frequency_DF, by = "Country_ID")
head(custData$Frequency_Country_ID)
```

```
## [1] 2079 2079 2079 2079 2079 2079
```

```
names(custData)
```

```
##  [1] "Country_ID"                          "Department_Name"
##  [3] "Title"                               "Annual_Salary"
##  [5] "Gross_Pay_Last_Paycheck"             "Gross_Year_To_Date"
##  [7] "Gross_Year_To_Date_FRS_Contribution" "Age"
##  [9] "Marital_Status"                      "Education"
## [11] "Occupation"                          "Household_Size"
## [13] "Years_Residence"                     "Eligible"
## [15] "Salary_Group"                        "Marital_Statusdivorced"
## [17] "Marital_Statusmarried"               "Marital_Statussingle"
## [19] "Marital_Statuswidowed"               "EducationBach."
## [21] "EducationHS-grad"                    "EducationMasters"
```

```
## [23] "OccupationCleric."                  "OccupationExec."
## [25] "OccupationProf."                    "OccupationSales"
## [27] "Salary_GroupLow"                    "Salary_GroupMedium"
## [29] "Salary_GroupHigh"                   "Salary_GroupVery High"
## [31] "Frequency_Title"                    "Frequency_Department"
## [33] "Frequency_Country_ID"
```

Standardisation/Normalisation

```
library(e1071)
skewness_Annual_Salary <- skewness(custData$Annual_Salary)
print(skewness_Annual_Salary)
```

```
## [1] 0.4673479
```

```
skewness_Gross_Pay_Last_Paycheck <- skewness(custData$Gross_Pay_Last_Paycheck)
print(skewness_Gross_Pay_Last_Paycheck)
```

```
## [1] 1.154914
```

```
skewness_Gross_Year_To_Date <- skewness(custData$Gross_Year_To_Date)
print(skewness_Gross_Year_To_Date)
```

```
## [1] 0.3794214
```

```
skewness_Gross_Year_To_Date_FRS_Contribution <- skewness(
                                  custData$Gross_Year_To_Date_FRS_Contribution)
print(skewness_Gross_Year_To_Date_FRS_Contribution)
```

```
## [1] 0.3898762
```

```
skewness_Age <- skewness(custData$Age)
print(skewness_Age)
```

```
## [1] -0.01893976
```

Robust scaling

```
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
robustScaling <- function(x)
{
  median <- median(x)
  iqr <- IQR(x)
  return((x-median)/iqr)
}

custData <- custData %>%
  mutate(Annual_Salary = robustScaling(Annual_Salary))

custData <- custData %>%
  mutate(Gross_Pay_Last_Paycheck = robustScaling(Gross_Pay_Last_Paycheck))

custData <- custData %>%
  mutate(Gross_Year_To_Date = robustScaling(Gross_Year_To_Date))

custData <- custData %>%
  mutate(Gross_Year_To_Date_FRS_Contribution = robustScaling(
                                    Gross_Year_To_Date_FRS_Contribution))
```

Z-Score normalisation

```
custData <- custData %>%
  mutate(Age = (Age - mean(Age)) / sd(Age))
```

Observe how the dataset has been transformed

```
head(custData)
```

```
##   Country_ID                  Department_Name                       Title
## 1      52769                      FIRE RESCUE                  FIREFIGHTER
## 2      52769                    WATER AND SEWER                 ACCOUNTANT 2
## 3      52769 REGULATORY AND ECONOMIC RESOURCES                    CHEMIST 1
## 4      52769                    WATER AND SEWER LIME PRODUCTION PLANT OPER 2
## 5      52769                           AVIATION           AIRPORT OPERS SPEC
## 6      52769      CORRECTIONS & REHABILITATION         CORRECTIONAL OFFICER
##   Annual_Salary Gross_Pay_Last_Paycheck Gross_Year_To_Date
## 1     0.8721491               0.5912230          0.8273373
## 2    -0.1562217              -0.2665496         -0.2668561
## 3    -0.4549265              -0.5109661         -0.8840829
## 4     0.2656517               0.4319194          0.6016796
## 5    -0.6211358              -0.6371731         -0.4169006
## 6     0.1516149               2.4464208          1.3098474
##   Gross_Year_To_Date_FRS_Contribution         Age Marital_Status Education
## 1                           0.8229277  2.08847768        married   HS-grad
```

```
## 2                         -0.2679970  0.02103927         single     Bach.
## 3                         -0.8838986  0.68795488        married   HS-grad
## 4                          0.5993743  2.02178612         single     Bach.
## 5                         -0.4174921 -0.77925948         single   Masters
## 6                          1.3041319  1.08810426         single   Masters
##   Occupation Household_Size Years_Residence Eligible Salary_Group
## 1    Cleric.              2               2        1    Very High
## 2      Exec.              3               5        1       Medium
## 3    Cleric.              2               2        0          Low
## 4      Sales              2               3        1         High
## 5      Prof.              2               4        0          Low
## 6      Prof.              2               4        1         High
##   Marital_Statusdivorced Marital_Statusmarried Marital_Statussingle
## 1                      0                    1                    0
## 2                      0                    0                    1
## 3                      0                    1                    0
## 4                      0                    0                    1
## 5                      0                    0                    1
## 6                      0                    0                    1
##   Marital_Statuswidowed EducationBach. EducationHS-grad EducationMasters
## 1                     0             0                1                0
## 2                     0             1                0                0
## 3                     0             0                1                0
## 4                     0             1                0                0
## 5                     0             0                0                1
## 6                     0             0                0                1
##   OccupationCleric. OccupationExec. OccupationProf. OccupationSales
## 1                 1                0               0               0
## 2                 0                1               0               0
## 3                 1                0               0               0
## 4                 0                0               0               1
## 5                 0                0               1               0
## 6                 0                0               1               0
##   Salary_GroupLow Salary_GroupMedium Salary_GroupHigh Salary_GroupVery High
## 1               0                  0                0                     1
## 2               0                  1                0                     0
## 3               1                  0                0                     0
## 4               0                  0                1                     0
## 5               1                  0                0                     0
## 6               0                  0                1                     0
##   Frequency_Title Frequency_Department Frequency_Country_ID
## 1            8206                16988                2079
## 2            1030                16925                2079
## 3              28                 6138                2079
## 4              67                16925                2079
## 5            1368                 8895                2079
## 6           10809                18158                2079
```

```
str(custData)
```

```
## 'data.frame':    191317 obs. of  33 variables:
##  $ Country_ID                        : int  52769 52769 52769 52769 52769 52769 52769 52769 52769 5
##  $ Department_Name                   : chr  "FIRE RESCUE" "WATER AND SEWER" "REGULATORY AND ECONOMI
##  $ Title                             : chr  "FIREFIGHTER" "ACCOUNTANT 2" "CHEMIST 1" "LIME PRODUCTI
```

```
##  $ Annual_Salary                     : num  0.872 -0.156 -0.455 0.266 -0.621 ...
##  $ Gross_Pay_Last_Paycheck           : num  0.591 -0.267 -0.511 0.432 -0.637 ...
##  $ Gross_Year_To_Date                : num  0.827 -0.267 -0.884 0.602 -0.417 ...
##  $ Gross_Year_To_Date_FRS_Contribution: num  0.823 -0.268 -0.884 0.599 -0.417 ...
##  $ Age                               : num  2.088 0.021 0.688 2.022 -0.779 ...
##  $ Marital_Status                    : Factor w/ 4 levels "divorced","married",..: 2 3 2 3 3 3 3 3 1
##  $ Education                         : Factor w/ 3 levels "Bach.","HS-grad",..: 2 1 2 1 3 3 3 3 1 1
##  $ Occupation                        : Factor w/ 4 levels "Cleric.","Exec.",..: 1 2 1 4 3 3 3 3 2 4
##  $ Household_Size                    : int  2 3 2 2 2 2 2 2 3 2 ...
##  $ Years_Residence                   : int  2 5 2 3 4 4 4 4 5 3 ...
##  $ Eligible                          : num  1 1 0 1 0 1 1 1 1 1 ...
##  $ Salary_Group                      : Factor w/ 4 levels "Low","Medium",..: 4 2 1 3 1 3 4 2 4 3 ..
##  $ Marital_Statusdivorced            : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ Marital_Statusmarried             : num  1 0 1 0 0 0 0 0 1 0 ...
##  $ Marital_Statussingle              : num  0 1 0 1 1 1 1 1 0 1 ...
##  $ Marital_Statuswidowed             : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ EducationBach.                    : num  0 1 0 1 0 0 0 0 1 1 ...
##  $ EducationHS-grad                  : num  1 0 1 0 0 0 0 0 0 0 ...
##  $ EducationMasters                  : num  0 0 0 0 1 1 1 1 0 0 ...
##  $ OccupationCleric.                 : num  1 0 1 0 0 0 0 0 0 0 ...
##  $ OccupationExec.                   : num  0 1 0 0 0 0 0 0 1 0 ...
##  $ OccupationProf.                   : num  0 0 0 0 1 1 1 1 0 0 ...
##  $ OccupationSales                   : num  0 0 0 1 0 0 0 0 0 1 ...
##  $ Salary_GroupLow                   : num  0 0 1 0 1 0 0 0 0 0 ...
##  $ Salary_GroupMedium                : num  0 1 0 0 0 0 0 1 0 0 ...
##  $ Salary_GroupHigh                  : num  0 0 0 1 0 1 0 0 0 1 ...
##  $ Salary_GroupVery High             : num  1 0 0 0 0 0 1 0 1 0 ...
##  $ Frequency_Title                   : int  8206 1030 28 67 1368 10809 7 723 1030 101 ...
##  $ Frequency_Department              : int  16988 16925 6138 16925 8895 18158 14 29331 8895 16925 .
##  $ Frequency_Country_ID              : int  2079 2079 2079 2079 2079 2079 2079 2079 2079 2079 ...
```

```
names(custData)
```

```
##  [1] "Country_ID"                          "Department_Name"
##  [3] "Title"                               "Annual_Salary"
##  [5] "Gross_Pay_Last_Paycheck"             "Gross_Year_To_Date"
##  [7] "Gross_Year_To_Date_FRS_Contribution" "Age"
##  [9] "Marital_Status"                      "Education"
## [11] "Occupation"                          "Household_Size"
## [13] "Years_Residence"                     "Eligible"
## [15] "Salary_Group"                        "Marital_Statusdivorced"
## [17] "Marital_Statusmarried"               "Marital_Statussingle"
## [19] "Marital_Statuswidowed"               "EducationBach."
## [21] "EducationHS-grad"                    "EducationMasters"
## [23] "OccupationCleric."                   "OccupationExec."
## [25] "OccupationProf."                     "OccupationSales"
## [27] "Salary_GroupLow"                     "Salary_GroupMedium"
## [29] "Salary_GroupHigh"                    "Salary_GroupVery High"
## [31] "Frequency_Title"                     "Frequency_Department"
## [33] "Frequency_Country_ID"
```

Remove irrelevant columns

```
# Create vector with all columns/attributes that need to be kept
keepColumns <- c("Annual_Salary", "Gross_Pay_Last_Paycheck",
                 "Gross_Year_To_Date", "Gross_Year_To_Date_FRS_Contribution",
                 "Age", "Household_Size", "Years_Residence",
                 "Marital_Statusdivorced", "Marital_Statusmarried",
                 "Marital_Statussingle", "Marital_Statuswidowed",
                 "EducationBach.", "EducationHS-grad", "EducationMasters",
                 "OccupationCleric.", "OccupationExec.", "OccupationProf.",
                 "OccupationSales", "Salary_GroupLow", "Salary_GroupMedium",
                 "Salary_GroupHigh", "Salary_GroupVery High",
                 "Frequency_Title", "Frequency_Department",
                 "Frequency_Country_ID", "Eligible")

# Remove irrelevant columns/attributes by keeping relevant ones
custData <- custData[keepColumns]

str(custData)
```

```
## 'data.frame':    191317 obs. of  26 variables:
##  $ Annual_Salary                     : num  0.872 -0.156 -0.455 0.266 -0.621 ...
##  $ Gross_Pay_Last_Paycheck           : num  0.591 -0.267 -0.511 0.432 -0.637 ...
##  $ Gross_Year_To_Date                : num  0.827 -0.267 -0.884 0.602 -0.417 ...
##  $ Gross_Year_To_Date_FRS_Contribution: num  0.823 -0.268 -0.884 0.599 -0.417 ...
##  $ Age                               : num  2.088 0.021 0.688 2.022 -0.779 ...
##  $ Household_Size                    : int  2 3 2 2 2 2 2 2 3 2 ...
##  $ Years_Residence                   : int  2 5 2 3 4 4 4 4 5 3 ...
##  $ Marital_Statusdivorced            : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ Marital_Statusmarried             : num  1 0 1 0 0 0 0 0 1 0 ...
##  $ Marital_Statussingle              : num  0 1 0 1 1 1 1 1 0 1 ...
##  $ Marital_Statuswidowed             : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ EducationBach.                    : num  0 1 0 1 0 0 0 0 1 1 ...
##  $ EducationHS-grad                  : num  1 0 1 0 0 0 0 0 0 0 ...
##  $ EducationMasters                  : num  0 0 0 0 1 1 1 1 0 0 ...
##  $ OccupationCleric.                 : num  1 0 1 0 0 0 0 0 0 0 ...
##  $ OccupationExec.                   : num  0 1 0 0 0 0 0 0 1 0 ...
##  $ OccupationProf.                   : num  0 0 0 0 1 1 1 1 0 0 ...
##  $ OccupationSales                   : num  0 0 0 1 0 0 0 0 0 1 ...
##  $ Salary_GroupLow                   : num  0 0 1 0 1 0 0 0 0 0 ...
##  $ Salary_GroupMedium                : num  0 1 0 0 0 0 0 1 0 0 ...
##  $ Salary_GroupHigh                  : num  0 0 0 1 0 1 0 0 0 1 ...
##  $ Salary_GroupVery High             : num  1 0 0 0 0 0 1 0 1 0 ...
##  $ Frequency_Title                   : int  8206 1030 28 67 1368 10809 7 723 1030 101 ...
##  $ Frequency_Department              : int  16988 16925 6138 16925 8895 18158 14 29331 8895 16925 .
##  $ Frequency_Country_ID              : int  2079 2079 2079 2079 2079 2079 2079 2079 2079 2079 ...
##  $ Eligible                          : num  1 1 0 1 0 1 1 1 1 1 ...
```

Export to CSV file

```
write.csv(custData, "CustData2_Prepared.csv", row.names = FALSE)
```