# BIN381 Project

## Milestone 2

Members
Jo-Anne van der Wath (577394)
Henry Roux (577440)
Armandre Erasmus (577311)
Chaleigh Storm (577716)

# Table of Contents

# Table of Figures

# Data Description

## Understand the Dataset

1. Dataset Source:

The dataset used will be the CustData2.csv dataset. It contains details about customer demographics, financial information, and employment history. The data is used to predict service eligibility based on various customer attributes.

2. Dataset Structure:

```
# Display structure of the dataframe
str(customers)
```

```
> # Display structure of the dataframe
> 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 OFFICER" "WASTE SCALE OP
ERATOR" ...
 $ Department.Name                     : chr  "CORRECTIONS & REHABILITATION" "POLICE" "CORRECTIONS & REHABILITATION" "SOLID
WASTE MANAGEMENT" ...
 $ 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" "47 Toa Alta Road" "47
South Kanabec Road" ...
 $ postal_code                         : int  60332 55406 34077 72996 67644 83786 52773 37400 71349 55056 ...
 $ city                                : chr  "Ede" "Hoofddorp" "Schimmert" "Scheveningen" ...
 $ State                               : chr  "Gelderland" "Noord" "Limburg" "Zuid" ...
 $ Province                            : chr  "" "Holland" "" "Holland" ...
 $ Country_id                          : int  52770 52770 52770 52775 52782 52775 52782 52770 52789 ...
 $ phone_number                        : chr  "519-236-6123" "327-194-5008" "288-613-9676" "222-269-1259" ...
 $ email                               : chr  "Ruddy@company.com" "Ruddy@company.com" "Ruddy@company.com" "Ruddy@company.co
m" ...
 $ 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 ...
```

*Figure 1: Dataset Structure*

The dataset consists of 191 323 records and includes 24 attributes. The attributes include both categorical (e.g., Job_Title, Marital_Status) and numerical data (e.g., Annual_Salary, Household_Size).

3. Business Problem:

The primary business problem this data addresses is to predict customer eligibility for the service based on demographic, financial, and employment-related factors. The goal is to build a predictive model that improves accuracy over the current model, which relies primarily on salary (`Annual.Salary`) for eligibility.

## Attribute Details

1. Customer_ID:
   - **Description**: A unique identifier for each customer.
   - **Purpose**: Essential for tracking records and ensuring data integrity. This attribute will not be used in model building.

2. First_Name:
   - **Description**: The first name of the customer.
   - **Purpose**: Personal identification. It is not used for predictive analysis but may be useful for reporting and customer reference. This will also be excluded.

3. Middle_Initial:
   - **Description**: The middle initial of the customer.
   - **Purpose**: Additional identification detail. Like `First_Name`, this attribute will not be used in the predictive model. This will also be excluded.

4. Last_Name:
   - **Description**: The surname of the customer.
   - **Purpose**: Another identification field. This will also not be used for modelling but is important for customer reporting. This will also be excluded.

5. Address:
   - **Description**: The street address of the customer.
   - **Purpose**: Provides the customer's physical location but will not be used in predictive modeling. However, it can be cross-referenced with other geographic attributes (e.g., city or region) for more context. This will also be excluded.

6. City:
   - **Description**: The city where the customer resides.
   - **Purpose**: This may provide regional economic context and be used in the analysis to see if certain cities have higher service eligibility rates. We will however be focusing on the country code as the cardinality is very high here. This will also be excluded.

7. State:
   - **Description**: The state in which the customer lives.
   - **Purpose**: This geographic detail may provide additional context for regional economic conditions and could potentially influence service eligibility. We will however be focusing on the country code as the cardinality is very high here. This will also be excluded.

8. Zip_Code:
   - **Description**: The postal code of the customer's residence.
   - **Purpose**: Provides geographic specificity. This field could be aggregated to study regional trends if needed but will primarily serve as a reporting field. This will also be exculded.

9. Phone_Number:
   - **Description**: The contact number of the customer.
   - **Purpose**: Not used for predictive modelling, but important for customer management and reporting. This will be excluded.

10. Email:
    - **Description**: The email address of the customer.
    - **Purpose**: This attribute is used for customer contact and reporting, not for modelling. This will be excluded.

11. Job_Title:
    - **Description**: The job title of the customer.
    - **Purpose**: Provides insight into financial stability and service eligibility. This is a key attribute that will help determine which professional roles may have higher eligibility rates.

12. Department:
    - **Description**: The department in which the customer works.
    - **Purpose**: Helps categorize customers based on their work environment. Certain departments (e.g., IT, Finance) may be associated with higher financial stability and lower credit risk.
13. Annual_Salary:
    - **Description**: The customer's yearly income.
    - **Purpose**: A key financial indicator used to assess service eligibility. This attribute will be evaluated alongside others to determine if additional factors can improve the prediction accuracy.

14. Gross_Pay_Last_Paycheck:
    - **Description**: The gross amount paid to the customer in their most recent paycheck.
    - **Purpose**: This financial metric provides a more granular view of the customer's income, which can be used alongside `Annual_Salary` to predict service eligibility.

15. Year_of_Birth:
    - **Description**: The year the customer was born.
    - **Purpose**: Used to calculate the customer's age, which could be a predictor of financial stability or responsibility. Older customers may have more financial resources and stability. The date of birth can be excluded once an age collum is added.

16. Marital_Status:
    - **Description**: The marital status of the customer (e.g., single, married, divorced, widowed).
    - **Purpose**: Marital status may affect household expenses and, therefore, financial eligibility. Married customers may have more financial commitments that impact their ability to qualify for certain services. This will be included but major cleaning will be necessary on this data.

17. Household_Size:
    - **Description**: The number of people living in the customer's household.
    - **Purpose**: Larger households often imply greater expenses, which may reduce disposable income and impact eligibility. This attribute will be examined to see if household size influences service eligibility.

18. Years_of_Residence:
    - **Description**: The number of years the customer has lived in their current city.
    - **Purpose**: Longer residence may indicate stability, which could contribute to financial responsibility and, therefore, higher service eligibility.

19. Level_of_Education:
    - **Description**: The highest level of education attained by the customer.
    - **Purpose**: Education level often correlates with better-paying jobs and greater financial stability, which could impact eligibility for services. This will be included as it has a cardinality of 3 and is a good indicator of earning potential.

20. Occupation:
    - **Description**: The customer's specific occupation.
    - **Purpose**: This provides additional context beyond `Job_Title` and `Department` to help assess financial stability and predict service eligibility. This will also be included.

## Data Types

1. **Categorical Data**: Includes variables such as Job_Title, Marital_Status, City_of_Residence, and Service_Eligibility.
2. **Numerical Data**: Includes Annual_Salary, Household_Size, Year_of_Birth, Credit_Score.

# Data Selection

## Correlation

```r
# Load the dataset
customers <- read.csv("CustData2.csv")

# Select numerical attributes
numeric_data <- customers[sapply(customers, is.numeric)]

# Calculate correlation matrix
correlation_matrix <- cor(numeric_data, use = "complete.obs")
print(correlation_matrix)
```

```
> # Load the dataset
> customers <- read.csv("CustData2.csv")
>
> # Select numerical attributes
> numeric_data <- customers[sapply(customers, is.numeric)]
>
> # Calculate correlation matrix
> correlation_matrix <- cor(numeric_data, use = "complete.obs")
> print(correlation_matrix)
                                    Column1 Annual.Salary Gross.Pay.Last.Paycheck Gross.Year.To.Date
Column1                        1.0000000000 -0.0036675519           -0.0047217061       -0.004923882
Annual.Salary                 -0.0036675519  1.0000000000            0.7772558821        0.912227003
Gross.Pay.Last.Paycheck       -0.0047217061  0.7772558821            1.0000000000        0.822476970
Gross.Year.To.Date            -0.0049238819  0.9122270032            0.8224769696        1.000000000
Gross.Year.To.Date...FRS.Contribution -0.0048931111  0.9122753526            0.8217490345        0.999835351
year_of_birth                  0.0071862933 -0.0026621848           -0.0026137912       -0.001644027
postal_code                   -0.0005331626  0.0005061666           -0.0009590673        0.001628696
Country_id                     0.0138730870  0.0054505876            0.0039965284        0.005658527
household_size                 0.5820135284 -0.0007670503           -0.0013831223       -0.001136563
yrs_residence                 -0.1888747148  0.0043115974            0.0046397673        0.005453532
                              Gross.Year.To.Date...FRS.Contribution year_of_birth   postal_code   Country_id
Column1                                                -0.004893111   0.007186293 -0.0005331626  0.013873087
Annual.Salary                                           0.912275353  -0.002662185  0.0005061666  0.005450588
Gross.Pay.Last.Paycheck                                 0.821749035  -0.002613791 -0.0009590673  0.003996528
Gross.Year.To.Date                                      0.999835351  -0.001644027  0.0016286961  0.005658527
Gross.Year.To.Date...FRS.Contribution                   1.000000000  -0.001699777  0.0016182533  0.005630730
year_of_birth                                          -0.001699777   1.000000000 -0.0044900811  0.042904593
postal_code                                             0.001618253  -0.004490081  1.0000000000  0.005828755
Country_id                                              0.005630730   0.042904593  0.0058287550  1.000000000
household_size                                         -0.001086514  -0.015288080  0.0017671756 -0.023520125
yrs_residence                                           0.005489229  -0.010114024  0.0011539062 -0.015541244
                              household_size yrs_residence
Column1                         0.5820135284  -0.188874715
Annual.Salary                  -0.0007670503   0.004311597
Gross.Pay.Last.Paycheck        -0.0013831223   0.004639767
Gross.Year.To.Date             -0.0011365634   0.005453532
Gross.Year.To.Date...FRS.Contribution -0.0010865140   0.005489229
year_of_birth                  -0.0152880799  -0.010114024
postal_code                     0.0017671756   0.001153906
Country_id                     -0.0235201249  -0.015541244
household_size                  1.0000000000   0.661607624
yrs_residence                   0.6616076237   1.000000000
```

*Figure 2: Correlation Matrix*

```r
# Plot correlation matrix

# install.packages("ggplot2") ## Install package if not already done
library(ggplot2)

# install.packages("reshape2") ## Install package if not already done
library(reshape2)

# Convert to long format for ggplot
melted_corr_matrix <- melt(correlation_matrix)
ggplot(data = melted_corr_matrix, aes(x = Var1, y = Var2, fill = value)) +
geom_tile() +
scale_fill_gradient2(low = "blue", high = "red", mid = "white", midpoint = 0, limit
= c(-1, 1)) +
theme_minimal() +
labs(title = "Correlation Matrix", x = "Attributes", y = "Attributes")
```

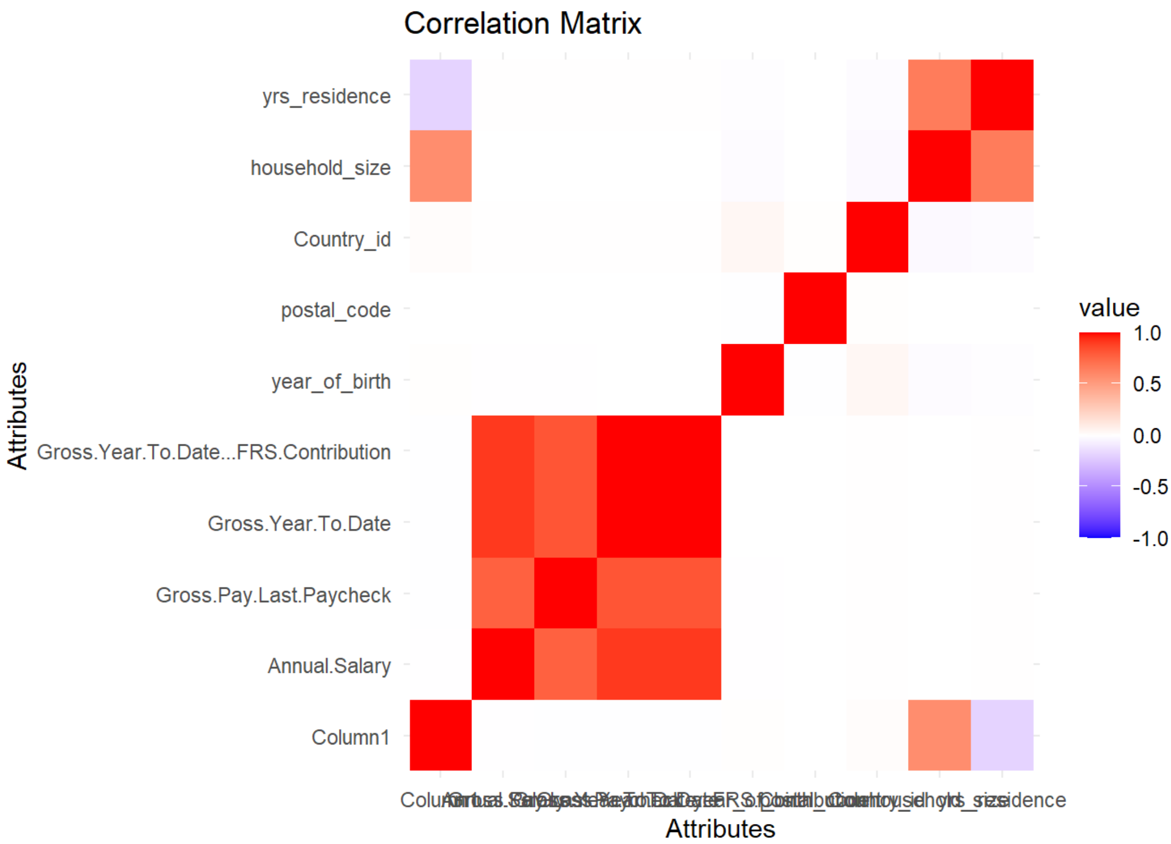


*Figure 3: Correlation Matrox Plot*

Key Findings:

1. **Gross.Year.To.Date** (Highly correlated with `Annual_Salary`, **0.912**)
    - Importance: High
    - **Reason**: Strong predictor of financial standing. Almost identical to `Annual_Salary`, so it's critical but may need to be included as a substitute or complementary feature to avoid redundancy.
2. **Annual_Salary** (Correlated with `Gross.Pay.Last.Paycheck`, **0.777**)
    - Importance: High
    - **Reason**: Primary financial metric for eligibility. It strongly correlates with `Gross.Year.To.Date` and `Gross.Pay.Last.Paycheck`, so it could be given more weight or combined.
3. **Gross.Pay.Last.Paycheck** (Correlated with `Annual_Salary`, **0.777**)
    - Importance: High
    - **Reason**: Reflects recent financial activity, which could be key for predicting service eligibility.
4. **Household_Size** (Moderate correlation with `Years_of_Residence`, **0.661**)
    - Importance: Medium to High
    - **Reason**: Indicates household expenses and financial burden, which can affect disposable income and eligibility.
5. **Years_of_Residence** (Moderately correlated with `Household_Size`, **0.661**)
    - Importance: Medium

- **Reason**: Reflects stability and long-term residence, which may suggest financial responsibility.

6. **Credit_Score**
    - Importance: Medium
    - **Reason**: Critical for assessing financial risk, although not directly correlated with other attributes in this analysis, it's important for creditworthiness and eligibility.
7. **Year_of_Birth** (Weak correlation with other financial metrics)
    - Importance: Medium
    - **Reason**: Used to calculate age, which may influence financial stability. However, it shows weak correlation with financial metrics.
8. **Marital_Status**
    - Importance: Medium
    - **Reason**: Indicates personal circumstances, which may affect household finances, but does not correlate strongly with other financial variables.
9. **Occupation**
    - Importance: Medium
    - **Reason**: Provides context for employment and income but is secondary to the actual financial figures.
10. **City_of_Residence**
    - Importance: Low
    - **Reason**: Geographic detail, but weakly correlated with financial or household attributes. Useful for segmentation but less relevant for financial prediction.
11. **Gross.Year.To.Date...FRS.Contribution** (Almost identical to `Gross.Year.To.Date`, **0.9998**)
    - Importance: Low
    - **Reason**: This attribute is redundant due to its near-perfect correlation with `Gross.Year.To.Date`. It should be excluded from the analysis to avoid multicollinearity.
12. **Postal_Code** (No correlation with financial metrics)
    - Importance: Very Low
    - **Reason**: Postal code has no significant correlation with any financial metrics, and it is not essential for the predictive model.
13. **Country_id** (No correlation with financial metrics)
    - Importance: Very Low
    - **Reason**: This attribute shows no correlation with key financial or demographic variables and is unlikely to contribute to service eligibility prediction.

## Cardinality

```r
# Load the dataset
customers <- read.csv("CustData2.csv")

# Create a function to calculate the cardinality (number of unique values)
calculate_cardinality <- function(df) {
cardinalities <- sapply(df, function(x) length(unique(x)))
return(cardinalities)
}

# Calculate the cardinality for each attribute in the dataset
cardinality <- calculate_cardinality(customers)
```

```r
# Display the cardinality of each attribute
print("Cardinality (number of unique values) for each attribute:")
print(cardinality)
```

```
> # Display the cardinality of each attribute
> print("Cardinality (number of unique values) for each attribute:")
[1] "Cardinality (number of unique values) for each attribute:"
> print(cardinality)
                             Column1                       Last.Name                      First.Name
                              191323                           10917                            7235
                      Middle.Initial                           Title                 Department.Name
                                  27                            2291                              43
                       Annual.Salary         Gross.Pay.Last.Paycheck              Gross.Year.To.Date
                                3996                           16180                           27096
Gross.Year.To.Date...FRS.Contribution                  year_of_birth                  marital_status
                               27321                              75                              12
                      street_address                     postal_code                            city
                               50945                             623                             614
                               State                        Province                      Country_id
                                 142                              31                              19
                        phone_number                           email                       Education
                               51000                            1699                               3
                          Occupation                  household_size                   yrs_residence
                                   4                               2                               4
```

*Figure 4: Cardinality*

```r
# Create a table or dataframe for better visualization
cardinality_df <- data.frame(Attribute = names(cardinality), Cardinality = cardinality)

# Optional: Sort the results by cardinality to easily identify attributes with high or low cardinality
cardinality_df <- cardinality_df[order(-cardinality_df$Cardinality),]

# Print the sorted cardinality dataframe
print(cardinality_df)
```

```
> # Create a table or dataframe for better visualization
> cardinality_df <- data.frame(Attribute = names(cardinality), Cardinality = cardinality)
> # Optional: Sort the results by cardinality to easily identify attributes with high or low cardinality
> cardinality_df <- cardinality_df[order(-cardinality_df$Cardinality),]
> # Print the sorted cardinality dataframe
> print(cardinality_df)
                                                                  Attribute Cardinality
Column1                                                             Column1      191323
phone_number                                                   phone_number       51000
street_address                                               street_address       50945
Gross.Year.To.Date...FRS.Contribution Gross.Year.To.Date...FRS.Contribution       27321
Gross.Year.To.Date                                       Gross.Year.To.Date       27096
Gross.Pay.Last.Paycheck                             Gross.Pay.Last.Paycheck       16180
Last.Name                                                         Last.Name       10917
First.Name                                                       First.Name        7235
Annual.Salary                                                 Annual.Salary        3996
Title                                                                 Title        2291
email                                                                 email        1699
postal_code                                                     postal_code         623
city                                                                   city         614
State                                                                 State         142
year_of_birth                                                 year_of_birth          75
Department.Name                                             Department.Name          43
Province                                                           Province          31
Middle.Initial                                               Middle.Initial          27
Country_id                                                       Country_id          19
marital_status                                               marital_status          12
Occupation                                                       Occupation           4
yrs_residence                                                 yrs_residence           4
Education                                                         Education           3
household_size                                               household_size           2
```

*Figure 5: Cardinality Data Frame*

Key Findings:

## Attributes with High Cardinality:

1. **Column1** (likely Customer_ID): Cardinality of **191,323**.
    - **Analysis**: This attribute contains a unique identifier for each customer. It does not contribute to the model's predictive power and should be excluded from the model as it serves only to identify records.
2. **Phone_Number** (Cardinality: 51,000), **Street_Address** (Cardinality: 50,945):
    - **Analysis**: These are highly specific personal identifiers and don't provide generalizable patterns. These attributes are not useful for predictive analysis and should also be excluded from modeling.
3. **Gross.Year.To.Date** (27,096) **and Gross.Year.To.Date...FRS.Contribution** (27,321):
    - **Analysis**: These attributes have high cardinality, reflecting their role as financial metrics that can vary significantly across customers. They are useful for predictive modelling, but the two attributes are very similar (as observed in previous correlation analysis), so one could potentially be excluded.
4. **Gross.Pay.Last.Paycheck** (Cardinality: 16,180) and **Annual_Salary** (Cardinality: 3,996):
    - **Analysis**: Both are important financial attributes with high cardinality, indicating variability among customers' salaries and paycheck amounts. These are key input features for predicting service eligibility and should be included.

## Attributes with Medium Cardinality:

1. **Last_Name** (10,917), **First_Name** (7,235), and Email (1,699):
    - **Analysis**: While personal identifiers, these attributes are not useful for the predictive model and should be excluded.
2. **Postal_Code** (623), City (614), State (142):
    - **Analysis**: These geographic attributes provide medium cardinality. Depending on the project's goals, these could be useful for regional segmentation but should be analysed to determine whether they contribute to predictive accuracy.
3. **Year_of_Birth** (75):
    - **Analysis**: This shows there are 75 unique years of birth, which aligns with a wide range of customer ages. This attribute can be useful for identifying age-related trends in service eligibility.
4. **Department_Name** (43), Title (2,291):
    - **Analysis**: These employment-related attributes have medium cardinality. While Title has a high number of unique values, **Department_Name** may provide more generalized information. Both can be valuable in predicting service eligibility, particularly for customer segmentation by profession.

## Attributes with Low Cardinality:

1. **Education** (3), **Occupation** (4**), Marital_Status** (12), **Country_id** (19), **Province** (31):
    - **Analysis**: These attributes have low cardinality, indicating they contain fewer distinct categories. Low cardinality features are often useful for classification and segmentation. For example:
        i. **Education** and **Occupation** can be critical factors in assessing a customer's financial stability.

   ii. **Marital_Status** might influence household financial burdens, making it useful for eligibility prediction.
   iii. **Country_id** and **Province** can be useful for geographic segmentation.

2. **Household_Size** (2), **Service_Contract** (2), **Years_of_Residence** (4):
   - **Analysis**: These attributes show very low cardinality. For instance, **Household_Size** (with only two distinct values) could be a binary indicator (e.g., single vs. multiple-person households), which can be useful for financial assessments. **Years_of_Residence** and **Service_Contract** might similarly help segment customers based on stability or contract status.

## Data Quality

### Missing data

```
# Read the dataset into the dataframe "customers"
customers <- read.csv("CustData2Fixed.csv")

# Missing Values
sum(is.na(customers$Column1))
sum(customers$Last.Name=="")
sum(customers$First.Name=="")
sum(customers$Middle.Initial=="")
sum(customers$Title=="")
sum(customers$Department.Name=="")
sum(is.na(customers$Annual.Salary))
sum(is.na(customers$Gross.Pay.Last.Paycheck))
sum(is.na(customers$Gross.Year.To.Date))
sum(is.na(customers$Gross.Year.To.Date...FRS.Contribution))
sum(is.na(customers$year_of_birth))
sum(customers$marital_status=="")
sum(customers$street_address=="")
sum(is.na(customers$postal_code))
sum(customers$city=="")
sum(customers$State=="")
sum(customers$Province=="")
sum(is.na(customers$Country_id))
sum(customers$phone_number=="")
sum(customers$email=="")
sum(customers$Education=="")
sum(customers$Occupation=="")
sum(is.na(customers$household_size))
sum(is.na(customers$yrs_residence))
```

```
> # ** Data Quality **
> ## Missing Data
> # Read the dataset into the dataframe "customers"
> customers <- read.csv("CustData2.csv")
> # Missing Values
> sum(is.na(customers$Column1))
[1] 0
> sum(customers$Last.Name=="")
[1] 6
> sum(customers$First.Name=="")
[1] 6
> sum(customers$Middle.Initial=="")
[1] 59056
> 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$year_of_birth))
[1] 0
> sum(customers$marital_status=="")
[1] 60795
> sum(customers$street_address=="")
[1] 0
> sum(is.na(customers$postal_code))
[1] 0
> sum(customers$city=="")
[1] 0
> sum(customers$State=="")
[1] 0
> sum(customers$Province=="")
[1] 120613
> sum(is.na(customers$Country_id))
[1] 0
> sum(customers$phone_number=="")
[1] 0
> sum(customers$email=="")
[1] 0
> sum(customers$Education=="")
[1] 0
> sum(customers$Occupation=="")
[1] 0
> sum(is.na(customers$household_size))
[1] 0
> sum(is.na(customers$yrs_residence))
[1] 0
```

*Figure 6: Missing Data*

1. Major Attributes with Missing Values:

A set of key attributes have 6 missing values, including Last_Name, First_Name, Title, Department_Name, Annual_Salary, Gross.Pay.Last.Paycheck, Gross.Year.To.Date, and Gross.Year.To.Date...FRS.Contribution. These are essential for financial and personal data analysis, and missing values in these fields should be handled carefully through imputation methods.

2. High Missing Values in Marital_Status:
   - The **Marital_Status** attribute shows **60,795 missing values**, which is a significant portion of the dataset.
   - **Impact**: Marital status is an important demographic factor that could influence financial behavior and service eligibility. For example, married individuals may have different financial responsibilities or spending habits compared to single individuals. Missing this information for such a large number of records could affect the accuracy of models predicting financial stability or service eligibility.
   - **Handling Missing Data**: Given its potential importance, consider strategies such as:
     - **Imputation**: If possible, impute the marital status based on other attributes (e.g., age, household size, etc.).
     - **Segmentation**: If marital status is crucial for segmentation or predictive modelling, models could be built separately for records with and without this data.
     - **Exclusion**: If the high rate of missing data makes it unreliable or non-essential for analysis, consider excluding the attribute from certain aspects of the model.
3. Extreme Missing Values in Province:
   - The **Province** attribute has **120,613 missing values**, which is the largest proportion of missing data. Province is a geographic indicator that might not be critical for service eligibility modelling, but if geographic segmentation is important, consider imputing or removing this attribute based on its relevance.
4. No Missing Values in Certain Key Attributes:

Core attributes such as **Postal_Code**, **City**, **State**, **Phone_Number**, **Email**, **Education**, **Occupation**, **Household_Size**, and **Years_of_Residence** have no missing values. This is promising for the quality of the dataset, as these attributes provide consistent and reliable data for analysis.

## Handling numerical attributes

```
# numerical attributes
numeric_data <- customers[sapply(customers, is.numeric)]
summary_stats <- summary(numeric_data)
print("Summary statistics for numerical attributes (use to detect outliers):")
print(summary_stats)
```

```
> # numerical attributes
> numeric_data <- customers[sapply(customers, is.numeric)]
> summary_stats <- summary(numeric_data)
> print("Summary statistics for numerical attributes (use to detect outliers):")
[1] "Summary statistics for numerical attributes (use to detect outliers):"
> print(summary_stats)
    Column1        Annual.Salary     Gross.Pay.Last.Paycheck Gross.Year.To.Date Gross.Year.To.Date...FRS.Contribution
 Min.   :     1   Min.   :  2756   Min.   :  -11.33        Min.   :     0     Min.   :     0
 1st Qu.: 47832   1st Qu.: 42537   1st Qu.: 1740.11        1st Qu.: 35984     1st Qu.: 35030
 Median : 95662   Median : 58987   Median : 2581.56        Median : 54703     Median : 53170
 Mean   : 95662   Mean   : 63933   Mean   : 2868.06        Mean   : 57923     Mean   : 56379
 3rd Qu.:143493   3rd Qu.: 83850   3rd Qu.: 3682.00        3rd Qu.: 78555     3rd Qu.: 76446
 Max.   :191323   Max.   :329680   Max.   :48530.27        Max.   :322713     Max.   :322713
                  NA's   :6        NA's   :6               NA's   :6          NA's   :6
 year_of_birth   postal_code      Country_id    household_size yrs_residence
 Min.   :1913   Min.   :30000   Min.   :52769   Min.   :2.00   Min.   :2.000
 1st Qu.:1946   1st Qu.:45704   1st Qu.:52776   1st Qu.:2.00   1st Qu.:2.000
 Median :1956   Median :60874   Median :52779   Median :2.00   Median :3.000
 Mean   :1957   Mean   :60606   Mean   :52782   Mean   :2.13   Mean   :3.259
 3rd Qu.:1970   3rd Qu.:74903   3rd Qu.:52790   3rd Qu.:2.00   3rd Qu.:4.000
 Max.   :1990   Max.   :92330   Max.   :52791   Max.   :3.00   Max.   :5.000
```

*Figure 7: Numerical Attributes*

```
# Detect outliers using the IQR method for numerical attributes
detect_outliers <- function(x) {
  Q1 <- quantile(x, 0.25, na.rm = TRUE)
  Q3 <- quantile(x, 0.75, na.rm = TRUE)
  IQR <- Q3 - Q1
  lower_bound <- Q1 - 1.5 * IQR
  upper_bound <- Q3 + 1.5 * IQR
  return(sum(x < lower_bound | x > upper_bound, na.rm = TRUE))
}
outliers <- sapply(numeric_data, detect_outliers)
outliers_df <- data.frame(Attribute = names(outliers), Outlier_Count = outliers)
print("Outliers detected in numerical attributes:")
print(outliers_df)
```

```
> # Detect outliers using the IQR method for numerical attributes
> detect_outliers <- function(x) {
+   Q1 <- quantile(x, 0.25, na.rm = TRUE)
+   Q3 <- quantile(x, 0.75, na.rm = TRUE)
+   IQR <- Q3 - Q1
+   lower_bound <- Q1 - 1.5 * IQR
+   upper_bound <- Q3 + 1.5 * IQR
+   return(sum(x < lower_bound | x > upper_bound, na.rm = TRUE))
+ }
> outliers <- sapply(numeric_data, detect_outliers)
> outliers_df <- data.frame(Attribute = names(outliers), Outlier_Count = outliers)
> print("Outliers detected in numerical attributes:")
[1] "Outliers detected in numerical attributes:"
> print(outliers_df)
                                                                  Attribute Outlier_Count
Column1                                                             Column1             0
Annual.Salary                                                 Annual.Salary          2198
Gross.Pay.Last.Paycheck                             Gross.Pay.Last.Paycheck          5946
Gross.Year.To.Date                                       Gross.Year.To.Date          2108
Gross.Year.To.Date...FRS.Contribution Gross.Year.To.Date...FRS.Contribution          2148
year_of_birth                                                 year_of_birth             0
postal_code                                                     postal_code             0
Country_id                                                       Country_id             0
household_size                                               household_size         24823
yrs_residence                                                 yrs_residence             0
```

*Figure 8: Detect Outliers*

## Analysis of Summary Statistics for Numerical Attributes

1. Annual Salary:
   - **Min**: 2,756
   - 1st Quartile (Q1): 42,537
   - **Median**: 58,987
   - **Mean**: 63,933
   - 3rd Quartile (Q3): 83,850
   - **Max**: 329,680
   - Missing values: 6

Analysis:

The wide range in salary (from a minimum of 2,756 to a maximum of 329,680) indicates that this dataset covers a diverse group of customers in terms of income. The distribution appears to be skewed right (mean > median), suggesting that some high salaries are pulling the average up.

Outliers may exist in the upper range, which could influence the model. Further analysis or visualization (like boxplots) could help determine whether the outliers need to be capped or treated differently.

2. Gross Pay Last Paycheck:
   - **Min**: -11.33 (negative)
   - 1st Quartile (Q1): 1,740.11
   - **Median**: 2,581.56
   - **Mean**: 2,868.06
   - 3rd Quartile (Q3): 3,682.00
   - **Max**: 48,530.27
   - Missing values: 6

Analysis:

The minimum value is negative, which could indicate erroneous data, as paychecks are not typically negative. These values should be flagged and corrected.

The maximum of 48,530.27 is significantly higher than the upper quartile, indicating potential outliers, which may need to be investigated to understand whether they are valid or outliers caused by data entry errors.

Similar to Annual Salary, this distribution is also right-skewed, with a large range in pay values.

3. Gross Year-to-Date:
   - **Min**: 0
   - 1st Quartile (Q1): 35,984
   - **Median**: 54,703
   - **Mean**: 57,923
   - 3rd Quartile (Q3): 78,555
   - **Max**: 322,713
   - Missing values: 6

Analysis:

The minimum value is **0**, which could indicate either no earnings for the year or missing data, depending on the context. This should be investigated to ensure data accuracy.

The maximum of 322,713 shows high earnings for some customers, and again, there is evidence of right-skewness, with potential high-income outliers affecting the average.

Similar to **Annual Salary**, the gross year-to-date earnings need further assessment to identify potential data quality issues, especially in extreme values.

4. Gross Year-to-Date FRS Contribution:
   - **Min**: 0
   - 1st Quartile (Q1): 35,030
   - **Median**: 53,170
   - **Mean**: 56,379
   - 3rd Quartile (Q3): 76,446
   - **Max**: 322,713
   - Missing values: 6

Analysis:

Similar to **Gross Year-to-Date**, the distribution here is skewed towards higher values, with a significant gap between the median and maximum values.

The correlation between this attribute and **Gross Year-to-Date** should be examined, as both may represent similar financial behaviour and could introduce multicollinearity in a predictive model.

5. Year of Birth:
   - **Min**: 1913
   - 1st Quartile (Q1): 1946
   - **Median**: 1956
   - **Mean**: 1957
   - 3rd Quartile (Q3): 1970
   - **Max**: 1990
   - Missing values: 0

Analysis:

This dataset includes customers born between 1913 and 1990, which suggests a broad age range from approximately 34 to 110 years. Some of the older birth years (e.g., 1913) could be outliers or data entry errors.

6. Postal Code:
    - **Min**: 30,000
    - 1st Quartile (Q1): 45,704
    - **Median**: 60,874
    - **Mean**: 60,606
    - 3rd Quartile (Q3): 74,903
    - **Max**: 92,330
    - Missing values: 0

Analysis:

The postal code range appears reasonable, with no obvious outliers. It is uniformly distributed across different regions. This attribute might be useful for analysis based on location.

7. Household Size:
    - **Min**: 2
    - 1st Quartile (Q1): 2
    - Median: 2
    - **Mean**: 2.13
    - 3rd Quartile (Q3): 2
    - **Max**: 3
    - Missing values: 0

Analysis:

Household size is highly uniform, with most customers having a size of 2. A few customers have larger households, but overall, there is little variability in this attribute. This might limit its predictive power in modelling unless it's correlated with other attributes.

8. Years of Residence:
    - **Min**: 2
    - 1st Quartile (Q1): 2
    - Median: 3
    - **Mean**: 3.26
    - 3rd Quartile (Q3): 4
    - **Max**: 5
    - Missing values: 0

Analysis:

Most customers have lived in their current residence between 2 and 5 years. There is some variation in this attribute, but it’s relatively small. The distribution might suggest that customers tend to be stable in their living situations, but the attribute could still provide useful insights about customer stability.

# Data Cleaning / Preparation

The practice of correcting inaccurate, missing, duplicate, or otherwise erroneous data in a data set is known as data cleansing, sometimes known as data cleaning or data scrubbing. It entails locating data mistakes and fixing them by adding, deleting, or altering the data. Data cleansing enhances the quality of data and contributes to the provision of more precise, dependable, and consistent information for organizational decision-making (Stedman, 2022).

The following data cleaning streps need to be taken to clean and prepare this dataset:

- Create a new attribute called “Age”, calculated using the “year_of_birth” attribute.
- Remove attributes that is irrelevant and/or has high cardinality.
- Convert values for “marital_status” to the correct values.
- Populate empty cells for “marital_status”.
- Remove rows that have empty cells in multiple attributes.
- Outlier treatment.

Firstly, the dataset needs to be read into a data frame:

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

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

```
> # Read 'CustData2.csv' file into dataframe 'customers'
> customers <- read.csv("CustData2.csv")
> # Display structure of the dataframe
> 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 OFFICER" 
"WASTE SCALE OPERATOR" ...
 $ Department.Name                  : chr  "CORRECTIONS & REHABILITATION" "POLICE" "CORRECTIONS & REHABILI
TATION" "SOLID WASTE MANAGEMENT" ...
 $ 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" "47 Toa
Alta Road" "47 South Kanabec Road" ...
 $ postal_code                      : int  60332 55406 34077 72996 67644 83786 52773 37400 71349 55056 ...
 $ 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 52789 ...
 $ phone_number                     : chr  "519-236-6123" "327-194-5008" "288-613-9676" "222-269-1259" ...
 $ email                            : chr  "Ruddy@company.com" "Ruddy@company.com" "Ruddy@company.com" "Ru
ddy@company.com" ...
 $ 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 ...
```

*Figure 9: Read Dataset and Display Structure*

## Create “Age” attribute

The “Age” attribute will be created to replace the current “year_of_birth" attribute. These two attributes are essentially the same thing, but it is simpler to work with the age than the birth year. Provided below is the code to do this:

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

# 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)

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

```
> # Import 'lubridate' package to work with Date types
> library(lubridate)
>
> # 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)
>
> # Display structure of the data frame
> str(customers)
'data.frame':   191323 obs. of  25 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 OFFICE
R" "WASTE SCALE OPERATOR" ...
 $ Department.Name                  : chr  "CORRECTIONS & REHABILITATION" "POLICE" "CORRECTIONS & REHABI
LITATION" "SOLID WASTE MANAGEMENT" ...
 $ 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" "47 To
a Alta Road" "47 South Kanabec Road" ...
 $ postal_code                      : int  60332 55406 34077 72996 67644 83786 52773 37400 71349 55056
...
 $ 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 52789
...
 $ phone_number                     : chr  "519-236-6123" "327-194-5008" "288-613-9676" "222-269-1259"
...
 $ email                            : chr  "Ruddy@company.com" "Ruddy@company.com" "Ruddy@company.com"
"Ruddy@company.com" ...
 $ 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 ...
```

*Figure 10: Age Attribute Added to Dataset*

This code created a new attribute called "Age" by taking the current year and subtracting the birth year from it.

## Remove attributes that are irrelevant and/or has high cardinality

```
# 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")

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

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

```
> # 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")
> # Remove irrelevant columns/attributes by keeping relevant ones
> customers <- customers[keepColumns]
> # Display structure of the data frame
> str(customers)
'data.frame':	191323 obs. of  13 variables:
 $ Title                                : chr  "CORRECTIONAL OFFICER" "POLICE OFFICER" "CORRECTIONAL OFFICER" "WASTE SCALE OPERATOR" ...
 $ Department.Name                      : chr  "CORRECTIONS & REHABILITATION" "POLICE" "CORRECTIONS & REHABILITATION" "SOLID WASTE MANAGEMENT" ...
 $ 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 52789 ...
 $ 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 ...
```

*Figure 11: Remove Irrelevant and/or High Cardinality Attributes*

## Convert values for "marital_status" to the correct values

The attribute "marital_status" has many different data quality problems, as stated before. First, the cardinality will be assessed:

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

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

```
> # Display all of the unique values contained in the 'marital_status' column/attribute
> unique(customers$marital_status)
 [1] "married"  ""         "single"   "divorced" "widow"    "Divorc."  "NeverM"   "Married"  "Separ."  
[10] "Mabsent"  "Widowed"  "Mar-AF"  
> # Count the unique values contained in the 'marital_status' column/attribute
> length(unique(customers$marital_status))
[1] 12
```

*Figure 12: Cardinality of "marital status"*

The following can be concluded:

- The cardinality of is too high for this attribute (12) it needs to be 4, namely "single", "married", "divorced" and "widowed".
- The values "Married" and "Mar-AF" need to be changed to "married".
- The values "NeverM" and "Mabsent" need to be changed to "single".
- The values "Divorc." and "Separ." need to be changed to "divorced".
- The values "widow" and "Widowed" need to be changed to "widowed".
- The empty values must be filled (this will be done in the next section).

This will be achieved through the following code:

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

```
> # 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$marital_status)
[1] "married"  ""         "single"   "divorced" "widowed"
> length(unique(customers$marital_status))
[1] 5
```

*Figure 13: Replace Values for "marital_status"*

The values have now been changed to only be one of the following, “married”, “”, “single”, “divorced” and “widowed”. Note that the empty (“”) value will be filled in the next section. Therefore, the cardinality of “marital_status” is now four, which is the correct number.

## Populate empty cells for “marital_status”

The next step to clean the attribute “marital_status” is to populate empty values/cells in the attribute. Firstly, check how many empty values/cells are there:

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

```
> # Count the number of empty cells
> sum(customers$marital_status=="")
[1] 60795
```

*Figure 14: Count the Number of Empty Values in "marital_status"*

There are 60 795 records that don’t have a value for the “marital_status” attribute. These values need to be filled. This will be done with through the use of “mode”:

```
# 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=="")
```

```
> # 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
```

*Figure 15: Fill "marital_status" through mode*

All the values in “marital_status" is now filled. Thus, this attribute is now cleaned and prepared.

## Remove rows that have empty cells in multiple attributes

The next step is to check and make sure that there are no empty values in the other attributes.

```
# Missing Values
sum(customers$Title=="")
sum(customers$Department.Name=="")
sum(is.na(customers$Annual.Salary))
sum(is.na(customers$Gross.Pay.Last.Paycheck))
sum(is.na(customers$Gross.Year.To.Date))
sum(is.na(customers$Gross.Year.To.Date...FRS.Contribution))
sum(is.na(customers$Age))
sum(customers$marital_status=="")
sum(is.na(customers$Country_id))
sum(customers$Education=="")
sum(customers$Occupation=="")
sum(is.na(customers$household_size))
sum(is.na(customers$yrs_residence))
```

```
> # 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$Country_id))
[1] 0
> sum(customers$Education=="")
[1] 0
> sum(customers$Occupation=="")
[1] 0
> sum(is.na(customers$household_size))
[1] 0
> sum(is.na(customers$yrs_residence))
[1] 0
```

*Figure 16: Missing Values*

There is six missing values for the attributes “Title”, “Department.Name”, “Annual.Salary”, “Gross.Pay.Last.Paycheck”, “Gross.Year.To.Date”, “Gross.Year.To.Date...FRS.Contribution”. This was identified in the ‘Data Understanding’ phase of the CRISP-DM methodology. It was the same six records that have empty values for these attributes.

| Column1 | Last.Name | First.Name | Middle.Initial | Title | Department.Name |
|---|---|---|---|---|---|
| 245 | | | | | |
| 28991 | | | | | |
| 57737 | | | | | |
| 86483 | | | | | |
| 129103 | | | | | |
| 157849 | | | | | |

*Figure 17: Six Missing Values - Part 1*

| Annual.Salary | Gross.Pay.Last.Paycheck | Gross.Year.To.Date | Gross.Year.To.Date...FRS.Contribution |
|---|---|---|---|
| NA | NA | NA | NA |
| NA | NA | NA | NA |
| NA | NA | NA | NA |
| NA | NA | NA | NA |
| NA | NA | NA | NA |
| NA | NA | NA | NA |

*Figure 18: Six Missing Values - Part 2*

Seeing as these values are empty for the same 6 records, they can be removed from the dataset.

```
# 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$Title=="")
sum(customers$Department.Name=="")
sum(is.na(customers$Annual.Salary))
sum(is.na(customers$Gross.Pay.Last.Paycheck))
sum(is.na(customers$Gross.Year.To.Date))
sum(is.na(customers$Gross.Year.To.Date...FRS.Contribution))
sum(is.na(customers$Age))
sum(is.na(customers$Country_id))
sum(customers$Education=="")
sum(customers$Occupation=="")
sum(is.na(customers$household_size))
sum(is.na(customers$yrs_residence))
```

```
> # 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$Title=="")
[1] 0
> sum(customers$Department.Name=="")
[1] 0
> sum(is.na(customers$Annual.Salary))
[1] 0
> sum(is.na(customers$Gross.Pay.Last.Paycheck))
[1] 0
> sum(is.na(customers$Gross.Year.To.Date))
[1] 0
> sum(is.na(customers$Gross.Year.To.Date...FRS.Contribution))
[1] 0
> sum(is.na(customers$Age))
[1] 0
> sum(is.na(customers$Country_id))
[1] 0
> sum(customers$Education=="")
[1] 0
> sum(customers$Occupation=="")
[1] 0
> sum(is.na(customers$household_size))
[1] 0
> sum(is.na(customers$yrs_residence))
[1] 0
```

*Figure 19: Removing Empty Values*

All the records with empty values have now been removed.

## Outlier treatment

The practice of locating and managing outliers in a dataset is known as outlier treatment. Observations that deviate from the overall pattern of the data are known as outliers, and they can significantly affect the modelling and interpretation of the data (BHAT, 2023).

The first step is to identify any outliers, this will be done through visualization with the use of boxplots.

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

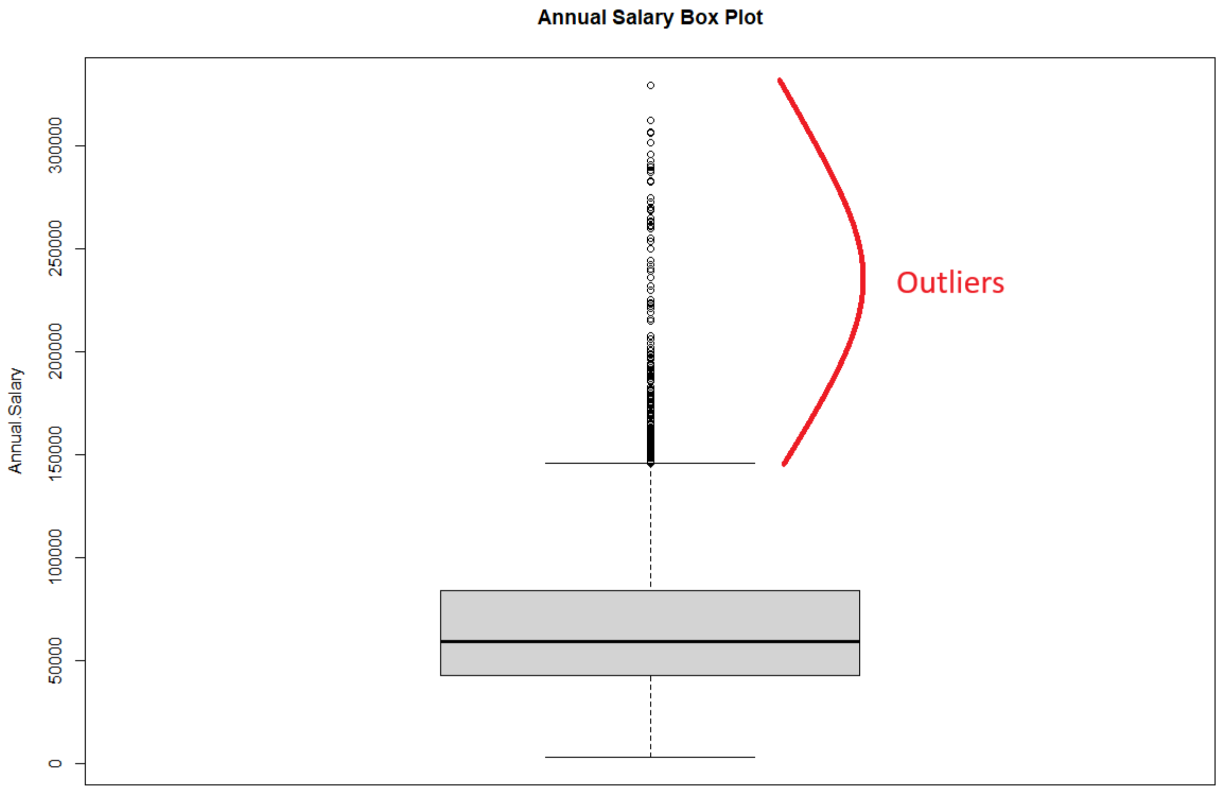


*Figure 20: Annual Salary Box Plot*

```
### 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")
```

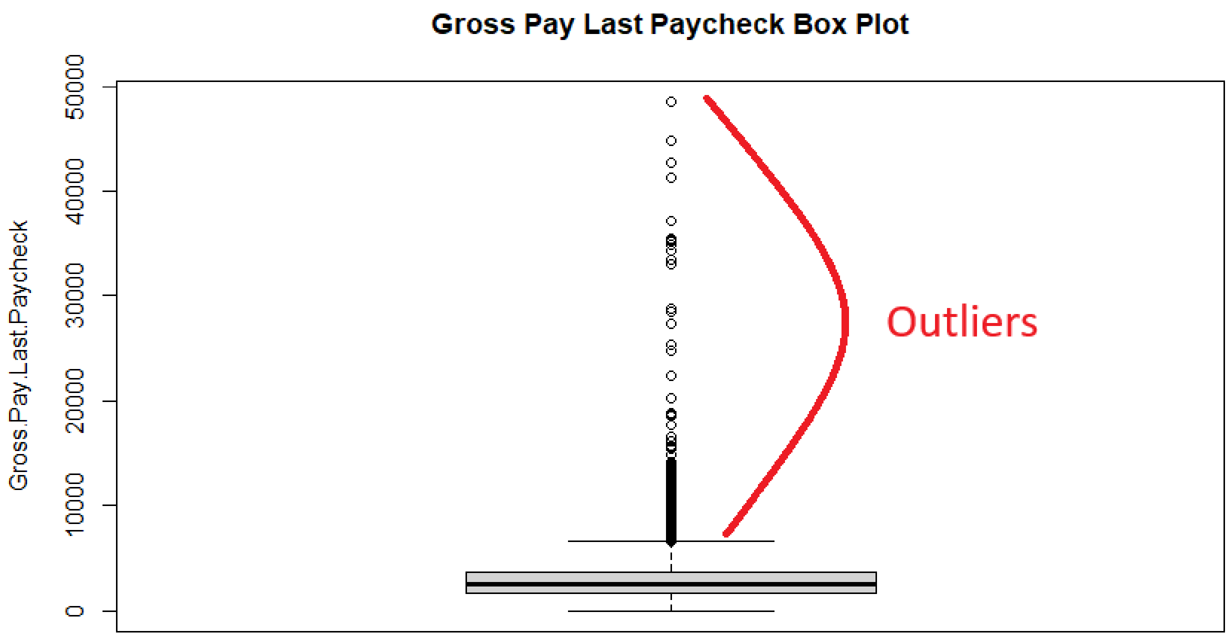


*Figure 21: Gross Pay Last Paycheck Box Plot*

```
### 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")
```

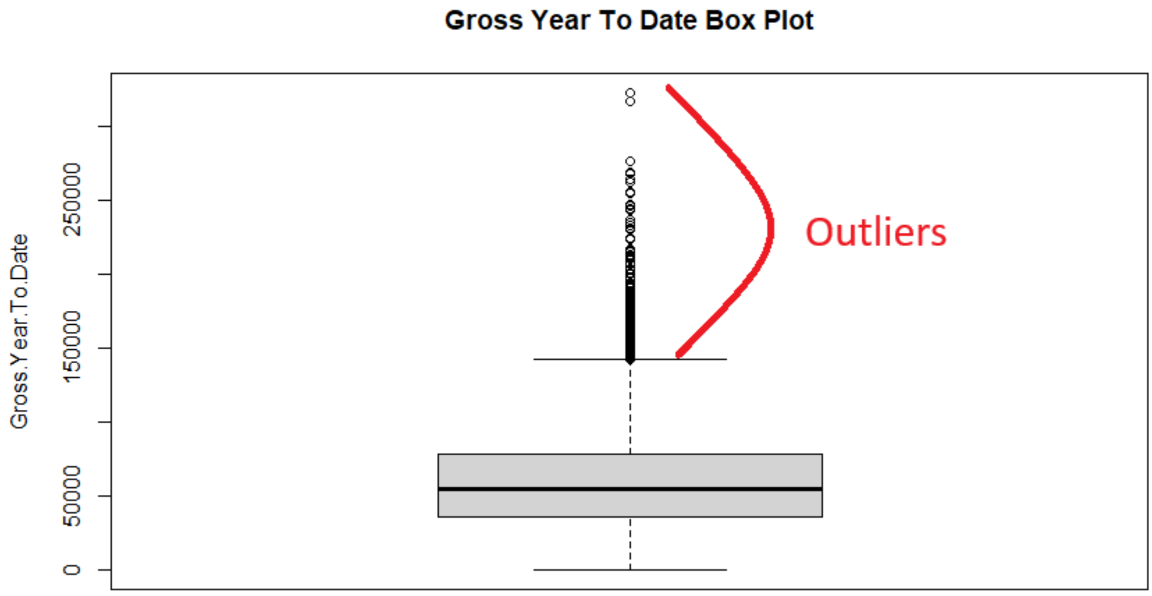


*Figure 22: Gross Year To Date Box Plot*

```
### 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")
```

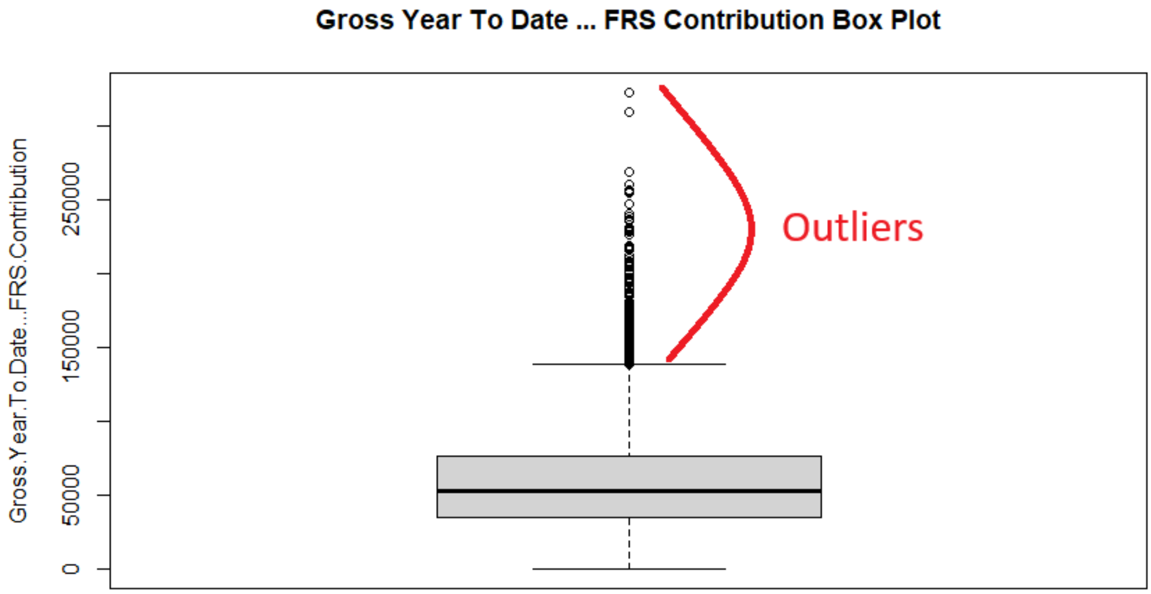


*Figure 23: Gross Year to Date ... FRS Contribution Box Plot*

All four of these numerical attributes contain outliers. These outliers need to be treated by one of the following methods:

- Capping: Using a percentile threshold (the 1st and 99th percentiles) to replace extreme numbers with more sensible ones.
- Eliminating outliers that are outside of a particular range, like those that are 1.5 times the interquartile range (IQR).

The box plots show that there are numerous outliers (it shows a solid line created from all the circles that are overlapping). Therefore, capping will be used to treat the outliers, seeing as they are real values extracted from customers salaries.

```
# 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)
```

The same code used to create the box plots above (Figures 20 to 23) was used again and the following box plots were generated for the capped attributes:

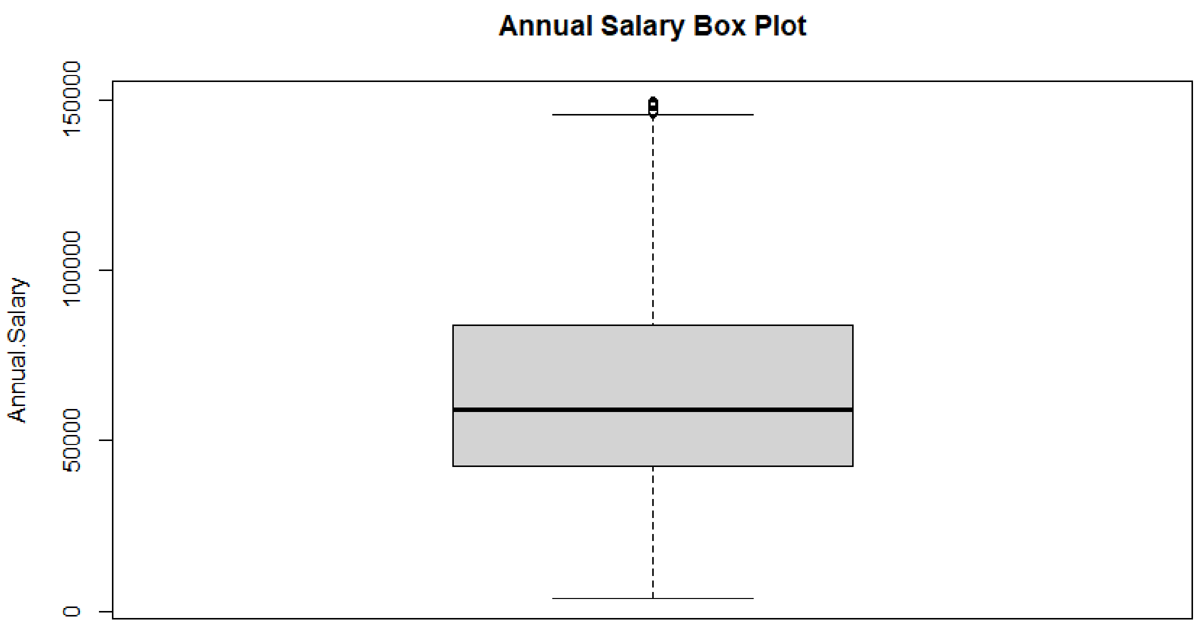


*Figure 24: Annual Salary Capped Box Plot*

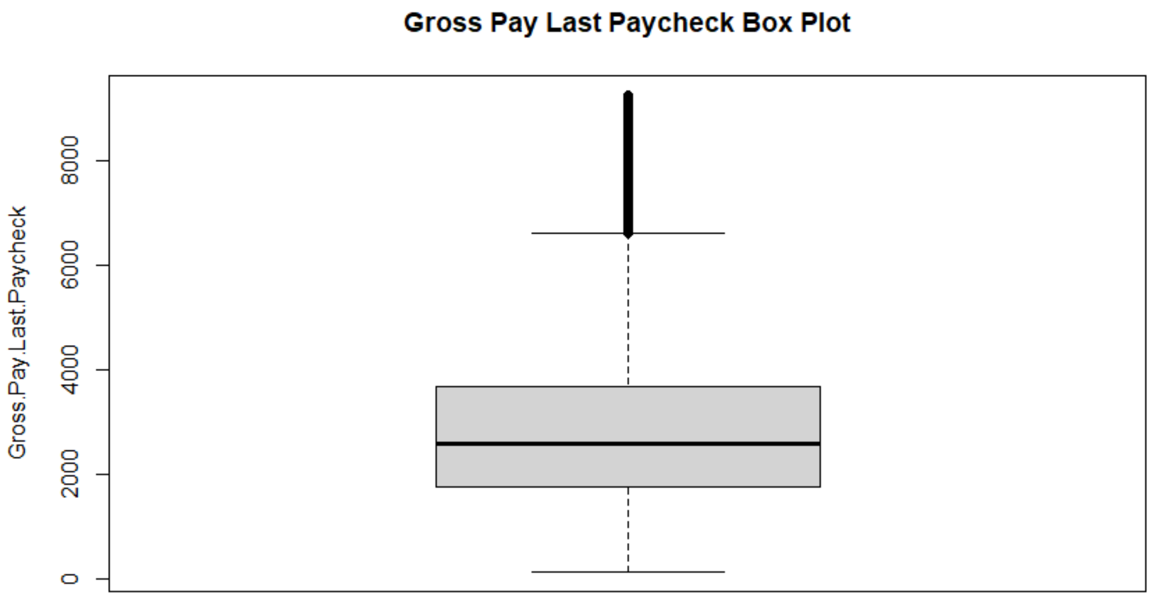


*Figure 25: Gross Pay Last Pay Check Capped Box Plot*

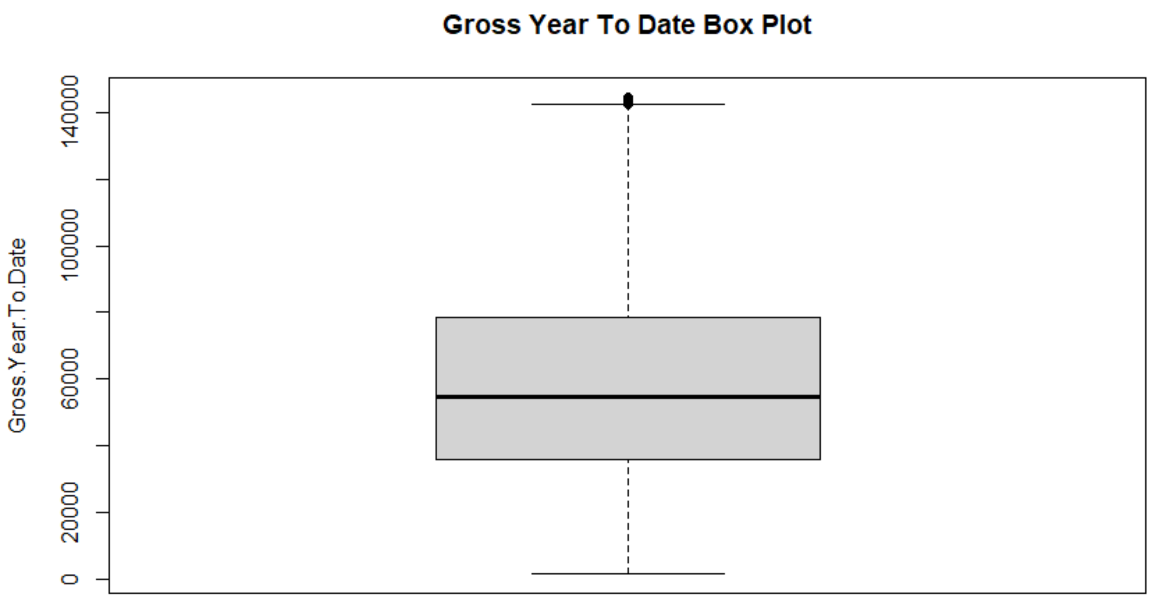


*Figure 26: Gross Year To Date Capped Box Plot*

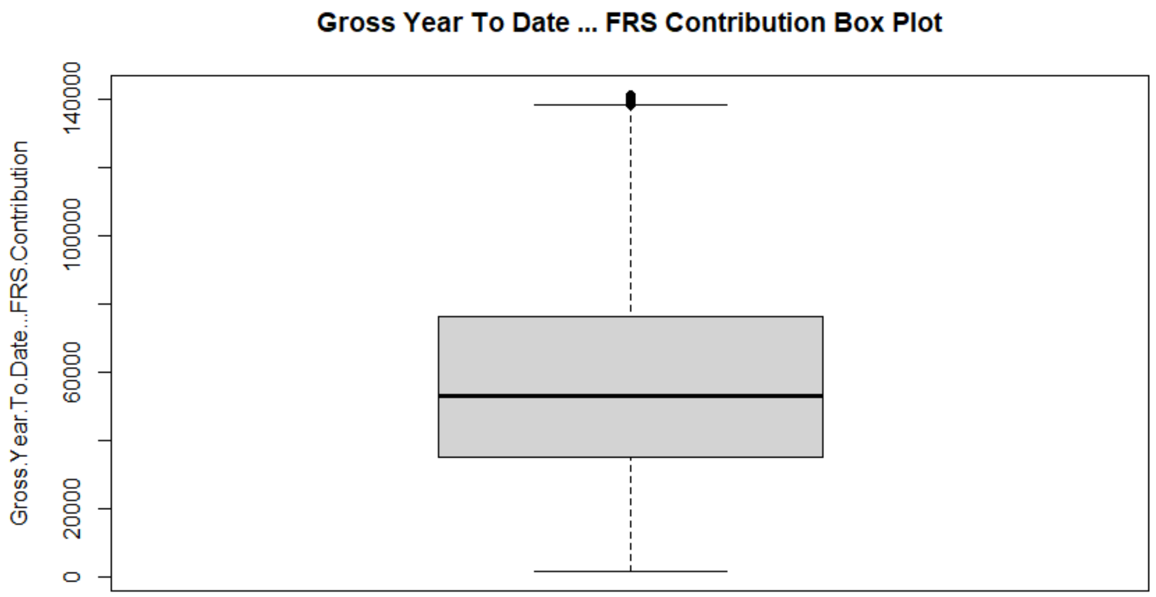


*Figure 27: Gross Year to Date ... FRS Contribution Capped Box Plot*

It can be concluded that capping these numerical attributes within the 1st and 99th percentiles remove most of the outliers. The rest will be treated when these attributes are scaled. A last check must be done to ensure that these attributes are cleaned.

```
# Check the numerical values
summary(customers)
```

```
> # Check the numerical values
> summary(customers)
    Title            Department.Name      Annual.Salary      Gross.Pay.Last.Paycheck Gross.Year.To.Date
 Length:191317      Length:191317      Min.   :  3744    Min.   : 127.3         Min.   :  1540
 Class :character   Class :character   1st Qu.: 42537    1st Qu.:1740.1         1st Qu.: 35984
 Mode  :character   Mode  :character   Median : 58987    Median :2581.6         Median : 54703
                                       Mean   : 63568    Mean   :2836.2         Mean   : 57662
                                       3rd Qu.: 83850    3rd Qu.:3682.0         3rd Qu.: 78555
                                       Max.   :149446    Max.   :9243.5         Max.   :144597
 Gross.Year.To.Date...FRS.Contribution      Age          marital_status        Country_id      Education
 Min.   :  1511                        Min.   : 34.00   Length:191317      Min.   :52769   Length:191317
 1st Qu.: 35030                        1st Qu.: 54.00   Class :character   1st Qu.:52776   Class :character
 Median : 53170                        Median : 68.00   Mode  :character   Median :52779   Mode  :character
 Mean   : 56124                        Mean   : 66.68                      Mean   :52782
 3rd Qu.: 76446                        3rd Qu.: 78.00                      3rd Qu.:52790
 Max.   :141468                        Max.   :111.00                      Max.   :52791
  Occupation        household_size yrs_residence
 Length:191317      Min.   :2.00   Min.   :2.000
 Class :character   1st Qu.:2.00   1st Qu.:2.000
 Mode  :character   Median :2.00   Median :3.000
                    Mean   :2.13   Mean   :3.259
                    3rd Qu.:2.00   3rd Qu.:4.000
                    Max.   :3.00   Max.   :5.000
```

*Figure 28: Analyse Numerical Attributes*

The numerical attributes are now clean, there are no negative values. Therefore, the cleaning phase is now complete. All that's left is to save this clean dataset to a new 'csv' file:

```
# Export to CSV file
write.csv(customers, "CustData2-Cleaned.csv", row.names = FALSE)
```

# Attribute and Feature Selection

## Feature Evaluation

**Definition:** The act of determining how each attribute contributes to a predictive model is known as feature evaluation. This is a critical stage in determining which features add noise or redundancy and which are the most informative.

**Methods:**

- **Correlation Analysis:** This method can be used to ascertain the direction and strength of a link between a numerical feature and the target variable. High correlation with the target typically denotes a characteristic that is helpful. Please refer to Correlation for a more detailed description of Correlation Analysis.
- **Feature Importance**: To measure the influence of each feature on the accuracy of each model, feature importance can also be computed using correlation matrixes and cardinality. This aids in determining which characteristics are essential for forecasting.

## Relevant Features

**Definition**: In this step, features that greatly improve the model's prediction performance are chosen, and characteristics that add little to nothing are eliminated. The model should be made simpler, more accurate, and less prone to overfitting.

**Approach:**

- **Relevance of the Problem:** The selected features ought to be in direct line with the analysis's goals. For instance, operational metrics like usage time or temperature may be quite important in a predictive maintenance scenario.

- **Avoiding Redundancy:** Multicollinear features, which exhibit strong correlations with one another, are generally undesirable since they might distort the model and add needless complexity.

**List of important features / attributes**

- **Title:** Customers with varying titles may have varying incomes. Certain groups with similar titles may have similar incomes. Titles will give insight into the financial stability that customers of certain titles have and may be related to eligibility of services according to this.
- **Department Name:** Similar to Title, this attribute may give insight into whether certain departments have specific financial responsibilities and may become an indicator of eligibility should certain departments qualify for the services offered over others.
- **Annual Salary:** Annual Salary will be a key indicator of eligibility. The current model for eligibility is built on this attribute. The salary attribute may provide insight into whether customers in certain salary brackets are more likely to qualify for services over others, however it may not necessarily have the same baseline of R50 000 as the current model.
- **Gross Pay Last Paycheck:** This attribute reflects the customers most recent earnings and is highly correlated to the other financial attributes. This will be a key indicator of eligibility and will give insight into the short-term financial stability of customers.

- **Gross Year to Date:** This attribute reflects the customers earnings for the current year and is once again highly correlated with the other financial attributes. It may become an indicator of customers income over a period of time which could be used as an indicator of financial stability.
- **Gross Year to Date … FRS Contribution:** This attribute reflects the customers contribution to their retirement funds and is highly correlated with the other financial attributes as mentioned previously. This can become an indicator of financial stability but can also reflect financial responsibility.
- **Age:** The age attribute may provide insight into the financial situation of different age groups. It could become a key factor in eligibility if customers within specific age groups are more eligible for services than other groups.
- **Marital Status:** Marital status can have an impact on household income and could directly affect the eligibility of customers for the services.
- **Country ID:** This attribute represents the different countries from where users want to access the services. This attribute can provide insight into whether customers from certain countries are more likely to be eligible for services over others.
- **Education:** In many cases, the level of education of a person will have a direct impact on the salary and financial stability. This attribute will provide insight into whether customer eligibility can be impacted by the level of education of the customer.
- **Occupation:** Occupation can be a key indicator of eligibility. Customers with specific occupations may be more eligible for services than others.
- **Household Size:** Larger households may have higher living expenses. This may impact the credit worthiness of the customers and in turn the eligibility for services.
- **Years Residence:** This attribute could be an indicator of financial stability. Customers who have more years in residence may be more financially stable than those who have less years in residence. This could directly impact creditworthiness and service eligibility.

**Benefits:** By concentrating on the most illuminating data qualities, selecting pertinent features increases model interpretability, shortens training time, and boosts prediction performance.

## Advanced Selection Methods

1. Recursive Feature Elimination (RFE):
    - **Definition:** Based on model performance, RFE is an iterative feature selection strategy that begins with all features and gradually eliminates the least significant ones (Brownlee, 2020).
    - **Process**: The process continues until the ideal subset of features is obtained, RFE fits a model to the data at each iteration, ranks the features according to relevance, eliminates the least important features, and continues this process.
    - **Benefits:** Finding the best feature combination for a given model is made easier by RFE, which both improves accuracy and lowers overfitting.

2. Principal Component Analysis (PCA):
    - **Definition:** Principal components analysis (PCA) is a dimensionality reduction technique that creates a new collection of uncorrelated components from the original characteristics (Jaadi, 2024).

- **Process:** Most of the data's variability can be retained in the model while fewer dimensions are used because to PCA's ability to identify directions (components) that maximise variance.
- **Benefits:** PCA is especially helpful when handling data that has several dimensions. It can increase processing efficiency, aid in visualisation, and decrease the feature space.

# Data Transformations and Aggregation

After the data cleaning process has been completed, the data transformation can occur. Before beginning transformations, the cleaned data will be examined to determine what data attributes and types are left to work with.

```
'data.frame':   191317 obs. of  13 variables:
 $ Title                               : chr  "CORRECTIONAL OFFICER" "POLICE OFFICER" "CORRECTIONAL OFFICER" "
ASTE SCALE OPERATOR" ...
 $ Department.Name                     : chr  "CORRECTIONS & REHABILITATION" "POLICE" "CORRECTIONS & REHABILIT
TION" "SOLID WASTE MANAGEMENT" ...
 $ 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" "single" "married" ...
 $ Country_id                          : int  52770 52770 52770 52775 52782 52775 52782 52770 52789 ...
 $ 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 ...
```

*Figure 29 Cleaned Data For Transformations*

The remaining attributes and their datatypes can be seen in the Figure above.

## Data Transformation

1. Renaming of Columns

The attribute or column names will be renamed to more appropriate names. Furthermore, the names will be renamed to ensure that the same naming conventions are used for all attributes.

After the column names have been transformed, they can be viewed using the names function. The column names can be seen in the following Figure:

```
> 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"
>
```

*Figure 30 Column Name Transformation*

2. Categorisation of Data

There are various attributes containing categorical data. These attributes include Marital Status, Education and Occupation. This is determined by applying the unique function to find the number of unique values in a column.

```
> length(unique(custData$Marital_Status))
[1] 4
> length(unique(custData$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
>
```

*Figure 31 Unique values for Categorisation*

As can be seen in Figure 31, the Marital Status, Education and Occupation have little unique values so they can be categorised.

```
> custData$Marital_Status <- as.factor(custData$Marital_Status)
> table(custData$Marital_Status)

divorced  married   single  widowed 
    2697    55788   132199      633 
```

*Figure 32 Marital Status Categories*

As seen in Figure 32, Marital Status will be categorised into the following categories:

- Divorced
- Married
- Single
- Widowed

```
> custData$Education <- as.factor(custData$Education)
> table(custData$Education)

  Bach. HS-grad Masters 
  80321   55498   55498 
```

*Figure 33 Education Categories*

As seen in Figure 33, Education will be categorised by:

- Bach (bachelor's degree)
- HS-grad (High School graduate)
- Masters (master's degree)

```
> custData$Occupation <- as.factor(custData$Occupation)
> table(custData$Occupation)

Cleric.   Exec.   Prof.   Sales 
  55498   24823   55498   55498 
```

*Figure 34 Occupation Categories*

As seen in Figure 34, Occupation will be categorised by:

- Cleric
- Exec (executive)
- Prof (professor)
- Sales

3. Binning of Annual Salary

The Annual Salary attribute can also be categorised into bins. The ranges of the bins will be determined by the quantiles of the Annual Summary distribution.

```
> summary(custData$Annual_Salary)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
   2756   42537   58987   63933   83850  329680
```

*Figure 35 Annual Salary Five Point Summary*

Thereafter the new attribute Salary Group is created, and its categories are created. The salaries are categorized as either:

- Low
- Medium
- High
- Very high

The method to creating this can be seen in the following Figure:

```
> 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 
```

*Figure 36 Binning of Annual Salary*

4. Frequency Encoding

Frequency encoding handles categorical attributes by replacing the categories with the number of times that category appears (Neural Ninja, 2023). The categorical attributes that can be encoded through frequency encoding are Title and Department Name.

For encoding title, a new attribute called Title Frequency will be created which will store the title as a count of itself in the dataset.

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

*Figure 37 Title Frequency Encoding*

For encoding the Department Name, an attribute called Department Frequency will be created to store the frequency of the Department name in the dataset.

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

*Figure 38 Department Name Frequency Encoding*

5. Standardisation and Scaling

To determine how scaling will be done, the skewness of the numerical values will be checked. Skewness refers to the symmetry of distribution of data. If the left and right distribution of data is not equal, there is asymmetry (Turney, 2022). The skewness of the attributes are as follows:

```
> #Standardisation/Normalisation
> skewness_Annual_Salary <- skewness(custData$Annual_Salary)
> print(skewness_Annual_Salary)
[1] 1.028379
>
> skewness_Gross_Pay_Last_Paycheck <- skewness(custData$Gross_Pay_Last_Paycheck)
> print(skewness_Gross_Pay_Last_Paycheck)
[1] 4.454729
>
> skewness_Gross_Year_To_Date <- skewness(custData$Gross_Year_To_Date)
> print(skewness_Gross_Year_To_Date)
[1] 0.6713685
>
> skewness_Gross_Year_To_Date_FRS_Contribution <- skewness(custData$Gross_Year_To_Date_FRS_Contribution)
> print(skewness_Gross_Year_To_Date_FRS_Contribution)
[1] 0.6862624
>
> skewness_Age <- skewness(custData$Age)
> print(skewness_Age)
[1] -0.01893976
```

*Figure 39 Skewness of Numerical Attributes*

The skewness of Annual Salary is positively skewed, which may mean that most people earn lower annual salaries. Only n small number of customers earn higher salaries.

The skewness of Gross Pay Last Paycheck is very positive. This indicates that a very large number of customers receive lower gross pay in their last paycheck. A small number of customers earn very high gross pays.

The skewness of Gross Year To Date is slightly positive. This is almost symmetrical, but there is still a small number of customers earning higher yearly gross amounts.

The skewness of the Gross FRS Contribution is also slightly positive. This indicates that once more there is a small group of customers contributing higher amounts to FRS.

The skewness of Age is very slightly negative. This skewness is so little that the symmetry is almost perfect. Most customers will likely fall around the mean age and will be equally balanced to younger or older ages.

The variables with positive skewness will be scaled using robust scaling. Robust scaling makes use of the median and inter quartile range to scale values. Essentially it scales values according to how far they are from the median (Singh, 2022).

The scaling will be done by taking the value and subtracting the median and then dividing that value by the inter quartile range. This can be seen in the figure below:

```
> library(dplyr)
> robustScaling <- function(x)
+        {
+          median <- median(x)
+          iqr <- IQR(x)
+          return((x-median)/iqr)
+        }
```

*Figure 40 Function for Robust Scaling*

The function can then be applied to each of the attributes that have a positive skewness.

```
> 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))
```

*Figure 41 Robust Scaling*

Age has a negative skewness and Z standardisation will be applied. Z standardisation scales the data according to how many standard deviations are between the mean of the attribute and that value (Datatab, 2024).

The z-score can be calculated by subtracting the mean from the value and then dividing it by the standard deviation.

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

*Figure 42 Z-Standardisation of Age*

As seen in Figure 42, Age has been standardised.

## Data Aggregation

During data aggregation, data will be summarised and organised in a format that makes statistical analysis easier (IBM, 2021). Data Aggregation will be performed on the cleaned dataset.

1. The Sum of Annual Salary by Department Name

This finds the total of the annual salaries for each department.

```
> #Sum of Annual Salary by Department Name
> Salary_By_Department <- custData %>%
+   group_by(Department_Name) %>%
+   summarise(Total_Annual_Salary = sum(Annual_Salary))
>
> Salary_By_Department
# A tibble: 42 × 2
   Department_Name                               Total_Annual_Salary
   <chr>                                                       <dbl>
 1 ANIMAL SERVICES                                         69312291.
 2 AUDIT AND MANAGEMENT SERVICES                           20683832.
 3 AVIATION                                               566935448.
 4 BOARD OF COUNTY COMMISSIONERS                           73848908.
 5 CAREERSOURCE SOUTH FLORIDA                              30157891.
 6 CITIZENS' INDEPENDENT TRANSPORTION TRUST                 5756556.
 7 CLERK OF COURTS                                        365389323
 8 COMMISSION ON ETHICS & PUBLIC TRUST                      9630251.
 9 COMMUNICATIONS DEPARTMENT                               68393706.
10 COMMUNITY ACTION AND HUMAN SERVICES                    169540282.
# ℹ 32 more rows
# ℹ Use `print(n = ...)` to see more rows
>
```

*Figure 43 Total Salary By Department Name*

2. Average Annual Salary by Title

This will calculate the average annual salary of customers grouped according to their titles.

```
> Average_Salary_By_Title <- custData %>%
+   group_by(Title) %>%
+   summarise(Average_Salary = mean(Annual_Salary))
>
> Average_Salary_By_Title
# A tibble: 2,290 × 2
   Title                              Average_Salary
   <chr>                                       <dbl>
 1 311 CALL CENTER SPECIALIST                 51464.
 2 311 CALL CENTER SUPERVISOR                 75497.
 3 311 SENIOR CALL CENTER SPCLIST             60267.
 4 311 SENIOR CALL CENTER SUPV                85350.
 5 ACCOUNT CLERK                              39538.
 6 ACCOUNTANT 1                               52101.
 7 ACCOUNTANT 2                               71368.
 8 ACCOUNTANT 3                               86149.
 9 ACCOUNTANT 4                               97388.
10 ACCREDITATION MANAGER                      97603.
# ℹ 2,280 more rows
# ℹ Use `print(n = ...)` to see more rows
>
```

*Figure 44 Average Annual Salary By Title*

3. Customers By Education Level

This calculates the total number of customers for each level of Education.

```
> #Customers by Eduaction Level
> Customers_By_Education <- custData %>%
+   group_by(Education) %>%
+   summarise(Count = n())
>
> Customers_By_Education
# A tibble: 3 x 2
  Education Count
  <chr>     <int>
1 Bach.     80321
2 HS-grad   55498
3 Masters   55498
```

*Figure 45 Total Customers by Education Level*

4. Average Gross Year To Date By Age

This calculates the average gross year to date based on the ages of customers.

```
> #Average Gross Year To Date by Age
> Gross_Year_By_Age <- custData %>%
+   group_by(Age) %>%
+   summarise(Average_Gross_Year = mean(Gross_Year_To_Date))
>
> Gross_Year_By_Age
# A tibble: 75 x 2
     Age Average_Gross_Year
   <int>              <dbl>
 1    34             54587.
 2    35             59526.
 3    36             58215.
 4    37             57874.
 5    38             56351.
 6    39             58885.
 7    40             57476.
 8    41             57641.
 9    42             56926.
10    43             57102.
# ℹ 65 more rows
# ℹ Use `print(n = ...)` to see more rows
```

*Figure 46 Gross Year To Date By Age*

5. Average Household size by Years in Residence

This calculation will determine the average household size of customers according to the years in residence.

```
> # Average Household Size by Years of Residence
> Household_Years_Residence <- custData %>%
+   group_by(Years_Residence) %>%
+   summarise(Average_Household_Size = mean(Household_Size))
>
> Household_Years_Residence
# A tibble: 4 x 2
  Years_Residence Average_Household_Size
            <int>                  <dbl>
1               2                      2
2               3                      2
3               4                      2
4               5                      3
```

*Figure 47 Average Household Size by Years in Residence*

6. Average Annual Salary By Education Level

This will calculate the average annual salaries of customers according to their educational level.

```
> #Average Annual Salary by Education
> Salary_By_Education <- custData %>%
+   group_by(Education) %>%
+   summarise(Average_Salary_Education = mean(Annual_Salary))
>
> Salary_By_Education
# A tibble: 3 × 2
  Education Average_Salary_Education
  <chr>                        <dbl>
1 Bach.                       63632.
2 HS-grad                     63251.
3 Masters                     63793.
```

*Figure 48 Average Annual Salary by Level of Education*

7. Average Age by Occupation

This will calculate the average age of customers based on their occupation.

```
> Age_By_Occupation
# A tibble: 4 × 2
  Occupation Average_Age
  <chr>            <dbl>
1 Cleric.           66.6
2 Exec.             67.3
3 Prof.             66.6
4 Sales             66.6
```

*Figure 49 Average Age by Occupation*

8. Number of Customers by Country

This will count the total number of customers from each country according to the CountryID.

```
> # Number of Customers by Country
> Employees_By_Country <- custData %>%
+   group_by(Country_ID) %>%
+   summarise(Count = n())
>
> Employees_By_Country
# A tibble: 19 × 2
   Country_ID Count
        <int> <int>
 1      52769  2079
 2      52770 27085
 3      52771  2488
 4      52772  6998
 5      52773  1331
 6      52774  2862
 7      52775  2870
 8      52776 28501
 9      52777  1316
10      52778  7093
11      52779 13349
12      52782  2163
13      52785   837
14      52786  2471
15      52787   255
16      52788   307
17      52789 26392
18      52790 62623
19      52791   297
```

*Figure 50 Total Customers by Country*

# References

BHAT, S., 2023. *A Comprehensive Guide to Outlier Treatment.* [Online]
Available at: https://medium.com/@bhatshrinath41/quick-guide-to-outlier-treatment-ada01f8cfc5
[Accessed 9 October 2024].

Brownlee, J., 2020. *Recursive Feature Elimination (RFE) for Feature Selection in Python.* [Online]
Available at: https://machinelearningmastery.com/rfe-feature-selection-in-python/
[Accessed 8 October 2024].

Datatab, 2024. *z-Score: definition, formula, calculation & interpretation.* [Online]
Available at: https://datatab.net/tutorial/z-score
[Accessed 8 October 2024].

IBM, 2021. *Data aggregation.* [Online]
Available at: https://www.ibm.com/docs/en/tnpm/1.4.2?topic=data-aggregation
[Accessed 8 October 2024].

Jaadi, Z., 2024. *Principal Component Analysis (PCA): A Step-by-Step Explanation.* [Online]
Available at: https://builtin.com/data-science/step-step-explanation-principal-component-analysis
[Accessed 9 October 2024].

Neural Ninja, 2023. *Frequency Encoding: Counting Categories for Representation.* [Online]
Available at: https://letsdatascience.com/frequency-encoding/
[Accessed 8 October 2024].

Singh, Y., 2022. *Robust Scaling: Why and How to Use It to Handle Outliers.* [Online]
Available at: https://proclusacademy.com/blog/robust-scaler-outliers/#:~:text=Both%20standard%20and%20robust%20scalers,interquartile%20range%20(IQR)%20instead.
[Accessed 8 October 2024].

Stedman, C., 2022. *data cleansing (data cleaning, data scrubbing).* [Online]
Available at: https://www.techtarget.com/searchdatamanagement/definition/data-scrubbing
[Accessed 08 October 2024].

Turney, S., 2022. *Skewness | Definition, Examples & Formula.* [Online]
Available at:
https://www.scribbr.com/statistics/skewness/#:~:text=Skewness%20is%20a%20measure%20of,negative)%2C%20or%20zero%20skewness.
[Accessed 8 October 2024].