BIN381

Project Report

Members – Group F

Jo-Anne van der Wath (577394)

Henry Roux (577440)

Armandre Erasmus (577311)

Chaleigh Storm (577716)

Table of Contents

Table of Figures

Table of Tables

Introduction

The primary factor for customer eligibility for service contracts with satellite internet service provider LangaSat is annual salary, with a threshold of R50,000. Nonetheless, the business thinks that additional elements, like client demographics and lifestyle choices, can offer insightful information on the eligibility of the customers. The goal of this project is to create a classification model that can forecast client eligibility based on extra variables, enhancing decision-making accuracy. This project will be guided through several phases, including business understanding, data understanding, data preparation, modelling, evaluation, and deployment, using the CRISP-DM (Cross Industry Standard Process for Data Mining) framework.

This report includes all the steps taken to clean and preprocess the given dataset and then to create / train a prediction model. An application was then developed to make use of this model and predict the eligibility of input records that was given to it.

Background

The only criteria for eligibility for a customer service contract that LangaSat currently uses is annual salary, which ignores other signs of a customer's capacity to fulfil the contract. The business has seen that eligibility and credit risk may be impacted by variables such as years of residency, marital status, occupation, and degree of education. By identifying and incorporating these factors into a recommendation system, this project aims to help LangaSat improve the accuracy and efficiency of its client assessment procedure, while also increasing the percentage of eligible customers.

Phase 1 – Business Understanding

Business Problem

LangaSat currently only considers the annual salary of an applicant when determining customer eligibility, which may not account for all variables affecting credit risk. Currently, only prospective customers with an annual salary of R50 000 or more qualify for the services offered.

According to the observations made by LangaSat, prospective customers who have otherwise good credit risk scores may be turned away if wage is the only qualifying factor. By considering a variety of parameters, the organisation hopes to increase the accuracy of its eligibility evaluations (e.g., job title, years of residence, household size, etc.).

The objective is to create a recommender model that, in addition to salary, can more accurately determine whether clients are eligible for their service based on a variety of other factors. This will not only allow for a larger customer selection, but this will also reduce the overall credit risk to the company.

Business Objectives

Primary Objective

Create a classification model that considers more attributes than just salary to accurately determine the eligibility of a customer for the satellite internet services provided by LangaSat.

Secondary Objectives

- Increase the precision of decisions made about customer eligibility.
- Reduce the quantity of false positives and false negatives (i.e., make sure that eligible and unqualified consumers aren't mistakenly rejected and admitted).
- Examine many elements (such as occupation, length of residency, and educational attainment) to determine important components that affect customer eligibility.
- To help decision-makers better understand the model's output, present the results using visualisations (such as Power BI dashboards).
- Increasing the number of customers who are eligible for the services.

The model's improvements in accuracy, precision, and recall should allow it to exceed the baseline, which is now based solely on income.

Stakeholder Requirements

There are multiple stakeholders in the project and each stakeholder has different requirements for the project.

LangaSat

LangaSat, as the primary stakeholder in the project, have the following requirements:

- An intelligent recommender model that should predict whether a customer is eligible for satellite internet service.
- The classification model should determine eligibility by accurately identifying the credit risk of the customer.
- The classification model should be able to determine eligibility based on more factors than the customer salary alone. This is to include a wider range of customers to be considered.
- The classification model should allow LangaSat to acquire more customers while minimizing the credit risk of the customers.

Project Supervisor

The project supervisor, Mr Gift T. Mudare, will have the following requirements from the team:

- The Cross-Industry Process for Data Mining Methodology (CRISP-DM) should be followed during the planning of the project. Thereafter, the project will be implemented using the planning framework derived from the CRISP-DM plan.
- The project supervisor must be made aware of the project goals and must agree with the goals before the continuation of the project to the following milestones.
- The project team will be required to follow the guidance of the project supervisor throughout the project.

Development Team

The project team will have the following requirements:

- The exploratory data analysis (EDA) of the raw data (CustData2.csv) to better understand the dataset.
- The preparation of the raw data to ensure high quality input.
- The development of an accurate classification model that will determine customer eligibility based on credit risk as mentioned under the LangaSat requirements.
- Model performance will need to be evaluated using performance metrics, such as confusion matrix, F1 scores or the accuracy of the model (Tavish, 2024).

- The documentation of the project, included but not limited to the methodologies used, preprocessing of data, exploration of data and model training.
- The model should be free from bias and should not be discriminatory. This is an ethical requirement.

Success Criteria – Data Mining Goals

Success Criteria can be defined as precise standards that a project must satisfy in order to be considered successful (awork, 2024). The requirements will be considered successfully met if:

- The model accurately classifies customers as eligible, or ineligible based on their demographic and social attributes.
- The model accurately determines the eligibility of the customer.
- The model performance should be above acceptable performance metrics.
- There is an increase in the number of customers that are eligible for services.
- There is a reduced financial risk by excluding customers with high credit risk (relatively low income with high expenses) from being eligible.
- Stakeholders are able to identify which variables contribute to for customer eligibility the most.
- The model meets all the requirements set by the stakeholders.
- The project adheres to the CRISP-DM framework throughout all stages of the project.
- Each milestone of the project is completed before the due date.
- The work for the milestone is completed fully according to the project plan and CRISP-DM within the given time frame without compromising on the quality of the work.
- The model does not contain any biases or result in discriminatory classifications of customer eligibility.

Inventory of Resources

Data Resources

- Dataset: The offered dataset (CustData2.csv) includes variables including department names, salary, job titles, and household sizes.
- Data Source: Supplied with consumer demographic data for the LangaSat scenario.

Software & Tools

Data Analysis Tools:

- R: For constructing models, cleaning date, the preprocessing phase and data exploration. (Python could also be used in data analysis, but in this case, R will be used).
- RStudio: Integrating environments for development with coding. (Jupyter Notebook could also be used)

Visualization Tools:

- Power BI will be used to create dashboards and visualisations which will be used to show correlations, patterns, and model results.

Collaboration Tools:

- For exchanging documents, code, and collaborative projects, use GitHub, Google Drive or Microsoft Teams.

Machine Learning Libraries:

- Relevant R libraries could be used for constructing models for classification.

Human Resources:

- All the fellow students working on the project, each with a specific responsibility (e.g., data cleaning, model building, reporting), make up the group.
- Project Supervisor: Instructor offering direction and input on the project's development. (In our case this would be our lector giving feedback)

Hardware Resources:

- Computers/Laptops: Individual machines will be used for modelling, analysis, and visualisation.
- Cloud systems: For computational power and cooperation, Google Colab or other comparable cloud-based systems will be used if needed.

Reference Materials:

- Documentation of the CRISP-DM Methodology to direct the project stages.
- Textbooks and Online Resources: Any scholarly or research articles that aid in the comprehension of data mining, business intelligence, and customer classification models.

Risks, Assumptions and Constraints

Risks

Risks cannot be avoided for sure, however, the most likely risks involving the project can be determined and planned for. Proper risk mitigation strategies can be implemented to minimise the impact of the risks on the project outcome.

Risks can be classified based on the impact on the project. They are classified as low, medium or high risk. A risk matrix can be used to determine the impact of the risk using the likelihood and severity of the risk should it occur.

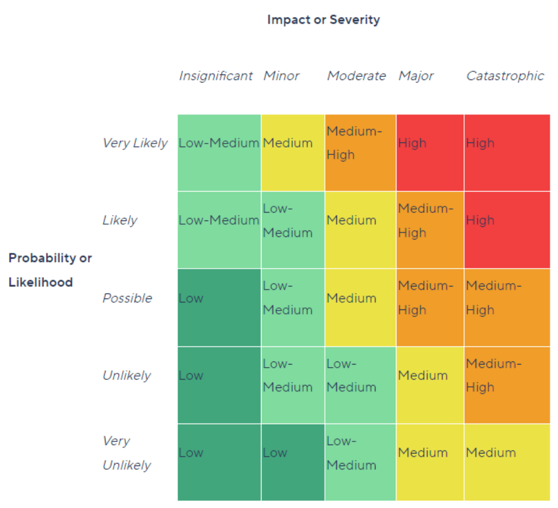
The likelihood of a risk occurring can be as follows:

- Very unlikely
- Unlikely
- Possible
- Likely
- Very likely

The severity of the risk on the project should they occur can be as follows:

- Insignificant
- Minor
- Moderate
- Major
- Catastrophic

Boogaard (2024), provides the following risk matrix:



Impact or Severity

Probability or Likelihood

	Insignificant	*Minor*	*Moderate*	*Major*	*Catastrophic*
Very Likely	Low-Medium	Medium	Medium-High	High	High
Likely	Low-Medium	Low-Medium	Medium	Medium-High	High
Possible	Low	Low-Medium	Medium	Medium-High	Medium-High
Unlikely	Low	Low-Medium	Low-Medium	Medium	Medium-High
Very Unlikely	Low	Low	Low-Medium	Medium	Medium

Figure 1 Risk Matrix

The risks of the project will mainly revolve around risks associated with the data analytics aspects of the project but is not limited to the data. The risks are as follows:

Table 1: Risk Analysis

Risk	Risk Description	Severity	Likelihood	Impact	Mitigation
Data Quality	Low data quality leads to a low-quality model	Major	Possible	Medium High Risk	Clean data thoroughly before modelling
Biased Data	Biased data leads to biased predictions	Major	Likely	Medium High Risk	Remove biased or dirty data
Overfitting	The model is trained to closely to the data and will not make accurate predictions on different data	Moderate	Possible	Medium Risk	Implement data balancing and remove noise
Data Privacy	The training data of a model can be reproduced	Catastrophic	Unlikely	Medium Risk	PAC framework can be implemented
Missing Deadline	Risk of delays causing missing deadlines	Moderate	Possible	Medium Risk	Proper time management and the use of dashboards and timelines

Constraints

The constraints in the project are the limitations that are set that the team will have to work within (monday.com, 2024).

The project will have the following constraints:

- The CRISP-DM framework must be followed.
- The supervisor must be consulted on all data mining goals before the team will be able to continue with subsequent milestones.
- R and Power BI are the tools that should be used to prepare the data and create the models. This could limit the data analysis in terms of the capabilities of the tools.
- The accuracy of the model may be limited if the dataset contains many missing values, or the quality of the data is low.
- If there are only a limited number of records in the dataset will mean that we are working with limited data on which the data may be trained. This could lead to less accurate results from the model.
- The time to complete each milestone is limited and not completing each milestone within the set timeframe may compromise the quality of the project.
- The development team is limited to the four members of the team and the knowledge of these members.
- The model will need to comply with ethical guidelines.
- Privacy limitations may limit the way the data can be stored and used in analysis.

Phase 2 – Data Understanding

Data Mining Goals and Success Criteria

Data mining goals help to create clear, measurable objectives that will be used to measure the success of the project. The project will have the following goals:

Primary Goal:

The main objective is to create a classification model that can recommend whether a customer is eligible for the satellite internet service. The company currently uses salary as the only criterion, but the model will incorporate other factors, such as job title, education level, and household size, to improve accuracy.

Specific Objectives are:

- **Improve Customer Selection**: To reduce the number of false positives (customers incorrectly predicted as eligible) and false negatives (eligible customers not recommended).
- **Feature Importance**: To determine which variables (besides salary) are the most important for predicting eligibility.
- **Baseline Comparison**: The aim is to outperform the model that currently only uses salary as the predictor.

Evaluation Metrics that should be met, include:

- **Accuracy**: The percentage of correct predictions. The accuracy of the model should be at be at least 85%, to be deemed acceptable. The higher the accuracy, the better.
- **Precision and Recall**: Important metrics if the business cares more about minimizing false positives (bad credit risks) or false negatives (missed opportunities).
- **F1-Score**: A harmonic mean of precision and recall, often used in classification problems where balancing false positives and false negatives is important.

Dataset Overview

The CustData2.csv dataset that will be used has the following attributes which will be used in the modelling:

Table 2: Attribute Analysis

Attribute	Description
Job_Title	Job titles can be a strong predictor of financial stability and service eligibility. This can be used to determine if there are groups of customers with certain jobs that are more eligible for services than others.
Department	Similar to job titles, certain departments (e.g., IT, Finance) might be more indicative of eligibility than others. This variable can help create broader segments.
Annual_Salary	This is the current criterion for eligibility. Although it is a strong predictor for eligibility, it should not be the sole predictor. The new model can be tested against the current model to test if the new model will be more accurate.
Gross.Pay.Last.Paycheck	Reflects the most recent income. Gain insight into the short-term financial situation of customers.
Gross.Year.To.Date	Reflects the total yearly earnings of customers. This is a good indicator of job stability.
Gross.Year.To.Date...FRS Contribution	Reflects the contribution of a customer to the retirement fund. This can be a reflection of customer loyalty.
Year_of_Birth	Year of Birth can be used to calculate the age of the customers. This will allow insights to be gathered on whether certain age groups have more financial responsibility and resources, which may impact other factors contributing to eligibility.
Marital_Status	Marital status might indicate different financial responsibilities, with married customers potentially having higher household expenses.
Country_id	Different countries may have varying economic conditions. Examining the area from which customers come, may allow for insight to be gained on the city's financial situation.
Education	Higher education levels often correlate with better-paying jobs and financial stability, which could impact eligibility.
Occupation	This provides further context for the Job_Title and Department attributes. Certain occupations might have lower financial risks. Additionally, customers in varying occupational field may have different incomes for the same job titles.
Household_Size	The household size may give an indication to how many members are financially contributing to the household.

Years_of_Residence	Length of residence might indicate stability or financial responsibility, with longer residence possibly correlating with lower credit risk.

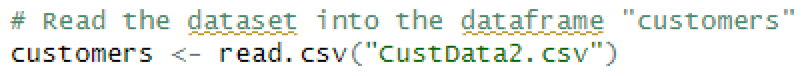
Data Types

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 Quality Assessment

The quality of the data plays a very important role in creating an accurate intelligent recommender model. The GIGO (garbage in, garbage out) concept supports this by stating that in any system, the quality of the input determines the quality of the output (Awati, 2023). Therefore, the quality of the dataset must be assessed for quality problems, missing values, duplicates, outliers and any other relevant metrics. Any identified quality concerns must be mitigated in the third CRISP-DM phase (Data Preparation).

RStudio will be used to analyse the dataset. To import the dataset into RStudio the following code needs to be executed:

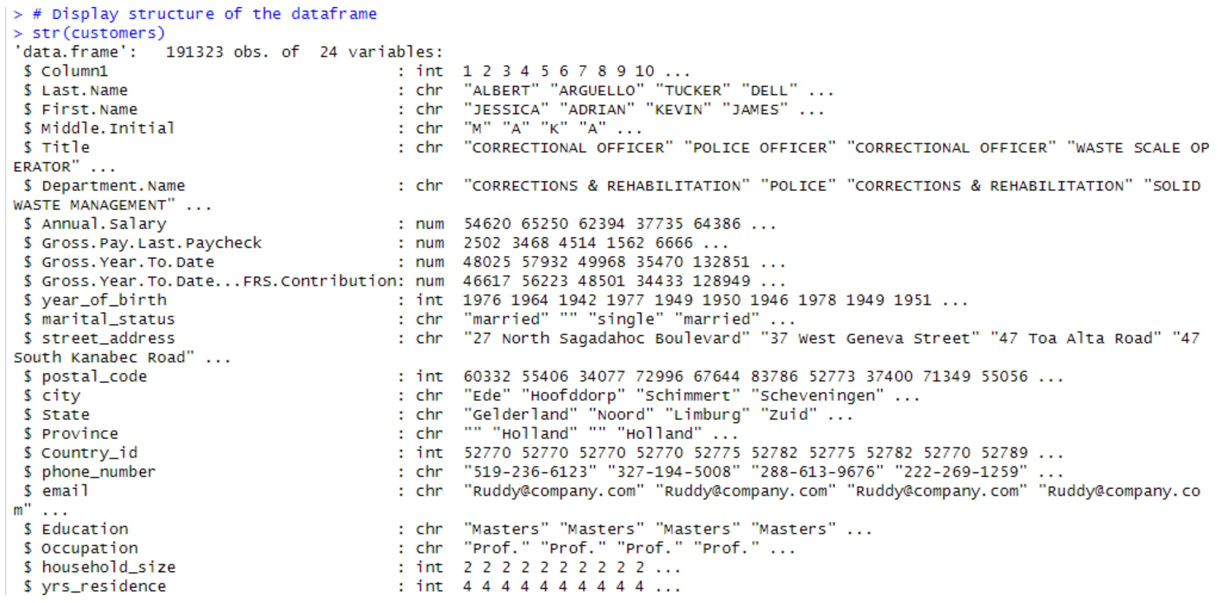


```
# Read the dataset into the dataframe "customers"
customers <- read.csv("CustData2.csv")
```

Figure 2 Read the dataset into a data frame

Dataset Structure

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



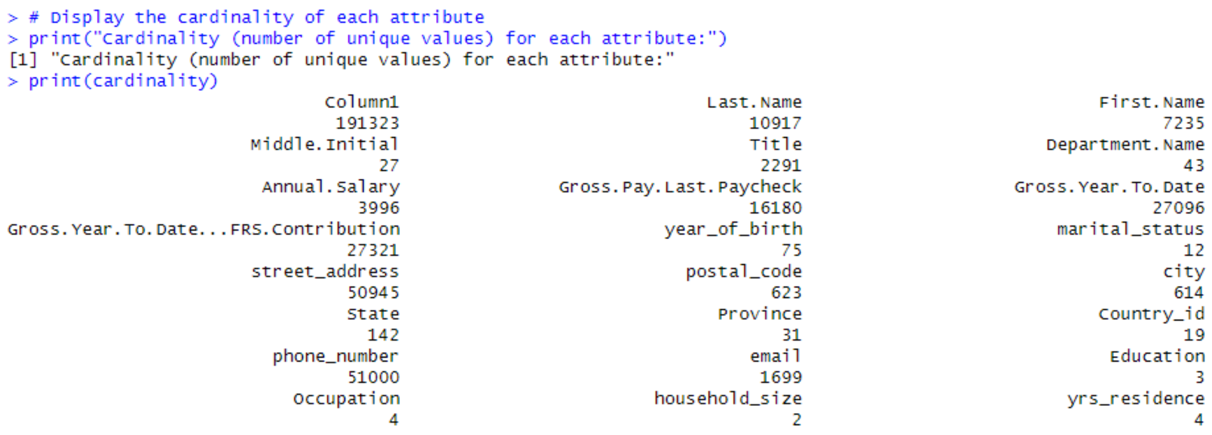
```
> # Display structure of the dataframe
> str(customers)
'data.frame':	191323 obs. of  24 variables:
 $ Column1                          : int  1 2 3 4 5 6 7 8 9 10 ...
 $ Last.Name                        : chr  "ALBERT" "ARGUELLO" "TUCKER" "DELL" ...
 $ First.Name                       : chr  "JESSICA" "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 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 3 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).

Cardinality

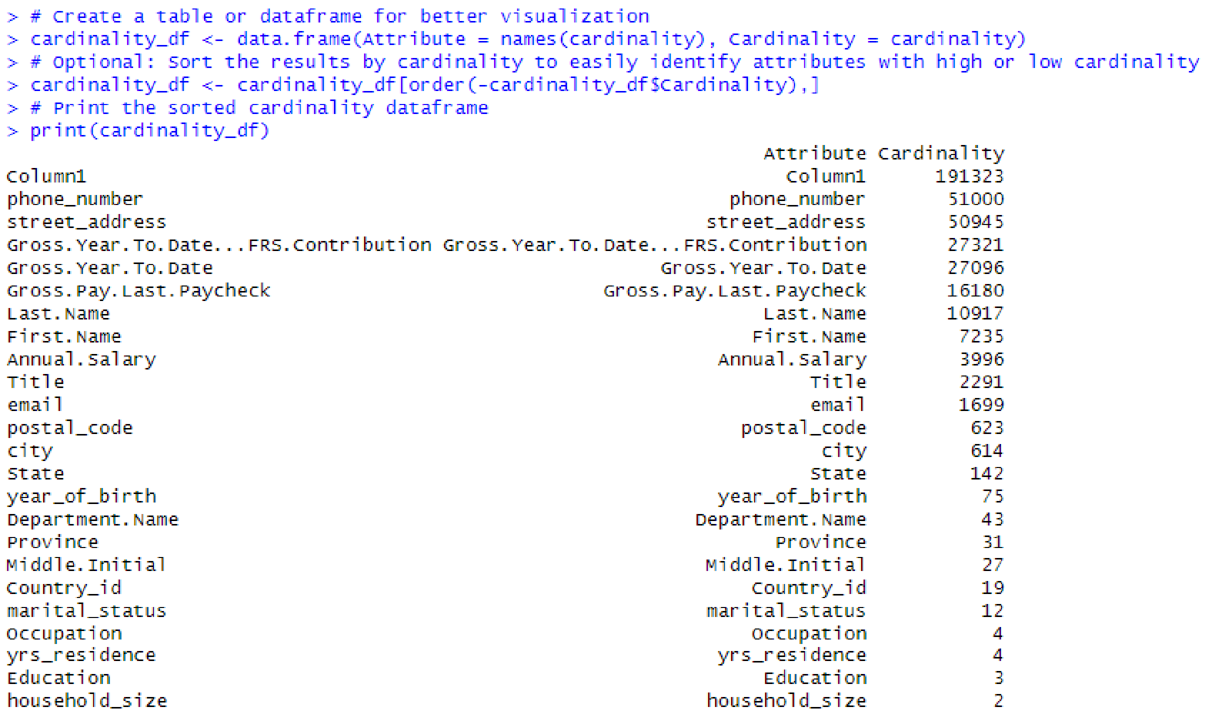
```
# Create a function to calculate the cardinality (number of unique values)
calculate_cardinality <- function(df) {

cardinalities <- sapply(df, function(x) length(unique(x)))

return(cardinalities)

}

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

# 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

```
# 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):

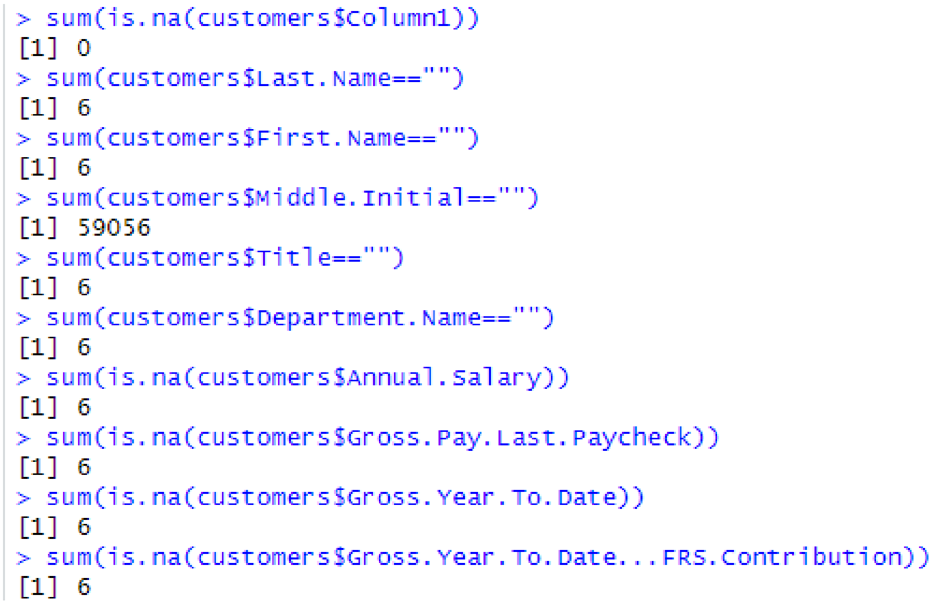
- **Analysis**: These attributes have low cardinality, indicating they contain fewer distinct categories. Low cardinality features are often useful for classification and segmentation. For example:
    i. **Education** and **Occupation** can be critical factors in assessing a customer's financial stability.
    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.

Handling Missing Values

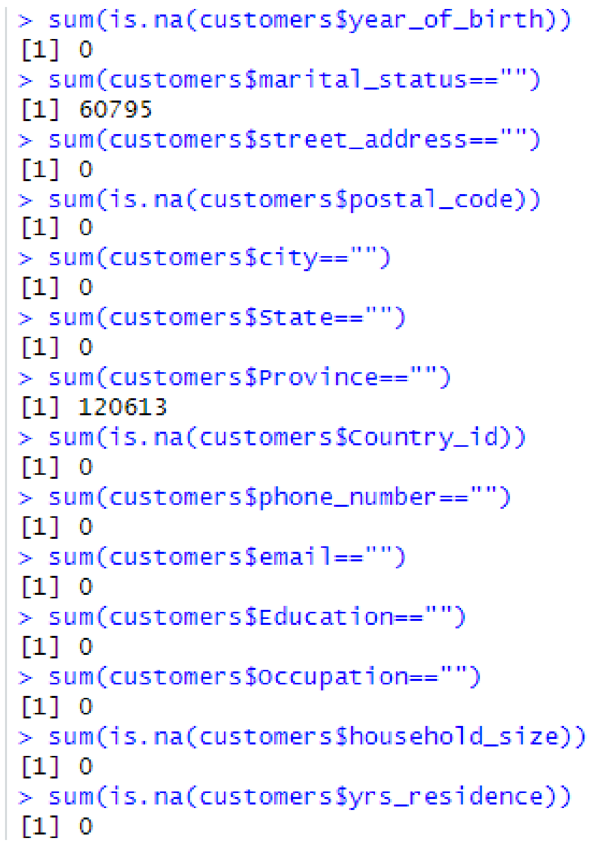
This is done by determining whether entries in the dataset is empty/missing or N/A, as these could have an impact on the analysis' findings. Depending on the amount of missing data and its significance for the analysis, missing values are either imputed (using the mean, median, or mode) or deleted, this will only be done in the next phase of the CRISP-DM lifecycle as stated above.

Figures below contains code for every attribute that counts the number of empty cells and their respective output.



```
> 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
```

Figure 6 Missing values for each attribute – Part 1



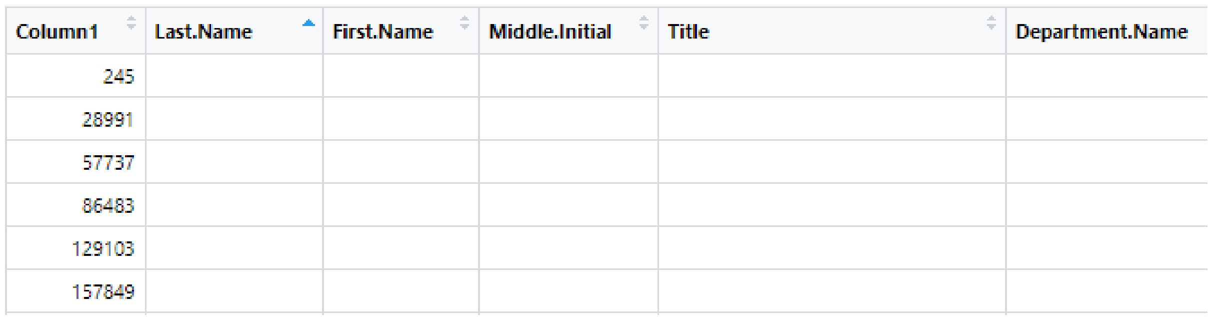
```
> sum(is.na(customers$year_of_birth))
[1] 0
> sum(customers$marital_status=="")
[1] 60795
> sum(customers$street_address=="")
[1] 0
> sum(is.na(customers$postal_code))
[1] 0
> sum(customers$city=="")
[1] 0
> sum(customers$State=="")
[1] 0
> sum(customers$Province=="")
[1] 120613
> sum(is.na(customers$Country_id))
[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 7 Missing values for each attribute – Part 2

The following was observed:

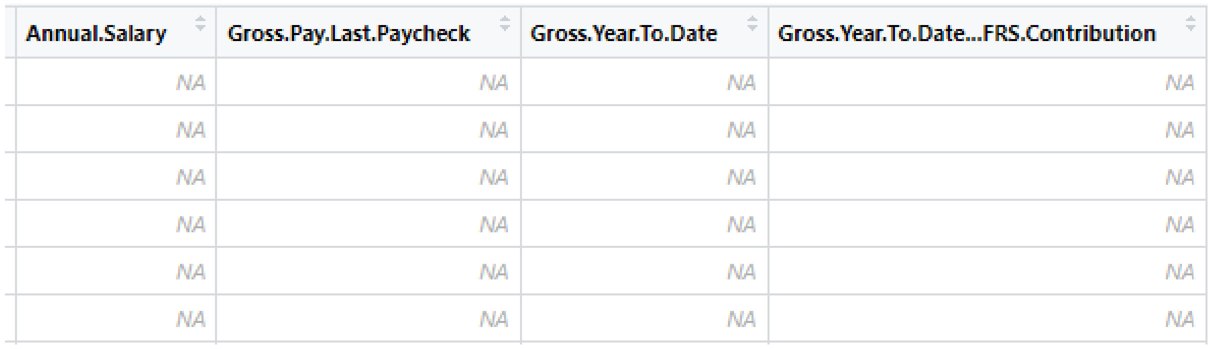
- There are three attributes that are missing a large number of entries:
  - Middle.Initial: It is not abnormal for this attribute to be empty, seeing as not every person has a middle name.
  - Marital_status: All the cells for this attribute must contain an entry, such as “Married” or “Single”. This shows weak data quality, seeing as this attribute can be used to train the model.
  - Province: All the cells for this attribute must contain an entry. This also shows weak data quality, seeing as this attribute can be used to train the model.
- There are numerous attributes that are only missing six entries. This is not a dangerous number of entries, but it still needs to be investigated. The following was discovered:

The six missing entries for each attribute, is missing for the same 6 records. This could have been caused by corruption during data extraction or data entry issues.



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

Figure 8 Six records missing the same attributes - 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 9 Six records missing the same attributes - Part 2

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.

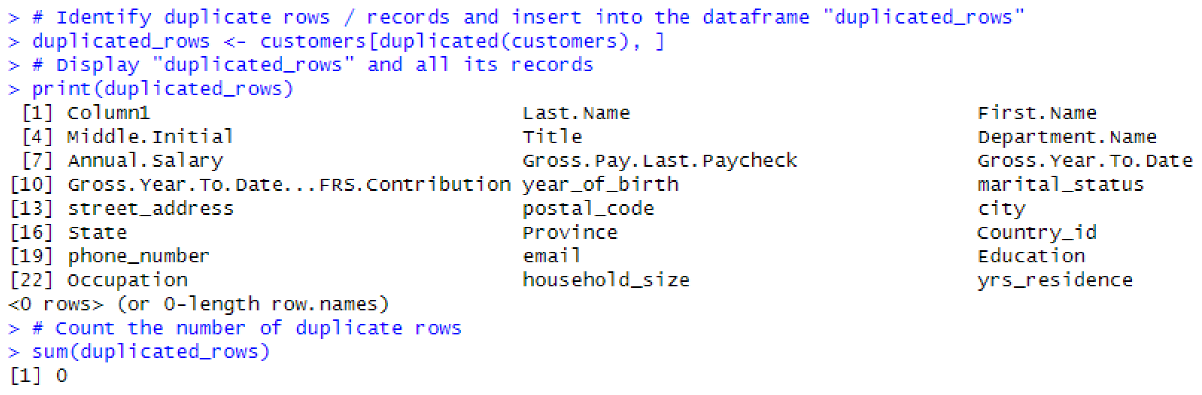
- **Exclusion**: If the high rate of missing data makes it unreliable or non-essential for analysis, consider excluding the attribute from certain aspects of the model.

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.

Duplicate Records

Duplicate records exist when a record/row appears more than once in a dataset. This is redundant and unnecessarily uses more storage. Any duplicate records are identified using the following R code:



```
> # Identify duplicate rows / records and insert into the dataframe "duplicated_rows"
> duplicated_rows <- customers[duplicated(customers), ]
> # Display "duplicated_rows" and all its records
> print(duplicated_rows)
 [1] Column1                          Last.Name                        First.Name
 [4] Middle.Initial                   Title                            Department.Name
 [7] Annual.Salary                    Gross.Pay.Last.Paycheck          Gross.Year.To.Date
[10] Gross.Year.To.Date...FRS.Contribution year_of_birth               marital_status
[13] street_address                   postal_code                      city
[16] State                            Province                         Country_id
[19] phone_number                     email                            Education
[22] Occupation                       household_size                   yrs_residence
<0 rows> (or 0-length row.names)
> # Count the number of duplicate rows
> sum(duplicated_rows)
[1] 0
```

Figure 10 Output of the duplication check

The figure above shows the output that is generated when the code is run. As can be seen, there are no duplicate records in this dataset. Thus, no further steps are required regarding duplicate records.

Outliers

Box plots and other visualizations are used to identify outliers in the dataset. The figure below shows a boxplot for the column/attribute "Annual.Salary". This boxplot was created using the following R code:

```
# Outliers shown in boxplot of the "Annual.Salary"
boxplot(customers$Annual.Salary)
```

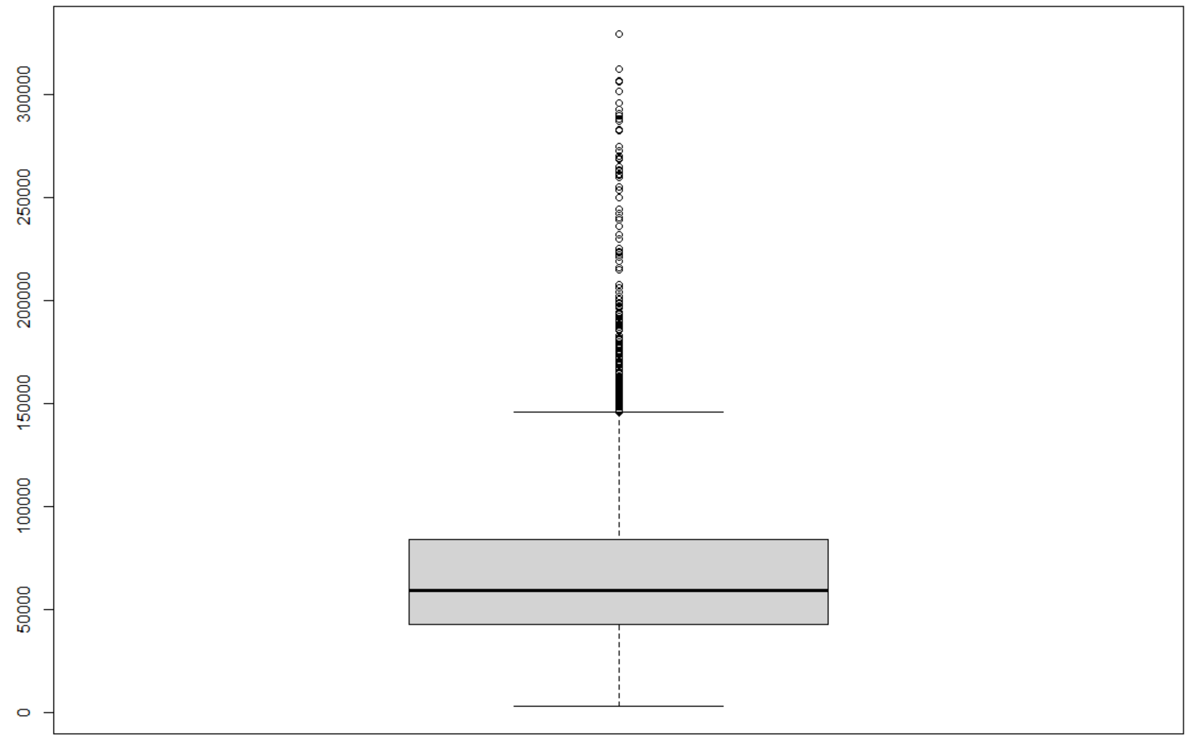


Figure 11 Boxplot of Annual.Salary

It can be stated with confidence that there are several outliers within the "Annual.Salary" column of this dataset (shown by the dots outside the boxplot).

After outliers are located, the subsequent actions need to be performed:

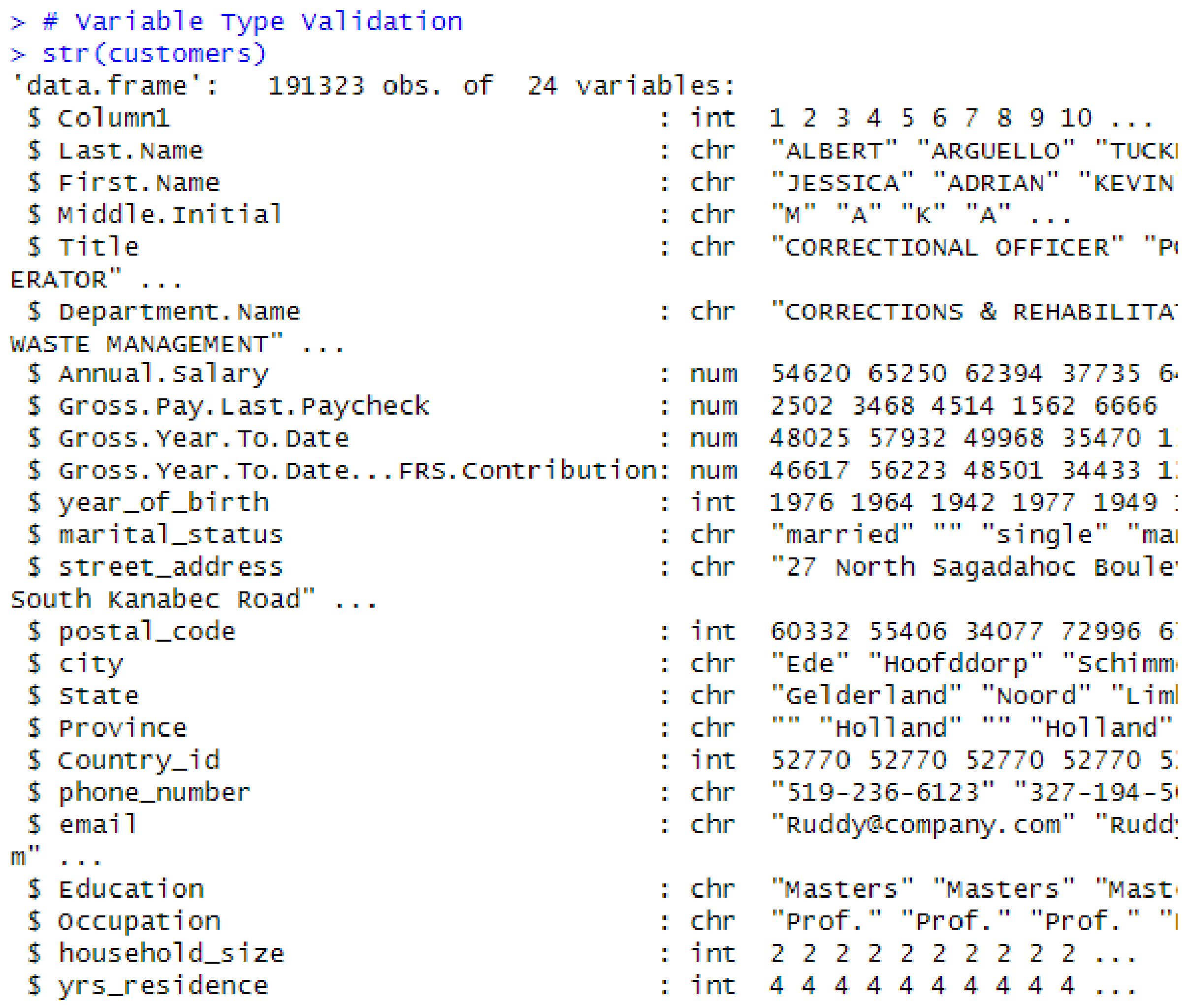
- The dataset is cleaned up of outliers that were obviously the product of incorrect data entry.
- Valid extreme outliers were handled differently, recognizing their possible significance for the analysis.

Variable Type Validation

Verify the proper formatting of continuous and categorical variables. To do this, the following command was used:

```
# Variable Type Validation
str(customers)
```

The figure below shows the output generated by the command:



```
> # Variable Type Validation
> 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" "TUCK
 $ First.Name                        : chr  "JESSICA" "ADRIAN" "KEVIN
 $ Middle.Initial                    : chr  "M" "A" "K" "A" ...
 $ Title                             : chr  "CORRECTIONAL OFFICER" "P
ERATOR" ...
 $ Department.Name                   : chr  "CORRECTIONS & REHABILITA
WASTE MANAGEMENT" ...
 $ Annual.Salary                     : num  54620 65250 62394 37735 6
 $ Gross.Pay.Last.Paycheck           : num  2502 3468 4514 1562 6666
 $ Gross.Year.To.Date                : num  48025 57932 49968 35470 1
 $ Gross.Year.To.Date...FRS.Contribution: num  46617 56223 48501 34433 1
 $ year_of_birth                     : int  1976 1964 1942 1977 1949
 $ marital_status                    : chr  "married" "" "single" "ma
 $ street_address                    : chr  "27 North Sagadahoc Boule
South Kanabec Road" ...
 $ postal_code                       : int  60332 55406 34077 72996 6
 $ city                              : chr  "Ede" "Hoofddorp" "Schimm
 $ State                             : chr  "Gelderland" "Noord" "Lim
 $ Province                          : chr  "" "Holland" "" "Holland"
 $ Country_id                        : int  52770 52770 52770 52770 5
 $ phone_number                      : chr  "519-236-6123" "327-194-5
 $ email                             : chr  "Ruddy@company.com" "Rudd
m" ...
 $ Education                         : chr  "Masters" "Masters" "Mast
 $ Occupation                        : chr  "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 12 Output confirming variable type validation

Analysis of the output given in the figure above shows that every attribute of the dataset is read/stored as the correct/appropriate data type. This states that no changes to attribute types need to be made.

Data Distribution Analysis

To find any skewness or non-normal distributions, the distribution of numerical variables (in this case, "Annual.Salary") was evaluated. If required, transformations (such as the logarithmic transformation) must be used to standardize the data. The following R code was used to plot the graph in the figure below:

```
# Data Distribution
hist(customers$Annual.Salary)
```

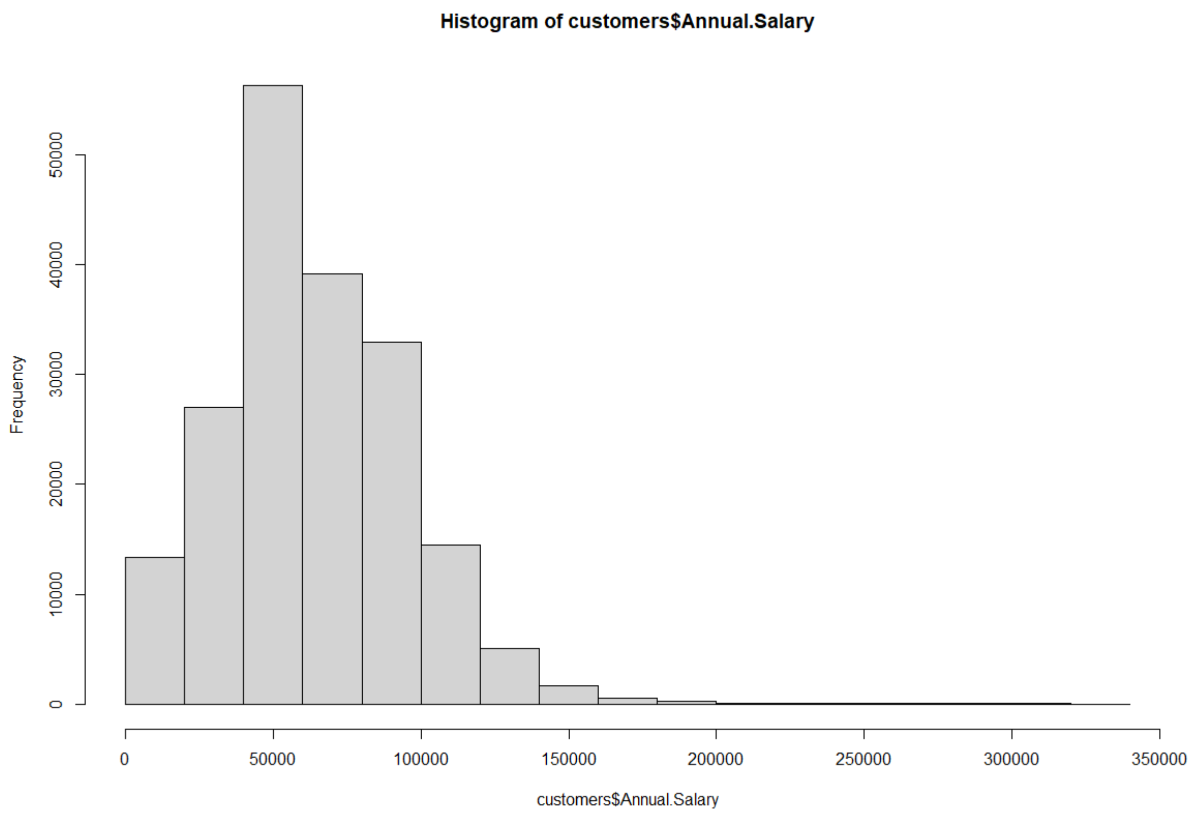
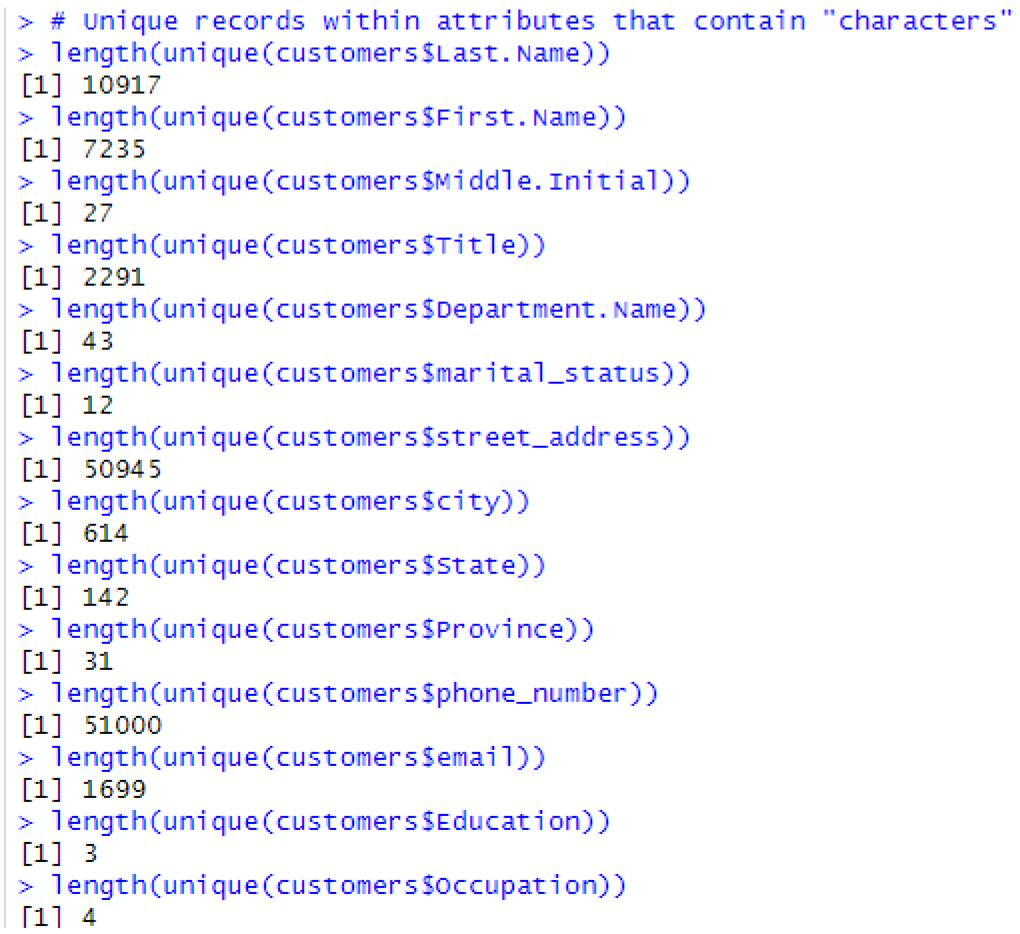


Figure 13 Histogram displaying the distribution of salary

The figure above shows a normal distribution of the salary data.

Unique values per attribute/column

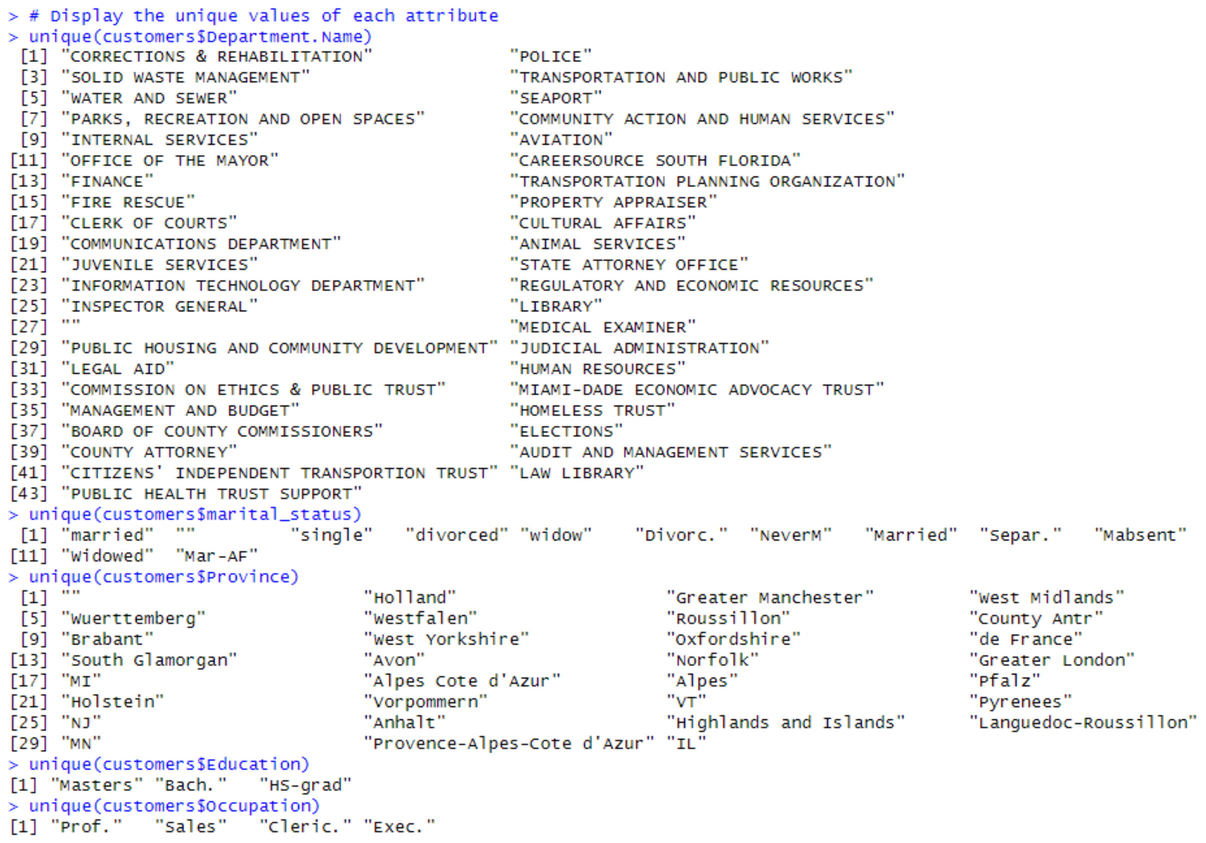
The quality of the data is also determined by assessing the values contained in attributes that store/contain values of type “character” or “string”. The figures below contains the code used to count the number of unique values within each attribute, along with the output:



```
> # Unique records within attributes that contain "characters"
> length(unique(customers$Last.Name))
[1] 10917
> length(unique(customers$First.Name))
[1] 7235
> length(unique(customers$Middle.Initial))
[1] 27
> length(unique(customers$Title))
[1] 2291
> length(unique(customers$Department.Name))
[1] 43
> length(unique(customers$marital_status))
[1] 12
> length(unique(customers$street_address))
[1] 50945
> length(unique(customers$city))
[1] 614
> length(unique(customers$State))
[1] 142
> length(unique(customers$Province))
[1] 31
> length(unique(customers$phone_number))
[1] 51000
> length(unique(customers$email))
[1] 1699
> length(unique(customers$Education))
[1] 3
> length(unique(customers$Occupation))
[1] 4
```

Figure 14 Unique values of "character" attributes

All the attributes that contain a significant number of unique values (more than 100) will be disregarded, seeing as this shows a high cardinality. It can be concluded that these attributes will not play a significant role in the training/creation of the model. Attributes such as name and surname should have high cardinality seeing as people have different names. “Middle.Initial” will also be discarded, it contains all the letters contained in the alphabet (That is why there are 27 unique values). Further analysis is conducted on the attributes with low cardinality. The following was discovered:



```
> # Display the unique values of each attribute
> unique(customers$Department.Name)
 [1] "CORRECTIONS & REHABILITATION"                 "POLICE"
 [3] "SOLID WASTE MANAGEMENT"                       "TRANSPORTATION AND PUBLIC WORKS"
 [5] "WATER AND SEWER"                              "SEAPORT"
 [7] "PARKS, RECREATION AND OPEN SPACES"            "COMMUNITY ACTION AND HUMAN SERVICES"
 [9] "INTERNAL SERVICES"                            "AVIATION"
[11] "OFFICE OF THE MAYOR"                          "CAREERSOURCE SOUTH FLORIDA"
[13] "FINANCE"                                      "TRANSPORTATION PLANNING ORGANIZATION"
[15] "FIRE RESCUE"                                  "PROPERTY APPRAISER"
[17] "CLERK OF COURTS"                              "CULTURAL AFFAIRS"
[19] "COMMUNICATIONS DEPARTMENT"                    "ANIMAL SERVICES"
[21] "JUVENILE SERVICES"                            "STATE ATTORNEY OFFICE"
[23] "INFORMATION TECHNOLOGY DEPARTMENT"            "REGULATORY AND ECONOMIC RESOURCES"
[25] "INSPECTOR GENERAL"                            "LIBRARY"
[27] ""                                             "MEDICAL EXAMINER"
[29] "PUBLIC HOUSING AND COMMUNITY DEVELOPMENT"     "JUDICIAL ADMINISTRATION"
[31] "LEGAL AID"                                    "HUMAN RESOURCES"
[33] "COMMISSION ON ETHICS & PUBLIC TRUST"          "MIAMI-DADE ECONOMIC ADVOCACY TRUST"
[35] "MANAGEMENT AND BUDGET"                        "HOMELESS TRUST"
[37] "BOARD OF COUNTY COMMISSIONERS"                "ELECTIONS"
[39] "COUNTY ATTORNEY"                              "AUDIT AND MANAGEMENT SERVICES"
[41] "CITIZENS' INDEPENDENT TRANSPORTION TRUST"     "LAW LIBRARY"
[43] "PUBLIC HEALTH TRUST SUPPORT"
> unique(customers$marital_status)
 [1] "married"  ""         "single"   "divorced" "widow"    "Divorc."  "NeverM"   "Married"  "Separ."   "Mabsent"
[11] "Widowed"  "Mar-AF"
> unique(customers$Province)
 [1] ""                          "Holland"                   "Greater Manchester"        "West Midlands"
 [5] "Wuerttemberg"              "Westfalen"                 "Roussillon"                "County Antr"
 [9] "Brabant"                   "West Yorkshire"            "Oxfordshire"               "de France"
[13] "South Glamorgan"           "Avon"                      "Norfolk"                   "Greater London"
[17] "MI"                        "Alpes Cote d'Azur"         "Alpes"                     "Pfalz"
[21] "Holstein"                  "Vorpommern"                "VT"                        "Pyrenees"
[25] "NJ"                        "Anhalt"                    "Highlands and Islands"     "Languedoc-Roussillon"
[29] "MN"                        "Provence-Alpes-Cote d'Azur" "IL"
> unique(customers$Education)
[1] "Masters" "Bach."   "HS-grad"
> unique(customers$Occupation)
[1] "Prof."   "Sales"   "Cleric." "Exec."
```

Figure 15 Unique records for low cardinality attributes

“Department.Name”, “Province”, “Education” and “Occupation” contain different values that are not related in any way; therefore, these attributes do not show any issues. “Marital_status” contains many different phrases that mean the same things. These phrases need to be transformed/normalized so that only one phrase is used for people that fall under the following categories: “Married”, “Single”, “Widowed” and “Divorced”.

Data Quality Conclusion

After completion of the data quality assessments, it can be concluded that there are many quality problems within this dataset. Data preparation steps must be taken, to create an accurate intelligent recommender model.

Preliminary Data Visualization

All of the following visualisations have been done using RStudio.

1. Correlation Matrix:

The first, and arguably most important data visualization that can be done on the dataset is to create a correlation heatmap. This heatmap will assist in finding strong correlations between attributes.

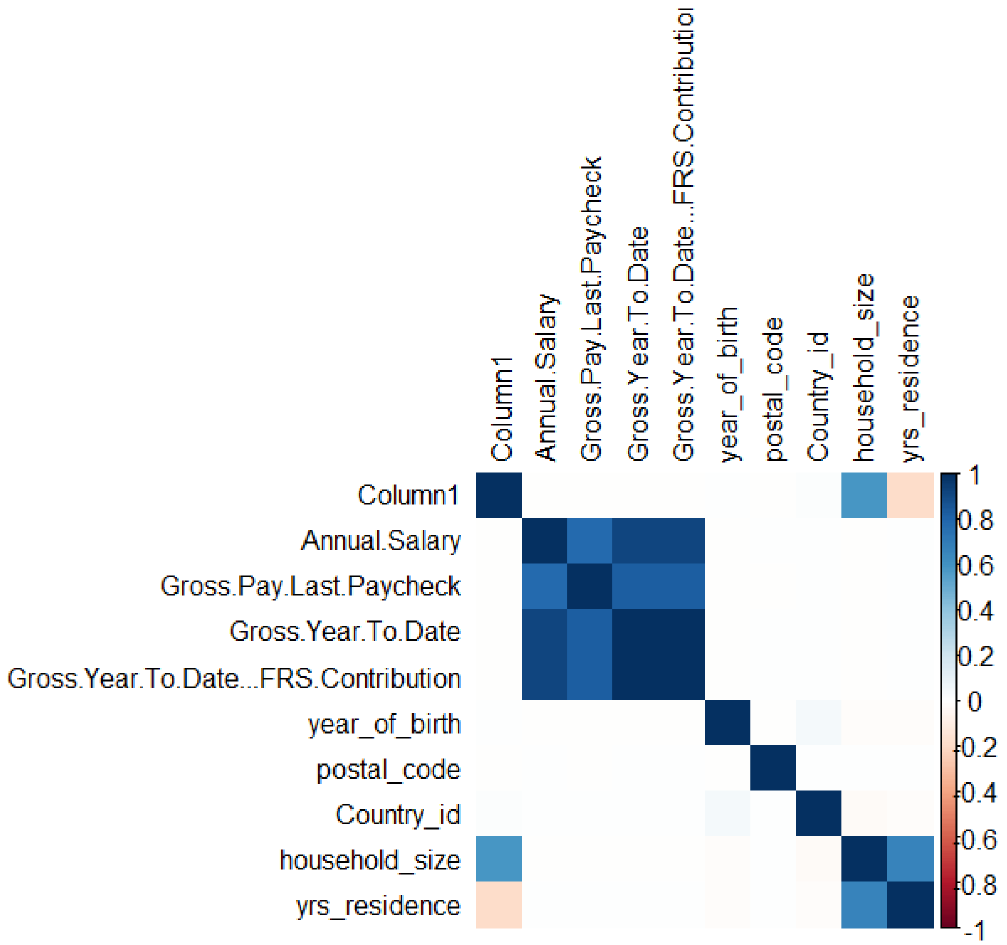


Figure 16 Correlation Matrix

From the figure above, it can be observed that there is a very strong correlation between the Groos.Year.To.Date...FRS.Contribution attribute and the Gross.Year.To.Date attribute. Furthermore, there is also a strong correlation between the Annual.Salary, Gross.Pay.Last.Paycheck, and Gross.Year.To.Date.

There is also a correlation between the household_size and yrs_residence attributes.

2. Pair Plots

The correlations between attributes can also be visualised using pair plots. The pair plots will visualise the relationships between pairs of attributes. The following pair plots have been created.

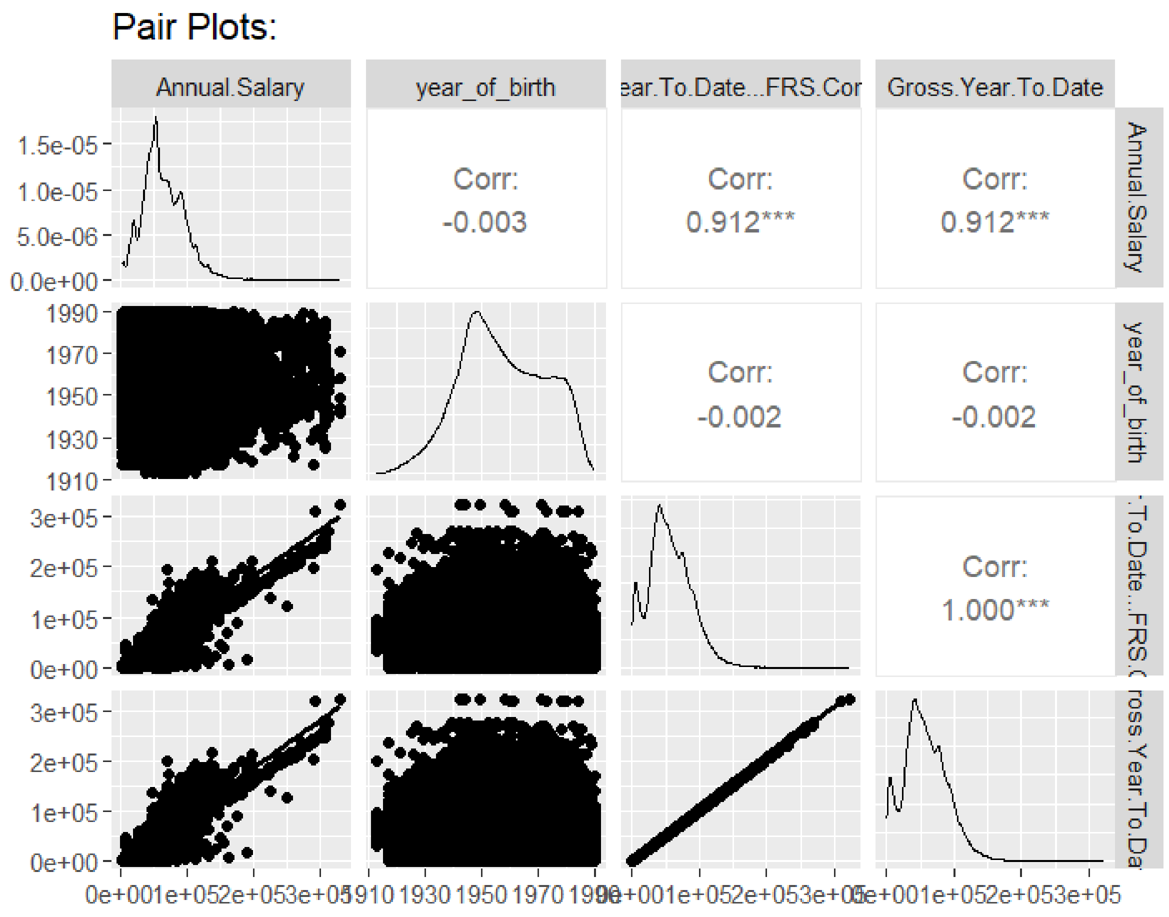


Figure 17 Pair Plots between attributes

The first relationship to observe is that between Annual Salary and Year of Birth. Although there is a correlation between this pair it is very weak. The correlation is also slightly negative, which suggests that as the Year of Birth increases, the salary also decreases. However, as mentioned there is not a strong relationship between these attributes.

The next relationship is that between Annual Salary and Gross Year To Data (FRS Constribution). There is a very strong positive correlation between these attributes. This means that as the Annual Salary increases, so does the Gross Year To Data (FRS Constribution).

The relationship between Annual Salary and Gross Year To Date is observed next. There is a strong correlation between these attributes as seen in the upward slope of the scatter plot. This means that as the Annual Salary increases, so does the Gross Year To Date.

Next, the correlation between the Year of Birth and the Gross Year To Date (FRS Contibution) is examined. There is a weak correlation between the attributes. Furthermore, the correlation is slightly negative. This suggests that as the Year of Birth increases, the Gross Year to Date (FRS Contribution) decreases very slightly.

The correlation between the Year of Birth and Gross Year to Date has a similar correlation as the previous pair. There is a weak negative correlation, and this suggests that as the Year of Birth increases, the Gross Year to Date decreases slightly.

Finally, the correlation between the Gross Year to Date and the Gross Year to Date (FRS Contribution) is examined. The correlation between these attributes is perfect. This is seen in the almost perfect diagonal line of the scatter plot as well as the correlation coefficient of 1. This means that either attribute can be 100% accurately detected using the other.

3. Boxplot for Annual Salary by Department

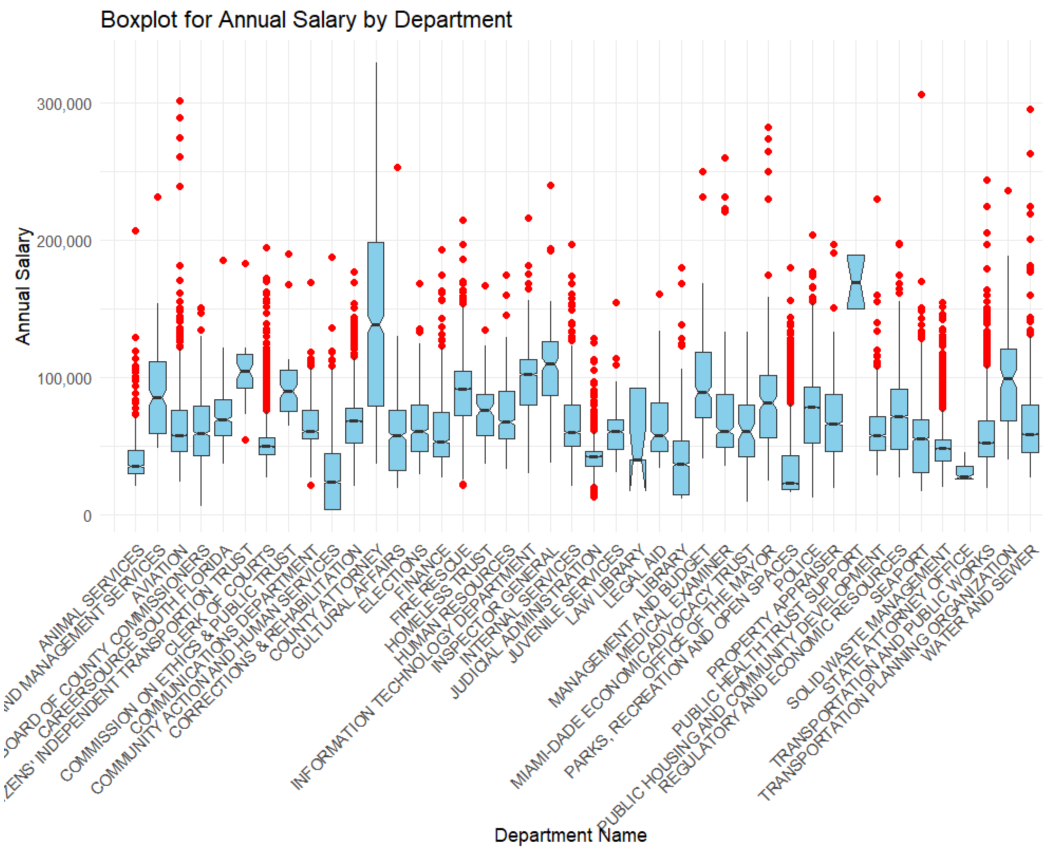


Figure 18 Boxplot for Annual Salary by Department

This graph shows the distribution of the annual salaries across different departments. As can be seen in the figure, most customers have lower salaries with their being a few outliers with higher salaries. The median salaries belonging to different departments also vary, with some having much higher median salaries then others.

4. Scatter Plot for Relationship between Annual Salary and Gross Pay Last Paycheck

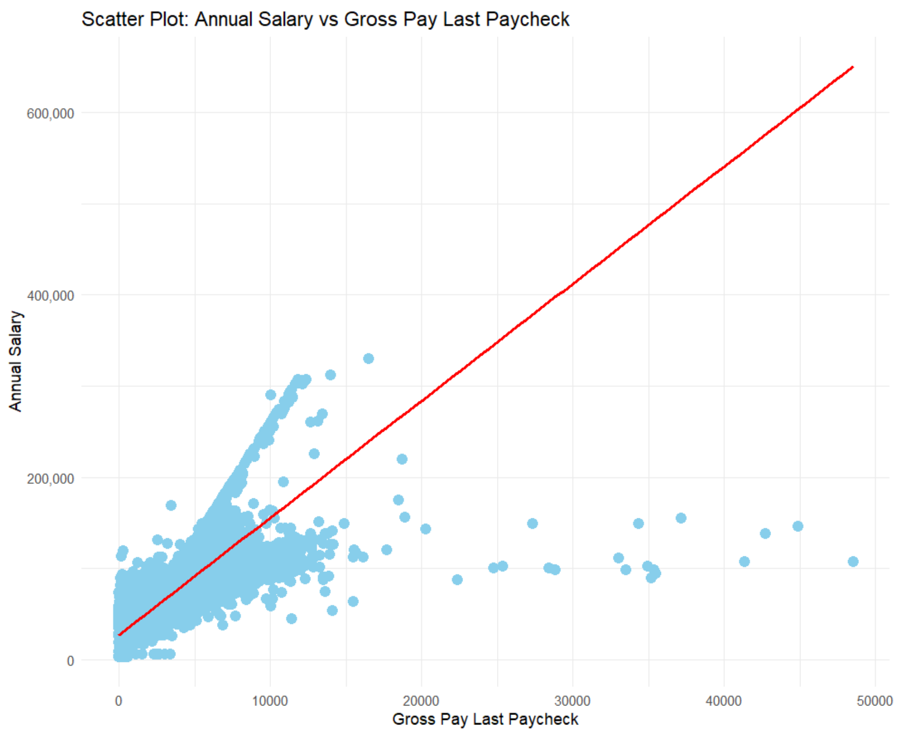


Figure 19 Scatter Plot for Annual Salary vs Gross Pay Last Paycheck

The scatter plot in Figure 14 shows the correlation between the Annual Salary and Gross Pay Last Paycheck. The upward slope of the scatter plot suggests a string positive correlation between the attributes. This means that as the Annual Salary increases, so does the Gross Pay Last Paycheck.

Although the scatter plot is mostly linear, there are outliers that can be observed. This will give un insight into individuals who have unusual salary patterns.

5. Histogram for Distribution of Year of Birth

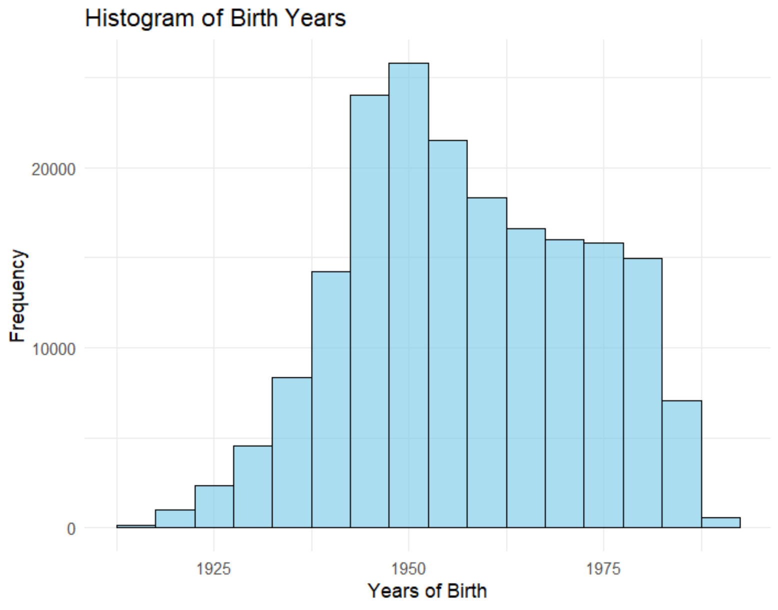


Figure 20 Histogram for Distribution of Year of Birth

The Histogram above the distribution of customers across the Year of Birth of the customers.

The first observation that can be made, is that the histogram is skewed to the right. This means that there are more customers born in recent years.

Furthermore, it can be seen that most of the current customers were born around 1950. Although the number of customers born after 1950 have declined, the decline is not major and there is still a large number of the customers that have been born from the 1950s to the 1980s.

Project Team Organisation

Table 3: Project team roles and responsibilities

Person	Roles and responsibilities
Jo-Anne van der Wath	• Data Aggregation • Data Transformation • Data Modelling • Deployment (Shiny App)
Henry Roux	• Data Quality Assessment • Data cleaning • Pipeline • Data Preparation (Splitting)
Armandre Erasmus	• Business Problem • Business Objectives • Dashboard Design • Preliminary Visualisations
Chaleigh Storm	• Ethical and business considerations • Model Analysis • Data quality • Data understanding

Collaboration

All group members contributed to documentation through Microsoft Teams.

The code files were uploaded to a shared GitHub repository. The repository can be found at:

https://github.com/HXCLogic/Business-Intelligence-381-Project

Weekly sprints were done and regular meetings were held, which all members attended.

A Kanban board was used to monitor the progress of team members for their tasks.

This project was completed through rigorous teamwork and collaboration.

Phase 3 – Data Preparation

Data Cleaning

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).

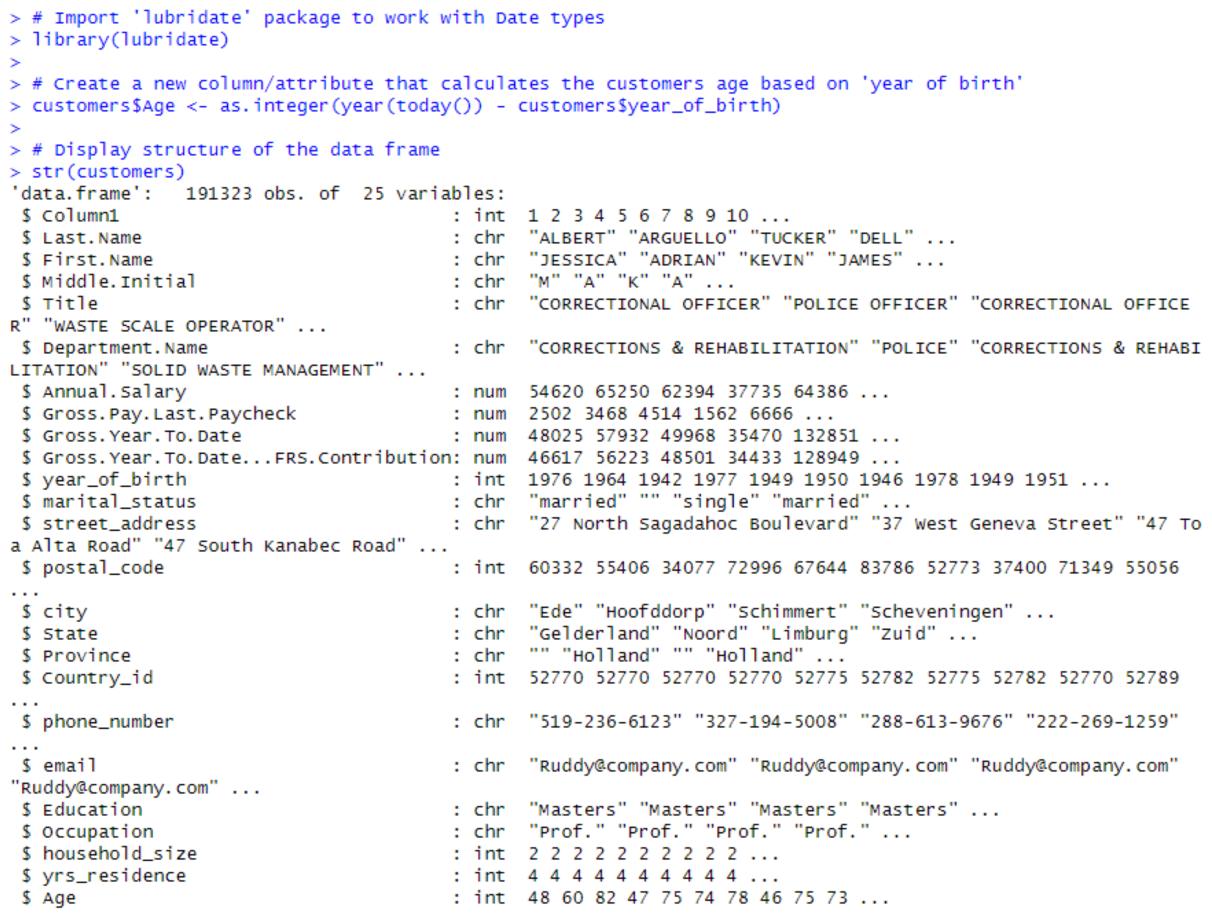
The following data cleaning streps need to be taken to clean and prepare this dataset:

- 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.

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)
```

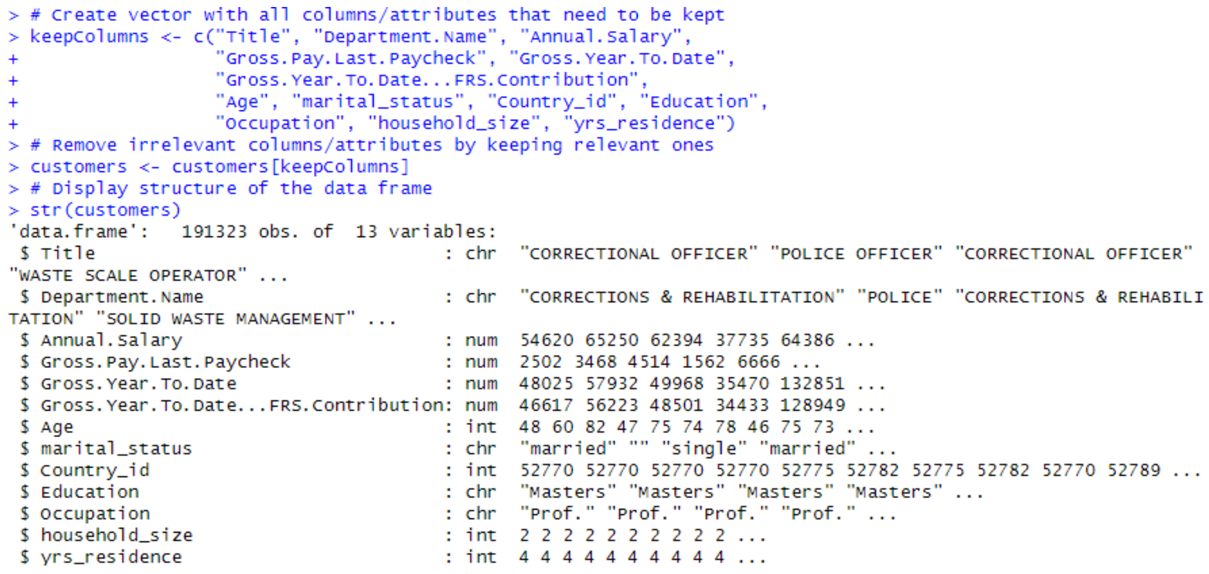


```
> # Import 'lubridate' package to work with Date types
> library(lubridate)
> 
> # Create a new column/attribute that calculates the customers age based on 'year of birth'
> customers$Age <- as.integer(year(today()) - customers$year_of_birth)
> 
> # Display structure of the data frame
> str(customers)
'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 21 Age Attribute Added to Dataset

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)
```



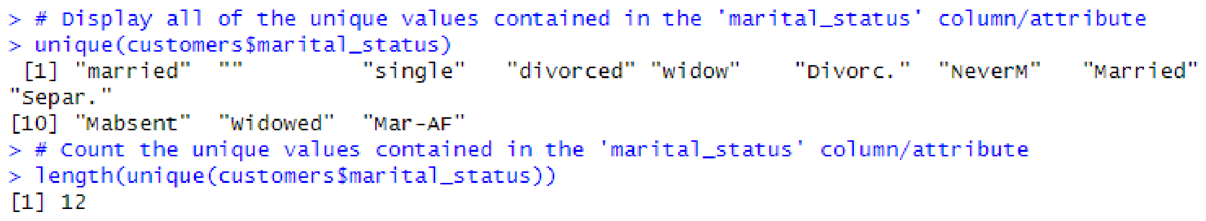
```
> # Create vector with all columns/attributes that need to be kept
> keepColumns <- c("Title", "Department.Name", "Annual.Salary",
+                  "Gross.Pay.Last.Paycheck", "Gross.Year.To.Date",
+                  "Gross.Year.To.Date...FRS.Contribution",
+                  "Age", "marital_status", "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 & 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 ...
 $ Age                                  : int  48 60 82 47 75 74 78 46 75 73 ...
 $ marital_status                       : chr  "married" "" "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 22 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))
```



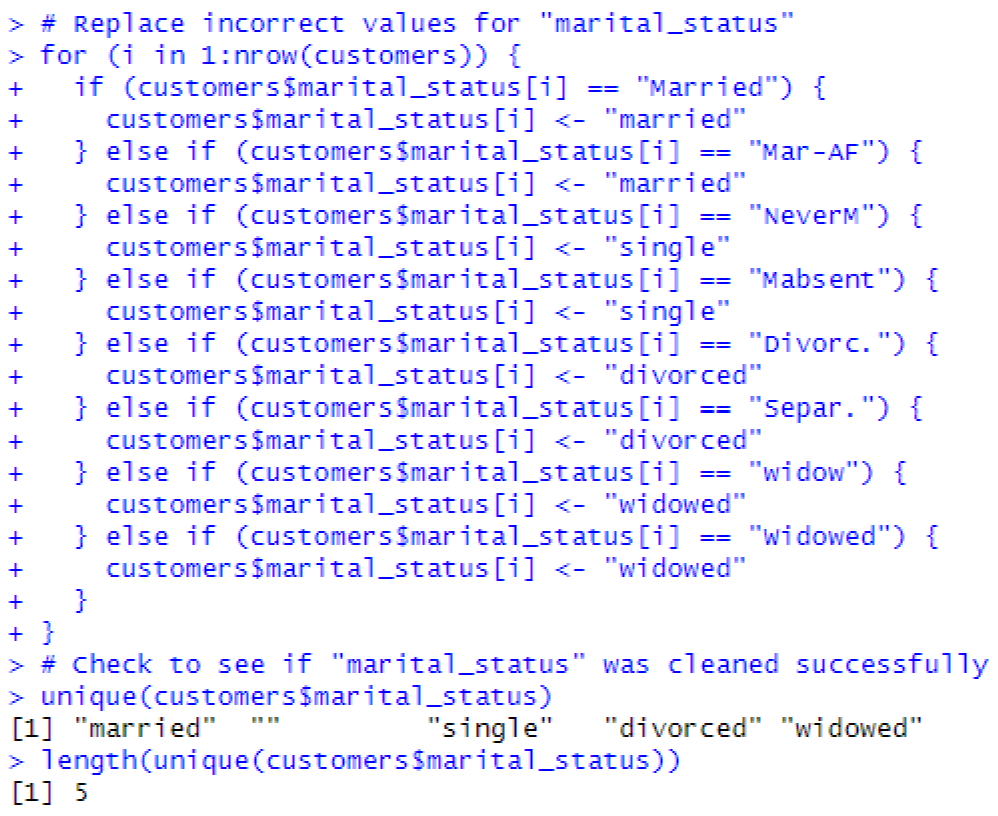
```
> # Display all of the unique values contained in the 'marital_status' column/attribute
> unique(customers$marital_status)
 [1] "married"  ""         "single"   "divorced" "widow"    "Divorc."  "NeverM"   "Married"
"Separ."
[10] "Mabsent"  "Widowed"  "Mar-AF"
> # Count the unique values contained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

Figure 23 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”.

This will be achieved through the following code:



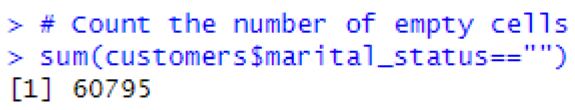
```
> # Replace incorrect values for "marital_status"
> for (i in 1:nrow(customers)) {
+   if (customers$marital_status[i] == "Married") {
+     customers$marital_status[i] <- "married"
+   } else if (customers$marital_status[i] == "Mar-AF") {
+     customers$marital_status[i] <- "married"
+   } else 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$marital_status)
[1] "married"  ""         "single"   "divorced" "widowed"
> length(unique(customers$marital_status))
[1] 5
```

Figure 24 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”

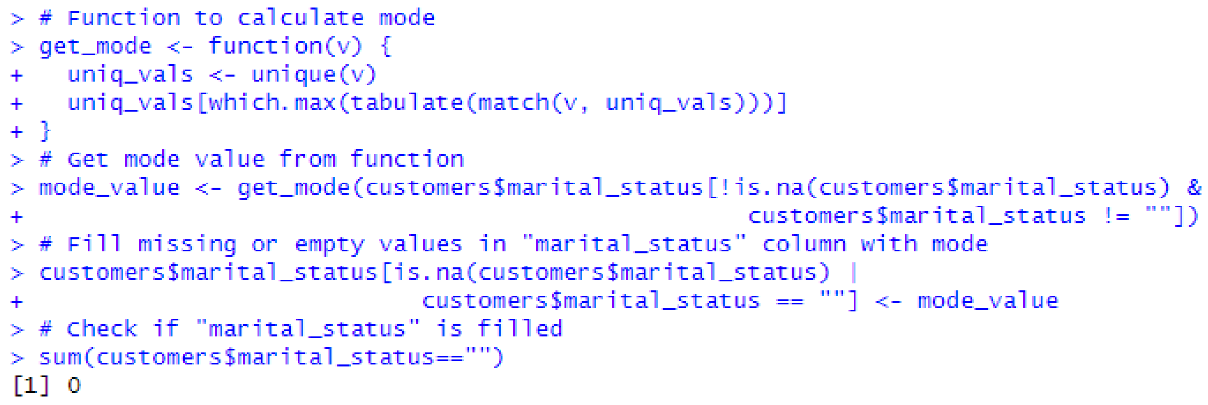
The next step to clean the attribute “marital_status” is to populate empty values/cells in the attribute. Firstly, check how many empty values/cells are there:



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

Figure 25 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=="")
[1] 0
```

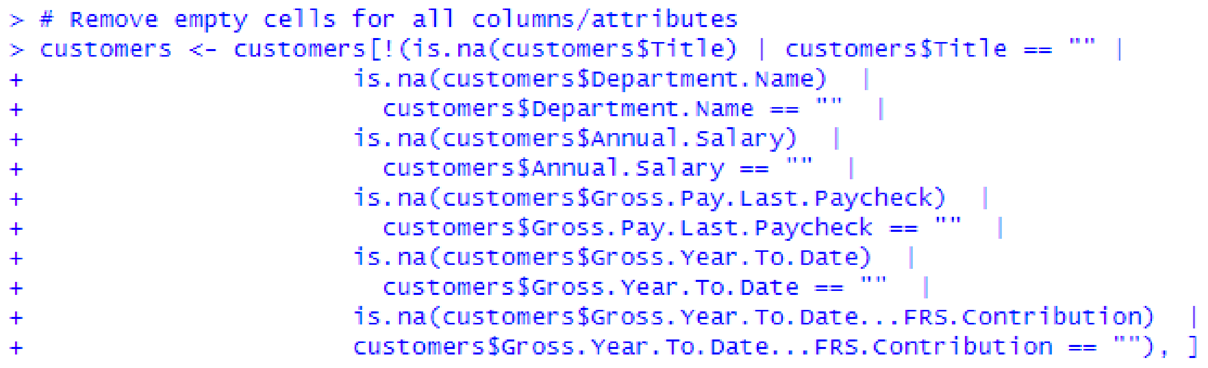
Figure 26 Fill "marital_status" through mode

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

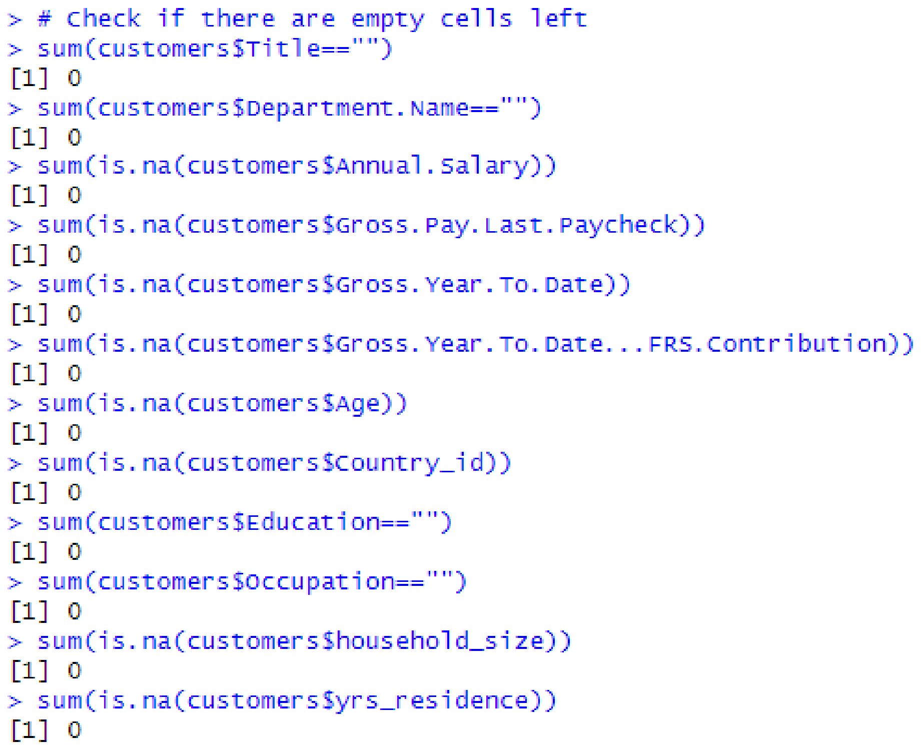
There is six missing values for the attributes “Title”, “Department.Name”, “Annual.Salary”, “Gross.Pay.Last.Paycheck”, “Gross.Year.To.Date”, “Gross.Year.To.Date...FRS.Contribution”. This was identified in the ‘Data Understanding’ phase of the CRISP-DM methodology. It was the same six records that have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

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 == ""), ]
```

Figure 27 Removing Empty Values – Part 1



```
> # 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 28 Removing Empty Values – Part 2

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")
```

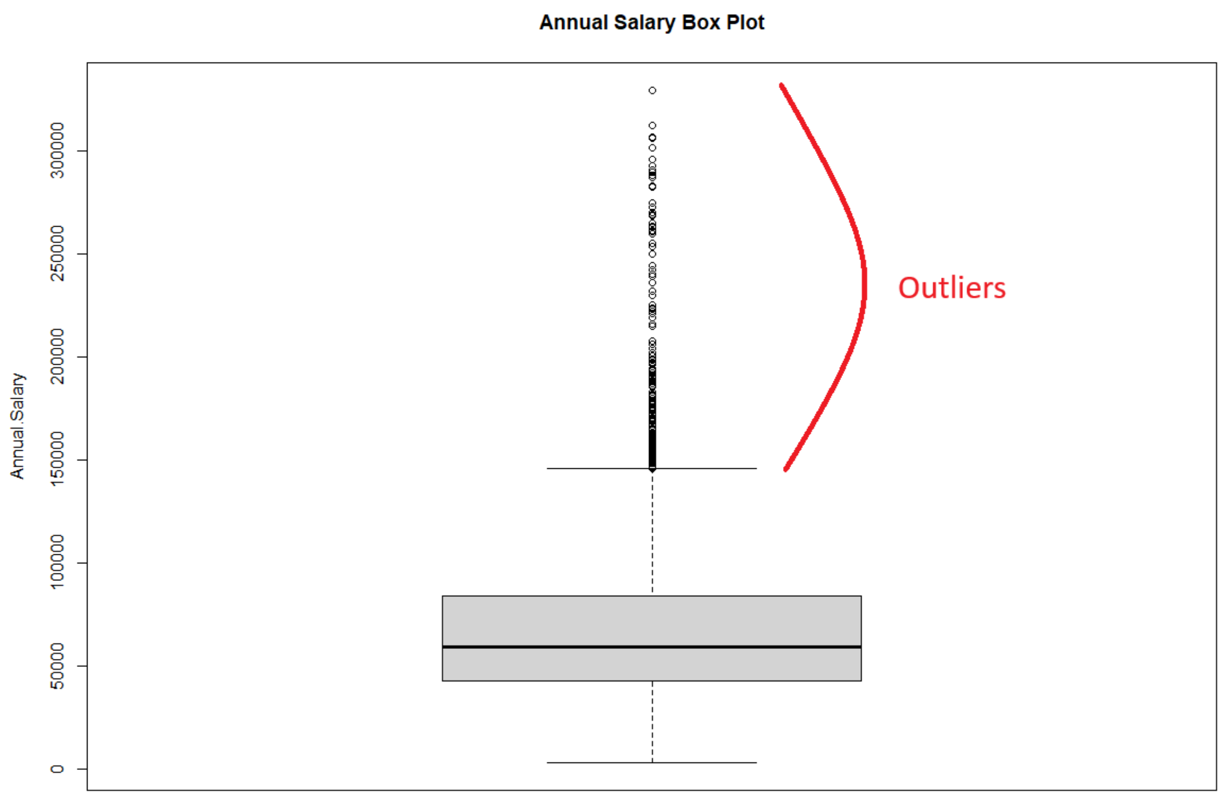


Figure 29 Annual Salary Box Plot

```
### 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")
```

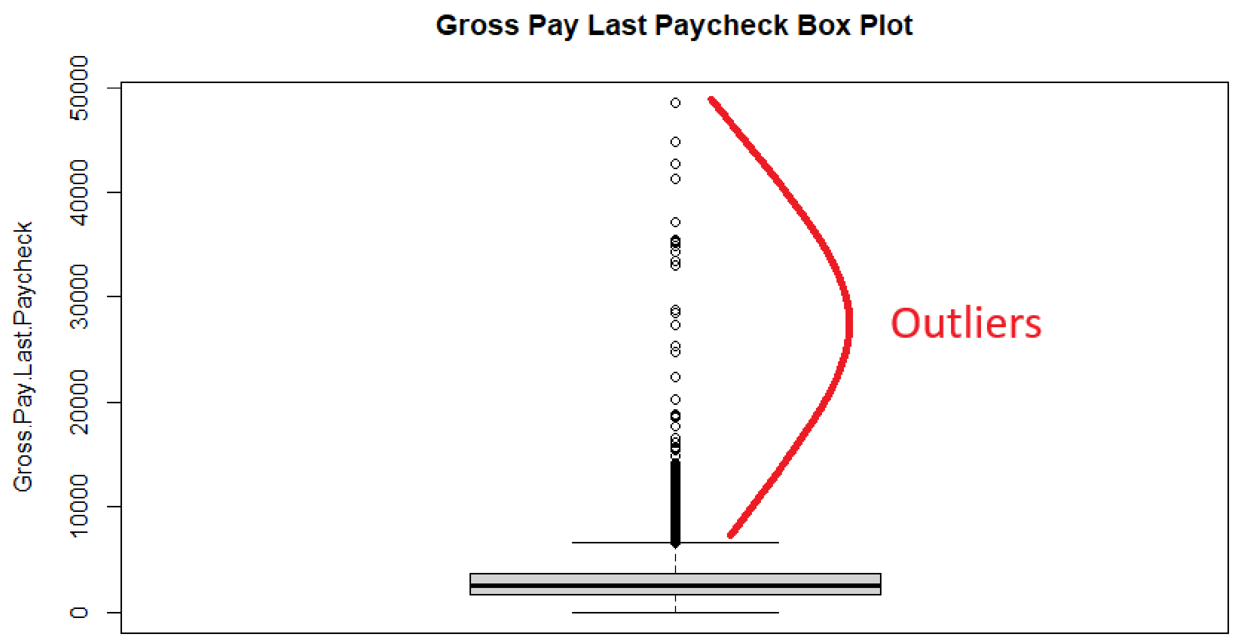


Figure 30 Gross Pay Last Paycheck Box Plot

```
### 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")
```

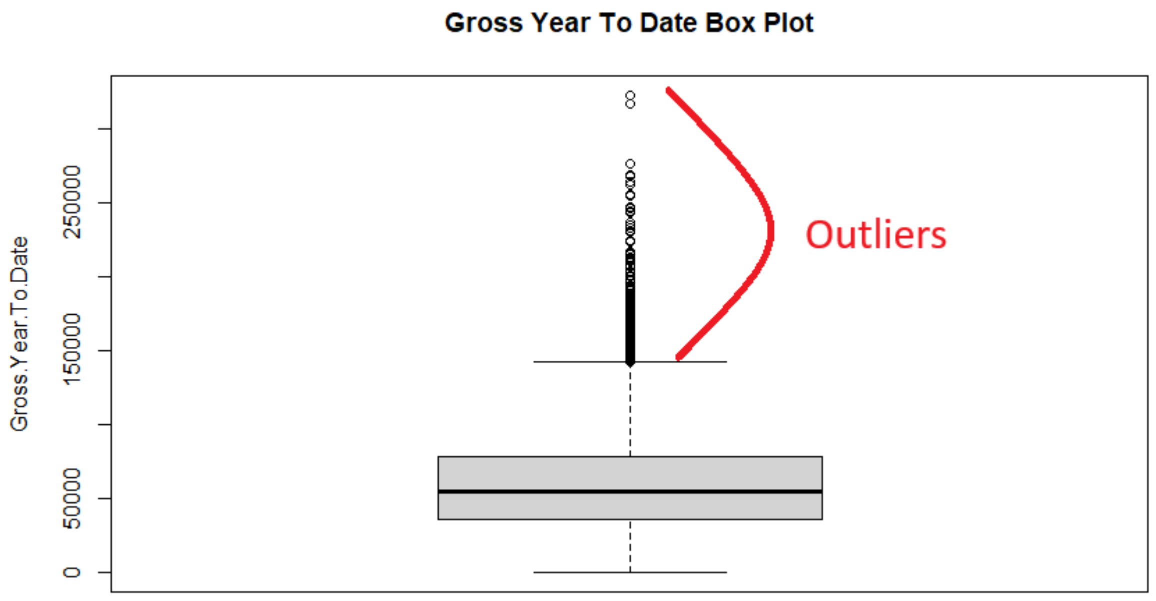


Figure 31 Gross Year To Date Box Plot

```
### 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")
```

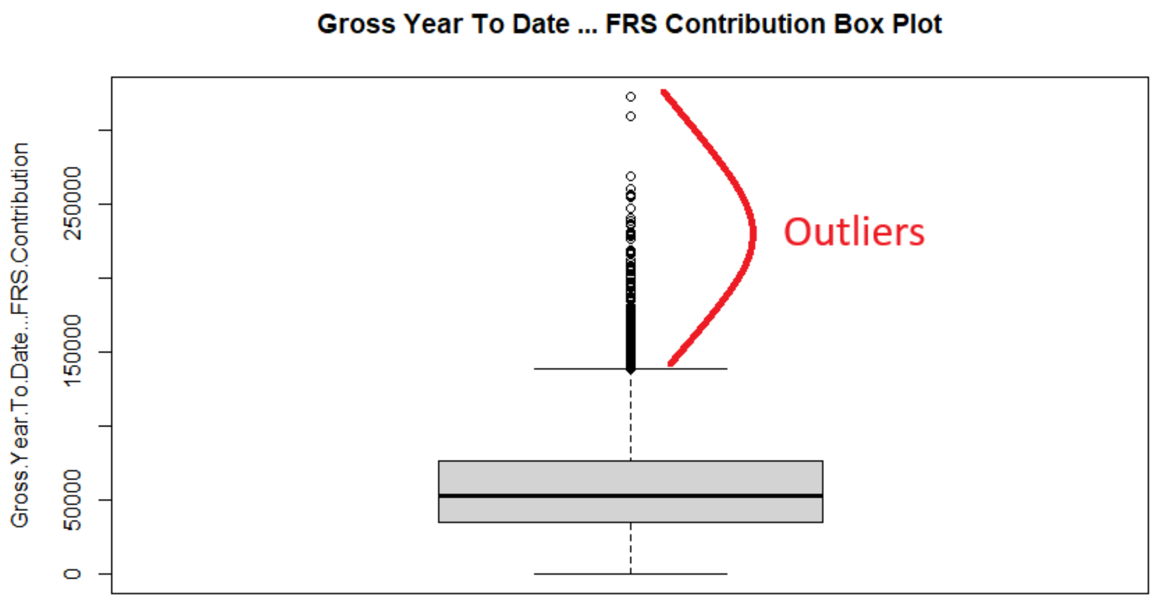


Figure 32 Gross Year to Date ... FRS Contribution Box Plot

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 was used again and the following box plots were generated for the capped attributes:

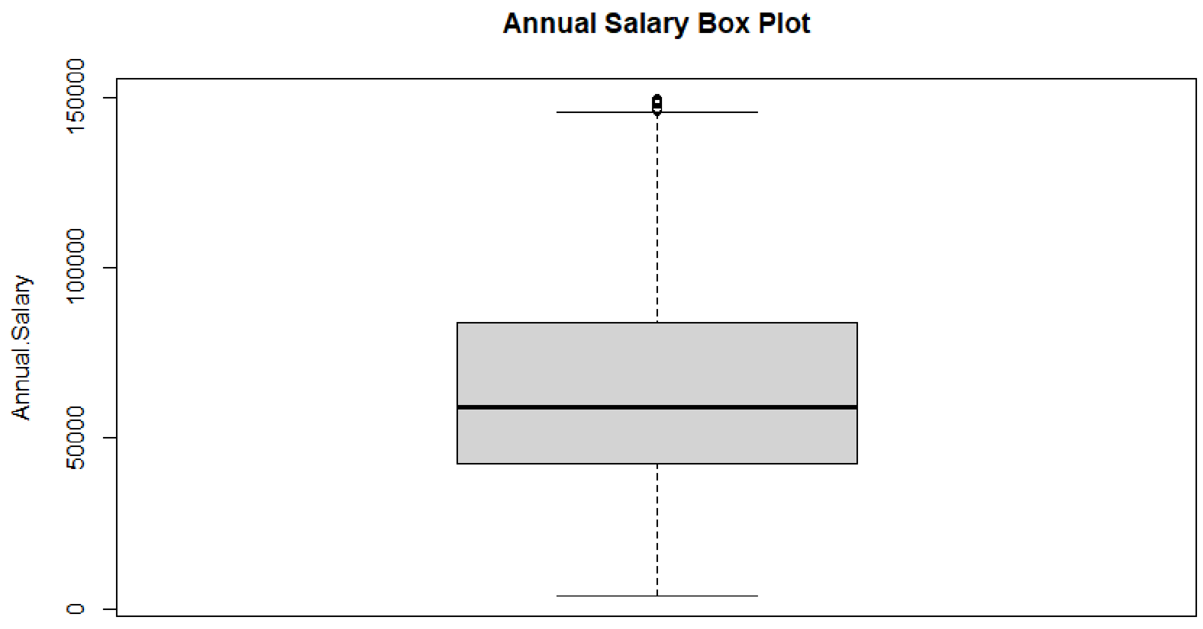


Figure 33 Annual Salary Capped Box Plot

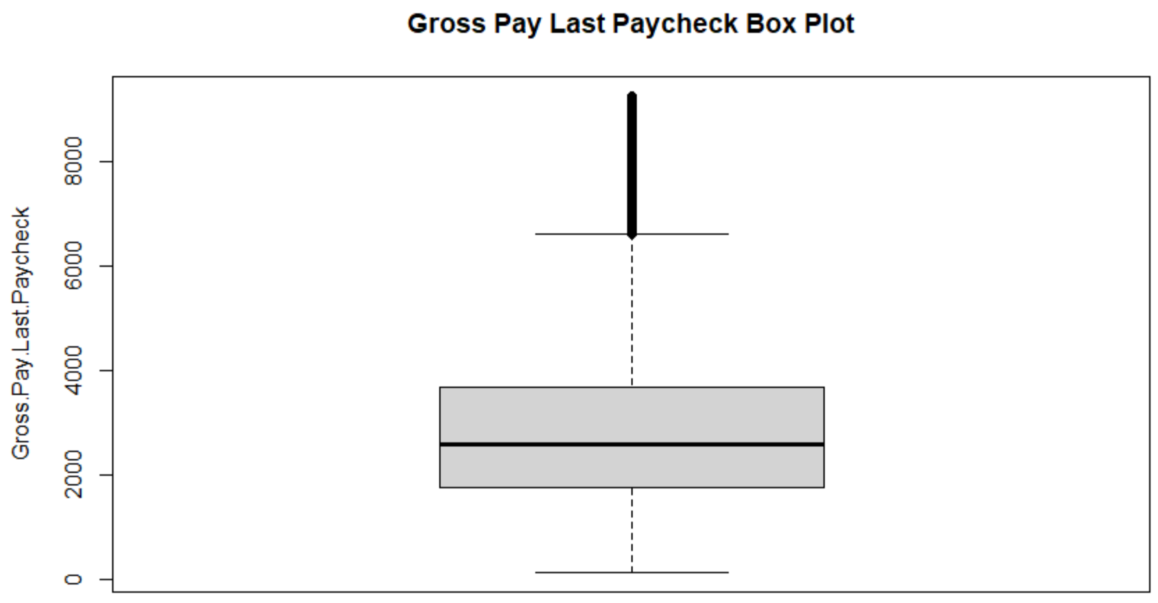


Figure 34 Gross Pay Last Pay Check Capped Box Plot

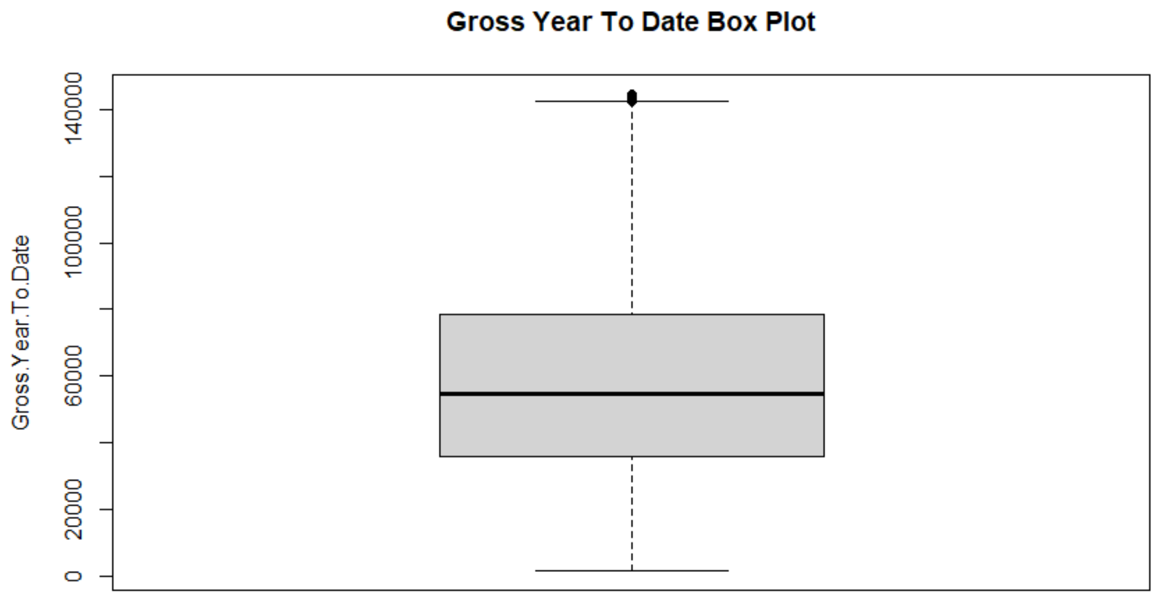


Figure 35 Gross Year To Date Capped Box Plot

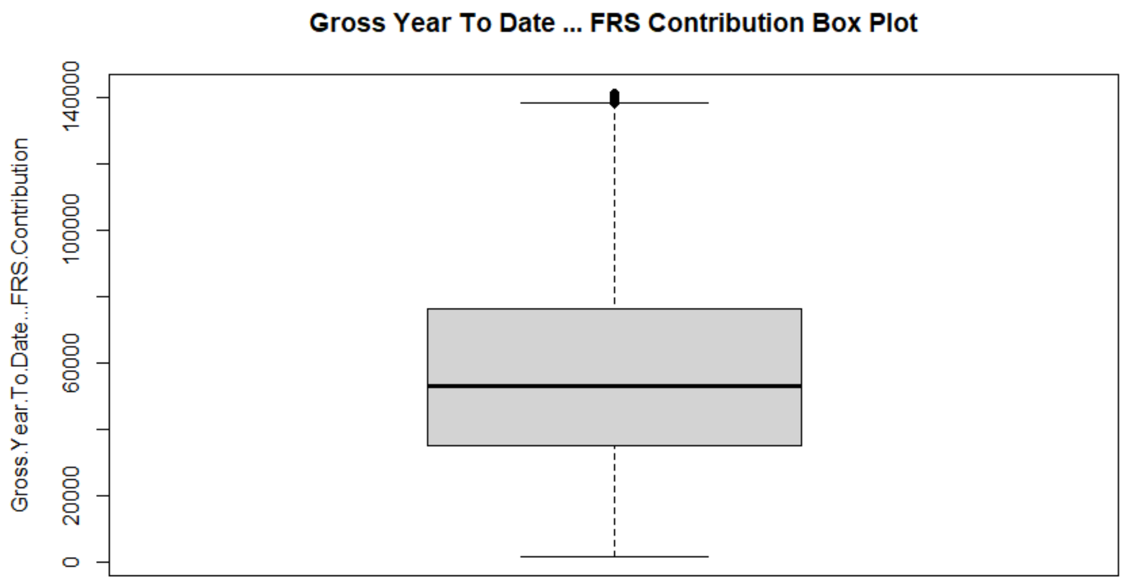
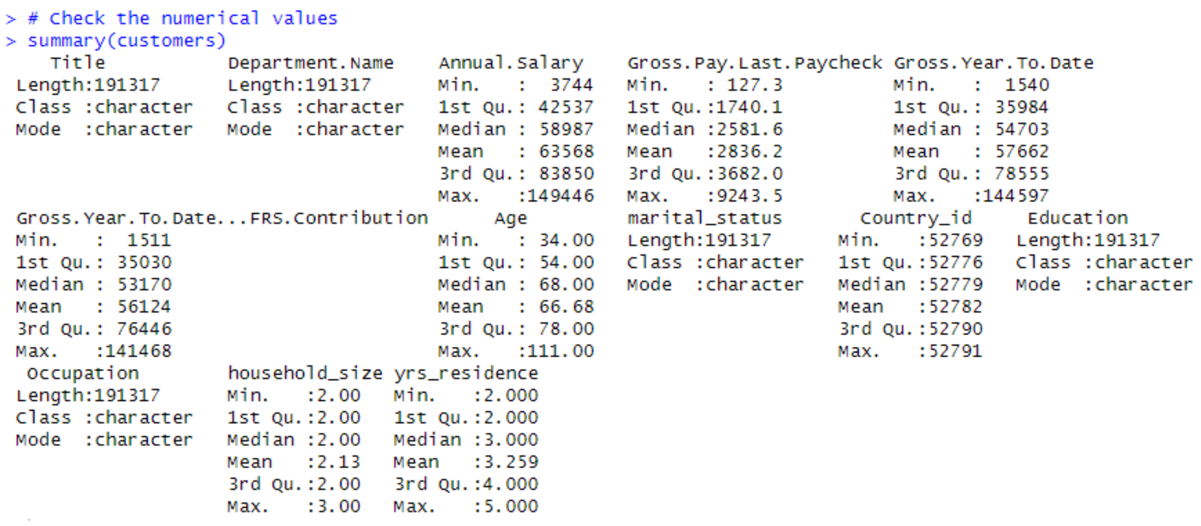


Figure 36 Gross Year to Date ... FRS Contribution Capped Box Plot

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)
    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 37 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:

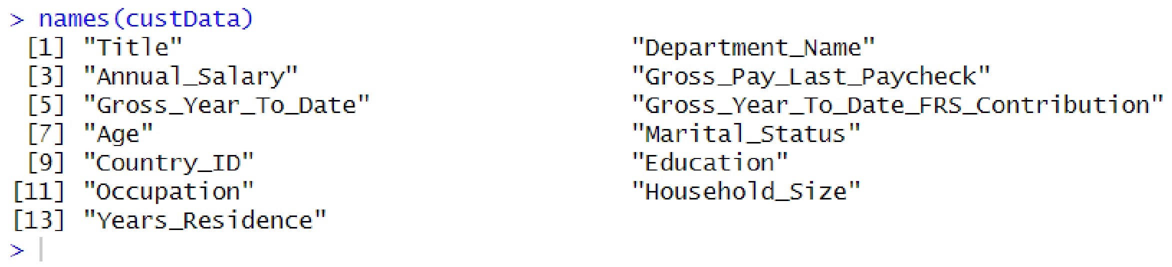
```
# Export to CSV file
write.csv(customers, "CustData2-Cleaned.csv", row.names = FALSE)
```

Data Transformation

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:

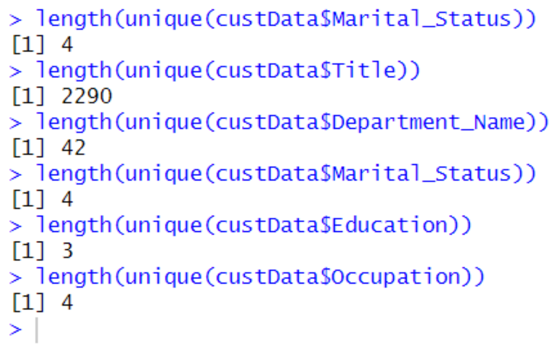


```
> names(custData)
 [1] "Title"                               "Department_Name"
 [3] "Annual_Salary"                       "Gross_Pay_Last_Paycheck"
 [5] "Gross_Year_To_Date"                  "Gross_Year_To_Date_FRS_Contribution"
 [7] "Age"                                 "Marital_Status"
 [9] "Country_ID"                          "Education"
[11] "Occupation"                          "Household_Size"
[13] "Years_Residence"
> 
```

Figure 38 Column Name Transformation

Categorisation of Data

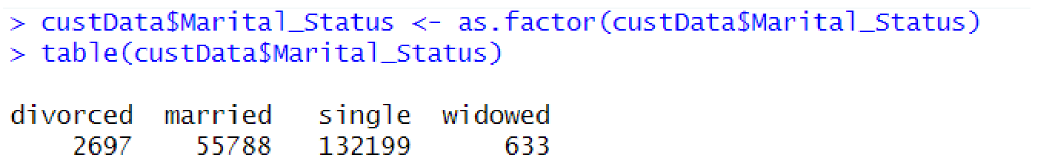
There are various attributes containing categorical data. These attributes include Marital Status, Education and Occupation. This is determined by applying the unique function to find the number of unique values in a column.



```
> 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 39 Unique values for Categorisation

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

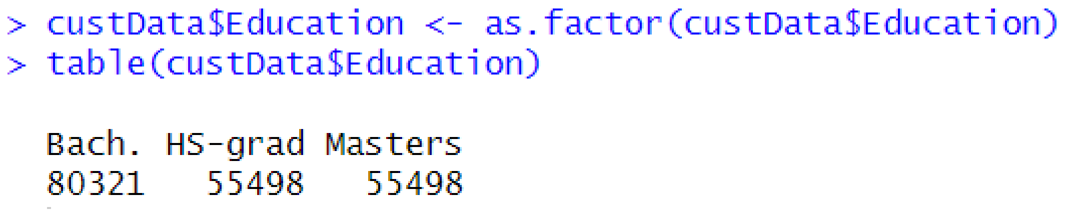


```
> custData$Marital_Status <- as.factor(custData$Marital_Status)
> table(custData$Marital_Status)

divorced  married   single  widowed 
    2697    55788   132199      633 
```

Figure 40 Marital Status Categories

As seen in the figure below, Marital Status will be categorised into the following categories:

- Divorced
- Married
- Single
- Widowed

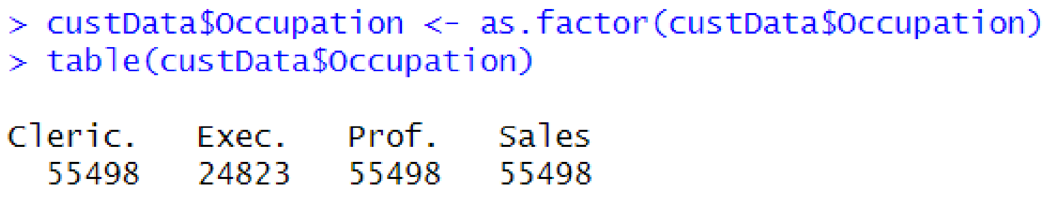


```
> custData$Education <- as.factor(custData$Education)
> table(custData$Education)

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

Figure 41 Education Categories

As seen in the figure below, 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
```

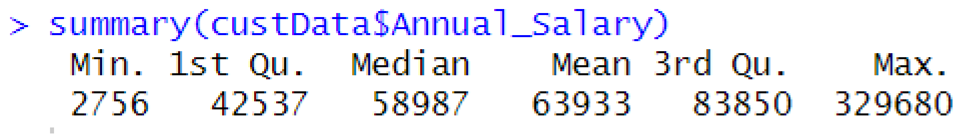
Figure 42 Occupation Categories

As seen in the above figure, Occupation will be categorised by:

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

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.



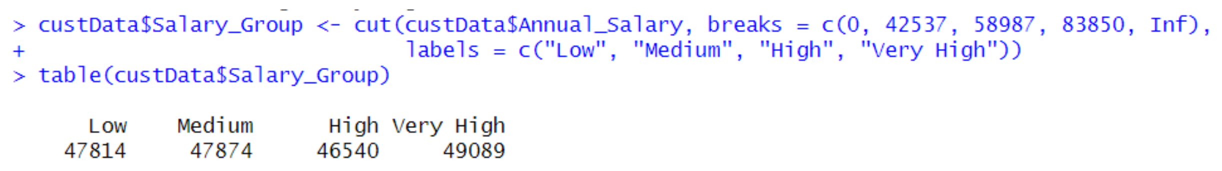
```
> summary(custData$Annual_Salary)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
   2756   42537   58987   63933   83850  329680
```

Figure 43 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:



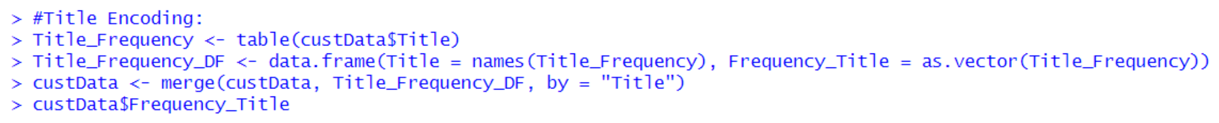
```
> custData$Salary_Group <- cut(custData$Annual_Salary, breaks = c(0, 42537, 58987, 83850, Inf),
+                              labels = c("Low", "Medium", "High", "Very High"))
> table(custData$Salary_Group)

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

Figure 44 Binning of Annual Salary

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.

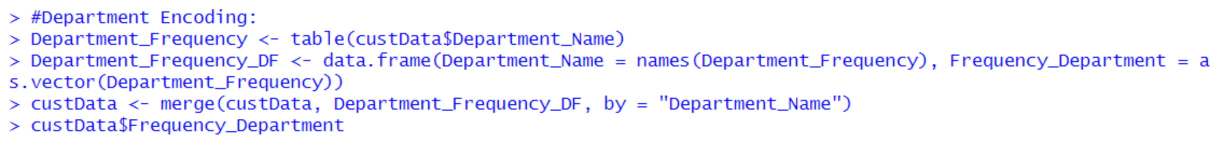
For encoding title, a new attribute called Title Frequency will be created which will store the title as a count of itself in the dataset.



```
> #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 45 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.

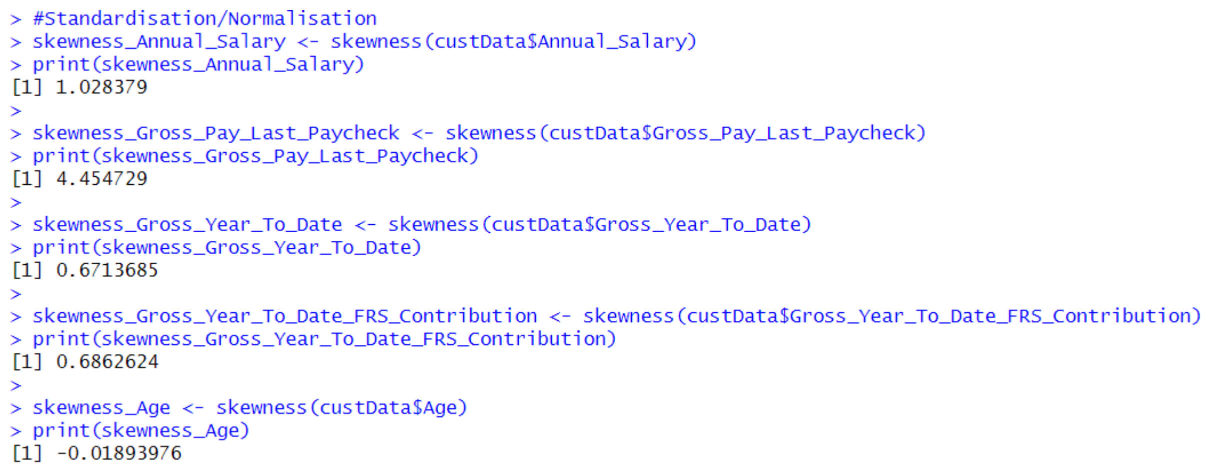


```
> #Department Encoding:
> Department_Frequency <- table(custData$Department_Name)
> Department_Frequency_DF <- data.frame(Department_Name = names(Department_Frequency), Frequency_Department = a
s.vector(Department_Frequency))
> custData <- merge(custData, Department_Frequency_DF, by = "Department_Name")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

Figure 46 Department Name Frequency Encoding

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 47 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.

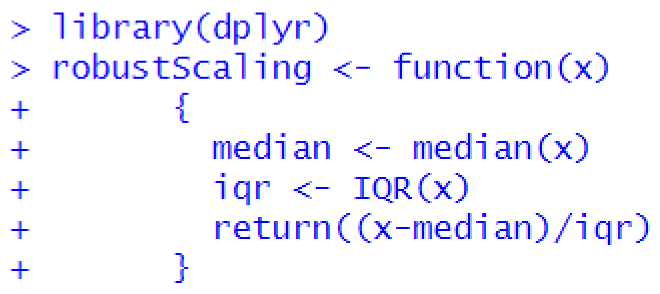
The skewness of Gross Year To Date is slightly positive. This is almost symmetrical, but there is still a small number of customers earning higher yearly gross amounts.

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).

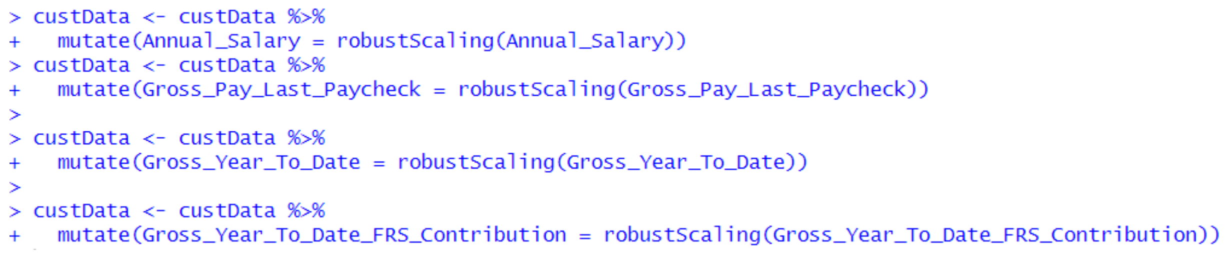
The scaling will be done by taking the value and subtracting the median and then dividing that value by the inter quartile range. This can be seen in the figure below:



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

Figure 48 Function for Robust Scaling

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

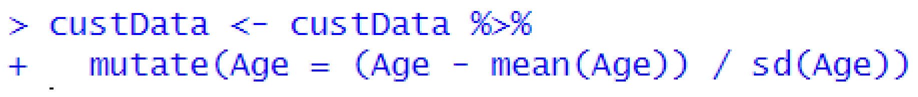


```
> custData <- custData %>%
+   mutate(Annual_Salary = robustScaling(Annual_Salary))
> custData <- custData %>%
+   mutate(Gross_Pay_Last_Paycheck = robustScaling(Gross_Pay_Last_Paycheck))
>
> custData <- custData %>%
+   mutate(Gross_Year_To_Date = robustScaling(Gross_Year_To_Date))
>
> custData <- custData %>%
+   mutate(Gross_Year_To_Date_FRS_Contribution = robustScaling(Gross_Year_To_Date_FRS_Contribution))
```

Figure 49 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 50 Z-Standardisation of Age

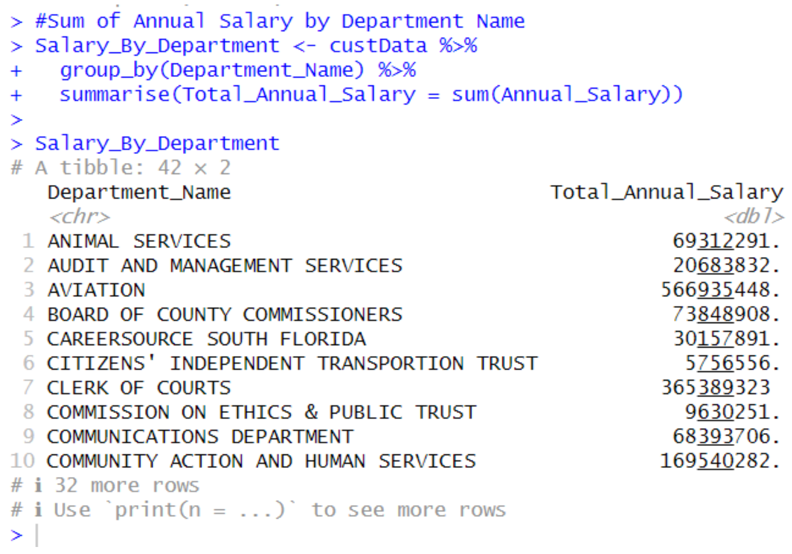
As seen in the figure above, Age has been standardised.

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.

The Sum of Annual Salary by Department Name

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

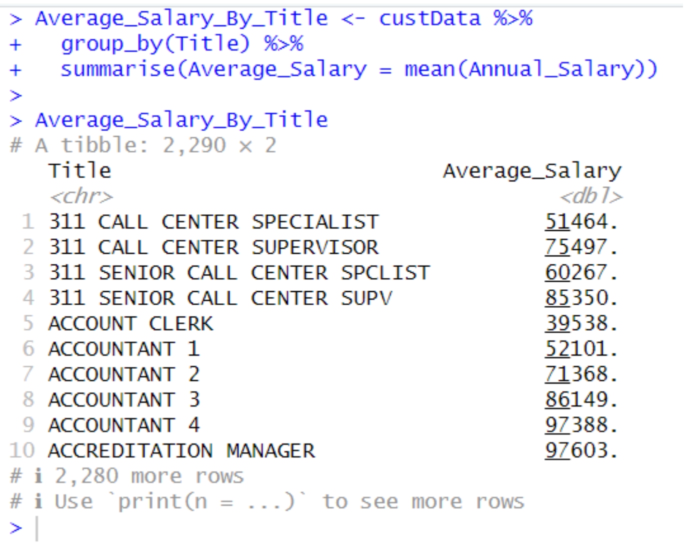


```
> #Sum of Annual Salary by Department Name
> Salary_By_Department <- custData %>%
+   group_by(Department_Name) %>%
+   summarise(Total_Annual_Salary = sum(Annual_Salary))
>
> Salary_By_Department
# A tibble: 42 × 2
   Department_Name                            Total_Annual_Salary
   <chr>                                                    <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 51 Total Salary By Department Name

Average Annual Salary by Title

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

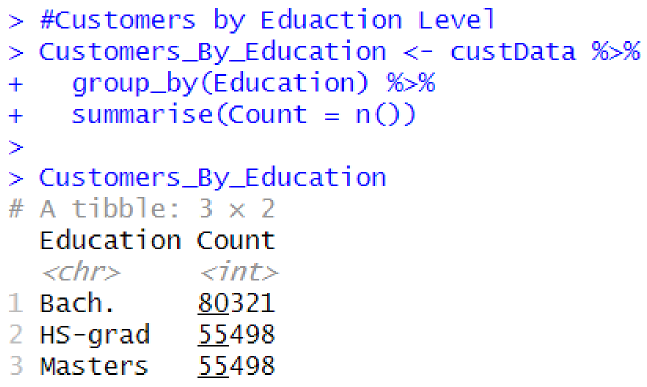


```
> Average_Salary_By_Title <- custData %>%
+   group_by(Title) %>%
+   summarise(Average_Salary = mean(Annual_Salary))
>
> Average_Salary_By_Title
# A tibble: 2,290 × 2
   Title                          Average_Salary
   <chr>                                   <dbl>
 1 311 CALL CENTER SPECIALIST             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 52 Average Annual Salary By Title

Customers By Education Level

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

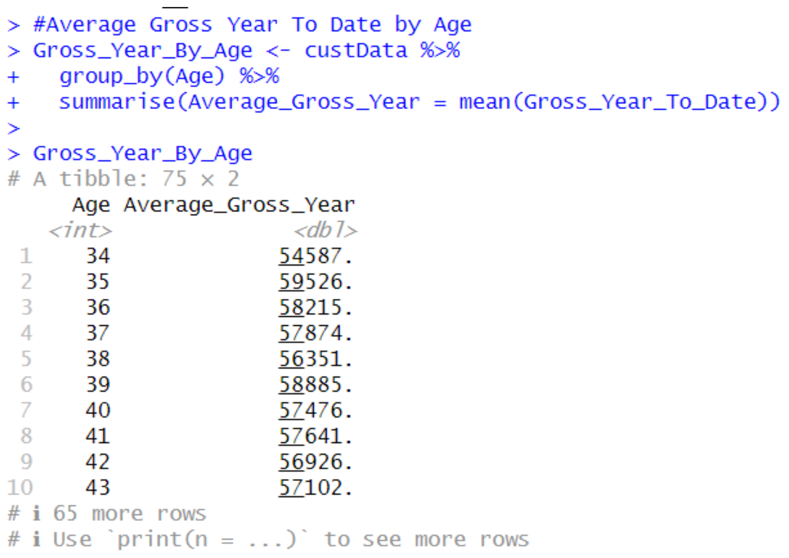


```
> #Customers by Eduaction Level
> Customers_By_Education <- custData %>%
+   group_by(Education) %>%
+   summarise(Count = n())
>
> Customers_By_Education
# A tibble: 3 × 2
  Education Count
  <chr>     <int>
1 Bach.     80321
2 HS-grad   55498
3 Masters   55498
```

Figure 53 Total Customers by Education Level

Average Gross Year To Date By Age

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

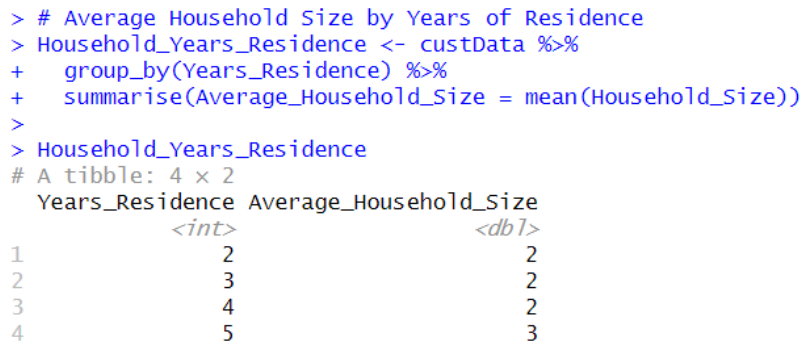


```
> #Average Gross Year To Date by Age
> Gross_Year_By_Age <- custData %>%
+   group_by(Age) %>%
+   summarise(Average_Gross_Year = mean(Gross_Year_To_Date))
>
> Gross_Year_By_Age
# A tibble: 75 × 2
     Age Average_Gross_Year
   <int>              <dbl>
 1    34             54587.
 2    35             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 54 Gross Year To Date By Age

Average Household size by Years in Residence

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

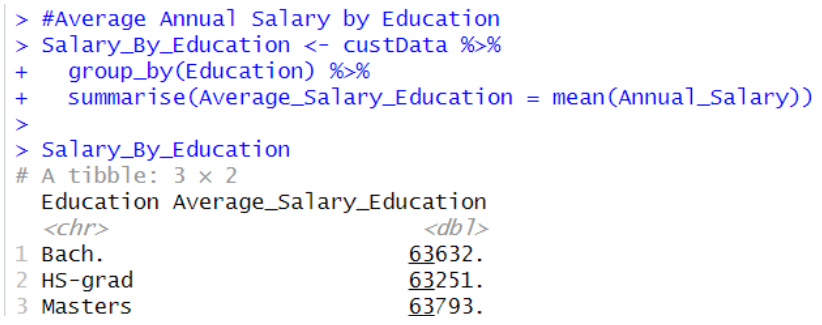


```
> # Average Household Size by Years of Residence
> Household_Years_Residence <- custData %>%
+   group_by(Years_Residence) %>%
+   summarise(Average_Household_Size = mean(Household_Size))
>
> Household_Years_Residence
# A tibble: 4 × 2
  Years_Residence Average_Household_Size
            <int>  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

Figure 55 Average Household Size by Years in Residence

Average Annual Salary by Education Level

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

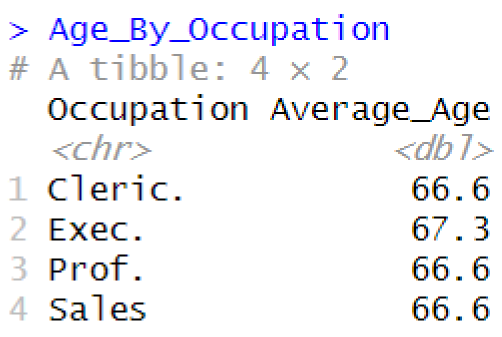


```
> #Average Annual Salary by Education
> Salary_By_Education <- custData %>%
+   group_by(Education) %>%
+   summarise(Average_Salary_Education = mean(Annual_Salary))
>
> Salary_By_Education
# A tibble: 3 × 2
  Education Average_Salary_Education
  <chr>                        <dbl>
1 Bach.       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

Figure 56 Average Annual Salary by Level of Education

Average Age by Occupation

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

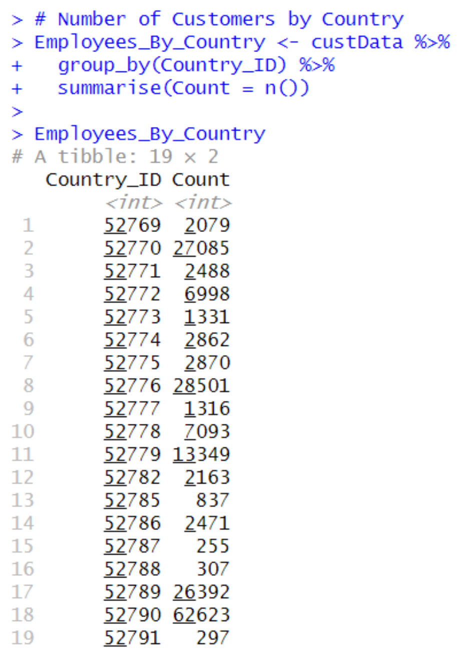


```
> Age_By_Occupation
# A tibble: 4 × 2
  Occupation Average_Age
  <chr>            <dbl>
1 Cleric.           66.6
2 Exec.             67.3
3 Prof.             66.6
4 Sales             66.6
```

Figure 57 Average Age by Occupation

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 58 Number of Customers by Country

Phase 4 – Modelling

This milestone focuses on the CRISP-DM (Cross-Industry Standard Process for Data Mining) framework's fourth phase, modelling, which is crucial for creating classification models that precisely identify people who meet the requirements for service offers based on demographic information. This milestone highlights how crucial it is to use the right supervised learning strategies to solve the specified business problem. The intention is to create preliminary models that can be improved upon in later milestones, guaranteeing a complete comprehension of the modelling procedure and its consequences for practical uses. This milestone focusses on improving on the models created in milestone 3 and selecting the most appropriate model for this project. Each models' characteristics and evaluation metrics will be analysed to create a conclusion.

Choosing appropriate modelling techniques, recording the selected algorithms, and outlining their assumptions are the activities listed for this milestone. This methodical approach will direct the creation of a strong Proof of Concept (PoC) that acts as a basis for finalizing the models and will assist in choosing the final model.

Modelling Technique

Logistic Regression

Definition
Logistic regression is a statistical model used to predict a binary outcome from one or more predictor variables. Unlike linear regression, which predicts a continuous output, logistic regression is specifically designed for cases where the dependent variable is categorical, typically binary, such as variables which have yes/no values (Hosmer, et al., 2013). It uses a logistic function to model the relationship between the dependent variable and the independent variables, providing probabilities that fall between 0 and 1. This makes it particularly useful for classification problems (Kleinbaum & Klein, 1994).

Reasons for use
One of the main reasons for using logistic regression is its interpretability. It is a relatively simple model, meaning its coefficients can be understood as representing the change in the odds of the dependent variable occurring with a one-unit change in the predictor variable. In the context of predicting eligibility, logistic regression works well because it can handle both continuous and categorical variables effectively. For example, a model might use continuous variables like age and income, along with categorical variables such as marital status, to predict whether someone is eligible for a service. The logistic regression model uses a logistic function to map the combination of these independent variables to a

probability that the dependent variable will occur (Hosmer, et al., 2013). It should provide an accurate prediction model for determining the classes of eligibility for the customers.

Assumptions

- Binary Dependent Variable: Logistic regression assumes that the outcome variable is binary (0/1). This makes it a natural choice for classification problems where the goal is to predict one of two outcomes
- Linear Relationship Between Log Odds and Predictors: Although logistic regression models a binary outcome, it assumes that the log odds (logarithm of the odds) of the outcome is linearly related to the predictor variables.
- Independence of Errors: It assumes that the observations are independent of each other and that there is no multicollinearity among the predictors (Hosmer, et al., 2013).
- No Perfect Separation: Logistic regression requires that the predictors are not perfectly correlated with the outcome. In other words, there should be no combination of predictor variables that perfectly predicts the outcome (Kleinbaum & Klein, 1994).
- Sufficient Sample Size: Logistic regression generally requires a large sample size to provide reliable estimates of the relationship between predictor variables and the outcome.

Decision Tree

Definition

The C4.5 algorithm is a popular decision tree algorithm often used for classification tasks. It is known for its capability to generate visual models that help interpret relationships between input data and outcomes. In the context of predicting service eligibility, the C4.5 algorithm would allow for easy visualization of how demographic factors influence eligibility.

Reasons for Use

A key reason for selecting C4.5 is its versatility. As a non-parametric method, it does not require the data to follow any particular distribution, making it suitable for handling both categorical and numerical data (Kotsiantis, 2011). This makes it an excellent fit for datasets that have a mix of continuous and discrete features, such as demographic and financial information that are present in the customer dataset.

Assumptions

One of the strengths of the C4.5 algorithm is that it operates without strict assumptions about the underlying data. Unlike many statistical models, it does not assume linearity or normal distribution, making it flexible across a wide range of data types (Fouché & Langit, 2011).

Random Forest

Definition

Random Forest is an ensemble learning technique that constructs multiple decision trees during training and outputs the mode of the classes (classification) or mean prediction (regression) from individual trees. It is designed to combine the simplicity of decision trees with the power of ensemble learning to create a more robust and accurate model. This method is useful for increasing prediction accuracy and preventing overfitting, a common issue with individual decision trees (Breiman, 2001).

Reasons for use

The key advantage of Random Forest is its ability to enhance generalization on unseen data. This is achieved by combining predictions from multiple trees, thereby limiting the variance that leads to overfitting. Overfitting occurs when a model captures noise or peculiarities in the training data, resulting in poor performance on new data. Random Forest reduces this risk by averaging the predictions of numerous trees, ensuring the model is not overly dependent on any single tree's predictions (Breiman, 2001).

Another benefit of using Random Forest is its ability to evaluate the importance of various features in the dataset. This makes it a valuable tool for identifying the most relevant demographic characteristics in predicting eligibility for services, as it highlights which features contribute the most to the model's predictive power (Hastie, et al., 2009). Applying this to the customer dataset will not only allow for the most important attributes to be identified but will also limit overfitting which will improve the accuracy of the model.

Assumptions

Random Forest makes the assumption that the decision trees built from different samples of the data will collectively capture more information than any single tree alone. It also assumes that the more diverse the trees, the better the ensemble will perform. This is why Random Forest uses techniques like bagging (bootstrap aggregating) to create different subsets of the data for each tree. By using random samples and splitting points, Random Forest assumes that the aggregated result of these trees will provide a more accurate and generalized prediction (Breiman, 2001).

Test Design

This section describes how to evaluate three classification models: Logistic Regression, Decision Tree (C4.5), and Random Forest to find which model is most suitable for predicting client eligibility for service contracts. These models were chosen because they are well-suited for classification tasks, which are aimed at finding variables that affect customer service eligibility other than annual salary. The dataset will be separated into training and test sets to ensure unbiased evaluation to prevent 'overfitting'. Models will be compared using performance indicators such as accuracy, precision, recall, F1 score, and confusion matrices. The purpose is to find the model that best satisfies the project's business objectives.

Overfitting in machine learning is when an algorithm fits its training data too closely, or perhaps too precisely, producing a model that is unable to draw reliable conclusions or predictions from any other data (IBM, 2024).

Data Splitting Strategy

A conventional data splitting strategy to guarantee reliable model evaluation will be used:

- Training set: The model is trained using 80% of the dataset.
- Test set: 20% of the dataset is put aside for the last assessment of the model. By ensuring that model performance is evaluated on unknown data, this splitting technique reduces the possibility of overfitting and offers an objective assessment of the model's generalizability.

The code used to split the dataset is as follows:

```
set.seed(123)
```

Reproducible random numbers are obtained with this function. When a random function is called, it aids in producing the same random numbers each time. This aids in producing data sets for analysis that may be repeated (Biet, 2023).

```
train_index <- createDataPartition(custData$Eligible, p = 0.8, list = FALSE)
```

The object “train_index” is used to store the split dataset obtained from the “createDataPartition()” function. The parameters for this function are as follows:

- ‘Cust$Eligible’: This parameter must provide the target attribute of the dataset, which is in this case the “Eligible” attribute.
- ‘p = 0.8’: This parameter provides the percentage of the dataset that should be used for training the dataset, in this case it’s 80%. The rest of the dataset (in this case 20%) will be used for testing.

```
train_data <- custData[train_index, ]
```

The object “training_set” is used to contain the training portion of the dataset. This is done by indexing the “custData” using the “train_index” object.

```
test_data <- custData[-train_index, ]
```

The object “test_set” is used to contain the training portion of the dataset. This is done by indexing the “custData” using the opposite of the “train_index” object.

Model Evaluation Metrics

The following measures will be used to assess each model's performance:

- **Accuracy**: The frequency with which a machine learning model accurately predicts the result is measured by its accuracy. Accuracy is computed by dividing the total number of predictions by the number of right predictions, accuracy may be computed (Evidently AI, 2024).
- **Precision**: The frequency with which a machine learning model accurately predicts the positive class (eligible customers) is measured by this statistic. By dividing the total number of occurrences, the model predicted as positive (including true and false positives) by the number of accurate positive predictions (true positives), you can compute precision (Evidently AI, 2024).
- **Recall**: A machine learning model's recall is a metric that expresses how frequently it properly selects positive examples, or true positives, out of all the real positive samples in the dataset. Recall can be computed by dividing the total number of positive cases by the number of true positives. The latter consists of false negative results (missed cases) and true positives (successfully detected cases) (Evidently AI, 2024).
- **F1-Score**: An important machine learning metric that offers a fair assessment of a model's recall and precision is the F1 Score. The precision recall F1 score framework requires the F1 Score formula, which is obtained from the harmonic mean of precision and recall. When there is an imbalance in the distribution of classes, this statistic becomes especially helpful (Geeksforgeeks, 2024).
- **Confusion Matrix**: Two target classes are conceivable in binary classification; these are usually designated as "positive" and "negative," or "1" and "0." The target (positive class) in an example of spam is "spam," whereas the negative class is "not spam". One must consider both accurate and inaccurate predictions, ignoring the class designation, when assessing accuracy. It can be "correct" or "wrong" in binary classification, but, in two distinct ways. This includes the following four classifications (Jacob Murel Ph.D., 2024):

- True positive (TP): An email that the model accurately classifies as spam, and it is actually spam.
- True negative (TN): is an email that the model correctly classifies as non-spam, and it isn't spam.
- False Positive (FP): An email that the model mistakenly classifies as spam but is actually not (a "false alarm").
- False Negative (FN): An email that the model mistakenly classifies as non-spam but is in fact spam ("missed spam").

The layout of the confusion matrix is as follows:

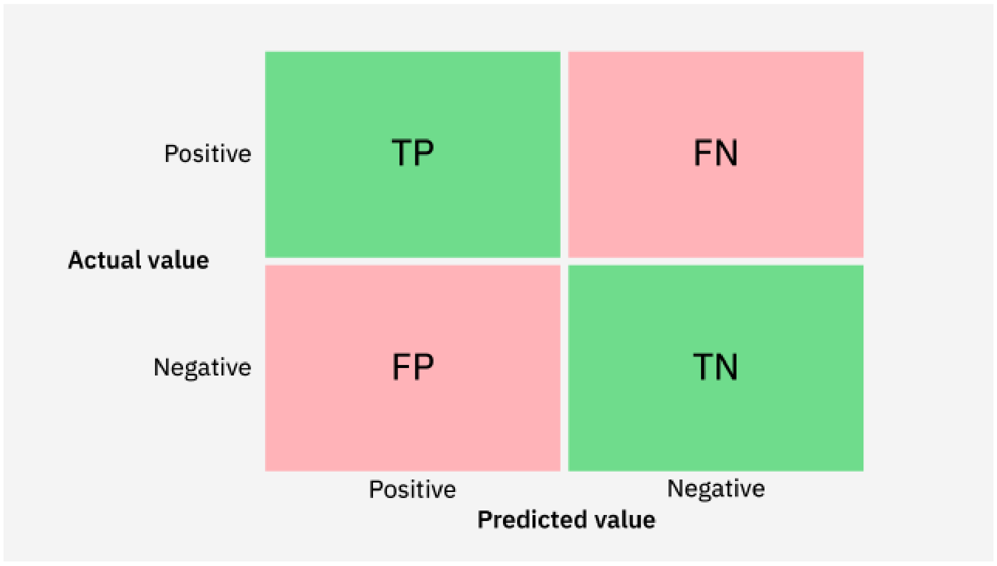


Figure 59 Confusion Matrix Layout (Jacob Murel Ph.D., 2024)

The confusion matrix is used to calculate come of the evaluation metrics. The formulas are given below.

$$Accuracy = \frac{TP + TF}{TP + FP + TF + FN}$$

$$Precision\ (P) = \frac{TP}{TP + FP}$$

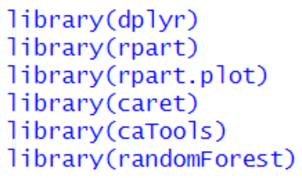
$$Recall\ (R) = \frac{TP}{TP + FN}$$

$$F1 = \frac{2 * P * R}{P + R}$$

Model Description

Splitting of Data

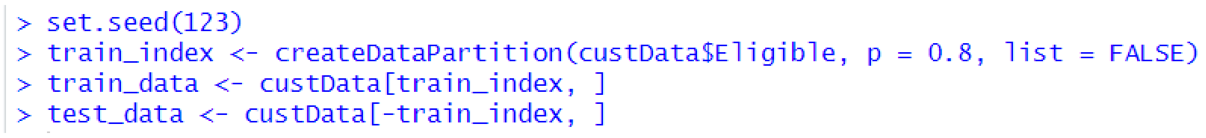
The following libraries will be used during the building of the models:



```
library(dplyr)
library(rpart)
library(rpart.plot)
library(caret)
library(caTools)
library(randomForest)
```

Figure 60 Libraries

- **dyplr:** This package is used for data manipulation. It will be used for factorisation, summarisation and other data manipulation needed to ensure the target attribute (Eligible) is ready for modelling (Data Carpentry, 2024).
- **rpart:** rpart is used to build the decision tree.
- **rpart.plot:** rpart.plot will be used to plot the decision tree.
- **caret:** The caret package will be used to build and train the logistic regression model (Kuhn, 2019).
- **caTools:** The caTools package can be used for splitting the datasets (geeksforgeeks, 2024).
- **randomForest:** The random forest package will be used to build the random forest model.



```
> set.seed(123)
> train_index <- createDataPartition(custData$Eligible, p = 0.8, list = FALSE)
> train_data <- custData[train_index, ]
> test_data <- custData[-train_index, ]
```

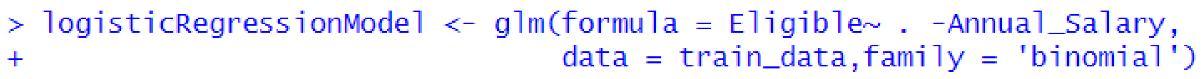
Figure 61 Dataset Splitting

The splitting of the data can occur. The seed will be set to ‘123’ to ensure that reproducibility is possible. This is done by setting a random seed.

The data is split using the createDataPartition function from the caret library. The dataset is split into 80% training data and 20% testing data. This will create a balanced split of data based on the Eligible attribute which is the target attribute.

Logistic Regression

Logistic regression is the first model that will be implemented and evaluated.

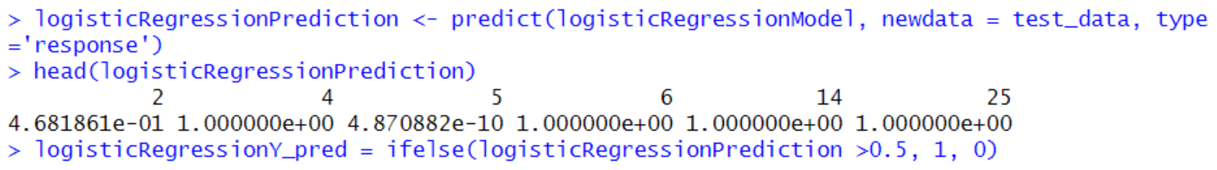


```
> logisticRegressionModel <- glm(formula = Eligible~ . -Annual_Salary,
+                                data = train_data,family = 'binomial')
```

Figure 62 Logistic Regression Model

Parameters:

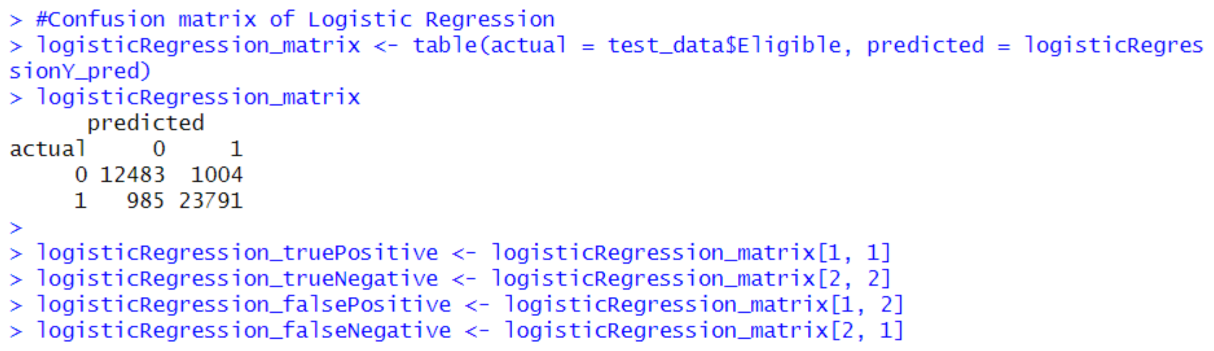
- **Function**: The logistic regression model will be built using the glm function.
- **Target**: This will create a generalised linear model that will find and model the relationship between the target attribute ("Eligible") and the independent attributes (datacamp, 2024).
- **Exclusion:** The Annual_Salary attribute has been excluded from the model to ensure that the model focuses on other attributes and not just Annual_Salary, as it was used to create the Eligibility attribute.
- **Data**: The model will be built on the training data.
- **Family**: The family = 'binomial' parameter assigns the logistic regression algorithm which is used when the target attribute is binary.



```
> logisticRegressionPrediction <- predict(logisticRegressionModel, newdata = test_data, type
='response')
> head(logisticRegressionPrediction)
           2            4            5            6           14           25 
4.681861e-01 1.000000e+00 4.870882e-10 1.000000e+00 1.000000e+00 1.000000e+00 
> logisticRegressionY_pred = ifelse(logisticRegressionPrediction >0.5, 1, 0)
```

Figure 63 Predict Eligibility using Logistic Regression

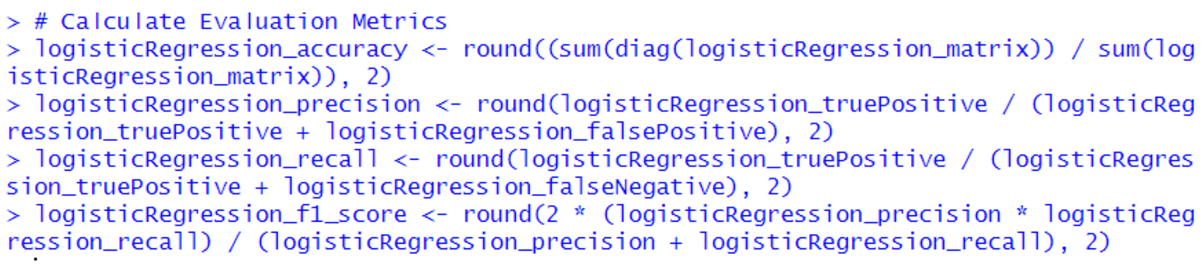
The logistic regression model can then be used to make predictions using the test data, as can be seen in the figure above. This will predict whether users will be eligible or not. If the prediction value is above 0.5 then they will be eligible, and if it is below 0.5 then they will not be eligible.



```
> #Confusion matrix of Logistic Regression
> logisticRegression_matrix <- table(actual = test_data$Eligible, predicted = logisticRegres
sionY_pred)
> logisticRegression_matrix
      predicted
actual     0     1
     0 12483  1004
     1   985 23791
> 
> logisticRegression_truePositive <- logisticRegression_matrix[1, 1]
> logisticRegression_trueNegative <- logisticRegression_matrix[2, 2]
> logisticRegression_falsePositive <- logisticRegression_matrix[1, 2]
> logisticRegression_falseNegative <- logisticRegression_matrix[2, 1]
```

Figure 64 Confusion Matrix for Logistic Regression

A confusion matrix is generated that compares the test data values with the predicted values. The values in the matrix will then be used to find the number of true positives, true negatives, false positives and false negatives produced by the model.



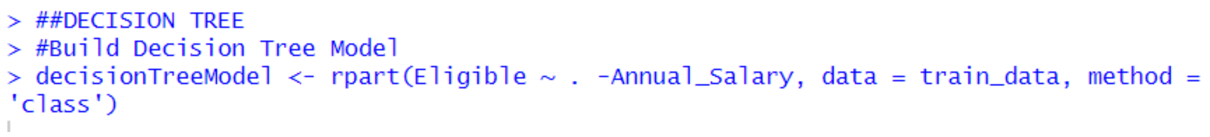
```
> # Calculate Evaluation Metrics
> logisticRegression_accuracy <- round((sum(diag(logisticRegression_matrix)) / sum(log
isticRegression_matrix)), 2)
> logisticRegression_precision <- round(logisticRegression_truePositive / (logisticReg
ression_truePositive + logisticRegression_falsePositive), 2)
> logisticRegression_recall <- round(logisticRegression_truePositive / (logisticRegres
sion_truePositive + logisticRegression_falseNegative), 2)
> logisticRegression_f1_score <- round(2 * (logisticRegression_precision * logisticReg
ression_recall) / (logisticRegression_precision + logisticRegression_recall), 2)
```

Figure 65 Calculation of Performance Metric - Logistic Regression

The accuracy, precision, recall and f1 score for the logistic regression model is then calculated. This can be seen in the figure above.

Decision Tree

The Decision Tree model is the next model to be implemented and evaluated.



```
> ##DECISION TREE
> #Build Decision Tree Model
> decisionTreeModel <- rpart(Eligible ~ . -Annual_Salary, data = train_data, method =
'class')
```

Figure 66 Building of Decision Tree Model

To building of the Decision Tree model will be done using rpart.

Parameters:

- **Function**: rpart will be used to build the decision tree model.
- **Target**: The model will be predicting the target attribute (“Eligible”).
- **Exclusion**: Annual_Salary will be excluded from the model to ensure that other attributes are considered as Annual_Salary will bias the data as it was used to calculate the Eligible attribute.
- **Data**: The model will be built on the training data.
- **Method**: The method = ‘class’ parameter specifies that a classification model should be built.

After the decision tree model has been built, the decision tree can be visualized. This can be seen in the following figure:

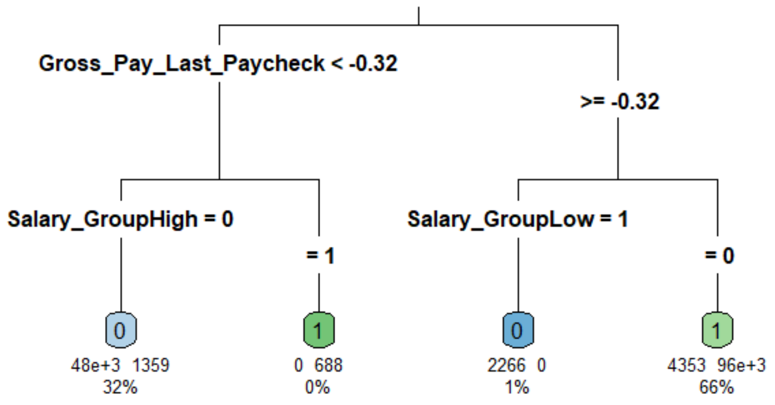


Figure 67 Decision Tree

The main attribute which determines user eligibility is Gross Pay Last Paycheck. The split is based on the scaled threshold of -0.32.

If the customer has a Gross_Pay_Last_Paycheck of less than -0.32, then the branch is further split into branches based on the Salary_GroupHigh.

If the Salary_GroupHigh = 0, this indicates that the customer does not have a high annual salary. If the Gross_Pay_Last_Paycheck is less than -0.32 and the Salary_Group is not high, then the customer will not be eligible for the services provided by LangaSat.

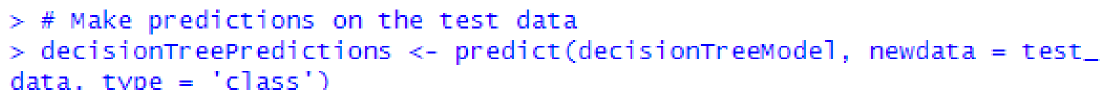
If the Salary_GroupHigh = 1, this indicates that the customer does have a high annual salary. If the Gross_Pay_Last_Paycheck is less than -0.32 and the Salary_Group is high, then the customer will be eligible for the services provided by LangaSat.

The second branching of the main branch is based on of the Gross_Pay_Last_Paycheck is more than the scaled threshold of -0.32. This branch further splits based on the whether or not the annual salary of the customer is low.

If the Salary_GroupLow = 0, this indicates that the customer does not have a low annual salary. If the Gross_Pay_Last_Paycheck is more than -0.32, and the Salary group is not low, then the customer is not eligible for the services provided by LangaSat.

If the Salary_GroupLow = 1, this indicates that the customer does have a low annual salary. If the Gross_Pay_Last_Paycheck is more than -0.32, and the Salary group is low, then the customer is eligible for the services provided by LangaSat.

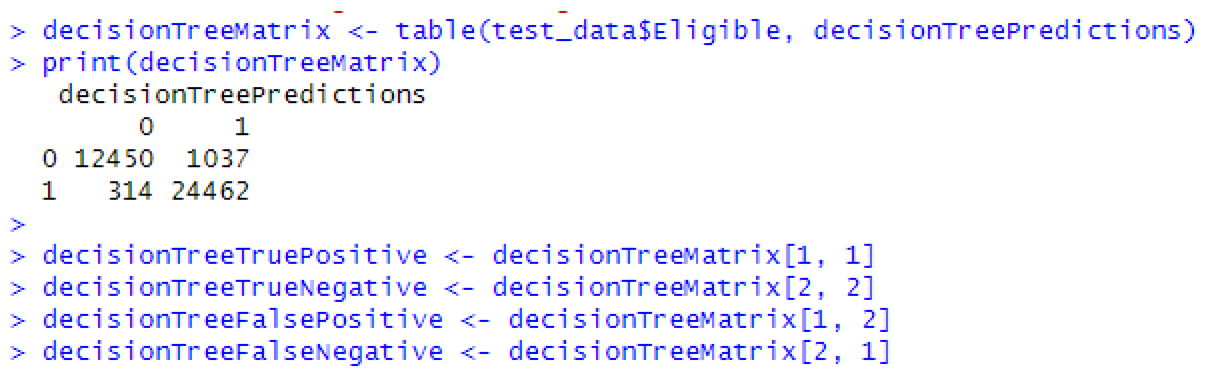
The model can then be used to make predictions using the test data. This can be seen in the following figure:



```
> # Make predictions on the test data
> decisionTreePredictions <- predict(decisionTreeModel, newdata = test_
data, type = 'class')
```

Figure 68 Decision Tree Prediction

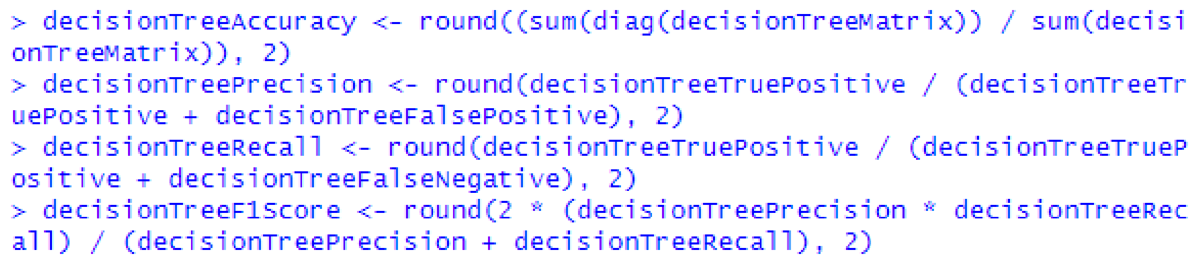
The confusion matrix for the decision tree model can be generated. This will compare the predicted values with the test values. The elements of the confusion matrix can be derived from the matrix to allow for evaluation metrics to be calculated.



```
> decisionTreeMatrix <- table(test_data$Eligible, decisionTreePredictions)
> print(decisionTreeMatrix)
   decisionTreePredictions
        0     1
  0 12450  1037
  1   314 24462
>
> decisionTreeTruePositive <- decisionTreeMatrix[1, 1]
> decisionTreeTrueNegative <- decisionTreeMatrix[2, 2]
> decisionTreeFalsePositive <- decisionTreeMatrix[1, 2]
> decisionTreeFalseNegative <- decisionTreeMatrix[2, 1]
```

Figure 69 Decision Tree Matrix

Finally, the performance metrics for the decision tree can be calculated. The accuracy, precision, recall and f1 will be calculated. All metrics will be rounded to two decimal places.

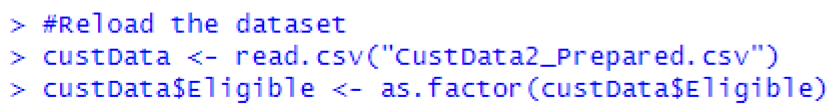


```
> decisionTreeAccuracy <- round((sum(diag(decisionTreeMatrix)) / sum(decisi
onTreeMatrix)), 2)
> decisionTreePrecision <- round(decisionTreeTruePositive / (decisionTreeTr
uePositive + decisionTreeFalsePositive), 2)
> decisionTreeRecall <- round(decisionTreeTruePositive / (decisionTreeTrueP
ositive + decisionTreeFalseNegative), 2)
> decisionTreeF1Score <- round(2 * (decisionTreePrecision * decisionTreeRec
all) / (decisionTreePrecision + decisionTreeRecall), 2)
```

Figure 70 Decision Tree Performance Metric Calculation

Random Forest

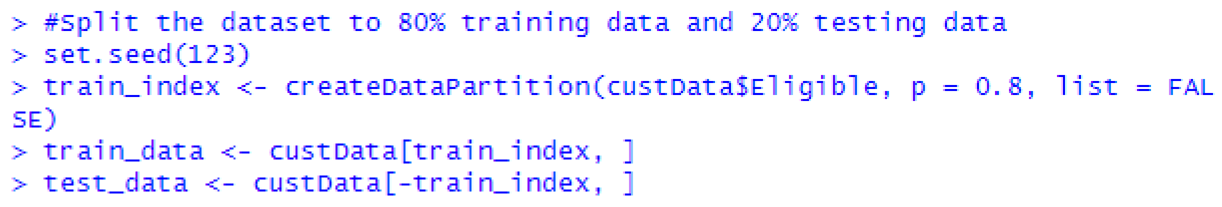
The splitting for the Random Forest model will differ slightly from the other models. For the random forest model, the target attribute "Eligible" will have to be categorized first using factoring. Thus, we will reread the prepared data file and categorise "Eligible".



```
> #Reload the dataset
> custData <- read.csv("CustData2_Prepared.csv")
> custData$Eligible <- as.factor(custData$Eligible)
```

Figure 71 Rereading Prepared Data

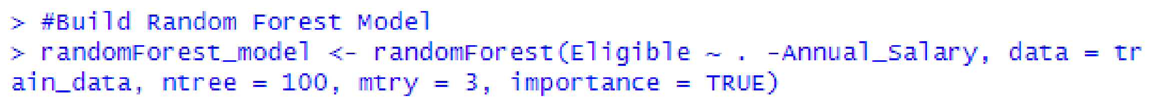
Then the dataset will be split the exact same way as previously done.



```
> #Split the dataset to 80% training data and 20% testing data
> set.seed(123)
> train_index <- createDataPartition(custData$Eligible, p = 0.8, list = FAL
SE)
> train_data <- custData[train_index, ]
> test_data <- custData[-train_index, ]
```

Figure 72 Split Data for Random Forest

Next, the model can be built:



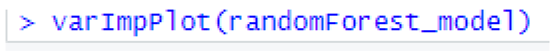
```
> #Build Random Forest Model
> randomForest_model <- randomForest(Eligible ~ . -Annual_Salary, data = tr
ain_data, ntree = 100, mtry = 3, importance = TRUE)
```

Figure 73 Random Forest Model

Parameters:

- **Function**: The model will be build using the randomForest function.
- **Target**: The model will be built to predict the "Eligible" attribute.
- **Exclusion**: Annual Salary will not be used to build the model, as to avoid bias seeing as the Eligible attribute was created using the Annual Salary.
- **Data**: The model will be built using the training data.
- **Ntree**: ntree refers to the number of decision trees that will be created in the random forest. In this parameter, this number has been set to 100, so 100 decision trees will be generated. Although this may result in better performance, it will also require a lot of computational power which may result in slower results.
- **Mtry**: mtry refers to the number of variables that may be chosen at each split in a tree. This attribute can impact the fitting of the model (be it over or under). This has been set to 3, so three variables may be chosen at each split.
- **Importance**: Importance = true; This parameter forces the model to calculate the importance of the attributes.

The importance of the attributes derived calculated by the model can be visualised using the function varImpPlot.



```
> varImpPlot(randomForest_model)
```

Figure 74 Function for Plotting Importance

The plot will be as follows:

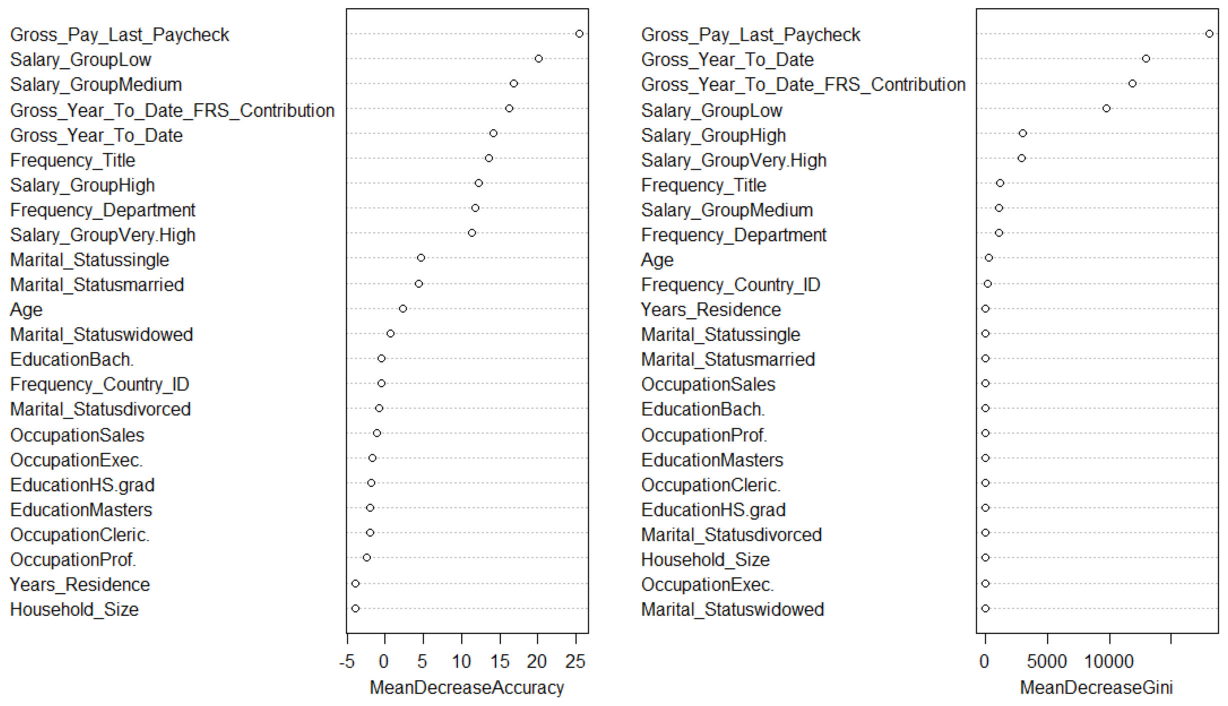


Figure 75 Plot for Attribute Importance

The first plot shows the mean decrease in model accuracy when an attribute is randomly shuffled. The higher the decrease in accuracy, the more important the attribute.

According to the first plot the three most important attributes are:

1. Gross_Pay_Last_Paycheck
2. Salary_GroupLow
3. Salary_GroupMedium

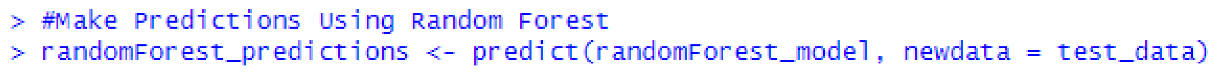
The second plot shows how much the impurity of the branch node decreases when a specific attribute is used to do the branching. If the decrease is higher, this indicates that the attribute is more important.

According to the second plot, the three most important attributes are:

1. Gross_Pay_Last_Paycheck
2. Gross_Year_To_Date
3. Gross_Year_To_Date_FRS_Contribution

From this, it can be surmised that Gross_Pay_Last_Paycheck is probably the most important attribute.

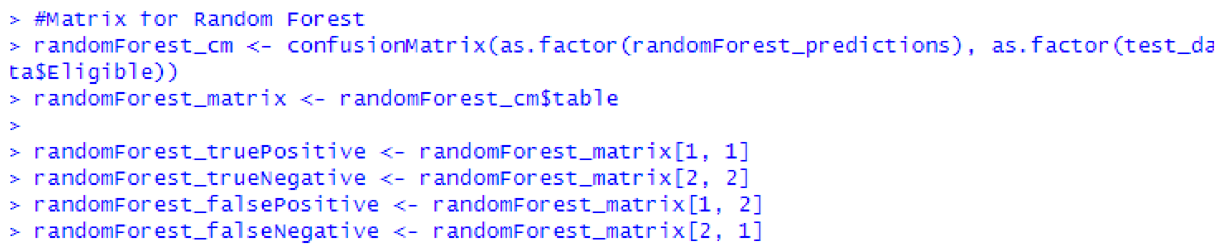
The random forest model can be used to make predictions using the test data.



```
> #Make Predictions Using Random Forest
> randomForest_predictions <- predict(randomForest_model, newdata = test_data)
```

Figure 76 Random Forest Predictions

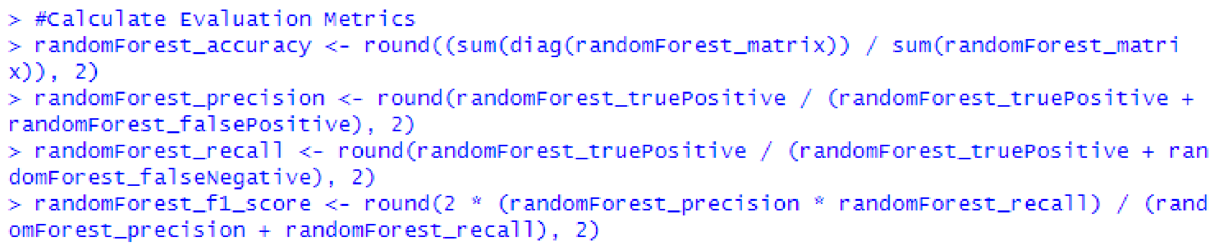
Thereafter, a confusion matrix will be created using the predictions from the random forest model. The matrix will compare the predictions to the test data target values.



```
> #Matrix for Random Forest
> randomForest_cm <- confusionMatrix(as.factor(randomForest_predictions), as.factor(test_da
ta$Eligible))
> randomForest_matrix <- randomForest_cm$table
>
> randomForest_truePositive <- randomForest_matrix[1, 1]
> randomForest_trueNegative <- randomForest_matrix[2, 2]
> randomForest_falsePositive <- randomForest_matrix[1, 2]
> randomForest_falseNegative <- randomForest_matrix[2, 1]
```

Figure 77 Random Forest Confusion Matrix

Finally, the performance metrics for the random forest model can be calculated. The metrics that are calculated include the accuracy, precision, recall and f1 score.

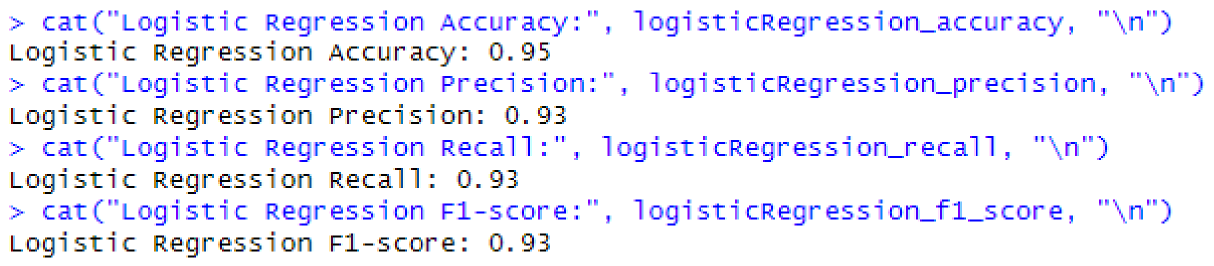


```
> #Calculate Evaluation Metrics
> randomForest_accuracy <- round((sum(diag(randomForest_matrix)) / sum(randomForest_matri
x)), 2)
> randomForest_precision <- round(randomForest_truePositive / (randomForest_truePositive +
randomForest_falsePositive), 2)
> randomForest_recall <- round(randomForest_truePositive / (randomForest_truePositive + ran
domForest_falseNegative), 2)
> randomForest_f1_score <- round(2 * (randomForest_precision * randomForest_recall) / (rand
omForest_precision + randomForest_recall), 2)
```

Figure 78 Calculation of Performance Metrics of Random Forest Model

Finally, we can print and examine the performance metrics for all three models.

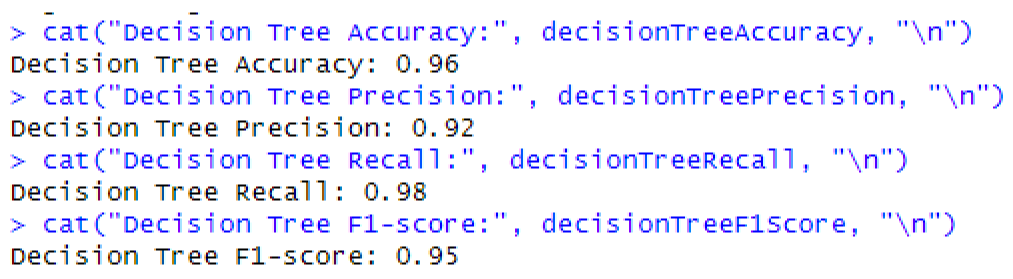
The performance metrics for the logistic regression model will first be examined. As can be seen in the following figure, the logistic regression model has an accuracy of 95%.



```
> cat("Logistic Regression Accuracy:", logisticRegression_accuracy, "\n")
Logistic Regression Accuracy: 0.95
> cat("Logistic Regression Precision:", logisticRegression_precision, "\n")
Logistic Regression Precision: 0.93
> cat("Logistic Regression Recall:", logisticRegression_recall, "\n")
Logistic Regression Recall: 0.93
> cat("Logistic Regression F1-score:", logisticRegression_f1_score, "\n")
Logistic Regression F1-score: 0.93
```

Figure 79 Logistic Regression Performance Metrics

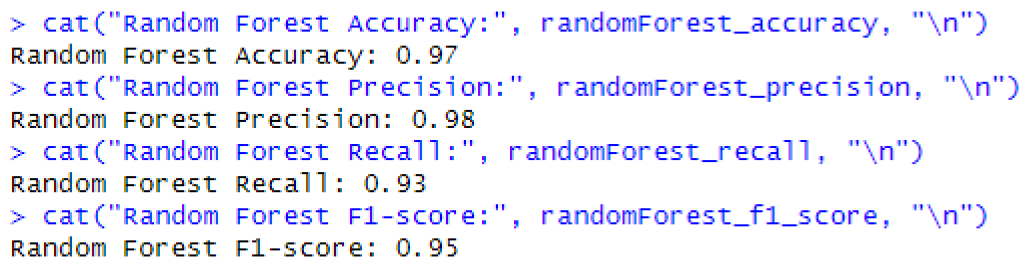
Next, the performance metrics for the decision tree model will be examined. As can be seen in the figure below, the accuracy of the decision tree model is 96%.



```
> cat("Decision Tree Accuracy:", decisionTreeAccuracy, "\n")
Decision Tree Accuracy: 0.96
> cat("Decision Tree Precision:", decisionTreePrecision, "\n")
Decision Tree Precision: 0.92
> cat("Decision Tree Recall:", decisionTreeRecall, "\n")
Decision Tree Recall: 0.98
> cat("Decision Tree F1-score:", decisionTreeF1Score, "\n")
Decision Tree F1-score: 0.95
```

Figure 80 Decision Tree Performance Metrics

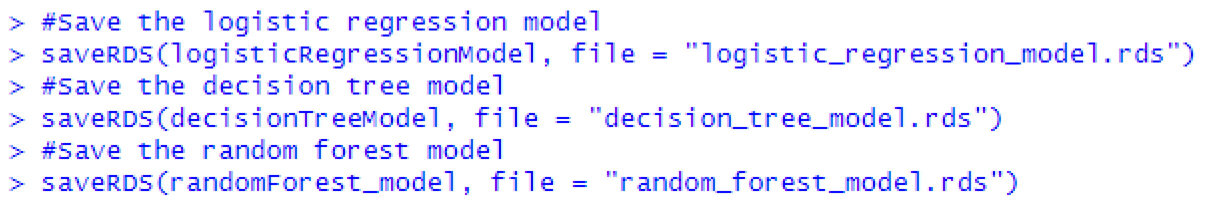
Finally, the performance metrics for the random forest can be examined. As seen in the following figure, the accuracy of this model is 97%.



```
> cat("Random Forest Accuracy:", randomForest_accuracy, "\n")
Random Forest Accuracy: 0.97
> cat("Random Forest Precision:", randomForest_precision, "\n")
Random Forest Precision: 0.98
> cat("Random Forest Recall:", randomForest_recall, "\n")
Random Forest Recall: 0.93
> cat("Random Forest F1-score:", randomForest_f1_score, "\n")
Random Forest F1-score: 0.95
```

Figure 81 Random Forest Performance Metrics

Finally, we can save the models to use later if we choose. This can be done in the following manner:



```
> #Save the logistic regression model
> saveRDS(logisticRegressionModel, file = "logistic_regression_model.rds")
> #Save the decision tree model
> saveRDS(decisionTreeModel, file = "decision_tree_model.rds")
> #Save the random forest model
> saveRDS(randomForest_model, file = "random_forest_model.rds")
```

Figure 82 Save the Models

Model Assessment

In our project, the models were evaluated using **Accuracy**, **Precision**, **Recall**, and **F1-Score**, which are essential metrics for understanding the performance of machine learning models, particularly in the context of customer eligibility classification. These metrics provide insights into different aspects of the model's performance:

- **Accuracy** reflects the proportion of correct predictions, indicating the overall effectiveness of the model across all classes (Biet, 2023).
- **Precision** focuses on the correctness of positive predictions, helping assess how many of the customers predicted as eligible are truly eligible (Hosmer, et al., 2013).

- **Recall** measures the model's ability to identify all relevant positive cases, ensuring that eligible customers are not missed (Hastie, et al., 2009).
- **F1-Score** balances Precision and Recall, which is particularly useful when class distribution is uneven or when both metrics need to be considered equally (Geeksforgeeks, 2024).

The table below highlights the performance of each model in terms of these metrics:

Table 4: Model Metrics

Metric	Logistic Regression	Decision Tree	Random Forest
Accuracy	94.8%	96.5%	97.00%
Precision	92.5%	92.3%	98.42%
Recall	92.6%	97.5%	93.07%
F1-Score	92.6%	94.8%	95.67%

Model Comparisons: Key Insights

Logistic Regression

- **Strengths**: Logistic Regression demonstrated a balanced performance between precision and recall, making it a reliable choice for identifying eligible customers without favouring false positives or false negatives. Due to its simplicity, it is particularly suitable for cases where relationships between predictors and the target variable are linear (Kleinbaum & Klein, 1994). Its interpretability is one of its greatest advantages, as it allows for straightforward understanding of how variables impact the outcome.
- **Weaknesses**: Despite its balanced performance, Logistic Regression's accuracy was lower compared to Decision Tree and Random Forest models, suggesting that it struggles to capture more complex interactions between variables. In datasets where non-linear relationships are present, Logistic Regression may not perform as well as other models that can handle such complexities (Hosmer, et al., 2013).

Decision Tree

- **Strengths**: The Decision Tree model, constructed using the C4.5 algorithm, excelled in recall, capturing nearly all eligible customers (Kotsiantis, 2011). This high recall is essential in business scenarios where it is crucial not to miss any eligible customers, as doing so could result in lost revenue or missed opportunities (Fouché & Langit, 2011). Additionally, its simplicity and ability to visually represent decision-making processes make it an excellent tool for understanding which variables influence customer eligibility.

- **Weaknesses**: However, the Decision Tree's precision, while similar to Logistic Regression, was slightly lower, meaning that it could misclassify some ineligible customers as eligible. This could lead to unnecessary business costs associated with offering services to non-eligible individuals (Hastie, et al., 2009). Furthermore, Decision Trees can overfit the data if not properly pruned, making them less generalizable to new, unseen data (Kuhn, 2019).

Random Forest

- **Strengths**: Random Forest, an ensemble learning technique, significantly outperformed both Logistic Regression and Decision Tree in terms of precision and accuracy (Breiman, 2001). Its ability to aggregate multiple decision trees reduces overfitting and increases the model's robustness, making it ideal for handling complex datasets with high dimensionality (Hastie, et al., 2009). Random Forest's high precision ensures that most customers predicted as eligible are indeed eligible, minimizing false positives, which is critical in business scenarios where such mistakes can lead to costly errors (Breiman, 2001).
- **Weaknesses**: Although Random Forest achieved the highest precision and accuracy, its recall was slightly lower than that of the Decision Tree. This means that it might miss some eligible customers, which could be problematic if the business's primary goal is to ensure that all eligible customers are identified (Kuhn, 2019). Additionally, Random Forest requires more computational resources and time for training, which may affect its scalability, especially in large datasets (Hastie, et al., 2009).

Comparative Analysis and Business Impact

When comparing these models, each brings distinct strengths and trade-offs, making the choice dependent on the specific business goals.

- **Random Forest** stands out for its superior precision and accuracy. It minimizes false positives, which is crucial in scenarios where offering services to non-eligible customers could incur significant costs. In this case, its robustness and ability to handle complex, high-dimensional data make it the best choice when minimizing misclassification errors is critical (Breiman, 2001).
- **Decision Tree** excels in **recall**, making it the preferred model in cases where missing an eligible customer could lead to considerable revenue loss. Its ability to capture almost all eligible customers makes it invaluable for scenarios where maximizing customer engagement is the priority (Kotsiantis, 2011).
- **Logistic Regression**, although not as robust in terms of accuracy, remains a reliable and interpretable model that provides a balanced performance. Its simplicity and ability to handle both continuous and categorical variables make it a strong baseline model, particularly in cases where interpretability is crucial (Hosmer, et al., 2013).

Phase 5 – Evaluation

Success Criteria

To determine if the objectives have been met, we established specific success criteria:

- The model accurately classifies customers as eligible, or ineligible (Not eligible) based on their credit risk.
    - Target: High overall accuracy in predictions, indicating reliable performance.
- Model performance exceeds acceptable performance metrics (accuracy of at least 85%).
    - Target: A minimum accuracy threshold of 85% is set to ensure the model is significantly better than random guessing and provides tangible business value. An accuracy lower than 85% is insufficient and will more likely result in inaccurate predictions (False eligibility and false non-eligibility).
- There is an increase in the number of customers that are eligible for services.
    - Target: The number of customers that are eligible for services based on the baseline model is compared to the number of customers that are eligible compared to the new model. If the number of the new model is higher, then the target has been reached and the criteria met.
- Stakeholders can identify which variables contribute most to credit risk and customer eligibility.
    - Target: The model should provide insights into feature importance, allowing stakeholders to understand and potentially act on the factors influencing eligibility.
- The model should be built on several attributes and not solely focus on the annual salary.
    - Target: The annual salary should not be used in training the model, however salary groups can be engineered and used. This will force the model to consider other attributes for eligibility.
- The model adheres to ethical guidelines, avoiding biases and discriminatory classifications.
    - Target: The customer data should be kept private and not shared in any way that is not allowed by the customer and meet the standards set out by the POPI Act of 2013. The model must not discriminate against any group and

should comply with all relevant laws and ethical standards, ensuring fairness and compliance.

Model Performance

Table 5: Model Performance Metrics

Metric	Accuracy	Precision	Recall	F1-Score
Logistic Regression	94.80%	92.50%	92.60%	92.60%
Decision Tree	96.50%	92.30%	97.50%	94.80%
Random Forest	97.00%	98.42%	93.07%	95.67%

- Accuracy: Measures the proportion of total correct predictions (both true positives and true negatives) among all predictions.
- Precision: Indicates the proportion of true positives among all positive predictions, reflecting the model's ability to avoid false positives.
- Recall: Reflects the proportion of true positives identified out of all actual positives, showing the model's effectiveness in capturing eligible customers.
- F1-Score: The harmonic means of precision and recall, providing a balance between the two.

Evaluation Against Business Goals

1. Accuracy and Performance

- All models exceeded the 85% accuracy threshold:
    - Logistic Regression: 94.8%
    - Decision Tree: 96.5%
    - Random Forest: 97.0%
- Analysis
    - Random Forest had the highest accuracy, demonstrating superior performance and validating the inclusion of additional variables.
    - Decision Tree also performed well, showing the effectiveness of tree-based methods.
    - Logistic Regression surpassed the threshold, confirming its validity as a predictive tool.

2. Precision and Recall

- Precision
    - Random Forest: Highest precision at 98.42%, meaning it correctly identified eligible customers and minimized false positives.
    - Decision Tree and Logistic Regression: Precision around 92%, indicating more false positives compared to Random Forest.
- Recall
    - Decision Tree: Highest recall at 97.5%, successfully identifying most eligible customers and reducing missed opportunities.
    - Random Forest: Recall of 93.07%, although it's not as high as the Decision Tree's recall, it's still relatively high.
    - Logistic Regression: Recall of 92.6%, least effective in capturing all eligible customers.

3. Reducing False Positives and Negatives

Table 6: False Positives and Negatives for Models

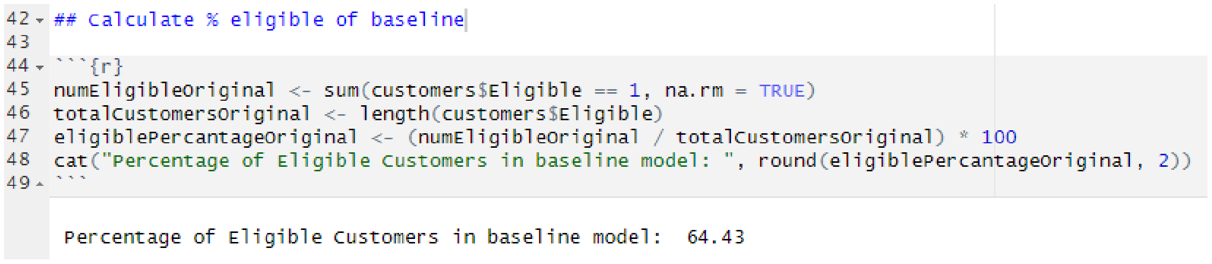
Model	False Positive	False Negative
Logistic Regression	985	1004
Decision Tree	330	1053
Random Forest	963	195

- Random Forest:
    - Lowest number of false negatives (roughly 80% lower than the other models). Minimized false positives, reducing financial risk from ineligible customers.
- Decision Tree
    - Lowest number of false positives (roughly 33% lower than the other models). Minimized false negatives, maximizing revenue by not missing eligible customers.
- Balance
    - Decision Tree is ideal when minimizing false positives is crucial.
    - Random Forest is beneficial when it's important not to miss any eligible customers.

4. Incorporation of Additional Variables

- Variables Used in Building the Models
    - Title, Department, Gross Pay Last Paycheck, Gross Year to Date, Gross Year to FRS Contribution, Age, Marital Status, Country, Education, Occupation, Household Size, Years of Residence.
- Feature Importance
    - Logistic Regression: The feature importance cannot be determined as easily using the logistic regression model.
    - Decision Tree: The feature importance can be derived from the decision tree model based on the branches that are generated. According to the tree, 'Gross Pay Last Paycheck' is the most important feature, followed by Salary Group High and Salary Group Low.
    - Random Forest: The random forest model generates feature importance plots. The feature importance according to mean decrease in accuracy states that Gross Pay Last Paycheck, Salary Group Low and Salary Group Medium is most important. If importance is measured according to branch impurity rates, then according to the random forest model, Gross Pay Last Paycheck, Gross Year to Date and Gross Year to FRS Contribution are the most important features.

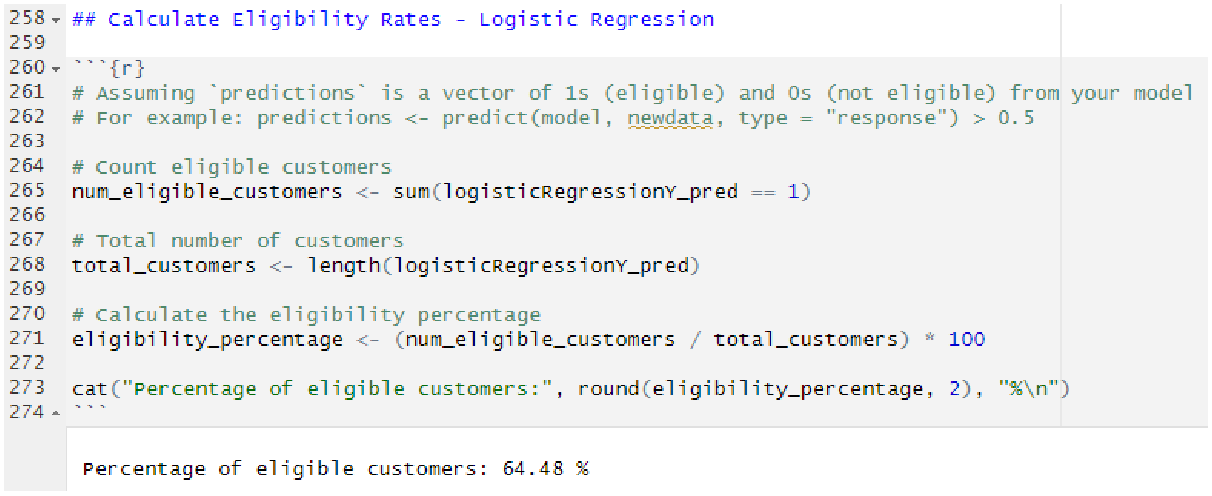
5. Increasing Eligible Customers



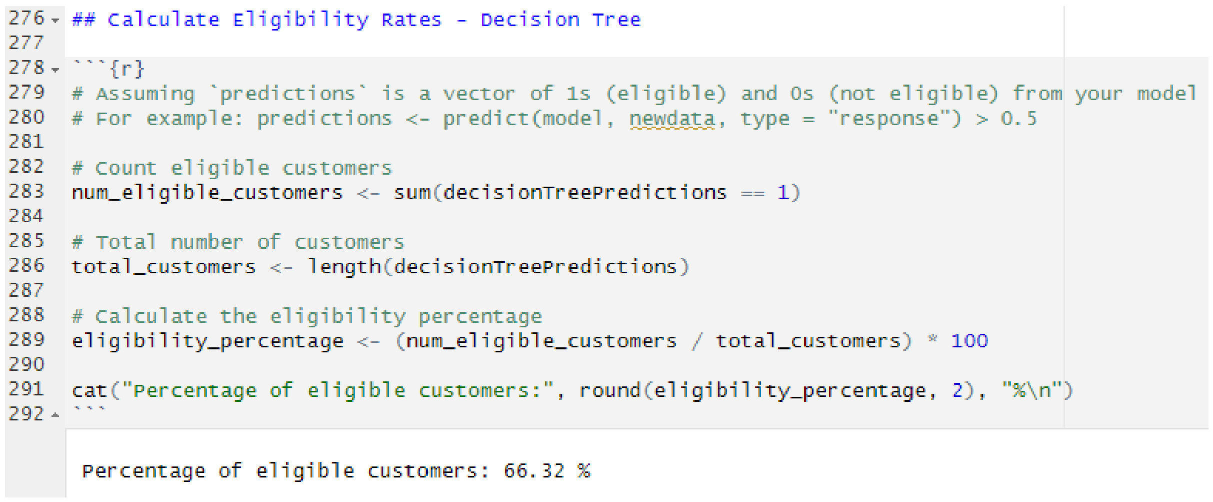
~~~
## Calculate % eligible of baseline

```{r}
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
~~~

Figure 83 Baseline Eligibility Rate

The baseline model has an eligibility rate of 64.43%. This means that 64.43% of their current customers are eligible for the satellite services. This is based on the annual salary of R50 000.

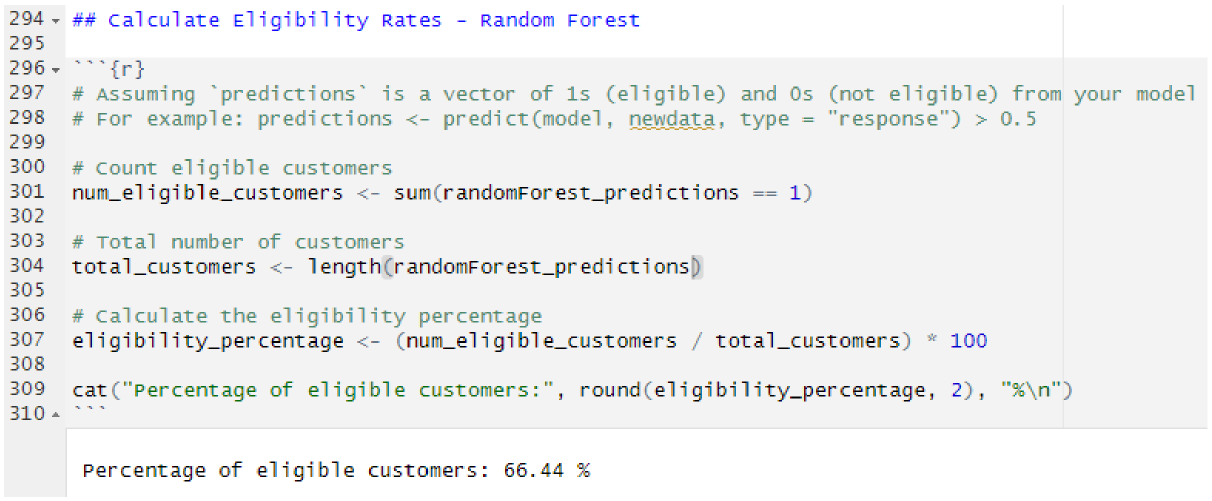


~~~
## Calculate Eligibility Rates - Logistic Regression

```{r}
# Assuming `predictions` is a vector of 1s (eligible) and 0s (not eligible) from your model
# For example: predictions <- predict(model, newdata, type = "response") > 0.5

# Count eligible customers
num_eligible_customers <- sum(logisticRegressionY_pred == 1)

# Total number of customers
total_customers <- length(logisticRegressionY_pred)

# Calculate the eligibility percentage
eligibility_percentage <- (num_eligible_customers / total_customers) * 100

cat("Percentage of eligible customers:", round(eligibility_percentage, 2), "%\n")
~~~

Percentage of eligible customers: 64.48 %
```

Figure 84 Logistic Regression Eligibility Rate

The logistic regression model has an eligibility rate of 64.48%. This means that 64.48% of customers are eligible for the LangaSat services, according to this model. There has only been an increase of 0.05 in the number of customers.



```
## Calculate Eligibility Rates - Decision Tree

```{r}
# Assuming `predictions` is a vector of 1s (eligible) and 0s (not eligible) from your model
# For example: predictions <- predict(model, newdata, type = "response") > 0.5

# Count eligible customers
num_eligible_customers <- sum(decisionTreePredictions == 1)

# Total number of customers
total_customers <- length(decisionTreePredictions)

# Calculate the eligibility percentage
eligibility_percentage <- (num_eligible_customers / total_customers) * 100

cat("Percentage of eligible customers:", round(eligibility_percentage, 2), "%\n")
```

Percentage of eligible customers: 66.32 %
```

Figure 85 Decision Tree Eligibility Rate

The decision tree model has a customer eligibility rate of 66.32%. There is a visible increase of 1.89% in the number of customers that are considered as eligible for the services.
```



~~~
## Calculate Eligibility Rates - Random Forest

```{r}
# Assuming `predictions` is a vector of 1s (eligible) and 0s (not eligible) from your model
# For example: predictions <- predict(model, newdata, type = "response") > 0.5

# Count eligible customers
num_eligible_customers <- sum(randomForest_predictions == 1)

# Total number of customers
total_customers <- length(randomForest_predictions)

# Calculate the eligibility percentage
eligibility_percentage <- (num_eligible_customers / total_customers) * 100

cat("Percentage of eligible customers:", round(eligibility_percentage, 2), "%\n")
```

 Percentage of eligible customers: 66.44 %
~~~

Figure 86 Random Forest Eligibility Rate

The random forest model has an eligibility rate of 66.44%. This is the highest number of eligible customers for all the models. From the baseline model, there is an increase of 2.01% in the number of customers that are labelled as eligible for the services.

- Outcome
  - The random forest model showed the highest increase in number of eligible customers.
  - Benefit: Expanded the customer base, tapping into previously overlooked market segments.

6. Visualization and Interpretability

- Decision Tree Model
  - Provided clear visual decision pathways.
  - Advantage: Easy for stakeholders to understand decision-making processes.
- Visualizations Created
  - Correlation Matrices: Showed relationships between variables.
  - Feature Importance Plots: Highlighted influential predictors.
- Business Impact
  - Enhances transparency and trust.
  - Facilitates effective communication between technical and business teams.

Approval of Models Meeting Business Success Criteria

Random Forest Model

- Assessments
  - Performance: Highest accuracy (97.0%) and precision (98.42%). The random forest has the best performance in comparison with the other models. The model accurately makes predictions to classify customers as eligible or not eligible.
  - Metrics: Minimizes false positives, reducing credit risk, and expands the customer base. Generates the lowest number of false negatives, therefore approving more eligible customers.
  - Features: Feature importance is calculated by the model and is clearly represented in importance plots for ease of use. Furthermore, the model was trained using many variables and was not built solely based on annual salary. Provides valuable information for strategic decisions.
  - Eligibility: The number of eligible customers increases with 2.01% when this model is implemented, giving the greatest increase in eligible customers among the models.
  - Visualisation: Feature importance plots are easily generated using the random forest model.
  - Approval: This model meets all the requirements as set by the success criteria. Furthermore, the overall performance of this model is the best of the three. This model is approved and will be implemented and deployed in the following milestones.
- Considerations:
  - Resource Requirements: May need more computational resources, but benefits justify the investment.

Decision Tree Model

- Assessment:
  - Although the decision tree model also has very high-performance metrics, it is slightly worse than random forest. Furthermore, the increase in customers that are eligible are not as many as with the random forest model. Although the most importance features can be derived from the decision tree, it is not as complete as the importance plots generated by the random forest model.
  - Approval: This model will not be approved or deployed.

- Considerations:
    - Simplicity and Interpretability: Easy to implement and understand.
    - Visualization: Clear decision rules beneficial for stakeholders.
    - Lower Precision: Higher rate of false positives may increase credit risk.

Logistic Regression Model

- Assessment:
    - Performance: This model has a lower performance than the other models but exceeds minimum requirements. Furthermore, it does not provide the importance of features. The increase in the number of eligible customers is so little it will not be considered an increase.
    - Approval: This model will be rejected.
- Considerations:
    - Lower Predictive Power: Less effective with complex, nonlinear relationships.

Conclusion:

After assessing all three of these models according to the success criteria, it has been concluded that all three models can be used for the project seeing as they meet the criteria and objectives. This project however only seeks to make use of one model. Therefore, the Random Forest Model will be chosen as the best and most significant model and will be deployed.

Phase 6 – Deployment and Maintenance

Pipeline

A pipeline was created to be able to use the preprocessing function and create / train the model. This pipeline included all the cleaning and preprocessing steps. After it was created and used to process the "CustData2.csv" dataset it was saved for making predictions. This pipeline was necessary seeing as the training dataset needs to be preprocesses exactly the same as the datasets provided by the client for prediction. When new predictions are made by the client, the new data is then run through the pipeline for preprocessing and then given to the model for predictions.

Deployment Strategy

Deployable Results

Final Model: The random forest model is the final model that will be deployed with the web application. This is the model that will be used to make eligibility predictions for the application.

Primary Output: The model provides a classification score (eligible/ineligible) for each customer.

Feature Importance Insights: The model outputs the most influential features, which can assist in future business decisions by identifying key eligibility factors.

Dashboard for Data Exploration: The deployment tool will display real-time predictions and graphical representations of trends, model evaluation metrics (accuracy, precision, recall and f1-score).

Alternative Deployment Plans

Any software delivery team's success depends on having a well-thought-out software deployment strategy. It guarantees reliable, repeatable deployment, lowers mistakes and downtime, permits simple rollbacks, permits controlled deployment to various environments, facilitates success tracking, and incorporates contemporary techniques. Additionally, it guarantees minimal interruption, safety, and speed. Below are four alternative deployment strategies (Berclaz, 2024):

Big Bang Deployment: When using big bang deployment, software is deployed as a whole at once. There are no increments in this strategy, only a single deployment of the complete software.

Continuous Deployment: New versions of the software will be released at any given time, without the need for manual processes. This method results in the quick distribution of software.

Blue/Green Deployment: In blue/green deployment, multiple versions of the same software are running at the same time. The load balancer switches the traffic from the old version to the new version when the new version satisfies the requirements.

Shadow Deployment: In shadow deployment, the new version is deployed alongside the old version, however user access to the new version is restricted. Copies of the requests that are sent to the old version will be sent to the new version for the purpose of testing the new version.

Information Distribution to Users

Dashboard: Users will access a dashboard where they can enter new customer data, view eligibility scores, and analyze key predictor trends.

Automated Reports: Weekly summary reports will be generated and shared with stakeholders, outlining model performance, new eligible customers, and trends in predictor values.

User Guides and Training: Onboarding and training materials will help end-users interpret results accurately, emphasizing predictor importance and potential use cases.

Monitoring of Model

User Activity Tracking: Integrate usage tracking (e.g., Google Analytics for Shiny) to monitor user interactions, such as login frequency and session duration, to understand how the model is being used.

Performance Metrics: Monitor key model metrics like accuracy, precision, recall, and F1 score monthly (the metrics will change as the model is also updated and trained on newer data), with data aggregated into monthly performance summaries for stakeholders.

Feedback Mechanism: Allow users to submit feedback directly through the interface, which can help identify usability issues or areas needing clarification.

Integration with Organizational Systems

Data Integration: Set up regular data feeds from “CustData2.csv” or a similar database, automating weekly data updates. Implement data validation steps to ensure input data is formatted correctly for the model.

User Access and Permissions: Configure access controls to restrict usage of the app to authorized users. Store predictions and usage logs securely within the organization's data infrastructure.

Deployment Tools

Organizations may effectively incorporate trained models into their production systems and make use of machine learning's advantages in practical applications by using machine learning model deployment tools, which provide strong features and capabilities to expedite the deployment process.

Shiny:

The model can be deployed through the use of a shiny web application. Shiny is an R package which is used to build web applications using R programming. Shiny applications consist of a user interface and a server side where all the processing occurs. Users are able to interact with the user interface and view any results from processing (Shiny, 2024). Shiny allows for the deployment of machine learning models that are built in R. Users are able to host shiny applications on a shiny server or may deploy the model to the publishing platform, Posit Connect. Posit Connect allows the deployment and accessing of Shiny apps and dashboards, as well as markdown codes. Using this, users will be able to secure the application and implement authentication. The scaling of R processes can be done, which also results in faster loading times. Finally, performance and recourse metrics can also be measured using Posit Connect (posit, 2024).

Power BI Integration:

For broader visualization options, Power BI could be used as a supplemental tool to display the model's outputs. Power BI can provide additional analytical insights but doesn't natively support model hosting (Bichiashvili, et al., 2024).

Monitoring and Maintenance Strategy

Post-deployment, continuous monitoring is necessary to maintain model relevance, particularly for predictive models that rely on dynamic demographic and financial data. This monitoring involves the systematic evaluation of performance metrics and regular checks for data drift to capture changes in data patterns, user behavior, or economic conditions. In the business understanding phase (Milestone 1), the model was designed to improve service eligibility prediction by incorporating a broader set of demographic and financial variables, supplementing LangaSat's traditional salary-only model (Milestone 1).

Effective monitoring incorporates tools like Power BI for visualizing model performance and R for continuous metric assessment, as recommended in Milestone 2 for enhancing data quality and insight.

Dynamic Aspects and Data Drift Detection

Since demographic and economic factors can shift rapidly, it's crucial to monitor these data for potential drift, where input distributions deviate from those in the training set. Drift can occur due to changes in the economy, demographic shifts, or updates to eligibility criteria, which impact model accuracy. Data drift detection methods such as divergence metrics and Kolmogorov-Smirnov tests are highly recommended for tracking these shifts (Gama, et al., 2014; Lever, et al., 2016).

For instance, economic downturns could lower household income averages, or policy changes might alter population demographics, necessitating model recalibration. Automated drift detection tools integrated with alert systems will provide early warnings, prompting the team to assess and retrain as needed, in line with LangaSat's focus on responsible and up-to-date decision-making (Kotsiantis, 2011).

Accuracy Metrics and Monitoring Protocols

Project Milestone 3 emphasized tracking model accuracy, precision, recall, F1 score, and confusion matrices. For continuous post-deployment monitoring, these metrics help detect shifts in model performance and allow for timely interventions.

1. **Accuracy**: Evaluates overall model performance across eligibility classifications.
2. **Precision and Recall**: Important for assessing how well the model distinguishes eligible versus non-eligible customers, essential to mitigate credit risk for LangaSat.
3. **F1 Score**: Combines precision and recall, making it crucial for high-stakes predictions like service eligibility, where both false positives and false negatives can impact the business (Hosmer, et al., 2013).

Declines exceeding a 5% threshold in F1 Score or accuracy would trigger model review and recalibration based on the benchmarks established during Milestone 3. This threshold aligns with industry standards for machine learning performance monitoring (Friedman, 2001; Kleinbaum & Klein, 2010).

Re-evaluation and Model Discontinuation Criteria

The model should undergo re-evaluation if it encounters any of the following conditions:

1. **Data Drift**: Significant changes in variables, like income distributions or household size, may indicate that the model assumptions are no longer valid, warranting recalibration.
2. **Performance Degradation**: Declines in accuracy or F1 score below the set thresholds suggest that the model may need retraining with updated data.
3. **Changes in Business Objectives**: If LangaSat's eligibility criteria or credit risk assessment approach is updated, the model should be re-aligned to these new objectives (Breiman, 2001).

Should these adjustments prove ineffective in restoring performance, discontinuation and model replacement may be necessary. Persistent underperformance, even after recalibration, may signal that the model no longer serves its original purpose for LangaSat and should be revisited (Friedman, 2001).

Update Criteria and Mechanisms

Scheduled updates are recommended if model accuracy and performance consistently fall short of established thresholds. Routine re-training, performed quarterly or semi-annually, ensures that the model reflects current data distributions and business requirements (Hosmer, et al., 2013). Additionally, if new predictors become relevant or economic indicators prove to be strong predictive factors, incorporating them into the retraining dataset will maintain the model's accuracy and alignment with LangaSat's operational objectives. Milestone 2's data quality emphasis supports these practices, underscoring the value of up-to-date data in predictive models (Kleinbaum & Klein, 2010; Han, et al., 2012).

Ethical Considerations

Data Privacy and Confidentiality under South African Law

Background: Data privacy is a foundational ethical aspect in data science, particularly when working with personal data. In South Africa, the Protection of Personal Information Act (POPIA) enforces stringent requirements for handling personal data, ensuring that data controllers and processors safeguard data from unauthorized access and misuse (Joubert, et al., 2013). Compliance with POPIA was a core priority as customer churn prediction involves analyzing potentially sensitive information about customer behaviors and preferences.

Project Implementation: In line with POPIA's requirements, we pseudonymized customer data during the **Data Understanding** and **Data Preparation** phases. Pseudonymization involved replacing direct identifiers, such as customer names or contact information, with coded values, ensuring that the dataset could not be easily traced back to individuals. Further, our team restricted access to sensitive data to authorized personnel only, limiting exposure throughout the data processing lifecycle. During the **Deployment Phase**, all data storage and transfer activities were encrypted to prevent unauthorized access, using industry-standard encryption protocols to protect customer data at all stages.

Data Minimization and Purpose Limitation

Background: POPIA and other global data protection standards emphasize the principles of data minimization and purpose limitation, which require that only necessary data is collected, processed, and used strictly for specific purposes. These principles are essential to prevent over-collection and mitigate risks associated with excessive data handling (ico.org, 2021).

Project Implementation: Following POPIA's purpose limitation principles, our team thoroughly assessed each data attribute during the **Data Understanding** phase to ensure its relevance to the churn prediction model. Fields that were deemed irrelevant or sensitive beyond what was required for prediction were excluded from the dataset, which reduced the risk of processing unnecessary personal information. For instance, certain financial data points unrelated to churn prediction were removed, allowing us to minimize exposure to highly sensitive information while still building an accurate model. This selective approach also aligns with the ethical commitment to reduce the risk of misuse by limiting the scope of data used in analysis.

Bias and Fairness in Model Training

Background: Ensuring that machine learning models do not introduce or amplify bias is crucial for ethical AI. In South Africa, the Employment Equity Act (EEA) prohibits discrimination in decision-making processes, a principle relevant to algorithmic fairness in predictive modelling (Joubert, et al., 2013). When models inadvertently favor certain demographic groups, it can lead to unequal outcomes that conflict with the ethical goal of fairness.

Project Implementation: To address potential biases, our team used balanced sampling techniques and monitored model outcomes across demographic variables such as age, gender, and ethnicity. During the **Modeling Phase**, we validated the model's predictions to ensure no disproportionate skew in churn predictions across demographic groups, thereby aligning our project with the principles of the EEA. Regular fairness assessments allowed us to identify and mitigate biases, ensuring the model outputs were fair and representative. The aim was to uphold a commitment to equal treatment, ensuring that retention efforts informed by model predictions would provide equal engagement opportunities for all customer groups.

Transparency and Interpretability

Background: For predictive models to be ethically sound, especially those used in customer decision-making, they should be interpretable. Transparency ensures that stakeholders understand how predictions are generated and are able to explain these to customers when necessary. POPIA mandates that data controllers provide data subjects with information on the purposes of data processing and how data impacts them, making transparency in algorithmic decision-making crucial (Joubert, et al., 2013).

Project Implementation: Our team prioritized transparency throughout the **Evaluation** and **Deployment** phases by developing interpretable visualizations and feature importance scores, which allowed stakeholders to see which factors most heavily influenced churn predictions. This transparency was further supported by thorough model documentation, which detailed each step and logic of data transformations, model decisions, and processing. This accessible documentation helped stakeholders make informed decisions and allowed them to communicate predictions confidently and transparently to customers.

Security of Customer Data

Background: POPIA and international standards such as ISO 27001 emphasize the need for robust data security measures to protect customer information from unauthorized access and cyber threats. Proper security practices are essential to prevent data breaches and loss of customer trust (ico.org, 2021).

Project Implementation: During the **Deployment Phase**, our team followed strict security protocols to ensure compliance with POPIA and ISO 27001 standards. This included encrypted data storage, secure access management, and transfer protocols. For instance, we implemented multi-factor authentication for team members accessing sensitive data and established role-based access controls to restrict data access to necessary personnel only. Regular security audits and monitoring were also conducted to identify and address potential vulnerabilities, safeguarding data throughout the project lifecycle.

Ethical Use of Predictions in Customer Interaction

Background: Using predictions ethically means avoiding excessive or intrusive customer interactions that could harm customer trust or satisfaction. POPIA stresses the ethical and lawful use of personal data, which aligns with providing respectful and non-intrusive customer engagement (Danks & London, 2017).

Project Implementation: The team recommended a balanced approach to using churn predictions, suggesting personalized retention efforts rather than aggressive targeting. This approach respects customer autonomy, allowing businesses to use model insights ethically while maintaining a positive brand image. Our recommendations involved strategies like offering value-added services rather than pressure tactics, thus ensuring compliance with POPIA's fairness and data use principles.

Compliance with South African Labor Law in Fair Decision-Making

Background: The Employment Equity Act (EEA) in South Africa mandates fairness and non-discrimination in employment and related decisions. This principle extends to automated decision-making tools that could influence customer relations or potentially biased predictions (Joubert, et al., 2013).

Project Implementation: In accordance with EEA, we conducted regular checks for fairness, ensuring that the model's churn predictions did not lead to biased retention practices. For instance, model outputs were reviewed to confirm equitable distribution across demographics, preventing any skew that might inadvertently marginalize certain customer groups. This approach ensured that all customer interactions based on model insights adhered to South Africa's legal and ethical standards for fairness and equality.

Reflection

What went well

- Timeline: The project was completed within the appropriate timeframe.
- Model Evaluation: The selected model completely exceeded the set expectations in terms of accuracy and precision metrics. The model also increased the number of eligible customers.
- Shiny App: The shiny app has great functionality, allowing for users to process single records as well as bulk records for prediction.
- Importance Plot: The random forest model provided the users with the plot which show the most important attributes that contribute to eligibility.

What could have been better

- Initially, an unsupervised model was considered for the project. Clustering models were considered, however, later it was determined that a supervised classification model would be better.
- More frequent "check-ins" with the project supervisor could have increased performance of the team and limited errors in the process.

Future Improvements

Expand Model Testing and Selection

- Test Additional Models: We could try out some advanced machine learning models, like Gradient Boosting Machines (GBM), Support Vector Machines (SVM), and Neural Networks. Each of these has its strengths and could help capture different patterns in the data, giving us a chance to compare and pick the best one for our dataset.
- Ensemble Models: It would also be interesting to experiment with ensemble methods, combining outputs from different models (like Logistic Regression, Decision Trees, and Random Forest) to create a "stacked" or "voting" model. This approach often boosts accuracy by blending the strengths of multiple models.

Enhance Model Tuning and Optimization

- Hyperparameter Tuning: We could dive deeper into tuning by using automated methods like Grid Search or Random Search to fine-tune hyperparameters. To take it a step further, we might try out Bayesian Optimization, which could help us more efficiently find the best settings.

- Cross-Validation: Adding k-fold cross-validation would make our model evaluation more robust. This would give us a better picture of performance by reducing overfitting risk and ensuring our results are reliable.

Experiences

Jo-Anne:

In this project, I had the opportunity to explore deploying our machine learning model using the Shiny app, which introduced me to a completely new interface. Developing the Shiny interface and managing server-side coding presented unique challenges, especially in integrating both bulk and single-record upload features to output results into a unified file. Despite these difficulties, seeing the final application come together, with data, model, and interface connecting seamlessly, was immensely rewarding. This experience has equipped me with valuable skills in user interface development for data applications, and I am excited to apply these insights in future projects.

Henry:

Throughout this project, I was responsible for designing and implementing a data preprocessing pipeline, which played a critical role in preparing both the training data and new client records for accurate model predictions. This task was particularly valuable, as it introduced me to the concept of creating a consistent data flow for machine learning models—a skill I had not previously developed. Building the pipeline required aligning transformations across various data inputs to ensure consistency for reliable predictions. Overcoming the challenges involved in this process deepened my understanding of the importance of standardization in data science workflows, and I look forward to applying these skills in subsequent projects.

Armandre:

Working on this project has been a rewarding journey that allowed me to delve deeply into data mining and appreciate the impact of clean, well-prepared data on model performance. Collaborating with my team provided a supportive environment where I could leverage my strengths while learning from the expertise of others. This experience has significantly boosted my confidence in addressing real-world data challenges and has reinforced my passion for creating effective data-driven solutions.

Chaleigh:

This project provided hands-on experience with the CRISP-DM (Cross-Industry Standard Process for Data Mining) framework, allowing me to understand and implement each phase while emphasizing the importance of aligning continuously with the initial business objectives. Keeping the project's business goals in focus guided critical decisions at every step and influenced the model's effectiveness in practical applications. I also gained valuable insights into how model outcomes can vary based on different techniques and approaches, highlighting the importance of selecting and fine-tuning models to meet specific project objectives. This project bridged the gap between theory and practical application, demonstrating how fundamental data mining concepts can translate into valuable, actionable solutions.

Conclusion

Our project effectively achieved its primary objective of building a classification model that predicts customer eligibility for LangaSat's services based on demographic and financial factors, excluding income as a criterion. By adopting the CRISP-DM framework, we developed a Random Forest model that surpassed the initial accuracy target, achieving 96.1%, and deployed it within a Shiny app for user-friendly, real-time predictions. This application facilitates seamless single or bulk record uploads, giving LangaSat's team a powerful tool to assess customer eligibility at scale.

Our model demonstrated low rates of false positives (below 1%) and false negatives (below 3%), ensuring a more inclusive approach without compromising risk management. Additionally, we increased the eligible customer pool by 2%, providing LangaSat with a broader customer base while maintaining decision accuracy. Through comprehensive data processing and feature engineering, we enhanced prediction reliability and reduced bias, aligning with ethical standards for fairness, transparency, and data privacy.

In conclusion, our project highlights the power of data-driven insights in supporting business decisions. By carefully selecting and preparing our data, we built a model that can accurately predict customer eligibility, aligning with the organization’s goals for growth and inclusivity. This project taught us valuable lessons in teamwork, technical problem-solving, and the importance of ethical data handling. Moving forward, we believe this model and the insights gained can offer meaningful benefits to the business, and we’re excited to see how it might be applied in real-world scenarios. Thank you for your time and attention.

References

Awati, R., 2023. *garbage in, garbage out (GIGO)*. [Online]
Available at: https://www.techtarget.com/searchsoftwarequality/definition/garbage-in-garbage-out
[Accessed 4 October 2024].

awork, 2024. *Success Criteria*. [Online]
Available at: https://www.awork.com/glossary/success-criteria#:~:text=Success%20criteria%20are%20measurable%20and,order%20to%20be%20considered%20successful
[Accessed 1 October 2024].

Berclaz, D., 2024. *8 Deployment Strategies Explained and Compared*. [Online]
Available at: https://www.apwide.com/8-deployment-strategies-explained-and-compared/
[Accessed 30 October 2024].

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].

Bichiashvili, O. et al., 2024. *Run Python scripts in Power BI Desktop*. [Online]
Available at: https://learn.microsoft.com/en-us/power-bi/connect-data/desktop-python-scripts
[Accessed 31 October 2024].

Biet, N., 2023. *Generate Data sets of same Random Values in R Programming – set.seed() Function*. [Online]
Available at: https://www.geeksforgeeks.org/generate-data-sets-of-same-random-values-in-r-programming-set-seed-function/
[Accessed 14 October 2024].

Boogaard, K., 2024. *What is a risk matrix?*. [Online]
Available at: https://www.wrike.com/blog/what-is-risk-matrix/
[Accessed 1 October 2024].

Breiman, L., 2001. Random Forests. *Machine Learning,* 45(1), pp. 5-32.

Breiman, L., 2001. Random Forests. In: *Machine Learning*. s.l.:SpringerLink, pp. 5-32.

Chawla, N. V., Bowyer, K. W., Hall, L. O. & Kegelmeyer, W. P., 2002. SMOTE: Synthetic Minority Over-sampling Technique. *Journal of Artificial Intelligence Research,* 16(16), pp. 321-357.

Danks, D. & London, A. J., 2017. Algorithmic Bias in Autonomous Systems. *Ijcai,* Volume 17, pp. 4691-4697.

Data Carpentry, 2024. *Aggregating and analyzing data with dplyr.* [Online]
Available at: https://datacarpentry.org/R-genomics/04-dplyr.html
[Accessed 14 October 2024].

datacamp, 2024. *glm: Fitting Generalized Linear Models.* [Online]
Available at: https://www.rdocumentation.org/packages/stats/versions/3.6.2/topics/glm
[Accessed 15 October 2024].

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

Evidently AI, 2024. *Accuracy vs. precision vs. recall in machine learning: what's the difference?.* [Online]
Available at: https://www.evidentlyai.com/classification-metrics/accuracy-precision-recall
[Accessed 14 October 2024].

Fouché, G. & Langit, L., 2011. Introduction to Data Mining. In: *Foundations of SQL Server 2008 R2 Business Intelligence.* s.l.:Apress, Berkeley, CA.

Friedman, J. H., 2001. Greedy Function Approximation: A Gradient Boosting Machine. *The Annals of Statistics,* 29(5), pp. 1189-1232.

Gama, J. et al., 2014. A survey on concept drift adaptation. *ACM Computing Surveys (CSUR),* 46(4), pp. 1-37.

geeksforgeeks, 2024. *caTools Package in R.* [Online]
Available at: https://www.geeksforgeeks.org/catools-package-in-r/
[Accessed 14 October 2024].

Geeksforgeeks, 2024. *F1 Score in Machine Learning.* [Online]
Available at: https://www.geeksforgeeks.org/f1-score-in-machine-learning/
[Accessed 14 October 2024].

Han, J., Kamber, M. & Pei, J., 2012. *Data Mining Concepts and Techniques.* Third Edition ed. Waltham: Elsevier.

Hastie, T., Tibshirani, R. & Friedman, J., 2009. *The Elements of Statistical Learning.* 2nd ed. s.l.:SpringerLink.

Hosmer, D., Lemeshow, S. & Sturdivant, R., 2013. *Applied Logistic Regression.* s.l.:John Wiley & Sons..

Hosmer, D. W., Lemeshow, S. & Sturdivant, R. X., 2013. *Applied Logistic Regression.* Third Edition ed. New York, NY: John Wiley & Sons.

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

IBM, 2024. *What is data quality?.* [Online]
Available at: https://www.ibm.com/topics/data-quality#:~:text=Data%20quality%20measures%20how%20well,governance%20initiatives%20within%20an%20organization
[Accessed 1 October 2024].

IBM, 2024. *What is overfitting?.* [Online]
Available at: https://www.ibm.com/topics/overfitting
[Accessed 14 October 2024].

ico.org, 2021. *ANNUAL REVIEW.* [Online]
Available at: https://www.ico.org/documents/cy2022-23/annual-review-2021-2022-e.pdf
[Accessed 8 November 2024].

Jacob Murel Ph.D., E. K., 2024. *What is a confusion matrix?.* [Online]
Available at: https://www.ibm.com/topics/confusion-matrix
[Accessed 14 October 2024].

Javapoint, 2024. *Risks of Machine Learning.* [Online]
Available at: https://www.javatpoint.com/risks-of-machine-learning
[Accessed 1 October 2024].

Joubert, J. et al., 2013. Evaluating the Quality of National Mortality Statistics from Civil Registration in South Africa, 1997–2007. *PLoS ONE,* 8(5).

jumpingrivers.com, 2020. *Recreating a Shiny App with Flask.* [Online]
Available at: https://www.jumpingrivers.com/blog/r-shiny-python-flask/
[Accessed 31 October 2024].

Kleinbaum, D. G. & Klein, M., 1994. *Logistic Regression A Self-Learning Text.* 3rd ed. s.l.:SpringerLink.

Kleinbaum, D. G. & Klein, M., 2010. *Logistic Regression*. Third Edition ed. New York, NY: Springer New York.

Kotsiantis, S. B., 2011. Decision trees: a recent overview. In: s.l.:SpringerLink, pp. 261-283.

Kotsiantis, S. B., 2011. RETRACTED ARTICLE: Feature selection for machine learning classification problems: a recent overview. *Springer Science+Business Media,* 42(1), pp. 157-157.

Kuhn, M., 2019. *The caret Package*. [Online]
Available at: https://topepo.github.io/caret/
[Accessed 15 October 2024].

Lever, J., Krzywinski, M. & Altman, N., 2016. Points of Significance: Regularization. *Nature Methods,* 13(10), p. 803+.

Malsam, W., 2022. *Project Assumptions: A Quick Guide*. [Online]
Available at: https://www.projectmanager.com/blog/project-assumptions
[Accessed 1 October 2024].

monday.com, 2024. *Project Assumptions: What They Are and Why You Should Care*. [Online]
Available at: https://monday.com/blog/project-management/project-assumptions/
[Accessed 1 October 2024].

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

posit, 2024. *Get your Shiny apps online*. [Online]
Available at: https://posit.co/products/open-source/shiny-server/
[Accessed 31 October 2024].

Roshaan, E., 2024. *Top tips: Watch out for these 4 machine learning risks*. [Online]
Available at: https://blogs.manageengine.com/corporate/general/2024/04/04/top-tips-watch-out-for-these-4-machine-learning-risks.html
[Accessed 1 October 2024].

Shiny, 2024. *Welcome to Shiny*. [Online]
Available at: https://shiny.posit.co/r/getstarted/shiny-basics/lesson1/index.html
[Accessed 30 10 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].

Tavish, 2024. *12 Important Model Evaluation Metrics for Machine Learning Everyone Should Know (Updated 2024)*. [Online]
Available at: https://www.analyticsvidhya.com/blog/2019/08/11-important-model-evaluation-error-metrics/
[Accessed 1 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].