Question 1:

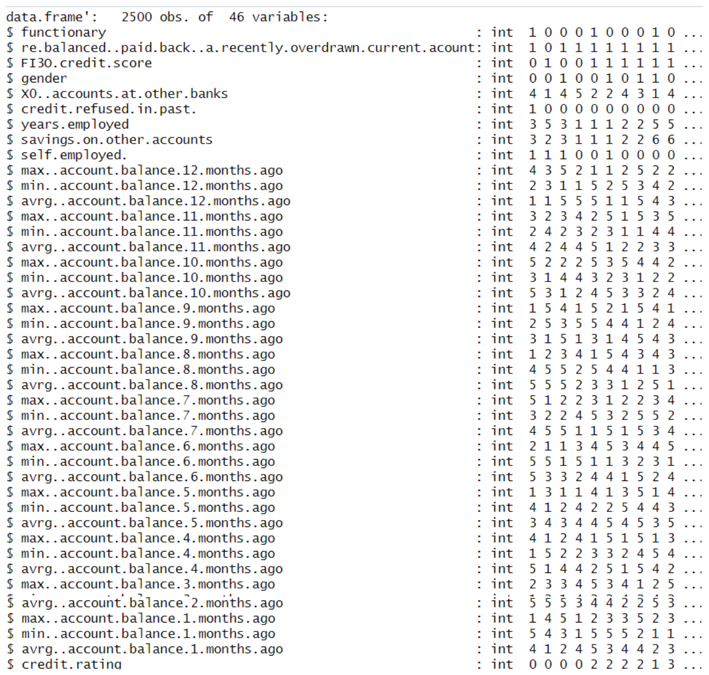
a) Analyse the general properties of the dataset:

I used the str() function to get the structure of the dataset and I found that the data is mainly made of the following attributes as shown in Figure 18:

- Functionary
- FI3O credit score
- Credit rating
- Gender
- X0 accounts at other banks
- Credit refused in past
- Years employed
- Savings on other accounts
- self employed
- Rebalanced (paid back) a recently overdrawn current account
- the max, minimum, and average account balance for every month of the last year

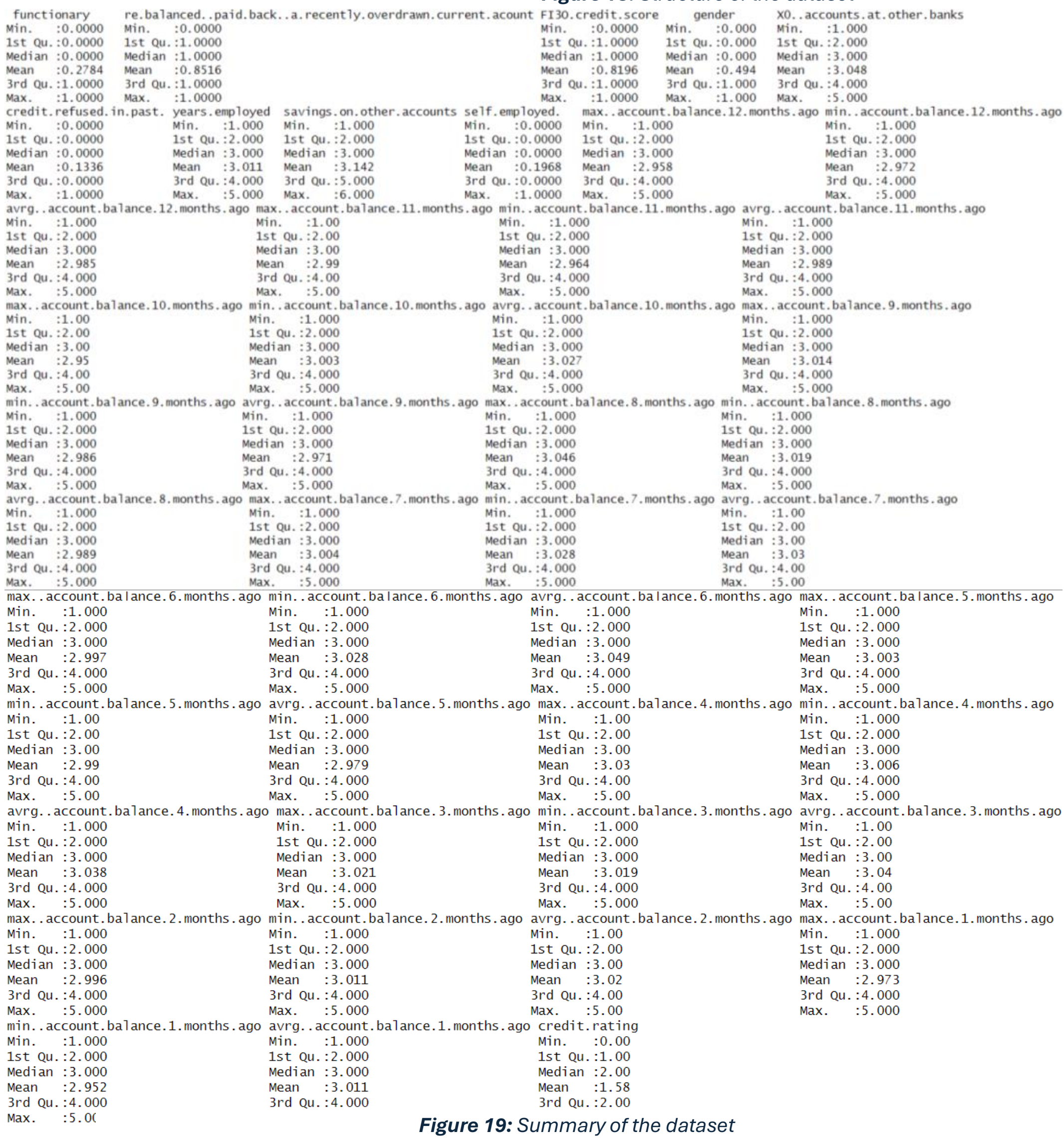
I also used the summary() function to print a summary of the dataset and found that for the attributes that have values other than 0 and 1, the mean was (refer to Figure 19):

- **years employed:** 3.011 indicating that on average, customers have been employed between 2 and 5 years.
- **savings on other accounts:** 3.142 indicating that on average, customers have 7.500 in their savings.
- **max, min, and average account balance:**



```
data.frame':   2500 obs. of  46 variables:
$ functionary                                           : int  1 0 0 0 1 0 0 0 1 0 ...
$ re.balanced..paid.back..a.recently.overdrawn.current.acount: int  1 0 1 1 1 1 1 1 1 1 ...
$ FI3O.credit.score                                     : int  0 1 0 0 1 1 1 1 1 1 ...
$ gender                                                : int  0 0 1 0 0 1 0 1 1 0 ...
$ X0..accounts.at.other.banks                           : int  4 1 4 5 2 2 4 3 1 4 ...
$ credit.refused.in.past.                               : int  1 0 0 0 0 0 0 0 0 0 ...
$ years.employed                                        : int  3 5 3 1 1 1 2 2 5 5 ...
$ savings.on.other.accounts                             : int  3 2 3 1 1 1 2 2 6 6 ...
$ self.employed.                                        : int  1 1 1 0 0 1 0 0 0 0 ...
$ max..account.balance.12.months.ago                    : int  4 3 5 2 1 1 2 5 2 2 ...
$ min..account.balance.12.months.ago                    : int  2 3 1 1 5 2 5 3 4 2 ...
$ avrg..account.balance.12.months.ago                   : int  1 1 5 5 5 1 1 5 4 3 ...
$ max..account.balance.11.months.ago                    : int  3 2 3 4 2 5 1 5 3 5 ...
$ min..account.balance.11.months.ago                    : int  2 4 2 3 2 3 1 1 4 4 ...
$ avrg..account.balance.11.months.ago                   : int  4 2 4 4 5 1 2 2 3 3 ...
$ max..account.balance.10.months.ago                    : int  5 2 2 2 5 3 5 4 4 2 ...
$ min..account.balance.10.months.ago                    : int  3 1 4 4 3 2 3 1 2 2 ...
$ avrg..account.balance.10.months.ago                   : int  5 3 1 2 4 5 3 3 2 4 ...
$ max..account.balance.9.months.ago                     : int  1 5 4 1 5 2 1 5 4 1 ...
$ min..account.balance.9.months.ago                     : int  2 5 3 5 5 4 4 1 2 4 ...
$ avrg..account.balance.9.months.ago                    : int  3 1 5 1 3 1 4 5 4 3 ...
$ max..account.balance.8.months.ago                     : int  1 2 3 4 1 5 4 3 4 3 ...
$ min..account.balance.8.months.ago                     : int  4 5 5 2 5 4 4 1 1 3 ...
$ avrg..account.balance.8.months.ago                    : int  5 5 5 2 3 3 1 2 5 1 ...
$ max..account.balance.7.months.ago                     : int  5 1 2 2 3 1 2 2 3 4 ...
$ min..account.balance.7.months.ago                     : int  3 2 2 4 5 3 2 5 5 2 ...
$ avrg..account.balance.7.months.ago                    : int  4 5 5 1 1 5 1 5 3 4 ...
$ max..account.balance.6.months.ago                     : int  2 1 1 3 4 5 3 4 4 5 ...
$ min..account.balance.6.months.ago                     : int  5 5 1 5 1 1 3 2 3 1 ...
$ avrg..account.balance.6.months.ago                    : int  5 3 3 2 4 4 1 5 2 4 ...
$ max..account.balance.5.months.ago                     : int  1 3 1 1 4 1 3 5 1 4 ...
$ min..account.balance.5.months.ago                     : int  4 1 2 4 2 2 5 4 4 3 ...
$ avrg..account.balance.5.months.ago                    : int  3 4 3 4 4 5 4 5 3 5 ...
$ max..account.balance.4.months.ago                     : int  4 1 2 4 1 5 1 5 1 3 ...
$ min..account.balance.4.months.ago                     : int  1 5 2 2 3 3 2 4 5 4 ...
$ avrg..account.balance.4.months.ago                    : int  5 1 4 4 2 5 1 5 4 2 ...
$ max..account.balance.3.months.ago                     : int  2 3 3 4 5 3 4 1 2 5 ...
$ avrg..account.balance.2.months.ago                    : int  5 5 5 3 4 4 2 2 5 3 ...
$ max..account.balance.1.months.ago                     : int  1 4 5 1 2 3 3 5 2 3 ...
$ min..account.balance.1.months.ago                     : int  5 4 3 1 5 5 5 2 1 1 ...
$ avrg..account.balance.1.months.ago                    : int  4 1 2 4 5 3 4 4 2 3 ...
$ credit.rating                                         : int  0 0 0 0 2 2 2 2 1 3 ...
```

Figure 18: *Structure of the dataset*

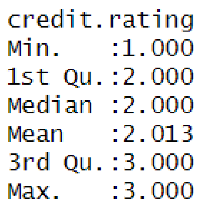


```
  functionary     re.balanced..paid.back..a.recently.overdrawn.current.acount FI3O.credit.score     gender       X0..accounts.at.other.banks
 Min.   :0.0000   Min.   :0.0000                                              Min.   :0.0000   Min.   :0.000   Min.   :1.000
 1st Qu.:0.0000   1st Qu.:1.0000                                              1st Qu.:1.0000   1st Qu.:0.000   1st Qu.:2.000
 Median :0.0000   Median :1.0000                                              Median :1.0000   Median :0.000   Median :3.000
 Mean   :0.2784   Mean   :0.8516                                              Mean   :0.8196   Mean   :0.494   Mean   :3.048
 3rd Qu.:1.0000   3rd Qu.:1.0000                                              3rd Qu.:1.0000   3rd Qu.:1.000   3rd Qu.:4.000
 Max.   :1.0000   Max.   :1.0000                                              Max.   :1.0000   Max.   :1.000   Max.   :5.000
 credit.refused.in.past. years.employed  savings.on.other.accounts self.employed.   max..account.balance.12.months.ago min..account.balance.12.months.ago
 Min.   :0.0000          Min.   :1.000   Min.   :1.000             Min.   :0.0000   Min.   :1.000                      Min.   :1.000
 1st Qu.:0.0000          1st Qu.:2.000   1st Qu.:2.000             1st Qu.:0.0000   1st Qu.:2.000                      1st Qu.:2.000
 Median :0.0000          Median :3.000   Median :3.000             Median :0.0000   Median :3.000                      Median :3.000
 Mean   :0.1336          Mean   :3.011   Mean   :3.142             Mean   :0.1968   Mean   :2.958                      Mean   :2.972
 3rd Qu.:0.0000          3rd Qu.:4.000   3rd Qu.:5.000             3rd Qu.:0.0000   3rd Qu.:4.000                      3rd Qu.:4.000
 Max.   :1.0000          Max.   :5.000   Max.   :6.000             Max.   :1.0000   Max.   :5.000                      Max.   :5.000
 avrg..account.balance.12.months.ago max..account.balance.11.months.ago min..account.balance.11.months.ago avrg..account.balance.11.months.ago
 Min.   :1.000                       Min.   :1.00                       Min.   :1.000                      Min.   :1.000
 1st Qu.:2.000                       1st Qu.:2.00                       1st Qu.:2.000                      1st Qu.:2.000
 Median :3.000                       Median :3.00                       Median :3.000                      Median :3.000
 Mean   :2.985                       Mean   :2.99                       Mean   :2.964                      Mean   :2.989
 3rd Qu.:4.000                       3rd Qu.:4.00                       3rd Qu.:4.000                      3rd Qu.:4.000
 Max.   :5.000                       Max.   :5.00                       Max.   :5.000                      Max.   :5.000
 max..account.balance.10.months.ago min..account.balance.10.months.ago avrg..account.balance.10.months.ago max..account.balance.9.months.ago
 Min.   :1.00                       Min.   :1.000                      Min.   :1.000                       Min.   :1.000
 1st Qu.:2.00                       1st Qu.:2.000                      1st Qu.:2.000                       1st Qu.:2.000
 Median :3.00                       Median :3.000                      Median :3.000                       Median :3.000
 Mean   :2.95                       Mean   :3.003                      Mean   :3.027                       Mean   :3.014
 3rd Qu.:4.00                       3rd Qu.:4.000                      3rd Qu.:4.000                       3rd Qu.:4.000
 Max.   :5.00                       Max.   :5.000                      Max.   :5.000                       Max.   :5.000
 min..account.balance.9.months.ago avrg..account.balance.9.months.ago max..account.balance.8.months.ago min..account.balance.8.months.ago
 Min.   :1.000                     Min.   :1.000                      Min.   :1.000                     Min.   :1.000
 1st Qu.:2.000                     1st Qu.:2.000                      1st Qu.:2.000                     1st Qu.:2.000
 Median :3.000                     Median :3.000                      Median :3.000                     Median :3.000
 Mean   :2.986                     Mean   :2.971                      Mean   :3.046                     Mean   :3.019
 3rd Qu.:4.000                     3rd Qu.:4.000                      3rd Qu.:4.000                     3rd Qu.:4.000
 Max.   :5.000                     Max.   :5.000                      Max.   :5.000                     Max.   :5.000
 avrg..account.balance.8.months.ago max..account.balance.7.months.ago min..account.balance.7.months.ago avrg..account.balance.7.months.ago
 Min.   :1.000                      Min.   :1.000                     Min.   :1.000                     Min.   :1.00
 1st Qu.:2.000                      1st Qu.:2.000                     1st Qu.:2.000                     1st Qu.:2.00
 Median :3.000                      Median :3.000                     Median :3.000                     Median :3.00
 Mean   :2.989                      Mean   :3.004                     Mean   :3.028                     Mean   :3.03
 3rd Qu.:4.000                      3rd Qu.:4.000                     3rd Qu.:4.000                     3rd Qu.:4.00
 Max.   :5.000                      Max.   :5.000                     Max.   :5.000                     Max.   :5.00
 max..account.balance.6.months.ago min..account.balance.6.months.ago avrg..account.balance.6.months.ago max..account.balance.5.months.ago
 Min.   :1.000                     Min.   :1.000                     Min.   :1.000                      Min.   :1.000
 1st Qu.:2.000                     1st Qu.:2.000                     1st Qu.:2.000                      1st Qu.:2.000
 Median :3.000                     Median :3.000                     Median :3.000                      Median :3.000
 Mean   :2.997                     Mean   :3.028                     Mean   :3.049                      Mean   :3.003
 3rd Qu.:4.000                     3rd Qu.:4.000                     3rd Qu.:4.000                      3rd Qu.:4.000
 Max.   :5.000                     Max.   :5.000                     Max.   :5.000                      Max.   :5.000
 min..account.balance.5.months.ago avrg..account.balance.5.months.ago max..account.balance.4.months.ago min..account.balance.4.months.ago
 Min.   :1.00                      Min.   :1.000                      Min.   :1.00                      Min.   :1.000
 1st Qu.:2.00                      1st Qu.:2.000                      1st Qu.:2.00                      1st Qu.:2.000
 Median :3.00                      Median :3.000                      Median :3.00                      Median :3.000
 Mean   :2.99                      Mean   :2.979                      Mean   :3.03                      Mean   :3.006
 3rd Qu.:4.00                      3rd Qu.:4.000                      3rd Qu.:4.00                      3rd Qu.:4.000
 Max.   :5.00                      Max.   :5.000                      Max.   :5.00                      Max.   :5.000
 avrg..account.balance.4.months.ago max..account.balance.3.months.ago min..account.balance.3.months.ago avrg..account.balance.3.months.ago
 Min.   :1.000                      Min.   :1.000                     Min.   :1.000                     Min.   :1.00
 1st Qu.:2.000                      1st Qu.:2.000                     1st Qu.:2.000                     1st Qu.:2.00
 Median :3.000                      Median :3.000                     Median :3.000                     Median :3.00
 Mean   :3.038                      Mean   :3.021                     Mean   :3.019                     Mean   :3.04
 3rd Qu.:4.000                      3rd Qu.:4.000                     3rd Qu.:4.000                     3rd Qu.:4.00
 Max.   :5.000                      Max.   :5.000                     Max.   :5.000                     Max.   :5.00
 max..account.balance.2.months.ago min..account.balance.2.months.ago avrg..account.balance.2.months.ago max..account.balance.1.months.ago
 Min.   :1.000                     Min.   :1.000                     Min.   :1.00                       Min.   :1.000
 1st Qu.:2.000                     1st Qu.:2.000                     1st Qu.:2.00                       1st Qu.:2.000
 Median :3.000                     Median :3.000                     Median :3.00                       Median :3.000
 Mean   :2.996                     Mean   :3.011                     Mean   :3.02                       Mean   :2.973
 3rd Qu.:4.000                     3rd Qu.:4.000                     3rd Qu.:4.00                       3rd Qu.:4.000
 Max.   :5.000                     Max.   :5.000                     Max.   :5.00                       Max.   :5.000
 min..account.balance.1.months.ago avrg..account.balance.1.months.ago credit.rating
 Min.   :1.000                     Min.   :1.000                      Min.   :0.00
 1st Qu.:2.000                     1st Qu.:2.000                      1st Qu.:1.00
 Median :3.000                     Median :3.000                      Median :2.00
 Mean   :2.952                     Mean   :3.011                      Mean   :1.58
 3rd Qu.:4.000                     3rd Qu.:4.000                      3rd Qu.:2.00
 Max.   :5.0(
```

Figure 19: *Summary of the dataset*

approximately 3 over the 12 months period indicating that customers usually have 7.500 as their max/min/average account balances which is the same as the savings on other accounts attribute.

- **credit rating:** 1.58 with the unknown data indicating that most customers have an A rating. However, if we remove the unknown data, then the credit rating becomes 2.013 (refer to Figure 20) indicating that the customers are usually of a B class.

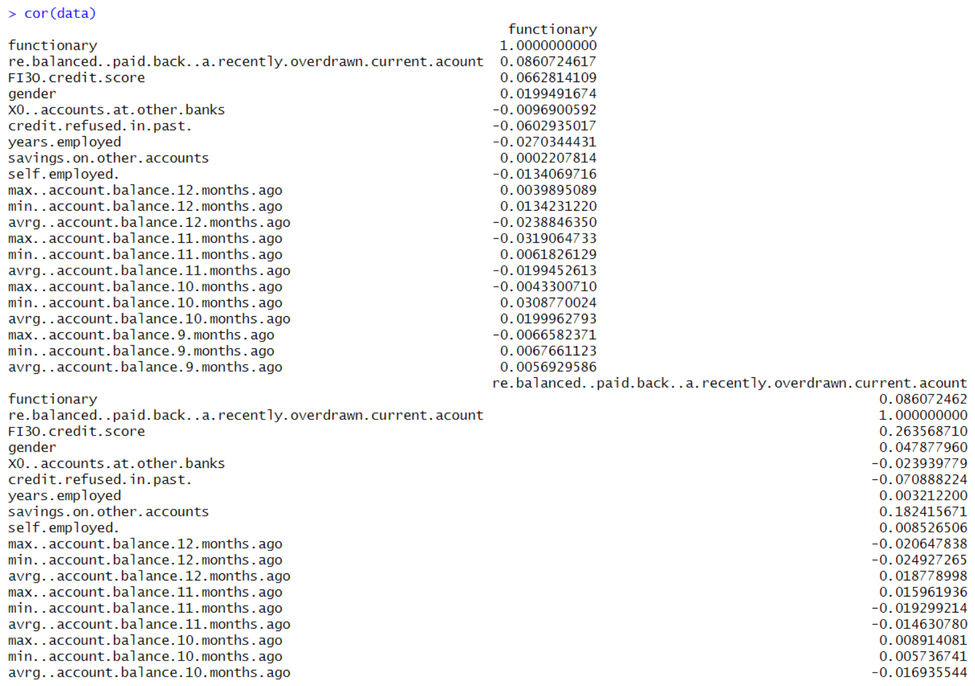


```
credit.rating
Min.   :1.000
1st Qu.:2.000
Median :2.000
Mean   :2.013
3rd Qu.:3.000
Max.   :3.000
```

Figure 20: *Credit summary without unknown data*

b) Statistical analysis creation and choosing the 5 most interesting attributes:

- To do statistical analysis I used the cor() function on the data set and got the output in Figure 19 (it shows some of the output only).
- The five most valuable attributes are:
    1) **FI3O credit score:** this is a very popular credit score method that helps lenders decide whether to lend or not (Hayes, Brock 2023). The higher the score, the lower the credit risk and vice versa (Hayes, Brock 2023).
    2) **credit refused in past?:** helps in indicating the customer's history of lending applications which indicates their credit worthiness.
    3) **years employed:** helps in indicating the financial stability of the customer and their ability to pay back the borrowed money.
    4) **savings on other accounts:** helps in indicating the customer's ability to manage their funds which will also indicate their ability to pay back the money borrowed.
    5) **self employed?:** depending on their business's success, a self-employed customer can be of a higher risk compared to a traditionally employed customer as the fluctuations in income and the business risk may put them at a disadvantage.



```
> cor(data)
                                                              functionary
functionary                                                  1.0000000000
re.balanced..paid.back..a.recently.overdrawn.current.acount  0.0860724617
FI30.credit.score                                            0.0662814109
gender                                                       0.0199491674
X0..accounts.at.other.banks                                 -0.0096900592
credit.refused.in.past.                                     -0.0602935017
years.employed                                              -0.0270344431
savings.on.other.accounts                                    0.0002207814
self.employed.                                              -0.0134069716
max..account.balance.12.months.ago                           0.0039895089
min..account.balance.12.months.ago                           0.0134231220
avrg..account.balance.12.months.ago                         -0.0238846350
max..account.balance.11.months.ago                          -0.0319064733
min..account.balance.11.months.ago                           0.0061826129
avrg..account.balance.11.months.ago                         -0.0199452613
max..account.balance.10.months.ago                          -0.0043300710
min..account.balance.10.months.ago                           0.0308770024
avrg..account.balance.10.months.ago                          0.0199962793
max..account.balance.9.months.ago                           -0.0066582371
min..account.balance.9.months.ago                            0.0067661123
avrg..account.balance.9.months.ago                           0.0056929586
                                                            re.balanced..paid.back..a.recently.overdrawn.current.acount
functionary                                                                                                 0.086072462
re.balanced..paid.back..a.recently.overdrawn.current.acount                                                 1.000000000
FI30.credit.score                                                                                           0.263568710
gender                                                                                                      0.047877960
X0..accounts.at.other.banks                                                                                -0.023939779
credit.refused.in.past.                                                                                    -0.070888224
years.employed                                                                                              0.003212200
savings.on.other.accounts                                                                                   0.182415671
self.employed.                                                                                              0.008526506
max..account.balance.12.months.ago                                                                         -0.020647838
min..account.balance.12.months.ago                                                                         -0.024927265
avrg..account.balance.12.months.ago                                                                         0.018778998
max..account.balance.11.months.ago                                                                          0.015961936
min..account.balance.11.months.ago                                                                         -0.019299214
avrg..account.balance.11.months.ago                                                                        -0.014630780
max..account.balance.10.months.ago                                                                          0.008914081
min..account.balance.10.months.ago                                                                          0.005736741
avrg..account.balance.10.months.ago                                                                        -0.016935544
```

Figure 21: *Statistical Analysis Output*

c) Figures 22, 25, and 28 show the node counts graphs for 3 SOM models: one with the whole dataset, one without the data with the unassessed credit rating, and one with only the 5 attributes I chose in part (b) respectively. The lighter the nodes' colours, the more data is allocated to them. Grey nodes indicate nodes with no data allocated to them. All 3 models run on the same conditions for 1000 iterations and a learning rate that starts at 0.9 then decreases till 0.01. I first implemented the SOM model that contains the whole dataset (refer to Figure 22) but when I ran the summary() function on it, I noticed that the mean distance to the closest node is quite high: 43.753 (refer to Figure 24). I ran the changes in the training progress graph and noticed that there was a linear decrease in the mean distance to the closest node before a sharp drop happened in it towards the end of the iteration cycle (refer to Figure 23). I then implemented the second SOM model that contains every attribute but the ones that contain the unassessed credit rating (refer to Figure 25). The mean distance between the nodes between the nodes decreased but was still high at 40.113 (refer to Figure 27). The changes in the training progress graph had the same result as that of the first model (refer to Figure 26). Hence, I decided to implement the third model that used only the 5 attributes I chose in part (b) (refer to Figure 28). Interestingly, the mean distance to the closest node became zero (refer to Figure 30) and the changes in the training progress graph illustrated a directly proportional relationship between the mean distance to the closest node and the iterations (refer to Figure 29). The node counts graph (refer to Figure 28), though, had many empty nodes which made me believe that if the right strategy is used to determine the correct attributes that affect the data (feature selection), then the mean distance to the closest node can decrease.

With such a small dataset, achieving 100% prediction accuracy is impossible as there is not enough data for the model to learn from. This is because there are not enough data to represent the relationship between the attributes. The fact that the distance between the nodes is high also indicates that there might be irrelevant attributes in the dataset which act as noise and hinders the accuracy of the prediction model.

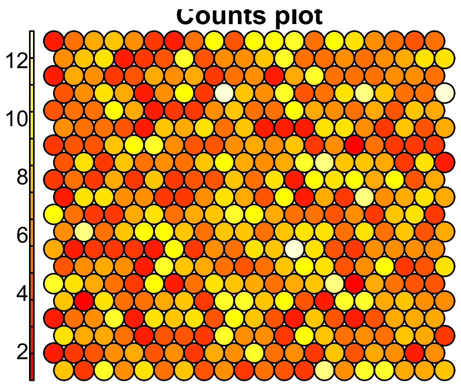


Figure 22: Whole Dataset's Model's Counts Plot

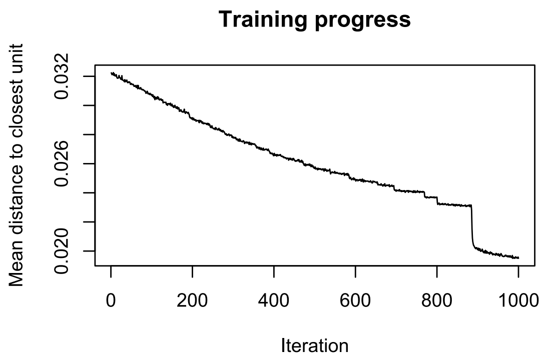
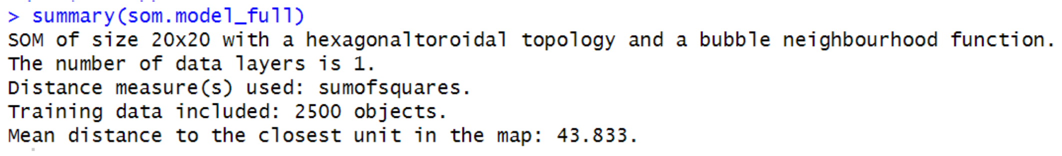


Figure 23: Whole Dataset's Model's Changes in the Training Progress Graph



```
> summary(som.model_full)
SOM of size 20x20 with a hexagonaltoroidal topology and a bubble neighbourhood function.
The number of data layers is 1.
Distance measure(s) used: sumofsquares.
Training data included: 2500 objects.
Mean distance to the closest unit in the map: 43.833.
```

Figure 24: Whole Dataset's Model's Summary

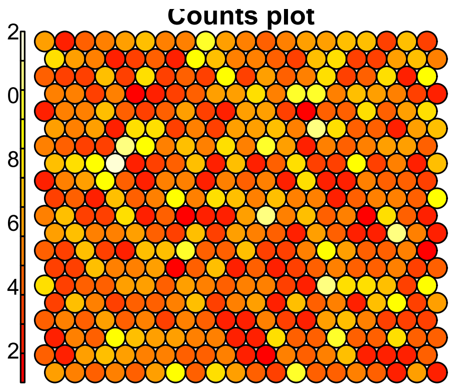


Figure 25: Counts Plot for Model with all Attributes except Credit Rating

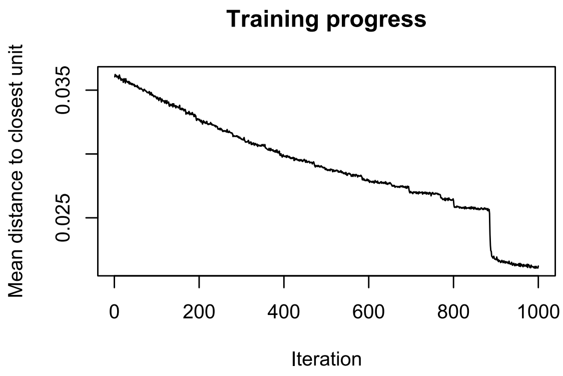
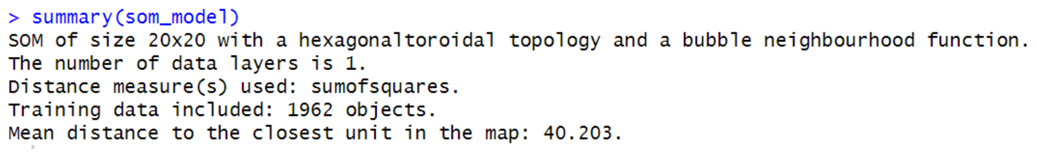


Figure 26: Changes in the Training Progress Graph for Model with all Attributes except Credit Rating



```
> summary(som_model)
SOM of size 20x20 with a hexagonaltoroidal topology and a bubble neighbourhood function.
The number of data layers is 1.
Distance measure(s) used: sumofsquares.
Training data included: 1962 objects.
Mean distance to the closest unit in the map: 40.203.
```

Figure 27: *Summary for Model with all Attributes except Credit Rating*

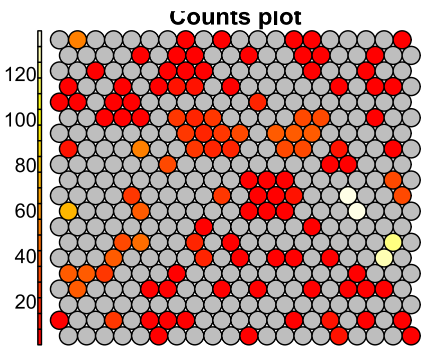


Figure 28: *Counts Plot for Model with Chosen Attributes*

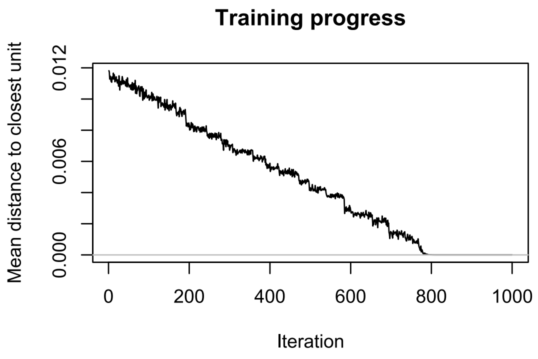
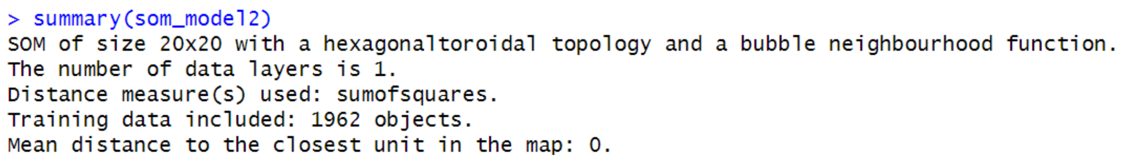


Figure 29: *Changes in the Training Progress Graph for Model with Chosen Attributes*



```
> summary(som_model2)
SOM of size 20x20 with a hexagonaltoroidal topology and a bubble neighbourhood function.
The number of data layers is 1.
Distance measure(s) used: sumofsquares.
Training data included: 1962 objects.
Mean distance to the closest unit in the map: 0.
```

Figure 30: *Summary for Model with all Chosen Attributes*

<u>Question 2:</u>

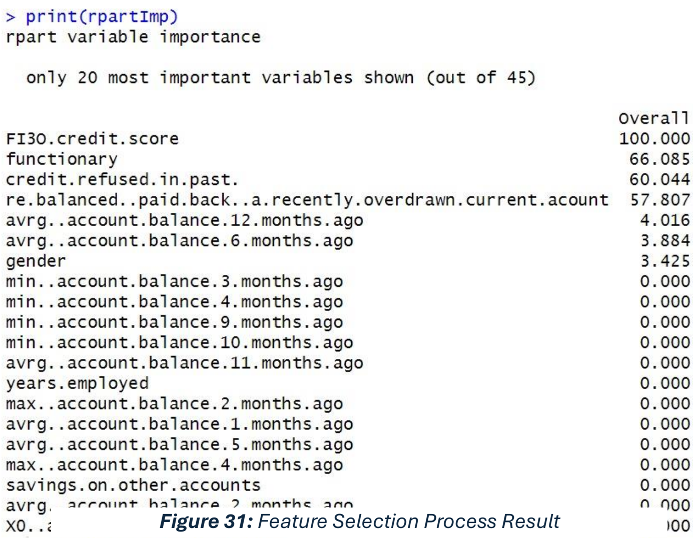
a) Strategy:

- Use feature selection in order to choose the most important attributes in the dataset (Prabhakaran n.d.). This can be done using the caret library where a seed is set for the randomness of the data splitting (Prabhakaran n.d.). Then, the model is trained using the train() function with credit rating as the target variable and the decision tree algorithm, rpart, is used as the method of training (Prabhakaran n.d.). Lastly, the varImp() function is used to determine the importance of the variables to the accuracy of the prediction capabilities of the model (Prabhakaran n.d.). Those variables are then printed out and the ones with an overall rate higher than zero are used in the next step (Prabhakaran n.d.).
- Use backpropagation to train the MLP and track the plot of the Weighted SSE while editing the code for the model and training set to change the number of neurones in the hidden data, the learning rate of the model, and the number of training iterations until the errors decrease in the Weighted SSE. Then, stop making changes and use those numbers to continue the task.

This strategy would ensure that only the most valuable attributes are used in the prediction process without the non-relevant ones which reduces redundant data caused by noise and reduces the rate of the deceptive data (H2O.ai n.d.). Hence, it reduces overfitting, training time, and improves the accuracy of the prediction model (H2O.ai n.d.).

b) The result of the feature selection process can be seen in Figure 31 where the first 7 variables where the ones who were determined to be the most important and therefore used in the backpropagation training (refer to Figure 32 for the model illustration).

The confusion matrix for the training and testing targets are shown in Figures 33 and 34 respectively:



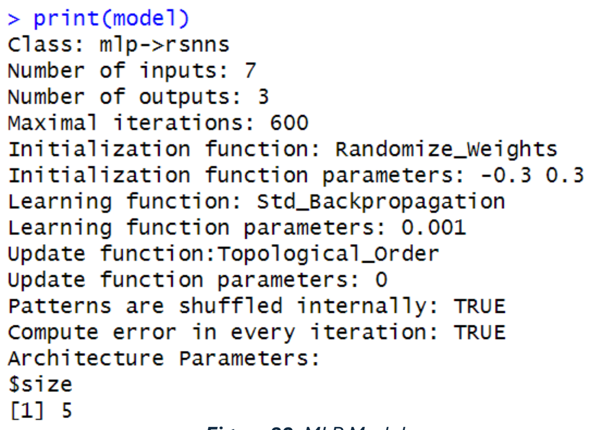
```
> print(rpartImp)
rpart variable importance

  only 20 most important variables shown (out of 45)

                                                         Overall
FI3O.credit.score                                        100.000
functionary                                               66.085
credit.refused.in.past.                                   60.044
re.balanced..paid.back..a.recently.overdrawn.current.acount  57.807
avrg..account.balance.12.months.ago                        4.016
avrg..account.balance.6.months.ago                         3.884
gender                                                     3.425
min..account.balance.3.months.ago                          0.000
min..account.balance.4.months.ago                          0.000
min..account.balance.9.months.ago                          0.000
min..account.balance.10.months.ago                         0.000
avrg..account.balance.11.months.ago                        0.000
years.employed                                             0.000
max..account.balance.2.months.ago                          0.000
avrg..account.balance.1.months.ago                         0.000
avrg..account.balance.5.months.ago                         0.000
max..account.balance.4.months.ago                          0.000
savings.on.other.accounts                                  0.000
avrg..account.balance.2.months.ago                         0.000
X0..                                                        00
```

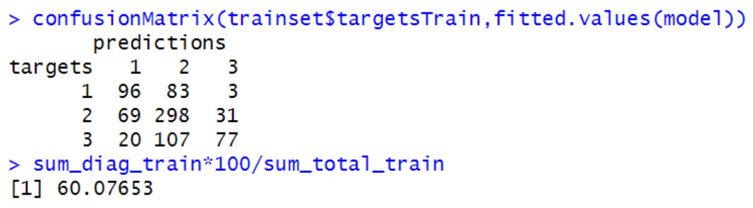
Figure 31: *Feature Selection Process Result*

- **Training set:**
    - **Accuracy percentage:** 60.07653%
    - **Total predictions – class 1:** 185 (96 + 69 + 20)
    - **Correct predictions – class 1:** 96
    - **Wrong predictions – class 1:** 89 (69 + 20)
    - **Total predictions – class 2:** 488 (83 + 298 + 107)
    - **Correct predictions – class 2:** 298
    - **Wrong predictions – class 2:** 190 (83 + 107)
    - **Total predictions – class 3:** 111 (3 + 31 + 77)
    - **Correct predictions – class 3:** 77
    - **Wrong predictions – class 3:** 34 (3 + 31)
    - **Notes:** because of the higher allocations of samples in the wrong classes, especially for in class 2, the model proves to still be in need of improvement



```
> print(model)
Class: mlp->rsnns
Number of inputs: 7
Number of outputs: 3
Maximal iterations: 600
Initialization function: Randomize_Weights
Initialization function parameters: -0.3 0.3
Learning function: Std_Backpropagation
Learning function parameters: 0.001
Update function:Topological_Order
Update function parameters: 0
Patterns are shuffled internally: TRUE
Compute error in every iteration: TRUE
Architecture Parameters:
$size
[1] 5
```

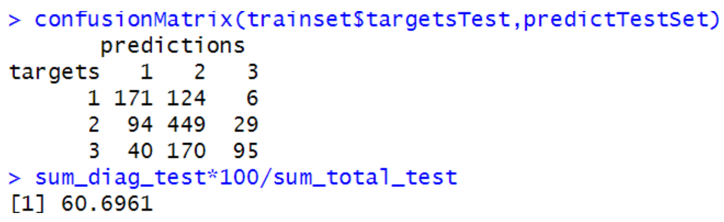
Figure 32: *MLP Model*



```
> confusionMatrix(trainset$targetsTrain,fitted.values(model))
       predictions
targets   1   2   3
      1  96  83   3
      2  69 298  31
      3  20 107  77
> sum_diag_train*100/sum_total_train
[1] 60.07653
```

Figure 33: *Confusion Matrix for the Training Targets*

- **Testing set:**
    - **Accuracy percentage:** 60.6961%
    - **Total predictions – class 1:** 305 (171 + 94 + 40)
    - **Correct predictions – class 1:** 171
    - **Wrong predictions – class 1:** 134 (94 + 40)
    - **Total predictions – class 2:** 743 (124 + 449 + 170)
    - **Correct predictions – class 2:** 449
    - **Wrong predictions – class 2:** 294 (124 + 170)
    - **Total predictions – class 3:** 130 (6 + 29 + 95)
    - **Correct predictions – class 3:** 95
    - **Wrong predictions – class 3:** 35 (6 + 29)
    - **Notes:** because of the higher allocations of samples in the wrong classes, especially for in class 2, the model proves to still be in need of improvement



```
> confusionMatrix(trainset$targetsTest,predictTestSet)
       predictions
targets   1   2   3
      1 171 124   6
      2  94 449  29
      3  40 170  95
> sum_diag_test*100/sum_total_test
[1] 60.6961
```

Figure 34: *Confusion Matrix for the Testing Targets*

Both confusion matrices show higher errors when classifying class 2, followed by class 1, then class 3 which indicates that maybe there was not enough data for class 2 for it to use and learn from.

Figure 35 demonstrates a Weighted SSE graph to illustrate the Weighted SSE (y-axis) of the model against the 1000 iterations (x-axis) the model ran for. The black line represents the IterativeFitError while the red one represents the IterativeTestError (RDocumentation n.d.). As indicated in the graph, at the beginning of the iterations the Weighted SSE was very high for both lines. However, it decreased as the iterations went on indicating the good choice of the learning rate value. The

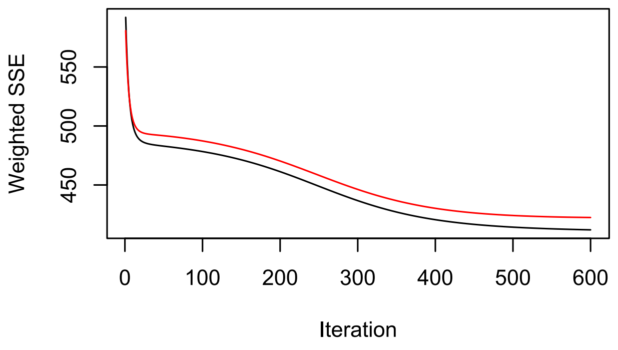


Figure 35: *Weighted SSE Graph*

black line has a lower trend than the red line and both lines which shows that the tests have more errors than the fit errors. This again indicates the need for improvement of the model.

c) There are only 2500 data is given in this dataset and with the fact that the model was predicting most of the data to be a class 2 indicates that there are not enough data for it to learn from. To improve accuracy of the dataset, we can try adding more hidden layers to increase the complexity of the model in order to help in capturing more complex relationships between the data which will help in the learning process and improves the MLP's performance. Use a cross validation technique, such as k-fold cross validation, to validate the results of the model against it in order to ensure that the performance of the model is improving because it is learning the relationships between the attributes and not because it is being overfit (Ray 2024). Increasing the amount of data will also undoubtedly help in training the model more effectively as there will be more situations for it to learn from.

References

Bonola, m, Loreti, P, Bianchi, G, Amici, R, & Rabuffi, A 2014, *CRAWDAD ROMA/TAXI*, IEEEDataPort, viewed 30 March 2024, <https://ieee-dataport.org/open-access/crawdad-romataxi>.

Google Maps n.d., Google Maps, viewed 30 March 2024, <https://www.google.com/maps>.

H2O.ai n.d., *Feature Selection*, H2O.ai, viewed 7 April 2024, <https://h2o.ai/wiki/feature-selection/#:~:text=In%20the%20machine%20learning%20process,why%20feature%20selection%20is%20important>.

Hayes, A, Brock, T 2023, *FICO Score*, Investopedia, viewed 7 April 2024, <https://www.investopedia.com/terms/f/ficoscore.asp#:~:text=FICO%20credit%20scores%20are%20a,be%20%E2%80%9Cgood%E2%80%9D%20credit%20scores>.

Prabhakaran, S n.d., *Feature Selection – Ten Effective Techniques with Examples*, machinelearningplus, viewed 7 April 2024, <https://www.machinelearningplus.com/machine-learning/feature-selection/#2variableimportancefrommachinelearningalgorithms>.

Ray, S 2024, ‘Enhance Model Performance through Cross Validation: A Guide in Python and R’, Analytics Vidhya, weblog post, viewed 7 April 2024, <https://www.analyticsvidhya.com/blog/2018/05/improve-model-performance-cross-validation-in-python-r/>.

RDocumentation n.d., *plotIterativeError: Plot iterative errors of an rsnns object*, RDocumentation, viewed 7 April 2024, <https://www.rdocumentation.org/packages/RSNNS/versions/0.4-17/topics/plotIterativeError>.