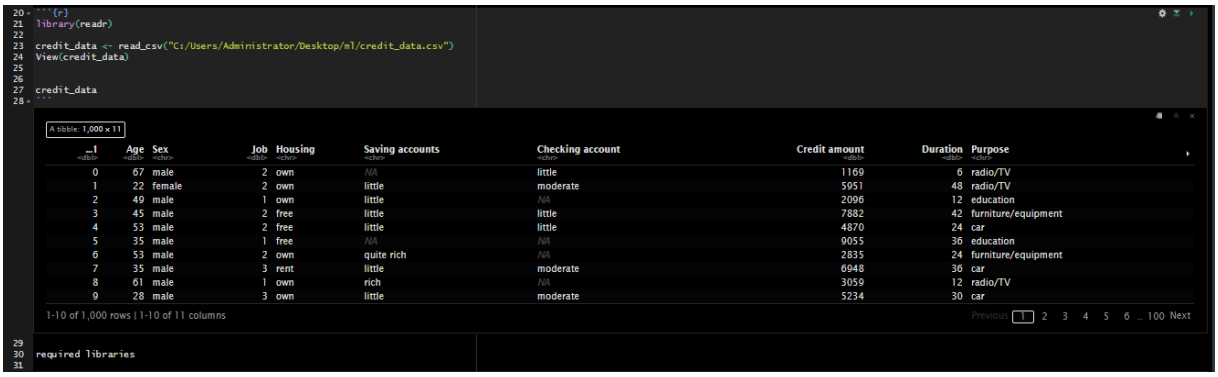
TITLE; Use logistical Regression to predict credit risk using the dataset "credit risk"

Data source; the data in the link below shows how individuals are categorized in terms of good or bad risk in relation to enquiries such as age, sex, housing, job, saving accounts, checking accounts, credit amount, duration and purpose https://drive.google.com/file/d/185S99Q-cGG2BQIXx1Qr4a9JbMQ57kdYe/view?usp=sharing

Overview of the dataset

Shows how the data in the above link is tabled with its relevant values.



```r
library(readr)

credit_data <- read_csv("C:/Users/Administrator/Desktop/ml/credit_data.csv")
View(credit_data)


credit_data
```

A tibble: 1,000 × 11

...1	Age	Sex	Job	Housing	Saving accounts	Checking account	Credit amount	Duration	Purpose
0	67	male	2	own	NA	little	1169	6	radio/TV
1	22	female	2	own	little	moderate	5951	48	radio/TV
2	49	male	1	own	little	NA	2096	12	education
3	45	male	2	free	little	little	7882	42	furniture/equipment
4	53	male	2	free	little	little	4870	24	car
5	35	male	1	free	NA	NA	9055	36	education
6	53	male	2	own	quite rich	NA	2835	24	furniture/equipment
7	35	male	3	rent	little	moderate	6948	36	car
8	61	male	1	own	rich	NA	3059	12	radio/TV
9	28	male	3	own	little	moderate	5234	30	car

1-10 of 1,000 rows | 1-10 of 11 columns

Previous 1 2 3 4 5 6 ... 100 Next

```
required libraries
```

Procedure

- Clean the dataset check if there are missing values (NA)
- If their exists NA values in our dataset; alternative one we delete the rows containing the NA values and alternative two we find the uniquely repeated variables in the columns with NA values; we then table it to calculate total thus replace the NA values with the mode variable formation of a cal_mode()

"The total sum of NA is 577 which are more than half of the total size of the dataset thus using alternative one makes analysis biased since more than half of the dataset would not be used to make analysis, hence alternative two is more appropriate; variables in the column

containing NA values are tabled to know the modal variable which is thus used to replace the NA variables.

Code

Sum (is.na (credit_data)) #this function finds the total number of NA values in a provided dataset

Table (credit_data$`Saving accounts`) #this function finds the total number of each variable that are in the columns with NA values.

mode1 <- names (sort (table (credit_data$`Saving accounts`), decreasing = TRUE)) [1] #this function identifies each variable in the columns having NA values by name, table all variables with their corresponding total number and hence sort the table in ascending order

mode1

credit_data$`Saving accounts`[is.na(credit_data$`Saving accounts`)] <- mode1 #this function replaces the NA values with the mode variable in model1.

mode2 <- names (sort (table (credit_data$`Checking account`), decreasing = TRUE))[1]

mode2

credit_data$`Checking account`[is.na(credit_data$`Checking account`)] <- mode2

credit_data

- Identify the dependent variable; since we are using the logistics regression model to perform analysis the dependent variable must be a binary character. In this case dependent variable is not binary hence we make it binary first.

Code

credit_data$Risk <- ifelse (credit_data$Risk == "good", 1, 0) #this functions makes two stage variables binary

credit_data

- split the dataset into Training and Testing sets we use train data to train our model, we use test data to predict our model

Code

Set. Seed (1023)

split1 <- sample(c (rep (0, 0.8* nrow (credit_data)), rep (1, 0.2 * nrow(credit_data))))

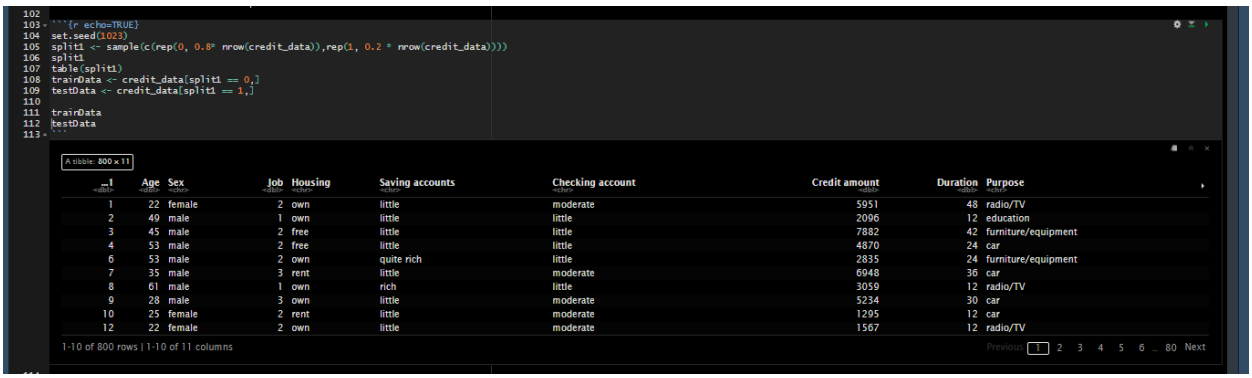
split1

table (split1)

trainData <- credit_data[split1 == 0,]

testData <- credit_data[split1 == 1,]

TrainData



```r
set.seed(1023)
split1 <- sample(c(rep(0, 0.8* nrow(credit_data)),rep(1, 0.2 * nrow(credit_data))))
split1
table(split1)
trainData <- credit_data[split1 == 0,]
testData <- credit_data[split1 == 1,]

trainData
testData
```

A tibble: 800 × 11

...1	Age	Sex	Job	Housing	Saving accounts	Checking account	Credit amount	Duration	Purpose
1	22	female	2	own	little	moderate	5951	48	radio/TV
2	49	male	1	own	little	little	2096	12	education
3	45	male	2	free	little	little	7882	42	furniture/equipment
4	53	male	2	free	little	little	4870	24	car
6	53	male	2	own	quite rich	little	2835	24	furniture/equipment
7	35	male	3	rent	little	moderate	6948	36	car
8	61	male	1	own	rich	little	3059	12	radio/TV
9	28	male	3	own	little	moderate	5234	30	car
10	25	female	2	rent	little	moderate	1295	12	car
12	22	female	2	own	little	moderate	1567	12	radio/TV

1-10 of 800 rows | 1-10 of 11 columns

Previous 1 2 3 4 5 6 ... 80 Next

TestData

```r
```{r echo=TRUE}
set.seed(1023)
split1 <- sample(c(rep(0, 0.8* nrow(credit_data)),rep(1, 0.2 * nrow(credit_data))))
split1
table(split1)
trainData <- credit_data[split1 == 0,]
testData <- credit_data[split1 == 1,]

trainData
testData
```
```

A tibble: 200 × 11

...1	Age	Sex	Job	Housing	Saving accounts	Checking account	Credit amount	Duration	Purpose
0	67	male	2	own	little	little	1169	6	radio/TV
5	35	male	1	free	little	little	9055	36	education
11	24	female	2	rent	little	little	4308	48	business
15	32	female	1	own	moderate	little	1282	24	radio/TV
16	53	male	2	own	little	little	2424	24	radio/TV
17	25	male	2	own	little	little	8072	30	business
18	44	female	3	free	little	moderate	12579	24	car
20	48	male	2	own	little	little	2134	9	car
27	42	female	2	rent	rich	rich	409	12	radio/TV
35	25	male	1	own	little	moderate	4746	45	radio/TV

1-10 of 200 rows | 1-10 of 11 columns

Previous 1 2 3 4 5 6 ... 20 Next

- create a logistics regression model using the trainData; this helps us to train our model Null Devience : shows how well the model predicts the response variable with only the intercept residual Devience : shows how well the model predicts the response variable with the inclusion of the independent/predictor variables

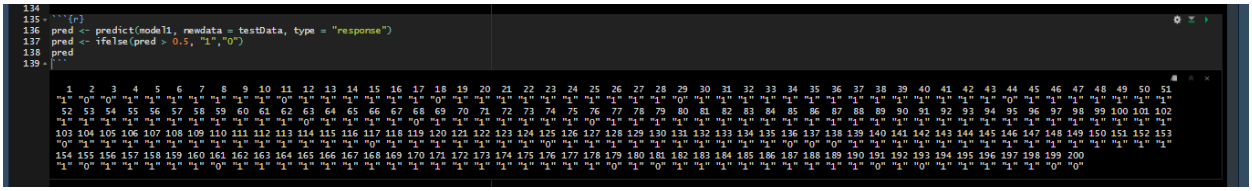
Code

```r
model1 <- glm(Risk~.,data = trainData,family = binomial)
model1

summary(model1)
```

```
Call:
glm(formula = Risk ~ ., family = binomial, data = trainData)

Coefficients:
                              Estimate Std. Error z value Pr(>|z|)
(Intercept)                  1.413e-01  6.673e-01   0.212 0.832345
...1                        -9.226e-05  2.891e-04  -0.319 0.749655
Age                          2.062e-02  8.350e-03   2.469 0.013555 *
Sexmale                      3.312e-01  1.823e-01   1.816 0.069365 .
Job                          7.921e-02  1.345e-01   0.589 0.556001
Housingown                   4.532e-01  2.889e-01   1.569 0.116756
Housingrent                  9.252e-02  3.423e-01   0.270 0.786960
`Saving accounts`moderate    1.962e-01  2.765e-01   0.710 0.477934
`Saving accounts`quite rich  3.558e-01  3.657e-01   0.973 0.330518
`Saving accounts`rich        1.051e+00  4.975e-01   2.113 0.034611 *
`Checking account`moderate  -3.769e-01  1.868e-01  -2.018 0.043607 *
`Checking account`rich       2.748e-01  3.783e-01   0.726 0.467589
`Credit amount`              8.584e-06  3.777e-05   0.227 0.820234
Duration                    -3.492e-02  8.984e-03  -3.887 0.000101 ***
Purposecar                  -1.700e-02  2.920e-01  -0.058 0.953568
Purposedomestic appliances  -3.371e-01  7.195e-01  -0.468 0.639441
Purposeeducation            -3.934e-01  4.044e-01  -0.973 0.330646
Purposefurniture/equipment   9.729e-02  3.225e-01   0.302 0.762909
Purposeradio/TV              5.372e-01  3.090e-01   1.739 0.082050 .
Purposerepairs              -3.852e-01  5.858e-01  -0.658 0.510851
Purposevacation/others      -2.900e-01  7.586e-01  -0.382 0.702274
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 963.44  on 799  degrees of freedom
Residual deviance: 891.06  on 779  degrees of freedom
AIC: 933.06

Number of Fisher Scoring iterations: 4
```

- use the model1 to make predictions on the testData

Code



```r
```{r}
pred <- predict(model1, newdata = testData, type = "response")
pred <- ifelse(pred > 0.5, "1","0")
pred
```
```

```
  1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51
"1" "0" "0" "1" "1" "1" "1" "1" "1" "1" "0" "1" "1" "1" "1" "1" "1" "0" "1" "1" "1" "1" "1" "1" "1" "1" "1" "1" "0" "1" "1" "1" "1" "1" "1" "1" "1" "1" "1" "1" "1" "1" "1" "0" "1" "1" "1" "1" "1" "1" "1"
 52  53  54  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72  73  74  75  76  77  78  79  80  81  82  83  84  85  86  87  88  89  90  91  92  93  94  95  96  97  98  99 100 101 102
"1" "1" "1" "1" "1" "1" "1" "1" "1" "1" "0" "1" "1" "1" "1" "1" "1" "0" "1" "1" "1" "1" "1" "1" "1" "1" "1" "1" "1" "1" "1" "1" "1" "1" "1" "1" "1" "1" "1" "1" "1" "1" "1" "1" "1" "1" "1" "1" "1" "1" "1"
103 104 105 106 107 108 109 110 111 112 113 114 115 116 117 118 119 120 121 122 123 124 125 126 127 128 129 130 131 132 133 134 135 136 137 138 139 140 141 142 143 144 145 146 147 148 149 150 151 152 153
"0" "1" "1" "1" "1" "1" "1" "1" "1" "1" "1" "1" "1" "0" "1" "1" "1" "1" "1" "1" "1" "1" "0" "1" "1" "1" "1" "1" "1" "1" "1" "1" "1" "1" "0" "0" "0" "1" "1" "1" "1" "1" "1" "1" "1" "1" "1" "1" "1" "1" "1"
154 155 156 157 158 159 160 161 162 163 164 165 166 167 168 169 170 171 172 173 174 175 176 177 178 179 180 181 182 183 184 185 186 187 188 189 190 191 192 193 194 195 196 197 198 199 200
"1" "0" "1" "1" "1" "1" "1" "0" "1" "1" "1" "1" "1" "1" "1" "1" "1" "1" "1" "1" "1" "1" "1" "1" "1" "0" "1" "0" "1" "1" "1" "1" "1" "1" "1" "0" "1" "0" "0" "1" "1" "1" "1" "1" "0" "0"
```

- Evaluate the model performance by use of confusion matrix the model has an accuracy of 69.5% hence we can consider this as a good model to predict credit risk

Code

```r
```{r}

library(caret)
pred <- as.factor(pred)
pred
testData1Risk <-as.factor(testData1Risk)
testData1Risk
confusionMatrix(testData1Risk,pred)

```
```

```
Confusion Matrix and Statistics

          Reference
Prediction   0   1
         0  15  53
         1   8 124

               Accuracy : 0.695
                 95% CI : (0.6261, 0.758)
    No Information Rate : 0.885
    P-Value [Acc > NIR] : 1

                  Kappa : 0.1906

 Mcnemar's Test P-Value : 1.765e-08

            Sensitivity : 0.6522
            Specificity : 0.7006
         Pos Pred Value : 0.2206
         Neg Pred Value : 0.9394
             Prevalence : 0.1150
         Detection Rate : 0.0750
   Detection Prevalence : 0.3400
      Balanced Accuracy : 0.6764

       'Positive' Class : 0
```