I. Dataset Preprocessing _ Harriet Onoriode Otomiewor (23103939)

2023-08-19

Read data

```
setwd('C:/Users/tinhl/OneDrive/Documents')
data <- read.csv(file ='WA_Fn-UseC_-Marketing-Customer-Value-Analysis.csv')
head(data)
```

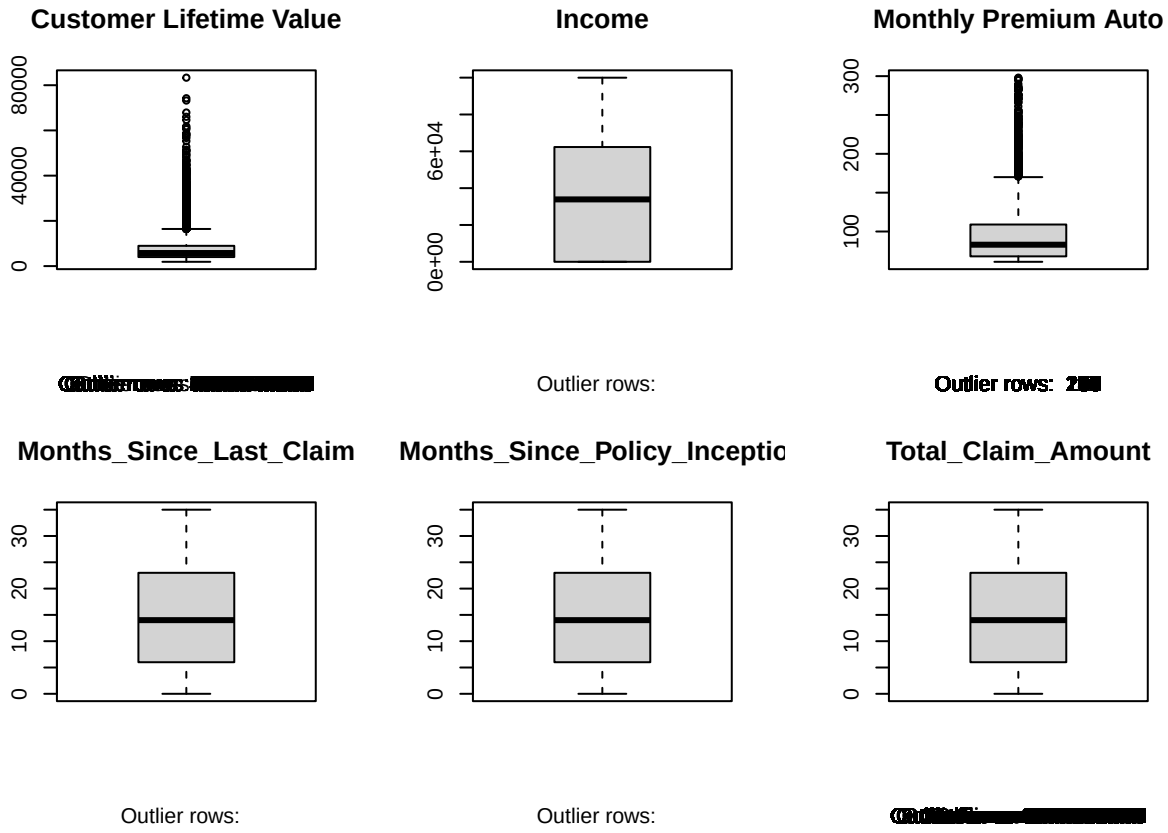
```
##   Customer      State Customer_Lifetime_Value Response Coverage Education
## 1  BU79786 Washington                2763.519       No    Basic  Bachelor
## 2  QZ44356    Arizona                6979.536       No Extended  Bachelor
## 3  AI49188     Nevada               12887.432       No  Premium  Bachelor
## 4  WW63253 California                7645.862       No    Basic  Bachelor
## 5  HB64268 Washington                2813.693       No    Basic  Bachelor
## 6  OC83172     Oregon                8256.298      Yes    Basic  Bachelor
##   Effective_To_Date EmploymentStatus Gender Income Location_Code Marital_Status
## 1         2/24/2011         Employed      F  56274      Suburban        Married
## 2         1/31/2011       Unemployed      F      0      Suburban         Single
## 3         2/19/2011         Employed      F  48767      Suburban        Married
## 4         1/20/2011       Unemployed      M      0      Suburban        Married
## 5          2/3/2011         Employed      M  43836         Rural         Single
## 6         1/25/2011         Employed      F  62902         Rural        Married
##   Monthly_Premium_Auto Months_Since_Last_Claim Months_Since_Policy_Inception
## 1                   69                     32                             5
## 2                   94                     13                            42
## 3                  108                     18                            38
## 4                  106                     18                            65
## 5                   73                     12                            44
## 6                   69                     14                            94
##   Number_of_Open_Complaints Number_of_Policies    Policy_Type       Policy
## 1                         0                  1 Corporate Auto Corporate L3
## 2                         0                  8  Personal Auto  Personal L3
## 3                         0                  2  Personal Auto  Personal L3
## 4                         0                  7 Corporate Auto Corporate L2
## 5                         0                  1  Personal Auto  Personal L1
## 6                         0                  2  Personal Auto  Personal L3
##   Renew_Offer_Type Sales_Channel Total_Claim_Amount Vehicle_Class Vehicle_Size
## 1           Offer1         Agent           384.8111  Two-Door Car      Medsize
## 2           Offer3         Agent          1131.4649 Four-Door Car      Medsize
## 3           Offer1         Agent           566.4722  Two-Door Car      Medsize
## 4           Offer1   Call Center           529.8813           SUV      Medsize
## 5           Offer1         Agent           138.1309 Four-Door Car      Medsize
## 6           Offer2           Web           159.3830  Two-Door Car      Medsize
```

```r
summary(data)
```

```
##    Customer            State           Customer_Lifetime_Value
##  Length:9134        Length:9134        Min.    : 1898
##  Class :character   Class :character   1st Qu.: 3994
##  Mode  :character   Mode  :character   Median : 5780
##                                        Mean    : 8005
##                                        3rd Qu.: 8962
##                                        Max.    :83325
##    Response            Coverage          Education         Effective_To_Date
##  Length:9134        Length:9134        Length:9134        Length:9134
##  Class :character   Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
##
##
##
##  EmploymentStatus      Gender              Income      Location_Code
##  Length:9134        Length:9134        Min.    :    0   Length:9134
##  Class :character   Class :character   1st Qu.:    0   Class :character
##  Mode  :character   Mode  :character   Median :33890   Mode  :character
##                                        Mean   :37657
##                                        3rd Qu.:62320
##                                        Max.   :99981
##  Marital_Status     Monthly_Premium_Auto Months_Since_Last_Claim
##  Length:9134        Min.   : 61.00       Min.   : 0.0
##  Class :character   1st Qu.: 68.00       1st Qu.: 6.0
##  Mode  :character   Median : 83.00       Median :14.0
##                     Mean   : 93.22       Mean   :15.1
##                     3rd Qu.:109.00       3rd Qu.:23.0
##                     Max.   :298.00       Max.   :35.0
##  Months_Since_Policy_Inception Number_of_Open_Complaints Number_of_Policies
##  Min.   : 0.00                 Min.   :0.0000           Min.   :1.000
##  1st Qu.:24.00                 1st Qu.:0.0000           1st Qu.:1.000
##  Median :48.00                 Median :0.0000           Median :2.000
##  Mean   :48.06                 Mean   :0.3844           Mean   :2.966
##  3rd Qu.:71.00                 3rd Qu.:0.0000           3rd Qu.:4.000
##  Max.   :99.00                 Max.   :5.0000           Max.   :9.000
##  Policy_Type           Policy           Renew_Offer_Type   Sales_Channel
##  Length:9134        Length:9134        Length:9134        Length:9134
##  Class :character   Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
##
##
##
##  Total_Claim_Amount Vehicle_Class      Vehicle_Size
##  Min.   :   0.099   Length:9134        Length:9134
##  1st Qu.: 272.258   Class :character   Class :character
##  Median : 383.945   Mode  :character   Mode  :character
##  Mean   : 434.089
##  3rd Qu.: 547.515
##  Max.   :2893.240
```

Boxplot to check for outliers

```
par(mfrow=c(2,3))
boxplot(data$Customer_Lifetime_Value, main="Customer Lifetime Value",
        sub=paste("Outlier rows: ", boxplot.stats(data$Customer_Lifetime_Value)$out))
boxplot(data$Income, main="Income",
        sub=paste("Outlier rows: ", boxplot.stats(data$Income)$out))
boxplot(data$Monthly_Premium_Auto, main="Monthly Premium Auto",
        sub=paste("Outlier rows: ", boxplot.stats(data$Monthly_Premium_Auto)$out))
boxplot(data$Months_Since_Last_Claim, main="Months_Since_Last_Claim",
        sub=paste("Outlier rows: ", boxplot.stats(data$Months_Since_Last_Claim)$out))
boxplot(data$Months_Since_Last_Claim, main="Months_Since_Policy_Inception",
        sub=paste("Outlier rows: ", boxplot.stats(data$Months_Since_Policy_Inception)$out))
boxplot(data$Months_Since_Last_Claim, main="Total_Claim_Amount",
        sub=paste("Outlier rows: ", boxplot.stats(data$Total_Claim_Amount)$out))
```



Remove Outliers Customer_Lifetime_Value

Filter data with data$Customer_Lifetime_Value <= 52000

```
data <- data[data$Customer_Lifetime_Value <= 52000,]
head(data[order(-data$Customer_Lifetime_Value),], 10)
```

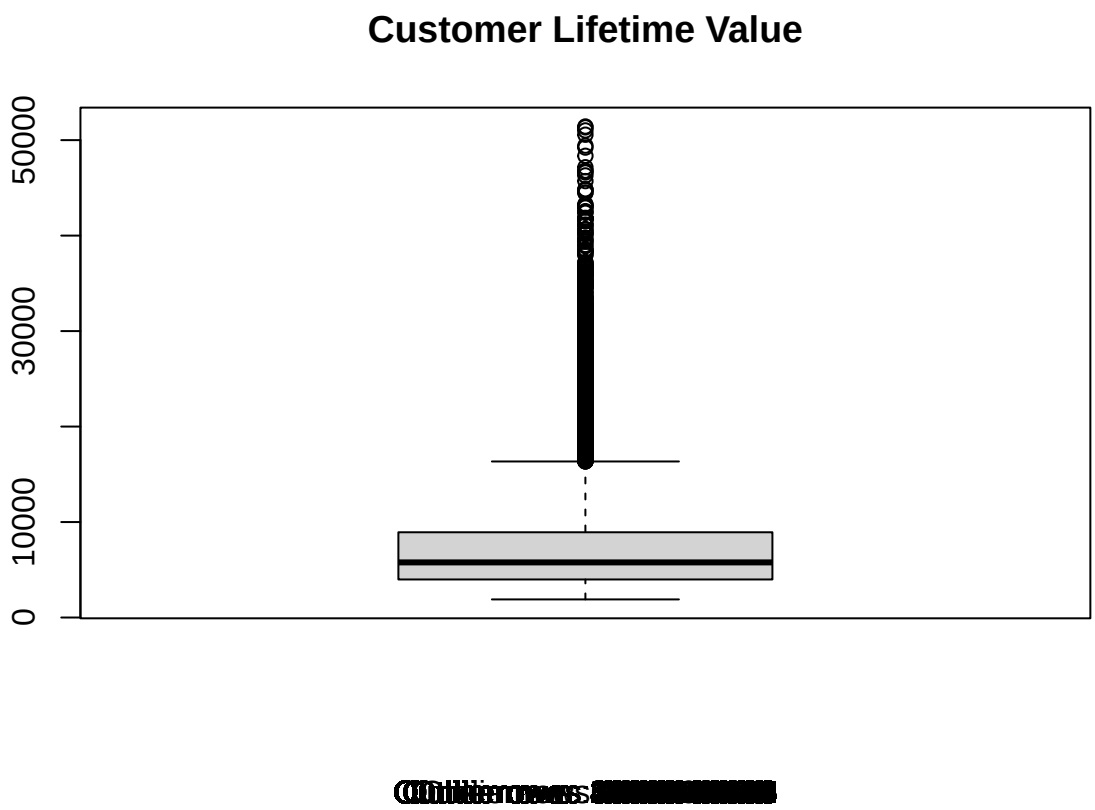
```
##      Customer      State Customer_Lifetime_Value Response Coverage
## 6555  AH58807    Arizona                51426.25       No    Basic
```

```
## 2191  KI58952 California               51337.91       No  Premium
## 6570  LW64678 California               51016.07       No  Premium
## 7836  QT84069     Oregon               50568.26       No Extended
## 6978  BR50492    Arizona               49423.80       No Extended
## 1559  RP30093     Oregon               49221.43       No  Premium
## 1813  LU42720     Nevada               48356.96       No Extended
## 3458  MJ77630     Oregon               47155.63       No Extended
## 1859  CP92616     Nevada               46805.22       No Extended
## 3211  KB44286     Oregon               46770.95       No    Basic
##                 Education Effective_To_Date EmploymentStatus Gender Income
## 6555              College          1/9/2011         Employed      F  84650
## 2191              College         2/24/2011         Employed      F  72794
## 6570               Master         2/19/2011         Employed      F  25167
## 7836               Master         2/28/2011         Employed      M  82081
## 6978             Bachelor          1/4/2011         Employed      M  85058
## 1559             Bachelor         1/23/2011         Employed      F  63035
## 1813              College         2/20/2011         Employed      M  52499
## 3458 High School or Below         2/10/2011         Employed      M  39891
## 1859 High School or Below         2/25/2011         Employed      M  83006
## 3211 High School or Below          2/1/2011         Employed      F  64403
##      Location_Code Marital_Status Monthly_Premium_Auto Months_Since_Last_Claim
## 6555         Urban        Married                 185                      13
## 2191         Rural         Single                 164                       3
## 6570         Urban        Married                 140                       3
## 7836         Urban        Married                 249                       1
## 6978         Urban        Married                 137                      34
## 1559      Suburban        Married                 153                      20
## 1813      Suburban       Divorced                 138                       0
## 3458         Urban        Married                 133                      12
## 1859         Urban        Married                 235                       8
## 3211         Rural         Single                 198                      11
##      Months_Since_Policy_Inception Number_of_Open_Complaints Number_of_Policies
## 6555                            39                         3                 2
## 2191                            47                         1                 2
## 6570                            76                         0                 2
## 7836                            62                         0                 2
## 6978                            82                         0                 2
## 1559                            97                         0                 2
## 1813                            61                         0                 2
## 3458                            31                         0                 2
## 1859                            61                         1                 2
## 3211                            86                         0                 2
##        Policy_Type       Policy Renew_Offer_Type Sales_Channel
## 6555  Personal Auto  Personal L2           Offer1         Agent
## 2191  Personal Auto  Personal L2           Offer1           Web
## 6570  Personal Auto  Personal L3           Offer2         Agent
## 7836  Personal Auto  Personal L1           Offer2        Branch
## 6978  Personal Auto  Personal L1           Offer1   Call Center
## 1559  Personal Auto  Personal L3           Offer1         Agent
## 1813  Personal Auto  Personal L2           Offer3   Call Center
## 3458  Personal Auto  Personal L3           Offer1         Agent
## 1859  Personal Auto  Personal L1           Offer3         Agent
## 3211 Corporate Auto Corporate L1           Offer1         Agent
##      Total_Claim_Amount Vehicle_Class Vehicle_Size
```

```
## 6555          660.47427    Luxury Car      Medsize
## 2191           50.45446          SUV        Large
## 6570          422.49429          SUV        Small
## 7836          753.76010    Luxury SUV        Small
## 6978          595.36978          SUV      Medsize
## 1559          734.40000          SUV      Medsize
## 1813         1171.53759          SUV      Medsize
## 3458          630.88866          SUV      Medsize
## 1859         1065.04989    Luxury SUV        Small
## 3211          111.17302    Luxury SUV        Small
```

Box plot for "Customer Lifetime Value"

```
boxplot(data$Customer_Lifetime_Value, main="Customer Lifetime Value",
        sub=paste("Outlier rows: ", boxplot.stats(data$Customer_Lifetime_Value)$out))
```



Identify character features

```
character_cols <- sapply(data, is.character)
head(data[, character_cols])
```

```
##   Customer      State Response Coverage Education Effective_To_Date
## 1  BU79786 Washington       No    Basic  Bachelor         2/24/2011
## 2  QZ44356    Arizona       No Extended  Bachelor         1/31/2011
## 3  AI49188     Nevada       No  Premium  Bachelor         2/19/2011
## 4  WW63253 California       No    Basic  Bachelor         1/20/2011
```

```
## 5  HB64268 Washington       No    Basic  Bachelor          2/3/2011
## 6  OC83172     Oregon      Yes    Basic  Bachelor         1/25/2011
##   EmploymentStatus Gender Location_Code Marital_Status    Policy_Type
## 1         Employed      F      Suburban        Married Corporate Auto
## 2       Unemployed      F      Suburban         Single  Personal Auto
## 3         Employed      F      Suburban        Married  Personal Auto
## 4       Unemployed      M      Suburban        Married Corporate Auto
## 5         Employed      M         Rural         Single  Personal Auto
## 6         Employed      F         Rural        Married  Personal Auto
##         Policy Renew_Offer_Type Sales_Channel Vehicle_Class Vehicle_Size
## 1 Corporate L3           Offer1         Agent  Two-Door Car      Medsize
## 2  Personal L3           Offer3         Agent Four-Door Car      Medsize
## 3  Personal L3           Offer1         Agent  Two-Door Car      Medsize
## 4 Corporate L2           Offer1   Call Center           SUV      Medsize
## 5  Personal L1           Offer1         Agent Four-Door Car      Medsize
## 6  Personal L3           Offer2           Web  Two-Door Car      Medsize
```

Check unique value

Print the unique values State

```
unique(data$State)
```

```
## [1] "Washington" "Arizona"    "Nevada"     "California" "Oregon"
```

Print the unique values Response

```
unique(data$Response)
```

```
## [1] "No"  "Yes"
```

Print the unique values Coverage

```
unique(data$Coverage)
```

```
## [1] "Basic"    "Extended" "Premium"
```

Print the unique values Education

```
unique(data$Education)
```

```
## [1] "Bachelor"             "College"              "Master"
## [4] "High School or Below" "Doctor"
```

Print the unique values EmploymentStatus

```
unique(data$EmploymentStatus)
```

```
## [1] "Employed"      "Unemployed"    "Medical Leave" "Disabled"
## [5] "Retired"
```

Print the unique values Gender

```
unique(data$Gender)
```

```
## [1] "F" "M"
```

Print the unique values Location_Code

```
unique(data$Location_Code)
```

```
## [1] "Suburban" "Rural"    "Urban"
```

Print the unique values Marital_Status

```
unique(data$Marital_Status)
```

```
## [1] "Married"  "Single"   "Divorced"
```

Print the unique values Policy_Type

```
unique(data$Policy_Type)
```

```
## [1] "Corporate Auto" "Personal Auto"  "Special Auto"
```

Print the unique values Policy

```
unique(data$Policy)
```

```
## [1] "Corporate L3" "Personal L3"  "Corporate L2" "Personal L1"  "Special L2"
## [6] "Corporate L1" "Personal L2"  "Special L1"   "Special L3"
```

Print the unique values Renew_Offer_Type

```
unique(data$Renew_Offer_Type)
```

```
## [1] "Offer1" "Offer3" "Offer2" "Offer4"
```

Print the unique values Sales_Channel

```
unique(data$Sales_Channel)
```

```
## [1] "Agent"       "Call Center" "Web"         "Branch"
```

Print the unique values Vehicle_Class

```r
unique(data$Vehicle_Class)
```

```
## [1] "Two-Door Car"  "Four-Door Car" "SUV"           "Luxury SUV"
## [5] "Sports Car"    "Luxury Car"
```

Print the unique values Vehicle_Size

```r
unique(data$Vehicle_Size)
```

```
## [1] "Medsize" "Small"   "Large"
```

Convert Character to Numeric

Response: No- 0, Yes- 1

```r
data$Response <- ifelse(data$Response == "No", 0, 1)
```

State

```r
data$State <- as.numeric(factor(data$State))
```

EmploymentStatus

```r
data$EmploymentStatus <- as.numeric(factor(data$EmploymentStatus))
```

Gender

```r
data$Gender <- as.numeric(factor(data$Gender))
```

Marital_Status

```r
data$Marital_Status <- as.numeric(factor(data$Marital_Status))
```

Sales_Channel

```r
data$Sales_Channel <- as.numeric(factor(data$Sales_Channel))
```

Vehicle_Class

```r
data$Vehicle_Class <- as.numeric(factor(data$Vehicle_Class))
```

Coverage (“Premium”,“Extended”,“Basic”)

```r
data$Coverage <- as.numeric(factor(data$Coverage, levels = c("Premium","Extended","Basic")))
```

Education (“Doctor”,“Master”,“Bachelor”,“College”,“High School or Below”)

```
data$Education <- as.numeric(factor(data$Education, levels = c("Doctor","Master","Bachelor","College","
```

Location_Code (“Urban”, “Suburban”, “Rural”)

```
data$Location_Code <- as.numeric(factor(data$Location_Code, levels = c("Urban", "Suburban", "Rural")))
```

Policy_Type (“Special Auto”, “Corporate Auto”, “Personal Auto”)

```
data$Policy_Type <- as.numeric(factor(data$Policy_Type, levels = c("Special Auto", "Corporate Auto", "P
```

Policy (“Special L1”, “Special L2”, “Special L3”, “Corporate L1”, “Corporate L2”, “Corporate L3”, “Personal L1”, “Personal L2”, “Personal L3”)

```
data$Policy <- as.numeric(factor(data$Policy, levels = c("Special L1", "Special L2", "Special L3", "Cor
```

Renew_Offer_Type (“Offer1”, “Offer2”, “Offer3”, “Offer4”)

```
data$Renew_Offer_Type <- as.numeric(factor(data$Renew_Offer_Type, levels = c("Offer1", "Offer2", "Offer
```

Vehicle_Size (“Small”, “Medsize”, “Large”)

```
data$Vehicle_Size <- as.numeric(factor(data$Vehicle_Size, levels = c("Small", "Medsize", "Large")))
```

```
summary(data)
```

```
##    Customer             State       Customer_Lifetime_Value    Response
##  Length:9118        Min.   :1.000   Min.   : 1898           Min.   :0.0000
##  Class :character   1st Qu.:2.000   1st Qu.: 3985           1st Qu.:0.0000
##  Mode  :character   Median :2.000   Median : 5774           Median :0.0000
##                     Mean   :2.741   Mean   : 7908           Mean   :0.1435
##                     3rd Qu.:4.000   3rd Qu.: 8930           3rd Qu.:0.0000
##                     Max.   :5.000   Max.   :51426           Max.   :1.0000
##     Coverage       Education     Effective_To_Date  EmploymentStatus
##  Min.   :1.00   Min.   :1.000   Length:9118        Min.   :1.000
##  1st Qu.:2.00   1st Qu.:3.000   Class :character   1st Qu.:2.000
##  Median :3.00   Median :4.000   Mode  :character   Median :2.000
##  Mean   :2.52   Mean   :3.711                      Mean   :2.825
##  3rd Qu.:3.00   3rd Qu.:5.000                      3rd Qu.:5.000
##  Max.   :3.00   Max.   :5.000                      Max.   :5.000
##      Gender         Income      Location_Code   Marital_Status
##  Min.   :1.00   Min.   :    0   Min.   :1.000   Min.   :1.00
##  1st Qu.:1.00   1st Qu.:    0   1st Qu.:2.000   1st Qu.:2.00
##  Median :1.00   Median :33899   Median :2.000   Median :2.00
##  Mean   :1.49   Mean   :37669   Mean   :2.021   Mean   :2.12
##  3rd Qu.:2.00   3rd Qu.:62358   3rd Qu.:2.000   3rd Qu.:3.00
##  Max.   :2.00   Max.   :99981   Max.   :3.000   Max.   :3.00
##  Monthly_Premium_Auto Months_Since_Last_Claim Months_Since_Policy_Inception
##  Min.   : 61.00       Min.   : 0.00           Min.   : 0.00
##  1st Qu.: 68.00       1st Qu.: 6.00           1st Qu.:24.00
##  Median : 83.00       Median :14.00           Median :48.00
```

```
##  Mean   : 93.02       Mean   :15.09           Mean   :48.06
##  3rd Qu.:109.00       3rd Qu.:23.00           3rd Qu.:71.00
##  Max.   :298.00       Max.   :35.00           Max.   :99.00
##  Number_of_Open_Complaints Number_of_Policies  Policy_Type        Policy
##  Min.   :0.0000            Min.   :1.000      Min.   :1.000   Min.   :1.000
##  1st Qu.:0.0000            1st Qu.:1.000      1st Qu.:2.000   1st Qu.:6.000
##  Median :0.0000            Median :2.000      Median :3.000   Median :8.000
##  Mean   :0.3847            Mean   :2.968      Mean   :2.701   Mean   :7.424
##  3rd Qu.:0.0000            3rd Qu.:4.000      3rd Qu.:3.000   3rd Qu.:9.000
##  Max.   :5.0000            Max.   :9.000      Max.   :3.000   Max.   :9.000
##  Renew_Offer_Type Sales_Channel   Total_Claim_Amount Vehicle_Class
##  Min.   :1.000    Min.   :1.000   Min.   :   0.099   Min.   :1.000
##  1st Qu.:1.000    1st Qu.:1.000   1st Qu.: 271.983   1st Qu.:1.000
##  Median :2.000    Median :2.000   Median : 383.296   Median :1.000
##  Mean   :1.971    Mean   :2.103   Mean   : 432.906   Mean   :3.036
##  3rd Qu.:3.000    3rd Qu.:3.000   3rd Qu.: 547.200   3rd Qu.:5.000
##  Max.   :4.000    Max.   :4.000   Max.   :2893.240   Max.   :6.000
##   Vehicle_Size
##  Min.   :1.00
##  1st Qu.:2.00
##  Median :2.00
##  Mean   :1.91
##  3rd Qu.:2.00
##  Max.   :3.00
```

Scale Data

```
columns_to_scale <- c(
  "Customer_Lifetime_Value", "Income", "Monthly_Premium_Auto",
  "Months_Since_Last_Claim", "Months_Since_Policy_Inception",
  "Number_of_Open_Complaints", "Number_of_Policies", "Total_Claim_Amount"
)

scale_data <- scale(data[columns_to_scale])
head(scale_data)
```

```
##   Customer_Lifetime_Value     Income Monthly_Premium_Auto
## 1             -0.79535661  0.6123357           -0.7047628
## 2             -0.14356772 -1.2398258            0.0288120
## 3              0.76978278  0.3652558            0.4396139
## 4             -0.04055488 -1.2398258            0.3809279
## 5             -0.78759991  0.2029605           -0.5873909
## 6              0.05381747  0.8304849           -0.7047628
##   Months_Since_Last_Claim Months_Since_Policy_Inception
## 1               1.6791513                    -1.5425994
## 2              -0.2079686                    -0.2171227
## 3               0.2886419                    -0.3604175
## 4               0.2886419                     0.6068223
## 5              -0.3072907                    -0.1454753
## 6              -0.1086465                     1.6457094
##   Number_of_Open_Complaints Number_of_Policies Total_Claim_Amount
## 1                -0.4223209         -0.8227086         -0.1666108
```

```
## 2               -0.4223209          2.1037920          2.4199345
## 3               -0.4223209         -0.4046371          0.4626965
## 4               -0.4223209          1.6857205          0.3359389
## 5               -0.4223209         -0.8227086         -1.0211564
## 6               -0.4223209         -0.4046371         -0.9475350
```

```
data <- data[, !(names(data) %in% columns_to_scale)]

head(data)
```

```
##   Customer State Response Coverage Education Effective_To_Date EmploymentStatus
## 1  BU79786     5        0        3         3         2/24/2011                2
## 2  QZ44356     1        0        2         3         1/31/2011                5
## 3  AI49188     3        0        1         3         2/19/2011                2
## 4  WW63253     2        0        3         3         1/20/2011                5
## 5  HB64268     5        0        3         3          2/3/2011                2
## 6  OC83172     4        1        3         3         1/25/2011                2
##   Gender Location_Code Marital_Status Policy_Type Policy Renew_Offer_Type
## 1      1             2              2           2      6                1
## 2      1             2              3           3      9                3
## 3      1             2              2           3      9                1
## 4      2             2              2           2      5                1
## 5      2             3              3           3      7                1
## 6      1             3              2           3      9                2
##   Sales_Channel Vehicle_Class Vehicle_Size
## 1             1             6            2
## 2             1             1            2
## 3             1             6            2
## 4             3             5            2
## 5             1             1            2
## 6             4             6            2
```

Combine Data & Scale Data

```
df <- cbind(data, scale_data)
head(df)
```

```
##   Customer State Response Coverage Education Effective_To_Date EmploymentStatus
## 1  BU79786     5        0        3         3         2/24/2011                2
## 2  QZ44356     1        0        2         3         1/31/2011                5
## 3  AI49188     3        0        1         3         2/19/2011                2
## 4  WW63253     2        0        3         3         1/20/2011                5
## 5  HB64268     5        0        3         3          2/3/2011                2
## 6  OC83172     4        1        3         3         1/25/2011                2
##   Gender Location_Code Marital_Status Policy_Type Policy Renew_Offer_Type
## 1      1             2              2           2      6                1
## 2      1             2              3           3      9                3
## 3      1             2              2           3      9                1
## 4      2             2              2           2      5                1
## 5      2             3              3           3      7                1
## 6      1             3              2           3      9                2
```

```
##   Sales_Channel Vehicle_Class Vehicle_Size Customer_Lifetime_Value     Income
## 1             1             6            2             -0.79535661  0.6123357
## 2             1             1            2             -0.14356772 -1.2398258
## 3             1             6            2              0.76978278  0.3652558
## 4             3             5            2             -0.04055488 -1.2398258
## 5             1             1            2             -0.78759991  0.2029605
## 6             4             6            2              0.05381747  0.8304849
##   Monthly_Premium_Auto Months_Since_Last_Claim Months_Since_Policy_Inception
## 1           -0.7047628              1.6791513                     -1.5425994
## 2            0.0288120             -0.2079686                     -0.2171227
## 3            0.4396139              0.2886419                     -0.3604175
## 4            0.3809279              0.2886419                      0.6068223
## 5           -0.5873909             -0.3072907                     -0.1454753
## 6           -0.7047628             -0.1086465                      1.6457094
##   Number_of_Open_Complaints Number_of_Policies Total_Claim_Amount
## 1                -0.4223209         -0.8227086         -0.1666108
## 2                -0.4223209          2.1037920          2.4199345
## 3                -0.4223209         -0.4046371          0.4626965
## 4                -0.4223209          1.6857205          0.3359389
## 5                -0.4223209         -0.8227086         -1.0211564
## 6                -0.4223209         -0.4046371         -0.9475350
```

```
dim(df)
```

```
## [1] 9118   24
```

```
summary(df)
```

```
##    Customer             State          Response         Coverage
##  Length:9118        Min.   :1.000   Min.   :0.0000   Min.   :1.00
##  Class :character   1st Qu.:2.000   1st Qu.:0.0000   1st Qu.:2.00
##  Mode  :character   Median :2.000   Median :0.0000   Median :3.00
##                     Mean   :2.741   Mean   :0.1435   Mean   :2.52
##                     3rd Qu.:4.000   3rd Qu.:0.0000   3rd Qu.:3.00
##                     Max.   :5.000   Max.   :1.0000   Max.   :3.00
##    Education     Effective_To_Date  EmploymentStatus     Gender
##  Min.   :1.000   Length:9118        Min.   :1.000    Min.   :1.00
##  1st Qu.:3.000   Class :character   1st Qu.:2.000    1st Qu.:1.00
##  Median :4.000   Mode  :character   Median :2.000    Median :1.00
##  Mean   :3.711                      Mean   :2.825    Mean   :1.49
##  3rd Qu.:5.000                      3rd Qu.:5.000    3rd Qu.:2.00
##  Max.   :5.000                      Max.   :5.000    Max.   :2.00
##  Location_Code   Marital_Status  Policy_Type        Policy
##  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.:6.000
##  Median :2.000   Median :2.00   Median :3.000   Median :8.000
##  Mean   :2.021   Mean   :2.12   Mean   :2.701   Mean   :7.424
##  3rd Qu.:2.000   3rd Qu.:3.00   3rd Qu.:3.000   3rd Qu.:9.000
##  Max.   :3.000   Max.   :3.00   Max.   :3.000   Max.   :9.000
##  Renew_Offer_Type Sales_Channel   Vehicle_Class    Vehicle_Size
##  Min.   :1.000    Min.   :1.000   Min.   :1.000   Min.   :1.00
##  1st Qu.:1.000    1st Qu.:1.000   1st Qu.:1.000   1st Qu.:2.00
##  Median :2.000    Median :2.000   Median :1.000   Median :2.00
```

```
##  Mean   :1.971    Mean   :2.103   Mean   :3.036   Mean   :1.91
##  3rd Qu.:3.000    3rd Qu.:3.000   3rd Qu.:5.000   3rd Qu.:2.00
##  Max.   :4.000    Max.   :4.000   Max.   :6.000   Max.   :3.00
##  Customer_Lifetime_Value     Income        Monthly_Premium_Auto
##  Min.   :-0.9292         Min.   :-1.2398   Min.   :-0.9395
##  1st Qu.:-0.6065         1st Qu.:-1.2398   1st Qu.:-0.7341
##  Median :-0.3300         Median :-0.1241   Median :-0.2940
##  Mean   : 0.0000         Mean   : 0.0000   Mean   : 0.0000
##  3rd Qu.: 0.1580         3rd Qu.: 0.8126   3rd Qu.: 0.4690
##  Max.   : 6.7278         Max.   : 2.0509   Max.   : 6.0148
##  Months_Since_Last_Claim Months_Since_Policy_Inception
##  Min.   :-1.4992         Min.   :-1.721718
##  1st Qu.:-0.9032         1st Qu.:-0.861949
##  Median :-0.1086         Median :-0.002181
##  Mean   : 0.0000         Mean   : 0.000000
##  3rd Qu.: 0.7853         3rd Qu.: 0.821764
##  Max.   : 1.9771         Max.   : 1.824828
##  Number_of_Open_Complaints Number_of_Policies Total_Claim_Amount
##  Min.   :-0.4223           Min.   :-0.8227    Min.   :-1.4993
##  1st Qu.:-0.4223           1st Qu.:-0.8227    1st Qu.:-0.5575
##  Median :-0.4223           Median :-0.4046    Median :-0.1719
##  Mean   : 0.0000           Mean   : 0.0000    Mean   : 0.0000
##  3rd Qu.:-0.4223           3rd Qu.: 0.4315    3rd Qu.: 0.3959
##  Max.   : 5.0662           Max.   : 2.5219    Max.   : 8.5230
```

Identify numeric features

```
numeric_cols <- sapply(df, is.numeric)
df <- df[, numeric_cols]
dim(df)
```

```
## [1] 9118   22
```

```
head(df)
```

```
##   State Response Coverage Education EmploymentStatus Gender Location_Code
## 1     5        0        3         3                2      1             2
## 2     1        0        2         3                5      1             2
## 3     3        0        1         3                2      1             2
## 4     2        0        3         3                5      2             2
## 5     5        0        3         3                2      2             3
## 6     4        1        3         3                2      1             3
##   Marital_Status Policy_Type Policy Renew_Offer_Type Sales_Channel
## 1              2           2      6                1             1
## 2              3           3      9                3             1
## 3              2           3      9                1             1
## 4              2           2      5                1             3
## 5              3           3      7                1             1
## 6              2           3      9                2             4
##   Vehicle_Class Vehicle_Size Customer_Lifetime_Value     Income
## 1             6            2             -0.79535661  0.6123357
```

```
## 2             1           2             -0.14356772 -1.2398258
## 3             6           2              0.76978278  0.3652558
## 4             5           2             -0.04055488 -1.2398258
## 5             1           2             -0.78759991  0.2029605
## 6             6           2              0.05381747  0.8304849
##   Monthly_Premium_Auto Months_Since_Last_Claim Months_Since_Policy_Inception
## 1           -0.7047628              1.6791513                    -1.5425994
## 2            0.0288120             -0.2079686                    -0.2171227
## 3            0.4396139              0.2886419                    -0.3604175
## 4            0.3809279              0.2886419                     0.6068223
## 5           -0.5873909             -0.3072907                    -0.1454753
## 6           -0.7047628             -0.1086465                     1.6457094
##   Number_of_Open_Complaints Number_of_Policies Total_Claim_Amount
## 1                -0.4223209         -0.8227086         -0.1666108
## 2                -0.4223209          2.1037920          2.4199345
## 3                -0.4223209         -0.4046371          0.4626965
## 4                -0.4223209          1.6857205          0.3359389
## 5                -0.4223209         -0.8227086         -1.0211564
## 6                -0.4223209         -0.4046371         -0.9475350
```

Check Missing Value

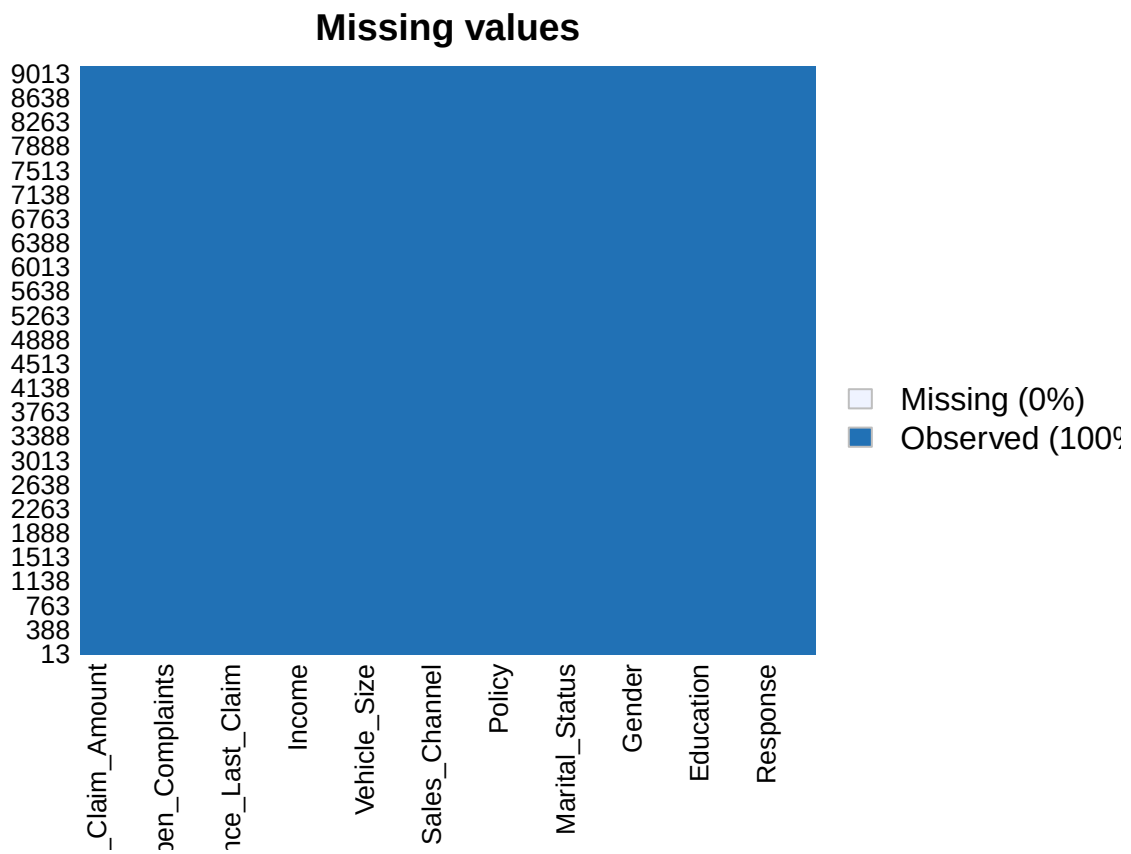
```
library(Amelia)
```

```
## Warning: package 'Amelia' was built under R version 4.2.3
```

```
## Loading required package: Rcpp
```

```
## ##
## ## Amelia II: Multiple Imputation
## ## (Version 1.8.1, built: 2022-11-18)
## ## Copyright (C) 2005-2023 James Honaker, Gary King and Matthew Blackwell
## ## Refer to http://gking.harvard.edu/amelia/ for more information
## ##
```

```
missmap(df, main="Missing values")
```



Save file

```
write.csv(df, file = "data_processed.csv", row.names = FALSE)
```

2. CLUSTERING _ Thi Tinh Lo (22236226)

2023-08-19

Load library

CLUSTERING:

1. PCA

2. PCA & Kmeans

3. PCA & HIERARCHICAL

4. PCA & DBSCAN

Read data

```
setwd('C:/Users/tinhl/OneDrive/Documents')
data <- read.csv(file ='data_processed.csv')
head(data)
```

```
##   State Response Coverage Education EmploymentStatus Gender Location_Code
## 1     5        0        3         3                2      1             2
## 2     1        0        2         3                5      1             2
## 3     3        0        1         3                2      1             2
## 4     2        0        3         3                5      2             2
## 5     5        0        3         3                2      2             3
## 6     4        1        3         3                2      1             3
##   Marital_Status Policy_Type Policy Renew_Offer_Type Sales_Channel
## 1              2           2      6                1             1
## 2              3           3      9                3             1
## 3              2           3      9                1             1
## 4              2           2      5                1             3
## 5              3           3      7                1             1
## 6              2           3      9                2             4
##   Vehicle_Class Vehicle_Size Customer_Lifetime_Value     Income
## 1             6            2             -0.76283596  0.6127939
## 2             1            2             -0.14923729 -1.2395490
## 3             6            2              0.71059732  0.3656898
## 4             5            2             -0.05226027 -1.2395490
## 5             1            2             -0.75553375  0.2033785
## 6             6            2              0.03658252  0.8309644
##   Monthly_Premium_Auto Months_Since_Last_Claim Months_Since_Policy_Inception
## 1          -0.70388612               1.6780075                   -1.5432025
```

```
## 2           0.02268979              -0.2081750                    -0.2173223
## 3           0.42957229               0.2881888                    -0.3606607
## 4           0.37144622               0.2881888                     0.6068735
## 5          -0.58763397              -0.3074478                    -0.1456531
## 6          -0.70388612              -0.1089022                     1.6460769
##   Number_of_Open_Complaints Number_of_Policies Total_Claim_Amount
## 1                -0.4222264         -0.8226028         -0.1696304
## 2                -0.4222264          2.1060447          2.4006056
## 3                -0.4222264         -0.4042246          0.4557088
## 4                -0.4222264          1.6876664          0.3297505
## 5                -0.4222264         -0.8226028         -1.0187877
## 6                -0.4222264         -0.4042246         -0.9456305
```

```
summary(data)
```

```
##      State          Response         Coverage       Education
##  Min.   :1.000   Min.   :0.0000   Min.   :1.000   Min.   :1.000
##  1st Qu.:2.000   1st Qu.:0.0000   1st Qu.:2.000   1st Qu.:3.000
##  Median :2.000   Median :0.0000   Median :3.000   Median :4.000
##  Mean   :2.742   Mean   :0.1432   Mean   :2.519   Mean   :3.712
##  3rd Qu.:4.000   3rd Qu.:0.0000   3rd Qu.:3.000   3rd Qu.:5.000
##  Max.   :5.000   Max.   :1.0000   Max.   :3.000   Max.   :5.000
##  EmploymentStatus     Gender     Location_Code   Marital_Status  Policy_Type
##  Min.   :1.000    Min.   :1.00   Min.   :1.000   Min.   :1.00   Min.   :1.000
##  1st Qu.:2.000    1st Qu.:1.00   1st Qu.:2.000   1st Qu.:2.00   1st Qu.:2.000
##  Median :2.000    Median :1.00   Median :2.000   Median :2.00   Median :3.000
##  Mean   :2.826    Mean   :1.49   Mean   :2.021   Mean   :2.12   Mean   :2.702
##  3rd Qu.:5.000    3rd Qu.:2.00   3rd Qu.:2.000   3rd Qu.:3.00   3rd Qu.:3.000
##  Max.   :5.000    Max.   :2.00   Max.   :3.000   Max.   :3.00   Max.   :3.000
##      Policy      Renew_Offer_Type Sales_Channel   Vehicle_Class
##  Min.   :1.000   Min.   :1.00     Min.   :1.000   Min.   :1.000
##  1st Qu.:6.000   1st Qu.:1.00     1st Qu.:1.000   1st Qu.:1.000
##  Median :8.000   Median :2.00     Median :2.000   Median :1.000
##  Mean   :7.425   Mean   :1.97     Mean   :2.103   Mean   :3.036
##  3rd Qu.:9.000   3rd Qu.:3.00     3rd Qu.:3.000   3rd Qu.:5.000
##  Max.   :9.000   Max.   :4.00     Max.   :4.000   Max.   :6.000
##   Vehicle_Size  Customer_Lifetime_Value     Income        Monthly_Premium_Auto
##  Min.   :1.00   Min.   :-0.8888         Min.   :-1.2395   Min.   :-0.9364
##  1st Qu.:2.00   1st Qu.:-0.5837         1st Qu.:-1.2395   1st Qu.:-0.7329
##  Median :2.00   Median :-0.3238         Median :-0.1240   Median :-0.2970
##  Mean   :1.91   Mean   : 0.0000         Mean   : 0.0000   Mean   : 0.0000
##  3rd Qu.:2.00   3rd Qu.: 0.1393         3rd Qu.: 0.8118   3rd Qu.: 0.4586
##  Max.   :3.00   Max.   :10.9621         Max.   : 2.0515   Max.   : 5.9515
##  Months_Since_Last_Claim Months_Since_Policy_Inception
##  Min.   :-1.4987         Min.   :-1.722376
##  1st Qu.:-0.9031         1st Qu.:-0.862345
##  Median :-0.1089         Median :-0.002315
##  Mean   : 0.0000         Mean   : 0.000000
##  3rd Qu.: 0.7846         3rd Qu.: 0.821881
##  Max.   : 1.9758         Max.   : 1.825250
##  Number_of_Open_Complaints Number_of_Policies Total_Claim_Amount
##  Min.   :-0.4222           Min.   :-0.8226    Min.   :-1.4939
##  1st Qu.:-0.4222           1st Qu.:-0.8226    1st Qu.:-0.5571
##  Median :-0.4222           Median :-0.4042    Median :-0.1726
```

```
##  Mean   : 0.0000           Mean   : 0.0000    Mean   : 0.0000
##  3rd Qu.:-0.4222           3rd Qu.: 0.4325    3rd Qu.: 0.3905
##  Max.   : 5.0700           Max.   : 2.5244    Max.   : 8.4652
```

1. PCA

Load required library

Perform PCA with 2 components

```
pca_2 <- prcomp(data, retx = TRUE, rank = 2)
```

Extract the reduced data with 2 principal components

```
reduced_data <- as.matrix(pca_2$x)
head(reduced_data)
```

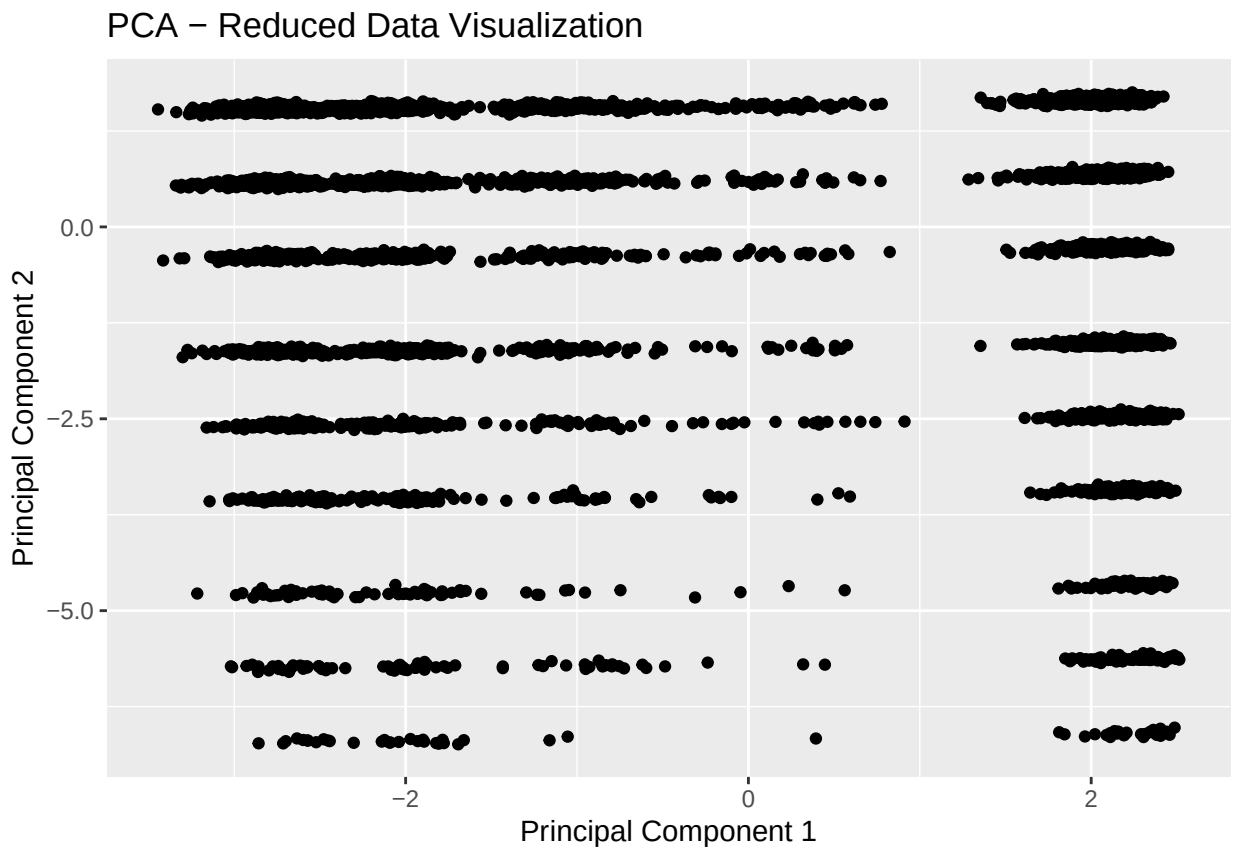
```
##            PC1        PC2
## [1,] -2.708670 -1.6141241
## [2,]  1.689366  1.6235686
## [3,] -3.124832  1.5136796
## [4,] -2.008754 -2.5456237
## [5,]  2.212617 -0.2484373
## [6,] -2.717361  1.5577718
```

PLOT THE COMPONENTS Convert the reduced_data matrix to a data frame

```
reduced_data_df <- as.data.frame(reduced_data)
```

Create a scatter plot of the reduced data

```
ggplot(data = reduced_data_df, aes(x = PC1, y = PC2)) +
  geom_point() +
  labs(x = "Principal Component 1", y = "Principal Component 2", title = "PCA - Reduced Data Visualizati
```



2. PCA & Kmeans

Determine number of clusters:

Select k on reduced data: wss cho ca bo du lieu (1 cum, 2 thuoc tinh)

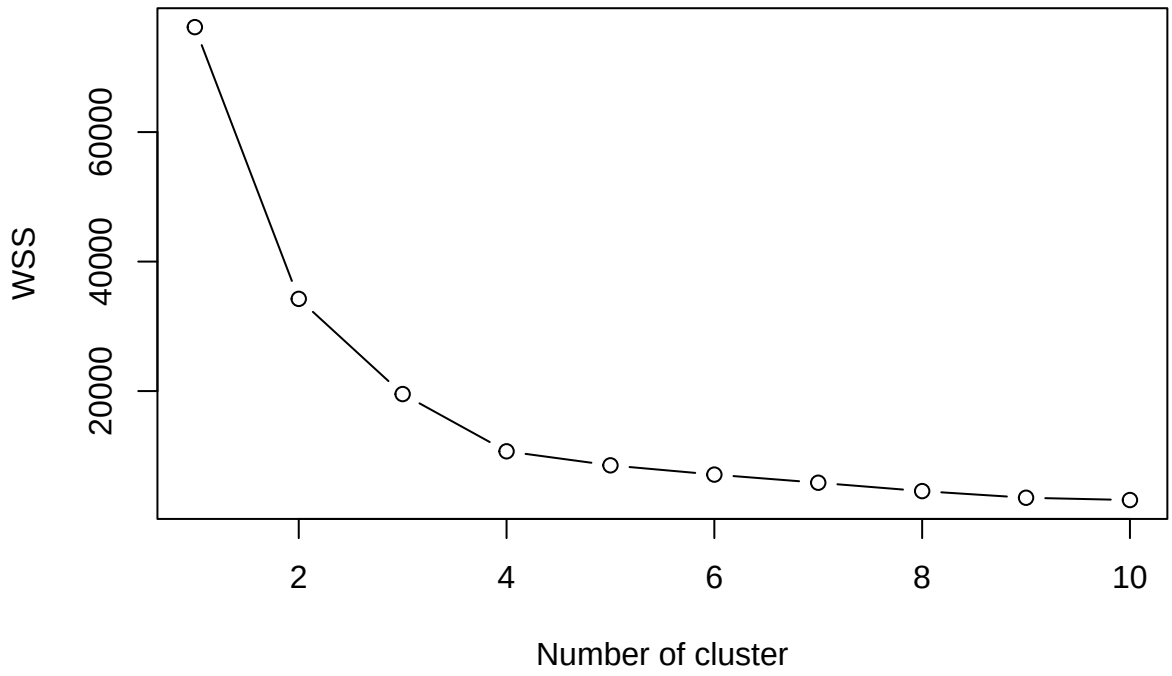
```
wss2 = (nrow(reduced_data) -1)*sum(apply(reduced_data, 2, var))

for (i in 1:10) {
  wss2[i] = sum(kmeans(reduced_data,
                       centers = i,
                       nstart = 20)$withinss)
  cat("Number of clusters (k):", i, "\tWSS:", wss2[i], "\n")
}
```

```
## Number of clusters (k): 1    WSS: 76211.11
## Number of clusters (k): 2    WSS: 34241.06
## Number of clusters (k): 3    WSS: 19538.75
## Number of clusters (k): 4    WSS: 10697.5
## Number of clusters (k): 5    WSS: 8530.665
## Number of clusters (k): 6    WSS: 7098.129
## Number of clusters (k): 7    WSS: 5844.288
## Number of clusters (k): 8    WSS: 4552.105
## Number of clusters (k): 9    WSS: 3525.282
## Number of clusters (k): 10   WSS: 3171.885
```

Plot wss

```
plot(1:10, wss2, type='b', xlab="Number of cluster",
     ylab="WSS")
```



-> select k=4

Buil model su dung kmeans

```
set.seed(20)
k.means.fit <- kmeans(reduced_data, centers = 4) # k = 4
```

Cluster

```
cluster_assignments <- k.means.fit$cluster
head(cluster_assignments)
```

```
## [1] 2 1 3 2 1 3
```

Cluster size

```r
cluster_size <- k.means.fit$size
cluster_size
```

```
## [1] 3529 1125 3259 1221
```

Compute cluster statistics

```r
km_stats <- cluster.stats(dist(reduced_data), cluster_assignments)
km_stats
```

```
## $n
## [1] 9134
##
## $cluster.number
## [1] 4
##
## $cluster.size
## [1] 3529 1125 3259 1221
##
## $min.cluster.size
## [1] 1125
##
## $noisen
## [1] 0
##
## $diameter
## [1] 3.232247 5.836666 3.834554 5.580128
##
## $average.distance
## [1] 0.9244987 1.7798028 1.2042583 1.5844477
##
## $median.distance
## [1] 0.9670176 1.4051993 1.0940504 1.0262449
##
## $separation
## [1] 0.06202023 0.18218947 0.06202023 0.18218947
##
## $average.toother
## [1] 4.540266 4.784090 4.582812 4.734827
##
## $separation.matrix
##            [,1]      [,2]       [,3]      [,4]
## [1,] 0.00000000 1.1825070 0.06202023 1.0919865
## [2,] 1.18250702 0.0000000 1.10546696 0.1821895
## [3,] 0.06202023 1.1054670 0.00000000 1.2309758
## [4,] 1.09198651 0.1821895 1.23097577 0.0000000
##
## $ave.between.matrix
##          [,1]     [,2]     [,3]     [,4]
## [1,] 0.000000 5.757412 4.417741 3.745849
## [2,] 5.757412 0.000000 3.760963 4.701800
```

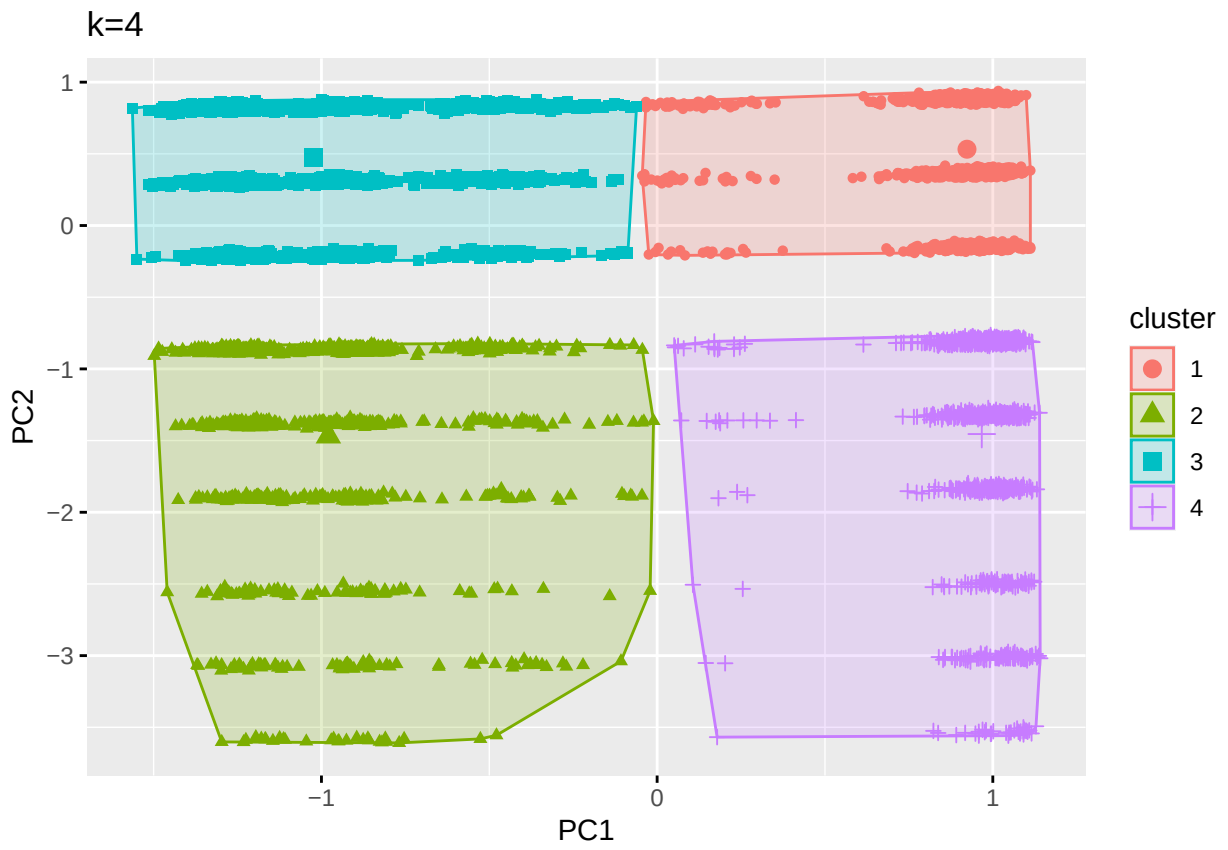
```
## [3,] 4.417741 3.760963 0.000000 5.817139
## [4,] 3.745849 4.701800 5.817139 0.000000
##
## $average.between
## [1] 4.625186
##
## $average.within
## [1] 1.217881
##
## $n.between
## [1] 28799284
##
## $n.within
## [1] 12911127
##
## $max.diameter
## [1] 5.836666
##
## $min.separation
## [1] 0.06202023
##
## $within.cluster.ss
## [1] 10697.5
##
## $clus.avg.silwidths
##         1         2         3         4
## 0.7221208 0.4848006 0.6412090 0.5470960
##
## $avg.silwidth
## [1] 0.6406251
##
## $g2
## NULL
##
## $g3
## NULL
##
## $pearsongamma
## [1] 0.7948632
##
## $dunn
## [1] 0.01062597
##
## $dunn2
## [1] 2.104643
##
## $entropy
## [1] 1.262072
##
## $wb.ratio
## [1] 0.263315
##
## $ch
## [1] 18637.97
```

```
##
## $cwidegap
## [1] 0.8239565 1.0679435 0.8076171 1.1398885
##
## $widestgap
## [1] 1.139889
##
## $sindex
## [1] 0.8893256
##
## $corrected.rand
## NULL
##
## $vi
## NULL
```

- Within-Cluster Sum of Squares (WCSS): 10697.5
- Silhouette Score: 0.6406251

Plot

```
fviz_cluster(k.means.fit, geo= "point",
             data = reduced_data) + ggtitle("k=4")
```

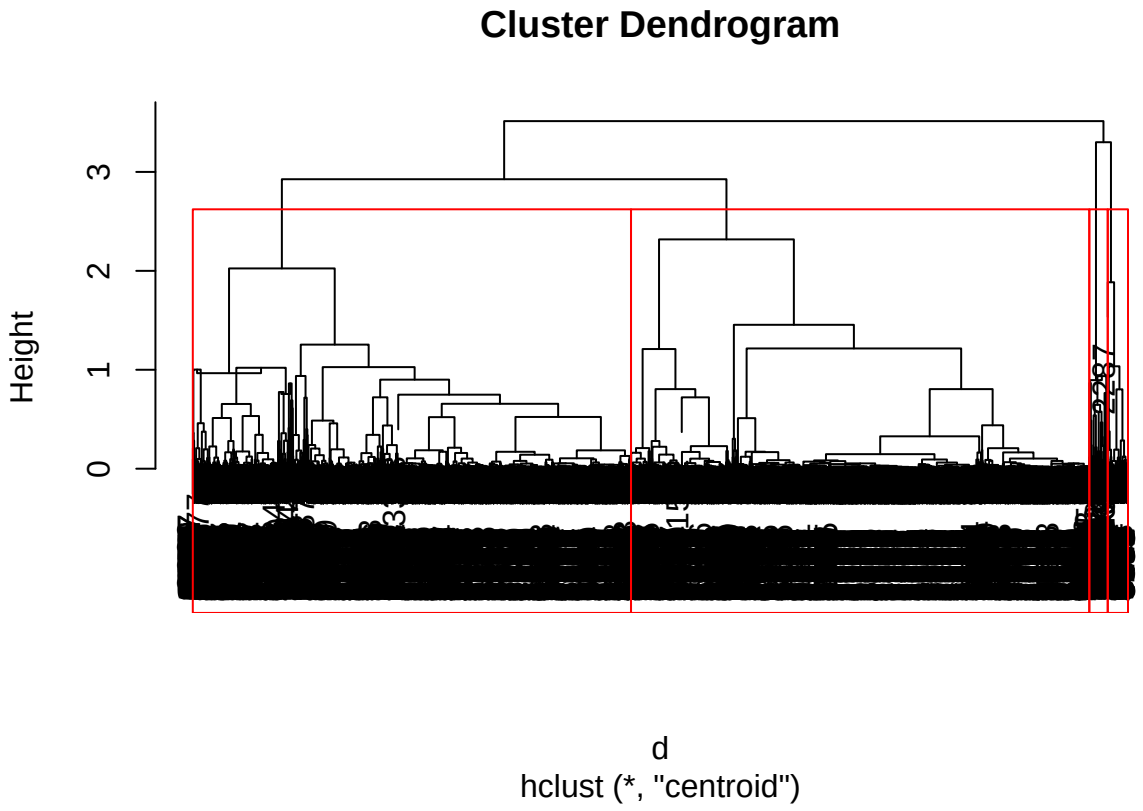


3. PCA & HIERARCHICAL

Build model

```
d <- dist(reduced_data, method = "euclidean")
H.fit <- hclust(d, method="centroid")
plot(H.fit) # display dendogram
groups <- cutree(H.fit, k=4)

# draw dendogram with red borders around the 4 clusters
rect.hclust(H.fit, k=4, border="red")
```



Compute cluster statistics

```
km_stats2 <- cluster.stats(dist(reduced_data), groups)
km_stats2
```

```
## $n
## [1] 9134
##
## $cluster.number
## [1] 4
```

```
##
## $cluster.size
## [1] 4280 4476  198  180
##
## $min.cluster.size
## [1] 180
##
## $noisen
## [1] 0
##
## $diameter
## [1] 6.460961 5.671284 2.903425 3.958864
##
## $average.distance
## [1] 1.9676881 1.6864982 0.8390447 1.2661665
##
## $median.distance
## [1] 1.8305758 1.0745543 0.9618243 1.1471376
##
## $separation
## [1] 0.06202023 0.06202023 0.96362898 0.96362898
##
## $average.toother
## [1] 4.892643 4.890309 6.457954 6.446843
##
## $separation.matrix
##            [,1]       [,2]     [,3]     [,4]
## [1,] 0.00000000 0.06202023 1.707449 1.067944
## [2,] 0.06202023 0.00000000 1.121918 1.670830
## [3,] 1.70744875 1.12191831 0.000000 0.963629
## [4,] 1.06794350 1.67082989 0.963629 0.000000
##
## $ave.between.matrix
##          [,1]     [,2]     [,3]     [,4]
## [1,] 0.000000 4.752696 7.266779 5.761122
## [2,] 4.752696 0.000000 5.769424 7.195410
## [3,] 7.266779 5.769424 0.000000 4.347357
## [4,] 5.761122 7.195410 4.347357 0.000000
##
## $average.between
## [1] 5.008759
##
## $average.within
## [1] 1.791604
##
## $n.between
## [1] 22502688
##
## $n.within
## [1] 19207723
##
## $max.diameter
## [1] 6.460961
##
```

```
## $min.separation
## [1] 0.06202023
##
## $within.cluster.ss
## [1] 22663.27
##
## $clus.avg.silwidths
##         1         2         3         4
## 0.5054180 0.5741699 0.8060398 0.6861620
##
## $avg.silwidth
## [1] 0.5491875
##
## $g2
## NULL
##
## $g3
## NULL
##
## $pearsongamma
## [1] 0.7796714
##
## $dunn
## [1] 0.009599228
##
## $dunn2
## [1] 2.209373
##
## $entropy
## [1] 0.8651775
##
## $wb.ratio
## [1] 0.3576942
##
## $ch
## [1] 7190.663
##
## $cwidegap
## [1] 1.1054670 1.0919865 1.4217827 0.8684766
##
## $widestgap
## [1] 1.421783
##
## $sindex
## [1] 0.9817452
##
## $corrected.rand
## NULL
##
## $vi
## NULL
```

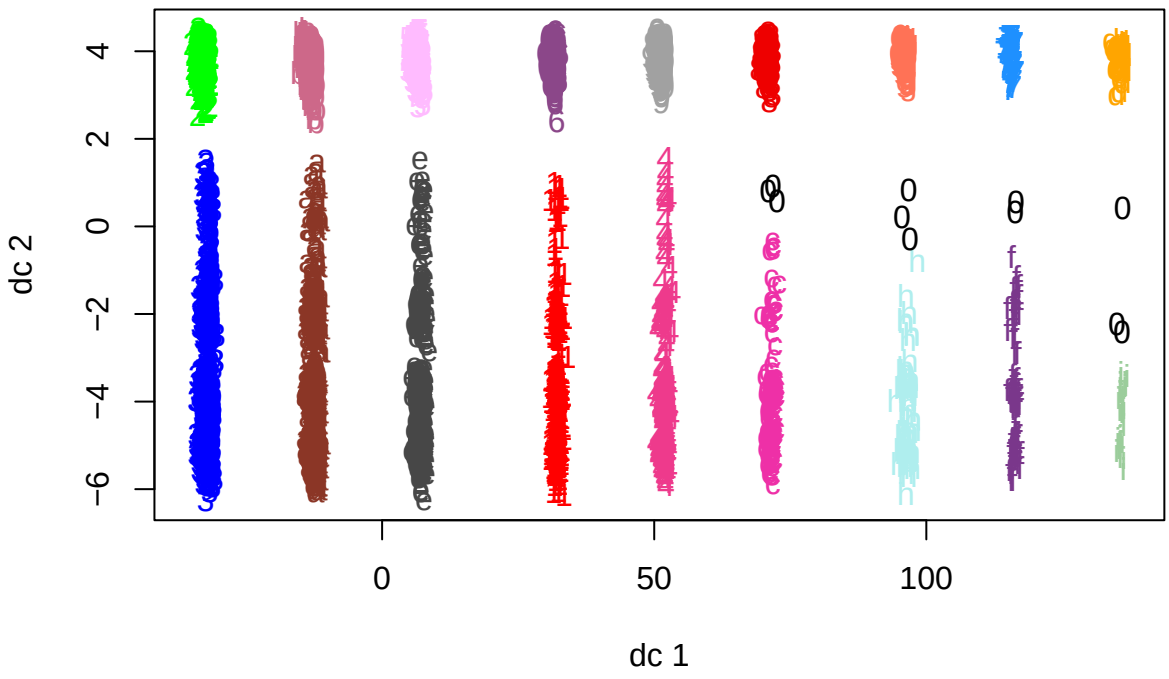
- Within-Cluster Sum of Squares (WCSS): 22663.27
- Silhouette Score: 0.5491875

4. PCA & DBSCAN

Compute DBSCAN clustering

```
dbscanOutput <- dbscan(reduced_data, eps = 0.5, MinPts = 4)

plotcluster(reduced_data, dbscanOutput$cluster)
```



Compute cluster statistics

```
# Convert DBSCAN clustering result to a factor (or character) vector
dbscan_groups <- as.numeric(dbscanOutput$cluster)

# Compute cluster statistics using silhouette analysis
km_stats3 <- cluster.stats(dist(reduced_data), dbscan_groups)
```

```
## Warning in cluster.stats(dist(reduced_data), dbscan_groups): clustering
## renumbered because maximum != number of clusters
```

```
km_stats3
```

```
## $n
```

```
## [1] 9134
##
## $cluster.number
## [1] 19
##
## $cluster.size
##  [1]   11  518 1742 1684  293  640  496   89  193  302 1071 1051  163   36  600
## [16]   73   72   73   27
##
## $min.cluster.size
## [1] 11
##
## $noisen
## [1] 0
##
## $diameter
##  [1] 3.6327285 3.8797206 1.0663997 4.2235206 4.0729273 0.9448388 1.1097313
##  [8] 0.6655829 0.8485908 0.8996427 4.1122224 1.1696807 3.0448889 0.6760789
## [15] 4.2409950 2.7824910 0.6695508 2.9049055 1.1980592
##
## $average.distance
##  [1] 1.7358521 0.7905011 0.1825330 0.8155939 0.8428800 0.1715980 0.1793347
##  [8] 0.1944357 0.1733012 0.1889430 0.7707866 0.1936637 0.7499257 0.1981019
## [15] 0.8464209 0.8428943 0.1794547 0.6641884 0.4262730
##
## $median.distance
##  [1] 1.6654233 0.6400105 0.1476807 0.6446970 0.6277721 0.1408765 0.1417026
##  [8] 0.1601176 0.1409518 0.1561879 0.6213090 0.1549261 0.5951093 0.1679974
## [15] 0.6588583 0.7187752 0.1488905 0.5991694 0.3836002
##
## $separation
##  [1] 0.2740646 0.7782195 0.5833978 0.5833978 0.7024019 0.6810997 0.7782195
##  [8] 0.8395609 0.8331072 0.7024019 0.5134423 0.5134423 0.5031280 0.8587491
## [15] 0.6810997 0.5572189 0.8395609 0.2740646 0.5004112
##
## $average.toother
##  [1] 5.737141 3.868740 3.963803 4.061594 4.314327 3.334757 3.736673 6.531148
##  [9] 4.798965 4.190580 3.570079 3.387299 4.903436 7.357506 3.472702 6.574767
## [17] 5.707968 5.823454 7.448882
##
## $separation.matrix
##            [,1]      [,2]      [,3]      [,4]      [,5]      [,6]      [,7]
##  [1,] 0.0000000 1.8600058 5.1337158 5.0037591 0.9016094 3.2893941 2.0917384
##  [2,] 1.8600058 0.0000000 3.2414504 3.0093438 0.8280823 1.5334331 0.7782195
##  [3,] 5.1337158 3.2414504 0.0000000 0.5833978 4.1485727 1.7742974 3.0031184
##  [4,] 5.0037591 3.0093438 0.5833978 0.0000000 3.9764038 2.0350709 3.2024041
##  [5,] 0.9016094 0.8280823 4.1485727 3.9764038 0.0000000 2.2823605 1.0800171
##  [6,] 3.2893941 1.5334331 1.7742974 2.0350709 2.2823605 0.0000000 1.0919865
##  [7,] 2.0917384 0.7782195 3.0031184 3.2024041 1.0800171 1.0919865 0.0000000
##  [8,] 1.4047847 4.2343028 7.1321612 7.2932176 3.2266930 5.2187461 3.9853569
##  [9,] 1.0533482 2.1734079 4.9312076 5.1149602 1.1810626 3.0132650 1.7908427
## [10,] 1.4453044 1.3984311 3.9529111 4.1737136 0.7024019 2.0397224 0.8065580
## [11,] 4.0466663 2.0532454 1.1899256 0.8026288 3.0057374 1.1575562 2.2265090
## [12,] 4.1589751 2.2722980 0.8021872 1.1082034 3.1757089 0.8239565 2.0533029
```

```
## [13,] 0.5031280 1.7677656 5.3294028 4.9097871 0.8390351 3.5721202 2.4463949
## [14,] 1.4217827 5.1456808 8.1090576 8.2335068 4.1479695 6.1979387 4.9633371
## [15,] 3.0947733 1.1054670 2.0081093 1.7714055 2.0625290 0.6810997 1.3331747
## [16,] 0.5572189 3.9813181 7.4498940 7.1260034 3.0195084 5.6227993 4.4224369
## [17,] 1.2471097 3.3572899 6.1852410 6.3660285 2.3442880 4.2709370 3.0385308
## [18,] 0.2740646 2.9933505 6.6463129 6.1402762 2.0331375 4.8521569 3.6766415
## [19,] 0.5004112 4.9816256 8.8352909 8.1345033 4.0324535 7.1047665 5.9562471
##           [,8]      [,9]     [,10]     [,11]     [,12]     [,13]     [,14]
##  [1,] 1.4047847 1.0533482 1.4453044 4.0466663 4.1589751 0.5031280 1.4217827
##  [2,] 4.2343028 2.1734079 1.3984311 2.0532454 2.2722980 1.7677656 5.1456808
##  [3,] 7.1321612 4.9312076 3.9529111 1.1899256 0.8021872 5.3294028 8.1090576
##  [4,] 7.2932176 5.1149602 4.1737136 0.8026288 1.1082034 4.9097871 8.2335068
##  [5,] 3.2266930 1.1810626 0.7024019 3.0057374 3.1757089 0.8390351 4.1479695
##  [6,] 5.2187461 3.0132650 2.0397224 1.1575562 0.8239565 3.5721202 6.1979387
##  [7,] 3.9853569 1.7908427 0.8065580 2.2265090 2.0533029 2.4463949 4.9633371
##  [8,] 0.0000000 2.0668560 3.0330420 6.3140816 6.1821238 2.8679969 0.8587491
##  [9,] 2.0668560 0.0000000 0.8331072 4.1408046 3.9787184 1.7447886 3.0447221
## [10,] 3.0330420 0.8331072 0.0000000 3.1999498 3.0026581 1.9971512 4.0096045
## [11,] 6.3140816 4.1408046 3.1999498 0.0000000 0.5134423 3.9674624 7.2578680
## [12,] 6.1821238 3.9787184 3.0026581 0.5134423 0.0000000 4.3617876 7.1592462
## [13,] 2.8679969 1.7447886 1.9971512 3.9674624 4.3617876 0.0000000 3.6136556
## [14,] 0.8587491 3.0447221 4.0096045 7.2578680 7.1592462 3.6136556 0.0000000
## [15,] 5.3942505 3.2375903 2.3011100 0.8076171 1.0509281 3.0030394 6.3349726
## [16,] 2.0869436 2.9066283 3.6852863 6.1853555 6.4761296 2.0720739 2.2426389
## [17,] 0.8395609 1.1219183 2.0870324 5.3871267 5.2350118 2.2505963 1.8137206
## [18,] 2.3017837 2.3861256 3.0284982 5.1925649 5.6754002 1.0679435 2.7564375
## [19,] 3.6673350 4.6185224 5.3232849 7.1582793 7.8768710 3.0667859 3.4755144
##           [,15]     [,16]     [,17]     [,18]     [,19]
##  [1,] 3.0947733 0.5572189 1.2471097 0.2740646 0.5004112
##  [2,] 1.1054670 3.9813181 3.3572899 2.9933505 4.9816256
##  [3,] 2.0081093 7.4498940 6.1852410 6.6463129 8.8352909
##  [4,] 1.7714055 7.1260034 6.3660285 6.1402762 8.1345033
##  [5,] 2.0625290 3.0195084 2.3442880 2.0331375 4.0324535
##  [6,] 0.6810997 5.6227993 4.2709370 4.8521569 7.1047665
##  [7,] 1.3331747 4.4224369 3.0385308 3.6766415 5.9562471
##  [8,] 5.3942505 2.0869436 0.8395609 2.3017837 3.6673350
##  [9,] 3.2375903 2.9066283 1.1219183 2.3861256 4.6185224
## [10,] 2.3011100 3.6852863 2.0870324 3.0284982 5.3232849
## [11,] 0.8076171 6.1853555 5.3871267 5.1925649 7.1582793
## [12,] 1.0509281 6.4761296 5.2350118 5.6754002 7.8768710
## [13,] 3.0030394 2.0720739 2.2505963 1.0679435 3.0667859
## [14,] 6.3349726 2.2426389 1.8137206 2.7564375 3.4755144
## [15,] 0.0000000 5.2207022 4.4723651 4.2234519 6.2159555
## [16,] 5.2207022 0.0000000 2.2625735 0.8515705 0.8684766
## [17,] 4.4723651 2.2625735 0.0000000 2.1235486 3.9928280
## [18,] 4.2234519 0.8515705 2.1235486 0.0000000 1.8429257
## [19,] 6.2159555 0.8684766 3.9928280 1.8429257 0.0000000
##
## $ave.between.matrix
##           [,1]     [,2]      [,3]     [,4]     [,5]      [,6]      [,7]
##  [1,] 0.000000 4.324866 7.0429497 7.123242 3.605267 5.2421881 4.1671732
##  [2,] 4.324866 0.000000 5.3376771 3.325686 1.362781 4.4560045 4.2866938
##  [3,] 7.042950 5.337677 0.0000000 4.259247 5.966318 1.9413991 3.1677132
##  [4,] 7.123242 3.325686 4.2592470 0.000000 4.266414 4.6921818 5.3386284
```

```
##  [5,] 3.605267 1.362781 5.9663177 4.266414 0.000000 4.8200704 4.3930227
##  [6,] 5.242188 4.456004 1.9413991 4.692182 4.820070 0.0000000 1.2493174
##  [7,] 4.167173 4.286694 3.1677132 5.338628 4.393023 1.2493174 0.0000000
##  [8,] 2.516063 5.959138 7.2821788 8.459060 5.330141 5.3539717 4.1253220
##  [9,] 2.792995 4.685100 5.0874868 6.650705 4.361812 3.1606382 1.9356206
## [10,] 3.398712 4.380971 4.1280058 5.952804 4.261769 2.2035042 0.9880646
## [11,] 6.237989 2.407100 4.4079953 1.342664 3.330331 4.4093908 4.8514704
## [12,] 6.130512 4.816888 0.9922248 4.368168 5.334123 0.9895985 2.2069250
## [13,] 3.014176 2.157884 6.6831512 5.191648 1.347678 5.3525352 4.7313100
## [14,] 2.888140 6.671797 8.2525688 9.308835 5.963998 6.3241181 5.0949921
## [15,] 5.361362 1.575433 4.6831017 2.191097 2.443632 4.2513833 4.4447833
## [16,] 2.489814 4.250999 8.4290792 7.357643 3.334993 6.8282500 5.9158647
## [17,] 2.441430 5.335326 6.3212609 7.647335 4.815081 4.3933596 3.1654430
## [18,] 2.661647 3.293503 7.7112177 6.390490 2.395260 6.2264340 5.4341470
## [19,] 2.950062 5.155370 9.3900403 8.289832 4.217562 7.7515890 6.7934745
##           [,8]      [,9]     [,10]    [,11]     [,12]    [,13]     [,14]
##  [1,] 2.5160634 2.7929950 3.3987121 6.237989 6.1305124 3.014176 2.8881395
##  [2,] 5.9591382 4.6850999 4.3809709 2.407100 4.8168883 2.157884 6.6717969
##  [3,] 7.2821788 5.0874868 4.1280058 4.407995 0.9922248 6.683151 8.2525688
##  [4,] 8.4590599 6.6507049 5.9528043 1.342664 4.3681682 5.191648 9.3088351
##  [5,] 5.3301405 4.3618121 4.2617687 3.330331 5.3341229 1.347678 5.9639985
##  [6,] 5.3539717 3.1606382 2.2035042 4.409391 0.9895985 5.352535 6.3241181
##  [7,] 4.1253220 1.9356206 0.9880646 4.851470 2.2069250 4.731310 5.0949921
##  [8,] 0.0000000 2.2071527 3.1653367 7.660632 6.3178999 4.826631 0.9991272
##  [9,] 2.2071527 0.0000000 0.9856117 5.964499 4.1239004 4.277170 3.1747244
## [10,] 3.1653367 0.9856117 0.0000000 5.349952 3.1653665 4.401604 4.1342899
## [11,] 7.6606318 5.9644988 5.3499524 0.000000 4.2925449 4.241637 8.4804452
## [12,] 6.3178999 4.1239004 3.1653665 4.292545 0.0000000 5.974096 7.2881381
## [13,] 4.8266311 4.2771705 4.4016037 4.241637 5.9740965 0.000000 5.3447393
## [14,] 0.9991272 3.1747244 4.1342899 8.480445 7.2881381 5.344739 0.0000000
## [15,] 6.8611905 5.3050373 4.7967597 1.352951 4.3592363 3.323698 7.6454453
## [16,] 4.1899716 4.7365166 5.2702662 6.400309 7.6070652 2.398861 4.3043645
## [17,] 0.9867580 1.2528702 2.2069062 6.888961 5.3571728 4.468300 1.9454313
## [18,] 4.4944343 4.5514411 4.9167232 5.432395 6.9385543 1.488589 4.8019892
## [19,] 4.5808654 5.4762997 6.0928355 7.326291 8.5529210 3.249353 4.4772396
##          [,15]    [,16]    [,17]    [,18]    [,19]
##  [1,] 5.361362 2.489814 2.441430 2.661647 2.950062
##  [2,] 1.575433 4.250999 5.335326 3.293503 5.155370
##  [3,] 4.683102 8.429079 6.321261 7.711218 9.390040
##  [4,] 2.191097 7.357643 7.647335 6.390490 8.289832
##  [5,] 2.443632 3.334993 4.815081 2.395260 4.217562
##  [6,] 4.251383 6.828250 4.393360 6.226434 7.751589
##  [7,] 4.444783 5.915865 3.165443 5.434147 6.793474
##  [8,] 6.861190 4.189972 0.986758 4.494434 4.580865
##  [9,] 5.305037 4.736517 1.252870 4.551441 5.476300
## [10,] 4.796760 5.270266 2.206906 4.916723 6.092836
## [11,] 1.352951 6.400309 6.888961 5.432395 7.326291
## [12,] 4.359236 7.607065 5.357173 6.938554 8.552921
## [13,] 3.323698 2.398861 4.468300 1.488589 3.249353
## [14,] 7.645445 4.304365 1.945431 4.801989 4.477240
## [15,] 0.000000 5.459429 6.138430 4.497260 6.378801
## [16,] 5.459429 0.000000 4.306340 1.306977 1.251941
## [17,] 6.138430 4.306340 0.000000 4.391362 4.879387
## [18,] 4.497260 1.306977 4.391362 0.000000 2.042898
```

```
## [19,] 6.378801 1.251941 4.879387 2.042898 0.000000
##
## $average.between
## [1] 3.93986
##
## $average.within
## [1] 0.4898487
##
## $n.between
## [1] 36878587
##
## $n.within
## [1] 4831824
##
## $max.diameter
## [1] 4.240995
##
## $min.separation
## [1] 0.2740646
##
## $within.cluster.ss
## [1] 2734.483
##
## $clus.avg.silwidths
##           1           2           3           4           5           6
## -0.006356621  0.431551912  0.817073907  0.404445638  0.381935353  0.827586672
##           7           8           9          10          11          12
##  0.819603681  0.801385164  0.825346034  0.805479502  0.434681132  0.802148964
##          13          14          15          16          17          18
##  0.455573103  0.802887620  0.385587724  0.314317227  0.818614524  0.495835791
##          19
##  0.661043396
##
## $avg.silwidth
## [1] 0.6162586
##
## $g2
## NULL
##
## $g3
## NULL
##
## $pearsongamma
## [1] 0.5417533
##
## $dunn
## [1] 0.06462272
##
## $dunn2
## [1] 0.567797
##
## $entropy
## [1] 2.397957
##
```

```
## $wb.ratio
## [1] 0.1243315
##
## $ch
## [1] 13606.87
##
## $cwidegap
##  [1] 1.44858419 0.21660155 0.08045847 0.11243927 0.18218947 0.08969957
##  [7] 0.21478880 0.05496253 0.06190555 0.07342215 0.16328964 0.11406510
## [13] 0.33641363 0.12213437 0.24294558 0.27838591 0.06828678 0.44529089
## [19] 0.16444935
##
## $widestgap
## [1] 1.448584
##
## $sindex
## [1] 0.8273376
##
## $corrected.rand
## NULL
##
## $vi
## NULL
```

- Within-Cluster Sum of Squares (WCSS): 2734.483
- Silhouette Score: 0.6162586

3. CLASSIFICATION _ Thi Tinh Lo (22236226)

2023-08-19

CLASSIFICATION:

1. LOGISTIC REGRESSION

2. DECISION TREE

3. RANDOM FOREST

4. NEUTRAL NETWORK

Read data

```
setwd('C:/Users/tinhl/OneDrive/Documents')
data <- read.csv(file ='data_processed.csv')
```

```
summary(data)
```

```
##      State          Response         Coverage       Education
##  Min.   :1.000   Min.   :0.0000   Min.   :1.000   Min.   :1.000
##  1st Qu.:2.000   1st Qu.:0.0000   1st Qu.:2.000   1st Qu.:3.000
##  Median :2.000   Median :0.0000   Median :3.000   Median :4.000
##  Mean   :2.742   Mean   :0.1432   Mean   :2.519   Mean   :3.712
##  3rd Qu.:4.000   3rd Qu.:0.0000   3rd Qu.:3.000   3rd Qu.:5.000
##  Max.   :5.000   Max.   :1.0000   Max.   :3.000   Max.   :5.000
##  EmploymentStatus     Gender     Location_Code   Marital_Status  Policy_Type
##  Min.   :1.000    Min.   :1.00   Min.   :1.000   Min.   :1.00   Min.   :1.000
##  1st Qu.:2.000    1st Qu.:1.00   1st Qu.:2.000   1st Qu.:2.00   1st Qu.:2.000
##  Median :2.000    Median :1.00   Median :2.000   Median :2.00   Median :3.000
##  Mean   :2.826    Mean   :1.49   Mean   :2.021   Mean   :2.12   Mean   :2.702
##  3rd Qu.:5.000    3rd Qu.:2.00   3rd Qu.:2.000   3rd Qu.:3.00   3rd Qu.:3.000
##  Max.   :5.000    Max.   :2.00   Max.   :3.000   Max.   :3.00   Max.   :3.000
##      Policy      Renew_Offer_Type Sales_Channel   Vehicle_Class
##  Min.   :1.000   Min.   :1.00     Min.   :1.000   Min.   :1.000
##  1st Qu.:6.000   1st Qu.:1.00     1st Qu.:1.000   1st Qu.:1.000
##  Median :8.000   Median :2.00     Median :2.000   Median :1.000
##  Mean   :7.425   Mean   :1.97     Mean   :2.103   Mean   :3.036
##  3rd Qu.:9.000   3rd Qu.:3.00     3rd Qu.:3.000   3rd Qu.:5.000
##  Max.   :9.000   Max.   :4.00     Max.   :4.000   Max.   :6.000
##   Vehicle_Size  Customer_Lifetime_Value     Income        Monthly_Premium_Auto
##  Min.   :1.00   Min.   :-0.8888         Min.   :-1.2395   Min.   :-0.9364
##  1st Qu.:2.00   1st Qu.:-0.5837         1st Qu.:-1.2395   1st Qu.:-0.7329
```

```
##  Median :2.00   Median :-0.3238         Median :-0.1240   Median :-0.2970
##  Mean   :1.91   Mean   : 0.0000         Mean   : 0.0000   Mean   : 0.0000
##  3rd Qu.:2.00   3rd Qu.: 0.1393         3rd Qu.: 0.8118   3rd Qu.: 0.4586
##  Max.   :3.00   Max.   :10.9621         Max.   : 2.0515   Max.   : 5.9515
##  Months_Since_Last_Claim Months_Since_Policy_Inception
##  Min.   :-1.4987         Min.   :-1.722376
##  1st Qu.:-0.9031         1st Qu.:-0.862345
##  Median :-0.1089         Median :-0.002315
##  Mean   : 0.0000         Mean   : 0.000000
##  3rd Qu.: 0.7846         3rd Qu.: 0.821881
##  Max.   : 1.9758         Max.   : 1.825250
##  Number_of_Open_Complaints Number_of_Policies Total_Claim_Amount
##  Min.   :-0.4222           Min.   :-0.8226    Min.   :-1.4939
##  1st Qu.:-0.4222           1st Qu.:-0.8226    1st Qu.:-0.5571
##  Median :-0.4222           Median :-0.4042    Median :-0.1726
##  Mean   : 0.0000           Mean   : 0.0000    Mean   : 0.0000
##  3rd Qu.:-0.4222           3rd Qu.: 0.4325    3rd Qu.: 0.3905
##  Max.   : 5.0700           Max.   : 2.5244    Max.   : 8.4652
```

Create the training and test data

```
set.seed(42)
n= nrow(data)
trainIndex = sample(1:n, size= round(0.7*n), replace=FALSE)
train = data[trainIndex,]
test = data[-trainIndex,]
```

Rows of training data and test data

```
nrow(train)
```

```
## [1] 6394
```

```
nrow(test)
```

```
## [1] 2740
```

1. LOGISTIC REGRESSION

Build model

Estimates a logistic regression model using the glm

```
set.seed(42)
model <- glm(Response ~ ., data = train, family = "binomial")
summary(model)
```

```
##
## Call:
## glm(formula = Response ~ ., family = "binomial", data = train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.2581  -0.6210  -0.4769  -0.3207   2.4466
##
## Coefficients:
##                                Estimate Std. Error z value Pr(>|z|)
## (Intercept)                    0.536327   0.391540   1.370  0.17075
## State                         -0.014333   0.028537  -0.502  0.61549
## Coverage                       0.035270   0.063974   0.551  0.58141
## Education                     -0.095212   0.033858  -2.812  0.00492 **
## EmploymentStatus              -0.016092   0.040050  -0.402  0.68784
## Gender                         0.003229   0.073719   0.044  0.96506
## Location_Code                  0.041637   0.064683   0.644  0.51976
## Marital_Status                -0.452459   0.058533  -7.730 1.08e-14 ***
## Policy_Type                   -0.124376   0.157612  -0.789  0.43004
## Policy                         0.006627   0.047463   0.140  0.88895
## Renew_Offer_Type              -0.575076   0.045859 -12.540  < 2e-16 ***
## Sales_Channel                 -0.195313   0.036487  -5.353 8.65e-08 ***
## Vehicle_Class                  0.013449   0.017328   0.776  0.43766
## Vehicle_Size                   0.279868   0.067903   4.122 3.76e-05 ***
## Customer_Lifetime_Value       -0.068328   0.041290  -1.655  0.09796 .
## Income                         0.116835   0.054228   2.155  0.03120 *
## Monthly_Premium_Auto          -0.152651   0.060331  -2.530  0.01140 *
## Months_Since_Last_Claim       -0.068249   0.037157  -1.837  0.06624 .
## Months_Since_Policy_Inception -0.015504   0.036538  -0.424  0.67133
## Number_of_Open_Complaints     -0.055188   0.037563  -1.469  0.14177
## Number_of_Policies            -0.079527   0.037483  -2.122  0.03386 *
## Total_Claim_Amount             0.255875   0.057674   4.437 9.14e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 5310.7  on 6393  degrees of freedom
## Residual deviance: 4963.2  on 6372  degrees of freedom
## AIC: 5007.2
##
## Number of Fisher Scoring iterations: 5
```

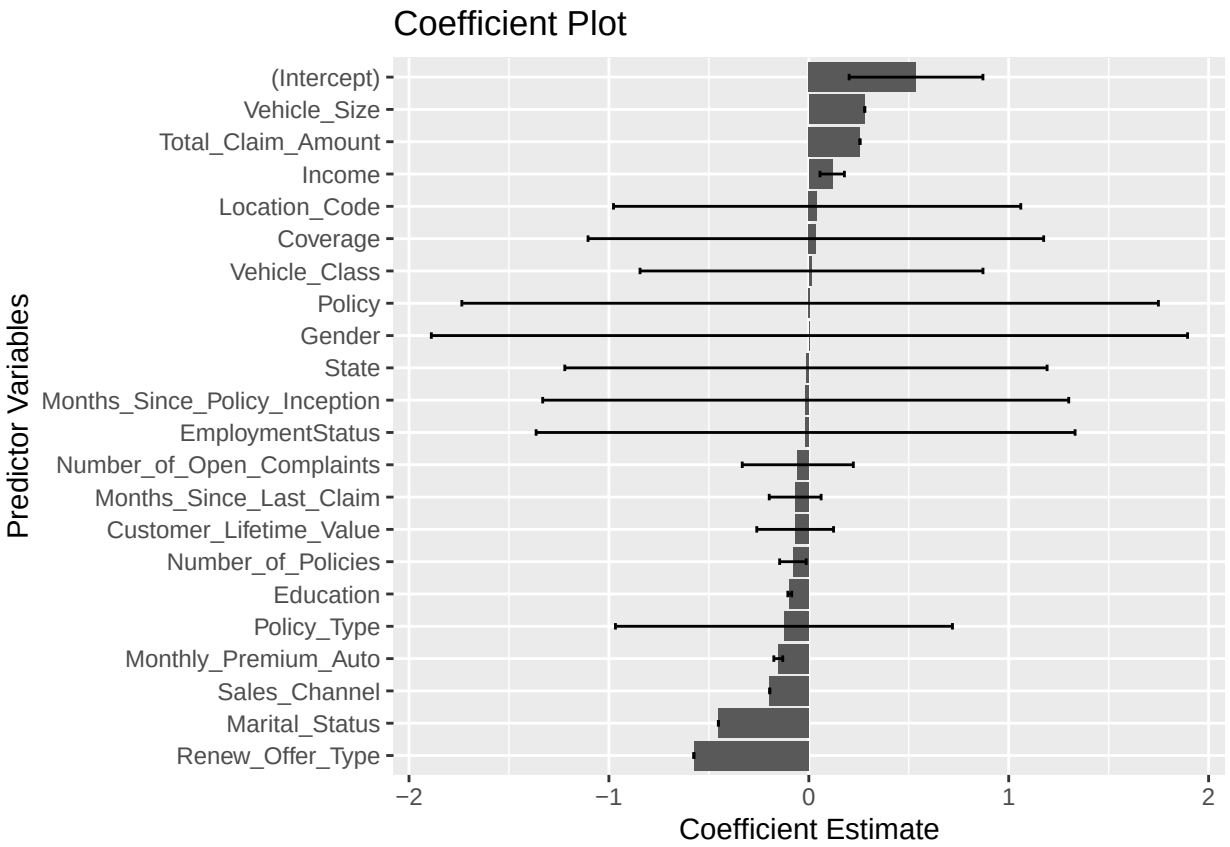
Coefficient

```
coef_df <- as.data.frame(summary(model)$coefficients)
coef_df$Variable <- rownames(coef_df)
coef_df
```

```
##                                  Estimate Std. Error      z value     Pr(>|z|)
## (Intercept)                   0.536326749 0.39154018   1.36978725 1.707533e-01
## State                        -0.014332795 0.02853706  -0.50225189 6.154903e-01
## Coverage                      0.035270029 0.06397387   0.55131926 5.814148e-01
```

```
## Education                     -0.095211822 0.03385833  -2.81206458 4.922462e-03
## EmploymentStatus              -0.016091667 0.04004989  -0.40179057 6.878382e-01
## Gender                         0.003229210 0.07371925   0.04380415 9.650605e-01
## Location_Code                  0.041637265 0.06468329   0.64370979 5.197636e-01
## Marital_Status                -0.452459425 0.05853304  -7.72998321 1.075608e-14
## Policy_Type                   -0.124376480 0.15761237  -0.78912894 4.300367e-01
## Policy                         0.006627466 0.04746276   0.13963508 8.889483e-01
## Renew_Offer_Type              -0.575076293 0.04585896 -12.54010630 4.503910e-36
## Sales_Channel                 -0.195313445 0.03648695  -5.35296773 8.652328e-08
## Vehicle_Class                  0.013449376 0.01732817   0.77615685 4.376564e-01
## Vehicle_Size                   0.279868113 0.06790348   4.12155763 3.763193e-05
## Customer_Lifetime_Value       -0.068328444 0.04129007  -1.65483971 9.795702e-02
## Income                         0.116835378 0.05422773   2.15453194 3.119848e-02
## Monthly_Premium_Auto          -0.152650512 0.06033083  -2.53022408 1.139897e-02
## Months_Since_Last_Claim       -0.068248866 0.03715682  -1.83677907 6.624252e-02
## Months_Since_Policy_Inception -0.015504278 0.03653847  -0.42432745 6.713270e-01
## Number_of_Open_Complaints     -0.055187914 0.03756252  -1.46922815 1.417709e-01
## Number_of_Policies            -0.079527422 0.03748276  -2.12170687 3.386236e-02
## Total_Claim_Amount             0.255874573 0.05767430   4.43654408 9.141462e-06
##                                                    Variable
## (Intercept)                                     (Intercept)
## State                                                 State
## Coverage                                           Coverage
## Education                                         Education
## EmploymentStatus                            EmploymentStatus
## Gender                                                Gender
## Location_Code                                  Location_Code
## Marital_Status                                Marital_Status
## Policy_Type                                      Policy_Type
## Policy                                                Policy
## Renew_Offer_Type                            Renew_Offer_Type
## Sales_Channel                                  Sales_Channel
## Vehicle_Class                                  Vehicle_Class
## Vehicle_Size                                    Vehicle_Size
## Customer_Lifetime_Value            Customer_Lifetime_Value
## Income                                                Income
## Monthly_Premium_Auto                  Monthly_Premium_Auto
## Months_Since_Last_Claim            Months_Since_Last_Claim
## Months_Since_Policy_Inception Months_Since_Policy_Inception
## Number_of_Open_Complaints         Number_of_Open_Complaints
## Number_of_Policies                       Number_of_Policies
## Total_Claim_Amount                       Total_Claim_Amount
```

```
ggplot(coef_df, aes(x = reorder(Variable, Estimate), y = Estimate)) +
  geom_bar(stat = "identity") +
  geom_errorbar(aes(ymin = Estimate - 1.96 * `Pr(>|z|)`, ymax = Estimate + 1.96 * `Pr(>|z|)`), width = 
  coord_flip() +
  labs(title = "Coefficient Plot", x = "Predictor Variables", y = "Coefficient Estimate")
```



TRAIN DATA

Predict train data

```
pred <- predict(model, newdata = train, type = "response")
pred_value <- ifelse(pred > 0.5, 1, 0)
result <- table(pred_value, train$Response)
result
```

```
##
## pred_value    0    1
##          0 5460  932
##          1    2    0
```

Calculate accuracy, precision, recall, F1-score

```
TN <- result[1, 1]
TP <- result[2, 2]
FP <- result[1, 2]
FN <- result[2, 1]

accuracy <- (TN + TP) / (TN + TP + FP + FN)
accuracy
```

```
## [1] 0.8539256
```

```r
precision <- TP / (TP + FP)
precision
```

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

```r
recall <- TP / (TP + FN)
recall
```

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

```r
f1_score <- 2 * (precision * recall) / (precision + recall)
f1_score
```

```
## [1] NaN
```

-> (TP) value is 0, it means that the model did not correctly predict any positive instances. Both precision and recall will also be 0, and the F1 score cannot be calculated.

TEST DATA

Predict test data

```r
pred2 <- predict(model, newdata = test, type = "response")
pred_value2 <- ifelse(pred2 > 0.5, 1, 0)
result <- table(pred_value2, test$Response)
result
```

```
##
## pred_value2    0    1
##           0 2363  376
##           1    1    0
```

Calculate accuracy, precision, recall, F1-score

```r
TN <- result[1, 1]
TP <- result[2, 2]
FP <- result[1, 2]
FN <- result[2, 1]

accuracy <- (TN + TP) / (TN + TP + FP + FN)
accuracy
```

```
## [1] 0.8624088
```

```r
precision <- TP / (TP + FP)
precision
```

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

```r
recall <- TP / (TP + FN)
recall
```

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

```r
f1_score <- 2 * (precision * recall) / (precision + recall)
f1_score
```

```
## [1] NaN
```

ROC Curve

```r
# Create ROC curves for train and test data
roc_data <- roc(train$Response, pred)
```

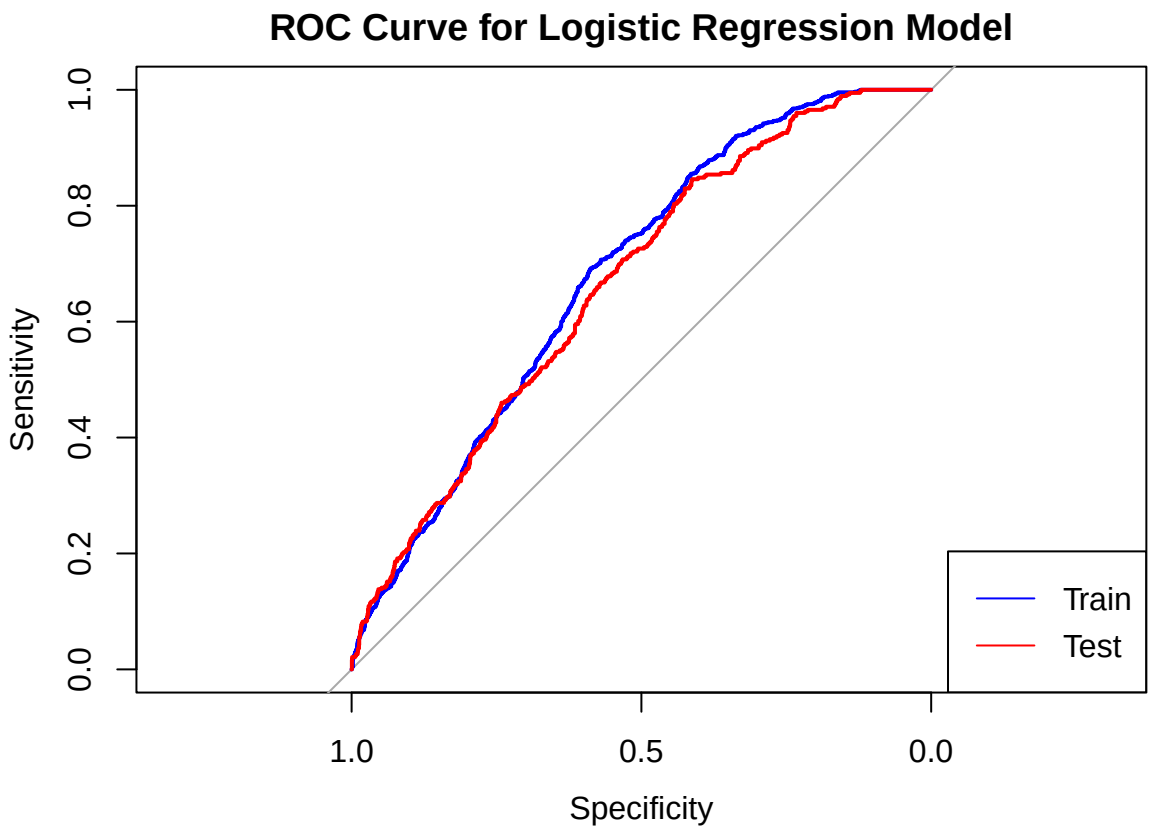
```
## Setting levels: control = 0, case = 1
```

```
## Setting direction: controls < cases
```

```r
roc_data2 <- roc(test$Response, pred2)
```

```
## Setting levels: control = 0, case = 1
## Setting direction: controls < cases
```

```r
# Combine ROC curves into a single plot
plot(roc_data, col = "blue", main = "ROC Curve for Logistic Regression Model")
lines(roc_data2, col = "red")
legend("bottomright", legend = c("Train", "Test"), col = c("blue", "red"), lty = 1)
```



2. DECISION TREE

Build model

```
set.seed(42)
data.tree = rpart(Response ~ .,data = train, method="class")
data.tree
```

```
## n= 6394
##
## node), split, n, loss, yval, (yprob)
##       * denotes terminal node
##
##   1) root 6394 932 0 (0.85423835 0.14576165)
##     2) Renew_Offer_Type>=2.5 1736  19 0 (0.98905530 0.01094470) *
##     3) Renew_Offer_Type< 2.5 4658 913 0 (0.80399313 0.19600687)
##       6) Marital_Status>=1.5 3908 681 0 (0.82574207 0.17425793)
##        12) Income< -0.900723 1117 107 0 (0.90420770 0.09579230) *
##        13) Income>=-0.900723 2791 574 0 (0.79433895 0.20566105)
##          26) EmploymentStatus< 3.5 2684 485 0 (0.81929955 0.18070045) *
##          27) EmploymentStatus>=3.5 107  18 1 (0.16822430 0.83177570) *
##       7) Marital_Status< 1.5 750 232 0 (0.69066667 0.30933333)
##        14) Income>=-0.3808399 438  89 0 (0.79680365 0.20319635) *
```

```
##        15) Income< -0.3808399 312 143 0 (0.54166667 0.45833333)
##          30) Months_Since_Last_Claim>=1.032735 39   2 0 (0.94871795 0.05128205) *
##          31) Months_Since_Last_Claim< 1.032735 273 132 1 (0.48351648 0.51648352)
##            62) Total_Claim_Amount< -0.4598129 22   0 0 (1.00000000 0.00000000) *
##            63) Total_Claim_Amount>=-0.4598129 251 110 1 (0.43824701 0.56175299)
##             126) Income< -0.8519737 96  37 0 (0.61458333 0.38541667) *
##             127) Income>=-0.8519737 155  51 1 (0.32903226 0.67096774)
##               254) EmploymentStatus< 3.5 98  47 1 (0.47959184 0.52040816)
##                 508) Education>=3.5 32   5 0 (0.84375000 0.15625000) *
##                 509) Education< 3.5 66  20 1 (0.30303030 0.69696970) *
##               255) EmploymentStatus>=3.5 57   4 1 (0.07017544 0.92982456) *
```

```
summary(data.tree)
```

```
## Call:
## rpart(formula = Response ~ ., data = train, method = "class")
##   n= 6394
##
##           CP nsplit rel error    xerror       xstd
## 1 0.01904506      0 1.0000000 1.0000000 0.03027482
## 2 0.01421674      4 0.9238197 0.9012876 0.02898280
## 3 0.01180258      8 0.8669528 0.8980687 0.02893881
## 4 0.01000000     10 0.8433476 0.8937768 0.02887997
##
## Variable importance
##          EmploymentStatus          Renew_Offer_Type                    Income
##                        37                        22                        15
##            Marital_Status       Total_Claim_Amount   Months_Since_Last_Claim
##                         6                         5                         4
##                 Education   Customer_Lifetime_Value             Location_Code
##                         4                         2                         2
## Number_of_Open_Complaints      Monthly_Premium_Auto
##                         1                         1
##
## Node number 1: 6394 observations,    complexity param=0.01904506
##   predicted class=0  expected loss=0.1457617  P(node) =1
##     class counts:  5462   932
##    probabilities: 0.854 0.146
##   left son=2 (1736 obs) right son=3 (4658 obs)
##   Primary splits:
##       Renew_Offer_Type   < 2.5        to the right, improve=86.62472, (0 missing)
##       Marital_Status     < 1.5        to the right, improve=20.03729, (0 missing)
##       Sales_Channel      < 1.5        to the right, improve=18.23999, (0 missing)
##       Total_Claim_Amount < -0.5269586 to the left,  improve=15.55030, (0 missing)
##       Income             < -0.900723  to the left,  improve=14.07473, (0 missing)
##   Surrogate splits:
##       Income < 2.047492   to the right, agree=0.729, adj=0.001, (0 split)
##
## Node number 2: 1736 observations
##   predicted class=0  expected loss=0.0109447  P(node) =0.2715045
##     class counts:  1717    19
##    probabilities: 0.989 0.011
##
## Node number 3: 4658 observations,    complexity param=0.01904506
```

```
##   predicted class=0  expected loss=0.1960069  P(node) =0.7284955
##     class counts:  3745   913
##    probabilities: 0.804 0.196
##   left son=6 (3908 obs) right son=7 (750 obs)
##   Primary splits:
##       Marital_Status     < 1.5        to the right, improve=22.96143, (0 missing)
##       Income             < -0.900723  to the left,  improve=21.71285, (0 missing)
##       EmploymentStatus   < 4.5        to the right, improve=21.19408, (0 missing)
##       Sales_Channel      < 1.5        to the right, improve=16.47464, (0 missing)
##       Total_Claim_Amount < -0.4618456 to the left,  improve=16.03380, (0 missing)
##   Surrogate splits:
##       Customer_Lifetime_Value < -0.8729307 to the right, agree=0.839, adj=0.001, (0 split)
##
## Node number 6: 3908 observations,    complexity param=0.01904506
##   predicted class=0  expected loss=0.1742579  P(node) =0.611198
##     class counts:  3227   681
##    probabilities: 0.826 0.174
##   left son=12 (1117 obs) right son=13 (2791 obs)
##   Primary splits:
##       Income             < -0.900723  to the left,  improve=19.25914, (0 missing)
##       EmploymentStatus   < 4.5        to the right, improve=18.79774, (0 missing)
##       Renew_Offer_Type   < 1.5        to the left,  improve=16.86070, (0 missing)
##       Sales_Channel      < 1.5        to the right, improve=12.97736, (0 missing)
##       Total_Claim_Amount < -0.5269586 to the left,  improve=10.55279, (0 missing)
##   Surrogate splits:
##       EmploymentStatus        < 4.5        to the right, agree=0.998, adj=0.994, (0 split)
##       Total_Claim_Amount      < 0.9343285  to the right, agree=0.751, adj=0.129, (0 split)
##       Marital_Status          < 2.5        to the right, agree=0.740, adj=0.090, (0 split)
##       Customer_Lifetime_Value < -0.8162598 to the left,  agree=0.728, adj=0.047, (0 split)
##
## Node number 7: 750 observations,    complexity param=0.01421674
##   predicted class=0  expected loss=0.3093333  P(node) =0.1172975
##     class counts:   518   232
##    probabilities: 0.691 0.309
##   left son=14 (438 obs) right son=15 (312 obs)
##   Primary splits:
##       Income                  < -0.3808399 to the right, improve=23.721620, (0 missing)
##       EmploymentStatus        < 3.5        to the left,  improve=14.556200, (0 missing)
##       Total_Claim_Amount      < -0.4255145 to the left,  improve=12.664510, (0 missing)
##       Customer_Lifetime_Value < -0.8157533 to the right, improve=10.373730, (0 missing)
##       Monthly_Premium_Auto    < -0.7765437 to the right, improve= 7.850406, (0 missing)
##   Surrogate splits:
##       EmploymentStatus              < 2.5        to the left,  agree=0.847, adj=0.631, (0 split)
##       Total_Claim_Amount            < 0.1364937  to the left,  agree=0.656, adj=0.173, (0 split)
##       Customer_Lifetime_Value       < -0.810636  to the right, agree=0.625, adj=0.099, (0 split)
##       Monthly_Premium_Auto          < -0.8056067 to the right, agree=0.591, adj=0.016, (0 split)
##       Months_Since_Policy_Inception < 1.55649    to the left,  agree=0.588, adj=0.010, (0 split)
##
## Node number 12: 1117 observations
##   predicted class=0  expected loss=0.0957923  P(node) =0.174695
##     class counts:  1010   107
##    probabilities: 0.904 0.096
##
## Node number 13: 2791 observations,    complexity param=0.01904506
```

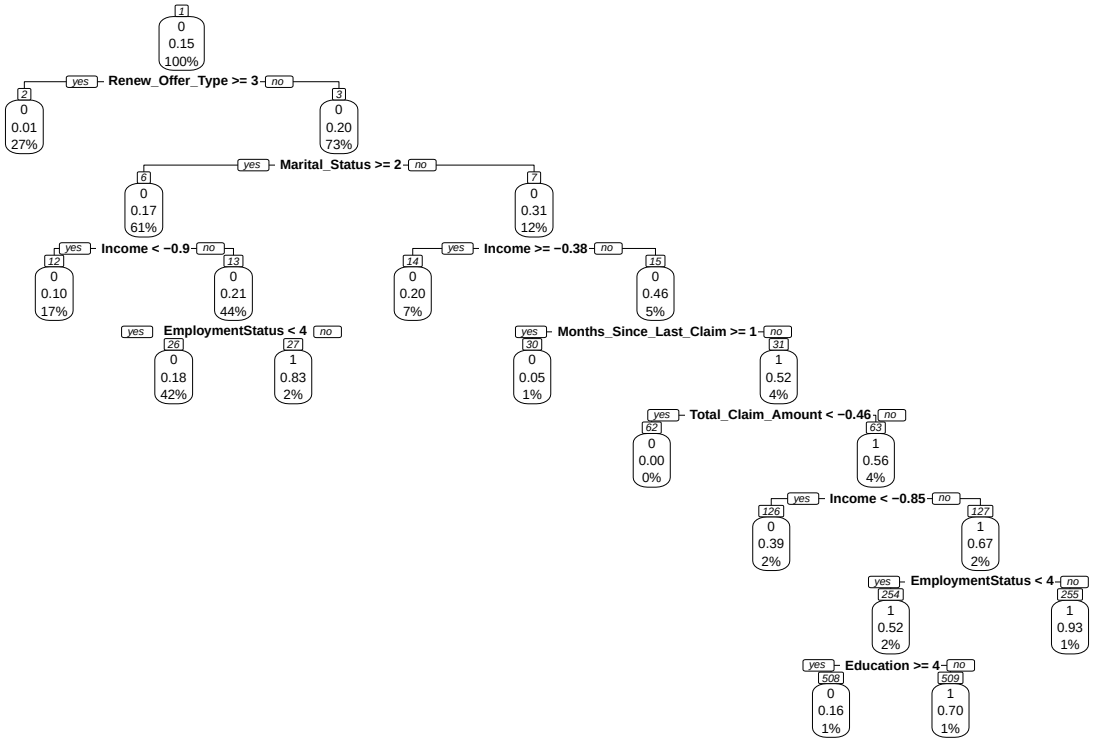
```
##   predicted class=0  expected loss=0.2056611  P(node) =0.436503
##     class counts:  2217   574
##    probabilities: 0.794 0.206
##   left son=26 (2684 obs) right son=27 (107 obs)
##   Primary splits:
##       EmploymentStatus   < 3.5        to the left,  improve=87.23662, (0 missing)
##       Total_Claim_Amount < -0.5269586 to the left,  improve=20.47424, (0 missing)
##       Location_Code      < 2.5        to the right, improve=12.51777, (0 missing)
##       Renew_Offer_Type   < 1.5        to the left,  improve=12.42603, (0 missing)
##       Sales_Channel      < 1.5        to the right, improve=12.24559, (0 missing)
##   Surrogate splits:
##       Income                  < -0.8884781 to the right, agree=0.965, adj=0.075, (0 split)
##       Customer_Lifetime_Value < -0.8349271 to the right, agree=0.963, adj=0.028, (0 split)
##
## Node number 14: 438 observations
##   predicted class=0  expected loss=0.2031963  P(node) =0.06850172
##     class counts:   349    89
##    probabilities: 0.797 0.203
##
## Node number 15: 312 observations,    complexity param=0.01421674
##   predicted class=0  expected loss=0.4583333  P(node) =0.04879575
##     class counts:   169   143
##    probabilities: 0.542 0.458
##   left son=30 (39 obs) right son=31 (273 obs)
##   Primary splits:
##       Months_Since_Last_Claim < 1.032735   to the right, improve=14.770150, (0 missing)
##       Income                  < -0.8519737 to the left,  improve=13.450730, (0 missing)
##       Total_Claim_Amount      < -0.4598129 to the left,  improve=12.923710, (0 missing)
##       EmploymentStatus        < 4.5        to the right, improve=10.991790, (0 missing)
##       Customer_Lifetime_Value < 0.2840898  to the left,  improve= 9.392127, (0 missing)
##
## Node number 26: 2684 observations
##   predicted class=0  expected loss=0.1807004  P(node) =0.4197685
##     class counts:  2199   485
##    probabilities: 0.819 0.181
##
## Node number 27: 107 observations
##   predicted class=1  expected loss=0.1682243  P(node) =0.01673444
##     class counts:    18    89
##    probabilities: 0.168 0.832
##
## Node number 30: 39 observations
##   predicted class=0  expected loss=0.05128205  P(node) =0.006099468
##     class counts:    37     2
##    probabilities: 0.949 0.051
##
## Node number 31: 273 observations,    complexity param=0.01421674
##   predicted class=1  expected loss=0.4835165  P(node) =0.04269628
##     class counts:   132   141
##    probabilities: 0.484 0.516
##   left son=62 (22 obs) right son=63 (251 obs)
##   Primary splits:
##       Total_Claim_Amount      < -0.4598129 to the left,  improve=12.765990, (0 missing)
##       Income                  < -0.8519737 to the left,  improve=12.549180, (0 missing)
```

```
##       EmploymentStatus        < 4.5        to the right, improve=10.253690, (0 missing)
##       Customer_Lifetime_Value < 0.2840898  to the left,  improve= 8.864469, (0 missing)
##       Location_Code           < 2.5        to the right, improve= 6.696476, (0 missing)
##   Surrogate splits:
##       Location_Code           < 2.5        to the right, agree=0.963, adj=0.545, (0 split)
##       Monthly_Premium_Auto    < -0.8927959 to the left,  agree=0.938, adj=0.227, (0 split)
##       Customer_Lifetime_Value < -0.833638  to the left,  agree=0.923, adj=0.045, (0 split)
##
## Node number 62: 22 observations
##   predicted class=0  expected loss=0  P(node) =0.003440726
##     class counts:    22     0
##    probabilities: 1.000 0.000
##
## Node number 63: 251 observations,    complexity param=0.01421674
##   predicted class=1  expected loss=0.438247  P(node) =0.03925555
##     class counts:   110   141
##    probabilities: 0.438 0.562
##   left son=126 (96 obs) right son=127 (155 obs)
##   Primary splits:
##       Income                  < -0.8519737 to the left,  improve=9.667781, (0 missing)
##       Customer_Lifetime_Value < -0.7443817 to the right, improve=8.763435, (0 missing)
##       EmploymentStatus        < 4.5        to the right, improve=7.937910, (0 missing)
##       Number_of_Policies      < 2.315234   to the right, improve=4.544674, (0 missing)
##       Monthly_Premium_Auto    < 0.2406626  to the right, improve=4.376346, (0 missing)
##   Surrogate splits:
##       EmploymentStatus          < 4.5        to the right, agree=0.988, adj=0.969, (0 split)
##       Vehicle_Size              < 2.5        to the right, agree=0.673, adj=0.146, (0 split)
##       Customer_Lifetime_Value   < 0.9578242  to the right, agree=0.661, adj=0.115, (0 split)
##       Number_of_Open_Complaints < 1.225431   to the right, agree=0.653, adj=0.094, (0 split)
##       Education                 < 4.5        to the right, agree=0.637, adj=0.052, (0 split)
##
## Node number 126: 96 observations
##   predicted class=0  expected loss=0.3854167  P(node) =0.01501408
##     class counts:    59    37
##    probabilities: 0.615 0.385
##
## Node number 127: 155 observations,    complexity param=0.01180258
##   predicted class=1  expected loss=0.3290323  P(node) =0.02424148
##     class counts:    51   104
##    probabilities: 0.329 0.671
##   left son=254 (98 obs) right son=255 (57 obs)
##   Primary splits:
##       EmploymentStatus             < 3.5        to the left,  improve=12.081750, (0 missing)
##       Number_of_Policies           < 2.315234   to the right, improve= 6.600872, (0 missing)
##       Customer_Lifetime_Value      < -0.7877115 to the right, improve= 5.275323, (0 missing)
##       Months_Since_Policy_Inception < -1.417781  to the left,  improve= 4.610038, (0 missing)
##       Vehicle_Class                < 5.5        to the right, improve= 4.585292, (0 missing)
##   Surrogate splits:
##       Number_of_Open_Complaints < 0.1269926  to the left,  agree=0.671, adj=0.105, (0 split)
##       Total_Claim_Amount        < -0.3658708 to the right, agree=0.671, adj=0.105, (0 split)
##       Months_Since_Last_Claim   < 0.8838254  to the left,  agree=0.665, adj=0.088, (0 split)
##       Customer_Lifetime_Value   < -0.8278004 to the right, agree=0.658, adj=0.070, (0 split)
##       Education                 < 3.5        to the left,  agree=0.652, adj=0.053, (0 split)
##
```

```
## Node number 254: 98 observations,    complexity param=0.01180258
##   predicted class=1  expected loss=0.4795918  P(node) =0.01532687
##     class counts:    47    51
##    probabilities: 0.480 0.520
##   left son=508 (32 obs) right son=509 (66 obs)
##   Primary splits:
##       Education                < 3.5        to the right, improve=12.602080, (0 missing)
##       Number_of_Open_Complaints < 0.1269926  to the right, improve= 5.367806, (0 missing)
##       Customer_Lifetime_Value   < -0.7911586 to the right, improve= 4.860449, (0 missing)
##       State                    < 4.5        to the right, improve= 4.083203, (0 missing)
##       Income                   < -0.3898919 to the left,  improve= 3.743864, (0 missing)
##   Surrogate splits:
##       Months_Since_Last_Claim   < 0.7349162  to the right, agree=0.704, adj=0.094, (0 split)
##       Number_of_Open_Complaints < 0.1269926  to the right, agree=0.704, adj=0.094, (0 split)
##       Total_Claim_Amount        < 3.150641   to the right, agree=0.694, adj=0.063, (0 split)
##       Vehicle_Size              < 2.5        to the right, agree=0.684, adj=0.031, (0 split)
##       Customer_Lifetime_Value   < 2.49875    to the right, agree=0.684, adj=0.031, (0 split)
##
## Node number 255: 57 observations
##   predicted class=1  expected loss=0.07017544  P(node) =0.008914607
##     class counts:     4    53
##    probabilities: 0.070 0.930
##
## Node number 508: 32 observations
##   predicted class=0  expected loss=0.15625  P(node) =0.005004692
##     class counts:    27     5
##    probabilities: 0.844 0.156
##
## Node number 509: 66 observations
##   predicted class=1  expected loss=0.3030303  P(node) =0.01032218
##     class counts:    20    46
##    probabilities: 0.303 0.697
```

plot tree

```
prp(data.tree, type = 2, extra = "auto", nn = TRUE, branch = 1, varlen = 0, yesno = 2)
```



Get the number of nodes

```
num_nodes <- data.tree$n
num_nodes
```

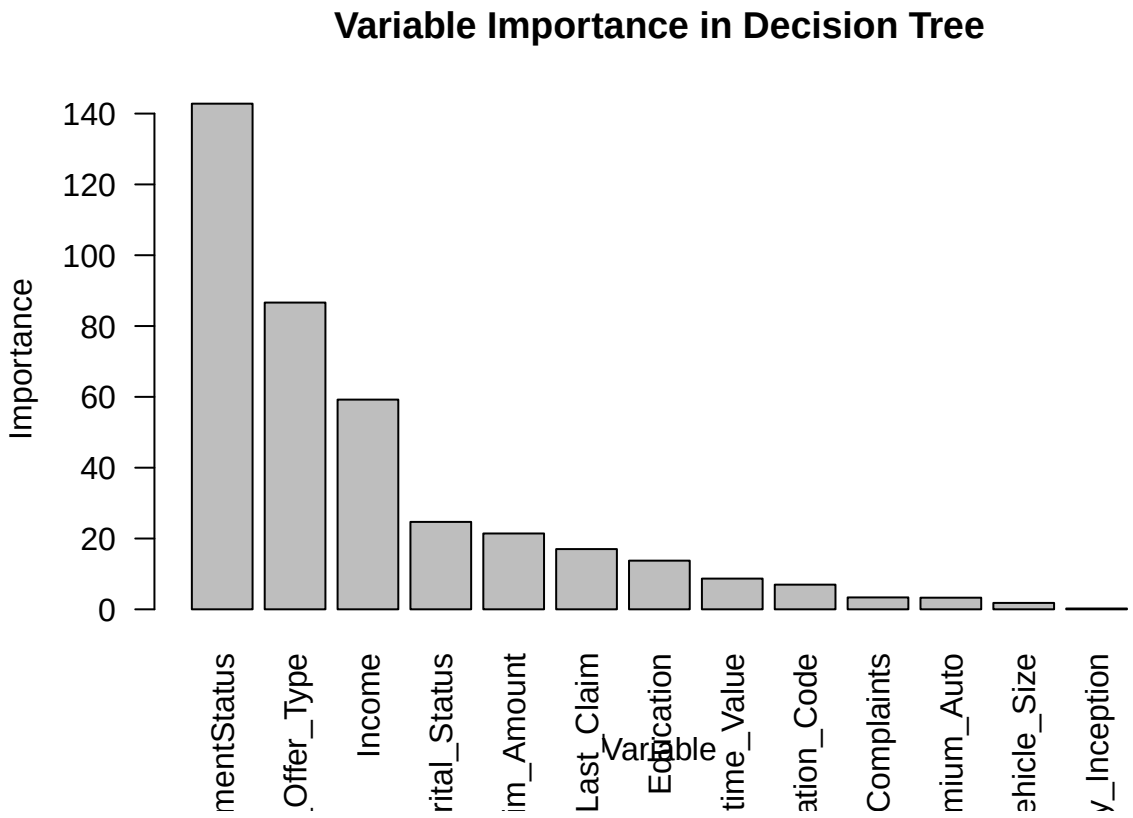
```
## [1] 4
```

Get variable importance

```
var_importance <- data.tree$variable.importance
var_importance
```

```
##              EmploymentStatus                Renew_Offer_Type
##                     142.8005446                     86.6247246
##                        Income                  Marital_Status
##                     59.2207982                     24.6856104
##            Total_Claim_Amount          Months_Since_Last_Claim
##                     21.4138733                     17.0113937
##                     Education          Customer_Lifetime_Value
##                     13.7414911                      8.6769706
##                 Location_Code        Number_of_Open_Complaints
##                      6.9632678                      3.3595621
##           Monthly_Premium_Auto                   Vehicle_Size
##                      3.2815157                      1.8036997
## Months_Since_Policy_Inception
##                      0.2280925
```

```r
# Create a bar plot of variable importance with rotated x-axis labels
barplot(var_importance, main = "Variable Importance in Decision Tree",
        xlab = "Variable", ylab = "Importance",
        names.arg = names(var_importance), las = 2)
```



TRAIN DATA Predict train data

```r
pred = predict(data.tree, train, type = 'prob')
pred_value <- ifelse(pred[, "1"] > 0.5, 1, 0)
# accuracy train data
result = table(pred_value, train$Response)
result
```

```
##
## pred_value    0    1
##          0 5420  744
##          1   42  188
```

Calculate accuracy, precision, recall, F1-score

```r
TN <- result[1, 1]
TP <- result[2, 2]
FP <- result[1, 2]
FN <- result[2, 1]
```

```r
accuracy <- (TN + TP) / (TN + TP + FP + FN)
accuracy
```

```
## [1] 0.8770723
```

```r
precision <- TP / (TP + FP)
precision
```

```
## [1] 0.2017167
```

```r
recall <- TP / (TP + FN)
recall
```

```
## [1] 0.8173913
```

```r
f1_score <- 2 * (precision * recall) / (precision + recall)
f1_score
```

```
## [1] 0.32358
```

TEST DATA

Predict test data

```r
pred2 = predict(data.tree, test, type = 'prob')
pred_value2 <- ifelse(pred2[, "1"] > 0.5, 1, 0)
# accuracy test data
result = table(pred_value2, test$Response)
result
```

```
##
## pred_value2    0    1
##           0 2340  312
##           1   24   64
```

Calculate accuracy, precision, recall, F1-score

```r
TN <- result[1, 1]
TP <- result[2, 2]
FP <- result[1, 2]
FN <- result[2, 1]

accuracy <- (TN + TP) / (TN + TP + FP + FN)
accuracy
```

```
## [1] 0.8773723
```

```r
precision <- TP / (TP + FP)
precision
```

```
## [1] 0.1702128
```

```r
recall <- TP / (TP + FN)
recall
```

```
## [1] 0.7272727
```

```r
f1_score <- 2 * (precision * recall) / (precision + recall)
f1_score
```

```
## [1] 0.2758621
```

ROC Curve

```r
# Create ROC data
roc_data <- roc(train$Response, pred_value)
```

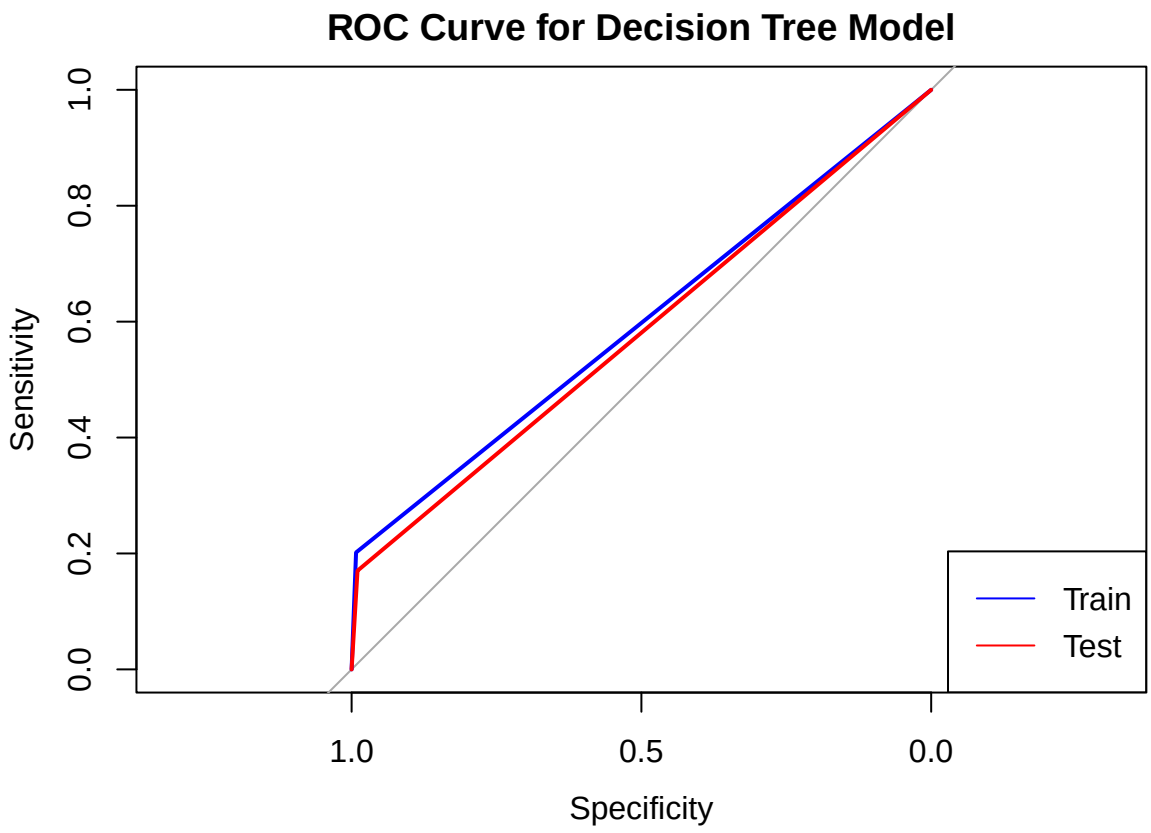
```
## Setting levels: control = 0, case = 1
```

```
## Setting direction: controls < cases
```

```r
roc_data2 <- roc(test$Response, pred_value2)
```

```
## Setting levels: control = 0, case = 1
## Setting direction: controls < cases
```

```r
# Combine ROC curves into a single plot
plot(roc_data, col = "blue", main = "ROC Curve for Decision Tree Model")
lines(roc_data2, col = "red")
legend("bottomright", legend = c("Train", "Test"), col = c("blue", "red"), lty = 1)
```



3. RANDOM FOREST

Build model

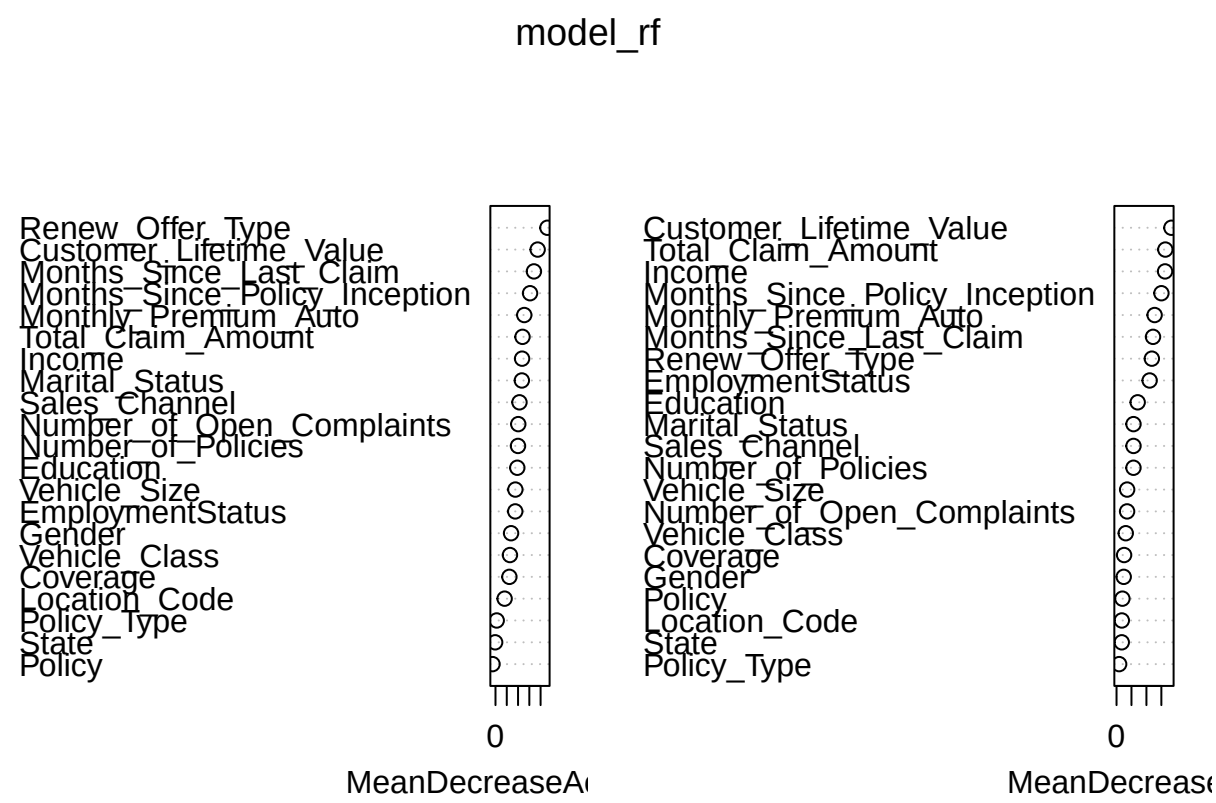
```
set.seed(42)
train$Response <- factor(train$Response)
model_rf <- randomForest(Response ~ ., data = train, ntree = 15, mtry = 7, importance = TRUE)
model_rf
```

```
##
## Call:
##  randomForest(formula = Response ~ ., data = train, ntree = 15,      mtry = 7, importance = TRUE)
##                Type of random forest: classification
##                      Number of trees: 15
## No. of variables tried at each split: 7
##
##         OOB estimate of  error rate: 1.77%
## Confusion matrix:
##      0   1 class.error
## 0 5374  82  0.01502933
## 1   31 901  0.03326180
```

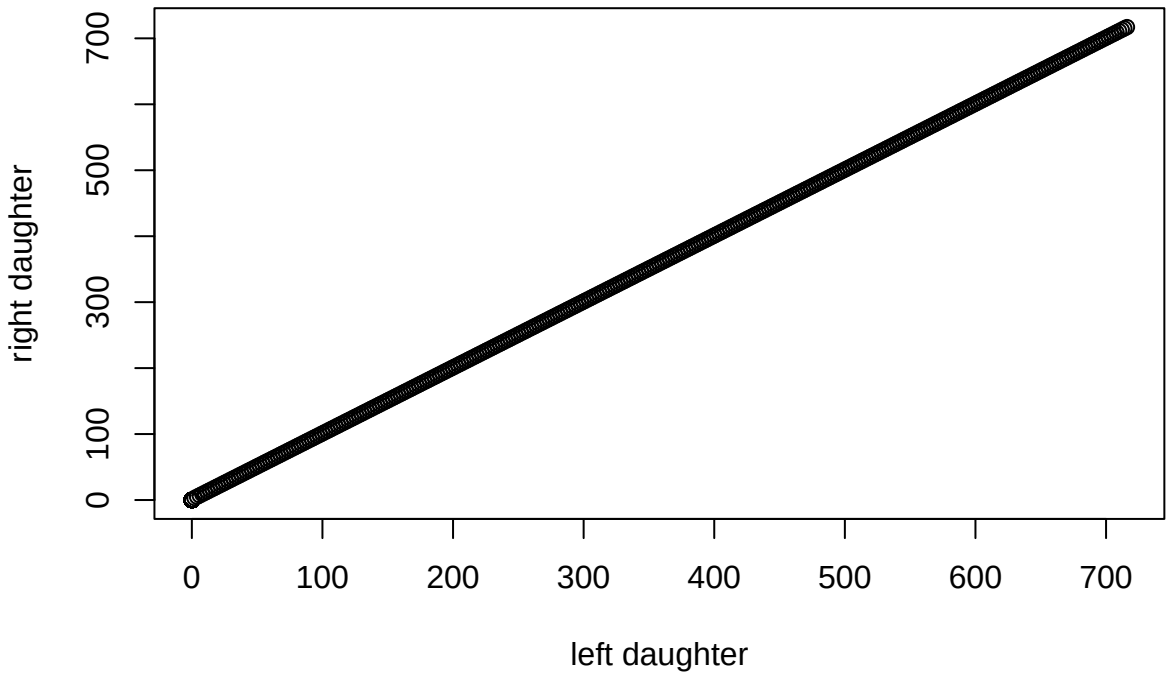
```
summary(model_rf)
```

```
##                 Length Class  Mode
## call                 6  -none- call
## type                 1  -none- character
## predicted         6394  factor numeric
## err.rate            45  -none- numeric
## confusion            6  -none- numeric
## votes            12788  matrix numeric
## oob.times         6394  -none- numeric
## classes              2  -none- character
## importance          84  -none- numeric
## importanceSD        63  -none- numeric
## localImportance      0  -none- NULL
## proximity            0  -none- NULL
## ntree                1  -none- numeric
## mtry                 1  -none- numeric
## forest              14  -none- list
## y                 6394  factor numeric
## test                 0  -none- NULL
## inbag                0  -none- NULL
## terms                3  terms  call
```

```r
varImpPlot(model_rf)
```




```r
library(randomForest)
tree_plot <- getTree(model_rf)
plot(tree_plot)
```



TRAIN DATA

Predicting on train set

```
pred <- predict(model_rf, train, type = "response")
# Checking classification accuracy
result = table(pred, train$Response)
result
```

```
##
## pred    0    1
##    0 5462    1
##    1    0  931
```

Calculate accuracy, precision, recall, F1-score

```
TN <- result[1, 1]
TP <- result[2, 2]
FP <- result[1, 2]
FN <- result[2, 1]

accuracy <- (TN + TP) / (TN + TP + FP + FN)
accuracy
```

```
## [1] 0.9998436
```

```
precision <- TP / (TP + FP)
precision
```

```
## [1] 0.998927
```

```
recall <- TP / (TP + FN)
recall
```

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

```
f1_score <- 2 * (precision * recall) / (precision + recall)
f1_score
```

```
## [1] 0.9994632
```

TEST DATA

Predicting on test set

```
pred2 <- predict(model_rf, test, type = "class")
# Checking classification accuracy
result = table(pred2, test$Response)
result
```

```
##
## pred2    0    1
##     0 2348   12
##     1   16  364
```

Calculate accuracy, precision, recall, F1-score

```
TN <- result[1, 1]
TP <- result[2, 2]
FP <- result[1, 2]
FN <- result[2, 1]

accuracy <- (TN + TP) / (TN + TP + FP + FN)
accuracy
```

```
## [1] 0.989781
```

```
precision <- TP / (TP + FP)
precision
```

```
## [1] 0.9680851
```

```r
recall <- TP / (TP + FN)
recall
```

```
## [1] 0.9578947
```

```r
f1_score <- 2 * (precision * recall) / (precision + recall)
f1_score
```

```
## [1] 0.962963
```

ROC Curve

```r
# Create a binary vector indicating if the predicted class is 1 or 0
pred_class <- as.numeric(pred == "1")
pred_class2 <- as.numeric(pred2 == "1")

# Create ROC data
roc_data <- roc(train$Response, pred_class)
```

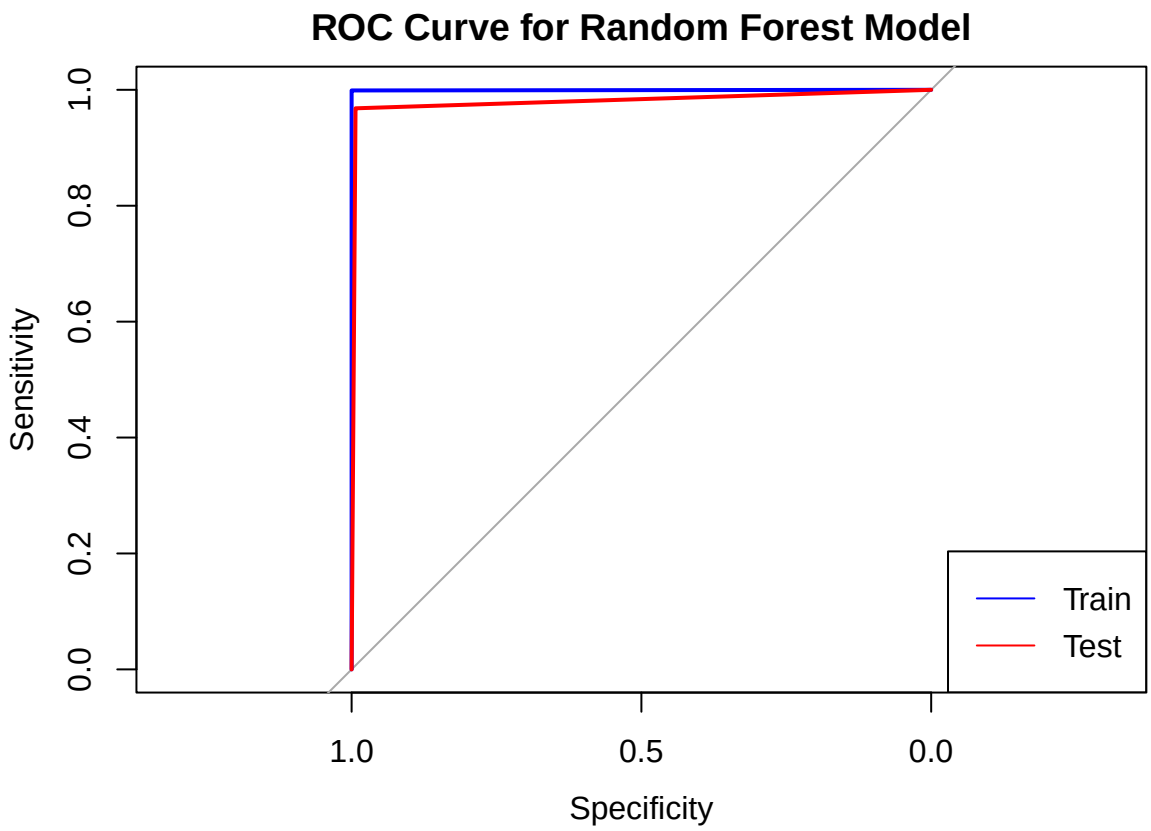
```
## Setting levels: control = 0, case = 1
```

```
## Setting direction: controls < cases
```

```r
roc_data2 <- roc(test$Response, pred_class2)
```

```
## Setting levels: control = 0, case = 1
## Setting direction: controls < cases
```

```r
# Combine ROC curves into a single plot
plot(roc_data, col = "blue", main = "ROC Curve for Random Forest Model")
lines(roc_data2, col = "red")
legend("bottomright", legend = c("Train", "Test"), col = c("blue", "red"), lty = 1)
```



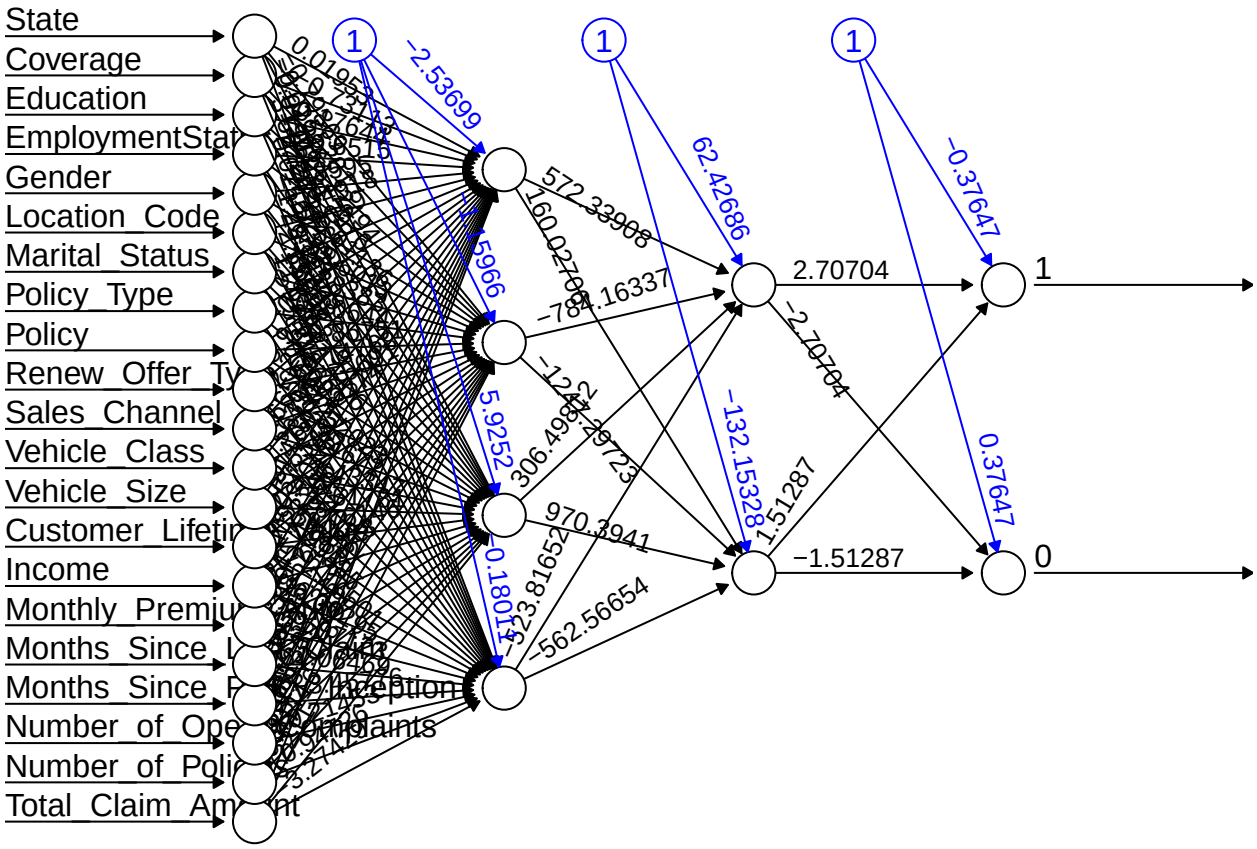
4. NEUTRAL NETWORK

Build model

```
set.seed(123)
model = neuralnet(Response ~ ., data=train, hidden=c(4,2),linear.output = FALSE, act.fct = "logistic")
summary(model)
```

```
##                       Length Class      Mode
## call                       6 -none-     call
## response               12788 -none-     logical
## covariate             134274 -none-     numeric
## model.list                 2 -none-     list
## err.fct                    1 -none-     function
## act.fct                    1 -none-     function
## linear.output              1 -none-     logical
## data                      22 data.frame list
## exclude                    0 -none-     NULL
## net.result                 1 -none-     list
## weights                    1 -none-     list
## generalized.weights        1 -none-     list
## startweights               1 -none-     list
## result.matrix            107 -none-     numeric
```

```
plot(model,rep = "best")
```



TRAIN DATA

Predicting on train set

```
pred <- predict(model, newdata = train)
pred_class <- ifelse(pmax(pred) > 0.5, 1, 0)[,2]
result <- table(train$Response, pred_class)
result
```

```
##    pred_class
##        0    1
##   0 5265  197
##   1  645  287
```

Calculate accuracy, precision, recall, F1-score

```
TN <- result[1, 1]
TP <- result[2, 2]
FP <- result[1, 2]
FN <- result[2, 1]
```

```r
accuracy <- (TN + TP) / (TN + TP + FP + FN)
accuracy
```

```
## [1] 0.868314
```

```r
precision <- TP / (TP + FP)
precision
```

```
## [1] 0.5929752
```

```r
recall <- TP / (TP + FN)
recall
```

```
## [1] 0.3079399
```

```r
f1_score <- 2 * (precision * recall) / (precision + recall)
f1_score
```

```
## [1] 0.4053672
```

TEST DATA

Predicting on test set

```r
pred2 <- predict(model, newdata = test)
pred_class <- ifelse(pmax(pred2) > 0.5, 1, 0)[,2]
result <- table(test$Response, pred_class)
result
```

```
##    pred_class
##        0    1
##   0 2237  127
##   1  265  111
```

Calculate accuracy, precision, recall, F1-score

```r
TN <- result[1, 1]
TP <- result[2, 2]
FP <- result[1, 2]
FN <- result[2, 1]

accuracy <- (TN + TP) / (TN + TP + FP + FN)
accuracy
```

```
## [1] 0.8569343
```

```r
precision <- TP / (TP + FP)
precision
```

```
## [1] 0.4663866
```

```r
recall <- TP / (TP + FN)
recall
```

```
## [1] 0.2952128
```

```r
f1_score <- 2 * (precision * recall) / (precision + recall)
f1_score
```

```
## [1] 0.3615635
```

ROC Curve

```r
# Create ROC curve
roc_data <- roc(train$Response, pred[,2])
```

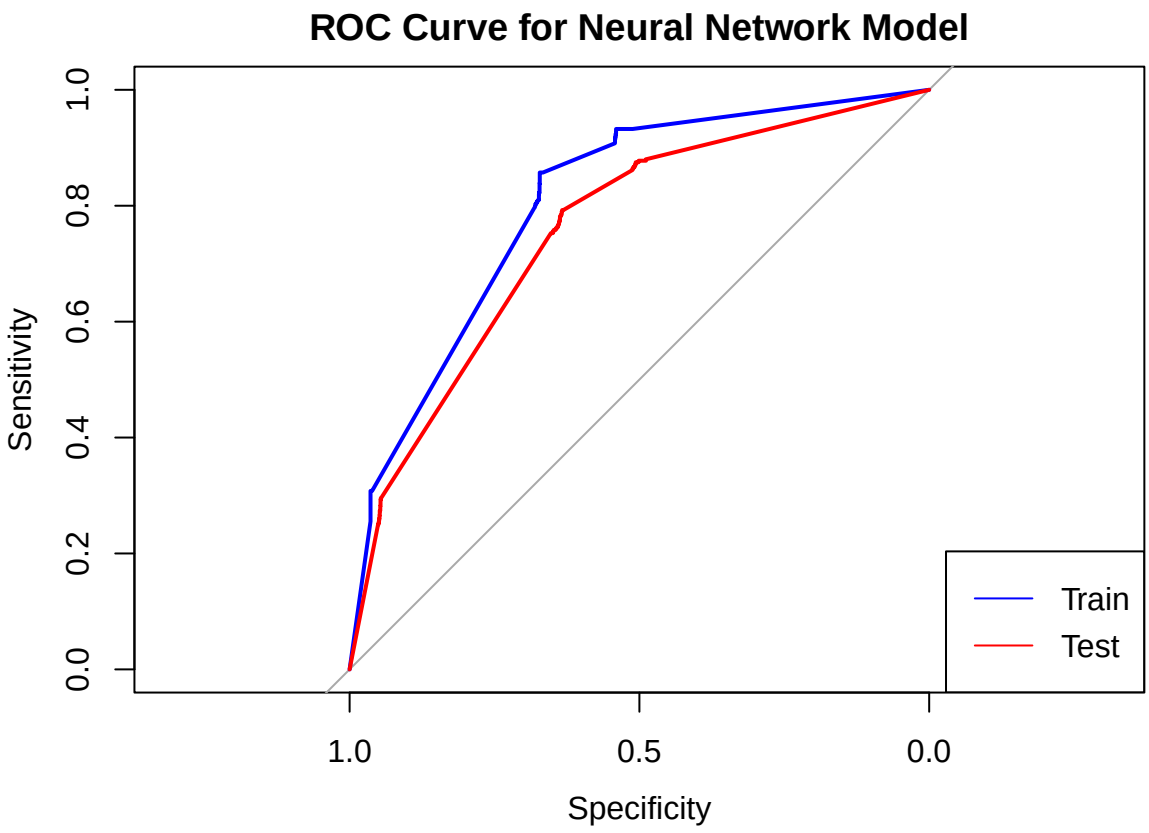
```
## Setting levels: control = 0, case = 1
```

```
## Setting direction: controls < cases
```

```r
roc_data2 <- roc(test$Response, pred2[,2])
```

```
## Setting levels: control = 0, case = 1
## Setting direction: controls < cases
```

```r
# Combine ROC curves into a single plot
plot(roc_data, col = "blue", main = "ROC Curve for Neural Network Model")
lines(roc_data2, col = "red")
legend("bottomright", legend = c("Train", "Test"), col = c("blue", "red"), lty = 1)
```


ROC Curve for Neural Network Model
Sensitivity
0.0
0.2
0.4
0.6
0.8
1.0
1.0
0.5
0.0
Specificity
Train
Test