SWPPredictor_RandomForest

Giriraj

2022-11-30

```r
# importing dataset for the prediction model
library(readr)
df <- read_csv("dataset/dataset.csv")
```

```
## Rows: 334 Columns: 15
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr  (3): Education, Age, Household_Income
## dbl (12): Mental_illness, Own_computer, days_hospitalized, Disabled, Interne...
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```r
head(df)
```

```
## # A tibble: 6 x 15
##   Mental_illness Educa~1 Own_c~2 days_~3 Disab~4 Inter~5 Live_~6 Lengt~7 Annua~8
##            <dbl> <chr>     <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>
## 1              0 High S~       0       0       0       1       0      24      35
## 2              1 Some P~       1       0       0       1       0       1      22
## 3              0 Comple~       1       0       0       1       0       0     100
## 4              0 Some U~       1       0       0       1       1      11       0
## 5              1 Comple~       1      35       1       1       0      33      32
## 6              0 High S~       1       0       0       1       1       0       0
## # ... with 6 more variables: Unemployed <dbl>, Read_books <dbl>, SWP <dbl>,
## #   Times_hospitalized <dbl>, Age <chr>, Household_Income <chr>, and
## #   abbreviated variable names 1: Education, 2: Own_computer,
## #   3: days_hospitalized, 4: Disabled, 5: Internet_access,
## #   6: Live_with_parents, 7: Length_of_resume_gap_month,
## #   8: Annual_income_and_SWP
```

```r
# manipulating data to represent actual value
# multiplying variable SWP * 100 to represent SWP received per month
# multiplying variable Annual_income_and_SWP * 1000 to represent Annual_income_and_SWP received per yea

df$Annual_income_and_SWP <- df$Annual_income_and_SWP *1000
df$SWP <- df$SWP * 100
df
```

```
## # A tibble: 334 x 15
```

```
##    Mental_illn~1 Educa~2 Own_c~3 days_~4 Disab~5 Inter~6 Live_~7 Lengt~8 Annua~9
##            <dbl> <chr>     <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>
##  1             0 High S~       0       0       0       1       0      24   35000
##  2             1 Some P~       1       0       0       1       0       1   22000
##  3             0 Comple~       1       0       0       1       0       0  100000
##  4             0 Some U~       1       0       0       1       1      11       0
##  5             1 Comple~       1      35       1       1       0      33   32000
##  6             0 High S~       1       0       0       1       1       0       0
##  7             0 Some U~       1       0       0       1       0       0    1000
##  8             1 Some U~       1       0       0       1       1       0   11000
##  9             0 Comple~       1       0       0       1       0       0   73000
## 10             1 Some M~       1       0       0       1       0       0   12000
## # ... with 324 more rows, 6 more variables: Unemployed <dbl>, Read_books <dbl>,
## #   SWP <dbl>, Times_hospitalized <dbl>, Age <chr>, Household_Income <chr>, and
## #   abbreviated variable names 1: Mental_illness, 2: Education,
## #   3: Own_computer, 4: days_hospitalized, 5: Disabled, 6: Internet_access,
## #   7: Live_with_parents, 8: Length_of_resume_gap_month,
## #   9: Annual_income_and_SWP
```

```r
# mutating data for better analysis and reporting

library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```r
df%>%
  mutate(Education=case_when(
    .$Education=="High School or GED" ~ 1,
    .$Education=="Some highschool" ~ 1,
    .$Education=="Completed Undergraduate" ~ 2,
    .$Education=="Some Undergraduate" ~ 2,
    .$Education=="Completed Masters" ~ 3,
    .$Education=="Some Maters" ~ 3,
    .$Education=="Some Phd" ~ 4,
    .$Education=="Completed Phd" ~ 4
    )) -> df
```

```r
# mutating data for better analysis and reporting

df%>%
  mutate(Age=case_when(
    .$Age =="18-29" ~ 1,
    .$Age =="30-44" ~ 2,
```

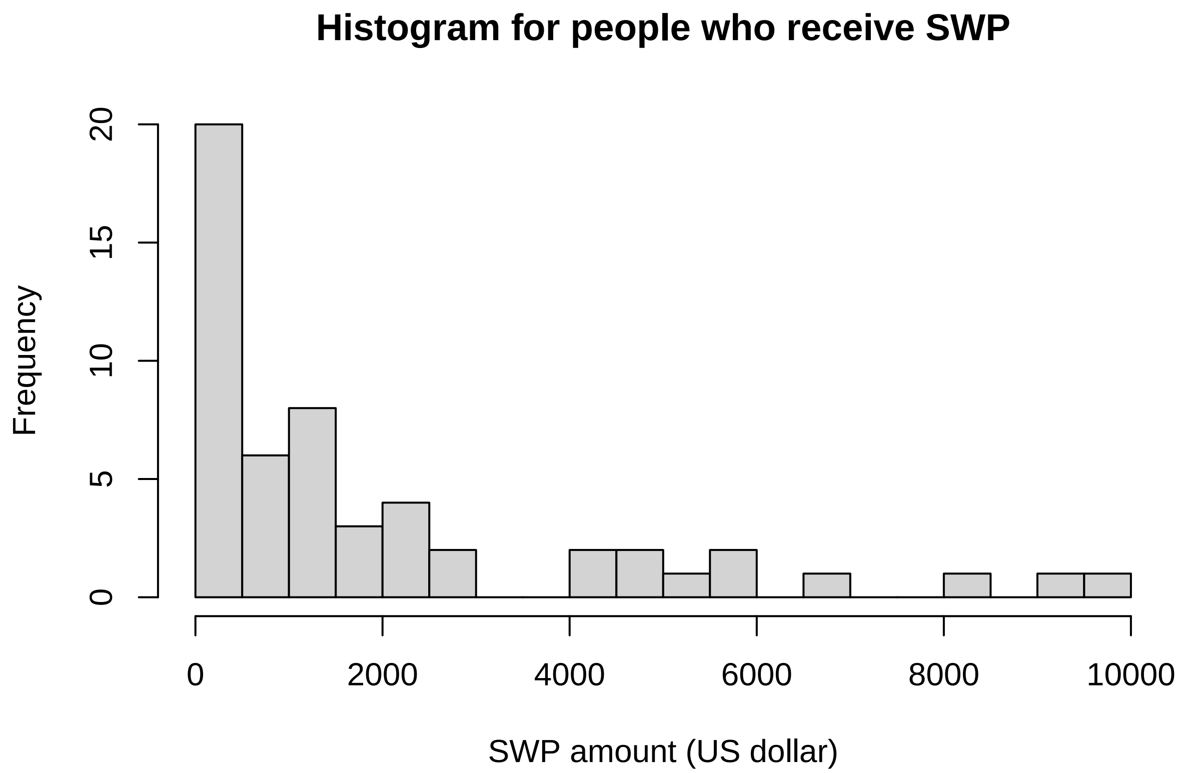
```
    .$Age =="45-60" ~ 3,
    .$Age =="Greater than 60" ~ 4
    )) -> df
```

```
# mutating data for better analysis and reporting

df%>%
  mutate(Household_Income=case_when(
    .$Household_Income =="$0-$9,999" ~ 1,
    .$Household_Income =="$10,000-$24,999" ~ 2,
    .$Household_Income =="$25,000-$49,999" ~ 3,
    .$Household_Income =="$50,000-$74,999" ~ 4,
    .$Household_Income =="$75,000-$99,999" ~ 5,
    .$Household_Income =="$100,000-$124,999" ~ 6,
    .$Household_Income =="$125,000-$149,999" ~ 7,
    .$Household_Income =="$150,000-$174,999" ~ 8,
    .$Household_Income =="$175,000-$199,999" ~ 9,
    .$Household_Income =="$200,000+" ~ 10,
    .$Household_Income =="Prefer not to answer" ~ 996
    )) -> df
```

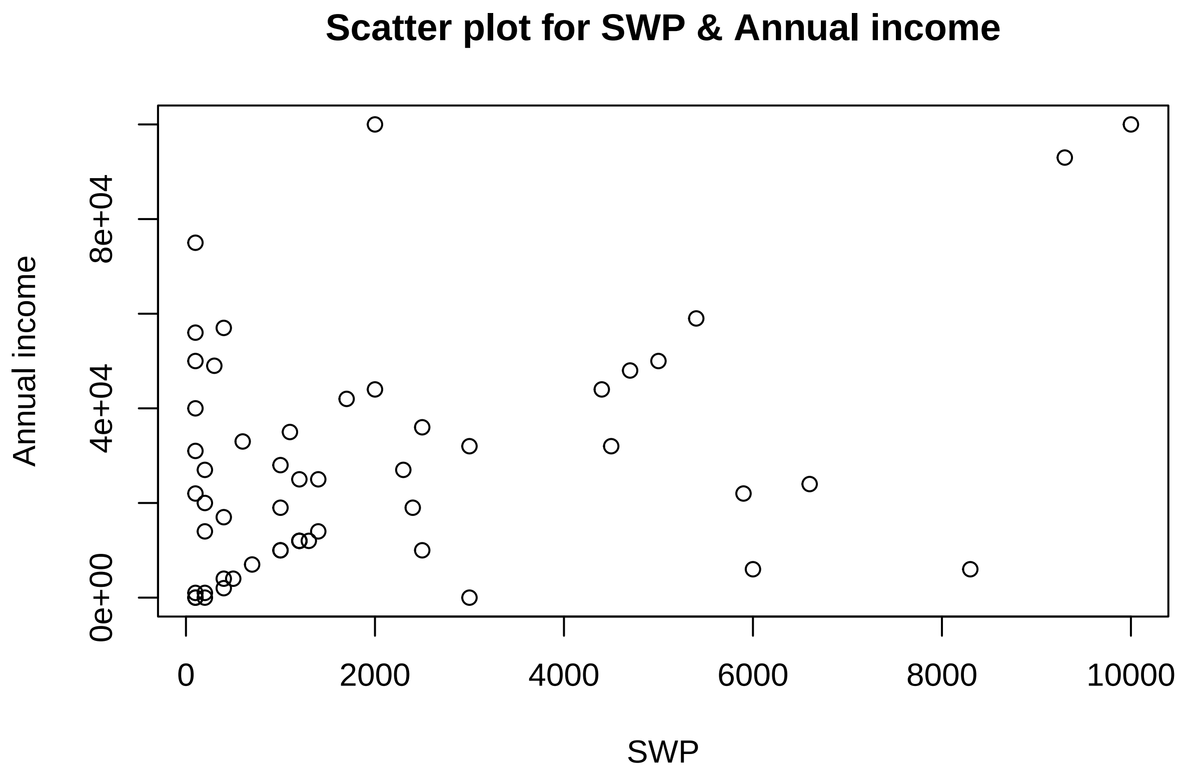
```
head(df)
```

```
## # A tibble: 6 x 15
##   Mental_illness Educa~1 Own_c~2 days_~3 Disab~4 Inter~5 Live_~6 Lengt~7 Annua~8
##            <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>
## 1              0       1       0       0       0       1       0      24   35000
## 2              1       4       1       0       0       1       0       1   22000
## 3              0       2       1       0       0       1       0       0  100000
## 4              0       2       1       0       0       1       1      11       0
## 5              1       2       1      35       1       1       0      33   32000
## 6              0       1       1       0       0       1       1       0       0
## # ... with 6 more variables: Unemployed <dbl>, Read_books <dbl>, SWP <dbl>,
## #   Times_hospitalized <dbl>, Age <dbl>, Household_Income <dbl>, and
## #   abbreviated variable names 1: Education, 2: Own_computer,
## #   3: days_hospitalized, 4: Disabled, 5: Internet_access,
## #   6: Live_with_parents, 7: Length_of_resume_gap_month,
## #   8: Annual_income_and_SWP
```

```
# filtering individuals who receive SWP for histogram
df$SWP_valid <- df$SWP != 0

# generating histogram for SWP to view its distribution
hist(df$SWP[df$SWP_valid], breaks = 20, main = 'Histogram for people who receive SWP', xlab = 'SWP amou
```




```
# generating scatter plot for SWP and Annual Income to view its relationship and understand its dependa
plot(x = df$SWP[df$SWP_valid], y = df$Annual_income_and_SWP[df$SWP_valid], main = 'Scatter plot for SWP
```




```r
# printing mean and standard deviation for SWP
print(paste('Mean value for SWP =', round(mean(df$SWP[df$SWP_valid]), 2)))
```

```
## [1] "Mean value for SWP = 2057.41"
```

```r
print(paste('Mean value for SWP =', round(sd(df$SWP[df$SWP_valid]), 2)))
```

```
## [1] "Mean value for SWP = 2488.36"
```

Creating linear regression models to get best set of variables to predict SWP

```r
# linear regression model for SWP ~ .
m1 <- lm(SWP ~. , data = df)
summary(m1)
```

```
##
## Call:
## lm(formula = SWP ~ ., data = df)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2191.0  -210.6    18.9   186.8  7361.0
##
## Coefficients:
```

```
##                              Estimate Std. Error t value Pr(>|t|)
## (Intercept)                  5.824e+02  3.841e+02   1.516   0.1305
## Mental_illness               2.639e+02  1.434e+02   1.840   0.0667 .
## Education                   -6.703e+01  7.379e+01  -0.908   0.3643
## Own_computer                 1.297e+01  1.735e+02   0.075   0.9404
## days_hospitalized           -2.577e+00  5.663e+00  -0.455   0.6493
## Disabled                    -3.684e+02  2.126e+02  -1.733   0.0841 .
## Internet_access              1.941e+01  2.971e+02   0.065   0.9479
## Live_with_parents           -1.441e+02  1.919e+02  -0.751   0.4532
## Length_of_resume_gap_month -2.869e+00  2.784e+00  -1.031   0.3035
## Annual_income_and_SWP        3.798e-03  1.879e-03   2.021   0.0441 *
## Unemployed                  -1.128e+02  1.400e+02  -0.805   0.4212
## Read_books                  -6.895e+02  1.738e+02  -3.968 8.96e-05 ***
## Times_hospitalized           2.167e+00  1.010e+01   0.215   0.8302
## Age                          9.578e+00  6.279e+01   0.153   0.8789
## Household_Income            -8.272e-02  1.751e-01  -0.473   0.6369
## SWP_validTRUE                2.262e+03  1.615e+02  14.004  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 971.1 on 316 degrees of freedom
##   (2 observations deleted due to missingness)
## Multiple R-squared:  0.4261, Adjusted R-squared:  0.3989
## F-statistic: 15.64 on 15 and 316 DF,  p-value: < 2.2e-16
```

```
# trimming features & checking for Multiple R-squared & Adjusted R-squared values to determine best sui
m2 <- lm(SWP ~ Education + Own_computer + days_hospitalized + Disabled + Internet_access + Live_with_pa
summary(m2)
```

```
##
## Call:
## lm(formula = SWP ~ Education + Own_computer + days_hospitalized +
##     Disabled + Internet_access + Live_with_parents + Length_of_resume_gap_month +
##     Annual_income_and_SWP + Unemployed + Read_books + Times_hospitalized +
##     Age + Household_Income, data = df)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1817.4  -302.2  -181.5   -92.1  9678.4
##
## Coefficients:
##                              Estimate Std. Error t value Pr(>|t|)
## (Intercept)                  1.145e+03  4.811e+02   2.381  0.01786 *
## Education                   -3.546e+01  9.303e+01  -0.381  0.70331
## Own_computer                -1.058e+01  2.203e+02  -0.048  0.96172
## days_hospitalized           -1.437e+00  7.055e+00  -0.204  0.83877
## Disabled                     5.628e+02  2.540e+02   2.216  0.02740 *
## Internet_access             -3.491e+02  3.756e+02  -0.930  0.35329
## Live_with_parents            4.788e+01  2.431e+02   0.197  0.84398
## Length_of_resume_gap_month -7.593e-01  3.497e+00  -0.217  0.82824
## Annual_income_and_SWP        2.412e-03  2.384e-03   1.012  0.31233
## Unemployed                   3.088e+01  1.773e+02   0.174  0.86182
## Read_books                  -7.144e+02  2.195e+02  -3.255  0.00126 **
## Times_hospitalized           1.622e+01  1.276e+01   1.271  0.20459
```

```
## Age                          3.754e+01  7.573e+01   0.496  0.62039
## Household_Income            -1.958e-01  2.219e-01  -0.883  0.37811
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1234 on 318 degrees of freedom
##   (2 observations deleted due to missingness)
## Multiple R-squared:  0.06802,    Adjusted R-squared:  0.02992
## F-statistic: 1.785 on 13 and 318 DF,  p-value: 0.04433
```

```
# trimming features & checking for Multiple R-squared & Adjusted R-squared values to determine best sui
m3 <- lm(SWP ~ Education + days_hospitalized + Disabled + Internet_access + Live_with_parents + Length_o
summary(m3)
```

```
##
## Call:
## lm(formula = SWP ~ Education + days_hospitalized + Disabled +
##     Internet_access + Live_with_parents + Length_of_resume_gap_month +
##     Annual_income_and_SWP + Unemployed + Read_books + Times_hospitalized +
##     Age + Household_Income, data = df)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1817.8  -302.9  -179.8   -92.1  9677.8
##
## Coefficients:
##                              Estimate Std. Error t value Pr(>|t|)
## (Intercept)                 1.141e+03  4.698e+02   2.428  0.01574 *
## Education                  -3.574e+01  9.271e+01  -0.385  0.70014
## days_hospitalized          -1.400e+00  7.003e+00  -0.200  0.84166
## Disabled                    5.633e+02  2.534e+02   2.223  0.02692 *
## Internet_access            -3.525e+02  3.686e+02  -0.956  0.33963
## Live_with_parents           4.803e+01  2.427e+02   0.198  0.84321
## Length_of_resume_gap_month -7.465e-01  3.481e+00  -0.214  0.83035
## Annual_income_and_SWP       2.414e-03  2.380e-03   1.014  0.31115
## Unemployed                  3.196e+01  1.756e+02   0.182  0.85567
## Read_books                 -7.140e+02  2.190e+02  -3.260  0.00123 **
## Times_hospitalized          1.617e+01  1.270e+01   1.273  0.20386
## Age                         3.694e+01  7.455e+01   0.495  0.62060
## Household_Income           -1.956e-01  2.215e-01  -0.883  0.37780
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1232 on 319 degrees of freedom
##   (2 observations deleted due to missingness)
## Multiple R-squared:  0.06801,    Adjusted R-squared:  0.03296
## F-statistic:  1.94 on 12 and 319 DF,  p-value: 0.02931
```

```
# trimming features & checking for Multiple R-squared & Adjusted R-squared values to determine best sui
m3.1 <- lm(SWP ~ Education + days_hospitalized + Disabled + Internet_access + Live_with_parents + Length
summary(m3.1)
```

```
##
```

```
## Call:
## lm(formula = SWP ~ Education + days_hospitalized + Disabled +
##     Internet_access + Live_with_parents + Length_of_resume_gap_month +
##     Annual_income_and_SWP + Unemployed + Read_books + Age + Household_Income,
##     data = df)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1750.7  -318.3  -182.2   -90.8  9639.0
##
## Coefficients:
##                              Estimate Std. Error t value Pr(>|t|)
## (Intercept)                 1.166e+03  4.699e+02   2.482   0.0136 *
## Education                  -4.406e+01  9.257e+01  -0.476   0.6344
## days_hospitalized           4.469e+00  5.277e+00   0.847   0.3977
## Disabled                    5.399e+02  2.530e+02   2.134   0.0336 *
## Internet_access            -3.681e+02  3.687e+02  -0.998   0.3188
## Live_with_parents           7.436e+01  2.420e+02   0.307   0.7588
## Length_of_resume_gap_month -7.804e-01  3.484e+00  -0.224   0.8229
## Annual_income_and_SWP       2.599e-03  2.378e-03   1.093   0.2751
## Unemployed                  1.651e+01  1.753e+02   0.094   0.9251
## Read_books                 -7.166e+02  2.192e+02  -3.269   0.0012 **
## Age                         3.940e+01  7.460e+01   0.528   0.5978
## Household_Income           -2.097e-01  2.214e-01  -0.947   0.3444
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1233 on 320 degrees of freedom
##   (2 observations deleted due to missingness)
## Multiple R-squared:  0.06328,    Adjusted R-squared:  0.03108
## F-statistic: 1.965 on 11 and 320 DF,  p-value: 0.03126
```

```r
# trimming features & checking for Multiple R-squared & Adjusted R-squared values to determine best sui
m4 <- lm(SWP ~ Education + Disabled + Internet_access + Live_with_parents + Length_of_resume_gap_month +
summary(m4)
```

```
##
## Call:
## lm(formula = SWP ~ Education + Disabled + Internet_access + Live_with_parents +
##     Length_of_resume_gap_month + Annual_income_and_SWP + Unemployed +
##     Read_books + Times_hospitalized + Age + Household_Income,
##     data = df)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1749.2  -301.6  -178.0   -91.0  9668.5
##
## Coefficients:
##                              Estimate Std. Error t value Pr(>|t|)
## (Intercept)                 1.139e+03  4.674e+02   2.437  0.01535 *
## Education                  -3.433e+01  9.169e+01  -0.374  0.70831
## Disabled                    5.561e+02  2.463e+02   2.258  0.02463 *
## Internet_access            -3.540e+02  3.665e+02  -0.966  0.33482
## Live_with_parents           4.923e+01  2.403e+02   0.205  0.83780
```

```
## Length_of_resume_gap_month -7.466e-01  3.456e+00  -0.216  0.82912
## Annual_income_and_SWP       2.432e-03  2.361e-03   1.030  0.30358
## Unemployed                  2.899e+01  1.730e+02   0.168  0.86704
## Read_books                 -7.146e+02  2.178e+02  -3.280  0.00115 **
## Times_hospitalized          1.410e+01  8.394e+00   1.680  0.09389 .
## Age                         3.622e+01  7.378e+01   0.491  0.62382
## Household_Income           -1.952e-01  2.201e-01  -0.887  0.37562
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1226 on 322 degrees of freedom
## Multiple R-squared:  0.06873,    Adjusted R-squared:  0.03691
## F-statistic:  2.16 on 11 and 322 DF,  p-value: 0.01632
```

```
# trimming features & checking for Multiple R-squared & Adjusted R-squared values to determine best sui
m5 <- lm(SWP ~ Education + Disabled + Internet_access + Length_of_resume_gap_month + Annual_income_and_S
summary(m5)
```

```
##
## Call:
## lm(formula = SWP ~ Education + Disabled + Internet_access + Length_of_resume_gap_month +
##     Annual_income_and_SWP + Unemployed + Read_books + Times_hospitalized +
##     Age + Household_Income, data = df)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1768.4  -301.7  -178.7   -89.4  9671.4
##
## Coefficients:
##                              Estimate Std. Error t value Pr(>|t|)
## (Intercept)                 1.163e+03  4.519e+02   2.574  0.01051 *
## Education                  -3.537e+01  9.141e+01  -0.387  0.69902
## Disabled                    5.617e+02  2.444e+02   2.298  0.02221 *
## Internet_access            -3.522e+02  3.659e+02  -0.963  0.33642
## Length_of_resume_gap_month -7.899e-01  3.445e+00  -0.229  0.81877
## Annual_income_and_SWP       2.371e-03  2.338e-03   1.014  0.31126
## Unemployed                  3.259e+01  1.719e+02   0.190  0.84971
## Read_books                 -7.187e+02  2.166e+02  -3.319  0.00101 **
## Times_hospitalized          1.425e+01  8.351e+00   1.707  0.08887 .
## Age                         3.109e+01  6.931e+01   0.449  0.65398
## Household_Income           -1.946e-01  2.197e-01  -0.886  0.37653
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1224 on 323 degrees of freedom
## Multiple R-squared:  0.0686, Adjusted R-squared:  0.03977
## F-statistic: 2.379 on 10 and 323 DF,  p-value: 0.009903
```

```
# trimming features & checking for Multiple R-squared & Adjusted R-squared values to determine best sui
m6 <- lm(SWP ~ Education + Disabled + Internet_access + Annual_income_and_SWP + Unemployed + Read_books
summary(m6)
```

```
##
```

```
## Call:
## lm(formula = SWP ~ Education + Disabled + Internet_access + Annual_income_and_SWP +
##     Unemployed + Read_books + Times_hospitalized + Age + Household_Income,
##     data = df)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1764.8  -294.8  -180.7   -93.6  9671.9
##
## Coefficients:
##                        Estimate Std. Error t value Pr(>|t|)
## (Intercept)           1.147e+03  4.458e+02   2.573  0.01053 *
## Education            -3.388e+01  9.104e+01  -0.372  0.71006
## Disabled              5.560e+02  2.428e+02   2.290  0.02268 *
## Internet_access      -3.471e+02  3.646e+02  -0.952  0.34187
## Annual_income_and_SWP  2.407e-03  2.330e-03   1.033  0.30237
## Unemployed            2.452e+01  1.680e+02   0.146  0.88406
## Read_books           -7.193e+02  2.162e+02  -3.327  0.00098 ***
## Times_hospitalized    1.426e+01  8.338e+00   1.710  0.08815 .
## Age                   3.206e+01  6.908e+01   0.464  0.64286
## Household_Income     -1.894e-01  2.182e-01  -0.868  0.38618
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1222 on 324 degrees of freedom
## Multiple R-squared:  0.06845,    Adjusted R-squared:  0.04258
## F-statistic: 2.645 on 9 and 324 DF,  p-value: 0.00572
```

Comparing Multiple R-squared and Adjusted R-squared of different Linear Regression Models to select the best set of variables

```r
# creating new dataset, m_data with the features in focus gathered from model 6 of linear regression
m_data <- df %>% select(2, 5, 6, 9, 10, 11, 12, 13, 14, 15)
head(m_data)
```

```
## # A tibble: 6 x 10
##   Education Disabled Inter~1 Annua~2 Unemp~3 Read_~4   SWP Times~5   Age House~6
##       <dbl>    <dbl>   <dbl>   <dbl>   <dbl>   <dbl> <dbl>   <dbl> <dbl>   <dbl>
## 1         1        0       1   35000       1       1     0       0     2       3
## 2         4        0       1   22000       0       1     0       0     1       4
## 3         2        0       1  100000       0       1     0       0     2       8
## 4         2        0       1       0       1       1     0       0     2       3
## 5         2        1       1   32000       0       1  3000       4     2       3
## 6         1        0       1       0       0       1     0       0     2       1
## # ... with abbreviated variable names 1: Internet_access,
## #   2: Annual_income_and_SWP, 3: Unemployed, 4: Read_books,
## #   5: Times_hospitalized, 6: Household_Income
```

Performing Logistic Regression to calculate accuracy of variables being able to predict SWP

```r
# creating new dataset, log_data for logistic regression
log_data <- m_data
```

```r
# mutatting values of SWP to represent if the person receives money from SWP or not
log_data$SWP[log_data$SWP != 0] <- 1
head(log_data)
```

```
## # A tibble: 6 x 10
##   Education Disabled Inter~1 Annua~2 Unemp~3 Read_~4   SWP Times~5   Age House~6
##       <dbl>    <dbl>   <dbl>   <dbl>   <dbl>   <dbl> <dbl>   <dbl> <dbl>   <dbl>
## 1         1        0       1   35000       1       1     0       0     2       3
## 2         4        0       1   22000       0       1     0       0     1       4
## 3         2        0       1  100000       0       1     0       0     2       8
## 4         2        0       1       0       1       1     0       0     2       3
## 5         2        1       1   32000       0       1     1       4     2       3
## 6         1        0       1       0       0       1     0       0     2       1
## # ... with abbreviated variable names 1: Internet_access,
## #   2: Annual_income_and_SWP, 3: Unemployed, 4: Read_books,
## #   5: Times_hospitalized, 6: Household_Income
```

```r
library(caTools)
library(ROCR)
```

```r
set.seed(10086)

# splitting dataset with split-ratio of 80%-20%
split <- sample.split(log_data$SWP, SplitRatio = 0.8)
train_reg <- subset(log_data, split == "TRUE")
test_reg <- subset(log_data, split == "FALSE")
```

```r
# training model
logistic_model <- glm(SWP ~ Education + Disabled + Internet_access + Annual_income_and_SWP + Read_books

# summary
summary(logistic_model)
```

```
##
## Call:
## glm(formula = SWP ~ Education + Disabled + Internet_access +
##     Annual_income_and_SWP + Read_books + Times_hospitalized +
##     Household_Income, family = "binomial", data = train_reg)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.5327  -0.5044  -0.4461  -0.3722   2.5463
##
## Coefficients:
##                         Estimate Std. Error z value Pr(>|z|)
## (Intercept)           -1.069e+00  1.010e+00  -1.058   0.2901
## Education              9.069e-02  2.332e-01   0.389   0.6974
## Disabled               2.419e+00  4.622e-01   5.234 1.66e-07 ***
```

```
## Internet_access       -7.260e-01  8.632e-01  -0.841   0.4003
## Annual_income_and_SWP -8.771e-06  6.887e-06  -1.274   0.2028
## Read_books            -2.741e-01  5.672e-01  -0.483   0.6289
## Times_hospitalized     3.237e-02  1.650e-02   1.962   0.0498 *
## Household_Income      -3.805e-04  7.952e-04  -0.478   0.6323
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 235.71  on 266  degrees of freedom
## Residual deviance: 195.40  on 259  degrees of freedom
## AIC: 211.4
##
## Number of Fisher Scoring iterations: 5
```

```r
# predicting train and test data based on model
predict_reg_train <- predict(logistic_model, train_reg, type = "response")
predict_reg <- predict(logistic_model, test_reg, type = "response")
```

```r
# calculating probabilities
predict_reg_train <- ifelse(predict_reg_train >0.5, 1, 0)
predict_reg <- ifelse(predict_reg >0.5, 1, 0)
```

```r
# evaluating model accuracy using confusion matrix

table(train_reg$SWP, predict_reg_train)
```

```
##    predict_reg_train
##       0   1
##   0 214  10
##   1  25  18
```

```r
table(test_reg$SWP, predict_reg)
```

```
##    predict_reg
##      0  1
##   0 53  3
##   1  8  3
```

```r
# generating accuracy

missing_classerr_train <- mean(predict_reg_train != train_reg$SWP)
acc_train <- round(1 - missing_classerr_train, 2)
print(paste('Accuracy for Train data =', acc_train))
```

```
## [1] "Accuracy for Train data = 0.87"
```

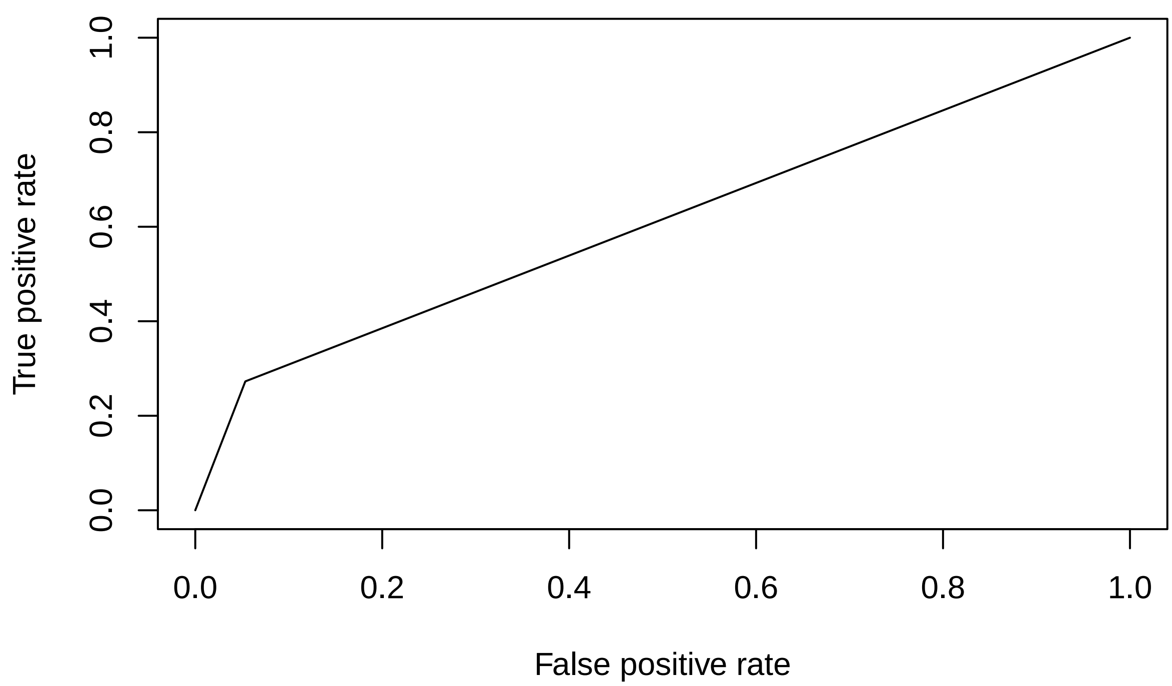
```r
missing_classerr <- mean(predict_reg != test_reg$SWP)
acc_test <- round(1 - missing_classerr, 2)
print(paste('Accuracy for Test data =', acc_test))
```

```
## [1] "Accuracy for Test data = 0.84"
```

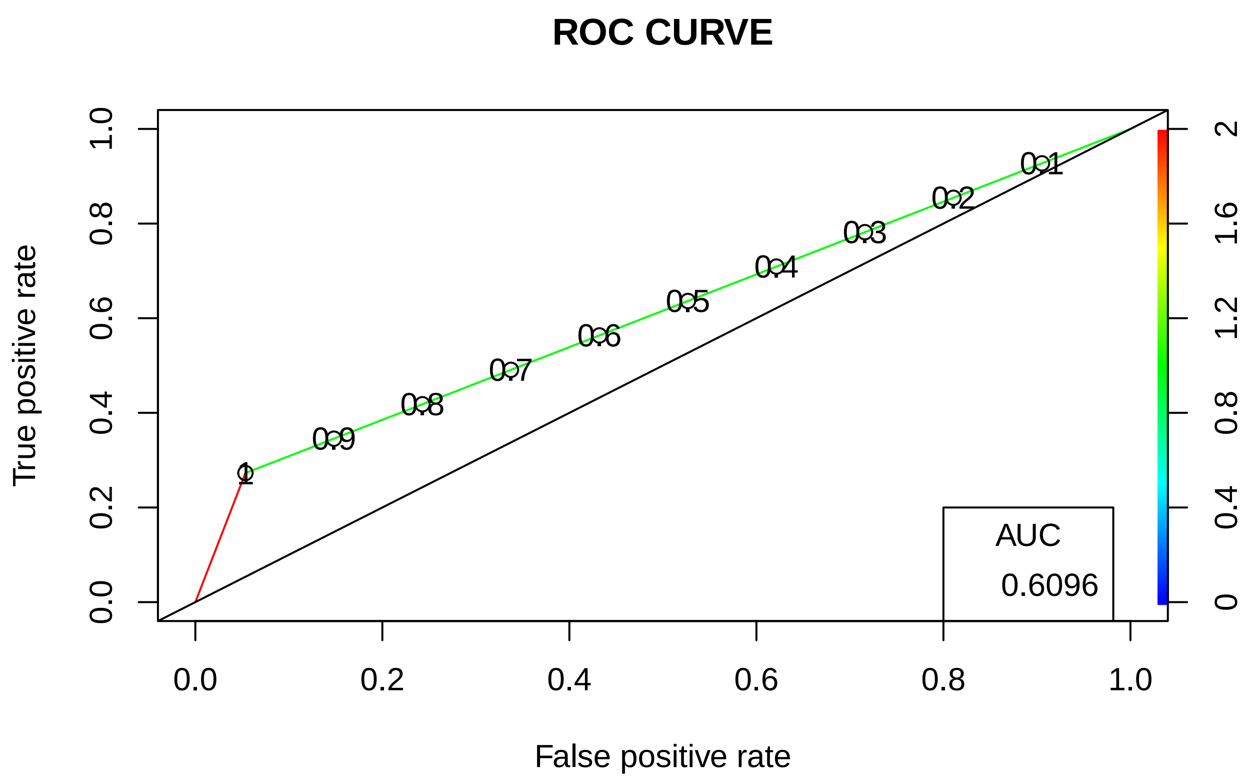
```r
set.seed(10086)
# ROC-AUC curve
ROCPred <- prediction(predict_reg, test_reg$SWP)
ROCPer <- performance(ROCPred, measure = "tpr",
                             x.measure = "fpr")

auc <- performance(ROCPred, measure = "auc")
auc <- auc@y.values[[1]]
auc
```

```
## [1] 0.6095779
```

```r
# plotting curve
plot(ROCPer)
```




```r
plot(ROCPer, colorize = TRUE,
    print.cutoffs.at = seq(0.1, by = 0.1),
    main = "ROC CURVE")
abline(a = 0, b = 1)

auc <- round(auc, 4)
legend(.8, .2, auc, title = "AUC", cex = 1)
```



Generating Decision Tree for variables selected from Linear Regression Model 6

```
library(DAAG)
library(party)
```

```
## Loading required package: grid

## Loading required package: mvtnorm

## Loading required package: modeltools

## Loading required package: stats4

## Loading required package: strucchange

## Loading required package: zoo

##
## Attaching package: 'zoo'

## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric
```

```
## Loading required package: sandwich
```

```
library(rpart)
library(rpart.plot)
library(mlbench)
library(caret)
```

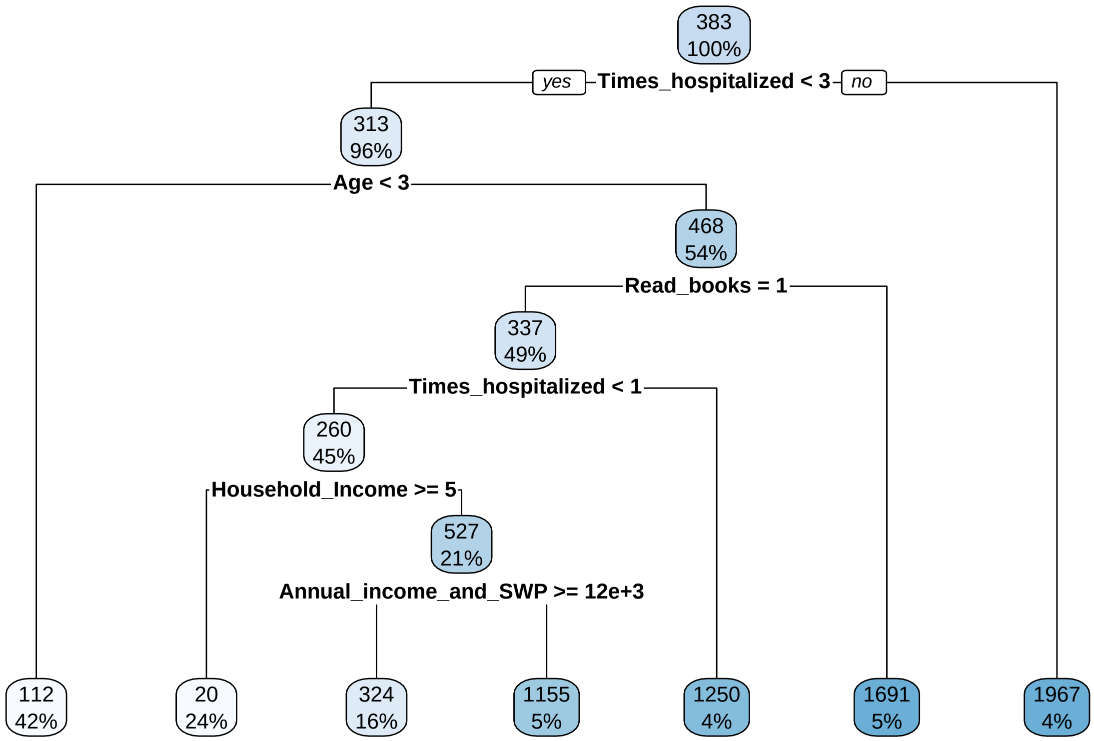
```
## Loading required package: ggplot2

## Loading required package: lattice
```

```
library(pROC)
```

```
## Type 'citation("pROC")' for a citation.

##
## Attaching package: 'pROC'

## The following objects are masked from 'package:stats':
##
##     cov, smooth, var
```

```
library(tree)
```

```
# subsetting data in ratio 80%-20% for decision tree
set.seed(1234)
ind <- sample(2, nrow(m_data), replace = T, prob = c(0.6, 0.4))
train <- m_data[ind == 1,]
test <- m_data[ind == 2,]

# generating and plotting tree
tree <- rpart(SWP ~., data = train)
rpart.plot(tree)
```



Constructing Random Forest model for predicting SWP using variables selected from Linear Regression Model 6

```
# creating prediction model and constructing its confusion matrix

library(randomForest)
```

```
## randomForest 4.7-1.1
```

```
## Type rfNews() to see new features/changes/bug fixes.
```

```
##
## Attaching package: 'randomForest'
```

```
## The following object is masked from 'package:ggplot2':
##
##     margin
```

```
## The following object is masked from 'package:dplyr':
##
##     combine
```

```
library(caret)
library(e1071)

set.seed(10086)

model <- randomForest(formula = SWP ~ ., data = m_data)
model
```

```
##
## Call:
##  randomForest(formula = SWP ~ ., data = m_data)
##                Type of random forest: regression
##                      Number of trees: 500
## No. of variables tried at each split: 3
##
##           Mean of squared residuals: 1600147
##                     % Var explained: -2.82
```

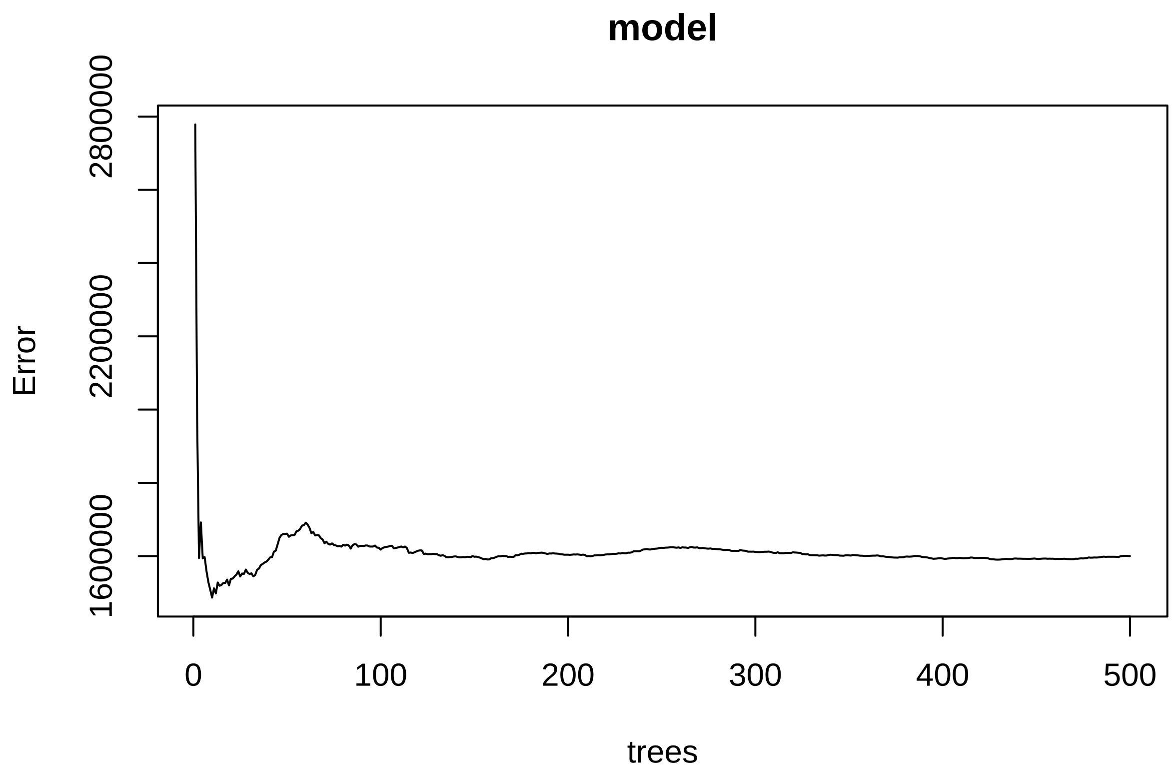
```
which.min(model$mse)
```

```
## [1] 10
```

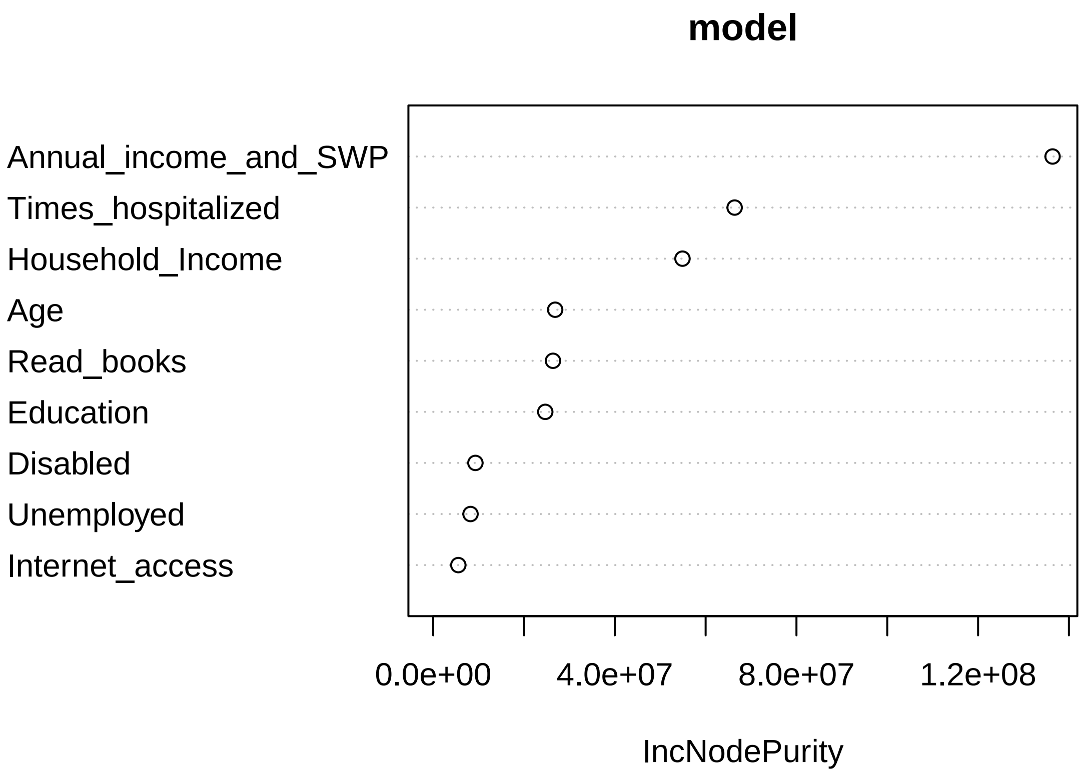
```
sqrt(model$mse[which.min(model$mse)])
```

```
## [1] 1219.242
```

```
plot(model)
```

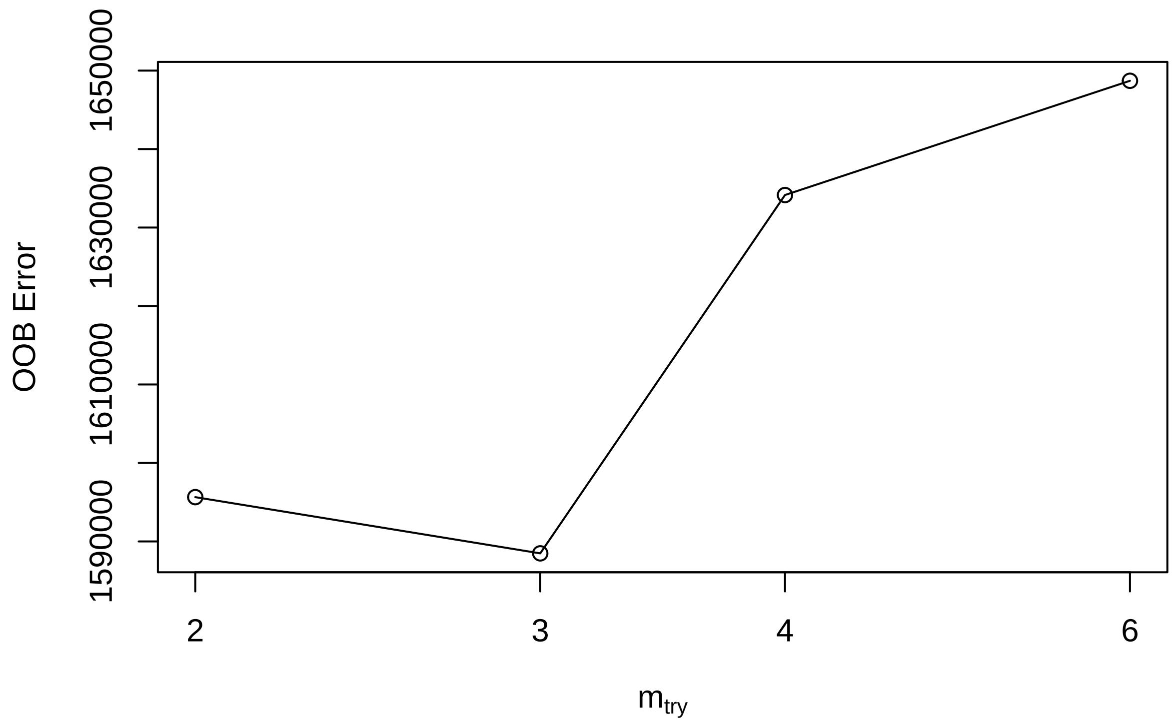



```
varImpPlot(model)
```




```
# constructing random forest
predictor_variable <- select(m_data, -SWP)
model_tuned <- tuneRF(
               x=predictor_variable, # defining predictor variables
               y=m_data$SWP, # defining response variable
               ntreeTry=500, # number of tree Random Forest will construct
               mtryStart=4,
               stepFactor=1.5,
               improve=0.01,
               trace=FALSE #don't show real-time progress
               )
```

```
## 0.02794358 0.01
## -0.004510427 0.01
## -0.03791205 0.01
```




```
# splitting data into train and test subsets
set.seed(10086)
ind <- sample(2, nrow(m_data), replace = T, prob = c(0.8, 0.2))
train <- m_data[ind == 1,]
test <- m_data[ind == 2,]

# Random Forest for train data
classifier_RF = randomForest(x = train[-7],
                             y = train$SWP,
                             ntree = 500)

classifier_RF
```

```
##
## Call:
##  randomForest(x = train[-7], y = train$SWP, ntree = 500)
##                Type of random forest: regression
##                      Number of trees: 500
## No. of variables tried at each split: 3
##
##           Mean of squared residuals: 1728089
##                     % Var explained: -4.73
```

```
y_pred = predict(classifier_RF, newdata = test[-7])
```

```r
# confusion matrix
confusion_mtx = table(test$SWP, y_pred)
confusion_mtx
```

```
##       y_pred
##        14.2125020690222 22.6046129814067 24.3775559062645 31.5854807139552
##   0                   1                1                1                1
##   100                 0                0                0                0
##   300                 0                0                0                0
##   400                 0                0                0                0
##   600                 0                0                0                0
##   1200                0                0                0                0
##   1300                0                0                0                0
##   1700                0                0                0                0
##   2000                0                0                0                0
##   2500                0                0                0                0
##   8300                0                0                0                0
##       y_pred
##        32.0216627304738 32.3645129720197 38.1387044865106 40.5523629362478
##   0                   1                1                1                1
##   100                 0                0                0                0
##   300                 0                0                0                0
##   400                 0                0                0                0
##   600                 0                0                0                0
##   1200                0                0                0                0
##   1300                0                0                0                0
##   1700                0                0                0                0
##   2000                0                0                0                0
##   2500                0                0                0                0
##   8300                0                0                0                0
##       y_pred
##        42.2497327457785 47.9543005016757 48.6221116719571 52.1365854961911
##   0                   1                1                1                1
##   100                 0                0                0                0
##   300                 0                0                0                0
##   400                 0                0                0                0
##   600                 0                0                0                0
##   1200                0                0                0                0
##   1300                0                0                0                0
##   1700                0                0                0                0
##   2000                0                0                0                0
##   2500                0                0                0                0
##   8300                0                0                0                0
##       y_pred
##        57.1898308271863 61.5853549993989 62.31656288288 63.0013327616998
##   0                   1                0              1                1
##   100                 0                0              0                0
##   300                 0                0              0                0
##   400                 0                0              0                0
##   600                 0                0              0                0
##   1200                0                0              0                0
##   1300                0                0              0                0
##   1700                0                0              0                0
```

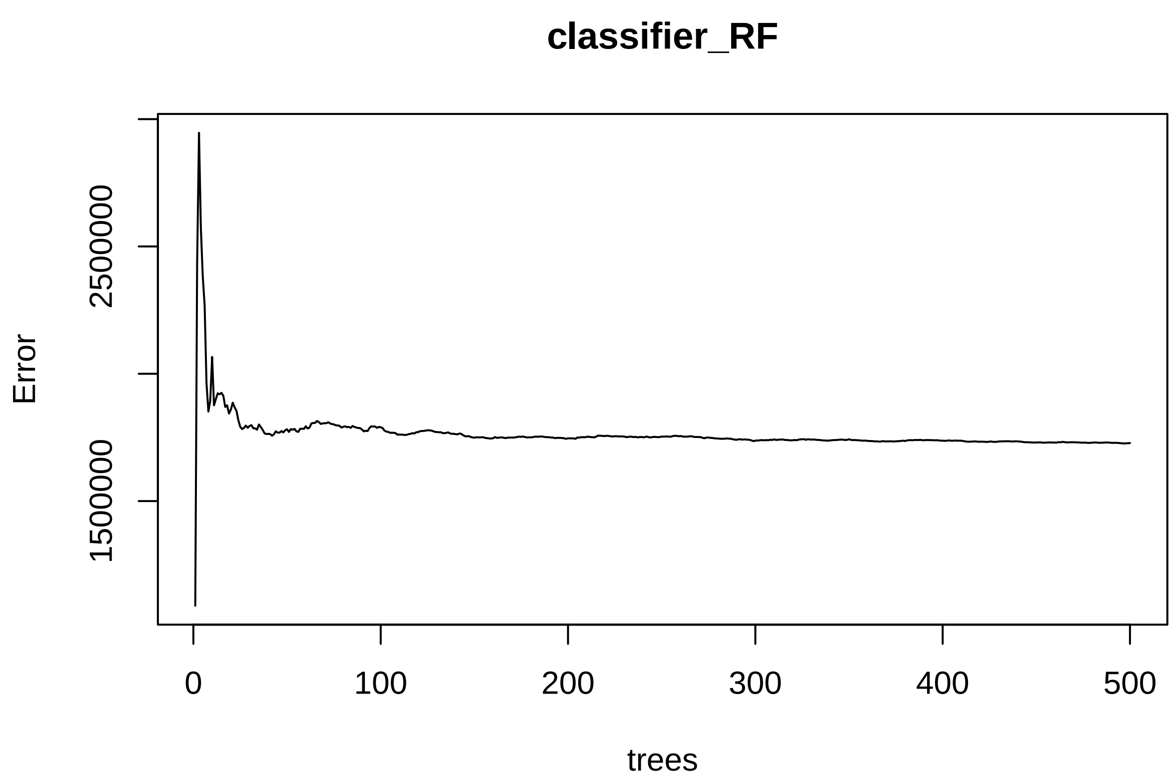
```
##   2000                  0                 0               0                 0
##   2500                  0                 1               0                 0
##   8300                  0                 0               0                 0
##       y_pred
##        65.4887652047182 65.6102268254917 67.5063658142683 70.3787044865106
##   0                   1                0                1                1
##   100                 0                1                0                0
##   300                 0                0                0                0
##   400                 0                0                0                0
##   600                 0                0                0                0
##   1200                0                0                0                0
##   1300                0                0                0                0
##   1700                0                0                0                0
##   2000                0                0                0                0
##   2500                0                0                0                0
##   8300                0                0                0                0
##       y_pred
##        76.2918212365827 82.7266728180279 93.0606451881454 105.950675127235
##   0                   1                1                1                1
##   100                 0                0                0                0
##   300                 0                0                0                0
##   400                 0                0                0                0
##   600                 0                0                0                0
##   1200                0                0                0                0
##   1300                0                0                0                0
##   1700                0                0                0                0
##   2000                0                0                0                0
##   2500                0                0                0                0
##   8300                0                0                0                0
##       y_pred
##        112.010571474747 112.202522353786 113.040068495693 113.690400147393
##   0                   1                1                1                1
##   100                 0                0                0                0
##   300                 0                0                0                0
##   400                 0                0                0                0
##   600                 0                0                0                0
##   1200                0                0                0                0
##   1300                0                0                0                0
##   1700                0                0                0                0
##   2000                0                0                0                0
##   2500                0                0                0                0
##   8300                0                0                0                0
##       y_pred
##        113.886134293551 117.328838372291 120.997042936985 124.062585454044
##   0                   1                1                0                0
##   100                 0                0                0                1
##   300                 0                0                0                0
##   400                 0                0                0                0
##   600                 0                0                0                0
##   1200                0                0                0                0
##   1300                0                0                0                0
##   1700                0                0                1                0
##   2000                0                0                0                0
##   2500                0                0                0                0
```

```
##   8300                0                0                0                0
##       y_pred
##        126.622628378169 134.507645895972 137.715804603579 144.965070731064
##   0                   1                1                1                1
##   100                 0                0                0                0
##   300                 0                0                0                0
##   400                 0                0                0                0
##   600                 0                0                0                0
##   1200                0                0                0                0
##   1300                0                0                0                0
##   1700                0                0                0                0
##   2000                0                0                0                0
##   2500                0                0                0                0
##   8300                0                0                0                0
##       y_pred
##        153.969673386332 185.109350180127 211.0895608832 212.240568527613
##   0                   1                1              1                1
##   100                 0                0              0                0
##   300                 0                0              0                0
##   400                 0                0              0                0
##   600                 0                0              0                0
##   1200                0                0              0                0
##   1300                0                0              0                0
##   1700                0                0              0                0
##   2000                0                0              0                0
##   2500                0                0              0                0
##   8300                0                0              0                0
##       y_pred
##        217.802260172226 228.66123886219 230.927050618509 231.975754699472
##   0                   1               1                1                1
##   100                 0               0                0                0
##   300                 0               0                0                0
##   400                 0               0                0                0
##   600                 0               0                0                0
##   1200                0               0                0                0
##   1300                0               0                0                0
##   1700                0               0                0                0
##   2000                0               0                0                0
##   2500                0               0                0                0
##   8300                0               0                0                0
##       y_pred
##        244.185988455988 277.5510104366 279.129869558541 369.694117267248
##   0                   0              1                1                0
##   100                 0              0                0                1
##   300                 0              0                0                0
##   400                 0              0                0                0
##   600                 1              0                0                0
##   1200                0              0                0                0
##   1300                0              0                0                0
##   1700                0              0                0                0
##   2000                0              0                0                0
##   2500                0              0                0                0
##   8300                0              0                0                0
##       y_pred
```

```
##        392.21924390689 470.431999247788 580.189027422342 584.891746031746
##   0                  1                1                1                1
##   100                0                0                0                0
##   300                0                0                0                0
##   400                0                0                0                0
##   600                0                0                0                0
##   1200               0                0                0                0
##   1300               0                0                0                0
##   1700               0                0                0                0
##   2000               0                0                0                0
##   2500               0                0                0                0
##   8300               0                0                0                0
##       y_pred
##        643.079711033088 655.190844155844 681.854124029045 721.452839856222
##   0                   1                0                0                0
##   100                 0                0                1                0
##   300                 0                0                0                0
##   400                 0                1                0                0
##   600                 0                0                0                0
##   1200                0                0                0                0
##   1300                0                0                0                0
##   1700                0                0                0                0
##   2000                0                0                0                0
##   2500                0                0                0                0
##   8300                0                0                0                1
##       y_pred
##        754.989323251823 781.714167809315 781.840868972052 807.571116168675
##   0                   1                1                1                0
##   100                 0                0                0                1
##   300                 0                0                0                0
##   400                 0                0                0                0
##   600                 0                0                0                0
##   1200                0                0                0                0
##   1300                0                0                0                0
##   1700                0                0                0                0
##   2000                0                0                0                0
##   2500                0                0                0                0
##   8300                0                0                0                0
##       y_pred
##        835.855145731885 893.925433506219 967.833866966367 984.81652904599
##   0                   0                1                0               1
##   100                 0                0                0               0
##   300                 0                0                1               0
##   400                 0                0                0               0
##   600                 0                0                0               0
##   1200                1                0                0               0
##   1300                0                0                0               0
##   1700                0                0                0               0
##   2000                0                0                0               0
##   2500                0                0                0               0
##   8300                0                0                0               0
##       y_pred
##        1050.16904761905 1247.88627705628 1586.64466617556 2022.4190446705
##   0                   0                1                1               0
```

```
##    100                     0                0                0               0
##    300                     0                0                0               0
##    400                     0                0                0               0
##    600                     0                0                0               0
##    1200                    0                0                0               0
##    1300                    1                0                0               0
##    1700                    0                0                0               0
##    2000                    0                0                0               1
##    2500                    0                0                0               0
##    8300                    0                0                0               0
```

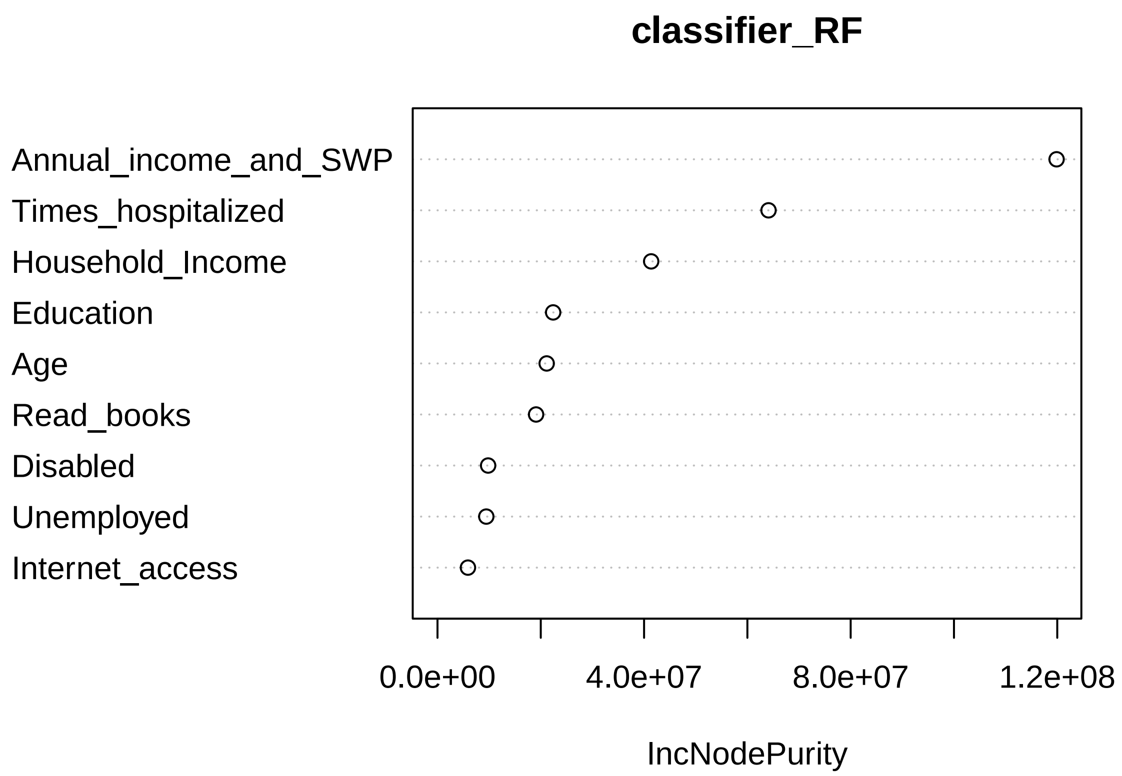
```
plot(classifier_RF)
```




```
# importance plot
importance(classifier_RF)
```

```
##                        IncNodePurity
## Education                   22393598
## Disabled                     9805584
## Internet_access              5897360
## Annual_income_and_SWP      119870244
## Unemployed                   9449113
## Read_books                  19091613
## Times_hospitalized          64098450
## Age                         21144514
## Household_Income            41374446
```

```r
# cariable importance plot
varImpPlot(classifier_RF)
```




```r
# creating new data to predict SWP value
new <- data.frame(Education=2, Disabled=0, Internet_access=1, Annual_income_and_SWP=58000, Unemployed=0

new1 <- data.frame(Education=4, Disabled=1, Internet_access=1, Annual_income_and_SWP=58000, Unemployed=

new2 <- data.frame(Education=2, Disabled=0, Internet_access=1, Annual_income_and_SWP=100000, Unemployed

new3 <- data.frame(Education=2, Disabled=1, Internet_access=0, Annual_income_and_SWP=1000, Unemployed=1

new4 <- data.frame(Education=4, Disabled=0, Internet_access=1, Annual_income_and_SWP=100000, Unemployed


predict(model, newdata=new)
```

```
##        1
## 135.1583
```

```r
predict(model, newdata=new1)
```

```
##        1
## 1111.902
```

```r
predict(model, newdata=new2)
```

```
##        1
## 78.32838
```

```r
predict(model, newdata=new3)
```

```
##        1
## 2398.222
```

```r
predict(model, newdata=new4)
```

```
##        1
## 176.4714
```