Introduction to Statistical Learning – Final Project

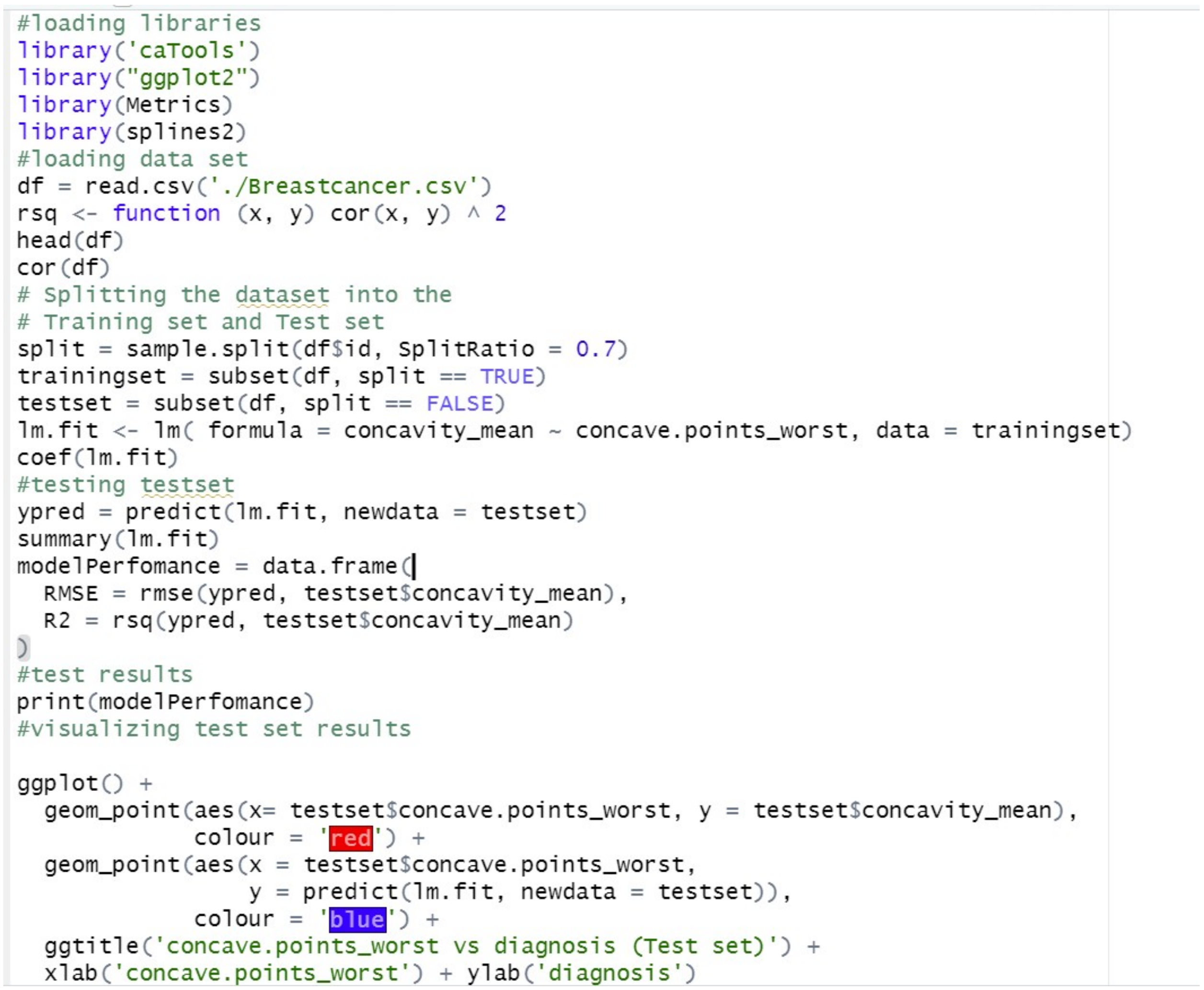
Radha Gude

16337132

Part 1 :- Regression

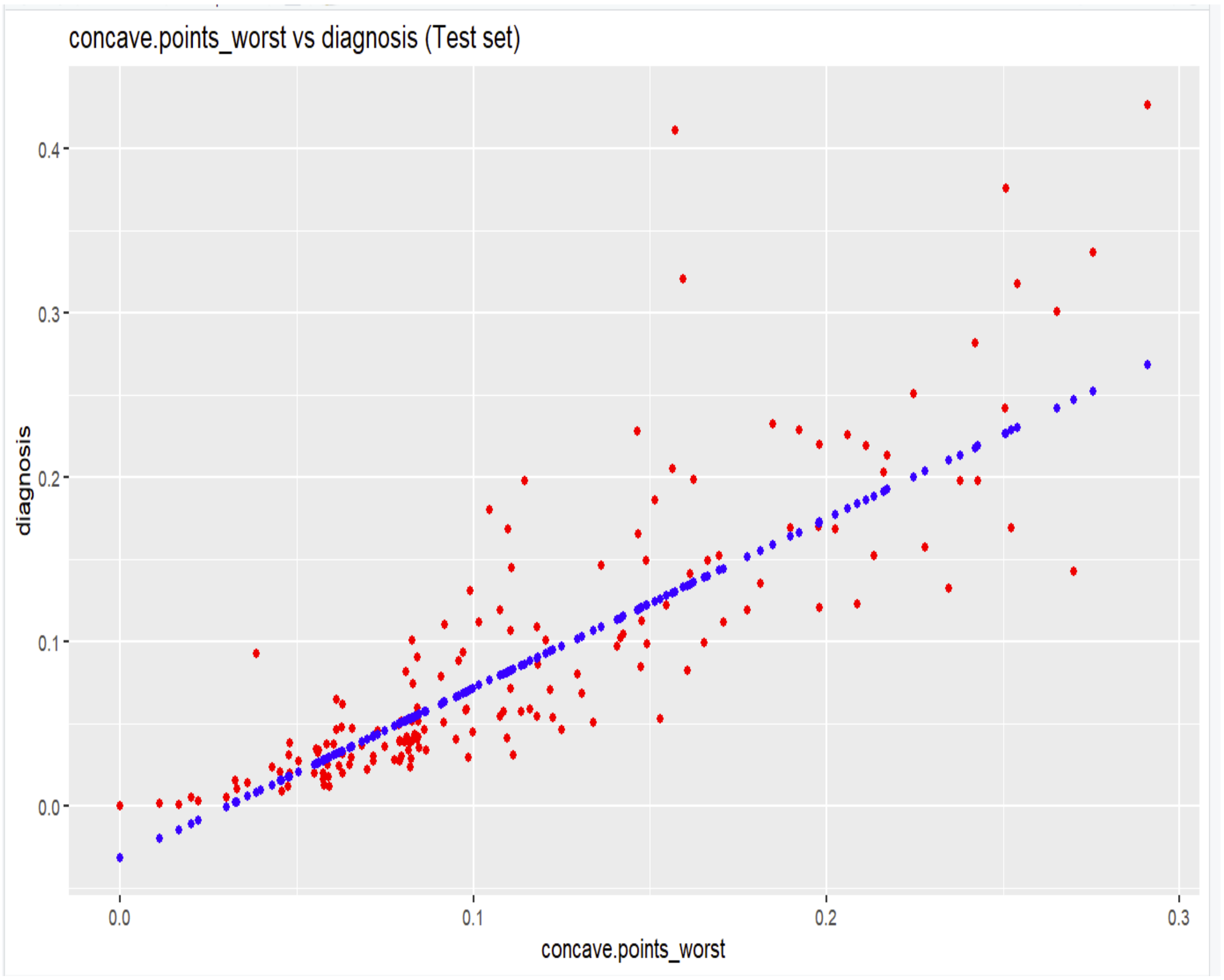
a) Linear Regression :

Code and Explanation :

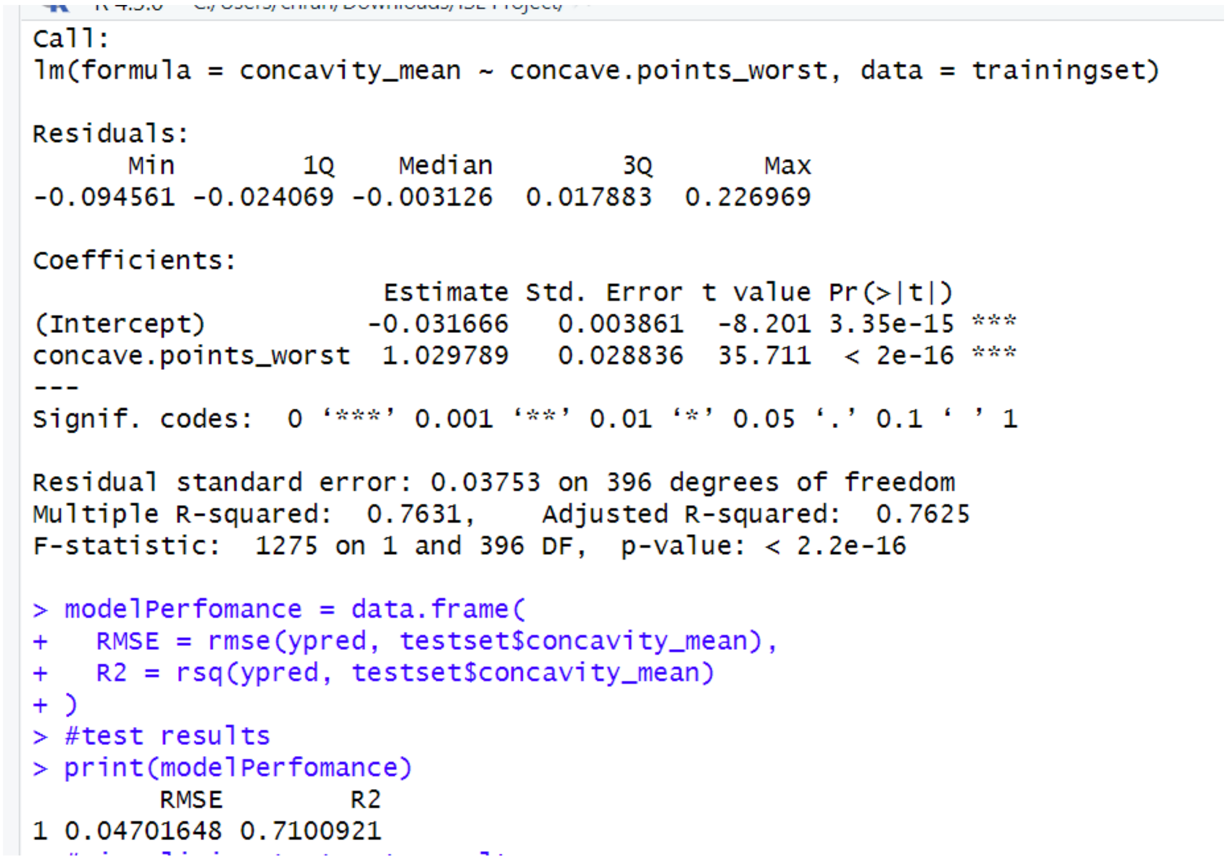


```
#loading libraries
library('caTools')
library("ggplot2")
library(Metrics)
library(splines2)
#loading data set
df = read.csv('./Breastcancer.csv')
rsq <- function (x, y) cor(x, y) ^ 2
head(df)
cor(df)
# Splitting the dataset into the
# Training set and Test set
split = sample.split(df$id, SplitRatio = 0.7)
trainingset = subset(df, split == TRUE)
testset = subset(df, split == FALSE)
lm.fit <- lm( formula = concavity_mean ~ concave.points_worst, data = trainingset)
coef(lm.fit)
#testing testset
ypred = predict(lm.fit, newdata = testset)
summary(lm.fit)
modelPerfomance = data.frame(
  RMSE = rmse(ypred, testset$concavity_mean),
  R2 = rsq(ypred, testset$concavity_mean)
)
#test results
print(modelPerfomance)
#visualizing test set results

ggplot() +
  geom_point(aes(x= testset$concave.points_worst, y = testset$concavity_mean),
             colour = 'red') +
  geom_point(aes(x = testset$concave.points_worst,
                 y = predict(lm.fit, newdata = testset)),
             colour = 'blue') +
  ggtitle('concave.points_worst vs diagnosis (Test set)') +
  xlab('concave.points_worst') + ylab('diagnosis')
```

Visualizing test results :



Summary :



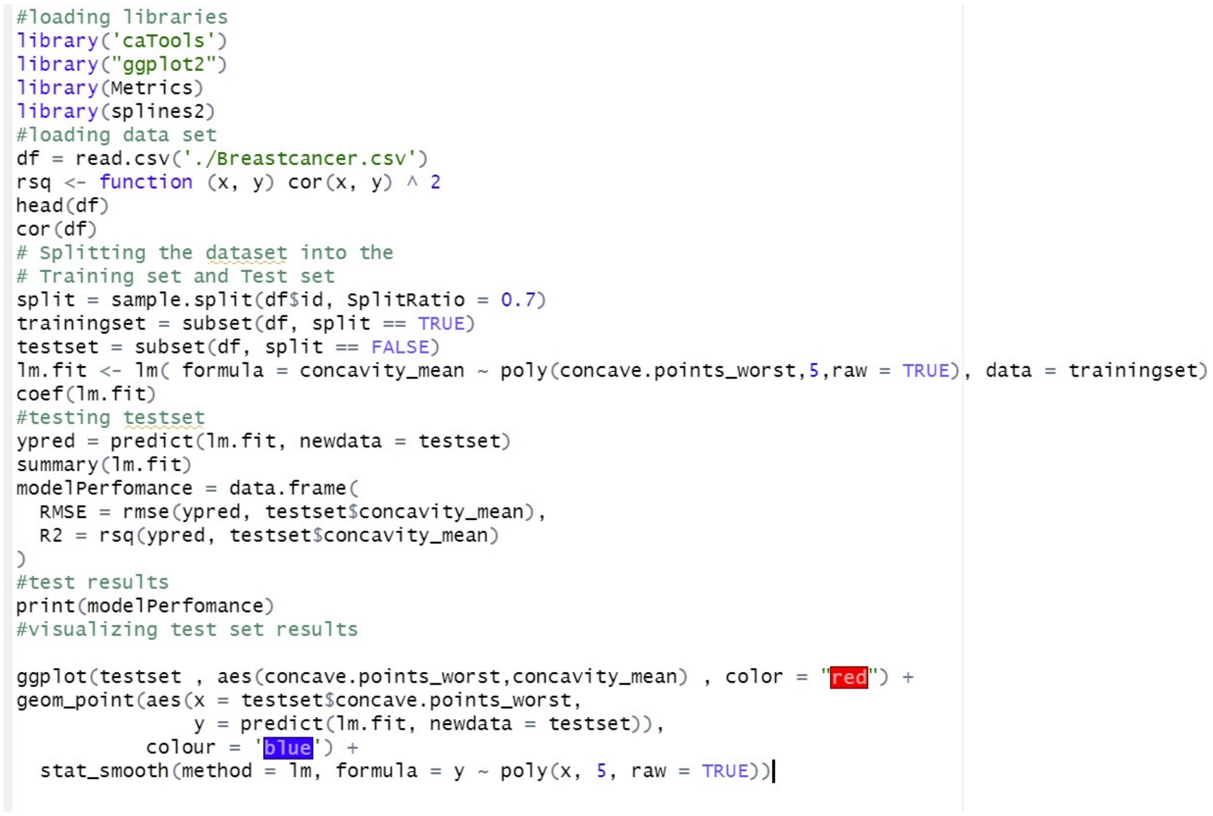
```
Call:
lm(formula = concavity_mean ~ concave.points_worst, data = trainingset)

Residuals:
      Min        1Q    Median        3Q       Max 
-0.094561 -0.024069 -0.003126  0.017883  0.226969 

Coefficients:
                      Estimate Std. Error t value Pr(>|t|)    
(Intercept)          -0.031666   0.003861  -8.201 3.35e-15 ***
concave.points_worst  1.029789   0.028836  35.711  < 2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.03753 on 396 degrees of freedom
Multiple R-squared:  0.7631,	Adjusted R-squared:  0.7625 
F-statistic:  1275 on 1 and 396 DF,  p-value: < 2.2e-16

> modelPerfomance = data.frame(
+   RMSE = rmse(ypred, testset$concavity_mean),
+   R2 = rsq(ypred, testset$concavity_mean)
+ )
> #test results
> print(modelPerfomance)
        RMSE        R2
1 0.04701648 0.7100921
```

Observations :

Correlation between concavity_mean and concave.points_worst is nearly 0.67. So concavity_mean can be linearly dependent on concave.points_worst.

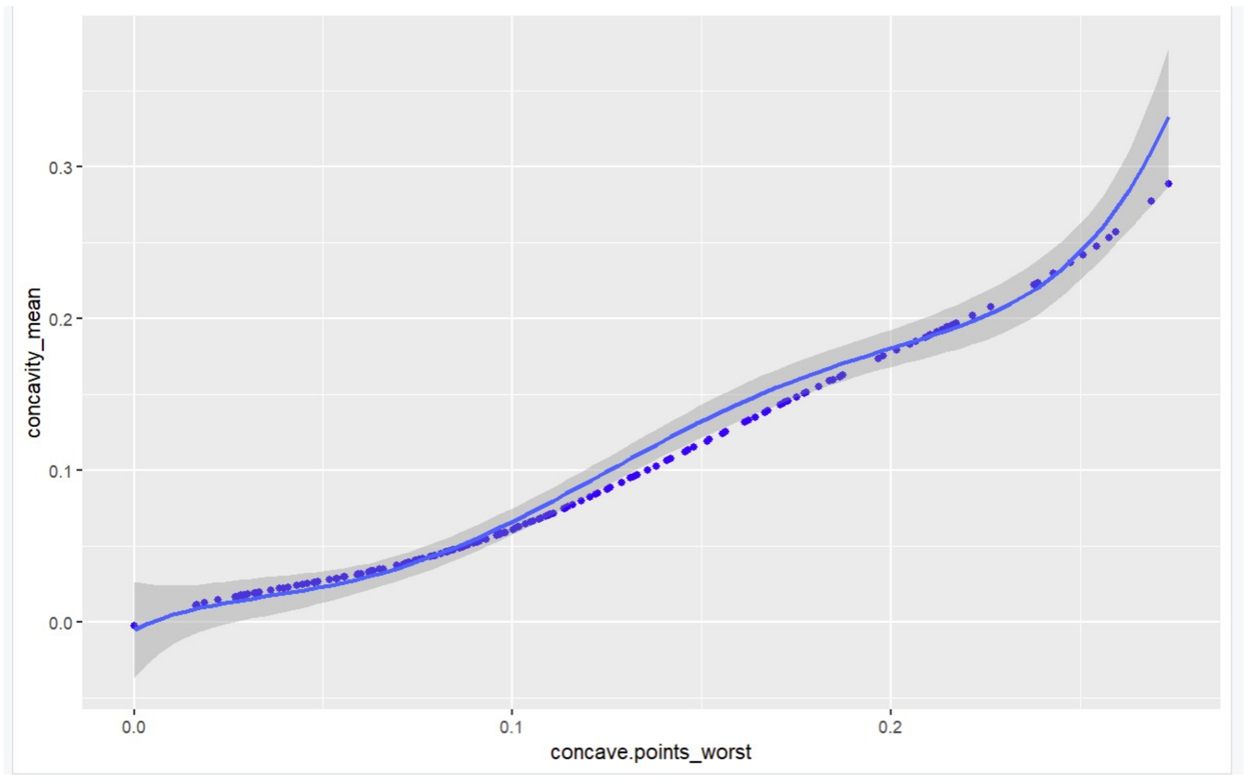
a) Polynomial Regression :

Code and Explanation :

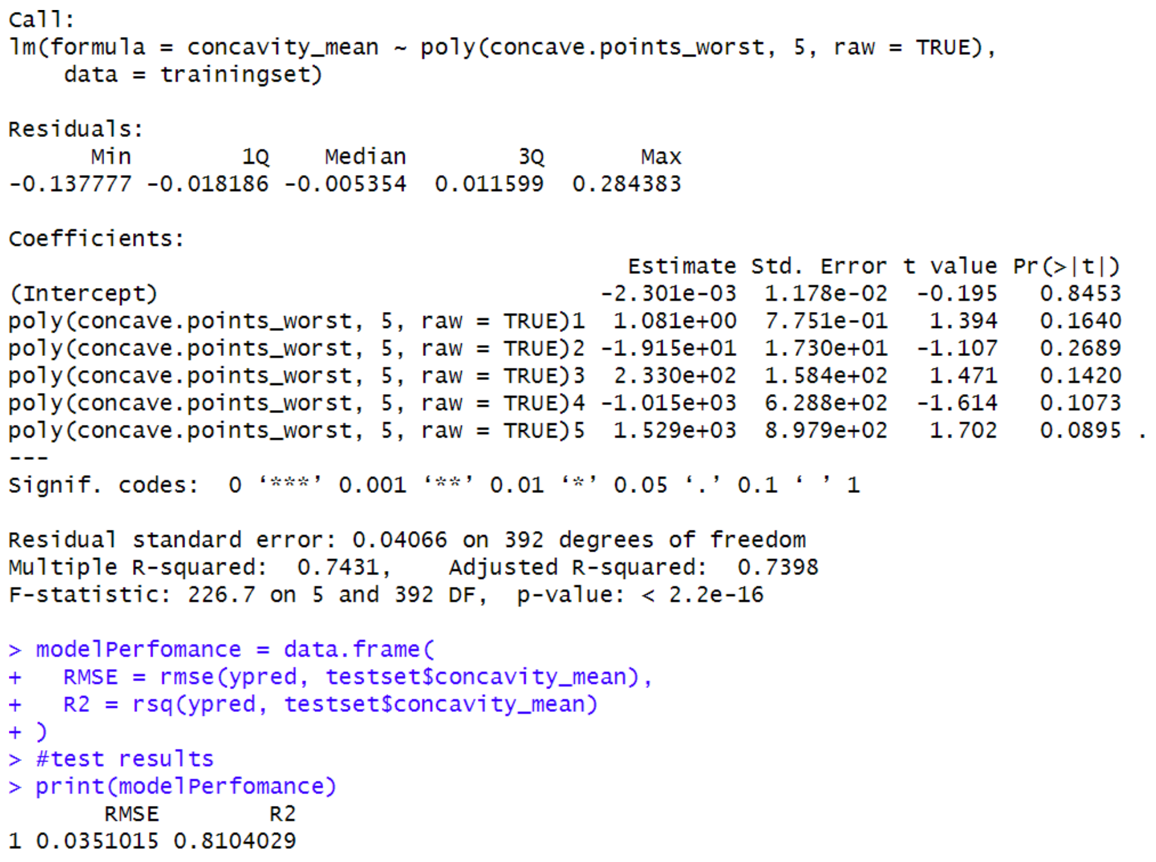


```
#loading libraries
library('caTools')
library("ggplot2")
library(Metrics)
library(splines2)
#loading data set
df = read.csv('./Breastcancer.csv')
rsq <- function (x, y) cor(x, y) ^ 2
head(df)
cor(df)
# Splitting the dataset into the
# Training set and Test set
split = sample.split(df$id, SplitRatio = 0.7)
trainingset = subset(df, split == TRUE)
testset = subset(df, split == FALSE)
lm.fit <- lm( formula = concavity_mean ~ poly(concave.points_worst,5,raw = TRUE), data = trainingset)
coef(lm.fit)
#testing testset
ypred = predict(lm.fit, newdata = testset)
summary(lm.fit)
modelPerfomance = data.frame(
  RMSE = rmse(ypred, testset$concavity_mean),
  R2 = rsq(ypred, testset$concavity_mean)
)
#test results
print(modelPerfomance)
#visualizing test set results

ggplot(testset , aes(concave.points_worst,concavity_mean) , color = "red") +
geom_point(aes(x = testset$concave.points_worst,
               y = predict(lm.fit, newdata = testset)),
           colour = 'blue') +
  stat_smooth(method = lm, formula = y ~ poly(x, 5, raw = TRUE))
```

Visualizing test results :



Summary :

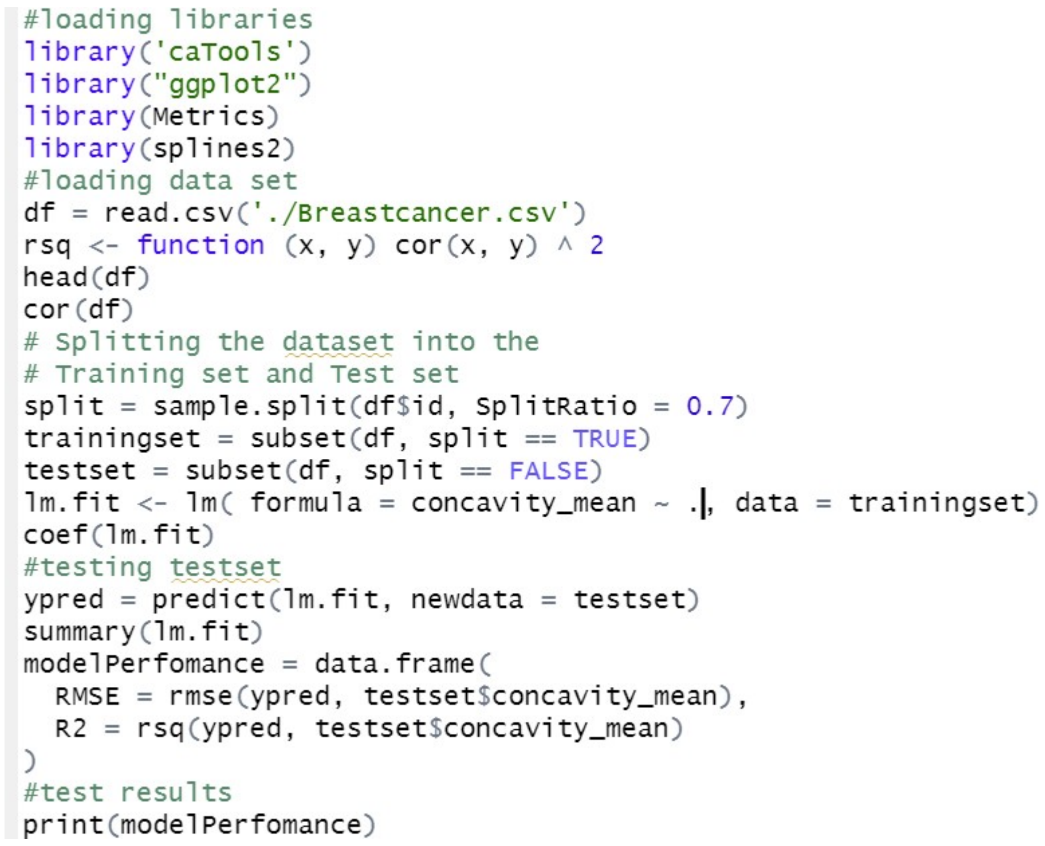


```
Call:
lm(formula = concavity_mean ~ poly(concave.points_worst, 5, raw = TRUE),
    data = trainingset)

Residuals:
      Min        1Q    Median        3Q       Max
-0.137777 -0.018186 -0.005354  0.011599  0.284383

Coefficients:
                                             Estimate Std. Error t value Pr(>|t|)
(Intercept)                                -2.301e-03  1.178e-02  -0.195   0.8453
poly(concave.points_worst, 5, raw = TRUE)1  1.081e+00  7.751e-01   1.394   0.1640
poly(concave.points_worst, 5, raw = TRUE)2 -1.915e+01  1.730e+01  -1.107   0.2689
poly(concave.points_worst, 5, raw = TRUE)3  2.330e+02  1.584e+02   1.471   0.1420
poly(concave.points_worst, 5, raw = TRUE)4 -1.015e+03  6.288e+02  -1.614   0.1073
poly(concave.points_worst, 5, raw = TRUE)5  1.529e+03  8.979e+02   1.702   0.0895 .
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.04066 on 392 degrees of freedom
Multiple R-squared:  0.7431,    Adjusted R-squared:  0.7398
F-statistic: 226.7 on 5 and 392 DF,  p-value: < 2.2e-16

> modelPerfomance = data.frame(
+   RMSE = rmse(ypred, testset$concavity_mean),
+   R2 = rsq(ypred, testset$concavity_mean)
+ )
> #test results
> print(modelPerfomance)
       RMSE        R2
1 0.0351015 0.8104029
```

Explanation :

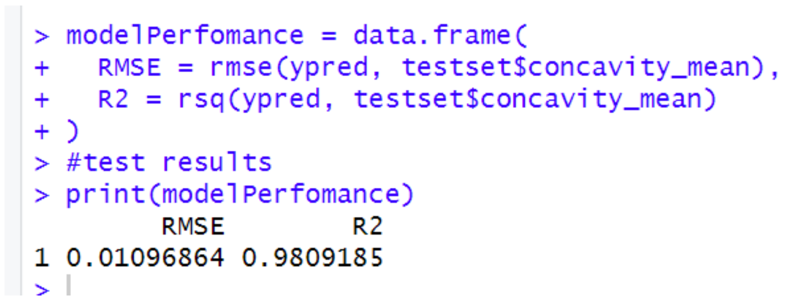
Predicting concavity_mean with concave.points_worst using polynomial regression of order 5 yields results with R2 = 0.76959 which is really good.

C) Multi Linear Regression :
Code and Explanation:



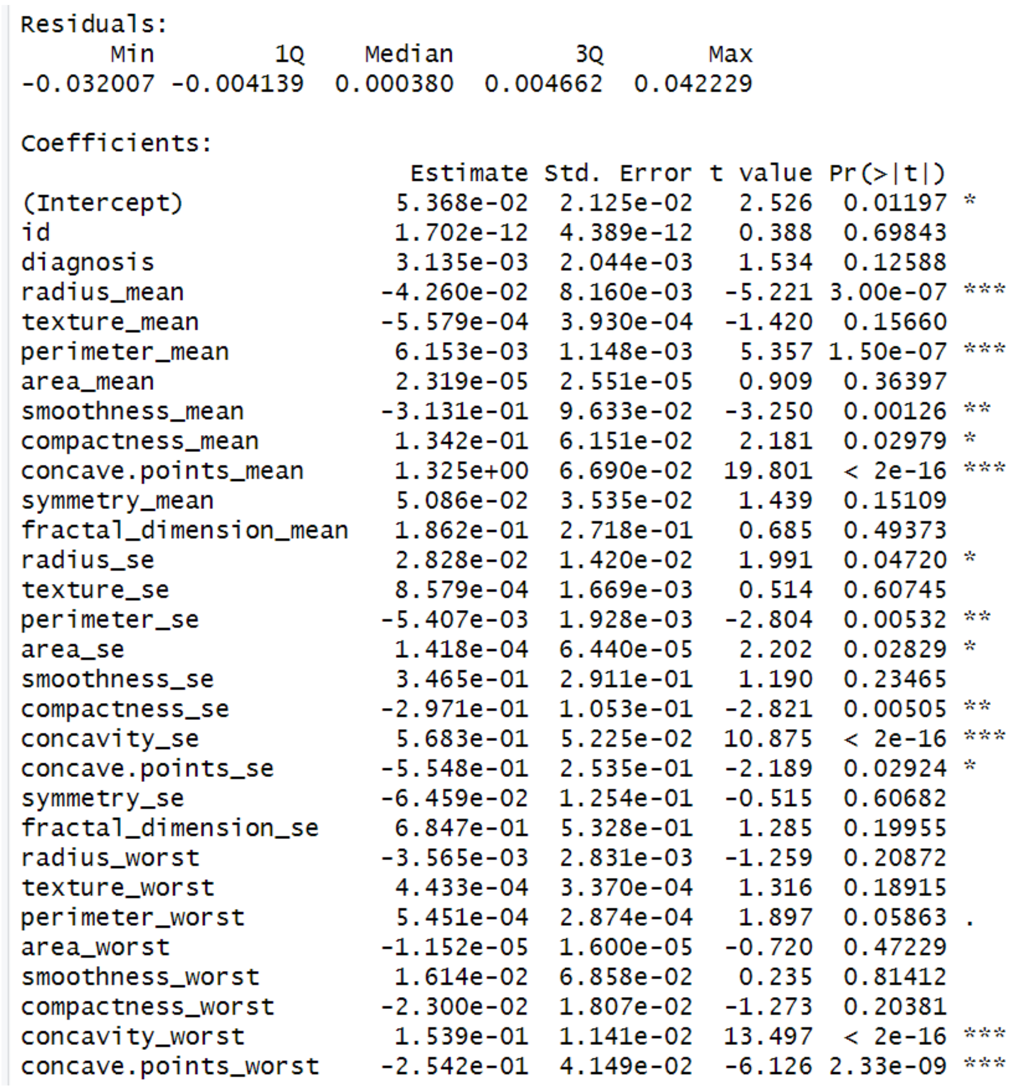
```r
#loading libraries
library('caTools')
library("ggplot2")
library(Metrics)
library(splines2)
#loading data set
df = read.csv('./Breastcancer.csv')
rsq <- function (x, y) cor(x, y) ^ 2
head(df)
cor(df)
# Splitting the dataset into the
# Training set and Test set
split = sample.split(df$id, SplitRatio = 0.7)
trainingset = subset(df, split == TRUE)
testset = subset(df, split == FALSE)
lm.fit <- lm( formula = concavity_mean ~ ., data = trainingset)
coef(lm.fit)
#testing testset
ypred = predict(lm.fit, newdata = testset)
summary(lm.fit)
modelPerfomance = data.frame(
  RMSE = rmse(ypred, testset$concavity_mean),
  R2 = rsq(ypred, testset$concavity_mean)
)
#test results
print(modelPerfomance)
```

Test Results :

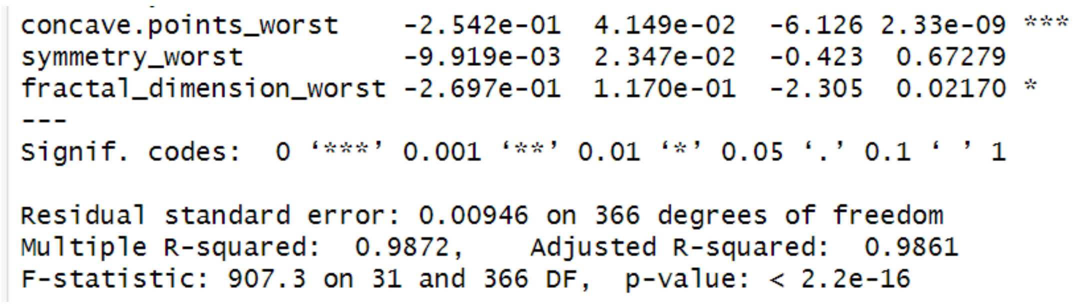


```
> modelPerfomance = data.frame(
+   RMSE = rmse(ypred, testset$concavity_mean),
+   R2 = rsq(ypred, testset$concavity_mean)
+ )
> #test results
> print(modelPerfomance)
        RMSE        R2
1 0.01096864 0.9809185
> 
```

Summary :



```
Residuals:
      Min        1Q    Median        3Q       Max 
-0.032007 -0.004139  0.000380  0.004662  0.042229 

Coefficients:
                          Estimate Std. Error t value Pr(>|t|)    
(Intercept)              5.368e-02  2.125e-02   2.526  0.01197 *  
id                       1.702e-12  4.389e-12   0.388  0.69843    
diagnosis                3.135e-03  2.044e-03   1.534  0.12588    
radius_mean             -4.260e-02  8.160e-03  -5.221 3.00e-07 ***
texture_mean            -5.579e-04  3.930e-04  -1.420  0.15660    
perimeter_mean           6.153e-03  1.148e-03   5.357 1.50e-07 ***
area_mean                2.319e-05  2.551e-05   0.909  0.36397    
smoothness_mean         -3.131e-01  9.633e-02  -3.250  0.00126 ** 
compactness_mean         1.342e-01  6.151e-02   2.181  0.02979 *  
concave.points_mean      1.325e+00  6.690e-02  19.801  < 2e-16 ***
symmetry_mean            5.086e-02  3.535e-02   1.439  0.15109    
fractal_dimension_mean   1.862e-01  2.718e-01   0.685  0.49373    
radius_se                2.828e-02  1.420e-02   1.991  0.04720 *  
texture_se               8.579e-04  1.669e-03   0.514  0.60745    
perimeter_se            -5.407e-03  1.928e-03  -2.804  0.00532 ** 
area_se                  1.418e-04  6.440e-05   2.202  0.02829 *  
smoothness_se            3.465e-01  2.911e-01   1.190  0.23465    
compactness_se          -2.971e-01  1.053e-01  -2.821  0.00505 ** 
concavity_se             5.683e-01  5.225e-02  10.875  < 2e-16 ***
concave.points_se       -5.548e-01  2.535e-01  -2.189  0.02924 *  
symmetry_se             -6.459e-02  1.254e-01  -0.515  0.60682    
fractal_dimension_se     6.847e-01  5.328e-01   1.285  0.19955    
radius_worst            -3.565e-03  2.831e-03  -1.259  0.20872    
texture_worst            4.433e-04  3.370e-04   1.316  0.18915    
perimeter_worst          5.451e-04  2.874e-04   1.897  0.05863 .  
area_worst              -1.152e-05  1.600e-05  -0.720  0.47229    
smoothness_worst         1.614e-02  6.858e-02   0.235  0.81412    
compactness_worst       -2.300e-02  1.807e-02  -1.273  0.20381    
concavity_worst          1.539e-01  1.141e-02  13.497  < 2e-16 ***
concave.points_worst    -2.542e-01  4.149e-02  -6.126 2.33e-09 ***
```

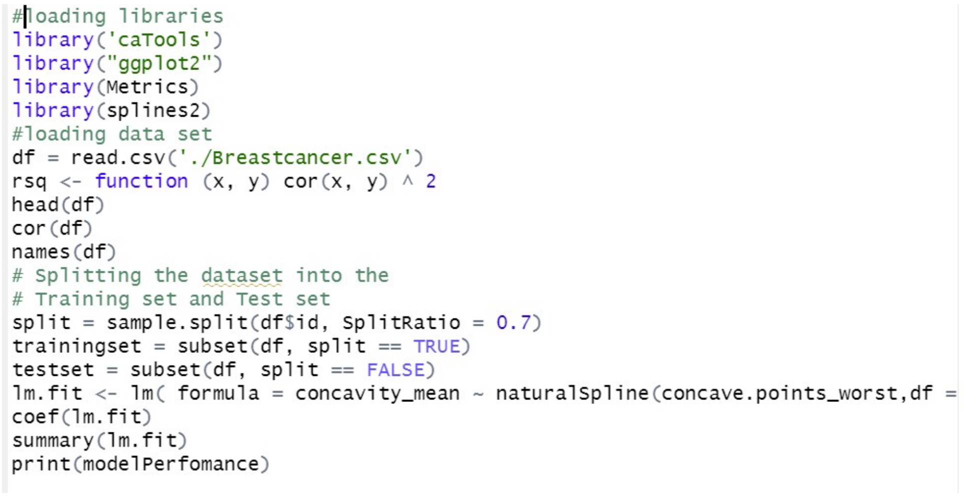


concave.points_worst   -2.542e-01 4.149e-02 -6.126 2.33e-09 ***

```
symmetry_worst          -9.919e-03  2.347e-02  -0.423  0.67279    
fractal_dimension_worst -2.697e-01  1.170e-01  -2.305  0.02170 *  
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.00946 on 366 degrees of freedom
Multiple R-squared:  0.9872,	Adjusted R-squared:  0.9861 
F-statistic: 907.3 on 31 and 366 DF,  p-value: < 2.2e-16
```

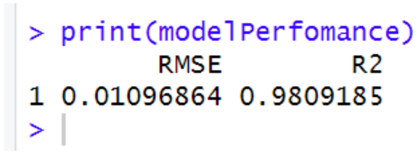
D) Natural Cubic Spline:

Code and Explanation :



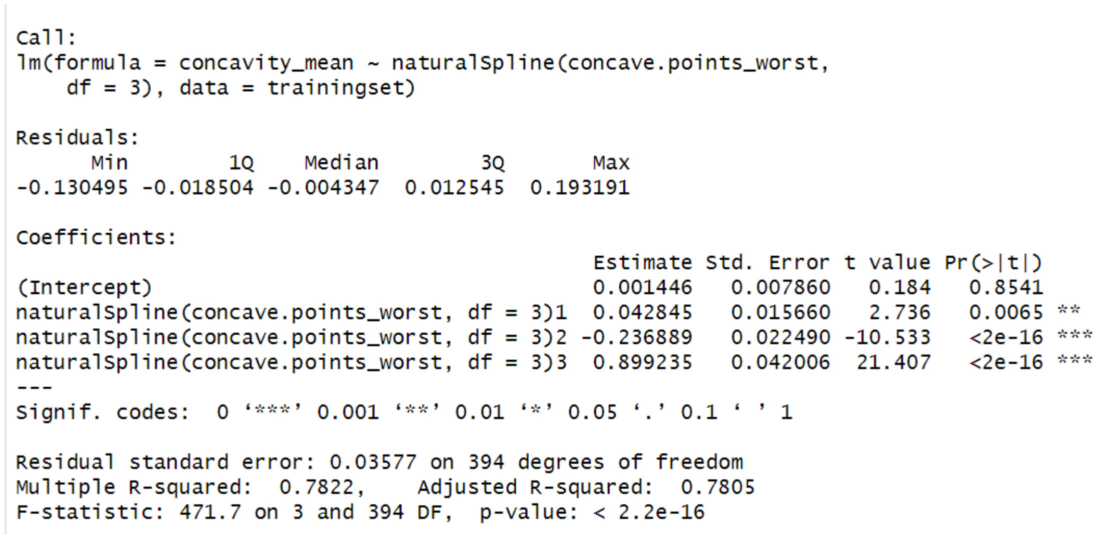
```
#loading libraries
library('caTools')
library("ggplot2")
library(Metrics)
library(splines2)
#loading data set
df = read.csv('./Breastcancer.csv')
rsq <- function (x, y) cor(x, y) ^ 2
head(df)
cor(df)
names(df)
# Splitting the dataset into the
# Training set and Test set
split = sample.split(df$id, SplitRatio = 0.7)
trainingset = subset(df, split == TRUE)
testset = subset(df, split == FALSE)
lm.fit <- lm( formula = concavity_mean ~ naturalSpline(concave.points_worst,df = 3), data = trainingset)
coef(lm.fit)
summary(lm.fit)
print(modelPerfomance)
```

Test Results :



```
> print(modelPerfomance)
        RMSE        R2
1 0.01096864 0.9809185
> 
```

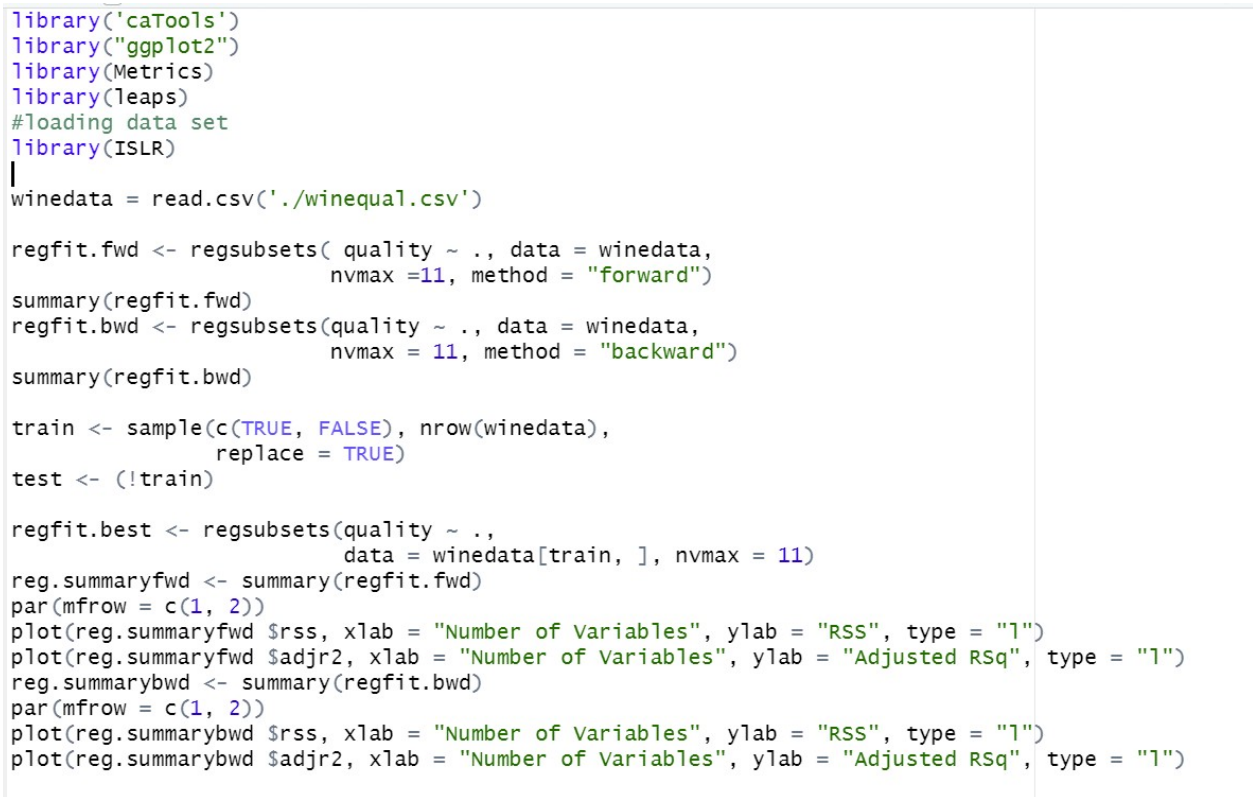
Summary :



```
Call:
lm(formula = concavity_mean ~ naturalSpline(concave.points_worst, 
    df = 3), data = trainingset)

Residuals:
      Min        1Q    Median        3Q       Max 
-0.130495 -0.018504 -0.004347  0.012545  0.193191 

Coefficients:
                                            Estimate Std. Error t value Pr(>|t|)    
(Intercept)                                 0.001446   0.007860   0.184   0.8541    
naturalSpline(concave.points_worst, df = 3)1  0.042845   0.015660   2.736   0.0065 ** 
naturalSpline(concave.points_worst, df = 3)2 -0.236889   0.022490 -10.533   <2e-16 ***
naturalSpline(concave.points_worst, df = 3)3  0.899235   0.042006  21.407   <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.03577 on 394 degrees of freedom
Multiple R-squared:  0.7822,	Adjusted R-squared:  0.7805 
F-statistic: 471.7 on 3 and 394 DF,  p-value: < 2.2e-16
```

Part 2 : Feature selection / Model Optimization Methods

Code :

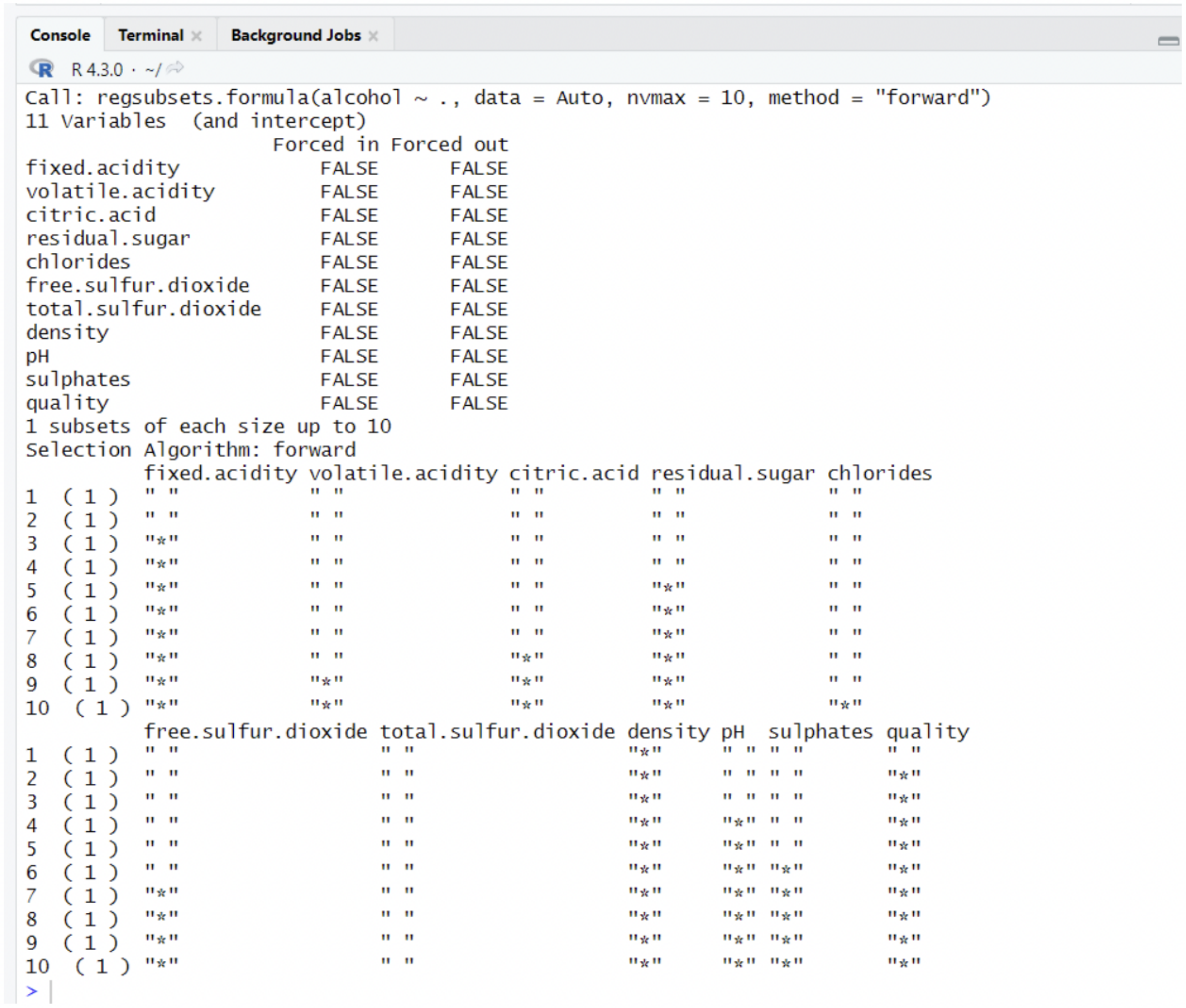


```
library('caTools')
library("ggplot2")
library(Metrics)
library(leaps)
#loading data set
library(ISLR)

winedata = read.csv('./winequal.csv')

regfit.fwd <- regsubsets( quality ~ ., data = winedata,
                         nvmax =11, method = "forward")
summary(regfit.fwd)
regfit.bwd <- regsubsets(quality ~ ., data = winedata,
                         nvmax = 11, method = "backward")
summary(regfit.bwd)

train <- sample(c(TRUE, FALSE), nrow(winedata),
                replace = TRUE)
test <- (!train)

regfit.best <- regsubsets(quality ~ .,
                          data = winedata[train, ], nvmax = 11)
reg.summaryfwd <- summary(regfit.fwd)
par(mfrow = c(1, 2))
plot(reg.summaryfwd $rss, xlab = "Number of Variables", ylab = "RSS", type = "l")
plot(reg.summaryfwd $adjr2, xlab = "Number of Variables", ylab = "Adjusted RSq", type = "l")
reg.summarybwd <- summary(regfit.bwd)
par(mfrow = c(1, 2))
plot(reg.summarybwd $rss, xlab = "Number of Variables", ylab = "RSS", type = "l")
plot(reg.summarybwd $adjr2, xlab = "Number of Variables", ylab = "Adjusted RSq", type = "l")
```

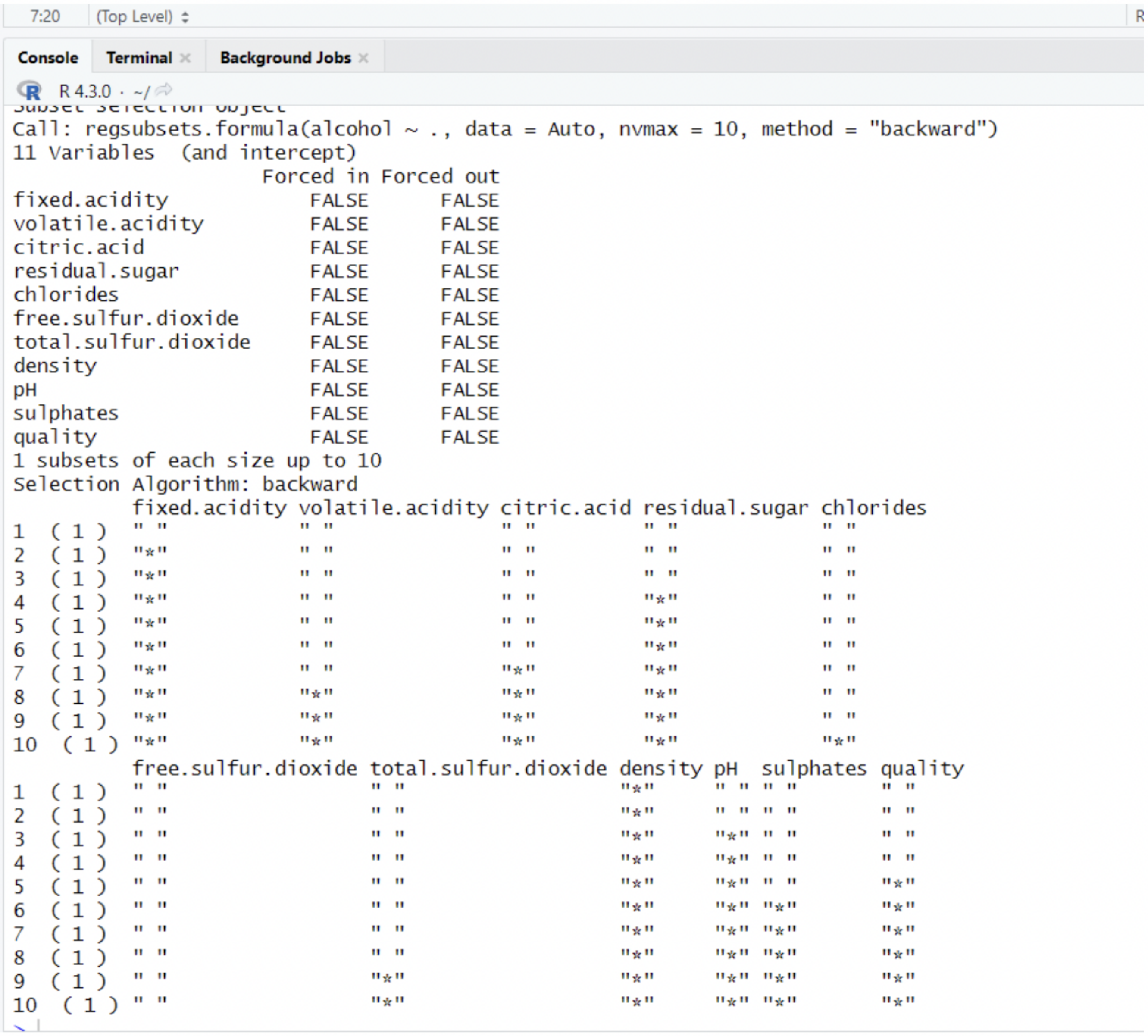
Forward:

Code:

```
library(ISLR2)

library(leaps)

Auto <- na.omit(Auto)

regfit.fwd <- regsubsets(alcohol ~ ., data = Auto,

 nvmax = 10, method = "forward")

summary(regfit.fwd)
```



```
Call: regsubsets.formula(alcohol ~ ., data = Auto, nvmax = 10, method = "forward")
11 Variables  (and intercept)
                     Forced in Forced out
fixed.acidity            FALSE      FALSE
volatile.acidity         FALSE      FALSE
citric.acid              FALSE      FALSE
residual.sugar           FALSE      FALSE
chlorides                FALSE      FALSE
free.sulfur.dioxide      FALSE      FALSE
total.sulfur.dioxide     FALSE      FALSE
density                  FALSE      FALSE
pH                       FALSE      FALSE
sulphates                FALSE      FALSE
quality                  FALSE      FALSE
1 subsets of each size up to 10
Selection Algorithm: forward
          fixed.acidity volatile.acidity citric.acid residual.sugar chlorides
1  ( 1 )  " "           " "              " "         " "            " "      
2  ( 1 )  " "           " "              " "         " "            " "      
3  ( 1 )  "*"           " "              " "         " "            " "      
4  ( 1 )  "*"           " "              " "         " "            " "      
5  ( 1 )  "*"           " "              " "         "*"            " "      
6  ( 1 )  "*"           " "              " "         "*"            " "      
7  ( 1 )  "*"           " "              " "         "*"            " "      
8  ( 1 )  "*"           " "              "*"         "*"            " "      
9  ( 1 )  "*"           "*"              "*"         "*"            " "      
10  ( 1 ) "*"           "*"              "*"         "*"            "*"      
          free.sulfur.dioxide total.sulfur.dioxide density pH  sulphates quality
1  ( 1 )  " "                 " "                  "*"     " " " "       " "    
2  ( 1 )  " "                 " "                  "*"     " " " "       "*"    
3  ( 1 )  " "                 " "                  "*"     " " " "       "*"    
4  ( 1 )  " "                 " "                  "*"     "*" " "       "*"    
5  ( 1 )  " "                 " "                  "*"     "*" " "       "*"    
6  ( 1 )  " "                 " "                  "*"     "*" "*"       "*"    
7  ( 1 )  "*"                 " "                  "*"     "*" "*"       "*"    
8  ( 1 )  "*"                 " "                  "*"     "*" "*"       "*"    
9  ( 1 )  "*"                 " "                  "*"     "*" "*"       "*"    
10  ( 1 ) "*"                 " "                  "*"     "*" "*"       "*"    
> 
```



7:20 (Top Level)

Console Terminal Background Jobs

R 4.3.0 · ~/

```
Subset selection object
Call: regsubsets.formula(alcohol ~ ., data = Auto, nvmax = 10, method = "backward")
11 Variables  (and intercept)
                     Forced in Forced out
fixed.acidity            FALSE      FALSE
volatile.acidity         FALSE      FALSE
citric.acid              FALSE      FALSE
residual.sugar           FALSE      FALSE
chlorides                FALSE      FALSE
free.sulfur.dioxide      FALSE      FALSE
total.sulfur.dioxide     FALSE      FALSE
density                  FALSE      FALSE
pH                       FALSE      FALSE
sulphates                FALSE      FALSE
quality                  FALSE      FALSE
1 subsets of each size up to 10
Selection Algorithm: backward
          fixed.acidity volatile.acidity citric.acid residual.sugar chlorides
1  ( 1 )  " "           " "              " "         " "            " "
2  ( 1 )  "*"           " "              " "         " "            " "
3  ( 1 )  "*"           " "              " "         " "            " "
4  ( 1 )  "*"           " "              " "         "*"            " "
5  ( 1 )  "*"           " "              " "         "*"            " "
6  ( 1 )  "*"           " "              " "         "*"            " "
7  ( 1 )  "*"           " "              "*"         "*"            " "
8  ( 1 )  "*"           "*"              "*"         "*"            " "
9  ( 1 )  "*"           "*"              "*"         "*"            " "
10  ( 1 ) "*"           "*"              "*"         "*"            "*"
          free.sulfur.dioxide total.sulfur.dioxide density pH  sulphates quality
1  ( 1 )  " "                 " "                  "*"     " " " "       " "
2  ( 1 )  " "                 " "                  "*"     " " " "       " "
3  ( 1 )  " "                 " "                  "*"     "*" " "       " "
4  ( 1 )  " "                 " "                  "*"     "*" " "       " "
5  ( 1 )  " "                 " "                  "*"     "*" " "       "*"
6  ( 1 )  " "                 " "                  "*"     "*" "*"       "*"
7  ( 1 )  " "                 " "                  "*"     "*" "*"       "*"
8  ( 1 )  " "                 " "                  "*"     "*" "*"       "*"
9  ( 1 )  " "                 "*"                  "*"     "*" "*"       "*"
10  ( 1 ) " "                 "*"                  "*"     "*" "*"       "*"
```

a) Forward Features

Code:

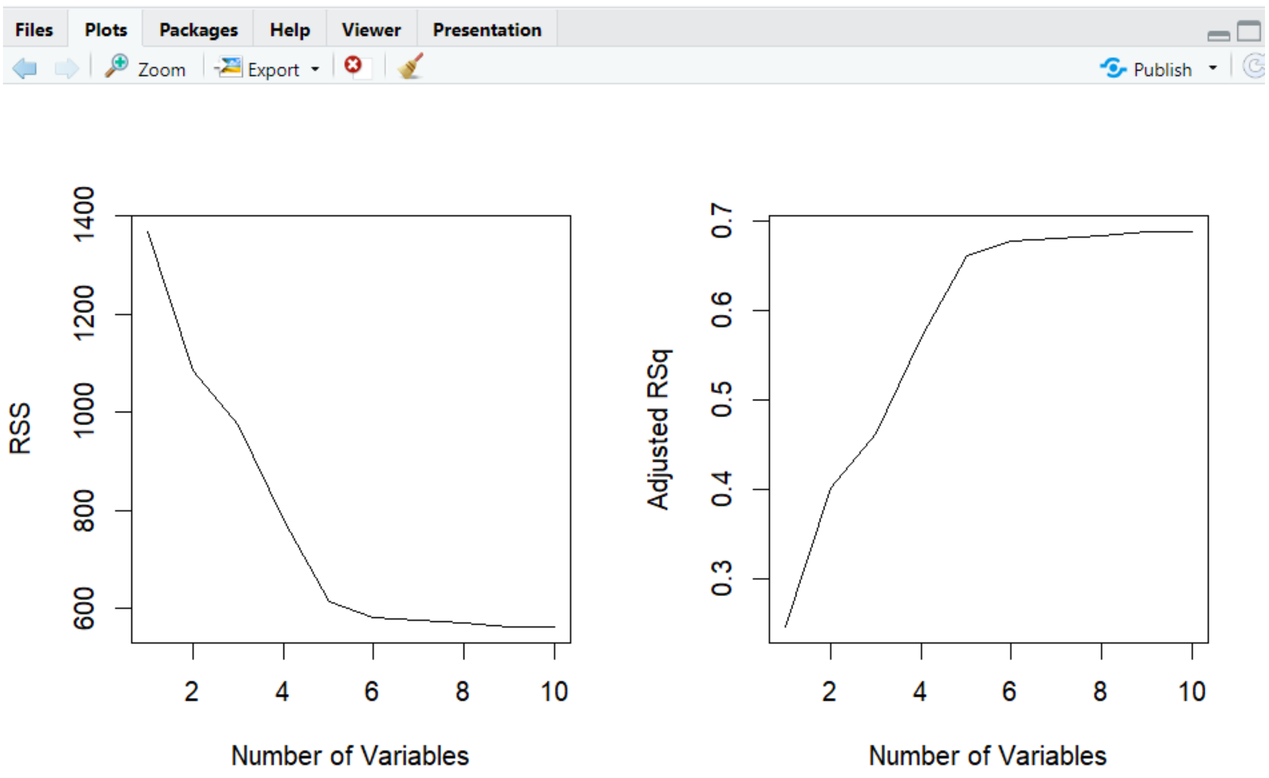
reg.summaryfwd <- summary(regfit.fwd)

par(mfrow = c(1, 2))

plot(reg.summaryfwd $rss, xlab = "Number of Variables", ylab = "RSS", type = "l")

plot(reg.summaryfwd $adjr2, xlab = "Number of Variables", ylab = "Adjusted RSq", type = "l")

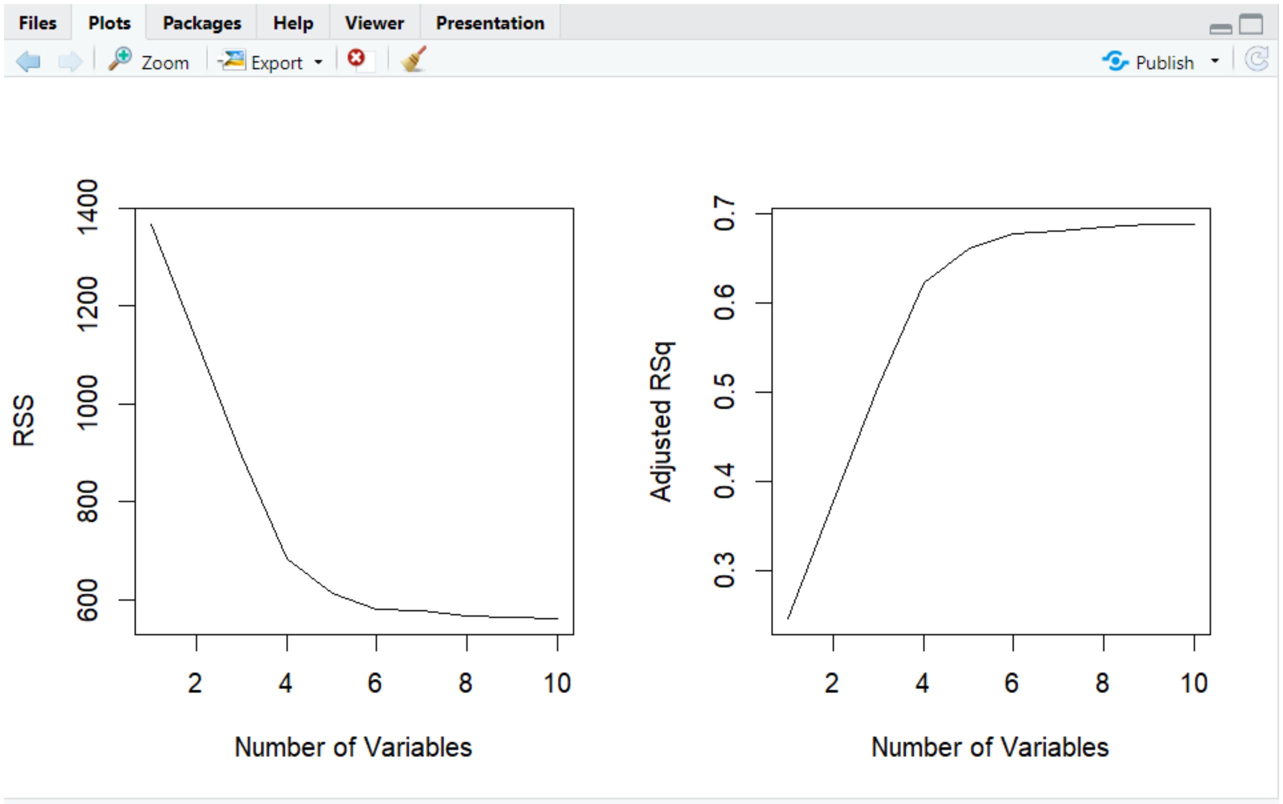
Plots :



b) Backward Features

```
reg.summarybwd <- summary(regfit.bwd)
par(mfrow = c(1, 2))
plot(reg.summarybwd $rss, xlab = "Number of Variables", ylab = "RSS", type = "l")
plot(reg.summarybwd $adjr2, xlab = "Number of Variables", ylab = "Adjusted RSq", type = "l")
```

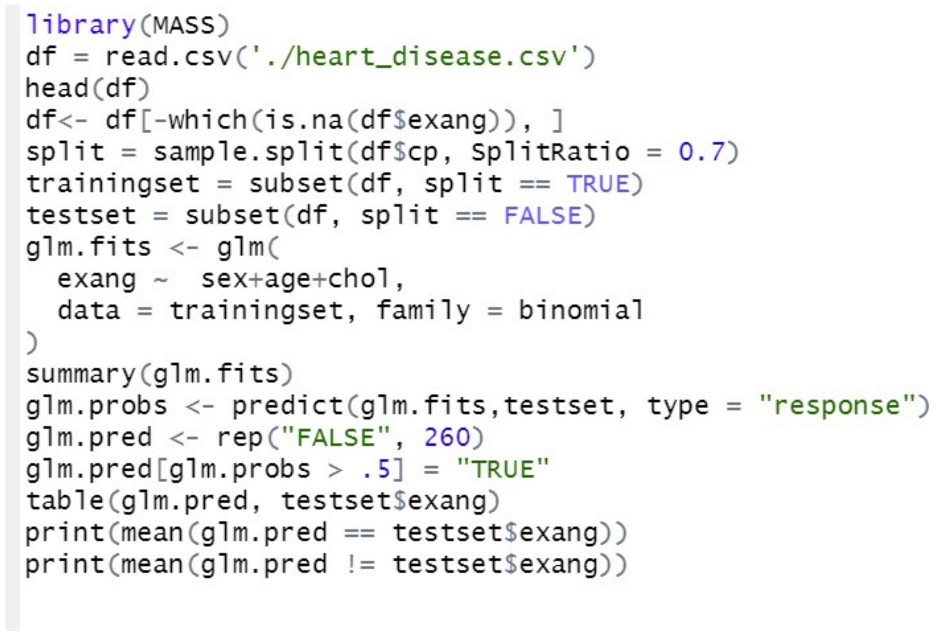
Plots:


Files
Plots
Packages
Help
Viewer
Presentation
Zoom
Export
Publish
RSS
1400
1200
1000
800
600
2
4
6
8
10
Number of Variables
Adjusted RSq
0.7
0.6
0.5
0.4
0.3
2
4
6
8
10
Number of Variables

Part 3 : Classification

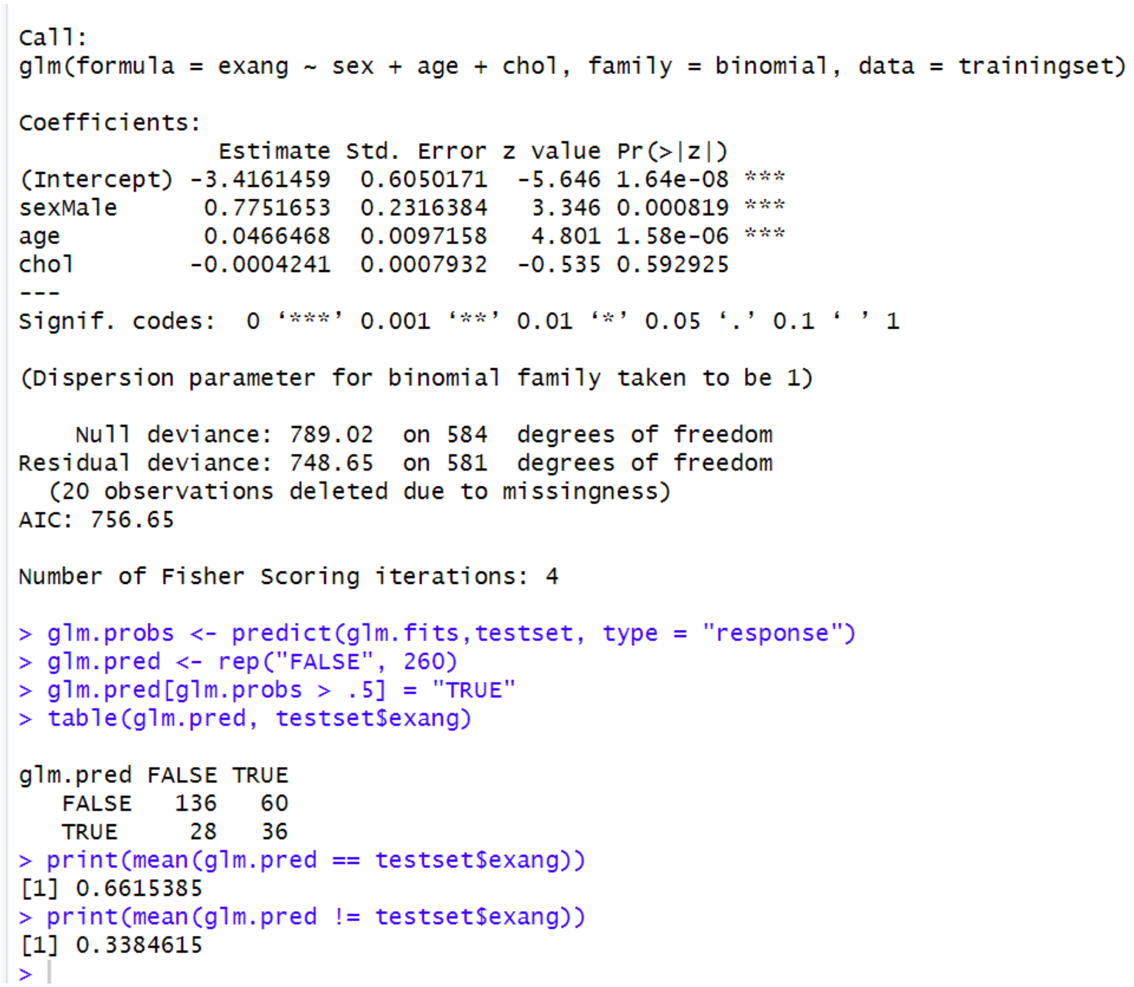
A) Logistic Regression :

Code :



```
library(MASS)
df = read.csv('./heart_disease.csv')
head(df)
df<- df[-which(is.na(df$exang)), ]
split = sample.split(df$cp, SplitRatio = 0.7)
trainingset = subset(df, split == TRUE)
testset = subset(df, split == FALSE)
glm.fits <- glm(
  exang ~  sex+age+chol,
  data = trainingset, family = binomial
)
summary(glm.fits)
glm.probs <- predict(glm.fits,testset, type = "response")
glm.pred <- rep("FALSE", 260)
glm.pred[glm.probs > .5] = "TRUE"
table(glm.pred, testset$exang)
print(mean(glm.pred == testset$exang))
print(mean(glm.pred != testset$exang))
```

Result :



```
Call:
glm(formula = exang ~ sex + age + chol, family = binomial, data = trainingset)

Coefficients:
              Estimate Std. Error z value Pr(>|z|)    
(Intercept) -3.4161459  0.6050171  -5.646 1.64e-08 ***
sexMale      0.7751653  0.2316384   3.346 0.000819 ***
age          0.0466468  0.0097158   4.801 1.58e-06 ***
chol        -0.0004241  0.0007932  -0.535 0.592925    
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 789.02  on 584  degrees of freedom
Residual deviance: 748.65  on 581  degrees of freedom
  (20 observations deleted due to missingness)
AIC: 756.65

Number of Fisher Scoring iterations: 4

> glm.probs <- predict(glm.fits,testset, type = "response")
> glm.pred <- rep("FALSE", 260)
> glm.pred[glm.probs > .5] = "TRUE"
> table(glm.pred, testset$exang)

glm.pred FALSE TRUE
   FALSE   136   60
   TRUE     28   36
> print(mean(glm.pred == testset$exang))
[1] 0.6615385
> print(mean(glm.pred != testset$exang))
[1] 0.3384615
> 
```

Explanation :

Based on above model we are able to predict exang with an 0.642 probability.

B) Linear Discriminant Analysis :

Code :

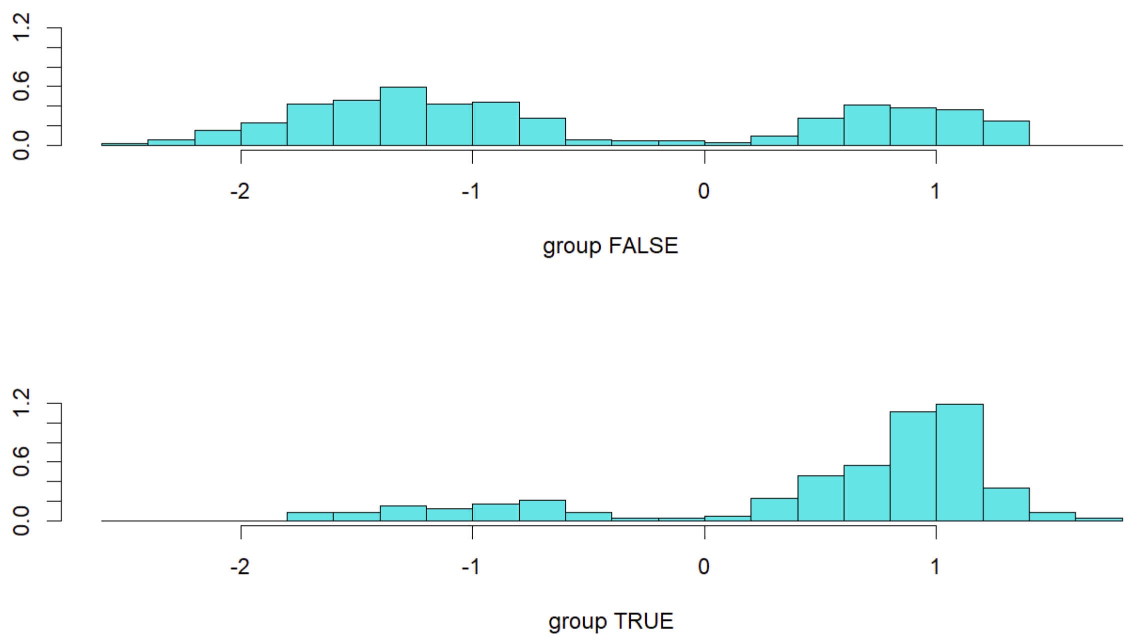
```
library(MASS)
df = read.csv('./heart_disease.csv')
head(df)
df<- df[-which(is.na(df$exang)), ]
split = sample.split(df$cp, SplitRatio = 0.7)
trainingset = subset(df, split == TRUE)
testset = subset(df, split == FALSE)
lda.fit <- lda(exang ~ cp + age + sex, data = trainingset)
plot(lda.fit)
lda.pred <- predict(lda.fit, testset)
lda.class <- lda.pred$class
table(lda.class, testset$exang)
mean(lda.class == testset$exang)
```

Results :

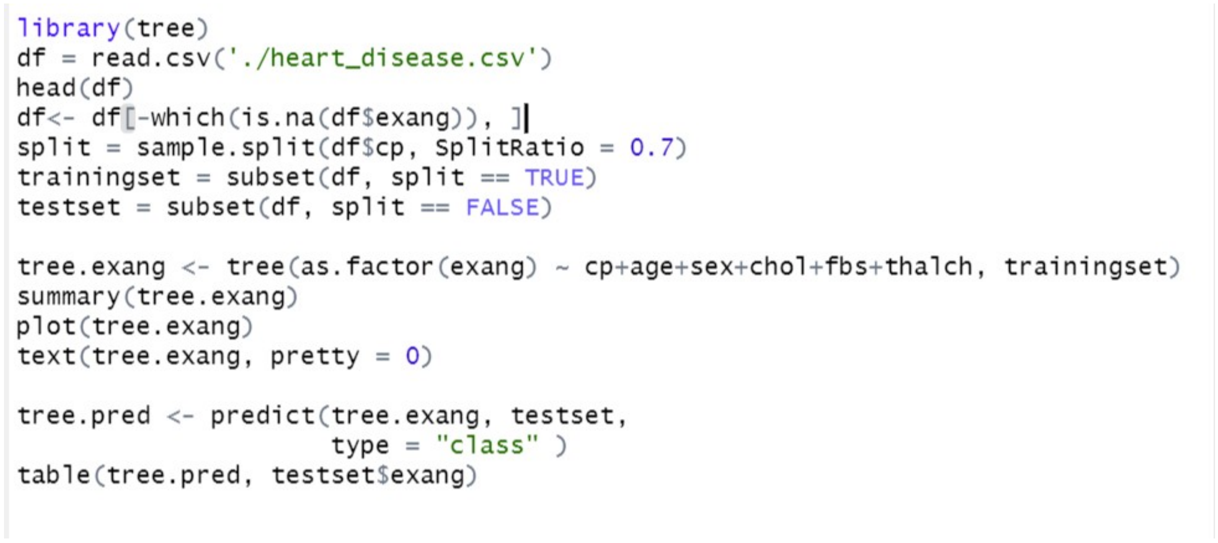
Confusion matrix is below:

```
lda.class FALSE TRUE
    FALSE   110   20
    TRUE     51   79
> mean(lda.class == testset$exang)
[1] 0.7269231
> 
```

Plots :

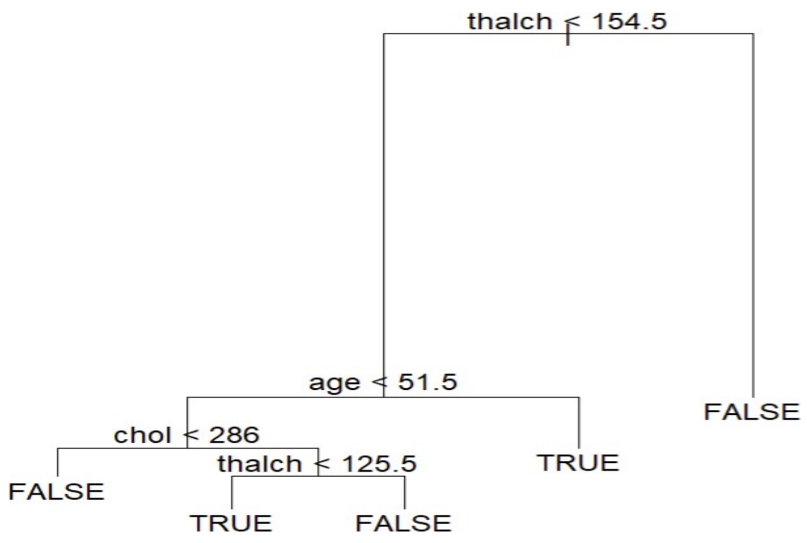


C) **Tree classifier :**
Code :

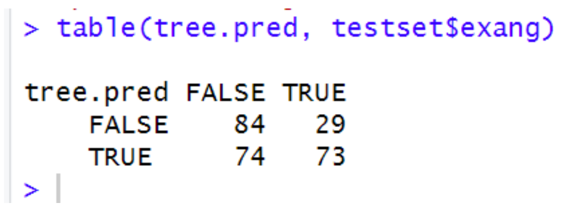


```
library(tree)
df = read.csv('./heart_disease.csv')
head(df)
df<- df[-which(is.na(df$exang)), ]
split = sample.split(df$cp, SplitRatio = 0.7)
trainingset = subset(df, split == TRUE)
testset = subset(df, split == FALSE)

tree.exang <- tree(as.factor(exang) ~ cp+age+sex+chol+fbs+thalch, trainingset)
summary(tree.exang)
plot(tree.exang)
text(tree.exang, pretty = 0)

tree.pred <- predict(tree.exang, testset,
                     type = "class" )
table(tree.pred, testset$exang)
```

Plots :



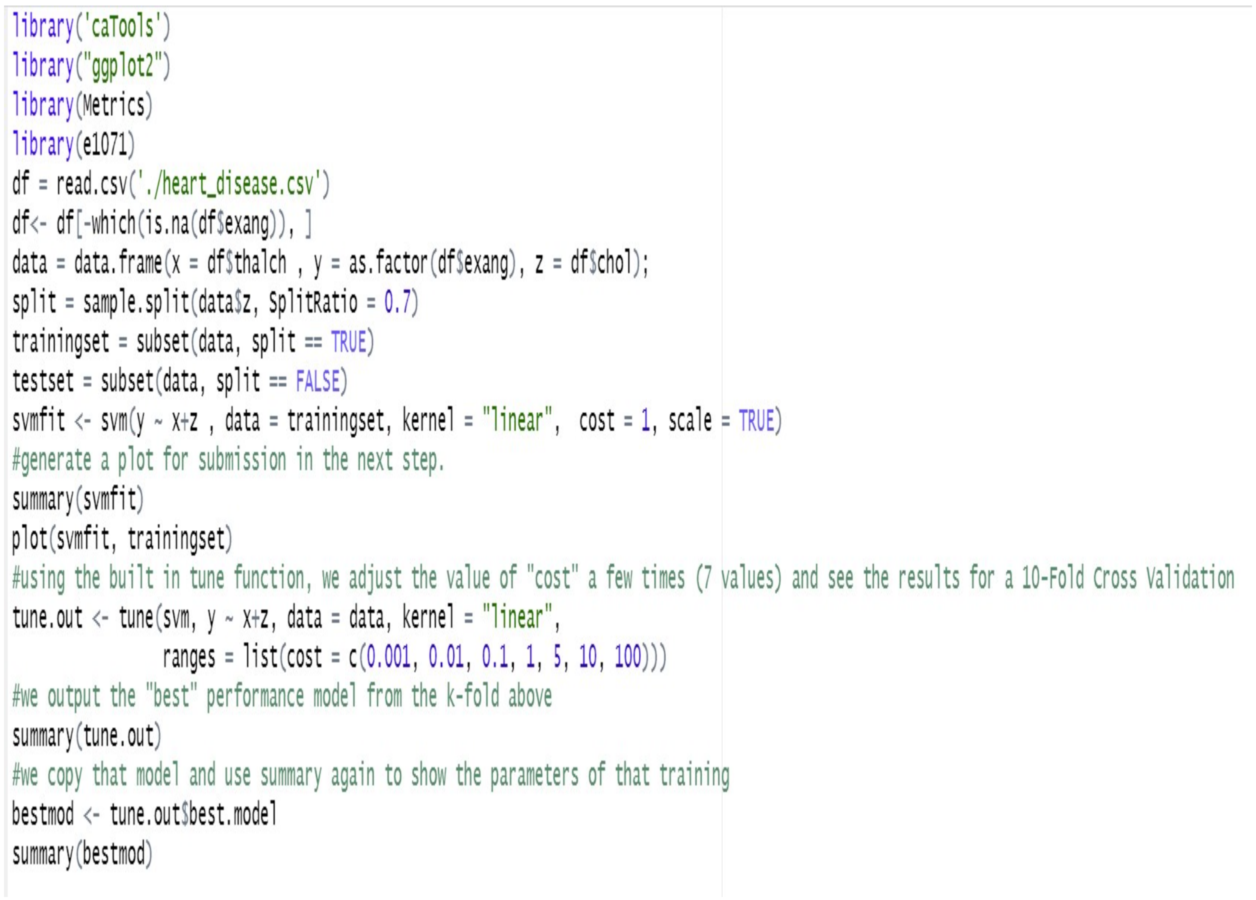
Confusion Matrix :



```
> table(tree.pred, testset$exang)

tree.pred FALSE TRUE
    FALSE    84   29
    TRUE     74   73
> 
```

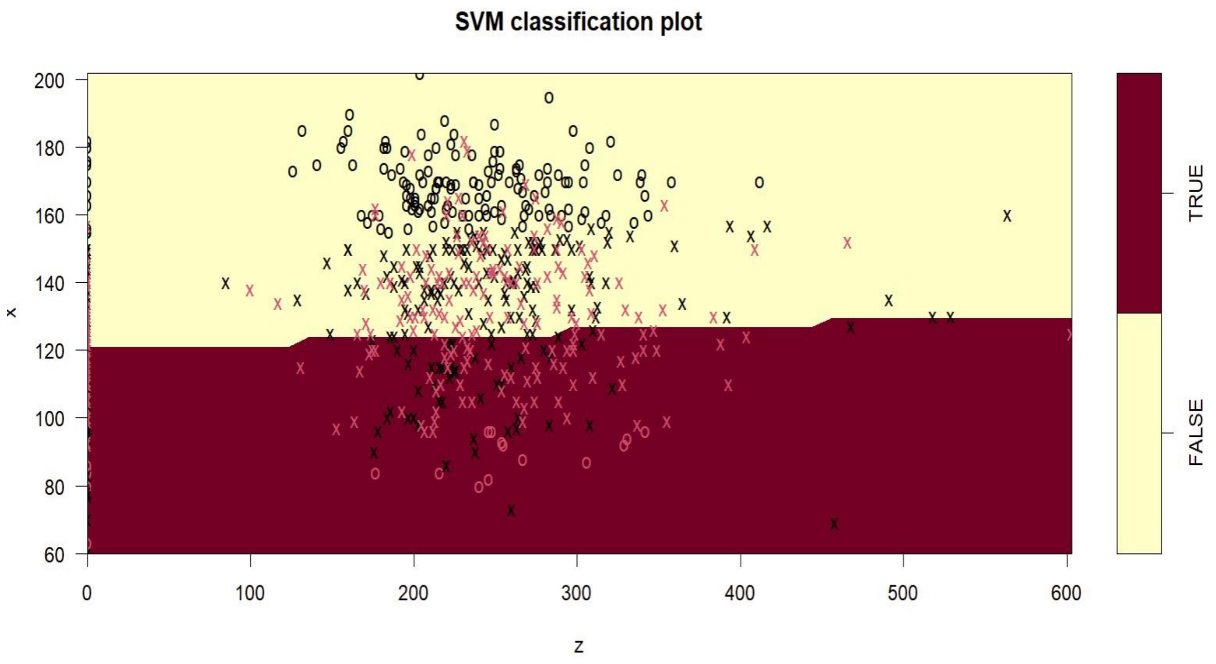
3. SVM Linear classifier :

Code:

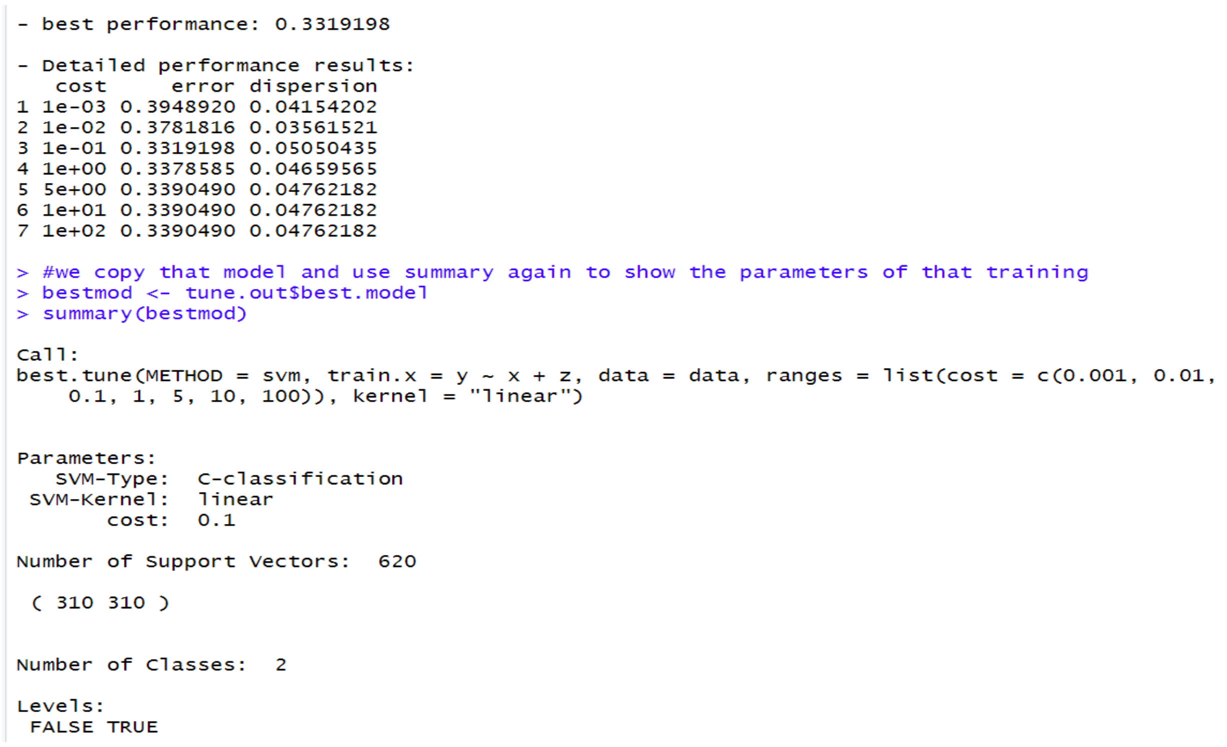


```
library('caTools')
library("ggplot2")
library(Metrics)
library(e1071)
df = read.csv('./heart_disease.csv')
df<- df[-which(is.na(df$exang)), ]
data = data.frame(x = df$thalch , y = as.factor(df$exang), z = df$chol);
split = sample.split(data$z, SplitRatio = 0.7)
trainingset = subset(data, split == TRUE)
testset = subset(data, split == FALSE)
svmfit <- svm(y ~ x+z , data = trainingset, kernel = "linear",  cost = 1, scale = TRUE)
#generate a plot for submission in the next step.
summary(svmfit)
plot(svmfit, trainingset)
#using the built in tune function, we adjust the value of "cost" a few times (7 values) and see the results for a 10-Fold Cross Validation
tune.out <- tune(svm, y ~ x+z, data = data, kernel = "linear",
               ranges = list(cost = c(0.001, 0.01, 0.1, 1, 5, 10, 100)))
#we output the "best" performance model from the k-fold above
summary(tune.out)
#we copy that model and use summary again to show the parameters of that training
bestmod <- tune.out$best.model
summary(bestmod)
```

SVM Plot :



Best Model Summary :



```
- best performance: 0.3319198

- Detailed performance results:
   cost     error  dispersion
1 1e-03 0.3948920 0.04154202
2 1e-02 0.3781816 0.03561521
3 1e-01 0.3319198 0.05050435
4 1e+00 0.3378585 0.04659565
5 5e+00 0.3390490 0.04762182
6 1e+01 0.3390490 0.04762182
7 1e+02 0.3390490 0.04762182

> #we copy that model and use summary again to show the parameters of that training
> bestmod <- tune.out$best.model
> summary(bestmod)

Call:
best.tune(METHOD = svm, train.x = y ~ x + z, data = data, ranges = list(cost = c(0.001, 0.01,
    0.1, 1, 5, 10, 100)), kernel = "linear")


Parameters:
   SVM-Type:  C-classification
 SVM-Kernel:  linear
       cost:  0.1

Number of Support Vectors:  620

 ( 310 310 )


Number of Classes:  2

Levels:
 FALSE TRUE
```

Explanation :

With this SVM classification I tried to predict exang based on chol and thalch. I was able to generate classifier with probability of 0.62.

Part 4 : Models

A) **(5 points) A friend is starting a company and wants your help to see if they can figure out what factors most closely relate to the relative level of success for key competitors. They have gathered a few factors about each company such as total inventory, number of employees, annual operation budget and total profits. What method might you use to help your friend determine if their business model might be a success? Why did you choose this model?**

Ans) Total inventory, headcount, annual operational budget, and total revenue are the four ways to identify anything. Every variable seems to have a linear relationship. To assess the likelihood that his business plan will be successful, I would suggest carrying out a multiple linear regression study.

B) **(5 points) An advertisement firm has hired you to help them optimize their mailing list. They currently are looking to promote their client's store by sending packages of coupons to select areas. We want to know which postal codes the company should mail to for maximum impact (shoppers come to the store with coupons). They currently have some survey data randomly sampled from homes in the area indicating how likely they were to shop at the client's location.What method might you try first to generate the mailing map? Why?**

Ans) We can use logistic regression to ascertain the likelihood that a customer will shop at the client's location. Based on overall results, we can pinpoint the particular regions where mailing will have the greatest impact.

C) **(5 points) A large company has been collecting data about their customers preferences for many years. They've hired you to help them transform the millions of samples and thousands of search and behavior features into a set of simplified features they can use to build a model which provides suggestions to their customers for future services. What method might you suggest first? Why?**

Ans) We must transform the enormous amount of data we have into traits that are simpler to comprehend. We can use PCA to get a linear combination of variables. The finest regression algorithms can then be chosen in order to acquire additional insights.

d)(5 points) A company that specializes in shipping fruit to grocery stores wants to save money by sorting out bad fruit from good fruit before it goes on the truck. They have presented you with a device that can measure features like weight, color, size, and look for possible bad spots. Each of these measurements is imprecise, and there is significantoverlap between the classes for most of the features. What supervised learning methods might you try? Why?

Ans) It is difficult to categorise this since, as was already mentioned, there is a lot of overlap between classes. I would have thought about multidimensional SVM if the characteristics were easily separable. However, it seems that the random forest classifier can more accurately handle these overlaps.

.