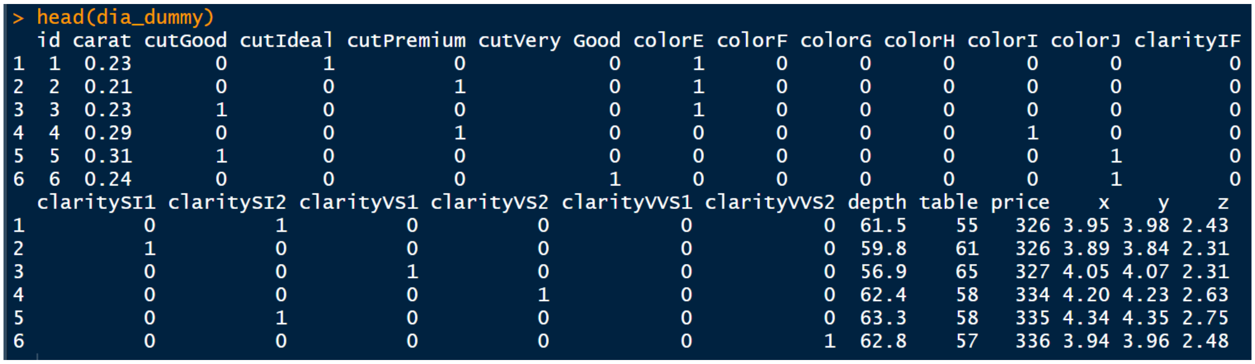
OUTPUTS:

Multiple regression model:

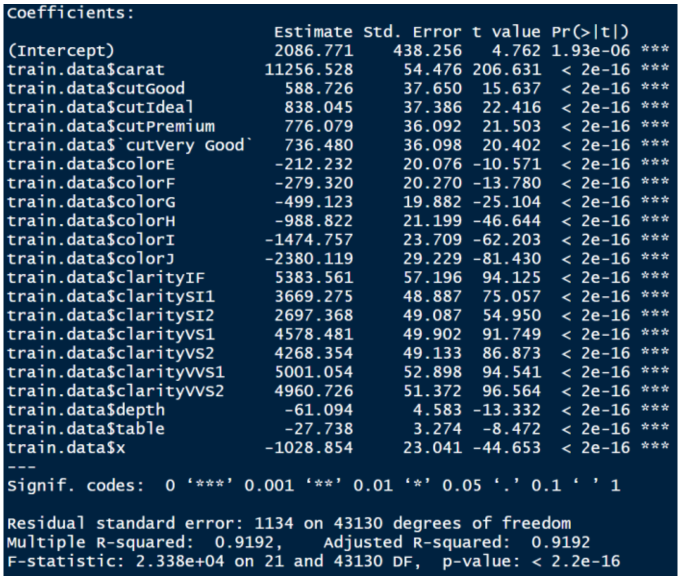
- Data preprocessing:



```
> head(dia_dummy)
  id carat cutGood cutIdeal cutPremium cutVery Good colorE colorF colorG colorH colorI colorJ clarityIF
1  1  0.23       0        1          0            0      1      0      0      0      0      0         0
2  2  0.21       0        0          1            0      1      0      0      0      0      0         0
3  3  0.23       1        0          0            0      1      0      0      0      0      0         0
4  4  0.29       0        0          1            0      0      0      0      0      1      0         0
5  5  0.31       1        0          0            0      0      0      0      0      0      1         0
6  6  0.24       0        0          0            1      0      0      0      0      0      1         0
  claritySI1 claritySI2 clarityVS1 clarityVS2 clarityVVS1 clarityVVS2 depth table price    x    y    z
1          0          1          0          0           0           0  61.5    55   326 3.95 3.98 2.43
2          1          0          0          0           0           0  59.8    61   326 3.89 3.84 2.31
3          0          0          1          0           0           0  56.9    65   327 4.05 4.07 2.31
4          0          0          0          1           0           0  62.4    58   334 4.20 4.23 2.63
5          0          1          0          0           0           0  63.3    58   335 4.34 4.35 2.75
6          0          0          0          0           0           1  62.8    57   336 3.94 3.96 2.48
```

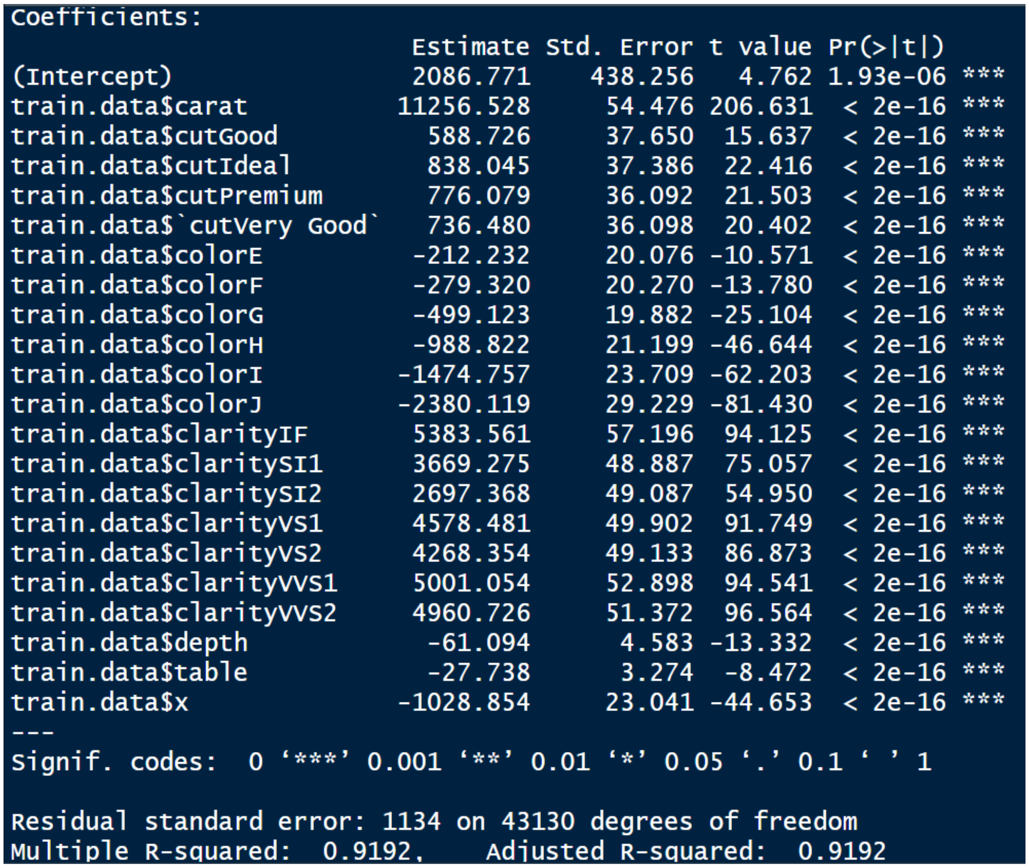
- Feature selection:

  Backward using p value elimination:



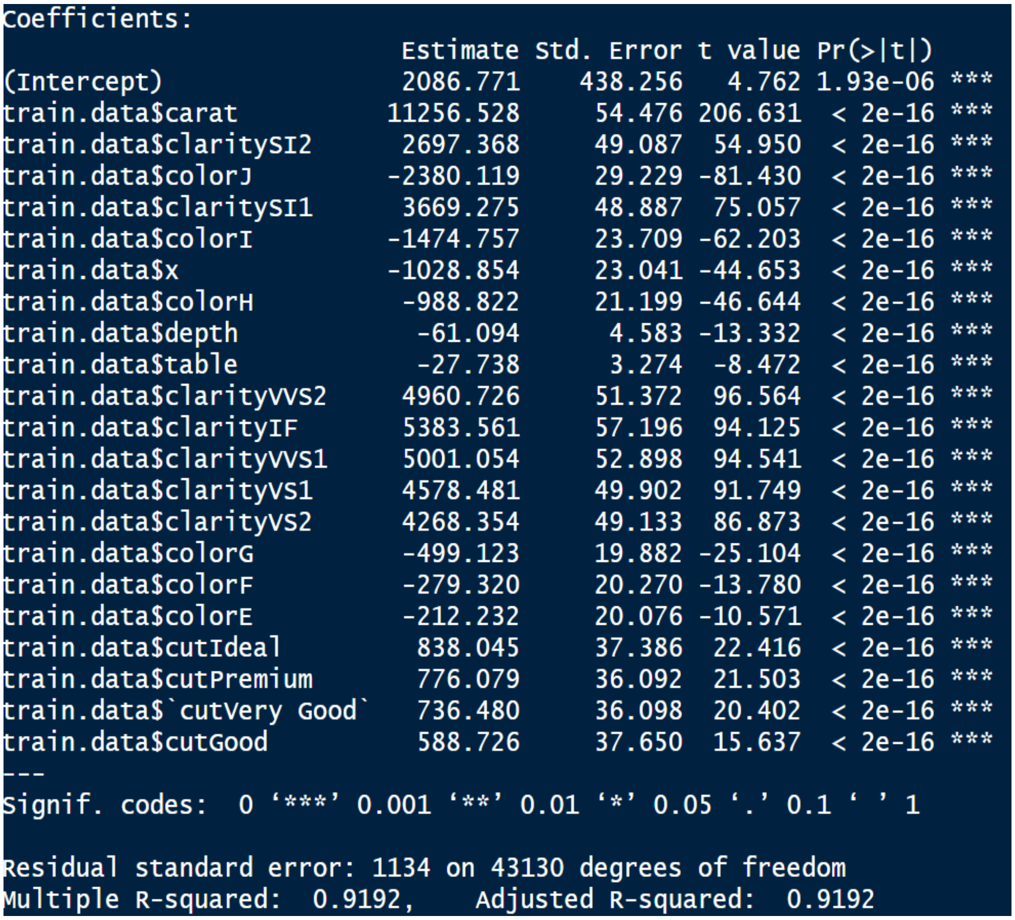
```
Coefficients:
                        Estimate Std. Error t value Pr(>|t|)    
(Intercept)             2086.771    438.256   4.762 1.93e-06 ***
train.data$carat       11256.528     54.476 206.631  < 2e-16 ***
train.data$cutGood       588.726     37.650  15.637  < 2e-16 ***
train.data$cutIdeal      838.045     37.386  22.416  < 2e-16 ***
train.data$cutPremium    776.079     36.092  21.503  < 2e-16 ***
train.data$`cutVery Good` 736.480    36.098  20.402  < 2e-16 ***
train.data$colorE       -212.232     20.076 -10.571  < 2e-16 ***
train.data$colorF       -279.320     20.270 -13.780  < 2e-16 ***
train.data$colorG       -499.123     19.882 -25.104  < 2e-16 ***
train.data$colorH       -988.822     21.199 -46.644  < 2e-16 ***
train.data$colorI      -1474.757     23.709 -62.203  < 2e-16 ***
train.data$colorJ      -2380.119     29.229 -81.430  < 2e-16 ***
train.data$clarityIF    5383.561     57.196  94.125  < 2e-16 ***
train.data$claritySI1   3669.275     48.887  75.057  < 2e-16 ***
train.data$claritySI2   2697.368     49.087  54.950  < 2e-16 ***
train.data$clarityVS1   4578.481     49.902  91.749  < 2e-16 ***
train.data$clarityVS2   4268.354     49.133  86.873  < 2e-16 ***
train.data$clarityVVS1  5001.054     52.898  94.541  < 2e-16 ***
train.data$clarityVVS2  4960.726     51.372  96.564  < 2e-16 ***
train.data$depth         -61.094      4.583 -13.332  < 2e-16 ***
train.data$table         -27.738      3.274  -8.472  < 2e-16 ***
train.data$x           -1028.854     23.041 -44.653  < 2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 1134 on 43130 degrees of freedom
Multiple R-squared:  0.9192,	Adjusted R-squared:  0.9192 
F-statistic: 2.338e+04 on 21 and 43130 DF,  p-value: < 2.2e-16
```

  Backward using aic



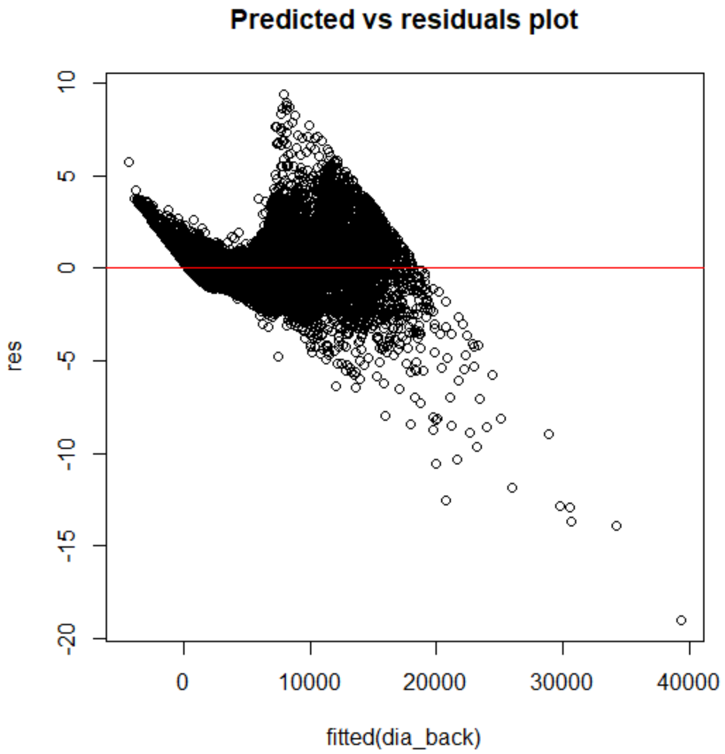
```
Coefficients:
                          Estimate Std. Error t value Pr(>|t|)    
(Intercept)               2086.771    438.256   4.762 1.93e-06 ***
train.data$carat         11256.528     54.476 206.631  < 2e-16 ***
train.data$cutGood         588.726     37.650  15.637  < 2e-16 ***
train.data$cutIdeal        838.045     37.386  22.416  < 2e-16 ***
train.data$cutPremium      776.079     36.092  21.503  < 2e-16 ***
train.data$`cutVery Good`  736.480     36.098  20.402  < 2e-16 ***
train.data$colorE         -212.232     20.076 -10.571  < 2e-16 ***
train.data$colorF         -279.320     20.270 -13.780  < 2e-16 ***
train.data$colorG         -499.123     19.882 -25.104  < 2e-16 ***
train.data$colorH         -988.822     21.199 -46.644  < 2e-16 ***
train.data$colorI        -1474.757     23.709 -62.203  < 2e-16 ***
train.data$colorJ        -2380.119     29.229 -81.430  < 2e-16 ***
train.data$clarityIF      5383.561     57.196  94.125  < 2e-16 ***
train.data$claritySI1     3669.275     48.887  75.057  < 2e-16 ***
train.data$claritySI2     2697.368     49.087  54.950  < 2e-16 ***
train.data$clarityVS1     4578.481     49.902  91.749  < 2e-16 ***
train.data$clarityVS2     4268.354     49.133  86.873  < 2e-16 ***
train.data$clarityVVS1    5001.054     52.898  94.541  < 2e-16 ***
train.data$clarityVVS2    4960.726     51.372  96.564  < 2e-16 ***
train.data$depth           -61.094      4.583 -13.332  < 2e-16 ***
train.data$table           -27.738      3.274  -8.472  < 2e-16 ***
train.data$x             -1028.854     23.041 -44.653  < 2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 1134 on 43130 degrees of freedom
Multiple R-squared:  0.9192,	Adjusted R-squared:  0.9192
```

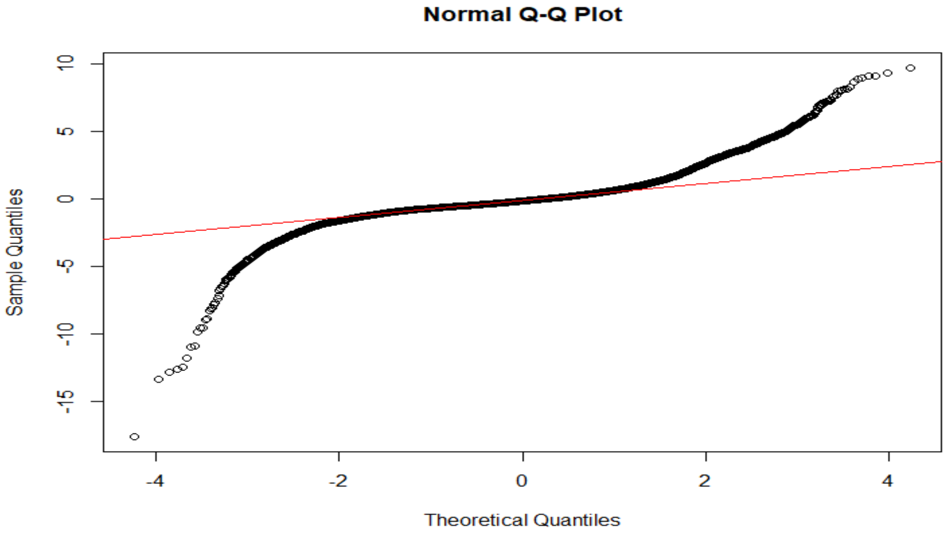
Forward using aic:



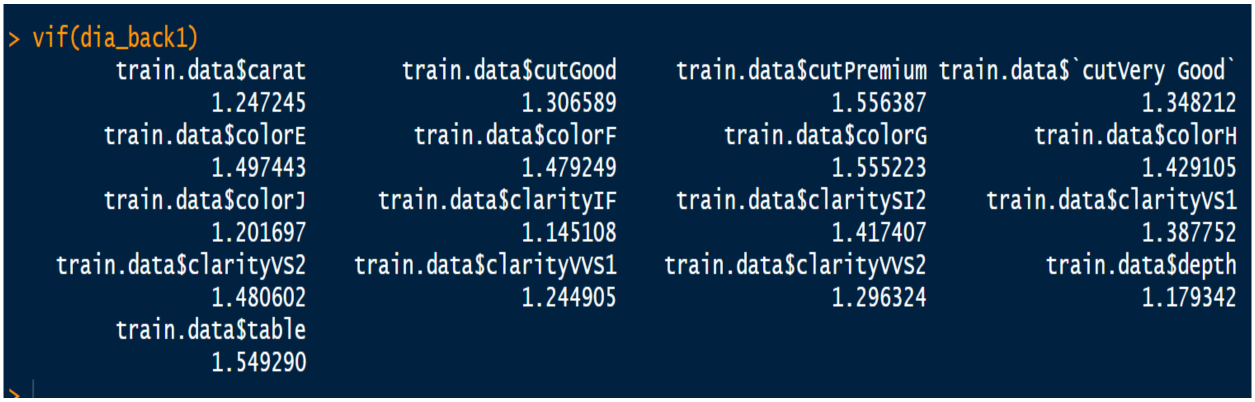
```
Coefficients:
                           Estimate Std. Error t value Pr(>|t|)    
(Intercept)                2086.771    438.256   4.762 1.93e-06 ***
train.data$carat          11256.528     54.476 206.631  < 2e-16 ***
train.data$claritySI2      2697.368     49.087  54.950  < 2e-16 ***
train.data$colorJ         -2380.119     29.229 -81.430  < 2e-16 ***
train.data$claritySI1      3669.275     48.887  75.057  < 2e-16 ***
train.data$colorI         -1474.757     23.709 -62.203  < 2e-16 ***
train.data$x              -1028.854     23.041 -44.653  < 2e-16 ***
train.data$colorH          -988.822     21.199 -46.644  < 2e-16 ***
train.data$depth            -61.094      4.583 -13.332  < 2e-16 ***
train.data$table            -27.738      3.274  -8.472  < 2e-16 ***
train.data$clarityVVS2     4960.726     51.372  96.564  < 2e-16 ***
train.data$clarityIF       5383.561     57.196  94.125  < 2e-16 ***
train.data$clarityVVS1     5001.054     52.898  94.541  < 2e-16 ***
train.data$clarityVS1      4578.481     49.902  91.749  < 2e-16 ***
train.data$clarityVS2      4268.354     49.133  86.873  < 2e-16 ***
train.data$colorG          -499.123     19.882 -25.104  < 2e-16 ***
train.data$colorF          -279.320     20.270 -13.780  < 2e-16 ***
train.data$colorE          -212.232     20.076 -10.571  < 2e-16 ***
train.data$cutIdeal         838.045     37.386  22.416  < 2e-16 ***
train.data$cutPremium       776.079     36.092  21.503  < 2e-16 ***
train.data$`cutVery Good`   736.480     36.098  20.402  < 2e-16 ***
train.data$cutGood          588.726     37.650  15.637  < 2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 1134 on 43130 degrees of freedom
Multiple R-squared:  0.9192,    Adjusted R-squared:  0.9192
```

- Residual analysis:



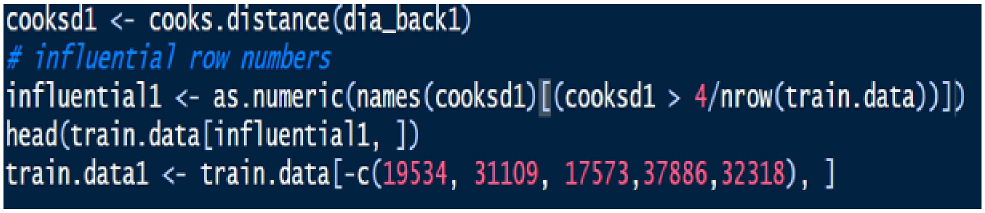


- Postprocessing
  Vif:

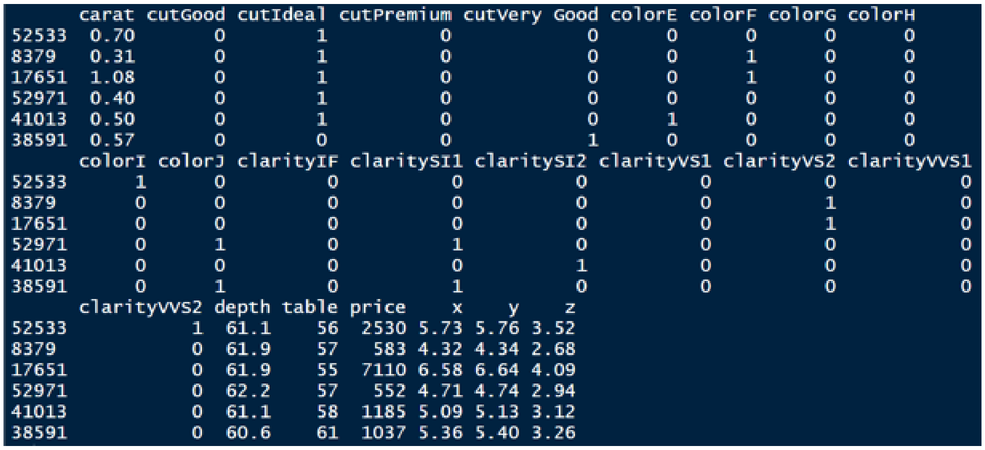


```
> vif(dia_back1)
          train.data$carat         train.data$cutGood      train.data$cutPremium train.data$`cutVery Good`
                  1.247245                   1.306589                   1.556387                   1.348212
         train.data$colorE          train.data$colorF          train.data$colorG          train.data$colorH
                  1.497443                   1.479249                   1.555223                   1.429105
         train.data$colorJ       train.data$clarityIF      train.data$claritySI2      train.data$clarityVS1
                  1.201697                   1.145108                   1.417407                   1.387752
      train.data$clarityVS2     train.data$clarityVVS1     train.data$clarityVVS2          train.data$depth
                  1.480602                   1.244905                   1.296324                   1.179342
          train.data$table
                  1.549290
>
```

Influential points:

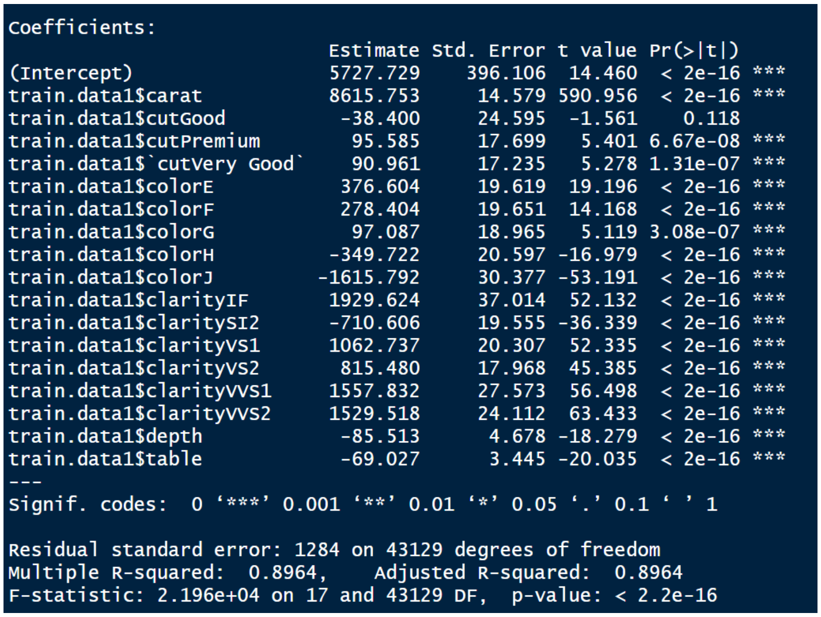


```
cooksd1 <- cooks.distance(dia_back1)
# influential row numbers
influential1 <- as.numeric(names(cooksd1)[(cooksd1 > 4/nrow(train.data))])
head(train.data[influential1, ])
train.data1 <- train.data[-c(19534, 31109, 17573,37886,32318), ]
```



```
      carat cutGood cutIdeal cutPremium cutVery Good colorE colorF colorG colorH
52533  0.70       0        1          0            0      0      0      0      0
8379   0.31       0        1          0            0      0      1      0      0
17651  1.08       0        1          0            0      0      1      0      0
52971  0.40       0        1          0            0      0      0      0      0
41013  0.50       0        1          0            0      1      0      0      0
38591  0.57       0        0          0            1      0      0      0      0
      colorI colorJ clarityIF claritySI1 claritySI2 clarityVS1 clarityVS2 clarityVVS1
52533      1      0         0          0          0          0          0           0
8379       0      0         0          0          0          0          1           0
17651      0      0         0          0          0          0          1           0
52971      0      1         0          1          0          0          0           0
41013      0      0         0          0          1          0          0           0
38591      0      1         0          1          0          0          0           0
      clarityVVS2 depth table price    x    y    z
52533           1  61.1    56  2530 5.73 5.76 3.52
8379            0  61.9    57   583 4.32 4.34 2.68
17651           0  61.9    55  7110 6.58 6.64 4.09
52971           0  62.2    57   552 4.71 4.74 2.94
41013           0  61.1    58  1185 5.09 5.13 3.12
38591           0  60.6    61  1037 5.36 5.40 3.26
```

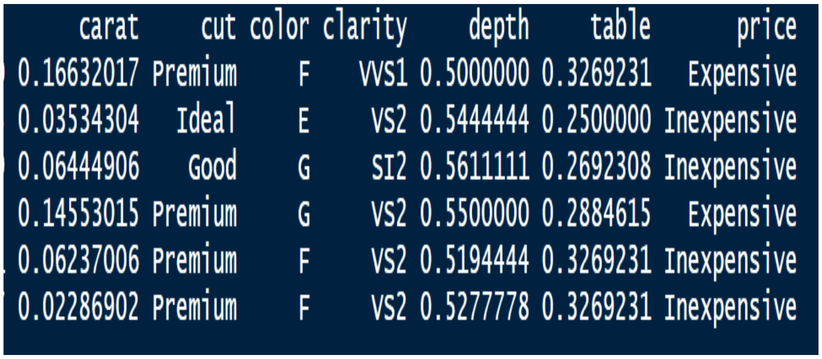
- Final model:

- 

```
Coefficients:
                            Estimate Std. Error t value Pr(>|t|)    
(Intercept)                 5727.729    396.106  14.460  < 2e-16 ***
train.data1$carat           8615.753     14.579 590.956  < 2e-16 ***
train.data1$cutGood          -38.400     24.595  -1.561    0.118    
train.data1$cutPremium        95.585     17.699   5.401 6.67e-08 ***
train.data1$`cutVery Good`    90.961     17.235   5.278 1.31e-07 ***
train.data1$colorE           376.604     19.619  19.196  < 2e-16 ***
train.data1$colorF           278.404     19.651  14.168  < 2e-16 ***
train.data1$colorG            97.087     18.965   5.119 3.08e-07 ***
train.data1$colorH          -349.722     20.597 -16.979  < 2e-16 ***
train.data1$colorJ         -1615.792     30.377 -53.191  < 2e-16 ***
train.data1$clarityIF       1929.624     37.014  52.132  < 2e-16 ***
train.data1$claritySI2      -710.606     19.555 -36.339  < 2e-16 ***
train.data1$clarityVS1      1062.737     20.307  52.335  < 2e-16 ***
train.data1$clarityVS2       815.480     17.968  45.385  < 2e-16 ***
train.data1$clarityVVS1     1557.832     27.573  56.498  < 2e-16 ***
train.data1$clarityVVS2     1529.518     24.112  63.433  < 2e-16 ***
train.data1$depth            -85.513      4.678 -18.279  < 2e-16 ***
train.data1$table            -69.027      3.445 -20.035  < 2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 1284 on 43129 degrees of freedom
Multiple R-squared:  0.8964,	Adjusted R-squared:  0.8964 
F-statistic: 2.196e+04 on 17 and 43129 DF,  p-value: < 2.2e-16
```

Classification

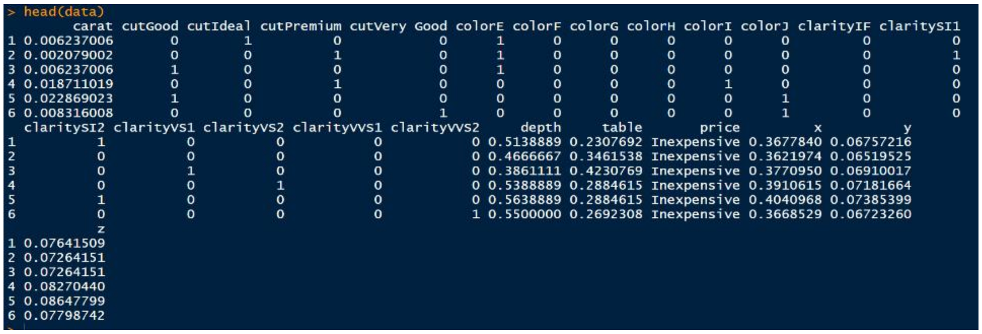
- Preprocessing:



```
       carat     cut color clarity     depth     table       price
 0.16632017 Premium     F    VVS1 0.5000000 0.3269231   Expensive
 0.03534304   Ideal     E     VS2 0.5444444 0.2500000 Inexpensive
 0.06444906    Good     G     SI2 0.5611111 0.2692308 Inexpensive
 0.14553015 Premium     G     VS2 0.5500000 0.2884615   Expensive
 0.06237006 Premium     F     VS2 0.5194444 0.3269231 Inexpensive
 0.02286902 Premium     F     VS2 0.5277778 0.3269231 Inexpensive
```

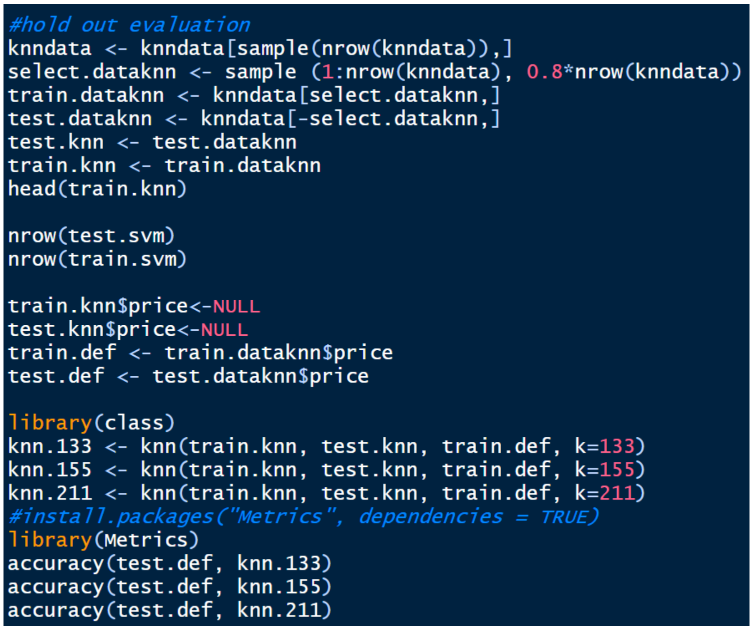
- Knn

  Preprocessing:



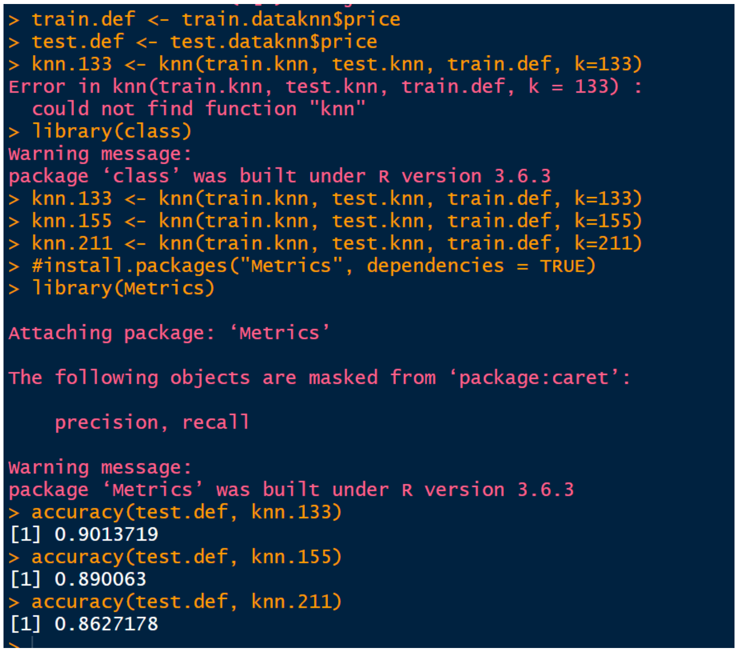
```
> head(data)
        carat cutGood cutIdeal cutPremium cutVery Good colorE colorF colorG colorH colorI colorJ clarityIF claritySI1
1 0.006237006       0        1          0            0      1      0      0      0      0      0         0          0
2 0.002079002       0        0          1            0      1      0      0      0      0      0         0          1
3 0.006237006       1        0          0            0      1      0      0      0      0      0         0          0
4 0.018711019       0        0          1            0      0      0      0      0      1      0         0          0
5 0.022869023       1        0          0            0      0      0      0      0      0      1         0          0
6 0.008316008       0        0          0            1      0      0      0      0      0      1         0          0
  claritySI2 clarityVS1 clarityVS2 clarityVVS1 clarityVVS2     depth     table       price         x          y
1          1          0          0           0           0 0.5138889 0.2307692 Inexpensive 0.3677840 0.06757216
2          0          0          0           0           0 0.4666667 0.3461538 Inexpensive 0.3621974 0.06519525
3          0          1          0           0           0 0.3861111 0.4230769 Inexpensive 0.3770950 0.06910017
4          0          0          1           0           0 0.5388889 0.2884615 Inexpensive 0.3910615 0.07181664
5          1          0          0           0           0 0.5638889 0.2884615 Inexpensive 0.4040968 0.07385399
6          0          0          0           0           1 0.5500000 0.2692308 Inexpensive 0.3668529 0.06723260
           z
1 0.07641509
2 0.07264151
3 0.07264151
4 0.08270440
5 0.08647799
6 0.07798742
```

Splitting of data:



```
#hold out evaluation
knndata <- knndata[sample(nrow(knndata)),]
select.dataknn <- sample (1:nrow(knndata), 0.8*nrow(knndata))
train.dataknn <- knndata[select.dataknn,]
test.dataknn <- knndata[-select.dataknn,]
test.knn <- test.dataknn
train.knn <- train.dataknn
head(train.knn)

nrow(test.svm)
nrow(train.svm)

train.knn$price<-NULL
test.knn$price<-NULL
train.def <- train.dataknn$price
test.def <- test.dataknn$price

library(class)
knn.133 <- knn(train.knn, test.knn, train.def, k=133)
knn.155 <- knn(train.knn, test.knn, train.def, k=155)
knn.211 <- knn(train.knn, test.knn, train.def, k=211)
#install.packages("Metrics", dependencies = TRUE)
library(Metrics)
accuracy(test.def, knn.133)
accuracy(test.def, knn.155)
accuracy(test.def, knn.211)
```

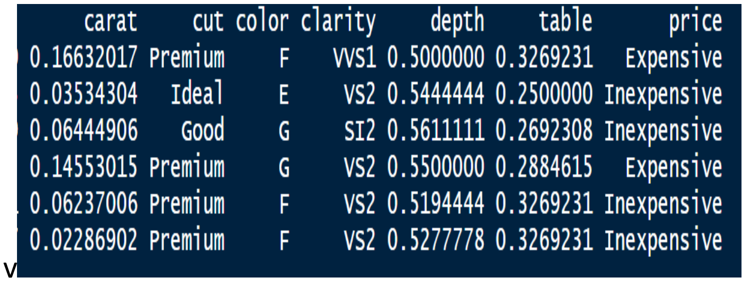
Accuracy:



```
> train.def <- train.dataknn$price
> test.def <- test.dataknn$price
> knn.133 <- knn(train.knn, test.knn, train.def, k=133)
Error in knn(train.knn, test.knn, train.def, k = 133) : 
  could not find function "knn"
> library(class)
Warning message:
package ‘class’ was built under R version 3.6.3 
> knn.133 <- knn(train.knn, test.knn, train.def, k=133)
> knn.155 <- knn(train.knn, test.knn, train.def, k=155)
> knn.211 <- knn(train.knn, test.knn, train.def, k=211)
> #install.packages("Metrics", dependencies = TRUE)
> library(Metrics)

Attaching package: ‘Metrics’

The following objects are masked from ‘package:caret’:

    precision, recall

Warning message:
package ‘Metrics’ was built under R version 3.6.3 
> accuracy(test.def, knn.133)
[1] 0.9013719
> accuracy(test.def, knn.155)
[1] 0.890063
> accuracy(test.def, knn.211)
[1] 0.8627178
```

- Naïve bayes:

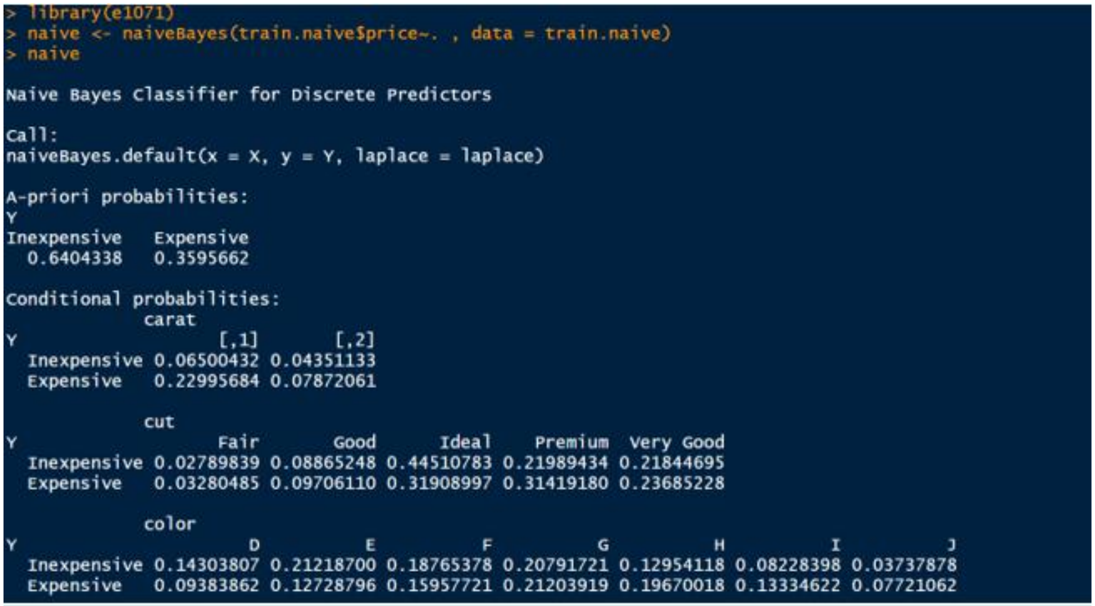
  Preprocessing:



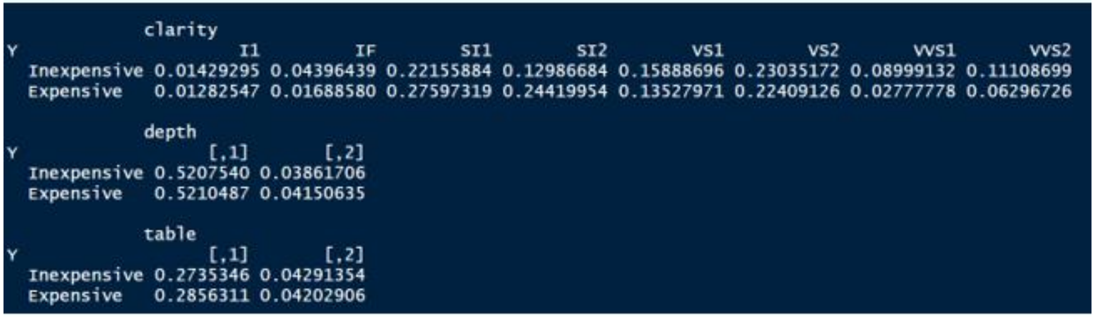
```
       carat     cut color clarity     depth     table       price
 0.16632017 Premium     F    VVS1 0.5000000 0.3269231   Expensive
 0.03534304   Ideal     E     VS2 0.5444444 0.2500000 Inexpensive
 0.06444906    Good     G     SI2 0.5611111 0.2692308 Inexpensive
 0.14553015 Premium     G     VS2 0.5500000 0.2884615   Expensive
 0.06237006 Premium     F     VS2 0.5194444 0.3269231 Inexpensive
 0.02286902 Premium     F     VS2 0.5277778 0.3269231 Inexpensive
```

v

output:

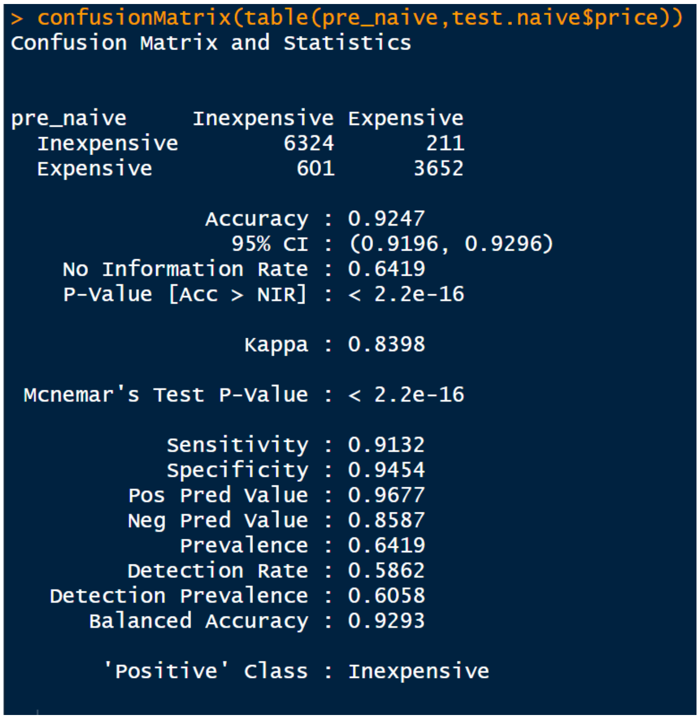


```
> library(e1071)
> naive <- naiveBayes(train.naive$price~. , data = train.naive)
> naive

Naive Bayes Classifier for Discrete Predictors

Call:
naiveBayes.default(x = X, y = Y, laplace = laplace)

A-priori probabilities:
Y
Inexpensive   Expensive
  0.6404338   0.3595662

Conditional probabilities:
             carat
Y                   [,1]       [,2]
  Inexpensive 0.06500432 0.04351133
  Expensive   0.22995684 0.07872061

             cut
Y                   Fair       Good      Ideal    Premium  Very Good
  Inexpensive 0.02789839 0.08865248 0.44510783 0.21989434 0.21844695
  Expensive   0.03280485 0.09706110 0.31908997 0.31419180 0.23685228

             color
Y                      D          E          F          G          H          I          J
  Inexpensive 0.14303807 0.21218700 0.18765378 0.20791721 0.12954118 0.08228398 0.03737878
  Expensive   0.09383862 0.12728796 0.15957721 0.21203919 0.19670018 0.13334622 0.07721062
```



```
             clarity
Y                     I1         IF        SI1        SI2        VS1        VS2       VVS1       VVS2
  Inexpensive 0.01429295 0.04396439 0.22155884 0.12986684 0.15888696 0.23035172 0.08999132 0.11108699
  Expensive   0.01282547 0.01688580 0.27597319 0.24419954 0.13527971 0.22409126 0.02777778 0.06296726

             depth
Y                  [,1]       [,2]
  Inexpensive 0.5207540 0.03861706
  Expensive   0.5210487 0.04150635

             table
Y                  [,1]       [,2]
  Inexpensive 0.2735346 0.04291354
  Expensive   0.2856311 0.04202906
```

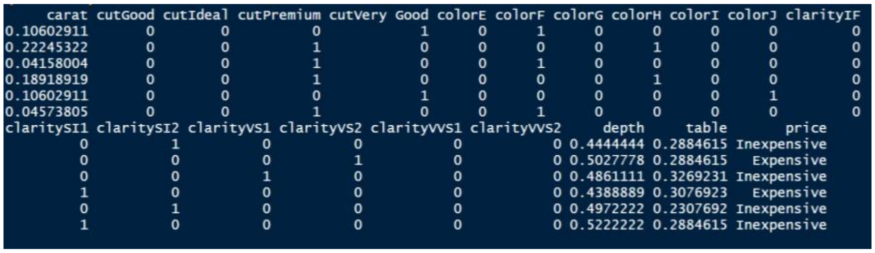
Accuracy:



```
> confusionMatrix(table(pre_naive,test.naive$price))
Confusion Matrix and Statistics


pre_naive     Inexpensive Expensive
  Inexpensive        6324       211
  Expensive           601      3652

               Accuracy : 0.9247
                 95% CI : (0.9196, 0.9296)
    No Information Rate : 0.6419
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.8398

 Mcnemar's Test P-Value : < 2.2e-16

            Sensitivity : 0.9132
            Specificity : 0.9454
         Pos Pred Value : 0.9677
         Neg Pred Value : 0.8587
             Prevalence : 0.6419
         Detection Rate : 0.5862
   Detection Prevalence : 0.6058
      Balanced Accuracy : 0.9293

       'Positive' Class : Inexpensive
```

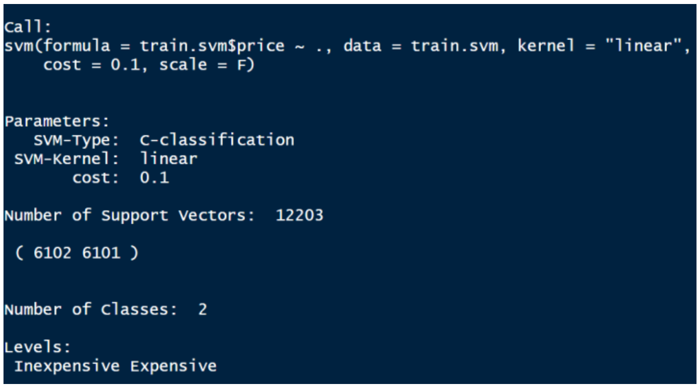
- Support vector machines:

  Preprocessing:



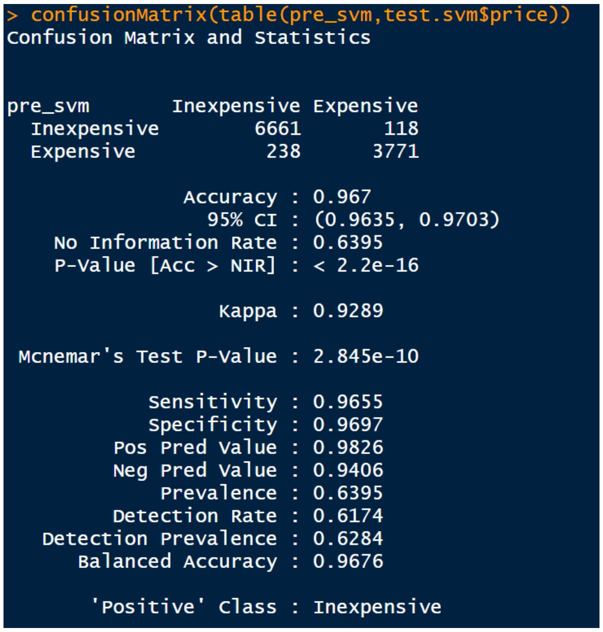
```
      carat cutGood cutIdeal cutPremium cutVery Good colorE colorF colorG colorH colorI colorJ clarityIF
0.10602911       0        0          0            1      0      1      0      0      0      0         0
0.22245322       0        0          1            0      0      0      0      1      0      0         0
0.04158004       0        0          1            0      0      1      0      0      0      0         0
0.18918919       0        0          1            0      0      0      0      1      0      0         0
0.10602911       0        0          0            1      0      0      0      0      0      1         0
0.04573805       0        0          1            0      0      1      0      0      0      0         0
claritySI1 claritySI2 clarityVS1 clarityVS2 clarityVVS1 clarityVVS2     depth     table       price
         0          1          0          0           0           0 0.4444444 0.2884615 Inexpensive
         0          0          0          1           0           0 0.5027778 0.2884615   Expensive
         0          0          1          0           0           0 0.4861111 0.3269231 Inexpensive
         1          0          0          0           0           0 0.4388889 0.3076923   Expensive
         0          1          0          0           0           0 0.4972222 0.2307692 Inexpensive
         1          0          0          0           0           0 0.5222222 0.2884615 Inexpensive
```

  Model:



```
Call:
svm(formula = train.svm$price ~ ., data = train.svm, kernel = "linear",
    cost = 0.1, scale = F)


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

Number of Support Vectors:  12203

 ( 6102 6101 )


Number of Classes:  2

Levels:
 Inexpensive Expensive
```

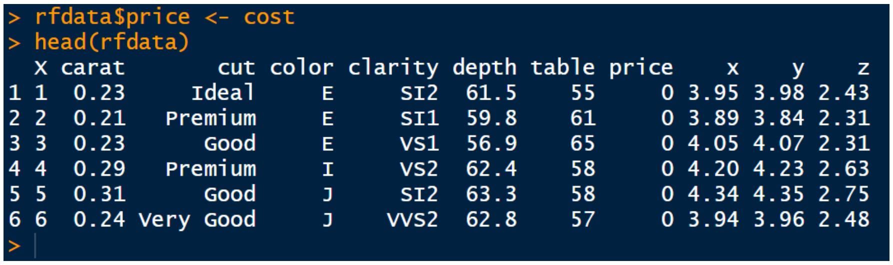
  Accuracy:



```
> confusionMatrix(table(pre_svm,test.svm$price))
Confusion Matrix and Statistics


pre_svm       Inexpensive Expensive
  Inexpensive        6661       118
  Expensive           238      3771

               Accuracy : 0.967
                 95% CI : (0.9635, 0.9703)
    No Information Rate : 0.6395
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.9289

 Mcnemar's Test P-Value : 2.845e-10

            Sensitivity : 0.9655
            Specificity : 0.9697
         Pos Pred Value : 0.9826
         Neg Pred Value : 0.9406
             Prevalence : 0.6395
         Detection Rate : 0.6174
   Detection Prevalence : 0.6284
      Balanced Accuracy : 0.9676

       'Positive' Class : Inexpensive
```

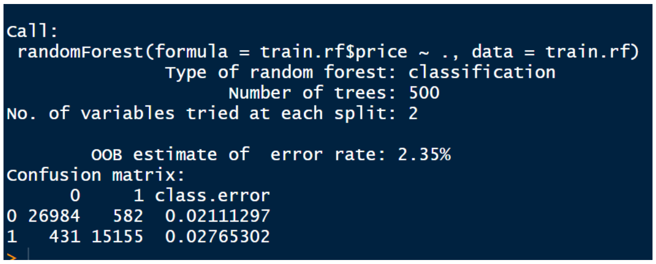
- Random forest:

  Preprocessing:



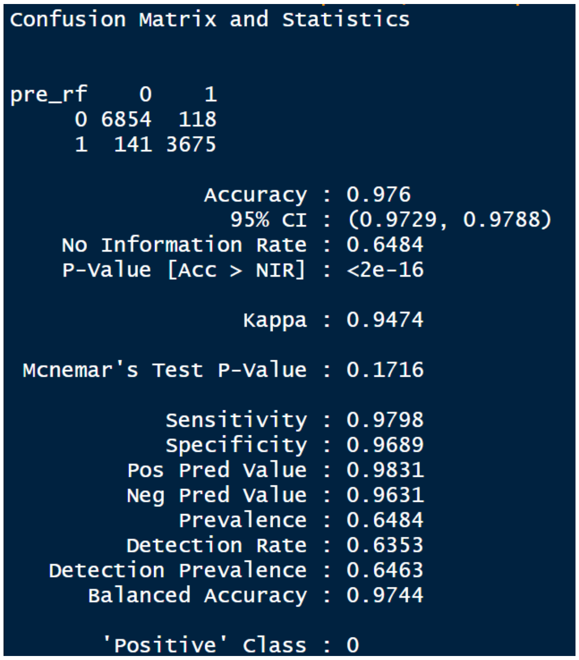
```
> rfdata$price <- cost
> head(rfdata)
  X carat       cut color clarity depth table price    x    y    z
1 1  0.23     Ideal     E     SI2  61.5    55     0 3.95 3.98 2.43
2 2  0.21   Premium     E     SI1  59.8    61     0 3.89 3.84 2.31
3 3  0.23      Good     E     VS1  56.9    65     0 4.05 4.07 2.31
4 4  0.29   Premium     I     VS2  62.4    58     0 4.20 4.23 2.63
5 5  0.31      Good     J     SI2  63.3    58     0 4.34 4.35 2.75
6 6  0.24 Very Good     J    VVS2  62.8    57     0 3.94 3.96 2.48
> 
```

Model:



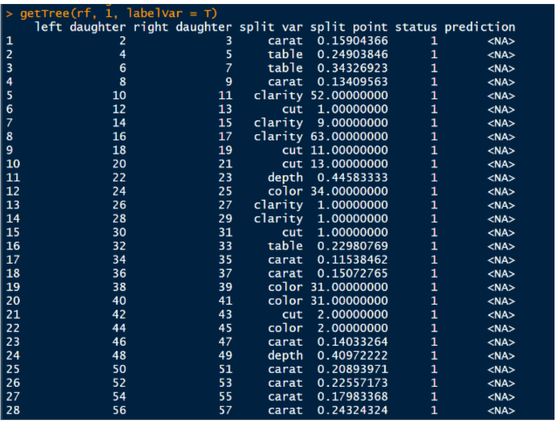
```
Call:
 randomForest(formula = train.rf$price ~ ., data = train.rf)
              Type of random forest: classification
                    Number of trees: 500
No. of variables tried at each split: 2

        OOB estimate of  error rate: 2.35%
Confusion matrix:
      0     1 class.error
0 26984   582  0.02111297
1   431 15155  0.02765302
>
```

Accuracy:



```
Confusion Matrix and Statistics


pre_rf    0    1
     0 6854  118
     1  141 3675

               Accuracy : 0.976
                 95% CI : (0.9729, 0.9788)
    No Information Rate : 0.6484
    P-Value [Acc > NIR] : <2e-16

                  Kappa : 0.9474

 Mcnemar's Test P-Value : 0.1716

            Sensitivity : 0.9798
            Specificity : 0.9689
         Pos Pred Value : 0.9831
         Neg Pred Value : 0.9631
             Prevalence : 0.6484
         Detection Rate : 0.6353
   Detection Prevalence : 0.6463
      Balanced Accuracy : 0.9744

       'Positive' Class : 0
```

Each forest evaluation:



c

```
> getTree(rf, 1, labelVar = T)
    left daughter right daughter split var split point status prediction
1               2              3     carat  0.15904366      1       <NA>
2               4              5     table  0.24903846      1       <NA>
3               6              7     table  0.34326923      1       <NA>
4               8              9     carat  0.13409563      1       <NA>
5              10             11   clarity 52.00000000      1       <NA>
6              12             13       cut  1.00000000      1       <NA>
7              14             15   clarity  9.00000000      1       <NA>
8              16             17   clarity 63.00000000      1       <NA>
9              18             19       cut 11.00000000      1       <NA>
10             20             21       cut 13.00000000      1       <NA>
11             22             23     depth  0.44583333      1       <NA>
12             24             25     color 34.00000000      1       <NA>
13             26             27   clarity  1.00000000      1       <NA>
14             28             29   clarity  1.00000000      1       <NA>
15             30             31       cut  1.00000000      1       <NA>
16             32             33     table  0.22980769      1       <NA>
17             34             35     carat  0.11538462      1       <NA>
18             36             37     carat  0.15072765      1       <NA>
19             38             39     color 31.00000000      1       <NA>
20             40             41     color 31.00000000      1       <NA>
21             42             43       cut  2.00000000      1       <NA>
22             44             45     color  2.00000000      1       <NA>
23             46             47     carat  0.14033264      1       <NA>
24             48             49     depth  0.40972222      1       <NA>
25             50             51     carat  0.20893971      1       <NA>
26             52             53     carat  0.22557173      1       <NA>
27             54             55     carat  0.17983368      1       <NA>
28             56             57     carat  0.24324324      1       <NA>
```