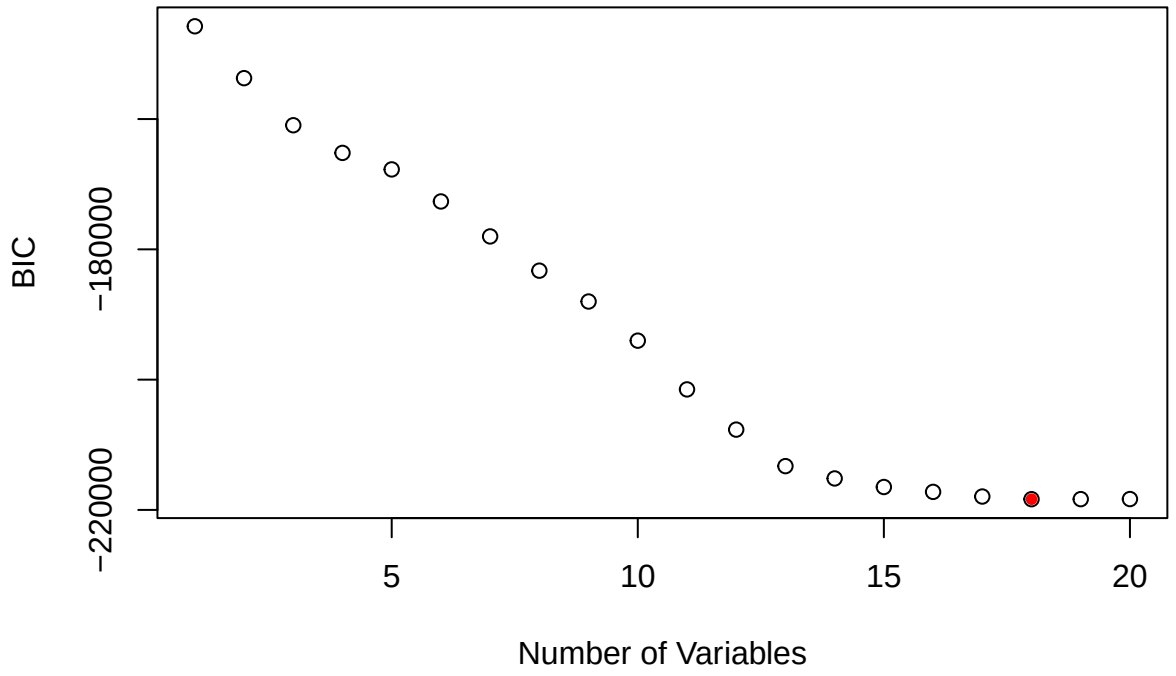
Multiple Linear Regression

We now introduce in the model other predictors which may be relevant to further increase the prediction accuracy of our simple model. For the moment, we will start from a multiple *linear* regression, therefore including no polynomial transformations of the variables. First thing to do is to perform feature selection, so that we are sure to include in our model only relevant predictors. We proceed in this task using best subset selection, selecting the optimal model according to the Schwartz's Information Criterion, whose values for different numbers of variables are plotted below.



The BIC is suggesting us to use 18 variables. These are:

```
## (Intercept)  log(carat)        cut2        cut3        cut4        cut5
##  7.34593968  1.88373072  0.07994385  0.11710146  0.13924527  0.16111046
##    clarity2    clarity3    clarity4    clarity5    clarity6    clarity7
##  0.42791565  0.59300280  0.74220198  0.81228490  0.94731388  1.01878750
##    clarity8      color2      color3      color4      color5      color6
##  1.11377385  0.13841145  0.25992235  0.35061184  0.41641036  0.45668723
##      color7
##  0.51098878
```

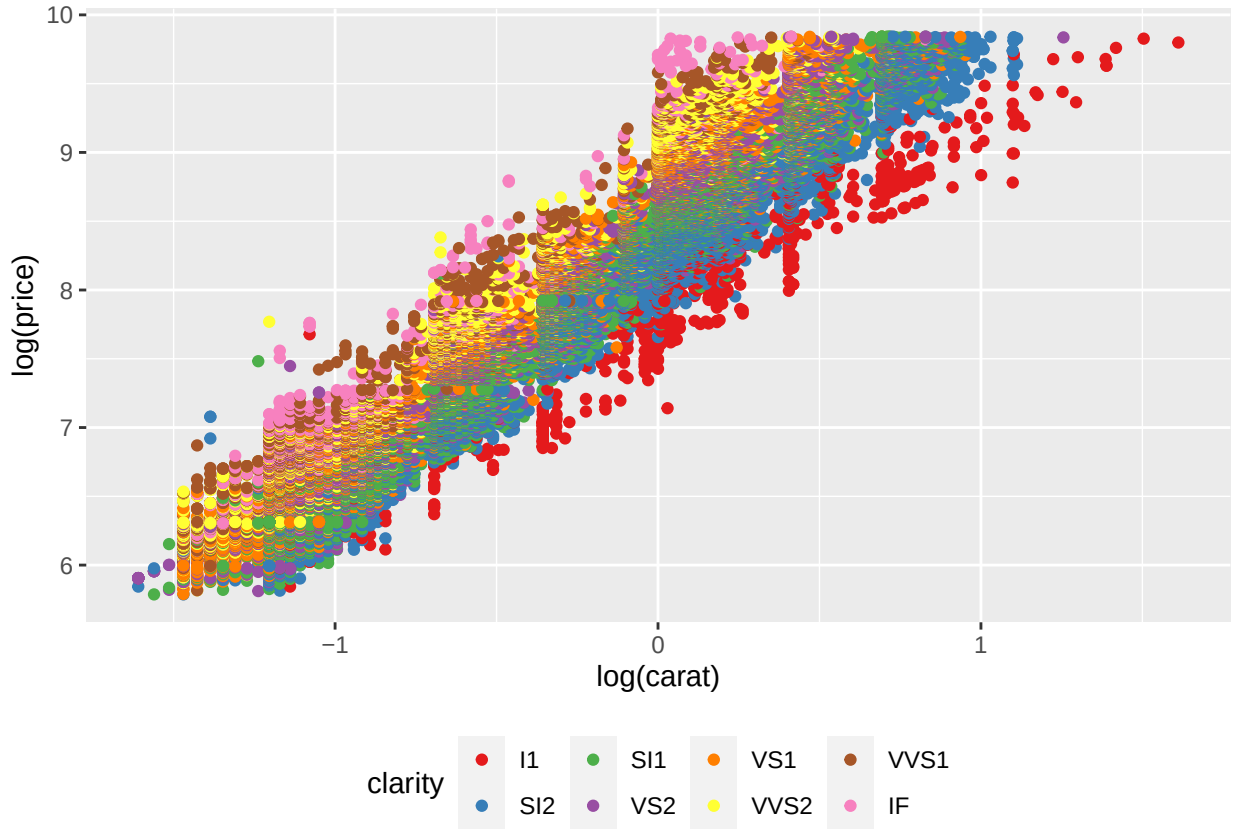
As we expected from the descriptive statistics, `table` and `depth` are excluded from the model as irrelevant predictors. We therefore proceed in estimating a linear model with all the categorical variables and the logarithm of `carat` as predictors.

```
##
## Call:
```

```
## lm(formula = log(price) ~ log(carat) + +cut + clarity + color,
##     data = dta, subset = train)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.80764 -0.08685  0.00003  0.08397  1.45752
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 7.343747   0.007320 1003.30   <2e-16 ***
## log(carat)  1.882194   0.001351 1392.94   <2e-16 ***
## cut2        0.079665   0.004672   17.05   <2e-16 ***
## cut3        0.116479   0.004350   26.78   <2e-16 ***
## cut4        0.139225   0.004299   32.38   <2e-16 ***
## cut5        0.160584   0.004264   37.66   <2e-16 ***
## clarity2    0.432788   0.006279   68.93   <2e-16 ***
## clarity3    0.596076   0.006241   95.51   <2e-16 ***
## clarity4    0.746794   0.006276  118.99   <2e-16 ***
## clarity5    0.816715   0.006369  128.23   <2e-16 ***
## clarity6    0.952661   0.006560  145.23   <2e-16 ***
## clarity7    1.023414   0.006753  151.55   <2e-16 ***
## clarity8    1.117465   0.007292  153.24   <2e-16 ***
## color2      0.136472   0.003764   36.26   <2e-16 ***
## color3      0.258112   0.003548   72.75   <2e-16 ***
## color4      0.347757   0.003449  100.82   <2e-16 ***
## color5      0.414692   0.003515  117.97   <2e-16 ***
## color6      0.453872   0.003533  128.47   <2e-16 ***
## color7      0.509474   0.003702  137.63   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1344 on 37736 degrees of freedom
## Multiple R-squared:  0.9825, Adjusted R-squared:  0.9825
## F-statistic: 1.176e+05 on 18 and 37736 DF,  p-value: < 2.2e-16

## [1] "MSE on the training set: 0.01805"
```

Notice how *all* the regression's coefficients are extremely significant, while `carat`'s coefficient has slightly increased: according to this model, a 1% increase in carat leads to a 1.88% increase in price, *keeping all the other factors fixed*. For what regards the categorical variables, notice how all the coefficients are positive; not only that, but they are strictly increasing as we move through the classes of each variable. That is because these variables are ordered from the worst level (baseline) to the best, and as we go towards the best quality cut (or clarity or color) diamonds, the price increases. Among the categorical variables, the coefficients with the largest magnitude are the ones of `clarity`: holding everything else constant, with respect to the baseline specification (lower-quality clarity) the best-quality clarity diamond has a price which is expected to be $e^{1.11} - 1 \sim 2$ times larger. Finally, the R^2 is significantly higher than the one of the simple models and has reached the very large value of 98% on the training set. Also the training MSE is much smaller than before, meaning that introducing the categorical variable had a substantial positive effect on the model accuracy. This may looks surprising, since according to the descriptive statistics the categorical variables seemed to carry no information about the response. In order to better understand the role played by these qualitative predictors, we plot the relationship between `price`, `carat` and `clarity`; however, each of the other two categorical predictors showed similar graphical results.

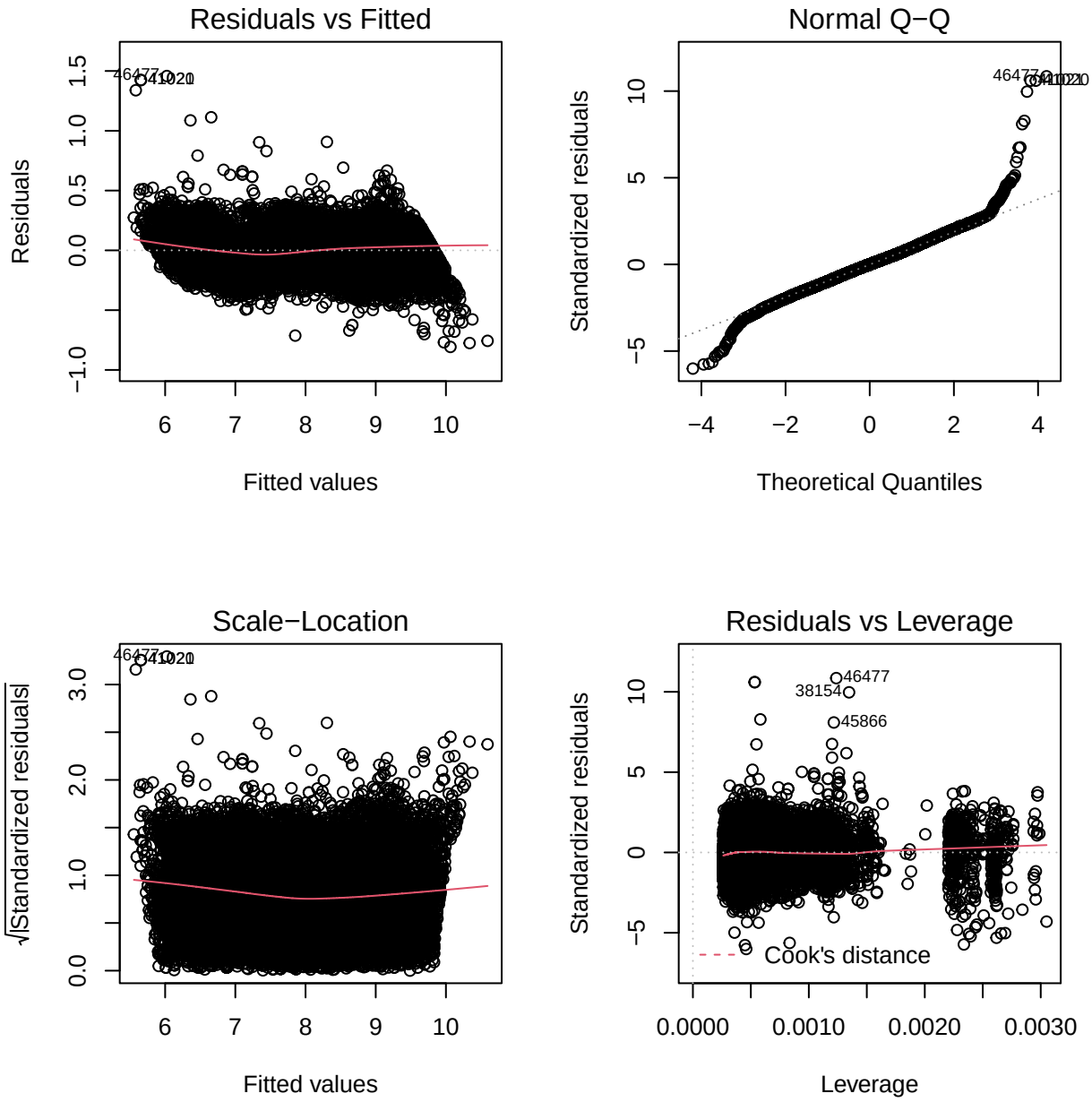


Basically, the categorical variables let us capture the price variability which could not be explained by `carat`. In fact, given a certain level of `carat`, notice how low-clarity diamonds (I1 class - red points) are on the lower part of the price range, while high-clarity diamonds (IF class- pink points) are on the upper bound. In the middle, colors are ordered symmetrically to the legend, where classes are ordered. Indeed, the current model predicts the diamond's price using a straight line always with the same slope, but with different intercept depending on the categorical variables' value of the considered diamond.

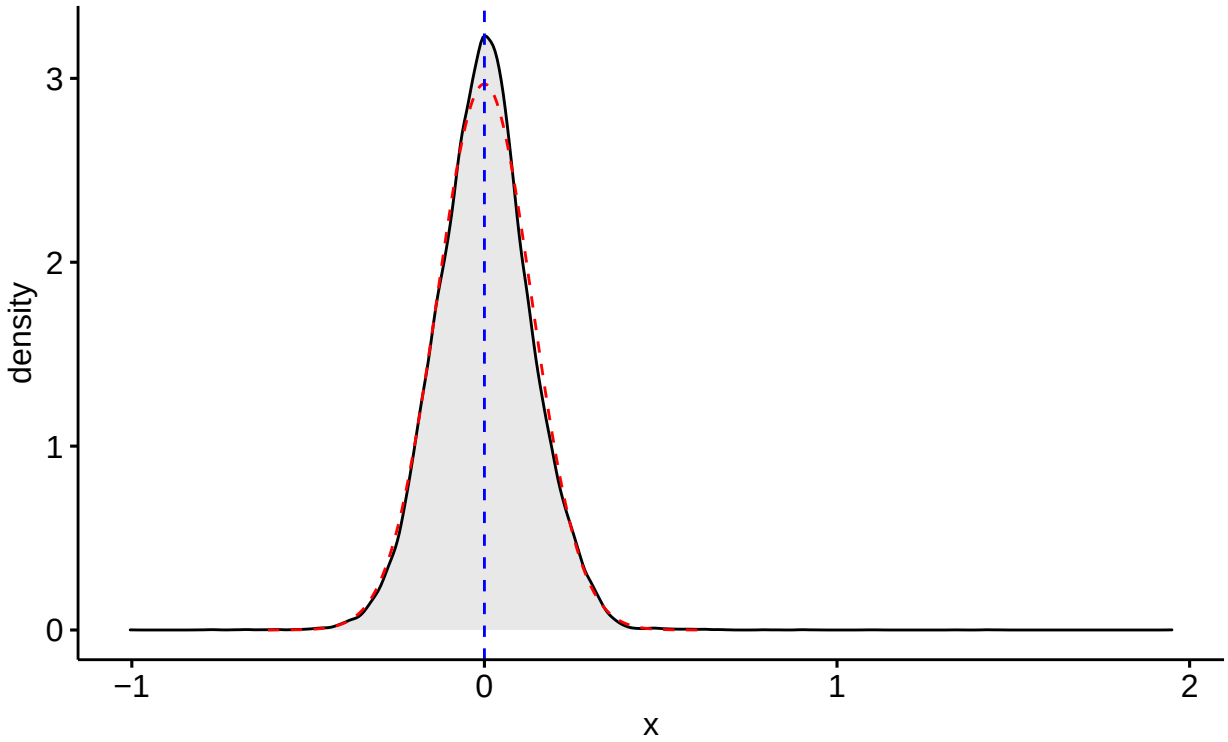
Let's now turn to the model diagnostic. Since we are in a multiple linear regression setting, it is better to check the Variance Inflation Factors to adequately check for multicollinearity.

```
##                GVIF Df GVIF^(1/(2*Df))
## log(carat) 1.305438  1        1.142558
## cut        1.100909  4        1.012089
## clarity    1.315889  7        1.019802
## color      1.137131  6        1.010767
```

The values of the VIF are very close to 1 (which is the minimum possible value) for all the predictors, and also after taking the square roots, all the values are well far from the threshold of 2, meaning that all our predictors carry independent information from one another. We now turn to the diagnostic checks used also in the other simple settings.



The `Residuals vs Fitted` plot is pretty similar to the one of the simple linear regression, highlighting once again that the linear model is not suitable in predicting small and large level of price. The `Scale-Location` plot shows a little trend, but overall the errors seem to show a constant variance. For what concerns the `Residuals vs Leverage` plot, the vast majority of the observations lie within the interval of ± 3 standardized residuals. There are also many outliers, but we do not feel like removing them, since those large residuals may be due to the simplicity of the linear model. Indeed, the `Normal Q-Q` plot shows very large residuals for some observations, but overall it seems like the normality assumption on the error terms is met, as confirmed by the plot below.



Finally, let's take a look at the performance of the model on an independent test set.

```
## [1] "MSE on the test set: 0.01756"
```

Performances on the test set and on the training set are very similar; in both cases, the test MSE is approximately 4 times smaller than the one obtained with the simplest models.

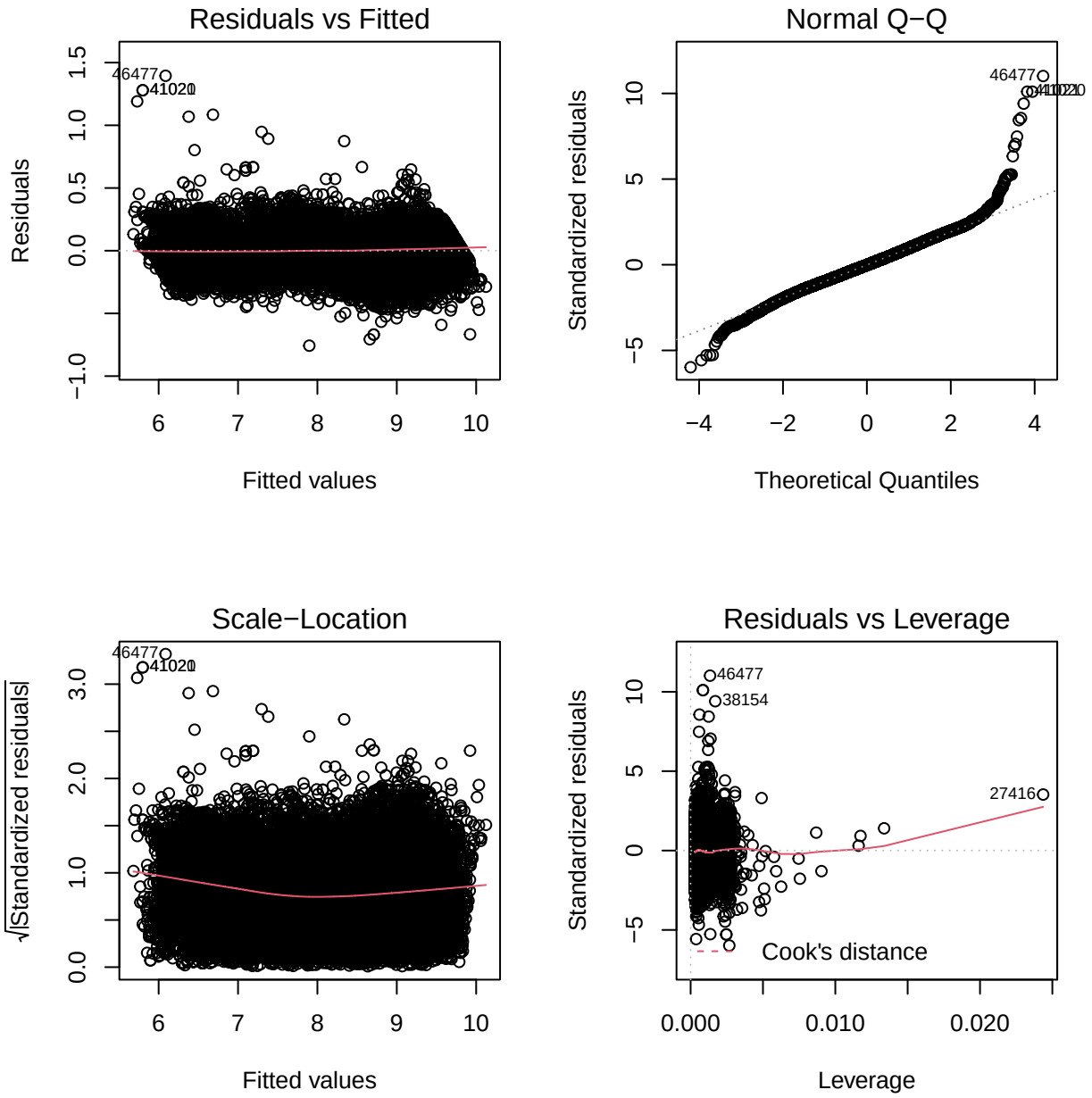
Multiple Polynomial Regression

In the previous paragraph, we saw `table` and `depth` are completely irrelevant for predicting our target variable. Since this hypothesis has a strong evidence also in the descriptive statistics, we will ignore those two variables in this last parametric model. Therefore, we will estimate a multiple polynomial regression composed of the orthogonal cubic polynomial of `log(carat)` plus the categorical variables. The resulting model's summary is shown below.

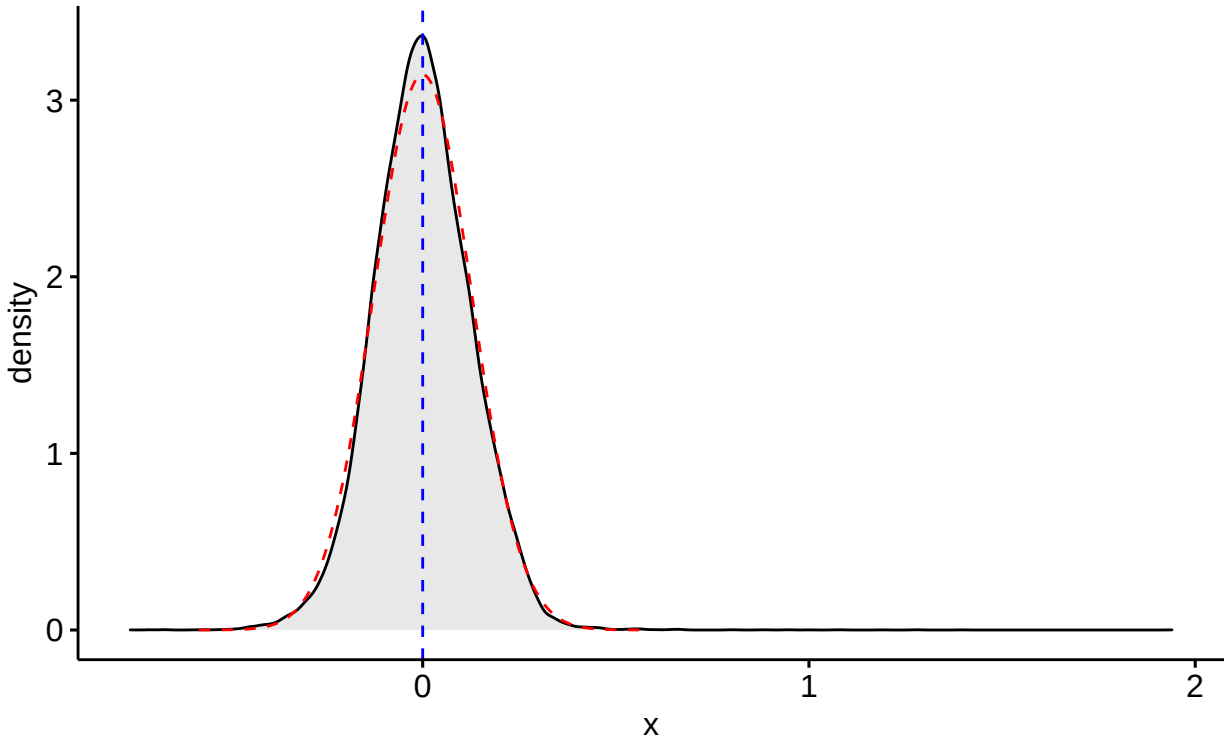
```
##
## Call:
## lm(formula = log(price) ~ poly(log(carat), 3) + cut + clarity +
##     color, data = dta, subset = train)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.75734 -0.08319 -0.00188  0.08115  1.39492
##
## Coefficients:
##                          Estimate Std. Error t value Pr(>|t|)
## (Intercept)              6.620552   0.007002  945.57   <2e-16 ***
## poly(log(carat), 3)1 255.452213   0.173169 1475.16   <2e-16 ***
## poly(log(carat), 3)2   2.705656   0.155126   17.44   <2e-16 ***
## poly(log(carat), 3)3 -10.139260   0.153028  -66.26   <2e-16 ***
## cut2                     0.075102   0.004407   17.04   <2e-16 ***
## cut3                     0.107057   0.004105   26.08   <2e-16 ***
## cut4                     0.134919   0.004058   33.25   <2e-16 ***
## cut5                     0.161846   0.004022   40.24   <2e-16 ***
## clarity2                 0.420905   0.005932   70.96   <2e-16 ***
## clarity3                 0.586006   0.005904   99.25   <2e-16 ***
## clarity4                 0.734188   0.005933  123.74   <2e-16 ***
## clarity5                 0.803357   0.006022  133.40   <2e-16 ***
## clarity6                 0.929245   0.006202  149.84   <2e-16 ***
## clarity7                 1.001659   0.006380  157.00   <2e-16 ***
## clarity8                 1.097951   0.006886  159.46   <2e-16 ***
## color2                   0.133655   0.003552   37.63   <2e-16 ***
## color3                   0.249880   0.003356   74.45   <2e-16 ***
## color4                   0.344490   0.003273  105.25   <2e-16 ***
## color5                   0.410454   0.003342  122.80   <2e-16 ***
## color6                   0.451161   0.003354  134.52   <2e-16 ***
## color7                   0.509270   0.003513  144.99   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1267 on 37734 degrees of freedom
## Multiple R-squared:  0.9844, Adjusted R-squared:  0.9844
## F-statistic: 1.193e+05 on 20 and 37734 DF,  p-value: < 2.2e-16
```

```
## [1] "MSE on the training set: 0.01605"
```

Once again, including the qualitative predictors led to an improvement: the adjusted R^2 increased by 0.2%, while the training MSE decreased by 10%. All the coefficients are extremely significant; the ones of the qualitative predictors are basically the same as they were in the multiple linear regression, so the interpretation does not change. We can therefore turn to the diagnostic of the model.



The `Scale-Location` plot is fine for what concerns homoskedasticity. The `Residuals vs Fitted` plot shows that there is no pattern left in the residuals; however, even the most complex model so far does not behave well at the boundaries of `price`, even though the trend is less pronounced than in linear models. The `Residuals vs Leverage` plot, as the Q-Q plot, shows that for few observations we have some very large residuals: since we are dealing with a very large data set, it is possible to observe (relatively) few large errors. However, as shown in the plot below, the normality assumption of the residuals holds.



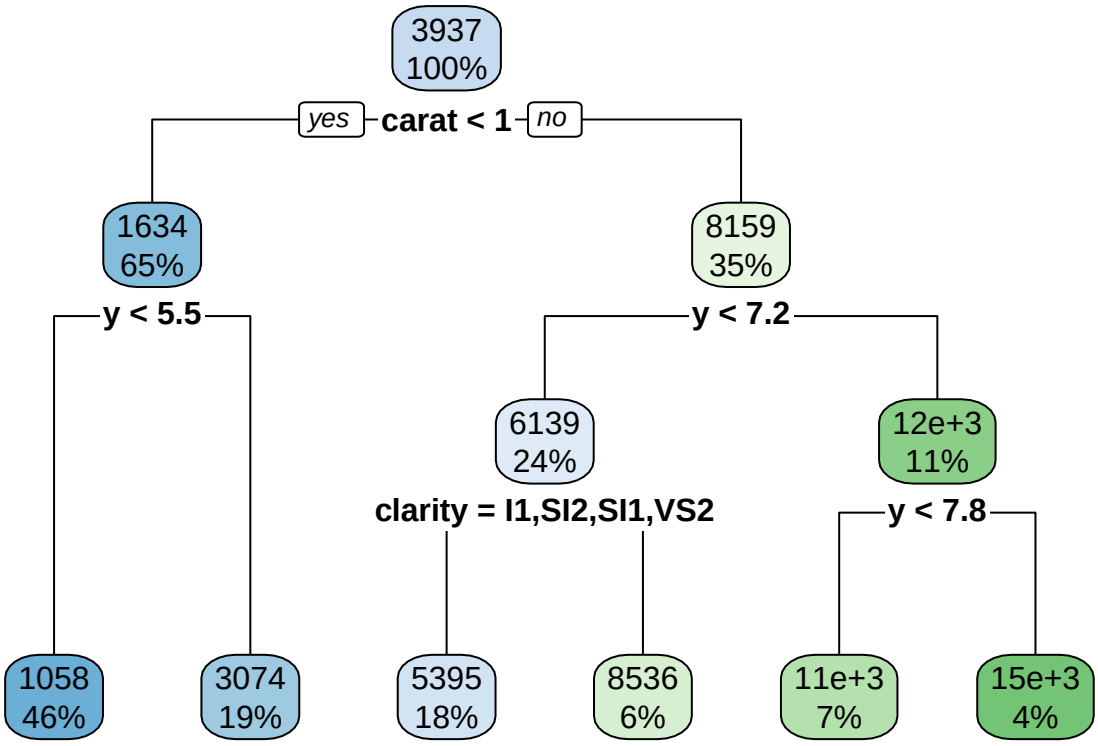
Finally, let's take a look at the performance of the model on an independent test set.

```
## [1] "MSE on the test set: 0.01574"
```

The test MSE is the lowest obtained so far, suggesting that the multiple polynomial regression is the best parametric approach among the four.

Regression tree

We now turn to non-parametric methods which, differently from before, make no assumption about the form of the relationship between the response and the predictors. We will start considering just a simple tree, which is the most interpretable approach among the non-parametric tree-based methods. What the tree predictor actually does, is to split the predictors' space into different region, and then predict a new observation using the mean of the response of the training observations belonging to the region in which the new observation falls. The splitting rules can be easily represented through a tree, hence this is where the name of this class of models come from. We therefore apply the recursive binary splitting to build a tree using observations from the same training set used for parametric methods. The resulting tree is plotted below.




```
## [1] "Variables importance:"
```

```
##   carat       y       x       z clarity   color   table   depth
##  506.83  495.85  492.09  476.93   50.72   14.77    0.38    0.09
```

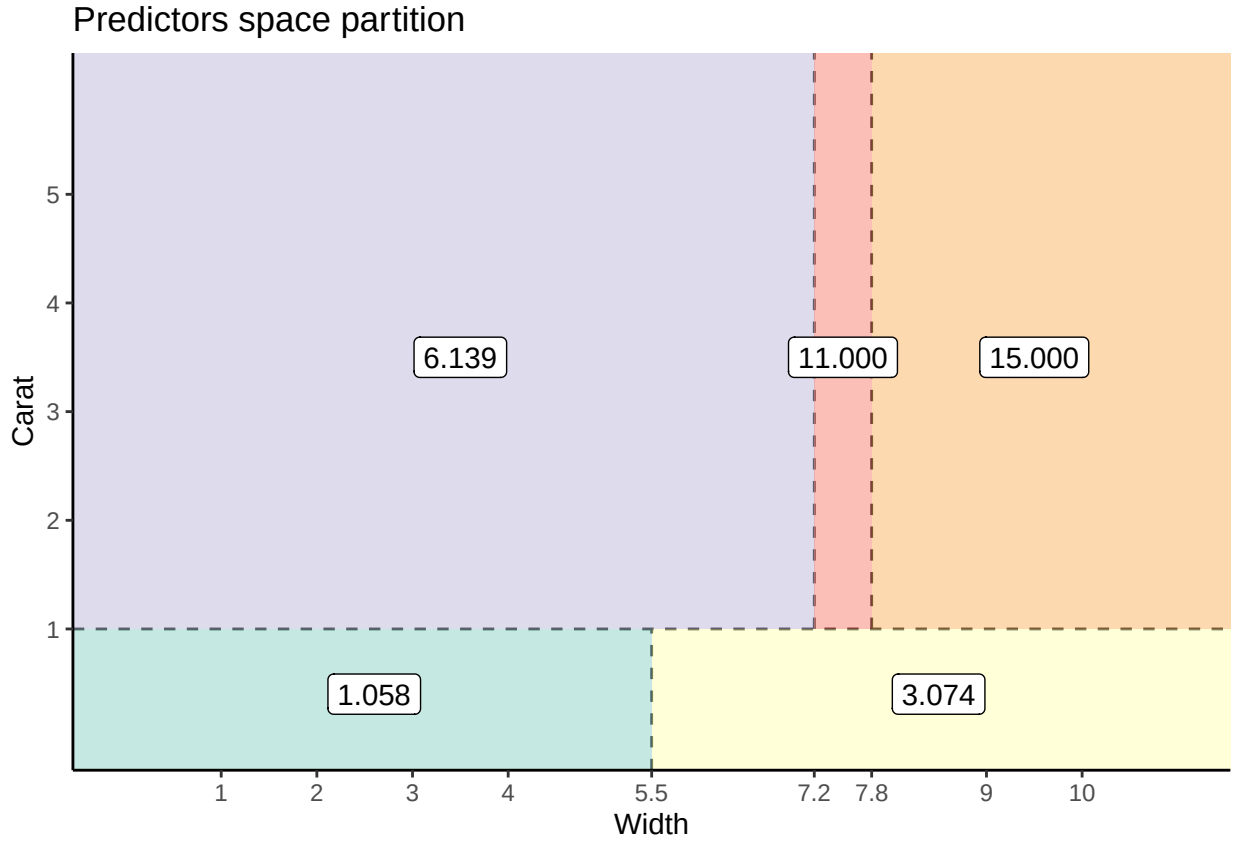
The tree is quite simple, and so is the interpretation. As one could expect, diamonds whose carat is less than 1 and whose width is less than 5.5 mm, get the lower price prediction (1058 USD). On the contrary, diamonds whose carat is more than 1 and whose width is more than 7.8 mm have the higher price prediction, around 15000 USD. The importance measures confirm once again the idea that the most relevant factors in determining a diamond's price are its weight (measured by `carat`) and its dimensions (`x`, `y` and `z`). Finally, also `clarity` (which was the categorical variable with the largest magnitude in the parametric models) plays a relevant role according to the tree. `table` and `depth` are once again not relevant in price prediction. Let's now take a closer look at the performance of the tree in the training set.

```
## [1] "Pseudo R-squared: 0.881069128815252"
```

```
## [1] "Training RMSE: 1378.60701796793"
```

The performances are comparable with our first simple linear regression model: the tree is able to explain 88% of price variability [1] and predicts the diamonds' price with an average error of 1378 USD on the training set.

If we ignore the node related to `clarity`, we can plot the partition defined by the tree in the bi-dimensional space given by `carat` and `y`, which we rename width.



Notice how the lower-left rectangle is showing the lowest level of price, while the top-right rectangle has the largest prediction for price. Let's now see how the model behaves on an independent test set.

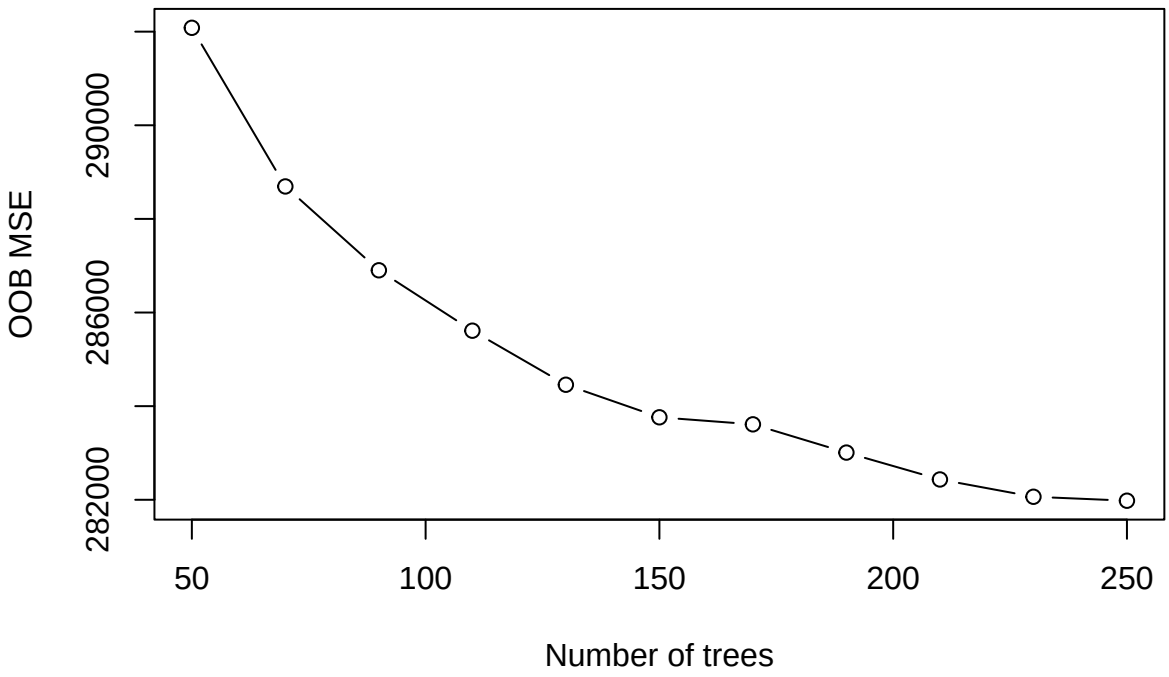
```
## [1] "Test RMSE: 1386.02986057605"
```

The test error made by the tree is slightly greater than its training counterpart. More precisely, the tree predicts the diamond's price with an average error of 1386 USD on the test set. Let's see if by applying ensemble methods we can get better prediction accuracy.

[1] The Pseudo R^2 is computed as 1 - RSS/TSS. Even though this may not be a proper metric for a regression tree, it was computed to gain some intuition about the explanatory power of the model and compare it with the one of the parametric regressions.

Bagging

We have seen tree's prediction accuracy is not so good; moreover, trees suffer from high variability: if we had grown another tree on a different training set, we would have obtained a rather different partition of the features space. A first step towards a more stable and a better-performing model is to consider an *ensemble* of trees, rather than a single one. Indeed, bagging involves growing B different trees on B randomly bootstrapped versions of the original training set, and then take the average of the single predictions. The bootstrapped training sets have the same size as the original one; it can be proven that each bootstrapped training set contains on average only 2/3 of the observations of the original data set. The remaining 1/3 observations are also known as *out-of-bag* (OOB) observations. Since these observations are not used for training purposes, we can use them to test the model and obtain a reliable estimate of the test error. We use OOB MSE estimate to select a reasonable number of trees for our bagged forest.



The elbow of the curve is around 150 trees; after that level we see some improvement, but this is mainly an effect of the scale on the vertical axis. The stabilization after 150 trees is even more evident if we consider the RMSE rather than the MSE.

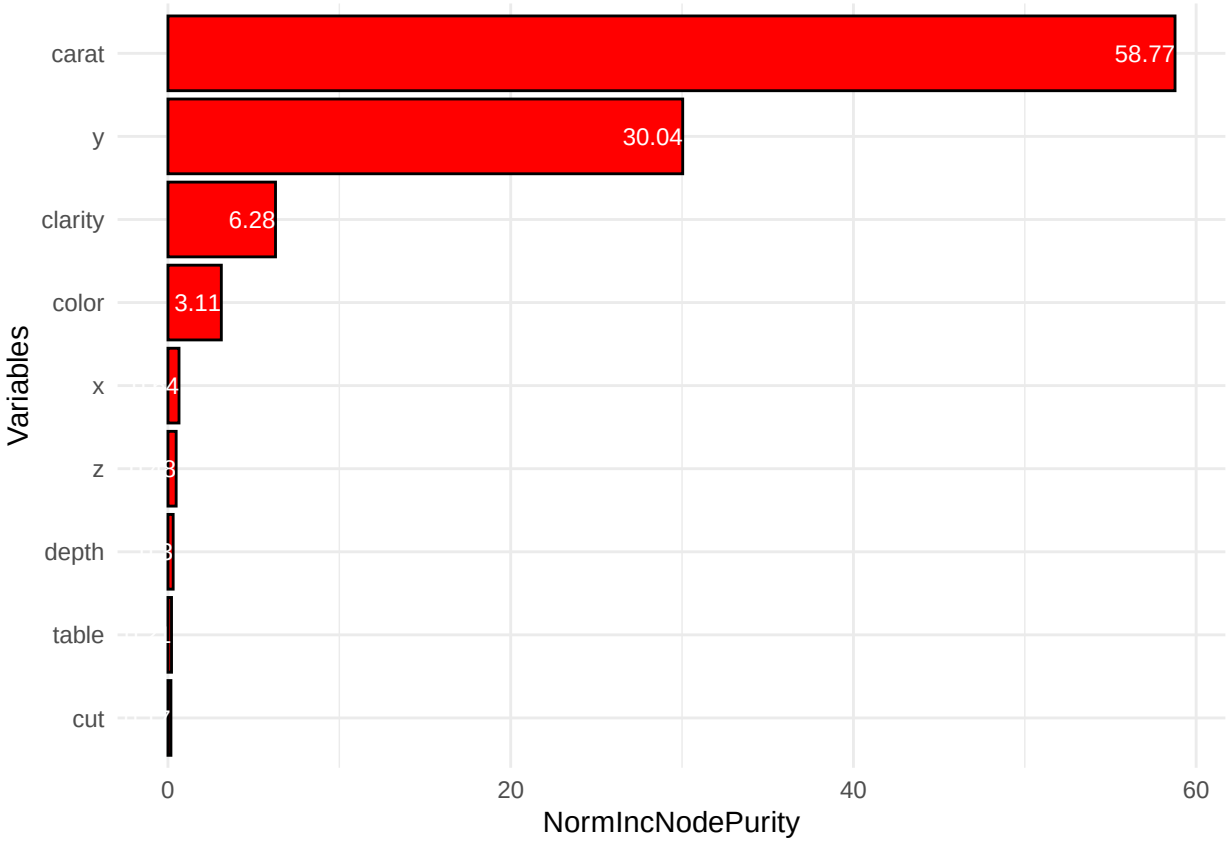
```
##       50       70       90      110      130      150      170      190
## 540.4462 537.3022 535.6327 534.4269 533.3455 532.6933 532.5522 531.9854
##      210      230      250
## 531.4452 531.0975 531.0183
```

We therefore grow a forest of 150 bagged trees. Some information about the model performance on the training set are shown below.

```
## [1] "Pseudo R-squared: 0.99642129048797"
```

```
## [1] "Training RMSE: 238.423729572804"
```

The R^2 on the training set is extremely high, almost reaching the maximum value of 1. Also the training RMSE is very low: on the training set, the bagged forest predicts the diamonds' price with an average error of 238 USD. This is a very good performance, considering that diamonds' price variation is very wide. Although bagged forests are not as interpretable as single trees, the prediction accuracy is much better, and we can still count on some measures of variables importance. The following plot shows such measures in our case: each split involves a variable, and for each split we can compute the reduction in RSS associated to it. Since in our case the RSS is of the order of tens of thousands, we normalized the importance of each variable so that they sum up to 100 to have more readable numbers.



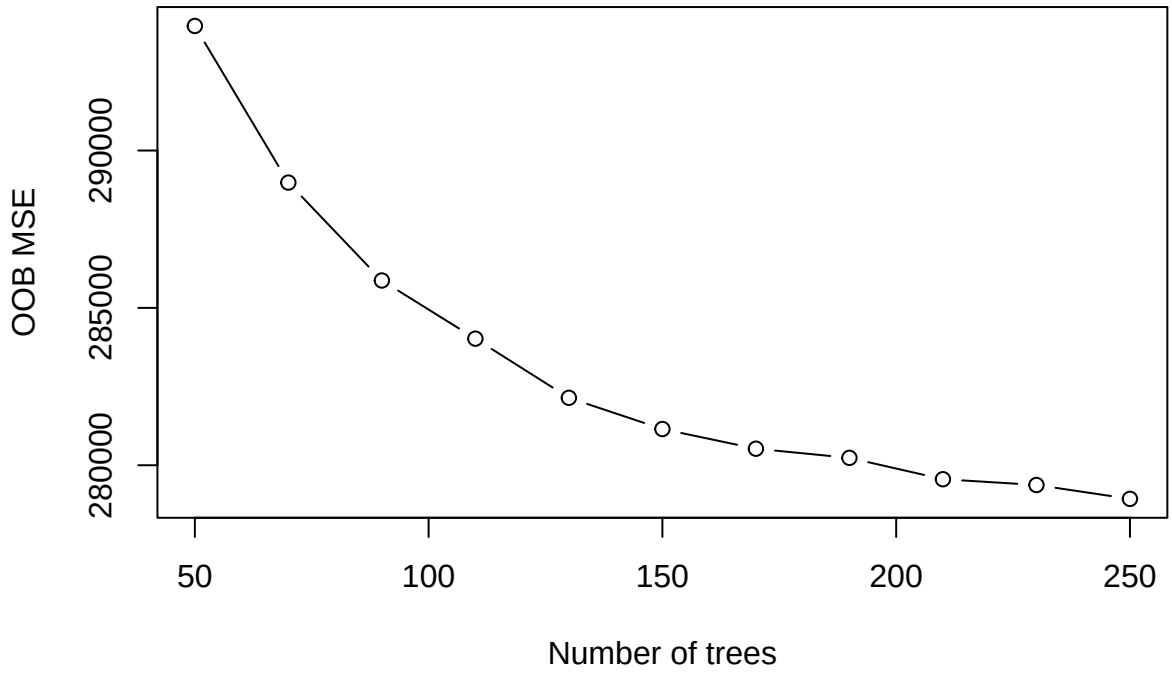
`carat` is once again the most important variable, as highlighted in all the models so far. The first and the second variable in terms of importance were included in the single tree model as well. After `color`, variables' importance become negligible. Coherent with the parametric models, `depth` and `table` are not really important, while, differently from before, `cut` is the least important variable.

```
## [1] "Test RMSE: 541.592974100532"
```

The test RMSE is more than double of its training counterpart. Nonetheless, we can say we are pretty satisfied with the performance of the model. Moreover, we know this difference cannot be caused by overfitting, since bagging does not overfit even with large values of B.

Random forest

Bagging suffers from a well-known drawback: if among the explanatory variables we have a very strong predictor, the bagged trees will be very similar to one another, because all the trees will consider this predictor for most of their splits. This results in highly correlated predictions, which are then averaged; however, averaging correlated quantities leads to an increase in the variance, rather than to its decrease. Random forest algorithm allows us to solve this issue by considering at each split only a random subset of the available predictors, so that in the end the trees will be different from one another and we will therefore obtain uncorrelated predictions, ultimately leading to better accuracy performance. As done for bagging, we select the number of trees using OOB MSE estimate.



Probably a random forest with 200 trees will be the optimal choice; however, to make a comparison with the bagged forest we grow a random forest of just 150 trees. In the end, there is not much difference in term of RMSE, as shown below.

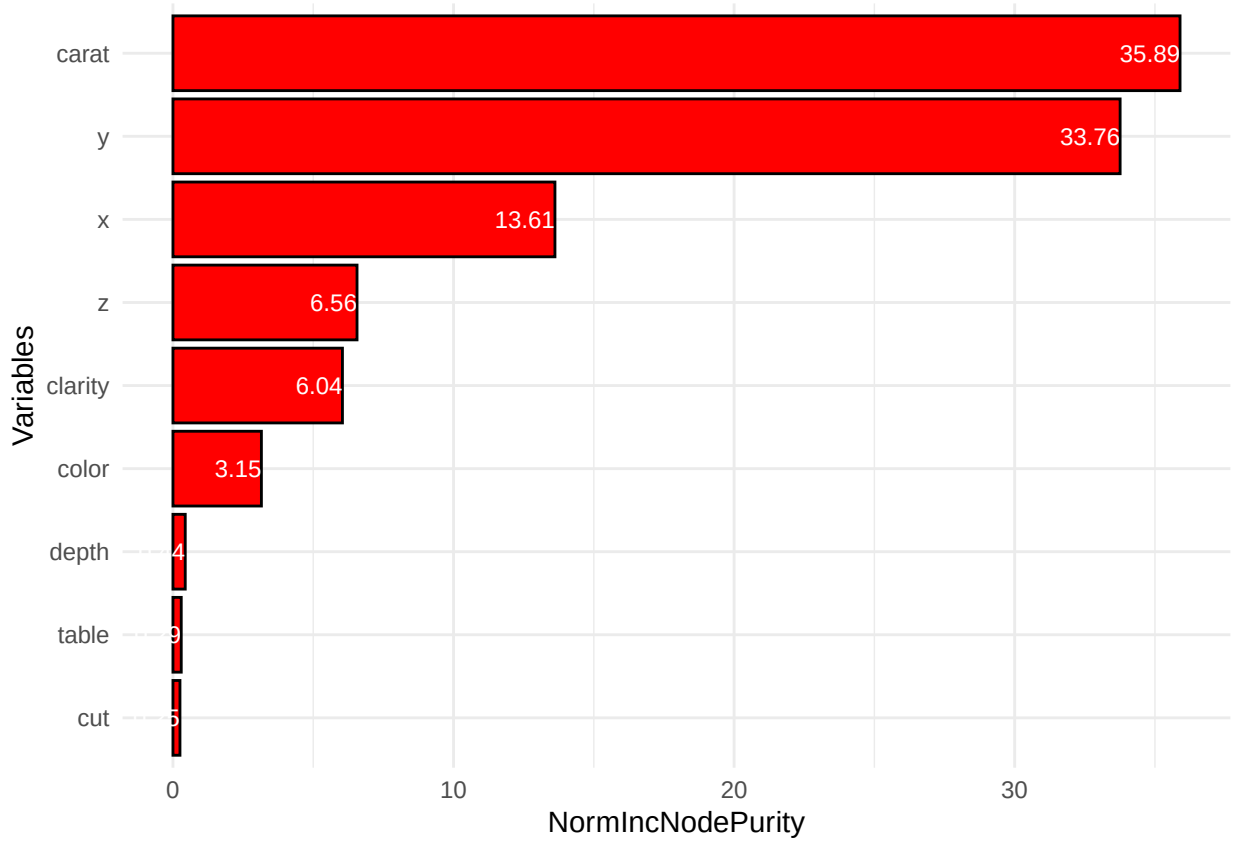
```
##       50       70       90      110      130      150      170      190
## 542.1790 537.5701 534.6684 532.9373 531.1708 530.2365 529.6460 529.3714
##      210      230      250
## 528.7302 528.5564 528.1396
```

```
## [1] "Model summary"
```

```
## [1] "Pseudo R-squared: 0.996065453898035"
```

```
## [1] "Training RMSE: 249.996295116017"
```

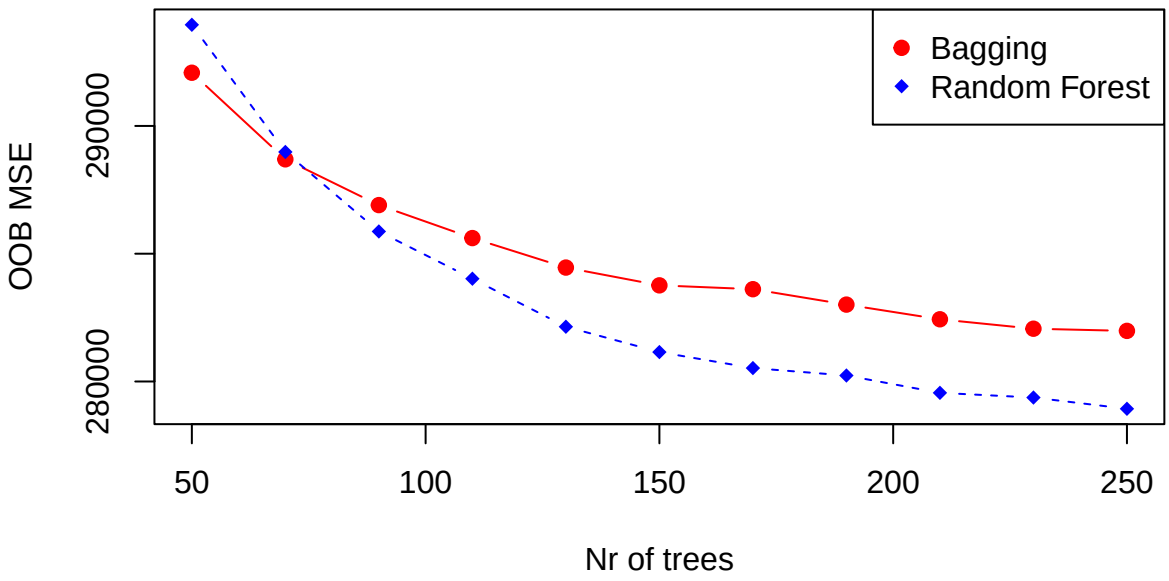
The R^2 is slightly lower and the training RMSE is slightly larger than the one we obtained with bagging. However, we should not really worry about those training metrics, since what really matters are the performances on the test set. Before looking at them, we show a plot summarising the variables' importance for the random forest.



`carat` is still the most important variable, followed closely by `y`. Differently from bagging, `x` and `z` gained much importance, coherent with the fact that they are a sort of surrogate for the first two variables. `clarity` is still the most important predictor among the categorical ones, while again `depth` and `table` are really not relevant. Finally, it is time to consider how the model behaves on an independent test set.

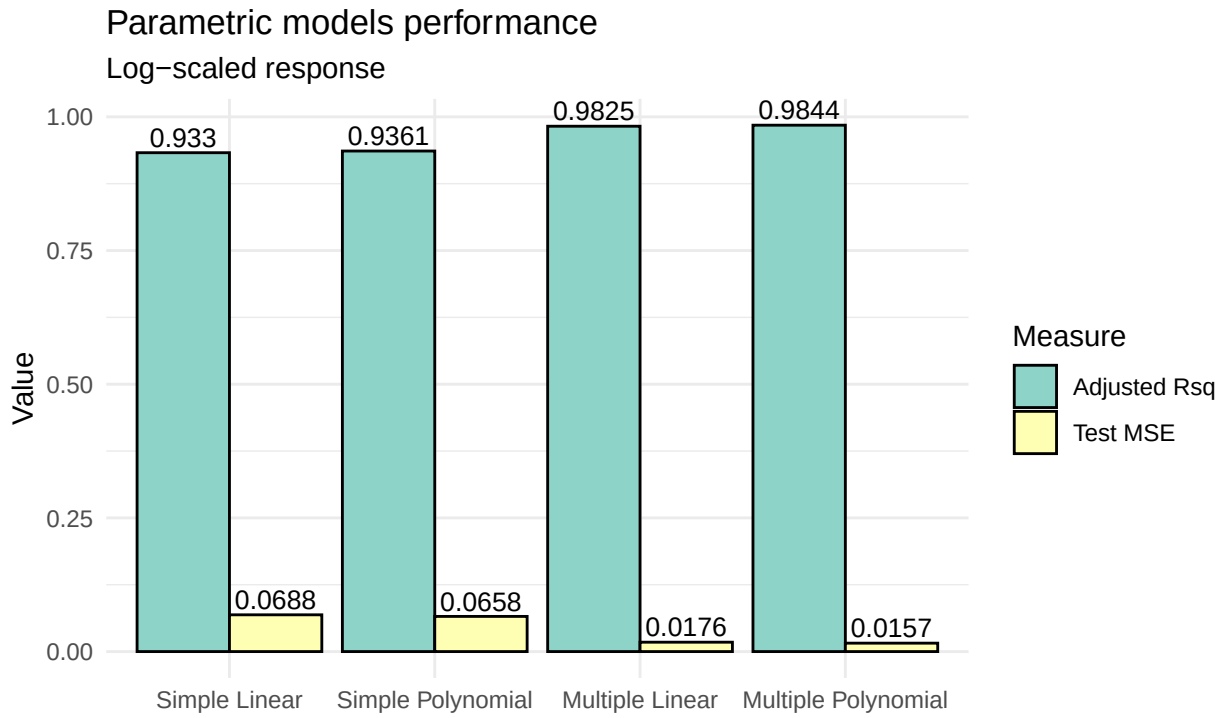
```
## [1] "Test RMSE: 536.210554058875"
```

Random forest managed to get some improvement over bagging, even though bagging performance were already very good. Notice that this is coherent with the fact that also OOB MSE estimates were lower for random forest, as shown by the plot below.

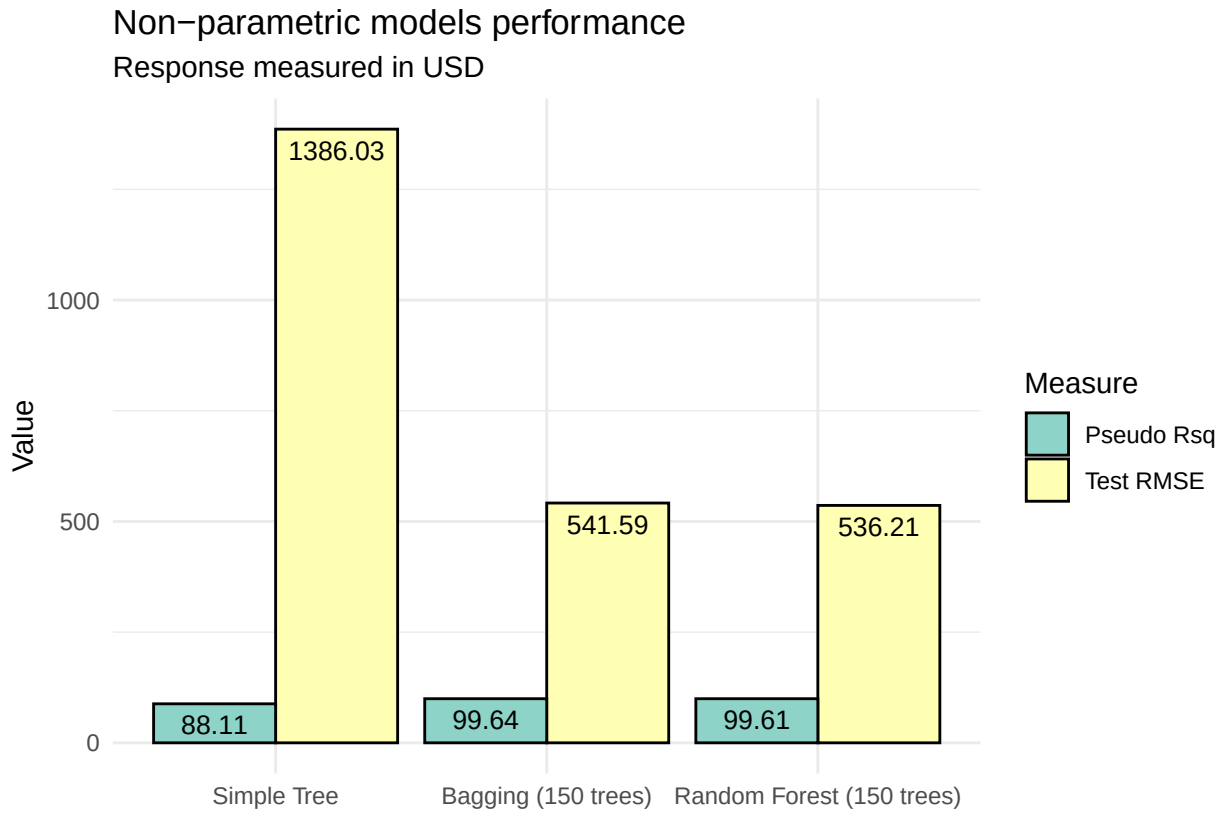


Conclusion

We now summarize the performances of the models built so far, starting from the parametric ones.



Here the models were ordered according to their complexity, and we can see how the adjusted R^2 increases constantly. Notice how the simple and the multiple models have pairwise similar values, both for the R^2 and for the test MSE; on the contrary, models involving multiple predictors perform much better than the ones including just `carat`. Before discussing the findings about non-parametric methods, it is worth recalling that, while linear and polynomial regressions were developed using log-transformed `price`, tree-based methods were using non-transformed target variable, so the numbers are really not comparable. Moreover, for the sake of a clear representation, the pseudo R^2 shown below is to be interpreted as a percentage.



As one could expect, the simple tree is really not competitive in terms of prediction with ensemble methods. The results about bagging and random forest are more interesting: while the former performs better on the training set (as shown by the pseudo R^2), the latter shows a smaller test RMSE. However, the two methods are really similar, and random forest prevails only by a little.

Key findings

All the model analyzed in this paper pointed out the importance of carats in predicting our target variable, so we may say this is the most relevant factor in determining a diamond's price, together with the dimensions of the diamond in terms of width, length, and depth (the `z` variable). However, weight and size are not the only relevant factors, as an important role is also played by the clarity of a diamond, followed by the color and the cut. On the contrary, the table and the depth (expressed as a percentage of the other two dimensions) are not relevant for a diamond's price.

Accuracy-interpretability trade-off. If we were interested only in accuracy, random forest would be the optimal choice among the analyzed algorithms. However, the model yielding the optimal trade-off between interpretability and prediction accuracy is probably the multiple linear regression. That model in fact allows us to draw some nice inferential hypotheses, such as:

- An increase in 1% of diamond's carat leads, on average, to an increase of 1.88% in the price;
- After carat, the most influential predictor for a diamond price is its clarity, with the best-quality clarity diamond being evaluated approximately twice as much as the worst-quality clarity diamond, keeping all the other factors fixed;
- The best-quality color diamond cost, on average, $(e^{0.5} - 1)100\% \sim 66\%$ more than the worst-quality color diamond, keeping all the other factors fixed;
- The best-quality cut diamond cost, on average, $(e^{0.15} - 1)100\% \sim 17\%$ more than the worst-quality cut diamond, keeping all the other factors fixed.