1 Pre-analysis

As a first step to this analysis, we identify possible features that may change the cost of a tube assembly quoted by a supplier. We also ask questions about the dataset where the answers to these questions will help us to choose the right machine learning algorithms to test. Since we are predicting the cost value, a continuous variable, we will use a regression algorithm family.

1.1 Questions

To archieve our goal of predicting the cost with a good accuracy, we need to answer the following questions.

1. What features are useful to determine the cost and what features to exclude from the analysis?
2. Is there a unique mathematical model describing the cost in function of the quantity for each supplier?
3. If there is a mathematical model, is it a linear or non-linear model?
4. Are there more than one model to estimate the cost where each model are independent of the others?

2 Possible Dependent Features

In this section, we will answer the question: *What features are used to determine the cost and what features to exclude from the analysis?*

2.1 Tube Physical Properties

As a supplier, we have to think on which tube features the cost will be based. We know that a tube assembly is made with one or more components. Some numerical tube properties may be helpful to check.

- The weight of the tube
- The quantity to purchase
- The volume (dependent of the diameter, the wall thickness and the length of the tube)
- The number of bends used with the bend radius. Logically, it is more difficult to bend a tube than to keep it linear, so it should be more expansive.
- The number of components to assemble a tube. Assembling many components need welds and connectors which should be more expansive.
- The material used to make the tube. Some type of material can be much expensive than others.
- The type of component used to assemble the tube. Some types of component may be more complicated to build by their shape.

2.2 Supplier Features

- The date when the supplier has quoted the price which is certainly less 20 years ago than today when not adjusted.
- The suppliers may use different mathematical models to quote their price. Some may set cheaper costs, some may set expensive costs.

3 Preparing & Cleaning the Dataset

In this section, we will explain why we chose to keep and exclude features and how we will clean the dataset.

From the dataset, we note that there are a total of 2048 components. These components are spread amoung the `comp_[type].csv` files uniquely. This means that we can create a single table `Component` by merging those files together. To avoid to many columns, we will remove some features that we do not want in this analysis.

The training and test sets are merged together where we set the cost to 0 for the test set.

The file `bill_of_materials.csv` gives us the list of components with their respective quantity used to assemble a tube. Thus, to calculate the total weight for each tube, we use the formula

$$W_T = \sum_{i=0}^{n} W_i * Q_i$$

where W_T is the total weight of the tube T, $W = (W_1, \ldots, W_n)$ is the vector of component weights, $Q = (Q_1, \ldots, Q_n)$ the vector of component quantities and $n \leq 8$ the number of possible components used to assemble a tube T.

Let the total volume estimation of a tube assembly be denoted by V_T. The volume is function of the length, the wall thickness and the diameter of the tube and its formula is

$$V_T = \pi L t(d - t)$$

, where t is the wall thickness, d the outside diameter and L the developed length of the tube.

Every ID used (e.g. tube_assembly_id, material, supplier, etc.) as a string in CSV files are converted to a positive integer without the leading zeros. The quote date is converted to a positive integer in the format `YYYYMMDD`.

To test and find data efficiently, we create a database `Caterpillar` with tables, views and indexes. The script to query this database is given by the file `DatabaseManipulation.R`. The script to insert in batch the data from the CSV files to the database is given by the file `DatabaseInsertions.R`.

We include librairies for graphs and tables to display in this document. We also connect to the Caterpillar database for the next queries to execute.

We prepare the train and test data needed for the analysis.

```
##           used (Mb) gc trigger (Mb) max used (Mb)
## Ncells  514634 27.5     940480 50.3   668225 35.7
## Vcells 1665914 12.8    2647842 20.3  2573503 19.7
```

4 Cost Models

In this section, we will answer the question: *Is there a unique mathematical model describing the cost in function of the quantity for each supplier?* The first objective is to check the existence of a mathematical model representing the cost in function of the quantity. The second objective is to show if the model is applied by a unique supplier. The last objective is to show if each supplier has its own model. If the unicity does not hold, then we have to check if a model is applied by more than one supplier or if a supplier can apply more than one model depending of other features. We will then answer the question: *If a mathematical model exists, is it a linear or non-linear model?*

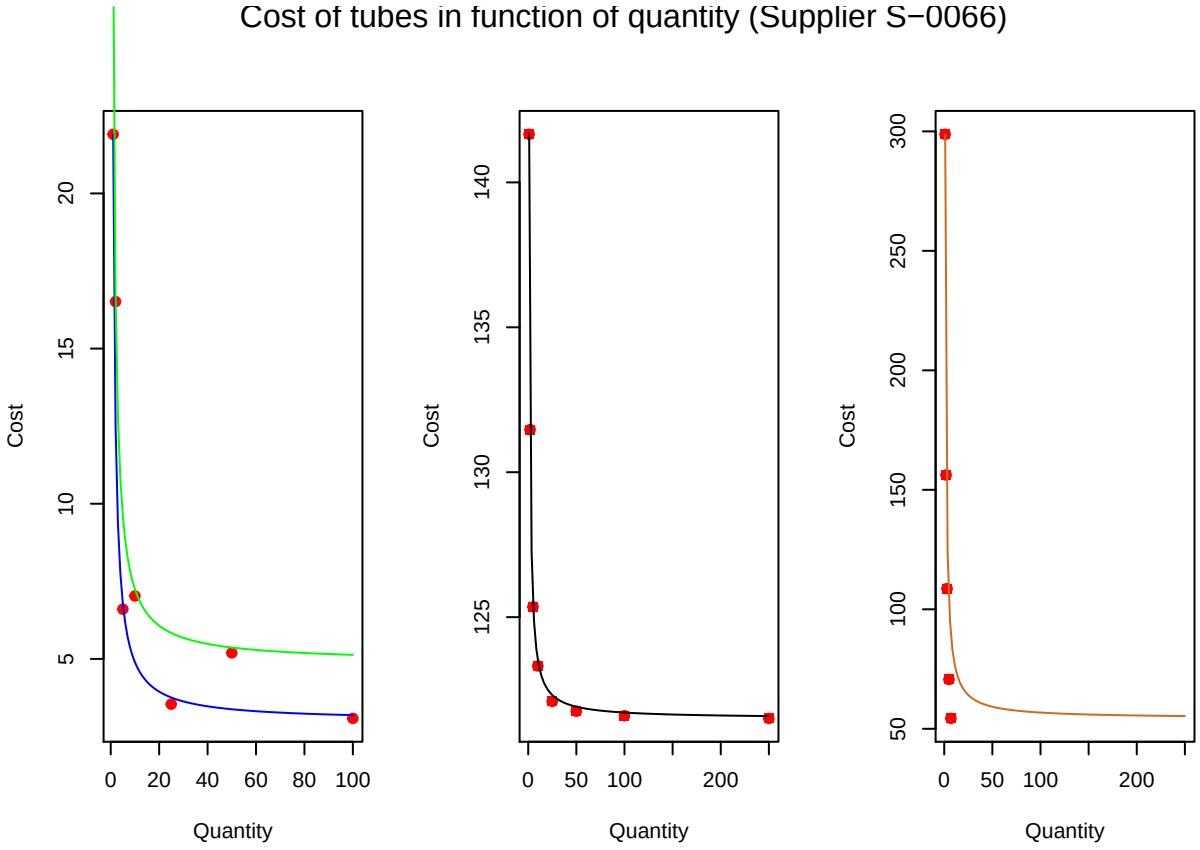
We denote $C_\beta(Q)$ our cost heuristic function of a tube assembly given by a supplier where β is our learning parameters and Q the vector of quantities.

4.1 Existence of a Mathematical Model

We start with few tube assemblies which are quoted by the supplier `S-0066`.

	fkTubeAssembly	supplierID	totalWeight	quantity	cost
9	2	66	0.02	1	21.91
10	2	66	0.02	2	12.34
11	2	66	0.02	5	6.60
12	2	66	0.02	10	4.69
13	2	66	0.02	25	3.54
14	2	66	0.02	50	3.22
15	2	66	0.02	100	3.08
16	2	66	0.02	250	3.00
33	5	66	0.21	1	28.37
34	5	66	0.21	2	16.51
35	5	66	0.21	5	9.40
36	5	66	0.21	10	7.03
37	5	66	0.21	25	5.60
38	5	66	0.21	50	5.19
39	5	66	0.21	100	5.01
40	5	66	0.21	250	4.90
17540	5000	66	0.29	1	141.66
17541	5000	66	0.29	2	131.47
17542	5000	66	0.29	5	125.35
17543	5000	66	0.29	10	123.32
17544	5000	66	0.29	25	122.09
17545	5000	66	0.29	50	121.75
17546	5000	66	0.29	100	121.60
17547	5000	66	0.29	250	121.51
51282	19365	66	5.04	1	298.78
51283	19365	66	5.04	2	156.20
51284	19365	66	5.04	3	108.67
51285	19365	66	5.04	5	70.64
51286	19365	66	5.04	7	54.35

Table 1: Tubes 2, 5, 5000 and 19365



From the graphs, we see that the curves estimating the red points are clearly hyperbolas of equation

$$C_T(Q) = \frac{\beta_0 - \beta_1}{Q} + \beta_1$$

where $Q \geq 1$ is the quantity for a tube assembly ID T, β_1 is the cost at the last level of purchase based on quantity and supplier (most of the time $Q = 250$), and β_0 is the cost at the first level of purchase based on quantity and supplier (most of the time $Q = 1$). This equation indicates that if Caterpillar buy more tubes of the same ID, cheaper will be the cost per tube. This proves the existence of a mathematical model representing the cost in function of the quantity.

If we take a look at the right most graph, we see that our curve doesn't seem to fit the points. However, the maximum quantity is 7 (not 250) for this tube which make the model less accurate assuming the same model is used. This assumption makes sense since

$$\lim_{Q \to \infty} C_T(Q) = \beta_1$$

which means that we need to find the right β_1 to match with any quantity. We also have to find the cost of one tube which is β_0.

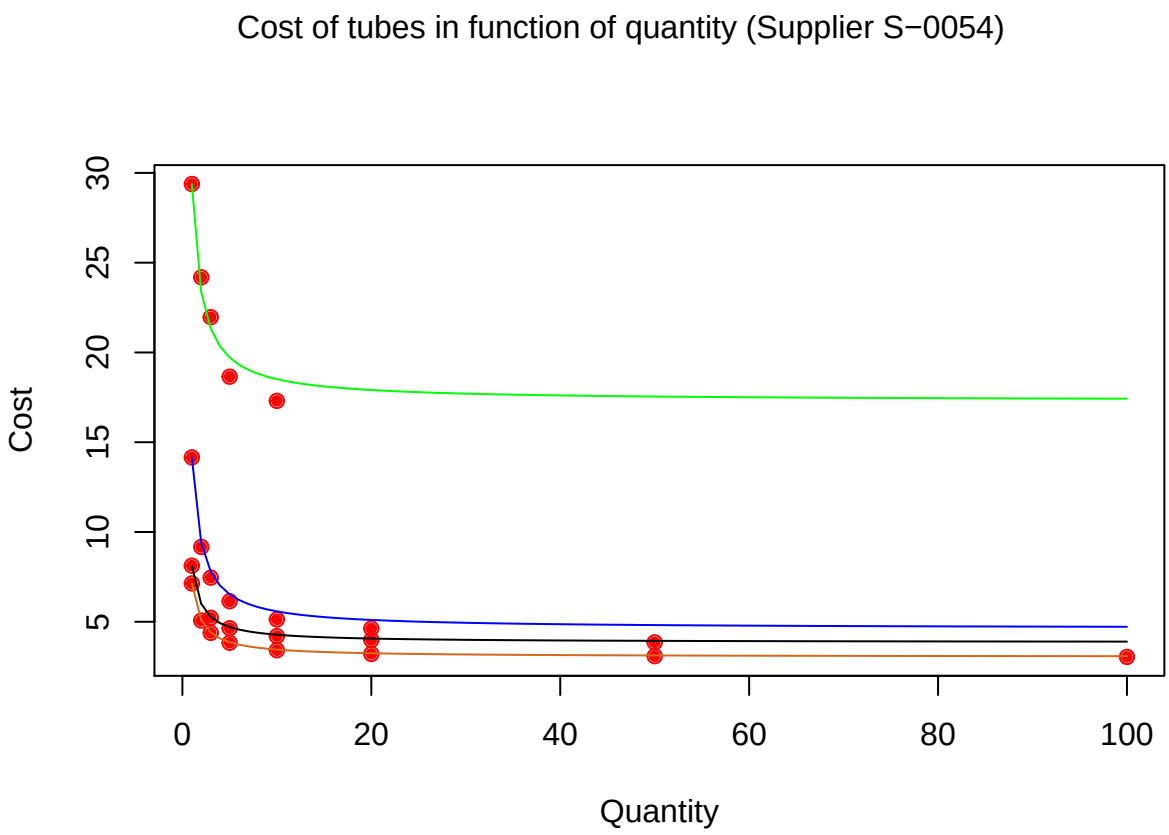
For example, if we take the tube `TA-19365`, we have $C_T(1) = \beta_0 = 298.7820145446$. We know that $C_T(2) = \frac{\beta_0+\beta_1}{2} = 156.1959237271 \Leftrightarrow \beta_1 = 13.60983291$. Therefore, the model for the tube `TA-19365` is $C_T(Q) = \frac{285.172181635}{Q} + 13.60983291$. With $Q = 7$, we obtain $C_T(7) = 54.348716001$ which has a square error of 0.000001422 from the original cost. With our estimated, i.e. $C_T(Q) = (244.434490855/Q) + 54.3475236892$, we have $C_T(7) = 89.266736668$ which has a square error of 1219.351435073. Thus, if Q is small (say $Q < 25$), the model may underfit.

4.2 Unicity of the Model per Supplier

We verify with few tube assemblies, which are quoted by the supplier `S-0054`, if the same model used for the supplier `S-0066` applies.

	fkTubeAssembly	supplierID	totalWeight	quantity	cost
627	130	54	0.04	1	14.16
628	130	54	0.04	2	9.17
629	130	54	0.04	3	7.46
630	130	54	0.04	5	6.15
631	130	54	0.04	10	5.13
632	130	54	0.04	20	4.63
1236	280	54	0.62	1	29.38
1237	280	54	0.62	2	24.18
1238	280	54	0.62	3	21.96
1239	280	54	0.62	5	18.65
1240	280	54	0.62	10	17.30
7074	1892	54	0.07	1	8.13
7075	1892	54	0.07	3	5.22
7076	1892	54	0.07	5	4.64
7077	1892	54	0.07	10	4.20
7078	1892	54	0.07	20	3.99
7079	1892	54	0.07	50	3.85
17588	5013	54	0.04	1	7.14
17589	5013	54	0.04	2	5.07
17590	5013	54	0.04	3	4.38
17591	5013	54	0.04	5	3.83
17592	5013	54	0.04	10	3.42
17593	5013	54	0.04	20	3.21
17594	5013	54	0.04	50	3.09
17595	5013	54	0.04	100	3.04

Table 2: Tubes 130, 280, 1892, 5013

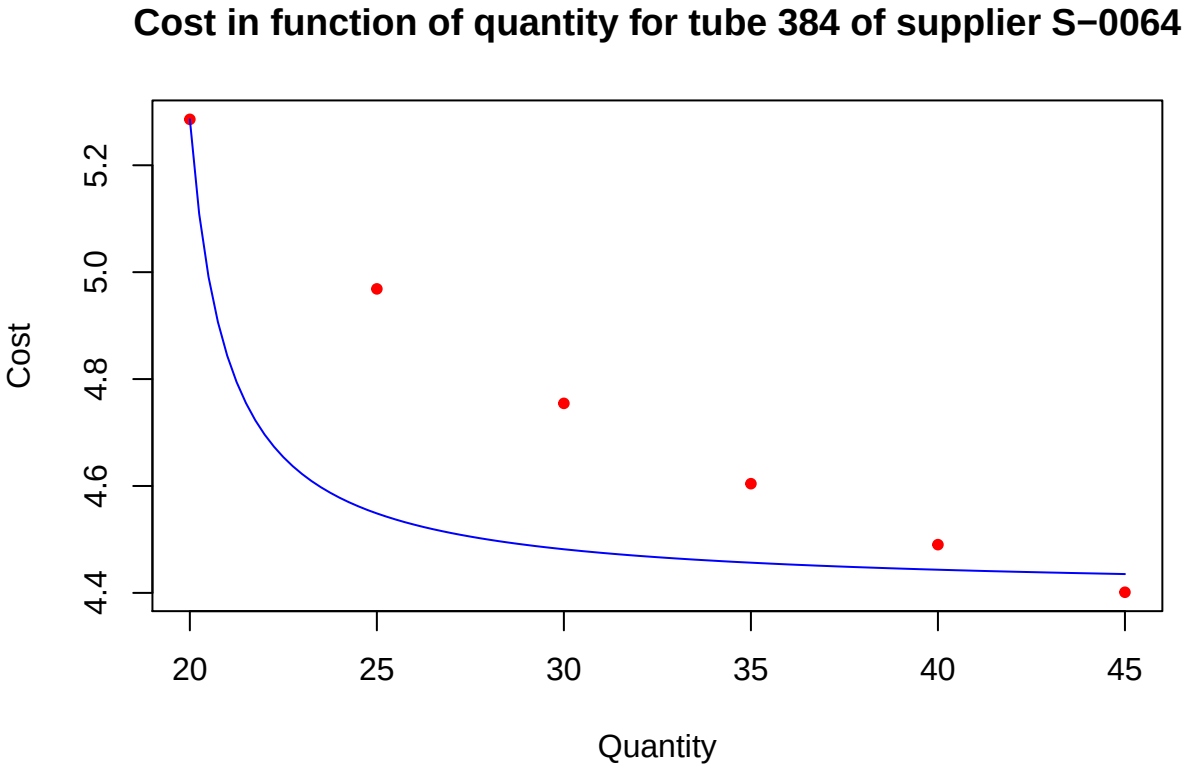


The model used by the supplier `S-0054` seems to be the same as the one used by the supplier `S-0066`, but if we look carefully at the curves, we see that greater is the quantity, more accurate is the estimate. This means that the model follows the same behaviour as the model used by the supplier `S-0066`.

This doesn't seem to be the case for the tube `TA-00384` from the supplier `S-0064`. This supplier provides 6 levels of purchase where the highest quantity is $Q = 45$. We use the same model as before but this time, the model doesn't fit the points.

	fkTubeAssembly	supplierID	totalWeight	quantity	cost
1590	384	64	0.38	20	5.29
1591	384	64	0.38	25	4.97
1592	384	64	0.38	30	4.75
1593	384	64	0.38	35	4.60
1594	384	64	0.38	40	4.49
1595	384	64	0.38	45	4.40

Table 3: TA-00384



Since the first quantity level is $Q_0 = 20$, we need to translate the model by subtracting x by $Q_0 - 1 = 19$. This gives the following model

$$C_T(Q) = \frac{\beta_0 - \beta_1}{Q - Q_0 - 1} + \beta_1$$

for all $Q \geq 1$. However, the model still underfits the data because the cost decreases much slower than the model used for our previous tests. Therefore, we can assume that a model with specific parameters are used to estimate the cost by one or many suppliers but not all. However, the general model is used by suppliers and may need to adjust the parameters to fit the data.

5 Decision Tree(s)

In this section, we will identify conditional paths which will tell us if decision trees will be useful or not. In the previous section, we have seen that the model can underfit if there are not enough quantity purchase levels and if the quantity is small. Otherwise, we can use the model to estimate the cost given a quantity and a tube assembly. Here are few points that identify some conditions.

- If there is only one quantity purchase level, we cannot estimate the cost. We need al least another feature on which the cost depends.
- If there are many quantity purchase levels but with $Q_0 > 1$, then we need to translate the model found at section 4.
- If the quantities are too low, then the model underfits. Thus, we need to add other cost-dependent features.
- Depending of the supplier, the model may be slightly different. Since we have 68 suppliers in the train and test sets, we can have at most 68 possible models.

Only with those conditions, we can build many decision or regression trees to help us estimating the cost.

6 Machine Learning Algorithms

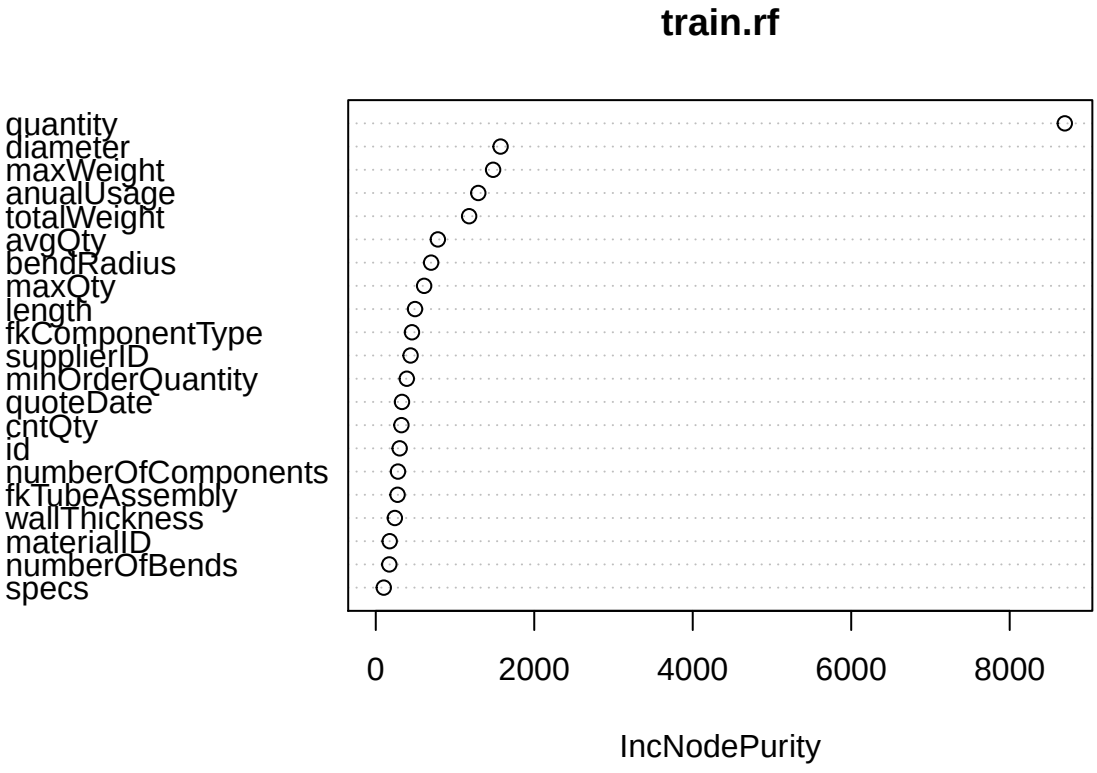
In this section, we present the machine learning algorithms we will try after the analysis given by the five previous sections. Per the section 5, we have seen that many decision trees can be built. Also, per the section 4, many models witch are simplifications of the general model can be used together to predict the cost. This leaves us two possible Ensemble algorithms: Random Forest or Boosting Trees for Regression (XGBoost).

6.1 Random Forest Algorithm

Suppose we want to create a tree for each supplier. This gives us a forest of 68 trees. Each of them may use or not the bracket pricing. The next level can be if they have been bended (at least one bend) or not. The next level can be the material used to make the tube where each of them has a certain probability of usage. We can go deeper, but this gives us a good example to show that a random forest algorithm is a good choice to predict the cost.

```
##
## Call:
##  randomForest(formula = log(cost + 1) ~ ., data = train, nTree = 20)
##                Type of random forest: regression
##                      Number of trees: 500
## No. of variables tried at each split: 7
##
##           Mean of squared residuals: 0.04257163
##                     % Var explained: 93.72

##    user  system elapsed
## 726.380   0.761 726.779
```



```
##                     IncNodePurity
## fkTubeAssembly          275.1177
## supplierID              439.4736
## quoteDate               330.9947
## anualUsage             1293.6987
## minOrderQuantity        391.5997
## quantity               8695.3275
## totalWeight            1178.2760
## maxWeight              1480.9618
## fkComponentType         456.8313
## numberOfComponents      280.5039
## diameter               1575.1161
## wallThickness           240.8338
## length                  495.4345
## bendRadius              698.6184
## materialID              175.9436
## specs                   101.1029
## numberOfBends           171.1472
## maxQty                  610.4802
## avgQty                  783.4290
## cntQty                  323.5661
## id                      301.7796
```

Using only all features of the train set to predict the cost with the random forest algorithm using regression gives a score of 0.412645. Here is a list of actions done to improve the prediction.

- Removing the day part of the quote date and keeping the date in the integer format `YYYYMM` improved the score with 82.4% of variances explained.
- Removing the bracket_pricing field improved the score with 84.56% of variances explained.
- Adding the total weight improved the score with 90.22% of variances explained.
- Adding the diameter of the tube improved the score with 91.96% of variances explained.
- Adding the wall thickness and length of the tube improved the score with 92.75% of variances explained.
- Adding the maximum and average quantity for each tube improved the score with 93.06% of variances explained.
- Adding the maximum weight of components for each tube improved the score with 93.17% of variances explained.
- Adding the component type for each tube improved the score with 93.31% of variances explained.
- Adding the typical bend radius for each tube improved the score with 93.40% of variances explained.
- Adding the number of quantities for each tube improved the score with 93.45% of variances explained.
- Adding the material ID for each tube improved the score with 93.59% of variances explained.
- Adding the number of specs for each tube improved the score with 93.61% of variances explained.
- Adding the number of components for each tube improved the score with 93.67% of variances explained.
- Adding the number of bends for each tube improved the score with 93.72% of variances explained.

Some features didn't improve the prediction.

- end_a_1x
- end_a_2x
- end_x_1x
- end_x_2x
- end_a
- end_x

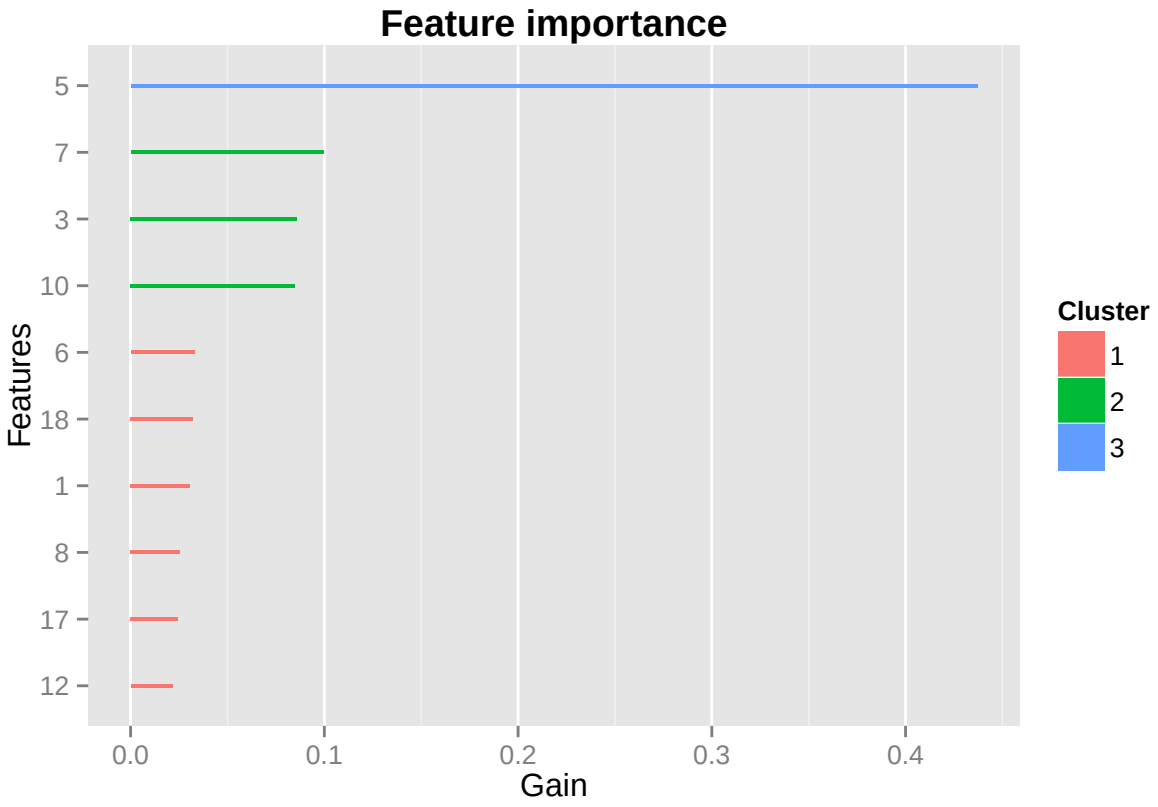
- num_bracket
- other
- num_boss
- number of quantity levels per tube
- bracket_pricing

6.2 Gradient Boosted Regression Trees

Before the learning we will use the cross validation to evaluate our error rate and to find the minimum number of trees needed to get a better prediction.

We can see from the graph that the number of trees almost stop varying from 1569 trees.

```
##  [1] "booster[0]"
##  [2] "0:[f5<9.5] yes=1,no=2,missing=1,gain=5779.41,cover=24032"
##  [3] "1:[f7<0.349] yes=3,no=4,missing=3,gain=1086.79,cover=11822"
##  [4] "3:[f3<74.5] yes=7,no=8,missing=7,gain=804.758,cover=11024"
##  [5] "7:[f5<4.5] yes=15,no=16,missing=15,gain=798.995,cover=9537"
##  [6] "15:[f18<1.16665] yes=31,no=32,missing=31,gain=269.124,cover=6432"
##  [7] "31:[f10<36.51] yes=63,no=64,missing=63,gain=314.939,cover=1360"
##  [8] "63:[f3<1.5] yes=107,no=108,missing=107,gain=50.8381,cover=1094"
##  [9] "107:leaf=0.0670047,cover=223"
## [10] "108:leaf=0.0945279,cover=871"
```



7 Results and Visualization

8 Conclusion