Machine Learning: Intermediate Report

Erwan Guyomarch and Thibault Pierotti (TPEG EKIP)

21 mai, 2019

Prof. Sebastian Engelke
Dr. Marta Pittavino
Sami Ghadfi

Contents

1 Executive Summary

This report is based on a Kaggle competition. We have made a first approach around the database by exploring it. Then, we applied several models to our data set to see which one allows us to have the best possible prediction (minimize Root Mean Squared Error). After that we went further and in addition to the previous experiments we have used cross-validation throughout the report, while implementing other models and methods.

Before combining models, our best results on the test set with simple models were *Random Forest*, *Elastic-Net* and *Monotone Multi-Layer Perceptron Neural Network*. The combinations of different models allowed us to achieve our best results in this competition.

2 Introduction

As part of the Machine Learning class, we participate in a data competition on the Kaggle platform. You will find our progress in this competition through this report, where we have put our efforts into an exploratory analysis. As this report progresses, we will discuss the selection of variables according to different models, and we will take the time to try to decipher what some of these variables can mean in real life in terms of quantity. The results of this report were computed with *R*.

3 Framework of the competiton

The purpose of this competition is to make the best possible prediction about the sale of sandwiches from the "*Sandwicheria*" restaurant over a 350-day period. To do this, this restaurant based in Geneva for 2 years provides us with a database covering the first 350 days (training data) of sandwich sales containing just over a hundred predictors as well as the number of sandwiches sold each day, and a second database containing the predictors of the last 350 days (test data) but not the sandwich sales during this period. The accuracy of the results will be measured using the RMSE (Root mean squared error, (1)), the aim being of course to have the lowest possible RMSE.

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{j=1}^{n}(y_j - \hat{y_j})^2}, \tag{1}$$

where n is the number of observations, $\hat{y}_j$ the prediction and y_j the actual observation. As the competition progresses, we will use different models in order to always produce better predictions. On the Kaggle platform we can visualize the public leaderboard which are based on 30% of the test data, the final results will be based on the remaining 70%, so it will be wise not to overfit the 30% of the test data of the public leaderboard.

4 Data used for this project

The data at our disposal is divided into training data and test data, concerning the train data it contains 350 rows corresponding to the first 350 days and 112 columns corresponding to the variable outcome y (the number of sandwiches sold) and 111 different predictors (e.g. weather forecast, day of the week etc. . .). The test data is built with the same structure for the next 350 days but we do not have the outcome variable in our possession. All the results and graphical representations of this report can be found in the Appendix section.

5 Exploring, cleaning and preprocessing

5.1 Missing values

Neither the training data nor the test data contain missing values.

5.2 Outcome variable

Based on the first 350 days of sandwich sales we have seen some details about the variable outcome. The average sales during this period was 447.6 sandwiches (median of 456), with the worst day with 3 sandwiches sold and the best with 872 sales (Table 1 and Figure 2).

Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
3.0	306.2	456.0	447.6	590.0	872.0

Table 1: Summary of the output variable y

We can have a first idea about the dispersion of these sales over these 350 days through a boxplot, the dispersion seems quite equal around the average. We can also look at the histogram to get another idea (Figure 3).

5.3 Quantitative and Qualitative variables

In order to better understand our 111 predictors, we researched which ones were qualitative and which ones were quantitative. With the assumption that a qualitative prediction would have a maximum of 12 levels, we could see that our training set included 52 qualitative variables (Table 2 and 5) (including for example X2 (2 levels), X4 (2 levels), X5 (2 levels), X7 (12 levels) etc. . .) and therefore 59 quantitative variables. Qualitative predictors were subsequently coded as factors.

Variables considered as factors (52)
X1 X4 X5 X7 X11 X12 X13 X14 X15 X16 X17 X19 X23 X25 X26 X27 X32 X33 X34
X35 X37 X39 X40 X42 X50 X51 X52 X55 X57 X58 X63 X64 X65 X68 X69 X70 X71
X73 X74 X75 X80 X82 X83 X86 X91 X93 X95 X97 X102 X105 X108 X111

Table 2: List of the factor variables

5.4 Correlation issues

To investigate the possible correlations between the variables we first tried to make a heat map, even if it seems that on this heat map the correlations were quite low the high number of variables made the visualization complicated. So we just manually checked if there seemed to be a lot of correlation above 0.5. We noticed some high correlations, in particular the variable X60 and X62 were correlated at 0.987, X34 and X75 were correlated at 0.852, and finally X104 and X68 at 0.648 (Table 5). So We have decided not to remove anything for the moment, it would be dangerous at this stage to do without one variable rather than another without more information.

Variables	Correlation
X62 - X60, X34 - X75	<= 0.8
X62 - X60, X104 - X68, X34 - X75	<= 0.5

Table 3: High correlations

5.5 Outliers and Normality issues

We tried to see if our data set had many outliers, most of the variables did not have any (based on the *boxplot.stats* function) and some had a maximum of 4 (Figure 4). For normality after looking at several qq-plots of the numeric variables, it would not seem that all of the data are normally distributed. Ultimately it should be remembered that here we are doing a prediction contest and not an explanation contest (in which we would look for relationships between the different variables), so if some assumptions are not correct it is not so serious compared to a relationship analysis.

5.6 Partitioning

In the first part, the part with linear models we have partitioned in the following way. We wanted to be able to do the first tests without going directly through the test set and the Kaggle platform, so we decided to split our training set in two, 70% into a training set and 30% into a validation set. You will therefore find in the results the RMSEs on the training set, the validation set and the test set (Kaggle result). We also create scaled sets, even if it wasn't mandatory with all methods.

However, after having noticed the limitations of such a methodology to make predictions on a new database, we decided to implement the cross validation method on the second report. So we redesigned some linear models using cross validation, and we used cross validation again for the rest of the report.

6 Data mining tools used in this report

In this report we used two baseline methods (mean method and kNN), several linear methods (linear simple regression, stepwise method, ridge and lasso regression, elastic-net), tree-based methods (simple tree, bagging tree, random forest) and finally a neural network model and then overall methods of several models.

7 Cross-Validation

In this report we have decided to validate our models using the Cross-Validation K-folds method. We used a simple method with 10 folds, and we will provide you with different graphs throughout our analysis. In this report, the cross-validation selects the model with the lowest RMSE.

$$CV_{(K)} = \frac{1}{K} \sum_{k=1}^{K} \text{RMSE}_{(k)}, \tag{2}$$

where K is the number of folds and $\text{RMSE}_{(k)}$ the RMSE (See equation (1)) the the kth element.

8 Baseline models

We have chosen to start with two model baselines. The first one simply extracted the average value of the number of sandwiches produced per day during the first 350 days, and the second one using the kNN method.

8.1 Mean model

The average is about 447.6 sandwiches per day on the training set.

$$\text{mean}(y) = \frac{1}{m}\sum_{i=1}^{m} y_i, \tag{3}$$

where m is the total number of the observations and y is the observations vector. So we simply predicted this average value for the 350 days of the test set, as we thought the result is not exceptional (here it is not the goal, this model being a baseline), we have a RMSE on the test of 199.83.

8.2 kNN model

$$\hat{y_0} = \frac{1}{k}\sum_{i:x_i \in N_k(x_0)} y_i = \text{ave}\{y_i : x_i \in N_k(x_0)\}, \tag{4}$$

where $N_k(x_0)$ are the k closest points x_i to a new predictor: x_0. To measure closeness between quantitative predictors, a simple Euclidean distance is used, but when it comes to qualitative predictors, the distance generally used is in terms of magnitude, so the quantitative variables must be normalized for the magnitudes to be similar. For the kNN method, we used cross validation to find the optimal tuning parameters (here k, the number of neighbours) to minimize the RMSE. We have done our tests ranging from 1 to 20 neighbours, we can see on the cross validation graph (todo) that the solution with 33 neighbours seems to minimize the RMSE. However if we refer to the *one-standard-error* rule, we can see that 6 neighbours will be more adapted to avoid overfitting (See Figure 5). On the test this second baseline model allowed us to have an RMSE of 161.10.

9 Linear models

9.1 Simple Linear Regression

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + ... + \beta_p X_p + \epsilon, \tag{5}$$

where Y is quantitative response variable, x_i is a predictor and β_i its coefficient, for p predictors. We had already investigated the possible problems related to correlation, normality or outliers, we decided to check the linear regression assumptions. On the diagnostic plots the results seem acceptable (see comments on Figure 6). To begin with it was interesting to make a linear regression with all the variables. It was not surprising to note that many variables had p-values greater than 5%. Concerning the predictions we found that this model worked very well on the training set (RMSE: 30.17), however on the Cross-Validation (here there is no tuning parameters so not graphical representation is needed) and the test set the results were much less impressive (RMSE: 119.05 and 162.33) probably because of an overfitting of the training set.

9.2 Linear Regression - Stepwise selection

For the second step of our model research we focused on a linear regression but this time with a stepwise selection. We used the AIC (Akaike Information Criterion) to compare the models:

$$AIC = 2k - 2\ln(L), \tag{6}$$

where k is the number of parameters and L is the maximum likelihood of the model. We used as "both" direction, this method chooses the model that minimizes the AIC, the AIC penalizes the number of variables of each model according to a parsimony criterion while trying to maximize the likelihood of the model. We therefore find ourselves with 79 variables instead of the previous 111, and with an AIC of 2018.4. On training and validation set the results were close to the simple linear regression, on the other hand on the test set we could see a clear improvement in our predictions (from 162.33 previously to 92.18 on the test test).

Concerning the variable selection, we extracted the variables that had the most impact on our models. We already applied the stepwise regression so we are here only looking at the results in a variable selection perspective. After having applied the stepwise regression, we are able to select 79 variables out of 111 which represent approximately a 28% reduction. Unfortunately, this method does not allow us to have a clear idea of the relative importance of each variables.

9.3 Ridge Regression

$$\sum_{i=1}^{n}\left(y_i - \beta_0 - \sum_{j=1}^{p}\beta_j x_{ij}\right)^2 + \lambda\sum_{j=1}^{p}\beta_j^2 = RSS + \sum_{j=1}^{p}\beta_j^2, \tag{7}$$

where λ (≥ 0) is called the *tuning parameter*. The βs are the coefficients (defined in the linear regression (5)) and x_{ij} is the ith observation of the jth predictor. The penality term $\lambda\sum_{j=1}^{p}\beta_j^2$ is called a shrinkage penalty and is small when $(\beta_1, ..., \beta_p)$ are close to 0. The Residual Sum of Squares (RSS) is:

$$\sum_{j=1}^{J}\sum_{i:x_i\in R_j}(y_i - \hat{y}_{R_j})^2. \tag{8}$$

The first shrinkage method that we used is the ridge regression. To create the model, we used the glmnet function. The tuning parameter λ that we are using has been selected using the a 10-fold cross validation (with the function *cv.glmnet*) and is 57.22. It is the value that gives the minimum mean cross-validated error. The RMSE for the training set, the CV and the test set are 48.51, 121.27 and 95.43. This model is better than the stepwise regression for the validation set, but not for the test set. We will compare the results to the lasso regression.

9.4 Lasso Regression

$$\sum_{i=1}^{n}\left(y_i - \beta_0 - \sum_{j=1}^{p}\beta_j x_{ij}\right)^2 + \lambda\sum_{j=1}^{p}|\beta_j| = RSS + \sum_{j=1}^{p}|\beta_j|, \tag{9}$$

where $\lambda(\geq 0)$ is called the *tuning parameter*. The second shrinkage method that we use is the lasso regression. The model has been fitted on the training data using the glmnet function. The tuning parameter λ that we are using has been selected using the a 10-fold cross validation (with the function *cv.glmnet*) and is 9.58. The results are so far the best we can have using the selected models. The RMSE of the training set is 69.97, the CV is 90.92 and the test set has a 75.36 score. Since the computations can become complicated with more than 100 predictors for multiple variable selection functions (such as regsubset), we can also use the lasso to do variable selection. In our situation, we obtain for the $\lambda = 7.95$ a reduced model of 25 predictors.

10 Tree models

10.1 Single regression tree model

$$\hat{y}_{R_j} = \frac{1}{n_j} \sum_{i:x_i \in R_j} y_i, \tag{10}$$

where $\hat{y}_{R_j}$ is the mean of the response of all training observations in R_j, where n_j denotes the number of training observations in R_j. The goal is to find boxes $R_1, ..., R_j$ that minimize the Residual Sum of Squares (RSS) (Equation (8)).

We started this part by creating a first tree based on all variables with a very low complexity parameter (0.001), a minimum observation per bucket of 1 and a depth value equal to 30, the length of this tree is 41. We can see on this tree that the first variable that comes into play is X60, and then come X35 and X75, in a third time it will again be X35 and then X104 that will be used to predict the result. It is interesting to note that these are variables that have already been highlighted in our selection of variables previously.

We then decided to pruned our tree for the purpose of parsimony, we can see that the length has increased here to 6 and that from a variable point of view this pruned tree uses X60, then X35 and X75. We used the cross-validation to find an optimal cp (Figure 14). In terms of results, this tree gives us an RMSE of 74.79, on cross-validation 105.82 and finally 111.24 on the test set. These trees can therefore be easily used for understanding purposes, but for purely predictive purposes the results are not excellent.

10.2 Bagging tree

We were then able to improve our results using the bagging method, which allows us to create averages of different trees, so we want to reduce the variance of our models in exchange for slightly increasing the bias. The use of bootstrap aggregating gives us an RMSE of 34.54 on training, 87.93 with cross-validation (here no tuning parameter), with 25 bootstrap replications and finally 85.49 on the test set. There is therefore a real improvement compared to simple trees.

$$\hat{f}_{bag}(x) = \frac{1}{B} \sum_{b=1}^{B} \hat{f}^{*b}(x), \tag{11}$$

where $\hat{f}^{*b}(x)$ is bth bootstrat training estimator and B the different bootstrapped sets created.

10.3 Random forest

Finally we implemented the random forest method. Random forest is an optimization of Bagging in his way to forcing each split of the data set to consider only a subset of the predictors. Here we will look at the results of this method, the results are close to the bagging method, 36.74 of RMSE on training, 88.36 with cross-validation and finally 86.16 on the test. Unfortunately, these good results in terms of predictions are at the expense of an intuitive understanding in terms of interpretability.

Nevertheless we can have some more information using cross validation, the cross validation method offers us models with a different number of predictors, we can see on the cross-validation figure that in this case the RMSE seems the lowest with about 56 predictors.

Concerning the variable selection, we built a random forest using all the variables. We can see on the variable importance (Figure 9). We selected the 8 most impactful variables: $X35, X62, X60, X75, X34, X104, X68$ and $X59$ (we will see that some of these variables will also be important in other models like the Pruned Tree).

11 Analysis of the important variables

We will now take a deeper look at 8 important variables (these variables seemed important in different models as we have already mentionned) (Figure 10 and 11). The categorical variables are represented in Barplots, and the numerical ones on Scatterplots. We will now try to find the real-world signification in the framework of the "*Sandwicheria*". We can see that the variable X34 has 12 categories whose values are between 15 and 25. It could represent months. The variable X75 is divided in 4 buckets almost equivalent (in volume). It could represent quarters of years. These deductions ca be supported if we remember that these 2 variables are strongly correlated (0.852). The variable X68 has 3 buckets. It could represent the status of the day. It can be a holiday or a weekend day (first category), a regular (working) day (second category, biggest bucket) or a special day such as a feast day like Halloween (third categorie, rare). The variable X35 has only two values and is one of the most important variables. It could represent a separation between a good weather and a bad one for a specific day we can see that for one of the two different categories, there is more sandwiches sold than the other category (Figure 12). Finally, the variable X60 and X62 are strongly correlated (0.9) so they should represent the same quantities, in this case maybe the number of clients. The variables X59 and X104 are two variables numeric. If we take a closer look to X104, we can distinguish cycles.

12 Other models

In this part, we deployed two techniques: *Elastic-net*, *Random Forest* and *Monotone Multi-Layer Perceptron Neural Network*. With only a few selected variables and interactions. For the interactions, the categorical variables were coded as numerical ones. For the variables and interactions selection we started from the last results (variable selection part) and throught multiple trials using CV, we finally found the following models:

12.1 Elastic-net with few variables and interaction

Here we used the Elastic-net model which is a combination between the Lasso and the Ridge methods.

$$\lambda \sum_{i=1}^{p} \Big((1-\alpha)\beta_j^2 + \alpha|\beta_j| \Big) \tag{12}$$

where $\alpha \in [0,1]$ (note: $\alpha = 0$ results in ridge regression and $\alpha = 1$ in lasso).

For this method, we used model described by the equation (13).

$$\begin{aligned} \hat{y} = X35 + X35^2 + X35^3 + X35 \times X60 + (X35 \times X60)^2 + \\ X60 + X60^2 + X60^3 + X75 + X75^2 + X34 + X34^2 + X68 + X68^2 + \\ X104 + X104^2 + X59 + X11 + X11^2 \end{aligned} \tag{13}$$

When applying this model on the CV, the hyperparameters show us that the best model was the result of a $\lambda = 0$ resulting in a Ridge situation. On the CV we have a RMSE of 66.09 while having 63.44 on the training and an excellent score of 65.34 for the test set on Kaggle.

12.2 Random forest with fewer variables and interactions

For this Random Forest, we used the model described by the equation (14)

$$\hat{y} = X35 + X62 + X60 + X75 + X34 + X68 + X104 + X59 + X7 + X11 \quad (14)$$

With this method we had a RMSE of 32.27 on the training set, 71.12 for the CV and 69.65 for the test set on Kaggle.

12.3 Monotone Multi-Layer Perceptron Neural Network

We tried a neural network model ("Monotone Multi-Layer Perceptron Neural Network" from the *caret* package) with which we had good results, 60.14 of RMSE on training, 63.76 on cross-validation and finally 65.32 on the test set. From a technical point of view the cross-validation allowed us to choose only 1 hidden unit for our model.

For this method, we used model described by the equation (15)

$$\begin{aligned}\hat{y} = X35 + X35^2 + X35 \times X60 + (X35 \times X60)^2 + \\ X60 + X60^2 + X60^3 + X75 + X75^2 + X34 + X34^2 + X68 + X68^2 + \\ X104 + X104^2 + X59 + X11 + X11^2 \quad (15)\end{aligned}$$

13 Ensembles

In order to have more robust results we have created different ensemble models. So we took the predictions from different models, and averaged them together. We only used these results on the kaggle set test.

13.1 Ensemble 1

Our ensemble 1 is the average of our predictions from the Elastic-Net model and those from the Neural Network model. This technique was satisfactory by lowering the RMSE on the test set to 64.52.

13.2 Ensemble 2

The second ensemble was this time the average between the predictions of 3 models, the Elastic-Net model, the Neural Network model and the Random Forest model. This set gave us an even better result with a score of 62.68 on the test set.

14 Results and discussion

The table below (Table 4) shows the different RMSEs obtained by each model, respectively on the training set, the validation set and the test set. As for the RMSEs on the test set, they correspond to the results after submission on kaggle. Since the file "SampleSubmission.csv" obtained a score of 199.83, we succeeded in our first mission by beating this score with each of our linear models. In the second report we had to beat Professor Engelke's score of 81.25, which several models did (Lasso, Elastic-net, Random forest), so the mission is fulfilled. The last column is based on the Kaggle competition Final set (*Private*, the remaining 70% of the test set).

	Training set	Cross-Validation	Public test set	Private test set
Mean model	197.36	197.35	199.83	186.29
kNN	156.61	162.33	161.10	151.57
Simple linear regression	**30.17**	119.05	162.33	180.26
Stepwise linear regression	31.42	112.7	92.18	107.07
Ridge regression	48.51	121.27	95.43	104.66
Lasso regression	69.97	90.92	75.36	81.01
Simple pruned tree	74.79	105.82	111.24	104.01
Bagging tree	36.74	88.36	85.49	78.11
Random Forest	34.54	87.93	86.16	85.25
Elastic net*	63.44	66.06	65.34	64.63
Random Forest*	31.54	71.12	69.65	62.95
MMLP Neural Network*	60.14	**63.76**	65.32	63.36
Ensemble 1	-	-	64.52	63.27
Ensemble 2	-	-	**62.68**	**61.74**

Table 4: Result table to compare the different models. The * (star sign) means "With few variables"

We can therefore see that on the training set it is always the simple linear model that allows the most reliable forecasts, but the simple linear model uses all the variables that must create a lot of noise on the new data in addition to a probable overfitting of our training set. It is therefore interesting to note that when we want to make predictions on new data, it was the lasso model that allows us to have the best results (validation set and test set) during the first report. With the addition of interaction and a selection of variables, it is the Elastic-net, the Random forest and the Neural Network models that have allowed us to have the most accurate predictions. We have combined some of these models to further improve the predictions and reach our best result with only one model, after that, the creation of the ensemble allowed us to improve our scores.

It is also interesting to note that overall the Cross-validation scores were quite close to those of the test set for several models, which confirms the importance of this tool during such competitions.

Not surprisingly, the ensemble methods allow us to improve our results, which is intuitively very logical since one model can perform well in one place and another model will perform better in another, so the combination of these models with each other reduces the average overall error of our forecasts. With the addition of the private test set we can get an idea of the robustness of our different models. We can see on the first simple models without variable selection that there are variations between the private leaderboard and the public. On the other hand, our latest models with variable selection, interaction and power addition, seem to be quite robust and representative of reality since the RMSEs are very close on the 2 different test sets.

To conclude we can look at a last graph (Figure 1), it represents the predictions for 100% of the test set, which we made using our second set built from an Elastic-Net model, a Random Forest model and a Neural Network model.

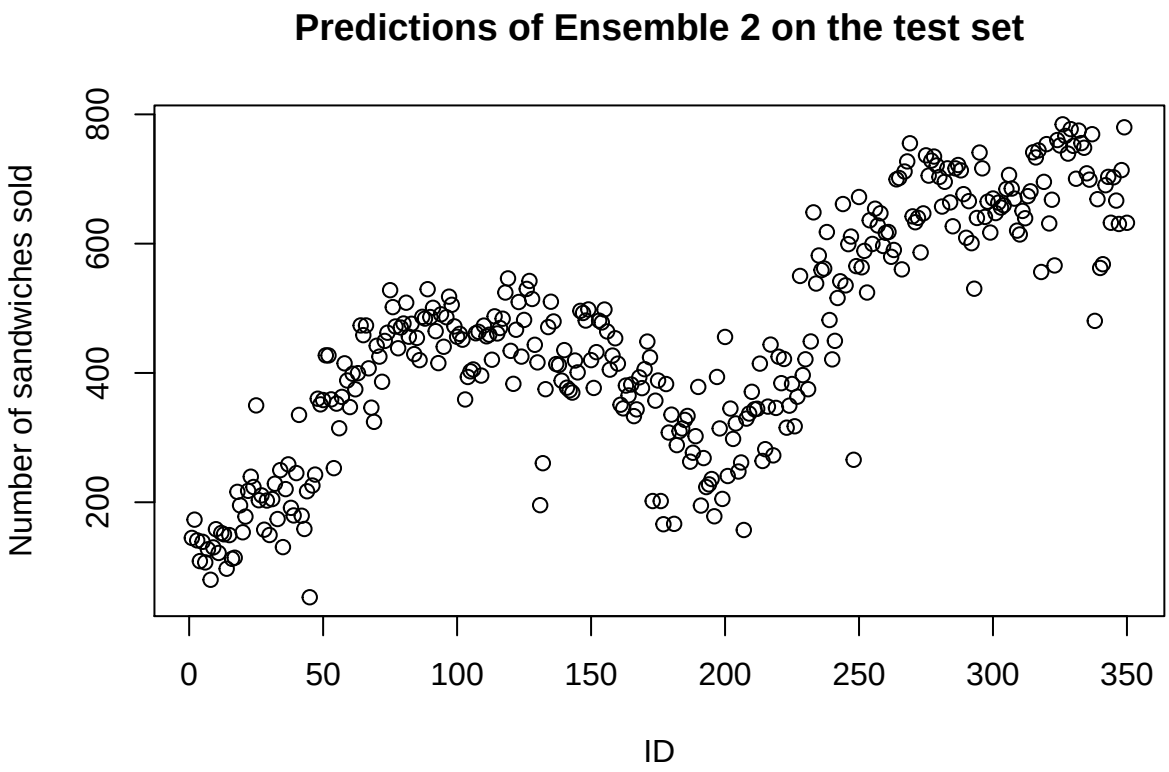

Figure 1: Predictions of our best ensemble model (Ensemble 2) on the test set

It should be noted that each of the 3 models taken individually gave us predictions with a similar pattern, however the predictions were a little higher or lower depending on the model, averaging these 3 models therefore allowed us to have predictions closer to reality by exploiting some strengths of each model.

15 Appendix

In this section you will find various tables and graphs that we have referred to throughout this report.

cp	RMSE	Rsquared	MAE
0.001	108.46	0.7097	84.37
0.01	105.82	0.7117	81.65
0.1	120.20	0.6287	96.39
0.2	139.66	0.4923	114.29
0.5	194.71	NaN	160.72

Table 5: Regression tree - Complexity parameters table

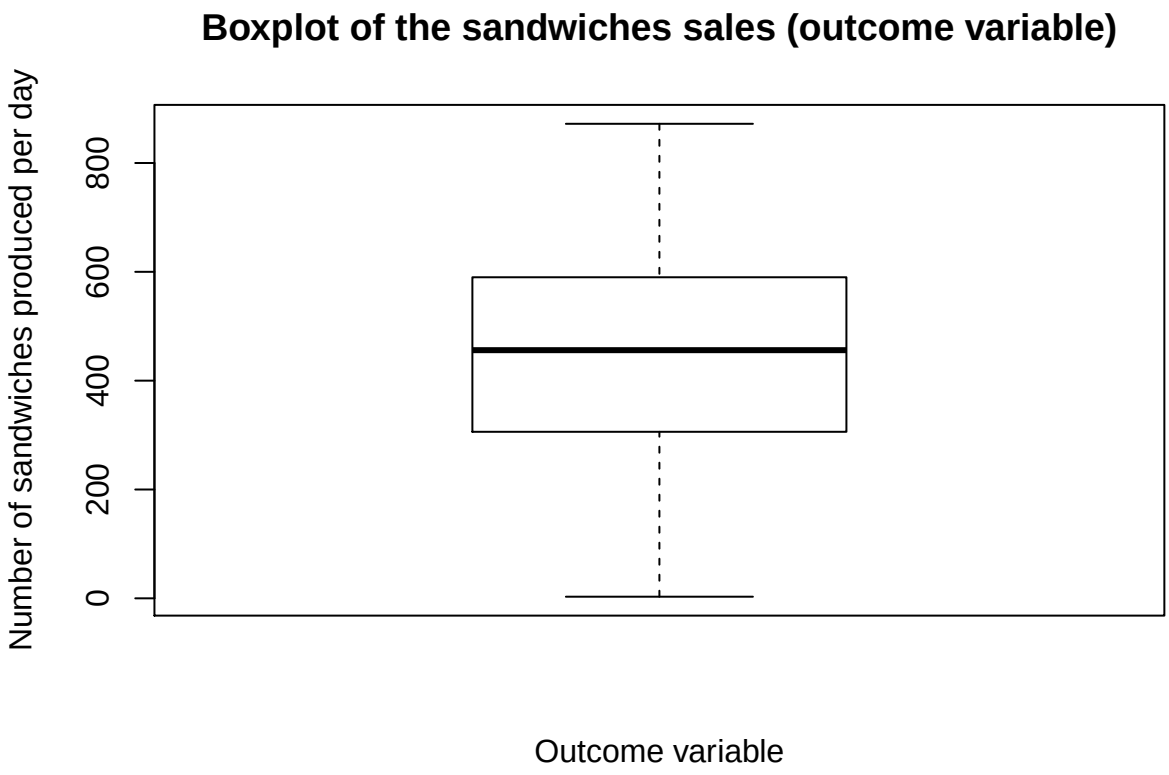

Figure 2: Boxplot of the sandwiches sales (outcome variable)

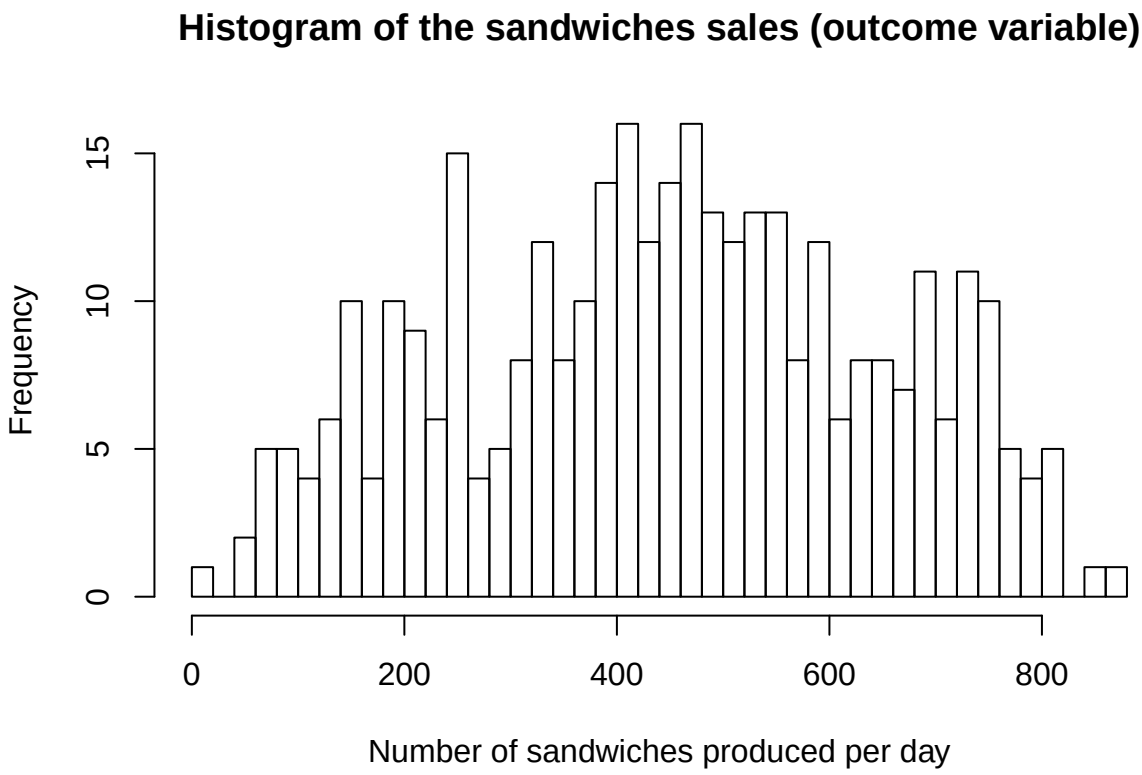

Figure 3: Histogram of the sandwiches sales (outcome variable)

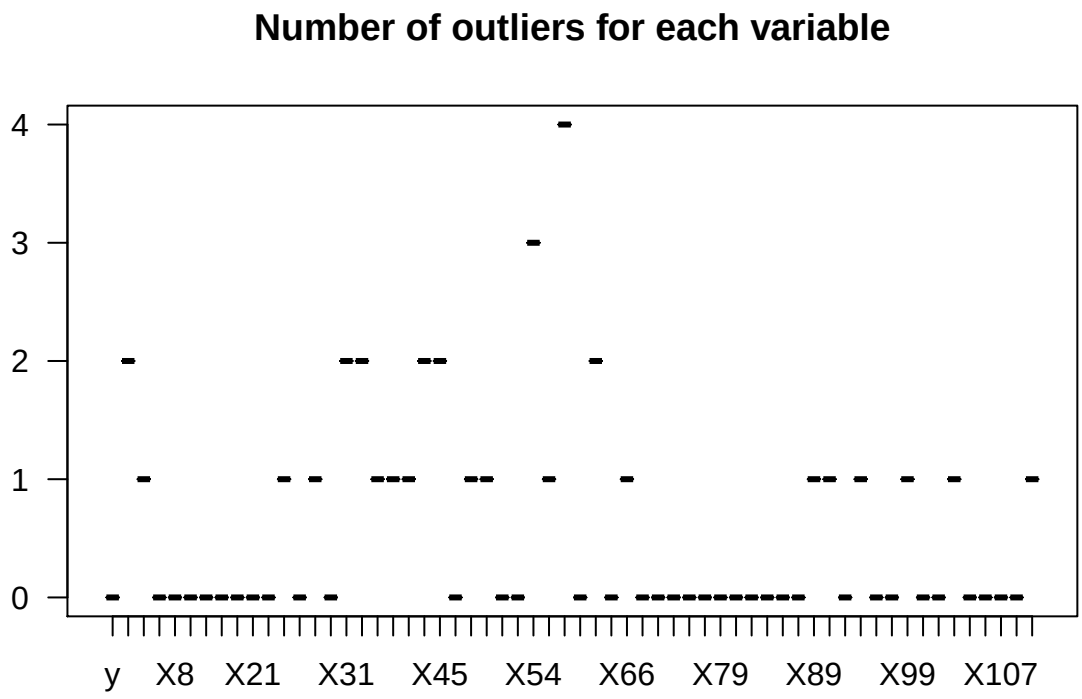

Figure 4: Number of outliers for each variable

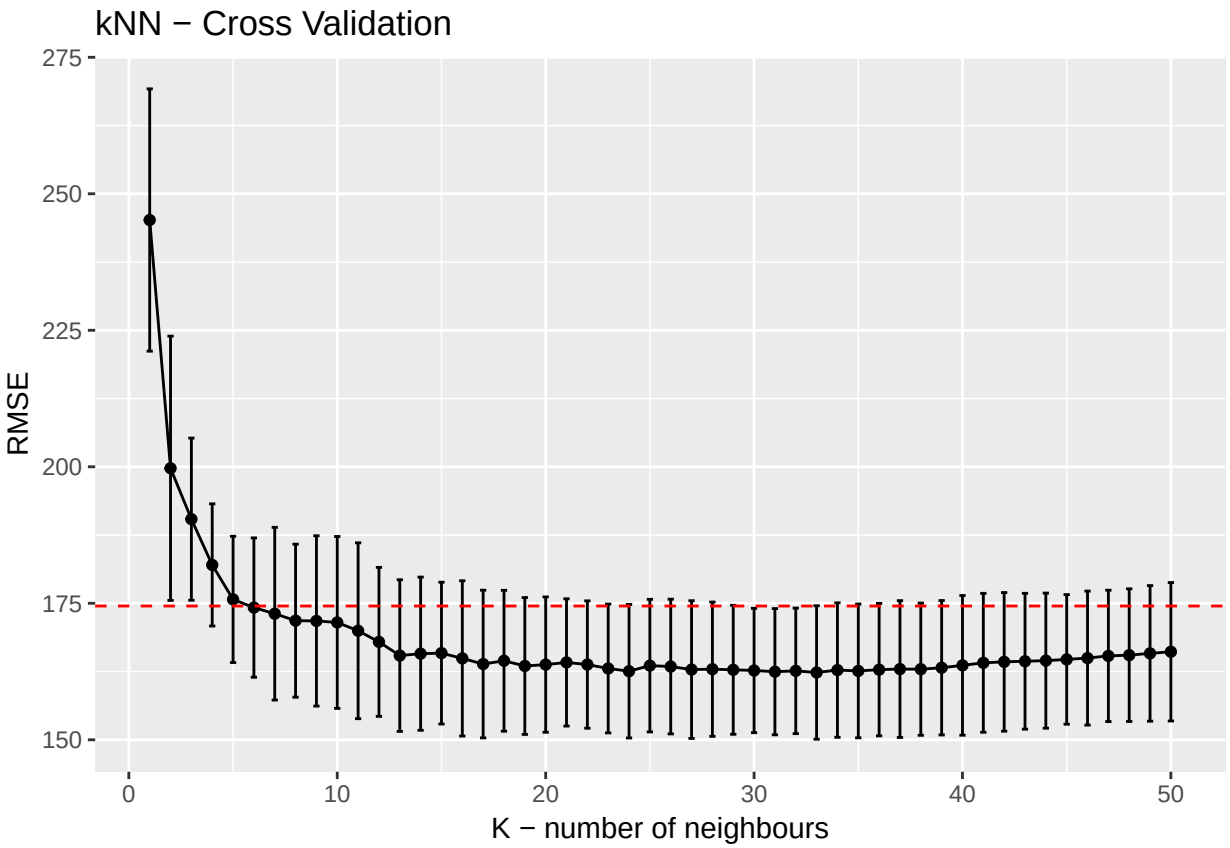

Figure 5: Neighbor selection for the kNN method

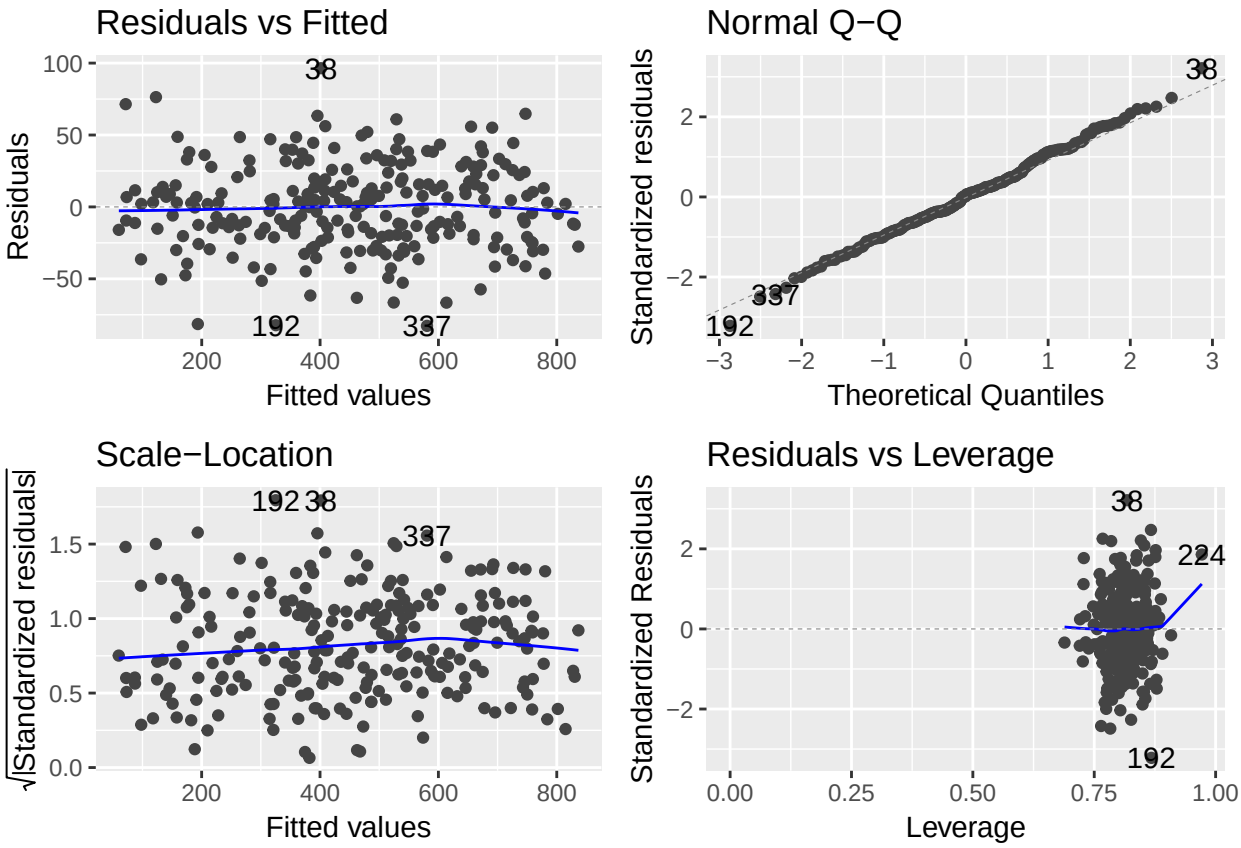

Figure 6: Residuals diagnostic plots. Residuals vs Fitted (top-left). We see that the blue line is horizontal, the assumption of linearity seems to be respected here. Normal Q-Q (top-right). The residuals points follow the line very well, the assumption of normalcy of the residues is quite acceptable here (and in our case here of prediction, a small violation would not have been very serious). Scale-Location (or Spread-Location) (bottom-left). The line here is also overall horizontal, which validates the assumption of homogeneity of the variance of the residues. Residuals vs Leverage (bottom-right). Here it would seem that there may be some influences (point outside Cook's distance), as mentioned above we do not have enough knowledge about the database to make a decision about these influencers.

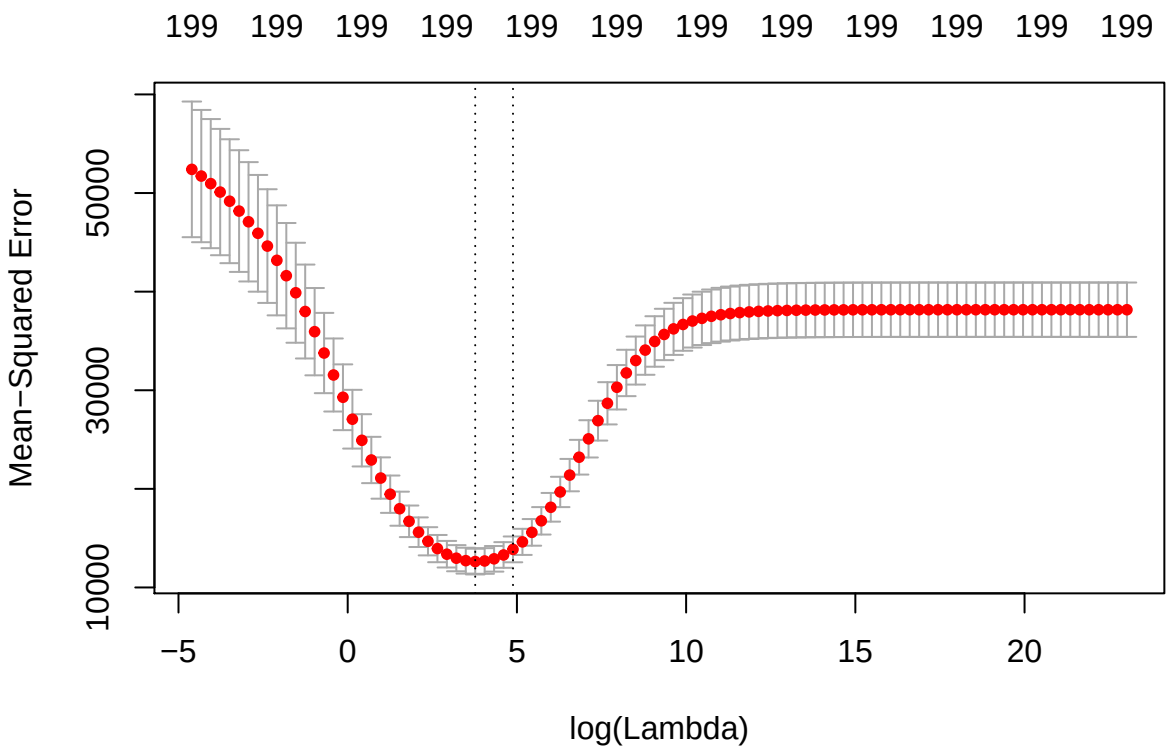

Figure 7: Ridge CV: RMSE for each lambda

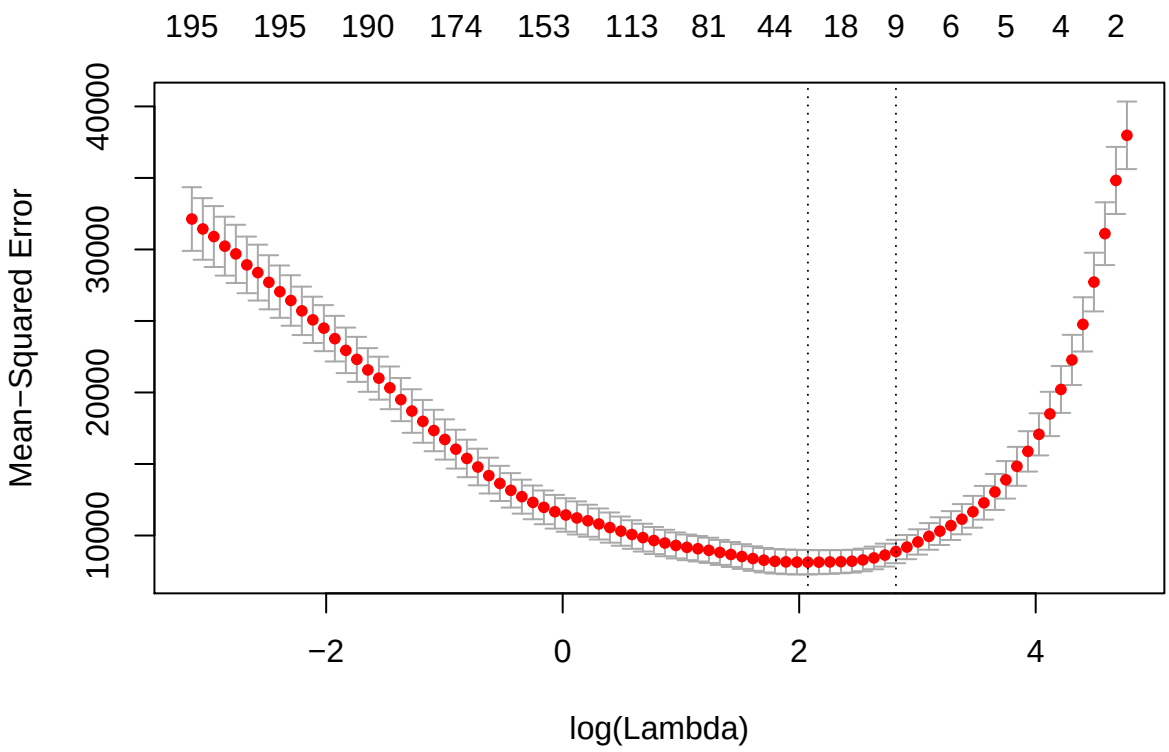

Figure 8: Lasso CV: RMSE for each lambda

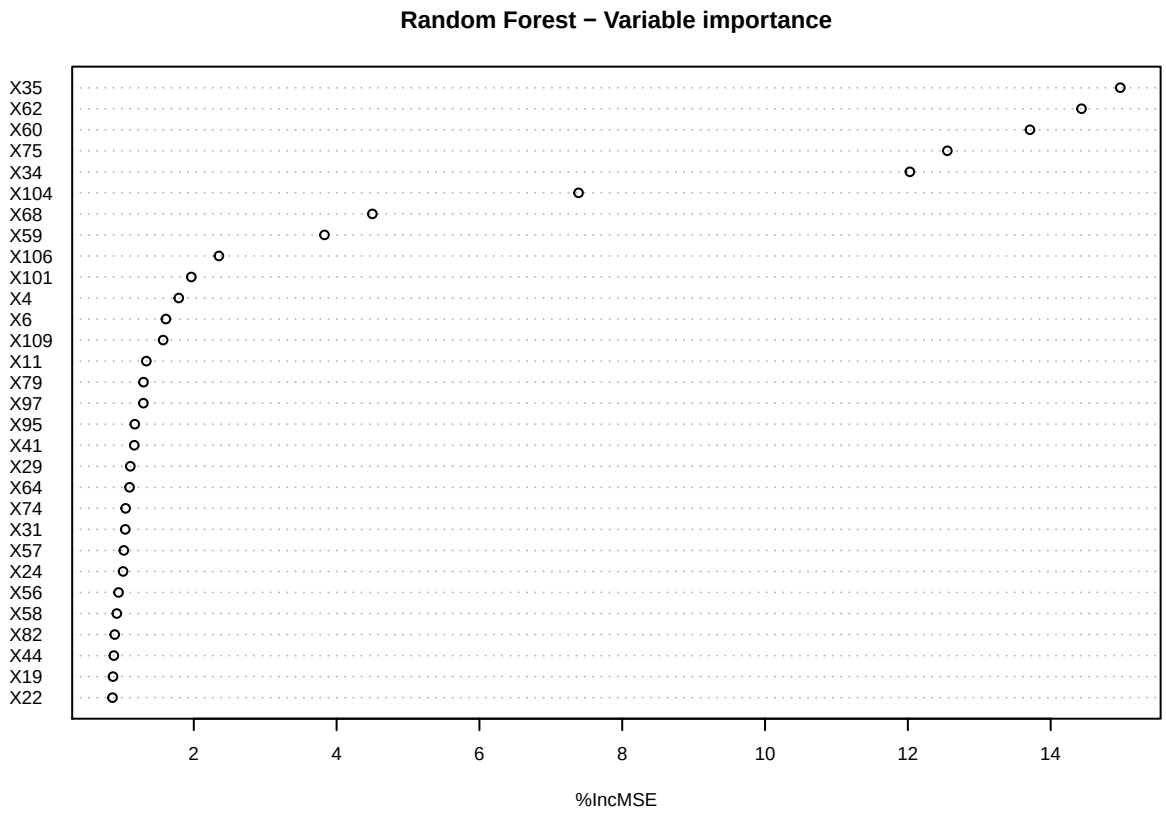

Figure 9: List of the variables and the representation of their importance

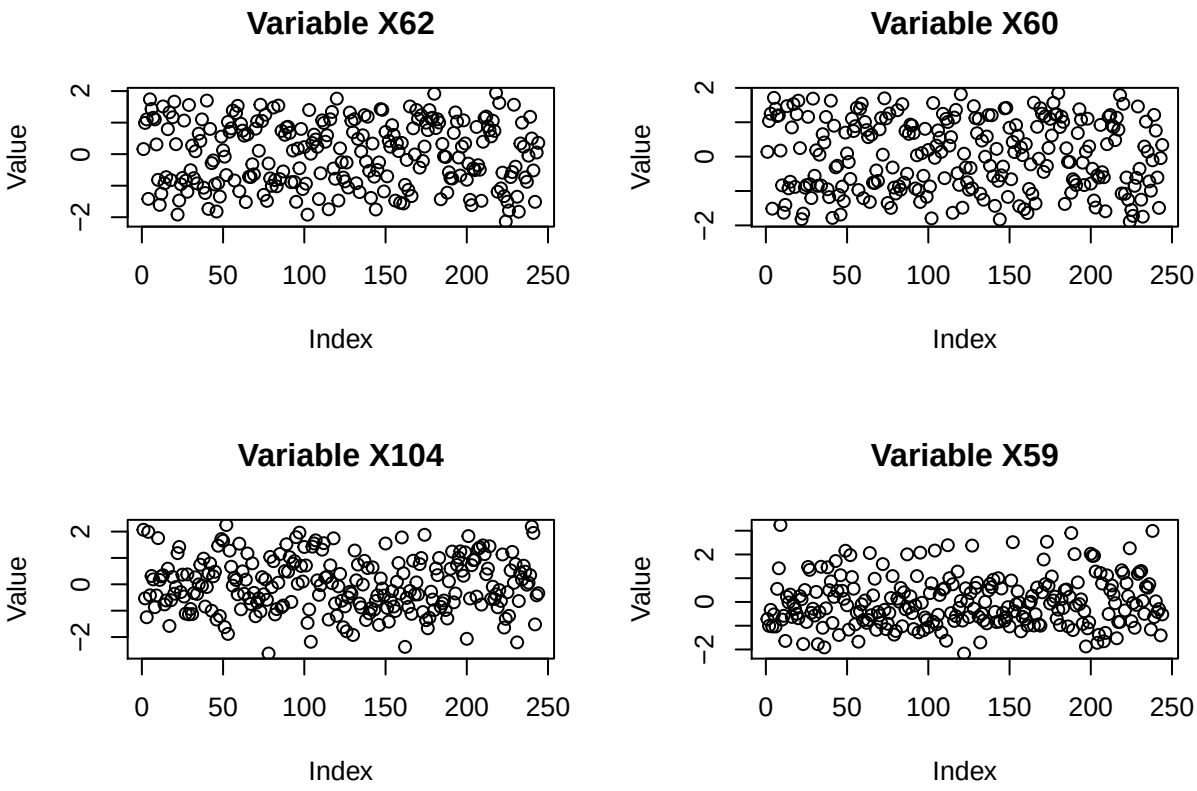

Figure 10: Investigation on the important variables - Quantitive variables

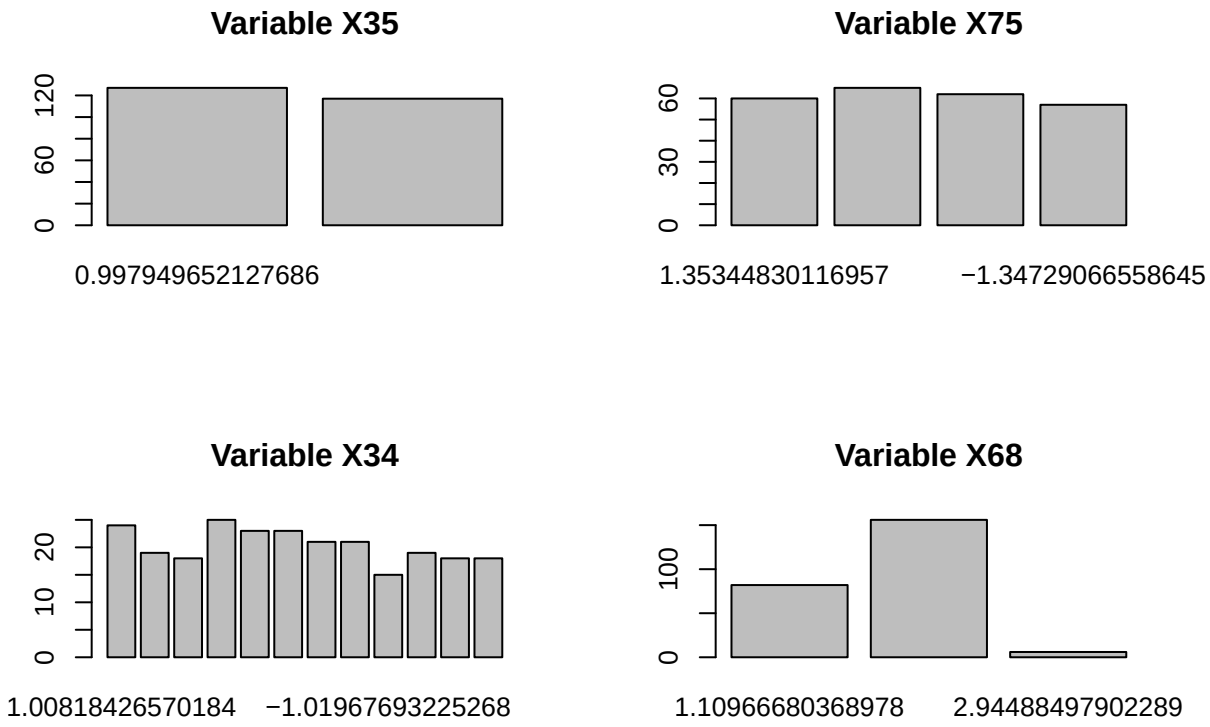

Figure 11: Investigation on the important variables - Categorical variables

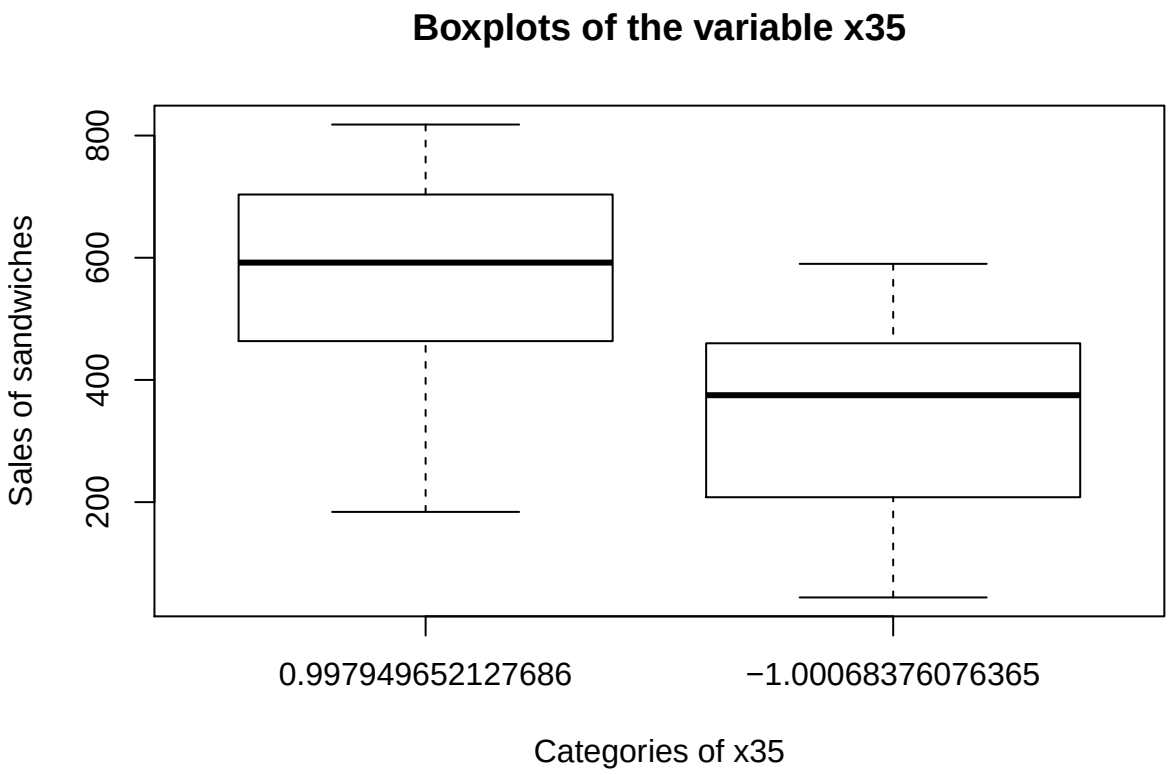

Figure 12: Differences on sales, based on the variable x35

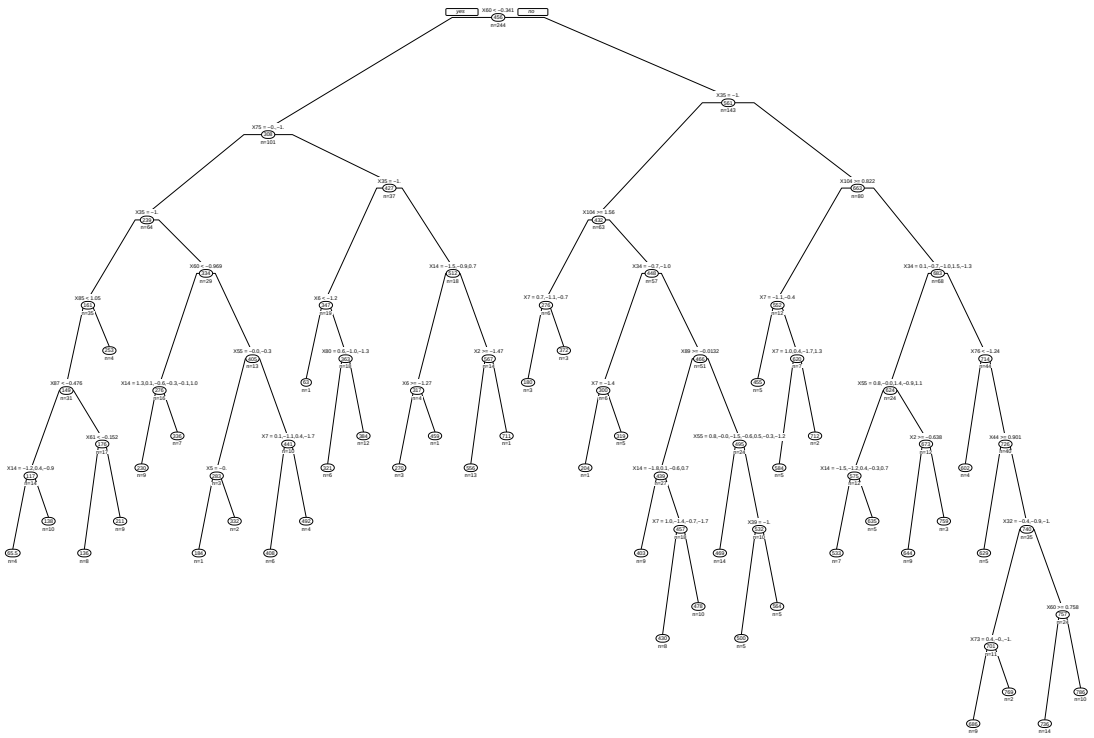

Figure 13: Default regression tree

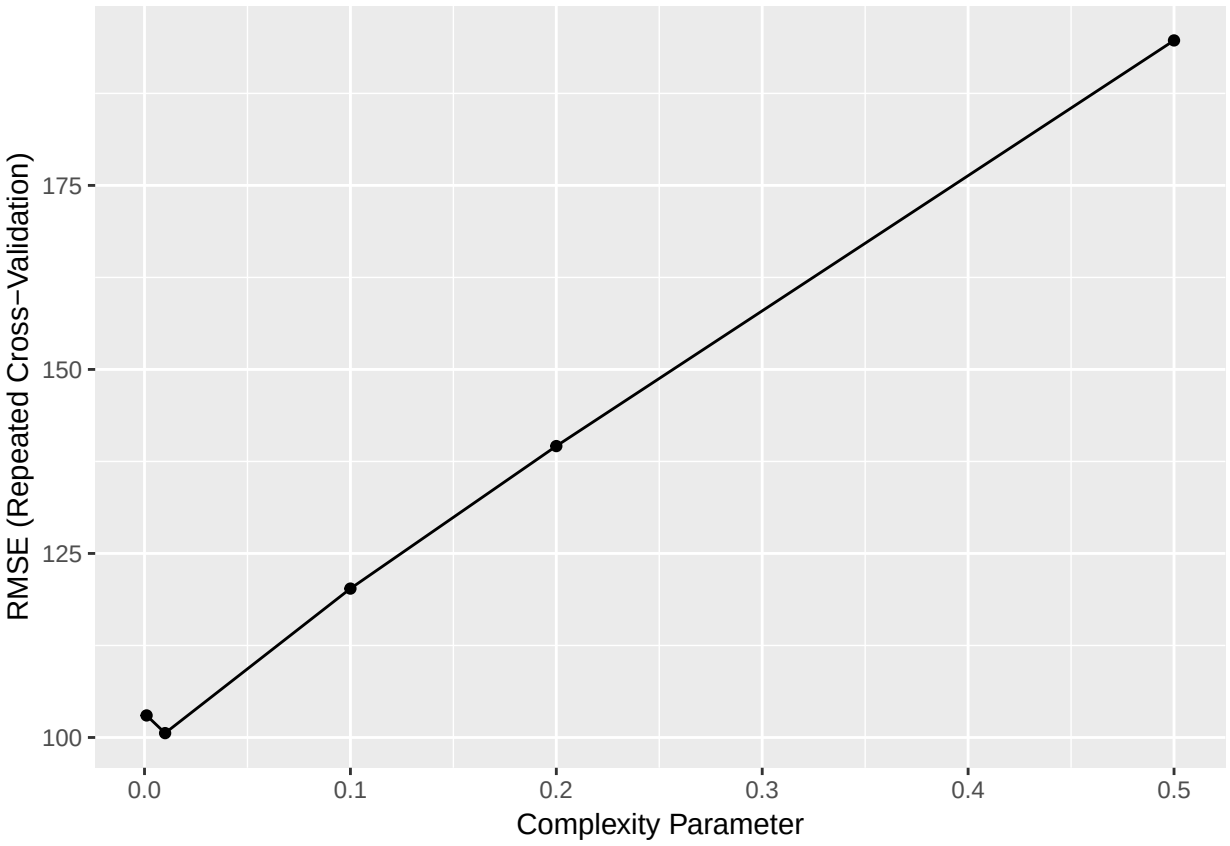

Figure 14: Regression tree - RMSE based on the complexity parameter

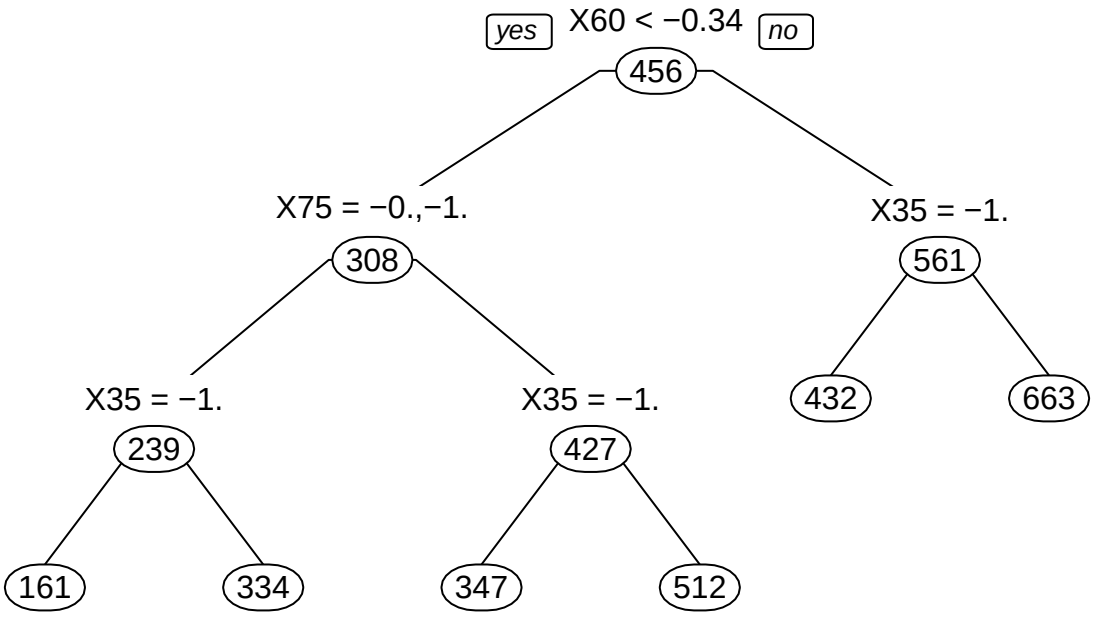

Figure 15: Pruned regression tree