Introduction

In a fast-shopping society, it is increasingly difficult to analyse what determines the success or failure of a product. With the popularisation of the Internet, the E-Commerce sector has also developed, becoming even bigger than traditional shopping. In this highly competitive environment, retailers large and small are having to resort to increasingly innovative methods to ensure that their product is successful.

In this project I aimed to analyse in the first phase whether product reviews present everywhere today in E-Commerce platforms influence the quantity of products sold by making predictions for products sold. How important is it for an E-Commerce platform to have a well set up rating section? How important is it for the seller to encourage and reward the rating received? How important is the price of the product in this equation? These are just some of the questions that this kind of analysis can answer, helping merchants and platforms in the industry.

I would also like to find other predictors with high influence on quantity sold by analyzing a sample of 1543 products of the E-wish retail platform. Is the consumer sensitive to the word discount and price reductions? Can the seller's rating be considered as important as the rating of the product itself? Can I get a good numerical prediction with these predictors?

Whether I refer to the influence of reviews, price or paid promotion there are many reference studies. I looked at some papers from large publications as well as smaller research studies.

For example, in research conducted by Dixa a well-known RPA noted that 93% of shoppers will read reviews before buying a product and that 79% of shoppers consider reviews as important as personal recommendations. Another study, from Northwestern University suggests that having at least five favourable reviews increases purchase rates by 270%.

In the following, I will try to answer these questions using both multiple linear regression and decision trees to get an overview of how certain predictors influence the quantity sold. After that I will use some more complex methods(bagging and random trees) while also making a comparison between methods.

Data set

The chosen dataset is entitled "Sales of summer clothes in E-wish". It comprises 1543 records with relevant data on sales of clothing products in the summer of 2021 on E-wish , a very popular clothing and accessories platform where various people sign up to sell. These products appeared when searching for the keyword "summer" in summer 2021.

The dataset contains 43 columns of both string and numeric type, but for this project I selected only those that I felt were most relevant:

- units_sold - the total number of units sold of a given product;
- Rating - the rating given by customers to the product with values between 1 and 5 (it is actually the arithmetic average of the other ratings);
- Rating_five_count - number of 5-star reviews;
- Rating_four_count - number of 4-star reviews;
- Rating_tree_count - number of 3-star reviews;
- Rating_two_count - number of 2-star reviews;
- Rating_one_count - number of 1-star reviews;
- Merchant_rating - the rating that the seller of the product has, the platform being one where anyone can sell;
- Merchant_rating_Count - number of reviews for the merchant;
- Price - the selling price of the discounted product;
- Badges-count- number of badges of the product;
- Shipping_option_price - number of shipping price options;
- Product_variation_inventory - the stock variation for the respective product;
- Badges_product_quality - if the product has a quality badge;
- Inventory_total - number of products in stock;
- Retail_price - the selling price of the product without discount;
- Countries_shipped_to - the number of countries to which this product is sold;
- Uses_ads_boots - binomial variable representing whether the product was advertised in the form of ads;
- Has_urgency_banner - if it appears with priority in searches.

From the dataset all variables chosen are numeric. Before starting any processing I selected these variables from the datasets.

Results and discussions

In the first phase I selected only the numerical columns because the qualitative ones were not of interest in the analysis. I named the data set "products".I noticed that some variables, although they were of factor type,were listed as numeric, so I used the move function to change their type to factor. In order to interpret the results I applied the summary function on the quantity sold (units_sold).

```
Min. 1st Qu.  Median Mean 3rd Qu.    Max.
1 100 1000 4339 5000 100000
```

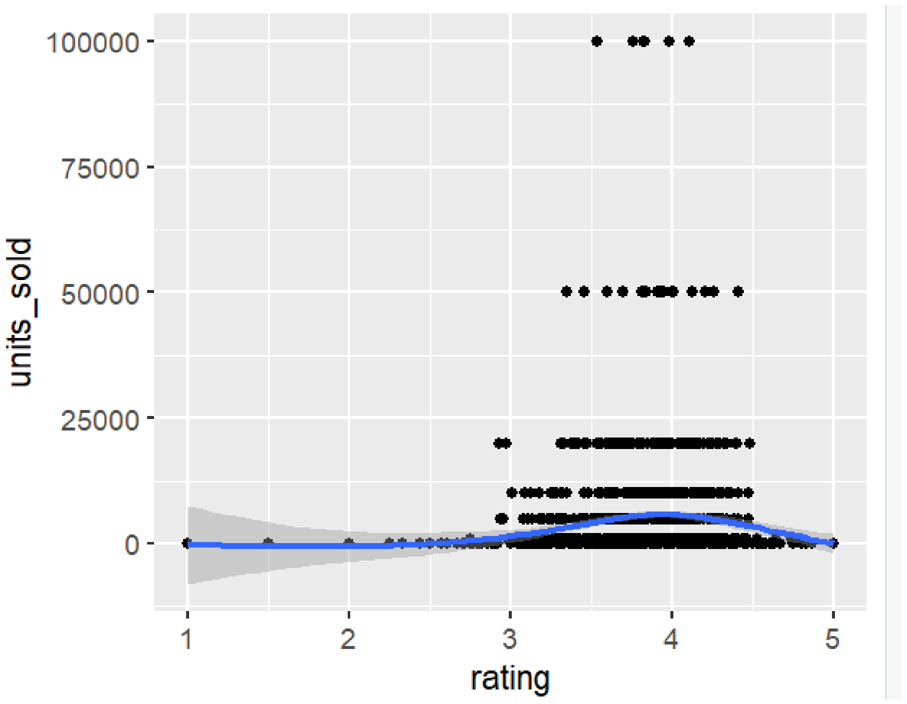
In the following I have tried to find the answer to the questions posed using four methods for numerical prediction:

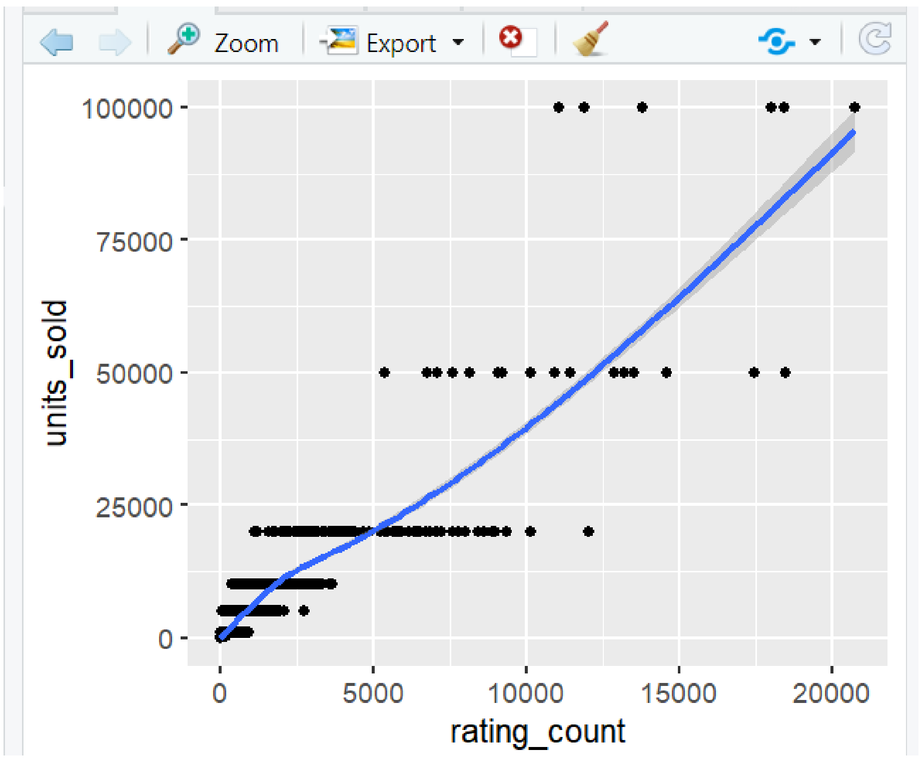
- Multiple linear regression
- Decision trees
- Bagging
- Random Trees

Due to the large number of columns, I decided to first perform a predictor selection using randomForest (I will detail this method below).

To begin with, I made graphs to see if I could observe links between the dependent variable units_sold and some of the independent variables.

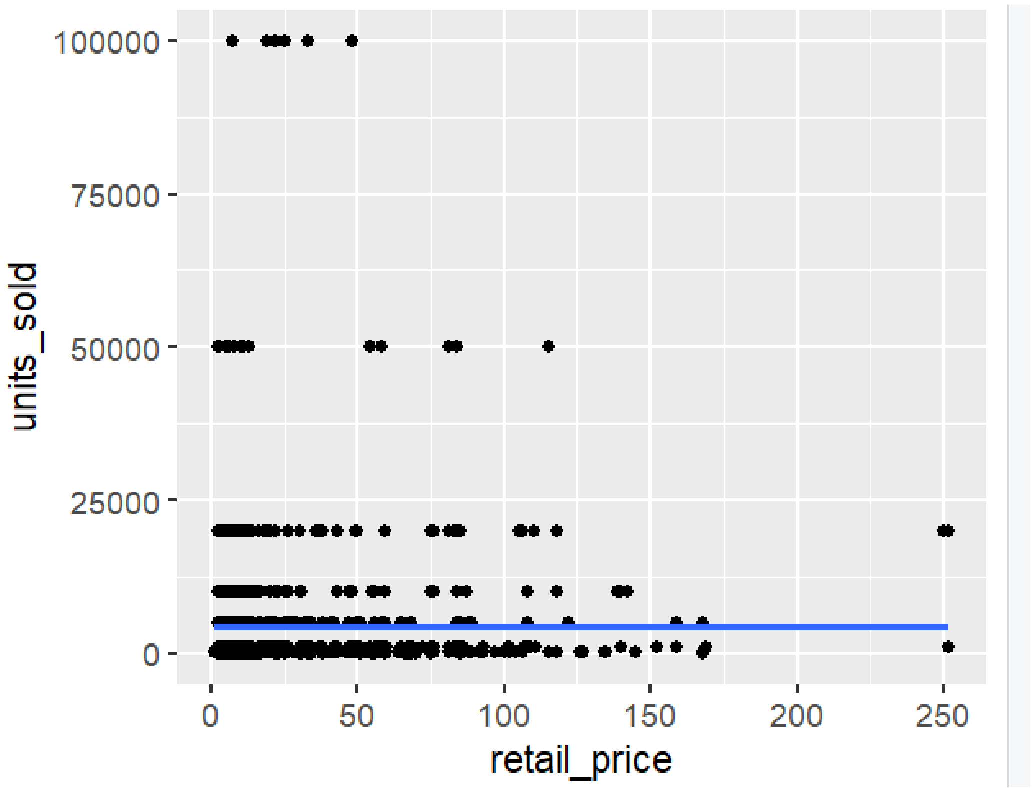
In terms of average ratings for a product, I observed that the highest quantities of products sold are for products rated 4 or close to 4.



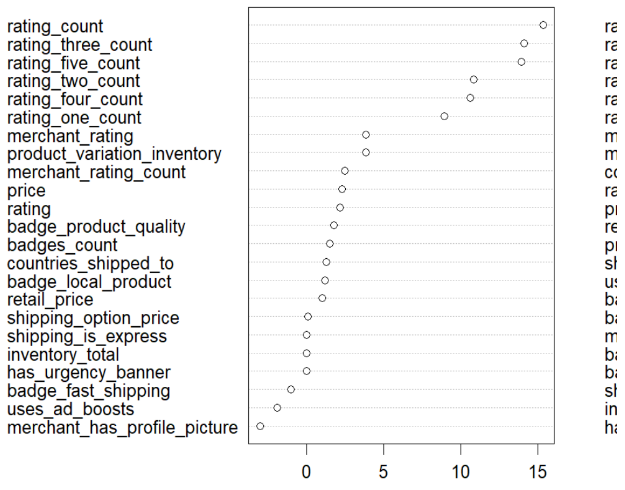


In the units_sold rating_count relationship a direct link is observed. The higher the number of reviews, the higher the quantity sold.

For the sale price, most products sold are around 50 monetary units.



In order to use the randomForest function it was necessary to remove the null values from the dataset so I created a dataset without null values called products_na. After doing randomForest using the randomForest library, I displayed the plot showing the MSE with varImpPlot. From the graph I can see that the most important predictors for quantity sold are the total number of ratings as well as the number of the five rating types, contrary to the results obtained with corrgram.

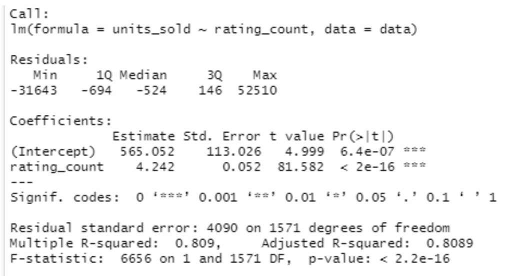


It can be seen that the most important predictors are in order: rating_count, rating_five_count, rating_three_count, rating_four_count, rating_two_count, rating_one_count, product_variation_inventory and merchant_rating_count and only then price and rating.

Linear regression

Although at first glance of the variables in the dataset I thought that the product rating and seller rating might be the most important predictors, I find from the calculations in R Studio that they as independent variables do not influence sales volume very much.

According to the MSE percentages, the **rating_count** variable (the number of ratings for a product) has the highest importance, so I continued with the analysis of this predictor.



```
Call:
lm(formula = units_sold ~ rating_count, data = data)

Residuals:
   Min     1Q Median     3Q    Max 
-31643   -694   -524    146  52510 

Coefficients:
             Estimate Std. Error t value Pr(>|t|)    
(Intercept)   565.052    113.026   4.999  6.4e-07 ***
rating_count    4.242      0.052  81.582  < 2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 4090 on 1571 degrees of freedom
Multiple R-squared:  0.809,	Adjusted R-squared:  0.8089 
F-statistic:  6656 on 1 and 1571 DF,  p-value: < 2.2e-16
```

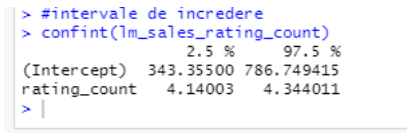
Formula obtained: units_sold ≈ 565.052 + 4.242 × rating_count.

The results in the picture support the importance of this variable. Note that the values of p are very close to 0, so I can consider that the null hypothesis is rejected. The average size by which the dependent variable units_sold can deviate from the regression line is 4090.

Calculations suggest that the number of pieces sold depends 80.9% on the number of ratings, which is a lot.

I also calculated the confidence intervals for the parameters βi with 95% confidence

$\beta i \in [\hat{\beta}i - 2SE(\hat{\beta}i), \hat{\beta}i + 2SE(\hat{\beta}i)]$



```
> #intervale de incredere
> confint(lm_sales_rating_count)
                  2.5 %     97.5 %
(Intercept)  343.35500 786.749415
rating_count   4.14003   4.344011
> |
```

CI for B_0 - [343.355, 786.75]: in the absence of the number of ratings, the number of pieces sold will be in the calculated range.

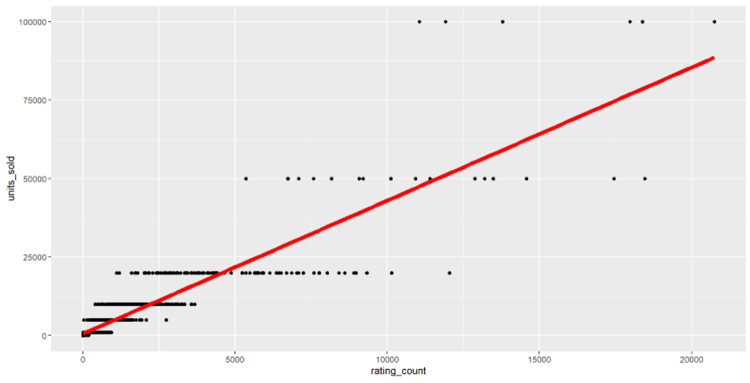
CI for B_1 - [4.14, 4.344]: an increase of 1000 ratings will result in an increase in the number of pieces sold between 4140 and 4344.

H_0 Null hypothesis: No relationship between X and Y is identified : I test $H_0 : \beta_1 = 0$ vs.

H_a There is a relationship between X and Y : I test $H_a : \beta_1 != 0$

Noting that the p-values are small, the null hypothesis is rejected, so there is a relationship between the number of pieces sold and the number of ratings the product has.

Next I generated a new dataset based on the original set, considering 100 values of the rating_count variable for the range 0-20000. Thus, the obtained data and the regression line are marked on the graph below:



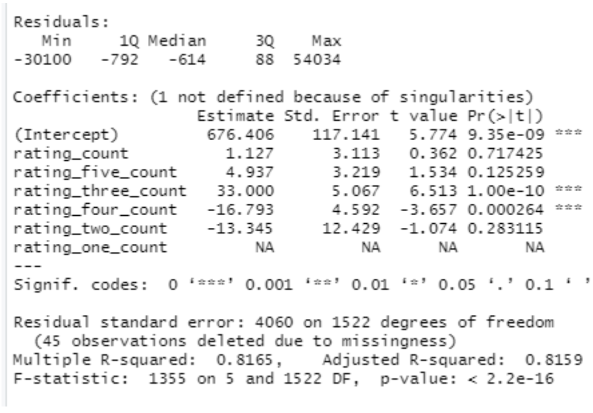
Taking into account the value of R^2 of 80.9% calculated above, I can conclude that there is a strong correlation between the variable units_sold and rating_count.

Next, I chose to analyse linear regression with multiple predictors.

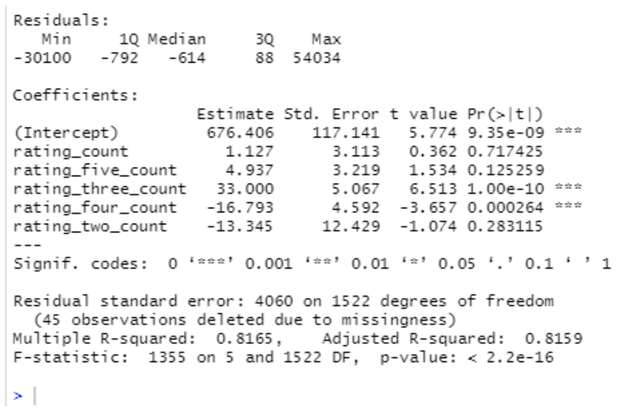
Because the dataset has a large number of variables, I first chose to analyze the first 6 most relevant predictors according to the MSE percentages, since their percentages were closer.

$$\textbf{units\_sold} \approx \beta_0 + \beta_1 \times \textbf{rating\_count} + \beta_2 \times \textbf{rating\_five\_count} + \beta_3 \times \textbf{rating\_three\_count} + \beta_4 \times \textbf{rating\_four\_count} + \beta_5 \times \textbf{rating\_two\_count} + \beta_6 \times \textbf{rating\_one\_count}$$

I have removed the variable rating_one_count because it is not relevant.



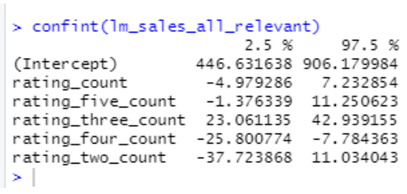
```
Residuals:
   Min     1Q Median     3Q    Max
-30100   -792   -614     88  54034

Coefficients: (1 not defined because of singularities)
                   Estimate Std. Error t value Pr(>|t|)
(Intercept)         676.406    117.141   5.774 9.35e-09 ***
rating_count          1.127      3.113   0.362 0.717425
rating_five_count     4.937      3.219   1.534 0.125259
rating_three_count   33.000      5.067   6.513 1.00e-10 ***
rating_four_count   -16.793      4.592  -3.657 0.000264 ***
rating_two_count    -13.345     12.429  -1.074 0.283115
rating_one_count         NA         NA      NA       NA
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 4060 on 1522 degrees of freedom
  (45 observations deleted due to missingness)
Multiple R-squared:  0.8165,	Adjusted R-squared:  0.8159
F-statistic:  1355 on 5 and 1522 DF,  p-value: < 2.2e-16
```



```
Residuals:
   Min     1Q Median     3Q    Max
-30100   -792   -614     88  54034

Coefficients:
                   Estimate Std. Error t value Pr(>|t|)
(Intercept)         676.406    117.141   5.774 9.35e-09 ***
rating_count          1.127      3.113   0.362 0.717425
rating_five_count     4.937      3.219   1.534 0.125259
rating_three_count   33.000      5.067   6.513 1.00e-10 ***
rating_four_count   -16.793      4.592  -3.657 0.000264 ***
rating_two_count    -13.345     12.429  -1.074 0.283115
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 4060 on 1522 degrees of freedom
  (45 observations deleted due to missingness)
Multiple R-squared:  0.8165,	Adjusted R-squared:  0.8159
F-statistic:  1355 on 5 and 1522 DF,  p-value: < 2.2e-16

> |
```

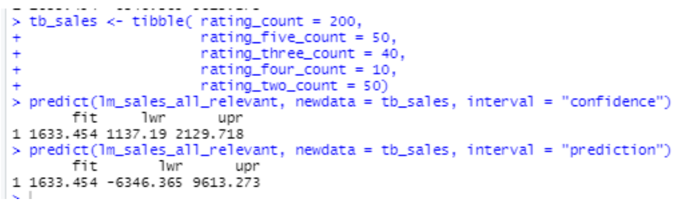
According to the R2 value, the dependent variable units_sold can be explained by 81.65% of the chosen predictors.

Given the confidence intervals for the βi parameters, with 95% confidence, I can see that the number of ratings of 3 positively influences the number of pieces sold the most.



```
> confint(lm_sales_all_relevant)
                         2.5 %     97.5 %
(Intercept)         446.631638 906.179984
rating_count         -4.979286   7.232854
rating_five_count    -1.376339  11.250623
rating_three_count   23.061135  42.939155
rating_four_count   -25.800774  -7.784363
rating_two_count    -37.723868  11.034043
> |
```

At an increase of 1000 ratings of 3, the number of pieces sold will increase between 23061 and 42939.



```
> tb_sales <- tibble( rating_count = 200,
+                     rating_five_count = 50,
+                     rating_three_count = 40,
+                     rating_four_count = 10,
+                     rating_two_count = 50)
> predict(lm_sales_all_relevant, newdata = tb_sales, interval = "confidence")
       fit     lwr      upr
1 1633.454 1137.19 2129.718
> predict(lm_sales_all_relevant, newdata = tb_sales, interval = "prediction")
       fit       lwr      upr
1 1633.454 -6346.365 9613.273
```

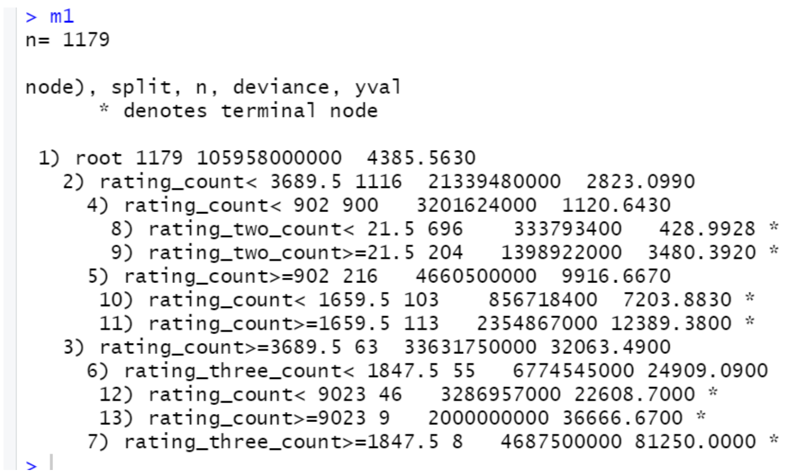
I also performed a prediction, so with the values in the image given for the predictors, the number of pieces sold is estimated to be 1633,454, with confidence interval [1137.19, 2129.718] and prediction interval [-6346.365, 9613.273].

Decision trees

I did not include all variables in the model because for some the MSE percentage was quite small compared to the others. The first step was to constantly divide the dataset into validation data and training data using the rpart library.

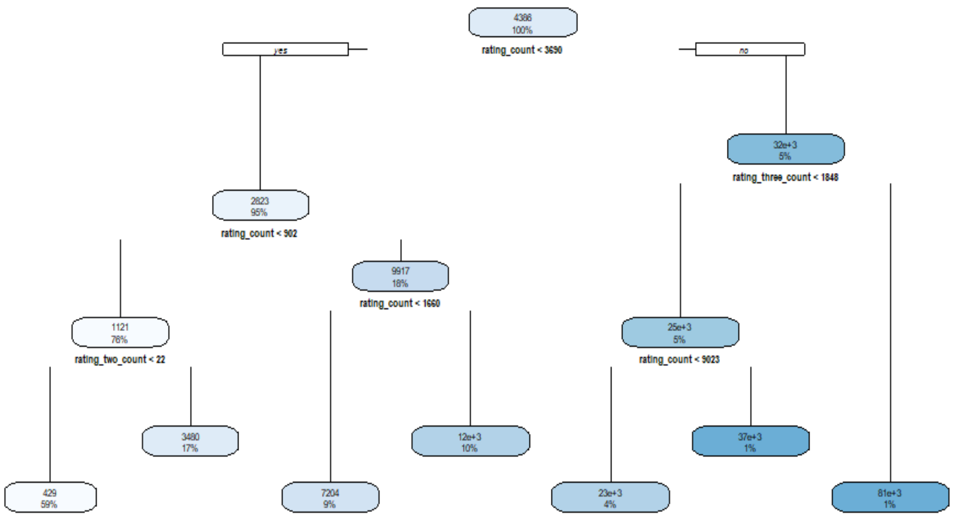
In the following I have considered a restricted version of the predictors including all rating variables. After applying the rpart function with the Anova method on the training dataset, I observed that in the analyzed case the internal nodes are split only by ratings. I obtained 6 internal nodes and 7 leaves.

m1 <- rpart(formula = units_sold ~ rating_count+rating_five_count+rating_four_count+rating_three_count+rating_two_count+rating_one_count+product_variation_inventory+merchant_rating_count+price+rating+badge_product_quality+badges_count+countries_shipped_to+badge_local_product+retail_price+shipping_option_price+shipping_is_express+inventory_total,data = products_train, method = "anova")

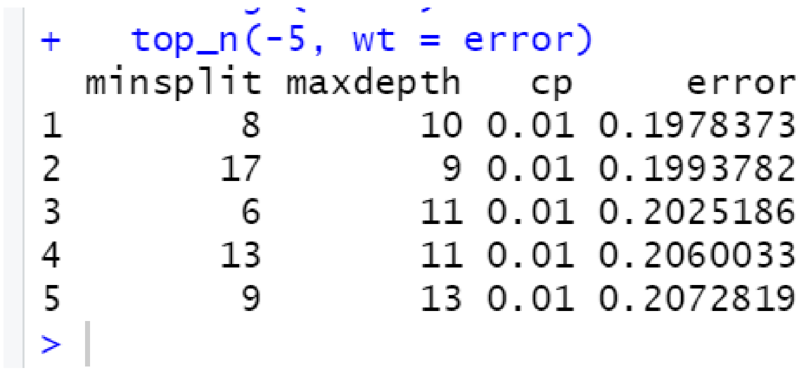


```
> m1
n= 1179

node), split, n, deviance, yval
      * denotes terminal node

 1) root 1179 105958000000  4385.5630  
   2) rating_count< 3689.5 1116  21339480000  2823.0990  
     4) rating_count< 902 900   3201624000  1120.6430  
       8) rating_two_count< 21.5 696    333793400   428.9928 *
       9) rating_two_count>=21.5 204   1398922000  3480.3920 *
     5) rating_count>=902 216   4660500000  9916.6670  
      10) rating_count< 1659.5 103    856718400  7203.8830 *
      11) rating_count>=1659.5 113   2354867000 12389.3800 *
   3) rating_count>=3689.5 63  33631750000 32063.4900  
     6) rating_three_count< 1847.5 55   6774545000 24909.0900  
      12) rating_count< 9023 46   3286957000 22608.7000 *
      13) rating_count>=9023 9   2000000000 36666.6700 *
     7) rating_three_count>=1847.5 8   4687500000 81250.0000 *
> |
```

Using the rpart.plot function I obtained the visual representation of the decision tree. It can be seen that the split at the top level is done according to the condition rating_count less than 3690. In the group in the first leaf there are 59% of observations, and in the group in the last leaf designating the most sold products only 1%. Also, for all internal nodes the condition is done with only one of the three variables: rating_count, rating_two_count and rating_three_count.

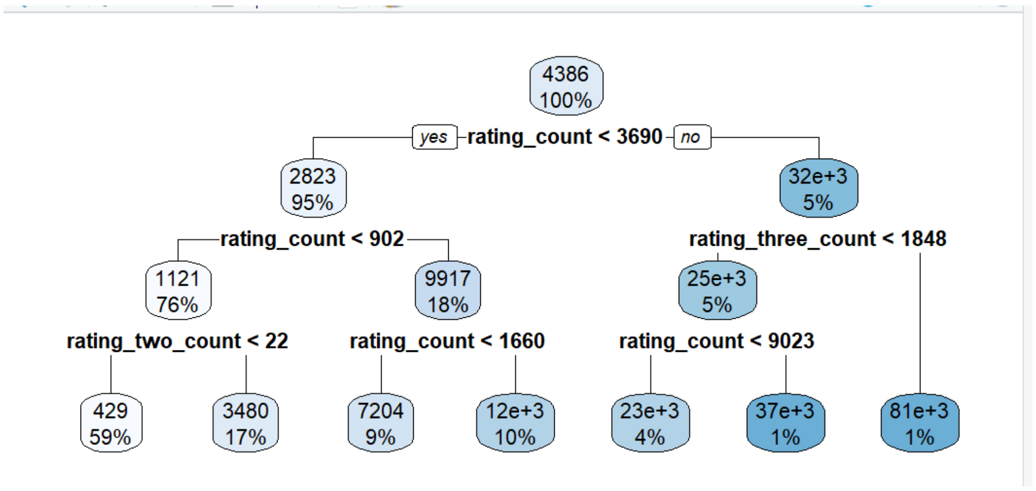


In the following I performed a grid search to automatically search through a range of different models because I wanted to find the optimal parameter setting using the expand.grid function. I then used a function to extract the minimum error associated with each minsplit (minimum number of splits in the tree or internal nodes) and maxdepth (maximum number of levels of the tree) and obtained the following result:



```
+   top_n(-5, wt = error)
  minsplit maxdepth   cp     error
1        8       10 0.01 0.1978373
2       17        9 0.01 0.1993782
3        6       11 0.01 0.2025186
4       13       11 0.01 0.2060033
5        9       13 0.01 0.2072819
> 
```

I chose the first one because it had the complexity parameter (cp) equal to 0.01. After applying the rpart function on the training dataset I obtained the optimal tree. It can be seen that the result for the optimal tree is the same as for the first tree made. This shows that the rpart function realized a good tree even without optimizing first.

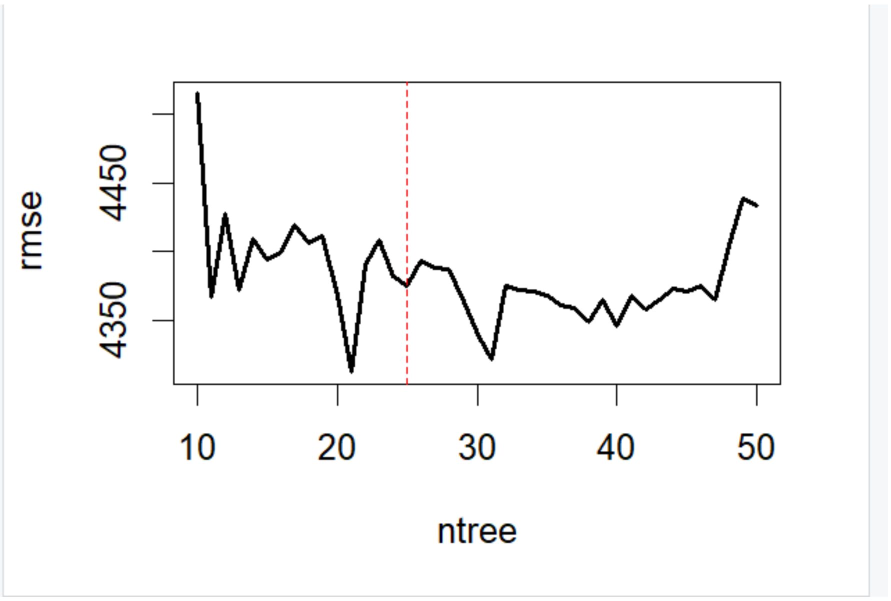


With the predict function I predict the value for quantity sold for the test data set, and then I applied RMSE (root mean square error) to predict the root mean square error and obtained a result of 3995.96. This means that on average the difference between what I predicted and the observations in the data set is 3995.96 units.

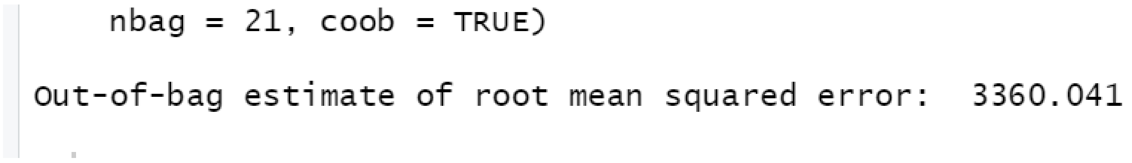
To make sure that the model is a good one I also ran the tree with other predictors (price, retail_price, etc) but I got almost triple RMSE.

Bagging

For bagging I used the products_na dataset and chose 25 for the number of replication copies for bootstrap. The result is an out of bag root mean square error of 4347.332 To make sure that the bootstrap number chosen is the best I try to see what results I get for a vector of 10-50 trees and with a for I check the error for each of them. Then to visualise the difference between rmse I made a graph. From the graph I can see that the rmSE decreases visibly for a number of 21 trees. Which is why I redo the function with out of bag equal to 21.

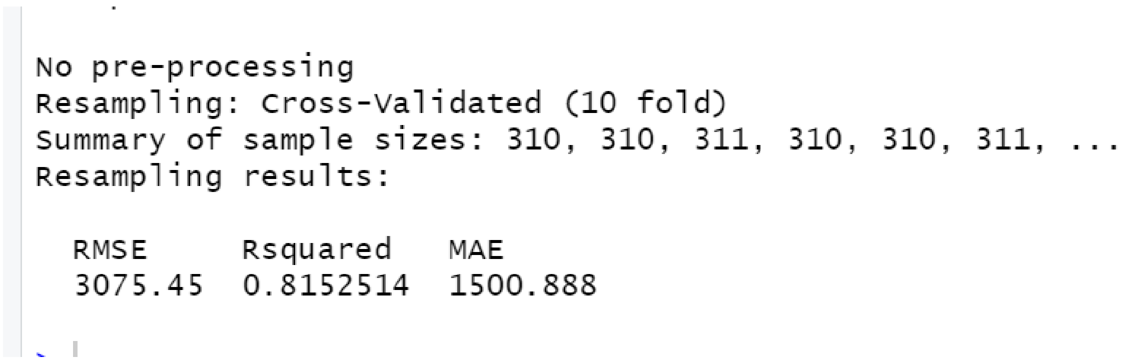


After reapplying the function I get a visibly improved RMSE of 3360.041.



```
     nbag = 21, coob = TRUE)

Out-of-bag estimate of root mean squared error:  3360.041
```

Finally I perform the model with bagging and for method 10 Cross validation. After applying the train method for model learning I obtain the following results:

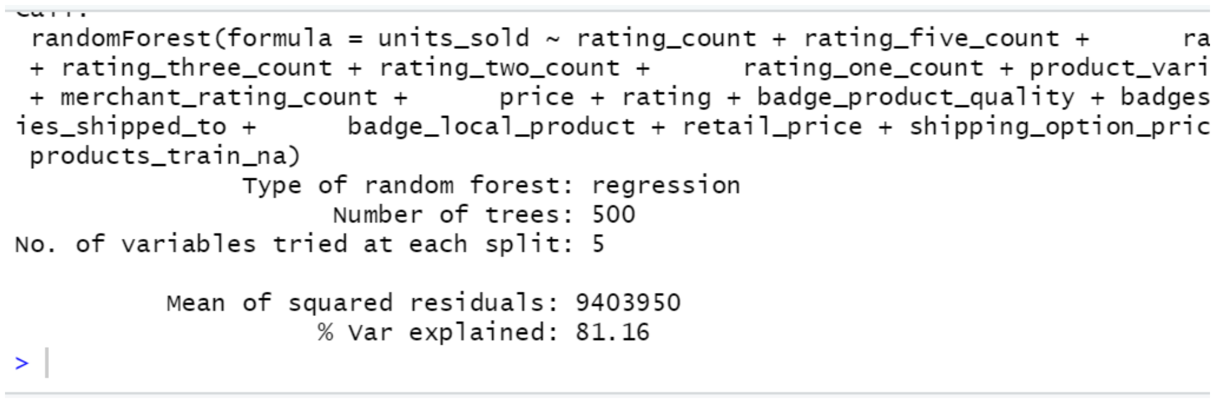


```
No pre-processing
Resampling: Cross-Validated (10 fold)
Summary of sample sizes: 310, 310, 311, 310, 310, 311, ...
Resampling results:

  RMSE     Rsquared   MAE
  3075.45  0.8152514  1500.888
```

The result obtained is even better than for decision trees.

Random Trees

I also used random trees at the beginning to do the predictor selection. The number of records will be smaller because in randomForest I have to omit the null values with na.omit. This gives 460 observations of which 114 for the test and 345 for the model training.



```
 randomForest(formula = units_sold ~ rating_count + rating_five_count +      ra
 + rating_three_count + rating_two_count +      rating_one_count + product_vari
 + merchant_rating_count +      price + rating + badge_product_quality + badges
ies_shipped_to +      badge_local_product + retail_price + shipping_option_pric
 products_train_na)
               Type of random forest: regression
                     Number of trees: 500
No. of variables tried at each split: 5

          Mean of squared residuals: 9403950
                    % Var explained: 81.16
>
```

Due to the fact that the RMSE does not appear in this case, I extracted from the mean squared of residual radical and obtained 3066.58 and that the dependent variable is 81.16% explained by the independent variables I chose.

Conclusion

The rating is indeed the best predictor for estimating the quantity of products sold, but it is not the average rating that is most important but the number of reviews, as well as the number of 5, 3 and 4 star reviews. An interesting finding is that the 3-star rating influences the dependent variable more than the 4-star rating. Also, the number of seller ratings is also very important, but not as important as the product reviews themselves.

Although I would be tempted to believe that the price of the product has a very big influence, it is only by ratings.

All the methods tried showed that the chosen predictors explain more than 80% of the quantity sold, but the most successful was the multiple linear regression where I obtained a fit of 1633.45 and that the dependent variable units_sold depends 81.65% on the chosen independent variables. For the multiple linear regression I chose only rating_count, rating_five_count, rating_three_count, rating_four_count and rating_two_count because the result with the other variables shows that they are not sufficiently important.

In the case of decision trees, bagging and random forest I obtained the following results:

- Decision trees: RMSE of 3995.96 for the optimised decision tree
- Bagging: RMSE of 3075.45, and the model explains 81.52% of the dependent variable
- Random forest: RMSE of 3066.58, and the variable is 81.16% explained.

I can see that in the case of randomForest and bagging with the Cross Validation method the results are very similar and visibly improved from those obtained for the decision trees. RMSE is intended to show us how much the predicted variables differ on average from the actual values in the observation. Given that the mean of units_sold is 4339, but the maximum value can reach 10000, I can conclude that the RMSEs obtained are not very high.