Decision Tree Model for the Graz Housing Market

Sabrina-Sigrid Spiegel

Inhaltsverzeichnis

[Introduction 2](#_Toc508779743)

[Dataset and pre-computation activities 3](#_Toc508779745)

[Code structure 5](#_Toc508779746)

[Data Cleaning 5](#_Toc508779748)

[Data Adding 8](#_Toc508779749)

[Cluster Analysis 11](#_Toc508779750)

[Regression Model 13](#_Toc508779751)

[Random Forest Model 15](#_Toc508779752)

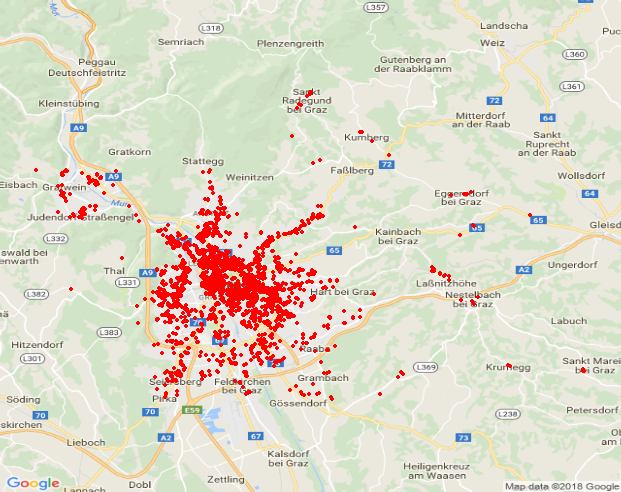
# Introduction

In this paper a decision tree Machine Learning model using a database on housing transactions in Graz is built. The goal is to get a better insight into which variables – or variable combinations – are most important in determining transaction prices of apartments in the Graz area. The outcome of this project will provide a good comparison to more traditional hedonic housing models and could possibly be the first step to the application of more complex ML techniques with this data set.

# An advantage of tree-based models is that they are easy to interpret and easy to implement. They can handle many types of predictors. These models do not require the user to specify the form of the predictors´ relationship to the response like a linear regression model requires. Models based on single trees have two well-known weaknesses: model instability and less predictive performance. Ensemble methods that combine many trees into one model combat these problems. Because of that a Random Forest tree model is used in this context. In Random Forests several trees are generated on the different bootstrapped samples from training data and decorrelated. The variance is reduced by averaging the trees. Building a lot of trees makes the correlation between the trees smaller. At a split on the training data, only a random sample of predictors is considered. This improves the predictive performance (Kuhn and Johnson, 2016).

# Dataset and pre-computation activities

The dataset is provided by the ZT datenforum which collects all transactions that are entered into the Austrian land registry and describes each transaction with up to 34 characteristics. Additionally, each transaction is assigned to one of five main-categories. For this research transactions of apartments in Graz and the surrounding area are used. The data set is based on actually transacted properties and consists of over 6000 observations for the 2014 to2017 period. Following figure gives an overview of the data and how the transactions are located in Graz and the surrounding area.



The first necessary step is cleaning the dataset. Luckily decision tree models are quite insensitive to missing data or skewness issues and can handle many different types of predictors without the need to pre-process them. However, decision tree models can react negatively to autocorrelation between predictors. I will consider this in the selection of predictor variables. Also, in contrast to other ML techniques, decision tree models work best with a limited number of strong predictor variables rather than a large number of input variables. A further reason for a careful selection (and transformation) of predictor variables is that decision tree models have an inbuilt preference for continuous variables and will tend to choose them over more granular predictors (Loh and Shih, 1997, Strobl et al. 2007, Loh 2010, Kuhn and Johnson 2016). Transforming continuous variables into level parameters should resolve this issue.

For these reasons, the following steps will be taken before applying any model:

1) Choose predictor variables

2) Set reasonable thresholds for excluding outliers

3) Construct reasonable levels for continuous variables to counter-act the inbuilt bias for choosing continuous variables

# Code structure

The code in R consists of four main parts. The goal of the first three parts is to clean, improve and extend the dataset. While in the first part the dataset is cleaned, part 2 and 3 extend the dataset with various variables through different methods. The result of the first 3 parts is the database of the essential part, in which models are applied.

## 

# Data Cleaning

After reading the dataset into R, several variables have to be modified that they can be used in R for the analysis. As some integer variables are changed by the code into factor variables, other variables need to be determined as numeric or as character variables. Numeric variables like floor space are cut into groups to be clearer.

*data1a$Kat <- cut(data1a$NutzFl, breaks = c(20, 35, 50, 70, 90, 110, 140, 170))*

Next reasonable thresholds are set to clean the dataset from outliers. One variable for example informs whether a transaction took place under special circumstances (bankruptcy, estate or a transaction between related people). Some of these special transactions are locked out because mostly these prices don´t orientate the market price. A certain range of longitude and latitude has to be determined.

*data1a <- dplyr::filter(data1a, trimws(Verwandtschaft) == "FALSCH")*

*data1a <- dplyr::filter(data1a, trimws(Konkurs) == "FALSCH")*

*data1a <- dplyr::filter(data1a,(preissqm > 800))*

*data1a <- dplyr::filter(data1a,(preissqm < 6000))*

*data1a <- dplyr::filter(data1a, trimws(longitude) > 15.3)*

*data1a <- dplyr::filter(data1a, trimws(latitude) > 47.0)*

The code also tries to handle missing values by several steps. Missing values are filled up with information of other variables. A missing postal code can fill up the name of the district or vice versa.

*subset1$Postleitzahl[subset1$GBName =='Lend'] <- '8020'*

*subset1$Postleitzahl[subset1$GBName =='Algersdorf'] <- '8020'*

*subset1$Postleitzahl[subset1$GBName =='Andritz'] <- '8045'*

*subset1$Postleitzahl[subset1$GBName =='Engelsdorf'] <- '8041'*

Because of the resulting amount of variables and information of these a limited set of characteristics that seems to have the best predictive power is chosen:

* Geographical location: name of district, postal code, cadastral community, longitude and latitude ­­­­­­­­­­­­­­­­­­­­­­­­­­­­­(missing values are excluded)
* Year of transaction: 2014, 2015, 2016 and 2017 (missing values are excluded)
* interior size: Excluding areas below 20m² and above 200m², as well as those with missing values, creating 8 to 11 levels for this size parameter
* Age of building or age of parification:
  + Unknown age
  + Before 1950 and no new parification
  + Before 1950 with new parification
  + 1950 to 1979
  + 1050 to 1979 with new parification
  + 1980 to 1999
  + 2000 to 2017
* Equipment of apartment: terrace, balcony, garden, cellar
* Category of Widmung and maximum building density
* Total costs
* Price per square meters
* Seller is property developer or not

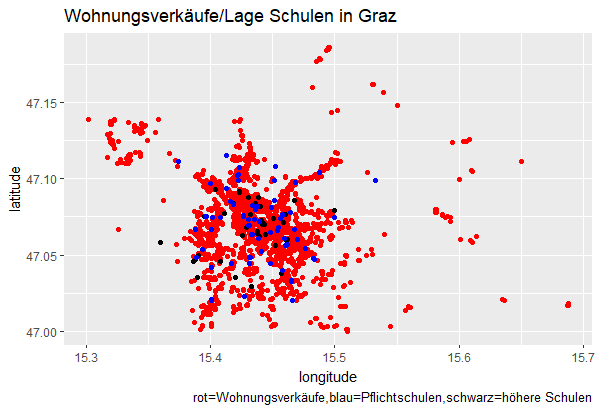
The adjusted and cleaned data set then is exported into a CVS file which is the data base for the next part of the code.

# Data Adding

In order to expand the data base as outcome of the first part of the code to allow for more detailed research results, the cleaned transaction database is now extended by additional characteristics. In particular the code includes publicly available information that influences the quality of the property and thus the sales price:

* Information about the location of different types of schools and the distance to the transacted apartment
* Information about the location of Kindergartens and nurseries and the distance to the transacted apartment
* Park and Ride positions and the distance to the transacted apartment
* Pharmacies positions and the distance to the transacted apartment

To link the new information with the existing data set, the location is introduced into the code through longitude and latitude. The type of the school is specified that primary schools and higher schools can be rated separately. Through linking these two data sets the location of transacted apartments (red) and schools (blue: primary schools and black: higher schools) can be illustrated:



In this context primary schools and grammar schools are considered. In order to calculate the distance of the transacted apartments to these type of schools, first subsets of dataframes with just longitude and latitude coordinates are built to be able to create a distance function between these two data sets.

*Schooldistance2 <- distm(data1b\_coord,data\_Schulen\_coord, fun=distHaversine )*

*Schooldistance2[1,1]*

*Schooldistance2[1,2]*

*min(Schooldistance2[1,])*

*data1b$nearestSchool<- data\_Schulen$NAME[max.col(-Schooldistance2)]*

*X <- Schooldistance2*

*X <- as\_data\_frame(X)*

*colnames(X) <- data\_Schulen$NAME*

*rownames(X)*

*resultA <- (sapply(seq(nrow(X)), function(i) { j <- which.min(X[i,]) }))*

*resultB <- (sapply(seq(nrow(X)), function(i) { j <- min(X[i,]) }))*

*data1b$nearestSchool\_meter <- resultB*

*data1b$nearest\_KAT3 <- data\_Schulen$KAT3[max.col(-Schooldistance2)]*

*head(data1b$nearest\_KAT3)*

*Schooldistance2\_NMS <-*

*distm(data1b\_coord,data\_Schulen\_NMS\_coord,fun=distHaversine )*

*head(Schooldistance2\_NMS)*

*Schooldistance2\_NMS[1,1]*

As a result the nearest primary school and grammar school and its distances to the transacted apartment can be created as new variables in the data set.

*data1b$nearestNMS<- data\_Schulen\_NMS$NAME[max.col(-Schooldistance2\_NMS)]*

For pharmacies, kindergartens and nurseries the procedure is the same and the distances to the next pharmacy, the next kindergarten, the next nursery and the next park and ride position from the transacted apartment are new variables in the data set.

Another price influencing location parameter can be created by calculating the distance from the transacted apartment to the city center. The data set is extended also with this variable.

*Hauptplatz\_coord <- c(15.438391, 47.070794)*

*Hauptplatz\_coord*

*HauptplatzDist <- distm(data1c\_coord,Hauptplatz\_coord,fun=distHaversine )*

*X <- HauptplatzDist*

*X <- as\_data\_frame(X)*

*rownames(X)*

*resultC <- (sapply(seq(nrow(X)), function(i) {*

*j <- min(X[i,]) }))*

*data1c$HauptplatzDist <- result*

The improved data set builds the data basis of the next part of the code.

# Cluster Analysis

After optimizing and extending the original dataset a cluster analysis should group the transactions into clusters that in one cluster the transactions are more similar than in another cluster. The cluster analysis extends the data set with another location parameter so that the area of Graz is not only divided into districts but also in clusters. First 20 clusters are calculated where only longitude and latitude matter. Every transaction then will belong to one of these clusters.

*d= as.matrix(dist(cbind(data\_new$lon\_1,data\_new$lat\_1)))*

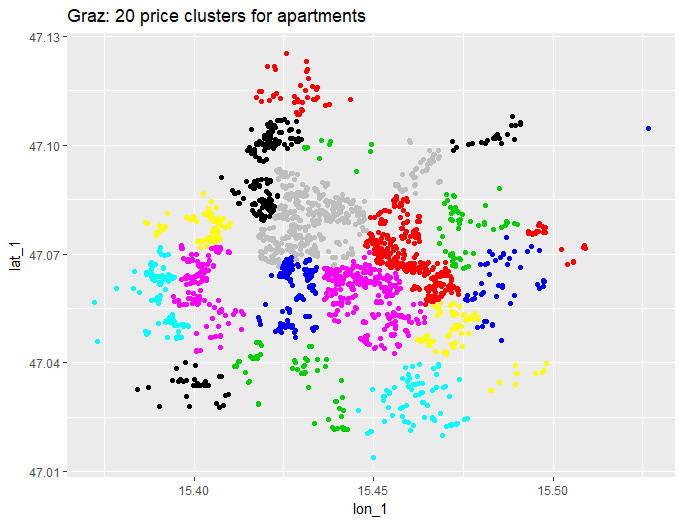
*d=ifelse(d<5,d,0) d=as.dist(d)*

*hc <- hclust(dist(data\_new))*

*clust <- cutree(hc, 20)*

*plot(hc)*

*graz\_clust <- cutree(hc,k=20)*



The mean floor space for every cluster is calculated and also saved as new variable.

*data1c <- data1c %>%*

*group\_by(graz\_clust) %>%*

*mutate(NutzFl\_mean = mean(NutzFl, na.rm=TRUE))*

Because in the first cluster analysis only the location matters, another cluster analysis should assess the price per square meter together with the location.

*daisy1 <- daisy(data\_sample, metric = c("gower"),*

*stand = FALSE, type = list(), weights=c(4,4,1))*

*…*

*d= as.matrix(dist(daisy1))*

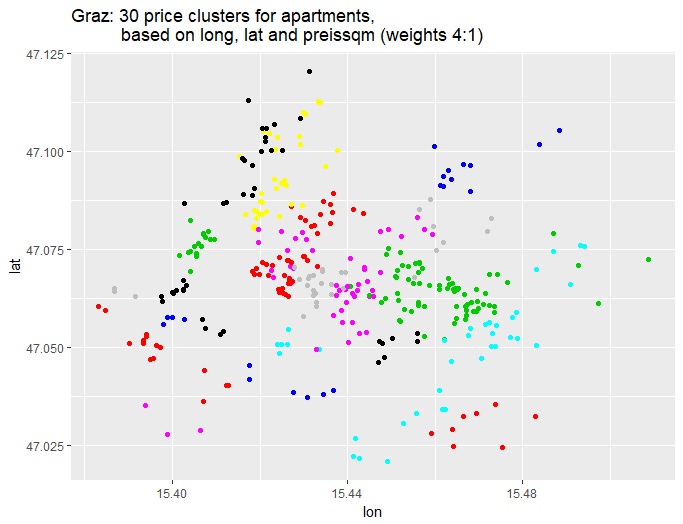
*d=ifelse(d< 0.001,d,0) d=as.dist(d)*

*hc <- hclust(dist(daisy1)) # hierarchical clustering*

*…*

*graz\_clust\_30 <- cutree(hc,k=30)*

Weighting longitude and latitude higher than the price gives following result:



As output of this analysis again the cluster the transacted apartment belongs to is saved as new variable.

# Regression Model

In the last part of the code a random forest model is applied in order to declare which variables are most important to in determining transaction prices of apartments in Graz.

When computing a random forest model in R it is important to avoid variables with more than 53 categories. These variables have to be deselected. Variables with too many missing values are eliminated (maximum building density and Widmung). Colinear variables are reduced to increase the significance of the data set. For the analysis apartments that are sold by property developers are cut to have a more homogenous market. The resulting data set an its variables are now checked for its skewness. An un-skewed distribution is roughly symmetric. A distribution with a large number of points on the left side of the distribution than on the right side is right-skewed. This means there is a greater concentration of data points at relatively small values and a small number of large values. (Kuhn and Johnson, 2016)

All numeric variables of the data set are checked for skewness:

*skew <- sapply(numeric\_columns,function(x){skewness(data4[[x]],na.rm = T)})*

This leads to following result:

*PLZ GstGroesse NutzFl*

*-0.3189388 1.9376709 1.4463829*

*LNGesamtFl BauFlGeb GesamtPreis*

*NaN 2.2435940 0.1132175*

*InventarPreis PkwApPreis SonstigesPreis*

*5.7996658 0.9745437 NaN*

*Steuersatz ParifizierungsJahr KellerFL*

*NaN -4.6853011 0.5984918*

*PKWFl TerasseFL BalkonFl*

*1.6400022 2.0329194 2.0611306*

*GartenFl Gerichtsnr latitude*

*2.3761587 1.3276495 -0.6576707*

*longitude preissqm logpreissqm*

*-1.1520965 0.2674659 -0.2319192*

*preissqm2 loggesamtpreis nearestNMS\_meter*

*0.7740510 -0.4397061 0.1989057*

*nearestVS\_meter nearestApotheke\_meter HauptplatzDist*

*0.2275280 1.2512482 -0.7072884*

*Personen\_total Prakt\_to\_Fach Aerzte\_total*

*0.9993442 1.1522077 4.6350577*

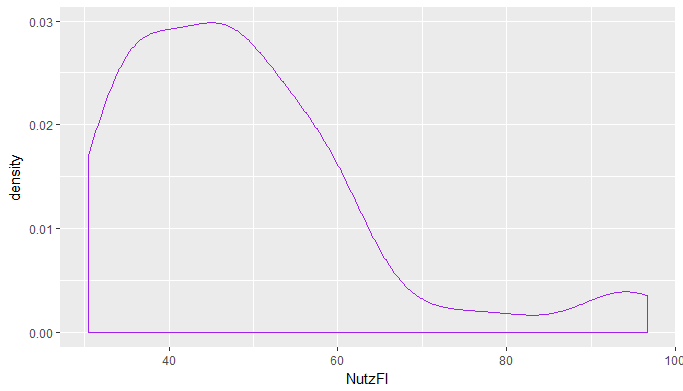
*Aerzte\_ratio Kindergarten\_meter Kinderkrippen\_meter*

*0.3230532 0.4635999 0.9652354*

*ParkandRide\_meter NutzFl\_mean*

*-0.4259278 NaN*

Looking at the skewness parameter of the variable NutzFl, the floor space, shows that more apartments that are sold in Graz have a smaller floor space than the average compared sold apartments sold with a floor space right to the average. The distribution is right skewed (skewness greater than 0).



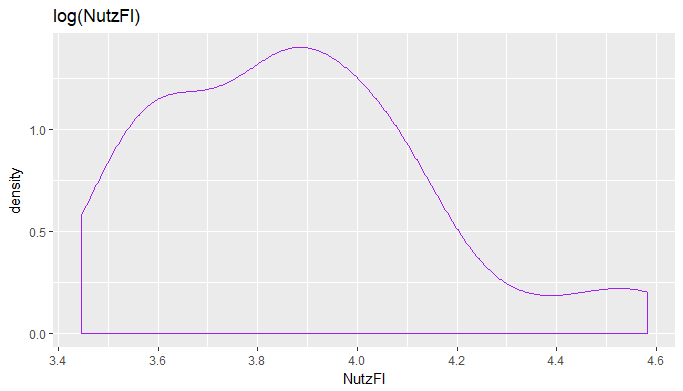
Replacing the data with the log, square root, or inverse can remove the skew (Kuhn and Johnson, 2016). A threshold for skewness is determined and all variables above this threshold are transformed with log(x+1).

*skew <- skew[skew > 0.75]*

*for(x in names(skew))*

*{data4[[x]] <- log(data4[[x]] + 1)}*

Looking at the variable NutzFl, the floor space, replacing it with the log has following effect:



## Random Forest Model

After the data pre-processing the model a random forest model is applied. As the output variable the price of the transacted apartment is used. It´s replaced with the log to resolve the skewness. The distribution of this variable is then not entirely symmetric but it´s better behaved than in natural units. For the predictor variables different combinations of the data set variables will be used to check which variable combinations lead to the highest explained variance.

First the model is applied to the original data set.

*data5 <- select(data4, c("GesamtPreis", "NutzFl", "longitude", "latitude","Parkplatz", "Keller","Balkon",*

*"Garten","Postleitzahl","AlterKategorie","logpreissqm", "jahrdatum", "landverkaeufer",*

*"landkaeufer", "statusverkaeufer", "statuskaufer"))*

The data is separated into test set and training set and then the Random Forest model is applied:

*set.seed(1234)*

*id <- sample(2, nrow(data5), prob=c(0.9,0.1), replace=TRUE)*

*data\_train\_a <- data5[id==1,]*

*data\_test\_a <- data5[id==2,]*

*…*

*my\_forest\_a <- randomForest(log(GesamtPreis)~., data = data\_train\_a,*

*ntree = 501,*

*na.action=na.omit, mtry = 8,*

*importance = TRUE, proximity = TRUE)*

The two main arguments in the formula are mtry for the numbers of predictors that are randomly sampled as candidates for each split and ntrees for the number of bootstrap samples. When the option importance is set equal to true, variable importance scores are not computed. This is too time consuming (Kuhn and Johnson, 2016)

Running the code leads to following outcome:

*Type of random forest: regression*

*Number of trees: 501*

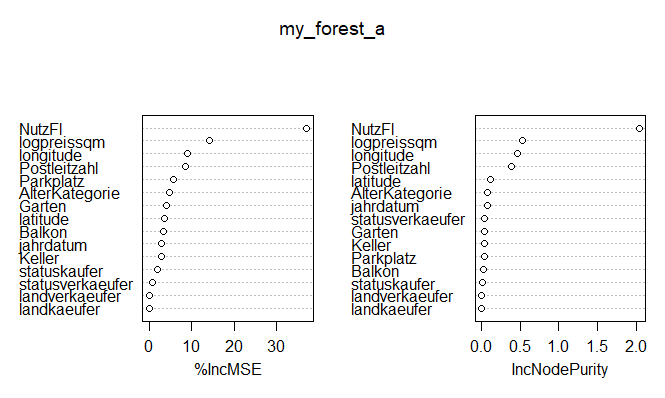
*No. of variables tried at each split: 8*

*Mean of squared residuals: 0.01613551*

*% Var explained: 79.11*

The % Var explained variance measures how well out-of-bag predictions explain the target variance of the training set. In this case we get 79.11 % variance explained with 501 created trees.

Using the formula importance in our random forest model indicates the increase of the Mean Squared Error when the given variable is randomly permuted. This analysis shows that the most important variable influencing the price of the apartment are the floorspace, the price per square meter, the longitude and the postal code. These variables increase the mean squared error by more than 10 %. Information about the buyer and the seller of the apartment have hardly influence.



In comparison to these results the model is now applied to the extended data set resulting from code B and C to check how more information influences the explained variance and how important these new variables are for the price determination.

Adding these variables increases the % Var increased variable up to 81.03 percent.

*Type of random forest: regression*

*Number of trees: 501*

*No. of variables tried at each split: 24*

*Mean of squared residuals: 0.01465113*

*% Var explained: 81.03*

Looking at the importance of the new variables shows that these new variables all have influence on the price determination. Environmental features like the distance to schools and childcare facilities have an influence between 2 and 5 %, the distance to the next pharmacy over 5 %.

