Model-Based Regression Analysis

James

April 18, 2019

Regression analysis is a set of statistical processes for estimating the relationships between a response variable and one or more explanatory variables. It is commonly used for prediction and forecasting, where its use has substantial overlap with the field of machine learning. The primary focus of this report is to discuss some of the advanced approaches applied in regression analysis, regarding their core concepts, practical usage, and some other directions of interest including:

- Deep dive into the data set, analyze its properties and apply them to some practical problems.
- Compare: smoothing, random forest, and boosting in terms of running speed, prediction performance, and other significant aspects.
- Explore: interpret the fitted statistical models, discuss how parameters, variables and other model components may potentially affect the performance.

1. Features and properties of the data

Ideally, tidy data sets are always demanded and preferred for data analysis, as they are easier to work with, without loss of information. In reality, missing data always presents due to either limited access to particular resources, or failure to keep the original records. This leads to the following section.

Preprocessing

Surely, there are many ways to impute missing data. Starting from the basic `mean value replacement`, to `K-nearest neighbor`, and random forest based `mice` imputation, there are certain times when one performs significantly better than the others. The size of the whole data set, the type and proportion of the missing data, are the key to decide which approach to choose. For a small size data with small proportion of missing data, `mean value replacement` operates well. For missing data in factors, typical treatment is to group them into a new category as "None", and add this new level to the factor. The variables treated this way are `ASSESSMENT_SUBNBHD` and `QUADRANT`.[1] For numeric missing data, directly replace them with one of mean, median or mode. The variables treated this way are `YR_RMDL`, `AYB`, `KITCHENS`, and `STORIES`.[2]

It is noticeable that when there exists factors with too many levels, typically over 50, the regression process is slowed down dramatically. Hence, as a safe check, it is necessary to expose properties of all variables,[3] and for those with exaggeratedly large levels, reduce the levels through an approach such that there is no or little information loss. In particular, `SALEDATE` and `ASSESSMENT_SUBNBHD` are the two variables that need treatment. `SALEDATE` should be converted to numeric variable corresponds to date, and `ASSESSMENT_SUBNBHD`, on the other hand, can be excluded in the model since there is a simpler and more crucial version named `ASSESSMENT_NBHD`.[4]

Outlier removal and information gathering

The Id numbers typically do not have correlation with the response variable, because it is a numeric vector from 1 to n in an increasing pattern, so it is commonly dropped from the data.[5] What comes as common sense is that the sale price has a correlation with the area of the property. Figure 1 clearly displays this correlation.[6]

There are two points with a gross building area over 12500 but price under 12500000. That is extremely rare, so it is advised that they are removed.[7] This approach is applied to some pairs of suspicious variables, to detect potential outliers. The treatment for outliers is that they should all be replaced by the means of the variable vectors. There exists other ways to handle them, one of them is to specify a cap like $max(X, 200000)$, but replace them will get the work fairly done. Above is an example of this treatment.

Figure 1: GBA vs Price

Next, a rough view of the distribution of response variable is necessary, as Figure 2 indicates.[8]

Both the head and tail are off the quantile-quantile line. This suggests that we need to apply transformation on price. Compare amongst exponential, logarithm and power function, the best transformation is logarithm. This result is based primarily on the QQ plot, along with skewness and kurtosis, as Figure 3 suggests.[9]

After the transformation, it is crucial to see the correlations between explanatory variables using a correlation matrix, as Figure 4 indicates.[10]

The deep green and red parts correspond to strong positive and negative correlation respectively. The numbers in the bubbles also indicates correlation. This correlogram only records numeric variables, but in the smoothing method, there is special treatment for potential correlation between a factor and a numeric variable. This information is useful for model building, and the details will be discussed later.

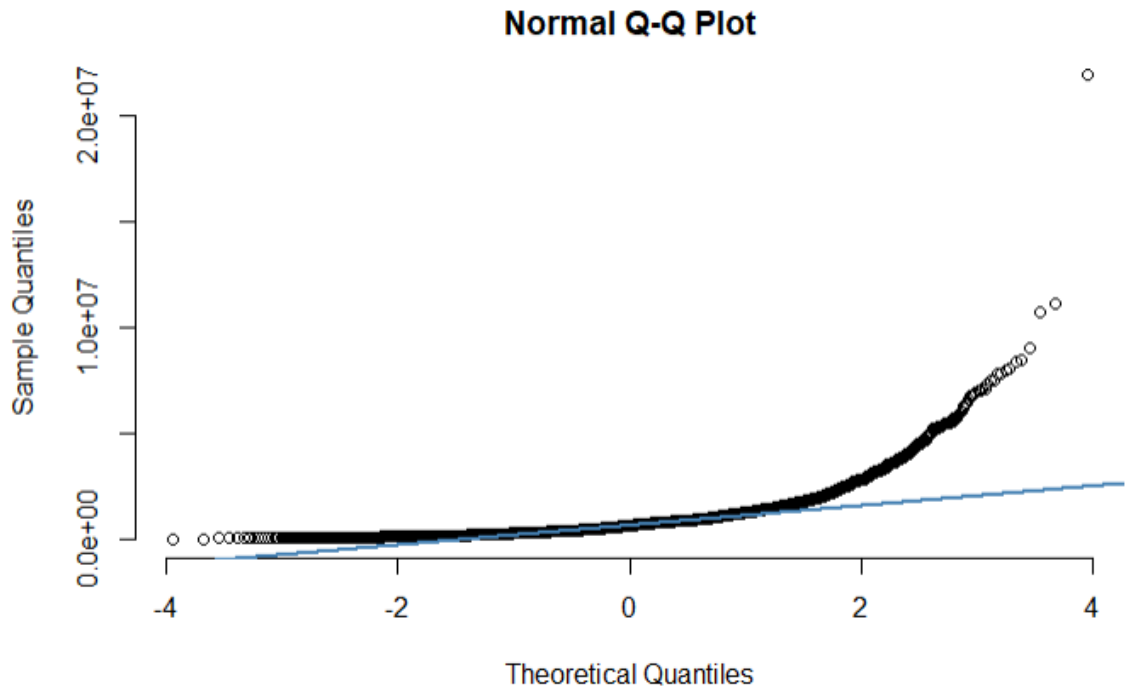


Figure 2: QQ plot for Price

2. Smoothing, random forest and boosting

Brief background knowledge

Assume that we have multivariate observations $y_1', y_2'...y_N'$ that we wish to predict using existing $y_1, y_2...y_N$, along with the features and properties of the other explanatory variates. One of the advanced approaches is to use a smoothing model for x_i,

$$\hat{\mu}(x_1, x_2...x_N) = \hat{\beta}_0 + \hat{\beta}_1 f_1(x_1 - \bar{x_1}) + \hat{\beta}_2 f_2(x_2 - \bar{x_2}) + ... + \hat{\beta_N} f_N(x_N - \bar{x_N}) \qquad (1.1)$$

where $x_1, x_2...x_N$ are the explanatory variates included in the model. In other words, the function is an extension of a linear model. Given a linear model formula, (1.1) will construct a linear model of identical form.

The model built in the form of (1.1), with information in Figure 4 can be used to make predictions. Hence, with all information combined, a model of form (1.1) is eventually built.[11]

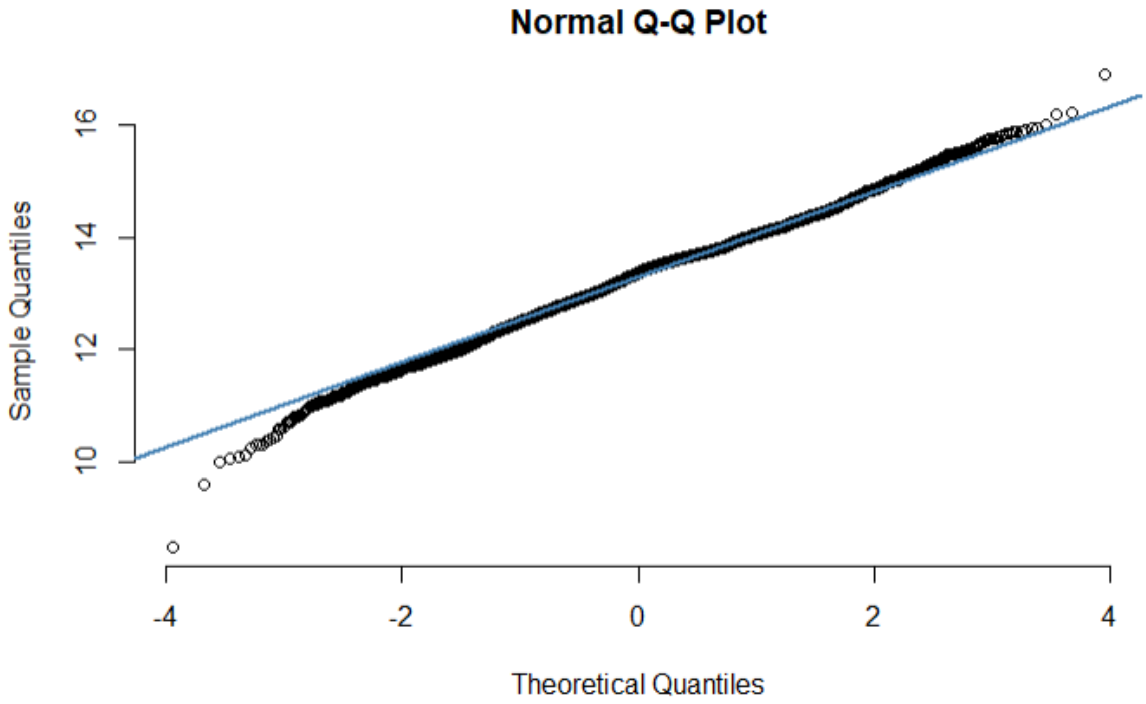


Figure 3: QQ plot for log(Price)

The other two approaches of interest, random forest and boosting, ultimately share some common concepts, but interpret them differently. They both fall into tree based methods category, and essentially form tight connection with linear model. The former starts from bagging,

$$\hat{f} = \frac{1}{B}\sum_{b=1}^{B} f_b(x') \qquad (1.2)$$

where B is the number of the selected random samples with replacement, f_b is the regression tree trained on $X_b = x_1, ...x_N$, $Y_b = y_1, ...y_N$, and x' are predictions for unseen samples. Random forest is sometimes known as "feature bagging", where the only difference is that it selects a random subset of features at each candidate split. The final model is fitted using the concept of (1.2), with parameter tuning to minimize the statistical measure `rmse`.[12]

Boosting, on the other hand, construct a combined prediction estimate,

$$\hat{\mu}(x) = \sum_{m=0}^{M} \beta_m \hat{\mu}_m(x) \qquad (1.3)$$

Random forest and bagging both are a special case of (1.3) where $\beta_m = \frac{1}{B}$ for all m, and $B = M + 1$, where

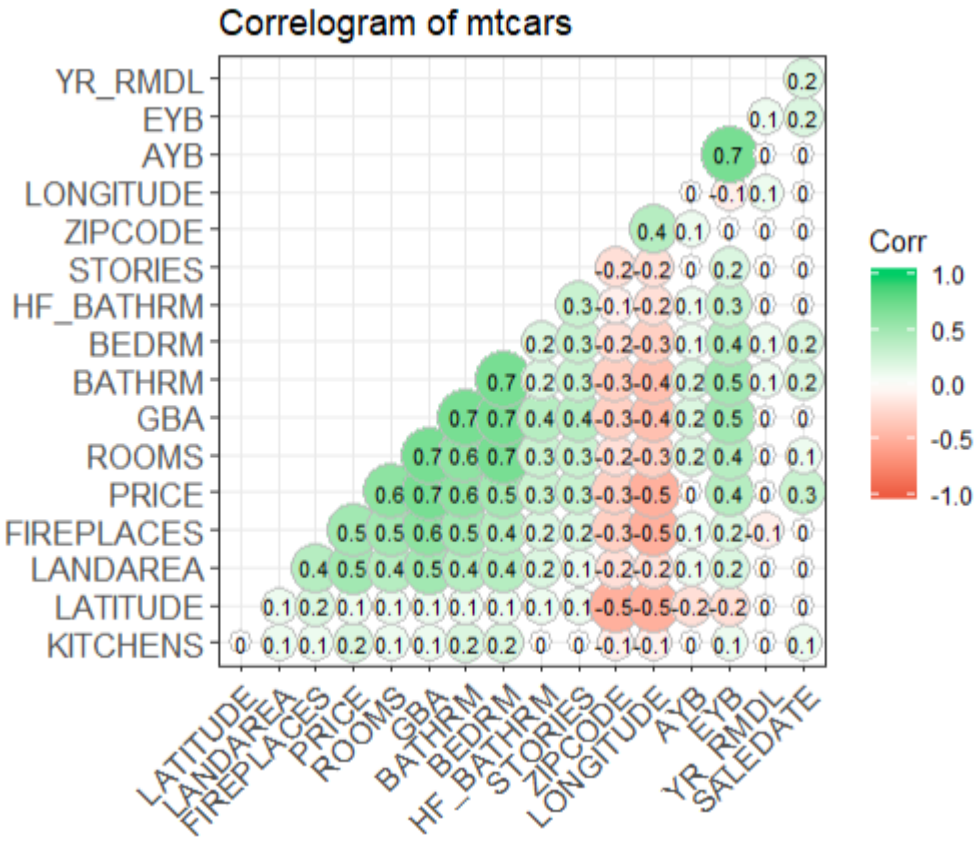


Figure 4: Correlogram

$\beta_0, \beta_1 ... \beta_M$ are parameters of our choice. Following (1.3), combined with the `xgboost` package, a boosting model is built with parameters carefully tuned.[13]

Comparison

There is one thing worth mentioning is that `xgboost` does not accept a data frame. It requires a matrix, dgCMatrix, or name of a local data file as a parameter. In other words, it only works with numeric variables. Hence, converting factors to numeric variables is a must before fitting.

The packages used to obtain the above three models are `mgcv`, `ranger` and `xgboost`, respectively. The fitted models are not necessarily the best outputs from the packages. Parameter tuning, variable interactions and other aspects can lead to different, and better models. There is method to find variable importance in `ranger`, which makes it convenient to decide which variables should be included in the model, as well as compare the result with the `backward selection`.[14]

Figure 5 is used for comparing different statistical approaches.

	Approach	Public score	Private score	Complexity	Preprocessing	Running time(s)	Stability
1	Smoothing	0.19344	0.17499	Complex	Simple	510	No seed needed
2	RF	0.19817	0.18131	Simple	Mild	7	Seed needed
3	Boosting	0.18372	0.17370	Mild	Complex	20	Seed needed

Figure 5: Comparison Table

Smoothing is rewarding, especially when the public score remains high, but eventually the private score drops down dramatically. Despite of the complexity of the model and the incredibly long running time, it has a pretty good prediction accuracy and fairly simple preprocessing. There is no seed required for this approach, which makes it fairly stable, and no need to worry about the impact of seed number on the performance of the model.

Random forest has the lowest prediction accuracy among all three approaches, but there is space for improvement. The model is tidy, because there is no need to manually build the interaction terms, unlike smoothing. With a running time of roughly 7 seconds, it holds the most efficient approach among the

others. Seed number has a potential impact on the performance of the model.

Boosting has the best prediction accuracy. The model is not too complex, but there are more parameters involved than random forest. The running time is considerably short compare to smoothing. However, due to specialarity of the implementation, a more complex preprocessing is required.

3. Practical usage of the data

Generally speaking, it is more expensive to buy a house nowadays than in the past 20-30 years, as Figure 6 suggests.[15]

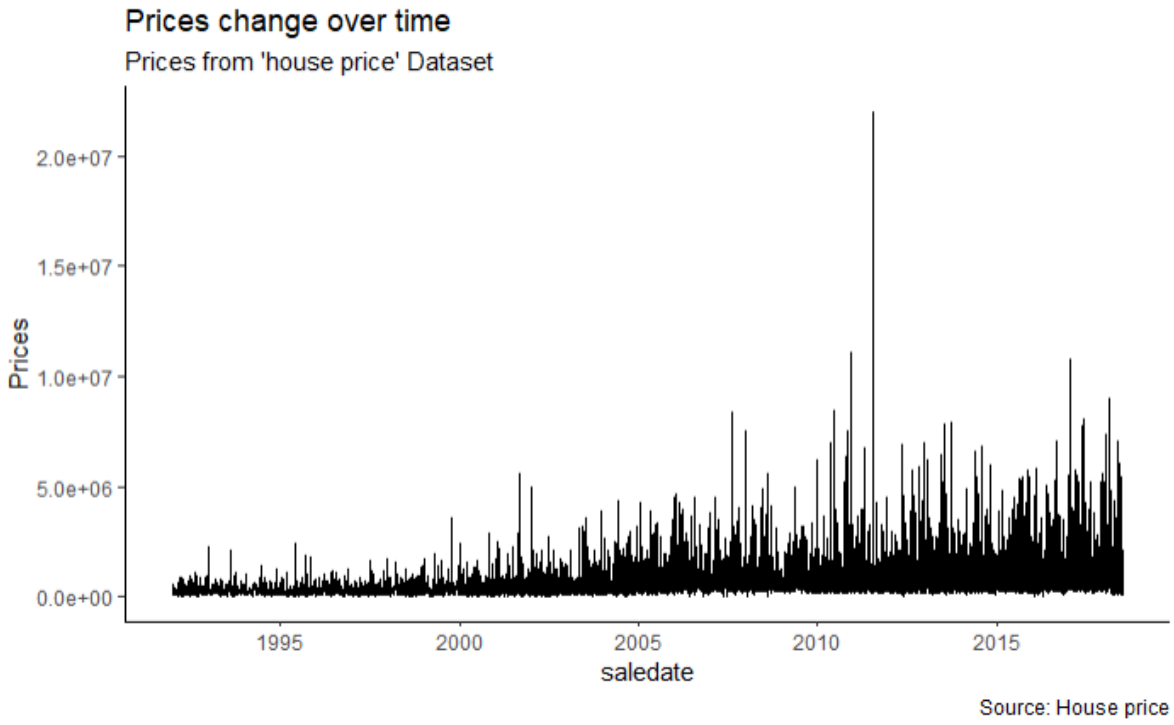


Figure 6: Prices vs sale dates

Obviously the shaded area from the beginning to 2005 is smaller than that from 2005 to the end, which indicates that the house price as a time series, in general had a positive increasing trend over the past few decades. While prices were increasing, some other features of houses were changing at the same time with respect to prices. As suggested by Figure 4, `GBA` has the strongest correlation with `Price`. Figure 7 is used to reveal this trend.[16]

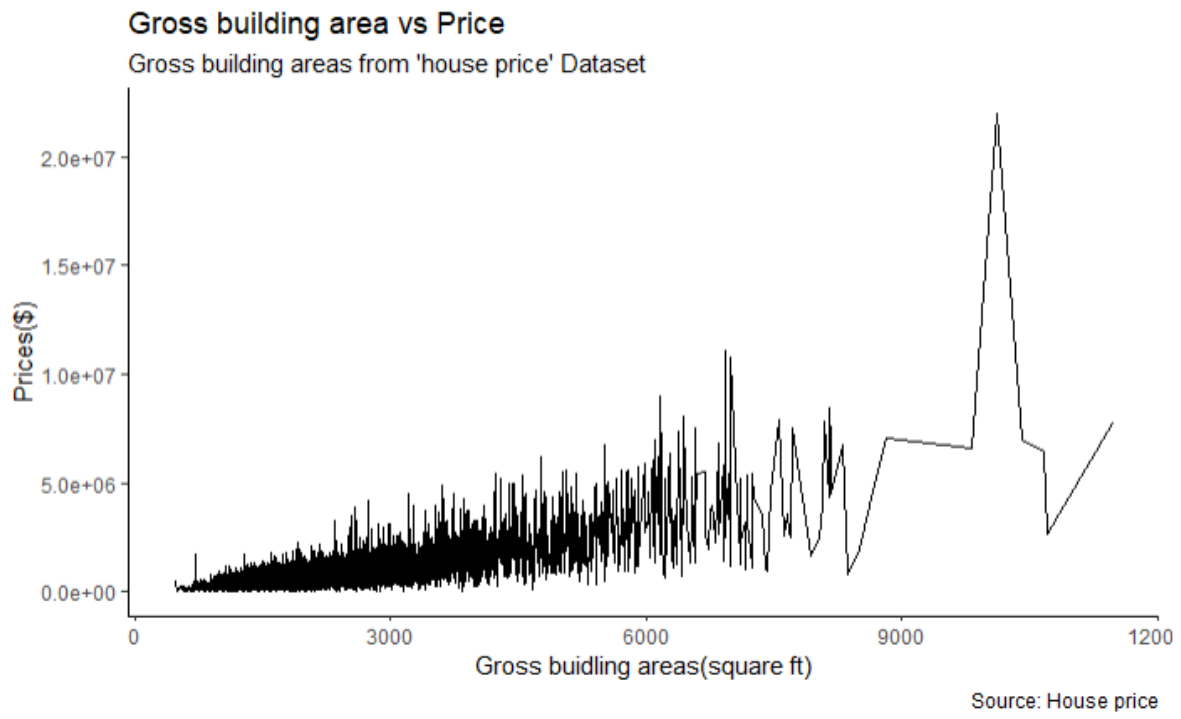


Figure 7: Trend of prices over GBA

This shows an increasing trend, meaning that in general, the larger the building, the higher the price. However, the house with the highest house price, does not necessarily have the largest building area. There is not enough evidence to show that it could be considered as an outlier, but there must be something else that makes this house the most expensive one amongst all the others. Hence, it is interesting to take a closer look at the detailed information about this house.[17] From the information, it can be conluded that this is a mansion, with incredibly large amount of rooms, and it is an old house. Sold in 2011-07-25, it was graded well, but it did not have an outstanding condition. It was not located at a significantly better area, because there were some other houses in the same area that were priced extremely low. The main winning point of the house is that it is fairly historical, and there are a lot of rooms. That is the primary reason why this can usually be a type of outlier, because deals like this do not always happen. Generally speaking, the seller may get a similar deal once in many years, so it is not a typical house an agent would normally choose to sell.

Practically, some of the outstanding deals should have the following features. In short, after a buyer buy the house, there should not be any complaining about how unfair the price was set, because all the other

excellent features make the house absolutely worth it. In other words, it should be graded well, the condition is fine, and should not be too aged unless it was rebuilt and improved recently. The evidences are shown as following.

As previously mentioned, all the deals that were graded either `Excellent` or `Superior` should be grouped together to find some common features.[18] Fortunately, there are a number of features that these houses all share.[19] By obtaining the mode of every explanatory variables in the grouped data set for good houses, the common factors are revealed as follows. For `HEAT`, most of these houses have the `Warm Cool` option. Ac is normally required, and generally they should have two stories in primary dwelling. A `Good` condition is enough to get an incredible grade. `Common Brick`, `Slate` and `Hardwood` are normally used for exterior wall, roof and interior wall, respectively. And what's more, we should be moderately confident to state that, as most of these good houses are in the neighbourhood `Berkley`, it should be a fine neighborhood to live in, and a housing agent would love to get some deals in `Berkley`. However, numeric features are not fit to be discussed in terms of mode, because there can be much more variabilities and results than factors, which makes the mode not accurate enough, so numeric features will not be included here.

4. Model Analysis

The software packages have already saved countless efforts and time for model building, but the parameter tuning, variable selection have to be manually constructed.

Variable importance

It is not a smart strategy to exclude all the relatively insignificant variables staight away. It requires experience and analysis to determine how important a variable really is. There are many variable selection methods, including filter method, wrapper method and embedded method. All these methods are conceptually the same, as they all try to select the best subset of variables from the whole set. Before the smoothing model was eventually fitted, three variable selection techniques were implemented and compared, in which all of them belong to the wrapper method category: forward selection, backward elimination and hybrid selection.[20] The output indicates something similar to the variable importance log implemented

using the `ranger` package.[14]. The difference is due to the different methodologies of the two approaches. The `SALEDATE` is the most important variable, followed by `LONGITUDE`, which is surprising, and then `GRADE`, `GBA`, etc. The result suggests that we should not use variable importance blindly without understanding the data.

Base on the experience from the smoothing fitting, variable selection improves the prediction accuracy. The code are included here.[21] The model after variable selection scored a 0.19629, and the model without variable selection scored a 0.20871, which is a huge difference.

Hypergrid parameter tuning

This is a practical technique for both random forest and boosting models.[22] Given the ranges of the parameters required, a hypergrid makes a matrix of all possible combinations among parameters, and fits models on them in order to compute statistical measure such as `OOB_RMSE` for assessment. Then the best n results can be ordered in terms of a specific criteria, so the output is clear and accurate. However, this method is time consuming. Another drawback of this method is that it requires a mildly efficient starting point, because it is inefficient to set a incredibly large matrix and wait for the results. In order to get a nice starting point, thoroughly under stand the model, and fairly know all the explanatory variables should be required.

Advanced Interactions

Additionally, adding interactions between factors and continuous variables may greatly boost the model prediction accuracy. This type of interactions cannot be obtained directly by observing Figure 4, but through experience and common knowledge. What naturally comes to mind is that there should be correlation between `CNDTN` and `AYB`. Figure 8 vividly reveals this correlation.[23]

The lines are crossing, and different levels of the factor are indeed correlated with the numeric variable. Similar interactions can be detected mainly through the plots. Another example is the interaction between `EYB` and `CNDTN`, as Figure 9 indicates.[12]

Figure 10 indicates another important interactions.

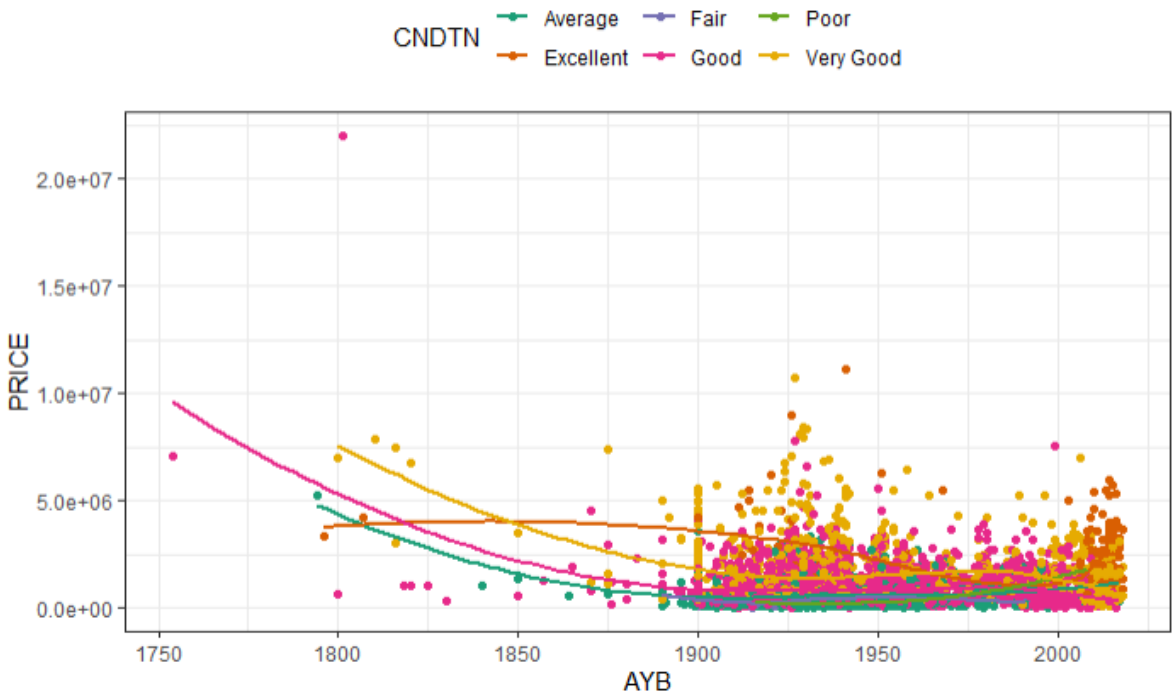


Figure 8: AYB vs Price, by CNDTN

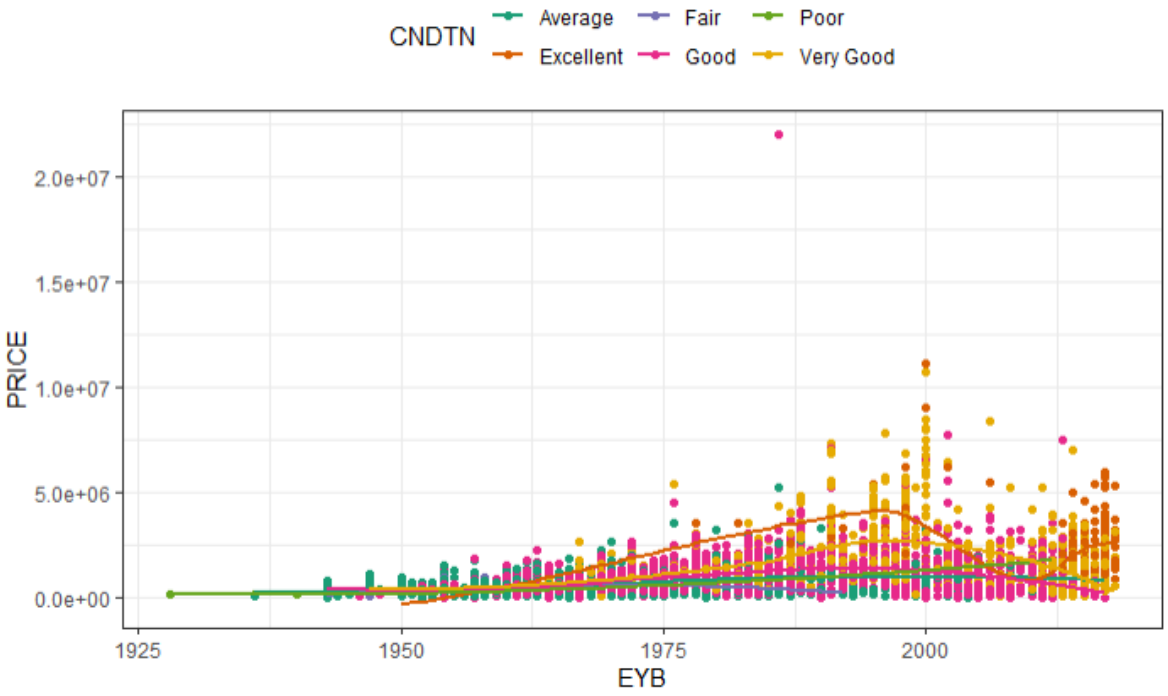


Figure 9: EYB vs Price, by CNDTN

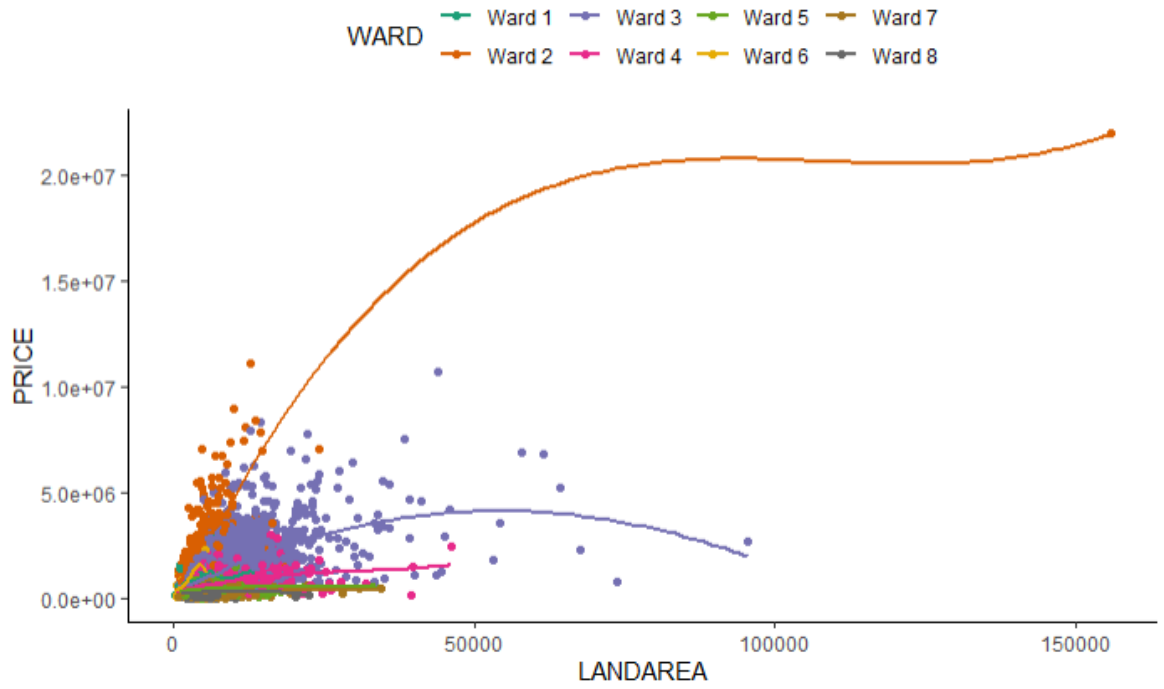


Figure 10: Price vs land area by ward

Conclusion

The most expensive house sold in record was built early, but repaired and reconstructed later. It has much more rooms than a regular house should have, indicating that it could be a mansion. Top graded houses are mostly sold in the neighborhood `Berkley`, and the most popular exterior wall, roof and interior wall choices for these houses are `Common Brick`, `Slate` and `Hardwood` respectively. There is AC in most of these houses, and the most popular choice of `HEAT` is `Warm Cool`. In general, the gross property area is positively correlated with the sale price, but there are exceptions. Sale date affects the sale price the most among all the other variables.

There is no universal rule of success in regression analysis. There are only better ways to handle different situations. Variable selection, interaction, parameter tuning, all of them require careful adjustment in order to improve the model prediction accuracy. It is crucial to gather basic information about data before the model fitting. In conclusion, if getting the job done efficiently and accurately is the main goal, either random forest or boosting can satisfy this requirement. If understanding more about the data frame, correlations between

explanatory variables and the response variable is the main goal, smoothing is statistically challenging and awarding.

Appendix

```
load("final.Rdata")
```

[1] Extra "None" level treatment

```
levels <- levels(dat$ASSESSMENT_SUBNBHD)
levels[length(levels) + 1] <- "None"

dat$ASSESSMENT_SUBNBHD <- factor(dat$ASSESSMENT_SUBNBHD , levels = levels)
dat$ASSESSMENT_SUBNBHD[is.na(dat$ASSESSMENT_SUBNBHD )] <- "None"

levels <- levels(dat$QUADRANT)
levels[length(levels) + 1] <- "None"

dat$QUADRANT <- factor(dat$QUADRANT , levels = levels)
dat$QUADRANT[is.na(dat$QUADRANT )] <- "None"
```

[2] Mean replacement method

```
dat$YR_RMDL[is.na(dat$YR_RMDL)] <- mean(dat$YR_RMDL, na.rm = TRUE)

dat$AYB[is.na(dat$AYB)] <- mean(dat$AYB, na.rm = TRUE)

dat$KITCHENS[is.na(dat$KITCHENS)] <- mean(dat$KITCHENS, na.rm = TRUE)

dat$STORIES[is.na(dat$STORIES)] <- mean(dat$STORIES, na.rm = TRUE)
```

[3] Data information

```
str(dat)
```

```
## 'data.frame':    12474 obs. of  31 variables:
##  $ BATHRM           : int  2 2 2 2 3 3 1 2 2 2 ...
##  $ HF_BATHRM        : int  1 1 1 1 1 1 0 0 0 1 ...
##  $ HEAT             : Factor w/ 13 levels "Air Exchng","Air-Oil",..: 12 6 12 6 12 12 8 6 6 6 ...
##  $ AC               : Factor w/ 2 levels "N","Y": 2 2 2 2 2 2 1 2 2 2 ...
##  $ ROOMS            : int  6 8 6 6 9 7 5 4 6 6 ...
##  $ BEDRM            : int  3 2 3 2 3 3 2 1 2 2 ...
##  $ AYB              : num  1890 1906 1900 1875 2006 ...
##  $ YR_RMDL          : num  2004 2014 2004 2007 2004 ...
##  $ EYB              : int  1969 1984 1957 1967 2013 2007 1947 1964 1976 1969 ...
##  $ STORIES          : num  1 2 2 2 3 2 1.5 1 2 2 ...
##  $ SALEDATE         : Factor w/ 4938 levels "1992-01-03 00:00:00",..: 2487 3682 787 4155 3472 1800 8
##  $ PRICE            : num  584700 880000 189000 1640000 1356000 ...
##  $ GBA              : int  892 1450 1662 1632 1524 1600 1890 1100 1022 2147 ...
##  $ STYLE            : Factor w/ 14 levels "1 Story","1.5 Story Fin",..: 4 4 4 4 7 4 2 1 4 4 ...
##  $ GRADE            : Factor w/ 12 levels "Above Average",..: 9 1 1 1 12 1 1 2 2 9 ...
##  $ CNDTN            : Factor w/ 6 levels "Average","Excellent",..: 4 4 1 6 6 4 1 1 4 4 ...
##  $ EXTWALL          : Factor w/ 22 levels "Adobe","Aluminum",..: 7 7 5 7 9 7 7 7 7 6 ...
##  $ ROOF             : Factor w/ 14 levels "Built Up","Clay Tile",..: 1 1 1 1 1 1 8 1 1 8 ...
##  $ INTWALL          : Factor w/ 10 levels "Carpet","Ceramic Tile",..: 4 4 4 4 4 4 4 6 4 5 ...
##  $ KITCHENS         : num  1 1 1 1 2 1 1 1 1 1 ...
##  $ FIREPLACES       : int  0 0 1 1 2 1 1 0 0 2 ...
##  $ LANDAREA         : int  771 1331 2190 2200 927 1487 1254 1425 1494 2421 ...
##  $ ZIPCODE          : int  20009 20001 20009 20005 20037 20037 20009 20009 20001 20003 ...
##  $ LATITUDE         : num  38.9 38.9 38.9 38.9 38.9 ...
```

```
##  $ LONGITUDE         : num  -77 -77 -77 -77 -77.1 ...
##  $ ASSESSMENT_NBHD   : Factor w/ 53 levels "16th Street Heights",..: 41 41 41 41 20 20 41 41 41 10 .
##  $ ASSESSMENT_SUBNBHD: Factor w/ 108 levels "001 A American University",..: 82 84 83 83 108 108 82 83
##  $ WARD              : Factor w/ 8 levels "Ward 1","Ward 2",..: 1 1 1 2 2 2 2 2 6 6 ...
##  $ QUADRANT          : Factor w/ 5 levels "NE","NW","SE",..: 2 2 2 2 2 2 2 2 2 3 ...
##  $ Id                : int  NA NA NA NA NA NA NA NA NA NA ...
##  $ Usage             : chr  "Training" "Training" "Training" "Training" ...
```

[4] Sale date convertion and sub-neighborhood removal

```
saledate <- as.Date(dat$SALEDATE)
saledate_dat <- as.numeric(saledate)
dat$SALEDATE <- saledate_dat


dat <- dat[,-27]
```

[5] No more missing values

```
dat <- dat[,-29]


# Proportion of missing values in each variable
pMiss <- function(x){sum(is.na(x))/length(x)}
apply(dat,2,pMiss)
```

```
##          BATHRM       HF_BATHRM            HEAT              AC           ROOMS
##               0               0               0               0               0
##           BEDRM             AYB         YR_RMDL             EYB         STORIES
##               0               0               0               0               0
##        SALEDATE           PRICE             GBA           STYLE           GRADE
##               0               0               0               0               0
```

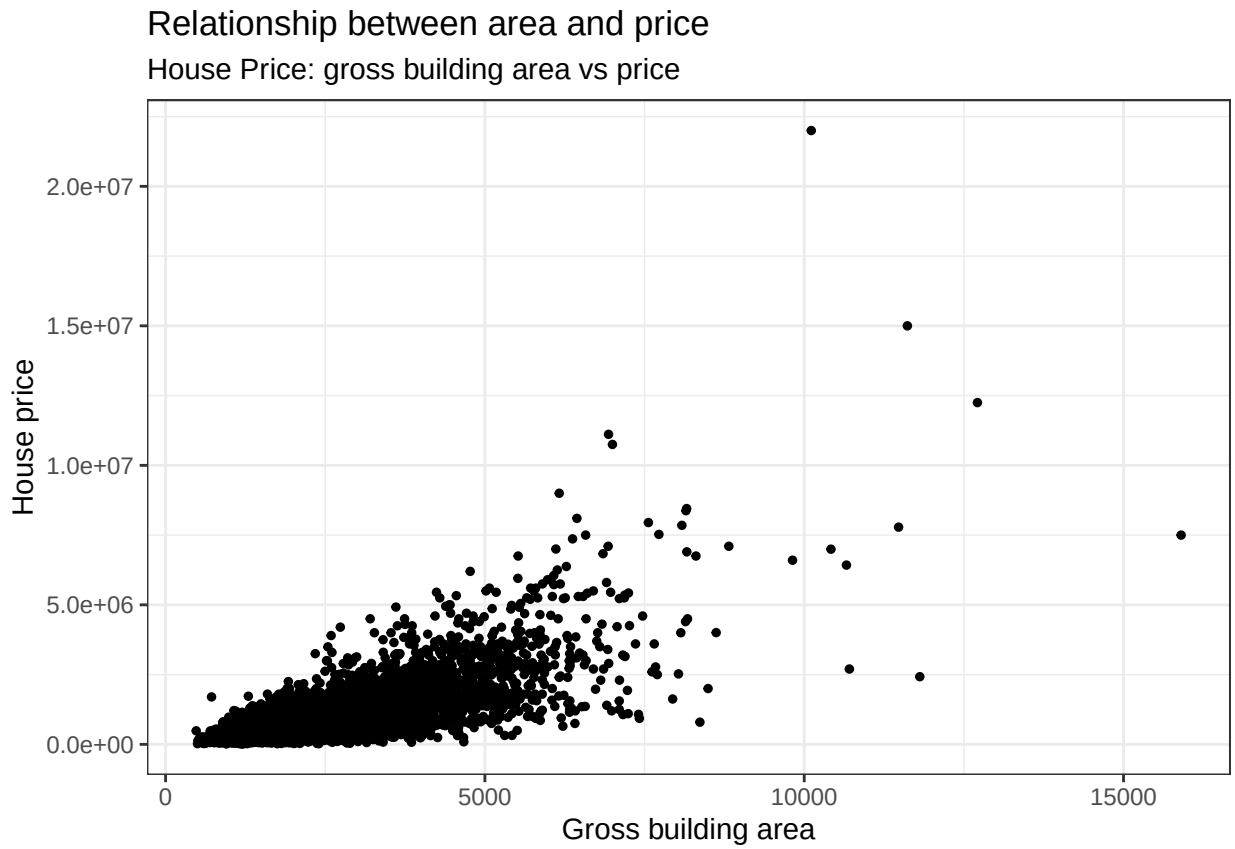
```
##           CNDTN         EXTWALL            ROOF         INTWALL        KITCHENS
##               0               0               0               0               0
##      FIREPLACES        LANDAREA         ZIPCODE        LATITUDE       LONGITUDE
##               0               0               0               0               0
## ASSESSMENT_NBHD            WARD        QUADRANT           Usage
##               0               0               0               0
```

[6] Price vs gross building area

```
library(ggplot2)
```

```
## Warning: package 'ggplot2' was built under R version 3.6.3
```

```
Gross_areas <- dat$GBA

Prices <- dat$PRICE


# Scatterplot

theme_set(theme_bw())  # pre-set the bw theme.

g <- ggplot(dat, aes(Gross_areas, Prices))

g + geom_jitter(width = .5, size=1) +

  labs(subtitle="House Price: gross building area vs price",

       y="House price",

       x="Gross building area",

       title="Relationship between area and price")
```



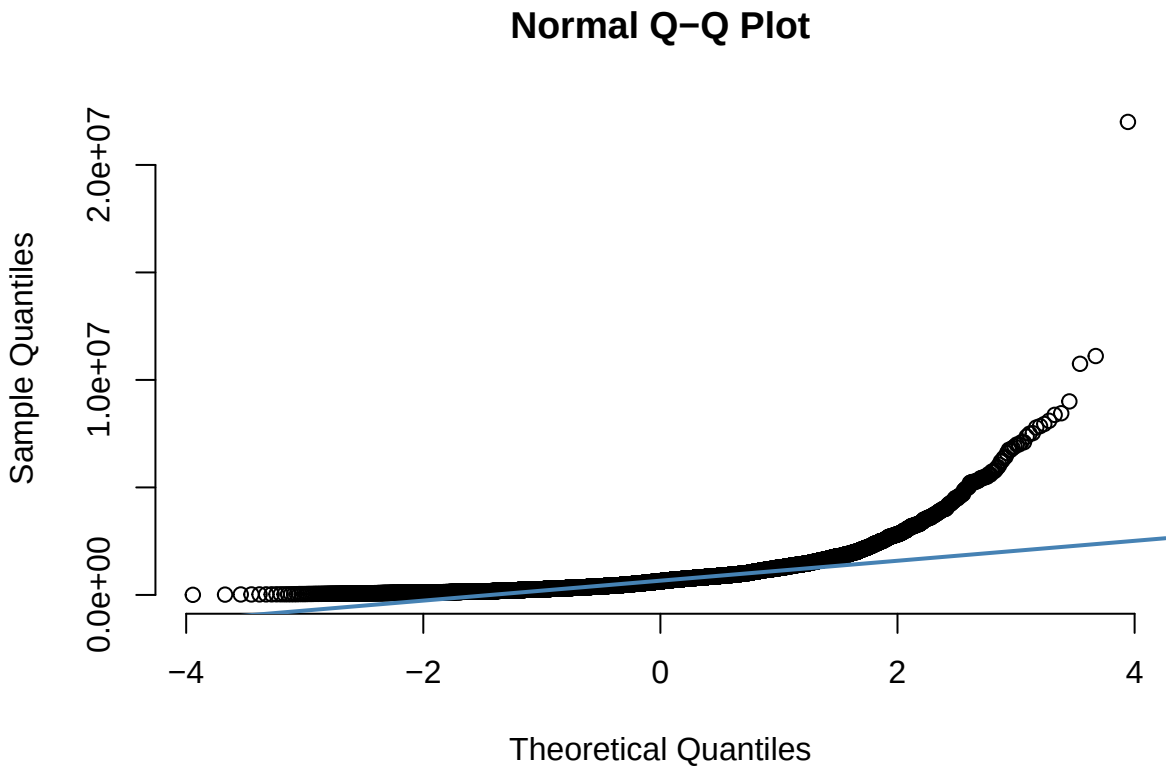
[7] Outlier replacement

```
for(i in 1:12474){
  if(Gross_areas[i]==max(Gross_areas)){
    Gross_areas[i] = mean(Gross_areas)
    Prices[i]=mean(Prices)
  }
}


for(i in 1:12474){
  if(Gross_areas[i]==max(Gross_areas)){
    Gross_areas[i] = mean(Gross_areas)
    Prices[i]=mean(Prices)
  }
```

```
}


dat$PRICE = Prices

dat$GBA = Gross_areas
```

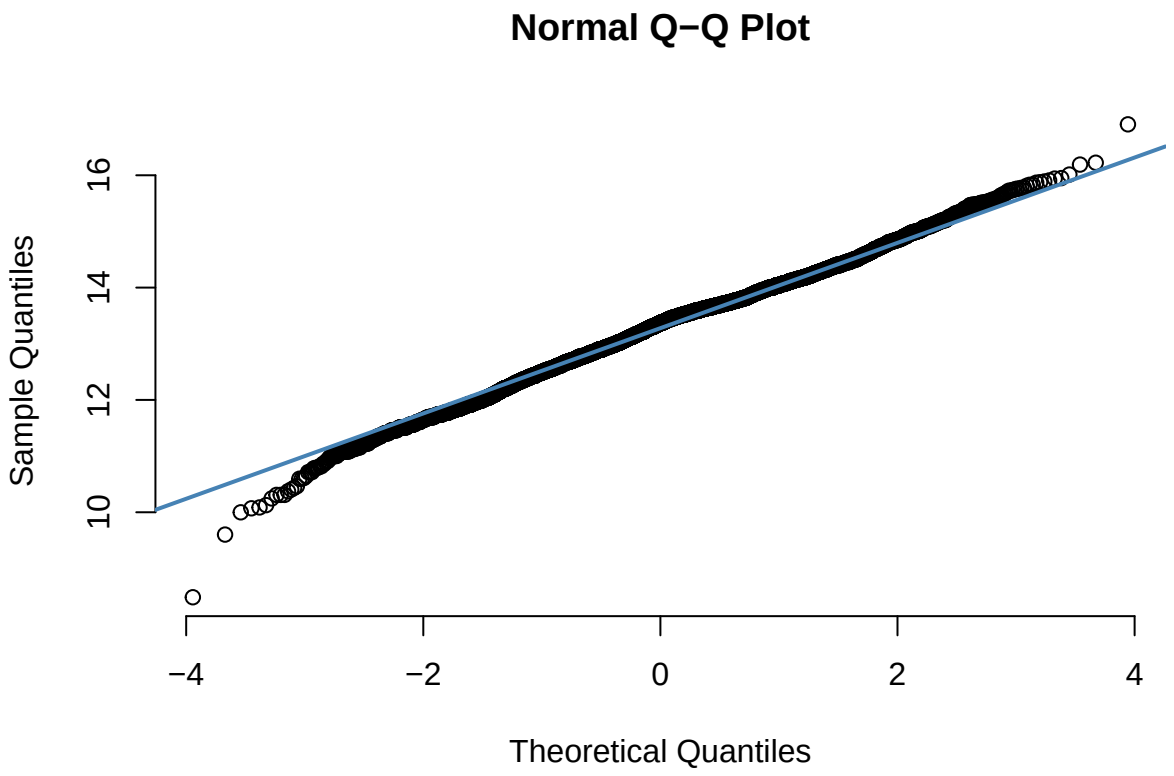
[8] Q-Q plot of price

```
qqnorm(dat$PRICE, pch = 1, frame = FALSE)

qqline(dat$PRICE, col = "steelblue", lwd = 2)
```



[9] Q-Q plot of logarithm price

```
qqnorm(log(dat$PRICE), pch = 1, frame = FALSE)

qqline(log(dat$PRICE), col = "steelblue", lwd = 2)
```



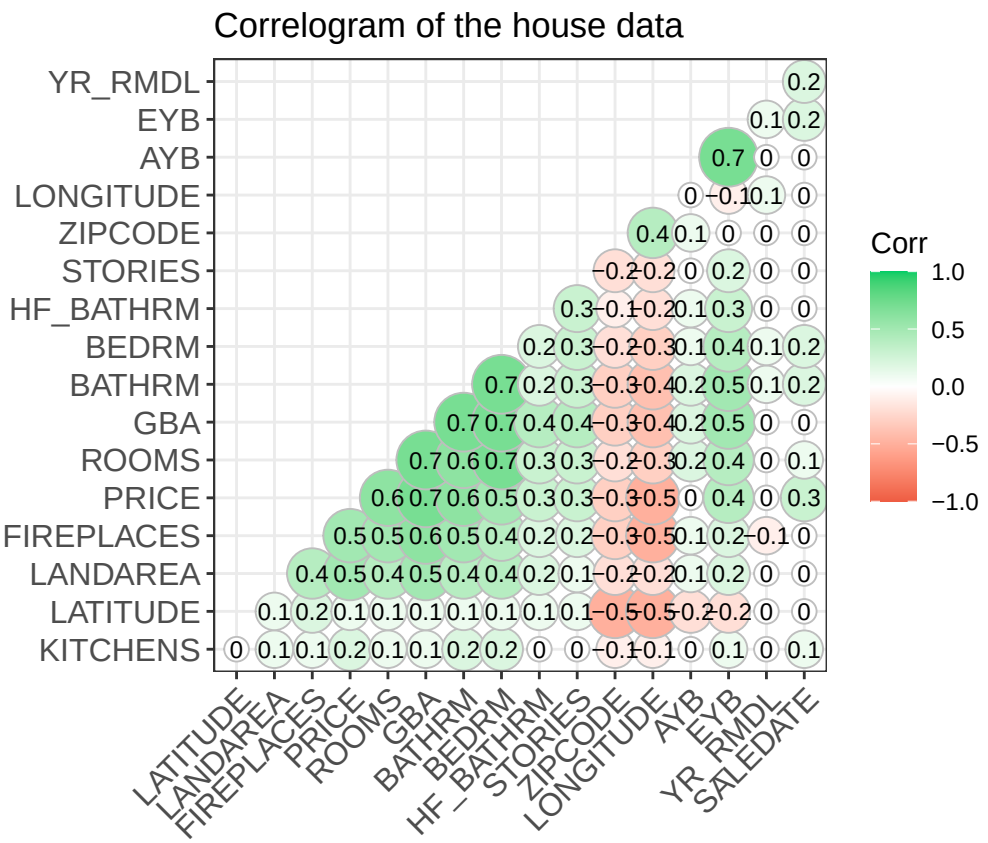
[10] Correlogram of the house data

```
library(ggcorrplot)
```

```
## Warning: package 'ggcorrplot' was built under R version 3.6.3
```

```
# Correlation matrix
numeric_dat <- dat[,c(1,2,5,6,7,8,9,10,11,12,13,20,21,22,23,24,25)]
corr <- round(cor(numeric_dat), 1)


# Plot
ggcorrplot(corr, hc.order = TRUE,
           type = "lower",
           lab = TRUE,
           lab_size = 3,
```

```
            method="circle",
            colors = c("tomato2", "white", "springgreen3"),
            title="Correlogram of the house data",
            ggtheme=theme_bw)
```



[11] A well constructed smoothing model

```
dtrain <- dat[c(1:10002),]
dtrain <- dtrain[,-29]
```

```
library(mgcv)


# Smoothing final model, expected run time is 8 mins.
fit <- bam(log(PRICE)~s(BATHRM,bs="cr")+HF_BATHRM+HEAT+AC+s(ROOMS,bs="cr")+s(BEDRM,bs="cr")
           +s(AYB,bs="cr",by=CNDTN)+s(EYB,bs="cr",by=CNDTN)+s(SALEDATE,bs="cr",by=GRADE)
```

```
            +s(GBA,bs="cr",by=GRADE)+GRADE+CNDTN+ROOF+INTWALL+s(FIREPLACES,bs="cr")
            +s(LANDAREA,bs="cr",by=GRADE)+s(LATITUDE,bs="cr")+s(LONGITUDE,bs="cr")
            +ASSESSMENT_NBHD+WARD+QUADRANT
            +ti(EYB,SALEDATE,k=15,bs="cr")+ti(AYB,LANDAREA,k=13,bs="cr")
            +ti(LATITUDE,LONGITUDE,k=13,bs="cr")+ti(GBA,ROOMS,k=11,bs="cr"), data=dtrain, select=TRUE)
```

[12] A well tuned ranger model

```
library(ranger)


final_model <- ranger(
    formula = log(PRICE) ~ BATHRM+HF_BATHRM+AC+
              ROOMS+BEDRM+AYB+YR_RMDL+
              EYB+STORIES+SALEDATE+
              GBA+GRADE+CNDTN+EXTWALL+
              FIREPLACES+LANDAREA+ZIPCODE+LATITUDE+
              LONGITUDE+ASSESSMENT_NBHD+WARD+QUADRANT,
    data = dtrain,
    num.trees = 520,
    mtry = 9,
    min.node.size = 3,
    sample.fraction = 1,
    splitrule="variance",
    respect.unordered.factors='order',
    regularization.usedepth=TRUE,
    keep.inbag=TRUE,
    x=dtrain[,-12],y=dtrain[,12])
```

[13] A well tuned boosting model

```
library(xgboost)


params <- list(booster = "gbtree", eta=0.03, gamma=0.01, max_depth=8,objective="reg:linear",
               min_child_weight=1, subsample=0.8, colsample_bytree=0.7)


fit <- xgboost(data=dtrain, label=label, params=params,nrounds=710,verbose = 0)


imp_table <- xgb.importance(model = fit)
imp_table
xgb.plot.importance(importance_matrix = imp_table,
                    rel_to_first = TRUE, plot=TRUE)
```

[14] Variable importance list

```
library(ranger)
```

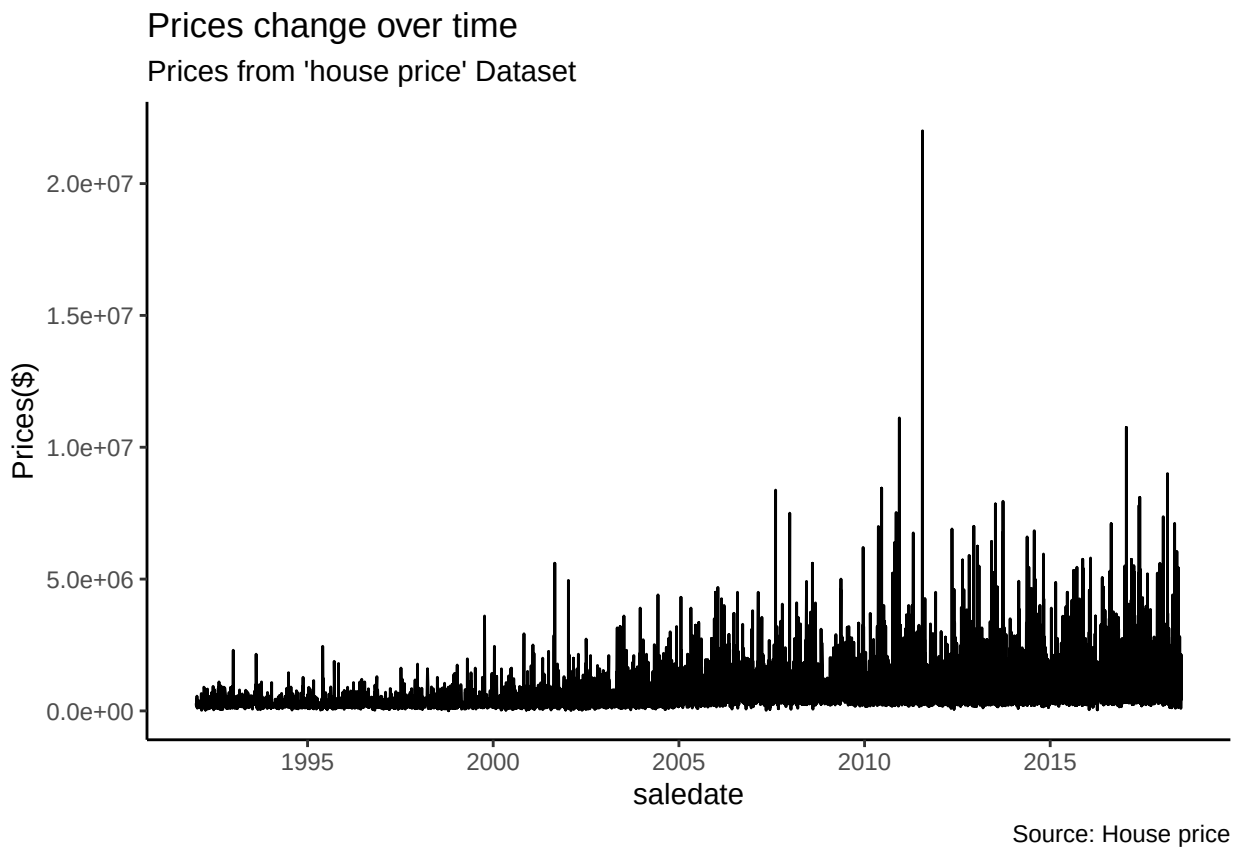
```
## Warning: package 'ranger' was built under R version 3.6.3
```

```
# The Full Model - 27 explanatory
full_fit <- ranger(log(PRICE)~BATHRM+HF_BATHRM+HEAT+AC+
             ROOMS+BEDRM+AYB+YR_RMDL+
             EYB+STORIES+SALEDATE+
             GBA+STYLE+GRADE+CNDTN+EXTWALL+
           ROOF+KITCHENS+INTWALL+
             FIREPLACES+LANDAREA+ZIPCODE+LATITUDE+
             LONGITUDE+ASSESSMENT_NBHD+WARD+QUADRANT, data=dtrain, num.trees=500,
             mtry=10,importance="impurity")


imp <- importance(full_fit)
imp
```

```
##          BATHRM       HF_BATHRM            HEAT              AC           ROOMS
##      363.449815       12.715605        8.408030       10.907672       58.070822
##           BEDRM             AYB         YR_RMDL             EYB         STORIES
##       90.797723       56.677099       59.574189      328.987292       12.689047
##        SALEDATE             GBA           STYLE           GRADE           CNDTN
##     1657.156753      543.630935        9.935470      710.858242       69.383874
##         EXTWALL            ROOF        KITCHENS         INTWALL      FIREPLACES
##       19.924213        8.928449        1.616037        7.304181       26.619379
##        LANDAREA         ZIPCODE        LATITUDE       LONGITUDE ASSESSMENT_NBHD
##       73.774058      352.281042      184.127471     1270.032016       38.357140
##            WARD        QUADRANT
##      434.311843        9.621198
```

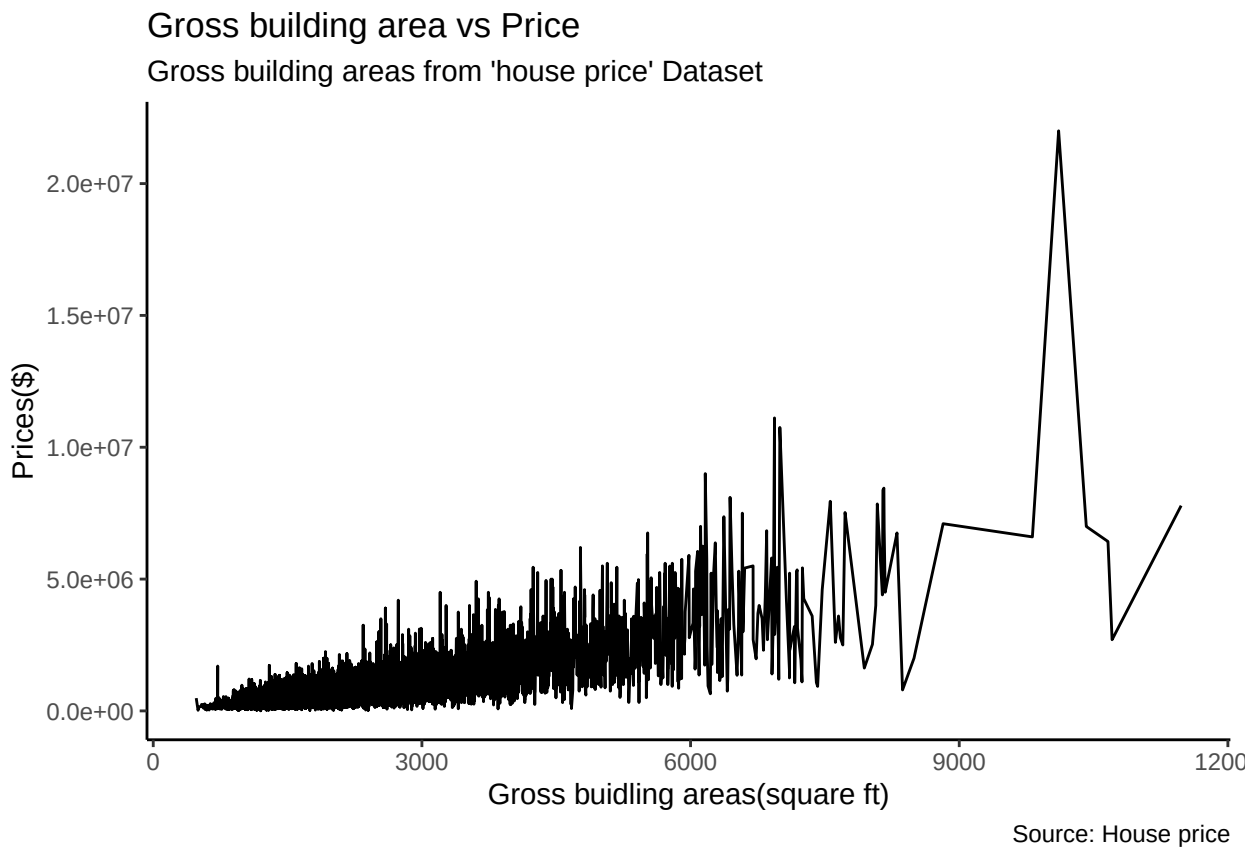
[15] Price vs sale date

```
theme_set(theme_classic())


# Allow Default X Axis Labels
ggplot(dat, aes(x=saledate)) +
  geom_line(aes(y=Prices)) +
  labs(title="Prices change over time",
       subtitle="Prices from 'house price' Dataset",
       caption="Source: House price",
       y="Prices($)")
```



[16] Gross building area vs price

```
theme_set(theme_classic())


# Allow Default X Axis Labels
ggplot(dat, aes(x=Gross_areas)) +
  geom_line(aes(y=Prices)) +
  labs(title="Gross building area vs Price",
       subtitle="Gross building areas from 'house price' Dataset",
       caption="Source: House price",
       x="Gross buidling areas(square ft)",
       y="Prices($)")
```



[17] The most expensive mansion

```
num = 0


for(i in 1:12474){
  if(Prices[i]==max(Prices)){
    num = i
  }
}


t(dat[num,])
```

```
##                  90
## BATHRM          "6"
```

```
## HF_BATHRM       "5"
## HEAT            "Warm Cool"
## AC              "Y"
## ROOMS           "17"
## BEDRM           "9"
## AYB             "1801"
## YR_RMDL         "2013"
## EYB             "1986"
## STORIES         "2.5"
## SALEDATE        "15180"
## PRICE           "2.2e+07"
## GBA             "10110"
## STYLE           "2.5 Story Fin"
## GRADE           "Exceptional-C"
## CNDTN           "Good"
## EXTWALL         "Common Brick"
## ROOF            "Slate"
## INTWALL         "Hardwood"
## KITCHENS        "2"
## FIREPLACES      "4"
## LANDAREA        "155905"
## ZIPCODE         "20007"
## LATITUDE        "38.91167"
## LONGITUDE       "-77.05661"
## ASSESSMENT_NBHD "Georgetown"
## WARD            "Ward 2"
## QUADRANT        "NW"
## Usage           "Training"
```

```
saledate[num]
```

```
## [1] "2011-07-25"
```

[18] Grouped highly graded houses

```
grades <- dat$GRADE

num_vector <- c()


for(i in 1:12474){

  if((grades[i]=="Excellent")|(grades[i]=="Superior")){

    num_vector <- c(num_vector,i)

  }

}


deals <- dat[num_vector,]

head(deals)
```

```
##    BATHRM HF_BATHRM          HEAT AC ROOMS BEDRM  AYB  YR_RMDL  EYB STORIES
## 58      2         1     Warm Cool  Y     9     3 1900 2004.387 1969     2.0
## 59      2         1    Forced Air  Y     5     2 1864 1990.000 1979     2.0
## 61      2         1 Hot Water Rad  Y     6     3 1902 2007.000 1992     2.0
## 62      4         1     Warm Cool  Y     8     4 1958 2010.000 1990     3.0
## 64      2         2     Warm Cool  Y     9     3 1900 2009.000 1983     3.0
## 67      3         0     Warm Cool  Y     7     3 1900 2002.000 1988     2.5
##    SALEDATE   PRICE  GBA         STYLE     GRADE     CNDTN      EXTWALL
## 58     8195  436000 1630       2 Story  Superior   Average       Stucco
## 59    11054  565000 1088       2 Story  Superior   Average Common Brick
## 61    16624 1375000 1568       2 Story  Superior Very Good Common Brick
## 62    15784 2006000 3322       3 Story  Superior      Good Common Brick
```

```
## 64    16286 1675000 2510       3 Story  Superior      Good Common Brick
## 67    13724 1800000 1596 2.5 Story Fin Excellent Very Good Brick/Siding
##             ROOF  INTWALL KITCHENS FIREPLACES LANDAREA ZIPCODE LATITUDE
## 58 Comp Shingle Hardwood        1          3     2961   20007 38.90701
## 59   Metal- Sms Hardwood        1          1     2745   20007 38.90758
## 61   Metal- Sms Hardwood        1          2     1232   20007 38.90895
## 62        Slate Hardwood        1          4     1782   20007 38.90940
## 64        Slate Hardwood        1          2     1385   20007 38.90963
## 67   Metal- Sms Hardwood        2          1     3600   20007 38.90840
##    LONGITUDE ASSESSMENT_NBHD   WARD QUADRANT    Usage
## 58 -77.05597      Georgetown Ward 2       NW Training
## 59 -77.05552      Georgetown Ward 2       NW Training
## 61 -77.06641      Georgetown Ward 2       NW Training
## 62 -77.06097      Georgetown Ward 2       NW Training
## 64 -77.06768      Georgetown Ward 2       NW Training
## 67 -77.07018      Georgetown Ward 2       NW Training
```

[19] Common features of highly graded houses

```
features <- c()


getmode <- function(v) {
   uniqv <- unique(v)
   uniqv[which.max(tabulate(match(v, uniqv)))]
}


for(i in 1:28){
  features <- c(features, getmode(deals[,i]))
}
```

```
for(i in 1:length(features)){
  if(is.factor(dat[,i])){
    level_num <- levels(dat[,i])
    feature <- level_num[features[i]]
    print(feature)
  }
}
```

```
## [1] "Warm Cool"
## [1] "Y"
## [1] "2 Story"
## [1] "Excellent"
## [1] "Good"
## [1] "Common Brick"
## [1] "Slate"
## [1] "Hardwood"
## [1] "Berkley"
## [1] "Ward 3"
## [1] "NW"
```

[20] Hybrid selection on variables

```
# Preliminary stuff that will be required by stepwise selection technique:
n <- dim(dtrain)[1]
sml <- bam(PRICE ~ 1, data = dtrain)
lrg <- bam(PRICE ~ ., data = dtrain)

# Hybrid Selection (backward + forward)
m_h <- stepAIC(object = sml, scope = list(upper = lrg, lower = sml),
```

```
direction = "both", trace = 0, k = log(n))

variable.names(m_h)[2:length(variable.names(m_h))]
```

[21] Compare smoothing models by ANOVA and AIC

```
# The Full Model - 27 explanatory
fit_full <- bam(log(PRICE)~BATHRM+HF_BATHRM+HEAT+AC+ROOMS+BEDRM+AYB+EYB+
                SALEDATE+GBA+GRADE+CNDTN+ROOF+INTWALL+FIREPLACES+
                LANDAREA+LATITUDE+LONGITUDE+ASSESSMENT_NBHD+WARD+QUADRANT, data=dtrain, select=TRUE)


# The Selected Model - 21 explanatory
fit_select <- bam(log(PRICE)~BATHRM+HF_BATHRM+HEAT+AC+ROOMS+BEDRM+AYB+EYB+
                SALEDATE+GBA+GRADE+CNDTN+ROOF+INTWALL+FIREPLACES+
                LANDAREA+LATITUDE+LONGITUDE+ASSESSMENT_NBHD+WARD+QUADRANT, data=dtrain, select=TRUE)


AIC(fit_full, fit_select)
anova(fit_full)
anova(fit_select)
gam.check(fit_select)
```

[22] Hypergrid parameter tuning

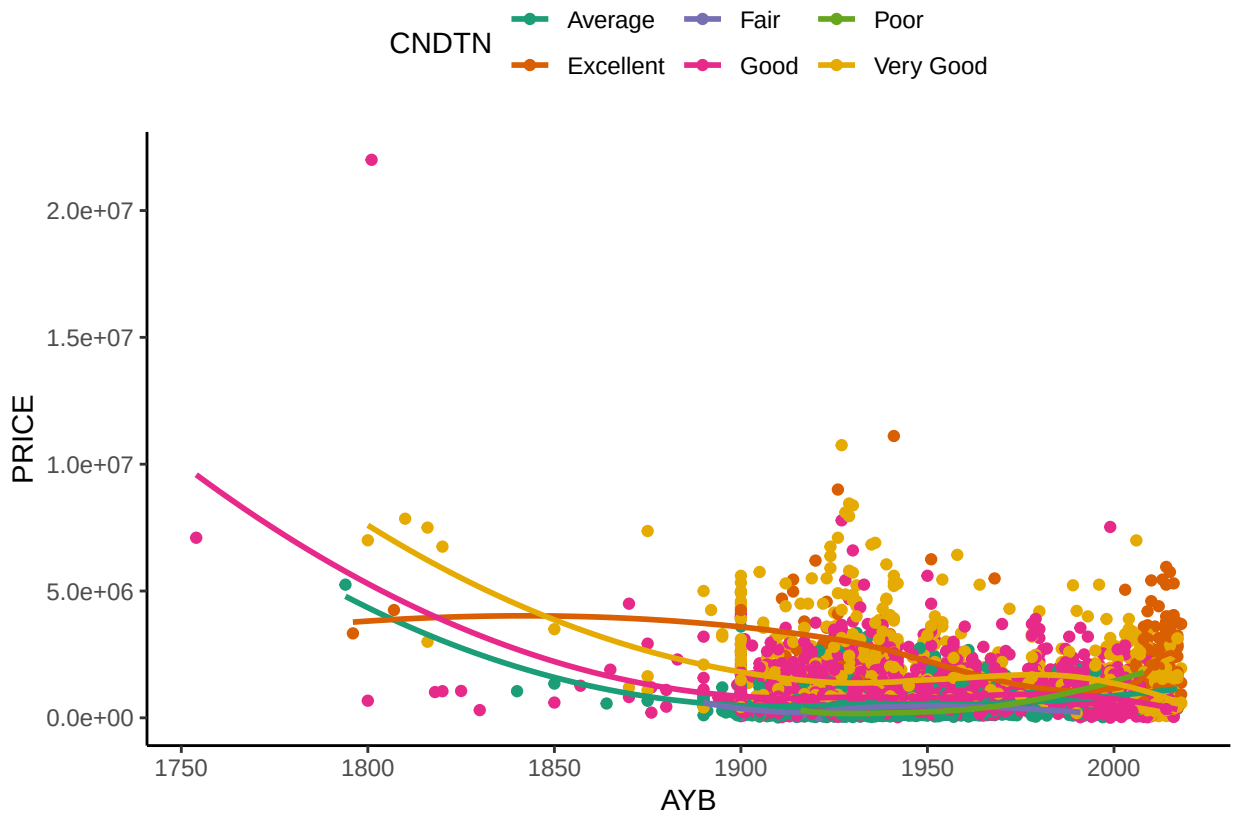
```
# hyperparameter grid search, with starting point
# mtry=9, node_size=4, num_tree=500, sample_size=1
# this has a public score of 0.1920
hyper_grid <- expand.grid(
  mtry = seq(8, 10, by = 1),
  node_size  = seq(3, 5, by = 1),
  num_tree = seq(490,510,by=10),
  sample_size = c(.80, 1),
```

```
  OOB_RMSE  = 0
)

for (i in 1:nrow(hyper_grid)) {
  # train model
  model <- ranger(
    formula = log(PRICE) ~ BATHRM+HF_BATHRM+AC+
              ROOMS+BEDRM+AYB+YR_RMDL+
              EYB+STORIES+SALEDATE+
              GBA+GRADE+CNDTN+EXTWALL+
              FIREPLACES+LANDAREA+ZIPCODE+LATITUDE+
              LONGITUDE+ASSESSMENT_NBHD+WARD+QUADRANT,
    data = dtrain,
    num.trees = hyper_grid$num_tree[i],
    mtry = hyper_grid$mtry[i],
    min.node.size = hyper_grid$node_size[i],
    sample.fraction = hyper_grid$sample_size[i],
    seed = set.seed(20660938))

  hyper_grid$OOB_RMSE[i] <- sqrt(model$prediction.error)
}

library(dplyr)

hyper_grid %>%
  dplyr::arrange(OOB_RMSE) %>%
  head(10)
```

[23] Interactions between factors and continuous variables

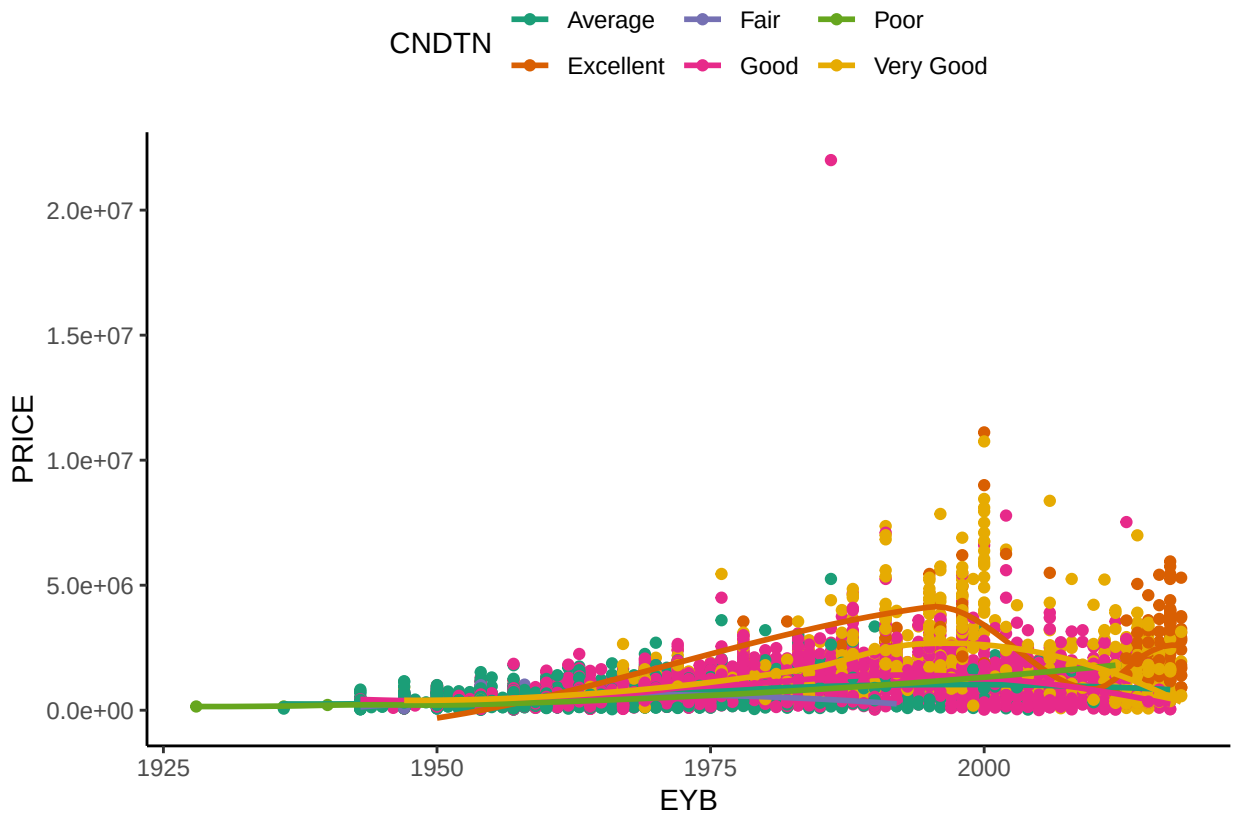
```
ggplot(dat,aes(x = AYB, y = PRICE, colour = CNDTN))+geom_point()+geom_smooth(method ='loess', se = FALSE
  scale_colour_brewer(type ='qual', palette ='Dark2')+theme(legend.position ='top')
```

```
## `geom_smooth()` using formula 'y ~ x'
```



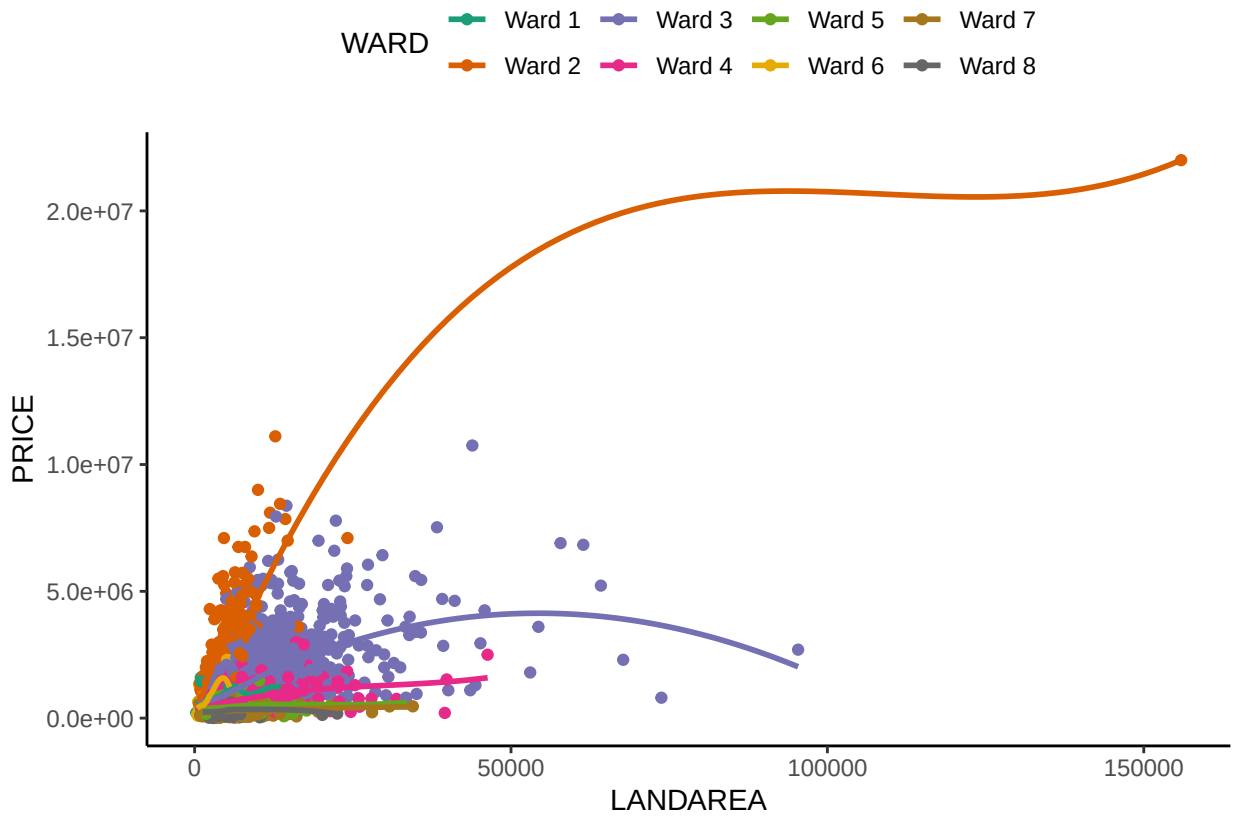

```
ggplot(dat,aes(x = EYB, y = PRICE, colour = CNDTN))+geom_point()+geom_smooth(method ='loess', se = FALSE
  scale_colour_brewer(type ='qual', palette ='Dark2')+theme(legend.position ='top')
```

```
## `geom_smooth()` using formula 'y ~ x'
```




```
ggplot(dat,aes(x = LANDAREA, y = PRICE, colour = WARD))+geom_point()+geom_smooth(method ='loess', se = F
  scale_colour_brewer(type ='qual', palette ='Dark2')+theme(legend.position ='top')
```

```
## `geom_smooth()` using formula 'y ~ x'
```


WARD
Ward 1
Ward 2
Ward 3
Ward 4
Ward 5
Ward 6
Ward 7
Ward 8
PRICE
2.0e+07
1.5e+07
1.0e+07
5.0e+06
0.0e+00
0
50000
100000
150000
LANDAREA