STA 521 Final Project Writeup

BayeStar

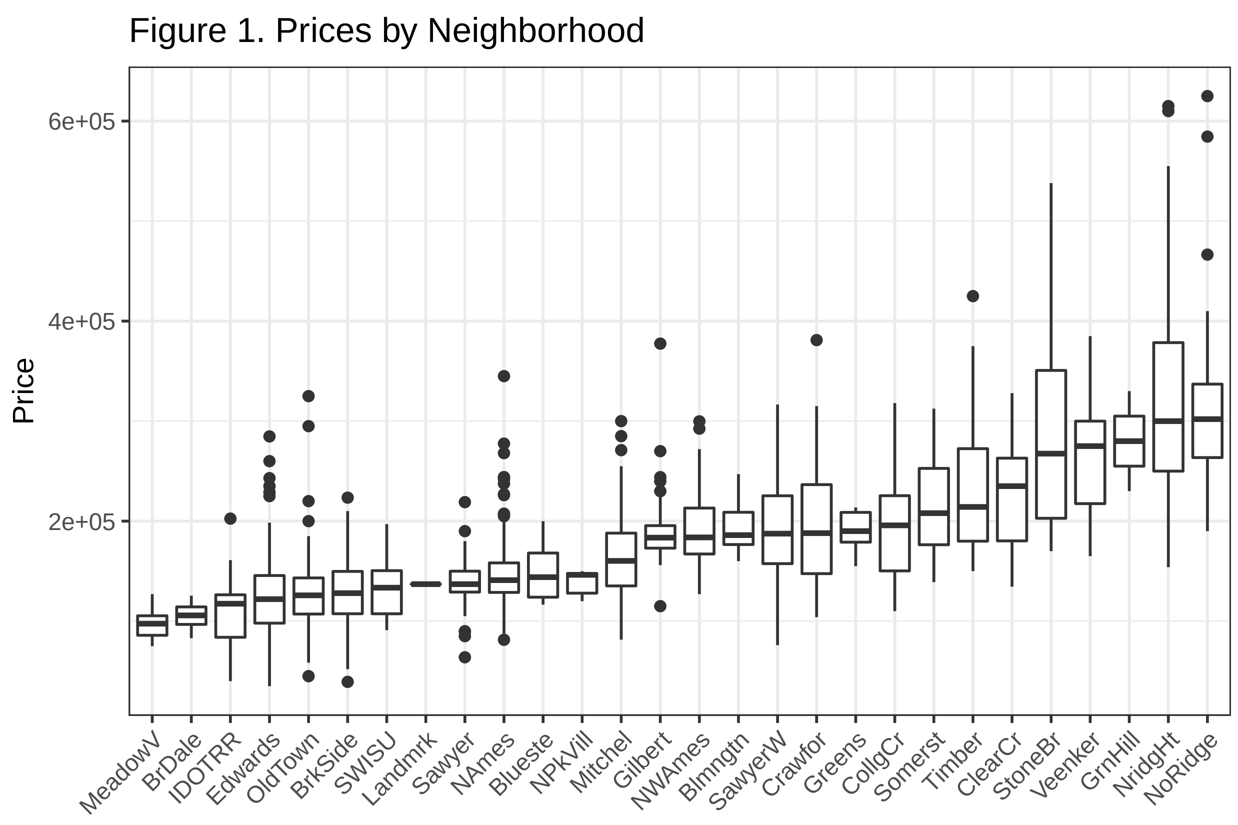
April 26, 2017

Exploratory data analysis

```r
load("ames_train.Rdata")
load("ames_test.Rdata")
```

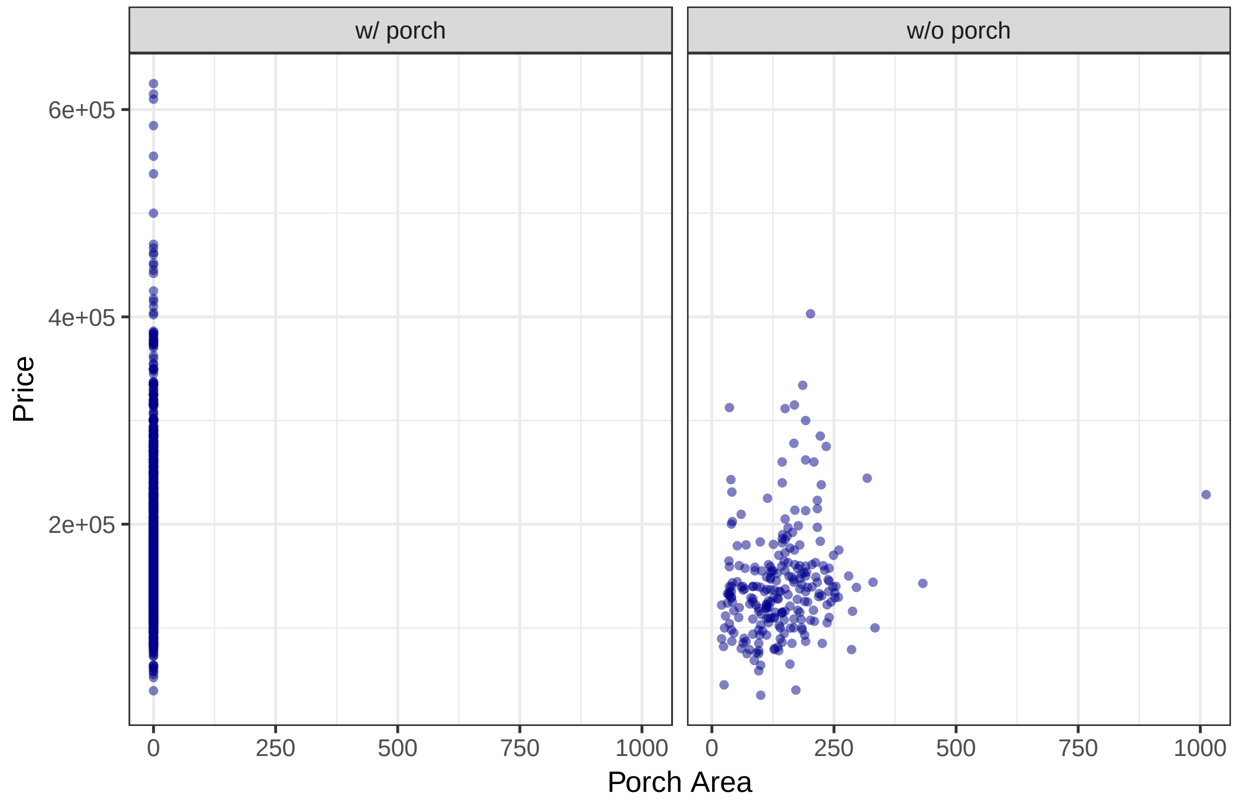
- Figure 1 shows the boxplot of prices by each neighborhood. It is apparent that prices vary a lot by different neighborhoods. Outliers are also captured and we considered delete some of them.
- Figure 2 is a scatterplot of prices against porch area. The left panel corresponds to houses without porch, while the right with porch. We can see that the relationship differs between with and without porch, so we decided to add a dummy variable indicating the existence of porch and an interaction between the dummy and porch area.
- Figure 3 shows the prices of houses built in various years. There is a non-linear trend, so we considered adding quadratic term of year to capture the non-linearity.
- Figure 4 depicts the relationship between prices and total square by neighborhood. (Only four of neighborhoods are shown for plot purpose.) Prices and total square are linearly related, but the linear relationship changes across neigborhood. We considered adding interaction term of total square and neighborhood to capture different slopes.
- Figure 5 is a vilion plot showing the distribution of prices by overall quality. Prices are higher, but also vary more for higher quality.

```r
# price v.s. neighborhood
ggplot(ames_train, aes(x=reorder(Neighborhood, price, FUN=median), y=price))+
  theme_bw()+
  theme(axis.text.x=element_text(angle=45, hjust=1))+
  geom_boxplot()+
  xlab('')+
  ylab('Price')+
  ggtitle("Figure 1. Prices by Neighborhood")
```

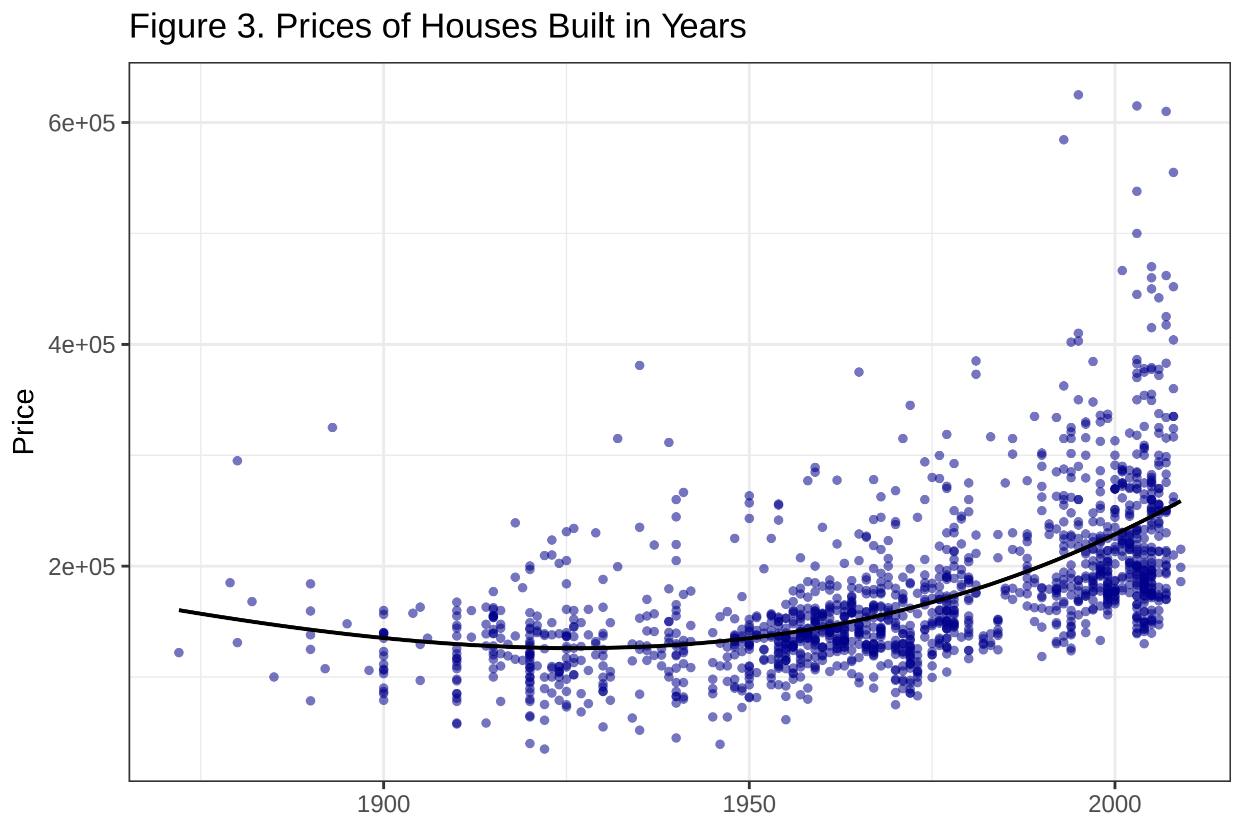


```r
# price v.s. Enclosed.Porch
ames_train %>%
  mutate(Porch = ifelse(Enclosed.Porch==0, "w/ porch", "w/o porch")) %>%
  ggplot(aes(x=Enclosed.Porch, y=price))+
  geom_point(col="dark blue", cex=1, alpha=0.5)+
  facet_wrap(~Porch)+
  xlab("Porch Area")+
  ylab("Price")+
  theme_bw()+
  ggtitle("Figure 2. Prices against Porch Area")
```

Figure 2. Prices against Porch Area

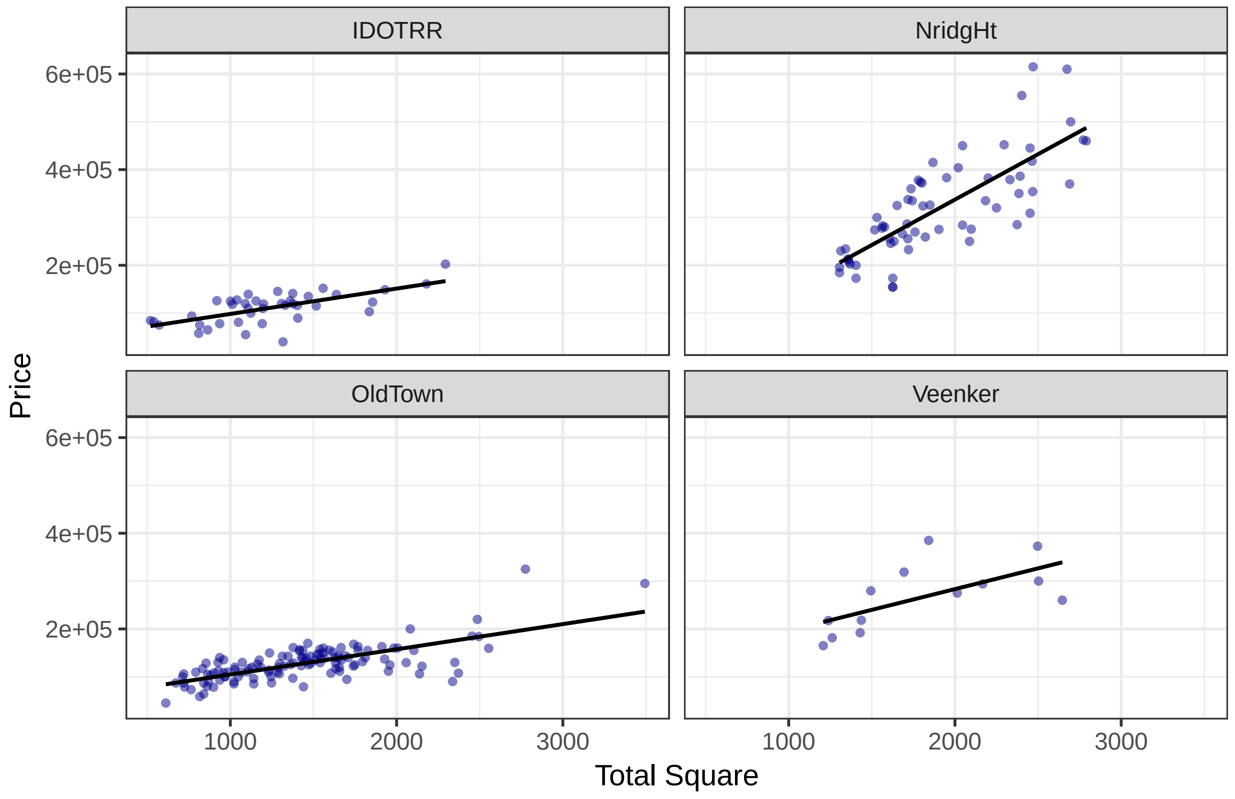


```
# year
ggplot(ames_train, aes(x=Year.Built, y=price,alpha = 0.5))+
  geom_point(col="dark blue", cex=1)+
  geom_smooth(method = "lm", formula = y ~ x + I(x^2)+I(x^3),
              col="black", se=F, size=0.7)+
  theme_bw()+
  theme(legend.position="none")+
  ylab('Price')+
  xlab('')+
  ggtitle("Figure 3. Prices of Houses Built in Years")
```



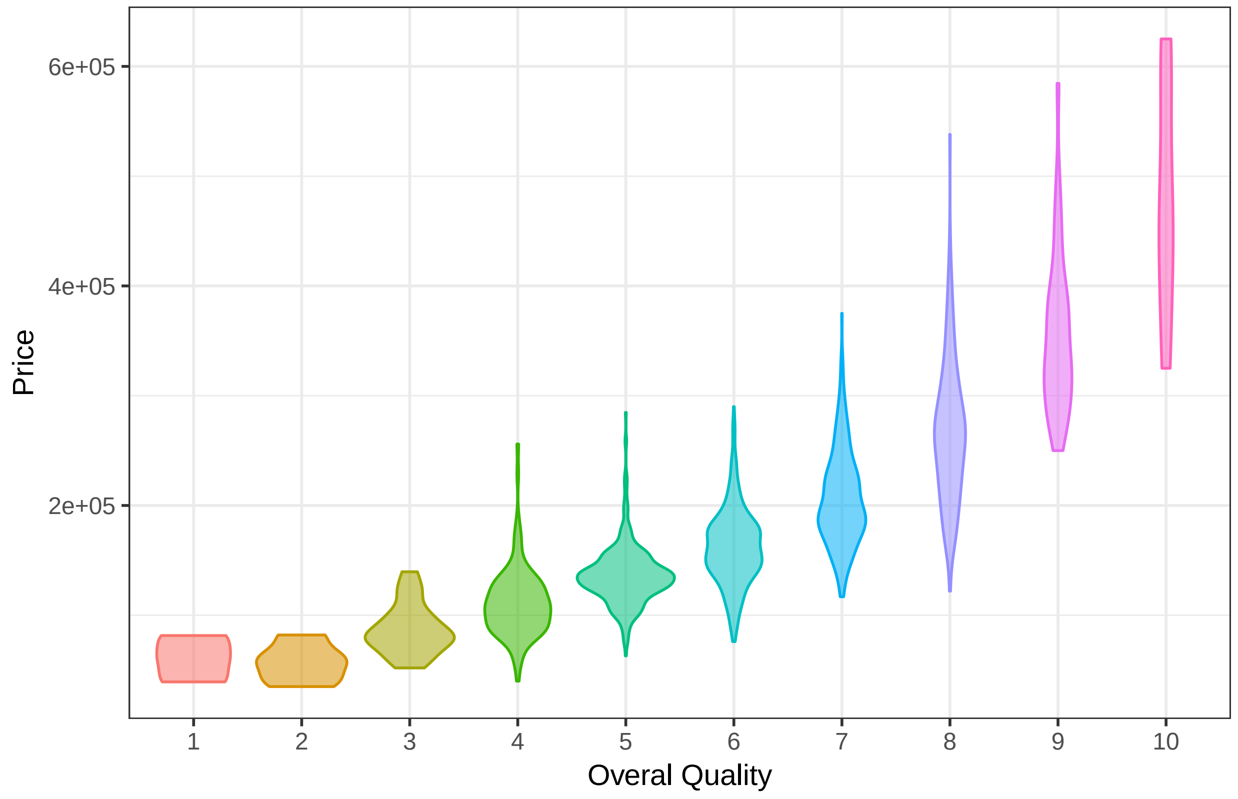
```r
# Neighborhood and TotalSq
ames_train %>%
  filter(Neighborhood=="OldTown" |Neighborhood=="NridgHt"|
           Neighborhood=="IDOTRR"|Neighborhood=="Veenker") %>%
  ggplot(aes(x=TotalSq, y=price))+
  geom_point(col="dark blue", cex=1, alpha=0.5)+
  facet_wrap(~Neighborhood)+
  geom_smooth(method="lm", se=F, col="black", size=0.7)+
  theme_bw()+
  xlab("Total Square")+
  ylab("Price")+
  ggtitle("Figure 4. Prices by Total Square in Different Neighborhoods")
```

Figure 4. Prices by Total Square in Different Neighborhoods



```
# Overall.Qual
ggplot(ames_train, aes(x=factor(Overall.Qual), y=price))+
  geom_violin(aes(color=factor(Overall.Qual),
                  fill = factor(Overall.Qual), alpha=0.5))+
  theme_bw() +
  theme(legend.position="none")+
  xlab("Overal Quality")+
  ylab("Price")+
  ggtitle("Figure 5. Prices by Overall Quality")
```

Figure 5. Prices by Overall Quality



Simple Model

Data Cleaning

```
# clean data
clean_data = function(xdata){
xdata %>%
    mutate(# replace NAs with new levels
           Alley = as.factor(ifelse(is.na(as.character(Alley)),
                                "No alley access", as.character(Alley))),
           Bsmt.Qual = as.factor(ifelse(as.character(Bsmt.Qual)=="Po",
                                    "Fa", as.character(Bsmt.Qual))),
           Bsmt.Qual = as.factor(ifelse(is.na(as.character(Bsmt.Qual)),
                                    "No Basement", as.character(Bsmt.Qual))),
           Bsmt.Cond = as.factor(ifelse(is.na(as.character(Bsmt.Cond)),
                                    "No Basement", as.character(Bsmt.Cond))),
           BsmtFin.Type.1 = as.factor(ifelse(is.na(as.character(BsmtFin.Type.1)),
                                    "No Basement", as.character(BsmtFin.Type.1))),
           BsmtFin.Type.2 = as.factor(ifelse(is.na(as.character(BsmtFin.Type.2)),
                                    "No Basement", as.character(BsmtFin.Type.2))),
           Bsmt.Exposure = as.factor(ifelse(is.na(as.character(Bsmt.Exposure))|
                                       as.character(Bsmt.Exposure) == "",
                                    "No Basement", as.character(Bsmt.Exposure))),
           Bsmt.Unf.Rate.SF = ifelse(Total.Bsmt.SF!=0, Bsmt.Unf.SF/Total.Bsmt.SF, 0),
           Bsmt.Full.Bath = ifelse(is.na(Bsmt.Full.Bath),0,Bsmt.Full.Bath),
           Bsmt.Half.Bath = ifelse(is.na(Bsmt.Half.Bath),0,Bsmt.Half.Bath),
           Fireplace.Qu = as.factor(ifelse(is.na(as.character(Fireplace.Qu)),
                                           "No Fireplace", as.character(Fireplace.Qu))),
           Garage.Type = as.factor(ifelse(is.na(as.character(Garage.Type)),
                                           "No Garage", as.character(Garage.Type))),
           Garage.Finish = as.factor(ifelse(is.na(as.character(Garage.Finish))|
                                              as.character(Garage.Finish) == "",
                                           "No Garage", as.character(Garage.Finish))),
           Garage.Qual = as.factor(ifelse(as.character(Garage.Qual)=="Ex",
                                          "Gd", as.character(Garage.Qual))),
           Garage.Qual = as.factor(ifelse(is.na(as.character(Garage.Qual)),
                                           "No Garage", as.character(Garage.Qual))),
           Garage.Cond = as.factor(ifelse(as.character(Garage.Cond)=="Ex",
                                          "Gd", as.character(Garage.Cond))),
           Garage.Cond = as.factor(ifelse(is.na(as.character(Garage.Cond))|
                                            as.character(Garage.Cond)=="Po",
                                           "No Garage", as.character(Garage.Cond))),
           # deal with new level issue in test data
           Fence = as.factor(ifelse(is.na(as.character(Fence)),
                                           "No Fence", as.character(Fence))),
           Misc.Feature = as.factor(ifelse(is.na(as.character(Misc.Feature)),
                                           "None", as.character(Misc.Feature))),
           Mas.Vnr.Type = as.factor(ifelse(as.character(Mas.Vnr.Type) == "",
                                           "None", as.character(Mas.Vnr.Type))),
           Mas.Vnr.Area = ifelse(is.na(Mas.Vnr.Area),0,Mas.Vnr.Area),
           Kitchen.Qual = as.factor(ifelse(as.character(Kitchen.Qual)=="Po",
                                           "Fa", as.character(Kitchen.Qual))),
           Heating.QC = as.factor(ifelse(as.character(Heating.QC)=="Po",
                                         "Fa", as.character(Heating.QC))),
           Electrical = as.factor(ifelse(as.character(Electrical) == "",
                                         "SBrkr", as.character(Electrical))),
           Condition.2 = as.factor(ifelse(as.character(Condition.2) %in%
                                            c("Artery","RRAn","RRAe"),
                                          "Feedr", as.character(Condition.2))),
           Neighborhood = as.factor(ifelse(as.character(Neighborhood)=="Blueste",
                                          "NPkVill", as.character(Neighborhood))),
           # create new variables
           Enclosed.Porch.is = as.factor(ifelse(Enclosed.Porch==0,"N","Y")),
           Pool.Area = as.factor(ifelse(Pool.Area==0,"N", "Y")),
           Garage.Yr.Blt = ifelse(is.na(Garage.Yr.Blt), Year.Built-2, Garage.Yr.Blt)
          )%>%
    dplyr::select(-c(Lot.Frontage,Pool.QC,Pool.Area))
}

# remove outliers
ames_train = clean_data(ames_train)
ames_train = ames_train[-c(462,168,183),]
ames_train = ames_train[ames_train$price<500000,-1]
remove_idx1 = c(1:nrow(ames_train))[ames_train$Neighborhood %in%
                                     c("Gilbert")&ames_train$price>350000]
remove_idx2 = c(1:nrow(ames_train))[ames_train$Neighborhood %in%
                                     c("NAmes")&ames_train$price>300000]
remove_idx3 = c(1:nrow(ames_train))[ames_train$Neighborhood %in%
```

```
                                                    c("Landmrk","GrnHill")]
ames_train = ames_train[-c(remove_idx1, remove_idx2, remove_idx3),]
```

Initial Model

We first putted in a few variables that are intuitively important as our base model and kept all the variables put in significant. The variables we chose include `area`, `Year.Built`, `Garage.Area`, `Overall.Qual`, `Kitchen.Qual`, `Garage.Cond`, `Neighborhood`, etc. Based on this base model, forward selection was used to select more variables that can be included in our model and improve the prediction accuracy. In this process we try to avoid the variables might be correlated with other variables, such as TotalSq an area. With the help of the diagonostic plots, we also removed some outliers, and finally end up with a model with 20 variables.

Based on the summary table of the selected model, all the selected continuous variables are extremely significant. These continuous variables include `area`, `Year.Built`, `Year.Remod.Add`, `Garage.Area`, `Overall.Qual`, `Lot.Area`, `BsmtFin.SF.1`, `Overall.Cond`, `Total.Bsmt.SF`, `Bsmt.Full.Bath` and `Screen.Porch`. Additionally, some categorical variables having a lot of significant levels are selected, such as `Neighborhood`, `Kitchen.Qual`, `Exter.Qual`.

Explanation of coeficients:

a. Area is the most significant continuous variable in the model with a coefficient of 1.373e-03, which indicates when the other conditions stay the same, one unit increase in area leads to exp(2.707e-04)=1.0002 times of original price. The reason for having such a small increasing ratio is that the 1.0002 times a price with a large magnitude can still lead to a big increment.

b. Kichen.Qual if the most significant categorical variable. The base case is level "Ex". Level "Gd" has a coefficient of -5.298e-02, which means properties with a good quality kitchen have prices exp(-5.298e-02)=0.948 times of the prices of properties with an excellent quality kitchen. Level "TA" and "Fa" have lower ratios 0.9296 and 0.9218 respectively, which indicates properties with a better kitchen will have a higher price.

```
model1 = lm(log(price) ~ area + Year.Built + Year.Remod.Add +
              Garage.Area + Overall.Qual +  log(Lot.Area) +
              BsmtFin.SF.1 + Overall.Cond + Total.Bsmt.SF +
              Central.Air + Bsmt.Full.Bath + Screen.Porch +
              Kitchen.Qual + Exter.Qual + Bldg.Type + Bsmt.Qual +
              Garage.Cond + Neighborhood + Heating.QC, data=ames_train)
summary(model1)
```

```
##
## Call:
## lm(formula = log(price) ~ area + Year.Built + Year.Remod.Add +
##     Garage.Area + Overall.Qual + log(Lot.Area) + BsmtFin.SF.1 +
##     Overall.Cond + Total.Bsmt.SF + Central.Air + Bsmt.Full.Bath +
##     Screen.Porch + Kitchen.Qual + Exter.Qual + Bldg.Type + Bsmt.Qual +
##     Garage.Cond + Neighborhood + Heating.QC, data = ames_train)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.43704 -0.04857 -0.00061  0.05308  0.29056
##
## Coefficients:
```

```
##                         Estimate Std. Error t value Pr(>|t|)    
## (Intercept)            3.274e+00  5.310e-01   6.166 9.11e-10 ***
## area                   2.707e-04  7.276e-06  37.200  < 2e-16 ***
## Year.Built             2.933e-03  2.116e-04  13.864  < 2e-16 ***
## Year.Remod.Add         5.162e-04  1.817e-04   2.841 0.004564 ** 
## Garage.Area            1.174e-04  1.862e-05   6.308 3.76e-10 ***
## Overall.Qual           5.916e-02  3.257e-03  18.165  < 2e-16 ***
## log(Lot.Area)          9.666e-02  8.119e-03  11.906  < 2e-16 ***
## BsmtFin.SF.1           7.896e-05  8.141e-06   9.700  < 2e-16 ***
## Overall.Cond           4.129e-02  2.760e-03  14.960  < 2e-16 ***
## Total.Bsmt.SF          9.179e-05  9.662e-06   9.501  < 2e-16 ***
## Central.AirY           4.975e-02  1.223e-02   4.069 4.98e-05 ***
## Bsmt.Full.Bath         2.446e-02  5.826e-03   4.199 2.85e-05 ***
## Screen.Porch           1.602e-04  4.472e-05   3.582 0.000352 ***
## Kitchen.QualFa        -1.039e-01  2.207e-02  -4.709 2.73e-06 ***
## Kitchen.QualGd        -5.401e-02  1.513e-02  -3.569 0.000370 ***
## Kitchen.QualTA        -7.573e-02  1.624e-02  -4.664 3.39e-06 ***
## Exter.QualFa          -7.914e-02  3.308e-02  -2.392 0.016884 *  
## Exter.QualGd          -2.498e-02  2.126e-02  -1.175 0.240253    
## Exter.QualTA          -1.936e-02  2.281e-02  -0.849 0.396240    
## Bldg.Type2fmCon       -1.976e-02  1.812e-02  -1.091 0.275618    
## Bldg.TypeDuplex       -6.840e-02  1.466e-02  -4.665 3.37e-06 ***
## Bldg.TypeTwnhs        -3.538e-02  1.965e-02  -1.800 0.072049 .  
## Bldg.TypeTwnhsE       -5.801e-03  1.389e-02  -0.418 0.676310    
## Bsmt.QualFa           -2.039e-02  2.114e-02  -0.964 0.335058    
## Bsmt.QualGd           -3.421e-02  1.255e-02  -2.725 0.006508 ** 
## Bsmt.QualNo Basement -5.622e-02  2.265e-02  -2.482 0.013185 *  
## Bsmt.QualTA           -4.078e-02  1.485e-02  -2.746 0.006112 ** 
## Garage.CondGd         -1.386e-02  3.984e-02  -0.348 0.727999    
## Garage.CondNo Garage  1.156e-02  1.814e-02   0.637 0.524265    
## Garage.CondTA          3.634e-02  1.503e-02   2.418 0.015746 *  
## NeighborhoodBrDale    -1.223e-01  3.511e-02  -3.484 0.000508 ***
## NeighborhoodBrkSide   -5.024e-02  3.197e-02  -1.572 0.116261    
## NeighborhoodClearCr    1.300e-02  3.389e-02   0.384 0.701314    
## NeighborhoodCollgCr   -8.371e-02  2.798e-02  -2.992 0.002819 ** 
## NeighborhoodCrawfor    5.529e-02  3.167e-02   1.746 0.081104 .  
## NeighborhoodEdwards   -1.281e-01  3.010e-02  -4.256 2.21e-05 ***
## NeighborhoodGilbert   -7.266e-02  2.952e-02  -2.461 0.013964 *  
## NeighborhoodGreens     5.698e-02  4.309e-02   1.322 0.186241    
## NeighborhoodIDOTRR    -1.367e-01  3.333e-02  -4.101 4.35e-05 ***
## NeighborhoodMeadowV   -1.636e-01  3.852e-02  -4.247 2.31e-05 ***
## NeighborhoodMitchel   -9.104e-02  2.990e-02  -3.045 0.002369 ** 
## NeighborhoodNAmes     -1.006e-01  2.908e-02  -3.459 0.000558 ***
## NeighborhoodNoRidge   -2.479e-02  3.087e-02  -0.803 0.422178    
## NeighborhoodNPkVill   -2.281e-02  3.331e-02  -0.685 0.493708    
## NeighborhoodNridgHt   -2.640e-02  2.960e-02  -0.892 0.372628    
## NeighborhoodNWAmes    -1.015e-01  2.967e-02  -3.420 0.000643 ***
## NeighborhoodOldTown   -1.110e-01  3.126e-02  -3.552 0.000394 ***
## NeighborhoodSawyer    -9.464e-02  3.021e-02  -3.132 0.001769 ** 
## NeighborhoodSawyerW   -1.057e-01  2.927e-02  -3.611 0.000315 ***
## NeighborhoodSomerst    3.355e-03  2.754e-02   0.122 0.903078    
## NeighborhoodStoneBr    1.455e-02  3.144e-02   0.463 0.643513    
## NeighborhoodSWISU     -8.399e-02  3.491e-02  -2.406 0.016270 *  
## NeighborhoodTimber    -7.561e-02  3.177e-02  -2.380 0.017455 *  
## NeighborhoodVeenker  -3.553e-02  3.691e-02  -0.963 0.335907
## Heating.QCFa         -2.819e-02  1.466e-02  -1.923 0.054739 .
## Heating.QCGd         -9.108e-03  7.133e-03  -1.277 0.201850
## Heating.QCTA         -1.393e-02  6.832e-03  -2.040 0.041574 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.08835 on 1428 degrees of freedom
## Multiple R-squared:  0.9412, Adjusted R-squared:  0.9389
## F-statistic: 408.3 on 56 and 1428 DF,  p-value: < 2.2e-16
```

Model Selection

We compared our base model and models with selected added terms from stepwise selection, then used anova to show the significance of adding those variables. All of the added variables are significant, which leads our to our final initial model.

```r
base = lm(log(price) ~ X1st.Flr.SF + Year.Built + Year.Remod.Add +
            Garage.Area + Overall.Qual + Kitchen.Qual + Neighborhood,
          data=ames_train)

base1 = lm(log(price) ~ area + Year.Built + Year.Remod.Add +
             Garage.Area + Overall.Qual + Kitchen.Qual +
             Neighborhood + log(Lot.Area) , data=ames_train)

base2 = lm(log(price) ~ area + Year.Built + Year.Remod.Add +
             Garage.Area + Overall.Qual + Kitchen.Qual +
             Neighborhood + log(Lot.Area) + BsmtFin.SF.1,
           data=ames_train)

base3 = lm(log(price) ~ area + Year.Built + Year.Remod.Add +
             Garage.Area + Overall.Qual + Kitchen.Qual + Neighborhood +
             log(Lot.Area) + BsmtFin.SF.1 + Overall.Cond , data=ames_train)

base4 = lm(log(price) ~ area + Year.Built + Year.Remod.Add +
             Garage.Area + Overall.Qual + Kitchen.Qual + Neighborhood +
             log(Lot.Area) + BsmtFin.SF.1 +  Overall.Cond + Total.Bsmt.SF,
           data=ames_train)

base5 = lm(log(price) ~ area + Year.Built + Year.Remod.Add + Garage.Area
           + Overall.Qual + Kitchen.Qual + Neighborhood + log(Lot.Area) +
             BsmtFin.SF.1 +  Overall.Cond + Total.Bsmt.SF + Central.Air, data=ames_train)

model1 = lm(log(price) ~ area + Year.Built + Year.Remod.Add + Garage.Area +
              Overall.Qual + Kitchen.Qual + Neighborhood +log(Lot.Area) +
              BsmtFin.SF.1 + Overall.Cond + Total.Bsmt.SF + Central.Air +
              Bsmt.Full.Bath + Screen.Porch  + Exter.Qual + Bldg.Type +
              Bsmt.Qual + Garage.Cond  + Heating.QC, data=ames_train)

kable(anova(base,base1,base2, base3, base4, base5, model1), caption = "ANOVA")
```

Res.Df	RSS	Df	Sum of Sq	F	Pr(>F)

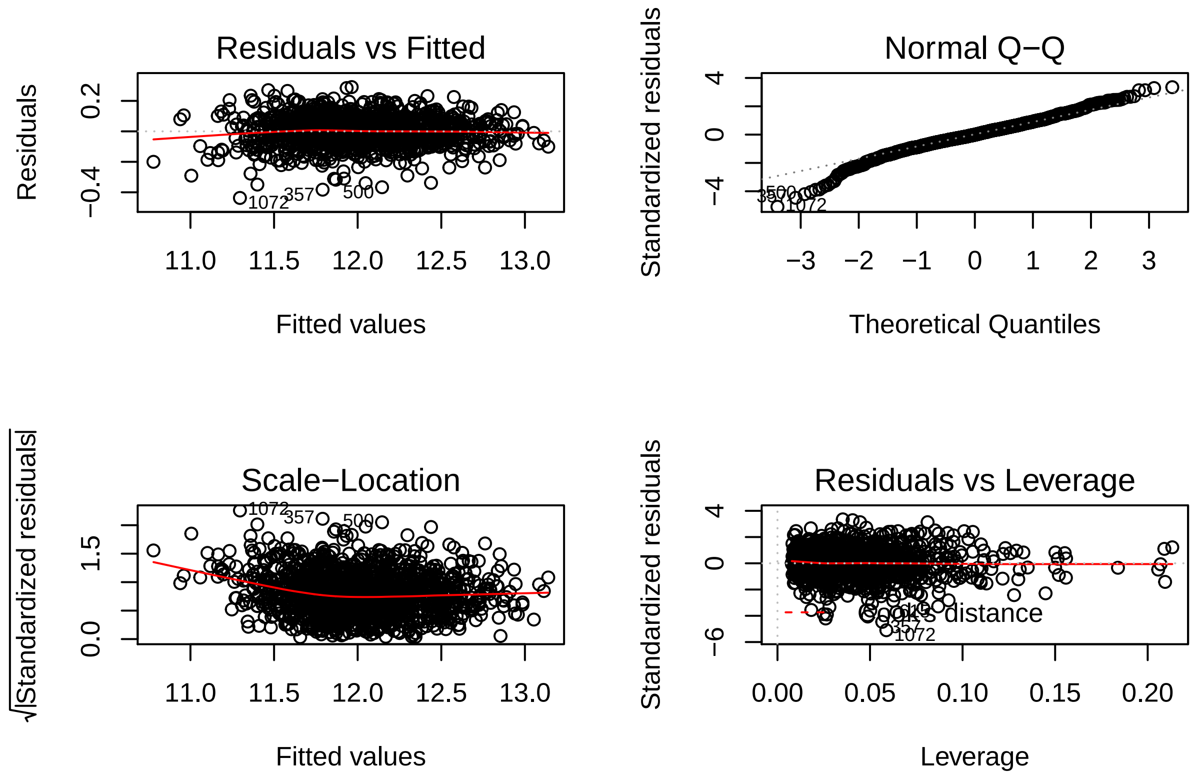
Table 1: ANOVA

Res.Df	RSS	Df	Sum of Sq	F	Pr(>F)
1452	29.32249	NA	NA	NA	NA
1451	19.35437	1	9.9681171	1277.023390	0
1450	15.67471	1	3.6796598	471.404135	0
1449	13.56107	1	2.1136434	270.780528	0
1448	12.32310	1	1.2379670	158.596939	0
1447	11.94493	1	0.3781686	48.447479	0
1428	11.14660	19	0.7983284	5.382867	0

Residual Diagnostics

The residual plots shows no non-constant variance, and the qq-plot shows a good normality of the redisuals except for a few potential outliers.

```r
par(mfrow = c(2,2)); plot(model1)
```



Model Evaluation

Since log transformation was used on `price`, it needs to be transformed back to the original scale.

```r
ames_test = clean_data(ames_test)
Yhat1 = predict(base, newdata=ames_test, interval = "pred")
Yhat1 = exp(Yhat1)
# test criteria
rmse = function(y, yhat){
  sqrt(mean((y-yhat)^2))
}

bias = function(yhat, y){
  mean(yhat-y)
}
maxDeviation = function(yhat, y){
  max(abs(yhat-y))
}
meanDeviation = function(yhat, y){
  mean(abs(yhat-y))
}
coverage = function(y, lwr, upr){
  mean(y>=lwr && y<=upr)
}
# evaluation
rmse1 = rmse(Yhat1[,1], ames_test$price)
bias1 = bias(Yhat1[,1], ames_test$price)
maxDeviation1 = maxDeviation(Yhat1[,1], ames_test$price)
meanDeviation1 = meanDeviation(Yhat1[,1], ames_test$price)
coverage1 = coverage(ames_test$price, Yhat1[,2], Yhat1[,3])
```

Model Checking

Based on our model above, we calculated prediction for the first observation in both the training and test data. For the training data, the prediction for the first observation (with PID 526354020) is 11.78. After converting it to original unit, we got 130614, which is very close to the true value 137000. For the test data, the prediction for the first observation is 12.21, which is 200787. This is also a reasonable prediction since the true value is 192100.

Complex Model

Variable Selection

We chose variables based on two conditions: colinearity and near zero variance. We first chose qualitative variables. We eliminated all categorical variable which have more than 1200 missing data. Then we use code book & reasoning and pairwise scatterplots to investigate collinearity among them. After selecting variables, we checked NAs in each variable and converted all NAs into a new categorical level. For quantative variables, we first investigated its skewness and did log transformation if skewness is greater than 0.75. Then we checked their variance and deleted those whose variance are nearly constant. Finally, We chose 36 variables : the type of dwelling ,the general zoning classification of the sale,Lot size, neighboorhood and others. All those variables have no missing value and no skewness over 0.75. In addition, the collinearity is insignificant in this case.

Final Model

We decided to use Gradient Boosting for linear regression to fit the data. We checked the data and first splitted the training set into qualitative and quantitative variables. And then we found out that among all qualitative ones, 12 have nearly zero variance, which indicate that the frequency of the most common value to the frequency of the second most common value is large. In this case, we deleted those variables. Among all quantitative variables, 8 have nearly zero variance, and thus we removed them. Since we chose quantitaive variables which have no missing values,there is no need to do imputation. Since imputation over qualitative variables introduces bias, we thus simply converted NAs into a new factor level.

Before fitting the model, we also preprocessed the data by scaling and doing PCA. Then, we need to convert qualitative predictors to dummy variables in order to use XGBoost model. Also, we set the depth of tree in each round to be 35 and ran the gradient boosting for 250 rounds.

For Xboost, we do not know how to draw the summary table, since we did PCA to reduce dimension, all selected features are hard to intepretate.

```
load("ames_train.Rdata")
load("ames_test.Rdata")
train_PID = ames_train$PID
test_PID = ames_test$PID
```

```
# clean data
clean_data = function(ames_train){
# replace NA with a new level
ames_train = ames_train %>% mutate(Alley = as.factor(ifelse(is.na(as.character(Alley)),
                                   "No alley access", as.character(Alley))),
           Bsmt.Qual = as.factor(ifelse(is.na(as.character(Bsmt.Qual)),
                                        "No Basement", as.character(Bsmt.Qual))),
           Bsmt.Cond = as.factor(ifelse(is.na(as.character(Bsmt.Cond)),
                                        "No Basement", as.character(Bsmt.Cond))),
           BsmtFin.Type.1 = as.factor(ifelse(is.na(as.character(BsmtFin.Type.1)),
                                        "No Basement", as.character(BsmtFin.Type.1))),
           BsmtFin.Type.2 = as.factor(ifelse(is.na(as.character(BsmtFin.Type.2)),
                                        "No Basement", as.character(BsmtFin.Type.2))),
           Bsmt.Exposure = as.factor(ifelse(is.na(as.character(Bsmt.Exposure))|
                                           as.character(Bsmt.Exposure) == "",
                                        "No Basement", as.character(Bsmt.Exposure))),
           Bsmt.Unf.Rate.SF = ifelse(Total.Bsmt.SF!=0, Bsmt.Unf.SF/Total.Bsmt.SF, 0),
           Bsmt.Full.Bath = ifelse(is.na(Bsmt.Full.Bath),0,Bsmt.Full.Bath),
           Bsmt.Half.Bath = ifelse(is.na(Bsmt.Half.Bath),0,Bsmt.Half.Bath),
           Fireplace.Qu = as.factor(ifelse(is.na(as.character(Fireplace.Qu)),
                                        "No Fireplace", as.character(Fireplace.Qu))),
           Garage.Type = as.factor(ifelse(is.na(as.character(Garage.Type)),
                                        "No Garage", as.character(Garage.Type))),
           Garage.Finish = as.factor(ifelse(is.na(as.character(Garage.Finish))|
                                           as.character(Garage.Finish) == "",
                                        "No Garage", as.character(Garage.Finish))),
           Garage.Qual = as.factor(ifelse(is.na(as.character(Garage.Qual)),
                                        "No Garage", as.character(Garage.Qual))),
           Garage.Cond = as.factor(ifelse(is.na(as.character(Garage.Cond)),
                                        "No Garage", as.character(Garage.Cond))),
           Pool.Area = as.factor(ifelse(Pool.Area==0,
                                        "N", "Y")),
           Fence = as.factor(ifelse(is.na(as.character(Fence)),
                                        "No Fence", as.character(Fence))),
          Misc.Feature = as.factor(ifelse(is.na(as.character(Misc.Feature)),
                                        "None", as.character(Misc.Feature))),
           Mas.Vnr.Type = as.factor(ifelse(as.character(Mas.Vnr.Type) == "",
                                        "None", as.character(Mas.Vnr.Type))),
```

```r
           Mas.Vnr.Area = ifelse(is.na(Mas.Vnr.Area),0,Mas.Vnr.Area),
           Kitchen.Qual = as.factor(ifelse(as.character(Kitchen.Qual)=="Po",
                                           "Fa", as.character(Kitchen.Qual))),
           Heating.QC = as.factor(ifelse(as.character(Heating.QC)=="Po",
                                         "Fa", as.character(Heating.QC))),
           Garage.Cond = as.factor(ifelse(as.character(Garage.Cond)=="Ex",
                                          "Gd", as.character(Garage.Cond))),
           Garage.Qual = as.factor(ifelse(as.character(Garage.Qual)=="Ex",
                                          "Gd", as.character(Garage.Qual))),
           Electrical = as.factor(ifelse(as.character(Electrical) == "",
                                         "SBrkr", as.character(Electrical))),
           Condition.2 = as.factor(ifelse(as.character(Condition.2) == "Artery",
                                          "Feedr", as.character(Condition.2))
           ))

#delete outliers
#ames_train = ames_train[ames_train$X1st.Flr.SF<3000,]
# do log transformation
ames_train = ames_train %>%
  mutate(price = price,
         area = log(area),
         Lot.Area = log(Lot.Area),
         Mas.Vnr.Area = log(Mas.Vnr.Area+1),
         BsmtFin.SF.1 = log(BsmtFin.SF.1+1),
         BsmtFin.SF.2 = log(BsmtFin.SF.2+1),
         Bsmt.Unf.SF = log(Bsmt.Unf.SF+1),
         Total.Bsmt.SF = log(Total.Bsmt.SF+1),
         X1st.Flr.SF = log(X1st.Flr.SF),
         X2nd.Flr.SF = log(X2nd.Flr.SF+1),
         Low.Qual.Fin.SF = log(Low.Qual.Fin.SF+1),
         Garage.Area = log(Garage.Area+1),
         Wood.Deck.SF = log(Wood.Deck.SF+1),
         Open.Porch.SF = log(Open.Porch.SF+1),
         Enclosed.Porch = log(Enclosed.Porch+1),
         X3Ssn.Porch = log(X3Ssn.Porch+1),
        Screen.Porch = log(Screen.Porch+1),
         Misc.Val = log(Misc.Val+1),
         TotalSq = log(TotalSq),
        Year.Built= Year.Built^2)
# select variables based on reasoning & code book, ggpair plot and near zero variance test.
ames_train= ames_train[,c("MS.SubClass","MS.Zoning","Lot.Area","Neighborhood",
                          "House.Style","Overall.Qual","Overall.Cond","Exterior.1st",
                          "Year.Remod.Add","Mas.Vnr.Area","Exter.Qual","Bsmt.Qual","BsmtFin.Type.1",
                          "Heating.QC","Full.Bath","X1st.Flr.SF","Kitchen.Qual","Fireplace.Qu",
                          "Bsmt.Exposure","Garage.Type","Wood.Deck.SF","price","Open.Porch.SF",
                          "Mas.Vnr.Type","Foundation","BsmtFin.SF.1","Bsmt.Unf.SF","Garage.Area",
                          "Garage.Finish",'Condition.1','TotalSq',"Central.Air","Kitchen.AbvGr",
                          "Bedroom.AbvGr","X2nd.Flr.SF","Lot.Shape","Year.Built","PID")]
}
```

```r
# remove outliers
ames_train = ames_train[-c(462,168,183),]
remove_idx1 = c(1:nrow(ames_train))[ames_train$Neighborhood %in%
                                    c("Gilbert")&ames_train$price>350000]
```

```r
remove_idx2 = c(1:nrow(ames_train))[ames_train$Neighborhood %in%
                                    c("NAmes")&ames_train$price>300000]
remove_idx3 = c(1:nrow(ames_train))[ames_train$Neighborhood %in%
                                    c("Landmrk","GrnHill")]
ames_train = ames_train[-c(remove_idx1, remove_idx2, remove_idx3),]
ames_train = ames_train[-which(ames_train$price>350000),]
ames_train = ames_train[-which(ames_train$price<50000),]
# clean both training and test dataset
ames_train = clean_data(ames_train)%>% dplyr:: select(-PID)
pid = ames_test$PID
ames_test = clean_data(ames_test) %>% dplyr:: select(-PID)
```

```r
#check missing data
missing.summary <- sapply(ames_train, function(x) sum(is.na(x)))
indexs.missing <- sapply(ames_train, function(x) sum(is.na(x))) > 0
num.variable.missing <- length(missing.summary[indexs.missing])
freq.table.miss <- data.frame(Variable =
                                names(missing.summary[indexs.missing]),
                              Number.of.Missing =
                                as.integer(missing.summary[indexs.missing]),
                              Porcentage.of.Missing=
                                as.numeric(prop.table(missing.summary[indexs.missing])))
freq.table.miss <- freq.table.miss %>%
  dplyr::select(Variable:Porcentage.of.Missing) %>%
  arrange(desc(Number.of.Missing))
indexs <- missing.summary < 200
training <- ames_train[, indexs]
# We retain SalePrice
SalePrice <- training$price
# We split the train data set into quantitative variables and qualitatives variables.
indexs.quantitative <- sapply(training, function(x) is.numeric(x))
training.quantitative <- training[, indexs.quantitative]
training.qualitative <- training[, !indexs.quantitative]
# we did near zero variance test.
nzv <- nzv(training.qualitative, saveMetrics = TRUE)
training.qualitative <- training.qualitative[, !nzv$nzv]
nzv2 <- nzv(training.quantitative, saveMetrics = TRUE)
training.quantitative <- training.quantitative[, !nzv2$nzv]
# We did pre-process tranformation here (centering, scaling and PCA) on original datasets.
training.quantitative.imputed <- training.quantitative
pre.proc <- preProcess(training.quantitative.imputed,
                       method = c("center", "scale", "pca"), thresh = 0.9)
training.quantitative.imputed.pc <- predict(pre.proc, training.quantitative.imputed)
# We create a matrix with dummy variables.
dummies <- dummyVars(~ ., data=training.qualitative)
training.dummies <- as.data.frame(predict(dummies, training.qualitative))
training.imputed <- cbind(training.dummies, training.quantitative.imputed.pc)
## do log transformation on price.
training.imputed$SalePrice <- log(SalePrice)
num.variables <- dim(training.imputed)[2]
train.xgboost <- training.imputed
```

```r
# We fit the xgboost model.
house.xgboost <- xgboost(data = data.matrix(train.xgboost[,-num.variables]),
                         label=data.matrix(train.xgboost[,num.variables]),
                 booster = "gblinear",
                 objective = "reg:linear",
                 max.depth = 35,
                 nround = 250,
                 lambda = 0,
                 lambda_bias = 0,
                 alpha = 0,
                 missing=NA,
                 verbose = 0)
```

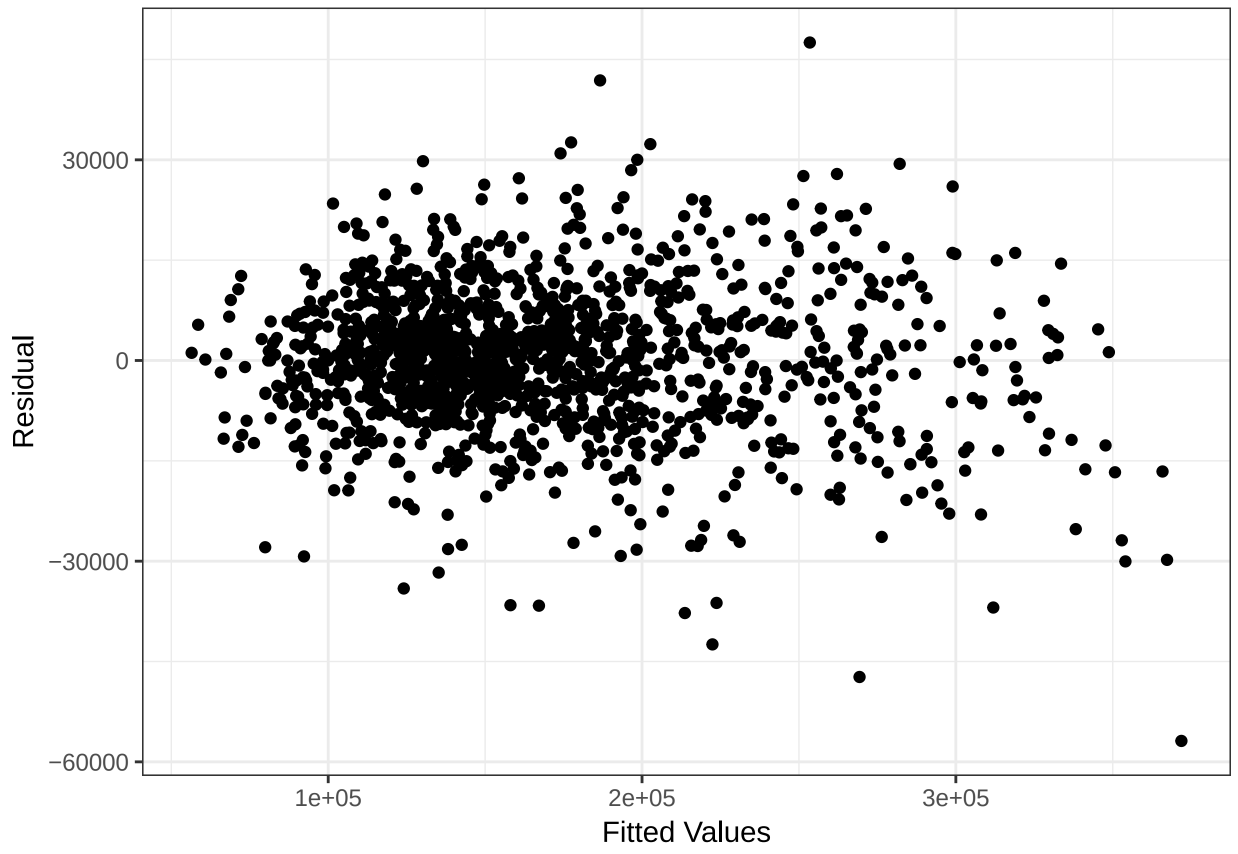
```
#define the transform function
transform_data = function(ames_test){
testing <- ames_test[, indexs]
indexs.quantitative <- sapply(testing, function(x) is.numeric(x))
#split the test data into quantitative and qualitative parts.
testing.quantitative <- testing[, indexs.quantitative]
testing.qualitative <- testing[, !indexs.quantitative]
#apply near zero variance test.
testing.qualitative <- testing.qualitative[, !nzv$nzv]
testing.quantitative <- testing.quantitative[,!nzv2$nzv]
testing.quantitative.imputed <- testing.quantitative
testing.quantitative.imputed.pc <- predict(pre.proc, testing.quantitative.imputed)
testing.dummies <- as.data.frame(predict(dummies, testing.qualitative))
testing.imputed <- cbind(testing.dummies, testing.quantitative.imputed.pc)
return (data.matrix(testing.imputed))}
```

```
# prediction
pred.training.xgboost <- predict(house.xgboost, transform_data(ames_test), missing=NA)
Yhat2 = exp(pred.training.xgboost)
```

Model Evaluation

1. Model Evaluation. For our complex model, we plotted residuals against fitted values. All the residuals clound around zero with no particular patterns, so our model does a decent job.

```
# training dataset
pred.training.xgboost <- predict(house.xgboost, transform_data(ames_train), missing=NA)
Yhat.train = exp(pred.training.xgboost)
# residual plot
residul = ames_train$price - Yhat.train
qplot(x = Yhat.train, y = residul) +
  theme_bw()+
  xlab("Fitted Values")+
  ylab("Residual")
```



We also checked the model using the following criteria. Coverage is not available for boosting. The results of both the simple model and the complex model are shown in the table below. Our complex model greatly improves prediction accuracy.

```r
rmse2 = rmse(Yhat2, ames_test$price)
bias2 = bias(Yhat2, ames_test$price)
maxDeviation2 = maxDeviation(Yhat2, ames_test$price)
meanDeviation2 = meanDeviation(Yhat2, ames_test$price)

res = data.frame(rmse = c(rmse1, rmse2),
                 bias = c(bias1, bias2),
                 maxDeviation = c(maxDeviation1, maxDeviation2),
                 meanDeviation = c(meanDeviation1, meanDeviation2),
                 coverage = c(coverage1, NA)
                 )
rownames(res) = c("simple model", "complex model")
kable(res, caption = "Model Evaluation")
```

Table 2: Model Evaluation

	rmse	bias	maxDeviation	meanDeviation	coverage
simple model	24736.35	-1533.2575	152712.76	18151.297	1
complex model	11591.92	531.2901	85781.68	8463.684	NA

2. Model Checking. Since gradient boosing is hard to compute by hand, we cannot directly use calculators to obtain house price of first observation in both train and test datasets. Instead, we used R function

to calculate them. For first observation in the training data, the prediction is 143400, which is a bit greater than the true value 137000. For the training data, the first prediction is 194059, close to the true value 192100.

3. Model Results. Top 10 undervalued and overvalued houses are shown in the tables below. The most undervalued house is the one with parcel ID 528102010. We may invest in this house and sell it after the price rises. The most overvalued house is the one with parcel ID 921128020. We may sell this house now since its value may drop in the future.

```r
residual = Yhat2 - ames_test$price
ntest = dim(ames_test)[1]
nleft = ntest - 10

least_over = sort(residual, partial=nleft)[nleft]
id1 = which(residual > least_over)
df1 = data.frame(
  PID = test_PID[id1],
  Predicted.Value = Yhat2[id1],
  Real.Value = ames_test$price[id1],
  Difference = residual[id1]
)
kable(df1, caption = "Undervalued Houses")
```

Table 3: Undervalued Houses

PID	Predicted.Value	Real.Value	Difference
535383070	190781.7	105000	85781.68
905377020	156380.5	130000	26380.55
528116010	318839.3	296000	22839.28
533251110	276656.2	255000	21656.22
528166120	440086.3	412500	27586.33
909452114	238349.1	203135	35214.09
528102010	359746.4	315000	44746.44
905200160	103925.1	80000	23925.12
532351140	133571.7	112000	21571.73
527182020	156061.6	130000	26061.57

```r
residual = ames_test$price - Yhat2
ntest = dim(ames_test)[1]
nleft = ntest - 10
id2 = which(residual > least_over)
df2 = data.frame(
  PID = test_PID[id2],
  Predicted.Value = Yhat2[id2],
  Real.Value = ames_test$price[id2],
  Difference = -residual[id2]
)
kable(df2, caption = "Overvalued Houses")
```

Table 4: Overvalued Houses

PID	Predicted.Value	Real.Value	Difference
909275110	205536.59	238000	-32463.41
535454070	134535.54	166000	-31464.46

PID	Predicted.Value	Real.Value	Difference
903400220	187095.19	214500	-27404.81
534128190	220476.82	256900	-36423.18
533254050	218032.71	241600	-23567.29
916226030	207580.74	241500	-33919.26
921128020	332128.06	378500	-46371.94
528365060	315977.23	341000	-25022.77
533206020	255458.24	280750	-25291.76
903426180	126924.84	149500	-22575.16
905201080	162691.01	187000	-24308.99
905427010	198606.82	235000	-36393.18
903450110	85850.28	110000	-24149.72
905376090	176323.94	216000	-39676.06

Conclusion

The whole project is aimed at predicting prices of properties in Ames area. Based on the explanations in the codebook, we cleaned the NAs by creating new levels and then fitted a linear model as our initial model with the help of forward selection. In the process of improving our model by forward selection, we found it always included some variables with problem of colinearity. Thus before we fit our complex model, we decided to first remove the problemtic variables. The problemetic variables include variables with few unique values, variables with possible colinearity, and variables with a lot of NAs. We also identified a few continuous variables needing log transformations. Using these transformed and selected variables, we decided to use booting on linear model. The result given by booting model highly improved the prediction accuracy. After satisfied with our further pruned model, we finally predicted the properties that are overvalued and undervalued.

When the number of predictors is very large, cleaning data and removing the redundant variables by reasoning is the most crucial step for making a good prediction, and it is even more important than finding a good model, because we found that forward selcetion in such circumstance is not able to deal with the problem of colinearity well, and the accuracy of different models do not have much difference. We found removing outliers is also important for making a good prediction and improving the accuracy. Booting is always a good model to try when we want to further imporve our model.