# STAT 425 Project Final Report

*Yiming Gao, Yuan Yuan*

*2016/12/14*

## 1 Motivation

According to the article "U.S. existing home sales decline, prices still strong", the demand for housing will increase due to the millennials, "a demographic group that has seen an improvement in job prospects since the end of the 2007-2009 recession". For the increasing demand in housing, predicting the house price will attract more and more attention. It is important for any actors in the property market to have an estimation for the house they are interested in. Sellers would ask for a price based on the valuation, while buyer would have to consider if that certain real estate is worth for the asked price or not, and how much they would like to pay for it. In this project, We are planning to use dataset containing house sale prices for King County, which includes Seattle from Kaggle. It includes homes sold between May 2014 and May 2015. The main response is the house price, and the predictors are house features including continuous and categorical ones.

The potential user could be people who are interested in investing real estates, real estate agencies. The data product will help potential users to have a reference when they want to either buy or sell a house, since there are many factors for people to make a final deal. And the negotiation skills of buyers and sellers may also come into play in reality. It is said in the article of "7 Online Tools to Help You Estimate Your Home???s Value" by Teresa Mears that people have to consider the kinds of variables that went into the estimate. The variables used in different models for the data product has been showed for users' consideration. Potential users can also have an idea of the overall housing market in King County through our data product.

There are many websites that provides house price estimation which are similar to the data product we produced such as Zillow, Redfin, Chase and Bank of America. For example, Redfin is a tool that shows you photos and listing information for the exact variables used to arrive at the value of your home. All of the online tools predict the house price by running through computer models using publicly available data. However, the exact dataset they are using is still private, same as the formulas used in modeling. And the available data also has effects on the accuracy of the prediction. As it is said in the article by Teresa Mears, it is difficult to price a house if it is an outlier. For example, a house is not similar to houses in the data will have less accurate prediction.

# 2 Exploratory Data Analysis

## 2.1 Variable Description

The housing data set has 21613 rows and 19 features with the target feature price. For prediction purpose, I dropped some irrelevant features such as latitute, longitute, zipcode and so on.

First we view the data, and find that some variables have many zero values, for example, waterfront. In this case, value 1/0 means whether it is waterfront or not. Some variables such as floors, condition and grade, we will treat them as categorical variables in this part, but will convert them to numeric later when we build models. We can gain some insight on the number of houses that were renovated. According to the variable description, if yr_renovated is not zero, then the house was renovated at that year. We can find out that 914 of them were renovated, and the percentage is 4.23%.

Categorical features, numeric features and percentage of renovation are as follows.

```
## [1] "id"        "date"      "floors"    "condition" "grade"     "zipcode"
```

```
##  [1] "price"         "bedrooms"      "bathrooms"     "sqft_living"
##  [5] "sqft_lot"      "waterfront"    "view"          "sqft_above"
##  [9] "sqft_basement" "yr_built"      "yr_renovated"  "lat"
## [13] "long"          "sqft_living15" "sqft_lot15"
```

```
## Percentage of houses renovated is 4.23 %.
```

## 2.2 Training and Test Set

We will use a training set to discover potentially predictive relationships, and a test set to assess the strength and utility of the predictive relationship. In this report, all the models and predictions are basd on the training and test set (with seed 1234). We choose to randomly separate out 70% of observations as data in our training set.

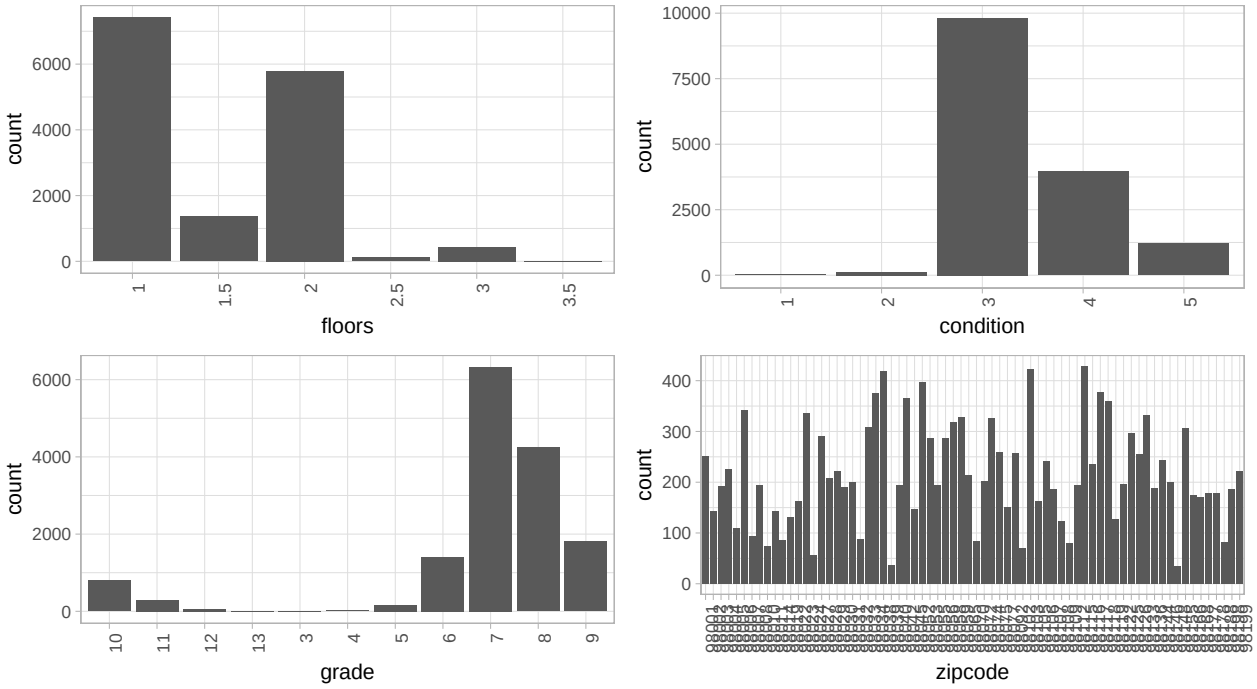
```
## Training set has 15129 rows and 21 columns.
```

```
## Testing set has 6484 rows and 21 columns.
```

## 2.3 Feature Visualization

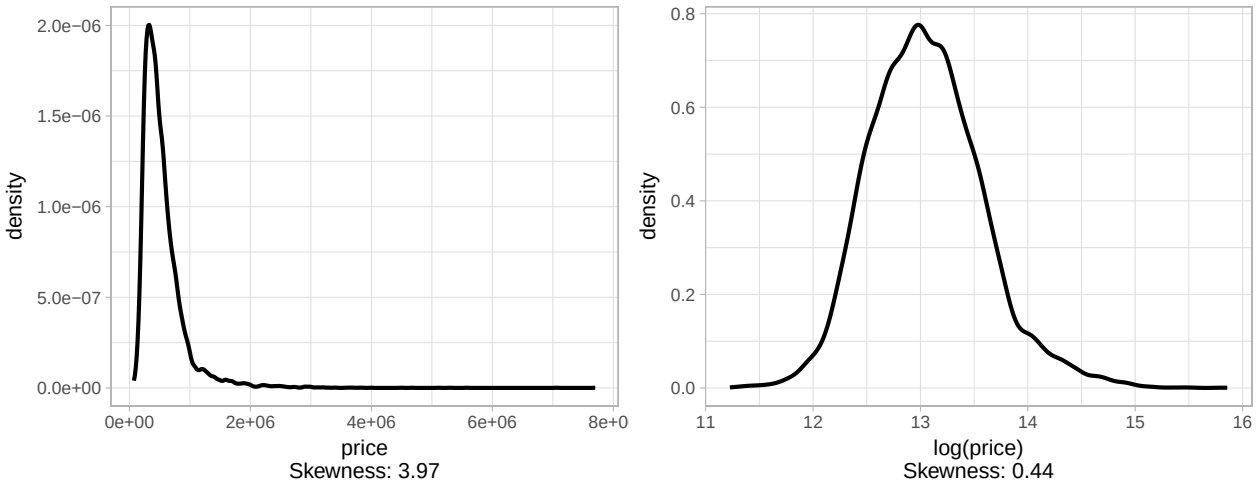
**- Barplots**

In order to get more insight into the data, we first create barplots for categorical features, indicating the distribution of different levels. As we can see, the majority of houses are on floor 1 or 2, with condition level at 3 or 4, and grade ranges from 6 to 9.



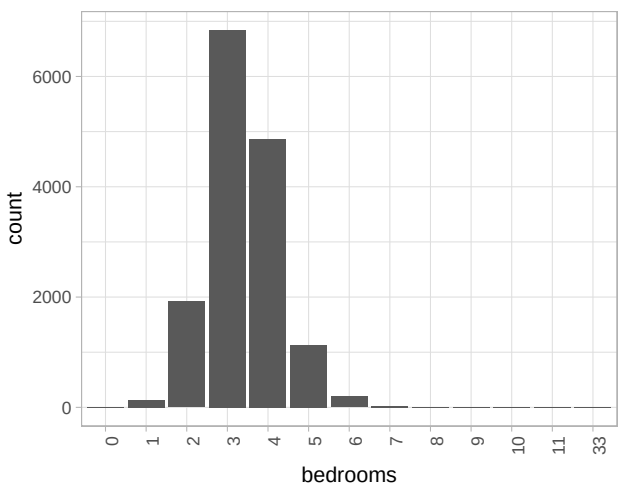

**- Density Plots**

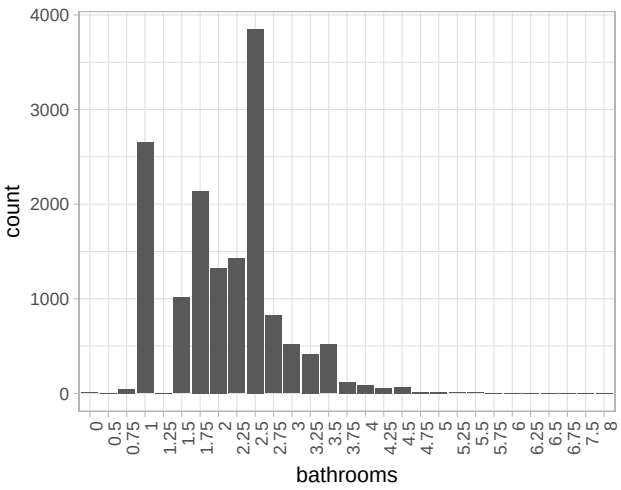
Density plots of the features indicates that the features are skewed. I only draw the density for target feature **price** and log-transformed price here. We can see that price before log transformation is highly skewed.




**- Histograms**

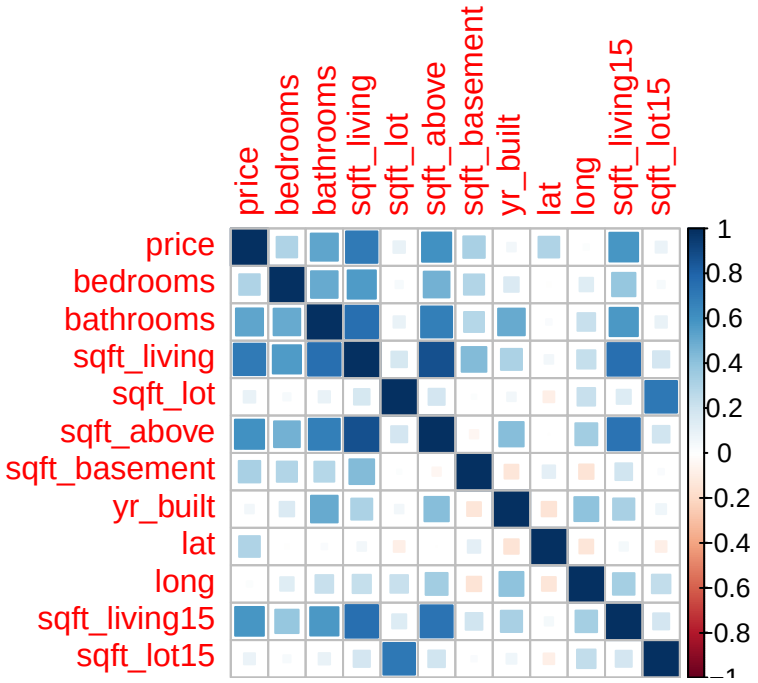
The histograms below show that majority of the houses have 3 or 4 bedrooms and 2.5 bathrooms (full baths and half baths).




**- Correlation Plot**

The plot below show that there are 4 variables are highly correlated with price. However, some of the variables are also highly correlated. We need to discover their relationships by further detailed research.




# 3 Linear Regression

## 3.1 Full Model

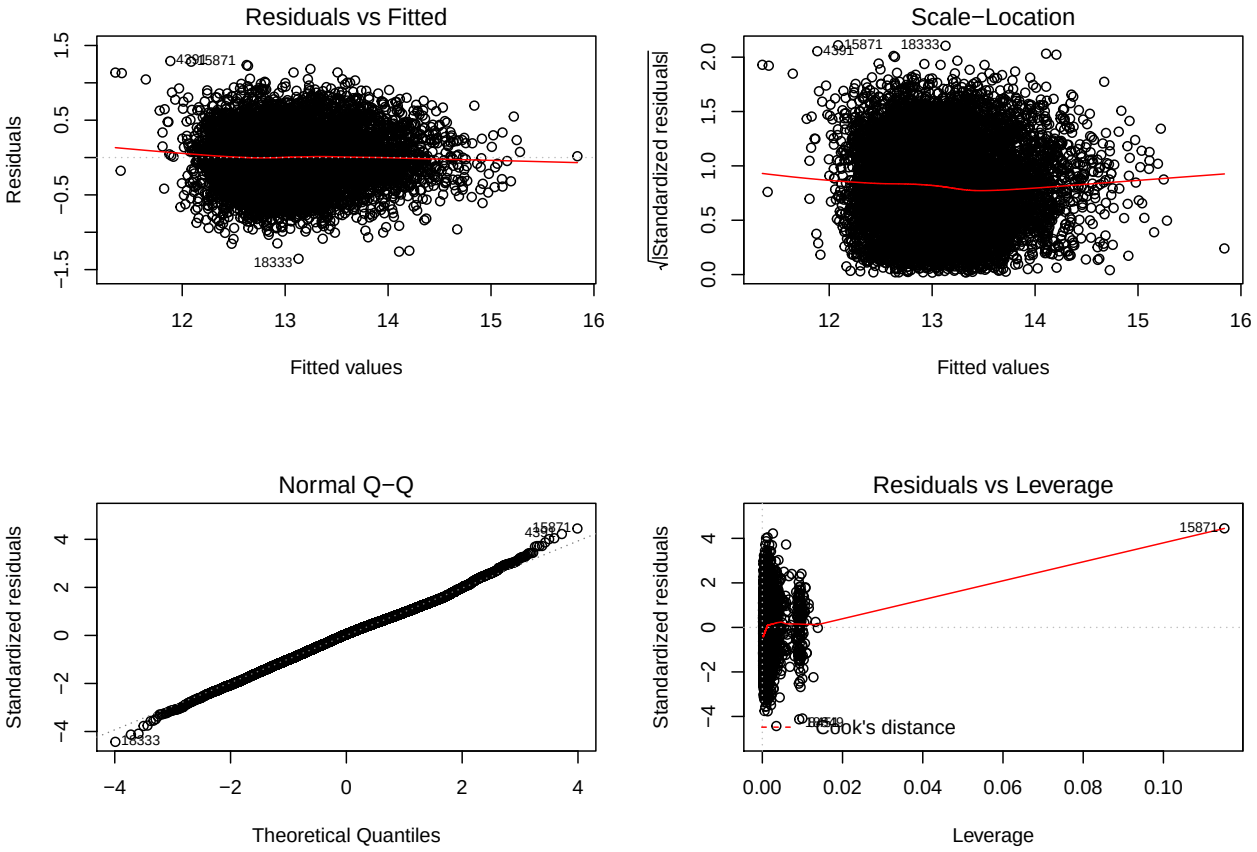
**- Model Summary**

We first remove some unnecessary features, and do some data convertion. Then we want to do log transformation on those highly skewed features with skewness larger than 0.75. So we do the transformation on price, sqft_ living, sqft_ lot, sqft_ above, sqft_ basement, sqft_ living 15 and sqft_ lot15.

```
##         price   sqft_living      sqft_lot    sqft_above sqft_basement
##      4.023511      1.471351     13.058206      1.446464      1.577746
## sqft_living15    sqft_lot15
##      1.108027      9.505424
```

Since the output of the full model is too lengthy, I will omit it in the report. All predictors excluding sqft_ lot are statistically significant with p-values less than 0.05. There is a completely linear relationship: sqft_ living = sqft_ above + sqft_ basement, i.e. rank deficiency. However, the result will not be affected. $R^2 = 0.6585$, which is good.

The diagnostic plots of the full model are as follows, they don't seem to show any potential problems, which the assumptions are not been violated.




**- Prediction**

We make the prediction on the test set, and compute the root mean squared error between actual values and predicted values in package "Metrics". The root mean squared error is 0.3043 for the full model.

```
## [1] 0.3043
```

## 3.2 Reduced Model

**- Variable Selection**

We use **AIC** and **BIC** criteria to determine the "best" linear model. Note that in BIC, we use the penalty of $logm$ where m is the size of the training set, in this case, $m = 15129$.

Both AIC and BIC procedure give us same 8 variables: bathrooms, sqft_ living, waterfront (with value 1), view, grade, yr_ built, sqft_ living15 and sqft_ lot15. Then we will fit a linear model based on those variables.

```
## [1] "(Intercept)"   "bathrooms"     "sqft_living"   "waterfront1"
## [5] "view"          "grade"         "yr_built"      "sqft_living15"
## [9] "sqft_lot15"
```

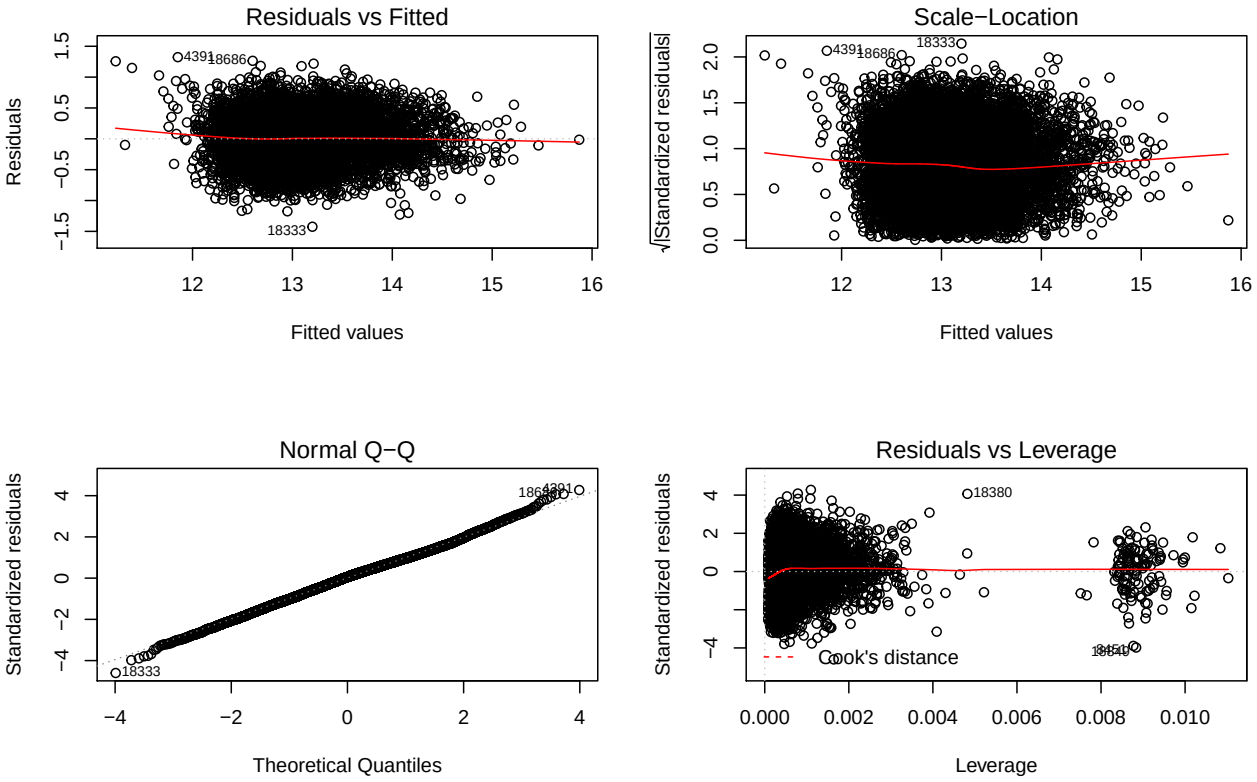
**- Reduced Model**

We built the model with 8 predictors selected by AIC and BIC. The model summary and diagnostic plots are as follows. We can see that $R^2$ is 0.6549, meaning 65.49% of the total variation is price can be explained by these variables. All the predictors are statistically significant. Compared to the full model, we can say that this reduced model is adequate enough, especially in term of simplicity.

The diagnostic plots don't show any potential problems either. It means the reduced model is acceptable. Then we compare its prediction strength with the full model.

```
##
## Call:
## lm(formula = price ~ bathrooms + sqft_living + waterfront + view +
##     grade + yr_built + sqft_living15 + sqft_lot15, data = train.reg)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.42481 -0.20482  0.01693  0.20870  1.32336
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)   19.228800   0.220218   87.32   <2e-16 ***
## bathrooms      0.077836   0.005674   13.72   <2e-16 ***
## sqft_living    0.287917   0.012186   23.63   <2e-16 ***
## waterfront1    0.412602   0.031230   13.21   <2e-16 ***
```

```
## view           0.045664   0.003863   11.82   <2e-16 ***
## grade          0.220039   0.003549   62.01   <2e-16 ***
## yr_built      -0.005812   0.000105  -55.35   <2e-16 ***
## sqft_living15  0.254099   0.012788   19.87   <2e-16 ***
## sqft_lot15    -0.075471   0.003479  -21.69   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.3098 on 15120 degrees of freedom
## Multiple R-squared:  0.6549, Adjusted R-squared:  0.6547
## F-statistic:  3586 on 8 and 15120 DF,  p-value: < 2.2e-16
```




**- Prediction**

We perform the prediction on the test data.The root mean squared error is 0.3084 for this reduced model.

```
## [1] 0.3084
```

# 4 XGBoost

## 4.1 Feature Importance

XGBoost stands for Extreme Gradient Boosting, which can build a model and make predictions. It supports various objective functions, including *regression*, *classification* and *ranking*. In particular, XGBoost can manage huge dataset very efficiently. That's why we consider using it here.

First we need to transfer dummy variable **waterfront** into numeric form. We simply transfer them by implementing "as.integer" method.

We will build the model using the following parameters:

- objective = "reg:linear": we will train a linear regression model
- max.depth=4: the depth of trees is 4
- nthread = 2: the number of cpu threads we are going to use
- nround = 10: there will be ten passes on the data

```
g.xgb <- xgboost(data = train.xgb, max.depth = 4, eta = 1, nthread = 2,
                 nround = 10, objective = "reg:linear")
```

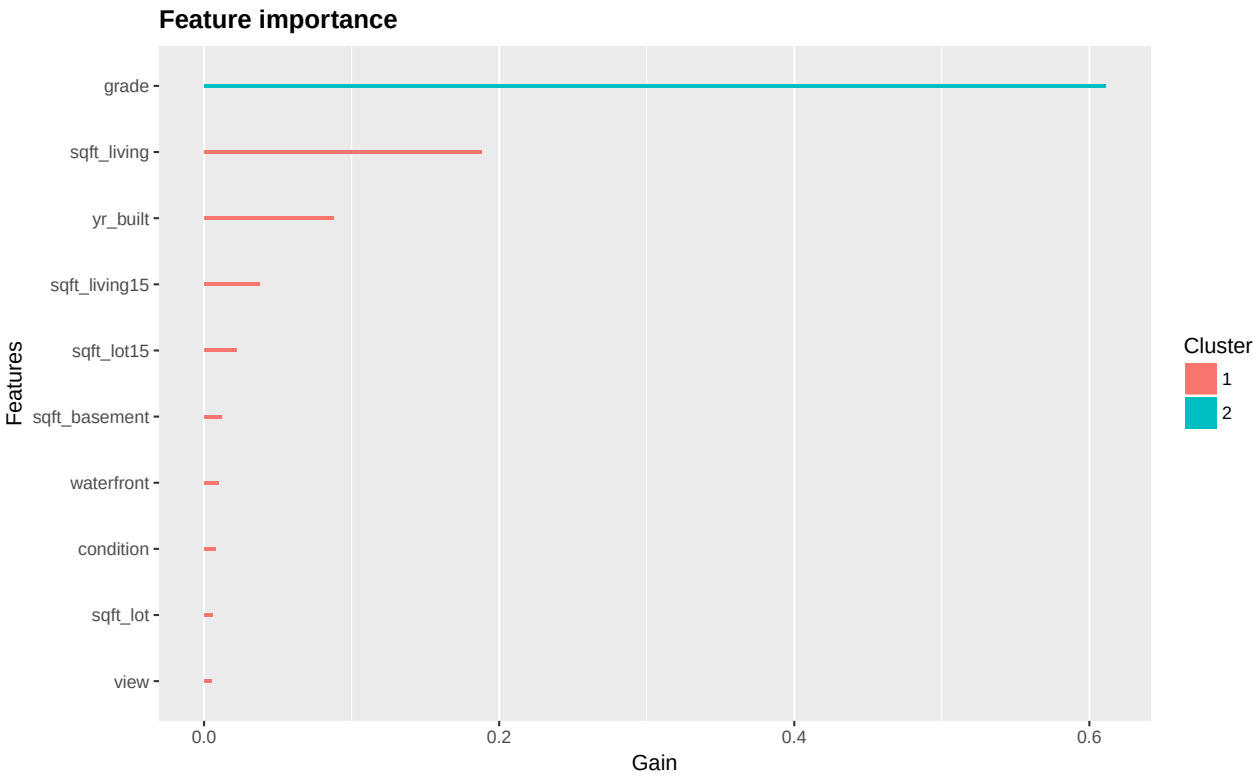
```
## [0]  train-rmse:0.402935
## [1]  train-rmse:0.347654
## [2]  train-rmse:0.315588
## [3]  train-rmse:0.305108
## [4]  train-rmse:0.298019
## [5]  train-rmse:0.292901
## [6]  train-rmse:0.289259
## [7]  train-rmse:0.286875
## [8]  train-rmse:0.284761
## [9]  train-rmse:0.283238
```

```
importance<-xgb.importance(feature_names = names(train.reg)[2:15],
                           model = g.xgb)
```

```
as.data.frame(importance)[1:10,]
```

```
##          Feature        Gain      Cover  Frequence
## 1          grade 0.610778217 0.16561933 0.13868613
```

```
## 2    sqft_living 0.188060902 0.17997783 0.17518248
## 3       yr_built 0.088077027 0.17196630 0.11678832
## 4  sqft_living15 0.037832202 0.09380235 0.11678832
## 5     sqft_lot15 0.021697325 0.05093666 0.10218978
## 6  sqft_basement 0.012003704 0.05623804 0.05839416
## 7     waterfront 0.009992712 0.02645272 0.00729927
## 8      condition 0.007588109 0.05459273 0.04379562
## 9       sqft_lot 0.005591871 0.07792442 0.07299270
## 10          view 0.004786621 0.02532321 0.02189781
```




The feature importance plot shows that in this training set (70% with seed 1234), the first 5 most important features for prediction are building grade (from 1 to 13), sqft_ living (total square footage of the house), yr_ built (when the house was built), sqft_ living15 (the average square footage of the 15 closest houses) and sqft_ lot15 (lot size of the house).

## 4.2 Prediction

To measure the model performance, we will make the prediction on the test set based on this XGBoost model, and compute the RMSE. The root mean square error is 0.3005, which is the smallest among models till now.

```
## [1] 0.3005
```

When I run the XGBoost model, I found that it wouldn't take a long time, especially in comparison with the random forest model which I'll mention later in the report. This is determined by both algorithm and implementation. We have to wait on training models on such a large data set in most cases. The training speed of XGBoost impressed me a lot. We are confident that XGBoost is one of the fastest learning algorithm of gradient boosting algorithm.

# 5 Random Forest

Instead of growing a single tree, we grow many, and introduce some sources of randomness. When it comes to problems where the focus is not so much in understanding the data, but in making predictions.

## 5.1 Model Summary

**- Gini Importance**

We first create a random forest with 500 trees (with "Randomforest" package). From the output below, we know that the amount of total variance of price explained by the random forest model is 72.49%.

We also output the importance matrix, in which *IncNodePurity* is the total decrease in node impurities, measured by the Gini Index from splitting on the variable, averaged over all trees. We reordered the importance by decreasing IncNodePurity. In this case, the first 5 most important features are grade, sqft_ living, sqft_ living15, yr_ built and sqft_ above.

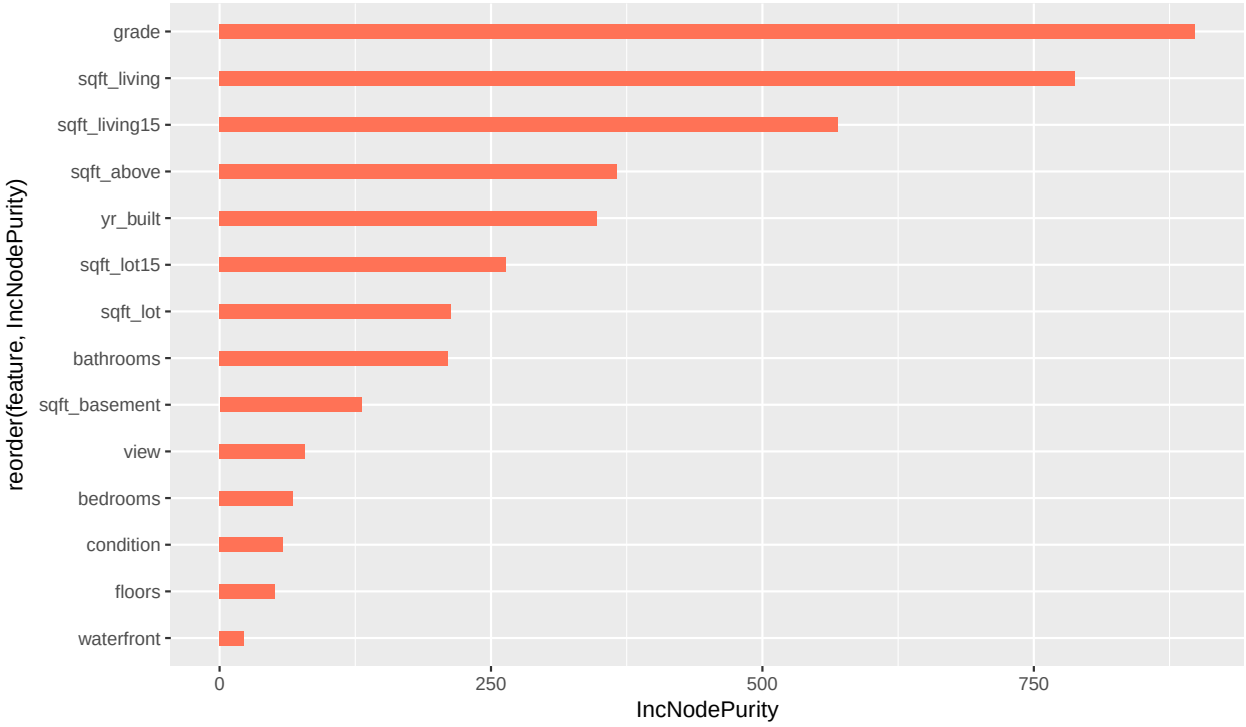
Gini importance, is based on Gini gain criterion employed in most traditional classification tree algorithms.

```
##
## Call:
##  randomForest(formula = price ~ ., data = train.reg, importance = TRUE)
##                Type of random forest: regression
##                      Number of trees: 500
## No. of variables tried at each split: 4
##
##           Mean of squared residuals: 0.07652468
##                     % Var explained: 72.47

##      imp.feature imp.IncNodePurity
## 1          grade         898.69393
## 2    sqft_living         787.80545
## 3  sqft_living15         569.32951
## 4     sqft_above         365.74195
```

```
## 5       yr_built         347.84557
## 6     sqft_lot15         263.50080
## 7       sqft_lot         213.10024
## 8      bathrooms         210.55273
## 9  sqft_basement         130.69750
## 10          view          78.94918
## 11      bedrooms          67.69852
## 12     condition          58.10139
## 13        floors          51.09844
## 14    waterfront          22.13145
```

We also plot the reordered feature importance for better understanding.




**- Permutation Importance**

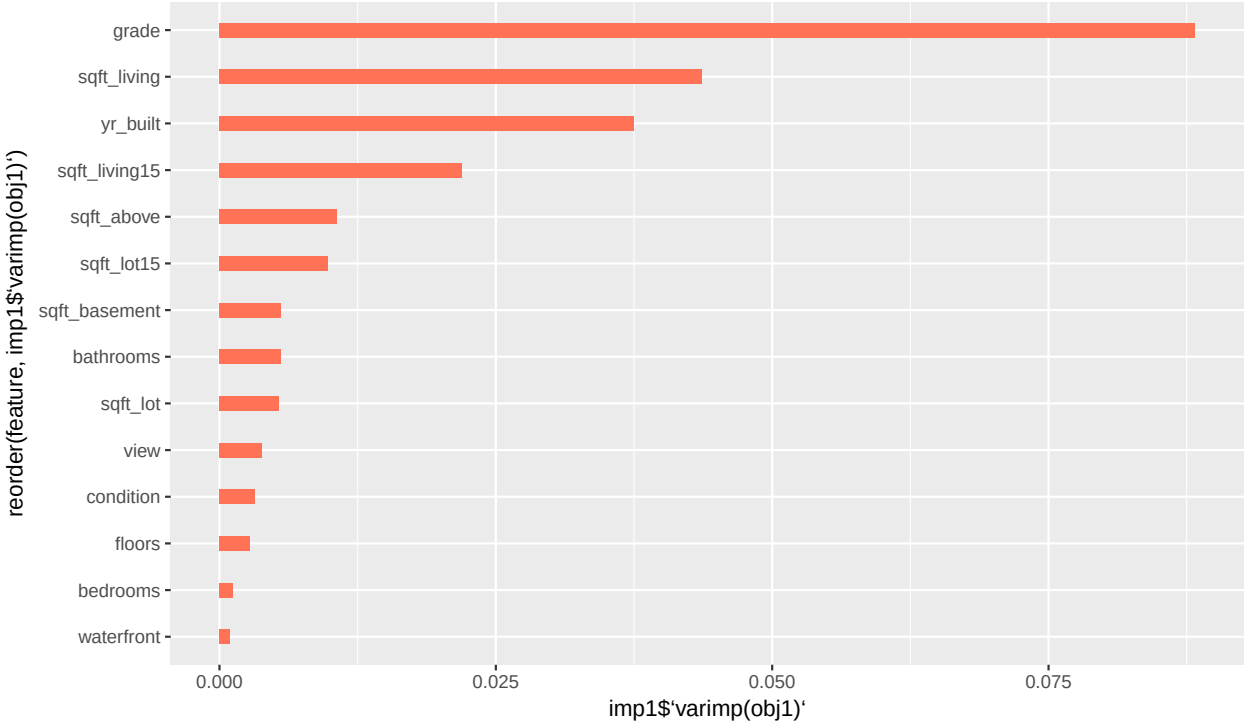
The permutation importance, on the other hand, is a reliable measure of variable importance for uncorrelated predictors when subsampling without replacement - instead of bootstrap sampling - and unbiased trees are used in the construction of the forest (Strobl et al., 2007b). Such an unbiased tree algorithm is available in the **party** package. It allows us to make reliable prediction and interpretability, especially for assessing the importance of each predictor in the complex ensemble of trees.

We use the **cForest** function in party package. Similar to the importance() function in **randomForest** package, a permutation importance is available via varimp() in **party**.

In this case, the first 5 most important features picked up by permutation importance is the same as previous, despite of the order.

```
##     imp1.feature imp1..varimp.obj1..
## 1          grade        0.0882733883
## 2    sqft_living        0.0436380533
## 3       yr_built        0.0375251438
## 4  sqft_living15        0.0219818477
## 5     sqft_above        0.0106528384
## 6     sqft_lot15        0.0098397328
## 7  sqft_basement        0.0055752685
## 8      bathrooms        0.0055181452
## 9       sqft_lot        0.0054145671
## 10          view        0.0038533952
## 11     condition        0.0032458963
## 12        floors        0.0027411518
## 13      bedrooms        0.0011835246
## 14    waterfront        0.0009325072
```

The feature importance plot is as follows.



### 5.2 Prediction

We make the predictions on test data based on this two random forest models. The RMSE are 0.2771, 0.2858, respectively. Both of them are smaller than any of the RMSE for previous models.

```
## [1] 0.2770 0.2858
```

## 6 Comparison

The summary information table of each procedures have been listed below. For XGBoost and Random forest models, they give almost same important features. Note that all the computataion are based on the training set and test set we've set at the very beginning.

We notice that those so-called "black box" models perform better in prediction. Black-box is a term used to identify certain predictive modeling techniques that are not capable of explaining their reasoning. We could justify their strong prediction power in our project.

| Approach | Number of variables | Root mean square error |
|---|---|---|
| Full model | 14 | 0.3043 |
| Reduced model | 8 | 0.3084 |
| XGBoost | | 0.3005 |
| Random forest (Gini importance) | | 0.2771 |
| Random forest (permutation importance) | | 0.2858 |

## 7 Discussion

Considering the accuracy of prediction, the data used for making predictions should have similar distribution as the data used for training. Since it is expected that the data distributions may move over time, deploying a model would be a continuous process. It is usually good to continuously monitor the incoming data and retrain models on new data if new incoming data is deviating vastly from the original data distribution. If it is not easy for monitoring data to detect a change in the data distribution, then a simpler strategy is to train the model periodically, for example, daily, weekly, or monthly. We will choose to retrain our model monthly, since the price of the house does not change frequently.

The data product should be used for predicting house price in the area of King County, since the data used in training contains only house sale prices for King County. And the prediction error will be larger for houses that are not similar to the houses in the dataset. There would be no lawsuit regarding of our data product, since the house sale data we got from Kaggle is publicly available for use. And it is released under CC0:

Public Domain License which means that we can copy, modify, distribute and perform on the data, even for commercial purposes, all without asking permission.

There is still some improvement to be made. For example, try some transformations to reduce the problems in diagnostics, create more detailed, dynamic plots, or build some ensemble models.

**The dashboard could be found at:** https://yiminggao.shinyapps.io/STAT425_YimingGao/

# 8 Work Cited


Mears, Teresa. "7 Online Tools to Help You Estimate Your Home???s Value." U.S. News, Web. Dec. 15, 2016.

Lucia Mutikani. "U.S. existing home sales decline, prices still strong." Reuters, Web. Dec.15, 2016.

Carolin Strobl, Torsten Hothorn, Achim Zelieis. "Party on! A New, Conditional Variable Importance Measure for Random Forests Available in the party Package." Department of Statistics, University of Munich. Dec.15, 2016.

Some plot functions from: https://www.kaggle.com/notaapple/house-prices-advanced-regression-techniques/detailed-exploratory-data-analysis-using-r