EN.570.616.01.SP22 Data Analytics in Environmental Health and Engineering

Hansen Guo, Xinyu Yang, Hangrui Liu

5/10/2022

Group Project

Chicago Home Price Forecast: The Influence of a House's Characteristics on Its Sale Price

1. Executive Summary

Housing has become one of the major challenges facing the world's metropolises. In addition to the impact of macroeconomic changes, the characteristics of the housing stock itself constitute a clear driver of price changes (Bailey, Muth & Nourse, 1963). Variations in the quality of different homes and different characteristics make it difficult to estimate the price of a real property, so real estate appraisal reviews have played an increasingly important role in the real estate industry (Pavlov, 2000). There is a strong and continuing demand for techniques to determine the accuracy of appraisal reports from lenders, institutional investors, courts, and others who make decisions based on the veracity of appraisal reports (Benjamin, Guttery & Sirmans, 2004; Isakson, 1998). The value of multiple regression analysis and machine learning techniques in this area has been documented as a stand-alone technique to check the accuracy of appraisal reports (Isakson, 2001; Mak, Choy & Ho, 2010).

Therefore, we selected a proprietary collection of real estate data provided by the Chicago Cook County Assessor and other private and public organizations for analytical modeling to examine home price issues. The dataset includes homes sold in three areas of Cook County (North Township, City of Chicago, and Southwest Township) for tax years 2003-2018. We will examine the impact of different variables on property values in Chicago. This allows us to dig deeper into the variables and provide a model that can more accurately estimate home values. In this way, individuals and professional organizations can better price homes based on available information.

We use 20 explanatory variables that include almost all aspects of Cook County homes. Statistical regression models and machine learning regression models are applied and further compared based on their performance to better estimate the final price of each home. This projectBy applying various techniques from data analysis (such as linear regression and multiple linear regression) to its real residential property prices helps us understand the changes over years and potentially forecast future prices. Secondly, based on the house location we can do clustering (unsupervised learning) about Walkfac (Car-dependent, Walkable, Somewhat walkable), and we also check if there is a correlation between Walkscore and Moreover, we also want to know if the price per foot is related to the house located in Chicago. This project employs methods including correlation matrix, types of statistical tests, and OLS regression to analyze the random sample of homes that have sold in the three regions of Cook County (Northern Townships, City of Chicago, Southwest Townships) during the tax years 2003-2018 and to predict sale price to analyze the data and build an This project has big and detailed data with the data of the tax years 2003-2018.

Based on the data analysis results, we can see all the machine learning methods, the most important factor is location, then is building square feet, it quite makes sense. And then we can see the facility is second most important to the house price which including central air and fireplace, age, property class. At the same time, we have established a high-accuracy prediction model, and made high-accuracy predictions on housing prices based on 14 variables. Finally, based on unsupervised learning, we cluster the houses according to latitude and longitude and summarize the features of different categories.

This project has big and detailed data with high accuracy models, and we believe it provided us with very insightful results about the Chicago house market . These models estimate the implied price of each feature in the price distribution, so it can better explain real-world phenomena and provide a more comprehensive understanding of the relationship between housing characteristics and prices.

2. Project Description

2.1 Describe the data set for the course project

2.1.1 Introduction

The data set is a collection of proprietary data sourced from the Cook County Assessor and other private and public organizations which includes homes that have sold in the three regions of Cook County (Northern Townships, City of Chicago, Southwest Townships) during tax years 2003-2018. We will study the effects of different variables on property values in Chicago.

2.1.2 Data Dictionary

1. Property Address: Property street address, not the address of the taxpayer.
2. Tax Year: Tax year of sale. Tax year refers to the year in which taxes due. Taxes are billed a year in arrears. If a sale occurred in calendar year 2018, it will be in tax year 2019.
3. Property Class: Represents different housing classes. Details can be found via: https://www.cookcountyassessor.com/assets/forms/classcode.pdf
4. Type of Residence: Type of residence - 1 = one story, 2 = two-story, 3 = three-story or higher, 4 = split level, 5 = 1.5 story, 6 = 1.6 story, 7 = 1.7 story , 8 = 1.8 story , 9 = 1.9 story (Note: residences with 1.5 - 1.9 stories are one story and have partial livable attics and are classified based on the square footage of the attic compared to the first floor of the house. So, 1.5 story houses have an attic that is 50% of the area of the first floor, 1.6 story houses are 60%, 1.7 are 70%, etc. However, what is recorded on the field card differs from what is in the database. All 1.5 - 1.9 story houses are coded as 5)
5. Rooms: Number of rooms in the property (excluding baths). Not to be confused with bedrooms.
6. Bedrooms: Number of bedrooms in the property, defined based on building square foot and the judgment of the person in the field.
7. Basement: Basement type - 0 = None, 1 = Full, 2 = Slab, 3 = Partial, 4 = Crawl
8. Fireplaces: Number of fireplaces, counted as the number of flues one can see from the outside of the building.
9. Central Air Is central airconditioning present?: - 1 = yes, 2 = no
10. Full Baths: Number of full bathrooms, defined as having a bath or shower. If this value is missing, the default value is set to 1.
11. Half Baths: Number of half baths, defined as bathrooms without a shower or bathtub.
12. Building Square Feet: Building square feet, as measured from the exterior of the property
13. Land Square: Feet Square feet of the land (not just the building) of the property. Note that land is divided into 'plots' and 'parcels' - this field applies to parcels, identified by PIN
14. Age: Age of the property. If missing, this defaults to 10. This field is a combination of original age and effective age where original age refers to the oldest component of the building and effective age is a relative judgement due to renovations or other improvements. For instance, if a property is completely demolished and built up again, the age resets to 1. But if portions of the original structure are kept, it may be more complicated to determine the age.
15. Longitude: Longitude coordinate of the property's location, as defined by the centroid of the parcel shape in GIS.
16. Latitude: Latitude coordinate of the property's location, as defined by the centroid of the parcel shape in GIS.
17. Sale Price: Sale price

2.1.3 Data source and links:

The data can be acquired from Cook County Government website via https://datacatalog.cookcountyil.gov/.

2.2 Idea and explanation

Based on the dataset we have already found, it includes a variety kind of parameters so we can use this to do several kinds of OLS analysis, prediction or other kinds of analysis. We used OLS to analyze the dataset. Besides, asset marketing is becoming more and more important in modern days. We can predict the developing tendency based on type of residence, buyers' age and so on. Moreover, we may also use this to

extrapolate population mobility on some level. In this dataset, we also have the coordinates of the residence, we can use this to see the cluster characteristics. The most important aspect we think is the sale price of a house. This point is one of the most important factors for both buyers and real estate dealers. Above all, we can do a lot of things according to this dataset. After cleaning the dataset, we first performed an exploratory analysis of the dataset and constructed more complex machine learning models based on the results.

2.3 Data clean

1. Subset the data As the single family homes as part of the process of building an accurate valuation model. Using Property_Class (202 - 210), subset the data to just properties whose class the Assessor has coded as "single family homes" of any age or size.
2. Bath The Half_Baths column contains the number of half bathrooms. For the purposes of my analysis, I want to code a half bathroom as a fraction of a full bathroom. we should multiply all the values in the column by .5 and reassign them to that column.

Drop all rows that have any missing values in the Half Baths and Full Baths columns. Create a new column, Total_Baths, where Total_Baths = Full_Baths + Half_Baths*.5 Next, drop the Full_Baths and Half_Baths columns Create a new column, Log_Sale_Price, where Log_Sale_Price = log(Sale_Price)

3.Data format Set its index to Tax_Year; make sure the old numerical index doesn't wind up as a new column Set the datatype for the Central_Air column to "category".

2.4 Exploratory Data Analysis

The original dataset size is 7691 rows × 20 columns. The variables of interest are [Rooms, Bedrooms, FirePlaces, Central_Air, Building_Square_Feet, Land_Square_Feet, Age, Longitude, Latitude, Walkscore, Total_Baths].We did some EDA and visualize some representative variables below to get an overview of our data and to detect some important matrices and trends that might help with our further analysis and model building.

```
rawdata <- read.csv(file="clean_data.csv",header=TRUE,sep=",")
rawdata <- na.omit(rawdata)
dim(rawdata)
```

```
## [1] 7691   20
```

```
summary(rawdata)
```

```
##       PIN            Property_Address   Property_Class   Census_Tract
##  Min.   :1.011e+12   Length:7691        Min.   :202.0   Min.   : 10201
##  1st Qu.:1.032e+13   Class :character   1st Qu.:203.0   1st Qu.:681300
##  Median :1.525e+13   Mode  :character   Median :203.0   Median :805802
##  Mean   :1.635e+13                      Mean   :203.5   Mean   :691258
##  3rd Qu.:2.034e+13                      3rd Qu.:203.0   3rd Qu.:818800
##  Max.   :3.331e+13                      Max.   :210.0   Max.   :843800
##  Type_of_Residence      Rooms           Bedrooms       Basement
##  Length:7691        Min.   : 0.000   Min.   :0.000   Length:7691
##  Class :character   1st Qu.: 5.000   1st Qu.:3.000   Class :character
##  Mode  :character   Median : 6.000   Median :3.000   Mode  :character
##                     Mean   : 5.807   Mean   :3.015
##                     3rd Qu.: 6.000   3rd Qu.:3.000
##                     Max.   :17.000   Max.   :9.000
##    Fireplaces      Central_Air     Building_Square_Feet Land_Square_Feet
##  Min.   :0.0000   Min.   :0.0000   Min.   :   0         Min.   :  1053
##  1st Qu.:0.0000   1st Qu.:0.0000   1st Qu.:1035         1st Qu.:  3780
##  Median :0.0000   Median :1.0000   Median :1225         Median :  5580
```

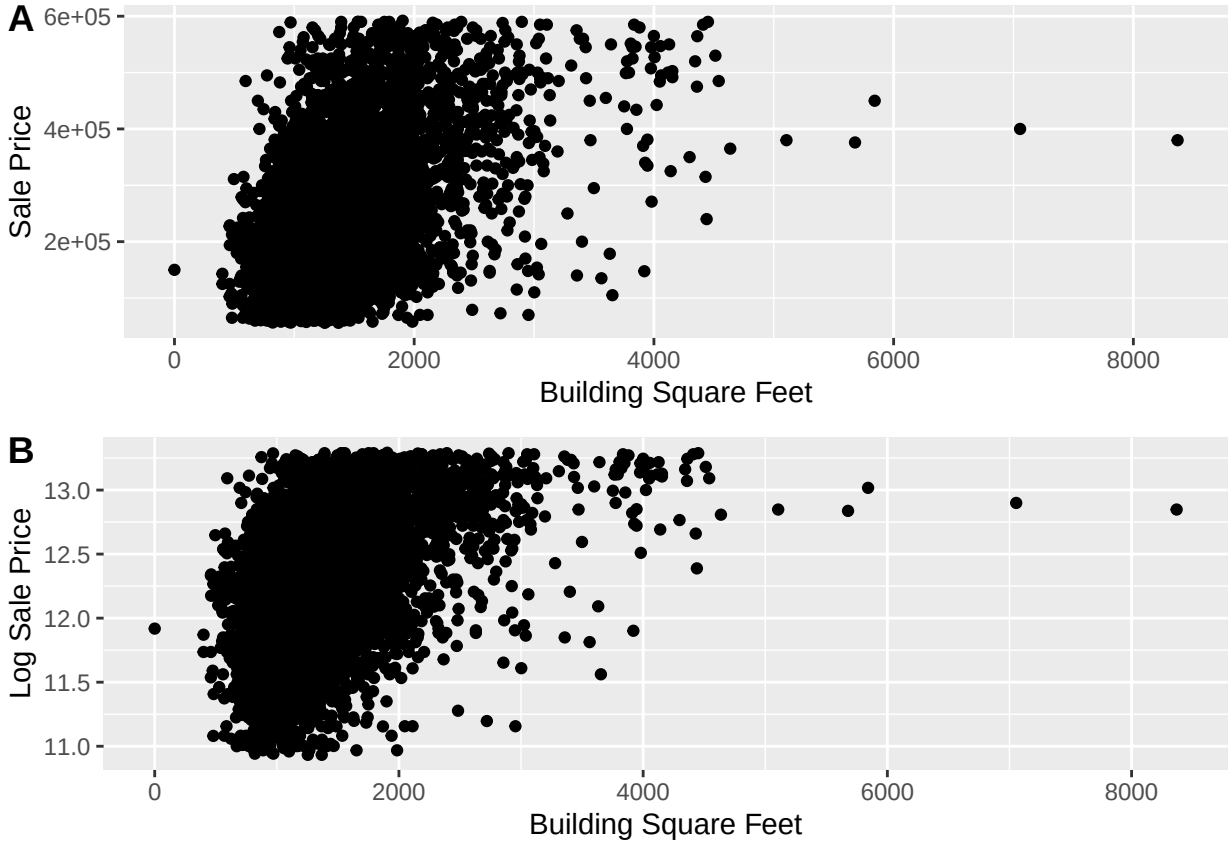
```
##  Mean   :0.2284   Mean   :0.5427   Mean   :1347         Mean   :  6980
##  3rd Qu.:0.0000   3rd Qu.:1.0000   3rd Qu.:1547         3rd Qu.:  7850
##  Max.   :3.0000   Max.   :1.0000   Max.   :8369         Max.   :450890
##       Age           Longitude         Latitude       Walkscore
##  Min.   :  1.00   Min.   :-88.25   Min.   :41.47   Min.   : 0.00
##  1st Qu.: 50.00   1st Qu.:-87.86   1st Qu.:41.74   1st Qu.:44.00
##  Median : 59.00   Median :-87.78   Median :41.88   Median :59.00
##  Mean   : 63.59   Mean   :-87.80   Mean   :41.86   Mean   :56.82
##  3rd Qu.: 80.00   3rd Qu.:-87.70   3rd Qu.:42.00   3rd Qu.:72.00
##  Max.   :163.00   Max.   :-87.53   Max.   :42.16   Max.   :99.00
##    Walkfac            Sale_Price      Total_Baths    Log_Sale_Price
##  Length:7691        Min.   : 56000   Min.   :0.000   Min.   :10.93
##  Class :character   1st Qu.:154000   1st Qu.:1.000   1st Qu.:11.94
##  Mode  :character   Median :217000   Median :1.000   Median :12.29
##                     Mean   :235520   Mean   :1.457   Mean   :12.26
##                     3rd Qu.:292000   3rd Qu.:2.000   3rd Qu.:12.58
##                     Max.   :592000   Max.   :5.500   Max.   :13.29
```

```
anyNA(rawdata)
```

```
## [1] FALSE
```

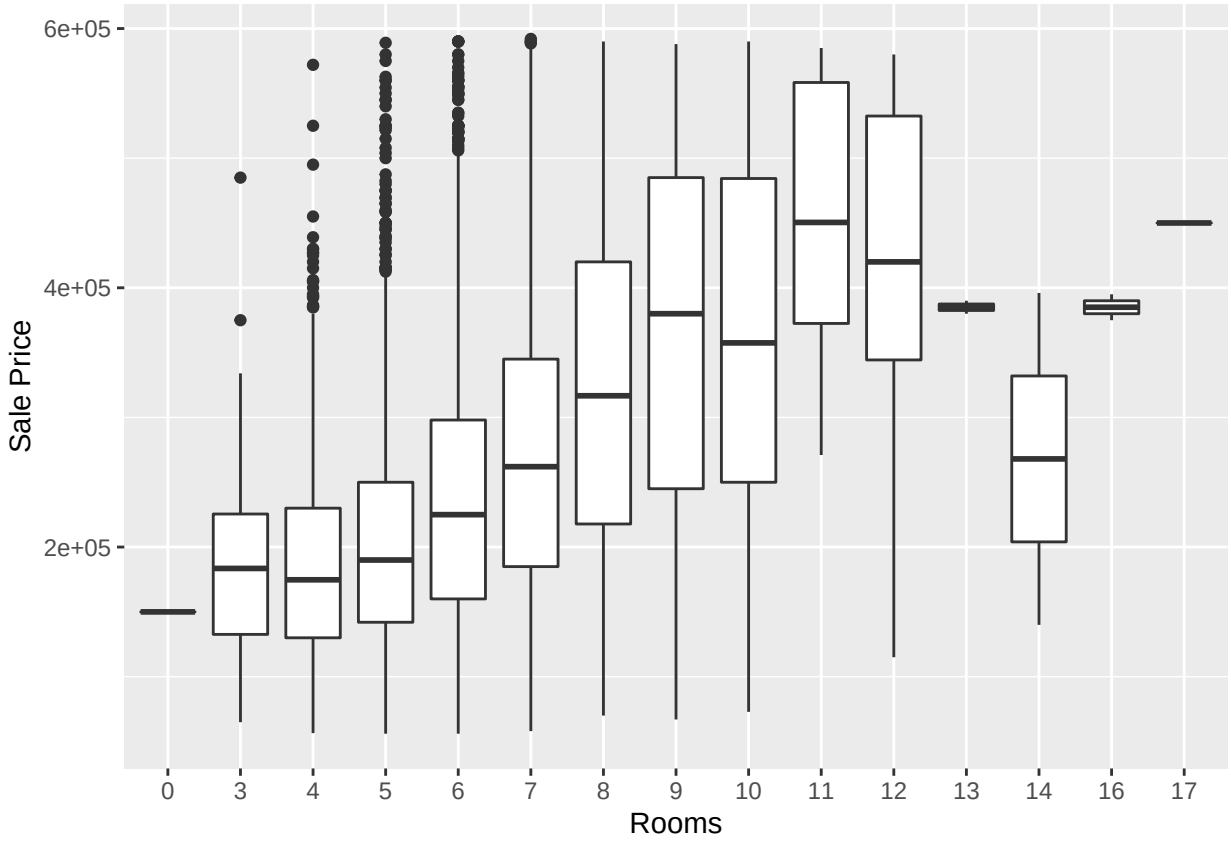
2.4.1 Area vs Price

Generally speaking, as the size of the house increases, the price of the house increases. For the establishment of subsequent models, we have added a Log_Sale_Price column ($Log\_Sale\_Price = Log(Sale\_Price)$), which may improve the accuracy of subsequent models. Although the proportional relationships exhibited by Sale Price and Log(Sale Price) are not identical, they show similar trends.



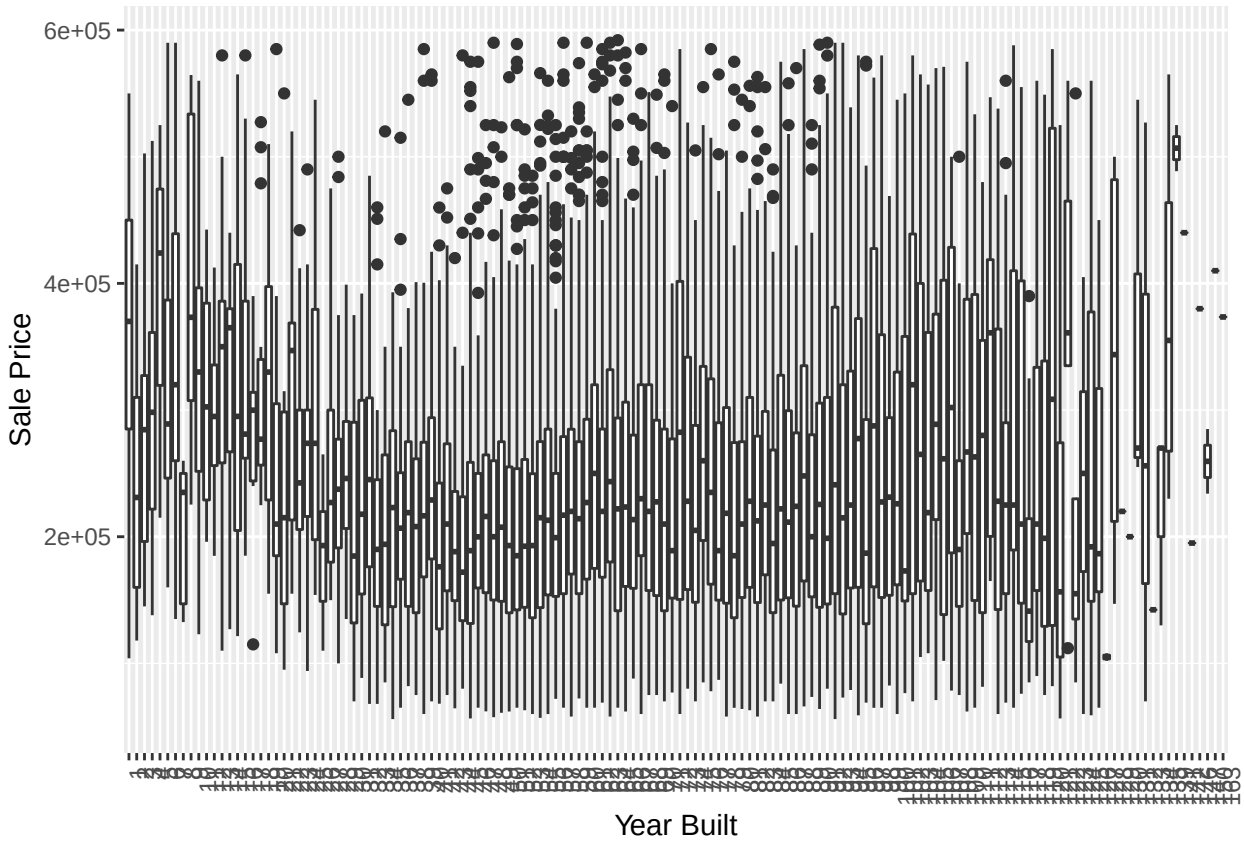
2.4.2 Rooms vs Price

The number of rooms may be positively correlated with area, which in turn is positively correlated with Sale_Price. Plotting the results according to the graph below, this is indeed the case. However, the positive correlation becomes weak or even disappears when the number of rooms exceeds 12, so items with more than 12 rooms can be regarded as an extreme case, and in the follow-up analysis, it can be decided whether to eliminate it according to the performance of the model.



2.4.3 Age vs Price

Generally speaking, the newer the house, the higher the price. But the correlation is not very strong.



2.4 Statistical learning techniques

Data analysis was performed using R models to compare between and complete predictions. These models are: (1) supervised data with an emphasis on inference (generalized linear models (e.g. linear regression, logistic regression, and polynomial regression); (2) supervised data with an emphasis on prediction (linear classifiers, decision trees, random forest, neural networks, non-linear SVMs); (3) unsupervised data (Clustering - K-means).

1. Supervised data with an emphasis on inference

To analyze the data, we apply logistic regression, which is an appropriate regression analysis for elucidating the link between a binary or dichotomous dependent variable and one or more independent variables. This approach requires few computer resources and scaled input characteristics, and it is simple to comprehend and normalize.

2. Supervised data with an emphasis on prediction

- Decision tree: When evaluating the classification tree outcomes, the decision tree is used to anticipate qualitative responses and corresponds to the most common class of training data in the region. This method enables us to establish a framework for quantifying the worth of the results and the likelihood of reaching them. It also provides for a more accurate representation of the projected outcomes.
- Random forests: In training data and output classification, random forests build a large number of decision trees. It outperforms bagged trees as an ensemble learning approach for classification in that its variance is minimized when we average the trees by combining the predictors of the trees such that each tree depends on the value of a random vector that is sampled individually. This model analyzes only a subset of the characteristics in each split of the tree rather than all of them. It outperforms decision trees on high-dimensional data and learns the data faster. The variance is averaged out when we average all of the decision trees in the random forest, resulting in a minimal bias.

- Neural networks: We utilized neural networks to generalize linear models that go through multi-stage processing to make judgments. These models are inspired by the structure of biological neural networks in the human brain and are meant to mimic how humans examine data. This method enables for the extraction of information from enormous volumes of data and the development of sophisticated models.

The cross-validation approach is used to evaluate the model's prediction performance by using one-third of the data subset as a test set and two-thirds as a training set. With 7691 rows, the dataset has a big enough test set to make predictions. The training error rate may underestimate the test error rate if cross-validation is not used.

3. Unsupervised data k-means clustering is the algorithm for clustering.Iteratively improving the μ_i prototypes of k clusters. It first pick k random objects as the initial μ_i prototypes, then find for each object the closest prototype and assign to that cluster. After that, we calculate the averages for each cluster to get new μ_i, and repeat until convergence.

2.4.1 Supervised data with an emphasis on prediction result statistics and evaluation criteria

We use the construction of confusion matrices to calculate the test accuracy (all correct predictions divided by the total number of test sets) as a result statistic to evaluate the accuracy of the model, which in turn is used for model performance evaluation. The test accuracy is simply calculated by (1 - test error rate). The higher the test accuracy achieved by a model, the more suitable it is for the project. With a total of 1 (or 2) targets to predict for this project, we may encounter situations where different models represent different optimal test accuracies. In order to make the evaluation process more objective and efficient, we set the evaluation rules in advance. First, the best-fitting model we are looking for must have the maximum number of best predictions among all models. Because we try to eliminate the unintended effects of extreme results, we choose models that give accurate predictions in most cases. If more than one model meets this requirement, we will choose the model with the greatest average test accuracy, which indicates better overall performance.

2.5 Module Build-up

2.5.1 Supervised data with an emphasis on inference

```
head(rawdata)
```

```
##           PIN               Property_Address Property_Class Census_Tract
## 1 8.12417e+12 613 S SCHOOL ST MOUNT PROSPECT            203       804901
## 2 5.31305e+12     919 E GLENWOOD RD GLENVIEW            204       801400
## 3 1.33231e+13            1747 N MERRIMAC AVE            203       250400
## 4 1.22322e+13                  3758 N PARIS             202       170500
## 5 1.53412e+13                 9542 W MONROE             203       818800
## 6 1.53432e+13              3717 CLEVELAND AV            202       818800
##   Type_of_Residence Rooms Bedrooms Basement Fireplaces Central_Air
## 1                 1     6        4     Full          1           1
## 2                 1     7        3     Full          1           1
## 3                 ?     6        3     Full          0           0
## 4           1-story     5        2     Full          0           0
## 5           1-story     5        3  Partial          0           0
## 6                 ?     5        3     Full          0           1
##   Building_Square_Feet Land_Square_Feet Age Longitude Latitude Walkscore
## 1                 1461             8712  60 -87.93069 42.05418        49
## 2                 1957            20500  67 -87.76312 42.07116        18
## 3                 1520             3937  90 -87.78104 41.91233        73
## 4                  904             3720  57 -87.83071 41.94842        47
## 5                 1033             6250  58 -87.86098 41.82934        55
## 6                 1223             7500  62 -87.85841 41.82309        50
##            Walkfac Sale_Price Total_Baths Log_Sale_Price
```

```
## 1     Car-Dependent     298000         1.5       12.60485
## 2     Car-Dependent     425000         1.5       12.95984
## 3     Very Walkable     285000         1.5       12.56024
## 4     Car-Dependent     226000         1.0       12.32829
## 5 Somewhat Walkable     178000         1.0       12.08954
## 6     Car-Dependent     212500         1.5       12.26670
```

```
phenotype <- 'Sale_Price'
predictors <- c(
"Property_Class","Rooms",   "Bedrooms", "Basement", "Fireplaces",   "Central_Air",  "Building_Square_Fe
)
```

Here, we treat the sale price as a response, and we choose basic charateritics of the house as predictors, such as longtitude, latitude, and building square feet.

```
response <- rawdata[,phenotype]
active_set <- rawdata[,predictors]
```

Linear regression of sale price

```
full <- lm(response ~ ., data = data.frame(active_set))
summary(full)
```

```
##
## Call:
## lm(formula = response ~ ., data = data.frame(active_set))
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -512194  -53694   -7620   45393  398299
##
## Coefficients:
##                            Estimate Std. Error t value Pr(>|t|)
## (Intercept)              -1.657e+06  7.351e+05  -2.255 0.024192 *
## Property_Class            1.587e+03  7.308e+02   2.172 0.029919 *
## Rooms                     1.843e+03  1.305e+03   1.412 0.157950
## Bedrooms                 -5.470e+02  2.033e+03  -0.269 0.787878
## BasementPartial          -1.839e+04  2.116e+03  -8.693  < 2e-16 ***
## Fireplaces                3.047e+04  2.329e+03  13.080  < 2e-16 ***
## Central_Air              -1.137e+04  2.384e+03  -4.769 1.88e-06 ***
## Building_Square_Feet      6.605e+01  3.493e+00  18.909  < 2e-16 ***
## Land_Square_Feet          7.222e-01  1.078e-01   6.701 2.22e-11 ***
## Age                      -9.531e+01  5.066e+01  -1.881 0.059942 .
## Longitude                 1.420e+05  1.026e+04  13.836  < 2e-16 ***
## Latitude                  3.325e+05  8.918e+03  37.281  < 2e-16 ***
## Walkscore                 1.110e+01  1.265e+02   0.088 0.930047
## WalkfacSomewhat Walkable  1.584e+03  3.838e+03   0.413 0.679820
## WalkfacVery Walkable      5.095e+03  5.654e+03   0.901 0.367491
## WalkfacWalker's Paradise  3.454e+04  9.741e+03   3.546 0.000394 ***
## Total_Baths               1.544e+04  2.367e+03   6.520 7.49e-11 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 83550 on 7674 degrees of freedom
## Multiple R-squared:  0.4232, Adjusted R-squared:  0.422
```

```
## F-statistic: 351.9 on 16 and 7674 DF,  p-value: < 2.2e-16
```

We can see BasementPartial, Fireplaces, Central_Air , Building_Square_Feet, Land_Square_Feet, Longitude, Latitude , WalkfacWalker's Paradise, Total_Baths is important factor to the sales price. Especially the location contribute to the sales price more, which is longitude and latitude, and then fireplaces and central air contribute to the sale price.

Then we delete bedrooms, to just check the model

```
full <- lm(response ~ . -Bedrooms   , data = data.frame(active_set))
summary(full)
```

```
##
## Call:
## lm(formula = response ~ . - Bedrooms, data = data.frame(active_set))
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -510890  -53779   -7510   45403  398659
##
## Coefficients:
##                            Estimate Std. Error t value Pr(>|t|)
## (Intercept)              -1.665e+06  7.344e+05  -2.268 0.023369 *
## Property_Class            1.592e+03  7.305e+02   2.179 0.029380 *
## Rooms                     1.668e+03  1.131e+03   1.475 0.140347
## BasementPartial          -1.840e+04  2.115e+03  -8.702  < 2e-16 ***
## Fireplaces                3.048e+04  2.328e+03  13.094  < 2e-16 ***
## Central_Air              -1.136e+04  2.383e+03  -4.767 1.91e-06 ***
## Building_Square_Feet      6.594e+01  3.467e+00  19.016  < 2e-16 ***
## Land_Square_Feet          7.224e-01  1.078e-01   6.704 2.18e-11 ***
## Age                      -9.539e+01  5.065e+01  -1.883 0.059716 .
## Longitude                 1.419e+05  1.026e+04  13.835  < 2e-16 ***
## Latitude                  3.325e+05  8.918e+03  37.282  < 2e-16 ***
## Walkscore                 1.057e+01  1.265e+02   0.084 0.933372
## WalkfacSomewhat Walkable  1.587e+03  3.837e+03   0.414 0.679219
## WalkfacVery Walkable      5.132e+03  5.652e+03   0.908 0.363865
## WalkfacWalker's Paradise  3.462e+04  9.735e+03   3.556 0.000378 ***
## Total_Baths               1.534e+04  2.343e+03   6.549 6.15e-11 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 83550 on 7675 degrees of freedom
## Multiple R-squared:  0.4232, Adjusted R-squared:  0.4221
## F-statistic: 375.4 on 15 and 7675 DF,  p-value: < 2.2e-16
```

choose numerical data to check the correlation

```
predictors_num <- c(
"Rooms",    "Bedrooms", "Fireplaces",   "Building_Square_Feet", "Land_Square_Feet","Age","Longitude","La
)
```

Check correlations

```
cor(rawdata[,predictors_num])
```

```
##                             Rooms    Bedrooms Fireplaces Building_Square_Feet
## Rooms                1.000000000  0.74442371  0.3488546            0.7396301
```

```
## Bedrooms              0.744423713  1.00000000  0.2878296            0.6334857
## Fireplaces            0.348854593  0.28782958  1.0000000            0.4599042
## Building_Square_Feet  0.739630100  0.63348567  0.4599042            1.0000000
## Land_Square_Feet      0.159294246  0.12823394  0.2267576            0.2217493
## Age                   0.007449176 -0.02258406 -0.1116230           -0.1168336
## Longitude            -0.114904540 -0.07670092 -0.1393627           -0.1446695
## Latitude              0.131183186  0.09103367  0.1154965            0.1463606
## Total_Baths           0.609607122  0.55935532  0.3924278            0.6767136
##                      Land_Square_Feet          Age   Longitude    Latitude
## Rooms                      0.15929425  0.007449176 -0.11490454  0.13118319
## Bedrooms                   0.12823394 -0.022584058 -0.07670092  0.09103367
## Fireplaces                 0.22675760 -0.111622974 -0.13936266  0.11549650
## Building_Square_Feet       0.22174929 -0.116833616 -0.14466951  0.14636064
## Land_Square_Feet           1.00000000 -0.130518370 -0.23429394  0.06065383
## Age                       -0.13051837  1.000000000  0.25544582  0.11454164
## Longitude                 -0.23429394  0.255445816  1.00000000 -0.59412695
## Latitude                   0.06065383  0.114541643 -0.59412695  1.00000000
## Total_Baths                0.16660488 -0.150704263 -0.16215891  0.13682259
##                      Total_Baths
## Rooms                  0.6096071
## Bedrooms               0.5593553
## Fireplaces             0.3924278
## Building_Square_Feet   0.6767136
## Land_Square_Feet       0.1666049
## Age                   -0.1507043
## Longitude             -0.1621589
## Latitude               0.1368226
## Total_Baths            1.0000000
```

we can see there is a strong correlation between rooms and bedrooms and Building_Square_Feet, and we can also see the rooms and bedroos is related to the total baths.

```
predictors <- c(
"Property_Class","Rooms",   "Bedrooms", "Basement", "Fireplaces",   "Central_Air",  "Building_Square_Fe
)
```

Here, we preprocess data for For logistic regression about central air. To determine if we can from predictors to see whether we have centrail air.

```
active_set_2 <- rawdata[,predictors]
```

Perform logistic regression about central air

```
glm.fit=glm(as.factor(rawdata$Central_Air)~.,
data=data.frame(active_set_2),family=binomial)

summary(glm.fit)
```

```
##
## Call:
## glm(formula = as.factor(rawdata$Central_Air) ~ ., family = binomial,
##     data = data.frame(active_set_2))
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -3.0520  -0.7763   0.1498   0.6857   3.0771
```

```
##
## Coefficients:
##                             Estimate Std. Error z value Pr(>|z|)
## (Intercept)               -3.817e+02  2.708e+01 -14.095  < 2e-16 ***
## Property_Class            -2.225e-02  2.313e-02  -0.962 0.336001
## Rooms                      1.973e-01  4.082e-02   4.834 1.34e-06 ***
## Bedrooms                  -2.041e-01  6.433e-02  -3.173 0.001509 **
## BasementPartial           -3.958e-01  6.872e-02  -5.760 8.42e-09 ***
## Fireplaces                 6.882e-01  8.168e-02   8.426  < 2e-16 ***
## Building_Square_Feet       4.705e-04  1.317e-04   3.572 0.000354 ***
## Land_Square_Feet           2.795e-05  8.558e-06   3.266 0.001091 **
## Age                       -3.701e-02  1.701e-03 -21.763  < 2e-16 ***
## Longitude                 -3.319e-01  3.753e-01  -0.884 0.376535
## Latitude                   8.591e+00  3.269e-01  26.279  < 2e-16 ***
## Walkscore                 -2.578e-02  4.341e-03  -5.938 2.89e-09 ***
## WalkfacSomewhat Walkable   7.743e-02  1.229e-01   0.630 0.528871
## WalkfacVery Walkable      -1.376e-01  1.829e-01  -0.752 0.452044
## WalkfacWalker's Paradise   7.944e-03  3.190e-01   0.025 0.980132
## Total_Baths                6.876e-01  7.701e-02   8.929  < 2e-16 ***
## Sale_Price                -2.549e-06  3.640e-07  -7.003 2.50e-12 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 10605.8  on 7690  degrees of freedom
## Residual deviance:  7047.6  on 7674  degrees of freedom
## AIC: 7081.6
##
## Number of Fisher Scoring iterations: 5
```

From the data, we can see many factors is related to the centrail air, we can see the most important factor is sale price and the building square feet. To our surprise, we can see the location is almost nothing to do with the central air, but we know location is very important to the sale price.

Perform logistic regression about Basement

```
glm.fit=glm(as.factor(rawdata$Basement)~.,
data=data.frame(active_set_2),family=binomial)

summary(glm.fit)
```

```
##
## Call:
## glm(formula = as.factor(rawdata$Basement) ~ ., family = binomial,
##     data = data.frame(active_set_2))
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -3.8009  -0.8715  -0.6681   1.1052   2.5891
##
## Coefficients:
##                             Estimate Std. Error z value Pr(>|z|)
## (Intercept)               -2.406e+02  1.939e+01 -12.405  < 2e-16 ***
```

```
## Property_Class            3.778e-02  1.944e-02   1.944   0.0519 .
## Rooms                    -2.202e-01  3.668e-02  -6.003 1.94e-09 ***
## Bedrooms                  1.113e-01  5.548e-02   2.005   0.0449 *
## Fireplaces                6.746e-02  6.385e-02   1.057   0.2907
## Central_Air              -2.926e-01  6.440e-02  -4.544 5.52e-06 ***
## Building_Square_Feet      5.220e-04  1.001e-04   5.212 1.87e-07 ***
## Land_Square_Feet          2.384e-05  5.834e-06   4.086 4.39e-05 ***
## Age                      -7.934e-03  1.403e-03  -5.653 1.57e-08 ***
## Longitude                -2.694e+00  2.773e-01  -9.717  < 2e-16 ***
## Latitude                 -4.708e-02  2.598e-01  -0.181   0.8562
## Walkscore                -1.394e-02  3.315e-03  -4.204 2.62e-05 ***
## WalkfacSomewhat Walkable -1.978e-01  9.870e-02  -2.004   0.0450 *
## WalkfacVery Walkable     -2.382e-01  1.483e-01  -1.607   0.1081
## WalkfacWalker's Paradise  4.876e-01  2.688e-01   1.814   0.0696 .
## Total_Baths              -7.043e-03  6.466e-02  -0.109   0.9133
## Sale_Price               -2.954e-06  3.261e-07  -9.058  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 10031.4  on 7690  degrees of freedom
## Residual deviance:  9015.5  on 7674  degrees of freedom
## AIC: 9049.5
##
## Number of Fisher Scoring iterations: 5
```

Now we want to check, how to know if the house has basement, we can see the sale price and land square feet is strong related to the basement.It also make sense, we have big land square feet and higher price, we will have basement.

Multinomial Regression about Walkfac

Now we want to check the multinational regression, we can see there are four level "Car-Dependent", "Somewhat Walkable","Very Walkable", "Walker's Paradise".

```
rawdata$Walkfac <- factor(rawdata$Walkfac,levels=c("Car-Dependent", "Somewhat Walkable","Very Walkable"
head(rawdata)
```

```
##           PIN               Property_Address Property_Class Census_Tract
## 1 8.12417e+12 613 S SCHOOL ST MOUNT PROSPECT            203       804901
## 2 5.31305e+12     919 E GLENWOOD RD GLENVIEW            204       801400
## 3 1.33231e+13            1747 N MERRIMAC AVE            203       250400
## 4 1.22322e+13                  3758 N PARIS             202       170500
## 5 1.53412e+13                 9542 W MONROE             203       818800
## 6 1.53432e+13              3717 CLEVELAND AV            202       818800
##   Type_of_Residence Rooms Bedrooms Basement Fireplaces Central_Air
## 1                 1     6        4     Full          1           1
## 2                 1     7        3     Full          1           1
## 3                 ?     6        3     Full          0           0
## 4           1-story     5        2     Full          0           0
## 5           1-story     5        3  Partial          0           0
## 6                 ?     5        3     Full          0           1
##   Building_Square_Feet Land_Square_Feet Age Longitude Latitude Walkscore
## 1                 1461             8712  60 -87.93069 42.05418        49
## 2                 1957            20500  67 -87.76312 42.07116        18
```

```
## 3                1520             3937  90 -87.78104 41.91233        73
## 4                 904             3720  57 -87.83071 41.94842        47
## 5                1033             6250  58 -87.86098 41.82934        55
## 6                1223             7500  62 -87.85841 41.82309        50
##   Walkfac Sale_Price Total_Baths Log_Sale_Price
## 1       1     298000         1.5       12.60485
## 2       1     425000         1.5       12.95984
## 3       3     285000         1.5       12.56024
## 4       1     226000         1.0       12.32829
## 5       2     178000         1.0       12.08954
## 6       1     212500         1.5       12.26670
```

```
#pom = polr(as.factor(rawdata$Walkfac) ~ Longitude + Latitude + Walkscore, data = rawdata, Hess=TRUE)
#summary(pom)
```

```
rawdata$Walkfac=as.factor(rawdata$Walkfac)
rawdata$Walkfac <- relevel(rawdata$Walkfac, ref = "1" )
```

```
mod.multinom <- multinom(as.factor(rawdata$Walkfac) ~ Longitude + Latitude+Walkscore  ,data=rawdata)
```

```
## # weights:  20 (12 variable)
## initial  value 10661.989931
## iter  10 value 3072.486845
## iter  20 value 458.270288
## iter  30 value 188.517666
## iter  40 value 174.324430
## iter  50 value 174.016628
## iter  60 value 173.954178
## iter  70 value 173.833139
## iter  80 value 173.656728
## iter  90 value 173.626156
## iter 100 value 173.484740
## final  value 173.484740
## stopped after 100 iterations
```

```
summary(mod.multinom)
```

```
## Call:
## multinom(formula = as.factor(rawdata$Walkfac) ~ Longitude + Latitude +
##     Walkscore, data = rawdata)
##
## Coefficients:
##   (Intercept) Longitude   Latitude Walkscore
## 2   -26.07952  5.615102   5.363429  5.896592
## 3   -59.34574 10.430400   3.288029 13.645791
## 4    24.77092  5.933737 -30.357739 24.014966
##
## Std. Errors:
##   (Intercept) Longitude  Latitude Walkscore
## 2 0.044072005 1.1782031 1.7576655 1.0914010
## 3 0.007719256 1.9217366 2.2396006 2.0798145
## 4 0.003842014 0.3733209 0.1258059 0.2161151
##
## Residual Deviance: 346.9695
## AIC: 370.9695
```

1.From the output above, the iteration number is 100, and converge fast to 173, can be used in comparisons of nested models
2.We can see one unit increase in the Longtitude is associated with 5 increase log odds for somewhat walkable vs car. We can see one unit increase in the walkscore is associated with 5.89 increase for car vs somewhat walkable.
3.With one unite in walkscore, is associated with 25 increase log odds to choose car vs walker paradise

```
Anova(mod.multinom)
```

```
## Analysis of Deviance Table (Type II tests)
##
## Response: as.factor(rawdata$Walkfac)
##           LR Chisq Df Pr(>Chisq)
## Longitude     23.5  3  3.217e-05 ***
## Latitude      12.5  3   0.005952 **
## Walkscore  15574.3  3  < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

We can see all the factors contribute to this model.

2.5.2 Supervised data with an emphasis on prediction

decision tree

```
predictors3 <- c(
"Property_Class","Rooms",   "Bedrooms", "Basement", "Fireplaces",   "Central_Air",  "Building_Square_Fe
)

active_set_3 <- rawdata[,predictors3]
```

Set a seed to divide the entire data set into test and training sets (1/3, 2/3).

```
attach(active_set_3)
set.seed(8)
cu.train = sample(1: nrow(active_set_3), 2*nrow(active_set_3)/3)
cu.test = active_set_3[-cu.train,]
CU.test = active_set_3[-cu.train]
tree.chicago=tree(Log_Sale_Price~., active_set_3, subset = cu.train)
```

```
## Warning in tree(Log_Sale_Price ~ ., active_set_3, subset = cu.train): NAs
## introduced by coercion
```

```
summary(tree.chicago)
```

```
##
## Regression tree:
## tree(formula = Log_Sale_Price ~ ., data = active_set_3, subset = cu.train)
## Variables actually used in tree construction:
## [1] "Latitude"             "Longitude"            "Building_Square_Feet"
## Number of terminal nodes:  11
## Residual mean deviance:  0.1266 = 647.7 / 5116
## Distribution of residuals:
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
## -1.65100 -0.21170  0.02424  0.00000  0.23630  1.17700
```

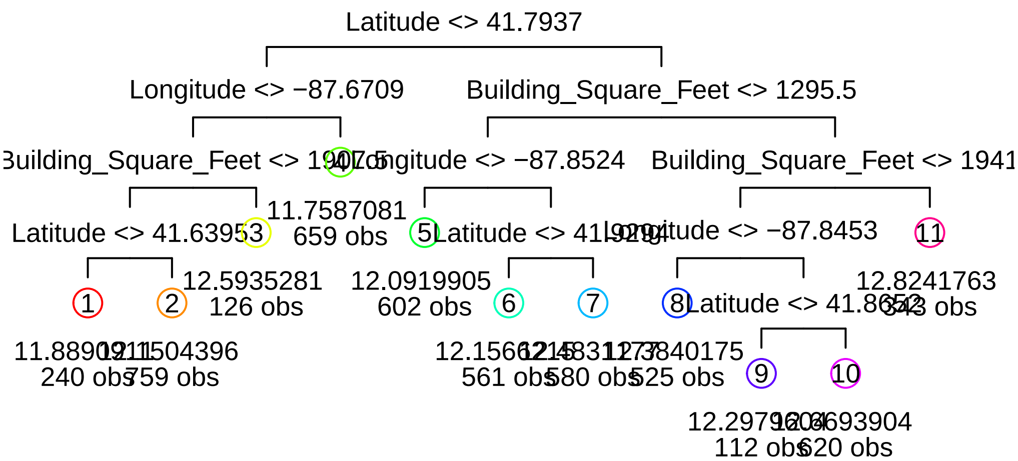
```
chicago.predict = predict(tree.chicago, cu.test)
```

```
## Warning in pred1.tree(object, tree.matrix(newdata)): NAs introduced by coercion
```

```r
mean((chicago.predict-cu.test$Log_Sale_Price)^2)
```

```
## [1] 0.1362816
```

```r
#plot(tree.chicago)
#text(tree.chicago,pretty=0)
draw.tree(tree.chicago, cex=0.8)
```

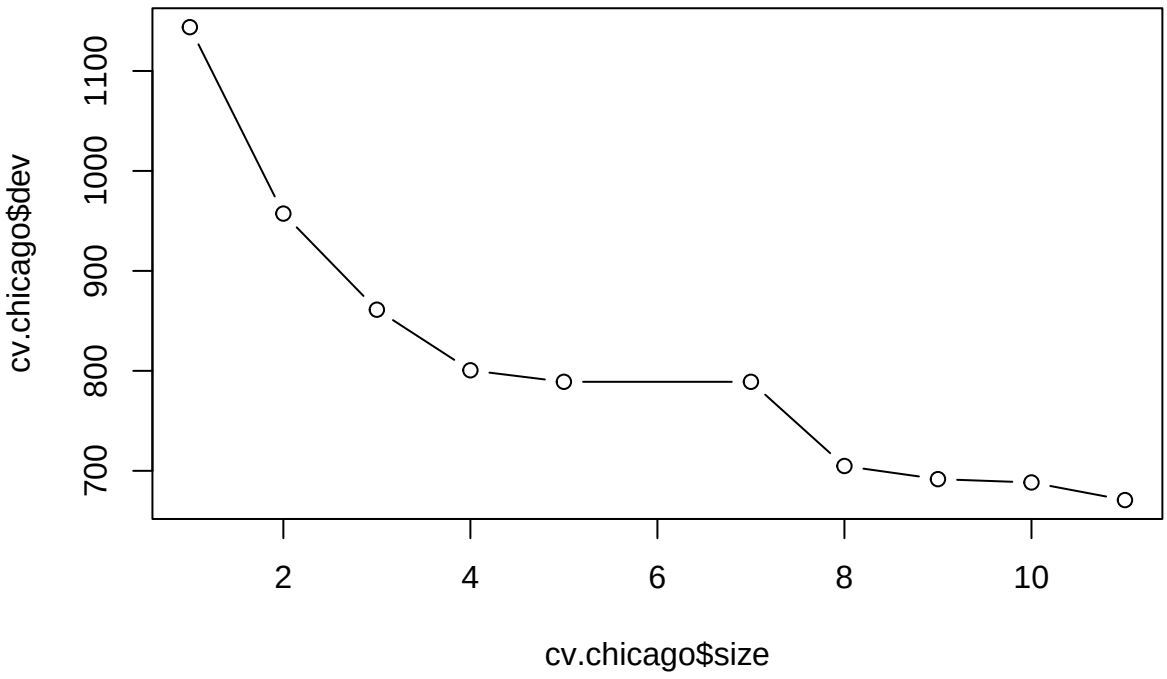



```r
#perform cross validation
cv.chicago = cv.tree(tree.chicago)
```

```
## Warning in tree(model = m[rand != i, , drop = FALSE]): NAs introduced by
## coercion

## Warning in pred1.tree(tree, tree.matrix(nd)): NAs introduced by coercion

## Warning in tree(model = m[rand != i, , drop = FALSE]): NAs introduced by
## coercion

## Warning in pred1.tree(tree, tree.matrix(nd)): NAs introduced by coercion

## Warning in tree(model = m[rand != i, , drop = FALSE]): NAs introduced by
## coercion

## Warning in pred1.tree(tree, tree.matrix(nd)): NAs introduced by coercion

## Warning in tree(model = m[rand != i, , drop = FALSE]): NAs introduced by
## coercion

## Warning in pred1.tree(tree, tree.matrix(nd)): NAs introduced by coercion

## Warning in tree(model = m[rand != i, , drop = FALSE]): NAs introduced by
## coercion

## Warning in pred1.tree(tree, tree.matrix(nd)): NAs introduced by coercion

## Warning in tree(model = m[rand != i, , drop = FALSE]): NAs introduced by
## coercion

## Warning in pred1.tree(tree, tree.matrix(nd)): NAs introduced by coercion

## Warning in tree(model = m[rand != i, , drop = FALSE]): NAs introduced by
## coercion

## Warning in pred1.tree(tree, tree.matrix(nd)): NAs introduced by coercion
```

```
## Warning in tree(model = m[rand != i, , drop = FALSE]): NAs introduced by
## coercion

## Warning in pred1.tree(tree, tree.matrix(nd)): NAs introduced by coercion

## Warning in tree(model = m[rand != i, , drop = FALSE]): NAs introduced by
## coercion

## Warning in pred1.tree(tree, tree.matrix(nd)): NAs introduced by coercion

## Warning in tree(model = m[rand != i, , drop = FALSE]): NAs introduced by
## coercion

## Warning in pred1.tree(tree, tree.matrix(nd)): NAs introduced by coercion
```

```
#perform cross validation
plot(cv.chicago$size, cv.chicago$dev, type='b')
```



we can see cv around cv=7 or 8 is best.

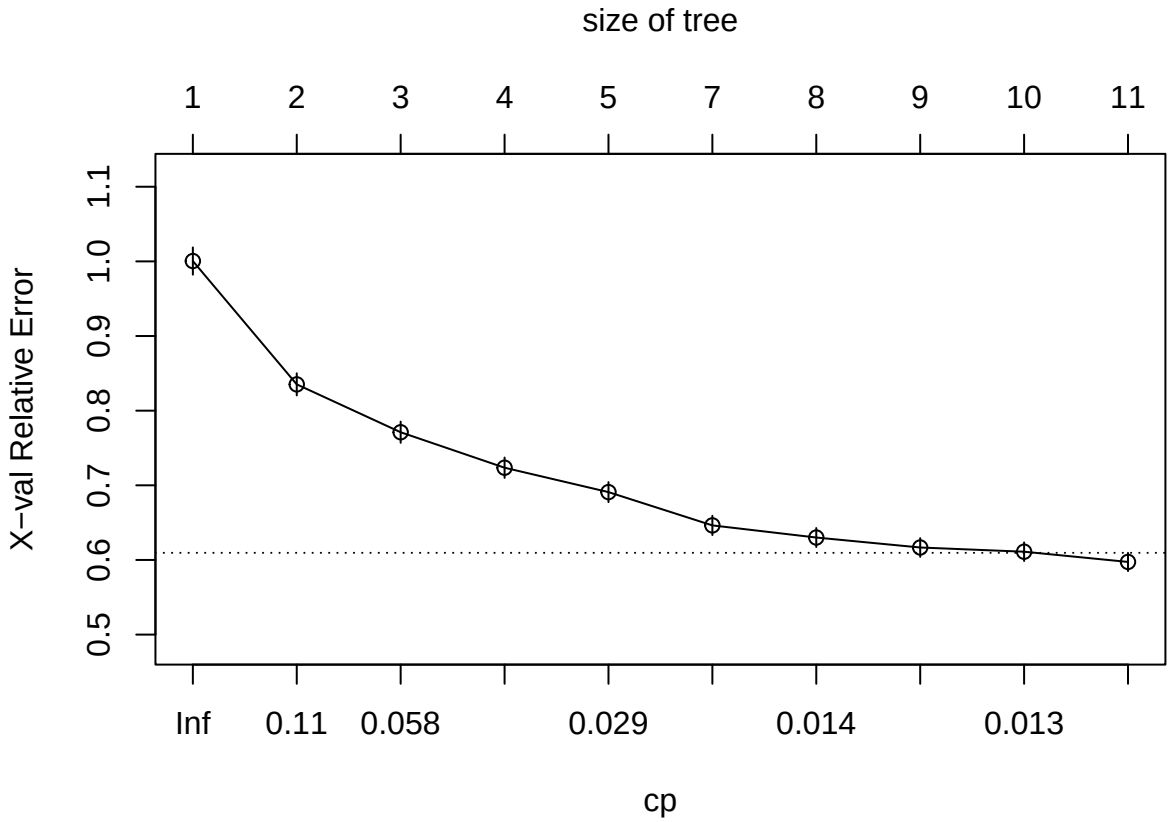
```
##this use rpart methods , we can see the prune tree
tree_chicago=rpart(response~. ,method="anova", data=data.frame(active_set))
prchicago<- prune(tree_chicago, cp=tree_chicago$cptable[which.min(tree_chicago$cptable[,"xerror"]),"CP"]
```

```
printcp(tree_chicago) # display the results
```

```
##
## Regression tree:
## rpart(formula = response ~ ., data = data.frame(active_set),
##     method = "anova")
##
## Variables actually used in tree construction:
## [1] Building_Square_Feet Latitude             Longitude
##
## Root node error: 9.2881e+13/7691 = 1.2077e+10
##
## n= 7691
```

```
##
##          CP nsplit rel error  xerror     xstd
## 1  0.171540      0   1.00000 1.00046 0.018260
## 2  0.069082      1   0.82846 0.83525 0.014883
## 3  0.047993      2   0.75938 0.77112 0.014205
## 4  0.030083      3   0.71138 0.72359 0.013596
## 5  0.027281      4   0.68130 0.69090 0.013387
## 6  0.015450      6   0.62674 0.64630 0.012920
## 7  0.013522      7   0.61129 0.63005 0.012712
## 8  0.013310      8   0.59777 0.61670 0.012536
## 9  0.011760      9   0.58446 0.61105 0.012479
## 10 0.010000     10   0.57270 0.59736 0.012203
```

```
plotcp(tree_chicago) # visualize cross-validation results
```




```
summary(tree_chicago) # detailed summary of splits
```

```
## Call:
## rpart(formula = response ~ ., data = data.frame(active_set),
##     method = "anova")
##   n= 7691
##
##            CP nsplit rel error    xerror       xstd
## 1  0.17154020      0 1.0000000 1.0004650 0.01826021
## 2  0.06908244      1 0.8284598 0.8352490 0.01488267
## 3  0.04799346      2 0.7593774 0.7711241 0.01420528
## 4  0.03008291      3 0.7113839 0.7235890 0.01359624
## 5  0.02728089      4 0.6813010 0.6909034 0.01338741
## 6  0.01544957      6 0.6267392 0.6463024 0.01291966
```

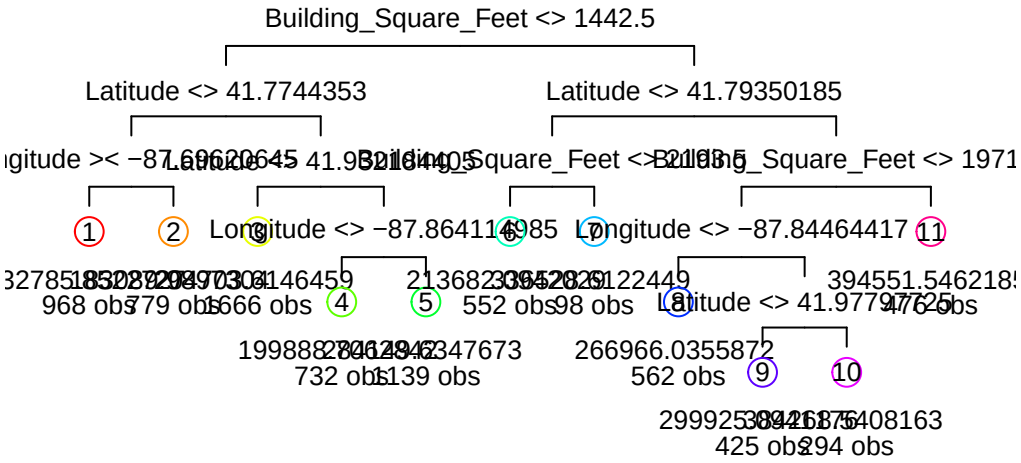
```
## 7  0.01352228      7 0.6112896 0.6300474 0.01271237
## 8  0.01331014      8 0.5977673 0.6166996 0.01253606
## 9  0.01175980      9 0.5844572 0.6110528 0.01247918
## 10 0.01000000     10 0.5726974 0.5973571 0.01220256
##
## Variable importance
## Building_Square_Feet             Latitude            Longitude
##                   22                   17                   11
##                Rooms             Bedrooms          Total_Baths
##                    9                    8                    7
##       Property_Class           Fireplaces     Land_Square_Feet
##                    7                    5                    4
##            Walkscore                  Age              Walkfac
##                    3                    2                    2
##          Central_Air             Basement
##                    1                    1
##
## Node number 1: 7691 observations,    complexity param=0.1715402
##   mean=235519.7, MSE=1.20766e+10
##   left son=2 (5284 obs) right son=3 (2407 obs)
##   Primary splits:
##       Building_Square_Feet < 1442.5    to the left,  improve=0.1715402, (0 missing)
##       Latitude             < 41.93228  to the left,  improve=0.1474151, (0 missing)
##       Property_Class       < 203.5     to the left,  improve=0.1260906, (0 missing)
##       Fireplaces           < 0.5       to the left,  improve=0.1062817, (0 missing)
##       Rooms                < 6.5       to the left,  improve=0.1027974, (0 missing)
##   Surrogate splits:
##       Rooms          < 6.5       to the left,  agree=0.828, adj=0.450, (0 split)
##       Bedrooms       < 3.5       to the left,  agree=0.815, adj=0.408, (0 split)
##       Property_Class < 203.5     to the left,  agree=0.813, adj=0.402, (0 split)
##       Total_Baths    < 1.75      to the left,  agree=0.806, adj=0.381, (0 split)
##       Fireplaces     < 0.5       to the left,  agree=0.760, adj=0.232, (0 split)
##
## Node number 2: 5284 observations,    complexity param=0.06908244
##   mean=204800.3, MSE=7.706308e+09
##   left son=4 (1747 obs) right son=5 (3537 obs)
##   Primary splits:
##       Latitude             < 41.77444  to the left,  improve=0.15757450, (0 missing)
##       Longitude            < -87.6704  to the right, improve=0.11958370, (0 missing)
##       Building_Square_Feet < 1136.5    to the left,  improve=0.04640453, (0 missing)
##       Property_Class       < 202.5     to the left,  improve=0.03364254, (0 missing)
##       Walkscore            < 75.5      to the left,  improve=0.02513207, (0 missing)
##   Surrogate splits:
##       Longitude        < -87.71972 to the right, agree=0.804, adj=0.407, (0 split)
##       Age              < 45.5      to the left,  agree=0.685, adj=0.049, (0 split)
##       Walkscore        < 13.5      to the left,  agree=0.675, adj=0.017, (0 split)
##       Land_Square_Feet < 25446     to the right, agree=0.672, adj=0.007, (0 split)
##       Total_Baths      < 3.25      to the right, agree=0.670, adj=0.001, (0 split)
##
## Node number 3: 2407 observations,    complexity param=0.04799346
##   mean=302956.7, MSE=1.505116e+10
##   left son=6 (650 obs) right son=7 (1757 obs)
##   Primary splits:
##       Latitude             < 41.7935   to the left,  improve=0.12304490, (0 missing)
```

```
##       Building_Square_Feet < 1971.5    to the left,  improve=0.10348850, (0 missing)
##       Longitude            < -87.6616  to the right, improve=0.07727960, (0 missing)
##       Property_Class       < 203.5     to the left,  improve=0.06712428, (0 missing)
##       Fireplaces           < 0.5       to the left,  improve=0.06663019, (0 missing)
##   Surrogate splits:
##       Longitude < -87.68484 to the right, agree=0.797, adj=0.248, (0 split)
##       Age       < 17.5      to the left,  agree=0.732, adj=0.006, (0 split)
##
## Node number 4: 1747 observations,    complexity param=0.0117598
##   mean=155216.8, MSE=3.367409e+09
##   left son=8 (968 obs) right son=9 (779 obs)
##   Primary splits:
##       Longitude            < -87.69621 to the right, improve=0.18566870, (0 missing)
##       Latitude             < 41.68363  to the left,  improve=0.13063250, (0 missing)
##       Land_Square_Feet     < 9987.5    to the left,  improve=0.05006554, (0 missing)
##       Building_Square_Feet < 1149      to the left,  improve=0.03856214, (0 missing)
##       Property_Class       < 202.5     to the left,  improve=0.02631906, (0 missing)
##   Surrogate splits:
##       Land_Square_Feet < 6302      to the left,  agree=0.645, adj=0.203, (0 split)
##       Latitude         < 41.73505  to the left,  agree=0.622, adj=0.153, (0 split)
##       Basement         splits as  LR, agree=0.599, adj=0.101, (0 split)
##       Age              < 59.5      to the right, agree=0.599, adj=0.101, (0 split)
##       Walkscore        < 27.5      to the right, agree=0.578, adj=0.053, (0 split)
##
## Node number 5: 3537 observations,    complexity param=0.02728089
##   mean=229290.7, MSE=8.035288e+09
##   left son=10 (1666 obs) right son=11 (1871 obs)
##   Primary splits:
##       Latitude             < 41.93218  to the left,  improve=0.06699057, (0 missing)
##       Building_Square_Feet < 1066.5    to the left,  improve=0.06279224, (0 missing)
##       Longitude            < -87.8399  to the left,  improve=0.05091406, (0 missing)
##       Property_Class       < 202.5     to the left,  improve=0.03969734, (0 missing)
##       Rooms                < 5.5       to the left,  improve=0.02328797, (0 missing)
##   Surrogate splits:
##       Central_Air      < 0.5       to the left,  agree=0.712, adj=0.390, (0 split)
##       Land_Square_Feet < 4916.5    to the left,  agree=0.653, adj=0.262, (0 split)
##       Longitude        < -87.90559 to the right, agree=0.627, adj=0.208, (0 split)
##       Age              < 73.5      to the right, agree=0.617, adj=0.186, (0 split)
##       Walkscore        < 67.5      to the right, agree=0.615, adj=0.182, (0 split)
##
## Node number 6: 650 observations,    complexity param=0.01352228
##   mean=232203.6, MSE=1.014714e+10
##   left son=12 (552 obs) right son=13 (98 obs)
##   Primary splits:
##       Building_Square_Feet < 2193.5    to the left,  improve=0.1904235, (0 missing)
##       Longitude            < -87.66609 to the right, improve=0.1701128, (0 missing)
##       Land_Square_Feet     < 9384.5    to the left,  improve=0.1537799, (0 missing)
##       Total_Baths          < 2.25      to the left,  improve=0.1185715, (0 missing)
##       Fireplaces           < 0.5       to the left,  improve=0.1087749, (0 missing)
##   Surrogate splits:
##       Rooms          < 8.5       to the left,  agree=0.869, adj=0.133, (0 split)
##       Total_Baths    < 3.25      to the left,  agree=0.868, adj=0.122, (0 split)
##       Walkscore      < 2.5       to the right, agree=0.857, adj=0.051, (0 split)
##       Property_Class < 207.5     to the left,  agree=0.855, adj=0.041, (0 split)
```

```
##       Bedrooms       < 4.5       to the left,  agree=0.852, adj=0.020, (0 split)
##
## Node number 7: 1757 observations,    complexity param=0.03008291
##   mean=329131.8, MSE=1.43283e+10
##   left son=14 (1281 obs) right son=15 (476 obs)
##   Primary splits:
##       Building_Square_Feet < 1971      to the left,  improve=0.11098930, (0 missing)
##       Fireplaces           < 0.5       to the left,  improve=0.07294090, (0 missing)
##       Property_Class       < 203.5     to the left,  improve=0.06612274, (0 missing)
##       Rooms                < 7.5       to the left,  improve=0.05909303, (0 missing)
##       Longitude            < -87.83647 to the left,  improve=0.05525795, (0 missing)
##   Surrogate splits:
##       Rooms            < 7.5       to the left,  agree=0.814, adj=0.315, (0 split)
##       Total_Baths      < 2.75      to the left,  agree=0.767, adj=0.141, (0 split)
##       Bedrooms         < 4.5       to the left,  agree=0.763, adj=0.126, (0 split)
##       Land_Square_Feet < 11710     to the left,  agree=0.747, adj=0.067, (0 split)
##       Fireplaces       < 1.5       to the left,  agree=0.746, adj=0.063, (0 split)
##
## Node number 8: 968 observations
##   mean=132785.9, MSE=1.961902e+09
##
## Node number 9: 779 observations
##   mean=183090, MSE=3.711783e+09
##
## Node number 10: 1666 observations
##   mean=204703.6, MSE=6.472305e+09
##
## Node number 11: 1871 observations,    complexity param=0.02728089
##   mean=251183.8, MSE=8.409421e+09
##   left son=22 (732 obs) right son=23 (1139 obs)
##   Primary splits:
##       Longitude            < -87.86411 to the left,  improve=0.20108220, (0 missing)
##       Walkscore            < 55.5      to the left,  improve=0.06122333, (0 missing)
##       Age                  < 74.5      to the left,  improve=0.06005468, (0 missing)
##       Building_Square_Feet < 1061.5    to the left,  improve=0.05260042, (0 missing)
##       Walkfac              splits as  LLRR, improve=0.05221116, (0 missing)
##   Surrogate splits:
##       Land_Square_Feet < 6817      to the right, agree=0.834, adj=0.577, (0 split)
##       Walkscore        < 48.5      to the left,  agree=0.754, adj=0.370, (0 split)
##       Walkfac          splits as  LRRR, agree=0.751, adj=0.363, (0 split)
##       Latitude         < 41.99624  to the right, agree=0.710, adj=0.258, (0 split)
##       Age              < 55.5      to the left,  agree=0.708, adj=0.253, (0 split)
##
## Node number 12: 552 observations
##   mean=213682.1, MSE=6.850621e+09
##
## Node number 13: 98 observations
##   mean=336528.6, MSE=1.58993e+10
##
## Node number 14: 1281 observations,    complexity param=0.01544957
##   mean=304822.8, MSE=1.198185e+10
##   left son=28 (562 obs) right son=29 (719 obs)
##   Primary splits:
##       Longitude      < -87.84464 to the left,  improve=0.09349124, (0 missing)
```

```
##       Basement       splits as  RL, improve=0.04368310, (0 missing)
##       Latitude       < 42.04956  to the left,  improve=0.03581356, (0 missing)
##       Fireplaces     < 0.5       to the left,  improve=0.03344614, (0 missing)
##       Property_Class < 203.5     to the left,  improve=0.02836835, (0 missing)
##   Surrogate splits:
##       Land_Square_Feet < 7074      to the right, agree=0.767, adj=0.468, (0 split)
##       Walkscore        < 48.5      to the left,  agree=0.754, adj=0.440, (0 split)
##       Walkfac          splits as  LRRR, agree=0.749, adj=0.429, (0 split)
##       Age              < 55.5      to the left,  agree=0.731, adj=0.386, (0 split)
##       Basement         splits as  RL, agree=0.710, adj=0.340, (0 split)
##
## Node number 15: 476 observations
##   mean=394551.5, MSE=1.477297e+10
##
## Node number 22: 732 observations
##   mean=199888.7, MSE=4.303068e+09
##
## Node number 23: 1139 observations
##   mean=284149.6, MSE=8.270719e+09
##
## Node number 28: 562 observations
##   mean=266966, MSE=8.031849e+09
##
## Node number 29: 719 observations,    complexity param=0.01331014
##   mean=334413.2, MSE=1.307354e+10
##   left son=58 (425 obs) right son=59 (294 obs)
##   Primary splits:
##       Latitude         < 41.97798  to the left,  improve=0.13151880, (0 missing)
##       Land_Square_Feet < 7497.5    to the left,  improve=0.08328073, (0 missing)
##       Fireplaces       < 0.5       to the left,  improve=0.06108352, (0 missing)
##       Property_Class   < 203.5     to the left,  improve=0.05114220, (0 missing)
##       Walkscore        < 60.5      to the right, improve=0.04346750, (0 missing)
##   Surrogate splits:
##       Land_Square_Feet < 5451      to the left,  agree=0.745, adj=0.378, (0 split)
##       Walkscore        < 60.5      to the right, agree=0.715, adj=0.303, (0 split)
##       Fireplaces       < 0.5       to the left,  agree=0.688, adj=0.238, (0 split)
##       Central_Air      < 0.5       to the left,  agree=0.682, adj=0.221, (0 split)
##       Walkfac          splits as  RRLL, agree=0.675, adj=0.204, (0 split)
##
## Node number 58: 425 observations
##   mean=299925.1, MSE=1.19241e+10
##
## Node number 59: 294 observations
##   mean=384268.5, MSE=1.053019e+10
```

```
#plot(tree_chicago ,uniform=TRUE, main="Regression Tree for Sale price ")
#text(tree_chicago,use.n=TRUE, all=TRUE, cex=.8)

draw.tree(tree_chicago, cex=0.8)
```



Here we can see the prune tree.

Random Forest

Set a seed to divide the entire data set into test and training sets (1/3, 2/3).

```
set.seed(1)
aa.train = sample(1:nrow(active_set_3), 2*nrow(active_set_3)/3)
aa.test = active_set_3[-aa.train, "Log_Sale_Price"]
```

```
# We tested tunegrid <- expand.grid(mtry = c(1:15)), to save time in Kint, we just use 12:14 here
tunegrid <- expand.grid(mtry = c(12:14))
rf_model <- train(Log_Sale_Price~., data = active_set_3, subset = aa.train, method = "rf", trControl = t
```

```
rf_model
```

```
## Random Forest
##
## 7691 samples
##   14 predictor
##
## No pre-processing
## Resampling: Cross-Validated (5 fold)
## Summary of sample sizes: 4100, 4102, 4103, 4102, 4101
## Resampling results across tuning parameters:
##
##   mtry  RMSE       Rsquared   MAE
##   12    0.3034476  0.5888319  0.2338388
##   13    0.3038206  0.5877960  0.2343525
##   14    0.3034219  0.5888348  0.2340642
##
## RMSE was used to select the optimal model using the smallest value.
## The final value used for the model was mtry = 14.
```

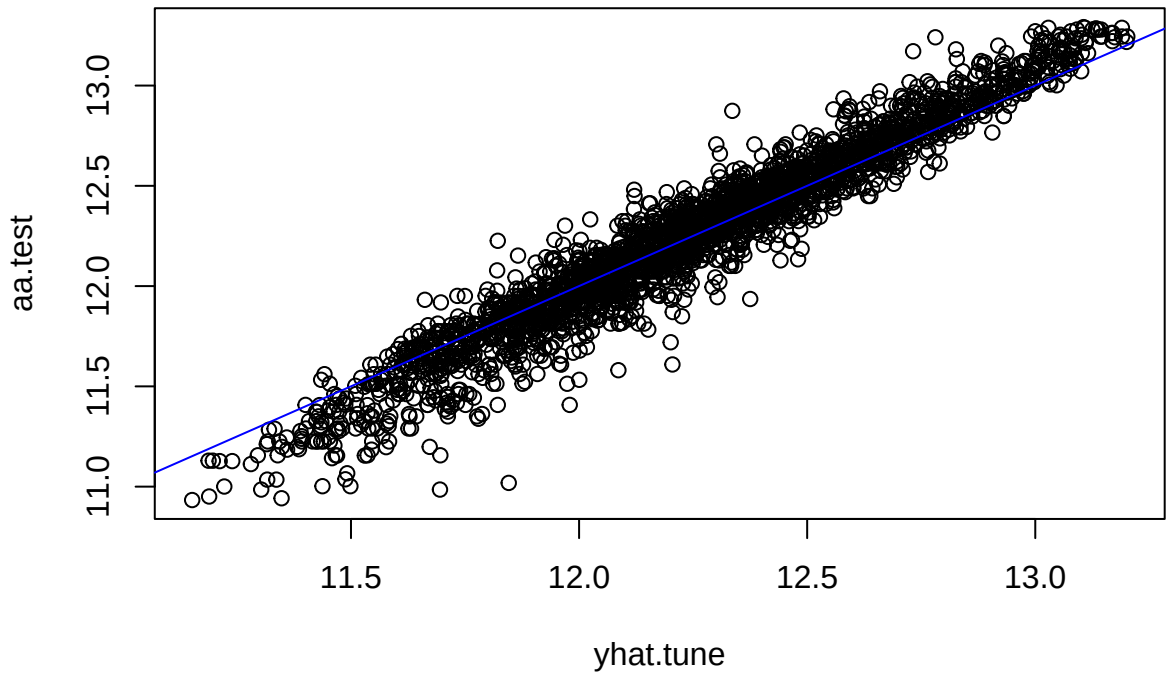
```
tune_rf <- randomForest(Log_Sale_Price~.,
                        data = active_set_3,
                        ntree = 500,
                        #mtry = as.integer(rf_model$bestTune),
                        mtry =13,
                        importance= TRUE,
                        na.action = na.omit,
```

```
                          replace = TRUE)

yhat.tune = predict(tune_rf, newdata = active_set_3[-aa.train,])
mean((yhat.tune - aa.test)^2)
```

```
## [1] 0.01638301
```

As a consequence of tweaking the forest, the mtry is set to 13 since the MSE/RMSE is decreased in this scenario. Then, using the modified model, forecast the Log_Sale_Price and see whether the MSE can be improved. As a result, the MSE is 0.01652111. As an example, consider the following forecast result.

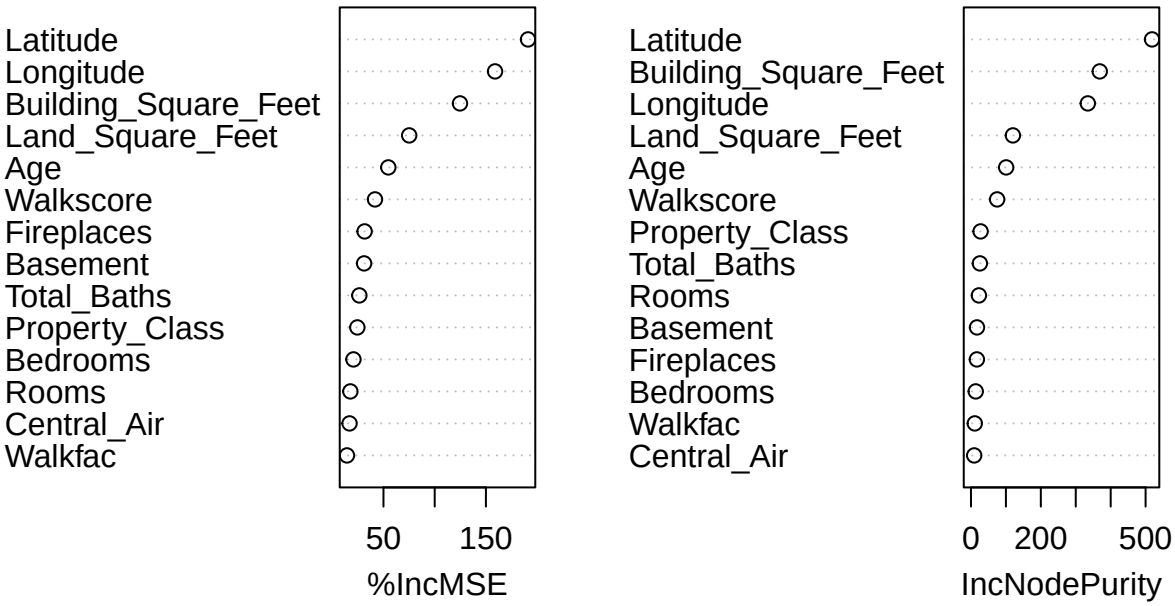
```
plot(yhat.tune, aa.test)
abline(0, 1, col = "blue")
```




```
importance(tune_rf)
```

```
##                          %IncMSE IncNodePurity
## Property_Class          24.51671     27.525884
## Rooms                   17.69879     22.654584
## Bedrooms                20.69721     13.621280
## Basement                31.10157     17.219637
## Fireplaces              31.59813     16.865403
## Central_Air             16.87565      9.187112
## Building_Square_Feet   124.66517    368.700892
## Land_Square_Feet        75.11796    120.027271
## Age                     54.74137    100.785873
## Longitude              158.91941    334.546470
## Latitude               191.11970    518.499092
## Walkscore               41.71597     74.887552
## Walkfac                 14.36915     10.908444
## Total_Baths             26.45195     25.487727
```

```
varImpPlot (tune_rf)
```



We can see according to random forest graph, we can see latitude and longtitude is most important, then it is building square feet, and then is land square feet. Now, we want to perform the random forest to check which factors are important to the sales price, without surprise, we can see location and building square feet contribute to much of the sales proce/

Perform SVM

We treat the walkfac as response. Kernel is poly

```
phenotype <- 'Walkfac'
predictors <- c(
"Property_Class","Rooms",   "Bedrooms", "Basement", "Fireplaces",   "Central_Air",  "Building_Square_Fe
)
```

Here we do the svm about classification of the walkfac. it has 4 level 1,2,3,4.

```
response <- rawdata[,phenotype]
active_set <- rawdata[,predictors]
```

Split the Train and Test set

```
sample = sample(1:nrow(active_set), nrow(active_set)*0.7)
svm.train = active_set[sample,]
svm.test = active_set[-sample,]
response_train <- response[sample]
response_test <- response[-sample]
```

Here we use the ksvm, and we use kernel is polydot.

```
chicago.svm_poly <- ksvm(response_train~.,data=data.frame(svm.train),
          kernel = "polydot", C = 1)
```

```
##  Setting default kernel parameters
```

Kernel is radial

```
chicago.svm_rad <- ksvm(response_train~.,data=data.frame(svm.train),
          kernel = "rbfdot", C = 1)
```

When kernel is poly

```
pred <- predict(chicago.svm_poly, newdata = data.frame(svm.test), type = "response")
ctable=table(pred, response_test)
```

```
accuracy <- sum(diag(ctable)/sum(ctable))
accuracy
```

```
## [1] 0.982669
```

we can see 98%

When kernel is rad

```
pred <- predict(chicago.svm_rad, newdata = data.frame(svm.test), type = "response")
ctable=table(pred, response_test)
```

```
accuracy <- sum(diag(ctable)/sum(ctable))
accuracy
```

```
## [1] 0.9566724
```

we can see accuracy is 95.7%

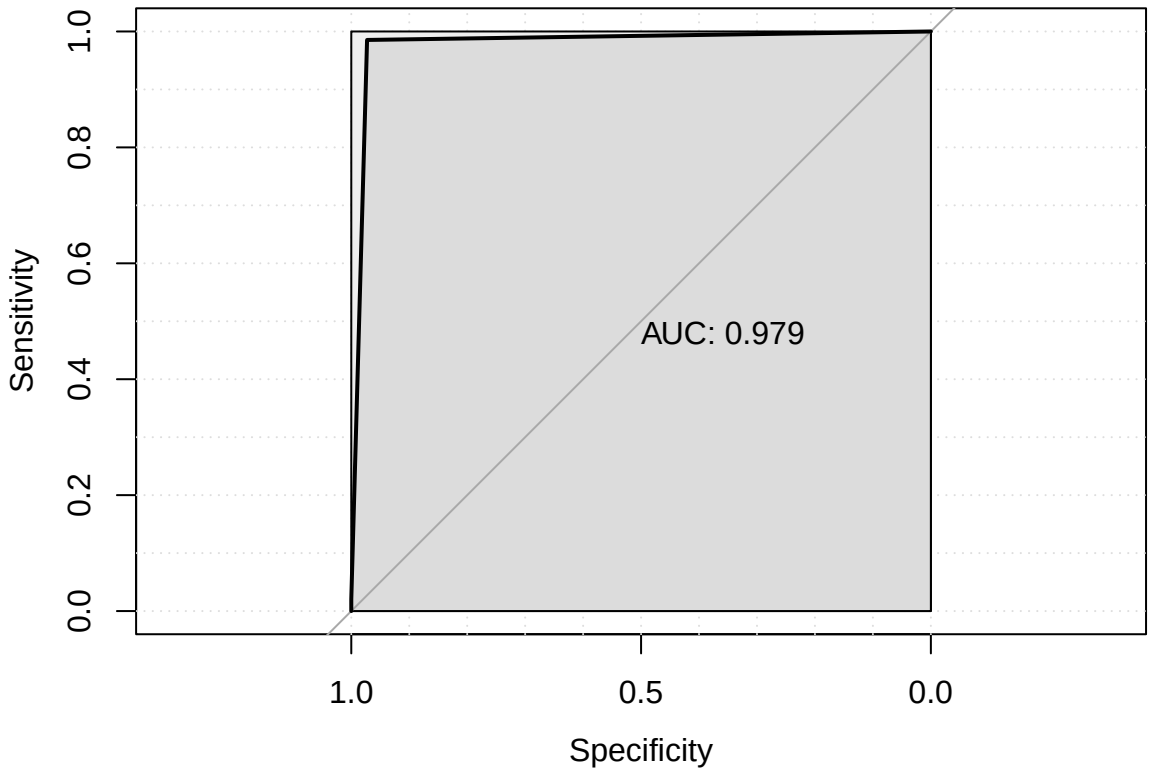
We can compare the accuracy between the kernel poly and radial, we can see the poly accuracy is high.

```
set.seed(20)
roc_test <- roc(response = response_test, predictor =as.numeric(pred),            plot=TRUE, auc.polygo
            print.auc=TRUE, show.thres=TRUE)
```

```
## Warning in roc.default(response = response_test, predictor = as.numeric(pred), :
## 'response' has more than two levels. Consider setting 'levels' explicitly or
## using 'multiclass.roc' instead
```

```
## Setting levels: control = 1, case = 2
```

```
## Setting direction: controls < cases
```



we can see it is very preicise

Neural network

Here, we train the neural network and we make the response is walkfac. We want to see the predictors, can predict the walk fac.

```
phenotype <- 'Walkfac'
predictors <- c(
"Rooms","Bedrooms", "Fireplaces","Central_Air","Building_Square_Feet",  "Land_Square_Feet","Age","Longi
)
```

```
response <- rawdata[,phenotype]
active_set <- rawdata[,predictors]
```

```
select_var <- c(
"Rooms","Bedrooms", "Fireplaces","Central_Air","Building_Square_Feet",  "Land_Square_Feet","Age","Longi
)
```

Train and Test set

```
subset=rawdata[,select_var]
set.seed(20)
sample = sample(1:nrow(active_set), nrow(active_set)*0.7)
neural.train =subset[sample,]
neural.test = subset[-sample,]
#response_train <- response[sample]
#response_test <- response[-sample]
```

```
neural_model <- neuralnet::neuralnet(Walkfac ~ ., data=neural.train, hidden=c(5,2), linear.output=FALSE
```

```
plot(neural_model)
```

```
walk_perform <- neuralnet::compute(neural_model,neural.test[1:11])
```

```
#store the net.results column
#prediction = walk_perform$net.result
walk_results <- data.frame(actual = neural.test$Walkfac , prediction = walk_perform$net.result)
#walk_results <- data.frame(actual = response_test , prediction = max.col(prediction))##choose max prob
head(walk_results)
```

```
##    actual prediction.1 prediction.2 prediction.3 prediction.4
## 5       2    0.3418097    0.3522205    0.2812522   0.02470799
## 10      1    0.3418097    0.3522205    0.2812522   0.02470799
## 11      3    0.3418097    0.3522205    0.2812522   0.02470799
## 13      2    0.3418097    0.3522205    0.2812522   0.02470799
## 17      1    0.3418097    0.3522205    0.2812522   0.02470799
## 20      3    0.3418097    0.3522205    0.2812522   0.02470799
```

2.5.3 Unupervised learning

```
rawdata %>% ggvis(~Longitude, ~Latitude, fill = ~Walkfac) %>% layer_points()
```

```
rawdata %>% ggvis(~Walkfac, ~Rooms, fill = ~Property_Class) %>% layer_points()
```

```
rawdata %>% ggvis(~Log_Sale_Price, ~Rooms, fill = ~Walkfac) %>% layer_points()
```

The data set doesn't need to be normalized

```
subset1 <- rawdata[c(14:15,17)]
set.seed(1234)
ind <- sample(2, nrow(subset1), replace=TRUE, prob=c(0.67, 0.33))
train <- subset1[ind==1, 1:3]
head(train)
```

```
##   Longitude Latitude Walkfac
## 1 -87.93069 42.05418       1
## 2 -87.76312 42.07116       1
## 3 -87.78104 41.91233       3
## 4 -87.83071 41.94842       1
## 6 -87.85841 41.82309       1
## 7 -87.87433 42.01341       1
```

```
test <- subset1[ind==2, 1:3]
head(test)
```

```
##    Longitude Latitude Walkfac
## 5  -87.86098 41.82934       2
## 11 -87.86910 41.92735       3
## 14 -87.83715 41.66106       1
## 16 -87.73230 42.02121       1
## 26 -87.84236 42.00146       1
## 28 -87.88604 41.91503       2
```

```
# Compose training labels
trainLabels <- subset1[ind==1,3]
# Compose test labels
```

```r
testLabels <- subset1[ind==2, 3]
# Inspect result
#print(testLabels)
```

```r
# Build the model
pred <- knn(train = train, test = test, cl = trainLabels, k=3)
# Inspect 'pred'
#pred
```

```r
CrossTable(x = testLabels, y = pred, prop.chisq=FALSE)
```

```
##
##
##    Cell Contents
## |-------------------------|
## |                       N |
## |           N / Row Total |
## |           N / Col Total |
## |         N / Table Total |
## |-------------------------|
##
##
## Total Observations in Table:  2514
##
##
##              | pred
##   testLabels |         1 |         2 |         3 |         4 | Row Total |
## -------------|-----------|-----------|-----------|-----------|-----------|
##            1 |       875 |         0 |         0 |         0 |       875 |
##              |     1.000 |     0.000 |     0.000 |     0.000 |     0.348 |
##              |     1.000 |     0.000 |     0.000 |     0.000 |           |
##              |     0.348 |     0.000 |     0.000 |     0.000 |           |
## -------------|-----------|-----------|-----------|-----------|-----------|
##            2 |         0 |       916 |         0 |         0 |       916 |
##              |     0.000 |     1.000 |     0.000 |     0.000 |     0.364 |
##              |     0.000 |     1.000 |     0.000 |     0.000 |           |
##              |     0.000 |     0.364 |     0.000 |     0.000 |           |
## -------------|-----------|-----------|-----------|-----------|-----------|
##            3 |         0 |         0 |       666 |         0 |       666 |
##              |     0.000 |     0.000 |     1.000 |     0.000 |     0.265 |
##              |     0.000 |     0.000 |     1.000 |     0.000 |           |
##              |     0.000 |     0.000 |     0.265 |     0.000 |           |
## -------------|-----------|-----------|-----------|-----------|-----------|
##            4 |         0 |         0 |         0 |        57 |        57 |
##              |     0.000 |     0.000 |     0.000 |     1.000 |     0.023 |
##              |     0.000 |     0.000 |     0.000 |     1.000 |           |
##              |     0.000 |     0.000 |     0.000 |     0.023 |           |
## -------------|-----------|-----------|-----------|-----------|-----------|
## Column Total |       875 |       916 |       666 |        57 |      2514 |
##              |     0.348 |     0.364 |     0.265 |     0.023 |           |
## -------------|-----------|-----------|-----------|-----------|-----------|
##
##
```

```
kmeans.re <- kmeans(subset1, 4)
kmeans.re$centers
```

```
##   Longitude Latitude  Walkfac
## 1 -87.84666 41.96890 2.000000
## 2 -87.69815 41.70622 2.000000
## 3 -87.73711 41.87211 3.076256
## 4 -87.86825 41.86256 1.000000
```

```
kmeans.re$totss
```

```
## [1] 5819.129
```

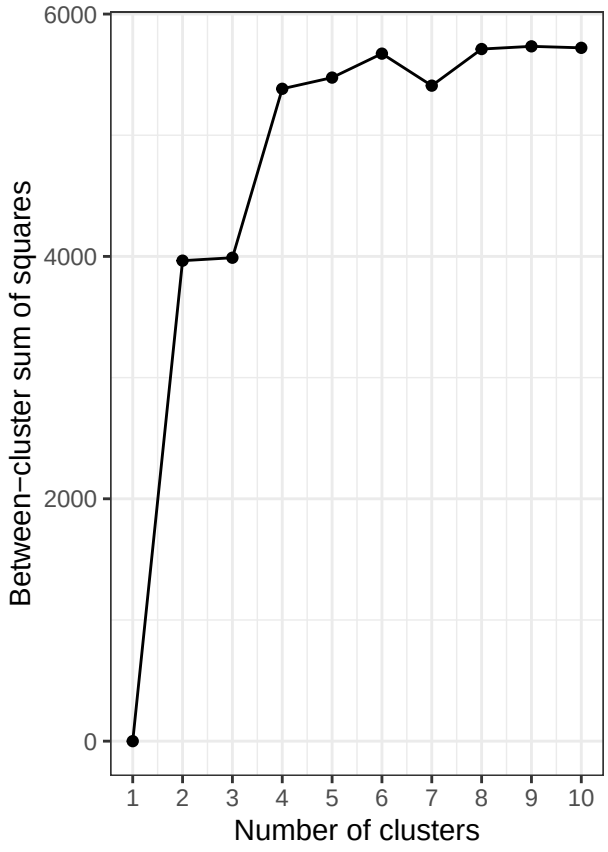
```
kmeans.re$size
```

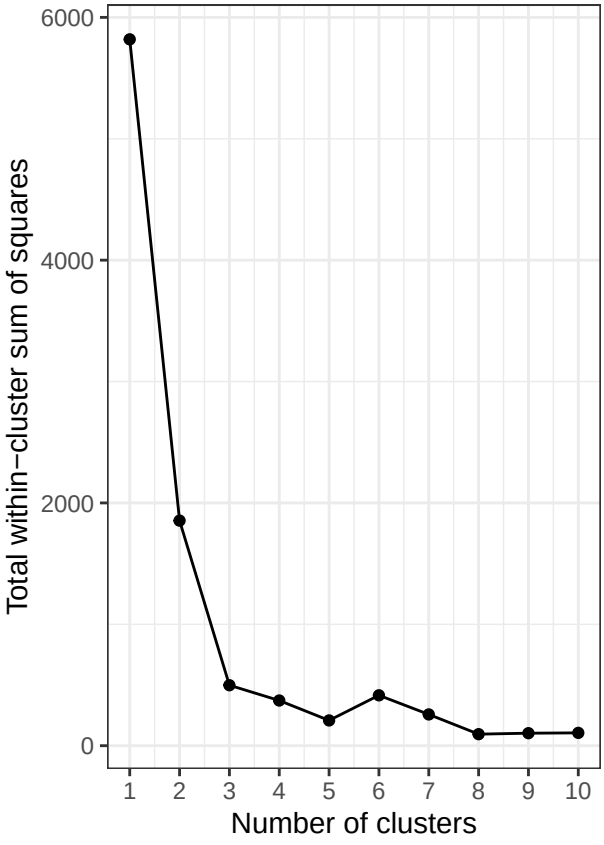
```
## [1] 1497 1259 2308 2627
```

```
cm <- table(subset1$Walkfac, kmeans.re$cluster)
cm
```

```
##
##        1    2    3    4
##   1    0    0    0 2627
##   2 1497 1259    0    0
##   3    0    0 2132    0
##   4    0    0  176    0
```

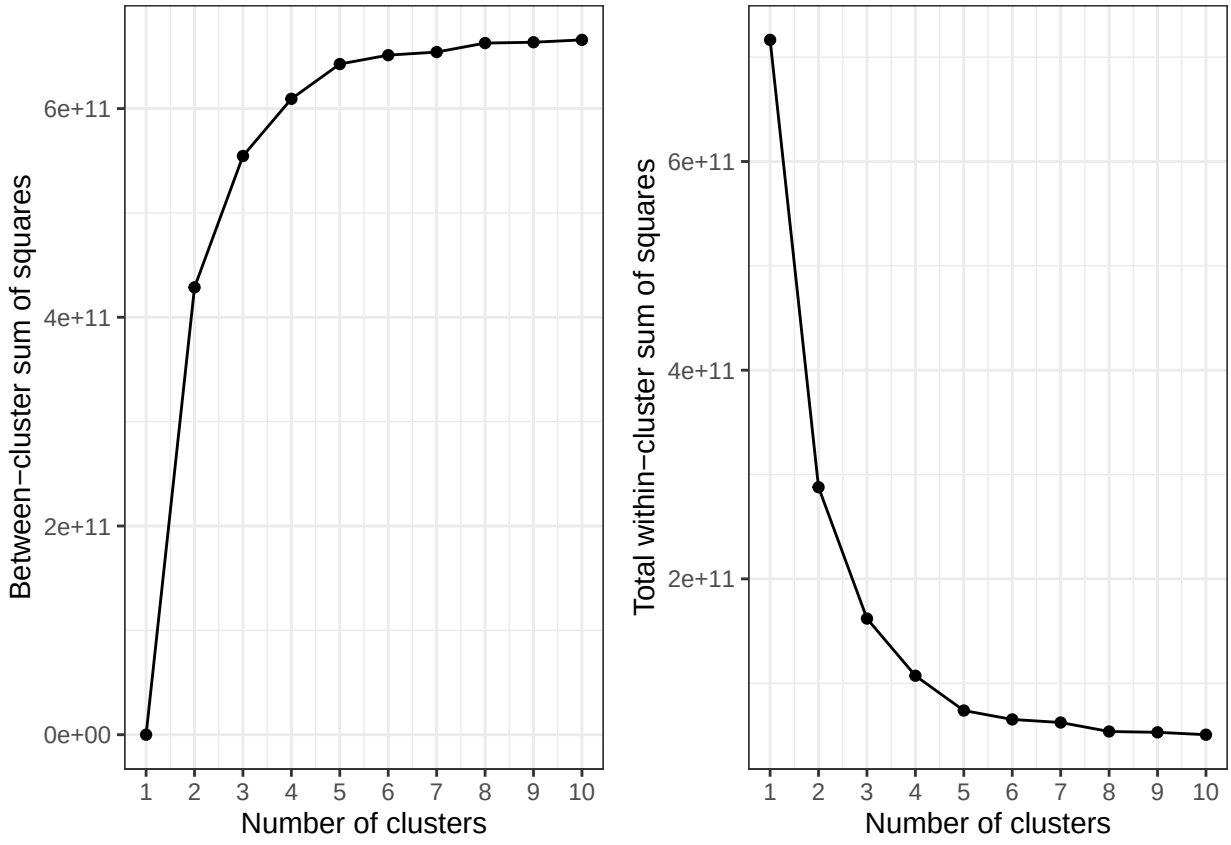
```
bss <- numeric()
wss <- numeric()

# Run the algorithm for different values of k
set.seed(1234)

for(i in 1:10){

  # For each k, calculate betweenss and tot.withinss
  bss[i] <- kmeans(subset1, centers=i)$betweenss
  wss[i] <- kmeans(subset1, centers=i)$tot.withinss

}

# Between-cluster sum of squares vs Choice of k
p3 <- qplot(1:10, bss, geom=c("point", "line"),
            xlab="Number of clusters", ylab="Between-cluster sum of squares") +
  scale_x_continuous(breaks=seq(0, 10, 1)) +
  theme_bw()

# Total within-cluster sum of squares vs Choice of k
p4 <- qplot(1:10, wss, geom=c("point", "line"),
            xlab="Number of clusters", ylab="Total within-cluster sum of squares") +
  scale_x_continuous(breaks=seq(0, 10, 1)) +
  theme_bw()

# Subplot
grid.arrange(p3, p4, ncol=2)
```






```
subset2 <- rawdata[c(3,6,7,11:15,17,20)]
bss <- numeric()
wss <- numeric()

# Run the algorithm for different values of k
set.seed(1234)

for(i in 1:10){

  # For each k, calculate betweenss and tot.withinss
  bss[i] <- kmeans(subset2, centers=i)$betweenss
  wss[i] <- kmeans(subset2, centers=i)$tot.withinss

}

# Between-cluster sum of squares vs Choice of k
p3 <- qplot(1:10, bss, geom=c("point", "line"),
            xlab="Number of clusters", ylab="Between-cluster sum of squares") +
  scale_x_continuous(breaks=seq(0, 10, 1)) +
  theme_bw()

# Total within-cluster sum of squares vs Choice of k
p4 <- qplot(1:10, wss, geom=c("point", "line"),
            xlab="Number of clusters", ylab="Total within-cluster sum of squares") +
  scale_x_continuous(breaks=seq(0, 10, 1)) +
  theme_bw()
```

```
# Subplot
grid.arrange(p3, p4, ncol=2)
```



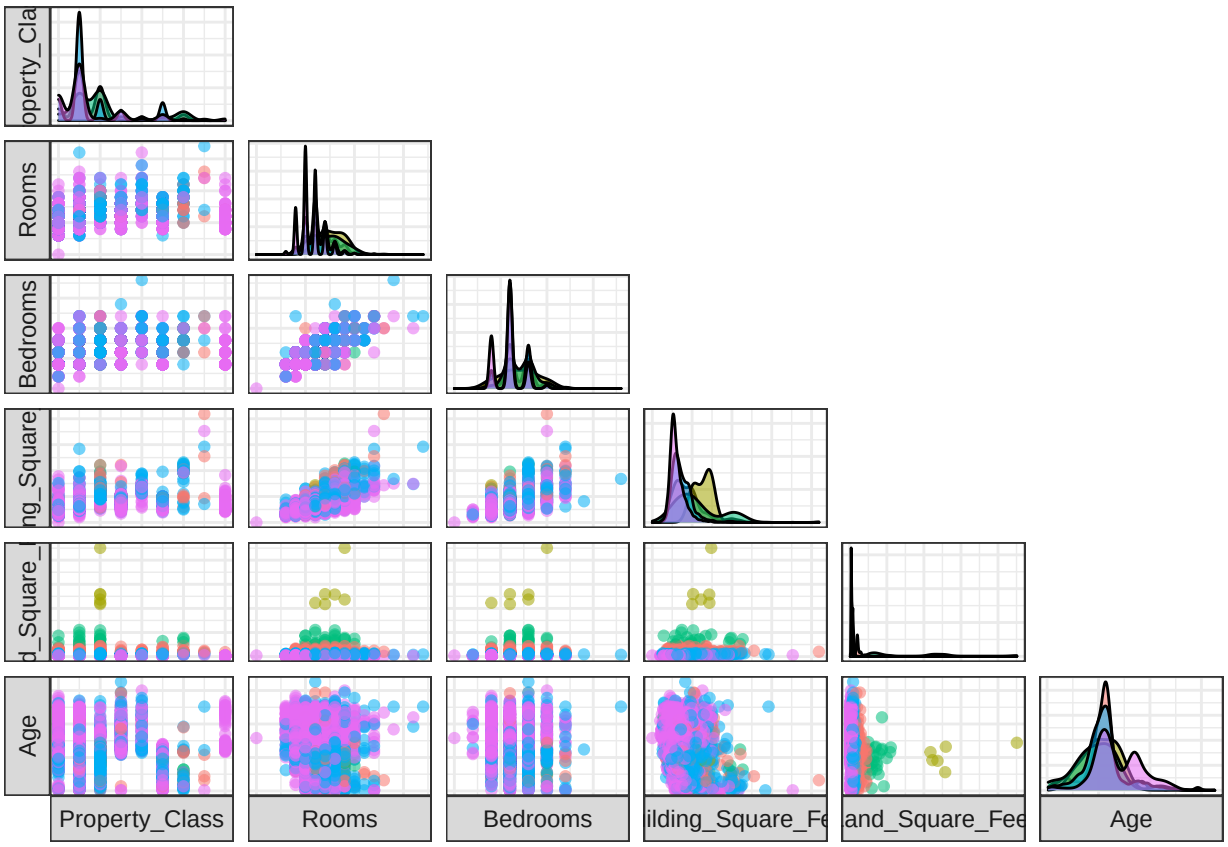

```
set.seed(1234)
kmeans.re2 <- kmeans(subset2, centers=5)
# Mean values of each cluster
aggregate(subset2, by=list(kmeans.re2$cluster), mean)
```

```
## Warning in mean.default(X[[i]], ...): argument is not numeric or logical:
## returning NA

## Warning in mean.default(X[[i]], ...): argument is not numeric or logical:
## returning NA

## Warning in mean.default(X[[i]], ...): argument is not numeric or logical:
## returning NA

## Warning in mean.default(X[[i]], ...): argument is not numeric or logical:
## returning NA

## Warning in mean.default(X[[i]], ...): argument is not numeric or logical:
## returning NA
```

```
##   Group.1 Property_Class    Rooms Bedrooms Building_Square_Feet
## 1       1       203.9038 6.665272 3.389121             1813.126
## 2       2       204.0000 7.666667 3.500000             2532.667
## 3       3       204.0800 7.160000 3.560000             2227.880
```

```
## 4       4       203.6215 6.176304 3.207845             1497.600
## 5       5       203.3537 5.571544 2.897534             1243.819
##   Land_Square_Feet      Age Longitude Latitude Walkfac Log_Sale_Price
## 1         23521.669 54.71967 -87.88496 41.85768      NA       12.51209
## 2        274161.833 52.50000 -88.20700 42.13609      NA       12.99510
## 3         72697.360 48.12000 -87.92524 41.81897      NA       12.76038
## 4          9028.542 54.64537 -87.88417 41.89328      NA       12.33066
## 5          4500.295 68.58650 -87.74781 41.84418      NA       12.21262
```

```
ggpairs(cbind(subset2, Cluster=as.factor(kmeans.re2$cluster)),
        columns=1:6, aes(colour=Cluster, alpha=0.5),
        lower=list(continuous="points"),
        upper=list(continuous="blank"),
        axisLabels="none", switch="both") +
        theme_bw()
```



3. Results

Linear regression

We do the linear regression, Sale price is our response, and predictors are “Property_Class”,“Rooms”,“Bedrooms”,“Basement”,“F
According to the linear regression result, we can see BasementPartial , Fireplaces, Central_Air , Building_Square_Feet, Land_Square_Feet, Longitude, Latitude , WalkfacWalker’s Paradise, Total_Baths is important factor to the sales price. Especially the location contribute to the sales price more, which is longitude and latitude, and then fireplaces and central air contribute to the sale price.

Logistic regression

Then we do the logistic regression about the central air, we want to know, based on the predictors, we want to know which is related to the central air, we can see, for example, rooms and BasementPartial and sales price is important to have central air, especially sales price, is important to the central air, the sales price higher, and we more likely to have central air. Also, we do the logistic regression about the basement.

Multinomial regression

Then We perform multinomial regression about Walkfac, we want to know how to perform 4 level "Car-Dependent", "Somewhat Walkable","Very Walkable", "Walker's Paradise". We can see longitude and latitude walkscore, is contribute to this formulation.

Decision tree

Then we do the decision tree method to predict sales price, we also can see the latitude and longitude is important to the sales price, and then is building square feet, this result is also same as linear regression. The MSE for this model is 0.1362816 and the RMSE is 0.367.

Random Forest

Then we perform random forest, we want to check which factor is important to sales price, we use the importance function, we can see the latitude, longitude and building square feet, land square feet and age is important to the sales price. The MSE for this model is 0.01652111 and the RMSE is 0.128.

SVM

Here, we use Walkfac, as response, and predictors, we want to know if we can use SVM methods. To classify them, we use different kernel "polydot","rbfdot",we can see if we use the polynomial kernel the accuracy is better than the radial kernel . The AUC for polynomial is 0.978. The result is very precise.

Neural network

From the model, we can see that with hidden layer value equals to 5 we can get more accurate outcomes comparing to hidden layer value equals to 2.

Unsupervised learning (K means clustering)

Here, we want to check whether there are clusters in the model. About the walkfac, it is hard to find some patterns from the graph. Then we can perform k means clustering to check, we can see from the graph, we can see 4 clusters is a good choice, which is also exactly data set tell us.

4. Conclusions

4.1 Significance

In supervised data with an emphasis on inference part, we find that there is a strong correlation between rooms and bedrooms and Building_Square_Feet. Then we use other methods do some related analysis and we get several points: 1.From multinomial regression about Walkfac, we explore the relationship among longitude, latitude and Walkfac, the iteration number is 100, and converge fast to 173, which can be used in comparisons of nested models. 2.We find one unit increase in the Longtitude is associated with 5 increase log odds for somewhat walkable vs car. We can see one unit increase in the walkscore is associated with 5.89 increase for car vs somewhat walkable. 3.With one unite in walkscore, is associated with 25 increase log odds to choose car vs walker paradise.

The result session focuses on the final performance of our prediction models, as well as the analysis of the significance of our research results. Cross-Validation test error rates for the outcomes predicted by models

are shown in the Supervised data with an emphasis on the prediction decision tree part. The model express variables' test error. For another part, random forest stands out with higher test accuracy. For SVM, different kernel produces different test accuracy. The radial kernel has a better effect than the poly kernel. However, compared to the results from previous plots, the initial guess is a little bit different. Overall, our selected models are capable of obtaining a more reliable result than a random guess.

In the random forest part, we can discover that latitude and longitude are the most important variables, the second is building square feet, and then land square feet. Then we examine the results by using random forest and we can see that the results are quite precise.

Based on the data analysis results, we can see all the machine learning methods, the most important factor is location, then is building square feet, it quite makes sense. And then we can see the facility is second most important to the house price which including central air and fireplace, age, property class. At the same time, we have established a high-accuracy prediction model, and made high-accuracy predictions on housing prices based on 14 variables. Finally, based on unsupervised learning, we cluster the houses according to latitude and longitude and summarize the features of different categories.

This project has big and detailed data with high accuracy models, and we believe it provided us with very insightful results about the Chicago house market . These models estimate the implied price of each feature in the price distribution, so it can better explain real-world phenomena and provide a more comprehensive understanding of the relationship between housing characteristics and prices.

4.2 Restrictions

This section will explain a project limitation and the accompanying recommendations for additional investigation. Several potentially valuable predictors (e.g., home quality, renovation time, and so on) were omitted from this research. Nonetheless, home quality is a challenging element to quantify and categorize. It will be impossible to obtain objective results if there are no generally acknowledged evaluation criteria. So, after careful study, we opted against using this variable in the model for the time being. In future studies, a scientific housing quality system needs to be established, and new predictors should be included in the model.

Appendix

R Code

knitr::opts_chunk$set(echo = TRUE) library(nnet) library(car) library(MASS) library(ggplot2) library(foreign) library(Hmisc) library(brant) library(openxlsx) library(ggvis) library(class) library(gmodels) library(skmeans) library(gridExtra) library(tidyverse) library(corrplot) library(gridExtra) library(GGally) library(knitr) library(tree) library(rpart) library(kernlab) library(doParallel) library(dplyr) library(cluster) library(factoextra) library(nnet) library(ISLR) library(foreign) library(Hmisc) library(brant) library(openxlsx) library(grid) library(pROC) library(tidyverse) library(xgboost) library(caret) library(tidyverse) library(kernlab) library(ranger) library(mltools) library(data.table) require(randomForest, quietly=TRUE) library(e1071) library(tidyverse)
library(corrplot) library(corrr) library(kernlab)
library(DT) library(ROCR) library(cvAUC) library(Deducer) library(psych) #Scatterplot matrix library(neuralnet) #artifical neural network library(maptree) library(ggpubr) library(randomForest) library (MASS)

rawdata <- read.csv(file="clean_data.csv",header=TRUE,sep=",") rawdata <- na.omit(rawdata) dim(rawdata) summary(rawdata)

anyNA(rawdata)

p1 <- rawdata %>% ggplot(aes(x = Building_Square_Feet, y = Sale_Price)) + geom_point() + xlab("Building Square Feet") + ylab("Sale Price") p2 <- rawdata %>% ggplot(aes(x = Building_Square_Feet, y = Log_Sale_Price)) + geom_point() + xlab("Building Square Feet") + ylab("Log Sale Price")

ggarrange(p1,p2,ncol=1,nrow=2,labels=c(“A”,“B”))

rawdata %>% ggplot(aes(x=as.factor(Rooms), y=Sale_Price)) + geom_boxplot() + xlab(“Rooms”) + ylab(“Sale Price”)

rawdata %>% ggplot(aes(x=as.factor(Age), y=Sale_Price)) + geom_boxplot() + xlab(“Year Built”) + ylab(“Sale Price”) + theme(axis.text.x = element_text(angle = 90, hjust = 1))

head(rawdata)

phenotype <- ‘Sale_Price’ predictors <- c(“Property_Class”,“Rooms”, “Bedrooms”, “Basement”, “Fireplaces”, “Central_Air”, “Building_Square_Feet”, “Land_Square_Feet”,“Age”,“Longitude”,“Latitude”,“Walkscore”,“Walkfac”)

response <- rawdata[,phenotype] active_set <- rawdata[,predictors]

full <- lm(response ~ ., data = data.frame(active_set)) summary(full)

full <- lm(response ~ . -Bedrooms , data = data.frame(active_set)) summary(full)

predictors_num <- c(“Rooms”, “Bedrooms”, “Fireplaces”, “Building_Square_Feet”, “Land_Square_Feet”,“Age”,“Longitude”)

cor(rawdata[,predictors_num])

predictors <- c(“Property_Class”,“Rooms”, “Bedrooms”, “Basement”, “Fireplaces”, “Central_Air”, “Building_Square_Feet”, “Land_Square_Feet”,“Age”,“Longitude”,“Latitude”,“Walkscore”,“Walkfac”,“Total_Baths”,“Sale_Price”)

active_set_2 <- rawdata[,predictors]

glm.fit=glm(as.factor(rawdata$Central_Air)~., data=data.frame(active_set_2),family=binomial)

summary(glm.fit)

glm.fit=glm(as.factor(rawdata$Basement)~., data=data.frame(active_set_2),family=binomial)

summary(glm.fit)

rawdata$Walkfac <- -factor(rawdata$Walkfac,levels=c(“Car-Dependent”, “Somewhat Walkable”,“Very Walkable”, “Walker’s Paradise”),labels=c(1,2,3,4))##reorder the data head(rawdata)

rawdata$Walkfac = as.factor(rawdata$Walkfac) rawdata$Walkfac <- -relevel(rawdata$Walkfac, ref = “1”)

mod.multinom <- multinom(as.factor(rawdata$Walkfac) ~ Longitude + Latitude+Walkscore ,data=rawdata) summary(mod.multinom)

Anova(mod.multinom)

predictors3 <- c(“Property_Class”,“Rooms”, “Bedrooms”, “Basement”, “Fireplaces”, “Central_Air”, “Building_Square_Feet”, “Land_Square_Feet”,“Age”,“Longitude”,“Latitude”,“Walkscore”,“Walkfac”,“Total_Baths”,“Log_Sale_Pr)

active_set_3 <- rawdata[,predictors3]

attach(active_set_3) set.seed(8) cu.train = sample(1: nrow(active_set_3), 2*nrow(active_set_3)/3) cu.test = active_set_3[-cu.train,] CU.test = active_set_3[-cu.train] tree.chicago=tree(Log_Sale_Price~., active_set_3, subset = cu.train) summary(tree.chicago)

chicago.predict = predict(tree.chicago, cu.test) mean((chicago.predict-cu.test$Log_Sale_Price)^2)

#plot(tree.chicago) #text(tree.chicago,pretty=0) draw.tree(tree.chicago, cex=0.8)

#perform cross validation cv.chicago = cv.tree(tree.chicago)

#perform cross validation plot(cv.chicago$size, cv.chicago$dev, type=‘b’)

##this use rpart methods , we can see the prune tree tree_chicago=rpart(response~. ,method="anova", data=data.frame(active_set)) prchicago<- prune(tree_chicago, cp=tree_chicago$cptable[which.min(tree_chicago$cptable[,"xerro

printcp(tree_chicago) # display the results plotcp(tree_chicago) # visualize cross-validation results summary(tree_chicago) # detailed summary of splits

#plot(tree_chicago ,uniform=TRUE, main="Regression Tree for Sale price") #text(tree_chicago,use.n=TRUE, all=TRUE, cex=.8)

draw.tree(tree_chicago, cex=0.8)

set.seed(1) aa.train = sample(1:nrow(active_set_3), 2*nrow(active_set_3)/3) aa.test = active_set_3[-aa.train, "Log_Sale_Price"]

tunegrid <- expand.grid(mtry = c(1:14)) rf_model <- train(Log_Sale_Price~., data = active_set_3, subset = aa.train, method = "rf", trControl = trainControl(method = "cv", number = 5), tuneGrid =tunegrid, prox = TRUE, allowParallel = FALSE)

rf_model

tune_rf <- randomForest(Log_Sale_Price~., data = active_set_3, ntree = 500, #mtry = as.integer(rf_model$bestTune), mtry =13, importance= TRUE, na.action = na.omit, replace = TRUE)

yhat.tune = predict(tune_rf, newdata = active_set_3[-aa.train,]) mean((yhat.tune - aa.test)^2)

plot(yhat.tune, aa.test) abline(0, 1, col = "blue")

importance(tune_rf) varImpPlot (tune_rf)

phenotype <- 'Walkfac' predictors <- c("Property_Class","Rooms", "Bedrooms", "Basement", "Fireplaces", "Central_Air", "Building_Square_Feet", "Land_Square_Feet","Age","Longitude","Latitude","Walkscore","Total_Baths")

response <- rawdata[,phenotype] active_set <- rawdata[,predictors]

sample = sample(1:nrow(active_set), nrow(active_set)*0.7) svm.train = active_set[sample,] svm.test = active_set[-sample,] response_train <- response[sample] response_test <- response[-sample]

chicago.svm_poly <- ksvm(response_train~.,data=data.frame(svm.train), kernel = "polydot", C = 1)

chicago.svm_rad <- ksvm(response_train~.,data=data.frame(svm.train), kernel = "rbfdot", C = 1)

pred <- predict(chicago.svm_poly, newdata = data.frame(svm.test), type = "response") ctable=table(pred, response_test)

accuracy <- sum(diag(ctable)/sum(ctable)) accuracy

pred <- predict(chicago.svm_rad, newdata = data.frame(svm.test), type = "response") ctable=table(pred, response_test)

accuracy <- sum(diag(ctable)/sum(ctable)) accuracy

set.seed(20) roc_test <- roc(response = response_test, predictor =as.numeric(pred), plot=TRUE, auc.polygon=TRUE, max.auc.polygon=TRUE, grid=TRUE, print.auc=TRUE, show.thres=TRUE)

phenotype <- 'Walkfac' predictors <- c("Rooms","Bedrooms", "Fireplaces","Central_Air","Building_Square_Feet", "Land_Square_Feet","Age","Longitude","Latitude","Walkscore","Total_Baths")

response <- rawdata[,phenotype] active_set <- rawdata[,predictors]

select_var <- c("Rooms","Bedrooms", "Fireplaces","Central_Air","Building_Square_Feet", "Land_Square_Feet","Age","Lo)

subset=rawdata[,select_var] set.seed(20) sample = sample(1:nrow(active_set), nrow(active_set)*0.7) neural.train =subset[sample,] neural.test = subset[-sample,] #response_train <- response[sample] #response_test <- response[-sample]

neural_model <- neuralnet::neuralnet(Walkfac ~ ., data=neural.train, hidden=c(5,2), linear.output=FALSE, threshold=0.01)

plot(neural_model)

walk_perform <- neuralnet::compute(neural_model,neural.test[1:11])

#store the net.results column #prediction = walk_perform$net.result walk_results <- data.frame(actual = neural.test$Walkfac , prediction = walk_perform$net.result) #walk_results <- data.frame(actual = response_test , prediction = max.col(prediction))##choose max probability as our prediction head(walk_results)

rawdata %>% ggvis(~Longitude, ~Latitude, fill = ~Walkfac) %>% layer_points()

rawdata %>% ggvis(~Walkfac, ~Rooms, fill = ~Property_Class) %>% layer_points()

rawdata %>% ggvis(~Log_Sale_Price, ~Rooms, fill = ~Walkfac) %>% layer_points()

subset1 <- rawdata[c(14:15,17)] set.seed(1234) ind <- sample(2, nrow(subset1), replace=TRUE, prob=c(0.67, 0.33)) train <- subset1[ind==1, 1:3] head(train) test <- subset1[ind==2, 1:3] head(test)

Compose training labels

trainLabels <- subset1[ind==1,3] # Compose test labels testLabels <- subset1[ind==2, 3] # Inspect result #print(testLabels)

Build the model

pred <- knn(train = train, test = test, cl = trainLabels, k=3) # Inspect 'pred' #pred

CrossTable(x = testLabels, y = pred, prop.chisq=FALSE)

kmeans.re <- kmeans(subset1, 4) kmeans.re$centers kmeans.re$totss kmeans.re$size

cm <- table(subset1$Walkfac, kmeans.re$cluster) cm

bss <- numeric() wss <- numeric()

Run the algorithm for different values of k

set.seed(1234)

for(i in 1:10){

For each k, calculate betweenss and tot.withinss bss[i] <- kmeans(subset1, centers=i)$betweenss wss[i] <- kmeans(subset1, centers = i)$tot.withinss

}

Between-cluster sum of squares vs Choice of k

p3 <- qplot(1:10, bss, geom=c("point", "line"), xlab="Number of clusters", ylab="Between-cluster sum of squares") + scale_x_continuous(breaks=seq(0, 10, 1)) + theme_bw()

Total within-cluster sum of squares vs Choice of k

p4 <- qplot(1:10, wss, geom=c("point", "line"), xlab="Number of clusters", ylab="Total within-cluster sum of squares") + scale_x_continuous(breaks=seq(0, 10, 1)) + theme_bw()

Subplot

grid.arrange(p3, p4, ncol=2)

subset2 <- rawdata[c(3,6,7,11:15,17,20)] bss <- numeric() wss <- numeric()

Run the algorithm for different values of k

set.seed(1234)

for(i in 1:10){

For each k, calculate betweenss and tot.withinss bss[i] <- kmeans(subset2, centers=i)$between ss wss[i] <- kmeans(subset2, centers = i)$tot.withinss

}

Between-cluster sum of squares vs Choice of k

p3 <- qplot(1:10, bss, geom=c("point", "line"), xlab="Number of clusters", ylab="Between-cluster sum of squares") + scale_x_continuous(breaks=seq(0, 10, 1)) + theme_bw()

Total within-cluster sum of squares vs Choice of k

p4 <- qplot(1:10, wss, geom=c("point", "line"), xlab="Number of clusters", ylab="Total within-cluster sum of squares") + scale_x_continuous(breaks=seq(0, 10, 1)) + theme_bw()

Subplot

grid.arrange(p3, p4, ncol=2)

set.seed(1234) kmeans.re2 <- kmeans(subset2, centers=5) # Mean values of each cluster aggregate(subset2, by=list(kmeans.re2$cluster), mean)

ggpairs(cbind(subset2, Cluster=as.factor(kmeans.re2$cluster)), columns=1:6, aes(colour=Cluster, alpha=0.5), lower=list(continuous="points"), upper=list(continuous="blank"), axisLabels="none", switch="both") + theme_bw()

Bibliography

Bailey, M. J., Muth, R. F., & Nourse, H. O. (1963). A regression method for real estate price index construction. Journal of the American Statistical Association, 58(304), 933-942.

Benjamin, J., Guttery, R., & Sirmans, C. F. (2004). Mass appraisal: An introduction to multiple regression analysis for real estate valuation. Journal of Real Estate Practice and Education, 7(1), 65-77.

Isakson, H. (1998). The review of real estate appraisals using multiple regression analysis. Journal of Real Estate Research, 15(2), 177-190.

Isakson, H. R. (2001). Using multiple regression analysis in real estate appraisal. The Appraisal Journal, 69(4), 424.

Mak, S., Choy, L., & Ho, W. (2010). Quantile regression estimates of Hong Kong real estate prices. Urban Studies, 47(11), 2461-247

Pavlov, A. D. (2000). Space-varying regression coefficients: A semi-parametric approach applied to real estate markets. Real Estate Economics, 28(2), 249-283.