STATS 504 - Assignment 4

November 15, 2021

Contents

Introduction . . . 1
Method . . . 1
Data Definitions . . . 1
Statistical Model . . . 1
Model Explanation and Results . . . 3
Exploratory Data Analysis . . . 3
Single Regression Tree . . . 4
Random Forest . . . 6
RMSE of different models . . . 7
Conclusion . . . 7
Appendix . . . 8
RMSE . . . 8
Codes . . . 9

Introduction

During 2012 to 2013 in New Taipei, some frauds gain illegal profit by buying and selling houses in an unusual price that is much different from the market. In order to determine which cases are controlled by frauds, accurate estimates of the house prices can really help. In this project, we aim to propose a model that is not only easy to interpret but is also able to predict house price in an accurate way according to some basic information of houses. **We use the real estate valuation data set from Sindian Dist., New Taipei during 2012 to 2013 to build our model.** User can input the information of target house, such as the house age and transaction date, and then our program would return the expected cost of the house in TWD at the time of transaction. We expect that, with the help of this model, people could determine whether the buying or selling house price is suspected or not. Moreover, the model also shows that, some features of the house play an important role in estimating the house price, such as the distance to the nearest MRT station and house age.

Method

Data Definitions

We first introduce the variables used in the model. Those variables correspond to the basic information of house, and we will delve deep into them later in the results section. There are six feature variables in total, they are: the transaction date, the house age (in year), the distance to the nearest MRT station (in meter), the number of convenience stores in the living circle on foot, the geographic coordinate latitude (in degree), and the geographic coordinate longitude (in degree). Besides, we have one more output variable, which is the house price of unit area in 10000 New Taiwan Dollar/Ping, where Ping is a local unit and 1 Ping equals to 3.3 meter squared.

Statistical Model

Our goal is to estimate the house price according to the house information, and the outcome variable house price is numerical and continuous. Therefore, due to the nature of the data, we can think of this as a

regression problem and we can use a regression model to explain the data. There are many methods related to regression. In this project, we mainly considered four of them: K nearest neighbors model(KNN), tree-based model, generalized additive model(GAM), and linear regression model. We selected the best model from those four methods, and our final choice is set to be Random Forest, which is a special case of tree-based model. Before going deep into the model, let us introduce the basic idea of those four methods first. In fact, they are easy to understand.

KNN is a very intuitive and natural method used for regression problems. Basically the idea is, if we want to know the estimated price of the target house, then what we need to do is just to find several neighbors that are close to this house, and we estimate the price of the target house by averaging the price of its neighbors. Since we have more information than location only, the neighbor here can be further understanded in a broader sense, which means that we consider houses that have the similar condition in all aspects as our target house. For example, we may consider houses that have nearly the same house age and similar distance to the nearest MRT station. Therefore, if we are going to use this method, then as pointed out by its name, what we have to care about is the number of neighbors k that we refer to. More specifically, if we consider a relatively large k, then we may average out the unique property of the target house, and thus incur higher bias in estimates. On the contrary, if we choose to use a small k, then we may suffer from large variance, since sometimes you can not find such a close neighbor for your target house. Therefore, we need to find a moderate number of neighbors to characterize our target house and then estimate the price in an accurate way.

Tree-based models view this problem in another perspective, but it is also reasonable and easy to interpret. The tree-based model estimates the price of the target house in several separate steps, and in each step we make a decision based only on one variable, which is usually considered the most important variable under current circumstances. For example, accessibility to public transportation is an asset and can drive the value of a property up, as many people view this as a priority when they buy houses. Therefore, a possible way to determine the price of the target house is, as in our data, we take a look at the distance to the nearest MRT station in the first step. If the distance is less than 2 kilometers, then we believe the house price is no less than 400,000 TWD per Ping. Then we go to our next step, we consider the house age, and assert that if the house age is greater than 20, the price would be no more than 600,000 TWD per ping. We continue to narrow down our search scope by iterating the aforementioned steps, and each time we only consider the most important variable at the current stage. Then finally we can have a very small interval of possible prices for the target house, and we estimate the house price by this value.

Generalized additive models can also be a possible choice for a regression problem. The basic idea of this model is to add up the effect of variables term by term, but rather than assuming a specific relation between outcome and predictors, GAM grants freedom to each variable. More specifically, house age may be related to the house price in a quadratic form, while distance to the nearest MRT station may in fact be in a cubic form. Generally, they can be in any form they want, and for each variable, we can use a smoothing spline to characterize its true relation with housing price. A smoothing spline is a statistical tool for modeling irregular curves, and it works well in many sophisticated situations.

Linear regression model, however, is a reduced model and is a special case of generalized additive model. Now every variable is assumed to be linear with the outcome house price, indicating that the price is proportional to each variable. Then, a unit increase in one variable will lead to a proportional amount of changes in the house price. So this may not be a good choice, since we know that variables like transaction date certainly are not linear to the housing price, otherwise the price will go to infinity one day in the future. The reason we want to try linear regression is that it provides a benchmark performance for our analysis. By comparing a specific model to our linear model, we know how well a model fitted.

Now we need to find a metric to measure the goodness of fit for each model. As is often the case in many regression problems, root mean squared error(RMSE) can be considered a good choice. More specifically, if for ith observation we define the fitted value as $\hat{y_i}$, and define the true value as y_i, then

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(\hat{y_i} - y_i)^2}$$

the smaller the RMSE, the better the fit. We also use 10-fold cross validation to verify our model. Cross validation is a statistical tool that can be helpful in model selection. The idea is, we divide the data into 10 folds uniformly, and each time we leave out one fold as our test set, and we use the remaining 9 folds to build our model. By doing so, we found that a special case of tree-based model, random forest, has the smallest RMSE. Therefore, we choose random forest to be our final model, and we will elaborate the model in the next section.

Model Explanation and Results

Exploratory Data Analysis

The data used for this analysis contains 414 house observations and eight variables, one being the house id, which is irrelevant to our purpose. Therefore, we exclude this variable, with seven remaining while six of them are predictors and house price is our outcome variable. There are no missing values for any of the variables. We plot the house price versus location below. Moreover, we also include the distance to nearest MRT station and show its effect on the price.

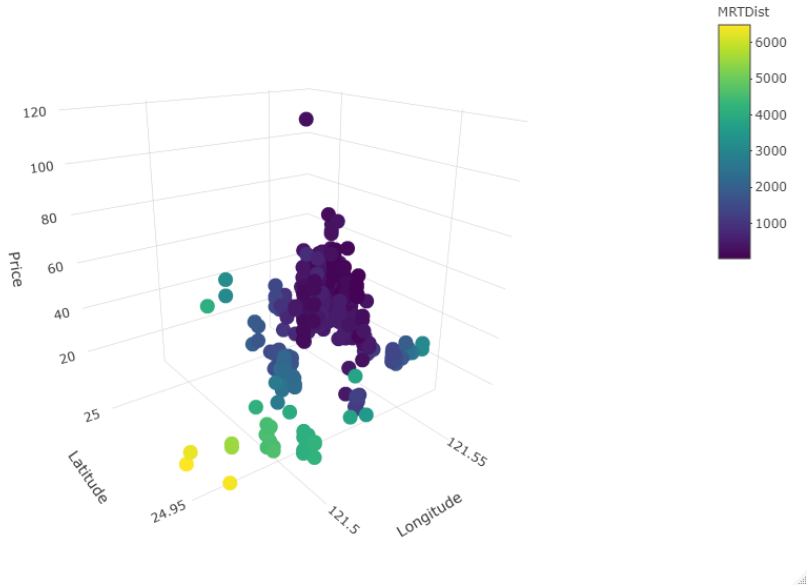

Figure 1: Scatterplot: Housing price vs. Location. The color represents the distance to the nearest MRT station.

As shown in Figure 1, houses with higher prices seems to be clustered around latitude 25 degree and longitude 121.55 degree. Besides, we use the color to represent the distance to the nearest MRT station. The deeper the color, the smaller the distance. As we can see, for all those houses whose prices are over 400,000 TWD, their distance to the nearest MRT station is smaller than 2000 meters. This justifies our choice of tree-based method, since these data can indeed be divided according to certain variables, one at a time respectively. Aside from this, one thing we notice that is, there is an outlier in the data set. As we can see clearly, on the top there is a case with house price nearly up to 1,200,000 TWD, which is way much higher than its counterpart. Considering that our main goal is model the price for the majority of the houses, we exclude this point from our data.

We can also plot the price against the house age and the distance to the nearest MRT station. Now we use color to represent the number of convenience stores. As shown in Figure 2, except for the outlier we identified, the house price seems to positively related to the number of stores. That is, the more convenience stores nearby, the higher the house price. Also, the effect of MRT distance is again justified in this plot, while house age seems to have a quadratic effect on the house price.

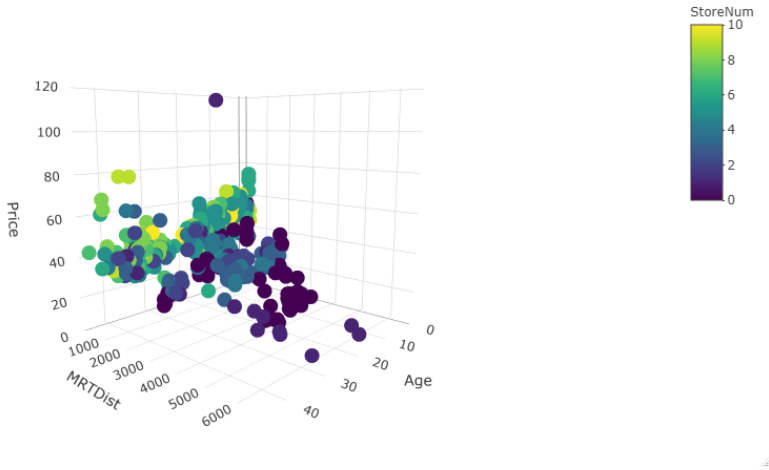

Figure 2: Scatterplot: Housing price vs. House Age and MRT Distance. The color represents the number of stores.

We can also plot the six covariates versus house prices per unit area separately on a two dimensional scatter graph, as shown in Figure 3. The house age shows a downward trend in the housing prices until the house age reaches about 25 while there seems to be an increase in the prices in houses older than 30 years of age. For latitude and longitude, the house prices are higher and vary more as both of the covariates increase. Number of convenience stores near also showed positive relationship with the house prices while as the distance to the nearest MRT station increased, the house prices decreased. The distance to the nearest MRT station also does not show a linear relationship with the house prices. The date of transaction date showed no clear relationship in the scatterplot and the house prices varied across each date.

To sum up, these exploratory data analysis give us confidence in applying a tree-based model. Next, let us illustrate how a single tree may look like.

Single Regression Tree

As we can see from Figure 4, the idea is very natural and straightforward. We make our decision based on many steps. In the first step, if the distance to the nearest MRT station is less than 827 meters, then we go to the next step on the right, otherwise we go left. Then, if the housing age is greater than or equal to 10 years, we go left, and so on and so forth. Eventually we reach the bottom nodes of the tree, in which the number **n** stands for the estimated house price and the **percentage** means that how many percent of the houses are estimated to this price.

For example, if a house has a distance to the nearest MRT station equal to 800, a house age equal to 8, a latitude equal to 20 degree, then the estimated price for this house is 380,000 TWD per Ping, and there are 9% of houses estimated with the same price. Therefore, by dividing the data into tiny pieces and groups, regression tree assigned estimated price for those houses case by case.

Regression tree has a strong performance in terms of prediction error. In fact, the RMSE of a single tree is actually lower than many methods, such as KNN, GAM with smoothing splines and linear regression. This shows its power in a regression problem. However, a single tree is not enough for our prediction, as it does not take all the variables into account. In order to make our prediction more reliable, we need to further distinguish a house from its similar counterparts, even though they are assigned with the same prediction price under the same regression tree.

Therefore, we need the help of random forest, which is selected to be our final model.

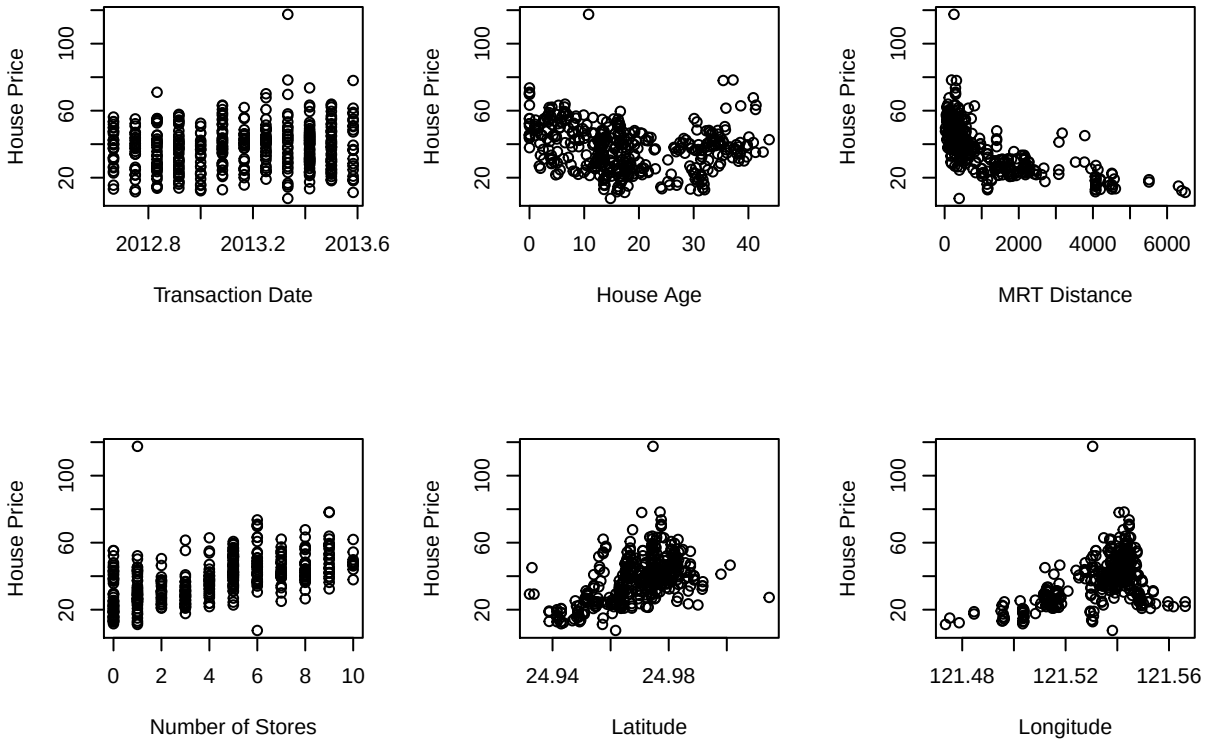

Figure 3: Scatterplot: Housing price vs. variables.

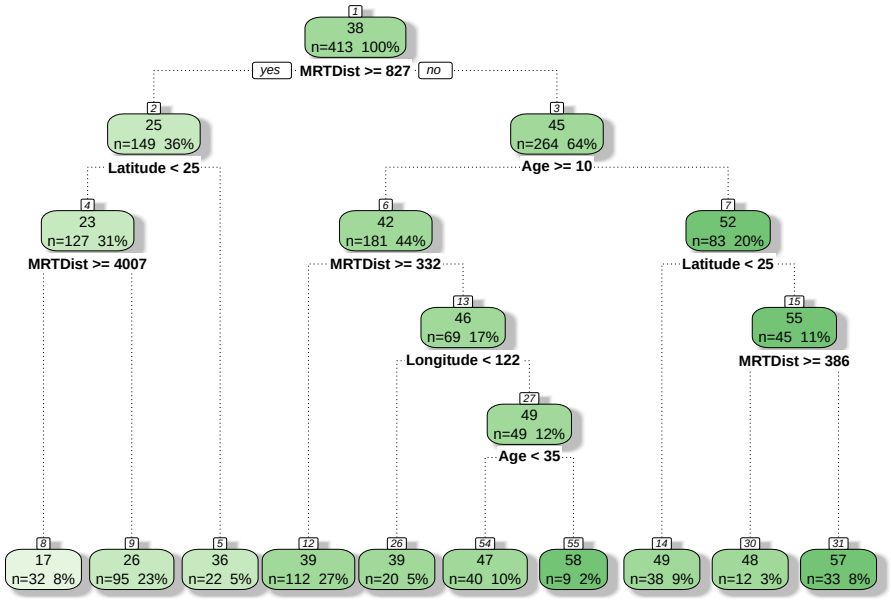

Figure 4: Single regression tree. Split the tree each time only according to the most important variable.

Random Forest

Generally, the regression tree is a process in which we keep making decisions step by step and only based on one variable each time, until we get a group of observations that have the same outcome. Although we are considering all the variables at each split, and we select the most important one, the variables used in the final tree do not necessarily include all the variables in the data.

Random forest, on the other hand, is building multiple regression trees and combining all of them, and makes prediction based on the "average" of the trees. Rather than having a large single tree, random forest is trying to combining a lot of them, but for each tree, we only consider a subset of the variables. For example, in tree 1, we only consider house age, longitude and number of convenience stores. In tree 2, we consider distance to nearest MRT station, house age and latitude, and so on and so forth. The reason to do this is that it helps reduce the correlation between those trees, and by combining all the trees together, we still use all the variables and we can have all the information that is contained in the variables.

However, if in each tree we use all the variables, then those trees can be highly correlated, which means that the information contained in each tree is redundant and thus may incurs large variance in our prediction.

Therefore, random forest not only de-correlates the trees, but also keeps all the information we have, and make prediction by averaging the results of many regression trees. Meanwhile, same as what we see in regression tree, random forest also gives the importance of the variables, as some variables are important in many trees in terms of splitting nodes while some may be trivial.

We quantify the importance and tabulate measure below:

Table 1: Importance measure of variables.

Variable Name	Percent Increase in Mean Squared Error	Increase in Node Purity
Distance to The Nearest MRT Station	23.01%	27281.75
House Age	21.03%	8396.23
Latitude	12.97%	12808.47
Longitude	12.65%	8242.15
Number of Convenience Stores	7.74%	8324.01
Transaction Date	4.06%	2456.13

As we can see in Table 1, distance to the nearest MRT station is very crucial to our prediction. Under same amount of variation(actually permutation), distance to the nearest MRT station causes highest increase in the mean squared error, which is around 23.01%, indicating that the structure of the model relies heavily on this variable.

In other words, an inaccurate input of this variable can cause very large prediction error in the house price estimates, hence users should be careful when measuring this variable for the target house. Besides, house age also plays an important role in our model, which may cause 21.03% increase in mean squared error if the variable is wrongly measured. Latitude and longitude, on the other hand, have the relatively same influence in prediction.

Recall that, in Figure 1 we visualize the distribution of the house price versus location, and from the table we see that indeed the price has no obvious incline in the axis of latitude against the axis of longitude.

Now we plot the results to better illustrate the points. As we can see from figure 5, the distance to the nearest MRT station and house age are twice as important as latitude and longitude, while transaction date seems to be trivial in this case. These conclusion are in line with our visualization.

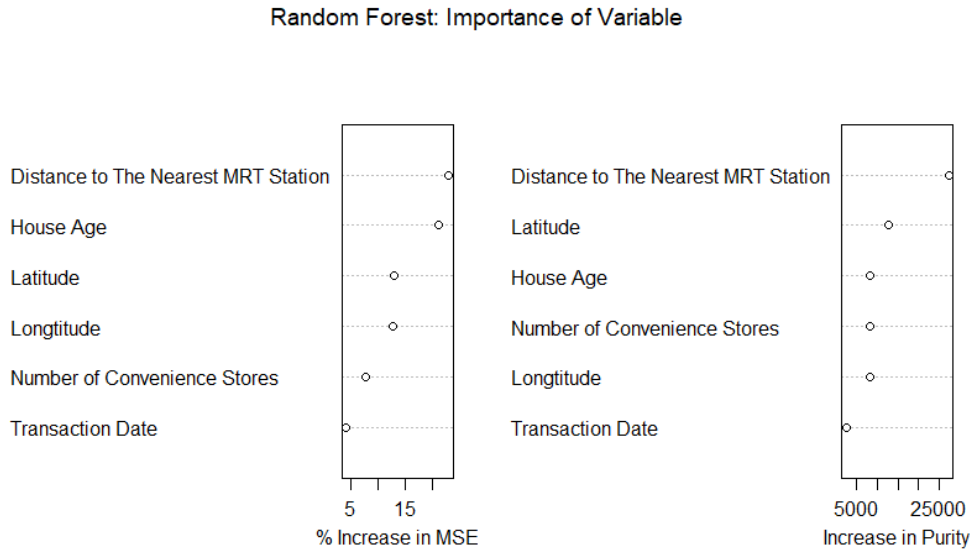

Figure 5: Importance measure of variables.

RMSE of different models

In this section, we shall see that the performance of random forest is way much better than all the other methods we tried. By using a 10-fold cross validation, we calculated the average RMSE of a lot of models with different parameters, and finally we selected the **random forest with each tree considering three variables only** as our best model.

Below we only show the best models related to four different methods and our random forest model. The average root mean squared error of our model is around 6.08, while the benchmark performance, as calculated from linear model, is around 8.03. Therefore, our model improved the prediction performance by 24% compared with linear model, by 13.56% compared with KNN, by 9.08% compared with single regression tree, and by 7.2% compared with GAM using smoothing splines.

A more detailed version of the performance can be found in appendix.

Table 2: RMSE of best model corresponding to each method.

Method	Parameter	CV_RMSE	Improvement
Random Forest	mtry(3)	6.080	-
GAM(Smoothing Spline)	-	6.541	7.20%
single regression tree	cp(0.001)	6.676	9.08%
KNN	K(9)	7.022	13.56%
Linear model	-	8.028	24.39%

Conclusion

The purpose of this analysis is to design a program for clients to predict the house price precisely. As our results point out, we use Random Forest to model the data, and comparing to other possible methods, our model provides best performance in house price estimate. Our model reveals that, the variables can be ranked by importance as Distance to The Nearest MRT Station, House Age, Latitude, Longitude, Number of Convenience Stores and Transaction Date. Distance to The Nearest MRT Station and House Age have dominant position in our model, while Transaction Date seems to be trivial. We believe that our model will accentuate the deviation between the actual and expected prices of the houses concerned. We combined our methods into a function called `HousePrice`, the input of the function should have the same form as our real data, and then our function would return the expected cost of the house in TWD at the time of sale.

Appendix

RMSE

For KNN, we computed RMSEs for k in the range of 5 to 15. The k in KNN specifies the number of observations in the proxy distance that is used to predict the observed value. Based on the results, the smallest RMSE is obtained when k = 9 with a value of 7.02. This is slightly larger than the result we saw in GAM.

When we build one regression tree, we also limit the tree complexity, which is the pruning parameter, to 0.1, 0.01, 0.005, and 0.001. This parameter helps to keep the tree from becoming too complex by stopping the splitting process when further split does not decrease the overall lack of fit by a factor of the four numbers specified. The lowest RMSE of 6.68 is observed when the complexity parameter is set to 0.001.

Given the fact that having only on one tree to predict the house price is not reliable, Random Forest is used to fit multiple regression trees and captured the results from the majority. Since Random Forest also constructs different regression trees, we controlled for the tree complexity by limiting the number of variables available for splitting at each tree node with 3, 4, 5, and 6. The lowest RMSE is 6.07 and is averaged from Random Forest trees constructed with the number of variables available for each split limited at 3. The averaged RMSE obtained from Random Forest is the smallest error out of all 5 models we have tried which suggest the model using Random Forest will be the best in estimating the expected house prices for houses in New Taipei with the smallest deviations of expected prices to the actual prices.

Table 3: RMSE of proposed methods, with different parameters

k	KNN RMSE	cp	TREE RMSE	mtry	Random Forest RMSE
9	7.022	0.001	6.676	3	6.080
8	7.048	0.005	6.825	4	6.102
10	7.086	0.01	6.994	5	6.257
7	7.105	0.1	9.254	6	6.273
5	7.138	-	-	-	-
6	7.148	-	-	-	-
11	7.188	-	-	-	-
12	7.227	-	-	-	-
13	7.299	-	-	-	-
14	7.318	-	-	-	-
15	7.347	-	-	-	-

Codes

```
## Stats 504, F21
## Assignment 4: New Taipei Housing Litigation
## Updated: November, 15, 2021
##
# 79: ------------------------------------------------------------------------
# libraries:
library(caret)
library(tidyverse)
library(gam)
library(rpart)
library(rattle)
library(randomForest)
library(gg3D)
library(plotly)
# directories: ---------------------------------------------------------------
path = './'
# data: ----------------------------------------------------------------------
housing_file = sprintf('%s/Real estate valuation data set.xlsx', path)
housing = readxl::read_xlsx(housing_file)[,-1]
names(housing) = c("Date", "Age", "MRTDist", "StoreNum",
                   "Latitude", "Longitude", "Price")
## ---------------------------------------------------------------------------
## ---------------------------------- EDA  -----------------------------------
## ---------------------------------------------------------------------------
## 3d scatter plot
plot_ly(data = housing,
        x = ~Longitude, y = ~Latitude, z = ~Price,
        type = "scatter3d", mode = "markers", color = -MRTDist)
plot_ly(data = housing,
        x = -Age, y = -MRTDist, z = -Price,
        type = "scatter3d", mode = "markers", color = -StoreNum)
names(housing) = c("Transaction Date", "House Age", "MRT Distance", "Number of Stores",
                   "Latitude", "Longitude", "House Price")
## 2d scatter plot
par(mfrow=c(2,3))
with(data = housing, plot(`Transaction Date`, `House Price`))
with(data = housing, plot(`House Age`, `House Price`))
with(data = housing, plot(`MRT Distance`, `House Price`))
with(data = housing, plot(`Number of Stores`, `House Price`))
with(data = housing, plot(`Latitude`, `House Price`))
with(data = housing, plot(`Longitude`, `House Price`))
names(housing) = c("Date", "Age", "MRTDist", "StoreNum",
                   "Latitude", "Longitude", "Price")
## ---------------------------------------------------------------------------
## -------------------------- Model Selection --------------------------------
## ---------------------------------------------------------------------------
## fix random seed
set.seed(2021504)

## filter out outlier
train = housing %>%
  mutate(StoreNum = as.integer(StoreNum)) %>%
  filter(Price <= 110)

## 10-fold Cross validation
K = 10
fold_size = floor(nrow(train)/K)
results_KNN = tibble(fold = numeric(), k = numeric(), RMSE = numeric())
results_tree = tibble(fold = numeric(), cp = numeric(), RMSE = numeric())
results_rf = tibble(fold = numeric(), mtry = numeric(), RMSE = numeric())
results_GAM = tibble(fold = numeric(), RMSE = numeric())
results_lm = tibble(fold = numeric(), RMSE = numeric())
for(i in 1:K){
    ## iteratively select K-1 folds as training data in CV procedure, remaining
    ## as test data.
    if(i!=K){
      CV_test_id = ((i-1)*fold_size+1):(i*fold_size)
    }else{
      CV_test_id = ((i-1)*fold_size+1):nrow(train)
    }
    CV_train = train[-CV_test_id,]
    CV_test = train[CV_test_id,]
    Test_feature = CV_test[,-7]
    Test_Y = CV_test[,7][[1]]
    ## KNN: ----------------------------------------------------------------
    for(j in 5:15){
    tuneGrid = expand.grid(
      k = j
    )
    MKNN = train(
      Price ~ .,
      data = CV_train,
      method = 'knn',
      preProcess = c("center", "scale"),
      tuneGrid = tuneGrid
    )
    KNN_Pred = predict(MKNN, Test_feature)
    ij_rmse = tibble(fold = i,
                     k = j,
                     RMSE = sqrt(mean((Test_Y - KNN_Pred)^2)))
    results_KNN = results_KNN %>%
      bind_rows(ij_rmse)
    }
    ## Regression Tree: ----------------------------------------------------
    cp_tune = c(1e-1, 1e-2, 5e-3, 1e-3)
    for(j in cp_tune){
      tree_mod = rpart(Price~., data = CV_train, cp = j)
      Tree_pred = predict(tree_mod, newdata = CV_test)
      ij_rmse = tibble(fold = i,
                       cp = j,
                       RMSE = sqrt(mean((Test_Y - Tree_pred)^2)))
      results_tree = results_tree %>%
        bind_rows(ij_rmse)
    }
    ## Random Forest: ------------------------------------------------------
    mtry_tune = c(3,4,5,6)
    for(j in mtry_tune){
      rf_mod = randomForest(Price~., data = CV_train, mtry = j,
                            ntree = 100, importance = TRUE)
      rf_pred = predict(rf_mod, newdata = CV_test)
      ij_rmse = tibble(fold = i,
                       mtry = j,
                       RMSE = sqrt(mean((Test_Y - rf_pred)^2)))
      results_rf = results_rf %>%
        bind_rows(ij_rmse)
    }
    ## GAM: ----------------------------------------------------------------
    MGam = gam(Price ~ Date + s(Age, 6.5) + s(MRTDist, 9.4) + StoreNum +
                 s(Latitude, 7.7) + s(Longitude, 8.9),
               data = CV_train)
    GAM_pred = predict(MGam, newdata = CV_test)
    gam_rmse = tibble(fold = i,
                      RMSE = sqrt(mean((Test_Y - GAM_pred)^2)))
    results_GAM = results_GAM %>%
      bind_rows(gam_rmse)
    ## Linear: -------------------------------------------------------------
    LMod = lm(Price ~ ., data = CV_train)
    Lm_pred = predict(LMod, newdata = CV_test)
    lm_rmse = tibble(fold = i,
                     RMSE = sqrt(mean((Test_Y - Lm_pred)^2)))
    results_lm = results_lm %>%
      bind_rows(lm_rmse)
}
## Find the best model, deciding by RMSE
KNN_best = results_KNN %>%
  group_by(k) %>%
  summarize(CV_RMSE = mean(RMSE)) %>%
  arrange(CV_RMSE) %>%
  .[1,] %>%
  mutate(`Parameter` = "K(9)") %>%
  select(-k)
tree_best = results_tree %>%
  group_by(cp) %>%
  summarize(CV_RMSE = mean(RMSE)) %>%
  arrange(CV_RMSE) %>%
  .[1,] %>%
  mutate(`Parameter` = "cp(0.001)") %>%
  select(-cp)
rf_best = results_rf %>%
  group_by(mtry) %>%
  summarize(CV_RMSE = mean(RMSE)) %>%
  arrange(CV_RMSE) %>%
  .[1,] %>%
  mutate(`Parameter` = "mtry(3)") %>%
  select(-mtry)
GAM_best = results_GAM %>%
  summarize(CV_RMSE = mean(RMSE)) %>%
  arrange(CV_RMSE) %>%
  mutate(`Parameter` = "-")
lm_best = results_lm %>%
  summarize(CV_RMSE = mean(RMSE)) %>%
  arrange(CV_RMSE) %>%
  mutate(`Parameter` = "-")
## tabulate the results
KNN_best %>%
  add_row(tree_best) %>%
  add_row(rf_best) %>%
  add_row(GAM_best) %>%
  add_row(lm_best) %>%
  arrange(CV_RMSE) %>%
  mutate(base = 6.07) %>%
  mutate(pct = (CV_RMSE - base)/CV_RMSE * 100) %>%
  mutate(Method = c("Random Forest", "GAM(Smoothing Spline)",
                    "single regression tree", "KNN", "Linear model"),
         CV_RMSE = sprintf("%.3f", CV_RMSE),
         Improvement = sprintf("%.2f%%", pct)) %>%
  select(Method, Parameter, CV_RMSE, Improvement) %>%
  mutate(Improvement = ifelse(Method == "Random Forest", "-", Improvement)) %>%
  knitr::kable(caption = cap_tab1)

## ---------------------------------------------------------------------------
## -------------------- A look on the single tree ----------------------------
## ---------------------------------------------------------------------------
## fitting a regression tree
tree_mod = rpart(Price~., data = train, cp = 0.01)
fancyRpartPlot(tree_mod, sub = "")
## ---------------------------------------------------------------------------
## -------------------- Final Model: Random Forest ---------------------------
## ---------------------------------------------------------------------------
set.seed(2021504)
## fitting a random forest, with mtry = 3
rf_mod = randomForest(Price ~ ., data = train, mtry = 3,
                      ntree = 100, importance = TRUE)
rownames(rf_mod$importance) = c("Transaction Date", "House Age",
                                "Distance to The Nearest MRT Station",
                                "Number of Convenience Stores", "Latitude",
                                "Longtitude")
## plot importance of variable
importance(rf_mod) %>%
  as_tibble() %>%
  mutate(variable = dimnames(importance(rf_mod))[[1]]) %>%
  relocate(variable) %>%
  arrange(desc(`%IncMSE`)) %>%
  mutate(`%IncMSE` = sprintf('%.2f%%', `%IncMSE`),
         IncNodePurity = sprintf('%.2f', IncNodePurity)) %>%
  rename(`Variable Name` = variable,
         `Percent Increase in Mean Squared Error` = `%IncMSE`,
         `Increase in Node Purity` = IncNodePurity)
colnames(rf_mod$importance) = c("% Increase in MSE",
                                "Increase in Purity")
varImpPlot(rf_mod, main = "Random Forest: Importance of Variable")
HousePrice = function(x){
```

```
  results = predict(rf_mod, newdata = x)
  return(list(`Expected Price` = results, documentation = summary(rf_mod)))
}
## ------------------------------------------------------------------------
## -------------------------------- Appendix ---------------------------------
## ------------------------------------------------------------------------
knn_tab = results_KNN %>%
  group_by(k) %>%
  summarize(`KNN RMSE` = mean(RMSE)) %>%
  arrange(`KNN RMSE`) %>%
  mutate(k = as.character(k), `KNN RMSE` = sprintf("%.3f", `KNN RMSE`))

tree_tab = results_tree %>%
  group_by(cp) %>%
  summarize(`TREE RMSE` = mean(RMSE)) %>%
  arrange(`TREE RMSE`) %>%
  mutate(cp = as.character(cp), `TREE RMSE` = sprintf("%.3f", `TREE RMSE`)) %>%
  bind_rows(tibble(cp = rep("-", 7), `TREE RMSE` = rep("-", 7)))

rf_tab = results_rf %>%
  group_by(mtry) %>%
  summarize(`Random Forest RMSE` = mean(RMSE)) %>%
  arrange(`Random Forest RMSE`) %>%
  mutate(mtry = as.character(mtry),
         `Random Forest RMSE` = sprintf("%.3f", `Random Forest RMSE`)) %>%
  bind_rows(tibble(mtry = rep("-", 7), `Random Forest RMSE` = rep("-", 7)))

knn_tab %>%
  bind_cols(tree_tab) %>%
  bind_cols(rf_tab)
```