Capstone Project

Yusuke Yamamoto

Introduction

In this report, we are going to analyse the housing price in Tokyo in Japan. In the course work, we looked at the Boston housing data set where we built a model using the number of rooms in home, neighbourhood poverty level and student-teacher ratio of nearby schools. The goal of this project is to understand whether the housing market in a different city has different dynamics where other factors determine the housing price. Predicting the housing price is important in many respects. One is to understand the economy of a country. It is imperative for policy makers to understand if the housing price is so high that the county is facing the bubble economy.
Another aspect is personal. Buying a house is an important decision for anyone because the price is high and most people need to apply for a mortgage. When it comes to selling a house, you also want to know if you are selling it at a fair price. Understanding the fair market value of a house you are trying to buy or sell is of significant interest to anyone.

The price of houses in Japan is said to be correlated mainly with the age of the house. Some areas are popular so the price tends to remain the same. People in Tokyo normally commute by train so which train lines are close to the house should also determined the house price. The location is also relevant, and it is often reported that the housing price in the west part of Tokyo tends to be higher than that in the east part. Some areas are known to be popular for well-off people. Another factor to consider is whether it is a house or apartment. Because the land is limited in the centre of Tokyo, people normally buy apartments. People who want to buy a house go to a suburb of Tokyo. The price of a house includes the price of its building and the land, so the pricing mechanism is considered to be different from that for apartments.

The problem we are trying to solve in this report is to build a regression model that can accurately predict the price of apartments in Tokyo. The model is to predict the price from different characteristics of the apartments such as size, age and location. The strategy to build the model is to use a machine learning technique so that the model can learn the complexity of the pricing mechanism using the combinations of the variables. This will help us automatically identify important variables to predict the price rather than we manually select which variables to include into the model by means of visual inspections or looking at correlation between variables. In order to understand the complexity of the pricing mechanism, we will use Random Forest, an ensemble method that builds different trees and then takes the average of the results of each tree. Because Random Forest builds many regression models with different weights and different variables, we can expect it performs better than a simple linear regression model.

The accuracy of the model is quantifiable by comparing the prediction and the actual price. Mean Squared Error (MSE) is used for the evaluation metric. This is a regression problem where we predict the price of apartments, and the predicted values take non-negative real numbers. R2 can be an alternative metric. In theory, it takes values between 0 and 1, and the close the value is to 1, the better the model is. The problem is that this holds only when we calculate R2 for the same data set that we use to train the model. Another problem is that R2 changes when you change the number of variables in the model, so you cannot compare two models that have different number of variables. For these reasons, we prefer MSE to R2.

Datasets and Inputs

We use data from Ministry of Land, Infrastructure, Transport and Tourism (MLIT) which cover the price of houses and apartments in Japan (http://www.land.mlit.go.jp/webland/servlet/MainServlet). The price is rounded to the nearest one million yen (approximately 10,000 USD). Other variables such as age, month of the transaction and the size in square meters are obfuscated. This is due to the privacy reason so that the users of the system cannot tell the exact price or location of each house and apartment.
The data have 28 features. MLIT web page has a data set for each quarter, and there are about 3500 records for the price of apartment in Tokyo per quarter. The price of apartment in Tokyo is used in the project.

We build a supervised model that can predict the price of apartments using the characteristics of the apartments such as age and location. To build a model, we use data in Q2 2015 through Q1 2016 as the training data set, Q2 2016 as the validation data set and Q3 2016 as the test data set.

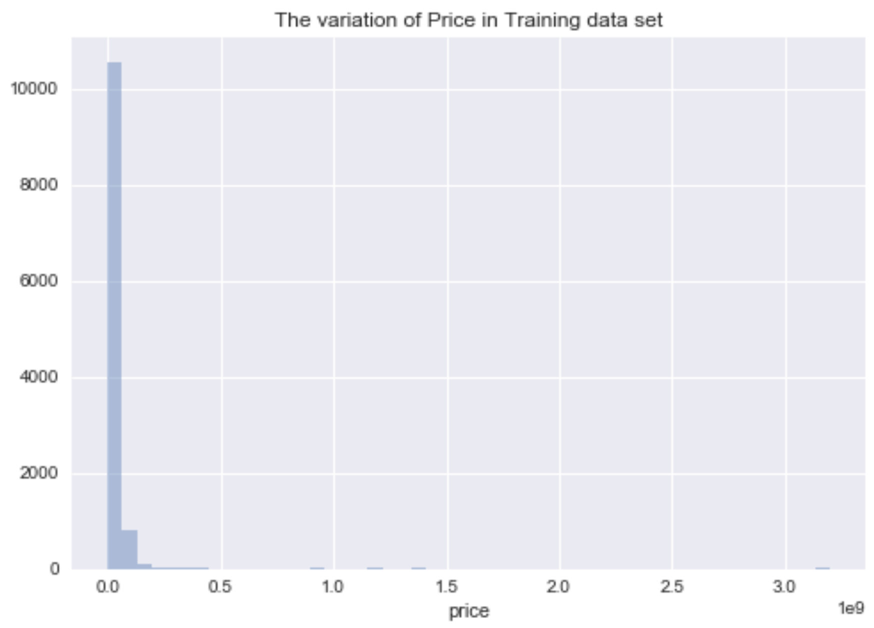
Analysis

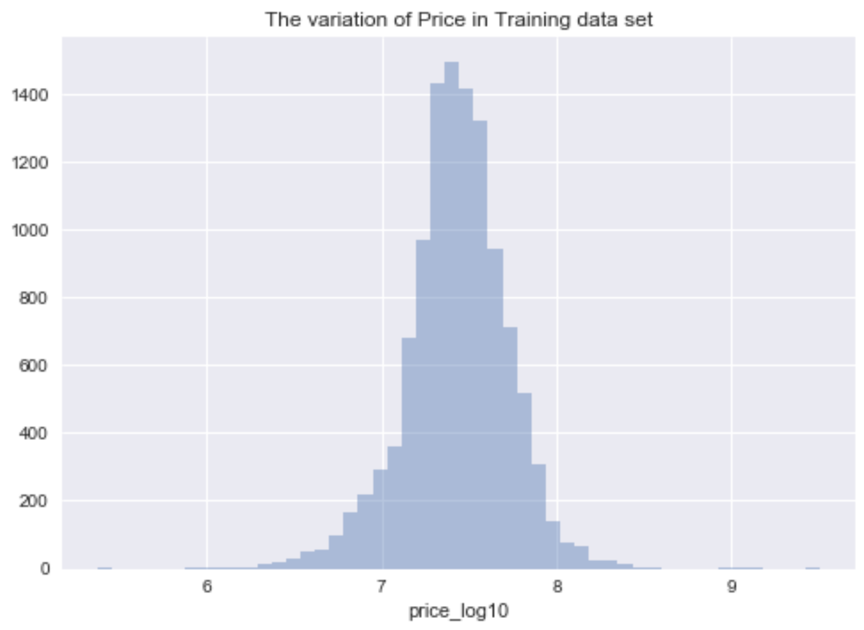
We first look at the distribution of the data set. This is to understand how price is distributed, what other variables seem to be correlated in what way, and what variables appear to be useful to predict the price. In the training data set, the price distribution is summarized in a table below. The average price is 33.6 million JPY (310K USD). 50% of the records take values between 18 million JPY and 40 million JPY.

Min	25%	Median	Mean	75%	Max
0.24 M	18 M	27 M	33.6 M	40 M	3,200 M

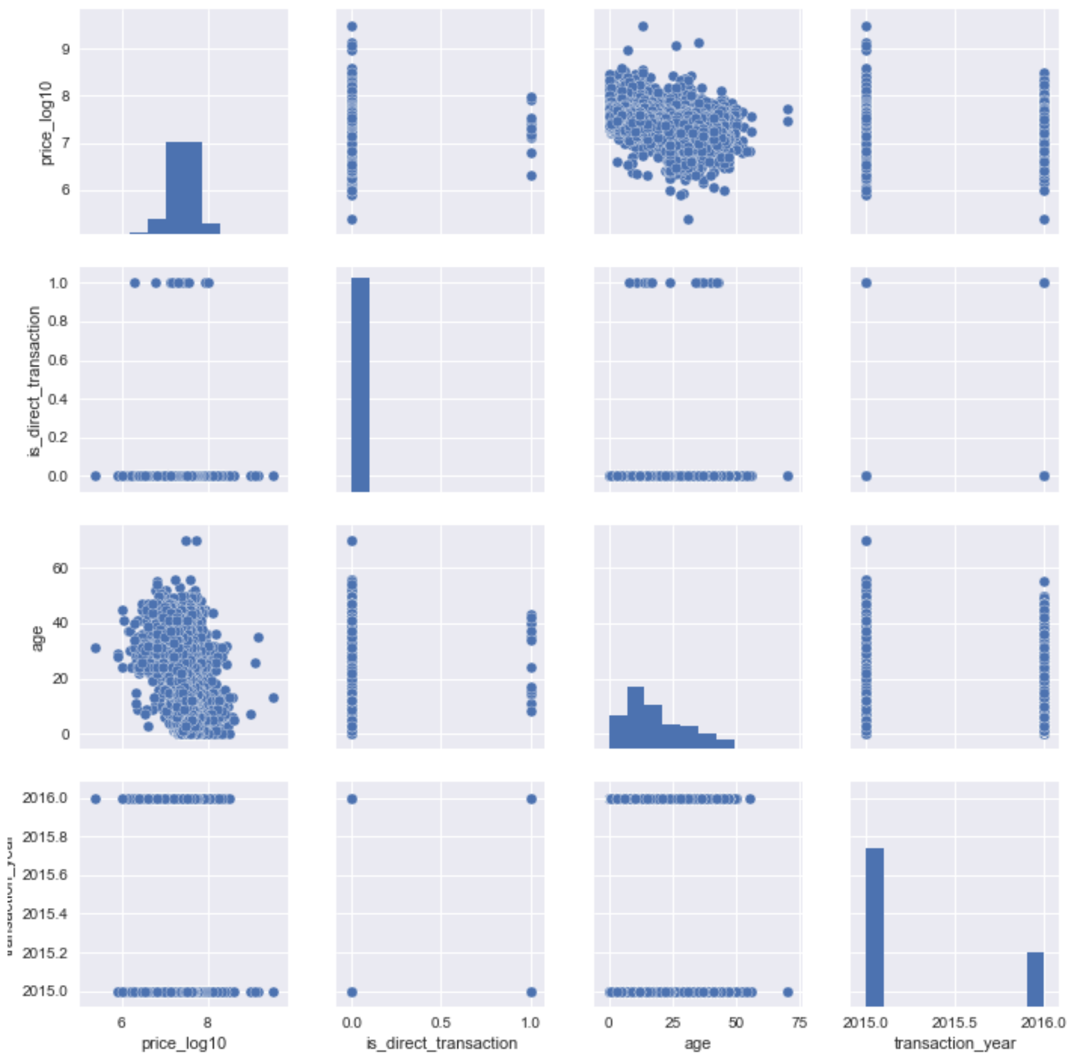
The histogram of price shows very skewed distribution. Almost all apartments are priced below 50 million JPY (460,000 USD), and you can observe some expensive apartments that cost well over 100 million JPY (920,000 USD).

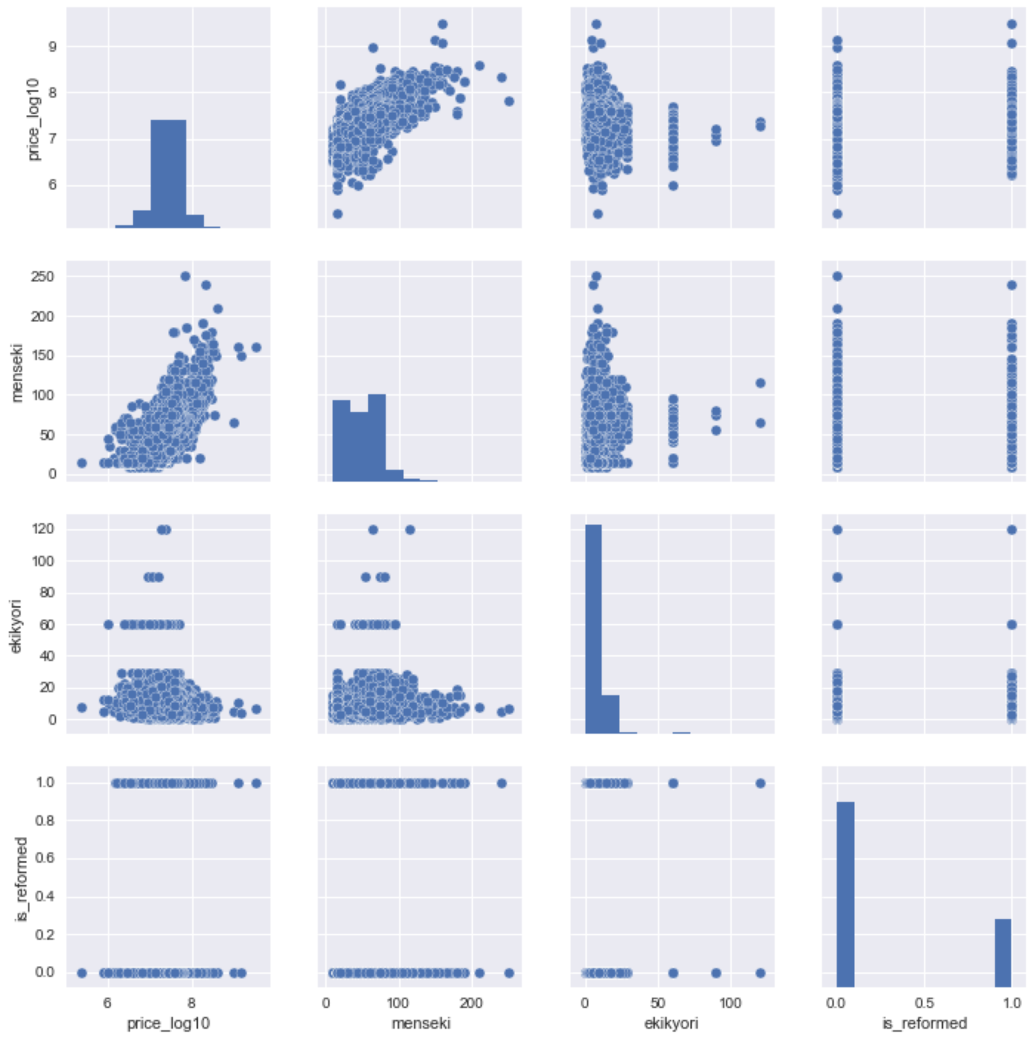
The skewed distribution of price can be difficult to model. Taking the log of price in base 10 makes the distribution almost symmetric.





We also look at the relationships of log price with other variables. Graphs below show the histogram of each variable, and the scatter plot of every combination of two variables. Most variables do not show an apparent correlation. The size of the room (menseki) appears to be positively correlated as the scatter plot shows so that the larger room costs higher, which is reasonable.





A positive correlation can also be observed between price and age. The older the room is, the lower the price is. The visual inspection does not show other correlations.

Data Preprocessing

From the original data set, we included records for apartments only that were sold between Q2 in 2015 and Q3 in 2016. The data between Q2 in 2015 and Q1 in 2016 were used as the training data, the data that were sold in Q2 in 2016 were used as the validation data, and the data in Q3 2016 were used as the test data set.
The distance between an apartment to the nearest station in minutes is available in the data set. Some records show only approximate distance such as 30 minutes - 60 minutes, 1 hour - 1.5 hours, 1.5 hours - 2 hours and more than 2 hours. These were replaced with 60 minutes, 90 minutes, 120 minutes and 120 minutes respectively.

Building-to-land ratio and floor-area ratios were denoted in percentages. These ratios were conveyed into decimal points.
In order to calculate the age of each apartment as of the transaction date, we calculated it from the different between the year it was build and the transaction year. Some records have the building year before 1945, and these were all treated as being built in 1945.
The data included character variables, such as town's name, area's name, the name of the nearest station, material of the apartment, zoning. We converted these into dummy variables so that each column has either 0 or zero entries. Regarding the names of the station and area's name, there are just too many types in these variables, so we did not use these variables. In order to avoid the model identifiability problem, we dropped one level from each dummy variable. In the end, we have the following dummy variables.

- town: indicates where the apartment is located in Tokyo.
- material: what the apartment building is made of; reinforced concrete (RC), steel reinforced concrete (SRC) and steel.
- purpose: how the room is used for; housing, office or other.
- zoning: indicates how certain land uses are permitted or prohibited such as residential and commercial. https://en.wikipedia.org/wiki/Zoning#Japan

Finally, any rows that has missing values for at least one variable were excluded. The number of records for the training, validation and testing reduced to 11479, 2895 and 1880 respectively from 14733, 3664 and 2464. There are 77 predicting variables.

Benchmark Model

We use a simple linear regression model using all the features in the data set as the benchmark model. The linear regression model tries to find the best linear combination of each variable by adjusting weights of each variable. A model looks like something like below:
$y = \beta_0 + \beta_1 \times x_1 + ... + \beta_m \times x_m$

The linear regression tries to minimize the sum of squared errors between the actual values and predicted values that are represented by the linear formula.
The target variable is the log transformed price, so the model can be written as follows:
$log(Price) = \beta_0 + \beta_1 \times x_1 + ... + \beta_m \times x_m$

The performance is measured against the validation data set. The table below shows the performance of the linear model.

	Training MSE	Validation MSE
Linear Model	0.0150	0.0196

The benchmark of the prediction model is 0.0196 for the validation data set.

Model Evaluation and Validation

We build a model using Random Forest. The method makes a collection of regression trees and takes the average of the estimate.
Pros of the method is that it can fit a data set better that has a complex distribution. For example, the linear model fits data well if the relationship between the target variable and the predicting variables is linear. The random forest model can perform well even if the relationship is non-linear.
Cons of the method is that it is computationally expensive. The random forest randomly samples the data set and build a regression tree using a subset of predicting variables that are also randomly selected. It then repeats this process to make a number of regression trees to take the average. Because of this algorithm, it does not have a closed analytical form to solve a model unlike the linear regression and takes long to find the solution.
Another disadvantage is that it has hyper-parameters to be tuned, and it can perform poorly if the hyper-parameters are not correctly tuned. This adds another complexity to the model, and you need to try different sets of hyper-parameters to see which combination works the best. One way to tune the hyper-parameter is the grid search method where we try out every possible combination of hyper-parameters and calculate the performance of each combination. The problem of this approach is the number of combinations increases exponentially as we add more hyper-parameters. For example, when we try five values of three hyper-parameters, the number of combinations is five to the power of three, which is 125. If we want to add one more hyper-parameter and try five different values of this hyper-parameter, the number of combinations increases to 625.
To avoid this problem we tuned the hyper-parameters by using randomized search in this project. Instead of trying out all parameter values, a fixed number of parameter values is sampled from the specified distributions. We tuned the following hyper-parameters using five-fold cross validation.

Name of Hyper-parameters	Explanation of the hyper-parameter and how it affects the performance.
n_estimators	The number of regression trees. Increasing the number of n_estimators normally improves the performance at the cost of computational resources.
max_depth	The maximum depth of the tree. If it is set high, it can learn the details of the data and can perform better, but it can also overfit if it is set too high.
max_features	The number of predicting variables to be included in one regression tree. Higher number of max_features generally improves the performance, but too high number can decrease the diversity of each regression tree to the detriment of the overall performance.

Name of Hyper-parameters	Explanation of the hyper-parameter and how it affects the performance.
min_samples_split	The minimum number of samples required to split an internal node of a tree. If this number is set low, the model can learn the details of the data, but can overfit.
min_samples_leaf	The minimum number of samples required to be at a leaf node, which is the end node of a regression tree. If it is set small, it can overfit.
bootstrap	whether bootstrap sampling is used or not.

The randomized search found the hyper-parameters as follows.

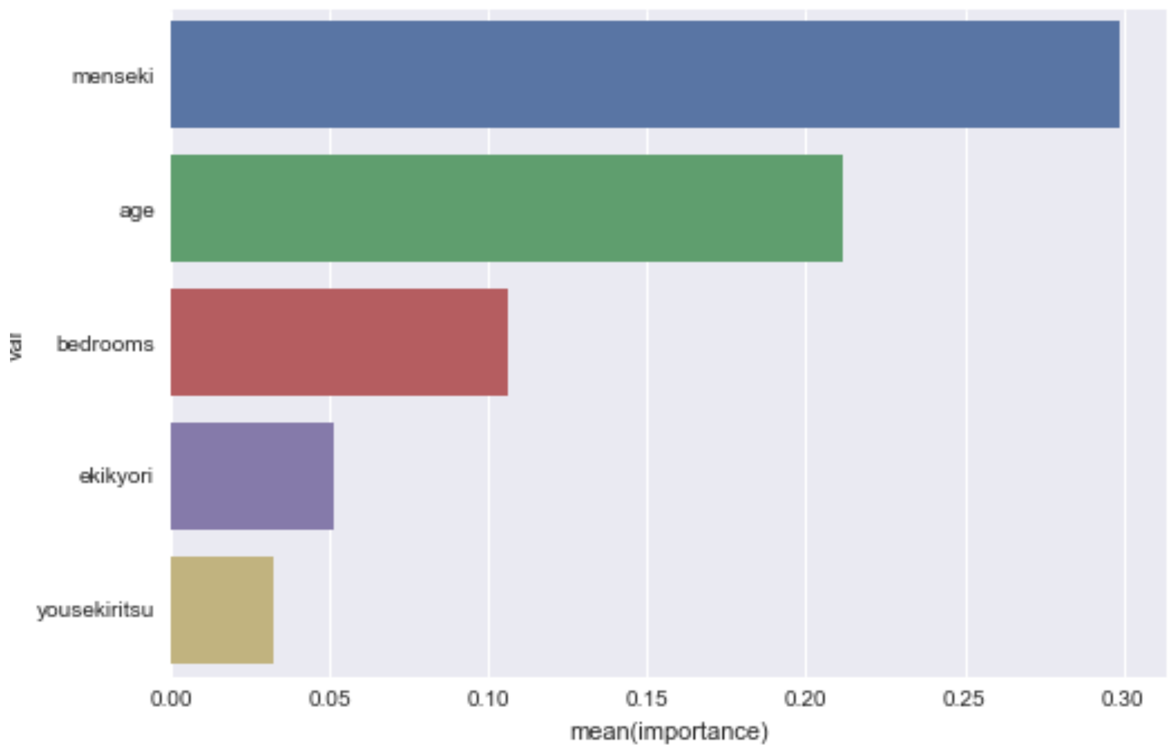
Name of Hyper-parameters	
n_estimators	500
max_depth	96
max_features	8
min_samples_split	8
min_samples_leaf	1
bootstrap	FALSE

Using this combination of hyper-parameters, the tuned model was fit into the training data set, the validation data set and the test data set with the following result.

	Training MSE	Validation MSE	Testing MSE
Tune Random Forest Model	0.0042	0.0138	0.0154

As the table above shows, the training MSE dropped significantly from 0.015 of the benchmark to 0.0042. The validation MSE also dropped from 0.0196 to 0.0138, which is about 30% (=1 - 0.0138/0.0196) improvement from the benchmark model, and we choose this as the final model. One thing to notice is that the validation MSE is nearly 10 times as high as the training MSE. The difference between the training and the validation MSE in the benchmark linear model was small (0.015 vs 0.0196), so the gap between the training and the validation performance became wider.

In order to test the robustness of the model, we also check if the model is working as expected. First we checked whether variables that seem to be correlated as we found in the exploratory visualization are included in the model. The graph below shows the feature importances in the final random forest model.
The first item is “menseki” (area of the apartment in square meters). “Age” measures how old the apartment was as of the transaction date, “bedrooms” counts the number of rooms, “ekikyori” measures the walking distance from the nearest station in minutes, and “yousekiritsu” is Floor Area Ratio, which is the ratio of a building's total floor area to



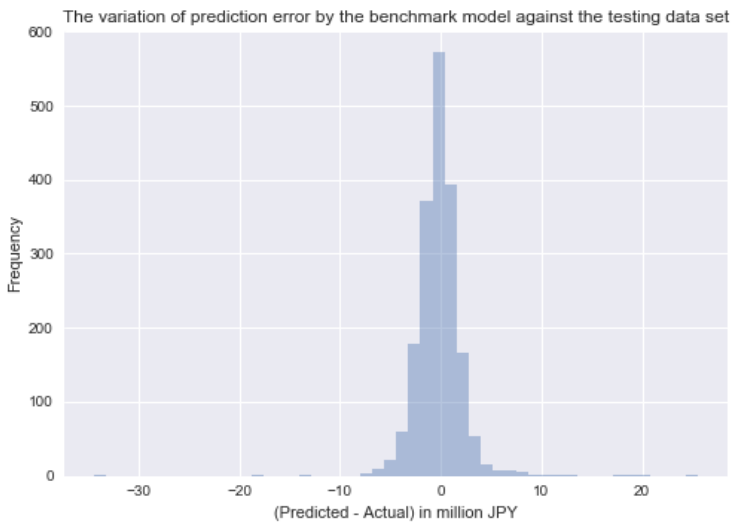
the size of the piece of land upon which it is built. As we saw in the scatter plots between the log price and variables in the exploratory visualizations, the area and the age are correlated with the price. The model correctly captures this correlation, and these variables are regarded as important features in the random forest model. Second, we checked whether the model behaves as we expected. As we saw in the exploratory visualisation, the age of the apartment negatively correlate with the price, for example. We can check if the model predicts higher price for a new apartment and lower price for an old apartment by changing age variable while the other variables fixed. We used one record from the test data set, and changed the top five important variables while the other variables fixed. The result is summarized in a table below.

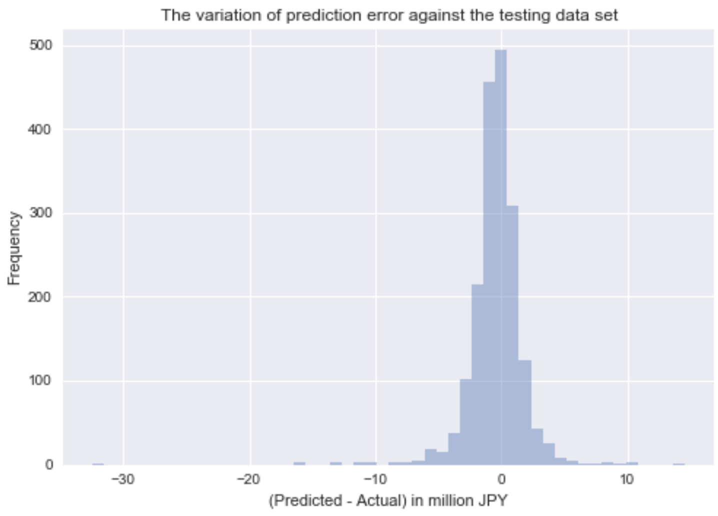
	Original Value	Changed Value	Predicted Price for Original Value	Predicted Price for changed values
Area	65.0	75.0	26,385,788	27,402,395
Age	9	15	26,385,788	24,488,223
Bedrooms	2	3	26,385,788	25,701,212
Walking Distance	4	10	26,385,788	24,205,063
Floor Area Ratio	500%	200%	26,385,788	22,416,153

When we increased the value of the area from 65 to 75 while fixing the other variables, the predicted price increased from 26,385,788 to 27,402,395. It means the model predicts higher price for a larger room, which is reasonable. The model appears to behave reasonably for the other variables too. As the age increase from 9 years to 15 years (i.e. becoming older), the model predicts a fall in price. As the walking distance from the nearest station increases from 4 minutes to 10 minutes (becoming less convenient for commuters), the predicted price falls. When we decreased Floor Area Ratio from 500% to 200% (the size of the building is smaller), the predicted price falls. An exception is Bedrooms. As the number of bedrooms increases, we would expect the higher price. The model predicts otherwise, and the predicted price falls. This may be due to a correlation between Area and Bedrooms. We can expect a positive correlation between the area and the number of bedrooms; as the area increases, we would expect the number of bedrooms is larger. Except for one variable, the model appears to work as we expect.

Model Justification

As we discussed in the previous chapter, MSE improved from the benchmark model. We also looked at the error distributions of the final model and the benchmark model. Histograms below show distributions of the difference between the predicted price and the actual price of every record in the testing data set.

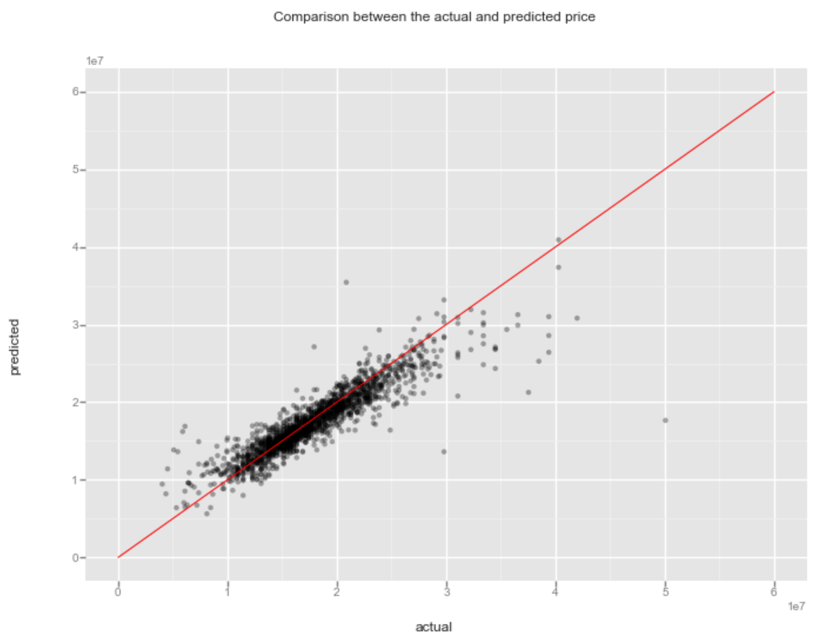




The graph on the left shows the distribution of the errors by the benchmark model, and the graph on the right shows the distribution of the error by the final random forest model. The error is calculated by (predicted price) - (actual price), so the error is negative when the model underestimates and positive when it overestimates. The shape of the distributions looks quite similar to each other. One difference can be observed in the upper tail of the distribution. In the benchmark model, we can see some observations exceeding 20M JPY. In the final model, these overestimated records improved, which is reflected in the smaller MSE.

Conclusion

We built a Random Forest model to predict the apartment price in Tokyo area that were sold between Q2 2015 through Q1 2016 and tested it on the data that were sold in Q3 2016. The model predicts the log-transformed apartment price, and we were able to achieve the better performance than the benchmark model.



A graph above shows the scatter plot between the actual (x-axis) and the predicted values for the testing data set. The red diagonal line means points where the predicted price and the actual price are equal. The graphs shows most points are scattered along the diagonal line, and it means the model predicted the price correctly for most cases. We can observe some points that are far from the diagonal line, which means the model predicted poorly for those points.
One thing to note is that we were not able to get the same level of improvement in the validation data set as we get in the training data set.
One possible reason is that the structure of the housing market is completely different from quarter to quarter. For example, the average market price in Q2 2016 may be higher than that in Q1 2016. The model does not have variables that are related the such market trend.
Another possibility is that the RF model may not be suitable to predict the apartment price. We did not test other models such as XGBoost or SVM. Correctly tuned models using those methods may have given rise to better performance.
Regarding some outliers where the model predicted the price poorly, there may be a data issue. One data point shows underestimate by more than 30M JPY (approximately 300K USD). It has price over 300M JPY (1M USD), even though it is nearly 40-year-old apartment with area of only 50 square meters. It is suspected that the true price may be 30M JPY rather than 300M JPY. We could remove the record as an outlier, but we do not know for sure if this is caused by an input error. It could be a famous apartment

designed by a famous architect. This is one of the challenging aspects of building a model for real estate. We would need a mechanism to correctly ignore records that are apparently made incorrectly such as mistyping price by adding one more zero.