Predicting House Prices in Toronto: A Machine Learning Approach

Albina Cako, BSc *York University, Certificate in Machine Learning*
Colin Green, BSc *York University, Certificate in Machine Learning*
Lucy Zhang, BSc *York University, Certificate in Machine Learning*
Sean X. Zhang, MSc *York University, Certificate in Machine Learning*

The project focus on building a machine learning model for predicting house price in Toronto. Although House Price Index (HPI) is commonly used for estimating the changes in housing price but it is not perfect for individual housing price prediction due to high correlation of housing price and other factors such as house location, income distribution, area, and etc. This project evaluates four different model and chooses the best one for deployment with ShinyApp. The app created for both buyer and seller to have an estimated house price due the location and different attributes of the house, and help users to make decisions.

Keywords: house prices, machine learning, caret, shiny

Introduction

The House Pricing Prediction app is created for estimate the house price for both buyer and seller based on different factors such as total Sqft, house locations, etc. The deployment was constructed using ShinyApp. An user friendly app for both buyer and seller, with simple click of factors users will get an estimated housing price. The app can be used for individual buyers who want to know the final price of the houses they are interested or for individual sellers to know what is the best listing price. The app uses regression model for prediction, which was trained by the data set of Toronto housing price. The housing price is strongly correlated with other factors, for increasing the model accuracy decreasing errors, it is important to try different factors and combinations. This project will comprehensively validate four different models: decision tree, random forest, K nearest neighbors, and gradient boosting machine. This report will go through data analysis, modeling implementation and provide an optimistic result for housing price prediction.

Objective

The objective of this project was to evaluate the application of machine learning algorithms to predict house prices in the Greater Toronto Area, and apply

Background

Purchasing a house is a big life decision for every individual and needs a considerable amount of research. Everyone has different purpose of buying houses, someone would prefer by the house at the best rate for living now, someone would buy houses for future investment. Selling the houses is also very important and needs to do research and decide what is the best leasing price. Commonly, people will ask advise from various websites, real estate agents or realtors before purchasing or leasing; However, due to the trend towards big data, house pricing prediction can be done by using machine learning strategies base on large amount of data from previous years more correctly. House Price Index (HPI) can measure the price changes of residential housing as a percentage change, In Canada the new Housing Price Index is calculated monthly by Statistics Canada. HPI is useful but because it is a rough indicator calculated from all transactions, it is inefficient for predicting a specific house with its attributes. The purpose of this project is to create an app for both buyers and sellers can easily check the predicted list price or final price based on the attributes of the house such as locations, square foot, number of bedrooms, etc.

Methodology

Data Preprocessing

The housing dataset, originally shared on Github[1], was extracted from Zoocasa.com in the summer of 2019. The dataset contained all completed property sales in the city of Toronto within a 1-year span. We performed several data exploration and cleaning steps to prepare this data for modeling.

Missingness

We assessed the dataset for missing values. Many parametric machine learning models do not accept missing data, and the accuracy of even non-parametric models are often negatively impacted by missingness. Thus, missing values ought to be either imputed or removed before data modeling. We then determined whether missing data was Missing Completely at Random (MCAR), Missing at Random (MAR), or Missing Not at Random (MNAR). Should the data be MCAR, then it is acceptable to simply remove each observation that is missing, as doing so would not introduce bias to the remaining observations. However, if there was a correlation between missingness and other data features, then imputation must be performed. Missingness correlation was assessed using the missing_compare() function from the finalfit library, which applies the Kruskal Wallis correlation test for numerical variables and Chi-squared correlation test for categorical variables to determine correlation. Using the MICE package in R, we then applied the following imputation methods: 1) simple, which imputes a value from a simple random sample from the rest of the data; 2) mean, which imputes the average of all observations; 3) random forest, which applies a random forest algorithm; and 4) CART, which imputes by classification and regression trees. The distribution of the imputed data were evaluated with a density plot and an imputed dataset was chosen based on best fit.

Assessing Parametric Fit

Outliers were visualized with the boxplot() function. Data were considered outliers if they were less than Q1 - 1.5 X Inter-Quartile Range and greater Q3 + 1.5 X Inter-Quartile Range. Normality of the distribution of variables were visualized with density plots. A correlogram with Pearson's R determined collinearity. Linear relationship between outcome variable and predictors was tested via scatterplots.

Data Curation

The following variables were removed as they did not have any data utility or were not easily parseable (i.e. free text): title, description, mls, type, full_link, full_address. A numeric 'bedrooms' column was created by combining bedrooms_ag and bedrooms_bg. We also removed district_code and city_district. Both were categorical variables with number of factors = 140; keeping these would significantly increase model training time. We also did not consider longitude and latitude, as including these variables in training sets would have required geocoding and district clustering; complexities which were outside our scope for this application. Mean_district_income was left as an approximation of the effect of districts on property price. After consultation with a real-estate expert, we decreased the number of property types by generalizing types to: Townhouse, Condo, Detached, Semi-Detached, and Plex. Thus, the predictors chosen were:

Table 1: Predictor variables

Label	Description
sqft	numeric
beds	numeric
bathrooms	numeric
parking	numeric
mean_district_income	numeric
type	categorical

We chose final_price as the target variable. While the dataset also contained a list_price variable. rather than training two models to predict on both list and final price, the predicted list price was instead approximated by a linear equation between list and final price from the original dataset.

Modeling

The data contained a mix of categorical and numerical variables. These variables did not satisfy the many requirements of parametric models, such as variable independence, normally distributed data, and linear relationship with outcome. Thus, several non-parametric models were used instead. We trained four different models using k-fold cross validation. The models were then tuned using various grid searches to improve the accuracy. The final model was then chosen based on three accuracy metrics: Root Mean-Squared Error (RMSE), Pearson correlation (R^2), and Mean Average Error (MAE).

Table 2: Non-parametric Models Used

Model	Description	Tuning parameters
Decision Tree	Decision trees repeatedly partition data at specific nodes until the data at the bottom of each branch (known as a leaf) is as homogenous as possible. The model increases in complexity with each additional partition and subsequently becomes more accurate.	cp (complexity)
Random Forest	Random Forest Model Random Forests are an ensemble learning method for classification and regression. This method will construct a multitude of decision trees and output the mean/average prediction problem for regression or the classes for classification problem. The algorithm can control the number of variable available for splitting at each tree or the number of trees to get a higher accuracy.	ntree, mtry

Model	Description	Tuning parameters
Gradient Boosting Machines	Gradient Boosting Machines (gbm) begin with creating a preliminary 'weak learner' decision tree, then sequentially grows more trees that aim to reduce the error of the last one. The algorithm optimizes the loss function by minimizing the residuals at each iteration (difference between predicted and actual value).	n.trees, shrinkage, interaction.depth, n.minobssinnode
XGBoost	XGBoost uses ensemble learning, which is a systematic solution that combines the predictive power of multiple learners. It outputs a single model that gives the combined output from many models. This allows the opportunity to not rely on the results of a single machine learning model. In this particular model, the trees are built sequentially, such that the next tree focuses on reducing the errors of the previous tree.	nrounds, max_depth, eta, gamma, colsample_bytree, min_child_weight, subsample

Deployment

The application was created using R shiny and hosted on the Shinyapps.io cloud. EDIT : this could probably all go to results section –> The user interface (UI) contains a map of Toronto for geographic navigation and also allows the user to select various inputs to predict property price. While the user would choose a district of interest from the front end, the back end links the district chosen with income and uses mean_district_income as the model input instead. We chose INSERT_MODEL HERE, since it was the most accurate model as the back-end for our application.

Results

The original housing dataset contained 21 variables and 15234 observations. Table 1 defines each variable of the dataset. Figure 1 plots the geographic distribution of properties based on longitude and latitude.

Table 3: Data Dictionary

Variable	Type
title	Title of the listing
final_price	Final price of the property
list_price	Listing price of the property
bedrooms	Number of bedrooms
bathrooms	Number of bathrooms
sqft	Area of property in square feet
parking	Number of parking spaces
description	Verbatim text description of the property
mls	MLS Listing ID
type	Property type
full_link	URL to listing
full_address	Full address of the property
lat	Latitude

Variable	Type
long	Longitude
city_district	Toronto district to which property belonged to
mean_district_income	Average household income of district
district_code	Numerical code of the district
final_price_transformed	Box-Cox transformation of final price
final_price_log	Log transformation of final price
bedrooms_ag	Number of bedrooms above ground
bedrooms_bg	Number of bedrooms below ground

Data Exploration

The mean final price property price in Toronto between 2018 and 2019 was $715,000, with a median number of 3 bedrooms and 2 bathrooms. The most common properties types were condo (58.2%), detached (28.8%), and semi-detached (9.4%).

```
##   final_price          bedrooms        bathrooms          sqft
##  Min.   :  103000   Min.   : 0.000   Min.   : 1.000   Min.   : 250
##  1st Qu.:  535000   1st Qu.: 2.000   1st Qu.: 1.000   1st Qu.: 650
##  Median :  715000   Median : 3.000   Median : 2.000   Median : 900
##  Mean   :  882714   Mean   : 2.875   Mean   : 2.122   Mean   :1116
##  3rd Qu.:  989000   3rd Qu.: 4.000   3rd Qu.: 3.000   3rd Qu.:1300
##  Max.   :13180000   Max.   :14.000   Max.   :14.000   Max.   :4374
##                                                       NA's   :4521
##     parking       mean_district_income
##  Min.   : 0.000   Min.   : 25989
##  1st Qu.: 1.000   1st Qu.: 34904
##  Median : 1.000   Median : 50580
##  Mean   : 1.559   Mean   : 56066
##  3rd Qu.: 2.000   3rd Qu.: 67757
##  Max.   :11.000   Max.   :308010
##

##
##        Condo     Detached          Plex Semi-Detached     Townhouse
##         8874         4388            70          1435           467
```

The 'sqft' variable was missing 1435 observations (approx. 30%). To test whether the missing data was MCAR, we compared whether other variables were associated with missingness. Using Chi-squared and Kruskal Wallis tests for categorical and numerical variables, respectively, we determined that housing and price had significant correlation ($p<0.01$) with sqft missingness, where properties missing sqft trended towards higher prices. Thus,we conclude that sqft is not MCAR, so we cannot simply remove observations where sqft is missing without introducing bias. To account for missing values, we chose to use the CART (Classification and Regression Trees) method of imputation (Figure 1). Blue represents the distribution of the original data, while red represents the distribution of imputed data. The mean sqft increased from 1116 to 1311 as a result of the imputation.

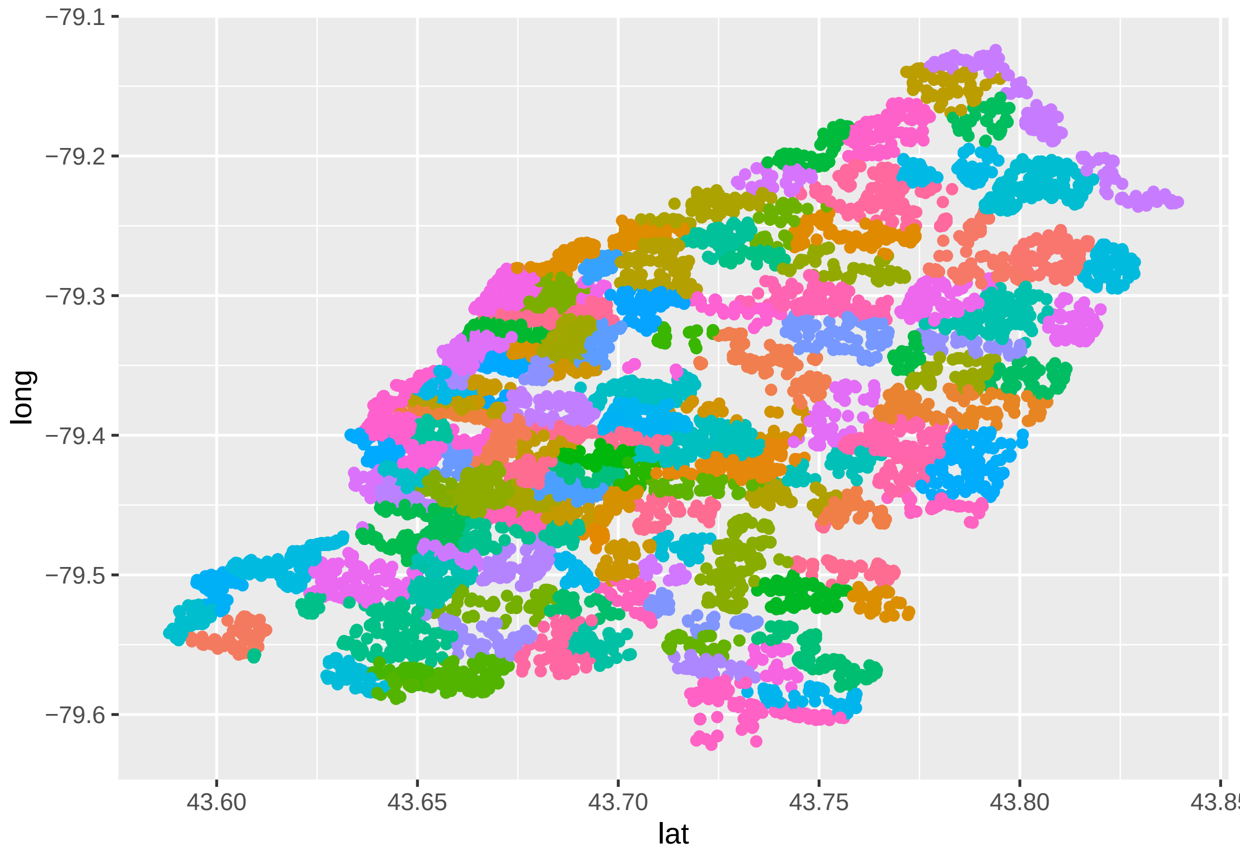

Figure 1: Latitude and Longitude distribution of Property Data

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##     250     750    1100    1311    1750    4374
```

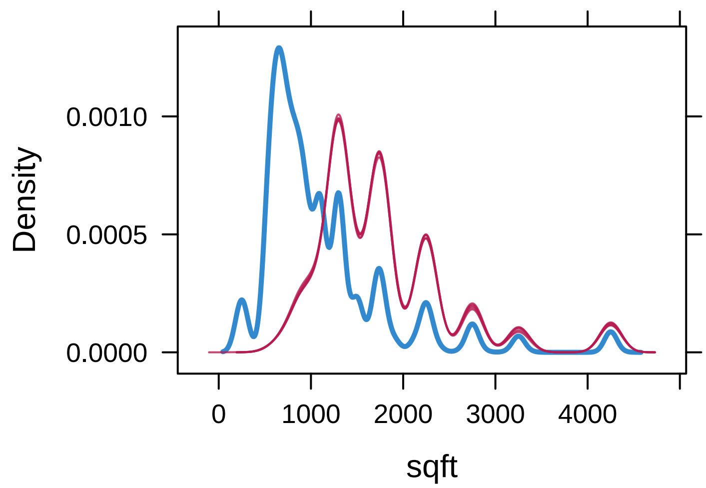

Figure 2: CART-imputed values for sqft

We then performed a correlation analysis based on Pearson's coefficient between each numeric predictor. We considered a correlation > 0.5, with $p < 0.05$ as a significant correlation. Figure 2 demonstrates significant correlation between many of our predictor variables.

We also summarize the distribution of predictor and target variables (Figure 3). Note the right skew in each predictor variable as well as amount of outliers in each of our predictor variables. Finally, we checked for whether a linear relationship existed between each predictor variables and the target variable (Figure 4). Overall, the data is unlikely to be well-fit for parametric machine-learning algorithms such as generalized linear regression. The data does not satisfy the assumptions of variable independence, normal distribution, nor a linear relationship between predictors and target variable. Thus, we chose to use non-parametric algorithms to model our data instead.

We chose final_price as our target variable, but still wished to include list_price in the deployment of our application. Therefore, the list price is predicted via the following linear regression: list_price = 1.03 * final_price - 36525.78.

1. Spirin, S. (2020). Slavaspirin/toronto-housing-price-prediction Available at: https://github.com/slavaspirin/Toronto-housing-price-prediction [Accessed October 26, 2020].

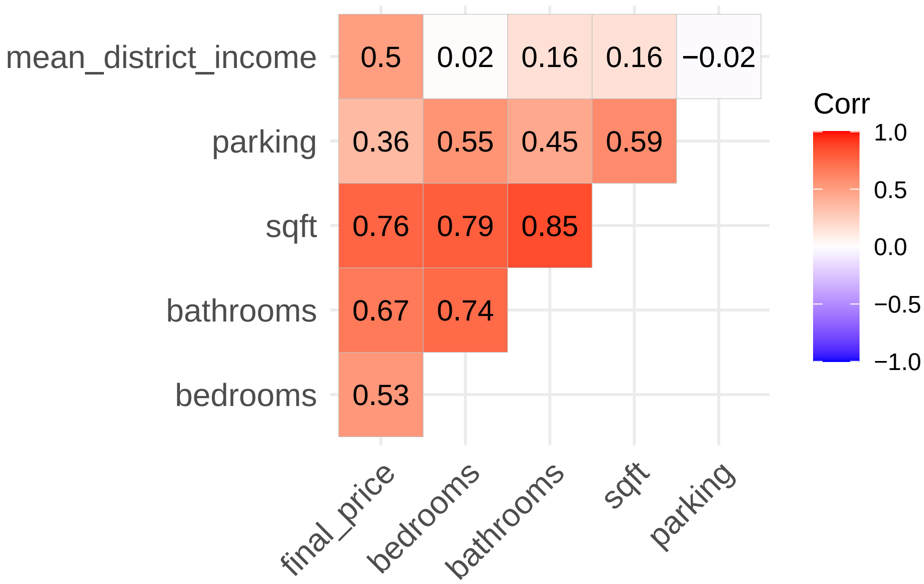

Figure 3: Correlogram

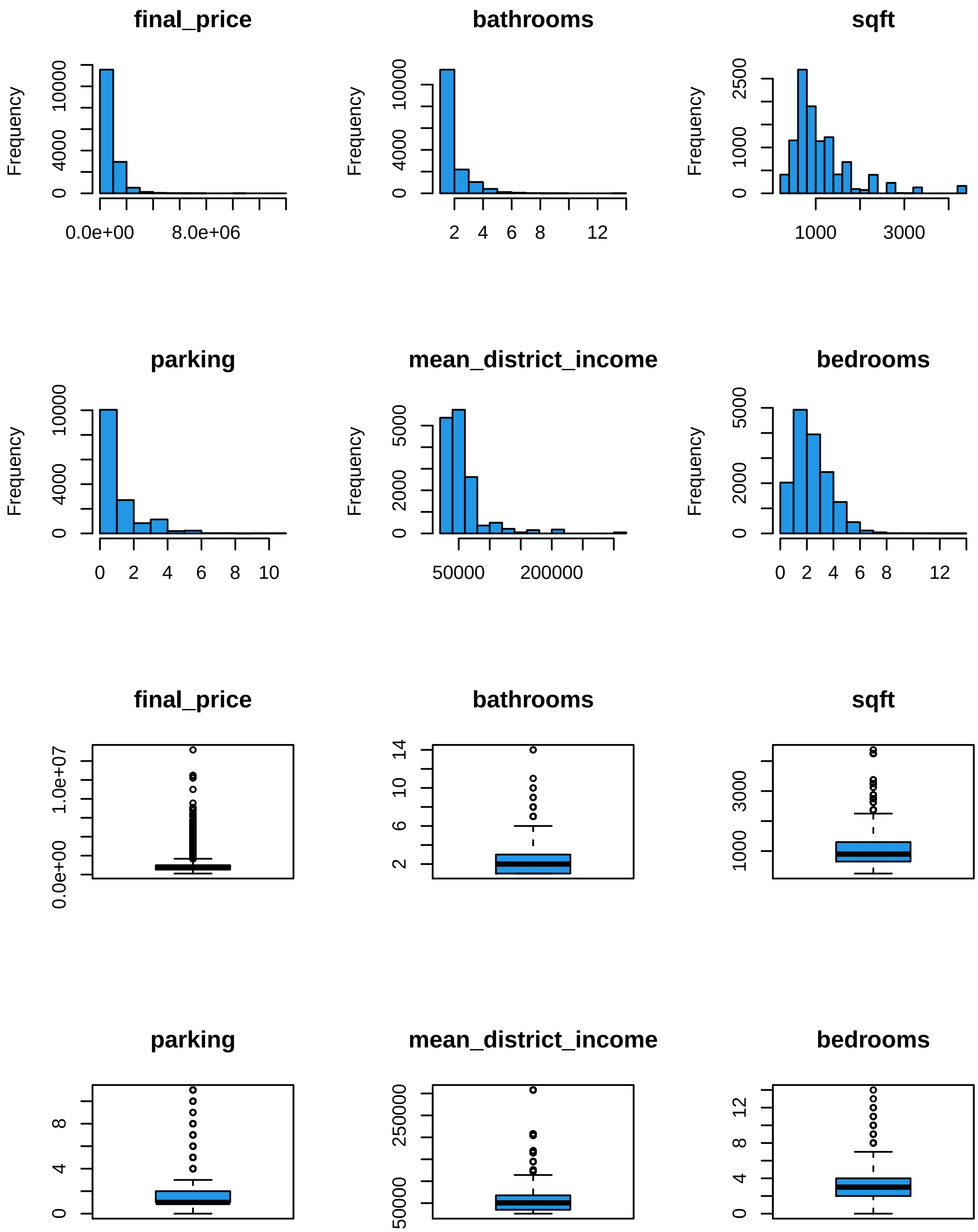

Figure 4: Distribution of predictor and target variables

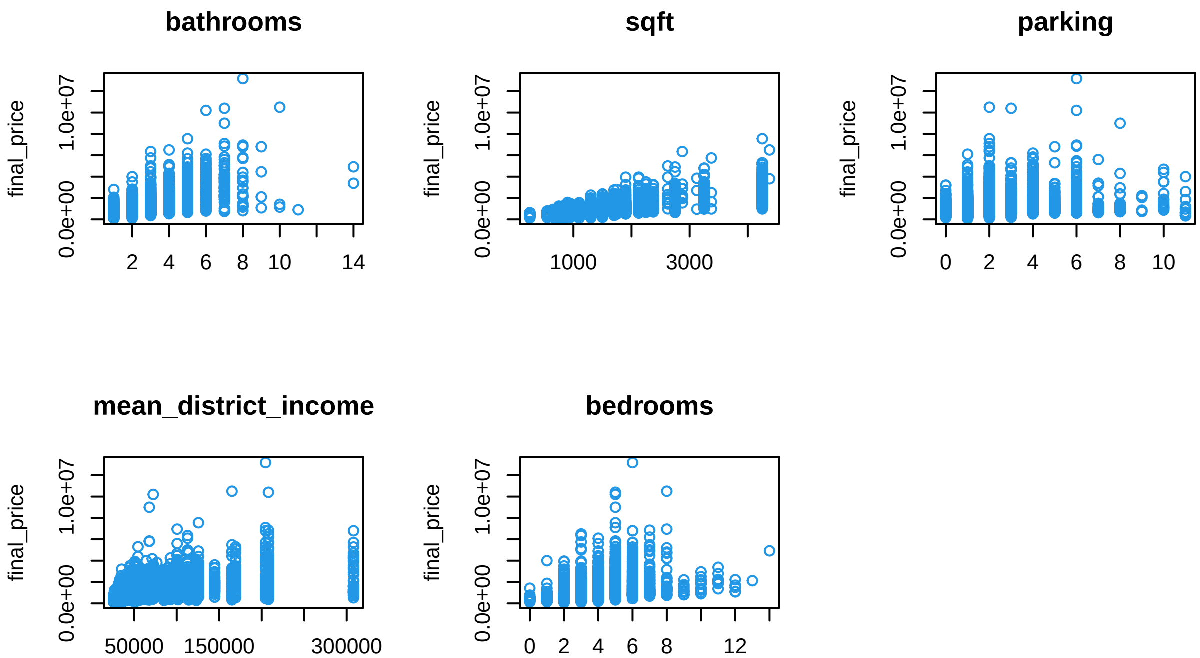

Figure 5: Scatterplot of Predictors vs. Final Price

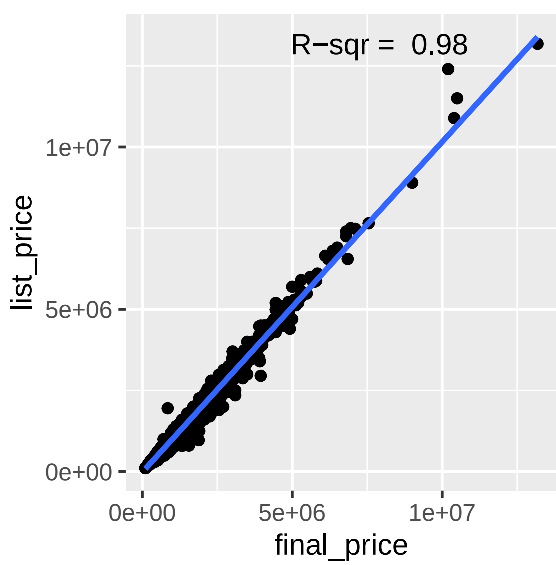

Figure 6: Final Price vs. List Price