Group 4 Final Project Report

Juan Falck, Euclides Rodriguez, Al Haque, Melvin Matanos

Utilizing Machine Learning Techniques to Predict Housing Prices in Kings County, Washington

Keywords:

Multiple Linear Regression, Decision Tree, Random Forest, eXtreme gradient boosting, House Prices

Abstract:

House prices are increasing every year, as such there are several factors that are considered when determining the price of a house. Sellers need to know the optimal market value of their properties, while buyers need to know how much to offer to make a competitive bid. As such, it is important to understand what the most important factors are to take into consideration when determining the price of a house. Our project aims to predict housing prices using sample data from Kings County, Washington. We will use multiple linear regression and a plethora of machine learning techniques such as decision trees, random forest, and xgboost. We predicted house prices with these models and looked at each important predictor that played a role in each prediction. Ultimately, xgboost did a better job at predicting house prices with a lower RMSE and a higher R-squared value.

Introduction:

The housing market is a significant component of any economy, and the value of any housing property is influenced by several factors such as location, size, grade of the house, number of bedrooms, and bathrooms. Given the importance of residential properties, accurate prediction of housing prices is crucial for many stakeholders such as buyers, sellers, and real estate agents.

Due to the sensitive nature of price predictions, there are significant ramifications in the real world if the accuracy of the predictions are off. This can lead to issues where sellers overestimate the value of their home at a higher price than the market price suggest, resulting in potential loss of buyers and a lower sales price. Conversely, if a buyer underestimates the value of a property and offers too low, they may miss the chance to purchase the home or end up paying more than they should have for the bid.

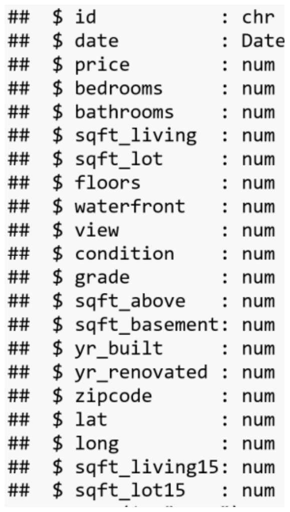
In 2021, Zillow, an American tech real estate marketplace company used an algorithm called ibuy to purchase homes that they believe would generate great profits. However, ibuy assumed incorrectly and overestimated house prices. When the housing market cooled down the model fail to account for the sudden shift, resulting in inaccurate predictions. As a result, Zillow owned many homes with no potential buyers, leading them to sell the properties at a lower price than they wanted. This led to Zillow losing over 300 million in its third quarter and a reduction of its workforce by 25% to compensate for the impact on its business. This example demonstrates the dangers of inaccurate house price predictions and highlights the need for precise predictive models.

The nature of predicting housing prices is a challenging and difficult task due to several reasons. First, house prices are influenced by several factors, and the relationship between these factors and prices may be complex and nonlinear. Secondly, the housing market is spontaneous and dynamic leading to many fluctuations due to changes in economic and social factors, making it difficult to develop and discern accurate predictions in the long term. However, house pricing predictions have many real-life applications. For example, people who are buying or selling homes can use predicted prices to make informed decisions about their housing properties. These predictions can help sellers determine the appropriate listing prices for their homes and help buyers make competitive bids. In addition, accurate predictions can help real estate agents advise their clients on the best strategies for buying or selling properties.

Methodology:

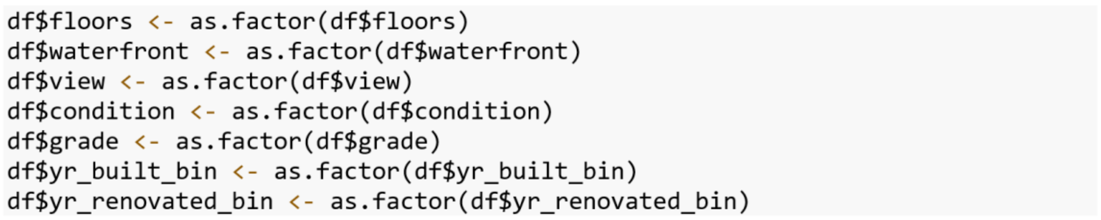
The aim of this report is to predict house prices using machine-learning techniques. The dataset can be found on Kaggle, which consists of housing prices in Kings County, Washington State in the US. The dataset contains historic data of houses sold between May 2014 and May 2015.

Upon initial inspection, the dataset contains 22 columns and 17346 observations. Most of the columns describe various characteristics of the house, for example, the number of bedrooms in the house, the number of bathrooms, the square feet of the living area,view of the house, and the condition and grade of the house. The columns are classified as integer, numeric, or character columns. Moving forward, let’s examine the year values to obtain a sense of the dataset’s range, the data records houses built as back as 1900 to as recent as 2015. Additionally, some of the houses underwent renovations throughout the year. Notably, we gather no missing/empty values within the dataset, sparing us the need to impute the values or remove any observations.



```
##  $ id           : chr
##  $ date         : Date
##  $ price        : num
##  $ bedrooms     : num
##  $ bathrooms    : num
##  $ sqft_living  : num
##  $ sqft_lot     : num
##  $ floors       : num
##  $ waterfront   : num
##  $ view         : num
##  $ condition    : num
##  $ grade        : num
##  $ sqft_above   : num
##  $ sqft_basement: num
##  $ yr_built     : num
##  $ yr_renovated : num
##  $ zipcode      : num
##  $ lat          : num
##  $ long         : num
##  $ sqft_living15: num
##  $ sqft_lot15   : num
```

Though the dataset may have no missing values we still have to perform some data cleaning before we train the model on the data. We will convert some columns into categorical, these columns are bedrooms, bathrooms, floors, waterfronts, view, condition, and grade. This is achieved by converting these columns into factors. We will also omit some columns in our model i.e. id, date, and zipcode. These fields are irrelevant in our model since it is simply an identifier of the transaction of the house. While date and zipcodes may cause the model to overfit on the training data since there are too many levels for the model to keep track of possibly.



```
df$floors <- as.factor(df$floors)
df$waterfront <- as.factor(df$waterfront)
df$view <- as.factor(df$view)
df$condition <- as.factor(df$condition)
df$grade <- as.factor(df$grade)
df$yr_built_bin <- as.factor(df$yr_built_bin)
df$yr_renovated_bin <- as.factor(df$yr_renovated_bin)
```

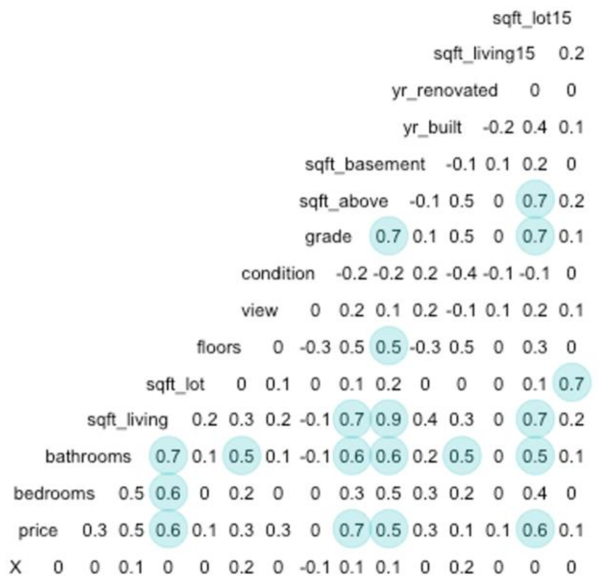
Additionally, we will categorize the year_built values of the houses to help reduce the number of variable levels. To do this, we'll create a new column called "yr_built_bin".
Houses built during the 2000s will be considered as "newly built", houses built after 1950 are considered "medium built" and houses built before 1950 are "old built". This will help reduce the distinct levels of the column and prevent overfitting on the training data. Furthermore, we will mutate another column called "yr_renovated_bin" where we will categorize houses renovated into two bins, houses renovated before 2000 will be considered as "recent renovation" and house renovated after 2000 will be "not recent renovation".

```
df <- df %>% mutate(yr_built_bin = case_when(
  yr_built >= 2000 ~ "new build",
  yr_built >=1950 & yr_built <2000 ~ "medium build",
  yr_built < 1950 ~ "old build"
  ))

df <- df %>% mutate(yr_renovated_bin = case_when(
  yr_renovated >= 2000 ~ "recent renovation",
  yr_renovated < 2000 ~ "not recent renovation"
  ))
```

Finally, we conclude our data-cleaning efforts by combining the relevant columns into a new dataset. We will transmute the price (our dependent variable), the cleaned version of our date column by using the parse_number function in R, and the rest of our independent variables which we believe will be instrumental in accurately predicting the house prices. These variables include the number of bedrooms, number of bathrooms, square foot of living, square foot of the lot, number of floors, condition of the house, zipcode, yr_built_bin, and yr_renovated_bin.

Experimentation and Results:

After cleaning the dataset, we generated a correlation matrix that shows the various pairwise correlation between each variable The matrix reveals minimal correlation between the predictors except for bathroom and price. We can see that bathroom has a positive correlation with many of the other predictors. However, our focus is on prediction rather than analysis. Price positively correlates with all the predictors, with the strongest correlations being sqft_living, grade, sqft_above, and sqft_living15. These predictors may be useful in predicting price.



															sqft_lot15
														sqft_living15	0.2
													yr_renovated	0	0
												yr_built	-0.2	0.4	0.1
											sqft_basement	-0.1	0.1	0.2	0
										sqft_above	-0.1	0.5	0	0.7	0.2
									grade	0.7	0.1	0.5	0	0.7	0.1
								condition	-0.2	-0.2	0.2	-0.4	-0.1	-0.1	0
							view	0	0.2	0.1	0.2	-0.1	0.1	0.2	0.1
						floors	0	-0.3	0.5	0.5	-0.3	0.5	0	0.3	0
					sqft_lot	0	0.1	0	0.1	0.2	0	0	0	0.1	0.7
				sqft_living	0.2	0.3	0.2	-0.1	0.7	0.9	0.4	0.3	0	0.7	0.2
			bathrooms	0.7	0.1	0.5	0.1	-0.1	0.6	0.6	0.2	0.5	0	0.5	0.1
		bedrooms	0.5	0.6	0	0.2	0	0	0.3	0.5	0.3	0.2	0	0.4	0
	price	0.3	0.5	0.6	0.1	0.3	0.3	0	0.7	0.5	0.3	0.1	0.1	0.6	0.1
X	0	0	0.1	0	0	0.2	0	-0.1	0.1	0.1	0	0.2	0	0	0

We have developed three multiple linear regression models to predict housing prices. The first model predicts price by all the predictors, resulting in an r-squared of 0.6502 and a root mean squared value of 163008. Although the high RMSE indicates poor model fit, the high r-squared suggests that the model explains a significant portion of the variance. The second model focuses on the correlated predictors with price, but this doesn't improve the model fit by much and has lowered the adjusted r-squared value from 0.6502 to 0.5388. We also tried a backward stepwise regression which yielded a model with almost all predictors with an adjusted r-squared value of 0.6502 and an RMSE of 163102.7. Overall our multiple linear regression approach struggles to make accurate predictions, but the high r-squared value explains approximately 60% of the variation in the data.

```
summary(lm(price ~ sqft_living + grade + sqft_above + sqft_living15,
           data = house))
```

```
## Residual standard error: 249400 on 17341 degrees of freedom
## Multiple R-squared:  0.5389, Adjusted R-squared:  0.5388
## F-statistic:  5068 on 4 and 17341 DF,  p-value: < 2.2e-16
```

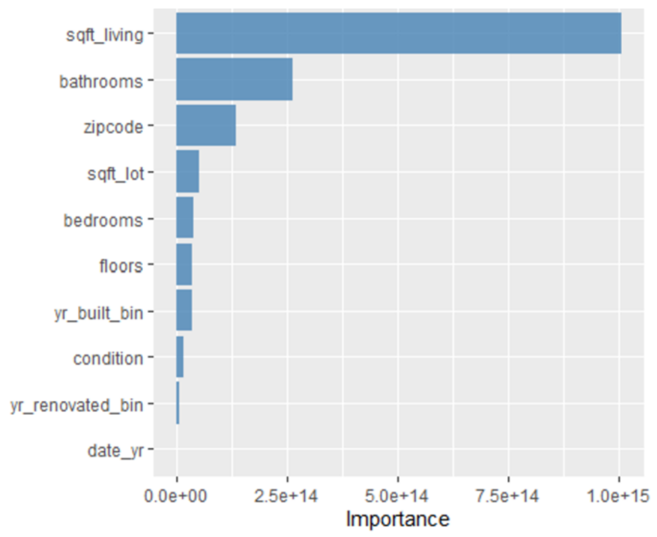
```
model_all <- lm(price ~ ., data = house)
```

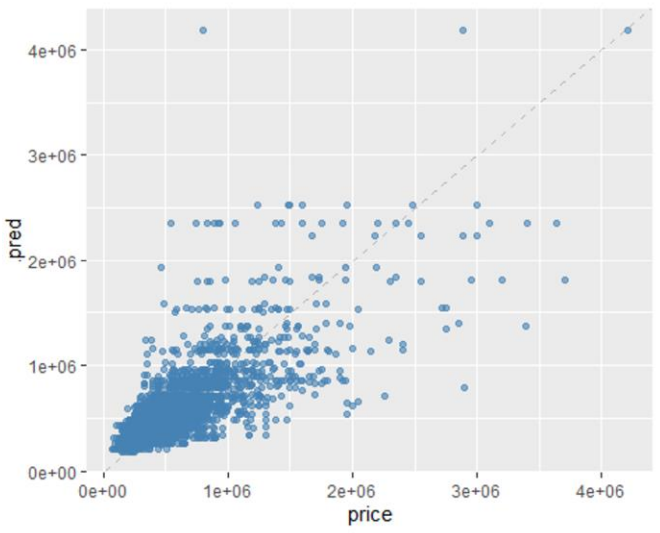
```
## Residual standard error: 217200 on 17330 degrees of freedom
## Multiple R-squared:  0.6505, Adjusted R-squared:  0.6502
## F-statistic:  2151 on 15 and 17330 DF,  p-value: < 2.2e-16
```

For our next approach, we will utilize machine-learning techniques to predict house prices from the dataset. We will use Tidymodels in R to select predictors and tune parameters to achieve the best possible R-squared and RMSE values. First, we'll split the dataset into training and testing sets, and stratify the sample by price to ensure equal variation of prices in both sets. Next, we'll create 10-fold cross-validations to resample the training set, into 10 folds to prevent resampling with an equal proportion of prices within each.

We began by building a decision tree model using Tidymodels, The model's cost_complexity, tree_depth and min_n parameters were tuned by the package to optimize predictions. After that, we set the engine to rpart, the mode to regression. and specified a tree grid with 4 levels. After running the models, we collected the metrics of the best model, which returned an R-squared of 0.586 and an RMSE of 228944. We then identify the most important variables using the VIP package from R, the analysis reveals that sqft_living, bathrooms, and zip codes are the most influential predictors. With the high RMSE and low R-squared value, the model doesn't appear to capture most of the variability within the data.

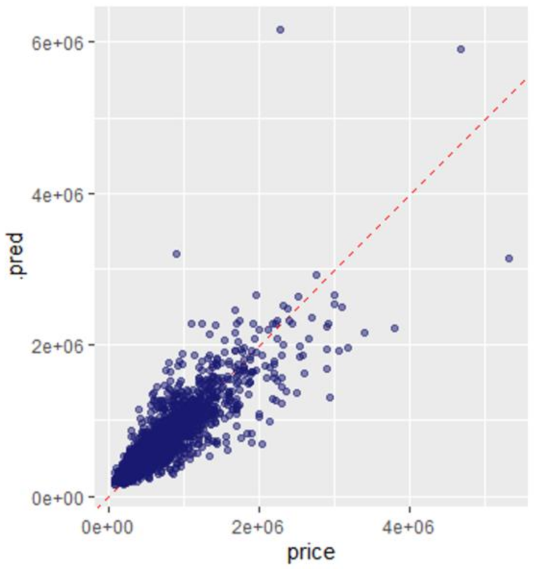
```
collect_metrics(final_rs)

## # A tibble: 2 x 4
##   .metric .estimator  .estimate .config
##   <chr>   <chr>           <dbl> <chr>
## 1 rmse    standard   228944.    Preprocessor1_Model1
## 2 rsq     standard        0.586 Preprocessor1_Model1
```

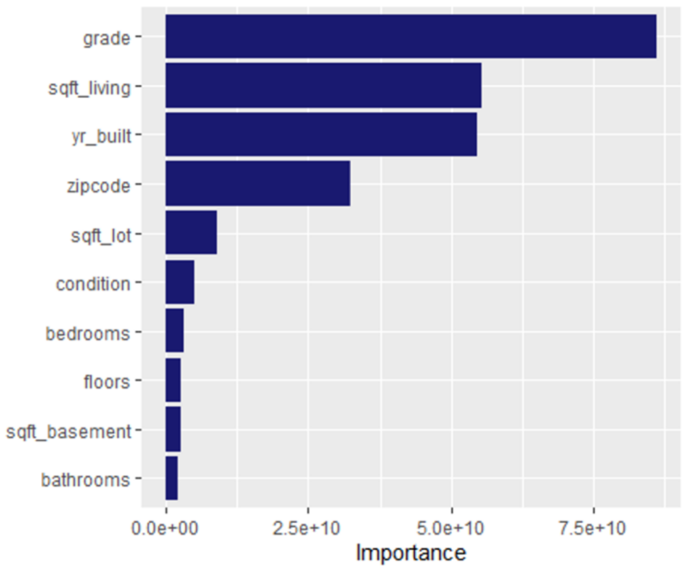




Next, we will try the random forest engine, we use the use-models library in R to assist us in the model creation for the random forest. This package will provide what parameters to tune and the workflow to be set. We set the ranger package for the engine, set the mode to regression, the number of trees to 1000, and set the number of candidate points to 11. Selecting the metric for the best model we get an RMSE of 173029 and an R-squared value of 0,775 a much better improvement than the decision tree model. Though the RMSE is still high, it is lower than the one in the decision tree and with a higher R-squared.

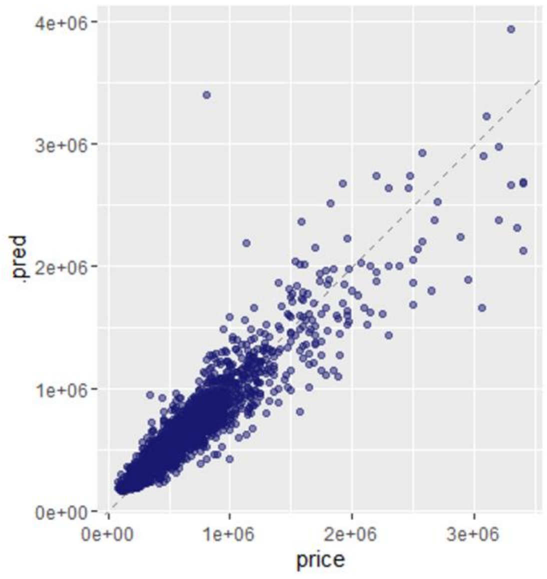
```
## # A tibble: 2 × 4
##   .metric .estimator  .estimate .config
##   <chr>   <chr>           <dbl> <chr>
## 1 rmse    standard   173029.    Preprocessor1_Model1
## 2 rsq     standard        0.775 Preprocessor1_Model1
```

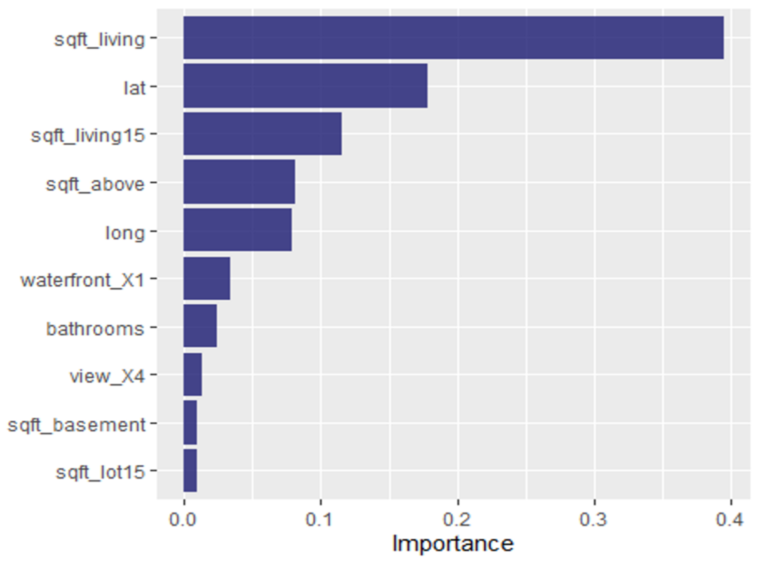




Finally, we will use the xgboost algorithm. We will allow Tidymodels to tune the tree_depth,min_n, sample_size,mtry, and learn_rate while we set the tree to 1000. We set the engine to xgboost, the mode to regression, and the xgb_grid size to 30. Before running the model, our team member Juan decided to add some data recipes to prepare the data, he normalized all the numeric columns and created dummy variables for factor variables. After running the model and gaining the metrics we get the R-squared value is 0.87 and the RMSe is 124007 which was an improvement from both the decision tree and random forests model. An analysis from the VIP package shows that sqft_living, lat, and sqft_living15 were the most important predictors for the model.

```
collect_metrics(house_fit)

## # A tibble: 2 x 4
##   .metric .estimator  .estimate .config
##   <chr>   <chr>           <dbl> <chr>
## 1 rmse    standard   124007.    Preprocessor1_Model1
## 2 rsq     standard        0.874 Preprocessor1_Model1
```





Discussion and Conclusions:

To conclude, predicting house prices is difficult, for our initial effort using multiple linear regression we are able to see that the models we’ve created are not a good fit for the data with high RMSE and low R-squared value. Once we started implementing the decision tree and other ensemble method was the fit getting more accurate, however, once we implemented random forest and xgboost we were able to reduce the RMSE and get a high R-squared even though the diagnostics were overplotted. Unfortunately, the columns in the dataset left much to be desired since columns like grade,sqft_living, and zip codes played a role in the creation of the model. If the dataset contained columns that represented actual considerations of house prices like neighborhoods, and distance from school then it would be much easier to interpret what actually influences house prices. For now, this is a good start for price predictions for the house with this dataset.