Final Report

Team 2 – Mckinlytic: Ellen Wang and Karl Hu

Table of Contents

I. Background and Relevance

The real estate and brokerage market in the United States had a volume of 195.85 billion dollars of market size in 2021 and has been experiencing a growth rate of over three percent in the past five years.

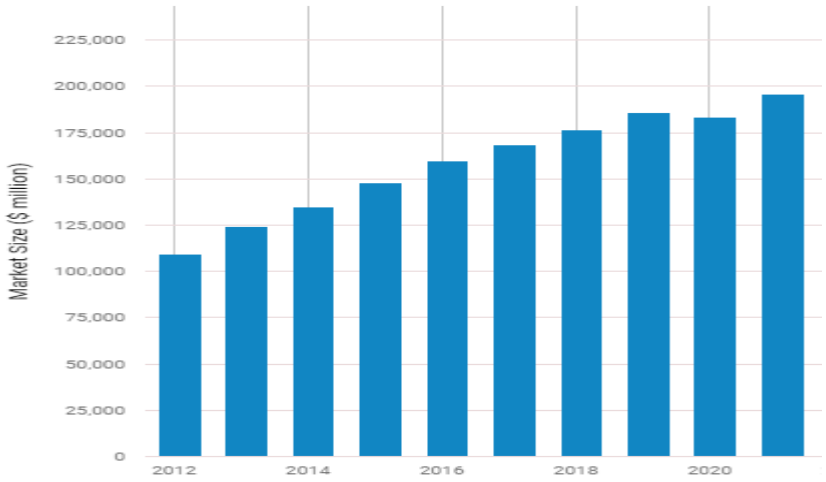


Chart 1. Real Estate Sales & Brokerage in the US - Market Size 2002–2021
Adapted from *ibisworld.com*

With the considerable market size and rapid growth rate, it is a market that could not be neglected. Building a model to estimate the price of the property, given the features of the property, especially in the preliminary screening stage, could be a helpful tool for buyers, sellers, real estate agents and constructors. It would be time-saving and efficient to utilize such a tool before precious labor hour is put into the analysis of the individual house.

II. Data Preparation and Visualization

The data used in this project covers selected US real estate property prices and is retrieved from *kaggle.com*. There are a total of 13 features (columns) and 545 samples (rows) in the dataset.

The dependent variable in the dataset is the price of the property.

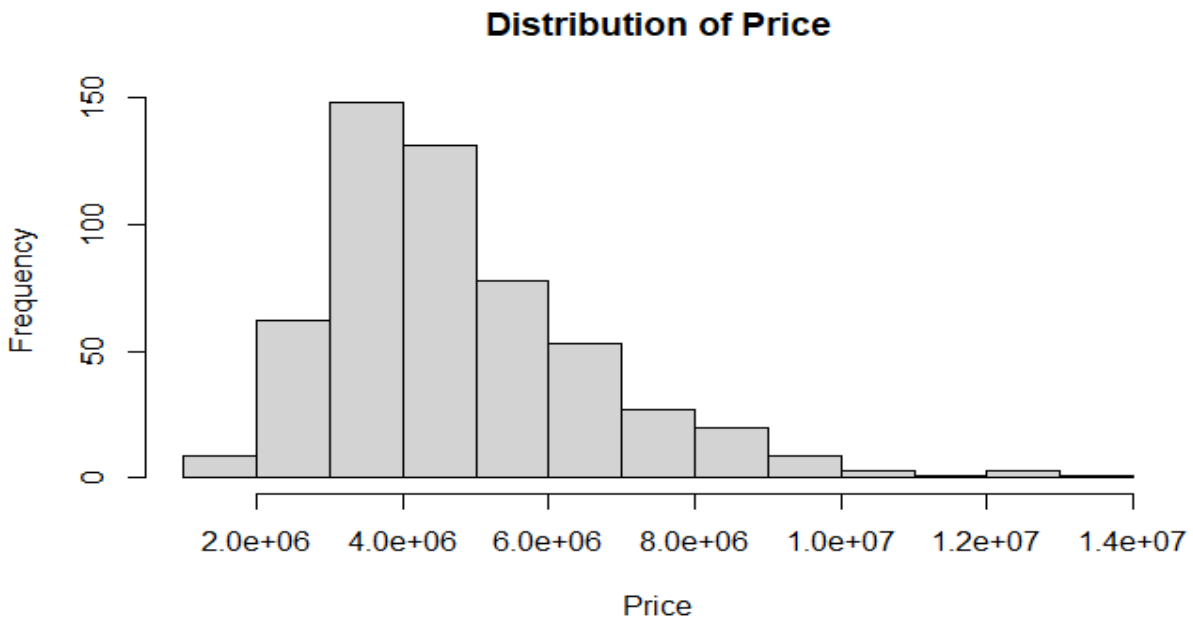


Chart 2. Price Distribution

As shown in Chart 2, the price data in the raw dataset has a fat tail and is slightly skewed to the right, with most houses having price of over 2 million dollars. The uneven distribution is not ideal for our numerical target prediction model construction, and we have thus decided to transform the variable into the log of price to make it more gaussian.

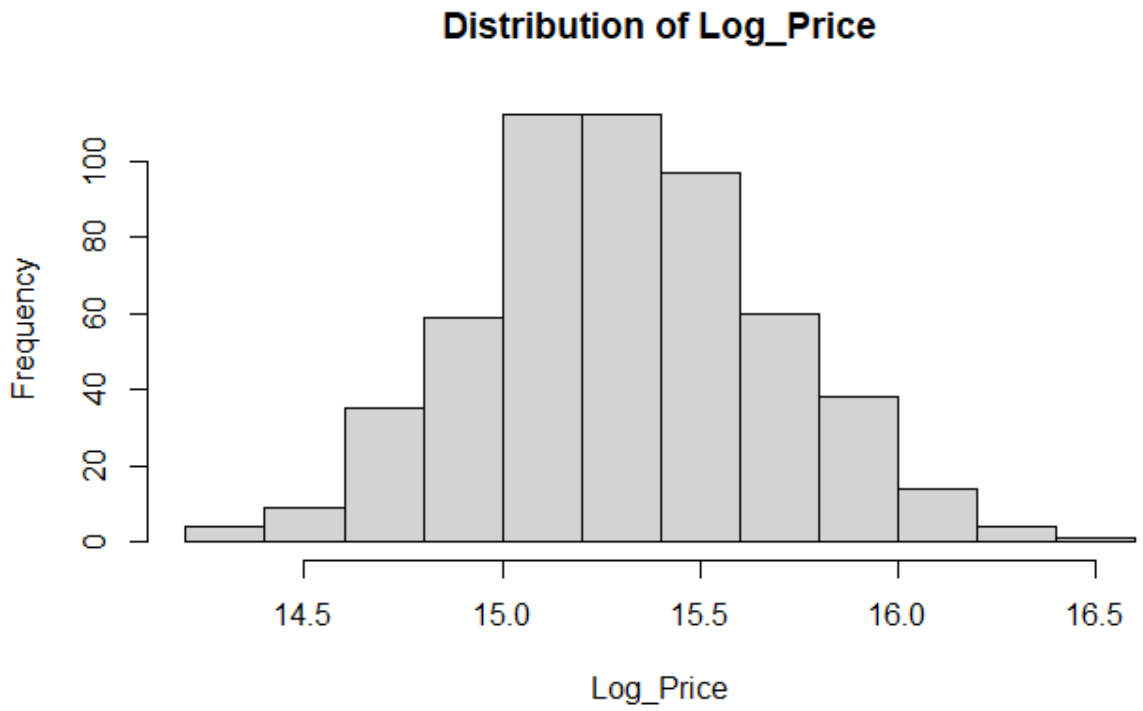


Chart 3. Log of Price Distribution

After the transformation, the frequency of log of price is more evenly distributed and the the numerical value is smaller. It would also be reasonable to predict the log of price as the result will be how much percentage increase price will benefit given a unit increase/change of a feature.

For the categorical target prediction, we have decided to create a new binary price level variable. The price level was determined by comparing the price of the single property with the average price of all properties within the dataset.

After transforming and generating the new target variables, we have then partitioned the data into two parts. The first part consists of sixty percent of all datapoints and was used for training of the models, and the second part, which consists the remaining forty percent, were used to validation purposes to avoid the issue of overfitting.

The independent variables are as follows:

1. Total area of the property – Numerical
2. Number of bedrooms – Numerical
3. Number of bathrooms – Numerical
4. Number of stories/floors – Numerical
5. Number of parking spaces – Numerical
6. Access to the main road – Categorical
7. Availability of a guest room – Categorical
8. Availability of a basement – Categorical
9. Availability of water heaters – Categorical
10. Availability of air conditioning – Categorical
11. Postal preferred area status – Categorical
12. Furnishing status – Categorical (3 classes)

There are a total of twelve independent variables and five are numerical. The remaining seven variables are all binary except for Furnishing Statue, which had three classes.

A selected number of variables here are visualized for preliminary analysis and simple relationship could be observed. The price has a positive relationship with area as shown in Chart 4. As the other numerical variables are all integers, a scatterplot would display limited information. Boxplot are applied for those variables. A weak positive relationship could also be observed from the chart 5. There is the existence of outliers for each variable but as it would often be the case in the real world, they were unaltered.

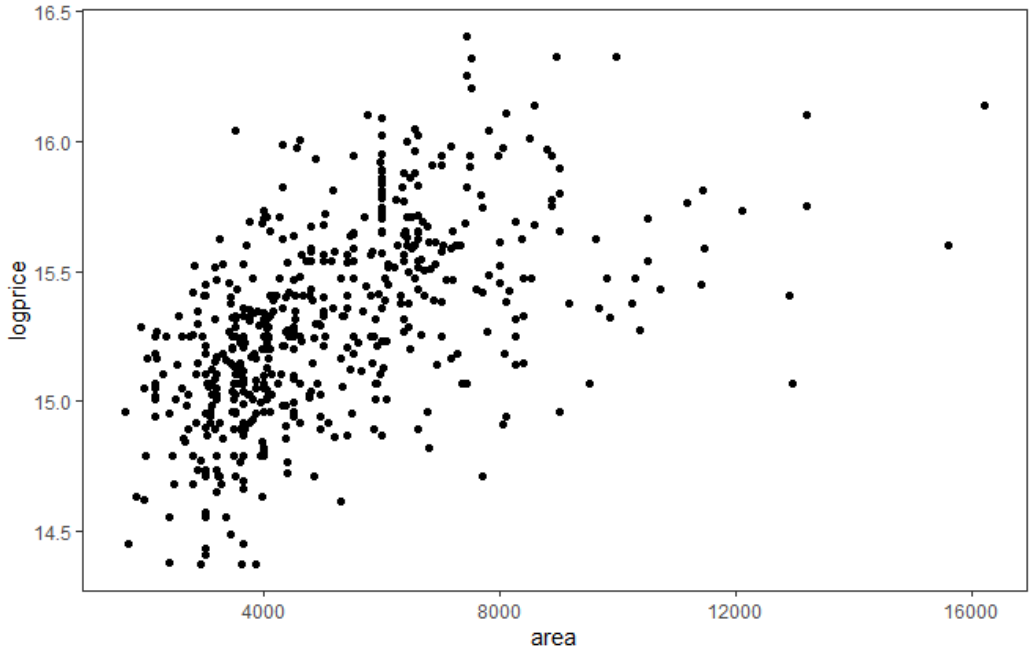


Chart 4. Relationship of log of price and area

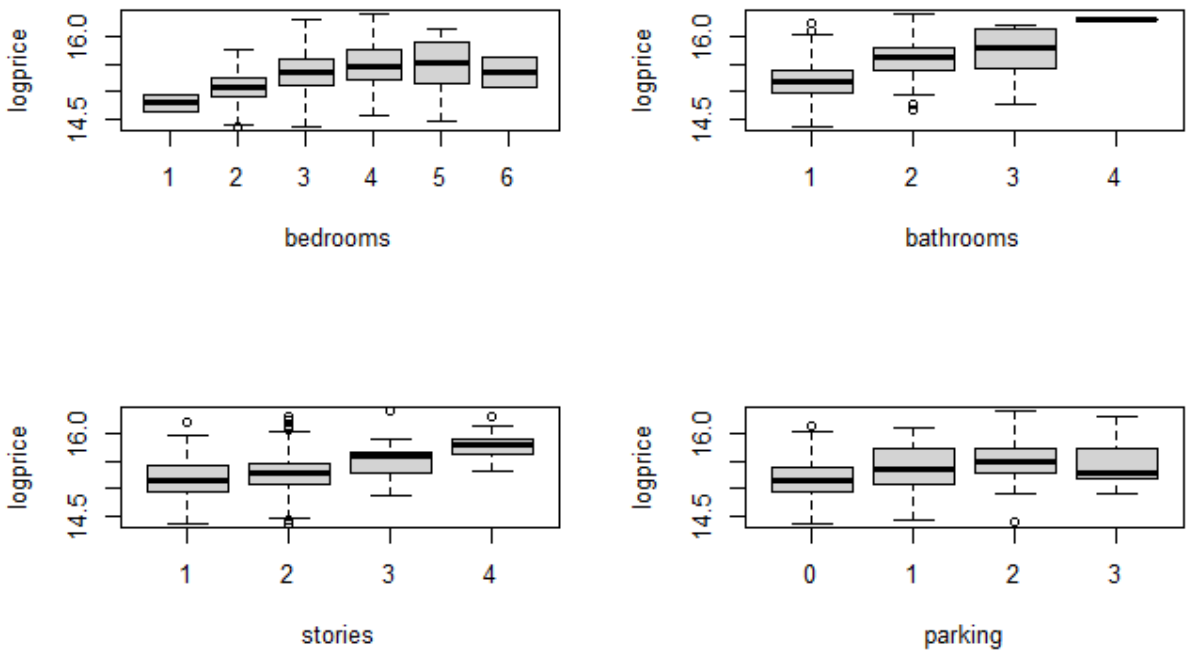


Chart 5. Boxplots of variables

III. Numerical Target Predicting

As explained in the previous section, the numerical target used was the log of price. The first model used in predicting numerical target (log of price) is simple multilinear regression. This variable was chosen as it is usually considered a reflection of the intrinsic value of a real estate property and is commonly used as one of the deciding factors of whether the buyer will have the willingness to purchase the property given the features and characteristics. An accurate prediction of it would matter as it could help buyers and sellers to price/budget on the property and help improve market efficiency.

The linear regression was first performed on the dataset with all given variables and three variables were statistically insignificant (number of bedrooms, the availability of a basement, furnishing status). We decided to remove those variables and has boosted the R-squared and lowered the RMSE. We have then removed a total of seven outliers through the examination on a series of fitting analysis plots. A new feature-engineered variable, area-squared, was also added to the model. An additional two cluster indexes were generated and added to the feature list of the model, utilizing KNN and hierarchical clustering methods, to cluster the datapoints into groups. The RMSE of this model was 0.2069 and R-Squared was 0.6907.

As shown in Chart 6, the residuals of the prediction have an acceptable evenness on the frequency distribution. It was not centered at zero as each house still has its own characteristics thus residuals should be natural. It is relatively skewed to the right, meaning that the model tends to offer a higher price than suggested.

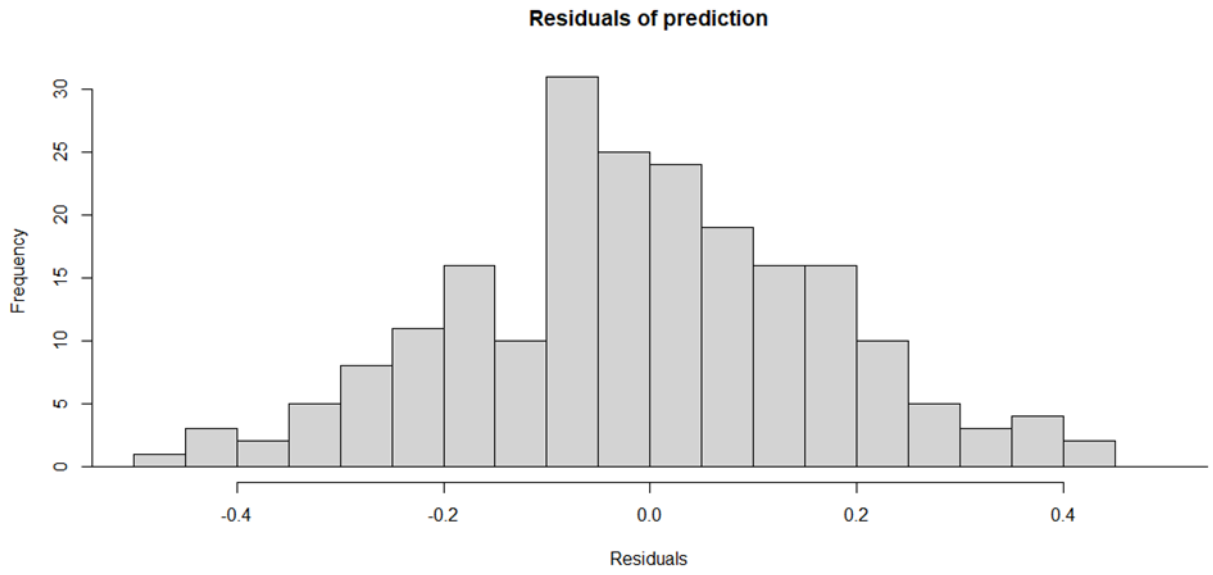


Chart 6. Residuals of the multilinear regression model

A regression tree was also performed on the dataset, but the accuracy metrics were much lower than the multilinear regression. The random forest method was also utilized. The grid search method was applied in searching the optimal hyperparameters, but due to extreme memory consumption of this model, the grid search only had a relatively small range. The random forest offered the most optimal accuracy metrics of R-squared being 0.6698 and RMSE being 0.2361.

Furthermore, we have tried to utilize DataRobot to build the prediction model of the numerical target. The best model available was the Eureqa Generalized Additive Models (Eureqa GAM) with 10000 generations.

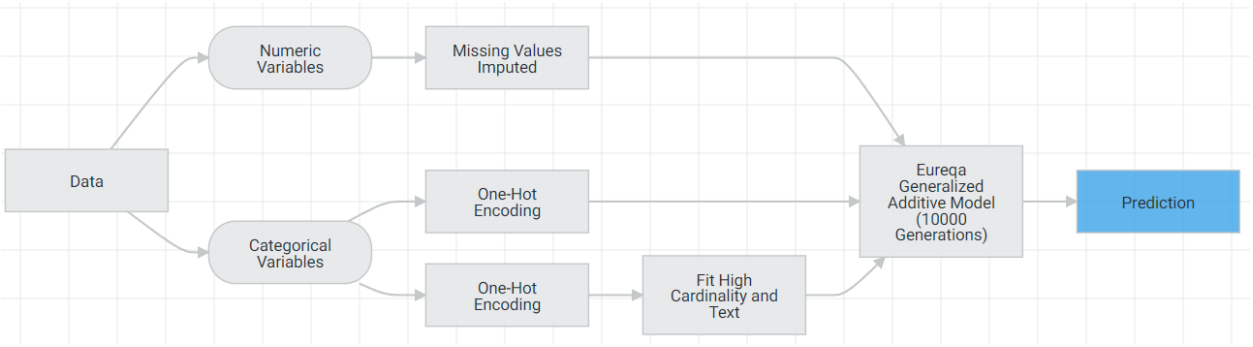


Chart 7. Blueprint of Eureqa GAM

This model was powered by the Eureqa engine. The One-Hot Encoding is essentially the process of dummy variable creation. The GAM model was then able to minimize error by smoothing the curves iteratively with generations. The model has a unique partition logic which helps improve the accuracy metrics.

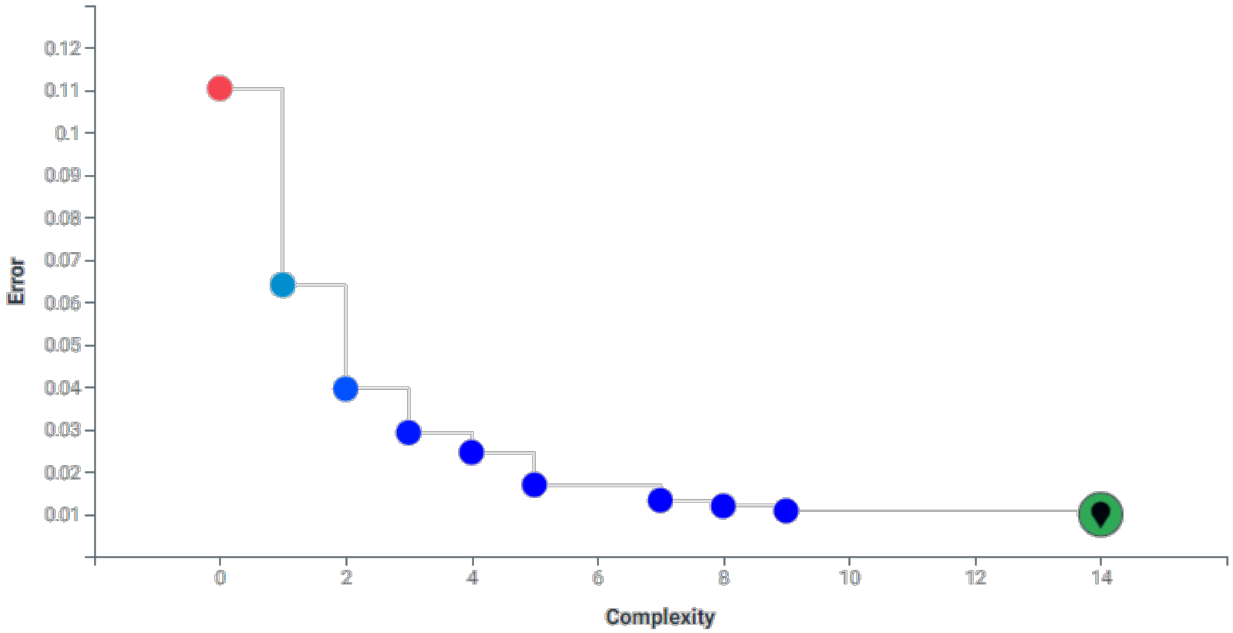


Chart 8. Complexity – Error Frontier of Eureqa GAM

As shown in Chart 8, the complexity index of the model was set to fourteen to minimize errors and it has chosen to utilize all the features available in the dataset. The RMSE of the Eureqa GAM model was 0.1986 and R-Squared was 0.6627.

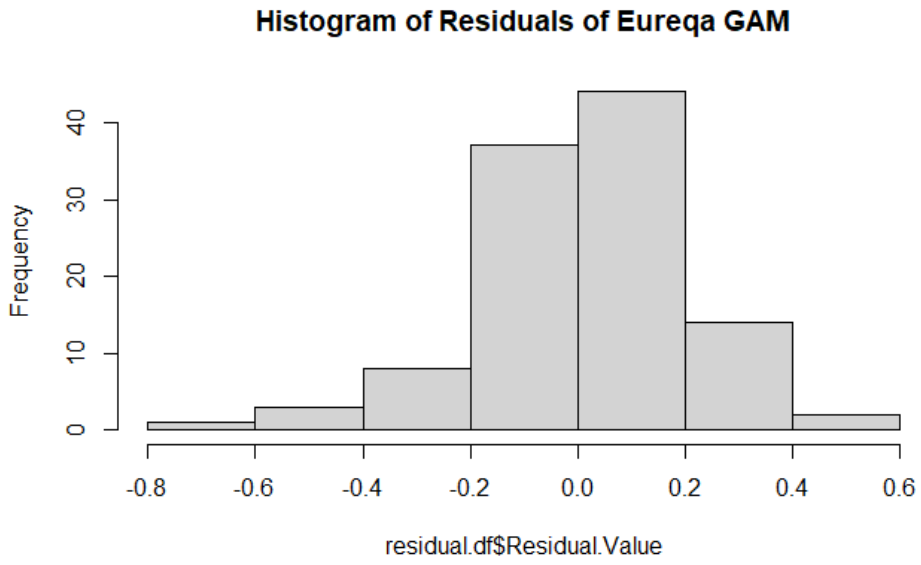


Chart 9. Residuals of Eureqa GAM

As shown in Chart 9, the residuals of the prediction of the Eureqa GAM have an acceptable evenness on the frequency distribution. It was not centered at zero as each house still has its own characteristics thus residuals should be natural. It is relatively skewed to the left, meaning that the model tend to offer a lower price than suggested.

The model has also offered a Variable Impact Frontier (VIF). It was similair to the VIF in our random forest model which would make sense since the variables are logically impactful on the pricing of a property.

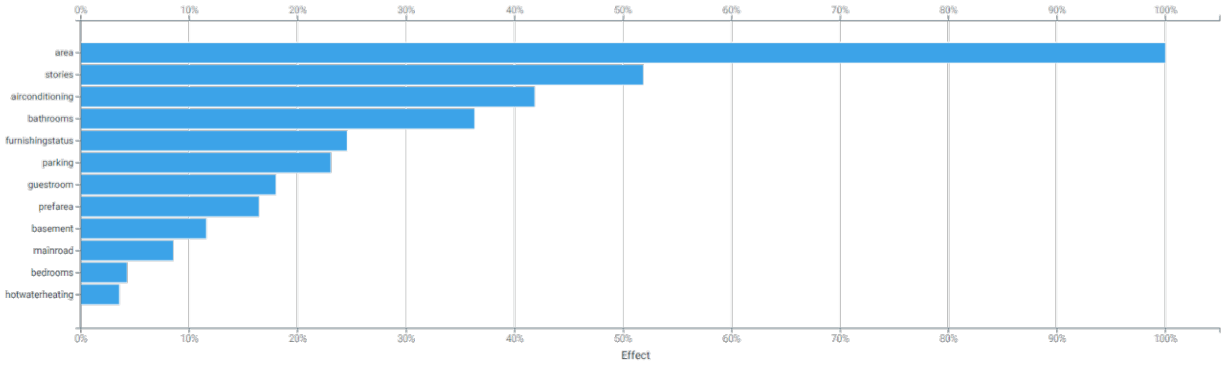


Chart 10. VIF of Eureqa GAM

DataRobot has also offered other models with lower accuracy including the elastic-net regressor and light gradient boosting on ridge regressor methods but due to the limitation of space in this report and the lower accuracy, they are not being included.

Comparing the best model hand-crafted in R and the DataRobot models, The Eureqa GAM had lower RMSE and multilinear regression had higher R-squared. The residuals of the prediction are also skewed towards different directions. The interpretability of both models are extremely high as Eureqa GAM also offered a similar beta index for each variable. It is ambiguous to determine one is better than the other and they both have highly applicable accuracy metrics. A potential method would be combing the results of the two models and test on the effectiveness and accuracy on the results again, but we were unable to figure out the proper way to complete such a plan due to time limitation and technical skill constraints.

IV. Categorical Target Predicting

In this part of the report, we will highlight some of our best hand-crafted classification models for the categorical target and compare them with the best models built by DataRobot. To measure the performance of our models, we used F1 score as our key metric, as we wanted our models to be accurate in terms of both sensitivity and specificity. Since the F1 score calculates the weighted average of both metrics, we considered it to be the most appropriate performance metric for our models. Among all the models built by hand, logistic regression and boosted tree turned out to be the best models in terms of the F1 score. The best logistic regression model, after removing 19 outliers, had a F1 score of .9161, while the boosted tree had a F1 score of .8653.

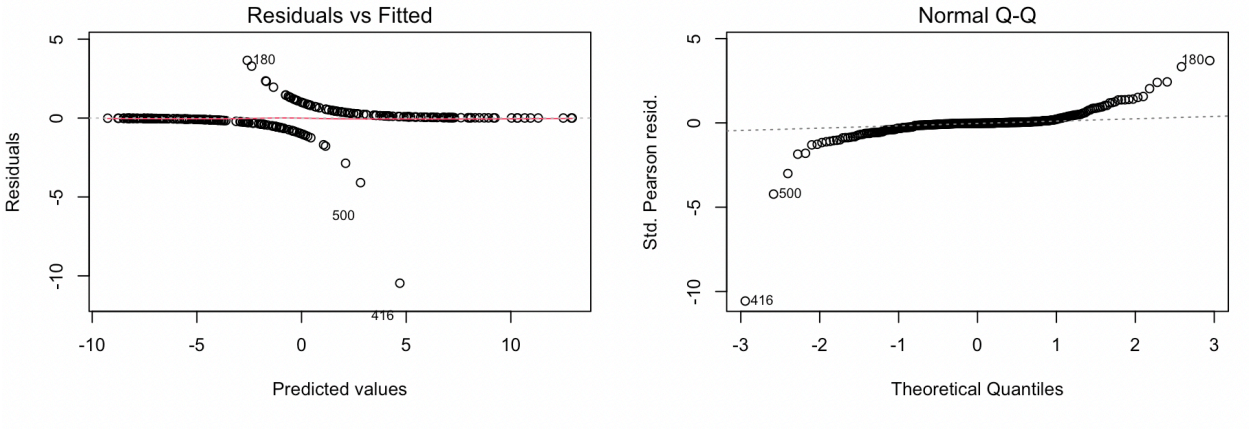
A. Best Hand-Crafted Classification Models

i. Logistic Regression

To better understand our best logistic regression model, it would be reasonable to explain some of the detailed model-building process, which eventually helped us land in the final logistic regression model. We first started with a simple logistic regression on all the variables, excluding the price. The reason why we wanted to exclude price, the numeric target, from the regression is that our categorical target, the price level, is built on the price variable. Therefore, if we predict the price level using the price, there will be a multicollinearity issue, which is something we wanted to avoid. We also added two new variables to the dataset, which are the cluster labels created from hierarchical clustering and K-means clustering methods, respectively. Then, we split the data into the training subset (60%) and the validation subset (40%), and ran the logistic regression on the training set. To understand which variables are significant and which are not, we tried various

regression models with different combinations of variables, and removed 6 insignificant variables until all the remaining variables in the regression became significant.

One important step which differentiated the logistic regression model from all the other hand-crafted models is the removal of outliers. We first plotted out the logistic regression and checked if there were any outliers by looking at the residuals vs. fitted and the normal Q-Q plots. If there were outliers in the dataset, we removed them from the training set and ran the logistic regression again on the new training set. We repeated the iterative process for 7 times, removed 19 outliers, and stopped until all the remaining outliers were almost aligning with the diagonal in the normal Q-Q plot, as shown below. By removing a fair number of outliers, we improved the F1 score from .8405 to .9161, which is a 9% improvement. This process also made the final logistic regression the best model across all the other hand-crafted models.



To better interpret our logistic regression model, we calculated the log odds of our model. Among all predictors, air conditioning, hot water heating, guestroom, and bathrooms turned out to

be the variables with the strongest predictive power. For every house with air conditioning, it is 25.28 times more likely to have a price level of above the mean price. Similarly, houses with hot water heating and bathrooms are 23.66 times and 16.48 times more likely to have a price above the average.

ii. Boosted Tree

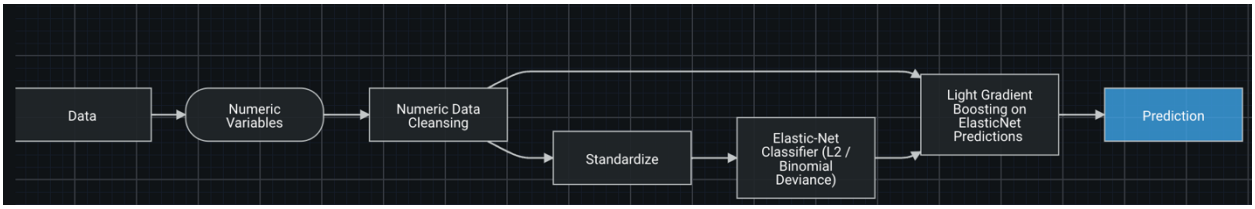
The second-best model by hand is the Boosted Tree model. We used a grid search to figure out the optimal values for the 3 hyperparameters, boos, mfinal, and coeflearn. In the nested for loop, we set the range of mfinal from 1 to 10. Boos could be either TRUE or FALSE, and coeflearn took one value out of the three in each iteration. The tree was then trained, used to make predictions, and generate the confusion matrix. When the for loop finished, we obtained a F1 score of .8653, with the best boos being FALSE, best mfinal being 8, and coeflearn being Breiman.

As the ensemble models are considered "black-box" methods, the interpretability is limited, and we were not allowed to generate a variable importance plot as we did for the logistic regression model. Therefore, we could not draw any conclusion on which predictors were more important in predicting the price level. The logistic regression model also had a higher F1 score compared to the boosted tree model. As a result, we could say that logistic regression is a better model for the following two reasons: 1) it has a higher F1 score, and 2) it does not sacrifice interpretability and is way simpler than the ensemble boosted tree.

B. Best DataRobot Models

i. Light Gradient Boosting on ElasticNet Predictions

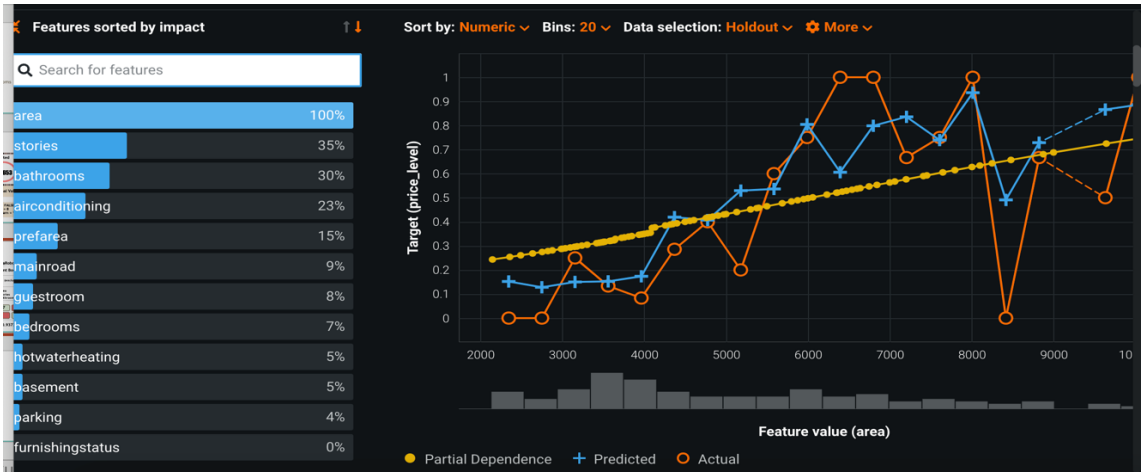
While it may be a bit surprising, the best DataRobot model is not the logistic regression model but the light gradient boosting on ElasticNet predictions, and the following graph shows the blueprint of the model. As demonstrated below, this best DataRobot model is way more complex than the logistic regression and the boosted tree models we built by hand, and thus has very different results compared to the previous models.



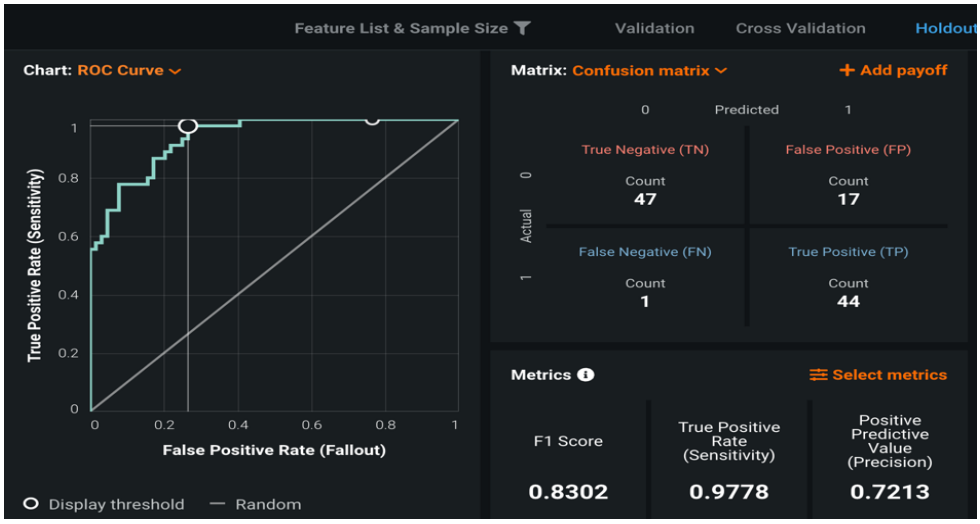
Just like all the other ensemble models, the light gradient boosting tree is a "black-box" model and we cannot really tell what is going on within the black boxes. Therefore, although the DataRobot model has better performance compared to our hand-made boosted tree model, it has a tradeoff between accuracy and interpretability, that is, it must sacrifice interpretability in order to improve the accuracy.

This model took 80% of the data as the sample and calculated the key performance metric for the holdout subset. When looking at the features sorted by impact, predictors including area (100%), stories (35%), bathrooms (30%) and air conditioning (23%) were identified as the top features that have the most predictive power. The following graph demonstrates the behavior of the top feature, area. As shown below, we can see that the partial dependence curve shows a positive, upward moving trend, and the predicted and actual curves also seem to move together,

indicating that the predication made using area was very similar to the actual result most of the time.

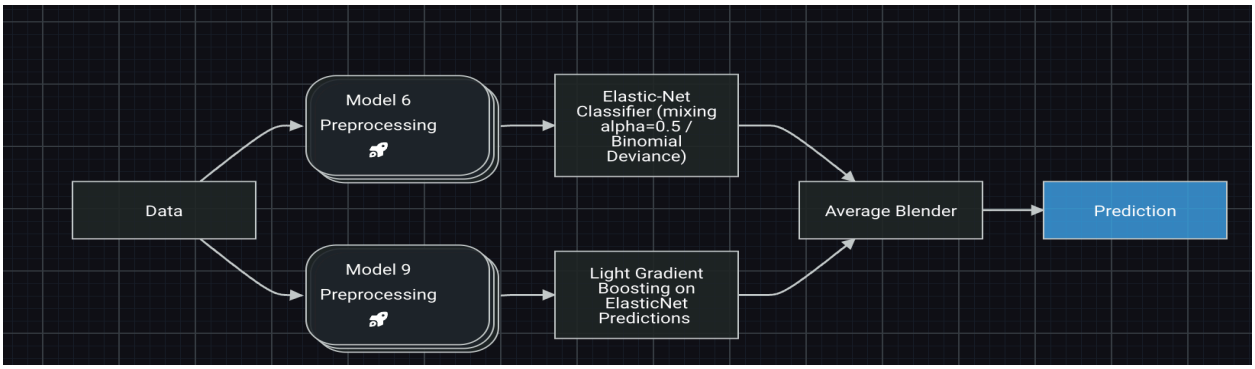


Finally, we wanted to look at the ROC curve and the confusion matrix to evaluate the performance of the DataRobot model. Just as shown below, the area under the ROC curve is .9372, and the F1 score is .8302. Both results were slightly lower compared to those of our best hand-crafted model, logistic regression. Therefore, we may want to draw the conclusion that logistic regression defeats the best DataRobot model as 1) it has better performance in terms of AUC and the F1 score, and 2) it does not sacrifice interpretability in exchange for accuracy.



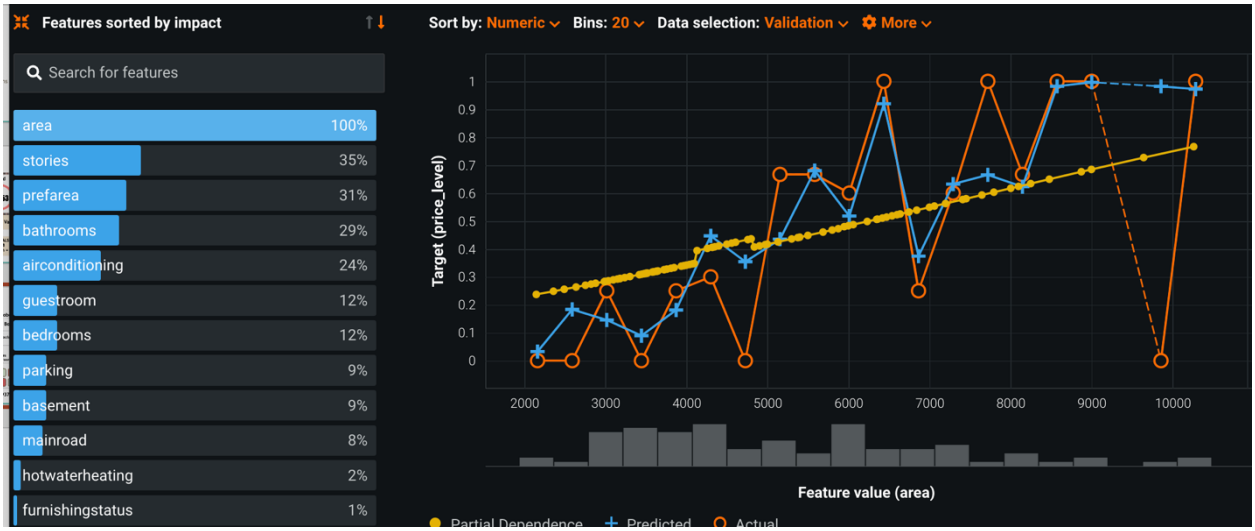
ii. AVG Blender

The second best DataRobot model is the Average Blender which has a blueprint as follows. We can see that the AVG Blender is even more complicated than the previous DataRobot model, as it is built on at least 15 different models for preprocessing, combines predictions from the Elastic-Net Classifier and the Light Gradient Boosting on ElasticNet, and returns the mean as the final prediction.

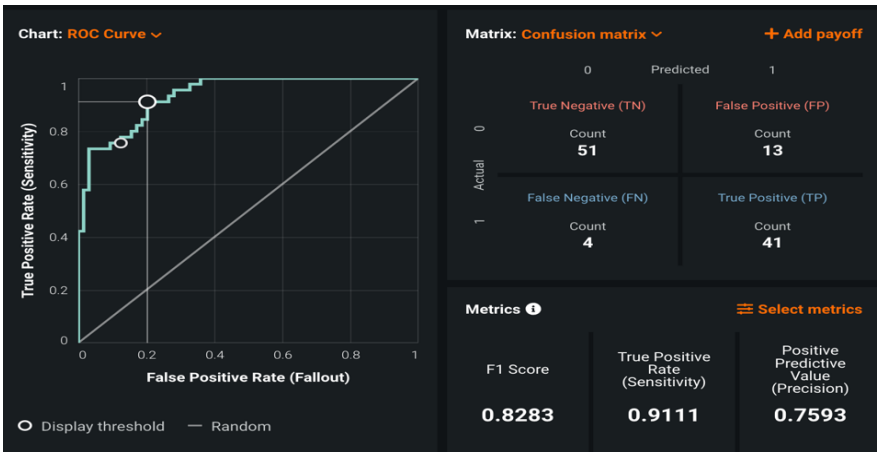


Similar to the previous DR model, this model took 63.85% of the data as the sample, and calculated the key performance metric for the holdout subset. When looking at the features sorted by impact, predictors including area (100%), stories (35%), prefarea (31%), bathrooms (29%), and air conditioning (24%) were identified as the top features that have the most predictive power. The following graph demonstrates the behavior of the top feature, area. One thing worth noting is that although the prediction and actual curves still tend to move together in the beginning, the prediction curve fails to capture the actual result when the feature value is very high, as indicated by the gap between the two curves at area value of around 9,000 to 10,000. This could explain part of the reason why the AVG Blender model is not as accurate as the previous model. The top features by importance are also slightly different compared to those of the previous model,

although there are still some common features including area, stories, bathrooms, air conditioning and so on.



To evaluate the performance of the AVG Blender, we looked at the ROC curve and the confusion matrix. The area under the ROC curve is .9354, and the F1 score is .8283. Both results were slightly lower compared to those of the previous DR model and logistic regression, the best model by hand. Since the AVG Blender model is way more complex in the model-building process and has lower F1 score, we would not choose it as our best model for usage.



C. Comparison of Best Hand-Crafted Models and Best DataRobot Models

	Best Hand-Crafted Models		Best DataRobot Models	
Model of Choice	Logistic Regression	Boosted Tree	Light Gradient Boosting on ElasticNet Predictions	AVG Blender
Feature List	Removed 6 variables	Removed 6 variables	All features (excluding price)	All features (excluding price)
Top Factors By Importance	1. air conditioning 2. hot water heating 3. guestroom 4. bathrooms	-	1. area 2. stories 3. bathrooms 4. air conditioning	1. area 2. stories 3. prefarea 4. bathrooms 5. air conditioning
F1 Score	0.9161	0.8653	0.8302	0.8283

As discussed earlier, the best hand-crafted models with the highest F1 scores are logistic regression and boosted tree, while the best DataRobot models are the Light Gradient Boosting on ElasticNet and AVG Blender. The table above demonstrates a comprehensive comparison between these models. The first difference comes from the feature list. While the best hand-crafted models removed 5 statistically insignificant variables and the price variable (to avoid collinearity), the DataRobot models included all the variables other than the price.

The difference in feature list also led to different results in terms of top factors by importance. For the top 2 DataRobot models, area is the most important factor and has the greatest predictive power. However, for logistic regression, area is not even in the top 5 important factors. Air conditioning, on the other hand, turned out to be the most important factor for the logistic regression model. The boosted tree model made by hand does not tell us anything about the top factors by importance, as we could not tell what is going on within the black boxes. Although it would be difficult to tell which factor is the most important one in our prediction, there are some

common factors among the models, such as bathrooms and air conditioning, which can provide useful information for our stakeholders. For instance, if we are building the models for real estate companies, who want to understand which factors are most important in determining house prices, then they should probably look at these common factors such as bathrooms and air conditioning, which have been identified as top factors by several best models.

Finally, comparison of the F1 scores across different models tells us which model is the best. Among all the models, logistic regression (with 19 outliers removed) has the highest F1 score of .9161, which is greater than that of the best DataRobot model, the Light Gradient Boosting on ElasticNet. If we do not care about being "fair" in terms of the model-building process and only care about building the most accurate model, then logistic regression is the way to go. It is also much simpler than the ensemble models built by DataRobot and has greater interpretability.

V. Conclusion and Reflection

Eventually, the last question to be answered is: what did we learn along this long journey? The first thing, and probably the most important lesson, is that, less is more. By "less", we are talking about less insignificant variables, outliers, and so on. By dropping some statistically insignificant variables, our accuracy improves by a huge amount, and all the remaining variables start to become significant as we exclude irrelevant variables from our regression. Dealing with outliers is another important issue, and we observed in our logistic regression model that, when we excluded a fair number of outliers from our dataset, the overall accuracy and the F1 score went up by 8-9%. Removing outliers is also a process that differentiates what humans can do from DataRobot. When

we have some outliers in our data, DataRobot might give us a reminder, but it is our choice to remove the outliers or not at the end of the day. There can be multiple ways to deal with outliers, and while humans can decide which way to go based on their understanding of the data and the business, DataRobot cannot easily achieve that. This is also the reason why we still need data analysts and data scientists who can make such decisions, even though automating tools like DataRobot can build models that are quite satisfactory for us.

Another lesson that we learned is the tradeoff between interpretability and accuracy. If we compare the best ensemble models built by DataRobot with the same type of model built by hand, we will observe that the DataRobot models tend to have higher accuracy and F1 scores than the same type of model that is hand-crafted. The reason is that the models built by DataRobot are way more complex (as shown in the blueprint earlier) than our hand-crafted models, and since they are the so-called “black-box” models, we do not really know what is happening inside the black boxes and thus interpretability is highly limited. That being said, if we are trying to build a model that has fair accuracy and needs to be interpreted well to derive insights about the business, we should be careful in choosing these complex, “black-box” models.