STUDY OF THE HOUSING MARKET IN THE CITY OF NEW YORK

Diego Rossi

mat. 1073945

Andrea Rota

mat. 1054128

1. INTRODUCTION

Understanding the housing market dynamics in a bustling metropolis like New York City is critical for stakeholders, ranging from potential homebuyers and investors to urban planners and policymakers. The housing market in New York City is characterized by its complexity, influenced by diverse factors such as economic trends, demographic shifts, regulatory policies, and geographic location. In this study, we aim to dissect these complexities and provide a comprehensive analysis of the New York City housing market.

This paper investigates two main questions within the context of New York City's real estate sector. First, we explore the factors that significantly influence the price of a house, utilizing various statistical and machine learning models to predict house prices based on a range of features. Second, we examine the potential to predict the sublocality of a house within New York City based on other variables, which can be particularly valuable for individuals and businesses looking to make location-specific real estate decisions.

In the subsequent sections, we will delve into the specifics of the dataset, outline the parameters used, and describe the methodologies applied for data cleaning, feature selection, and model implementation. We will also discuss the results obtained from various predictive models, comparing their performance and interpreting their implications for the housing market in New York City.

By the end of this study, we aim to offer insights that can aid in making informed real estate decisions, contributing to a deeper understanding of the market forces at play in one of the world's most dynamic urban environments.

All the code used in this study is available on GitHub in the repository "statistical-learning-unibg" [1] under an MIT license.

2. DATASET

This dataset, sourced from Kaggle [2], provides comprehensive insights into the New York real estate market, featuring house prices alongside crucial details such as broker titles, property types, bedroom and bathroom counts, square footage, addresses, and geographical coordinates. With data spanning administrative and local areas, this resource offers a valuable foundation for analyzing trends and making informed decisions within the region's housing sector. The dataset comprises 4802 observations.

2.1. Parameters

The dataset consists of 17 variables presented below:

1. **BROKERTITLE** *(cat)*: Title of the broker
2. **TYPE** *(cat)*: Type of the house
3. **PRICE** *(num)*: Price of the house
4. **BEDS** *(num)*: Number of bedrooms
5. **BATH** *(num)*: Number of bathrooms
6. **PROPERTYSQFT** *(num)*: Square footage of the property
7. **ADDRESS** *(cat)*: Full address of the house
8. **STATE** *(cat)*: State of the house
9. **MAIN_ADDRESS** *(cat)*: Main address information
10. **ADMINISTRATIVE_AREA_LEVEL_2** *(cat)*: Administrative area level 2 information
11. **LOCALITY** *(cat)*: Locality information
12. **SUBLOCALITY** *(cat)*: Sublocality information
13. **STREET_NAME** *(cat)*: Street name
14. **LONG_NAME** *(cat)*: Long name
15. **FORMATTED_ADDRESS** *(cat)*: Formatted address
16. **LATITUDE** *(num)*: Latitude coordinate of the house
17. **LONGITUDE** *(num)*: Longitude coordinate of the house

In Figure 1 and Figure 2 distributions of the numerical variables can be observed.

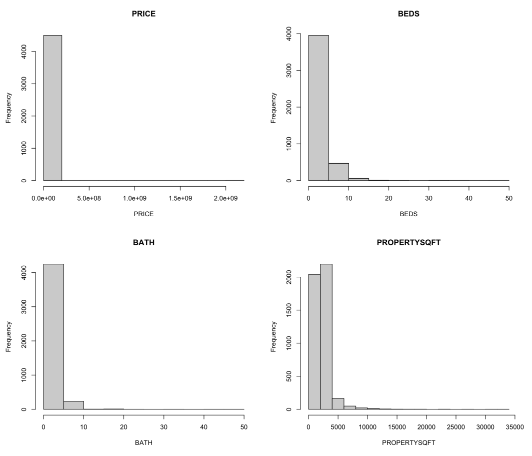


Figure 1: Bar plot of selected numerical variables, showing their distribution and comparison.

3. QUESTIONS

In this study, we address two key questions related to the real estate market in New York City. First, we aim to predict the price of a house based on various features. Second, we focus on predicting the sublocality of a house using other relevant variables. These analyses can provide valuable insights for potential homebuyers and real estate professionals navigating the complex and diverse housing market of New York City.

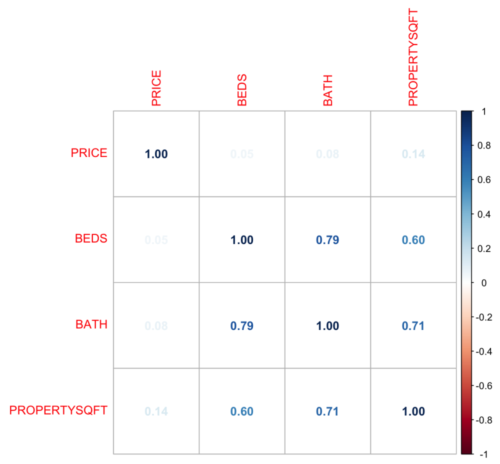


Figure 2: Correlation plot illustrating relationships between variables.

4. PREDICTING THE PRICE OF A HOUSE BASED ON OTHER VARIABLES

4.1. Data Cleaning

Before we started cleaning the data, our dataset had 4,802 observations and 17 variables. We began by checking for any missing data and subsequently removed parameters that were not relevant to the current study. By the end of this initial phase, we confirmed that there were no missing values. Next, we turned our attention to the columns related to address and location information. We decided to eliminate several of these columns: ADDRESS, STATE, MAIN_ADDRESS, ADMINISTRATIVE_AREA_LEVEL_2, LOCALITY, STREET_NAME, LONG_NAME, FORMATTED_ADDRESS, LATITUDE and LONGITUDE. Only the SUBLOCALITY column was retained, as it provided the ideal balance between specificity and the number of unique values (21), fitting well with the focus of our study. Since the variables PRICE, BATH, BEDS, and PROPERTYSQFT contained many outliers, as shown in Figure 1, we decided to apply a filtering process to remove them. Using the interquartile range (IQR) technique, we calculated the quartiles and then determined the lower and upper limits for each variable.

```
IQR <- Q3-Q1
upper_limit <- Q3+2*IQR
lower_limit <- Q1-2*IQR
```

To get more unique values for the BATH variable, we decided to use 2*IQR instead of the usual 1.5*IQR. This adjustment facilitates the application of a spline to the variable in the GAMs model. Observations falling outside these revised limits were removed from the dataset, ensuring the reliability and consistency of our analyzed data by minimizing the influence of outliers. Next, we turned the categorical variables into dummy variables. This means we changed the original categories into binary columns (0 or 1) and then removed the original categorical columns to avoid multicollinearity. Since we ended up with a lot of variables, mostly due to the dummy variables, we decided to group the less common values for SUBLOCALITY, TYPE, and BROKERTITLE. For any category that appeared less frequently than a set threshold, we replaced it with a new category called "Others." Specifically, for BROKERTITLE, we grouped categories appearing fewer than 10 times; for SUBLOCALITY and TYPE, we grouped categories appearing fewer than 4 and 5 times, respectively. This step reduced the number of variables from 963 to 120, adding three new variables: BROKERTITLE_Others, SUBLOCALITY_Others, and TYPE_Others. As a result, our dataset comprised 120 variables with 3,820 observations.

4.2. Methods

For the analysis, we used the following models:

- Linear regression (Section 4.2.1).
- GAMs model (Section 4.2.2).
- Ridge regression (Section 4.2.5).
- Lasso regression (Section 4.2.6).

For validation, we split the dataset into a training set (70% of the dataset) and a test set (30% of the dataset). Finally, to compare the models, we chose to calculate the RMSE for each model on the test set and the correlation between the actual values of the test set and the predictions made by the models.

4.2.1. Linear Regression

To begin, we chose to use a linear regression model. To make the PRICE less variable in terms of values, we decided to apply a logarithmic transformation.

```
lm_fit <- lm(log(PRICE) ~ ., data = df[train,])
```

```
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.4284 on 2558 degrees of freedom
Multiple R-squared:  0.6513,	Adjusted R-squared:  0.6356
F-statistic: 41.54 on 115 and 2558 DF,  p-value: < 2.2e-16
```

Figure 3: Linear regression results.

The results shown in the Figure 3 indicate that the regression model is statistically significant ($p-value < 2.2e-16$), with an R-squared of 0.6513, suggesting that approximately 65.13% of the variability in the data is explained by the model.

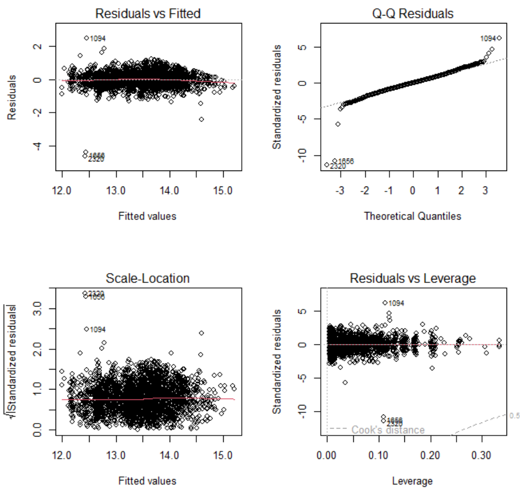


Figure 4: Residuals plots.

In the Figure 4, the "Residuals vs Fitted" plot shows that the residuals are randomly distributed around the horizontal line. Additionally, the "Q-Q Residuals" plot indicates that the residuals approximately follow a normal distribution, except for some deviations at the tails. In the "Residuals vs Leverage" plot, there are a few points with high leverage that could potentially influence the model.

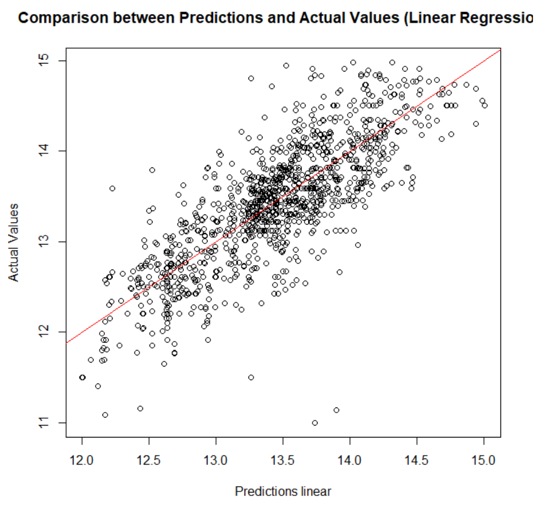


Figure 5: Plot of actual values vs model predictions (Linear).

The linear model has a correlation between the model's predictions and the actual values equal to 0.795 (calculated with the logarithmic transformation on PRICE), and the test RMSE is 398534 $. While conducting predictions on the test set, we encountered a multicollinearity issue among the model predictors, a topic we will delve into Section 4.2.3.

4.2.2. GAMs

To capture potential nonlinear relationships between the variables, we decided to use a GAMs model.

```
gam_model <- gam(log(PRICE) ~ s(BEDS,4) +
    s(BATH,4)+poly(PROPERTYSQFT,4)+., data =
    df[train,]);
```

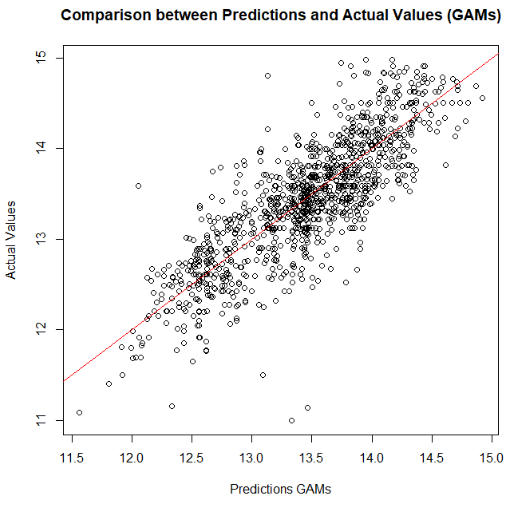


Figure 6: Plot of actual values vs model predictions (GAMs).

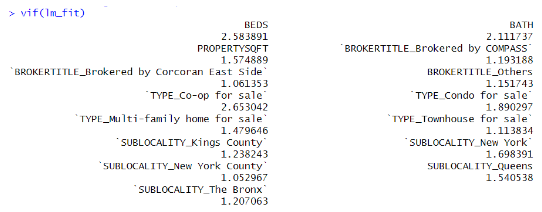
The GAMs model has a correlation between the model's predictions and the actual values equal to 0.826 (calculated with the logarithmic transformation on PRICE), and the test RMSE is 371022 $. While conducting predictions on the test set, we encountered a multicollinearity issue among the model predictors, a topic we will delve into Section 4.2.3.

4.2.3. Multicollinearity problem

While attempting to generate predictions using the `predict()` function with both the linear regression and GAMs model, we encountered a warning message indicating rank-deficient fits and doubtful cases within the "non-estim" attribute. Initially, multicollinearity was suspected. However, after examining the correlation matrix, no significant correlations among predictors were found. Further investigation involved calculating VIF values for each variable on the linear model, during which RStudio flagged an error related to aliased coefficients within the model. Utilizing the `alias()` function, we identified and removed the problematic variables: `BROKERTITLE_Others`, `SUBLOCALITY_Others`, and `TYPE_Others`. Upon recalculating the VIF values, certain variables exhibited exceptionally large values. To address this issue, we implemented a predictor selection strategy based on their correlation values with the `PRICE` variable (We removed the variables with correlation between -0.1 and +0.1). Our refined dataset now comprised 16 variables and 3820 observations.

Finally, we reapplied the linear regression and GAMs models to the refined dataset. The new linear model has a correlation between the model's predictions and the actual values equal to 0.770

(calculated with the logarithmic transformation on `PRICE`), and the test RMSE is 410627 \$. The new GAMs model has a correlation between the model's predictions and the actual values equal to 0.803 (calculated with the logarithmic transformation on `PRICE`), and the test RMSE is 378639 \$. By opting for the most significant variables within the dataset, we successfully avoided encountering any warning messages during predictions. Additionally, the subsequently calculated VIF values for the linear model were all below 5, as shown in Figure 7 .



```
> vif(lm_fit)
                                        BEDS                                        BATH 
                                    2.583891                                    2.111737 
                                PROPERTYSQFT          `BROKERTITLE_Brokered by COMPASS` 
                                    1.574889                                    1.193188 
`BROKERTITLE_Brokered by Corcoran East Side`                          BROKERTITLE_Others 
                                    1.061353                                    1.151743 
                       `TYPE_Co-op for sale`                       `TYPE_Condo for sale` 
                                    2.653042                                    1.890297 
           `TYPE_Multi-family home for sale`                   `TYPE_Townhouse for sale` 
                                    1.479646                                    1.113834 
                 `SUBLOCALITY_Kings County`                    `SUBLOCALITY_New York` 
                                    1.238243                                    1.698391 
              `SUBLOCALITY_New York County`                         SUBLOCALITY_Queens 
                                    1.052967                                    1.540538 
                  `SUBLOCALITY_The Bronx` 
                                    1.207063 
```

Figure 7: VIF values (linear regression) in the new dataset.

4.2.4. *Shrinkage Methods*

We chose to use Lasso regression and Ridge regression as another strategy to tackle the problem of collinearity between predictors in the dataset, allowing us to avoid modifying the dataset. These methods apply a penalty to the regression coefficients, causing them to tend towards zero. Since multicollinearity can lead to instability in coefficient estimates, the addition of a penalty promotes the estimation of more stable and generalizable coefficients.

4.2.5. *Ridge Regression*

```
x<-model.matrix(PRICE ~ ., df)
y<-log(df$PRICE)
cv_model <- cv.glmnet(x[train, ], y[train], alpha = 0,
      nfolds = 10)
opt_lambda <- cv_model$lambda.min
model <- glmnet(x[train,], y[train], alpha = 0, lambda
      = opt_lambda, standardize = TRUE)
```

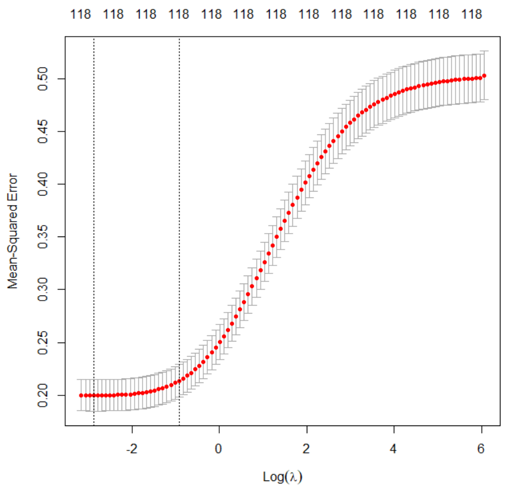


Figure 8: Plot of the lambda in Ridge Regression.

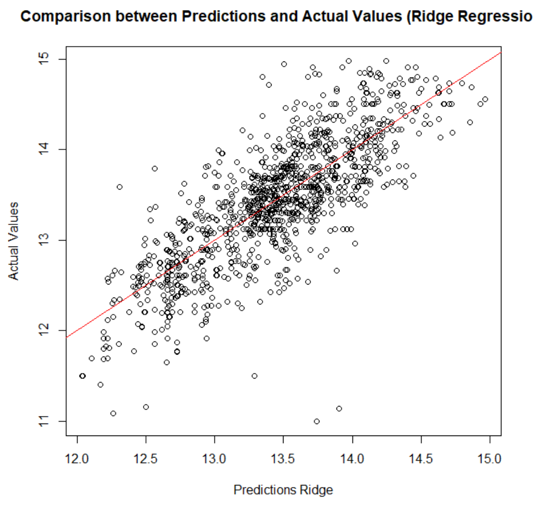


Figure 9: Plot of actual values vs model predictions (Ridge).

The Ridge regression model has an optimal lambda of 0.046, as shown in Figure 8. The correlation between the model's predictions and the actual values is 0.795 (calculated with the logarithmic transformation on `PRICE`), and the test RMSE is 397415 \$. We didn't encounter any warning messages during the forecasts.

4.2.6. *Lasso Regression*

```
cv_lasso <- cv.glmnet(x[train,],y[train],alpha=1,nfolds
      = 10);
opt_lambda <- cv_lasso$lambda.min
model <- glmnet(x[train,],y[train],alpha = 1,lambda =
      opt_lambda, standardize = TRUE)
```

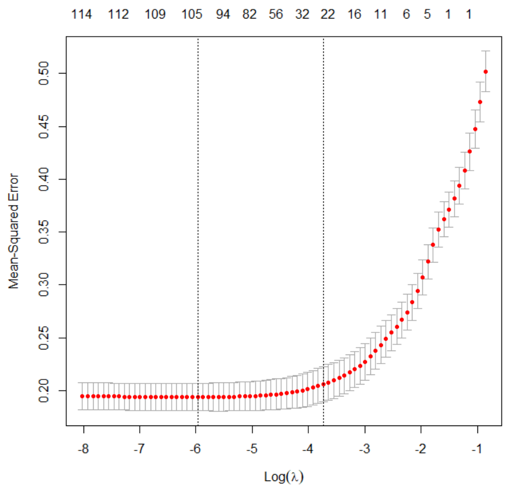


Figure 10: Plot of the lambda in Lasso Regression.

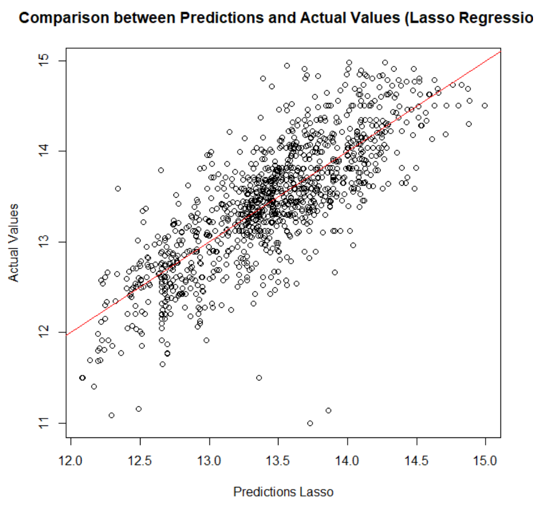
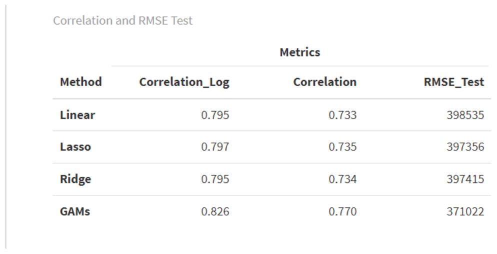


Figure 11: Plot of actual values vs model predictions (Lasso).

The Lasso regression model has an optimal lambda of 0.04, as shown in Figure 10. The correlation between the model's predictions and the actual values equal to 0.797 (calculated with the logarithmic transformation on `PRICE`), and the test RMSE is 397355 $. We didn't encounter any warning messages during the forecasts.

4.3. Results



Correlation and RMSE Test

	Metrics		
Method	**Correlation_Log**	**Correlation**	**RMSE_Test**
Linear	0.795	0.733	398535
Lasso	0.797	0.735	397356
Ridge	0.795	0.734	397415
GAMs	0.826	0.770	371022

Figure 12: Performance on the initial dataset.

In conclusion, among the models tested, the GAMs model demonstrated the best overall performance. With a correlation of 0.825 and an RMSE of 371021 $ on the initial dataset, and a correlation of 0.803 and an RMSE of 378639 $ on the refined dataset, the GAMs outperforms both the linear model and the Lasso and Ridge regression models. The linear model, despite showing good initial performance with a correlation of 0.795 and an RMSE of 398534 $, experienced a slight decline after removing problematic variables (correlation of 0.770 and RMSE of 410627 $). Ridge and Lasso regression, while improving coefficient stability through applied penalties and not suffering from multicollinearity, did not achieve the precision of the GAMs, with correlations of 0.795 and 0.797 respectively, and an RMSE around 397000 $. Therefore, the GAMs model proves to be the most effective for price prediction. Its high flexibility allow it to better adapt to the data configuration, capturing potential nonlinear relationships within the data.

5. PREDICTING THE PRICE OF A HOUSE BASED ON OTHER VARIABLES

In this section, we will address the second question, which focuses on predicting the sub-locality of a house based on various other variables. We chose this question because New York is a vast and diverse city, and building a model that helps people understand where to start looking for a house based on specific variables can be incredibly valuable. Such a model can assist in making informed decisions, tailored to individual needs and preferences, within the complexity of New York's many neighborhoods.

5.1. Data cleaning

Before the data cleaning process, the dataset contained 4802 observations and 17 variables. Initially, a check was performed to identify any missing data, followed by the removal of parameters not relevant to the current study. At the end of this phase, it was found that there were no missing values. Subsequently, columns related to the address (`ADDRESS`, `STATE`, `MAIN_ADDRESS`, `ADMINISTRATIVE_AREA_LEVEL_2`, `LOCALITY`, `STREET_NAME`, `LONG_NAME`, `FORMATTED_ADDRESS`) were eliminated, retaining only the `SUBLOCALITY` column. In addition, the `LATITUDE`, `LONGITUDE` columns have been eliminated since they contain the value to be predicted, and the `BROKERTITLE` column.

Given the distribution of numerical variables in Figure 1, we decided to handle potential outliers in the dataset using the Interquartile Range (IQR) method with a multiplier of 1.5.

Following the data filtering process, additional steps were taken to enhance the dataset's suitability for analysis. Initially, the categorical variables `SUBLOCALITY` and `TYPE` were converted into factor variables using the `as.factor()` function. This transformation ensures that these variables are treated as categorical factors rather than continuous variables, making them suitable for inclusion in statistical models and analyses.

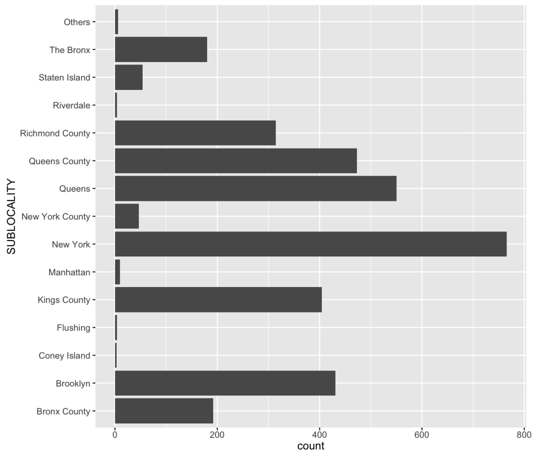


Figure 13: Bar plot illustrating the distribution of sublocality values.

To explore the hidden correlation between property characteristics, two interaction variables were created: `beds-bath`, representing the interaction between the number of bedrooms and bathrooms, and `bath-property_square_feet`, representing the interaction between the number of bathrooms and the property's square footage.

Subsequently, within the SUBLOCALITY variable, values that appeared less than three times—indicative of rare occurrences—were identified. To address this issue, a new category labeled "Others" was created to encompass these infrequent occurrences. This consolidation provides a simplified representation of less common sub-localities while retaining meaningful information, thus enhancing the clarity and interpretability of the variable for subsequent analyses.

After completing the data cleaning process, the dataset now contains 3438 observations and 8 variables. In Figure 13, we can see the distribution of sub-locality values.

5.2. Methods

To address class imbalance in the SUBLOCALITY variable, the Synthetic Minority Over-sampling Technique (SMOTE) was applied using the `performanceEstimation::smote()` function. SMOTE generates synthetic samples for the minority class (SUBLOCALITY) by interpolating new instances based on the characteristics of existing data points. In this application, the parameters `perc.over` and `perc.under` were specified to adjust the balance between classes. Specifically, `perc.over` was set to 25% and `perc.under` was set to 50%.

After applying SMOTE, the resulting dataset underwent a process to remove any remaining missing values. This data cleaning step ensures that the dataset is complete and free of missing values, preparing it for subsequent analysis tasks.

The values specified for `perc.over` and `perc.under` were chosen deliberately to avoid overfitting the models. Tests with higher values, such as 50/100, had resulted in model accuracy as high as 95/97%, indicating overfitting to the training data. This balanced parameter choice aims to improve dataset representativeness without compromising the generalization of the analysis models.

The dataset was divided into training and test sets with a 70/30 split. All the models discussed in this section are trained using repeated cross-validation with 3 folds and 10 repeats, and they are configured with specific parameters:

- `ntree = 500`
- `tuneLength = 10`, this tuning process will explore 10 different hyperparameter settings to find the optimal configuration.

This thorough training and tuning process aims to enhance the accuracy and generalization capability of the models.

5.2.1. *Classification tree*

The first model trained is a classification tree. A maximum depth (`maxdepth`) of 12 was chosen for the tree, as it achieved a peak accuracy of 42.6% during the training phase, as shown in Figure 14. Figure 15 presents the visualization of the decision tree structure.

When testing the model with the test dataset, an accuracy of 44.7% was achieved. The confusion matrix for this model, displayed in Figure 16, provides further insights into the model's performance, illustrating how well the model distinguishes between different classes.

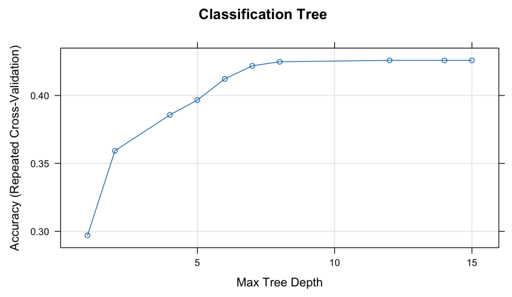


Figure 14: Training accuracy of the classification tree model.

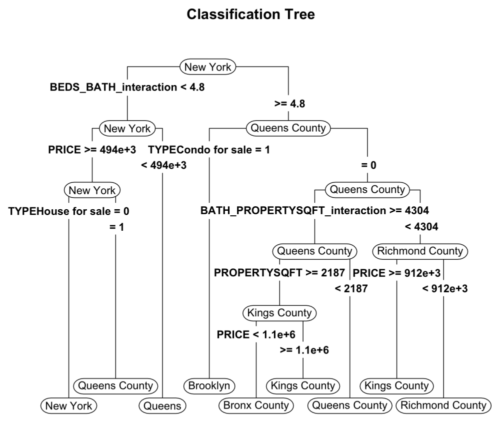
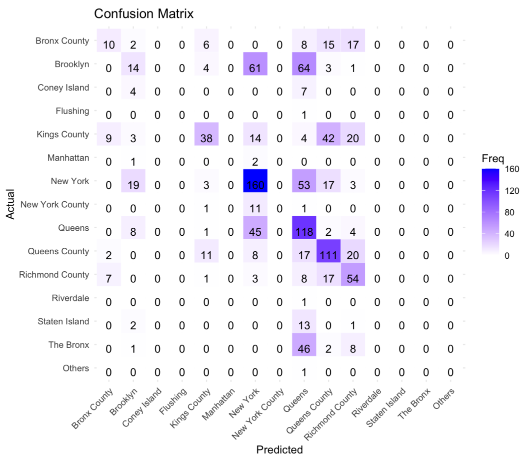


Figure 15: Visualization of the decision tree structure.



	Bronx County	Brooklyn	Coney Island	Flushing	Kings County	Manhattan	New York	New York County	Queens	Queens County	Richmond County	Riverdale	Staten Island	The Bronx	Others
Bronx County	10	2	0	0	6	0	0	0	8	15	17	0	0	0	0
Brooklyn	0	14	0	0	4	0	61	0	64	3	1	0	0	0	0
Coney Island	0	4	0	0	0	0	0	0	7	0	0	0	0	0	0
Flushing	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0
Kings County	9	3	0	0	38	0	14	0	4	42	20	0	0	0	0
Manhattan	0	1	0	0	0	0	2	0	0	0	0	0	0	0	0
New York	0	19	0	0	3	0	160	0	53	17	3	0	0	0	0
New York County	0	0	0	0	1	0	11	0	1	0	0	0	0	0	0
Queens	0	8	0	0	1	0	45	0	118	2	4	0	0	0	0
Queens County	2	0	0	0	11	0	8	0	17	111	20	0	0	0	0
Richmond County	7	0	0	0	1	0	3	0	8	17	54	0	0	0	0
Riverdale	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0
Staten Island	0	2	0	0	0	0	0	0	13	0	1	0	0	0	0
The Bronx	0	1	0	0	0	0	0	0	46	2	8	0	0	0	0
Others	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0

Figure 16: Confusion matrix of the classification tree model.

5.2.2. *Random forest*

The second model trained is a random forest. A `mtry` value of 16 was chosen for the model, as it achieved a peak accuracy of 71.9% during the training phase, as shown in Figure 17.

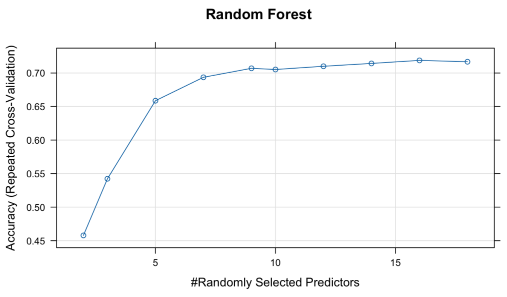


Figure 17: Training accuracy of the random forest model.

To gain insights into the model's performance and interpretability, we visualized the variable importance in Figure 18 and the error metrics of the trained random forest model in Figure 19.

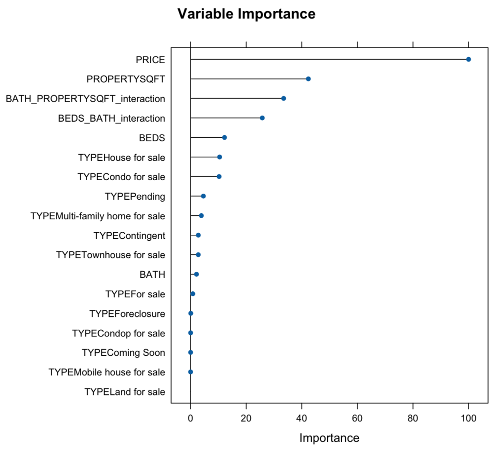


Figure 18: Variable importance plot for the random forest model.

When testing the model with the test dataset, an accuracy of 75.0% was achieved. The confusion matrix for this model, displayed in Figure 20, provides further insights into the model's performance, illustrating how well the model distinguishes between different classes.

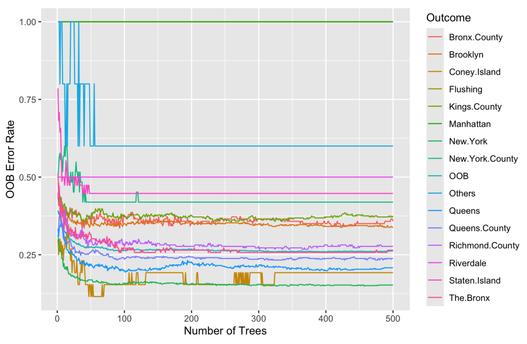


Figure 19: Error metrics plot for the random forest model.

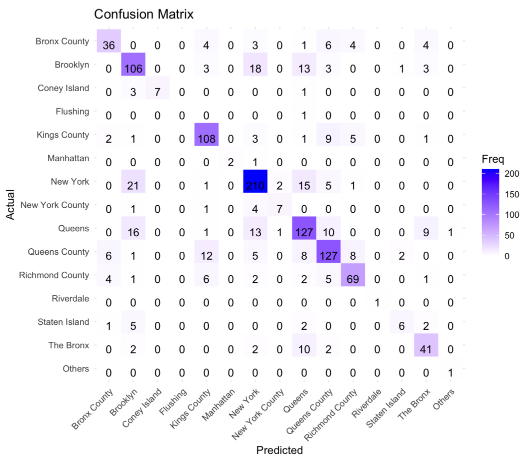


Figure 20: Confusion matrix of the random forest model.

5.2.3. *Bagging*

The third model trained is a bagging model. Seven predictors were selected for the model, as it achieved a peak accuracy of 71.1% during the training phase; the variable importance is visualized in Figure 21.

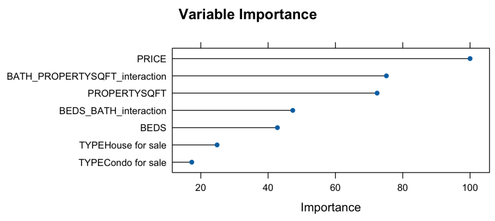


Figure 21: Variable importance plot for the bagging model.

When testing the model with the test dataset, an accuracy of 75.1% was achieved and the confusion matrix for this model, displayed in Figure 22, provides further insights into the model's performance, illustrating how well the model distinguishes between different classes.

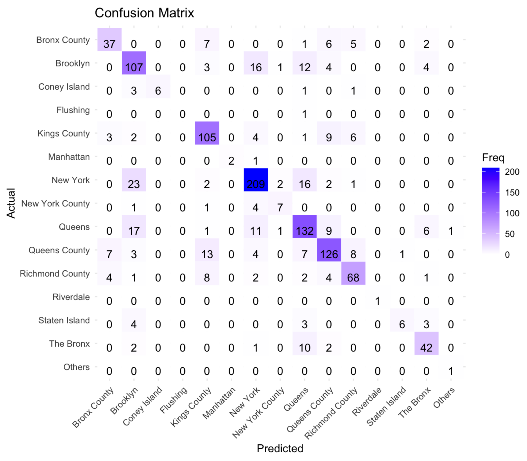


Figure 22: Confusion matrix of the bagging model.

5.2.4. *Boosting*

The fourth model trained is a boosting model. We used this configuration with the `caret` package:

```
gbmGrid <-  expand.grid(
    interaction.depth = 4,
    n.trees = ntree,
    shrinkage = 0.01,
    n.minobsinnode = 10
)
```

It achieved a peak accuracy of 57.2% during the training phase and an accuracy of 59.8% was achieved with the training dataset, the confusion matrix for this model is displayed in Figure 23.

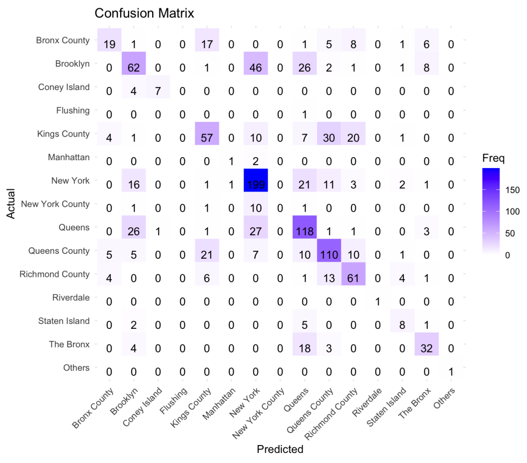


Figure 23: Confusion matrix of the boosting model.

5.2.5. *Support Vector Machine*

The fourth model trained is a Support Vector Machine (SVM) with a radial basis function kernel. Prior to training the model, a preprocessing step was defined using the `preProcess()` function with methods `"center"`, `"scale"`, and `"zv"`. This preprocessing ensures that the data is centered and scaled, and that zero-variance predictors are removed, enhancing the performance and stability of the SVM model, the preprocessor was then applied to both the training and testing datasets.

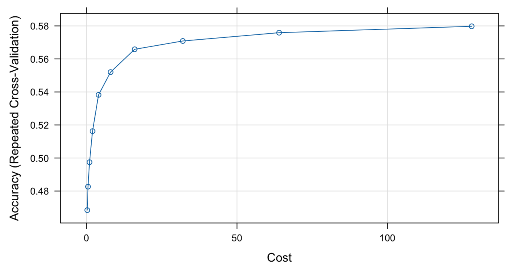


Figure 24: Training accuracy of the SVM model.

A `cost` value of 128 was chosen for the model, as it achieved a peak accuracy of 57.7% during the training phase, as shown in Figure 24 and when testing the SVM model with the test dataset, it achieved an accuracy of 59.1%; the confusion matrix for the SVM model, displayed in Figure 25, provides further insights into the model's performance, illustrating how well the model distinguishes between different classes.

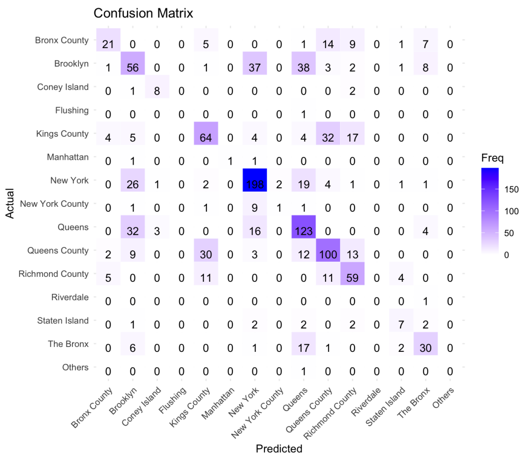


Figure 25: Confusion matrix of the SVM model.

5.3. Discussion and interpretation of the results

The performance of the various models in predicting the sublocality of a house varied significantly, reflecting their respective strengths and limitations. The classification tree model, with an accuracy of 44.7%, demonstrated limited ability to generalize, likely due to its simplicity and sensitivity to the specific structure of the training data. In contrast, the random forest model achieved a much higher accuracy of 75.0%, highlighting its robustness and capacity to capture complex relationships in the data through ensemble learning and feature randomness. Similarly, the bagging model performed comparably well with an accuracy of 75.1%, underscoring the benefits of reducing variance through multiple iterations of decision trees.

The boosting model, while effective in handling bias, achieved a moderate accuracy of 59.8%. This result suggests that while boosting can enhance prediction accuracy by focusing on difficult-to-predict instances, it may be more sensitive to overfitting, particularly with the chosen parameters. The Support Vector Machine (SVM) model, achieving an accuracy of 59.1%, demonstrated reasonable performance, benefiting from the preprocessing steps that standardized the data. However, its lower accuracy relative to the ensemble methods suggests that SVM might not fully capture the non-linear relationships present in the dataset.

Overall, the ensemble methods—random forest and bagging—provided the best performance, indicating their effectiveness in handling the diversity and complexity of the features in predicting sub-localities. These results emphasize the importance of model selection and parameter tuning in achieving optimal predictive performance. The higher accuracy of these models suggests that ensemble techniques, which combine multiple decision trees to reduce variance and improve generalization, are particularly well-suited for this type of classification problem. A graphical comparison of the model accuracies is shown in Figure 26.

It is also important to note that the imbalanced nature of the dataset posed a challenge in model construction. This issue was partially addressed through the application of the Synthetic Minority Over-sampling Technique (SMOTE), which helped to balance the class distribution by generating synthetic samples for the minority classes. Despite this challenge, it was still possible to construct models, particularly the random forest and bagging models, which can assist future house buyers with no experience in the New York real estate market by directing them to the correct area of the city from the start. This demonstrates the practical utility of these models in making informed decisions within the complex landscape of New York's many neighborhoods.

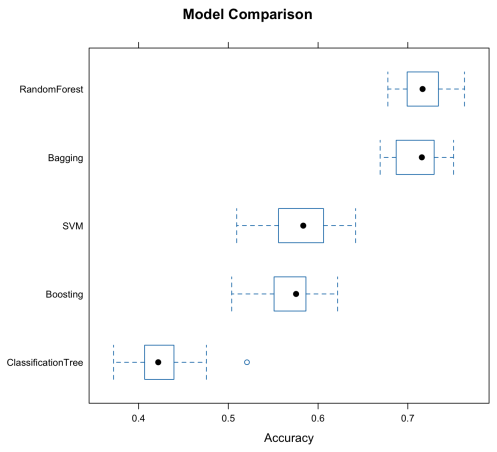


Figure 26: Comparison of model accuracies for predicting sub-locality.

6. REFERENCES

[1] "andrearoota/statistical-learning-unibg: Statistical learning project for the master's degree course in computer engineering at the university of bergamo," https://github.com/andrearoota/statistical-learning-unibg, 2024 (accessed Jun 6, 2024).

[2] "New york housing market," www.kaggle.com, 2024 (accessed May 3, 2024).