Statistical Modeling and Analysis for Apartment Rent

Introduction

Renting an apartment in the United States can often feel like navigating a maze of uncertainty. Rental prices fluctuate widely not only across states and cities but also between neighborhoods, building types, and amenities offered. For many individuals — students seeking affordable housing, professionals relocating to major urban centers, and families struggling with rising costs — understanding what drives rental prices is essential. As urbanization continues and housing affordability becomes a mounting challenge, there is growing interest in using data-driven methods to better understand and predict rental prices.

From a research standpoint, housing price prediction has received considerable attention in economics, urban planning, and real estate analytics. Early foundational work by **Glaeser and Gyourko (2005)** explored the economic determinants of urban housing affordability, highlighting the role of supply constraints and regional market forces. More recently, **Kok et al. (2017)** analyzed how green building certifications affect rental premiums, reflecting a growing interest in amenity-driven pricing. Furthermore, **Dubé, Legros, and Thériault (2014)** emphasized the spatial structure of housing markets, demonstrating that location-based models can dramatically improve rental predictions. These studies reinforce the idea that rental prices are driven by a combination of structural features, local market trends, and geographic patterns — motivating the use of flexible modeling techniques that can capture these complex relationships.

In this report, we apply a series of statistical and machine learning methods to the **"Apartments for Rent in US"** dataset, which is publicly available via the **UC Irvine Machine Learning Repository**. The dataset was collected in **December 2018** and compiles real-world listings from multiple sources including **RentLingo**, **Zumper**, **Craigslist**, and **HotPads**, providing a cross-sectional snapshot of the U.S. rental housing market. Despite being slightly dated, the dataset remains valuable for methodological experimentation and modeling, as it contains a wide variety of property types and listing contexts.

The dataset includes over **25,000 apartment listings**, each with structured attributes such as:

- **Numerical variables**: square footage, number of bedrooms and bathrooms, rental price.
- **Categorical variables**: city name, state, pet policy, amenity availability.
- **Geospatial variables**: latitude and longitude.
- **Textual data**: description titles and body text, which we excluded from quantitative modeling for this analysis.

To gain an initial understanding of the data, we performed exploratory data analysis (EDA), including variable distributions, scatterplots, and pairwise correlations. We found that **square footage**, **bedroom count**, and **bathroom count** were the most strongly correlated numerical variables with price, while **city/state** and **latitude/longitude** revealed substantial geographic variation in rental patterns. Certain amenities, such as in-unit laundry or gym access, also showed association with higher rental prices.

Our primary goal in this project is to investigate whether rental prices can be predicted accurately based on available listing features, and to assess which models provide both predictive power and interpretability. We employed a range of models — from classical linear regression to more flexible nonlinear techniques:

- **Linear regression** to establish a baseline.
- **Smoothing splines and Generalized Additive Models (GAMs)** to account for nonlinear trends in variables like size and location.
- **Random forests** to capture complex interactions and non-additive effects among predictors.
- **Support Vector Machines (SVM)** to classify listings as priced above or below the median — a practical classification problem relevant to renters.

This multi-model approach allows us not only to benchmark different techniques but also to understand the trade-offs between model complexity, interpretability, and predictive accuracy. The findings aim to inform renters, landlords, and policymakers about the relative importance of structural and geographic factors in rental pricing, while showcasing how statistical learning techniques can provide actionable insights in real estate analytics.

Data

We considered the “Apartments for Rent in US” dataset from the UC Irvine Machine Learning Repository. The dataset comprises thousands of apartment rental listings collected in December 2018 from multiple sources, including:

- RentLingo
- Craigslist
- Zumper
- HotPads
- PadMapper
- ApartmentFinder

The dataset includes listings from various cities and states across the United States, making it geographically diverse and representative of multiple housing markets. It contains both structured (e.g., number of bedrooms, square footage, price) and unstructured (e.g., textual descriptions, amenities lists) data, allowing for rich exploratory and predictive analyses

Variables

Categorical

- **source**: Source website from which the listing was scraped
- **state**: US state abbreviation
- **cityname**: City name
- **pets_allowed**: Type of pets allowed (e.g., dogs, cats, both, none)
- **price_type**: Price currency and payment type (e.g., per month)
- **amenities**: List of amenities (e.g., air conditioning, gym, refrigerator)

Boolean

- **fee**: Indicates whether a broker fee is required
- **has_photo**: Whether the listing contains photos

Text

- **title**: Title text of the apartment listing
- **body**: Detailed description of the listing
- **address**: Complete address of the apartment
- **price_display**: Human-readable version of the price

Ordinal

- **bedrooms**: Number of bedrooms in that apartment
- **bathroom**: Number of bathrooms in that apartment

Continuous

- **square_feet**: Size of the apartment in square feet
- **longitude**: Geographic coordinate (longitude)
- **latitude**: Geographic coordinate (latitude)
- **time**: Timestamp of when the listing was created
- **price**: Rental price (response variable)

Methods

Data Preprocessing

We are using only these columns, amenities, bathrooms, bedrooms, pets_allowed, price, sq_feet, city, state and discarding the others. Lets see what preprocessing we have done for each column.

Amenities

The Amenities column originally contained multiple amenities in a single string, separated by commas (e.g., "Gym, Pool, Parking"). To make this information more usable for modeling, we performed a multi-label one-hot encoding. Specifically, we:

- Identified all 27 unique amenities present across the dataset.
- Created a new binary column for each amenity, indicating "Yes" or "No" in each listing.

As a result, the original Amenities column was transformed into 27 individual columns — one for each amenity — enabling the model to capture detailed information about each property's features.

The 27 unique features are: Parking, Dishwasher, Pool, Refrigerator, Patio/Deck, Cable or Satellite, Storage, Gym, Internet Access, Clubhouse, Garbage Disposal, Washer Dryer, Fireplace, Playground, AC, Elevator, Tennis, Gated, Wood Floors, Hot Tub, Basketball, TV, View, Doorman, Alarm, Golf, Luxury.

Bathrooms

Since an apartment cannot realistically have zero bathrooms, any missing values in the bathrooms column were replaced with 1. There were 34 such rows.

Bedrooms

Since there can be an apartment with no bedrooms, any missing values in the bedrooms column were replaced with 0. There were 7 such rows which were studio apartments or hotel rooms.

Pets_allowed

This column contained four possible values: "Cats,Dogs", "Cats", "Dogs", and NULL. Based on these entries, we created two new columns — Cats and Dogs — with values "Yes" or "No" to indicate whether each pet type is allowed for a given listing.

City name

Replaced the null rows with "Others". Also created a new column called rent_tire, where we grouped them into three tiers based on their average rent values:

- Tier 1: avg_rent > $3500
- Tier 2: $1500 ≤ avg_rent ≤ $3500
- Tier 3: avg_rent < $1500

In addition, we created a new column(cityname_te) using target encoding, where each city was replaced with the average price for that city.

State

We also applied target encoding to the state column, replacing each state with the average price of listings within that state.

Latitude and Longitude

There were 10 null rows in these two columns. We removed these rows from our analysis.

Random Forest

To address the problem of predicting apartment prices, I employed a Random Forest regression model — an ensemble learning technique that builds multiple decision trees and aggregates their results for improved accuracy and generalizability.

Data Splitting

The dataset was randomly split into 70% training and 30% testing sets to ensure unbiased evaluation.

Model Implementation

A Random Forest model was trained on the training set using:

- ntree = 500: to build 500 decision trees.
- mtry = sqrt(p): where p is the number of predictors, as commonly recommended for regression tasks.

The model was configured to calculate variable importance, helping identify which features most influence price predictions.

Model Evaluation

Predictions were generated for the test set. Root Mean Squared Error (RMSE) was used as the primary performance metric, providing a measure of average prediction error in the same units as the target variable (price).

A plot was generated to visually inspect which features had the greatest predictive power called variable importance. This plot shows the most influential features in predicting apartment prices using the Random Forest model.

- %IncMSE: Measures how much prediction error increases when a variable is permuted. Higher values indicate more important variables.
- IncNodePurity: Reflects each variable's total contribution to reducing variance across the trees.

In both plots, features like rent_tier, square_feet, bedrooms, and bathrooms are the most important, while variables such as Doorman, Tennis, and Gated contribute the least.

Hyperparameter tuning

To improve prediction accuracy, the goal was to find the optimal mtry value — the number of predictors randomly selected at each split. A tuning grid was created to explore a range of mtry values, starting from 2 up to twice the square root of the number of predictors. A 5-fold cross-validation was set up using trainControl to evaluate the performance of each mtry setting. The Random Forest model was then trained using the train() function with 500 trees, the defined tuning grid, and feature importance tracking enabled. The final output (rf_tuned) provided the best-performing mtry value, cross-validated RMSE scores, and a ranked list of variable importance.

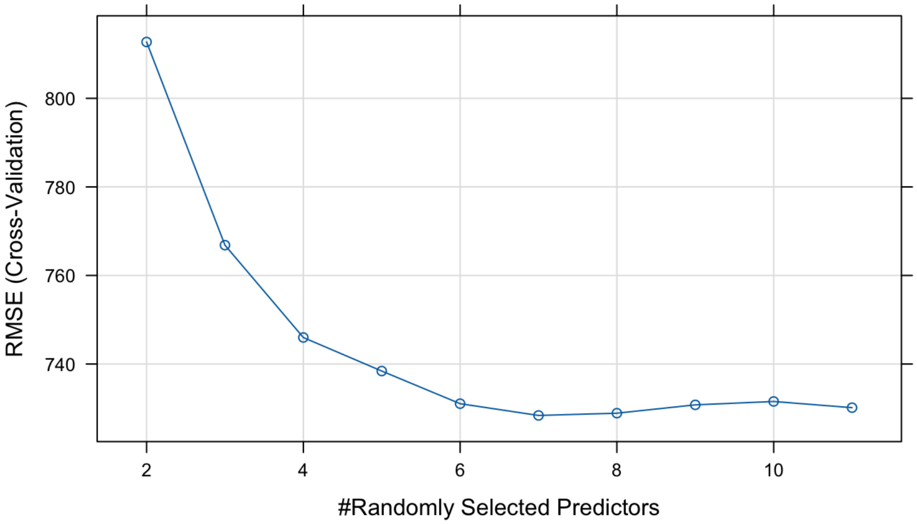

Figure 1: RMSE vs. mtry in Random Forest

The plot shows the cross-validated RMSE for different mtry values (number of predictors randomly selected at each split). RMSE decreases as mtry increases, stabilizing around mtry = 7, suggesting it as the optimal choice for minimizing prediction error.

Support Vector Machine Summary

A Support Vector Machine (SVM) classifier with a radial basis function (RBF) kernel was used to categorize apartments into high- and low-price groups. This algorithm performs effectively in high-dimensional feature spaces and is especially good at managing nonlinear decision boundaries such as the one we are dealing with.

Data Splitting

To ensure class balance, stratified sampling was used to randomly divide the dataset into 80% training and 20% testing groups. This guarantees that there are enough examples for the model to learn from and that it gets tested on unseen data.

Model Implementation

Two models were built and evaluated:

- Baseline SVM (using only numeric features: bedrooms, bathrooms, square_feet)
- Extended SVM (with additional location-based features: cityname, state, latitude, longitude using one-hot encoding)
- Kernel: Radial Basis Function (RBF)
- Cost (C): 1 (controls margin-width vs classification error)
- Gamma (γ)**:** Set to 1 / number of numeric predictors, following common practice for SVMs

All numeric predictors were standardized using **z-score normalization (centering and scaling)** to ensure equal influence.

Model Evaluation

Performance was assessed using:

- Accuracy
- Sensitivity (Recall) for Low-price class
- Specificity (Recall) for High-price class
- Balanced Accuracy
- Kappa Statistic
- Area Under the ROC Curve (AUC)

These results indicate a substantial improvement in predictive power after including location-based features, especially for detecting high-price apartments (specificity).

Linear Regression

Initial Thought

Our initial thought before applying linear regression was to also do regularization to improve the model. However, later we decided not to run any sort of regularization techniques due to the limited number of predictors and irrelevancy. Anyways, first we did basic exploratory data analysis (EDA) to see which kind of relationships we have. Then we fit the model with different combinations of predictors to find a good model. All of the steps are laid out below:

EDA

In this step, we plotted scatterplots to see which relationships emerge. The response is price as mentioned earlier, and there was only one primary numerical predictor of square footage. Other predictors included bedrooms, bathrooms, and others. We observed a positive linear relationship between square feet and our response price.

Fitting the Linear Model

After the EDA, we began fitting several linear models with different predictors to see which ones are most relevant by checking associated p-values. Our first main model included bedrooms, bathrooms, and square feet predictors. Next, we'll discuss cross validation.

Cross Validation

We ran 10-fold cross validation on this model using cv.glm() function. The resulting MSE and RMSE were 566669 and 752, respectively. We thought that this model can be improved, so we added factor(state) to the model and ran CV again. Now the RMSE dropped to 551 which indicates improvement. Furthermore, we decided to tweak the dataset to extract more meaningful predictors, and those steps will be mentioned below.

Further Improvements

We also developed three other variations of a linear regression model:

Model 1: Included all available predictors. After fitting the model, we examined the p-values and identified several insignificant variables that contributed little to the model's predictive power.

Model 2: We removed the insignificant predictors identified in Model 1 and retrained the model. The performance metrics — RMSE and R squared — remained consistent with the original model, indicating that the removed variables did not add value to the prediction.

Model 3: In this version, we scaled numeric variables such as price, cityname_te, state_te, and square_feet using min-max normalization. This ensured that features with larger scales did not disproportionately influence the model. The model maintained performance while being more robust to feature scale. There is not much difference in the R squared and RMSE values between model 2 and model 3.

Smoothing Splines

To model apartment rental prices with greater flexibility and to capture potential nonlinear relationships, we explored the use of smoothing splines. Our selection of predictors was initially guided by a Pearson correlation analysis among the numerical variables in the dataset. The analysis revealed that rental price was most strongly correlated with square footage, followed by the number of bedrooms and bathrooms. Based on these findings, we began by using square footage as the primary predictor for our initial spline models.

While simple linear regression models using square footage alone produced reasonable results, further examination of the residuals showed evidence of systematic patterns and heteroscedasticity. These diagnostic results suggested a violation of the linearity assumption, motivating the adoption of more flexible modeling approaches.

Smoothing splines are well-suited for such scenarios, as they enable the fitted function to flexibly adjust across the range of the predictor variable. Unlike polynomial regression, which can produce unrealistic behavior at the extremes, smoothing splines divide the predictor space into segments and fit smoothly connected piecewise polynomials. This flexibility is especially valuable for modeling complex or nonlinear trends in rental prices with respect to apartment size.

As an initial approach, we fitted a natural spline model using knot placements at the quartiles of the square footage distribution. Given the sensitivity of spline-based methods to outliers and influential observations, we filtered the dataset to include only listings with square footage under 6,000 and rental price under 10,000, reducing the influence of extreme values and improving the stability of the fit.

To determine the optimal level of smoothness for the spline, we employed leave-one-out cross-validation (LOOCV), which evaluates predictive performance by iteratively excluding individual observations. This process identified an optimal degrees of freedom value of approximately 74, offering a balance between model flexibility and generalization error. Using this setting, we evaluated model performance using root mean square error (RMSE), coefficient of determination (R-squared), and 10-fold cross-validation error.

Building upon the single-predictor spline model, we next extended the framework using a generalized additive model (GAM), which allows for the inclusion of multiple predictors, each represented as a smooth function. In addition to square footage, we incorporated the number of bedrooms, number of bathrooms, and geographic coordinates (longitude and latitude) as additional smooth terms. This multivariate model allowed us to capture both structural and spatial influences on rental price.

The generalized additive model provided substantial improvements over the baseline smoothing spline, with a noticeable reduction in RMSE and an increase in explained variance by approximately 24%. This validated the importance of incorporating additional predictors and accounting for nonlinearities in multiple variables.

Finally, we developed a comprehensive model that integrated all significant predictors identified during earlier linear modeling, including both numerical and location-based variables. This final model achieved the highest explanatory power, with an R-squared value of 0.78 and a 10-fold cross-validated RMSE of 561.78, making it the most accurate and robust model in our analysis.

Results

Random Forest

Model	RMSE	R-squared
Random Forest	623.42	0.60

Table 1: Random Forest model’s RMSE and R-squared values

The Random Forest model achieved an RMSE of 623.42, indicating the average prediction error in price. The R-squared value of 0.60 suggests that the model explains 60% of the variance in the rental price, reflecting moderate predictive performance.

Linear Regression

The significance of all the predictors and the multicollinearity is shown in the plot below for the most significant predictors in Linear Regression Model 2.

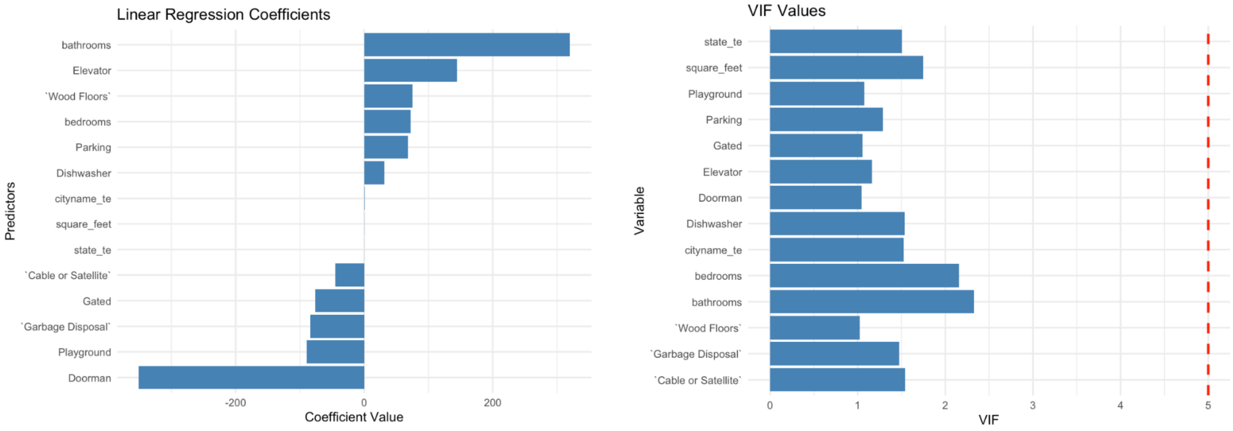

Figure 2: Significant predictors and their VIF values for Linear Regression Model 2

Linear Regression Model	RMSE	R-squared
a. Model 1 (with all predictors)	540.01	0.75
b. Model 2 (with only the significant predictor)	540.83	0.75
c. Model 3 (with scaled significant predictors)	0.01*	0.75

Table-2: Comparison of the three Linear Regression Models on RMSE & R-squared metrics

Smoothing Splines

Table Summary of Smoothing Splines:

Splines**	**RMSE**	**R-squared**	**K-fold CV (k = 10)**
d. Smooth Splines (with square_feet only)	731.34	0.26	748.12
e. Smooth Splines (with square_feet, bedrooms, bathrooms,longitude and latitude)	601.88	0.50	605.41
f. Smooth Splines (with significant predictors)	453.39	0.78	561.78

Table-3: Comparison among the three spline models on RMSE, R-squared & k-fold CV

**Note: As all numerical variables, including the response variable, were scaled using min-max normalization, the observed RMSE of 0.01 corresponds to an approximately $723 in the original price scale.*

***Note: All spline models were fitted on a subset of the data after filtering out outliers and high leverage points. Therefore, comparisons involving these models should take this difference in data scope into account.*

Support Vector Machines (SVM)

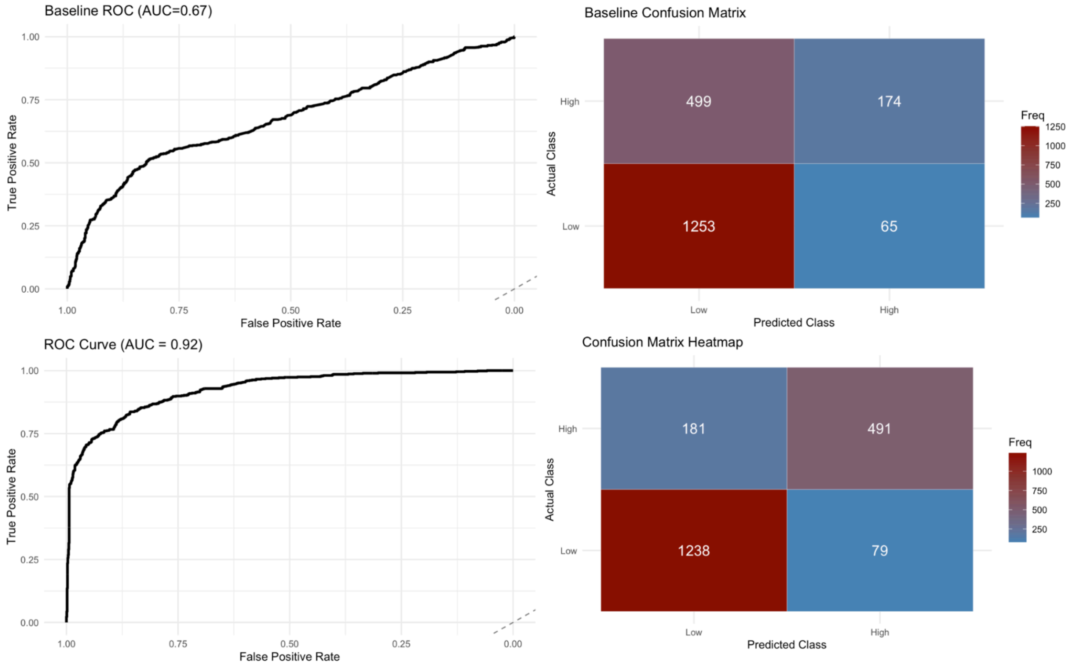

Figure 3: SVM Performance With and Without Location Feature

- **Baseline Model** (Bedrooms, Bathrooms, SqFt)
  - *Top‑left:* ROC curve (AUC = 0.67) for the SVM trained using only the three numeric predictors as number of bedrooms, number of bathrooms, and square footage.
  - *Top‑right:* Corresponding confusion matrix showing true vs. predicted classes.

- **Enhanced Model** (+ Location)
  - *Bottom‑left:* ROC curve (AUC = 0.92) for the SVM trained on the original three features plus location.
  - *Bottom‑right:* Corresponding confusion matrix illustrating the improved classification performance once location is included.

What Improved for SVM from the Baseline to the Enhanced Model:

Metrics	**Baseline (no location)**	**Enhanced (with location)**
Accuracy	0.72	0.87
Sensitivity	0.95	0.94
Specificity	0.26	0.73
Balanced Accuracy	0.60	0.83
Kappa	0.24	0.70
AUC	0.67	0.92

Table-4 Performance metrics comparison between Baseline and Enhanced SVMs

Discussion

Meaning of our results

In completing our analysis on the provided dataset, we performed several statistical learning methods such as linear regression, smoothing splines, random forests and support vector machines. We'll first discuss the regression methods and at the end we'll discuss classification ones.

Our linear regression model went through multiple tweaks as we tried to optimize it as much as we could. It did come out to be one of the best performing models. The lattermost improved model with added predictor of amenities performed the best among earlier linear models with RMSE of ~ 540. The most important predictors were bathrooms and elevator which is interesting because earlier models had square footage or other predictors at higher rank. This could be because we didn't expand on the amenities before and things like elevators came out

to be important. The VIF values in that model are also pretty low with all of them under 2.5, so there's no risk of multicollinearity.
Linear regression has an advantage over other models in the sense that it is most interpretable. We can easily use coefficients to tell how much price would change if we increased a particular predictor by one unit. On the other hand, in most cases there is a downside to this method and that is less accurate predictions. However, in our situation it performed strongly may be because the data supported it well.

Next model we used was random forests. If we go straight to its performance, then it's RMSE value was 623.42 which is higher than the linear regression. In general, random forests should perform better with less interpretability but here it performed slightly worse.

Another model we worked with was smoothing splines, which uses multiple functions to deal with non linear data. As the results show, smoothing splines with many predictors resulted in an RMSE value of 453 which is the *lowest* for our project! In general, the strength of these splines is that they can deal with non linearity in data really well, but downside could be overfitting if not careful and less interpretability than simpler models like linear regression.

Now, the only classification method used was SVMs. We used this to get a new perspective or gain new insights from our dataset. Because this is different from regression, we can't directly compare it with regression methods discussed earlier. So for SVMs, we fit two models. Earlier model was without the location predictor because we didn't know that adding it would improve our linear model significantly so we decided to incorporate that into SVM as well. Now for ROC curves we want the curve to touch the top left corner because that is going towards the optimum point. We want to maximize the true positive rate. If we look closely at the two plots in the SVM results section, we can easily see that adding location significantly increases the true positive rate of our model. This is confirmed in our metrics reporting. First of all, accuracy of the model improved alongside great improvement in true negative rate as well.

SVMs excel at classifying the data really well but may lack in dealing with large datasets as the training time seems to increase (needs computation power).

Usage and Future Implications

The results obtained from this work can be used by future researchers who are working on a similar topic. We've shown that predictors like amenities, square footage, bathrooms and others play a key role in predicting the price of a property (apartments in this case) and these models can be extended by adding more relevant predictors such as specific regions, sentiment based, etc.

Limitations

Our work may have some limitations that we discuss now. Linear regression like some other methods come with some assumptions. We didn't rigorously test those assumptions so interpretations of results may not be fully concrete. This could come under improving the project in the future by either us or other researchers. Another limitation could be relying just on CV even though it is a strong method but it still uses test data from original data, so for full clarity a brand new test data can be used which wasn't available to us.

References

1) Glaeser, E. L., & Gyourko, J. (2005). Urban decline and durable housing. *Journal of Political Economy*, 113(2), 345–375. https://doi.org/10.1086/427231
2) Kok, N., Monkkonen, P., & Quigley, J. M. (2017). The capitalization of environmental amenities: Green building certification and the rental premium. *Real Estate Economics*, 45(1), 1–29. https://doi.org/10.1111/1540-6229.12147
3) Dubé, J.-P., Legros, D., & Thériault, M. (2014). The spatial structure of housing markets: Evidence from the Montreal metropolitan area. *Urban Studies*, 51(11), 2283–2302. https://doi.org/10.1177/0042098013513834
4) Dataset link: UC Irvine Machine Learning Repository. (2018). *Apartments for Rent in US*. Retrieved from https://archive.ics.uci.edu/ml/datasets/Apartments+for+Rent+in+US