Name: Saher Saeed
Student ID: 23095056
Github Link: https://github.com/saeedsahar/svn-random-decision-repo

Classification Model Comparison Report

Introduction

This report evaluates the performance of Support Vector Machine (SVM), Decision Tree, and Random Forest classifiers using a dataset of rental listings in the United States. The objective is to classify whether a listing's rent price is above the median.

Dataset Overview

This dataset consists of **10,000 rental listings** across the United States, with **22 features**. It includes numeric attributes like **price**, **square footage**, and **geolocation**, along with categorical fields such as **state** and **property type**.

Key Highlights:

- 8 numeric features
- 14 categorical features
- 8,839 missing values (handled via imputation)

Variable Name	Description
price	Monthly rent price in dollars
bedrooms	Number of bedrooms
bathrooms	Number of bathrooms
square_feet	Total area in square feet
address	Street address of the rental listing
city	US state abbreviation
ip_code	ZIP code of the rental
longitude	Latitude coordinate
posting_date	Date when the listing was posted
url	Derived size category based on square footage (Small/Medium/Large)

Data Distribution:

Small (n=2171), Medium (n=679), Large (n=82) showing a right-skewed distribution toward smaller properties.

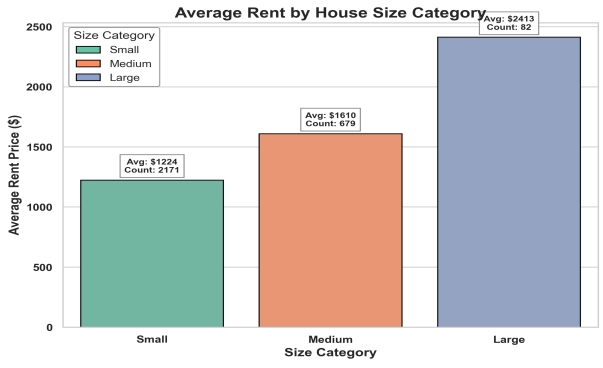


Figure-1 : Average Rent by House Size Category

Descriptive Statistics of Selected Features

Most rental listings have around 1–2 bedrooms and bathrooms. Price and square footage show extreme right-skew, indicating a few high-value outliers.

Index	Count	Mean	Std	Min	Skewness	Kurtosis
bathrooms	2109	1.233	1	4	1.895	3.332
bedrooms	2109	1.667	0	6	1.055	0.324
price ($)	2109	1138.21	222	2450	0.597	0.048
square_feet	2109	868.849	225	3740	1.899	4.524
latitude	2109	36.7872	5.166	46.881	−0.210	−1.261
longitude	2109	−90.284	9.021	46.386	−0.324	−1.922
time	1.69314e	1.63314.05	1.578	1.5736e	1.57558e+09	2.951
time	2109	1.57728e	e+09	1.57736	1.57736e+09	2.951

Preprocessing Steps

Key steps included handling missing data, encoding categories, removing outliers, scaling features, and splitting the dataset—ensuring clean, model-ready input.

Step	Description
Load Dataset	Fetched the UCI dataset using `fetch_ucirep(id=55)`.
Merge Data	Replace empty strings to appropriate dataFrames.
Replace Empty Strings	Replaced empty strings with NAN values (`dl.nrp,naan`).
Drop Missing Values	Removed rows with missing values using `df.dropna = True`.
Missing Values	Imputed missing values with mean
Convert Data Types	Handling conversions like numeric dietection via select_dtypes() during preprocessing
Outlier Removal	Applied IQR method to remove outliers from numerical columns.
Save Cleaned Data Copy	Store cleaned Data Copy ((for SVM).
Categorical Encoding	State encoding using LabelEncoder.
Feature Selection	Selected 5 relevant numerical and categorial features
Train-Test Split	Splaited StandardScaler (for

Model Evaluations

Support Vector Machine (SVM)

The Support Vector Machine (SVM) classifier shows **balanced training and validation accuracy (65%)**, indicating consistent generalisation. However, it performs better on **low-priced properties**, with high recall but struggles to correctly identify high-priced ones.

Metric	Class 0 (≤ Median)	Class 1 (> Median)	Overall
Precision	0.6	0.81	0.65
Recall	0.91	0.39	0.65
F1-Score	0.73	0.53	0.65
Accuracy			0.65
Support (n)	213	209	422.0

SVM – Best Parameters: C=10, Kernel=RBF
After tuning, the SVM model achieved **74.6% accuracy**, with balanced performance across both classes:

Metric	Class 0 (≤ Median)	Class 1 (> Median)	Overall
Precision	0.72	0.78	0.75
Recall	0.82	0.67	0.75
F1-Score	0.76	0.72	0.74
Support (n)	213.0	209.0	422.0

SVM – Confusion Matrix Summary

- **Correct:** 174 (low) & 141 (high)
- **Incorrect:** 39 (high → low) & 68 (low → high)

The tuned SVM shows balanced performance, slightly better at identifying **low-priced** homes.

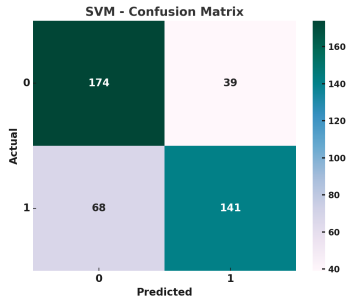


Figure-2 Confusion Matrix for SVM

Random Forest Performance on Dataset

The Random Forest model achieved **99% training accuracy** and **77% validation accuracy**, showing strong generalisation. It performed consistently across both classes, with precision, recall, and F1-scores all around **0.77–0.78**, indicating balanced and reliable classification of rental categories.

Metric	Class 0 (≤ Median)	Class 1 (> Median)	Overall
Precision	0.78	0.77	0.77
Recall	0.77	0.78	0.77
F1-Score	0.78	0.77	0.77
Accuracy			0.77
Support (n)	213	209	422.0

Random Forest – Best Parameters: max_depth=10, n_estimators=100

Tuned model achieved **77% accuracy** with strong, balanced performance. Deeper trees and more estimators enhanced generalisation and reduced overfitting.

Random Forest – Confusion Matrix Summary

- **Correct:** 171 (low) & 159 (high)
- **Incorrect:** 42 (high → low) & 50 (low → high)

The tuned Random Forest model shows **balanced classification** with fewer misclassifications, especially for high-priced homes.

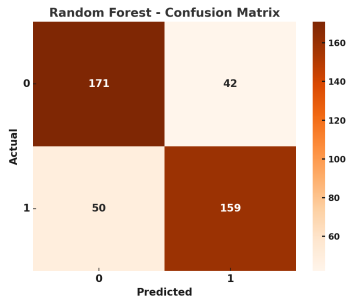


Figure-3 Confusion Matrix for Random Forest

Decision Tree – Accuracy: 73%

The model shows balanced performance across both classes with precision and recall around 0.73. Slight overfitting is evident (99% train accuracy), but it generalises reasonably well on validation data.

Decision Tree – Best Parameters: max_depth=10, min_samples_split=10

Metric	Class 0 (≤ Median)	Class 1 (> Median)	Overall
Precision	0.73	0.73	0.73
Recall	0.74	0.72	0.73
F1-Score	0.73	0.72	0.73
Accuracy			0.73
Support (n)	213	209	422.0

Tuned model reached **73% accuracy** with balanced performance. Depth and split control helped reduce overfitting.

Decision Tree – Confusion Matrix Summary

- **Correct:** 160 (low) & 165 (high)
- **Incorrect:** 53 (high → low) & 44 (low → high)

Slight edge in detecting high-priced homes; overall stable performance.

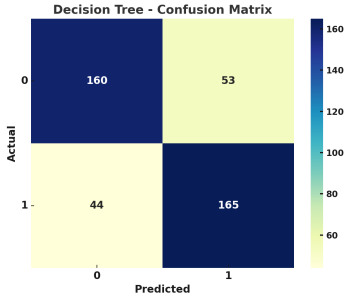
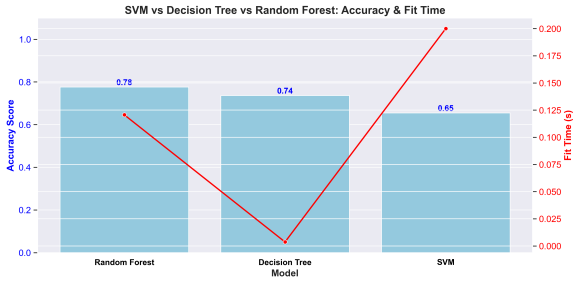


Figure-4 Confusion Matrix for Decision

Accuracy vs. Fit Time

The chart below compares the three models in terms of accuracy and training time. Random Forest performs best overall, while Decision Tree trains the fastest.



Final Model Comparison – Conclusion

- **SVM** (Accuracy: 74.6%) – Tuned with C=10, kernel='rbf'; improved but slightly favours low-priced homes.
- **Random Forest** (Accuracy: 77%) – Best performer with max_depth=10, n_estimators=100; balanced and reliable.
- **Decision Tree** (Accuracy: 73%) – Solid with max_depth=10, min_samples_split=10; slightly overfits.

Winner: Random Forest – Best balance of accuracy and generalisation.

References

1. **SVM Theory and Application**
    - Cortes, C., & Vapnik, V. (1995). *Support-vector networks*. Machine Learning, 20(3), 273–297.
    - https://doi.org/10.1007/BF00994018
2. **Random Forest Introduction**
    - Breiman, L. (2001). *Random forests*. Machine Learning, 45(1), 5–32.
    - https://doi.org/10.1023/A:1010933404324
3. **Decision Tree Algorithms**
    - Quinlan, J. R. (1986). *Induction of decision trees*. Machine Learning, 1(1), 81–106.
    - https://doi.org/10.1007/BF00116251
4. **Model Evaluation Metrics**
    - Saito, T., & Rehmsmeier, M. (2015). *The precision-recall plot is more informative than the ROC plot when evaluating binary classifiers on imbalanced datasets*. PLOS ONE, 10(3).
    - https://doi.org/10.1371/journal.pone.0118432
5. **Scikit-learn Documentation** (for implementation and tuning references)
    - https://scikit-learn.org/stable/