Izmir Houses Rent Prices Prediction Model

Amar Kaid Yousef ATOUM
Department of Computer Engineering
Dokuz Eylul University
Izmir, Türkiye
ammarkaid321@gmail.com

***Abstract*— This paper explores a practical approach to predicting rental prices in İzmir's housing market and how it can benefit real estate professionals and renters alike. By using data mining techniques, the study identifies key factors—like neighborhood characteristics and socioeconomic indicators—that influence rental costs. The model is designed to make accurate predictions even when detailed historical pricing data is unavailable. The findings demonstrate how combining various data sources can create a clearer picture of rental trends, helping stakeholders make smarter, data-driven decisions.**

Keywords—rental price prediction, data mining, İzmir housing market, socioeconomic factors, real estate insights

I. INTRODUCTION

Predicting rental prices is an essential application of data mining techniques, particularly in dynamic markets like İzmir's housing sector. The goal is to understand and anticipate how factors such as neighborhood amenities, socioeconomic indicators, and housing features influence rental costs. This insight is invaluable for renters, landlords, and real estate professionals when making decisions related to pricing, investments, or relocations. This study focuses on using data mining methods to analyze key attributes like location, property size, and demographic trends to predict rental prices, even when comprehensive historical data is unavailable. For example, if a property shares similar characteristics with another in a specific neighborhood, this predictive model can estimate its potential rental value. This approach not only helps stakeholders optimize their decisions but also provides a clearer understanding of market trends in İzmir, enhancing transparency and efficiency in the real estate industry.

II. LITERATURE STUDIES

The literature on rental price prediction models offers insights into how various data mining and machine learning techniques have been applied to real estate markets. This section reviews key studies to establish the groundwork for the development of an İzmir housing rental prediction model. It highlights advancements in predictive techniques, identifies critical variables, and addresses challenges such as data availability and preprocessing.

Raju Raju, Arham Neyaz and Aleem Ahmed (2020) demonstrated the effectiveness of random forest algorithms in predicting rental prices, noting their ability to handle large datasets with diverse attributes [1]. Yoshida, T., Murakami, D., & Seya, H. (2024) compared regression models with neural networks, concluding that machine learning methods provide superior accuracy, especially when spatial factors are included [2].

One of the common challenges in predictive modeling, as noted by Wang, Z., Akande, O., Poulos, J., & Li, F. (2022)., is handling missing or incomplete data. Techniques like imputation and synthetic data generation have been proposed to address this issue, but their efficacy varies across datasets [3].

While previous studies have focused on global or national markets, there is limited research specific to İzmir's housing market. This gap underscores the need for localized models that account for regional socioeconomic factors and unique market dynamics.

This review demonstrates the potential of machine learning in predicting rental prices and identifies key variables influencing costs. However, it also highlights the lack of localized studies focusing on İzmir. This research aims to fill this gap by leveraging regional data and advanced predictive techniques to enhance model accuracy.

III. DATASET

The dataset used in this research was collected through web scraping from a real estate website called **Hepsiemlak**, a popular online platform in Turkey . Hepsiemlak specializes in listing properties for sale and rent across the country, providing detailed information about each listing to assist users in making informed decisions. This dataset includes 13 columns and over 5000 rows of UpToDate listings, capturing various aspects of the rental market in İzmir. By analyzing these attributes, the research aims to predict rental prices based on property features and neighborhood characteristics.

A. Preset Features

Numerical Features:

- **Price:** The rental cost of the property in Turkish Lira.
- **Area (m²):** The size of the property in square meters.
- **Deposit:** The amount of security deposit required.

Binary Features:

- **Furniture Status:** Indicates whether the property is furnished or unfurnished.

Nominal Features:

- **Location:** Detailed address information, including city, district, and neighborhood.
- **Heating Type:** Type of energy source for heating, such as natural gas or electricity.
- **Fuel Type:** Type of energy source for stoves, such as natural gas or electricity.

Ordinal Features:

- **Floor:** The floor level of the property
- **Age of Property:** Age of the house.

B. *Preset Features*

Two new features were derived from the existing dataset to provide deeper insights and enable advanced analysis:

Price per Square Meter (PRICE_PER_M2): To assess the cost-effectiveness of a rental property, the **PRICE_PER_M2** feature was created by dividing the rental price by the property area (in square meters). This allows for comparisons between properties of varying sizes.

Age Group (AGE_GROUP): Using the age of the property, properties were categorized into four groups:

- **New (0–5 years):** Recently built properties.
- **Moderate Age (5–15 years):** Properties in good condition.
- **Old (15–30 years):** Older properties.
- **Very Old (30+ years):** Properties that may require renovation or maintenance.

IV. DATA PREPROCESSING

Data preprocessing, also known as data preparation or data cleaning, involves identifying and correcting faulty or inconsistent data records within a dataset. In our project, data preprocessing was carried out using Python and its robust libraries, including Pandas, Numpy, and Matplotlib. These tools facilitated efficient handling, visualization, and cleaning of the dataset.

Upon initial observation, the dataset contained numerous columns irrelevant to the problem at hand, such as detailed property descriptions and redundant categorical features. These columns were pruned, leaving only essential features like price, deposit, monthly_fee, furniture, floor, location, and area_m2, among others. This pruning process significantly reduced dimensionality and improved data interpretability, as described in section III.

The dataset exhibited missing values in several columns. A targeted approach was implemented to handle these:

- **Numerical Columns:** For ordinal numeric features such as age and area_m2, missing values were filled with the mean of the respective column.
- **Categorical Columns:** Missing values in nominal features like furniture and heating_type were filled with the mode. For instance, missing furniture values were replaced with "Eşyalı değil".
- **Dependency-Based Columns:** In cases where the deposit column was missing or zero, it was replaced with the value from the price column

Extreme outliers were identified in columns such as price and monthly_fee. These were removed based on thresholds derived from the interquartile range (IQR) method, ensuring a more robust dataset for analysis and modeling.

Several new features were derived to enhance the dataset:

1. **PRICE_PER_M2:** Calculated by dividing price by area_m2, this feature highlights the cost-effectiveness of a property.
2. **AGE_GROUP:** Derived using binning, AGE_GROUP categorized properties as "New", "Moderate Age", "Old", or "Very Old" based on their age.
3. **Location Splitting:** The location column was split into City, District, and Neighborhood for better geographical analysis.

Categorical data was further grouped and standardized for better interpretability:

- **Furniture Grouping:** Simplified into "Eşyalı" (Furnished) and "Eşyalı değil" (Not Furnished).
- **Floor Categorization:** Floors such as "Zemin Kat" (Ground Floor) and "Çatı Katı" (Top Floor) were grouped into broader categories like "Ground Floor" and "Top Floor". Numeric floors were retained for clarity.
- **Heating Type:** Features like heating_type were standardized to "Kombi" (Central) or "None" for missing entries.

To ensure compatibility with machine learning models, categorical columns were encoded into numerical values:

- **Label Encoding:** Applied to binary columns like furniture (Furnished vs. Not Furnished), encoding them as 1 and 0.
- **One-Hot Encoding:** For features with multiple categories, such as heating_type and location, one-hot encoding was applied, creating separate columns for each category.

By removing redundant columns, handling missing data, engineering new features, and encoding categorical data, the dataset was transformed into a structured, analysis-ready format. These preprocessing steps not only improved data quality but also enhanced compatibility with machine learning models, paving the way for more accurate predictions and insightful analyses.

V. ANALYSIS OF DATA

To make the analysis of the data .csv file is read with pandas library in Python into a dataframe. With df.head(5) and df.describe(include='object') quick summary of the can be accessed. For further analysis matplotlib.pyplot is used to generate plots such as box plot, bar chart or pie chart.

The first histogram represents the distribution of rental prices in TL. The graph shows a significant concentration of listings below the 100,000 TL mark, with a sharp drop in frequency as the price increases. The presence of outliers is evident, with a few properties listed above 600,000 TL.

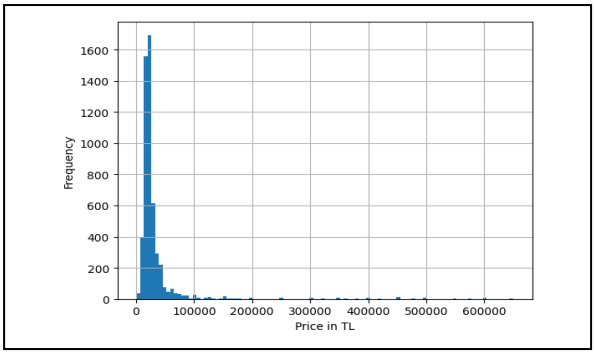

Fig. 1. Distribution of Rental Prices

Most properties fall into the affordable price range (<100,000 TL), making them accessible for middle-income renters. Outliers (extremely high-priced properties) might represent luxurious or exclusive listings, which could distort average price metrics if not handled properly.

The second boxplot reinforces the observation of outliers in rental prices. The majority of data points cluster near the lower end of the range, while the whiskers extend far, indicating substantial variation in the upper range.

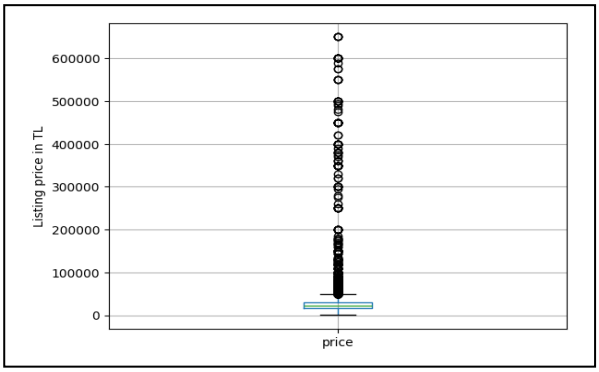

Fig. 2. Boxplot of Rental Prices

The median price lies well below the maximum value, indicating a skewed distribution. Cleaning or capping these outliers might improve the robustness of the data for predictive modeling.

The third boxplot in the ipynb file was very big to be imported to the report. However, being grouped by districts reveals distinct patterns in rental prices across different districts. Districts such as Çeşme have significantly higher median prices, while others like Aliağa have lower median prices. Districts like Çeşme likely represent premium or tourist-friendly areas, justifying the higher rental prices. Grouping properties by district allows for region-specific pricing strategies and better understanding of localized trends.

The fourth boxplot categorizes prices by the number of rooms in each property. Listings with more rooms generally show higher rental prices, although there are some anomalies (e.g., studios with high prices).

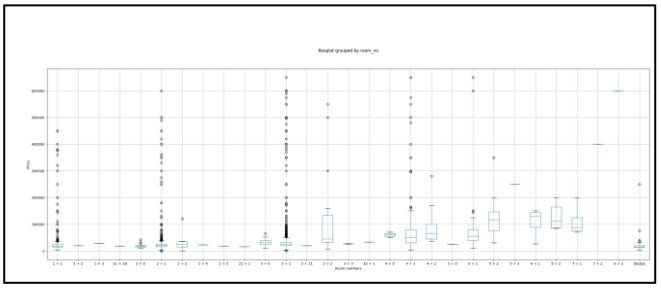

Fig. 3. Rental Prices by Room Numbers

Larger properties (e.g., 4+1, 5+1) command premium prices, aligning with market expectations.

The variation within categories suggests additional factors, such as location or amenities, influence pricing.

The fifth boxplot compares rental prices between furnished (Eşyalı) and unfurnished (Eşyalı değil) properties. While both categories display similar median prices, furnished properties exhibit slightly higher variability.

Furnished properties might appeal to a niche market, such as expatriates or short-term renters, justifying the higher variability. Understanding the impact of furniture status can aid in targeted marketing strategies.

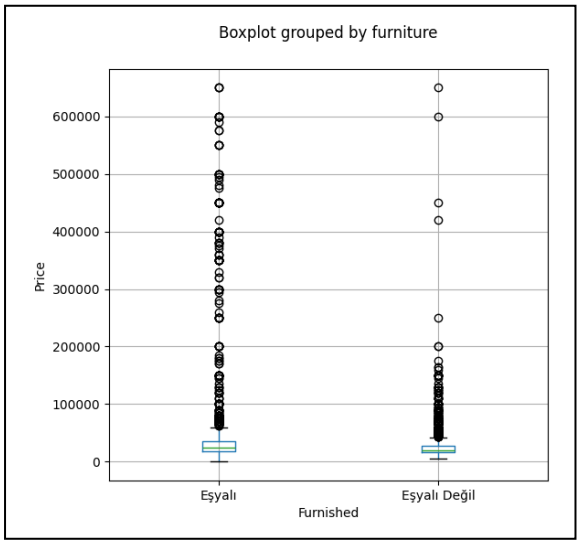

Fig. 4. Furniture Status vs. Rental Prices

The bar chart on the left shows that unfurnished (Eşyalı değil) properties significantly outnumber furnished ones. However, the bar chart on the right reveals that furnished properties are more likely to be in the higher price range.

Furnished properties are less common but have a higher chance of being premium listings. Including furniture status as a predictive feature in modeling could enhance price predictions.

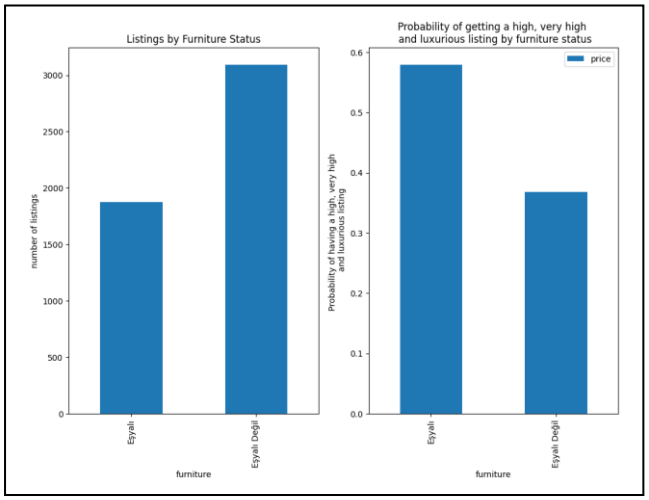

Fig. 5. Listings by Furniture Status

The pie chart below illustrates the proportion of different data types in the dataset:

- **60% categorical features** dominate, indicating a need for careful encoding before model training.
- **33% float64 and 6.7% int64** reflect the importance of numerical features like price, area, and deposit.

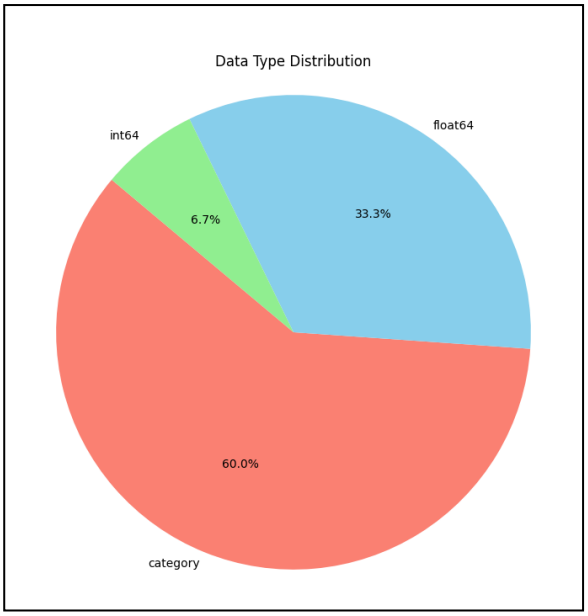

Fig. 6. Data Type Distribution

Proper preprocessing, including label and one-hot encoding, is essential for categorical features. Scaling of numerical features will ensure balanced input for machine learning models.

The final bar chart demonstrates feature importance based on importance_gain. The PRICE_PER_M2 feature has the highest importance, followed by deposit and area m2.

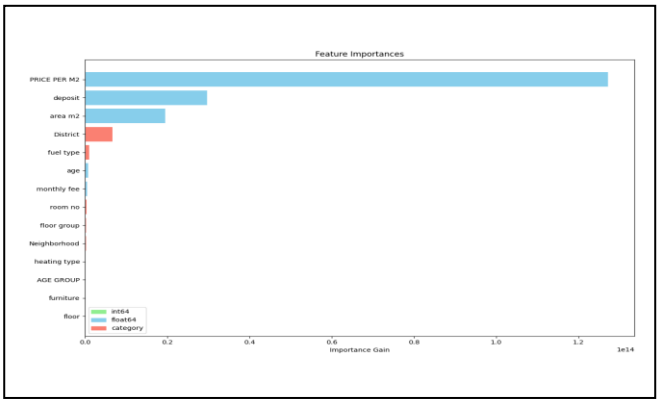

Fig. 7. Feature Importance

PRICE_PER_M2 is the most critical feature, highlighting the influence of property size on rental pricing. Features like District, fuel type, and monthly fee also contribute meaningfully but to a lesser extent. Features with negligible importance (furniture, floor) might require further exploration or could be excluded from certain models.

VI. APPLIED MACHINE LEARNING METHODS

In the context of **Izmir houses rent price prediction**, various **regression** methods have been employed to model and estimate rental costs based on diverse features collected from web-scraped real estate listings. Since the target feature in this study is **continuous** rather than **binary** or **categorical**, regression algorithms were deemed more appropriate than classification or clustering techniques. Consequently, **Linear Regression, Decision Tree, Random Forest, Gradient Boosting, Support Vector Regression (SVR),** and **Neural Networks** were implemented.

During the development of these models, Python's **scikit-learn (sklearn)** library was primarily utilized. Sklearn is a widely used Python package in the fields of data science and machine learning; it provides a multitude of tools for tasks such as data preprocessing, dimensionality reduction, model selection, regression, classification, clustering, and offers numerous built-in datasets for model testing and validation.

Additionally, **LightGBM (LGB)** was integrated to enhance regression performance and to derive comprehensive **feature importance** insights. LightGBM's efficiency with large datasets and its ability to produce high-accuracy predictions made it a valuable component in improving the ensemble models. Moreover, the feature importance functionality within LightGBM greatly assisted in identifying key attributes—such as the number of rooms, district location, and floor level—that exert the most significant influence on rental price estimation.

In this study, **Seaborn** played a pivotal role in the visualization process. Seaborn's capability to generate detailed heatmaps and comparative plots proved invaluable for illustrating key metrics of each model (such as **Random Forest, Gradient Boosting, SVR**, and **Neural Network**), facilitating both the comprehension and interpretation of each algorithm's effectiveness. By presenting how different features and hyperparameters influence predicted rental values, Seaborn charts contributed substantially to evaluating model performance and conveying findings.

The broader data handling framework involved **NumPy**, **Pandas**, and **Matplotlib** for fundamental tasks in data loading, preprocessing, statistical analysis, and visual exploration. NumPy provided efficient array structures and numerical functions, Pandas was essential for flexible data manipulation, and Matplotlib served to construct a variety of plots, enabling a quick and comprehensive look at data distributions, correlations, and model outcomes.

Two complementary evaluation methods were employed to assess the regression models. First, an **n-fold (n=5) cross-validation** strategy was adopted to gauge the robustness of each algorithm across multiple, non-overlapping data segments. Second, the data was partitioned into a **train-test split (80% training, 20% testing)** for a more conventional, straightforward performance check. Model hyperparameters—including the depth and number of estimators for Decision Trees and Random Forest, learning rate and estimators for Gradient Boosting, and kernel/regularization parameters for SVR—were iteratively tuned under both evaluation schemes.

This combination of techniques and tools ensured a thorough examination of the predictive accuracy and generalizability of the selected regression models, resulting in a cohesive methodology for accurately estimating **Izmir rental housing prices** based on web-scraped real estate data.

A. Linear Regression

This model was implemented via the LinearRegression() class from the scikit-learn library. The built-in fit method was utilized to train the model using the processed training

set, and subsequently, the predict method was invoked for inference on the test set. Key hyperparameters—such as normalize and fit_intercept—were kept at their default values to gauge baseline performance before applying further tuning. By interpreting the learned coefficients, Linear Regression offered a straightforward view of how each feature affected the predicted rental price.

B. Decision Tree Algorithm

A Decision Tree Regressor was established by calling the DecisionTreeRegressor() function from scikit-learn. Its parameters, including max_depth, min_samples_split, and min_samples_leaf, were fine-tuned to enhance the model's fit and minimize overfitting. Training was achieved with the fit method, while predictions on unseen data were facilitated by the predict method. Visualizing feature importances and tree structures proved useful in identifying the input variables most influential to rental pricing.

C. Random Forest Regression

The Random Forest model was created using the RandomForestRegressor() class. Its essential parameters—like the number of estimators (n_estimators) and the maximum features considered at each split (max_features)—were scrutinized to balance the trade-off between accuracy and computational overhead. The ensemble's strength emerged from aggregating predictions of numerous individual trees. Additionally, metrics such as RMSE and MAE were tracked to measure improvements from hyperparameter tuning.

For further refinements and to complement the Random Forest approach, LightGBM was also employed. By leveraging the lgb.train function, parameters such as num_leaves, learning_rate, and num_boost_round could be adjusted. This helped in boosting the predictive capabilities of tree-based ensembles and facilitated a detailed feature importance analysis.

D. Gradient Boosting Regression

Gradient Boosting was executed using the GradientBoostingRegressor(). Parameters, including the learning rate (learning_rate), number of estimators (n_estimators), and subsampling fraction (subsample), were tested and iteratively tuned. Through an additive training process, each new weak learner (decision tree) attempted to correct the errors of the previous iteration, resulting in more refined predictions over multiple boosting rounds. This algorithm often converged to accurate estimations for the rental prices in relatively few iterations.

E. Support Vector Regression (SVR) Regression

Implemented via the SVR() class, this model demanded specific hyperparameter tuning of C, epsilon, and kernel (most commonly 'rbf'). A log transformation (np.log1p) of the target variable (price) helped manage skewness and stabilize variances. After training on the transformed labels, the predictions were subsequently inversely transformed (np.expm1) to return them to the original price scale. Iterating over a range of parameter values yielded improved generalization performance and reduced error on test samples.

F. Neural Network

A Multi-Layer Perceptron Regressor (MLPRegressor()) was introduced for capturing complex, nonlinear patterns. Adjustable aspects of the neural network comprised hidden layer sizes, activation functions (e.g., 'relu'), and optimizers. By scaling features with StandardScaler and increasing the number of training epochs (max_iter), the model was tuned for more stable learning. Despite being sensitive to hyperparameter choices, the MLPRegressor displayed robust performance on the rental pricing data after appropriate scaling and parameter optimization.

VII. RESULTS OF THE EXPERIMENTS

The results of the classification methods were individually analyzed and assessed for each respective method

A. Linear Regression

1) Train-Test Split

Evaluation Metrics:

Metric	Mean Value
RMSE	8505.31
MAE	5248.93
R^2	0.66
GCV	119519.61

Explanation based on results:

RMSE (8505 TL) reveals a moderate difference between predicted and actual values.

MAE (5249 TL) shows the average absolute error in predictions.

R^2 (0.66) indicates that 66% of the variation in rental prices is explained by the model.

GCV (119519.61) suggests potential overfitting or underfitting, implying further model refinement or feature engineering could improve results.

2) 5-Fold Cross Validation

Evaluation Metrics:

Metric	Mean Value
RMSE	8523.96
MAE	5288.04
R^2	0.59
GCV	13971.84

Explanation based on results:

RMSE (8523.96) highlights the typical deviation of predictions from true values across folds.

MAE (5288.04) points to the average absolute error in these predictions.

R^2 (0.59) indicates the model captures around 59% of the variation in rental prices.

GCV (13971.84) suggests some potential overfitting or underfitting, warranting further fine-tuning and feature refinement.

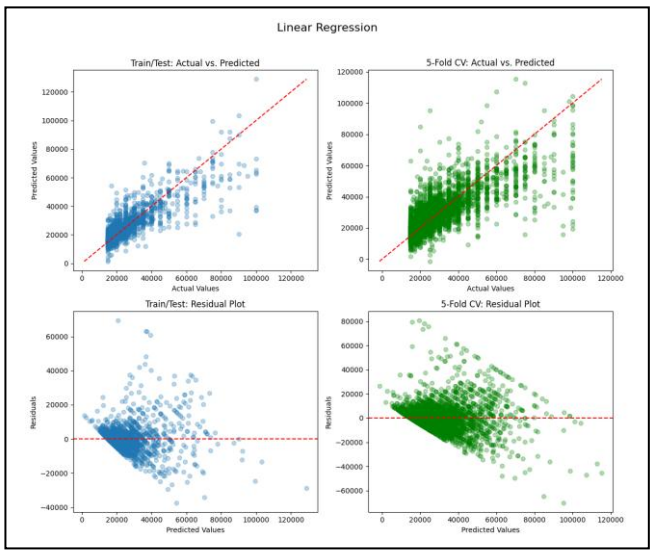

Fig. 8. LR Train/Test & 5-fold Plots

B. Decision Tree Algorithm

1) Train-Test Split

Evaluation Metrics:

Metric	Mean Value
RMSE	8694.91
MAE	4504.90
R^2	0.65
GCV	2588.18

Explanation based on results:

RMSE (8694.91) indicates how far, on average, predictions deviate from actual values.

MAE (4504.90) shows the typical absolute difference between predicted and true values.

R^2 (0.65) suggests the model accounts for 65% of the variability in rental prices.

GCV (2588.18) implies that further tuning (e.g., adjusting tree depth) might enhance predictive consistency

2) 5-Fold Cross Validation

Evaluation Metrics:

Metric	Mean Value
RMSE	9334.50
MAE	4775.34
R^2	0.51
GCV	16755.32

Explanation based on results:

RMSE (9334.50) reflects the average gap between predictions and actual values across folds.

MAE (4775.34) indicates the mean absolute discrepancy in those predictions.

R^2 (0.51) shows the model explains about half of the total variance in rental prices.

GCV (16755.32) signals that tuning parameters (e.g., tree depth or splitting criteria) may enhance accuracy and reduce errors.

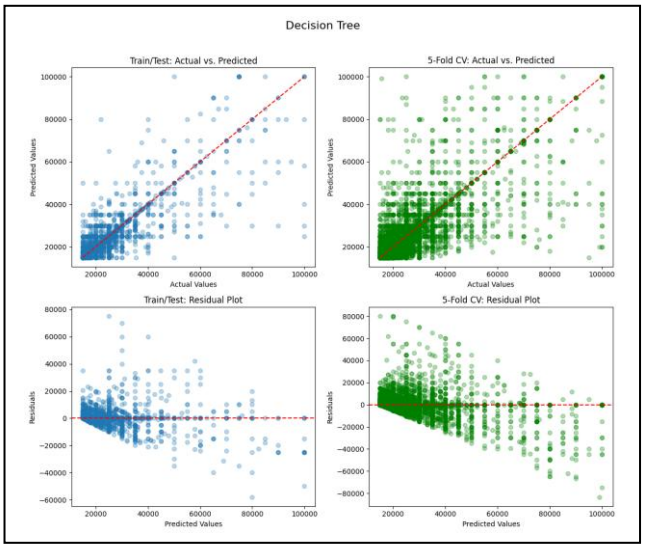

Fig. 9. Decision Tree Train/Test & 5-fold Plots

C. Random Forest

1) Train-Test Split

Evaluation Metrics:

Metric	Mean Value
RMSE	7276.75
MAE	4021.43
R^2	0.75
GCV	37922.67

Explanation based on results:

RMSE (7276.75) shows a moderate average distance between predicted and actual values.

MAE (4021.43) indicates that on average, forecasts deviate by roughly 4021 TL.

R^2 (0.75) reveals that 75% of the variability in rental prices is accounted for by the model.

GCV (37922.67) suggests there is still room for optimization to reduce overall prediction error.

2) 5-Fold Cross Validation

Evaluation Metrics:

Metric	Mean Value
RMSE	7162.07
MAE	4139.59
R^2	0.71
GCV	9863.87

Explanation based on results:

RMSE (7162.07) highlights the average gap between predictions and actual values across folds.

MAE (4139.59) indicates that on average, the model's estimates are off by about 4139 TL.

R^2 (0.71) suggests that 71% of the variance in rental prices is explained by the model.

GCV (9863.87) implies the model generalizes reasonably well, though further adjustments could still fine-tune performance.

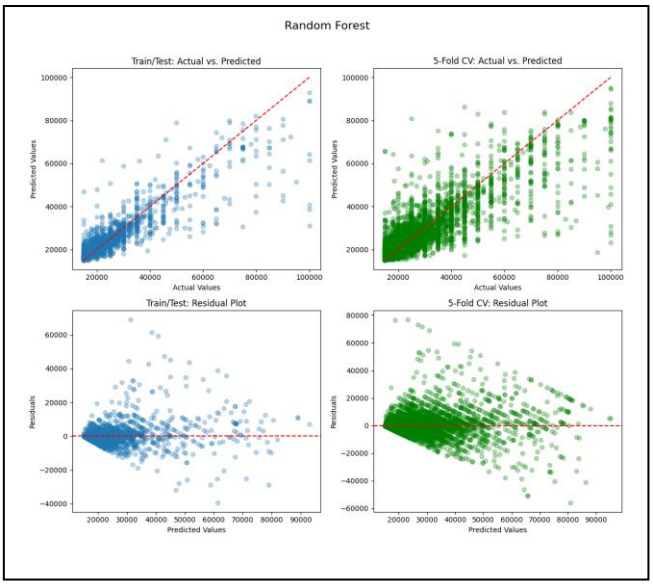

Fig. 10. Random Forest Train/Test & 5-fold Plots

D. Gradient boosting

1) Train-Test Split

Evaluation Metrics:

Metric	Mean Value
RMSE	8528.91
MAE	5457.63
R^2	0.66
GCV	52096.86

Explanation based on results:

RMSE (8528.91) indicates the average deviation of predictions from actual values.

MAE (5457.63) reflects the mean absolute error of the predicted prices.

R^2 (0.66) reveals that 66% of the variation in rental prices is captured by the model.

GCV (52096.86) implies additional hyperparameter or feature engineering may further improve predictive accuracy.

2) 5-Fold Cross Validation

Evaluation Metrics:

Metric	Mean Value
RMSE	8150. 54
MAE	5259.32
R^2	0.62
GCV	12774.51

Explanation based on results:

RMSE (8150.54) indicates the average discrepancy between predicted and actual values across folds.

MAE (5259.32) reveals the typical absolute error in the model's predictions.

R^2 (0.62) signifies that 62% of the variation in rental prices is explained by the model.

GCV (12774.51) suggests that additional parameter tuning or feature engineering may further enhance performance.

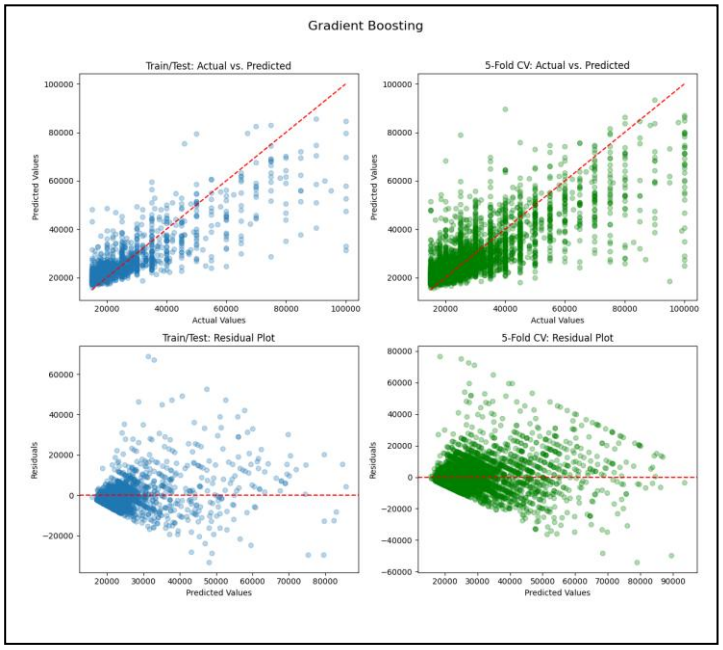

Fig. 11. Gradient Boosting Train/Test & 5-fold Plots

E. Support Vector Regression (SVR) Regression

1) Train-Test Split

Evaluation Metrics:

Metric	Mean Value
RMSE	8874.59
MAE	4791.32
R^2	0.63
GCV	36787.54

Explanation based on results:

RMSE (8874.59) shows the average discrepancy between predicted and actual values.

MAE (4791.32) indicates that predictions differ by about 4791 TL on average.

R^2 (0.63) reveals that 63% of the variation in rental prices is explained by the model.

GCV (36787.54) suggests potential gains through further hyperparameter tuning or feature refinement.

2) 5-Fold Cross Validation

Evaluation Metrics:

Metric	Mean Value
RMSE	8222.56
MAE	4567.00
R^2	0.61
GCV	17664.39

Explanation based on results:

RMSE (8222.56) quantifies how far, on average, predictions deviate from actual values.

MAE (4567.00) reflects the typical absolute discrepancy in those predictions.

R^2 (0.61) indicates that 61% of the variation in rental prices is captured by the model.

GCV (17664.39) suggests further tuning of hyperparameters or additional feature engineering could enhance overall accuracy.

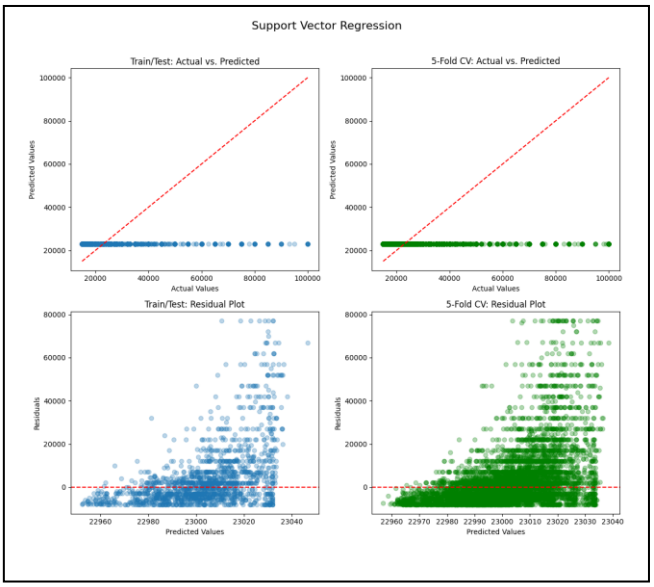

Fig. 12. SVR Train/Test & 5-fold Plots

F. Neural network

1) Train-Test Split

Evaluation Metrics:

Metric	Mean Value
RMSE	7961.08
MAE	4719.30
R^2	0.70
GCV	39961.38

Explanation based on results:

RMSE (7961.08) points to the average prediction-error magnitude.

MAE (4719.30) shows the typical absolute discrepancy in estimates.

R^2 (0.70) signifies the model explains 70% of the variability in rental prices.

GCV (39961.38) indicates additional refinement or feature engineering could further lower error.

2) 5-Fold Cross Validation

Evaluation Metrics:

Metric	Mean Value
RMSE	9082.85
MAE	5242.15
R^2	0.53
GCV	15864.06

Explanation based on results:

RMSE (9082.85) shows the typical deviation from actual rental prices.

MAE (5242.15) indicates an average absolute error of about 5242 TL.

R^2 (0.53) suggests the model accounts for 53% of the variance in prices.

GCV (15864.06) signals potential for further tuning or feature engineering to boost accuracy.

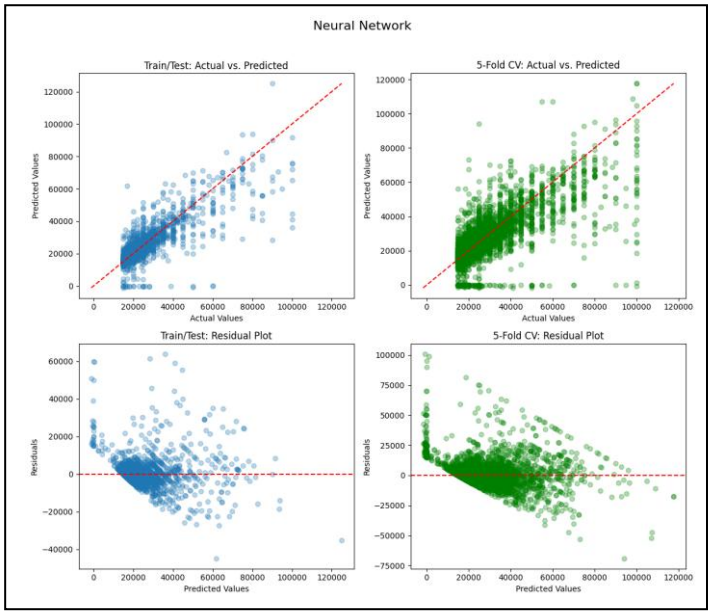

Fig. 13. Neural Network Train/Test & 5-fold Plots

VIII. CONCLUSION

Based on the above experimental findings, Random Forest emerged as the most robust overall regressor. It attained the highest R2R2 scores in both train-test split (0.75) and 5-fold cross validation (0.71), indicating that it captures a large portion of the variance in Izmir's rental prices. This

suggests that the ensemble approach of multiple decision trees significantly boosts predictive accuracy and stability.

Meanwhile, the Neural Network model showed strong performance on the train-test split (R2=0.70R2=0.70), but its 5-fold cross-validation score (R2=0.53R2=0.53) declined more markedly, implying some degree of overfitting. Gradient Boosting and Linear Regression displayed moderate results, whereas Decision Tree alone had lower generalization capability (particularly evident in its cross-validation R2R2 of 0.51). Support Vector Regression (SVR) offered competitive performance but still trailed Random Forest regarding both consistency and overall accuracy.

Although Random Forest outperformed the other models by a clear margin in this study, all the regressors exhibited room for enhancement. Further steps—such as hyperparameter tuning, feature engineering, and gathering additional data—could improve model robustness. Moreover, employing advanced techniques like stacked ensembling or automated feature selection may refine predictions further. Overall, the current findings highlight Random Forest as a strong candidate for estimating Izmir rental prices, but they also underscore the potential benefits of continued model experimentation and optimization.

REFERENCES

[1] R. Raju, A. Neyaz, A. Ahmed, and A. Singh, "Machine Learning for Rental Price Prediction: Regression Techniques and Random Forest Model," SSRN Electronic Journal, Sep. 2023.

[2] Yoshida, T., Murakami, D., & Seya, H. (2024). Spatial prediction of apartment rent using regression-based and machine learning-based approaches with a large dataset. *The Journal of Real Estate Finance and Economics*, *69*(1), 1–28

[3] Wang, Z., Akande, O., Poulos, J., & Li, F. (2022). Are deep learning models superior for missing data imputation in surveys? Evidence from an empirical comparison. *Survey Methodology*, *48*(2), 375-39