CS5228 Final Project - Group 39

JAMAL AHAMED, LALITHA RAVI, PARUL BANSAL, SREELAKSHMI
{A0268403E, A0268254X, A0268257R, A0268357N}
{e1101702@u.nus.edu , e1101553@u.nus.edu , e1101556@u.nus.edu , e1101656@u.nus.edu}
National University of Singapore, Singapore

***Abstract*—This report demonstrates the application of the knowledge acquired in the CS5228 course, showcasing our implementation of data mining methods, exploratory data analysis (EDA), data preprocessing, and model training to predict the rental prices of HDB flats in Singapore.**

I. Introduction

This project tackles the critical issue of accommodation affordability in Singapore. Over recent years, there has been a significant surge in housing costs, both in terms of resale prices and rental rates. Additionally, landlords and real estate agents are eager to discover effective strategies for maximizing rental income. The limited availability of land has made affordable housing a prominent concern for the Singaporean government, leading to the possibility of introducing 'cooling measures' to influence the housing market. The main aim of this project is to provide a comprehensive solution to the housing challenges faced by both potential renters and landlords in Singapore. By harnessing historical data and predictive analytics, we strive to equip individuals and stakeholders with valuable insights into the Singaporean housing market.

II. MOTIVATION AND GOALS

A. Motivation

The significance of this project is underscored by the prevalence of HDB flats in Singapore, which house a substantial-close to 80 percent of the resident population. This substantial housing sector also underpins the expansive resale market for HDB flats in Singapore. Price plays an undeniable role in the decision-making process for buyers, sellers, and real estate agents. For instance, a prospective buyer with children might be keen to understand the cost associated with selecting a relatively new HDB flat situated in proximity to both a primary school and a train station. A seller, on the other hand, may be interested in determining the potential selling price of a 100-square-meter, high-floor HDB flat located near a shopping mall. Real estate agents are tasked with advising their seller clients on setting an optimal price that attracts potential buyers while ensuring sellers can maximize their profits.

In this context, it is profoundly meaningful to discern which property characteristics wield the greatest influence on the market value of HDB flats. By doing so, this project aims to enable not only a better understanding of the factors that drive HDB flat prices but also the ability to predict the resale price of an HDB flat based on its various properties.

B. Goals

Our project is centered on the precise prediction of HDB flat resale prices using a comprehensive set of property attributes, ranging from size and room count to type, model, location, and nearby amenities. Employing various regression techniques, we aim to determine the most accurate and effective approach for this specific task. In addition to price prediction, we will conduct a thorough analysis of attribute importance to unveil the factors that have the most significant impact on HDB flat market values. Furthermore, a detailed error analysis will be undertaken to identify any discrepancies in predictions and understand their underlying causes, providing a comprehensive view of our model's performance. Our project's broader objectives include insightful discussions and findings that transcend raw predictions, contributing to a more profound understanding of Singapore's HDB resale market.

III. EXPLORATORY DATA ANALYSIS AND DATA PREPROCESSING

A. Exploratory Data Analysis

Exploratory Data Analysis (EDA) is a fundamental process for delving into datasets, aimed at uncovering initial insights, identifying patterns, detecting anomalies, and generating summary statistics and visual representations. Different EDA steps carried out are listed below

- During our preliminary data exploration, we utilized the info() and head() functions to generate an overview and statistics of our train and test datasets. The info() output for the train data revealed 60,000 records with 16 columns, including 15 independent features and the dependent feature monthly_rent. We identified the data types of each column and confirmed the absence of null values in both columns and records. Similarly, the test dataset consisted of 30,000 records with 15 independent features, and no null values were present.
- Simple univariate and statistical analysis on numerical columns 'floor_area_sqm' 'lease_commence_date', 'monthly_rent' using ".describe()" function, shows that dataset comprises apartments with floor areas ranging from 34 to 215 sqm. The lease commencement years vary from 1966 to 2019. Monthly rents range from 300 to 6950

SGD. On average, apartments have a floor area of 94.58 sqm, leases commenced around 1990, and the average monthly rent is 2591.51 SGD.

- A basic univariate analysis of categorical columns such as town, region, flat_type, and flat_model reveals the distribution of available rental properties. The results indicate that 4-room and 3-room flat types are the most common. Among flat models, model_a, improved, and new generation are predominant. This analysis provides insights into the popularity of specific housing types and models in different towns and regions. In Figure 1 the distribution of flat types indicates that 4-room HDBs are the most common, followed by 3-room, 5-room, 2-room, and executive flats. This suggests a preference for larger units, possibly due to family dynamics or preferences for more spacious living. In Figure 2 the distribution of flat models indicates a clear preference for model-a flats, followed by improved and new generation models. This suggests that model-a flats are generally more popular among potential buyers or renters, possibly due to their design, amenities, location, or other factors. Figure 3 highlights the pronounced variation in average monthly rent across Singapore's distinct regions. The Central region commands the highest average monthly rent, reflecting its prime location, proximity to amenities, and strong demand from renters. The East region follows with a relatively high average rent due to its accessibility and desirability. Conversely, the North region, with its more suburban character, typically exhibits lower average monthly rents. These regional disparities in rent prices underscore the diverse housing dynamics across Singapore's landscape

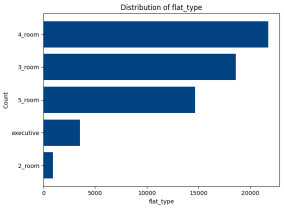

Fig. 1. Distribution of flat types

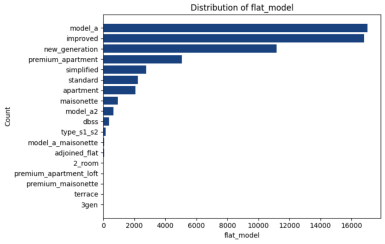

Fig. 2. Distribution of flat models

- A bivariate analysis of categorical data concerning average monthly rent was conducted using histograms. This analysis revealed price distributions across regions,

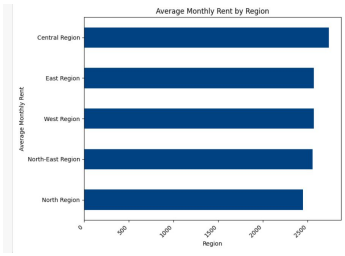

Fig. 3. Average of monthly rent across different regions

towns, and planning areas. Additionally, boxplots were employed to compare flat types, indicating a consistent increase in average monthly rent: 2-room $<$ 3-room $<$ 4-room $<$ 5-room $<$ executive. This finding led to the decision to use ordinal encoding for this column. The box plot of flat models and average monthly rent displayed price variations among different flat models. We also explored how the floor area (in square meters) of rental properties influences the monthly rent (in dollars) using a scatterplot in Figure 4.1 where we can clearly observe a positive correlation between the floor area (in square meters) and the monthly rent of residential properties. This suggests that tenants and property owners consider larger living spaces as a valuable factor in determining rental costs. Furthermore, the plot of lease commencement dates and monthly rent demonstrated that newer apartments commanded higher rental prices, confirming the relationship between property age and rent. This is depicted in Figure 4.2 .

- In our analysis, we also employed multivariate techniques to uncover intricate patterns and dependencies among multiple variables simultaneously within the dataset. Notably, we used a heatmap to visualize the multivariate relationship between 'region,' 'town,' and the median monthly rent. This heatmap, as shown in Figure 5, provides a visual representation of how towns within regions are associated with different rental rates. The central region seems to have higher monthly rent compared to other regional towns.

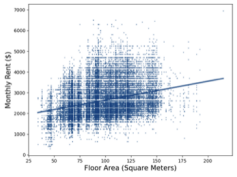

Fig. 4.1. Correlation between floor area and monthly rent

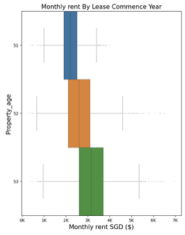

Fig. 4.2. Monthly Rent Distribution by Property Age

Fig. 4. Analysis

B. Data Preprocessing

Data preprocessing is a fundamental and indispensable phase in data mining. It serves as the critical foundation for

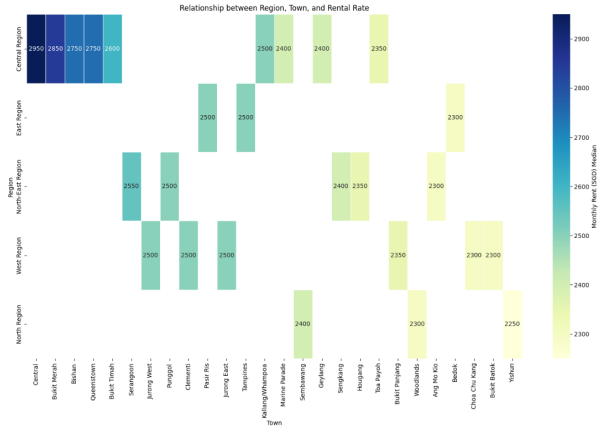

Fig. 5. Heatmap illustrating the median monthly rent across various regions and towns in Singapore.

robust and accurate data analysis and modeling. This essential process involves tasks such as handling missing values, standardizing data formats, removing duplicates, and transforming data into a consistent structure. Data preprocessing not only enhances data quality but also reduces noise and inconsistencies, making the dataset more amenable to advanced analytical techniques. By ensuring that the data is clean, organized, and relevant, preprocessing paves the way for meaningful insights, pattern discovery, and predictive modeling.

1) Data Cleaning:

- We began the analysis by inspecting the dataset for missing values. Ensuring data completeness is fundamental for reliable predictions.
- As a part of data cleaning we identified redundant columns. The furnished column exclusively contained the value **furnished** and the **elevation** column had a uniform value of 0.0 across all records. Since these constant values provide no variation in the dataset, they don't contribute valuable insights into the impact on the dependent variable. Consequently, we dropped these columns from our analysis.
- Standardizing column names and values to a consistent format is important for clarity and ease of analysis. We standardized 'flat_type' and 'flat_model' to *snake_case* and merged S1 and S2 types into 'type_s1_s2' as they both represent special apartments and have similar rental ranges. Additionally, 'town,' 'street_name,' 'subzone,' 'planning_area,' and 'region' were formatted using *title case conversion* (first letter capitalized, rest in lowercase) for consistency.
- After completing the initial preprocessing and standardization steps, we identified and removed duplicate records as they can introduce bias and inaccuracies in the dataset. Using the code snippet `train_duplicates = train_pdf[train_pdf.duplicated()]`, we found 523 duplicate records. We removed these duplicates, retaining only the first occurrence of each unique record ensuring that each data point is unique, reducing the risk of overfitting and data redundancy.

2) Feature Engineering: Feature engineering enhances data interpretability for predicting HDB rental prices. We focus on adapting features to better suit our problem. Some features require encoding or transformation, and new features may need to be created. Numeric features in the core dataset like floor_area_sqm were retained for treebased models and scaled using standard scaling for linear regression. The monthly_rent remained unchanged as it's the target variable. The rent_approval_date was converted into timestamps, splitting it into 'year' and 'month' for richer temporal information.

However, dealing with categorical features is crucial. While decision trees handle them directly, many models require numeric input. To address this, each category is represented by a unique integer using encoding techniques. This ensures accurate processing by machine learning algorithms.

- **Flat type**: We noticed that the flat type has an internal ordering where

$$2room < 3room < 4room < 5room < executive$$

This can be observed in the bivariate analysis of the average rental price as well in Figure 6.

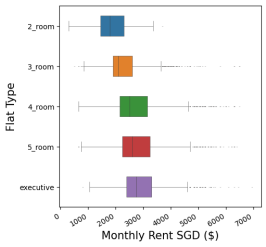

Fig. 6. Boxplot of Monthly Rent by Flat Type

- **Block**: The 'block' column initially contained a vast number of unique values. To mitigate this, we removed the alpha part of the values, resulting in around 980 unique values. Given that this is huge we rule out using One hot encoding (OHE). Despite this reduction, on checking the average number of records per unique 'block' value was approximately 60, indicating extremely low variability in a dataset of almost 60,000 records. Considering that block ranges can again vary from town to town, the actual variability is even lower. Given these factors, it was rational to drop the 'block' column from further analysis.
- **Street Name**: The 'street_name' column contains 543 unique values, making One-Hot Encoding impractical. Upon comparing unique 'street_name' values in the test dataset, we found an additional value "Jalan Ma'Mor" not present in the training dataset. This inconsistency can cause unreliable predictions. To maintain model integrity, we exclude this column from analysis, ensuring consistency between training and testing features.
- **Planning area and town**: After analyzing both the 'town' and 'planning_area' columns, it's evident that there are 26 unique values in 'town' and 29 unique values in

'planning_area'. Out of these, 24 values are common between them. The dissimilar values from 'town' are 'Central', 'Kallang/Whampoa', and from 'planning_area' are 'Novena', 'Downtown Core', 'Rochor', 'Outram', 'Kallang'.
Considering the domain knowledge, Novena, Downtown Core, and Rochor are smaller regions within the central town, and Kallang belongs to the Kallang/Whampoa town. Recognizing the inherent order between these two columns and the absence of extra planning areas in the test set not present in the training set, we opt to drop the 'town' column and solely use 'planning_area' for our analysis. Given that there are nearly 29 values we choose to target encode this column.

- **Subzone**: We calculated the average number of records per unique 'subzone' value in the dataset, resulting in an average frequency of 391.30. This high average suggests that 'subzone' could be a suitable candidate for target encoding. A higher average number indicates stable encoding, enhancing the model's predictive power. Given the large number of values in the subzone and a good representation of the dataset, we used target encoding for the subzone.
- **Region**: There are a total of 5 regions as the number is relatively small and there is no inherent order we perform one hot encoding for the column.
- **Property age**: We introduced a new feature called property age to enhance prediction accuracy. It was derived by subtracting the "rent_approval_date" and "lease_commence_date" features. It indicates how old or new an HDB flat is, a crucial factor for people to rent the apartment. Newer flats are preferred due to their reduced likelihood of age-related issues, making them more attractive to tenants.

3) Auxiliary Data - MRT's, Shopping Malls and Schools: We enhanced our dataset by incorporating auxiliary data related to an HDB flat's location. Despite the low correlation coefficients between 'latitude', 'longitude', and 'monthly rental', we utilized these features to establish a connection between a flat's location and nearby amenities. The value of a flat not only depends on its attributes but also on the convenience of its neighborhood, such as proximity to schools, shopping malls, and MRT stations. To assess this convenience, we calculated distances between flats and various facilities using 'latitude' and 'longitude'. We then created new features representing the closest facility type to each flat, indicating how far the nearest amenity is and the count of amenities within a specific distance. The new features we generated are as follows:

- nearest_mrt_exist: Nearest existing MRT distance to HDB
- nearest_mrt_planned: Nearest planned MRT distance to HDB.
- nearest_school: Nearest primary school distance to HDB.
- nearest_mall: Nearest mall distance to HDB.
- mrt_within_0.5_km and mrt_within_1_km: Gives number of existing MRTs within 500 mts and 1 km. As walking for an MRT at a further distance could be undesirable.
- school_within_1_km and school_within_2_km: Gives the number of primary schools within 1 km and 2 km.
- malls_within_1_km and malls_within_2_km: Gives the number of shopping malls within 1 km and 2 km.

4) Auxiliary Data - COE(Certificate of Entitlement data): COE data and rental prices may seem like two unrelated factors, given their distinct domains within the automotive and real estate sectors. They are subject to different influences and operate on different time scales. However, upon closer examination, our analysis has revealed a significant connection between these apparently disparate variables. Our initial investigation focused on understanding the relationship between monthly rental prices for housing and COE prices. We examined monthly average, maximum, and minimum COE prices with respect to rental prices and identified a strong positive correlation between these two sets of data. The positive correlations suggest that as COE prices rise, the cost of vehicle ownership may increase, potentially leading to increased demand for rental properties, which, in turn, could result in higher rental prices. These findings underscore the interconnected nature of financial and real estate markets, demonstrating that economic factors can impact rental rates. The Figure 7 shows the corresponding correlations. Based on our findings, we have introduced three new features to capture the influence of COE prices on rental prices:

- mean_coe: Average value of Certificate of Entitlement (COE) prices for a month.
- max_coe: The highest recorded COE price within a month.
- min_coe: The lowest recorded COE price within a month.

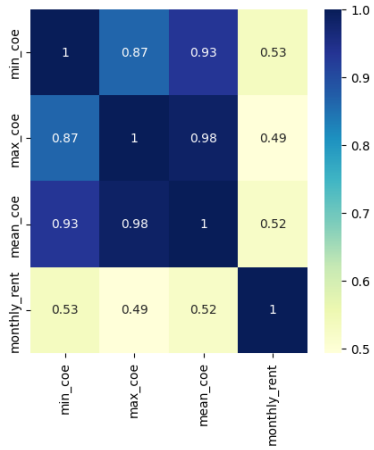

Fig. 7. COE Price Correlation with monthly rent.

5) Auxiliary Data - Stock Prices: Stock prices and rental prices are two very different indicators in the financial and real estate markets, respectively. They are governed by different factors, but we identified that average, maximum, and minimum monthly rental prices have a strong negative correlation

with the monthly average, maximum, and minimum stock prices, indicating that an increase/decrease in the stock price is an indirect indicator of increase/decrease in the general rental prices. Since stock prices vary every week/day and rentals are fixed for a month, we also identified a strong negative correlation between average, monthly rental prices and average, maximum, and minimum previous month's stock prices. This was also done because when deciding rent whether we are in the middle or start of the month, we are not aware of the current month's average but have full information on how the financial market behaved in the previous month. The correlations are presented in Figure 8. Hence, we are using the average, maximum, and minimum of last month's stock prices to gather additional in The new features we generated are as follows:

- last_month_mean: Mean of (average stock price for a month) across all the stocks for the previous month
- last_month_max: Mean of (maximum stock prices) across all stocks for the previous month
- last_month_min: Mean of (minimum stock prices) across all stocks for the previous month

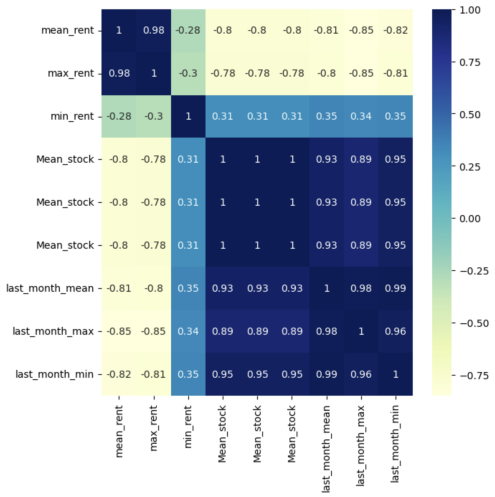

Fig. 8. Stock Price Correlation with monthly rent.

C. Feature Selection

In our project, we employed the feature selection technique using Recursive Feature Elimination (RFE) in combination with GridSearchCV/RandomisedSearchCV (based on model complexity) to identify the most relevant features for a range of machine learning models, including Linear Regression, XGBoost, Random Forest, and LGBM.

By employing a robust five-fold cross-validation approach where the dataset was shuffled before splitting, and a random seed was set for reproducibility, we ensured the reliability of our feature selection method. We applied Recursive Feature Elimination (RFE) to each base model. Then progressively eliminated the least influential features while iteratively fitting each base model at each step. This process continues until the desired number of features is reached.

The GridSearchCV/RandomisedSearchCV was configured to explore the optimal number of features, scanning through a feature range from 1 to 44, and measured performance using the negative root mean squared error (RMSE) as the scoring metric. The goal was to identify the best set of features that minimized RMSE for each model.

This process delivered an optimal feature set for each model, enhancing model efficiency and interpretability, while also offering insights into feature importance specific to each algorithm. These features were considered the most relevant for predicting HDB flat rental prices. The final feature sets can be used for model training and evaluation. Our primary project objective of constructing accurate HDB flat rental price prediction models was achieved in part due to this meticulous feature selection process.

IV. DATA MINING METHODS

We employed regression models to predict the continuous variable of HDB flat resale prices. Various types of regression models were trained and evaluated using the dataset. We rounded the predictions to the nearest 10 because the rent is expressed in multiple of tens.

A. Base Model

We opted for Linear Regression as our base model due to its simplicity. To ensure the model's effectiveness, we split the train data into train and validation sets. On all features, the model achieved RMSE scores of approximately 507.20 (train) and 504.20 (validation). Testing it on Kaggle data yielded an RMSE of 501.09, indicating no overfitting.

To enhance performance, we systematically reduced features using grid search and Recursive Feature Elimination (RFE) techniques, with two scoring mechanisms: R2 and RMSE (calculated using neg_mean_squared_error), thus evaluating the model's performance with different feature subsets. Despite this, Linear Regression demonstrated optimal performance with all features for both the scoring mechanisms Figure 9. We selected the best-performing 44 features based on grid search results. Re-running the model on these features resulted in a Kaggle RMSE of 501.03. With an R2 score hovering around 0.5, it's evident that the model's predictive power is limited given that the output is always a regression line, plane or hyperplane, so it might be restrictive for this dataset.

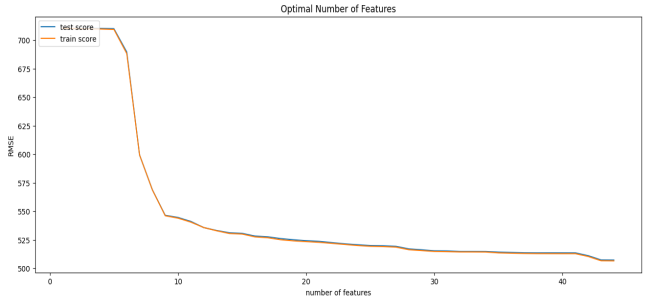

Fig. 9. Model performance for varying number of features wrt RMSE

B. Random Forest

Random Forest Regressor is a robust and versatile ensemble learning algorithm that employs a collection of decision trees to generate predictions. It is particularly well-suited for regression tasks, where the goal is to predict a continuous numerical target variable. Random Forest regressor excels in handling complex and high-dimensional data, effectively capturing non-linear relationships. Each decision tree in the ensemble is built using a randomly sampled subset of the training data, introducing diversity among the trees.

a) Feature Selection: Here We leveraged RandomizedSearchCV, combined with the Recursive Feature Elimination (RFE) technique, to determine the most suitable feature set for the Random Forest Regressor. This approach prioritizes RMSE as the scoring criterion. The visualization of this process can be seen in Figure 10. Following a comprehensive analysis, we ascertained that the optimal number of features was 40, with the highest score achieved at 450. As Random Forest Regressors, like other tree-based models, are insensitive to feature scaling, we chose to forgo feature standardization and employed the features directly in the modeling process.

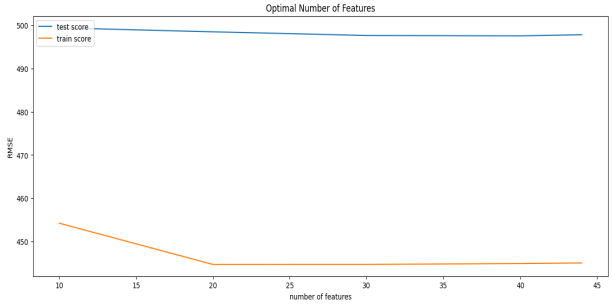

Fig. 10. Mean CV Scores of Random Forest using Various Features

b) Hyperparameter Tuning: In our approach to avoid overfitting in the Random Forest Regressor, we adopted KFold cross-validation with three splits. Our hyperparameter tuning process encompassed the careful selection of key parameters to construct a robust model configuration. For the number of trees in the forest 'n_estimators', we employed a range between 10 and 50 with 10 intervals. The 'max_features' were set to include 'auto' and 'sqrt'. The 'max_depth' was defined from 1 to 20 in 10 increments, appending 'None' to consider all available nodes. To ensure appropriate node splitting, we assessed 'min_samples_split' with 2, 5, and 10 samples, while 'min_samples_leaf' incorporated 1, 2, 4, 5, and 7 samples per leaf node. Further, the model's training process was refined by considering both 'True' and 'False' values for the 'bootstrap' method.

c) Results: After conducting an in-depth analysis using the Random Forest Regressor, the best hyperparameters were determined to be n_estimators=90, min_samples_split=10, 'min_samples_leaf=2, max_features="auto", max_depth=11, and bootstrap=True. The model was trained and evaluated using these parameters. The resulting performance metrics exhibited a train RMSE score of 450.61, highlighting the capacity of the model to effectively learn from the training dataset. Moreover, considering the test set evaluation, the model attained a test mean RMSE score of 497.55, demonstrating a consistent performance on unseen data. The R2 score stood at 0.52, showcasing a moderate level of explained variance in the target variable. Additionally, the standard deviation of the Mean Squared Error (MSE) during cross-validation was measured at 0.69, indicating the stability and reliability of the model's predictions

C. LGBM

LGBM stands for Light Gradient Boosting Machine, is a gradient boosting framework that uses tree-based learning algorithms. LGBM has lower memory usage and achieves high accuracy and speed by employing a histogram-based algorithm that buckets continuous feature values into discrete bins.

a) Feature Selection and scaling: As mentioned above, we used GridSearchCV along with the RFE module to select the best set of features, using RMSE as the scoring criteria. For the LGBM model, the RMSE score didn't change much after 5 features. This is shown in figure 11. Therefore we decided to choose 20 features as the optimal number of features as the feature importance score for greater than 20 features dropped below 30, with the highest score being 353. Since LGBM is a tree-based model, and tree-based models are not affected by the scale of features, we chose not to scale the features and use them directly instead.

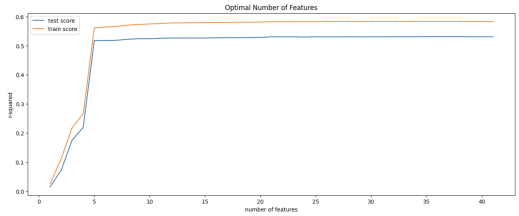

Fig. 11. LGBM Mean CV Score for Different Number of Features

b) Hyperparameter Tuning: Tree-based models tend to overfit, and we need to tune the model to make sure that the model generalizes well without the risk of overfitting/underfitting. We used the following hyperparameter using GridSearchCV with 5 folds: max depth: [5, 8, 10], learning rate: [0.01, 0.1, 0.2], num of estimators: [50, 100, 200], min child samples: [10,20, 30], reg lambda: [0.1, 0.5]

c) Results: After cross-validation, max depth: 5, min child samples:30, num estimators:200 and reg lambda:0.1 gave the best cross-validation cross with an average RMSE of 490 with a standard deviation of 1.96. The final training set without cross-validation had an RMSE of 467 The validation set had an RMSE of 487.07. The test and train scores are very close to each other, indicating that the model is able to generalize well. The mean cross-validation r2 score is 0.53.

D. XGBoost

XGBoost, short for "Extreme Gradient Boosting," is a highly effective gradient-boosting framework for supervised

machine learning, excelling in tasks like regression and classification. It distinguishes itself with its exceptional speed and accuracy, employing techniques such as gradient boosting, regularization, and parallel processing to achieve outstanding results.

a) Feature Selection and scaling: Feature selection is a crucial step in model building, as it helps identify the most relevant features for prediction. Here also, feature selection is performed using Recursive Feature Elimination (RFE) and GridSearchCV, using RMSE as the scoring metric. An initial XGBoost model is trained using all features. The model received an RMSE score of 542.2(on train data) and 539.9(on validation data), with 540.2 on testing it on kaggle data respectively. The closeness between RMSE on the validation data and training data (542.2 vs. 539.9) indicates that the model generalizes well and is not suffering from overfitting.

RFE is applied to select the top N features, where N is determined through hyperparameter tuning. Among our original set of 44 features, our meticulous approach, guided by hyperparameter tuning, revealed that a subset of 20 features proved optimal for further analysis.

b) Hyperparameter Tuning: : Hyperparameter tuning is a crucial step in optimizing the model's performance. Here we used randomized search methods to tune hyperparameters for the XGBoost Regressor. The following hyperparameters are considered in the randomized search with 10 folds: max depth: [3,4,5], min child weight: [1, 5, 10], learning rate: [0.01,0.1,0.2,0.5], subsample: [0.6, 0.8, 1.0], n estimators: [100,200,300], colsample bytree: 0.5,0.6,0.8], eval metric: Set to 'rmse', objective: Set to 'reg:squarederror'. The best hyperparameters and the corresponding lowest RMSE are identified through randomized search.

c) Results: : By Hyperparameter Tuning with Randomized Search, we received a best hyperparameter combination {'subsample': 1.0, 'objective': 'reg:squarederror', 'n estimators': 200, 'min child weight': 10, 'max depth': 5, 'learning rate': 0.1, 'eval metric': 'rmse', 'colsample bytree': 0.6} with least RMSE of 490.9. After refining our model by utilizing the best set of features and with the optimal set of configuration received from tuning, we achieved a Kaggle Test RMSE score of 483.9. This signifies the significant improvement and effectiveness of our model. For the final training and validation dataset, we achieved a train RMSE of 468.2 and a validation RMSE of 487.3. A lower training RMSE indicates that the model fits the training data quite well, but the performance gap between the training and validation RMSE (training RMSE $<$ validation RMSE) suggests there might be some degree of overfitting.

V. RESULTS AND ERROR ANALYSIS

A. Results

The Results for the different models are summarised below in Figure 14 with their key predictors. As our final model, we chose LGBM with a test RMSE of 483.39. Flat Type, Floor Area, Distance to the nearest MRT both existing and upcoming, and distance to the nearest school and mall were the key indicators identified by the model for predicting the monthly rental prices.

To understand the contribution of these variables, we further built another model with just the features listed above and achieved the train and validation RMSE and r2 scores of 625 and 635, 0.23 and 0.20 respectively. This indicates that other features used also bring in a lot of information pivotal to estimating the rents more correctly.

B. Error Analysis

The plot of Predicted vs Actual Figure 12 shows that the model is generalized well for the training and validation set. We can also see some outliers, where the predictions are very far away from line y=x. We also notice that predictions are not centered around the line y=x, but actually seem to deviate a bit.

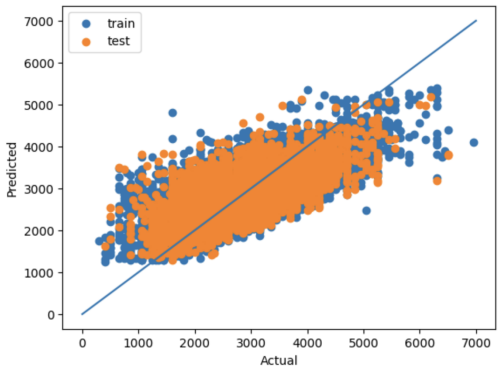

Fig. 12. Model Predictions Vs Actual Monthly Rent

From Figure 14 we can see that residuals are not randomly located around the center, but the range of errors actually increases as the actual value increases. This indicates the presence of heteroscedasticity. This implies that our model is not able to predict higher rent accurately.

LGBM and other tree-based model assumes that the data is iid. The presence of time variables like month and year, indicate a pattern and breaks the assumption of Identical Independent Distribution(iid) data. Hence due to the presence of this heteroscedasticity, the errors are normally distributed with a skewness towards the right for both train and validation sets.

VI. CONCLUSION

In this project, we tried to build an estimator to predict the monthly HDB rental prices in Singapore. We used various feature types from location, flat information, geographic information, and market indicators to estimate the rents. We were able to achieve the RMSE of 483.39 on the test set (Kaggle) with a r2 score of 0.53 on the validation set indicating that there's a lot of variance in the predicted and actual values. Apart from the features used above, rent can vary for a variety of reasons such as the presence and after-effects of the pandemic, re-modeling of the HDB flat, inflation, and other socio-economic factors. From this project, we infer that

monthly rents are very dynamic in nature, and it is possible to estimate the range of the rent, rather than predicting precisely.

					10 Fold Cross Validation		
Model	Num of Optimal Features	Train Rmse	Val Rmse	Test Rmse(Kaggle)	Train R2	Val R2	Top 5 Features
Linear Regression	44	507.201	504.202	501.03	0.498	0.502	flat_model_terrace, flat_model_premium_apartment_loft, flat_model_premium_maisonette, flat_model_dbss, flat_model_type_s1_s2
Random Forest	40	450.61	494.56	497.54	0.515	0.515	flat_type, min_coe, last_month_max, mean_coe, subzone_encoded
XgBoost	20	465.7	485.2	483.97	0.574	0.54	flat_type, floor_area_sqm, property_age, nearest_mrt_planned, mrt_within_0.5
LGBM	20	467.19	487.07	483.39	0.57	0.52	floor_area_sqm, nearest_mrt_planned, nearest_school, nearest_mrt_exist, subzone_encoded

Fig. 13. Comparision of Different Models

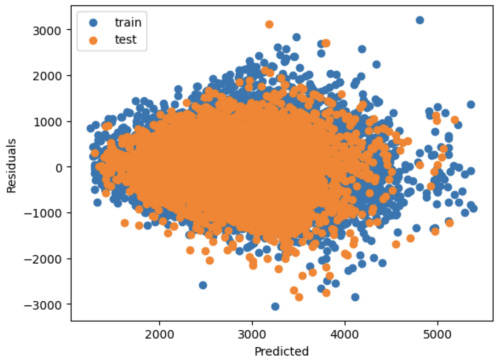

Fig. 14. Train and Test Residuals vs Predicted Value

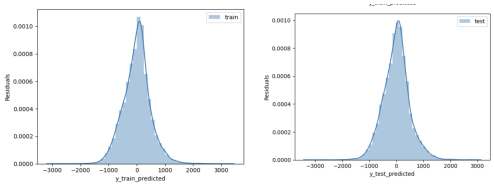

Fig. 15. Residuals for Train and Validation Set

REFERENCES

[1] Bergstra, J., Bengio, Y. (2012). Random Search for Hyper-Parameter Optimization. Journal of Machine Learning Research, 13, 281-305. https://www.jmlr.org/papers/volume13/bergstra12a/bergstra12a.pdf

[2] scikit-learn documentation (version 1.3.2): https://scikit-learn.org/stable/modules/cross_validation.html.

[3] scikit-learn documentation (version 1.3.2): https://scikit-learn.org/stable/modules/feature_selection.html#recursive-feature-elimination.

[4] Scikit-Learn Linear Regression Documentation: https://scikit-learn.org/stable/modules/linear_model.html

[5] Ke, G., Meng, Q., Finley, T., Wang, T., Ma, W., Ye, Q., Liu, T. (2017). LightGBM: A Highly Efficient Gradient Boosting Decision Tree. In Advances in Neural Information Processing Systems (NeurIPS).

[6] Scikit-Learn Linear Regression Documentation: https://scikit-learn.org/stable/modules/linear_model.html

[7] Scikit-Learn Random Forest Documentation: https://scikit-learn.org/stable/modules/ensemble.htmlrandom-forests