**Task 1a**

To maximise the usable sample from the datasets, this subtask leverages population size by analysing both the train and test datasets and transforming suitable variables to discrete categories. Applying IQR, 6.21% of the population with extremely low or high rental price has been excluded.

T-test and Analysis of Variance (ANOVA) were selected to assess the statistical significance of discrete variables that contributed to price. T-test is effective tools to test the null hypothesis of whether a binary variable such as balcony\_is\_available and the number of availabe amenities, has a significant difference in the mean monthly rental price [1].

Several studies have used one-way ANOVA or t-tests to determine whether the impact of amenities on rental prices changed significantly over time[4], to test whether the mean seasonal price indices for beef, pork, and lamb differed across months and between meats [5] and to evaluate the performance of different construction delivery systems[6]. Considering the mix of binary, ordinal, and multi-label categorical, these tools are suitable to get interchangeable metrics (f and p value) to rank the contributing power.

Aligning with the majority of data type, some multi-categorical variables (e.g. utility\_payment) and multi-label variables were transformed to binary by filtering out groups with an imbalanced population whose lacks power for detecting significance at p<0.05.

It is intuitive to presume that the apartments with more listed items in the record is likely to contribute to high-priced apartments. One advantage of ANOVA is the flexibility handling for both ordinal-numeric, multi-categorical and even binary fields [3].

For testing binary fields, F-value from ANOVA is derived by the square of the T-value from the t-test of binary fields. Following assumption 2, Table1b, , F-value was used for determining significance to high rental price. This approach lacks the ability to evaluate the combinatorial effect of multi-label (e.g. appliances = {fridge,\_coffee\_maker}), but we can mitigate by investigating the number of items a record contains or the effect of whether each one-hot encoded label exists has a significantly different mean.

Applying field-specific strategies to each attribute, the data preprocessing phase was both rigorous and complex. Handling of missing values is customized to each individual variable, based on data type. By dropping rows with missing values only in the relevant field prior to conducting statistical tests, the majority of the rental price data remains available for analysis. Details of strategies of decision making on pre-processing can be referred to Figure 1.

A diagram of a flowchart

AI-generated content may be incorrect.

*Figure1. Flow of preprocessing for task1.*

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Count | mean | std | min | max | 25% | 50% | 75% | 95% |
| 38710 | 1341 | 589 | 395 | 3700 | 910 | 1200 | 1600 | 2500 |

*Table1a. Descriptive Statistics Table of monthly rent($)*

*A screenshot of a computer

AI-generated content may be incorrect.*

*Table 1b. Descriptive Statistics Table of monthly rent($)*

Based on the F-values(Table2a,c), new\_construction, number\_of\_bathrooms and elevator are the three most significant discrete variables that contribute to high-priced properties. Although duration has the third-highest F-value and the largest mean difference, its impact is likely due to the inherent price increase when converting daily rates to monthly rates, which is an expected outcome. Note that number\_of\_bathrooms has more significance than number\_of\_rooms by having a greater f-value (3050 vs. 1014) , a greater mean difference across intervals, and a smaller standard error and narrower 95% CI. This means that the variance in price explained by number\_of\_bathrooms is nearly three times higher than that of number\_of\_rooms, especially for the larger groups, indicating more precise estimates with stronger reliability. Elevator gives the third great f-value and mean, median difference.

A table of numbers and a few black text

AI-generated content may be incorrect.

*Table 2a. T-tests result of top ten most significant indicators*

A table with numbers and letters

AI-generated content may be incorrect.

*Table 2b. Describtiive statistics of top six most significant indicators*

*A screenshot of a data

AI-generated content may be incorrect.*

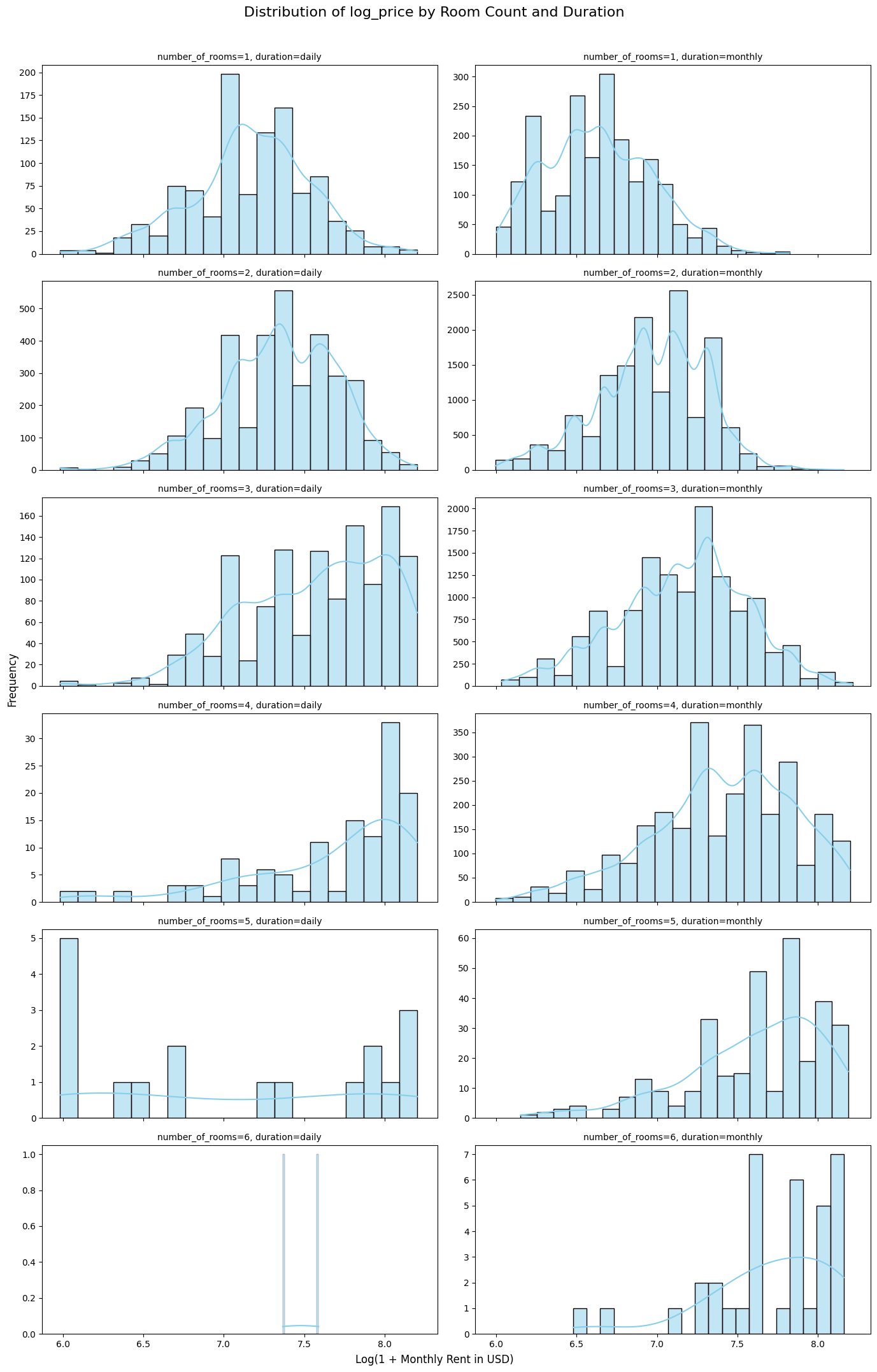
*Table 2c. F-test from ANOVA*

**Task1b**

Correlation is a statistical tool applied to measure the strength and direction between two variables. As stated in chapter 7 in [1], assuming a normal distribution, Pearson aims to measure the degree of linear associations between two continuous variables. Whereas, according to the definition in chapter 14 in [7], Spearman is for assessing monotonic relationship between two ordinal or non-normally distributed variables without needing the assumptions of linearity or normality.

While it is sensible to assume a normal distribution on the overall log price distribution, this assumption only holds from number\_rooms = 1 to 3. The distribution (Figure 2) becomes increasingly skewed or irregular for larger room counts (4-6 rooms), especially when grouped by duration. Spearman has been widely used in housing price studies to rank the determinants of housing demands[8] and correlation between floor, year\_on\_ construction and rent[9].

Based on the discrete, ordinal and categorical nature of the data, it is justified to use Spearman’s rank correlation coefficient, rs, boxsplots and descriptive statistics to assess correlations between number\_of\_rooms-price, duration-price and number\_or\_rooms-duration. Following the preprocessing steps (Figure1), the underpopulated groups such as number\_of\_rooms > 6, are removed using a threshold of sample size >= 30. This step ensures that correlations are inferred only from statistically meaningful group sizes, improving the robustness and validity of the analysis. The result is presented in Table and Figure 3.



*Figure2. Price distribution of log\_price by room count and duration*

A table with numbers and symbols

AI-generated content may be incorrect.

*Table 3. Spearman Correlation coefficient among number\_rooms, duration and log\_price*

*A screenshot of a graph

AI-generated content may be incorrect.*

*Figure 3. Distributions and CI of Log-Transformed Rent by Rooms Count and Duration.*

In monthly rent, number\_of\_rooms shows weak-moderate positive correlation with monthly price, verifying that larger apartments are generally more expensive. The weak negative correlation between duration and rent can be explained by the more expensive cost for daily (daily:0, monthly:1). A weak positive between number\_of\_rooms and duration implies that the bigger apartments tend to be rented monthly.

A moderate negative between duration and unit price means daily rent is more costly. Despite that bigger apartment is more expensive in monthly rent, it has a lower cost in unit price.

**Task1c**

In task 1c, the preprocessing has adopted flows in both Figure 1,5. To enhance the breadth of meaning about address, processes such as translation and extracting geographical data from external API, reconstruction of address, have been conducted. Besides parsing cities or districts from the addresses, lists of redefined cities and their respective coordinates in Armenia and district in Yerevan, have been used to classify missing values by selecting the location with the closest distance. This decision is based on the study by Truong el at. [16], where the address is presented in the form of coordinates, district, and distances to the capital.

Assuming the area of study is limited to Armenia, all unpopulated outlier in other countries have been removed. Distance is derived by using coordinates to obtain the distance to the centre of Yerevan. This gives extra specification of how address determines the rental price based on the assumption that the proximity to the capital, correlates with higher rental prices. While removal of outliers with extreme values is only on training set to prevent overfitting to noise, they are retained in test set to ensure fairness.

A diagram of a data flow

AI-generated content may be incorrect.

*Figure 5. Flow of preprocessing for task1c*

Various regression tools can be applied to effectively predict rent price. A summary of tools used on relevant research can be in tableX [10-16] . Kim & Kim have stated Random Forest(RF) and XGBoost outperformed traditional regression in spartial datasets, and Lasso helped in variable selection [13]. Raju et al. have proved XGBoost showed a slightly better accuracy, but RF offered better interpretability [11]. Li et al. show that XGBoost achieved the best performance overall and recommend it for structure rental data due to its ability to capture nonlinear relationships.[12]

A white grid with black and red text

AI-generated content may be incorrect.

*Table4a. Tools used across studies*

RF, XGBoost, Lasso and Ridge are four regression models selected to evaluated e the significance of address as a predictor of rental price. Each model was trained and analysed under two conditions, with and without address-related features. While Lasso and Ridge represent linear models with L2 and L1 regularization respectively, RF and XGBoost are tools with non-linear modelling capabilities. The best model are picked to assess the feature importance metrics for indicating the first three additional attributes that contributed to high rent rates. The models were optimized using 10-fold cross-validation with GridSearchCV to determine the best hyperparameters. These tuned models were then retrained on the entire training dataset and evaluated on the full test set. The results are presented from Table 4b-6d.

A screenshot of a computer

AI-generated content may be incorrect.

*Table4b. performance metrics of models with address features in 10-fold CV*

A screenshot of a computer

AI-generated content may be incorrect.

*Table4c. performance metrics of models without address features in 10-fold CV*

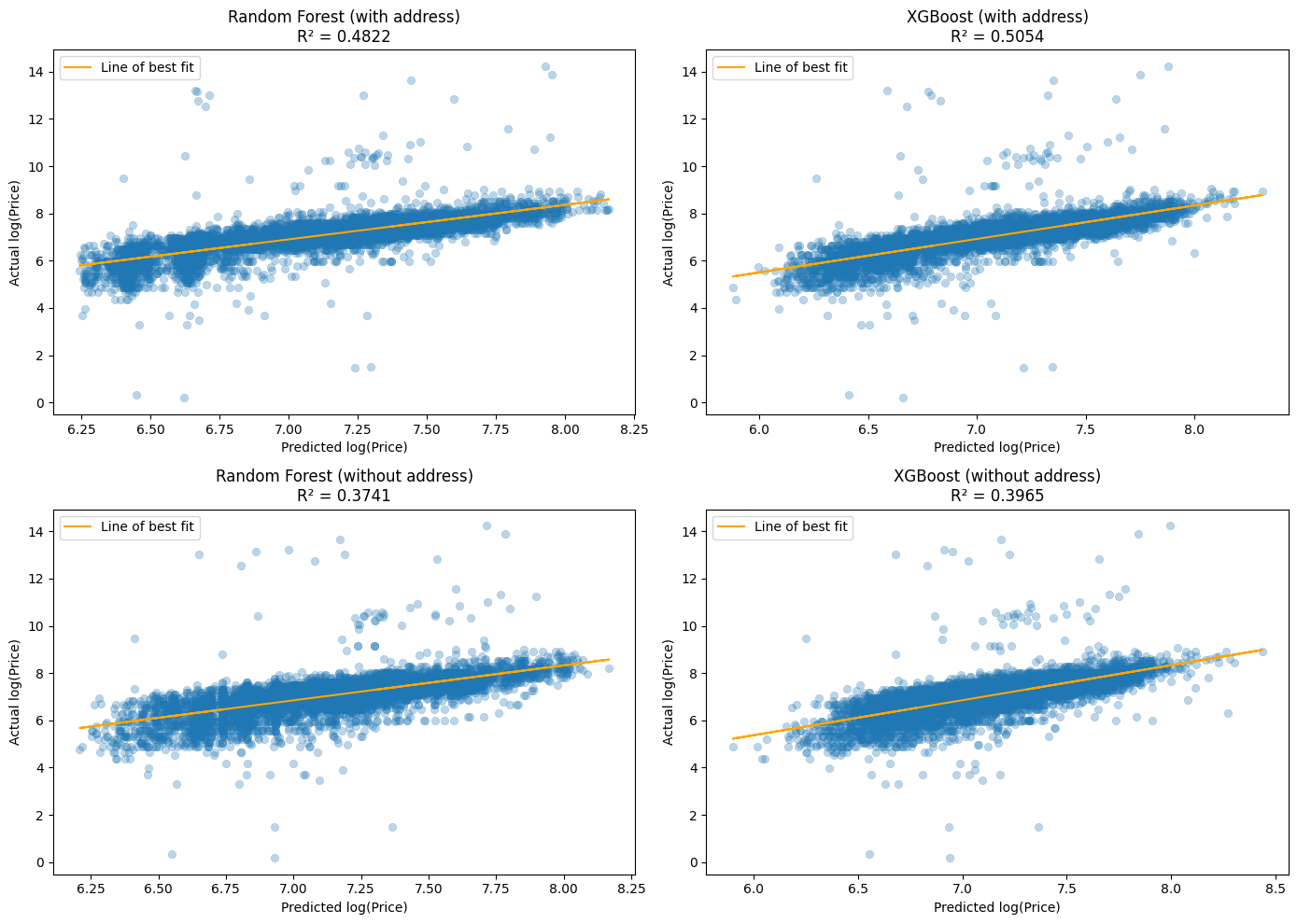
In the result of the validation stage, between Ridge and XGBoost, demonstrates an increase of about 0.1 in and a 0.02 decrease in RMSE . This pattern presents in RF, XGBoost and Ridge.

XGBoost clearly outperforms in test with or without address (Table 4b-c), where Ridge follows tightly by little differennce in RMSE. 3 out of 4 results support that address is a significant predictor of price increase. The absent of effect of address in Lasso is likely due to the L1 regularisation that eliminates highly dependent variables(e.g. address).

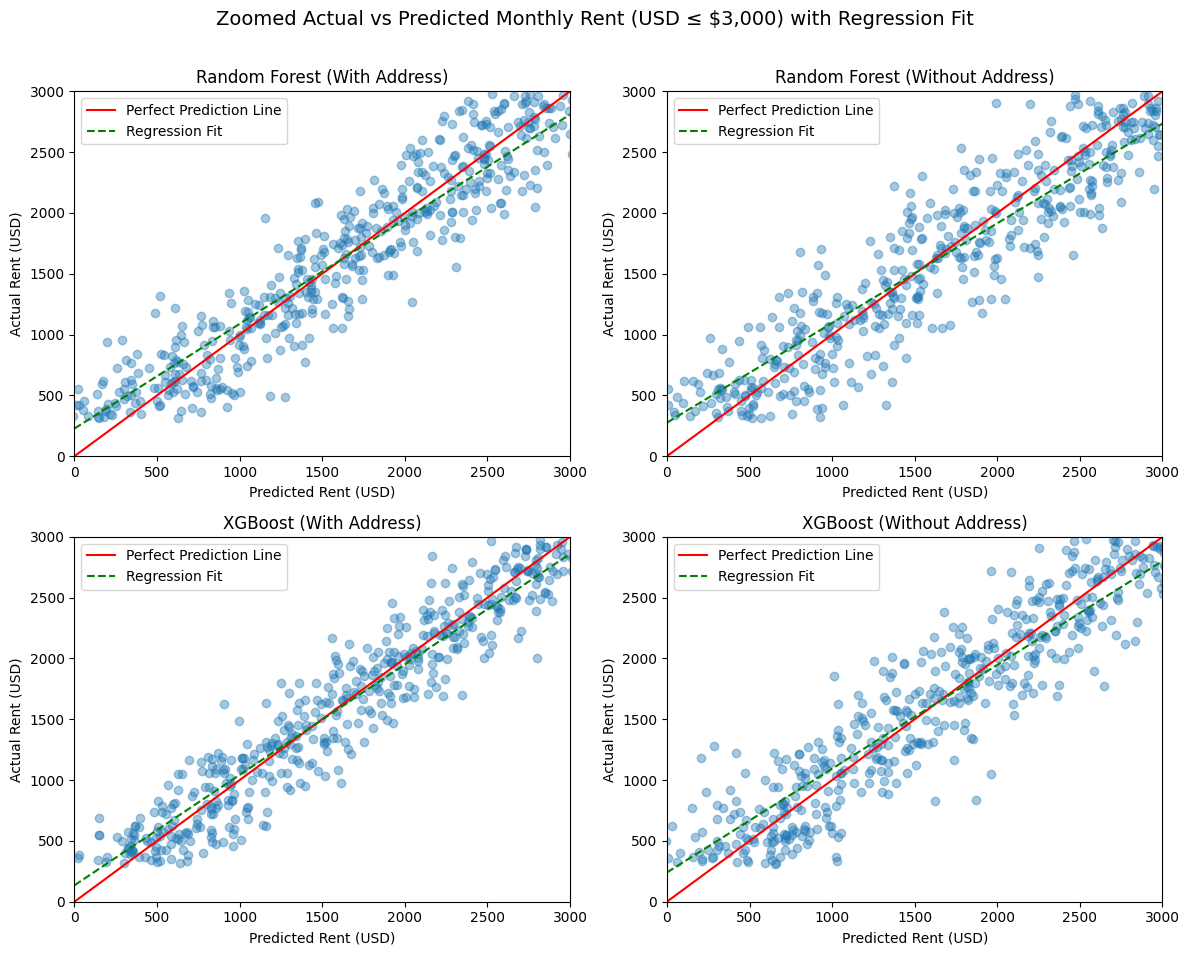
A screenshot of a computer

AI-generated content may be incorrect.

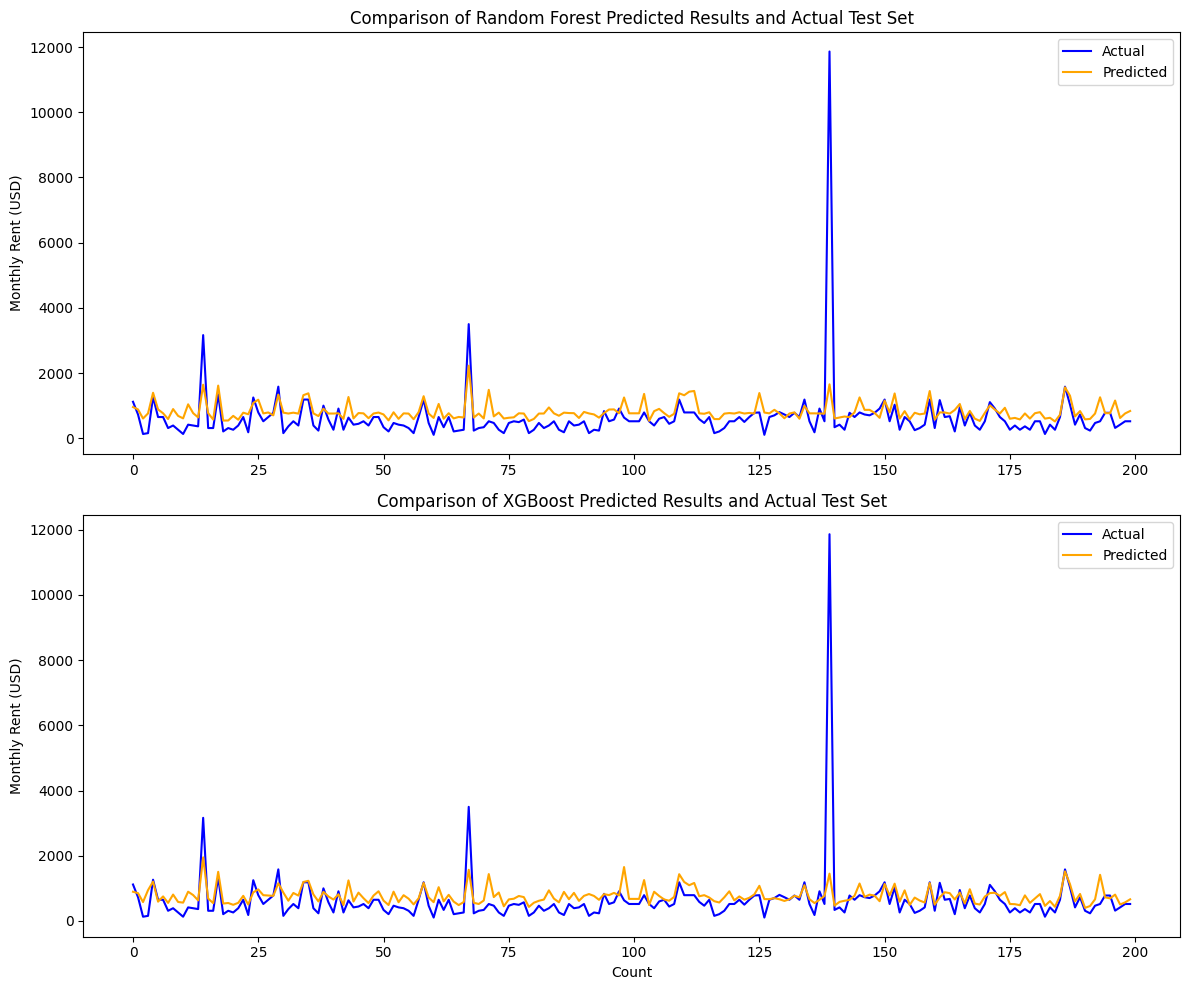
*Table 4d. Comparsion of test result with or without address*



*Figure 6a. Effect of Address Feature on Model Performance: Actual vs Predicted Log(Price) using Random Forest and XGBoost*



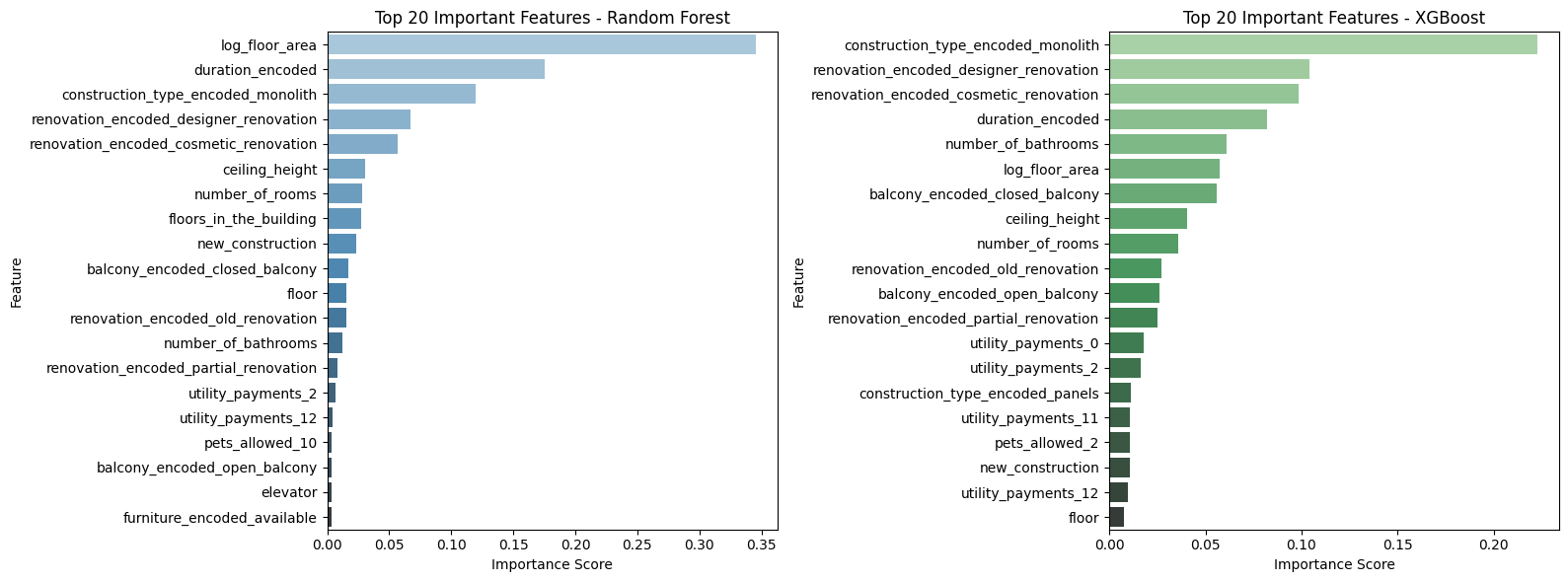
*Figure 6b. Model Prediction Accuracy for Monthly Rent (USD): Random Forest and XGBoost (Zoom in)*



*Figure 6c. Model Prediction Accuracy for Monthly Rent (USD): Random Forest and XGBoost*

A less variance in predicted log price by XGBoost can be observed(Figure 6a).

In a closer look at the prediction among best fitted line, compared to Random Forest, the predictions made by XGBoost appear more concentrated along the x-axis, indicating less variance in predicted values. From the plots, XGBoost model produced a predicted curve that more closely follow the shape of actual values (Figure 6c) and a more accurate slope in regression fit (Figure 6b). This result aligned with previous studies[12,13,16].



*Figure 6d. Importance scores ranked by RF and XGBoost (without address)*

While XGBoost has better prediction on rent price, feature importance ranked using RF gives more interpretable result of the top three attributes. Excluding duration, floor\_area, construction\_type\_monolith, and renovation\_designer\_renovation are the most significant features associated with high rental price.

**Task2 Part1**

A diagram of a document

AI-generated content may be incorrect.

*Figure 7, ERD of schema design for apartment\_for\_rent*

It is reasonable to model a many-to-one relationship between apartment and address due to repeated, generic addresses. This can prevent duplicate addresses to save space, enhance grouping of attributes in analytics. Assuming similar cleaning and preprocessing from task1 precede the loading of csv to SQL db. The address table consists of both the raw and translated address strings, along with derived or parsed city and district, and geographic coordinates(lat,lon) queried from external APIs and STREET\_ADDRESS. One key feature of this design is the decomposition of composite Street\_Address which comprises Street, Street\_No, Block and Block\_No. Following approaches by Elmasri et al. [17, p65-67], enforcing atomicity and improved query flexibility. No attributes as multi-value fields are in APARTMENT. Using APPLIANCE as an example, the normalisation process is shown below.

A screenshot of a phone

AI-generated content may be incorrect.

*A screenshot of a computer

AI-generated content may be incorrect.*

*A table of appliances with text

AI-generated content may be incorrect. A screenshot of a table

AI-generated content may be incorrect.*

*Table 5. Normalisation process from UNF to 3NF*

The UNF is not atomic due to the multi-label values in Appliances that violates 1NF.

Leveraging junction table, APPARTMENT\_APPLIANCE with APPLICANCE as a map, 1NF is derived by decomposing the multi-value into a single item in a row. Given the overlapping characteristics of these fields, this design allows reusing existing id to represent the item rather than duplicating entries each time a new combination appears. In 2NF, the multi-valued non-key attribute, Appliances is stored separately to remove partial dependencies. A junction table, Apartment\_Appliance establish a mony-to-many relationship between Apartment\_No and Appliances\_No. Using a composite key , while Appliances\_No also serves as foreign key referencing Appliance for its description. In 3NF, the address-related fields are the transitive dependency on Address\_No. Removing them and use an ADDRESS table to represent the whole details. This improves efficiency when update address is needed, especially for incomplete address entries(City=’Yerevan’, District=’Kentron’, Null, Null..) one entry is updated in ADDRESS table rather than updating multiple cells.

**A screenshot of a computer code

AI-generated content may be incorrect.**

A screenshot of a computer program

AI-generated content may be incorrect.

A screenshot of a computer code

AI-generated content may be incorrect.

*Figure 8a. SQL sample code for task2bi*

Task2bi involves four sub-tasks, insert only if it does not exist, incrementing address\_no if a new address was generated and finally insert apartment with a newly-added or exisiting address\_no. This approach follows chapter 3 [18] to use NOT EXISTS with a subquery to ensure no such address exists and chapter 9 [19] to apply CASE functions such as IFNULL for condition check before setting address\_no. A select clause with multi-condition checks is used for task2bii.

As described by Silberschatz et al. in chapter 7[20], a GROUP\_BY clause is used to aggregate data across records into groups of currency and return the average price in each currency.

*A screenshot of a computer

AI-generated content may be incorrect.*

*Figure 8b. SQL sample code for task2bii*

*A black text on a white background

AI-generated content may be incorrect.*

*Figure 8c.. SQL sample code for task2biii*

**Task2 Part2**

To meet the growing needs, a hybrid approach of using both relational database and Hadoop Distributed File System (HDFS) is proposed based on Figure 9.

A white background with black text

AI-generated content may be incorrect.

*Figure 9. Assumptions made about the scenarios*

A diagram of a diagram

AI-generated content may be incorrect.

*Figure 10. Proposed architecture of Spark+HDFS*

Addressing the limitation of vertical scaling in traditional SQL systems, a future-proof solution, HDFS offers a scalable horizontal scaling solution. Hadoop is an open-source framework designed for handling large datasets across clusters [21]. It splits the big file into blocks, such as 64MB or 128MB, and spreads them across machines in the clusters to support parallel processing. While a modest increase in data is anticipated , based on the ERD design, the junction tables such as APARTMENT\_APPLIANCES imply a high-cardinality relationship that could result in a rapid growth of rows over time.

Yahoo! and Twitter have integrated HDFC into their architecture for improved fault tolerance and scalability for large-scale data. As Shavachko et al. [22] illustrate the flexibility of efficient global data operations achieved by elastic scaling, the compatibility with SQL-based queries over distributed storage, and the ability to support batch analysis and near real-time queries. A comprehensive study in HDFS applied in Yahoo! has been described in [23] that gives measurable insight into how distributed storage architectures support high-throughput data ingestion. While near real-time alerts can be achieved by scheduled jobs, a considerable effort has been dedicated in enabling HDFS to support real-time event-driven triggers with appropriate orchestration as shown in [24].

Kafka is a distributed log system that collects live updates. Flink is a distributed system that processes streams of data from Kafka (or other sources) in real-time, using logic defined by the user[27].

Given assumption 3, Kafka+Flink can offer low-latency joining of multiple event streams for real-time alerts. It seamlessly handles delayed data, assuming a continuous input stream, the event time and watermarking system in Flink can align the count of data with a defined window. For example, when there is a delay in translating data (Google Translate API) or getting lat, lon, an alert, “if more than 10 apartments are added in the last 5 minutes, send an alert.”, may be perturbed by the delayed response time from API calls which may lead to inaccurate alerts. However, it requires a high setup cost and manpower (DevOps) to maintain performance due to the complexity of state management.

Using cloud-native data warehouse can support parallel processing, high-cardinality analytics and compatible with alerting pipelines.

For example, the count records of multiple-label variables require monitoring to avoid the explosive cardinality. The data is loaded into BigQuery or Redshift, then scheduled SQL queries count records in multiple-label junction table to detect crossing thresholds and create alerts via integrations with dashboard tools.

The studies by AWS engineers compares old cloud data warehouses and on-premise systems [30]. It highlights Redshift’s flexibility of handling rapid increase of requests through elastic resource scaling.

A white sheet with black text

AI-generated content may be incorrect.

*Table 6. Comparison of Data Pipeline Architectures*

In Table 6, the pros and cons of different data pipeline architectures are summarised from studies[22-30]. While cloud-based data warehouse such as Amazon Redshift also offers scalable, high-speed SQL queries with data lake integration, it inherits risks of vendor lock-in and high potential costs. Given assumption 5 (Figure 9), cloud-native or hybrid-cloud are not being considered in the initial phase, as demonstrated in Twitter’s migration [23], there is a performance gap in replacing on-premise machines by cloud machines. Kafka+Flink can also be considered when the listings involved and processing pipelines become more complex, especially in scenarios requiring frequent, real-time alerts. However, based on assumption 2 (Figure 9), , rental price alerts do not demand strict real-time responsiveness. For example, flagging properties with unusually high renovation activity or price surges within a 24-hour window can still meet operational needs. By assumption 4 (Figure 9), Sparks stands out by its speed in batch analytics in ML pipelines. While Spark does not offer millisecond-level responsiveness like Flink, it offers a balanced responsiveness–cost trade-off for batch-driven alert systems with hourly or daily triggers.

**Task 3**

The first privacy issue is the risk of re-identification through indirect identifiers. Even though names are absent from the database, the geographical and personal information such as lat, lon and age, can facilitate re-identification of individuals, particularly when cross-referenced with external datasets or publicly available information [31]. In the context of customer-facing application, this risk becomes significant if address-level details are displayed or processed without anoymisation.

For example, relating apartment rental price with demographic data can enable inference of personal behaviors and socio-economic status. This may lead to profiling of groups (e.g. comparing status quo of people from different city), possibly resulting in biased service offerings or price discrimination. Simchi-Levi et al. [32] highlight how dynamic pricing can effectively learn from demand patterns, which, when paired with detailed personal data, raises concerns about unfair price discrimination in systems that interact directly with customers. Lobel el al. [33] point out the feasibility of using linear models for personalised pricing that implies the ease of individual targeting.

Differential privacy(DP) is a broad, formal framework for ensuring individual privacy when analysing or querying data[32-35]. It measures how much the presence or absence of a single individual’s data can change the result of an analysis. Adding calibrated random noise at some stage of data extraction, query outputs remain statistically valid. For example, adding DP tools, such as Privacy library in python code in the SQL interface for analysts, in the data extraction pipeline, can reduce the risk adversarial re-identification.

Inference attacks through machine learning models can reveal characteristics of individuals by making queries on or reverse-engineering the ML model used by adversaries. Fredrikson et al. demonstrate the possibility of reconstructing sensitive attributes from confidence scores of ML models by linking them to a face recognition system.[36]. In this study, given the tendency of adding ML models to client interface.

To mitigate this, the ML model should be trained by only essential features, excluding gender and age, for example. Several studies [37-39] has demonstrated regularization techniques such as dropout can reduce overfitting, memorization of data, in the ML models and deep neural network, thereby mitigating risk of privacy leakage attacks.

Lastly, cloud-based solutions connected multiple APIs could potentially expand the exposure to a broader network. This implies any unauthorized data access can occur through poorly configured APIs or compromised systems. In this study, when the ML model is centralised on a cloud data pipeline, it could unexpectedly disclose personal and sensitive information and compromise the entire dataset by a single breach.

Several real-world data leak cases because of improper access restrictions are discussed in [31]. To mitigate this, using encryption for message passing and enforcing role-based access control across HDFS and SQL layers. Deploying ML model in local disk, federated learning has been proven to reduce the risk of re-identifications and inference attacks[40,41] . In the given scenario, federated learning allows shared models to be trained in local branches and aggregates updates to the global model. Local machinces passe only trained weights to the aggregate server to reduce risk of exposing personal data from local branches (Figure 11).

*A diagram of a branch model

AI-generated content may be incorrect.*

*Figure11. FL in Client-server architecture*

**Reference:**

[1] A. Field, *Discovering Statistics Using IBM SPSS Statistics*, 4th ed. London, UK: SAGE Publications, 2013.

[2] T.-K. Kim, “T test as a parametric statistic,” *Korean Journal of Anesthesiology*, vol. 68, no. 6, pp. 540–546, 2015.

[3] G. Kutner, C. Nachtsheim, J. Neter, and W. Li, *Applied Linear Statistical Models*, 5th ed. New York, NY, USA: McGraw-Hill/Irwin, 2005.

[4] R. J. Herring and G. A. Lind, “Rental Amenities and the Stability of Hedonic Pricing Models,” *The Journal of Real Estate Research*, vol. 43, no. 3, pp. 329–354, 2021.

[5] J. M. Marsh, J. Mintert, and T. C. Schroeder, “Differences in Seasonal Price Patterns Among Beef, Pork and Lamb,” *Agricultural and Resource Economics Review*, vol. 26, no. 2, pp. 123–132, Oct. 1997.

[6] M. A. El Wardani, M. J. Messner, and D. R. Horman, “Construction Delivery Systems: A Comparative Analysis of the Performance of Systems Within School Districts,” *Journal of Construction Engineering and Management*, vol. 132, no. 6, pp. 475–484, 2006.

[7] P. Newbold, W. L. Carlson, and B. Thorne, *Statistics for Business and Economics*, 8th ed. Upper Saddle River, NJ: Pearson Prentice Hall, 2010.

[8] J. I. Ezeokoli and S. O. Ogbodo, “Determinants of housing demands and residential rent costs in an emerging city in Southern Nigeria,” *International Journal of Advanced Academic Research (Social & Management Sciences)*, vol. 6, no. 6, pp. 1–14, Jun. 2020.

[9] R. Czerny and A. Rzeńca, "Apartment Rental Market in Border Cities of Poland," *Sustainability*, vol. 13, no. 12, p. 6863, 2021.

[10] H. Bhagat, M. Agarwal, and M. Pandey, “Machine Learning for Rental Price Prediction: Regression Random Forest Model,” *International Journal of Scientific Research in Computer Science, Engineering and Information Technology*, 2020.

[11] A. Raju, A. Neyaz, A. Ahmed, and A. Singh, *Machine Learning for Rental Price Prediction: Regression Techniques and Random Forest Model*, NIET, Greater Noida, India.

[12] S. Li, J. Wang, and H. Wang, “Research on Renting Price Prediction Based on Machine Learning,” in *Proceedings of the 2020 3rd International Conference on Artificial Intelligence and Big Data*, 2020, pp. 98–102.

[13] M. Kim and Y. Kim, “Spatial Prediction of Apartment Rent Using Regression-Based and Machine Learning-Based Approaches with a Large Dataset,” *ISPRS International Journal of Geo-Information*, vol. 10, no. 8, p. 531, 2021.

[14] M. Benvenuti et al., “An Automatic Tool for the Determination of Housing Rental Prices: An Analysis of the Italian Context,” *Sustainability*, vol. 14, no. 3, p. 1171, 2022.

[15] X. Li, Y. Wang, S. Basu, K. Kumbier, and B. Yu, “A Debiased MDI Feature Importance Measure for Random Forests,” *Proceedings of the 33rd Conference on Neural Information Processing Systems (NeurIPS)*, 2019.

[16] Q. T. M. Nguyen, H. D. Bo, and M. Mei, "Housing price prediction via improved regression models," *International Journal of Advanced Computer Science and Applications (IJACSA)*, vol. 12, no. 6, pp. 384–390, 2021.

[17] R. Elmasri and S. B. Navathe, *Fundamentals of Database Systems*, 7th ed., Boston, MA, USA: Pearson, 2016

[18] A. Silberschatz, H. F. Korth, and S. Sudarshan, *Database System Concepts*, 6th ed. New York: McGraw-Hill, 2011.

[19] P. DuBois, *MySQL*, 3rd ed. Indianapolis, IN, USA: Sams Publishing, 2005.

[20] C. J. Date, *An Introduction to Database Systems*, 8th ed. Boston: Pearson/Addison Wesley, 2004.

[21] GeeksforGeeks. (n.d.). How does Hadoop handle parallel processing of large datasets across a distributed cluster? Retrieved July 8, 2025, from <https://www.geeksforgeeks.org/data-engineering/how-does-hadoop-handle-parallel-processing-of-large-datasets-across-a-distributed-cluster/>

[22] K. Shvachko, H. Kuang, S. Radia, and R. Chansler, “The Hadoop Distributed File System,” *Proc. IEEE 26th Symp. on Mass Storage Systems and Technologies (MSST)*, Incline Village, NV, USA, pp. 1–10, May 2010

[23] N. Agarwal, H. Purohit, and M. Ahmad, “Serving Hybrid Cloud SQL Interactive Queries at Twitter,” Proc. VLDB Endowment, vol. 13, no. 12, pp. 3131–3144, Aug. 2020

[24] A. Jain, V. Srinivas, D. Song, A. S. Miglani, J. Chopra, and S. Shah, “Serving Hybrid Cloud SQL Interactive Queries at Twitter,” *Proc. VLDB Endowment*, vol. 15, no. 12, pp. 3531–3543, Aug. 2022.

[25] L. Shi, “Real Estate Marketing Adaptive Decision-Making Algorithm Based on Big Data Analysis,” *Mobile Information Systems*, vol. 2022, Article ID 1462125, 9 pages, 2022.

[26] S. Roy, V. Narayanan, and V. Arora, “Joins for Hybrid Cloud Data Warehouses: A Fortune 500 Case Study,” *Proc. 2019 IEEE Int. Conf. on Big Data (Big Data)*, Los Angeles, CA, USA, pp. 131–140, Dec. 2019

[27] A. Phung, “Apache Flink Stream Processing Use Cases With Examples,” *Confluent Blog*, Jun. 2023. [Online]. Available: <https://www.confluent.io/blog/apache-flink-stream-processing-use-cases-with-examples/>

[28] A. Salih, A. M. Zeki, and W. Ismail, “Performing Evaluation of Distributed Computing Environments with Hadoop and Spark Frameworks,” *Int. J. Comput. Sci. Netw. Secur.*, vol. 22, no. 1, pp. 146–152, Jan. 2022.

[29] G. Grünbichler, S. Rass, and M. Hutle, “Real-Time Event Joining in Practice with Kafka and Flink,” *Proceedings of the 2022 IEEE International Conference on Big Data (Big Data)*, Osaka, Japan, Dec. 2022, pp. 3120–3129.

[30] A. Gupta, I. Rao, M. Sarwal, V. Sreekanti, S. Mehrotra, and R. Srivastava, “Amazon Redshift Re-invented,” *Proceedings of the VLDB Endowment*, vol. 16, no. 12, pp. 3674–3686, Aug. 2023.

[31] M. Gürses and B. Hoboken, *Privacy by Design: A Counterfactual Analysis of Google and Facebook Privacy Incidents*, SSRN Electronic Journal, 2011.

[32] D. Simchi-Levi, K. Simchi-Levi, and E. Hazan, “Dynamic Pricing,” *Operations Research*, vol. 63, no. 2, pp. 237–253, Mar.–Apr. 2015.

[33] A. Lobel, "On the Surprising Sufficiency of Linear Models for Dynamic Pricing with Demand Learning," *Management Science*, vol. 69, no. 4, pp. 2073–2096, Apr. 2023.

[34] Z. Liu, Y. Li, C. Joe-Wong, and H. Yu, “Privacy-preserving dynamic personalized pricing with demand learning,” *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 36, no. 5, pp. 5793–5801, Jun. 2022.

[35] C. Dwork, V. Feldman, M. Hardt, T. Pitassi, O. Reingold, and A. Roth, *The Algorithmic Foundations of Differential Privacy*, 2014.

[36] M. Fredrikson, S. Jha, and T. Ristenpart, “Model inversion attacks that exploit confidence information and basic countermeasures,” in *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security (CCS)*, Denver, CO, USA, 2015.

[37] R. Shokri, M. Stronati, C. Song, and V. Shmatikov, “Membership inference attacks against machine learning models,” *2017 IEEE Symposium on Security and Privacy (SP)*, pp. 3–18, 2017.

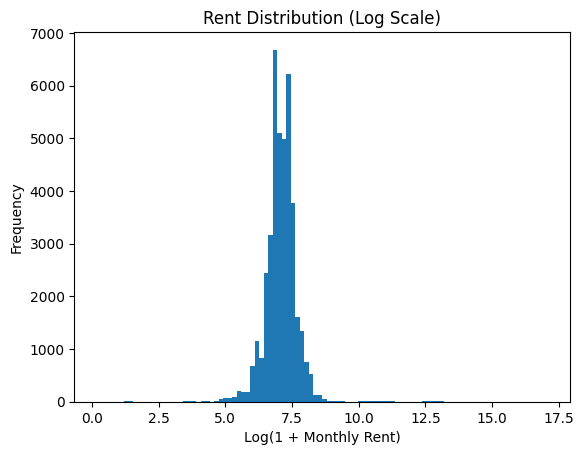
[38] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*, 2nd ed. New York: Springer, 2009.

[39] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, “Dropout: A Simple Way to Prevent Neural Networks from Overfitting,” *Journal of Machine Learning Research*, vol. 15, pp. 1929–1958, 2014.

[40] J. Xu, S. Liu, S. K. Das, and L. Wang, "An overview of implementing security and privacy in federated learning," *Computer Networks*, vol. 203, p. 108597, 2022. [Online]. Available: doi:10.1016/j.comnet.2021.108597.

[41] T. Li, A. K. Sahu, A. Talwalkar, and V. Smith, "Federated learning: Challenges, methods, and future directions," *IEEE Signal Processing Magazine*, vol. 37, no. 3, pp. 50–60, May 2020.

**Appendix**



*Figure 1a. Price distribution of the original dataset(train + test)*

*A screenshot of a computer

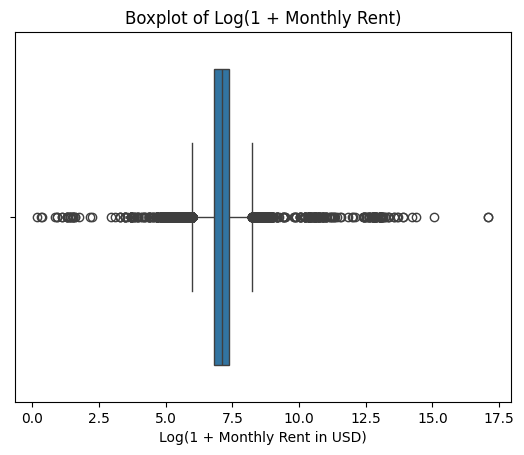
AI-generated content may be incorrect.*

*Figure 1b. Descriptive statistics of price in the original dataset(train + test)*

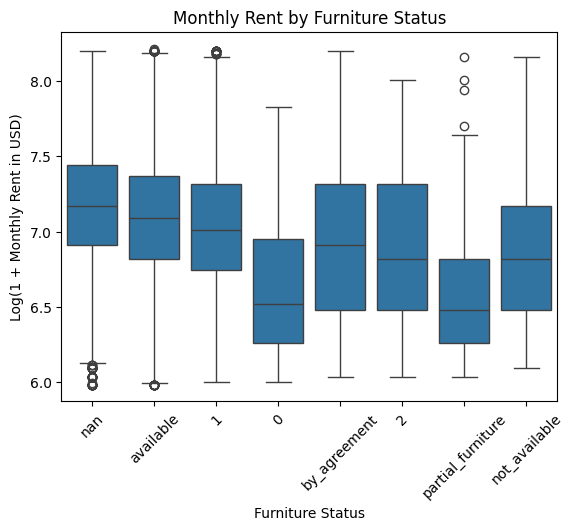
*A screenshot of a computer

AI-generated content may be incorrect.*

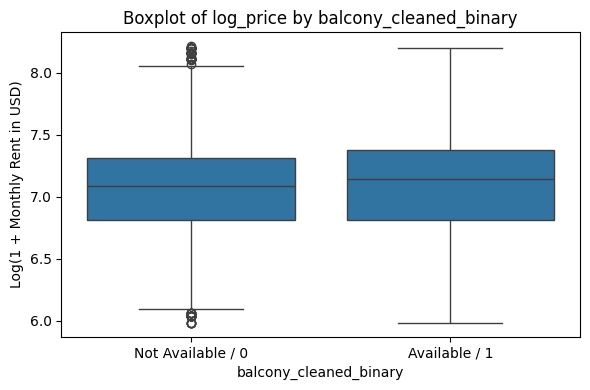
*Figure 1c. Descriptive statistics of price in the trimmed dataset (train + test)*



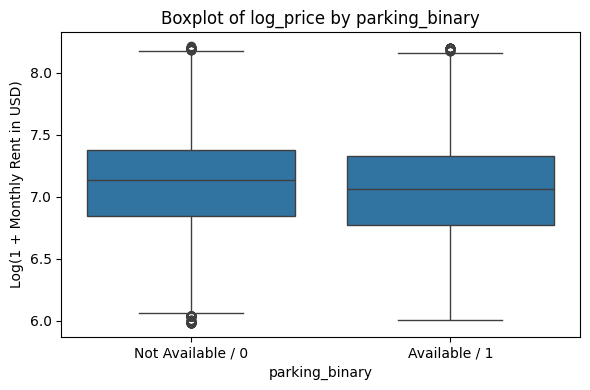
*Figure 1d. Boxplot of the original dataset(train + test)*



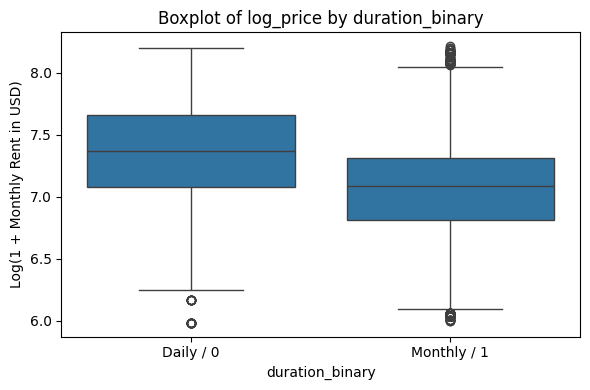
*Figure2. Boxplot of the furniture (train + test)*



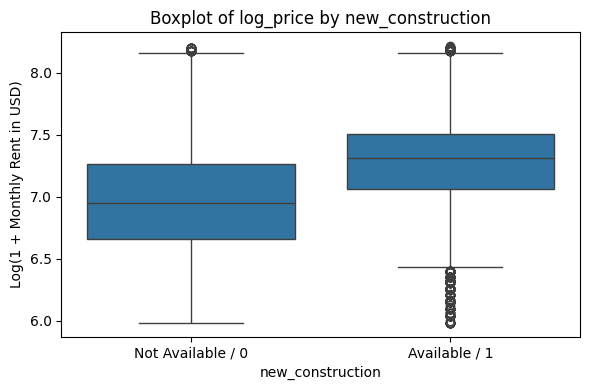
*Figure2a. Boxplot of the balcony\_binary (train + test)*



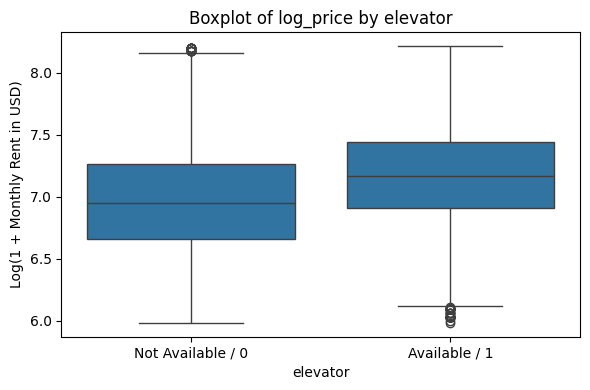
*Figure2b. Boxplot of the parking binary (train + test)*



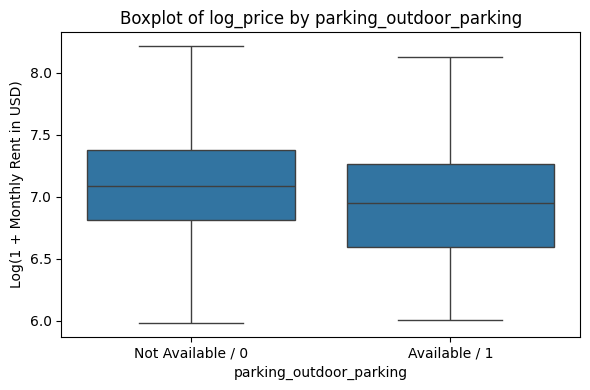
*Figure2c. Boxplot of the parking binary (train + test)*



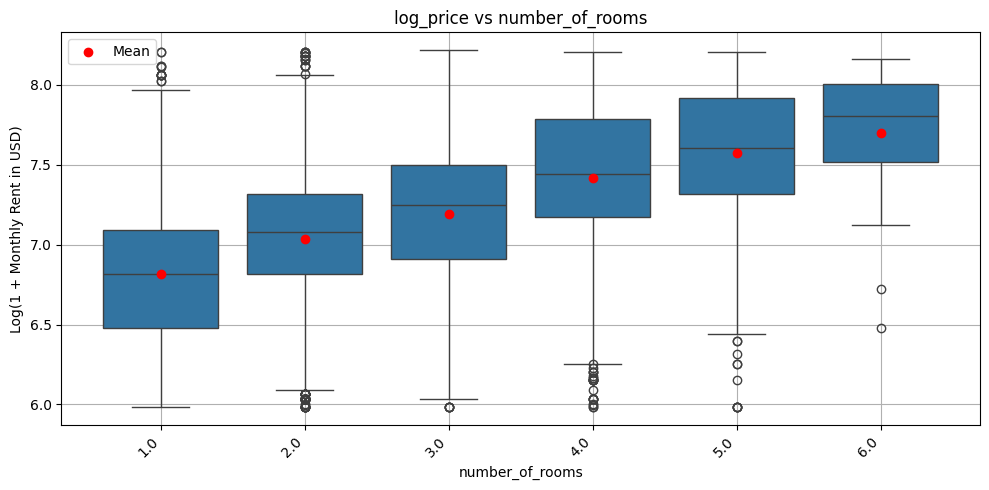
*Figure2d . Boxplot of the new\_construction binary (train + test)*



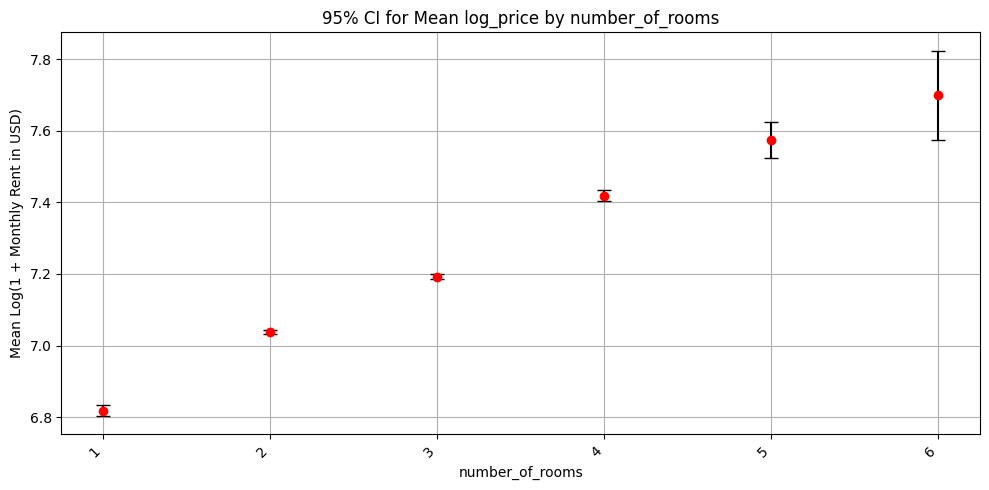
*Figure2e . Boxplot of the new\_construction binary (train + test)*



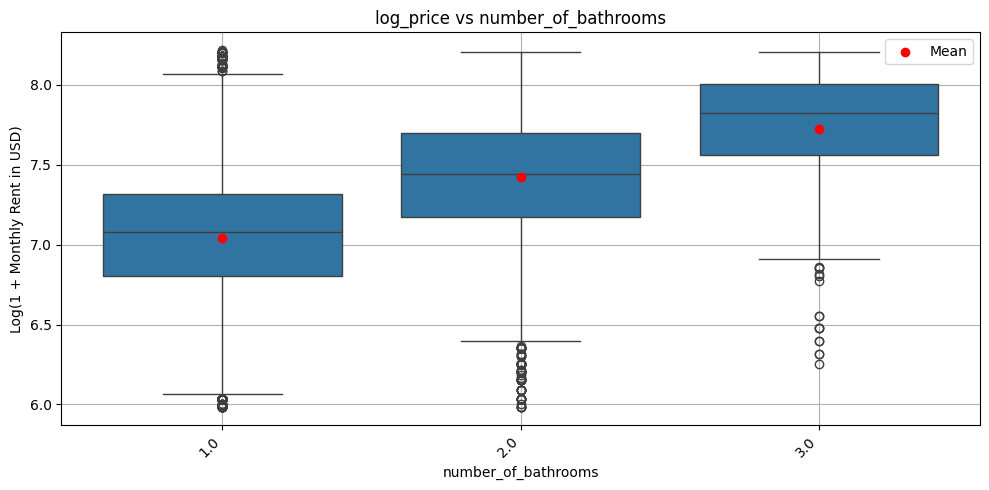
*Figure2f . Boxplot of the new\_construction binary (train + test)*



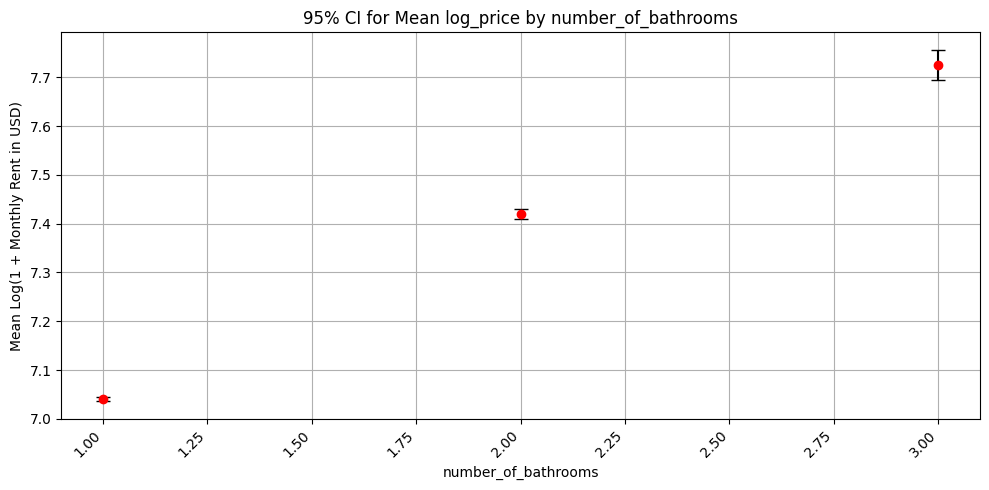
*Figure3a . Boxplot of the number\_of\_rooms (train + test)*



*Figure 3b . 95% CI for log\_price by number\_of\_rooms*



*Figure3c . Boxplot of the number\_of\_bathrooms (train + test)*



*Figure 3d . 95% CI for log\_price by number\_of bathrooms*

A screenshot of a computer

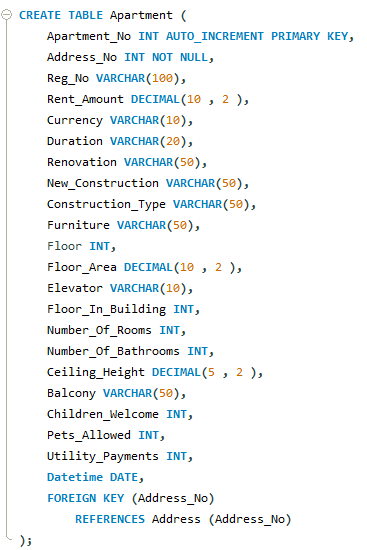
AI-generated content may be incorrect.

*Table 1. t-test result for binary variables.*

A screenshot of a computer code

AI-generated content may be incorrect.

*Figure 4a SQL for create table Address*



*Figure 4b SQL for create tables Apartment*

A screenshot of a computer program

AI-generated content may be incorrect.

*Figure 4c SQLs for create table Amenity, Appliance, Parking*

A screenshot of a computer program

AI-generated content may be incorrect.

*Figure 4d. SQLs for create junction tables Amenity, Appliance, Parking*