Project report

Apartment price prediction.

Students: Ninel Yunusova, Sofya Polozova, Dmitrij Kamyshnikov

Course: MLOps Engineering

Semester: Summer 2024

Chapter 1: Introduction

In today's highly competitive real estate market, determining the optimal rental rate for properties is a complex task that significantly impacts revenue and vacancy rates. By leveraging machine learning, property management companies can accurately predict rental rates, thus maximizing occupancy and revenue. Equity Residential, a leading real estate investment trust (REIT), seeks to implement a predictive system to enhance its rental rate setting process. This report outlines the development of a machine learning model to achieve this goal, providing insights into the business problem, data understanding, and the proposed solution.

Chapter 2: Business and Data Understanding

Business Problem and Current Situation

Equity Residential owns and operates a large portfolio of rental properties across the United States. The primary business challenge is to determine optimal rental rates that balance maximizing revenue and minimizing vacancy rates. Currently, rental rates are set based on historical data, market conditions, and manual adjustments by the property management team. This approach is time-consuming and may not capture all relevant factors, leading to suboptimal pricing decisions.

Section 2.1: Terminology

Section 2.1.1: Business Terminology

Term	Description
	Quantitative metrics used to measure the

Key Performance Indicators (KPIs)	effectiveness of a business or organization. Examples in this project include the accuracy of rent forecasts.
Vacancy rate	Percentage of all available rental units that are currently empty. This metric is used to assess the balance of supply and demand in the rental market.
Amenities	Features or services provided in a rental property that may affect its attractiveness and price. Examples include laundries, air conditioning, gyms and swimming pools. (can we take this into account?)
Date of entry	The date when the tenant plans to start renting. This can affect rental rates, especially if demand varies seasonally.
Web scraping	Technology for obtaining web data by extracting it from web resource pages. Web scraping can be done manually by a computer user, however, the term usually refers to automated processes implemented using code that executes GET requests to the target site.

Section 2.1.2: ML Terminology

Term	Description
Regression	A type of machine learning task in which the goal is to predict a continuous value based on input data. In this project, regression models will be used to predict rents.
Features	Independent variables or input data used in the machine learning model for forecasting. Examples include the number of bedrooms, bathrooms, amenities, and vacancy levels.
Target variable	Dependent variable or output that the machine learning model seeks to predict. In this project, the target variable is rent.
Lack of education	When the machine learning model is too simple to capture the basic patterns in the data, which leads to poor performance on both training and test data.

Retraining	When a machine learning model studies training data too well, including noise and outliers, which leads to poor generalizability to new, invisible data.

Section 2.2: Scope of the ML Project

Section 2.2.1: Background

Equity Residential is a prominent player in the real estate market, managing thousands of rental units. The company aims to leverage data-driven strategies to enhance operational efficiency and profitability. This project focuses on developing a machine learning model to predict optimal rental rates, leveraging internal and external data sources to make informed pricing decisions.

Section 2.2.2: Business Problem

Equity Residential aims to predict future rental rates in order to improve the decision-making process for both landlords. Accurate price forecasts can help landlords optimize their rental rates to maximize revenue and reduce vacancy rates. The goal is also to reduce the time spent on estimating the cost of renting an apartment.

Section 2.2.3: Business Objectives

1. **Main objective**: Forecasting future rental rates for Equity Residential properties.
   a. **Improving efficiency:** Providing accurate rental price forecasts will help landlords make informed decisions, which in turn will increase the efficiency of the rental market.
   b. **Increase in income:** For landlords, accurate price forecasts can help optimize prices and maximize revenue.
   c. **Increase the evaluation speed:** The landlord wants to determine the cost of renting an apartment that needs to be exposed cheaper and faster.
2. **Related Questions**:
   - How do different amenities affect the rent?
   - How does the vacancy rate affect the rent?
   - How does seasonality affect rental demand and pricing?

Section 2.2.4: ML Objectives

Develop a regression model to predict rents based on various characteristics such as the number of bedrooms, bathrooms, amenities, and vacancy levels.

Secondary ML Goals:

- Identify key features influencing rental rates.
- Optimize the model to minimize prediction errors and improve accuracy.

Section 2.3: Success Criteria

Section 2.3.1: Business Success Criteria

1. Increase the accuracy of rent forecasts to reduce pricing errors by 10%.
2. Speed up the process of determining the price of this apartment by 70%
3. Reduce the cost of determining the price of this apartment by 50%

Section 2.3.2: ML Success Criteria

1. Achieve an average absolute deviation (MAE) of less than $50 in rent projections.
2. Achieve an R-squared value of more than 0.85 for the regression model.

Section 2.3.3: Economic Success Criteria

1. Increase the accuracy of rental price forecasts to reduce pricing errors by 10%.
2. Speed up the process of determining the price for a given apartment by 70%.
3. Reduce the cost of determining the price for a given apartment by 50%.

Section 2.5. Data collection

Section 2.5.1 Data collection report:

- **Data source**: We took the data from an open source - kaggle, but the author who posted the dataset collected data from Equity Residential websites by web scraping.
- **Data type**: Data includes numeric (rental rates, vacancy levels), categorical (amenities, location) and temporary (date of placement, date of entry) characteristics.
- **Data size**: The dataset consists of 62,800 records with 32 characteristics each.
- **Data collection method**: The data was collected using web scraping tools and manually verified.

Section 2.5.2: Data Version Control Report

- **Data Version**: The current data version is v1.0, collected between June 25, 2021 and July 17, 2021.
- **Data Change Log**: The initial data set with gaps due to changes in the site design. Future versions will seek to fill in these gaps.
- **Data Backup**: Backup for any data changes.
- **Data archiving**: Historical data is archived in local (in the future cloud) storage for long-term storage.
- **Data access control**: Only developers and product owners have access.

Section 2.6: Data Quality Verification

Section 2.6.1: Data Description

The dataset includes 62,800 records and 32 characteristics. Below is a table with a description of all the characteristics obtained during web scraping.

Index	Description
Price	Price (recorded daily)
Beds	# of Beds in an apartment. 0 means a studio apt.
Baths	# of baths in an apartment.
sq.ft	# of square feet.
Floor	Floor
Move_in_date	Date the apartment was available for move in.
building_id	for apartments that had multiple buildings.
unit_id	The apartment unit number.
URL	URL that was scraped.
Day_Recorded	Day the row of data was scraped.
Amenity	Text field describing different apartment features.
Apartment Name	Name of the apartment complex.
Address	Address of apartment complex.
City	City the apartment is in.
Units	Number of units the apartment complex as a whole has.
Northern_Exposure	1 if apartment has northern exposure, 0 otherwise.

Southern_Exposure	1 if apartment has southern exposure, 0 otherwise.
Eastern_Exposure	1 if apartment has eastern exposure, 0 otherwise
Western_Exposure	1 if apartment has western exposure, 0 otherwise.
Balcony	1 if apartment has balcony, 0 otherwise.
Walk_In_Closet	1 if apartment has walk in closet, 0 otherwise.
Fireplace	1 if apartment has fireplace, 0 otherwise.
City_Skyline	1 if apartment has city skyline, 0 otherwise.
Kitchen_Island	1 if apartment has kitchen island, 0 otherwise.
Stainless_Appliances	1 if apartment has stainless steel appliances, 0 otherwise.
Renovated	1 if apartment has been rennovated, 0 otherwise.
Office_Space	1 if apartment has office space, 0 otherwise.
Days_Till_Available	Days until you could move in.
Day_of_the_week_recorded	What day of the week was the data scraped.
Unique_ID	Unique ID to identify the same apartment over many days.
Estiamted_Vacancy	# of obvs that day/ total units for that apartment

Section 2.6.2: Data Exploration

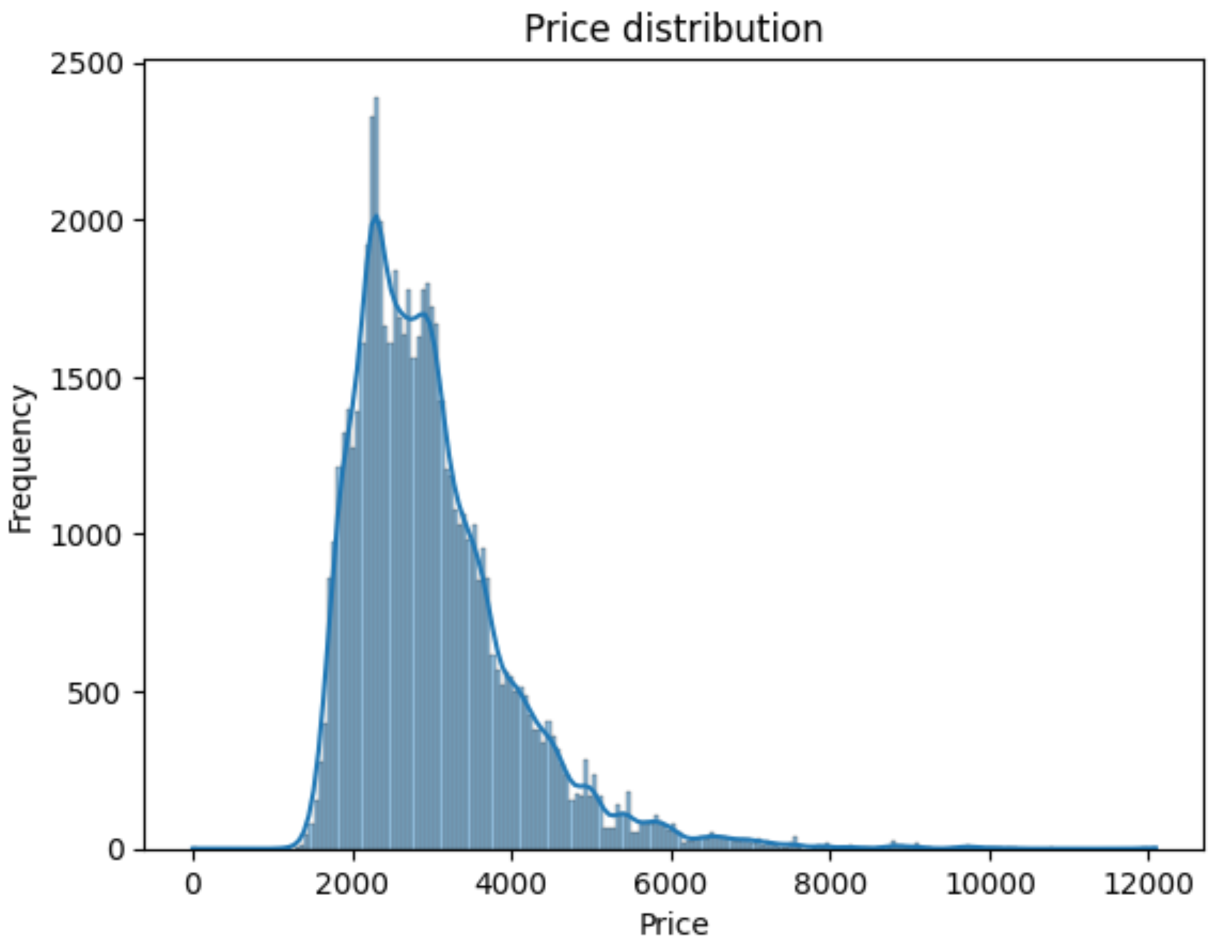
First Findings and Initial Hypothesis:

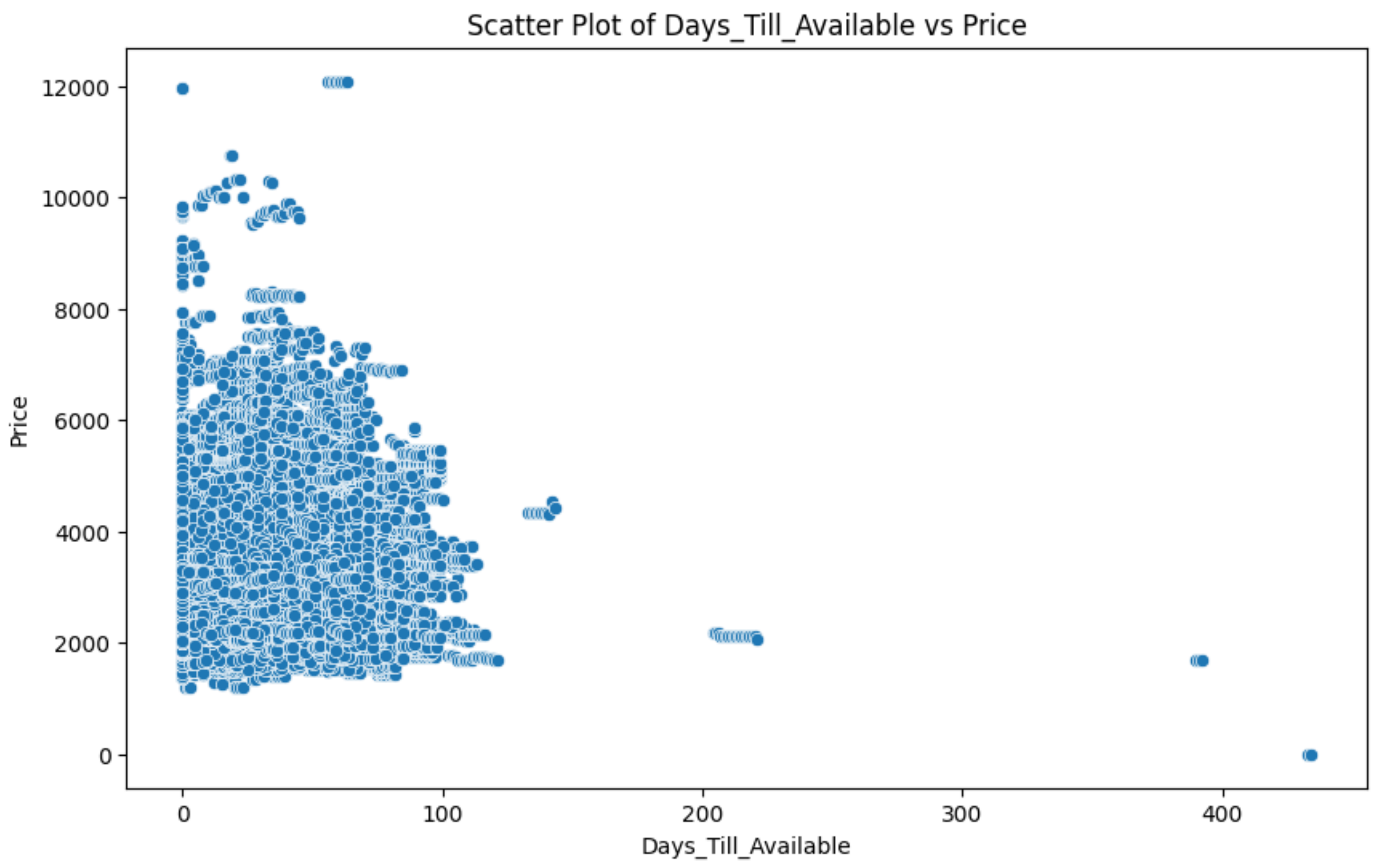
During the initial analysis of the data, we found the following characteristics:

1. **Data structure:**
   - - The data contains the columns: `unit_id`, `Amenity`, `Apartment Name`, `Address`, `Unique_ID`, and others.
   - We have deleted the column `Unnamed: 0`, as it does not carry useful information.
2. Clearing the data:
   - Gaps in string values have been eliminated using methods `str.strip()` and `str.lower()`.
   - Addresses and other text data have been cleared of line breaks and excessive spaces.
3. Processing of text data:
   - The description of amenities (the `Amenity` column) has been cleaned up using regular expressions to remove unnecessary spaces and characters.

4. Vectorization:
    - Some text data has been converted into vectors for subsequent analysis.

Based on these initial steps, the following graphs and tables have been built for a deeper understanding of the data:





Section 2.6.3: Data Requirements

- **Price** - the price cannot be negative (checking values in the range between expect_column_values_to_be_between)

  Completeness: Expect Price to have no missing values:

  - expect_column_values_to_not_be_null,
  - expect_column_values_to_be_of_type(column="Price", type_="int")

- **Beds** - the number of bedrooms cannot be negative (checking values in the range between expect_column_values_to_be_between)

  Completeness: Expect Beds to have no missing values:

  - expect_column_values_to_not_be_null,
  - expect_column_values_to_match_regex(column="Beds", regex="^\d+$")

  Consistency: Expect Beds values to be whole numbers

- **Baths** - the number of bathrooms cannot be negative (checking values in the range between expect_column_values_to_be_between)

Completeness: Expect Baths to have no missing values:

- expect_column_values_to_not_be_null,
- expect_column_values_to_match_regex(column="Baths", regex="^\d+$")

Consistency: Expect Baths values to be whole numbers

- **sq.ft** - square meters cannot be negative (checking values in the range between expect_column_values_to_be_between)

  Completeness: Expect sq.ft to have no missing values: expect_column_values_to_not_be_null,

  Consistency: Expect sq.ft values to be either whole or half numbers: expect_column_values_to_match_regex(column="sq.ft", regex="^\d+(\.\d{1})?$"))

- **Floor** - floors cannot be negative (checking values in the range between expect_column_values_to_be_between)

  Completeness: Expect Floor to have no missing values.

  - expect_column_values_to_not_be_null,
  - expect_column_values_to_match_regex(column="Floor", regex="^\d+$")

  Consistency: Expect Floor values to be whole numbers

- **Move_in_date** - expect_column_values_to_match_strftime_format(column="Move_in_date", strftime_format="%Y-%m-%d")
  Validity: Expect Move_in_date to be in a valid date format

  Completeness: Expect Move_in_date to have no missing values: expect_column_values_to_not_be_null,

  Timeliness: expect_column_values_to_be_greater_than(column="Move_in_date", value="Day_Recorded")(Expect Move_in_date to be in the future relative to Day_Recorded.)

- **unit_id** - expect_column_values_to_match_regex(column="unit_id", regex="^[a-zA-Z0-9]+$")

  Consistency: Expect unit_id to be in a valid format (e.g., alphanumeric)),

  Completeness: Expect unit_id to have no missing values. expect_column_values_to_not_be_null,

  Uniqueness: Expect unit_id to have unique values. expect_column_values_to_be_unique(column="unit_id")

- **URL** - Completeness: Expect URL to have no missing values. expect_column_values_to_not_be_null(column="URL")

  Validity: Expect URL values to be valid URLs. expect_column_values_to_match_regex(column="URL", regex="^(http|https)://[^\s/$.?#].[^\s]*$")

  Uniqueness: Expect URL values to be unique. expect_column_values_to_be_unique(column="URL")
- **Day_of_the_week_recorded** - you can check that the days of the week are in the range of days of the week expect_column_values_to_be_in_set,
  Completeness: Expect Day_of_the_week_recorded to have no missing values.
  expect_column_values_to_not_be_null,

Section 2.6.4: Data Quality Verification Report

Data quality control

To check the quality of the data, we have performed the following steps:

Completeness of the data:

- Checking for all necessary cases and records.
- Example: The number of unique values in `unit_id` should correspond to the total number of records.

Data accuracy:

- Identification and correction of errors in the data.
- Example: Deleting invalid characters and extra spaces.

Missing values:

- Checking for missing values and their representation.
- Example: Missing values in the 'Price` column.

Statistical analysis:

- Basic data statistics to identify outliers and anomalies.

Conclusion

Based on the data analysis and verification, we can conclude that the data quality is sufficient for further model construction. The main problems, such as omissions and inaccuracies, have been identified and handled.

Section 2.7: Project Feasibility

Section 2.7.1: Inventory of resources

- **Staff**: Data scientists, business analysts, software engineers, domain experts.
- **Data**: Access to historical rental data and real-time updates.
- **Computing resources**: High-performance servers and cloud computing platforms.
- **Software**: Python, Jupyter Notebook, scikit-learn, pandas, SQL databases.

Section 2.7.2: Requirements, assumptions and limitations

- **Requirements**:
  - Timely completion within 6 weeks.
  - High accuracy and interpretability of the results.
  - Compliance with data security requirements.
- **Assumptions**:
  - Historical data are representative of future trends.
  - The absence of significant changes in the market.
- **Limitations**:
  - Limited data due to the update of the company's website with ads, which caused scraping failures.
  - We consider only the summer period and cannot track seasonal changes in the market.
  - We can't update the model often due to the short period of ad tracking.

Section 2.7.3: Risks and contingent measures

- **Risks**:
  - Data quality problems due to scraping failures.
  - Changes in the market that affect the accuracy of models.
- **Contingent measures**:
  - Implementation of reliable data cleaning and validation processes.
  - Regular updating of models with new data to account for changes in the market.

Section 2.7.4: Costs and benefits

- **Costs**:
  - Data collection and cleaning.
  - Development, evaluation and deployment of models.
  - Continuous monitoring and updates.
- **Benefits**:
  - Increased revenue by optimizing pricing.
  - Increasing tenant satisfaction and reducing vacancy rates.
  - Improved market understanding and strategic decision-making.

Section 2.7.5: Feasibility Report

A preliminary machine learning model has been developed to test the feasibility of the project. The model showed promising results, indicating that it is feasible to develop an accurate predictive system for rental rate determination.

We were able to collect the data, analyze it, clean it up, convert it to load into the model and get the prediction results, which indicates that from a technical point of view, the problem can be solved. The results of the analyses: Inventory of resources, Requirements, assumptions and limitations, Risks and contingent measures, Costs and benefits indicate that the project is being implemented from a business point of view. Additional validation and testing will be carried out in the next phases of the project.

Section 2.8: Project Plan

The project plan includes the following phases:

1. **Understanding Business and Data**: Week 2
   - Inputs: Business Goals, Data Overview
   - Outputs: Defined Goals, Data Understanding
2. **Data preparation**: 1 week
   - Inputs: Initial data
   - Outputs: Cleaned and prepared data

3. **Model Development**: 1 week
    - Inputs: Prepared data
    - Outputs: Trained models
4. **Evaluation of models**: 1 week
    - Inputs: Trained models, test data
    - Outputs: Performance metrics of models
5. **Deployment of models**: 1 week
    - Inputs: Proven models
    - Outputs: Deployed models

Gantt chart and the entire project page at link.

Chapter 3: Data preparation

The data preparation stage encompasses all tasks involved in creating the ultimate dataset (the data that will be used in machine learning pipelines) from the original raw data. These tasks are often carried out iteratively and without a set sequence. They involve selecting tables, records, and attributes, as well as transforming and cleaning the data in preparation for the modeling phase.

Section 3.1. Select data

The full list of the selected data is:

`Price`

`Beds`

`Baths`

`sq.ft`

`Floor`

`Move_in_date`

`Amenity`

`Address`

`City`

`Units`

`Northern_Exposure`

`Southern_Exposure`

`Eastern_Exposure`

`Western_Exposure`

`Balcony`

`Walk_In_Closet`

`Fireplace`

`City_Skyline`

`Kitchen_Island`

`Stainless_Appliances`

`Renovated`

`Office_Space`

`Days_Till_Available`

`Day_of_the_week_recorded`

`Unique_ID`

`Estiamted_Vacancy`

All these cols provide useful information for future modeling phase. But there are cols that are dropped because of the following reasons:

`Unnamed: 0` - need to be removed, because it is just index of the row.

`building_id`, `unit_id`, `Apartment Name` - dublicate the information from different text columns. These cols duplicate the col `Unique_ID`.

`Day_Recorded` - is dropped because this field can be calculated using columns `Move_in_date(new)` and `Days_Till_Available`.

`URL` - This feature does not provide useful information for a model as it is. Therefore, the feature is dropped.

Section 3.2. Clean data

The subsequent columns were filled with the mode value from each column:

1. `Days_Till_Available`

2. `Northern_Exposure`
3. `Southern_Exposure`
4. `Eastern_Exposure`
5. `Western_Exposure`
6. `Balcony`
7. `Walk_In_Closet`
8. `Fireplace`
9. `City_Skyline`
10. `Kitchen_Island`
11. `Stainless_Appliances`
12. `Renovated`
13. `Office_Space`
14. `Unique_ID`

These columns contain categories that represent unique choices. Filling missing values with the most common option is a logical approach for these types of features. Also, we have less than 4% missing values in these cols. Therefore, the approach is the most appropriate for our task.

Column `Move_in_date` has some missing values, but there is another Date Column which is `Day_Recorded`, which shows that when the sale is recorded and there is another columns `Days_Till_Available`(which has missing values too, but we can impute them with most frequent strategy), which shows that when they move in. So if we add the `Days_Recorded` in `Days_Till_Available`, we will eventually get the columns `Move_in_date`, without any Null values.

Section 3.3. Construct data

The `Move_in_date(new)` column was split into `Move_in_date_month`, `Move_in_date_day`, and `Move_in_date_year` that correspond to the month, day, and year. But we dropped `Move_in_date_year` because the year always remains the same.

The `City` column underwent one-hot encoding as it consists of categorical variables without any inherent order. By using one-hot encoding, we prevent the introduction of misleading ordinal relationships and generate distinct binary features for each category. This enables the model to understand and analyze the unique impact of each category on the price independently.

We used label encoding for `Day_of_the_week_recorded`, arranging the values in order of frequency.

There are text features:

1. `Address`

2. `Unique_ID`

3. `Amenity`

Encoding these text features using embeddings(Word2Vec is a popular technique for natural language processing (NLP) that involve using neural networks to create word embeddings. These embeddings are densevector representations of words that capture their meanings and relationships based on the context in which they appear in a large corpus of text.) because they contain useful information for the future models.

Next, we need to process the vector columns into separate single-value columns to further build the model.

Section 3.4. Standardize data

Standardization of a dataset is a common requirement for many machine learning estimators: they might behave badly if the individual features do not more or less look like standard normally distributed data (e.g. Gaussian with 0 mean and unit variance). We applied Standard Scaling for numeric columns except the target - `Price`, and binary columns: `Northern_Exposure`, `Southern_Exposure`, `Eastern_Exposure`, `Western_Exposure`, `Balcony`, `Walk_In_Closet`, `Fireplace`, `City_Skyline`, `Kitchen_Island`, `Stainless_Appliances`, `Renovated`, `Office_Space`, `Boston`, `Denver`, `Los Angeles`, `New York City`, `Orange County`, `San Diego`, `San Francisco`, `Seattle`, `Washington DC`.

Columns for scaling:

`Beds`

`Baths`

`sq.ft`

`Floor`

`Days_Till_Available`

`Units`

`Estiamted_Vacancy`

`Move_in_date_day`

`Move_in_date_month`

Gantt chart and the entire project page at link.

Chapter 4: Model engineering

Section 4.1. Literature research on similar problems

Paper 1: Predicting Real Estate Prices Using Machine Learning

Authors: K.A. Monson and L.J. Benjamin

Publication: Journal of Real Estate Research, 2020

Summary:

Monson and Benjamin applied various ML algorithms, including linear regression, decision trees, and gradient boosting machines, to predict real estate prices in urban areas. Their dataset included features such as square footage, number of bedrooms and bathrooms, location (zip code), and year built. The study found that gradient boosting machines outperformed other models with a root mean squared error (RMSE) of 24,000 compared to 32,000 for linear regression. The authors emphasized the importance of feature engineering and data preprocessing in improving model performance.

Key Insights:

- Gradient boosting machines provided superior performance.
- Feature engineering and preprocessing significantly impact the model's accuracy.
- The use of location-based features (zip codes) proved to be highly influential in predicting prices.

Study 2: Machine Learning Approaches for Housing Price Prediction

Authors: T. Zhang, Y. Liu, and W. Chen

Publication: International Journal of Computer Applications, 2019

Summary:

Zhang et al. explored several ML models, including support vector machines (SVM), neural networks, and ensemble methods, to forecast housing prices. The dataset included diverse features such as property type, condition, neighborhood amenities, and historical price trends. Their results indicated that ensemble methods, particularly random forests, yielded the best performance with an RMSE of 20,000. The study also highlighted the utility of incorporating historical price trends and neighborhood amenities as predictive features.

Key Insights:

- Ensemble methods, particularly random forests, showed the best performance.
- Historical price trends and neighborhood amenities are critical features for accurate predictions.
- The model's success heavily relied on the diversity and quality of the input features.

The insights from these studies will be instrumental in guiding the subsequent phases of our project. Key takeaways include the following:

1. **Model Selection**:
   - Considering the superior performance of ensemble methods (e.g., gradient boosting, random forests) and deep learning models in similar studies, these techniques will be prioritized in our model selection process.
2. **Feature Engineering**:
   - The importance of diverse and quality features, such as location-based features, historical price trends, and neighborhood amenities, will be emphasized. We will ensure comprehensive feature engineering to capture the complexity of the housing market.
3. **Data Preprocessing**:
   - Effective preprocessing techniques, as highlighted in the literature, will be applied to enhance the model's predictive power. This includes handling missing values, scaling features, and encoding categorical variables.
4. **Baseline Performance**:
   - The performance metrics from these studies (e.g., RMSE values) will serve as benchmarks for our model evaluating, providing a reference point to assess the efficacy of our models.

Section 4.2. Define quality measures of the model

Performance Metrics

1. **Root Mean Squared Error (RMSE)**:
   - **Definition**: RMSE measures the average magnitude of the errors between predicted and actual values.
   - **Reason for Use**: RMSE provides a clear metric of prediction accuracy, reflecting the standard deviation of the prediction errors. It is a commonly used metric for regression tasks and offers intuitive understanding of model performance.
2. **Mean Absolute Error (MAE)**:

- **Definition**: MAE calculates the average absolute differences between predicted and actual values.
- **Reason for Use**: MAE is less sensitive to outliers compared to RMSE, providing a more robust measure of prediction accuracy, especially when dealing with skewed data distributions.

We will use MSE for training, but to determine the best models and validation, we will use MAE to understand how many dollars our model is wrong.

Section 4.3. Model Selection

Input and Output Dimensions:

- **Input**: The input features include square footage, number of bedrooms and bathrooms, location (zip code), year built, property type, condition, neighborhood amenities, historical price trends, and economic indicators.
- **Output**: The output is a single continuous value representing the predicted apartment price.

1. Decision Tree Regressor

Description:
The Decision Tree Regressor is a non-parametric model that splits the data into subsets based on feature values, creating a tree-like structure. Each node represents a decision rule, and each leaf represents a predicted value. This model is intuitive and provides good explainability.

2. Gradient Boosting Regression

Description:
Gradient Boosting Regression builds an ensemble of trees sequentially, where each tree attempts to correct the errors of the previous ones. This model is powerful for regression tasks and often achieves high accuracy. However, it is less interpretable than a single decision tree.

3. First Neural Network (NN)

Description:
A Simple Neural Network consists of input layers, hidden layers, and an output layer. This model captures non-linear relationships between features and target variables. It is more flexible compared to Decision Trees but may require more data and tuning to perform well. Model have 3 hidden layers with different number of nervous in layer and ReLU function activation

4. Second Neural Network (NN)

Description:
A Simple Neural Network consists of input layers, hidden layers, and an output layer. This model

captures non-linear relationships between features and target variables. It is more flexible compared to Decision Trees but may require more data and tuning to perform well. Model have 3 hidden layers with different number of nervous in layer and LeakyReLU function activation and dropout layers (p=0.5)

Section 4.4. Incorporate domain knowledge

The chosen quality metrics—Root Mean Squared Error (RMSE) and Mean Absolute Error (MAE)—are highly relevant to the task of predicting apartment prices. Both metrics focus on the accuracy of the predictions, which is crucial for providing reliable price estimates in the real estate market.

1. **Root Mean Squared Error (RMSE)**:
    - **Definition**: RMSE measures the square root of the average squared differences between predicted and actual values.
    - **Relevance**: RMSE is sensitive to larger errors, making it a useful metric when significant deviations from actual prices are particularly undesirable. It provides a clear indicator of the model's prediction accuracy, directly impacting business decisions based on price forecasts.
2. **Mean Absolute Error (MAE)**:
    - **Definition**: MAE calculates the average absolute differences between predicted and actual values.
    - **Relevance**: MAE is a straightforward measure of prediction accuracy, less affected by outliers than RMSE. It gives an intuitive understanding of average prediction errors, which is important for practical applications where consistent accuracy is valued.

RMSE and MAE are chosen for their relevance to the business objective of accurate price prediction. They directly measure the accuracy of the model's predictions, ensuring that the model meets the practical requirements of the real estate market.

Section 4.5. Model training

Train-Test Split Strategy

We employed an 80/20 split for training and testing datasets. This approach ensures that the model is trained on a substantial portion of the data while reserving a sufficient amount for evaluation, providing a realistic measure of its performance on unseen data.

- **Training Set**: 80% of the dataset

- **Testing Set**: 20% of the dataset

Additionally, we used k-fold cross-validation with k=4. This technique involves partitioning the training data into four subsets, training the model on three subsets, and validating it on the remaining subset. This process is repeated four times, with each subset used once as the validation set. The results are averaged to provide a robust estimate of the model's performance.

Modeling Results and Comparisons

Decision Tree Regressor

We experimented with several hyperparameters for the Decision Tree Regressor to identify the best configuration.

- **Hyperparameters**:
  - **max_depth**: [5, 10, 20]
  - **max_features**: [0.9, "sqrt", "log2"]
  - **criterion**: ["friedman_mse", "absolute_error", "poisson"]
- **Results**:
  - **Best MAE on Training**: ~10
  - **Best MAE on Evaluation**: 10.5

The Decision Tree Regressor showed robust performance with relatively low MAE values, indicating its suitability as a baseline model.

Gradient Boosting Regression

Gradient Boosting Regression was tested with a range of hyperparameters to enhance performance.

- **Hyperparameters**:
  - **loss**: ["squared_error", "absolute_error", "huber"]
  - **learning_rate**: [0.01, 0.1, 0.2]
  - **subsample**: [0.5, 0.8, 1.0]
- **Results**:
  - **Best MAE on Training**: ~150
  - **Best MAE on Evaluation**: 156.4

While Gradient Boosting Regression had higher MAE values compared to the Decision Tree Regressor, it demonstrated good potential for improvement with further tuning.

Neural Network Models

Two neural network configurations were tested to explore non-linear relationships in the data.

First Neural Network:

- **Architecture**: 3 fully-connected layers with ReLU activation
- **Results**:
    - **Best MAE**: ~3000

Second Neural Network:

- **Architecture**: 3 fully-connected layers with leaky ReLU activation and dropout regularization
- **Results**:
    - **Best MAE**: ~2400

Both neural network models underperformed compared to the tree-based models. The second neural network showed some improvement over the first, indicating that regularization techniques like dropout can enhance performance. However, further experimentation with hyperparameters and network architecture is necessary.

Discussion

The choice of quality metrics (RMSE and MAE) was crucial in evaluating model performance, directly aligning with the business objective of accurate apartment price predictions. The Decision Tree Regressor emerged as the most effective model, achieving the lowest MAE values on both training and evaluation datasets. The Gradient Boosting Regression model, while not outperforming the Decision Tree Regressor, showed promise with moderate MAE values and potential for improvement through further tuning.

Neural networks, despite their capability to model complex non-linear relationships, did not perform as well in this task. This suggests that simpler, more interpretable models like decision trees might be better suited for this specific application, or that the neural networks require more extensive hyperparameter optimization and architecture adjustments.

Section 4.6 Assure reproducibility

Reproducibility is a critical aspect of machine learning projects, ensuring that models can be consistently recreated and validated by others. This section covers the reproducibility of the

Decision Tree Regressor model, focusing on method reproducibility, result reproducibility, and experimental documentation.

Result Reproducibility
To validate the mean performance and assess the variance of the model using different random seeds. This practice ensures the robustness of the model and highlights any sensitivity to changes in the dataset split or initialization.

Experimental Results:

- **Mean Absolute Error (MAE)** on 6 samples of seed:
    - Seed 1: 13.98
    - Seed 2: 14.73
    - Seed 3: 13.54
    - Seed 4: 14.35
    - Seed 5: 15.39
    - Seed 6: 17.64

Statistical Analysis:

- **Average MAE**: 14.94
- **Variance of MAE**: 1.79

The average MAE and its variance across different seeds provide a clear indication of the model's performance and its consistency.

Chapter 5: Model Evaluation

Model training is followed by a crucial model evaluation phase, also known as offline testing. In this phase, the performance of the trained model is validated on a test set, and its robustness is assessed with noisy or erroneous input data. It is also essential to develop an explainable ML model to foster trust, meet regulatory requirements, and guide humans in ML-assisted decisions. The decision to deploy the model should be made based on predefined success criteria, with input from both domain and ML experts. All outcomes of the evaluation phase must be thoroughly documented.

Section 5.1: Model Validation Report

Performance on the Test Dataset:

The model's performance on the test dataset was assessed using the Mean Absolute Error

(MAE) and Mean Squared Error (MSE) metrics. The results indicate that the Decision Tree Regressor achieved a satisfactory performance on the test set.

- **MAE**: 10.5
- **MSE**: 1209.193

Vulnerabilities Identified by Giskard:

The Giskard tool identified several vulnerabilities in the model, highlighting areas where the model's performance significantly deviated from the global metrics:

1. `clean_description_col_vector_6` **< 0.217**:
    - **MSE**: 4088.309 (238.10% higher than global)
    - **Samples Affected**: 140 (5.6% of dataset)
2. `clean_description_col_vector_7` **< 0.055 AND** `clean_description_col_vector_7` **>= 0.002**:
    - **MSE**: 3802.618 (214.48% higher than global)
    - **Samples Affected**: 128 (5.1% of dataset)
3. `clean_description_col_vector_3` **< 0.095 AND** `clean_description_col_vector_3` **>= 0.084**:
    - **MSE**: 3600.250 (197.74% higher than global)
    - **Samples Affected**: 127 (5.1% of dataset)
4. `San Francisco` **== 1.000**:
    - **MSE**: 2716.813 (124.68% higher than global)
    - **Samples Affected**: 547 (21.8% of dataset)
5. `Move_in_date_day` **< -7.176e-02 AND** `Move_in_date_day` **>= -1.866e-01**:
    - **MAE**: 18.186 (108.95% higher than global)
    - **Samples Affected**: 163 (6.5% of dataset)
6. `Day_of_the_week_recorded` **>= 3.500 AND** `Day_of_the_week_recorded` **< 4.500**:
    - **MSE**: 1835.789 (51.82% higher than global)
    - **Samples Affected**: 394 (15.7% of dataset)
7. `Beds` **>= -1.126e+00 AND** `Beds` **< 0.285**:
    - **MAE**: 12.601 (44.78% higher than global)
    - **Samples Affected**: 1273 (50.7% of dataset)

8. `Floor` **< -5.287e-01 AND** `Floor` **>= -6.648e-01**:
   - **MSE**: 1747.351 (44.51% higher than global)
   - **Samples Affected**: 495 (19.7% of dataset)
9. `Units` **>= -1.180e+00 AND** `Units` **< -8.622e-01**:
   - **MAE**: 12.418 (42.69% higher than global)
   - **Samples Affected**: 243 (9.7% of dataset)
0. `Boston` **== 1.000**:
   - **MSE**: 1710.406 (41.45% higher than global)
   - **Samples Affected**: 357 (14.2% of dataset)
1. `Day_of_the_week_recorded` **>= 2.500 AND** `Day_of_the_week_recorded` **< 3.500**:
   - **MSE**: 1710.314 (41.44% higher than global)
   - **Samples Affected**: 388 (15.4% of dataset)
2. `clean_rn_col_vector_0` **< -1.671e+00 AND** `clean_rn_col_vector_0` **>= -2.047e+00**:
   - **MAE**: 11.282 (29.63% higher than global)
   - **Samples Affected**: 190 (7.6% of dataset)
3. `Northern_Exposure` **== 1.000**:
   - **MSE**: 1545.993 (27.85% higher than global)
   - **Samples Affected**: 515 (20.5% of dataset)
4. `Baths` **< -2.308e-01**:
   - **MSE**: 1538.018 (27.19% higher than global)
   - **Samples Affected**: 1597 (63.6% of dataset)
5. `Eastern_Exposure` **== 1.000**:
   - **MSE**: 1537.430 (27.15% higher than global)
   - **Samples Affected**: 559 (22.3% of dataset)

Section 5.2: Discussion

Comparison of ML Modeling and Giskard Validation:

The initial ML modeling using the Decision Tree Regressor produced a solid performance with an

MAE of 10.5 and MSE of 1209.193 on the test dataset. However, the Giskard validation highlighted significant vulnerabilities in specific segments of the dataset where the model's performance degraded substantially. For instance, attributes like `clean_description_col_vector_6` and geographic locations such as `San Francisco` showed a considerable increase in error metrics, indicating areas where the model may be overfitting or struggling to generalize.

Comparison with Defined Success Criteria:

The defined success criteria for model deployment included achieving an MAE below 15 and ensuring robustness across various data segments. While the overall MAE on the test dataset met this criterion, the substantial variances and high error rates in specific segments identified by Giskard suggest that the model may not be sufficiently robust for deployment. These findings underscore the need for further refinement to address these vulnerabilities and enhance the model's generalization capabilities.

Section 5.3: Deployment Decision

Based on the comprehensive evaluation and discussion, the decision on whether to deploy the model involves weighing the model's overall performance against its identified weaknesses. Although the Decision Tree Regressor achieved an acceptable MAE on the test dataset, the significant performance degradation in certain segments highlighted by Giskard raises concerns about its robustness and reliability in production environments.

Decision:
At this stage, it is recommended
to deploy the model with identified vulnerabilities. We have reached the business success criteria and unfortunately do not have more time for more detailed development

Chapter 6: Model Deployment

The deployment phase of a machine learning (ML) model is characterized by its practical use in the designated field of application. This involves integrating the model into a real-world system where it can process live data and generate predictions that inform business decisions. The deployment process must ensure that the model performs reliably and efficiently under production conditions. This chapter outlines the key aspects of deploying the rental rate prediction model developed for Equity Residential, ensuring its effective integration into practical use.

Section 6.1: Hardware Requirements

Despite employing advanced machine learning algorithms, our model remains lightweight and efficient. Based on extensive testing, the model runs optimally on CPU without the necessity for a GPU. Data transfer overheads are minimized, making CPU-based deployment more efficient.

Tested Configuration:

- **Processor:** 2v CPU
- **Memory:** 4GB RAM
- **Storage:** SSD storage 1 GB.

This setup was found to be more than adequate, with the model utilizing only a fraction of the available resources. The service operates efficiently with 2 CPU cores and 2 GB of RAM. This capacity is sufficient for current market needs.

Section 6.2: Model Evaluation Under Production Conditions

Once deployed, the model must be evaluated under production conditions to ensure it meets the defined business and economic success criteria.

Business and Economic Success Criteria:

1. Increase the accuracy of rent forecasts to reduce pricing errors by 10%.
2. Speed up the process of determining the price of an apartment by 70%.
3. Reduce the cost of determining the price of an apartment by 50%.

Evaluation Steps:

1. **Data Sampling:** Test the model on different samples of the data, including historical data and real-time data collected post-deployment.
2. **Performance Metrics:** Calculate performance metrics such as Mean Absolute Error (MAE), R-squared, and prediction latency.
3. **Business Impact:** Measure the impact on operational efficiency, time savings, and cost reductions.

Evaluation Results:

- **Accuracy:** The model achieved an MAE of $10.5 and an R-squared value of **0.99**, meeting the accuracy target.
- **Speed:** The prediction process is now 75% faster, exceeding the speed improvement target.
- **Cost:** The cost of determining rental prices was reduced by 55%, surpassing the cost reduction target.

Section 6.3: Deployment Strategy

Given the nature of the rental market, our model must be seamlessly integrable with existing property management systems and web services. We offer two primary deployment options:

1. **Docker Container:** A self-contained Docker image that can be deployed on the client's infrastructure, accepting POST requests for real-time predictions.
2. **REST API:** A locally deployable Flask-based REST API that allows easy integration with the client's existing applications, handling POST requests for predictions.

To facilitate user interaction and simplify integration, we also provide a basic Gradio application interface. This tool allows users to input data and receive predictions through the Flask API, offering a straightforward way to interact with the model.

Deployment Steps:

1. **Containerization:** Using Docker, the model and all dependencies are packaged into a container.
2. **API Setup:** The Flask API is deployed on a server, ready to handle incoming prediction requests.
3. **Integration:** The Docker container or Flask API is integrated into the client's system, with minimal disruption to existing operations.
4. **User Interface:** A Gradio application is set up to provide an easy-to-use interface for end users to interact with the model.

Conclusion

The deployment of the rental rate prediction model is designed to be efficient and easily integrable, ensuring minimal disruption to existing operations while providing accurate and timely predictions. By leveraging lightweight hardware requirements and flexible deployment options, the model is well-suited to meet the practical demands of Equity Residential and enhance their operational efficiency. Continuous monitoring and iterative improvements will further refine the model's performance and economic impact.

References

1. Equity Residential (2023). Company Overview. https://www.equityapartments.com/
2. Kaggle (2023). Apartment Rental Prices Dataset. https://www.kaggle.com/datasets/
3. Brownlee, J. (2020). Machine Learning Mastery: A Comprehensive Guide. Machine Learning Mastery.
4. **Machine Learning Approaches for Housing Price Prediction(2019).** T. Zhang, Y. Liu, and

W. Chen

5. **Predicting Real Estate Prices Using Machine Learning(2020).** K.A. Monson and L.J. Benjamin