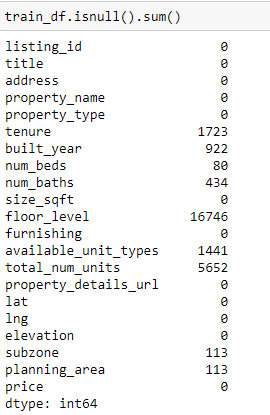
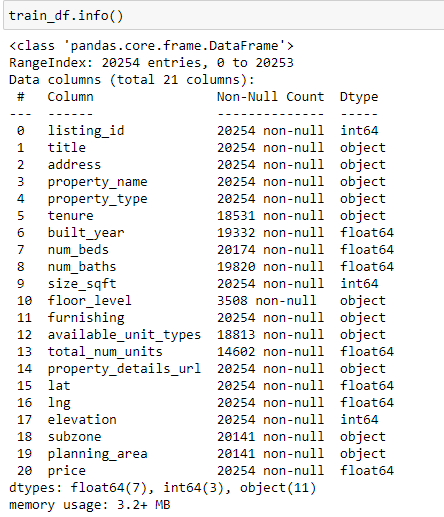
## Motivation

Our primary responsibilities for task 1 are those related to data preprocessing, such as handling missing data, categorical data transformation, and feature extraction. In the meantime, we concentrate on feature extraction (data mining) on the auxiliary dataset to gather additional data that could enhance the model's prediction performance. A few of our data preprocessing strategies will assume that we will use tree-base models for this task. In task 2, we intend to run recommendation models and evaluate the outcomes of various solutions using the previously extracted and transformed variables from task 1.



## Exploratory Data Analysis & Preprocessing

### Price

Figure X.1 shows that there are two unique price records that are outside of the price distribution. For building the model, we were concerned that the results of these 2 samples would be skewed, and we assumed that such irrational property prices were unlikely to be forecast for actual use. So, we made the decision to remove these two outstanding records. The price distribution of the reset records is presented in Figure X.2. Graphical user interface, application, Teams

Description automatically generated

### Title

The 'title' attribute always contains the following details: “{n bed} {property type} for sale in {location}”. Even though this information may overlap with those of other variables or be irrelevant for modeling, they can still be used to impute missing values or conduct sanity check for crucial attributes like the number of beds and property type. Figure x compares the features taken from the title to the current fields.

Graphical user interface, application

Description automatically generated

### Address

We simply discard this attribute from modeling in order to address the overfitting problem.

### Latitude & Longitude

We primarily concentrated on the incorrect records for latitude and longitude features. Given that the goal is to forecast Singapore house property prices, the latitude and longitude ranges should be close by (1.290270, 103.851959). However, a few records are located outside the Singapore region, and further research has revealed that those records (by title, address) belong to Singapore. Since there aren't many incorrect records (figure x.1), we decide to use Google to find the correct location (figure x.2) and manually amend the entries. The results before and after the correction are shown in figures x.3 and x.4.Graphical user interface, application

Description automatically generated

### Property name

For the same reason as the Address field handling, we removed this feature.

### Property type

Transforming this property into ordinal categorical data with potential ranks is the goal of handling it. Prior to the change, we must address the issues listed below that existed in this column.

1. Uniform lower-case letters
2. Different types of HDB types/description: “hdb”, “hdb {n} rooms”, “hdb Executive”

A picture containing scatter chart

Description automatically generated

Regarding the second issue, we chose to remove the extra "n rooms" information from the property attribute because it is already included in the num beds and num baths information. Figure x.1 shows fewer noise property types as compared to Figure X.

Before converting them to a numerical representation, we decided to rank each category by its average price in consideration of improved separation and understanding by the tree-base models (figure x.2).Graphical user interface, chart, application

Description automatically generated

### Tenure

Three main activities must be completed in order to preprocess this feature: cleaning up noisy records, handling of missing data, and categorical data conversion.

Graphical user interface, application

Description automatically generated

There are a few unrecognized tenure types in these categories when compared to the Singapore standard types of tenures (figure y) shown in figure x, which presented the raw data tenure categories. Our solution was to merge them with the neighboring tenure categories in terms of the length of the lease.

There are 1595 missing values in this column overall, so we just placed them to a new category called "others" to handle the missing data. Additionally, we think that records falling under the "others" category can be handled by other features when utilizing a tree-based model. And the figure z shows the result distribution of each category.

Since there aren't many categories in this column, we chose to apply one-hot encoding to complete the final task, categorical data conversion.

A screenshot of a computer

Description automatically generated with low confidence

### Built year

We only consider how to impute the missing values from this column for the "built year" attribute. With the help of other features like property type, location (latitude and longitude), and block number (extracted from title address), we used a strategy to impute the missing records as much as we could. We found a common group that might contain the unique built year and used it to impute the missing records that belong to the same group.

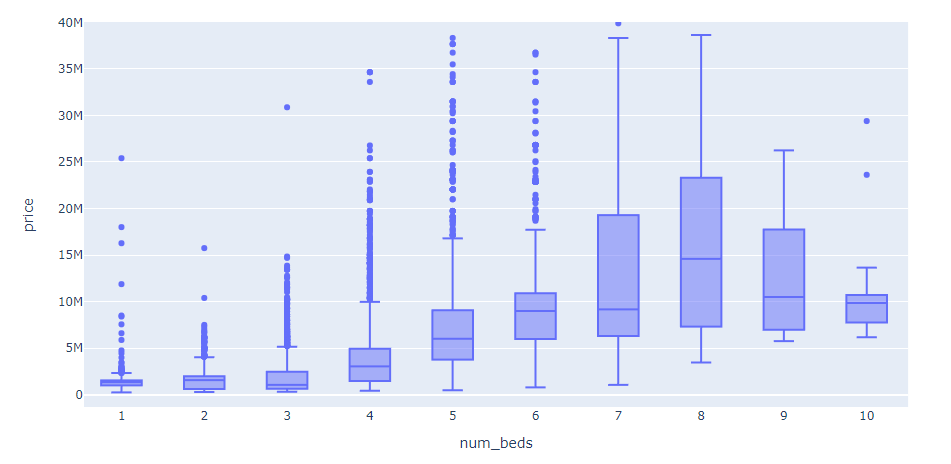
After the imputation, we were able to reduce the number of records missing the built year from 922 to 494. Regression algorithms like XGboost and LightBoost, which can handle the missing value by default. In the case of other regressors, we simply discard these incomplete records. The graphs in figure x below demonstrate the relationship between built year and price for the top 4 types of properties by population.

Graphical user interface, application

Description automatically generated

### Num beds

The 80 missing values for the number of beds were imputed using an approach that mainly relied on two sources: the number of beds data we obtained from the title field as indicated in earlier sections and the unique value or median value of grouping by the property size in sqft. We can handle every missing value from this field using these 2 sources. Figure X shows that the price increases in tandem with the number of bedrooms.



### Num baths

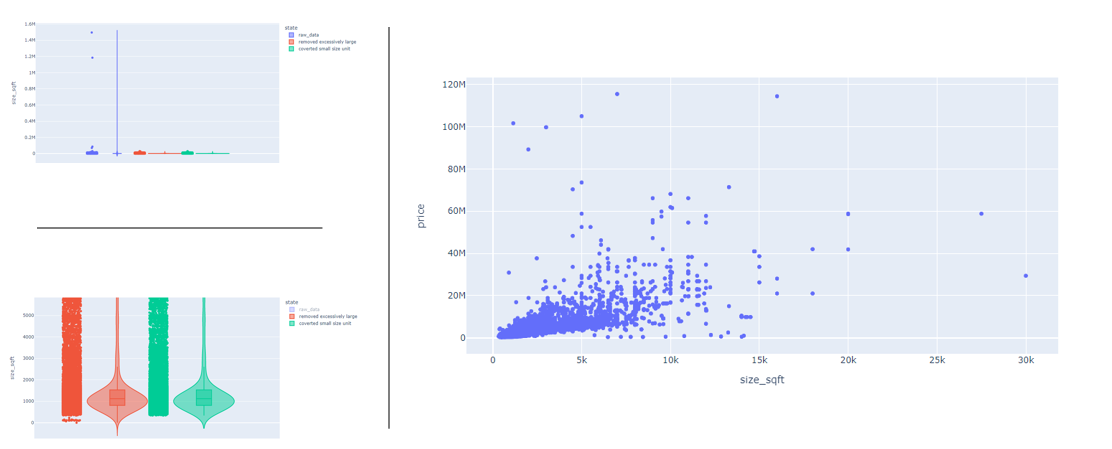
The "num baths" column includes 432 missing values; thus, we used a similar procedure to "num beds" to estimate the data. In order to complete this task, we decided to group data by "property type," "size sqft," and "num beds" and utilize either the unique or median value for imputation. Only 1 record remained NaN after imputation, therefore we decided to remove it. Figure X shows that the relationship between the number of bathrooms and the price follows a similar trend as the bedrooms.

Chart

Description automatically generated

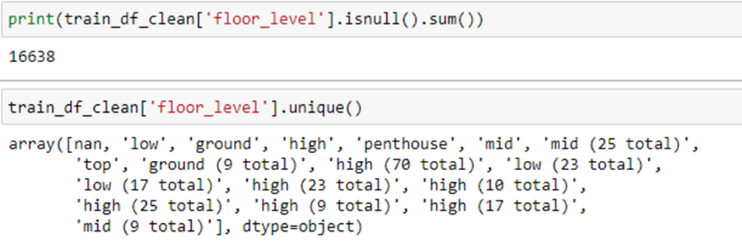
### size\_sqft

There are multiple incorrect records under the feature "size sqft," as we discovered. To start, we use the z score to eliminate excessively big values. Two records that included "size sqft" values more than one million have been removed from Figures x.1. In addition, some records include exceptionally small size, and we assume this is because the incorrect unit (sqm) was used. As a result, we also convert them using the ratio 10.76391 to convert from square meters to square feet (Figure x.2). It is clear from Figure x.3 that the size of the property has a positive relationship with the cost of the property.



### floor\_level

Figure m shows the quantity of blank values and distinct values in the "floor level" field. Since there are too many missing records, we decided to group them together under the heading "nan" for handling missing data. It's worth noting that this method is similar to how the XGboost algorithms handle Nan values by default. Additionally, we choose to separate the total level for some records from the "floor level," which will divide all values into categories in ['ground', 'low', 'mid', 'high', 'top', 'penthouse', 'nan', ‘no\_level’]. And we defined the properties such as bungalow, townhouse, and others as "no level." Then the total level specified earlier will likewise be regarded as an independent variable Figure x.1 illustrates how the floor level affects the price of a house for the majority of properties, and Figure x.2 demonstrates how the price of a high-floor house varies depending on the total number of building levels on the property. We choose one hot encoder for this field since it has a small number of features for category encoding.



Graphical user interface, chart, application, Teams

Description automatically generated

### Furnishing

In raw data, there are total 5 categories under furnishing feature: ['unspecified' 'partial' 'unfurnished' 'fully' 'na']. After encoding, we combine the categories "unspecified" and "na" to minimize the dimensionality. Due to the small number of features after encoding and the fact that there is no particular correlation(Figure x) between furnishing and price, we decided to use one hot encoder for this task.

Chart, box and whisker chart

Description automatically generated

### available\_unit\_types

The following 4 features can be extracted from "available unit types" (Figure x):

* min\_br\_available: the building's smallest number of bedrooms available
* max\_br\_available: the building's largest number of bedrooms available
* number\_of\_types\_available: how many distinct types of units are offered by the building
* has\_studio: Indicates whether a building has a studio-type property

Chart, box and whisker chart

Description automatically generated

### planning\_area

### Others (subszone, property\_details\_url, elevation)

Fields like subzone and property\_details\_url are simply removed from the model building in the case of the curse of dimensionality. Additionally, we remove elevation from the training dataset because it only contains one unique value and makes no contribution to the model at all.