Predicting the Resale Price of HDB Flats in Singapore

[*]Final Project of NUS CS5228-2020/2021 (Semester 2)

Team 5

Xiong Kexin
A0196551X
e0389037@u.nus.edu

Yao Yihang
A0209966M
e0454417@u.nus.edu

Liu Xing
A0225169B
e0575803@u.nus.edu

***Abstract*—Based on the properties of HDB flats in Singapore, the resale price can be predicted by data mining methods. This report will show how we did implement the knowledge we learnt through CS5228 course, practice EDA, data preprocessing and model training and complete the prediction of the resale price of HDB flats in Singapore.**

***Index Terms*—price prediction, knowledge discovery, data mining, regression task.**

I. Introduction

The resale market of HDB flats is big business in Singapore. To find a good prices as either a buyer or a seller, it is important to have good understanding of what affects the market value of a HDB flat. The goal of this project is to predict the resale price of a HDB flat based on its properties. To make better prediction, exploratory data analysis and data preprocessing, feature engineering and model selection is implemented.

II. Motivation and Goals

A. Motivation

The majority of the residential housing in Singapore are HDB flats, which are home to approximately 78.7% of the resident population. This also contributes to the large size of the resale market of HDB flats in Singapore.

Price is definitely a significant factor that buyers, sellers or agents would take into account. For example, a buyer who have children may want to know how much it costs to choose a relative new HDB flat that is near both a primary school and a train station. A seller may need to know how much he/she can get for selling a 100-sqm, high-floor HDB flat, which is near a shopping mall. Also, the agents are supposed to give advice to their seller clients about how to set a proper price which will attract potential buyers without the loss of the profit of sellers.

In this case, it is meaningful to figure out what properties have an influence on the market value of HDB flats and also do the prediction of the resale price of a HDB flat given its several properties.

B. Goals

Our project aims at predicting the resale price of an HDB flat based on its properties, including size, number of rooms, type, model, location, nearby amenities and its so on. Since it is obviously a regression task, we plan to use different regression techniques to do this prediction and compare their results. In addition, other results such as the importance of different attributes, error analysis, and other necessary discussion will be included.

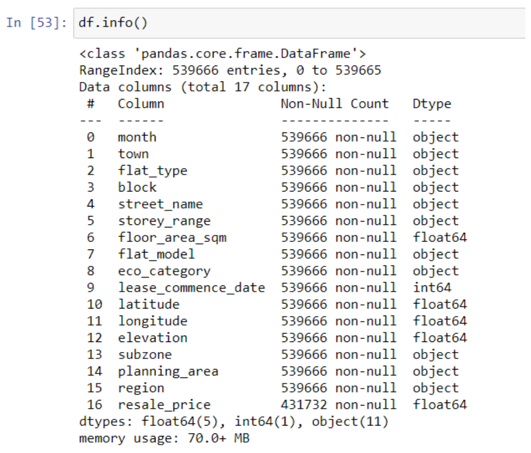
III. Exploratory Data Analysis and Preprocessing

A. Exploratory Data Analysis

Exploratory Data Analysis is a process of performing investigations on datasets so as to gain initial impressions, to discover patterns, to spot anomalies and to get summary statics and visualisations.

In good practice to understand the data and get insights from it, we first loaded the datasets and did Exploratory Data Analysis on it. By using "head()" function of pandas library, we can see the raw data structure, which consists of 16 features and 1 target ('resale price'). By using ".shape", we found out there are 431732 samples of training data, and 107934 samples of testing data, which is more than sufficient. It is also a good practice to know the columns and their corresponding data types, along with finding whether they contain null values or not. By using ".info()", we can simply get those information.

We then look into dependent features to get more insights. By using histograms, we visualized the distribution of a feature over different categories. Fig.2 shows the distribution of "flat model", which turns out to be imbalanced. We also used boxplot to preliminary explore the correlation between a categorical feature and our target label "resale price". Fig.3 shows the boxplot of "flat type" over target resale price. The obvious floating indicates that there're might be strong correlation between "flat type" feature and our target resale price.



```
In [53]: df.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 539666 entries, 0 to 539665
Data columns (total 17 columns):
 #   Column               Non-Null Count   Dtype
---  ------               --------------   -----
 0   month                539666 non-null  object
 1   town                 539666 non-null  object
 2   flat_type            539666 non-null  object
 3   block                539666 non-null  object
 4   street_name          539666 non-null  object
 5   storey_range         539666 non-null  object
 6   floor_area_sqm       539666 non-null  float64
 7   flat_model           539666 non-null  object
 8   eco_category         539666 non-null  object
 9   lease_commence_date  539666 non-null  int64
 10  latitude             539666 non-null  float64
 11  longitude            539666 non-null  float64
 12  elevation            539666 non-null  float64
 13  subzone              539666 non-null  object
 14  planning_area        539666 non-null  object
 15  region               539666 non-null  object
 16  resale_price         431732 non-null  float64
dtypes: float64(5), int64(1), object(11)
memory usage: 70.0+ MB
```

Fig. 1. Basic information of dataset

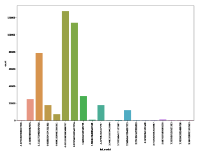


Fig. 2. flat model

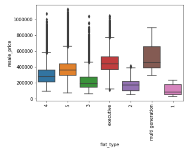


Fig. 3. flat type

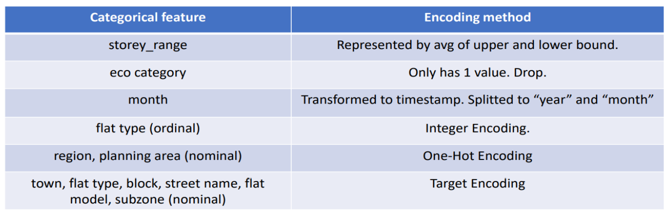
B. *Feature Engineering*

Feature engineering is a process of transforming the given data into form of which is easier to explain and interpret. Here we aim at making the features better suited to our problem at hand, the prediction of HDB resale price. Some features need to be encoded or transformed, some new features need to be invented. In this section, we focus on transformation and encoding of existing features.

For a feature to be useful, it must have a mathematical relationship to the target for models to learn. For numeric features, we decided to leave them as they are, since after several experiments, we found that both clustering technique and normalization do not contribute to our final result. The feature ”elevation” contains only 1 value ”0.”, which is considered equivalent to all data points. So we simply drop this column.

However, there’re still many features that are given in non-numeric form, which are categorial features. These features can take on values from a limited set of values. Some algorithms like decision tree can directly learn from categorical data, but many machine learning techniques and algorithms cannot operate on label data directly. They require all input variables to be numeric. That’s why for each category, we need to introduce an integer number representing it.

First of all, we found that categorial feature ’storey range’ is actually a clustering of numeric feature ’storey’, saved in object data type. Therefore, we using the average of upper and lower bound to represent this feature, which somehow reflects the difference of storey height among HDB flats. After that, we observed that feature ’eco category’ contains only 1 value, so we drop that column. What’s more, ’month’ feature has format of ”Year-Month”, which contains the temporal informaition. Instead of simply treating them as sets of values. We transformed it into timestamps, and also splitted it into two two features ”year” and ”month” to capture richer information.



Categorical feature	Encoding method
storey_range	Represented by avg of upper and lower bound.
eco category	Only has 1 value. Drop.
month	Transformed to timestamp. Splitted to “year” and “month”
flat type (ordinal)	Integer Encoding.
region, planning area (nominal)	One-Hot Encoding
town, flat type, block, street name, flat model, subzone (nominal)	Target Encoding

Fig. 4. Encoding of Categorical Features

One way of encoding is called integer encoding, which is simply using integers to represent different values of variables. Since the integers have natural internal ordering, this method can only be applied on ordinal variables, where the variables also have natural internal ordering. We observed that, feature ”flat type” has an internal ordering in the size of the house. There are seven different values, ’1 room’ ’2 room’ ’3 room’ ’4 room’ ’5 room’ ’executive’ and ’multi generation’, which are encoded as 1-7.

Another common way of encoding is One-Hot, which means to expand the variable to a dimension of number of unique values, and label the true unit as ”1”, while others remain zero. This method can be applied on nominal variables where there’s no natural orders. However, one-hot encoding is not suitable for features with too many values, or it will add large, sparse dimensionality of spaces and heavy burden. We finally applied One-Hot encoding on feature ”region” and ”planning area”.

For those nominal variables with large number of unique values, Target Encoding becomes the best choice. Target encoding is basically replacing a categorical value with the mean of the target variable (resale price). We used target encoding to encode features ’town’, ’flat type’, ’block’, ’street name’, ’flat model’ and ’subzone’. Therefore, target encoding turns out to be the most popular and applicable way in our case.

C. *Feature Generation*

In addition to the features that the core dataset file contains, we came up with some new features that can improve the accuracy of our predictions.

There are two features related to time in the original data: ”month” and ”lease_commence_date”. The feature ”month” does not only include the information about which month but also which year when a HDB flat was sold. And the feature ”lease_commence_date” tells us when the lease for a flat commenced.

a) Year of the sale: We can know the year when the sale happened from the existing feature ”month”. This feature can be helpful because the price of a flat may be influenced by the economy and policies of the year.

- year: This feature represents the year when the sale happened.

b) Number of years before sale: By subtracting the feature ”year” and the feature ”lease_commence_date”, the new feature named ”year_before_sale” is obtained. This new feature can be used to measure how old and new a HDB is, which buyers will pay attention to in real resale market. Compared to an old flat that could have many inevitable problems like conduit ageing, a relatively new flat is usually preferred by buyers.

- year_before_sale: This feature represents the number of years between the time when the lease for a flat commenced and the time when this sale happened.

D. Auxiliary Data Using

a) Distance to nearest facilities: Although the correlation coefficients between both ‘latitude’ and ‘longitude’ and ‘resale_price’ are very low, we decide to use these two features to relate an HDB flat’s location to its surrounding facilities(leveraging the auxiliary data). It is because that the value of a flat is not only depend on its own attributes, but also on how easy and convenient its neighborhood is to live in. For example, buyers who have children are likely to buy a flat that near to schools; buyers who enjoy going shopping may want to live near shopping malls.

To assess the degree of the convenience of a flat, we firstly calculated the distance between flats and every facilities (including MRT stations, shopping malls, commercial centers, markets, and schools), by using features ‘latitude’ and ‘longitude’.

Then, we take the smallest distance to certain type of facility as a new feature, which represents how far the nearest facility is from the HDB flat. The new features we generated are as follows:

- to_commercial: The distance between a HDB flat and the nearest commercial centre.
- to_gov_markets: The distance between a HDB flat and the nearest market.
- to_malls: The distance between a HDB flat and the nearest shopping mall.
- to_primary_schools: The distance between a HDB flat and the nearest primary school.
- to_secondary_schools: The distance between a HDB flat and the nearest secondary school.
- to_train_stations The distance between a HDB flat and the nearest train station.

It is noteworthy that, when computing the distance to train(MRT) station, we took the ”opening year” into consideration, which means that only stations opening before the sale year are considered. Besides, not all of these new features are finally used in our experiment. According to the experiment results, some of them are helpful, but others are not.

b) Number of facilities in the neighborhood: Besides the distance to the nearest facilities, we also assume the number of facilities in the neighborhood an important factor. Therefore, we generate new features to represent the number of facilities within a radius of 2 kilometers:

- num_of_cc: The number of commercial centres within a radius of 2 kilometers to a HDB flat.
- num_of_gov_market: The number of markets within a radius of 2 kilometers to a HDB flat.
- num_of_pri_school: The number of primary school within a radius of 2 kilometers to a HDB flat.
- num_of_sec_school: The number of secondary school within a radius of 2 kilometers to a HDB flat.
- num_of_shopping_mall: The number of shopping malls within a radius of 2 kilometers to a HDB flat.
- num_of_train_station: The number of train stations within a radius of 2 kilometers to a HDB flat.

c) Demand for Sold Flats: We collected ”resale price index” and ”demand for sold flats” as additional data from Data.gov.sg. However, since the ”resale price index” can be considered our target in a sense, we dropped it and only took ”demand for sold flats” as additional data feature. ”Demand for sold flats” does not perform well and was not used in our final experiment.

E. Feature Selection

Combining correlation analysis and the importance weight of features, we select features that have both a relatively high correlation and importance weight to train the model. For features in original data, we decided to drop block, town and street_name. For features calculated from auxiliary data, features selected are as follows:

- num_of_cc
- num_of_gov_market
- num_of_pri_school
- num_of_sec_school
- num_of_shopping_mall
- num_of_train_station

It is worth noting that, even if we thought feature correlation to target as an important criterion to select feature, the correlation value does not always matching the feature importance when trained on regressors. Here are two histgrams, showing the correlations and importance of the 5 most important features when applied on XGBoost:

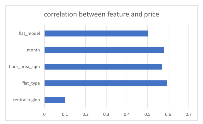


Fig. 5. correlations

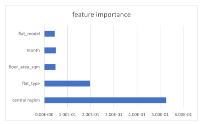


Fig. 6. importance to XGBoost

We can see from these two histgrams , feature ”centre region” is fairly important to XGBoost when doing regression, but the correlation of it is not the highest.

IV. Data Mining Methods

The resale price of HDB flats is continuous variable, therefore, we need regression model to predict on it. Different kinds of regressors are trained on the data and evaluated.

A. Models

1) Linear Regression: The first model we try is basic linear regression. However, the result is not good because the model is rather biased, the output is always a regression line, plane or hyperplane, so it might be restrictive for this dataset. Therefore, linear regression is taken out of consideration. Seeing the poor performance of linear regression, we seek for more complicated models for the prediction.

2) Random Forest: Random forest is the second model we try. It has a much better performance than linear regression. This is the model that we got a relatively high score for the first time. However, long training time makes further improvement based on random forest difficult. When we want to test the effect of feature engineering, or try hyper-parameter tuning, we found it takes too much time to train the random forest regressor, because trees in the forest are trained sequentially. In this situation, further improvement done with random forest is not practical and efficient for us.

3) Xgboost: In boosting methods, trees can be trained in parallel. Besides, training xgboost can be speed up by with GPU. Therefore, we switched to Xgboost seeking for better performance as well as shorter training time. After first trying Xgboost with default parameters, we get a score around 19000, which is not bad. For the training time, it takes around 20 minutes on CPU with default parameters. And after setting the training on GPU, it only takes around 2 minutes. Therefore, Xgboost is chosen as our final prediction model. After finalizing our choice, we also did hyper-parameter tuning based on Xgboost to make further improvements.

B. Hyper-parameter Tuning

Xgboost is a tree-based model, therefore, adjusting parameters for tree booster must have an influence on the performance of the tree. For tree booster parameters, we first use grid search to find the best parameter for max_depth and n_estimators. After grid search, we find the best max_depth is around 12, and the best n_estimators is around 500. If the max_depth and the number of estimators is greater, the scores goes down because of overfitting. Next, learning_rate and minimum_child_weight is found with max_depth and n_estimators fixed. Minimum_child_weight is another parameter for tree booster, it is the minimum sum of instance weight (hessian) needed in a child. If the tree partition step results in a leaf node with the sum of instance weight less than min_child_weight, then the building process will give up further partitioning. Min_child_weight is a parameter relevant to the number of the features. We need to adjust it manually every time when the number of features is changed, but the best values are from 10 to 16. Learning_rate is step size shrinkage used in update to prevents overfitting. After each boosting step, we can directly get the weights of new features, and eta shrinks the feature weights to make the boosting process more conservative. Because of the synergy between multiple parameters, learning rate is first adjusted through grid search and then adjusted in a small range manually and the best learning rate is around 0.06.

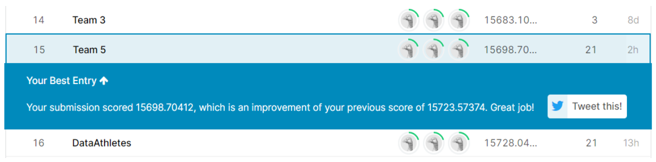
V. Evaluation and Interpretation

Linear Regression gets the worst score. The cross-validation score with default parameters is around 50000. Because linear regression is biased, the output is always a regression line, plane or hyperplane, so it might be restrictive for this dataset.

Random forest get the highest score with default parameters, which is over 17000. The superior performance of random forest mainly depends on randomly sampled samples and features and integrated algorithms. The former makes it more stable against overfitting, and the latter makes it more accurate.

Xgboost is the second best with default parameters. The difference between Xgboost and random forest is small, xgboost gets over 19000. However, with hyper-parameter tuning, Xgboost got the highest score. Xgboost performs well because it improves accuracy by continuously improving the loss residual and reinforces the knowledge learned by mistakes.

Final Result: We finally got 15698.7 on Kaggle competition, ranking 15th.



14	Team 3	15683.10...	3	8d
15	Team 5	15698.70...	21	2h

Your Best Entry

Your submission scored 15698.70412, which is an improvement of your previous score of 15723.57374. Great job!

16	DataAthletes	15728.04...	21	13h

Fig. 7. Encoding of Categorical Features

Breakdown of Workload

Here is a breakdown of our workload.

TABLE I

Tasks	Team Members		
	Xiong Kexin	***Yao Yihang***	***Liu Xing***
Exploratory Data Analysis	✓	✓	✓
Preprocessing	✓	✓	✓
Model Training	✓	✓	✓
Report Writing	✓	✓	✓