Singapore Real Estate Prices-Prediction and Recommendation with Machine Learning

* CS5228 Final Project

Lan Fangzhou	Gan Kaiyuan	Yi Han	Chen Xuanhe	Wang Zihong
A0228554Y	A0248326B	A0250721R	A0228535B	A0230339L

***Abstract*—This paper describes a data mining model for predicting the Singapore housing prices as well a method to recommend house user may like. The raw data includes house factors and auxiliary data which includes transportation factors of target real estate, locations of various centers such as commercial centers, markets, schools, shopping malls. The data was went through several rounds of cleaning and preprocessing. Intuitively, the location of property is a primary factor affecting the price and thus we make use of the location information to join with auxiliary information. The output of cleaned data was went through dimension reduction by Normalization and Label Encoding.**

Though we have made great efforts on data processing and model selection, the error rate is nevertheless much higher than we expected. We argue that there are still many factors which can not be obtained by the given data but influence much on the price. There is still value in the practice to figure out a more practical way of house prediction and recommendation that satisfy the requirements of real estate in the industry.

***Index Terms*—data preprocessing, data mining, machine learning, prediction, recommendation**

I. Motivation & Goals

A. Motivation

In spite of concerns about global economic slowdowns, Singapore's real estate market has shown greater resilience than elsewhere.

In the housing market, housing price ranges are of great interest for both buyers and sellers. There are many factors that affect housing resale prices, such as floor, size, location, etc. Areas with good accessibility to MRT stations, schools and shopping malls contribute to the rise in housing prices.

Therefore, it is critical to find out how various factors affect housing prices and to make predictions. In addition, it is important to provide users with a range of options based on the information they are interested in about a property.

B. Goals

- Understand how Singapore housing price correlates with multiple factors. Recommend similar housing from a housing sample.
- Recommend users with housing based on user's preference on affordable price, subzone, school and mall requirements, minimum distance to MRT, etc.
- Learn how to overcome obstacles in data preposessing: visualizing correlation between features and price, selecting useful features, handling NA data, caculating derived features from original features and text encoding.
- Learn how to select appropriate models for regression and recommendation tasks.

II. Exploratory Data Analysis & Preprocessing

A. Data Cleaning & Feature Selection

- Dropping features with too much NA values: Here are the number NA values in each feature out of the 20254 records in the train dataset: tenure 1723, built_year 922, num_beds 80, num_baths 434, floor_level 16746, available_unit_types 1441, total_num_units 5652, subzone 113, planning_area 113. Because there are too many missing values(more than 25%) in floor_level, total_num_units, we simply drop thes features because they are hard to feed to regressors.
- Dropping useless feature: Then we find some other features that has no contribution to this task, and therefore dropping them:
  - The elevation feature only contains the value of 0, so we drop it.
  - The listing_id and property_details_url feature only act as a id label, and has almost nothing to do with the housing price, so we drop them.
  - The address and property_name feature are hard to parse and hard to feed to a regressor. Besides, the information in these 2 features are mostly covered by planning_area and subzone features, so we drop address and property_name features.
- Handing NA values: for the features with not too many NA values, we need to find appropraite methods to fill these NA value. For numeric features like built_year, num_beds, and num_baths, we fill NA by using mean value. For categorical features like available_unit_types and tenure, we fill NA using mode value.
  Besides, there is a further improvement on the NA filling of tenure feature: filling the tenure feature by the mode of each property_type, the reasons are as followed:
  - The property_type feature has no NA value
  - Each property_type has different value distribution: for example, the condo type is about half 99 year

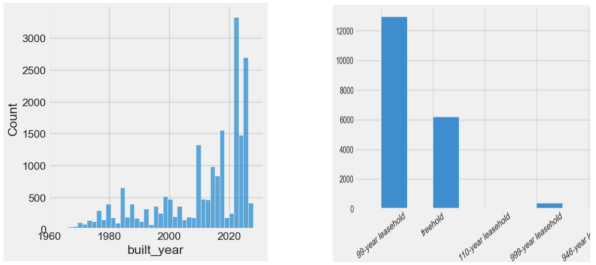
tenure and half freehold, but the hdb type is all 99 year tenure.

B. Feature Preprocessing

- Normalize cases: for features like property_type, there are values with upper case and lower case characters. We need to change it all to lower case or upper case before feed these categorical feature to a regressor.
- Filter out outliers: filter out size_sqft that are too high or too low; filter out price that are negative values; fill with mean value: for lat / lng with outlier values in the test set(some of them are not in Singapore, actually in Philippine), filling them with mean value in the subzone
- Filter out unreasonable price per sqft: we calculate the price per sqft, which is an important indicator for the housing price, using price and size_sqft features. Then we filter out housing with unreasonable high or unreasonable low price per sqft, to make the data cleaner before feeding into a regressor.
- Simplify categorical features: some categorical features like tenure and available_unit_types has too many types. It may lead to underfit if these types are not available in test data because we can only predict the price without this feature, or may lead to overfit when these types are available in test data because the model would be too complex.
  - For the tenure feature: we merge different types into 3 types based on tenure time ranges: tenure-100, tenure-1000, freehold.
  - For the available_unit_types features, we parse strings like "studio, 1, 2, 3, 4, 5 br" and transfer it into 3 features: minimum bedroom number, maximum bedroom number, and whether has studio or not. It reduce the number unique types from 187 to less than 10 for all these features. More importantly, these new features achieve much clearer semantics than the original complex string.
- Auxiliary Datasets: we also take into consideration the auxilary datasets. The main features we considers are as followed.
  - sg-shopping-mall: Compute the min distantce from property to one shopping mall, it is related to the convenience of housings.
  - sg-mrt: Compute the min distantce from property to one mrt station, it is related to the convenience of housings.
  - sg-commercial: Compute the min distantce from property to one commercial center, it is related to the convenience of housings.
  - sg-primary-school: Compute the nearest distance to primary schools for each property, because primary school admissions are related to housing's distance to primary schools.
  - sg-secondary-school: Compute the nearest distance to secondary schools for each property
  - sg-subzone: Compute the density of population within a subzone, it may be a good indicator of whether the house is for ordinary people or for rich people

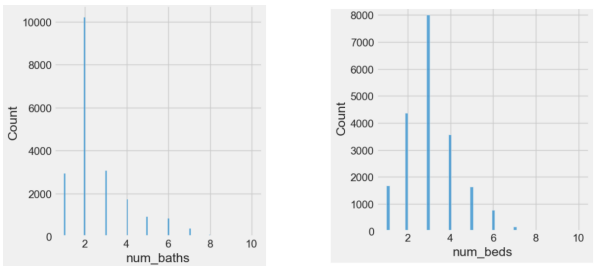
C. Exploratory Data Analysis (EDA)

After preprocessing, there are 19939 entries, each with 19 non-null attributes.



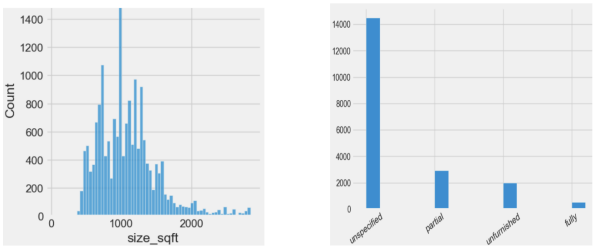
((a)) built year ((b)) tenure

Fig. 1: distribution of built year / tenure



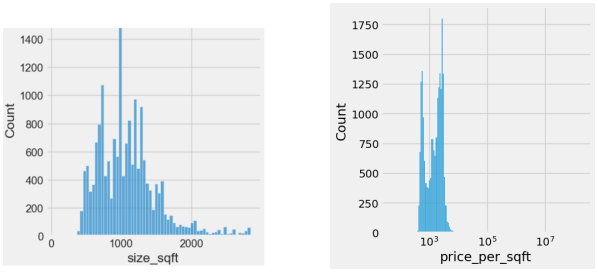
((a)) num baths ((b)) num beds

Fig. 2: distribution of size sqft / furnish status



((a)) size sqft ((b)) furnishing

Fig. 3: distribution of room status



((a)) price ((b)) price per sqft

Fig. 4: distribution of prices / prices per sqft

	built_year	num_beds	num_baths	size_sqft	lat	\
count	19944.000000	19944.000000	19944.000000	19944.000000	19944.000000	
mean	2010.799338	3.091306	2.591005	1615.775371	1.436303	
std	15.505853	1.234577	1.401084	1556.319931	1.570444	
min	1963.000000	1.000000	1.000000	65.000000	1.239621	
25%	2001.000000	2.000000	2.000000	797.000000	1.307929	
50%	2016.000000	3.000000	2.000000	1119.000000	1.329511	
75%	2023.000000	4.000000	3.000000	1507.000000	1.373123	
max	2028.000000	10.000000	10.000000	9600.000000	69.486768	

	lng	price	price_per_sqft
count	19944.000000	1.994400e+04	19944.000000
mean	103.855566	2.844548e+06	1739.537656
std	3.621256	4.142697e+06	1192.874725
min	-77.065364	2.499000e+05	219.682152
25%	103.806576	8.274000e+05	705.000000
50%	103.842068	1.669500e+06	1640.625000
75%	103.881514	3.156000e+06	2520.000000
max	121.023232	1.155000e+08	44625.000000

Fig. 5: summary of numeric features

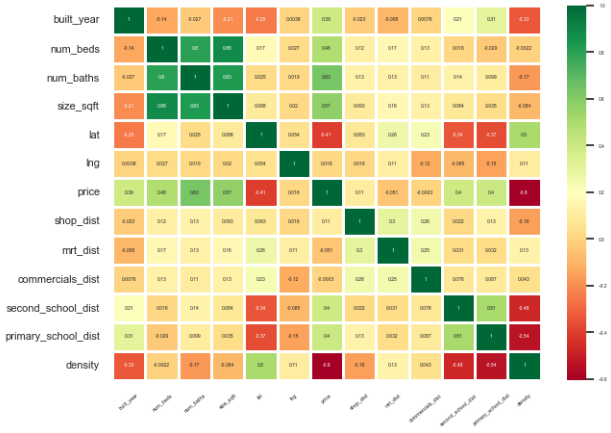


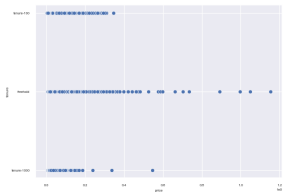
Fig. 6: correlations between numeric features

- Explorative Data Anaylysis (EDA): before further processing, we make a simple explorative data analysis for different kinds of features in the train dataset. The results are shown in section II-C, section II-C, section II-C, section II-C. The summary of numeric features are shown in fig. 5. It can be found that these features all have reasonable semantics and most of them follow normal distribution. There are no obvious outliers according to fig. 5.
- Correlation Analysis for numerical features: it is important to learn about how each feature relates to each others before determining which features we should feed to the regressors. The correlations between numeric features are shown as a matrix in fig. 6. We use the spearman correlation which is suitable for continuous variables. Here we make some discussions about what we find on these correlations.
  - Top indicators of price: num_baths(0.63), density(-0.6), size_sqft(0.57), num_beds(0.45), lat(-0.41), second_school_dist(0.4), primary_school_dist(0.4), build_year(0.39), available_unit_type_max_room(0.39). These top indicators are of top consideration to feed into the regressor models.
  - From these indicators, num of rooms and room size are easy to understand as important positive features, and built year is obviously a negative feature.
  - Density is a important negative feature because rich people prefer to live in condos or houses within lower density areas: it also can account for why longer distance to schools lead to higher price.
  - About lat feature, it is useful because that the CBD and commercial centres are mostly located at south of Singapore. However, the lat feature is mostly a duplicate for planning_area and subzone features.
- Correlation Analysis for categorical features: the correlation between tenure, furnishing, and property type to price is shown in section II-C, section II-C. It can be found that there are some correlation between each value of these feature to the price. Therefore it is reasonable to feed these categorical features into regression models (after doing one-hot encoding).

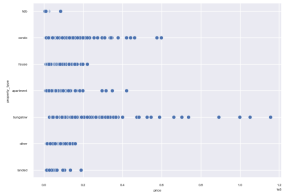
((a)) tenure correlation

((b)) furnishing correlation

Fig. 7: correlation of tenure and furnishing to price



((a)) tenure correlation



((b)) property type correlation

Fig. 8: correlation of property type to price

III. Data Mining Methods

A. Task 1: Prediction of Resale Price

Task 1 is a regression task to predict property resale prices. Different regressors are trained and evaluated.

1) Model Selection:

The model we are mainly testing about are all Decision Tree based models because it could be easier to handle mixed numeric and categorical features. We tried a number of regressors including Decision Tree, Random Forest [1], Gradient Boosting [2], LightGBM [3], XGBoost [4] and Cat Boost [5].

In the case of random forest [1], the collection is made up of many decision trees. Random forest is trained using a random subset of training data (sample bagging) and random subset of input features (feature bagging) to obtain diverse trees. Bagging decrease the high variance and tendency of a weak learner

model to overfit a dataset. Instead of bagging and creating many weak learner models to prevent overfitting, an ensemble model may use boosting technique to train a strong learner using a sequence of weaker learners. XGBoost [4], CatBoost [5] and LightGBM [3] have emerged as the most optimized boosting techniques for gradient-boosted tree algorithms.

LightGBM [3] construct decision trees using Gradient-Based Obe-Side Sampling (GOSS). GOSS looks at the gradients of different cuts affecting a loss function and updates an underfit tree according to a selection of the largest gradients and randomly sampled small gradients. GOSS allows LightGBM to find the most influential cuts. Similar to LightGBM [3], XGBoost [4] also uses the gradients of different cuts to select the next cut, but it also uses hessian, or second derivative in its ranking of cuts. Computing this next derivative comes at a slight cost, but it also allows a greater estimation of the cut to use. CatBoost [5] is based on gradient boosted decision trees. During training, a set of decision trees is built consecutively. Each successive tree is built with reduced loss compared to the previous trees. The number of trees is controlled by the starting parameters. To prevent overfitting, use the overfitting detector. When it is triggered, trees stop being built.

In our methods, we make several step of experiments to make the final result. First, we try decision tree regression and limited the size and height of the regressor by pruning. Second, we try random forest because it can effectively decrease the possibility of overfitting. And we also performed GradientBoosting, LightGBM and XGBoost as contrast. Finally, we use a stacking regressors to integrate the aforementioned models by simply average all the output by stacking model and the process is shown in fig. 9. The dataset is split into k folds, and in k successive rounds, k-1 folds are used to fit the first level regressor. In each round, the first-level regressors are then applied to the remaining 1 subset that was not used for model fitting in each iteration. The resulting predictions are then stacked and provided as input data to the second-level regressor. After the training of the StackingCVRegressor, the first-level regressors are fit to the entire dataset for optimal predicitons.

B. Task 2: Recommendation

The main goal of Task 2 is to build a recommender system to recommend most relevant properties given a row from the dataset and additional parameters.

The advantages of building a good recommendation engine can be categorized into two main parts:

- User perspective
  - Identify relative properties
  - Minimize effort to find relevant properties
  - Maximize satisfaction
  - Optimize spending of money and attention
- Provider perspective
  - Maximize transactions and sales of properties
  - Gain competitive advantages

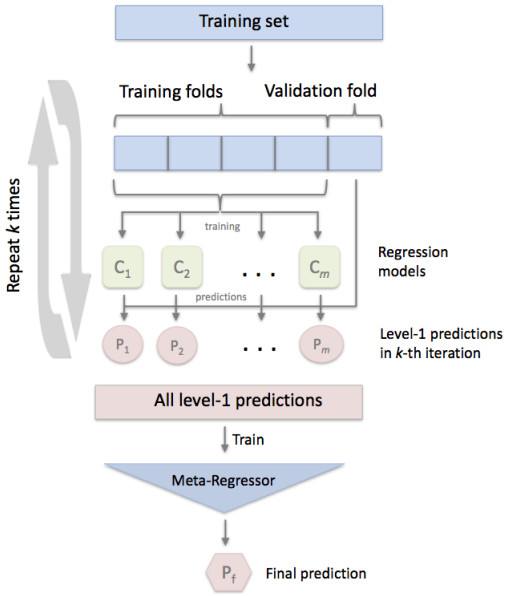


Fig. 9: The working process of StackingCVRegressor.

The most prevalent and mainstream methods of constructing a recommendation engine are Collaborative filtering (CF) and Content-based recommendation systems.

Therefore, we attempt to utilize the method of Matrix Factorization in Collaborative filtering to process the data. But it is worth noting that we need to preprocess the data to get clean data such as dropping those outliers as the longitude and latitude are out the range of Singapore. After preprocessing, there are 1980 rows of data, each with 7 non-null features. We put the data into a Non-negative Matrix Factorization (NMF).

We also try to use traditional content-based recommendation systems to return top k recommendations. Since the difference of each row is small, we compute the pairwise item similarity (cosine similarity) of given row and any other row in the dataset. The closer the value is to 1, the higher the similarity is and vice versa.

C. Task3: Open-tasks

For part 3, our group decided to do two open tasks: 1) Geographical visualizations, and 2) a Bargain system which combines the data mining methods we use in regression and recommendation.

1) Geographical Data Visualization:

Subtask 1 is the geographical visualization. We decide to plot the data on a real map of Singapore. Datapoints on the map represent the price per sqft of that space. After preprocessed the data, We utilize the folium [6] library to visualize the data in different forms, including heat map, dot map and a interactive clustering map.

- First we drop the datapoints that are not the house in Singapore according to the longitude and latitude.
- Then use the Map() function with latitude and longitude input to create the map.

- Use the Circle() function to circle the coordinates of housing.
- Create the color scale using LinearColormap() function to present the 'price_per_sqft' with colors.
- Use data in the form of 'latitude of home', 'longitude of home', 'price_per_sqft' to generate the heatmap with HeatMap() function.
- Use MarkerCluster() function, we mark each home with 'price_per_sqft' and combine near markers into a cluster.

2) Is it a Bargain:

Except for predicting the house price given a house (task 1) and finding other simialr house resources of it using recommendation (task2), our team are interested in indicating whether a house is a bargain using the information we get in the data, and what else we can do if we combine these two.

So, in this subtask, we designed a bargin system to analyze whether a home is a bargain (For buyers, the house is defined as a bargain if its price is lower than market value) or not.

The basic idea is as follows. The input of this task is a list of house attributes which can either represent a house listed on the website (containing the price), or a pure input by the user when searching houses. For simplicity in tests, we use the format in task2. The main goal is to not only give recommendations according to the input, but also an indicator telling whether each recommendation is a bargain or not (and the bargain extent).

Specifically, we fist use a similar content-based recommendation system as task 2 to provide similar housings for a given row. The difference is dropping the price feature when computing similarities, since in this task, we only want find houses sharing similarities despite the prices.

After that, we use the task 1 prediction model to predict the price of the given row and we consider it as the reasonable price. However, the row has fewer features than the input features needed for the prediction model. Thus, we perform additional data preprocessing steps on the given row:

- We group the original training data by 'property_name'. By merging the given row with the training data according to the 'property_name', we can impute 'lat' and 'lng' with mean and impute 'available_unit_types', 'furnishing', 'subzone' and 'tenure' with mode.
- For features 'built_year', 'num_beds' and 'num_baths', we fill NA by the mean of training data.
- For 'available_unit_types', we fill NA by the mode of training data.

For each recommended home:

- if price >1.2*reasonable price, it is labeled as 'Expensive';
- if reasonable price <price <1.2*reasonable price, it is labeled as 'Acceptable';
- if 0.8*reasonable price <price <reasonable price, it is labeled as 'Bargain';
- if price <0.8*reasonable price, it is labeled as 'Very good value'.

IV. Evaluation & Interpretation

A. Preprocessing Evalution

We make use of the test data evaluation on Kaggle to better evaluate the effectiveness of preprocessing. Here are the improvements of several data preprocessing steps:

1) Preprocess property_type and tenure: we combine different categories of property_type and tenure to a much fewer number of categories: the test loss reduce from 2462596.54062 to 2142983.24009.
2) Preprocess available_unit_types and change it into three features: min room number, max room number, is studio or not: the test loss reduce from 2142983.24009 to 1948479.55603.
3) Filter outliers with price per $sqft < 100$ or $sqft > 100000$ (price per sqft is price / size_sqft): the test loss reduce from 1948479.55603 to 1871981.75605.
4) Fillna of tenure by group of property_type: the test loss reduce from 1871981.75605 to 1775960.98699.
5) If we do not normalize features using scale normalizer: the test loss increases from 1775960.98699 to 2027453.99205.
6) Fill in outlier of lng, lat using mean value in subzone, and then drop lng: the test loss reduce from 1775960.98699 to 1768645.33245.

B. Task 1: Real-Estate Price Prediction

We apply several decision tree models to predict the real estate prices and compare the methods both qualitatively and quantitatively.

We predict real estate prices directly from different kinds of decision tree models. In our settings, the train set is split by ratio of 4:1 into training set and validation set. We choose mean square error (MSE) as our metrics for performance evaluation as well as objective function.

From the result fig. 10, we can see that the LightGBM [3] performs worst and CatBoost [5] performs best. It is peculiar that the performance LightGBM of degrades drastically comparing to XGBoost [4] because the LightGBM [3] was raised to solve the problem such as long training time, large memory usage of XGBoost [4]. In the split of tree node, XGBoost uses a pre-sorted methods while LightGBM make use of histogram which decrease the usage of memory and increase the speed of training. However, the histogram can not calculate the split point of tree node as accurate as pre-sorted. In addition, the training set are too small and prices variation are too high so that LightGBM performs worse than XGBoost. We also report quantitative analysis of different decision tree models and the result is shown in table I.

We also compute the importance of different features of the cleaned dataset by implement XGBoost Feature Importance. As shown in fig. 11, the most important feature that used by XGBoost is the size of real estate, minimum distance to the nearest shopping mall and built year. That is quite intuitive because in reality, the property with larger size is often sold with higher price and the property which is closer to shopping

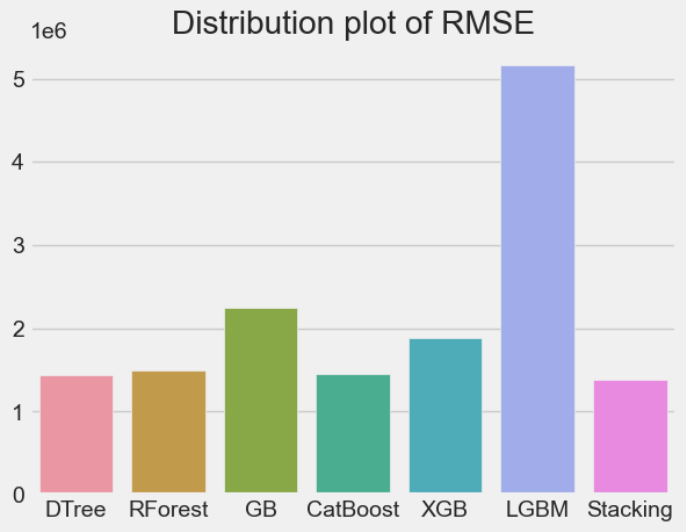


Fig. 10: Histograms of different decision tree models for real estate price prediction. Note that, the last column of the result is by stacking all the model, and computing the value by average the output of them

TABLE I: Quantitative analysis of different decision tree model. The table reports the result of Root Mean Square Error (RMSE) of each model on validation set. The value is divided by $1e6$

MODEL	RMSE OF VALIDATION SET
DECISION TREE	1.45 ± 0.1
RANDOM FOREST [1]	1.51 ± 0.3
GRADIENT BOOSTING [2]	2.24 ± 0.4
CATBOOST [5]	1.45 ± 0.2
LIGHTGBM [3]	5.14 ± 0.4
XGBOOST [4]	1.84 ± 0.3

malls such as the apartment near Orchard Road is often value more on the price. While the lowest importance features are furnishing type of the property, tenure of the property and available unit type is studio (a boolean value).

C. Task 2: Real-Estate Recommendation

For NMF, through enough time of training, we can easily find that the outcome is far from satisfactory as the loss is quite large and the predicted values don't realize reasonable results. It is possible that the method of matrix factorization mainly focuses on the rating matrix though we have employed the normalization on the data to make the values of different features as close as possible.

In the function of get-top-recommendations, we return top k rows with top k highest similarity with the given row. And we can discover that the final result is pretty close to the given row and seemingly reasonable. The result is shown in fig. 12.

D. Task 3: Open Task

1) Geo Visualization:

fig. 13 shows the distribution of houses on the map and

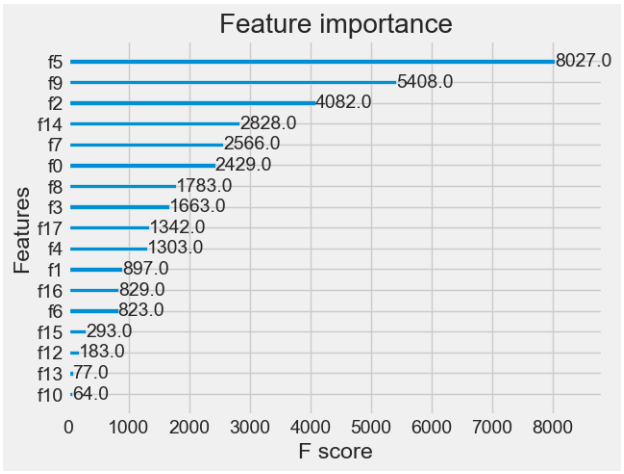


Fig. 11: Sorting of feature importance by XGBoost [4]. The top three important features are house size, the minimum distance to the nearest shopping mall and built year. The top three unimportant features are furnishing type of the property, tenure of the property and available_unit_type_is_studio.

	listing_id	title	property_name	property_type	built_year	num_beds	num_baths	size_sqft	planning_area	price
10	954673	5 bed house for sale in pasir ris beach park	pasir ris beach park	bungalow	1977.0	5.0	6.0	7534	pasir ris	9030000.0

```
k = 10
df_recommendations = get_top_recommendations(row, k=k)
df_recommendations.head(k)
```

	listing_id	title	property_name	property_type	built_year	num_beds	num_baths	size_sqft	planning_area	price
0	954673	5 bed house for sale in pasir ris beach park	pasir ris beach park	Bungalow	1977.0	6.0	6.0	7535	pasir ris	9030000.0
1	226790	6 bed house for sale in pasir ris beach park	pasir ris beach park	Bungalow	1977.0	6.0	6.0	7535	pasir ris	9030000.0
2	819222	5 bed house for sale in pasir ris beach park	pasir ris beach park	bungalow	1977.0	5.0	8.0	7535	pasir ris	9030000.0
3	825826	5 bed house for sale in pasir ris beach park	pasir ris beach park	Bungalow	1977.0	5.0	8.0	7535	pasir ris	8925000.0
4	590775	4 bed condo for sale in the peak	the peak	condo	1988.0	4.0	5.0	5500	queenstown	7980000.0
5	919788	6 bed house for sale in pasir ris beach park	pasir ris beach park	bungalow	1977.0	6.0	8.0	7535	pasir ris	9030000.0
6	675124	5 bed house for sale in chwee chian view	chwee chian view	bungalow	1991.0	5.0	5.0	5000	queenstown	6489000.0
7	971430	5 bed house for sale in pasir ris beach park	pasir ris beach park	bungalow	1977.0	5.0	6.0	5091	pasir ris	6510000.0
8	243201	5 bed house for sale in pasir ris beach park	pasir ris beach park	bungalow	1977.0	5.0	4.0	5990	pasir ris	6489000.0
9	731776	5 bed house for sale in clementi park	clementi park	bungalow	1985.0	5.0	5.0	8000	clementi	13629000.0

Fig. 12: Task 2 output

circles represent the coordinates of housing.

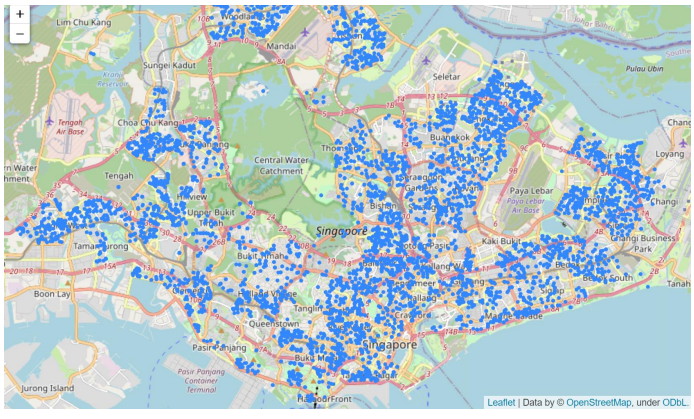


Fig. 13: Circle map pf properties

In fig. 14, the color of the circles shows the price_per_sqft of housing and a red circle indicates the high price.

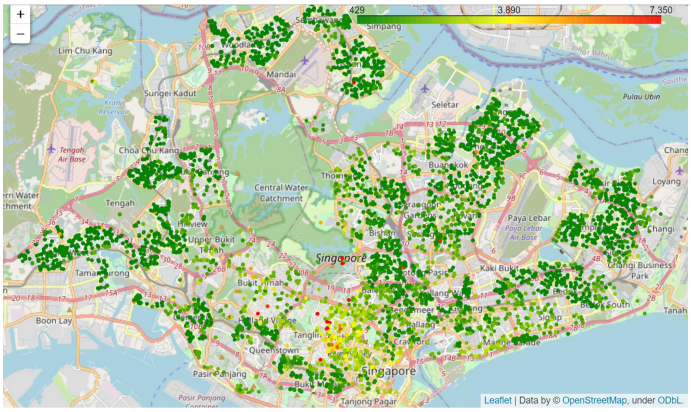


Fig. 14: Circle map pf properties

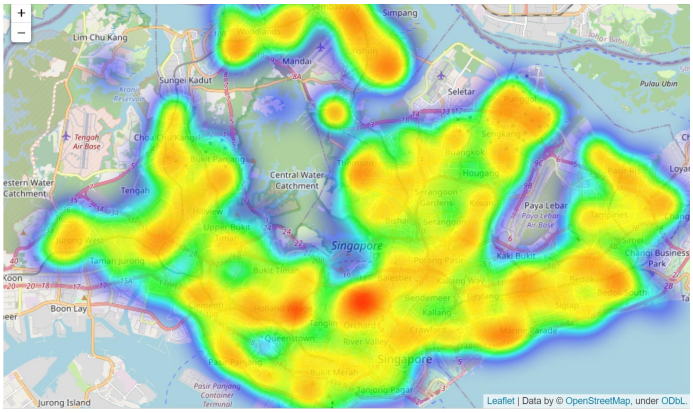


Fig. 15: HeatMap of price_per_sqft (overview)

Fig. 16: HeatMap of price_per_sqft (enlarged)

fig. 15 and fig. 15 are heatmaps on which the color represents the price per sqft of housing. As the figures shows, the color near Holland Village and Newtown are pretty red which indicating high house prices (per sqft). In the enlarged view new NUS, we can see more clearly that the high price of Holland Village especially compared to NUS campus where is blank meaning no house for sell.

In fig. 17, nearby houses are grouped into the same clusters and the labels represent the price_per_sqft of the house. When zooming in on the image, each cluster is further clustered.

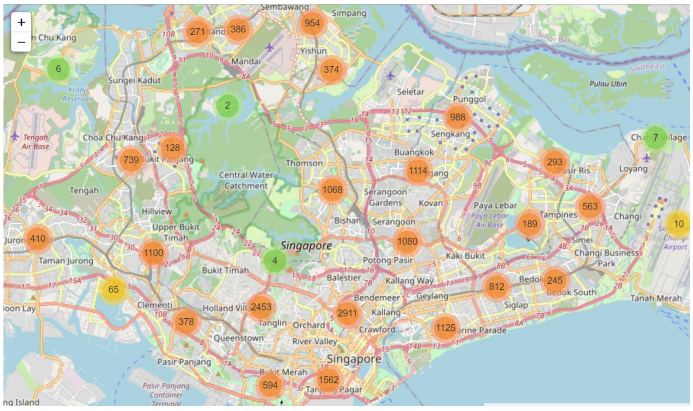


Fig. 17: Cluster map of Properties

2) Is it a Bargain:

Given a row as fig. 18 shown, we predict the reasonable price is 590911 Singapore dollars.

fig. 19 shows 10 recommended houses and bargain=1 indicates the house is a bargain.

	listing_id	title	property_name	property_type	built_year	num_beds	num_baths	size_sqft	planning_area	price
20	942099	hdb flat for sale in 275a bishan street 24	natura loft	hdb 4 rooms	2011.0	3.0	2.0	1022	bishan	945000.0

Fig. 18: The input including housing information

```
is_bargain(housing,reasonable_price).head(k)
```

	listing_id	title	property_name	property_type	built_year	num_beds	num_baths	size_sqft	planning_area	price	bargain
0	942099	hdb flat for sale in 275a bishan street 24	natura loft	hdb 4 rooms	2011.0	3.0	2.0	1022	bishan	945000.0	0
1	706957	hdb flat for sale in 54 havelock road	havelock view	hdb	2013.0	3.0	2.0	958	bukit merah	1050000.0	0
2	458413	hdb flat for sale in 275a bishan street 24	natura loft	hdb 3 rooms	2011.0	3.0	2.0	1292	bishan	1344000.0	0
3	709639	hdb flat for sale in 53 havelock road	havelock view	Hdb 5 Rooms	2013.0	3.0	2.0	1227	bukit merah	1365000.0	0
4	513288	hdb flat for sale in 53 havelock road	havelock view	hdb 5 rooms	2013.0	3.0	2.0	1227	bukit merah	1365000.0	0
5	837248	hdb flat for sale in 633c senja road	senja green	hdb	2013.0	3.0	2.0	1001	bukit panjang	659400.0	0
6	908051	hdb flat for sale in 807c chai chee road	ping yi greens	hdb 3 rooms	2016.0	3.0	2.0	1001	bedok	829500.0	0
7	213501	hdb flat for sale in 635a senja road	senja gateway	hdb	2015.0	3.0	2.0	990	bukit panjang	606900.0	1
8	862524	hdb flat for sale in 185 bedok north road	vista 8	hdb	2005.0	3.0	2.0	990	bedok	661500.0	0
9	447266	hdb flat for sale in 442b fajar road	fajar hills	Hdb	2016.0	3.0	2.0	996	bukit panjang	572200.0	1

Fig. 19: Recommended housing with bargain indicator

V. Conclusion

Throughout the project, we process and analyze data in depth. We use algorithms learned in class to build prediction and recommendation models and understand the application of these models to real-world problems. We also made some experiments using Deep Neural Networks. However we didn't contain this work in the repor since though due to the limitation of room and the performance is not so desirable.

A. Team Work Contribution

Member Name	Contribution
Gan Kaiyuan, Lan Fangzhou	EDA and data pre-processing; Task 1 implementation and tuning; Report writing
Yi Han, Chen Xuanhe	Data pre-processing; Task2 and Task3 implementation; Report writing
Wang Zihong	Task 3 implementation; Report writing.

REFERENCES

[1] L. Breiman, "Random forests," *Machine Learning*, vol. 45, no. 1, pp. 5–32, 2001.

[2] A. Natekin and A. Knoll, "Gradient boosting machines, a tutorial," *Frontiers in neurorobotics*, vol. 7, p. 21, 2013.

[3] G. Ke, Q. Meng, T. Finley, T. Wang, W. Chen, W. Ma, Q. Ye, and T.-Y. Liu, "Lightgbm: A highly efficient gradient boosting decision tree," in *Advances in Neural Information Processing Systems* (I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, eds.), vol. 30, Curran Associates, Inc., 2017.

[4] T. Chen and C. Guestrin, "Xgboost: A scalable tree boosting system," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, (New York, NY, USA), p. 785–794, Association for Computing Machinery, 2016.

[5] L. Prokhorenkova, G. Gusev, A. Vorobev, A. V. Dorogush, and A. Gulin, "Catboost: unbiased boosting with categorical features," in *Advances in Neural Information Processing Systems* (S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, eds.), vol. 31, Curran Associates, Inc., 2018.

[6] python visualization, "Folium."