Singapore HDB: Resale Price per Square Meter Prediction

Team 4: Li Rongrong, Zhang Tieyang, Soo Yik Hong

1. Introduction (Zhang Tieyang, Li Rongrong)

Housing Development Board (HDB) flats are a vital component of the housing market in Singapore, which provided by the government to offer safe and affordable housing options for Singaporen. Predicting the price per sqm of Singapore HDB has real-world significance, for instance, help people decide when to buy or sell a property for the best return, or ensuring social equity in the market and low-income families have chance to buy a suitable housing.

In this report, we will describe the dataset used for this project, how we attempted to get a comprehensive understanding of the dataset, which machine learning models we used, and some comparison and interpretation of our models.

2. Dataset & Pre-processing (Zhang Tieyang)

2.1 Download dataset

we got our original dataset from the website of Singapore government (https://beta.data.gov.sg/), which has following features:

month	town	flat_type	block	street_name	storey_range	floor_area_sqm	flat_model	lease_commence_date	remaining_lease	resale_price
Jan-23	ANG MO KIO	2 ROOM	406	ANG MO KIO AVE 10	01 TO 03	44	Improved	1979	55.41666667	267000
Jan-23	ANG MO KIO	2 ROOM	323	ANG MO KIO AVE 3	04 TO 06	49	Improved	1977	53.5	300000
Jan-23	ANG MO KIO	2 ROOM	314	ANG MO KIO AVE 3	04 TO 06	44	Improved	1978	54.08333333	280000

1) ***'month'***: the date of resold
2) ***'town'***: the town where the resold housing is located
3) ***'flat_type'***: the flat type of the resold housing
4) ***'block'***: the block of the resold housing
5) ***'street_name'***: the name of the street where the resold housing is located
6) ***'storey_range'***: the range of storey of the resold housing
7) ***'floor_area_sqm'***: the area of the resold housing
8) ***'flat_model'***: the flat model of the resold housing
9) ***'lease_commence_Date'***: Remaining lease term of the house (year)
10) ***'remaining_lease'***: the remaining lease of the resold housing
11) ***'resale_price'***: the resale price of the resold housing

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 163824 entries, 0 to 163823
Data columns (total 11 columns):
 #   Column               Non-Null Count   Dtype
---  ------               --------------   -----
 0   month                163824 non-null  object
 1   town                 163824 non-null  object
 2   flat_type            163824 non-null  object
 3   block                163824 non-null  object
 4   street_name          163824 non-null  object
 5   storey_range         163824 non-null  object
 6   floor_area_sqm       163824 non-null  float64
 7   flat_model           163824 non-null  object
 8   lease_commence_date  163824 non-null  int64
 9   remaining_lease      163824 non-null  object
 10  resale_price         163824 non-null  float64
dtypes: float64(2), int64(1), object(8)
memory usage: 13.7+ MB
```

We only use data of year 2023 for our project (20429 rows).

2.2 Expansion of dataset

To improve the performance of models, we first consider finding additional features that may influence the price per sqm. We first considered to get the latitudes and longitudes of all housing using its 'street_name' and 'block'. Meanwhile, noticed that the price may be related to the distance of a certain housing to the nearest MRT station and to a popular primary school, we also decided to collect the latitude and longitude of all MRT and LRT stations as well as the famous primary school in Singapore. We then found a dataset from Kaggle contains the geolocation of all stations (https://www.kaggle.com/datasets/yxlee245/singapore-train-station-coordinates), and with the help of OneMap API (https://www.onemap.gov.sg/), we manage to obtain the following features for each housing:

1/2) ***'lat/lng'***: Latitude and longitude of the housing
3/4) ***'min_mrt_lat/lng'***: Latitude and longitude of the nearest MRT/LRT station.

5/6) ***'min_primary_lat/lng'***: Latitude and longitude of the nearest popular primary school.

lat	lng	min_mrt_lat	min_mrt_lng	min_primary_lat	min_primary_lng
1.362004539	103.8538799	1.370025	103.849588	1.349782238	103.8545293
1.367908494	103.8477141	1.370025	103.849588	1.360583434	103.8330203
1.366227071	103.8500859	1.370025	103.849588	1.349782238	103.8545293
1.366227071	103.8500859	1.370025	103.849588	1.349782238	103.8545293

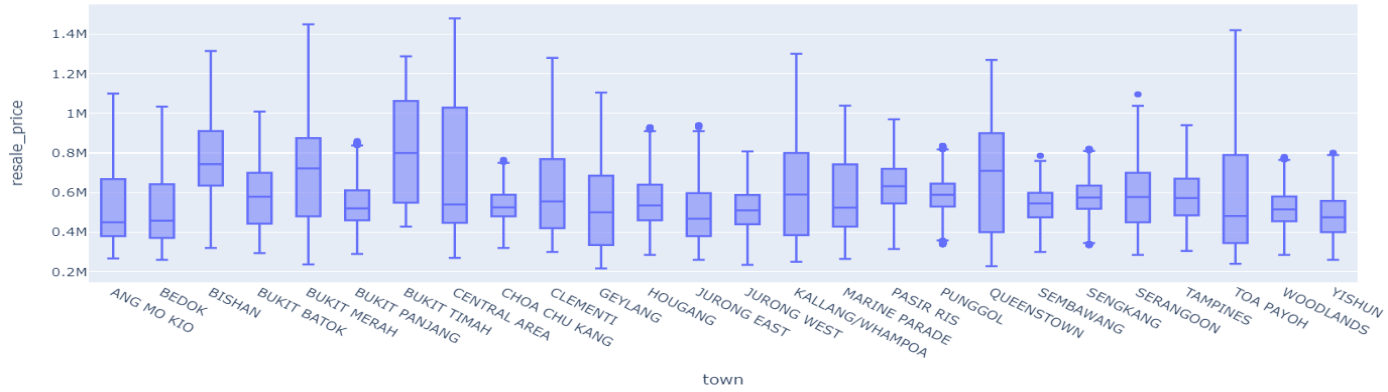
2.3 Create dummy variables

To utilize the columns with qualitative features, we created one hot encoding of categorical variables, made them all to dummy variables. Example:

1 ROOM	2 ROOM	3 ROOM	4 ROOM	5 ROOM	EXECUTIVE	MULTI-GENERATION
FALSE	TRUE	FALSE	FALSE	FALSE	FALSE	FALSE
FALSE	TRUE	FALSE	FALSE	FALSE	FALSE	FALSE
FALSE	TRUE	FALSE	FALSE	FALSE	FALSE	FALSE
FALSE	TRUE	FALSE	FALSE	FALSE	FALSE	FALSE

2.4 Drop outliers / missing values

To get better performance, we try to drop the outliers (Interquartile Range):



Then we try to find rows with missing value. Because the dataset has enough samples, we directly drop all those rows, and we still have 96.5784% data of the original dataset after dropping, which is enough for training models (19370 rows).

2.5 Get price on per sqm basis

We divided the resale price by the area to create a new feature 'resale_price_per_sqm', which is our new target variable of prediction.

3. Feature Engineering & Visualization (Zhang Tieyang, Li Rongrong)

3.1 Add transit time & distances to station / primary school

Since we already have the latitude and longitude of the nearest station and good primary school, we then used them to create new features. First, we calculate the Euclidean distance from a certain housing to its nearest station and popular primary school using their latitude and longitude. After that, we used the same API from OneMap to get the transit time, walking time, and walking distance as well, since sometimes, smaller Euclidean distance does not always mean shorter transit/walking time. Finally, we add these features to our dataset. Examples:

1) ***'min_dis_mrt_euclidean'***: Euclidean distance to the nearest station calculated from latitude and longitude.
2) ***'minMrt_transitTime'***: Time to reach the nearest station by public transportation.
3) ***'minMrt_walkTime'***: Time to walk to the nearest station.
4) ***'minMrt_walkDistance'***: Distance of walking to the nearest station.

minMrt_transitTime	minMrt_walkDistance	minMrt_walkTime	minPrimary_transitTime	minPrimary_walkDistance	minPrimary_walkTime	min_dis_mrt_euclidean	min_dis_primary_euclidean
451	1126	811	887	2192	1579	0.009096608	0.012239541
277	453	327	871	2692	1939	0.002826861	0.016418401
220	507	365	1231	2744	1976	0.003830421	0.017034574
220	507	365	1231	2744	1976	0.003830421	0.017034574

3.2 Drop useless column

For the future model training process, we need to drop some columns that has been used to generate new features, but is useless for training. So we dropped columns ‘month’, ‘street_name’, and ‘block’.

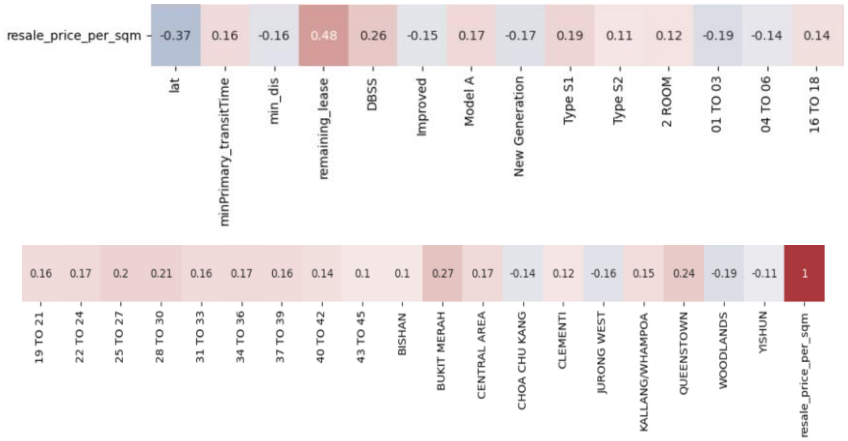
3.3 Data size and structure

Our final version of dataset has 19730 rows and 87 columns, most of which are dummy variables. Data types including int64, float64, and Boolean.

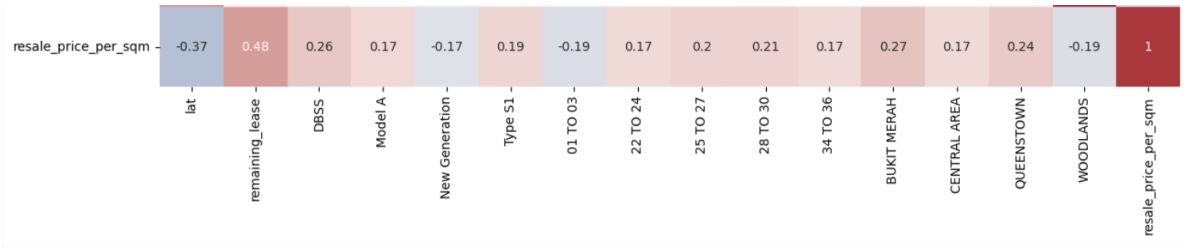
3.4 Covariance matrix and feature selection

Covariance is a statistical measure that quantifies how two variables change together, by generating covariance matrix for our dataset, we can visualize which variables are correlated with ‘resale_price_per_sqm’. We first noticed there are many features that has very little effect on price, and some of them are highly correlated with each other. Finally, Taking correlations of 0.1 and 0.168 (absolute value) as thresholds, we selected two sets of feature sets:

Feature set 1 with higher complexity, has **33 features** in total:

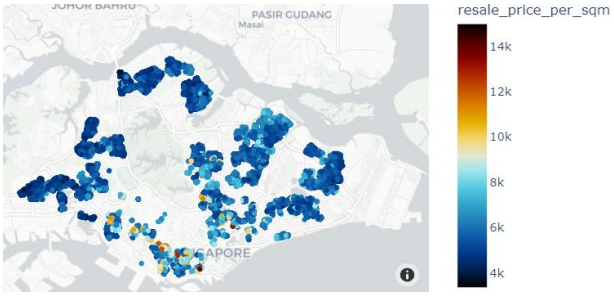


Feature set 2 with lower complexity, has **15 features** in total:



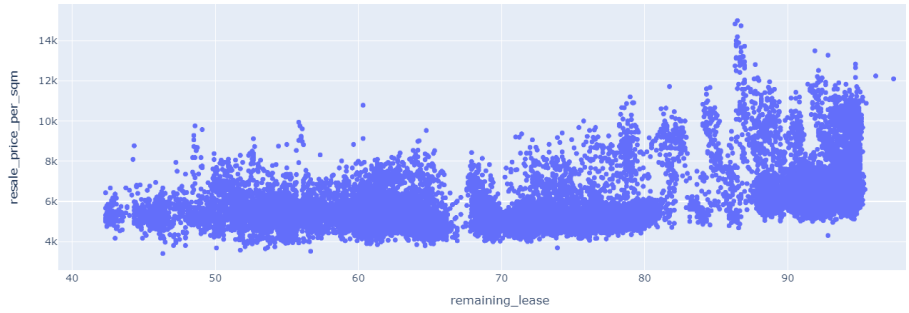
3.5 Map / Plot to verify the insights

Our new feature ‘lat’ does play a important role in predicting price per sqm, the lower the latitude, the higher the price per sqm; Vice versa.



Ordinary Linear Regression	MAE
33 Features include ‘lat’ & ‘minPrimary_transitTime’	506.50
32 Features exclude ‘lat’	550.99
31 Features exclude ‘lat’,‘minPrimary_transitTime’	551.04

Also, ‘remaining_lease’ is highly correlated with price per sqm. The longer the remaining lease, generally the higher the price per sqm.



4. Modelling & Optimization

We use feature set 1 with 33 features and feature set 2 with 16 features.

4.1.1 Linear Regression (Soo Yik Hong)

Pseudo code

1. Import training and testing data
2. Data pre-processing
   - 2.1 Separate target variable, Y (resale_price_per_sqm) from features in both datasets
   - 2.2 Extract categorical features in both training and testing datasets and convert TRUE/FALSE values to 1/0
   - 2.3 Calculate the mean and standard deviation of numerical columns (‘lat’, ‘minPrimary_transitTime’, ‘min_dis’, ‘remaining_lease’) in training dataset
   - 2.4 Standardize training and testing datasets (numerical columns only) by using the mean and standard deviation
   - 2.5 Recombine standardized numerical features with categorical features (1/0)
3. Train a linear regression model by using the scaled training data prepared in step 2
4. Use the trained model to make predictions on the scaled testing data
5. Calculate mean absolute error (MAE) and root mean squared error (RMSE)

Result

Table 1. Features with best linearity

Data Type	Column name	Estimate	Std. Error	t_value	Pr (> | t |)
numerical	Remaining_lease	805.1	7.795	103.288	< 2e-16
binary	40 TO 42	2706.8	171.031	15.827	< 2e-16

The linear regression model analysis revealed that the binary feature denoted as ‘40 TO 42’ possesses the largest coefficient of 2706.8 when its value is TRUE. Both features in the table above have the most significant positive effect on resale price, and they are statistically significant. The calculated RMSE of 670.4898 is indicative of the model’s predictive performance and signifies that there is room for improvement.

4.1.2 PCA + Linear Regression (Soo Yik Hong)

Pseudo code

1. Import PCA function, and fit it with scaled training data (numerical columns only)
   - 1.1 Determine the eigenvalues of each PCs
   - 1.2 Select the number of PCs based on variance coverage (95%)
2. Based on selected PCs, transform scaled training and testing data
3. Train a linear regression model by using the PCA transformed scaled training data prepared in step 4.

Result

Table 2. Variance ratio of each PC

	PC 1	PC 2	PC 3	PC 4

Variance ratio	0.35259718	0.25444489	0.23506246	0.15789547

Based on table 2 above, the minimum principal components used for regression modelling must be 4 to cover more than 95% of the variance in the training data. The PCA implies that the numerical columns: 'lat', 'minPrimary_transitTime', 'min_dis' and 'remaining_lease' are equally important and cannot be reduced dimensionally. In addition, the RMSE will be larger if fewer principal components (<4) are used for modelling.

4.1.3 Linear Regression + Ridge/Lasso Regression (Soo Yik Hong)

Pseudo code

1. Create a new column that takes log-based (Y)
2. Determine the maximum lambda for Ridge regression
   2.1 Transpose scaled training data (X)
   2.2 Perform matrix multiplication between transposed X and log-based (Y)
   2.3 Absolute the matrix, and find maximum value within the matrix
3. Setup a grid of lambda from 1e-6 to max lambda with 5000 steps
4. Train Ridge regression with cross validation in the grid (1 lambda = 1 model), and determine the best lambda with lowest cross validation error
5. Re-train the Ridge regression with the best lambda value
6. Use the re-trained model to make predictions on the scaled testing data
7. Convert predicted value (Y) back to normal value, ridge_pred = exp(ridge_pred)
8. Calculate root mean squared error (RMSE) between predicted values (Y) and actual testing data (Y)

Result

Table 3. Ridge parameters

	Value
Max. λ	1396
Best. λ	1.6755

The Ridge regression model demonstrates a 9.6% improvement over ordinary linear regression, addressing multicollinearity through pre-processing. Removal of highly correlated features reduces L2 penalty effect, yielding a modest enhancement with an RMSE of 650.4424. Lasso regression, employing same set of lambda values, achieves a comparable RMSE (650.9404) but exhibits overfitting with the smallest lambda (1e-6). The L1 penalty forces some coefficients to zero, beneficial for high-dimensional data but unsuitable here.

Conclusion

Model	Test Error (RMSE)
Linear Regression	670.4898
PCA + Linear Regression	670.4898
Linear Regression + Lasso regularization	650.9404
Linear Regression + Ridge regularization	650.4424

The best model for the project is the combination of linear regression with Ridge regularization, as it exhibits a balance by effectively reducing the multicollinearity between numerical and binary categorical columns, resulting in improved performance without overfitting.

4.2 Decision tree (Li Rongrong)

According to code results in process trying different models to fit, it was found that tree-based models performance were better than linear models generally. To select which model is the most appropriate, we started from decision tree. In the *sklearn* library, many parameters of the tree can be adjusted:

ccp_alpha: parameters for cost complexity pruning

max_depth: maximum depth of the tree

max_features: maximum number of features to consider when splitting each node

max_leaf_nodes: maximum number of leaf nodes allowed

min_samples_leaf: minimum number of samples on leaf nodes

Using bagging to increase effect of decision tree is also a good choice. Thanks to convenience of the *sklearn* library, we could building and try it easily. Because parameters between decision tree and bagging are highly similar, we do not

introduce them again. To avoiding overfitting, we used 5-fold cross-validation during building all tree-based models.

After many adjustments, the best parameter and error are shown in the table below.

Model	The best parameters set	MAE
Decision tree	{'max_depth': [15], 'min_samples_split': [17], 'min_samples_leaf': [2], 'ccp_alpha':[0.1], 'random_state':[817]}	323.2293
Bagging	{"n_estimators":[285], "max_features":[28], 'random_state': [817]}	271.6265

4.3 Random forest (Li Rongrong)

Random forest is convergence of different decision tree, a special situation of bagging. If bagging perform well, random forest can be predicted having great effect. Like decision tree, *sklearn* library also provide many parameters for users to adjust. Because these parameters are highly same as Decision tree, here we only introduce some features that are different from decision trees:

bootstrap: specify whether to use bootstrap sampling for training

max_depth: the maximum depth of each tree contained in a random forest is also the same as a single decision tree.

n_estimators: number of base decision trees included in the random forest

In the table showing improvement process, all parameters adjusted are displayed.

Model	The best parameters set	MAE
Random forest	{"n_estimators":[290], "max_features":[21], 'ccp_alpha': [0.2],'max_depth': [21], 'random_state': [817] }	271.1573

4.5 Neural Network: Back Propagation – Mini-Batch Gradient Descent (Zhang Tieyang)

In theory, neural networks have the capability to approximate almost any function. However, such a powerful ability comes at the cost of sacrificing interpretability and longer training times.

For this project, also as the first time working on NN, we visited the website for TensorFlow v2.14.0 at https://www.tensorflow.org/api_docs and looked up for all the available parameters. Finally, we decided to attempt training the NN using stochastic descent algorithm with mini-batches in the backpropagation process.

Our NN has 1 input layer, 1 hidden layer, and 1 output layer, the parameters we decide to tune for it are shown below:

Optimizer: the algorithm we used for training, we only tried 2 of them: '*SGD' & '*Adam'

Metrics: the loss function (calculate the gradients) we used for training, we only tried 2 of them: 'mae' & 'mse'

Units: the # of unit of hidden layer, we tried 7, 8, 9

Batch_size: the # of batches we used for each epoch, we tried 16, 32, 64

Learning_rate: rate for both optimizers to change the weights in NN, we tried 0.008, 0.01 for 'sgd' + 'mae'; 0.0008, 0.001 for 'adam' + 'mse'

Model: we tried feature group with size 33 & 15

Activation function: always use 'relu'

The best result appears when using sgd + mae with model of 33 features. We then record the best model with 8 units for hidden layer, 32 batches for each epoch, and learning rate of 0.01 using sgd algorithm and mean absolute error as metric. Store this model into 'best_model.h5'.

units	batch_size	learning_rate	minimum_mae_error	R^2
8	32	0.01	381.7575	0.842157

5. Prediction & Interpretation (Zhang Tieyang, Li Rongrong)

5.1 Comparison between best models

Validation set approach:

	MAE	R^2
Ordinary Linear regression	506.496	0.757
Linear Regression - Ridge	509.611	0.754
Linear Regression - Lasso	509.632	0.754
Decision Tree	349.386	0.871
Bagging	280.463	0.921
Random Forest	280.729	0.921
FNN(Mini Batch-SGD)	381.757	0.842

Cross-Validation (5-fold):

	MAE
Linear Regression - Ridge	508.593
Linear Regression - Lasso	508.623
Decision Tree	351.592
Bagging	288.216
Random Forest	284.332

As a result, we concluded that Random Forest might be the best model for this project.

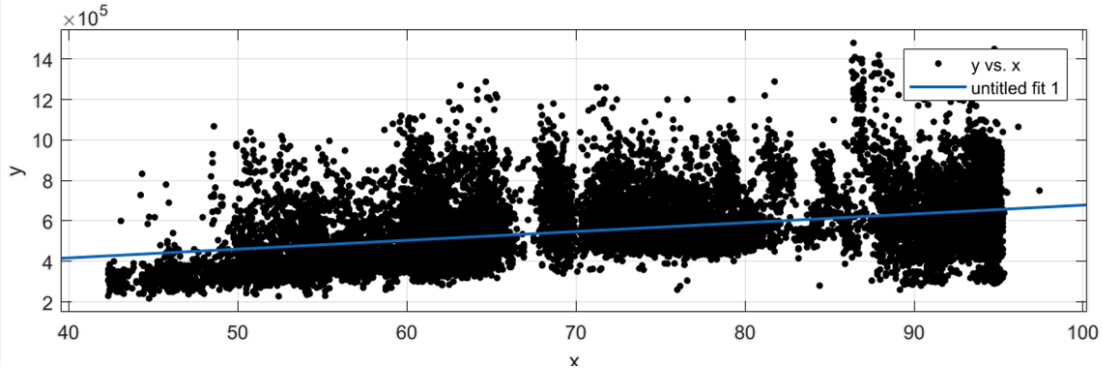
5.2 Discovery on features importance

For the question "Adding which features could improve the most when using only 5 features (whose covariance > 0.3) fitting model", if it only allowed to add one variable, it would be '01 TO 03', a dummy variable converted from storey range. If it only allowed to add two variables, it would be '01 TO 03' and '04 TO 06'. This shows that apart from the features that have the greatest influence to resale price, the condition of the HDB flat itself has a greater impact on its selling price than external factors.

This discovery can be easy to explain: when people make decision on purchasing what kinds of flats, they firstly care about flats themselves, such as their condition, their floor and their available area. After that, people will pay attention to other factors like flats location to judge whether living in these flats is convenient.

5.3 Something unreasonable but appears

Of course. As is known to us all, the longer remaining lease is, the higher price buyer should pay. But based on this data set, thing doesn't seem to be the case. (picture processed by Matlab)



It is obvious that when the remaining lease exceeds a certain point between 80 and 90, the position of the dense area of the scatter plot does not rise but falls. Figures from code also show the same result: R-square of 2 degree polynomial (0.1925) is larger than linear model (0.1529). After investigation, the reason for this phenomenon is that Singapore's policy stipulates that the sum of the remaining lease and the age of the purchaser cannot exceed a certain integer. Therefore, when remaining lease increasing to a high level, HDB must reduce its price to make sure someone is willing to buy it.

APPENDIX

Which primary schools are "great"?

Singapore's policy is to determine the primary school that children should attend according to the town. Naturally, parents want their children to go to the best school and they would rather purchase HDB flats near great primary schools. But which primary schools can be defined as "great"? We need a specific concept, or a numeric judgement to select primary schools. According to government policy, there are three phase of registration for primary schools in Singapore and every phase has unique requirements.

Phase A: Children whose siblings are studying will be given priority.
Phase B: Children's parents who have a relationship with the school or have special social status.
Phase C: All citizens and non-PR international students.

About phases B and C, there is a ratio of the number of applicants admitted to the number of planned admissions in the related website. This percentage can indirectly reflect the popularity of the school, which could be regarded as a concept of "great". We obtained 56 primary schools that not only appears in the top part of Phase B but also in Phase C and regard them as "great" primary schools.