# Introduction

The goal of the competition was to predict housing prices in Singapore. Two different datasets were provided to us – Private housing & HDB public housing. The target was to predict the resale prices for test sets of these two datasets. The evaluation criteria used is MAPE – Mean Average Percentage Error. In the following sections, we will be presenting the approach we followed to tackle this competition and the results obtained.

# Data Set Exploration and Feature Engineering

## HDB Training Data

* Contains houses data sold from early 1990s to 2017
* The features provided are shown in Table 1 of Appendix
* Contains features that are categorical as well as numerical
* Data appears to have some outliers
* Data is reasonably clean

## Private Housing Training Data

* Contains houses data sold from early 1990s to 2017
* Contains features that are categorical as well as numerical
* The features provided are shown in Table 1 of Appendix
* Contains some outliers (e.g. some houses with large floor area, but low price).
* Contains dirty data (e.g. tenure descriptions are incomplete)

## Test Data

* Both data sets are targeted at predicting the resale price of houses sold in 2017.
* Data is reasonably clean

## Training Data Selection

* Since test data set is aiming at housing prices in 2017, we decided to train our models with data from 2015, 2016, 2017. Our assumption is that buyers and sellers do not need to know old house price. But they need to know house prices that are based on recent house prices in current market so that they can buy or sell at competitive price range.
* For private housing dataset, we noticed that test data set is aiming at only Resale, so we initially decided to exclude all other sale types from our training data. But When we evaluated our models in Kaggle, we were not getting good results compared to when we had included all types of sale data. Hence, we went back to considering the private data for past 3 years.

## Data Preparation

* Both Label Encoding as well as Feature Hashing are used to represent categorical features (e.g. area, storey range, property type)
* One Hot Encoding is not used because it generates too many feature columns that computation time takes extremely long. Instead, Feature Hashing is used.
* Since house price is positively skewed in both the data sets, we applied BoxCox transformation on target feature to achieve a reasonable normal distribution. This can be seen in the figure 1 below.
* Like in the case of target variable, we tried performing BoxCox transformation on the positively skewed features of both datasets, but we saw that this did not help us to reduce error. Hence it was not used.

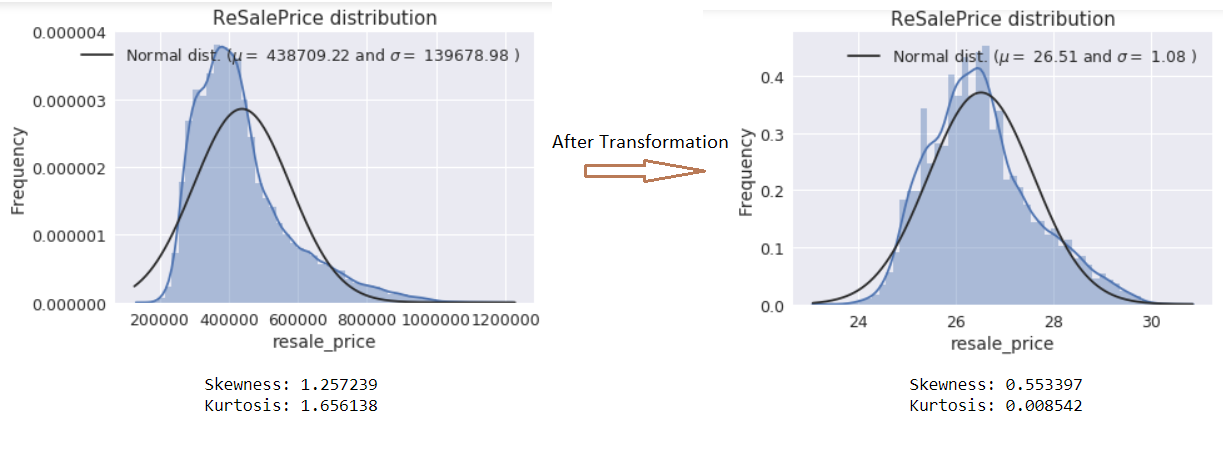


Figure 1.1: BoxCox transformation on HDB data with lamda=0.1

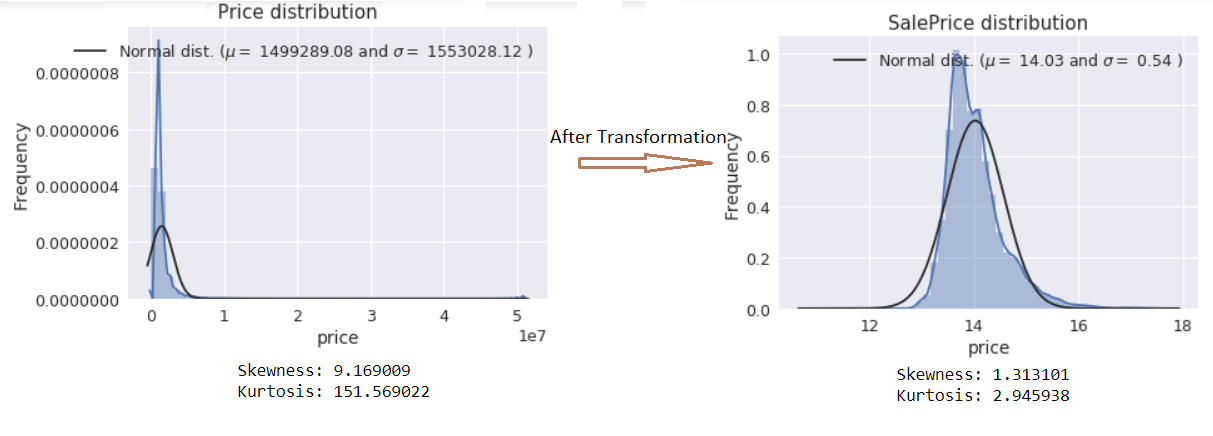
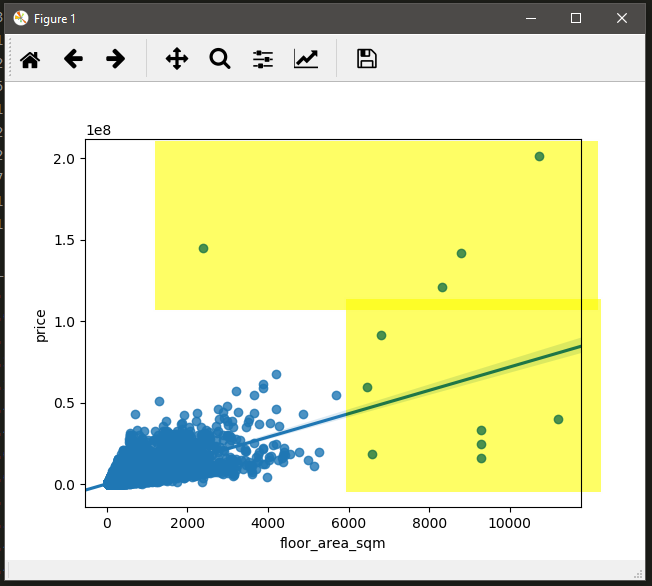


Figure 1.2: BoxCox transformation on Private data with lamda=0 (log transformation)

* There were outliers in both HDB and Private datasets. these records are removed from training data set. The figure 2 below provides the description about the same. For Private the outliers were found for floor\_area\_sqm where as for HDB it was seen in latitude and longitude features.

Geo Plot using Lat long feature for HDB Scatter plot - floor area vs price for private

Figure 2: Outliers highlighted for HDB (left – in circle) and Private (right – yellow shaded) dataset.

## Feature Engineering and Selection – HDB Data

* For HDB data the features provided were reasonably clean and hence we did not feel the need to perform extensive feature engineering on them.
* We rely on the Pearson correlation characteristics of training data sets to identify potential features (Figure 3 of appendix). We also identified the rank of each features with respect to correlation with resale price. Table 2 of appendix provides information on the same.
* When we arrived at selecting models for training, we found that the boosted trees models of GradientBoostingRegresor (GBR) and XGBRegressor (XGB) to perform best. Since these models have inherent property of identifying feature importance, we did not remove any colinearly related features, but we let the algorithms discover the feature importance themselves

## Feature Engineering and Selection – Private Data

For Private data it was seen that lot of the features had dirty data. So many of the features were transformed.

* The address feature was parsed into two features - address\_block and address\_street. This was done since we observed in HDB case, these two features were leading to good results. The records where the address was not properly captured were discarded.
* The tenure feature had lot of incorrect data for tenure start date. So instead of considering the tenure start we considering only the tenure duration.
* The unit and floor feature values were missing for almost 15% of the data. We believed unit feature to be not significant hence it was dropped but the floor feature would be significant and hence the missing data needed to be imputed.
* We saw that the floor values were missing mostly for records belonging to individual houses or mansion and the associated prices will be on the higher side. The mean of price for these records was found to be 3.8 \* 10^6. We then identified the range for the floors by taking the records with floor values and having price value around this mean. The floor range was approximated to 10-15. We then ran our models with different value for floor in this range and saw that decent results were found for floor value of 10. Hence the missing floor value was replaced with 10.
* Similar in the case of HDB, we calculated Pearson correlation co-efficient (figure 4) and ranking features with respect to correlation with resale price (table 2). But since XGB and GBR were ultimately chosen, the multicollinearity was not removed.

1. Evaluating Algorithms

* We evaluated a few candidate algorithms both Linear and Ensemble types
* Linear algorithms we evaluated from scikit-learn include -
  1. Linear Regression
  2. Lasso
  3. ElasticNet
  4. KneighborsRegressor
  5. DecisionTreeRegressor
  6. SVR
* Ensemble algorithms we evaluated from scikit-learn include –
  1. AdaBoostRegressor
  2. GradientBoostingRegressor
  3. RandomForestRegressor
  4. ExtraTreesRegressor
  5. XGBRegressor
* In general, Linear algorithms yield higher error rate than Ensemble algorithms. Figure 5 below shows a sample of evaluation over 20,000 records of HDB dataset. Same holds true for Private Housing dataset.
* Among Ensemble algorithms, RandomForest and ExtraTrees seems promising, but during training with larger dataset, their performance degrades.
* Only GradientBoostingRegresor (GBM) and XGBRegressor (XGB) yielded acceptable training time and comparatively good error rates

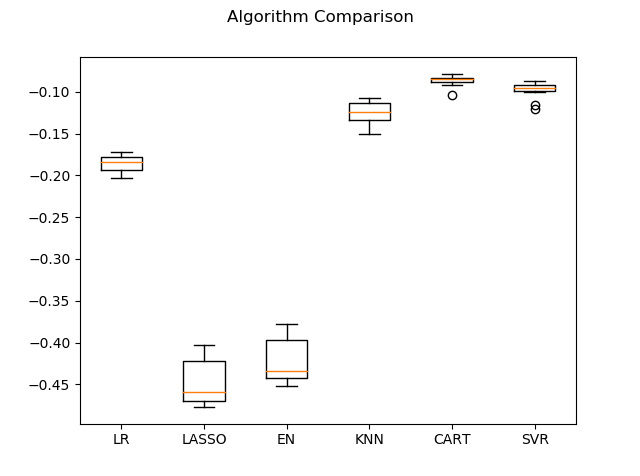
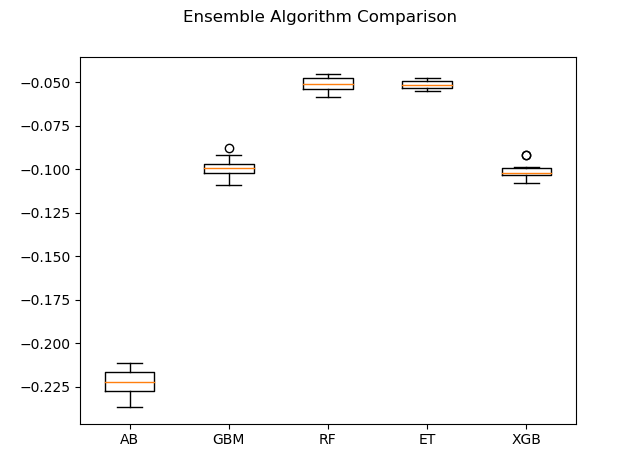
 

Figure 5: Algorithm Comparison for HDB dataset (~20K records)

## Tuning the models

* We used scikit-learn’s GridSearchCV and RandomizedSearchCV to find optimal parameters for the chosen two algorithms: GBM, and XGB. (Figure 6)
* However, it takes extremely long time for GBM to complete GridSearchCV or RandomizedSearchCV. So, we started to experiment with tuning parameters manually.

## Training the models

* KFold cross validation scoring is used for finding optimal model.
* Train-test-split approach (90 - 10) is used for training and predicting the test data.

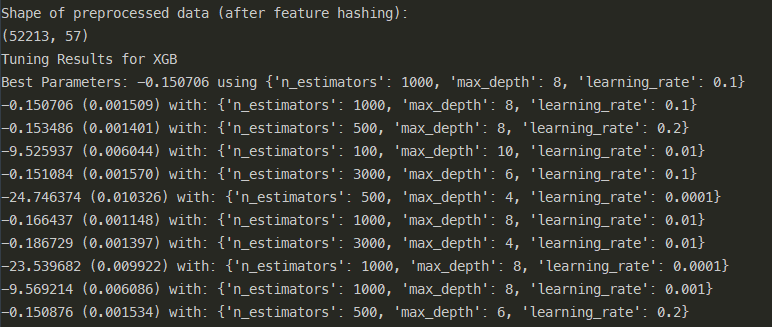


Figure 6: XGBRegressor best parameters found via RandomizedSearchCV (took 6 hours on Xeon workstation laptop to obtain this!)

1. Results Obtained

As mentioned earlier the best test split results were observed for XGBoost and GradientBoostingRegressor models. Hence ensemble of these two were tried for predicting results

* For HDB with weighted average ensemble of XGB and GBR were used to obtain best results. The weight for GBR (70%) was given more as it’s results was the better among the two. The best MAPE obtained is 3.65.
* For Private only GBR was found to give best results, Hence only GBR was used for prediction. The best MAPE obtained is 3.6.
* For Kaggle submission, the best MAPE we obtained was 5.27 in public leaderboard and 5.29 in private leaderboard which resulted for us to get the final rank of 21.

# Appendix: Screenshots of Data Exploration

|  |  |
| --- | --- |
| HDB | Private Housing |
| 1. block 2. flat\_model 3. flat\_type 4. floor\_area\_sqm 5. lease\_commence\_date 6. month 7. resale\_price - Target variable 8. storey\_range 9. street\_name 10. town 11. latitude 12. longitude 13. postal\_code 14. floor | 1. project\_name 2. address 3. floor\_area\_sqm 4. type\_of\_land 5. price 6. contract\_date 7. property\_type 8. tenure 9. completion\_date 10. type\_of\_sale 11. postal\_district 12. postal\_sector 13. postal\_code 14. region 15. area 16. month 17. latitude 18. longitude 19. floor\_num 20. unit\_num |

Table 1: List of available features for HDB and Private

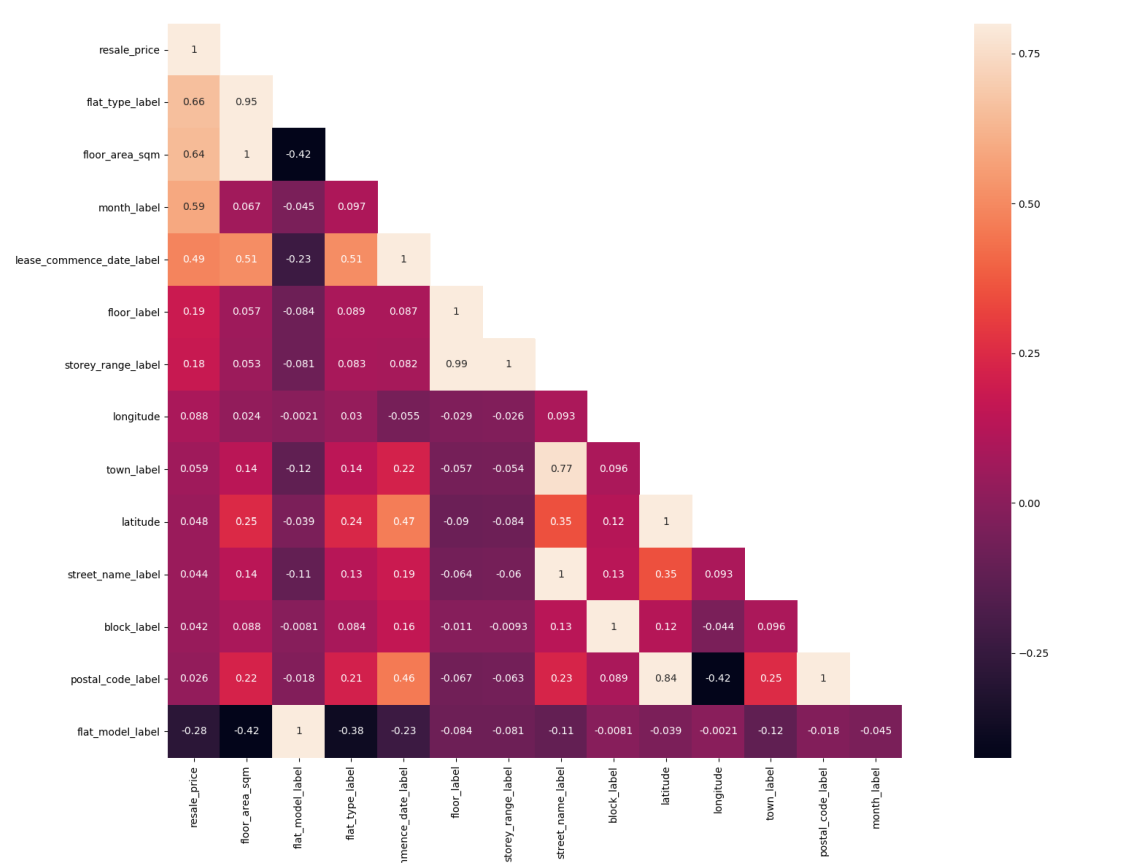


Figure 3 HDB Training Data Correlation

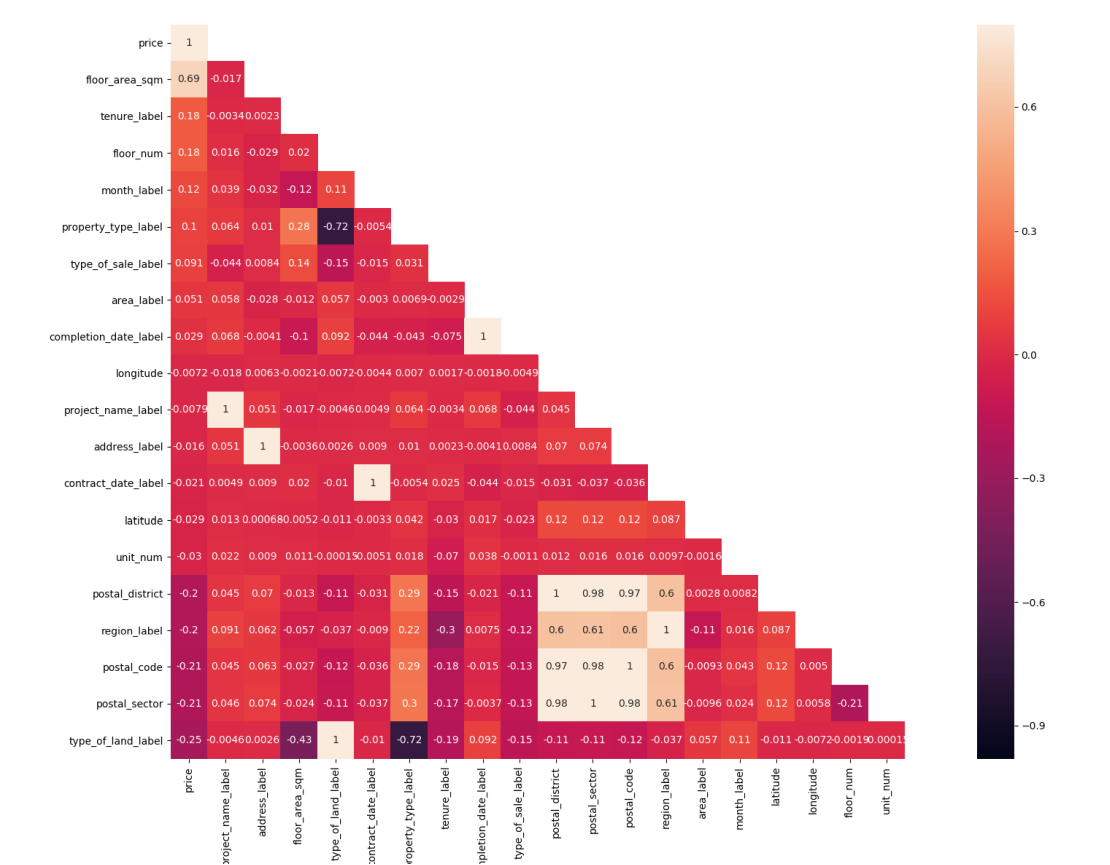


Figure 4 Private Housing Training Data Correlation

|  |  |
| --- | --- |
| HDB | Private Housing |
| 1. flat\_type 2. floor\_area\_sqm 3. month 4. lease\_commence\_date 5. floor 6. storey\_range 7. longitude 8. latitude 9. town 10. street\_name 11. block 12. postal\_code 13. flat\_model | 1. floor\_area\_sqm 2. tenure 3. type\_of\_sale 4. property\_type 5. floor\_num 6. address\_block(derived from address) 7. month 8. area 9. contract\_date 10. longitude 11. address\_street (derived from address) 12. project\_name 13. unit\_num 14. completion\_date 15. postal\_district 16. postal\_sector 17. postal\_code 18. latitude 19. region |

Table 2: Features ranked (in descending order) for HDB and Private based on co relation with price

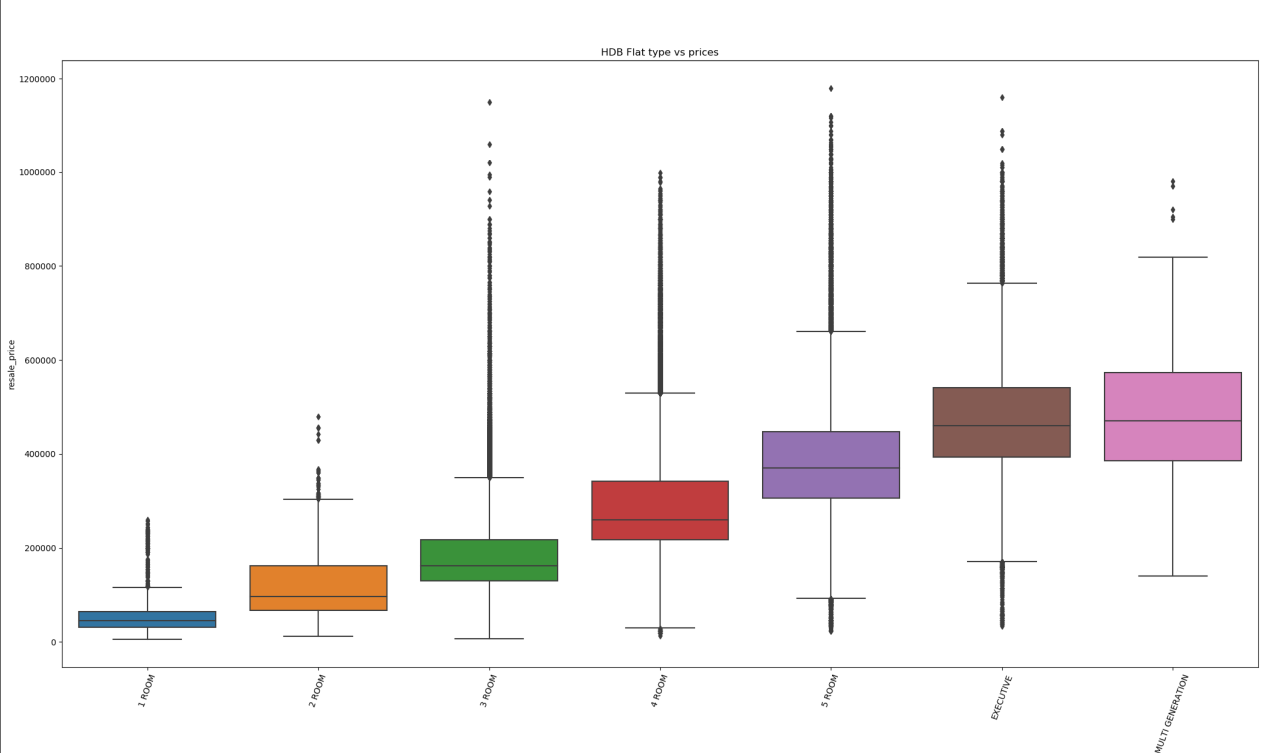


Figure 7: HDB Flat Type vs Price (left to right: 1 room, 2 room, etc)

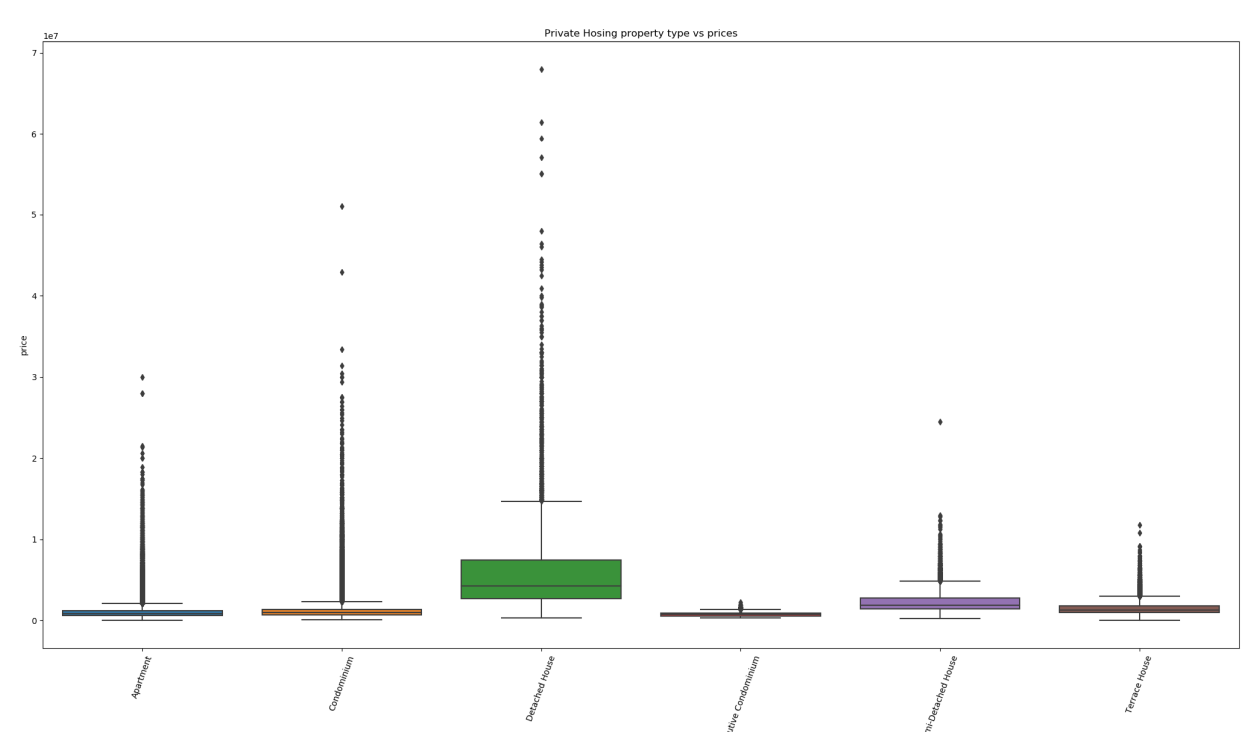


Figure 8: Private Housing property type vs Price

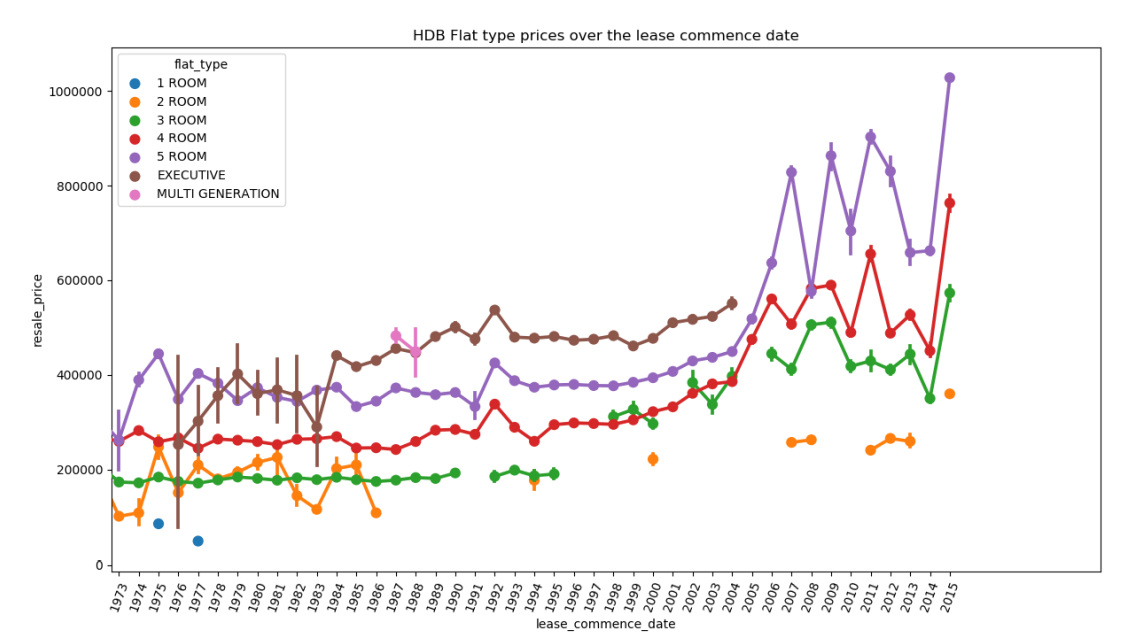


Figure 9: HDB flat type prices over the years

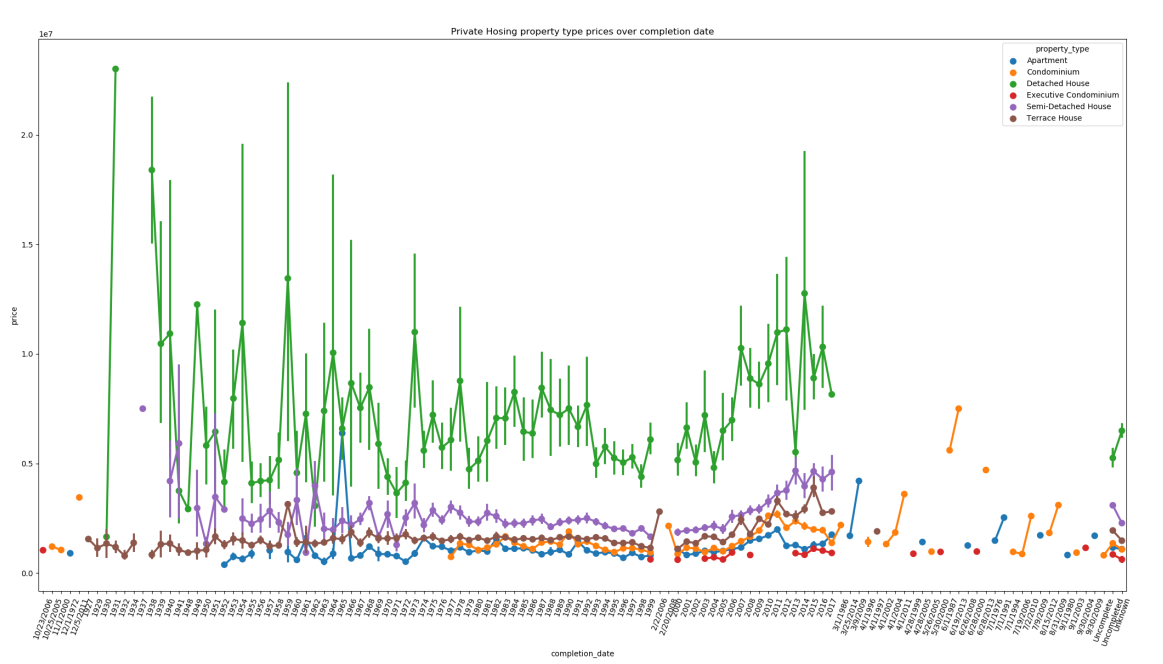


Figure 10: Private Housing property type prices over the years (green colour shows “detached house” type)