Assignment-1,Phase- II

## 1 Introduction

The objective of this project was to build regression model’s to predict house prices in Melbourne which is a dependent variable. House price’s depends on various factors such as suburb, distance from CBD, rooms, number of bedrooms to name a few, which are independent variables. Therefore this is a regression predictive model.The data has been sourced from Kaggle (Melbourne Housing Market), which in turn has been scraped from Domain.com.au.

In Phase I, the data was cleaned and the descriptive features were re-categorised to be less granular. In Phase II, three regression models has been built on the cleaned data. The rest of this report is organised as follow. Section 2 describes an overview of the methodology. Section 3 discusses the regressors’ fine-tuning process and detailed performance analysis of each regressor.Section 4 compares the performance of the regressors using the same resampling method. Section 5 critiques the methodology. The last section concludes with a summary.

# 1.1 Reading the Data in :

## Loading required package: ParamHelpers

## Warning: replacing previous import 'BBmisc::isFALSE' by  
## 'backports::isFALSE' when loading 'mlr'

## -- Attaching packages ----------------------------------------------------------------- tidyverse 1.2.1 --

## v ggplot2 2.2.1 v purrr 0.2.4  
## v tibble 1.4.2 v dplyr 0.7.4  
## v tidyr 0.8.0 v stringr 1.3.0  
## v readr 1.1.1 v forcats 0.3.0

## -- Conflicts -------------------------------------------------------------------- tidyverse\_conflicts() --  
## x dplyr::filter() masks stats::filter()  
## x dplyr::lag() masks stats::lag()

##   
## Attaching package: 'data.table'

## The following objects are masked from 'package:dplyr':  
##   
## between, first, last

## The following object is masked from 'package:purrr':  
##   
## transpose

##   
## Attaching package: 'lubridate'

## The following objects are masked from 'package:data.table':  
##   
## hour, isoweek, mday, minute, month, quarter, second, wday,  
## week, yday, year

## The following object is masked from 'package:base':  
##   
## date

##   
## Attaching package: 'scales'

## The following object is masked from 'package:purrr':  
##   
## discard

## The following object is masked from 'package:readr':  
##   
## col\_factor

## corrplot 0.84 loaded

## Warning: package 'spFSR' was built under R version 3.4.4

## Loading required package: parallelMap

## Loading required package: parallel

## Loading required package: tictoc

## Warning: package 'kknn' was built under R version 3.4.4

# 1.2 Data cleaning and imputation

## 'data.frame': 11517 obs. of 20 variables:  
## $ Suburb : Factor w/ 319 levels "Abbotsford","Aberfeldie",..: 1 1 1 1 1 1 1 1 1 1 ...  
## $ Address : Factor w/ 11315 levels "1 Abercrombie St",..: 5035 8269 8907 2742 1750 1719 2987 11288 870 3701 ...  
## $ Rooms : int 2 3 4 4 3 2 4 2 2 2 ...  
## $ Type : Factor w/ 3 levels "h","t","u": 1 1 1 1 1 3 1 1 1 1 ...  
## $ Price : int 1035000 1465000 1600000 NA 1876000 NA NA 1636000 1097000 NA ...  
## $ Method : Factor w/ 9 levels "PI","PN","S",..: 3 6 8 5 3 1 9 3 3 6 ...  
## $ SellerG : Factor w/ 276 levels "@Realty","Abercromby's",..: 23 23 173 173 173 23 23 173 23 23 ...  
## $ Distance : num 2.5 2.5 2.5 2.5 2.5 2.5 2.5 2.5 2.5 2.5 ...  
## $ Postcode : int 3067 3067 3067 3067 3067 3067 3067 3067 3067 3067 ...  
## $ Bedroom2 : int 2 3 3 3 4 2 6 2 3 2 ...  
## $ Bathroom : int 1 2 1 2 2 2 2 1 1 1 ...  
## $ Car : int 0 0 2 2 0 1 0 2 2 1 ...  
## $ Landsize : int 156 134 120 400 245 4292 230 256 220 176 ...  
## $ BuildingArea : num 79 150 142 220 210 82 147 107 75 80 ...  
## $ YearBuilt : int 1900 1900 2014 2006 1910 2009 1860 1890 1900 1925 ...  
## $ CouncilArea : Factor w/ 33 levels "Banyule City Council",..: 32 32 32 32 32 32 32 32 32 32 ...  
## $ Lattitude : num -37.8 -37.8 -37.8 -37.8 -37.8 ...  
## $ Longtitude : num 145 145 145 145 145 ...  
## $ Regionname : Factor w/ 8 levels "Eastern Metropolitan",..: 3 3 3 3 3 3 3 3 3 3 ...  
## $ Propertycount: int 4019 4019 4019 4019 4019 4019 4019 4019 4019 4019 ...

# 1.3 Summarizing the Columns

Feature Summary before pre processing

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| name | type | na | mean | disp | median | mad | min | max | nlevs |
| Suburb | factor | 0 | NA | 9.804637e-01 | NA | NA | 1.00000 | 225.0000 | 319 |
| Address | factor | 0 | NA | 9.996527e-01 | NA | NA | 1.00000 | 4.0000 | 11315 |
| Rooms | integer | 0 | 3.141530e+00 | 9.796162e-01 | 3.00000 | 1.482600e+00 | 1.00000 | 12.0000 | 0 |
| Type | factor | 0 | NA | 2.420769e-01 | NA | NA | 902.00000 | 8729.0000 | 3 |
| Price | integer | 2630 | 1.092902e+06 | 6.793819e+05 | 900000.00000 | 4.670190e+05 | 131000.00000 | 9000000.0000 | 0 |
| Method | factor | 0 | NA | 4.463836e-01 | NA | NA | 7.00000 | 6376.0000 | 9 |
| SellerG | factor | 0 | NA | 8.989320e-01 | NA | NA | 1.00000 | 1164.0000 | 276 |
| Distance | numeric | 0 | 1.108401e+01 | 6.808435e+00 | 9.90000 | 5.633880e+00 | 0.00000 | 48.1000 | 0 |
| Postcode | integer | 0 | 3.114582e+03 | 1.111806e+02 | 3101.00000 | 8.154300e+01 | 3000.00000 | 3977.0000 | 0 |
| Bedroom2 | integer | 0 | 3.116871e+00 | 9.812718e-01 | 3.00000 | 1.482600e+00 | 0.00000 | 12.0000 | 0 |
| Bathroom | integer | 0 | 1.681775e+00 | 7.426089e-01 | 2.00000 | 1.482600e+00 | 1.00000 | 9.0000 | 0 |
| Car | integer | 0 | 1.700703e+00 | 1.007583e+00 | 2.00000 | 1.482600e+00 | 0.00000 | 26.0000 | 0 |
| Landsize | integer | 0 | 5.282349e+02 | 1.012193e+03 | 489.00000 | 3.128286e+02 | 0.00000 | 42800.0000 | 0 |
| BuildingArea | numeric | 0 | 1.533866e+02 | 8.797206e+01 | 135.00000 | 5.930400e+01 | 0.00000 | 3112.0000 | 0 |
| YearBuilt | integer | 0 | 1.964376e+03 | 3.759271e+01 | 1970.00000 | 4.447800e+01 | 1196.00000 | 2106.0000 | 0 |
| CouncilArea | factor | 0 | NA | 8.924199e-01 | NA | NA | 1.00000 | 1239.0000 | 33 |
| Lattitude | numeric | 0 | -3.780878e+01 | 8.893860e-02 | -37.80648 | 7.920050e-02 | -38.17928 | -37.3902 | 0 |
| Longtitude | numeric | 0 | 1.449972e+02 | 1.185898e-01 | 145.00290 | 1.028776e-01 | 144.42379 | 145.5264 | 0 |
| Regionname | factor | 0 | NA | 6.662325e-01 | NA | NA | 52.00000 | 3844.0000 | 8 |
| Propertycount | integer | 0 | 7.485988e+03 | 4.307225e+03 | 6567.00000 | 3.994124e+03 | 249.00000 | 21650.0000 | 0 |

# 1.4 Take out the NA values in dataset

## [1] "Suburb" "Address" "Rooms" "Type"   
## [5] "Price" "Method" "SellerG" "Distance"   
## [9] "Postcode" "Bedroom2" "Bathroom" "Car"   
## [13] "Landsize" "BuildingArea" "YearBuilt" "CouncilArea"   
## [17] "Lattitude" "Longtitude" "Regionname" "Propertycount"

## 'data.frame': 8887 obs. of 20 variables:  
## $ Suburb : Factor w/ 319 levels "Abbotsford","Aberfeldie",..: 1 1 1 1 1 1 1 1 1 1 ...  
## $ Address : Factor w/ 11315 levels "1 Abercrombie St",..: 5035 8269 8907 1750 11288 870 7639 1741 2801 7770 ...  
## $ Rooms : int 2 3 4 3 2 2 3 2 2 3 ...  
## $ Type : Factor w/ 3 levels "h","t","u": 1 1 1 1 1 1 1 3 1 1 ...  
## $ Price : int 1035000 1465000 1600000 1876000 1636000 1097000 1350000 750000 1310000 1200000 ...  
## $ Method : Factor w/ 9 levels "PI","PN","S",..: 3 6 8 3 3 3 8 3 3 3 ...  
## $ SellerG : Factor w/ 276 levels "@Realty","Abercromby's",..: 23 23 173 173 173 23 173 23 120 120 ...  
## $ Distance : num 2.5 2.5 2.5 2.5 2.5 2.5 2.5 2.5 2.5 2.5 ...  
## $ Postcode : Factor w/ 194 levels "3000","3002",..: 53 53 53 53 53 53 53 53 53 53 ...  
## $ Bedroom2 : int 2 3 3 4 2 3 3 2 2 3 ...  
## $ Bathroom : int 1 2 1 2 1 1 2 2 1 2 ...  
## $ Car : int 0 0 2 0 2 2 2 1 2 1 ...  
## $ Landsize : int 156 134 120 245 256 220 214 0 238 113 ...  
## $ BuildingArea : num 79 150 142 210 107 75 190 94 97 110 ...  
## $ YearBuilt : int 1900 1900 2014 1910 1890 1900 2005 2009 1890 1880 ...  
## $ CouncilArea : Factor w/ 33 levels "Banyule City Council",..: 32 32 32 32 32 32 32 32 32 32 ...  
## $ Lattitude : num -37.8 -37.8 -37.8 -37.8 -37.8 ...  
## $ Longtitude : num 145 145 145 145 145 ...  
## $ Regionname : Factor w/ 8 levels "Eastern Metropolitan",..: 3 3 3 3 3 3 3 3 3 3 ...  
## $ Propertycount: int 4019 4019 4019 4019 4019 4019 4019 4019 4019 4019 ...  
## - attr(\*, "na.action")=Class 'omit' Named int [1:2630] 4 6 7 10 14 19 34 61 63 67 ...  
## .. ..- attr(\*, "names")= chr [1:2630] "4" "6" "7" "10" ...

# 1.5 Feature Selection and which all variables are to be used for modelling

## 'data.frame': 8887 obs. of 15 variables:  
## $ Suburb : Factor w/ 319 levels "Abbotsford","Aberfeldie",..: 1 1 1 1 1 1 1 1 1 1 ...  
## $ Rooms : int 2 3 4 3 2 2 3 2 2 3 ...  
## $ Type : Factor w/ 3 levels "h","t","u": 1 1 1 1 1 1 1 3 1 1 ...  
## $ Price : int 1035000 1465000 1600000 1876000 1636000 1097000 1350000 750000 1310000 1200000 ...  
## $ Method : Factor w/ 9 levels "PI","PN","S",..: 3 6 8 3 3 3 8 3 3 3 ...  
## $ Distance : num 2.5 2.5 2.5 2.5 2.5 2.5 2.5 2.5 2.5 2.5 ...  
## $ Bedroom2 : int 2 3 3 4 2 3 3 2 2 3 ...  
## $ Bathroom : int 1 2 1 2 1 1 2 2 1 2 ...  
## $ Car : int 0 0 2 0 2 2 2 1 2 1 ...  
## $ Landsize : int 156 134 120 245 256 220 214 0 238 113 ...  
## $ BuildingArea : num 79 150 142 210 107 75 190 94 97 110 ...  
## $ YearBuilt : int 1900 1900 2014 1910 1890 1900 2005 2009 1890 1880 ...  
## $ CouncilArea : Factor w/ 33 levels "Banyule City Council",..: 32 32 32 32 32 32 32 32 32 32 ...  
## $ Regionname : Factor w/ 8 levels "Eastern Metropolitan",..: 3 3 3 3 3 3 3 3 3 3 ...  
## $ Propertycount: int 4019 4019 4019 4019 4019 4019 4019 4019 4019 4019 ...

The attributes Address, SellerG, Postcode,Latitude and Longitude have been removed due to high number of levels. For region, Council Area and Region name have been considered.

## 2 Methodology

## 2.1 Feature Selection

Three regression models have been considered. Linear Regression(LM),K-Nearest Neighbour(KNN),Random Forest(RF). Each regression model has been trained to make prediction. The data has been split into 80% training set and 20% test set. Fine tuning process has been ran and stratified sampling has been used to cater to the imbalance class of the target feature. The use of normalization has been done too on the features. Using the selected best features from the data set and using tuned hyperparametrs, predictions were made on the test data set.Model training (hyperparameter tuning), relied on rmse (root mean squared error).In addition to rmse,sse, mse, rsq and mae values were also calculated.The modelling has been implemented in R using the mlr package and the feature selection has been done using spFSR package.

## Warning in makeTask(type = type, data = data, weights = weights, blocking =  
## blocking, : Empty factor levels were dropped for columns: Suburb,Method

## Warning in spsaKernel(task = task, wrapper = wrapper, measure = measure, :   
## Stratification for learner of type regr is not supported.

## Warning in spsaKernel(task = task, wrapper = wrapper, measure = measure, :   
## cv.stratify is reset to FALSE.

## SPSA-FSR begins:

## Wrapper = kknn

## Measure = rmse

## Number of selected features = 0

##   
## iter value st.dev num.ft best.value  
## 1 400779.34082 27843.91484 6 400779.3 \*  
## 2 496821.97029 23660.75974 6 400779.3  
## 3 520366.94588 46754.04638 3 400779.3  
## 4 432069.72663 25989.6791 8 400779.3  
## 5 406748.02676 25926.27369 7 400779.3  
## 6 419112.46473 32004.69039 6 400779.3  
## 7 423353.77189 28005.10035 6 400779.3  
## 8 673792.28959 21845.3379 6 400779.3  
## 9 474715.15957 33981.44656 8 400779.3  
## 10 612986.85844 25220.50565 3 400779.3  
## 11 482145.6572 26737.1908 8 400779.3  
## 12 543288.78193 23182.44675 7 400779.3  
## 13 528072.58552 33237.60553 6 400779.3  
## 14 421389.08838 27417.16954 8 400779.3  
## 15 398753.81454 30338.0068 10 398753.8 \*  
## 16 566556.25189 25537.13903 7 398753.8  
## 17 453901.68888 22910.07682 8 398753.8  
## 18 545115.14846 27605.03312 9 398753.8  
## 19 444847.75288 30706.70158 7 398753.8  
## 20 462998.82456 35471.8218 5 398753.8  
## 21 562211.47185 27585.21114 7 398753.8  
## 22 415443.40473 30855.33625 8 398753.8  
## 23 459962.34044 32116.96338 10 398753.8  
## 24 489200.33403 28864.05086 8 398753.8  
## 25 519698.46616 28759.10835 7 398753.8  
## 26 489495.01603 34905.11833 7 398753.8  
## 27 450789.75534 19982.77354 6 398753.8  
## 28 451367.9022 31066.7305 8 398753.8  
## 29 492952.32163 26885.592 7 398753.8  
## 30 755079.99203 34205.15335 3 398753.8  
## 31 482240.45092 33838.98778 8 398753.8  
## 32 468037.0201 22400.3401 10 398753.8  
## 33 426934.2053 33717.11765 8 398753.8  
## 34 588264.48139 28354.33913 6 398753.8  
## 35 513396.29288 26069.96578 5 398753.8

##   
## Best iteration = 15

## Number of selected features = 10

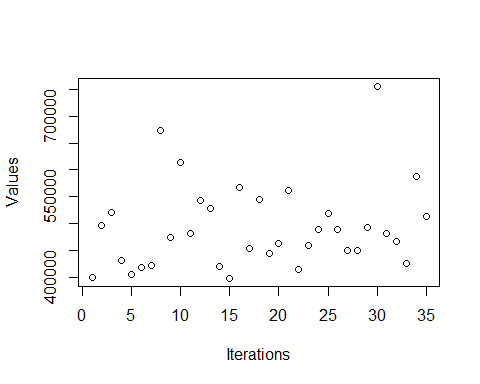
## Best measure value = 398753.81454

## Std. dev. of best measure = 30338.0068

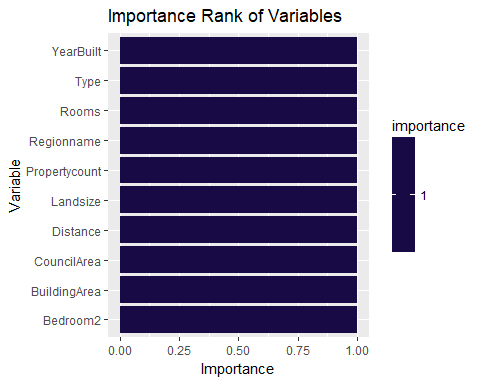
## Run time = 28.59 minutes.

## $target  
## [1] "Price"  
##   
## $importance  
## features importance  
## 1 Rooms 1  
## 2 Type 1  
## 3 Distance 1  
## 4 Bedroom2 1  
## 5 Landsize 1  
## 6 BuildingArea 1  
## 7 YearBuilt 1  
## 8 CouncilArea 1  
## 9 Regionname 1  
## 10 Propertycount 1  
##   
## $nfeatures  
## [1] 10  
##   
## $niters  
## [1] 35  
##   
## $name  
## [1] "K-Nearest-Neighbor regression"  
##   
## $best.iter  
## [1] 15  
##   
## $best.value  
## [1] 398753.8  
##   
## $best.std  
## [1] 30338.01  
##   
## attr(,"class")  
## [1] "summary.spFSR"

## [1] "Rooms" "Type" "Distance" "Bedroom2"   
## [5] "Landsize" "BuildingArea" "YearBuilt" "CouncilArea"   
## [9] "Regionname" "Propertycount"



## features importance  
## 1 Rooms 1  
## 2 Type 1  
## 3 Distance 1  
## 4 Bedroom2 1  
## 5 Landsize 1  
## 6 BuildingArea 1  
## 7 YearBuilt 1  
## 8 CouncilArea 1  
## 9 Regionname 1  
## 10 Propertycount 1

 The important features that were selected are type,rooms, property count, landsize, council area and building area. Here the variables can be seen to have equal importance and are coloured dark navy blue.

## 2.1 Feature filtering as per feature selection

Important features have been considered while building the models.

## 2.2 Splitting the data set

The data set has been split into 80% training set and 20% test set.

## 2.3 Task Preparartion for knn and Random Forest models

## 3 Hyper Parameter Tuning

## 3.1 Linear Regression

No parameter tuning has been done for linear regression learner.

## 3.2 K- Nearest Neighbour

By using the optimal kernel, grid search on k was done, where k = 2,3…20. The outcome was 9 with a rmse of 3.43e+05 .

### 3.3 Random Forest

We tune-fined the number of variables randomly sampled as candidates at each split (i.e. mtry). Therefore, we experimented mtry = 2, 3, and 4. We left other hyperparameters,such as the number of trees to grow at the default value. The result was 4 with a mean test error of 3.02e+05.

## [Tune] Started tuning learner regr.kknn for parameter set:

## Type len Def Constr Req Tunable  
## k discrete - - 2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,... - TRUE  
## Trafo  
## k -

## With control class: TuneControlGrid

## Imputation value: Inf

## [Tune-x] 1: k=2

## [Tune-y] 1: rmse.test.rmse=4.53e+05; time: 0.0 min

## [Tune-x] 2: k=3

## [Tune-y] 2: rmse.test.rmse=4.3e+05; time: 0.0 min

## [Tune-x] 3: k=4

## [Tune-y] 3: rmse.test.rmse=4.16e+05; time: 0.0 min

## [Tune-x] 4: k=5

## [Tune-y] 4: rmse.test.rmse=4.08e+05; time: 0.0 min

## [Tune-x] 5: k=6

## [Tune-y] 5: rmse.test.rmse=4.02e+05; time: 0.0 min

## [Tune-x] 6: k=7

## [Tune-y] 6: rmse.test.rmse=3.99e+05; time: 0.0 min

## [Tune-x] 7: k=8

## [Tune-y] 7: rmse.test.rmse=3.97e+05; time: 0.0 min

## [Tune-x] 8: k=9

## [Tune-y] 8: rmse.test.rmse=3.95e+05; time: 0.0 min

## [Tune-x] 9: k=10

## [Tune-y] 9: rmse.test.rmse=3.94e+05; time: 0.0 min

## [Tune-x] 10: k=11

## [Tune-y] 10: rmse.test.rmse=3.93e+05; time: 0.0 min

## [Tune-x] 11: k=12

## [Tune-y] 11: rmse.test.rmse=3.92e+05; time: 0.0 min

## [Tune-x] 12: k=13

## [Tune-y] 12: rmse.test.rmse=3.92e+05; time: 0.0 min

## [Tune-x] 13: k=14

## [Tune-y] 13: rmse.test.rmse=3.91e+05; time: 0.0 min

## [Tune-x] 14: k=15

## [Tune-y] 14: rmse.test.rmse=3.91e+05; time: 0.0 min

## [Tune-x] 15: k=16

## [Tune-y] 15: rmse.test.rmse=3.91e+05; time: 0.0 min

## [Tune-x] 16: k=17

## [Tune-y] 16: rmse.test.rmse=3.91e+05; time: 0.0 min

## [Tune-x] 17: k=18

## [Tune-y] 17: rmse.test.rmse=3.91e+05; time: 0.0 min

## [Tune-x] 18: k=19

## [Tune-y] 18: rmse.test.rmse=3.91e+05; time: 0.0 min

## [Tune-x] 19: k=20

## [Tune-y] 19: rmse.test.rmse=3.91e+05; time: 0.0 min

## [Tune] Result: k=18 : rmse.test.rmse=3.91e+05

## [Tune] Started tuning learner regr.randomForest for parameter set:

## Type len Def Constr Req Tunable Trafo  
## mtry discrete - - 2,3,4 - TRUE -

## With control class: TuneControlGrid

## Imputation value: Inf

## [Tune-x] 1: mtry=2

## [Tune-y] 1: rmse.test.rmse=2.82e+05; time: 2.0 min

## [Tune-x] 2: mtry=3

## [Tune-y] 2: rmse.test.rmse=2.77e+05; time: 2.9 min

## [Tune-x] 3: mtry=4

## [Tune-y] 3: rmse.test.rmse=2.76e+05; time: 3.6 min

## [Tune] Result: mtry=4 : rmse.test.rmse=2.76e+05

## 3.4 Creating learners as per tuned hyper parameters

Learners have been created for K-Nearest Neighbour and Random Forest.

## 3.5 Final modelling using tuned learners

## 4 Evaluation

Pred1 shows the prediction done by the linear regression model where rmse is 4.629529e+05. Pred 2 shows the prediction done by the knn model where rmse is 3.893729e+05. Pred 3 shows the prediction done by the random forest model where rmse is 3.581481e+05.

performance(pred1, measures = list(sse, mse, rmse, rsq, mae))

## sse mse rmse rsq mae   
## 4.355122e+14 2.449450e+11 4.949192e+05 4.819701e-01 3.156568e+05

performance(pred2, measures = list(sse, mse, rmse, rsq, mae))

## sse mse rmse rsq mae   
## 3.200186e+14 1.799880e+11 4.242499e+05 6.193465e-01 2.406641e+05

performance(pred3, measures = list(sse, mse, rmse, rsq, mae))

## sse mse rmse rsq mae   
## 1.957701e+14 1.101069e+11 3.318237e+05 7.671367e-01 1.698403e+05

Based on rmse value it can be concluded that the random forest model performs the best out here. (Note by- However the difference is large between the real and predicted values because most of the house prices are in millions and billions. Therefore a slight difference causes a huge variance.)

## 5 Discussion

The previous section showed that all regressors did not perform accurately in predicting the house prices despite the stratified sampling. This implies the imbalance class problem was prevalent. A betterapproach would be a cost-sensitive classification where we could have allocated more cost to true positive groups.Another alternative would be under- or oversampling toadjust the class balance, despite the risk of inducing biases. The linear regression model did not perform well in this scenario. This highlights the KNN regressor might not be appropriate given there were many categorical features in the data. The RF outperformed other models because it had the bagging mechanism to improve its accuracy. Having said this, it was “unfair” to the linear regression model and KNN regressor because the random forest was able to run multiple bagged models at each iteration during the resampling.

## 6 Conclusion

Among three regressors, the Random Forest produces the best performance in predicting house prices.Thedata was split into training and test sets. Via stratified samplingthe optimal value of the selected hyperparameter of each regressor was selected. Despite this,the imbalance class issue still persisted and therefore reduced the accuracy of the house prices.For future works, we proposed to consider cost-sensitive classification and under/over-sampling methods to mitigate the class imbalance.

## References