Class	5
Full Name	Venkat Subramanian
Matriculation Number	U1921075D

Declaration of Academic Integrity

By submitting this assignment for assessment, I declare that this submission is my own work, unless otherwise quoted, cited, referenced or credited. I have read and understood the Instructions to CBA.PDF provided and the Academic Integrity Policy.

I am aware that failure to act in accordance with the University's Academic Integrity Policy may lead to the imposition of penalties which may include the requirement to revise and resubmit an assignment, receiving a lower grade, or receiving an F grade for the assignment; suspension from the University or termination of my candidature.

I consent to the University copying and distributing any or all of my work in any form and using third parties to verify whether my work contains plagiarised material, and for quality assurance purposes.

[X] I have read and accept the above.

Table of Contents

Answer to Q1:

Question 1A)

The code is as follow:

data1 = fread("resale-flat-prices-201701-202103.csv",stringsAsFactors = T)

Result : We can see that the textual data such as flat_type and town is treated as categorical data instead of text data

```
> summary(data1)
     month                town                   flat_type         block
 2018-07: 2539   SENGKANG   : 7763   1 ROOM           :   43   2      :  325
 2021-01: 2498   JURONG WEST: 6758   2 ROOM           : 1432   8      :  307
 2020-12: 2491   WOODLANDS  : 6723   3 ROOM           :22458   1      :  277
 2020-09: 2482   PUNGGOL    : 6414   4 ROOM           :39085   101    :  264
 2020-07: 2456   YISHUN     : 6411   5 ROOM           :23722   109    :  257
 2020-06: 2438   TAMPINES   : 6282   EXECUTIVE        : 7587   114    :  257
 (Other):79469   (Other)    :54022   MULTI-GENERATION:   46   (Other):92686
            street_name      storey_range   floor_area_sqm              flat_model
 YISHUN RING RD    : 1397   04 TO 06:21919   Min.   : 31.00   Model A          :30864
 BEDOK RESERVOIR RD: 1072   07 TO 09:19842   1st Qu.: 82.00   Improved         :23635
 PUNGGOL FIELD     : 1068   10 TO 12:17655   Median : 95.00   New Generation   :12684
 PUNGGOL DR        : 1039   01 TO 03:16950   Mean   : 97.76   Premium Apartment:10394
 ANG MO KIO AVE 10 :  978   13 TO 15: 8869   3rd Qu.:113.00   Apartment        : 3839
 FERNVALE RD       :  934   16 TO 18: 4087   Max.   :249.00   Simplified       : 3818
 (Other)           :87885   (Other) : 5051                    (Other)          : 9139
 lease_commence_date          remaining_lease   resale_price     remaining_lease_yrs
 Min.   :1966        94 years 09 months:  740   Min.   : 140000   Min.   :45.00
 1st Qu.:1984        94 years 11 months:  711   1st Qu.: 335000   1st Qu.:65.00
 Median :1995        94 years 10 months:  710   Median : 415000   Median :75.00
 Mean   :1995        94 years 08 months:  692   Mean   : 445997   Mean   :74.79
 3rd Qu.:2004        94 years 07 months:  636   3rd Qu.: 522888   3rd Qu.:84.00
 Max.   :2019        94 years 06 months:  602   Max.   :1258000   Max.   :98.00
                     (Other)           :90282
```

Figure 1Summary of dataset

Question 1B)

The code is as follows :

data1$remaining_lease_yrs = as.numeric(substring(data1$remaining_lease_yrs,0,2))

Result:

From the result we can see that the derived colum has numeric class type

```
> class(data1$remaining_lease_yrs)
[1] "numeric"
```

Figure 2:Data type of derived column

Question 1C)

Code:

data1$remaining_lease = NULL

data1$lease_commence_data = NULL

Result: We can see that the columns remaining_lease and lease_commence_data are removed from the dataset

```
> colnames(data1)
 [1] "month"              "town"               "flat_type"          "block"
 [5] "street_name"        "storey_range"       "floor_area_sqm"     "flat_model"
 [9] "resale_price"       "remaining_lease_yrs"
```

Figure 3: Column names of the dataset after 1c

Question 1D)

Code:

data1$block_street = paste(data1$block,data1$street_name)

data1$block_street = factor(data1$block_street)

data1$block = NULL

data1$street_name = NULL

Result:

From the summary of block_street it is observable that the block string has been concatenated with street.

```
> summary(data1$block_street)
 139A LOR 1A TOA PAYOH   139B LOR 1A TOA PAYOH   138B LOR 1A TOA PAYOH       308A PUNGGOL WALK
                   85                      79                      64                      64
 138C LOR 1A TOA PAYOH          8 BOON KENG RD         211 BOON LAY PL           636B SENJA RD
                   62                      59                      56                      55
    91 TANGLIN HALT RD   588C ANG MO KIO ST 52 80A TELOK BLANGAH ST 31      12A MARSILING LANE
                   55                      53                      53                      52
```

Figure 4 Summary of Block Street

By using colnames of the dataset, we can check that the 2 columns block, and street names are indeed removed from the dataframe

```
> colnames(data1)
[1] "month"              "town"               "flat_type"          "storey_range"
[5] "floor_area_sqm"     "flat_model"         "resale_price"       "remaining_lease_yrs"
[9] "block_street"
```

Figure 5 Column names after 1d

Answer to Q2:

Question 2A)

i) The month and year with the lowest amount of transaction is 05,2020 with 369 transactions

ii) The month and year with the highest amount of transaction is 07 2018 with 2539 transactions

Question 2B)

i) The town with the lowest transaction volume is Bukit Timah with a volume of 264

ii) The town with the highest transaction volume is Sengkang with a volume of 7763

Question 2C)

Rows 1 to 5 Represent the top 5 resale prices while rows 6-10 represent bottom 5 resale prices. It is ordered in descending order

Top 5

Bottom 5

	flat_type	block_street	town	floor_area_sqm	storey_range	resale_price
1	5 ROOM	1B CANTONMENT RD	CENTRAL AREA	107	43 TO 45	1258000
2	5 ROOM	1A CANTONMENT RD	CENTRAL AREA	105	49 TO 51	1248000
3	5 ROOM	1B CANTONMENT RD	CENTRAL AREA	107	40 TO 42	1232000
4	5 ROOM	273B BISHAN ST 24	BISHAN	120	28 TO 30	1220000
5	5 ROOM	273B BISHAN ST 24	BISHAN	120	25 TO 27	1218888
6	2 ROOM	72 CIRCUIT RD	GEYLANG	42	10 TO 12	160000
7	1 ROOM	7 TELOK BLANGAH CRES	BUKIT MERAH	31	10 TO 12	157000
8	2 ROOM	68 CIRCUIT RD	GEYLANG	45	04 TO 06	150000
9	2 ROOM	52 LOR 6 TOA PAYOH	TOA PAYOH	43	01 TO 03	150000
10	3 ROOM	26 TOA PAYOH EAST	TOA PAYOH	67	10 TO 12	140000

Figure 6 Top and Bottom resale pricing in decreasing order

Question 2D)

By plotting the resale price across different flat types, we are able to see an increasing trend whereby if the number of rooms increase the resale price increases as well

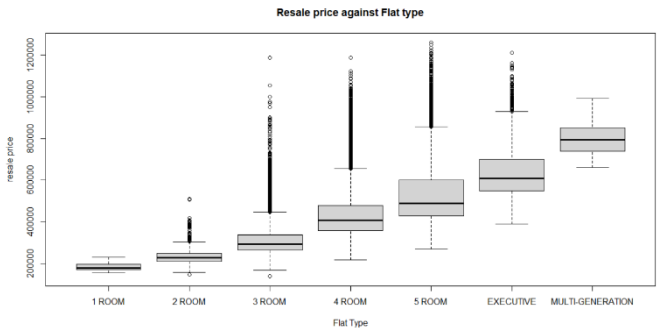

Figure 7: Resale Price against Flat Type

An increase in the number of rooms would mean that the total floor area of the flat would be higher. When floor area is plotted against resale price, the similar positive trend is observed. And the assumption that the flat types with higher number of rooms have higher floor area holds true. This shows that maybe floor area is the underlying factor to the resale price rather than the flat type

Figure 8: Resale Price against Floor area with Flat types

Figure 9 Shows the resale price of different flats across the different towns. It seems that Bukit timah tend to have more expensive flats (represented by the highest median). This high price could be a deterrent to buyers which would explain why bukit timah had the lowest number of transactions

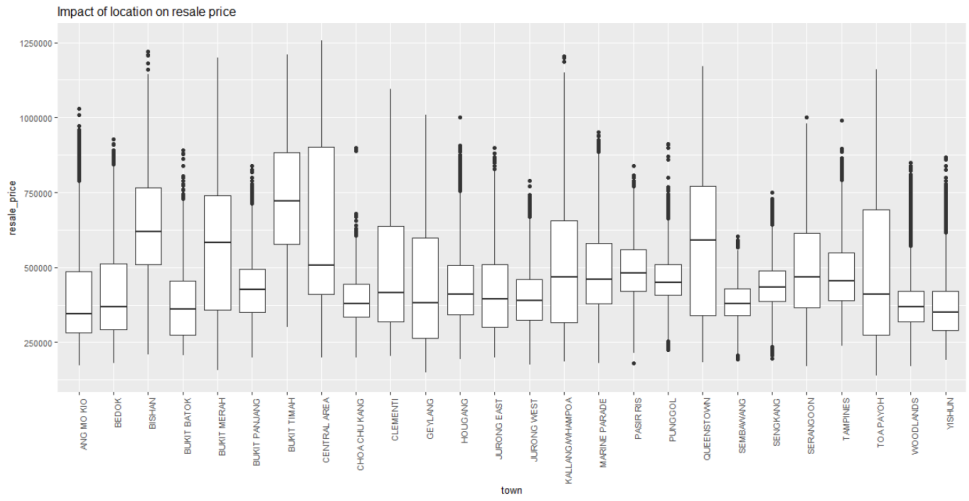

Figure 9: Resale Prices across different Towns

However, it should be noted that Sengkang had the highest number of transactions despite not having the lowest resale prices. Maybe the resale prices across different towns are affected by other variables.

Another possible reasons why certain flats have higher price than the others might be due to its remaining lease. Figure 10 shows the relationship between remaining lease and resale price. Both attributes do have a positive correlation, but the correlation is not as strong as floor area (shown in Fig 8)

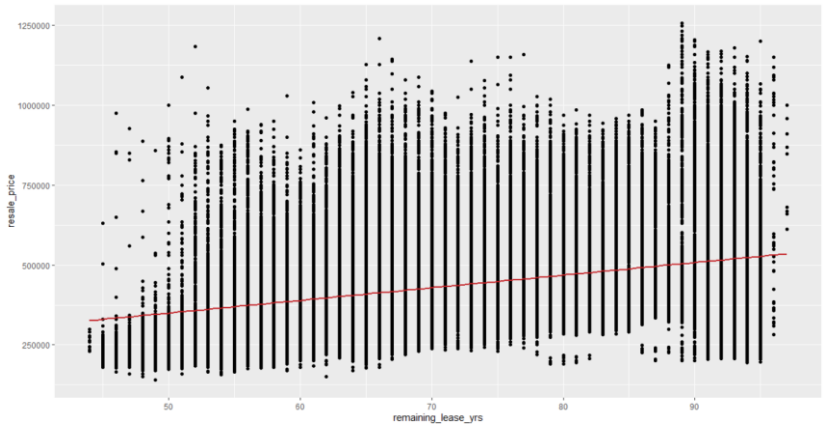

Figure 10:Reslae Price against Remaining Lease

Could resale price and Floor area affect the prices across the different towns? Fig 11 shows the trend of floor area across different towns. The plot shows that pasir ris had the highest floor area and by the moderate positive correlation (around 0.6) between floor area and resale price, Pasir ris should have the highest resale price across the different areas. Could this be due to remaining lease instead?

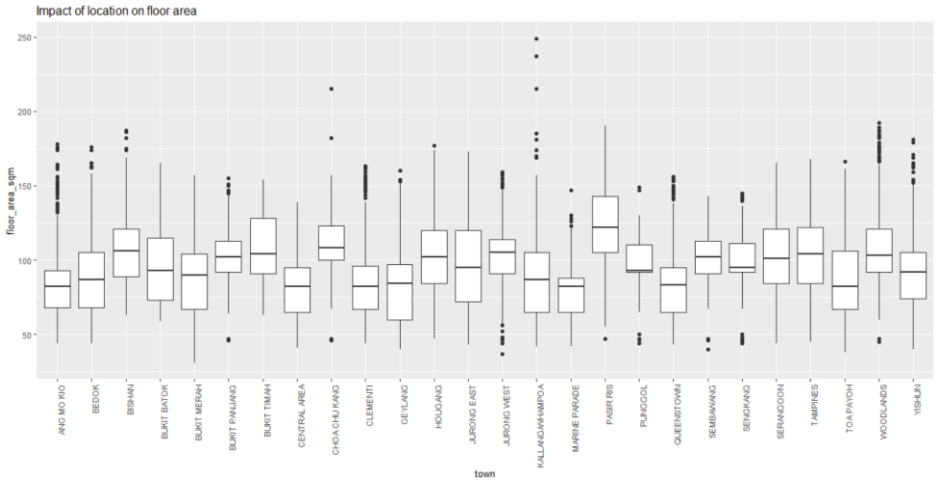

Figure 11: Floor area across different towns

Figure 12 shows the trend of remaining lease across different towns. Punggol had the highest remaining lease years followed by Sengkang. However, the remaining lease for Punggol and Sengkang flats are slightly similar and the floor area is the same is as well. Therefore, the features in the town such as Malls, Train station and School will affect the resale price of the flat located in a certain town rather than the lease and floor area. As a result, town could play an important role in predicting resale price

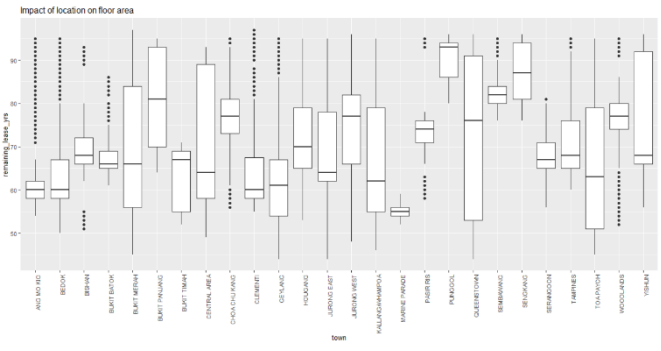

Figure 12: Remaining Lease Years across different Towns

By plotting the resale price across different flat model, it is observable that the type 1 and type 2 buildings had the top resale prices and 2-room had the lowest resale price

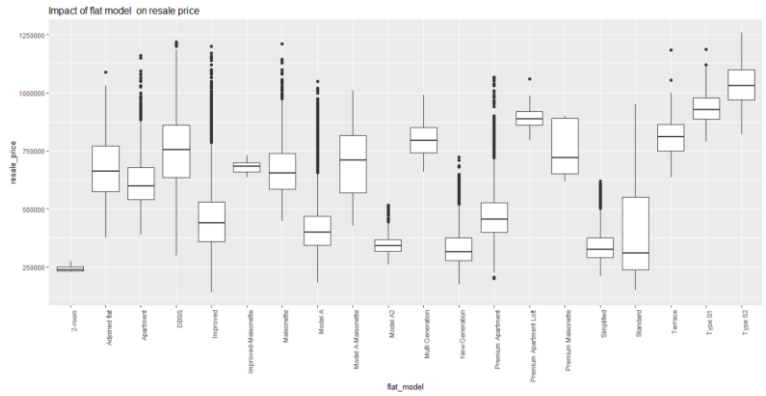

Figure 13:Resale Price across different flat model

In Figure 14 we can see that the premium apartment loft and type 1, 2 flats had the highest remaining lease. But if we look at figure 15, the floor area of these type of flats is smaller than the others. The same observation is seen for 2-room type. Therefore, for certain flat models' different aspect of the flat might play an important in determining the resale price of the flat.

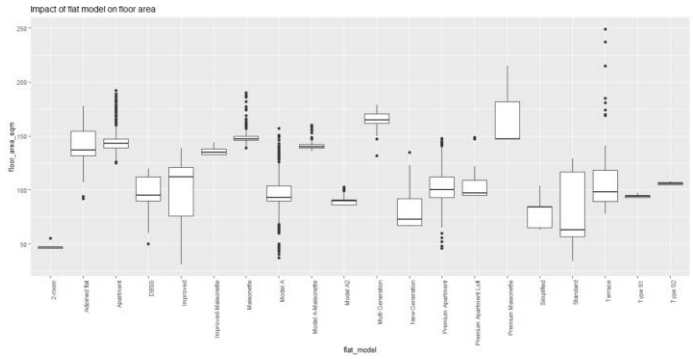

Figure 15:Floor Area across different Flat models

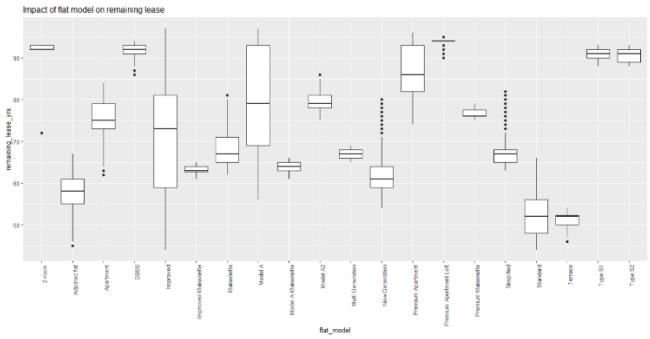

Figure 14:Resale Price across different flat models

Figure 16 shows the relationship between storey range and the resale price. With flats that have a higher level tend to have a higher resale value. This could possibly be due to the potential view that the buyers could experience. This intangible asset could be a reason to add value to the resale value

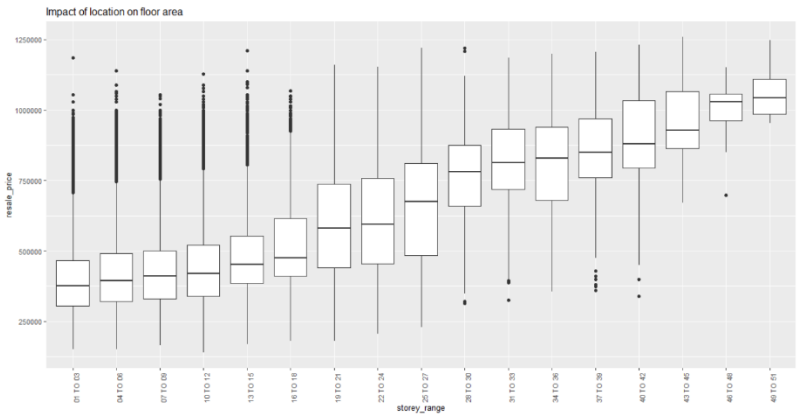

Figure 16:Resale Price across different Storey range

When the storey range was further explored it seem that it is affected by the remaining lease attribute. From Figure 17 we are able to see that the range of the remaining lease decreases across the different storey range, implying that the newer building tends to have higher storey range. Therefore, these flats that had a higher storey range could also have a more years on their lease contributing to the increase in resale price

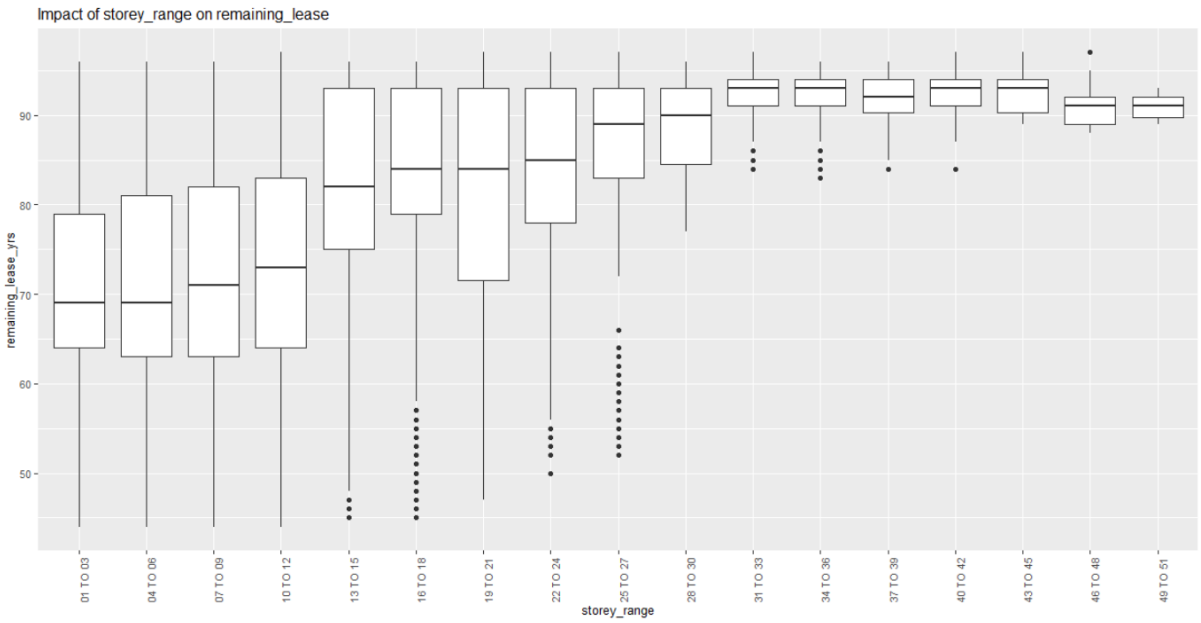

Figure 17: Remaining lease across different storey range

Therefore, from the exploratory analysis, it was concluded that the resale price might be affected by the remaining lease of the flat, the floor area as well as the town located.

Answer to Q3:

Question3A)

Code:

data2 = data.frame(data1)

Result:

Using the function data.frame we are able to copy data1 into data2. And we can check also check that data2 is not a pointer pointing to data1 by using the tracemem function which returns false

```
> tracemem(data1)==tracemem(data2)
[1] FALSE
```

Figure 18: Code output to show a copy has made

Question 3B)

Code:
data2 = data2[data2$flat_type!="1 ROOM" & data2$flat_type!="MULTI-GENERATION",]

data2$flat_type = droplevels.factor(data2$flat_type)

Result:

From the levels we can see that the 2 levels are indeed removed from the dataframe

```
> levels(data2$flat_type)
[1] "2 ROOM"    "3 ROOM"    "4 ROOM"    "5 ROOM"    "EXECUTIVE"
> summary(data2$flat_type)
   2 ROOM    3 ROOM    4 ROOM    5 ROOM EXECUTIVE
     1432     22458     39085     23722      7587
```

Figure 19: Code output that shows the new levels as well as the number of entries per flat model

Question 3C)

Code:

data2$block_street = NULL

Result:

From the colnames we can see it was removed successfully

```
> colnames(data2)
[1] "month"              "town"               "flat_type"          "storey_range"
[5] "floor_area_sqm"     "flat_model"         "resale_price"       "remaining_lease_yrs"
```

Figure 20:Code output after dropping the columns

Question 3D & 3E)

Before combining the levels

```
> summary(data2$storey)
01 TO 03 04 TO 06 07 TO 09 10 TO 12 13 TO 15 16 TO 18 19 TO 21 22 TO 24 25 TO 27 28 TO 30
   16936    21894    19822    17625     8869     4087     1779     1326      729      459
31 TO 33 34 TO 36 37 TO 39 40 TO 42 43 TO 45 46 TO 48 49 TO 51
     212      202      198       99       18       21        8
```

Figure 21: Code output before combining the levels of storey range

After combining the levels

```
> summary(data2$storey)
01 TO 03 04 TO 06 07 TO 09 10 TO 12 13 TO 15 16 TO 18 19 TO 21 22 TO 24 25 TO 27 28 TO 30
   16936    21894    19822    17625     8869     4087     1779     1326      729      459
31 TO 33 34 TO 36 37 TO 39 40 to 51
     212      202      198      146
```

Figure 22: Code output after combining the levels in Storey_range together

By comparing the number of transactions in each of the different we can be assured that the rest of the levels were not modified and the transaction for storey range 40 to 51 is 146 which is equal to 99+18+21+8. Therefore, the levels in storey has been correctly merged

Question 3F)

Code:

data2$storey_range = NULL

Result:

Again, by checking the colnames of data2, we can be assured that the column storey_range has been removed

```
> colnames(data2)
[1] "month"              "town"               "flat_type"          "floor_area_sqm"
[5] "flat_model"         "resale_price"       "remaining_lease_yrs" "storey"
```

Figure 23: Code output to show the remaining column names after dropping

Question 3G)

```
> summary(data2$flat_model)
        Adjoined flat            Apartment                 DBSS             Improved
                  178                 3839                 1673                23592
           Maisonette              Model A  Model A-Maisonette             Model A2
                 2847                30864                  168                 1164
       New Generation Premium Apartment Premium Apartment Loft           Simplified
                12684                10394                   52                 3818
             Standard              Terrace              Type S1             Type S2
                 2677                   56                  155                   91
```

Figure 24: Number of transactions per flat model after dropping the levels

Question 3H)

Data2 has 94252 rows and 8 columns. The number of rows shrunk by 0.11 percent due to data cleaning and preparation

```
> dim(data2)
[1] 94252     8
```

Figure 25: The new dimension of the data2 dataset

Question 3I)

Most of the steps carried out in this section was to reduce the number of levels in each of the categorical data which reduces bias and computational time as well.

In part c we removed the column block_street which had 9008 levels. One of the reasons for this is that the additional column does not provide any additional information. The information provided by block_street is already provided by the attributed town.

In part b, we removed 2 levels in flat_type. 1-Room only had 43 cases and Multi-Generation only had 46 cases. If we combine these 2 total cases it is only 0.094% of the dataset. These levels do not have sufficient data to be considered significant. This same concept is applied to part g where we removed insignificant levels from the data.

In part d however we did not merely remove the levels with insignificant data. From question 2d we saw that as the storey_range increases from one level to another, the resale price increases. By removing these levels, the relationship between the different levels will not be as stark as before. Making it better to combine the insignificant levels to one rather than to remove.

By reducing the number of levels in each of the categorical variable we can reduce the computational time required to train the model. For example, in random forest the time complexity for k predictor categories is 2^{k-1} due to the dummy variables(Wright & König, 2019). Therefore, if the number of levels in the categorical variable increases the computational time required increases exponentially.

Furthermore, for models such as Random Forest, the variable selection will tend to be bias if we have more levels in the categorical data (Strobl, Boulesteix, Zeileis, & Hothorn, 2007). Therefore, suboptimal predictor variables that do not improve the accuracy of the model is selected merely because the predictor has more categories.

Thus, for the afore mentioned reason we had to reduce the number of levels to reduce the time complexity and the biasness in variable selection

Answer to Q4:

Model Type	Train RMSE	Test RMSE
Linear Regression	56620	55725
MARS with Degree 2	54092	53806
Random Forest	24759	39307

* Before running the random forest seed 2021 was set again even after the train test split to reset the numbers generated to create random bootstrap samples

The Root-Mean-Square Error of the 3 different models is as shown. Random forest had the lowest RMSE for both train and test sets. Due to its low RMSE, Random forest is able to predict the resale value of the property much closer to the actual price. Another point to take note is that the trainset error of both linear regression and MARS have lower RMSE than the test set. This could imply that these models have not trained fully on the dataset (under trained) Therefore, Random Forest performed the best.

Answer to Q5:

Since the trees in Random Forest is only trained on $\frac{2}{3}$ of the bootstrap sample and validated on the remaining one-third of the sample. The Mean-Square-error (MSE) on the last tree represent the accumulated Out-Of-Bag (OOB) MSE. Therefore, we are able to obtain the RMSE of the OOB by simply square rooting the last MSE of the random Forest object. This gives a RMSE of $ 39622 (rounded off to the nearest dollar)

OOB RMSE vs Test Set RMSE

The OOB samples are similar to the test set in a way that they were not seen by the model during the training process. OOB has the advantage of delivering internal accuracy without the need to separate the dataset into train and test set (Ramosaj & Pauly, 2019). Furthermore it is also shown that OOB errors are as accurate as using a test set the same size as the training set (Breiman, 2001).

Some might argue that the OOB error is not representative of the model since the OOB samples are not tested on all the trees unlike the test set samples. We know that each tree in Random Forest is trained on the bootstrap sample and one-third of this sample is not used to grow the tree (Archer & Kimes, 2008).And these samples that is not used to grow the tree are called OOB. If we consider the total number of samples that have been tested by all the trees it would give us

$Total\ nuber\ of\ samples\ tested = \frac{1}{3}(Size\ of\ trainset) * n$, where n represent the number of trees

And if we rearrange the equation, we will get

$$Total\ nuber\ of\ samples\ tested = \frac{1}{3}(n) * Size\ of\ trainset$$

This would mean that the OOB samples are only tested on one-third of the total trees in the model (Soifua, 2018). These people would argue that since the test set samples are tested on all the trees the OOB error would rather overestimate the performance of the model when compared to test set error.

However, the number of trees does not seem to play a significant role on the accuracy of the model as a mere 50 can be used for classification and 25 used for regression (Breiman, 1996). In fact, since OOB samples are only tested on one-third of the trees it would require lesser computation power to test the samples when compared to the test set

But across different studies and papers, test set error is commonly used rather than OOB error. A possible reason for this phenomenon is that when working with models usually multiple models are generated, and its effectiveness is tested on the same test set to make evaluation of these models fair. As a result OOB is commonly used to calibrate certain parameters of the model such as the number of trees (shown in figure 26) and the variable importance(Calhoun et al., 2020)

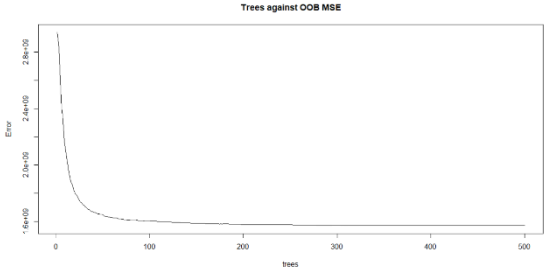

Figure 26 : OOB MSE across different tree in Random Forest

In conclusion if we are evaluating just different random forest, OOB RMSE can be used to replace test set RMSE. But in cases where multiple models are being evaluated, the test set RMSE would be a better representation of the model's effectiveness.

Answer to Q6:

Pros of Using Gini Based Variable Importance

By using Gini Importance, we are able to explicitly remove irrelevant features that are present on a spectral data (Menze et al., 2009). Also Gini Calculations are already performed during training phase of the trees, Gini importance is simple and fast to compute (Nembrini, König, & Wright, 2018)

Cons of Using Gini Based Variable Importance

Although it was previously mentioned that Gini Importance is able to explicitly remove irrelevant features present in spectral data, it is only applicable to an optimal subset of features (Menze et al., 2009).Furthermore it was mentioned in Question 3d that the Gini Importance tend to be biased towards variable with a lot of categories and categories that have a high frequency (Boulesteix, Bender, Lorenzo Bermejo, & Strobl, 2012).

Gini Importance VS Permutation importance

Earlier it was mentioned that the Gini importance tend to give bias result to variables that have more frequent categories or simply more levels in them. This not a issue in permutation importance as the variable importance is based on the decrease in accuracy resulting from the permutation of OOB observation (Boulesteix et al., 2012).

The effect of this biasness can be clearly seen in the random forest generated in Question 4

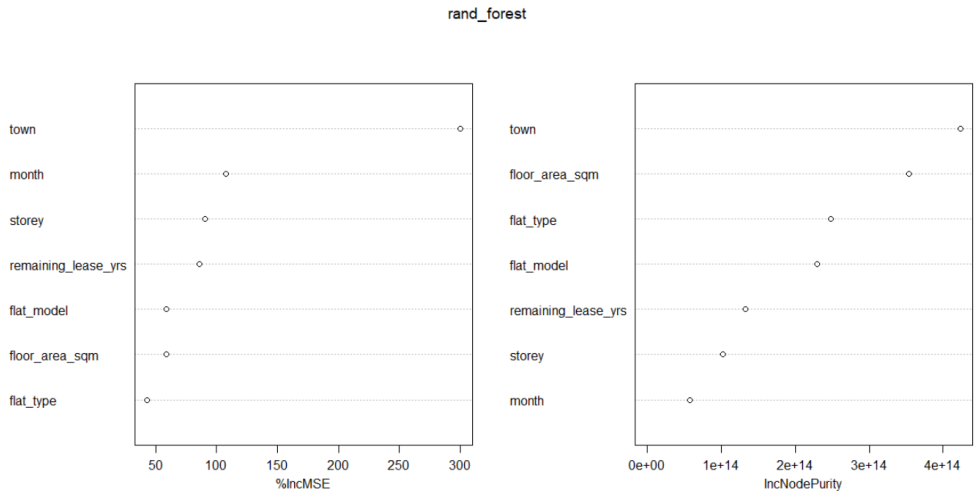

Figure 27: Variable Importance by Gini and Permutation

If we look at the Variable importance derived using Gini Importance (the graph on the right) we can see that variable such as Month and Storey are selected as the important features in the model even though they are correlated with remaining_lease_yrs . And if we look at the Variable importance derived using Permutation Importance the important factors that we discussed in question 2 do have a higher importance associated with it when compared to those in Gini importance. This reinforces the point that Gini Importance tend to favour variable with more levels (month had 51 levels). Therefore, just by comparing Permutation importance and Gini Importance permutation importance is a better measure to evaluate variable importance.

But permutation importance is not necessarily the best , since it is based on the permutation of OOB observation it is computationally intensive on high dimensional data (Nembrini et al., 2018). Also the OOB error might overestimate the true prediction error of the model (Janitza & Hornung, 2018) and since permutation importance is based on the decrease in accuracy, the overestimation of the OOB error might give false results on the variable importance

Furthermore both Permutation Importance and Gini Importance, may overstate the importance of correlated predictors (Hooker & Mentch, 2019). Instead of using these 2 methods for future purposes another method that has some promise is the Actual Impurity Reduction Variable Importance Measure (VIM). This method of VIM has shown that it requires lesser computational power than permutation importance while giving unbiased results (Nembrini et al., 2018)

Answer to Q7:

As discussed earlier random forest creates n samples of size n. And approximately one-third of this data in the bootstrapped sample is not used to test the model while the remaining two-third are used to train the model. However, this would make the OOB sample different for different tree. And some trees in the model would be able to have a better performance since the OOB sample could be similar to the in bag data (Gajowniczek, Grzegorczyk, Ząbkowski, & Bajaj, 2020). Due the similarity of the OOB sample to its in-bag sample the tree could have a lower OOB error thereby having a higher weight assigned to it. And as mentioned by Soifua, some of the OOB data of the tree could be harder to predict. Due to the ranking system, the trees that are allocated with similar data are 'praised' and the trees with data that is different from what is trained on is 'penalised'. Therefore, evaluating on a test set that is randomly generated for every tree does not ensure fairness and it might be biased to certain trees resulting in only a moderate improvement in the accuracy of the weighted random forest compared to random forest.

Previously in question 5 it was discussed on how researchers tend to use test set error to evaluate multiple models at once. The trees in random forest could be treated as a sperate model therefore, to assign appropriate weights each tree needs to be tested on the same test set rather than the OOB samples which is randomly generated. This is similar to the algorithm behind the Fairly weighted Random Forest. Since the weight is allocated based on the tree's performance on a fixed test set (the k fold that is not used to train) , these weights would be suitable and appropriate since all the trees are evaluated fairly

References:

Archer, K. J., & Kimes, R. V. (2008). Empirical characterization of random forest variable importance measures. *Computational statistics & data analysis, 52*(4), 2249-2260. doi:10.1016/j.csda.2007.08.015

Boulesteix, A.-L., Bender, A., Lorenzo Bermejo, J., & Strobl, C. (2012). Random forest Gini importance favours SNPs with large minor allele frequency: impact, sources and recommendations. *Briefings in Bioinformatics, 13*(3), 292-304. doi:10.1093/bib/bbr053

Breiman, L. (1996). Bagging predictors. *Machine learning, 24*(2), 123-140. doi:10.1007/BF00058655

Breiman, L. (2001). Random Forests. *Machine learning, 45*(1), 5-32. doi:10.1023/A:1010933404324

Calhoun, P., Hallett, M. J., Su, X., Cafri, G., Levine, R. A., & Fan, J. (2020). Random forest with acceptance–rejection trees. *Computational statistics, 35*(3), 983-999. doi:10.1007/s00180-019-00929-4

Gajowniczek, K., Grzegorczyk, I., Ząbkowski, T., & Bajaj, C. (2020). Weighted Random Forests to Improve Arrhythmia Classification. *Electronics, 9*(1), 10.3390/electronics9010099. doi:10.3390/electronics9010099

Hooker, G., & Mentch, L. (2019). *Please Stop Permuting Features: An Explanation and Alternatives*.

Janitza, S., & Hornung, R. (2018). On the overestimation of random forest's out-of-bag error. *PloS one, 13*(8), e0201904-e0201904. doi:10.1371/journal.pone.0201904

Menze, B. H., Kelm, B. M., Masuch, R., Himmelreich, U., Bachert, P., Petrich, W., & Hamprecht, F. A. (2009). A comparison of random forest and its Gini importance with standard chemometric methods for the feature selection and classification of spectral data. *BMC bioinformatics, 10*(1), 213. doi:10.1186/1471-2105-10-213

Nembrini, S., König, I. R., & Wright, M. N. (2018). The revival of the Gini importance? *Bioinformatics, 34*(21), 3711-3718. doi:10.1093/bioinformatics/bty373

Ramosaj, B., & Pauly, M. (2019). Consistent estimation of residual variance with random forest Out-Of-Bag errors. *Statistics & Probability Letters, 151*, 49-57. doi:https://doi.org/10.1016/j.spl.2019.03.017

Soifua, B. (2018). *A Comparison of R, SAS, and Python Implementations of Random Forests*.

Strobl, C., Boulesteix, A.-L., Zeileis, A., & Hothorn, T. (2007). Bias in Random Forest Variable Importance Measures: Illustrations, Sources and a Solution." BMC Bioinformatics, 8(1), 25. *BMC bioinformatics, 8*, 25. doi:10.1186/1471-2105-8-25

Wright, M. N., & König, I. R. (2019). Splitting on categorical predictors in random forests. *PeerJ, 7*, e6339-e6339. doi:10.7717/peerj.6339