

**ANL252**

**Python For Data Analytics**

**Group-based Assignment**

**July 2023 Presentation**

**Submitted by:**

|  |  |
| --- | --- |
| **Name** | **PI No.** |
| Goh Jun Jie | K2210106 |
| Lim Soon Meng | Y2310177 |
| Angela Liu Qiyu | J2181353 |
| Nur Fazillah Binte Abdul Rahman | K1870536 |

**Tutorial Group:** T03\_GBA 4

**Submission Date:** 13 OCTOBER 2023

We, members of GBA group 4, do hereby declare that we each contributed to this assignment and that we collectively agree to a shared grade.

|  |  |  |
| --- | --- | --- |
| **Name** | **Contribution** | **Signature** |
| Angela Liu Qiyu | I did part a and b. | Angela |
| Goh Jun Jie | I did part d. | Jun Jie |
| Lim Soon Meng | I did part a, c and d. | Soon Meng |
| Nur Fazillah Binte Abdul Rahman | I did part d. | Fazillah |

**Question 1(a)**

#import pandas and numpy with alias pd and np

import pandas as pd

import numpy as np

#read\_csv() method under pandas to load data file from HDB

df\_hdb = pd.read\_csv("GBA\_HDB.csv")

#We can use .info() method to show number of entries and columns.

#Name of Columns and number of Null entries are also shown to us.

df\_hdb.info()

#we can use the shape parameter to show the number of rows and columns.

df\_hdb.shape

In order to read a dataset and identify its dimensions using Python, we can use libraries like pandas. After importing libraries, we use .read\_csv() to load the data from the CSV and stores it in a pandas DataFrame object called df. We can use the .shape parameter to quickly return us the rows and columns of the data. We can also use the .info() method to show the same as well as the column names and null entries.

(149 words)

**Question 1(b)**

#the read\_csv() method used in part A allows us to specify na\_filter and na\_values argument.

#na\_filter is True by default which converts all empty entries in the data to NaN.

#na\_values allows us to specify other entries to consider as NaN. This is not required for the HDB dataset.

#we chain the isnull() method and sum() method to return us the count of missing values in each column.

df\_hdb.isnull().sum(axis=0)

#we can chain isnull() and any() method along axis 1 to find all entries with missing values.

df\_hdb.isnull().any(axis=1)

#we then subset the result from the main dataframe to reflect the full entries.

df\_hdb[df\_hdb.isnull().any(axis=1)]

#we can also add the .index and .values parameters to return just the index number.

df\_hdb[df\_hdb.isnull().any(axis=1)].index.values

It is necessary to handle missing values because they can cause errors and lead to inaccurate analysis, injecting biases into both statistical analysis and machine learning models. This is due to the fact that the absence of data is often not randomly distributed but tends to cluster within specific subsets of data points. (Tamboli, 2023) This in turn affects real-world decision making.

(200 words)

**Question 1(c)**

#we can use .dropna() method to simply drop entries with missing values and only consider complete entries.

df\_dropmissing = df\_hdb.dropna(axis=0, how='any')

#we can also fill values into the missing values by deriving meaningful values through the rest of the data.

#From part B we find that missing values exist for 'flat\_type', 'street\_name' and 'resale\_values' column.

#To fill values for 'flat type' and ‘street\_name’, we can use .fillna() with the ffill method.

df\_fillmissing = df\_hdb.copy()

df\_fillmissing['flat\_type'].fillna(method='ffill', inplace=True)

df\_fillmissing['street\_name'].fillna(method='ffill', inplace=True)

df\_fillmissing['resale\_price'].interpolate(method='linear', inplace=True)

One way to treat missing values is to drop all entries with missing values with the .dropna() method. However, this is not practical in all situations as the missing data may be critical and there may be a common cause for the missing data. Dropping these entries may skew our results.

For the missing entries in ‘flat\_type’ and ‘street\_name’ we can use .fillna() with ‘ffill’ method as data is categorical and are arranged in order. For ‘resale\_price’, it makes more sense to use .interpolate() with ‘linear’ method as data is numerical that varies over a big range and is arranged in order.

(200 words)

**Question 1(d)**

#Import necessary libraries and modules

import matplotlib.pyplot as plt

import seaborn as sns

df\_pricevsarea = df\_dropmissing[["floor\_area\_sqm", "resale\_price"]]

df\_pricevsarea

|  |  |  |
| --- | --- | --- |
|  | **floor\_area\_sqm** | **resale\_price** |
| **0** | 60.0 | 255000.0 |
| **1** | 68.0 | 275000.0 |
| **2** | 69.0 | 285000.0 |
| **3** | 68.0 | 290000.0 |
| **4** | 68.0 | 290000.0 |
| **...** | ... | ... |
| **1245** | 134.0 | 460000.0 |
| **1246** | 133.0 | 500000.0 |
| **1247** | 146.0 | 525888.0 |
| **1248** | 142.0 | 538000.0 |
| **1249** | 146.0 | 550000.0 |

1076 rows × 2 columns

FLOOR\_AREA\_SQM = df\_pricevsarea["floor\_area\_sqm"]

RESALE\_PRICE = df\_pricevsarea["resale\_price"]

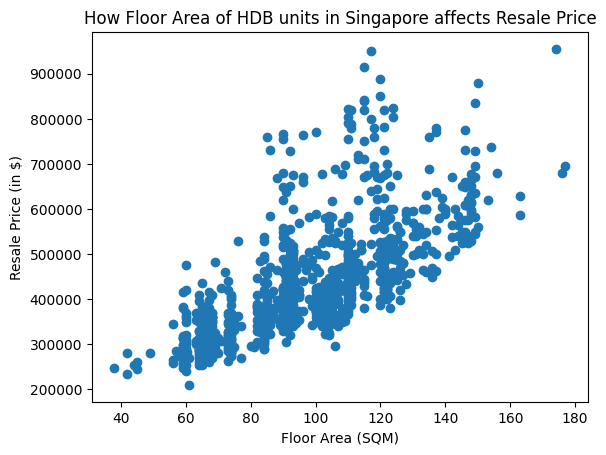
plt.scatter(FLOOR\_AREA\_SQM, RESALE\_PRICE, color = None, marker = None,

linewidths = None, edgecolors = None)

plt.xlabel("Floor Area (SQM)")

plt.ylabel("Resale Price (in $)")

plt.title("How Floor Area of HDB units in Singapore affects Resale Price")



This chart is a scatter plot because we are trying to study the correlation between two variables and demonstrate how the floor area of a HDB unit in Singapore affects the resale price. By plotting the scatter plot of resale price (in $) against floor area (in sqm), we can observe an overall positive correlation between floor area and resale price, where an increase in the floor area of a HDB unit would also increase its resale price value. However, from the scatter plot we can also observe that despite the overall positive correlation, there are still some HDB units with higher resale price value than other HDB units of a larger floor area. Therefore, it can be inferred that there are also many other factors at play influencing the resale price value, for example the location and condition of the HDB unit, and the remaining lease duration of the HDB unit, which we will analyze in our second chart.

sns.lmplot(data=df\_dropmissing, x='remaining\_lease', y='resale\_price', scatter=False, lowess=True)

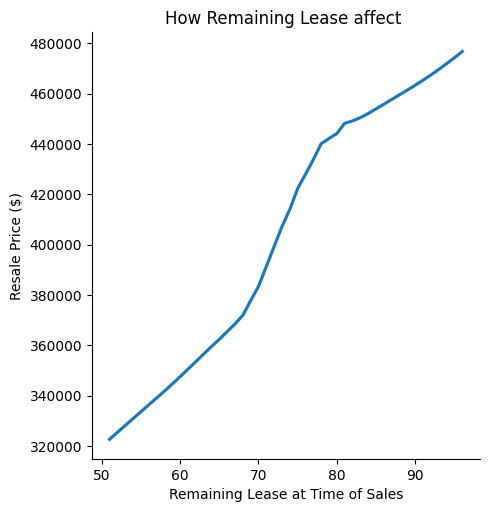
plt.xlabel('Remaining Lease at Time of Sales')

plt.ylabel('Resale Price ($)')

plt.title('How Remaining Lease affect ')

df\_pricelease = df\_dropmissing[['resale\_price','remaining\_lease']]

df\_pricelease



|  |  |  |
| --- | --- | --- |
|  | **resale\_price** | **remaining\_lease** |
| **0** | 255000.0 | 70 |
| **1** | 275000.0 | 65 |
| **2** | 285000.0 | 64 |
| **3** | 290000.0 | 63 |
| **4** | 290000.0 | 64 |
| **...** | ... | ... |
| **1245** | 460000.0 | 69 |
| **1246** | 500000.0 | 77 |
| **1247** | 525888.0 | 72 |
| **1248** | 538000.0 | 72 |
| **1249** | 550000.0 | 72 |

1076 rows × 2 columns

This is a regression plot of resale prices against the remaining lease with lowest argument set as ‘True’ to give weight to changes at the local level. The scatter plot is also hidden to present a more readable graph. This allows us to clearly see how remaining lease left in the flat affects the resale prices. The graph shows us that there is a sharp fall from 80 years of remaining lease to 70 years. This suggests that a flat with a longer remaining lease has a higher resale value hence, seller can consider selling off the flat while the remaining lease has 80 or more years.

sns.boxplot(data=df\_dropmissing, x='resale\_price', y='town')

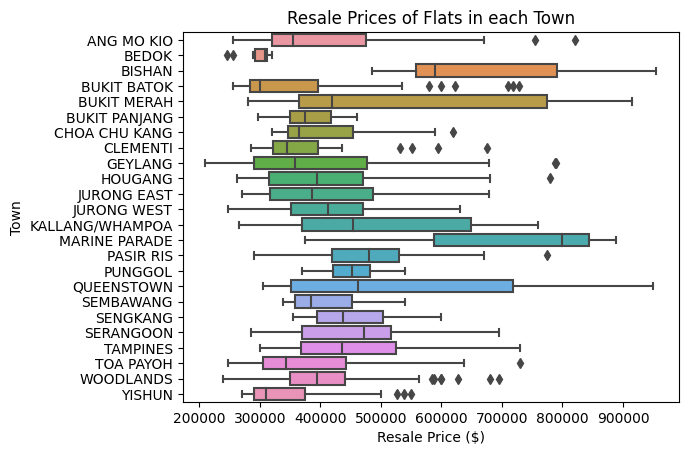
plt.xlabel('Resale Price ($)')

plt.ylabel('Town')

plt.title('Resale Prices of Flats in each Town')

df\_pricetown = df\_dropmissing[['town','resale\_price']]

df\_pricetown



|  |  |  |
| --- | --- | --- |
|  | **town** | **resale\_price** |
| **0** | ANG MO KIO | 255000.0 |
| **1** | ANG MO KIO | 275000.0 |
| **2** | ANG MO KIO | 285000.0 |
| **3** | ANG MO KIO | 290000.0 |
| **4** | ANG MO KIO | 290000.0 |
| **...** | ... | ... |
| **1245** | YISHUN | 460000.0 |
| **1246** | YISHUN | 500000.0 |
| **1247** | YISHUN | 525888.0 |
| **1248** | YISHUN | 538000.0 |
| **1249** | YISHUN | 550000.0 |

1076 rows × 2 columns

This box chart shows the quartile resale prices of flats in each town. The highest resale price is in Marine Parade, while the lowest is in Yishun. This shows that flats in towns like Marine Parade and Bishan are more desirable and people are willing to pay more of a premium for them. On the other hand, flats in Bukit Batok and Yishun commands a lower price. Flat resale prices in Queenstown and Bukit Merah vary greatly and there may be other factors causing the big differences.

(467 words excluding table)

# Bibliography

Tamboli, N. (2023, July 14). *Effective strategies for Handling Missing Values in Data Analysis*. Retrieved from Analytics Vidhya: https://www.analyticsvidhya.com/blog/2021/10/handling-missing-value/#Why\_Do\_We\_Need\_to\_Care\_About\_Handling\_Missing\_Data