**Declaration Page**

We, members of group 10, do hereby declare that we each contributed to this assignment and that we collectively agree to a shared grade.

|  |  |  |
| --- | --- | --- |
| Name | Contribution | Signature |
| Mohammad Danial Bin Mohammad Ismail (Team Lead) | I did question C |  |
| Ong Lee Zhu | I did question 1 (a) and (b) |  |
| Darrion Goh Ing Hean | I did question D |  |
| Muhammad Yusuf Bin Zainudin | I did question D |  |



**ANL 252**

**PYTHON FOR DATA ANALYTICS**

**Group-Based Assignment**

**July 2023 Presentation**

**Group 10**

|  |  |
| --- | --- |
| **Mohammad Danial Bin Mohammad Ismail** | **J2310430** |
| **Ong Lee Zhu** | **Z2270006** |
| **Darrion Goh Ing Hean** | **E2310655** |
| **Muhammad Yusuf Bin Zainudin** | **Q2111756** |

**Submission Date: 13/10/2023**

Q1 (a)

In order to read the dataset with Python, firstly we need to install the pandas using pip and import pandas in Jupyter: import pandas as pd

Second step is to load a dataset in Python and open it in the format of pandas by covert the GBA\_HDB.csv data file to pandas DataFrame by using the read\_csv () function:

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

imports

The function output: (1250 rows x 11 columns), which means that the dataset is content of 1250 rows and 11 columns. The function of read\_csv( ) is a reader to convert a specific format of data files into pandas DataFrame and to allows pandas to read it’s dataset. Other than excel (or csv) data file, pandas also provide readers to import other sources data file such as SPSS, state, html, and so on by using read\_” source of file”( ). For example, read\_html() for HTML data.

import pandas as pd

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

size = data.size

print("size = {}".format(size))

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

shape = data.shape

print("shape = {}".format(shape))

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

df\_ndim = data.ndim

series\_ndim = data["street\_name"].ndim

print("ndim of dataframe = {}\nndim of series ={}". format(df\_ndim, series\_ndim))

The output of side = 13750, shape = (1250, 11), ndim of dataframe = 2 and ndim of series =1

There are two dimensions in a pandas DataFrame, rows and columns. We can use the function of pandas size( ), pandas shape ( ) and pandas ndim ( ) for dimensionality purposes. Thus, the dimensions of GBA\_HDB.csv can be performed as above.

Q1 (b)

import pandas as pd

C\_Imports = pd.read\_csv("GBA\_HDB.csv")

C\_Imports

C\_Imports.isnull().sum(axis=0)

Output:

month 0

town 0

flat\_type 40

block 0

street\_name 1

storey\_range 0

floor\_area\_sqm 0

flat\_model 0

lease\_commence\_date 0

remaining\_lease 0

resale\_price 134

dtype: int64

In overall from the output as above, the function of .isnull().sumsum(axis=0) can help us to identify the variables with missing value by using Python. There are three variables with missing values which flat\_type (40 missing values/data), street\_name (1 missing value/data), and resale\_price (134 missing values/ data) from total size of 13750 dataset (1250 rows x 11columns).

missings = C\_Imports.isnull().any()

missings

Output:

month False

town False

flat\_type True

block False

street\_name True

storey\_range False

floor\_area\_sqm False

flat\_model False

lease\_commence\_date False

remaining\_lease False

resale\_price True

dtype: bool

Besides that, we can also use .any() to double check and confirm the missing value is exist or not (as above). “True” means missing value is exist and “False” means that missing value is not exist.

Q1 (b)\_continued

import pandas as pd

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

bool\_series = pd.isnull(data["flat\_type"])

data[bool\_series]

import pandas as pd

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

bool\_series = pd.isnull(data["street\_name"])

data[bool\_series]

import pandas as pd

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

bool\_series = pd.isnull(data["resale\_price"])

data[bool\_series]

After using the function of .isnull().sumsum(axis=0), and function of .any() to identified the variables with missing values, we can find out these specified or individual variable with missing value by using the functions of bool\_series (as above). Missing values / data occurs by omitted or no information provided. It is a common mistake and error in reality. Missing values will lead to inaccuracy of data analysis and the outcome of the statistic biased based on the size of the missing values. The more missing values involved, the less accurate results will be.

Q1 (c)

*Deleting missing data*

import pandas as pd

data = {'Set A': [1, 30, None, None, 90],

'Set B': [20, None, 60, 80, None]}

df = pd.DataFrame(data)

df\_deleted\_rows = df.dropna(axis=0)

df\_deleted\_columns = df.dropna(axis=1)

print("Original DataFrame:")

print(df)

print("\nDataFrame with Rows Containing Missing Values Removed:")

print(df\_deleted\_rows)

print("\nDataFrame with Columns Containing Missing Values Removed:")

print(df\_deleted\_columns)

The first method to treat missing data is by deleting the missing value. This method will then delete the missing data allowing a complete set of data. By using .dropna(), it specifies the row with missing values that should be removed. It also combines and remove missing data in rows and columns. All missing data are then deleted by the function .dropna().

*Replacing missing data*

df.fillna(df.mean(), inplace=True)

df.fillna(df.median(), inplace=True)

df.fillna(df.mode(), inplace=True)

Another way to treat missing data is to replace missing values using variable mean, median and mode. By using .fillna(), it replaces all missing values with value that is specified using the variable mean,median and mode. This method is beneficial to use when the missing value is at random. By using this method, the user can still keep the overall distribution of the data.

*Ignoring missing data*

The last method is to ignore the missing data. Ignoring missing value is the easiest method in handling missing values as it does not require much effort to process the data. Though ignoring missing value is a way to treat missing data, it may result in a loss of information by processing the data with missing values. This method is suitable for small number of missing data.

[Word Count: 193 words]

Q1 (d)

A graph of a number of red bars

Description automatically generated with medium confidence

The chart above shows the average price per remaining year of lease per square metre across all flat types for each location. The reason for doing this is because both remaining lease and floor area affect the price of a house. Thus, by doing this, we are essentially trying to get a fairer comparison of house prices strictly by location since it takes out the consideration of remaining lease and floor area.

A general observation that can be made is that the closer the location is to the central business district, the higher the average price. However, there are anomalies. For example, Marine Parade. While Marine Parade is not as near to the central business district as Queenstown, its average price is much higher. There could be both subjective and objective factors causing this.

(134 words)

|  |  |
| --- | --- |
| Town | price\_per\_remaining\_lease\_per\_sqm |
| SEMBAWANG | 46.16136 |
| WOODLANDS | 47.00553 |
| CHOA CHU KANG | 47.4298 |
| BUKIT PANJANG | 49.43549 |
| PUNGGOL | 50.69949 |
| SENGKANG | 51.25136 |
| PASIR RIS | 51.3639 |
| JURONG WEST | 51.89906 |
| YISHUN | 56.85979 |
| BUKIT BATOK | 59.15885 |
| HOUGANG | 60.31989 |
| TAMPINES | 61.7097 |
| JURONG EAST | 64.71766 |
| SERANGOON | 68.75459 |
| BEDOK | 71.81728 |
| ANG MO KIO | 74.48318 |
| BISHAN | 79.1162 |
| CLEMENTI | 79.32078 |
| GEYLANG | 82.42217 |
| KALLANG/WHAMPOA | 83.13479 |
| TOA PAYOH | 84.5987 |
| BUKIT MERAH | 94.01502 |
| QUEENSTOWN | 94.05244 |
| MARINE PARADE | 111.7664 |

Table 1: Table for Average Price per Remaining Lease per sqm by Location

!pip install ipynb

import pandas as pd

import matplotlib.pyplot as plt

import numpy as py

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

hdb.dropna(inplace=True)

hdb["price\_per\_remaining\_lease\_per\_sqm"] = hdb["resale\_price"] / hdb ["remaining\_lease"] / hdb["floor\_area\_sqm"]

hdb = hdb.drop(columns = ["month",

"flat\_model",

"block",

"street\_name",

"storey\_range",

"lease\_commence\_date",

"resale\_price",

"remaining\_lease",

"flat\_type"

])

summary\_hdb = hdb.groupby(["town"])["price\_per\_remaining\_lease\_per\_sqm"].mean().reset\_index()

summary\_hdb = summary\_hdb.sort\_values(by="price\_per\_remaining\_lease\_per\_sqm")

summary\_hdb.to\_csv("Average Price per Remaining Lease per sqm by Location.csv", index=False)

print(summary\_hdb)

plt.figure(figsize=(15, 8))

plt.bar(summary\_hdb["town"], summary\_hdb["price\_per\_remaining\_lease\_per\_sqm"], color="maroon")

plt.title("Average Price per Remaining Lease per sqm by Location")

plt.xlabel("Town")

plt.ylabel("Average Price per Remaining Lease per sqm")

plt.grid(axis='y', linestyle='--', alpha=1)

plt.xticks(rotation=90)

plt.show()

A graph of different colored dots

Description automatically generated

The chart above shows the average floor area of each flat type in each year that their lease commenced. We can observe that each flat type has a specific range of floor area regardless of their lease commencement year. The floor area range for each flat type corresponds to their number of rooms. However, we can see an anomaly in 1969 where the average floor area of 4-room flats was as big as 5-room flats. Another anomaly can be seen in 2007 where the average floor area of 3-room flats was as big as 4-room flats.

Secondly, we can observe that there is a declining trend in average floor area across all flat types since the 1990s. Executive flats saw the sharpest decline in average floor area the newer they are, having an average floor area that is similar to older 5-room flats. The decline in average floor area of all flats could have been a decision made by HDB to maximise the number of units they can build in a given area as land in Singapore is becoming increasingly scarce.

(181 words)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| lease\_commence\_date | 2 ROOM | 3 ROOM | 4 ROOM | 5 ROOM | EXECUTIVE |
| 1967 | 0 | 62.825 | 0 | 0 | 0 |
| 1968 | 38 | 61 | 0 | 0 | 0 |
| 1969 | 49 | 60.83333333 | 122 | 0 | 0 |
| 1970 | 42 | 66 | 0 | 0 | 0 |
| 1971 | 42 | 59.5 | 0 | 0 | 0 |
| 1972 | 0 | 68.71428571 | 0 | 0 | 0 |
| 1973 | 0 | 65.4 | 0 | 0 | 0 |
| 1974 | 0 | 62.875 | 87.71428571 | 0 | 0 |
| 1975 | 0 | 62.71428571 | 86 | 120 | 0 |
| 1976 | 0 | 65.83333333 | 90.33333333 | 120.5 | 0 |
| 1977 | 0 | 66.83333333 | 94 | 120 | 0 |
| 1978 | 44 | 70.92857143 | 94.4 | 0 | 0 |
| 1979 | 0 | 69.69565217 | 92.45454545 | 114.4 | 0 |
| 1980 | 0 | 68.11111111 | 91.25 | 119.3333333 | 0 |
| 1981 | 0 | 68.875 | 92 | 119.3333333 | 0 |
| 1982 | 0 | 67.30769231 | 95.5 | 120.5 | 0 |
| 1983 | 0 | 71.94444444 | 99.33333333 | 120.6 | 0 |
| 1984 | 0 | 70.78125 | 98.48148148 | 131.6 | 154 |
| 1985 | 0 | 69.29268293 | 93.2 | 127.9 | 148 |
| 1986 | 45 | 65.26315789 | 91.53846154 | 123 | 149.8 |
| 1987 | 0 | 70.6 | 92.26666667 | 121.3333333 | 147 |
| 1988 | 0 | 70.55555556 | 95.96 | 122 | 145.3333333 |
| 1989 | 0 | 74 | 104.3333333 | 124 | 153.75 |
| 1990 | 0 | 0 | 102.125 | 121 | 146 |
| 1991 | 0 | 0 | 104 | 0 | 148 |
| 1992 | 0 | 0 | 106.1538462 | 127.6666667 | 148 |
| 1993 | 0 | 73 | 102.9285714 | 127.8461538 | 148.375 |
| 1994 | 0 | 0 | 94 | 0 | 150 |
| 1995 | 0 | 0 | 103.6 | 131.8 | 153 |
| 1996 | 0 | 0 | 103.5333333 | 126.5555556 | 148 |
| 1997 | 0 | 0 | 99.64285714 | 120.3571429 | 140 |
| 1998 | 0 | 73.5 | 99.72222222 | 120.6666667 | 140 |
| 1999 | 0 | 0 | 99.5 | 120.3333333 | 141.25 |
| 2000 | 0 | 0 | 93.94736842 | 115.2380952 | 133.4 |
| 2001 | 0 | 0 | 89.42105263 | 111.16 | 131.6 |
| 2002 | 0 | 60 | 88.2 | 110.3333333 | 126 |
| 2003 | 0 | 0 | 90.2 | 110.95 | 125 |
| 2004 | 0 | 0 | 90.61538462 | 111.5 | 0 |
| 2005 | 0 | 0 | 90.5 | 110.25 | 0 |
| 2006 | 0 | 60 | 89 | 110 | 0 |
| 2007 | 0 | 85.5 | 91.25 | 115 | 0 |
| 2008 | 0 | 0 | 91.25 | 109.75 | 0 |
| 2009 | 0 | 0 | 88.42857143 | 0 | 0 |
| 2010 | 0 | 64 | 93.16666667 | 0 | 0 |
| 2011 | 0 | 68.5 | 93.07692308 | 0 | 0 |
| 2012 | 0 | 67 | 91.25 | 120.5 | 0 |

Table 2: Table for Average Floor Area of Flat Types by Lease Commence Date

!pip install ipynb

import pandas as pd

import matplotlib.pyplot as plt

import numpy as py

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

hdb.dropna(inplace=True)

hdb = hdb.drop(columns=["month",

"block",

"street\_name",

"storey\_range",

"flat\_model",

"resale\_price",

"town",

"remaining\_lease"

])

hdb = hdb.sort\_values(by=["flat\_type", "lease\_commence\_date"])

hdb = hdb.groupby(['flat\_type', 'lease\_commence\_date'])['floor\_area\_sqm'].mean().reset\_index()

hdb.rename(columns={'floor\_area\_sqm': 'average\_floor\_area\_sqm'}, inplace=True)

pivot\_hdb = hdb.pivot(index='lease\_commence\_date', columns='flat\_type', values='average\_floor\_area\_sqm')

pivot\_hdb.fillna(0, inplace=True)

pivot\_hdb.reset\_index(inplace=True)

pivot\_hdb.to\_csv("Average Floor Area of Flat Types by Lease Commence Date.csv", index=False)

print(pivot\_hdb)

hdb.rename(columns={'average\_floor\_area\_sqm': 'floor\_area\_sqm'}, inplace=True)

plt.figure(figsize=(15, 8))

flat\_types = hdb['flat\_type'].unique()

for i, flat\_type in enumerate(flat\_types):

data = hdb[hdb['flat\_type'] == flat\_type]

plt.scatter(data['lease\_commence\_date'], data['floor\_area\_sqm'], label=flat\_type, s=40)

plt.xlabel("Lease Commence Date")

plt.ylabel("Average Floor Area (sqm)")

plt.title("Average Floor Area of Flat Types by Lease Commence Date")

plt.legend(title="Flat Type", loc="upper right")

plt.show()

A graph of blue bars

Description automatically generatedThe chart above shows the average resale price of the flat by their storey range. The reason why we want to know this is the storey range will affect the resale price of the flat. Therefore, by putting the data into the graph, it will show whether the price of the flat will increase or decrease according to the flat’s storey ranges.

From the chart above, we can observe that there is a pattern that there is an increase in the average price of the flat the higher the storey range goes. The storey ranges from 01 to 03 recorded the lowest price while from storey 31 to 33, it recorded the highest price. Hence, we can conclude that the higher the storey range, the resale price of the flat will increase as well.

(134 words)

|  |  |
| --- | --- |
| Storey Range | Resale Price ($) |
| 01 TO 03 | 393219.908257 |
| 04 TO 06 | 406680.595420 |
| 07 TO 09 | 421916.380392 |
| 10 TO 12 | 439169.154639 |
| 3 TO 15 | 487850.597222 |
| 16 TO 18 | 512445.837838 |
| 19 TO 21 | 568567.204211 |
| 22 TO 24 | 647250.000000 |
| 25 TO 27 | 702500.000000 |
| 28 TO 30 | 740000.000000 |
| 31 TO 33 | 766000.000000 |

Table 3: Average of Resale Price according to Storey Range

#Import all necessary libraries

!pip install ipynb

import pandas as pd

import matplotlib.pyplot as plt

import numpy as py

#Read excel file and drop null values

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

hdb.dropna(inplace=True)

#Calculate the average of Resale Price by Storey Range

mean\_y = file\_clean.groupby(['storey\_range'])['resale\_price'].mean()

print(mean\_y)

plt.figure(figsize=(15,6))

#Bar Graph

plt.title("Average Resale Price by Storey Range")

mean\_y.plot(kind = 'bar', xlabel = "Storey Range", ylabel ="Average Resale Price ($)")

**References**

Q1 (a)

Pandas df.size, df.shape and df.ndim Methods

[*https://www.geeksforgeeks.org/python-pandas-df-size-df-shape-and-df-ndim/*](https://www.geeksforgeeks.org/python-pandas-df-size-df-shape-and-df-ndim/)

Q1(b)

Pandas isnull() and notnull() Method

[*https://www.geeksforgeeks.org/python-pandas-isnull-and-notnull/*](https://www.geeksforgeeks.org/python-pandas-isnull-and-notnull/)

Q1(c)

How to Deal with Missing Data in Python ? (Data Science Learner Team. N.A)<https://www.datasciencelearner.com/deal-with-missing-data-python/>

Pandas DataFrame: dropna() function (N.A. 19 August 2022)<https://www.w3resource.com/pandas/dataframe/dataframe-dropna.php>

Q1(d)

Pandas plotting by groupby and average

<https://stackoverflow.com/questions/62983600/plotting-by-groupby-and-average>