

**ANL252 Python for Data Analytics**

**Group-Based Assignment July 2023**

|  |  |
| --- | --- |
| **Name (Group 12)** | **PI No.** |
| Goh Jia Yi Samuel | Q2311483 |
| Tan Wee Huat | K2270304 |
| Chen Shi Yun | Q2311793 |
| Hamreesh S/O Arivalagan | B2210812 |

**Declaration Page**

We, members of group \_\_\_\_\_12\_\_\_\_\_\_\_ , do hereby declare that we each contributed to this assignment and that we collectively agree to a shared grade.

|  |  |  |
| --- | --- | --- |
| Name | Contribution | Signature |
| Samuel (Team Lead) | I did question 1d part 1 & 2 |  |
| Tan Wee Huat | I did question 1d part 3 |  |
| Hamreesh | I did question 1a and 1b |  |
| Chen Shi Yun | I did question 1c |  |

Question 1a)

# Import the necessary libraries

import pandas as pd

# Store the GBA\_HDB.csv file into the variable hdb\_df as a Pandas DataFrame

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

# print the dimensions of the dataframe

rows, columns = hdb\_df.shape

print(f"Number of rows: {rows}")

print(f"Number of columns: {columns}")

Based on the code above, we import the Pandas library as pd. Next, we store the 'GBA\_HDB.csv' file into the variable hdb\_df as a Pandas DataFrame using the read\_csv method. To successfully read the CSV file, you need to have the CSV file and the Jupyter Notebook in the same directory. Finally, we print the dimensions of the DataFrame using the shape method of the Pandas library. The output of the code is as seen below:

Number of rows: 1250

Numbe of columns: 11

Question 1b)

hdb\_df.isna().sum()

Based on the code above, we use the Pandas library’s methods isna() and sum() to find the number of missing values for a given variable. The isna() method which stands for ‘*is not available’* is used to identify missing or NA (Not Available) values in the DataFrame*.* It returns a Boolean (True or False), where True means that the value is missing (NaN or None). The sum() method counts the number of True values for each column (variable).

As seen in the output of the code below, the variables with missing values are ‘flat\_type’, ‘street\_name’, and ‘resale\_price’:

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

It is necessary to address missing values in Python because of several reasons. First, missing values in a dataframe could distort statistical measures and relationships, causing inaccurate analysis. Next, handling missing values is necessary to avoid bias in the analysis. For example, if one chooses to drop the rows of a dataframe with missing values, it may disproportionately affect the results of certain groups (labels). Finally, if the dataset is to be used in machine learning models, it is essential to handle missing data as machine learning models, in general, are not capable of handling missing data.

Question 1c)

In addressing the missing data within our HDB resale flat dataset, specific strategies have been tailored to the nature and significance of each feature with incomplete records. For the flat\_type column, which has 40 missing entries, a feasible approach is to utilize the mode — the most frequently occurring flat type in the dataset. This method assumes that the most common flat type is a reasonable representation of any missing data, especially if a certain type of flat dominates the dataset in frequency.

In contrast, the street\_name column has a solitary missing value. Given the minimal impact of a single record on the dataset's overall size and subsequent analysis, it's justifiable to simply remove the row containing this missing value. This ensures the integrity of the dataset while sacrificing negligible data.

Lastly, the resale\_price column poses a unique challenge with 134 missing values. Here, a nuanced approach is adopted: we can impute the missing values using the median resale price, but with a twist. Instead of resorting to the median resale price of the entire dataset, a more refined method would be to determine the median resale price specific to the corresponding town and flat\_type of the missing record. This rationale stems from the observation that resale prices can be influenced significantly by both the town and the type of flat, making this imputation strategy potentially more accurate and context-aware than a blanket median or mean imputation for the entire dataset.

# Impute remaining `resale\_price` missing values using the overall median resale price

overall\_median\_price = hdb\_data['resale\_price'].median()

hdb\_data['resale\_price'].fillna(overall\_median\_price, inplace=True)

# Check if all missing values have been treated

final\_remaining\_missing\_values = hdb\_data.isnull().sum()

final\_remaining\_missing\_values

Question 1d)

**Box plot**

# Import the necessary libraries

import pandas as pd

import matplotlib.pyplot as plt

# Read the data from the CSV file into a DataFrame

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

# Extract the "floor\_area\_sqm" column from the DataFrame

floor\_area\_sqm = df ["floor\_area\_sqm"]

# Set the title and labels for the box plot

plt.title('Box Plot of Floor Area (sqm)')

plt.ylabel('Floor Area (sqm)')

# Create and display the box plot of floor area data

plt.boxplot(floor\_area\_sqm)

# Calculate and print summary statistics for the floor\_area\_sqm column

floor\_area\_stats = floor\_area\_sqm.describe().reset\_index()

floor\_area\_stats.columns = ['Statistic', 'Value']

print(floor\_area\_stats)

# Calculate quartiles and interquartile range (IQR) to identify outliers

q1 = floor\_area\_sqm.quantile (q = .25)

q3 = floor\_area\_sqm.quantile (q = .75)

iqr = q3 - q1

# Determine outliers based on the IQR method

outliers = (floor\_area\_sqm < q1 - 1.5 \* iqr) | (floor\_area\_sqm > q3 + 1.5 \* iqr)

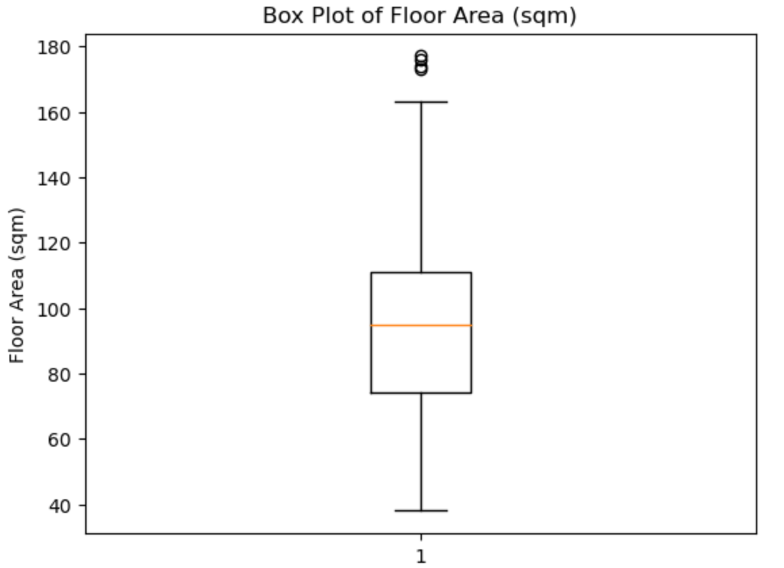
# Print a boolean Series indicating whether each data point is an outlier or not

print(outliers)

# Create a new boolean Series to check if each data point is an outlier (True) or not (False)

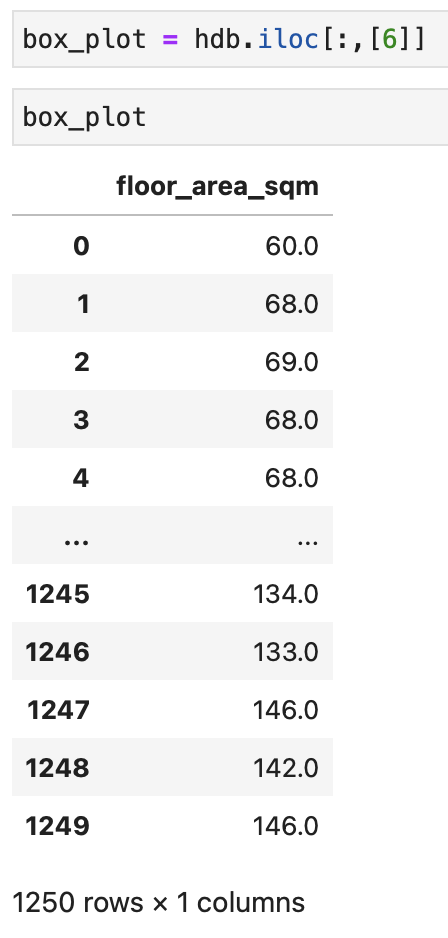
outliers\_1 = floor\_area\_sqm[outliers] == "True"

print(outliers\_1)



The majority of the data points are concentrated between 74 and 111, with median value around 95. The interquartile range is 37 which makes the lower outlier limit at 18.5 and the higher outlier limit at 166.5. With a minimum of 38 and a maximum of 177, this indicates a wide range of housing sizes. Comparably, the slightly larger mean of 96.6 than median suggests that house sizes tend to be slightly larger

There are several outliers identified at 173,174,176 and 177 which are notably larger than typical floor area observed in the dataset. Those houses above 170 sqm are rare finds within the HDB property market. Hence, with the presence of outliers, the median and interquartile range will be better statistics to observe.



**Scatter plot**

# Import the necessary libraries

import pandas as pd

import matplotlib.pyplot as plt

# Read the data from the CSV file into a DataFrame

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

# Extract the 'remaining\_lease' and 'resale\_price' columns from the DataFrame

remaining\_lease = df ["remaining\_lease"]

resale\_price = df ["resale\_price"]

# Create a scatter plot DataFrame with the selected columns (9 and 10)

scatter\_plot = df.iloc[:,[9,10]]

# Calculate the mean value of 'resale\_price' and fill missing values with this mean

mean\_values = resale\_price.mean()

resale\_price = resale\_price.fillna(mean\_values)

# Count the number of missing values in 'resale\_price' and print the count

missing\_count = resale\_price.isna().sum()

print(missing\_count)

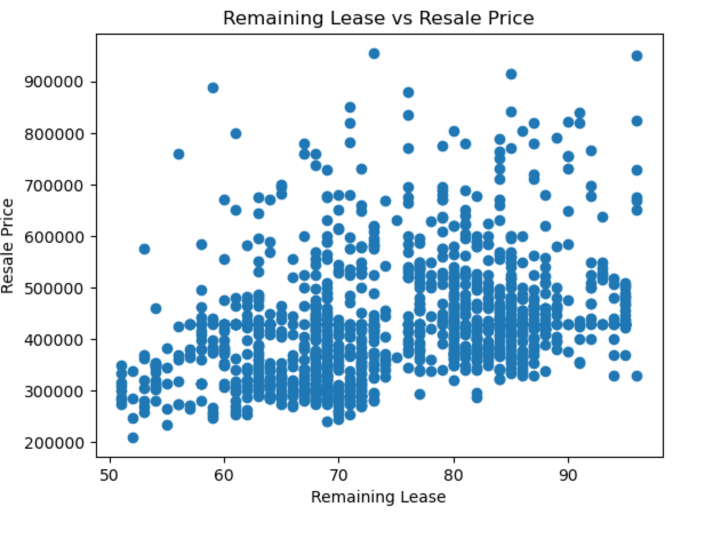
# Create a scatter plot to visualise the relationship between 'remaining\_lease' and 'resale\_price'

plt.scatter (remaining\_lease , resale\_price)

plt.xlabel('Remaining Lease')

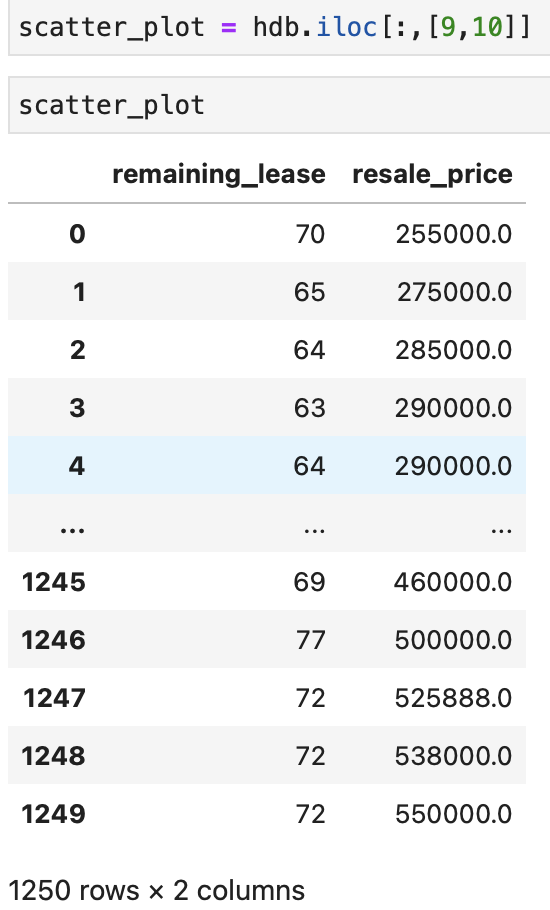
plt.ylabel('Resale Price')

plt.title('Remaining Lease vs Resale Price')



The missing data has been filled with the mean to reduce the overall variance in the dataset. This can make the data more predictable and lead to more statistical inferences.

There is a positive correlation between resale prices and the remaining lease period with a few potential outliers which suggests that on average HDB flats with a longer remaining lease period tend to have higher resale prices. A large amount of data is concentrated between resale prices of 300,000 and 500,000. The close clustering suggests a low variability in the data. The data suggests that buyers may be willing to pay more for flats with longer leases.



**Histogram**

# Import the necessary libraries

import pandas as pd

import matplotlib.pyplot as plt

# Read the data from the CSV file into a DataFrame

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

# Extract the resale price column

resale\_prices = df['resale\_price']

# Define price intervals

price\_range = [200000, 250000, 300000, 350000, 400000, 450000, 500000, 550000, 600000, 650000]

# Create a histogram with price intervals

plt.figure(figsize=(10, 6))

plt.hist(resale\_prices, bins=price\_range, edgecolor='k', alpha=0.7)

plt.xlabel('Resale Price')

plt.ylabel('Frequency')

plt.title('Histogram of Resale Prices by Price Bands')

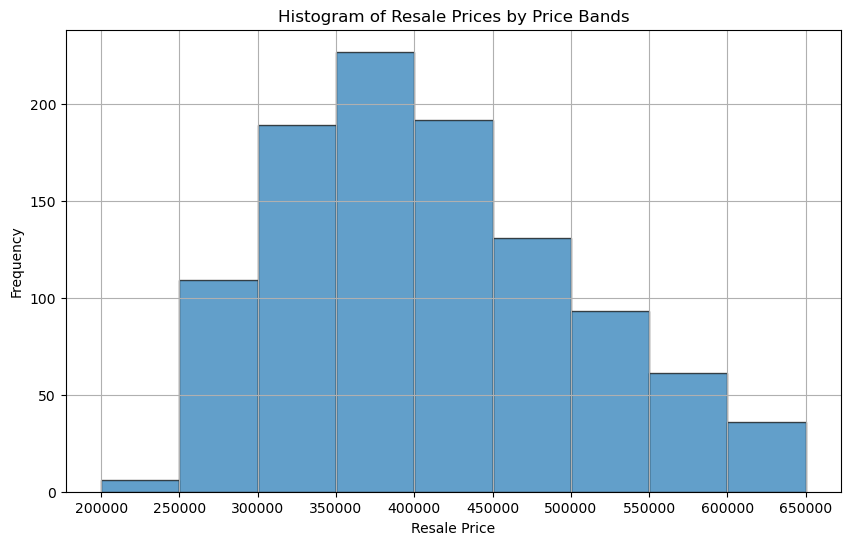
plt.grid(True)

# Add labels to the x-axis indicating the price intervals

plt.xticks(price\_range)

# Histogram

plt.show()



This histogram most resembles a distribution that is skewed more to the right as visually, the data distribution has a longer right tail and a shorter left tail. There are more data points on the left side of the histogram and the frequency of the values decreases gradually for higher values. This suggests that the dataset contains outliers or unusually high values that push the mean and right tail to higher values. We can observe an anomaly for the dataset between 200000 and 250000 with a steep increase. With this right-skewed distribution, this could suggest that higher-priced HDBs could be outliers that result in this uneven distribution of values. However, there are some resale values that are absent which could influence the distribution and shape of the histogram.