GROUP ASSIGNMENT PART 1

CT032-3-3-FAI

Further Artificial Intelligence

APU3F2209CSIS & APD3F2209CSIS

HAND IN DATE: 15 DECEMBER 2022

WEIGHTAGE: 50%

TP Number	Student Name
TP056066	Yan Mun Kye
TP061524	Hor Shen Hau
TP056267	Tan Sheng Jeh
TP060810	Sia De Long

Table of Contents

1.0 Introduction

In this era of globalization, interest on prediction, classification, and clustering modelling have drawn significant growing amounts of attention from researchers among the world. This is because the data generated by both human and machine is increasing to where it could not be easily absorbed, interpreted and make a critical decision from it by a human ability. Therefore, those three modelling are being wide use by the world which need to be improve to maximise the efficiency and effective for them (NetApp, n.d.).

First and foremost, predictive modelling is one of the statistical techniques that full use of the past as well as current existing for enabling the machine learning and data mining to anticipate and forecast the most likely happening outcomes in the future. The way of how the concept work is by analysing the past and current existing data then incorporating what it discovered into a model which is generated to predict expected events, while it can be apply on almost anything that have the need to predict. (Ali, 2020).

Moving on, classification modelling is one of the techniques that belongs under supervised machine learning process. The process of it is involving predicting the class of given data points where the classes can be targets, labels or categories depend on wide-variety of tasks which it allow to implemented (Asiri, 2022), which proved that it is using supervised learning method since the definition of it is using labelled datasets to train algorithms in order to classify data or predict possible events (IBM Cloud Education, 2020).

Last but not least, clustering modelling is the opposite of classification modelling technique which mean it is one of the unsupervised machine learning techniques. It can be used to identifying and grouping similar data points on a graph that come from a large dataset without the need to concern about specifying possible outcome for the model. Hence, it is frequently used to group data into structures for the purpose of easing the effort to understand and manipulate (DATA SCIENCE, 2020).

In this assignment, the team will be focusing on prediction modelling where the team will develop a system that learn the skill to predict the housing price. After a complete trained model is created, the model will be receiving input related to house attributes which enabling the model to predict the price for the given attributes and the confidence for the prediction is expected to be above seventy percent.

1.1 Problem Statement

According to the study, purchasing a 60 square metre apartment is now out of reach for those with average yearly wages in the professional service sector in the majority of cities worldwide based on the Global Real Estate Bubble Index 2022. A professional service sector worker that works in professional service sector can now only afford one-third less housing space than they could before the COVID-19 pandemic because of ultra-cheap financing circumstances and demand that is surpassing supply. Prices have soared by an average of 60% in inflation-adjusted terms in cities that being described as bubble risk zones while actual wages and rentals have increased by just approximately 12%. The cities under the study have a statistic that nominal house price growth has increased to 10% between mid-2021 to mid-2022, which is the most significant increase since 2007 which is the year before the previous financial crisis, while Cities in North America experienced the regional price growth of nearly 15%. However, in Tokyo, rents are now generally 7% higher than they were prior to the pandemic because a substantial recovery came after the decline happened in 2020 (Harper, 2022). To conclude, the housing price is very unstable for the past few years since the pandemic situation happened, so the need of predicting the housing price precisely become crucial to overcome the issue for many stakeholders involved to it and to help preventing the housing bubble to burst.

1.2 Chosen Dataset

For the purpose to train a prediction model, a well-chosen dataset is important and crucial for the accuracy of the result, while it should always relate to the target field to achieving the objectives. Hence, the team make a decision that the dataset will be picking on the trusted and reliable website which is suggested by the tutor of the module named Kaggle.

With the discussion among the team members on every possible consideration for the assignment, the chosen dataset would be Housing Prices Dataset authored by (M Yasser, n.d.). The reason of the decision including the update frequency of the dataset which is updating annually meaning that the dataset might be updated to a more appropriate dataset to use. Besides that, the source of the dataset is from Google using the collection methodology of research while it is licensed by CC0: Public Domain which saying that the dataset is reliable and open to use without any copyright issue. Most importantly, the dataset will be helping the team on study for the problem statement as the dataset is described as a dataset that used for a simple yet challenging regression problem project which will predict the housing price based on certain attributes, while the dataset is small but it has strong multicollinearity between attributes that affecting the housing price causing the complexity increase and making it more challenging. To show the integrity of the dataset the activity overview is shown as the figure 1 below.

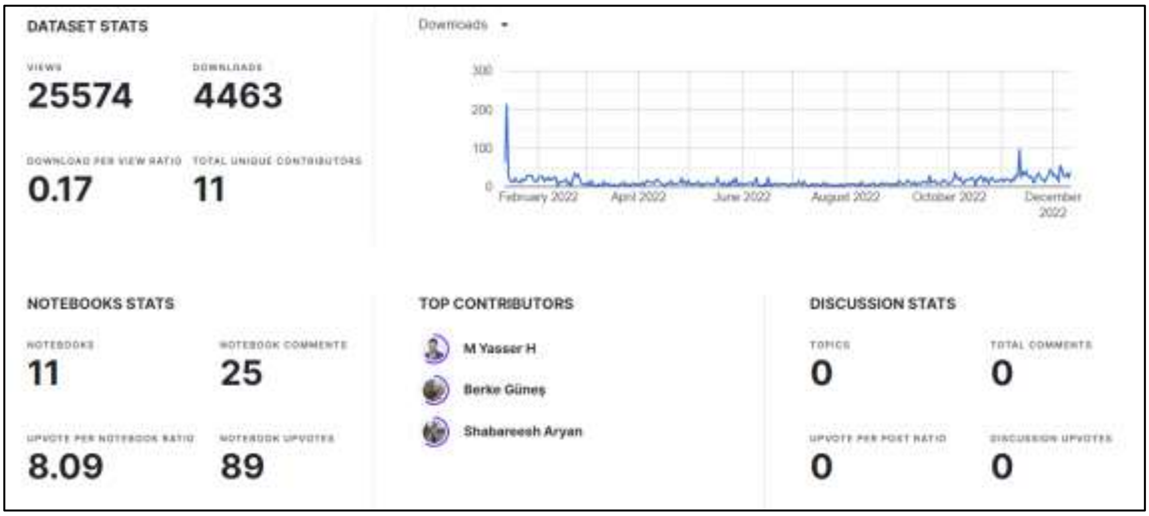


Figure 1: Dataset Activity Overview

The dataset will be consisting 500 rows of data carrying the housing price and many attributes that followed with it which are shown as the figure 2 below.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N
1	price	area	bedrooms	bathrooms	stories	mainroad	guestroom	basement	hotwaterheating	airconditioning	parking	prefarea	furnishingstatus	
2	13300000	7420	4	2	3	yes	no	no	no	yes	2	yes	furnished	
3	12250000	8960	4	4	4	yes	no	no	no	yes	3	no	furnished	
4	12250000	9960	3	2	2	yes	no	yes	no	no	2	yes	semi-furnished	
5	12215000	7500	4	2	2	yes	no	yes	no	yes	3	yes	furnished	
6	11410000	7420	4	1	2	yes	yes	yes	no	yes	2	no	furnished	
7	10850000	7500	3	3	1	yes	no	yes	no	yes	2	yes	semi-furnished	
8	10150000	8580	4	3	4	yes	no	no	no	yes	2	yes	semi-furnished	
9	10150000	16200	5	3	2	yes	no	no	no	no	0	no	unfurnished	

Figure 2: Dataset Attributes

To summarise the column the data type is categorise to three different category which are integer, sentimental and fixed category where integer data will only store number value, sentimental data will only store value of “yes” or “no” and fixed category data will store data that defined to the column.

Integer

- **price** – the total housing price
- **area** – the area of the house
- **bedroom** – the number of bedrooms built in the house
- **bathrooms -** the number of bathrooms built in the house
- **stories** – the number of stories in the house
- **parking** – the number of parking available specific for the house owner

Sentimental (yes or no)

- **mainroad** – indicate that the house is connecting to main road
- **guestroom -** indicate that the house is having a guestroom
- **basement**- indicate that the house is having a basement
- **hotwaterheating -** indicate that the house is having an infrastructure to heat water
- **airconditioning** - indicate that the house is having an air conditioner
- **prefarea-** indicate that the house is having a prefarea

Fixed Category

- **furnishingstatus** – indicate that the house is furnished with the defined value of (furnished, semi-furnished and unfurnished)

1.3 Data Preprocessing and Exploration

Hor Shen Hau

Importing Libraries

```
#import libraries
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
```

Before anything is done, the necessary libraries that will be used has to be imported.

Reading the selected dataset

```
In [7]: housing = pd.read_csv (r'D:\University Course Materials APU\Year 3 Semester 1\FAI\Group Assignment\housing\Housing.csv')
        housing.head()
```

Out[7]:

	price	area	bedrooms	bathrooms	stories	mainroad	guestroom	basement	hotwaterheating	airconditioning	parking	prefarea	furnishingstatus
0	13300000	7420	4	2	3	yes	no	no	no	yes	2	yes	furnished
1	12250000	8960	4	4	4	yes	no	no	no	yes	3	no	furnished
2	12250000	9960	3	2	2	yes	no	yes	no	no	2	yes	semi-furnished
3	12215000	7500	4	2	2	yes	no	yes	no	yes	3	yes	furnished
4	11410000	7420	4	1	2	yes	yes	yes	no	yes	2	no	furnished

```
In [8]: print(housing.shape)

(545, 13)
```

The housing dataset is read using the read_csv function and the columns and rows are read into a dataframe. The dataset is previewed using housing.head() to see if the dataset has been read correctly. The shape function is then used to see the number of rows and columns present in the dataset.

Summary of the housing dataset

```
In [28]: housing.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 545 entries, 0 to 544
Data columns (total 13 columns):
 #   Column            Non-Null Count  Dtype 
---  ------            --------------  ----- 
 0   price             545 non-null    int64 
 1   area              545 non-null    int64 
 2   bedrooms          545 non-null    int64 
 3   bathrooms         545 non-null    int64 
 4   stories           545 non-null    int64 
 5   mainroad          545 non-null    object
 6   guestroom         545 non-null    object
 7   basement          545 non-null    object
 8   hotwaterheating   545 non-null    object
 9   airconditioning   545 non-null    object
 10  parking           545 non-null    int64 
 11  prefarea          545 non-null    object
 12  furnishingstatus  545 non-null    object
dtypes: int64(6), object(7)
memory usage: 55.5+ KB
```

The .info() function is used to get a summary of the dataset and to check for any null values in the dataset. The summary shows that there are no null values in the dataset.

```
In [9]: #separating categorical and numerical data in the housing dataset
categorical_vars = housing.columns[housing.dtypes=="object"]
numerical_vars = housing.columns[housing.dtypes!="object"]

print(categorical_vars)
print(numerical_vars)

Index(['mainroad', 'guestroom', 'basement', 'hotwaterheating',
       'airconditioning', 'prefarea', 'furnishingstatus'],
      dtype='object')
Index(['price', 'area', 'bedrooms', 'bathrooms', 'stories', 'parking'], dtype='object')

In [10]: #to determine the % of missing data for each column in the housing dataset (categorical_var)
housing[categorical_vars].isnull().sum().sort_values(ascending=False)/len(housing)

Out[10]: mainroad            0.0
guestroom           0.0
basement            0.0
hotwaterheating     0.0
airconditioning     0.0
prefarea            0.0
furnishingstatus    0.0
dtype: float64

In [11]: housing[numerical_vars].isnull().sum().sort_values(ascending=False)/len(housing)

Out[11]: price        0.0
area         0.0
bedrooms     0.0
bathrooms    0.0
stories      0.0
parking      0.0
dtype: float64

In [12]: # 0% null values in the dataset.
```

The categorical and numerical data in the housing dataset is then separated and checked separately for null and missing values.

In [13]:
```
# checking for highly correlated variables which if present may cause the trained model to overfit.
housing.corr()
```

Out[13]:

	price	area	bedrooms	bathrooms	stories	parking
price	1.000000	0.535997	0.366494	0.517545	0.420712	0.384394
area	0.535997	1.000000	0.151858	0.193820	0.083996	0.352980
bedrooms	0.366494	0.151858	1.000000	0.373930	0.408564	0.139270
bathrooms	0.517545	0.193820	0.373930	1.000000	0.326165	0.177496
stories	0.420712	0.083996	0.408564	0.326165	1.000000	0.045547
parking	0.384394	0.352980	0.139270	0.177496	0.045547	1.000000

In [14]:
```
# no collinearity is seen, model will not overfit.
```

In [15]:
```
logprice = np.log(housing['price'])
logprice.skew()
#skewness coefficient of 0.14
```

Out[15]: 0.14086257299872787

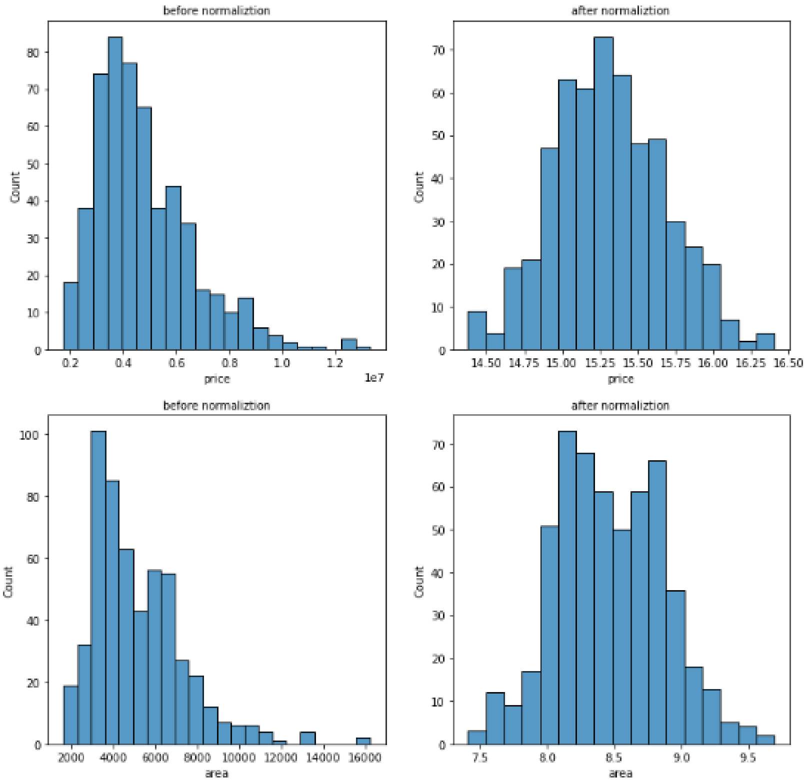
In [16]:
```
logarea = np.log(housing['area'])
logarea.skew()
#skewness coefficient of 0.13
```

Out[16]: 0.1335202187004955

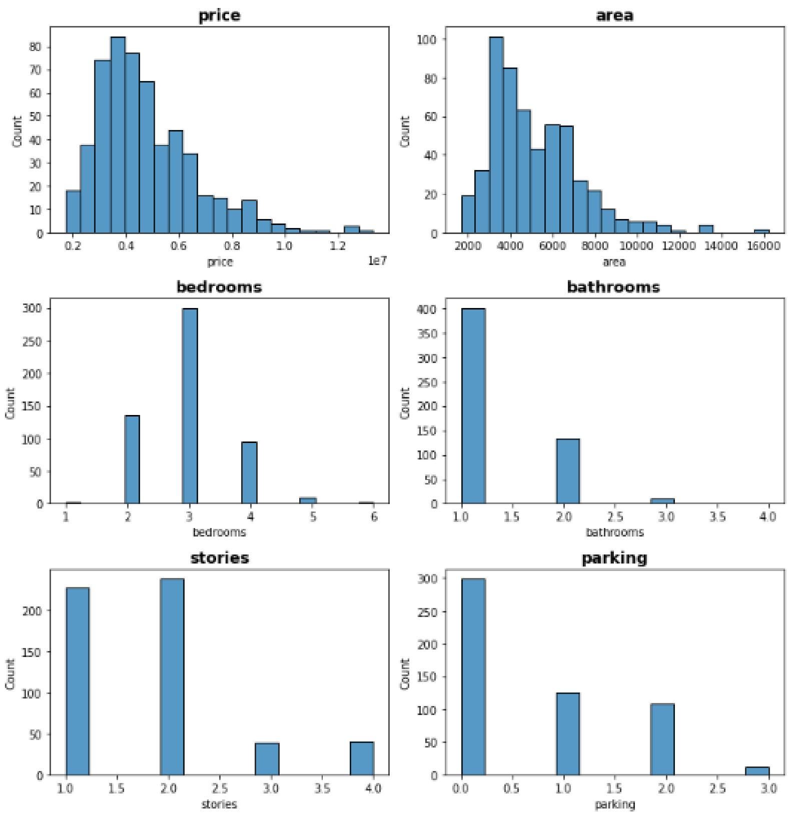
Checking for highly correlated variables in the dataset to check if there are any highly correlated variables which may end up with the model overfitting. The checking of the skewness of the data is also checked and verified by the plotting of the graph before and after normalization of the area data.

In [17]:
```
plt.figure(figsize=[12,12])
plt.subplot(2,2,1)
sns.histplot(x=housing.price)
plt.title("before normaliztion",fontsize=10)
plt.subplot(2,2,2)
sns.histplot(x=(np.log(housing['price'])))
plt.title("after normaliztion",fontsize=10)

plt.subplot(2,2,3)
sns.histplot(x=housing.area)
plt.title("before normaliztion",fontsize=10)
plt.subplot(2,2,4)
sns.histplot(x=(np.log(housing['area'])))
plt.title("after normaliztion",fontsize=10)

#it is seen that after log transformation of the price data, the data is normalized with a skewness coefficent of 0.14.
#this is further confirmed with a visualization of the area data.
```

Out[17]:
```
Text(0.5, 1.0, 'after normaliztion')
```

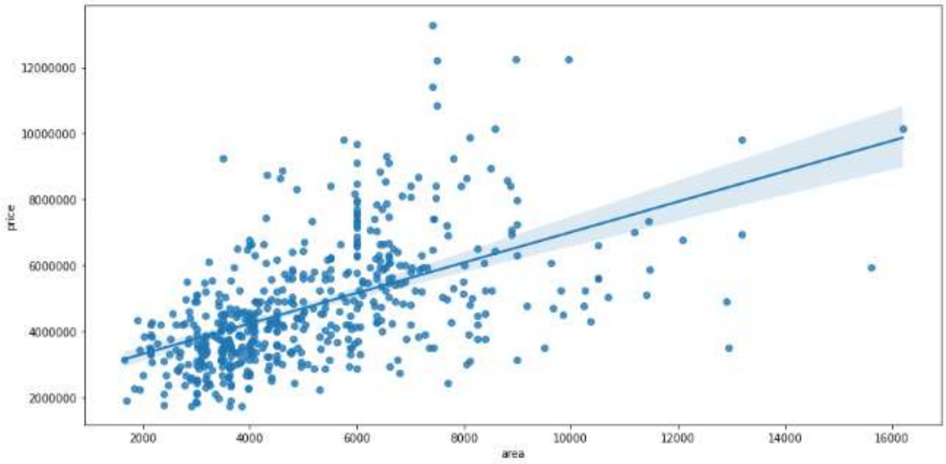


```
In [18]: plt.figure(figsize=(10,20))
         for i,num in enumerate(housing.select_dtypes(exclude='O')):
             ax = plt.subplot(6,2, i + 1)
             sns.histplot(x=housing[num])
             plt.title(num, fontsize=14, fontweight='bold')
         plt.tight_layout()
         plt.show()
```



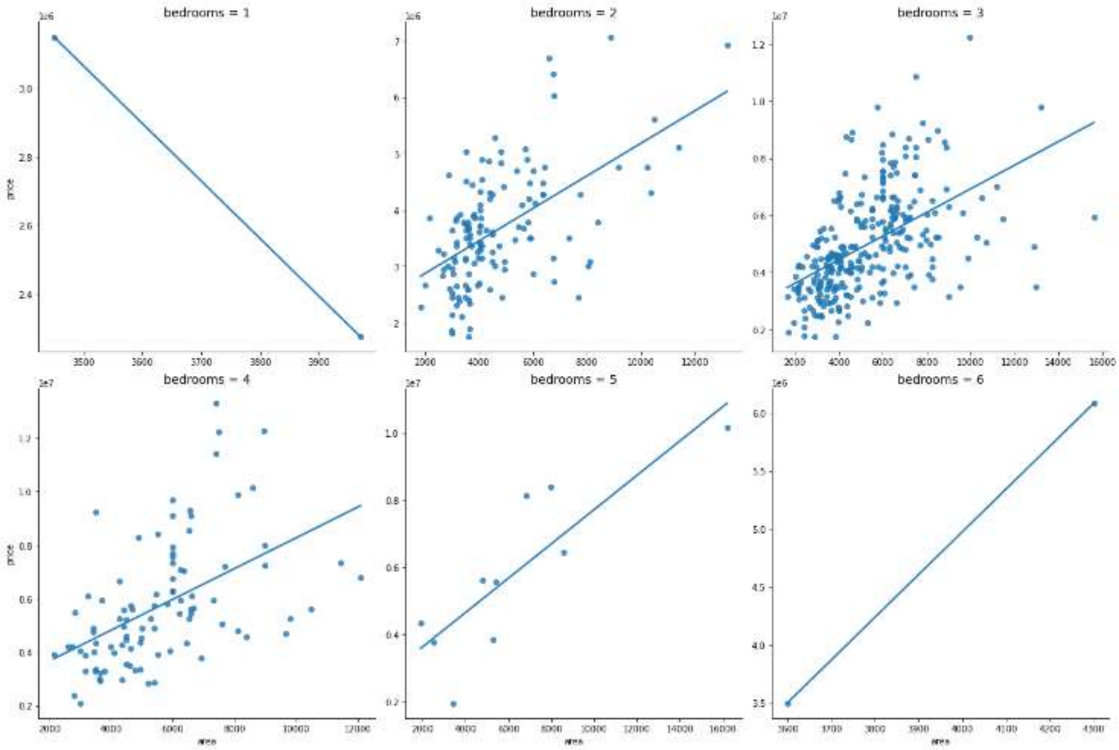
Visualization of every data column in the dataset.

```
In [19]: plt.figure(figsize=(14,7))
         sns.regplot(x=housing['area'], y=housing['price'])
         plt.ticklabel_format(style='plain',axis='y')
         # it can be noted that the price of the houses have a positive correlation to that of the area (size) of the house with some out
```



There is positive correlation of price to that of the area in the housing dataset with some noticeable outliers.

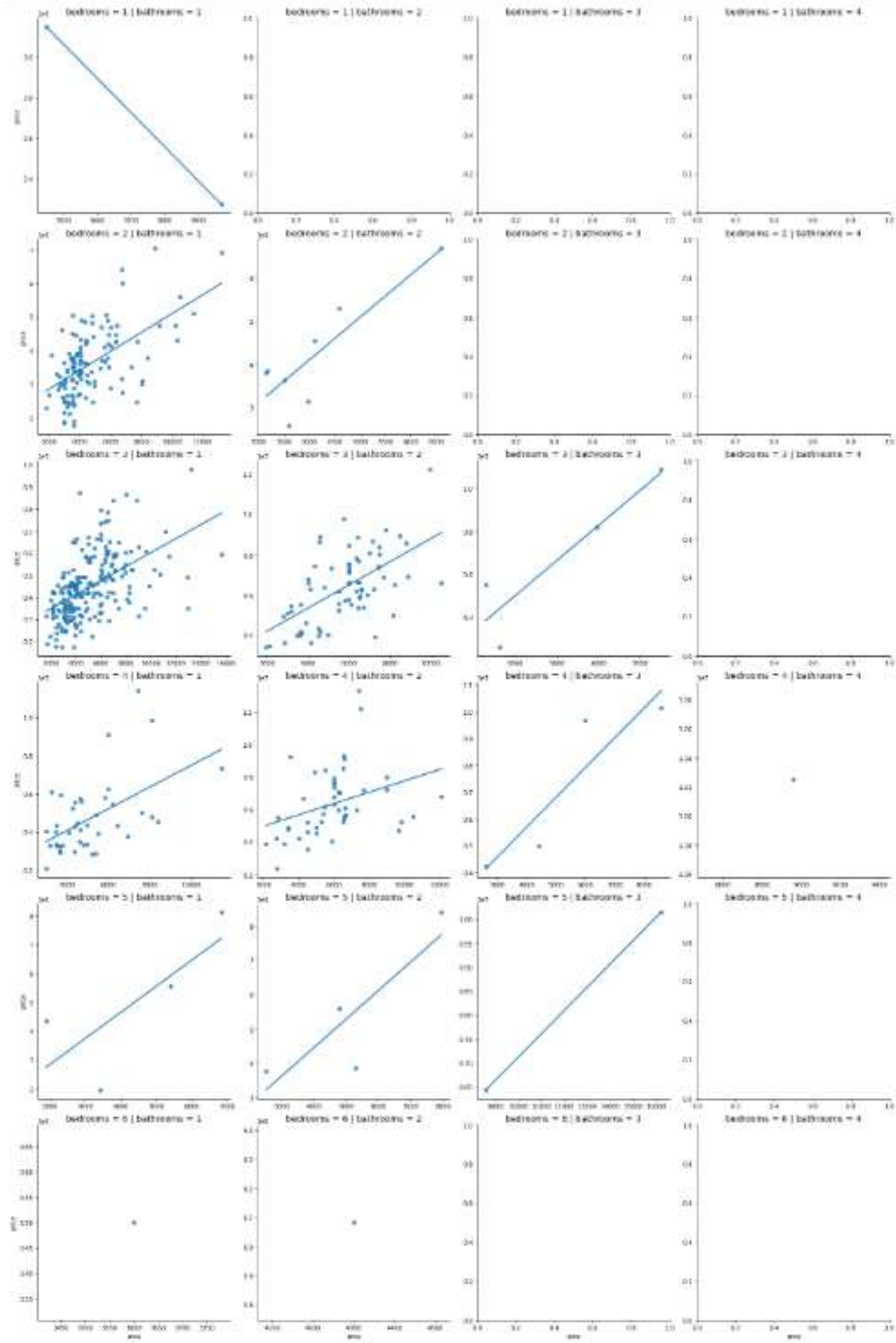
In [20]:
```
priceperroom = sns.lmplot(data=housing,x='area', y='price', col='bedrooms',col_wrap=3
             ,facet_kws=dict(sharex=False, sharey=False), ci=None, height=6, aspect=1)
priceperroom.set_titles(size=14)
plt.show()
#the price of rooms by area has a positive correlation
```



There is also a noticeable positive correlation in the price per room of the housing.

In [21]:

```
priceperbathbed = sns.lmplot(data=housing,x='area', y='price', row='bedrooms',col='bathrooms'
            ,facet_kws=dict(sharex=False, sharey=False), ci=None)
priceperbathbed.set_titles(size=14)
plt.show()
# the price of the number of rooms and bathrooms in a house has a positive correlation
```



Price per bedroom and bathroom can also be seen with a positive correlation of price in the visualization of the data.

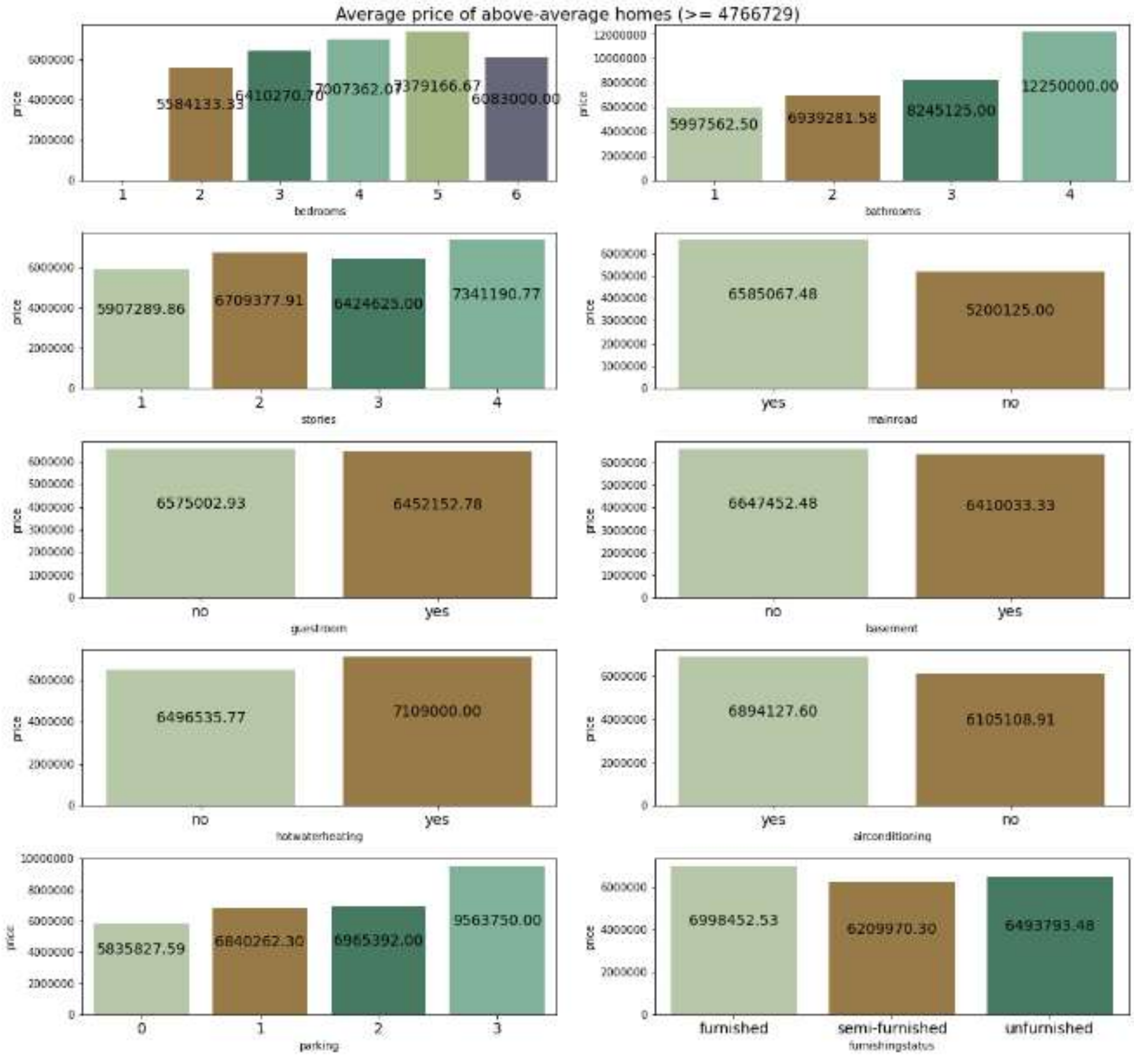
```
In [22]: # minimum, maximum, mean, median price
         print('Minimum Price:',housing['price'].min())
         print('Maximum Price:',housing['price'].max())
         print('Mean Price:',housing['price'].mean()) #average price of houses
         print('Median Price:',housing['price'].median())
         print('Standard Deviation of Price:',housing['price'].std())

         Minimum Price: 1750000
         Maximum Price: 13300000
         Mean Price: 4766729.247706422
         Median Price: 4340000.0
         Standard Deviation of Price: 1870439.615657394
```

Checking for the minimum price, maximum price, average (mean) price, median price and the standard deviation of housing price based on the dataset.

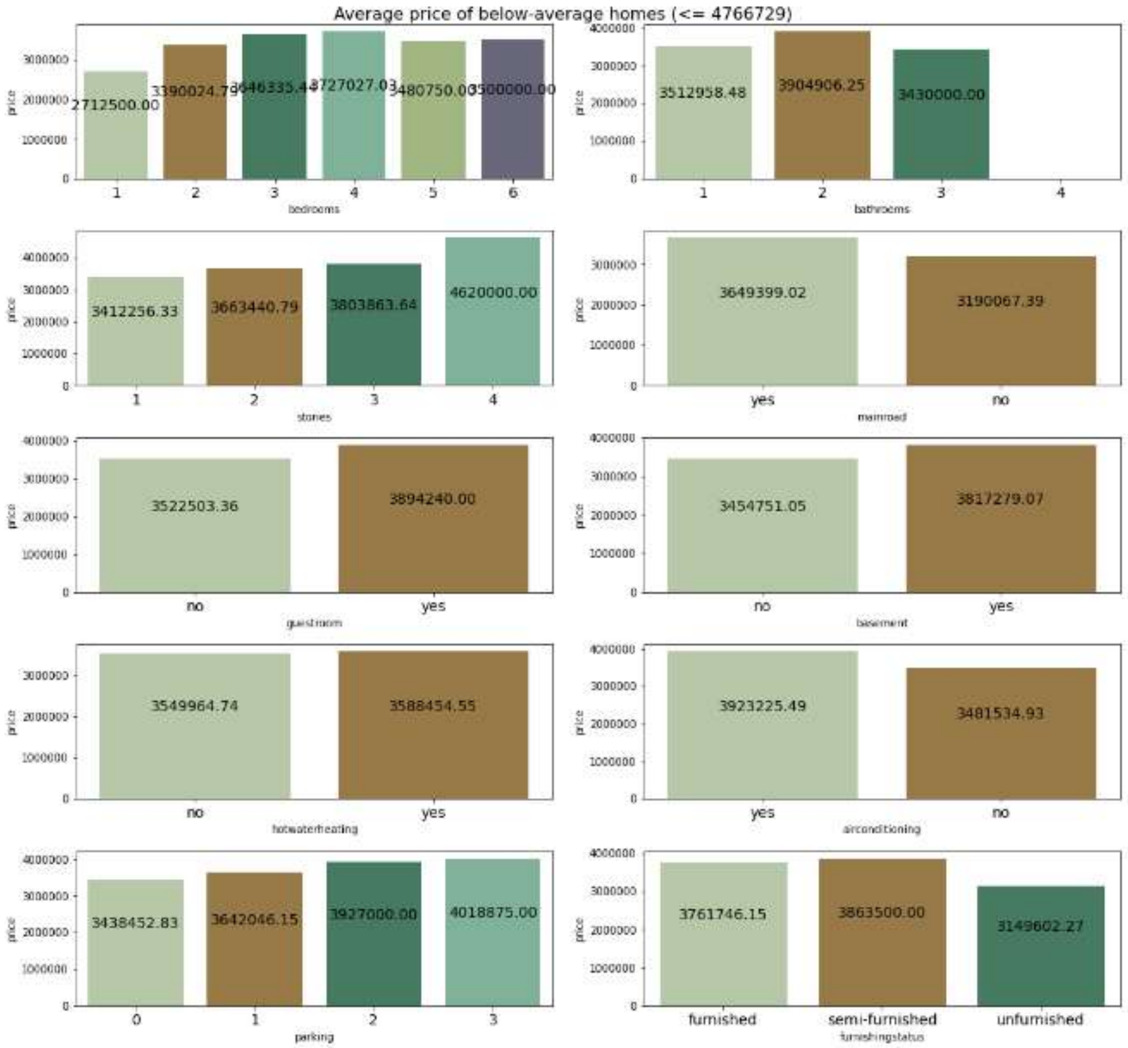
```
In [23]: #separating houses above the average (mean) price and below.
         aboveavg = housing[housing['price']>4766729]
         belowavg = housing[housing['price']<= 4766729]
         print('Percentage of houses with above avg price:',len(aboveavg) / len(housing) * 100,'%')
         print('Percentage of houses with above avg price:', len(belowavg) / len(housing) * 100,'%')

         Percentage of houses with above avg price: 40.73394495412844 %
         Percentage of houses with above avg price: 59.26605504587156 %
```

Finding the percentage of houses with above average price and below average price in the dataset.

In [26]:
```python
#above average houses
categories = ['bedrooms', 'bathrooms', 'stories', 'mainroad',
        'guestroom', 'basement', 'hotwaterheating', 'airconditioning',
        'parking', 'furnishingstatus']


plt.figure(figsize=(15,30))
plt.suptitle('Average price of above-average homes (>= 4766729)', fontsize=16, y=0.99)
for i, d in enumerate(categories):
    ax = plt.subplot(11,2,i + 1)
    sns.barplot(x=housing[d], y=aboveavg['price'], ci=None, palette=color)
    plt.ticklabel_format(style='plain',axis='y')
    plt.xticks(fontsize=14)
    for n in ax.containers:
        ax.bar_label(n,fmt='%.2f',label_type='center',fontsize=14, padding=18)
plt.tight_layout()
plt.show()
```



In [27]:
```
#below average houses
categories = ['bedrooms', 'bathrooms', 'stories', 'mainroad',
        'guestroom', 'basement', 'hotwaterheating', 'airconditioning',
        'parking', 'furnishingstatus']


plt.figure(figsize=(15,30))
plt.suptitle('Average price of below-average homes (<= 4766729)', fontsize=16, y=0.99)
for i, d in enumerate(categories):
    ax = plt.subplot(11,2,i + 1)
    sns.barplot(x=housing[d], y=belowavg['price'], ci=None, palette=color)
    plt.ticklabel_format(style='plain',axis='y')
    plt.xticks(fontsize=14)
    for n in ax.containers:
        ax.bar_label(n,fmt='%.2f',label_type='center',fontsize=14, padding=18)
plt.tight_layout()
plt.show()
```



It is noticeable that regardless of houses above average price or below, houses with access to mainroad, basement, guestroom, hot water heating, air conditioning and parking are generally of higher price. It can also be deduced that the houses that are below average with 2 bathrooms cost the most while the above average houses with 4 bathrooms cost the most. It can be seen from the plots above that both below and above average houses with 4 stories have the highest average price.

Houses with above-average price and 5 bedrooms have the highest average price while houses below average price with 4 bedrooms have the highest average price.

Yan Mun Kye

```
In [2]: import pandas as pd
        import seaborn as sns
        import matplotlib.pyplot as plt
```

Importing the necessary libraries for EDA operations.

```
In [3]: df = pd.read_csv("data/Housing.csv")
        df.head()
```

Out[3]:

	price	area	bedrooms	bathrooms	stories	mainroad	guestroom	basement	hotwaterheating	airconditioning	parking	prefarea	furnishingstatus
0	13300000	7420	4	2	3	yes	no	no	no	yes	2	yes	furnished
1	12250000	8960	4	4	4	yes	no	no	no	yes	3	no	furnished
2	12250000	9960	3	2	2	yes	no	yes	no	no	2	yes	semi-furnished
3	12215000	7500	4	2	2	yes	no	yes	no	yes	3	yes	furnished
4	11410000	7420	4	1	2	yes	yes	yes	no	yes	2	no	furnished

```
In [4]: df.describe()
```

Out[4]:

	price	area	bedrooms	bathrooms	stories	parking
count	5.450000e+02	545.000000	545.000000	545.000000	545.000000	545.000000
mean	4.766729e+06	5150.541284	2.965138	1.286239	1.805505	0.693578
std	1.870440e+06	2170.141023	0.738064	0.502470	0.867492	0.861586
min	1.750000e+06	1650.000000	1.000000	1.000000	1.000000	0.000000
25%	3.430000e+06	3600.000000	2.000000	1.000000	1.000000	0.000000
50%	4.340000e+06	4600.000000	3.000000	1.000000	2.000000	0.000000
75%	5.740000e+06	6360.000000	3.000000	2.000000	2.000000	1.000000
max	1.330000e+07	16200.000000	6.000000	4.000000	4.000000	3.000000

```
In [5]: df.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 545 entries, 0 to 544
Data columns (total 13 columns):
 #   Column            Non-Null Count  Dtype 
---  ------            --------------  ----- 
 0   price             545 non-null    int64 
 1   area              545 non-null    int64 
 2   bedrooms          545 non-null    int64 
 3   bathrooms         545 non-null    int64 
 4   stories           545 non-null    int64 
 5   mainroad          545 non-null    object
 6   guestroom         545 non-null    object
 7   basement          545 non-null    object
 8   hotwaterheating   545 non-null    object
 9   airconditioning   545 non-null    object
 10  parking           545 non-null    int64 
 11  prefarea          545 non-null    object
 12  furnishingstatus  545 non-null    object
dtypes: int64(6), object(7)
memory usage: 55.5+ KB
```

Reading from the dataset and observing a summary of the dataset and its values as well as column datatypes.

In [6]:
```
df.nunique().sort_values()
```

Out[6]:
```
mainroad            2
guestroom           2
basement            2
hotwaterheating     2
airconditioning     2
prefarea            2
furnishingstatus    3
bathrooms           4
stories             4
parking             4
bedrooms            6
price             219
area              284
dtype: int64
```

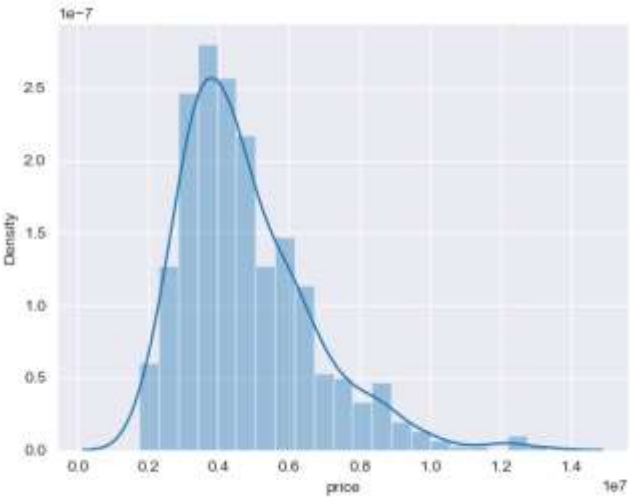
Only `price` and `area` is continous numerical features. `hotwaterheating`, `airconditioning`, `prefarea` and `furnishingstatus` are categorical values, while `bathrooms`, `stories`, `parking` and `bedrooms` are discrete numerical features.

It can be observed that only price and area are continuous numerical features while hotwaterheating, airconditioning, prefarea and furnishingstatus are categorical values and bathrooms, stories, parking and bedrooms are discrete numerical features in the dataset.

In [7]:
```
sns.distplot(df.price)
```

```
C:\Users\munky\miniconda3\envs\FAI\lib\site-packages\seaborn\distributions.py:2619: FutureWarning: `distplot` is a deprecated f
unction and will be removed in a future version. Please adapt your code to use either `displot` (a figure-level function with s
imilar flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```

Out[7]: <AxesSubplot:xlabel='price', ylabel='Density'>

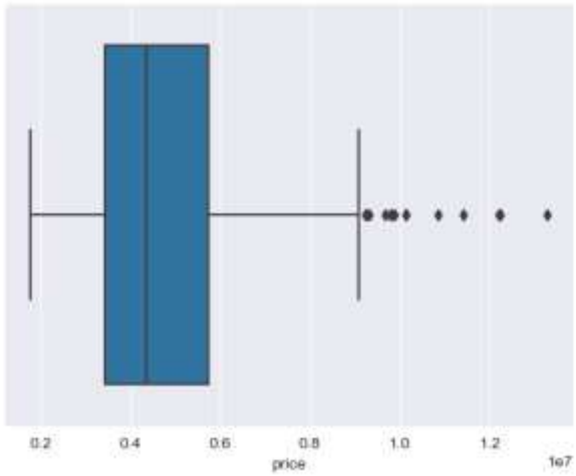


Plotting a distribution plot to visualize the distribution of price in the dataset.

In [8]:
```
sns.boxplot(df.price)
```

```
C:\Users\munky\miniconda3\envs\FAI\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pass the following variable as a keyword arg: x. From version 0.12, the only valid positional argument will be `data`, and passing other arguments without an ex plicit keyword will result in an error or misinterpretation.
  warnings.warn(
```

Out[8]: `<AxesSubplot:xlabel='price'>`



In [9]:
```
q1 = df.price.quantile(.25)
q3 = df.price.quantile(.75)
IQR = q3-q1
price_outlier = df[df.price > q3 + 1.5*IQR]
price_outlier
```

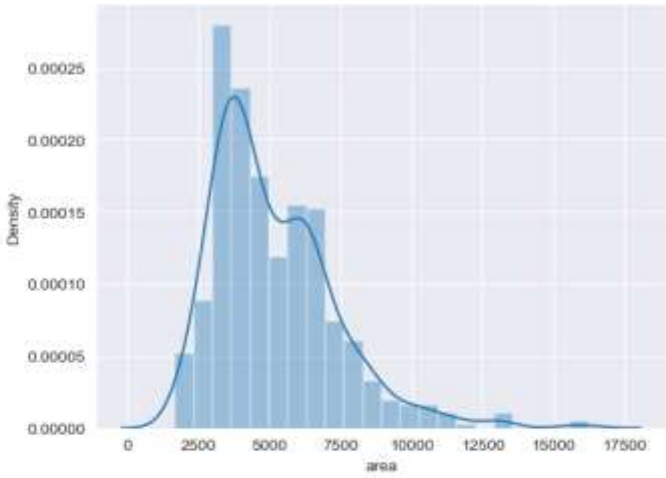
Out[9]:

	price	area	bedrooms	bathrooms	stories	mainroad	guestroom	basement	hotwaterheating	airconditioning	parking	prefarea	furnishingstatus
0	13300000	7420	4	2	3	yes	no	no	no	yes	2	yes	furnished
1	12250000	8960	4	4	4	yes	no	no	no	yes	3	no	furnished
2	12250000	9960	3	2	2	yes	no	yes	no	no	2	yes	semi-furnished
3	12215000	7500	4	2	2	yes	no	yes	no	yes	3	yes	furnished
4	11410000	7420	4	1	2	yes	yes	yes	no	yes	2	no	furnished
5	10850000	7500	3	3	1	yes	no	yes	no	yes	2	yes	semi-furnished
6	10150000	8580	4	3	4	yes	no	no	no	yes	2	yes	semi-furnished
7	10150000	16200	5	3	2	yes	no	no	no	no	0	no	unfurnished
8	9870000	8100	4	1	2	yes	yes	yes	no	yes	2	yes	furnished
9	9800000	5750	3	2	4	yes	yes	no	no	yes	1	yes	unfurnished
10	9800000	13200	3	1	2	yes	no	yes	no	yes	2	yes	furnished
11	9681000	6000	4	3	2	yes	yes	yes	yes	no	2	no	semi-furnished
12	9310000	6550	4	2	2	yes	no	no	no	yes	1	yes	semi-furnished
13	9240000	3500	4	2	2	yes	no	no	yes	no	2	no	furnished
14	9240000	7800	3	2	2	yes	no	no	no	no	0	yes	semi-furnished

Observation: As we can see from the histogram, the data for the prices is slightly skewed to the left. This means there are more houses that are around $ 400,000 and there are 15 houses that are on the extreme high end which are the outliers shown on the box plot.

Plotting of the price in a boxplot for the observation of outliers with the histogram.

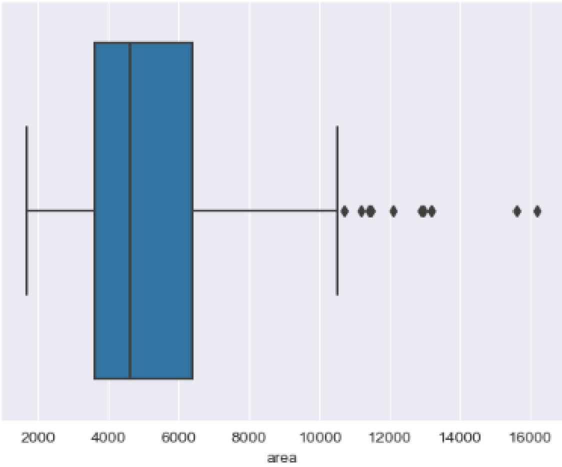
```
In [10]: sns.distplot(df.area)

C:\Users\munky\miniconda3\envs\FAI\lib\site-packages\seaborn\distributions.py:2619: FutureWarning: `distplot` is a deprecated f
unction and will be removed in a future version. Please adapt your code to use either `displot` (a figure-level function with s
imilar flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)

Out[10]: <AxesSubplot:xlabel='area', ylabel='Density'>
```



Plotting a distribution plot to visualize the distribution of area in the dataset.

In [11]:
```python
sns.boxplot(x = df.area)
```

Out[11]: <AxesSubplot:xlabel='area'>



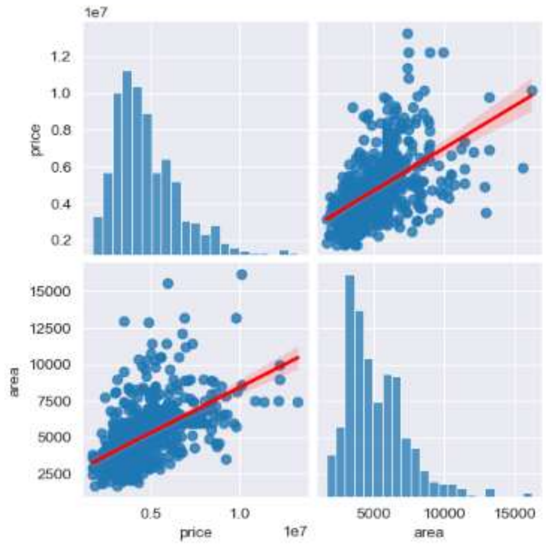
In [12]:
```python
q1 = df.area.quantile(.25)
q3 = df.area.quantile(.75)
IQR = q3-q1
area_outlier = df[df.area > q3 + 1.5*IQR]
area_outlier
```

Out[12]:

	price	area	bedrooms	bathrooms	stories	mainroad	guestroom	basement	hotwaterheating	airconditioning	parking	prefarea	furnishingstatus
7	10150000	16200	5	3	2	yes	no	no	no	no	0	no	unfurnished
10	9800000	13200	3	1	2	yes	no	yes	no	yes	2	yes	furnished
56	7343000	11440	4	1	2	yes	no	yes	no	no	1	yes	semi-furnished
64	7000000	11175	3	1	1	yes	no	yes	no	yes	1	yes	furnished
66	6930000	13200	2	1	1	yes	no	yes	yes	no	1	no	furnished
69	6790000	12090	4	2	2	yes	no	no	no	no	2	yes	furnished
125	5943000	15600	3	1	1	yes	no	no	no	yes	2	no	semi-furnished
129	5873000	11460	3	1	3	yes	no	no	no	no	2	yes	semi-furnished
186	5110000	11410	2	1	2	yes	no	no	no	no	0	yes	furnished
191	5040000	10700	3	1	2	yes	yes	yes	no	no	0	no	semi-furnished
211	4900000	12900	3	1	1	yes	no	no	no	no	2	no	furnished
403	3500000	12944	3	1	1	yes	no	no	no	no	0	no	unfurnished

Observation: Just like the price, the area of the house in this dataset is skewed slightly to the left. There are 12 outliers

In [13]:
```
sns.pairplot(df[["price", "area"]], kind='reg', plot_kws={'line_kws':{'color':'red'}})
```
Out[13]: <seaborn.axisgrid.PairGrid at 0x1fa714f45e0>



In [14]:
```
df[["price", "area"]].corr()
```
Out[14]:

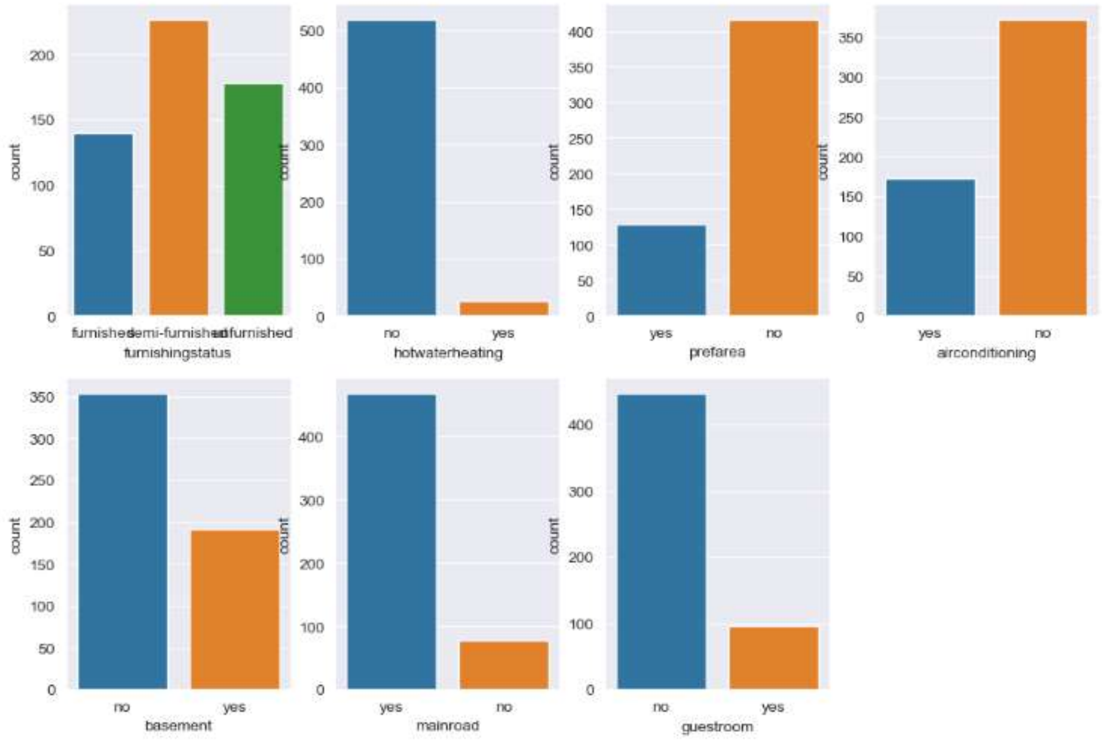
	price	area
price	1.000000	0.535997
area	0.535997	1.000000

Observation: It is shown that price and area has a fairly strong positive correlation.

Type *Markdown* and LaTeX: α^2

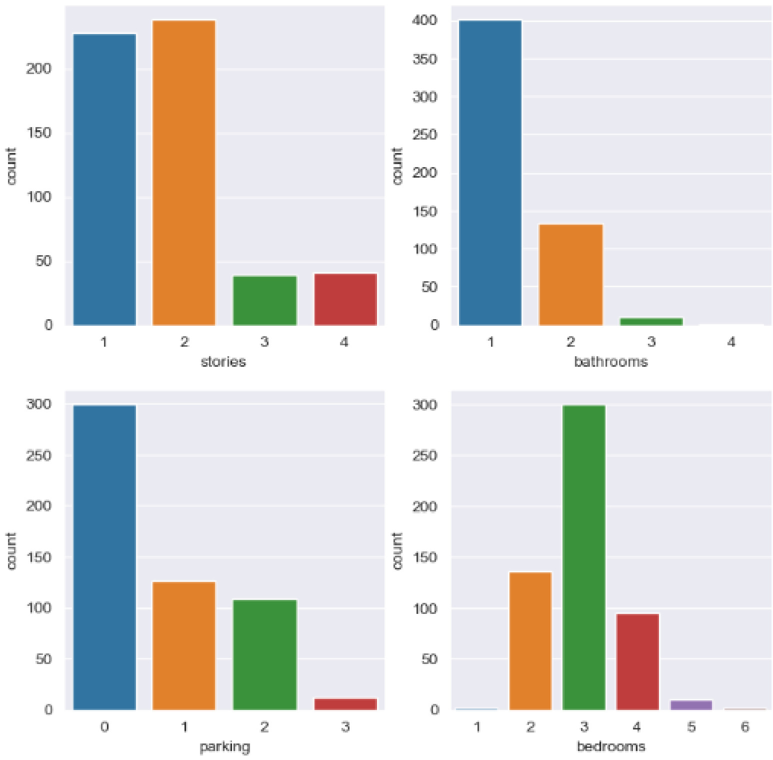
In [21]:
```
plt.figure(figsize=[12,8])
plt.subplot(2,4,1)
sns.countplot(x = df.furnishingstatus)
plt.subplot(242)
sns.countplot(x = df.hotwaterheating)
plt.subplot(243)
sns.countplot(x = df.prefarea)
plt.subplot(244)
sns.countplot(x = df.airconditioning)
plt.subplot(245)
sns.countplot(x=df.basement)
plt.subplot(246)
sns.countplot(x=df.mainroad)
plt.subplot(247)
sns.countplot(x=df.guestroom)
```

Out[21]: `<AxesSubplot:xlabel='guestroom', ylabel='count'>`



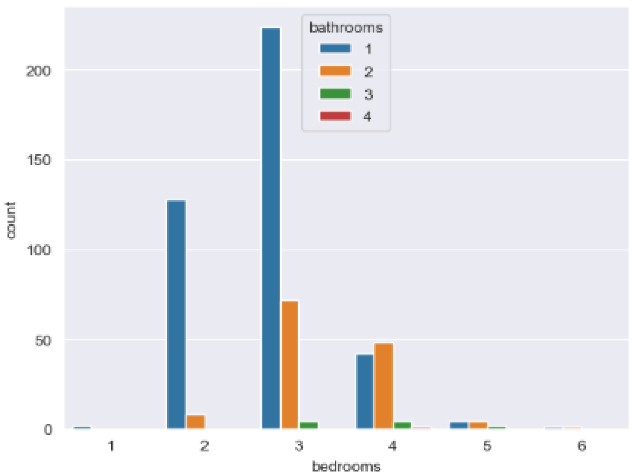
In [40]:
```
plt.figure(figsize=[8,8])
# stories bathrooms parkng bedrooms
plt.subplot(2,2,1)
sns.countplot(x = df.stories)
plt.subplot(222)
sns.countplot(x = df.bathrooms)
plt.subplot(223)
sns.countplot(x = df.parking)
plt.subplot(224)
sns.countplot(x = df.bedrooms)
```

Out[40]: <AxesSubplot:xlabel='bedrooms', ylabel='count'>



In [44]:
```
sns.countplot(data = df, x=df.bedrooms, hue='bathrooms')
```

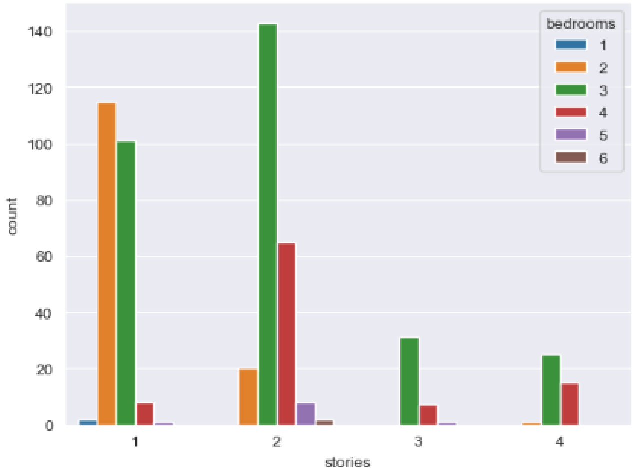
Out[44]: <AxesSubplot:xlabel='bedrooms', ylabel='count'>



Observation: Most houses with 2 bathrooms has at least 2 bedrooms. While most houses still have only 1 bathroom despite having more than 1 bedrooms.

In [45]:
```
sns.countplot(data=df, x='stories', hue='bedrooms')
```

Out[45]: `<AxesSubplot:xlabel='stories', ylabel='count'>`



Observation: For houses that has more than 1 story, there are at least 2 bedroom. However, there is also a house with only 1 story with 4 or more bedrooms. Mostly, single story house have 2 or 3 bedrooms, while 2-story house will hae 3 or 4 bedrooms. For 3 and 4 story houses, the most common is 3 bedrooms.

Conclusion

The developers had carried out detailed exploration for the dataset and several insights has been identified. However, the chosen attribute for the model development will be determined in the implementation part of the system in the second part of this assignment.

2.0 Background Study

2.1 Methods

Regression is a type of supervised learning algorithm that is being utilized for prediction through learning and building relationships between current statistical data and the target value. For instance, it can be used to estimate the house selling price. There are many types of Regression algorithms available currently and one of them is called Linear Regression which is being used to determine the optimal fit straight line and the values of intercept and figure that had the least error in which error represented the difference between the actual value and estimated value. For instance, the difference between actual house price and predicted house price will be a sample error. There are 2 types of categories for Linear Regression namely Simple Linear Regression and Multiple Linear Regression. The Simple Linear Regression consists of just 1 independent variable and the model will need to determine the straight relationship between the independent variable and the dependent variable while Multiple Linear Regression consists of more than 1 independent variable for the model to determine relationship with dependent variable. The equation of Simple Linear Regression is y = b0 + b1x where x is an independent variable, y is a dependent variable, b0 is the intercept and b1 is the figure. Meanwhile, for the Multiple Linear Regression, the equation would be y = b0 + b1x1 + b2x2 + b3x3… + bnxn where x1,x2,x3,...,xn are the independent variables, y is the dependent variable, b0 is the intercept and b1,b2,b3,...,bn are the figures (Pathak & Chaudhari, 2021). Therefore, linear regression is useful for determining and predicting future outcomes based on the relationship between 2 variables.

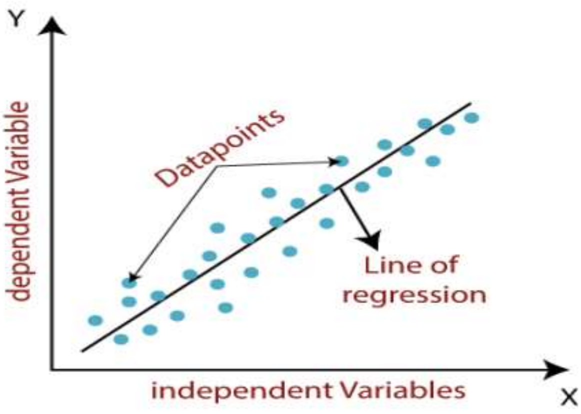


Figure 3: Linear Regression (Pathak & Chaudhari, 2021)

Moving on, another Regression algorithm that exists is known as Polynomial Regression which is a special form of Simple Linear Regression. Polynomial Regression is not able to fit a linear regression line between independent and dependent variables as there is no relationship between the variables. Instead, the Polynomial Regression model is utilizing a curve that is being suited against 2 variables as it can be achieved through fitting a polynomial equation of degree n on the non-linear data that develop a full-figured relationship between dependent and independent variables. The independent variables in Polynomial Regression are not independent of each other and the equation will be Y = a + b1x1 + b2x2 + b3x3+...+bnxn. One of the benefits of this type of regression is that it can provide the best prediction of the relationship between independent and dependent variables and a huge range of curves can be fit into the Polynomial Regression model through diversifying the degree of the model. However, this type of regression also had its own disadvantages such as being too sensitive towards the outliers in the dataset as the outliers increase variance and some unnoticed data point might cause it to perform badly (Sanyal et.al., 2022). So, Polynomial Regression will have a great performance when working on non-linear problems as it can work on any size of dataset.

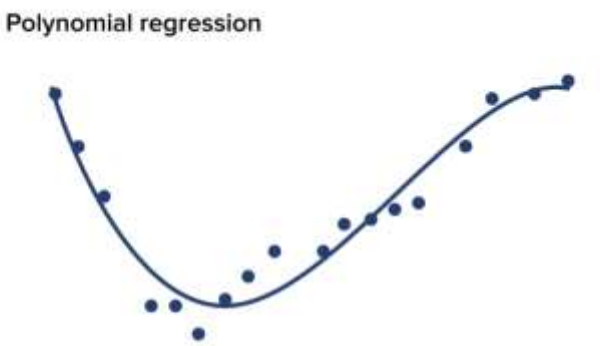


Figure 4: Polynomial Regression (Sanyal et.al., 2022)

Furthermore, the third type of Regression algorithm is known as Ridge Regression which is a way of predicting the coefficients of multiple regression models when the independent variables correspond very correspond with each other in a data set that suffer multicollinearity. For instance, the house price is highly correlated with the land area of the house. The approximation of the coefficient of the model is done to determine whether the independent variables are highly correlated and this model also consists of bias which is known as Ridge Regression penalty to achieve better prediction. The complexity of the model is decreased through a regularization technique that is better known as L2 Regularization which adds magnitude of the coefficient as penalty to the loss function. The bias is calculated as λ* in which λ is the parameter tuning and the values of alpha will be controlling the penalty term. If the value of alpha is higher, the penalty will be bigger and the size of coefficients will be reduced. By getting the coefficients to decrease, the parameters will be shrinked and it can prevent multicollinearity from happening and the model complexity will also be reduced. The equation of Ridge Regression is Lossridge = Σ = (yi - y*)^2 + λb^2 (Sanyal et.al., 2022). Therefore, Ridge Regression is suitable to be used when the data set consists of a high number of predictor variables compared to the number of observations and when multicollinearity appears.

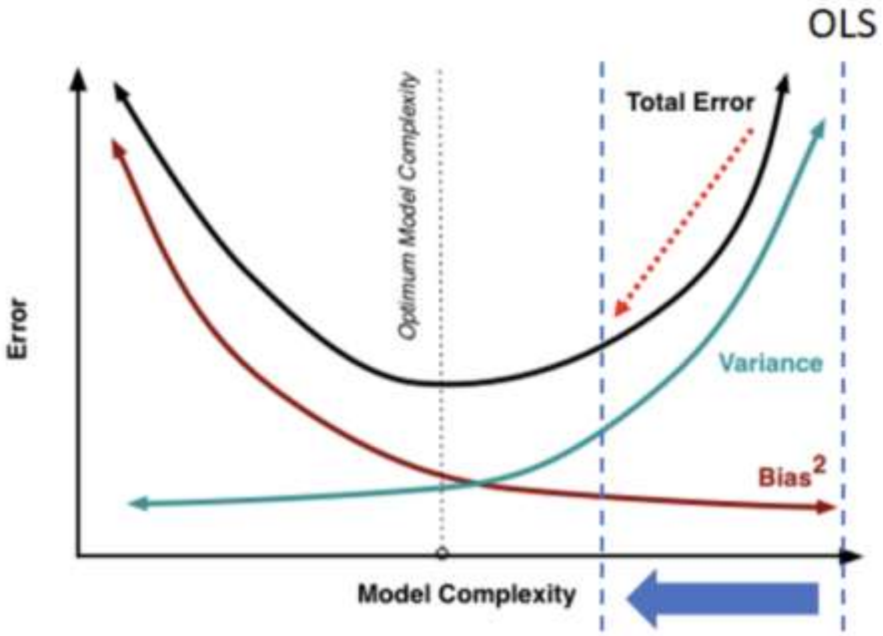


Figure 5: Ridge Regression (Sanyal et.al., 2022)

2.2 Chosen Method

After evaluating all Regression models, one of the chosen methods will be Multiple Linear Regression. The reason why Multiple Linear Regression is chosen is because it has the ability to find out the influence of predictor variables to the value of criterion. For instance, through this method, the size of the house and number of rooms which have a strong relationship with the price of home will be determined. Besides, it can also determine outliers or data anomalies. For instance, it can determine whether a house price is overpriced compared to others on the same location through identifying the correlation between house price and location (Weedmark, 2018). Furthermore, Multiple Linear Regression is suitable because of the multiple variables available on the dataset that is going to be used for training. Therefore, Multiple Linear Regression is one of the most suitable methods for the system that needs to make predictions for housing prices.

Moving on, the next method that will be chosen to evaluate against the Multiple Linear Regression is Polynomial Regression. One of the reasons why it is being chosen is because it can obtain a minimum error or the least cost function. For instance, if the variables in the data set were correlated but the relationship does not look linear enough, then Polynomial Regression can make the result more accurate by fitting a polynomial line on the graph. Besides, it can fit a wide range of functions and a huge range of curvature which makes it suitable for a data set that comes with a lot of attributes (Pant, 2019). Therefore, Polynomial Regression is a suitable method to evaluate against Multiple Linear Regression because of its ability to improve the accuracy and errors on Multiple Linear Regression which is crucial considering the housing dataset is fluctuating.

3.0 Design

3.1 Core features

For this assignment, the developers had decided to build a system that helps users in estimating house prices based on the given criteria.

In the proposed application, the developer will first train 2 models, which are Linear Regression and Polynomial Regression. The 2 models are both regression models which had been explained in the section above. Both models will be trained with the same training data that had been cleaned and preprocessed. Both trained models will be stored for further use.

Users are able to choose the model that they prefer for the prediction. The main part of the proposed application is to allow users to predict and estimate the house price based on the prompted criteria. The proposed application will prompt the user for information such as the area of the house, the number of bedrooms, number of bathrooms, presence of basement and presence of guestrooms. The application will then use the model chosen previously to predict the house price. This allows the user to get an estimated value of the house before they commit to purchasing the house.

3.2 Architectural Design

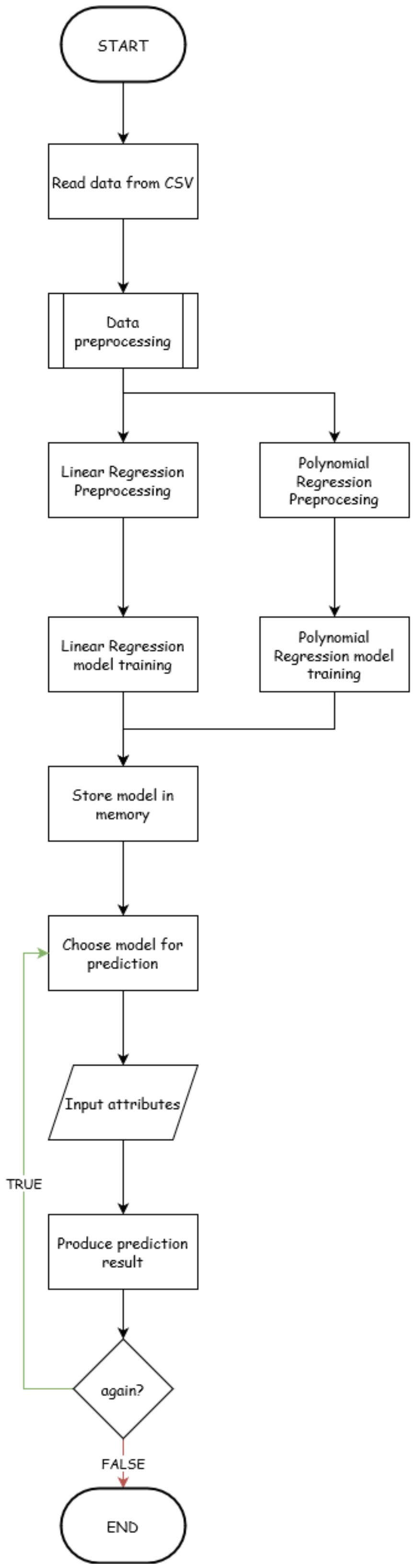


Figure 6: Architectural Design

References

Ali, R. (23 September, 2020). *Oracle NetSuite*. Retrieved from Predictive Modeling: Types, Benefits, and Algorithms: https://www.netsuite.com/portal/resource/articles/financial-management/predictive-modeling.shtml

Asiri, S. (15th November, 2022). *Builtin*. Retrieved from An Introduction to Classification in Machine Learning: https://builtin.com/machine-learning/classification-machine-learning

DATA SCIENCE. (3rd February, 2020). *Explorium*. Retrieved from Clustering — When You Should Use it and Avoid It: https://www.explorium.ai/blog/clustering-when-you-should-use-it-and-avoid-it/

Harper, J. (18th November, 2022). *DW*. Retrieved from Will the global housing bubble burst?: https://www.dw.com/en/will-the-global-housing-bubble-burst/a-63791297

IBM Cloud Education. (19th August, 2020). *IBM*. Retrieved from Supervised Learning: https://www.ibm.com/cloud/learn/supervised-learning#:~:text=Supervised%20learning%2C%20also%20known%20as,data%20or%20predict%20outcomes%20accurately.

M Yasser, H. (n.d.). *Housing Prices Dataset.* Retrieved from Kaggle: https://www.kaggle.com/datasets/yasserh/housing-prices-dataset?resource=download

NetApp. (n.d.). *NetApp*. Retrieved from What is artificial intelligence?: https://www.netapp.com/artificial-intelligence/what-is-artificial-intelligence/

Pant, A. (2019). Introduction to Linear Regression and Polynomial Regression. https://towardsdatascience.com/introduction-to-linear-regression-and-polynomial-regression-f8adc96f31cb.

Pathak, S., Chaudhari, A. (2021). Comparison of Machine Learning Algorithms for House Price Prediction using Real Time Data, International journal of engineering research & technology, 10(12). (pp. 2278-0181). IJERT.

Sanyal, S., Biswas, S. K., Das, D., Chakraborty, M., & Purkayastha, B. (2022, June). Boston House Price Prediction Using Regression Models. In *2022 2nd International Conference on Intelligent Technologies (CONIT)* (pp. 1-6). IEEE.

Weedmark, D. (2018). The Advantages & Disadvantages of a Multiple Regression Model. https://sciencing.com/advantages-disadvantages-multiple-regression-model-12070171.html.

Workload Matrix

Workload	Responsible Students
Introduction • Problem Statement • Chosen Dataset	Sia De Long
Introduction • Data Preprocessing and Exploration	Hor Shen Hau & Yan Mun Kye
Background Study • Methods • Chosen Method	Tan Sheng Jeh
Design • Core Features • Architectural Design	Yan Mun Kye