Data Exploration using Pandas

Melbourne Housing Dataset

In [1]:

```
import pandas as pd
```

In [2]:

```
estate=pd.read_csv("C:/Users/akari/OneDrive/Desktop/estate.csv")
```

Using .read.csv() helps to import the csv file into the notebook

In [3]:

```
estate
```

Out[3]:

	Suburb	Address	Rooms	Type	Price	Method	SellerG	Date	Distance	Postcode	...	Bathroom	Car	Landsize	BuildingArea
0	Abbotsford	85 Turner St	2	h	1480000	S	Biggin	03-12-2016	2.5	3067	...	1	1.0	202	Nal
1	Abbotsford	25 Bloomburg St	2	h	1035000	S	Biggin	04-02-2016	2.5	3067	...	1	0.0	156	79.(
2	Abbotsford	5 Charles St	3	h	1465000	SP	Biggin	04-03-2017	2.5	3067	...	2	0.0	134	150.(
3	Abbotsford	40 Federation La	3	h	850000	PI	Biggin	04-03-2017	2.5	3067	...	2	1.0	94	Nal
4	Abbotsford	55a Park St	4	h	1600000	VB	Nelson	04-06-2016	2.5	3067	...	1	2.0	120	142.(
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	.
13575	Wheelers Hill	12 Strada Cr	4	h	1245000	S	Barry	26-08-2017	16.7	3150	...	2	2.0	652	Nal
13576	Williamstown	77 Merrett Dr	3	h	1031000	SP	Williams	26-08-2017	6.8	3016	...	2	2.0	333	133.(
13577	Williamstown	83 Power St	3	h	1170000	S	Raine	26-08-2017	6.8	3016	...	2	4.0	436	Nal
13578	Williamstown	96 Verdon St	4	h	2500000	PI	Sweeney	26-08-2017	6.8	3016	...	1	5.0	866	157.(
13579	Yarraville	6 Agnes St	4	h	1285000	SP	Village	26-08-2017	6.3	3013	...	1	1.0	362	112.(

13580 rows × 21 columns

In [4]:

```
estate.head(10)
```

Out[4]:

	Suburb	Address	Rooms	Type	Price	Method	SellerG	Date	Distance	Postcode	...	Bathroom	Car	Landsize	BuildingArea	Yearl
0	Abbotsford	85 Turner St	2	h	1480000	S	Biggin	03-12-2016	2.5	3067	...	1	1.0	202	NaN	
1	Abbotsford	25 Bloomburg St	2	h	1035000	S	Biggin	04-02-2016	2.5	3067	...	1	0.0	156	79.0	19
2	Abbotsford	5 Charles St	3	h	1465000	SP	Biggin	04-03-2017	2.5	3067	...	2	0.0	134	150.0	19
3	Abbotsford	40 Federation La	3	h	850000	PI	Biggin	04-03-2017	2.5	3067	...	2	1.0	94	NaN	
4	Abbotsford	55a Park St	4	h	1600000	VB	Nelson	04-06-2016	2.5	3067	...	1	2.0	120	142.0	20
5	Abbotsford	129 Charles St	2	h	941000	S	Jellis	07-05-2016	2.5	3067	...	1	0.0	181	NaN	
6	Abbotsford	124 Yarra St	3	h	1876000	S	Nelson	07-05-2016	2.5	3067	...	2	0.0	245	210.0	19
7	Abbotsford	98 Charles St	2	h	1636000	S	Nelson	08-10-2016	2.5	3067	...	1	2.0	256	107.0	18
8	Abbotsford	6/241 Nicholson St	1	u	300000	S	Biggin	08-10-2016	2.5	3067	...	1	1.0	0	NaN	
9	Abbotsford	10 Valiant St	2	h	1097000	S	Biggin	08-10-2016	2.5	3067	...	1	2.0	220	75.0	19

10 rows × 21 columns

Here are the first 10 observations which helps us to know how the dataset is framed

In [5]:

```
estate.tail(5)
```

Out[5]:

	Suburb	Address	Rooms	Type	Price	Method	SellerG	Date	Distance	Postcode	...	Bathroom	Car	Landsize	BuildingArea
13575	Wheelers Hill	12 Strada Cr	4	h	1245000	S	Barry	26-08-2017	16.7	3150	...	2	2.0	652	NaN
13576	Williamstown	77 Merrett Dr	3	h	1031000	SP	Williams	26-08-2017	6.8	3016	...	2	2.0	333	133.0
13577	Williamstown	83 Power St	3	h	1170000	S	Raine	26-08-2017	6.8	3016	...	2	4.0	436	NaN
13578	Williamstown	96 Verdon St	4	h	2500000	PI	Sweeney	26-08-2017	6.8	3016	...	1	5.0	866	157.0
13579	Yarraville	6 Agnes St	4	h	1285000	SP	Village	26-08-2017	6.3	3013	...	1	1.0	362	112.0

5 rows × 21 columns

Here we can see last 5 observations to know about the dataset

In [6]:

```
estate.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 13580 entries, 0 to 13579
Data columns (total 21 columns):
 #   Column         Non-Null Count  Dtype  
---  ------         --------------  -----  
 0   Suburb         13580 non-null  object 
 1   Address        13580 non-null  object 
 2   Rooms          13580 non-null  int64  
 3   Type           13580 non-null  object 
 4   Price          13580 non-null  int64  
 5   Method         13580 non-null  object 
 6   SellerG        13580 non-null  object 
 7   Date           13580 non-null  object 
 8   Distance       13580 non-null  float64
 9   Postcode       13580 non-null  int64  
 10  Bedroom2       13580 non-null  int64  
 11  Bathroom       13580 non-null  int64  
 12  Car            13518 non-null  float64
 13  Landsize       13580 non-null  int64  
 14  BuildingArea   7130 non-null   float64
 15  YearBuilt      8205 non-null   float64
 16  CouncilArea    12211 non-null  object 
 17  Lattitude      13580 non-null  float64
 18  Longtitude     13580 non-null  float64
 19  Regionname     13580 non-null  object 
 20  Propertycount  13580 non-null  int64  
dtypes: float64(6), int64(7), object(8)
memory usage: 2.2+ MB
```

Info gives us the count of the observation under each variable and their datatypes. As we can see that 7 variables are in integer type, 8 in object and 6 in float.

In [7]:

```
estate.describe()
```

Out[7]:

	Rooms	Price	Distance	Postcode	Bedroom2	Bathroom	Car	Landsize	BuildingArea	Yea
count	13580.000000	1.358000e+04	13580.000000	13580.000000	13580.000000	13580.000000	13518.000000	13580.000000	7130.000000	8205.0
mean	2.937997	1.075684e+06	10.137776	3105.301915	2.914728	1.534242	1.610075	558.416127	151.967650	1964.6
std	0.955748	6.393107e+05	5.868725	90.676964	0.965921	0.691712	0.962634	3990.669241	541.014538	37.2
min	1.000000	8.500000e+04	0.000000	3000.000000	0.000000	0.000000	0.000000	0.000000	0.000000	1196.0
25%	2.000000	6.500000e+05	6.100000	3044.000000	2.000000	1.000000	1.000000	177.000000	93.000000	1940.0
50%	3.000000	9.030000e+05	9.200000	3084.000000	3.000000	1.000000	2.000000	440.000000	126.000000	1970.0
75%	3.000000	1.330000e+06	13.000000	3148.000000	3.000000	2.000000	2.000000	651.000000	174.000000	1999.0
max	10.000000	9.000000e+06	48.100000	3977.000000	20.000000	8.000000	10.000000	433014.000000	44515.000000	2018.0

Describe function as its name describes the dataset such as the count of observation under each variable, their mean, Minimum value, Maximum value, and quaters. Looking at the dataset we can find that the houses are there with no bedrooms to maximum 20 bedrooms. There are houses with no bathrooms and maximum 8 bathrooms. Likewise it goes to every variable.

In [8]:

```
estate.count()
```

Out[8]:

```
Suburb          13580
Address         13580
Rooms           13580
Type            13580
Price           13580
Method          13580
SellerG         13580
Date            13580
Distance        13580
Postcode        13580
Bedroom2        13580
Bathroom        13580
Car             13518
Landsize        13580
BuildingArea     7130
YearBuilt        8205
CouncilArea     12211
Lattitude       13580
Longtitude      13580
Regionname      13580
Propertycount   13580
dtype: int64
```

The count function gives us the count of observations under each variable. When we look at the output there are missing data in the variable Building Area and yearbuilt. So these data wont be that useful for decision making

In [9]:

```
estate.shape
```

Out[9]:

```
(13580, 21)
```

The shape function gives us the count of variables and observations. Here we have 13580 observations and 21 variables.

In [10]:

```
estate.columns
```

Out[10]:

```
Index(['Suburb', 'Address', 'Rooms', 'Type', 'Price', 'Method', 'SellerG',
       'Date', 'Distance', 'Postcode', 'Bedroom2', 'Bathroom', 'Car',
       'Landsize', 'BuildingArea', 'YearBuilt', 'CouncilArea', 'Lattitude',
       'Longtitude', 'Regionname', 'Propertycount'],
      dtype='object')
```

The columns function shows the names of all the variables of the datatset

In [11]:

```
estate.index
```

Out[11]:

```
RangeIndex(start=0, stop=13580, step=1)
```

The index function describes the start of index and where it ends and the number of steps between index. That is the first observation always starts with 0 and the difference between the index is 1

In [12]:

```
estate.sort_values(['Price'],ascending=False)
```

Out[12]:

	Suburb	Address	Rooms	Type	Price	Method	SellerG	Date	Distance	Postcode	...	Bathroom	Car	Landsize	BuildingAr
12094	Mulgrave	35 Bevis St	3	h	9000000	PI	Hall	29-07-2017	18.8	3170	...	1	1.0	744	117
7692	Canterbury	49 Mangarra Rd	5	h	8000000	VB	Sotheby's	13-05-2017	9.0	3126	...	5	4.0	2079	464
9575	Hawthorn	49 Lisson Gr	4	h	7650000	S	Abercromby's	17-06-2017	5.3	3122	...	2	4.0	1690	284
3616	Kew	15 Barry St	6	h	6500000	S	Jellis	13-08-2016	5.6	3101	...	6	3.0	1334	365
12557	Middle Park	136 Page St	5	h	6400000	S	Marshall	09-09-2017	3.0	3206	...	2	1.0	553	308
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
7940	Hawthorn	17/17 Park St	1	u	160000	VB	HAR	08-04-2017	4.6	3122	...	1	0.0	322	Na
7303	Albion	8/6 Ridley St	1	u	145000	PI	Biggin	28-05-2016	13.9	3020	...	1	1.0	36	Na
1927	Coburg	171 Moreland Rd	4	h	145000	PI	Jellis	04-06-2016	7.8	3058	...	1	1.0	536	164
1805	Caulfield	30 Pyne St	4	h	131000	PI	Rodney	25-02-2017	8.9	3162	...	1	2.0	499	155
2652	Footscray	202/51 Gordon St	1	u	85000	PI	Burnham	03-09-2016	6.4	3011	...	1	0.0	0	Na

13580 rows × 21 columns

Sort_values() helps to sort the values either in ascending or descending order. Here the dataset is sorted based on the price from the highest to the lowest.

In [13]:

```
estate['Rooms'].value_counts()
```

Out[13]:

```
3     5881
2     3648
4     2688
1      681
5      596
6       67
7       10
8        8
10       1
Name: Rooms, dtype: int64
```

Value_counts() helps us to get the data on how many times the values has been repeated. Here we can find how many houses in Melbourne have the certain number of rooms.

In [14]:

```
estate.drop_duplicates()
```

Out[14]:

	Suburb	Address	Rooms	Type	Price	Method	SellerG	Date	Distance	Postcode	...	Bathroom	Car	Landsize	BuildingArea
0	Abbotsford	85 Turner St	2	h	1480000	S	Biggin	03-12-2016	2.5	3067	...	1	1.0	202	Nal
1	Abbotsford	25 Bloomburg St	2	h	1035000	S	Biggin	04-02-2016	2.5	3067	...	1	0.0	156	79.(
2	Abbotsford	5 Charles St	3	h	1465000	SP	Biggin	04-03-2017	2.5	3067	...	2	0.0	134	150.(
3	Abbotsford	40 Federation La	3	h	850000	PI	Biggin	04-03-2017	2.5	3067	...	2	1.0	94	Nal
4	Abbotsford	55a Park St	4	h	1600000	VB	Nelson	04-06-2016	2.5	3067	...	1	2.0	120	142.(
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	.
13575	Wheelers Hill	12 Strada Cr	4	h	1245000	S	Barry	26-08-2017	16.7	3150	...	2	2.0	652	Nal
13576	Williamstown	77 Merrett Dr	3	h	1031000	SP	Williams	26-08-2017	6.8	3016	...	2	2.0	333	133.(
13577	Williamstown	83 Power St	3	h	1170000	S	Raine	26-08-2017	6.8	3016	...	2	4.0	436	Nal
13578	Williamstown	96 Verdon St	4	h	2500000	PI	Sweeney	26-08-2017	6.8	3016	...	1	5.0	866	157.(
13579	Yarraville	6 Agnes St	4	h	1285000	SP	Village	26-08-2017	6.3	3013	...	1	1.0	362	112.(

13580 rows × 21 columns

In [15]:

```
estate
```

Out[15]:

	Suburb	Address	Rooms	Type	Price	Method	SellerG	Date	Distance	Postcode	...	Bathroom	Car	Landsize	BuildingArea
0	Abbotsford	85 Turner St	2	h	1480000	S	Biggin	03-12-2016	2.5	3067	...	1	1.0	202	Nal
1	Abbotsford	25 Bloomburg St	2	h	1035000	S	Biggin	04-02-2016	2.5	3067	...	1	0.0	156	79.(
2	Abbotsford	5 Charles St	3	h	1465000	SP	Biggin	04-03-2017	2.5	3067	...	2	0.0	134	150.(
3	Abbotsford	40 Federation La	3	h	850000	PI	Biggin	04-03-2017	2.5	3067	...	2	1.0	94	Nal
4	Abbotsford	55a Park St	4	h	1600000	VB	Nelson	04-06-2016	2.5	3067	...	1	2.0	120	142.(
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	.
13575	Wheelers Hill	12 Strada Cr	4	h	1245000	S	Barry	26-08-2017	16.7	3150	...	2	2.0	652	Nal
13576	Williamstown	77 Merrett Dr	3	h	1031000	SP	Williams	26-08-2017	6.8	3016	...	2	2.0	333	133.(
13577	Williamstown	83 Power St	3	h	1170000	S	Raine	26-08-2017	6.8	3016	...	2	4.0	436	Nal
13578	Williamstown	96 Verdon St	4	h	2500000	PI	Sweeney	26-08-2017	6.8	3016	...	1	5.0	866	157.(
13579	Yarraville	6 Agnes St	4	h	1285000	SP	Village	26-08-2017	6.3	3013	...	1	1.0	362	112.(

13580 rows × 21 columns

Drop_duplicates() helps us to remove the duplicated data so that we can perform any function and derive conclusion.

In [16]:

```
Rate=estate[['Rooms','Price']]
```

In [17]:

```
Rate
```

Out[17]:

	Rooms	Price
0	2	1480000
1	2	1035000
2	3	1465000
3	3	850000
4	4	1600000
...	...	...
13575	4	1245000
13576	3	1031000
13577	3	1170000
13578	4	2500000
13579	4	1285000

13580 rows × 2 columns

This is making subsets of the dataset. Here it is the subset of Rooms and Price which helps us to understand how the price differs based on the number of rooms.

In [18]:

```
size=estate[estate['Landsize']>1000]
```

In [19]:

```
size
```

Out[19]:

	Suburb	Address	Rooms	Type	Price	Method	SellerG	Date	Distance	Postcode	...	Bathroom	Car	Landsize	BuildingArea
22	Abbotsford	138/56 Nicholson St	3	u	1090000	S	Jellis	18-03-2017	2.5	3067	...	2	2.0	4290	27.0
51	Airport West	3 Deidre Ct	3	h	895000	PI	Rendina	10-09-2016	13.5	3042	...	1	6.0	1063	133.0
146	Altona	29 Rose St	4	h	1780000	SP	Greg	04-06-2016	13.8	3018	...	3	6.0	1057	220.0
234	Armadale	526 Orrong Rd	4	h	3000000	VB	Jellis	04-06-2016	6.3	3143	...	2	2.0	1581	NaN
236	Armadale	9/19 Mercer Rd	3	u	735000	PI	Marshall	07-05-2016	6.3	3143	...	2	1.0	2113	14.0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
13494	Monbulk	21 David St	4	h	720000	SP	Fletchers	26-08-2017	34.1	3793	...	2	2.0	1611	NaN
13495	Moonee Ponds	1/53 Buckley St	2	u	435000	S	Nelson	26-08-2017	6.2	3039	...	1	1.0	1475	66.0
13527	Reservoir	1 Don St	4	h	1112000	S	RW	26-08-2017	12.0	3073	...	2	10.0	1002	170.0
13547	Sunbury	37 Ligar St	4	h	763000	S	Brad	26-08-2017	31.7	3429	...	2	2.0	1011	NaN
13553	Surrey Hills	20 Albert Cr	4	h	2720000	S	Kay	26-08-2017	10.2	3127	...	3	2.0	1005	NaN

666 rows × 21 columns

This filters the data of your needs. Here the dataset is filtered based on the landsize more than 1000

In [20]:

```
estate['Rest_land']=estate['Landsize']-estate['BuildingArea']
```

In [21]:

```
estate
```

Out[21]:

	Suburb	Address	Rooms	Type	Price	Method	SellerG	Date	Distance	Postcode	...	Car	Landsize	BuildingArea	YearBuilt
0	Abbotsford	85 Turner St	2	h	1480000	S	Biggin	03-12-2016	2.5	3067	...	1.0	202	NaN	NaN
1	Abbotsford	25 Bloomburg St	2	h	1035000	S	Biggin	04-02-2016	2.5	3067	...	0.0	156	79.0	1900.0
2	Abbotsford	5 Charles St	3	h	1465000	SP	Biggin	04-03-2017	2.5	3067	...	0.0	134	150.0	1900.0
3	Abbotsford	40 Federation La	3	h	850000	PI	Biggin	04-03-2017	2.5	3067	...	1.0	94	NaN	NaN
4	Abbotsford	55a Park St	4	h	1600000	VB	Nelson	04-06-2016	2.5	3067	...	2.0	120	142.0	2014.0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
13575	Wheelers Hill	12 Strada Cr	4	h	1245000	S	Barry	26-08-2017	16.7	3150	...	2.0	652	NaN	1981.0
13576	Williamstown	77 Merrett Dr	3	h	1031000	SP	Williams	26-08-2017	6.8	3016	...	2.0	333	133.0	1995.0
13577	Williamstown	83 Power St	3	h	1170000	S	Raine	26-08-2017	6.8	3016	...	4.0	436	NaN	1997.0
13578	Williamstown	96 Verdon St	4	h	2500000	PI	Sweeney	26-08-2017	6.8	3016	...	5.0	866	157.0	1920.0
13579	Yarraville	6 Agnes St	4	h	1285000	SP	Village	26-08-2017	6.3	3013	...	1.0	362	112.0	1920.0

13580 rows × 22 columns

This makes us to add new varaiable in the dataset. Here we have added a new varaiable rest_land by taking difference between the landsize and the Buliding area. So that the buyers can use the free space according to their pupose such as gardening etc.,

In [22]:

```
estate_index=estate.set_index("SellerG")
```

In [23]:

```
estate_index
```

Out[23]:

SellerG	Suburb	Address	Rooms	Type	Price	Method	Date	Distance	Postcode	Bedroom2	...	Car	Landsize	BuildingArea	Year
Biggin	Abbotsford	85 Turner St	2	h	1480000	S	03-12-2016	2.5	3067	2	...	1.0	202	NaN	
Biggin	Abbotsford	25 Bloomburg St	2	h	1035000	S	04-02-2016	2.5	3067	2	...	0.0	156	79.0	19
Biggin	Abbotsford	5 Charles St	3	h	1465000	SP	04-03-2017	2.5	3067	3	...	0.0	134	150.0	19
Biggin	Abbotsford	40 Federation La	3	h	850000	PI	04-03-2017	2.5	3067	3	...	1.0	94	NaN	
Nelson	Abbotsford	55a Park St	4	h	1600000	VB	04-06-2016	2.5	3067	3	...	2.0	120	142.0	20
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
Barry	Wheelers Hill	12 Strada Cr	4	h	1245000	S	26-08-2017	16.7	3150	4	...	2.0	652	NaN	19
Williams	Williamstown	77 Merrett Dr	3	h	1031000	SP	26-08-2017	6.8	3016	3	...	2.0	333	133.0	19
Raine	Williamstown	83 Power St	3	h	1170000	S	26-08-2017	6.8	3016	3	...	4.0	436	NaN	19
Sweeney	Williamstown	96 Verdon St	4	h	2500000	PI	26-08-2017	6.8	3016	4	...	5.0	866	157.0	19
Village	Yarraville	6 Agnes St	4	h	1285000	SP	26-08-2017	6.3	3013	4	...	1.0	362	112.0	19

13580 rows × 21 columns

Set_index helps us to set the variable as the index. That is According to the varaiable the data will be lookedafter

In [24]:

```
estate_index=estate_index.reset_index()
```

In [25]:

```
estate_index
```

Out[25]:

	SellerG	Suburb	Address	Rooms	Type	Price	Method	Date	Distance	Postcode	...	Car	Landsize	BuildingArea	YearBuilt
0	Biggin	Abbotsford	85 Turner St	2	h	1480000	S	03-12-2016	2.5	3067	...	1.0	202	NaN	NaN
1	Biggin	Abbotsford	25 Bloomburg St	2	h	1035000	S	04-02-2016	2.5	3067	...	0.0	156	79.0	1900.0
2	Biggin	Abbotsford	5 Charles St	3	h	1465000	SP	04-03-2017	2.5	3067	...	0.0	134	150.0	1900.0
3	Biggin	Abbotsford	40 Federation La	3	h	850000	PI	04-03-2017	2.5	3067	...	1.0	94	NaN	NaN
4	Nelson	Abbotsford	55a Park St	4	h	1600000	VB	04-06-2016	2.5	3067	...	2.0	120	142.0	2014.0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
13575	Barry	Wheelers Hill	12 Strada Cr	4	h	1245000	S	26-08-2017	16.7	3150	...	2.0	652	NaN	1981.0
13576	Williams	Williamstown	77 Merrett Dr	3	h	1031000	SP	26-08-2017	6.8	3016	...	2.0	333	133.0	1995.0
13577	Raine	Williamstown	83 Power St	3	h	1170000	S	26-08-2017	6.8	3016	...	4.0	436	NaN	1997.0
13578	Sweeney	Williamstown	96 Verdon St	4	h	2500000	PI	26-08-2017	6.8	3016	...	5.0	866	157.0	1920.0
13579	Village	Yarraville	6 Agnes St	4	h	1285000	SP	26-08-2017	6.3	3013	...	1.0	362	112.0	1920.0

13580 rows × 22 columns

This function helps to reset the index to its normal position starting from 0

In [26]:

```
estate_loc=estate.loc[(estate.Suburb=='Yarraville')&(estate.Rooms>=5)]
```

In [27]:

```
estate_loc
```

Out[27]:

	Suburb	Address	Rooms	Type	Price	Method	SellerG	Date	Distance	Postcode	...	Car	Landsize	BuildingArea	YearBuilt
6654	Yarraville	47 Bayview Rd	5	h	1815000	S	Village	08-10-2016	7.0	3013	...	2.0	644	172.0	1890.0
6693	Yarraville	129 Roberts St	5	h	1137500	S	hockingstuart	22-05-2016	7.0	3013	...	4.0	446	244.0	1950.0
6718	Yarraville	10 Tuppen St	5	h	1287000	S	hockingstuart	27-11-2016	7.0	3013	...	2.0	488	213.0	1930.0
12212	Yarraville	54 Pentland Pde	6	h	2450000	VB	Village	29-07-2017	6.3	3013	...	2.0	1087	388.5	1920.0

4 rows × 22 columns

loc is the function which is used to locate a particular set of data. Here we have Located few observation which is the suburban area Yarraville and the houses in that region which has more than 5 rooms.

In [28]:

```
estate_iloc=estate.iloc[0:5,4:6]
```

In [29]:

```
estate_iloc
```

Out[29]:

	Price	Method
0	1480000	S
1	1035000	S
2	1465000	SP
3	850000	PI
4	1600000	VB

iloc is the function which helps us to locate the particular set of observation using index. With the help of this function and index we have located the price and the method variables.

In [30]:

```
estate.agg=estate.groupby("Regionname")['Price'].mean()
```

In [31]:

```
estate.agg
```

Out[31]:

```
Regionname
Eastern Metropolitan          1.104080e+06
Eastern Victoria              6.999808e+05
Northern Metropolitan         8.981711e+05
Northern Victoria             5.948293e+05
South-Eastern Metropolitan    9.229438e+05
Southern Metropolitan         1.372963e+06
Western Metropolitan          8.664205e+05
Western Victoria              3.975234e+05
Name: Price, dtype: float64
```

Groupby is an aggregriate function which groups the data. Here we have grouped the data with regionname and price to know the re;ationship between the variables

In [32]:

```
estate['Rooms']=estate['Rooms'].astype('object')
```

In [33]:

```
estate.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 13580 entries, 0 to 13579
Data columns (total 22 columns):
 #   Column         Non-Null Count  Dtype  
---  ------         --------------  -----  
 0   Suburb         13580 non-null  object 
 1   Address        13580 non-null  object 
 2   Rooms          13580 non-null  object 
 3   Type           13580 non-null  object 
 4   Price          13580 non-null  int64  
 5   Method         13580 non-null  object 
 6   SellerG        13580 non-null  object 
 7   Date           13580 non-null  object 
 8   Distance       13580 non-null  float64
 9   Postcode       13580 non-null  int64  
 10  Bedroom2       13580 non-null  int64  
 11  Bathroom       13580 non-null  int64  
 12  Car            13518 non-null  float64
 13  Landsize       13580 non-null  int64  
 14  BuildingArea   7130 non-null   float64
 15  YearBuilt      8205 non-null   float64
 16  CouncilArea    12211 non-null  object 
 17  Lattitude      13580 non-null  float64
 18  Longtitude     13580 non-null  float64
 19  Regionname     13580 non-null  object 
 20  Propertycount  13580 non-null  int64  
 21  Rest_land      7130 non-null   float64
dtypes: float64(7), int64(6), object(9)
memory usage: 2.3+ MB
```

Astype changes the type of the variable that we want to change either as object or integer or nay format that we want to change. If we look here the variable rooms are in object type so if we want perform any arithmetic opration over it changing it into integer makes the work easy

In [34]:

```
estate['Rooms']=estate['Rooms'].astype('int')
```

In [35]:

```
estate.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 13580 entries, 0 to 13579
Data columns (total 22 columns):
 #   Column         Non-Null Count  Dtype  
---  ------         --------------  -----  
 0   Suburb         13580 non-null  object 
 1   Address        13580 non-null  object 
 2   Rooms          13580 non-null  int32  
 3   Type           13580 non-null  object 
 4   Price          13580 non-null  int64  
 5   Method         13580 non-null  object 
 6   SellerG        13580 non-null  object 
 7   Date           13580 non-null  object 
 8   Distance       13580 non-null  float64
 9   Postcode       13580 non-null  int64  
 10  Bedroom2       13580 non-null  int64  
 11  Bathroom       13580 non-null  int64  
 12  Car            13518 non-null  float64
 13  Landsize       13580 non-null  int64  
 14  BuildingArea   7130 non-null   float64
 15  YearBuilt      8205 non-null   float64
 16  CouncilArea    12211 non-null  object 
 17  Lattitude      13580 non-null  float64
 18  Longtitude     13580 non-null  float64
 19  Regionname     13580 non-null  object 
 20  Propertycount  13580 non-null  int64  
 21  Rest_land      7130 non-null   float64
dtypes: float64(7), int32(1), int64(6), object(8)
memory usage: 2.2+ MB
```