RomainBlanchotMidTerm

March 5, 2025

1 CPE232: Data Models

1.1 ### Portion 2: Midterm Exam Coding

[ ]:
```
# Run this cell if using Google Colab
# from google.colab import drive
# drive.mount('/content/drive')
```

The provided melb_data.csv dataset contains real estate data for properties sold in Melbourne, Australia. Your task is to perform exploratory data analysis (EDA) based on the following five subtasks.

1.1.1 Subtask #1: Getting to know the dataset

1.1) Determine overview info of the data via NumPy (Total points = 3) [8 points] Use NumPy library to display information of the data: size, shape, and number of dimensions.

[3]:
```
import numpy as np
data = np.loadtxt('./melb_data.csv', delimiter=",", dtype=str) #TODO: update␣
↪the filename at this line
```

[6]:
```
# Write your code here
print(data.size) # size
print(data.shape) # qhape
print(data.ndim) # number of dimensions
```

```
285201
(13581, 21)
2
```

1.2) Get more statistics of the data via Pandas (Total points = 10) Use Pandas to display information in the following cells.

[7]:
```
# Load the CSV file
import pandas as pd
df = pd.read_csv('./melb_data.csv') #TODO: update the filename at this line
```

[1 point] The *first 9 records* of all attributes.

[8]:
```
# Write your code here
df.head(9)
```

[8]:

	Suburb	Address	Rooms	Type	Price	Method	SellerG	\
0	Abbotsford	85 Turner St	2	h	1480000.0	S	Biggin	
1	Abbotsford	25 Bloomburg St	2	h	1035000.0	S	Biggin	
2	Abbotsford	5 Charles St	3	h	1465000.0	SP	Biggin	
3	Abbotsford	40 Federation La	3	h	850000.0	PI	Biggin	
4	Abbotsford	55a Park St	4	h	1600000.0	VB	Nelson	
5	Abbotsford	129 Charles St	2	h	941000.0	S	Jellis	
6	Abbotsford	124 Yarra St	3	h	1876000.0	S	Nelson	
7	Abbotsford	98 Charles St	2	h	1636000.0	S	Nelson	
8	Abbotsford	6/241 Nicholson St	1	u	300000.0	S	Biggin	

	Date	Distance	Postcode	…	Bathroom	Car	Landsize	BuildingArea	\
0	3/12/2016	2.5	3067.0	…	1.0	1.0	202.0	NaN	
1	4/02/2016	2.5	3067.0	…	1.0	0.0	156.0	79.0	
2	4/03/2017	2.5	3067.0	…	2.0	0.0	134.0	150.0	
3	4/03/2017	2.5	3067.0	…	2.0	1.0	94.0	NaN	
4	4/06/2016	2.5	3067.0	…	1.0	2.0	120.0	142.0	
5	7/05/2016	2.5	3067.0	…	1.0	0.0	181.0	NaN	
6	7/05/2016	2.5	3067.0	…	2.0	0.0	245.0	210.0	
7	8/10/2016	2.5	3067.0	…	1.0	2.0	256.0	107.0	
8	8/10/2016	2.5	3067.0	…	1.0	1.0	0.0	NaN	

	YearBuilt	CouncilArea	Lattitude	Longtitude	Regionname	\
0	NaN	Yarra	-37.7996	144.9984	Northern Metropolitan	
1	1900.0	Yarra	-37.8079	144.9934	Northern Metropolitan	
2	1900.0	Yarra	-37.8093	144.9944	Northern Metropolitan	
3	NaN	Yarra	-37.7969	144.9969	Northern Metropolitan	
4	2014.0	Yarra	-37.8072	144.9941	Northern Metropolitan	
5	NaN	Yarra	-37.8041	144.9953	Northern Metropolitan	
6	1910.0	Yarra	-37.8024	144.9993	Northern Metropolitan	
7	1890.0	Yarra	-37.8060	144.9954	Northern Metropolitan	
8	NaN	Yarra	-37.8008	144.9973	Northern Metropolitan	

	Propertycount
0	4019.0
1	4019.0
2	4019.0
3	4019.0
4	4019.0
5	4019.0
6	4019.0
7	4019.0
8	4019.0

```
[9 rows x 21 columns]
```

[1 point] The *last 7 records* of all attributes.

[9]:
```
# Write your code here
df.tail(7)
```

[9]:
```
              Suburb        Address  Rooms Type      Price Method  \
13573       Werribee    5 Nuragi Ct      4    h   635000.0      S
13574    Westmeadows     9 Black St      3    h   582000.0      S
13575  Wheelers Hill   12 Strada Cr      4    h  1245000.0      S
13576   Williamstown  77 Merrett Dr      3    h  1031000.0     SP
13577   Williamstown    83 Power St      3    h  1170000.0      S
13578   Williamstown   96 Verdon St      4    h  2500000.0     PI
13579     Yarraville     6 Agnes St      4    h  1285000.0     SP

             SellerG        Date  Distance  Postcode  …  Bathroom  Car  \
13573  hockingstuart  26/08/2017      14.7    3030.0  …       2.0  1.0
13574            Red  26/08/2017      16.5    3049.0  …       2.0  2.0
13575          Barry  26/08/2017      16.7    3150.0  …       2.0  2.0
13576       Williams  26/08/2017       6.8    3016.0  …       2.0  2.0
13577          Raine  26/08/2017       6.8    3016.0  …       2.0  4.0
13578        Sweeney  26/08/2017       6.8    3016.0  …       1.0  5.0
13579        Village  26/08/2017       6.3    3013.0  …       1.0  1.0

       Landsize  BuildingArea  YearBuilt  CouncilArea Lattitude  Longtitude  \
13573     662.0         172.0     1980.0          NaN -37.89327   144.64789
13574     256.0           NaN        NaN          NaN -37.67917   144.89390
13575     652.0           NaN     1981.0          NaN -37.90562   145.16761
13576     333.0         133.0     1995.0          NaN -37.85927   144.87904
13577     436.0           NaN     1997.0          NaN -37.85274   144.88738
13578     866.0         157.0     1920.0          NaN -37.85908   144.89299
13579     362.0         112.0     1920.0          NaN -37.81188   144.88449

                     Regionname Propertycount
13573        Western Metropolitan       16166.0
13574       Northern Metropolitan        2474.0
13575  South-Eastern Metropolitan        7392.0
13576        Western Metropolitan        6380.0
13577        Western Metropolitan        6380.0
13578        Western Metropolitan        6380.0
13579        Western Metropolitan        6543.0

[7 rows x 21 columns]
```

[3 points] The *first 5 records* of all attributes with a specific condition: `Landsize < 500`

[10]:
```
# Write your code here
df[df['Landsize'] < 500].head()
```

[10]:

	Suburb	Address	Rooms	Type	Price	Method	SellerG	\
0	Abbotsford	85 Turner St	2	h	1480000.0	S	Biggin	
1	Abbotsford	25 Bloomburg St	2	h	1035000.0	S	Biggin	
2	Abbotsford	5 Charles St	3	h	1465000.0	SP	Biggin	
3	Abbotsford	40 Federation La	3	h	850000.0	PI	Biggin	
4	Abbotsford	55a Park St	4	h	1600000.0	VB	Nelson	

	Date	Distance	Postcode	…	Bathroom	Car	Landsize	BuildingArea	\
0	3/12/2016	2.5	3067.0	…	1.0	1.0	202.0	NaN	
1	4/02/2016	2.5	3067.0	…	1.0	0.0	156.0	79.0	
2	4/03/2017	2.5	3067.0	…	2.0	0.0	134.0	150.0	
3	4/03/2017	2.5	3067.0	…	2.0	1.0	94.0	NaN	
4	4/06/2016	2.5	3067.0	…	1.0	2.0	120.0	142.0	

	YearBuilt	CouncilArea	Lattitude	Longtitude	Regionname	\
0	NaN	Yarra	-37.7996	144.9984	Northern Metropolitan	
1	1900.0	Yarra	-37.8079	144.9934	Northern Metropolitan	
2	1900.0	Yarra	-37.8093	144.9944	Northern Metropolitan	
3	NaN	Yarra	-37.7969	144.9969	Northern Metropolitan	
4	2014.0	Yarra	-37.8072	144.9941	Northern Metropolitan	

	Propertycount
0	4019.0
1	4019.0
2	4019.0
3	4019.0
4	4019.0

[5 rows x 21 columns]

[1 point] Descriptive statistics of *ALL attributes*

[12]:
```
# Write your code here
df.describe()
```

[12]:

	Rooms	Price	Distance	Postcode	Bedroom2	\
count	13580.000000	1.358000e+04	13580.000000	13580.000000	13580.000000	
mean	2.937997	1.075684e+06	10.137776	3105.301915	2.914728	
std	0.955748	6.393107e+05	5.868725	90.676964	0.965921	
min	1.000000	8.500000e+04	0.000000	3000.000000	0.000000	
25%	2.000000	6.500000e+05	6.100000	3044.000000	2.000000	
50%	3.000000	9.030000e+05	9.200000	3084.000000	3.000000	
75%	3.000000	1.330000e+06	13.000000	3148.000000	3.000000	
max	10.000000	9.000000e+06	48.100000	3977.000000	20.000000	

```
           Bathroom           Car       Landsize  BuildingArea    YearBuilt  \
count  13580.000000  13518.000000   13580.000000   7130.000000  8205.000000
mean       1.534242      1.610075     558.416127    151.967650  1964.684217
std        0.691712      0.962634    3990.669241    541.014538    37.273762
min        0.000000      0.000000       0.000000      0.000000  1196.000000
25%        1.000000      1.000000     177.000000     93.000000  1940.000000
50%        1.000000      2.000000     440.000000    126.000000  1970.000000
75%        2.000000      2.000000     651.000000    174.000000  1999.000000
max        8.000000     10.000000  433014.000000  44515.000000  2018.000000

           Lattitude    Longtitude  Propertycount
count  13580.000000  13580.000000   13580.000000
mean     -37.809203    144.995216    7454.417378
std        0.079260      0.103916    4378.581772
min      -38.182550    144.431810     249.000000
25%      -37.856822    144.929600    4380.000000
50%      -37.802355    145.000100    6555.000000
75%      -37.756400    145.058305   10331.000000
max      -37.408530    145.526350   21650.000000
```

[1 point] Descriptive statistics of *one selected attribute*: `Price`

[14]:
```
# Write your code here
df['Price'].describe()
```

[14]:
```
count    1.358000e+04
mean     1.075684e+06
std      6.393107e+05
min      8.500000e+04
25%      6.500000e+05
50%      9.030000e+05
75%      1.330000e+06
max      9.000000e+06
Name: Price, dtype: float64
```

[3 points] Descriptive statistics of *three selected attributes*: `Distance`, `Landsize`, `Propertycount`

[17]:
```
# Write your code here
df[['Distance', 'Landsize', 'Propertycount']].describe()
```

[17]:
```
           Distance       Landsize  Propertycount
count  13580.000000   13580.000000   13580.000000
mean      10.137776     558.416127    7454.417378
std        5.868725    3990.669241    4378.581772
min        0.000000       0.000000     249.000000
25%        6.100000     177.000000    4380.000000
50%        9.200000     440.000000    6555.000000
75%       13.000000     651.000000   10331.000000
```

```
max       48.100000  433014.000000   21650.000000
```

1.1.2 Subtask #2: Inspect data

2.1) Find out if there are any missing values (Total points = 7) Use the command below (`df.info()`).

[16]:
```
df.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  float64
 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  float64
 10  Bedroom2       13580 non-null  float64
 11  Bathroom       13580 non-null  float64
 12  Car            13518 non-null  float64
 13  Landsize       13580 non-null  float64
 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  float64
dtypes: float64(12), int64(1), object(8)
memory usage: 2.2+ MB
```

[2 points] How many attributes contain missing values?

ANS: 4

[5 points] For those attributes answered in 2.1, how many values are missing for each attribute?

ANS:

Car : 13580 - 13518 = BuildingArea : 13580 - 7130 = YearBuilt : 13580 - 8205 = Council area : 13580 - 12211 =

1.1.3 Subtask #3: Handle missing data

[60]:
```
# copy dataframes
df_remove_rows = df
df_impute_zero = df
```

3.1 Simply remove rows (Total points = 15) [3 points] Use the dataframe `df_remove_rows`. Remove rows that contain missing values, then display the first 5 rows.

[61]:
```
# Write your code here
df_remove_rows = df_remove_rows.dropna()
df_remove_rows.head()
```

[61]:

	Suburb	Address	Rooms	Type	Price	Method	SellerG	\
1	Abbotsford	25 Bloomburg St	2	h	1035000.0	S	Biggin	
2	Abbotsford	5 Charles St	3	h	1465000.0	SP	Biggin	
4	Abbotsford	55a Park St	4	h	1600000.0	VB	Nelson	
6	Abbotsford	124 Yarra St	3	h	1876000.0	S	Nelson	
7	Abbotsford	98 Charles St	2	h	1636000.0	S	Nelson	

	Date	Distance	Postcode	…	Bathroom	Car	Landsize	BuildingArea	\
1	4/02/2016	2.5	3067.0	…	1.0	0.0	156.0	79.0	
2	4/03/2017	2.5	3067.0	…	2.0	0.0	134.0	150.0	
4	4/06/2016	2.5	3067.0	…	1.0	2.0	120.0	142.0	
6	7/05/2016	2.5	3067.0	…	2.0	0.0	245.0	210.0	
7	8/10/2016	2.5	3067.0	…	1.0	2.0	256.0	107.0	

	YearBuilt	CouncilArea	Lattitude	Longtitude	Regionname	\
1	1900.0	Yarra	-37.8079	144.9934	Northern Metropolitan	
2	1900.0	Yarra	-37.8093	144.9944	Northern Metropolitan	
4	2014.0	Yarra	-37.8072	144.9941	Northern Metropolitan	
6	1910.0	Yarra	-37.8024	144.9993	Northern Metropolitan	
7	1890.0	Yarra	-37.8060	144.9954	Northern Metropolitan	

	Propertycount
1	4019.0
2	4019.0
4	4019.0
6	4019.0
7	4019.0

[5 rows x 21 columns]

[1 point] Display data shape after the removal of missing values.

[62]:
```
# Write your code here
print(df_remove_rows.shape)
```

```
(6196, 21)
```

[2 points] Compute average *price* after the removal of missing values.

[63]:
```
# Write your code here
df_remove_rows['Price'].mean()
```

[63]:
```
np.float64(1068828.202065849)
```

[4 points] Compute the difference of the average *price* before and after the removal of missing values.

[64]:
```
# Write your code here
df['Price'].mean() - df_remove_rows['Price'].mean()
```

[64]:
```
np.float64(6855.87738923193)
```

[5 points] Discuss if this method should be used, and why

ANS: We can see, in a small dataset, it's preferable to drop the rows with NA/null data because they can impact significantly and negativly our results

3.2 Replace missing values with zeros (Total points = 14) [3 points] Use the dataframe `df_impute_zero`. Replace missing values with zeros, then display the first 5 rows.

[65]:
```
# Write your code here
df_impute_zero = df_impute_zero.fillna(0)
df_impute_zero.head()
```

[65]:

	Suburb	Address	Rooms	Type	Price	Method	SellerG	\
0	Abbotsford	85 Turner St	2	h	1480000.0	S	Biggin	
1	Abbotsford	25 Bloomburg St	2	h	1035000.0	S	Biggin	
2	Abbotsford	5 Charles St	3	h	1465000.0	SP	Biggin	
3	Abbotsford	40 Federation La	3	h	850000.0	PI	Biggin	
4	Abbotsford	55a Park St	4	h	1600000.0	VB	Nelson	

	Date	Distance	Postcode	…	Bathroom	Car	Landsize	BuildingArea	\
0	3/12/2016	2.5	3067.0	…	1.0	1.0	202.0	0.0	
1	4/02/2016	2.5	3067.0	…	1.0	0.0	156.0	79.0	
2	4/03/2017	2.5	3067.0	…	2.0	0.0	134.0	150.0	
3	4/03/2017	2.5	3067.0	…	2.0	1.0	94.0	0.0	
4	4/06/2016	2.5	3067.0	…	1.0	2.0	120.0	142.0	

	YearBuilt	CouncilArea	Lattitude	Longtitude	Regionname	\
0	0.0	Yarra	-37.7996	144.9984	Northern Metropolitan	
1	1900.0	Yarra	-37.8079	144.9934	Northern Metropolitan	
2	1900.0	Yarra	-37.8093	144.9944	Northern Metropolitan	
3	0.0	Yarra	-37.7969	144.9969	Northern Metropolitan	
4	2014.0	Yarra	-37.8072	144.9941	Northern Metropolitan	

	Propertycount
0	4019.0

```
1        4019.0
2        4019.0
3        4019.0
4        4019.0

[5 rows x 21 columns]
```

[2 points] Compute an average of the attribute `BuildingArea` after the data imputation.

[73]:
```
# Write your code here
df_impute_zero['BuildingArea'].mean()
# df_impute_zero.groupby('BuildingArea')['Price'].mean()
```

[73]:
```
np.float64(79.78861146539029)
```

[4 points] Compute the difference of the average `BuildingArea` before versus after replacing missing values with zeros.

[92]:
```
# Write your code here
df_impute_zero['BuildingArea'].mean() - df['BuildingArea'].mean()
```

[92]:
```
np.float64(-72.17903842240776)
```

[5 points] Discuss if this method should be used, and why

ANS: Both method are ok, there is no significantly differences between null/NA & 0

3.3 Replace missing values with average (Total points = 21) [3 points] Create a new dataframe `df_new` which contains only these attributes: `Rooms`, `Price`, `Distance`, `BuildingArea`. Then, likewise, display the first 5 rows.

[72]:
```
# Write your code here
df_new= df[['Rooms', 'Price', 'Distance', 'BuildingArea']]
df_new.head()
```

[72]:

	Rooms	Price	Distance	BuildingArea
0	2	1480000.0	2.5	NaN
1	2	1035000.0	2.5	79.0
2	3	1465000.0	2.5	150.0
3	3	850000.0	2.5	NaN
4	4	1600000.0	2.5	142.0

We are going to replace missing values in the attribute `BuildingArea` with its average.

[3 points] First, compute average value of the attribute `BuildingArea`.

[83]:
```
# Write your code here
BuildingAreaMean = df_new['BuildingArea'].mean()
print("The mean of the building area is :", BuildingAreaMean)
```

```
The mean of the building area is : 151.96764988779805
```

[3 points] Now replace any missing values in `BuildingArea` with its average. Also display the first 10 rows.

[85]:
```
# Write your code here
df_new = df_new.fillna(BuildingAreaMean)

df.head(10)
```

[85]:

	Suburb	Address	Rooms	Type	Price	Method	SellerG
0	Abbotsford	85 Turner St	2	h	1480000.0	S	Biggin
1	Abbotsford	25 Bloomburg St	2	h	1035000.0	S	Biggin
2	Abbotsford	5 Charles St	3	h	1465000.0	SP	Biggin
3	Abbotsford	40 Federation La	3	h	850000.0	PI	Biggin
4	Abbotsford	55a Park St	4	h	1600000.0	VB	Nelson
5	Abbotsford	129 Charles St	2	h	941000.0	S	Jellis
6	Abbotsford	124 Yarra St	3	h	1876000.0	S	Nelson
7	Abbotsford	98 Charles St	2	h	1636000.0	S	Nelson
8	Abbotsford	6/241 Nicholson St	1	u	300000.0	S	Biggin
9	Abbotsford	10 Valiant St	2	h	1097000.0	S	Biggin

	Date	Distance	Postcode	…	Bathroom	Car	Landsize	BuildingArea
0	3/12/2016	2.5	3067.0	…	1.0	1.0	202.0	NaN
1	4/02/2016	2.5	3067.0	…	1.0	0.0	156.0	79.0
2	4/03/2017	2.5	3067.0	…	2.0	0.0	134.0	150.0
3	4/03/2017	2.5	3067.0	…	2.0	1.0	94.0	NaN
4	4/06/2016	2.5	3067.0	…	1.0	2.0	120.0	142.0
5	7/05/2016	2.5	3067.0	…	1.0	0.0	181.0	NaN
6	7/05/2016	2.5	3067.0	…	2.0	0.0	245.0	210.0
7	8/10/2016	2.5	3067.0	…	1.0	2.0	256.0	107.0
8	8/10/2016	2.5	3067.0	…	1.0	1.0	0.0	NaN
9	8/10/2016	2.5	3067.0	…	1.0	2.0	220.0	75.0

	YearBuilt	CouncilArea	Lattitude	Longtitude	Regionname
0	NaN	Yarra	-37.7996	144.9984	Northern Metropolitan
1	1900.0	Yarra	-37.8079	144.9934	Northern Metropolitan
2	1900.0	Yarra	-37.8093	144.9944	Northern Metropolitan
3	NaN	Yarra	-37.7969	144.9969	Northern Metropolitan
4	2014.0	Yarra	-37.8072	144.9941	Northern Metropolitan
5	NaN	Yarra	-37.8041	144.9953	Northern Metropolitan
6	1910.0	Yarra	-37.8024	144.9993	Northern Metropolitan
7	1890.0	Yarra	-37.8060	144.9954	Northern Metropolitan
8	NaN	Yarra	-37.8008	144.9973	Northern Metropolitan
9	1900.0	Yarra	-37.8010	144.9989	Northern Metropolitan

	Propertycount
0	4019.0
1	4019.0

```
2        4019.0
3        4019.0
4        4019.0
5        4019.0
6        4019.0
7        4019.0
8        4019.0
9        4019.0

[10 rows x 21 columns]
```

[3 points] Now, compute a standard deviation of `BuildingArea` after the data imputation.

[86]:
```
# Write your code here
BuildingAreStandard = df_new['BuildingArea'].std()

print("The standard deviation is:", BuildingAreStandard)
```

```
The standard deviation is: 392.0029618346695
```

[4 points] Compute the difference of the `BuildingArea` standard deviation before versus after replacing missing values with its mean value.

[97]:
```
# Write your code here
BuildingAreStandard - df['BuildingArea'].std()
```

[97]:
```
-149.01157579168176
```

[5 points] From the result, discuss if this method should be used, and why

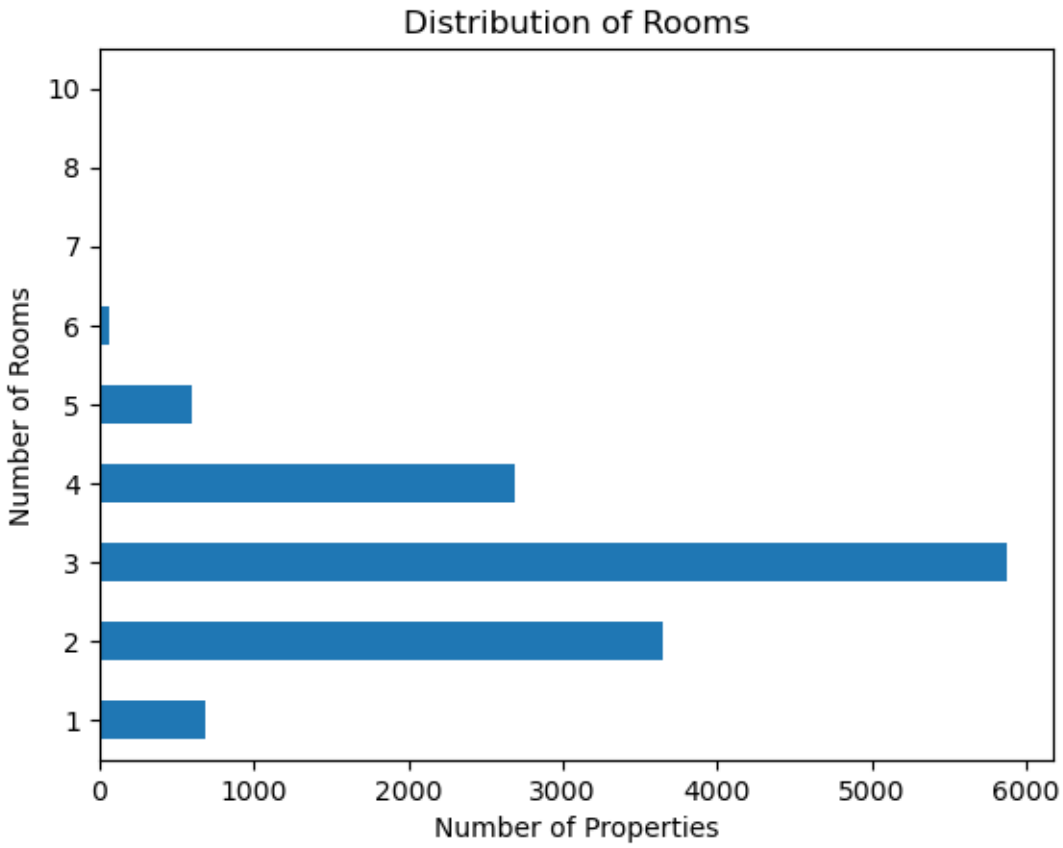
ANS:

1.1.4 Subtask #4: Create some visualizations

[39]:
```
import matplotlib.pyplot as plt
```

4.1 Investigate types of rooms (Total points = 8) [5 points] *Use the original dataframe* `df`, create a *horizontal* bar chart to present the attributes `Rooms`. Also display the chart title and data labels.

[106]:
```
# Write your code here
df['Rooms'].value_counts().sort_index().plot(kind='barh')
plt.title('Distribution of Rooms')
plt.xlabel('Number of Properties')
plt.ylabel('Number of Rooms')
plt.show()
```



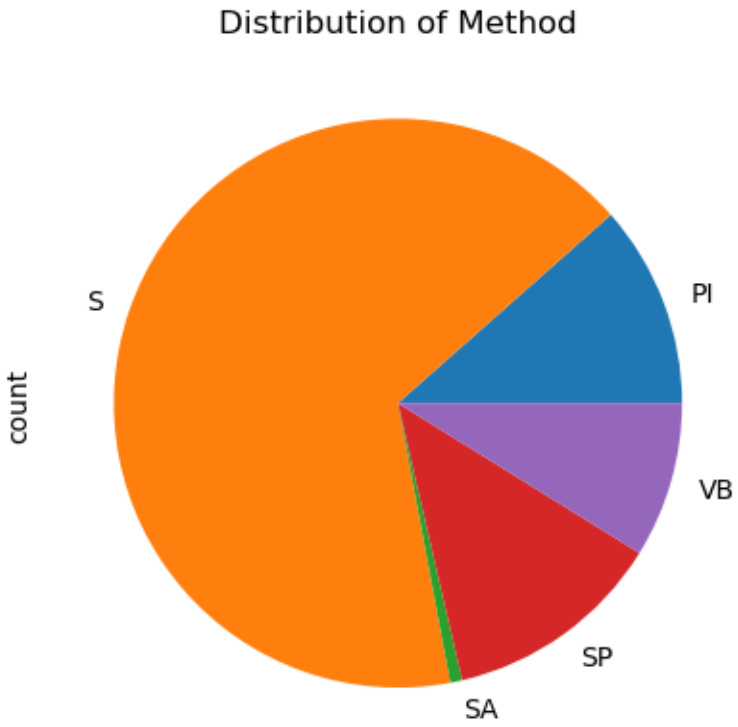
[3 points] Briefly describe this visualization in your own words, and does this information help with the decision making?

ANS: in the x axis we have number of properties, and in the y axies we have distribution of rooms, we can se a normal distribution, most diponibility in 3 Rooms and less with 1 and 6

4.2 Investigate types of methods (Total points = 8) [5 points] *Use the original dataframe* `df`, create a pie chart to present the attributes `Method`. Also display the chart title and data labels.

[105]:
```
# Write your code here

df['Method'].value_counts().sort_index().plot(kind='pie')
plt.title('Distribution of Method')
plt.show()
```



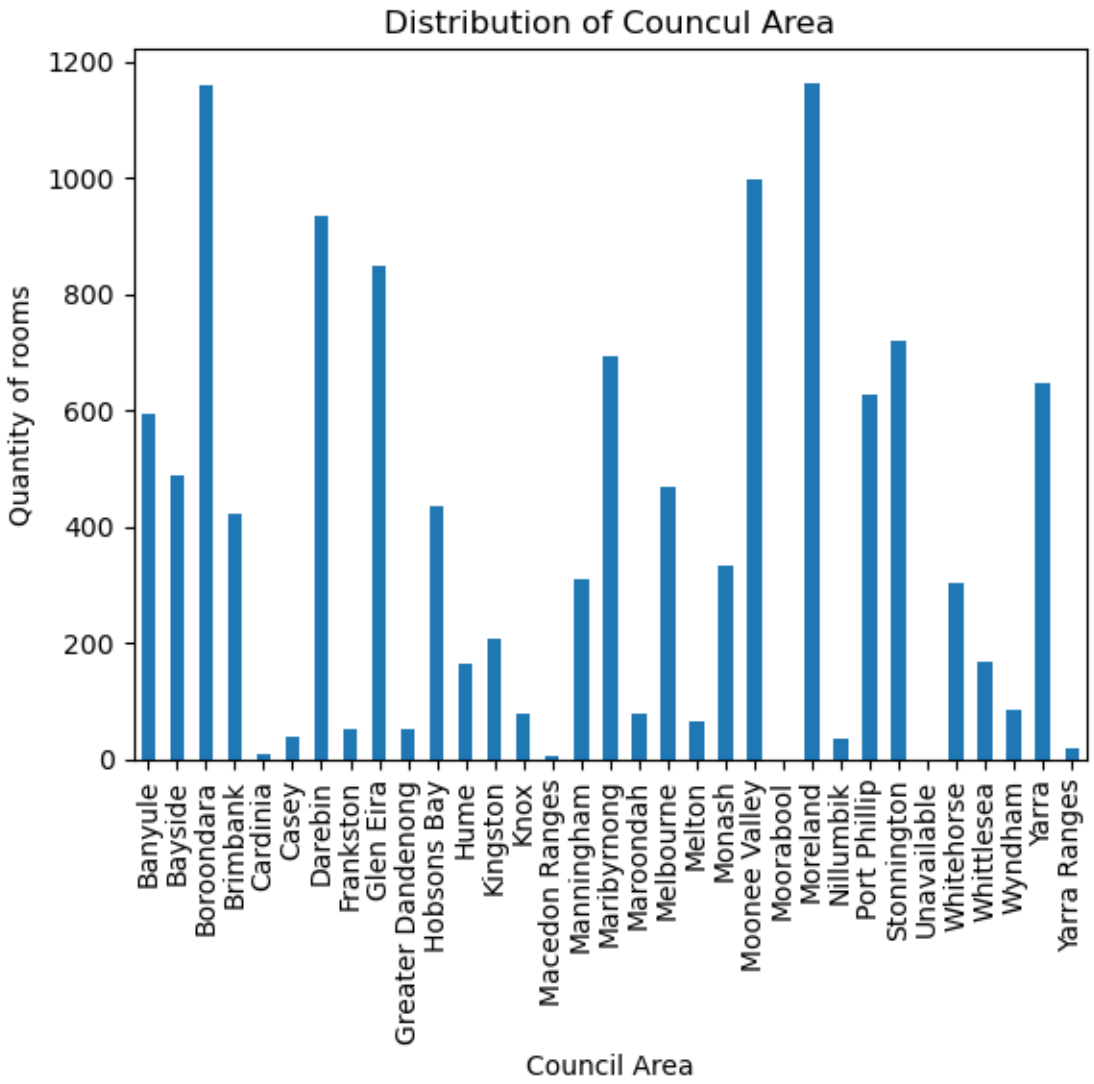
[3 points] Briefly describe this visualization in your own words, and does this information help with the decision making?

ANS:

4.3 Investigate council areas (Total points = 8) [5 points] *Use the original dataframe* `df`, create a bar chart to present the attributes `CouncilArea`. Also display the chart title and data labels.

[104]:
```
# Write your code here

df['CouncilArea'].value_counts().sort_index().plot(kind='bar')
plt.title('Distribution of Councul Area')
plt.xlabel('Council Area')
plt.ylabel('Quantity of rooms')
plt.show()
```



[3 points] Briefly describe this visualization in your own words, and does this information help with the decision making?

ANS:

1.1.5 Subtask #5: Group the data

[4 points] Group the data by `Regionname` and `Type`, then display the *sum* of these attributes: `Price`, `Bedroom2`, `Bathroom`, `Car`, and `Landsize`. The expected output looks like the provided snapshot.

```
[ ]: # Write your code here
```

[2 points] Briefly describe your understanding from this output.

ANS: