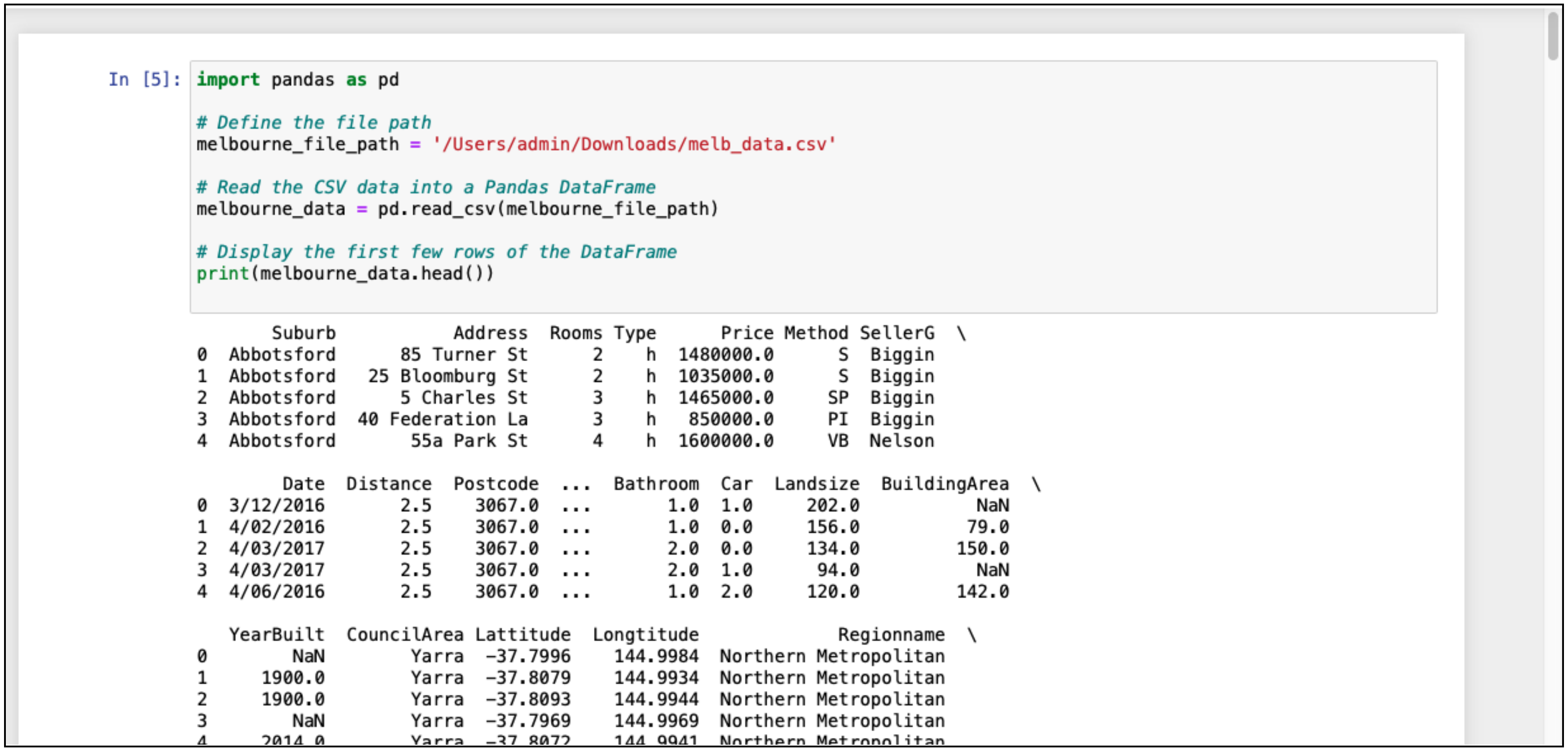
Melbourne housing(Data Cleaning)

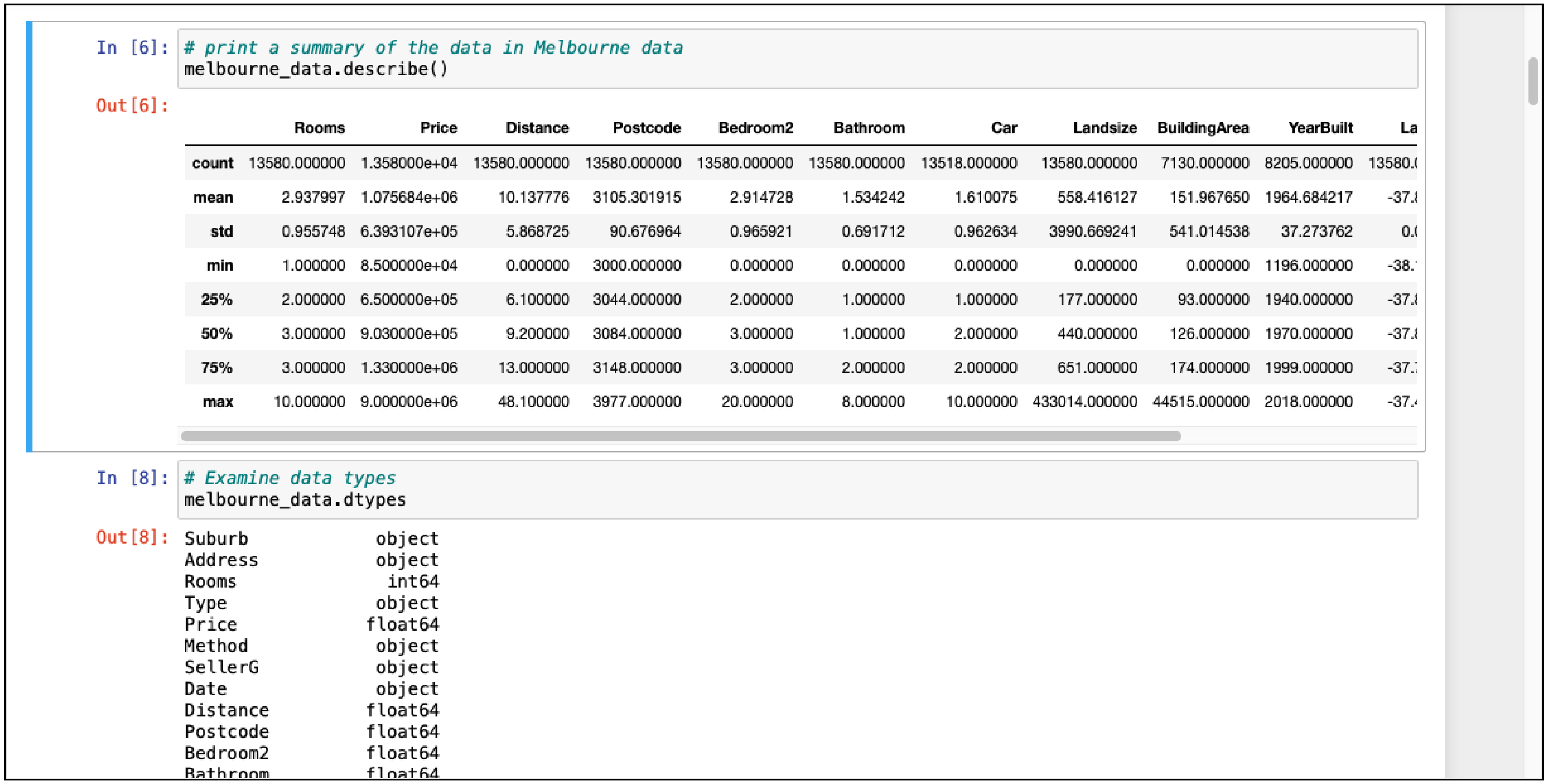
Load the data:



```
In [5]: import pandas as pd

        # Define the file path
        melbourne_file_path = '/Users/admin/Downloads/melb_data.csv'

        # Read the CSV data into a Pandas DataFrame
        melbourne_data = pd.read_csv(melbourne_file_path)

        # Display the first few rows of the DataFrame
        print(melbourne_data.head())
```

```
       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
```

Examine our Data:



In [6]:
```
# print a summary of the data in Melbourne data
melbourne_data.describe()
```

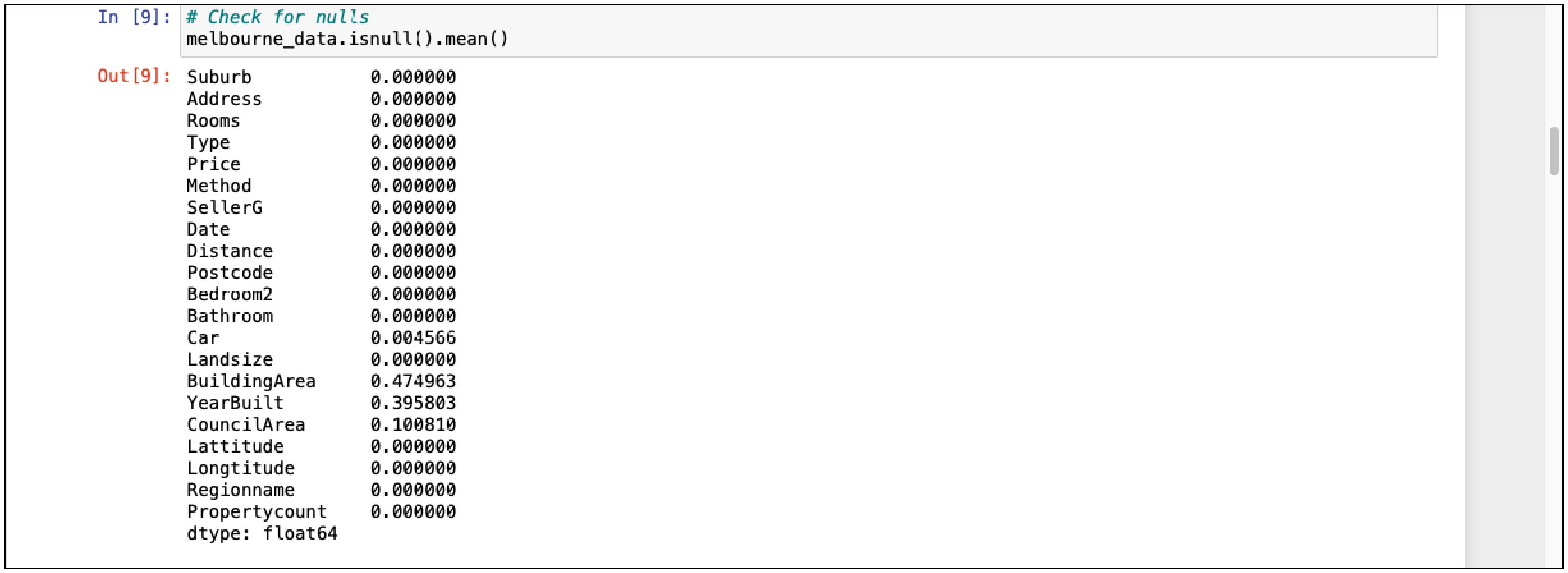
Out[6]:

	Rooms	Price	Distance	Postcode	Bedroom2	Bathroom	Car	Landsize	BuildingArea	YearBuilt	La
count	13580.000000	1.358000e+04	13580.000000	13580.000000	13580.000000	13580.000000	13518.000000	13580.000000	7130.000000	8205.000000	13580.
mean	2.937997	1.075684e+06	10.137776	3105.301915	2.914728	1.534242	1.610075	558.416127	151.967650	1964.684217	-37.
std	0.955748	6.393107e+05	5.868725	90.676964	0.965921	0.691712	0.962634	3990.669241	541.014538	37.273762	0.
min	1.000000	8.500000e+04	0.000000	3000.000000	0.000000	0.000000	0.000000	0.000000	0.000000	1196.000000	-38.
25%	2.000000	6.500000e+05	6.100000	3044.000000	2.000000	1.000000	1.000000	177.000000	93.000000	1940.000000	-37.
50%	3.000000	9.030000e+05	9.200000	3084.000000	3.000000	1.000000	2.000000	440.000000	126.000000	1970.000000	-37.
75%	3.000000	1.330000e+06	13.000000	3148.000000	3.000000	2.000000	2.000000	651.000000	174.000000	1999.000000	-37.
max	10.000000	9.000000e+06	48.100000	3977.000000	20.000000	8.000000	10.000000	433014.000000	44515.000000	2018.000000	-37.

In [8]:
```
# Examine data types
melbourne_data.dtypes
```

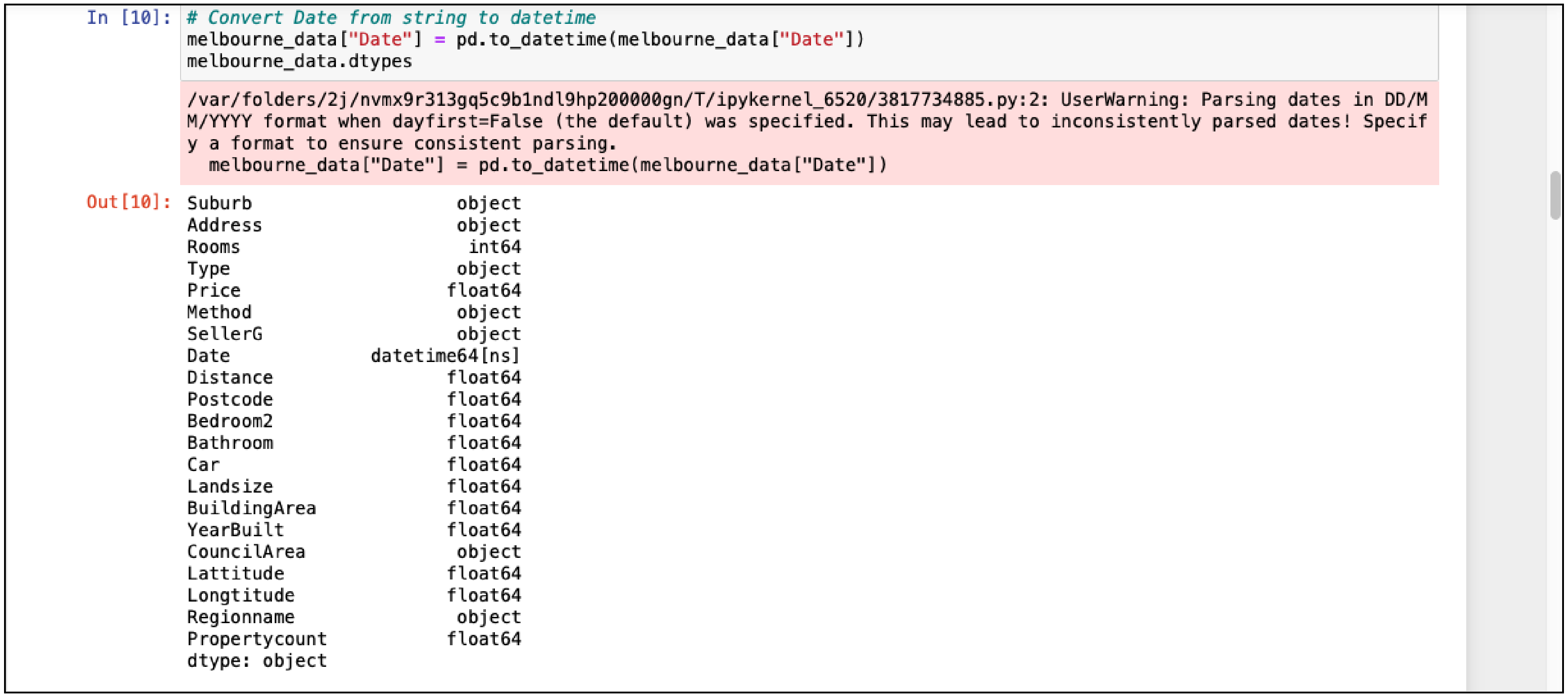
Out[8]:
```
Suburb           object
Address          object
Rooms             int64
Type             object
Price           float64
Method           object
SellerG          object
Date             object
Distance        float64
Postcode        float64
Bedroom2        float64
Bathroom        float64
```

Checked for Nulls:



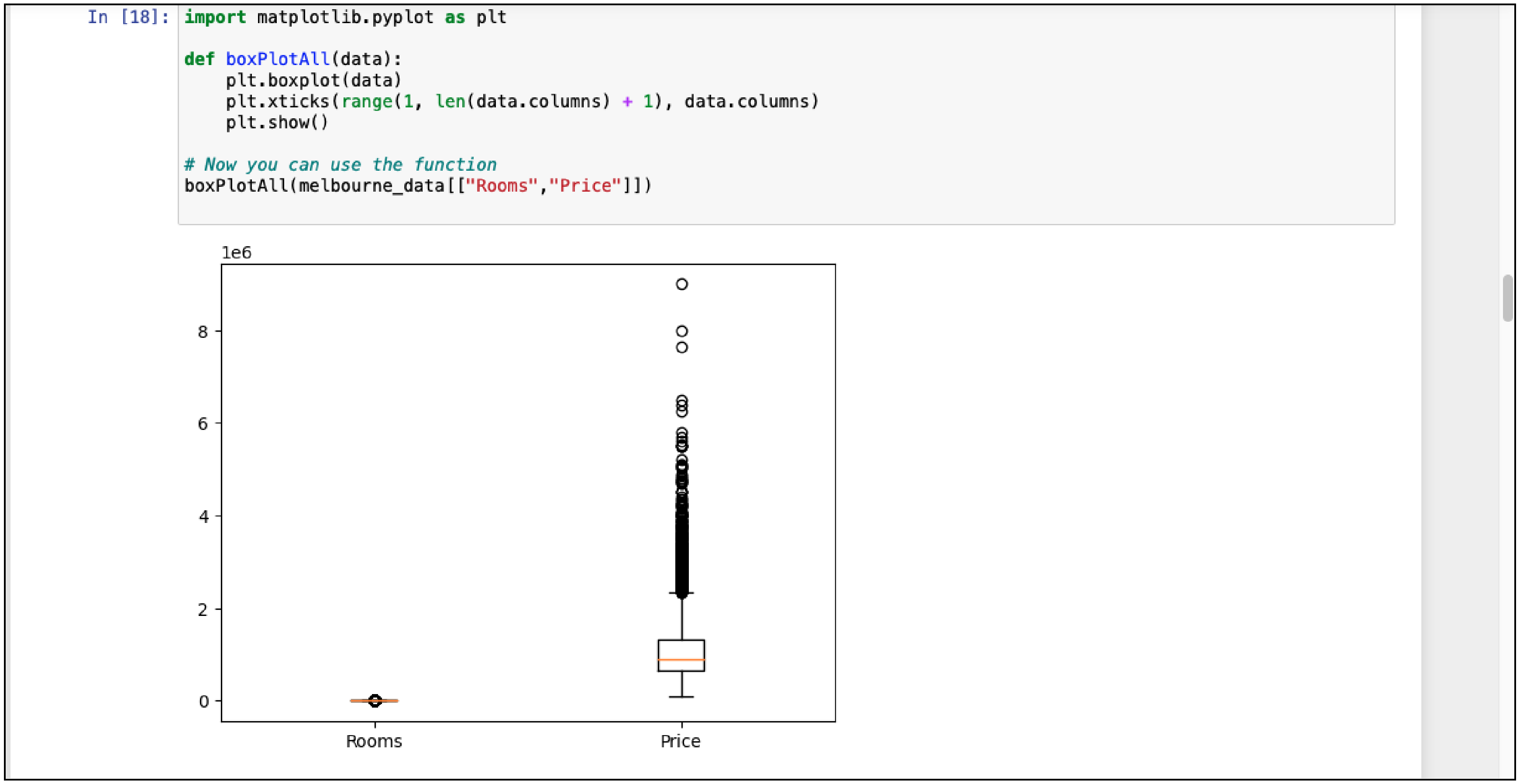
```
In [9]: # Check for nulls
        melbourne_data.isnull().mean()

Out[9]: Suburb           0.000000
        Address          0.000000
        Rooms            0.000000
        Type             0.000000
        Price            0.000000
        Method           0.000000
        SellerG          0.000000
        Date             0.000000
        Distance         0.000000
        Postcode         0.000000
        Bedroom2         0.000000
        Bathroom         0.000000
        Car              0.004566
        Landsize         0.000000
        BuildingArea     0.474963
        YearBuilt        0.395803
        CouncilArea      0.100810
        Lattitude        0.000000
        Longtitude       0.000000
        Regionname       0.000000
        Propertycount    0.000000
        dtype: float64
```

Processed Date:



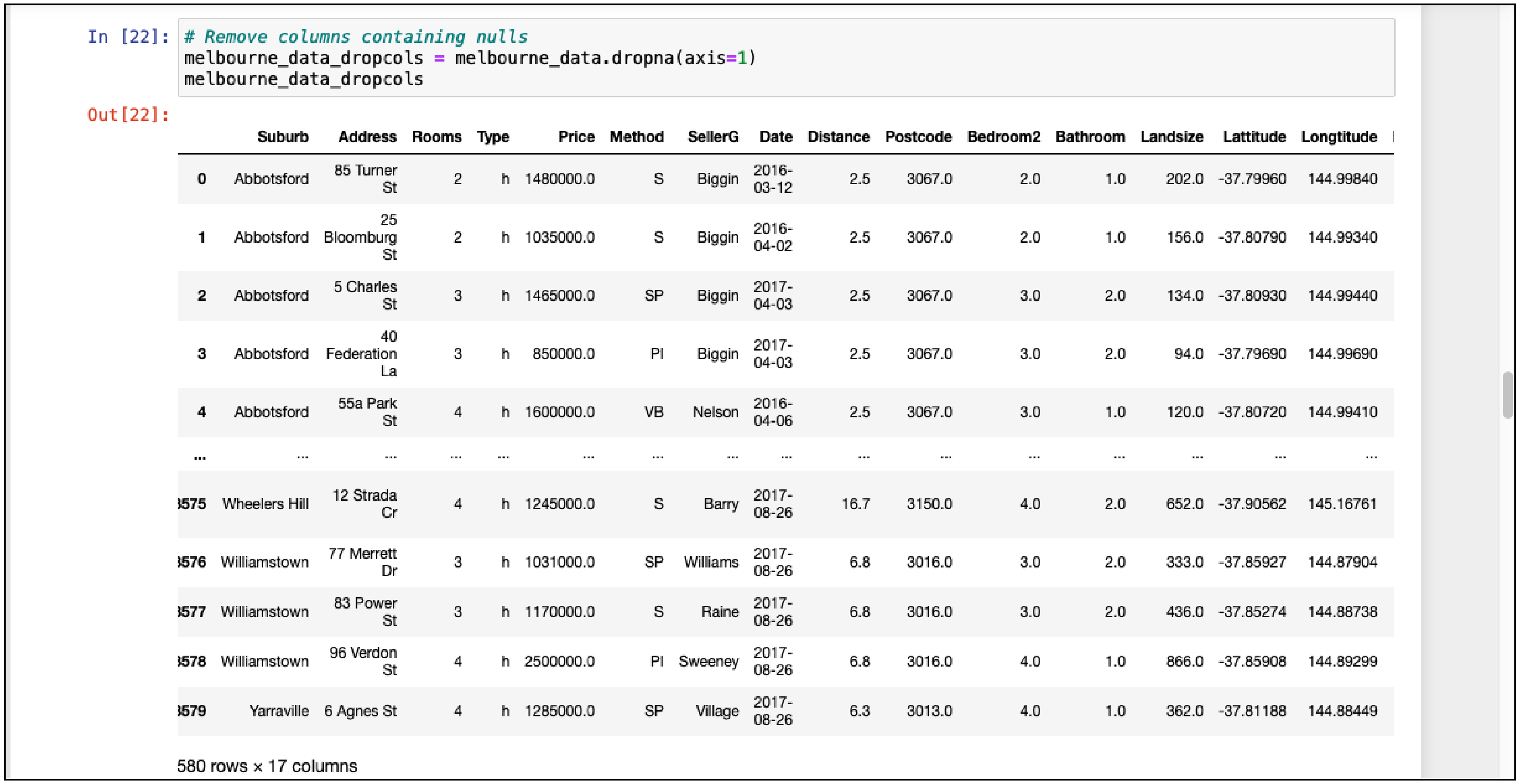
```
In [10]: # Convert Date from string to datetime
         melbourne_data["Date"] = pd.to_datetime(melbourne_data["Date"])
         melbourne_data.dtypes

         /var/folders/2j/nvmx9r313gq5c9b1ndl9hp200000gn/T/ipykernel_6520/3817734885.py:2: UserWarning: Parsing dates in DD/M
         M/YYYY format when dayfirst=False (the default) was specified. This may lead to inconsistently parsed dates! Specif
         y a format to ensure consistent parsing.
           melbourne_data["Date"] = pd.to_datetime(melbourne_data["Date"])

Out[10]: Suburb                   object
         Address                  object
         Rooms                     int64
         Type                     object
         Price                   float64
         Method                   object
         SellerG                  object
         Date             datetime64[ns]
         Distance                float64
         Postcode                float64
         Bedroom2                float64
         Bathroom                float64
         Car                     float64
         Landsize                float64
         BuildingArea            float64
         YearBuilt               float64
         CouncilArea              object
         Lattitude               float64
         Longtitude              float64
         Regionname               object
         Propertycount           float64
         dtype: object
```

Plotted boxplot to identify outliers:



```
In [18]: import matplotlib.pyplot as plt

         def boxPlotAll(data):
             plt.boxplot(data)
             plt.xticks(range(1, len(data.columns) + 1), data.columns)
             plt.show()

         # Now you can use the function
         boxPlotAll(melbourne_data[["Rooms","Price"]])
```

Removed all columns containing nulls:



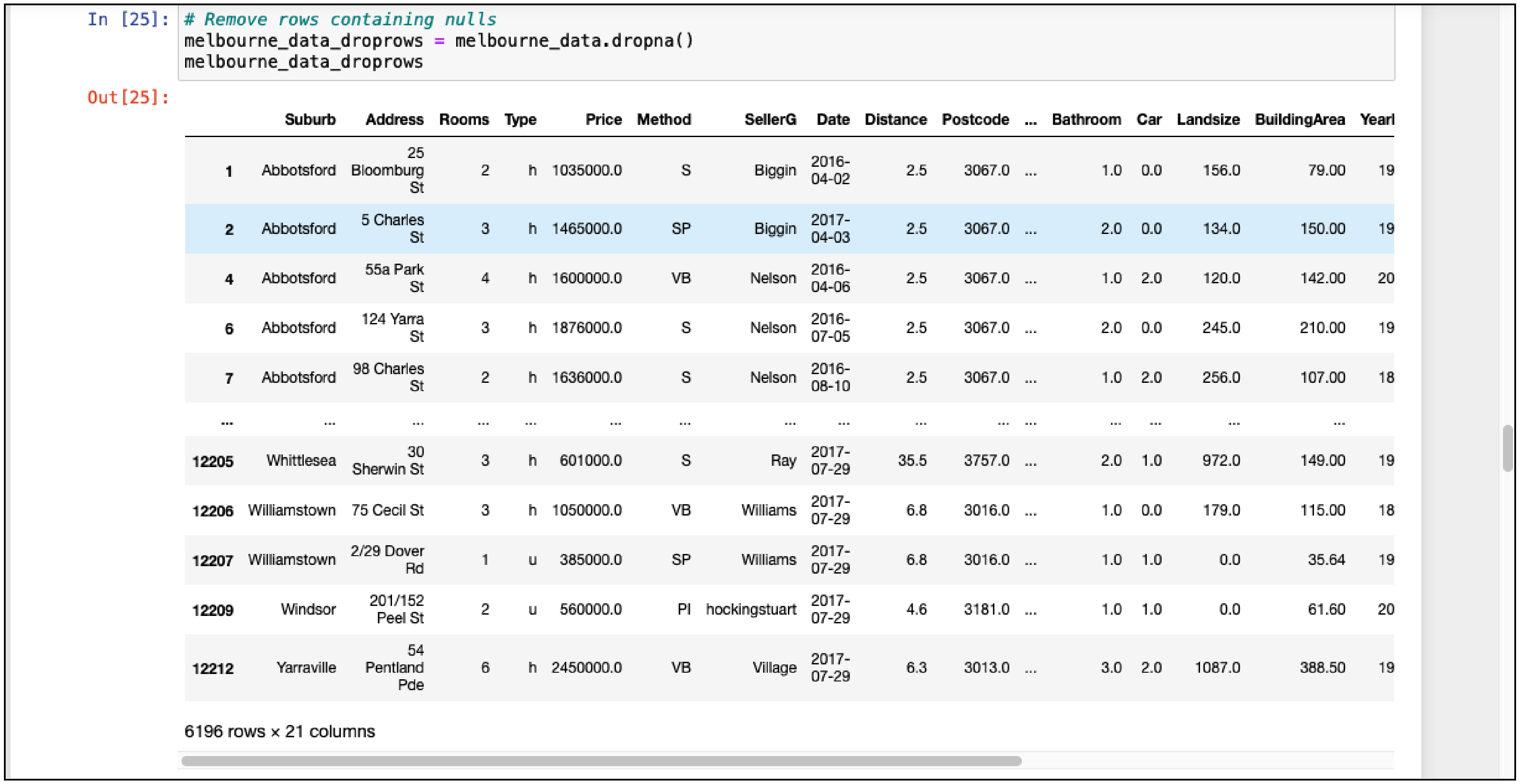
In [22]:
```python
# Remove columns containing nulls
melbourne_data_dropcols = melbourne_data.dropna(axis=1)
melbourne_data_dropcols
```

Out[22]:

	Suburb	Address	Rooms	Type	Price	Method	SellerG	Date	Distance	Postcode	Bedroom2	Bathroom	Landsize	Lattitude	Longtitude
0	Abbotsford	85 Turner St	2	h	1480000.0	S	Biggin	2016-03-12	2.5	3067.0	2.0	1.0	202.0	-37.79960	144.99840
1	Abbotsford	25 Bloomburg St	2	h	1035000.0	S	Biggin	2016-04-02	2.5	3067.0	2.0	1.0	156.0	-37.80790	144.99340
2	Abbotsford	5 Charles St	3	h	1465000.0	SP	Biggin	2017-04-03	2.5	3067.0	3.0	2.0	134.0	-37.80930	144.99440
3	Abbotsford	40 Federation La	3	h	850000.0	PI	Biggin	2017-04-03	2.5	3067.0	3.0	2.0	94.0	-37.79690	144.99690
4	Abbotsford	55a Park St	4	h	1600000.0	VB	Nelson	2016-04-06	2.5	3067.0	3.0	1.0	120.0	-37.80720	144.99410
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
3575	Wheelers Hill	12 Strada Cr	4	h	1245000.0	S	Barry	2017-08-26	16.7	3150.0	4.0	2.0	652.0	-37.90562	145.16761
3576	Williamstown	77 Merrett Dr	3	h	1031000.0	SP	Williams	2017-08-26	6.8	3016.0	3.0	2.0	333.0	-37.85927	144.87904
3577	Williamstown	83 Power St	3	h	1170000.0	S	Raine	2017-08-26	6.8	3016.0	3.0	2.0	436.0	-37.85274	144.88738
3578	Williamstown	96 Verdon St	4	h	2500000.0	PI	Sweeney	2017-08-26	6.8	3016.0	4.0	1.0	866.0	-37.85908	144.89299
3579	Yarraville	6 Agnes St	4	h	1285000.0	SP	Village	2017-08-26	6.3	3013.0	4.0	1.0	362.0	-37.81188	144.88449

580 rows × 17 columns

Removed all rows containing nulls:



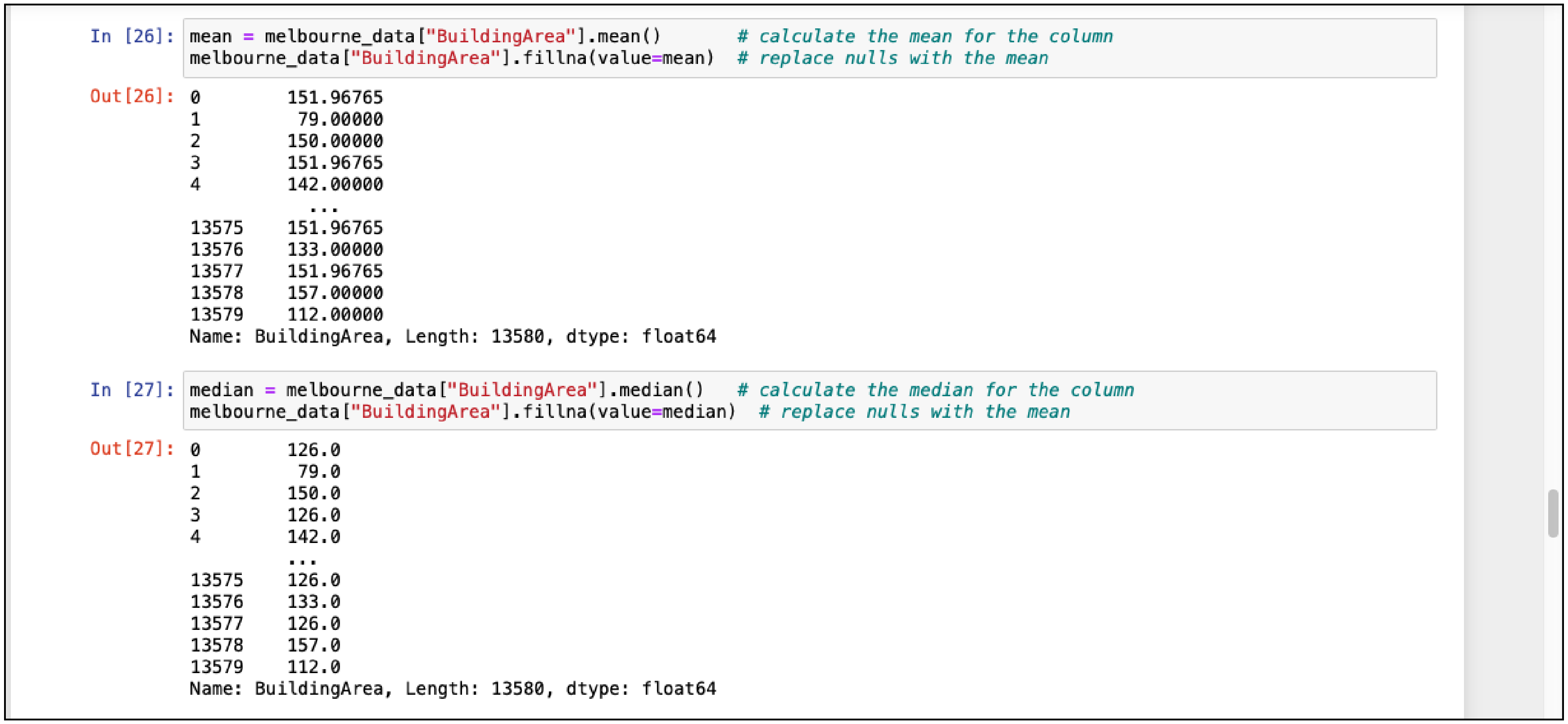
```
In [25]: # Remove rows containing nulls
         melbourne_data_droprows = melbourne_data.dropna()
         melbourne_data_droprows
```

Out[25]:

	Suburb	Address	Rooms	Type	Price	Method	SellerG	Date	Distance	Postcode	...	Bathroom	Car	Landsize	BuildingArea	Year
1	Abbotsford	25 Bloomburg St	2	h	1035000.0	S	Biggin	2016-04-02	2.5	3067.0	...	1.0	0.0	156.0	79.00	19
2	Abbotsford	5 Charles St	3	h	1465000.0	SP	Biggin	2017-04-03	2.5	3067.0	...	2.0	0.0	134.0	150.00	19
4	Abbotsford	55a Park St	4	h	1600000.0	VB	Nelson	2016-04-06	2.5	3067.0	...	1.0	2.0	120.0	142.00	20
6	Abbotsford	124 Yarra St	3	h	1876000.0	S	Nelson	2016-07-05	2.5	3067.0	...	2.0	0.0	245.0	210.00	19
7	Abbotsford	98 Charles St	2	h	1636000.0	S	Nelson	2016-08-10	2.5	3067.0	...	1.0	2.0	256.0	107.00	18
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
12205	Whittlesea	30 Sherwin St	3	h	601000.0	S	Ray	2017-07-29	35.5	3757.0	...	2.0	1.0	972.0	149.00	19
12206	Williamstown	75 Cecil St	3	h	1050000.0	VB	Williams	2017-07-29	6.8	3016.0	...	1.0	0.0	179.0	115.00	18
12207	Williamstown	2/29 Dover Rd	1	u	385000.0	SP	Williams	2017-07-29	6.8	3016.0	...	1.0	1.0	0.0	35.64	19
12209	Windsor	201/152 Peel St	2	u	560000.0	PI	hockingstuart	2017-07-29	4.6	3181.0	...	1.0	1.0	0.0	61.60	20
12212	Yarraville	54 Pentland Pde	6	h	2450000.0	VB	Village	2017-07-29	6.3	3013.0	...	3.0	2.0	1087.0	388.50	19

6196 rows × 21 columns

Impute nulls with the mean for the column:



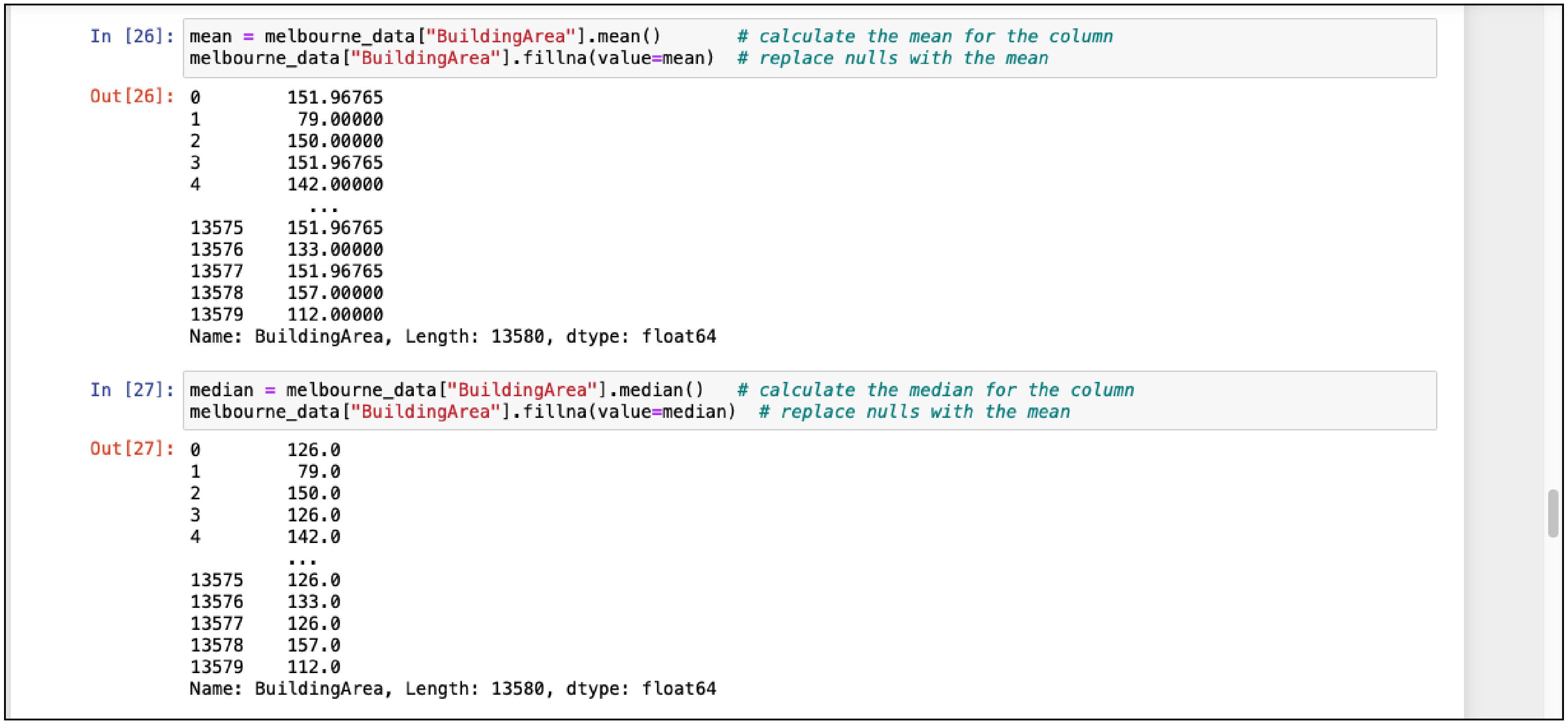
```
In [26]: mean = melbourne_data["BuildingArea"].mean()       # calculate the mean for the column
         melbourne_data["BuildingArea"].fillna(value=mean)  # replace nulls with the mean
```

```
Out[26]: 0        151.96765
         1         79.00000
         2        150.00000
         3        151.96765
         4        142.00000
                    ...
         13575    151.96765
         13576    133.00000
         13577    151.96765
         13578    157.00000
         13579    112.00000
         Name: BuildingArea, Length: 13580, dtype: float64
```

```
In [27]: median = melbourne_data["BuildingArea"].median()   # calculate the median for the column
         melbourne_data["BuildingArea"].fillna(value=median)  # replace nulls with the mean
```

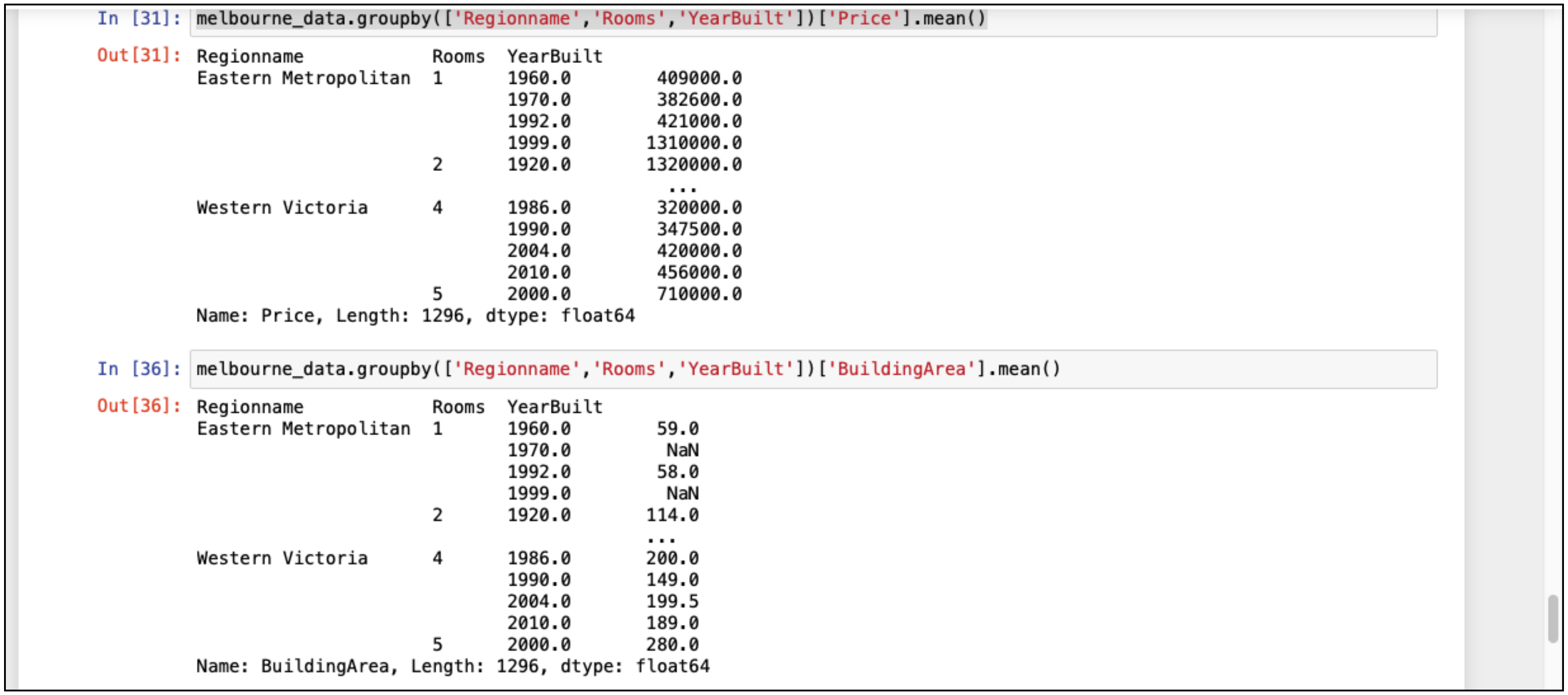
```
Out[27]: 0        126.0
         1         79.0
         2        150.0
         3        126.0
         4        142.0
                  ...
         13575    126.0
         13576    133.0
         13577    126.0
         13578    157.0
         13579    112.0
         Name: BuildingArea, Length: 13580, dtype: float64
```

Impute nulls with the median for the column:



```
In [26]: mean = melbourne_data["BuildingArea"].mean()       # calculate the mean for the column
         melbourne_data["BuildingArea"].fillna(value=mean)  # replace nulls with the mean

Out[26]: 0        151.96765
         1         79.00000
         2        150.00000
         3        151.96765
         4        142.00000
                    ...
         13575    151.96765
         13576    133.00000
         13577    151.96765
         13578    157.00000
         13579    112.00000
         Name: BuildingArea, Length: 13580, dtype: float64

In [27]: median = melbourne_data["BuildingArea"].median()   # calculate the median for the column
         melbourne_data["BuildingArea"].fillna(value=median)  # replace nulls with the mean

Out[27]: 0        126.0
         1         79.0
         2        150.0
         3        126.0
         4        142.0
                  ...
         13575    126.0
         13576    133.0
         13577    126.0
         13578    157.0
         13579    112.0
         Name: BuildingArea, Length: 13580, dtype: float64
```

Computed the mean of each region name, property size and year built:



```
In [31]: melbourne_data.groupby(['Regionname','Rooms','YearBuilt'])['Price'].mean()

Out[31]: Regionname            Rooms  YearBuilt
         Eastern Metropolitan  1      1960.0        409000.0
                                      1970.0        382600.0
                                      1992.0        421000.0
                                      1999.0       1310000.0
                               2      1920.0       1320000.0
                                                     ...
         Western Victoria      4      1986.0        320000.0
                                      1990.0        347500.0
                                      2004.0        420000.0
                                      2010.0        456000.0
                               5      2000.0        710000.0
         Name: Price, Length: 1296, dtype: float64

In [36]: melbourne_data.groupby(['Regionname','Rooms','YearBuilt'])['BuildingArea'].mean()

Out[36]: Regionname            Rooms  YearBuilt
         Eastern Metropolitan  1      1960.0        59.0
                                      1970.0         NaN
                                      1992.0        58.0
                                      1999.0         NaN
                               2      1920.0       114.0
                                                   ...
         Western Victoria      4      1986.0       200.0
                                      1990.0       149.0
                                      2004.0       199.5
                                      2010.0       189.0
                               5      2000.0       280.0
         Name: BuildingArea, Length: 1296, dtype: float64
```