

01_Exploratory_Analysis

October 03, 2023

1 flats-in-cracow exploratory data analysis

1.1 Imports

```
[1]: import matplotlib.pyplot as plt
import pandas as pd
import numpy as np

from sklearn.impute import KNNImputer
from pylab import rcParams
from pathlib import Path
```

1.2 Setup

```
[2]: # Create directory for images
Path("img").mkdir(parents=True, exist_ok=True)

# Set default figure size
rcParams['figure.figsize'] = (4, 4)

# Tell pandas how to display floats
pd.options.display.float_format = "{:,.2f}".format
```

1.3 Data loading

```
[3]: path = '../flats-data/cleaned_data.csv'
```

```
[4]: data = pd.read_csv(path, lineterminator='\n')
```

```
[5]: data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4592 entries, 0 to 4591
Data columns (total 17 columns):
 #   Column      Non-Null Count  Dtype
---  -
 0   District    4592 non-null   object
```

```

1  Amount      4592 non-null  int64
2  Seller      4592 non-null  object
3  Area        4592 non-null  int64
4  Rooms       4592 non-null  int64
5  Bathrooms   4592 non-null  int64
6  Parking     4592 non-null  object
7  Garden      4592 non-null  bool
8  Balcony     4592 non-null  bool
9  Terrace     4592 non-null  bool
10 Floor       4592 non-null  bool
11 New         4592 non-null  bool
12 Estate      4592 non-null  bool
13 Townhouse   4592 non-null  bool
14 Apartment   4592 non-null  bool
15 Land        4592 non-null  bool
16 Studio      4592 non-null  bool
dtypes: bool(10), int64(4), object(3)
memory usage: 296.1+ KB

```

```
[6]: data.head()
```

```

[6]:   District  Amount  Seller  Area  Rooms  Bathrooms  Parking  Garden  \
0  krowodrza  595000  realtor   78     4           2  no parking  False
1  podgorze  449000  realtor   61     3           1  no parking  False
2  nowa huta  449000  realtor   58     3           1  no parking  False
3  krowodrza  595000  realtor   78     4           2  no parking  False
4  krowodrza  430000  realtor   48     2           1    garage  False

   Balcony  Terrace  Floor   New  Estate  Townhouse  Apartment  Land  Studio
0     True    False  False  False  False     False     False  False  False
1     True    False   True  False  False     False     False  False  False
2     True    False  False   True  False     False     False  False  False
3     True    False  False  False  False     False     False  False  False
4     True    False   True  False   True     False     False  False  False

```

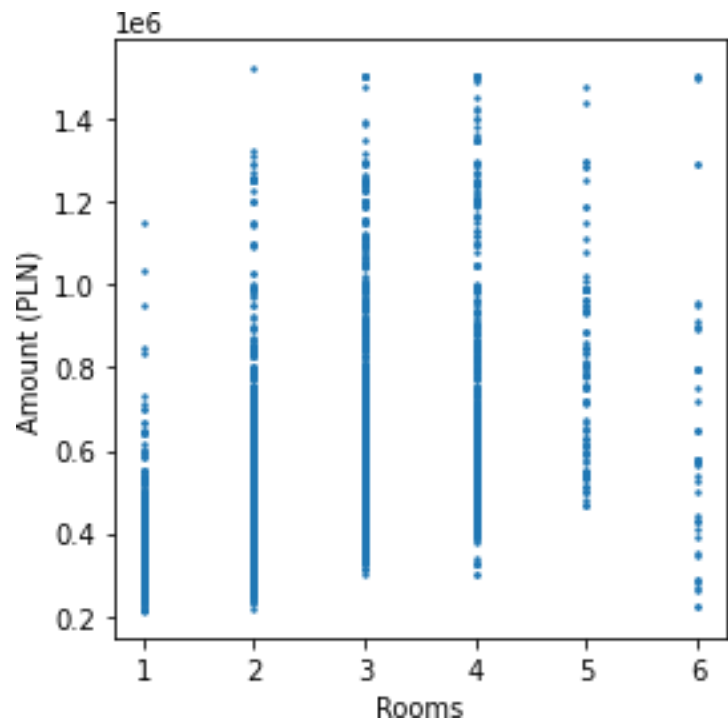
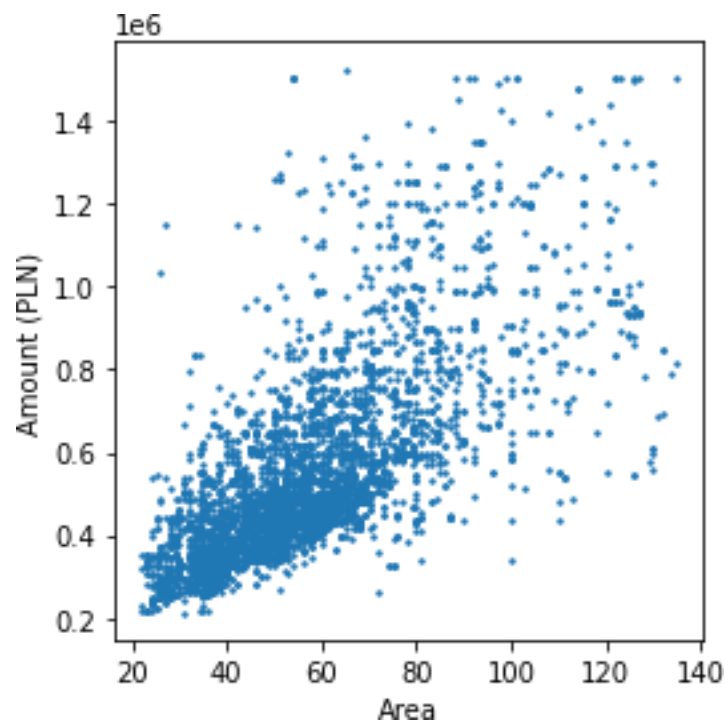
1.4 Numeric features

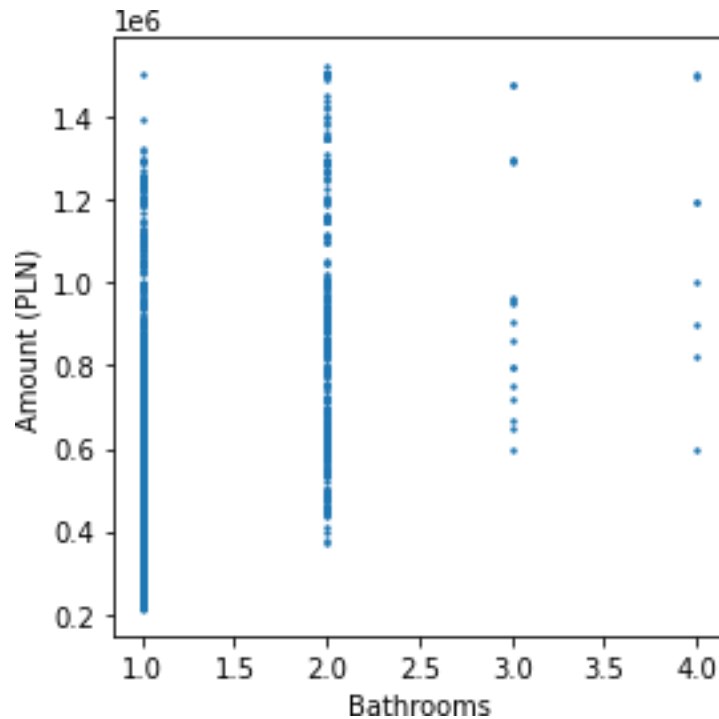
To visually inspect the data we are going to make histograms for each of the numeric columns.

```

[7]: numeric = list(data.select_dtypes('number').columns)
for col in numeric:
    if col != 'Amount':
        plt.scatter(data[col], data['Amount'], s=2)
        plt.xlabel(f'{col}')
        plt.ylabel(f'Amount (PLN)')
        plt.tight_layout()
        plt.show()

```





```
[8]: data.select_dtypes('number').corr()
```

```
[8]:
```

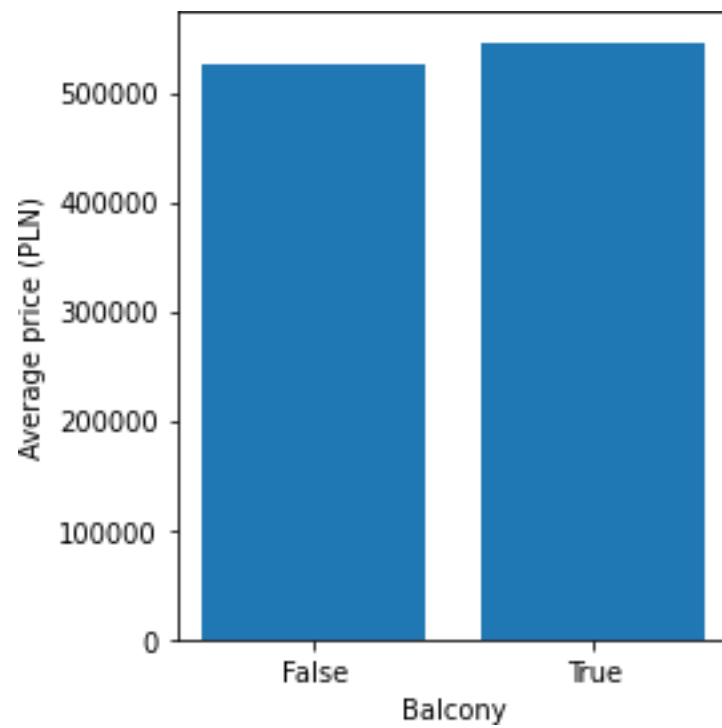
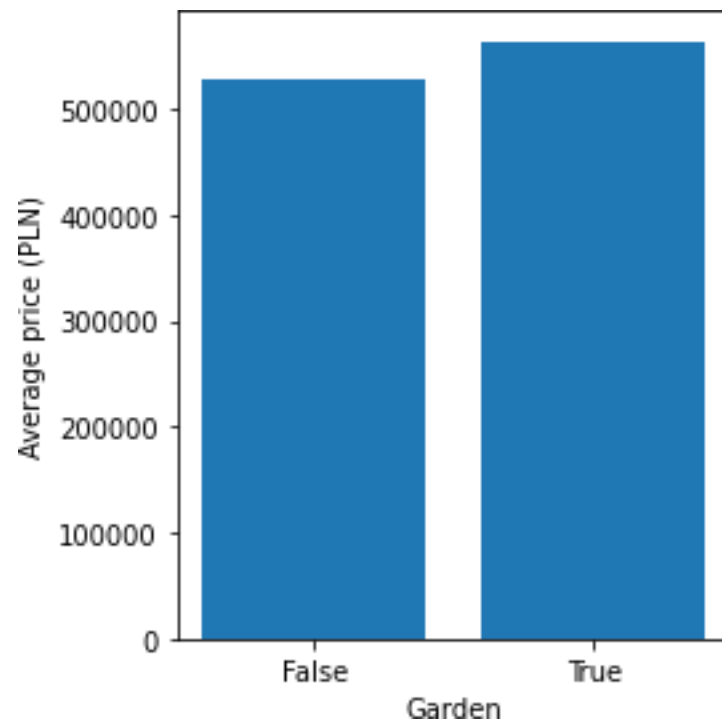
	Amount	Area	Rooms	Bathrooms
Amount	1.00	0.71	0.46	0.43
Area	0.71	1.00	0.74	0.47
Rooms	0.46	0.74	1.00	0.36
Bathrooms	0.43	0.47	0.36	1.00

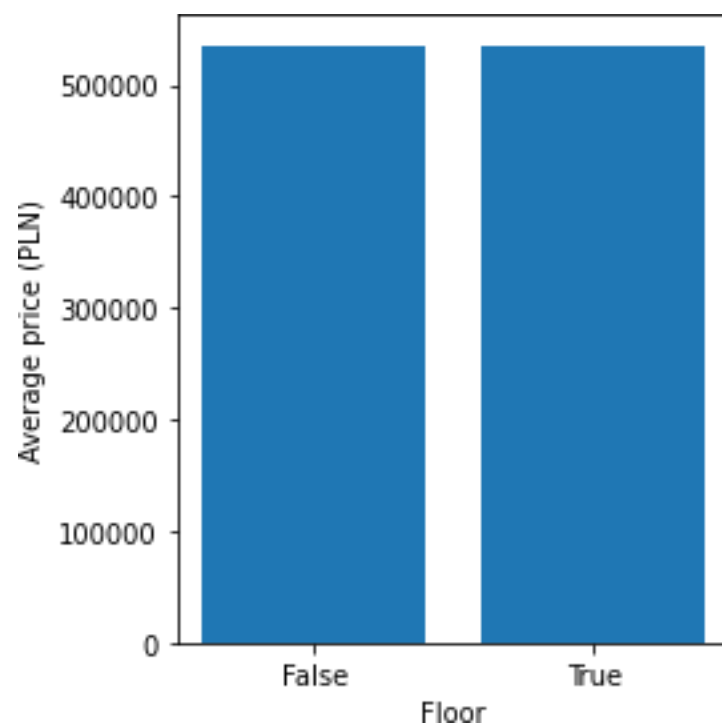
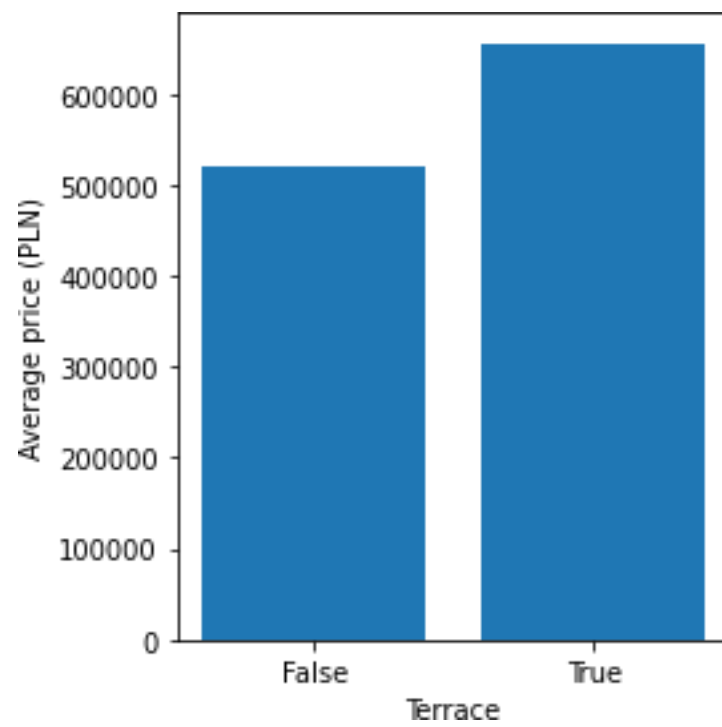
1.5 Binary features

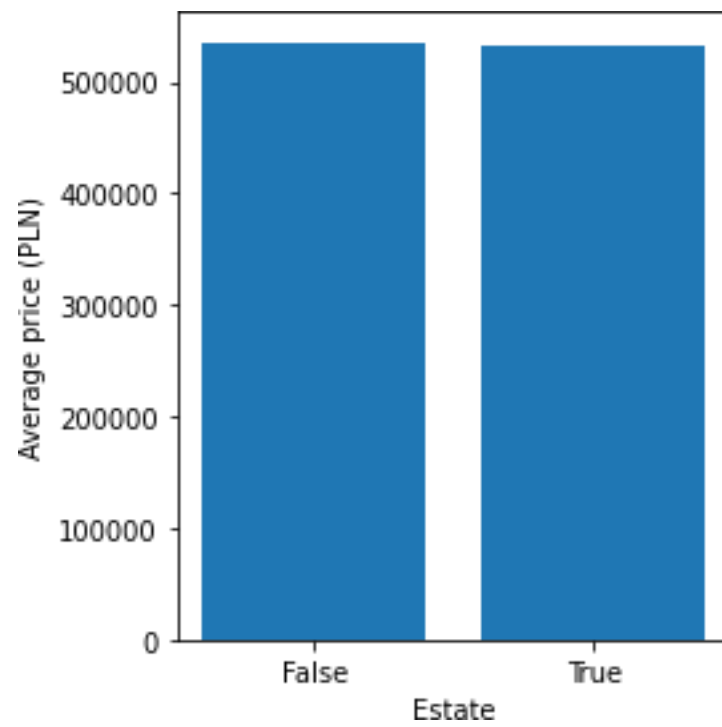
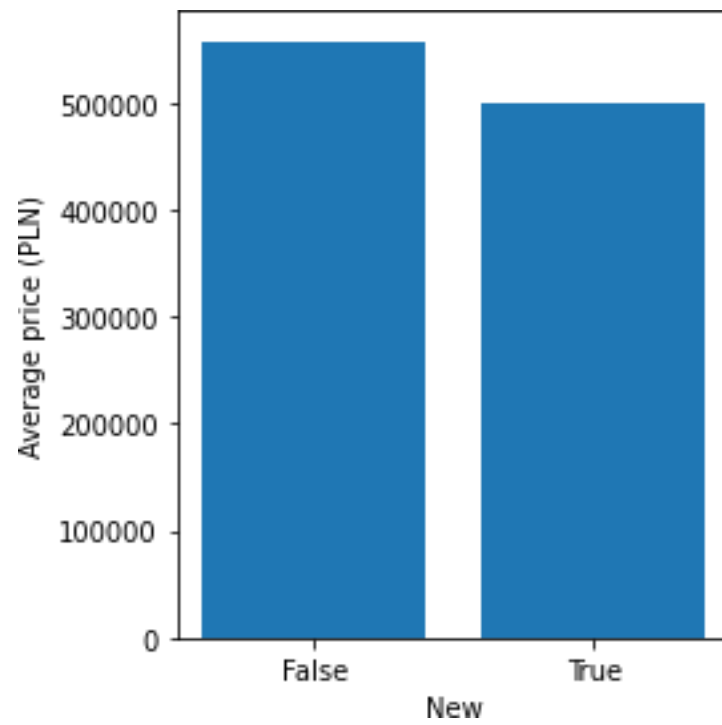
We are going to group the data and compare averages.

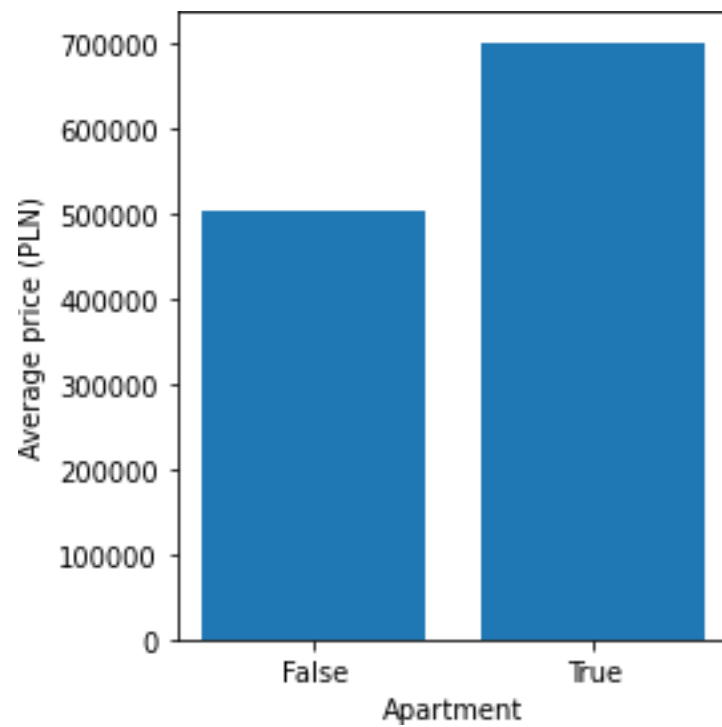
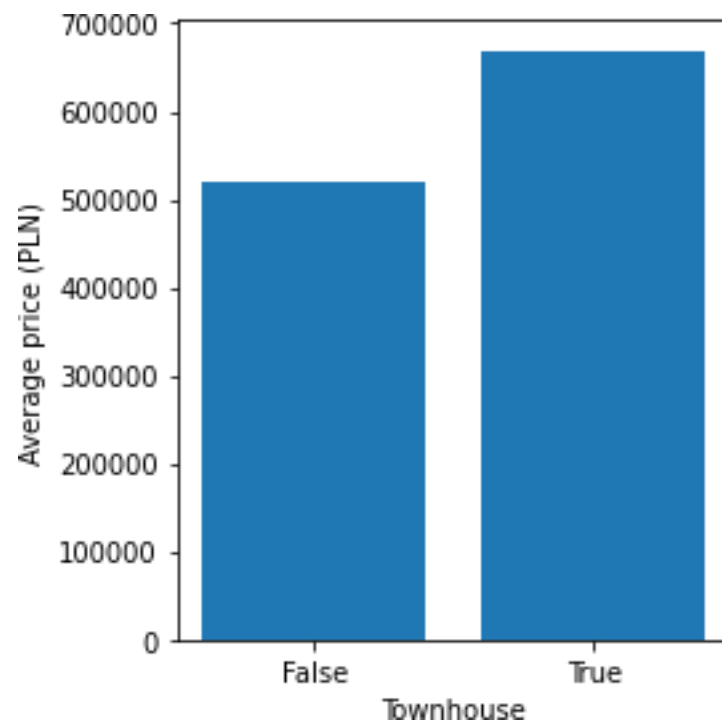
```
[9]: binary = list(data.select_dtypes(bool).columns)
for col in binary:
    group = data[[col, 'Amount']]
    group = group.groupby([col], as_index=False)
    group = group.mean()
    group = group.sort_values('Amount', ascending=False)
    plt.bar(group[col], height=group['Amount'])
    plt.xticks(ticks=group[col].to_list(),
               labels=group[col].to_list())
plt.ylabel('Average price (PLN)')
plt.xlabel(f'{col}')
```

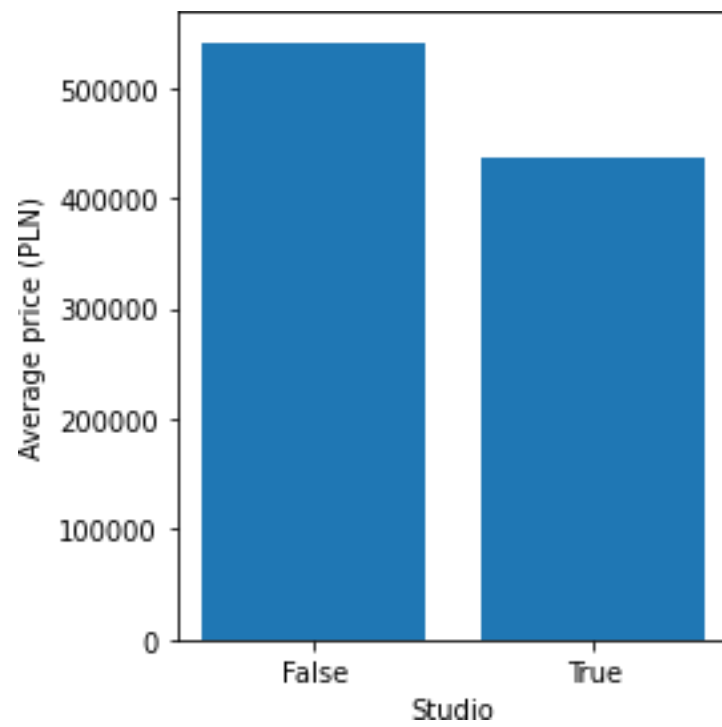
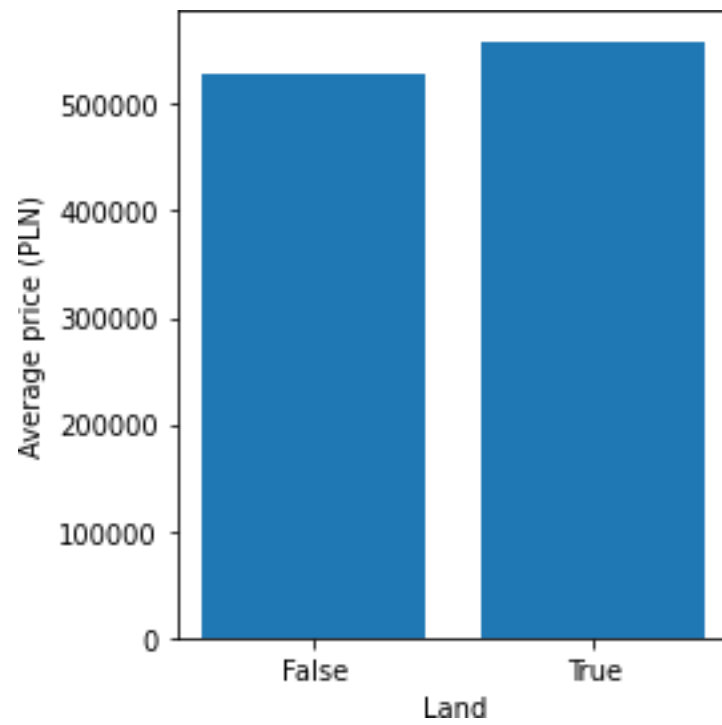
```
plt.tight_layout()  
plt.show()
```











We check correlation of binary columns with Amount.

```
[10]: C = data[binary].corrwith(data['Amount'])
      C.name = 'Correlation'
      C = C.to_frame()
      C = C.sort_values('Correlation', ascending=False)
      C = C.reset_index()
      C = C.rename(columns={'index': 'Column'})
      C
```

```
[10]:
```

	Column	Correlation
0	Apartment	0.33
1	Townhouse	0.20
2	Terrace	0.19
3	Garden	0.06
4	Land	0.06
5	Balcony	0.04
6	Floor	-0.00
7	Estate	-0.00
8	Studio	-0.11
9	New	-0.13

1.6 Categorical features

Again, we group the rows and compare averages.

```
[11]: categorical = list(data.select_dtypes('object').columns)
      for col in categorical:
          group = data[[col, 'Amount']]
          group = group.groupby([col], as_index=False)
          group = group.mean()
          group = group.sort_values('Amount', ascending=False)

          plt.bar(group[col], group['Amount'])
          plt.ylabel('Average price (PLN)')
          plt.xlabel(f'{col}')
          plt.xticks(rotation=90)

          plt.tight_layout()
          plt.savefig(f'img/feature_{col.lower()}.png')
          plt.show()
```

