Housing Market Analysis

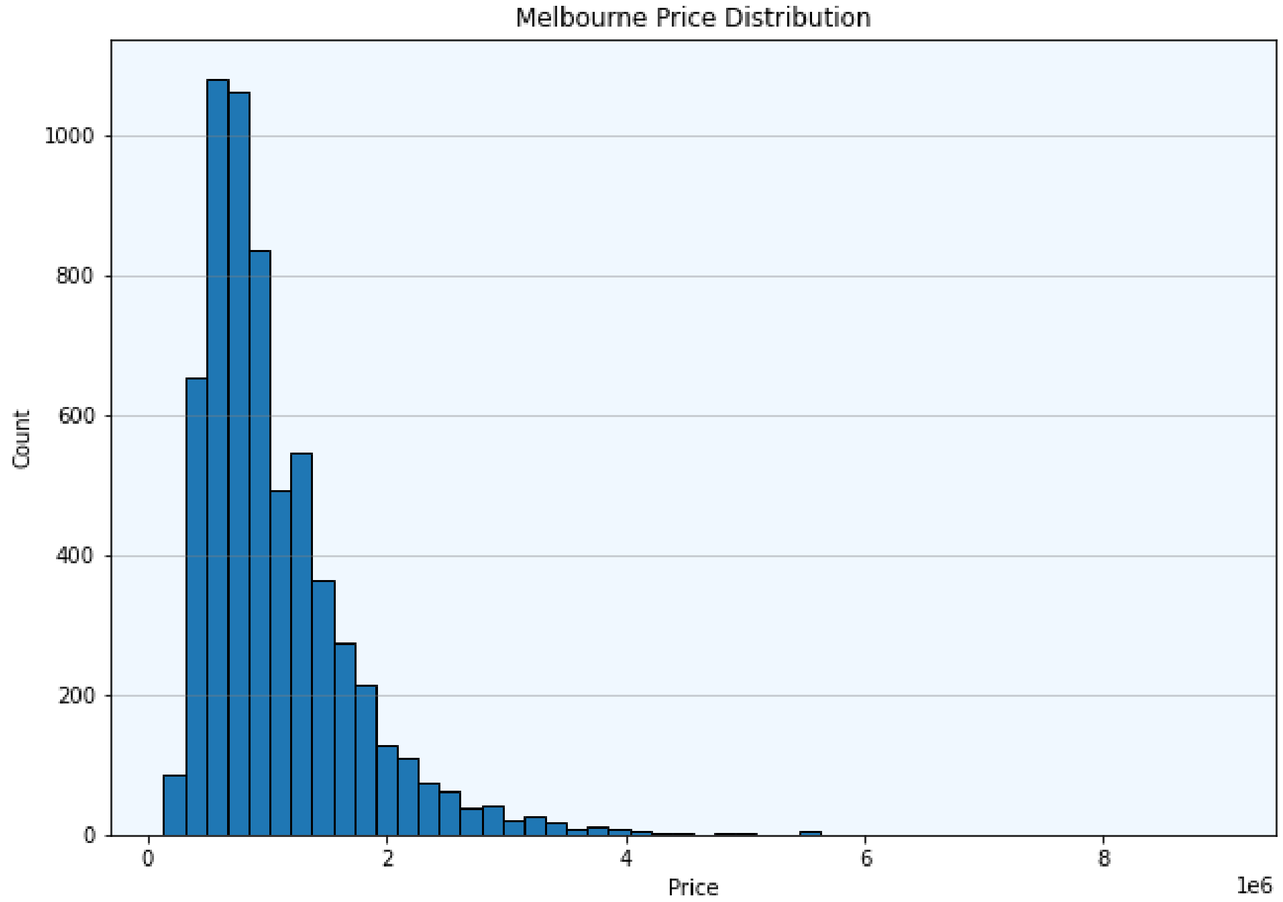
MELBOURNE, AUSTRALIA

Table of Contents

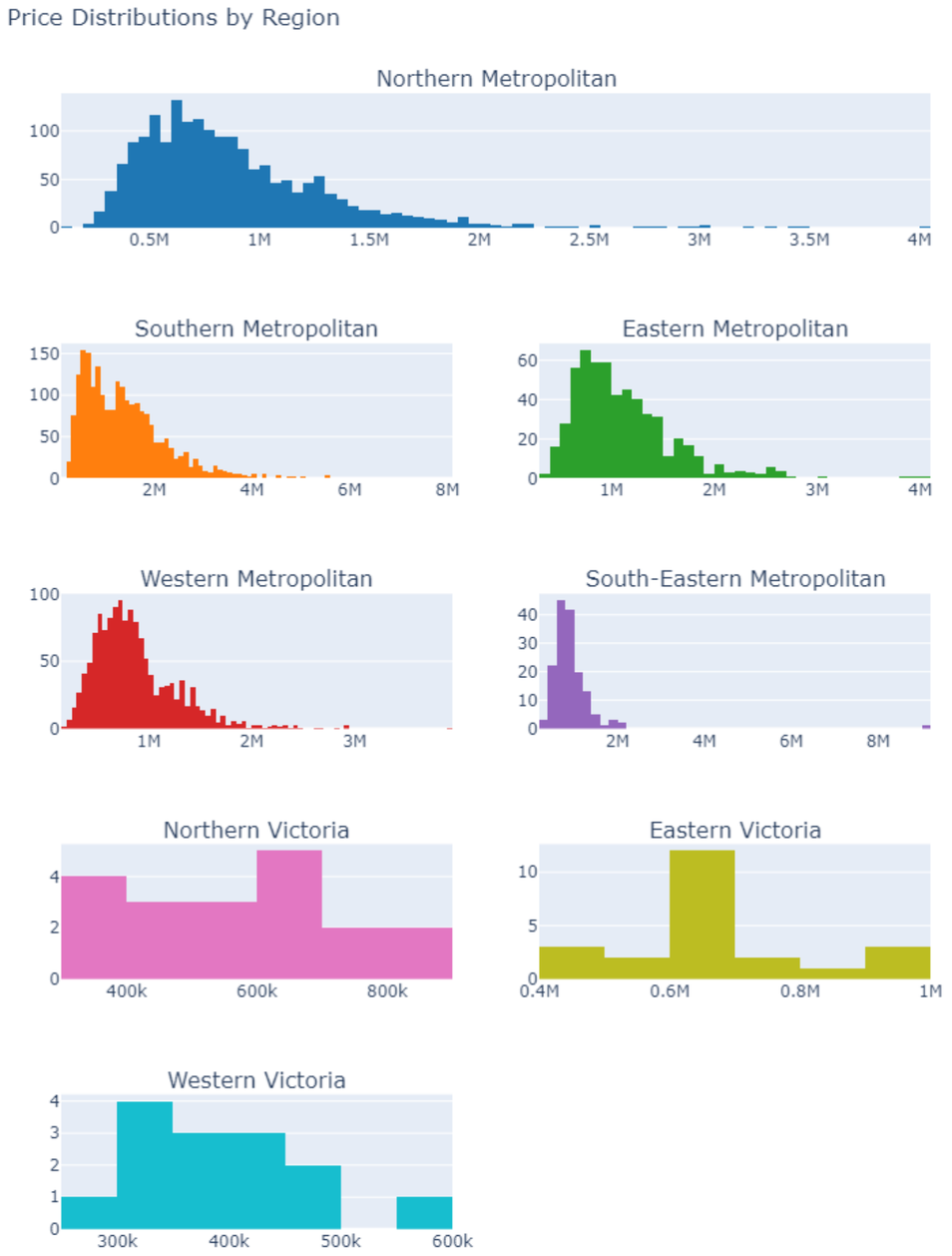
Background

- The goal is to predict the sales price for homes in Melbourne, Australia (2016-2017)
- Data consists of 13,580 data points and 19 features describing every aspect of the house.
- Our framework utilizes these Python libraries:
  - Pandas: handling structured data
  - Scikit Learn: ML
  - NumPy: linear algebra and mathematics
  - Seaborn: Data visualization
  - Plotly, Matplotlib: Data visualization
  - Javascript: Data visualization

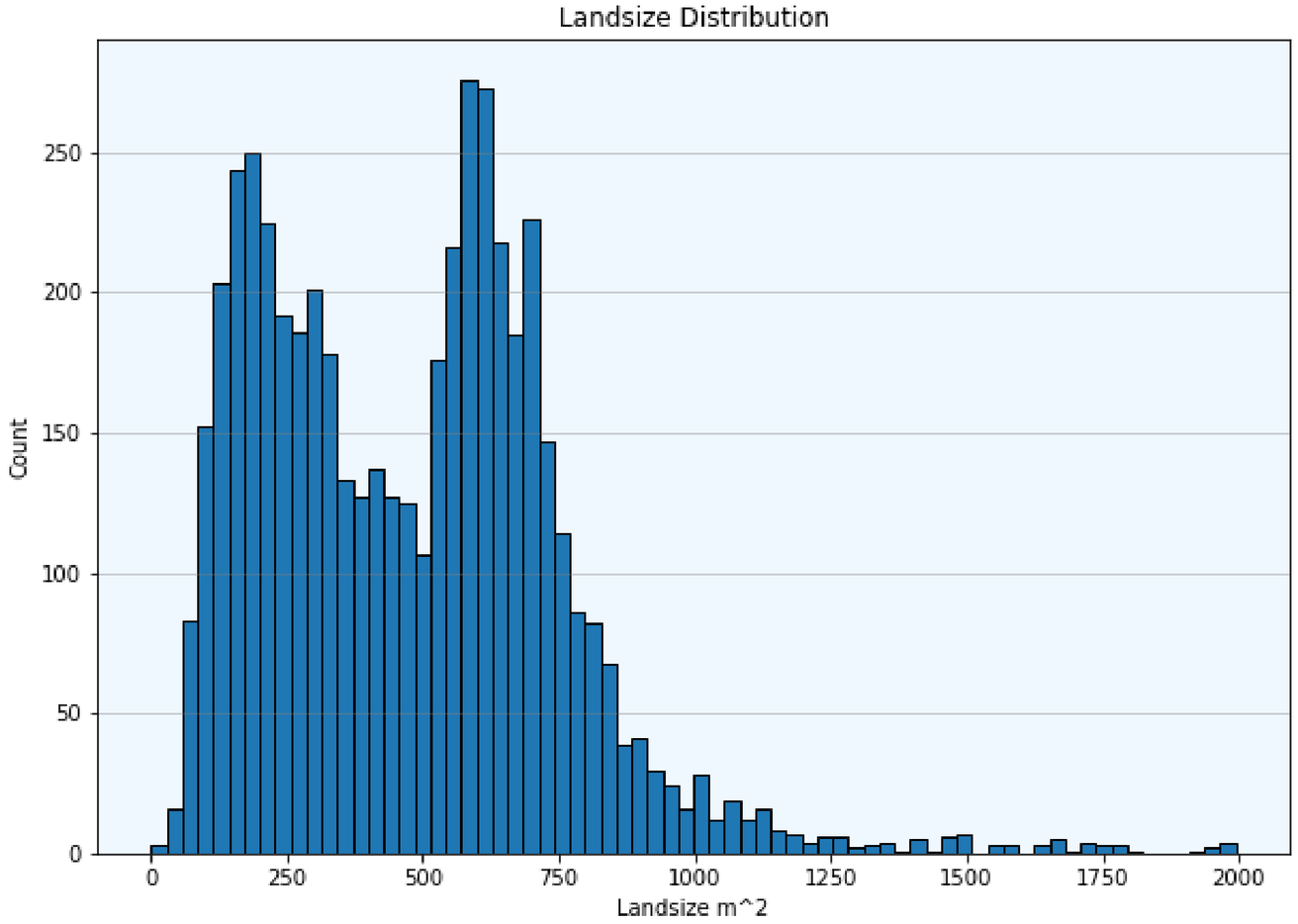
Exploratory Data Analysis



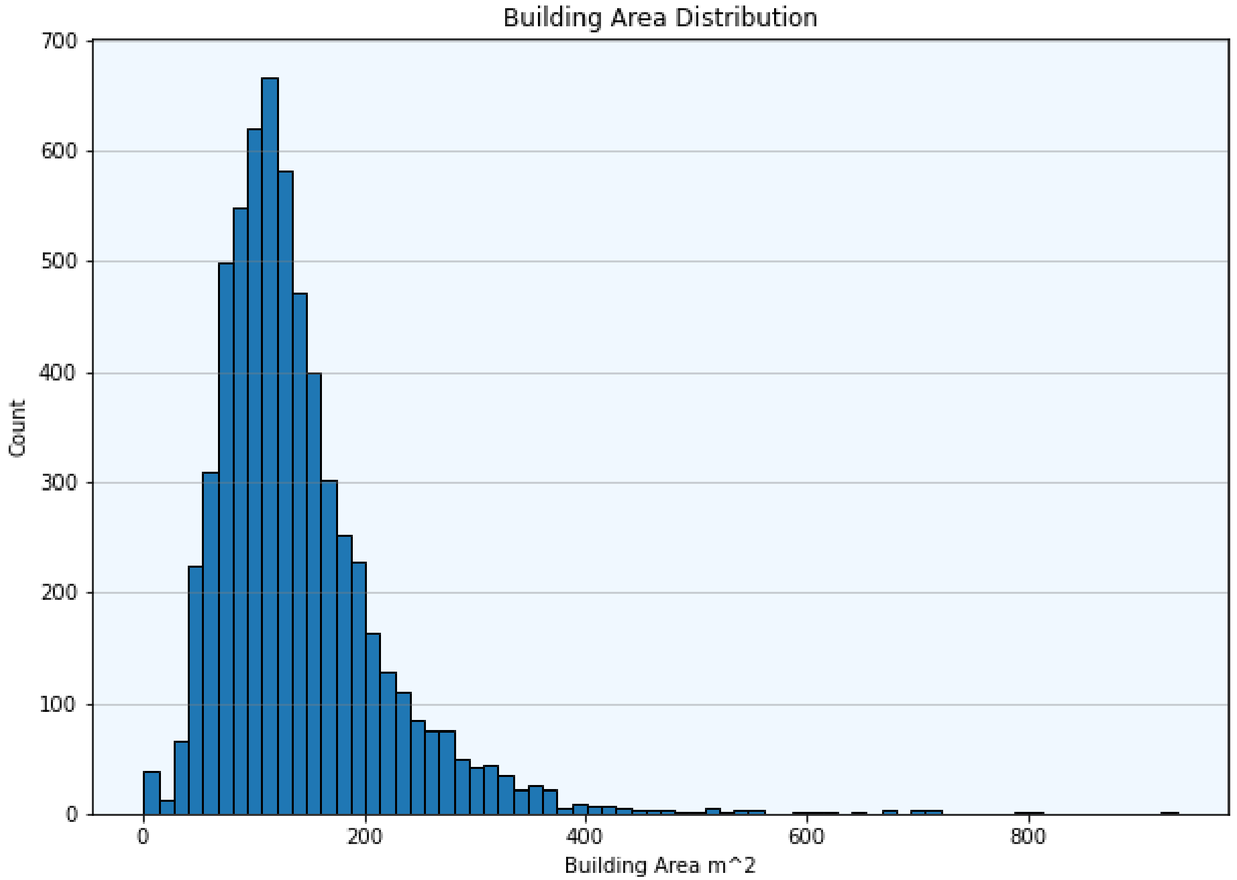
Exploratory Data Analysis



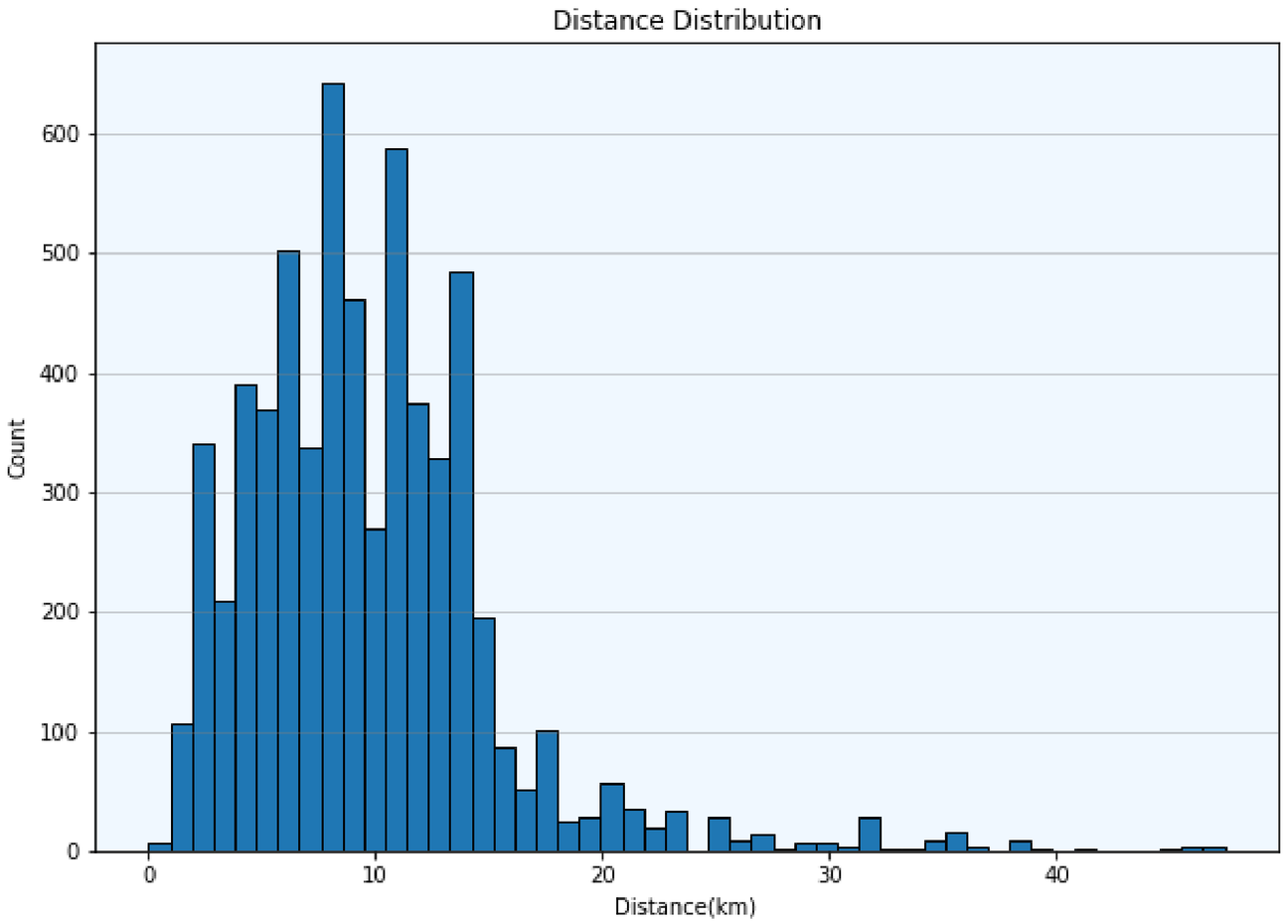
Exploratory Data Analysis



Exploratory Data Analysis



Exploratory Data Analysis



Clusters

In [6]:
```
method_encoded = pd.get_dummies(df['Method'])
suburb_encoded = pd.get_dummies(df['Suburb'])
type_encoded = pd.get_dummies(df['Type'])
seller_encoded = pd.get_dummies(df['SellerG'])
zip_encoded = pd.get_dummies(df['Postcode'])
region_encoded = pd.get_dummies(df['Regionname'])
```

In [7]:
```
df_scaled_encoded = pd.concat([df_scaled, method_encoded, type_encoded, regio

df_scaled_encoded
```

Out[7]:

	Price	Rooms	Distance	Bedroom2	Bathroom	Car	Landsize	BuildingArea
0	-0.050108	-0.959224	-1.292159	-0.929954	-0.810257	-1.692272	-0.351031	-0.688873
1	0.586833	0.070641	-1.292159	0.100999	0.595611	-1.692272	-0.375546	0.092828
2	0.786802	1.100507	-1.292159	0.100999	-0.810257	0.458562	-0.391147	0.004749
3	1.195630	0.070641	-1.292159	1.131952	0.595611	-1.692272	-0.251853	0.753421
4	0.840128	-0.959224	-1.292159	-0.929954	-0.810257	0.458562	-0.239595	-0.380597

In [4]:
```
# Use the StandardScaler module and fit_transform function to
# scale all columns with numerical values
df_scaled = StandardScaler().fit_transform(df[["Price", "Rooms" , "Distance", "B

# Diplay the first three rows of the scaled data
df_scaled[0:3]
```

Out[4]:
```
array([[-0.05010829, -0.95922445, -1.29215935, -0.92995401, -0.8102572 ,
        -1.69227228, -0.35103059, -0.6888734 , -1.68182726],
       [ 0.58683256,  0.0706412 , -1.29215935,  0.10099876,  0.59561051,
        -1.69227228, -0.37554647,  0.09282823, -1.68182726],
       [ 0.78680236,  1.10050686, -1.29215935,  0.10099876, -0.8102572 ,
         0.4585624 , -0.39114749,  0.00474917,  1.31009469]])
```

In [5]:
```
df_scaled = pd.DataFrame(
    df_scaled,
    columns=["Price", "Rooms" , "Distance", "Bedroom2", "Bathroom", "Car", "Land
)
df_scaled
```

Out[5]:

	Price	Rooms	Distance	Bedroom2	Bathroom	Car	Landsize	BuildingArea	Ye
0	-0.050108	-0.959224	-1.292159	-0.929954	-0.810257	-1.692272	-0.351031	-0.688873	-1.
1	0.586833	0.070641	-1.292159	0.100999	0.595611	-1.692272	-0.375546	0.092828	-1.
2	0.786802	1.100507	-1.292159	0.100999	-0.810257	0.458562	-0.391147	0.004749	1.
3	1.195630	0.070641	-1.292159	1.131952	0.595611	-1.692272	-0.251853	0.753421	-1.
4	0.840128	-0.959224	-1.292159	-0.929954	-0.810257	0.458562	-0.239595	-0.380597	-1.

Clusters

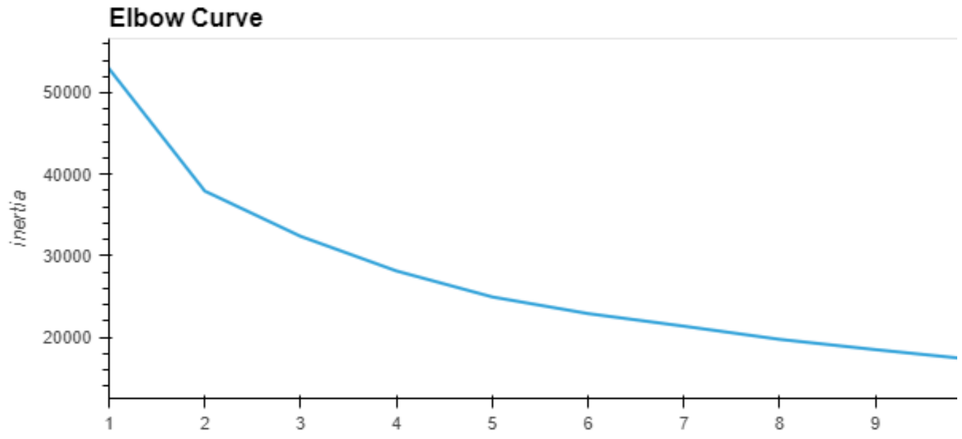
Out[22]:

	k	inertia
0	1	52963.009877
1	2	37938.148518
2	3	32379.945373
3	4	28131.521577
4	5	24942.215790

WITHOUT PCA

In [23]:
```
# Plot a line chart with all the inertia values computed with
# the different values of k to visually identify the optimal value for k.
elbow = df_elbow.hvplot.line(
    x="k",
    y="inertia",
    title="Elbow Curve",
    xticks=k
)
elbow
```

Out[23]:



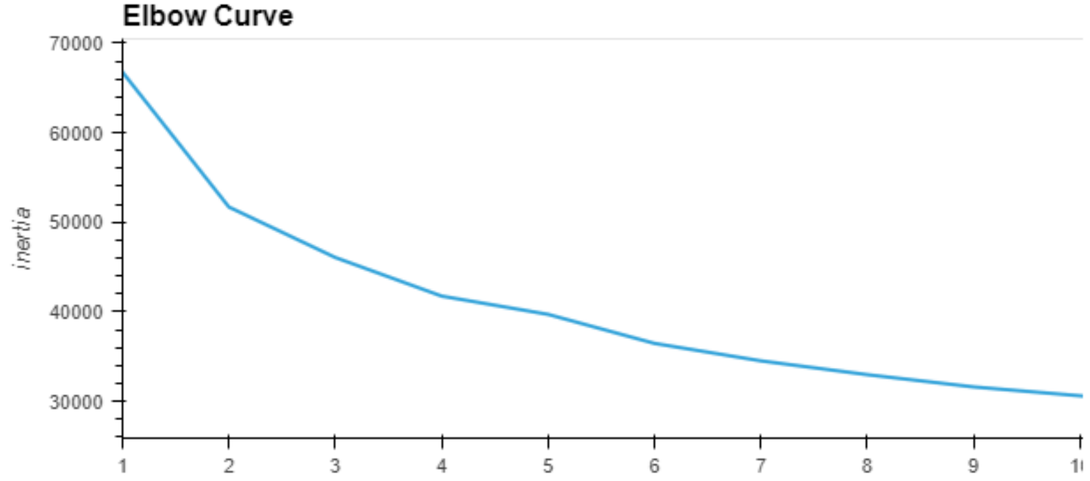
Out[12]:

	k	inertia
0	1	66751.236927
1	2	51698.889556
2	3	46068.421030
3	4	41758.325041
4	5	39719.331794

WITH PCA

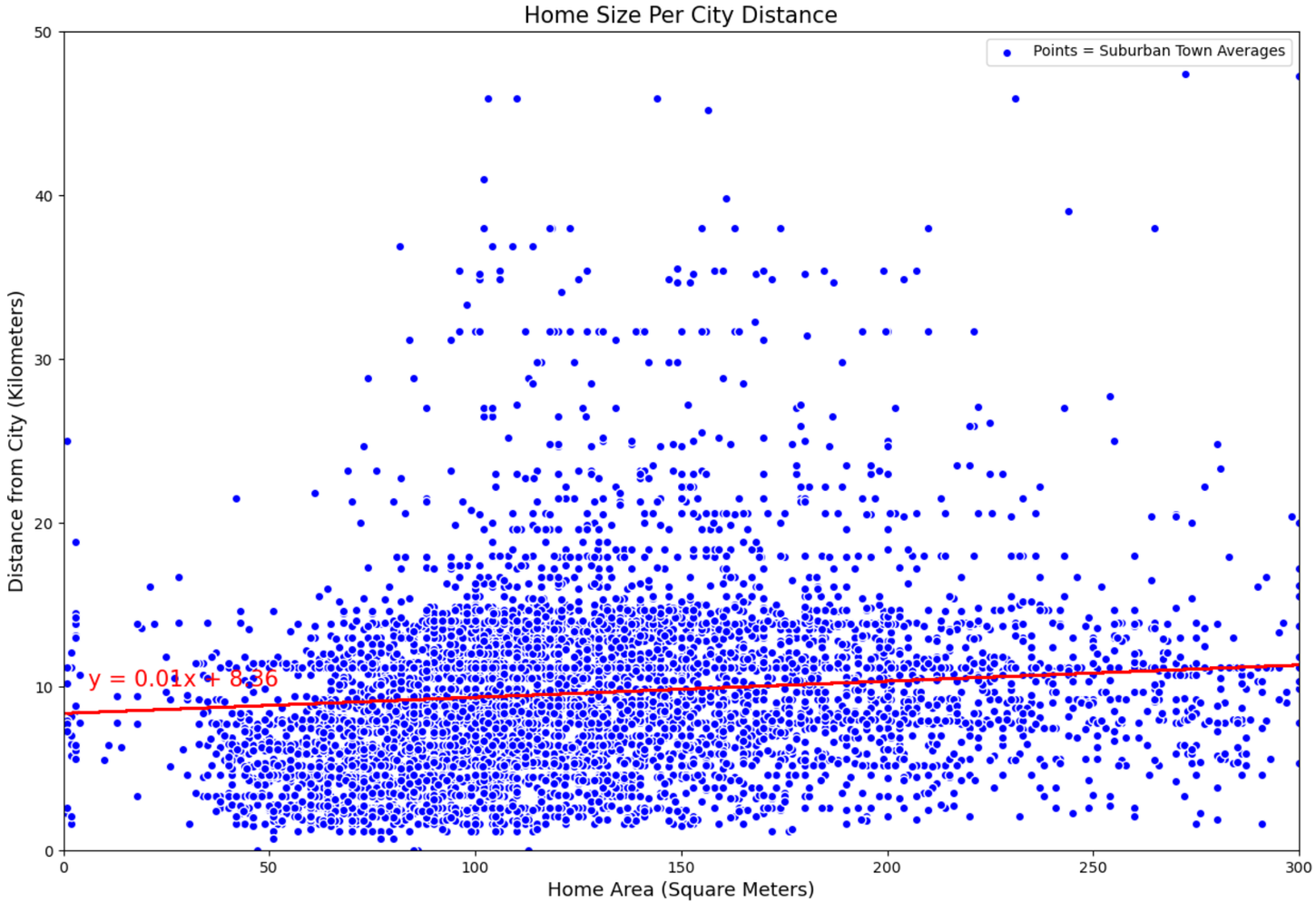
In [13]:
```
# Plot a line chart with all the inertia values computed with
# the different values of k to visually identify the optimal value for k.
elbow = df_elbow.hvplot.line(
    x="k",
    y="inertia",
    title="Elbow Curve",
    xticks=k
)
elbow
```

Out[13]:

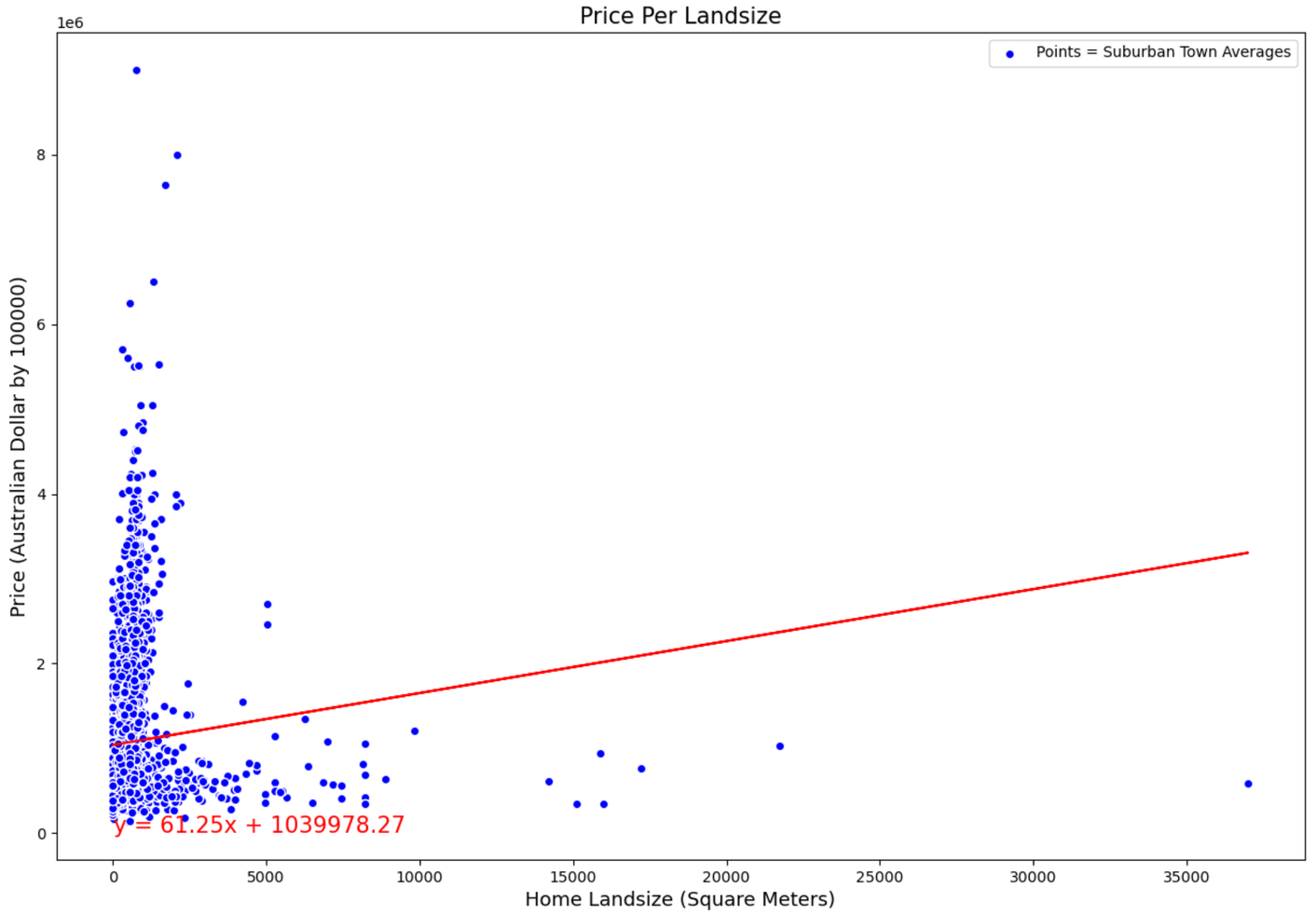


LINK

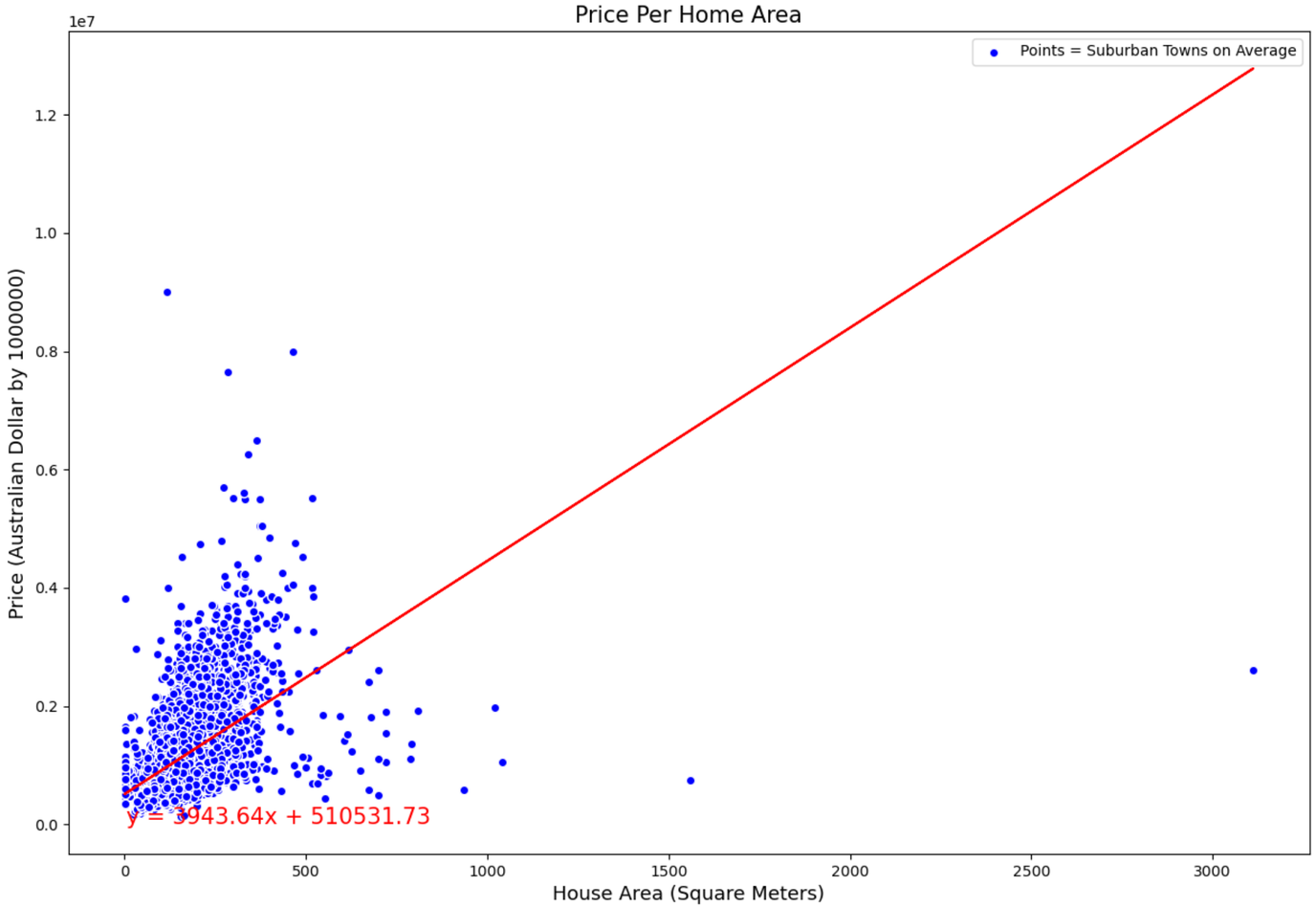
Correlation



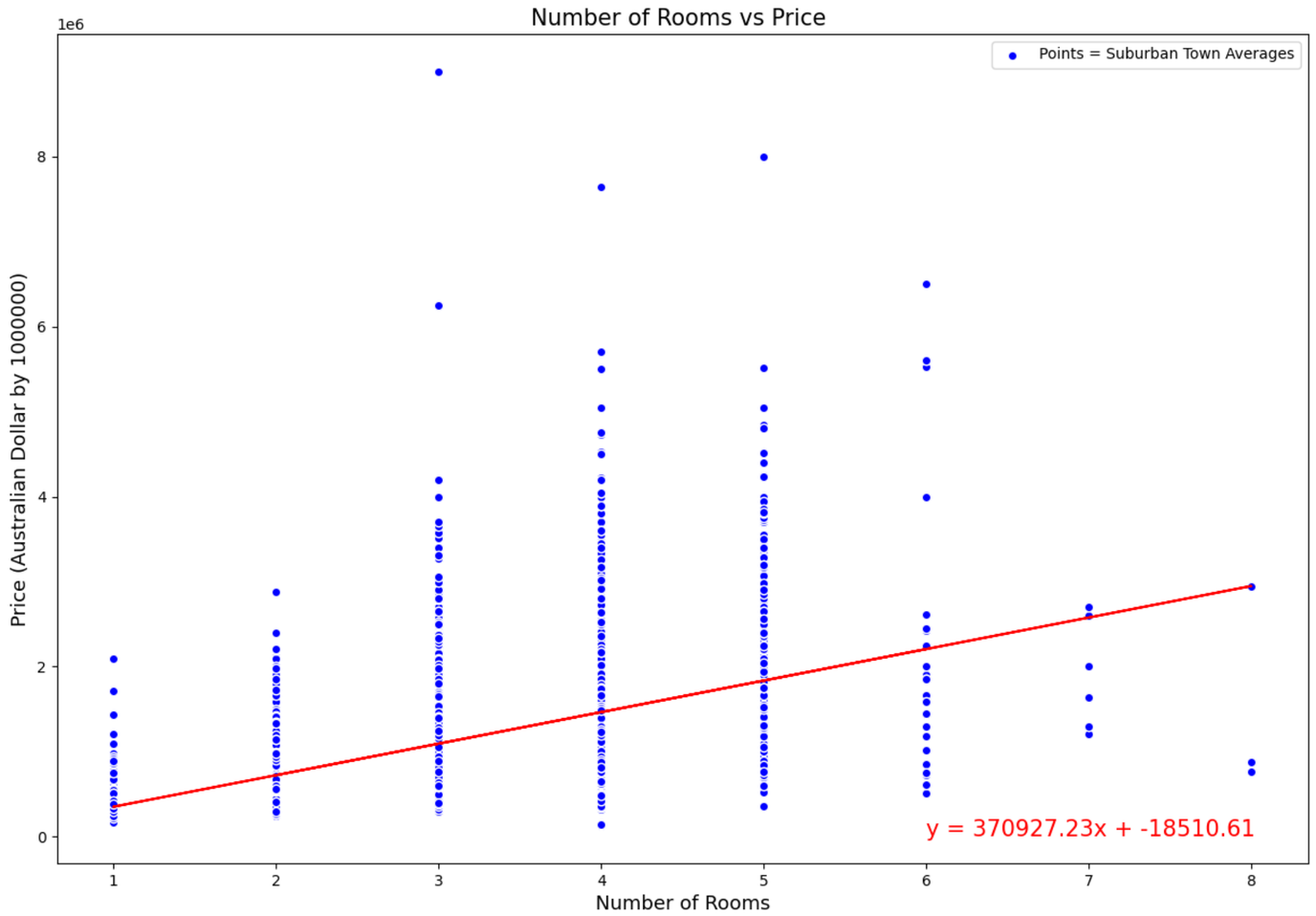
Correlation



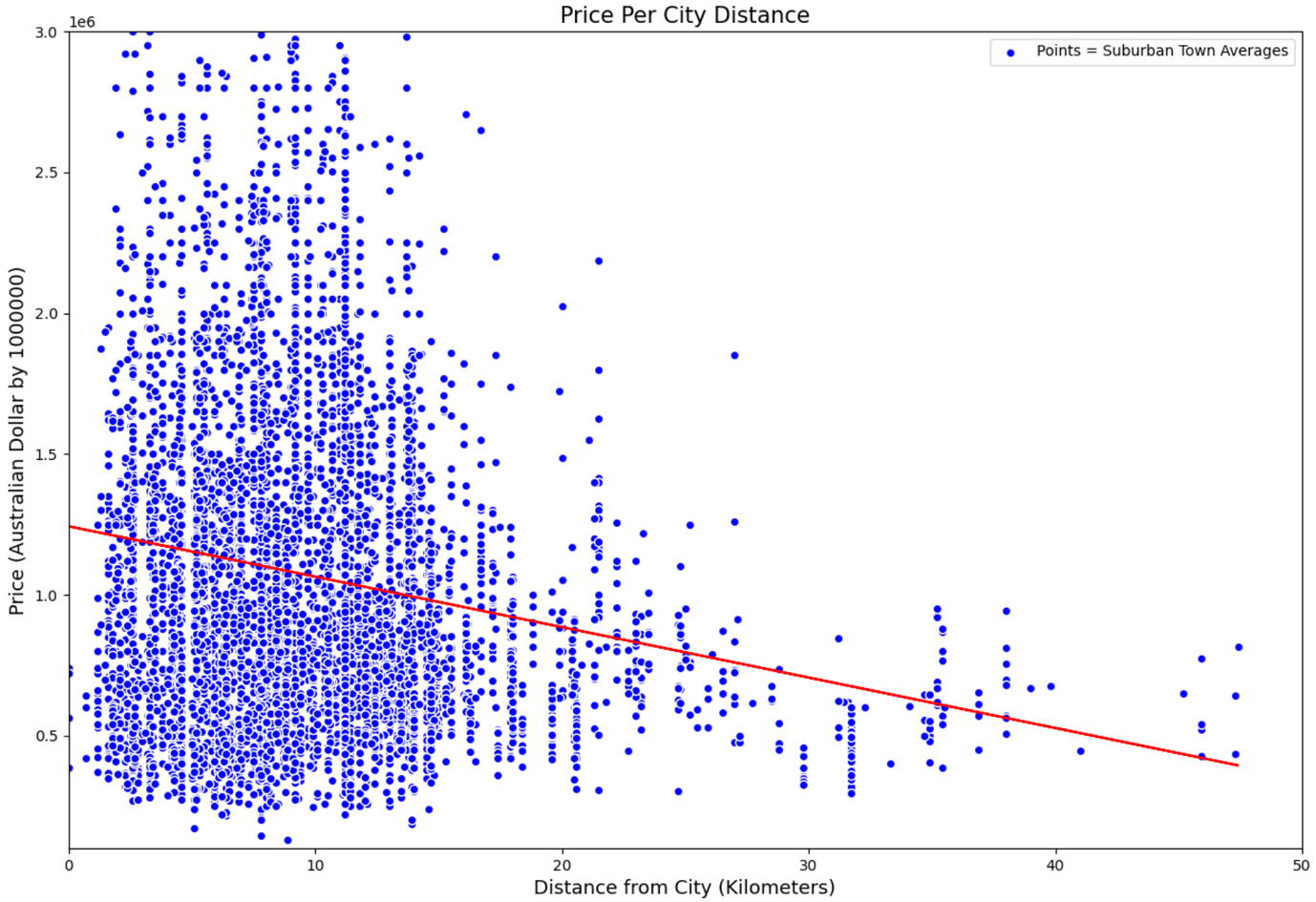
Correlation



Correlation

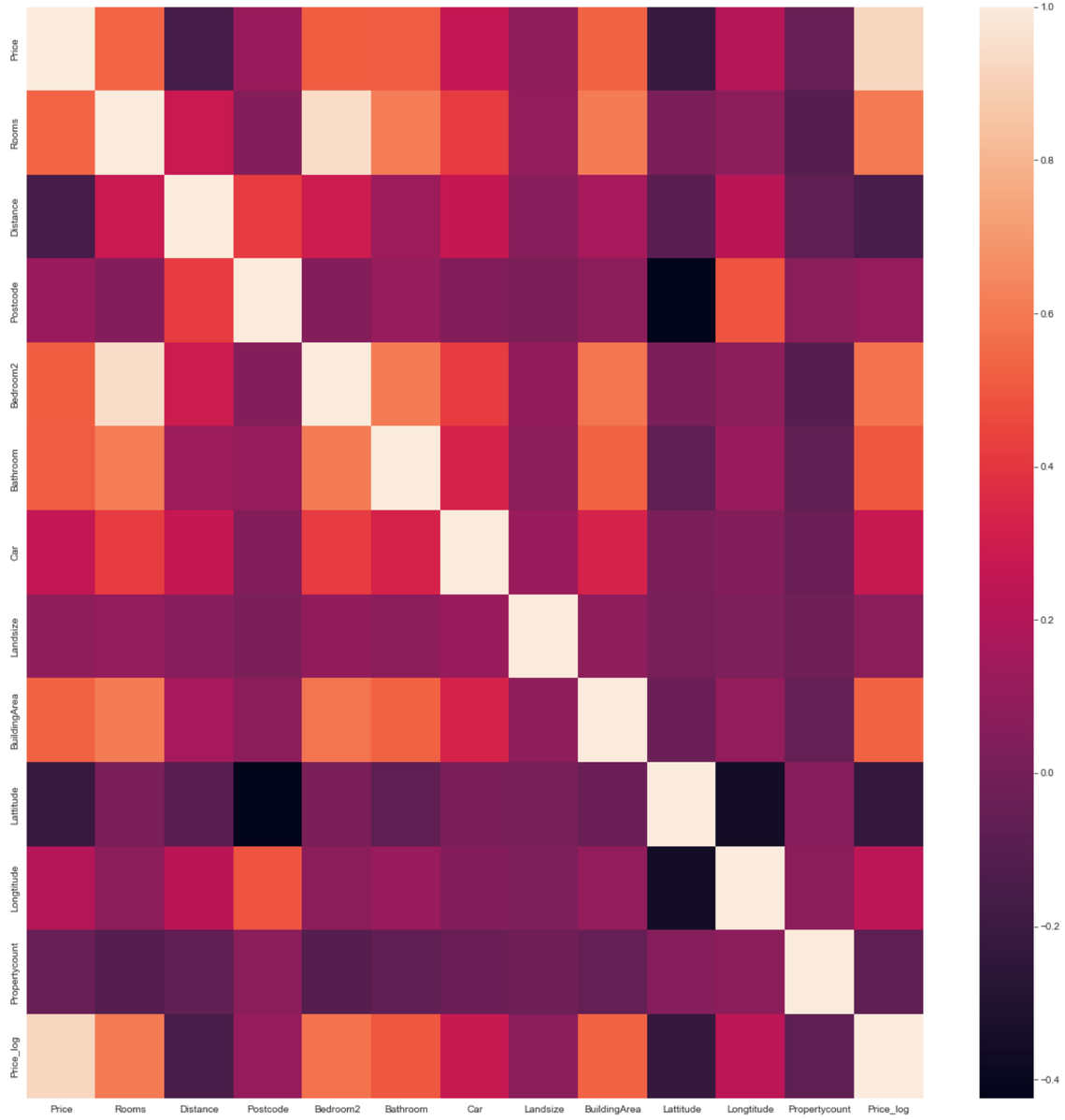


Correlation

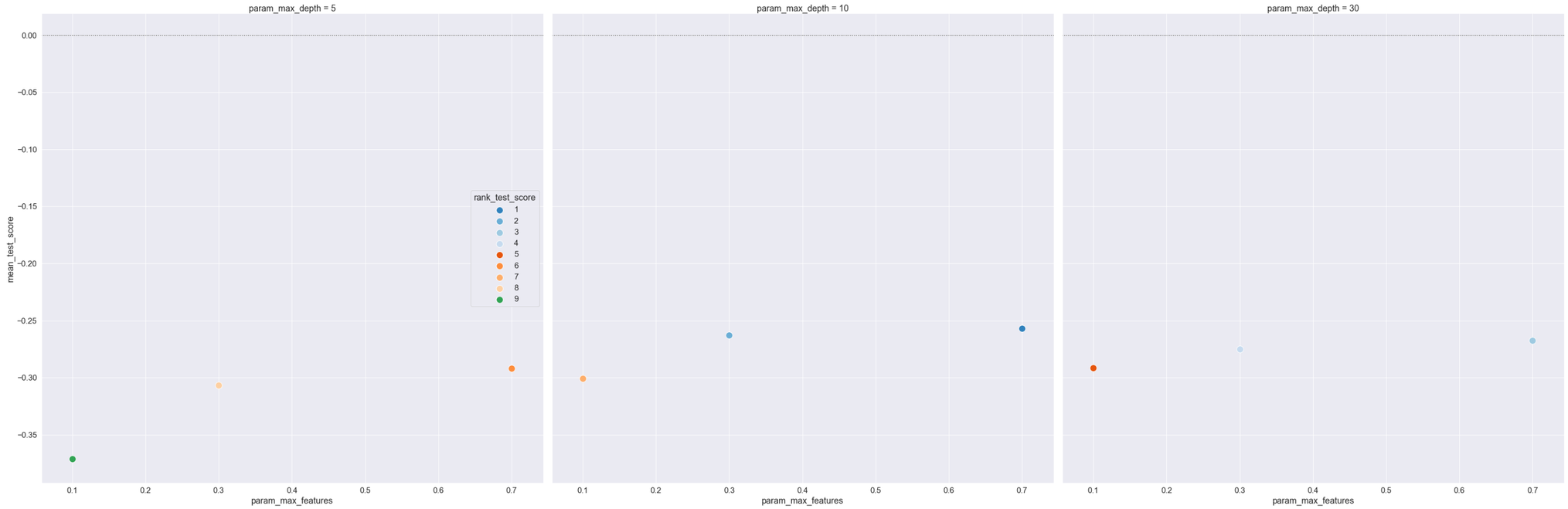


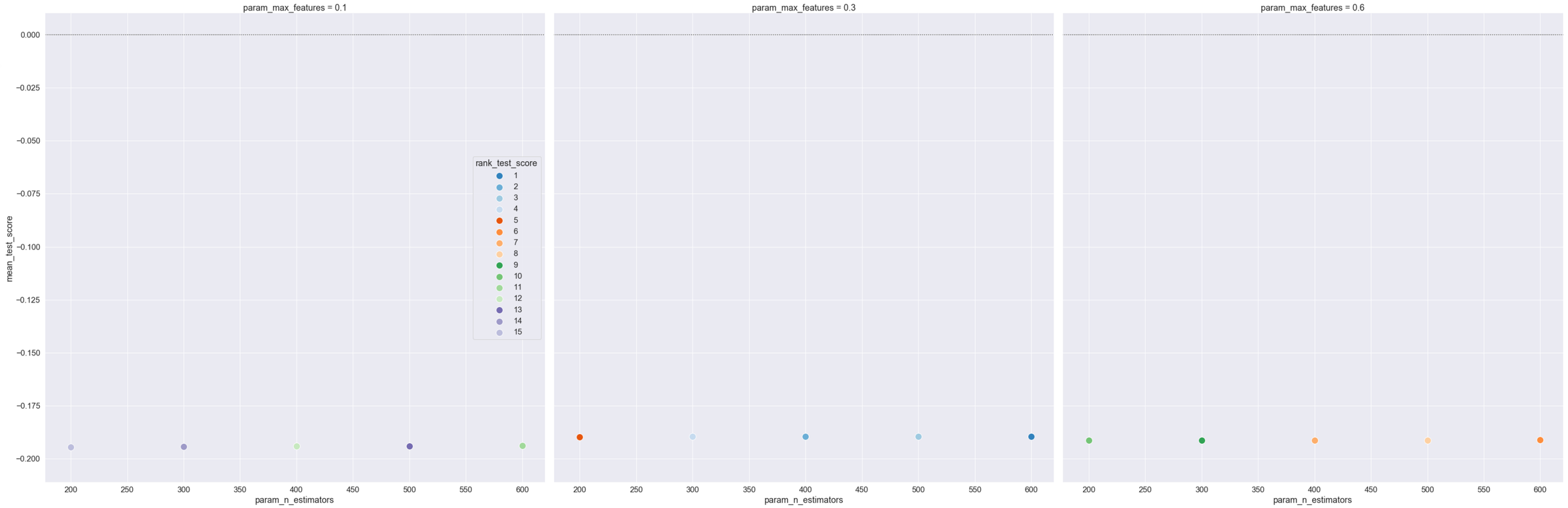
Correlation Matrix

Presentations are communication tools that can be used as demonstrations, lectures, speeches, reports, and more. It is mostly presented before an audience.



Correlation Matrix




param_max_features = 0.1
param_max_features = 0.3
param_max_features = 0.6
mean_test_score
0.000
−0.025
−0.050
−0.075
−0.100
−0.125
−0.150
−0.175
−0.200
200
250
300
350
400
450
500
550
600
param_n_estimators
rank_test_score
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15


Thank You