AI ASSIGNMENT 1

Name: Prateek Bajpai

Class: Data Science & Big Data Analytics

1. Write a python Program to display the last digit of number(hint: Any number divide by 10 will return last digit)

```
# Program to display last digit of number.

number = int(input("Enter a number: "))
last_digit = number % 10

print(f"Last digit of the given number {number} is: {last_digit}")
```

```
Enter a number: 96
Last digit of the given number 96 is: 6
```

1. Write a function to Calculate Area and Perimeter of Rectangle

```
# Function to calculate area and perimeter of rectangle.

def area(l, b):
  return(l*b)

def perimeter(l,b):
  return(2*(l+b))


l = int(input("Length of rectangle = "))
b = int(input("Breadth of rectangle = "))

print("Area of rectangle = ", area(l,b))
print("Perimeter of rectangle = ", perimeter(l,b))
```

```
Length of rectangle = 6
Breadth of rectangle = 7
Area of rectangle =  42
Perimeter of rectangle =  26
```

1. Write a python Program to find out given No is Prime or Not.

```
# Program to find prime number

num = int(input("Enter a number: "))

if num == 1:
    print(num, "is not a prime number")
elif num > 1:
   # check for factors
   for i in range(2,num):
```

```
        if (num % i) == 0:
            print(num,"is not a prime number")
            print(i,"times",num//i,"is",num)
            break
    else:
        print(num,"is a prime number")


else:
    print(num,"is not a prime number")
```

```
Enter a number: 23
23 is a prime number
```

1. Write a Python program that prompts the user to input three numbers and then determines and prints the largest among them.

```
# Program to find largest of 3 numbers

num1 = float(input("Enter 1st number: "))
num2 = float(input("Enter 2nd number: "))
num3 = float(input("Enter 3rd number: "))

if (num1 >= num2) and (num1>=3):
   largest = num1

elif (num2 >= num1) and (num2 >= num3):
   largest = num2

else:
   largest = num3

print("Largest from the given number = ", largest)
```

```
Enter 1st number: 23
Enter 2nd number: 23.1
Enter 3rd number: 20
Largest from the given number =  23.1
```

1. Write a Python program to calculate electricity bill (accept no of unit from user) according to following criteria: Unit Price First 100 units no charge Next 100 units Rs 5 per Unit After 200 Units Rs 10 per Unit (For Example if input unit is 350 then total bill amount is 2000)

```
# Program to calculate electricity bill.

units = float(input("Please enter electricity unit consumed = "))

def electricityBill(units):

   if (units <= 100):
```

```
    return (units * 0);

  elif (units <= 200):
    return ((100 * 0) + (units - 100) * 5);

  elif (units > 200):
    return ((100 * 0) + (100 * 5) + (units - 200) * 10);

  return 0;

print("Your electricity bill = Rupees ", electricityBill(units))
```

```
Please enter electricity unit consumed = 350
Your electricity bill = Rupees  2000.0
```

1. Write a program to accept numbers from 1-7 and display name of day like- 1 for sunday, 2 for Monday and so on.

```
# Program to display dat of week.

day = int(input("Enter a number from 1 to 7: "))

if day == 1:
  print("1... Its Sunday :)")

elif day == 2:
  print("2... Its Monday :-(")

elif day == 3:
  print("3... Its Tuesday :/")

elif day == 4:
  print("4... Its Wednesday :'(")

elif day == 5:
  print("5... Its Thursday :|")

elif day == 6:
  print("6... Its Friday!!!")

elif day == 7:
  print("7... Its Saturday :~)")

else:
  print("Number is out of week's range.")
```

```
Enter a number from 1 to 7: 4
4... Its Wednesday :'(
```

1. Write a python program to check given word is Palindrome or not

```
# Program to check a Palindrome word

word = input("Enter a word: ")

rev_word = reversed(word)

if list(rev_word) == list(word):
  print("Its a Palindrome")
else:
  print("Given word is not a Palindrome")
```

```
Enter a word: level
Its a Palindrome
```

1. Download the Dataset- https://www.kaggle.com/datasets/mohamedafsal007/house-price-dataset-of-india Load the dataset.

```
# Loading the dataset

import pandas as pd
import numpy as np

from google.colab import files
uploaded = files.upload()
```

```
<IPython.core.display.HTML object>
```

```
Saving House Price India.csv to House Price India.csv
```

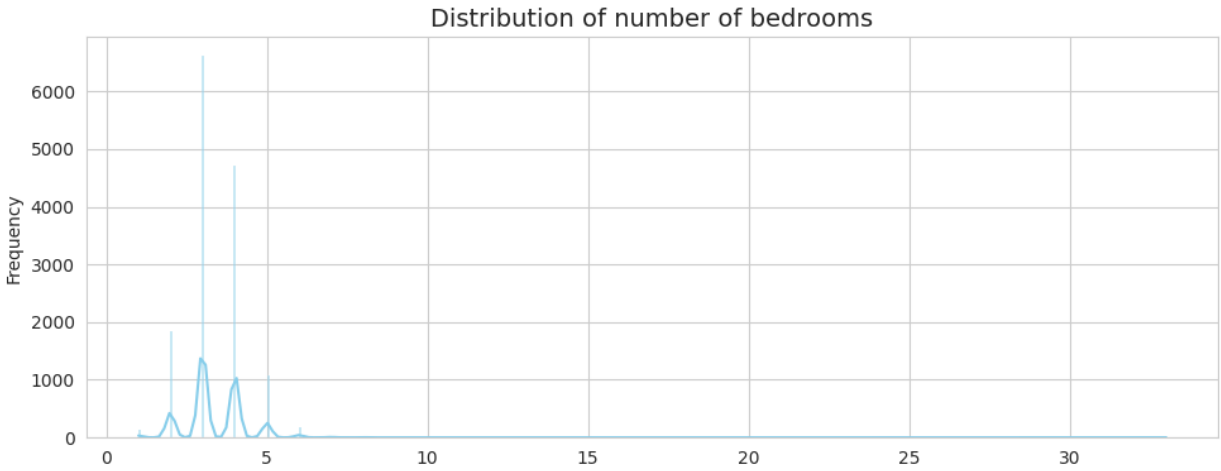
```
# Reading the dataset

df = pd.read_csv('House Price India.csv')

df.head()
```

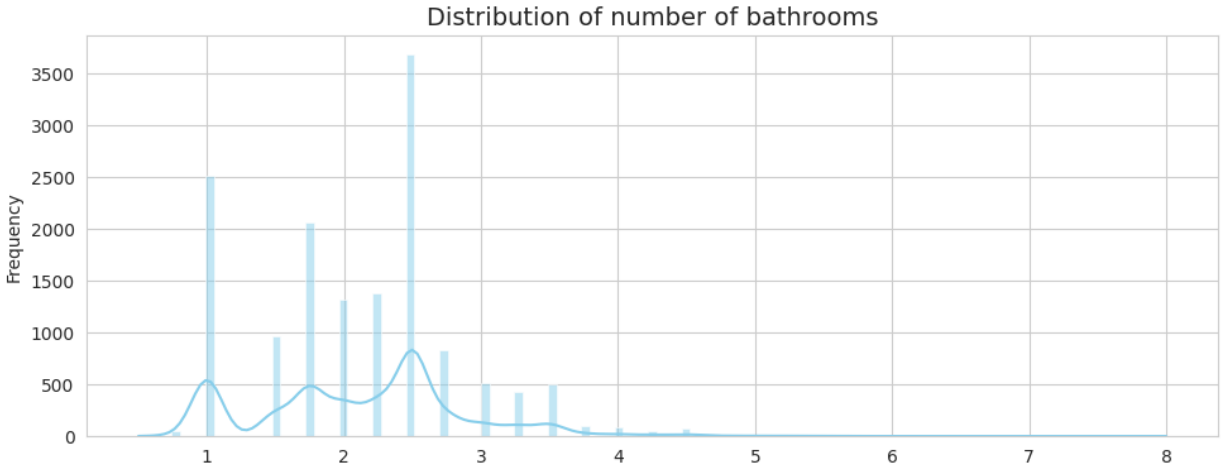
```
{"type":"dataframe","variable_name":"df"}
```

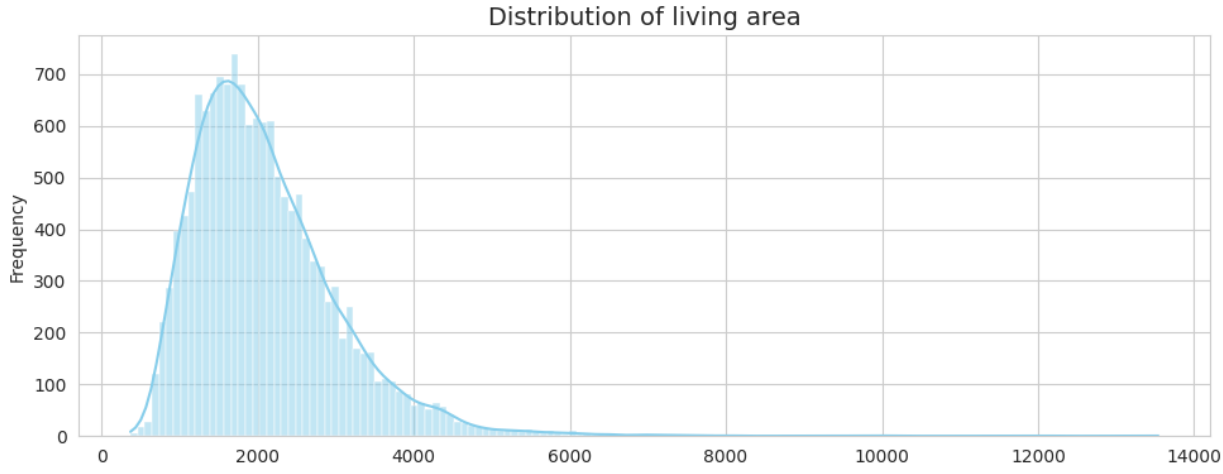
1. Perform Univariate Analysis

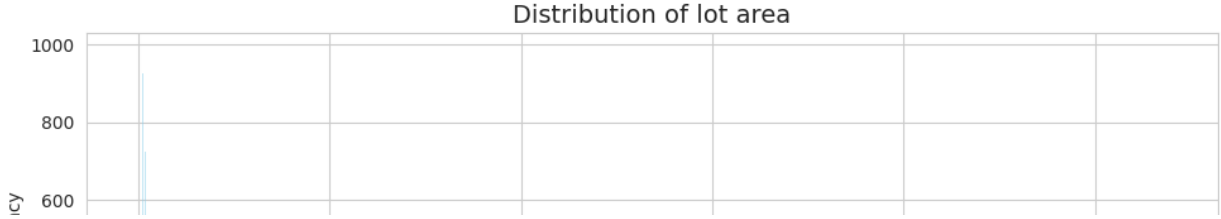
```
import matplotlib.pyplot as plt
import seaborn as sns

# Set the aesthetic style of the plots
sns.set_style('whitegrid')

# List of numerical variables for univariate analysis
numerical_vars = ['number of bedrooms', 'number of bathrooms', 'living
area', 'lot area', 'Price']

# Plotting distributions of numerical variables
fig, axes = plt.subplots(len(numerical_vars), 1, figsize=(10, 20))
```

```
for i, var in enumerate(numerical_vars):
    sns.histplot(df[var], ax=axes[i], kde=True, color='skyblue')
    axes[i].set_title('Distribution of ' + var, fontsize=14)
    axes[i].set_xlabel('')
    axes[i].set_ylabel('Frequency')

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


Distribution of number of bedrooms
Frequency
0
1000
2000
3000
4000
5000
6000
0
5
10
15
20
25
30


Distribution of number of bathrooms
Frequency
0
500
1000
1500
2000
2500
3000
3500
1
2
3
4
5
6
7
8


Distribution of living area
Frequency
0
100
200
300
400
500
600
700
0
2000
4000
6000
8000
10000
12000
14000


Distribution of lot area
1000
800
600

Conclusion from univariate analysis.

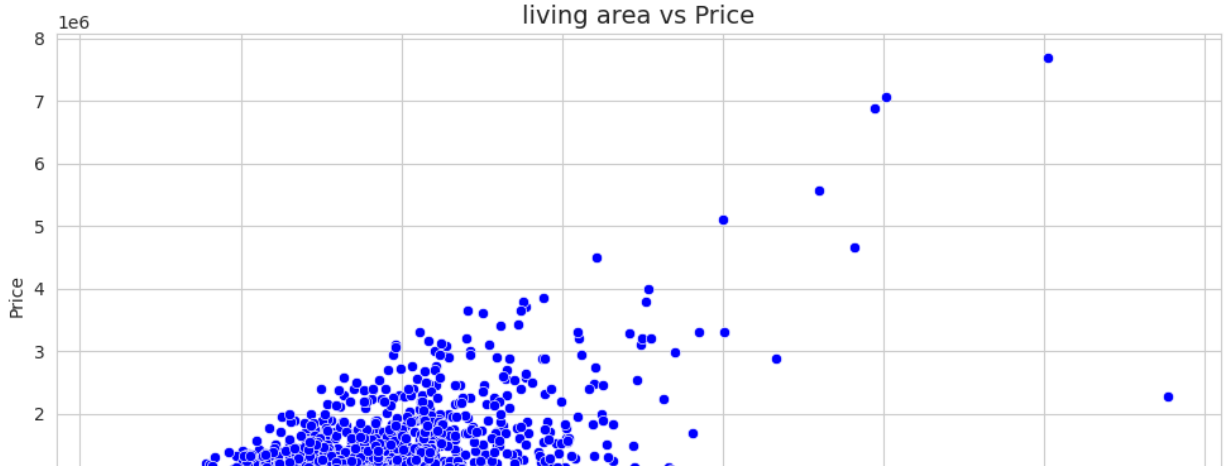
- Number of Bedrooms: Most houses have between 2 to 5 bedrooms, with a peak around 3 to 4 bedrooms.
- Number of Bathrooms: The distribution is similar to that of bedrooms, with most houses having between 1 to 3 bathrooms.
- Living Area: The living area shows a right-skewed distribution, indicating that while most houses have a smaller living area, there are a few houses with significantly larger living areas.
- Lot Area: Similar to the living area, the lot area is also right-skewed, with most houses having smaller lot areas but some exceptions with very large lot areas.
- Price: The price distribution is right-skewed, showing that most houses are in the lower price range, with a few houses being significantly more expensive.

1. Perform Bivariate Analysis

```
# Bivariate analysis with scatter plots for numerical variables
against Price

# Plotting relationships
fig, axes = plt.subplots(len(numerical_vars)-1, 1, figsize=(10, 20))
# Exclude 'Price' from variables

for i, var in enumerate(numerical_vars[:-1]):  # Exclude 'Price' from
variables
    sns.scatterplot(data=df, x=var, y='Price', ax=axes[i],
color='blue')
    axes[i].set_title(var + ' vs Price', fontsize=14)
    axes[i].set_xlabel(var)
    axes[i].set_ylabel('Price')

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


number of bedrooms vs Price
1e6
Price
number of bedrooms


number of bathrooms vs Price
1e6
Price
number of bathrooms


living area vs Price
1e6
Price

Conclusions from Bivariate Analysis.

- Number of Bedrooms vs Price: There seems to be a positive relationship between the number of bedrooms and the price, indicating that houses with more bedrooms tend to be more expensive.
- Number of Bathrooms vs Price: Similar to bedrooms, there is a positive relationship between the number of bathrooms and the price. More bathrooms generally mean a higher price.
- Living Area vs Price: The living area shows a strong positive correlation with price. Larger living areas significantly contribute to higher house prices.
- Lot Area vs Price: The relationship between lot area and price is less clear than for living area. While there is some indication that larger lot areas can lead to higher prices, the scatter is more dispersed, suggesting other factors also play a significant role in determining the price.

1. Perform Descriptive Statistics

```
df.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 14620 entries, 0 to 14619
Data columns (total 23 columns):
 #   Column                                 Non-Null Count  Dtype  
---  ------                                 --------------  -----  
 0   id                                     14620 non-null  int64  
 1   Date                                   14620 non-null  int64  
 2   number of bedrooms                     14620 non-null  int64  
 3   number of bathrooms                    14620 non-null  float64
 4   living area                            14620 non-null  int64  
 5   lot area                               14620 non-null  int64  
 6   number of floors                       14620 non-null  float64
 7   waterfront present                     14620 non-null  int64  
 8   number of views                        14620 non-null  int64  
 9   condition of the house                 14620 non-null  int64  
 10  grade of the house                     14620 non-null  int64  
 11  Area of the house(excluding basement)  14620 non-null  int64  
 12  Area of the basement                   14620 non-null  int64  
 13  Built Year                             14620 non-null  int64  
 14  Renovation Year                        14620 non-null  int64  
 15  Postal Code                            14620 non-null  int64  
 16  Lattitude                              14620 non-null  float64
 17  Longitude                              14620 non-null  float64
 18  living_area_renov                      14620 non-null  int64  
 19  lot_area_renov                         14620 non-null  int64  
 20  Number of schools nearby               14620 non-null  int64  
 21  Distance from the airport              14620 non-null  int64  
 22  Price                                  14620 non-null  int64
```

```
dtypes: float64(4), int64(19)
memory usage: 2.6 MB
```

```
df.describe()
```

```
{"type":"dataframe"}
```

```
# Getting dimensions of the dataset
```

```
df.shape
```

```
(14620, 23)
```

```
# Checking for null values
```

```
df.isnull().sum()
```

```
id                                          0
Date                                        0
number of bedrooms                          0
number of bathrooms                         0
living area                                 0
lot area                                    0
number of floors                            0
waterfront present                          0
number of views                             0
condition of the house                      0
grade of the house                          0
Area of the house(excluding basement)       0
Area of the basement                        0
Built Year                                  0
Renovation Year                             0
Postal Code                                 0
Lattitude                                   0
Longitude                                   0
living_area_renov                           0
lot_area_renov                              0
Number of schools nearby                    0
Distance from the airport                   0
Price                                       0
dtype: int64
```

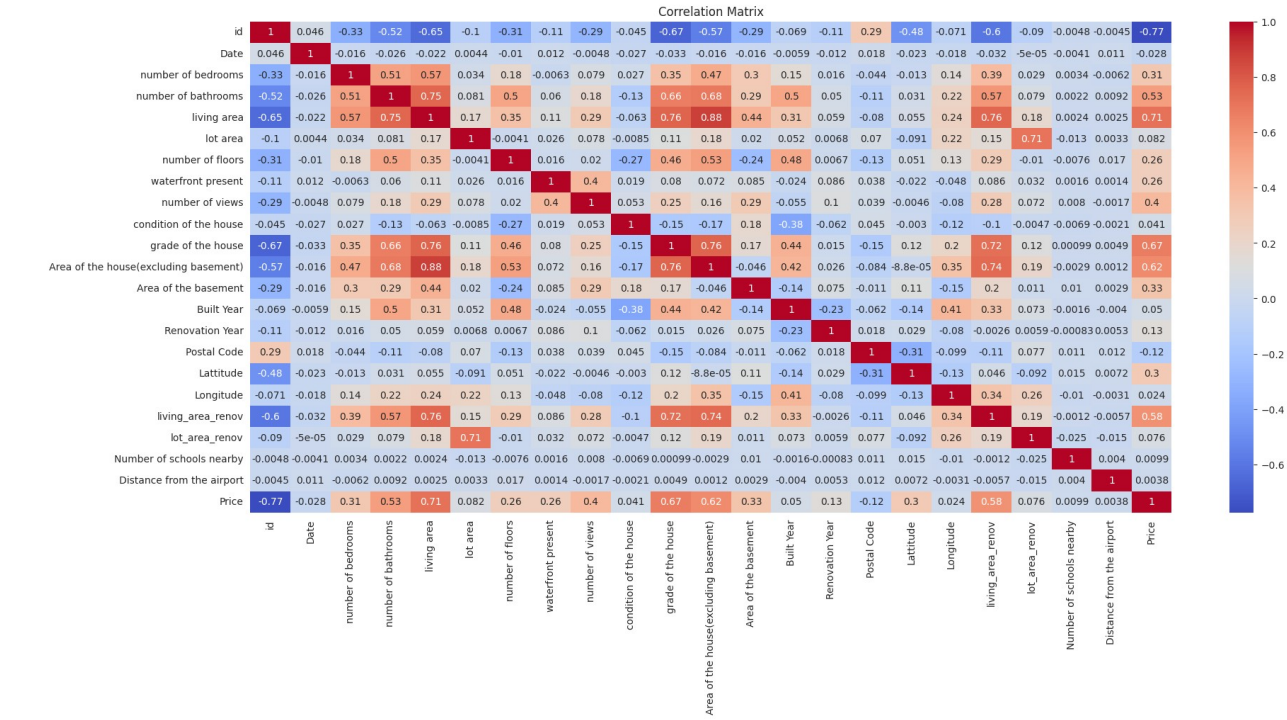
```
# Calculating descriptive statistics for 'Price'
min_price = df['Price'].min()
max_price = df['Price'].max()
range_price = max_price - min_price
variance_price = df['Price'].var()
std_dev_price = df['Price'].std()
q1_price = df['Price'].quantile(0.25)
q2_price = df['Price'].quantile(0.5)
q3_price = df['Price'].quantile(0.75)
```

```
iqr_price = q3_price - q1_price

# Printing the results
print(f'Minimum Price: {min_price}')
print(f'Maximum Price: {max_price}')
print(f'Range: {range_price}')
print(f'Variance: {variance_price}')
print(f'Standard Deviation: {std_dev_price}')
print(f'Q1 (25th percentile): {q1_price}')
print(f'Q2 (Median): {q2_price}')
print(f'Q3 (75th percentile): {q3_price}')
print(f'Interquartile Range (IQR): {iqr_price}')
```

```
Minimum Price: 78000
Maximum Price: 7700000
Range: 7622000
Variance: 135080050939.43219
Standard Deviation: 367532.38080396695
Q1 (25th percentile): 320000.0
Q2 (Median): 450000.0
Q3 (75th percentile): 645000.0
Interquartile Range (IQR): 325000.0
```

```
# Setting the background color to white for visibility
plt.figure(facecolor='white')

# Drawing a boxplot for the 'Price' column
sns.boxplot(x=df['Price'])
plt.title('Boxplot of House Prices')
plt.xlabel('Price')

# Displaying the plot
plt.show()
```


```
# Heatmap for correlation matrix
plt.figure(figsize=(21, 9), facecolor='white')
sns.heatmap(df.corr(), annot=True, cmap='coolwarm')
plt.title('Correlation Matrix')
plt.show()
```


Correlation Matrix
id 1 0.046 -0.33 -0.52 -0.65 -0.1 -0.31 -0.11 -0.29 -0.045 -0.67 -0.57 -0.29 -0.069 -0.11 0.29 -0.48 -0.071 -0.6 -0.09 -0.0048 -0.0045 -0.77
Date 0.046 1 -0.016 -0.026 -0.022 0.0044 -0.01 0.012 -0.0048 -0.027 -0.033 -0.016 -0.016 -0.0059 -0.012 0.018 -0.023 -0.018 -0.032 -5e-05 -0.0041 0.011 -0.028
number of bedrooms -0.33 -0.016 1 0.51 0.57 0.034 0.18 -0.0063 0.079 0.027 0.35 0.47 0.3 0.15 0.016 -0.044 -0.013 0.14 0.39 0.029 0.0034 -0.0062 0.31
number of bathrooms -0.52 -0.026 0.51 1 0.75 0.081 0.5 0.06 0.18 -0.13 0.66 0.68 0.29 0.5 0.05 -0.11 0.031 0.22 0.57 0.079 0.0022 0.0092 0.53
living area -0.65 -0.022 0.57 0.75 1 0.17 0.35 0.11 0.29 -0.063 0.76 0.88 0.44 0.31 0.059 -0.08 0.055 0.24 0.76 0.18 0.0024 0.0025 0.71
lot area -0.1 0.0044 0.034 0.081 0.17 1 -0.0041 0.026 0.078 -0.0085 0.11 0.18 0.02 0.052 0.0068 0.07 -0.091 0.22 0.15 0.71 -0.013 0.0033 0.082
number of floors -0.31 -0.01 0.18 0.5 0.35 -0.0041 1 0.016 0.02 -0.27 0.46 0.53 -0.24 0.48 0.0067 -0.13 0.051 0.13 0.29 -0.01 -0.0076 0.017 0.26
waterfront present -0.11 0.012 -0.0063 0.06 0.11 0.026 0.016 1 0.4 0.019 0.08 0.072 0.085 -0.024 0.086 0.038 -0.022 -0.048 0.086 0.032 0.0016 0.0014 0.26
number of views -0.29 -0.0048 0.079 0.18 0.29 0.078 0.02 0.4 1 0.053 0.25 0.16 0.29 -0.055 0.1 0.039 -0.0046 -0.08 0.28 0.072 0.008 -0.0017 0.4
condition of the house -0.045 -0.027 0.027 -0.13 -0.063 -0.0085 -0.27 0.019 0.053 1 -0.15 -0.17 0.18 -0.38 -0.062 0.045 -0.003 -0.12 -0.1 -0.0047 -0.0069 -0.0021 0.041
grade of the house -0.67 -0.033 0.35 0.66 0.76 0.11 0.46 0.08 0.25 -0.15 1 0.76 0.17 0.44 0.015 -0.15 0.12 0.2 0.72 0.12 0.00099 0.0049 0.67
Area of the house(excluding basement) -0.57 -0.016 0.47 0.68 0.88 0.18 0.53 0.072 0.16 -0.17 0.76 1 -0.046 0.42 0.026 -0.084 -8.8e-05 0.35 0.74 0.19 -0.0029 0.0012 0.62
Area of the basement -0.29 -0.016 0.3 0.29 0.44 0.02 -0.24 0.085 0.29 0.18 0.17 -0.046 1 -0.14 0.075 -0.011 0.11 -0.15 0.2 0.011 0.01 0.0029 0.33
Built Year -0.069 -0.0059 0.15 0.5 0.31 0.052 0.48 -0.024 -0.055 -0.38 0.44 0.42 -0.14 1 -0.23 -0.062 -0.14 0.41 0.33 0.073 -0.0016 -0.004 0.05
Renovation Year -0.11 -0.012 0.016 0.05 0.059 0.0068 0.0067 0.086 0.1 -0.062 0.015 0.026 0.075 -0.23 1 0.018 0.029 -0.08 -0.0026 0.0059 -0.00083 0.0053 0.13
Postal Code 0.29 0.018 -0.044 -0.11 -0.08 0.07 -0.13 0.038 0.039 0.045 -0.15 -0.084 -0.011 -0.062 0.018 1 -0.31 -0.099 -0.11 0.077 0.011 0.012 -0.12
Lattitude -0.48 -0.023 -0.013 0.031 0.055 -0.091 0.051 -0.022 -0.0046 -0.003 0.12 -8.8e-05 0.11 -0.14 0.029 -0.31 1 -0.13 0.046 -0.092 0.015 0.0072 0.3
Longitude -0.071 -0.018 0.14 0.22 0.24 0.22 0.13 -0.048 -0.08 -0.12 0.2 0.35 -0.15 0.41 -0.08 -0.099 -0.13 1 0.34 0.26 -0.01 -0.0031 0.024
living_area_renov -0.6 -0.032 0.39 0.57 0.76 0.15 0.29 0.086 0.28 -0.1 0.72 0.74 0.2 0.33 -0.0026 -0.11 0.046 0.34 1 0.19 -0.0012 -0.0057 0.58
lot_area_renov -0.09 -5e-05 0.029 0.079 0.18 0.71 -0.01 0.032 0.072 -0.0047 0.12 0.19 0.011 0.073 0.0059 0.077 -0.092 0.26 0.19 1 -0.025 -0.015 0.076
Number of schools nearby -0.0048 -0.0041 0.0034 0.0022 0.0024 -0.013 -0.0076 0.0016 0.008 -0.0069 0.00099 -0.0029 0.01 -0.0016 -0.00083 0.011 0.015 -0.01 -0.0012 -0.025 1 0.004 0.0099
Distance from the airport -0.0045 0.011 -0.0062 0.0092 0.0025 0.0033 0.017 0.0014 -0.0017 -0.0021 0.0049 0.0012 0.0029 -0.004 0.0053 0.012 0.0072 -0.0031 -0.0057 -0.015 0.004 1 0.0038
Price -0.77 -0.028 0.31 0.53 0.71 0.082 0.26 0.26 0.4 0.041 0.67 0.62 0.33 0.05 0.13 -0.12 0.3 0.024 0.58 0.076 0.0099 0.0038 1
id
Date
number of bedrooms
number of bathrooms
living area
lot area
number of floors
waterfront present
number of views
condition of the house
grade of the house
Area of the house(excluding basement)
Area of the basement
Built Year
Renovation Year
Postal Code
Lattitude
Longitude
living_area_renov
lot_area_renov
Number of schools nearby
Distance from the airport
Price
1.0
0.8
0.6
0.4
0.2
0.0
−0.2
−0.4
−0.6