

## DA ASSIGNMENT – 3

**NAME : E.PATHMESH**

**REGISTER NO 911720104044**

Load the dataset :

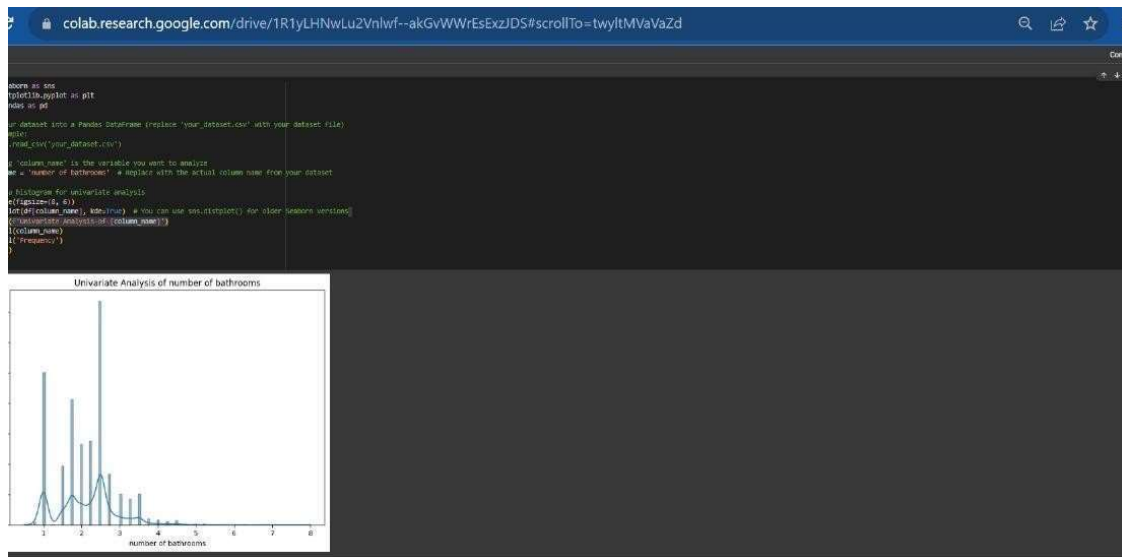
```
colab.research.google.com/drive/1R1yLHNwLu2Vnlwf--akGvWWrEsExzJDS

Untitled0.ipynb
File Edit View Insert Runtime Tools Help Last edited on October 5
+ Code + Text
Connect
(x)
import pandas as pd
df=pd.read_csv("/content/drive/MyDrive/House Price India.csv")
df.head()
```

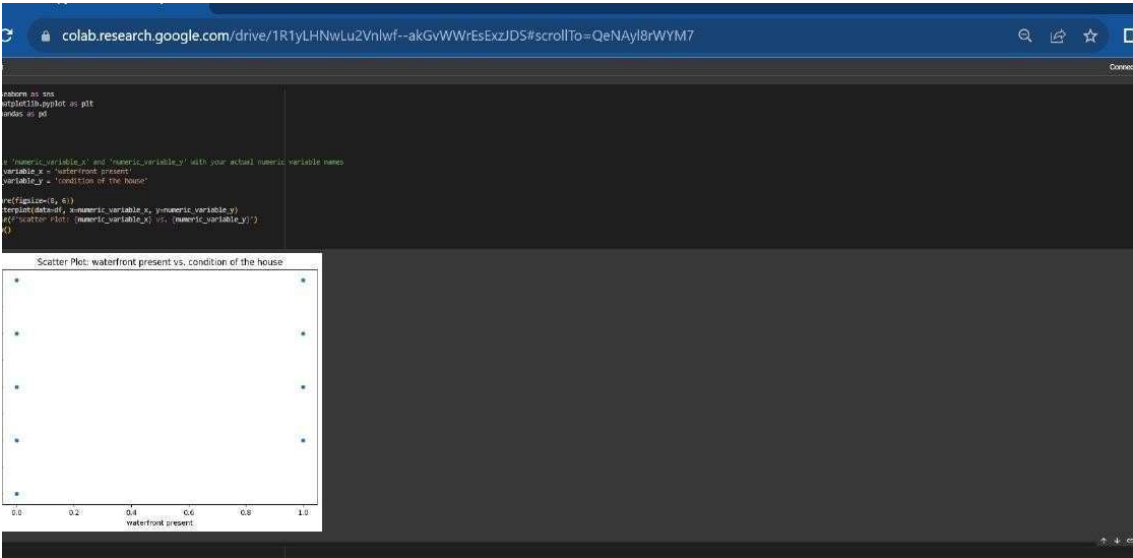
	id	Date	number of bedrooms	number of bathrooms	living area	lot area	number of floors	waterfront present	number of views	condition of the house	...	Built Year	Renovation Year	Postal Code	Latitude	Longitude
0	6762810145	42491	5	2.50	3650	9050	2.0	0	4	5	...	1921	0	122003	52.8645	-114.557
1	6762810635	42491	4	2.50	2920	4000	1.5	0	0	5	...	1909	0	122004	52.8878	-114.470
2	6762810998	42491	5	2.75	2910	9480	1.5	0	0	3	...	1939	0	122004	52.8852	-114.468
3	6762812605	42491	4	2.50	3310	42998	2.0	0	0	3	...	2001	0	122005	52.9532	-114.321
4	6762812919	42491	3	2.00	2710	4500	1.5	0	0	4	...	1929	0	122006	52.9047	-114.485

5 rows x 23 columns

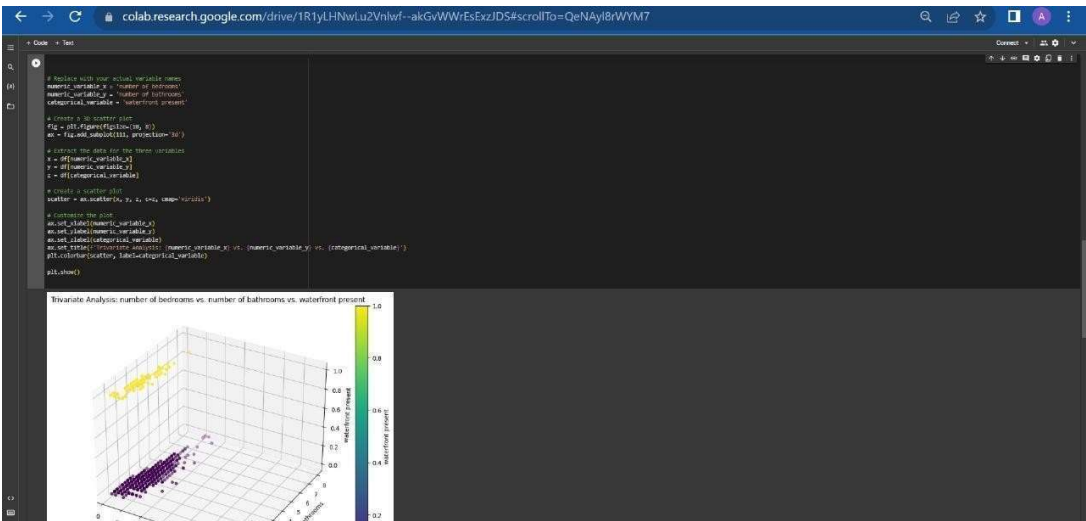
Univariate Analysis :



Bi - Variate Analysis :



Multi-Variate Analysis :



```

# Basic descriptive statistics for numeric columns
numeric_stats = ds.describe()
print(numeric_stats)


```

	id	date	number of bedrooms	number of bathrooms	
count	1.420000e+05	142000.00000	142000.00000	142000.00000	
mean	4.762222e+00	42045.310464	1.000000	1.212500	
std	6.227210e+00	427.319951	0.792643	2.122923	
min	4.762222e+00	42045.000000	1.000000	0.500000	
q1	4.762222e+00	42461.000000	1.000000	1.000000	
q2	4.762222e+00	42546.000000	1.000000	1.750000	
q3	4.762222e+00	42608.000000	1.000000	2.125000	
max	4.762222e+00	42642.000000	4.000000	2.750000	
min	4.762222e+00	42751.000000	22.000000	8.000000	

	listing area	lat	area	number of floors	underground percent	
count	142000.00000	1.420000e+05	142000.00000	142000.00000	142000.00000	
mean	2080.230778	1.5991770e+04	1.560708	0.007013		
std	508.129213	1.793502e+04	0.440270	0.401703		
min	370.000000	5.200000e+03	1.000000	0.000000		
q1	1040.000000	5.413010e+03	1.000000	0.000000		
q2	1830.000000	7.420000e+03	1.000000	0.000000		
q3	2730.000000	1.000000e+04	2.000000	0.000000		
max	51040.00000	1.6742110e+04	3.000000	1.000000		

	number of cities	countriy	of the house	built Year	
count	142000.00000	142000.00000	...	142000.00000	
mean	6.422118	2.300000	...	2070.000000	
std	6.762729	6.605513	...	24.497629	
min	0.000000	2.000000	...	1980.000000	
q1	0.000000	2.000000	...	2081.000000	
q2	0.000000	2.000000	...	2170.000000	
q3	0.000000	4.000000	...	2207.000000	
max	0.000000	5.000000	...	2241.000000	

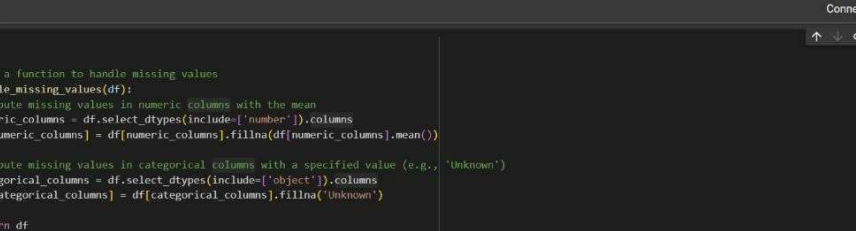
	Renovation year	Postal code	Latitude	Longitude	
count	142000.00000	142000.00000	142000.00000	142000.00000	
mean	20.310000	122017.482704	52.707000	114.480000	
std	444.100000	51.400444	0.117627	0.140106	
min	0.000000	112000.000000	52.100000	113.700000	
q1	0.000000	122017.000000	52.700000	114.170000	
q2	0.000000	122017.000000	52.700000	114.400000	
q3	0.000000	122017.000000	52.700000	114.110000	
max	2000.000000	122017.000000	53.400000	115.500000	

	listing area_renov	lat_area_renov	Number of schools nearby	
count	142000.00000	142000.00000	142000.00000	
mean	1096.362572	12761.500000	2.002244	
std	534.037146	24057.000000	0.521240	
min	400.000000	601.000000	1.000000	
q1	1140.000000	5607.000000	1.000000	
q2	1150.000000	7020.000000	2.000000	
q3	1200.000000	10107.000000	3.000000	
max	5100.000000	54017.000000	3.000000	

	distance from the airport	price	
count	142000.00000	1.420000e+05	
mean	46.958892	9.25922e+05	
std	5.710000	9.125720e+01	
min	30.000000	7.000000e+04	
q1	37.000000	2.000000e+05	
q2	50.000000	6.000000e+05	
q3	70.000000	2.000000e+06	



The screenshot shows a Google Colab notebook interface. The top bar includes navigation icons and a URL: `colab.research.google.com/drive/TR1yLHNwLu2Vnlwf--akGvWWrEsEcjJD5#scrollTo=jfvXn-NyX4fW`. The notebook has two tabs: '+ Code' (selected) and '+ Text'. On the right, there are icons for 'Connect', a user profile, and a dropdown menu. Below these are icons for undo, redo, copy, paste, and a terminal icon.

The code cell contains a Python script to handle missing values in a DataFrame:

```
# Define a function to handle missing values
def handle_missing_values(df):
    # Impute missing values in numeric columns with the mean
    numeric_columns = df.select_dtypes(include=['number']).columns
    df[numeric_columns] = df[numeric_columns].fillna(df[numeric_columns].mean())

    # Impute missing values in categorical columns with a specified value (e.g., 'Unknown')
    categorical_columns = df.select_dtypes(include=['object']).columns
    df[categorical_columns] = df[categorical_columns].fillna('Unknown')

    return df

# Apply the function to handle missing values
df = handle_missing_values(df)

# Check if there are any remaining missing values
missing_values_count = df.isnull().sum().sum()
if missing_values_count == 0:
    print("All missing values have been handled.")
else:
    print(f"There are still {missing_values_count} missing values in the dataset.")

# Save the cleaned dataset to a new file (optional)
df.to_csv('cleaned_dataset.csv', index=False) # Replace with your desired file name
```

At the bottom of the code cell, a status message indicates: "All missing values have been handled."