

DA ASSIGNMENT – 3

NAME : POLAMREDDY KAVYA

REGISTER NO : 723920104046

Load the dataset :

```
colab.research.google.com/drive/1R1yLHNwLu2VnWf--akGvWWrEsExzJDS

Untitled0.ipynb
File Edit View Insert Runtime Tools Help Last edited on October 5

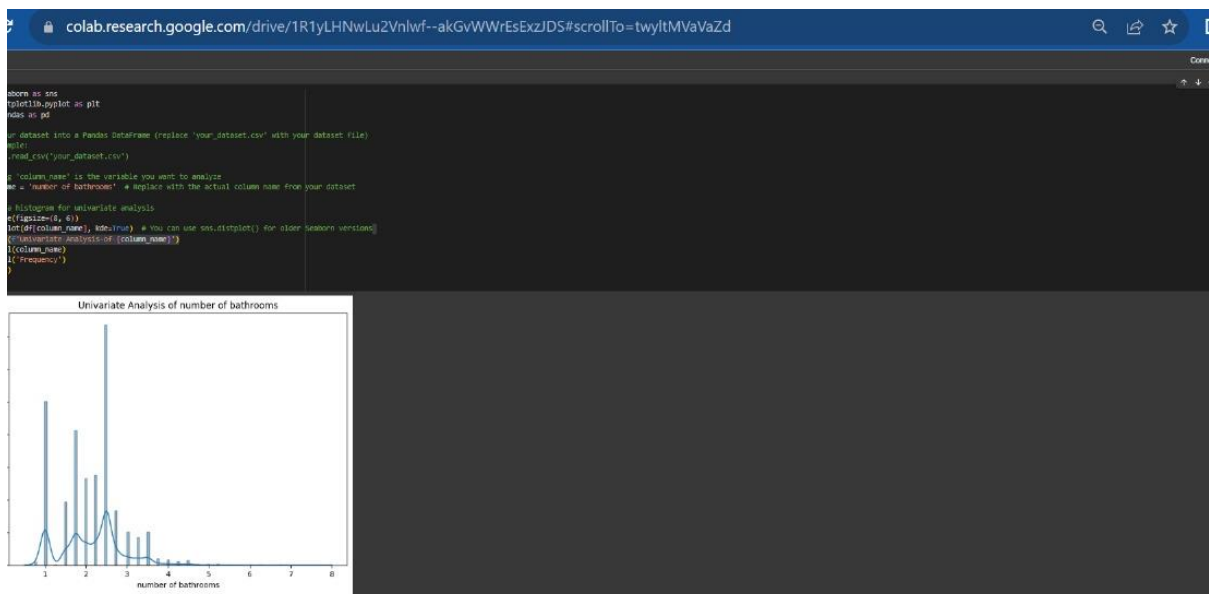
+ Code + Text
Connect

import pandas as pd
df=pd.read_csv("/content/drive/MyDrive/House Price India.csv")
df.head()
```

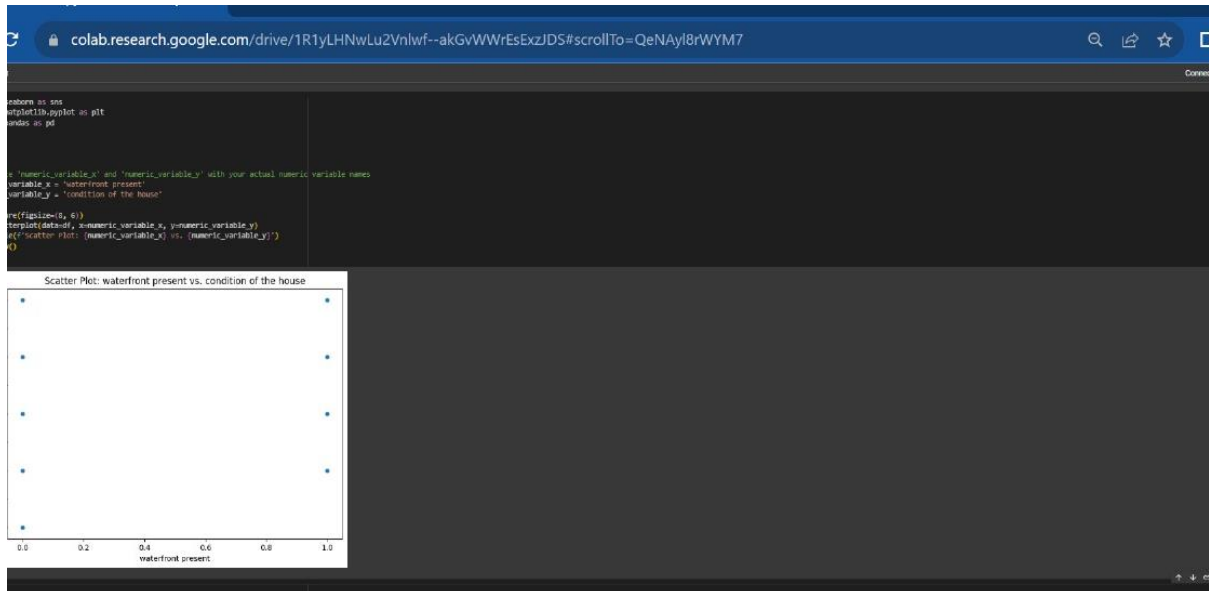
	id	Date	number of bedrooms	number of bathrooms	living area	lot area	number of floors	waterfront present	number of views	condition of the house	...	Built Year	Renovation Year	Postal Code	Latitude	Longitude
0	6762810145	42491	5	2.50	3650	9050	2.0	0	4	5	...	1921	0	122003	52.8645	-114.557
1	6762810635	42491	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

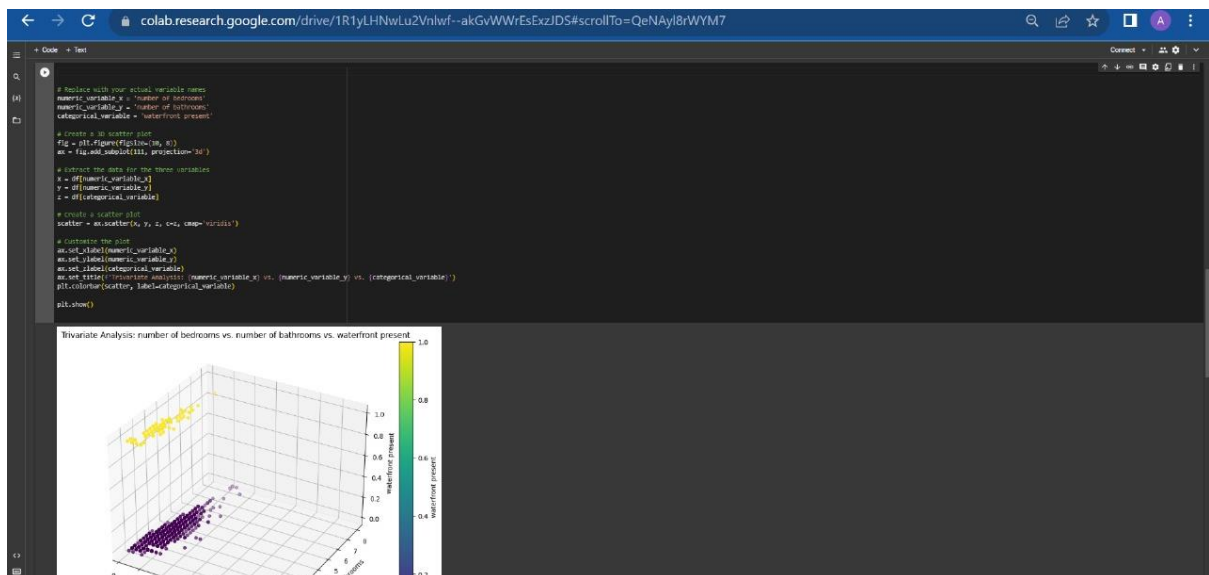
Univariate Analysis :



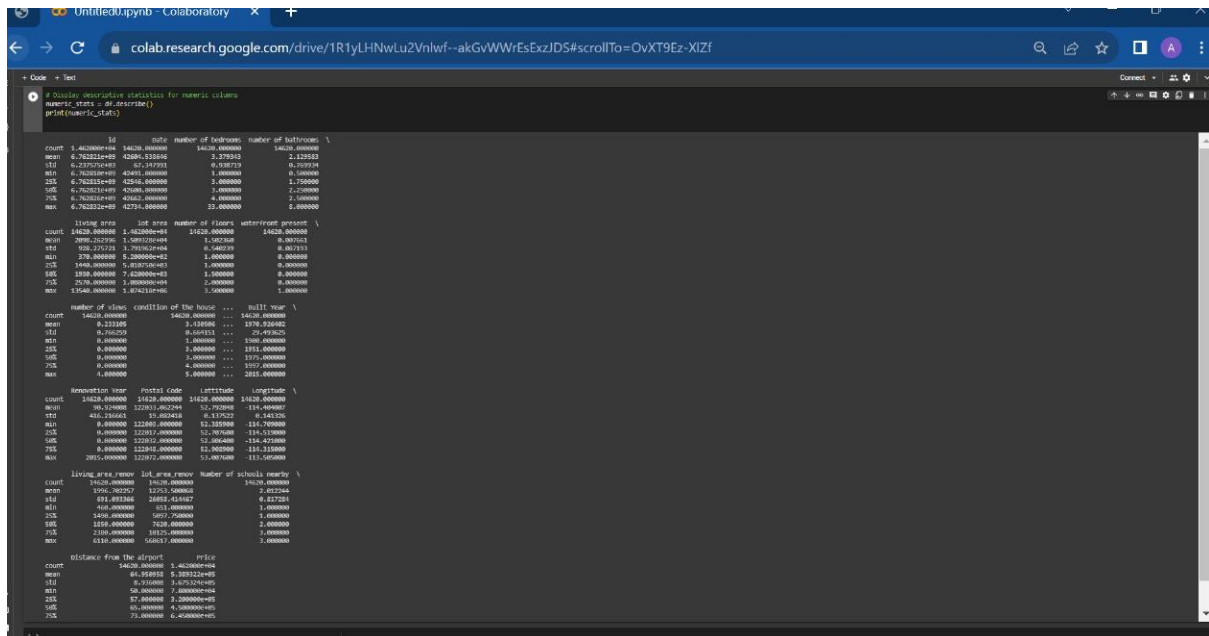
Bi - Variate Analysis :



Multi-Variate Analysis :



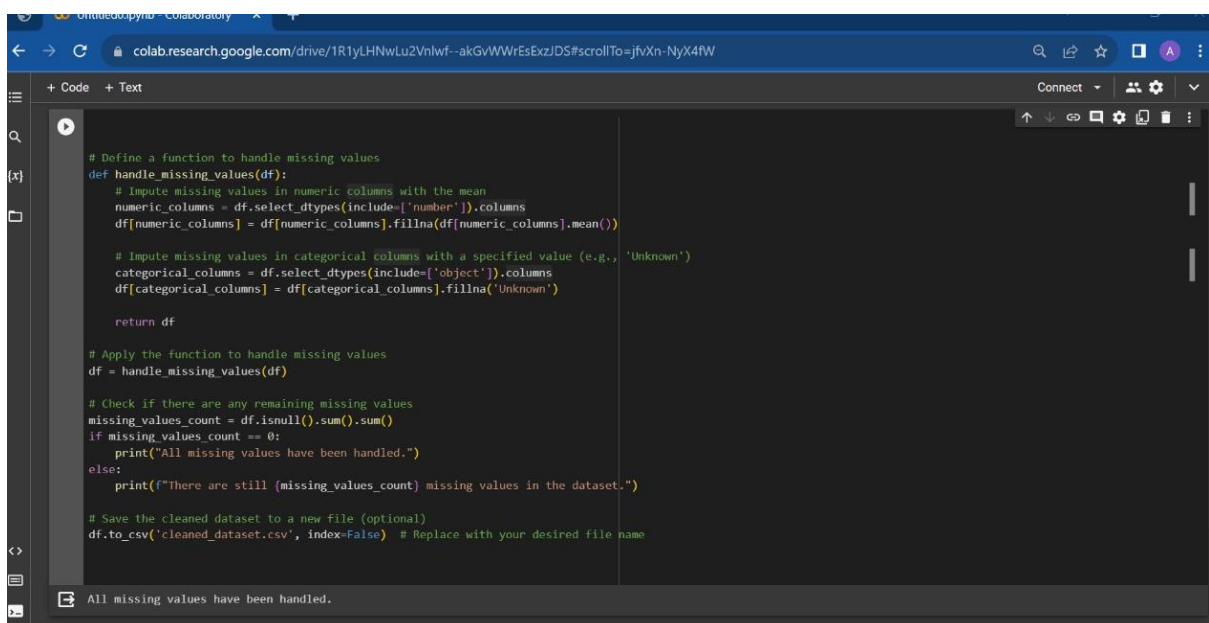
Descriptive statistics on the dataset :



```
# Display descriptive statistics for numeric columns
numeric_stats = df.describe()
print(numeric_stats)
```

	count	mean	std	min	max	25%	75%
living_area	14628.000000	14628.000000	14628.000000	14628.000000	14628.000000	14628.000000	14628.000000
lot_area	14628.000000	14628.000000	14628.000000	14628.000000	14628.000000	14628.000000	14628.000000
number of floors	14628.000000	1.000000	0.000000	1.000000	1.000000	1.000000	1.000000
waterfront present	14628.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
number of views	14628.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
condition of the house	14628.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
built year	14628.000000	1970.000000	10.000000	1950.000000	1990.000000	1960.000000	1980.000000
renovation year	14628.000000	1970.000000	10.000000	1950.000000	1990.000000	1960.000000	1980.000000
Postal Code	14628.000000	14628.000000	14628.000000	14628.000000	14628.000000	14628.000000	14628.000000
latitude	14628.000000	45.500000	0.000000	45.000000	46.000000	45.250000	45.750000
longitude	14628.000000	-73.500000	0.000000	-74.000000	-73.000000	-73.750000	-73.250000
living_area_renov	14628.000000	14628.000000	14628.000000	14628.000000	14628.000000	14628.000000	14628.000000
lot_area_renov	14628.000000	14628.000000	14628.000000	14628.000000	14628.000000	14628.000000	14628.000000
number of schools nearby	14628.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
distance from the airport	14628.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000

Handle the Missing values :



```
# 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
```

All missing values have been handled.