**REAL ESTATE**

**Overview of the Dataset:**

The dataset for "Real Estate" contains the following features:

1. PropertyID: Unique identifier for each property.
2. xrCompositeLandUseID: Identifier for the composite land use category of the property.
3. xrBuildingTypeID: Identifier for the type of building on the property.
4. ParcelID: Identifier for the parcel associated with the property.
5. LocationStartNumber: Starting number of the property's location.
6. ApartmentUnitNumber: Number of the apartment unit, if applicable.
7. StreetNameAndWay: Name of the street or way where the property is located.
8. xrPrimaryNeighborhoodID: Identifier for the primary neighborhood where the property is located.
9. LandSF: Area of land in square feet.
10. TotalFinishedArea: Total finished area of the property.
11. LivingUnits: Number of living units in the property.
12. OwnerLastName: Last name of the property owner.
13. OwnerFirstName: First name of the property owner.
14. PrimaryGrantor: Grantor associated with the property.
15. SaleDate: Date of the property sale.
16. SalePrice: Price at which the property was sold.
17. TotalAppraisedValue: Total appraised value of the property.
18. LegalReference: Legal reference associated with the property.
19. xrSalesValidityID: Identifier for the sales validity status.
20. xrDeedID: Identifier for the deed associated with the property.

This dataset appears to provide information about various aspects of real estate properties, including their identifiers, location, land use, building type, area, ownership, sales details, and appraised values.

Top of Form

**Overview of the Tool used: Python**

Python is a versatile and widely used programming language that is extensively utilized for data analysis, including the analysis of the "Telecom Users" dataset.:

1. Versatile and Easy to Use
2. Abundance of Libraries
3. Data Manipulation and Exploration
4. Data Visualization
5. Machine Learning and Statistical Analysis
6. Integration and Collaboration
7. Scalability and Performance

**Summary:**

Number of Rows: **4735**

Number of Columns: **20**

**Pre-processing:**

import pandas as pd

import numpy as np

df=pd.read\_csv("real-estate-sales.csv")

**# Feature Selection**

df1=df.copy()

df1.drop(df.columns[[0,1,2,3,5,12,18,19]], axis=1, inplace=True)

df1.isnull().sum()

**#Filling Null values**

df1['LocationStartNumber']=df1['LocationStartNumber'].fillna(0)

df1['TotalFinishedArear']=df1['TotalFinishedArea'].fillna(0)

df1['LivingUnitsr']=df1['LivingUnits'].fillna(0)

df1['LandSFr']=df1['LandSF'].fillna(0)

**#Finding Mean Values**

df1['LandSF'].fillna(df1['LandSF'].mean())

df1['TotalFinishedArea'].fillna(df1['TotalFinishedArea'].mean())

df1['LivingUnits'].fillna(df1['LivingUnits'].mean())

1**. Total Sales per Quarter of the Real Estate**

**Code:**

df['SaleDate'] = pd.to\_datetime(df['SaleDate'])

df1['Quarter'] = df1['SaleDate'].dt.quarter

quarterly\_sales = df1.groupby('Quarter')['SalePrice'].sum()

import matplotlib.pyplot as plt

plt.figure(figsize=(6,6))

plt.barh(quarterly\_sales.index, quarterly\_sales)

plt.xlabel("Amount of Sales")

plt.ylabel("Quarter")

plt.title("Quarterly Sale")

**Output:**

Quarter

1 682912951

2 247882984

3 653169968

4 1887341730

**Chart:**

****

1. **Average Sale price per Area of the Real Estate**

**Code:**

avg\_sale = df1.groupby(['StreetNameAndWay'])['SalePrice'].mean()

plt.figure(figsize=(10,6))

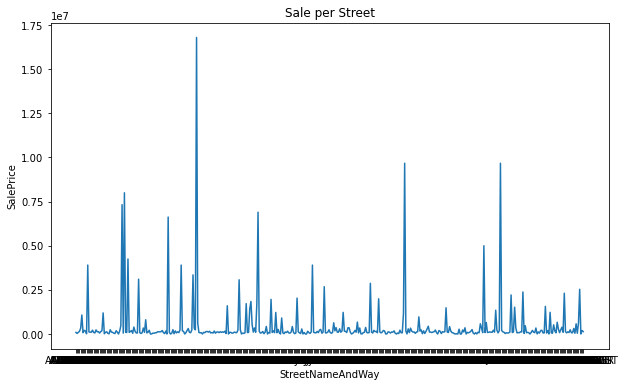
plt.plot(avg\_sale)

plt.xlabel("StreetNameAndWay")

plt.ylabel("SalePrice")

plt.title("Sale per Street")

**Chart:**

****

1. **Which month has the biggest Sale**

**Code:**

from datetime import datetime

import calendar

df1['month'] = df1['SaleDate'].dt.month

month\_number = df1['month'].loc[df1['SalePrice'].idxmax()]

Max\_amount = df1['SalePrice'].loc[df1['SalePrice'].idxmax()]

max\_sales\_month=calendar.month\_name[month\_number]

print("The Higest sale is->", Max\_amount, "->in the Month of--> ", max\_sales\_month)

**Output:**

The Highest sale is-> 70500000 ->in the Month of--> May

1. **Maximum sales per Area**

**Code:**

df1.reset\_index(drop=True)

Max\_Sale = df1.groupby(['StreetNameAndWay'])['SalePrice'].max().reset\_index(name='max')

print(Max\_Sale)

plt.figure(figsize=(10,6))

plt.bar(range(len(Max\_Sale)), Max\_Sale["max"])

plt.xlabel("StreetNameAndWay")

plt.ylabel("SalePrice")

plt.title("Maximum Sale per Street")

**Chart:**

|  |  |
| --- | --- |
|  |  |

1. **Total Finished Area VS Sale Price**

**Code:**

# Extract the variables

x = df['TotalFinishedArea']

y = df['SalePrice']

# Create the scatter plot

plt.scatter(x, y, s=10, alpha=0.5, color='blue')

# Set labels and title

plt.xlabel('Total Finished Area')

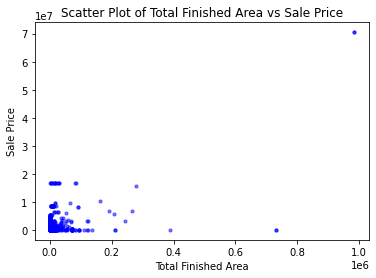
plt.ylabel('Sale Price')

plt.title('Scatter Plot of Total Finished Area vs Sale Price')

# Display the plot

plt.show()

**Chart:**



1. **StreetNameAndWay VS Sale Price**

**Code:**

# Extract the variables

x = df['StreetNameAndWay']

y = df['SalePrice']

# Create the scatter plot

plt.scatter(x, y, s=10, alpha=0.5, color='blue')

# Set labels and title

plt.xlabel('StreetNameAndWay')

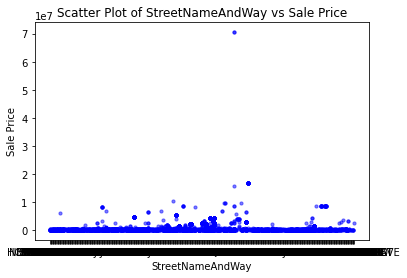
plt.ylabel('Sale Price')

plt.title('Scatter Plot of StreetNameAndWay vs Sale Price')

# Display the plot

plt.show()

**Chart:**

****

1. **Top 5 Sales**

**Code:**

Max\_sale= df1.sort\_values(by ='SalePrice', ascending=False)[['StreetNameAndWay','SalePrice']]

Top\_sale = Max\_sale.head(5)

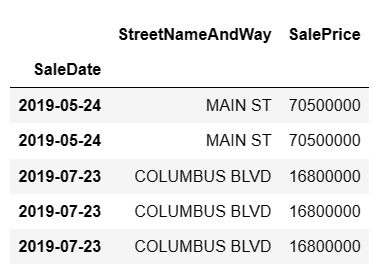
Top\_sale

(0r)

Sort\_Max\_sale= Max\_sale.sort\_values(by ='SalePrice', ascending=False)

Sort\_Max\_sale.nlargest(5,'SalePrice')

**Output:**

****