Lab Report 02

Datasets & Data Handling

Name MaazAhmad
Reg No B23F722AI170
Section Ai yellow
Submitted to Abdullah Sajid

Prerequisites

First, import the necessary libraries

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt

# --- STEP 0: CREATE DUMMY DATA (Run this once to generate the file) ---
data = {
    'Area': [1200, 1500, np.nan, 1800, 2000, 2200, 1100, np.nan, 1400, 1600, 2500, 3000],
    'Bedrooms': [2, 3, 3, 4, 4, 5, 2, 3, 3, 3, 5, 5],
    'Price': [200000, 250000, 245000, 320000, 350000, 400000, 190000, 260000, 230000, 270000, 450000, 500000]
}
df_dummy = pd.DataFrame(data)
df_dummy.to_csv('housing.csv', index=False)
print("Dummy 'housing.csv' created successfully.")
# ----------------------------------------------------------------------
```

```
Dummy 'housing.csv' created successfully.
```

Task no 1

Loading and Inspection

```
# 1. Load Data
df = pd.read_csv('housing.csv')

# 2. Inspect Rows
print("First 10 Rows:\n", df.head(10))
print("\nLast 10 Rows:\n", df.tail(10))

# 3. Check Columns and Types
print("\nColumn Names:", df.columns.tolist())
print("\nData Types:\n", df.dtypes)
```

```
First 10 Rows:
      Area  Bedrooms   Price
0  1200.0         2  200000
1  1500.0         3  250000
2     NaN         3  245000
3  1800.0         4  320000
4  2000.0         4  350000
5  2200.0         5  400000
6  1100.0         2  190000
7     NaN         3  260000
8  1400.0         3  230000
9  1600.0         3  270000

Last 10 Rows:
       Area  Bedrooms   Price
2      NaN         3  245000
3   1800.0         4  320000
4   2000.0         4  350000
5   2200.0         5  400000
6   1100.0         2  190000
```

```
7      NaN         3  260000
8   1400.0         3  230000
9   1600.0         3  270000
10  2500.0         5  450000
11  3000.0         5  500000

Column Names: ['Area', 'Bedrooms', 'Price']

Data Types:
 Area        float64
Bedrooms      int64
Price         int64
dtype: object
```

Description

Loading and Inspection:
Is code main hum ne pd.read_csv use kar ke housing dataset load kiya aur head(10) aur tail(10) se start aur end wali rows check ki. Sath hi df.dtypes print kiya taake hamein pata chalay ke kaunsa column number hai aur kaunsa text.

Task no 2

Data Statistics & Filtering

```
# 1. Calculate Statistics for 'Price'
price_mean = df['Price'].mean()
price_median = df['Price'].median()
price_std = df['Price'].std()

print(f"Price Mean: {price_mean:.2f}")
print(f"Price Median: {price_median:.2f}")
print(f"Price Std Dev: {price_std:.2f}")

# 2. Filter Data (Bedrooms > 3)
large_houses = df[df['Bedrooms'] > 3]
print("\nHouses with > 3 Bedrooms:\n", large_houses)
```

```
Price Mean: 305416.67
Price Median: 265000.00
Price Std Dev: 100033.14

Houses with > 3 Bedrooms:
       Area  Bedrooms   Price
3   1800.0         4  320000
4   2000.0         4  350000
5   2200.0         5  400000
10  2500.0         5  450000
11  3000.0         5  500000
```

Description

Data Statistics & Filtering:
Hum ne 'Price' column ka Mean, Median aur Standard Deviation calculate kiya taake data ki central tendency ka andaza ho sakay. Phir hum ne dataset par condition lagayi (Bedrooms > 3) taake sirf baray gharon ka data filter ho kar samnay aaye.

Task no 3

Handling Null Values

```
# 1. Identify Missing Values
print("Missing Values before clean:\n", df.isnull().sum())

# 2. Fill missing 'Area' with Median (Imputation)
```

```
area_median = df['Area'].median()
df['Area'] = df['Area'].fillna(area_median)

# 3. Drop rows that still have nulls (if any remain in other columns)
df.dropna(inplace=True)

print("\nMissing Values after clean:\n", df.isnull().sum())
print("New Shape of Data:", df.shape)
```

```
Missing Values before clean:
 Area        2
Bedrooms    0
Price       0
dtype: int64

Missing Values after clean:
 Area        0
Bedrooms    0
Price       0
dtype: int64
New Shape of Data: (12, 3)
```

Description

Handling Null Values:
Sab se pehle isnull().sum() se check kiya ke kahan data missing hai, phir 'Area' column ki missing values ko Median se fill (impute) kiya. Jo baqi rows main missing data reh gaya tha, unhein dropna use kar ke remove kar diya taake dataset clean ho jaye.

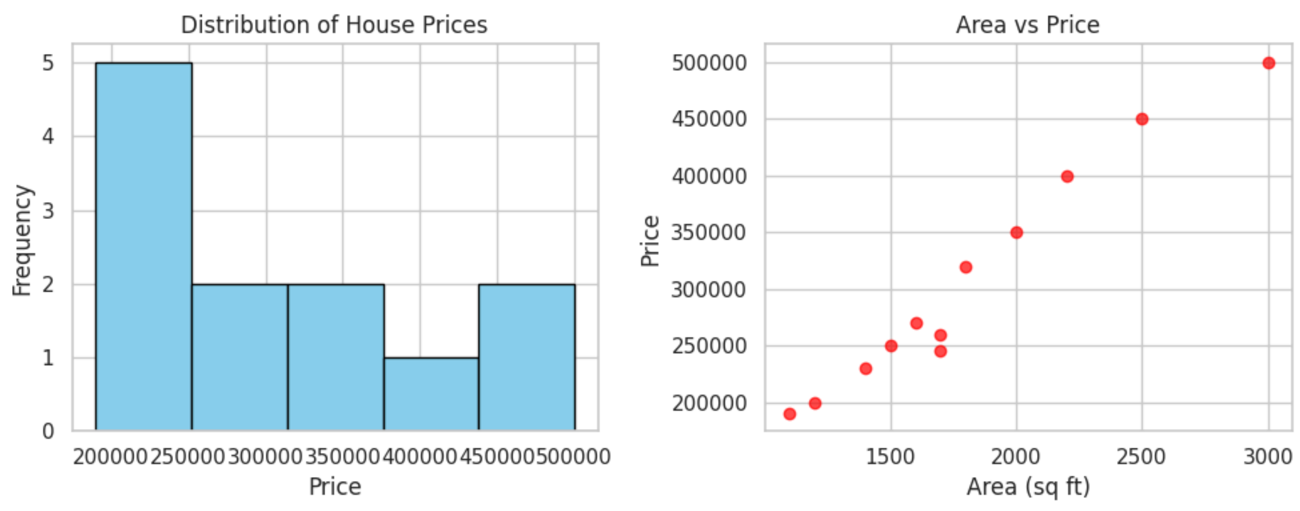
Task no 4

Simple Visualization

```
# 1. Histogram of Price
plt.figure(figsize=(10, 4))
plt.subplot(1, 2, 1) # 1 row, 2 columns, plot 1
plt.hist(df['Price'], bins=5, color='skyblue', edgecolor='black')
plt.title('Distribution of House Prices')
plt.xlabel('Price')
plt.ylabel('Frequency')

# 2. Scatter Plot (Area vs Price)
plt.subplot(1, 2, 2) # 1 row, 2 columns, plot 2
plt.scatter(df['Area'], df['Price'], color='red', alpha=0.7)
plt.title('Area vs Price')
plt.xlabel('Area (sq ft)')
plt.ylabel('Price')

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



Description

Simple Visualization:
Hum ne matplotlib use kar ke 'Price' ka histogram banaya taake prices ki frequency distribution dekh saken. Us ke baad 'Area' aur 'Price' ka scatter plot banaya taake visual form main dekh saken ke area barhnay se price par kya asar hota hai.

Lab Summary

1. **Dataset Introduction:** This lab introduces the concept of datasets (rows/columns) and using Pandas (Python Data Analysis Library) to handle them efficiently.
2. **Loading Data:** The first step is always loading data into a DataFrame using functions like read_csv, which makes the data ready for manipulation.
3. **Exploratory Data Analysis (EDA):** We learned to inspect the data structure using .info(), .head(), and .describe() to understand features before applying any logic.
4. **Statistical Analysis:** Calculating basic statistics like Mean (average), Median (middle value), and Standard Deviation helps in understanding the distribution of data.
5. **Data Filtering:** Boolean indexing allows us to extract specific subsets of data that meet certain criteria (e.g., finding all houses with > 3 bedrooms).
6. **Handling Missing Values (NaN):** Real-world data is messy; we must identify null values using .isnull() to prevent errors in machine learning models.
7. **Imputation vs. Dropping:** We learned two ways to fix missing data: Imputation (filling with mean/median) to save data, or Dropping rows if data is sufficient.
8. **Data Visualization:** Using Matplotlib for basic plots (Histogram, Scatter) is crucial to visualize patterns and correlations (like Area vs. Price) that numbers alone cannot show.
