LAB Manual

PART A

(PART A : TO BE REFFERED BY STUDENTS)

**Experiment No.03**

PART B

(PART B : TO BE COMPLETED BY STUDENTS)

***(Students must submit the soft copy as per following segments within two hours of the practical. The soft copy must be uploaded on the Blackboard or emailed to the concerned lab in charge faculties at the end of the practical in case there is no Black board access available)***

|  |  |
| --- | --- |
| Roll No.: C026 | Name: Anirbaan Ghatak |
| Class : B | Batch : B1 |
| Date of Experiment: 08/08/2023 | Date of Submission: 09/08/2023 |
| Grade : |  |

**B.1 Software Code written by student:**

#Aim: Perform Exploratory Data Analysis on real world dataset using Pandas library

# i.   Read different types of data files (.csv, excel, text file etc.)

# ii.  Obtain metadata of given dataset

# iii. Handling Missing Values in dataset

# iv.  Exploratory data analysis using different commands

#Name: Anirbaan Ghatak

#Roll no.: C026

import pandas as pd

df = pd.read\_csv('USA\_Housing.csv')

df.head(10)

df.dtypes

df.shape

df.size

df.isna().count()

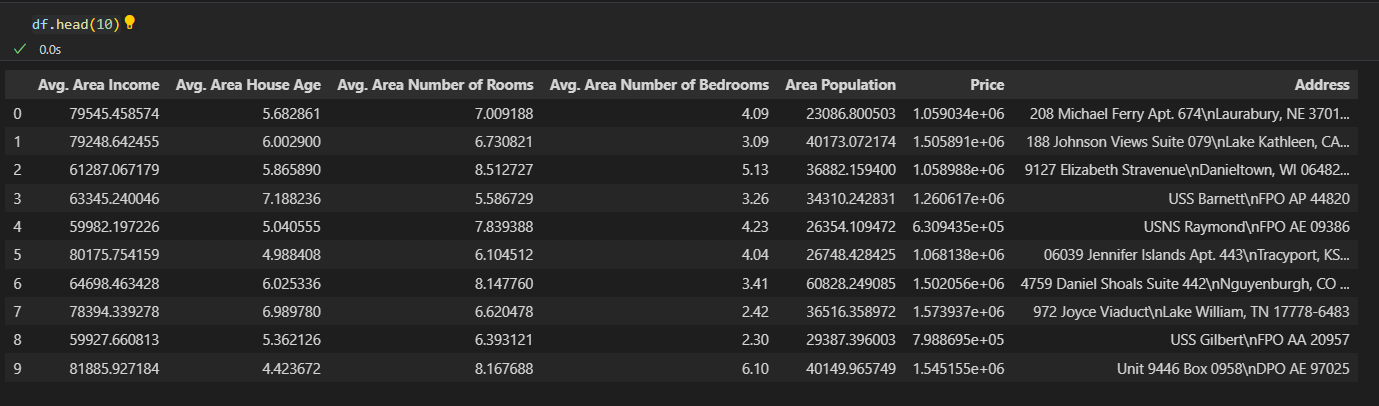
df.fillna(df.mean(), inplace=True)

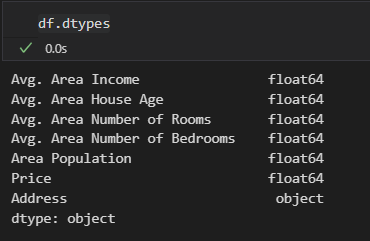
df.dropna(inplace=True)

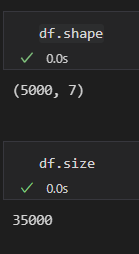
df.head(10)

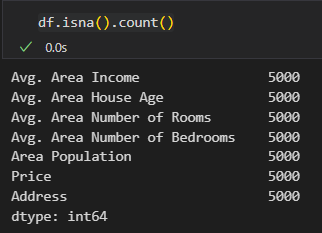
df.tail()

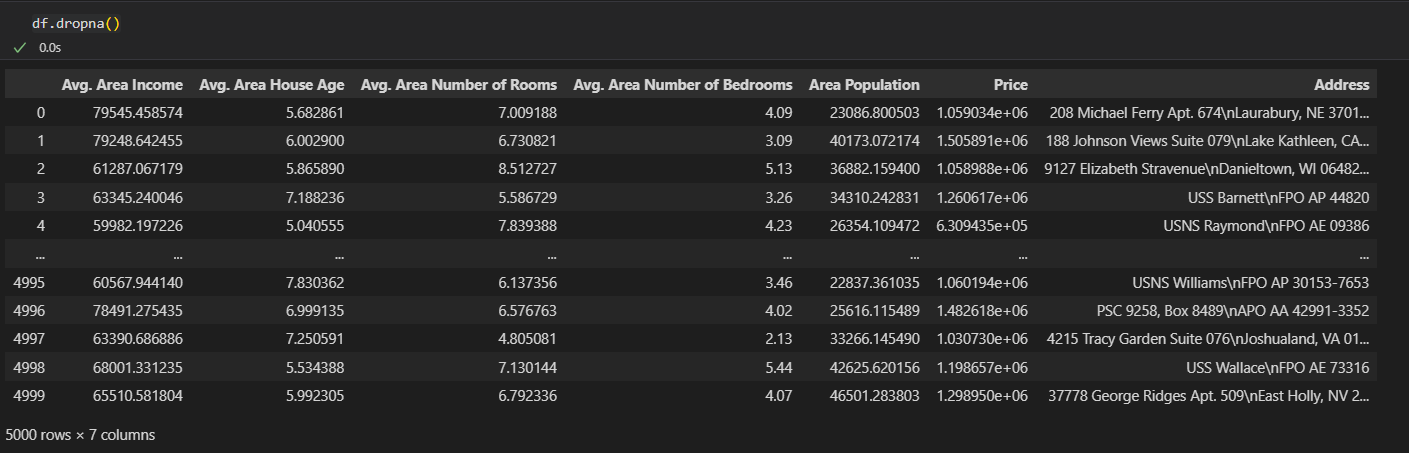
**B.2 Input and Output:**

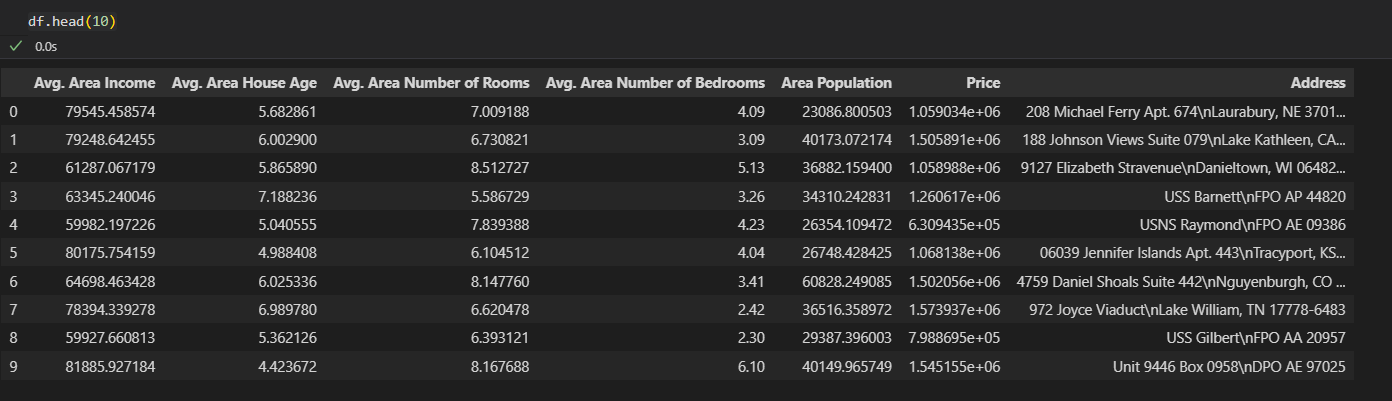
******

****

****

****

****

****

****

**B.3 Observations and learning:**

*The code loads inspects the csv file selected with .head() and .tail(), handles missing values, and extracts metadata using .dtypes, .shape, and .size. It underscores data preprocessing's significance, emphasizes metadata's role, and showcases missing value treatment's necessity.*

**B.4 Conclusion:**

*This code establishes a foundational exploratory data analysis step, with room to advance by incorporating visualizations, statistical analysis, and domain-specific exploration for comprehensive insights.*

**B.5 Question of Curiosity**

1. Explore the commands which are not described in the document and you have used for exploring your selected dataset. List those commands here and describe it in brief.

Additional Pandas commands that can be used for exploring the selected dataset

df.describe(): This command generates summary statistics of the numerical columns in the DataFrame. It provides information such as mean, standard deviation, minimum, maximum, and percentiles, giving an overview of the data distribution.

df.info(): This command provides a concise summary of the DataFrame's information, including the number of non-null values and data types of each column. It's useful for understanding the completeness of the dataset and identifying data types.

df['Column\_Name'].value\_counts(): This command counts the occurrences of unique values in a specific column. It helps to understand the distribution of categorical data and identify dominant categories.

df.corr(): This command computes the correlation matrix for numerical columns. It shows how different columns are related to each other, highlighting potential patterns or dependencies.

df.groupby('Column\_Name').mean(): This command groups the data by unique values in a specific column and calculates the mean of numerical columns for each group. It's useful for analyzing trends within different categories.

pd.to\_datetime(df['Date\_Column']): This command converts a column containing date strings into a Pandas datetime format, enabling time-based analysis and visualization.

df['Column\_Name'].apply(function): This command applies a custom function to a column's values, allowing for data transformation or feature engineering.

df['Column\_Name'].plot(kind='hist'): This command generates a histogram for a specific column, visualizing the distribution of numerical data.

df['Column\_Name'].plot(kind='box'): This command creates a box plot for a numerical column, helping to identify outliers and understand the spread of the data.

pd.pivot\_table(df, values='Value\_Column', index='Index\_Column', columns='Column\_Name'): This command creates a pivot table that summarizes data based on specified index and column values. It's useful for cross-tabulation and multidimensional analysis.

These commands provide additional tools for exploring and understanding the dataset. They offer insights into data distribution, relationships, and trends, enhancing the depth of Exploratory Data Analysis.