DSC540-T301_2245_1 Data Preparation

Assignment Week 3 & 4;

Author: Zemelak Goraga;

Date: 4/6/2024

Activity 3.01

In [4]:
```
# Step 1: Load necessary libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
```

In [5]:
```
# Step 2: Read in the Boston housing dataset
housing_data = pd.read_csv('Boston_housing.csv')
```

In [6]:
```
# Step 3: Check the first 10 records and find the total number of records
print("First 10 records:")
print(housing_data.head(10))
print("\nTotal number of records:", len(housing_data))
```

```
First 10 records:
      CRIM    ZN  INDUS  CHAS    NOX     RM    AGE     DIS  RAD  TAX  PTRATIO  \
0  0.00632  18.0   2.31     0  0.538  6.575   65.2  4.0900    1  296     15.3
1  0.02731   0.0   7.07     0  0.469  6.421   78.9  4.9671    2  242     17.8
2  0.02729   0.0   7.07     0  0.469  7.185   61.1  4.9671    2  242     17.8
3  0.03237   0.0   2.18     0  0.458  6.998   45.8  6.0622    3  222     18.7
4  0.06905   0.0   2.18     0  0.458  7.147   54.2  6.0622    3  222     18.7
5  0.02985   0.0   2.18     0  0.458  6.430   58.7  6.0622    3  222     18.7
6  0.08829  12.5   7.87     0  0.524  6.012   66.6  5.5605    5  311     15.2
7  0.14455  12.5   7.87     0  0.524  6.172   96.1  5.9505    5  311     15.2
8  0.21124  12.5   7.87     0  0.524  5.631  100.0  6.0821    5  311     15.2
9  0.17004  12.5   7.87     0  0.524  6.004   85.9  6.5921    5  311     15.2

        B  LSTAT  PRICE
0  396.90   4.98   24.0
1  396.90   9.14   21.6
2  392.83   4.03   34.7
3  394.63   2.94   33.4
4  396.90   5.33   36.2
5  394.12   5.21   28.7
6  395.60  12.43   22.9
7  396.90  19.15   27.1
8  386.63  29.93   16.5
9  386.71  17.10   18.9

Total number of records: 506
```

In [7]:
```
# Step 4: Create a smaller DataFrame excluding specified columns
smaller_df = housing_data.drop(columns=['CHAS', 'NOX', 'B', 'LSTAT'])
smaller_df
```

Out[7]:

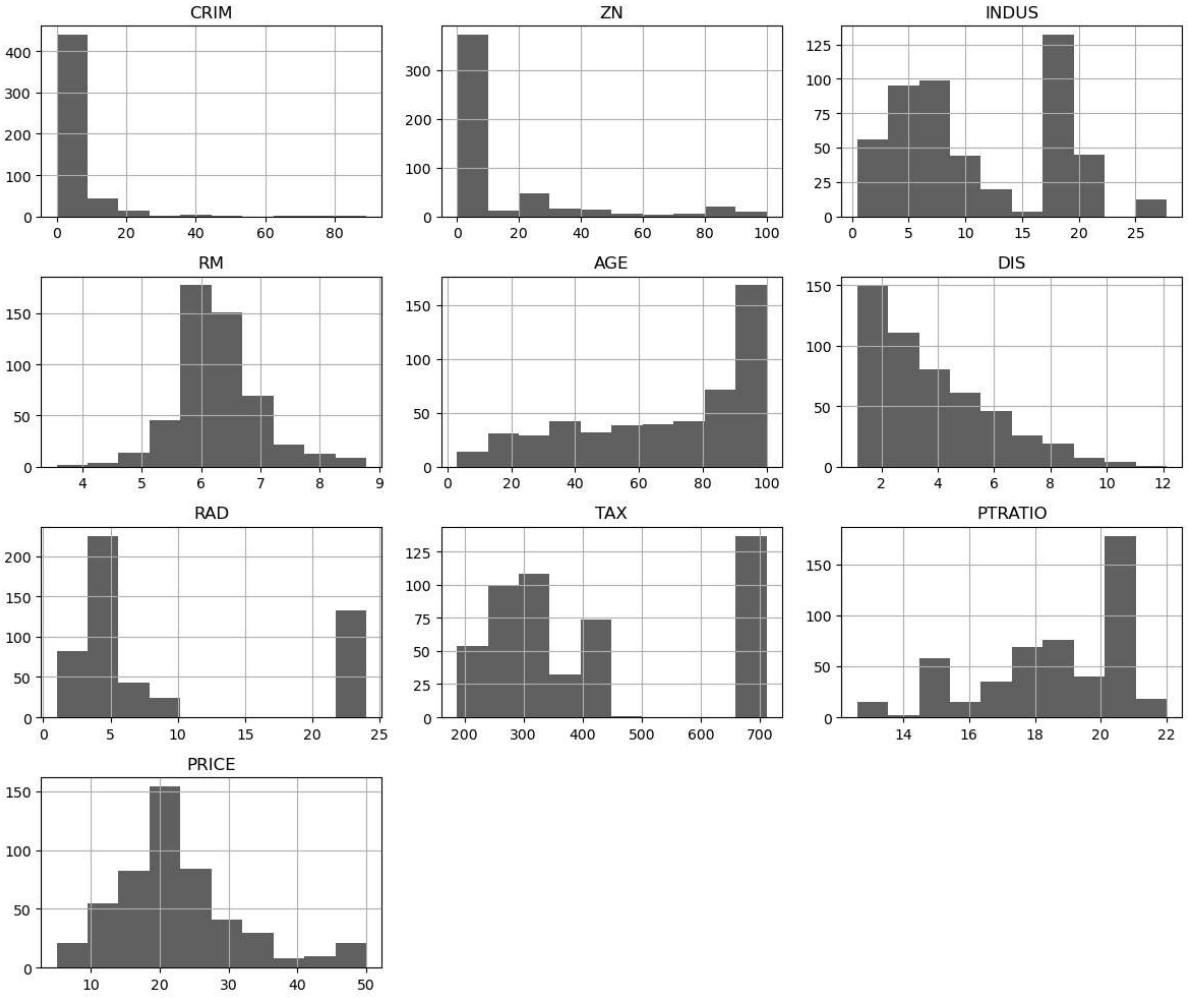
	CRIM	ZN	INDUS	RM	AGE	DIS	RAD	TAX	PTRATIO	PRICE
0	0.00632	18.0	2.31	6.575	65.2	4.0900	1	296	15.3	24.0
1	0.02731	0.0	7.07	6.421	78.9	4.9671	2	242	17.8	21.6
2	0.02729	0.0	7.07	7.185	61.1	4.9671	2	242	17.8	34.7
3	0.03237	0.0	2.18	6.998	45.8	6.0622	3	222	18.7	33.4
4	0.06905	0.0	2.18	7.147	54.2	6.0622	3	222	18.7	36.2
...	...	...	...	...	...	...	...	...	...	...
501	0.06263	0.0	11.93	6.593	69.1	2.4786	1	273	21.0	22.4
502	0.04527	0.0	11.93	6.120	76.7	2.2875	1	273	21.0	20.6
503	0.06076	0.0	11.93	6.976	91.0	2.1675	1	273	21.0	23.9
504	0.10959	0.0	11.93	6.794	89.3	2.3889	1	273	21.0	22.0
505	0.04741	0.0	11.93	6.030	80.8	2.5050	1	273	21.0	11.9

506 rows × 10 columns

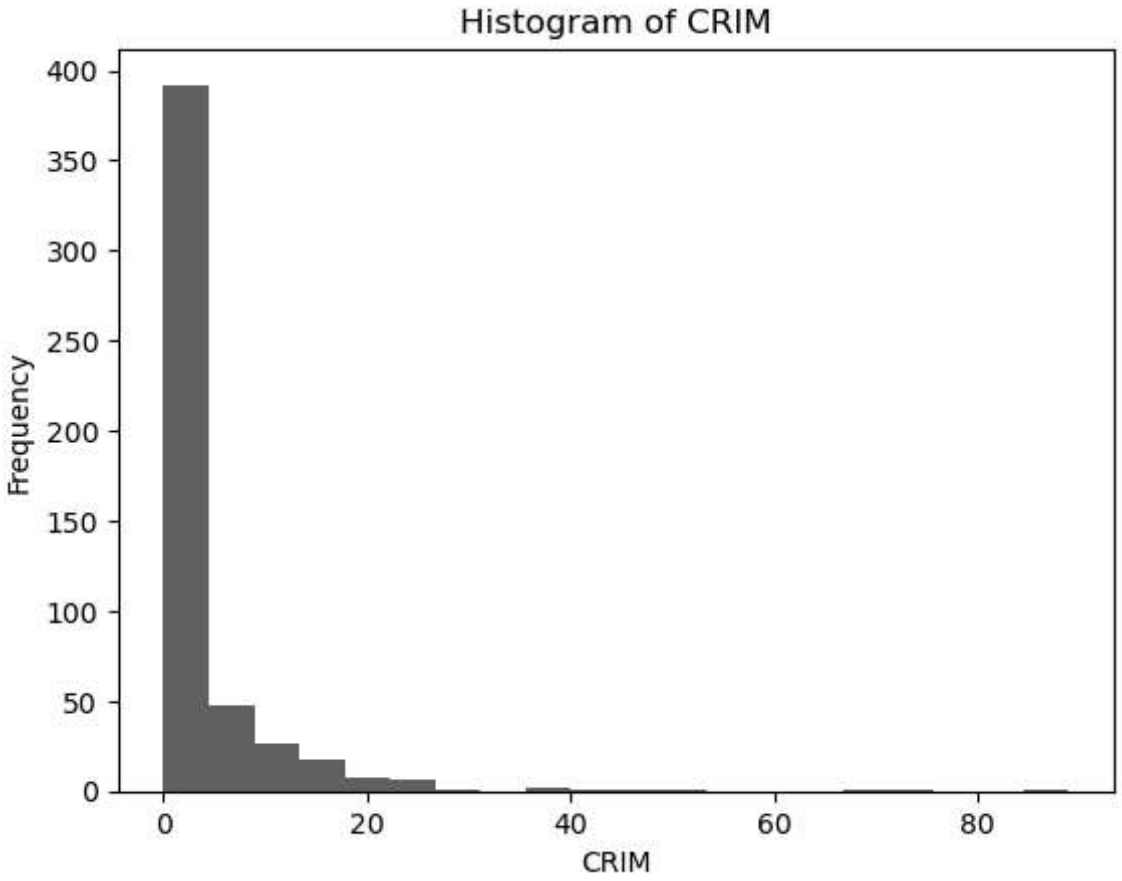
In [8]:
```
# Step 5: Check the last seven records of the new DataFrame
print("\nLast seven records of the new DataFrame:")
print(smaller_df.tail(7))
```

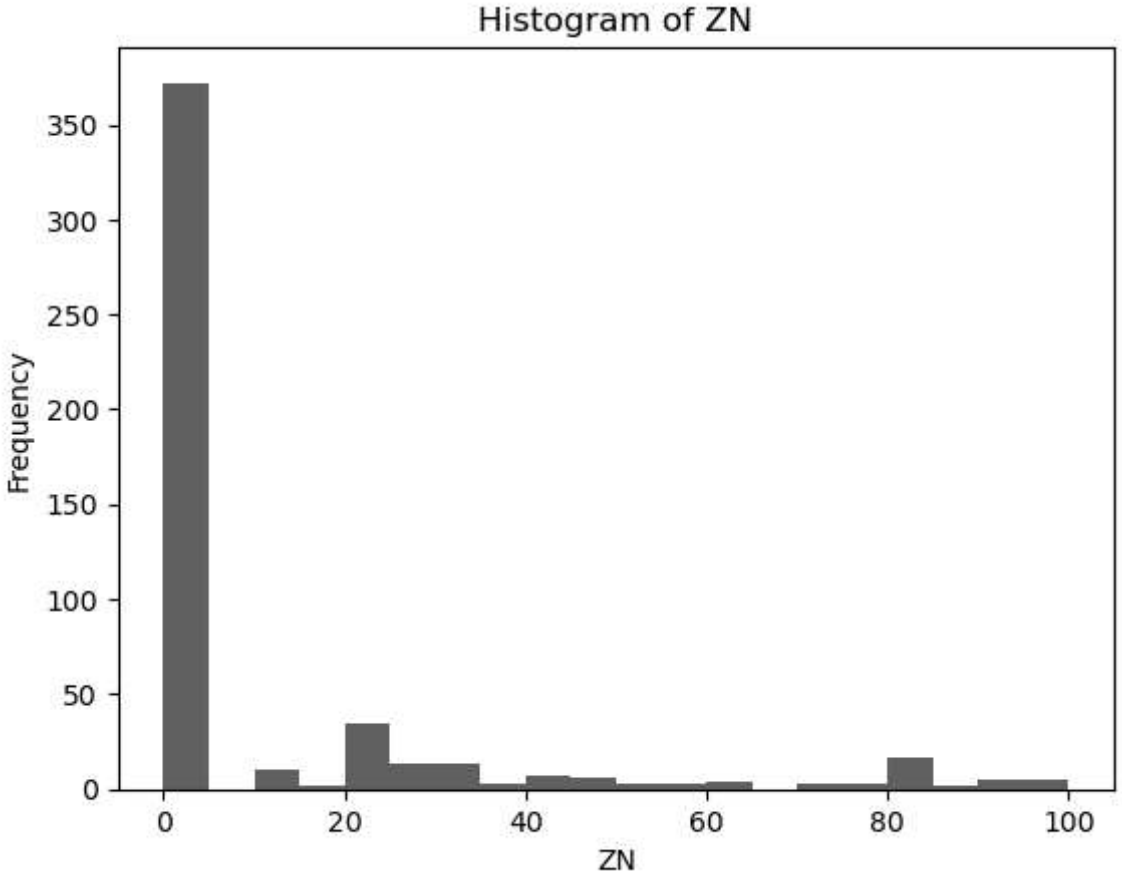
```
Last seven records of the new DataFrame:
        CRIM   ZN  INDUS     RM   AGE     DIS  RAD  TAX  PTRATIO  PRICE
499  0.17783  0.0   9.69  5.569  73.5  2.3999    6  391     19.2   17.5
500  0.22438  0.0   9.69  6.027  79.7  2.4982    6  391     19.2   16.8
501  0.06263  0.0  11.93  6.593  69.1  2.4786    1  273     21.0   22.4
502  0.04527  0.0  11.93  6.120  76.7  2.2875    1  273     21.0   20.6
503  0.06076  0.0  11.93  6.976  91.0  2.1675    1  273     21.0   23.9
504  0.10959  0.0  11.93  6.794  89.3  2.3889    1  273     21.0   22.0
505  0.04741  0.0  11.93  6.030  80.8  2.5050    1  273     21.0   11.9
```

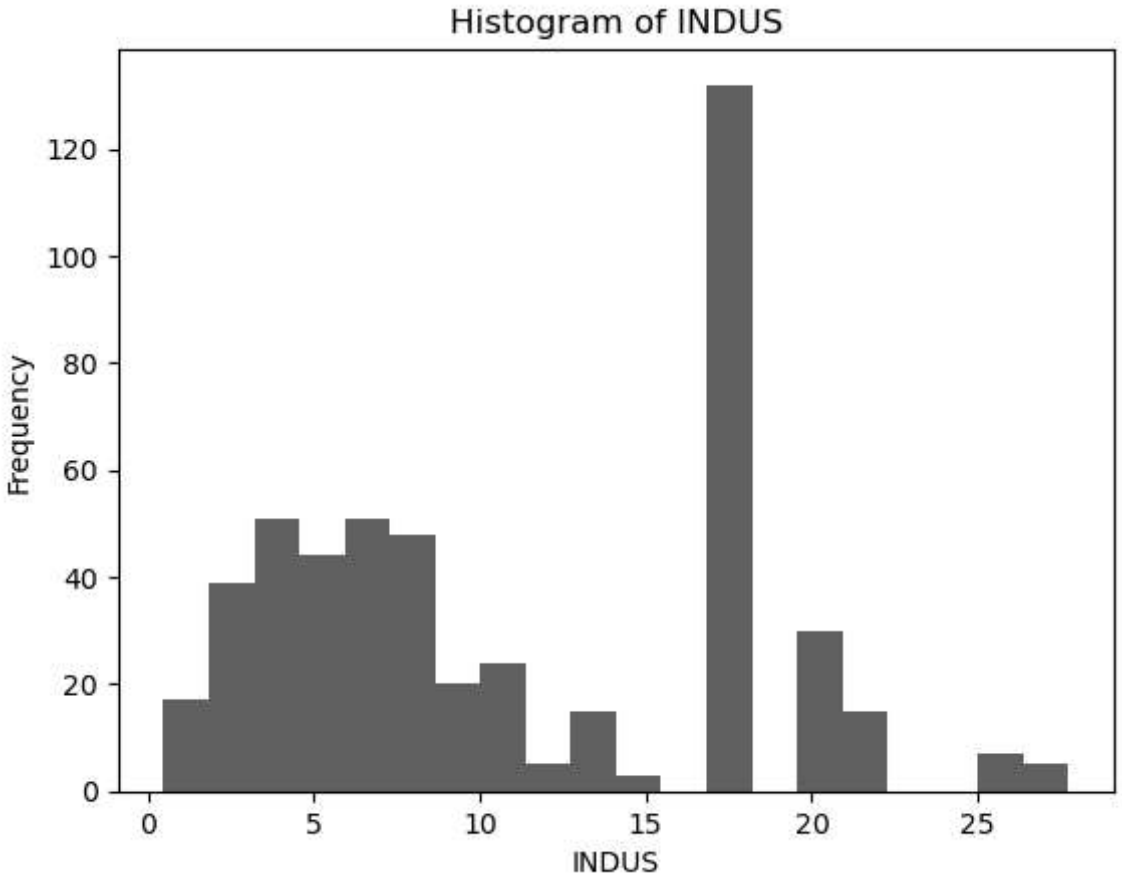
In [9]:
```
# Step 6: Plot histograms of all variables in the new DataFrame
smaller_df.hist(figsize=(12, 10))
plt.tight_layout()
plt.show()
```

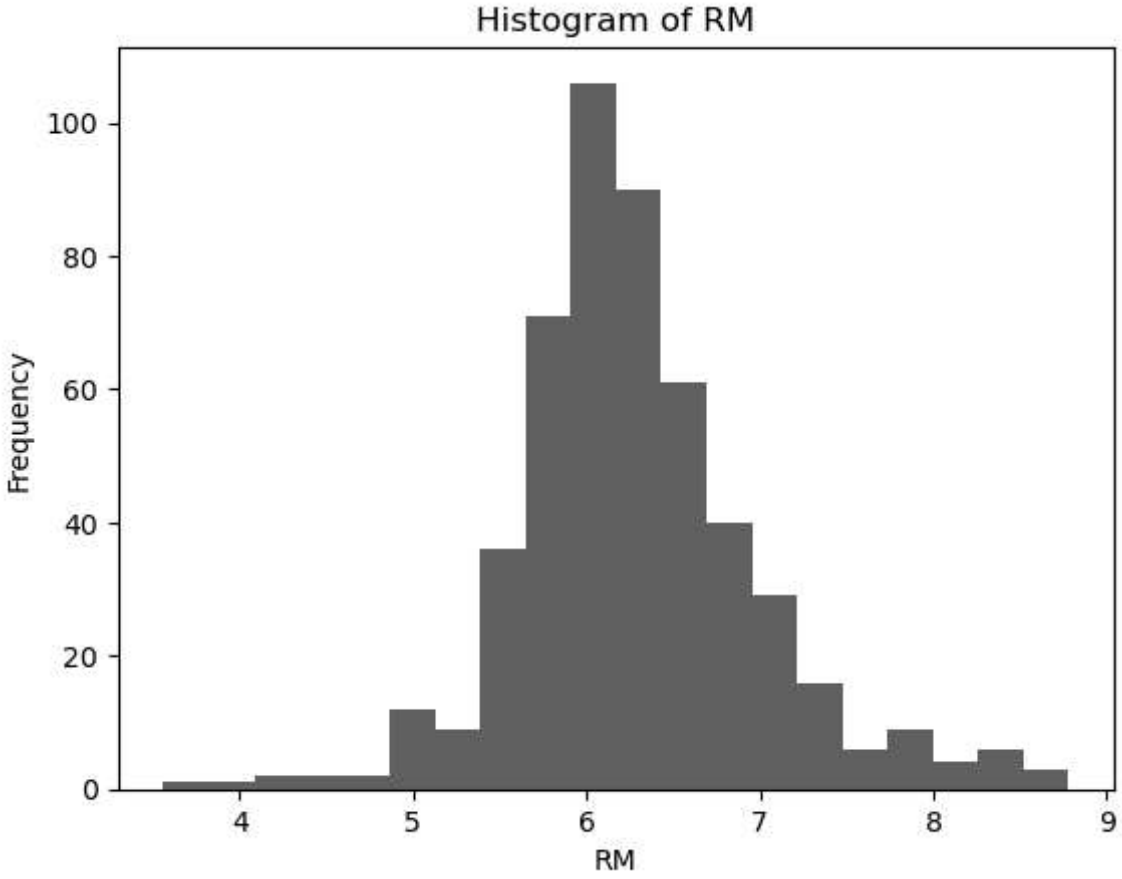


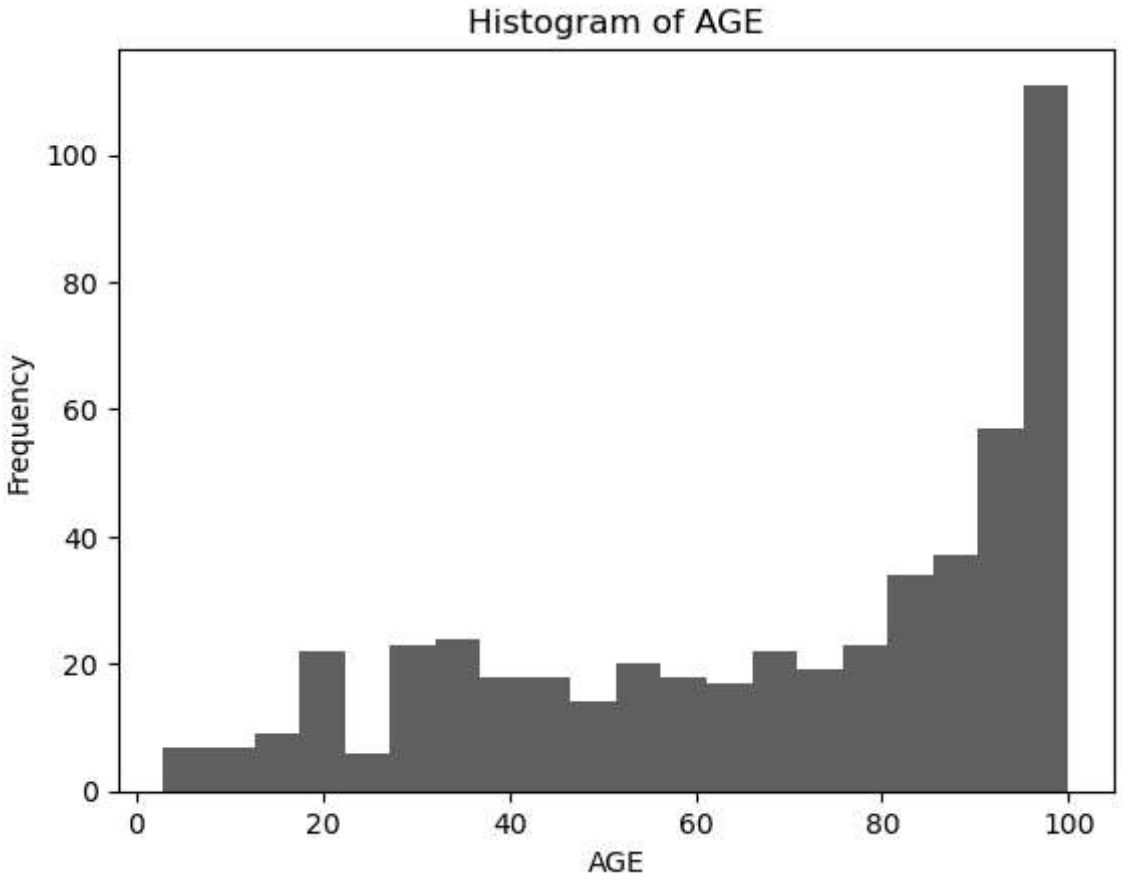
In [10]:
```
# Step 7: Plot histograms of all variables using a for loop
for column in smaller_df.columns:
    plt.hist(smaller_df[column], bins=20)
    plt.title(f'Histogram of {column}')
    plt.xlabel(column)
    plt.ylabel('Frequency')
    plt.show()
```

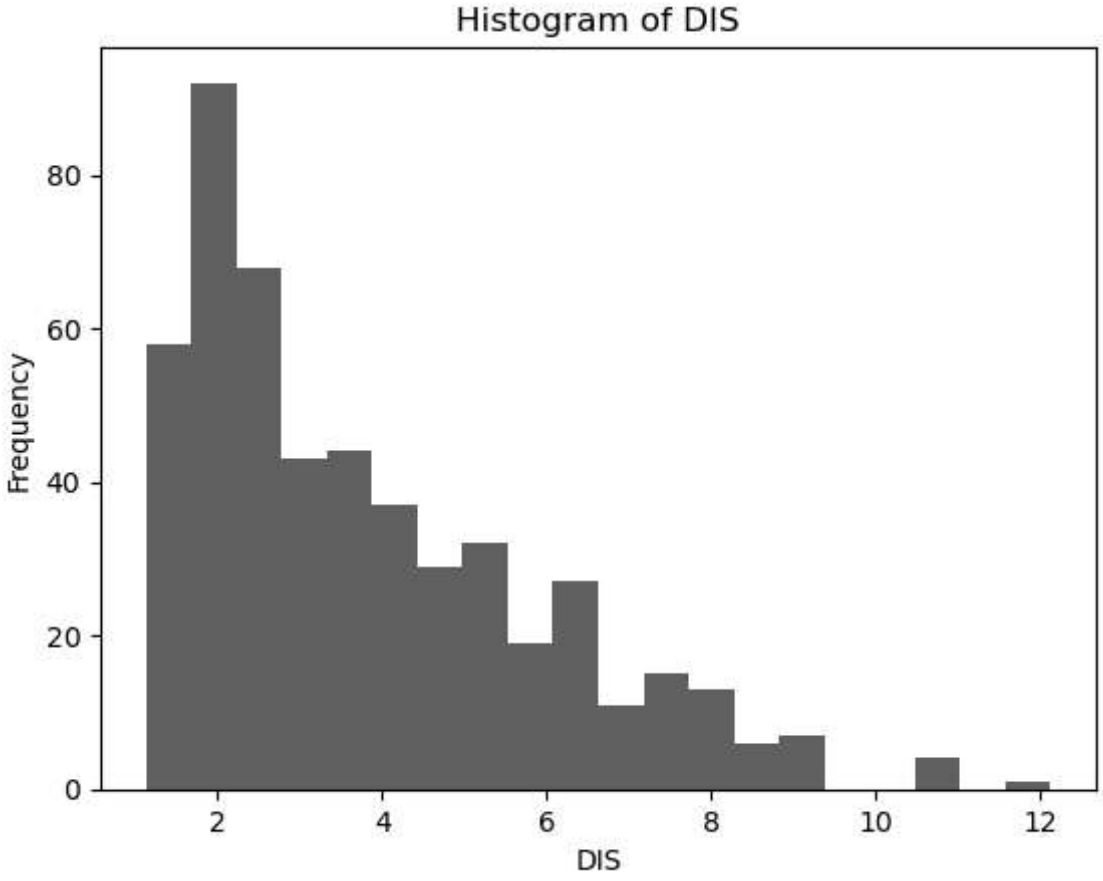

Histogram of CRIM
Frequency
CRIM
0
50
100
150
200
250
300
350
400
0
20
40
60
80

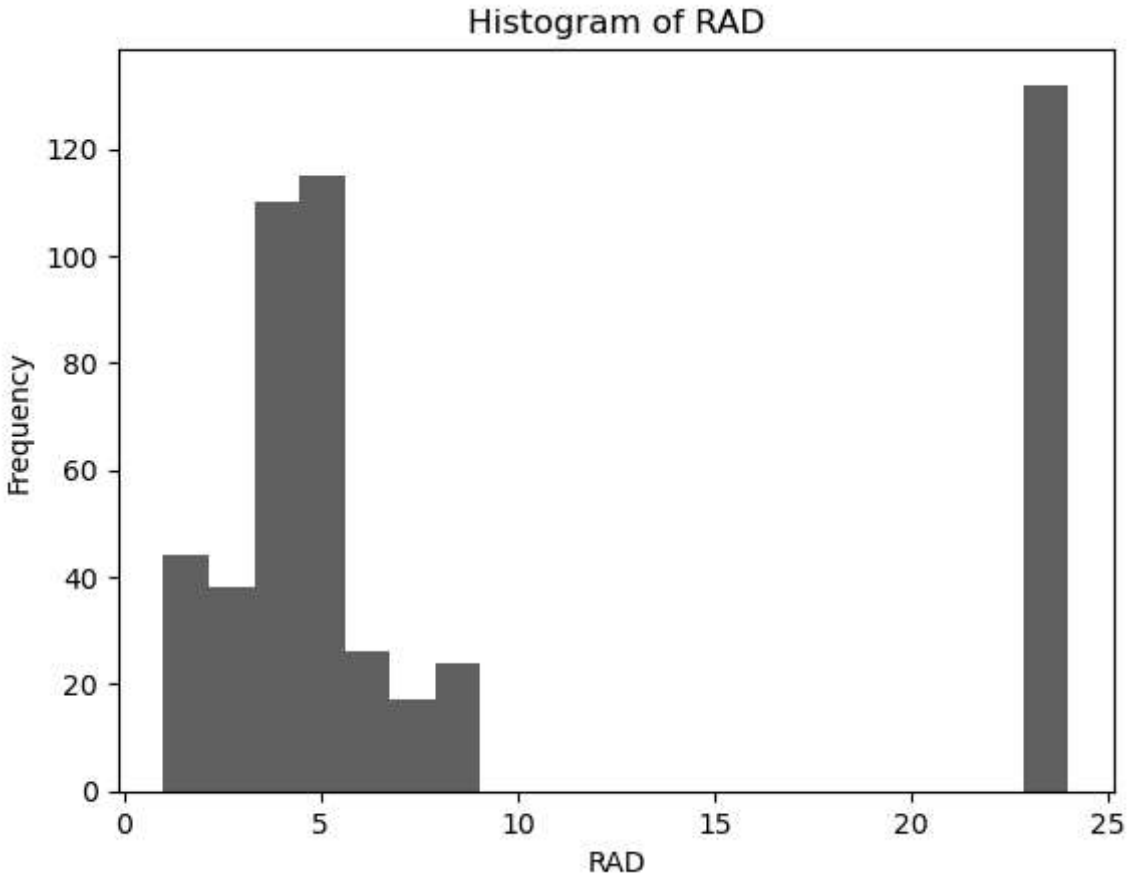

Histogram of ZN
Frequency
ZN
0
50
100
150
200
250
300
350
0
20
40
60
80
100

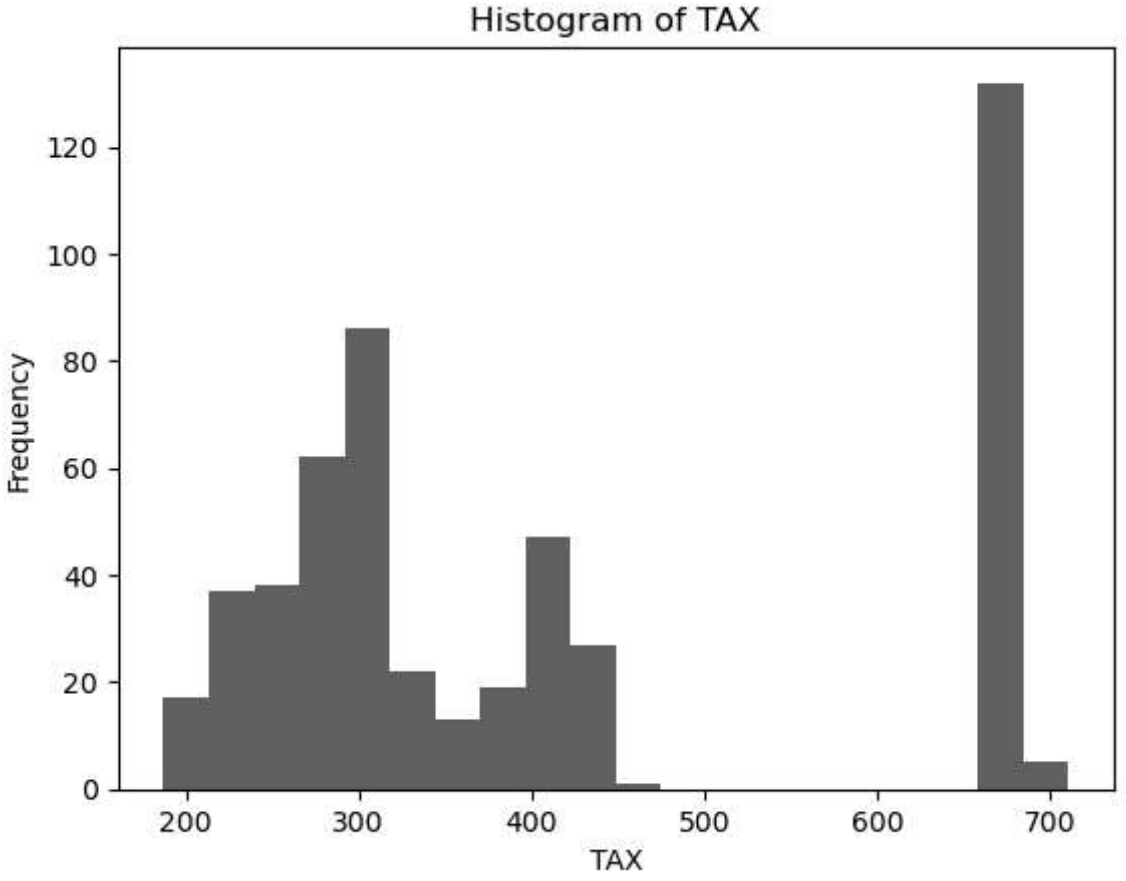

Histogram of INDUS
Frequency
INDUS
0
20
40
60
80
100
120
0
5
10
15
20
25

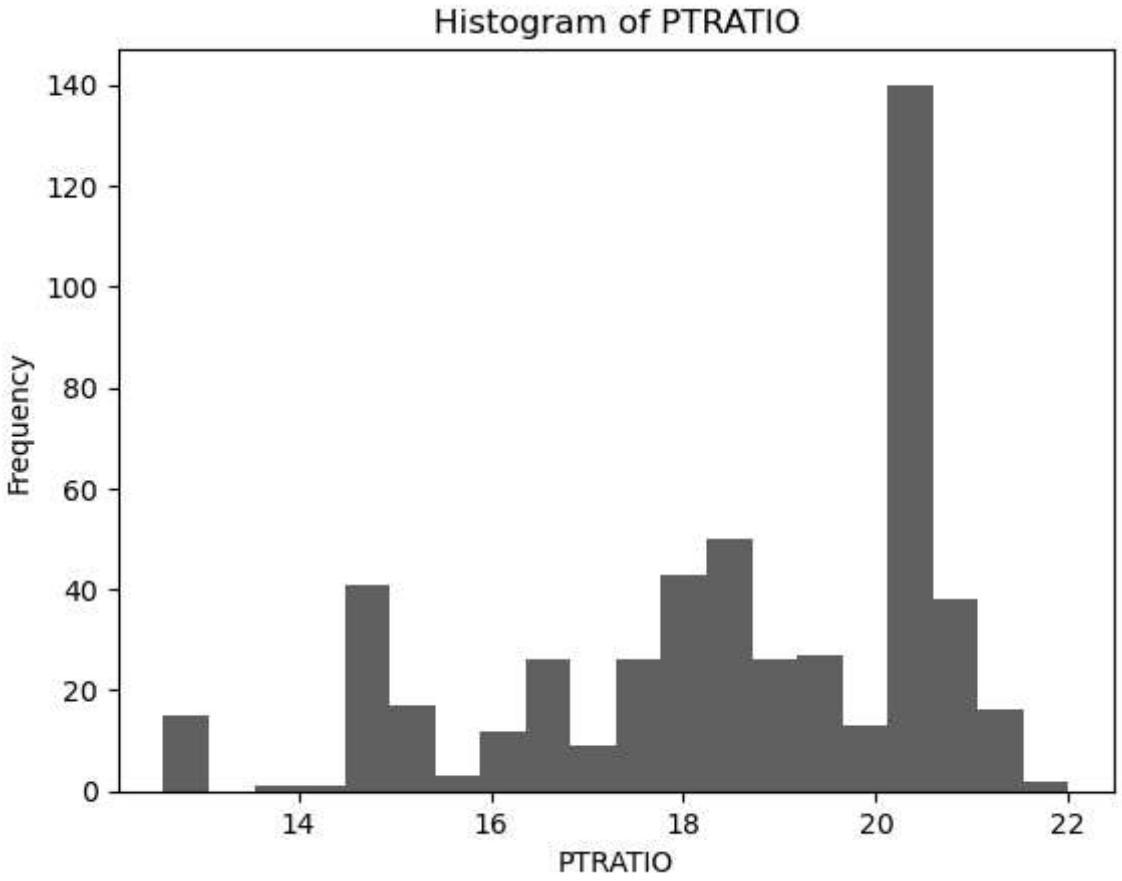

Histogram of RM
Frequency
RM
0
20
40
60
80
100
4
5
6
7
8
9

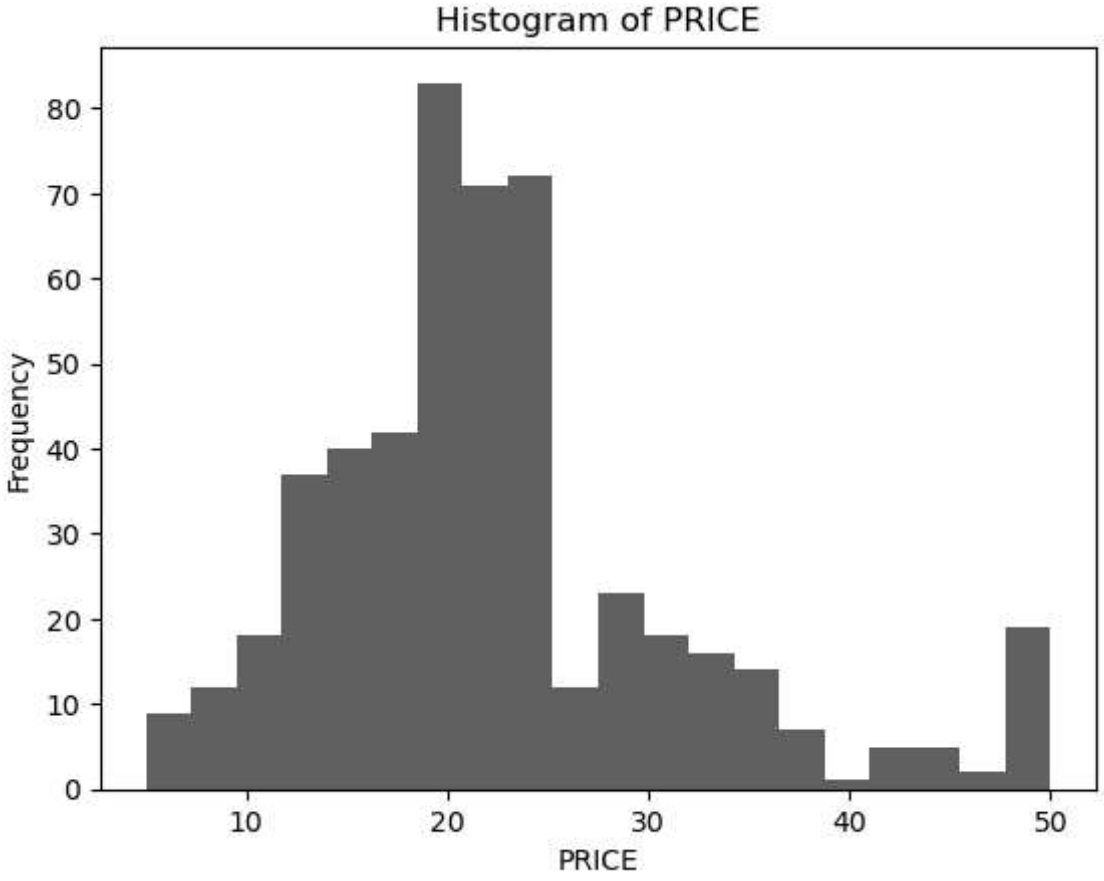

Histogram of AGE
Frequency
AGE
0
20
40
60
80
100


Histogram of DIS
Frequency
DIS
0
20
40
60
80
2
4
6
8
10
12

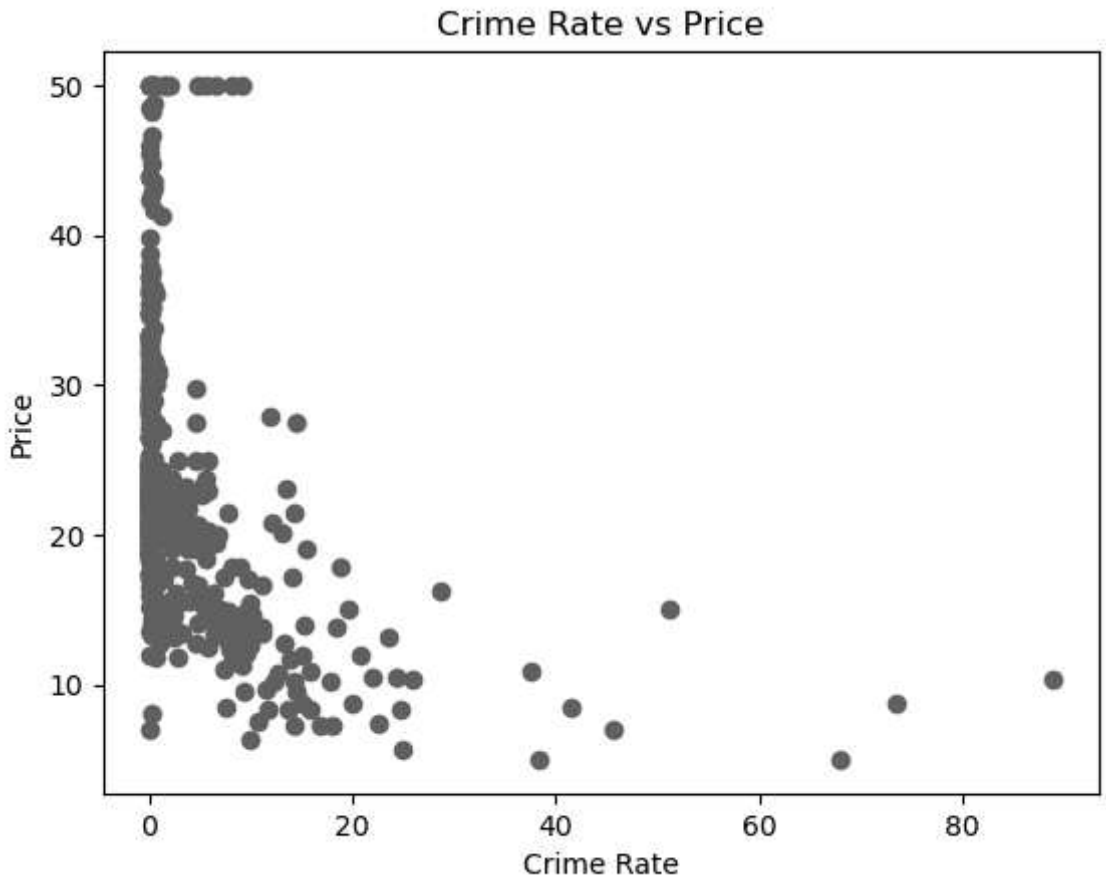

Histogram of RAD
Frequency
RAD
0
20
40
60
80
100
120
0
5
10
15
20
25


Histogram of TAX
Frequency
TAX
0
20
40
60
80
100
120
200
300
400
500
600
700

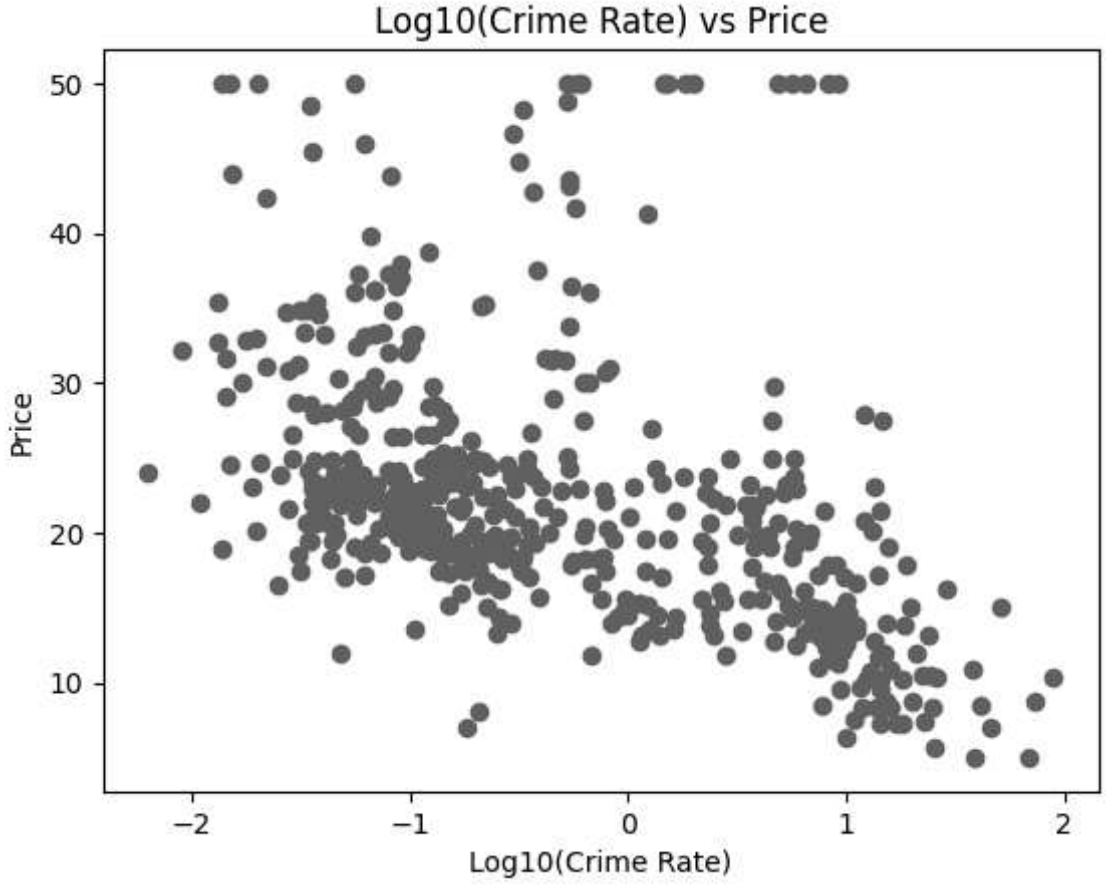




In [11]:
```
# Step 8: Create a scatter plot of crime rate versus price
plt.scatter(housing_data['CRIM'], housing_data['PRICE'])
plt.title('Crime Rate vs Price')
plt.xlabel('Crime Rate')
plt.ylabel('Price')
plt.show()
```



In [ ]:
```
# Step 9: Plot log10(crime) versus price
plt.scatter(np.log10(housing_data['CRIM']), housing_data['PRICE'])
plt.title('Log10(Crime Rate) vs Price')
plt.xlabel('Log10(Crime Rate)')
plt.ylabel('Price')
plt.show()
```



In [12]:
```
# Step 10: Calculate and print some useful statistics
mean_rooms_per_dwelling = smaller_df['RM'].mean()
median_age = smaller_df['AGE'].median()
mean_distance_to_employment_centers = smaller_df['DIS'].mean()
percentage_low_price = (housing_data['PRICE'] < 20).mean() * 100

print("\nMean Rooms per Dwelling:", mean_rooms_per_dwelling)
print("Median Age:", median_age)
print("Mean Distance to Employment Centers:", mean_distance_to_employment_centers)
print("Percentage of Houses with Low Price (< $20,000):", percentage_low_price)
```

```
Mean Rooms per Dwelling: 6.284634387351787
Median Age: 77.5
Mean Distance to Employment Centers: 3.795042687747034
Percentage of Houses with Low Price (< $20,000): 41.50197628458498
```

Discussion Activity 3.01

The primary objective of this analysis is to explore the Boston housing dataset and extract meaningful insights regarding the factors affecting housing prices.

Specifically, the aims are:

Investigate the distribution of different features in the dataset. Identify correlations between variables, particularly with respect to housing prices. Calculate descriptive statistics to understand the central tendencies and variability of key features. Explore the impact of crime rate on property prices. Assess the proportion of houses with prices below $20,000.

I begin by loading the dataset into a pandas DataFrame and conducting preliminary data inspection. This involves checking the first few records, the data types of variables, and identifying any missing values. Subsequently, I performed data wrangling by excluding certain columns as specified in the problem statement.

Following data preprocessing, I visualized the distribution of variables through histograms and explore pairwise relationships using scatter plots. I then calculate descriptive statistics such as mean rooms per dwelling, median age, mean distances to employment centers, and the percentage of houses with low prices.

The analysis reveals several key findings. Firstly, the distribution of housing prices exhibits a right-skewed pattern, indicating that a majority of properties have lower prices. Secondly, there exists a strong positive correlation between the number of rooms per dwelling and property prices, suggesting that larger houses tend to be more expensive. Thirdly, the scatter plot depicting crime rate versus price illustrates a negative relationship, indicating that areas with higher crime rates tend to have lower property values. Finally, the calculated statistics provide insights into the average characteristics of houses in the Boston area.

Activity 4.01

In [109…
```
# Step 1: Load the necessary libraries
import pandas as pd
import matplotlib.pyplot as plt
```

In [110…
```
# Step 2: Read the adult income dataset

income_data = pd.read_csv('adult_income_dataset.csv')
income_data.head()
```

Out[110]:

	39	State-gov	77516	Bachelors	13	Never-married	Adm-clerical	Not-in-family	Male	2174	0	40	Unite Stat
0	50	Self-emp-not-inc	83311	Bachelors	13	Married-civ-spouse	Exec-managerial	Husband	Male	0	0	13	Unite Stat
1	38	Private	215646	HS-grad	9	Divorced	Handlers-cleaners	Not-in-family	Male	0	0	40	Unite Stat
2	53	Private	234721	11th	7	Married-civ-spouse	Handlers-cleaners	Husband	Male	0	0	40	Unite Stat
3	28	Private	338409	Bachelors	13	Married-civ-spouse	Prof-specialty	Wife	Female	0	0	40	Cul
4	37	Private	284582	Masters	14	Married-civ-spouse	Exec-managerial	Wife	Female	0	0	40	Unite Stat

In [111…
```
# Step 3: Create a script that will read a text file line by line
def read_file_line_by_line(file_path):
    with open(file_path, 'r') as file:
        for line in file:
            print(line)
```

In [112…
```
names = []
with open('adult_income_names.txt','r') as f:
    for line in f:
        f.readline()
        var=line.split(":")[0]
        names.append(var)
names
```

Out[112]:
```
['age',
 'workclass',
 'fnlwgt',
 'education',
 'education-num',
 'marital-status',
 'occupation',
 'relationship',
 'sex',
 'capital-gain',
 'capital-loss',
 'hours-per-week',
 'native-country']
```

In [113…
```
names.append('income')
```

In [116…
```
income_data = pd.read_csv("adult_income_dataset.csv",names=names)
income_data.head()
```

Out[116]:

	age	workclass	fnlwgt	education	education-num	marital-status	occupation	relationship	sex	cap
0	39	State-gov	77516	Bachelors	13	Never-married	Adm-clerical	Not-in-family	Male	
1	50	Self-emp-not-inc	83311	Bachelors	13	Married-civ-spouse	Exec-managerial	Husband	Male	
2	38	Private	215646	HS-grad	9	Divorced	Handlers-cleaners	Not-in-family	Male	
3	53	Private	234721	11th	7	Married-civ-spouse	Handlers-cleaners	Husband	Male	
4	28	Private	338409	Bachelors	13	Married-civ-spouse	Prof-specialty	Wife	Female	

In [122…

```
# Create a copy of the DataFrame

# Read the CSV file into a DataFrame
income_data = pd.read_csv("adult_income_dataset.csv", names=names)

# Create a copy
income_data_headed = income_data.copy()
```

In [121…

```
income_data_headed.head()
```

Out[121]:

	age	workclass	fnlwgt	education	education-num	marital-status	occupation	relationship	sex	cap
0	39	State-gov	77516	Bachelors	13	Never-married	Adm-clerical	Not-in-family	Male	
1	50	Self-emp-not-inc	83311	Bachelors	13	Married-civ-spouse	Exec-managerial	Husband	Male	
2	38	Private	215646	HS-grad	9	Divorced	Handlers-cleaners	Not-in-family	Male	
3	53	Private	234721	11th	7	Married-civ-spouse	Handlers-cleaners	Husband	Male	
4	28	Private	338409	Bachelors	13	Married-civ-spouse	Prof-specialty	Wife	Female	

In [175…

```
# Step 5: Find the missing values
missing_values = income_data.isnull().sum()
print("Missing Values:")
print(missing_values)
```

```
Missing Values:
age               0
workclass         0
fnlwgt            0
education         0
education-num     0
marital-status    0
occupation        0
relationship      0
sex               0
capital-gain      0
capital-loss      0
hours-per-week    0
native-country    0
income            0
dtype: int64
```

In [176…

```
# Step 6: Create a DataFrame with only age, education, and occupation by using subs
subset_data = income_data[['age', 'education', 'occupation']]
subset_data
```

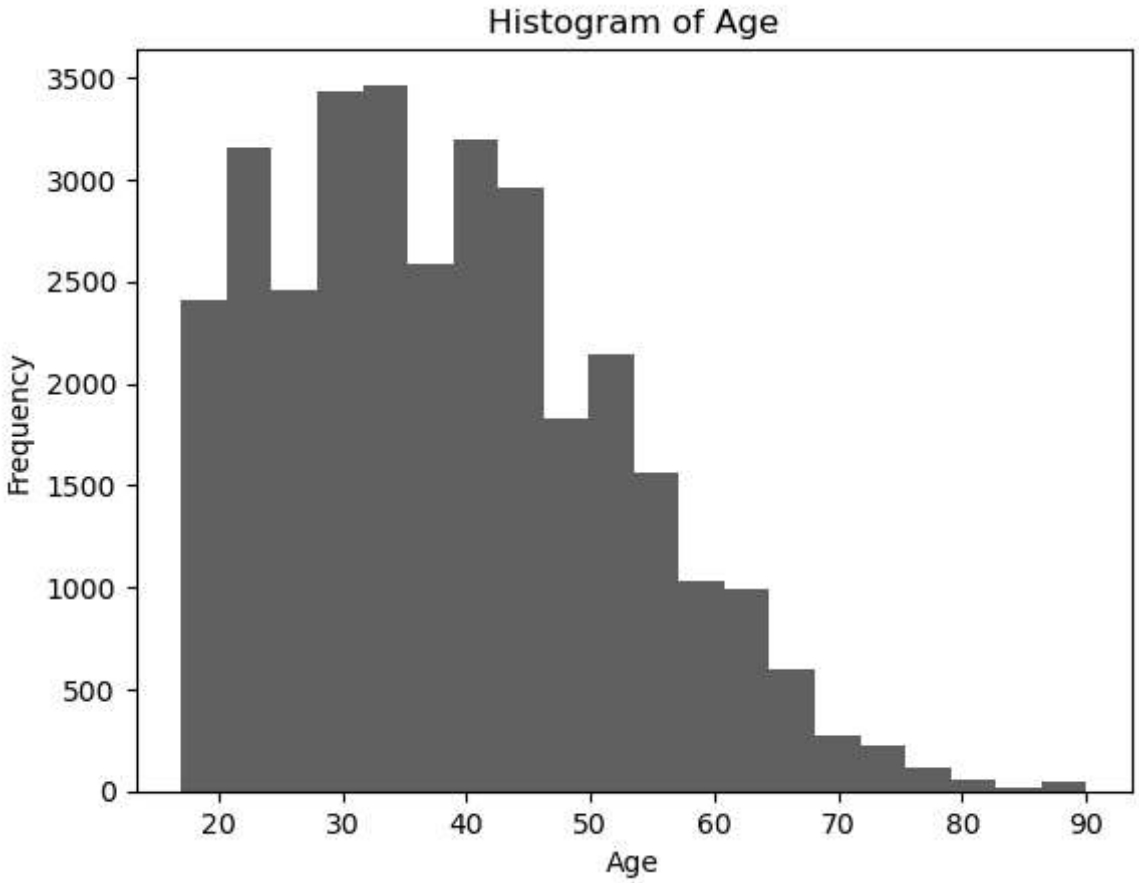
Out[176]:

	age	education	occupation
0	39	Bachelors	Adm-clerical
1	50	Bachelors	Exec-managerial
2	38	HS-grad	Handlers-cleaners
3	53	11th	Handlers-cleaners
4	28	Bachelors	Prof-specialty
...	...	...	...
32556	27	Assoc-acdm	Tech-support
32557	40	HS-grad	Machine-op-inspct
32558	58	HS-grad	Adm-clerical
32559	22	HS-grad	Adm-clerical
32560	52	HS-grad	Exec-managerial

32561 rows × 3 columns

In [177…

```
# Step 7: Plot a histogram of age with a bin size of 20
plt.hist(income_data['age'], bins=20)
plt.title('Histogram of Age')
plt.xlabel('Age')
plt.ylabel('Frequency')
plt.show()
```



In [178…
```
# Step 8: Create a function to strip the whitespace characters
def strip_whitespace(x):
    if isinstance(x, str):
        return x.strip()
    else:
        return x
```

In [180…
```
# Step 9: Use the apply method to apply this function to all the columns with strir
income_data = income_data.applymap(strip_whitespace)
```

In [181…
```
# Step 10: Find the number of people who are aged between 30 and 50

import pandas as pd

# convert the 'age' column to integers
income_data['age'] = income_data['age'].astype(int)

# Filter the DataFrame based on age
people_between_30_50 = income_data[(income_data['age'] >= 30) & (income_data['age']

# Get the number of people aged between 30 and 50
num_people_between_30_50 = len(people_between_30_50)
print("Number of people aged between 30 and 50:", num_people_between_30_50)
```

```
Number of people aged between 30 and 50: 16390
```

In [182…
```
# Step 11: Group the records based on age and education to find how the mean age is
age_education_mean = income_data.groupby(['age', 'education']).mean()
age_education_mean
```

Out[182]:

age	education	fnlwgt	education-num	capital-gain	capital-loss	hours-per-week
17	**10th**	187111.630435	6.0	266.659420	46.434783	21.543478
	11th	188696.555556	7.0	30.411111	36.911111	19.927778
	12th	178750.702703	8.0	0.000000	46.513514	20.189189
	5th-6th	270942.000000	3.0	0.000000	0.000000	48.000000
	7th-8th	127804.000000	4.0	0.000000	0.000000	31.000000
...	**...**	...	...	...	...	...
90	**Bachelors**	165312.333333	13.0	2327.000000	0.000000	31.666667
	HS-grad	132201.285714	9.0	1394.142857	468.714286	37.428571
	Masters	150378.250000	14.0	5012.750000	0.000000	47.500000
	Prof-school	87372.000000	15.0	20051.000000	0.000000	72.000000
	Some-college	153924.333333	10.0	0.000000	0.000000	32.833333

965 rows × 5 columns

In [ ]:

In [172…
```
# Step 12: Group by occupation and show the summary statistics of age

# Remove rows where the occupation is "?"
income_data_cleaned = income_data[income_data['occupation'] != "?"]

# Group by occupation and show the summary statistics of age
occupation_summary_stats = income_data_cleaned.groupby('occupation')['age'].describ

# Check if there are any occupations with data beyond the 75th percentile
if '75%' in occupation_summary_stats:
    highest_percentile_profession = occupation_summary_stats[occupation_summary_sta
    print("Profession with the largest share of the workforce above the 75th percer
else:
    print("There is no occupation with data beyond the 75th percentile.")
```

```
Profession with the largest share of the workforce above the 75th percentile: Priv
-house-serv
```

In [ ]:
```
N.B. After the '?' removed from the occupation column, the dataset 'income_data' wa
```

In [173…
```
# Step 12: Group by occupation and show the summary statistics of age
occupation_summary_stats = income_data_cleaned.groupby('occupation')['age'].describ
oldest_profession = occupation_summary_stats[occupation_summary_stats['mean'] == oc
print("Profession with the oldest workers on average:", oldest_profession)
highest_percentile_profession = occupation_summary_stats[occupation_summary_stats['
print("Profession with the largest share of the workforce above the 75th percentile
```

```
Profession with the oldest workers on average: Exec-managerial
Profession with the largest share of the workforce above the 75th percentile: Priv
-house-serv
```

In [183…
```
# Summary statistics of age

occupation_summary_stats
```

Out[183]:

occupation	count	mean	std	min	25%	50%	75%	max
Adm-clerical	3770.0	36.964456	13.362998	17.0	26.0	35.0	46.0	90.0
Armed-Forces	9.0	30.222222	8.089774	23.0	24.0	29.0	34.0	46.0
Craft-repair	4099.0	39.031471	11.606436	17.0	30.0	38.0	47.0	90.0
Exec-managerial	4066.0	42.169208	11.974548	17.0	33.0	41.0	50.0	90.0
Farming-fishing	994.0	41.211268	15.070283	17.0	29.0	39.0	52.0	90.0
Handlers-cleaners	1370.0	32.165693	12.372635	17.0	23.0	29.0	39.0	90.0
Machine-op-inspct	2002.0	37.715285	12.068266	17.0	28.0	36.0	46.0	90.0
Other-service	3295.0	34.949621	14.521508	17.0	22.0	32.0	45.0	90.0
Priv-house-serv	149.0	41.724832	18.633688	17.0	24.0	40.0	57.0	81.0
Prof-specialty	4140.0	40.517633	12.016676	17.0	31.0	40.0	48.0	90.0
Protective-serv	649.0	38.953775	12.822062	17.0	29.0	36.0	47.0	90.0
Sales	3650.0	37.353973	14.186352	17.0	25.0	35.0	47.0	90.0
Tech-support	928.0	37.022629	11.316594	17.0	28.0	36.0	44.0	73.0
Transport-moving	1597.0	40.197871	12.450792	17.0	30.0	39.0	49.0	90.0

In [184…

```
# Step 13: Use subset and groupby to find outliers
outliers = income_data_cleaned.groupby('occupation').apply(lambda x: x[(x['age'] <
outliers
```

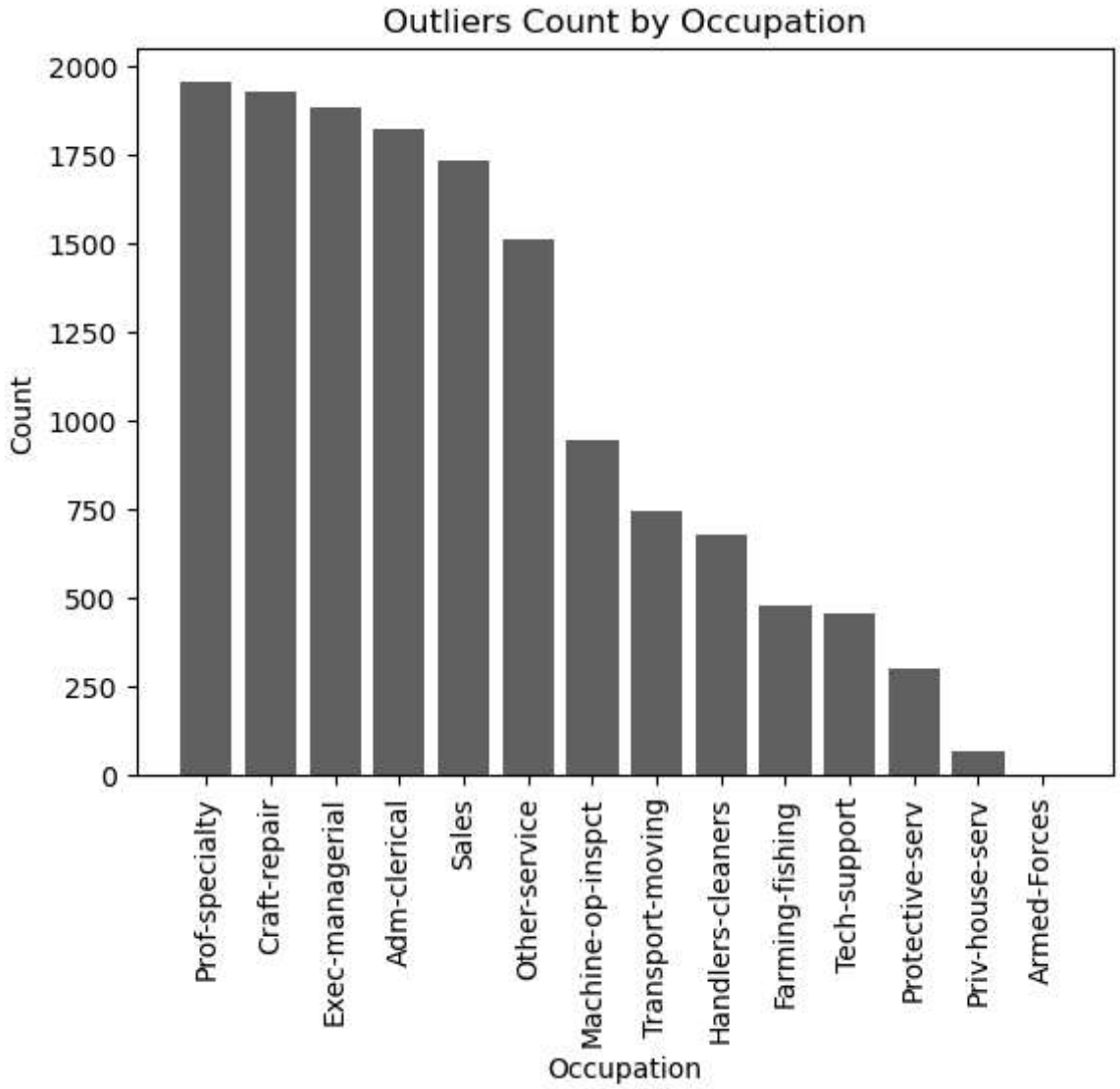
Out[184]:

occupation		age	workclass	fnlwgt	education	education-num	marital-status	occupation	relatio
Adm-clerical	**12**	23	Private	122272	Bachelors	13	Never-married	Adm-clerical	Own-
	37	19	Private	544091	HS-grad	9	Married-AF-spouse	Adm-clerical	
	43	49	Private	94638	HS-grad	9	Separated	Adm-clerical	Unma
	67	53	Private	169846	HS-grad	9	Married-civ-spouse	Adm-clerical	
	126	20	Private	111697	Some-college	10	Never-married	Adm-clerical	Own-
...	**...**	...	...	...	...	...	...	...	
Transport-moving	**32289**	24	Private	34568	Assoc-voc	11	Never-married	Transport-moving	N f
	32377	28	Private	180299	HS-grad	9	Married-civ-spouse	Transport-moving	Hus
	32435	25	Private	175128	HS-grad	9	Married-civ-spouse	Transport-moving	Hus
	32441	53	Private	304504	Some-college	10	Married-civ-spouse	Transport-moving	Hus
	32498	57	Private	153918	HS-grad	9	Married-civ-spouse	Transport-moving	Hus

14518 rows × 14 columns

In [185…

```
# Step 14: Plot the values on a bar chart
outliers_count = outliers['occupation'].value_counts()
plt.bar(outliers_count.index, outliers_count.values)
plt.title('Outliers Count by Occupation')
plt.xlabel('Occupation')
plt.ylabel('Count')
plt.xticks(rotation=90)
plt.show()
```



In [186…

```
# Step 15: Merge the data using common keys

import pandas as pd

# Summary statistics DataFrame
occupation_summary_stats = income_data_cleaned.groupby('occupation')['age'].describ

# Merge the data using common keys (occupation)
merged_data = pd.merge(income_data_cleaned, occupation_summary_stats, on='occupatio

# Print the merged DataFrame
print(merged_data)
```

```
       age     workclass  fnlwgt      education  education-num  \
0       39     State-gov   77516      Bachelors             13
1       23       Private  122272      Bachelors             13
2       30   Federal-gov   59951   Some-college             10
3       19       Private  544091        HS-grad              9
4       49       Private   94638        HS-grad              9
...    ...           ...     ...            ...            ...
30713   17       Private   79682           10th              6
30714   27       Private  363053            9th              5
30715   24       Private  213902        7th-8th              4
30716   19       Private  188568   Some-college             10
30717   17       Private  141590           11th              7

           marital-status       occupation     relationship      sex  \
0           Never-married     Adm-clerical    Not-in-family     Male
1           Never-married     Adm-clerical        Own-child   Female
2      Married-civ-spouse     Adm-clerical        Own-child     Male
3       Married-AF-spouse     Adm-clerical             Wife   Female
4               Separated     Adm-clerical        Unmarried   Female
...                   ...              ...              ...      ...
30713       Never-married  Priv-house-serv   Other-relative     Male
30714       Never-married  Priv-house-serv        Unmarried   Female
30715       Never-married  Priv-house-serv        Own-child   Female
30716       Never-married  Priv-house-serv    Not-in-family   Female
30717       Never-married  Priv-house-serv        Own-child   Female

       capital-gain  ...  native-country  income   count       mean  \
0              2174  ...   United-States   <=50K  3770.0  36.964456
1                 0  ...   United-States   <=50K  3770.0  36.964456
2                 0  ...   United-States   <=50K  3770.0  36.964456
3                 0  ...   United-States   <=50K  3770.0  36.964456
4                 0  ...   United-States   <=50K  3770.0  36.964456
...             ...  ...             ...     ...     ...        ...
30713             0  ...   United-States   <=50K   149.0  41.724832
30714             0  ...          Mexico   <=50K   149.0  41.724832
30715             0  ...     El-Salvador   <=50K   149.0  41.724832
30716             0  ...   United-States   <=50K   149.0  41.724832
30717             0  ...   United-States   <=50K   149.0  41.724832

             std   min   25%   50%   75%   max
0      13.362998  17.0  26.0  35.0  46.0  90.0
1      13.362998  17.0  26.0  35.0  46.0  90.0
2      13.362998  17.0  26.0  35.0  46.0  90.0
3      13.362998  17.0  26.0  35.0  46.0  90.0
4      13.362998  17.0  26.0  35.0  46.0  90.0
...          ...   ...   ...   ...   ...   ...
30713  18.633688  17.0  24.0  40.0  57.0  81.0
30714  18.633688  17.0  24.0  40.0  57.0  81.0
30715  18.633688  17.0  24.0  40.0  57.0  81.0
30716  18.633688  17.0  24.0  40.0  57.0  81.0
30717  18.633688  17.0  24.0  40.0  57.0  81.0

[30718 rows x 22 columns]
```

N.B. The original dataset was merged with the summary statistics DataFrame based on the 'occupation' column, adding summary statistics for each occupation to each corresponding row in the original dataset.

In [ ]:

Discussion Activity 4.01

The primary objective of this analysis is to investigate the factors that influence income levels among adults.

Specifically,the aims are:

Identify demographic attributes associated with higher income levels. Analyze the distribution of income across different occupations. Detect outliers in the dataset and assess their impact on the analysis. Provide insights into potential avenues for further research and exploration.

The analysis begins with data preprocessing, including handling missing values, data cleaning, and feature selection. We then proceed to explore the dataset through descriptive statistics, visualizations, and advanced data wrangling techniques.

Key steps in the analysis include:

Data loading and preprocessing: Reading the dataset from a CSV file, handling missing values, and cleaning the data. Exploratory data analysis: Visualizing the distribution of demographic attributes such as age, education, and occupation. Statistical analysis: Calculating summary statistics, identifying outliers, and assessing their impact on the analysis. Comparative analysis: Comparing income levels across different demographic groups and occupations. Interpretation and discussion: Interpreting the findings, discussing insights gained from the analysis, and proposing future research directions.

The analysis reveals several noteworthy insights into the factors influencing income levels among adults. Education emerges as a significant predictor of income, with higher education levels corresponding to higher median incomes. Additionally, certain occupations, particularly those in managerial and professional fields, tend to command higher salaries. However, the presence of outliers suggests the need for cautious interpretation, as extreme values may distort the overall analysis. Furthermore, age exhibits a complex relationship with income, indicating potential nonlinear effects that merit deeper exploration.

Excercise 3

In [2]:
```
# Import required library
import pandas as pd
```

In [3]:
```
# Creating Series 1
data1 = [7.3, -2.5, 3.4, 1.5]
index1 = ['a', 'c', 'd', 'e']
series1 = pd.Series(data1, index=index1)
```

In [7]:
```
# Creating Series 2
data2 = [-2.1, 3.6, -1.5, 4, 3.1]
index2 = ['a', 'c', 'e', 'f', 'g']
series2 = pd.Series(data2, index=index2)
```

In [8]:
```
# Adding Series 1 and Series 2
addition_result = series1.add(series2, fill_value=0)
print("Addition Result:")
print(addition_result)
```

```
Addition Result:
a    5.2
c    1.1
d    3.4
e    0.0
f    4.0
g    3.1
dtype: float64
```

In [9]:
```
# Subtracting Series 1 from Series 2
subtraction_result = series2.subtract(series1, fill_value=0)
print("\nSubtraction Result:")
print(subtraction_result)
```

```
Subtraction Result:
a   -9.4
c    6.1
d   -3.4
e   -3.0
f    4.0
g    3.1
dtype: float64
```

Discussion Excercise 3

The task at hand involves creating two pandas Series objects, each with specified data and index values, and performing addition and subtraction operations on them. Additionally, the aim was to investigate how the operations handle missing indices and data points, and discuss the implications of these behaviors.

Data Preparation: Two pandas Series objects, Series 1 and Series 2, are created with specified data and index values. Arithmetic Operations: Addition and subtraction operations are performed on Series 1 and Series 2. Handling Missing Values: The operations are executed with the consideration of missing indices or data points using the fill_value=0 argument.

The addition operation combines corresponding elements from both Series, resulting in a new Series with the union of indices from both input Series. Any missing index or data point is treated as zero during the operation. Similarly, the subtraction operation computes the difference between corresponding elements of the Series, also considering missing values as zero.

Short Report on Overall activities

Title: Exploratory Analysis of Diverse Datasets: Insights and Methodologies

Introduction: This comprehensive report integrates findings from three distinct analyses: an exploratory data analysis (EDA) of the Boston Housing Dataset, an analysis of the Adult Income Dataset, and an investigation into arithmetic operations on pandas Series in Python. The aim is to synthesize key insights and methodologies from these analyses to provide a comprehensive understanding of data exploration, analysis, and manipulation techniques.

Statement of the Problem: The overarching objective of this report is to explore diverse datasets, ranging from housing attributes to demographic information and computational operations, to uncover patterns, relationships, and implications for further research and decision-making. Specifically, the analyses aim to:

Explore factors influencing housing prices in the Boston area. Identify demographic and occupational determinants of income levels among adults. Investigate the implementation and implications of basic arithmetic operations on pandas Series.

Methodology: The methodologies employed in each analysis vary based on the nature of the dataset and the objectives of the investigation.

Boston Housing Dataset Analysis: The analysis begins with data loading and preprocessing using pandas. Descriptive statistics, visualizations (such as histograms and scatter plots), and correlation analyses are conducted to explore relationships between variables. The methodology emphasizes data wrangling, visualization, and statistical analysis.

Adult Income Dataset Analysis: Data preprocessing involves handling missing values and feature selection. Descriptive statistics, visualizations, and statistical analysis are employed to identify patterns and relationships between demographic attributes, occupations, and income levels. Comparative analysis is used to compare income levels across different demographic groups and occupations.

Arithmetic Operations on Pandas Series: Two pandas Series objects are created, and addition and subtraction operations are performed. The analysis investigates how these operations handle missing values using the fill_value=0 argument.

Result and Discussion

Boston Housing Dataset: Total number of records: 506 Mean Rooms per Dwelling: 6.28 Median Age: 77.5 Mean Distance to Employment Centers: 3.80 Percentage of Houses with Low Price (< $20,000) : 41.50 Insights from the Boston Housing Dataset provide valuable in formation a . The distribution of housing prices indicates a significant proportion of affordable housi 20,000. Additionally, variables such as the number of rooms per dwelling and distance to employment centers offer insights into factors influencing property prices. Correlations between variables, such as crime rate and property prices, further contribute to understanding housing dynamics in the area.

Adult Income Dataset: Number of people aged between 30 and 50: 16,390 Profession with the oldest workers on average: Exec-managerial Profession with the largest share of the workforce above the 75th percentile: Priv-house-serv Analysis of the Adult Income Dataset highlights the significance of demographic attributes such as age, education, and occupation in determining income levels among adults. With 16,390 people aged between 30 and 50, the dataset provides insights into the demographic composition of the workforce. The identification of professions with the oldest workers on average, such as Exec-managerial, and those with the largest share of the workforce above the 75th percentile, such as Priv-house-serv, offers valuable insights for workforce planning and labor market analysis.

Arithmetic Operations on Pandas Series: The practical implementation of addition and subtraction operations on pandas Series provides a foundational understanding of data manipulation techniques. Considerations for handling missing values, as demonstrated in the arithmetic operations, are crucial for accurate data analysis. These operations serve as building blocks for more complex data manipulations and analyses in diverse domains.

Conclusions: The comprehensive analysis of diverse datasets, including the Boston Housing Dataset and the Adult Income Dataset, offers valuable insights into housing dynamics, workforce demographics, and data manipulation techniques. Understanding factors influencing housing prices and income levels, as well as implementing basic arithmetic operations on pandas Series, contributes to informed decision-making and further research in related domains. The findings from these analyses lay the groundwork for exploring complex relationships within datasets and informing future research directions.

Way Forward: Moving forward, future research could involve more advanced analyses, predictive modeling, and exploration of alternative methodologies. Additionally, continued investigation into the datasets and techniques discussed in this report could lead to deeper insights and practical applications in various fields, including urban planning, socioeconomic research, and data analysis.

In [ ]: