Boston Housing Market

April 7, 2024

0.0.1 Generating Statistics for Boston Housing Market

These steps will help you complete this activity:

1. Load the necessary libraries.
2. Read in the Boston housing dataset (given as a .csv file) from the local directory.
3. Check the first 10 records. Find the total number of records.
4. Create a smaller DataFrame with columns that do not include CHAS, NOX, B, and LSTAT.
5. Check the last seven records of the new DataFrame you just created.
6. Plot the histograms of all the variables (columns) in the new DataFrame.
7. Plot them all at once using a for loop. Try to add a unique title to a plot.
8. Create a scatter plot of crime rate versus price.
9. Plot using log10(crime) versus price.
10. Calculate some useful statistics, such as mean rooms per dwelling, median age, mean distances to five Boston employment centers, and the percentage of houses with a low price (< $20,000).

[2]:
```
# Load the necessary libraries.

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

[5]:
```
# Read in the Boston housing dataset (given as a .csv file) from the local␣
↪directory.

housing_data_df = read_csv("/Users/siddharthabhaumik/Downloads/Boston_housing.
↪csv")

# Checking the first 10 records.

housing_data_df.head(10)
```

[5]:

	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
```

[8]:
```
# Find the total number of records.

housing_data_df.shape
```

[8]:
```
(506, 14)
```

[15]:
```
# Create a smaller DataFrame with columns that do not include CHAS, NOX, B, and␣
 ↪LSTAT.
new_housing_data_df = housing_data_df.drop(columns=['CHAS', 'NOX', 'B',␣
 ↪'LSTAT'])
```

[17]:
```
# Checking the total number of rows and columns for the new dataset
new_housing_data_df.shape

# 4 columns dropped from the dataset
```

[17]:
```
(506, 10)
```

[18]:
```
# Checking the first 10 records of new housing dataset

new_housing_data_df.head(10)

# Columns=['CHAS', 'NOX', 'B', 'LSTAT'] doesn't exist in the new dataset
```

[18]:
```
      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
5	0.02985	0.0	2.18	6.430	58.7	6.0622	3	222	18.7	28.7
6	0.08829	12.5	7.87	6.012	66.6	5.5605	5	311	15.2	22.9
7	0.14455	12.5	7.87	6.172	96.1	5.9505	5	311	15.2	27.1
8	0.21124	12.5	7.87	5.631	100.0	6.0821	5	311	15.2	16.5
9	0.17004	12.5	7.87	6.004	85.9	6.5921	5	311	15.2	18.9

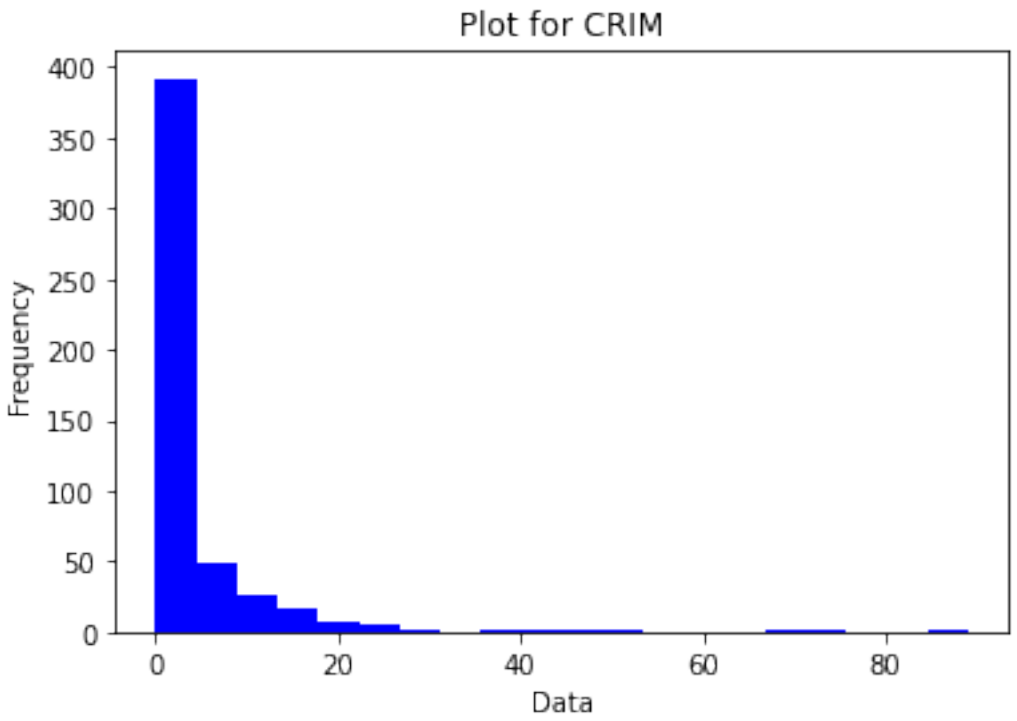
[19]:
```
# Checking the last seven records of the new Dataset
new_housing_data_df.tail(7)
```

[19]:

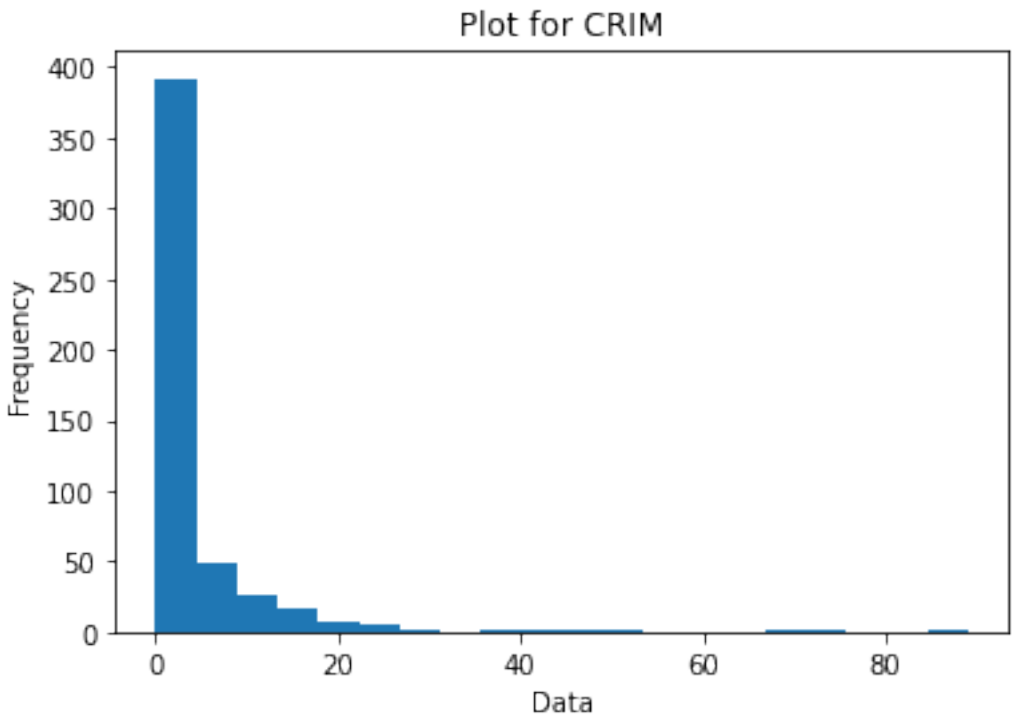
	CRIM	ZN	INDUS	RM	AGE	DIS	RAD	TAX	PTRATIO	PRICE
499	0.17783	0.0	9.69	5.569	73.5	2.3999	6	391	19.2	17.5
500	0.22438	0.0	9.69	6.027	79.7	2.4982	6	391	19.2	16.8
501	0.06263	0.0	11.93	6.593	69.1	2.4786	1	273	21.0	22.4
502	0.04527	0.0	11.93	6.120	76.7	2.2875	1	273	21.0	20.6
503	0.06076	0.0	11.93	6.976	91.0	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

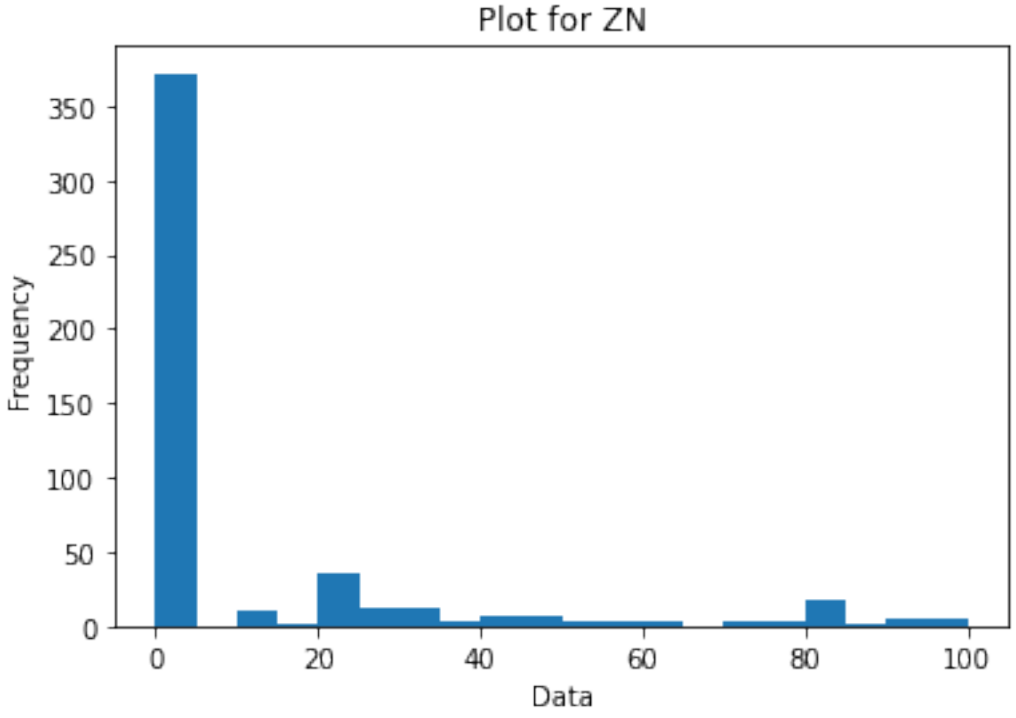
[32]:
```
# Plotting histogram for CRIM column

plt.hist(new_housing_data_df.CRIM, bins=20, color='blue')
plt.xlabel('Data')
plt.ylabel('Frequency')
plt.title('Plot for CRIM')
```

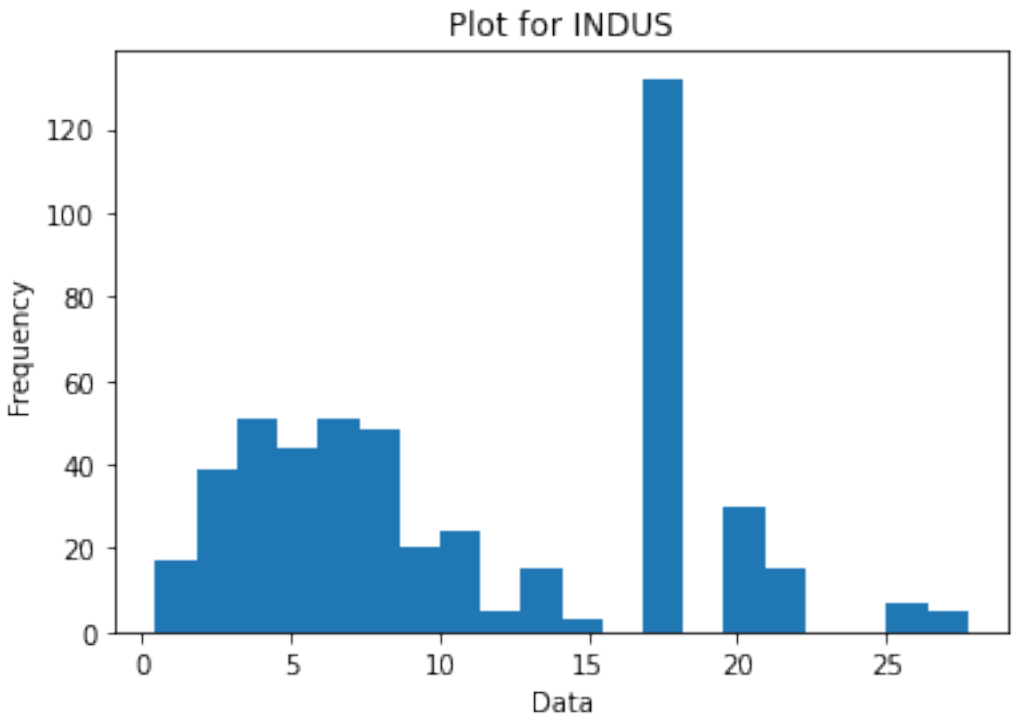
[32]:
```
Text(0.5, 1.0, 'Plot for CRIM')
```

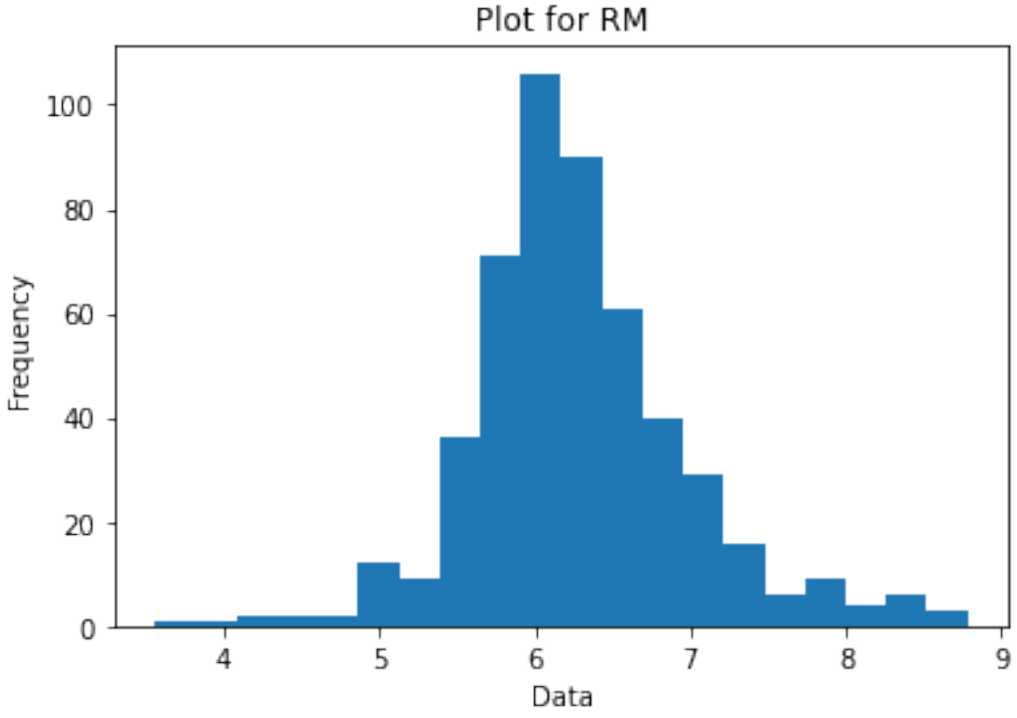


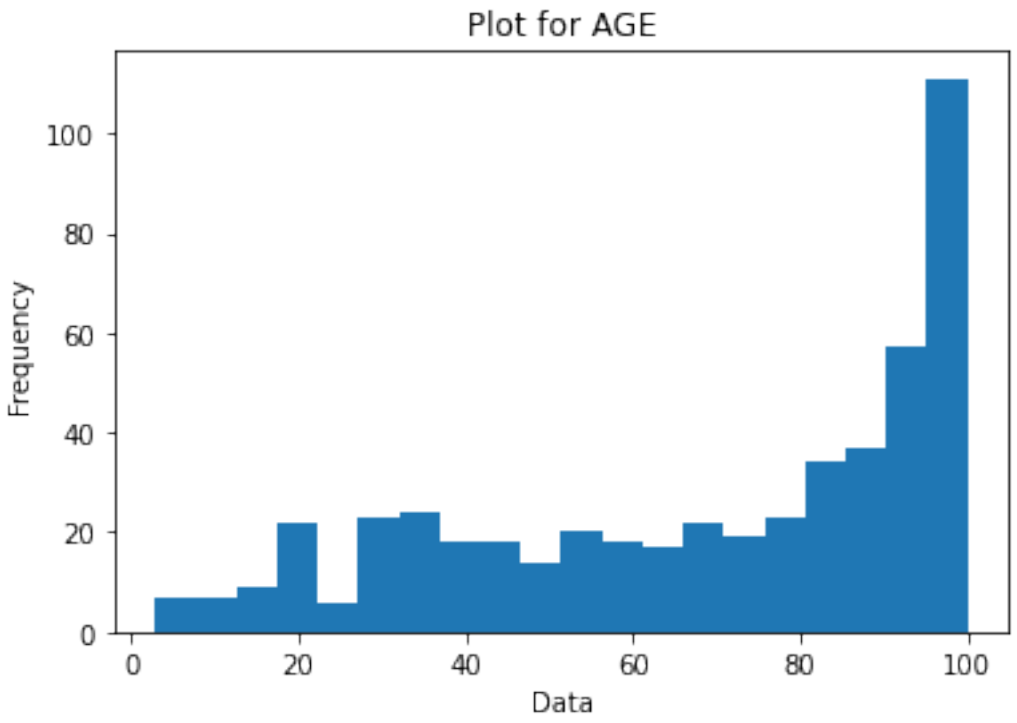
[34]:
```
# Plotting histograms for all columns in a loop

for i in new_housing_data_df.columns:
    plt.xlabel("Data")
    plt.ylabel("Frequency")
    plt.title("Plot for "+i)
    plt.hist(new_housing_data_df[i],bins=20)
    plt.show()
```

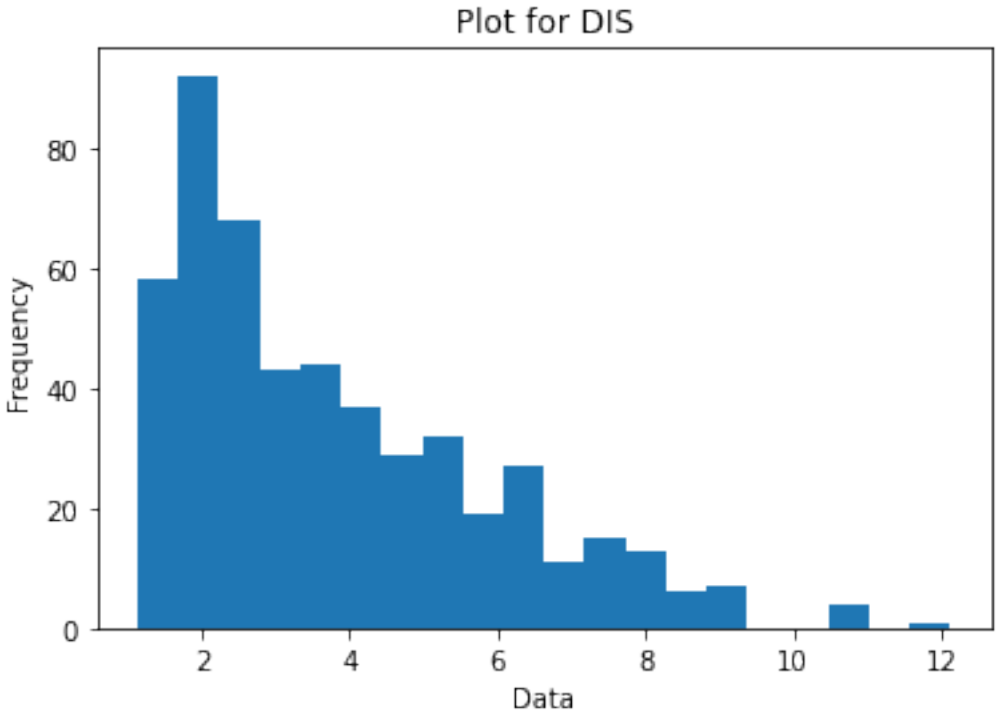

Plot for CRIM
Frequency
Data
0
50
100
150
200
250
300
350
400
0
20
40
60
80

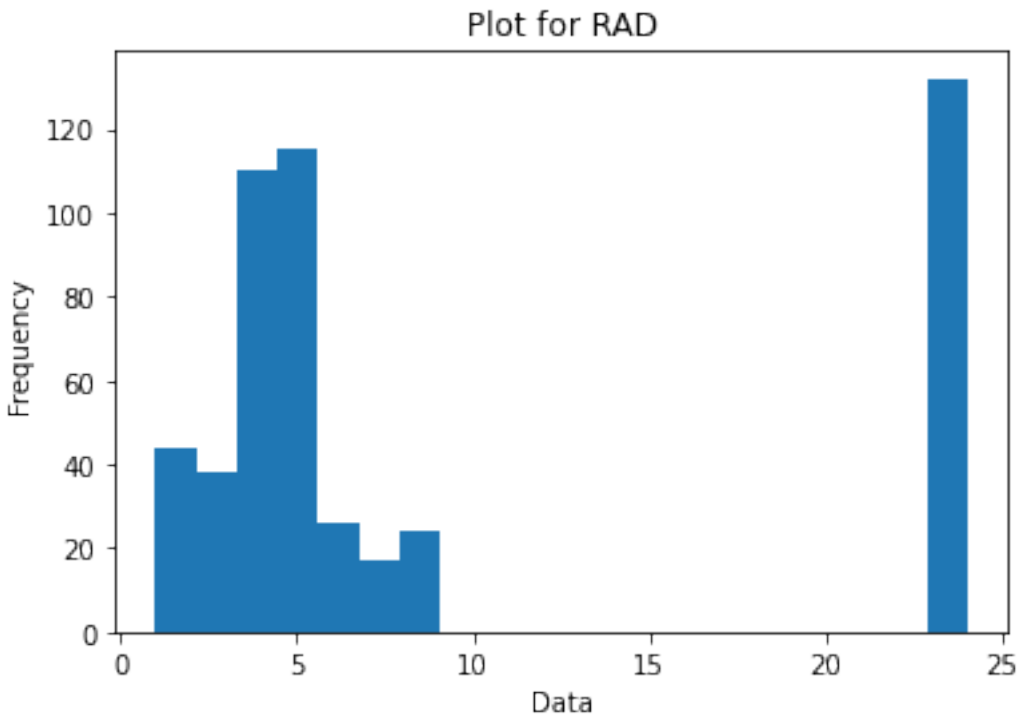

Plot for ZN
Frequency
Data
0
50
100
150
200
250
300
350
0
20
40
60
80
100

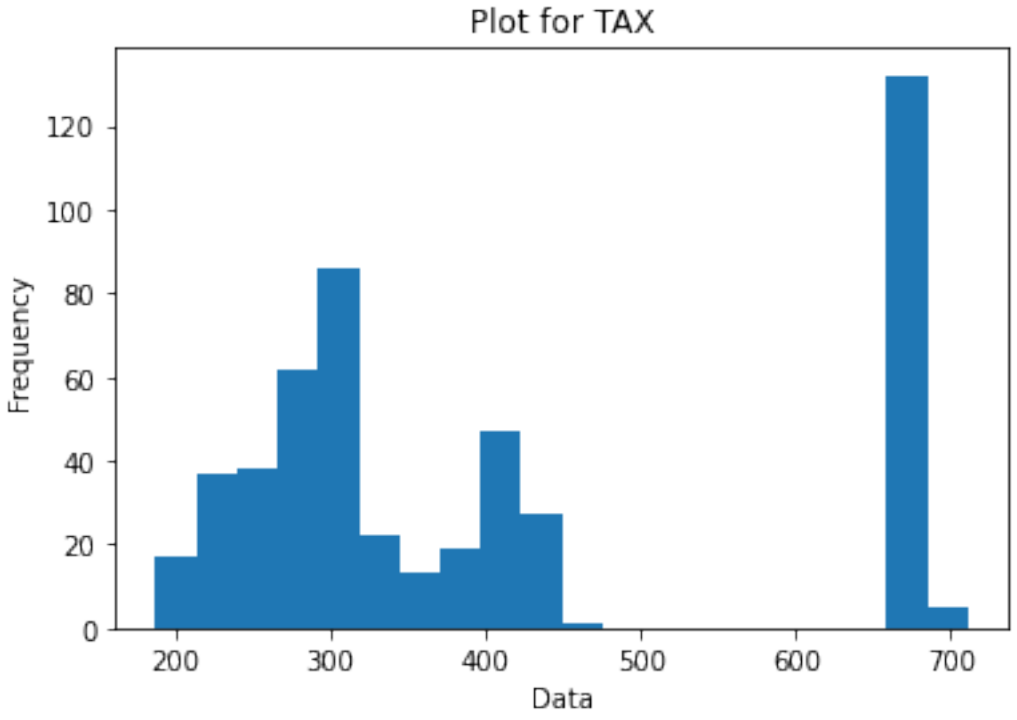

Plot for INDUS
Frequency
Data
0
20
40
60
80
100
120
0
5
10
15
20
25

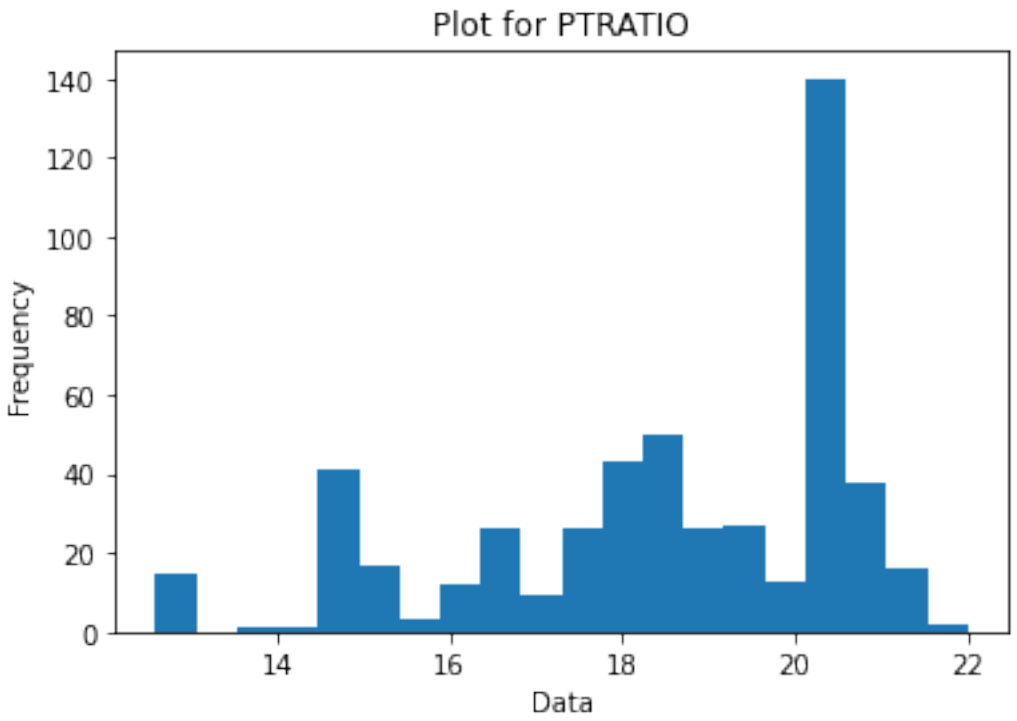

Plot for RM
Frequency
Data
0
20
40
60
80
100
4
5
6
7
8
9

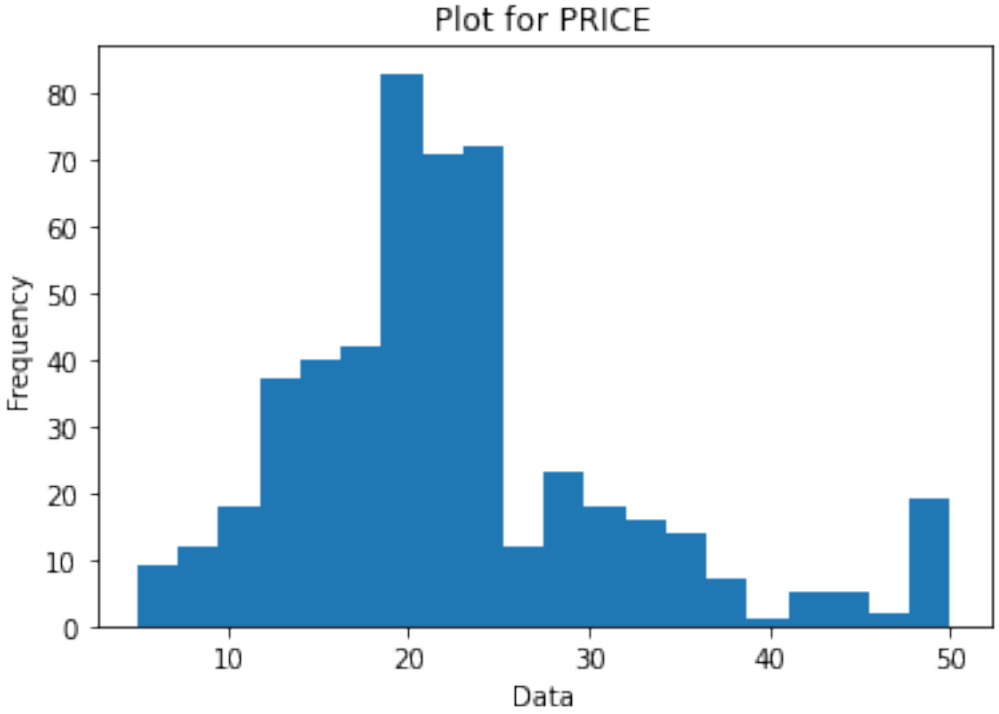

Plot for AGE
Frequency
Data
0
20
40
60
80
100


Plot for DIS
Frequency
Data
0
20
40
60
80
2
4
6
8
10
12

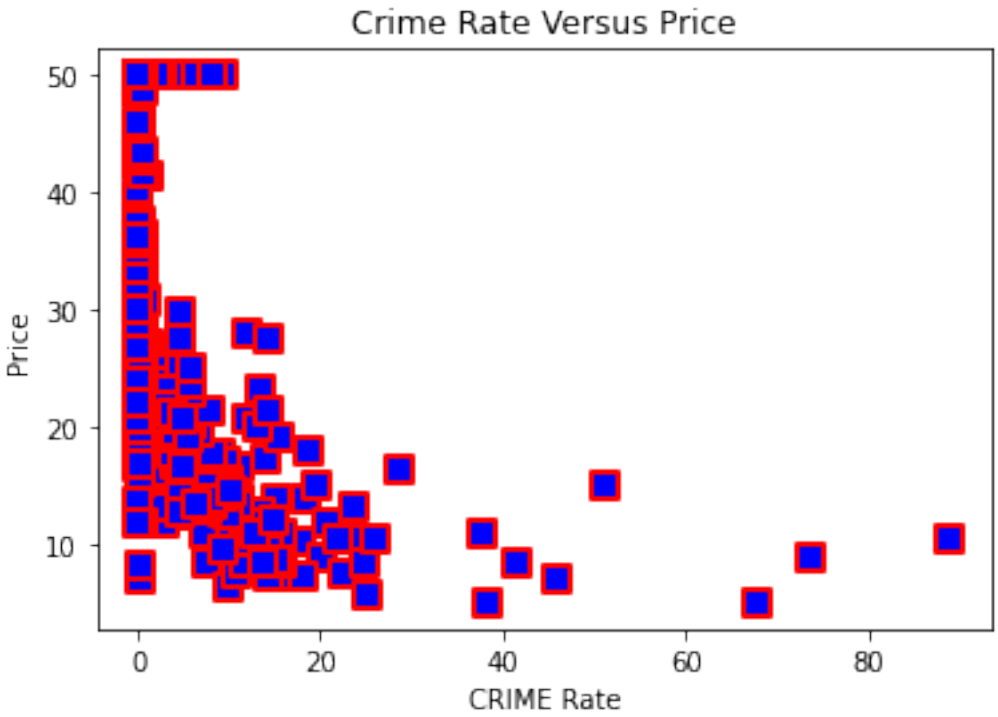

Plot for RAD
Frequency
Data
0
20
40
60
80
100
120
0
5
10
15
20
25


Plot for TAX
Frequency
Data
0
20
40
60
80
100
120
200
300
400
500
600
700


Plot for PTRATIO
Frequency
Data
0
20
40
60
80
100
120
140
14
16
18
20
22


Plot for PRICE
Frequency
Data
0
10
20
30
40
50
60
70
80
10
20
30
40
50

[50]:
```
# Create a scatter plot of crime rate versus price.

plt.scatter(new_housing_data_df["CRIM"],new_housing_data_df["PRICE"],
            c="blue",
            linewidths = 2,
            marker ="s",
            edgecolor ="red",
            s = 100)

plt.xlabel("CRIME Rate")
plt.ylabel("Price")
plt.title("Crime Rate Versus Price")
plt.show()
```



[49]:
```
# Plot using log10(crime) versus price.
plt.scatter(np.
 ↪log10(new_housing_data_df["CRIM"]),new_housing_data_df["PRICE"],c="blue")
plt.xlabel("Crime Rate Log")
plt.ylabel("Price")
plt.title("Crime rate Log versus Price")
plt.show()
```

[53]:
```
# Calculate some useful statistics, such as
# mean rooms per dwelling,
new_housing_data_df['RM'].mean()
```

[53]: 6.284634387351779

[54]:
```
# median age,
new_housing_data_df['AGE'].median()
```

[54]: 77.5

[55]:
```
# mean distances to five Boston employment centers,
new_housing_data_df['DIS'].mean()
```

[55]: 3.795042687747036

[56]:
```
# the percentage of houses with a low price (< $20,000).
# Creating a new dataset with Prices less than 20
cheap_house_df = new_housing_data_df["PRICE"] < 20
```

[65]:
```
# Checking top 10 rows
cheap_house_df.head(10)
```

[65]:
```
0    False
1    False
2    False
3    False
4    False
5    False
6    False
7    False
8     True
9     True
Name: PRICE, dtype: bool
```

[67]:
```
# New dataset contains boolean values: True & False with True for rows below 20
final_df=cheap_house_df.mean()
```

[69]:
```
# percentage of houses with a low price (< $20,000).
print(final_df*100)
```

```
41.50197628458498
```