Question 9. We will now consider the Boston housing data set, from the ISLR2 library.

(a) Based on this data set, provide an estimate for the population mean of medv . Call this estimate ˆμ.

(b) Provide an estimate of the standard error of ˆμ. Interpret this result. Hint: We can compute the standard error of the sample mean by dividing the sample standard deviation by the square root of the number of observations.

(c) Now estimate the standard error of ˆμ using the bootstrap. How does this compare to your answer from (b)?

(d) Based on your bootstrap estimate from (c), provide a 95 % con- fidence interval for the mean of medv . Compare it to the results obtained using t.test(Boston medv). Hint: You can approximate a 95 % confidence interval using the formula [ˆμ − 2SE(ˆμ), ˆμ + 2SE(ˆμ)].

(e) Based on this data set, provide an estimate, ˆμ med , for the median value of medv in the population.

(f) We now would like to estimate the standard error of ˆμ med . Unfortunately, there is no simple formula for computing the standard error of the median. Instead, estimate the standard error of the median using the bootstrap. Comment on your findings.

(g) Based on this data set, provide an estimate for the tenth percentile of medv in Boston census tracts. Call this quantity ˆμ 0.1 . (You can use the quantile() function.)

(h) Use the bootstrap to estimate the standard error of ˆμ 0.1 . Comment on your findings.

--

Boston Dataset

- id - Number assigned to every entry.
- CRIM - per capita crime rate by town.
- ZN - proportion of residential land zoned for lots over 25,000 sq.ft.
- INDUS - proportion of non-retail business acres per town.

- CHAS - Charles River dummy variable (1 if tract bounds river; 0 otherwise)
- NOX - nitric oxides concentration (parts per 10 million)
- RM - average number of rooms per dwelling.
- AGE - proportion of owner-occupied units built prior to 1940.
- DIS - weighted distances to five Boston employment centres.
- RAD - index of accessibility to radial highways.
- TAX - full-value property-tax rate per 10,000 Dollars.
- PTRATIO - pupil-teacher ratio by town.
- LSTAT - percent lower status of the population.
- MEDV - Median value of owner-occupied homes in $1000's.

1. Importing Necessary Library

In [1]:

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

2. Loading the Data

In [2]:

```
bostonData = pd.read_csv('boston.csv')
bostonData.head()
```

Out[2]:

	Unnamed: 0	crim	zn	indus	chas	nox	rm	age	dis	rad	tax	ptratio	lsta
0	1	0.00632	18.0	2.31	0	0.538	6.575	65.2	4.0900	1	296	15.3	4.9
1	2	0.02731	0.0	7.07	0	0.469	6.421	78.9	4.9671	2	242	17.8	9.1
2	3	0.02729	0.0	7.07	0	0.469	7.185	61.1	4.9671	2	242	17.8	4.0
3	4	0.03237	0.0	2.18	0	0.458	6.998	45.8	6.0622	3	222	18.7	2.9
4	5	0.06905	0.0	2.18	0	0.458	7.147	54.2	6.0622	3	222	18.7	5.3

In [3]:

```
bostonData = bostonData.drop('Unnamed: 0',axis=1)
bostonData.head()
```

Out[3]:

	crim	zn	indus	chas	nox	rm	age	dis	rad	tax	ptratio	lstat	medv
0	0.00632	18.0	2.31	0	0.538	6.575	65.2	4.0900	1	296	15.3	4.98	24.0
1	0.02731	0.0	7.07	0	0.469	6.421	78.9	4.9671	2	242	17.8	9.14	21.6
2	0.02729	0.0	7.07	0	0.469	7.185	61.1	4.9671	2	242	17.8	4.03	34.7
3	0.03237	0.0	2.18	0	0.458	6.998	45.8	6.0622	3	222	18.7	2.94	33.4
4	0.06905	0.0	2.18	0	0.458	7.147	54.2	6.0622	3	222	18.7	5.33	36.2

3.Shape of the Data

Shape of the data helps us understand the number of rows and number of columns present in out dataset.

In [4]:

```
bostonData.shape
```

Out[4]:

```
(506, 13)
```

4. Handling missing Values

- **Let's see if our data has any missing values.**
- **If it does then we will handle it by replacing it with the appropriate value based on data**

In [5]:

```
bostonData.isnull().sum()
```

Out[5]:

```
crim       0
zn         0
indus      0
chas       0
nox        0
rm         0
age        0
dis        0
rad        0
tax        0
ptratio    0
lstat      0
medv       0
dtype: int64
```

There are no missing values in the Dataset

5. Descriptive Statistics of our Dataset

5.1 Data type of all the variables in the Dataset

In [6]:

```
bostonData.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 506 entries, 0 to 505
Data columns (total 13 columns):
 #   Column   Non-Null Count  Dtype  
---  ------   --------------  -----  
 0   crim     506 non-null    float64
 1   zn       506 non-null    float64
 2   indus    506 non-null    float64
 3   chas     506 non-null    int64  
 4   nox      506 non-null    float64
 5   rm       506 non-null    float64
 6   age      506 non-null    float64
 7   dis      506 non-null    float64
 8   rad      506 non-null    int64  
 9   tax      506 non-null    int64  
 10  ptratio  506 non-null    float64
 11  lstat    506 non-null    float64
 12  medv     506 non-null    float64
dtypes: float64(10), int64(3)
memory usage: 51.5 KB
```

All our features are Quantitative

6. Based on this data set, provide an estimate for the population mean of medv . Call this estimate μˆ.

In [29]:

```
mu_hat = bostonData['medv'].mean()
print("Estimate for population mean of medv is: {:.5f}".format(mu_hat))
```

```
Estimate for population mean of medv is: 22.53281
```

6.1 Provide an estimate of the standard error of μˆ. Interpret this result. Hint: We can compute the standard error of the sample mean by dividing the sample standard deviation by the square root of the number of observations.

In [12]:

```
import math
```

$$StandardErrorofSample = \frac{(StandardDeviation(MEDV))}{(SquareRootofLengthofBostonData)}$$

In [14]:

```
muHat_SE = (np.std(bostonData['medv'])/math.sqrt(len(bostonData)))
```

In [18]:

```
print("Estimate of Standard Error for sample mean of medv is : {:.4f}".format(muHat_SE))
```

```
Estimate of Standard Error for sample mean of medv is : 0.4085
```

6.2 Now estimate the standard error of μˆ using the bootstrap. How does this compare to your answer from (b)?

6.2.1 Resampling Data

In [21]:

```
from sklearn.utils import resample
```

In [44]:

```
def boot(df):
    return resample(df)

sampleError = []
for i in range(1000):
    df = boot(bostonData)
    sampleError.append(df['medv'].mean())
```

In [49]:

```
boot_Standard_error = np.std(sampleError)
```

In [50]:

```
boot_Standard_error
```

Out[50]:

```
0.39034792796686735
```

6.2.2 Standard Error of MEDV using BootStrap

In [51]:

```
print("Estimate of Standard Error of MEDV using Bootstrap Method is : {}".format(boot_Sta
```

```
Estimate of Standard Error of MEDV using Bootstrap Method is : 0.390347927
96686735
```

The Standard Error of MEDV we achieved in (6.1) was 0.4085 and the Standard Error of MEDV using Bootstrap is 0.39034792796686735. We can say that there is a negligible difference between in both Standard Errors.

6.3 Based on your bootstrap estimate from (c) (6.2.2), provide a 95% confidence interval for the mean of medv. Compare it to the results obtained using t.test(Boston$medv)

6.3.1 95 % Confidence Interval

Suppose we want to generate a 95% confidence interval estimate for an unknown population mean. This means that there is a 95% probability that the confidence interval will contain the true population mean. Thus,

P([sample mean] - margin of error < μ < [sample mean] + margin of error) = 0.95.

In [31]:

```
print("95% CONFIDENCE INTERVAL FOR UNKNOWN POPULATION MEAN is {0} and {1}".format((22.53
```

```
95% CONFIDENCE INTERVAL FOR UNKNOWN POPULATION MEAN is 22.12421 and 22.941
41
```

6.3.2 T-test

This is a test for the null hypothesis that the expected value (mean) of a sample of independent observations a is equal to the given population mean.

In [33]:

```
from scipy import stats
```

In [37]:

```
stats.ttest_1samp(bostonData['medv'],popmean=0.0)
```

Out[37]:

```
Ttest_1sampResult(statistic=55.11114583037392, pvalue=9.370623727132662e-2
16)
```

The 95 % Confidence Interval for Unknown population mean is 22.12421 and 22.94141 and the p-Value for Median value of owner-occupied homes in $1000's(MEDV) is 9.370623727132662e-216 which makes this feature significantly associated.

6.4 Based on this data set, provide an estimate, μmedˆ, for the median value of medv in the population.

In [39]:

```
print("Estimate for population median of medv is: ",bostonData['medv'].median())
```

```
Estimate for population median of medv is:  21.2
```

6.5 We now would like to estimate the standard error of μmedˆ. Unfortunately, there is no simple formula for computing the standard error of the median. Instead, estimate the standard error of the median using the bootstrap. Comment on your findings.

In [52]:

```
sampleMedian = []

for i in range(1000):
    df = boot(bostonData)
    sampleMedian.append(df['medv'].median())
```

In [55]:

```
print("Estimate of standard error of sample median of medv (using bootstrap) is: ",np.st
```

```
Estimate of standard error of sample median of medv (using bootstrap) is:
0.3874382241080502
```

6.6 Based on this data set, provide an estimate for the tenth percentile of medv in Boston suburbs. Call this quantity μ0.1ˆ. (You can use the quantile() function.)

In [56]:

```
print("Estimate for the tenth percentile of medv is: " ,bostonData['medv'].quantile(q=0.1
```

```
Estimate for the tenth percentile of medv is:  12.75
```

6.7 Use the bootstrap to estimate the standard error of μ0.1ˆ. Comment on your findings.

In [59]:

```
sample_percentile = []

for i in range(1000):
    df = boot(bostonData)
    sample_percentile.append(df['medv'].quantile(q=0.1))
```

In [60]:

```
print("Estimate of standard error for the tenth percentile of medv (using bootstrap) is:
```

```
Estimate of standard error for the tenth percentile of medv (using bootstr
ap) is:  0.5113456365316906
```

In [ ]:

```

```

In [ ]:

```

```