Statistics for Data Science with Python

Project Case: Boston Housing Daata

In [4]:
```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
import scipy.stats
import statsmodels.api as sm
```

In [3]:
```
boston_url = 'https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBMDe
boston_df=pd.read_csv(boston_url)
```

In [5]:
```
boston_df.describe()
```

Out[5]:

	Unnamed: 0	CRIM	ZN	INDUS	CHAS	NOX	RM	AGE
count	506.000000	506.000000	506.000000	506.000000	506.000000	506.000000	506.000000	506.00000C
mean	252.500000	3.613524	11.363636	11.136779	0.069170	0.554695	6.284634	68.574901
std	146.213884	8.601545	23.322453	6.860353	0.253994	0.115878	0.702617	28.148861
min	0.000000	0.006320	0.000000	0.460000	0.000000	0.385000	3.561000	2.90000C
25%	126.250000	0.082045	0.000000	5.190000	0.000000	0.449000	5.885500	45.02500C
50%	252.500000	0.256510	0.000000	9.690000	0.000000	0.538000	6.208500	77.50000C
75%	378.750000	3.677083	12.500000	18.100000	0.000000	0.624000	6.623500	94.07500C
max	505.000000	88.976200	100.000000	27.740000	1.000000	0.871000	8.780000	100.00000C

In [6]:
```
boston_df.head(10)
```

Out[6]:

	Unnamed: 0	CRIM	ZN	INDUS	CHAS	NOX	RM	AGE	DIS	RAD	TAX	PTRATIO	LSTAT
0	0	0.00632	18.0	2.31	0.0	0.538	6.575	65.2	4.0900	1.0	296.0	15.3	4.98
1	1	0.02731	0.0	7.07	0.0	0.469	6.421	78.9	4.9671	2.0	242.0	17.8	9.14
2	2	0.02729	0.0	7.07	0.0	0.469	7.185	61.1	4.9671	2.0	242.0	17.8	4.03
3	3	0.03237	0.0	2.18	0.0	0.458	6.998	45.8	6.0622	3.0	222.0	18.7	2.94
4	4	0.06905	0.0	2.18	0.0	0.458	7.147	54.2	6.0622	3.0	222.0	18.7	5.33
5	5	0.02985	0.0	2.18	0.0	0.458	6.430	58.7	6.0622	3.0	222.0	18.7	5.21
6	6	0.08829	12.5	7.87	0.0	0.524	6.012	66.6	5.5605	5.0	311.0	15.2	12.43
7	7	0.14455	12.5	7.87	0.0	0.524	6.172	96.1	5.9505	5.0	311.0	15.2	19.15
8	8	0.21124	12.5	7.87	0.0	0.524	5.631	100.0	6.0821	5.0	311.0	15.2	29.93
9	9	0.17004	12.5	7.87	0.0	0.524	6.004	85.9	6.5921	5.0	311.0	15.2	17.10

Task : Familiarize yourself with the dataset

The following describes the dataset variables:

· 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

· PTRATIO - pupil-teacher ratio by town

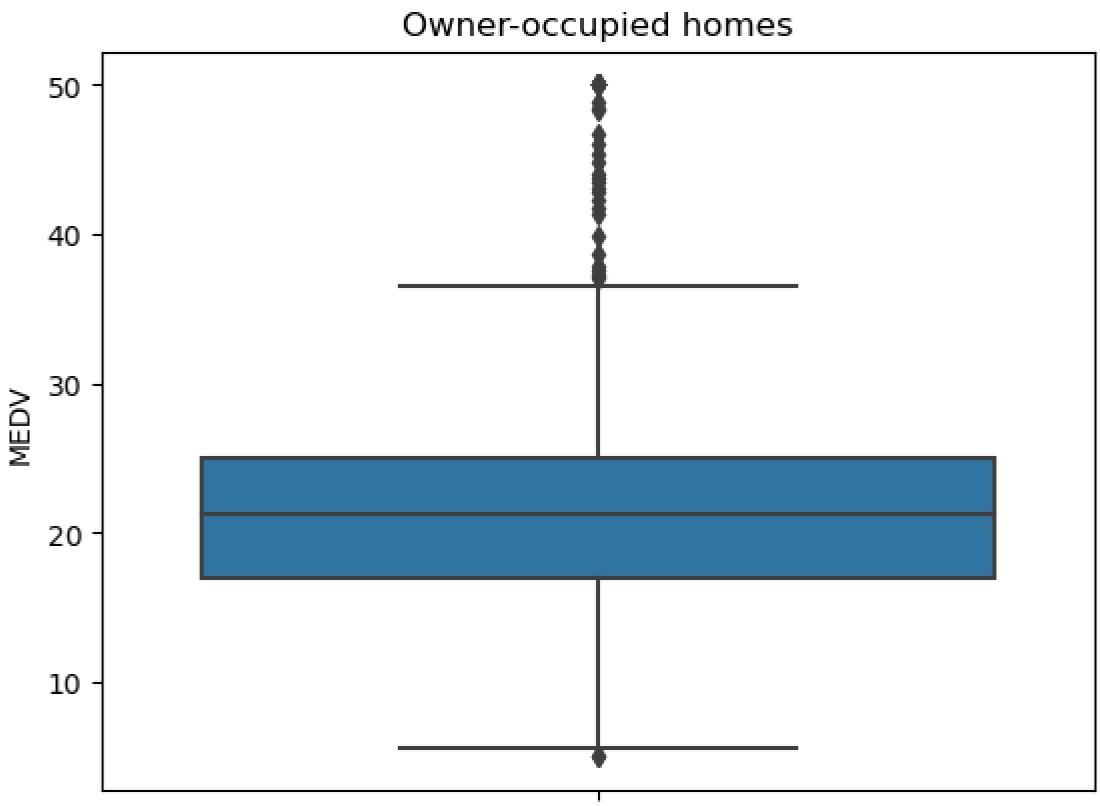
· LSTAT - % lower status of the population

· MEDV - Median value of owner-occupied homes in $1000's

Task: Generate basic statistics and visualizations for upper management.

In [9]:
```
ax = sns.boxplot(y = 'MEDV', data = boston_df)
ax.set_title('Owner-occupied homes')
```

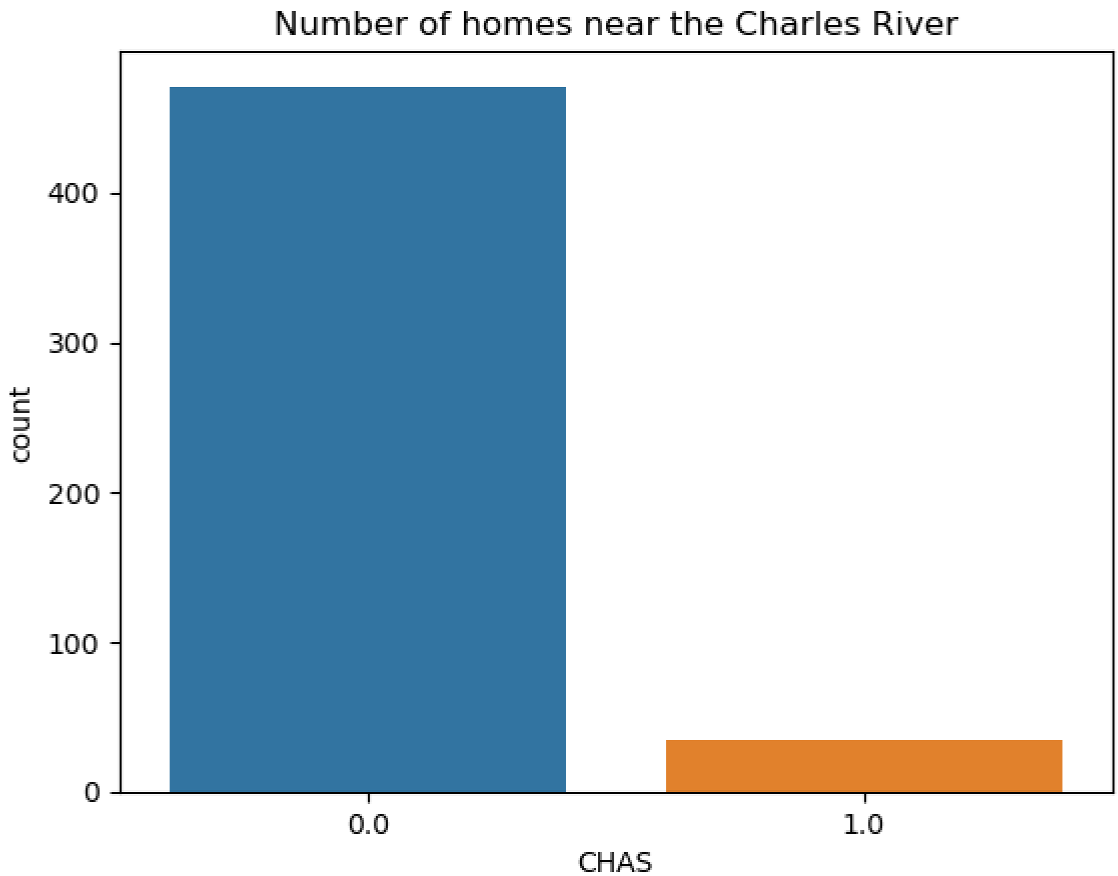
Out[9]:
```
Text(0.5, 1.0, 'Owner-occupied homes')
```



The boxplot above shows the median value for the variable MEDV among with outliers

In [11]:
```
ax2 = sns.countplot(x = 'CHAS', data = boston_df)
ax2.set_title('Number of homes near the Charles River')
```

Out[11]:
```
Text(0.5, 1.0, 'Number of homes near the Charles River')
```

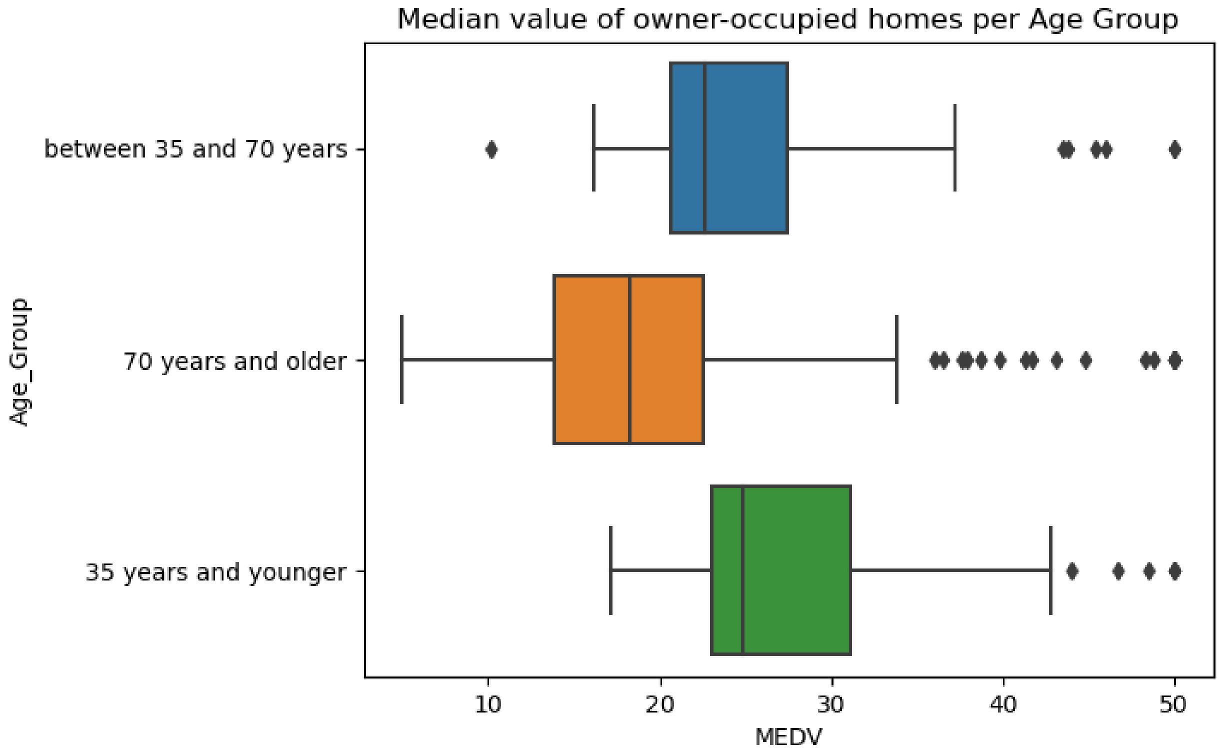


The histogram shows that the majority of the houses are not near the Charles River

In [12]:
```
boston_df.loc[(boston_df['AGE'] <= 35), 'Age_Group'] = '35 years and younger'
boston_df.loc[(boston_df['AGE'] > 35) & (boston_df['AGE'] < 70), 'Age_Group'] = 'betwe
boston_df.loc[(boston_df['AGE'] >= 70), 'Age_Group'] = '70 years and older'
```

In [13]:
```
ax3 = sns.boxplot(x = 'MEDV', y = 'Age_Group', data = boston_df)
ax3.set_title('Median value of owner-occupied homes per Age Group')
```

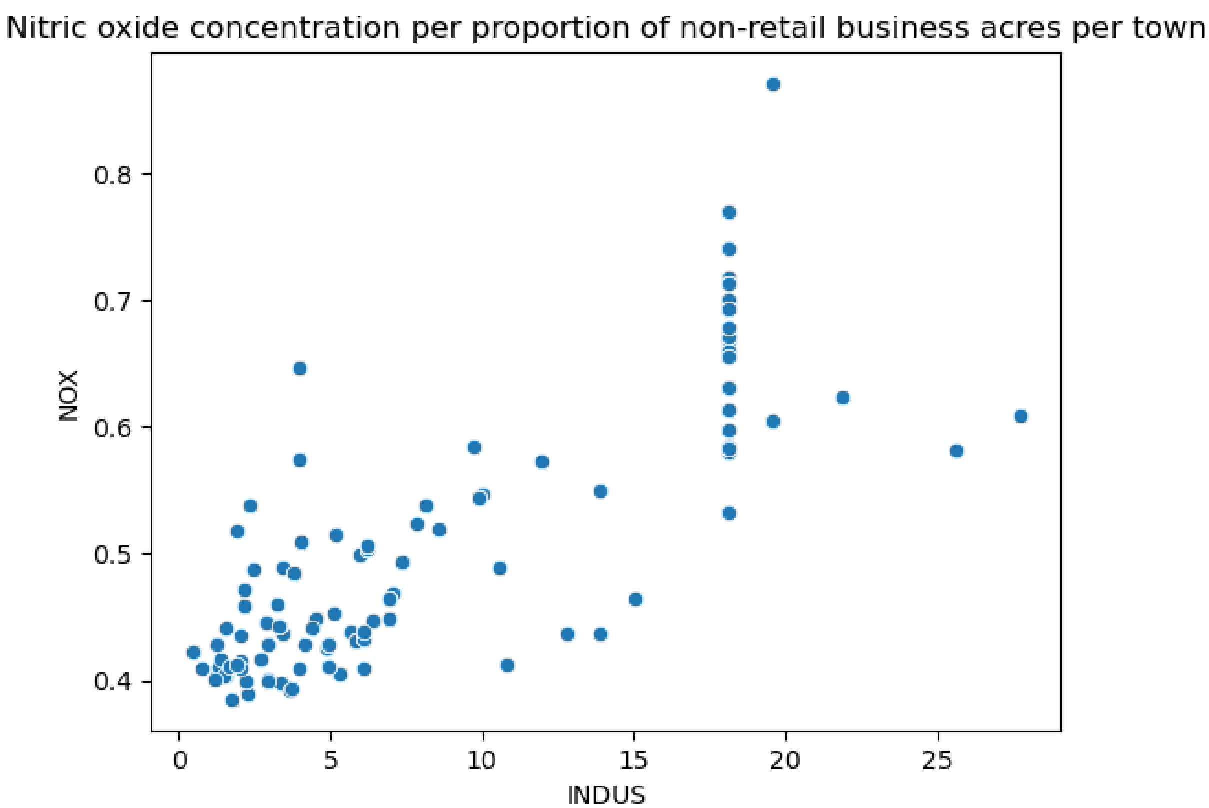
Out[13]:
```
Text(0.5, 1.0, 'Median value of owner-occupied homes per Age Group')
```



The boxplot above shows that on average the median value of owner occupied homes is higher when the Age is lower

In [14]:
```
ax4 = sns.scatterplot(y = 'NOX', x = 'INDUS', data = boston_df)
ax4.set_title('Nitric oxide concentration per proportion of non-retail business acres
```

Out[14]:
```
Text(0.5, 1.0, 'Nitric oxide concentration per proportion of non-retail business acre
s per town')
```



Values in the bottom-left section of the scatter plot indicates a strong relation between low Nitric oxide concentration and low proportion of non-retail business acres per town.

Generally, a higher proprtion of non-retail business acres per town produces a higher concentration of Nitric oxide.

Task : Use the appropriate tests to answer the questions provided.

Is there a significant difference in the median value of houses bounded by the Charles river or not?

Hypothesis:

Null Hypothesis -> There's no significant difference in median value between houses bounded and not bounded by the Charles River

Alternative Hypothesis -> There's a significant difference in median value between houses bounded and not bounded by the Charles River

In [15]:
```
boston_df.loc[(boston_df['CHAS'] == 0), 'CHAS_T'] = 'FAR'
boston_df.loc[(boston_df['CHAS'] == 1), 'CHAS_T'] = 'NEAR'
boston_df.head(5)
```

Out[15]:

	Unnamed: 0	CRIM	ZN	INDUS	CHAS	NOX	RM	AGE	DIS	RAD	TAX	PTRATIO	LSTAT
0	0	0.00632	18.0	2.31	0.0	0.538	6.575	65.2	4.0900	1.0	296.0	15.3	4.98
1	1	0.02731	0.0	7.07	0.0	0.469	6.421	78.9	4.9671	2.0	242.0	17.8	9.14
2	2	0.02729	0.0	7.07	0.0	0.469	7.185	61.1	4.9671	2.0	242.0	17.8	4.03
3	3	0.03237	0.0	2.18	0.0	0.458	6.998	45.8	6.0622	3.0	222.0	18.7	2.94
4	4	0.06905	0.0	2.18	0.0	0.458	7.147	54.2	6.0622	3.0	222.0	18.7	5.33

In [16]:
```
scipy.stats.ttest_ind(boston_df[boston_df['CHAS_T'] == 'FAR']['MEDV'],
                      boston_df[boston_df['CHAS_T'] == 'NEAR']['MEDV'], equal_var = Tr
```

Out[16]:
```
Ttest_indResult(statistic=-3.996437466090509, pvalue=7.390623170519905e-05)
```

Given the p-value is less than 0.05, we reject the Null Hypothesis, meaning there is not a statistical difference in median value betwenn houses near the Charles River and houses far away

Is there a difference in median values of houses of each proportion of owner-occupied units built before 1940?

Hypothesis

Null Hypotesis: There isn't statistical difference in Median values of houses (MEDV) for each proportion of owner occpied units built prior to 1940

Alternative Hypothesis: There is statistical difference in Median values of houses (MEDV) for each proportion of owner occpied units built prior to 1940

In [17]:
```
from statsmodels.formula.api import ols
lm = ols('MEDV ~ AGE', data = boston_df).fit()
table = sm.stats.anova_lm(lm)
print(table)
```

```
             df        sum_sq      mean_sq          F        PR(>F)
AGE         1.0   6069.761065  6069.761065  83.477459  1.569982e-18
Residual  504.0  36646.534350    72.711378        NaN           NaN
```

Given p-value is less than 0.05, we fail to accept the Null Hypothesis --> There is statistical difference in Median values of houses (MEDV) for each proportion of owner occpied units built prior to 1940

Can we conclude that there is no relationship between Nitric oxide concentrations and the proportion of non-retail business acres per town?

Null Hypothesis: Nitric Oxide concentration is not correlated with the proportion of non-retail business acres per town

Alternative Hypothesis: Nitric Oxide concentration is correlated with the proportion of non-retail business acres per town

In [18]:
```
scipy.stats.pearsonr(boston_df['NOX'], boston_df['INDUS'])
```

Out[18]:
```
PearsonRResult(statistic=0.7636514469209151, pvalue=7.913361061238693e-98)
```

Given the Pearson Coefficient is 0.76365 and p-value less than 0.05, we reject the Null Hypothesis as there is a positive correlation between Nitric oxide concentration and proportion of non-retail business acres per town

The positive relationship is confirmed also with the Scatter Plot

What is the impact of an additional weighted distance to the five Boston employment centres on the median value of owner-occupied homes?

In [19]:
```
x = boston_df['DIS']
y = boston_df['MEDV']

x = sm.add_constant(x)

model = sm.OLS(y, x).fit()
predisction = model.predict(x)

model.summary()
```

Out[19]:

OLS Regression Results

Dep. Variable:	MEDV	**R-squared:**	0.062
Model:	OLS	**Adj. R-squared:**	0.061
Method:	Least Squares	**F-statistic:**	33.58
Date:	Mon, 24 Jul 2023	**Prob (F-statistic):**	1.21e-08
Time:	00:23:45	**Log-Likelihood:**	-1823.9
No. Observations:	506	**AIC:**	3652.
Df Residuals:	504	**BIC:**	3660.
Df Model:	1		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	18.3901	0.817	22.499	0.000	16.784	19.996
DIS	1.0916	0.188	5.795	0.000	0.722	1.462

Omnibus:	139.779	**Durbin-Watson:**	0.570
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	305.104
Skew:	1.466	**Prob(JB):**	5.59e-67
Kurtosis:	5.424	**Cond. No.**	9.32

Notes:

[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.

In [ ]:
```
```