Chapter 6.
Descriptive Statistics

6.0 Introduction

- Data: a mixture of nature and noise.
- Is the noise manageable?
  - The noise is desired to be representable by a probability distribution.
- Statistical inference:
  - The science of deducing properties of an underlying probability distribution from data

- Can we have information on the underlying probability distribution?
  - The information is given in the form of (functions of) data.

6.1 Experimentation
6.1.1 Samples

- **Population**: the set of all the possible observations available from a particular probability distribution.
- **Sample**: a subset of a population.
- **Random sample**: a sample where the elements are chosen at random from the population
- A sample is desired to be representative of the population.
- Types of observations: numerical and nominal

6.1.2 Examples

- Example 1: Machine breakdowns
  - Suppose that an engineer in charge of the maintenance of a machine keeps records on the breakdown causes over a period of a year.

  - Suppose that 46 breakdowns were observed by the engineer (Figure 6.2).

- What is the population from which this sample is drawn?

  - The population consists of all the breakdowns that occurred and will occur to the machine.

Breakdown cause	Frequency
Electrical	9
Mechanical	24
Misuse	13
Total	46

FIGURE 6.2

Data set of machine breakdowns

Example 1: Machine breakdowns

- Factors to consider to check representativeness of data:
  - Quality of operators
  - Working load on the machine
  - Particularity of data observation (e.g., more rainy days than other years)

Example 2: Defective computer chips

- The chip boxes are selected at random from …..

- Points to check on data:
  - What is the data type?
  - Are the data representative?
  - How the randomness of data realized?
- Statistical problem:
  - What is the population from which the data are sampled?
    - The population consists of all the chip boxes that are produced over a certain period of time.

6.2 Data presentation

6.2.1 Bar and Pareto charts

Python codes for bar charts

```
auto=pd.read_csv('C:/data/autos.txt')
print(auto)
  cars₩ttrucks₩tsuvs
0          12₩t26₩t47
1          29₩t52₩t43
2          62₩t43₩t63
3          38₩t51₩t62
4        93₩t122₩t159
```

6.2 Data presentation

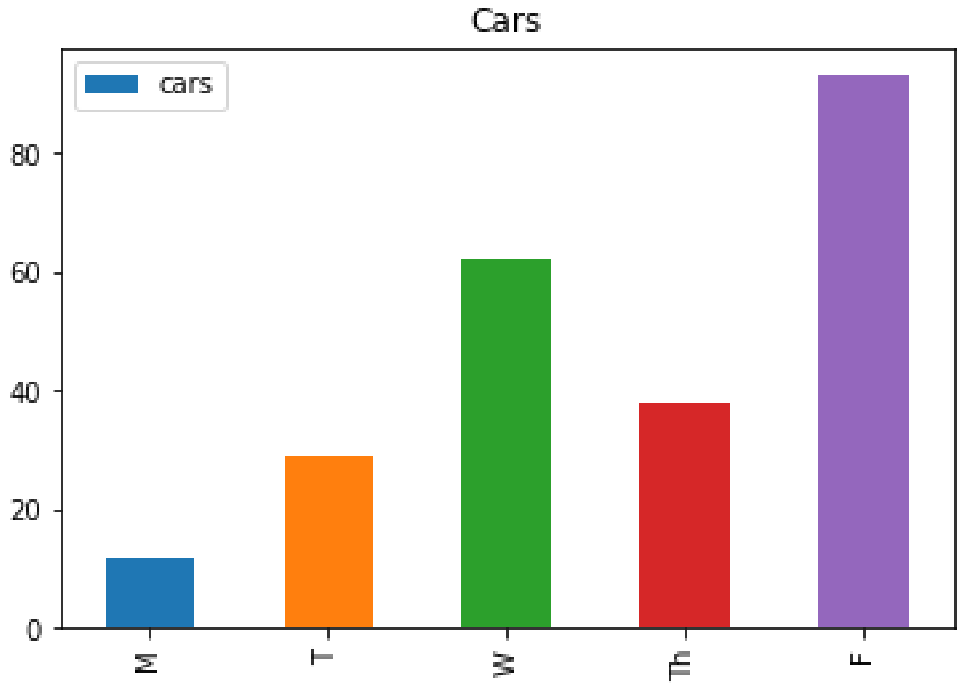
```
auto = pd.read_csv('C:/data/autos.txt', sep='₩t')
    #  To remove “₩t” in the last table, use the
    #  option ‘sep’
```

```
print(auto)
   cars  trucks  suvs
0    12      26    47
1    29      52    43
2    62      43    63
3    38      51    62
4    93     122   159
```

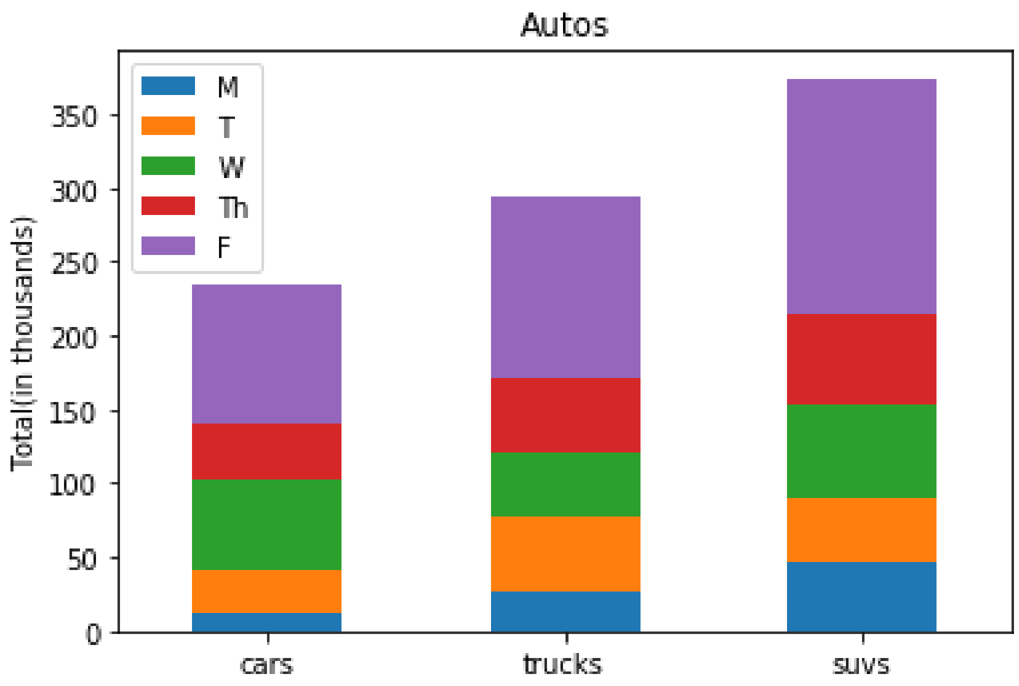
```
print(list(auto))
['cars', 'trucks', 'suvs']
```

```
auto.index = ['M','T','W','Th','F']

print(auto)
    cars  trucks  suvs
M     12      26    47
T     29      52    43
W     62      43    63
Th    38      51    62
F     93     122   159
```

```
auto.plot.bar(y='cars', title='Cars')
plt.show()    # for bar plots
```

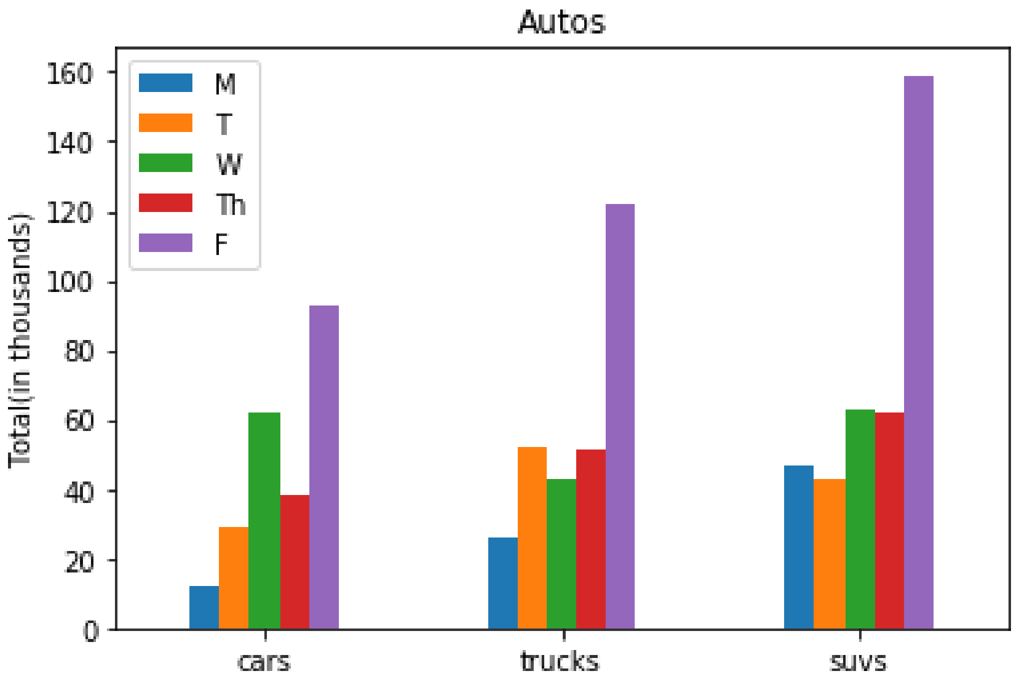


```
print(auto.apply(sum))
cars      234
trucks    294
suvs      374
dtype: int64
```



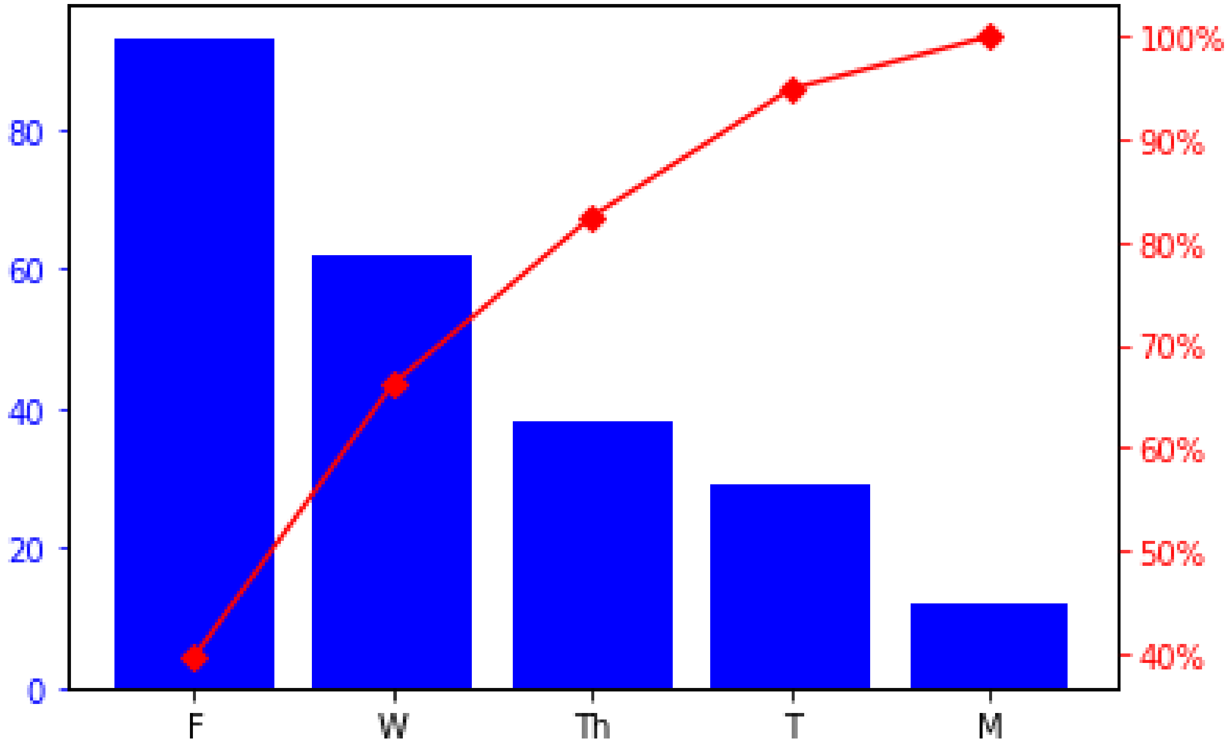

```
auto.T.plot.bar(y=['M','T','W','Th','F'], titl
      # 'rot' is for the angle of the x-label.
plt.ylabel('Total(in thousands)')
plt.show()    # for a stacked bar plot
```

```
auto.T.plot.bar(y=['M','T','W','Th','F'], title='Autos', rot=0)
plt.ylabel('Total(in thousands)')
plt.show()     # for a side-by-side bar plot
```



6.2.1 Bar and Pareto charts

Python codes for Pareto charts



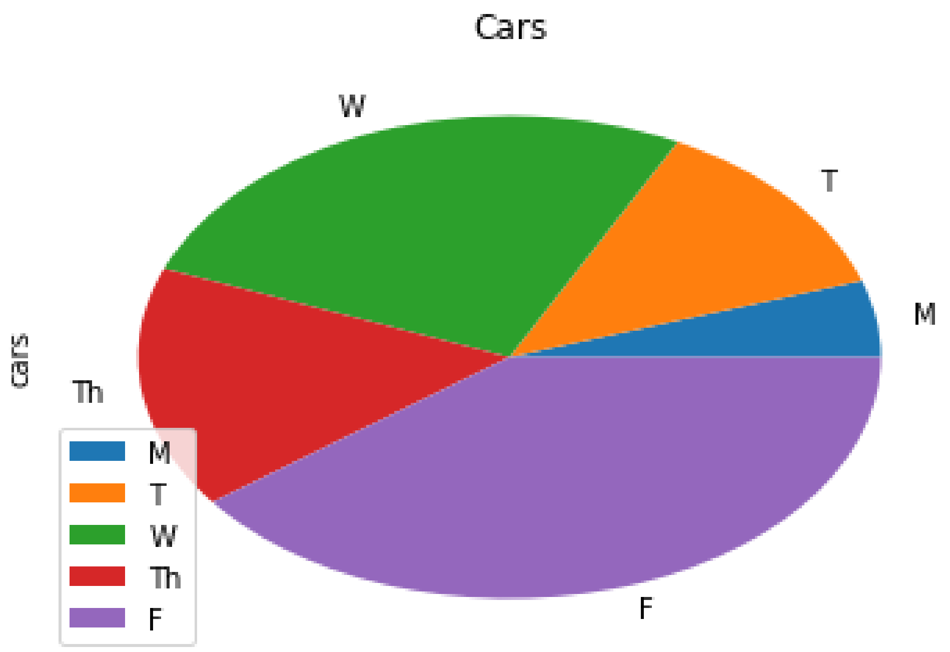

```
import pandas as pd
import matplotlib.pyplot as plt
from matplotlib.ticker import PercentFormatter
```

```
auto_ord= auto.sort_values(by='cars',ascending=False)
auto_ord["cumpercentage"] = auto_ord["cars"].cumsum()/auto["cars"].sum()*100
Print(list(auto_ord))
    ['cars', 'trucks', 'suvs', 'cumpercentage']

fig, ax = plt.subplots()
auto.index=['F','W','Th','T','M']     # index reordered
ax.bar(auto.index, auto_ord["cars"], color="b")
  # Colors may be given as "C0", "C1",....
ax2 = ax.twinx()
ax2.plot(auto.index, auto_ord["cumpercentage"], color="r", marker="D", ms=5)
ax2.yaxis.set_major_formatter(PercentFormatter())

ax.tick_params(axis="y", colors="b")
ax2.tick_params(axis="y", colors="r")
plt.show()
```

6.2.2 Pie charts

Python codes for pie charts

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

auto.plot.pie(y='cars', title='Cars')
plt.savefig('C:/.../fig/plot1.png',bbox_inches='tight')
          # to save the plot as file 'plot1.png' with a tig
          # margin in directory 'C:/.../fig'.
plt.show()   # for pie charts
```



6.2.3 Histograms

- Useful for reading a general trend in data.
- Stem-and-leaf plots are a special type of a histogram which shows the data values in the picture.

Python codes for histograms and stem-and-leaf plots

```
> airquality = pd.read_csv('C:/.../data/airquality.csv')
> print(airquality.head())
   Unnamed: 0  Ozone  Solar.R  Wind  Temp  Month  Day
0           1   41.0    190.0   7.4    67      5    1
1           2   36.0    118.0   8.0    72      5    2
2           3   12.0    149.0  12.6    74      5    3
3           4   18.0    313.0  11.5    62      5    4
4           5    NaN      NaN  14.3    56      5    5
```

Python codes for histograms and stem-and-leaf plots

```
> airquality = pd.read_csv('C:/.../data/airquality.csv', index_col=0)
> print(airquality.head())
   Ozone  Solar.R  Wind  Temp  Month  Day
1   41.0    190.0   7.4    67      5    1
2   36.0    118.0   8.0    72      5    2
3   12.0    149.0  12.6    74      5    3
4   18.0    313.0  11.5    62      5    4
5    NaN      NaN  14.3    56      5    5

> print(airquality.shape)
(153, 6)
```

Python codes for histograms and stem-and-leaf plots

```
> print(airquality.describe())
            Ozone     Solar.R        Wind        Temp       Month         Day
count  116.000000  146.000000  153.000000  153.000000  153.000000  153.000000
mean    42.129310  185.931507    9.957516   77.882353    6.993464   15.803922
std     32.987885   90.058422    3.523001    9.465270    1.416522    8.864520
min      1.000000    7.000000    1.700000   56.000000    5.000000    1.000000
25%     18.000000  115.750000    7.400000   72.000000    6.000000    8.000000
50%     31.500000  205.000000    9.700000   79.000000    7.000000   16.000000
75%     63.250000  258.750000   11.500000   85.000000    8.000000   23.000000
max    168.000000  334.000000   20.700000   97.000000    9.000000   31.000000
```

```
> from stemgraphic import stem_graphic
   # install 'stemgraphic' by typing "pip install stemgraphic" in
   # the Anaconda Prompt.
> stem_graphic(airquality['Ozone'])
> plt.show()   # for Stem-and-leaf plots
```

Python codes for histograms and stem-and-leaf plots

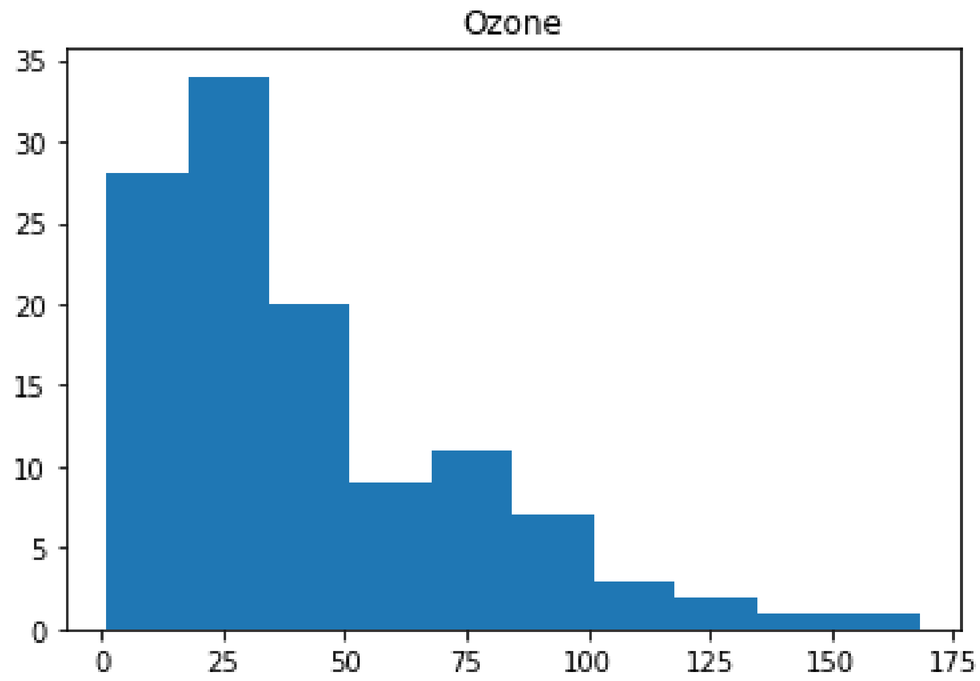
Key: aggr|stem|leaf

16 | *8* = **16**.8x10 = 168.0

aggr	stem	leaf
116	16	8
115	15	
115	14	
115	13	5
114	12	2
113	11	058
110	10	8
109	9	1677
105	8	024559
99	7	13367889
91	6	134456
85	5	0299
81	4	01444556789
70	3	001222455667799
55	2	0000111123333334478889
33	1	011122333344446666688889
10	0	1467778999

Python codes for histograms and stem-and-leaf plots

```
> airquality.hist(['Ozone'], grid=False)
> plt.show()    # for a histogram
```



6.2.4 Outliers

- An outlier is an observation which is not from the distribution from which the main body of the sample is collected.
- Outliers need to be removed for analysis.

6.3 Sample statistics

Sample: $X_1, \cdots, X_n$

6.3.1 Sample mean

$$\overline{X} = \frac{\sum_{i=1}^{n} X_i}{n}$$

6.3.2 Sample median

The (n+1)/2-th smallest of the sample when n is odd;

The average of the n/2-th and the (n+1)/2-th smallest of the sample when n is even.

6.3.3 r% sample trimmed mean

The average of the subset of the sample which is obtained by removing the top r% and the bottom r% from the sample.

6.3.4 Sample mode

The value of the sample at which the sample frequency is the largest.

Sample: $X_1, \cdots, X_n$

6.3.5 Sample variance

$$S^2 = \frac{\sum_i^n \left(X_i - \overline{X}\right)^2}{n-1}.$$

S is called the sample standard deviation.

6.3.6 100p-th sample quantile

The value y which satisfies

$$\frac{\#(X_i \le y)}{n} \ge p \text{ and } \frac{\#(X_i \ge y)}{n} \ge 1-p$$

The 25-th and 75-th sample quantiles are called in particular the 1-st (Q_1)and the 3-rd sample quartiles(Q_3) respectively.

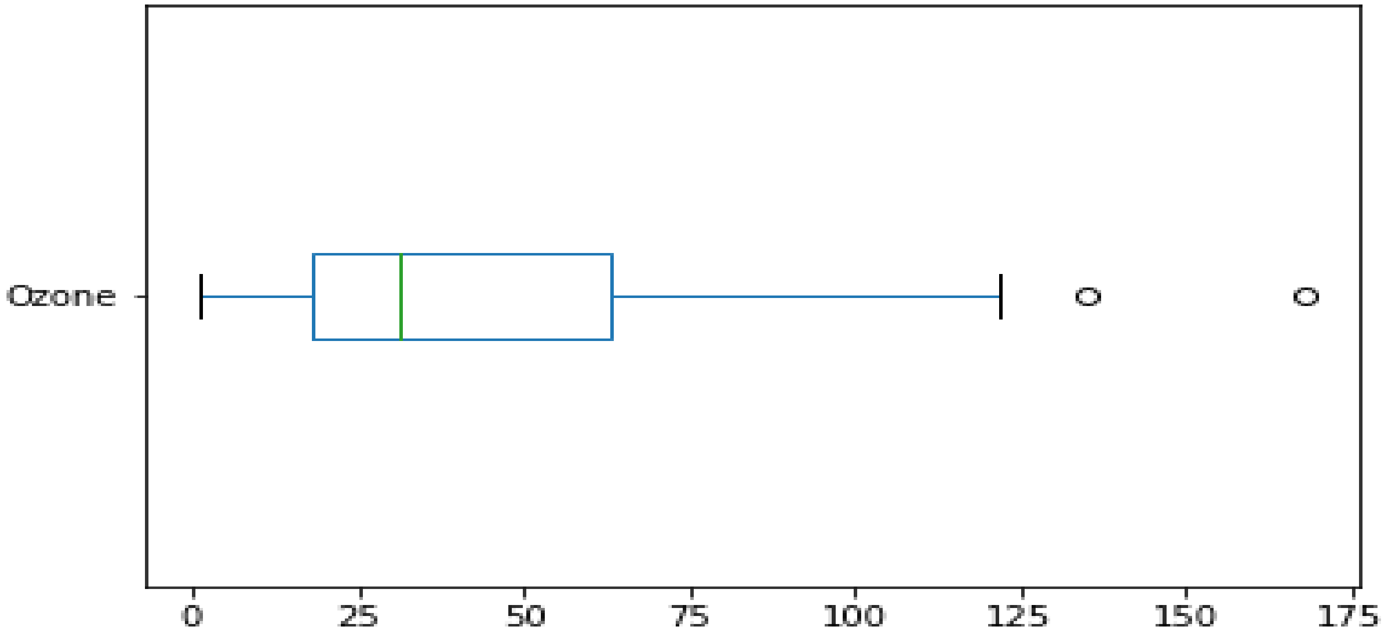
The Inter-quartile range (IQR) is the difference, $Q_3 - Q_1$.

6.3.7 Boxplots

Used for reading a general shape of the distribution of data.

Useful for finding candidates of outliers.

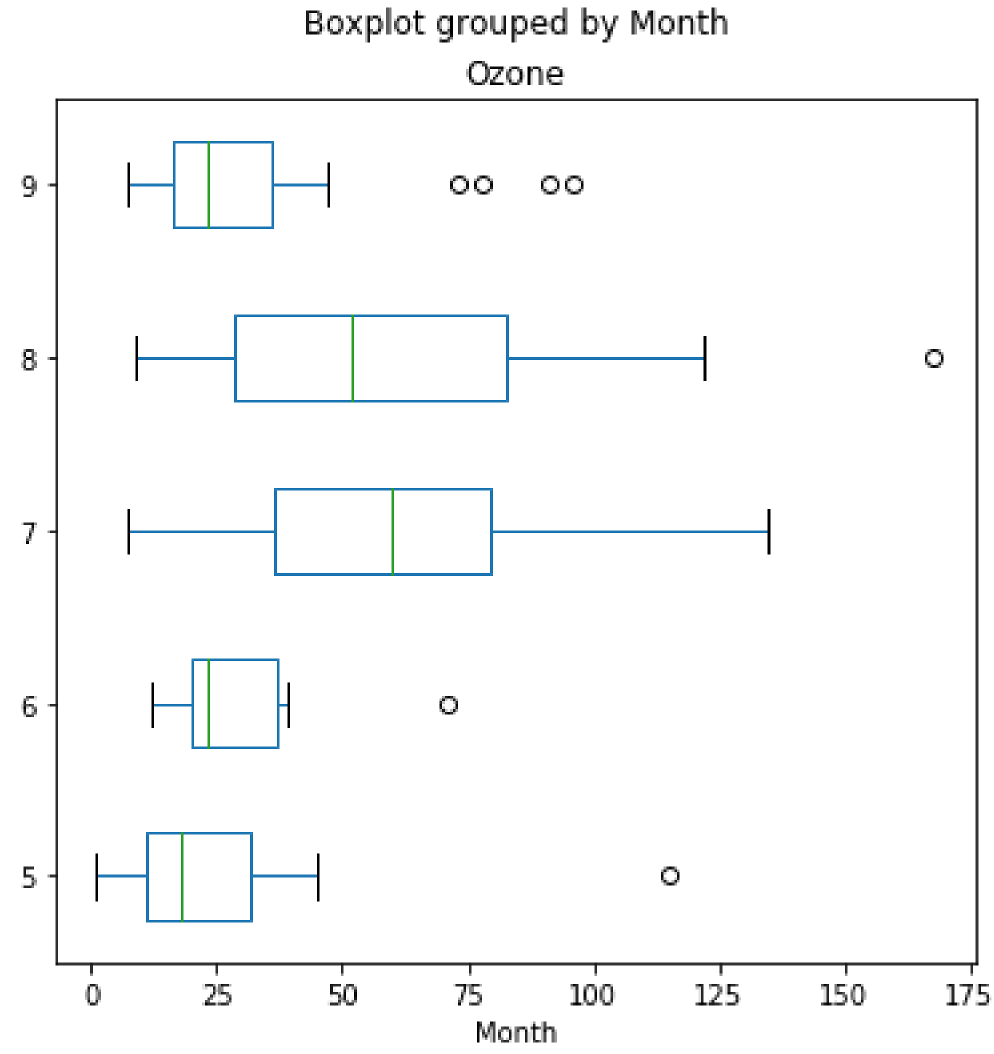
```
> airquality.boxplot(['Ozone'], grid=False, vert=False)
> plt.show()
```



> airquality.boxplot('Ozone', by='Month', grid=False, vert=False, figsize=(6,6))

> plt.show()

\# for boxplots grouped by Month



Sample: $X_1, \cdots, X_n$

6.3.8 Coefficient of variation

$$CV = \frac{S}{\overline{X}}$$

Chapter Summary

6.1 Experimentation

6.2 Data Presentation

6.3 Sample Statistics