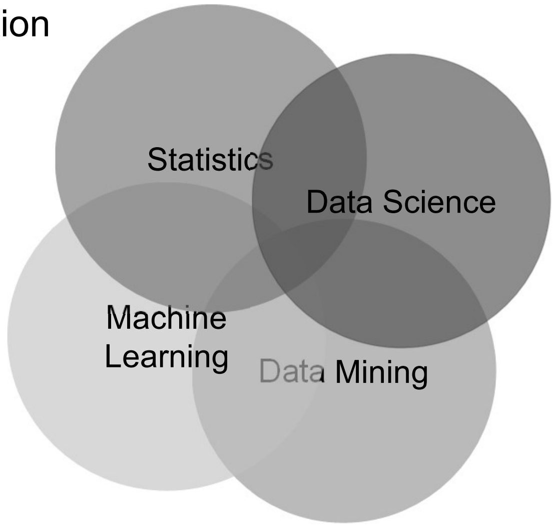
Introduction to Statistics and Data Science

- Definition



- Commonality is to improve decision making through the analysis of data!

Introduction to Statistics and Data Science

History

- 17~18 centuries: the foundation of probability theory
- 19 century : used probability distribution (Laplace, Gauss…)
- 1940's : the first neural network is introduced as a mathematical model
- 1950's : classification, pattern recognition problems are solved
- 1956~1960 : 'machine-learning' and 'artificial intelligence' are developed
- 1960~ : deep learning, vision, natural-language processing
- 1980~ : 'data mining' is used with big data
- 2000~ : visualization

Introduction to Statistics and Data Science

Definition

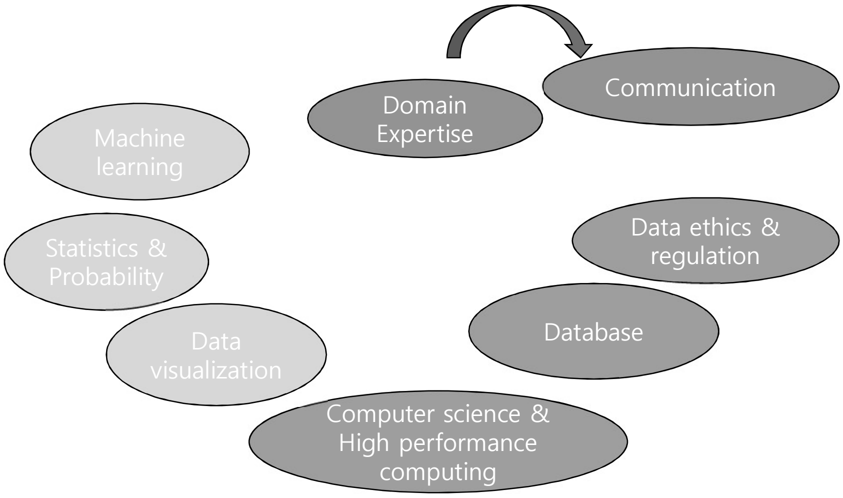
- Statistics is the branch of science that deals with the collection and analysis of data

- "Statistics = Data Science?" (C.F. Jeff Wu's , 1997)

  -. Availability of large/complex data sets in massive database
  -. Growing use of computational algorithms and models
  -. Statistics can be renamed "data science"

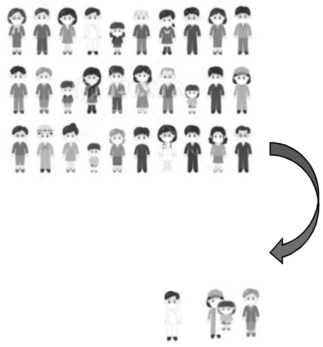
Introduction to Statistics and Data Science

What you have to learn…



Introduction to Statistics and Data Science

- Population and Sample



- Population: all the subjects of interests

  -. Infinite population
  -. Finite population

- Sample: a part of population

Images from Google

Introduction to Statistics and Data Science

- Data types

  - Discrete data
    : Countable data, categorical data

    Example: gender (male or female), number of defects

  - Continuous data
    : uncountable data and typically expressed as real numbers

    Example: height, weight

Introduction to Statistics and Data Science

- Representative measurements
  - Mean
$$\bar{X} = \frac{1}{n}\sum_{i=1}^{n} x_i$$
  - Median
$$m = \begin{cases} x_{(\frac{n+1}{2})} & if\ n\ is\ odd\ number \\ \dfrac{x_{(\frac{n}{2})} + x_{(\frac{n}{2}+1)}}{2} & if\ n\ is\ even\ number \end{cases}$$
  - Mode

    : the most frequent number

Introduction to Statistics and Data Science

- Dispersion measurements
  - Variance
$$S^2 = \frac{1}{n-1}\sum_{i=1}^{n}(x_i - \bar{x})^2$$
  - Standard deviation
$$s = \sqrt{S^2} = \sqrt{\frac{1}{n-1}\sum_{i=1}^{n}(x_i - \bar{x})^2}$$
  - Range
$$R = Maximum\ - minimum$$
  - Inter-quantile Range (IQR) $IQR = Q3\ - Q1$

count 3.0 mean 2.0 std 1.0 min 1.0 25% 1.5 50% 2.0 75% 2.5 max 3.0

count 3.0 mean 2.0 std 1.0 min 1.0 25% 1.5 50% 2.0 75% 2.5 max 3.0

Introduction to Statistics and Data Science

np.sum(x, *axis*)
-. axis = 0: column-wise sum
-. axis = 1: row-wise sum
-. axis = None : total sum of all elements

np.mean(x, *axis*)
-. axis = 0: column-wise mean
-. axis = 1: row-wise mean
-. axis = None : mean of all elements

count 3.0 mean 2.0 std 1.0 min 1.0 25% 1.5 50% 2.0 75% 2.5 max 3.0

count 3.0 mean 2.0 std 1.0 min 1.0 25% 1.5 50% 2.0 75% 2.5 max 3.0

Introduction to Statistics and Data Science

np.var(x, *axis, ddof*)
-. axis = 0: column-wise variance
-. axis = 1: row-wise variance
-. axis = None : total variance of all elements

-. ddof : delta degrees of freedom, the divisor in calculation
ddof = 0: n
ddof = 1: n-1

np.std(x, *axis, ddof*) : standard deviation

count 3.0 mean 2.0 std 1.0 min 1.0 25% 1.5 50% 2.0 75% 2.5 max 3.0
count 3.0 mean 2.0 std 1.0 min 1.0 25% 1.5 50% 2.0 75% 2.5 max 3.0

Introduction to Statistics and Data Science

np.percentile(x, *q*, *axis*)
-. axis = 0: column-wise variance
-. axis = 1: row-wise variance
-. axis = None : total variance of all elements

-. q : a sequence of percentiles between 0 and 100

count 3.0 mean 2.0 std 1.0 min 1.0 25% 1.5 50% 2.0 75% 2.5 max 3.0
count 3.0 mean 2.0 std 1.0 min 1.0 25% 1.5 50% 2.0 75% 2.5 max 3.0

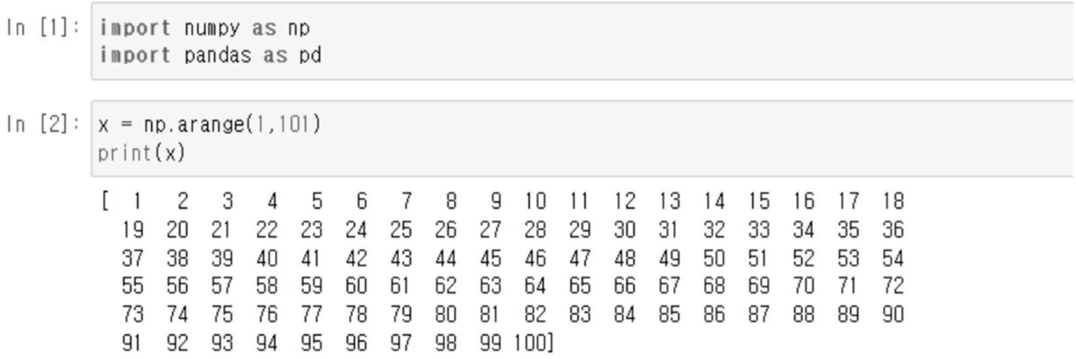
Introduction to Statistics and Data Science

```
def fname(x):
      …
      return y
```

-. *def* : a function
-. x : inputs
-. y : outputs

Introduction to Statistics and Data Science

- Practice



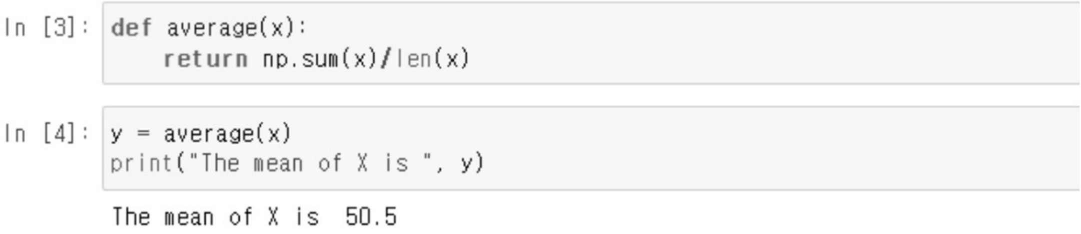
```
In [1]: import numpy as np
        import pandas as pd
```

```
In [2]: x = np.arange(1,101)
        print(x)
```

```
[  1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18
  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36
  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54
  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72
  73  74  75  76  77  78  79  80  81  82  83  84  85  86  87  88  89  90
  91  92  93  94  95  96  97  98  99 100]
```

Introduction to Statistics and Data Science

- Practice : computation



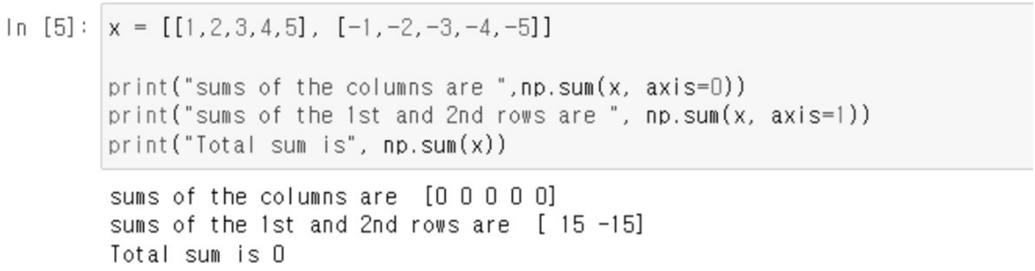
```
In [3]: def average(x):
            return np.sum(x)/len(x)
```

```
In [4]: y = average(x)
        print("The mean of X is ", y)
```

```
The mean of X is  50.5
```

Introduction to Statistics and Data Science

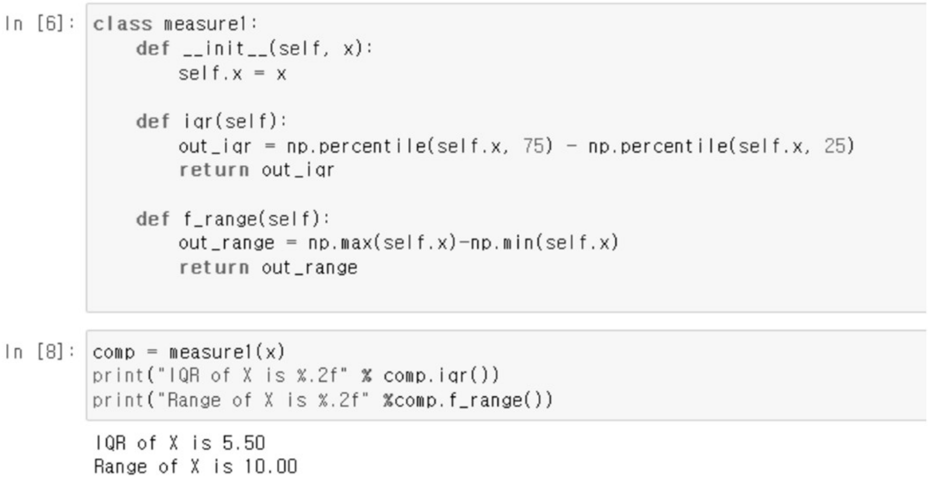
- Practice : computation using *axis*



```
In [5]: x = [[1,2,3,4,5], [-1,-2,-3,-4,-5]]

        print("sums of the columns are ",np.sum(x, axis=0))
        print("sums of the 1st and 2nd rows are ", np.sum(x, axis=1))
        print("Total sum is", np.sum(x))

        sums of the columns are  [0 0 0 0 0]
        sums of the 1st and 2nd rows are  [ 15 -15]
        Total sum is 0
```

Introduction to Statistics and Data Science

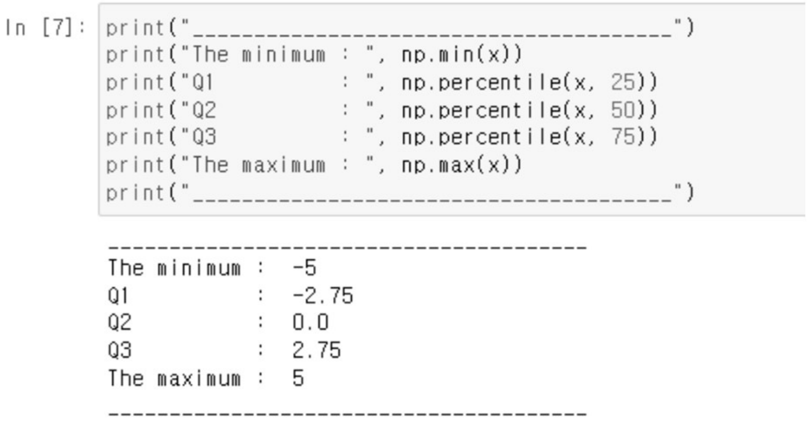
- Practice



```
In [6]: class measure1:
            def __init__(self, x):
                self.x = x

            def iqr(self):
                out_iqr = np.percentile(self.x, 75) - np.percentile(self.x, 25)
                return out_iqr

            def f_range(self):
                out_range = np.max(self.x)-np.min(self.x)
                return out_range

In [8]: comp = measure1(x)
        print("IQR of X is %.2f" % comp.iqr())
        print("Range of X is %.2f" %comp.f_range())

        IQR of X is 5.50
        Range of X is 10.00
```

Introduction to Statistics and Data Science

- Practice : 5 measures



```
In [7]: print("________________________________________")
        print("The minimum : ", np.min(x))
        print("Q1          : ", np.percentile(x, 25))
        print("Q2          : ", np.percentile(x, 50))
        print("Q3          : ", np.percentile(x, 75))
        print("The maximum : ", np.max(x))
        print("________________________________________")
```

```
----------------------------------------
The minimum :  -5
Q1          :  -2.75
Q2          :  0.0
Q3          :  2.75
The maximum :  5
----------------------------------------
```

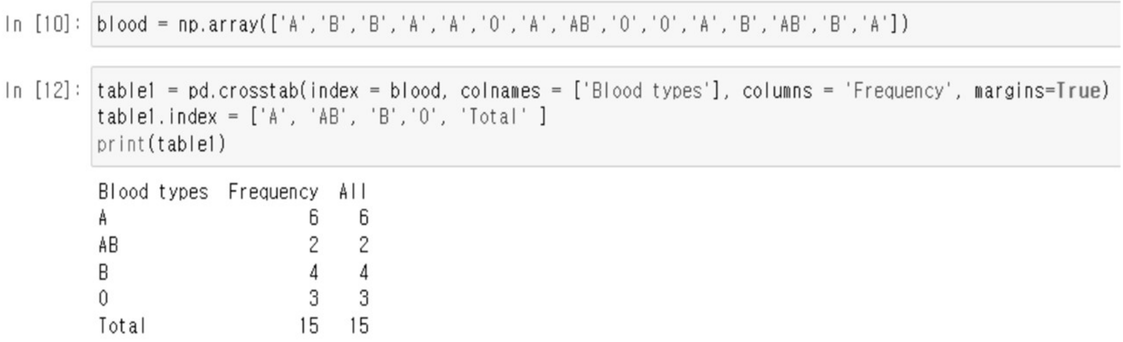
Introduction to Statistics and Data Science

- Frequency Table
  - For discrete data,
    - The frequency table is the numeric table summarized by frequencies per class
    - Class : distinctive values or factor
    - Frequency : how many times the given values or factors are appeared in the data
    - Relative Frequency (RF) : $RF = \frac{Frequency}{n}$

pd.crosstab(*index, columns, colnames, margins, margins_name*)

-. index : values to group by in the rows
-. columns: values to group by in the columns
-. colnames: name of the column
-. margins : row / column's margin
-. margins_name: name of the row or column that will contain the total

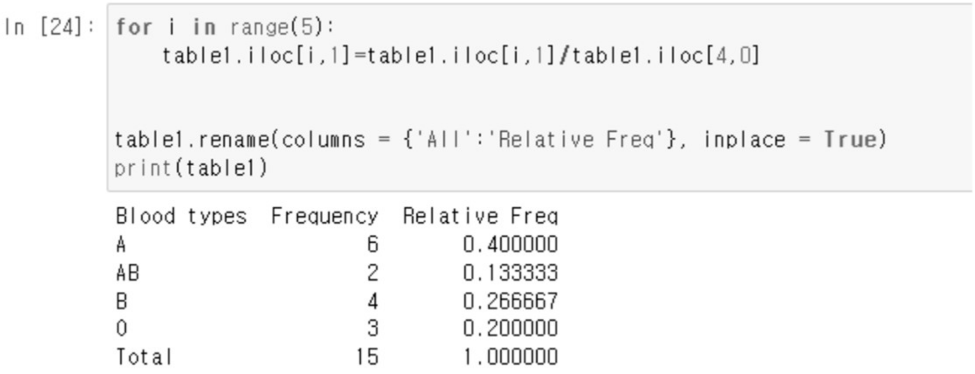
- Practice



```
In [10]: blood = np.array(['A','B','B','A','A','O','A','AB','O','O','A','B','AB','B','A'])

In [12]: table1 = pd.crosstab(index = blood, colnames = ['Blood types'], columns = 'Frequency', margins=True)
         table1.index = ['A', 'AB', 'B','O', 'Total' ]
         print(table1)

         Blood types  Frequency  All
         A                    6    6
         AB                   2    2
         B                    4    4
         O                    3    3
         Total               15   15
```

Introduction to Statistics and Data Science

- Practice : Relative frequency



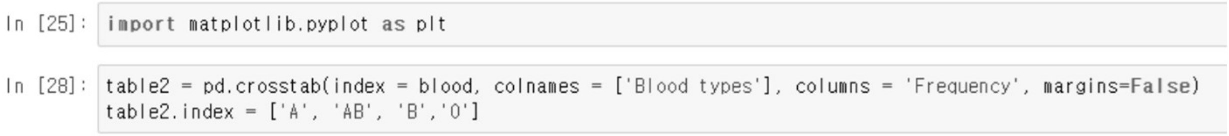
```
In [24]: for i in range(5):
             table1.iloc[i,1]=table1.iloc[i,1]/table1.iloc[4,0]


         table1.rename(columns = {'All':'Relative Freq'}, inplace = True)
         print(table1)

         Blood types  Frequency  Relative Freq
         A                    6       0.400000
         AB                   2       0.133333
         B                    4       0.266667
         O                    3       0.200000
         Total               15       1.000000
```

Introduction to Statistics and Data Science

- Visualization
  - Bar graph
    - The frequency table is the numeric table summarized by frequencies per class

Introduction to Statistics and Data Science

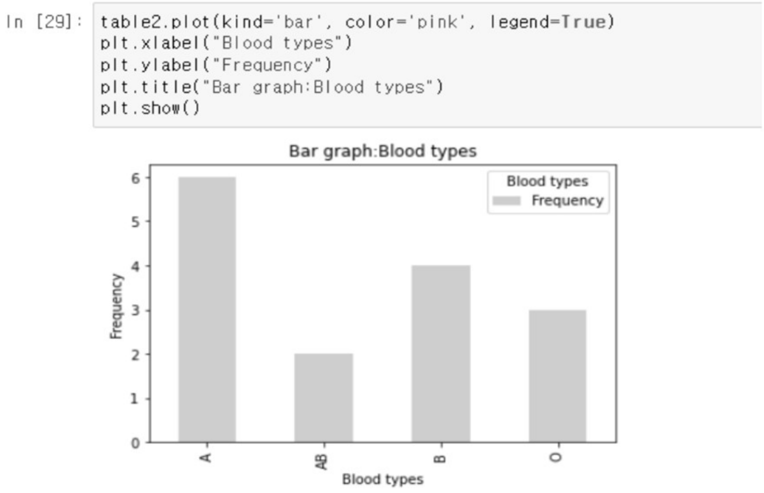
- Practice



```
In [25]: import matplotlib.pyplot as plt

In [28]: table2 = pd.crosstab(index = blood, colnames = ['Blood types'], columns = 'Frequency', margins=False)
         table2.index = ['A', 'AB', 'B','O']
```

Introduction to Statistics and Data Science

- Practice



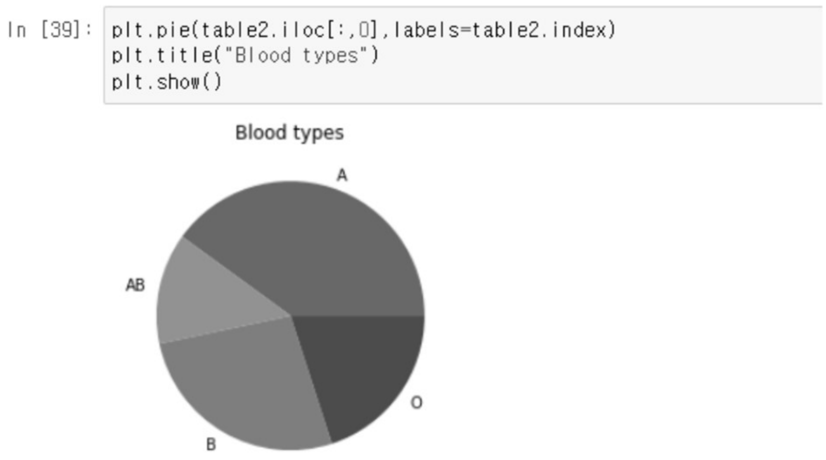
```
In [29]: table2.plot(kind='bar', color='pink', legend=True)
         plt.xlabel("Blood types")
         plt.ylabel("Frequency")
         plt.title("Bar graph:Blood types")
         plt.show()
```

Introduction to Statistics and Data Science

- Visualization
  - Pie graph
    - Each class is represented as the slice of the circle.
    - The bigger slice indicates the bigger relative frequency of the class
    - Generally, relative frequency is shown as percentage (%)

Introduction to Statistics and Data Science

- Practice



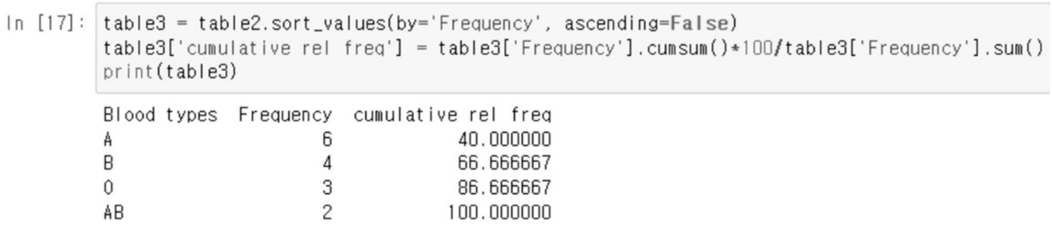
```
In [39]: plt.pie(table2.iloc[:,0],labels=table2.index)
         plt.title("Blood types")
         plt.show()
```

Introduction to Statistics and Data Science

- Visualization
  - Pareto graph
    - Sorted by the frequency in descending order
    - Also show cumulative relative frequencies as percentages

Introduction to Statistics and Data Science

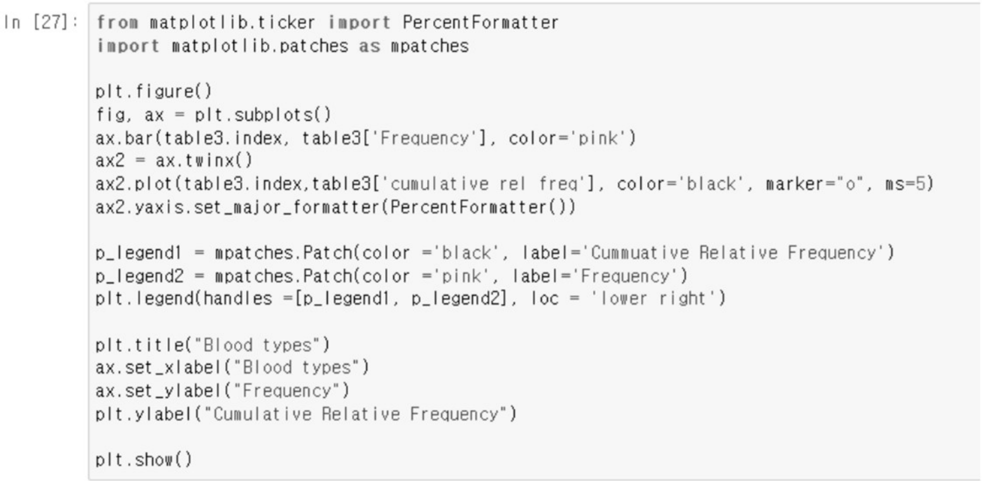
- Practice



```
In [17]: table3 = table2.sort_values(by='Frequency', ascending=False)
         table3['cumulative rel freq'] = table3['Frequency'].cumsum()*100/table3['Frequency'].sum()
         print(table3)

         Blood types  Frequency  cumulative rel freq
         A                    6            40.000000
         B                    4            66.666667
         O                    3            86.666667
         AB                   2           100.000000
```

Introduction to Statistics and Data Science

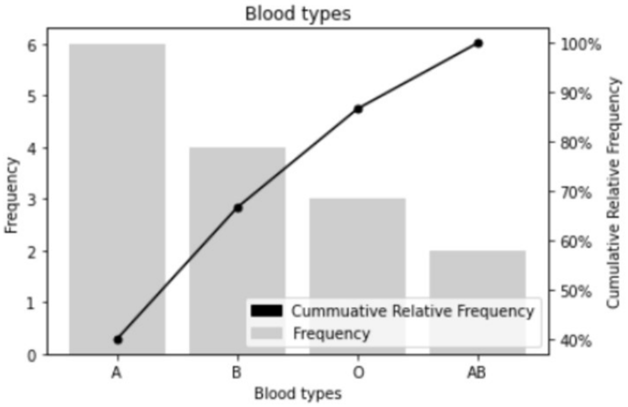
- Practice



```
In [27]: from matplotlib.ticker import PercentFormatter
         import matplotlib.patches as mpatches

         plt.figure()
         fig, ax = plt.subplots()
         ax.bar(table3.index, table3['Frequency'], color='pink')
         ax2 = ax.twinx()
         ax2.plot(table3.index,table3['cumulative rel freq'], color='black', marker="o", ms=5)
         ax2.yaxis.set_major_formatter(PercentFormatter())

         p_legend1 = mpatches.Patch(color ='black', label='Cummuative Relative Frequency')
         p_legend2 = mpatches.Patch(color ='pink', label='Frequency')
         plt.legend(handles =[p_legend1, p_legend2], loc = 'lower right')

         plt.title("Blood types")
         ax.set_xlabel("Blood types")
         ax.set_ylabel("Frequency")
         plt.ylabel("Cumulative Relative Frequency")

         plt.show()
```

Introduction to Statistics and Data Science

- Practice : Pareto chart



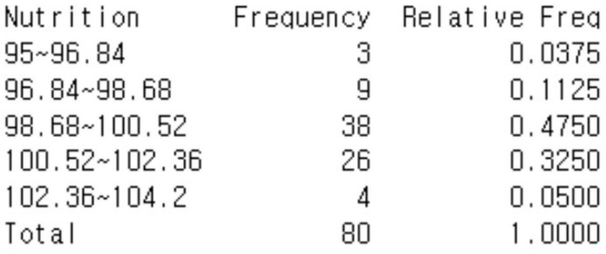
Introduction to Statistics and Data Science

- Frequency table
  - For continuous data
    - Class interval : A interval of a class, lower limit and upper limit should be shown.
    - Class representative value : median value of a class interval
    - Frequency : the number of observations in the class interval
    - Relative Frequency (RF)

$$RF = \frac{Frequency}{n}$$

Introduction to Statistics and Data Science

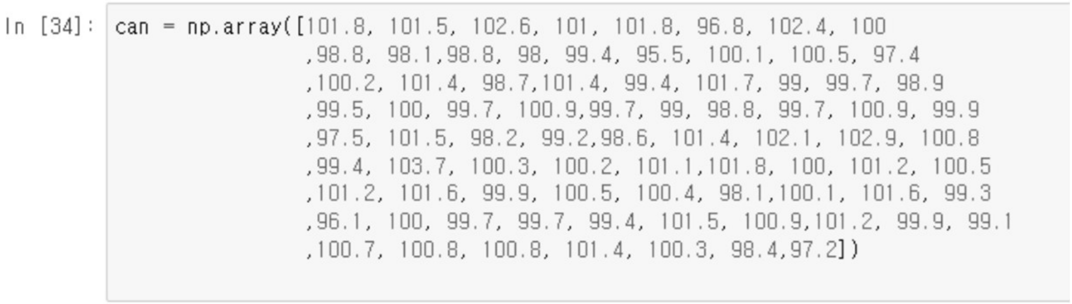
- Frequency table



```
Nutrition         Frequency  Relative Freq
95~96.84                  3         0.0375
96.84~98.68               9         0.1125
98.68~100.52             38         0.4750
100.52~102.36            26         0.3250
102.36~104.2              4         0.0500
Total                    80         1.0000
```

Introduction to Statistics and Data Science

- Practice



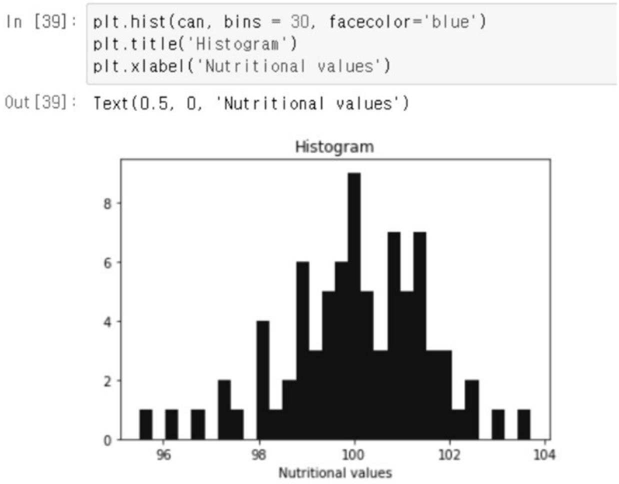
```
In [34]: can = np.array([101.8, 101.5, 102.6, 101, 101.8, 96.8, 102.4, 100
                         ,98.8, 98.1,98.8, 98, 99.4, 95.5, 100.1, 100.5, 97.4
                         ,100.2, 101.4, 98.7,101.4, 99.4, 101.7, 99, 99.7, 98.9
                         ,99.5, 100, 99.7, 100.9,99.7, 99, 98.8, 99.7, 100.9, 99.9
                         ,97.5, 101.5, 98.2, 99.2,98.6, 101.4, 102.1, 102.9, 100.8
                         ,99.4, 103.7, 100.3, 100.2, 101.1,101.8, 100, 101.2, 100.5
                         ,101.2, 101.6, 99.9, 100.5, 100.4, 98.1,100.1, 101.6, 99.3
                         ,96.1, 100, 99.7, 99.7, 99.4, 101.5, 100.9,101.2, 99.9, 99.1
                         ,100.7, 100.8, 100.8, 101.4, 100.3, 98.4,97.2])
```

Introduction to Statistics and Data Science

- Visualization
  - Histogram
    - X axis is representative values of bins
    - Y axis is frequencies or relative frequencies

Introduction to Statistics and Data Science

- Practice : Histogram



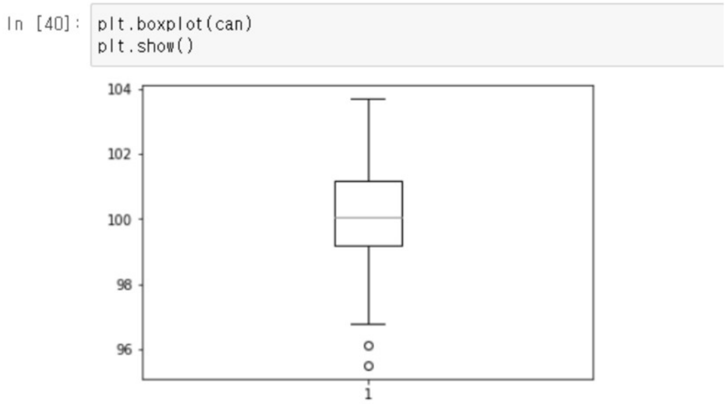
```
In [39]: plt.hist(can, bins = 30, facecolor='blue')
         plt.title('Histogram')
         plt.xlabel('Nutritional values')

Out[39]: Text(0.5, 0, 'Nutritional values')
```

Introduction to Statistics and Data Science

- Visualization
  - Box-whisker plot
    - Box is made with Q1, Q2, Q3 values
    - Whisker is made with the length of (IQR * 1.5)
    - Often, mean value is also shown.

Introduction to Statistics and Data Science

- Practice : Histogram



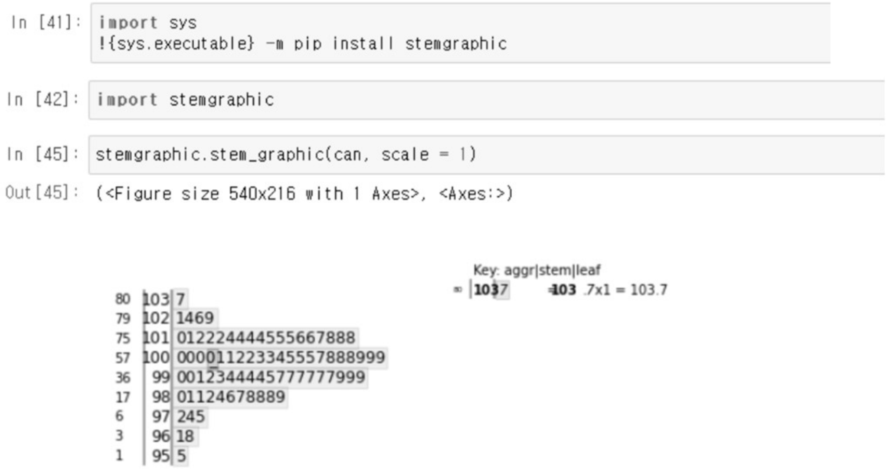
```
In [40]: plt.boxplot(can)
         plt.show()
```

Introduction to Statistics and Data Science

- Visualization
  - Stem-leaf plot
    - Stem is the bigger unit of the numbers.
    - Leaf is the other parts of the numbers.
    - Leaf does not have to be shown in order
    - Stem can be different by the users

Introduction to Statistics and Data Science

- Practice



```
In [41]: import sys
         !{sys.executable} -m pip install stemgraphic

In [42]: import stemgraphic

In [45]: stemgraphic.stem_graphic(can, scale = 1)
Out[45]: (<Figure size 540x216 with 1 Axes>, <Axes:>)
```

```
                                          Key: aggr|stem|leaf
                                       80 |1037      =103 .7x1 = 103.7
80 |103|7
79 |102|1469
75 |101|01222444455566788 8
57 |100|0000112233455578889 99
36 |  99|00123444457777779 99
17 |  98|01124678889
6  |  97|245
3  |  96|18
1  |  95|5
```