第1章　基本統計量の計算

量的データ X_1, ・・・, X_N が与えられたとき，その分布の状態を表す統計量として平均値(mean)とか分散(variance)などが算出される．平均値と分散は次式で与えられる．

$$mean = \frac{1}{N}\sum_{i=1}^{N} X_i, \qquad variance = \frac{1}{N}\sum_{i=1}^{N} (X_i - mean)^2$$

これは，データが次のように

```
data = [10, 30, 20, 50]
```

リスト型として与えられているとき，リスト1.1のスクリプトで算出できる．実行結果は，リスト1.2のようになる．

リスト1.1　基本統計量の計算

```
data = [10, 30, 20, 50]
print('data = ', data)
sum = 0
for v in data:
    sum += v
print('sum = ', sum)
mean = sum / len(data)
print('mean =', mean)

ssum = 0.0
for v in data:
    ssum += (v - mean) ** 2
var = ssum / len(data)
print('var = ', var)
```

リスト1.2　リスト1.1の実行結果

```
data =  [10, 30, 20, 50]
sum =  110
mean = 27.5
var =  218.75
```


Pythonには，いろいろな機能がパッケージによって提供されている．上の計算はパッケージnumpyに用意されている関数によっても算出できる．Numpyについてのドキュメントは

・https://docs.scipy.org/doc/

から入手できる．

リスト1.3のスクリプトでは，numpyを用いて平均値，分散，標準偏差を求めている．実行結果をリスト1.4に示す．

リスト1.3　Numpyを利用した統計量の計算

```
import numpy as np

data = [10, 30, 20, 50]
print('data = ', data)
v_sum = np.sum(data)
print('sum = ', v_sum)
v_mean = np.mean(data)
print('mean = ', v_mean)
v_var = np.var(data)
print('var = ', v_var)
v_std = np.std(data)
print('v_std = ', v_std, '    v_std ** 2 = ', v_std**2)
```

リスト1.4　リスト1.3の実行結果

```
data =  [10, 30, 20, 50]
sum =  110
mean =  27.5
var =  218.75
v_std =  14.79019945774904    v_std ** 2 =  218.75
```

統計量としては，ほかに最小値，中央値，最大値，第1四分位数，第2四分位数(中央値)，第3四分位数などがあるが，numpyを利用してこれらを求めるスクリプトをリスト1.5に，実行結果をリスト1.6に示す．

リスト1.5　四分位数などの計算

```
import numpy as np

data = [10, 30, 20, 50]
print ('data = ', data)
v_min = np.amin (data)
print ('v_min = ', v_min)
v_med = np.median (data)
print('median = ', v_med)
```

```
v_max = np.amax(data)
print('v_max = ', v_max)

Q1 = np.percentile(data, 25)
Q2 = np.percentile(data, 50)
Q3 = np.percentile(data, 75)
print('Q1 = {0}　Q2 = {1}　Q3 = {2} '.format(Q1, Q2, Q3))
```

リスト 1.6　リスト 1.5 の実行結果

```
data = [10, 30, 20, 50]
v_min = 10
median = 25.0
v_max = 50
Q1 = 17.5　Q2 = 25.0　Q3 = 35.0
```

リスト 1.5 のスクリプトにおけるデータは

```
data = [10, 30, 20, 50]
```

とリスト data で与えられていて，データ数は 4 個である．四分位数は大きさの順で考えて 1 番目と 2 番目の平均値，2 番目と 3 番目の平均値，3 番目と 4 番目の平均値であることが 1 つの考え方であるが，リスト 1.6 の出力を見ると，第 1 四分位数と第 3 四分位数はそのようにはなっていない．これは，区間 [1, 4] を 4 等分して加重平均を求めているからである．第 1 四分位数が大きさの順で 1 番目と 2 番目のデータの平均値になるようにするには，区間 [0.5, 4.5] において 4 等分する必要がある．

一般的に考える．N 個のデータ X_1, ・・・, X_N を大きさの順に並べたとき，i 番目のデータは数直線上の区間 $[1-0.5, N+0.5]$ において区間 $[i-0.5, i+0.5]$ を占めると考える(図 1.1)．

$100p$ パーセンタイルの値は，全区間 $[1-0.5, N+0.5]$ における左端の点から Np 右の位置の値として算出する．これは，リスト 1.7 の関数 MyPercentile によって求めることができる．

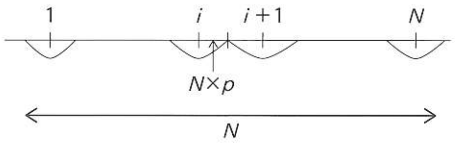

図 1.1　パーセンタイル値の算出．N 個のデータを大きさの順に並べたとき，第 i 番目のデータは数直線上の区間 $[1-0.5, N+0.5]$ において区間 $[i-0.5, i+0.5]$ を占めると考える．

リスト 1.7　区間 $[1-0.5, N+0.5]$ に基づく四分位数

```
def MyPercentile( d, p ):
    x = sorted(d)
    n = len(x)
    pos = n * (p / 100.0) + 0.5
    if pos <= 1:
        return x[0]
    elif pos >= n:
        return x[n - 1]
    else:
        i = int(pos)
        v = (pos - i) * x[i] + (i + 1 - pos) * x[i - 1]
        return v

data = [10, 30, 20, 50]
print('data = ', data)

myQ1 = MyPercentile(data, 25)
myQ2 = MyPercentile(data, 50)
myQ3 = MyPercentile(data, 75)
print('MyQ1 = {0}　MyQ2 = {1}　MyQ3 = {2} '.format(myQ1, myQ2, myQ3))
```

実行結果をリスト 1.8 に示す．4 個のデータの四分位数(25 パーセンタイル，50 パーセンタイル，および 75 パーセンタイル)が，大きさの順で考えて 1 番目と 2 番目の平均値，2 番目と 3 番目の平均値，3 番目と 4 番目の平均値になっていることが確認できる．

リスト 1.8　リスト 1.7 の実行結果

```
data = [10, 30, 20, 50]
```

```
MyQ1 = 15.0   MyQ2 = 25.0   MyQ3 = 40.0
```

パーセンタイルを求めるときには，データを大きさの順に並べる必要がある．並替えの関数として sorted と sort がある．この2つの関数の比較のため，リスト 1.9 では

```
data_1 = sorted(data)
data.sort()
```

と用いられている．関数 sorted は，並べ替えたものを実引数とは別のリストとして返すものである．関数 sort は，関数を呼び出したリストの並替えを行うものである．リスト 1.9 の実行結果をリスト 1.10 に示す．関数 sorted と sort の違いが確認できる．

リスト 1.9　並替えの関数 sort と sorted

```
data = [10, 30, 20, 50]
print ('data = ', data)

data_1 = sorted (data)
print ('data_1 = ', data_1)
print ('data   = ', data)

import copy

data_copy = copy.copy (data)
data.sort ()
print ('data   = ', data)
print ('data_copy = ', data_copy)
```

リスト 1.10　リスト 1.9 の実行結果

```
data =  [10, 30, 20, 50]
data_1 =  [10, 20, 30, 50]
data   =  [10, 30, 20, 50]
data   =  [10, 20, 30, 50]
data_copy =  [10, 30, 20, 50]
```

リスト 1.7 の関数 MyPercentile では，引数として読み込んだリストを関数 sorted で並べ替えたものを用いている．これは，一般に関数の引数としてリストを用いた場合，その引数への操作が呼出し元のリストに影響するからである．これを確かめるために関数 replace0 を用意した(リスト 1.11)．

リスト 1.11　関数の引数の値の変化

```
def replace0( d, c ):
    d[0] = c

def replace( d, c ):
    d = c

data = [10, 30, 20, 50]
print ('data = ', data)

replace0(data, 'a')
print ('data = ', data)

v = 1
print ('v = ', v)
replace( v, 2 )
print ('v = ', v)
```

リスト 1.11 の実行結果(リスト 1.12)を見れば，呼出し元のリスト data の要素の値が変化しているのがわかる．引数がリストでない関数 replace の場合は，関数内で値が変わっても呼出し元の実引数の値は変わらない．

リスト 1.12　リスト 1.11 の実行結果

```
data =  [10, 30, 20, 50]
data =  ['a', 30, 20, 50]
v =  1
v =  1
```

Python には種々の便利な機能が用意されているので，これらを活用すれば簡明なスクリプトを書くことができる．必要な機能を既存の Python ライブラリに見つけることができない場合は，必要な処理を行うための Python スクリプトを書けばよい．