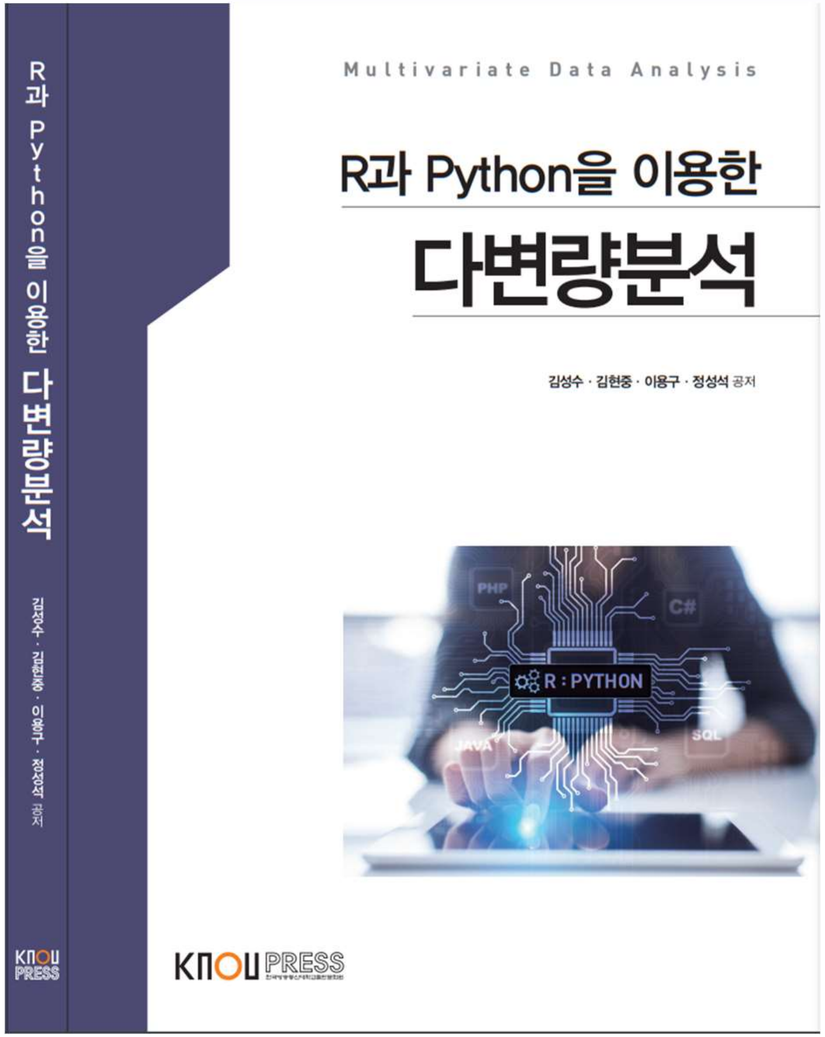


R과 Python을 이용한 다변량분석

김성수 · 김현중 · 이용구 · 정성석 공저

KNOU PRESS

Multivariate Data Analysis

R과 Python을 이용한

다변량분석

김성수 · 김현중 · 이용구 · 정성석 공저

KNOU PRESS

목차

Ch1. 다변량 시각화 (Multivariate Visualization)
; 1,2강
Ch2. 주성분분석 (Principal Component Analysis)
; 3강
Ch3. 인자분석 (Factor Analysis)
; 4, 5강
Ch4. 군집분석 (Cluster Analysis)
; 6,7 강
Ch5. 다차원분석(MDS)
; 8강
Ch6. 정준상관분석(Canonical Correlation)
; 9강
Ch7. 판별분석(Discriminant Analysis)
; 10, 11강
Ch8. 로지스틱 회귀분석(Logistic Regression)
; 12강
Ch9. 나무모형 (Tree Model)
; 13,14강
총정리 : 15강

1강. 다변량 시각화(1)

- 파이썬 기초
- 기술통계량 및 분할표

1. 파이썬 기초

파이썬 기초

- import libraries
- comment
- pip
- if
- for
- Functions
- class
- Read text(excel) files
- Numpy
- Pandas

참고문헌

(1) Statistics and Machine Learning in Python, E. Duchesnay, T. Lofstedt
ftp://ftp.cea.fr/pub/unati/people/educhesnay/pystatml/StatisticsMachineLearningPythonDraft.pdf

(2) Practical Machine Learning with R and Python , Tinniam V Ganesh
https://github.com/tvganesh/PracticalMachineLearningWithRandPython

import libraries

```
# 'generic import' of math module
import math
math.sqrt(25)

# import a function from math
from math import sqrt
sqrt(25)

# import multiple functions at once from math
from math import cos, floor
import os
os.getcwd()                    # in R, getwd()
os.chdir(“c:/data/pydata”)     # in R, setwd(“c:/data/rdata”)
import pandas as pd
bmi = pd.read_csv(“bmi.csv”)
```

주석 (Comments)

# comments in line

“””

Comments in sentences

Brian Heinold, A Practical Introduction to Python Programming, 2012.

Edouard Duchesnay, Tommy Löfstedt. Statistics and Machine Learning in Python, 2018.

“””

help (“modules”) : 설치된 모듈 목록 보기

```
…

_functools      csv          numpy             tabnanny
_hashlib        ctypes       numpydoc          tarfile
_heapq          curl         odbc              tblib
_imp            curses       odo               telnetlib
brain_typing    marshal      snowballstemmer   zict
brain_uuid      math         socket            zipapp
bs4             matplotlib   socketserver      zipfile
builtins        mccabe       socks             zipimport
bz2             menuinst     sockshandler      zlib
cProfile        mglearn      sortedcollections zmq

…
```

pip

pip : Python 패키지를 설치하고 관리하는 프로그램

예) 패키지 mglearn, graphviz 설치하기 (Graphviz : 구글 검색 참조)

Dos 창에서

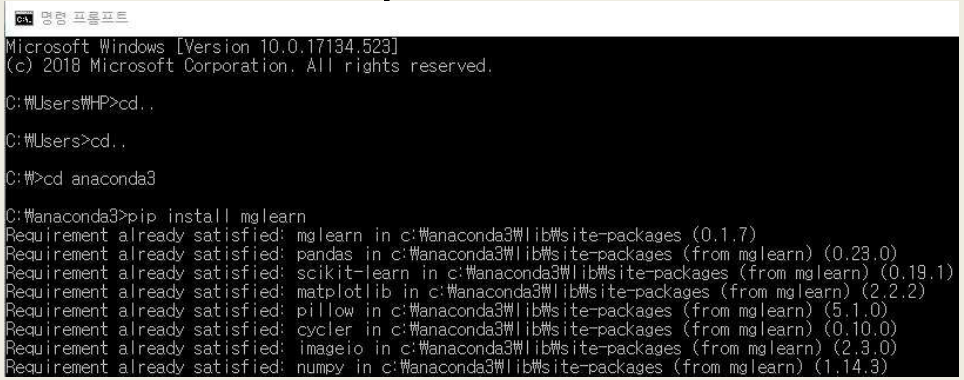
```
pip install mglearn
pip install graphviz
```

참고1 : python –m pip install –upgrade pip

참고2 : Path 설정하기

Windows 시스템 – 제어판 – 시스템 및 보완 – 시스템 – 고급 시스템 설정 – 환경변수– 시스템 변수에서 Path 편집–새로만들기에서

C:₩anaconda3₩, C:₩anaconda3₩Scripts



```
Microsoft Windows [Version 10.0.17134.523]
(c) 2018 Microsoft Corporation. All rights reserved.

C:₩Users₩HP>cd..

C:₩Users>cd..

C:₩>cd anaconda3

C:₩anaconda3>pip install mglearn
Requirement already satisfied: mglearn in c:₩anaconda3₩lib₩site-packages (0.1.7)
Requirement already satisfied: pandas in c:₩anaconda3₩lib₩site-packages (from mglearn) (0.23.0)
Requirement already satisfied: scikit-learn in c:₩anaconda3₩lib₩site-packages (from mglearn) (0.19.1)
Requirement already satisfied: matplotlib in c:₩anaconda3₩lib₩site-packages (from mglearn) (2.2.2)
Requirement already satisfied: pillow in c:₩anaconda3₩lib₩site-packages (from mglearn) (5.1.0)
Requirement already satisfied: cycler in c:₩anaconda3₩lib₩site-packages (from mglearn) (0.10.0)
Requirement already satisfied: imageio in c:₩anaconda3₩lib₩site-packages (from mglearn) (2.3.0)
Requirement already satisfied: numpy in c:₩anaconda3₩lib₩site-packages (from mglearn) (1.14.3)
```

basic operations

```
# Number
10**3          # 1,000
10 / 4           # 2.5
10 / float(4)    # 2.5
5 % 4              # modulo 1 – remainder
10 // 4            # floor division 2
# Boolean operations
# comparisons (these return True)
5 〉3
5 〉= 3
5 != 3
5 == 5
# boolean operations (these return True)
5 〉3 and 6 〉3 5 〉3 or 5 〈 3 not False
False or not False and True
```

data types

```
# determine the type of an object
type(2)          # returns 'int'
type(2.0)        # returns 'float'
type('two')          # returns 'str'
type(True)           # returns 'bool'
type(None)           # returns 'NoneType'

# check if an object is of a given type
isinstance(2.0, int)              # returns False
isinstance(2.0, (int, float)) # returns True

# convert an object to a given type
float(2)
int(2.9)
str(2.9)
# zero, None, and empty containers are converted to False
bool(0)
bool(None)
bool('')   # empty string
bool([])   # empty list
bool({})   # empty dictionary
```

Tuples, lists and dictionaries

The 3 basic data types in Python are : - Tuples - List - Dictionary

Tuples: Tuples are immutable python objects which are enclosed with paranthesis. Immutability implies that objects cannot be added or removed to tuples. Hence we cannot

add or remove elements from tuples. However a tuple can be removed using the del() commands

List: List are a sequence of disimilar objects enclosed within square brackets. Objects

can be added to lists using append() and deleted using remove()

Dictionary: Dictionaries are a name(key)-value pair enclosed within curly braces. The name- value pairs are separated using a ':'. The keys must be unique in the dictionary

The length of tuples, lists and dictionaries can be obtained with the len()

```
# Tuples are enclosed in paranthesis
mytuple=(1,3,7,6,"test")
print(mytuple)
# Lists are enclosed in square bracket
mylist = [1, 2, 7, 4, 12 ]
#Dictionary - These are similar to name-value pairs
mydict={'Name':'Ganesh','Age':54,'Occupation':'Engineer'}
print(mydict)
print(mydict['Age'])

# No of elements in tuples, lists and dictionaries can be got with len()
print("Length of tuple=",len(mytuple))
print("Length of list =", len(mylist))
print("Length of dictionary =",len(mydict))
```

Lists

```
# empty list
 empty =  []
 # empty = list()
 empty.append(23)
 empty.append(45)
 empty
 Out[20]:  [23, 45]
```

Lists

```
# list slicing [start:end:stride]
weekdays = ['mon','tues','wed','thurs','fri']
weekdays[0]   # element 0
weekdays[0:3] # elements 0, 1, 2
weekdays[:3]  # elements 0, 1, 2
weekdays[3:]  # elements 3, 4
weekdays[−1]  # last element (element 4)
weekdays[::2]  # every 2nd element (0, 2, 4)
weekdays[::−1] # backwards (4, 3, 2, 1, 0)

# sort a list
simpsons.sort()
simpsons.sort(reverse=True) # sort in reverse
simpsons.sort(key=len)      # sort by a key

# conatenate +, replicate *
[1, 2, 3] + [4, 5, 6]
["a"] * 2 + ["b"] * 3
```

Tuples

```
# create a tuple
digits = (0, 1, 'two') # create a tuple directly
digits = tuple([0, 1, 'two']) # create a tuple from a list

# examine a tuple
digits[2]           # returns 'two'
len(digits)         # returns 3
digits.count(0)  # counts the number of instances of that value (1)
digits.index(1)  # returns the index of the first instance of that value (1)
```

Dictionaries

```
# create a dictionary (two ways)
family = {'dad':'homer', 'mom':'marge', 'size':6}
family = dict(dad='homer', mom='marge', size=6)

# examine a dictionary
family['dad']  # returns 'homer'
len(family)     # returns 3
family.keys()   # returns list: ['dad', 'mom', 'size']
family.values() # returns list: ['homer', 'marge', 6]
family.items()  # returns list of tuples:
                # [('dad', 'homer'), ('mom', 'marge'), ('size', 6)]
'mom' in family # returns True
'marge' in family # returns False (only checks keys)
```

Set : 집합 – list 와 달리 중복과 순서가 없음

```
# create a set
languages = {'python', 'r', 'java'} # create a set directly
snakes = set(['cobra', 'viper', 'python']) # create a set from a list

# examine a set
len(languages) # returns 3
'python' in languages # returns True

# set operations 1
languages & snakes # returns intersection: {'python'}
languages | snakes  # returns union: {'cobra', 'r', 'java', 'viper', 'python'}
languages – snakes  # returns set difference: {'r', 'java'}
snakes – languages  # returns set difference: {'cobra', 'viper'}

 # set operations 2
s1 = {1,2,3,4,5}
s2 = {2,4,6}
print(s1.intersection(s2))   # 교집합 {2, 4}
print(s1.union(s2))          # 합집합
{1, 2, 3, 4, 5, 6}
```

```
print(s1.difference(s2))      # 차집합
{1, 3, 5}
```

Conditional statements (조건문)

```
x = 3
# if statement
if x 〉 0:
    print('positive')

# if/else statement
if x 〉 0:
    print('positive')
else:
    print('zero or negative')

# if/elif/else statement
if x 〉 0:
    print('positive')
elif x == 0:
   print('zero)
else:
   print('negative')
```

Loops (반복문)

```
# range returns a list of integers
range(0, 3) # returns [0, 1, 2]: includes first value but excludes second value
range(0, 5, 2) # returns [0, 2, 4]: third argument specifies the 'stride'

fruits = ['apple', 'banana', 'cherry']
for fruit in fruits:
    print(fruit.upper())

for fruit in fruits:
    if fruit == 'banana':
        print("Found the banana!")
        break  # exit the loop and skip the 'else' block
    else:
        # this block executes ONLY if the for loop completes without hitting 'break'
        print("Can't find the banana")

count = 0
while count < 5:
    print("This will print 5 times")
    count += 1       # equivalent to 'count = count + 1'
```

Functions

```
# define a function with no arguments and no return values
def print_text():
    print('this is text')

# call the function
print_text()

# define a function with one argument and no return values
def print_this(x):
    print(x)

# call the function
print_this(3)       # prints 3
n = print_this(3) # prints 3, but doesn't assign 3 to n
                   # because the function has no return statement
# define a function with one argument and one return value
def square_this(x):
    return x ** 2
# call the function
square_this(3) # prints 9
var = square_this(3) # assigns 9 to var, but does not print 9
```

Functions - 계속

```
# default arguments
def power_this(x, power=2):
   return x ** power

power_this(2) # 4
power_this(2, 3) # 8

# return two values from a single function
def min_max(nums):
   return min(nums), max(nums)

# return values can be assigned to a single variable as a tuple
nums = [1, 2, 3]
min_max_num = min_max(nums) # min_max_num = (1, 3)
```

Functions - lambda

- Lambda operations all you to create small anonymous function which computes something

```
# operations on list

a = [5,2,3,1,7]
b = [1,5,4,6,8]

# Create a lambda function to add 2 numbers
add_fct = lambda x,y:x+y

add_fct(a,b)
Out[12]: [5, 2, 3, 1, 7, 1, 5, 4, 6, 8]

# Add all elements of lists a and b
print(list(map(add_fct, a,b)))
[6, 7, 7, 7, 15]
```

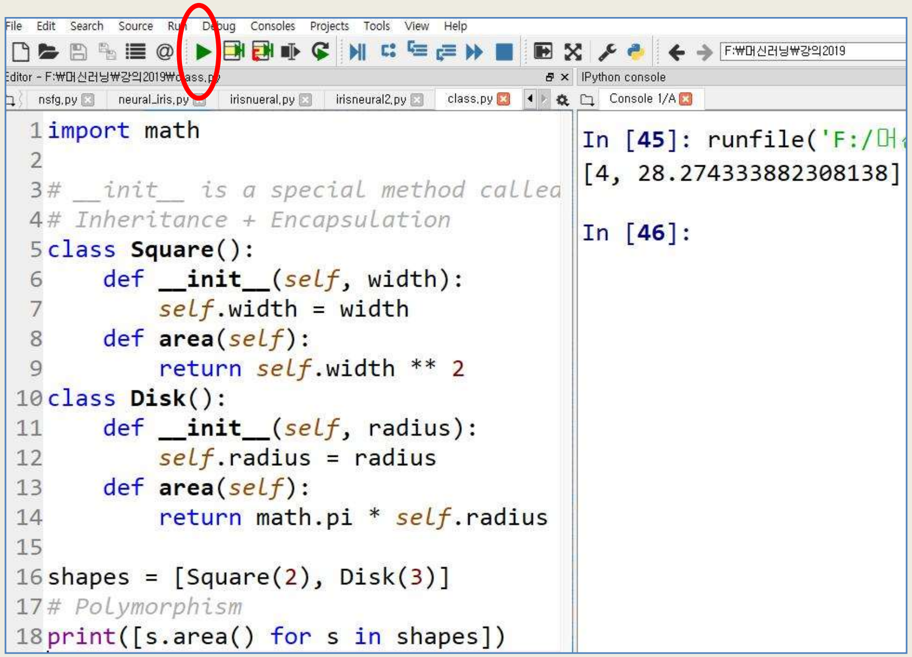
Ref: Practical machine learning with R and Python:3rd ed., Tinniam V Ganesh

Object oriented programming (OOP)

```
import math
# Inheritance + Encapsulation
class Square():
   def __init__(self, width):
            self.width = width

   def area(self):
        return self.width ** 2

class Disk():
   def __init__(self, radius):
       self.radius = radius

   def area(self):
       return math.pi * self.radius ** 2

shapes = [Square(2), Disk(3)]
# Polymorphism
print([s.area() for s in shapes])
```

수행결과

```
[4, 28.274333882308138]
```



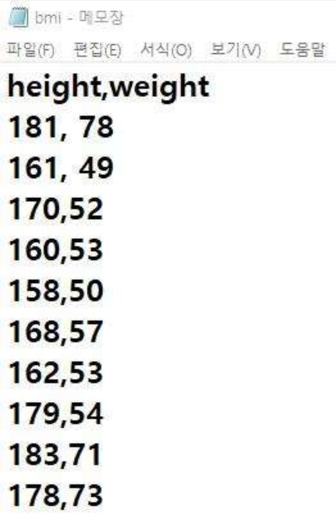
```
import math

# __init__ is a special method callea
# Inheritance + Encapsulation
class Square():
    def __init__(self, width):
        self.width = width
    def area(self):
        return self.width ** 2
class Disk():
    def __init__(self, radius):
        self.radius = radius
    def area(self):
        return math.pi * self.radius

shapes = [Square(2), Disk(3)]
# Polymorphism
print([s.area() for s in shapes])
```

```
In [45]: runfile('F:/머
[4, 28.274333882308138]

In [46]:
```

```
a= Square(2)

a.area()

Out[12]: 4
```

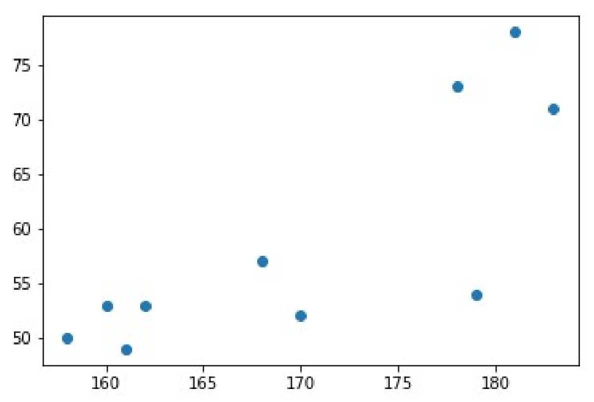
Read text file using pandas

```
import os
import pandas as pd
import matplotlib.pyplot as plt
# Set the current working directory
os.chdir(“c:/data/pydata”)
os.getcwd()     # 'c:₩₩data₩₩pydata'
# data = pd_read.csv(‘c:/data/pydata/bmi.csv’)
data = pd.read_csv(“bmi.csv”)
data.head()
Out[13]:
height  weight
0    181     78
1    161     49
2    170     52
3    160     53
4   158     50

weig = data[‘weight’]
heig = data[‘height’]
bmi = weig/(heig/100)**2
plt.scatter(heig, weig)
plt.show()
```



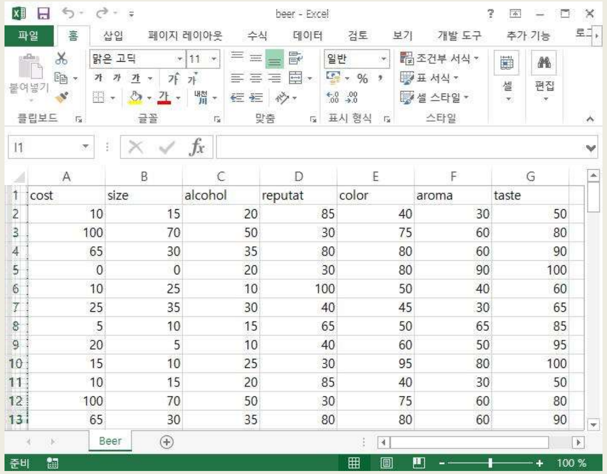
```
height,weight
181, 78
161, 49
170,52
160,53
158,50
168,57
162,53
179,54
183,71
178,73
```

bmi.csv



Read Excel file using pandas

```
import os
import pandas as pd
import matplotlib.pyplot as plt
# Set the current working directory
os.chdir(“c:/data/pydata”)
os.getcwd()    # 'c:₩₩data₩₩pydata'
beer = pd.read_excel(“beer.xlsx, sheet_name=‘Beer’)
beer.head()
Out[9]:
   cost  size  alcohol  reputat  color  aroma  taste
0    10    15       20       85     40     30     50
1   100    70       50       30     75     60     80
2    65    30       35       80     80     60     90
3     0     0       20       30     80     90    100

4    10    25       10      100     50     40     60
beer['cost']   # beer.cost
Out[12]:
0    10
1   100
2    65
3     0
4    10
5    25
```



	A	B	C	D	E	F	G
1	cost	size	alcohol	reputat	color	aroma	taste
2	10	15	20	85	40	30	50
3	100	70	50	30	75	60	80
4	65	30	35	80	80	60	90
5	0	0	20	30	80	90	100
6	10	25	10	100	50	40	60
7	25	35	30	40	45	30	65
8	5	10	15	65	50	65	85
9	20	5	10	40	60	50	95
10	15	10	25	30	95	80	100
11	10	15	20	85	40	30	50
12	100	70	50	30	75	60	80
13	65	30	35	80	80	60	90

beer.xlsx

Numpy

NumPy is one of the most fundamental package for scientific computing with Python. Numpy includes the support for handling large, multi-dimensional arrays and matrices, along with a large collection of high-level mathematical functions to operate on these arrays.

```
import numpy as np
#Create a 1d numpy array
data1 = [6, 7.5, 8, 0, 1]

arr1 = np.array(data1)
print(arr1)
```

Python의 기본 자료구조
- list / tuple / dictionary / set

: The Python languge was not initially designed for numerical computing
: 1995 matrix-sig group
: 2006 numpy

Numpy

```
# Create numpy array in a single line
import numpy as np
arr1= np.array([6, 7.5, 8, 0, 1])
#Print the array
print(arr1)

### 2D array

#Create a 2d numpy array
import numpy as np
data2 = [[1, 2, 3, 4], [5, 6, 7, 8]]
arr2 = np.array(data2)
# Print the 2d array
print(arr2)
```

2. 기술통계량 및 분할표

기술통계량 및 분할표

예제) 한 설문조사에서 다음 6개 문항에 대하여 표본 추출된 40명을 대상으로 조사한 자료가 다음과 같이 정리되어 있다.

예 제

문항 1. 귀하의 성별은?

1) 남자 2) 여자

문항 2. 결혼하셨습니까?

1) 미혼 2) 기혼 3) 이혼

문항 3. 귀하의 나이는? (단위: 세)

문항 4. 귀하의 직업은?

1) 회사원 2) 공무원 3) 노무자 4) 정치가

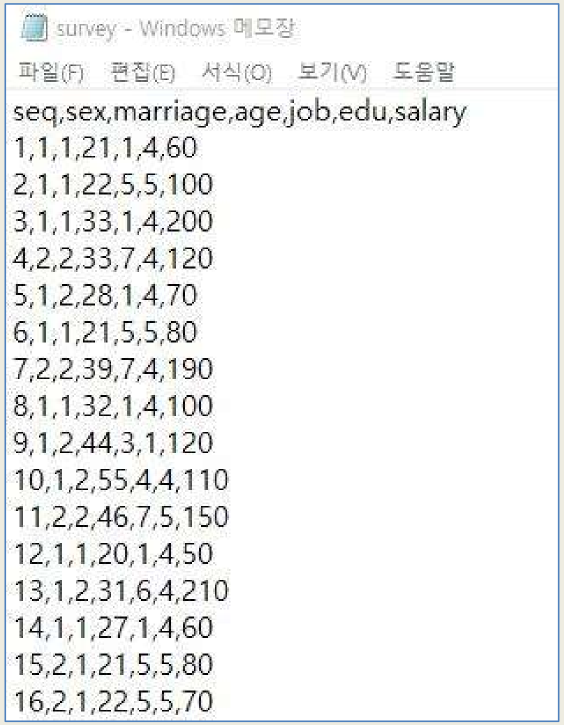
5) 학생 6) 기업가 7) 주부 8) 기타

문항 6. 가족의 월수입은? (단위: 만원)

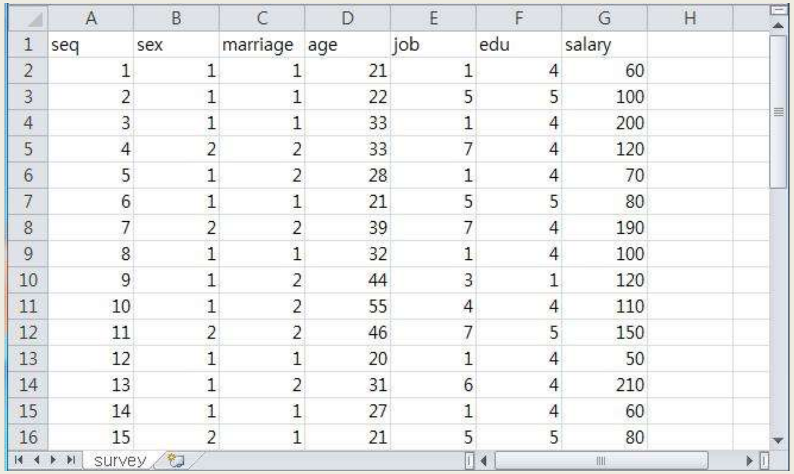
R을 이용한 기술통계량 구하기

◆ csv 파일 읽기

```
> survey = read.csv(“c:/data/mva/survey.csv”)
```



```
seq,sex,marriage,age,job,edu,salary
1,1,1,21,1,4,60
2,1,1,22,5,5,100
3,1,1,33,1,4,200
4,2,2,33,7,4,120
5,1,2,28,1,4,70
6,1,1,21,5,5,80
7,2,2,39,7,4,190
8,1,1,32,1,4,100
9,1,2,44,3,1,120
10,1,2,55,4,4,110
11,2,2,46,7,5,150
12,1,1,20,1,4,50
13,1,2,31,6,4,210
14,1,1,27,1,4,60
15,2,1,21,5,5,80
16,2,1,22,5,5,70
```



	A	B	C	D	E	F	G	H
1	seq	sex	marriage	age	job	edu	salary	
2	1	1	1	21	1	4	60	
3	2	1	1	22	5	5	100	
4	3	1	1	33	1	4	200	
5	4	2	2	33	7	4	120	
6	5	1	2	28	1	4	70	
7	6	1	1	21	5	5	80	
8	7	2	2	39	7	4	190	
9	8	1	1	32	1	4	100	
10	9	1	2	44	3	1	120	
11	10	1	2	55	4	4	110	
12	11	2	2	46	7	5	150	
13	12	1	1	20	1	4	50	
14	13	1	2	31	6	4	210	
15	14	1	1	27	1	4	60	
16	15	2	1	21	5	5	80	

R을 이용한 기술통계량 구하기

```
> survey = read.csv("c:/data/mva/survey.csv")
> head(survey,3)
  seq sex marriage age job edu salary
1   1   1        1  21   1   4     60
2   2   1        1  22   5   5    100
3   3   1        1  33   1   4    200
> mean(survey$age)
[1] 34.275
> sd(survey$age)
[1] 11.60236
> survey$sex = factor(survey$sex, levels=c(1:2), labels=c("Male", "Female"))
> survey$marriage = factor(survey$marriage, levels=c(1:3),
                          labels=c("Unmarried","Married","Divorced"))
> survey$job = factor(survey$job, levels=c(1:8),
                     labels=c('a','b','c','d','e','f','g','other'))
> survey$edu = ordered(survey$edu, levels=c(1:5),
                      labels=c('none','elem','med','high','college'))
```

R을 이용한 기술통계량 구하기

```
> summary(survey[,-1])
      sex             marriage         age                 job             edu
 Male  :27    Unmarried:15    Min.   :20.00    a        :12    none   : 1
 Female:13    Married  :23    1st Qu.:24.75    f        : 7    elem   : 1
              Divorced : 2    Median :32.00    g        : 6    med    : 3
                              Mean   :34.27    c        : 5    high   :19
                              3rd Qu.:42.50    e        : 5    college:16
                              Max.   :59.00    d        : 3
                                               (Other): 2
     salary
 Min.   : 50.0
 1st Qu.: 77.5
 Median :105.0
 Mean   :130.2
 3rd Qu.:175.0
 Max.   :349.0
```

R을 이용한 그룹별 기술통계량 구하기

나이에 대한 (성별, 결혼상태, 성별x결혼상태) 평균 및 표준편차

```
〉tapply(survey$age, survey$sex, mean)
    Male   Female
33.96296 34.92308

〉with(survey, tapply(age, sex, sd))
    Male   Female
11.96945 11.24323

〉with(survey, tapply(age, marriage, mean))
Unmarried Married Divorced
24.66667 39.13043 50.50000

〉with(survey, tapply(age, marriage, sd))
Unmarried   Married  Divorced
 4.151879 10.467718 12.020815
```

R을 이용한 그룹별 기술통계량 구하기

- 나이에 대한 (성별, 결혼상태, 성별x결혼상태) 평균 및 표준편차

```
> sex_ma = list(survey$sex, survey$marriage)
> table(sex_ma)
          sex_ma.2
sex_ma.1  Unmarried Married Divorced
  Male           10      15        2
  Female          5       8        0

> with(survey, tapply(age, sex_ma, mean))
       Unmarried  Married Divorced
Male        24.8 37.86667     50.5
Female      24.4 41.50000       NA

> with(survey, tapply(age, sex_ma, sd))
     Unmarried   Married Divorced
Male    4.709329 11.230486 12.02082
Female  3.209361  9.071147       NA
```

R을 이용한 기술통계량 구하기

빈도표 및 분할표(성별, 교육)

```
> table(survey$sex)
  Male Female
    27      13
> table(survey$edu)
   none    elem     med    high college
      1       1       3      19      16
> table(survey$sex, survey$edu)
         none elem med high college
  Male      1    1   1   13      11
  Female    0    0   2    6       5
> sex_edu = table(survey$sex, survey$edu)
> summary(sex_edu)
Number of cases in table: 40
Number of factors: 2
Test for independence of all factors:
        Chisq = 2.5781, df = 4, p-value = 0.6307
        Chi-squared approximation may be incorrect
```

파이썬을 이용한 기술통계량 구하기

```
import numpy as np
import pandas as pd
# 데이터 읽기
survey = pd.read_csv("c:/data/mva/survey.csv")
survey.head(3)
Out[4]:
   seq  sex  marriage  age  job  edu  salary
0    1    1         1   21    1    4      60
1    2    1         1   22    5    5     100
2    3    1         1   33    1    4     200

# 평균 구하기
survey["age"].mean()
Out[5]: 34.275
# 표준편차 구하기
survey["age"].std()
Out[6]: 11.602359397542536
```

파이썬을 이용한 기술통계량 구하기

```
# 범주형 변수로 변환하기
survey["sex"] = survey["sex"].astype("category")
survey["job"] = survey["job"].astype("category")
survey["edu"] = survey["edu"].astype("category")
survey.marriage = survey.marriage.astype("category")

# 연속인 변수의 기술통계량 구하기
survey.iloc[:, 1:].describe()
Out[7]:
               age      salary
count    40.000000    40.00000
mean     34.275000   130.22500
std      11.602359    72.19258
min      20.000000    50.00000
25%      24.750000    77.50000
50%      32.000000   105.00000
75%      42.500000   175.00000
max      59.000000   349.00000
```

파이썬을 이용한 그룹별 기술통계량 구하기

- **나이에 대한 (성별, 결혼상태, 성별x결혼상태) 평균 및 표준편차**

```
agestat_by_sex = survey.groupby("sex")["age"].describe()
agestat_by_sex
Out[17]:
  count      mean        std  min  25%  50%  75%  max
sex
1  27.0 33.962963 11.969453 20.0 24.0 32.0 43.0 59.0
2  13.0 34.923077 11.243232 21.0 26.0 33.0 41.0 56.0

agestat_by_sex["mean"]   # 표준편차 : std
Out[18]:
sex
1 33.962963
2 34.923077
Name: mean, dtype: float64
```

파이썬을 이용한 그룹별 기술통계량 구하기

나이에 대한 (성별, 결혼상태, 성별x결혼상태) 평균 및 표준편차

```
# (sex, marriage)를 그룹으로 age의 기술통계량 구하기
agestat_by_sex_marriage = survey.groupby(["sex","marriage"])["age"].describe()
agestat_by_sex_marriage
Out[19]:
              count       mean        std   min    25%   50%    75%   max
sex marriage
1   1          10.0  24.800000   4.709329  20.0  21.00  23.5  26.75  33.0
    2          15.0  37.866667  11.230486  22.0  31.00  34.0  46.50  56.0
    3           2.0  50.500000  12.020815  42.0  46.25  50.5  54.75  59.0
2   1           5.0  24.400000   3.209361  21.0  22.00  24.0  26.00  29.0
    2           8.0  41.500000   9.071147  27.0  37.50  41.0  46.75  56.0
agestat_by_sex_marriage["mean"]    # 표준편차 : std
Out[21]:
sex  marriage
1    1          24.800000
     2          37.866667
     3          50.500000
2    1          24.400000
     2          41.500000
Name: mean, dtype: float64
```

파이썬을 이용한 기술통계량 구하기

빈도표 및 분할표(성별, 교육)

```
sex_freq = pd.crosstab(index=survey.sex, columns='count')
sex_freq
Out[31]:
col_0 count
  sex
1  27
2  13

# (sex, edu)의 분할표 구하기
sex_edu_table = pd.crosstab(index=survey.sex, columns=survey.edu)
sex_edu_table
Out[35]:
edu    1 2 3   4   5
sex
1      1 1 1  13  11
2      0 0 2   6   5
```

파이썬을 이용한 기술통계량 구하기

빈도표 및 분할표(성별, 교육)

```
# (sex, edu)의 분할표-카이제곱 검정
from scipy.stats import chi2_contingency

chi2_contingency(sex_edu_table)
Out[37]:
(2.578097665816964,
 0.6307078881367414,
 4,
 array([[ 0.675, 0.675, 2.025, 12.825, 10.8 ],
        [  0.325, 0.325, 0.975, 6.175, 5.2 ]]))

# help(chi2_contingency)
```

다음시간에는

2강 다변량 시각화(2)

수고했습니다.