데이터 시각화와 통계

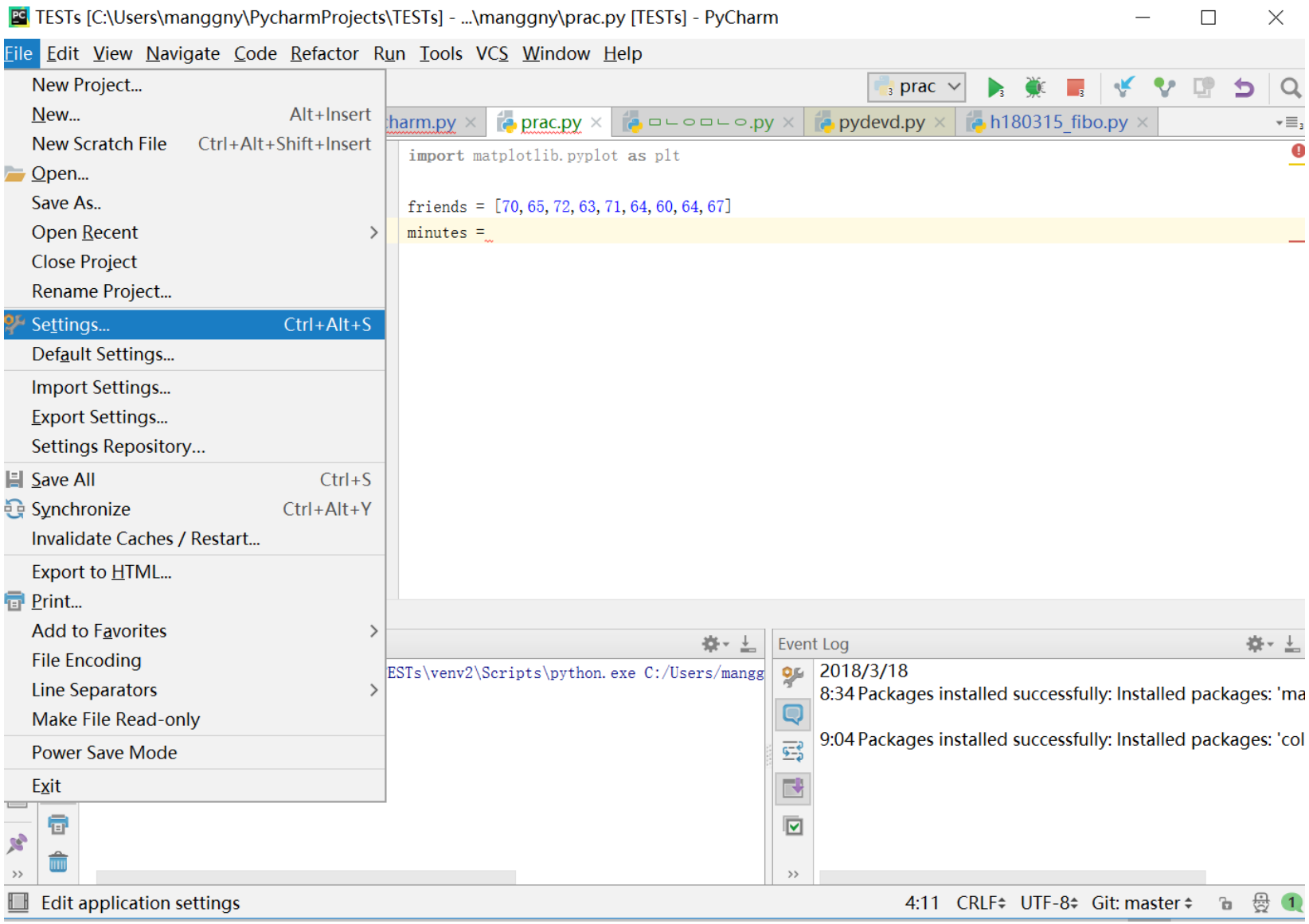
임재근

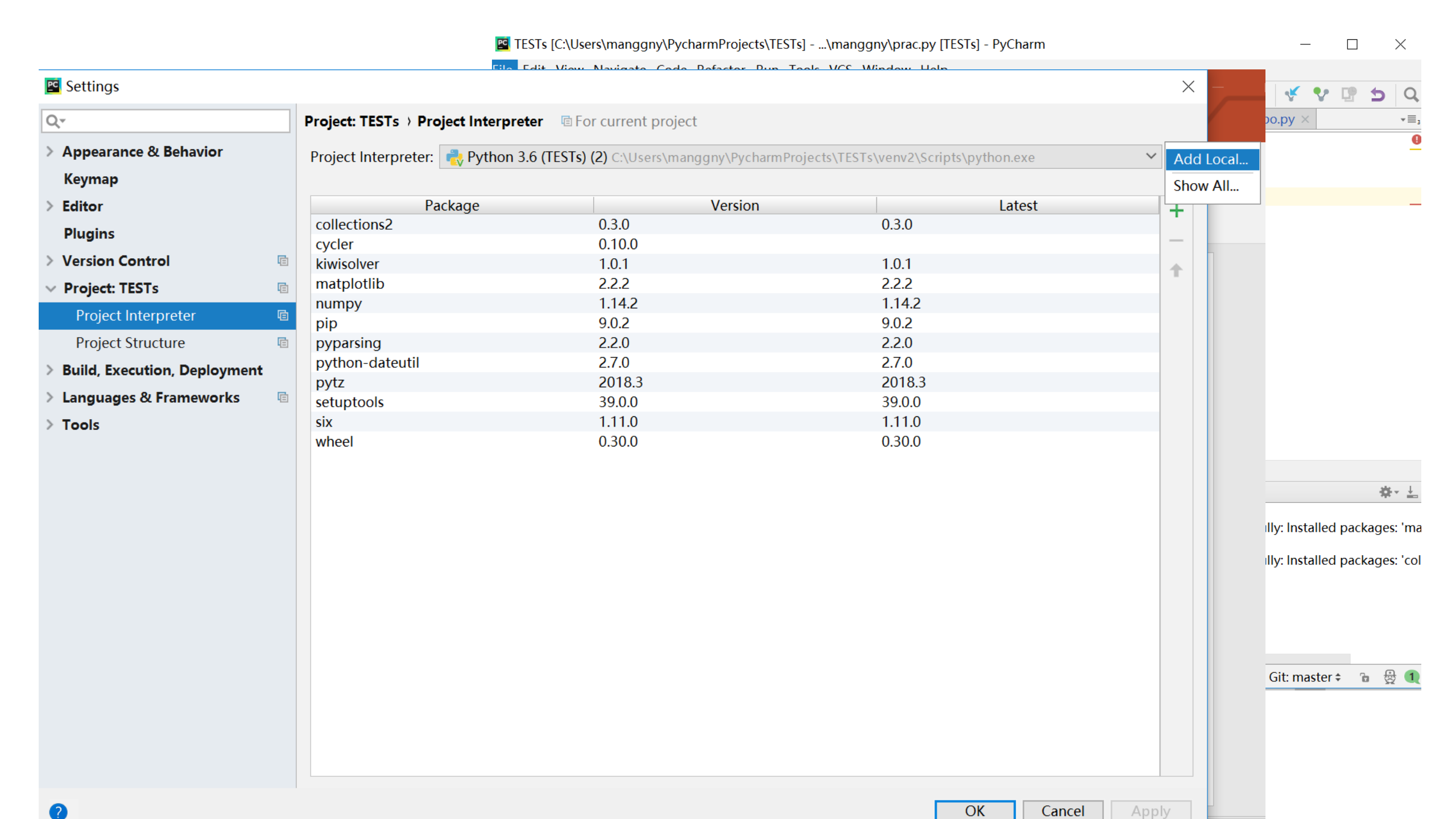
데이터 시각화란?

- 목적 : 데이터의 탐색과 전달.

- 우선 환경설정을 같이 해 봅시다!

환경설정


TESTs [C:\Users\manggny\PycharmProjects\TESTs] - ...\manggny\prac.py [TESTs] - PyCharm
File Edit View Navigate Code Refactor Run Tools VCS Window Help
New Project...
New... Alt+Insert
New Scratch File Ctrl+Alt+Shift+Insert
Open...
Save As..
Open Recent
Close Project
Rename Project...
Settings... Ctrl+Alt+S
Default Settings...
Import Settings...
Export Settings...
Settings Repository...
Save All Ctrl+S
Synchronize Ctrl+Alt+Y
Invalidate Caches / Restart...
Export to HTML...
Print...
Add to Favorites
File Encoding
Line Separators
Make File Read-only
Power Save Mode
Exit
prac
prac.py
pydevd.py
h180315_fibo.py
import matplotlib.pyplot as plt
friends = [70, 65, 72, 63, 71, 64, 60, 64, 67]
minutes =
ESTs\venv2\Scripts\python.exe C:/Users/mangg
Event Log
2018/3/18
8:34 Packages installed successfully: Installed packages: 'ma
9:04 Packages installed successfully: Installed packages: 'col
Edit application settings
4:11 CRLF UTF-8 Git: master


TESTs [C:\Users\manggny\PycharmProjects\TESTs] - ...\manggny\prac.py [TESTs] - PyCharm
Settings
Appearance & Behavior
Keymap
Editor
Plugins
Version Control
Project: TESTs
Project Interpreter
Project Structure
Build, Execution, Deployment
Languages & Frameworks
Tools
Project: TESTs › Project Interpreter
For current project
Project Interpreter: Python 3.6 (TESTs) (2) C:\Users\manggny\PycharmProjects\TESTs\venv2\Scripts\python.exe
Add Local...
Show All...
Package
Version
Latest
collections2 0.3.0 0.3.0
cycler 0.10.0
kiwisolver 1.0.1 1.0.1
matplotlib 2.2.2 2.2.2
numpy 1.14.2 1.14.2
pip 9.0.2 9.0.2
pyparsing 2.2.0 2.2.0
python-dateutil 2.7.0 2.7.0
pytz 2018.3 2018.3
setuptools 39.0.0 39.0.0
six 1.11.0 1.11.0
wheel 0.30.0 0.30.0
OK
Cancel
Apply
Git: master

환경설정

- 아나콘다 다운로드 후, 설치 위치를 기본설정대로 c/user/사용자이름/anaconda 로 설치후, 인터프리터를 아나콘다 위치에 있는 python.exe로 지정 !

그래프 그리기

- 막대 그래프 그리기!!

```
import matplotlib.pyplot6 as plt
#from matplotlib import pyplot as plt

movies=['Annie','Ben-hur','Casablanca','gandhi','west side
story']
num_oscars = [5,11,3,8,10]

xs = [i+0.1 for i, _ in enumerate(movies)]

# 축의 위치가 xs이고, 높이가 num_oscars인 막대를 그리자
plt.bar(xs,num_oscars)
plt.ylabel("# of academy awards")
plt.title("my favarite movies")

#막대의 가운데에 오도록 영화 제목 레이블을 달자
plt.xticks([i+0.1 for i, _ in enumerate(movies)],movies)
plt.show()
```

- Enumerate(x)
- 리스트 x의 인덱스와 원소 값을 인용.
- 예) for i, element in enumerate(some_list):

    do~~ on element

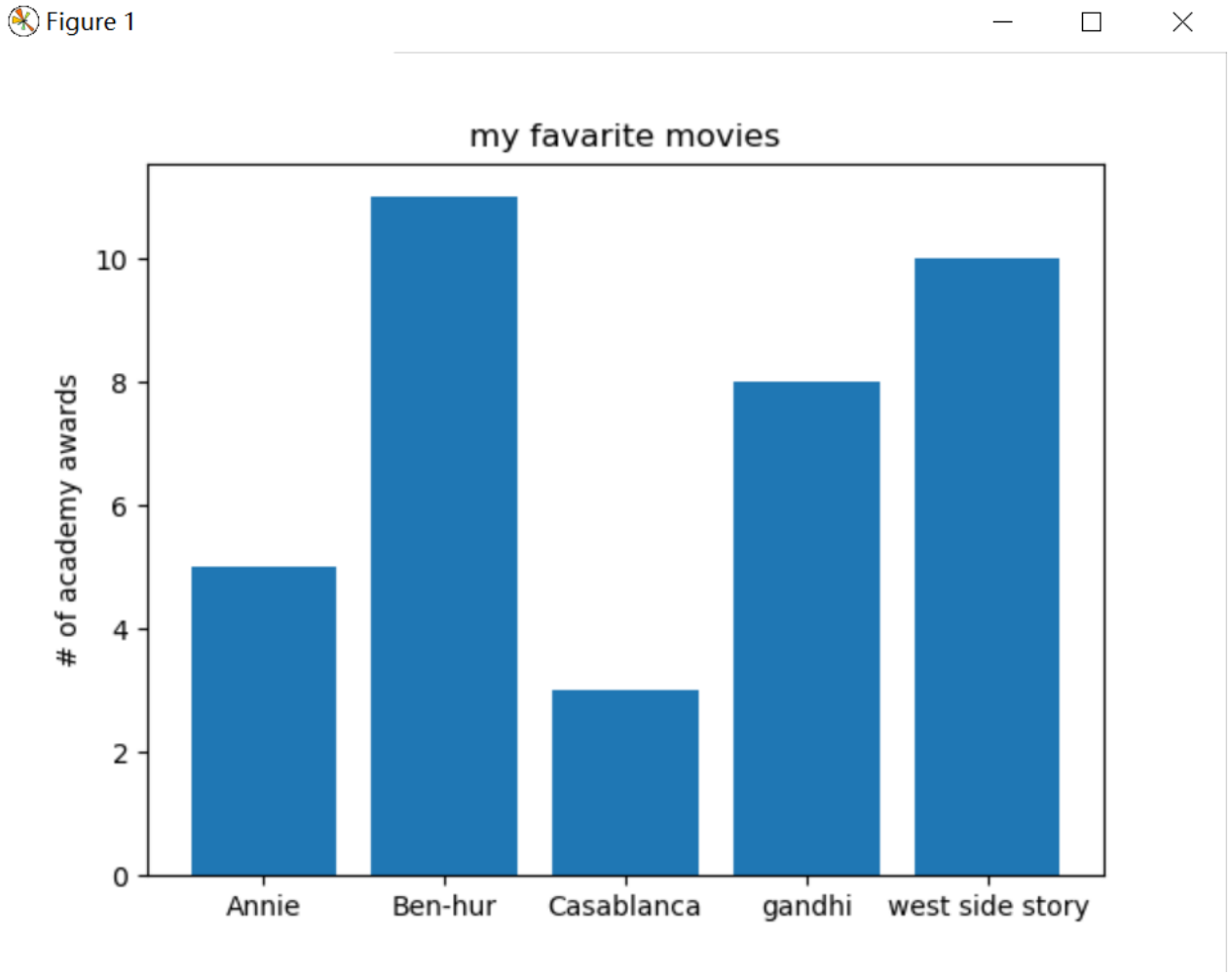
#i는 인덱스, element는 원소

(range를 사용하지 않는 방법으로, pythonic한 방법)

그래프 그리기

- 막대 그래프 그리기!!

```
import matplotlib.pyplot6 as plt
#from matplotlib import pyplot as plt

movies=['Annie','Ben-hur','Casablanca','gandhi','west side
story']
num_oscars = [5,11,3,8,10]

xs = [i+0.1 for i, _ in enumerate(movies)]

# 축의 위치가 xs이고, 높이가 num_oscars인 막대를 그리자
plt.bar(xs,num_oscars)
plt.ylabel("# of academy awards")
plt.title("my favarite movies")

#막대의 가운데에 오도록 영화 제목 레이블을 달자
plt.xticks([i+0.1 for i, _ in enumerate(movies)],movies)
plt.show()
```



그래프 그리기

- 히스토그램 그리기!
- Lambda 表达式

对于简单的函数，也存在一种简便的表示方式，即：lambda表达式

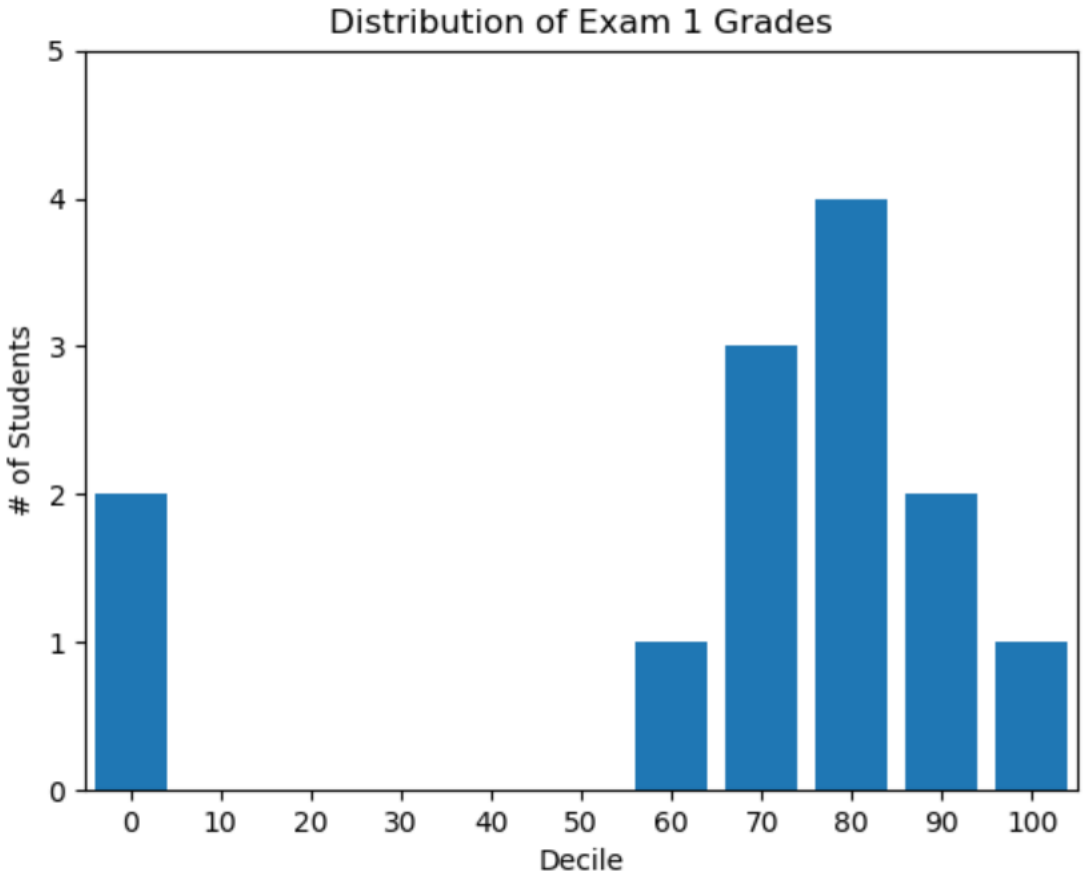
```
# ###################### 普通函数 ######################
# 定义函数（普通方式）
def func(arg):
    return arg + 1

# 执行函数
result = func(123)

# ###################### lambda ######################

# 定义函数（lambda表达式）
my_lambda = lambda arg : arg + 1

# 执行函数
result = my_lambda(123)
```

lambda存在意义就是对简单函数的简洁表示

그래프 그리기

- 히스토그램 코드!

```
import matplotlib.pyplot as plt
import collections

grades = [83,95,91,87,70,0,85,82,100,67,73,77,0]
decile = lambda grade: grade // 10*10

histogram = collections.Counter(decile(grade) for grade in grades)

print(histogram)

plt.bar([x for x in histogram.keys()],histogram.values(),8)
# 앞에부터 막대 위치, 막대 높이 그리고 막대 너비.

plt.axis([-5,105,0,5]) # x축은 -5~105, y축은 0~5

plt.xticks([10*i for i in range(11)])
plt.xlabel("Decile")
plt.ylabel("# of Students")
plt.title("Distribution of Exam 1 Grades")
plt.show()
```



그래프 그리기

- 선 그래프 그리기!

```
import matplotlib.pyplot as plt

variance = [1,2,4,8,16,32,64,128,256]
bias_squared = [256,128,64,32,16,8,4,2,1]
total_error = [x+y for x,y in
zip(variance,bias_squared)]
xs = [i for i, _ in enumerate(variance)]

plt.plot(xs,variance,'g-',label='variance') # 초록
실선
plt.plot(xs,bias_squared,'r-',label = 'bias^2') #
붉은 실선
plt.plot(xs,total_error,'b:',label='total error')
#파랑 점선

plt.legend(loc=9) # 각 시리즈에 이미 라벨을 달아놔
서, 레전드는 위치만 정해준다.
plt.xlabel("model complexity")
plt.title("The Bias-Variance Tradeoff")
plt.show()
```

- 举例：
  m = [[1,2,3], [4,5,6], [7,8,9]]
  n = [[2,2,2], [3,3,3], [4,4,4]]
  p = [[2,2,2], [3,3,3,]

zip(m, n)将返回([1, 2, 3], [2, 2, 2]), ([4, 5, 6], [3, 3, 3]), ([7, 8, 9], [4, 4, 4])

m[0], n[0]	m[1], n[1]	m[2], n[2]
[1,2,3] [2,2,2]	[4,5,6] [3,3,3]	[7,8,9] [4,4,4]

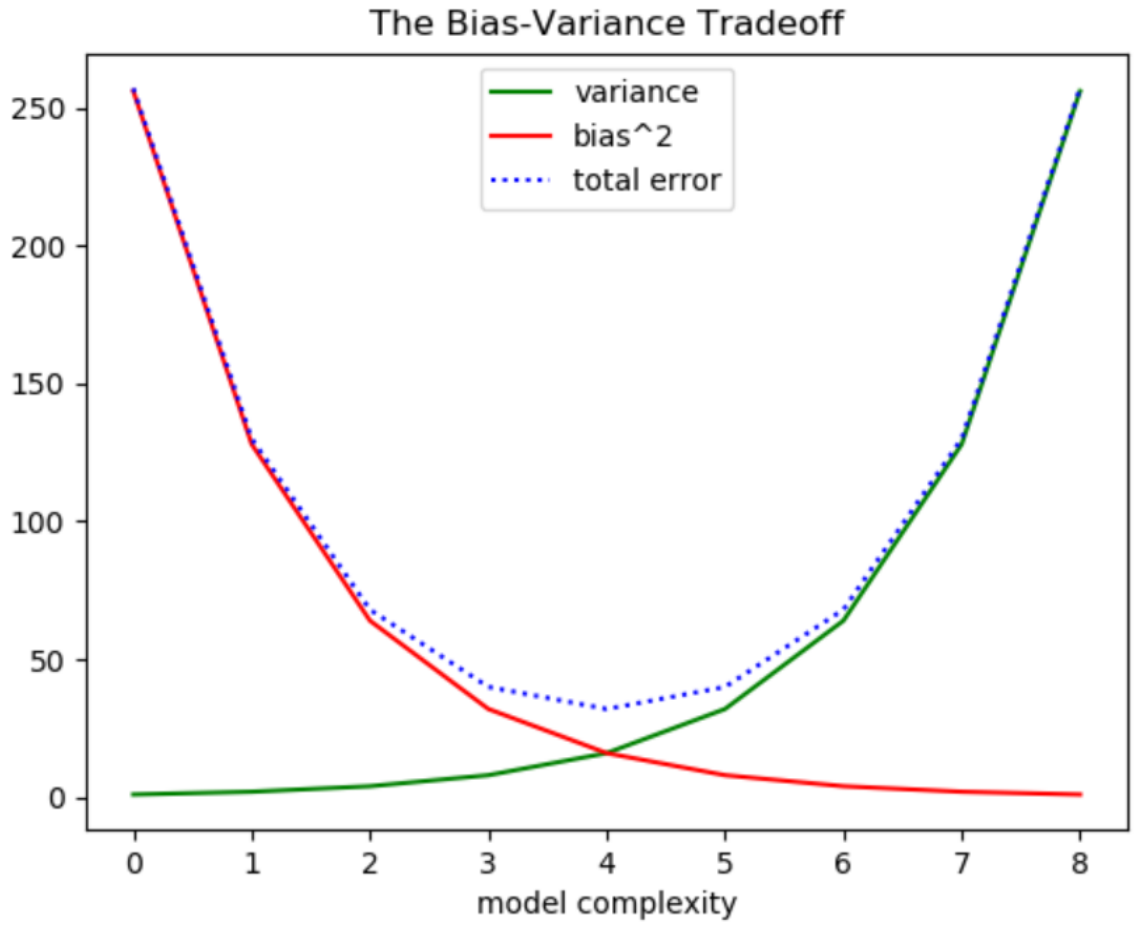
zip(m, p)将返回([1, 2, 3], [2, 2, 2]), ([4, 5, 6], [3, 3, 3])

m[0], n[0]	m[1], n[1]	m[2], n[2]
[1,2,3] [2,2,2]	[4,5,6] [3,3,3]	*[7,8,9]*

그래프 그리기

- 선 그래프 그리기!

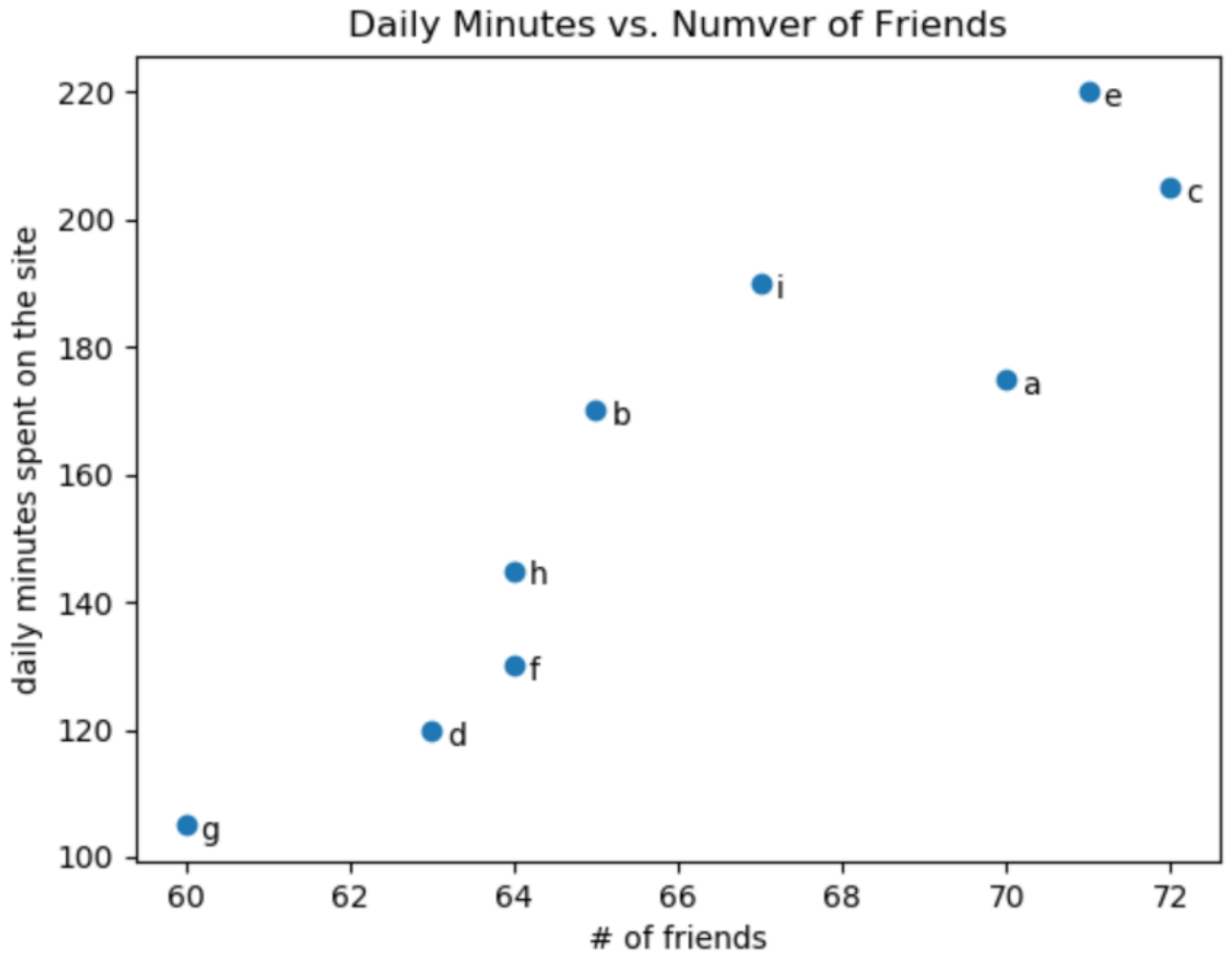
```
import matplotlib.pyplot as plt

variance = [1,2,4,8,16,32,64,128,256]
bias_squared = [256,128,64,32,16,8,4,2,1]
total_error = [x+y for x,y in
zip(variance,bias_squared)]
xs = [i for i, _ in enumerate(variance)]

plt.plot(xs,variance,'g-',label='variance') # 초록
실선
plt.plot(xs,bias_squared,'r-',label = 'bias^2') #
붉은 실선
plt.plot(xs,total_error,'b:',label='total error')
#파랑 점선

plt.legend(loc=9) # 각 시리즈에 이미 라벨을 달아놔
서, 레전드는 위치만 정해준다.
plt.xlabel("model complexity")
plt.title("The Bias-Variance Tradeoff")
plt.show()
```



산점도

- 코드

```
import matplotlib.pyplot as plt

friends = [70,65,72,63,71,64,60,64,67]
minutes =  [175,170,205,120,220,130,105,145,190]
labels = ['a','b','c','d','e','f','g','h','i']

plt.scatter(friends,minutes)

#레이블을 달자 !
for label, friend_count, minute_count in
zip(labels,friends,minutes):

plt.annotate(label,xy=(friend_count,minute_count),xytext=
(5,-5),textcoords='offset points')

plt.title("Daily Minutes vs. Numver of Friends")
plt.xlabel("# of friends")
plt.ylabel("daily minutes spent on the site")
plt.show()
```



plt.axis("equal")을 사용하면 x,y의 축 값을 같게 해 비교할수 있다

통계

- 데이터셋 설명: 최소/최대값 등등..
- 중심 경향성(central tendency) : 평균(sum(x)/n), 중앙값(50%), 분위(quantile, 일정 퍼센트 위치의 값), 최빈값(mode, 빈도가 가장 높은 값).
- 산포도(dispersion) : 全距（range, 최대값-최소값)
- 분산(variance)

```
def de_mean(x): # x의 모든 데이터에서 평균 뺌
    x_bar = mean(x)
    return [x_i - x_bar for x_i in x]

def variance(x):
    n = len(x)
    deviations = de_mean(x)
    return sum_of_squares(deviations)/(n-1)
```

표준편차(standard deviation)

```
def standard_deviation(x):
    return math.sqrt(variance(x))
```

상관관계

- 공분산(covariance): 두 변수가 각각의 평균에서 얼마나 멀리 떨어져 있는가.

```
def covariance(x,y):
    n=len(x)
    return dot(de_mean(x),de_mean(y))/(n-1)
```

Dot() : 각 성분별로 곱한 값을 더해 준다. 즉, x,y가 모두 평균 보다 크다면, 양수*양수 = 양수 일것이고, 그게 아니면 양수*음수 = 음수 일것. 즉 공분산이 양수거나 음수면 서로 상관이 있지만, 0이면 그와 같은 관계가 존재 x.

상관관계

- 상관관계(correlation): 공분산에서 각각의 표준편차를 나눠준 값

```
def correaltion(x,y):
    stdev_x = standard_deviation(x)
    stdev_y = standard_deviation(y)
    if stdev_x>0 and stdev_y>0:
        return covariance(x,y)/stdev_y/stdev_y
    else:
        return 0
```

%% 상관관계 사용시 주의사항 : 심슨의 역설(혼재 변수 누락), 상관관계 =0은 관계 없음이 아닌 선형적 관계 없음임,
상관관계 !=인과관계(x가 y 일으킬수도, 반대일수도, 제3인자일수도; 기사에서 많이 보임).