05

다양한 패키지로 데이터 분석하기

05-5 판다스로 통계 데이터 다루기
05-6 실전 통계 분석 맛보기
05-7 멧플롯립으로 그래프 그리기

05-5 판다스로 통계 데이터 다루기

기초통계량 살펴보기

- 가상의 설문 데이터를 분석한 survey.csv 파일 읽기

```
>>> import os, re
>>> import pandas as pd
>>> os.chdir(r'C:\Users\user\python')
>>> df2 = pd.read_csv('survey.csv')
```

05-5 판다스로 통계 데이터 다루기

기초통계량 살펴보기

- head()를 통하여 데이터 살펴보기

```
>>> df2.head()
  sex  income  English  jobSatisfaction  stress
0   m    3000      500                5       5
1   f    4000      600                4       4
2   f    5000      700                3       2
3   m    6000      800                2       2
4   m    4000      700                2       5
```

05-5 판다스로 통계 데이터 다루기

기초통계량 살펴보기

- 평균과 합 구하기
  - mean()을 통하여 평균값 계산

```
>>> df2.mean()
income              4304.217391
English              608.695652
jobSatisfaction        3.304348
stress                 3.347826
dtype: float64
```

05-5 판다스로 통계 데이터 다루기

기초통계량 살펴보기

- 평균과 합 구하기
  - 수입의 평균과 합 계산

```
>>> df2.income.sum()
98997
```

```
>>> df2.income.mean()
4304.217391304348
```

05-5 판다스로 통계 데이터 다루기

기초통계량 살펴보기

- 중앙값 구하기
  - 수입의 중앙값 계산

```
>>> df2.income.median()
4999.0
```

05-5 판다스로 통계 데이터 다루기

기초통계량 살펴보기

- 기초통계량 요약해서 출력
  - describe() 함수 사용

```
>>> df2.describe()
              income     English  jobSatisfaction     stress
count     23.000000   23.000000        23.000000  23.000000
mean    4304.217391  608.695652         3.304348   3.347826
std     1019.478341   99.603959         1.258960   1.433644
min     3000.000000  500.000000         1.000000   1.000000
25%     3000.000000  500.000000         2.500000   2.000000
50%     4999.000000  600.000000         3.000000   4.000000
75%     5000.000000  700.000000         4.000000   5.000000
max     6000.000000  800.000000         5.000000   5.000000
```

05-5 판다스로 통계 데이터 다루기

기초통계량 살펴보기

- 수입의 기초통계량 출력

```
>>> df2.income.describe()
count       23.000000
mean      4304.217391
std       1019.478341
min       3000.000000
25%       3000.000000
50%       4999.000000
75%       5000.000000
max       6000.000000
Name: income, dtype: float64
```

05-5 판다스로 통계 데이터 다루기

기초통계량 분석하기

- 빈도 분석하기
  - value_counts() 함수 사용

```
df.변수.value_counts()
```

05-5 판다스로 통계 데이터 다루기

기초통계량 분석하기

- df2에서 m(남성)과 f(여성)의 빈도 분석

```
>>> df2.sex.value_counts()
m    14
f     9
Name: sex, dtype: int64
```

05-5 판다스로 통계 데이터 다루기

기초통계량 분석하기

- 두 집단 평균 구하기
  - groupby() 함수 사용
  - 그룹을 나누어서 연산 진행

```
df.groupby(그룹을 나누는 변수).연산
```

05-5 판다스로 통계 데이터 다루기

기초통계량 분석하기

- 남성과 여성으로 나누어서 평균 구하기

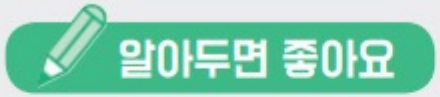
```
>>> df2.groupby(df2.sex).mean()

            income     English  jobSatisfaction    stress
sex
f     4333.111111  633.333333         3.666667  3.111111
m     4285.642857  592.857143         3.071429  3.500000
```

05-6 실전 통계 분석 맛보기

싸이파이 패키지로 t검정 하기

- 설문조사 결과를 둘로 나누고 평균을 구한 후 의미 있는 차이인지 확인
- 평균의 차이가 의미 있는지 확인하는 것은 t검정 간편함



t검정이란?

t검정이란 두 집단의 분산을 알 수 없을 때 두 집단이 t분포를 따른다고 가정하고 평균 등을 비교하는 통계 검정 방법을 의미합니다. 실제로 두 집단의 평균을 비교할 때 많이 사용하는 방법입니다.

05-6 실전 통계 분석 맛보기

싸이파이 패키지로 t검정 하기

- 싸이파이 모듈을 사용하여 쉽게 t검정 실행 가능
- 싸이파이에서 stats 모듈만 임포트

```
>>> from scipy import stats
```

05-6 실전 통계 분석 맛보기

싸이파이 패키지로 t검정 하기

- 수입 두 집단으로 나누기

```
>>> male = df2.income[df2.sex == 'm']
>>> female = df2.income[df2.sex == 'f']
```

05-6 실전 통계 분석 맛보기

싸이파이 패키지로 t검정 하기

- t검정 실행
  - pvalue가 0.05미만이거나 0.01미만이면 유의미한 차이 존재
  - 0.916… 이므로 큰 차이 없음

```
>>> stats.ttest_ind(male, female)
Ttest_indResult(statistic =-0.10650308143428423, pvalue=0.9161940781163369)
```

05-6 실전 통계 분석 맛보기

싸이파이 패키지로 t검정 하기

- ttest_result에 t검정 결과 저장

```
>>> ttest_result = stats.ttest_ind(male, female)


>>> print(ttest_result)
ttest_indResult(statistic=-0.10650308143428423, pvalue=0.9161940781163369)
```

05-6 실전 통계 분석 맛보기

싸이파이 패키지로 t검정 하기

- ttest_resul의 인덱스 0에는 ‘statistic’값이, 인덱스 1에는 ‘pvalue’가 들어감

```
>>> print(ttest_result[0])
-0.10650308143428423
>>> print(ttest_result[1])
0.9161940781163369
```

05-6 실전 통계 분석 맛보기

싸이파이 패키지로 t검정 하기

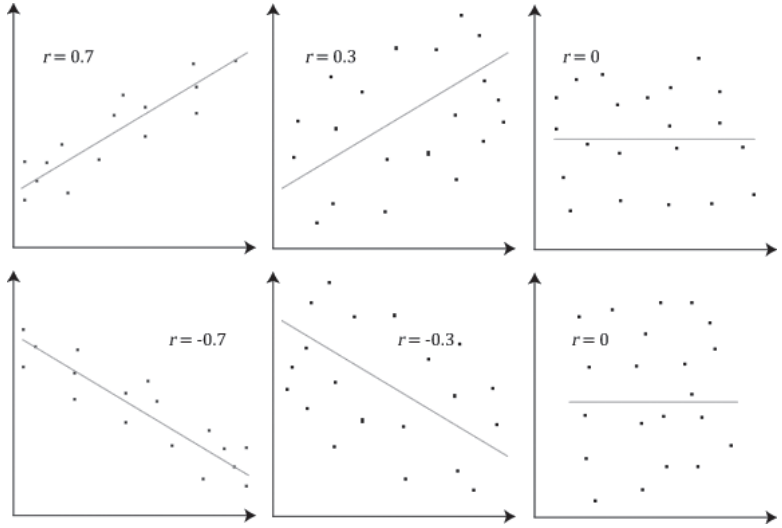
- 유의미한 값인지 조건문으로 쉽게 판단 가능

```
if ttest_result[1] > .05:
    print('p-value는 %f로 95% 수준에서 유의하지 않음' % ttest_result[1])
else:
    print('p-value는 %f로 95% 수준에서 유의함' % ttest_result[1])
```

05-6 실전 통계 분석 맛보기

피어슨과 스피어만 상관관계 분석 알아보기

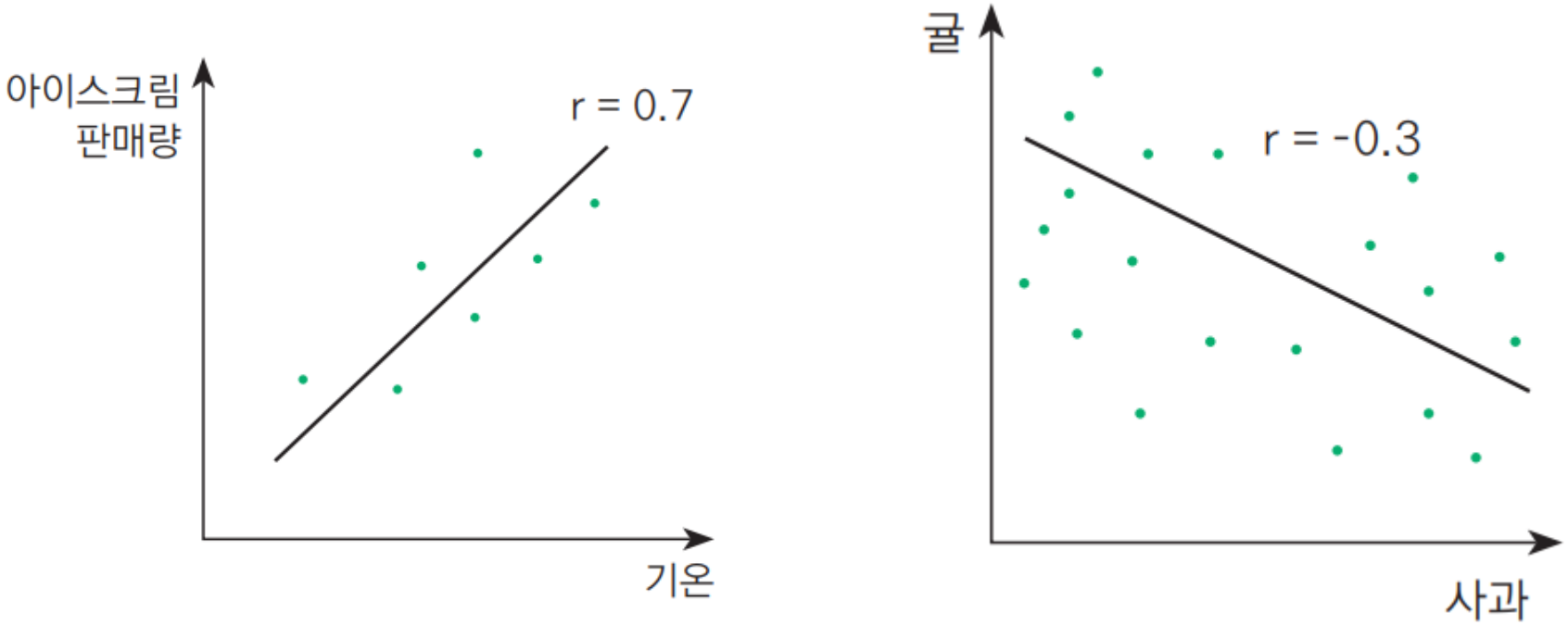
- 상관관계 분석은 두 변수가 얼마나 관련 있는지 알아보는 방법
- 방법은 크게 피어슨과 스피어만이 존재
- 해당 분석으로 '상관계수'(r)이 도출됨



05-6 실전 통계 분석 맛보기

피어슨과 스피어만 상관관계 분석 알아보기

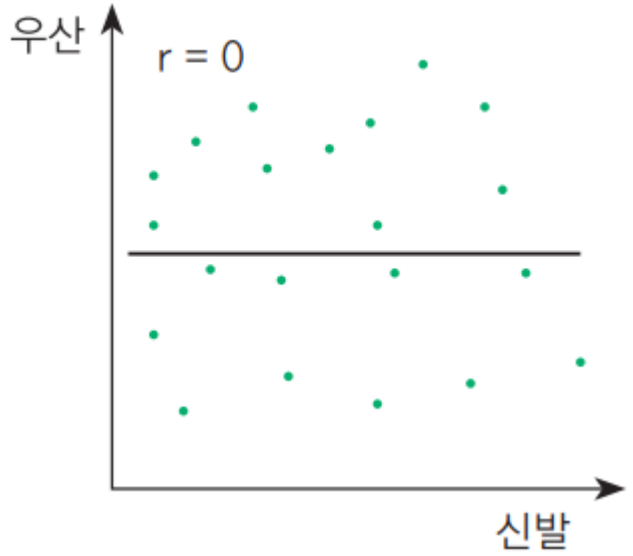
- 상관 계수가 0이 아닌 것의 예시



05-6 실전 통계 분석 맛보기

피어슨과 스피어만 상관관계 분석 알아보기

- 상관 계수가 0인 것 즉 관계가 없는 것의 예시



05-6 실전 통계 분석 맛보기

두 변수의 상관관계 분석하기

- 판다스의 corr() 함수를 통하여 상관관계 분석 가능
- 피어슨과 스피어만이 존재

```
df.corr()
df.corr(method = 'spearman')
```

05-6 실전 통계 분석 맛보기

두 변수의 상관관계 분석하기

- 피어슨 상관관계 분석 결과

```
>>> corr = df2.corr()
>>> corr
                    income   English      jobSatisfaction    stress
income            1.000000  0.599452       -0.040108 -0.137920
English           0.599452   1.000000         -0.312051  0.073351
jobSatisfaction   -0.040108  -0.312051           1.000000  0.165338
stress            -0.137920  0.073351         0.165338  1.000000
```

05-6 실전 통계 분석 맛보기

두 변수의 상관관계 분석하기

- 스피어만 상관관계 분석 결과

```
>>> df2.corr(method = 'spearman')

                income   English       jobSatisfaction    stress
income          1.000000  0.543705        -0.100683  -0.170584
English          0.543705  1.000000         -0.309747  0.068223
jobSatisfaction    -0.100683  -0.309747          1.000000  0.154982
stress           -0.170584  0.068223          0.154982  1.000000
```

05-6 실전 통계 분석 맛보기

두 변수의 상관관계 분석하기

- 수입과 스트레스의 상관관계만 보기

```
>>> df2.income.corr(df2.stress)
-0.13791959796123449
```

05-6 실전 통계 분석 맛보기

두 변수의 상관관계 분석하기

- 앞에서 구한 corr 객체를 CSV 파일로 저장
- 데이터프레임을 CSV 파일로 저장할 때는 to_csv() 사용

```
>>> corr.to_csv('corr.csv')
```

05-6 실전 통계 분석 맛보기

회귀 분석 알아보기

- 상관관계 분석에서 파악할 수 없는 변수 사이의 관계 파악 가능

$$y = a + bx + \varepsilon$$

05-6 실전 통계 분석 맛보기

이런 상황이라면?

- 영어 점수와 직업 만족도 사이에 인과관계가 있는지 분석하기

05-6 실전 통계 분석 맛보기

Statsmodels 패키지로 회귀 분석하기

- Statsmodels 패키지 임포트

```
>>> import statsmodels.formula.api as smf
```

05-6 실전 통계 분석 맛보기

Statsmodels 패키지로 회귀 분석하기

- ols() 함수를 통하여 회귀 분석 진행

```
ols(formula = '종속 변수 ~ 독립 변수', data = 데이터프레임)
```

05-6 실전 통계 분석 맛보기

Statsmodels 패키지로 회귀 분석하기

- ols() 함수를 통하여 회귀 분석 진행

```
>>> model = smf.ols(formula = 'jobSatisfaction~English', data = df2)


>>> result = model.fit()
```

05-6 실전 통계 분석 맛보기

Statsmodels 패키지로 회귀 분석하기

- 회귀 분석 결과 확인

```
>>> print(result.summary())
                            OLS Regression Results
==============================================================================
Dep. Variable:        jobSatisfaction   R-squared:                       0.097
Model:                            OLS   Adj. R-squared:                  0.054
Method:                 Least Squares   F-statistic:                     2.266
Date:                Wed, 04 Dec 2019   Prob (F-statistic):              0.147
Time:                        22:26:19   Log-Likelihood:                -36.243
No. Observations:                  23   AIC:                             76.49
Df Residuals:                      21   BIC:                             78.76
Df Model:                           1
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
Intercept      5.7052      1.615      3.532      0.002       2.346       9.065
English       -0.0039      0.003     -1.505      0.147      -0.009       0.002
==============================================================================
Omnibus:                        0.120   Durbin-Watson:                   0.777
Prob(Omnibus):                  0.942   Jarque-Bera (JB):                0.306
Skew:                          -0.126   Prob(JB):                        0.858
Kurtosis:                       2.495   Cond. No.                     3.90e+03
==============================================================================
```

05-6 실전 통계 분석 맛보기

Statsmodels 패키지로 회귀 분석하기

- 다중 회귀 분석 연습
- 회귀 분석보다 더 많은 변수 사용

```
smf.ols(formula = '종속 변수 ~ 독립 변수1 + ... + 독립 변수n', data = 데이터프레임)
```

05-6 실전 통계 분석 맛보기

Statsmodels 패키지로 회귀 분석하기

- 다중 회귀 분석 연습

```
>>> model2 = smf.ols(formula = 'jobSatisfaction~English + stress + income', data = df3)
>>> result = model2.fit()
```

05-7 맷플롯립으로 그래프 그리기

그래프 만들고 출력하기

- 맷플롯립 사용을 위해 임포트
- 가장 많이 쓰는 pyplot 모듈을 임포트

```
1 from matplotlib import pyplot as plt
2 import matplotlib.pyplot as plt
```

05-7 맷플롯립으로 그래프 그리기

그래프 만들고 출력하기

- 그래프를 그리기 위하여 데이터 정의

```
>>> import matplotlib.pyplot as plt
>>> x = [1, 4, 9, 16, 25, 36, 49, 64]
```

05-7 맷플롯립으로 그래프 그리기

그래프 만들고 출력하기

- plot() 함수를 사용해 그래프 생성

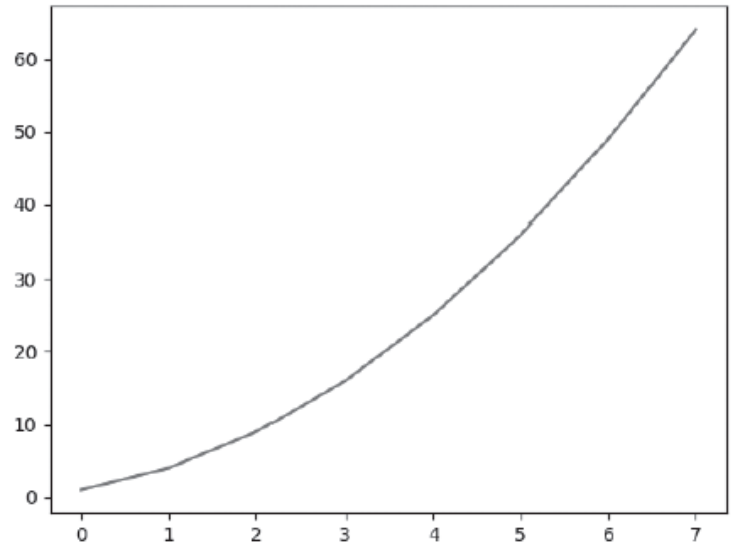
```
>>> plt.plot(x)
 [<matplotlib.lines.Line2D object at 0x000002213BA83D30>]
```

05-7 맷플롯립으로 그래프 그리기

그래프 만들고 출력하기

- show() 함수를 사용해 그래프 출력

```
>>> plt.show(x)
```



05-7 맷플롯립으로 그래프 그리기

그래프 모양과 색 지정하기

- plot 함수의 매개변수를 추가하여 모양과 색 수정 가능

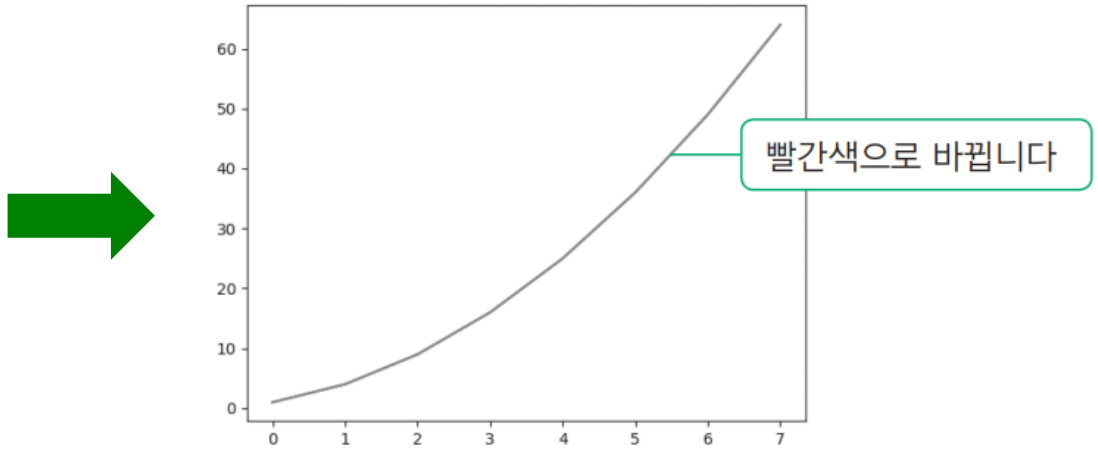
```
plt.plot(그래프 자료, 모양 + 색)
```

05-7 맷플롯립으로 그래프 그리기

그래프 모양과 색 지정하기

- plot 함수의 매개변수를 추가하여 모양과 색 수정 가능

```
>>> plt.plot(x, color = 'r')
[<matplotlib.lines.Line2D object at 0x0000017891899400>]
>>> plt.show(x)
```

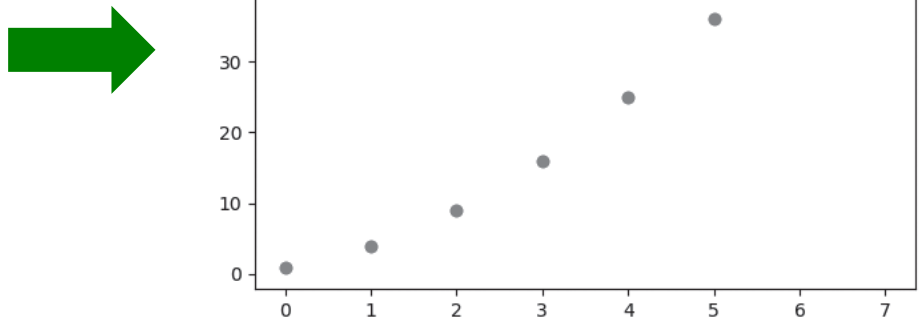


05-7 맷플롯립으로 그래프 그리기

그래프 모양과 색 지정하기

- r = red, o = 점

```
>>> plt.plot(x, 'or')
[<matplotlib.lines.Line2D object at 0x000001788F76A9E8>]
>>> plt.show(x)
```



05-7 맷플롯립으로 그래프 그리기

그래프 모양과 색 지정하기

- 맷플롯립 그래프 색 지정 문자

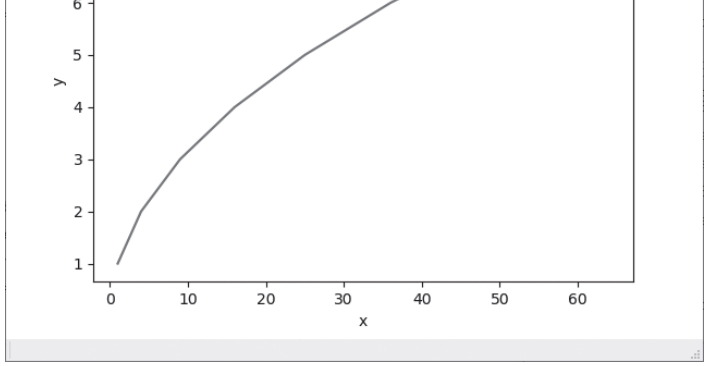
문자	색
'b'	파랑
'g'	초록
'r'	빨강
'c'	청록
'm'	자홍
'y'	노랑
'k'	검정
'w'	하양

05-7 맷플롯립으로 그래프 그리기

축 이름 지정하기

- 그래프의 축 이름 정하기

```
>>> x
[1, 4, 9, 16, 25, 36, 49, 64]
>>> y = [i for i in range(1, 9)]
>>> y
[1, 2, 3, 4, 5, 6, 7, 8]
>>> plt.plot(x, y)
>>> plt.xlabel('x')
>>> plt.ylabel('y')
>>> plt.title('matplotlib sample')
>>> plt.show()
```



05-7 맷플롯립으로 그래프 그리기

그래프를 이미지 파일로 저장하기

- 그래프를 이미지 파일로 저장하기