



학습 목표

- 파이썬 데이터 시각화 모듈인 Matplotlib에 대해서 알 수 있다.
- Matplotlib 차트의 종류에 대해서 알 수 있다.
- 데이터 시각화를 통해 데이터를 효율적으로 관리할 수 있다.

목차

1. 데이터 시각화 개요
2. Matplotlib
3. 데이터 시각화 정리

1. 데이터 시각화 개요

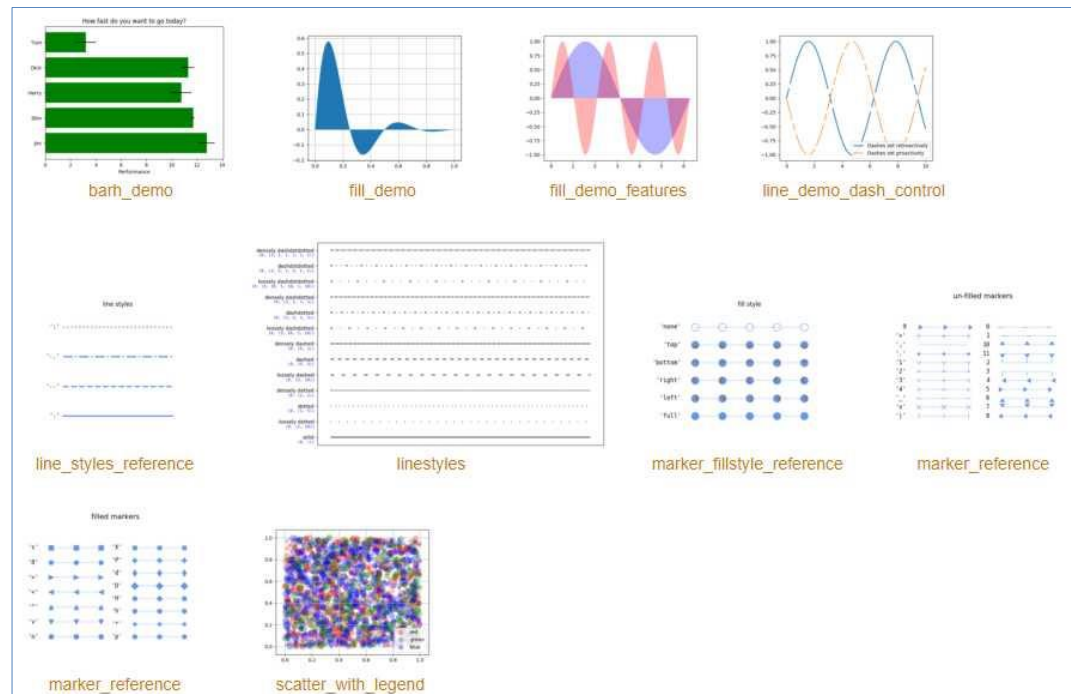
- 데이터 시각화 개요

- 데이터를 분석하고 파악하기 위해서 데이터를 일목요연하게 나열하거나 표의 형태로 만들어서 확인한다.
- 분석된 데이터를 시각적 요소를 활용하여 요약하여 표현한다면 데이터가 가지는 의미와 방향을 손쉽게 확인할 수 있다.
- 데이터 시각화는 다양한 형태의 차트를 이용하여 표현할 수 있다.
- 데이터를 표현하는 차트에서 도형의 모양이나 크기, 색상 등을 구분하여 표시하면 데이터와의 관계, 분포, 관련성 등을 시각적으로 파악하기가 훨씬 쉬워진다.
- 파이썬에서는 Matplotlib라는 데이터 시각화 모듈을 이용하여 차트를 만들 수 있다.
- 이번 장에서는 Matplotlib를 이용하여 데이터를 시각화 하는 방법을 알아본다.

2. Matplotlib

- matplotlib 소개

- 파이썬에서 데이터를 차트(chart)나 플롯(plot)으로 표현하기 위해 가장 많이 사용하는 패키지
- 간단한 설정으로 막대 차트, 라인 차트, 파이 차트, 스캐터 등을 표현
- 오픈소스

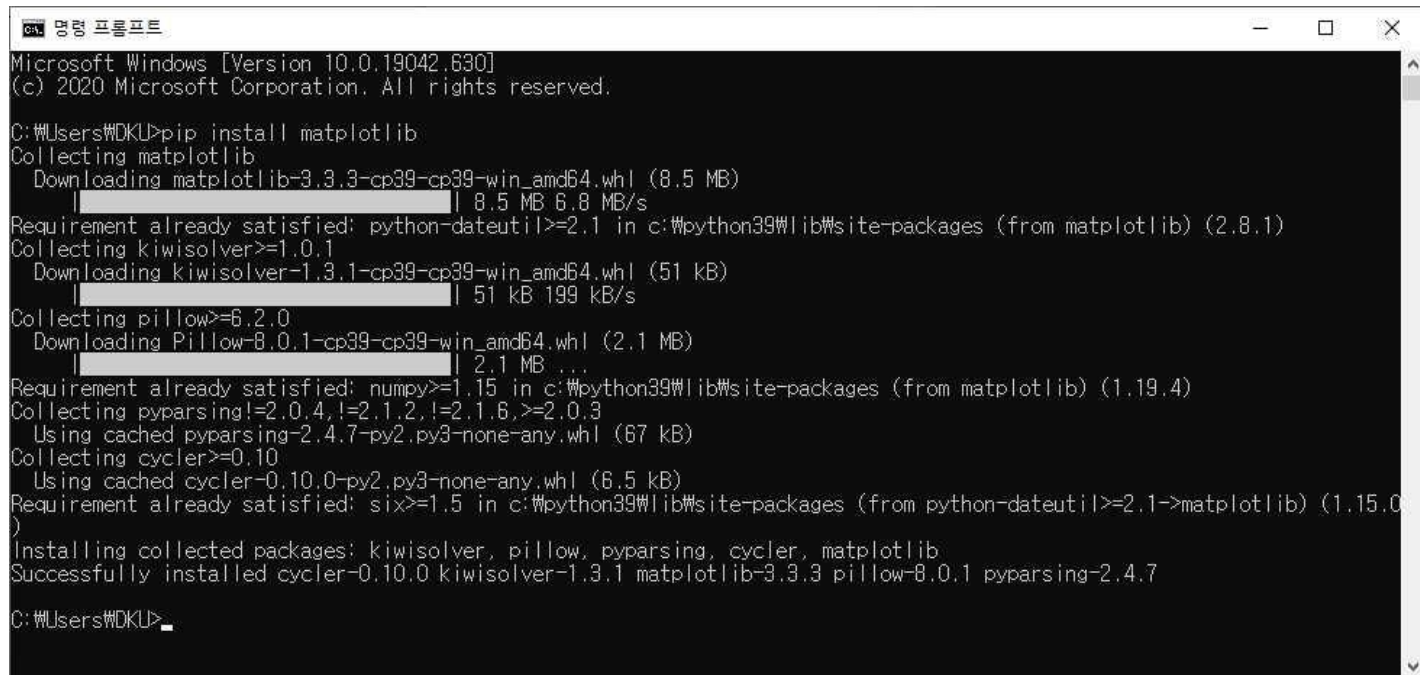


<https://matplotlib.org/>

2. Matplotlib

- matplotlib 설치

```
pip install matplotlib
```



```
명령 프롬프트
Microsoft Windows [Version 10.0.19042.630]
(c) 2020 Microsoft Corporation. All rights reserved.

C:\Users\WDKJ>pip install matplotlib
Collecting matplotlib
  Downloading matplotlib-3.3.3-cp39-cp39-win_amd64.whl (8.5 MB)
    |#####| 8.5 MB 6.8 MB/s
Requirement already satisfied: python-dateutil>=2.1 in c:\python39\lib\site-packages (from matplotlib) (2.8.1)
Collecting kiwisolver>=1.0.1
  Downloading kiwisolver-1.3.1-cp39-cp39-win_amd64.whl (51 kB)
    |#####| 51 kB 199 kB/s
Collecting pillow>=6.2.0
  Downloading Pillow-8.0.1-cp39-cp39-win_amd64.whl (2.1 MB)
    |#####| 2.1 MB ...
Requirement already satisfied: numpy>=1.15 in c:\python39\lib\site-packages (from matplotlib) (1.19.4)
Collecting pyparsing!=2.0.4,!=2.1.2,!=2.1.6,>=2.0.3
  Using cached pyparsing-2.4.7-py2.py3-none-any.whl (67 kB)
Collecting cycler>=0.10
  Using cached cycler-0.10.0-py2.py3-none-any.whl (6.5 kB)
Requirement already satisfied: six>=1.5 in c:\python39\lib\site-packages (from python-dateutil>=2.1->matplotlib) (1.15.0)
Installing collected packages: kiwisolver, pillow, pyparsing, cycler, matplotlib
Successfully installed cycler-0.10.0 kiwisolver-1.3.1 matplotlib-3.3.3 pillow-8.0.1 pyparsing-2.4.7

C:\Users\WDKJ>
```

matplotlib에서 필요한 의존성 패키지는 pip로 설치 시 자동으로 같이 설치 해준다.

2. Matplotlib

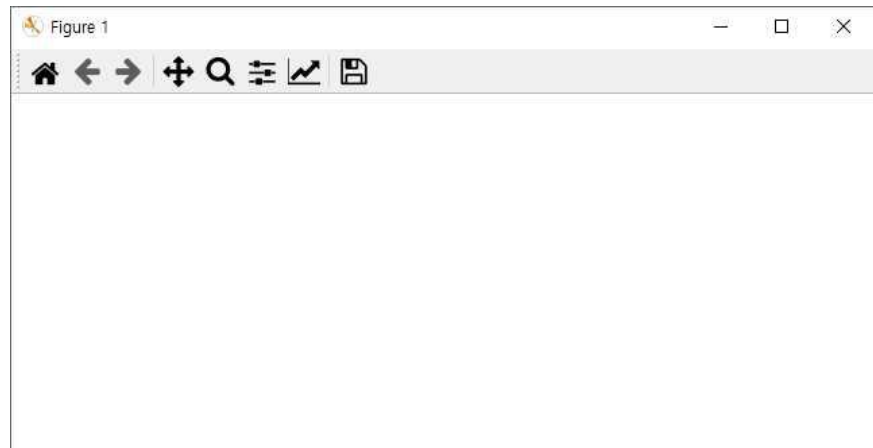
- matplotlib 시작
 - figure : 그래프로 표현할 윈도우 틀
 - show() : 화면에 나타냄

[코드 6-1] matplotlib 시작하기

```
import matplotlib.pyplot as plt
```

```
plt.figure()
```

```
plt.show()
```



2. Matplotlib

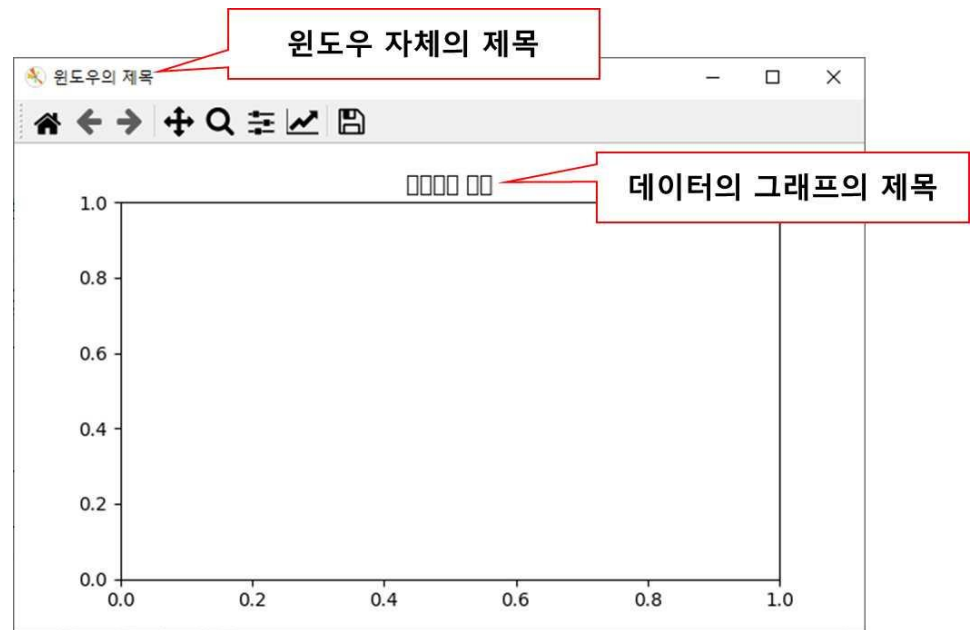
- matplotlib 제목 넣기
 - plt.figure(num="윈도우의 제목")
 - plt.title("데이터의 제목")

[코드 6-2] 그래프 제목 표현하기

```
import matplotlib.pyplot as plt

plt.figure(num="윈도우의 제목")
plt.title("데이터의 제목")

plt.show()
```



2. Matplotlib

- matplotlib 글꼴 변경

- `plt.rcParams.update({'font.family': 'malgun gothic', 'font.size': 12})`

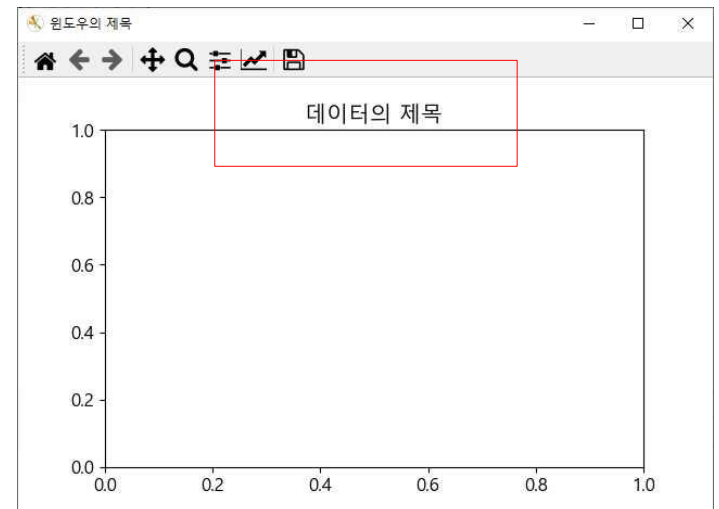
[코드 6-3] 글꼴 변경

```
import matplotlib.pyplot as plt

# 차트 윈도우 제목
plt.figure(num="윈도우의 제목")

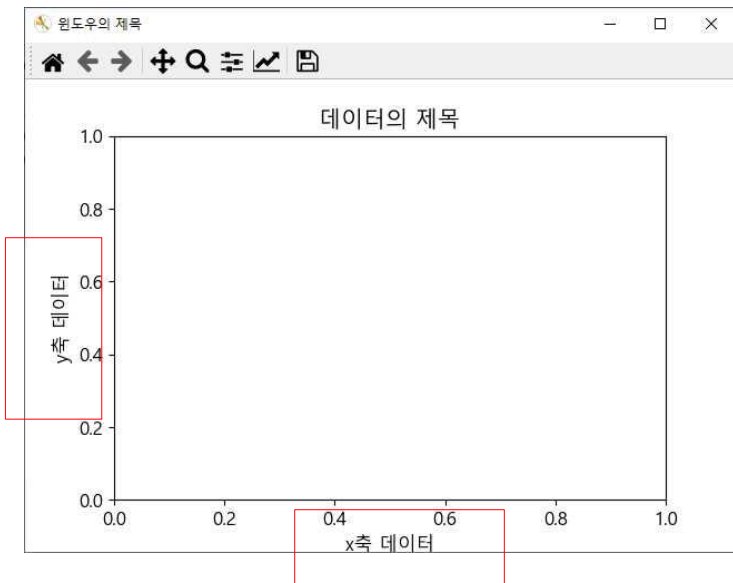
# 차트 글꼴
plt.rcParams.update({'font.family': 'malgun gothic', 'font.size': 12})

# 차트 제목
plt.title("데이터의 제목")
plt.show()
```



2. Matplotlib

- x축, y축 제목 넣기
 - xlabel(), ylabel()



[코드 6-4] x축 y축 이름 설정

```
import matplotlib.pyplot as plt

# 차트 윈도우 제목
plt.figure(num="윈도우의 제목")

# 차트 글꼴
plt.rcParams.update({'font.family': 'malgun gothic', 'font.size': 12})

# 차트 제목
plt.title("데이터의 제목")

# x, y 축 제목
plt.xlabel("x축 데이터")
plt.ylabel("y축 데이터")

plt.show()
```

2. Matplotlib

- 라인 차트(line chart)
 - 각 좌표의 점 표현

[코드 6-5] 라인 차트 실행

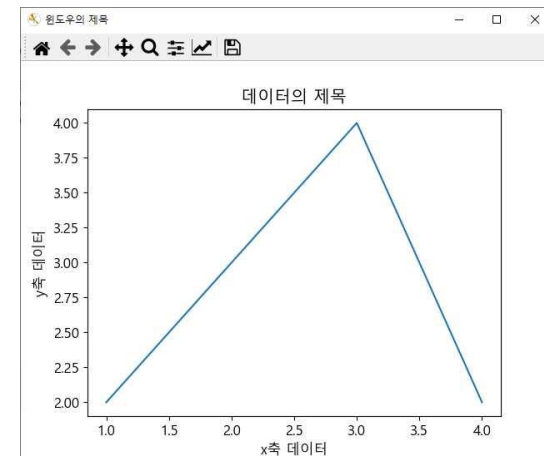
```
import matplotlib.pyplot as plt

# 차트 윈도우 제목
plt.figure(num="윈도우의 제목")

# 차트 제목, 글꼴 코드 생략
# x, y 축 제목
plt.xlabel("x축 데이터")
plt.ylabel("y축 데이터")

plt.plot([1, 3, 4], [2, 4, 2], marker='o')

plt.show()
```



보다 많은 점 표현은 아래에서

https://matplotlib.org/stable/api/markers_api.html

2. Matplotlib

- pandas의 DataFrame을 차트로 표현하기
 - DataFrame 클래스도 plot() 메서드를 가지고 있다.
 - 따라서 DataFrame.plot() 메서드를 이용하면 plt.plot()을 이용한 것과 마찬가지로 DataFrame의 데이터를 plot()의 데이터처럼 사용할 수 있다.
 - DataFrame의 plot() 메서드의 기본 형식은 아래와 같다.

```
DataFrame.plot(kind='line', x=column, y=columns, color=color, ax=None)
```

- 인수 중 x, y값은 옵션이며, x값은 x축으로 선택할 열 이름이고, y값은 y축으로 설정할 열 이름들이다. 값을 설정하지 않을 경우 DataFrame의 모든 데이터에서 인덱스를 x축으로 하고, 나머지를 y축으로 인식한다. 숫자가 아닌 경우는 그래프로 나타나지 않는다.
- x축으로 설정한 열 이름은 자동으로 x축 제목으로 나타난다. 만약 x축에 설정할 열 이름이 DataFrame에 없으면 KeyError가 난다.
- kind는 차트의 종류를 나타낸다. 라인 차트일 경우 kind="line"으로 인자 값을 설정하면 된다.

2. Matplotlib

[코드 6-6] DataFrame 라인 차트 실행

```
import pandas as pd
import matplotlib.pyplot as plt

data = pd.read_csv('sample.csv')

# 차트 윈도우 제목
plt.figure(num="성적 그래프")

# 차트 글꼴
plt.rcParams.update({'font.family': 'malgun gothic', 'font.size': 12})

# 차트 제목
plt.title("학번별 성적")

# x, y 축 제목
plt.xlabel("학번")
plt.ylabel("성적")

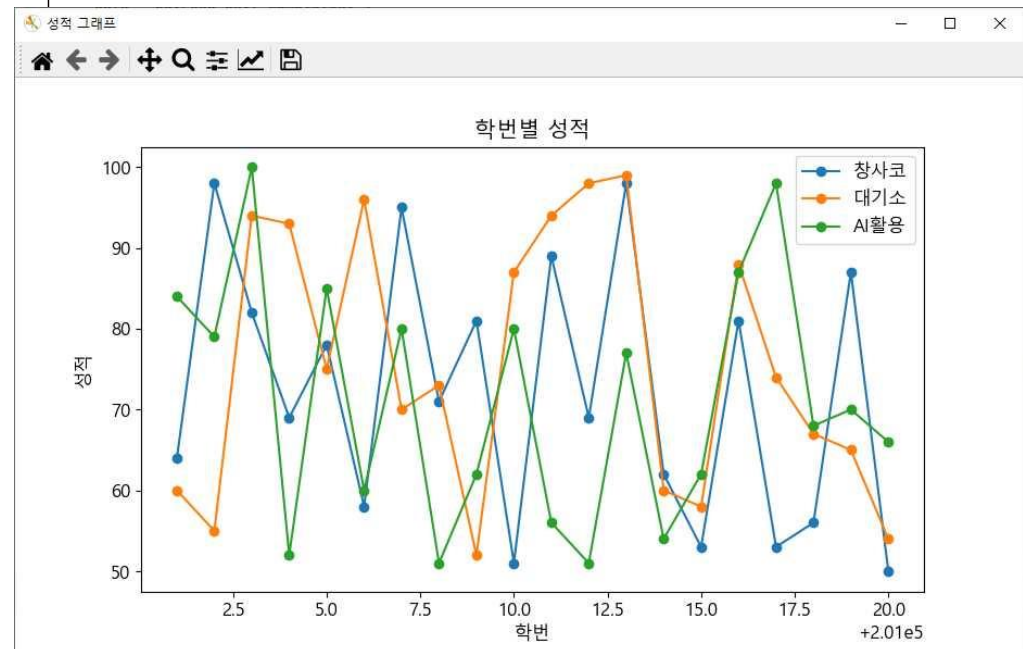
# plt 적용, 한개의 화면에서 나타나도록 하기 위한 axes 설정
ax = plt.gca()

columns = ['프로그래밍기초', '빅데이터분석', 'AI활용']
data.plot(kind='line', x='학번', y=columns, ax=ax, marker='o')

plt.show()
```

라인차트(line chart)

사용한 sample 데이터



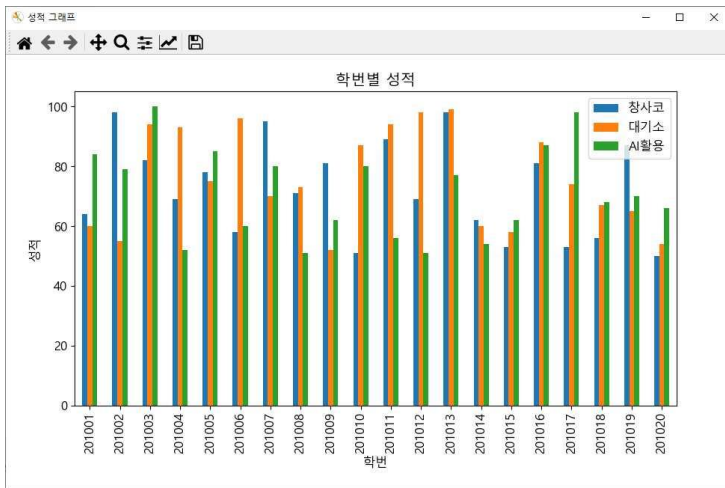
2. Matplotlib

- 막대 차트(bar chart)

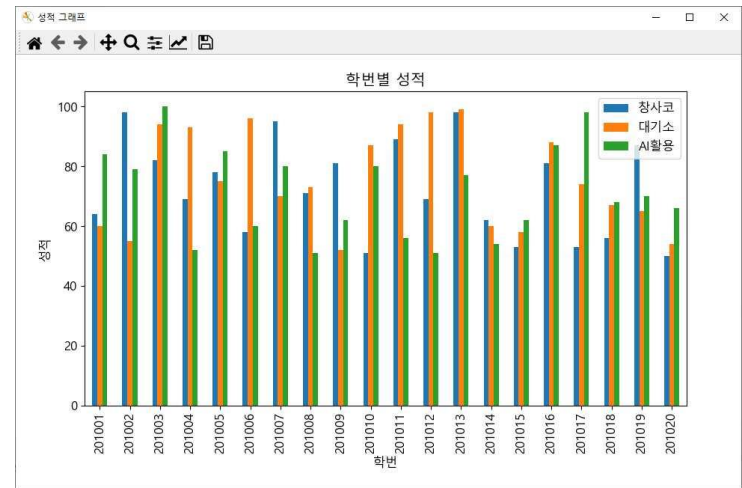
- 라인차트와 유사하고, kind='bar' 로 변경

- stacked는 하나의 막대에 누적 여부 #kind : 그래프 종류 'line', 'bar', 'barh', 'kde'

```
DataFrame.plot(kind='bar', x=column, y=columns, color=color, ax=None, stacked=False)
```



stacked=False



stacked=True

2. Matplotlib



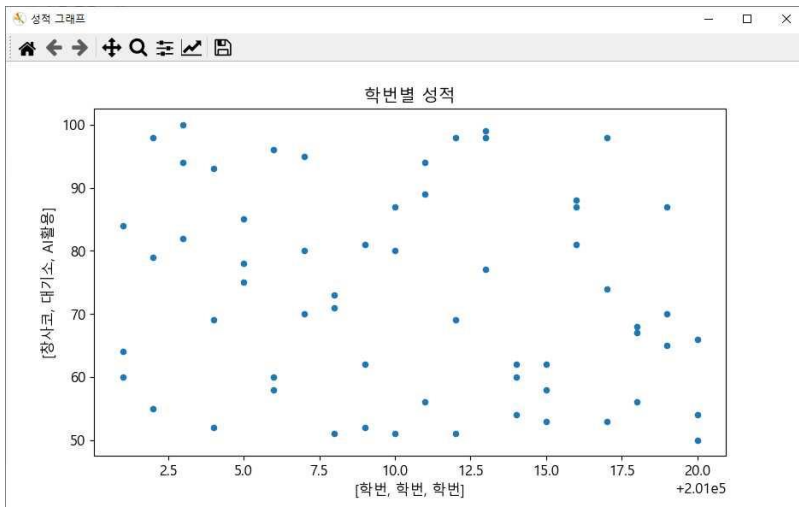
- 스캐터 차트(scatter chart)

- x축, y축 좌표의 값을 점으로 표현
- 기본 형식에서 x, y값은 필수이며, x축의 size와 y축의 size가 같아야 한다.
- 따라서 x축 1개와 y축 1개를 하나씩 매치시켜서 그래프로 표현하는 방법을 사용한다.
- x와 y의 값이 같아도 Error는 발생하지 않으나, 그래프의 모양이 직선으로 나타나 분석의 의미가 없어진다.
- s인자는 점의 크기를 설정한다.

```
DataFrame.plot(kind='scatter', x=columns, y=columns, color=color, ax=None, s=area)
```

2. Matplotlib

- 스캐터 차트(scatter chart)



[코드 6-8] DataFrame 스캐터 차트 실행

```
import pandas as pd
import matplotlib.pyplot as plt

data = pd.read_csv('sample.csv')

# 차트 윈도우 제목
plt.figure(num="성적 그래프")

# 차트 글꼴
plt.rcParams.update({'font.family': 'malgun gothic', 'font.size': 12})

# 차트 제목
plt.title("학번별 성적")

# x, y 축 제목
plt.xlabel("학번")
plt.ylabel("성적")

# plt 적용, 한개의 화면에서 나타나도록 하기 위한 axes 설정
ax = plt.gca()

xcolumns = ['학번', '학번', '학번']
ycolumns = ['장사코', '대기소', '시활용']

data.plot(kind='scatter', x=xcolumns, y=ycolumns, ax=ax)

plt.show()
```


2. Matplotlib

- 다수의 스캐터 차트(scatter chart)

- 하나의 plot() 메서드에 모든 열들을 넣었을 때는 점의 색상이 모두 같은 색으로 나타난다.
- 로 다른 색상으로 구분해주기 위해서는 data.plot()을 따로따로 만들어서 색상(color) 값을 다르게 설정하고,
- ax를 일치시켜서 하나의 그래프로 나타나게 한다.
- data.plot()을 여러 개 만들 때는 x축, y축 값을 하나씩 입력한 코드를 반복하면 된다.

```
color = ['#209FDF', '#99CA53', '#F6A625', '#6D5FD5', '#BF593E', '#FF0000', '#0000FF']
```

```
data.plot(kind='scatter', x='학번', y='프로그래밍기초', ax=ax, color=color[0], s=10)
```

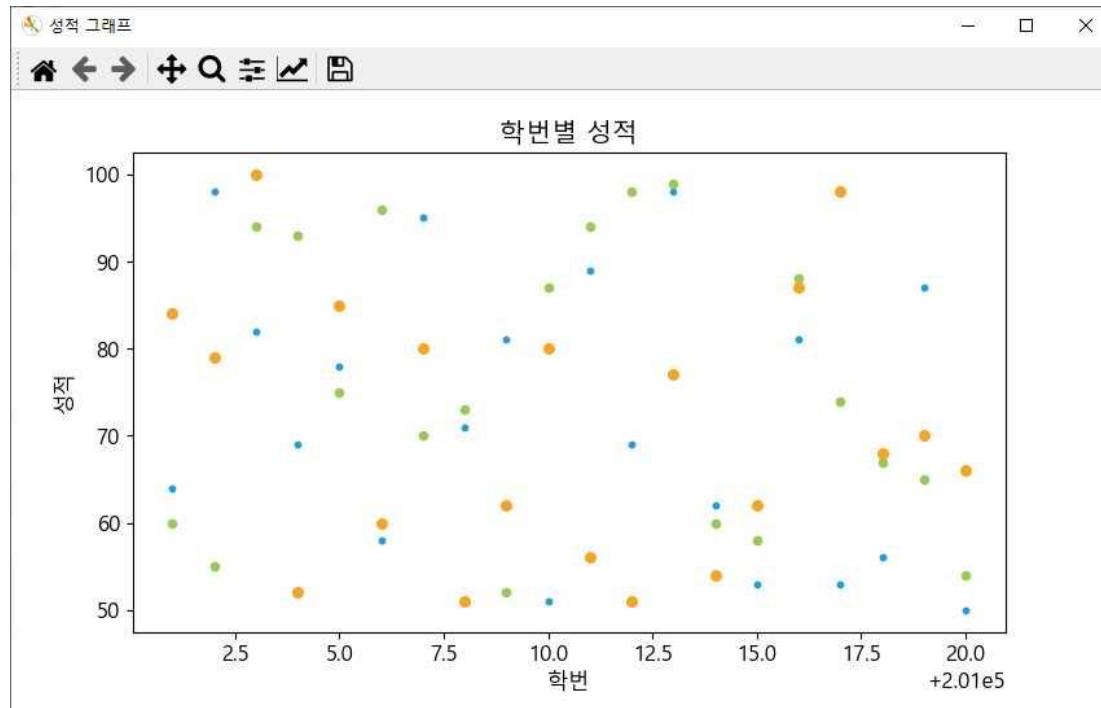
```
data.plot(kind='scatter', x='학번', y='빅데이터분석', ax=ax, color=color[1], s=20)
```

```
data.plot(kind='scatter', x='학번', y='AI활용', ax=ax, color=color[2], s=30)
```

2. Matplotlib

- 다수의 스캐터 차트(scatter chart)

—



2. Matplotlib

- 파이 차트(pie chart)

- 원 모양의 그래프에서 주어진 값들의 비율을 나타내는 차트

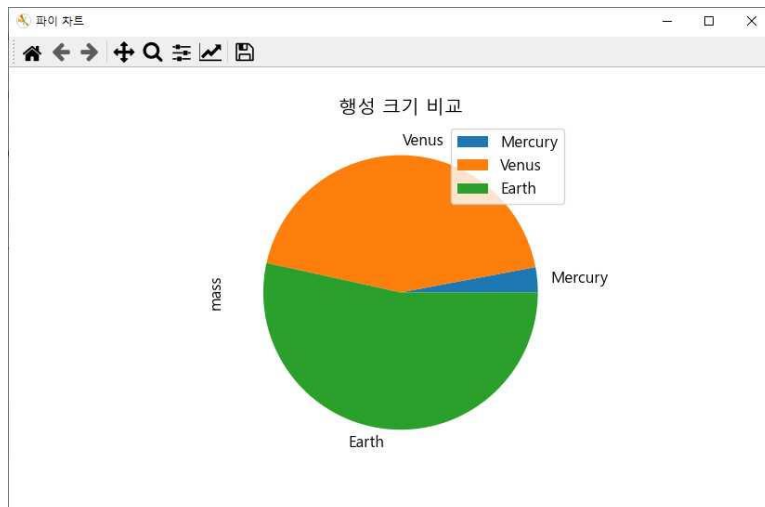
```
DataFrame.plot(kind='pie', y=column, ax=None)
```

파이 차트를 표현하기 위해 간단하게 DataFrame을 만들어서 적용

```
data = pd.DataFrame({'mass': [0.330, 4.87, 5.97], 'radius': [2439.7, 6051.8, 6378.1]}, index=['Mercury', 'Venus', 'Earth'])
```

2. Matplotlib

- 파이 차트(pie chart)



[코드 6-10] 파이 차트 실행

```
import pandas as pd
import matplotlib.pyplot as plt

data = pd.DataFrame({'mass': [0.330, 4.87, 5.97], 'radius': [2439.7, 6051.8, 6378.1]}, index=['Mercury', 'Venus', 'Earth'])

# 차트 윈도우 제목
plt.figure(num="파이 차트")

# 차트 글꼴
plt.rcParams.update({'font.family': 'malgun gothic', 'font.size': 12})

# 차트 제목
plt.title("행성 크기 비교")

# plt 적용, 한개의 화면에서 나타나도록 하기 위한 axes 설정
ax = plt.gca()

data.plot(kind='pie', y='mass', ax=ax)

plt.show()
```

2. Matplotlib

- 히스토 그램(histogram)

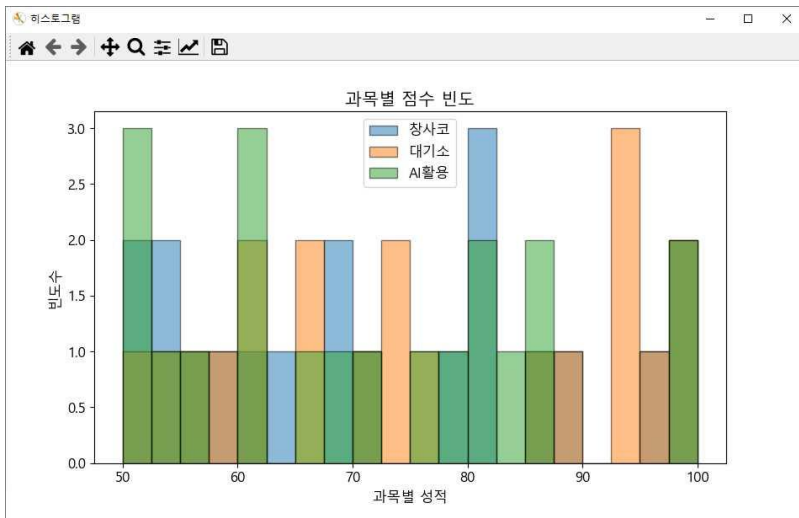
- 데이터 분석에서 변수의 분포, 중심 경향, 퍼짐 정도, 치우침 정도 등을 파악하기 위해 사용.

```
DataFrame.plot(kind='hist', by=None, bins=10, ax=None, alpha=1, edgecolor='black', linewidth=1.0)
```

- 도수 분포의 상태를 막대 모양으로 표현한 그래프
- 인수중 bins는 x축을 의미하며 히스토그램으로 나타낼 막대의 수를 의미
- bins 기본값은 10이며, 분포도를 좀 더 세분화하여 보고 싶다면 bins값을 크게 하면 된다.
- alpha는 투명도를 설정한다. 여러 개의 열들이 동시에 히스토그램으로 나타날 때 중복되면 먼저 생성된 히스토그램이 나중에 생성된 히스토그램에 의해 가려져서 보이지 않으므로, alpha값을 0.5로 주어 반 투명하게 만들어서 확인할 수 있다..

2. Matplotlib

- 히스토그램(histogram)



[코드 6-11] 히스토그램

```
import pandas as pd
import matplotlib.pyplot as plt

data = pd.read_csv('sample.csv')
# 학번열 삭제
data = data.drop('학번', axis=1)

# 차트 윈도우 제목
plt.figure(num="히스토그램")

# 차트 글꼴
plt.rcParams.update({'font.family': 'malgun gothic', 'font.size': 12})

# 차트 제목
plt.title("과목별 점수 빈도")

# plt 적용, 한개의 화면에서 나타나도록 하기 위한 axes 설정
ax = plt.gca()

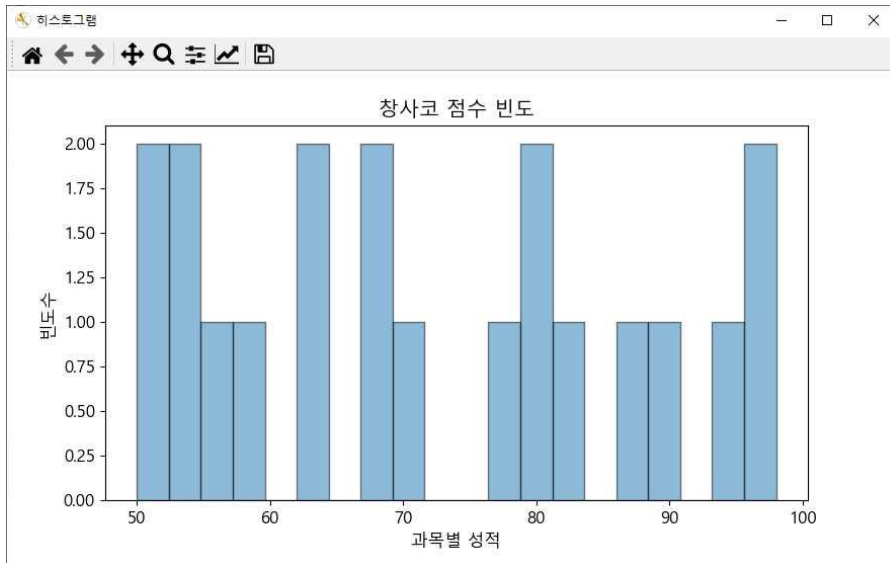
data.plot(kind='hist', by=None, bins=20, ax=ax, alpha=0.5, edgecolor='black', linewidth=1)

# x, y 축 제목
plt.xlabel("과목별 성적")
plt.ylabel("빈도수")

plt.show()
```

2. Matplotlib

- 단일 히스토 그램(histogram)



[코드 6-12] 단일 데이터 히스토그램

```
import pandas as pd
import matplotlib.pyplot as plt

data = pd.read_csv('sample.csv')
# 학번열 삭제
data = data.drop('학번', axis=1)

# 프로그래밍기초 데이터를 data1에 series로 저장
data1 = data['프로그래밍기초']

# 차트 윈도우 제목
plt.figure(num="히스토그램")

# 차트 글꼴
plt.rcParams.update({'font.family': 'malgun gothic', 'font.size': 12})

# 차트 제목
plt.title("프로그래밍기초 점수 빈도")

# plt 적용, 한개의 화면에서 나타나도록 하기 위한 axes 설정
ax = plt.gca()

data1.plot(kind='hist', by=None, bins=20, ax=ax, alpha=0.5, edgecolor='black', linewidth=1)

# x, y 축 제목
plt.xlabel("과목별 성적")
plt.ylabel("빈도수")

plt.show()
```