Visualization of data using Matplotlib

프로젝트 : 영화 상영 건수에 대한 시각화

윤수경, 강민서, 한지원, 길태호, 전주혁

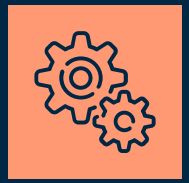
목차

01

데이터 시각화

02

코드설명



데이터 시각화

plot() 함수

ex) myframe. plot(kind=' bar', rot=0, title='지역별 차량 등록 대수', legend=True)

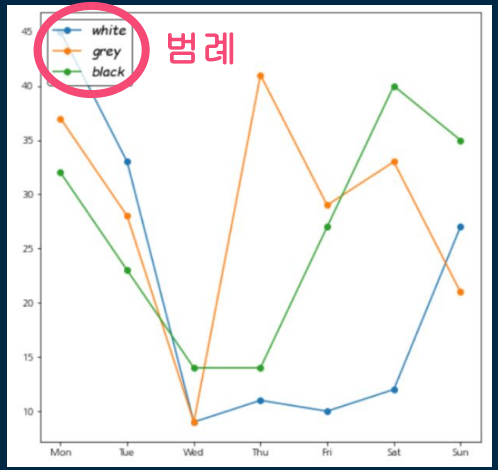
kind •그래프의 종류를 설정
•'line', 'bar*, 'barh'(수평 막대 그래프), 'pie', 'kde'(커널 밀도 추정 그래프) 등을 그릴 수 있음

grid •축의 그리드(눈금)를 표시할 것인가의 여부를 설정, 기본값 True

rot •축에 보이는 눈금 이름을 지정한 각도만큼 회전시킴(0~360)

title •그래프에 들어가 창의 제목을 문자열로 설정

legend •범례를 추가하는 옵션, 기본값 True



plot() 함수

ex) myframe. plot(kind=' bar', rot=0, title='지역별 차량 등록 대수', legend=True)

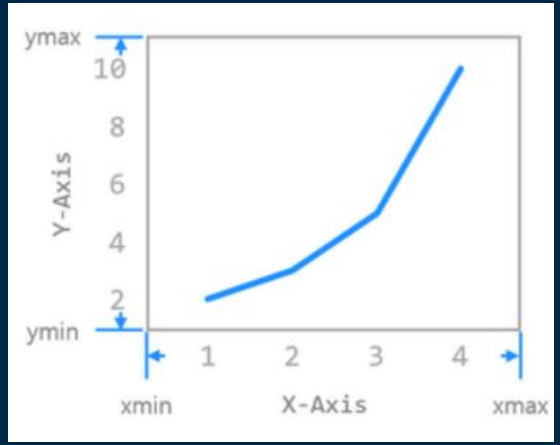
se_index • 객체의 색인을 눈금으로 사용할 것인지의 여부를 설정, False면 색인을 x축의 눈금으로 사용하지 않음

ylim •y축의 상한값과 하한값을 설정

xlim •x축의 상한값과 하한값을 설정

axis •X, Y축이 표시되는 범위를 지정하거나 반환

입력값이 없으면 데이터에 맞게 자동으로 범위를 지정

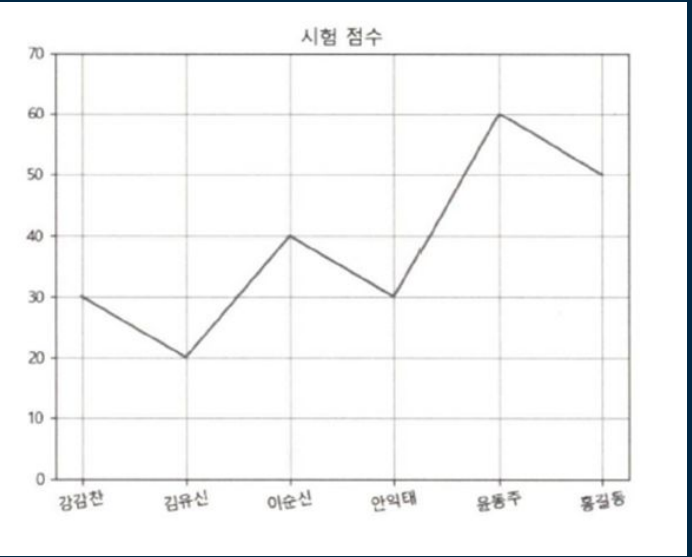


1)꺾은선 그래프

-꺾은선 그래프는 주로 수량의 변화를 시간의 흐름에 따라 보여줄 때 많이 사용됨

-시간에 따른 수량의 변화를 연속적으로 알아보기 쉽다는 장점이 있고, 확인되지 않는 중간의 값들도 개략적으로 예상할 수 있음

-plot()의 kind을 'line'으로 하면 꺾은선 그래프를 그릴 수 있음

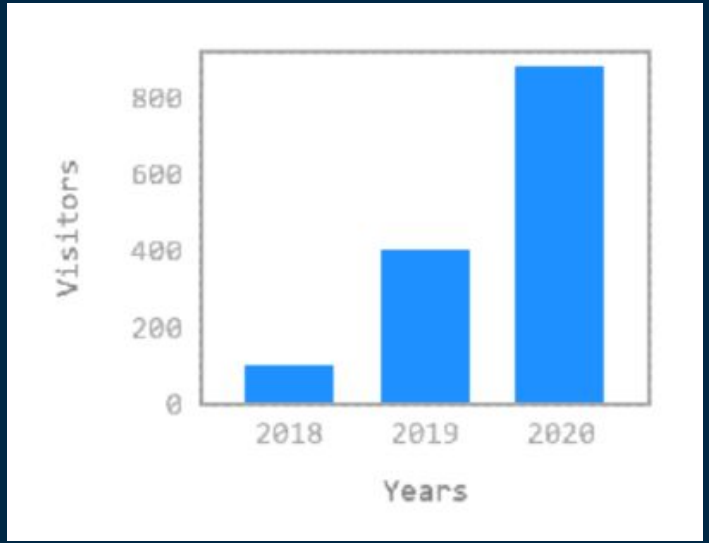


2)막대 그래프

-막대그래프는 여러 가지 통계 정보들의 양을 막대 모양으로 나타냄

-수량의 많고 적음을 한눈에 쉽게 알아보기에 편리

-plot()의 kind을 'bar'으로 하면 막대 그래프를 그릴 수 있음



3) 산점도 그래프

-산점도 그래프는 2개의 연속형 변수 간의 관계(영향력)를 보기 위하여 직교 좌표의 x축과 y축에 관측점을 찍어서 만든 그래프

-두 변수의 관계를 시각적으로 검토할 때 유용

-plot()의 kind을 'scatter'으로 하면 산점도 그래프를 그릴 수 있음

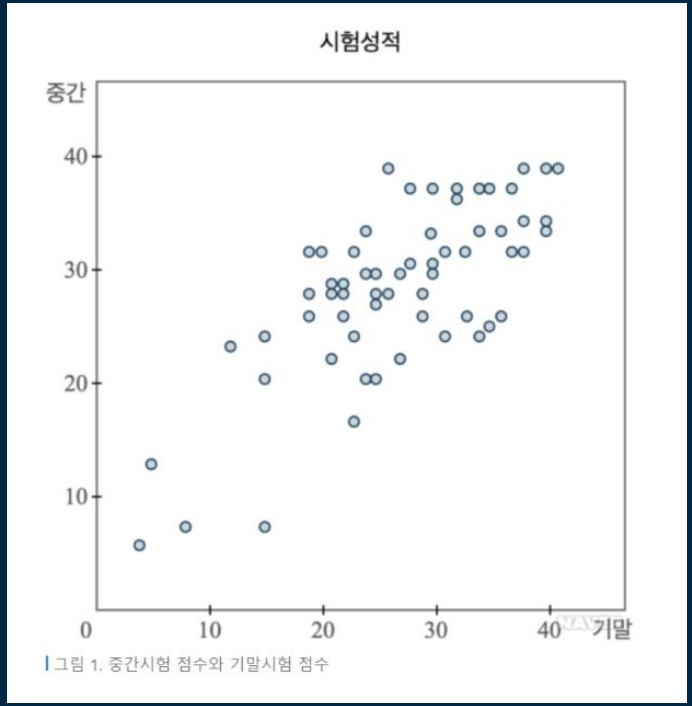


그림 1. 중간시험 점수와 기말시험 점수

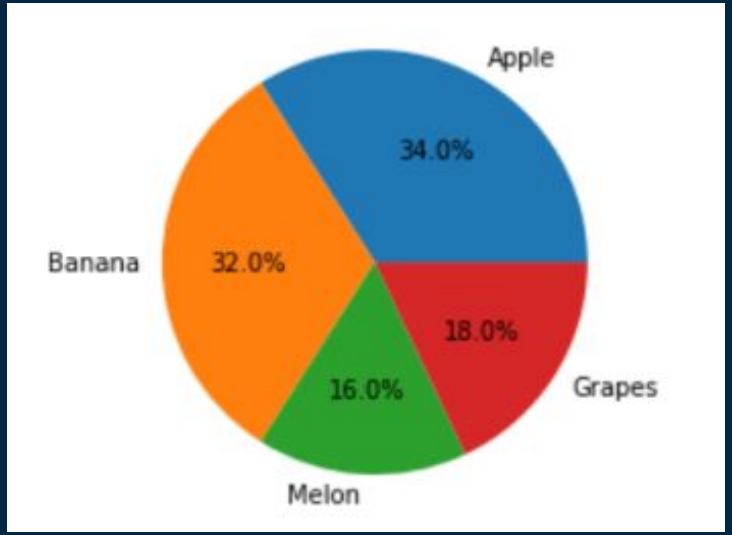
4) 파이 그래프

-파이 그래프는 범주별 구성 비율을 원형으로 표현한 그래프

-전체적인 비율을 쉽게 파악할 수 있음

-matplotlib.pyplot 모듈의 pie() 함수를 이용해서 만들 수 있음

-plot()의 kind을 'pie'으로 하면 파이 그래프를 그릴 수 있음



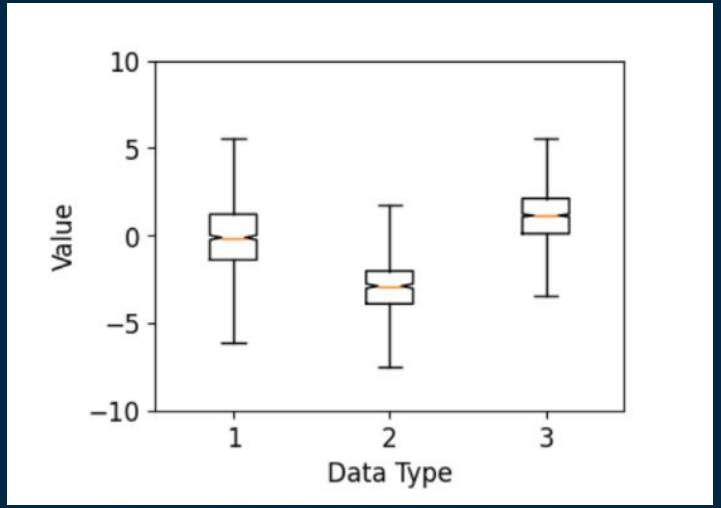
5) 상자 수염 그래프

-상자 수염 그래프 또는 상자 그래프는 데이터로부터 얻어낸 통계량인 5가지 요약 수치를 가지고 그려줌

-5가지 요약 수치는 최솟값, 제1~3사분위, 최댓값을 의미 (사분위 수는 데이터를 4등분한 지점을 의미)

-여러 개의 범주형 데이터에 대하여 하나의 공간에 수월하게 표현할 수 있는 것이 장점

-plot()의 kind을 'box'으로 하면 상자수염 그래프를 그릴 수 있음



코드 설명

프로그램 요구사항

-매장별 상영 건수에 대하여 합계, 평균, 개수를 구하여 각각 수평 막대 그래프를 그려 본다.

-매장별 상영 건수에 대하여 평균과 개수를 구하여 각각 수직 막대 그래프를 그려 본다.

-매장별 상영 건수에 대한 비율을 구하여 파이 그래프를 그려 본다.

코드 설명

```
import numpy as np #numpy 라이브러리 import
import pandas as pd #pandas 라이브러리 import
import matplotlib #matplotlib 라이브러리 import
import matplotlib.pyplot as plt# matplotlib.pyplot 라이브러리 import
from matplotlib import font_manager, rc #font_manager, rc 라이브러리 import
from math import sqrt  #sqrt 라이브러리 import
```

import 모듈 as 이름 : 모듈 이름 정하기
from 모듈 import 함수 : 모듈에서 원하는 함수만 가져오기

프로젝트에 사용된 라이브러리

넘파이 : 데이터 처리와 수치 계산에 사용되는 라이브러리

판다스 : 파이썬 데이터 처리를 위한 라이브러리. 파이썬을 이용한 데이터 분석과 같은 작업에서 필수 라이브러리라고 알려져있다.

맷플롯립 : 데이터를 차트(chart)나 플롯(plot)으로 시각화하는 라이브러리

코드 설명

```
#한글 깨짐 방지를 위해 폰트 바꾸기
Font_location = 'c:/windows/fonts/malgun.ttf'  #NanumGothicLight 폰트의 위치 저장
font_name = font_manager.FontProperties(fname=font_location).get_name()
#폰트 이름 얻어오기
matplotlib.rc('font',family=font_name)   #rc를 통해 폰트를 설정
#plt.rc('font', family='NanumBarunGothic')
```

글꼴 위치 찾기

```
for font in font manager.fontManager.ttflist:
    if 'malgun' in font.name:

        print(font.name, font.fname)
```

코드 설명

```
#csv 파일을 읽어 데이터프레임 만들기

theaterfile = 'theater.csv'  #영화 및 극장 상영정보를 가지고 있는 csv
colnames = ['id','theater', 'region','bindo']  #해당 데이터들의 칼럼 정보를
지정,csv 파일에는 칼럼들에 대한 header 정보가 없음
dftheater = pd.read_csv(theaterfile, names=colnames, header =  None)
#csv 파일을 읽어 데이터프레임 만들기
dftheater = dftheater.rename(index=dftheater.id)
#데이터프레임 id 칼럼의 이름 변경
```

코드 설명

```
dftheater = dftheater.reindex(columns=[ 'theater', 'region','bindo'])
#데이터프레임의 색인을 다시 지정하기
dftheater.index.name =  'id'
#데이터 프레임의 인덱스 이름을 'id'로 지정
print('전제 조회')
print(dftheater)   #데이터 프레임 전체 출력
```

코드 설명

theater.csv 파일

	A	B	C	D	E
1	brother	daehan	강남	15	
2	brother	megabox	강남	15	
3	brother	cgv	강남	30	
4	brother	daehan	신촌	20	
5	brother	megabox	신촌	30	
6	brother	cgv	신촌	15	
7	hulk	daehan	강남	25	
8	hulk	megabox	강남	15	
9	hulk	cgv	강남	15	
10	hulk	daehan	신촌	25	
11	hulk	megabox	신촌	20	
12	hulk	cgv	신촌	15	
13	king	daehan	강남	30	
14	king	megabox	강남	20	
15	king	cgv	강남	25	
16	king	daehan	신촌	25	
17	king	megabox	신촌	25	
18	king	cgv	신촌	15	
19	lucky	daehan	강남	25	
20	lucky	megabox	강남	20	
21	lucky	cgv	강남	25	
22	lucky	daehan	신촌	30	
23	lucky	megabox	신촌	15	
24	lucky	cgv	신촌	20	

```
전체 조회
        theater region  bindo
id
brother   daehan     강남     15
brother  megabox     강남     15
brother      cgv     강남     30
brother   daehan     신촌     20
brother  megabox     신촌     30
brother      cgv     신촌     15
hulk      daehan     강남     25
hulk     megabox     강남     15
hulk         cgv     강남     15
hulk      daehan     신촌     25
hulk     megabox     신촌     20
hulk         cgv     신촌     15
king      daehan     강남     30
king     megabox     강남     20
king         cgv     강남     25
king      daehan     신촌     25
king     megabox     신촌     25
king         cgv     신촌     15
lucky     daehan     강남     25
lucky    megabox     강남     20
lucky        cgv     강남     25
lucky     daehan     신촌     30
lucky    megabox     신촌     15
lucky        cgv     신촌     20
```

코드 설명

```
#데이터프레임 dftheater의 극장별 상영 횟수를 집계하기(데이터 그룹화에 대해서는 교재 p368 참조)
print('극장별 상영 횟수 집계')
mygrouping = dftheater.groupby( 'theater')['bindo']
#극장별 상영 건수에 대한 데이터 그룹화
sumSeries = mygrouping.sum()
#그룹화 한 데이터에서 상영건수 합계를 sumSeries에 저장
meanSeries = mygrouping.mean()
#그룹화 한 데이터에서 상영건수 평균를 meanSeries 저장
```

groupby 객체에 사용 가능한 함수

sum() : 누락된 데이터를 배제하고, 총합을 구한다
mean() : 누락된 데이터를 배제하고, 평균값을 구한다.

코드 설명

id	theater	region	bindo
brother	daehan	강남	15
brother	megabox	강남	15
brother	cgv	강남	30
brother	daehan	신촌	20
brother	megabox	신촌	30
brother	cgv	신촌	15
hulk	daehan	강남	25
hulk	megabox	강남	15
hulk	cgv	강남	15
hulk	daehan	신촌	25
hulk	megabox	신촌	20
hulk	cgv	신촌	15
king	daehan	강남	30
king	megabox	강남	20
king	cgv	강남	25
king	daehan	신촌	25
king	megabox	신촌	25
king	cgv	신촌	15
lucky	daehan	강남	25
lucky	megabox	강남	20
lucky	cgv	강남	25
lucky	daehan	신촌	30
lucky	megabox	신촌	15
lucky	cgv	신촌	20

데이터그룹화

theater	bindo
daehan	15
daehan	20
daehan	25
daehan	25
daehan	30
daehan	25
daehan	25
daehan	30

theater	bindo
megabox	15
megabox	30
megabox	15
megabox	20
megabox	20
megabox	25
megabox	20
megabox	15

theater	bindo
cgv	30
cgv	15
cgv	15
cgv	15
cgv	25
cgv	15
cgv	25
cgv	20

코드 설명

```
sizeSeries = mygrouping.size()
#그룹화 한 데이터에서 극장별 개수를 sizeSeries에 저장

#dftheater 데이터프레임의 합계, 평균, 개수에 대한 새로운 데이터프레임 df 만들기
print('Series 3개를 이용하여 DataFrame을 만들어 낸다. ')
df = pd.concat([sumSeries, meanSeries, sizeSeries], axis= 1)
#그룹화한 데이터의 합계, 평균, 개수를 합쳐 새로운 데이터프레임 df 만들기
df.columns = ['합계','평균','개수']  #df의 칼럼을 설정
print(df)  #df 출력
```

axis : 축
{0/'색인', 1/'열'}, 기본값 0

코드 설명

```
극장별 상영 횟수 집계
Series 3개를 이용하여 DataFrame을 만들어 낸다.
            합계      평균  개수
theater
cgv         160    20.000   8
daehan      195    24.375   8
megabox     160    20.000   8
```

코드 설명 (수평막대 그래프 그리기)

```
#수평막대 그래프 그리기
df.plot(kind='barh', rot=0)#그래프의 유형은 세로막대, 회전 0도로 하여 그래프 그리기
# df.plot(kind='barh', rot=0, alpha=0.7, legend=True, stacked=True)
plt.title(str(mysize)+'개 매장 집계 데이터')  #그래프의 제목 설정
# plt.show()
filename='visualizationExam_01.png'   #파일 이름 설정
plt.savefig(filename) #파일 저장
```

수평 막대 그래프
:plot()의 kind을 'barh'으로 하면 수평 막대 그래프를
그릴 수 있음 (수직 막대 그래프는 'bar')

코드 설명 (수평막대 그래프 그리기)

코드 설명

```
#사전과 집계 메소드를 활용해 데이터 만들기
print('집계 메소드를 사전에 담아 전달하기')
print('지역의 개수와 상영 회수의 총합')
mydict = {'bindo': 'sum', 'region': 'size'}#빈도와 지역에 대한 합계와 개수를
구하기 위한 사전 정보,sum을 bindo로 size를 region으로 변경하여 mydict에 저장
result = dftheater.groupby ('theater').agg(mydict)
#dttheater 데이터프레임에서의 theater 정보를 행, mydict를 열로 한 그룹핑한
데이터프레임 result 만들기
print(result)#데이터프레임 result 출력
```

agg() = aggregate()
: 함수 이름을 전달하여 값을 추출할 수 있다

코드 설명

```
집계 메소드를 사전에 담아 전달하기
지역의 개수와 상영 회수의 총합
          bindo  region
theater
cgv         160       8
daehan      195       8
megabox     160       8
```

코드 설명

```
#집계 데이터를 넘파이로 이용해 출력하기
print('넘파이를 이용한 출력')
result = mygrouping.agg([np.count_nonzero, np.mean, np.std])
#집계 데이터의 0이 아닌 데이터의 수와 평균, 표준편차를 넘파이로 구현
print(result)#그룹핑 결과 출력
```

코드 설명

```
넘파이를 이용한 출력
         count_nonzero    mean       std
theater
cgv                  8  20.000  5.976143
daehan               8  24.375  4.955156
megabox              8  20.000  5.345225
```

코드 설명

```
# 총합에 root를 씌워서 반환해주는 myroot 함수
def myroot(values):
  mysum = sum(values)#총합 구하기
  return sqrt(mysum)#총합에 루트 씌워 반환

# 총합에 root를 씌운 다음 somevalue를 더하여 반환해주는 plus_add 함수
def plus_add(values, somevalue):
  result = myroot(values)#myroot 함수를 호출하여 총합에 루트 씌우기
  return result + somevalue#결과에 somevalue 더해 반환
```

코드 설명

```
#데이터 만들기
mygrouping = dftheater.groupby ('theater')['bindo']#데이터의 theater을 칼럼을 그룹핑하여 데이터 만들기
print ('groupby 와 사용자 정의 함수 사용하기')
result = mygrouping.agg (myroot)#mygrouping에 루트를 씌워 result에 결과 데이터 저장
print(result)#결과 데이터 출력
print('-' * 30)
#매개변수를 2개 사용하여 데이터 만들기
print('groupby와 사용자 정의 함수 (매개변수 2개) 사용하기')
result = mygrouping.agg (plus_add, somevalue=3)#plus_add함수를 사용하여 루트를 씌운 값에 3을 더하여 그룹핑한 결과 데이터 result 만들기
print(result)#결과값 출력
```

코드 설명

groupby 와 사용자 정의 함수 사용하기
theater
cgv 12.649111
daehan 13.964240
megabox 12.649111
Name: bindo, dtype: float64
---
groupby와 사용자 정의 함수 (매개변수 2개) 사용하기
theater
cgv 15.649111
daehan 16.964240
megabox 15.649111
Name: bindo, dtype: float64

코드 설명 (수직 막대 그래프 그리기)

```
#수직 막대 그래프 그리기

print('컬럼 2개 이상을 그룹핑하기')
newgrouping = dftheater.groupby (['theater', 'region'])['bindo']#데이터의
theater와 region 칼럼을 그룹화
result = newgrouping.count ()#그룹핑 결과의 빈도수 구하기
print(result)#결과 데이터 출력
print('-' * 30)
```

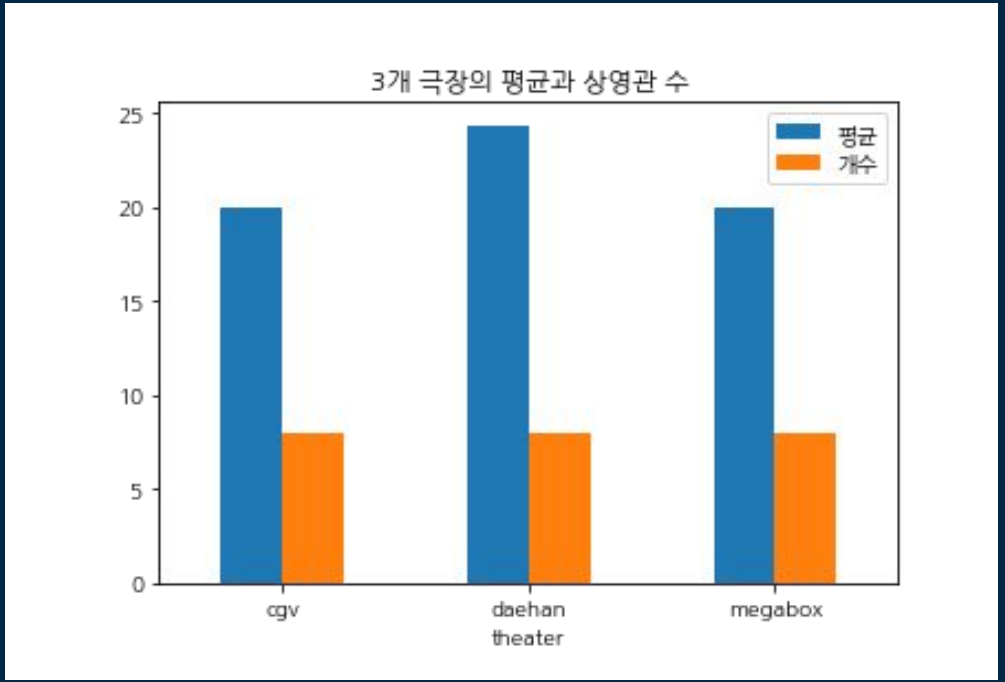
코드 설명 (수직 막대 그래프 그리기)

```
newDf = df.loc[:, ['평균','개수']]#그래프의 범례(legend)를 평균과 개수로 설정
newDf.plot(kind='bar', rot=0)#막대 그래프, 회전 0도로 그래프 그리기
plt.title('3개 극장의 평균과 상영관 수')
# pit.show()
filename='visualizationExam_02.png' #파일 이름 지정
plt.savefig(filename)#파일 저장
print(filename + '파일 저장됨')
```

수직 막대 그래프
:plot()의 kind을 'bar'으로 하면 수직 막대 그래프를 그릴 수 있음 (수평 막대 그래프는 'barh')

코드 설명 (수직 막대 그래프 그리기)

```
컬럼 2개 이상을 그룹핑하기
theater  region
cgv      강남        4
         신촌        4
daehan   강남        4
         신촌        4
megabox  강남        4
         신촌        4
Name: bindo, dtype: int64
------------------------------
visualizationExam_01.png 파일 저장됨
```



코드 설명 (파이 그래프 그리기)

```
#3개 극장에 대한 파이 그래프 그리기

labels = []  #labels : 원주 외곽에 보여줄 라벨
explode = (0, 0.03, 0.06)
#부각 정도 explode, megabox와 daehan은 그래프에서 더 부각되어 보이도록 설정
for key in sumSeries.index:  #sumSeries의 index의 모든 key에 대해 반복
  mydata = key + '(' + str(sumSeries[key]) + ')'
  #극장 이름(상영건수)의 형식으로 데이터 저장
  labels.append(mydata)  #mydata를 레이블에 추가
fig1, ax1 = plt.subplots()  #그림의 프레임 fig1, 그래프가 그려지는 공간 ax1으로
mytuple = tuple(labels)  #레이블을 튜플로 변경
```

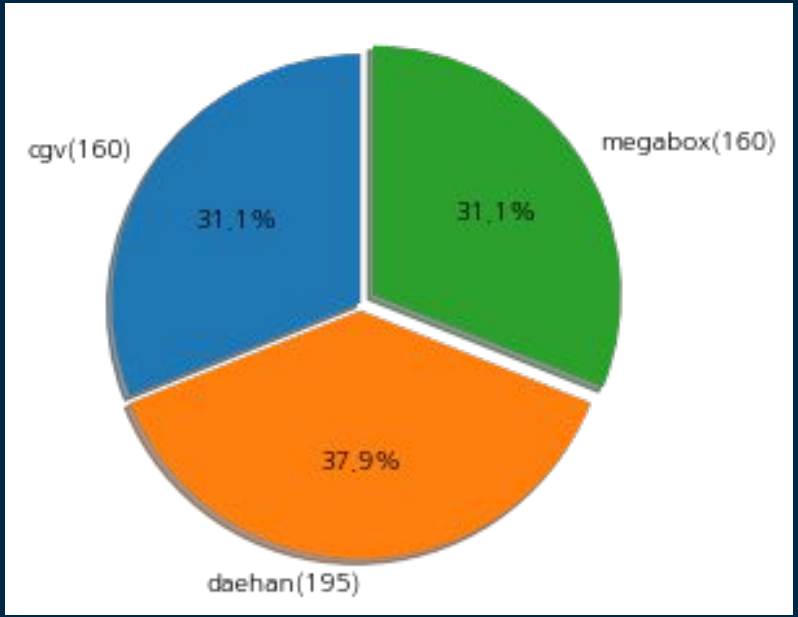
explode : 파이 그래프에서 툭 튀어나온 정도 (부각 정도)

코드 설명 (파이 그래프 그리기)

```
ax1.pie(sumSeries, explode=explode, labels=mytuple,
autopct='%1.1f%%',shadow=True, startangle=90)#sumSeries의 데이트를 이용,
이전에 지정한 부각 정도,레이블 지정, 값은 소수점 한자리로 하여 파이그래프 그리기
ax1.axis('equal')#파이 차트가 원의 형태를 유지하도록 설정
# plt.show()
filename='visualizationExam_03.png' #파일 이름 지정
plt.savefig(filename)#파일 저장
print (filename + ' 파일 저장됨')
```

autopct : 부채꼴 안에 표시될 숫자의 형식을 지정
shadow : True로 설정하면, 파이 차트에 그림자가 표시
startangle : 부채꼴이 그려지는 시작 각도

코드 설명 (파이 그래프 그리기)



THANKS

Do you have any questions?