WEEK08_STAGE3,4_장하은

STAGE3

#글씨를 중심으로 확인해주세요!

```
import pandas as pd

# 앞으로 pd.판다스 명령어 형식으로 진행하면 됨
#내용물은 딕셔너리 형식으로 만들어야함
s1 = pd.Series({
    '김': 100,
    '나': 141,
    '박': 129,
    '이': 124

})
s1.name = '수학'

print(s1)

s2 = pd.Series({
    '김': 30,
    '나': 5,
    '박': 1239,
    '이': 141

})
print(s2)

s3 = pd.Series([8,3,13,2,])
s3.name = '국어'
print(s3)
#시리즈를 리스트로 만들면 김나박이 없고 0123으로 나옴

s4 = pd.Series([7,16,19,278,])
print(s4)
```

```
김    100
나    141
박    129
이    124
Name: 수학, dtype: int64
김      30
나       5
박    1239
이     141
dtype: int64
0     8
1     3
2    13
3     2
Name: 국어, dtype: int64
0      7
1     16
2     19
3    278
dtype: int64
```

데이터프레임

In [85]:
```
df = pd.DataFrame({'출생지' : ['한국', '미국'],
    '나이' : [20,30]
})
#대소문자 구별 확실하게
#앞서 시리즈를 만들고 어딘가에 저장한 것 처럼 df에 =으로 저장

print(df)
#마치 표처럼
df
#제대로 표로 보기 좋음
```

```
  출생지  나이
0  한국  20
1  미국  30
```

Out[85]:

	출생지	나이
0	한국	20
1	미국	30

일련의 과정을 거쳐 만들어낸 표

In [99]:
```
df2
```

Out[99]:

	수학	영어	미술
김	100	30	1
나	141	5	2
박	129	1239	3
이	124	141	4

```
#프린트로 감쌌을때
print(df2.sum())
print(df2.mean(axis=1))
print(df2['수학'].mean())
# 세줄 모두 나옴
```

```
수학     494
영어    1415
미술      10
dtype: int64
김      43.666667
나      49.333333
박     457.000000
이      89.666667
dtype: float64
123.5
```

```
#프린트로 안감쌀때
#가장 아래 행만 출력 됨
df2.sum()
df2.mean(axis=1)
df2['수학'].mean()
```

```
123.5
```

각 명령과 기능 ¶

In [114]:
```
print(df2.loc[df2['영어']>=100, '수학'].max())
# 그 중 최댓값만 보여줘
```

```
129
```

In [115]:
```
print(df2.loc[df2['영어']>=100, '수학'].min())
# 그 중 최소값만 보여줘
```

```
124
```

In [116]:
```
print(df2.loc[df2['영어']>=100, '수학'].median())
#중간값만 보여줘
```

```
126.5
```

In [120]:
```
print(df2.loc[df2['영어']>=100, '수학'].var())
#분산 보여줘
```

```
12.5
```

In [121]:
```
print(df2.loc[df2['영어']>=100, '수학'].std())
#표준편차 보여줘
```

```
3.5355339059327378
```

In [126]:
```
print(df2.loc[df2['영어']>=100, '수학'].average())
#?
#일생 회화에서는 average를 많이 쓰고, mean은 수학자나 통계학자들이 주로 쓴다
#mean은  Arithmetic Mean이나, Harmonic Mean 등... 다양한 종류의 평균값이 존재한다
# 그 중에서 Arithmetic(산수) Mean 즉, 아주 쉽게 다 더해서 나누는 것을 Average라고 한다.
```

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
<ipython-input-126-96a3a7b47622> in <module>
----> 1 print(df2.loc[df2['영어']>=100, '수학'].average())
      2 #?
      3 #일생 회화에서는 average를 많이 쓰고, mean은 수학자나 통계학자들이 주로 쓴다
      4 #mean은  Arithmetic Mean이나, Harmonic Mean 등,,, 다양한 종류의 평균값이 존재한다
      5 # 그 중에서 Arithmetic Mean 즉, 아주 쉽게 다 더해서 나누는 것을 Average라고 한다.

C:₩ProgramData₩Anaconda3₩lib₩site-packages₩pandas₩core₩generic.py in __getattr__(self, name)
   5272             if self._info_axis._can_hold_identifiers_and_holds_name(name):
   5273                 return self[name]
-> 5274             return object.__getattribute__(self, name)
   5275
   5276     def __setattr__(self, name: str, value) -> None:

AttributeError: 'Series' object has no attribute 'average'
```

In [123]:
```
print(df2.loc[df2['영어']>=100, '수학'].count())
#갯수 보여줘
```

```
2
```

미션1 ¶

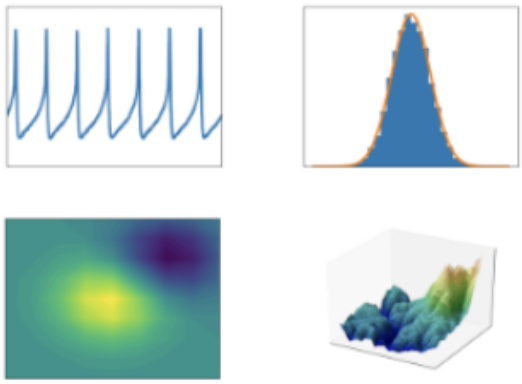
In [111]:
```
print(df2.loc[df2['영어']>=100, '수학'].mean())
# 해석하라(영어 점수가 100점 이상인 학생만 뽑아 그들의 수학점수 평균을 보여달라)
#.[](대괄호는) (콤마)를 기준으로 왼쪽은 행, 오른쪽은 열을 필터링합니다.
```

```
126.5
```

STAGE4

- **matplotlib**

파이썬으로 기본적인 차트들을 쉽게 그릴 수 있도록 도와주는 가장 유명한 데이터 시각화 라이브러리



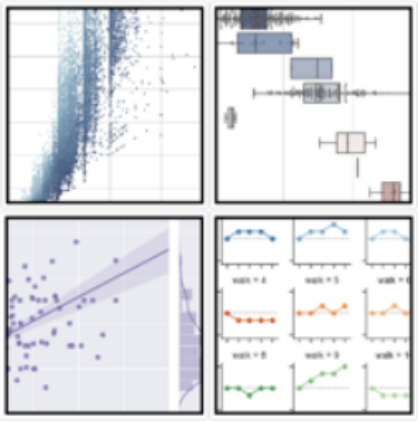
matplotlib으로 할 수 있는 일들

1. 막대 차트 그리기
2. 파이 차트 그리기
3. 라인 차트 그리기
4. …

* 기본적인 차트, plotting 작업에 충실

- **seborn**

matplotlib을 기반으로 만들어져 통계 데이터 시각화에 최적화된 인기 라이브러리



seaborn으로 할 수 있는 일들

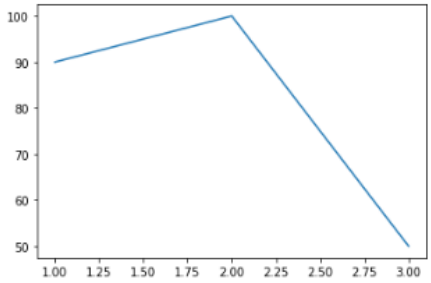
1. 데이터의 분포도 그리기
2. 히트맵 그리기
3. 박스 플롯 그리기
4. …

* 시각화 탁월하고 차트를 그리기 위한 계산 작업을 자동으로 처리.더 깊은 분석을 목표로 탐색적 분석에서 사용

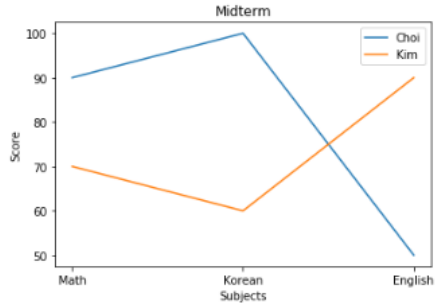
matplotlib 속성

In [1]:
```
import matplotlib.pyplot as plt

%matplotlib inline
```

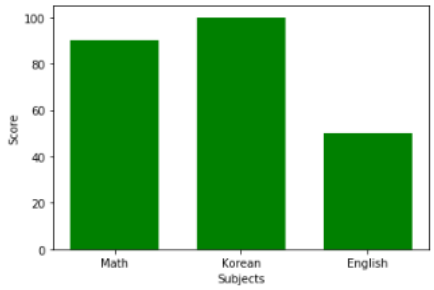
In [2]:
```
# (1,90) (2,100) (3,50)
plt.plot([1,2,3],[90,100,50])
plt.show()
```



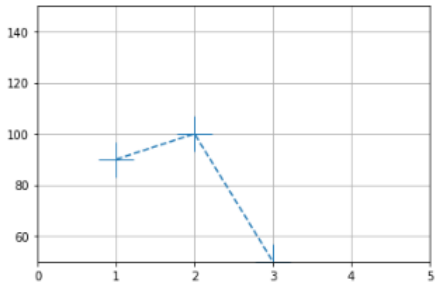
In [5]:
```
plt.plot(['Math', 'Korean', 'English'],[90,100,50])
plt.plot(['Math', 'Korean', 'English'],[70,60,90])
#한글은 글자가 깨짐. 해결방법은 나중에

#차트 구분하기
plt.legend(['Choi','Kim'])

#차트 세부사항 표시하기
plt.xlabel('Subjects')
plt.ylabel('Score')
plt.title('Midterm')

plt.show()
```



In [22]:
```
x = ['Math', 'Korean', 'English']
y = [90, 100, 50]

plt.xlabel('Subjects')
plt.ylabel('Score')

#막대그래프
plt.bar(x, y, width = 0.7, color = 'green' )

#색상에서 색상코드 입력가능

plt.show()
```



In [24]:
```
plt.plot([1,2,3],[90,100,50], marker='+', markersize = 30, linestyle='--')
# marker o > 동그라미 v > 삼각형 s> 사각형 + > 플러스 k > 오류

plt.grid(True)
#모눈
plt.axis([0,5,50,150])
# x축과 y축의 범위. 숫자를 줄이면 더 자세한 그래프

plt.show()
```



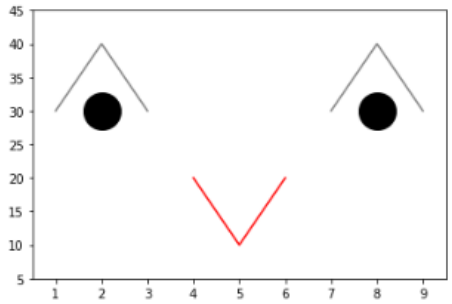
참고자료

line style	사용
	-
	--
	-.
	:
표시안됨	빈문자

marker	사용
	o
	v
	^
	s
	+
	.

미션1 ¶

In [41]:
```
#눈썹
plt.plot([1,2,3],[30,40,30],color = 'grey')
plt.plot([7,8,9],[30,40,30],color = 'grey')
#눈
plt.plot([2],[30],color = 'black',marker='o', markersize = 30)
plt.plot([8],[30],color = 'black',marker='o', markersize = 30)
#입
plt.plot([4,5,6],[20,10,20],color = 'red')
#테이블 범위
plt.axis([0.5,9.5,5,45])
plt.show()
```



판다스로 csv 데이터 불러오기

In [42]:
```
import pandas as pd
```

In [47]:
```
train = pd.read_csv('data/train.csv')
#''안에 들어있는 파일 csv를 판다스화(표로 그려줘)
train.head(5)
# .head는 간략화 / ()안에 숫자를 넣으면 그만큼 나옴
```

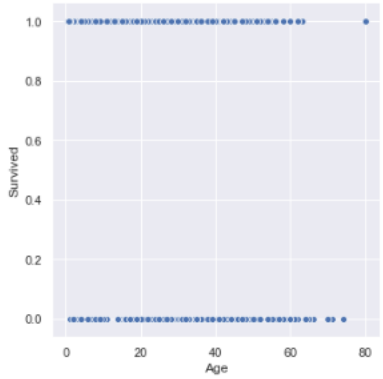
Out[47]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123	S
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	NaN	S

In [48]:
```
import matplotlib.pyplot as plt
import seaborn as sns
sns.set()
```

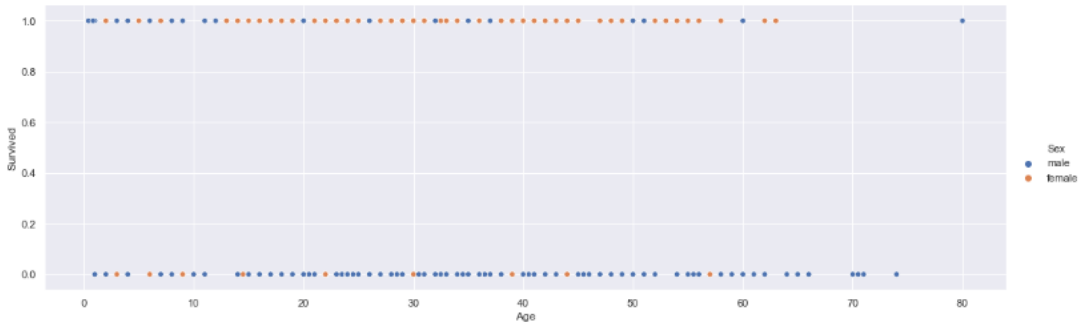
In [50]:
```
sns.relplot(data=train, x='Age', y='Survived')
```

Out[50]:
```
<seaborn.axisgrid.FacetGrid at 0x18d53196c48>
```



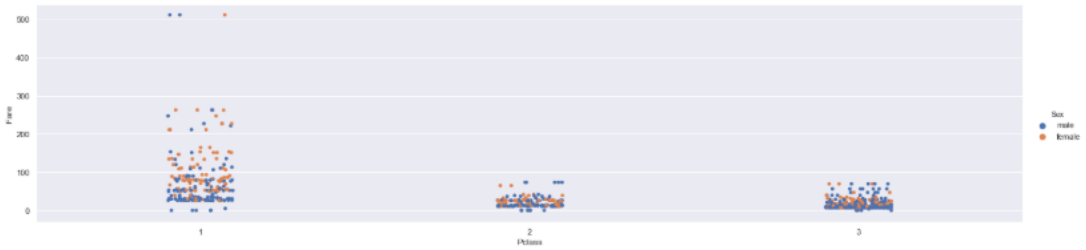
In [54]:
```
sns.relplot(data=train,x='Age', y='Survived', hue='Sex', aspect=3)
#aspect 정수값은 가로의 길이가 세로의 정수값배가 되도록 만들어줌
```

Out[54]:
```
<seaborn.axisgrid.FacetGrid at 0x18d5351d488>
```

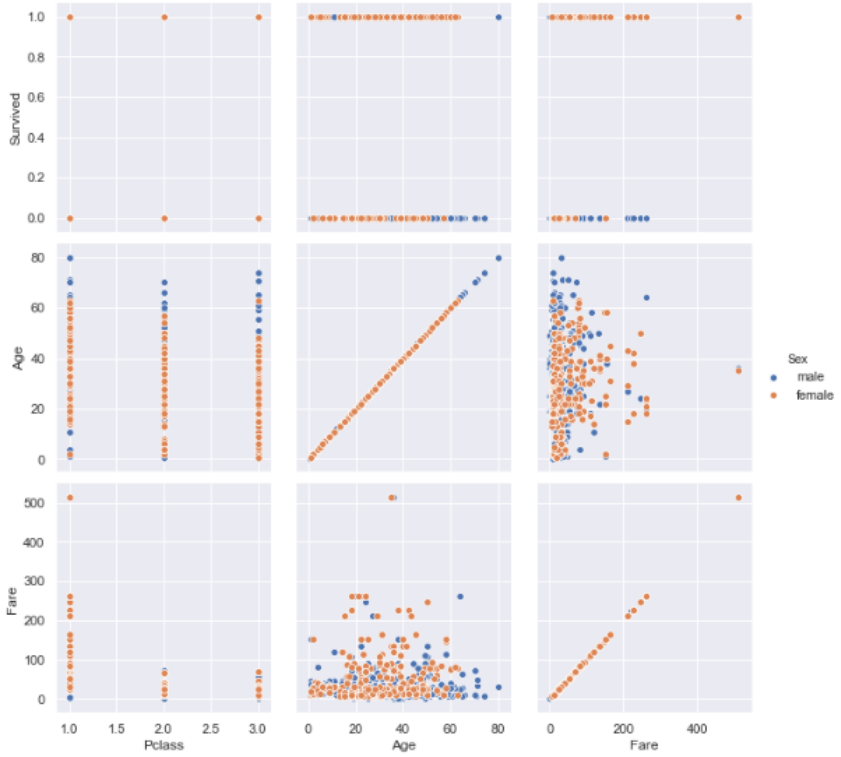


In [61]:
```
#카테고리 플롯 배우기
sns.catplot(data=train, x='Pclass', y='Fare', aspect=4, hue='Sex')
#jitter=False 추가시 직선모양 > jitter=True가 기본값
#대소문자 구분안하면 오류남
```

Out[61]:
```
<seaborn.axisgrid.FacetGrid at 0x18d54ad62c8>
```



```
In [65]: sns.pairplot(data=train,hue='Sex',
                      x_vars=['Pclass','Age','Fare'],
                       y_vars=['Survived','Age','Fare'], height=3)
Out[65]: <seaborn.axisgrid.PairGrid at 0x18d5676e048>
```

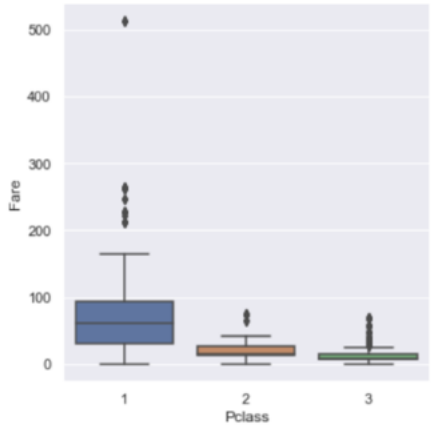


박스 그림 상자(Box Plot) 그려보기

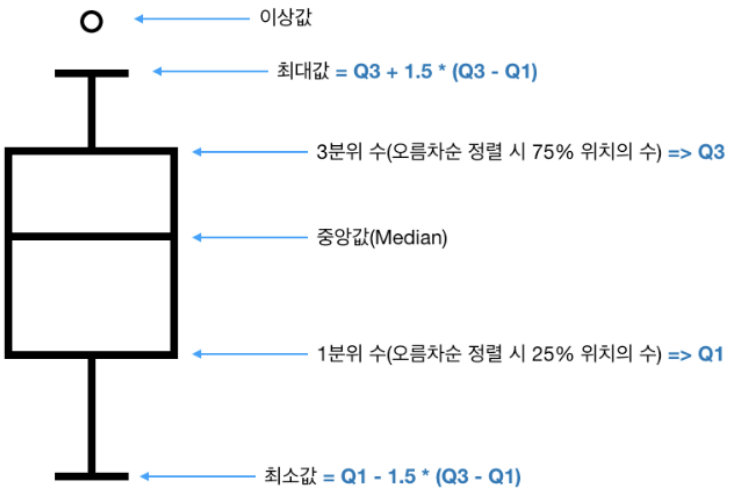
```
In [177]: sns.catplot(data=train, x='Pclass', y='Fare', kind='box')
          sns.catplot(data=train, x='Pclass', y='Fare', kind='box', hue='Sex')
```

executed in 457ms, finished 18:05:54 2019-01-02

```
Out[177]: <seaborn.axisgrid.FacetGrid at 0x1a25104978>
```



Box Plot 이해하기



Challenge2

In [1]:
```
import pandas as pd

train = pd.read_csv('data/train.csv')
#''안에 들어있는 파일 csv를 판다스화(표로 그려줘)
train.head(5)
# .head는 간략화 / ()안에 숫자를 넣으면 그만큼 나옴
```

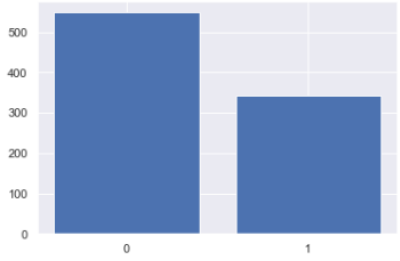
Out[1]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123	S
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	NaN	S

In [16]:
```
#train.loc[조건>죽은사람>train survived가 0인사람==,열]
#count를 해야 갯수가 나옴
dead_cnt = train.loc[train['Survived'] == 0, 'PassengerId'].count()
survived_cnt = train.loc[train['Survived'] == 1, 'PassengerId'].count()
print(dead_cnt, survived_cnt)
```

```
549 342
```

In [15]:
```
#dead는 0, 산사람은 1
plt.bar(['0', '1'], [dead_cnt, survived_cnt])
plt.show()
```



In [12]:
```
#차트 복사
dead_cnt = train[train['Survived'] == 0].shape[0]
survived_cnt = train[train['Survived'] == 1].shape[0]

# 단계2 : 각 그래프에 다른 색, 축 이름, 범례 설정
plt.bar(['0'], [dead_cnt], color='blue') # 파란색 지정
plt.bar(['1'], [survived_cnt], color='red') # 빨간색 지정
#'0','1'로 하는 이유는 숫자가 아니라 문자로 인식시키기 위해

plt.xlabel('Survived') # x축 이름
plt.ylabel('count') # y 축 이름
plt.legend(['Dead', 'Survived']) # 범례

plt.show()
```