이변량_범주 vs 숫자

1.환경준비

(1) 라이브러리 불러오기

In [2]:
```
import pandas as pd
import numpy as np

import matplotlib.pyplot as plt
import seaborn as sns

import scipy.stats as spst
```

(2) 데이터 불러오기

- 데이터 불러오기 : 다음의 예제 데이터를 사용합니다.

> 타이타닉 생존자

변수	설명	값 설명
survived	생존여부	0 - 사망, 1- 생존
Pclass	Ticket class	1 = 1st, 2 = 2nd, 3 = 3rd
Sex	성별	
Age	Age in years	
Sibsp	# of siblings / spouses aboard the Titanic	
Parch	# of parents / children aboard the Titanic	
Ticket	Ticket number	
Fare	Passenger fare	
Cabin	Cabin number	
Embarked	Port of Embarkation	C = Cherbourg, Q = Queenstown, S = Southampton

In [3]:
```
# 타이타닉 데이터
titanic = pd.read_csv('https://raw.githubusercontent.com/DA4BAM/dataset/master/titanic.0.csv')
titanic.head()
```

Out[3]:

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

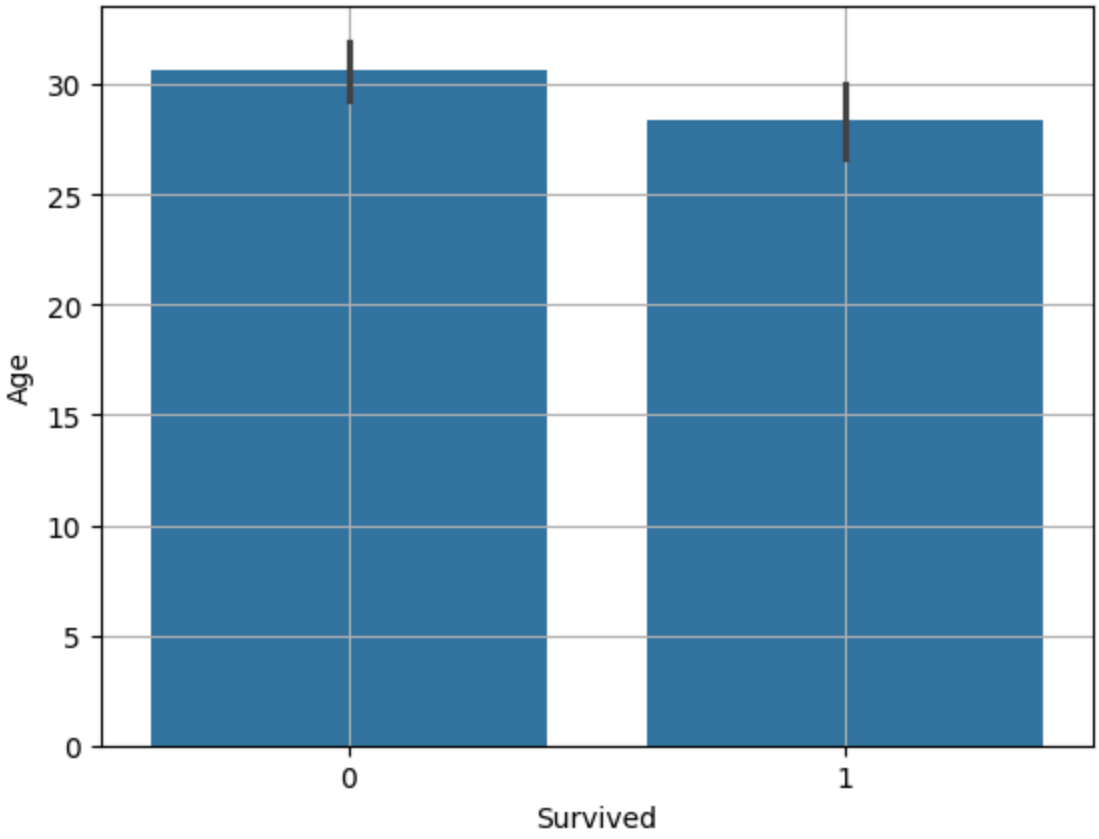
범주별 숫자를 비교할 때 사용되는 방식은 **범주별 평균 비교** 입니다.

2.시각화

titanic data에서 Age를 Y로 두고 비교해 봅시다.

(1) 평균 비교 : barplot

In [4]:
```
sns.barplot(x="Survived", y="Age", data=titanic)
plt.grid()
plt.show()
```



In [5]:
```
titanic.loc[:10, ['Survived','Age']]
```

Out[5]:

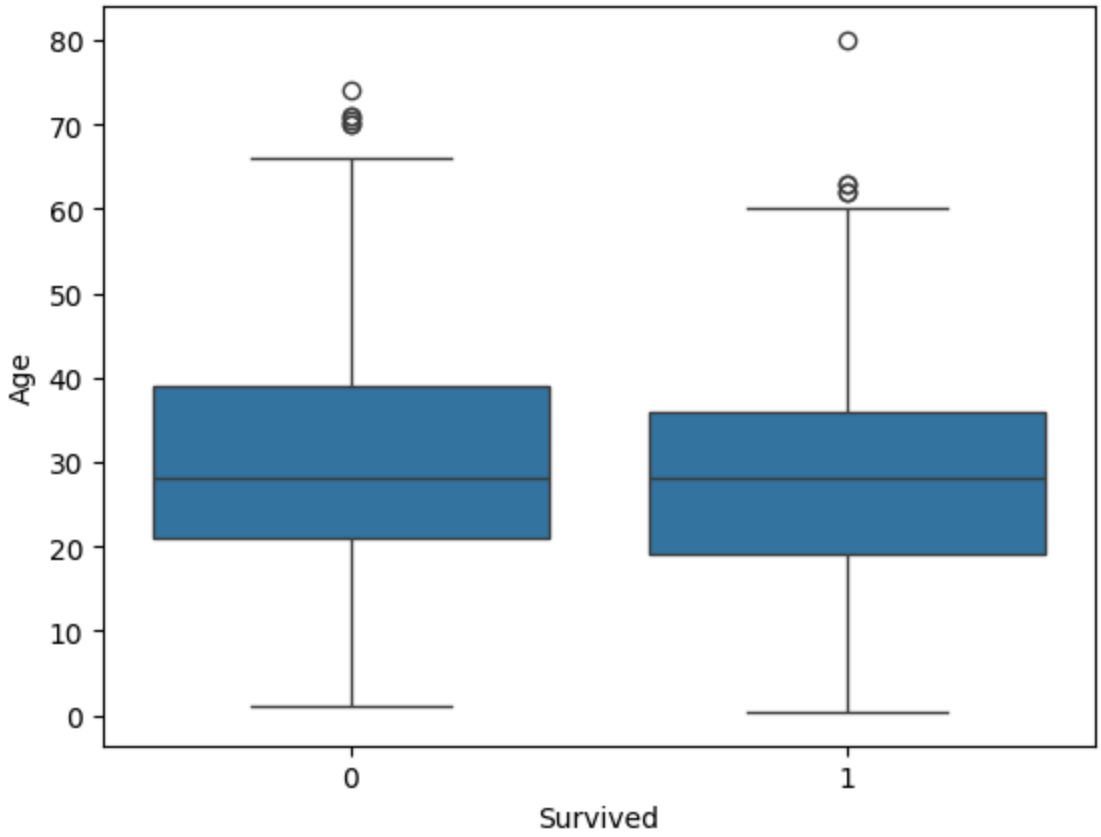
	Survived	Age
0	0	22.0
1	1	38.0
2	1	26.0
3	1	35.0
4	0	35.0
5	0	NaN
6	0	54.0
7	0	2.0
8	1	27.0
9	1	14.0
10	1	4.0

위 두 범주간에 평균에 차이가 있나요?

- 평균에 차이가 조금 있고 신뢰구간이 살짝 겹친다 ==> 애매하다

(2) (추가) boxplot

In [6]:
```
sns.boxplot(x='Survived', y = 'Age', data = titanic)
plt.show()
```



3.수치화

(1) t-test

두 집단의 평균을 비교합니다.

- 예제는 Two sample T-test와 양측검정만 다룹니다.
- 우리는 X --> Y의 관계에서 ,두 변수간에 관련이(차이가) 있는지, 없는지를 확인하는 것이 제일 중요하기 때문입니다.

- 주의사항 : 데이터에 NaN이 있으면 계산이 안됩니다. .notnull() 등으로 NaN을 제외한 데이터를 사용해야 합니다.

- t 통계량
    - 두 평균의 차이를 표준오차로 나눈 값.
    - 기본적으로는 두 평균의 차이로 이해해도 좋습니다.
    - 우리의 가설(대립가설)은 차이가 있다는 것이므로, t 값이 크던지 작던지 하기를 바랍니다.
    - 보통, t 값이 -2보다 작거나, 2보다 크면 차이가 있다고 봅니다.

- 이번엔 타이타닉 데이터로 시도해 봅시다.
  - 생존여부 --> Age : 생존여부 별로 나이에 차이가 있을것이다.

1) 데이터 준비

In [7]:
```
# 먼저 NaN이 있는지 확인해 봅시다.
titanic.isna().sum()
```

Out[7]:
```
PassengerId      0
Survived         0
Pclass           0
Name             0
Sex              0
Age            177
SibSp            0
Parch            0
Ticket           0
Fare             0
Cabin          687
Embarked         2
dtype: int64
```

In [8]:
```
# NaN 행 제외
temp = titanic.loc[titanic['Age'].notnull()]
```

In [9]:
```
# 두 그룹으로 데이터 저장
died = temp.loc[temp['Survived']==0, 'Age']
survived = temp.loc[temp['Survived']==1, 'Age']
```

2) t-test

In [10]:
```
# statistic : 통계량  2보다 크므로, 차이가 있으나 크지 않다
# statistic 양수는 died 평균값이 survived 보다 크다
spst.ttest_ind(died, survived)
```

Out[10]:
```
TtestResult(statistic=2.06668694625381, pvalue=0.03912465401348249, df=712.0)
```

In [11]:
```
spst.ttest_ind(survived, died)
```

Out[11]:
```
TtestResult(statistic=-2.06668694625381, pvalue=0.03912465401348249, df=712.0)
```

- [문1] 성별에 따라 운임에 차이가 있을 것이다.
  - 시각화와 수치화로 확인해 봅시다.

In [12]:
```
target = 'Fare'
feature = 'Sex'
```

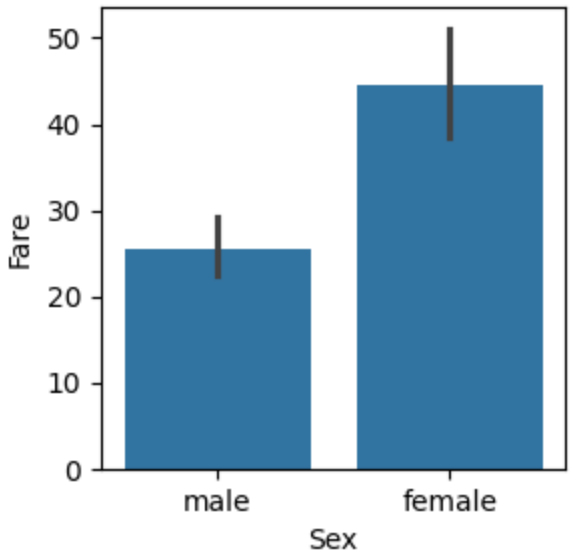
In [13]:
```
F_male = titanic.loc[titanic[feature]=='male', target]
F_female = titanic.loc[titanic[feature]=='female', target]
```

In [14]:
```
spst.ttest_ind(F_male, F_female)
```

Out[14]:
```
TtestResult(statistic=-5.529140269385719, pvalue=4.2308678700429995e-08, df=889.0)
```

In [15]:
```
plt.figure(figsize=(3, 3))
sns.barplot(x=feature, y=target, data=titanic)
```

Out[15]:
```
<Axes: xlabel='Sex', ylabel='Fare'>
```



- [문2] 생존여부에 따라 운임에 차이가 있을 것이다.
  - 시각화와 수치화로 확인해 봅시다.

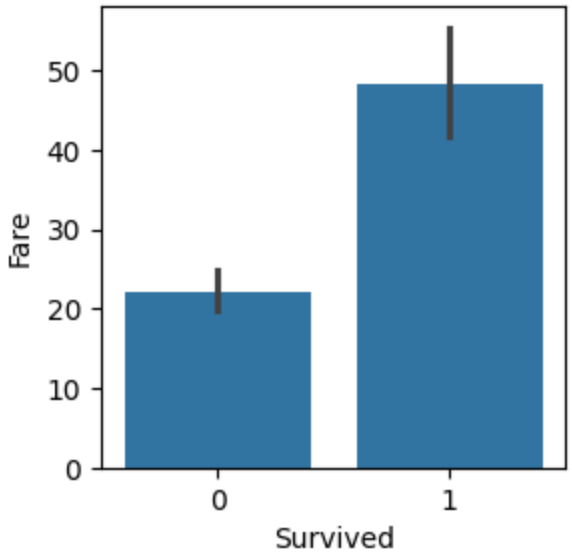
In [16]:
```
died = temp.loc[temp['Survived']==0, target]
survived = temp.loc[temp['Survived']==1, target]
```

In [17]:
```
spst.ttest_ind(died, survived)
```

Out[17]:
```
TtestResult(statistic=-7.428289683271724, pvalue=3.155994570484417e-13, df=712.0)
```

In [18]:
```
plt.figure(figsize=(3, 3))
sns.barplot(x='Survived', y=target, data=titanic)
```

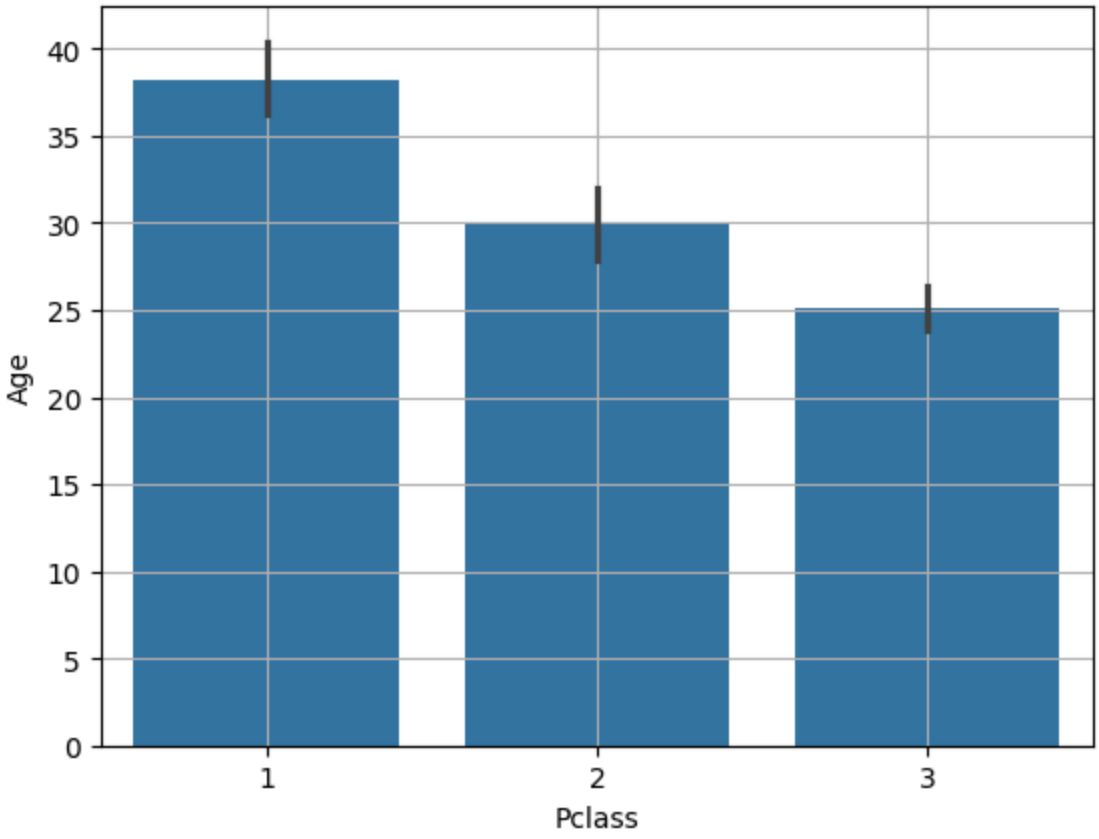
Out[18]:
```
<Axes: xlabel='Survived', ylabel='Fare'>
```



(2) anova

- 분산 분석 **AN**alysis **O**f **VA**riance
- 여러 집단 간에 차이는 어떻게 비교할 수 있을까요?
- 여기서 기준은 전체 평균 입니다.
- F 통계량 =
  - (집단 간 분산)/(집단 내 분산) = (전체 평균 – 각 집단 평균)/(각 집단의 평균 – 개별 값)
  - 값이 대략 2~3 이상이면 차이가 있다고 판단합니다.

In [19]:
```
# Pclass(3 범주) --> Age
sns.barplot(x="Pclass", y="Age", data=titanic)
plt.grid()
plt.show()
```



1) 데이터 준비

In [20]:
```
# 1) 분산 분석을 위한 데이터 만들기
# NaN 행 제외
temp = titanic.loc[titanic['Age'].notnull()]
# 그룹별 저장
P_1 = temp.loc[temp.Pclass == 1, 'Age']
P_2 = temp.loc[temp.Pclass == 2, 'Age']
P_3 = temp.loc[temp.Pclass == 3, 'Age']
```

2) anova

In [21]:
```
spst.f_oneway(P_1, P_2, P_3)
```

Out[21]:
```
F_onewayResult(statistic=57.443484340676214, pvalue=7.487984171959904e-24)
```

- [문1] 승선지역(Embarked)별로 운임에 차이가 있을 것이다.
  - 시각화와 수치화로 확인해 봅시다.

In [22]:
```
# 변수 설정
target = 'Fare'
feature = 'Embarked'
```

In [23]:
```
# 고유값 확인
titanic[feature].value_counts()
```

Out[23]:
```
Embarked
S    644
C    168
Q     77
Name: count, dtype: int64
```

In [24]:
```
# NaN값 확인
titanic[feature].isna().sum()
```

Out[24]:
```
2
```

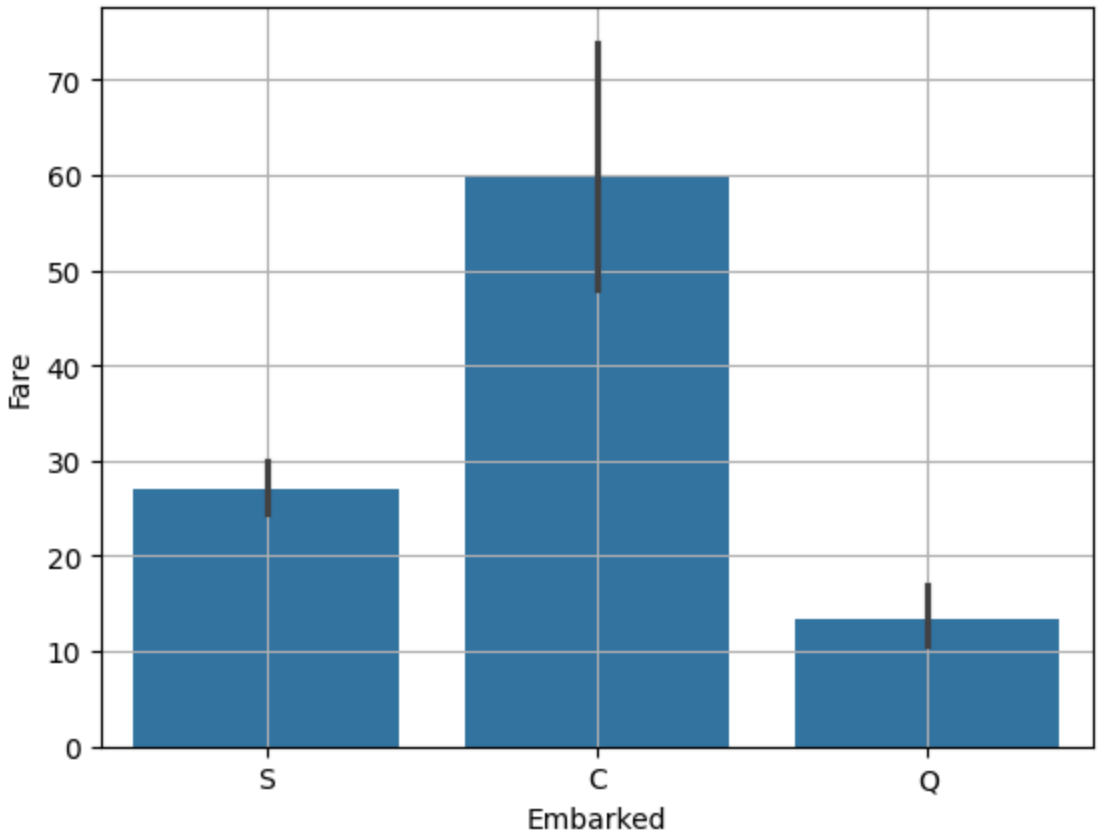
In [25]:
```
# NaN 제외
temp = titanic.loc[titanic[feature].notna()]
```

In [26]:
```
ride_S = temp.loc[temp.Embarked == 'S', target]
ride_C = temp.loc[temp.Embarked == 'C', target]
ride_Q = temp.loc[temp.Embarked == 'Q', target]
```

In [27]:
```
spst.f_oneway(ride_S, ride_C, ride_Q)
```

Out[27]:
```
F_onewayResult(statistic=38.14030520011266, pvalue=1.2896450252631794e-16)
```

In [28]:
```
sns.barplot(x=feature, y=target, data=titanic)
plt.grid()
plt.show()
```



- [문2] 객실등급(Pclass)별로 운임에 차이가 있을 것이다.
  - 시각화와 수치화로 확인해 봅시다.

In [29]:
```
temp['Pclass'].value_counts()
```

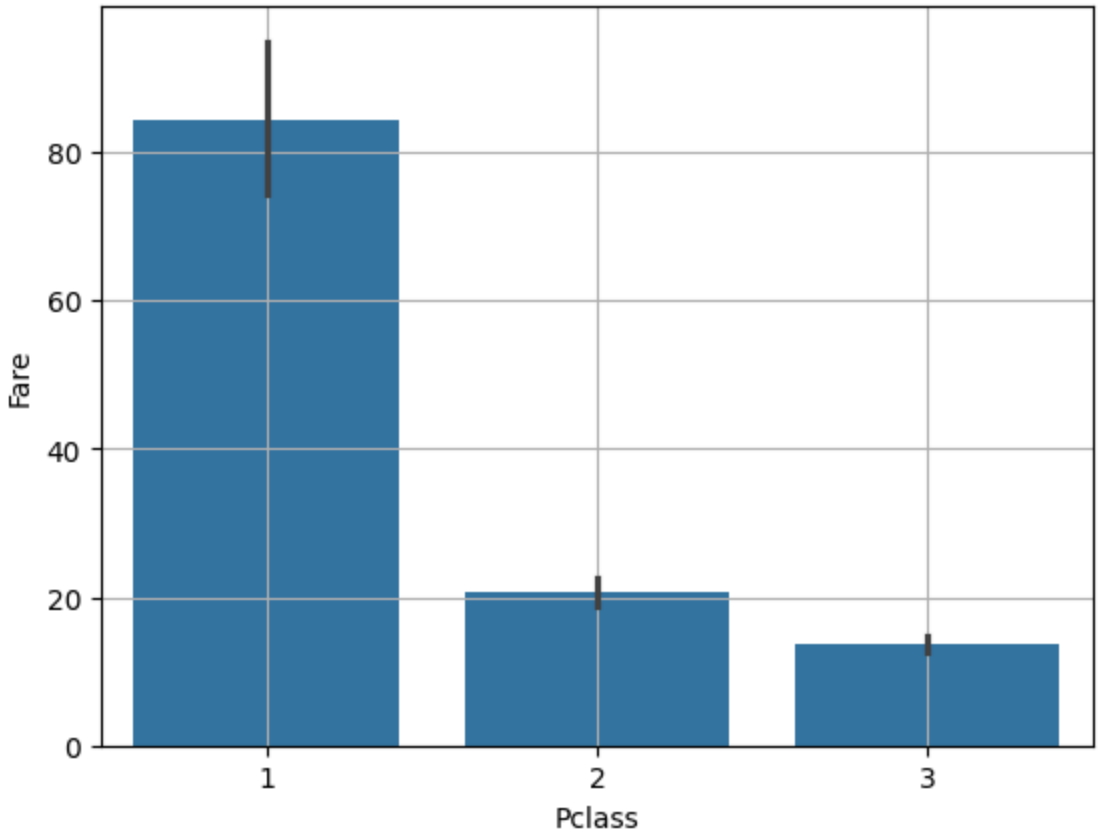
Out[29]:
```
Pclass
3    491
1    214
2    184
Name: count, dtype: int64
```

In [30]:
```
P_1 = temp.loc[temp.Pclass == 1, 'Fare']
P_2 = temp.loc[temp.Pclass == 2, 'Fare']
P_3 = temp.loc[temp.Pclass == 3, 'Fare']
```

In [31]:
```
spst.f_oneway(P_1, P_2, P_3)
```

Out[31]:
```
F_onewayResult(statistic=240.38829529293864, pvalue=3.9731247008614907e-84)
```

In [32]:
```
sns.barplot(x='Pclass', y=target, data=titanic)
plt.grid()
plt.show()
```



4.복습문제

air quality 데이터셋으로 다음 문제를 풀어 봅시다.

- 라이브러리 불러오기

In [33]:
```
import pandas as pd
import numpy as np
```

```
import random as rd

import matplotlib.pyplot as plt
import seaborn as sns

import scipy.stats as spst
```

- 데이터불러오기

In [34]:
```
# 뉴욕시 공기 오염도 데이터
air = pd.read_csv('https://raw.githubusercontent.com/DA4BAM/dataset/master/air2.csv')
air['Date'] = pd.to_datetime(air['Date'])
air['Month'] = air.Date.dt.month
air['Weekday'] = air.Date.dt.weekday
air['Weekend'] = np.where(air['Weekday'] >=5, 1, 0)
air.head()
```

Out[34]:

	Ozone	Solar.R	Wind	Temp	Date	Month	Weekday	Weekend
0	41	190.0	7.4	67	1973-05-01	5	1	0
1	36	118.0	8.0	72	1973-05-02	5	2	0
2	12	149.0	12.6	74	1973-05-03	5	3	0
3	18	313.0	11.5	62	1973-05-04	5	4	0
4	19	NaN	14.3	56	1973-05-05	5	5	1

- 1) 주말여부(Weekend) --> 오존농도(Ozone)와의 관계를 시각화하고, 가설검정을 수행해 봅시다.

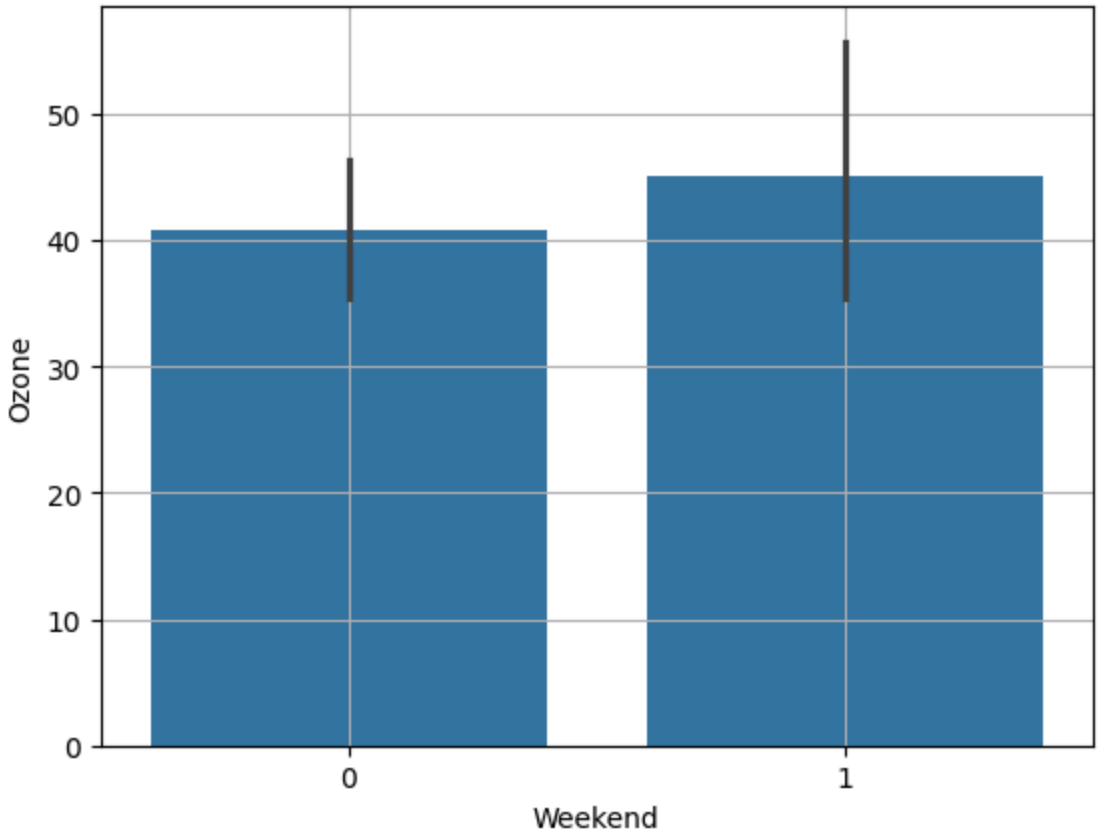
In [35]:
```
import scipy.stats as spst

we_0 = air.loc[air['Weekend']==0, 'Ozone']
we_1 = air.loc[air['Weekend']==1, 'Ozone']

spst.ttest_ind(we_0, we_1)
```

Out[35]:
```
TtestResult(statistic=-0.7671489829911908, pvalue=0.4441907648291733, df=151.0)
```

In [44]:
```
spst.f_oneway(we_0, we_1)
```

Out[44]:
```
F_onewayResult(statistic=0.5885175621044181, pvalue=0.44419076482917763)
```

In [36]:
```
# 두 값을 비교 할 때 barplot 사용
sns.barplot(x="Weekend", y="Ozone", data= air)
plt.grid()
plt.show()
```



- 2) 요일(Weekday) --> 오존농도(Ozone)와의 관계를 시각화하고, 가설검정을 수행해 봅시다.

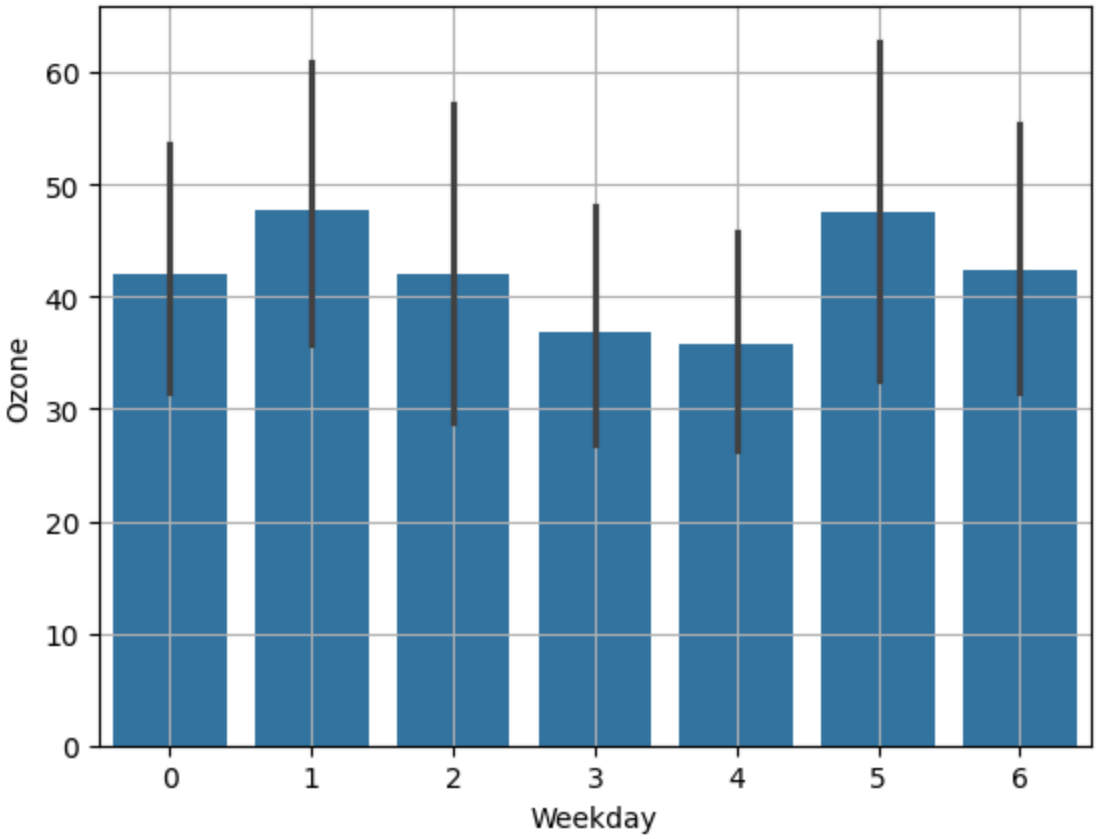
In [47]:
```
spst.pearsonr(air['Weekday'], air['Ozone'])
```

Out[47]:
```
PearsonRResult(statistic=-0.012899957964560387, pvalue=0.8742505879138937)
```

In [50]:
```
mon = air.loc[air['Weekday']==0, 'Ozone']
tu = air.loc[air['Weekday']==1, 'Ozone']
w = air.loc[air['Weekday']==2, 'Ozone']
th = air.loc[air['Weekday']==3, 'Ozone']
fri = air.loc[air['Weekday']==4, 'Ozone']
sat = air.loc[air['Weekday']==5, 'Ozone']
sun = air.loc[air['Weekday']==6, 'Ozone']

spst.f_oneway(mon, tu, w, th, fri, sat, sun)
```

Out[50]:
```
F_onewayResult(statistic=0.5098923426664418, pvalue=0.8001433644111904)
```

In [52]:
```
sns.barplot(x='Weekday', y='Ozone', data=air)
plt.grid()
plt.show()
```



- 3) 월(Month) --> 오존농도(Ozone)와의 관계를 시각화하고, 가설검정을 수행해 봅시다.

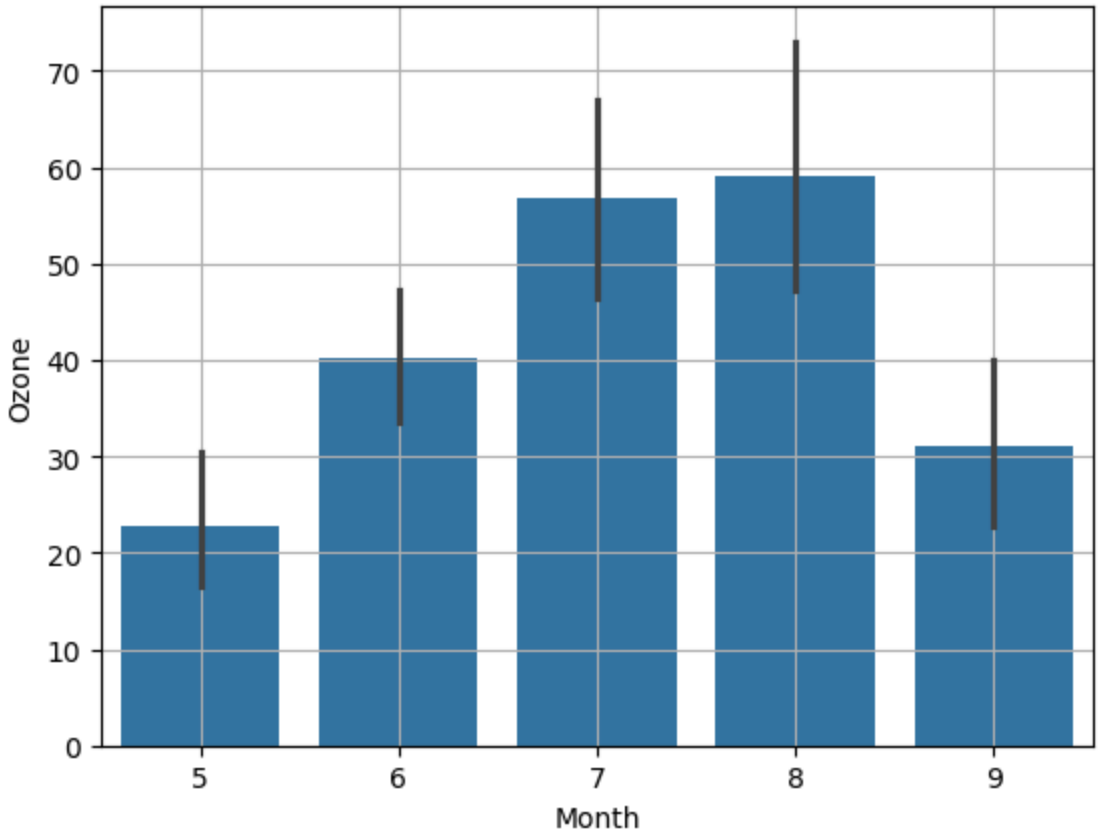
In [53]:
```
spst.pearsonr(air['Month'], air['Ozone'])
```

Out[53]:
```
PearsonRResult(statistic=0.1741967552245008, pvalue=0.03127803016499359)
```

In [55]:
```
m_5 = air.loc[air['Month']==5, 'Ozone']
m_6 = air.loc[air['Month']==6, 'Ozone']
m_7 = air.loc[air['Month']==7, 'Ozone']
m_8 = air.loc[air['Month']==8, 'Ozone']
m_9 = air.loc[air['Month']==9, 'Ozone']

spst.f_oneway(m_5, m_6, m_7, m_8, m_9)
```

Out[55]:
```
F_onewayResult(statistic=10.702965130677123, pvalue=1.2027079954529325e-07)
```

In [56]:
```
sns.barplot(x='Month', y='Ozone', data=air)
plt.grid()
plt.show()
```



In [ ]: