실습2 – 단순비교분석 및 분산분석

Jongchan Kim, Ph.D.
Division of Data Science
College of Software Digital Healthcare Convergence
Yonsei University Mirae Campus

연세대학교
YONSEI UNIVERSITY

독립표본/두표본 t 검정 (Two sample t-test, Independent t-test)

```
import numpy as np
from scipy import stats

# 예시 데이터 (그룹 A와 그룹 B의 데이터)
group_A = [12, 14, 16, 19, 15, 14, 13, 18, 16, 17]
group_B = [22, 23, 20, 21, 25, 22, 23, 24, 26, 25]

# 독립표본 t-검정 (equal_var=False는 등분산성을 가정하지 않음)
t_stat, p_value = stats.ttest_ind(group_A, group_B, equal_var=False)

print(f't-statistic: {t_stat}')
print(f'p-value: {p_value}')
```

독립표본/두표본 t 검정에 대한 신뢰구간 계산

```
import numpy as np
import scipy.stats as stats

# 두 그룹의 데이터
group_A = [12, 14, 16, 19, 15, 14, 13, 18, 16, 17]
group_B = [22, 23, 20, 21, 25, 22, 23, 24, 26, 25]

# 그룹 평균
mean1 = np.mean(group_A)
mean2 = np.mean(group_B)

# 표본 표준편차
std1 = np.std(group_A, ddof=1)
std2 = np.std(group_B, ddof=1)

# 표본 크기
n1 = len(group_A)
n2 = len(group_B)

# 표준 오차
SE = np.sqrt(std1**2 / n1 + std2**2 / n2)

# 자유도
df = n1 + n2 - 2

# t-분포의 임계값 (95% 신뢰수준)
t_critical = stats.t.ppf(0.975, df)

# 신뢰구간 계산
confidence_interval = (mean1 - mean2) - t_critical * SE,
(mean1 - mean2) + t_critical * SE

print(f"독립표본 t-검정 신뢰구간: {confidence_interval}")
```

쌍체/대응표본 t 검정 (Paired t-test)

```
import numpy as np
from scipy import stats

# 예시 데이터 (같은 그룹에서 처리 전후 데이터)
before = [12, 14, 16, 19, 15, 14, 13, 18, 16, 17]
after = [15, 17, 18, 20, 19, 17, 16, 21, 19, 18]

# 대응표본 t-검정
t_stat, p_value = stats.ttest_rel(before, after)

print(f't-statistic: {t_stat}')
print(f'p-value: {p_value}')
```

쌍체/대응표본 t 검정에 대한 신뢰구간 계산

```
# 대응 데이터
before = [12, 14, 16, 19, 15, 14, 13, 18, 16, 17]
after = [15, 17, 18, 20, 19, 17, 16, 21, 19, 18]

# 차이 계산
differences = [b - a for a, b in zip(before, after)]

# 차이의 평균과 표준편차
mean_diff = np.mean(differences)
std_diff = np.std(differences, ddof=1)

# 표본 크기
n = len(differences)

# 표준 오차
SE_diff = std_diff / np.sqrt(n)

# 자유도
df_diff = n - 1

# t-분포의 임계값 (95% 신뢰수준)
t_critical_diff = stats.t.ppf(0.975, df_diff)

# 신뢰구간 계산
confidence_interval_diff = (mean_diff - t_critical_diff * SE_diff,
mean_diff + t_critical_diff * SE_diff)

print(f"대응표본 t-검정 신뢰구간: {confidence_interval_diff}")
```

랜덤화 검정

```
import numpy as np

# 두 그룹의 데이터
group1 = np.array([12, 15, 14, 10, 13])
group2 = np.array([22, 25, 20, 21, 24])

# 그룹 평균 차이 계산
observed_diff = np.mean(group1) - np.mean(group2)

# 모든 데이터를 합친 후 랜덤 샘플링을 위해 결합
all_data = np.concatenate([group1, group2])

# 랜덤화 검정 설정
n_permutations = 10000  # 랜덤화 횟수
n_group1 = len(group1)
n_group2 = len(group2)

# 랜덤화에 의한 평균 차이 저장 리스트
random_diffs = []

# 랜덤화 검정 실행
for _ in range(n_permutations):
    np.random.shuffle(all_data)  # 데이터를 무작위로 섞음
    random_group1 = all_data[:n_group1]  # 섞인 데이터에서 그룹 1 크기만큼 선택
    random_group2 = all_data[n_group1:]  # 나머지를 그룹 2로 선택
    random_diff = np.mean(random_group1) - np.mean(random_group2)  # 평균 차이 계산
    random_diffs.append(random_diff)

# p-value 계산: 관측된 차이보다 큰 차이가 나올 확률
random_diffs = np.array(random_diffs)
p_value = np.sum(np.abs(random_diffs) >= np.abs(observed_diff)) / n_permutations

print(f"관측된 평균 차이: {observed_diff}")
print(f"p-value: {p_value}")
```

z 검정

```
import numpy as np
import scipy.stats as stats

# 가설 설정
mu = 100  # 모집단 평균 (귀무가설 H0에서 주어진 값)
sigma = 15  # 모집단 표준편차

# 표본 데이터
data = np.array([110, 115, 120, 130, 112, 118, 122, 127, 113, 116])

# 표본 평균과 크기
sample_mean = np.mean(data)
n = len(data)

# Z 통계량 계산
z = (sample_mean - mu) / (sigma / np.sqrt(n))

# p-value 계산 (양측 검정)
p_value = 2 * (1 - stats.norm.cdf(abs(z)))

# p-value 계산 (단측 검정)
P_value = 1 - stats.norm.cdf(abs(z))

# 결과 출력
print(f"Z 통계량: {z}")
print(f"p-value: {p_value}")
```

ANOVA 검정

```
import pandas as pd
import numpy as np
import scipy.stats as stats
import statsmodels.api as sm
from statsmodels.formula.api import ols

# 가상의 데이터 생성
np.random.seed(42)

# 세 집단의 데이터 생성
group1 = np.random.normal(24, 5, 30)
# 평균 24, 표준편차 5, 표본크기 30
group2 = np.random.normal(26, 5, 30)
# 평균 26, 표준편차 5, 표본크기 30
group3 = np.random.normal(30, 5, 30)
 # 평균 30, 표준편차 5, 표본크기 30

# 데이터프레임으로 변환
data = pd.DataFrame({
    'value': np.concatenate([group1, group2, group3]),
    'group': np.repeat(['Group1', 'Group2', 'Group3'], repeats=30)
})

# 데이터 확인
print(data.head())

# ANOVA 검정 수행
model = ols('value ~ group', data=data).fit()
anova_table = sm.stats.anova_lm(model, typ=2)

# 결과 출력
print(anova_table)
```