본 보고서의 목차는 과제 발제 pdf의 순서를 기본적으로 따른다.
코드를 설명하며 보고서를 작성하겠다.

2022147005 산업공학과 문형서

0. 필요 라이브러리 import

```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

from scipy import stats
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score
```

1. data 로드 및 구조 확인

```
iris = sns.load_dataset('iris')
# Display first few rows of the dataset
print(iris.head())
```

✓ 0.3s

```
   sepal_length  sepal_width  petal_length  petal_width species
0           5.1          3.5           1.4          0.2  setosa
1           4.9          3.0           1.4          0.2  setosa
2           4.7          3.2           1.3          0.2  setosa
3           4.6          3.1           1.5          0.2  setosa
4           5.0          3.6           1.4          0.2  setosa
```

생성 + 코드 + Markdown

```
iris.info()
```

✓ 0.0s

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 150 entries, 0 to 149
Data columns (total 5 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   sepal_length  150 non-null    float64
 1   sepal_width   150 non-null    float64
 2   petal_length  150 non-null    float64
 3   petal_width   150 non-null    float64
 4   species       150 non-null    object
dtypes: float64(4), object(1)
memory usage: 6.0+ KB
```

2. 기술통계량

```
# 기술통계량(종별 petal_length)
desc = iris.groupby("species")["petal_length"].describe()
count = iris["species"].value_counts()
desc, count
```

✓ 0.0s

```
(            count   mean       std  min  25%   50%    75%  max
 species
 setosa       50.0  1.462  0.173664  1.0  1.4  1.50  1.575  1.9
 versicolor   50.0  4.260  0.469911  3.0  4.0  4.35  4.600  5.1
 virginica    50.0  5.552  0.551895  4.5  5.1  5.55  5.875  6.9,
 species
 setosa        50
 versicolor    50
 virginica     50
 Name: count, dtype: int64)
```

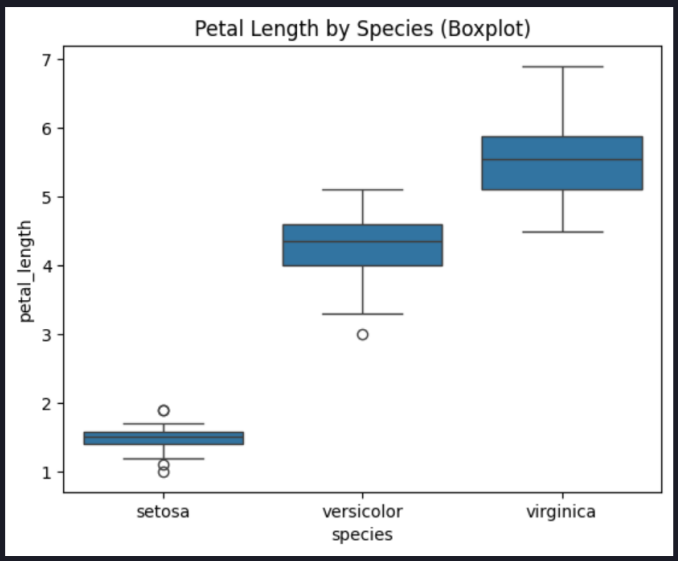
과제 명세에서 종별 통계량을 요구했기에 종을 기준으로 그룹바이를 시행한다.

추후 필요한 통계량을 출력한다.

3. 시각화

```
# 시각화
plt.figure()
sns.boxplot(data=iris, x="species", y="petal_length")
plt.title("Petal Length by Species (Boxplot)")
plt.show()
```

✓ 0.1s



명세에서 바라는 요구사항을 만족시키는 형태로 만든다.

4. 정규성 검정

```python
# 정규성 검정 (Shapiro-Wilk test)
alpha = 0.05
shapiro_results = []

for sp, grp in iris.groupby("species"):
    stat, p = stats.shapiro(grp["petal_length"])
    shapiro_results.append((sp, stat, p, "정규성 기각" if p < alpha else "정규성 기각 못함"))

shapiro_df = pd.DataFrame(shapiro_results, columns=["species", "W", "p_value", "decision@0.05"])
shapiro_df
```

0.0s Data Wrangler에서 'shapiro_df' 열기

	species	W	p_value	decision@0.05
0	setosa	0.9549767850318988	0.0548114671955363	정규성 기각 못함
1	versicolor	0.96600440254332	0.1584778381565757 3	정규성 기각 못함
2	virginica	0.9621864428612802	0.10977536903223506	정규성 기각 못함

데이터가 정규분포에서 추출되었다고 보아도 되는지를 검정하기 위해 정규성 검정을 시행한다. 명세에서 요구한 방법을 사용하였고 유의 수준으로 0.05를 설정하였고, 정규성을 기각하지 못하는 p-value가 나왔다.

우리는 본 과정에서 0.05보다 작은 값은 없기에 정규성을 기각하지 못하였고, 추후의 과정에서 정규성을 만족한다고 해석할 수 있다.

5. 등분산성 검정

```python
# 등분산성 검정
groups = [grp["petal_length"].values for _, grp in iris.groupby("species")]
stat, p = stats.levene(*groups, center="median")  # 보통 median 기반이 더 견고
stat, p, ("등분산성 기각" if p < alpha else "등분산성 기각 못함")
```

0.0s

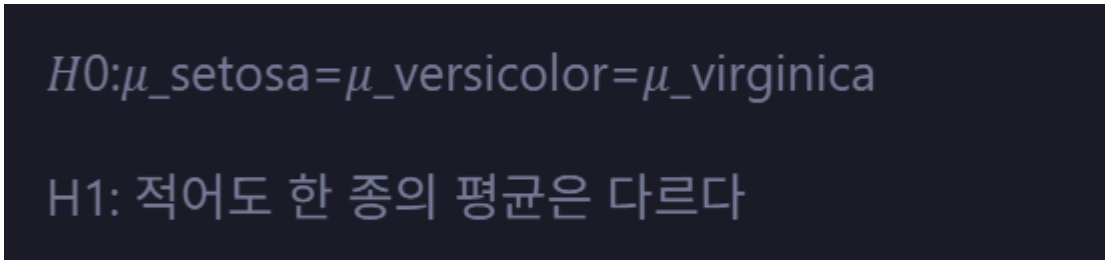
```
(np.float64(19.480338801923573), np.float64(3.1287566394085304e-08), '등분산성 기각')
```

등분산성은 분산이 같다는 의미이고, 이를 검정하는 것이 본 코드에서의 목표이다.

정규성과 등분산성이 성립해야 추후 진행할 아노바 분석의 결과를 유의미하게 해석할 수 있다. (만일 하나라도 성립하지 않다면 아노바 분석의 결과가 유의미하지 않을 수 있음.)

과학적 표기법으로 인해 수가 커 보일 수 있지만 엄밀하게 따지면 p-value는 0.05보다 과하게 작은 수로 결과가 나온다. 이로 하여금 등분산성은 기각된다. (등분산 상태가 아니다) 다만 명세에서 언급된 바로 하여금 등분산성을 가정하고 추후 아노바 분석을 진행하도록 한다.

6. 가설 수립



[H0: 세 종의 평균이 동일하다] 라고 서술할 수 있다.

7. 아노바

```
# scipy f_oneway
sp_names = iris["species"].unique()
data_by_sp = [iris.loc[iris["species"] == sp, "petal_length"].values for sp in sp_names]

F, p = stats.f_oneway(*data_by_sp)
F, p, ("귀무가설 기각" if p < alpha else "귀무가설 기각 못함")
```

✓ 0.0s

```
(np.float64(1180.1611822529785), np.float64(2.856776610961989e-91), '귀무가설 기각')
```

일원분산분석은 종(위에서 언급한 세개의 종)이 달라질 때 petal_length의 평균이 같은지, 다른지를 한 번에 검정하는 작업이다.

우리는 우선 평균이 모두 같은가를 확인하는 것이 목표다. (단순히 표본의 평균이 정량적으로 동일한지를 보는 것이 아니라 샘플링 과정에서의 모수 등을 모두 고려하여 통계적으로 동일하다고 볼 수 있는지를 파악)

본 과정에서의 결과를 보면, p-value의 값이 과학적표기법으로 하여금 낮은 수 (0.05보다 작은 수)로 나왔다. 따라서 우리는 H0를 기각해야 한다. 따라서 통계적으로 매우 강하게 다르다 (H1을 받아들인다)는 결과를 얻어낼 수 있다.

8. 사후검정

```python
from statsmodels.stats.multicomp import pairwise_tukeyhsd

tukey = pairwise_tukeyhsd(endog=iris["petal_length"],
                          groups=iris["species"],
                          alpha=0.05)
print(tukey)
```

✓ 0.2s

```
   Multiple Comparison of Means - Tukey HSD, FWER=0.05
=====================================================
  group1     group2   meandiff p-adj lower  upper  reject
-----------------------------------------------------
    setosa versicolor    2.798   0.0 2.5942 3.0018   True
    setosa  virginica     4.09   0.0 3.8862 4.2938   True
versicolor  virginica    1.292   0.0 1.0882 1.4958   True
-----------------------------------------------------
```

우리는 앞선 과정으로 3종의 평균이 강하게 다르다라는 사실을 알 수 있었다. 그렇다면 어느 쌍이 다른가를 엄밀하게 따지는 과정이 바로 사후검정 과정이다.

각 범주끼리의 쌍이 다른지를 모두 검정하는 과정이다.

여기서도 H0: 해당 쌍의 평균이 동일하다, H1: 해당 쌍의 평균이 다르다. 의 가설 수립을 통해 검정이 시행되며 모든 결과에서 H0을 기각하는 모습을 확인할 수 있다.

따라서 3종의 평균이 모두모두 다르다는 사실을 알 수 있다.

9. 결과 분석

2번에서 진행된 기술통계량과 3번에서 진행된 시각화 과정 중 하나인, 박스플롯에서 종(species)에 따라 petal_length의 분포와 중심값이 뚜렷하게 달라 평균 차이가 있을 가능성이 관찰된다.

4에서 진행된 (Shapiro–Wilk) 정규성 검정에서는 세 종 모두 유의수준 0.05에서 정규성을 기각하지 못해(정규성 가정을 반박할 근거 부족) 이후 모수 검정을 진행할 수 있다.

반면 5에서 진행된 Levene 등분산성 검정은 p-value가 매우 작아 등분산성이 기각되어 분산이 다를 수 있는 가능성이 높으나, 과제 명세에 따라 등분산을 가정하고 ANOVA를 수행하였다.

6에서 가설을 수립하고 7에서 시행한 일원분산분석 결과 p-value가 극히 작아 "세 종의

평균이 동일하다"는 귀무가설이 기각되며, 종에 따라 petal_length 평균이 통계적으로 매우 유의하게 다르다는 결과를 얻을 수 있었다.

최종적으로 8에서 진행한 Tukey HSD 사후검정에서도 모든 종 쌍에서 평균 차이가 유의하여, 세 종의 평균은 서로 모두 다르며 대체로 setosa < versicolor < virginica의 순서를 지지한다는 결과를 얻을 수 있다.

이는 통계 분석의 기본적인 흐름을 따라가는 매우 좋은 실습 과제였습니다!

10. 회귀 분석

- 입력: sepal_length, sepal_width, petal_width
- 타겟: petal_length
- Train/Test 분리
- Linear Regression 학습
- MSE, R^2, 회귀계수 출력 및 해석

회귀 분석은 입력으로 하여금 타겟을 그나마 가깝게 예측하는 과정이다.

시계열 데이터 형식이 아니기에 셔플을 허용하여 데이터 셋을 트레인과 테스트로 분리하게 한다.

변수가 여러 종류이기에 (입력) 다중선형회귀 방식을 차용한다.

```python
X = iris[["sepal_length", "sepal_width", "petal_width"]]
y = iris["petal_length"]

X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.2, random_state=42
)

model = LinearRegression()
model.fit(X_train, y_train)

pred = model.predict(X_test)

mse = mean_squared_error(y_test, pred)
r2 = r2_score(y_test, pred)

coef = pd.Series(model.coef_, index=X.columns)
intercept = model.intercept_

mse, r2, intercept, coef
```

✓ 0.0s

```
(0.13001626031382682,
 0.9603293155857664,
 np.float64(-0.2621959025887066),
 sepal_length    0.722815
 sepal_width    -0.635816
 petal_width     1.467524
 dtype: float64)
```

mse가 0.13 수준이며, r^2는 0.96 수준인 것으로 보아 강한 설명력을 띄는 모델이 만들어졌음을 알 수 있다.

(해당 mse는 타겟 변수의 스케일을 고려하지 않았기에 추후 보정이 필요할 수 있음)

계수를 통해 입력 변수가 어떤 영향력을 미치는지도 볼 수 있다. (이는 스케일을 함께 봐야 한다.)

이상입니다! 감사합니다.