# 기초 통계 및 ML 보고서

민세희

July 17, 2025

## 1 Iris 데이터셋 기반 기초 통계 분석

### 1.1 데이터 준비 및 탐색

#### 데이터 로드

seaborn 라이브러리의 `load_dataset()` 함수를 이용하여 Iris 데이터셋을 불러오고, `head()` 함수를 통해 전체 데이터의 구조를 확인하였다.

#### 기술통계량 산출

`groupby()` 함수를 통해품종별로 Petal Length에 대한 평균, 표준편차, 사분위수 등의 통계치를 산출하였다. Setosa의 평균은 1.46cm로 가장 짧았고, Virginica는 5.55cm로 가장 길었다.

| Species | count | mean | std | min | 25% | 50% | 75% | max |
|---|---|---|---|---|---|---|---|---|
| Setosa | 50 | 1.46 | 0.17 | 1.0 | 1.4 | 1.5 | 1.575 | 1.9 |
| Versicolor | 50 | 4.26 | 0.47 | 3.0 | 4.0 | 4.35 | 4.6 | 5.1 |
| Virginica | 50 | 5.55 | 0.55 | 4.5 | 5.1 | 5.55 | 5.875 | 6.9 |

Table 1: Species별 Petal Length에 대한 기술통계량

#### 시각화

Boxplot을 통해 그룹 간 Petal Length의 분포 차이를 시각화하였다. 종별로 비교한 결과, 평균 Petal Length는 virginica > versicolor > setosa 순으로 높게 나타났다.

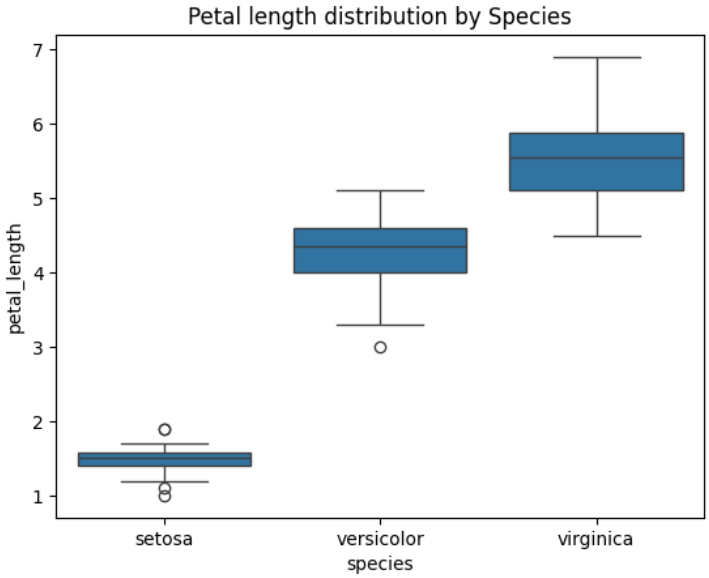


Figure 1: Species별 Petal Length 분포

## 1.2 통계적 검정

### 정규성 검정

- 귀무가설 ($H_0$): 데이터는 정규성을 따른다
- 대립가설 ($H_1$): 데이터는 정규성을 따르지 않는다
- 유의수준: $\alpha = 0.05$

Shapiro-Wilk 검정을 통해 품종별로 정규성을 검정하였다. 유의수준 $\alpha = 0.05$ 기준으로, 모든 품종의 p-value는 각각 0.0548 (Setosa), 0.1585 (Versicolor), 0.1098 (Virginica)로, 모두 유의수준보다 크기 때문에 귀무가설을 기각하지 못하였다. 즉, 세 그룹 모두 정규성을 만족한다고 판단하였다.

### 등분산성 검정

- 귀무가설 ($H_0$): 데이터는 등분산을 따른다
- 대립가설 ($H_1$): 데이터는 등분산을 따르지 않는다
- 유의수준: $\alpha = 0.05$

Levene 검정을 통해 그룹 간 분산이 동일한지를 검정하였다.
세 그룹 간 쌍별로 비교한 결과:

- Setosa와 Versicolor: p = 2.74e-07 $<$ 0.05 $\rightarrow$ 등분산 아님
- Setosa와 Virginica: p = 8.87e-09 $<$ 0.05 $\rightarrow$ 등분산 아님
- Versicolor와 Virginica: p = 0.304 $>$ 0.05 $\rightarrow$ 등분산 만족

이처럼 일부 쌍에서는 등분산 가정이 성립하지 않았지만, 전체 분석에서는 과제 지시에 따라 등분산성을 가정하고 이후 ANOVA를 진행하였다.

#### 가설 수립

- 귀무가설 ($H_0$): 3개 Species 간 Petal Length의 평균이 모두 같다
- 대립가설 ($H_1$): 적어도 하나의 Species는 Petal Length의 평균이 다르다
- 유의수준: $\alpha = 0.05$

#### ANOVA

세 그룹 간 평균 Petal Length의 차이를 검정하기 위해 one-way ANOVA를 실시하였다. 그 결과 F값은 약 1180.16, p-value는 $2.86 \times 10^{-91}$로, 유의수준보다 매우 작게 나타났다. 따라서 귀무가설을 기각하고, 적어도 한 그룹 이상에서 평균 차이가 존재함을 확인하였다.

#### 사후 검정

Tukey HSD 사후검정을 통해 어떤 그룹 간 차이가 유의한지를 확인하였다. 결과는 다음과 같다:

- Setosa vs. Versicolor: 평균 차이 2.798, $p < 0.001$ → 유의미한 차이
- Setosa vs. Virginica: 평균 차이 4.09, $p < 0.001$ → 유의미한 차이
- Versicolor vs. Virginica: 평균 차이 1.292, $p < 0.001$ → 유의미한 차이

즉, 모든 그룹 간 평균 Petal Length에 통계적으로 유의한 차이가 있었다.

### 1.3 결과 요약

Boxplot, ANOVA, 그리고 Tukey HSD 사후검정 결과를 종합하면 다음과 같다.

Boxplot을 통해 각 품종별 평균 Petal Length는 **virginica** > **versicolor** > **setosa** 순으로 높은 경향을 보였다. ANOVA 결과, 세 그룹 간 평균에 통계적으로 유의미한 차이가 있음이 확인되었으며, Tukey HSD 사후검정을 통해 모든 그룹 쌍 간에 유의미한 평균 차이가 존재함이 드러났다.

따라서, **virginica의 Petal Length가 가장 길고, setosa가 가장 짧으며, versicolor는 중간 수준**이라는 결론을 내릴 수 있다.

## 2 신용카드 사기 분류 모델

### 2.1 데이터 전처리

#### 데이터 로드 및 기본 탐색

신용카드 사기 탐지 데이터셋 `creditcard.csv`를 불러오고, `head()` 함수를 통해 전체 데이터의 구조를 확인하였다. 데이터의 Class 변수 분포는 다음과 같았다.

- 정상 거래 (Class=0): 27,725건
- 사기 거래 (Class=1): 93건

Class=1의 비율은 약 0.33%로, 극심한 클래스 불균형을 보이는 데이터임을 확인하였다.

### 샘플링

Class=1 (사기 거래) 데이터는 전체 93건을 모두 유지하였고, Class=0 (정상 거래) 데이터 중 10,000건을 무작위로 추출하였다. 이후 두 데이터를 병합하여 학습용 데이터셋 `credit_sample`을 구성하였다. 샘플링 이후 Class 비율은 다음과 같다:

- Class=0: 약 99.08%
- Class=1: 약 0.92%

### 데이터 전처리

금액 변수인 `Amount`만 표준화를 적용하였다. `StandardScaler`를 이용해 `Amount_Scaled`라는 새로운 컬럼으로 대체하고, 기존 `Amount` 컬럼은 제거하였다. 이후 데이터셋을 다음과 같이 분리하였다:

- `X`: 입력 변수 (Class 제외한 모든 컬럼)
- `y`: 타깃 변수 (Class)

### 데이터 분할

전체 데이터를 학습용(train)과 테스트용(test)으로 8:2 비율로 분할하였다. 이때 `train_test_split()` 함수의 `stratify=y` 옵션을 통해 클래스 비율이 유지되도록 하였다. 분할 후 클래스 비율은 다음과 같이 확인되었다:

- Train set – Class=1: 0.009165
- Test set – Class=1: 0.009411

### SMOTE

모델이 소수 클래스를 충분히 학습할 수 있도록 유도하기 위하여 `SMOTE`를 적용하였다. 학습 데이터에 대해 `SMOTE`를 적용하여 소수 클래스(Class=1)를 오버샘플링하였다. 적용 전에는 소수 클래스의 수가 74건에 불과했지만, SMOTE 적용 후 8000건으로 균형이 맞춰졌다. 이를 통해 모델이 소수 클래스를 충분히 학습할 수 있는 기반을 마련하였다.

- Before SMOTE – Class=1: 74건
- After SMOTE – Class=1: 8,000건

## 2.2 모델 학습 및 성능 평가

### 모델 학습

Logistic Regression 모델을 기반으로 GridSearchCV를 이용해 하이퍼파라미터 튜닝을 수행하였다. 탐색한 파라미터는 다음과 같다:

- `C`: [0.01, 0.05, 0.1, 0.2, 0.5, 1, 5, 10]
- `penalty`: l2
- `solver`: liblinear
- `class_weight`: balanced

교차검증은 `cv=5`, 평가 지표는 `average_precision`으로 설정하였다. 최적의 파라미터는 다음과 같이 선정되었다:

- C = 10, penalty = l2, solver = liblinear, class_weight = balanced

최종 모델은 테스트 데이터에 대해 예측 확률(`predict_proba`)을 기반으로 threshold를 조정하여 F1-score가 최대가 되는 최적 threshold를 탐색하였다. 탐색 결과, `threshold = 0.8`일 때 F1-score가 `0.9744`로 가장 높았다.

### 최종 성능 평가

최적 threshold = 0.8을 기준으로 최종 예측을 수행한 결과, 다음과 같은 성능을 확인하였다:

| **Class** | **Precision** | **Recall** | **F1-score** | **Support** |
|---|---|---|---|---|
| 0 (정상) | 1.00 | 1.00 | 1.00 | 2000 |
| 1 (사기) | 0.95 | 1.00 | 0.97 | 19 |
| Accuracy | | | 1.00 | |
| Macro avg | 0.97 | 1.00 | 0.99 | 2019 |
| Weighted avg | 1.00 | 1.00 | 1.00 | 2019 |

Table 2: Logistic Regression 모델 성능 요약 (threshold = 0.8)

또한, PR-AUC (Average Precision Score)는 **0.9974**로 매우 높은 값을 기록하였다.

과제에서 제시한 성능 기준:

- Recall $\geq 0.80$
- F1-score $\geq 0.88$
- PR-AUC $\geq 0.90$

모든 항목을 초과 달성하였으며, Logistic Regression 모델이 불균형 데이터 상황에서도 SMOTE와 threshold 조정, 하이퍼파라미터 조정을 통해 충분히 효과적으로 작동했음을 확인할 수 있다.