

통계적 가설검정 입문

기댓값과 분산으로 이해하는 가설검정의 원리

이 가이드는 기댓값과 분산 개념은 알고 있지만
통계적 가설검정은 처음 접하는 독자를 대상으로 합니다.

들어가며

이 가이드는 통계적 가설검정을 처음 배우는 분들을 위한 입문서입니다. 기댓값과 분산이라는 두 개념만 알고 있다면 충분히 따라올 수 있도록 구성했습니다.

1 부에서는 가설검정이 “왜” 그런 절차를 따르는지를 다루고, 2 부에서는 실제로 가설검정을 “어떻게” 수행하는지를 다룹니다. 증명은 가능한 한 생략하고, 핵심적인 직관과 결과 위주로 설명합니다.

이 가이드에서 다루지 않는 것

다중검정 문제, p-hacking 등 가설검정을 반복적으로 사용할 때 생기는 함정은 다루지 않습니다. Z-test와 t-test 외의 다양한 검정 방법(카이제곱검정, 분산분석 등)은 별도의 가이드에서 다룰 예정입니다.

1 부. 가설검정은 무엇을 하는 도구인가

1 장. 불확실성 속의 의사결정

“이 신약은 기존 약보다 효과가 있는가?” “이번 달 불량률이 예년과 달라졌는가?” — 이런 질문들의 공통점은, 우리가 모든 데이터를 가지고 있지 않은 상태에서 행동을 정해야 한다는 점입니다. 표본 일부만 보고 전체에 대한 결론을 내려야 하므로, 우리의 결정은 항상 틀릴 가능성을 안고 있습니다.

이때 “틀리면 손해를 본다”는 생각이 자연스럽게 따라옵니다. 신약이 효과가 없는데 있다고 판단하면 불필요한 비용과 위험을 감수하게 되고, 반대로 효과가 있는데 없다고 판단하면 좋은 치료법을 놓치게 됩니다. 가설검정은 바로 이렇게 “불확실한 정보로부터 합리적인 행동을 고르는” 절차입니다.

이 가이드는 이 합리적 선택의 논리를 기댓값과 분산이라는, 이미 알고 계신 도구로 풀어نا갑니다.

2 장. 손실에서 위험으로

어떤 결정이 “얼마나 틀렸는지”를 숫자로 표현한 것을 손실이라고 부릅니다. 가장 단순한 형태는, 맞으면 손실 0, 틀리면 손실 1 을 주는 방식입니다.

$$\text{손실} = 0 \text{ (결정이 맞았을 때)}, \text{ 손실} = 1 \text{ (결정이 틀렸을 때)}$$

그런데 손실 하나만 보고는 결정이 좋은지 평가하기 어렵습니다. 데이터는 매번 다르게 관측되는 확률적인 대상이기 때문에, 같은 결정 규칙이라도 어떤 표본을 만나느냐에 따라 맞을 수도, 틀릴 수도 있습니다. 그래서 손실을 표본의 확률분포에 대해 평균을 낸 것, 즉 손실의 기댓값을 봅니다. 이것을 위험이라고 부릅니다.

가설검정은 이 틀에서 다음과 같은 특수한 경우에 해당합니다.

- 할 수 있는 행동은 “귀무가설을 기각한다” 또는 “기각하지 않는다” 두 가지뿐입니다.
- 손실은 결정이 맞으면 0, 틀리면 1 인 단순한 형태를 가정합니다.

이 단순한 틀 안에서, 위험(손실의 기댓값)은 결국 “내가 틀릴 확률”이 됩니다. 그리고 가설검정에서 틀리는 방법은 한 가지가 아니라 두 가지입니다. 이걸 다음 장에서 살펴봅니다.

3 장. 두 가지 틀리는 방법, 그리고 그 사이의 선택

귀무가설 H_0 이 실제로 참인지 거짓인지, 그리고 우리가 그것을 기각했는지 기각하지 않았는지를 표로 정리하면 다음과 같습니다.

	H_0 을 기각하지 않음	H_0 을 기각함
H_0 이 실제로 참	올바른 결정	1 종 오류 (확률 α)
H_0 이 실제로 거짓	2 종 오류 (확률 β)	올바른 결정

1 종 오류 확률을 α , 2 종 오류 확률을 β 라고 부릅니다. 이 둘은 사실 위험을 H_0 가 참인 경우와 거짓인 경우로 나누어 본 것에 지나지 않습니다.

여기서 핵심적인 사실 하나를 짚어야 합니다. 검정통계량의 분포를 그려보면, 표본크기가 정해져 있는 한, 기각역(검정통계량이 어느 값을 넘으면 기각한다고 정한 경계)을 어느 쪽으로 옮기든 α 와 β 가 함께 줄어드는 일은 일어나지 않습니다. 기각역을 넓히면 α 는 커지고 β 는 작아지며, 기각역을 좁히면 그 반대가 됩니다.

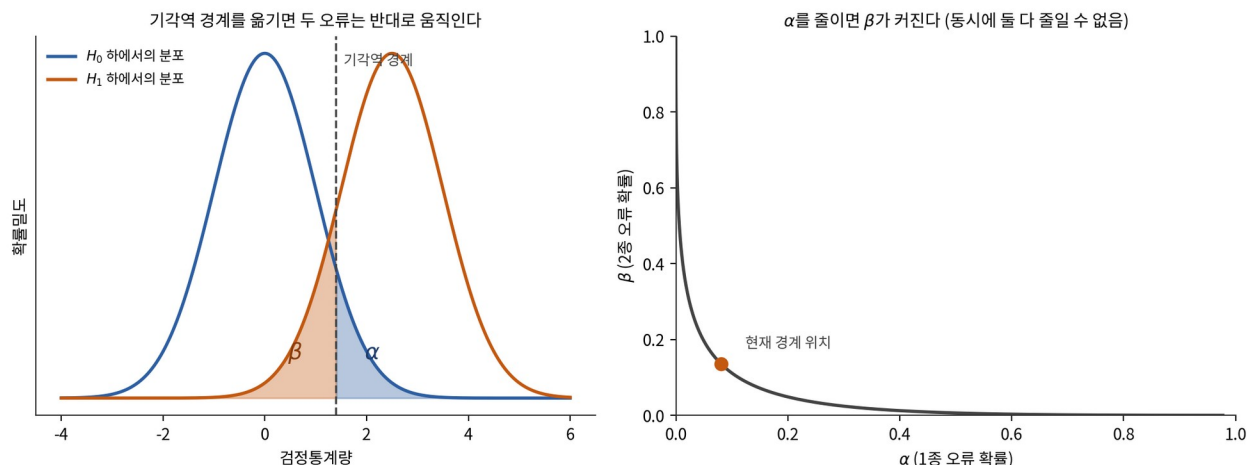


그림 1. 기각역의 위치에 따른 α 와 β 의 관계. 왼쪽: 귀무가설과 대립가설 하에서의 분포와 기각역. 오른쪽: α 를 줄이면 β 가 커지는 관계 곡선.

그래프 오른쪽이 보여주듯, α 를 0에 가깝게 만들면 β 는 1에 가까워지고, 그 반대도 마찬가지입니다. 두 오류를 동시에 0으로 만드는 기각역은 존재하지 않습니다. 두 오류를 함께 줄이는 방법은 표본을 늘리는 것뿐이며, 이는 7장에서 다룹니다.

그렇다면 어떻게 기각역을 정해야 할까요? 통계학에서 택한 해법은 비대칭적입니다. 두 오류 중 하나를 먼저 “감수할 수 있는 수준”으로 고정하고, 그 안에서 나머지 오류를 최대한 줄이는 방식입니다. 관행적으로 α 를 0.05와 같은 작은 값으로 먼저 정합니다. 이는 “멀쩡한 것을 잘못되었다고 판단하는 오류”를 더 심각하게 보고, 이 오류가 일어날 확률을 엄격하게 통제하겠다는 선택입니다.

α 를 고정한 다음에는 그 제약 안에서 β 를 최소화하는, 즉 검정력($1 - \beta$, 실제 효과가 있을 때 그것을 잡아낼 확률)을 최대화하는 기각역을 찾습니다. 이 기각역의 형태가 어떻게 결정되는지에 대한 수학적 이론이 있지만(네이만-피어슨 보조정리라고 부릅니다), 이 가이드에서는 그 결론만 기억하면 충분합니다. 즉 $\alpha = 0.05$ 라는 숫자 자체가 무슨 특별한 최적점이어서가 아니라, “먼저 허용 가능한 수준을 정하고, 그 안에서 최선을 다한다”는 절차의 결과로 그 숫자가 쓰이는 것입니다.

더 알아보기

이 절차의 수학적 근거를 “네이만-피어슨 보조정리(Neyman-Pearson Lemma)”라고 부릅니다. 관심 있는 독자는 이 키워드로 검색해 더 깊이 있는 내용을 찾아볼 수 있습니다.

2 부. 가설검정 수행하기

4 장. 검정통계량 만들기

이제 직접 검정을 수행하는 방법을 살펴봅니다. 가장 먼저 필요한 것은 표본으로부터 계산할 수 있는 하나의 숫자, 즉 검정통계량입니다.

예를 들어 모집단의 평균이 특정 값 μ_0 인지 검정하고 싶다고 합시다. 표본평균 $\bar{X}$ 는 모집단에서 표본을 뽑을 때마다 달라지는 확률변수이고, 그 기댓값과 분산은 다음과 같이 알려져 있습니다.

$$E(\bar{X}) = \mu, \quad \text{Var}(\bar{X}) = \sigma^2 / n$$

이 사실을 이용하면, 만약 귀무가설($\mu = \mu_0$)이 참이라면 $\bar{X}$ 가 μ_0 근처에서 표준편차 $\sigma/\sqrt{n}$ 정도로 흩어져 있어야 한다는 것을 알 수 있습니다. 이를 표준화한 것이 검정통계량입니다.

$$Z = (\bar{X} - \mu_0) / (\sigma / \sqrt{n})$$

귀무가설이 참이고, 모집단이 정규분포를 따르거나 표본이 충분히 크다면(중심극한정리) 이 Z는 표준정규분포를 따릅니다. 따라서 실제로 계산한 Z 값이 표준정규분포에서 나오기 매우 어려운(즉 절댓값이 매우 큰) 값이라면, “귀무가설이 참인데 우연히 이런 표본이 나왔다”고 보기보다는 “애초에 귀무가설이 틀렸다”고 의심하는 것이 합리적입니다. 이것이 가설검정의 기본 논리입니다.

5 장. 가설검정 5 단계 실전 절차

실제 가설검정은 다음의 다섯 단계를 거칩니다.

1. 가설 설정: 귀무가설 H_0 과 대립가설 H_1 을 정합니다.
2. 유의수준 결정: α 를 정합니다 (보통 0.05).
3. 검정통계량 계산: 표본 데이터로부터 Z (또는 다른 통계량)를 계산합니다.
4. 기각역 또는 p 값 판단: 계산된 통계량이 기각역에 들어가는지, 또는 p 값이 α 보다 작은지 확인합니다.
5. 결론: H_0 을 기각하거나 기각하지 않습니다.

양측검정은 “ μ 가 μ_0 과 다르다”처럼 방향을 정하지 않은 대립가설을, 단측검정은 “ μ 가 μ_0 보다 크다”처럼 방향을 정한 대립가설을 사용합니다. 양측검정은 분포의 양쪽 꼬리에, 단측검정은 한쪽 꼬리에만 기각역을 둡니다.

간단한 예제를 코드로 살펴보겠습니다. 어느 공정에서 만든 부품의 평균 길이가 50mm 여야 하는데, 표본 36 개를 측정해 평균 50.8mm, 모표준편차 2mm 로 알려져 있다고 합시다.

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

mu0, sigma, n = 50, 2, 36
xbar = 50.8
alpha = 0.05
```

```

z = (xbar - mu0) / (sigma / np.sqrt(n))
p_value = 2 * (1 - stats.norm.cdf(abs(z))) # 양측검정

print(f"Z 통계량: {z:.3f}")
print(f"p 값: {p_value:.4f}")
print("귀무가설 기각" if p_value < alpha else "귀무가설 기각 못함")

# 출력:
# Z 통계량: 2.400
# p 값: 0.0164
# 귀무가설 기각

```

6 장. p 값 제대로 이해하기

p 값은 “귀무가설이 참이라고 가정했을 때, 지금 관측한 결과(혹은 그보다 더 극단적인 결과)가 나올 확률”입니다.

흔한 오해

p 값은 “귀무가설이 참일 확률”이 아닙니다. p 값은 귀무가설이 참이라는 가정 “하에서” 계산되는 조건부 확률이지, 귀무가설 자체의 확률이 아닙니다.

p 값이 작다고 해서 효과의 크기가 큰 것도 아닙니다. p 값은 표본크기에 크게 영향을 받습니다 (7 장에서 다룹니다).

p 값과 α 를 비교하는 방식은 다음과 같습니다. p 값이 α 보다 작으면 귀무가설을 기각하고, 그렇지 않으면 기각하지 않습니다. 이것은 “검정통계량이 기각역 안에 들어가는가”를 확인하는 것과 동일한 결론을 줍니다.

p 값에 대해서는 흔한 오해가 세 가지 있습니다. 첫째, p 값은 “귀무가설이 참일 확률”이 아닙니다. 귀무가설이 참이라고 가정한 상태에서 계산한 확률입니다. 둘째, 귀무가설을 기각하지 못했다고 해서 귀무가설이 참이라는 뜻은 아닙니다. 차이를 입증할 증거가 충분하지 않다는 뜻일 뿐입니다. 셋째, p 값이 작다고 효과가 큰 것은 아닙니다. 표본이 매우 크면 실무적으로 의미 없는 작은 차이도 p 값이 아주 작게 나옵니다.

7 장. 검정력과 표본크기

검정력은 $1 - \beta$, 즉 실제로 효과가 있을 때 그것을 검정이 잡아낼 확률입니다. 검정력은 다음 세 가지 요인에 영향을 받습니다.

- 표본크기 n: n 이 커지면 $\bar{X}$ 의 분산(σ^2/n)이 작아져, 작은 차이도 더 뚜렷하게 감지할 수 있습니다.
- 효과크기: 귀무가설과 실제 값 사이의 차이를 표준편차로 나눈 값($d = (\mu_1 - \mu_0) / \sigma$)입니다. 효과크기가 클수록 검정력은 높아집니다.
- 유의수준 α : α 를 크게 잡을수록(기각역을 넓힐수록) 검정력도 높아지지만, 이는 3 장에서 본 trade-off 에 따라 1 종 오류 위험을 키우는 대가를 치릅니다.

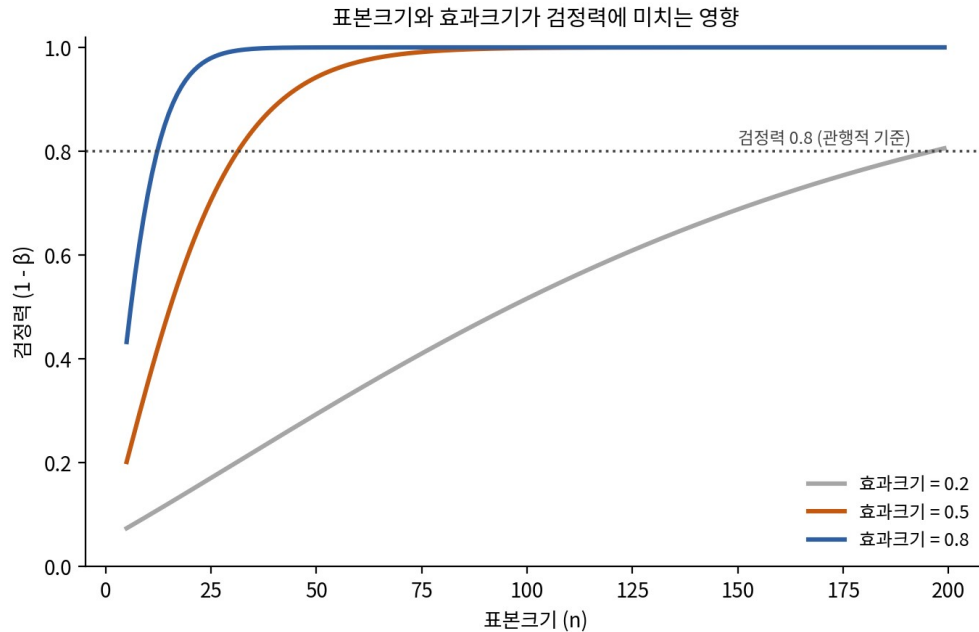


그림 2. 표본크기와 효과크기에 따른 검정력의 변화. 효과크기가 클수록, 표본크기가 클수록 검정력이 빠르게 1에 가까워진다.

이 그래프에서 주목할 점은, 효과크기가 작을 때(회색 선)는 검정력 0.8에 도달하기 위해 상당히 큰 표본크기가 필요하다는 것입니다. 이는 실무에서 “표본을 얼마나 모아야 하는가”를 결정할 때 중요한 정보입니다.

```
# Python: statsmodels로 검정력 곡선 그리기
from statsmodels.stats.power import NormalIndPower
import numpy as np

analysis = NormalIndPower() # 첫 인자=효과크기, ratio=0 → 1 표본 검정력
ns = np.arange(5, 200)
power_02 = [analysis.power(0.2, nobs1=n, alpha=0.05, ratio=0) for n in ns]
power_05 = [analysis.power(0.5, nobs1=n, alpha=0.05, ratio=0) for n in ns]
power_08 = [analysis.power(0.8, nobs1=n, alpha=0.05, ratio=0) for n in ns]

# 위 결과를 그림 2와 같이 그래프로 그리면 표본크기에 따른
# 검정력의 변화를 직접 확인할 수 있습니다.
```

8 장. 검정 방법은 상황에 따라 달라진다 — Z-test 와 t-test

4장에서 만든 Z 통계량은 모표준편차 σ 를 “안다”고 가정했습니다. 그런데 현실에서는 σ 를 정확히 아는 경우가 오히려 드뭅니다. 보통은 표본 자체로부터 표준편차를 추정한 s 를 대신 사용해야 합니다.

$$T = (\bar{X} - \mu_0) / (s / \sqrt{n})$$

σ 대신 s 를 쓰면 분포가 바뀝니다. s 자체도 표본에 따라 달라지는 확률변수이기 때문에, 그만큼 불확실성이 하나 더 늘어나는 셈입니다. 그 결과 T 는 (모집단이 정규분포를 따를 때) 표준정규분포가

아니라 자유도 $n-1$ 인 t 분포라는, 표준정규분포보다 꼬리가 더 두꺼운 분포를 따릅니다. 표본크기가 작을수록(자유도가 낮을수록) 이 차이는 더 두드러집니다.

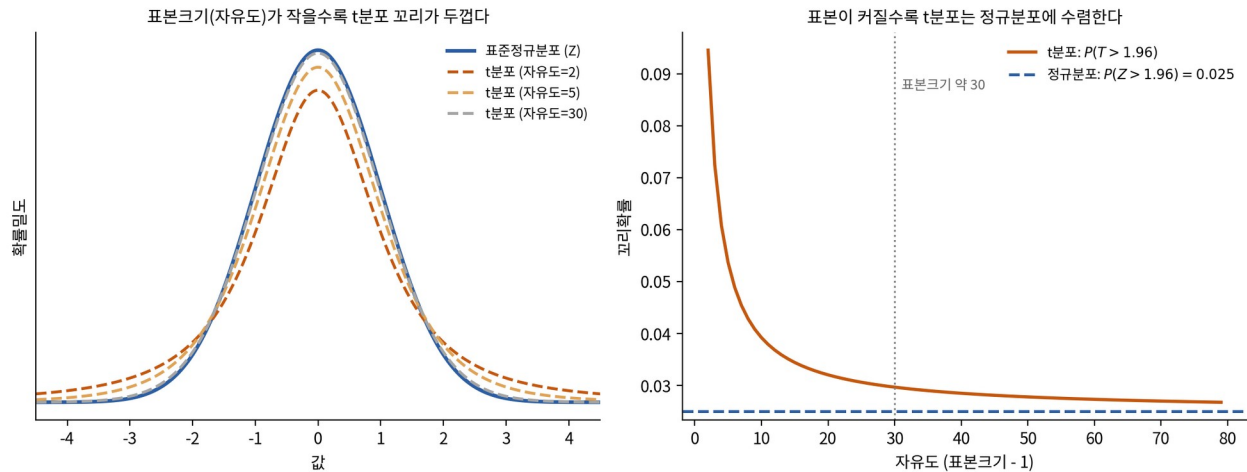


그림 3. 왼쪽: 자유도에 따른 t 분포와 표준정규분포의 비교. 자유도가 작을수록 꼬리가 두껍다. 오른쪽: 자유도가 커질수록 t 분포의 꼬리확률이 표준정규분포의 꼬리확률에 수렴한다.

그렇다면 한 가지 의문이 생깁니다. 현실에서 σ 를 아는 경우가 드물다면, Z-test 는 왜 배우는 걸까요?

답은 그림 3 의 오른쪽 그래프에 있습니다. t 분포는 표본크기(자유도)가 커질수록 표준정규분포에 가까워집니다. 즉 표본이 충분히 크면 s 를 σ 대신 써도 그 추정 오차가 무시할 만큼 작아져서, t-test 와 Z-test 의 결과가 사실상 같아집니다. 다시 말해 두 검정은 σ 를 아느냐 모르느냐로 구분되는 서로 다른 검정이지만, 표본이 충분히 크면 t-test 의 결과가 Z-test 로 근사되어 사실상 같아집니다.

실제로 통계 소프트웨어나 실무에서는 σ 를 정말로 아는 드문 경우를 제외하면 거의 항상 t-test 를 기본으로 사용합니다. Z-test 를 먼저 배우는 이유는, 더 단순한 가정에서 출발해 가설검정의 핵심 논리(표준화와 기각역)를 먼저 익히고, 그다음 그 가정을 현실에 맞게 하나씩 완화해 나가는 학습 순서 때문입니다.

참고

“표본크기가 30 이상이면 Z 와 t 가 거의 같다”는 표현을 종종 보게 되는데, 이는 어디까지나 경험적인 기준입니다. 모집단의 분포가 정규분포에서 많이 벗어나 있으면 30 보다 더 큰 표본이 필요할 수 있고, 반대로 모집단이 정규분포에 가까우면 더 작은 표본으로도 충분할 수 있습니다.

이 두 검정을 코드로 비교해 보겠습니다. 같은 데이터에 대해 σ 를 안다고 가정한 경우(Z-test)와 모른다고 가정한 경우(t-test)를 각각 수행합니다.

아래 예제에서는 두 검정의 결론이 같립니다. 그 원인은 표본크기($n=12$)가 작아서가 아니라, Z-test 에서 가정한 $\sigma=2$ 가 실제 데이터의 표본표준편차(약 0.54)보다 훨씬 크기 때문입니다. σ 를 잘못 알고 있으면 Z-test 의 결론이 뒤집힐 수 있다는 점, 그래서 σ 를 정말 아는 경우가 아니면 t-test 를 써야 한다는 점을 보여주는 예제입니다.

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

```

data = np.array([51.2, 49.8, 50.5, 50.9, 49.6, 50.7, 51.1, 50.3,
                 50.0, 50.6, 49.9, 51.0])
mu0 = 50
n = len(data)
xbar = data.mean()

# Z-test: 모표준편차를 2로 "안다고" 가정
sigma = 2
z = (xbar - mu0) / (sigma / np.sqrt(n))
p_z = 2 * (1 - stats.norm.cdf(abs(z)))

# t-test: 표본표준편차로 추정 (scipy 내장 함수 사용)
t_stat, p_t = stats.ttest_1samp(data, mu0)

print(f"Z-test: Z={z:.3f}, p={p_z:.4f}")
print(f"t-test: T={t_stat:.3f}, p={p_t:.4f}")

```

```

# R
data <- c(51.2, 49.8, 50.5, 50.9, 49.6, 50.7, 51.1, 50.3,
          50.0, 50.6, 49.9, 51.0)
mu0 <- 50

# t-test (R 기본 함수)
t.test(data, mu = mu0)

# Z-test는 R 기본 함수가 없어 직접 계산
sigma <- 2; n <- length(data); xbar <- mean(data)
z <- (xbar - mu0) / (sigma / sqrt(n))
p_z <- 2 * (1 - pnorm(abs(z)))
cat("Z =", z, " p =", p_z, "\n")

```

이 가이드는 가설검정의 가장 기초적인 논리와 절차만을 다루었습니다. 검정을 한 번 수행하는 법은 익혔지만, 검정을 여러 번 반복하거나 결과를 본 뒤 가설을 바꾸는 경우에는 별도의 주의가 필요합니다. 또한 Z-test 와 t-test 외에도 데이터의 형태와 가설의 구조에 따라 다양한 검정 방법이 존재하며, 이는 후속 가이드에서 다룰 예정입니다.