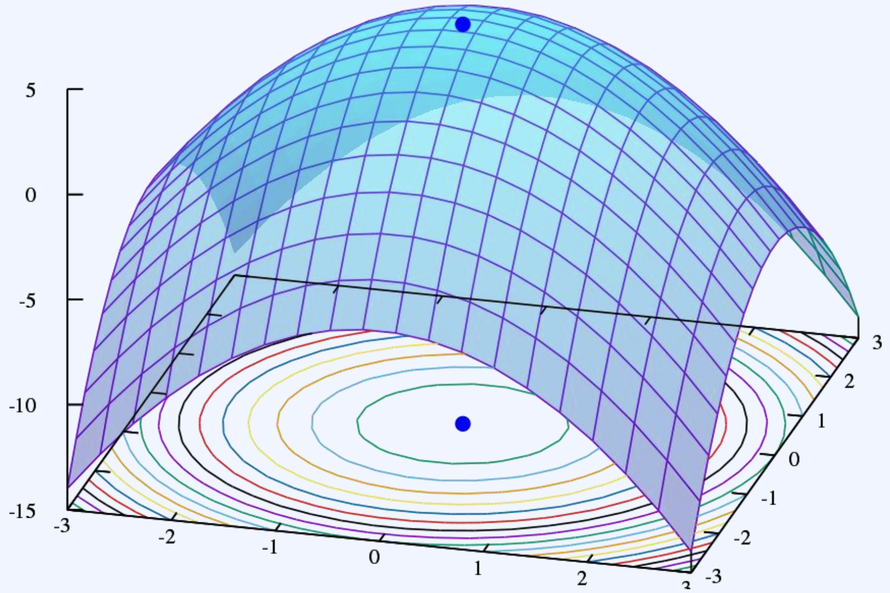
1. 최적화 모델

1 최적화 모델 개요

최적화 문제란?

최적화 문제(optimization problem)란 특정 제약조건하에 어떤 목적 함수를 최대화하거나 최소화하는 변수의 값을 찾는 문제입니다.



대부분의 머신러닝 모델 학습도 최적화 문제임

머신러닝 자동화의 핵심 문제인 모델 선택과 하이퍼파라미터 튜닝 문제 역시 최적화 문제

최적화 모델의 구성

최적화 모델은 목적 함수, 결정 변수, 제약식으로 구성됩니다.

$\text{minimize } f(\boldsymbol{x})$ — 목적식

$\text{subject to } \boldsymbol{x} \in \mathrm{S}$ — 제약식

S에 속하는 $\boldsymbol{x}$ 가운데 $f(\boldsymbol{x})$를 최소화하는 $\boldsymbol{x}$를 찾으라는 의미

→ $\boldsymbol{x}^* = \underset{\boldsymbol{x} \in \mathrm{S}}{\operatorname{argmin}} f(\boldsymbol{x})$

- f: 목적 함수
- $\boldsymbol{x}$: 결정 변수
- S: 실행 가능 공간 (탐색 공간)

최적화 모델 예제

maximize $2x_1 + x_2$

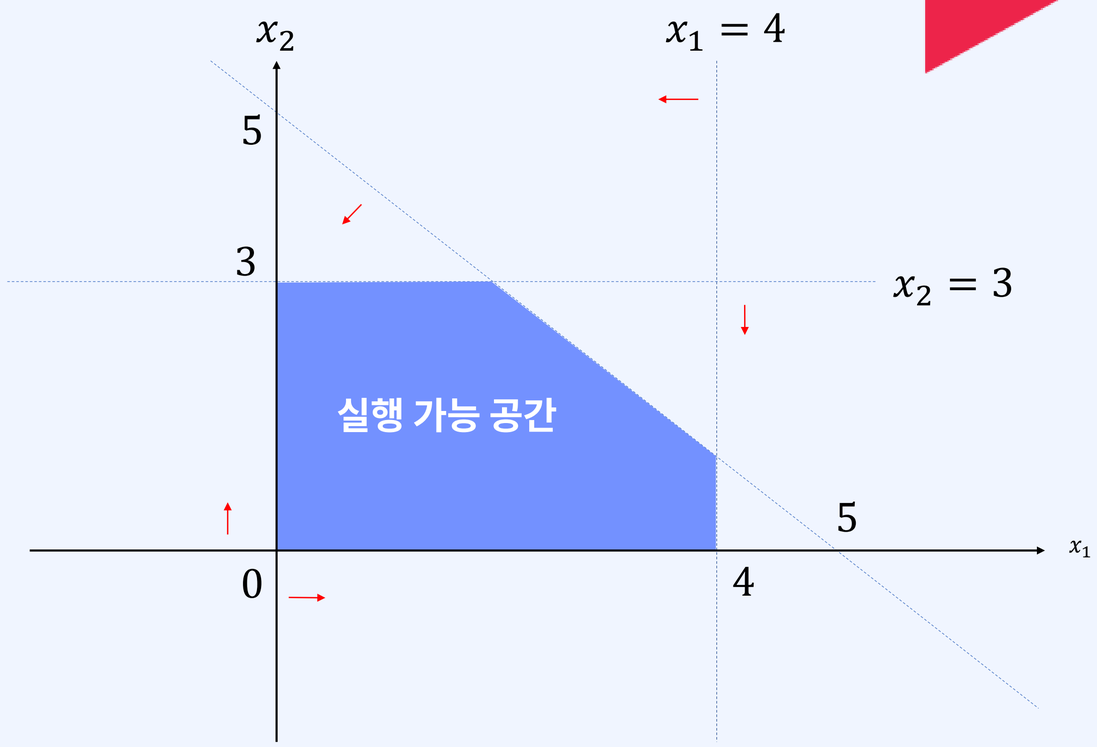
목적식

subject to $0 \leq x_1 \leq 4$

$0 \leq x_2 \leq 3$

$x_1 + x_2 \leq 5$

제약식

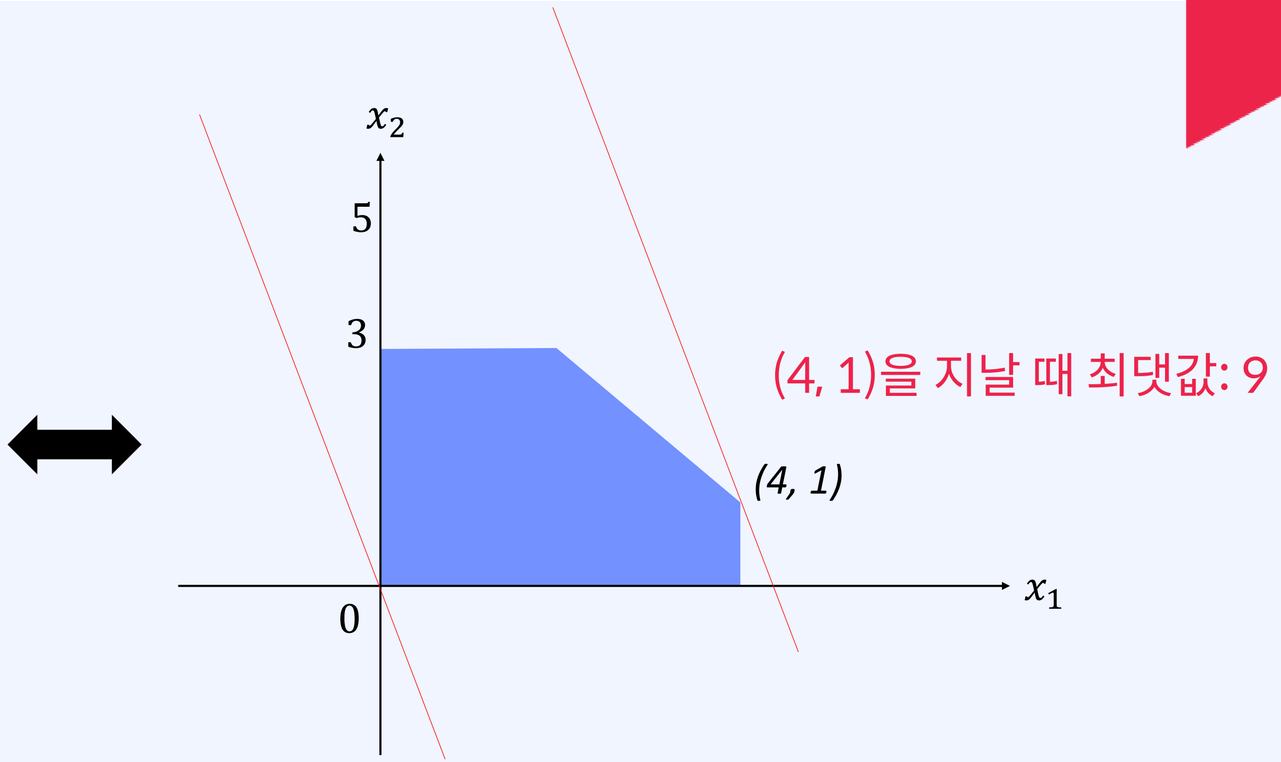


그래프 기반의 해법

목적 함수 정리: 기울기가 -2이고 절편이 z인 일차 함수

$z = 2x_1 + x_2$

$\rightarrow x_2 = -2x_1 + z$



그래프를 통해 실행 가능 공간을 찾고 그 가운데 목적 함수를 최대화하는 해를 손쉽게 찾음

BUT 그래프 기반의 해법은 최적화 문제 대부분에 적용할 수 없음

→
- 현실 문제를 최적화 모델로 표현하는 것이 중요
- 문제 유형에 맞는 적절한 해법이 무엇이고 어떻게 사용하는지 알아야 함

1. 최적화 모델

2 머신러닝 자동화를 위한 최적화

개요

머신러닝 자동화의 핵심은 사용자가 입력한 데이터에 적합한 전처리, 모델, 하이퍼파라미터 등을 빨리 찾는 것입니다.

예측 성능을 최대화하거나 예측 오차를 최소화하는 전처리, 모델, 하이퍼파라미터를 선택하는 최적화 문제를 해결하는 것이 머신러닝 자동화의 핵심

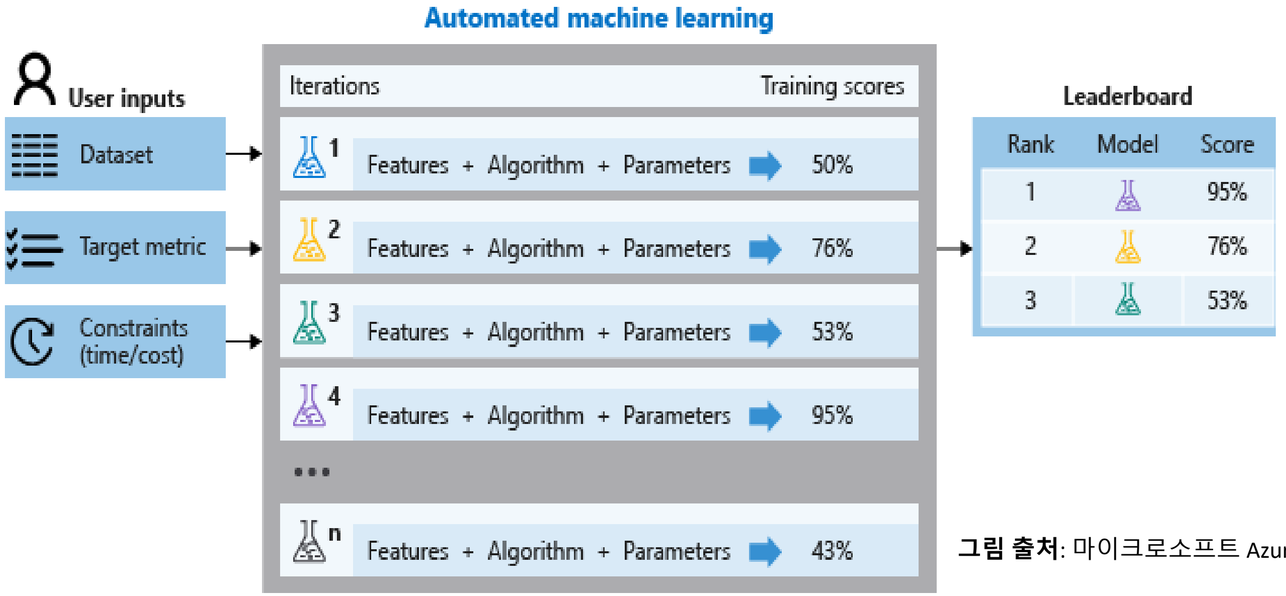


그림 출처: 마이크로소프트 Azure [1)]

관련 문제

- 하이퍼파라미터 튜닝 문제
- 모델 선택과 하이퍼파라미터 최적화 문제
- 파이프라인 생성 문제

머신러닝 자동화는 여러 실험(탐색)을 통해 최적의 전처리, 모델, 하이퍼 파라미터를 찾는 최적화 문제를 해결하는 것이라 할 수 있음

[1)] https://docs.microsoft.com/ko-kr/azure/machine-learning/concept-automated-ml

하이퍼파라미터 튜닝 문제 : 개요

하이퍼파라미터 튜닝 문제는 특정 데이터와 모델에 대해 성능을 최대화하는 모델의 하이퍼파라미터를 선택하는 문제입니다.

$$\boldsymbol{\lambda}^* = \underset{\boldsymbol{\lambda} \in \Lambda}{\operatorname{argmax}} P(\boldsymbol{\lambda}; D)$$

- $\boldsymbol{\lambda} = (\lambda_1, \lambda_2, \dots, \lambda_k)$: 하이퍼파라미터 벡터
- D: 사용자 입력 데이터
- $P(\boldsymbol{\lambda}; D)$: 하이퍼파라미터가 $\boldsymbol{\lambda}$인 모델을 데이터 D로 학습했을 때 모델의 성능
  (예시) 모델이 결정 나무이고 하이퍼파라미터가 깊이(λ_1)와 분지 기준(λ_2)이라면, $P(\lambda_1 = 5, \lambda_2 = Gini; D)$는 깊이가 5이고 분지 기준이 지니 계수인 결정 나무를 데이터 D로 학습했을 때의 성능
- Λ: 하이퍼 파라미터 $\boldsymbol{\lambda}$의 실행 가능 공간
  (예시) 결정 나무의 최대 깊이에 대한 실행 가능 공간은 자연수 집합이며, 라쏘의 계수 페널티에 대한 가중치 α는 0보다 크거나 같은 실수 집합

하이퍼파라미터 튜닝 문제
: 성능 척도

하이퍼파라미터 튜닝 문제의 목적 함수인 성능은 k-겹 교차 검증을 통해 주로 평가합니다.

$$P(\boldsymbol{\lambda}|D) = \frac{1}{k} \times \sum_{i=1}^{k} P\left(\boldsymbol{\lambda}; D^{(i)}, D^{(-i)}\right)$$

- $D^{(i)}$: $i(i = 1, 2, \cdots, k)$번째 폴드
- $D^{(-i)}$: D에서 i번째 폴드를 제외한 데이터, $D^{(-i)} = D - D^{(i)}$
- $P\left(\boldsymbol{\lambda}; D^{(i)}, D^{(-i)}\right)$: $D^{(i)}$를 평가 데이터로, $D^{(-i)}$를 학습 데이터로 사용했을 때의 성능

하이퍼파라미터 튜닝 문제
: 특징

하이퍼파라미터 튜닝 문제는 분석적으로 풀 수 없고 푸는데 시간이 오래 걸립니다.

분석적으로 해결할 수 없어, 여러 해를 평가하고 비교하여 좋은 해를 찾아야 합니다.

탐색 공간을 모두 탐색하는 것은 비현실적이고 비효율적임
(예: 라쏘의 하이퍼파라미터 α는 0 이상이기만 하면 되므로 선택할 수 있는 경우의 수가 무한대임)

하이퍼파라미터에 대한 실행 가능 공간은 사용자가 직접 정의하며, 이 과정에서 모델과 하이퍼파라미터에 대한 이해가 큰 역할을 함
(예 1: 라쏘의 하이퍼파라미터 α는 특정 수치 이상 설정할 필요가 없음)
(예 2: 결정 나무에서 불순도 지표는 탐색할 필요가 없음)

하나의 해를 평가하려면 모델을 학습해야 하므로 오랜 시간이 걸림. 따라서 탐색 공간을 잘 정의하는 것과 좋을 것이라 예상되는 해를 먼저 탐색하는 것이 매우 중요함

모델 선택과 하이퍼파라미터 최적화 문제

모델 선택과 하이퍼파라미터 최적화 문제는 하이퍼파라미터 튜닝 문제에 모델 선택을 추가한 문제입니다.

$$\mathcal{M}^*, \boldsymbol{\lambda}^* = \underset{\mathcal{M} \in \Omega, \boldsymbol{\lambda} \in \Lambda}{\operatorname{argmax}} P(\mathcal{M}, \boldsymbol{\lambda}; D)$$

- Ω: 선택 가능한 모델 집합
- Λ: 선택 가능한 하이퍼파라미터 집합
- 하이퍼파라미터 튜닝 문제의 탐색 공간보다 모델 선택과 하이퍼파라미터 최적화 문제의 탐색 공간이 더 넓고 복잡함
- 결정 변수 $\mathcal{M}$에 따라 다른 결정 변수 $\boldsymbol{\lambda}$의 탐색 공간이 달라짐
  (예시) $\mathcal{M}$이 신경망이라면 $\boldsymbol{\lambda}$는 은닉층의 구조 등을 탐색해야 하지만, $\mathcal{M}$이 결정 나무라면 $\boldsymbol{\lambda}$는 최대 깊이 등을 탐색해야 함

파이프라인 생성 문제

파이프라인 생성 문제는 데이터가 주어졌을 때 가장 적절한 머신러닝 파이프라인을 찾는 문제로, 여기에는 특징 공학과 스케일링 등의 전처리도 포함됩니다.

$$G^* = \underset{G}{\operatorname{argmax}}\, P(G; D)$$

- G: 머신러닝 파이프라인

- 머신러닝 자동화에 필요한 최적화 문제의 일반적인 형태의 문제임
- 탐색할 수 있는 파이프라인 구조는 무한하기에 파이프라인 생성 문제의 탐색 공간은 선택과 하이퍼파라미터 최적화 문제보다도 훨씬 넓음
- 사용자가 직접 보통 1~3 개 정도의 파이프라인을 미리 정의해서 사용함

파이프라인 생성 문제
: 해 구조

파이프라인 생성 문제의 해인 파이프라인은 순환하지 않는 비방향성 그래프(directed acyclic graph; DAG)로 정의할 수 있으며, 이 그래프의 각 노드에는 전처리, 모델, 하이퍼파라미터 등이 대응됩니다.

$$G = \{(\mathcal{M}^{(1)}, \boldsymbol{\lambda}^{(1)}), (\mathcal{M}^{(2)}, \boldsymbol{\lambda}^{(2)}), \cdots, (\mathcal{M}^{(n)}, \boldsymbol{\lambda}^{(n)})\}$$

- $\mathcal{M}^{(i)}$: i번째 노드의 전처리 및 지도 학습 모델
- $\boldsymbol{\lambda}^{(i)}$: $\mathcal{M}^{(i)}$의 하이퍼파라미터

해 예시

$\mathcal{M}$	min-max 정규화	필터링	결정 나무
$\boldsymbol{\lambda}$	없음	• 클래스 관련성 척도: 상호 정보량 • 선택하는 특징 개수: 20개	• 깊이: 5 • 분지 기준: 엔트로피

문제의 특징

머신러닝 자동화를 위한 최적화 문제를 해결하기 어려운 이유는 넓은 탐색 공간, 결정 변수 간 의존성, 긴 해 평가 시간, 블랙박스로 요약할 수 있습니다.

구분	설명
넓은 탐색 공간	• 탐색 공간을 구성하는 변수가 여러 개이고 변수마다 탐색 범위가 넓음
결정 변수 간 의존성	• 사용하는 모델에 따라 선택할 수 있는 하이퍼파라미터가 다름 • 몇몇 하이퍼파라미터는 다른 하이퍼파라미터에 의해 사용 여부가 결정됨
긴 해 평가 시간	• k-겹 교차 검증 등으로 모델 학습과 평가를 거쳐서 해를 평가하므로 시간이 오래 걸림
블랙박스	• 데이터에 따라 모델과 하이퍼파라미터의 성능이 천차만별임

- 머신러닝 자동화에서 고려하는 최적화 문제는 분석적으로 풀 수 없어 이 문제를 풀기 위한 다양한 알고리즘이 개발되고 있음
- 어떠한 알고리즘도 최적해를 찾을 가능성은 매우 희박하며, 심지어 최적해를 찾더라도 이 해가 실제 최적해인지조차 알 수 없음