▣ **강의명:** CS109A: 데이터 과학 입문

▣ **주차:** Lecture 23

▣ **교수명:** Pavlos Protopapas, Kevin Rader, Chris Gumb

▣ **목적:** Lecture 23의 핵심 개념 학습

Contents

부스팅(Boosting)과 그레디언트 부스팅(Gradient Boosting)

약한 모델들을 모아 강력한 모델을 만드는 앙상블 기법 완전 정복

□ 핵심 요약

문서 핵심 요약 이 문서는 머신러닝의 강력한 앙상블 기법인 '부스팅(Boosting)'의 개념부터 '그레디언트 부스팅(Gradient Boosting)'의 수학적 원리까지 다룹니다.

- **핵심 아이디어:** 단순한 모델(Weak Learner)을 순차적으로 연결하여, 앞선 모델의 실수를 뒤따르는 모델이 바로잡습니다.
- **목표:** 편향(Bias)을 줄여 예측 성능을 극대화합니다. (랜덤 포레스트가 분산(Variance)을 줄이는 것과 대조적)
- **원리:** 잔차(Residual)를 새로운 모델의 목표값(Target)으로 학습하며, 이는 수학적으로 경사 하강법(Gradient Descent)과 동일합니다.
- **실전:** 학습률(Learning Rate) 조정을 통해 과적합(Overfitting)을 방지하는 것이 중요합니다.

1 필수 용어 정리

본격적인 학습에 앞서, 아래 용어들을 먼저 숙지하면 이해가 훨씬 수월합니다.

용어 (한글)	원어 (English)	쉬운 설명	비고
약한 학습기	Weak Learner	성능이 턱걸이 수준인 단순한 모델 (예: 깊이가 1인 트리)	부스팅의 재료
강한 학습기	Strong Learner	약한 학습기를 모아 만든 고성능 모델	최종 결과물
잔차	Residual	정답과 내 예측값의 차이 $(y - \hat{y})$, 즉 '오차'	다음 모델의 목표
앙상블	Ensemble	여러 모델을 조합하여 성능을 높이는 기법	협동 작업
스텀프	Stump	가지가 딱 한 번만 뻗은 아주 얕은 결정 트리	깊이(Depth)=1
학습률	Learning Rate	모델을 업데이트할 때 얼마나 반영할지 결정하는 보폭 (λ)	과적합 방지용

Table 1: 부스팅 학습을 위한 필수 용어 사전

2 부스팅(Boosting)이란 무엇인가?

2.1 왜 필요한가? (배경)

우리는 이전에 **랜덤 포레스트(Random Forest)**와 **배깅(Bagging)**을 배웠습니다. 이들은 깊고 복잡한 트리(Overfitting 되기 쉬움)를 여러 개 만들어 평균을 냄으로써 **분산(Variance)**을 줄이는 전략을 썼습니다.

하지만 **부스팅**은 정반대의 접근을 취합니다.

- 아주 단순하고 멍청한 모델(High Bias, Low Variance)에서 시작합니다.
- 이 모델의 **실수(편향, Bias)**를 줄이는 방향으로 모델을 계속 추가합니다.
- 결과적으로 **편향을 획기적으로 줄여** 강력한 예측 모델을 만듭니다.

직관적 비유: 어려운 시험 통과하기 여러분이 통과하기 매우 어려운 시험을 앞두고 있다고 상상해 봅시다. 한 명의 천재가 문제를 다 푸는 것이 아니라, 평범한 학생들의 지혜를 모으는 방식입니다.

1. **학생 1 (단순한 규칙):** "일단 기출문제를 보니, 4번은 정답이 아니야." → 정확도 60%
2. **학생 2 (실수 보완):** "학생 1이 틀린 문제들을 보니, 보기에 '과적합'이라는 단어가 있으면 그게 정답이더라." → 정확도 65%
3. **학생 3 (추가 보완):** "학생 1, 2가 틀린 걸 보니, '교차 검증'이 답인 경우가 많아." → 정확도 70%

이 세 학생의 의견을 적절히 합치면(가중치 부여), 혼자서는 풀 수 없던 문제를 90점 이상으로 통과할 수 있습니다. 이것이 바로 **부스팅**입니다.

2.2 부스팅의 핵심 메커니즘

부스팅은 **순차적(Iterative)**이고 **덧셈적(Additive)**인 과정입니다.

$$T_{final}(x) = \sum_{h \in H} \lambda_h T_h(x)$$

- T_h: 약한 학습기 (개별 학생의 의견)
- λ_h: 가중치 (그 학생의 의견을 얼마나 신뢰할 것인가)
- T_{final}: 최종 모델 (강한 학습기)

중요한 점: 한 번에 모든 모델을 만드는 것이 아니라, **앞선 모델이 저지른 실수를 다음 모델이 해결**하도록 순서대로 만듭니다.

3 그레디언트 부스팅 (Gradient Boosting)

부스팅 중에서도 가장 널리 쓰이고 강력한 방법이 **그레디언트 부스팅**입니다. 이름이 어렵게 들리지만, 원리는 간단합니다. **"이전 모델의 오차(잔차)를 새로운 모델이 학습한다"**는 것입니다.

3.1 핵심 원리: 잔차(Residual) 학습하기

비유: 골프 퍼팅 목표 지점(홀컵)까지 공을 보내야 합니다.

1. **첫 번째 샷 (T_0):** 공을 쳤는데 홀컵까지 10m가 남았습니다. (잔차 = 10m)
2. **두 번째 샷 (T_1):** 이제 목표는 홀컵이 아니라, '남은 거리 10m'를 보내는 것입니다. 쳤는데 2m가 남았습니다. (잔차 = 2m)
3. **세 번째 샷 (T_2):** 이제 목표는 '남은 거리 2m'입니다.

이렇게 계속 남은 거리(잔차)를 메우는 샷을 더해나가는 것이 그레디언트 부스팅입니다.

3.2 동작 알고리즘 (단계별 설명)

데이터가 입력값 x와 정답 y로 구성되어 있다고 합시다.

Step 1: 아주 단순한 첫 번째 모델(T_0)을 만듭니다.

- 예: 그냥 전체 데이터의 평균값으로 예측합니다. 당연히 많이 틀립니다.

Step 2: 잔차(Residual)를 계산합니다.

- $r_0 = y - T_0(x)$
- 즉, (실제 정답) - (첫 번째 모델의 예측값) = (아직 맞추지 못한 남은 오차)입니다.

Step 3: 잔차를 예측하는 두 번째 모델(T_1)을 만듭니다.

- 이번에는 y를 맞추는 게 아니라, r_0(오차)를 맞추도록 학습합니다.
- 즉, "얼마나 더 더해야 정답에 가까워지는가?"를 배웁니다.

Step 4: 모델을 업데이트합니다.

- $T_{new} = T_0 + \lambda \times T_1$
- 여기서 λ(람다)는 **학습률(Learning Rate)**입니다. 두 번째 모델의 의견을 100% 반영하지 않고, 조금만 반영하여 과적합을 막습니다.

Step 5: 반복합니다.

- 다시 새로운 잔차를 계산하고, 그 잔차를 맞추는 모델 $T_2, T_3, \ldots$를 계속 더해 나갑니다.

주의: 실제로는 거대한 나무 하나를 만드는 게 아니다 "트리를 합친다"고 해서 실제로 노드와 가지를 물리적으로 합쳐서 거대한 트리 하나를 만드는 것이 아닙니다. 추론(예측) 시에는 T_0의 출력값, T_1의 출력값, ... T_n의 출력값을 모두 계산한 뒤 **숫자들을 더해서** 최종 값을 냅니다.

4 시각적 예시와 흐름

그레디언트 부스팅이 데이터를 어떻게 학습하는지 1차원 데이터 예시로 살펴봅니다.

4.1 데이터 상황

- x: 입력 변수 (0 8 사이의 값)
- y: 출력 변수 (구불구불한 곡선 형태의 데이터)

4.2 학습 과정 시각화

단계	모델의 행동	결과 해석
Round 1	단순한 스텀프(Stump) 하나로 전체 평균을 예측 (T_0)	데이터의 복잡한 패턴을 전혀 못 잡음. 잔차(오차)가 매우 큼.
Round 2	Round 1의 잔차(실제값 - 예측값)를 목표로 하는 새 스텀프 학습 (T_1)	큰 오차가 있던 부분을 보정함. 전체 모델 모양이 데이터에 약간 가까워짐.
Round 3	현재까지 모델($T_0 + \lambda T_1$)의 잔차를 다시 계산해 T_2 학습	세밀한 굴곡을 맞추기 시작함. 오차가 점점 0에 가까워짐.
...	**반복 (Iteration)**	**점점 정밀한 모델 완성**

Table 2: 그레디언트 부스팅의 단계별 학습 흐름

4.3 수식 요약

$$\text{최종 예측 } \hat{y} = T_0(x) + \lambda T_1(x) + \lambda T_2(x) + \cdots + \lambda T_N(x)$$

- 각 T_i는 이전 단계까지 해결하지 못한 '나머지 오차'를 해결하는 전문가입니다.
- λ(학습률)가 작을수록, 더 많은 트리가 필요하지만 더 정교하고 안정적인 모델이 됩니다.

5 심화: 왜 '그레디언트(Gradient)' 부스팅인가?

"잔차를 학습하는 것"이 왜 "경사 하강법(Gradient Descent)"과 같은지 이해하면, 이 알고리즘의 본질을 깨닫게 됩니다.

5.1 경사 하강법 (Gradient Descent) 복습

우리가 산 정상에서 가장 낮은 계곡(손실 함수 L의 최소값)으로 내려가려 합니다.

- 눈을 가리고 있다면, 발끝으로 경사를 느낍니다.
- 경사가 가장 가파르게 올라가는 방향(Gradient, ∇L)의 **반대 방향**으로 발을 내디뎌야 내려갈 수 있습니다.
- 공식: $w_{new} = w_{old} - \lambda \times \nabla L$ (파라미터 업데이트)

5.2 함수 공간(Function Space)에서의 하강

일반적인 머신러닝은 파라미터(w, 가중치)를 수정합니다. 하지만 부스팅은 **함수(모델의 예측값 $\hat{y}$) 자체**를 수정하여 정답에 다가갑니다.

우리가 최소화하고 싶은 손실 함수가 평균 제곱 오차(MSE)라고 가정해 봅시다.

$$L(y, \hat{y}) = \frac{1}{2}(y - \hat{y})^2$$

이 손실 함수를 예측값 $\hat{y}$에 대해 미분해 봅니다(경사를 구합니다).

$$\frac{\partial L}{\partial \hat{y}} = -(y - \hat{y})$$

어라? 결과가 익숙합니다.

- $y - \hat{y}$는 바로 **잔차(Residual)**입니다.
- 즉, **-(잔차)**가 곧 **기울기(Gradient)**입니다.
- 경사 하강법은 기울기의 '반대 방향'으로 가는 것이므로, **-(-잔차) = 잔차 방향**으로 이동하면 됩니다.

□ 핵심 요약

핵심 결론 **"잔차를 더해주는 것"**은 수학적으로 **"손실 함수의 기울기(Gradient) 반대 방향으로 이동하여 에러를 줄이는 것"**과 완벽하게 동일합니다. 따라서 이 방법을 **그레디언트 부스팅**이라고 부릅니다.

6 실전 가이드: 모델 튜닝하기

그레디언트 부스팅을 실제로 사용할 때 가장 중요한 두 가지 설정(Hyperparameter)이 있습니다.

6.1 1. 학습률 (Learning Rate, λ)

- **정의:** 새로 추가되는 트리의 의견을 얼마나 반영할지 결정하는 비율 (보통 0.01 0.1 사이).
- **λ가 너무 클 때:** 학습이 빠르지만, 최적점을 지나치거나(Overshooting) 과적합될 위험이 큽니다. (성큼성큼 걷다가 구덩이에 빠짐)
- **λ가 너무 작을 때:** 학습이 매우 느리고 많은 트리가 필요하지만, 일반화 성능이 좋습니다. (아주 조심스럽게 발을 내딛음)

6.2 2. 반복 횟수 (Number of Iterations)

- 트리를 몇 개나 더할 것인가를 결정합니다.
- 너무 많으면 훈련 데이터에 과도하게 맞춰져(Overfitting) 테스트 성능이 떨어질 수 있습니다.
- **조기 종료(Early Stopping):** 검증 데이터(Validation Set)의 성능이 더 이상 좋아지지 않으면 학습을 멈추는 기법을 주로 사용합니다.

6.3 튜닝 전략

보통 **학습률을 낮게($\lambda \downarrow$)** 설정하고, **트리 개수를 늘리는($N \uparrow$)** 방식이 성능이 가장 좋습니다. 다만 계산 시간이 오래 걸린다는 단점이 있습니다.

7 자주 묻는 질문 (FAQ) 및 오해 바로잡기

Q1. 랜덤 포레스트와 부스팅 중 뭐가 더 좋나요?

A. 정답은 "데이터에 따라 다르다"입니다.

- **랜덤 포레스트:** 병렬 학습이 가능해 빠르고, 튜닝 없이도 무난하게 좋은 성능을 냅니다. (Overfitting에 강함)
- **그레디언트 부스팅:** 순차 학습이라 느리고 튜닝이 까다롭지만, 잘 튜닝하면 일반적으로 랜덤 포레스트보다 **더 높은 정확도**를 냅니다. (Kaggle 대회 우승 알고리즘의 주역)

테이블 형태(Tabular) 데이터에서는 두 모델 모두 딥러닝보다 더 좋은 성능을 낼 때가 많습니다.

Q2. 'Inference'란 무슨 뜻인가요?

A. 문맥에 따라 다릅니다.

- **통계학:** 데이터로부터 모집단의 특성을 추론하고 가설을 검정하는 것 (예: p-value 구하기).
- **머신러닝/공학:** 학습된 모델에 새로운 데이터를 넣어 **예측(Prediction)** 값을 뽑아내는 과정.

부스팅 강의나 자료에서 "Inference가 느리다"고 하면 "예측값을 계산하는 데 시간이 걸린다"는 뜻입니다.

8 학습 마무리 체크리스트

이 문서를 다 읽은 후, 아래 질문에 스스로 답할 수 있다면 완벽하게 이해한 것입니다.

- ☐ **개념:** 부스팅이 배깅(랜덤 포레스트)과 근본적으로 어떻게 다른지 설명할 수 있는가? (힌트: 병렬 vs 순차, 분산 감소 vs 편향 감소)
- ☐ **직관:** "약한 학습기"와 "강한 학습기"의 관계를 비유를 들어 설명할 수 있는가?
- ☐ **알고리즘:** 그레디언트 부스팅이 다음 트리를 만들 때 사용하는 '타겟 값'이 무엇인가? (힌트: 잔차)
- ☐ **수학:** 잔차(Residual)가 왜 손실 함수의 기울기(Gradient)와 관련이 있는지 설명할 수 있는가?
- ☐ **실전:** 학습률(λ)이 높을 때와 낮을 때의 장단점을 알고 있는가?

"여러 개의 멍청한 모델이 힘을 합치면, 하나의 천재 모델보다 똑똑할 수 있다."
– **부스팅의 철학**