모델 선택

I. 의사 결정 트리의 가지치기

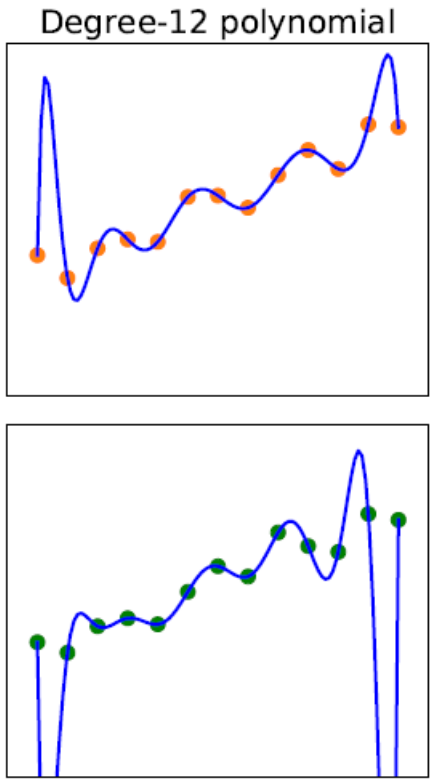
II. 모델 선택

III. 손실 함수

IV. 정규화(Regularization)

- *"Normalization" (정규화): 데이터의 스케일을 조정하여 다른 데이터와 비교 가능하게 만드는 과정.*
- *"Regularization" (정규화): 모델의 복잡성을 제어하여 과적합을 방지하거나 효율적인 학습을 돕는 방법.*

I. 과대적합 이슈



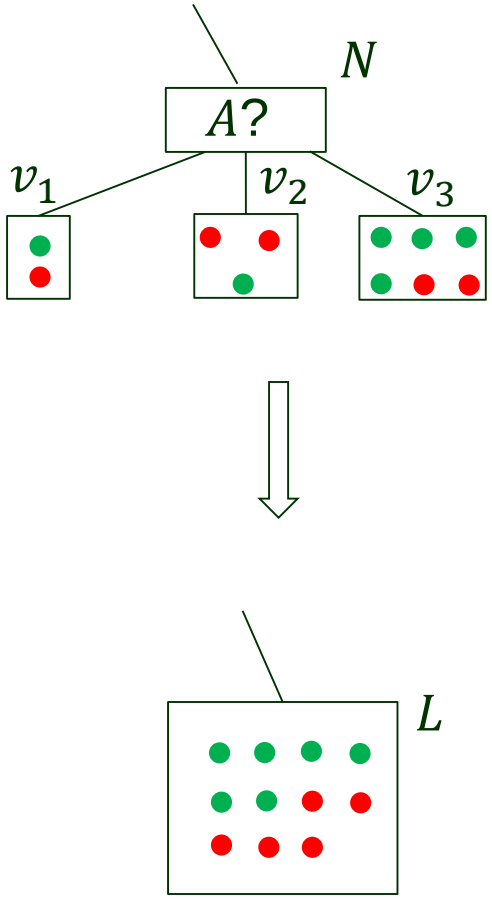
과대적합

- ♦ 훈련 데이터에 너무 많은 주의를 기울여 보지 않은 데이터에 대해서는 성능이 저조
- ♦ 속성의 수가 증가할수록 발생 가능성이 높아짐
- ♦ 훈련 예제가 많아질수록 발생 가능성이 낮아짐

♠ 차수가 높은 다항식일수록 과대적합 가능성이 높음

♠ 노드 수가 많은 의사결정 트리는 과적합 가능성이 더 높음

의사결정트리 가지치기



- 전체 의사결정 트리 구성
- 다음 과정 반복:
  - ♦ 자식 노드로 리프 노드만 갖는 테스트 노드 N을 고려
  - ♦ 테스트가 *관련이 없어 보이면* 노드를 리프 노드 L로 대체
  - ♣ 속성 A가 관련이 없음을 결정하는 방법

정보 이득이 낮은 지 체크

$$Gain(A) = B\left(\frac{p}{p+n}\right) - \sum_{k=1}^{d} \frac{p_k + n_k}{p+n} B\left(\frac{p_k}{p_k + n_k}\right)$$

유의성 검정

유의성 검정:*표본 자료를 이용해 모집단 값에 대한 주장을 평가하는 과정*

- 노드 N 은 p 개의 긍정적 예시와 n 개의 부정적 예시를 가지고 있음
- 노드 N 에서 속성 A를 테스트하기 위해 집합을 d 개의 하위 집합으로 분할
- $1 \le k \le d$ 인 하위 는 집합 k는 p_k 개의 긍정적 예시와 n_k 개의 부정적 예시를 가짐
- 하위 집합 k 에서 예상되는 긍정적 예시와 부정적 예시의 예측치는 다음 식으로 구함

$$\hat{p}_k = p \cdot \frac{p_k + n_k}{p + n} \qquad \hat{n}_k = n \cdot \frac{p_k + n_k}{p + n}$$

- 전체 편차의 측정치는 다음 식으로 주어짐

$\chi^2_{0.05,3} = 7.815$

$$\Delta = \sum_{k=1}^{d} \left(\frac{(p_k - \hat{p}_k)^2}{\hat{p}_k} + \frac{(n_k - \hat{n}_k)^2}{\hat{n}_k} \right)$$

자유도가 $d-1$ 인 χ^2(카이 제곱) 분포

χ^2 가지치기 적용:

Δ<7.82의 경우, 5% 수준에서 속성이 관련성이 없다는 것을 의미. Δ 값이 높을수록 속성이 더 중요하다는 것을 나타냄

II. 모델 선택

Goal: 미래에 마주치게 될 샘플에 *최적 적합화*된 가설을 선택

- 미래의 예시들은 과거와 비슷하다고 가정

Stationarity(정상성, or 안정성):

$$\boldsymbol{P}(E_j) = \boldsymbol{P}(E_{j+1}) = \boldsymbol{P}(E_{j+2}) = \cdots$$ // 모든 샘플의 사전 확률이 동일

$$\boldsymbol{P}(E_j) = \boldsymbol{P}(E_j \mid E_{j-1}, E_{j-2}, \ldots)$$ // 각 샘플은 이전 샘플에 독립

최적 적합화

전체 샘플을 3개 세트로 분리:

- *학습 세트*: 후보 모델(가설) 학습용
- *검증 세트*: 후보 모델 평가 및 최적 모델 선정
- *테스트 세트*: 최적 모델에 대한 최종 불편향 평가

2가지 작업:

- *모델 선택:* 적합한 가설 공간 선택
- *최적화* (aka *학습*): 해당 공간에서 최적의 가설 선택

모델 선택 vs. 교차 검증

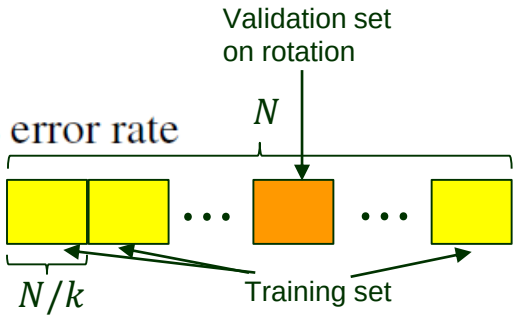
function MODEL-SELECTION(*Learner*, *examples*, *k*) **returns** a (hypothesis, error rate) pair

err ← an array, indexed by *size*, storing validation-set error rates
training_set, *test_set* ← a partition of *examples* into two sets
for *size* = 1 **to** ∞ **do**
  err[*size*] ← CROSS-VALIDATION(*Learner*, *size*, *training_set*, *k*)
  if *err* is starting to increase significantly **then**
    best_size ← the value of *size* with minimum *err*[*size*]
    h ← *Learner*(*best_size*, *training_set*)
    return *h*, ERROR-RATE(*h*, *test_set*)

// size는 모델 클래스의 하이퍼파라미터. e.g., 의사 결정 트리의 노드 수, 다항식의 차수 등

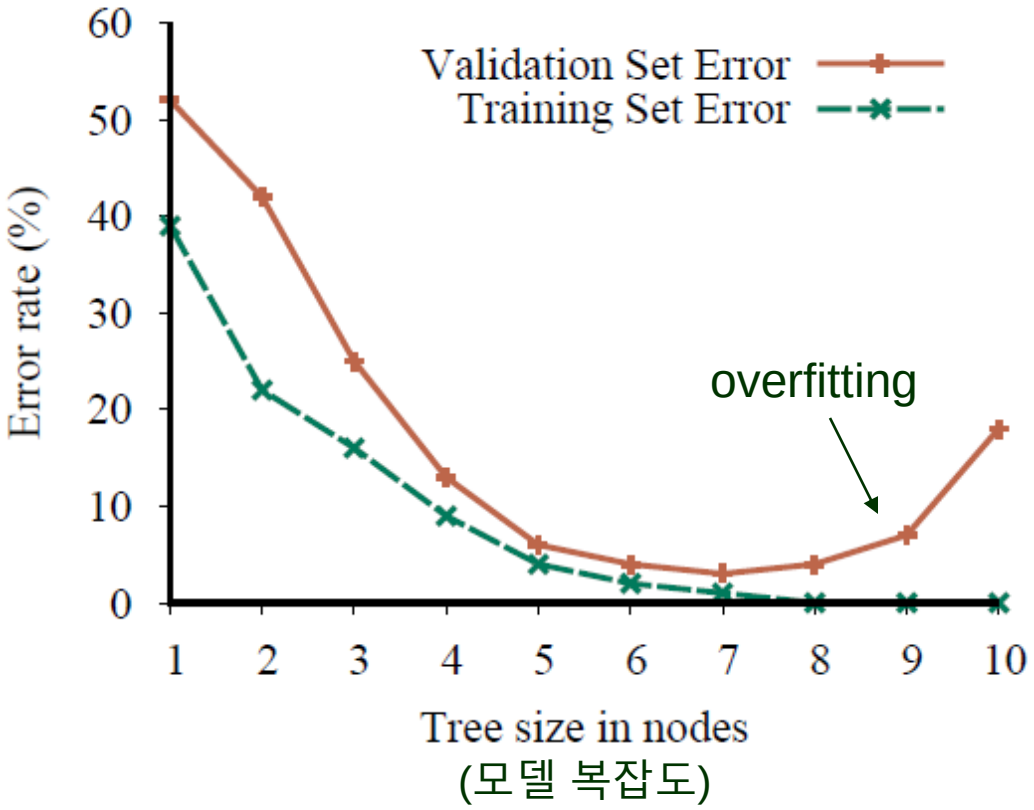
function CROSS-VALIDATION(*Learner*, *size*, *examples*, *k*) **returns** error rate

N ← the number of *examples*
errs ← 0
for *i* = 1 **to** *k* **do**
  validation_set ← *examples*[(*i* − 1) × *N*/*k*:*i* × *N*/*k*]
  training_set ← *examples* − *validation_set*
  h ← *Learner*(*size*, *training_set*)
  errs ← *errs* + ERROR-RATE(*h*, *validation_set*)
return *errs* / *k* // *average error rate on validation sets, across k-fold cross-validation*



// 전체 샘플을 균등한 크기로 나눈 k개의 하위 집합을 하나씩 선택하여 검증 세트로 설정. k 값이 커지면 과적합으로 인해 오차도 커짐

학습 오차와 검증 오차 (1)



데이터: 식당 자리 대기 문제에서 얻은 데이터

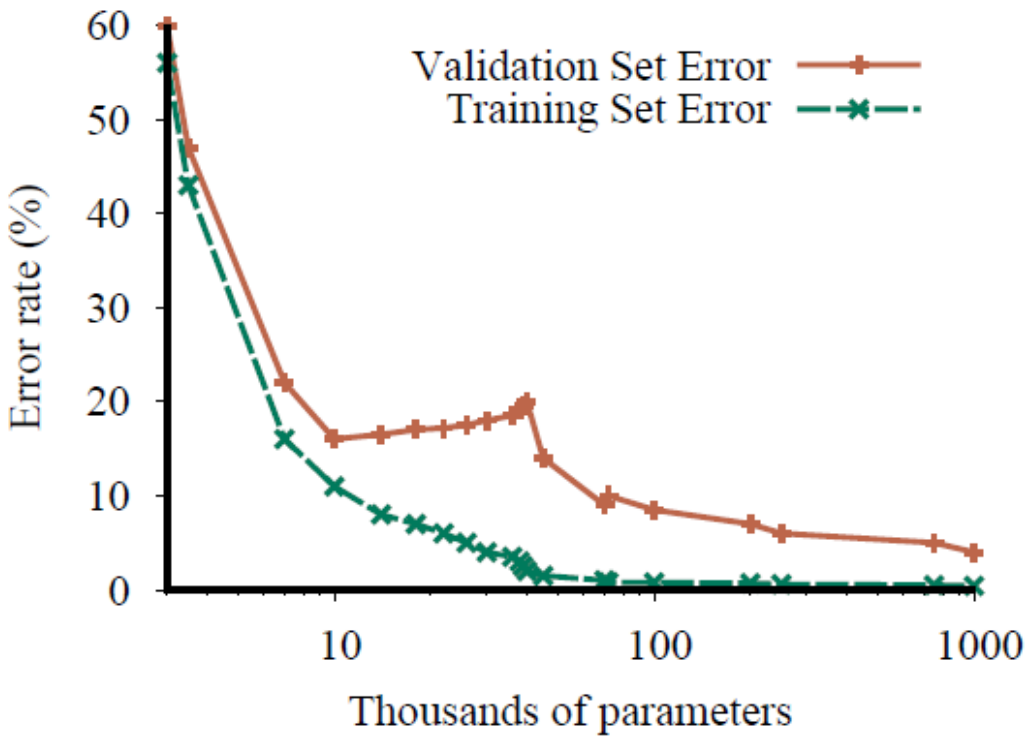
모델 클래스: 의사 결정 트리

하이퍼파라미터: 노드 개수

♦ 학습 세트의 오차는 모델 복잡도(트리 크기)가 증가함에 따라 단조 감소

♠ 의사 결정 트리의 크기가 증가함에 따라 검증 오류는 처음에는 감소하다가 (트리 크기 = 7까지), 이후에는 모델이 과적합되기 시작하면서 다시 증가

학습 오차와 검증 오차 (2)



데이터: MNIST 숫자 이미지 데이터

모델 클래스 : 합성곱 신경망 (CNNs)

하이퍼파라미터: 정규 파라미터의 수

- 학습 세트 오차는 단조 감소

- 검증 오차는 초기에는 U자형 곡선을 보이다가 다시 감소하기 시작하여, 최대 파라미터 수 (1,000,000)에서 최저값에 도달

III. 손실 함수

- 모든 오차가 동일한 가중치를 갖는 것은 아님

  스팸 이메일 필터링: 정상 이메일을 스팸으로 잘못 분류하는 것이 스팸을 정상 이메일로 잘못 분류하는 것보다 더 위험함

- 기계 학습은 유틸리티 함수를 극대화하는 대신 *손실 함수*를 최소화

  $L(x, y, \hat{y})$: 실제 정답이 $f(x) = y$ 이고 예측값이 $h(x) = \hat{y}$ 일 경우 손실된 유틸리티의 양

$L(y, \hat{y})$ ⇐ 단순화된 표현

$L(x, y, \hat{y})$ = Utility (주어진 입력 x에 대해 y를 적용한 결과) − Utility (주어진 x에 대해 를 $\hat{y}$ 적용한 결과)

// 스팸을 정상 이메일로 예측

$L(spam, nospam) = 1$

truth prediction

// 정상 이메일을 스팸으로 예측

$L(nospam, spam) = 10$

일반화 손실

절대값 손실 : $L_1(y, \hat{y}) = |y - \hat{y}|$

제곱 오차 손실 : $L_2(y, \hat{y}) = (y - \hat{y})^2$

0/1 손실 : $L_{0/1}(y, \hat{y}) = 0$, if $y = \hat{y}$, 다른 모든 경우 1

기계 학습은 모든 입력-출력 쌍의 집합 $\mathcal{E}$에 대해 기대 손실을 최소화하는 가설을 선택

손실 함수 L 에 대한 가설 h 의 *일반화 손실*:

$$GenLoss_L(h) = \sum_{(x,y)\in\mathcal{E}} L\big(y, h(x)\big)\underbrace{P(x,y)}$$

사전 결합 확률 *(<- 가중치 비슷한 역할)*

*최적의 가설*은 다음 식에 의해 선택:

$$h^* = \underset{h\in\mathcal{H}}{\operatorname{argmin}}\, GenLoss_L(h)$$

경험적 손실

♠ 실제 상황에서는 $P(x, y)$ 가 주어지지 않는 경우가 많은데, 이럴 경우 $GenLoss_L(h)$ 를 추정할 수 밖에 없음

크기가 N인 샘플 집합 E에 대한 *경험적 손실*:

$$EmpLoss_{L,E}(h) = \sum_{(x,y)\in\mathcal{E}} L\big(y, h(x)\big)\frac{1}{N}$$

최적의 가설을 추정하는 공식은 아래와 같이 정의:

$$\hat{h}^* = \underset{h\in\mathcal{H}}{\operatorname{argmin}}\, EmpLoss_L(h)$$

참 함수와의 차이

참 함수 f가 가설 공간 $\mathcal{H}$에 존재할 경우 학습 문제가 *실현 가능*함

학습을 통해 구한 최적의 가설 $\hat{h}^*$는 참 함수 f와 다를 수 있으며 그 이유는 다음과 같음

- ♠ 실현 불가능성. $f \notin \mathcal{H}$인 경우, 데이터가 많아도 f를 복원할 수 없음
- ♠ 분산. 학습 과정에서 서로 다른 샘플 집합에 대해 다른 가설이 반환됨
- ♠ 노이즈. f 가 비결정적이거나 노이즈가 있는 경우, 동일한 x 에 대해 다른 값을 $f(x)$ 로 반환할 수 있습니다 (예: 사람의 키 측정).
- ♠ 계산 복잡성. 큰 가설 공간 $\mathcal{H}$ 를 체계적으로 탐색하는 것은 계산적으로 다루기 어려울 수 있음. 종종 전체 공간의 일부만 탐색하여 상당히 좋은 (하지만 최적은 아닌) 가설을 얻음

IV. 정규화

최소화를 이용한 모델 선택:

$$\hat{h}^* = \underset{h \in \mathcal{H}}{\operatorname{argmin}}\, Cost\,(h)$$

단,

$$Cost(h) = EmpLoss(h) + \lambda Complexity(h)$$

손실과 가설 복잡성을 균형있게
조절하는 양수 값의 하이퍼파라미터

정규화 함수

정규화: 복잡한 가설에 명시적으로 패널티를 부여하여 더 규칙적인 함수가 선호되도록 하는 과정

- 정규화 함수의 선택은 가설 공간에 따라 달라짐

  가설 공간이 다항식인 경우, 계수들의 제곱의 합을 정규화 함수로 사용할 수 있으며, 계수 제곱의 합 값이 작으면 흔들림 현상이 줄어듬

하이퍼파라미터 튜닝

교차 검증을 통해 여러 하이퍼파라미터에 적용될 수 있는 좋은 값을 선택하는 것은 어려운 작업임

- *수작업 튜닝*. 일부 값을 추측하고 모델을 훈련시킨 후 성능을 측정하고, 그 다음에 새로운 값을 제안

- *그리드 탐색*. 모든 값의 조합을 테스트하여 검증 데이터에서 가장 성능이 좋은 하나를 선택. 공간의 크기가 크거나 값이 연속적인 경우, 무작위 샘플링 적용

- *베이지안 최적화*. 하이퍼파라미터 조정을 기계 학습 문제 자체로 취급

입력: 하이퍼파라미터 값들의 벡터 $\boldsymbol{x}$

출력: 검증 세트에 대한 결과 모델의 총 손실 y

학습 세트: $(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_N)$ 학습 과정에서 생성된 데이터

작업: N 개의 쌍 $\left(y_i, h(\boldsymbol{x_i})\right)$에 대하여 y를 최소화해주는 함수 $y = h(\boldsymbol{x})$ 를 탐색