인 공 지 능

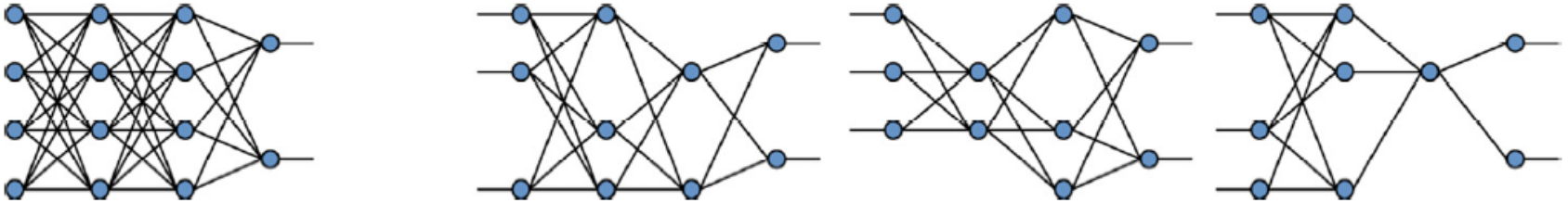
[심층학습 최적화 IV]

jaekoo@kookmin.ac.kr

KMU 국민대학교
KOOKMIN UNIVERSITY

5.4.4 드롭아웃

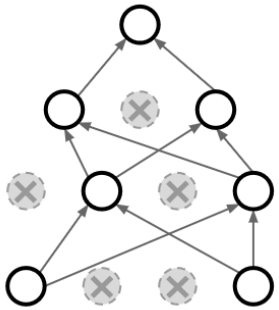
- 드롭아웃dropout 규제 기법
  - 완전연결층의 노드 중 일정 비율 (일반적으로 p=0.5 적용)을 임의 선택하여 제거
    → 남은 부분 신경망 학습

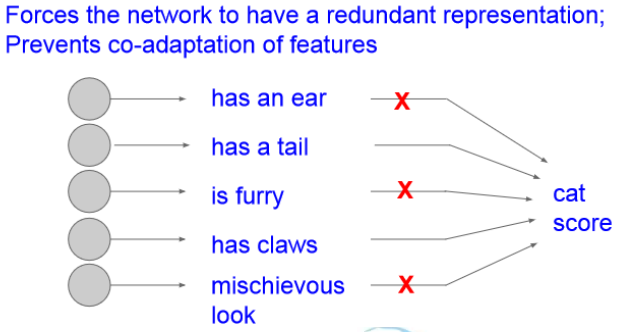


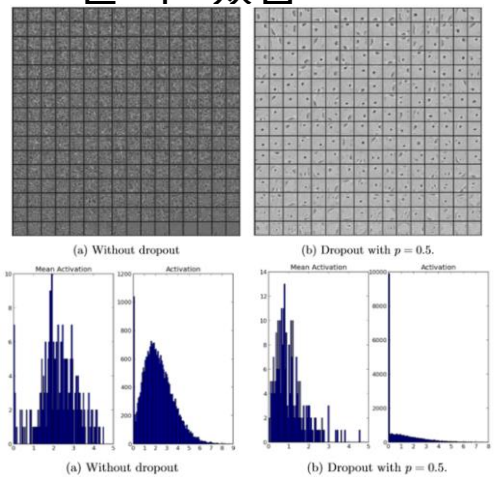
(a) 원래 신경망(4-4-4-2 구조) (b) 드롭아웃된 3개의 신경망 예시

그림 5-27 드롭아웃된 신경망

  - 많은 부분 신경망을 만들고, 앙상블 결합하는 기법으로 볼 수 있음







5.4.4 드롭아웃

- 신경망의 완전연결층에 드롭아웃을 적용함 ([알고리즘 5-8])

알고리즘 5–8 드롭아웃을 채택한 기계 학습 알고리즘

입력: 드롭아웃 비율 p_{input}, p_{hidden}
출력: 최적해 $\hat{\Theta}$

1. 난수를 생성하여 초기해 Θ를 설정한다.
2. while (! 멈춤 조건) // 수렴 조건
3. 미니배치 $\mathbb{B}$를 샘플링한다.
4. for (i=1 to $|\mathbb{B}|$) // $\mathbb{B}$의 샘플 각각에 대해
5. 입력층은 p_{input}, 은닉층은 p_{hidden} 비율로 드롭아웃을 수행한다.
6. 드롭아웃된 부분 신경망 $\Theta_i^{dropout}$로 전방 계산을 한다.
7. 오류 역전파를 이용하여 $\Theta_i^{dropout}$를 위한 그레이디언트 $\nabla_i^{dropout}$를 구한다.
8. $\nabla_1^{dropout}, \nabla_2^{dropout}, \cdots, \nabla_{|\mathbb{B}|}^{dropout}$의 평균 $\nabla_{ave}^{dropout}$를 계산한다.
9. $\Theta = \Theta - \rho\nabla_{ave}^{dropout}$ // 가중치 갱신
10. $\hat{\Theta} = \Theta$

5.4.4 드롭아웃

- 6번째 줄의 전방 계산

l번째 은닉층의 j번째 노드의 연산:

$$z_j^l = \tau_l(s_j^l)$$

이때 $s_j^l = \mathbf{u}_j^l \mathbf{z}^{l-1}$

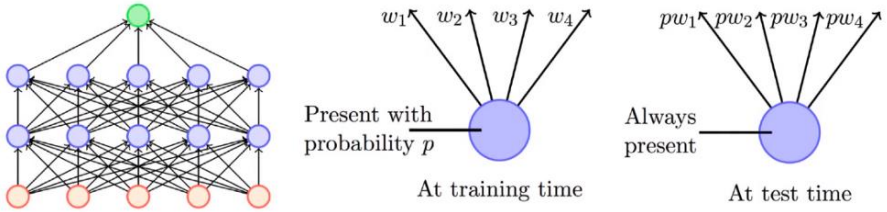
$\Rightarrow$

드롭아웃 적용:

$$z_j^l = \tau_l(s_j^l)$$

이때 $\begin{cases} \tilde{\mathbf{z}}^{l-1} = \mathbf{z}^{l-1} \odot \boldsymbol{\pi}^{l-1} \\ s_j^l = \mathbf{u}_j^l \tilde{\mathbf{z}}^{l-1} \end{cases}$ (5.35)

- 참거짓boolean 배열 $\boldsymbol{\pi}$로 노드 제거 여부를 표시
- $\boldsymbol{\pi}$는 (미니배치) 샘플마다 독립적으로 정하는데, 난수로 설정함
- 일반적으로 입력층 제거 비율 $P_{input} = 0.2$, 은닉층 제거 비율 $P_{hidden} = 0.5$로 설정



Intuition:
Scale the output of each neuron by p

5.4.4 드롭아웃

예측 단계

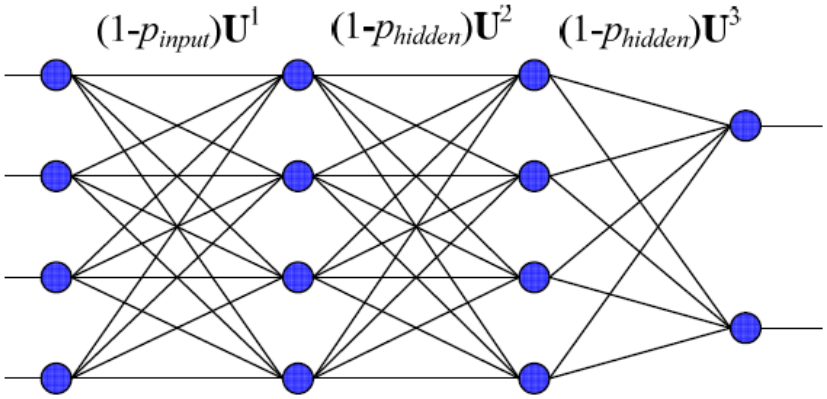


그림 5-28 드롭아웃의 예측 단계

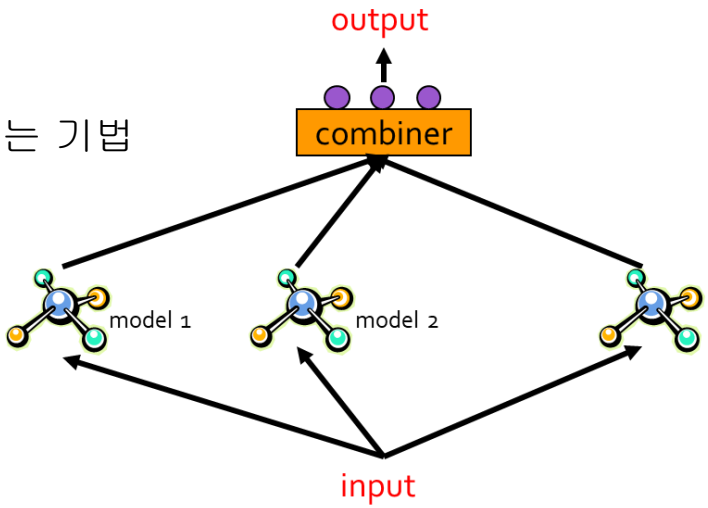
- 앙상블 효과 모방
  - 학습 과정에서 가중치가 (1-드롭아웃 비율)만큼만 참여했기 때문에 p만큼 보정

메모리와 계산 효율

- 추가 메모리는 참거짓 배열 $\boldsymbol{\pi}$, 추가 계산은 작음
- 실제 부담은 신경망의 크기에서 옴: 보통 은닉 노드 수를 $\frac{1}{P_{hidden}}$만큼 늘림

5.4.5 앙상블 기법

- 앙상블ensemble
  - 서로 다른 여러 개의 모델을 결합하여 일반화 오류를 줄이는 기법
  - 현대 기계학습은 앙상블도 규제로 여김

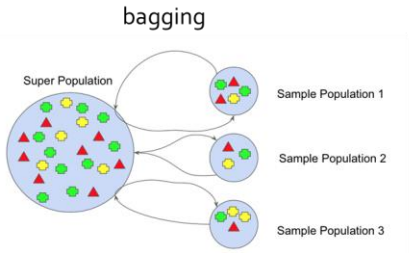


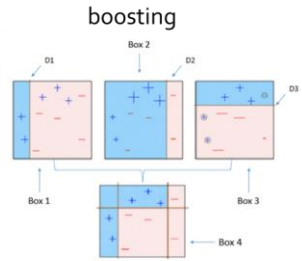
- 두 가지 일

1. 서로 다른 예측기를 학습하는 일
   - 서로 다른 구조의 신경망 여러 개를 학습 또는

     같은 구조를 사용하지만 서로 다른 초기값과 하이퍼 매개변수를 설정하고 학습
   - 예, 배깅bagging (bootstrap aggregating) (훈련집합을 여러 번 샘플링하여 서로 다른 훈련집합을 구성)
   - 예, 부스팅boosting (i번째 예측기가 틀린 샘플을 i+1번째 예측기가 잘 인식하도록 연계성을 고려)
2. 학습된 예측기를 결합하는 일 → 모델 평균model averaging
   - 여러 모델의 출력에서 평균을 구하거나 투표하여 최종 결과 결정

- 자세한 내용은 12장 참고

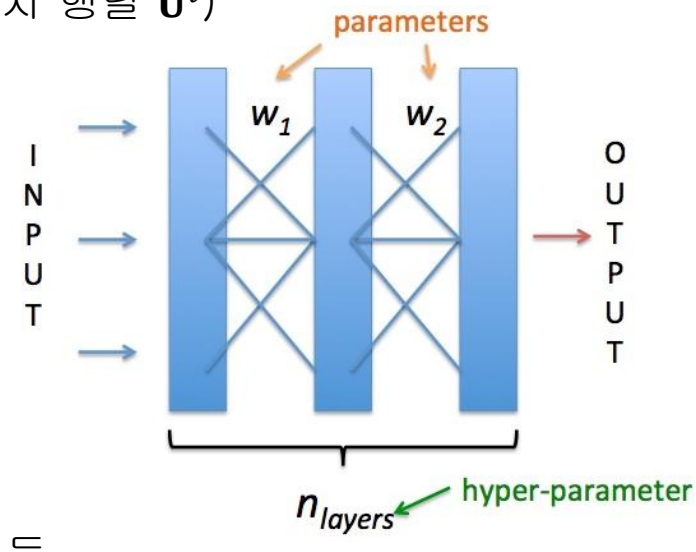




5.5 하이퍼 매개변수 최적화

학습 모델에는 두 종류의 매개변수가 있음

- 내부 매개변수parameter 혹은 가중치weight
  - 신경망의 경우, 가중치 $\boldsymbol{\Theta}$로 표기 (예, 식 (4.1)의 가중치 행렬 $\mathbf{U}^{l}$)
  - 학습 알고리즘이 최적화함
    - 주어진 데이터로부터 결정됨
- 하이퍼 매개변수hyper-parameter
  - 모델의 외부에서 모델의 동작을 조정함
    - 사람에 의해서 결정됨
  - 예, 은닉층의 개수, CNN의 필터 크기와 보폭, 학습률 등



5.5 하이퍼 매개변수 최적화

하이퍼 매개변수 선택

- 표준 참고 문헌이 제시하는 기본값을 사용하면 됨
  - 보통 여러 후보 값 또는 범위를 제시
- 후보 값 중에서 주어진 데이터에 최적인 값 선택 ← 하이퍼 매개변수 최적화

알고리즘 5-9 하이퍼 매개변수 최적화

입력: 훈련집합 $\mathbb{X}$, 검증집합 $\mathbb{X}_{validate}$, 하이퍼 매개변수집합 $\mathcal{H}$
출력: 최적 하이퍼 매개변숫값 $\hat{\mathrm{H}}$

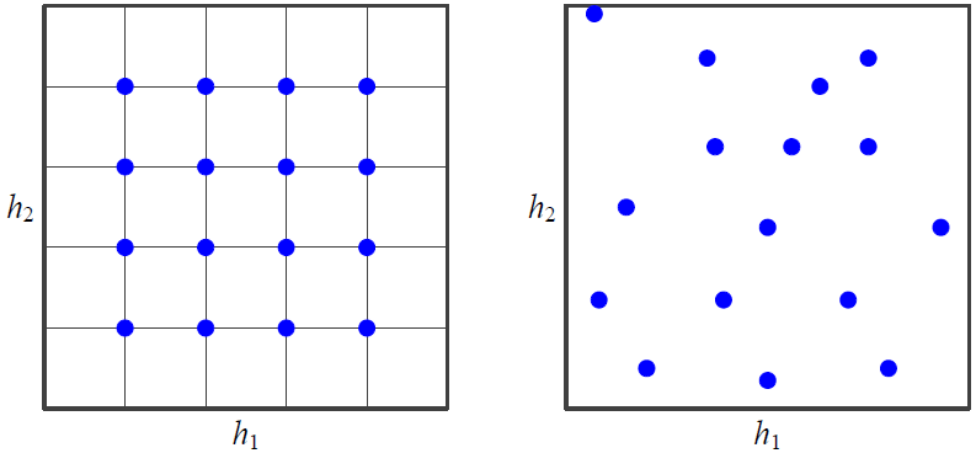
1 $q_{max} = 0$
2 repeat
3 하이퍼 매개변수 조합 $\tilde{\mathrm{H}}$을 생성한다.
4 $\tilde{\mathrm{H}}$으로 설정된 모델을 $\mathbb{X}$로 학습한다.
5 학습된 모델을 $\mathbb{X}_{validate}$로 성능 q를 측정한다.
6 if $(q > q_{max})$ $q_{max} = q$, $\hat{\mathrm{H}} = \tilde{\mathrm{H}}$
7 until(멈춤 조건)

- 라인 3을 구현하는 방법에 따라 수동 탐색, 격자 탐색, 임의 탐색

→ 최근 학습을 통한 자동 탐색 방법들이 연구됨 (예, 자동화된 기계학습automated machine learning)

5.5 하이퍼 매개변수 최적화

격자 탐색grid search과 임의 탐색random search



(a) 격자 탐색 (b) 임의 탐색

그림 5-29 하이퍼 매개변수 탐색 방법

- 임의 탐색은 난수로 하이퍼 매개변수 조합을 생성함

로그 공간log space 간격으로 탐색

- 어떤 하이퍼 매개변수는 로그 공간를 사용해야 함
- 예, 학습률 범위가 [0.0001~0.1]일 때
  - 등 간격은 0.0001, 0.0002, 0.0003, …, 0.0998, 0.0999로 총 1000개 값을 조사
  - 로그 공간 간격은 2배씩 증가시킴

    즉 0.0001, 0.0002, 0.0004, 0.0008, …, 0.0256, 0.0512, …를 조사함

5.5 하이퍼 매개변수 최적화

- 차원의 저주 문제 발생
  - 매개변수가 m개이고 각각이 q개 구간이라면 q^m개의 점을 조사해야 함
- 임의 탐색이 우월함
  - [Bergstra2012]의 실험 (32개의 매개변수) → 임의 탐색이 유리함
  - [그림 5-30]은 임의 탐색이 유리한 이유를 설명

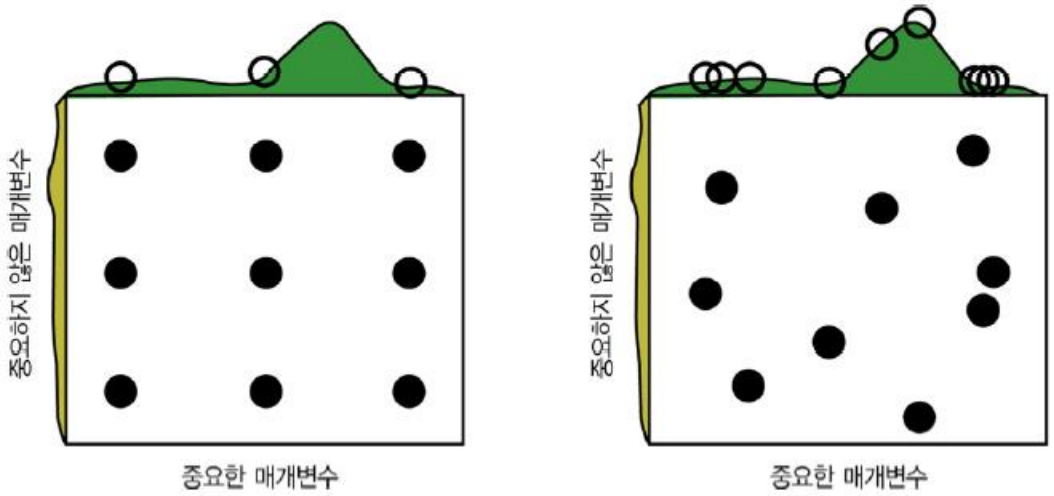


그림 5-30 격자 탐색과 임의 탐색의 성능 비교

- 크게 했다가 점차 세밀해지는coarse-fine 탐색

Coarse to fine search

```
val acc: 0.412000, lr: 1.405206e-04, reg: 4.793564e-01, (1 / 100)
val acc: 0.214000, lr: 7.231888e-06, reg: 2.321281e-04, (2 / 100)
val acc: 0.208000, lr: 2.119571e-06, reg: 8.011857e+01, (3 / 100)
val acc: 0.196000, lr: 1.551131e-05, reg: 4.374936e-05, (4 / 100)
val acc: 0.079000, lr: 1.753300e-05, reg: 1.200424e+03, (5 / 100)
val acc: 0.223000, lr: 4.215128e-05, reg: 4.196174e+01, (6 / 100)
val acc: 0.441000, lr: 1.750259e-04, reg: 2.110807e-04, (7 / 100)
val acc: 0.241000, lr: 6.749231e-05, reg: 4.226413e+01, (8 / 100)
val acc: 0.482000, lr: 4.296863e-04, reg: 6.642555e-01, (9 / 100)
val acc: 0.079000, lr: 5.401602e-06, reg: 1.599828e+04, (10 / 100)
val acc: 0.154000, lr: 1.618508e-06, reg: 4.925252e-01, (11 / 100)
```

```
val acc: 0.527000, lr: 5.340517e-04, reg: 4.097824e-01, (0 / 100)
val acc: 0.492000, lr: 2.279484e-04, reg: 9.991345e-04, (1 / 100)
val acc: 0.512000, lr: 8.680827e-04, reg: 1.349727e-02, (2 / 100)
val acc: 0.461000, lr: 1.828377e-04, reg: 1.220193e-02, (3 / 100)
val acc: 0.468000, lr: 1.113730e-04, reg: 5.244389e-02, (4 / 100)
val acc: 0.498000, lr: 9.477776e-04, reg: 2.001293e-03, (5 / 100)
val acc: 0.469000, lr: 1.484369e-04, reg: 4.328313e-01, (6 / 100)
val acc: 0.522000, lr: 5.586261e-04, reg: 2.312685e-04, (7 / 100)
val acc: 0.530000, lr: 5.000183e-04, reg: 8.259964e-02, (8 / 100)
val acc: 0.489000, lr: 1.979168e-04, reg: 1.010889e-04, (9 / 100)
val acc: 0.498000, lr: 2.836031e-04, reg: 2.486271e-03, (10 / 100)
val acc: 0.475000, lr: 2.021162e-04, reg: 2.287887e-01, (11 / 100)
val acc: 0.460000, lr: 1.135527e-04, reg: 3.985040e-02, (12 / 100)
val acc: 0.515000, lr: 6.947668e-04, reg: 1.562888e-02, (13 / 100)
val acc: 0.531000, lr: 9.471549e-04, reg: 1.433895e-03, (14 / 100)
val acc: 0.509000, lr: 3.140888e-04, reg: 2.057518e-01, (15 / 100)
val acc: 0.514000, lr: 6.438349e-04, reg: 3.033781e-01, (16 / 100)
val acc: 0.502000, lr: 3.921784e-04, reg: 2.787126e-04, (17 / 100)
val acc: 0.509000, lr: 9.752279e-04, reg: 2.050865e-03, (18 / 100)
val acc: 0.500000, lr: 2.412048e-04, reg: 4.997821e-04, (19 / 100)
val acc: 0.466000, lr: 1.319314e-04, reg: 1.189915e-02, (20 / 100)
val acc: 0.516000, lr: 8.039527e-04, reg: 1.520291e-02, (21 / 100)
```

5.6 2차 미분을 이용한 방법

- 5.6.1 뉴턴 방법
- 5.6.2 켤레 경사도 방법
- 5.6.3 유사 뉴턴 방법

5.6 2차 미분을 이용한 최적화

경사 하강법 다시 보기

예제 5-2 경사 하강법의 동작

변수가 2개인 아래 함수 J의 최저점을 구하는 문제를 생각하자. 먼저 1차 도함수인 그레이디언트 ∇J를 구한다.

$$J(\mathbf{w}) = J(w_1, w_2) = (w_1 - 2)^2 + 4(w_2 - 1)^2$$
$$\nabla J(\mathbf{w}) = \left(\frac{\partial J}{\partial w_1}, \frac{\partial J}{\partial w_2}\right)^{\mathrm{T}} = (2(w_1 - 2),\ 8(w_2 - 1))^{\mathrm{T}}$$

시작점이 $\mathbf{w}_1 = (4,2)^{\mathrm{T}}$라고 가정하면 $\mathbf{w}_1$에서의 그레이디언트는 $\nabla J(\mathbf{w}_1) = (4,8)^{\mathrm{T}}$이다. 식 (2.58)에 따라 새로운 점 $\mathbf{w}_2$를 계산하면, $\mathbf{w}_2 = (3.2,0.4)^{\mathrm{T}}$가 된다. 이때 학습률 ρ는 0.2라고 가정하자.

$$\mathbf{w}_2 = \mathbf{w}_1 - \rho\nabla J(\mathbf{w}_1) = \begin{pmatrix}4\\2\end{pmatrix} - 0.2\begin{pmatrix}4\\8\end{pmatrix} = \begin{pmatrix}3.2\\0.4\end{pmatrix}$$

$\mathbf{w}_2$에서 그레이디언트는 $\nabla J(\mathbf{w}_2) = (2.4, -4.8)^{\mathrm{T}}$이고, 새로운 점은 $\mathbf{w}_3 = (3.2,0.4)^{\mathrm{T}} - 0.2(2.4, -4.8)^{\mathrm{T}} = (2.72,1.36)^{\mathrm{T}}$가 된다. 비슷한 계산을 반복하면 다음과 같은 궤적을 그리며 최저점 $(2,1)^{\mathrm{T}}$로 접근하는 것을 알 수 있다.

$$\mathbf{w}_1 = \begin{pmatrix}4\\2\end{pmatrix} \rightarrow \mathbf{w}_2 = \begin{pmatrix}3.2\\0.4\end{pmatrix} \rightarrow \mathbf{w}_3 = \begin{pmatrix}2.72\\1.36\end{pmatrix} \rightarrow \mathbf{w}_4 = \begin{pmatrix}2.432\\0.784\end{pmatrix} \rightarrow \mathbf{w}_5 = \begin{pmatrix}2.2592\\1.1296\end{pmatrix} \rightarrow \cdots$$

1차 미분를 사용하는 경사 하강법

- 현재 기계학습의 주류 알고리즘
- 두 가지 개선책 [Bottou2017]
  - 경사도의 잡음을 줄임 (예, 미니배치 사용 등)
  - 2차 미분 정보를 활용 (5.6절 주제)

5.6 2차 미분을 이용한 최적화

- 경사 하강법을 더 빠르게 할 수 있나?
  - [그림 5-31]에서 파란 경로는 경사 하강법이 해를 찾아가는 과정

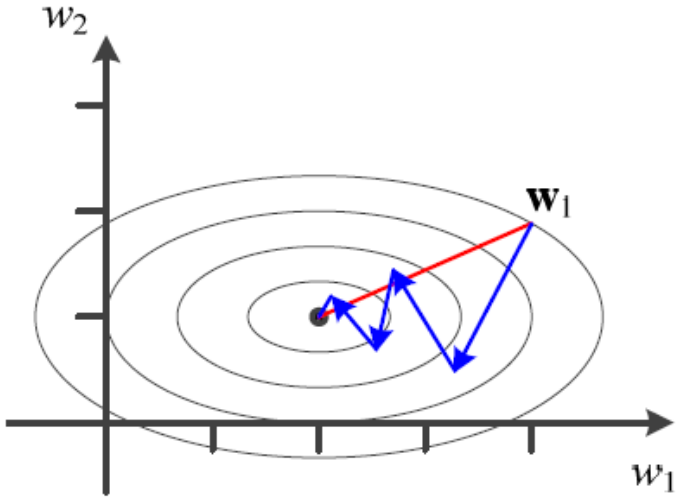


그림 5-31 1차 미분을 사용하는 경사 하강법을 더 빠르게 할 수 있는가

  - 1차 미분 정보로는 빨간 경로를 알 수 없음

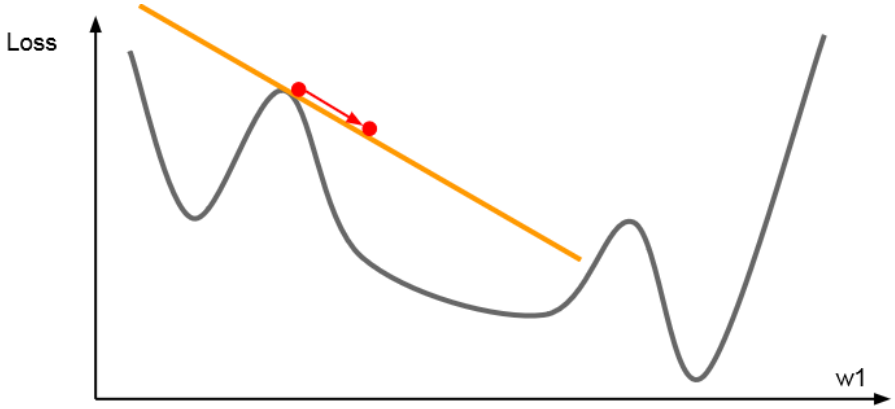
    ∵ 1차 미분은 현재 위치에서 지역적인 기울기 정보만 알려주기 때문

→ 뉴턴 방법Newton methods은 2차 미분 정보를 활용하여 빨간 경로를 알아냄

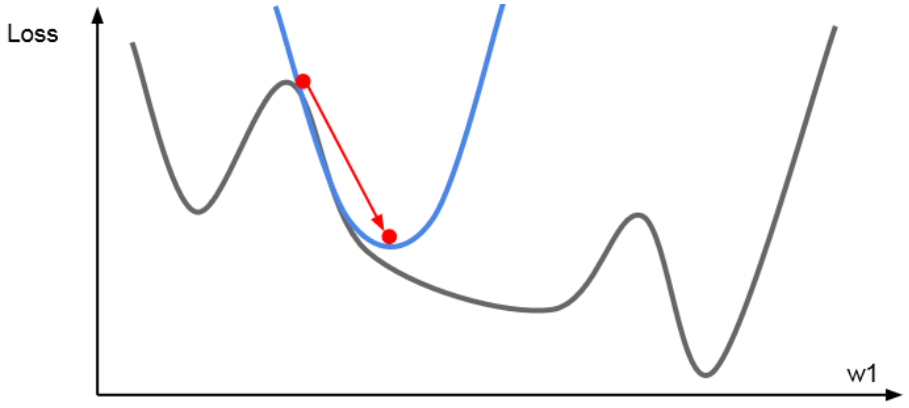
5.6 2차 미분을 이용한 최적화

- 1차 미분 최적화와 2차 미분 최적화 비교

1차 미분의 최적화
- 경사도 사용하여 선형 근사 사용
- 근사치 최소화



2차 미분의 최적화
- 경사도와 헤시안을 사용하여 2차 근사 사용
- 근사치의 최소값



5.6.1 뉴턴 방법

- 테일러 급수를 적용하면,

← 테일러 급수: 주어진 함수를 정의역에서 특정 점의 미분계수들을 계수로 가지는 다항식의 극한(멱급수)으로 표현함

$$J(w+\delta) \approx J(w) + J'(w)\delta + \frac{J''(w)}{2}\delta^2 \tag{5.36}$$

  - 식 (5.37)은 변수가 여러 개일 때 ($\mathbf{H}$는 헤시언Hessian 행렬)

$$J(\mathbf{w}+\boldsymbol{\delta}) \approx J(\mathbf{w}) + \nabla J^{\mathrm{T}}\boldsymbol{\delta} + \frac{1}{2}\boldsymbol{\delta}^{\mathrm{T}}\mathbf{H}\boldsymbol{\delta} \tag{5.37}$$

  - δ로 미분하면,

$$\frac{\partial J(w+\delta)}{\partial \delta} \approx J'(w) + \delta J''(w)$$

  - [그림 5-32]처럼 $w+\delta$를 최소점이라 가정하면,

$$\frac{\partial J(w+\delta)}{\partial \delta} \approx J'(w) + \delta J''(w) = 0$$

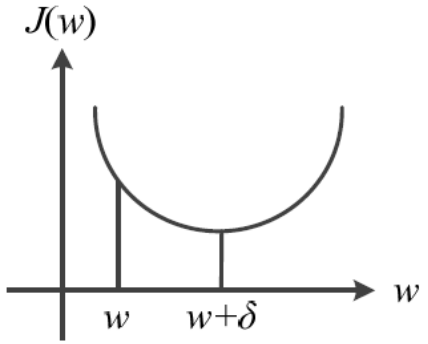


그림 5-32 변수가 하나인 상황의 뉴턴 방법

5.6.1 뉴턴 방법

- 식을 조금 정리하면,

$$\delta = -\frac{J'(w)}{J''(w)} = -\left(J''(w)\right)^{-1} J'(w) \qquad (5.38)$$

- 변수가 여러 개인 경우로 확장하면,

$$\boldsymbol{\delta} = -\mathbf{H}^{-1}\nabla J \qquad (5.39)$$

5.6.1 뉴턴 방법

예제 5-3 **뉴턴 방법**

[예제 5-2]의 함수 $J(\mathbf{w}) = J(w_1, w_2) = (w_1 - 2)^2 + 4(w_2 - 1)^2$을 다시 활용한다. 그레이디언트와 헤시언 행렬을 구하면 아래와 같다.

$$\nabla J(\mathbf{w}) = \begin{pmatrix} 2w_1 - 4 \\ 8w_2 - 8 \end{pmatrix}$$

$$\mathbf{H} = \begin{pmatrix} 2 & 0 \\ 0 & 8 \end{pmatrix}$$

시작점 $(4,2)^T$를 현재 점 $\mathbf{w}_1$이라고 하면, $\mathbf{w}_1$에서 그레이디언트는 $\nabla J(\mathbf{w}_1) = \begin{pmatrix} 4 \\ 8 \end{pmatrix}$이다. 이것을 식 (5.39)에 대입하면 아래와 같다.

$$\boldsymbol{\delta} = -\begin{pmatrix} 2 & 0 \\ 0 & 8 \end{pmatrix}^{-1} \begin{pmatrix} 4 \\ 8 \end{pmatrix} = -\begin{pmatrix} 1/2 & 0 \\ 0 & 1/8 \end{pmatrix} \begin{pmatrix} 4 \\ 8 \end{pmatrix} = -\begin{pmatrix} 2 \\ 1 \end{pmatrix}$$

다음 점 $\mathbf{w}_2$는 아래 식으로 계산할 수 있고, 이 점은 최저점임을 알 수 있다.

$$\mathbf{w}_2 = \mathbf{w}_1 + \boldsymbol{\delta} = \begin{pmatrix} 4 \\ 2 \end{pmatrix} - \begin{pmatrix} 2 \\ 1 \end{pmatrix} = \begin{pmatrix} 2 \\ 1 \end{pmatrix}$$

5.6.1 뉴턴 방법

뉴턴 방법의 적용

- [예제 5.3]은 2차 함수에 뉴턴 방법을 적용했으므로,
  3차 항 이상을 무시한 식 (5.36)을 사용했음에도 최적 경우 제시
- 기계학습이 사용하는 목적함수는 2차 함수보다 복잡한 함수이므로 한 번에 최적해에 도달 불가능
  → [알고리즘 5-10]과 같이 반복하는 뉴턴 방법을 사용해야 함

알고리즘 5-10 뉴턴 방법

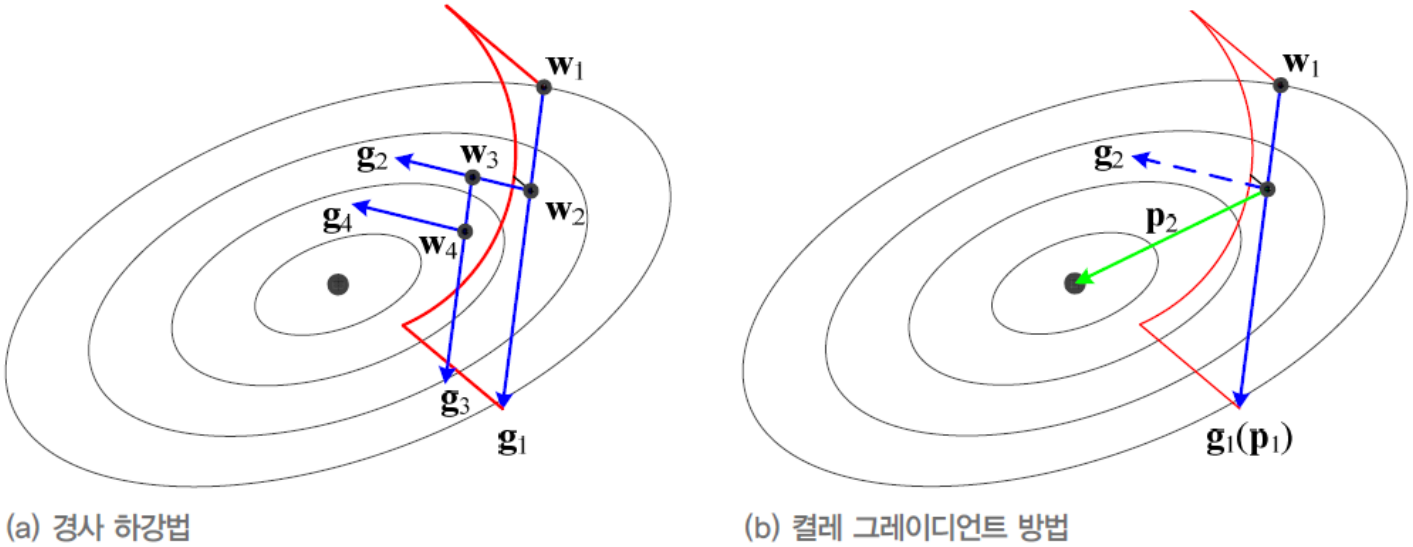
입력 : 훈련집합 $\mathbb{X}$, $\mathbb{Y}$
출력 : 최적해 $\hat{\boldsymbol{\Theta}}$

1 난수를 생성하여 초기해 $\boldsymbol{\Theta}$를 설정한다.
2 repeat
3 $\quad \boldsymbol{\delta} = -\mathbf{H}^{-1}\nabla J$를 계산한다. // 식 (5.39)
4 $\quad \boldsymbol{\Theta} = \boldsymbol{\Theta} + \boldsymbol{\delta}$
5 until (멈춤 조건)
6 $\hat{\boldsymbol{\Theta}} = \boldsymbol{\Theta}$

- 줄 3에서 헤시언 $\mathbf{H}$를 구해야 함
  → 매개변수의 개수를 m이라 할 때 $O(m^3)$이라는 과다한 연산량 필요
  → 켤레 경사도 방법이 대안 제시

5.6.2 켤레 경사도 방법

- 직선 탐색line search: 이동 크기를 결정하기 위해 직선으로 탐색하고, 미분
  - [그림 5-33(a)]는 경사 하강법이 학습률을 직선 탐색으로 찾는 상황을 예시



(a) 경사 하강법 (b) 켤레 그레이디언트 방법

그림 5-33 경사 하강법과 켤레 그레이디언트 방법의 비교

  - 켤레 경사도 방법conjugate gradient method
    - [그림 5-33(a)]의 경사 하강법은 근본적으로 이전 경사 하강법과 같음
      - $\mathbf{g}_2$로 직선 탐색할 때 직전에 사용한 $\mathbf{g}_1$ 정보를 전혀 고려하지 않음
    - [그림 5-33(b)]의 켤레 경사도는 직전 정보를 사용하여 해에 빨리 접근

5.6.3 유사 뉴턴 방법

유사 뉴턴 방법quasi-Newton methods의 기본 개념

- 문제점
  - 경사 하강법: 수렴 효율성 낮음
  - 뉴턴 방법: 헤시안 행렬 연산 부담 → 헤시언 **H**의 역행렬을 근사하는 행렬 **M**을 사용
- 대표적으로 점진적으로 헤시안을 근사화하는 LFGS가 많이 사용됨
- 기계 학습에서는 **M**을 저장하는 메모리를 적게 쓰는limited memory L-BFGS를 주로 사용함
  - 전체 배치를 통한 갱신을 할 수 있다면, L-BFGS 사용을 고려함

기계학습에서 2차 미분 정보의 활용

- 현재 널리 활용되지는 않지만 연구 계속되고 있음

"… These methods, such as those built on noise reduction and second-order techniques, offer the ability to attain improved convergence rates, overcome the adverse effects of high nonlinearity and ill-conditioning, and exploit parallelism and distributed architectures in new way. … 잡음 감소와 2차 미분과 같은 방법은 수렴 속도를 향상하고, 심한 비선형과 불량 조건 같은 부정적 효과를 극복하며, 새로운 방식으로 병렬분산 계산 구조를 이용하는 길을 제시한다." [Bottou2017]