BITAMIN

9주차 복습세션

1조 정세웅 한형진 황예은

특별출연 김민지 감사합니다

목차

1. 다양한 분류 알고리즘
1-1. 로지스틱 회귀

: **이진 분류 (Binary Classification)**을 풀기 위한 대표적인 알고리즘
- 예측하고자 하는 것 : class 1에 속할 확률 p

- **Odds**

$$odds(p) = \frac{p}{1-p}$$

  - probability p → odds(p)
    [0, 1] [0, inf)

- **Logit** (Log-Odds)

$$logit(p) = log(\frac{p}{1-p})$$

  - probability p → logit(p)
    [0, 1] (-inf, inf)

logit을 사용하는 이유!

1. 다양한 분류 알고리즘

1-1. 로지스틱 회귀

- **Logit을 사용하는 이유**

- linear layer output이 probability가 되도록 하려면?

$$p = Wx + b$$

[0, 1] 범위로 들어오도록 W, b 규제하기 어려움

- linear layer output이 logit이 되도록 하려면?

$$log(\frac{p}{1-p}) = Wx + b$$

(-inf, inf) 만족시키기 위해 따로 규제할 필요X

1. 다양한 분류 알고리즘
1-1. 로지스틱 회귀

- **logit으로부터 확률 p 구하기**

$$log(\frac{p}{1-p}) = Wx + b$$

$$\frac{1-p}{p} = \frac{1}{exp(Wx+b)}$$

$$p = \frac{exp(Wx+b)}{1+exp(Wx+b)} = \frac{1}{1+exp(-(Wx+b))} = \boxed{sigmoid(Wx + b)}$$

이게 바로 로지스틱 회귀의 가설함수!

1. 다양한 분류 알고리즘
1-1. 로지스틱 회귀

- 가설 함수

$$h(x) = sigmoid(Wx + b)$$

- 손실 함수 : **Binary Cross Entropy**

$$Cost(h(x), y) = -ylog(h(x)) - (1 - y)log(1 - h(x))$$

- 예측 : **threshold 0.5**

$$h(x) \geq 0.5 \rightarrow predict\ 1$$

$$h(x) < 0.5 \rightarrow predict\ 0$$

1. 다양한 분류 알고리즘

1-2. 확률적 경사 하강법

- **경사 하강법 (GD)**
  : 매 iteration마다 모든 indices i = 1, …, N 사용

$$\theta^{k+1} = \theta^k - \frac{\alpha_k}{N}\sum_{i=1}^{N} \nabla f_i(\theta^k)$$

- **확률적 경사 하강법 (SGD)**
  : k번째 iteration에 하나의 랜덤한 index i(k) 사용

$$i(k) \sim \mathrm{Uniform}\{1, \dots, N\}$$
$$\theta^{k+1} = \theta^k - \alpha_k \nabla f_{i(k)}(\theta^k)$$

1. 다양한 분류 알고리즘

1-2. 확률적 경사 하강법

- **Minibatch SGD with replacement**
  : k번째 iteration에 B개의 랜덤 indices 복원 추출

$$i(k,1),\ldots,i(k,B) \sim \text{Uniform}\{1,\ldots,N\}$$

$$\theta^{k+1} = \theta^k - \frac{\alpha_k}{B}\sum_{b=1}^{B} \nabla f_{i(k,b)}(\theta^k)$$

- **Minibatch SGD without replacement**
  : k번째 iteration에 B개의 랜덤 indices 비복원 추출

$$\sigma^k \sim \text{permutation}(N)$$

$$\theta^{k+1} = \theta^k - \frac{\alpha_k}{B}\sum_{b=1}^{B} \nabla f_{\sigma^k(b)}(\theta^k)$$

(+) 배치 사이즈 B 정하는 법

수학적으로는,
- 노이즈/랜덤성이 크면 ↑
- 노이즈/랜덤성이 작으면 ↓

현실적으로는,
- GPU 메모리가 허용하는 한 ↑↑
  (배치가 클수록 효율적인 GPU 연산이 이뤄짐)

1. 다양한 분류 알고리즘
1-2. 확률적 경사 하강법

- **Cyclic SGD**
  : 매 iteration마다 정해진 순열대로 하나의 index 사용

$$\theta^{k+1} = \theta^k - \alpha_k \nabla f_{\mathrm{mod}(k,N)+1}(\theta^k)$$

예) 1, 2, 3, … N,
1, 2, 3, …, N,

- **Shuffled Cyclic SGD**
  : 매 iteration마다 정해진 순열대로 하나의 index 사용
  매 epoch마다 다른 순열 사용

$$\theta^{k+1} = \theta^k - \alpha_k \nabla f_{\sigma^{\lfloor \frac{k}{N} \rfloor}(\mathrm{mod}(k,N)+1)}(\theta^k)$$

where $\sigma^0, \sigma^1, \ldots$ is a sequence of random permutations

예) 1, 2, 3, …, N,
4, 1, 5, …, 8,
N, 7, 3, …, 1

1. 다양한 분류 알고리즘
1-2. 확률적 경사 하강법

- **딥러닝에선 주로...**
  - shuffled cyclic minibatch SGD (without replacement)
  - GPU 메모리가 허용하는 한 최대의 배치사이즈 B

(+) 용어

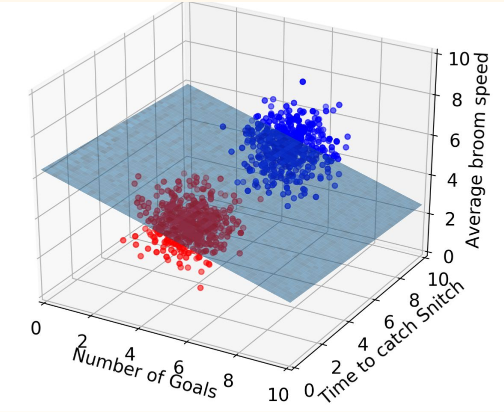
- **epoch** : 모든 indices (모든 데이터)가 한 번 사용된 최적화/학습 진행 유닛
  - GD : 1 iteration = 1 epoch
  - SGD, cyclic SGD, shuffled cyclic SGD : N iterations = 1 epoch
  - minibatch SGD : N/B iterations = 1 epoch

2. 서포트 벡터 머신 (SVM)
2-1. 선형 SVM

: **이진 분류 (Binary Classification)**을 풀기 위한 대표적인 알고리즘

- **선형 구분 가능 (Linearly Separable)**
  : 다차원 공간에 분포한 두 집단이 하나의 다차원 평면(hyperplane)으로 구분 가능
  → **결정 경계 (Decision Boundary)**
  Wx+b

  데이터를 분류하는 hyperplane은 여러개가 나올 수 있
  최적의 결정 경계는?



2. 서포트 벡터 머신 (SVM)

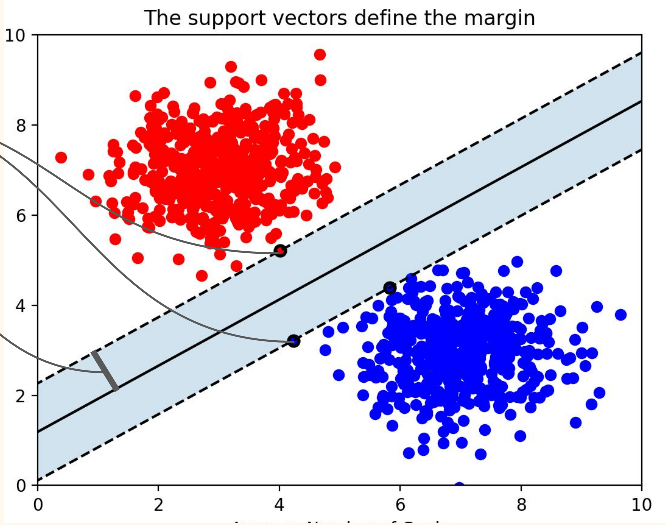
2-1. 선형 SVM

- **최적의 결정 경계는?**

- 마진(Margin)이 최대인 결정 경계

서포트 벡터(support vector)
: 결정 경계와 가까이 있는 데이터 포인트

마진 (Margin)
: 결정 경계와 서포트 벡터 사이의 거리



2. 서포트 벡터 머신 (SVM)
2-1. 선형 SVM

- **최적의 결정 경계는?**

- 마진(Margin)이 최대인 결정 경계
- **라그랑주 승수법**을 통해 다음과 같은 해를 찾을 수 있다

$$\mathbf{w} = \sum_{i=1}^{n} \alpha_i y_i \mathbf{x_i}$$

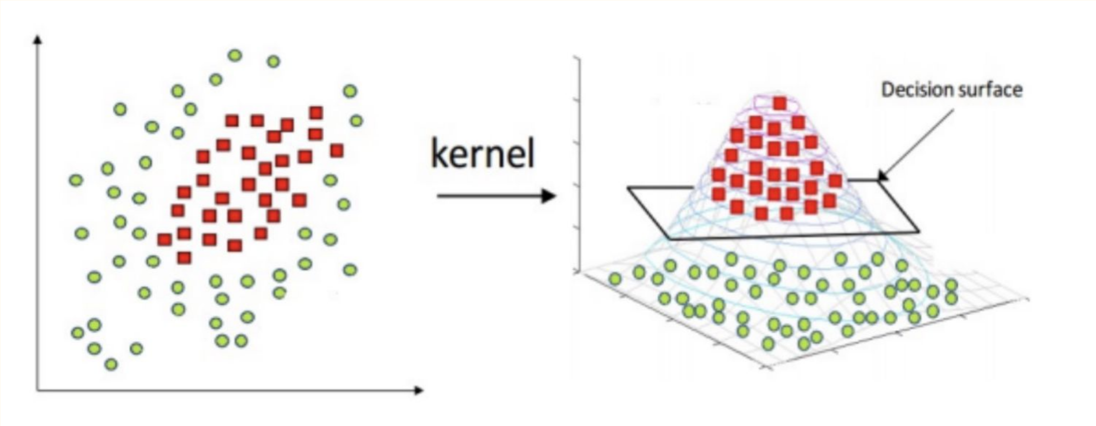
$$b = \frac{1}{N_{SV}} \sum_{i=1}^{N_{SV}} (\mathbf{w} \cdot \mathbf{x_i} - y_i)$$

2. 서포트 벡터 머신 (SVM)

2-1. 비선형 SVM

- **Kernel-SVM**

- 저차원 공간을 고차원 공간으로 매핑
- 예) Polynomia Kernel, Sigmoid Kernel, Gaussian RBF Kernel



선형 구분 불가능　　　　　　선형 구분 가능