

11. Support vector machines

- Remind) Logistic Regression : classify 2 (or F) class based on decision boundary
if inner product $\langle \theta, x \rangle$ is larger than threshold.

* Support Vector Machines (SVM)

Goal: 고차원 공간에서 선형 결정 경계를 통해 data를 분류하자

Idea: Margin을 최대화해서 일반적인 성능이 가장 좋은 classifier를 찾자.

비슷하게 SVM에서 decision boundary 확률 but data distribution 가정 안함

11.1 The perceptron algorithm (Limit & Problem)

* 시작점: Consider binary classification

- Dataset : $D = \{(x_1, y_1), \dots, (x_n, y_n)\} \in \mathbb{R}^d \times \{-1, 1\}$

- Goal: Hyperplane을 따라 2가지 클래스로 분류하는 classifier를 찾자.

- Classifier f :
$$f(x) = \begin{cases} 1, & \langle \theta, x \rangle \geq b \\ -1, & \langle \theta, x \rangle < b \end{cases}$$

* 전제 조건: data is linearly separable, data가 선형적으로 완전히 분리가능해야만 작동

* 문제점 : 1) 해가 존재하지 않을 수 있음 (linearly non-separable data)

2) 여러 해가 존재하면 어떤 θ 를 선택할지 기준 없음 (θ and b aren't necessarily unique)

→ SVM Motivation

① fully linear separable을 가정 → ② 완벽히 선형분리되지 않는 case (duality 도입)

11.2 Hard-Margin SVM

* Goal: fully linearly separable data에 대해 Margin을 최대화하는 Hyperplane 찾기

In more detail: we want to maximize margin $m \geq 0$ s.t.

1) "+1"로 분류된 모든 points는 Hyperplane의 양쪽 (positive side)에 있고, 그 Hyperplane까지의 거리는 최소 m .

1) "-1"로 분류된 모든 points는 Hyperplane의 음수쪽 (negative side)에 있고, 그 Hyperplane까지의 거리는 최소 m .

* distance of a point x to a hyperplane $\mathcal{H} = \{x : \langle \theta, x \rangle = b\}$:

$$\text{distance}(x, \mathcal{H}) = \frac{|\langle \theta, x \rangle - b|}{\|\theta\|_2} = \frac{|\langle \theta, x - x_0 \rangle|}{\|\theta\|_2} = \frac{|\langle \theta, x \rangle - b|}{\|\theta\|_2} \quad ***$$

By definition, the vector θ is perpendicular to Hyperplane $\mathcal{H} \Rightarrow$ Any vector lies on Hyperplane.

만약 $x_0, x_1 \in \mathcal{H}$ 일때 $x_1 - x_0$ vector는 \mathcal{H} 위에 있고, orthogonal to θ :

$$\langle x_1 - x_0, \theta \rangle = \langle x_1, \theta \rangle - \langle x_0, \theta \rangle = b - b = 0$$

$\Rightarrow \theta$ 가 \mathcal{H} 에 perpendicular, 최소 거리($x \sim \mathcal{H}$)는 projection of $x - x_0$ onto θ , where $x_0 \in \mathcal{H}$.

이걸 수식으로 표현하면 ***

이걸 우리 Goal에 대응하면 : $y_i \frac{|\langle \theta, x_i \rangle - b|}{\|\theta\|_2} \geq m$.

* Optimization Task :

1) Margin 포함

$$\max_{\theta, b} m \text{ subject to } y_i \frac{|\langle \theta, x_i \rangle - b|}{\|\theta\|_2} \geq m, \quad i \in \{1, \dots, n\}$$

2) scale normalization

$$\|\theta\|_2 = \frac{1}{m} \text{로 설정하면} \rightarrow \max_{\theta, b} \frac{1}{\|\theta\|_2} \text{ subject to } y_i \frac{|\langle \theta, x_i \rangle - b|}{\|\theta\|_2} \geq 1$$

$$\text{미분할때 수식 상에 만들고, min으로 하려고... } f(\theta) = \frac{1}{2} \|\theta\|^2 = \frac{1}{2} \theta^T \theta \rightarrow \nabla_{\theta} f(\theta) = \nabla_{\theta} \left(\frac{1}{2} \theta^T \theta \right) = \theta$$

11.3 Soft-Margin SVM → 현실 data는 대부분 완벽히 선형분리 불가능 → duality 적용 → slack variables ξ_i

Noise or Overlap이 있는 현실 data를 어떻게 다룰까? → Margin 조건을 일부 위반 허용하자

• 새로운 constraint : $y_i(\langle \theta, x_i \rangle - b) \geq 1 - \xi_i$ ($\xi_i \geq 0$)

we take the slack variable in the objective into account by penalizing large values of ξ_i

• 새로운 optimization : $\min_{\theta, b} \frac{1}{2} \|\theta\|^2 + \lambda \sum_{i=1}^n \xi_i$ subject to $y_i(\langle \theta, x_i \rangle - b) \geq 1 - \xi_i$, $\xi_i \geq 0$

$$\min_{\theta, b} \frac{1}{2} \|\theta\|^2 + \lambda \sum_{i=1}^n \max(1 - y_i(\langle \theta, x_i \rangle - b), 0) \quad \begin{cases} y_i(\langle \theta, x_i \rangle - b) \geq 1 - \xi_i \\ \xi_i \geq 0 \text{ 포함} \end{cases}$$

$$\min_{\theta, b} \frac{1}{2} \|\theta\|^2 + \lambda \sum_{i=1}^n \xi_i \quad \text{subject to} \quad \xi_i = \max(1 - y_i(\langle \theta, x_i \rangle - b), 0)$$

12 Nonlinear Features Kernels [Idea!!!]

12.1 Linear estimators

- 우리가 지금까지 본 대부분의 모델 (Linear Reg, Logistic Reg, SVM...) 은 모두 Linear estimator.
- 즉, 학습 결과인 예측 함수는 모두 $\langle \theta, x \rangle$ 같은 형태로 선형적인 결정을 내림. Kernel을 결합해서 비선형에서도 사용가능한 강력한 classifier를 만들자.

12.2 The Representer Theorem

- 많은 경우의 최적해 θ 는 단순한 training data의 linear combination으로 표현 가능. i.e. $\hat{\theta} = \sum_i a_i x_i$ 형태
- 이렇게 식을 정리하면 kernel methods에서 최적해를 θ 대신 a 로 바꾸고, inner product만 알면 됨.

12.3 Nonlinear Features Kernel

- 구체적인 Idea를 도입하자.
- 1) data를 고차원 feature space로 보내는 function $\phi(x)$ 를 정의하자.
- 2) But 고차원 vector calculation 비싸니까, inner product $\langle \phi(x), \phi'(x) \rangle$ 만 효율적으로 계산하자 (kernel trick)
- ex) Polynomial, Gaussian Kernel 등

12.4 SVMs with a Gaussian kernel

- SVM에 Kernel을 결합해서 non-linear decision boundary로 확장해보자.
- dual form으로 생각하면 모든 연산이 inner product에만 의존
 - 이 inner product를 kernel function으로 바꿈

Dual Form ???

: 원래 최적화 문제 (primal problem)를 변형한 또 다른 형태의 최적화 문제. 원래 문제에 Lagrange Multiplier 도입해서 Dual Problem 구성하고, 이를 통해 원래 문제 해 유도 이 계산 효과적으로 ...

- General primal problem

$$\min_{\theta} f(\theta) \quad \text{s.t.} \quad g_i(\theta) \leq 0, \quad h_i(\theta) = 0$$

- Lagrangian problem

$$L(\theta, \alpha, \beta) = f(\theta) + \sum_i \alpha_i g_i(\theta) + \sum_j \beta_j h_j(\theta)$$

- Dual function

$$g(\alpha, \beta) = \min_{\theta} L(\theta, \alpha, \beta)$$

- Dual Problem

$$\max_{\alpha \geq 0, \beta} g(\alpha, \beta)$$

| Primal | Dual |
|-----------------------------------|----------------------|
| 비용을 최소화하자 | 제약조건을 고려해 보자. |
| 각 자원에 가격(α)을 붙여 총 가치 극대화 | 어떤 자원이 귀한가? |
| 내가 얼마나 싸게 만들 수 있나? | 이 제약 조건들은 얼마나 가치 있나? |

e.g) $\min_x x^2 \quad \text{s.t.} \quad x \geq 1$

* Primal

$$x=1 \rightarrow \min = 1^2 = 1.$$

* Dual

$$L(x, \alpha) = x^2 + \alpha(1-x)$$

$$g(\alpha) = \min_x L(x, \alpha) = \min_x x^2 - \alpha x + \alpha$$

• 최소조건 $\frac{d}{dx} L = 2x - \alpha = 0 \Rightarrow x = \frac{\alpha}{2}$

• 제약조건 $x \geq 1 \Rightarrow x = \frac{\alpha}{2} \geq 1 \Rightarrow \alpha \geq 2.$

$$\Rightarrow g(\alpha) = L(\frac{\alpha}{2}, \alpha) = -\frac{\alpha^2}{4} + \alpha.$$

• 이걸 $\alpha \geq 2$ 에서 최대화하면 $\alpha = 2. \Rightarrow x = 1.$

e.g) $\min_{\theta \in \mathbb{R}^d} \frac{1}{2} \|X\theta - y\|^2 + \frac{\lambda}{2} \|\theta\|^2$

* Primal

$$\theta^* = (X^T X + \lambda I)^{-1} X^T y$$

여기서 $X \in \mathbb{R}^{d \times n}$ 인데, X^{-1} 를 구해야 함..

$d = 10^6, n = 10^3$ 이면 $\Rightarrow X^{-1} \quad 10^6 \times 10^6$ 역행렬 구해야 함

* Dual

$$\min_{\alpha \in \mathbb{R}^n} \frac{1}{2} \alpha^T (K + \lambda I) \alpha - y^T \alpha$$

• $K = X X^T \in \mathbb{R}^{n \times n}$ (귀할 행렬)

• $(K + \lambda I)^{-1} \alpha$ 역행렬 $\in \mathbb{R}^{n \times n} \quad 10^3 \times 10^3$ 행렬

12. Nonlinear features and kernels

12.1 Linear estimators

$\text{loss}(z, y) = \text{예측} \sim \text{실제값 간의 Loss}$

일반적 $\min_{\theta} J(\theta)$ subject to $J(\theta) = \frac{1}{n} \sum_{i=1}^n \text{loss}(h_{\theta}(x_i), y_i) + r(\theta)$

Regularization Term

$$h_{\theta}(x_i) = \langle \theta, x_i \rangle$$

1) Logistic Regression : Loss can be formulated as function of inner products $\langle \theta, x \rangle$

$$\text{loss}(\langle \theta, x \rangle, y) = \log(1 + e^{-y \langle \theta, x \rangle})$$

• 위 일반식에서 $r(\theta) = 0$.

• 벡터 x 를 $\langle \theta, x \rangle > 0$ 이면 1, 아니면 -1로 구별.

2) Ridge / Linear Regression

$$\text{loss}(\langle \theta, x \rangle, y) = (\langle \theta, x \rangle - y)^2, \quad r(\theta) = \frac{\lambda}{2} \|\theta\|_2^2$$

Ridge Regression

3) Support vector machine (SVM)

$$\text{loss}(\langle \theta, x \rangle, y) = \max(1 - y \langle \theta, x \rangle - b, 0), \quad r(\theta) = \frac{\lambda}{2} \|\theta\|_2^2$$

이런 모든 model은 loss와 예측이 $\langle \theta, x \rangle$ 에만 의존하며.
linear decision boundary만 표현가능
그러면
Regularized empirical의 해 $\hat{\theta}$ 는
risk minimization
어떤 구조를 가질까?

12.2 The Representer Theorem

• Theorem 1: Representer theorem

- 가정: 만약 $r(\theta) = r(\|\theta\|_2)$, $h_{\theta}(x) = \langle \theta, x \rangle$, loss is convex

- 최적해 minimizer $\hat{\theta}$: $\hat{\theta} = \sum_{i=1}^n d_i x_i$, 즉, linear combination으로 표현가능.

LA) SPAN !!!

이 Linear combination 구조 덕분에 1) parameter θ 가 아닌 d_i 만 학습하면 됨.

2) inner product만으로 학습가능 \rightarrow Kernel trick

즉, model을 feature vector 공간에서 생각하지 않고 d_i 를 통해 training sample space에서 표현할 수 있음

ex) 병원, 환자 \rightarrow 100개 features 수감. (혈압, 체온, 나이, ...) \Rightarrow 각각 환자 data는 100차원 vector $x \in \mathbb{R}^{100}$
but 지금까지 병원이 방문 환자 수 = 50명. \Rightarrow training data는 $x_1, x_2, \dots, x_{50} \in \mathbb{R}^{100}$

(x_i 가 만드는 subspace는 많아봐야 max. 50차원)

모델 예측이 $\hat{y} = \langle \theta, x \rangle$ 형태니까 100차원 전체 공간에서 아무 θ 를 찾는게 아니라,

실제로는 training sample (50개)들이 span 하는 50차원 Subspace 안에서 예측값이 달라짐.

때문에 training loss는 $\langle \theta, x_i \rangle$ 만 사용되는데 이런 x_i 방향 영향 받지만,

그나 training data에 직교인 부분은 영향 X

loss 영향 X, regularization 증가. (제한 없애는게 낫지)

(12.2 The Representer Theorem 증명)

Proof) $J(\theta) = \frac{1}{n} \sum_{i=1}^n \text{loss}(\langle \theta, x_i \rangle, y_i) + r(\|\theta\|)$

만약 θ 가 x_1, x_2, \dots, x_n 이 span 하는 subspace에 속하지 않는다면, 거기에서 가장 가까운 점으로 projection 하면, object function의 값이 줄거나 같아진다. \rightarrow 따라서 optimum은 항상 이 subspace 안에 있음.

step 1: $\theta = \theta_{\parallel} + \theta_{\perp}$

- $\theta_{\parallel} \in \text{span}\{x_1, \dots, x_n\}$ 즉 training data로 span되는 공간 위
- $\theta_{\perp} \in \text{orthogonal complement}$ 그 공간에 수직인 부분
- (기하적: 벡터를 subspace 위로 정사영)

step 2: loss는 θ_{\perp} 와 무관

$$\langle \theta, x_i \rangle = \langle \theta_{\parallel} + \theta_{\perp}, x_i \rangle = \langle \theta_{\parallel}, x_i \rangle + \underbrace{\langle \theta_{\perp}, x_i \rangle}_{\text{직교} = 0}$$

$$\langle \theta, x_i \rangle = \langle \theta_{\parallel}, x_i \rangle$$

step 3: Regularizer는 θ_{\perp} 를 늘릴수록 증가

$$r(\|\theta\|) = r(\|\theta_{\parallel} + \theta_{\perp}\|) \geq r(\|\theta_{\parallel}\|)$$

즉, loss는 그대로인데 regularizer는 더 작거나 같아짐

$\Rightarrow \theta_{\perp}$ 는 값에 늘게 좋음 $\hat{=}$ optimum은 항상 $\theta_{\perp} = 0$

$$\Rightarrow \hat{\theta} \in \text{span}\{x_1, \dots, x_n\} \Rightarrow \hat{\theta} = \sum_i \alpha_i x_i$$

12.3 Nonlinear Features and Kernels

기존 model은 모두 linear라서 복잡한 decision boundary 나누기 어려움

\rightarrow non-linear feature map $\phi(x)$ 을 도입해서 고차원 공간에서 선형분리 가능하게 하자

그런데 $\phi(x)$ 를 그대로 바꾸면 연산 복잡해짐, kernel을 사용하자.

• Kernel Function $K(x, x')$

$$K(x, x') = \langle \phi(x), \phi(x') \rangle$$

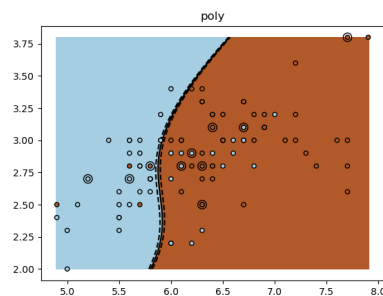
\rightarrow 고차원 $\phi(x)$ 는 계산 하지 않아도 됨

\rightarrow 대신 inner product 계산

• 자주 쓰이는 Kernel

1) Polynomial : $K(x, x') = (\langle x, x' \rangle + c)^d$

다항식 형태 변환



\Rightarrow Gaussian (RBF) :

$$K(x, x') = \exp\left(-\frac{\|x - x'\|^2}{2\sigma^2}\right)$$

무한 차원 확장 가능

3) Sigmoid

$$K(x, x') = \tanh \langle \alpha \langle x, x' \rangle + c \rangle$$

신경망과 유사.

12.4 Support Vector machine with a Gaussian kernel

- SVM - Margin을 최대화하는 결정 경계 학습.

Hard-Margin SVM: $f(x) = \text{sign}(\langle \theta^*, x \rangle + b^*)$

↪ dual form:

$$\theta^* = \sum_{i=1}^n \alpha_i x_i \Rightarrow \chi(x) = \text{sign} \left(\sum_{i=1}^n \alpha_i y_i \langle x_i, x \rangle + b \right)$$

(inner product 기반이 kernel 사용 가능)

- kernelized SVM

$$f(x) = \text{sign} \left(\sum_{i=1}^n \alpha_i x_i K(x, x_i) + b \right)$$

- * Gaussian (RBF) 사용 시:

$$K(x, x') = \exp \left(- \frac{\|x - x'\|^2}{2\sigma^2} \right)$$

