

Contents

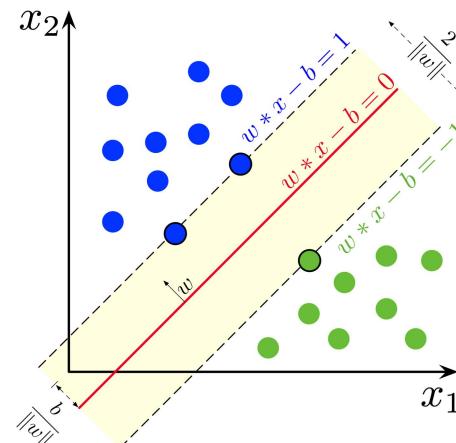
- ▼ 1 SVM의 분류방법
 - 1.1 개략적인 모형
 - 1.2 최적화 문제
- ▼ 2 최적화 문제와 support vector의 식별
 - 2.1 Hard margin SVM
 - 2.2 Classification
 - 2.3 Soft margin SVM
 - 2.4 선형모형과 SVM
 - 2.5 Dual problem
 - 2.6 dot product 와 유사성 similarity
 - 2.7 sklearn.svm.SVC
 - 2.8 전처리 및 hyperparameter
 - 2.9 Multiclass SVM
- ▼ 3 Feature space의 확장과 nonlinear decision function
 - 3.1 Kernel trick
 - 3.2 Dual problem
 - ▼ 3.3 Kernel function, positive-definite function
 - 3.3.1 Kernel 함수의 예
 - 3.3.2 Kernel 함수
 - 3.4 Kernels을 이용한 SVM
- ▼ 4 Support Vector Machine의 특징
 - 4.1 장점
 - 4.2 단점
 - 4.3 분석
- ▼ 5 SVR, support vector regression
 - 5.1 Hard margin
 - 5.2 Soft margin
 - 5.3 Prediction

11월 2주

Support Vector Machine

- Support Vector Machine (SVM)은 discriminant 모형으로 [wiki \(\[https://en.wikipedia.org/wiki/Support-vector_machine\]\(https://en.wikipedia.org/wiki/Support-vector_machine\)\)](https://en.wikipedia.org/wiki/Support-vector_machine)에 따르면 기본모형은 63년, 비선형변환에 사용하는 kernel trick은 93년에 발표되었다.
- 최초 제안된 모형은 분류모형이었지만 이후 다른 분석에도 사용할 수 있는 algorithm이 개발되면서 SVM은 전체 모형을 아우르는 용어로 사용하기도 한다.
 - SVC (Support Vector Classification)
 - SVR (Support Vector Regression)
 - SVC (Support Vector Clustering)
- 이후 classification을 중심으로 모형의 특징을 설명한다.
- SVM로 예측확률을 구할 수는 없지만 logistic regression, decision tree 와 함께 가장 많이 사용하는 분류모형이다.
- SVM은 '선형분류기'를 이용하여 오분류 건수를 최소화하는 것을 목적으로 한다.
- Feature의 수에 구애를 받지 않으며 관찰값의 수보다 더 많은 feature가 있는 표본에 적용이 가능하다.
- Kernel function으로 표본공간을 변환하여 비선형 decision boundary를 사용할 수 있다.
- 계산시간이 길다.
- Feature가 많은 경우 overfitting에 취약하다.
- 모형의 어떤 특성이 class를 결정에 영향을 미치는지 알기 어렵다.

Figure 1. Support Vector Machine



- Support vector machine은 support vector(경계에 있는 data point)를 지나는 초평면의 거리를 극대화하는 \mathbf{w} 를 계수로 하는 초평면을 decision boundary로 사용하는 분류기이다.

Contents ⚙

- ▼ 1 SVM의 분류방법
 - 1.1 개략적인 모형
 - 1.2 최적화 문제
- ▼ 2 최적화 문제와 support vector의 식별
 - 2.1 Hard margin SVM
 - 2.2 Classification
 - 2.3 Soft margin SVM
 - 2.4 선형모형과 SVM
 - 2.5 Dual problem
 - 2.6 dot product 와 유사성 similarity
 - 2.7 sklearn.svm.SVC
 - 2.8 전처리 및 hyperparameter
 - 2.9 Multiclass SVM
- ▼ 3 Feature space의 확장과 nonlinear decision function
 - 3.1 Kernel trick
 - 3.2 Dual problem
 - ▼ 3.3 Kernel function, positive-definite function
 - 3.3.1 Kernel 함수의 예
 - 3.3.2 Kernel 함수
 - 3.4 Kernels을 이용한 SVM
- ▼ 4 Support Vector Machine의 특징
 - 4.1 장점
 - 4.2 단점
 - 4.3 분석
- ▼ 5 SVR, support vector regression
 - 5.1 Hard margin
 - 5.2 Soft margin
 - 5.3 Prediction

1 SVM의 분류방법

- 두 유형을 구분하는 선은 보통 유일하지 않으므로 두 유형을 구분하는 '길'(street, 선이 아니라) 중 폭이 가장 넓은 것을 선택한다.
 - Support vector는 decision boundary 위의 관찰값들을 의미한다.
 - Support vector는 decision boundary를 결정하는 표본들이다.
 - Support vector를 움직이면 길의 폭이 달라진다.
 - Support vector 이외의 관찰값은 길을 결정하는데 아무런 영향을 미치지 못한다.
- 결과적으로 support vector들의 특징을 잘 설명하는 feature들을 제외한 다른 feature들의 weight는 0이 된다.
- SVM은 linear regression이나 Bayes모형들과는 다르게 weight vector에 영향을 주는 표본들만을 고려한다.
- Support vector 이외의 표본은 값을 조금 바꾸거나 제거, 추가해도 분류기에 아무런 영향을 미치지 않는다.
- 완벽한 분리가 어려울 때에는 '오분류'에 대해 penalty를 부여하여 분류
- Penalty는 오분류의 여부가 아니라 올바른 분류 영역에서의 거리에 비례하여 커지도록 한다.

1.1 개략적인 모형

- 간단한 notation을 위해 $y \in \{-1, 1\}$, 각 유형에 해당하는 support vector는 각각 x^- 와 x^+ 로 표시한다. Feature는 shape이 (K, n) 인 array로 표시한다.
- 분류기에 사용하는 linear decision boundary는 다음과 같은 특성을 갖는 초평면이다.

$$\begin{aligned}\mathbf{w}^\top \mathbf{x} + b &\geq 0 \text{ for } y_i = 1 \\ \mathbf{w}^\top \mathbf{x} + b &< 0 \text{ for } y_i = -1\end{aligned}$$

- Support vector를 지나는 supporting hyperplane의 일반식은 다음과 같으며

$$\tilde{\mathbf{w}}^\top \mathbf{x} + \tilde{b} = \pm 1$$

- 양변을 a 로 나누어 정규화시켜 사용한다.

$$\mathbf{w}^\top \mathbf{x} + b = \pm 1$$

- Margin 혹은 margin of separation은 초평면에서 가장 가까운 표본들과의 거리로 최적분류기는 margin을 가장 크게하는 초평면이다.

Contents ☰

- ▼ 1 SVM의 분류방법
 - 1.1 개략적인 모형
 - 1.2 최적화 문제
- ▼ 2 최적화 문제와 support vector의 식별
 - 2.1 Hard margin SVM
 - 2.2 Classification
 - 2.3 Soft margin SVM
 - 2.4 선형모형과 SVM
 - 2.5 Dual problem
 - 2.6 dot product 와 유사성 similarity
 - 2.7 sklearn.svm.SVC
 - 2.8 전처리 및 hyperparameter
 - 2.9 Multiclass SVM
- ▼ 3 Feature space의 확장과 nonlinear decision function
 - 3.1 Kernel trick
 - 3.2 Dual problem
 - ▼ 3.3 Kernel function, positive-definite function
 - 3.3.1 Kernel 함수의 예
 - 3.3.2 Kernel 함수
 - 3.4 Kernels을 이용한 SVM
- ▼ 4 Support Vector Machine의 특징
 - 4.1 장점
 - 4.2 단점
 - 4.3 분석
- ▼ 5 SVR, support vector regression
 - 5.1 Hard margin
 - 5.2 Soft margin
 - 5.3 Prediction

1.2 최적화 문제

- Decision boundary와 supporting hyperplane 사이의 거리, 즉 margin은 두 supporting hyperplane 사이의 거리의 절반이며 최적화문제는 두 초평면 사이의 거리를 가장 크게 하는 초평면을 구하는 것이다.

- 평행인 두 supporting hyperplane 그리고 hyperplane과 직각인 vector \mathbf{w} 를 이용하여 margin λ 를 \mathbf{w} 의 함수로 표시한다.

$$x^- = x^+ + \lambda \mathbf{w}, \quad \mathbf{w}^\top x^- = \mathbf{w}^\top x^+ + \lambda \mathbf{w}^\top \mathbf{w}$$

- \mathbf{w} 는 $\mathbf{w}^\top x^+ + b = 1, \mathbf{w}^\top x^- + b = -1$ 과 같이 정규화된 vector이므로,

$$\mathbf{w}^\top x^- = -1 - b = -(\mathbf{w}^\top x^+ + b) - b = -\mathbf{w}^\top x^+ - 2b$$

- $|\lambda|$ 를 거리로 생각하면,

$$\$ \$ - \mathbf{w}^\top x^+ - 2b = \mathbf{w}^\top x^+ + \lambda \mathbf{w}^\top \mathbf{w} \lambda$$

- $2(\underbrace{\mathbf{w}^\top x^+ + b}_{=1}) = \lambda \mathbf{w}^\top \mathbf{w}$, $\text{quad } \text{distance or street width} = \frac{2}{\|\mathbf{w}\|^2}$

다른 유도 방법

- 앞서와 같이 \mathbf{w} 를 정규화하지 않고 거리를 구할 수도 있다.
- $(\mathbf{w}^\top x + b)$ 의 절대값이 커질수록 $\mathbf{w}^\top x + b = 0$ 에서 멀어지므로 모든 절대값 $|\mathbf{w}^\top x_i + b|$ 이 가장 작은 표본의 값을 극대화하면 길의 넓이를 가장 크게 할 수 있다.
- \mathbf{w}^* 의 값은 유일하게 결정되지 않으므로 단위벡터의 제약을 가하거나 목적함수를 표준화시킨다.
- 많은 교재나 블로그에서 이 목적함수도 많이 사용한다.

$$\max_{\mathbf{w}} \min_{x_i} \underbrace{\frac{|\mathbf{w}^\top x_i + b|}{\|\mathbf{w}\|^2}}_{\text{distance between support vectors}} = \max_{\mathbf{w}} \min_{x_i} \frac{y_i (\mathbf{w}^\top x_i + b)}{\|\mathbf{w}\|^2}$$

Contents ☰

- ▼ 1 SVM의 분류방법
 - 1.1 개략적인 모형
 - 1.2 최적화 문제
- ▼ 2 최적화 문제와 support vector의 식별
 - 2.1 Hard margin SVM
 - 2.2 Classification
 - 2.3 Soft margin SVM
 - 2.4 선형모형과 SVM
 - 2.5 Dual problem
 - 2.6 dot product 와 유사성 similarity
 - 2.7 sklearn.svm.SVC
 - 2.8 전처리 및 hyperparameter
 - 2.9 Multiclass SVM
- ▼ 3 Feature space의 확장과 nonlinear decision function
 - 3.1 Kernel trick
 - 3.2 Dual problem
 - 3.3 Kernel function, positive-definite function
 - 3.3.1 Kernel 함수의 예
 - 3.3.2 Kernel 함수
 - 3.4 Kernels을 이용한 SVM
- ▼ 4 Support Vector Machine의 특징
 - 4.1 장점
 - 4.2 단점
 - 4.3 분석
- ▼ 5 SVR, support vector regression
 - 5.1 Hard margin
 - 5.2 Soft margin
 - 5.3 Prediction

2 최적화 문제와 support vector의 식별

2.1 Hard margin SVM

- SVM 분류의 최적화 문제는 다음과 같다.

$$\min_{\mathbf{w}, b} \frac{\mathbf{w}^\top \mathbf{w}}{2}$$

subject to $(\mathbf{w}^\top \mathbf{x}_i + b)y_i \geq 1$ for all i

- 전형적인 quadratic optimization 문제로 제약조건으로 인해 해가 존재하지 않을 수 있지만, 해가 존재한다면 유일한 전역적극소점이다.

$$\mathbf{w}^* = \sum_{i=1}^n a_i^* y_i \mathbf{x}_i, \quad b^* = \frac{1}{y_i} - (\mathbf{w}^*)^\top \mathbf{x}_i, \quad \sum_{i=1}^n a_i^* y_i = 0, \quad \begin{array}{ll} a_i^* = 0 & \text{if } \mathbf{x}_i \notin (X^+ \cup X^-) \\ a_i^* > 0 & \text{if } \mathbf{x}_i \in (X^+ \cup X^-) \end{array}$$

- 여기서 X^+ 와 X^- 은 각 support vector를 지나는 supporting hyperplane이며 이 집합에 포함된 표본의 a_i^* 만 0보다 크다. b^* 는 $a_i^* > 0$ 인 임의의 \mathbf{x}_i 를 선택해서 계산할 수 있다.

일계조건 유도

- Lagrangean은

$$\begin{aligned} \mathcal{L}(\mathbf{w}, b, a) &= \frac{\mathbf{w}^\top \mathbf{w}}{2} - \sum_{i=1}^n a_i y_i (\mathbf{w}^\top \mathbf{x}_i + b - 1) \\ &= \frac{\mathbf{w}^\top \mathbf{w}}{2} - \sum_{i=1}^n a_i y_i (\mathbf{w}^\top \mathbf{x}_i + b) + \sum_{i=1}^n a_i \end{aligned}$$

이 문제의 Karush-Kuhn-Tucker 조건으로부터 weight vector를 계산한다.

$$\begin{aligned} \frac{\partial \mathcal{L}(\mathbf{w}^*, b^*, a^*)}{\partial \mathbf{w}} &= \mathbf{w}^* - \sum_{i=1}^n a_i^* y_i \mathbf{x}_i = 0 \\ \frac{\partial \mathcal{L}(\mathbf{w}^*, b^*, a^*)}{\partial b} &= \sum_{i=1}^n a_i^* y_i = 0 \\ a_i^* y_i ((\mathbf{w}^*)^\top \mathbf{x}_i + b^* - 1) &= 0, \quad a_i^* \geq 0 \end{aligned}$$

Contents ⚙

- ▼ 1 SVM의 분류방법
 - 1.1 개략적인 모형
 - 1.2 최적화 문제
- ▼ 2 최적화 문제와 support vector의 식별
 - 2.1 Hard margin SVM
 - 2.2 Classification
 - 2.3 Soft margin SVM
 - 2.4 선형모형과 SVM
 - 2.5 Dual problem
 - 2.6 dot product 와 유사성 similarity
 - 2.7 sklearn.svm.SVC
 - 2.8 전처리 및 hyperparameter
 - 2.9 Multiclass SVM
- ▼ 3 Feature space의 확장과 nonlinear decision function
 - 3.1 Kernel trick
 - 3.2 Dual problem
 - ▼ 3.3 Kernel function, positive-definite function
 - 3.3.1 Kernel 함수의 예
 - 3.3.2 Kernel 함수
 - 3.4 Kernels을 이용한 SVM
- ▼ 4 Support Vector Machine의 특징
 - 4.1 장점
 - 4.2 단점
 - 4.3 분석
- ▼ 5 SVR, support vector regression
 - 5.1 Hard margin
 - 5.2 Soft margin
 - 5.3 Prediction

2.2 Classification

- 최적화의 해를 decision boundary에 대입하면

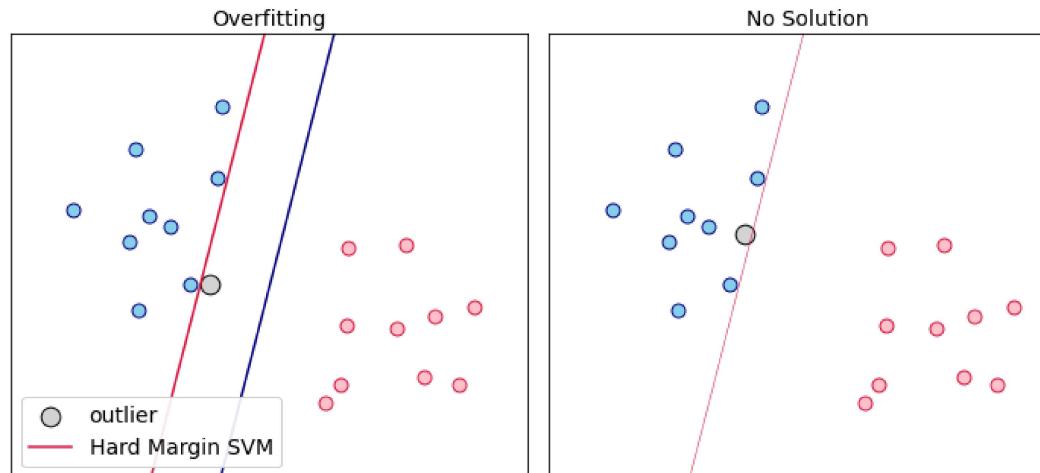
$$(\mathbf{w}^*)^\top \mathbf{x}_j + b^* = \sum_{i=1}^n a_i^* y_i \mathbf{x}_i^\top \mathbf{x}_j + b^*$$

- 따라서 SVM의 분류기는 다음과 같다.

$$y^{\text{pred}}(\mathbf{x}_j) = \text{sign} \left[\sum_{i=1}^n a_i^* y_i (\mathbf{x}_i \cdot \mathbf{x}_j) + b^* \right]$$

- Support vector를 기준으로 같은 decision region에 속한 feature에는 양수, 반대의 경우에는 음수를 지정한다.

Figure 2. Hard margin SVM



Contents ⚙

- ▼ 1 SVM의 분류방법
 - 1.1 개략적인 모형
 - 1.2 최적화 문제
- ▼ 2 최적화 문제와 support vector의 식별
 - 2.1 Hard margin SVM
 - 2.2 Classification
 - 2.3 Soft margin SVM
 - 2.4 선형모형과 SVM
 - 2.5 Dual problem
 - 2.6 dot product 와 유사성 similarity
 - 2.7 sklearn.svm.SVC
 - 2.8 전처리 및 hyperparameter
 - 2.9 Multiclass SVM
- ▼ 3 Feature space의 확장과 nonlinear decision function
 - 3.1 Kernel trick
 - 3.2 Dual problem
 - ▼ 3.3 Kernel function, positive-definite function
 - 3.3.1 Kernel 함수의 예
 - 3.3.2 Kernel 함수
 - 3.4 Kernels을 이용한 SVM
- ▼ 4 Support Vector Machine의 특징
 - 4.1 장점
 - 4.2 단점
 - 4.3 분석
- ▼ 5 SVR, support vector regression
 - 5.1 Hard margin
 - 5.2 Soft margin
 - 5.3 Prediction

2.3 Soft margin SVM

- 초평면으로 완벽한 구분이 불가능하다면 최적화 문제의 해가 존재하지 않는다.
- Hard margin 모형은 overfitting에 취약하다.
- 오류의 크기에 따라 penalty를 적용한다.

$$\min_{\mathbf{w}, b} \frac{\mathbf{w}^\top \mathbf{w}}{2} + \delta \sum_{i=1}^n \eta_i$$

subject to $(\mathbf{w}^\top \mathbf{x}_i + b)y_i \geq 1 - \eta_i$ for all i
 $\eta_i \geq 0$ for all i

- Hard margin SVM은 $\delta = \infty$ 인 문제에 해당한다.
- Soft margin SVM의 최적화문제는 duality를 이용하여 풀 수 있다.

2.4 선형모형과 SVM

- $\mathbf{w}^\top \mathbf{x} + b$ 의 부호를 이용해 예측을 한다는 점에서 SVM은 선형모형이다.
- Loss function의 선택에 따라 분류모형을 구분해 볼 수 있다.

$$\begin{aligned} & \min_{\mathbf{w}, b} \sum_{i=1}^n (y_i - (\mathbf{w}^\top \mathbf{x}_i + b))^2 \\ & \min_{\mathbf{w}, b} \sum_{i=1}^n \ln(1 + \exp(-y_i \cdot (\mathbf{w}^\top \mathbf{x}_i + b))) \end{aligned}$$

- 이 선형확률모형을 이용해 hinge loss로 알려진 다음과 같은 loss를 극소화하면 SVM이다.

$$\max(0, 1 - y \cdot \hat{y}), \quad y = \pm 1, \quad \hat{y} = \mathbf{w}^\top \mathbf{x} + b$$

- y 와 \hat{y} 의 부호가 같으면 올바른 예측이고, 예측이 '확실'하다면 $|\hat{y}| > 1$ 이라면 그 값은 0이다.
- $|\hat{y}| < 1$ 이라면 해당 표본은 길 가운데 위치하고 손실은 0과 1사이의 값이 된다.
- y 와 \hat{y} 의 부호가 다르면 손실값은 1보다 크다. 분류가 정확히 되지 않은 값들만 w 선택에 고려한다.

- 길의 넓이를 크게하면서 예측오차를 극소화하는 방법으로 \mathbf{w} 를 선택한다.

$$\min_{\mathbf{w}} \left(\|\mathbf{w}\|^2 + C \times \frac{1}{n} \sum_{i=1}^n \max(0, 1 - y_i(\mathbf{w}^\top \mathbf{x}_i + b)) \right)$$

- Soft margin 모형의 최적화 문제와 완전히 동일한 구조이며 이 hinge loss function은 다른 분류모형에서도 많이 사용한다.
- C 가 0에 가까울수록 분류오류에 대한 비중이 작아지고 'regularized term' $\|\mathbf{w}\|^2$ 의 비중이 더 커진다. Sklearn의 parameter setting 방법이 그대로 적용된다.

Contents ☰

- ▼ 1 SVM의 분류방법
 - 1.1 개략적인 모형
 - 1.2 최적화 문제
- ▼ 2 최적화 문제와 support vector의 식별
 - 2.1 Hard margin SVM
 - 2.2 Classification
 - 2.3 Soft margin SVM
 - 2.4 선형모형과 SVM
 - 2.5 Dual problem
 - 2.6 dot product 와 유사성 similarity
 - 2.7 sklearn.svm.SVC
 - 2.8 전처리 및 hyperparameter
 - 2.9 Multiclass SVM
- ▼ 3 Feature space의 확장과 nonlinear decision function
 - 3.1 Kernel trick
 - 3.2 Dual problem
 - ▼ 3.3 Kernel function, positive-definite function
 - 3.3.1 Kernel 함수의 예
 - 3.3.2 Kernel 함수
 - 3.4 Kernels을 이용한 SVM
- ▼ 4 Support Vector Machine의 특징
 - 4.1 장점
 - 4.2 단점
 - 4.3 분석
- ▼ 5 SVR, support vector regression
 - 5.1 Hard margin
 - 5.2 Soft margin
 - 5.3 Prediction

2.5 Dual problem

- 앞에서 구한 KKT 필요조건에서 \mathbf{w} 와 b 를 a 의 함수로 정리하고,

$$\mathbf{w}^* = \sum_{i=1}^n a_i^* y_i \mathbf{x}_i, \quad \sum_{i=1}^n a_i^* y_i = 0$$

- Lagrangean에 대입해 정리하면 다음과 같다.(유도과정은 다음 슬라이드)

$$\mathcal{L}(\mathbf{w}(a), b(a), a) = -\frac{1}{2} \sum_{i=1}^n \sum_{j=1}^n a_i a_j y_i y_j (\mathbf{x}_i \cdot \mathbf{x}_j) - b \sum_{i=1}^n a_i y_i + \sum_{i=1}^n a_i$$

- 따라서 동일한 Lagrangean을 갖는 dual problem은 다음과 같다.

$$\begin{aligned} \max_{a \geq 0} \mathcal{L}_{\text{Dual}}(a) &= -\frac{1}{2} \sum_{i=1}^n \sum_{j=1}^n a_i a_j y_i y_j (\mathbf{x}_i \cdot \mathbf{x}_j) + \sum_{i=1}^n a_i \\ \text{subject to } \sum_{i=1}^n a_i y_i &= 0, \quad 0 \leq a_i \leq \delta \end{aligned}$$

- a_i 에 대한 부등호제약을 조심하면 primary보다 최적값을 쉽게 구할 수 있으며 해석이 직관적이다.
- Dual 문제를 고려하는 중요한 이유는 $(\mathbf{x}_i \cdot \mathbf{x}_j)$ 와 같이 문제를 표본의 내적형태로 표시할 수 있고, 사소하게는 \mathbf{w} 와 b 에 대한 dependency를 제거할 수 있다.
- Primary도 마찬가지지만 dual 문제의 라그랑지안은 a 에 대해 이차함수이고 따라서 해가 존재한다면 전역적극대점이 된다. Decision tree나 neural network 같이 국지적극점이나 어중간한 해를 완전히 배제할 수 있다.

Contents ☰

- ▼ 1 SVM의 분류방법
 - 1.1 개략적인 모형
 - 1.2 최적화 문제
- ▼ 2 최적화 문제와 support vector의 식별
 - 2.1 Hard margin SVM
 - 2.2 Classification
 - 2.3 Soft margin SVM
 - 2.4 선형모형과 SVM
 - 2.5 Dual problem
 - 2.6 dot product 와 유사성 similarity
 - 2.7 sklearn.svm.SVC
 - 2.8 전처리 및 hyperparameter
 - 2.9 Multiclass SVM
- ▼ 3 Feature space의 확장과 nonlinear decision function
 - 3.1 Kernel trick
 - 3.2 Dual problem
 - ▼ 3.3 Kernel function, positive-definite function
 - 3.3.1 Kernel 함수의 예
 - 3.3.2 Kernel 함수
 - 3.4 Kernels을 이용한 SVM
- ▼ 4 Support Vector Machine의 특징
 - 4.1 장점
 - 4.2 단점
 - 4.3 분석
- ▼ 5 SVR, support vector regression
 - 5.1 Hard margin
 - 5.2 Soft margin
 - 5.3 Prediction

Dual Problem 유도

$$\begin{aligned}\mathcal{L}(\mathbf{w}(a), b(a), a) &= \frac{\mathbf{w}^\top \mathbf{w}}{2} - \sum_{i=1}^n a_i y_i (\mathbf{w}^\top \mathbf{x}_i + b) + \sum_{i=1}^n a_i \\&= \frac{1}{2} \sum_{i=1}^n a_i y_i \mathbf{w}^\top \mathbf{x}_i - \sum_{i=1}^n a_i y_i (\mathbf{w}^\top \mathbf{x}_i + b) + \sum_{i=1}^n a_i \\&= -\frac{1}{2} \sum_{i=1}^n a_i y_i \mathbf{w}^\top \mathbf{x}_i - b \sum_{i=1}^n a_i y_i + \sum_{i=1}^n a_i \\&= -\frac{1}{2} \sum_{i=1}^n a_i y_i \left(\sum_{i=1}^n a_i y_i \mathbf{x}_i \right) \cdot \mathbf{x}_i - b \sum_{i=1}^n a_i y_i + \sum_{i=1}^n a_i \\&= -\frac{1}{2} \sum_{i=1}^n \sum_{j=1}^n a_i a_j y_i y_j (\mathbf{x}_i \cdot \mathbf{x}_j) - b \sum_{i=1}^n a_i y_i + \sum_{i=1}^n a_i\end{aligned}$$

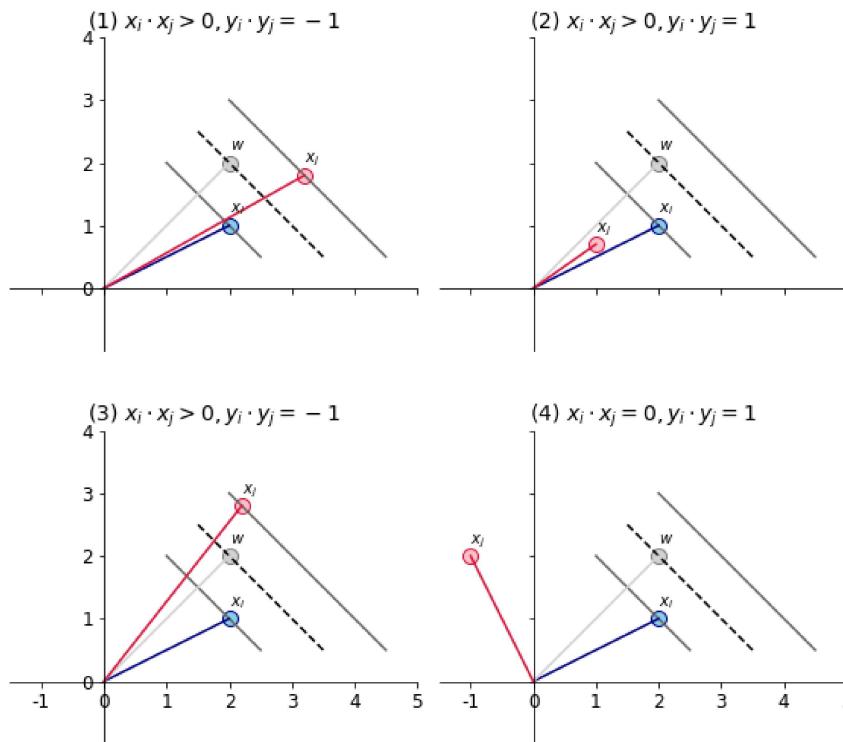
2.6 dot product 와 유사성 similarity

- 단위벡터들의 유사성은 벡터들의 사잇각으로 측정할 수 있다. 대부분의 AI 모형들에서 벡터들의 유사성은 뺄셈이나 dot product로 정의하여 사용한다.
- Dual problem의 목적함수에서 $a_i a_j$ 가 0보다 큰 경우를 생각해보자.
- 두 벡터의 유사성이 높다면 $(\mathbf{x}_i \cdot \mathbf{x}_j)$ 의 값은 1에 가깝고, 유사성이 낮다면 0에 가까운 값이 된다.
- 따라서 유사성이 낮은 경우는 목적함수에 별다른 영향을 주지 못하고, 유사성이 높은 경우에는 상대적으로 큰 영향력을 갖는다.
- 유사성이 높으면 $y_i y_j = 1$ 이라면 목적함수의 크기를 감소시키고 따라서 최적화과정에서 해당 관찰값의 기여도를 줄인다.
- 반대로 $y_i y_j = -1$ 이라면 목적함수를 크게 하므로 기여도를 증가시킨다. 실제 이러한 관찰값들은 정보 측면에서 더 큰 가치를 갖는다.
- 이를 정리하면 $a_i a_j$ 는 길을 바꾸기 위해 감수해야하는 '비용'을, $(\mathbf{x}_i \cdot \mathbf{x}_j)$ 는 유사성을, $y_i y_j$ 는 길을 넓이는데 필요한 추가 정보의 가치를 나타낸다.

Contents ☰

- ▼ 1 SVM의 분류방법
 - 1.1 개략적인 모형
 - 1.2 최적화 문제
- ▼ 2 최적화 문제와 support vector의 식별
 - 2.1 Hard margin SVM
 - 2.2 Classification
 - 2.3 Soft margin SVM
 - 2.4 선형모형과 SVM
 - 2.5 Dual problem
 - 2.6 dot product 와 유사성 similarity
 - 2.7 sklearn.svm.SVC
 - 2.8 전처리 및 hyperparameter
 - 2.9 Multiclass SVM
- ▼ 3 Feature space의 확장과 nonlinear decision function
 - 3.1 Kernel trick
 - 3.2 Dual problem
 - ▼ 3.3 Kernel function, positive-definite function
 - 3.3.1 Kernel 함수의 예
 - 3.3.2 Kernel 함수
 - 3.4 Kernels을 이용한 SVM
- ▼ 4 Support Vector Machine의 특징
 - 4.1 장점
 - 4.2 단점
 - 4.3 분석
- ▼ 5 SVR, support vector regression
 - 5.1 Hard margin
 - 5.2 Soft margin
 - 5.3 Prediction

Figure 3. Contribution of Observations



- Figure 3에서는 support vector x_i 와 다른 관찰값들을 비교하였다. 최적점에선 (1)과 (3)을 제외하고 $a_j^* = 0$ 이다.

- (1)의 두 벡터는 아주 유사하며 다른 class에 속하므로 margin width를 넓게 하는데 중요한 역할을 한다.
- (2)의 두 벡터 사이의 유사성은 높지만 같은 class에 속하므로 추가 정보를 제공하지 않는다.
- (3) (1)보다 유사성은 다소 떨어지지만 다른 class에 속하므로 의미있는 정보를 포함하지만 (1)보다는 중요성이 떨어진다.
- (4) 유사성이 전혀 없고 같은 class에 속하므로 가치가 없는 표본이다. 사잇각이 90도 이상이면 $a_j^* = 0$ 이므로 고려하지 않는다.

Contents ⚙

- ▼ 1 SVM의 분류방법
 - 1.1 개략적인 모형
 - 1.2 최적화 문제
- ▼ 2 최적화 문제와 support vector의 식별
 - 2.1 Hard margin SVM
 - 2.2 Classification
 - 2.3 Soft margin SVM
 - 2.4 선형모형과 SVM
 - 2.5 Dual problem
 - 2.6 dot product 와 유사성 similarity
 - 2.7 sklearn.svm.SVC
 - 2.8 전처리 및 hyperparameter
 - 2.9 Multiclass SVM
- ▼ 3 Feature space의 확장과 nonlinear decision function
 - 3.1 Kernel trick
 - 3.2 Dual problem
 - ▼ 3.3 Kernel function, positive-definite function
 - 3.3.1 Kernel 함수의 예
 - 3.3.2 Kernel 함수
 - 3.4 Kernels을 이용한 SVM
- ▼ 4 Support Vector Machine의 특징
 - 4.1 장점
 - 4.2 단점
 - 4.3 분석
- ▼ 5 SVR, support vector regression
 - 5.1 Hard margin
 - 5.2 Soft margin
 - 5.3 Prediction

2.7 sklearn.svm.SVC

```
sklearn.svm.SVC(*, C=1.0, kernel='rbf', degree=3, gamma='scale', coef0=0.0, shrinking=True, probability=False, tol=0.001, cache_size=200, class_weight=None, verbose=False, max_iter=-1, decision_function_shape='ovr', break_ties=False, random_state=None)
```

- C: float, default=1.0
Regularization parameter. The strength of the regularization is inversely proportional to C. Must be strictly positive. The penalty is a squared l2 penalty.
Soft margin SVM의 penalty에 해당하는 값으로 C값이 작을수록 hard margin에 가까워진다.
- kernel: {'linear', 'poly', 'rbf', 'sigmoid', 'precomputed'}, default='rbf'
Specifies the kernel type to be used in the algorithm. It must be one of 'linear', 'poly', 'rbf', 'sigmoid', 'precomputed' or a callable. If none is given, 'rbf' will be used. If a callable is given it is used to pre-compute the kernel matrix from data matrices; that matrix should be an array of shape (n_samples, n_samples)
kernel 종류에 따라 degree , gamma , coef0 와 같은 parameter값을 조정해 주어야 한다.
- decision_function_shape: {'ovo', 'ovr'}, default='ovr'

2.8 전처리 및 hyperparameter

- C의 역수는 regularization strength로 예측오차의 크기에 대한 penalty의 크기이다.
- 예측오차의 크기는 supporting hyperplane과의 거리이지만 최적화과정에서 정규화한 \mathbf{w} 를 구하므로 표본의 정규화는 필수적이진 않다.
- 정규화를 하지 않으면 연산속도가 느려지고, C 값에 따라 weight matrix의 값이 크게 달라지므로 SVC에서는 feature를 정규화하는 것이 일반적이다.

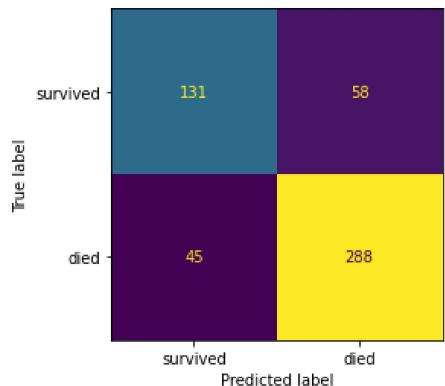
Contents ☰

- ▼ 1 SVM의 분류방법
 - 1.1 개략적인 모형
 - 1.2 최적화 문제
- ▼ 2 최적화 문제와 support vector의 식별
 - 2.1 Hard margin SVM
 - 2.2 Classification
 - 2.3 Soft margin SVM
 - 2.4 선형모형과 SVM
 - 2.5 Dual problem
 - 2.6 dot product 와 유사성 similarity
 - 2.7 sklearn.svm.SVC
 - 2.8 전처리 및 hyperparameter
 - 2.9 Multiclass SVM
- ▼ 3 Feature space의 확장과 nonlinear decision function
 - 3.1 Kernel trick
 - 3.2 Dual problem
 - ▼ 3.3 Kernel function, positive-definite function
 - 3.3.1 Kernel 함수의 예
 - 3.3.2 Kernel 함수
 - 3.4 Kernels을 이용한 SVM
- ▼ 4 Support Vector Machine의 특징
 - 4.1 장점
 - 4.2 단점
 - 4.3 분석
- ▼ 5 SVR, support vector regression
 - 5.1 Hard margin
 - 5.2 Soft margin
 - 5.3 Prediction

precision recall f1-score support

survived	0.74	0.69	0.72	189
died	0.83	0.86	0.85	333

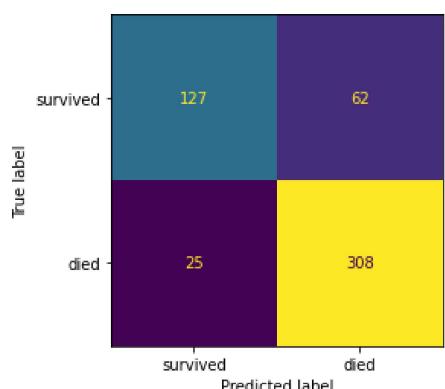
accuracy			0.80	522
macro avg	0.79	0.78	0.78	522
weighted avg	0.80	0.80	0.80	522



precision recall f1-score support

survived	0.84	0.67	0.74	189
died	0.83	0.92	0.88	333

accuracy			0.83	522
macro avg	0.83	0.80	0.81	522
weighted avg	0.83	0.83	0.83	522



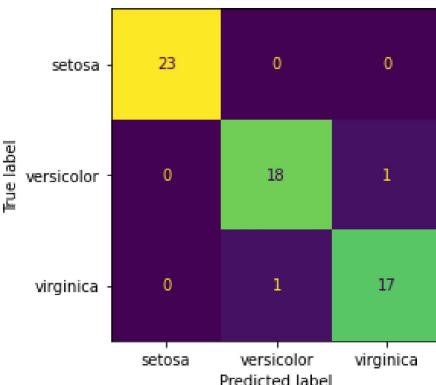
Contents ⚙

- ▼ 1 SVM의 분류방법
 - 1.1 개략적인 모형
 - 1.2 최적화 문제
- ▼ 2 최적화 문제와 support vector의 식별
 - 2.1 Hard margin SVM
 - 2.2 Classification
 - 2.3 Soft margin SVM
 - 2.4 선형모형과 SVM
 - 2.5 Dual problem
 - 2.6 dot product 와 유사성 similarity
 - 2.7 sklearn.svm.SVC
 - 2.8 전처리 및 hyperparameter
 - 2.9 Multiclass SVM
- ▼ 3 Feature space의 확장과 nonlinear decision boundary
 - 3.1 Kernel trick
 - 3.2 Dual problem
 - ▼ 3.3 Kernel function, positive-definite function
 - 3.3.1 Kernel 함수의 예
 - 3.3.2 Kernel 함수
 - 3.4 Kernels을 이용한 SVM
- ▼ 4 Support Vector Machine의 특징
 - 4.1 장점
 - 4.2 단점
 - 4.3 분석
- ▼ 5 SVR, support vector regression
 - 5.1 Hard margin
 - 5.2 Soft margin
 - 5.3 Prediction

2.9 Multiclass SVM

- SVM은 기본적으로 두 유형 문제에 적용한다.
- 다유형 문제에 적용할 때는 multinomial logistic regression과 다르게 one-vs-rest로 학습시킨 뒤, supporting hyperplane을 기준으로 가장 깊숙히 자리한 유형을 선택하는 것이 일반적이다.
- Sklearn에서 keyword `decision_function_shape`의 default이며, 'ovo'를 선택할 수 있다.

	precision	recall	f1-score	support
setosa	1.00	1.00	1.00	23
versicolor	0.95	0.95	0.95	19
virginica	0.94	0.94	0.94	18
accuracy			0.97	60
macro avg	0.96	0.96	0.96	60
weighted avg	0.97	0.97	0.97	60



3 Feature space의 확장과 nonlinear decision boundary

- 눈에 띄는 차이가 있지만 선형으로 분리가 '어려운' 자료라면 자료의 비선형변환으로 효과적인 분류가 가능할 수 있다.
- 자료의 비선형변환이나 비선형 decision boundary 모두 유사한 결과를 얻을 수 있지만 몇 가지 차이가 있다.
- 비선형 decision boundary를 사용할 수 있지만 변수의 수가 많으면 자유도가 높아 함수의 형태를 결정하기 쉽지 않다.
- 비선형변환은 일반적으로 계산량이 기하급수적으로 증가시키는 경향이 있지만, kernel trick을 적용할 수 있다면 선형분류기의 특징인 계산상의 장점을 유지할 수 있다.
- Kernel trick은 표본공간에 제한을 가해 암묵적인 방식으로 curse of dimensionality를 완화하는데 효과적이다.

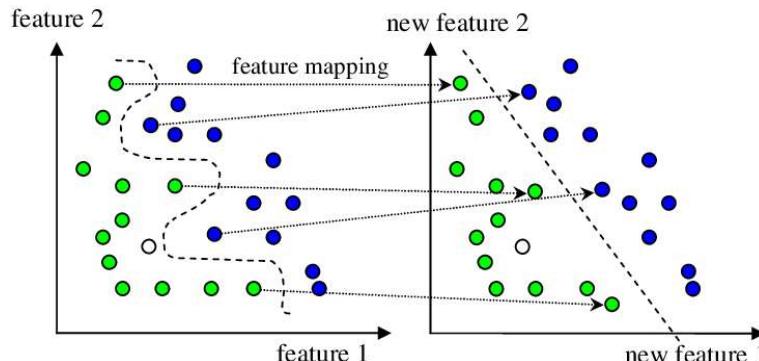
Contents ☰

- ▼ 1 SVM의 분류방법
 - 1.1 개략적인 모형
 - 1.2 최적화 문제
- ▼ 2 최적화 문제와 support vector의 식별
 - 2.1 Hard margin SVM
 - 2.2 Classification
 - 2.3 Soft margin SVM
 - 2.4 선형모형과 SVM
 - 2.5 Dual problem
 - 2.6 dot product 와 유사성 similarity
 - 2.7 sklearn.svm.SVC
 - 2.8 전처리 및 hyperparameter
 - 2.9 Multiclass SVM
- ▼ 3 Feature space의 확장과 nonlinear decision boundary
 - 3.1 Kernel trick
 - 3.2 Dual problem
 - ▼ 3.3 Kernel function, positive-definite function
 - 3.3.1 Kernel 함수의 예
 - 3.3.2 Kernel 함수
 - 3.4 Kernels을 이용한 SVM
- ▼ 4 Support Vector Machine의 특징
 - 4.1 장점
 - 4.2 단점
 - 4.3 분석
- ▼ 5 SVR, support vector regression
 - 5.1 Hard margin
 - 5.2 Soft margin
 - 5.3 Prediction

3.1 Kernel trick

- Feature space의 차원이 높으면 높을수록 분류에는 유리하다. 특히 선형분류기의 경우 그 효과는 더 극적일 수 있다.
 - 모든 n 개의 구분되는 labeled points에 대해 선형으로 완벽한 분리가 가능한 feature transformation이 존재한다.
- Input space의 vector를 더 높은 차원의 feature space에 대응시키면서 관찰값들 사이의 유사성을 유지할 수 있다면 표본의 특성을 다양한 측면에서 분석에 고려할 수 있다.
- Feature space의 차원을 크게 하기 위해서는 비선형변환이 필수적이고 따라서 몇 가지 부작용을 수반한다.
 - 연산시간과 memory요구량이 크게 증가할 수 있다. Gaussian kernel tranformation 같은 경우 feature space의 차원이 무한하다.
 - 개별 input, x_k 의 영향을 식별하기 어려워진다.
- 이런 특징들은 feature space를 극단적으로 증가시킨 neural network의 '한계'와 유사하지만 그 과정은 상당히 다르다.
- Kernel trick의 비선형변환은 개별 input에 적용하지만 neural network은 일반적으로 선형결합에 대해 변환을 적용한다.
- 이상 언급한 문제 중 연산시간 문제를 해결하기 위해 machine learning에서 흔히 사용하는 방법중 하나가 kernel function을 이용한 kernel trick이다.

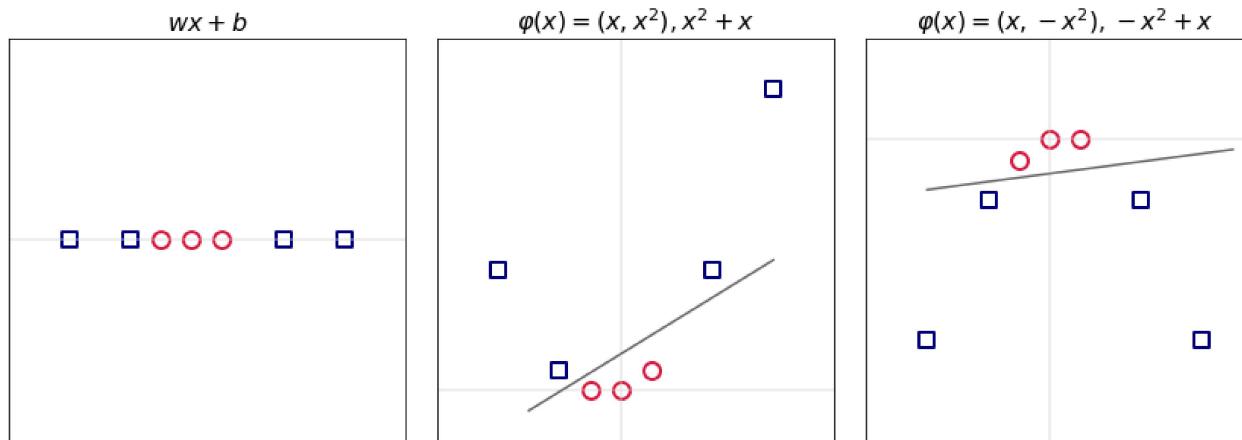
Figure 4. Nonlinear Decision Boundary vs. Feature Transformation



Contents ☰

- ▼ 1 SVM의 분류방법
 - 1.1 개략적인 모형
 - 1.2 최적화 문제
- ▼ 2 최적화 문제와 support vector의 식별
 - 2.1 Hard margin SVM
 - 2.2 Classification
 - 2.3 Soft margin SVM
 - 2.4 선형모형과 SVM
 - 2.5 Dual problem
 - 2.6 dot product 와 유사성 similarity
 - 2.7 sklearn.svm.SVC
 - 2.8 전처리 및 hyperparameter
 - 2.9 Multiclass SVM
- ▼ 3 Feature space의 확장과 nonlinear decision function
 - 3.1 Kernel trick
 - 3.2 Dual problem
 - ▼ 3.3 Kernel function, positive-definite function
 - 3.3.1 Kernel 함수의 예
 - 3.3.2 Kernel 함수
 - 3.4 Kernels을 이용한 SVM
- ▼ 4 Support Vector Machine의 특징
 - 4.1 장점
 - 4.2 단점
 - 4.3 분석
- ▼ 5 SVR, support vector regression
 - 5.1 Hard margin
 - 5.2 Soft margin
 - 5.3 Prediction

Figure 5. Nonlinear Mapping



3.2 Dual problem

- Dual 문제의 목적함수에는 관찰값이 $(\mathbf{x}_i \cdot \mathbf{x}_j)$ 형태로 포함되어 있다.

$$\max_{a \geq 0} \mathcal{L}_{\text{Dual}}(a) = -\frac{1}{2} \sum_{i=1}^n \sum_{j=1}^n a_i a_j y_i y_j (\mathbf{x}_i \cdot \mathbf{x}_j) + \sum_{i=1}^n a_i$$

$$y^{\text{pred}}(\mathbf{x}_j) = \text{sign} \left[\sum_{i=1}^n a_i^* y_i (\mathbf{x}_i \cdot \mathbf{x}_j) + b^* \right]$$

- 자료의 정보는 모두 dot product의 형태로만 모형에 반영된다. Kernel trick은 이런 구조를 갖는 모든 algorithm에 적용이 가능하다.
- 따라서 자료를 동일한 dot product 형태로 유지하면서 연산횟수를 크게 증가시키지 않을 수 있다면 curse of dimensionality 문제를 효과적으로 해결할 수 있다.

$$\max_{a \geq 0} \mathcal{L}_{\text{Dual}}(a) = -\frac{1}{2} \sum_{i=1}^n \sum_{j=1}^n a_i a_j y_i y_j \underbrace{(\mathbf{x}_i \cdot \mathbf{x}_j)}_{k(\mathbf{x}_i \cdot \mathbf{x}_j) = \phi(\mathbf{x}_i) \cdot \phi(\mathbf{x}_j)} + \sum_{i=1}^n a_i$$

Contents ↗

- ▼ 1 SVM의 분류방법
 - 1.1 개략적인 모형
 - 1.2 최적화 문제
- ▼ 2 최적화 문제와 support vector의 식별
 - 2.1 Hard margin SVM
 - 2.2 Classification
 - 2.3 Soft margin SVM
 - 2.4 선형모형과 SVM
 - 2.5 Dual problem
 - 2.6 dot product 와 유사성 similarity
 - 2.7 sklearn.svm.SVC
 - 2.8 전처리 및 hyperparameter
 - 2.9 Multiclass SVM
- ▼ 3 Feature space의 확장과 nonlinear decision function
 - 3.1 Kernel trick
 - 3.2 Dual problem
 - ▼ 3.3 Kernel function, positive-definite function
 - 3.3.1 Kernel 함수의 예
 - 3.3.2 Kernel 함수
 - 3.4 Kernels을 이용한 SVM
- ▼ 4 Support Vector Machine의 특징
 - 4.1 장점
 - 4.2 단점
 - 4.3 분석
- ▼ 5 SVR, support vector regression
 - 5.1 Hard margin
 - 5.2 Soft margin
 - 5.3 Prediction

3.3 Kernel function, positive-definite function, similarity function, feature map

- Kernel 함수는 두 개의 vector를 다른 공간의 inner product로 mapping하는 함수이다. 즉, $\mathbf{x} \in \mathbb{R}^K$ 에 대해 다음과 같은 성질을 갖는다.

$$k : \mathbf{x} \times \mathbf{x} \rightarrow \varphi(\mathbf{x}) \cdot \varphi(\mathbf{x}), \quad \varphi : \mathbb{R}^K \rightarrow \mathbb{R}^d$$

- Kernel 함수를 이용하면 feature map φ 의 구체적인 형태나 대응 공간을 명시적으로 고려할 필요가 없다.
- Kernel 함수로 사용이 가능한 함수는 Mercer's condition으로 확인할 수 있다.

Definition 대칭적이고 연속인 함수 $k : [a, b] \times [a, b] \rightarrow \mathbb{R}$ 이 주어진 (c_1, \dots, c_n) 과 input의 모든 원소 $\mathbf{x}_i, \mathbf{x}_j$ 에 대해 다음 조건을 만족하면 non-negative definite이라고 한다.

$$\sum_{i=1}^n \sum_{j=1}^n k(\mathbf{x}_i, \mathbf{x}_j) c_i c_j \geq 0$$

- 실제 응용에선 similarity의 measure로 사용할 수 있는 함수라면 유도과정에서 요구하는 조건과는 무관하게 사용하기도 하며 이 역시 kernel로 부른다.

3.3.1 Kernel 함수의 예

- $k(x, z) = (x \cdot z)^2$ 에 해당하는 값을 얻기 위해 자료를 변환한다고 생각해 보자.

$$\begin{aligned} k(x, z) &= (x \cdot z)^2 = (x_1 \times x_2 + z_1 \times z_2)^2 \\ &= x_1^2 z_1^2 + 2x_1 z_1 x_2 z_2 + x_2^2 z_2^2 \\ &= (\sqrt{2}x_1 x_2, x_1^2, x_2^2) \cdot (\sqrt{2}z_1 z_2, z_1^2, z_2^2) \\ &= \varphi(x) \cdot \varphi(z) \end{aligned}$$

이므로 x 를 $(x_1 x_2, x_1^2, x_2^2)$ 로 변환해야 한다. 따라서 x 와 z 에 대해 각각 3번씩의 곱셈과 inner product를 계산하기 위해 추가로 3번의 곱셈에 2번의 덧셈을 해야 한다. 총 11번의 곱셈과 덧셈을 해야 한다.

- 반면 $(x \cdot z)^2$ 의 계산에는 3번의 곱셈과 한번의 덧셈이면 충분하다.
- 다른 예

$$\begin{aligned} \varphi(x_1, x_2) &= (x_1 x_2, x_1^2, x_2^2) \\ k(x, z) &= x \cdot z + \|x\|^2 \|z\|^2 \end{aligned}$$

Contents

- ▼ 1 SVM의 분류방법
 - 1.1 개략적인 모형
 - 1.2 최적화 문제
- ▼ 2 최적화 문제와 support vector의 식별
 - 2.1 Hard margin SVM
 - 2.2 Classification
 - 2.3 Soft margin SVM
 - 2.4 선형모형과 SVM
 - 2.5 Dual problem
 - 2.6 dot product 와 유사성 similarity
 - 2.7 sklearn.svm.SVC
 - 2.8 전처리 및 hyperparameter
 - 2.9 Multiclass SVM
- ▼ 3 Feature space의 확장과 nonlinear decision boundary
 - 3.1 Kernel trick
 - 3.2 Dual problem
 - ▼ 3.3 Kernel function, positive-definite function
 - 3.3.1 Kernel 함수의 예
 - 3.3.2 Kernel 함수
 - 3.4 Kernels을 이용한 SVM
- ▼ 4 Support Vector Machine의 특징
 - 4.1 장점
 - 4.2 단점
 - 4.3 분석
- ▼ 5 SVR, support vector regression
 - 5.1 Hard margin
 - 5.2 Soft margin
 - 5.3 Prediction

3.3.2 Kernel 함수

- Linear kernels은 단순한 내적

$$k(x, z) = x \cdot z$$

- Polynomial kernel은 p 차까지의 항을 포함

$$k(x, z) = (1 + x \cdot z)^p$$

- Radial basis function(RBF) kernel/Gaussian kernel은 차수가 무한인 다항식

$$k(x, z) = \exp\left(-\frac{(x - z)^\top(x - z)}{2\sigma^2}\right) \text{ for } \sigma > 0, \quad \exp(ax \cdot z) = \frac{\sum_{i=0}^{\infty} \alpha^i (x \cdot z)^i}{i!}$$

- Sigmoid kernel은 logistic function과 유사

$$k(x, z) = \tanh(\beta_0 x^\top z + \beta_1)$$

3.4 Kernels을 이용한 SVM

$$\begin{aligned} \max_{a \geq 0} & -\frac{1}{2} \sum_{i=1}^n \sum_{j=1}^n a_i a_j y_i y_j k(\mathbf{x}_i, \mathbf{x}_j) + \sum_{i=1}^n a_i \\ \text{subject to} & \quad k(\mathbf{x}_i, \mathbf{x}_j) = \varphi(\mathbf{x}_i) \cdot \varphi(\mathbf{x}_j) \\ & \sum_{i=1}^n a_i y_i = 0 \\ & \sum_{i=1}^n a_i \geq 0 \end{aligned}$$

- 분류는 다음과 같이 결정된다.

$$y^{\text{pred}} = \text{sign} \left[\sum_{i=1}^n a_i y_i k(\mathbf{x}, \mathbf{x}_i) + b \right]$$

여기서 b 는 두 번째 등호제약에 해당하는 Lagrange multiplier이다. 분류 역시 support vector의 정보만 이용한다.

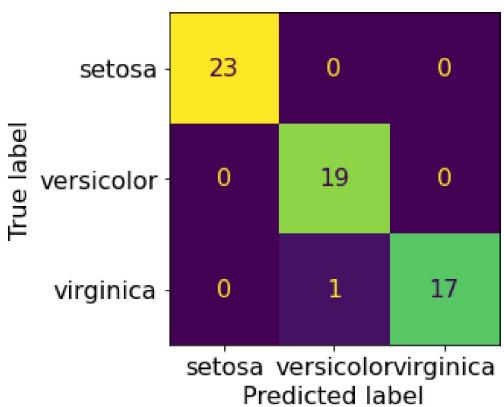
- 유도과정은 이곳 [노트](http://www.stat.cmu.edu/~larry/sml/SVM.pdf) (<http://www.stat.cmu.edu/~larry/sml/SVM.pdf>)를 참고

Contents ☰

- ▼ 1 SVM의 분류방법
 - 1.1 개략적인 모형
 - 1.2 최적화 문제
- ▼ 2 최적화 문제와 support vector의 식별
 - 2.1 Hard margin SVM
 - 2.2 Classification
 - 2.3 Soft margin SVM
 - 2.4 선형모형과 SVM
 - 2.5 Dual problem
 - 2.6 dot product 와 유사성 similarity
 - 2.7 sklearn.svm.SVC
 - 2.8 전처리 및 hyperparameter
 - 2.9 Multiclass SVM
- ▼ 3 Feature space의 확장과 nonlinear decision function
 - 3.1 Kernel trick
 - 3.2 Dual problem
 - ▼ 3.3 Kernel function, positive-definite function
 - 3.3.1 Kernel 함수의 예
 - 3.3.2 Kernel 함수
 - 3.4 Kernels을 이용한 SVM
- ▼ 4 Support Vector Machine의 특징
 - 4.1 장점
 - 4.2 단점
 - 4.3 분석
- ▼ 5 SVR, support vector regression
 - 5.1 Hard margin
 - 5.2 Soft margin
 - 5.3 Prediction

precision recall f1-score support

setosa	1.00	1.00	1.00	23
versicolor	0.95	1.00	0.97	19
virginica	1.00	0.94	0.97	18
accuracy			0.98	60
macro avg	0.98	0.98	0.98	60
weighted avg	0.98	0.98	0.98	60



Contents ☰

- ▼ 1 SVM의 분류방법
 - 1.1 개략적인 모형
 - 1.2 최적화 문제
- ▼ 2 최적화 문제와 support vector의 식별
 - 2.1 Hard margin SVM
 - 2.2 Classification
 - 2.3 Soft margin SVM
 - 2.4 선형모형과 SVM
 - 2.5 Dual problem
 - 2.6 dot product 와 유사성 similarity
 - 2.7 sklearn.svm.SVC
 - 2.8 전처리 및 hyperparameter
 - 2.9 Multiclass SVM
- ▼ 3 Feature space의 확장과 nonlinear decision function
 - 3.1 Kernel trick
 - 3.2 Dual problem
 - ▼ 3.3 Kernel function, positive-definite function
 - 3.3.1 Kernel 함수의 예
 - 3.3.2 Kernel 함수
 - 3.4 Kernels을 이용한 SVM
- ▼ 4 Support Vector Machine의 특징
 - 4.1 장점
 - 4.2 단점
 - 4.3 분석
- ▼ 5 SVR, support vector regression
 - 5.1 Hard margin
 - 5.2 Soft margin
 - 5.3 Prediction

4 Support Vector Machine의 특징

4.1 장점

- 탄탄한 이론에 근거한 모형으로 일반화가 쉬우며 목적함수는 "local minima"를 갖지 않는다.
- Feature의 수가 많은 text나 image 등의 분류에 효율적이다.
 - 모형의 복잡도(계산 분량)는 feature의 수보다는 표본수가 더 크게 좌우한다. $\mathcal{O}(n^2)$
- Classifier는 표본의 일부만을 사용하므로 decision region의 안쪽에 위치한 outlier들의 영향을 받지 않는다.
- 다중공선성에서 자유롭다.
- Weight vector를 이용하여 feature selection에 이용할 수 있다.
- Regularization을 통해 (soft margin svm) overfitting 문제를 쉽게 통제할 수 있다.
- 완벽한 분리가 가능한 자료에도 적용할 수 있다.
- 작은 표본에서 다른 모형들에 비해 상대적 성능이 좋다.

4.2 단점

- 상대적으로 계산시간이 많이 걸리며 대형자료처리에는 상당한 계산시간이 필요
- 예측에 대한 확률을 추정하지 않는다.
 - probability=True로 지정하면 predict_proba와 predict_log_proba를 사용할 수 있지만 믿을만하지 못하다. 참고 (<https://scikit-learn.org/stable/modules/svm.html#scores-and-probabilities>)
- 선형으로 분리가 가능한 경우 logistic regression과 별다른 차이가 없다.
- Class의 관찰값들이 많이 중첩되어 있을 때는 효율성이 떨어진다.

4.3 분석

- 분류기의 복잡성은 support vector의 수에 의해 결정되므로 모형을 선택할 때는 n_support_ attribute 등을 참고
- Feature의 분포에 따라 적합한 rescaling은 필수
- RBF kernel을 가장 많이 사용하지만 feature의 수가 많을 경우에는 보통 linear kernel로 시작한다.

예제

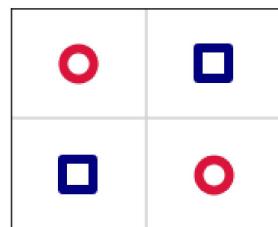
- XOR 문제는 두 변수의 논리연산에서 정확히 둘 중 하나가 참이고 다른 하나가 거짓일 경우 참인 표본의 분류문제이다.
- Deep neural network으로 쉽게 해결이 가능하지만 이 이전엔 오래동안 해결이 되지 않은 분류 상의 난제 중의 하나였다.
- 적당한 kernel을 고르는 것이 관건이다.

```
features = np.array([[1, 1], [-1, -1], [-1, 1], [1, -1]])
targets = ["square", "square", "circle", "circle"]
```

Contents ⚙

- ▼ 1 SVM의 분류방법
 - 1.1 개략적인 모형
 - 1.2 최적화 문제
- ▼ 2 최적화 문제와 support vector의 식별
 - 2.1 Hard margin SVM
 - 2.2 Classification
 - 2.3 Soft margin SVM
 - 2.4 선형모형과 SVM
 - 2.5 Dual problem
 - 2.6 dot product 와 유사성 similarity
 - 2.7 sklearn.svm.SVC
 - 2.8 전처리 및 hyperparameter
 - 2.9 Multiclass SVM
- ▼ 3 Feature space의 확장과 nonlinear decision function
 - 3.1 Kernel trick
 - 3.2 Dual problem
 - ▼ 3.3 Kernel function, positive-definite function
 - 3.3.1 Kernel 함수의 예
 - 3.3.2 Kernel 함수
 - 3.4 Kernels을 이용한 SVM
- ▼ 4 Support Vector Machine의 특징
 - 4.1 장점
 - 4.2 단점
 - 4.3 분석
- ▼ 5 SVR, support vector regression
 - 5.1 Hard margin
 - 5.2 Soft margin
 - 5.3 Prediction

Figure 5. XOR Problem



5 SVR, support vector regression

- Hyperplane을 기준으로 길의 폭을 사전에 지정하고 길에서 벗어난 관찰값에 penalty를 부과

5.1 Hard margin

$$\min_{\mathbf{w}, b} \frac{\|\mathbf{w}\|^2}{2}$$

subject to $|y_i - (\mathbf{w}^\top x_i + b)| \leq \varepsilon$

5.2 Soft margin

$$\min_{\mathbf{w}, b} \frac{\|\mathbf{w}\|^2}{2} + C \sum_{i=1}^n (\eta_i^+ + \eta_i^-)$$

subject to $\varepsilon - \eta_i^- \leq y_i - (\mathbf{w}^\top x_i + b)$
 $y_i - (\mathbf{w}^\top x_i + b) \leq \varepsilon + \eta_i^+$
 $\eta_i^+, \eta_i^- \geq 0$

5.3 Prediction

$$\hat{y}_j = \sum_{i=1}^n (\alpha_i^+ - \alpha_i^-)(x_i \cdot x_j) + b$$

Contents ☰

- ▼ 1 SVM의 분류방법
 - 1.1 개략적인 모형
 - 1.2 최적화 문제
- ▼ 2 최적화 문제와 support vector의 식별
 - 2.1 Hard margin SVM
 - 2.2 Classification
 - 2.3 Soft margin SVM
 - 2.4 선형모형과 SVM
 - 2.5 Dual problem
 - 2.6 dot product 와 유사성 similarity
 - 2.7 sklearn.svm.SVC
 - 2.8 전처리 및 hyperparameter
 - 2.9 Multiclass SVM
- ▼ 3 Feature space의 확장과 nonlinear decision function
 - 3.1 Kernel trick
 - 3.2 Dual problem
 - ▼ 3.3 Kernel function, positive-definite function
 - 3.3.1 Kernel 함수의 예
 - 3.3.2 Kernel 함수
 - 3.4 Kernels을 이용한 SVM
- ▼ 4 Support Vector Machine의 특징
 - 4.1 장점
 - 4.2 단점
 - 4.3 분석
- ▼ 5 SVR, support vector regression
 - 5.1 Hard margin
 - 5.2 Soft margin
 - 5.3 Prediction

```
sklearn.svm.SVR(*, kernel='rbf', degree=3, gamma='scale', coef0=0.0, tol=0.001, C=1.0, epsilon=0.1, shrinking=True, cache_size=200, verbose=False, max_iter=-1)
```

- 표본이 10,000개 이상이라면 훈련이 어려울수도 있다.

The implementation is based on libsvm. The fit time complexity is more than quadratic with the number of samples which makes it hard to scale to datasets with more than a couple of 10000 samples. For large datasets consider using LinearSVR or SGDRegressor instead, possibly after a Nystroem transformer.