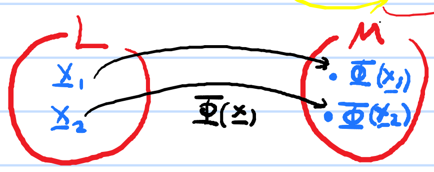
Kernel Function (커널 함수)

- 원래의 특징공간 L에서 정의된 두 특징벡터 $\underline{x}_1, \underline{x}_2$에 대해 어떤 함수 $K(\underline{x}_1, \underline{x}_2)$가 다음을 만족한다면, $K(\underline{x}_1, \underline{x}_2)$를 커널함수라고 부른다.

$$\Rightarrow K(\underline{x}_1, \underline{x}_2) = \Phi(\underline{x}_1)\cdot\Phi(\underline{x}_2)$$

vector



* 차원이 커지거나 무한할 수 있음 (용이 차라)

이를 만족하는 같은 K더라도 여러개의 Φ가 있을 수 있음.

- 실용적으로 중요한 것은 Kernel Function K를 찾는 것이고, 변환함수 Φ는 알지 못해도 됨.
- 하나의 Kernel Function K에 대응되는 서로 다른 변환함수 Φ가 존재할 수 있음.

– Kernel Trick (딥러닝 알고리즘에서 가장 근본적인 기초가 됨!)
  - $\Phi(\underline{x}) \in H$인 공간 H에 대하여, H공간에서의 내적 연산 $(\Phi(\underline{x}_1)\cdot\Phi(\underline{x}_2))$을 원래 특징공간 L에서의 Kernel Function 연산으로 대체하는 것.
  - 특징 벡터 $\underline{x}$를 명시적으로 H로 변환하지 않으면서 새로운 공간의 좋은 특성을 활용하는 효과를 얻음. ex) 선형분리가능

    Φ를 사용하지 않음

  - Kernel substitution (커널 대치)

많이 쓰이는 Kernel Function의 예시.

- Polynomial Kernel

$$\Rightarrow K(\underline{x}_1, \underline{x}_2) = (\underline{x}_1\cdot\underline{x}_2 + 1)^p$$

다항식 형태

- Radial Basis Function (RBF) kernel

$$\Rightarrow K(\underline{x}_1, \underline{x}_2) = \exp\left(\frac{-\|\underline{x}_1 - \underline{x}_2\|_2^2}{2\sigma^2}\right)$$

오일러 함수

- Hyperbolic Tangent Kernel.

$$\Rightarrow K(\underline{x}_1, \underline{x}_2) = \tanh(\alpha\underline{x}_1\cdot\underline{x}_2 + \beta)$$

tanh

* Kernel Function 자체만으로는 사용하는 변환함수 Φ없이는 새로운 특징공간 H에 대한 정보가 없음. (But, ok)

SVM Recap.

- Cost function

$$C(\underline{\alpha}) = \sum_{i=1}^{n}\alpha_i - \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_i\alpha_j y_i y_j \underline{x}_i\cdot\underline{x}_j$$

vector의 내적

Kernel Func. 사용시,

$$\Phi(\underline{x}_i)\cdot\Phi(\underline{x}_j) = \boxed{K(\underline{x}_i, \underline{x}_j)}$$

- Constraints.

$$\sum_{i=1}^{n}\alpha_i y_i = 0, \quad 0 \le \alpha_i \le C, \quad i = 1, \cdots, n$$

* Kernel SVM은, 선형분리불가능한 공간 L 대신 선형분리가능한 공간 H에서의 선형 SVM과 같음.

공간 내의, 주어진 Dataset.

Kernel SVM을 사용한 예측.

- 변환된 공간 H에서의 선형모델을 학습한 것이므로, 변환함수 Φ가 필요함.

$$d(\underline{x}) = \underline{w}^T\cdot\underline{x} + w_0 \qquad \Phi(\underline{x})$$

⇒ 예측시 Φ가 필요하기 때문에 의미X.

$$y_{pred} = \begin{cases} 1, & d(\underline{x}) > 0 \\ -1, & \text{otherwise} \end{cases}$$

- 예측모델을 적절히 변경하여, Φ 대신 K의 형태로 만들 수 있어야 함.

$$d(\underline{x}) = \underline{w}^T\cdot\Phi(\underline{x}) + w_0$$

$$= \Big(\sum_{\alpha_k\neq 0}\alpha_k y_k \Phi(\underline{x}_k)\Big)^T\Phi(\underline{x}) + w_0.$$

(SVM에서의 기본제약조건 에서의 식)

$$= \sum_{\alpha_k\neq 0}\alpha_k y_k \Phi(\underline{x}_k)\cdot\Phi(\underline{x}) + w_0.$$

$$= \sum_{\alpha_k\neq 0}\alpha_k y_k \boxed{K(\underline{x}_k, \underline{x})} + w_0.$$

- 예측모델에서 Kernel Func. 계산에 $\underline{x}_k$가 포함.

  예측시 Dataset을 다시 확인해야 함. (Memory가 많이 필요)

  → $\alpha_k \neq 0$인 sample들 (Support vectors)을 메모리에 저장하고 있어야 계산(예측)이 가능하다. (메모리 기반 방법) ⇒ 한계점

  → 비선형 SVM의 예측에 걸리는 시간은 Support vector의 수에 비례.