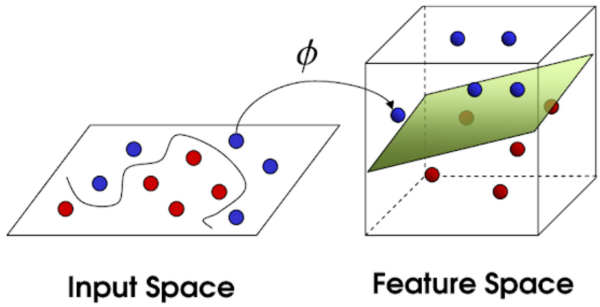
ESC 2024 Spring Session 1st Week

Kernel Methods

전인태, 정예준

Introduction : 아래와 같은 상황에서 classification을 수행하고 싶다고 하자.



왼쪽 Input Space에서는 linear model만으로는 classification을 수행할 수 없다. 그러나, 오른쪽 feature space에서는 linear model만으로도 해결이 가능한 것을 볼 수 있다. 이때, Input Space $\mathbf{X}$를 feature space $\phi(\mathbf{X})$로 Mapping 해주는 함수를 Basis function이라고 한다.

하지만, 실제 상황에서는 이런 Basis function을 직접 찾기가 매우 까다롭기 때문에, kernel을 사용하는데, kernel function은 다음과 같이 정의된다.

$$k(\mathbf{x_1}, \mathbf{x_2}) = \phi(\mathbf{x_1}^T)\phi(\mathbf{x_2})$$

ϕ를 정의하는 것에 따라 다양한 kernel을 만들 수 있으며, 필요에 따라 적절한 kernel을 선택해 사용하면 된다.

1. Dual Representations

아래 Ridge Regression의 Loss function을 떠올려보자.

$$J(\mathbf{w}) = \frac{1}{2}\sum_{n=1}^{N}\{\mathbf{w^T}\phi(\mathbf{x_n}) - t_n\}^2 + \frac{\lambda}{2}\mathbf{w^T w}$$

$a_n = -\frac{1}{\lambda}\{\mathbf{w^T}\phi(\mathbf{x_n}) - t_n\}$로 정의하고, $\mathbf{a} = (a_1, ..., a_N)^T$로 정의하자. 이렇게 하면, loss function을 원래의 parameter $\mathbf{w}$가 아닌, 새로운 parameter $\mathbf{a}$에 대한 식으로 표현할 수 있다. 같은 식을 서로 다른 parameter를 이용해 표현했으므로, 이것을 Dual Representation이라고 한다.

$$J(\mathbf{a}) = \frac{1}{2}\mathbf{a^T \Phi \Phi^T \Phi \Phi^T a} - \mathbf{a^T \Phi \Phi^T t} + \frac{1}{2}\mathbf{t^T t} + \frac{\lambda}{2}\mathbf{a^T \Phi \Phi^T a}$$

여기서, $\mathbf{\Phi}$는 n번째 행이 $\phi(\mathbf{x_n})^T$인 design matrix이다.

식을 조금 더 간단히 하기 위해 Gram matrix $\mathbf{K}$를 $\mathbf{K} = \mathbf{\Phi}^{\mathbf{T}}\mathbf{\Phi}$로 정의하면,

$$J(\mathbf{a}) = \frac{1}{2}\mathbf{a}^{\mathbf{T}}\mathbf{K}\mathbf{K}\mathbf{a} - \mathbf{a}^{\mathbf{T}}\mathbf{K}\mathbf{t} + \frac{1}{2}\mathbf{t}^{\mathbf{T}}\mathbf{t} + \frac{\lambda}{2}\mathbf{a}^{\mathbf{T}}\mathbf{K}\mathbf{a}$$

이 식을 $\mathbf{a}$에 대해 미분하고 =0으로 두어 풀면

$$\mathbf{a} = (\mathbf{K} + \lambda\mathbf{I}_N)^{-1}\mathbf{t}$$

다시 원래의 regression model로 돌아와 위에서 구한 값들을 대입하면,

$$y(\mathbf{x}) = \mathbf{w}^T\phi(\mathbf{x}) = \mathbf{a}^{\mathbf{T}}\mathbf{\Phi}\phi(\mathbf{x}) = \mathbf{k}(\mathbf{x})^{\mathbf{T}}(\mathbf{K} + \lambda\mathbf{I}_N)^{-1}\mathbf{t}$$

Dual Representation을 통해 나타내면, 원래 식과 달리 Basis function $\phi(\mathbf{x})$에 대한 식 대신 kernel function 에 대한 식으로 표현할 수 있게 되었다. 찾기 어려운 Basis function 대신 Kernel만을 이용해 식을 나타내는 것이 Dual Representation을 사용한 이유이다.

2. Constructing Kernels

아무 함수나 kernel로 사용해서는 안되고, kernel의 조건을 충족하는 식을 사용해야 한다. 이때, Valid한 Kernel의 조건은 Gram Matrix $\mathbf{K}$가 모든 가능한 $\{\mathbf{x_n}\}$에 대해 positive semi-definite한 것이다.
여기에 더해, valid한 두 kernel이 있다면, 아래의 kernel들도 모두 valid한 kernel이다.

Given valid kernels $k_1(\mathbf{x}, \mathbf{x}')$ and $k_2(\mathbf{x}, \mathbf{x}')$, the following new kernels will also be valid:

$$k(\mathbf{x}, \mathbf{x}') = ck_1(\mathbf{x}, \mathbf{x}') \quad (6.13)$$
$$k(\mathbf{x}, \mathbf{x}') = f(\mathbf{x})k_1(\mathbf{x}, \mathbf{x}')f(\mathbf{x}') \quad (6.14)$$
$$k(\mathbf{x}, \mathbf{x}') = q\left(k_1(\mathbf{x}, \mathbf{x}')\right) \quad (6.15)$$
$$k(\mathbf{x}, \mathbf{x}') = \exp\left(k_1(\mathbf{x}, \mathbf{x}')\right) \quad (6.16)$$
$$k(\mathbf{x}, \mathbf{x}') = k_1(\mathbf{x}, \mathbf{x}') + k_2(\mathbf{x}, \mathbf{x}') \quad (6.17)$$
$$k(\mathbf{x}, \mathbf{x}') = k_1(\mathbf{x}, \mathbf{x}')k_2(\mathbf{x}, \mathbf{x}') \quad (6.18)$$
$$k(\mathbf{x}, \mathbf{x}') = k_3\left(\boldsymbol{\phi}(\mathbf{x}), \boldsymbol{\phi}(\mathbf{x}')\right) \quad (6.19)$$
$$k(\mathbf{x}, \mathbf{x}') = \mathbf{x}^{\mathrm{T}}\mathbf{A}\mathbf{x}' \quad (6.20)$$
$$k(\mathbf{x}, \mathbf{x}') = k_a(\mathbf{x}_a, \mathbf{x}'_a) + k_b(\mathbf{x}_b, \mathbf{x}'_b) \quad (6.21)$$
$$k(\mathbf{x}, \mathbf{x}') = k_a(\mathbf{x}_a, \mathbf{x}'_a)k_b(\mathbf{x}_b, \mathbf{x}'_b) \quad (6.22)$$

where $c > 0$ is a constant, $f(\cdot)$ is any function, $q(\cdot)$ is a polynomial with nonnegative coefficients, $\boldsymbol{\phi}(\mathbf{x})$ is a function from $\mathbf{x}$ to $\mathbb{R}^M$, $k_3(\cdot, \cdot)$ is a valid kernel in $\mathbb{R}^M$, $\mathbf{A}$ is a symmetric positive semidefinite matrix, $\mathbf{x}_a$ and $\mathbf{x}_b$ are variables (not necessarily disjoint) with $\mathbf{x} = (\mathbf{x}_a, \mathbf{x}_b)$, and k_a and k_b are valid kernel functions over their respective spaces.

이를 이용해 이미 증명된 여러 kernel들이 있고, 우리는 필요에 따라 적절히 이들을 선택하여 사용하면 된다. 대표적으로 아래의 kernel들을 사용할 수 있다.

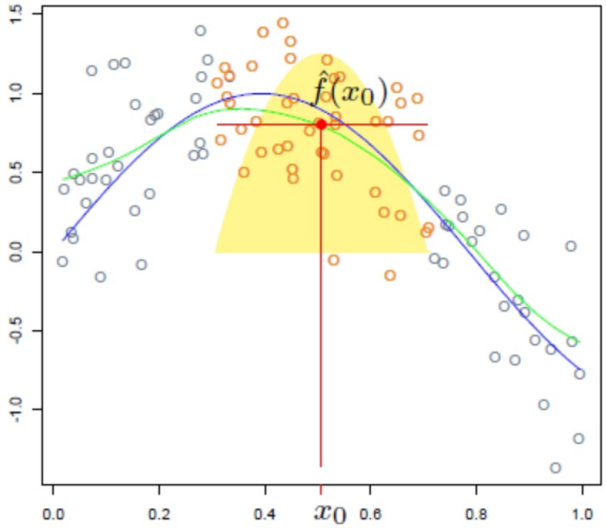
1. Polynomial Kernel : $k(\mathbf{x_1}, \mathbf{x_2}) = (\mathbf{x_1^T x_2})^2$
2. **Gaussian Kernel :** $k(\mathbf{x_1}, \mathbf{x_2}) = exp(-||\mathbf{x_1} - \mathbf{x_2}||^2/2\sigma^2)$
3. Sigmoid Kernel : $k(\mathbf{x_1}, \mathbf{x_2}) = tanh\{a(\mathbf{x_1^T x_2}) + b\}, \quad (a, b > 0)$

3. Radial basis function Networks

Radial basis function이란 중심 $\mu_{\mathbf{j}}$로부터의 거리에만 의존하는 Basis function을 말한다. 즉, $\phi_j(\mathbf{x}) = h(||\mathbf{x} - \mu_{\mathbf{j}}||)$ 이고, 함수 h로는 Gaussian 함수 등을 적용해 볼 수 있을 것이다. 이와 같은 Radial Basis Function을 이용하여 kernel을 만들면 어떤 중심으로부터 떨어진 거리를 측정할 수 있고, 이를 이용하여 kernel regression을 적용할 수 있다. kernel regression의 기본 아이디어는 예측하고자 하는 data point와 가까운 점들에게 더 큰 가중치를 주어 평균을 내겠다는 것이다. 가장 대표적으로 *Nadaraya-Watson* model이 있는데, 새로운 data point x_{new}에서의 fitted value는 아래와 같다.

$$\hat{f}(x_{new}) = \sum_{i=1}^{n} \frac{\mathbf{K}_\lambda(x_{new}, x_i)}{\sum_{i=1}^{n} \mathbf{K}_\lambda(x_{new}, x_i)} y_i$$

여기서, $\mathbf{K}_\lambda(x_{new}, x_i)$는 가까울수록 큰 값을 가지는 kernel로, 대표적으로 Gaussian kernel을 고려할 수 있다. 즉, x_{new}와 가까운 점일수록 더 큰 가중치를 가져 예측에 더 많은 영향을 주는 것을 확인할 수 있다.



weight가 연속함수이기 때문에 초록색 fitted lin e또한 부드러워지고, kernel의 표준편차 λ가 커질수록 더 멀리 떨어진 점에게도 가중치를 더 주기 때문에 fitted line이 더 평평해진다. 하지만, *Nadaraya-Watson* model을 포함한 kernel regression에는 한계가 있는데, 바로 한 point를 예측할 때마다 다른 모든 training data와의 거리를 계산하고 가중평균을 해야하기 때문에 연산량이 매우 많아지게 된다. 이것은 모든 비모수적인 방법이 공유하는 문제점이다.

4. Gaussian Process

평균함수와 공분산함수(커널)를 통해 임의의 변수들의 joint distribution이 정규분포를 따르도록 정의한 것을 가우시안 프로세스(Gaussian Process; GP)라고 한다. 예를 들어 $t \in \mathbb{R}$에 대해 $X(t) = tA,\ A \sim N(0,1)$라고 정의된 t에 대한 확률 변수의 함수는 GP이다. 그 이유는 다음과 같이 mean function과 kernel을 정의할 수 있으며, $X(t_i)$들의 joint distribution이 정규분포를 따르기 때문이다 :

$$\text{Mean} : m(t) = E(X(t)) = E(tA) = tE(A) = 0,$$
$$\text{Kernel} : Cov(X(t), X(s)) = Cov(tA, sA) = ts \cdot Var(A) = ts.$$

이제 GP를 이용하여 회귀 모델을 만들 수 있는데, 이러한 회귀 모델을 Gaussian Process Regression(GPR)라고 부른다. GPR을 이해하기 위하여, GPR을 Non-parametric Bayesian Regression Model이라고 생각할 수 있다. 이는 GPR이 회귀 모델의 random noise variable $t_n = y_n + \epsilon_n$(t : target variable, y : model prediction)의 y들의 벡터인 $\mathbf{y}$의 결합분포가 GP를 따른다는 가정을 하기 때문이다. 이것이 비모수적인 이유는, 회귀 모델의 매개변수인 $\mathbf{w}$에 대한 사전 분포(prior distribution)를 두는 것이 아닌, 함수 자체에 대한 사전 분포를 설정하기 때문이다. 우리가 회귀 모델에서 noise를 i.i.d.로 가정하듯, GPR에서는 noise를 isotropic하다고 가정한다. 이때 isotropic은 방향에 따라 값이 변하지 않으며 noise 간 독립적인, 즉, 공분산 행렬이 항등 행렬의 상수배로 나타난다 가정한다. 이에 따라 우리는 다음과 같은 분포를 얻는다 :

$$\begin{aligned}\text{likelihood} : &p(\mathbf{t}|\mathbf{y}) = N(\mathbf{t}|\mathbf{y}, \beta^{-1}\mathbf{I}_N),\\ \text{prior} : &p(\mathbf{y}) = N(\mathbf{y}|\mathbf{0}, \mathbf{K}).\end{aligned}$$

이때 $\mathbf{y}$의 공분산 행렬 $\mathbf{K}$는 그램 행렬로, $(\mathbf{K})_{ij} = k(x_i, x_j)$로 정의된다. (평균을 $\mathbf{0}$으로 가정하고, 여러 종류의 커널 중 이번 GPR을 위해 사용할 커널을 결정한 것 자체가 사전 가정이므로 함수 자체에 대한 prior라고 볼 수 있음.)

식 (2.115)에 값들을 대입함으로써 우리는 간단히

$$p(\mathbf{t}) = \int p(\mathbf{t}|\mathbf{y})p(\mathbf{y})d\mathbf{y} = N(\mathbf{y}|\mathbf{0}, \mathbf{C})$$

를 구할 수 있고, 이때 공분산 행렬 $\mathbf{C}$는 다음과 같이 정의된다 :

$$C(\mathbf{x}_n, \mathbf{x}_m) = k(\mathbf{x}_n, \mathbf{x}_m) + \beta^{-1}\delta_{nm}.$$

또한 이를 $N+1$개의 변수에 확장하면

$$p(\mathbf{t}_{N+1}) = N(\mathbf{t}_{N+1}|\mathbf{0}, \mathbf{C}_{N+1})$$

가 되며, 공분산 행렬 $\mathbf{C}_{N+1}$을 다음과 같이 partition하자 :

$$\mathbf{C}_{N+1} = \begin{pmatrix}\mathbf{C}_N & \mathbf{k}\\ \mathbf{k}^T & c\end{pmatrix}.$$

이때 공분산의 각 원소는 $(\mathbf{k})_i = k(\mathbf{x}_i, \mathbf{x}_{N+1})$, $c = k(\mathbf{x}_{N+1}, \mathbf{x}_{N+1}) + \beta^{-1}$이다.

이제 식 (2.81), (2.82)의 결과물을 활용하여 predictive distribution인 $p(t_{N+1}|\mathbf{t})$가 정규분포이며, 이때의 평균과 공분산이 다음과 같음을 알 수 있다 :

$$\begin{aligned}\text{Mean} : m(\mathbf{x}_{N+1}) &= \mathbf{k}^T\mathbf{C}_N^{-1}\mathbf{t},\\ \text{Cov} : Cov(\mathbf{x}_{N+1}) &= c - \mathbf{k}^T\mathbf{C}_N^{-1}\mathbf{t}.\end{aligned}$$

결국 위의 예측 식에서 평균과 공분산의 $\mathbf{k}$는 예측하고자 하는 새로운 input인 $\mathbf{x}_{N+1}$에 대한 함수이므로, predictive distribution의 평균과 분산이 모두 $\mathbf{x}_{N+1}$에 따라 변하는 정규분포임을 알 수 있다. 우리는 이전 세션에서 베이지안 회귀 모델을 공부할 때 예측하려는 input에 따라 예측 결과가 변함을 확인할 수 있었는데, 비모수 베이지안 회귀 모델이라 볼 수 있는 GPR 역시 예측하려는 input에 따라 예측 결과가 변함을 확인할 수 있다.

GPR의 output에 시그모이드 함수와 같은 활성화 함수를 적용하여, 우리는 회귀 모델과 로지스틱 회귀 모델의 관계처럼 GPR을 Classification에도 활용할 수 있다.