kernel classification and regression

- ridge regression

given data and response $(x_i, y_i)_{i=1}^n$, $x_i \in \mathbb{R}^D$

$$X = \begin{bmatrix} x_1^T \\ \vdots \\ x_n^T \end{bmatrix}_{n \times D} \qquad y = \begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix}_{n \times 1}$$

$n > D$

linear regression $x_i^T \beta \approx y_i$, $\beta \in \mathbb{R}^D$

regularize by $\|\beta\|_2^2$

$$\hat{\beta} := \underset{\beta \in \mathbb{R}^D}{\arg\min} \sum_{i=1}^n (y_i - x_i^T \beta)^2 + \lambda \|\beta\|_2^2, \quad \lambda > 0$$

$$L(\beta) = \|X\beta - y\|_2^2 + \lambda \|\beta\|_2^2$$
$$= \beta^T X^T X \beta - 2y^T X \beta + \|y\|^2 + \lambda \beta^T \beta$$

$$\frac{\partial L}{\partial \beta} = 2\left((X^T X)\beta - X^T y + \lambda \beta \right) = 0$$

$$(X^T X + \lambda I)\hat{\beta} = X^T y$$

$(X^T X)_{D \times D}$ PSD

$(X^T X + \lambda I)$ invertible

$$\Rightarrow \hat{\beta} = (X^T X + \lambda I)^{-1} X^T y$$

The predictive model: $x^{(te)}$ new data sample

$$\hat{y}(x^{(te)}) = x^{(te)T} \hat{\beta}$$
$$= x^{(te)T} (X^T X + \lambda I)^{-1} (X^T y)$$
$$= y^T X (X^T X + \lambda I)^{-1} x^{(te)}$$

(claim)
$$= y^T (X X^T + \lambda I)^{-1} X x^{(te)}$$

pf. Suppose $X_{n \times D} = U_{n \times D} S_{D \times D} V^T_{D \times D}$

$X^T X = V S^2 V^T \quad D \times D$

$X X^T = U S^2 U^T \quad n \times n \quad (\text{rank} \le D)$

$$① = X(X^T X + \lambda I)^{-1} = U S V^T \left(V(S^2 + \lambda I) V^T \right)^{-1}$$
$$= U S (S^2 + \lambda I)^{-1} V^T$$

$$② = (X X^T + \lambda I)^{-1} X = \left(U(S^2 + \lambda I) U^T \right)^{-1} U S V^T$$
$$= U (S^2 + \lambda I)^{-1} S V^T$$

bcz S is diagonal, ① = ②. #

Thus

$$\hat{y}(x^{(te)}) = y^T (X X^T + \lambda I)^{-1} X x^{(te)}$$
$$= \sum_{i=1}^n \left((G + \lambda I)^{-1} g_{te} \right)_i y_i \qquad (*)$$

$G := X X^T$, $G_{ij} = x_i^T x_j$ Gram matrix $n \times n$

$g_{te} := X x^{(te)}$, $(g_{te})_i = x_i^T x^{(te)}$, $i = 1, \dots, n$

Rk (*) is a weighted avg of y_i (training labels) and only involves

$$x_i^T x_j, \quad x_i^T x^{(te)}$$

- kernel ridge regression

idea: change $x^T y$ to $\langle \phi(x), \phi(y) \rangle_{\mathcal{H}} = k(x, y)$

equivalently, use feature map $\phi(x_i)$ in place of x_i in the regression.

If we just change the inner-product, we get

$$\hat{y}(x^{(te)}) = y^T (G + \lambda I)^{-1} g_{te}, \qquad (*')$$

$$G = (k(x_i, x_j))_{i,j=1}^n$$

$$g_{te} = (k(x_i, x^{(te)}))_{i=1}^n$$

We show that (*') is same as if we use $\phi(x_i)$ in the regression.

the model is to find $\beta \in \mathcal{H}$,

$$\langle \phi(x_i), \beta \rangle_{\mathcal{H}} \approx y_i, \quad i = 1, \dots, n$$

$$\min_{\beta \in \mathcal{H}} \sum_{i=1}^n \left(\langle \phi(x_i), \beta \rangle_{\mathcal{H}} - y_i \right)^2 + \lambda \|\beta\|_{\mathcal{H}}^2$$

claim: $\hat{\beta} \in \text{span}\{\phi(x_1), \dots, \phi(x_n)\}$,

(bcz o.t. $\beta = P\beta + r$, $P\beta$ is the orthogonal projection to the n-dim subspace, if $r \neq 0$, $\|\beta\|_{\mathcal{H}}^2$ is larger, and r makes no change to the prediction of y_i)

thus $\beta = \sum_{j=1}^n \alpha_j \phi(x_j)$, $\alpha \in \mathbb{R}^n$ TBD.

$$\langle \phi(x_i), \beta \rangle_{\mathcal{H}} = \sum_{j=1}^n \alpha_j \underbrace{\langle \phi(x_i), \phi(x_j) \rangle_{\mathcal{H}}}_{= k(x_i, x_j)}$$
$$= (G\alpha)_i,$$

$$\|\beta\|_{\mathcal{H}}^2 = \left\langle \sum_{j=1}^n \alpha_j \phi(x_j), \sum_{j'} \alpha_{j'} \phi(x_{j'}) \right\rangle_{\mathcal{H}}$$
$$= \alpha^T G \alpha.$$

$$\min_{\alpha} \|G\alpha - y\|_2^2 + \lambda \alpha^T G \alpha = L(\alpha)$$

$$\frac{\partial L}{\partial \alpha} = 2\left(G^T G \alpha - G^T y + \lambda G \alpha \right) = 0$$

$$\Rightarrow G(G + \lambda I)\alpha = G y, \quad (G = G^T, \text{PSD})$$

$$\Rightarrow \hat{\alpha} = (G + \lambda I)^{-1} y$$

$$\hat{y}(x') = \langle \phi(x'), \hat{\beta} \rangle_{\mathcal{H}}$$
$$= \sum_{i=1}^n \hat{\alpha}_i \langle \phi(x'), \phi(x_i) \rangle_{\mathcal{H}}$$
$$= \sum_{i=1}^n k(x', x_i) \hat{\alpha}_i = \hat{\alpha}^T g_{te}$$
$$= y^T (G + \lambda I)^{-1} g_{te} \quad \text{which is } (*') \ \#.$$

- Classification

$$\{(x_i, y_i)\}_{i=1}^n, \quad y_i = \pm 1$$

$$\min_{f \in \mathcal{H}} \sum_{i=1}^n l(y_i, f(x_i)) + \lambda \|f\|_{\mathcal{H}}^2$$

$l(y, f(x))$ loss eg. logistic loss

$$\hat{\Pr}[l(x) = 1 \mid x] = \frac{e^{f(x)}}{1 + e^{f(x)}}$$

kernel logistic regression

Rk. generalize to M-class classification

multi-logit regression:

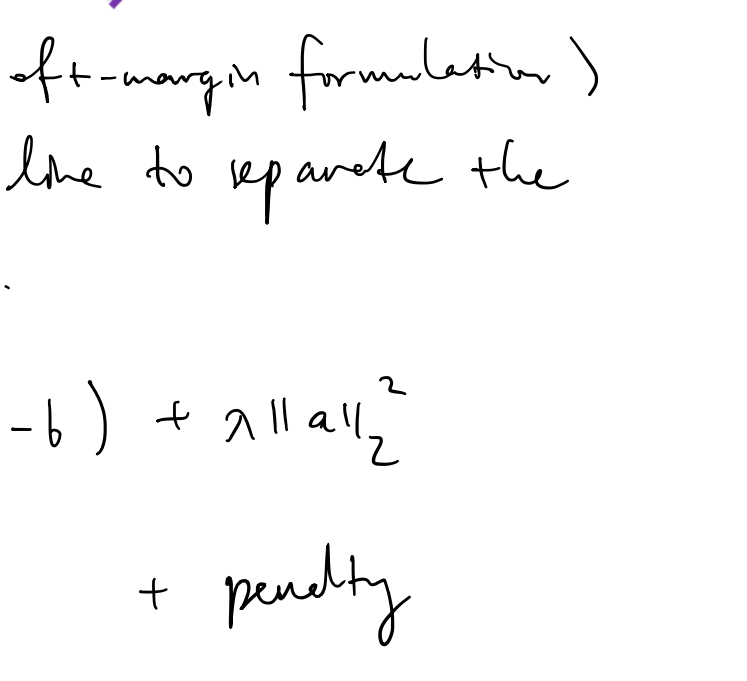
$$\hat{\Pr}[l(x) = m \mid x] = \frac{e^{f_m(x)}}{\sum_{j=1}^M e^{f_j(x)}},$$

$f_1, \dots, f_M \in \mathcal{H}$

This is same as in softmax layer in NN, where $f_1, \dots, f_M$ are last hidden layer activations in an NN.

- Kernel logistic regression is related to SVM

- linear SVM



$y_i = \pm 1$

$$\begin{cases} \min\limits_{a \in \mathbb{R}^D, b \in \mathbb{R}} \|a\|_2 \\ \text{s.t. } y_i(a^T x_i - b) \ge 1 \end{cases}$$

Convex optimization

- relax to hinge loss (soft-margin formulation) when there is no line to separate the two classes perfectly.

$$\min_{a, b} \sum_{i=1}^n l(y_i, a^T x_i - b) + \lambda \|a\|_2^2$$

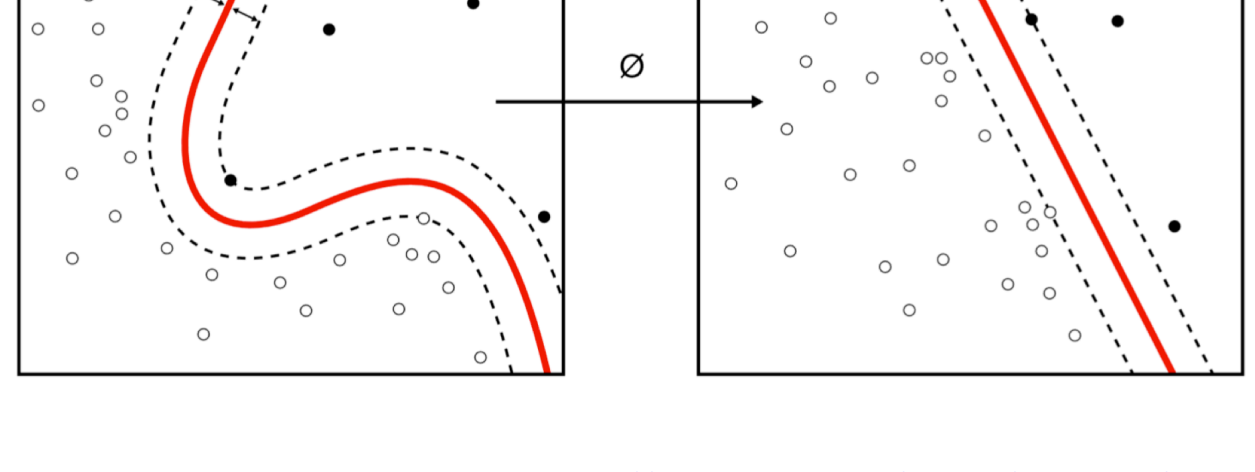
loss + penalty

- kernel SVM

$$\beta := \begin{bmatrix} a \\ -b \end{bmatrix} \qquad \bar{x}_i = \begin{bmatrix} x \\ 1 \end{bmatrix},$$

$\beta^T \bar{x}_i \rightarrow$ kernel inner product

$\beta \in \mathbb{R}^{D+1}$ $\langle \beta, \phi(x_i) \rangle_{\mathcal{H}}$, $\beta \in \mathcal{H}$



Code implementation in Python sklearn: https://scikit-learn.org/stable/modules/svm.html