Unique name: lyifeng UMID:44603814

EECS 545 HW 1

Study group: Kunyi Lu, Chenrui Shu

Problem 1:

(a) The code is attached. The error rate on test data is : 0.001250.

(b) For a two-class Naive Bayes classifier, y equal c (1) or $\bar{c}$ (0) and

x has dimension D with every element equal 0 or 1, we can have:

$$w_c = (\log\frac{\theta_{c11}}{1-\theta_{c11}}, \log\frac{\theta_{c21}}{1-\theta_{c21}}, \cdots, \log\frac{\theta_{cD1}}{1-\theta_{cD1}}, \log\pi_c + \sum_{d=1}^{D}\log(1-\theta_{cd1}))^T$$

$$w_{\bar{c}} = (\log\frac{\theta_{\bar{c}11}}{1-\theta_{\bar{c}11}}, \log\frac{\theta_{\bar{c}21}}{1-\theta_{\bar{c}21}}, \cdots, \log\frac{\theta_{\bar{c}D1}}{1-\theta_{\bar{c}D1}}, \log\pi_{\bar{c}} + \sum_{d=1}^{D}\log(1-\theta_{\bar{c}d1}))^T$$

$$w = w_c - w_{\bar{c}}$$

Then we can get:

$$w^T * \varphi(x) > 0 \Leftrightarrow (w_c^T - w_{\bar{c}}^T) * \varphi(x) > 0 \Leftrightarrow w_c^T * \varphi(x) > w_{\bar{c}}^T * \varphi(x)$$

Since $(1-\theta_{cd1}) = \theta_{cd0}$, $(1-\theta_{\bar{c}d1}) = \theta_{\bar{c}d0}$

$$w_c^T * \varphi(x) = \log \pi_c * \prod_{d=1}^{D}(\frac{\theta_{cd1}}{1-\theta_{cd1}})^{I(x_d=1)} * \prod_{d=1}^{D}(1-\theta_{cd1}) =$$

$$\log \pi_c * \prod_{d=1}^{D}(\theta_{cd1})^{I(x_d=1)} * \prod_{d=1}^{D}(\theta_{cd0})^{I(x_d=0)} = \log P(y=c \mid x=(x1, x2, \cdots, xd)^T)$$

$$w_{\bar{c}}^T * \varphi(x) = \log \pi_{\bar{c}} * \prod_{d=1}^{D}(\frac{\theta_{\bar{c}d1}}{1-\theta_{\bar{c}d1}})^{I(x_d=1)} * \prod_{d=1}^{D}(1-\theta_{\bar{c}d1})$$

$$= \log \pi_{\bar{c}} * \prod_{d=1}^{D}(\theta_{\bar{c}d1})^{I(x_d=1)} * \prod_{d=1}^{D}(\theta_{\bar{c}d0})^{I(x_d=0)} = \log P(y=\bar{c} \mid x=(x1, x2, \cdots, xd)^T)$$

So we can point out that

$$\boldsymbol{w_c}^{\boldsymbol{T}} * \varphi(\mathrm{x}) > \boldsymbol{w_{\bar{c}}}^{\boldsymbol{T}} * \varphi(\mathrm{x}) \Leftrightarrow \boldsymbol{P}(\boldsymbol{y} = \mathrm{c} \mid \boldsymbol{x} = (\boldsymbol{x}1, \boldsymbol{x}2, \cdots, \boldsymbol{xd})^{\boldsymbol{T}}) > \boldsymbol{P}(\boldsymbol{y} = \bar{\boldsymbol{c}} \mid \boldsymbol{x} = (\boldsymbol{x}1, \boldsymbol{x}2, \cdots, \boldsymbol{xd})^{\boldsymbol{T}})$$

$$\Leftrightarrow \boldsymbol{P}(\boldsymbol{y} = 1 \mid \boldsymbol{x} = (\boldsymbol{x}1, \boldsymbol{x}2, \cdots, \boldsymbol{xd})^{\boldsymbol{T}}) > \boldsymbol{P}(\boldsymbol{y} = 0 \mid \boldsymbol{x} = (\boldsymbol{x}1, \boldsymbol{x}2, \cdots, \boldsymbol{xd})^{\boldsymbol{T}})$$

Which means the linear classifier listed in the question has the same function with two-class Naive Bayes binary classifier hence the two-class Naive Bayes classifier can be represented as a linear classifier. Therefore Naive Bayes binary classifier is a linear classifier.

Problem 2:

Pr.2. Since K_1 K_2 are positive-definite kernel. For any $X = (x_1 \cdots x_N)^T$ $X^T K_1 X > 0$ $X^T K_2 X > 0$

[A matrix G is positive-definite $\Leftrightarrow$ $X^T G X > 0$ for any X]

(i) $\because$ $X^T K_1 X > 0$ for any X $\therefore a X^T K_1 X > 0$ when $a > 0$ $\therefore X^T (a K_1) X > 0$

$\therefore K(x, x') = a k_1(x, x')$ is a valid kernel

(ii) $\because$ $X^T K_1 X > 0$ $X^T K_2 X > 0$ for any X. $\therefore X^T K_1 X + X^T K_2 X = X^T (K_1 + K_2) X > 0$

$\therefore K(x, x') = k_1(x, x') + k_2(x, x')$ is a valid kernel

(iii) Do a decomposition that $K_2 = R'R$ where $R = (r_{ij})$ is a real-valued matrix

Then for any Y. $Y'KY = \sum_{ij} y_i K(x_i, x_j) y_j$

$$= \sum_{ij} y_i k_1(x_i x_j) k_2(x_i x_j) y_j$$

$$= \sum_{ij} y_i k_1(x_i x_j) \left(\sum_k r_{ki} r_{kj}\right) y_j$$

$$= \sum_k \sum_{ij} (r_{ki} y_i) k_1(x_i x_j) (r_{kj} y_j)$$

$$= \sum_k h_k' K_1 h_k > 0 \qquad [h_{ki} = r_{ki} y_i]$$

$\therefore k(x, x') = k_1(x, x') k_2(x, x')$ is a valid kernel

(iv) $\because K = f(x_i) f(x_j)$ $\therefore$ For any Y, $Y^T K Y = \sum_{ij} y_i f(x_i) f(x_j) y_j = [y_1^2 f(x_1)^2 + \cdots + y_n^2 f(x_n)^2 + 2 y_1 f(x_2) + \cdots + 2 y_n f(x_{n-1})]$

$$= \left(\sum_{i=1}^{n} y_i f(x_i)\right)^2$$

$$> 0$$

$\therefore K = f(x) f(x')$ is a valid kernel

(v) from (iii) we know $k = k_1 k_2$ is a valid kernel

$\therefore$ $k = k_1^d = k_1 \cdot k_1 \cdot k_1 \cdots k_1$, every time we multiply two valid kernel

$\therefore$ By apply (iii), $k = k_1^d$ is a valid kernel

(vi) $k' = p(k)$ where k is a arbitrary kernel.

① when k is a valid kernel, Then $k' = \sum a k^d$, k' is the sum of polynomial term with product of valid kernels and positive coefficient. By applying (i) and (v), $k' = p(k)$ is a valid kernel

② If k is not a valid kernel. Then $k' = p(k)$ is not a valid kernel.

(b) The Gaussian kernel $k(x,x') = \exp\left(-\frac{\|x-x'\|^2}{2\sigma^2}\right) = \exp\left(-\frac{(x-x')^2}{2\sigma^2}\right) = \exp\left(\frac{-x^2+2xx'-x'^2}{2\sigma^2}\right)$

As $e^x = 1 + x + \frac{x^2}{2} + \cdots + \frac{x^n}{n}$ (By power series and Taylor's theorem)

$$\therefore k(x,x') = \exp\left(\frac{-x^2-x'^2}{2\sigma^2}\right)\left(1 + \frac{2xx'}{2\sigma^2} + \left(\frac{2xx'}{2\sigma^2}\right)^2 / 2! + \cdots + \left(\frac{2xx'}{2\sigma^2}\right)^n / n!\right)$$

$$= \exp\left(\frac{-x^2-x'^2}{2\sigma^2}\right)\left(1 \cdot 1 + \sqrt{\frac{1}{\sigma^2}}x\sqrt{\frac{1}{\sigma^2}}x' + \cdots + \sqrt{\frac{1}{\sigma^{2n}n!}}x^n\sqrt{\frac{1}{\sigma^{2n}n!}}x'^n\right)$$

$$= \phi(x)^T\phi(x')$$

where $\phi(x) = \exp\left(\frac{-x^2}{2\sigma^2}\right)\left[1, \sqrt{\frac{1}{\sigma^2}}x, \sqrt{\frac{1}{\sigma^4 2!}}x^2, \cdots \sqrt{\frac{1}{\sigma^{2n}n!}}x^n\right]^T$

Problem 3:

(a) The prediction tor y(x) for Kernel Perceptron when given a new sample x' is:

$$\overline{\mathrm{y}} = \mathrm{sgn}(\sum_{i} \boldsymbol{\alpha}_i \boldsymbol{y}_i \boldsymbol{K}(\boldsymbol{x}_i, \boldsymbol{x}'))$$

And the pseudocode is shown below:

Initialize α to an all-zeros vector of length n, the number of training samples.

For some fixed number of iterations, or until some stopping criterion is met:

For each training example x_j with ground truth label $y_j \in \{-1, 1\}$:

Let $\overline{\mathrm{y}} = \mathrm{sgn}(\sum_{i} \boldsymbol{\alpha}_i \boldsymbol{y}_i \boldsymbol{K}(\boldsymbol{x}_i, \boldsymbol{x}_j))$

If $\hat{y} \neq y_i$, perform an update by incrementing the mistake counter:

$\boldsymbol{a}_j \leftarrow \boldsymbol{a}_j + 1$

(b) Two plots are shown in Fig 1 and Fig 2.

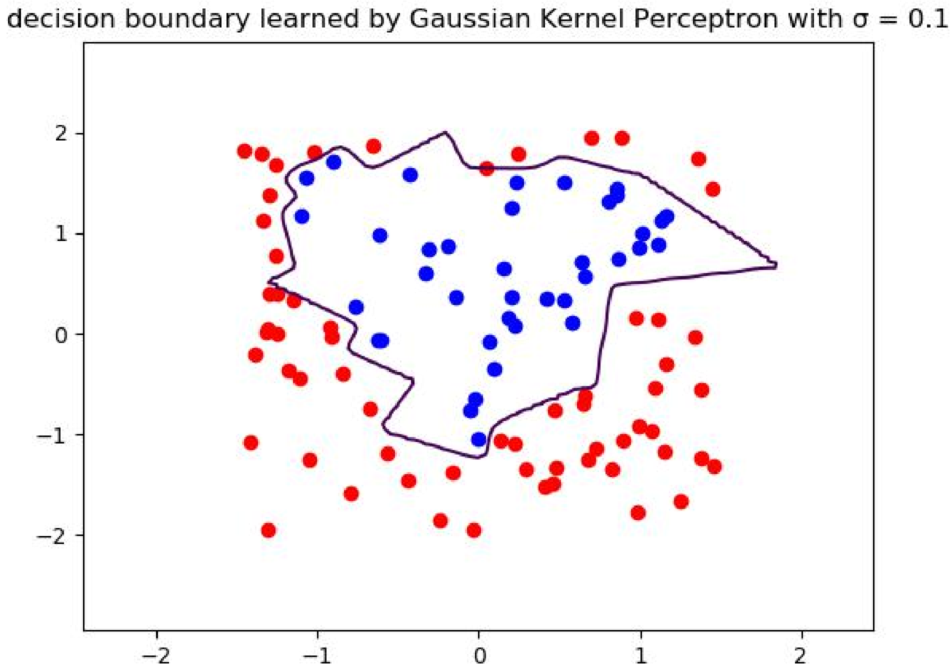

Fig 1 decision boundary with σ = 0.1

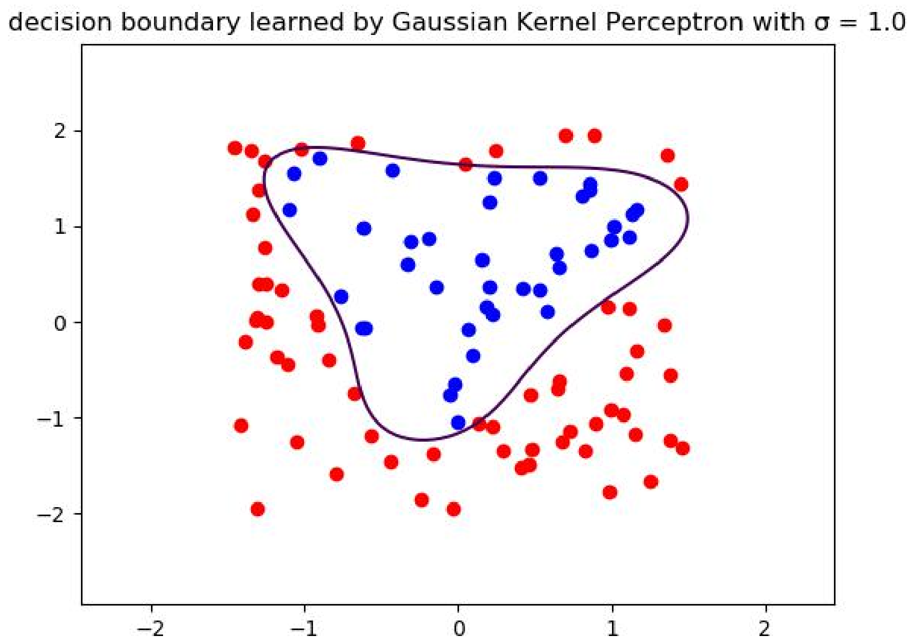

Fig 2 decision boundary with σ = 1.0

Problem 4:

Two plots are shown in Fig 3 and Fig 4.

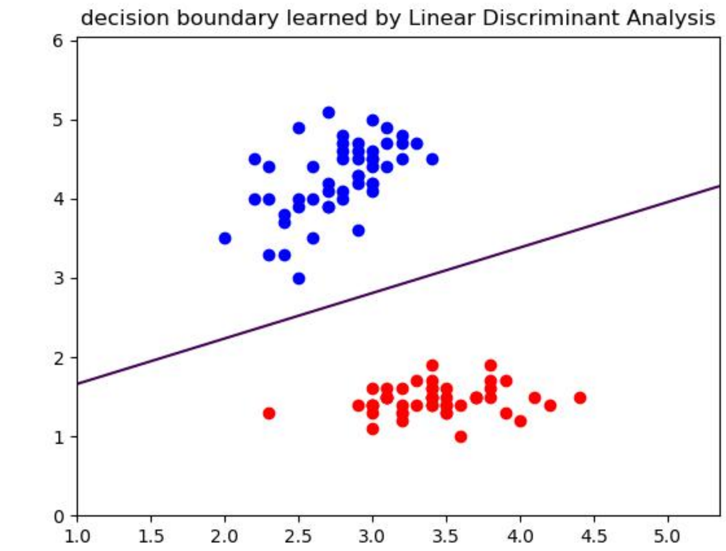

Fig 3 decision boundary by Linear Discriminant Analysis

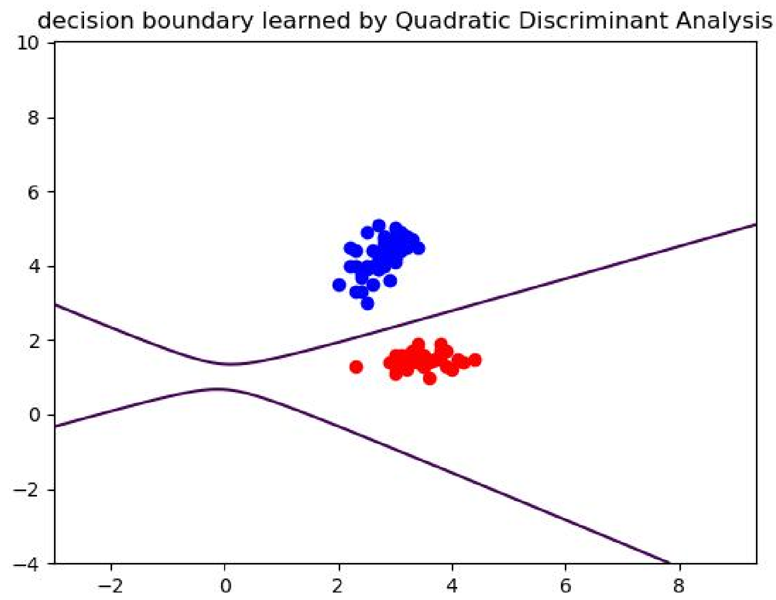

Fig 4 decision boundary by Quadratic Discriminant Analysis