Logistic regression

This method focuses on $K=2$ assuming that log odds of class 1 vs. class 0 is linear in x:

$$\log \frac{P(Z=1 \mid X=x)}{P(Z=2 \mid X=x)} = \beta_0 + \beta_1^T x \quad \text{with } \beta_0 \in \mathbb{R},\ \beta_1 \in \mathbb{R}^p$$

- Note that LDA also has log odds linear in x with $\hat{\beta}_0, \hat{\beta}_1$ depend on $\hat{\mu}_1, \hat{\mu}_2, \hat{\Sigma}$.

$$\log \frac{P(Z=1 \mid X=x)}{P(Z=2 \mid X=x)} = \log \frac{\pi_1 \cdot f(x;\mu_1,\Sigma)}{\pi_2 \cdot f(x;\mu_2,\Sigma)}$$

$$= \log \frac{\cancel{\frac{1}{(2\pi)^{p/2}}}\,\cancel{\frac{1}{|\Sigma|^{1/2}}}\, e^{-\frac{1}{2}(x-\mu_1)^T\Sigma^{-1}(x-\mu_1)}}{\cancel{\frac{1}{(2\pi)^{p/2}}}\,\cancel{\frac{1}{|\Sigma|^{1/2}}}\, e^{-\frac{1}{2}(x-\mu_2)^T\Sigma^{-1}(x-\mu_2)}} + \log\frac{\pi_1}{\pi_2} =$$

$$= -\frac{1}{2}(x-\mu_1)^T\Sigma^{-1}(x-\mu_1) + \frac{1}{2}(x-\mu_2)^T\Sigma^{-1}(x-\mu_2) + \log\frac{\pi_1}{\pi_2} =$$

$$= \underbrace{(\mu_1-\mu_2)^T\Sigma^{-1}}_{\beta_1^T} x \underbrace{- \mu_1^T\Sigma^{-1}\mu_1 + \mu_2^T\Sigma^{-1}\mu_2 + \log\frac{\pi_1}{\pi_2}}_{\beta_0}$$

To simplify the derivations we will relabel class 2 into 0, i.e. $y_i \in \{0, 1\}$, and include the intercept in features, i.e. $\beta = \begin{pmatrix} \beta_0 \\ \beta_1 \end{pmatrix} \in \mathbb{R}^{p+1}$ and $X \in \mathbb{R}^{n \times (p+1)}$

Logistic regression directly estimates β.

- $P(Z=1 | X=x) = \dfrac{1}{1+e^{-\beta^T x}}$

Denote by $\pi(x) = P(Z=1 | X=x) \Rightarrow 1-\pi(x) = P(Z=0 | X=x)$

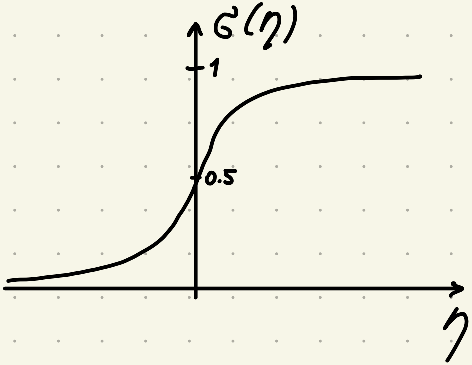
$$\log\left(\frac{\pi(x)}{1-\pi(x)}\right) = \beta^T x \quad \Rightarrow \quad \frac{1-\pi(x)}{\pi(x)} = \frac{1}{\pi(x)} - 1 = e^{-\beta^T x} \quad \Rightarrow$$

$$\pi(x) = \frac{1}{1+e^{-\beta^T x}}$$

$$P(Z=1 | X=x) = \sigma(\beta^T x) \quad \text{where} \quad \sigma(\eta) = \frac{1}{1+e^{-\eta}} = \frac{e^{\eta}}{1+e^{\eta}}$$

This function is called sigmoid function.

Properties: $\sigma(\eta) = \frac{1}{1+e^{-\eta}}$



- $\sigma(\eta) \geq 0.5$ for $\eta \geq 0.5$

  $\sigma(\eta) \leq 0.5$ for $\eta \leq 0.5$

| $P(Z=1 / X=x) \geq 0.5$ if $\beta_0 + \beta^T x \geq 0$

| $P(Z=2 / X=x) \leq 0.5$ if $\beta_0 + \beta^T x \leq 0$

| Decision boundary $\beta_0 + \beta^T x = 0$.

- $\sigma'(\eta) = \sigma(\eta)(1-\sigma(\eta))$

| $\sigma'(\eta) = \frac{e^{-\eta}}{(1+e^{-\eta})^2} = \frac{1}{1+e^{-\eta}} \cdot \frac{e^{-\eta}}{1+e^{-\eta}} = \sigma(\eta) \cdot (1-\sigma(\eta))$

- $(\log \sigma(\eta))' = 1 - \sigma(\eta)$, $(\log(1-\sigma(\eta)))' = -\sigma(\eta)$

| $(\log \sigma(\eta))' = \frac{\sigma'(\eta)}{\sigma(\eta)} = \frac{\cancel{\sigma(\eta)}(1-\sigma(\eta))}{\cancel{\sigma(\eta)}}$

Logistic regression finds β by maximizing log-likelihood:

$$\ell(X,y;\beta) = \sum_{i=1}^{n} \log P(z = y_i \mid X = x_i)$$

$$\ell(X,y;\beta) = \sum_{i:y_i=1} \log \sigma(\beta^T x_i) + \sum_{i:y_i=0} \log\left(1-\sigma(\beta^T x_i)\right) =$$
$$= \sum_{i=1}^{n} y_i \log \sigma(\beta^T x_i) + (1-y_i) \log\left(1-\sigma(\beta^T x_i)\right)$$

Maximizing log-likelihood has no explicit solution. We need to apply one of the optimization algorithms.

Detour: Newton's method

Given a function $f: \mathbb{R}^p \to \mathbb{R}$, solve

$$\underset{x}{\text{minimize}}\ f(x)$$

Let $x = \begin{pmatrix} x_1 \\ \vdots \\ x_p \end{pmatrix}$ and $\nabla f(x) = \begin{pmatrix} \frac{\partial f(x)}{\partial x_1} \\ \vdots \\ \frac{\partial f(x)}{\partial x_p} \end{pmatrix} \in \mathbb{R}^p$ be the gradient

Let $\nabla^2 f(x) = \begin{pmatrix} \frac{\partial f(x)}{\partial x_1^2} & \frac{\partial f(x)}{\partial x_1 \partial x_2} & \cdots & \frac{\partial f(x)}{\partial x_1 \partial x_p} \\ \vdots & & & \vdots \\ \frac{\partial f(x)}{\partial x_1 \partial x_p} & \frac{\partial f(x)}{\partial x_2 \partial x_p} & \cdots & \frac{\partial f(x)}{\partial x_p^2} \end{pmatrix} \in \mathbb{R}^{p \times p}$ be the Hessian.

Gradient descent update: $x^{(t+1)} = x^{(t)} - \nabla f(x^{(t)})$

Newton's method update: $x^{(t+1)} = x^{(t)} - \left(\nabla^2 f(x^{(t)})\right)^{-1} \nabla f(x^{(t)})$

Motivation for Newton's update:

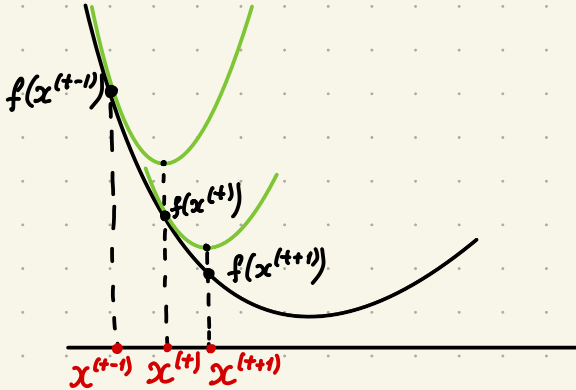
- Let's take the second order approximation of $f(x)$ at point $x \in \mathbb{R}^p$

$$f(y) \simeq f(x) + \nabla f(x)^T (y-x) + \frac{1}{2}(y-x)^T \nabla^2 f(x)(y-x)$$

- value y that minimizes the approximation corresponds to Newton's update.

$$\nabla_y f(y) \simeq \nabla f(x) + \nabla^2 f(x)(y-x) = 0$$

Then $y = x - \left(\nabla^2 f(x)\right)^{-1} \nabla f(x)$



Iteratively reweighted least squares

$$\ell(X,y;\beta) = \sum_{i=1}^{n} y_i \log \sigma(\beta^T x_i) + (1-y_i)\log(1-\sigma(\beta^T x_i))$$

- $\nabla_\beta \ell(X,y;\beta) = X^T(y-\pi)$ where $\pi = \begin{pmatrix} P(z=1 \mid X=x_1) \\ \vdots \\ P(z=1 \mid X=x_n) \end{pmatrix}$

Denote by $\pi_i = P(z=1 \mid X=x_i)$

$$\nabla_\beta = \sum_{i=1}^{n} y_i (1-\sigma(\beta^T x_i)) \cdot x_i - (1-y_i)\sigma(\beta^T x_i) x_i =$$

$$= \sum_{i=1}^{n} y_i x_i - \sigma(\beta^T x_i) x_i = \sum_{i=1}^{n} (y_i - \sigma(\beta^T x_i)) x_i =$$

$$= \sum_{i=1}^{n} (y_i - \pi_i) x_i = X^T(y-\pi)$$

- $\nabla_\beta \ell(X, y; \beta) = -X^T W X$ where $W = \mathrm{diag}(\pi * (1-\pi))$

$$\nabla_\beta^2 = -\sum_{i=1}^{n} \sigma(\beta^T x_i)(1-\sigma(\beta^T x_i))\, x_i x_i^T =$$
$$= -\sum_{i=1}^{n} \pi_i (1-\pi_i)\, x_i x_i^T = -X^T W X$$

- At iteration t, we obtain $\beta^{(t)}, \pi^{(t)}$ and $W^{(t)}$ then Newton's update implies

$$\beta^{(t+1)} = (X^T W^{(t)} X)^{-1} X^T W^{(t)} z^{(t)} \text{ for some } z^{(t)}$$

$$\beta^{(t+1)} = \beta^{(t)} + (X^T W^{(t)} X)^{-1} X^T (y - \pi^{(t)}) =$$
$$= (X^T W^{(t)} X)^{-1} X^T (W^{(t)} X \beta^{(t)} + y - \pi^{(t)}) =$$
$$= (X^T W^{(t)} X)^{-1} X^T W^{(t)} \big(\underbrace{X \beta^{(t)} + (W^{(t)})^{-1} (y - \pi^{(t)})}_{z^{(t)}}\big) =$$
$$= (X^T W^{(t)} X)^{-1} X^T W^{(t)} z^{(t)}$$

This is regression $z^{(t)} \sim X$ with weights $W^{(t)}$.

Logistic regression via IRLS:

Input: X, y, initialization $\beta^{(0)}$

At iteration $t = 1, 2, \dots$

Step 1 Compute $z^{(t)} = X\beta^{(t)} + (W^{(t)})^{-1}(y - \pi^{(t)})$

$W^{(t)} = \mathrm{diag}(\pi^{(t)} * (1 - \pi^{(t)}))$

where $\pi^{(t)} \simeq \begin{pmatrix} P(z=1 \mid X = x_1) \\ \cdots \\ P(z=1 \mid X = x_n) \end{pmatrix} = \begin{pmatrix} \sigma(\beta^{(t)T} x_1) \\ \cdots \\ \sigma(\beta^{(t)T} x_n) \end{pmatrix}$

Step 2 Solve weighted regression problem with response $z^{(t)}$, features X and weights $W^{(t)}$.

Output: coefficient vector β.

Extensions of logistic regression

① Multiple classes $K > 2$

$$P(Z=k \mid X=x) = \frac{e^{\beta_{0k}+\beta_{1k}^T x}}{\sum_{j=1}^{K} e^{\beta_{0j}+\beta_{1j}^T x}} \quad \text{for } k=1\ldots K$$

Softmax: $(\eta_1 \ldots \eta_K) \longrightarrow \left(\frac{e^{\eta_1}}{\sum_{j=1}^{K} e^{\eta_j}}, \ldots, \frac{e^{\eta_K}}{\sum_{j=1}^{K} e^{\eta_j}}\right)$

Fit $\{\beta_{0k}, \beta_{1k}\}_{k=1}^{K-1}$ by MLE.

Decision rule $h(x) = \underset{k=1..K}{\operatorname{argmax}}\, P(Z=k \mid X=x)$

- Log-odd are linear functions in x

$$\log \frac{P(Z=k \mid X=x)}{P(Z=K \mid X=x)} = \log\left(\frac{e^{\beta_{0k}+\beta_{1k}^T x}}{e^{\beta_{0K}+\beta_{1K}^T x}}\right)$$

$$= \overbrace{(\beta_{0k}-\beta_{0K})}^{\tilde{\beta}_{0k}} + \overbrace{(\beta_{1k}-\beta_{1K})}^{\tilde{\beta}_{1k}}{}^T x = \tilde{\beta}_{0k} + \tilde{\beta}_{1k} x$$

② In high dimensions ($p > n$) add regularization:

minimize $-\ell(X, y; \beta) + \lambda \, Pen(\beta)$

Example: penalties

$Pen(\beta) = \|\beta\|_2^2$ "ridge"

$Pen(\beta) = \|\beta\|_1$ "lasso"

$Pen(\beta) = (1-\alpha)\frac{\|\beta\|_2^2}{2} + \alpha\|\beta\|_1$ "elastic net"

k-nearest neighbors (KNN)

This is a distribution-free and memory-based method.

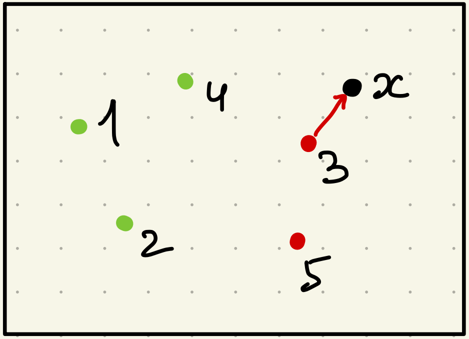
Given data $X \in \mathbb{R}^{n \times p}$ labels $y \in \mathbb{R}^n$ classify a point $x \in \mathbb{R}^p$ by:

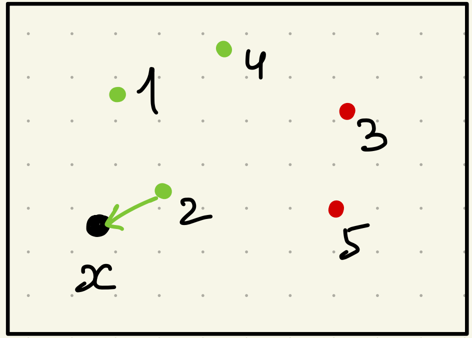
- find $N_k(x) \subseteq \{1, \dots, n\}$ the set of indices for k-nearest neighbors of x in the training data
- Classify x according to a majority vote

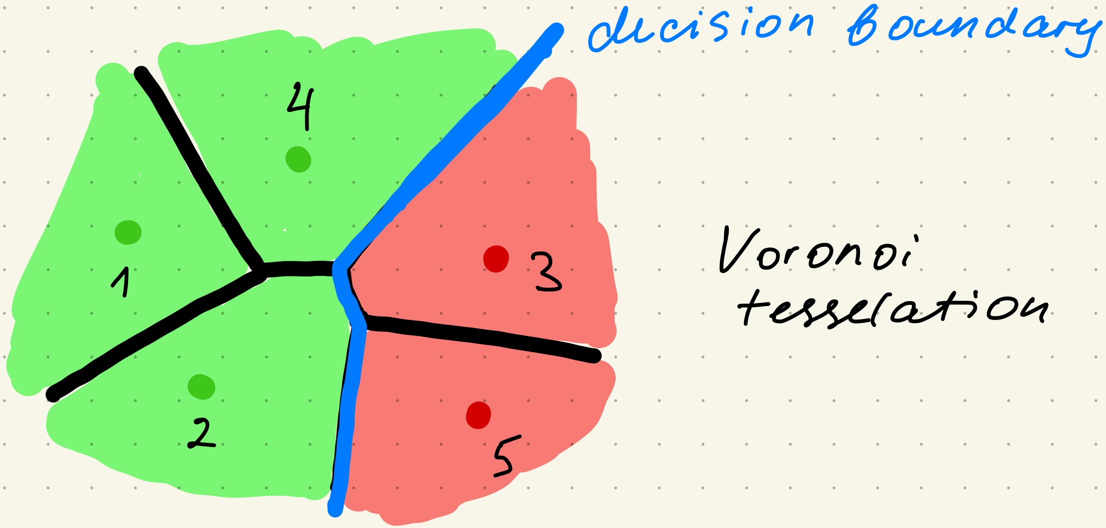
"π_k"

$$h(x) = \underset{k=1 \dots K}{\operatorname{argmax}} \frac{1}{K} \sum_{i \in N_k(x)} I(y_i = k)$$

Example: 1-NN, Euclidean distance

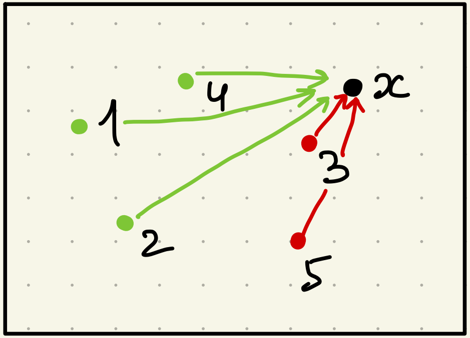


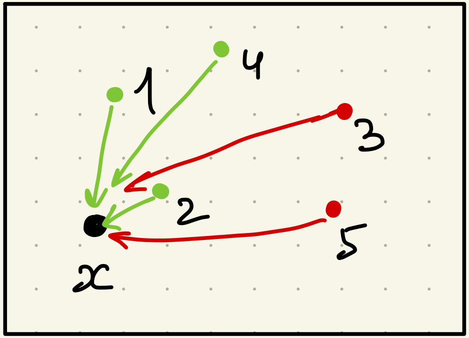


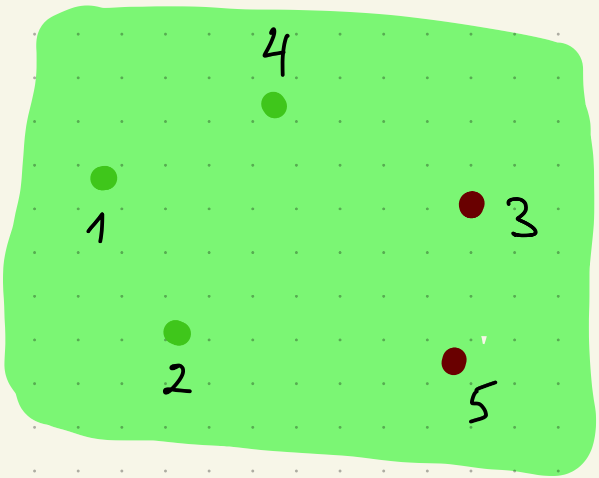


Voronoi tesselation

Example: n-NN, Euclidean distance







Dominant class prediction

1NN vs Bayes classifier

If we know $P(Z|X)$, Bayes optimal classifier

is $h(x) = \underset{k=1\ldots K}{\arg\max}\, P(Z=k \mid X=x) = \underset{k=1\ldots K}{\arg\max}\, \pi_k(x) = k^*$

Then the error for x is $\varepsilon_{BO} = 1 - \pi_{k^*}(x)$

For 1NN denote by x_{NN} the nearest neighbor of x

Then the error for x is $\varepsilon_{NN} = \sum_{k=1}^{K} \pi_k(x)\,(1 - \pi_k(x))$

If $n \to \infty$ then $\pi_k(x) = \pi_k(x_{NN})$ for $k = 1 \ldots K$

x is in class k with probability $\pi_k(x)$

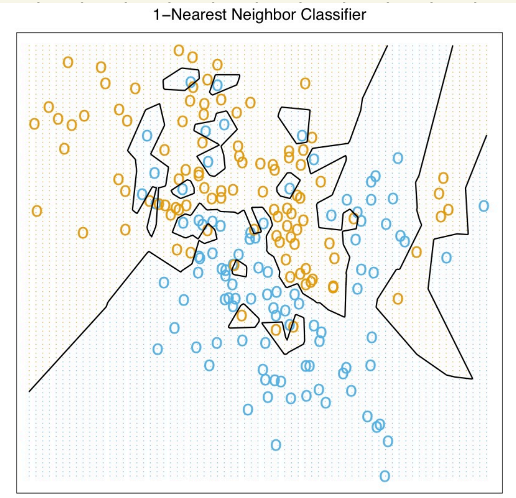
x_{NN} is not in class k with probability $1 - \pi_k(x)$

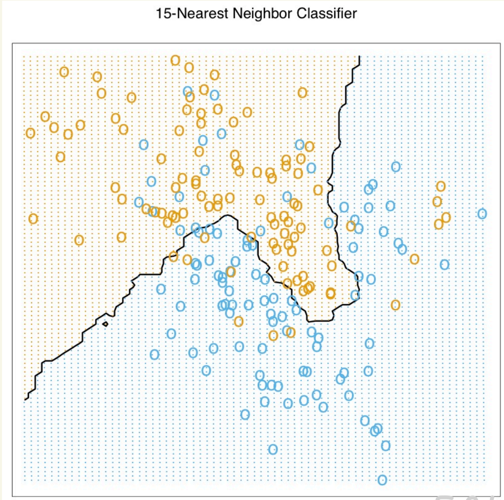
One can show that $\varepsilon_{BO} \le \varepsilon_{NN} \le 2\,\varepsilon_{BO}$

$K=2$, $\varepsilon_{NN} = 2\, p_{k^*}(x)\,(1 - p_{k^*}(x))$

Practical aspects of KNN

① K has significant impact on the result



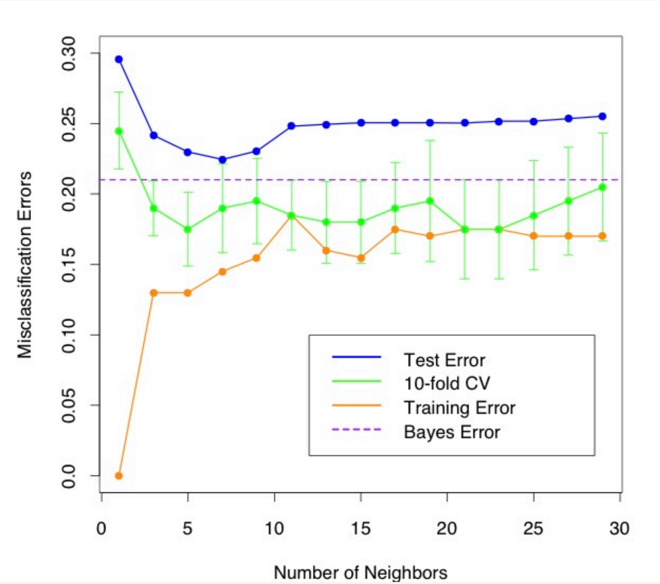


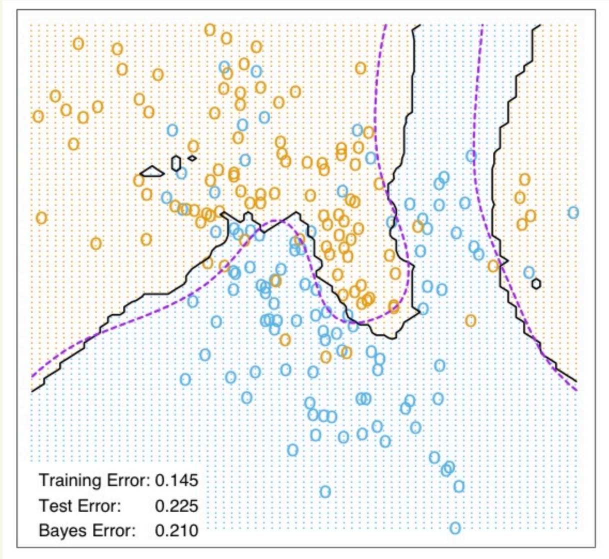
ESLII

Small K: capture local information, may overfit

Large K: more stable, may underfit

Use cross-validation to select K.





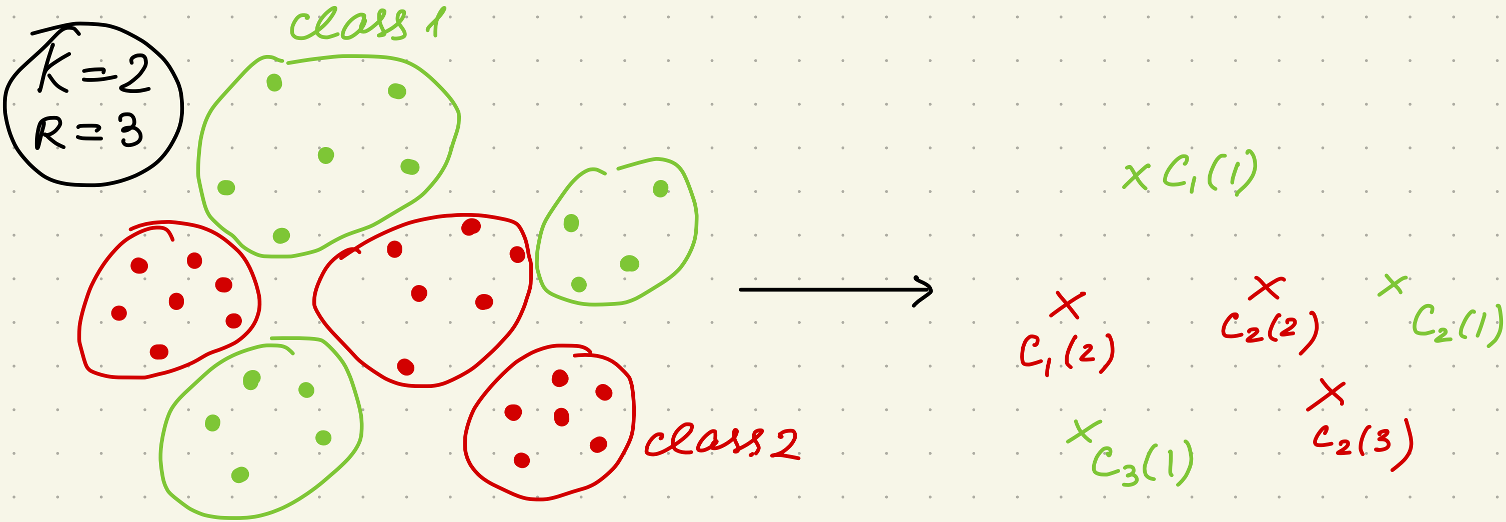
(2) Scaling is important, standarize X.

(3) KNN requires storing X and y

For class k, take $\{x_i : i \in C_k\}$ and perform R-Means clustering.

Denote the centroids by $c_1(k) \dots c_r(k)$

Replace $\{x_i : i \in C_k\} \longrightarrow \{c_r(k) : r = 1 \dots R\}$



4 KNN has equal vote weight

$$h(x) = \underset{k=1\dots K}{\operatorname{argmax}} \sum_{i \in N_k(x)} \frac{1}{K} I(y_i = k)$$

Ties can be broken at random, alternatively:

$$h(x) = \underset{k=1\dots K}{\operatorname{argmax}} \sum_{i \in N_k(x)} w(x, x_i)\, I(y_i = k)$$

$$\text{e.g.} \quad w(x, x_i) = \frac{1}{\|x_i - x\|^2}$$