Statistical Data Mining and Machine Learning
Hilary Term 2016

Dino Sejdinovic
Department of Statistics
Oxford

Slides and other materials available at:
http://www.stats.ox.ac.uk/~sejdinov/sdmml

Kernel Methods

Kernel trick in general

- In a learning algorithm, if only inner products $x_i^\top x_j$ are explicitly used, rather than data items x_i, x_j directly, we can replace them with a kernel function $k(x_i, x_j) = \langle \varphi(x_i), \varphi(x_j) \rangle$, where $\varphi(x)$ could be **nonlinear, high- and potentially infinite-dimensional** features of the original data.
  - Kernel ridge regression
  - Kernel logistic regression
  - Kernel PCA, CCA, ICA
  - Kernel K-means

Gram matrix

- The **Gram matrix** is the matrix of dot-products, $\mathbf{K}_{ij} = \varphi(x_i)^\top \varphi(x_j)$.

$$\mathbf{K} = \begin{pmatrix} — \varphi(x_1)^\top — \\ \vdots \\ — \varphi(x_i)^\top — \\ \vdots \\ — \varphi(x_n)^\top — \end{pmatrix} \cdot \begin{pmatrix} | & & | & & | \\ \varphi(x_1) & \cdots & \varphi(x_j) & \cdots & \varphi(x_n) \\ | & & | & & | \end{pmatrix}$$

- Since $\mathbf{K} = \Phi\Phi^\top$, it is symmetric and positive semidefinite.
- Recall: Gram matrix closely related to the distance matrix (MDS)
- Assuming features are centred, the sample covariance of features is $\Phi^\top\Phi$.
- Many kernel methods, e.g. kernel PCA, make use of the duality between the Gram and the sample covariance matrix.

Kernel: an inner product between feature maps

Definition (kernel)

Let $\mathcal{X}$ be a non-empty set. A function $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ is a **kernel** if there exists a **Hilbert space**[a] and a map $\varphi : \mathcal{X} \to \mathcal{H}$ such that $\forall x, x' \in \mathcal{X}$,

$$k(x, x') := \langle \varphi(x), \varphi(x') \rangle_{\mathcal{H}} .$$

[a] a vector space equipped with an inner product $\langle \cdot, \cdot \rangle$ which is also a complete metric space; can have infinitely many dimensions, e.g. the space ℓ^2 of all square-summable sequences or the space L^2 of all square-integrable functions

- Almost no conditions on $\mathcal{X}$ (eg, $\mathcal{X}$ itself need not have an inner product, e.g., documents).
- Think of kernel as a **similarity measure between features**

What are some simple kernels? E.g., for text documents? For images?

- A single kernel can correspond to multiple sets of underlying features.

$$\varphi_1(x) = x \qquad \text{and} \qquad \varphi_2(x) = \begin{pmatrix} x/\sqrt{2} & x/\sqrt{2} \end{pmatrix}^\top$$

Positive semidefinite functions

If we are given a “measure of similarity” with two arguments, $k(x, x')$, how can we determine if it is a valid kernel?

1. Find a feature map?
   - Sometimes not obvious (especially if the feature vector is infinite dimensional)
2. A simpler direct property of the function: positive semidefiniteness.

Positive semidefinite functions

Definition (Positive semidefinite functions)

A symmetric function $\kappa : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ is positive semidefinite if $\forall n \geq 1,\ \forall (a_1, \ldots a_n) \in \mathbb{R}^n,\ \forall (x_1, \ldots, x_n) \in \mathcal{X}^n$,

$$\sum_{i=1}^{n} \sum_{j=1}^{n} a_i a_j \kappa(x_i, x_j) \geq 0.$$

- Kernel $k(x, y) := \langle \varphi(x), \varphi(y) \rangle_{\mathcal{H}}$ for a Hilbert space $\mathcal{H}$ is positive semidefinite.

$$\begin{aligned} \sum_{i=1}^{n} \sum_{j=1}^{n} a_i a_j k(x_i, x_j) &= \sum_{i=1}^{n} \sum_{j=1}^{n} \langle a_i \varphi(x_i), a_j \varphi(x_j) \rangle_{\mathcal{H}} \\ &= \left\| \sum_{i=1}^{n} a_i \varphi(x_i) \right\|_{\mathcal{H}}^2 \geq 0. \end{aligned}$$

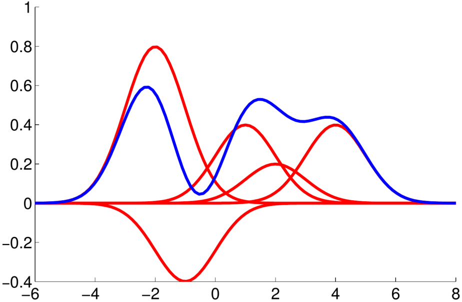
Positive semidefinite functions are kernels

Moore-Aronszajn Theorem

Every positive semidefinite function is a kernel for some Hilbert space $\mathcal{H}$.

- $\mathcal{H}$ is usually thought of as a space of functions (**Reproducing kernel Hilbert space - RKHS**)

Gaussian RBF kernel $k(x, x') = \exp\left(-\frac{1}{2\gamma^2}\|x - x'\|^2\right)$ has an infinite-dimensional $\mathcal{H}$ with elements $h(x) = \sum_{i=1}^{m} a_i k(x_i, x)$ (recall that $w^\top \varphi(x)$ in SVM has exactly this form!).



Reproducing kernel

Definition (Reproducing kernel)

Let $\mathcal{H}$ be a Hilbert space of functions $f : \mathcal{X} \to \mathbb{R}$ defined on a non-empty set $\mathcal{X}$. A function $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ is called **a reproducing kernel** of $\mathcal{H}$ if it satisfies

- $\forall x \in \mathcal{X}, \;\; k_x = k(\cdot, x) \in \mathcal{H}$,
- $\forall x \in \mathcal{X}, \forall f \in \mathcal{H}, \;\; \langle f, k(\cdot, x)\rangle_{\mathcal{H}} = f(x)$ (the reproducing property).

In particular, for any $x, y \in \mathcal{X}$, $k(x, y) = \langle k(\cdot, y), k(\cdot, x)\rangle_{\mathcal{H}} = \langle k(\cdot, x), k(\cdot, y)\rangle_{\mathcal{H}}$.

Can forget all about $\varphi(x)$ and just treat $k(\cdot, x)$ as a feature of x (it is a perfectly valid Hilbert-space valued feature)!

RKHS

Definition (Reproducing kernel Hilbert space)

A Hilbert space $\mathcal{H}$ of functions $f : \mathcal{X} \to \mathbb{R}$, defined on a non-empty set $\mathcal{X}$ is said to be a Reproducing Kernel Hilbert Space (RKHS) if evaluation functionals $\delta_x : \mathcal{H} \to \mathbb{R}$, $\delta_x f = f(x)$ are continuous $\forall x \in \mathcal{X}$.

Theorem (Norm convergence implies pointwise convergence)

If $\lim_{n\to\infty} \|f_n - f\|_{\mathcal{H}} = 0$*, then* $\lim_{n\to\infty} f_n(x) = f(x), \forall x \in \mathcal{X}$.

- If two functions $f, g \in \mathcal{H}$ are close in the norm of $\mathcal{H}$, then $f(x)$ and $g(x)$ are close for all $x \in \mathcal{X}$
- This is a property of particularly “nice” functional spaces. For example, does not hold on spaces endowed with L_2 norm: x^n on $[0, 1]$ converges to 0 in L_2 but not pointwise.

Back to SVMs

Maximum margin classifier in RKHS: Looking for a decision function of form $\text{sign}(w(x))$ where $w \in \mathcal{H}_k$. Because we are in an RKHS, $w(x) = \langle w, k(\cdot, x)\rangle_{\mathcal{H}_k}$.

$$\min_{w \in \mathcal{H}_k} \left(\frac{1}{2} \|w\|^2_{\mathcal{H}_k} + C \sum_{i=1}^{n} h\left(y_i \langle w, k(\cdot, x_i)\rangle_{\mathcal{H}_k} \right) \right)$$

for the RKHS $\mathcal{H}$ with kernel $k(x, x')$. Maximizing the margin equivalent to minimizing $\|w\|^2_{\mathcal{H}}$: for many RKHSs a smoothness constraint on function w. Why can we solve this infinite-dimensional optimization problem? Because we know that $w \in \text{span}\{k(\cdot, x_i) : i = 1, \ldots, n\}$ – Representer Theorem.

Representer theorem

Standard supervised learning setup: we are given a set of paired observations $(x_1, y_1), \dots (x_n, y_n)$.
Goal: find the function f^* in the RKHS $\mathcal{H}$ which solves the regularized empirical risk minimization problem.

$$\min_{f \in \mathcal{H}} \hat{R}(f) + \Omega\left(\|f\|_{\mathcal{H}}^2\right),$$

where empirical risk is

$$\hat{R}(f) = \frac{1}{n} \sum_{i=1}^{n} L(y_i, f(x_i), x_i),$$

and Ω is a non-decreasing function.

- Classification: L could be a hinge loss $L(y, f(x), x) = (1 - yf(x))_+$ or a logistic loss $L(y, f(x), x) = \log(1 + \exp(-yf(x))$.
- Regression: $L(y, f(x), x) = (y - f(x))^2$.

Representer theorem

Theorem (Representer Theorem)

There is a solution to

$$\min_{f \in \mathcal{H}} \hat{R}(f) + \Omega\left(\|f\|_{\mathcal{H}}^2\right)$$

that takes the form

$$f^* = \sum_{i=1}^{n} \alpha_i k(\cdot, x_i).$$

If Ω is strictly increasing, all solutions have this form.

Representer theorem: proof

Proof: Denote f_s projection of f onto the subspace

$$\mathrm{span}\left\{k(\cdot, x_i) : \ i = 1, \ldots, n\right\}$$

such that

$$f = f_s + f_\perp,$$

where $f_s = \sum_{i=1}^{n} \alpha_i k(\cdot, x_i)$ and $f_\perp$ is orthogonal to $\mathrm{span}\left\{k(\cdot, x_i) : \ i = 1, \ldots, n\right\}$.
Regularizer:

$$\|f\|_{\mathcal{H}}^2 = \|f_s\|_{\mathcal{H}}^2 + \|f_\perp\|_{\mathcal{H}}^2 \geq \|f_s\|_{\mathcal{H}}^2 ,$$

then

$$\Omega\left(\|f\|_{\mathcal{H}}^2\right) \geq \Omega\left(\|f_s\|_{\mathcal{H}}^2\right).$$

Representer theorem: proof

Proof (cont.): Individual terms $f(x_i)$ in the loss:

$$f(x_i) = \langle f, k(\cdot, x_i)\rangle_{\mathcal{H}} = \langle f_s + f_\perp, k(\cdot, x_i)\rangle_{\mathcal{H}} = \langle f_s, k(\cdot, x_i)\rangle_{\mathcal{H}},$$

so

$$L(y_i, f(x_i), x_i) = L(y_i, f_s(x_i), x_i) \forall i \implies \hat{R}(f) = \hat{R}(f_s).$$

Hence

- The empirical risk only depends on the components of f lying in the subspace spanned by canonical features.
- Regularizer $\Omega(\ldots)$ is minimized when $f = f_s$.
- If Ω is strictly non-decreasing, then $\|f_\perp\|_{\mathcal{H}} = 0$ is required at the minimum.

Kernel Methods – Discussion

- The framework of kernel methods allows building flexible machine learning models.
- **Nonparametric** method: parameter space (e.g., normal vector w in SVM) can be infinite-dimensional
- Kernels can be defined over more complex structures than vectors, e.g. graphs, strings, images, bags of instances, probability distributions.
- In naïve implementation, computational cost is at least quadratic in the number of observations, often $O(n^3)$ computation and $O(n^2)$ memory, but there are various approximations with good scaling up properties.
- Further reading:
  - Bishop, Pattern Recognition and Machine Learning, Chapter 6.
  - Schölkopf and Smola, Learning with Kernels, 2001.
  - Rasmussen and Williams, Gaussian Processes for Machine Learning, 2006.

Smoothing and Nearest Neighbours

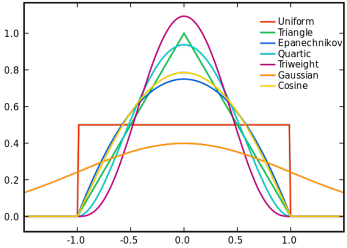
Nonlinear Methods

- Nonlinearity by data transformation: $x \mapsto \varphi(x)$ (explicit or implicit).
- A **global** approach. Decision function and optimal parameters can depend on training examples in the whole domain $\mathcal{X}$.
- Alternative approach: decision function $f(x)$ depends only on instances in the **local neighbourhood** of x.

Smoothing kernels

- Recall the plug-in generative classifier $f(x) = \text{argmax}_{l \in \{1,\ldots,K\}} \hat{\pi}_l \hat{g}_l(x)$
- What if we do not want to assume that the true class-l conditional density $g_l(x)$ takes any particular form (i.e., multivariate normal)?
- Use a **kernel density estimate**

$$\hat{g}_l(x) = \frac{1}{n_l} \sum_{i:\, y_i = l} \kappa(x - x_i)$$



$\underbrace{\text{smoothing (Parzen) kernel}}_{\text{local similarity}} \neq \underbrace{\text{positive-semidefinite (Mercer) kernel}}_{\text{inner product between features}}$

Smoothing kernels

- **Kernel density estimate**

$$\hat{g}_l(x) = \frac{1}{n_l} \sum_{i:\, y_i = l} \kappa(x - x_i)$$

- since $\hat{\pi}_l = \frac{n_l}{n}$, discrimination based on total similarity of x to instances in each of the classes:

$$f(x) = \underset{l \in \{1,\dots,K\}}{\operatorname{argmax}} \sum_{i:\, y_i = l} \kappa(x - x_i)$$

- Posterior class probabilities

$$\hat{\mathbb{P}}\left(Y = l | X = x\right) = \frac{\hat{\pi}_l \hat{g}_l(x)}{\sum_{j=1}^{K} \hat{\pi}_j \hat{g}_j(x)} = \frac{\sum_{i:\, y_i = l} \kappa(x - x_i)}{\sum_{j=1}^{n} \kappa(x - x_j)}$$

k-Nearest Neighbours

- Prediction at a data vector x is determined by the set $ne_{\mathrm{k}}(x)$ of k nearest neighbours of x among the training set.
- Classification: **majority vote** of the neighbours:

$$f_{\mathsf{kNN}}(x) = \operatorname*{argmax}_{l} \; |\{j \in ne_{\mathrm{k}}(x) : y_j = l\}|.$$

- Regression: average among the neighbours:

$$f_{\mathsf{kNN}}(x) = \frac{\sum_{j \in ne_{\mathrm{k}}(x)} y_j}{\mathrm{k}}.$$

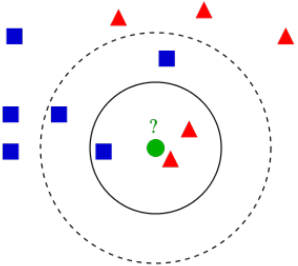
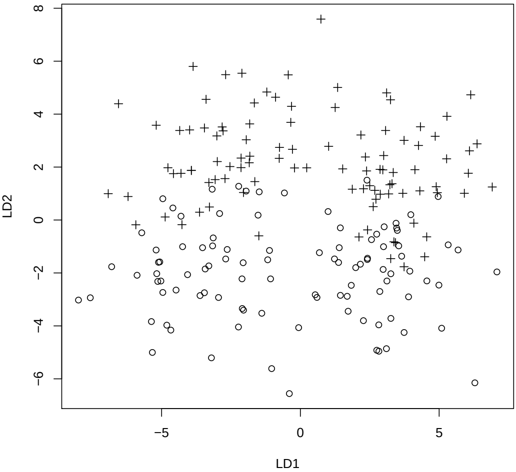


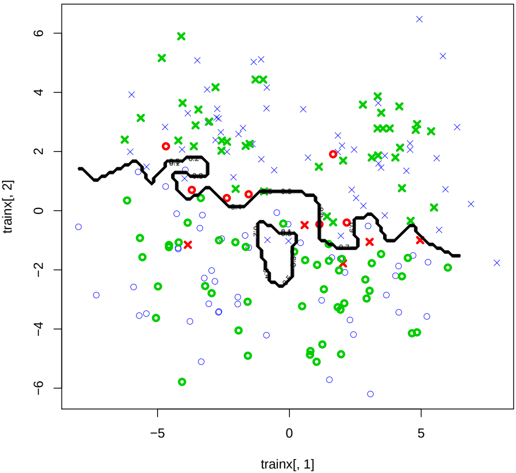
figure by A. Ajanki

k-Nearest Neighbour Demo



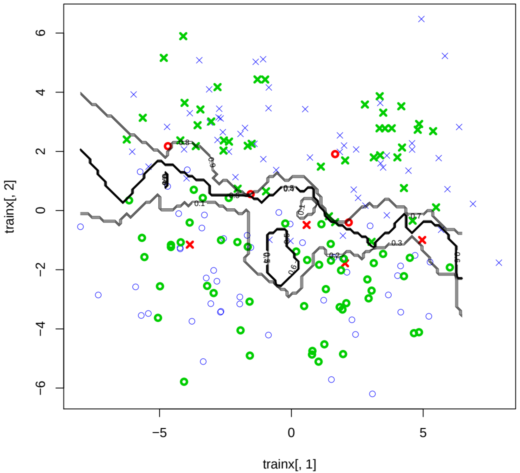
Data

k-Nearest Neighbour Demo



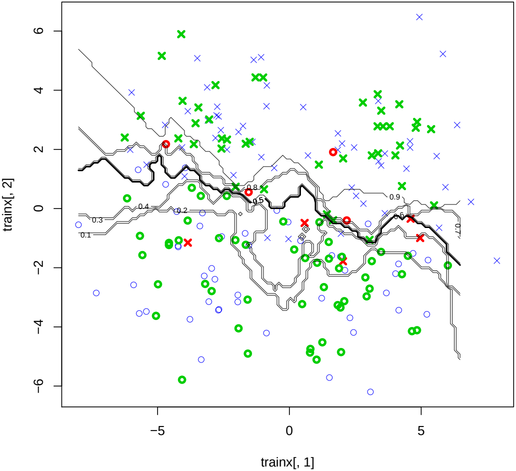
Result of 1NN

k-Nearest Neighbour Demo



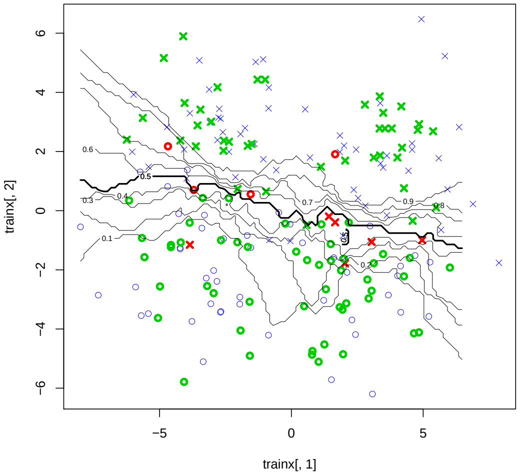
Result of 3NN

k-Nearest Neighbour Demo



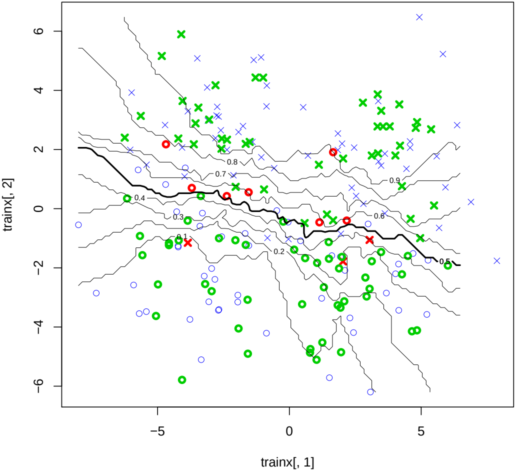
Result of 5NN

k-Nearest Neighbour Demo



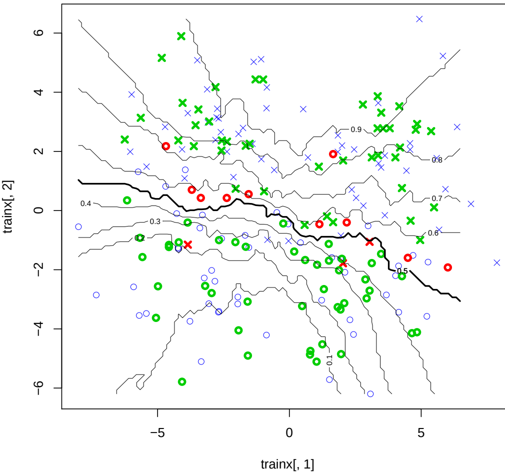
Result of 11NN

k-Nearest Neighbour Demo



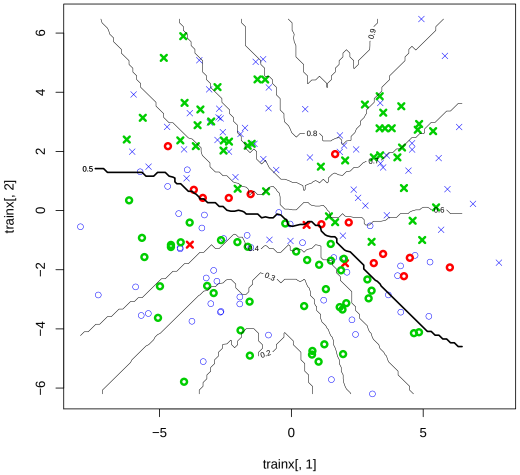
Result of 21NN

k-Nearest Neighbour Demo



Result of 31NN

k-Nearest Neighbour Demo



Result of 51NN

k-Nearest Neighbour Demo – R Code I

```
library(MASS)
## load crabs data
data(crabs)
ct <- as.numeric(crabs[,1])-1+2*(as.numeric(crabs[,2])-1)
## project to first two LD
cb.lda <- lda(log(crabs[,4:8]),ct)
cb.ldp <- predict(cb.lda)
x <- as.matrix(cb.ldp$x[,1:2])
y <- as.numeric(crabs[,2])-1
x <- x + rnorm(dim(x)[1]*dim(x)[2])*1.5
eqscplot(x,pch=2*y+1,col=1)
n      <- length(y)

#get training indices
i <- sample(rep(c(TRUE,FALSE),each=n/2),n,replace=FALSE)

kNN <- function(k,x,y,i,gridsize=100) {

  p      <- dim(x)[2]

  train  <- (1:n)[i]
  test   <- (1:n)[!i]
  trainx <- x[train,]
  trainy <- y[train]
  testx  <- x[test,]
  testy  <- y[test]

  trainn <- dim(trainx)[1]
  testn  <- dim(testx)[1]

  gridx1 <- seq(min(x[,1]),max(x[,2]),length=gridsize)
  gridx2 <- seq(min(x[,2]),max(x[,2]),length=gridsize)
  gridx  <- as.matrix(expand.grid(gridx1,gridx2))
  gridn  <- dim(gridx)[1]
```

k-Nearest Neighbour Demo – R Code II

```
  # calculate distances
  trainxx <- t((trainx*trainx) %*% matrix(1,p,1))
  testxx  <- (testx*testx)   %*% matrix(1,p,1)
  gridxx  <- (gridx*gridx)   %*% matrix(1,p,1)
  testtraindist <- matrix(1,testn,1) %*% trainxx +
    testxx %*% matrix(1,1,trainn) -
    2*(testx %*% t(trainx))
  gridtraindist <- matrix(1,gridn,1) %*% trainxx +
    gridxx %*% matrix(1,1,trainn) -
    2*(gridx %*% t(trainx))

  # predict
  testp <- numeric(testn)
  gridp <- numeric(gridn)
  for (j in 1:testn) {
    nearestneighbors <- order(testtraindist[j,])[1:k]
    testp[j] <- mean(trainy[nearestneighbors])
  }
  for (j in 1:gridn) {
    nearestneighbors <- order(gridtraindist[j,])[1:k]
    gridp[j] <- mean(trainy[nearestneighbors])
  }
  predy <- as.numeric(testp>.5)

  plot(trainx[,1],trainx[,2],pch=trainy*3+1,col=4,lwd=.5)
  points(testx[,1],testx[,2],pch=testy*3+1,col=2+(predy==testy),lwd=3)
  contour(gridx1,gridx2,matrix(gridp,gridsize,gridsize),
          levels=seq(.1,.9,.1),lwd=.5,add=TRUE)
  contour(gridx1,gridx2,matrix(gridp,gridsize,gridsize),
          levels=c(.5),lwd=2,add=TRUE)
}
```

Asymptotic Performance of 1NN

- Let $(x_i, y_i)_{i=1}^n$ be training data where $x_i \in \mathbb{R}^p$ and $y_i \in \{1, 2, ..., K\}$.
- We define

$$f_{\text{Bayes}}(x) := \arg\max_{l \in \{1,\ldots,K\}} \pi_l g_l(x),$$
$$f_{\text{1NN}}^{(n)}(x) := y_j, \text{ s.t. } x_j \text{ is the nearest neigbour of } x.$$

- The (optimal) Bayes risk and 1NN risk are:

$$R_{\text{Bayes}} = \mathbb{E}\left[\mathbf{1}\left(Y \neq f_{\text{Bayes}}(X)\right)\right]$$
$$R_{\text{1NN}}^{(n)} = \mathbb{E}\left[\mathbf{1}\left(Y \neq f_{\text{1NN}}^{(n)}(X)\right)\right]$$

- As $n \to \infty$, $R_{\text{1NN}}^{(n)} \to R_{\text{1NN}}$, where

$$\boxed{R_{\text{Bayes}} \leq R_{\text{1NN}} \leq 2R_{\text{Bayes}} - \frac{K}{K-1} R_{\text{Bayes}}^2.}$$

k-Nearest Neighbours – Discussion

- Simple and essentially model-free, i.e., weaker assumptions than LDA, Naïve Bayes and logistic regression.
- Not useful for understanding relationships between attributes and class predictions.
- Sensitive to the choice of distance and to the choice of the number of neighbours k
- High computational cost:
  - Need to store **all** training data.
  - Need to compare each test data vector to **all** training data.
  - Need **a lot of data** in high dimensions.
- Mitigation: compute approximate nearest neighbours, using **kd-trees**, **cover trees**, **random forests**.