Vietnam National University - Ho Chi Minh

Optimization, Machine Learning and Kernel Methods.

Machine Learning - Kernel Methods

Marco Cuturi - Princeton University

Regression, Classification and other Supervised Tasks

- **Two associated random variables**
  - A random variable x, taking values in $\mathcal{X}$,
  - A random variable y, taking values in $\mathcal{Y}$.
- **Two samples of (x, y) i.i.d. distributed from their joint law**
  - $\{(\mathbf{x}_1, \mathbf{y}_1), \cdots, (\mathbf{x}_n, \mathbf{y}_n)\}$, n couples of $\mathcal{X} \times \mathcal{Y}$.

Challenge: **predict** $\mathbf{y}$ when given only $\mathbf{x}$.

- In practice, **find** a function $\mathcal{X} \to \mathcal{Y}$ for which $f(\mathbf{x})$ is not too different from y on average.

Binary Classification

- $\mathcal{Y} = \{-1, 1\}$.
- f needs to be a functions that, given $\mathbf{x}$ predicts a label,

$$f : \mathcal{X} \mapsto \{-1, 1\}$$

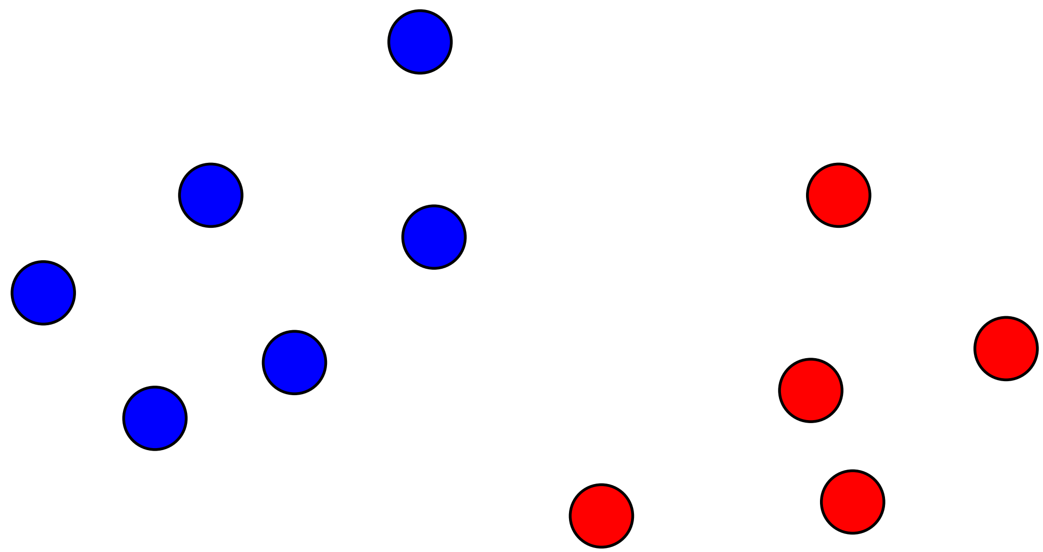
  of course, many possible choices for f's shape.
- We review here **linear** hyperplanes in $\mathcal{X} = \mathbb{R}^d$ first.
- We represent it in $\mathbb{R}^2$ for simplicity.

Next slides will cover an important algorithm, the **SVM** algorithm

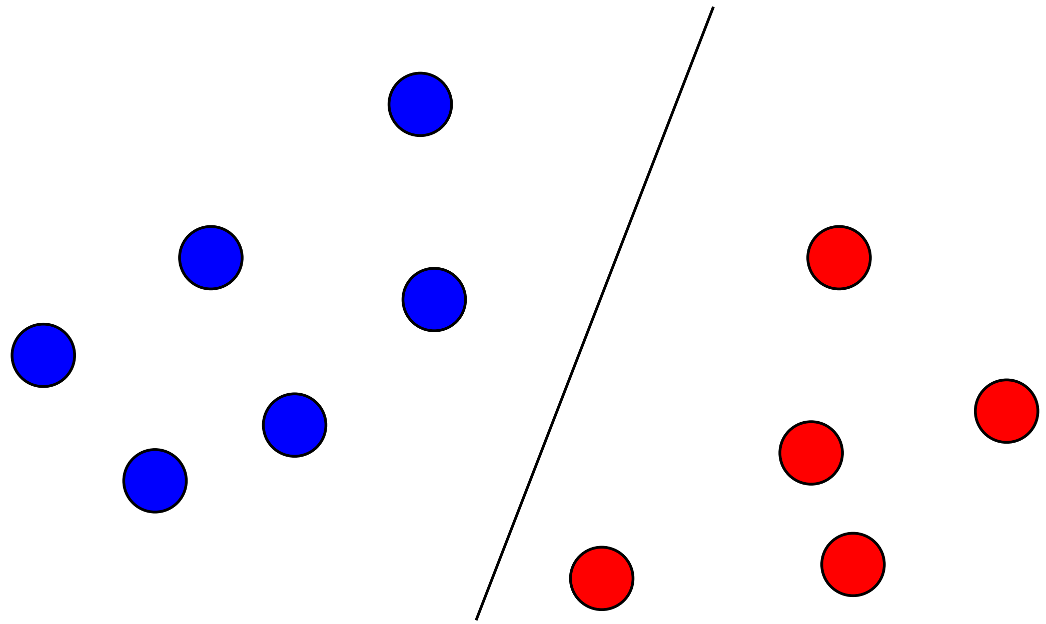
- this algorithm can be naturally expressed in terms of *kernels*. we review later other algorithms for which this is also the case.

thanks to Jean-Philippe Vert for many of the following figures and slides.

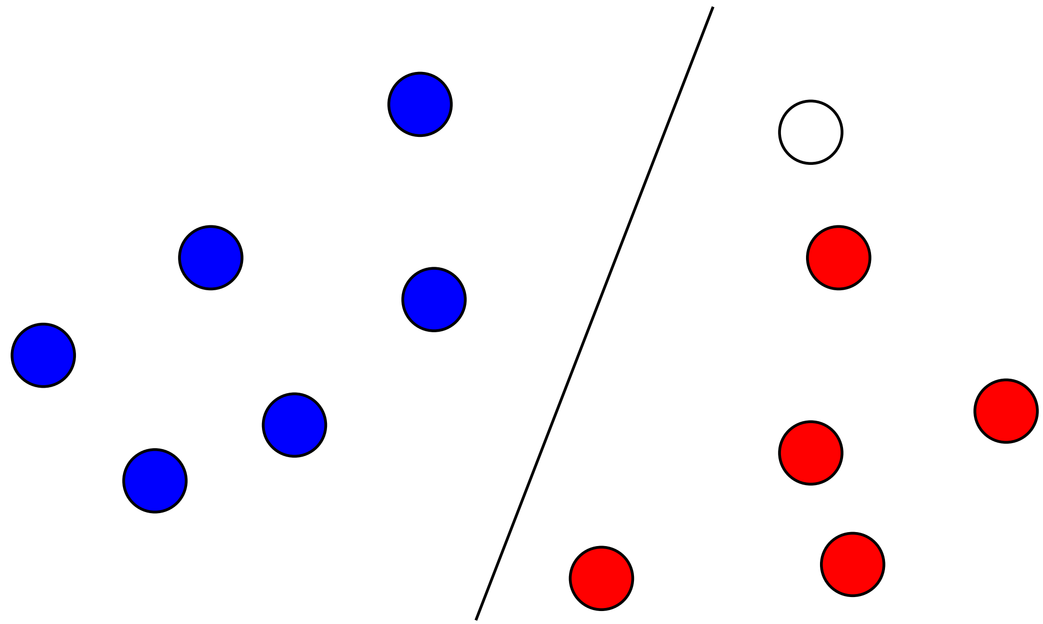
Linear classifier, some degrees of freedom



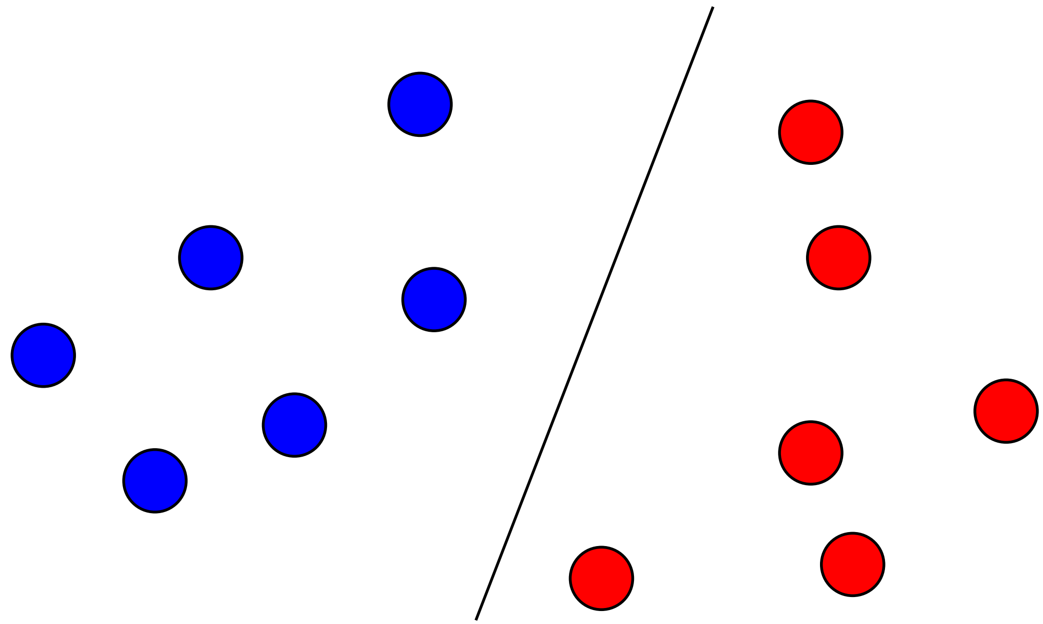
Linear classifier, some degrees of freedom



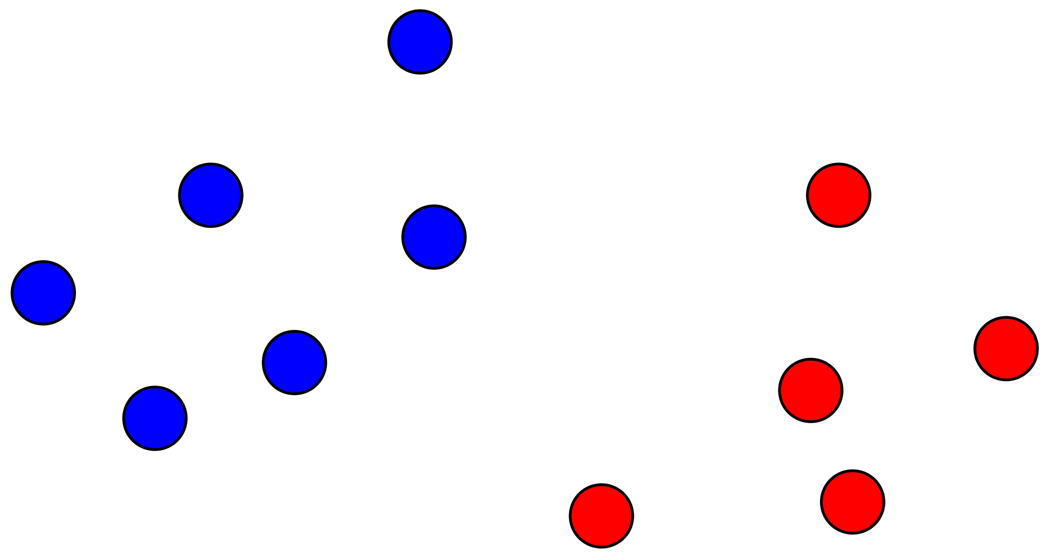
Linear classifier, some degrees of freedom



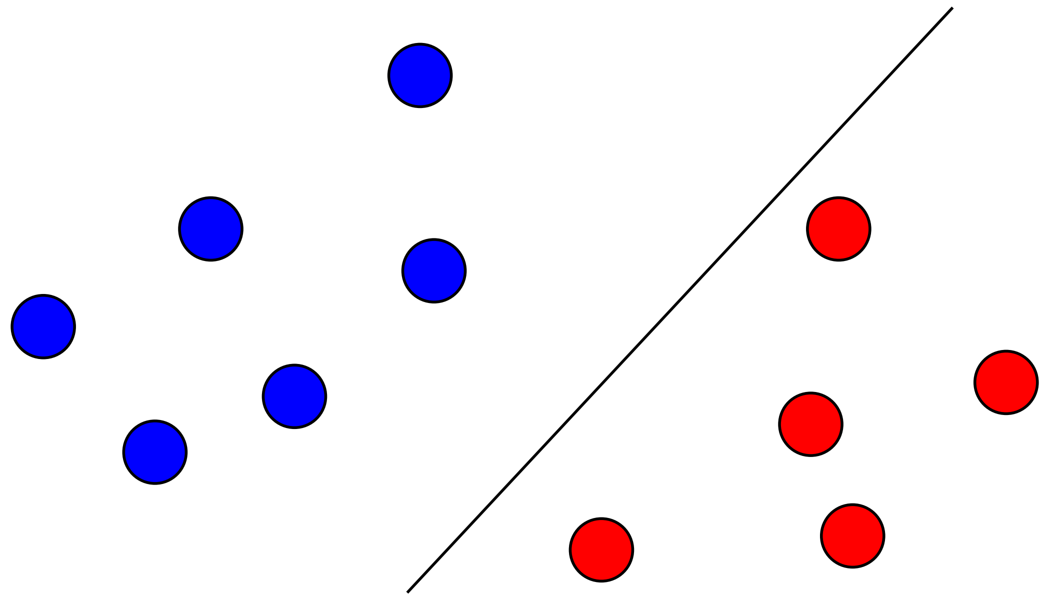
Linear classifier, some degrees of freedom



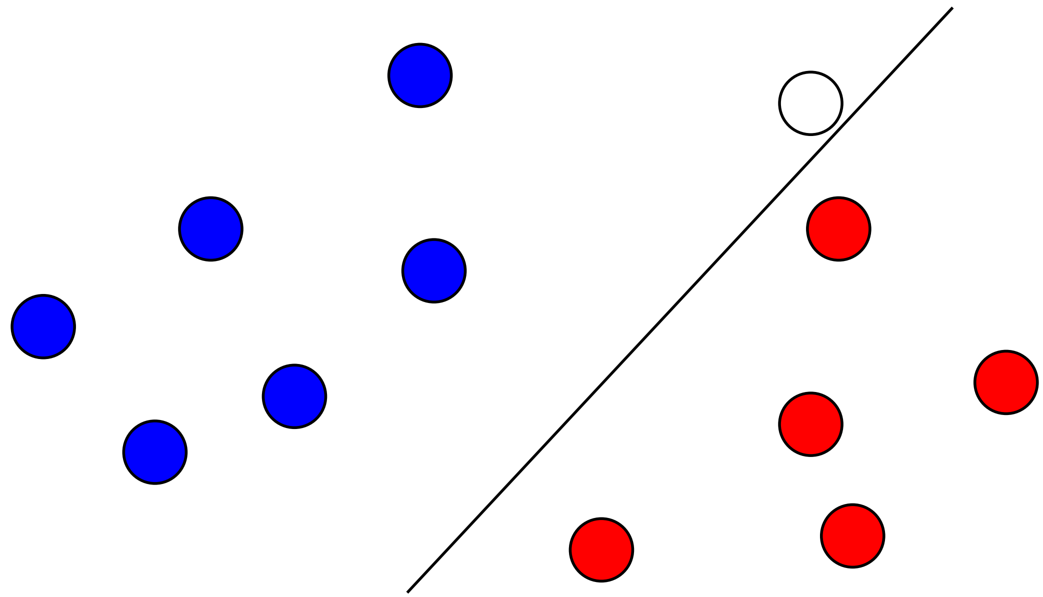
Linear classifier, some degrees of freedom



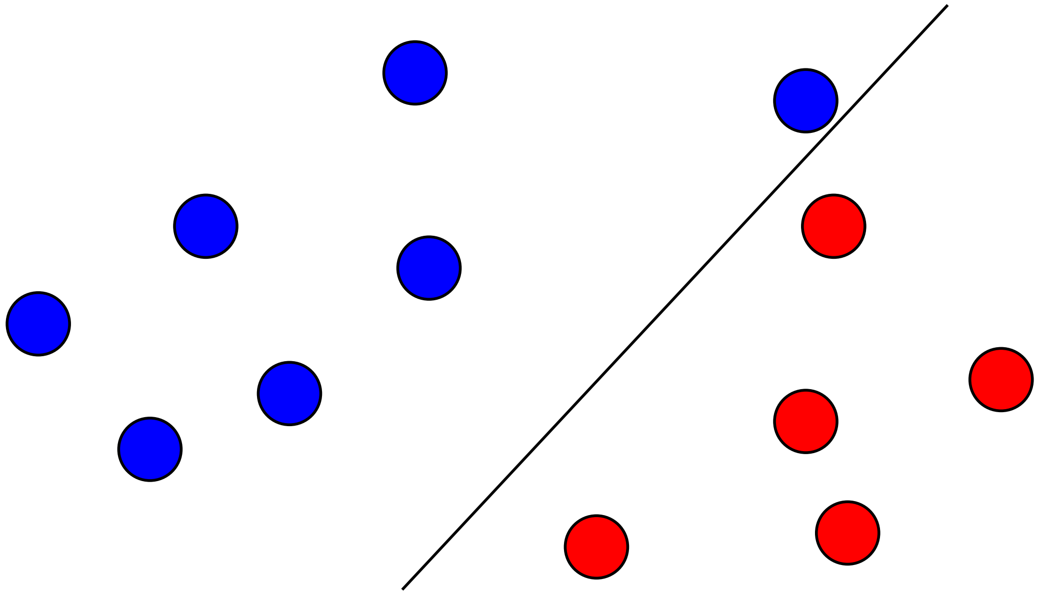
Linear classifier, some degrees of freedom



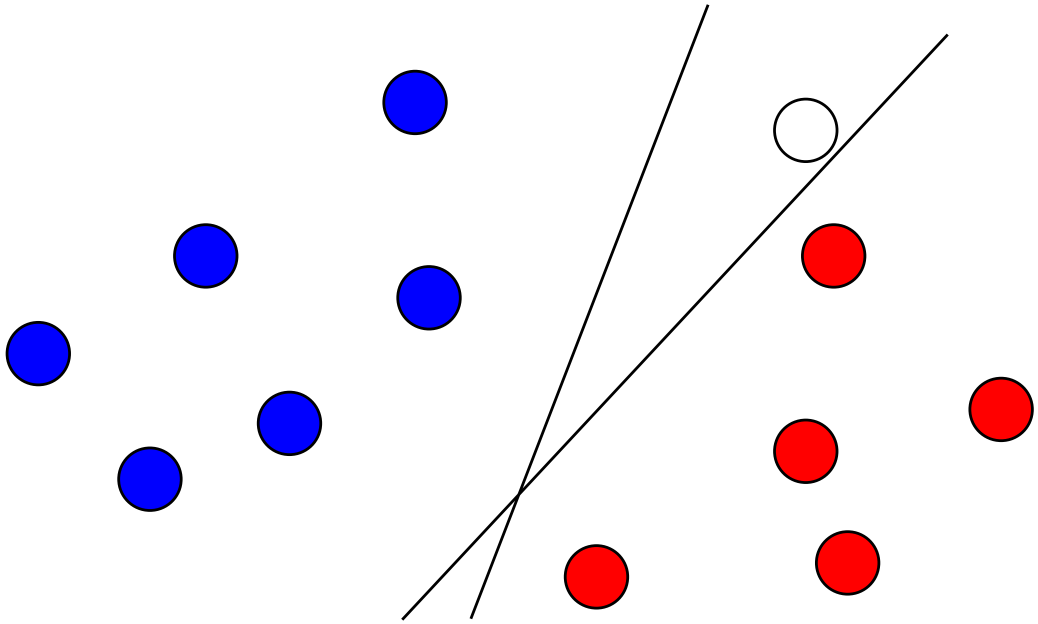
Linear classifier, some degrees of freedom



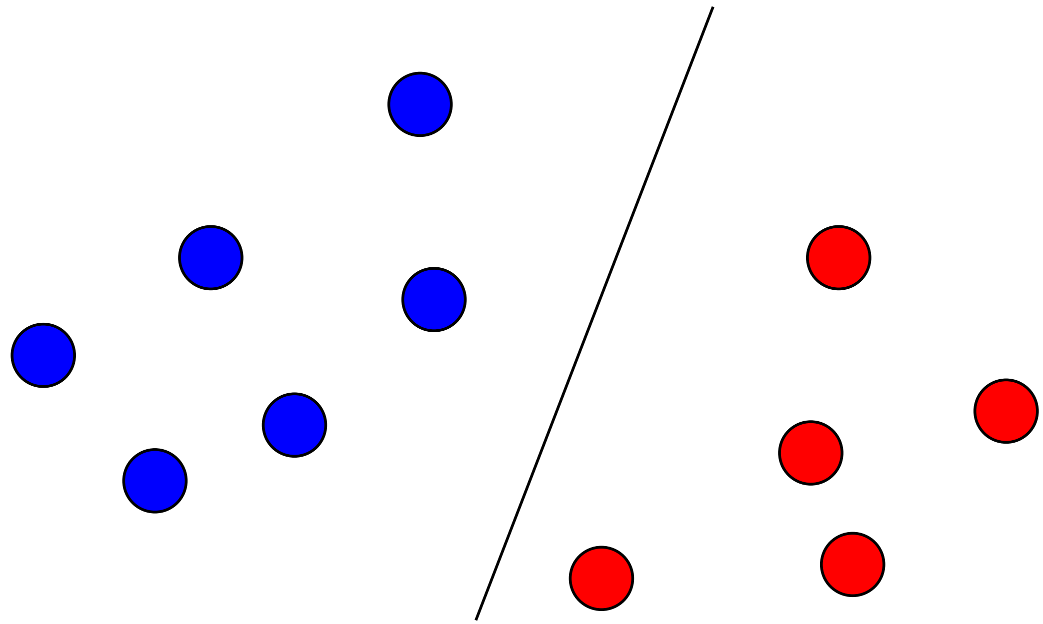
Linear classifier, some degrees of freedom



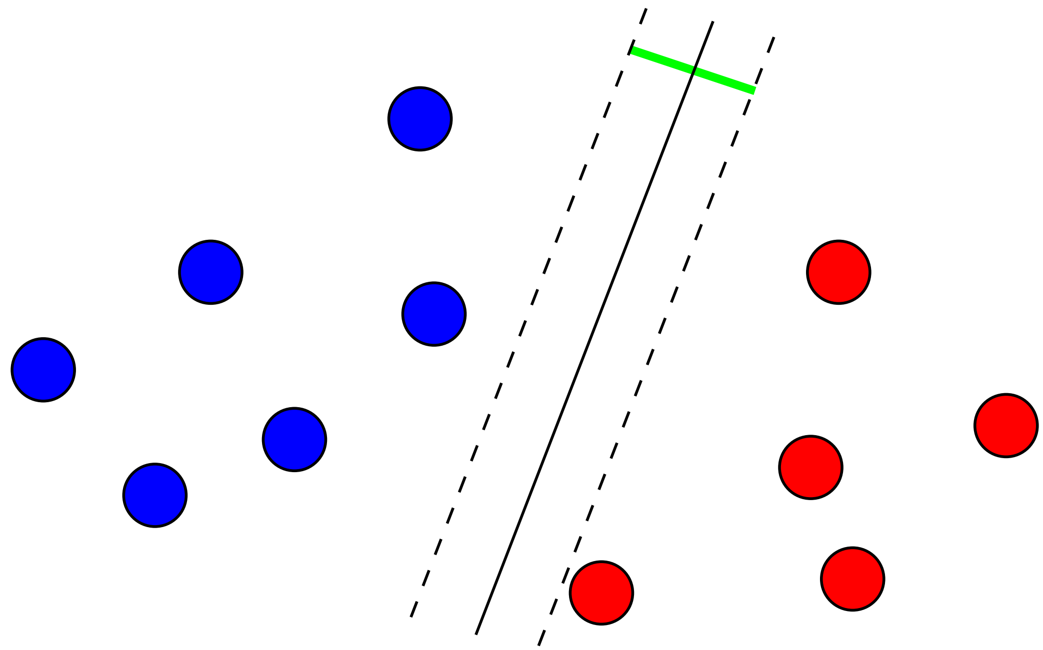
Which one is better?



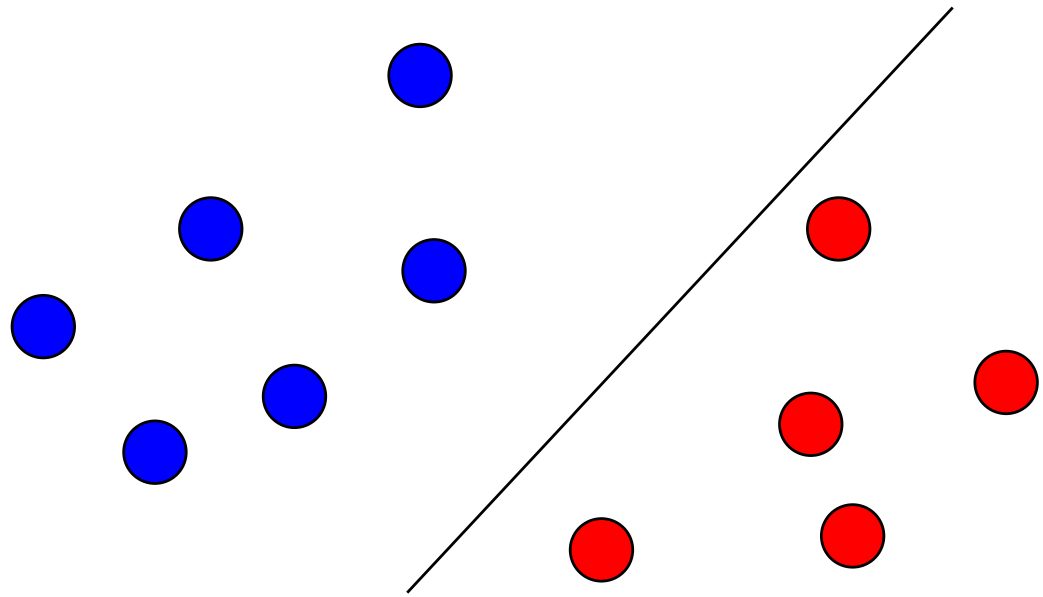
A criterion to select a linear classifier: the margin



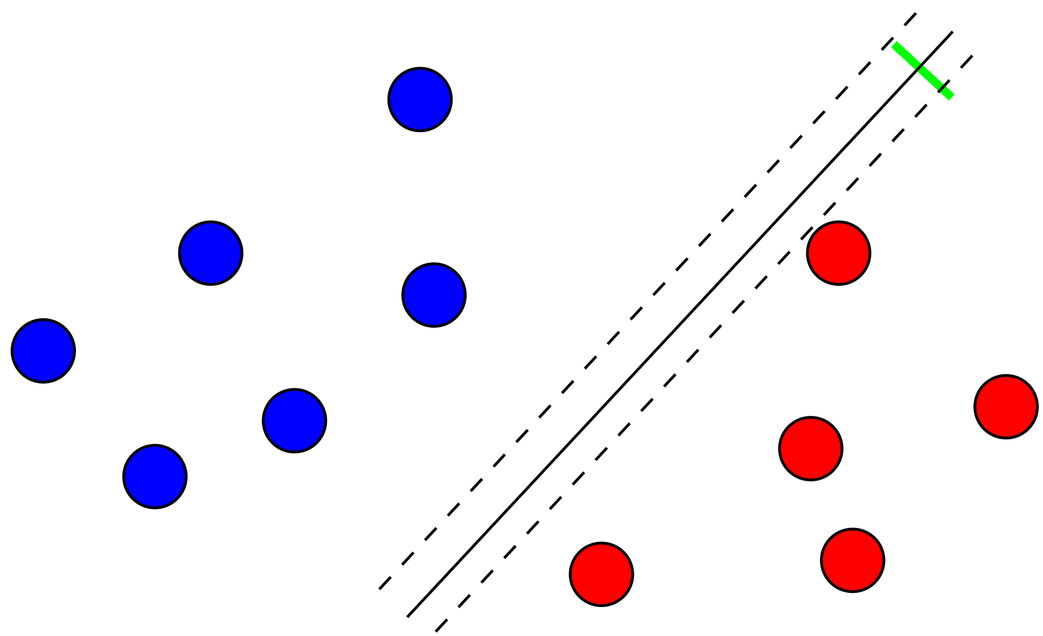
A criterion to select a linear classifier: the margin



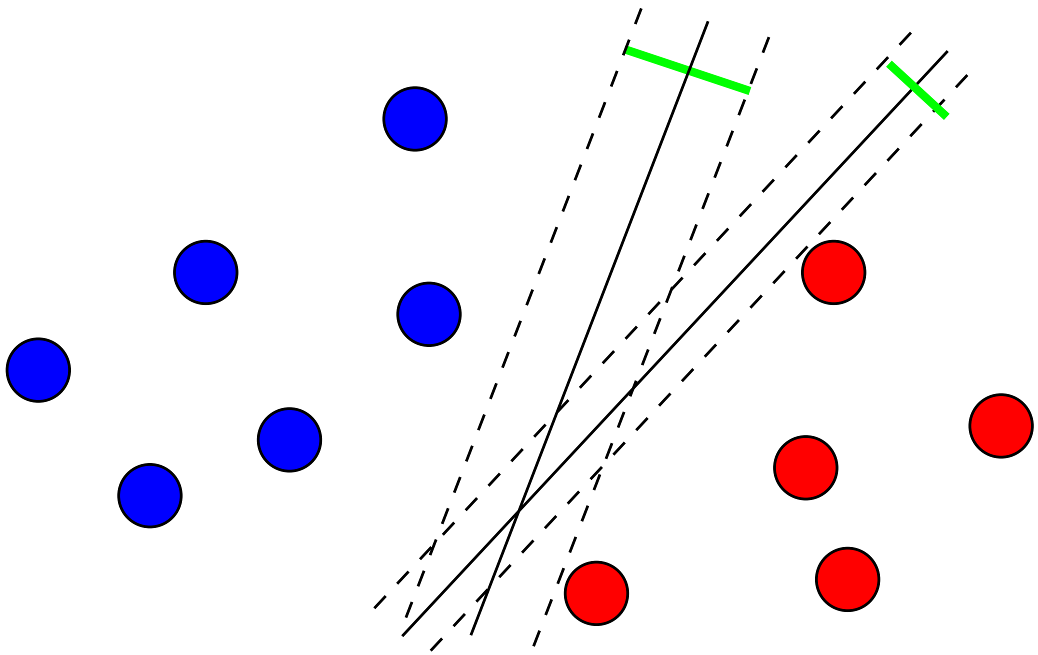
A criterion to select a linear classifier: the margin



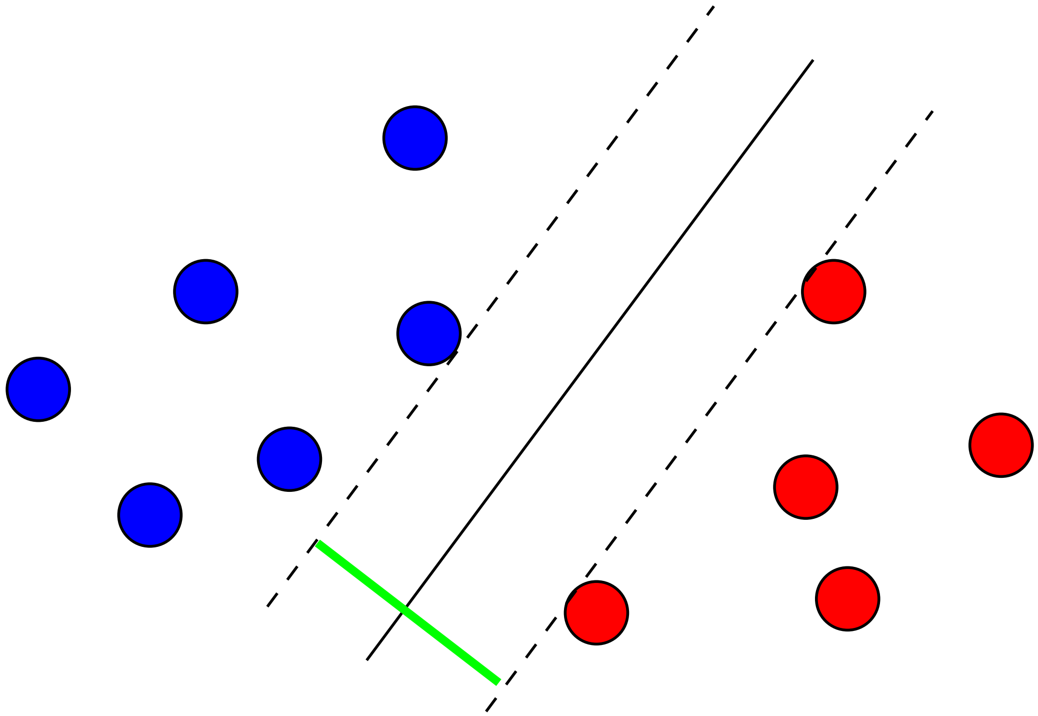
A criterion to select a linear classifier: the margin



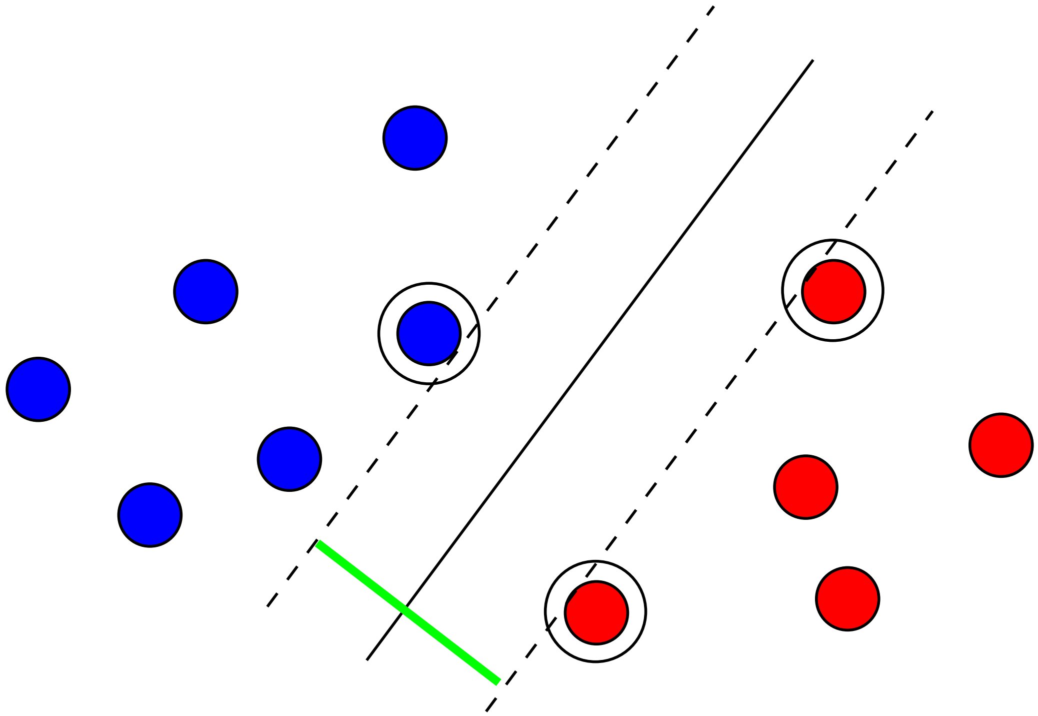
A criterion to select a linear classifier: the margin



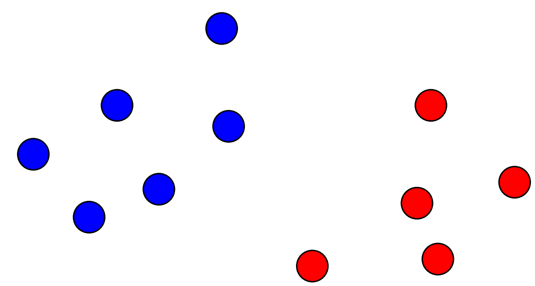
Largest Margin Linear Classifier



Support Vectors with Large Margin



In equations



- The **training set** is a finite set of n data/class pairs:

$$\mathcal{T} = \{(\mathbf{x}_1, \mathbf{y}_1), \ldots, (\mathbf{x}_N, \mathbf{y}_N)\},$$

  where $\mathbf{x}_i \in \mathbb{R}^d$ and $\mathbf{y}_i \in \{-1, 1\}$.

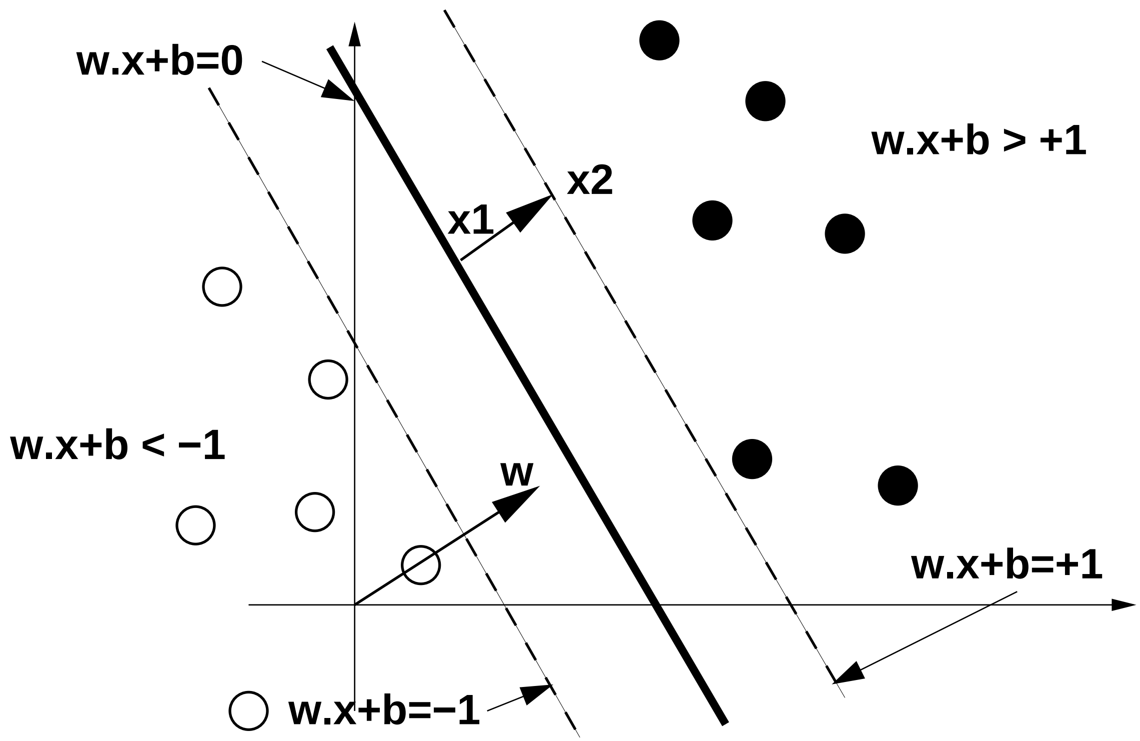
- We assume (for the moment) that the data are **linearly separable**, i.e., that there exists $(\mathbf{w}, b) \in \mathbb{R}^d \times \mathbb{R}$ such that:

$$\begin{cases} \mathbf{w}^T \mathbf{x}_i + b > 0 & \text{if } \mathbf{y}_i = 1, \\ \mathbf{w}^T \mathbf{x}_i + b < 0 & \text{if } \mathbf{y}_i = -1. \end{cases}$$

How to find the largest separating hyperplane?

For the linear classifier $f(\mathbf{x}) = \mathbf{w}^T\mathbf{x} + b$ consider the *interstice* defined by the hyperplanes

- $f(\mathbf{x}) = \mathbf{w}^T\mathbf{x} + b = +1$
- $f(\mathbf{x}) = \mathbf{w}^T\mathbf{x} + b = -1$



The margin is $2/||\mathbf{w}||$

- Indeed, the points $\mathbf{x}_1$ and $\mathbf{x}_2$ satisfy:

$$\begin{cases} \mathbf{w}^T\mathbf{x}_1 + b = 0, \\ \mathbf{w}^T\mathbf{x}_2 + b = 1. \end{cases}$$

- By subtracting we get $\mathbf{w}^T(\mathbf{x}_2 - \mathbf{x}_1) = 1$, and therefore:

$$\gamma = 2||\mathbf{x}_2 - \mathbf{x}_1|| = \frac{2}{||\mathbf{w}||}.$$

where γ is the margin.

All training points should be on the appropriate side

- For positive examples ($y_i = 1$) this means:

$$\mathbf{w}^T \mathbf{x}_i + b \geq 1$$

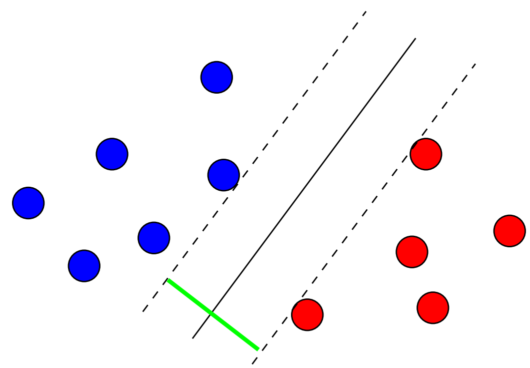
- For negative examples ($y_i = -1$) this means:

$$\mathbf{w}^T \mathbf{x}_i + b \leq -1$$

- in both cases:

$$\forall i = 1, \ldots, n, \qquad \mathbf{y}_i \left(\mathbf{w}^T \mathbf{x}_i + b \right) \geq 1$$

Finding the optimal hyperplane



- Find $(\mathbf{w}, b)$ which minimize:

$$||\mathbf{w}||^2$$

under the constraints:

$$\forall i = 1, \ldots, n, \qquad \mathbf{y}_i \left(\mathbf{w}^T \mathbf{x}_i + b\right) - 1 \geq 0.$$

This is a classical quadratic program on $\mathbb{R}^{d+1}$
linear constraints - **quadratic objective**

Lagrangian

- In order to minimize:

$$\frac{1}{2}||\mathbf{w}||^2$$

under the constraints:

$$\forall i = 1, \ldots, n, \qquad y_i \left(\mathbf{w}^T \mathbf{x}_i + b\right) - 1 \geq 0.$$

- introduce **one dual variable α_i for each constraint**,
- namely, for **each training point**. The Lagrangian is, for $\alpha \succeq 0$,

$$L(\mathbf{w}, b, \alpha) = \frac{1}{2}||\mathbf{w}||^2 - \sum_{i=1}^{n} \alpha_i \left(y_i \left(\mathbf{w}^T \mathbf{x}_i + b\right) - 1\right).$$

The Lagrange dual function

$$g(\alpha) = \inf_{\mathbf{w}\in\mathbb{R}^d, b\in\mathbb{R}} \left\{ \frac{1}{2}\|\mathbf{w}\|^2 - \sum_{i=1}^{n} \alpha_i \left(y_i \left(\mathbf{w}^T \mathbf{x}_i + b \right) - 1 \right) \right\}$$

is only defined when

$$\mathbf{w} = \sum_{i=1}^{n} \alpha_i \mathbf{y}_i \mathbf{x}_i, \quad (\text{ derivating w.r.t } \mathbf{w}) \quad (*)$$

$$0 = \sum_{i=1}^{n} \alpha_i \mathbf{y}_i, \quad (\text{derivating w.r.t } b) \quad (**)$$

substituting $(*)$ in g, and using $(**)$ as a constraint, get the dual function $g(\alpha)$.

- To solve the dual problem, **maximize** g w.r.t. α.
- Strong duality holds. KKT gives us $\alpha_i(\mathbf{y}_i \left(\mathbf{w}^T\mathbf{x}_i + b\right) - 1) = 0$, either $\alpha_i = 0$ or $\mathbf{y}_i \left(\mathbf{w}^T\mathbf{x}_i + b\right) = 1$.
- $\alpha_i \neq 0$ **only** for points on the support hyperplanes $\{(\mathbf{x}, \mathbf{y}) | \, \mathbf{y}_i(\mathbf{w}^T\mathbf{x}_i + b) = 1\}$.

Dual optimum

The dual problem is thus

$$\begin{array}{ll} \text{maximize} & g(\alpha) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j \mathbf{x}_i^T \mathbf{x}_j \\ \text{such that} & \alpha \succeq 0, \sum_{i=1}^{n} \alpha_i \mathbf{y}_i = 0. \end{array}$$

This is a quadratic program on $\mathbb{R}^n$, with *box constraints*.
α^* can be found efficiently using dedicated optimization softwares

Recovering the optimal hyperplane

- Once α^* is found, we recover $(\mathbf{w}^T, b^*)$ corresponding to the optimal hyperplane.
- $\mathbf{w}^T$ is given by $\mathbf{w}^T = \sum_{i=1}^n y_i \alpha_i \mathbf{x}_i^T$,
- b^* is given by the conditions on the support vectors $\alpha_i > 0$, $\mathbf{y}_i(\mathbf{w}^T \mathbf{x}_i + b) = 1$,

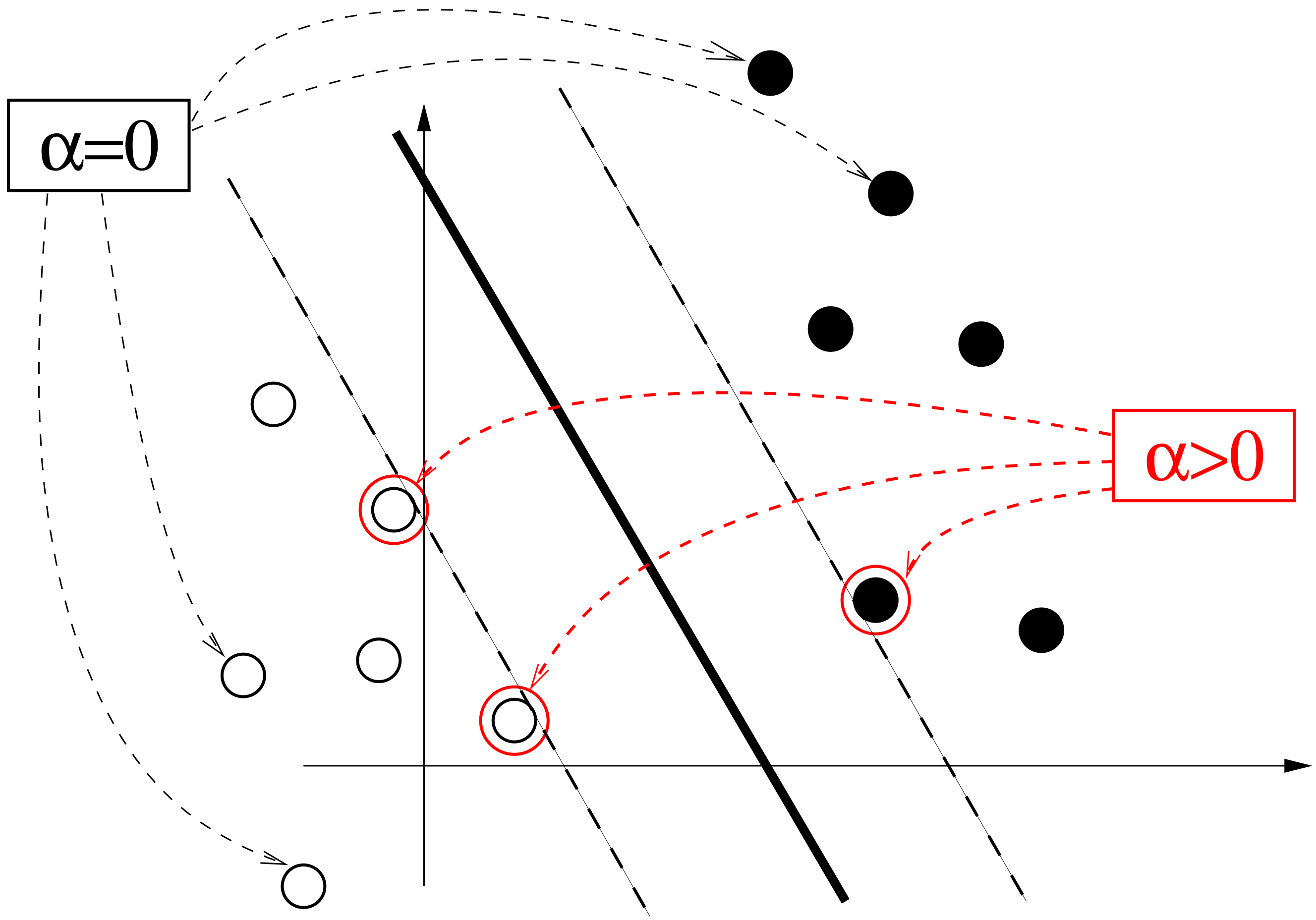
$$b^* = -\frac{1}{2}\left(\min_{\mathbf{y}_i=1,\alpha_i>0}(\mathbf{w}^T\mathbf{x}_i) + \max_{\mathbf{y}_i=-1,\alpha_i>0}(\mathbf{w}^T\mathbf{x}_i)\right)$$

- the **decision function** is therefore:

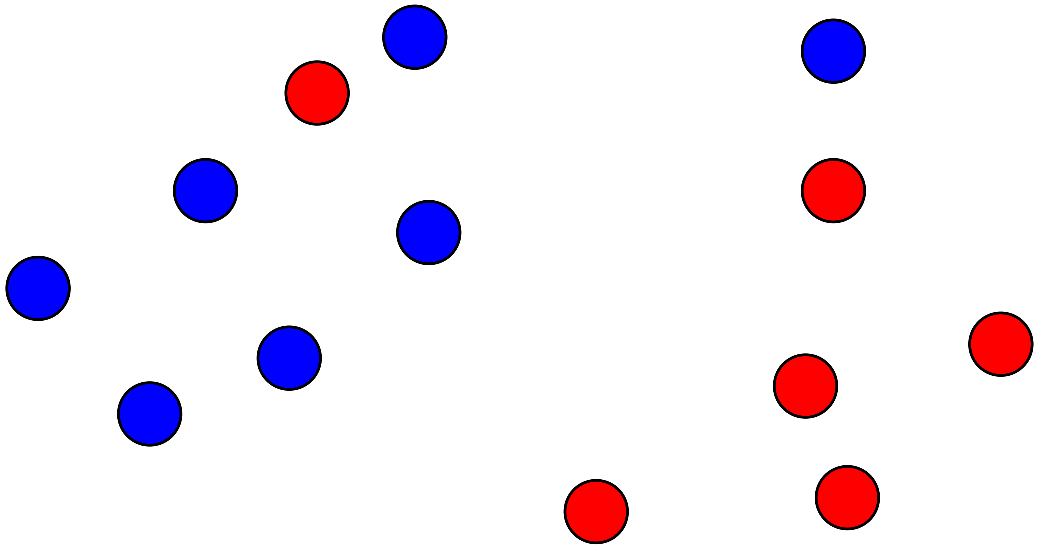
$$\begin{aligned} f^*(\mathbf{x}) &= \mathbf{w}^T\mathbf{x} + b^* \\ &= \sum_{i=1}^n y_i \alpha_i \mathbf{x}_i^T \mathbf{x} + b^*. \end{aligned}$$

- Here the **dual** solution gives us directly the **primal** solution.

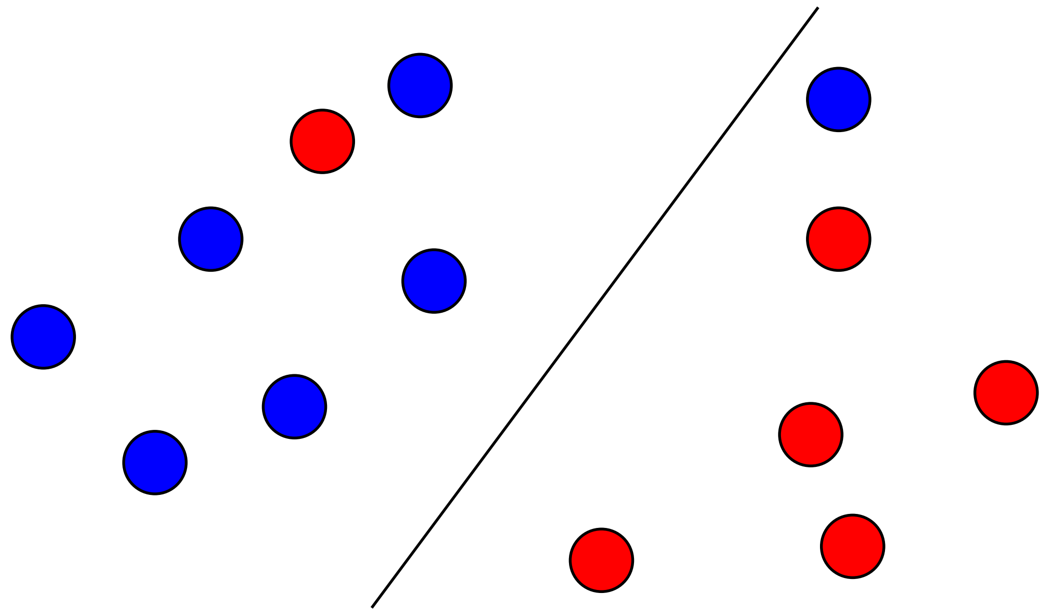
Interpretation: support vectors



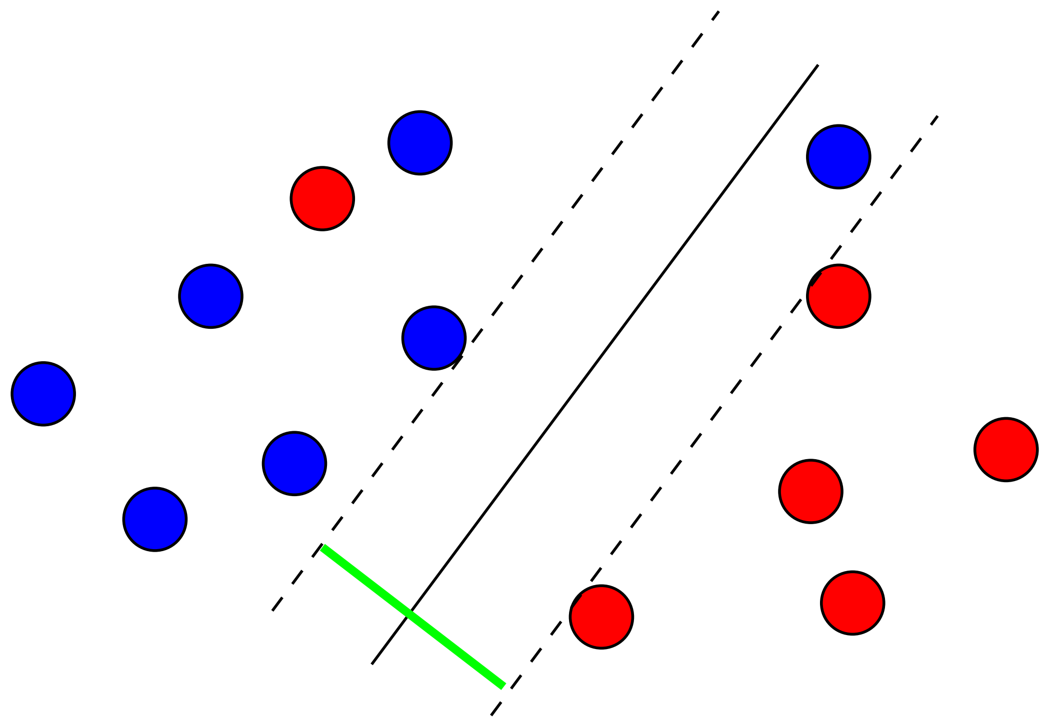
What happens when the data is not linearly separable?



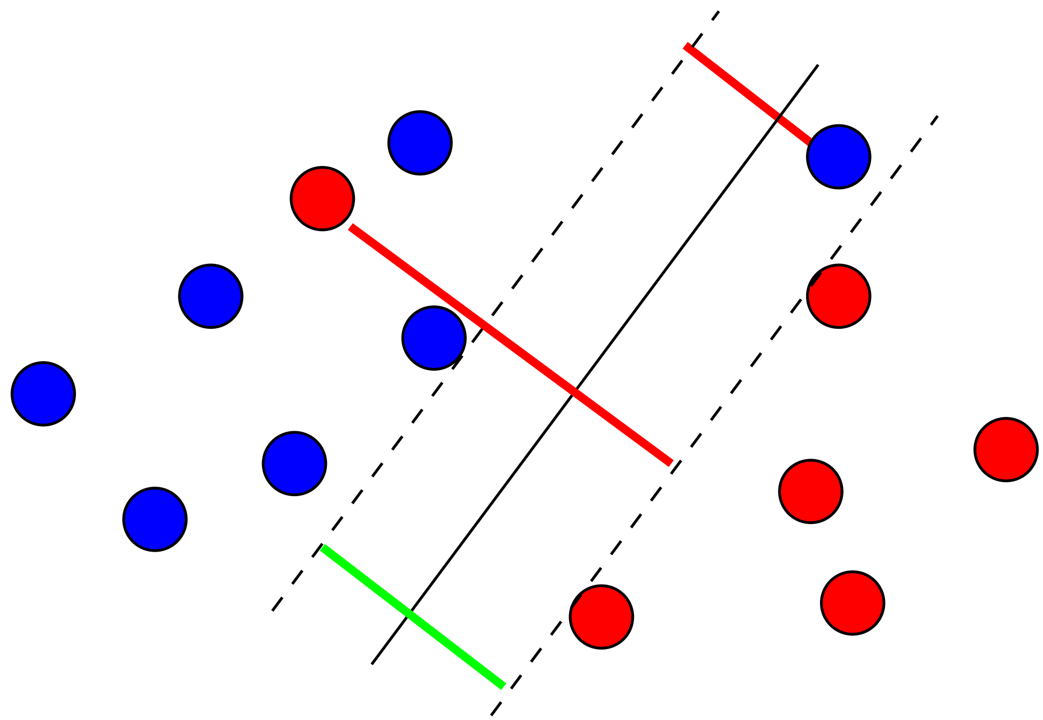
What happens when the data is not linearly separable?



What happens when the data is not linearly separable?



What happens when the data is not linearly separable?

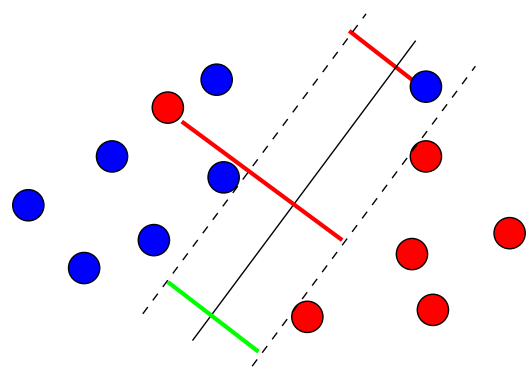


Soft-margin SVM

- Find a trade-off between **large margin** and **few errors**.
- Mathematically:

$$\min_f \left\{ \frac{1}{\mathrm{margin}(f)} + C \times \mathrm{errors}(f) \right\}$$

- C is a parameter



Soft-margin SVM formulation

- The **margin** of a labeled point $(\mathbf{x}, \mathbf{y})$ is

$$\text{margin}(\mathbf{x}, \mathbf{y}) = \mathbf{y}\left(\mathbf{w}^T \mathbf{x} + b\right)$$

- The **error** is
  - 0 if $\text{margin}(\mathbf{x}, \mathbf{y}) > 1$,
  - $1 - \text{margin}(\mathbf{x}, \mathbf{y})$ otherwise.

- The soft margin SVM solves:

$$\min_{\mathbf{w}, b} \{\|\mathbf{w}\|^2 + C \sum_{i=1}^{n} \max\{0, 1 - \mathbf{y}_i \left(\mathbf{w}^T \mathbf{x}_i + b\right)\}$$

- $c(u, y) = \max\{0, 1 - yu\}$ is known as the **hinge loss**.

- $c(\mathbf{w}^T \mathbf{x}_i + b, \mathbf{y}_i)$ associates a mistake cost to the decision $\mathbf{w}, b$ for example $\mathbf{x}_i$.

Dual formulation of soft-margin SVM

- The soft margin SVM program

$$\min_{\mathbf{w},b}\{\|\mathbf{w}\|^2 + C\sum_{i=1}^{n}\max\{0, 1-\mathbf{y}_i\left(\mathbf{w}^T\mathbf{x}_i+b\right)\}$$

can be rewritten as

$$\begin{array}{ll}\text{minimize} & \|\mathbf{w}\|^2 + C\sum_{i=1}^{n}\xi_i\\ \text{such that} & \mathbf{y}_i\left(\mathbf{w}^T\mathbf{x}_i+b\right) \geq 1-\xi_i\end{array}$$

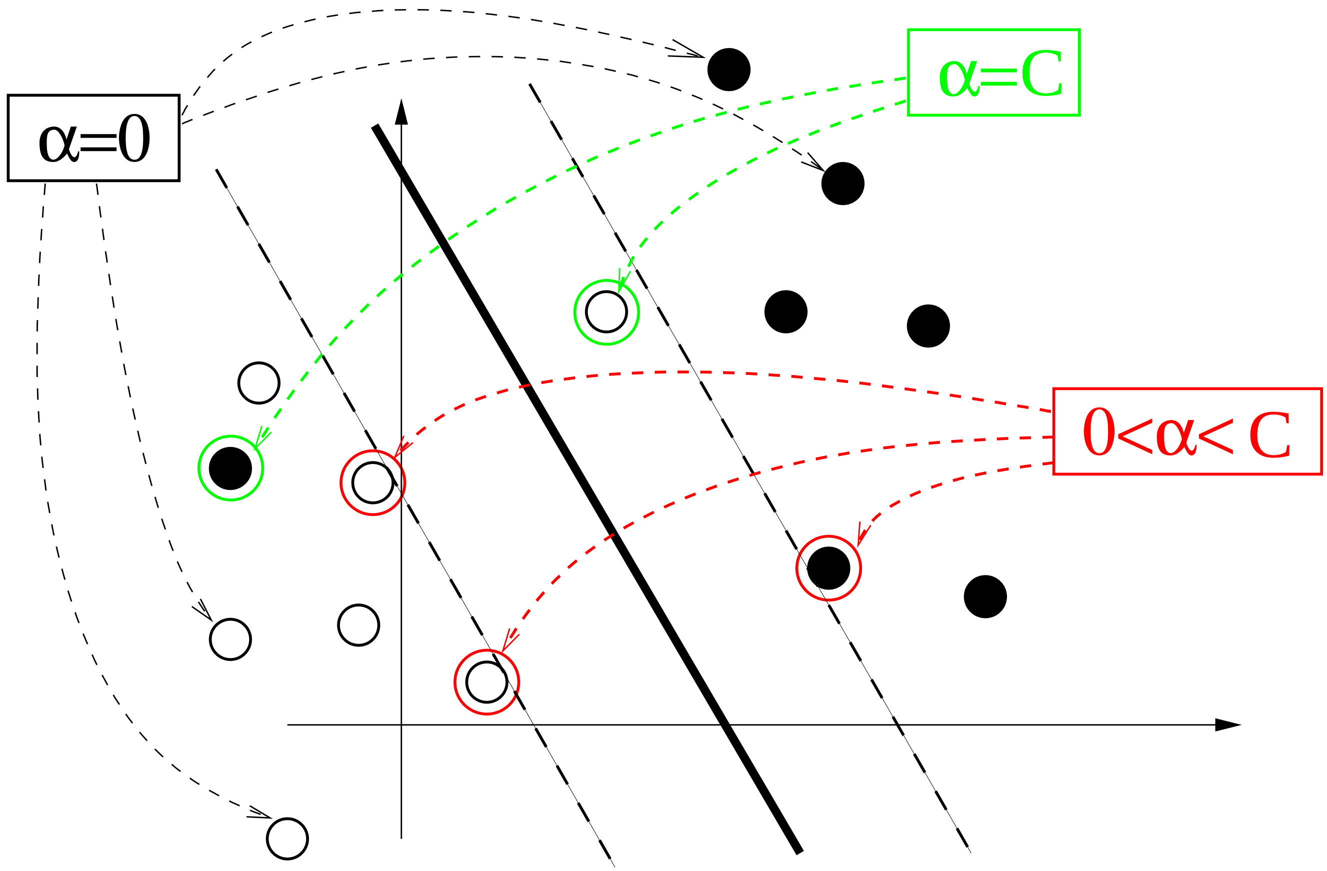
- In that case the dual function

$$g(\alpha) = \sum_{i=1}^{n}\alpha_i - \frac{1}{2}\sum_{i,j=1}^{n}\alpha_i\alpha_j\mathbf{y}_i\mathbf{y}_j\mathbf{x}_i^T\mathbf{x}_j,$$

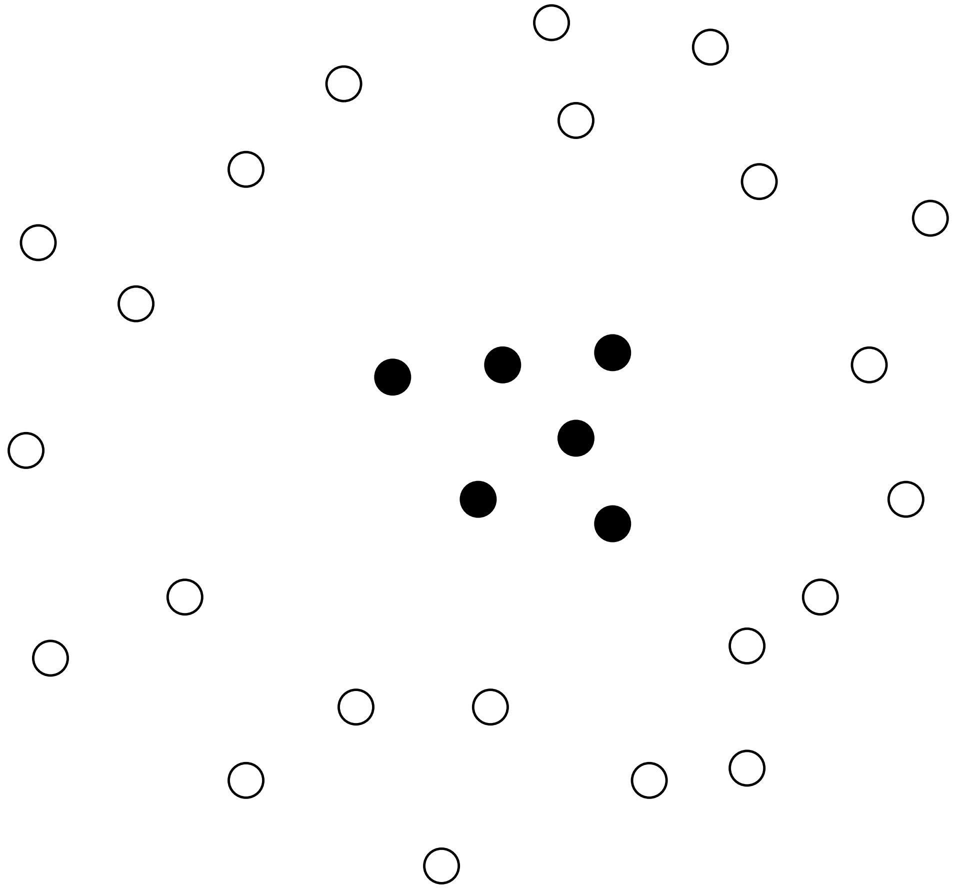
which is finite under the constraints:

$$\begin{cases}0 \leq \alpha_i \underline{\leq} \mathbf{C}, & \text{for } i = 1,\ldots,n\\ \sum_{i=1}^{n}\alpha_i\mathbf{y}_i = 0.\end{cases}$$

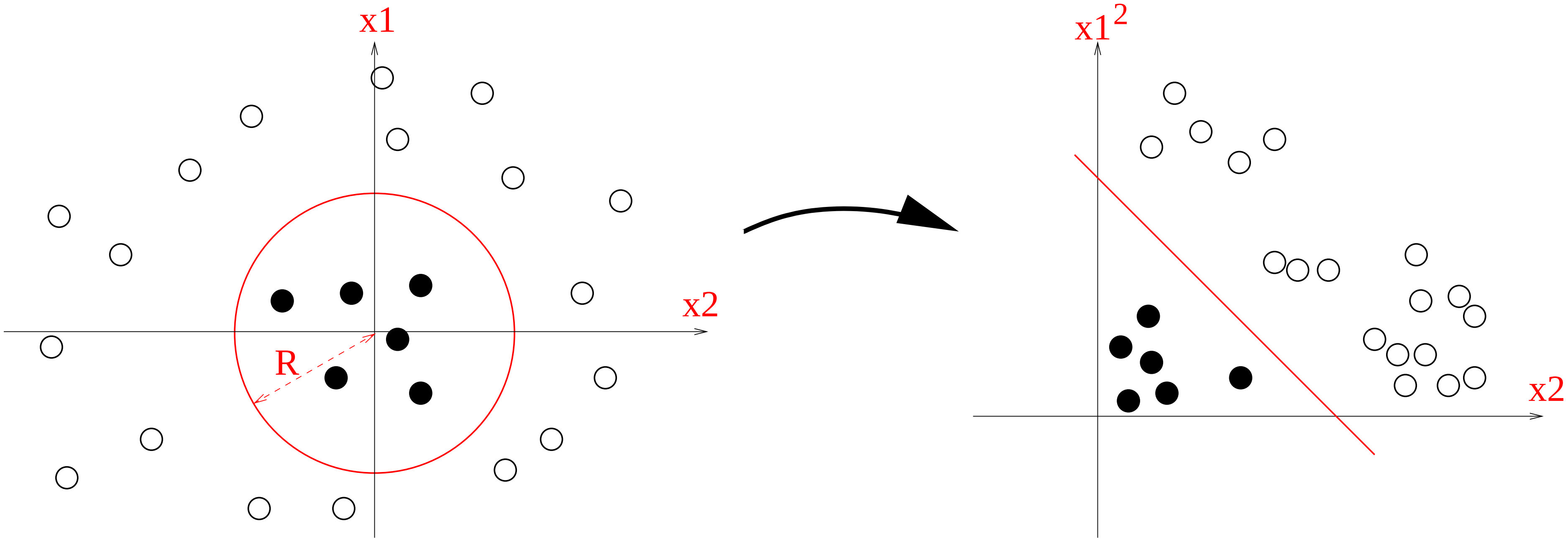
Interpretation: bounded and unbounded support vectors



Sometimes linear classifiers are not interesting



Solution: non-linear mapping to a feature space



Let $\phi(\mathbf{x}) = (x_1^2, x_2^2)'$, $\mathbf{w} = (1,1)'$ and $b = 1$. Then the decision function is:

$$f(\mathbf{x}) = x_1^2 + x_2^2 - R^2 = \langle \mathbf{w}, \phi(\mathbf{x}) \rangle + b,$$

Kernel trick for SVM's

- use a mapping ϕ from $\mathcal{X}$ to a feature space,
- which corresponds to the **kernel** k:

$$\forall \mathbf{x}, \mathbf{x}' \in \mathcal{X}, \quad k(\mathbf{x}, \mathbf{x}') = \langle \phi(\mathbf{x}), \phi(\mathbf{x}') \rangle$$

- Example: if $\phi(\mathbf{x}) = \phi\left(\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}\right) = \begin{bmatrix} x_1^2 \\ x_2^2 \end{bmatrix}$, then

$$k(\mathbf{x}, \mathbf{x}') = \langle \phi(\mathbf{x}), \phi(\mathbf{x}') \rangle = (x_1)^2 (x_1')^2 + (x_2)^2 (x_2')^2.$$

Training a SVM in the feature space

Replace each $\mathbf{x}^T\mathbf{x}'$ in the SVM algorithm by $\langle \phi(\mathbf{x}), \phi(\mathbf{x}') \rangle = k(\mathbf{x}, \mathbf{x}')$

- The dual problem is to maximize

$$g(\alpha) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j \boldsymbol{k(\mathbf{x_i}, \mathbf{x_j})},$$

under the constraints:

$$\begin{cases} 0 \leq \alpha_i \leq C, & \text{for } i = 1, \dots, n \\ \sum_{i=1}^{n} \alpha_i \mathbf{y}_i = 0. \end{cases}$$

- The **decision function** becomes:

$$\begin{aligned} f(\mathbf{x}) &= \langle \mathbf{w}, \phi(x) \rangle + b^* \\ &= \sum_{i=1}^{n} y_i \alpha_i \boldsymbol{k(\mathbf{x_i}, \mathbf{x})} + b^*. \end{aligned} \tag{1}$$

The kernel trick

- **The explicit computation of $\phi(\mathbf{x})$ is not necessary. The kernel $k(\mathbf{x}, \mathbf{x}')$ is enough.**

- The SVM optimization for α works **implicitly** in the feature space.

- The SVM is a kernel algorithm: only need to input K and $\mathbf{y}$:

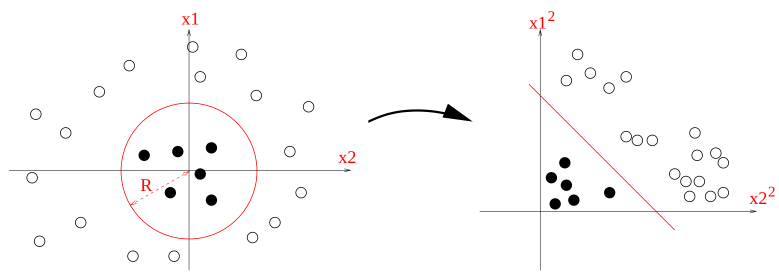
$$\begin{array}{ll} \text{maximize} & g(\alpha) = \alpha^T \mathbf{1} - \frac{1}{2}\alpha^T(\mathbf{y}^T \boldsymbol{K} \mathbf{y})\alpha \\ \text{such that} & 0 \leq \alpha_i \leq C, \quad \text{for } i = 1, \ldots, n \\ & \sum_{i=1}^n \alpha_i \mathbf{y}_i = 0. \end{array}$$

- in the end the solution $f(\mathbf{x}) = \sum_{i=1}^n y_i \alpha_i k(\mathbf{x}_i, \mathbf{x}) + b$.

Kernel example: polynomial kernel

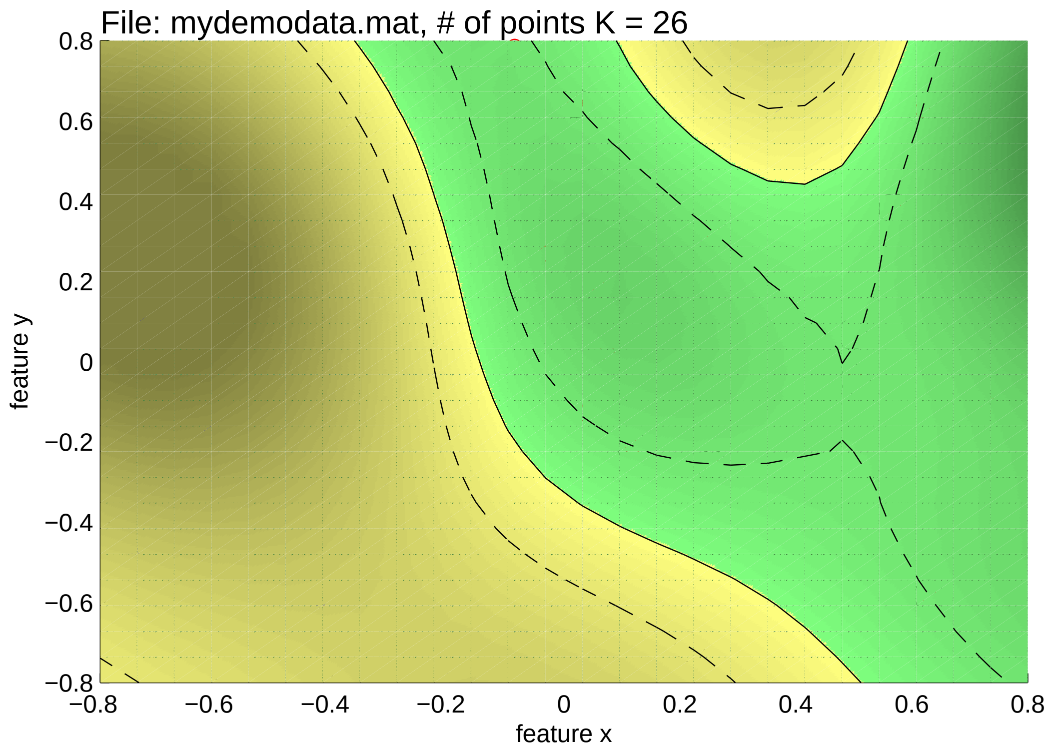
- For $\mathbf{x} = (x_1, x_2)^\top \in \mathbb{R}^2$, let $\phi(\mathbf{x}) = (x_1^2, \sqrt{2}x_1x_2, x_2^2) \in \mathbb{R}^3$:

$$\begin{aligned}\boldsymbol{K}(\mathbf{x}, \mathbf{x}') &= x_1^2 x_1'^2 + 2x_1x_2x_1'x_2' + x_2^2x_2'^2 \\ &= \{x_1x_1' + x_2x_2'\}^2 \\ &= \{\mathbf{x}^T\mathbf{x}'\}^2 \,.\end{aligned}$$



Some demonstrations using Matlab

- playing with a few kernels and a few points



SVM's: a particular case of a more general framework, penalized estimation

Empirical Risk Minimization

- Starting with $\{(\mathbf{x}_1, \mathbf{y}_1), \cdots, (\mathbf{x}_n, \mathbf{y}_n)\}$, n couples of $\mathcal{X} \times \mathcal{Y}$,
- A class of functions $\mathcal{F}$,
- A cost function $\boldsymbol{c} : \mathcal{Y} \times \mathcal{Y}$, $c \geq 0$, which penalizes discrepancies (hinge, least squares *etc.*)
- find a function which minimizes

$$\hat{f} = \underset{f \in \mathcal{F}}{\operatorname{argmin}} \frac{1}{n} \sum_{i=1}^{n} \boldsymbol{c}(f(\mathbf{x}_i), y_i)$$

  and use this f as a decision function.
- As usual in minimizations, we like:
  - Convex problems, unique minimizers
  - Stable solutions numerically.

Linear least squares

- When $\mathcal{X} = \mathbb{R}^d$, $\mathcal{Y} = \mathbb{R}$,
- $\boldsymbol{\mathcal{F} = \{\mathbf{x} \mapsto \beta^T \mathbf{x} + b\,, \beta \in \mathbb{R}^d, b \in \mathbb{R}\}}$, $\boldsymbol{c(\mathbf{y}_1, \mathbf{y}_2) = \|\mathbf{y}_1 - \mathbf{y}_2\|^2}$,
- The problem is known as **regression** with the **least squares criterion**.
- In this case, the minimizer

$$\operatorname*{argmin}_{f \in \mathcal{F}} \frac{1}{n} \sum_{i=1}^{n} \|f(\mathbf{x}_i) - \mathbf{y}_i\|^2 = \operatorname*{argmin}_{\beta \in \mathbb{R}^d, b \in \mathbb{R}} \frac{1}{n} \sum_{i=1}^{n} \|\beta^T \mathbf{x}_i + b - \mathbf{y}_i\|^2$$

  is **unique assuming** $n > d$ and no degeneracy.

- Why?

$$R : (b, \beta) \rightarrow \frac{1}{n} \sum_{i=1}^{n} \|\beta^T \mathbf{x}_i + b - \mathbf{y}_i\|^2 = \frac{1}{n} \|X^T \left[\begin{smallmatrix} b \\ \beta \end{smallmatrix}\right] - y\|^2$$

  is convex, where $X = \begin{bmatrix} 1 & 1 & \cdots & 1 \\ \vdots & \vdots & \cdots & \vdots \\ \mathbf{x}_1 & \mathbf{x}_2 & \cdots & \mathbf{x}_n \\ \vdots & \vdots & \cdots & \vdots \end{bmatrix} \in \mathbb{R}^{d+1 \times n}$ and $y = \begin{bmatrix} \mathbf{y}_1 \\ \vdots \\ \mathbf{y}_n \end{bmatrix} \in \mathbb{R}^n$.

Linear least squares

- Notice that

$$R(b,\beta) = \frac{1}{n}\left(\left[\begin{smallmatrix} b \\ \beta \end{smallmatrix}\right]^T XX^T \left[\begin{smallmatrix} b \\ \beta \end{smallmatrix}\right] - 2y^T X^T \left[\begin{smallmatrix} b \\ \beta \end{smallmatrix}\right] + \|y\|^2 \right)$$

- Let us take the gradient of that function

$$n\nabla R = 2XX^T \left[\begin{smallmatrix} b \\ \beta \end{smallmatrix}\right] - 2Xy$$

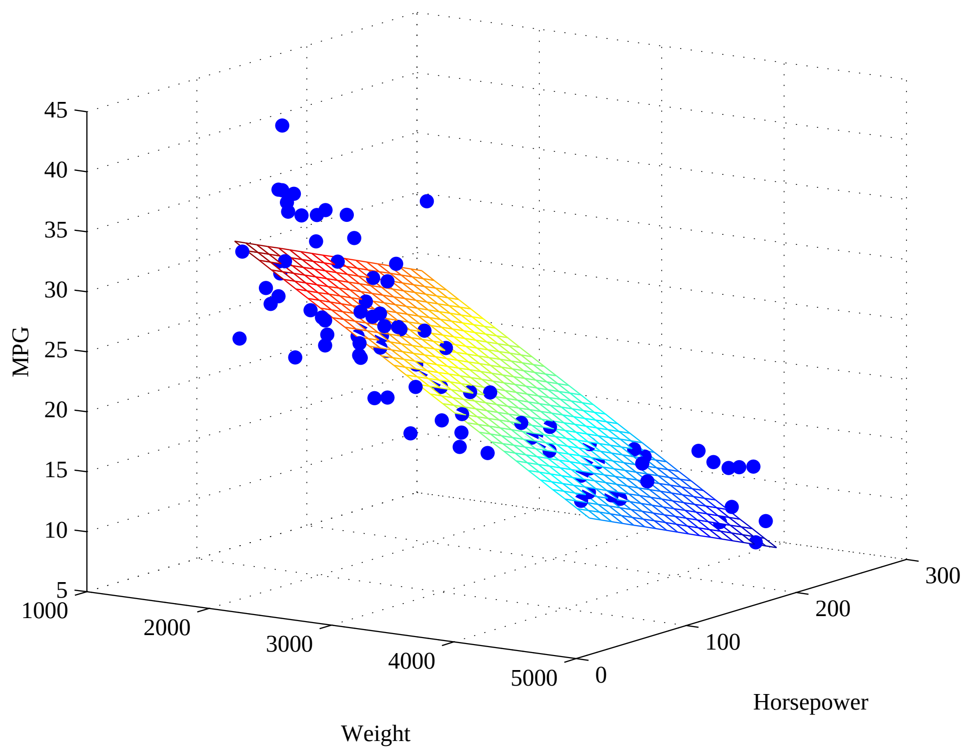
- Hence this gradient is zero for $\left[\begin{smallmatrix} b \\ \beta \end{smallmatrix}\right] = (XX^T)^{-1}Xy$
- $XX^T \in \mathbf{S}_+^n$. This works if $XX^T \in \mathbb{R}^{d+1}$ is **invertible**, that is $XX^T \in \mathbf{S}_{++}^n$.
- Remark:

$$XX^T = \begin{bmatrix} n & n\bar{x}_1 & n\bar{x}_2 & \cdots & n\bar{x}_d \\ n\bar{x}_1 & & & & \\ n\bar{x}_2 & & & & \\ \vdots & & \mathbf{X}\mathbf{X}^T & & \\ n\bar{x}_d & & & & \end{bmatrix} = \begin{bmatrix} n & n\mu^T \\ n\mu & \mathbf{X}\mathbf{X}^T \end{bmatrix}$$

where $\mathbf{X}$ is simply the $d \times n$ sample matrix without the constant 1.

Example in $\mathbb{R}^3$

- Sample of cars: x desribes weight and horsepower of a car.
- y is the miles-per-gallon : high is eco-friendly, low is bad.



- The hyperplane fits the data quite well, $\begin{bmatrix} b \\ \beta \end{bmatrix} \begin{bmatrix} 47.7694 \\ -0.0066 \\ -0.0420 \end{bmatrix} \begin{bmatrix} b \\ \beta \end{bmatrix}$.

Linear least-squares is not the ideal tool though...

- What happens when $d \gg n$? (XX^T) is **no longer invertible**...
  - high-dimensional data in genomics,
  - images analysis ($e.g.$lots of features)

- What happens when (XX^T) is **badly conditioned** ($\frac{\lambda_{\min}(XX^T)}{\lambda_{\max}(XX^T)} \approx 0$)?
  - if $\lambda_{\min}(XX^T) = 1e-10$, $\lambda_{\max}\left((XX^T)^{-1}\right) = 1e10!!$
  - Very bad numerical stability of the solution...

- When $d \gg n$, we might want to do **variable selection**,
  - $i.e.$pick a subset d' of the d variables which is relevant to predict $\mathbf{y}$.
  - $i.e.$favor vectors β such that $\|\beta\|_0 = \operatorname{card} \beta_i \neq 0$ is small.

Penalized Least-Squares

- For all these problems, there is an appropriate penalization:

$$(\hat{\beta}, \hat{b}) = \underset{\beta\in\mathbb{R}^d, b\in\mathbb{R}}{\operatorname{argmin}} \frac{1}{n}\sum_{i=1}^{n} \|\beta^T \mathbf{x}_i + b - \mathbf{y}_i\|^2 + \lambda \|(\beta, b)\|$$

  - we recover **least-square regression** when $\lambda = 0$;
  - **ridge regression** when $\lambda > 0$ and $\|(\beta, b)\| = \|(\beta, b)\|_2^2 = b^2 + \left(\sum_{i=1}^n \beta_i^2\right)^2$:

$$\left[\begin{smallmatrix} b \\ \beta \end{smallmatrix}\right] = \left(XX^T + \lambda \left[\begin{smallmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & 0 & 0 \\ \vdots & 0 & \ddots & 0 \\ 0 & 0 & 0 & 1 \end{smallmatrix}\right]\right)^{-1} Xy$$

  - the **lasso** when $\lambda > 0$ and $\|(\beta, b)\| = \|(\beta, b)\|_1 = |b| + \sum_{i=1}^n |\beta_i|$;

What about the case where linearity does not work?

- Many examples show that life is not always linear... **kernels at the rescue**.

- Let us take a further look at $\beta = (XX^T)^{-1}Xy$.
- For any new point, $\beta^T\mathbf{x}$ plays the same role as $\mathbf{w}^T\mathbf{x}$ in the SVM.
- We consider a new point $\mathbf{x} \in \mathbb{R}^d$ with the constant 1, *i.e.* $\mathbf{x} \leftarrow \left[\begin{smallmatrix} 1 \\ \mathbf{x} \end{smallmatrix}\right]$.
- $[b, \beta^T]\mathbf{x} = \mathbf{x}^T(XX^T + \lambda I_d)^{-1}Xy$.

Kernel ridge regression

- A simple inversion trick states that $(XX^T + \lambda I_d)^{-1} X y = X(\lambda I_n + X^T X)^{-1} y$

- Hence $[b, \beta^T]\mathbf{x} = \mathbf{x}^T X(\lambda I_n + X^T X)^{-1} y = \begin{bmatrix} \mathbf{x}^T\mathbf{x}_1 \\ \mathbf{x}^T\mathbf{x}_2 \\ \vdots \\ \mathbf{x}^T\mathbf{x}_d \end{bmatrix}^T (\lambda I_n + [\, \mathbf{x}_i^T \mathbf{x}_j \,])^{-1} y$!

- Bottom line: we have shown how to compute a regression tool which only depends on dot-products.

- Dot-products can be replaced by **kernels**!

$$f(\mathbf{x}) = \begin{bmatrix} k(\mathbf{x},\mathbf{x}_1) \\ k(\mathbf{x},\mathbf{x}_2) \\ \vdots \\ k(\mathbf{x},\mathbf{x}_d) \end{bmatrix}^T (\lambda I_n + \left[k(\mathbf{x}_i, \mathbf{x}_j)\right])^{-1} y$$

Kernel methods

- **Many** other standard **linear** algorithms,
  - Principal Component Analysis,
  - Canonical Correlation Analysis,
  - Fisher Discriminant analysis,
  - $etc.$can be modified to **incorporate** kernel similarities.

Algorithms based on **kernels** are known as **kernel methods**.

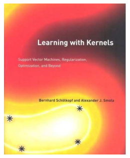
Kernel Methods

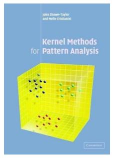
A reasonably large academic subfield

- Widespread popularity in machine learning now

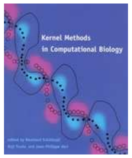
- Gained momentum in the late 90's with the support vector machine,
- Rooted in much older maths.
- Kernel methods are a pluridisciplinary field, publications appearing in
  - computer science (*nips, journ. of machine learning, ICML..*),
  - statistics and functional analysis (*annals of statistics..*),
  - optimization (*Mathematical Programming..*),
  - Different application subfields (*Neural Computation..*)

Kernel Methods

- Standard text-books:
  - Introduction [SS02] 

  - More about kernels [STC04] 

  - More learning theory [SC08] 

  - First chapters [STV04] 

  - “Mathematical” perspective [BTA03]. The real deal: [BCR84].
- Some short surveys,
  - journal papers [HHS08], [MMR+01]
  - a survey on my webpage (local copy, not arxiv): key to all citations!
- On the web:
  - Courses by J.-P. Vert, Francis Bach, Kenji Fukumizu, Stéphane Canu.

Some terminology

Etymology : from old english *cyrnel*, diminutive of corn (seed)

the word **kernel** appears in different different contexts...

- The *linux* kernel...
- Kernel of a linear operator of $\mathcal{X}$: $\ker(L) = \{x \in \mathcal{X} | L(x) = 0\}$.
- Kernel of a matrix in $\mathbb{R}^{d\times d}$, $i.e.$ its nullspace $\{\mathbf{x} \in \mathbb{R}^d | A\mathbf{x} = \mathbf{0}\}$.
- In set theory, for a function $f : \mathcal{X} \mapsto \mathcal{Y}$, $\ker(f) = \{(x, x') | \, f(x) = f(x')\}$.
- Kernel of an integral transform T, $Tf(u) = \int_{t_1}^{t_2} k(t,u) f(t) dt$
- Smoothing kernel, a function $k \geq 0, k(u) = k(-u), \int_{-\infty}^{\infty} k(u) du = 1$.
- $K(t,x,y) = \frac{1}{(4\pi t)^{d/2}} e^{-\frac{\|x-y\|^2}{4t}}$ solves heat equation $K(t,x,y) = \Delta_x K(t,x,y)$

sets, subspaces, **one-**variable, **two-**variables, **three-**variables function...

Moral of the story

No need to look for a common or primitive meaning

- Kernel is just a word mathematicians fancy (unfortunately!)
- People enjoy it because of its vague "core" meaning.
- Don't feel you have missed something if you do not see the connection between different *kernel* objects in mathematics. There might be none...
- Will mention some links during the lecture between different definitions.

What is a kernel

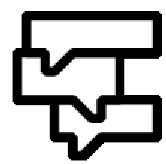
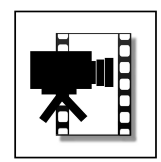
In the context of this lecture...

- A kernel k is a function

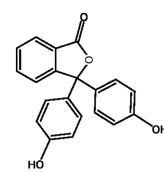
$$
\begin{array}{rccc}
k: & \mathcal{X} \times \mathcal{X} & \longmapsto & \mathbb{R} \\
 & (\mathbf{x}, \mathbf{y}) & \longrightarrow & k(\mathbf{x}, \mathbf{y})
\end{array}
$$

- which compares two objects of a space $\mathcal{X}$, *e.g.*...
  - strings, texts and sequences,

  - images, audio and video feeds,

  - graphs, interaction networks and 3D structures



- whatever actually... time-series of graphs of images? graphs of texts?...

Fundamental properties of a kernel

symmetric

$$k(\mathbf{x},\mathbf{y}) = k(\mathbf{y},\mathbf{x}).$$

positive-(semi)definite

for any *finite* family of points $\mathbf{x}_1, \cdots, \mathbf{x}_n$ of $\mathcal{X}$, the matrix

$$K = \begin{bmatrix} k(\mathbf{x}_1,\mathbf{x}_1) & k(\mathbf{x}_1,\mathbf{x}_2) & \cdots & k(\mathbf{x}_1,\mathbf{x}_i) & \cdots & k(\mathbf{x}_1,\mathbf{x}_n) \\ k(\mathbf{x}_2,\mathbf{x}_1) & k(\mathbf{x}_2,\mathbf{x}_2) & \cdots & k(\mathbf{x}_2,\mathbf{x}_i) & \cdots & k(\mathbf{x}_2,\mathbf{x}_n) \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ k(\mathbf{x}_i,\mathbf{x}_1) & k(\mathbf{x}_i,\mathbf{x}_2) & \cdots & k(\mathbf{x}_i,\mathbf{x}_i) & \cdots & k(\mathbf{x}_2,\mathbf{x}_n) \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ k(\mathbf{x}_n,\mathbf{x}_1) & k(\mathbf{x}_n,\mathbf{x}_2) & \cdots & k(\mathbf{x}_n,\mathbf{x}_i) & \cdots & k(\mathbf{x}_n,\mathbf{x}_n) \end{bmatrix} \succeq 0$$

is positive semidefinite (has a nonnegative spectrum).

$\boxed{K \text{ is often called the } \textbf{Gram matrix} \text{ of } \{\mathbf{x}_1, \cdots, \mathbf{x}_n\} \text{ using } k}$

What can we do with a kernel?

The setting

- Pretty simple setting: a set of objects $\mathbf{x}_1, \cdots, \mathbf{x}_n$ of $\mathcal{X}$
- **Sometimes** additional information on these objects
  - labels $\mathbf{y}_i \in \{-1, 1\}$ or $\{1, \cdots, \#(\text{classes})\}$,
  - scalar values $\mathbf{y}_i \in \mathbb{R}$,
  - associated object $\mathbf{y}_i \in \mathcal{Y}$
- A kernel $k : \mathcal{X} \times \mathcal{X} \mapsto \mathbb{R}$.

A few intuitions on the possibilities of kernel methods

$\boxed{\text{Important concepts and perspectives}}$

- The functional perspective: represent **points as functions**.
- The new or **alternative dot-product** perspective.
- **Nonlinearity** : linear combination of kernel evaluations.
- Summary of a sample through its **kernel matrix**.

Represent any point in $\mathcal{X}$ as a function

For every $\mathbf{x}$, the map

$$\mathbf{x} \longrightarrow k(\mathbf{x}, \cdot)$$

associates to $\mathbf{x}$ a function $k(\mathbf{x}, \cdot)$ from $\mathcal{X}$ to $\mathbb{R}$.

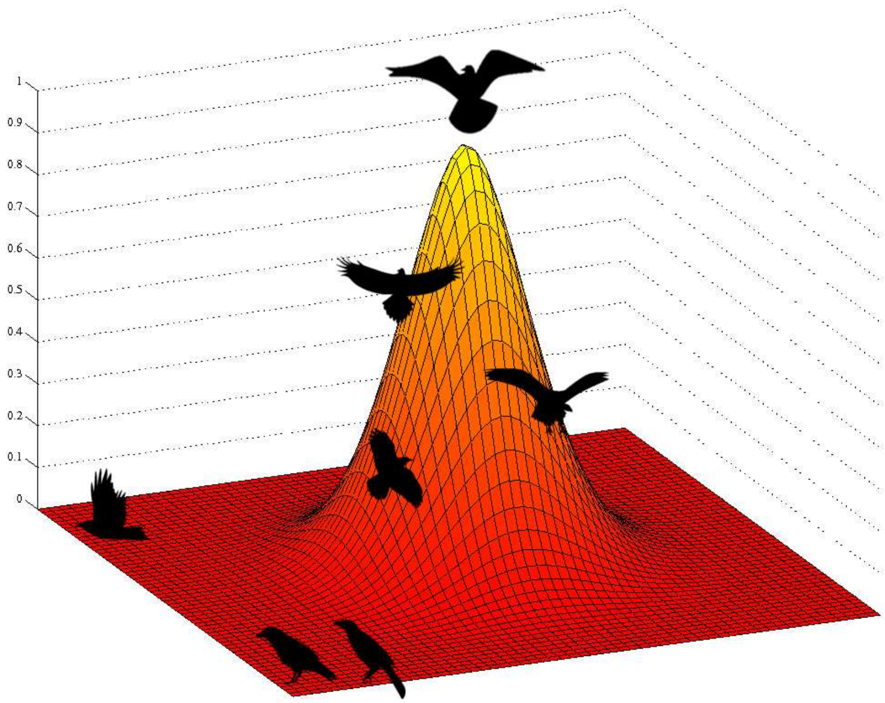
- Suppose we have a kernel k on bird images

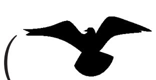
- Suppose for instance

$k\,($  $,$  $) = .32$

Represent any point in $\mathcal{X}$ as a function

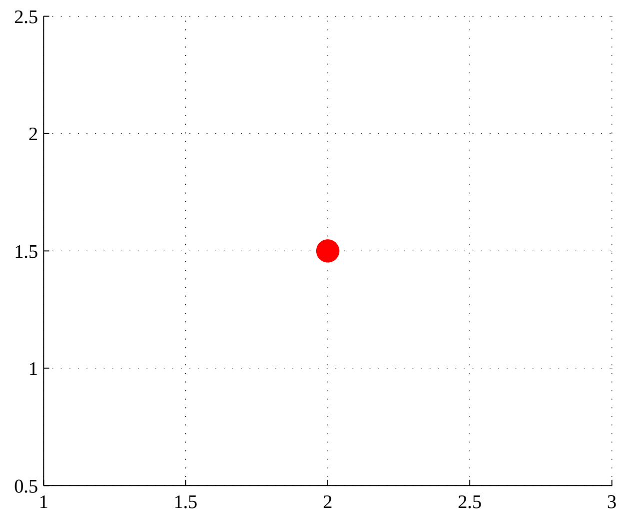
- We examine one image in particular: 
- With kernels, we get a **representation** of that bird as a real-valued function, defined on the space of birds, represented here as $\mathbb{R}^2$ for simplicity.



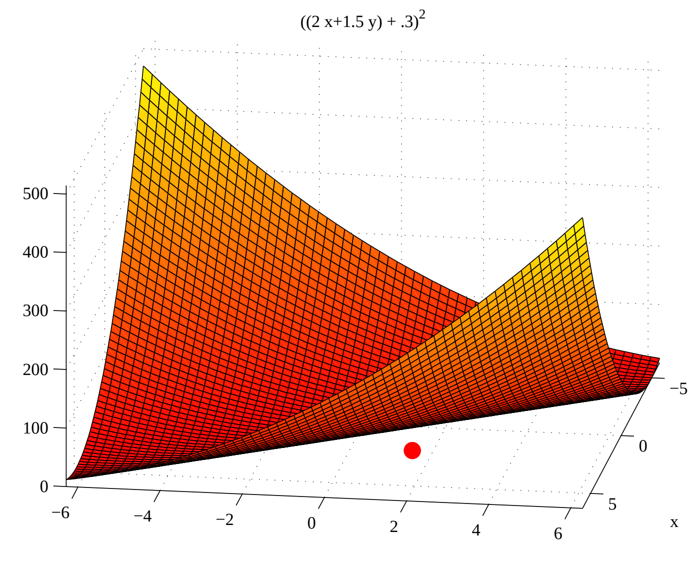
schematic plot of $k\,($  $, \cdot\ )$.

Represent any point in $\mathcal{X}$ as a function

- If the bird example was confusing...
- $k\left(\begin{bmatrix} x \\ y \end{bmatrix}, \begin{bmatrix} x' \\ y' \end{bmatrix}\right) = \left(\begin{bmatrix} x & y \end{bmatrix}\begin{bmatrix} x' \\ y' \end{bmatrix} + .3\right)^2$
- From a point in $\mathbb{R}^2$ to a function defined over $\mathbb{R}^2$.



$\longrightarrow$



- We assume implicitly that the **functional representation** will be more useful than the **original representation**.

Dot-product perspective

- Suppose $\mathcal{X} = \mathbb{R}^d$.
- The simplest kernel: $k(\mathbf{x}, \mathbf{y}) = \mathbf{x}^T \mathbf{y}$.
- For a data sample $X = \{\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_n\}$.
- In matrix form, $X = \begin{bmatrix} \vdots & \vdots & \cdots & \vdots \\ \mathbf{x}_1 & \mathbf{x}_2 & \cdots & \mathbf{x}_n \\ \vdots & \vdots & \cdots & \vdots \end{bmatrix} \in \mathbb{R}^{d \times n}$.
- In standard linear algebra, the Gram matrix of X is

$$K = \left[\mathbf{x}_i^T \mathbf{x}_j\right]_{1 \le i,j \le n} = X^T X.$$

Dot-product perspective

- Consider a different kernel $k_G(\mathbf{x}, \mathbf{y}) = \exp\left(-\frac{\|\mathbf{x}-\mathbf{y}\|^2}{\sigma^2}\right)$,

$$K_G = \left[k_G(\mathbf{x}_i, \mathbf{x}_j)\right]_{1 \leq i,j \leq n}.$$

- obviously $\mathbf{x}_i^T \mathbf{x}_j \neq k_G(\mathbf{x}_i, \mathbf{x}_j)$.
- is there a representation $\xi_i \in \mathbb{R}^{??}$ for each point such that $\xi_i^T \xi_j = k_G(\mathbf{x}_i, \mathbf{x}_j)$?
- Linear algebra to the rescue: $K = PDP^T$, $U = P\sqrt{D}P^T$, hence $K = U^T U$, providing $U = \begin{bmatrix} \vdots & \vdots & \cdots & \vdots \\ \xi_1 & \xi_2 & \cdots & \xi_n \\ \vdots & \vdots & \cdots & \vdots \end{bmatrix} \in \mathbb{R}^{\boldsymbol{n} \times n}$.

Dot-product perspective

- In summary, we have defined n vectors such that

$$\left[k_G(\mathbf{x}_i, \mathbf{x}_j)\right] = \left[\xi_i^T \xi_j\right]$$

- Great: for each $\mathbf{x}_i$ we have a vector representation ξ_i.
- Problem:
  - this representation depends explicitly on the sample X.
  - For a new $\mathbf{x}_{n+1}$, difficult to find ξ_{n+1} such that $\xi_{n+1}^T \xi_j = k_G(\mathbf{x}_{n+1}, \mathbf{x}_j)$.
- **We will see that there exists a mapping ϕ**, such that
  - $\phi : \mathcal{X} \rightarrow \mathcal{H}$ where $\mathcal{H}$ is a dot-product space,
  - which gives a dot product representation for k,

$$k_G(\mathbf{x}, \mathbf{y}) = \langle \phi(\mathbf{x}), \phi(\mathbf{y}) \rangle.$$

for **all points** $(\mathbf{x}, \mathbf{y})$...

Decision functions as linear combination of kernel evaluations

- Linear decisions functions are a major tool in statistics, that is functions

$$f(\mathbf{x}) = \beta^T \mathbf{x} + \beta_0.$$

- Implicitly, a point $\mathbf{x}$ is processed depending on its characteristics x_i,

$$f(\mathbf{x}) = \sum_{i=1}^{d} \boldsymbol{\beta_i} x_i + \boldsymbol{\beta_0}.$$

  the free parameters are scalars $\boldsymbol{\beta_0, \beta_1, \cdots, \beta_d}$.

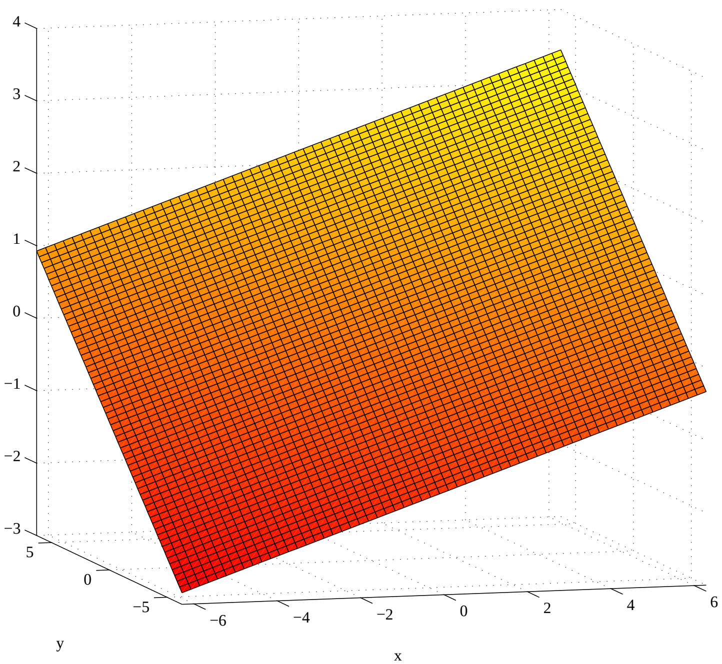
- Kernel methods yield candidate decision functions

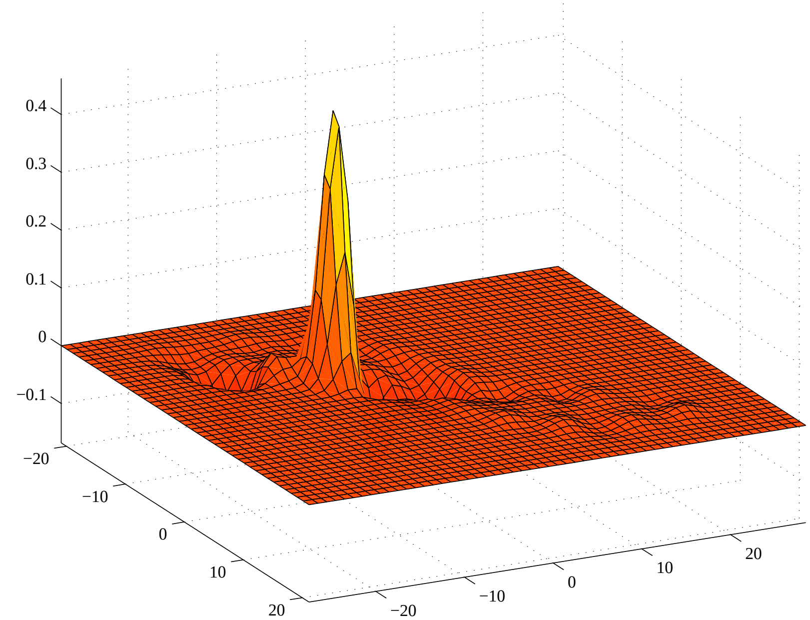
$$f(\mathbf{x}) = \sum_{j=1}^{\boldsymbol{n}} \boldsymbol{\alpha_j} k(\mathbf{x}_j, \mathbf{x}) + \boldsymbol{\alpha_0}.$$

  the free parameters are scalars $\boldsymbol{\alpha_0, \alpha_1, \cdots, \alpha_n}$.

Decision functions as linear combination of kernel evaluations

- linear decision surface / linear expansion of **kernel surfaces** (here $k_G(\mathbf{x}_i, \cdot)$)





- Kernel methods are considered **non-linear** tools.
- Yet not completely "nonlinear" $\longrightarrow$ only one-layer of nonlinearity.

$$\boxed{\text{kernel methods use the data as a functional base to define decision functions}}$$

Decision functions as linear combination of kernel evaluations

database $\{\mathbf{x}_i, i = 1, \ldots, N\}$

$$f(\mathbf{x}) = \sum_{i=1}^{N} \alpha_i \; k\,(\mathbf{x}_i, \mathbf{x})$$

kernel definition

weights α estimated
with a kernel machine

- f is any predictive function of interest of a new point $\mathbf{x}$.
- Weights α are **optimized** with a kernel machine ($e.g.$ support vector machine)

intuitively, kernel methods provide decisions based on how *similar* a point $\mathbf{x}$ is to each instance of the training set

The Gram matrix perspective

- Imagine a little task: you have read 100 novels so far.

- You would like to know whether you will enjoy reading a **new** novel.
- A few options:
  - read the book...
  - have friends read it for you, read reviews.
  - try to guess, based on the novels you read, if you will like it

The Gram matrix perspective

$\boxed{\text{Two distinct approaches}}$

- Define what **features** can characterize a book.
  - Map each book in the library onto vectors

$$\longrightarrow \qquad \mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_d \end{bmatrix}$$

typically the x_i's can describe...

▷ # pages, language, year 1st published, country,
▷ coordinates of the main action, keyword counts,
▷ author's prizes, popularity, booksellers ranking

- Challenge: find a decision function using 100 ratings and features.

The Gram matrix perspective

- Define what makes **two novels similar**,
  - Define a kernel k which quantifies novel similarities.
  - Map the library onto a Gram matrix

$$\longrightarrow \quad K = \begin{bmatrix} k(b_1,b_1) & k(b_1,b_2) & \cdots & k(b_1,b_{100}) \\ k(b_2,b_1) & k(b_2,b_2) & \cdots & k(b_2,b_{100}) \\ \vdots & \vdots & \ddots & \vdots \\ k(b_n,b_1) & k(b_n,b_2) & \cdots & k(b_{100},b_{100}) \end{bmatrix}$$

- Challenge: find a decision function that takes this 100×100 matrix as an input.

The Gram matrix perspective

Given a new novel,

- with the **features approach**, the prediction can be rephrased as **what are the features of this new book**? what **features** have I found in the past that were good indicators of my taste?

- with the **kernel approach**, the prediction is rephrased as **which novels this book is similar or dissimilar to?** what **pool of books** did I find the most influentials to define my tastes accurately?

kernel methods **only use kernel similarities**, do not consider features.

Features can help define similarities, but **never considered elsewhere**.

The Gram matrix perspective

In summary

- A feature based analysis of a data-driven problem:

$$\text{objects } o_1, \cdots, o_n \quad \longrightarrow \quad \text{feature vectors } X = \begin{bmatrix} \vdots & \vdots & \cdots & \vdots \\ \mathbf{x}_1 & \mathbf{x}_2 & \cdots & \mathbf{x}_n \\ \vdots & \vdots & \cdots & \vdots \end{bmatrix} \in \mathbb{R}^{\boldsymbol{d} \times n}$$

- A similarity based analysis of a data driven problem:

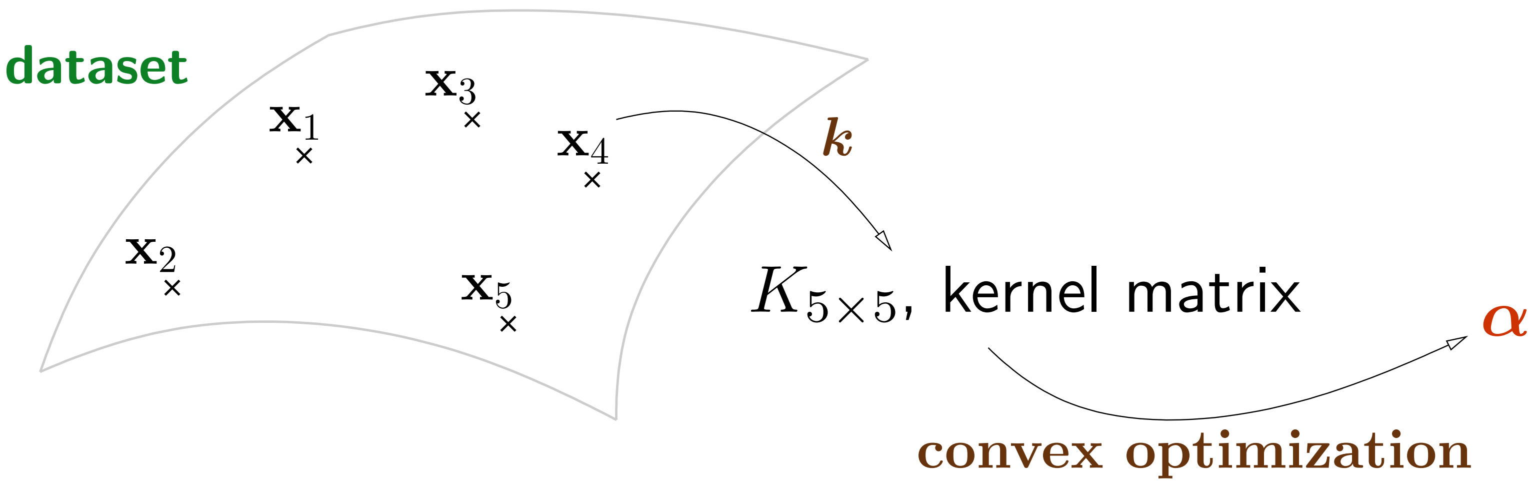
$$\text{objects } o_1, \cdots, o_n \quad \rightarrow \quad \text{Gram } K = \begin{bmatrix} k(o_1, o_1) & k(o_1, o_2) & \cdots & k(o_1, o_n) \\ k(o_2, o_1) & k(o_2, o_2) & \cdots & k(o_2, o_n) \\ \vdots & \vdots & \ddots & \vdots \\ k(o_n, o_1) & k(o_n, o_2) & \cdots & k(o_n, o_n) \end{bmatrix} \in \mathbb{R}^{\boldsymbol{n} \times n}$$

- Some parallels (can define $K = X^T X$ or $X = \sqrt{K}$ or Cholesky) but...

Algorithms use either features *or* (kernel) similarities.

The Gram matrix perspective

in kernel methods, clear separation between the kernel...



and **Convex optimization** (thanks to psdness of K, more later) to output the α's.

Mathematical Considerations

different definitions and properties of the same mathematical object

An intuitive perspective: Feature maps

Theorem 1. *A function k on $\mathcal{X} \times \mathcal{X}$ is a positive definite kernel if and only if there exists a set T and a mapping ϕ from $\mathcal{X}$ to $l^2(T)$, the set of real sequences $\{u_t, t \in T\}$ such that $\sum_{t\in T} |u_t|^2 < \infty$, where*

$$\forall (\mathbf{x}, \mathbf{y}) \in \mathcal{X} \times \mathcal{X}, \; k(\mathbf{x}, \mathbf{y}) = \sum_{t \in T} \phi(\mathbf{x})_t \, \phi(\mathbf{y})_t = \langle \phi(\mathbf{x}), \phi(\mathbf{y}) \rangle_{l^2(X)}$$

- A very popular perspective in the machine learning world.
- Equivalent to previous definitions, less stressed in the RHKS literature.

$$\mathbf{x} \longrightarrow \phi(\mathbf{x}) = \begin{bmatrix} \vdots \\ \vdots \\ \phi(\mathbf{x})_t \\ \vdots \\ \vdots \end{bmatrix}_{t \in T}$$

where the ϕ_t are a set of – possibly infinite but countable – features.

kernels $\rightarrow$ Gram matrices

- If $X = \{\mathbf{x}_i\}_{i \in I}$ in $\mathcal{X}$,
$$K_X = [k(\mathbf{x}_i, \mathbf{x}_j)]_{i,j \in I} \succeq 0.$$
- If one applies *any* transformation of K_X which keeps eigenvalues nonnegative,
$$\begin{array}{cccc} r: & \mathbf{S}_n & \longmapsto & \mathbf{S}_n \\ & K & \longrightarrow & r(K), \end{array}$$

  $r(K)$ is a valid positive definite matrix and hence a kernel on X.
- examples: $K + t(t > 0), K^2, e^K, etc.$
- in fact, if $K = P\Delta P^T$, any transformation that preserves the spectrum's non-negativity would be ok.
- Yet... this kernel is only valid on X, the sample, not the whole space $\mathcal{X}$.

$$\boxed{\text{Meaning somehow... Gram matrices} \not\rightarrow \text{kernels}}$$

positive definite kernels and distances

- Kernels are often called similarities.
- the **higher** $k(\mathbf{x},\mathbf{y})$, the more similar $\mathbf{x}$ and $\mathbf{y}$.
- With distances, the **lower** $d(\mathbf{x},\mathbf{y})$, the closer $\mathbf{x}$ and $\mathbf{y}$.

- Many distances exist in the literature. Can they be used to define kernels?

what is the link between kernels and distances?
high similarity $\stackrel{?}{=}$ **small distance**

- At least true for the Gaussian kernel $k(\mathbf{x},\mathbf{y}) = e^{-\|\mathbf{x}-\mathbf{y}\|^2/2\sigma^2}$...
- Important theorems taken from [BCR84].

Distances

Definition 1 (Distances, or metrics). *A* ***nonnegative-valued*** *function d on $\mathcal{X} \times \mathcal{X}$ is a distance if it satisfies, $\forall \mathbf{x}, \mathbf{y}, \mathbf{z} \in \mathcal{X}$:*

(i) $d(\mathbf{x}, \mathbf{y}) \geq 0$, *and* $d(\mathbf{x}, \mathbf{y}) = 0$ *if and only if* $\mathbf{x} = \mathbf{y}$ *(non-degeneracy)*

(ii) $d(\mathbf{x}, \mathbf{y}) = d(\mathbf{y}, \mathbf{x})$ *(symmetry),*

(iii) $d(\mathbf{x}, \mathbf{z}) \leq d(\mathbf{x}, \mathbf{y}) + d(\mathbf{y}, \mathbf{z})$ *(triangle inequality)*

- Very simple example: if $\mathcal{X}$ is a Hilbert space, $\|\mathbf{x} - \mathbf{y}\|$ is a distance. It is usually called a... Hilbertian distance.
- By extension, any distance $d(\mathbf{x}, \mathbf{y})$ which can be written as $\|\phi(\mathbf{x}) - \phi(\mathbf{y})\|$ where ϕ maps $\mathcal{X}$ to any Hilbert space is called a **Hilbertian metric**.
- Useful. To build Gaussian kernel, Laplace kernels $k(\mathbf{x}, \mathbf{y}) = e^{-t\|\mathbf{x}-\mathbf{y}\|}$...
- Yet does not suffice:

the missing link: negative definite kernels

Definition 2 (Negative Definite Kernels)**.** *A symmetric function* $\psi : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ *is a negative definite (n.d.) kernel on* $\mathcal{X}$ *if*

$$\sum_{i,j=1}^{n} c_i c_j \, \psi\left(x_i, x_j\right) \leq 0 \tag{1}$$

holds for any $n \in \mathbb{N}, x_1, \ldots, x_n \in \mathcal{X}$ *and* $c_1 \ldots, c_n \in \mathbb{R}$ *such that* $\sum_{i=1}^{n} c_i = 0$.

- Example $\psi(\mathbf{x}, \mathbf{y}) = \|\mathbf{x} - \mathbf{y}\|^2$.
    - prove by decomposing into $\|\mathbf{x}_i\|^2 + \|\mathbf{x}_j\|^2 - 2\langle \mathbf{x}_i, \mathbf{x}_j \rangle$
- $\mathcal{N}(\mathcal{X})$ is also a closed convex cone.

important example: k is p.d. $\Rightarrow -k$ is n.d.
Converse completely false.

negative definite kernels & positive definite kernels

A first link between these two kernels:

Proposition 2. *Let* $x_0 \in \mathcal{X}$ *and let* $\psi : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ *be a symmetric kernel. Let*

$$\varphi(\mathbf{x}, \mathbf{y}) \stackrel{\text{def}}{=} \psi(\mathbf{x}, x_0) + \psi(\mathbf{y}, x_0) - \psi(\mathbf{x}, \mathbf{y}) - \psi(x_0, x_0).$$

Then k *is positive definite* $\Leftrightarrow$ ψ *is negative definite.*

- Example: $\|\mathbf{x} - x_0\|^2 + \|\mathbf{y} - x_0\|^2 - \|\mathbf{x} - \mathbf{y}\|^2$ is a p.d. kernel.

Proof.

- $\Rightarrow$ For $\mathbf{x}_1, \cdots, \mathbf{x}_n$, and $c_1, \cdots, c_n$ s.t. $\sum_{i=1}^n c_i = \mathbf{0}$,

$$\sum_{i,j=1}^{n} c_i c_j \varphi(\mathbf{x}_i, \mathbf{x}_j) = -\sum_{i,j=1}^{n} c_i c_j \psi(\mathbf{x}_i, \mathbf{x}_j) \geq 0.$$

- $\Leftarrow$ For $\mathbf{x}_1, \cdots, \mathbf{x}_n$ and $c_1, \cdots, c_n$, let $c_0 = -\sum_{i=1}^n$. Set $\mathbf{x}_0 = x_0$. Then

$$\begin{aligned}
0 &\geq \sum_{i,j=0}^{n} c_i c_j \psi(\mathbf{x}_i, \mathbf{x}_j) \\
&= \sum_{i,j=1}^{n} c_i c_j \psi(\mathbf{x}_i, \mathbf{x}_j) + \sum_{i=1}^{n} c_i c_0 \psi(\mathbf{x}_i, x_0) + \sum_{j=1}^{n} c_0 c_j \psi(x_0, \mathbf{x}_j) + c_0^2 \psi(x_0, x_0). \\
&= \sum_{i,j=1}^{n} \left[\psi(\mathbf{x}_i, x_0) + \psi(\mathbf{x}_j, x_0) - \psi(\mathbf{x}_i, \mathbf{y}_j) - \psi(x_0, x_0)\right] = \sum_{i,j=1}^{n} c_i c_j \varphi(\mathbf{x}_i, \mathbf{x}_j).
\end{aligned}$$

negative definite kernels & positive definite kernels

Proposition 3. *For a p.d. kernel $k \geq 0$ on $\mathcal{X} \times \mathcal{X}$, the following conditions are equivalent*

(i) $-\log k \in \mathcal{N}(\mathcal{X})$,

(ii) k^t *is positive definite for all* $t > 0$.

If k satisfies either, k is said to be ***infinitely divisible***,

Proof.

- $-\log k = \lim_{n\to\infty} n(1 - k^{\frac{1}{n}})$ which is the limit of a series of n.d. kernels if (ii) is true, hence $(ii) \Rightarrow (i)$.
- conversely, if $-\log k \in \mathcal{N}(\mathcal{X})$ we use Proposition 2. Writing $\psi = -\log k$ and choosing $x_0 \in \mathcal{X}$ we have

$$k^t = e^{-t\psi(\mathbf{x},\mathbf{y})} = e^{t\psi(x_0,x_0)} \boldsymbol{e^{t\varphi(\mathbf{x},\mathbf{y})}} \boldsymbol{e^{-t\psi(\mathbf{x},x_0)}} \boldsymbol{e^{-t\psi(\mathbf{y},x_0)}} \in \mathcal{P}(\mathcal{X})$$

negative definite kernels: $(\text{Hilbertian distance})^2$ + ...

Proposition 4. *Let $\psi : \mathcal{X} \times \mathcal{X}$ be a n.d. kernel. Then there is a Hilbert space H and a mapping ϕ from X to H such that*

$$\psi(\mathbf{x}, \mathbf{y}) = \|\phi(\mathbf{x}) - \phi(\mathbf{y})\|^2 + f(\mathbf{x}) + f(\mathbf{y}), \tag{2}$$

where $f : \mathcal{X} \to \mathbb{R}$. If $\psi(x, x) = 0$ for all $\mathbf{x} \in \mathcal{X}$ then f can be chosen as zero. If the set $\{(\mathbf{x}, \mathbf{y}) |\, \psi(\mathbf{x}, \mathbf{y}) = 0\}$ is exactly $\{(\mathbf{x}, \mathbf{x}), \mathbf{x} \in \mathcal{X}\}$ then $\sqrt{\psi}$ is a Hilbertian distance.

Proof. Fix x_0 and define

$$\varphi(\mathbf{x}, \mathbf{y}) \stackrel{\text{def}}{=} \frac{\mathbf{1}}{\mathbf{2}} \left[\psi(\mathbf{x}, x_0) + \psi(\mathbf{y}, x_0) - \psi(\mathbf{x}, \mathbf{y}) - \psi(x_0, x_0)\right].$$

By Proposition 2 φ is p.d. hence there is a RKHS and mapping ϕ such that $\varphi(\mathbf{x}, \mathbf{y}) = \langle \phi(\mathbf{x}), \phi(\mathbf{y}) \rangle$. Hence

$$\begin{aligned} \|\phi(\mathbf{x}) - \phi(\mathbf{y})\|^2 &= \varphi(\mathbf{x}, \mathbf{x}) + \varphi(\mathbf{y}, \mathbf{y}) - 2\varphi(\mathbf{x}, \mathbf{y}) \\ &= \psi(\mathbf{x}, \mathbf{y}) - \frac{\psi(\mathbf{x}, \mathbf{x}) + \psi(\mathbf{y}, \mathbf{y})}{2}. \end{aligned}$$

distances & negative definite kernels

- whenever a n.d. kernel ψ
  - vanishes on the *diagonal*, $i.e.$ on $\{(x,x), x \in \mathcal{X}\}$,
  - is 0 only on the diagonal, to ensure non-degeneracy,

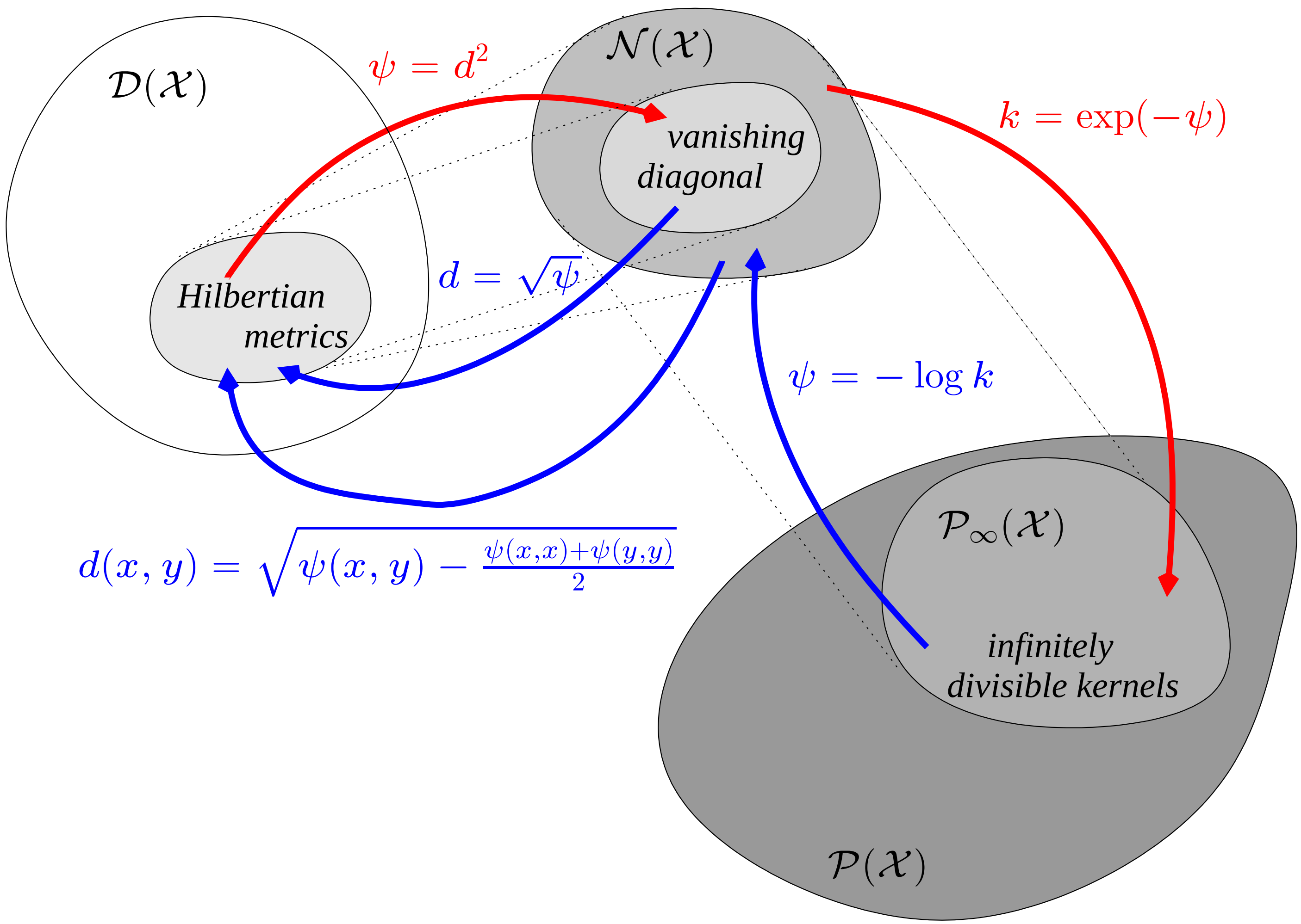
$\rightarrow \sqrt{\psi}$ is a Hilbertian distance for $\mathcal{X}$.

- **More generally**, for a n.d. kernel ψ,

$$\sqrt{\psi(\mathbf{x},\mathbf{y}) - \frac{\psi(\mathbf{x},\mathbf{x})}{2} - \frac{\psi(\mathbf{y},\mathbf{y})}{2}}$$ is a (pseudo)**metric** for $\mathcal{X}$.

- On the contrary, to each distance does not always correspond a n.d. kernel (Monge-Kantorovich distance, edit-distance $etc.$.)

In summary...



- Set of distances on $\mathcal{X}$ is $\mathcal{D}(\mathcal{X})$, Negative definite kernels $\mathcal{N}(\mathcal{X})$, positive and infinitely divisible positive kernels $\mathcal{P}(\mathcal{X})$ and $\mathcal{P}_\infty(\mathcal{X})$ respectively.

Some final remarks on $\mathcal{N}(\mathcal{X})$ and $\mathcal{P}(\mathcal{X})$

- $\mathcal{N}(\mathcal{X})$ is a cone. Additionally,
  - if $\psi \in \mathcal{N}(\mathcal{X}), \forall c \in \mathbb{R}$, $\psi + c \in \mathcal{N}(\mathcal{X})$.
  - if $\psi(x,x) \geq 0$ for all $x \in \mathcal{X}$, $\psi^\alpha \in \mathcal{N}(\mathcal{X})$ for $0 < \alpha < 1$ since

$$\psi^\alpha = \frac{\alpha}{\Gamma(1-\alpha)} \int_0^\infty t^{-\alpha-1}(1 - e^{-t\psi})dt$$

    and $\log(1+\psi) \in \mathcal{N}(\mathcal{X})$ since

$$\log(1+\psi) = \int_0^\infty (1 - e^{-t\psi}) \frac{e^{-t}}{t} dt.$$

  - if $\psi > 0$, then $\log(\psi) \in \mathcal{N}(\mathcal{X})$ since

$$\log(\psi) = \lim_{c \to \infty} \log\left(\psi + \frac{1}{c}\right) = \lim_{c \to \infty} \log\left(1 + c\psi\right) - \log c$$

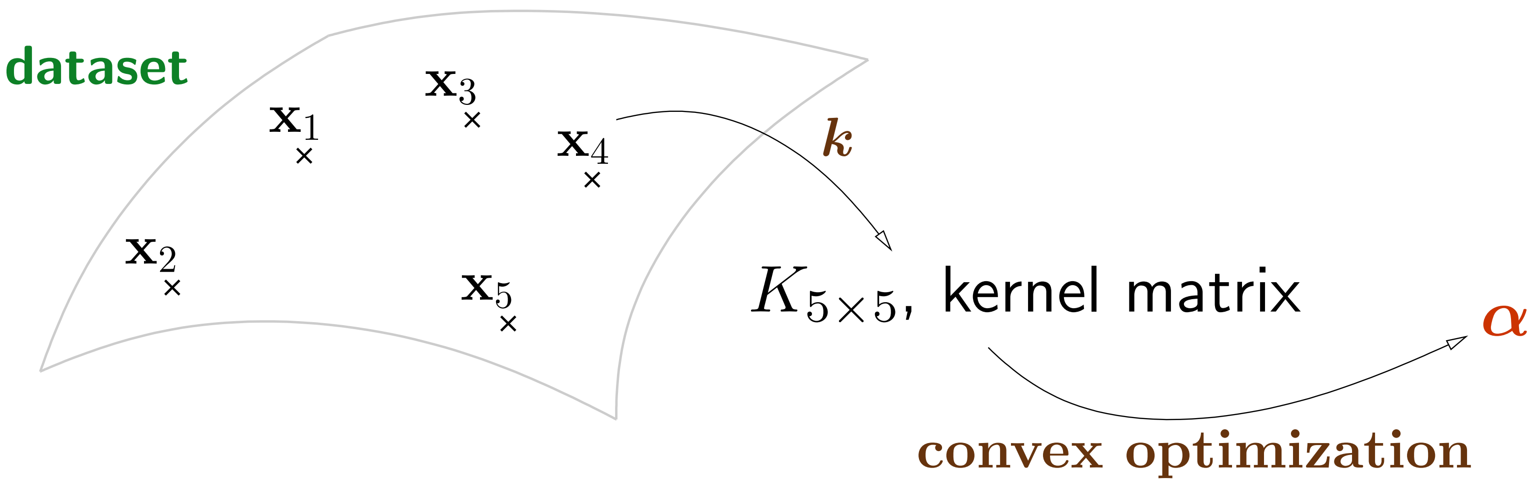
Some final remarks on $\mathcal{D}(\mathcal{X}), \mathcal{N}(\mathcal{X}), \mathcal{P}(\mathcal{X})$

- $\mathcal{P}(\mathcal{X})$ is a cone. Additionally,
  - The pointwise product $k_1 k_2$ of two p.d. kernels if a p.d. kernel
  - $k^n \in \mathcal{P}(\mathcal{X})$ for $n \in \mathbb{N}$. $(k+c)^n$ too...as well as $\exp(k) \in \mathcal{P}(\mathcal{X})$:
    - $\exp(k) = \sum_{i=0}^{\infty} \frac{k^i}{i!}$, a limit of p.d. kernels.
    - $\exp(k) = \exp(-(-k))$ where $-k \in \mathcal{N}(\mathcal{X})$.
- The sum of two infinitely divisible kernels is not necessarily infinitely divisible.
  - $-\log k_1$ and $-\log k_2$ might be in $\mathcal{N}(\mathcal{X})$, but $-\log(k_1 + k_2)$?...

Defining kernels

Intuitively an important issue...

Remember that kernel methods drop all previous information



to proceed exclusively with K.

if the kernel K is poorly informative, the optimization cannot be very useful...
it is therefore **crucial** that the kernel quantifies **noteworthy similarities**.

Kernels on vectors

(relatively) easy case: **we are only given feature vectors**, with **no access** to the original data.

- Reminder (copy paste of previous slide!): for a family of kernels $k_1, \cdots, k_n, \cdots$
  - The sum $\sum_{i=1}^{n} \lambda_i k_i$ is p.d., given $\lambda_1, \ldots, \lambda_n \geq 0$
  - The product $k_1^{a_1} \cdots k_n^{a_n}$ is p.d., given $a_1, \ldots, a_n \in \mathbb{N}$
  - $\lim_{n \to \infty} k_n$ is p.d. (if the limit exists!).
- Using these properties we can prove the p.d. of
  - the polynomial kernel $k_p(x, y) = (\langle \mathbf{x}, \mathbf{y} \rangle + b)^d, \quad b > 0, d \in \mathbb{N}$,
  - the Gaussian kernel $k_\sigma(x, y) = e^{-\frac{\|\mathbf{x}-\mathbf{y}\|^2}{2\sigma^2}}$ which can be rewritten as

$$k_\sigma(x, y) = \left[e^{-\frac{\|\mathbf{x}\|^2}{2\sigma^2}} e^{-\frac{\|\mathbf{y}\|^2}{2\sigma^2}} \right] \cdot \left[\sum_{i=0}^{\infty} \frac{\langle \mathbf{x}, \mathbf{y} \rangle^i}{i!} \right]$$

Kernels on vectors

- the Laplace kernels, using some n.d. kernel weaponry,

$$k_\lambda(x,y) = e^{-\lambda\|\mathbf{x}-\mathbf{y}\|^{\boldsymbol{a}}}, \quad 0 < \lambda,\, 0 < \boldsymbol{a} \leq 2$$

- the all-subset Gaussian kernel in $\mathbb{R}^d$,

$$k(x,y) = \prod_{i=1}^{d}\left(1 + ae^{-b(x_i-y_i)^2}\right) = \sum_{I\subset\{1,\cdots,d\}} a^{\#(I)}e^{-b\|\mathbf{x}_I-\mathbf{y}_I\|^2}.$$

- A variation on the Gaussian kernel: Mahalanobis kernel,

$$k_\Sigma(x,y) = e^{-(\mathbf{x}-\mathbf{y})^T\Sigma^{-1}(\mathbf{x}-\mathbf{y})},$$

  idea: correct for discrepancies between the magnitudes and correlations of different variables.
- Usually Σ is the empirical covariance matrix of a sample of points.

Kernels on vectors

- These kernels can be seen as *meta*-kernels which can use any feature representation.
- Example: Gaussian kernel of Gaussian kernel feature maps,

$$k_{G^2}(\mathbf{x},\mathbf{y}) = k_G\left(e^{-\frac{\|\mathbf{x}-\cdot\|^2}{2\sigma^2}}, e^{-\frac{\|\mathbf{y}-\cdot\|^2}{2\sigma^2}}\right) = e^{-\frac{2-e^{-\frac{\|\mathbf{x}-\mathbf{y}\|^2}{2\sigma^2}}}{2\lambda^2}}.$$

- Not sure this is very useful though!

- Indeed, the real challenge is not to define funky kernels,

$$\boxed{\text{the challenge is to tune the parameters } b, d, \sigma, \Sigma.}$$

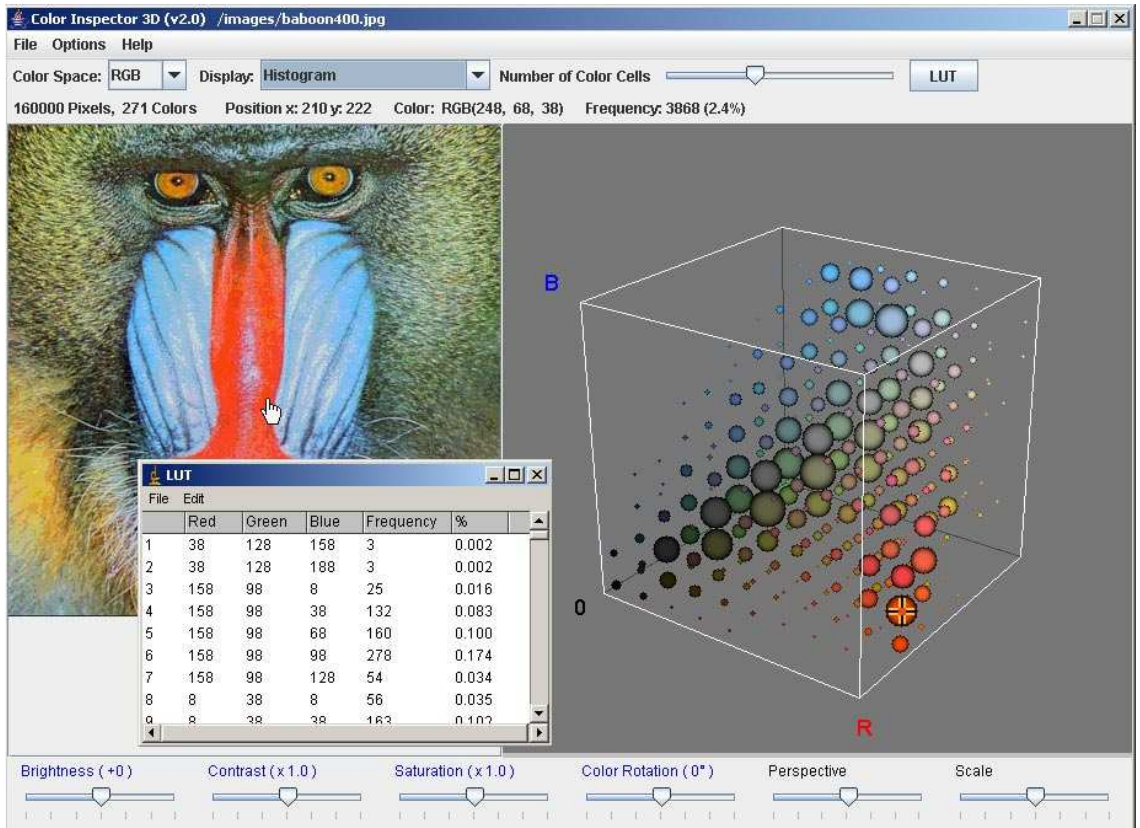
Kernels on structured objects

- Structured objects?
  - texts, webpages, documents
  - sounds, speech, music,
  - images, video segments, movies,
  - 3d structures, sequences, trees, graphs
- Structured objects means
  - objects with **a tricky structure**,
  - which cannot be simply embedded in a vector space of small dimensionality,
  - without obvious algebraic properties,

$$\boxed{\text{structured object} = \text{that which cannot be represented in a (small) Euclidian space}}$$

Vectors in $\mathbb{R}^n_+$ and Histograms

- A powerful and popular feature representation for structured objects: **histograms of smaller building-blocks of the object**:



- histograms are simple instances of **probability measures**,
  - nonnegative coordinates, sum up to 1.

Standard metrics for Histograms

Information geometry, introduced yesterday, studies distances between densities.

- Reference : [AN01]
- An abridged bestiary of **negative definite distances** on the probability simplex:

$$\psi_{JD}(\theta,\theta') = h\left(\frac{\theta+\theta'}{2}\right) - \frac{h(\theta)+h(\theta')}{2},$$

$$\psi_{\chi^2}(\theta,\theta') = \sum_i \frac{(\theta_i-\theta'_i)^2}{\theta_i+\theta'_i}, \quad \psi_{TV}(\theta,\theta') = \sum_i |\theta_i-\theta'_i|,$$

$$\psi_{H_2}(\theta,\theta') = \sum_i |\sqrt{\theta_i}-\sqrt{\theta'_i}|^2, \quad \psi_{H_1}(\theta,\theta') = \sum_i |\sqrt{\theta_i}-\sqrt{\theta'_i}|.$$

- Recover kernels through

$$k(\theta,\theta') = e^{-t\psi}, \quad t>0$$

Information Diffusion Kernel [LL05,ZLC05]

- Solve the heat equation on the multinomial manifold, using the Fisher metric
- Approximate the solution with

$$k_{\Sigma_d}(\theta,\theta') = e^{-\frac{1}{t}\arccos^2(\sqrt{\theta}\cdot\sqrt{\theta'})},$$

- $\arccos^2$ is the **squared** **geodesic distance** between θ and θ' as elements from the unit sphere ($\theta_i \to \sqrt{\theta_i}$).
- In [ZLC05]: the use of

$$k_{\Sigma_d}(\theta,\theta') = e^{-\frac{1}{t}\arccos(\sqrt{\theta}\cdot\sqrt{\theta'})},$$

  is advocated.
- the geodesic distance is a n.d. kernel on the *whole sphere* ($\arccos^2$ is not).

Transportation Metrics for Histograms

Beyond information geometry, the family of **transportation distances**.

- Suppose $\mathbf{r} = (r_1, \cdots, r_d)$ and $\mathbf{c} = (c_1, \cdots, c_d)$ are two histograms in $\mathbb{R}_+^n$.
- Define the set of transportations

$$U(\mathbf{r}, \mathbf{c}) = \{F \in \mathbb{R}_+^{d \times d} \,|\, F\mathbf{1} = \mathbf{r}, F^T\mathbf{1} = \mathbf{c}\}.$$

- Transportation distances between $\mathbf{r}$ and $\mathbf{c}$:

$$d_{\text{cost}}(\mathbf{r}\mathbf{c}) = \min_{F \in U(\mathbf{r}, \mathbf{c})} \text{cost}(F).$$

Monge-Kantorovich: $\text{cost}(F) = \langle F, D \rangle$ where D is a n.d. matrix.

- d_{cost} is **not** n.d. in the general case.
- Alternatives:

$$k_{\text{cost}}(\mathbf{r}\mathbf{c}) = \int_{F \in U(\mathbf{r}, \mathbf{c})} e^{-\text{cost}(F)}.$$

- works when $\text{cost} = 0$: the volume of $U(\mathbf{r}, \mathbf{c})$ is a p.d. kernel of $\mathbf{r}$ and $\mathbf{c}$. [Cut07]

Statistical Modeling and Kernels

Histograms cannot always summarize efficiently the structures of $\mathcal{X}$

- Statistical models of complex objects provide richer explanations:
  - Hidden Markov Models for sequences and time-series,
  - VAR, VARMA, ARIMA *etc.* models for time-series,
  - Branching processes for trees and graphs
  - Random Markov Fields for images *etc.*
- $\{\mathbf{x}_1, \cdots, \mathbf{x}_n\}$ are interpreted as i.i.d realizations of one or many densities on $\mathcal{X}$.
- These densities belong to a model $\boldsymbol{\{p_\theta, \theta \in \Theta \subset \mathbb{R}^d\}}$

Can we use **generative** (statistical) **models** in **discriminative** (kernel and metric based) **methods**?

Fisher Kernel

- The Fisher kernel [JH99] between two elements $\mathbf{x}, \mathbf{y}$ of $\mathcal{X}$ is

$$k_{\hat\theta}(\mathbf{x},\mathbf{y}) = \left(\frac{\partial \ln \boldsymbol{p_\theta}(\mathbf{x})}{\partial \theta}\Big|_{\hat{\boldsymbol\theta}}\right)^T \boldsymbol{J}_{\hat{\boldsymbol\theta}}^{-1} \left(\frac{\partial \ln \boldsymbol{p_\theta}(\mathbf{y})}{\partial \theta}\Big|_{\hat{\boldsymbol\theta}}\right),$$

  - $\hat{\boldsymbol\theta}$ has been selected using sample data ($e.g.$ MLE),
  - $\boldsymbol{J}_{\hat{\boldsymbol\theta}}^{-1}$ is the Fisher information matrix computed in $\hat\theta$.

- The statistical model $\boldsymbol{\{p_\theta, \theta \in \Theta\}}$ provides:

  - finite dimensional *features* through the **score vectors**,
  - A ***Mahalanobis metric*** associated with these vectors through $J_{\hat\theta}$.

- Alternative formulation:

$$k_{\hat\theta}(x,y) = e^{-\frac{1}{\sigma^2}\left(\nabla_{\hat\theta}\ln p_\theta(\mathbf{x}) - \nabla_{\hat\theta}\ln p_\theta(\mathbf{y})\right)^T J_{\hat\theta}^{-1}\left(\nabla_{\hat\theta}\ln p_\theta(\mathbf{x}) - \nabla_{\hat\theta}\ln p_\theta(\mathbf{y})\right)}.$$

with the meta-kernel idea.

Fisher Kernel Extended [TKR+02,SG02]

- Minor extensions, useful for binary classification:
- Estimate $\hat{\theta}_1$ and $\hat{\theta}_2$ for each class respectively,
- consider the score vector of the likelihood ratio

$$\phi_{\hat{\theta}_1,\hat{\theta}_2} : \mathbf{x} \mapsto \left(\left.\frac{\partial \ln \frac{p_{\theta_1}(\mathbf{x})}{p_{\theta_2}(\mathbf{x})}}{\partial \vartheta}\right|_{\hat{\vartheta}=(\hat{\theta}_1,\hat{\theta}_2)} \right),$$

  where $\vartheta = (\theta_1, \theta_2)$ is in Θ^2.

- Use this logratio's score vector to propose instead the kernel

$$(x, y) \mapsto \phi_{\hat{\theta}_1,\hat{\theta}_2}(\mathbf{x})^T \phi_{\hat{\theta}_1,\hat{\theta}_2}(\mathbf{y}).$$

Mutual Information Kernel: densities as feature extractors

- More **bayesian** flavor $\rightarrow$ drops maximum-likelihood estimation of θ. [See02]
- Instead, use **prior knowledge** on $\boldsymbol{\{p_\theta, \theta \in \Theta\}}$ through a **density** ω on Θ
- Mutual information kernel k_ω:

$$k_\omega(\mathbf{x}, \mathbf{y}) = \int_\Theta p_\theta(\mathbf{x}) p_\theta(\mathbf{y})\, \omega(d\theta).$$

- The feature maps $0 \leq p_\theta(\mathbf{x}) \leq 1$ and $0 \leq p_\theta(\mathbf{y}) \leq 1$.

k_ω is big whenever many **common** densities p_θ score high probabilities for **both** $\mathbf{x}$ and $\mathbf{y}$

- Explicit computations sometimes possible, **namely conjugate priors**.
- Example: context-tree kernel for strings.

Mutual Information Kernel & Fisher Kernels

The Fisher kernel is a maximum *a posteriori* approximation of the MI kernel.

- What? How? by setting the prior ω to the multivariate Gaussian density

$$\mathcal{N}(\hat{\theta}, J_{\hat{\theta}}^{-1}),$$

  an approximation known as Laplace's method,

- Writing

$$\Phi(x) = \nabla_{\hat{\theta}} \ln p_\theta(x) = \frac{\partial \ln p_\theta(x)}{\partial \theta}\Big|_{\hat{\theta}}$$

  we get

$$\log p_\theta(x) \approx \log p_{\hat{\theta}}(x) + \Phi(x)(\theta - \hat{\theta}).$$

Mutual Information Kernel & Fisher Kernels

- Using $\mathcal{N}(\hat{\theta}, J_{\hat{\theta}}^{-1})$ for ω yields

$$\begin{aligned} k(x,y) &= \int_\Theta p_\theta(\mathbf{x}) p_\theta(\mathbf{y})\, \omega(d\theta), \\ &\approx C \int_\Theta e^{\log p_{\hat{\theta}}(x) + \Phi(x)^T(\theta-\hat{\theta})} e^{\log p_{\hat{\theta}}(y) + \Phi(y)^T(\theta-\hat{\theta})} \;\; e^{-(\theta-\hat{\theta})^T J_{\hat{\theta}}(\theta-\hat{\theta})} d\theta \\ &= C p_{\hat{\theta}}(x) p_{\hat{\theta}}(y) \int_\Theta e^{(\Phi(x)+\Phi(y))^T(\theta-\hat{\theta}) + (\theta-\hat{\theta})^T J_{\hat{\theta}}(\theta-\hat{\theta})} d\theta \\ &= C' p_{\hat{\theta}}(x) p_{\hat{\theta}}(y) e^{\frac{1}{2}(\Phi(x)+\Phi(y))^T J_{\hat{\theta}}^{-1}(\Phi(x)+\Phi(y))} \end{aligned} \tag{1}$$

- the kernel
$$\tilde{k}(x,y) = \frac{k(x,y)}{\sqrt{k(x,x)k(y,y)}}$$
is equal to the Fisher kernel in exponential form.

Marginalized kernels - Graphs and Sequences

- Similar ideas: leverage **latent variable models.** [TKA02,KTI03]
- For **location** or **time-based** data,
  - the probability of emission of a token x_i is conditioned by
  - an **unobserved** latent variable $s_i \in \mathcal{S}$, where $\mathcal{S}$ is a finite space of possible states.
- for observed sequences $\mathbf{x} = (x_1, \cdots, x_n), \mathbf{y} = (y_1, \cdots, y_n)$, sum over all possible state sequences the **weighted** product of **these probabilities**:

$$k(x, y) = \sum_{s \in \mathcal{S}} \sum_{s' \in \mathcal{S}} \boldsymbol{p(s|x)\, p(s'|y)\, \kappa\left((x, s), (y, s')\right)}$$

- closed form computations exist for graphs & sequences.

Kernels on MLE parameters

- Use model directly to extract a single representation from observed points:

$$x \mapsto \hat{\theta}_x, \quad y \mapsto \hat{\theta}_y,$$

  through MLE for instance.

- compare $\mathbf{x}$ and $\mathbf{y}$ through a kernel k_Θ on Θ,

$$k(x, y) = k_\Theta(\hat{\theta}_{\mathbf{x}}, \hat{\theta}_{\mathbf{y}}).$$

- Bhattacharrya affinities:

$$k_\beta(\mathbf{x}, \mathbf{y}) = \int_{\mathcal{X}} p_{\hat{\theta}_{\mathbf{x}}}(z)^\beta p_{\hat{\theta}_{\mathbf{y}}}(z)^\beta dz$$

  for $\beta > 0$.