Polyhedral Computation

Linear Classifiers & the SVM

mcuturi@i.kyoto-u.ac.jp

Statistical Inference

- **Statistical**: useful to study random systems...
  - Mutations, environmental changes *etc.*→ **life is random**!
- **Inference**: learn rules using observations assuming some "stationarity".

in this talk: "**yes/no**" rules = "**binary classification**"

- given an image, does it contain the photograph of a human face?
- given a patient's genome, is it safe/effective to give him medecine `XYZ`?
- given a patient's genome, is (s)he at risk of developing Parkinson's disease?
- *etc.*

"**binary classification**" ⇒ **simple 0/1 predictions** for **well-understood problems**

Data

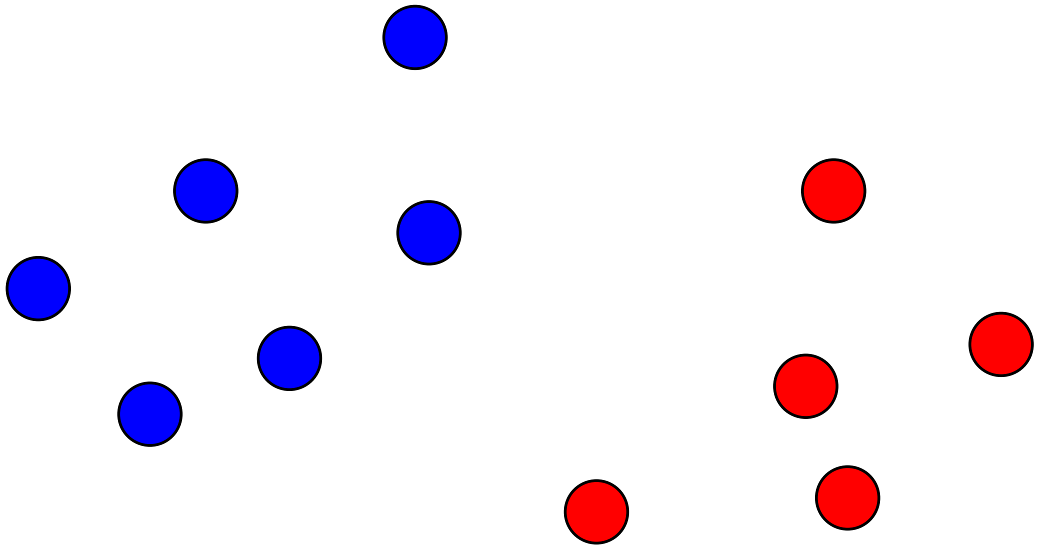
- The **Data** we have: a bunch of vectors $\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, \cdots, \mathbf{x}_N$.
- Ideally, to infer a "yes/no" rule, we need the **correct answer** for each vector.
- We consider thus a set of **pairs of vector/bit**

$$\text{"training set"} \;=\; \left\{ \left(\mathbf{x}_i = \begin{bmatrix} x_1^i \\ x_2^i \\ \vdots \\ x_d^i \end{bmatrix} \in \mathbb{R}^d,\; \mathbf{y}_i \in \{0,1\} \right)_{i=1..N} \right\}$$

- For illustration purposes **only** we will consider **vectors in the plane**, $d = 2$.
- Points are easier to represent in 2 dimensions than in 20.000...
- The ideas for $d \gg 3$ are **exactly the same**.

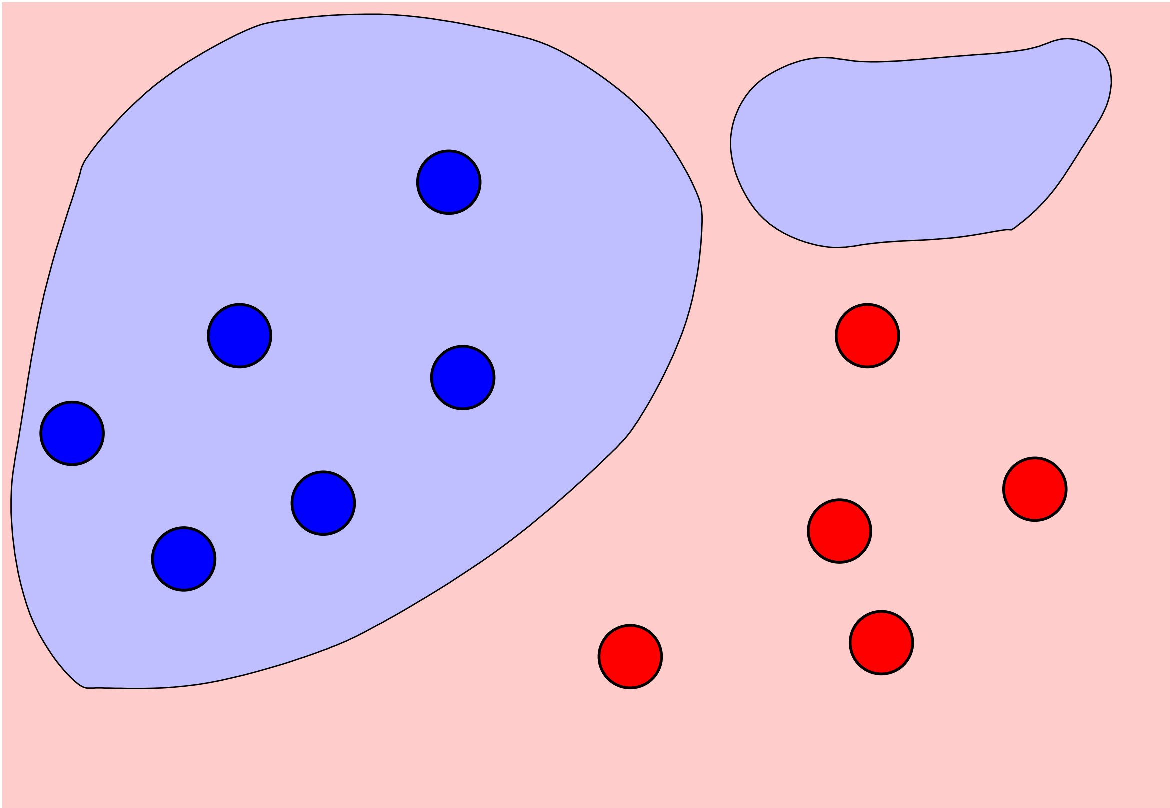
Many thanks to J.P. Vert for some of the following slides

Classification Separation Surfaces for Vectors



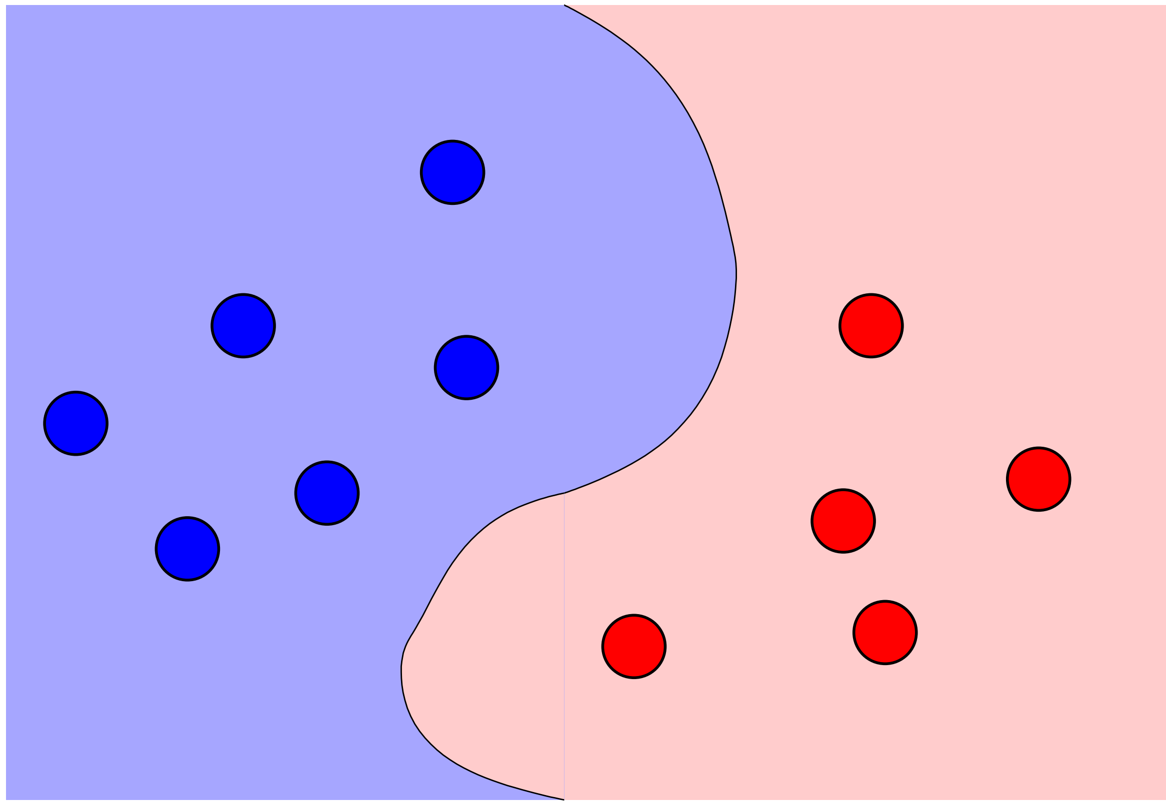
What is a classification rule?

Classification Separation Surfaces for Vectors



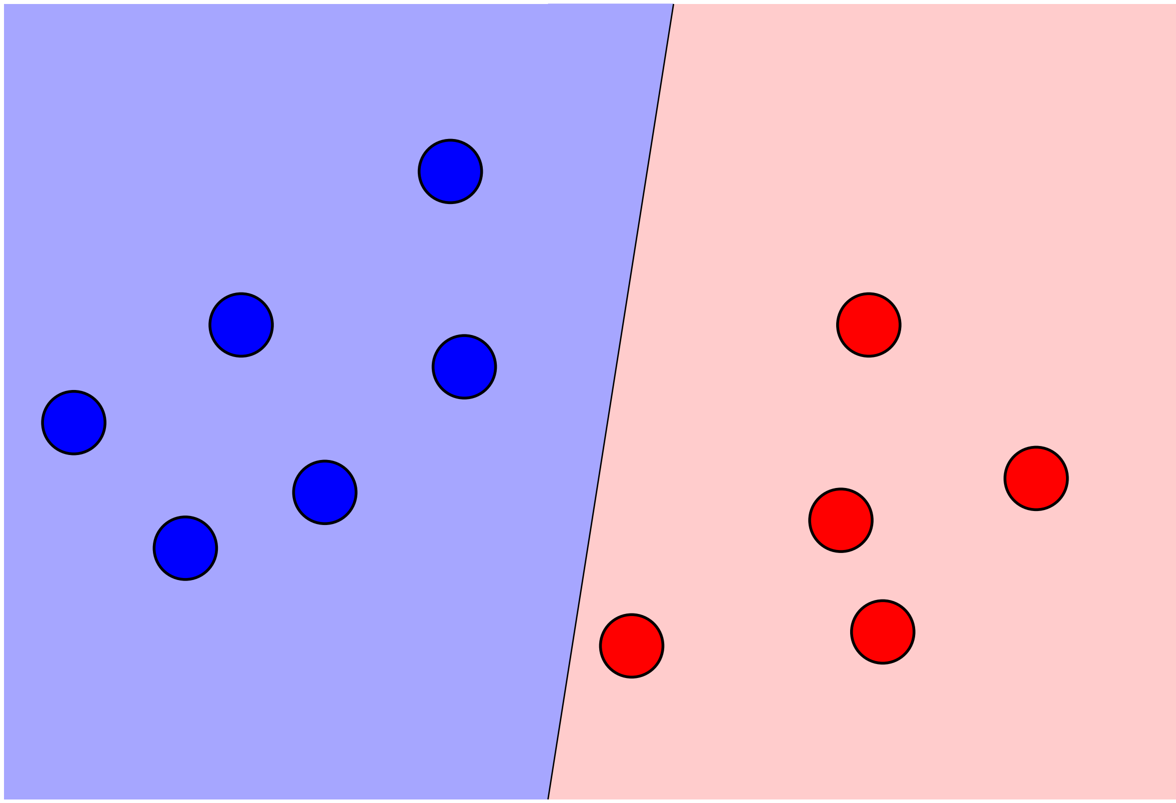
Classification rule = a partition of $\mathbb{R}^d$ into two sets

Classification Separation Surfaces for Vectors



Can be defined by a single surface, *e.g.* a curved line

Classification Separation Surfaces for Vectors



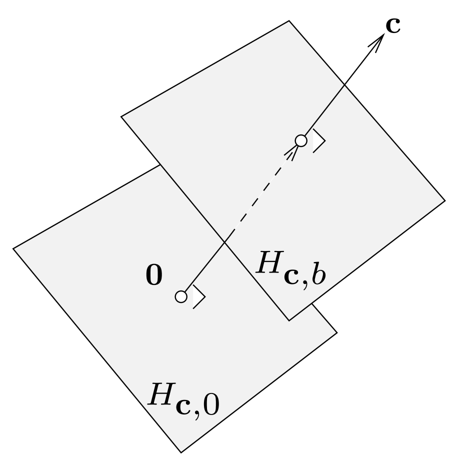
Even more **simple**: using **straight lines** and halspaces.

Linear Classifiers

- **Straight lines** (hyperplanes when $d > 2$) are **the simplest type** of classifiers.
- A hyperplane $H_{\mathbf{c},b}$ is a set in $\mathbb{R}^d$ defined by
  - a normal vector $\mathbf{c} \in \mathbb{R}^d$
  - a constant $b \in \mathbb{R}$. as

$$H_{\mathbf{c},b} = \{\mathbf{x} \in \mathbb{R}^d \,|\, \mathbf{c}^T\mathbf{x} \;=\; b\}$$

- Letting b vary we can "slide" the hyperplane across $\mathbb{R}^d$

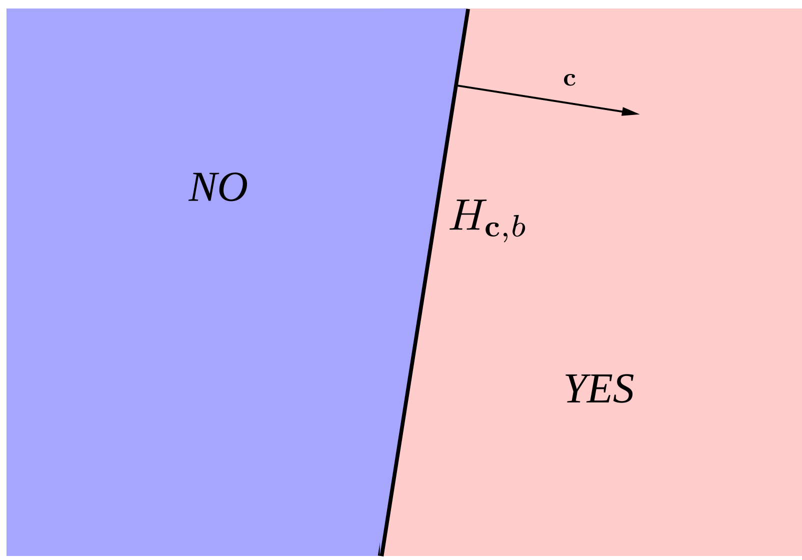


Linear Classifiers

- Exactly like lines in the plane, hypersurfaces **divide** $\mathbb{R}^d$ into **two** halfspaces,

$$\left\{\mathbf{x} \in \mathbb{R}^d \,|\, \mathbf{c}^T\mathbf{x} < b\right\} \cup \left\{\mathbf{x} \in \mathbb{R}^d \,|\, \mathbf{c}^T\mathbf{x} \geq b\right\} = \mathbb{R}^d$$

- Linear classifiers attribute the "yes" and "no" answers given arbitrary $\mathbf{c}$ and b.



- Assuming we only look at halfspaces for the decision surface...

  ...*how to* ***choose*** *the* ***"best"*** $(\mathbf{c}^\star, b^\star)$ *given a* ***training sample****?*

Linear Classifiers

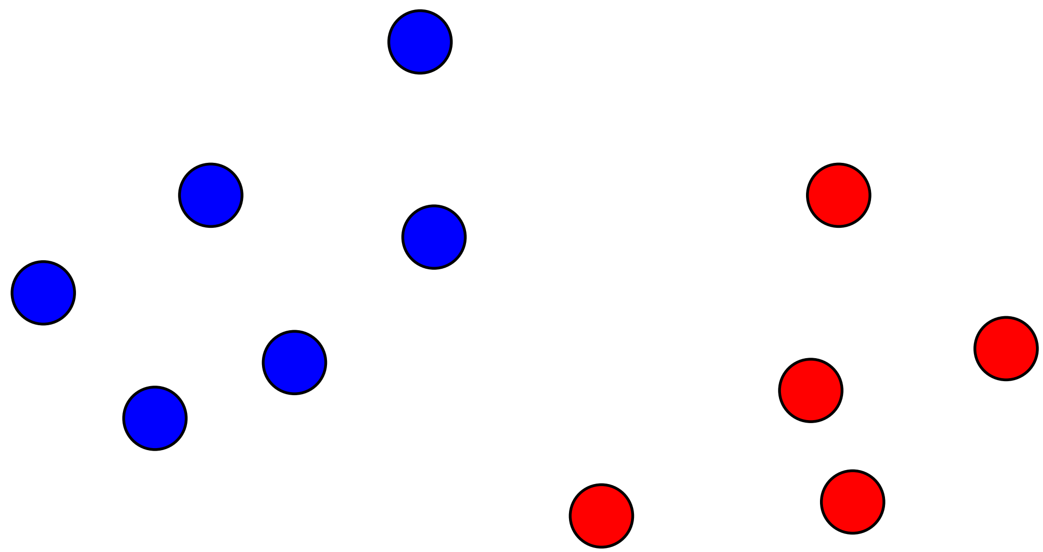
- This specific question,

$$\text{“training set”} \;\left\{\left(\mathbf{x}_i \in \mathbb{R}^d,\ \mathbf{y}_i \in \{0,1\}\right)_{i=1..N}\right\} \xrightarrow{????} \text{“best”}\, \mathbf{c}^\star,\ b^\star$$

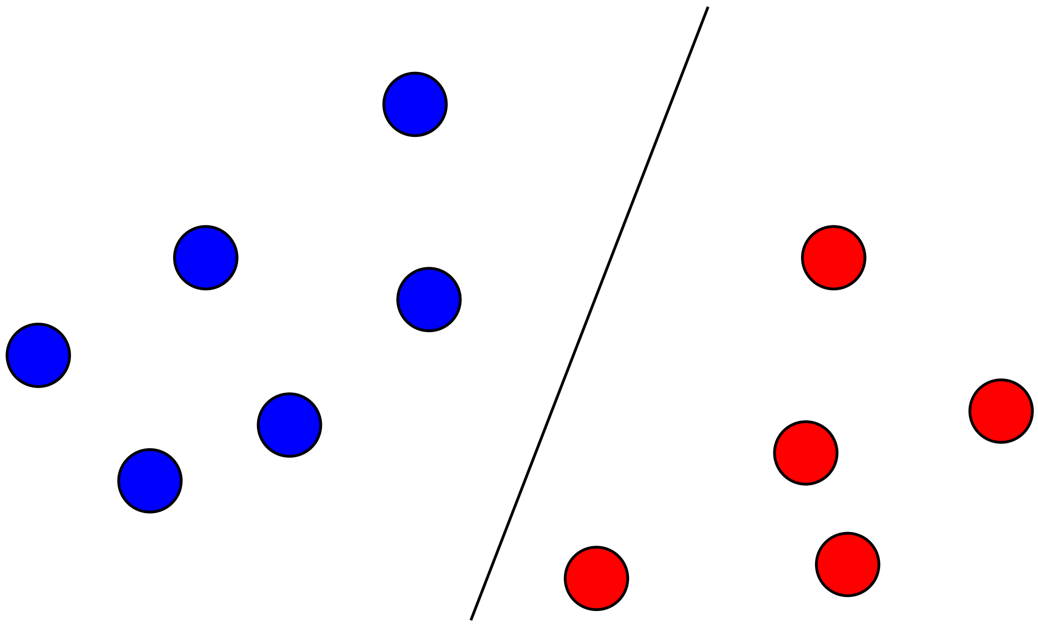
  has different answers. Depends on the meaning of “best” [4]:
- **Linear Discriminant Analysis** (or Fisher's Linear Discriminant);
- **Logistic regression** maximum likelihood estimation;
- **Perceptron**, a one-layer neural network;
- *etc.*

$\boxed{\text{Today's focus: the } \textbf{Support vector machine}}$ [5]

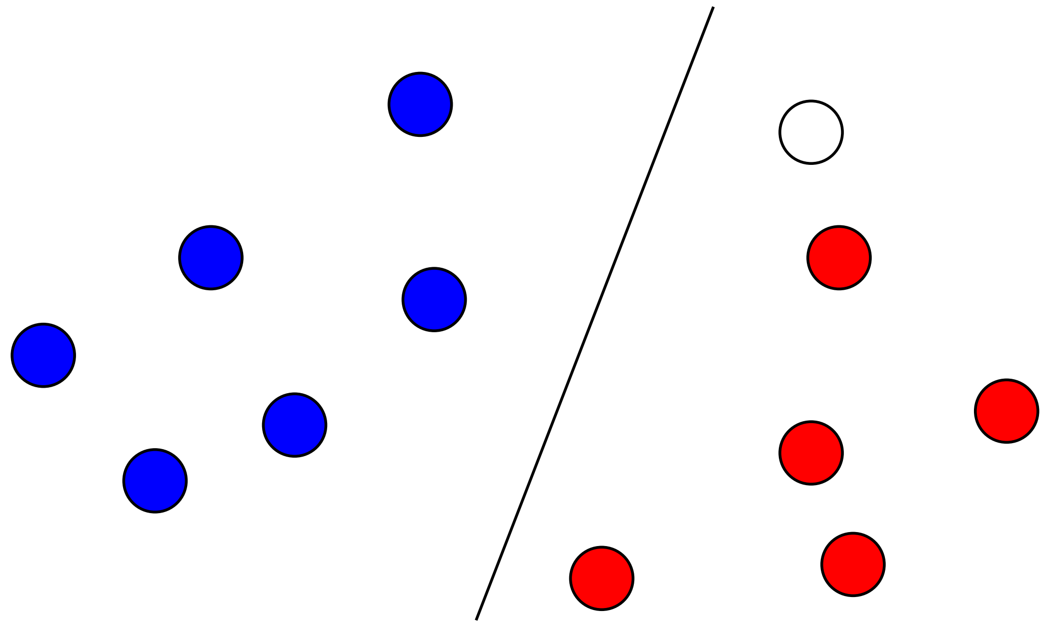
Classification Separation Surfaces for Vectors



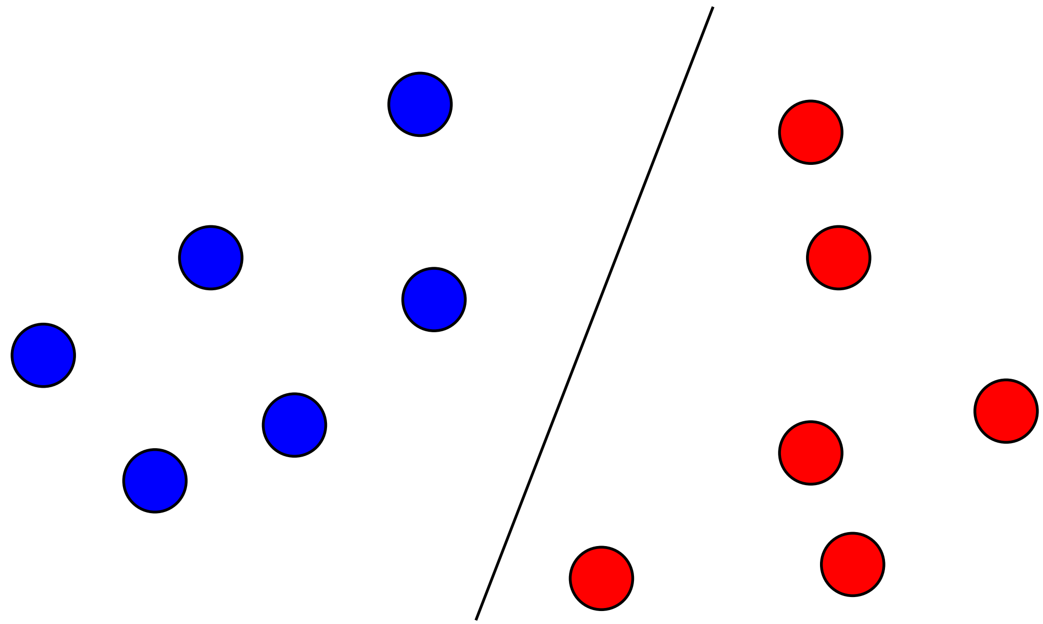
Classification Separation Surfaces for Vectors



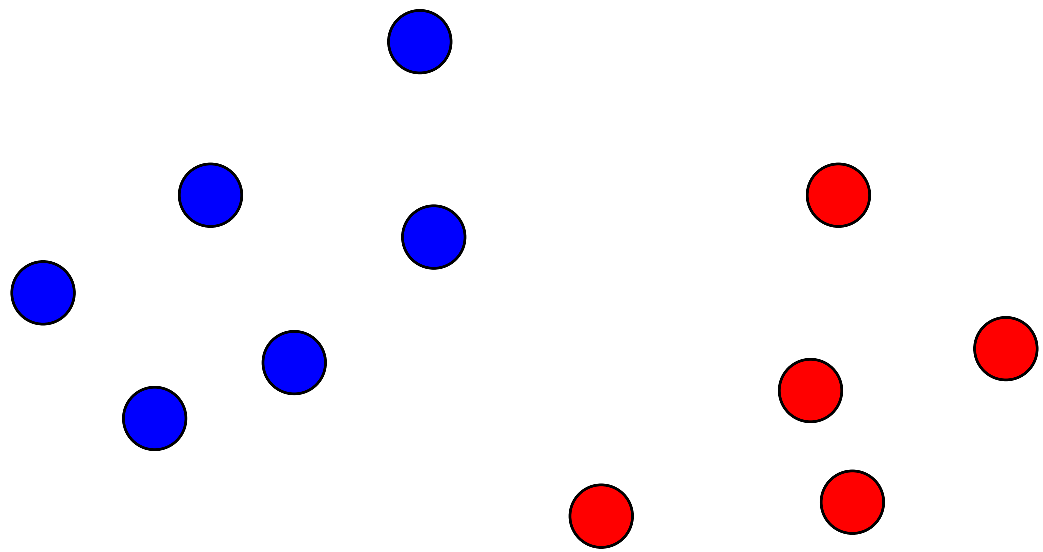
Classification Separation Surfaces for Vectors



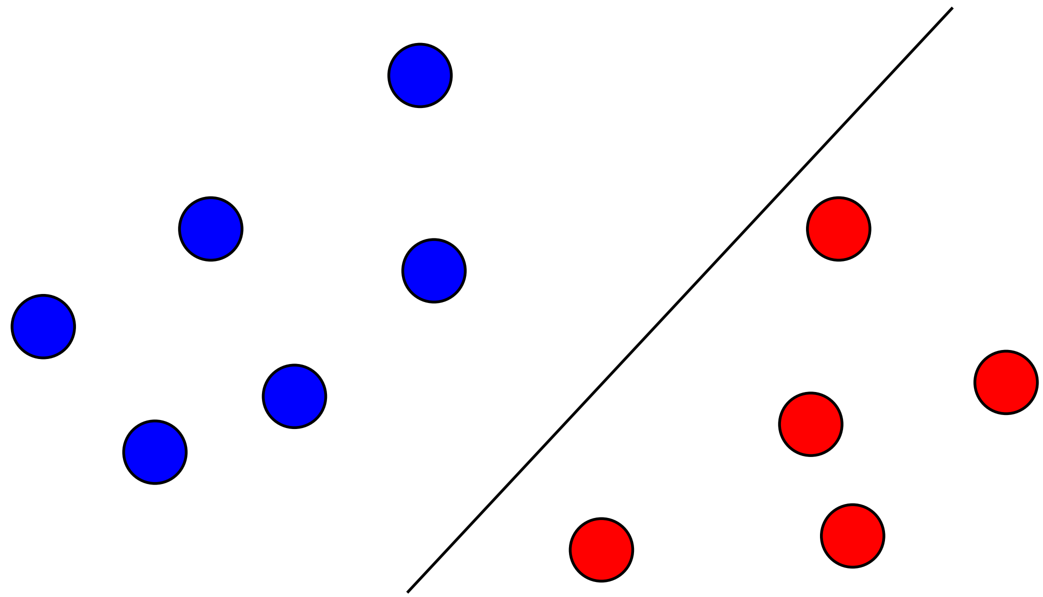
Classification Separation Surfaces for Vectors



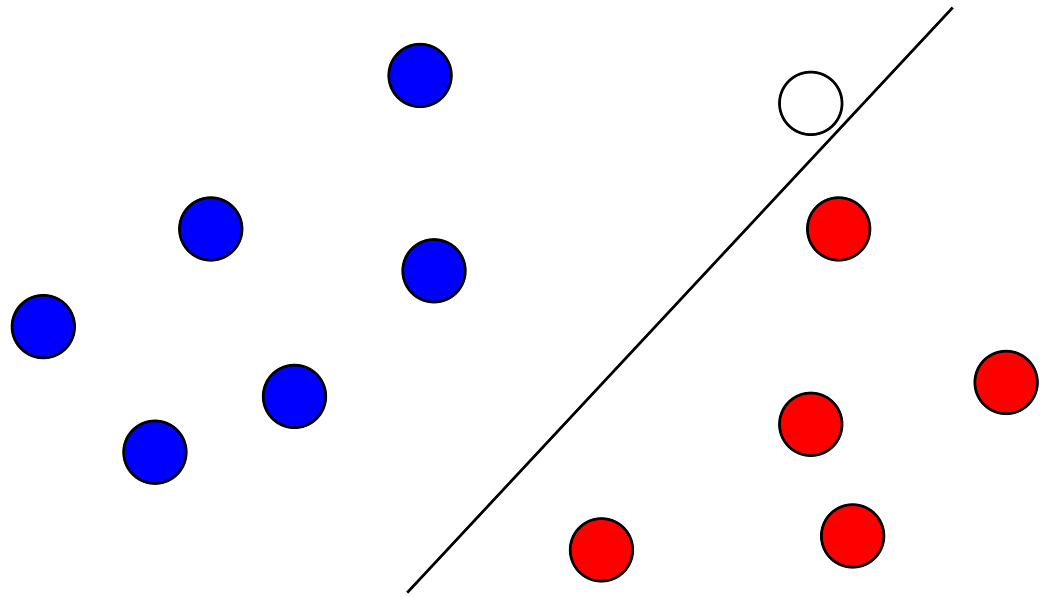
Linear classifier, some degrees of freedom



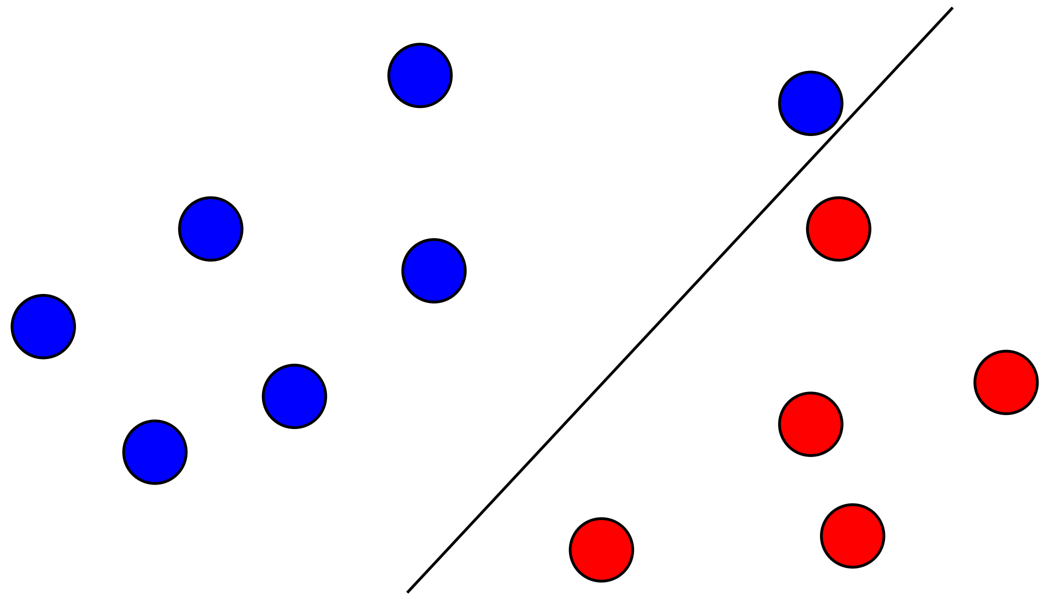
Linear classifier, some degrees of freedom



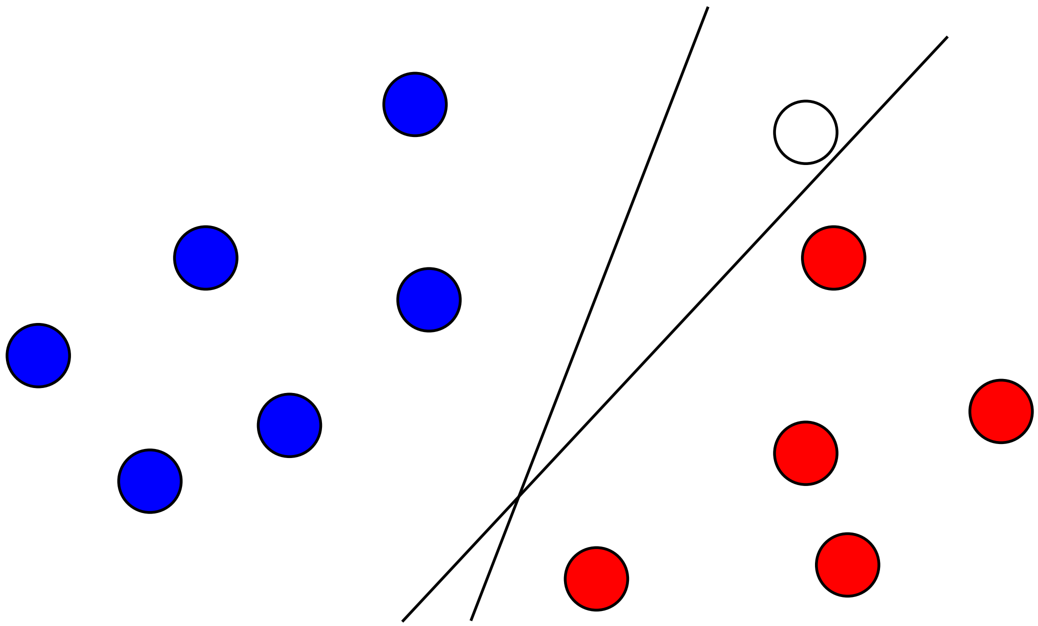
Linear classifier, some degrees of freedom



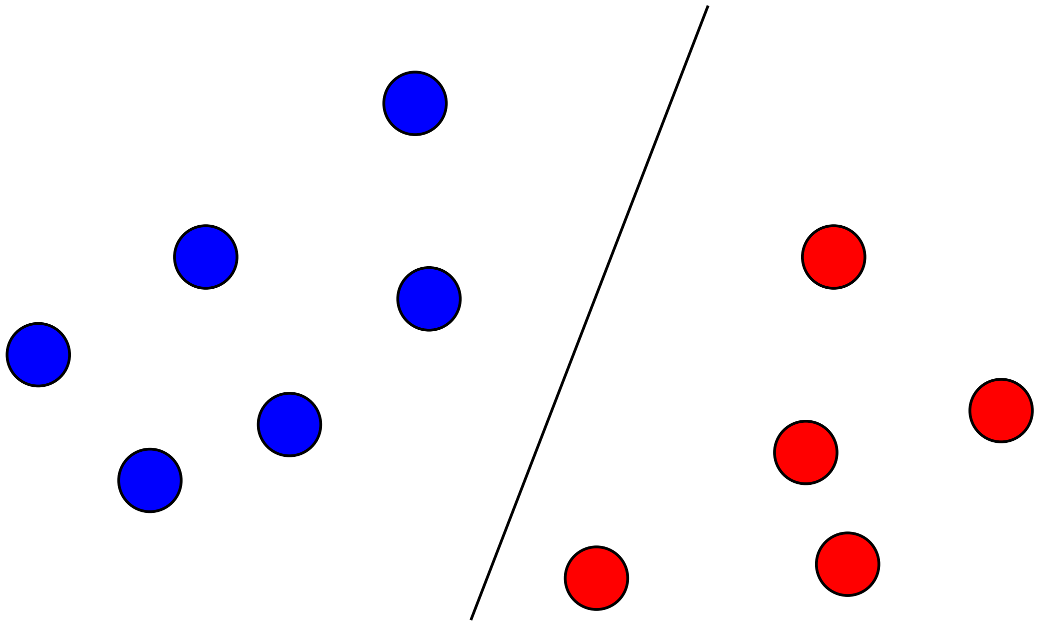
Linear classifier, some degrees of freedom



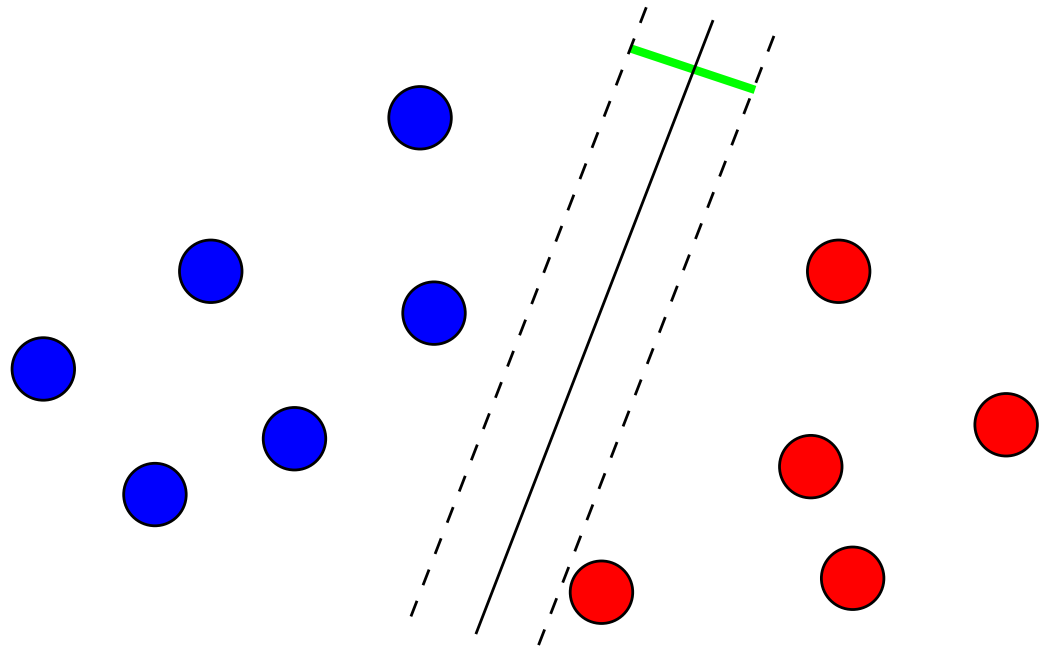
Which one is better?



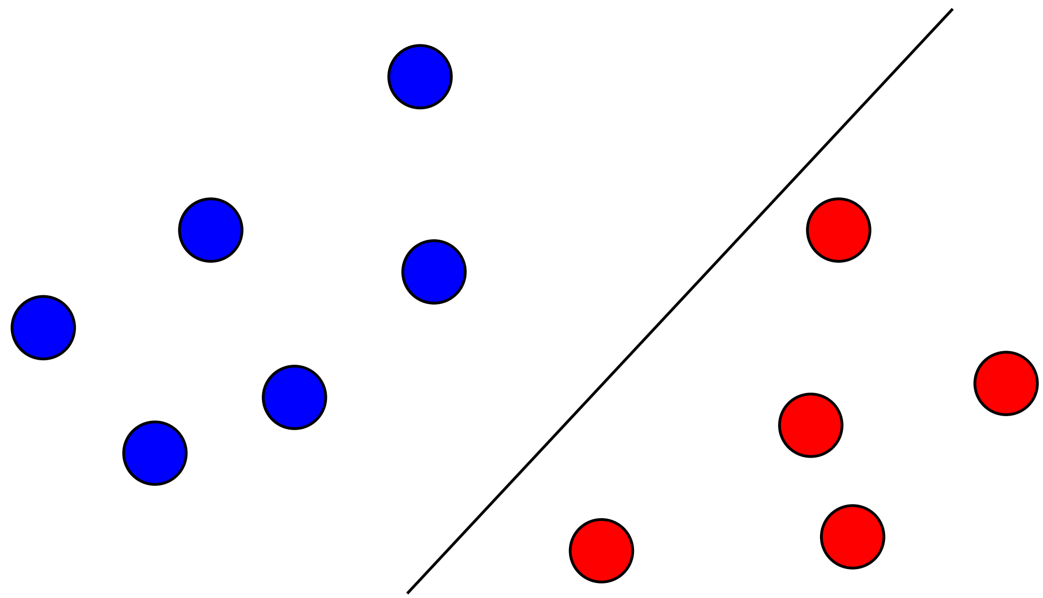
A criterion to select a linear classifier: the margin [2]



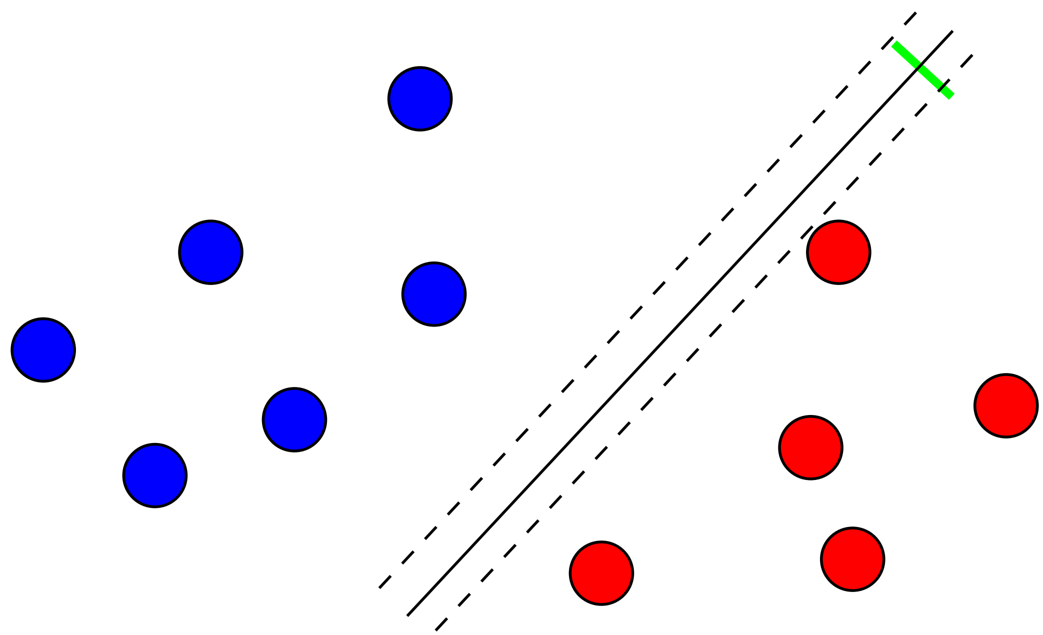
A criterion to select a linear classifier: the margin [2]



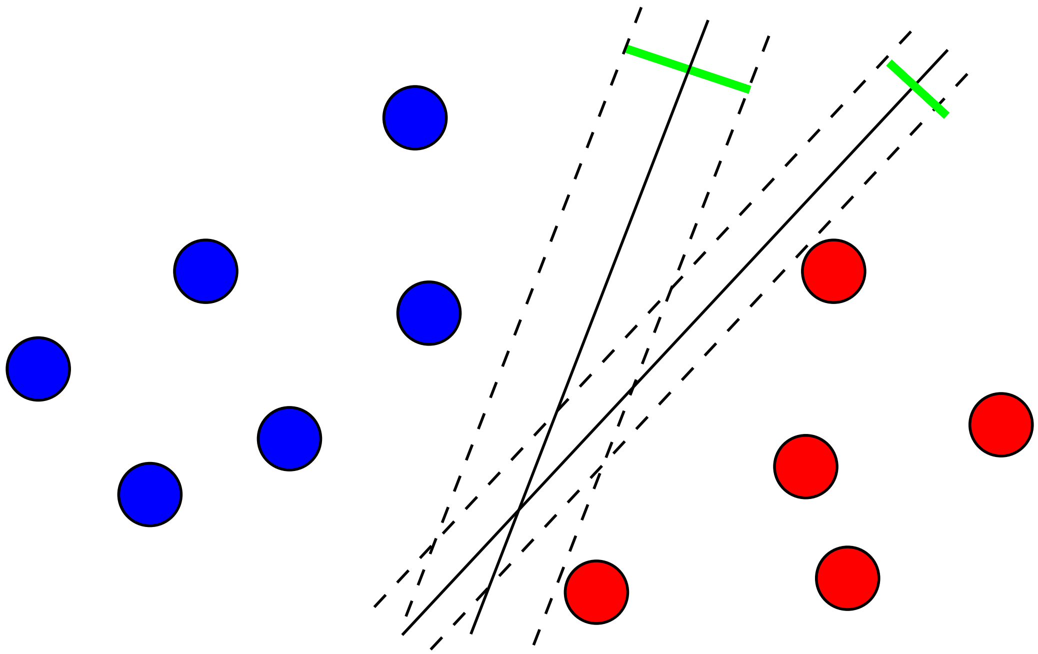
A criterion to select a linear classifier: the margin [2]



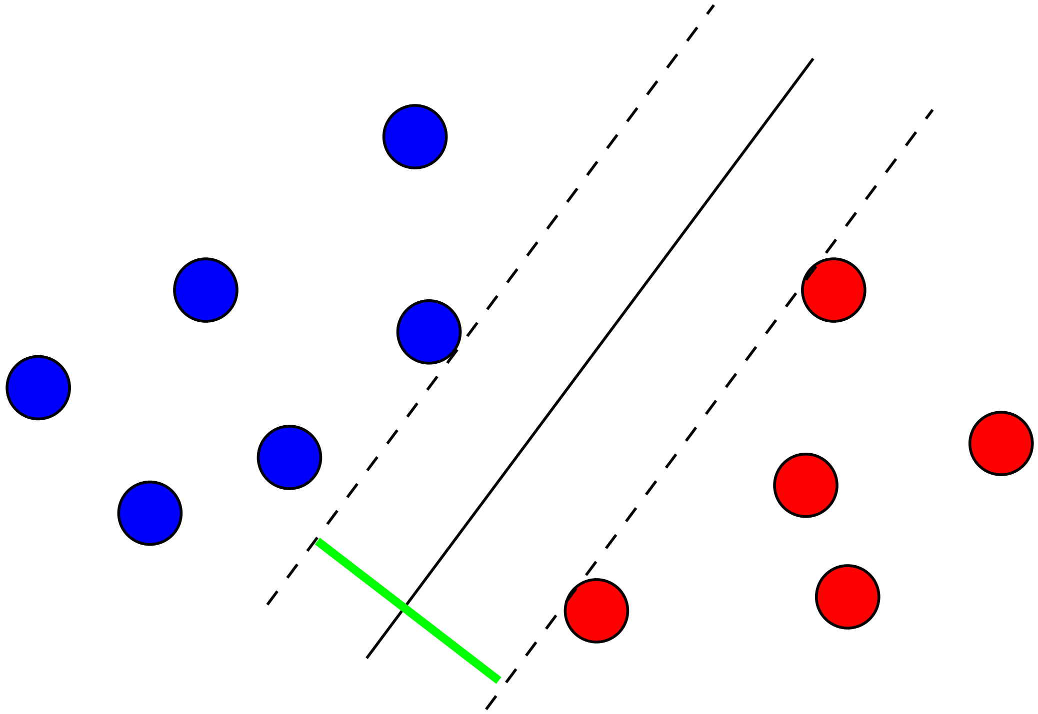
A criterion to select a linear classifier: the margin [2]



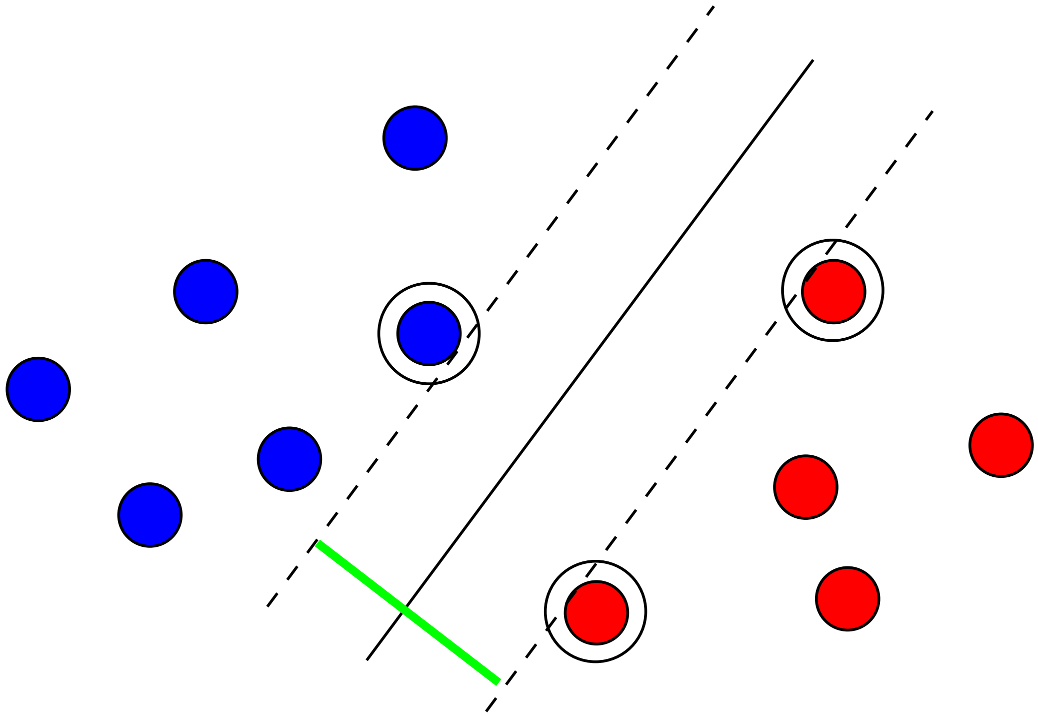
A criterion to select a linear classifier: the margin [2]



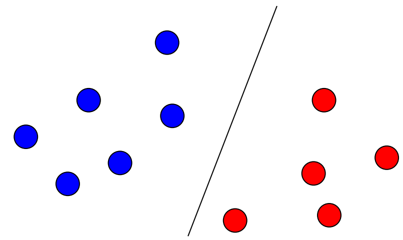
Largest Margin Linear Classifier [2]



Support Vectors with Large Margin



In equations



- 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:

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

- Next, we give a formula to compute the margin as a function of $\mathbf{w}$.
- Obviously, for any $t \in \mathbb{R}$,

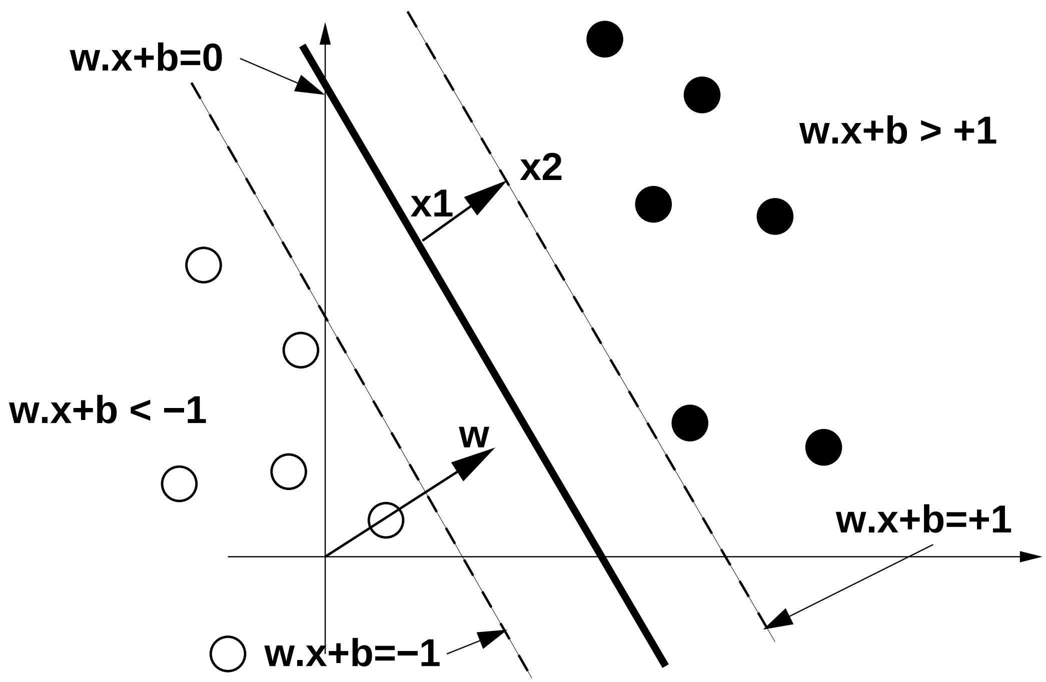
$$H_{\mathbf{w},b} = H_{t\mathbf{w},tb}$$

- Thus $\mathbf{w}$ and b are defined up to a multiplicative constant.
- We need to take care of this in the definition of the margin

How to find the largest separating hyperplane?

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

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



- Consider $\mathbf{x}_1$ and $\mathbf{x}_2$ such that $\mathbf{x}_2 - \mathbf{x}_1$ is parallel to w.

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

- Margin $= 2/\|\mathbf{w}\|$: 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:

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

where γ is by definition the **margin**.

All training points should be on the appropriate side

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

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

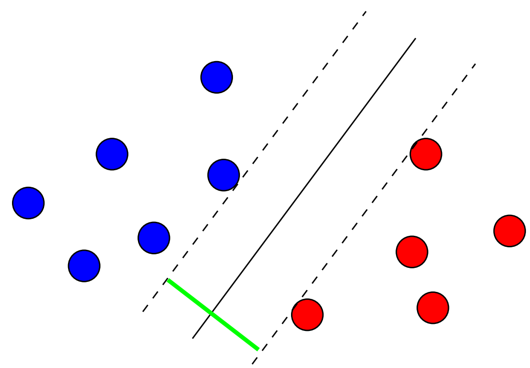
- For negative examples ($y_i = -1$) this means:

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

- 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**,
- one constraint for **each training point**.
- the **Lagrangian** is, for $\alpha \succeq 0$ (that is for each $\alpha_i \geq 0$)

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

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

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

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

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 $\alpha_i(\mathbf{y}_i \left(\mathbf{w}^T \mathbf{x}_i + b\right) - 1) = 0$,
  ...*hence*, either $\boldsymbol{\alpha_i = 0}$ or $\boldsymbol{\mathbf{y}_i \left(\mathbf{w}^T \mathbf{x}_i + b\right) = 1}$.
- $\alpha_i \neq 0$ **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** in $\mathbb{R}^n$, with *box constraints*.
α^* can be computed using optimization software
($e.g.$ built-in `matlab` function)

Recovering the optimal hyperplane

- With α^*, 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 $\alpha_i > 0$, $\mathbf{y}_i(\mathbf{w}^T \mathbf{x}_i + b) = 1$,

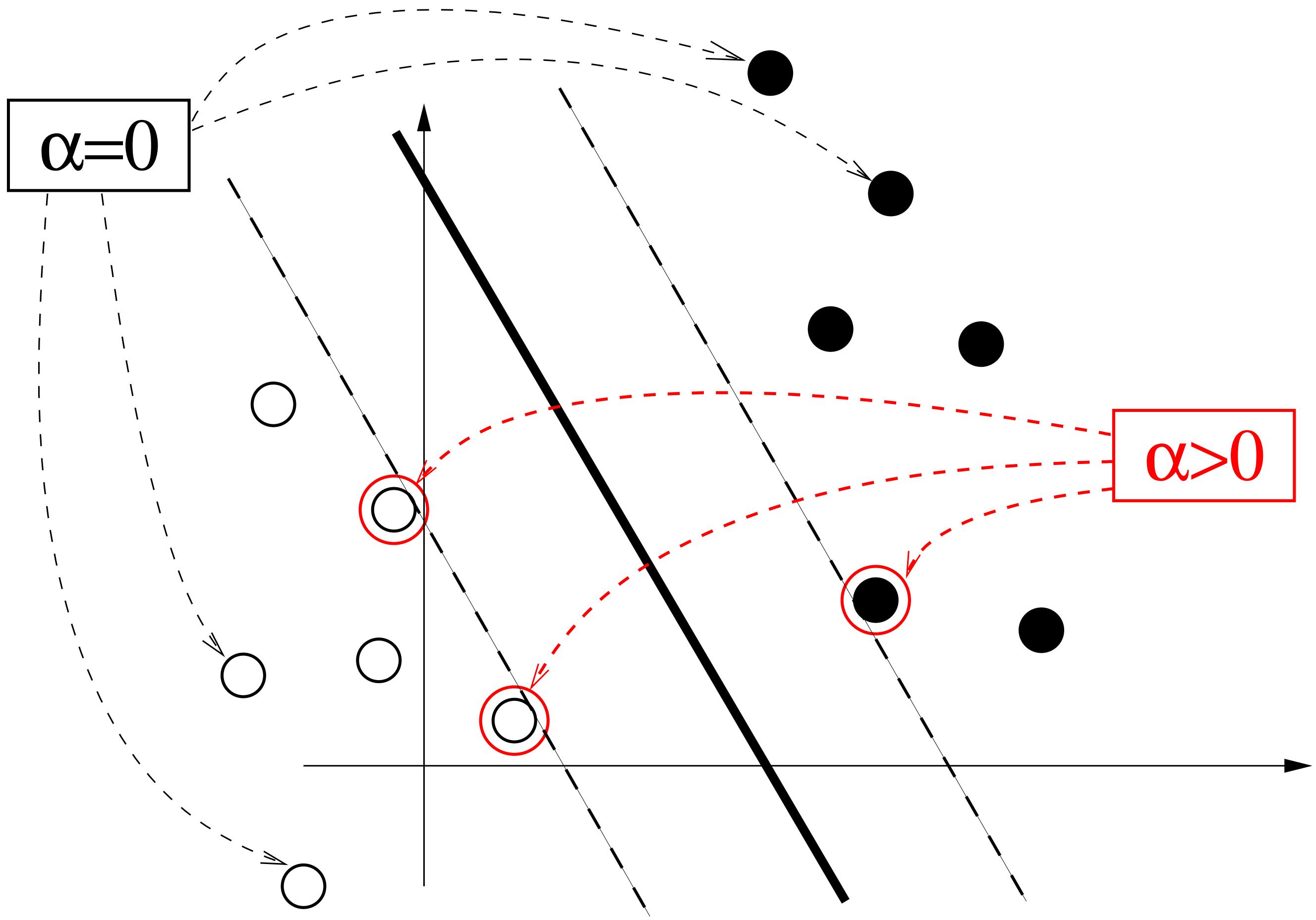
$$b^* = -\frac{1}{2}\left(\min_{\mathbf{y}_i=1,\alpha_i>0}(\mathbf{w}^T\mathbf{x}_i) + \max_{\mathbf{y}_i=-1,\alpha_i>0}(\mathbf{w}^T\mathbf{x}_i)\right)$$

- the **decision function** is therefore:

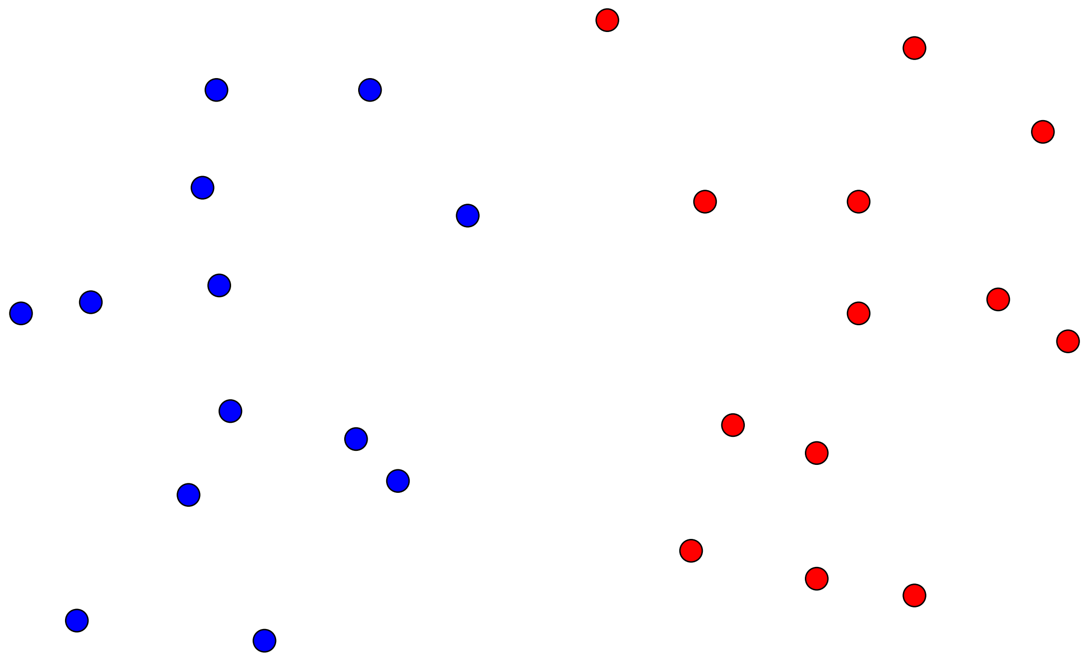
$$\begin{aligned} f^*(\mathbf{x}) &= \mathbf{w}^T\mathbf{x} + b^* \\ &= \sum_{i=1}^n y_i \alpha_i \mathbf{x}_i^T \mathbf{x} + b^*. \end{aligned}$$

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

Interpretation: support vectors

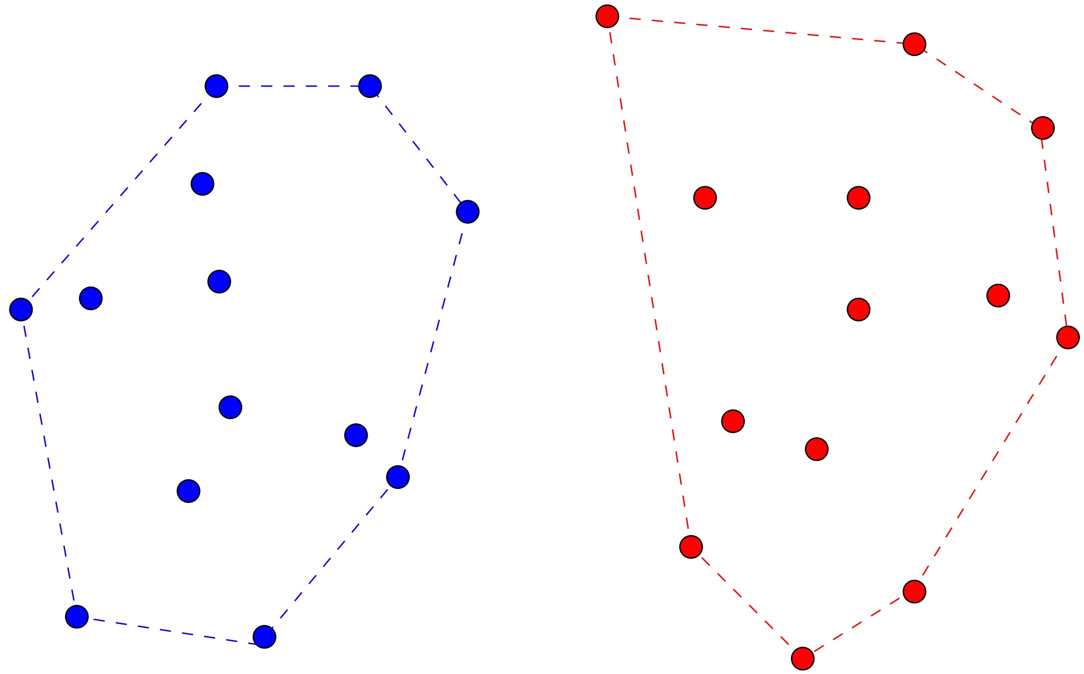


Another interpretation: Convex Hulls [1]



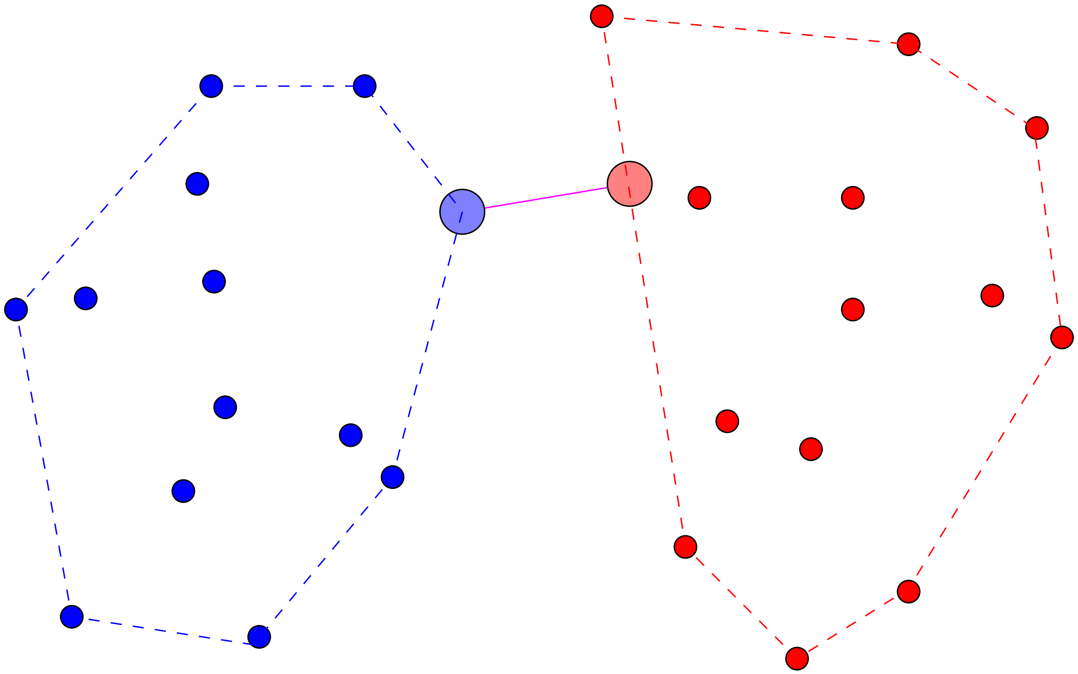
go back to 2 sets of points that are linearly separable

Another interpretation: Convex Hulls



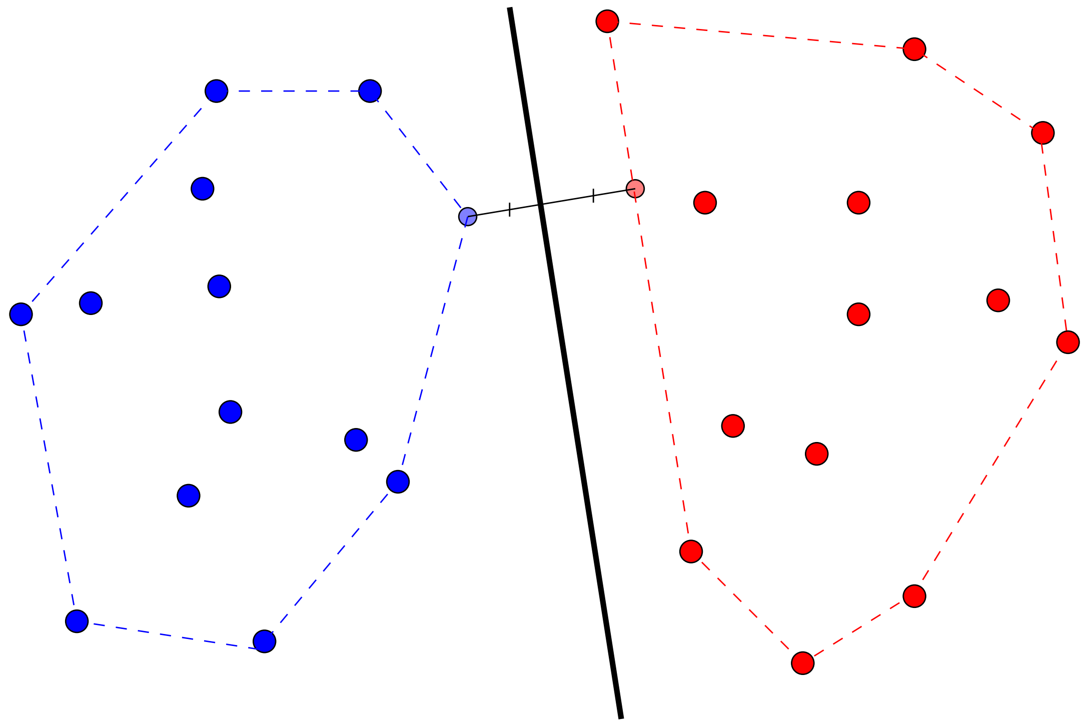
$$\boxed{\text{Linearly separable} = \text{convex hulls do not intersect}}$$

Another interpretation: Convex Hulls



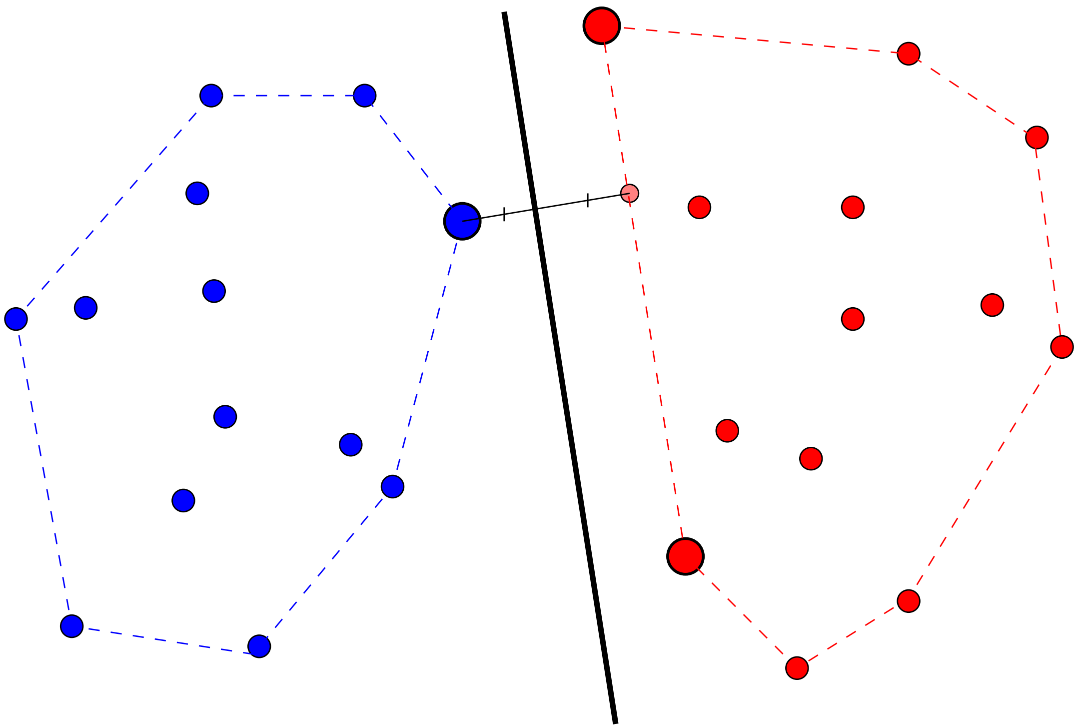
Find two closest points, one in each convex hull

Another interpretation: Convex Hulls



The SVM = bisection of that segment

Another interpretation: Convex Hulls



support vectors = extreme points of the faces on which the two points lie

A brief proof through duality

- Suppose that
  - all the points of the blue set are in a matrix $A \in \mathbb{R}^{d \times n_{-1}}$,
  - all the points of the red set are in a matrix $B \in \mathbb{R}^{d \times n_1}$

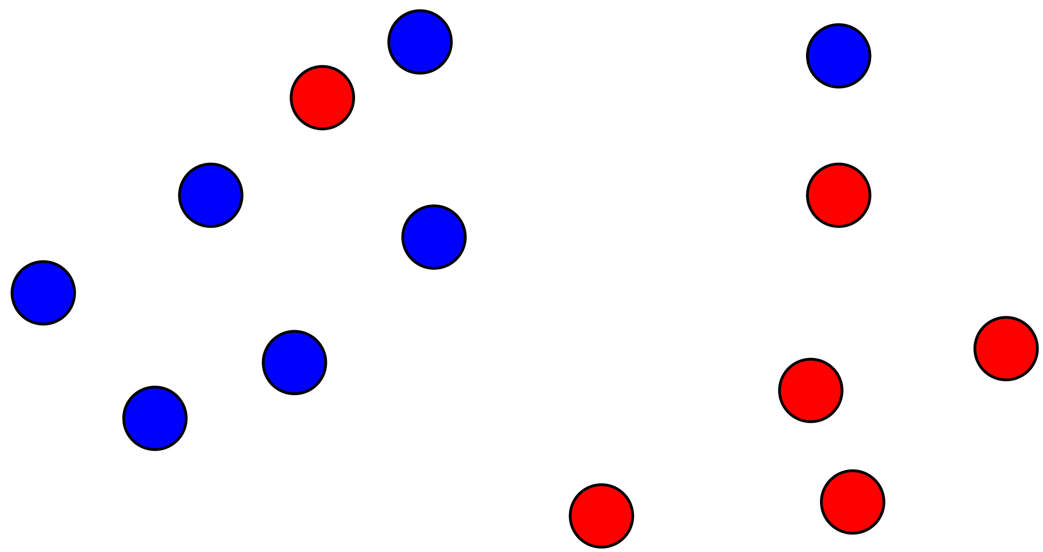
$$\boldsymbol{A} = \begin{bmatrix} \vdots & \cdots & \vdots \\ \boldsymbol{x_1} & \cdots & \boldsymbol{x_{n_{-1}}} \\ \vdots & \cdots & \vdots \end{bmatrix} \in \mathbb{R}^{\boldsymbol{d \times n_{-1}}}, \quad B = \begin{bmatrix} \vdots & \cdots & \vdots \\ x'_1 & \cdots & x'_{n_1} \\ \vdots & \cdots & \vdots \end{bmatrix} \in \mathbb{R}^{d \times n_1}.$$

- Finding the two points in question, and the minimal distance, is given by

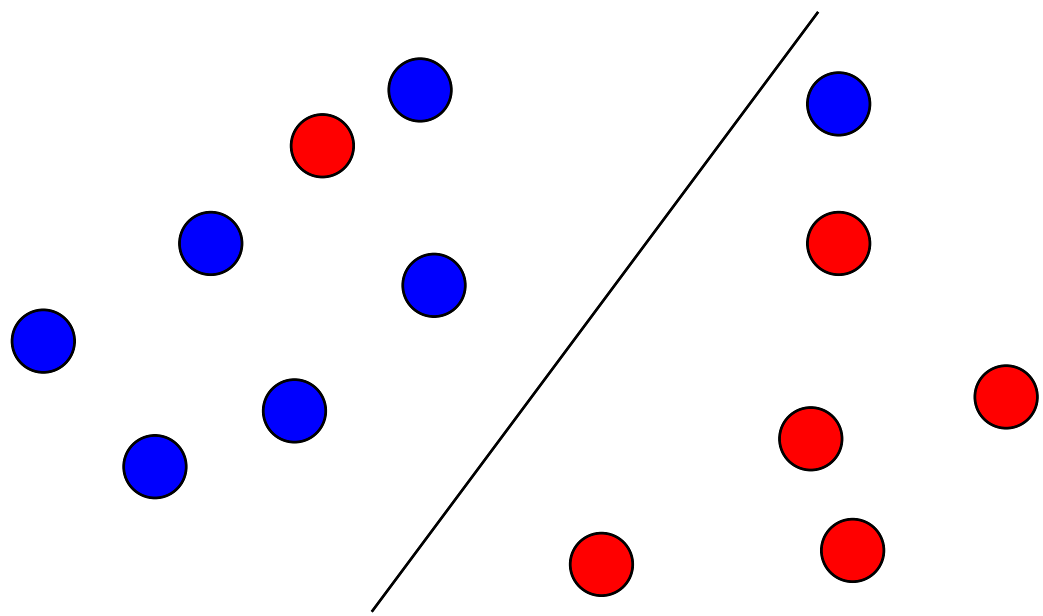
$$\begin{array}{ll} \text{minimize} & \|\boldsymbol{A}\mathbf{u} - B\mathbf{v}\|^2 \\ \text{subject to} & \mathbb{1}_{n_{-1}}^T \mathbf{u} = \mathbb{1}_{n_1}^T \mathbf{v} = 1 \\ & 0 \leq \mathbf{u} \in \mathbb{R}^{n_{-1}}, \mathbf{v} \in \mathbb{R}^{n_1} \end{array}$$

- Possible to prove that the primal SVM program, slightly modified, has this dual.
- A bit tedious unfortunately.

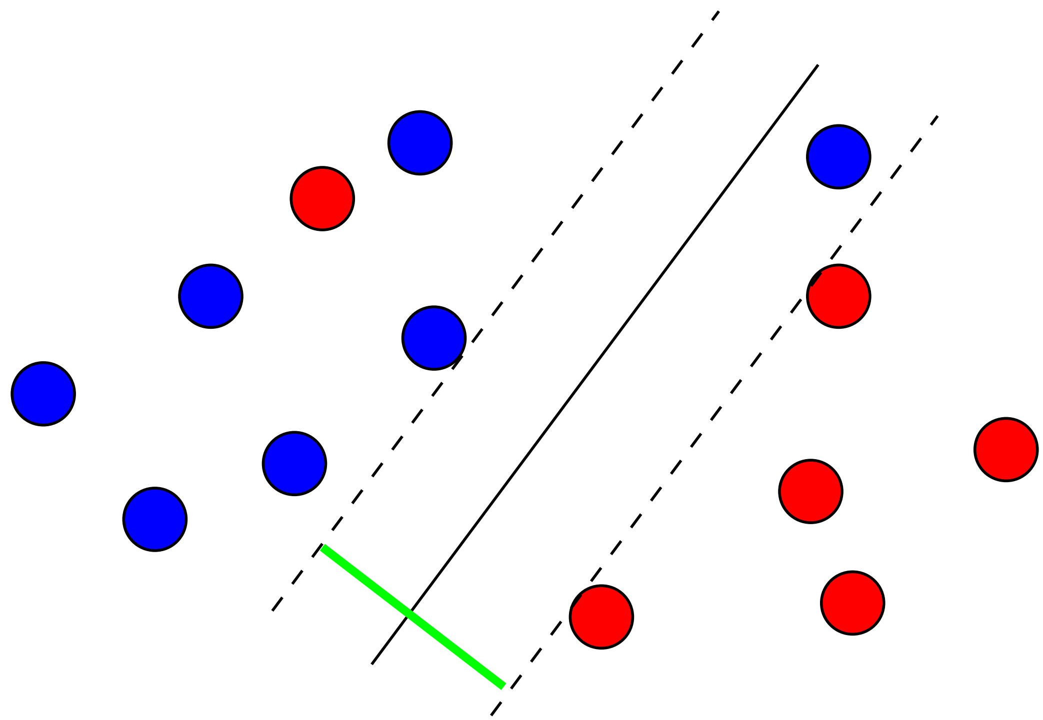
What happens when the data is not linearly separable?



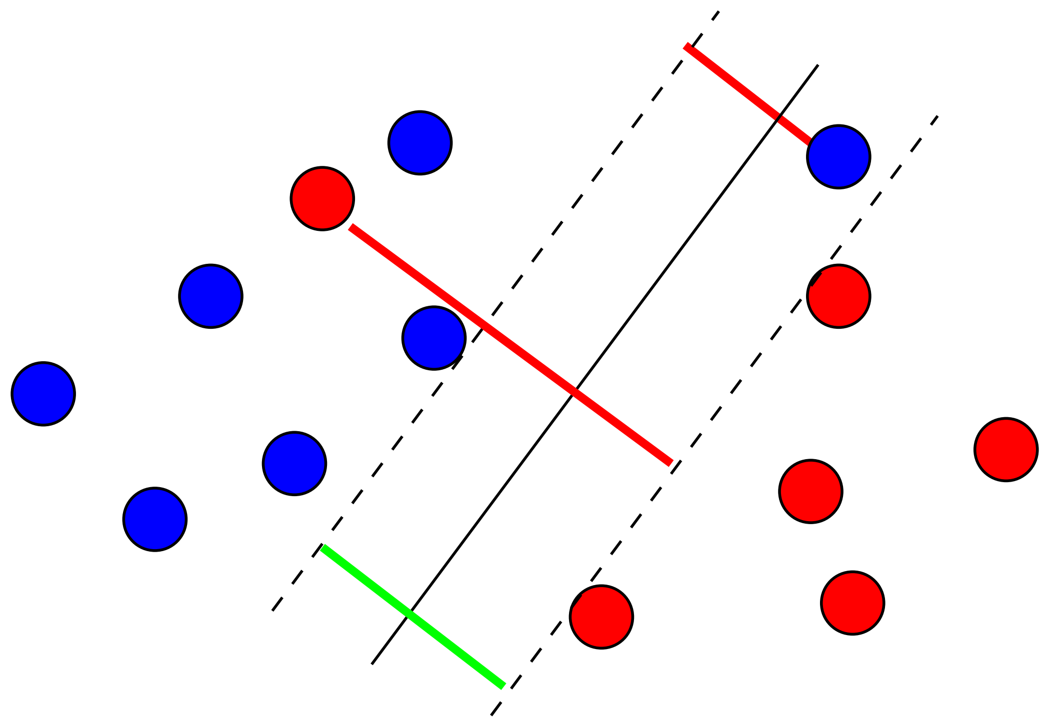
What happens when the data is not linearly separable?



What happens when the data is not linearly separable?



What happens when the data is not linearly separable?

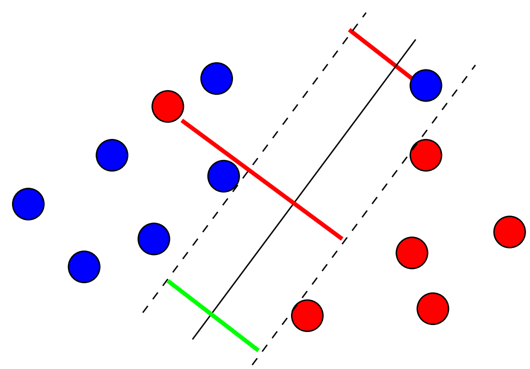


Soft-margin SVM [3]

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

- Mathematically:
$$\min_f \left\{ \frac{1}{\text{margin}(f)} + C \times \text{errors}(f) \right\}$$

- C is a parameter



Soft-margin SVM formulation [3]

- 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

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

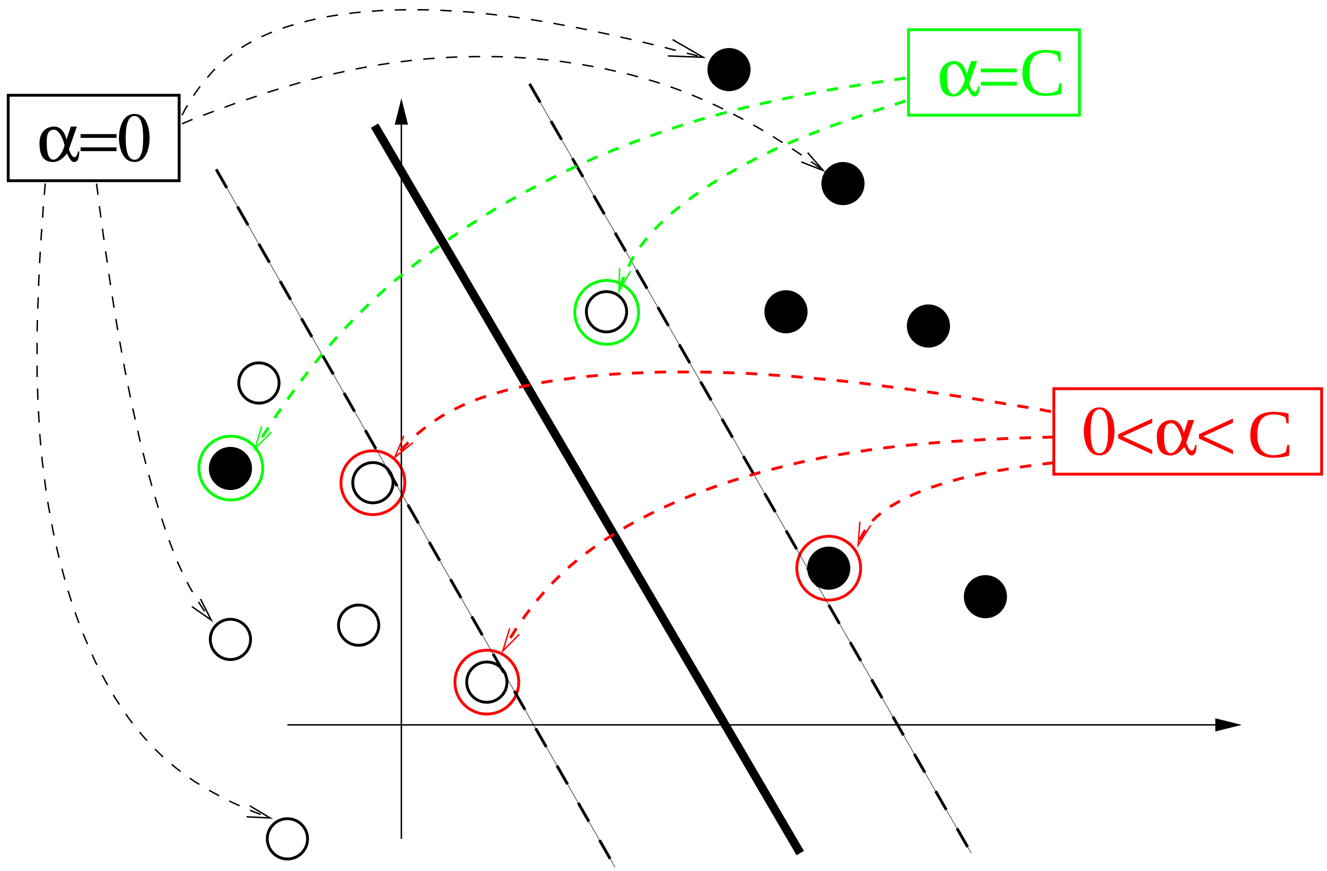
- In that case the dual function

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

which is finite under the constraints:

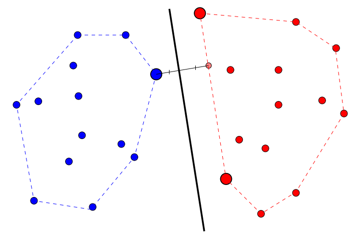
$$\begin{cases} 0 \leq \alpha_i \leq C, & \text{for } i = 1,\dots,n \\ \sum_{i=1}^{n}\alpha_i\mathbf{y}_i = 0. \end{cases}$$

Interpretation: bounded and unbounded support vectors

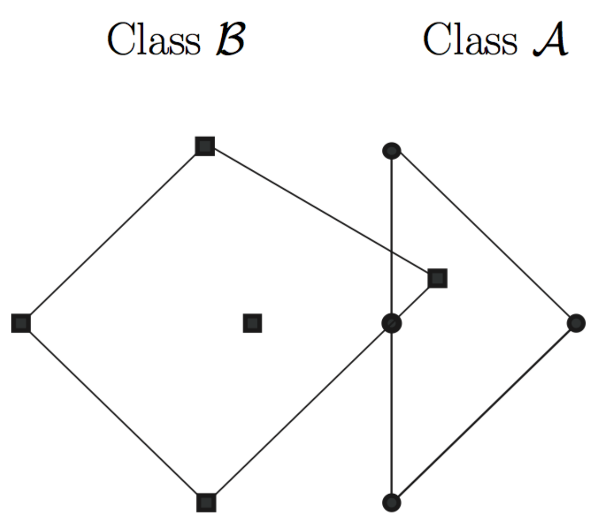


What about the convex hull analogy?

- Remember the separable case



- Here we consider the case where the two sets are not linearly separable, $i.e.$ their convex hulls **intersect.**



What about the convex hull analogy?

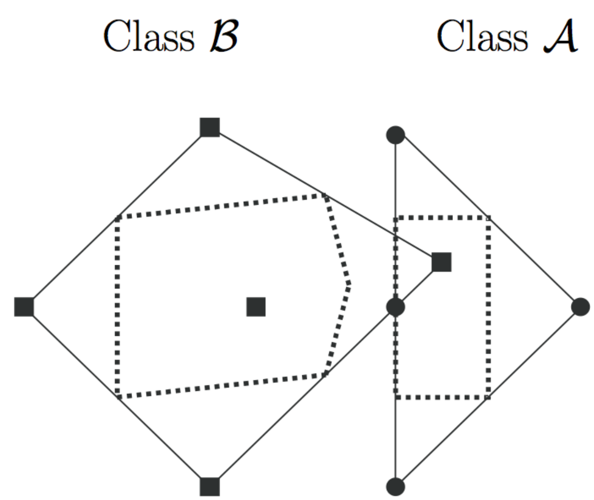
- To follow with the convex hull analogy, consider instead the **reduced** convex hull,

  Definition 1. *Given a set of* n *points* $\mathcal{A}$, *and* $0 \leq C \leq 1$, *the set of finite combinations*

  $$\sum_{i=1}^{n} \lambda_i \mathbf{x}_i, 1 \leq \lambda_i \leq C, \sum_{i=1}^{n} \lambda_i = 1,$$

  is the (C*) reduced convex hull of* $\mathcal{A}$

- Using $C = 1/2$, the reduced convex hulls of $\mathcal{A}$ and $\mathcal{B}$,



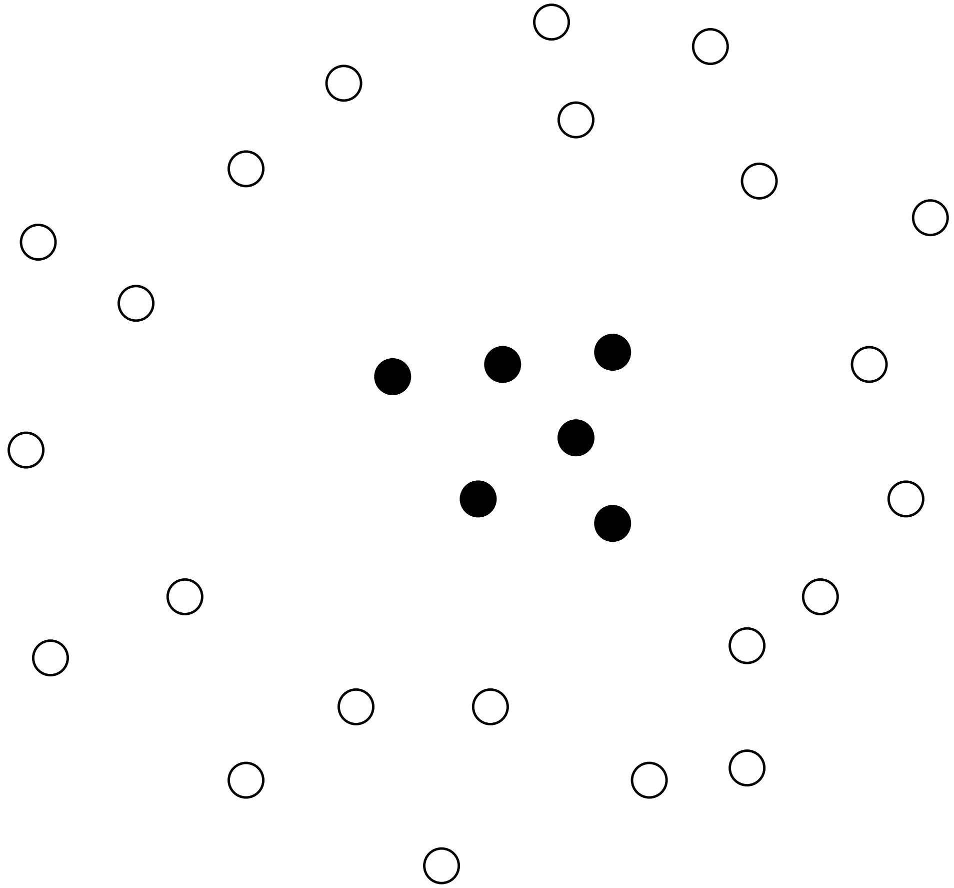
Closest Points in the Reduced Convex Hull

- Idea: find the closest points for the two **reduced** convex hulls

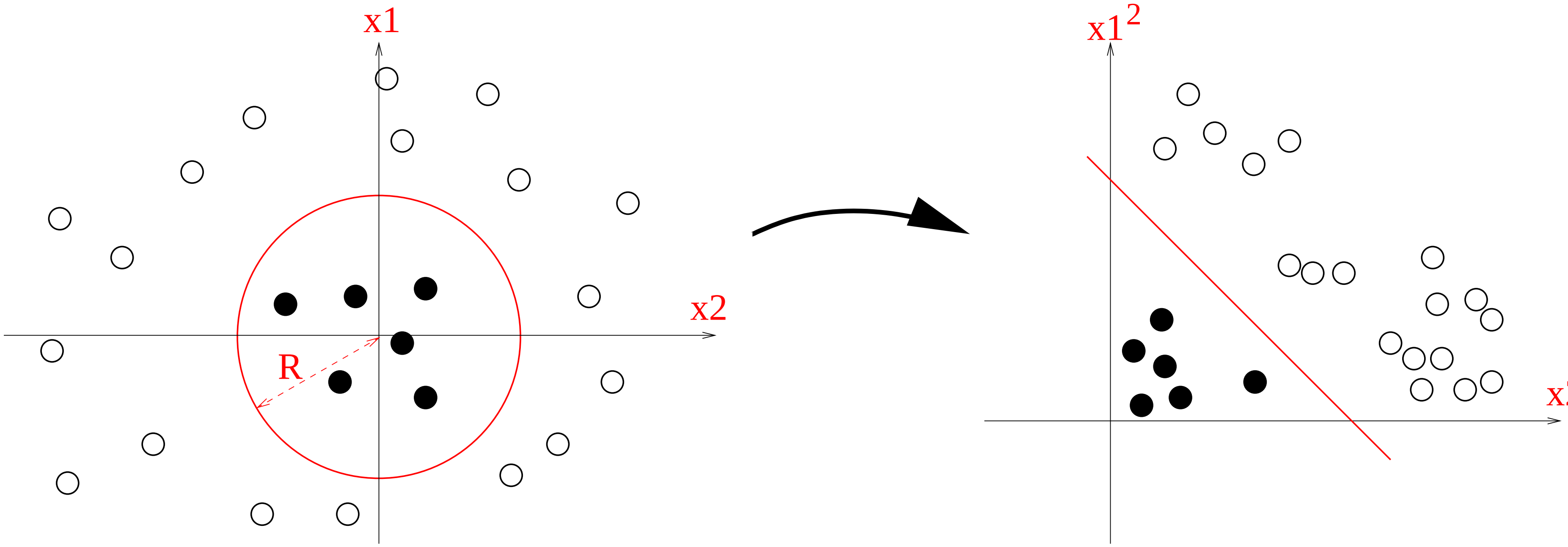
$$
\begin{array}{ll}
\text{minimize} & \|A\mathbf{u} - B\mathbf{v}\|^2 \\
\text{subject to} & \mathbb{1}^T\mathbf{u} = \mathbb{1}^T\mathbf{v} = 1 \\
& \mathbf{u} \leq C\mathbb{1},\ \mathbf{v} \leq C\mathbb{1} \\
& 0 \leq \mathbf{u} \in \mathbb{R}^{n-1}, \mathbf{v} \in \mathbb{R}^{n_1}
\end{array}
$$

- Again, can prove that the soft-margin SVM with C constant accepts as a dual the formulation above.

Sometimes linear classifiers are of little use



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 [3]

- 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 $\langle \phi(\mathbf{x}), \phi(\mathbf{x}') \rangle = k(\mathbf{x}, \mathbf{x}')$

- **Reminder**: 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 [5]

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 $\boldsymbol{K}$ and $\mathbf{y}$:

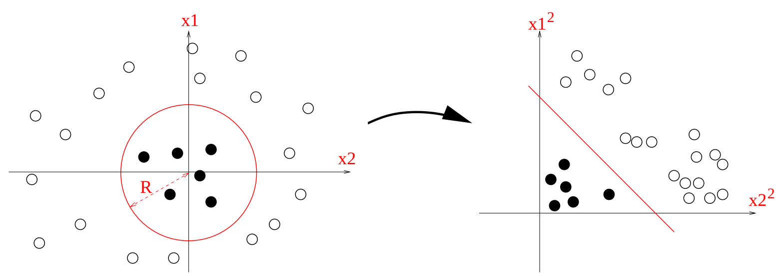
$$\begin{array}{rl} \text{maximize} & g(\alpha) = \alpha^T \mathbf{1} - \frac{1}{2}\alpha^T(\mathbf{y}^T \boldsymbol{K} \mathbf{y})\alpha \\ \text{such that} & 0 \leq \alpha_i \leq C, \quad \text{for } i = 1, \ldots, n \\ & \sum_{i=1}^n \alpha_i \mathbf{y}_i = 0. \end{array}$$

- $\boldsymbol{K}$'s **positive definiten** $\Leftrightarrow$ **problem has an optimum**
- the decision function is $f(\cdot) = \sum_{i=1}^n \alpha_i \boldsymbol{k}(\mathbf{x}_i, \cdot) + b$.

Kernel example: polynomial kernel

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

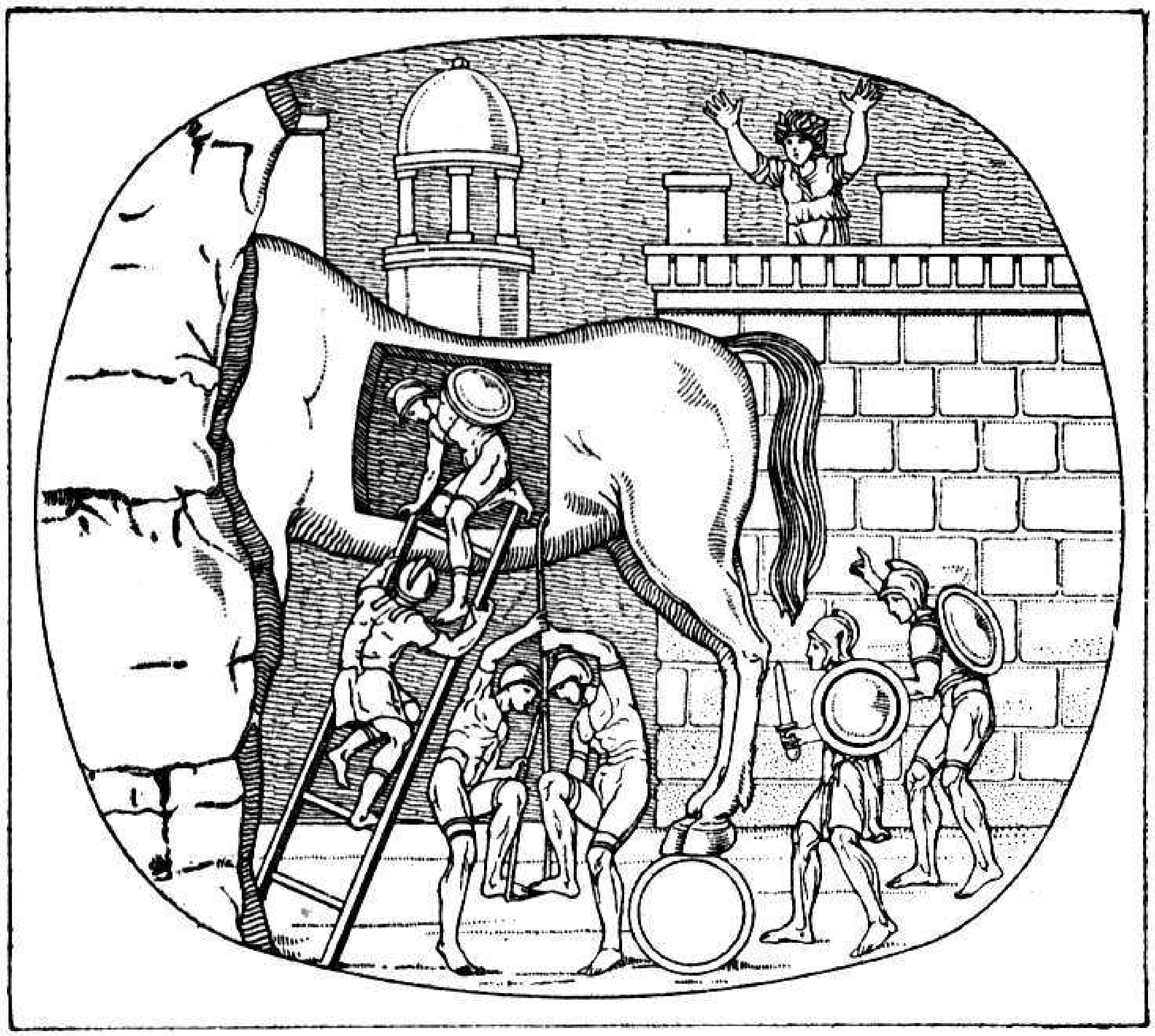
$$\begin{aligned} \boldsymbol{K}(\mathbf{x}, \mathbf{x}') &= x_1^2 x_1'^2 + 2x_1x_2x_1'x_2' + x_2^2x_2'^2 \\ &= \{x_1x_1' + x_2x_2'\}^2 \\ &= \{\mathbf{x}^T\mathbf{x}'\}^2 . \end{aligned}$$



- Many more:

Kernels are Trojan Horses onto Linear Models

- With kernels, complex structures can enter the realm of linear models



References

[1] K.P. Bennett and E.J. Bredensteiner. Duality and Geometry in SVM Classifiers. In *Proceedings of the Seventeenth International Conference on Machine Learning*, page 64. Morgan Kaufmann Publishers Inc., 2000.

[2] B. E. Boser, I. M. Guyon, and V. N. Vapnik. A training algorithm for optimal margin classifiers. In *Proceedings of the 5th annual ACM workshop on Computational Learning Theory*, pages 144–152. ACM Press, 1992.

[3] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine Learning*, 20:273, 1995.

[4] T. Hastie, R. Tibshirani, and J. Friedman. *Elements of Statistical Learning: Data Mining, Inference, and Prediction (2nd edition)*. Springer Verlag, 2009.

[5] Bernhard Schölkopf and Alexander J. Smola. *Learning with Kernels: Support Vector Machines, Regularization , Optimization, and Beyond*. MIT Press, 2002.