Foundations of Machine Learning
École Centrale Paris — Fall 2015

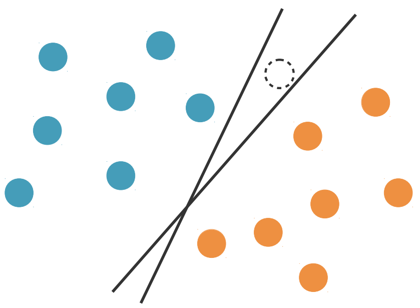
# 9. Support Vector Machines

Chloé-Agathe Azencott
Centre for Computational Biology, Mines ParisTech
chloe-agathe.azencott@mines-paristech.fr

## Learning objectives

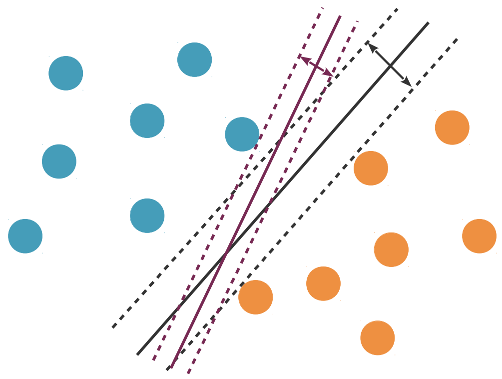
- Define a **large-margin classifier** in the separable case.
- Write the corresponding **primal** and **dual** optimization problems.
- Re-write the optimization problem in the case of **non-separable data.**
- Use the **kernel trick** to apply soft-margin SVMs to **non-linear** cases.
- Define kernels for **real-valued data, strings, and graphs.**

## The linearly separable case: hard-margin SVMs

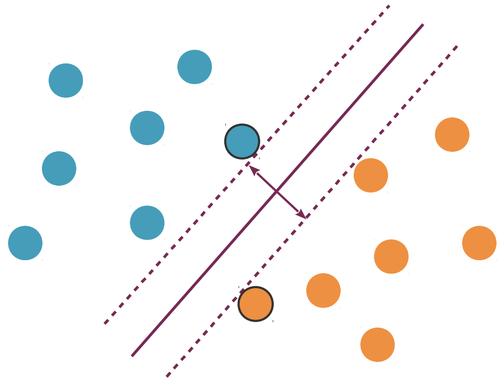


Assume data is linearly separable:
there exists a line that separates + from -

## Margin of a linear classifier



## Largest margin classifier: Support vector machines



## Formalization

- Training set

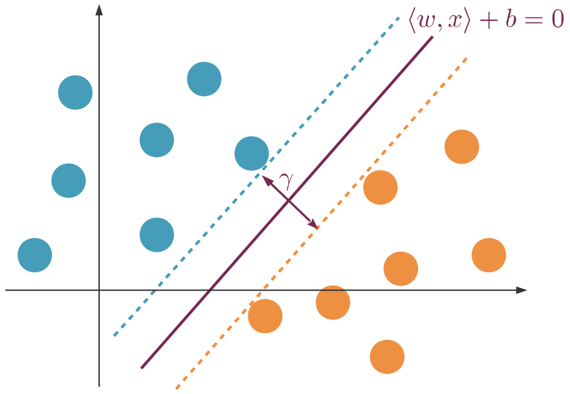
$\mathcal{S} = \{(x^1, y^1), \ldots, (x^n, y^n)\} \qquad x^i \in \mathbb{R}^p \qquad y^i \in \{-1, +1\}$

- Assume the data to be linearly separable

$$\exists (w, b) \in \mathbb{R}^p \times \mathbb{R} \text{ s.t. } \begin{cases} \langle w, x^i \rangle + b > 0 & \text{if } y^i = +1 \\ \langle w, x^i \rangle + b < 0 & \text{if } y^i = -1 \end{cases}$$

- Goal: Find (w*, b*) that define the hyperplane with largest margin.

# Largest margin hyperplane




**What is the size of the margin γ?**

# Optimization problem

- **Margin maximization:**
  minimize $||w||^2$
- **Correct classification of the training points:**
  - For negative examples:
    $$y^i = 1 \text{ and } \langle w, x^i\rangle + b \geq 1$$
  - For positive examples:
    $$y^i = -1 \text{ and } \langle w, x^i\rangle + b \leq -1$$
  - Summarized as:
    $$y^i.(\langle w, x^i\rangle + b) \geq 1$$

This is a classic quadratic optimization problem.

# Karush-Kuhn-Tucker conditions

- **minimize f(w) under the constraint g(w) ≥ 0**

  $f(w) = ||w||^2$ $\quad\quad g(w) = y(\langle w, x\rangle + b) - 1$

  **Case 1:** the unconstraind minimum lies in the feasible region.

  $$\nabla_w f(w) = 0 \text{ and } g(w) \geq 0$$

  **Case 2:** it does not.

  $$\nabla_w f(w) = \alpha \nabla g(w) \text{ and } g(w) = 0, \alpha > 0$$

  - **Summarized as:**

  $$\begin{cases} \nabla_w (f(w) - \alpha g(w)) = 0 \\ \alpha g(w) = 0 \end{cases} \text{ and } \alpha \geq 0.$$

## Karush-Kuhn-Tucker conditions

- **minimize f(w) under the constraint g(w) ≥ 0**

$f(w) = ||w||^2 \qquad g(w) = y(\langle w, x \rangle + b) - 1$

$$\begin{cases} \nabla_w (f(w) - \alpha g(w)) = 0 \\ \alpha g(w) = 0 \end{cases} \quad \text{and } \alpha \geq 0.$$

**Lagrangian:** $L(w, \alpha) = f(w) - \alpha g(w)$

α is called the **Lagrange multiplier.**

## Karush-Kuhn-Tucker conditions

- **minimize f(w) under the constraints $g_i$(w) ≥ 0**

$f(w) = ||w||^2 \qquad g_i(w) = y^i(\langle w, x^i \rangle + b) - 1_{i=1,\dots,n}$

$$\begin{cases} \nabla_w (f(w) - \alpha_i g_i(w)) = 0 \\ \alpha_i g_i(w) = 0 \end{cases} \quad \text{and } \alpha_i \geq 0.$$

**Use n Lagrange multipliers**

– **Lagrangian:**

$$L(w, \alpha) = f(w) - \sum_{i=1}^{n} \alpha_i g_i(w)$$

## Duality

- **Lagrangian** $L(w, \alpha) = f(w) - \sum_{i=1}^{n} \alpha_i g_i(w)$
- **Lagrange dual function** $q : \mathbb{R}^r \to \mathbb{R}$

$$q(\alpha) = \inf_{x \in \mathcal{X}} L(x, \alpha)$$

- **q is concave in α** (even if L is not convex)
- The dual function yields lower bounds on the optimal value of the primal problem when $\alpha \in \mathbb{R}^r_+$

$$q(\alpha) \leq f^* \; \forall \alpha \in \mathbb{R}^r_+$$

## Duality

- **Primal problem:** minimize f s.t. g(x) ≤ 0.
- **Lagrange dual problem:** maximize q.
- **Weak duality:**

  If f* optimizes the primal and d* optimizes the dual,
  then d* ≤ f*.
  Always hold.
- **Strong duality:** f* = d*

  Holds under specific conditions (constraint qualification),
  e.g. Slater's: **f convex and h affine.**

## Back to hard-margin SVMs

- Minimize $||w||^2$
  under the n constraints $y^i(\langle w, x^i \rangle + b) - 1 \geq 0$

- We introduce one **dual variable** $\alpha_i$ for each constraint (i.e. each training point)
- **Lagrangian:**

$$L(w, b, \alpha) = \frac{1}{2}||w||^2 - \sum_{i=1}^{n} \alpha^i \left( y^i(\langle w, x^i \rangle + b) - 1 \right).$$

$$w \in \mathbb{R}^p \quad \alpha \in \mathbb{R}^n_+ \quad b \in \mathbb{R}$$

## Lagrangian of the SVM

$$L(w, b, \alpha) = \frac{1}{2}||w||^2 - \sum_{i=1}^{n} \alpha^i \left( y^i(\langle w, x^i \rangle + b) - 1 \right).$$

- L(w, b, α) is **convex quadratic in w** and minimized for:

$$\nabla_w L = w - \sum_{i=1}^{n} \alpha_i y^i x^i = 0 \Rightarrow w = \sum_{i=1}^{n} \alpha_i y^i x^i.$$

- L(w, b, α) is **affine in b**. It minimum is - ∞ except if:

$$\nabla_b L = \sum_{i=1}^{n} \alpha_i y^i = 0$$

# SVM dual problem

- **Lagrange dual function:**

$$
\begin{aligned}
q(\alpha) &= \inf_{w\in\mathbb{R}^p, b\in\mathbb{R}} L(w,b,\alpha) \\
&= \begin{cases} \sum_{i=1}^n \alpha_i - \frac{1}{2}\sum_{i=1}^n \sum_{j=1}^n y^i y^j \alpha_i \alpha_j \langle x^i, x^j\rangle & \text{if } \sum_{i=1}^n \alpha_i y^i = 0 \\ -\infty & \text{otherwise.} \end{cases}
\end{aligned}
$$

- **Dual problem:**

  maximize q(α)

  subject to α ≥ 0.

- Maximizing a quadratic function under box constraints can be solved efficiently using dedicated software.

# Optimal hyperplane

- Once the optimal α* is found, we recover (w*, b*)

$$w^* = \sum_{i=1}^n \alpha_i^* y^i x^i$$

- The **decision function** is hence:

$$
\begin{aligned}
f^*(x) &= \langle w^*, x\rangle + b^* \\
&= \sum_{i=1}^n \alpha_i y^i \langle x^i, x\rangle + b
\end{aligned}
$$

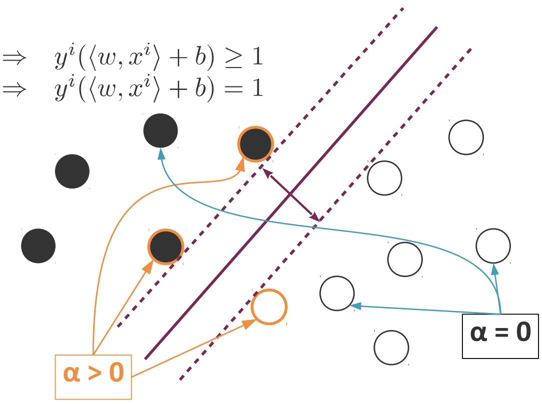
- **KKT conditions:**

  Either $\alpha_i = 0$ or $g_i=0$ $\quad \begin{cases} \nabla_w (f(w) - \alpha_i g_i(w)) = 0 \\ \alpha_i g_i(w) = 0 \end{cases}$ and $\alpha_i \geq 0.$

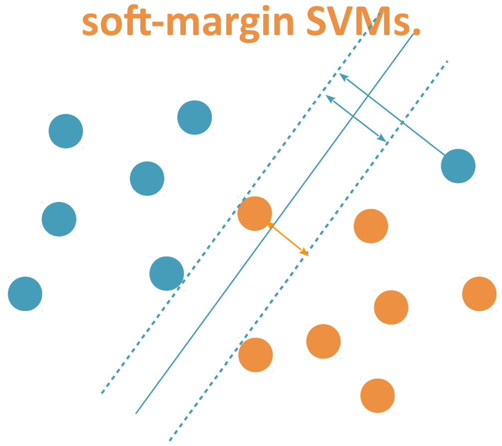
$$
\begin{aligned}
\alpha_i = 0 &\Rightarrow y^i(\langle w, x^i\rangle + b) \geq 1 \\
\alpha_i > 0 &\Rightarrow y^i(\langle w, x^i\rangle + b) = 1
\end{aligned}
$$

# Support vectors

$$
\begin{aligned}
\alpha_i = 0 &\Rightarrow y^i(\langle w, x^i\rangle + b) \geq 1 \\
\alpha_i > 0 &\Rightarrow y^i(\langle w, x^i\rangle + b) = 1
\end{aligned}
$$



## The non-linearly separable case: soft-margin SVMs.



## Soft-margin SVMs

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

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

- **Error:**

$$\begin{cases} 0 & \text{if } y(\langle w, x \rangle + b) \geq 1 \\ 1 - y(\langle w, x \rangle + b) & \text{otherwise.} \end{cases}$$

- The **soft-margin SVM** solves:

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

## The C parameter

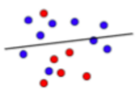
$$\min_f \left( \frac{1}{\text{margin}(f)} + C \times \ \text{error}(f) \right)$$

- **Large C**

  makes few errors

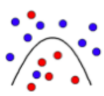
- **Small C**

  ensures a large margin

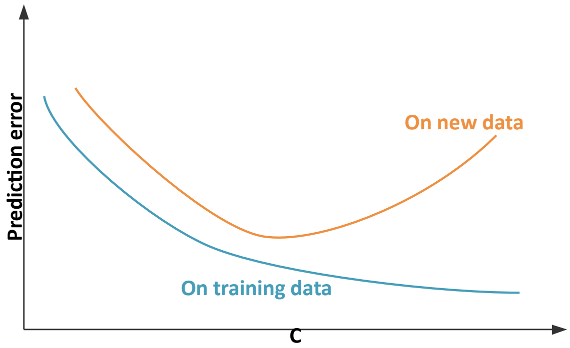


- **Intermediate C**

  finds a tradeoff



# It is important to control C

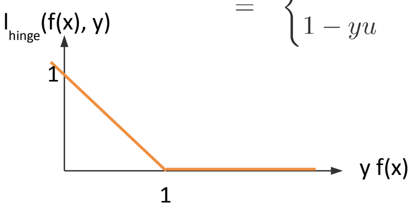



# Hinge loss

$$\arg\min_{w,b}\left(\sum_{i=1}^{n} l_{\text{hinge}}(\langle w, x^i\rangle + b, y^i) + \lambda ||w||^2\right)$$

- λ = 1/C
- **Hinge loss** function:

$$\begin{aligned} l_{\text{hinge}}(u,y) &= \max(1-yu, 0) \\ &= \begin{cases} 0 & \text{if } yu \geq 1 \\ 1-yu & \text{otherwise} \end{cases} \end{aligned}$$




# Slack variables

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

is equivalent to:

$$\begin{aligned} \arg\min \quad & ||w||^2 + C\textstyle\sum_{i=1}^{n}\xi_i \\ \text{s. t. } y^i(\langle w, x^i\rangle + b) &\geq 1 - \xi_i \\ \xi_i &\geq 0 \;\; \forall i \end{aligned}$$

slack variable

## Dual formulation of the soft-margin SVM

- Maximize

$$L(\alpha) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y^i y^j x^i x^j$$

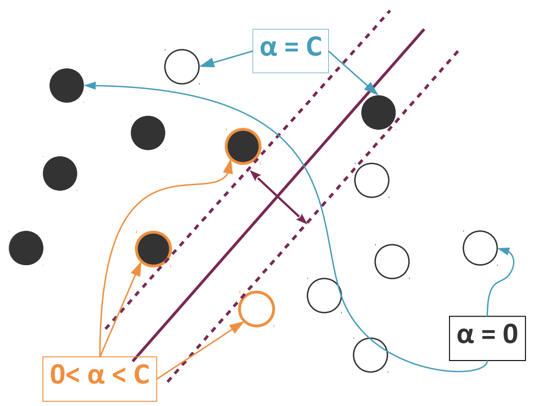
- under the constraints

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

- **KKT conditions:**

$$\begin{array}{rcll} \alpha_i = 0 & \Rightarrow & y^i(\langle w, x^i \rangle + b) \geq 1 & \text{“easy”} \\ \alpha_i = C & \Rightarrow & y^i(\langle w, x^i \rangle + b) \leq 1 & \text{“hard”} \\ 0 < \alpha_i < C & \Rightarrow & y^i(\langle w, x^i \rangle + b) = 1 & \text{“somewhat hard”} \end{array}$$

## Support vectors of the soft-margin SVM




## Primal vs. dual

- Primal: (w, b) has **dimension (p+1).**

$$\arg\min_{w,b} \left( \sum_{i=1}^{n} l_{\text{hinge}}(\langle w, x^i \rangle + b, y^i) + \lambda ||w||^2 \right)$$

  Favored if the data is **low-dimensional.**

- Dual: α has **dimension n.**

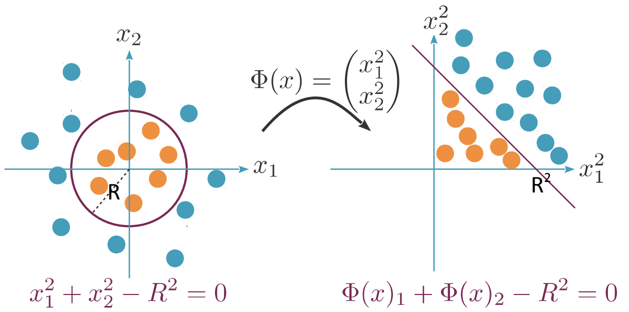
$$\arg\max L(\alpha) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y^i y^j x^i x^j$$

$$0 \leq \alpha_i \leq C \text{ and } \sum_{i=1}^{n} \alpha_i y^i = 0$$

  Favored is there is **little data** available.

# The non-linear case: kernel SVMs.

## Non-linear mapping to a feature space




## Kernels

For a given mapping

$$\Phi : \mathcal{X} \mapsto \mathcal{H}$$

from the space of objects X to some Hilbert space H, the **kernel** between two objects x and x' is the inner product of their images in the feature spaces.

$$\forall x, x' \in \mathcal{X}, K(x, x') = \langle \Phi(x), \Phi(x') \rangle_{\mathcal{H}}$$

$$K(x, x') = \langle \Phi(x), \Phi(x') \rangle_{\mathcal{H}} = x_1^2 {x'_1}^2 + x_2^2 {x'_2}^2$$

**Kernels allow us to formalize the notion of similarity.**

## Kernel tricks

- Many linear algorithms (in particular, linear SVMs) can be performed in the feature space H **without explicitly computing the images ϕ(x)**, but instead by computing kernels K(x, x')
- It is sometimes easy to compute kernels which correspond to large-dimensional feature spaces: **K(x, x') is often much simpler to compute than ϕ(x).**

## SVM in the feature space

- **Train:**

$$\arg\max L(\alpha) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y^i y^j \langle \Phi(x^i), \Phi(x^j) \rangle_{\mathcal{H}}$$

  - under the constraints

$$0 \leq \alpha_i \leq C \text{ and } \sum_{i=1}^{n} \alpha_i y^i = 0$$

- **Predict** with the decision function

$$f(x) = \sum_{i=1}^{n} \alpha_i y^i \langle \Phi(x^i), \Phi(x^j) \rangle_{\mathcal{H}} + b$$

## SVM with a kernel

- **Train:**

$$\arg\max L(\alpha) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y^i y^j K(\Phi(x^i), \Phi(x^j))$$

  - under the constraints

$$0 \leq \alpha_i \leq C \text{ and } \sum_{i=1}^{n} \alpha_i y^i = 0$$

- **Predict** with the decision function

$$f(x) = \sum_{i=1}^{n} \alpha_i y^i K(\Phi(x^i), \Phi(x^j)) + b$$

## Polynomial kernels

For $x = (x_1, x_2) \in \mathbb{R}^2 : \Phi(x) = (x_1^2, \sqrt{2}x_1x_2, x_2^2) \in \mathbb{R}^3$

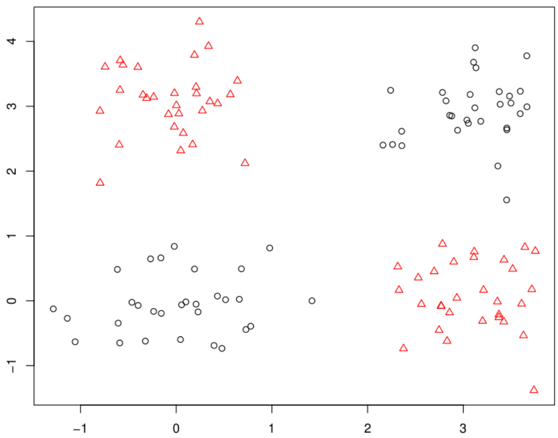
$$\begin{aligned} K(x, x') &= x_1^2 {x_1'}^2 + 2x_1x_2x_1'x_2' + x_2^2 {x_2'}^2 \\ &= \langle x, x' \rangle^2 \end{aligned}$$

More generally, for $\mathcal{X} = \mathbb{R}^p$

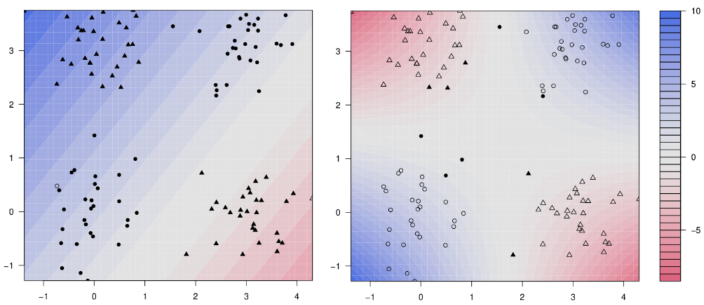
$$K(x, x') = (\langle x, x' \rangle + 1)^d$$

is an inner product in a feature space of all monomials of degree up to d.

## Toy example



## Toy example: linear vs polynomial SVM



## Which functions are kernels?

- A function K(x, x') defined on a set X is a **kernel** iff it exists a Hilbert space H and a mapping ϕ: X →H such that, for any x, x' in X:

$$K(x, x') = \langle \Phi(x), \Phi(x') \rangle_{\mathcal{H}}$$

- A function K(x, x') defined on a set X is **positive definite** iff it is **symmetric** and satisfies:

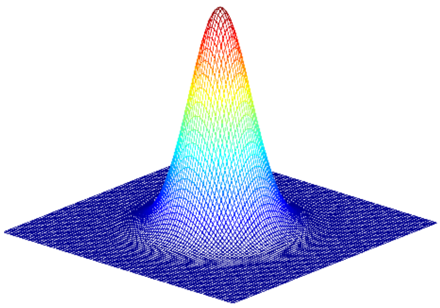
$$\forall N \in \mathbb{N}, \forall (x^1, x^2, \ldots, x^N) \in \mathcal{X}^N \text{ and } (a_1, a_2, \ldots, a_N) \in \mathbb{R}^N$$

$$\sum_{i=1}^{N} \sum_{j=1}^{N} a_i a_j K(x^i, x^j) \geq 0$$

- Theorem [Aronszajn, 1950]: **K is a kernel iff it is positive definite.**

## Gaussian kernel

$$K(x, x') = \exp\left(-\frac{||x - x'||^2}{2\sigma^2}\right)$$



## Kernels for strings

## Protein sequence classification

**Goal:** predict which proteins are secreted or not, based on their sequence.

- Secreted proteins:
  MASKATLLLAFTLLFATCIARHQQRQQQQNQCQLQNIEA...
  MARSSLFTFLCLAVFINGCLSQIEQQSPWEFQGSEVW...
  MALHTVLIMLSLLPMLEAQNPEHANITIGEPITNETLGWL...
  ...
- Non-secreted proteins:
  MAPPSVFAEVPQAQPVLVFKLIADFREDPDPRKVNLGVG...
  MAHTLGLTQPNSTEPHKISFTAKEIDVIEWKGDILVVG...
  MSISESYAKEIKTAFRQFTDFPIEGEQFEDFLPIIGNP..
  ...

## Substring-based representations

- Represent strings based on the presence/absence of substrings of fixed length.

$$\Phi(x) = \{\Phi_u(x)\}_{u \in \mathcal{A}^k}$$

  - Number of occurences of u in x: **spectrum kernel** [Leslie et al., 2002].
  - Number of occurences of u in x, up to m mismatches: **mismatch kernel** [Leslie et al., 2004].
  - Number of occcurences of u in x, allowing gaps, with a weight decaying exponentially with the number of gaps: **substring kernel** [Lohdi et al., 2002].

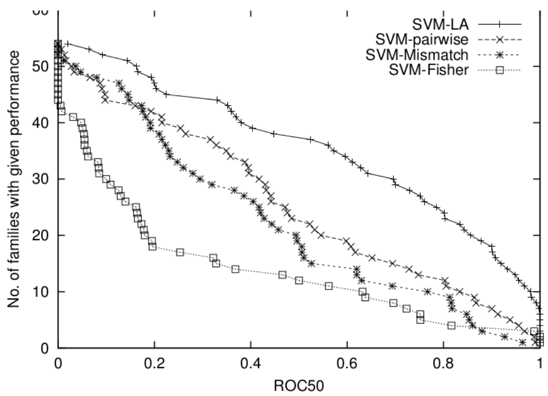
## Spectrum kernel

$$K(x, x') = \sum_{u \in \mathcal{A}^k} \Phi_u(x)\Phi_u(x')$$

- **Implementation:**
  - Formally, a sum over $|A^k|$ terms
  - At most $|x| - k + 1$ non-zero terms in $\Phi(x)$
  - Hence: Computation in $O(|x|+|x'|)$
- **Fast prediction** for a new sequence x:

$$\begin{aligned} f(x) &= \langle w, \Phi(x) \rangle + b \\ &= \sum_{u \in \mathcal{A}^k} w_u \Phi_u(x) + b \\ &= \sum_{j=1}^{|x|-k+1} w_{x_j x_{j+1} \ldots x_{j+k-1}} + b \end{aligned}$$

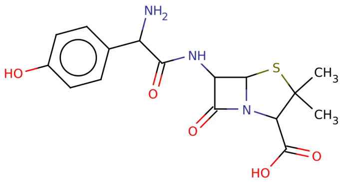
## The choice of kernel matters



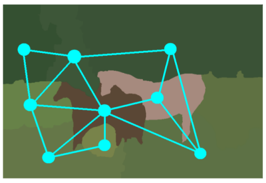

**Performance of several kernels on the SCOP superfamily recognition kernel** [Saigo et al., 2004]

# Kernels for graphs

- **Molecules**




- **Images**



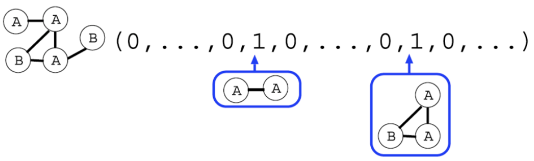
[Harchaoui & Bach, 2007]

# Subgraph-based representations

| | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 1 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 1 |

no occurrence of the 1st feature

1+ occurrences of the 10th feature




# Tanimoto & MinMax

- The Tanimoto and MinMax similarities are kernels

$$s(x^1, x^2) = \frac{\sum_{j=1}^{p} (x_j^1 \text{ AND } x_j^2)}{\sum_{j=1}^{p} (x_j^1 \text{ OR } x_j^2)}$$

$$s(x^1, x^2) = \frac{\sum_{j=1}^{p} \min(x_j^1, x_j^2)}{\sum_{j=1}^{p} \max(x_j^1, x_j^2)}$$

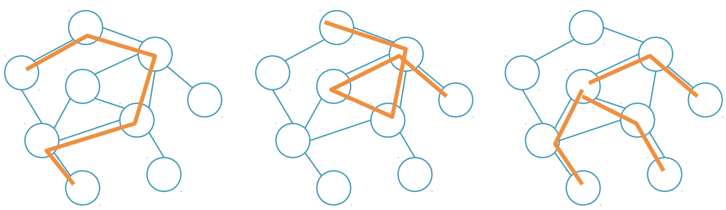
## Which subgraphs to use?

- **Indexing by all subgraphs...**
  - Computing all subgraph occurences is NP-hard.
  - Actually, finding whether a given subgraph occurs in a graph is NP-hard in general.

## Which subgraphs to use?

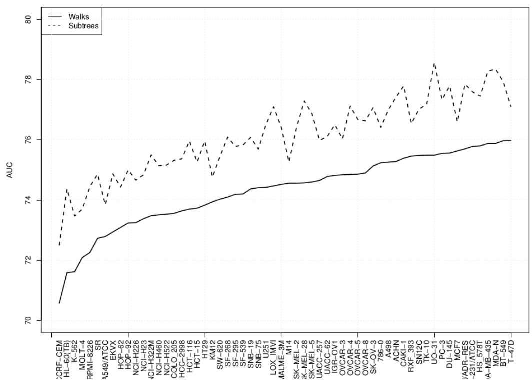
- **Specific subgraphs** that lead to computationally efficient indexing:
  - Subgraphs selected based on **domain knowledge**
    E.g. chemical fingerprints
  - All **frequent subgraphs** [Helma et al., 2004]
  - All **paths** up to length k [Nicholls 2005]
  - All **walks** up to length k [Mahé et al., 2005]
  - All **trees** up to depth k [Rogers, 2004]
  - All **shortest paths** [Borgwardt & Kriegel, 2005]
  - All **subgraphs up to k vertices (graphlets)** [Shervashidze et al., 2009]

## Which subgraphs to use?



Path of length 5 Walk of length 5 Tree of depth 2

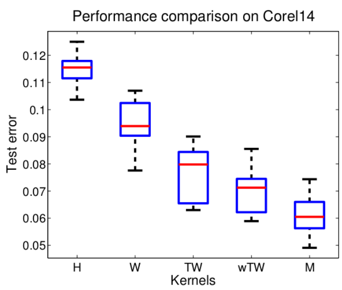
# The choice of kernel matters




Predicting inhibitors for 60 cancer cell lines [Mahé & Vert, 2009]

# The choice of kernel matters

- COREL14: 1400 natural images, 14 classes
- **Kernels:** histogram (H), walk kernel (W), subtree kernel (TW), weighted subtree kernel (wTW), combination (M).




[Harchaoui & Bach, 2007]

# Summary

- Linearly separable case: **hard-margin SVM**
- Non-separable, but still linear: **soft-margin SVM**
- Non-linear: **kernel SVM**
- Kernels for
  - real-valued data
  - strings
  - graphs.