ELEC2870 - Machine learning: regression and dimensionality reduction

Support Vector Machines

Michel Verleysen
Machine Learning Group
Université catholique de Louvain
Louvain-la-Neuve, Belgium
michel.verleysen@uclouvain.be

Outline

- Introduction
- Large-margin classifier
  - Motivation
  - Optimization problem
- Soft margin classifier
  - Motivation
  - Optimization problem
- Mapping to feature space
  - Motivation
  - Dual SVM problem and the kernel trick
  - Kernels and Mercer's condition
- Other kernel methods
- Discussion and conclusions

Outline

- Introduction
- Large-margin classifier
  - Motivation
  - Optimization problem
- Soft margin classifier
  - Motivation
  - Optimization problem
- Mapping to feature space
  - Motivation
  - Dual SVM problem and the kernel trick
  - Kernels and Mercer’s condition
- Other kernel methods
- Discussion and conclusions

Support Vector Machines

- Decision surface is a hyperplane (line in 2-D), as in the perceptron
- Decision surface is the maximal margin hyperplane
- Regularization can handle misclassifications for non-linearly separable problems

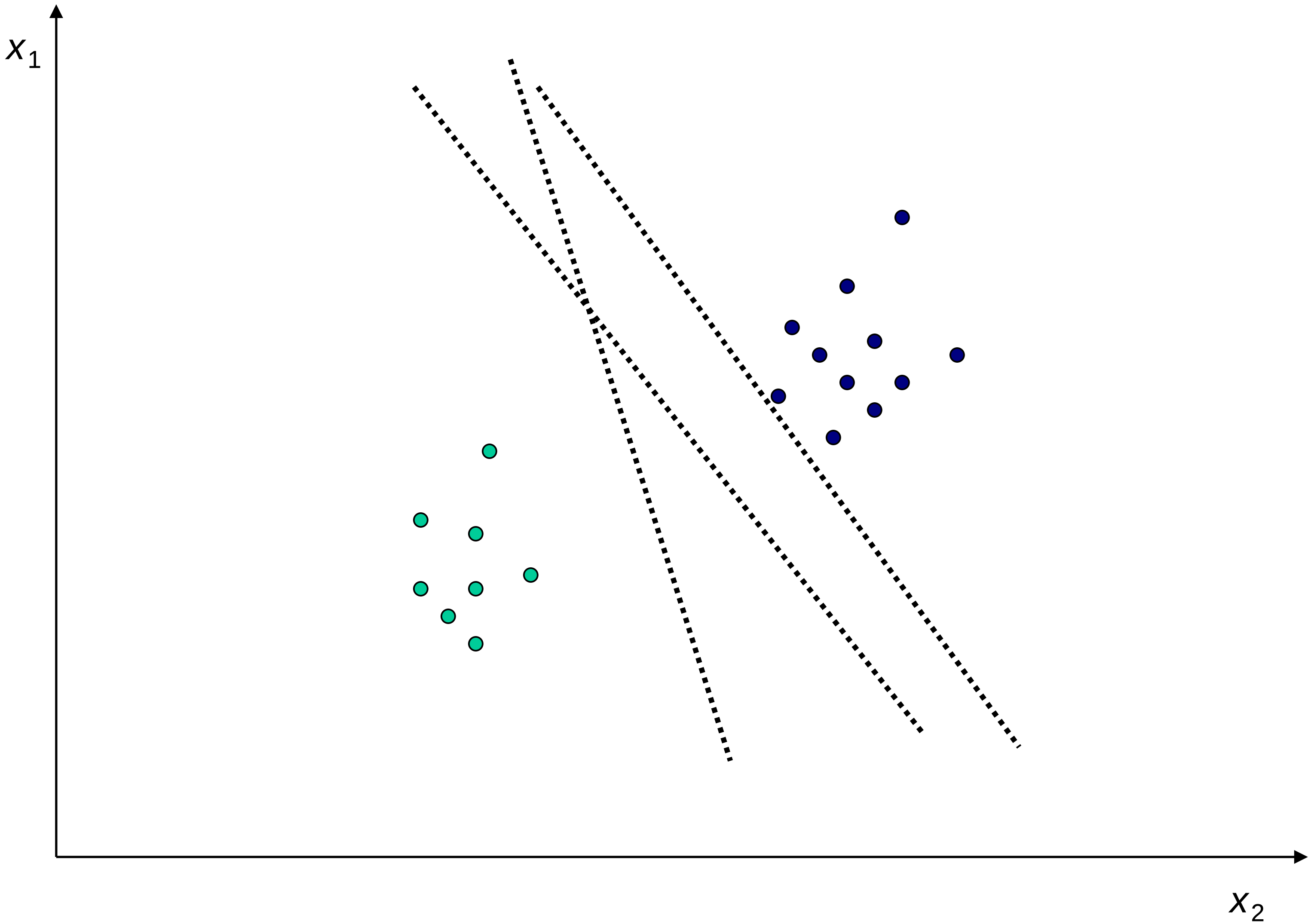
- Decision surface is built in a feature space, not the original data space
- Feature space is built implicitely (not explicitely), thanks to the kernel trick

- Objective function is quadratic
  - $\rightarrow$ single minimum
  - $\rightarrow$ efficient algorithms

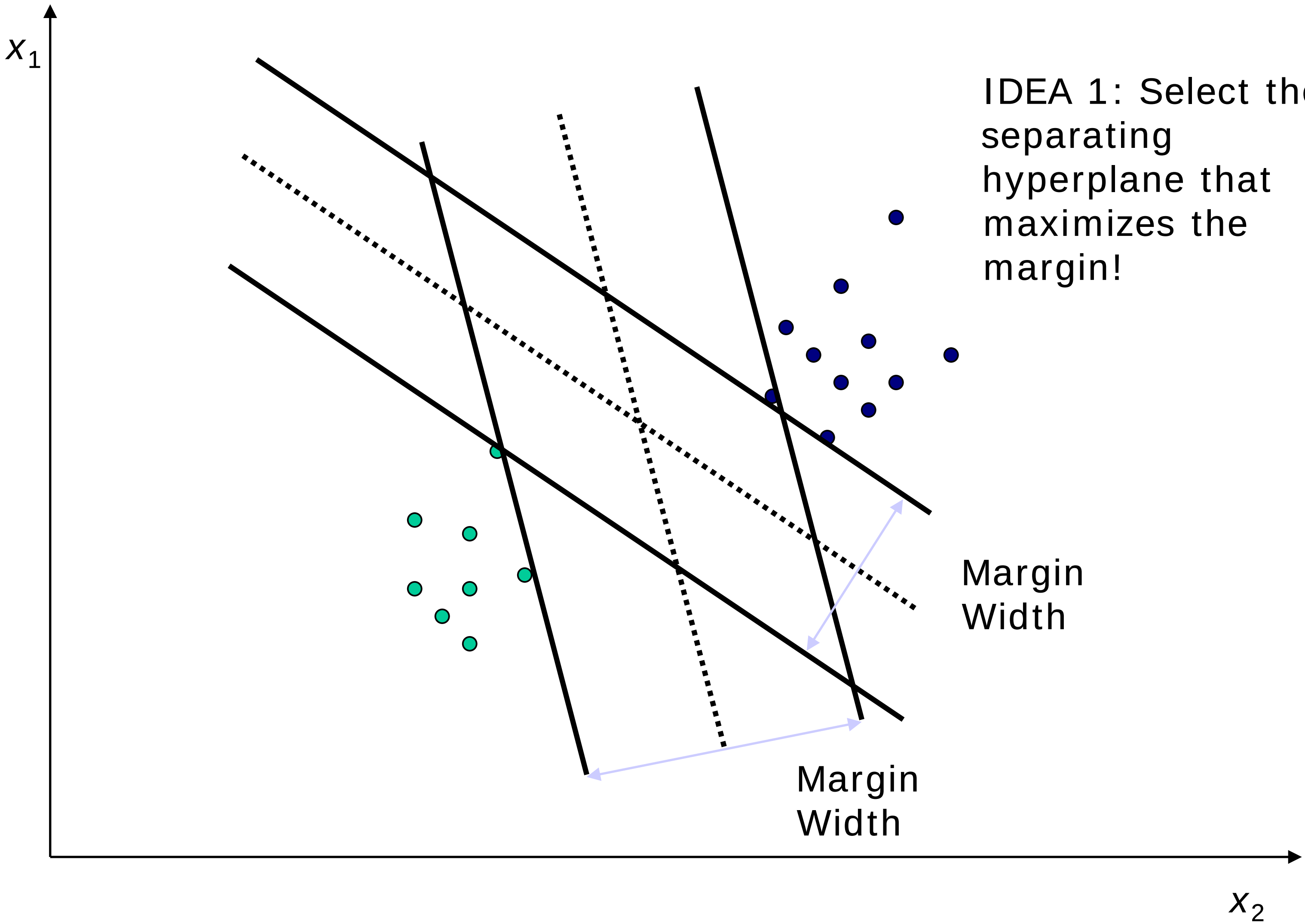
- # parameters is # data, not # dimensions

Outline

Which separating hyperplane to use?

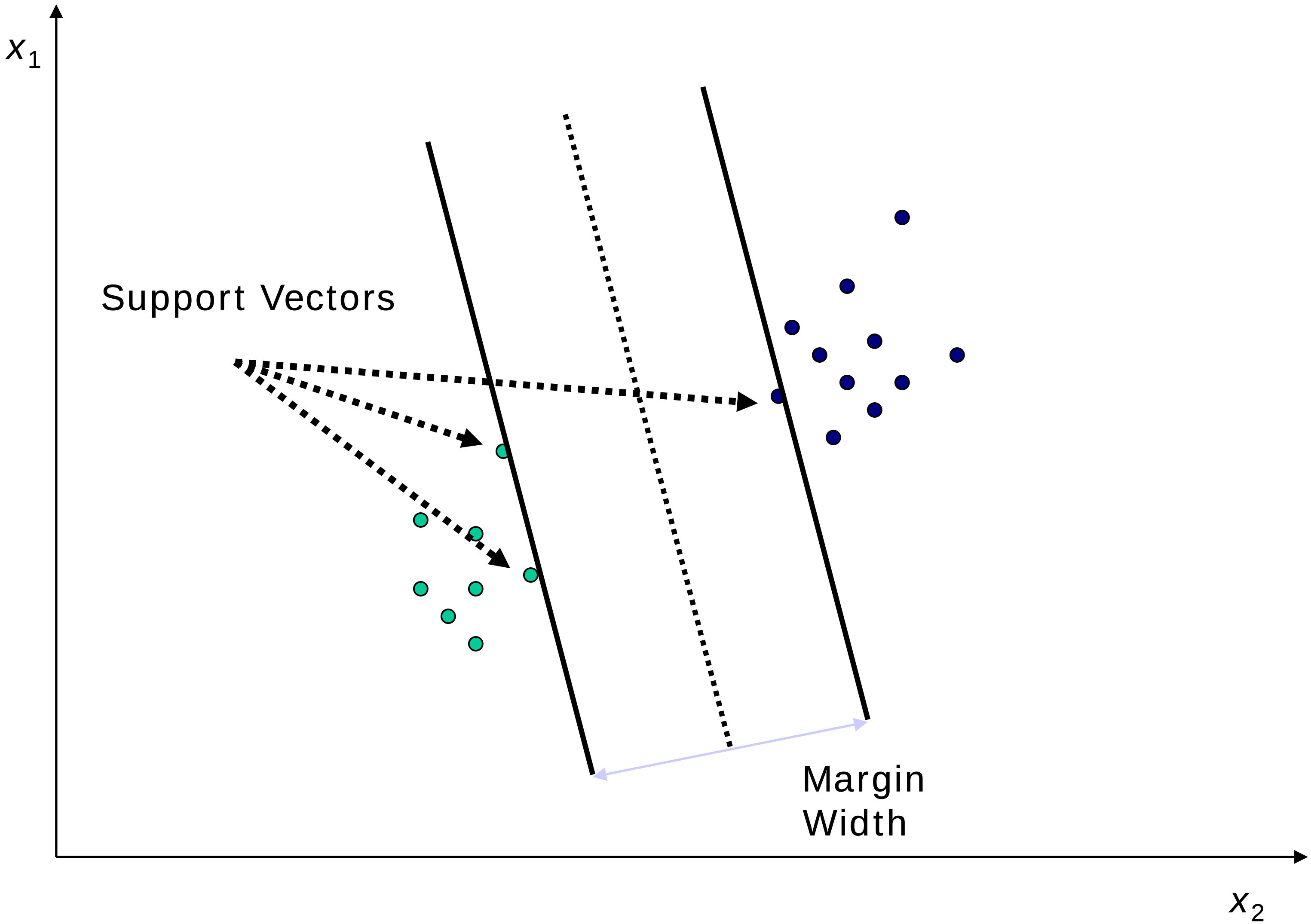


Maximizing the margin

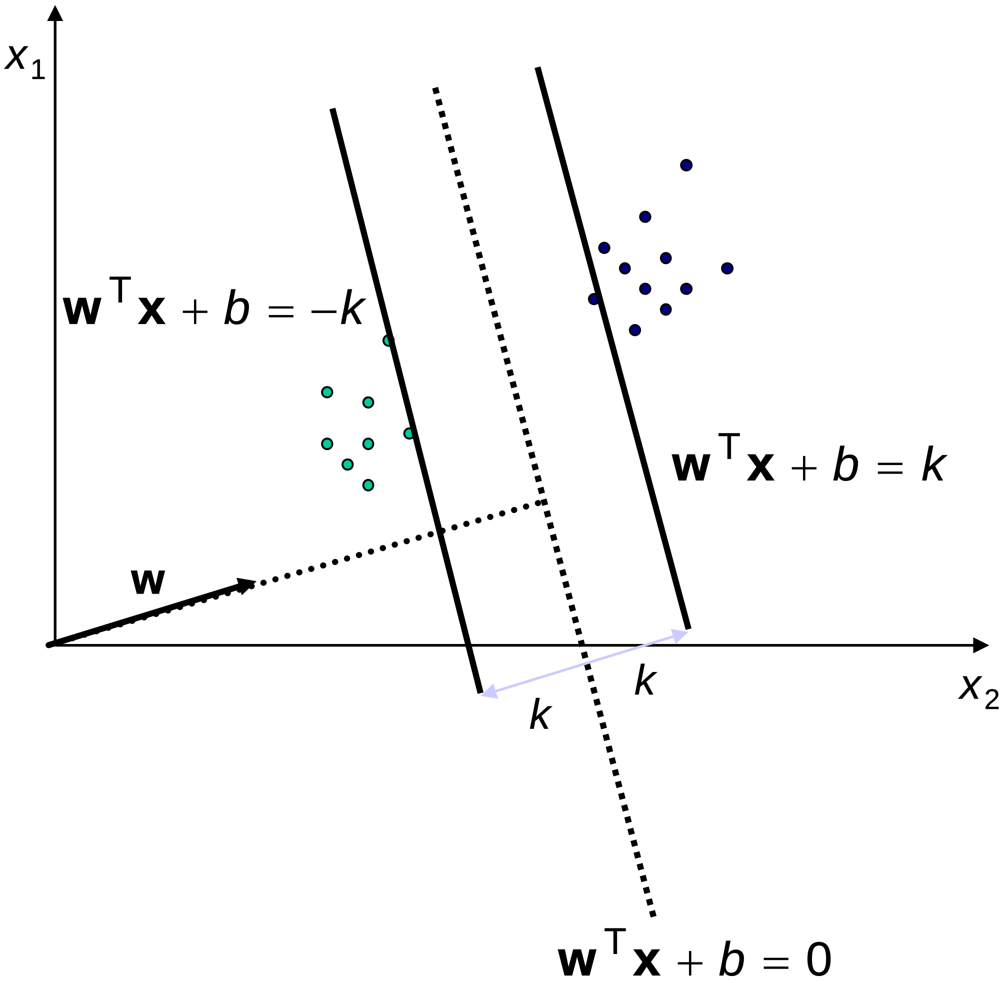


IDEA 1: Select the separating hyperplane that maximizes the margin!

Support vectors



Setting up the optimization problem



- The width of the margin is

$$\frac{2|k|}{\|\mathbf{w}\|}$$

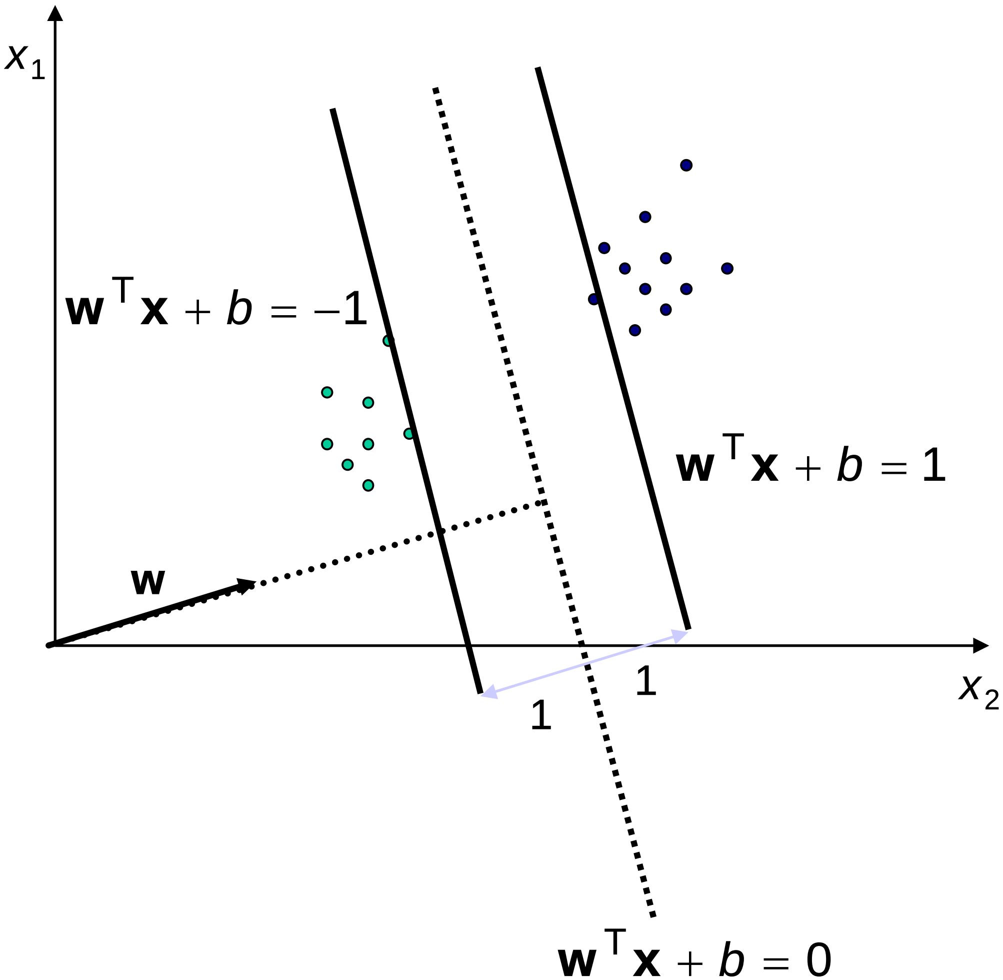
- So the problem is

$$\max_{\mathbf{w},b} \frac{2|k|}{\|\mathbf{w}\|}$$

$$\text{s.t. } \mathbf{w}^T\mathbf{x} + b \geq k, \forall \mathbf{x} \in C^1$$

$$\text{and } \mathbf{w}^T\mathbf{x} + b \leq -k, \forall \mathbf{x} \in C^2$$

Setting up the optimization problem



- There is a scale and unit of data for which $k=1$
- Then the problem becomes

$$\max_{\mathbf{w},b} \frac{2}{\|\mathbf{w}\|}$$

$$\text{s.t. } \mathbf{w}^T\mathbf{x} + b \geq 1, \forall \mathbf{x} \in C^1$$

$$\text{and } \mathbf{w}^T\mathbf{x} + b \leq -1, \forall \mathbf{x} \in C^2$$

Setting up the optimization problem

- If class 1 corresponds to 1 and class 2 corresponds to -1, we can rewrite

$$\mathbf{w}^{\mathsf{T}}\mathbf{x}^{i} + b \geq 1, \forall \mathbf{x}^{i} \text{ with } y^{i} = 1$$

$$\mathbf{w}^{\mathsf{T}}\mathbf{x}^{i} + b \leq -1, \forall \mathbf{x}^{i} \text{ with } y^{i} = -1$$

as

$$y^{i}\left(\mathbf{w}^{\mathsf{T}}\mathbf{x}^{i} + b\right) \geq 1, \forall \mathbf{x}^{i}$$

- Then the problem becomes

$$\max_{\mathbf{w},b} \frac{2}{\|\mathbf{w}\|} \quad \text{s.t. } y^{i}\left(\mathbf{w}^{\mathsf{T}}\mathbf{x}^{i} + b\right) \geq 1, \forall \mathbf{x}^{i}$$

or

$$\min_{\mathbf{w},b} \|\mathbf{w}\|^{2} \quad \text{s.t. } y^{i}\left(\mathbf{w}^{\mathsf{T}}\mathbf{x}^{i} + b\right) \geq 1, \forall \mathbf{x}^{i}$$

Linear, hard-margin SVM formulation

- Find $\mathbf{w}, b$ that solves

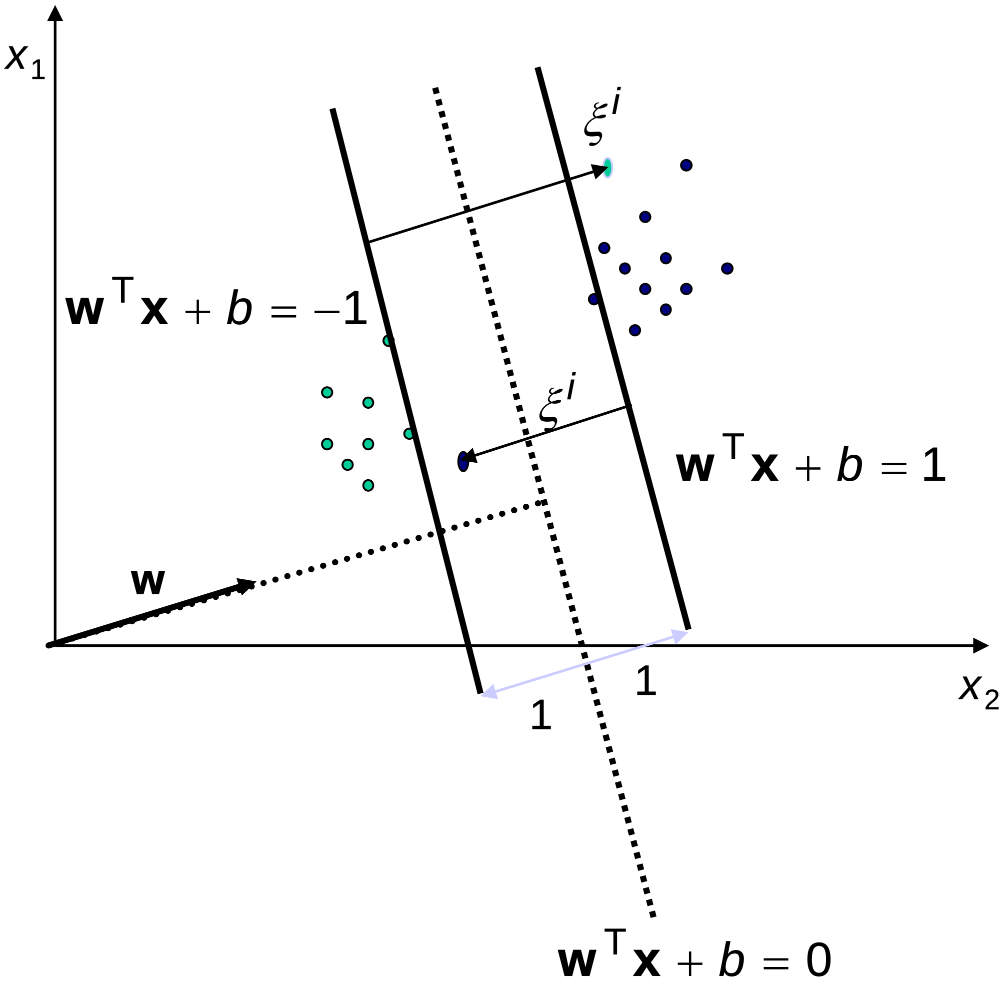
$$\min_{\mathbf{w},b} \|\mathbf{w}\|^2$$
$$\text{s.t. } y^i\left(\mathbf{w}^\mathrm{T}\mathbf{x}^i + b\right) \geq 1, \forall \mathbf{x}^i$$

- Problem is convex so, there is a unique global minimum value (when feasible)
- There is also a unique minimizer, i.e. **w** and *b* value that provides the minimum
- Non-solvable if the data is not linearly separable
- Quadratic Programming
  - Very efficient computationally with modern constraint optimization engines (handles thousands of constraints and training instances)

Outline

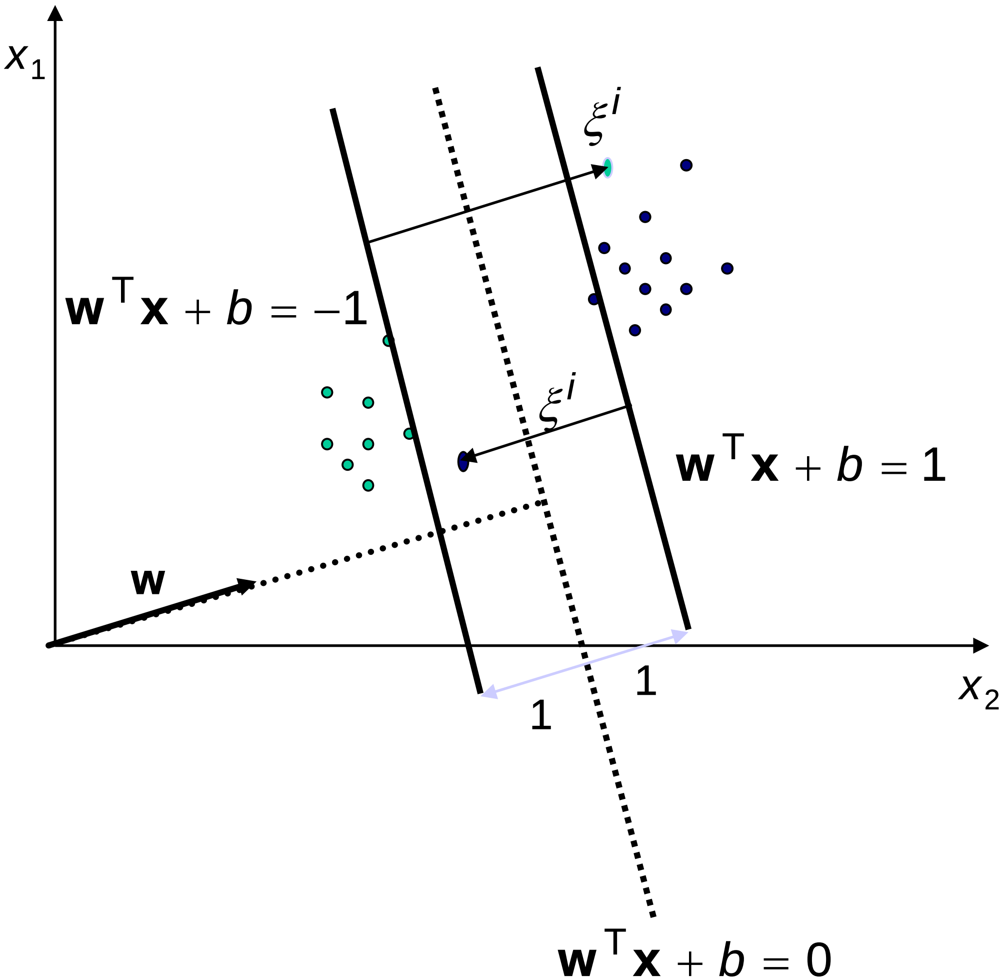
- Introduction
- Large-margin classifier
  - Motivation
  - Optimization problem
- Soft margin classifier
  - Motivation
  - Optimization problem
- Mapping to feature space
  - Motivation
  - Dual SVM problem and the kernel trick
  - Kernels and Mercer's condition
- Other kernel methods
- Discussion and conclusions

Nonlinearly separable data



- Introduce slack variables ξ^i
- Allow some instances to fall within the margin, but penalize them

Formulating the optimization problem



- Constraints become :

$$y^i\left(\mathbf{w}^T\mathbf{x}^i + b\right) \geq 1 - \xi^i, \forall \mathbf{x}^i$$

$$\xi^i \geq 0$$

- Objective function penalizes for misclassified instances and those within the margin

$$\min_{\mathbf{w},b} \|\mathbf{w}\|^2 + C\sum_i \xi^i$$

- C trades-off margin width and misclassifications

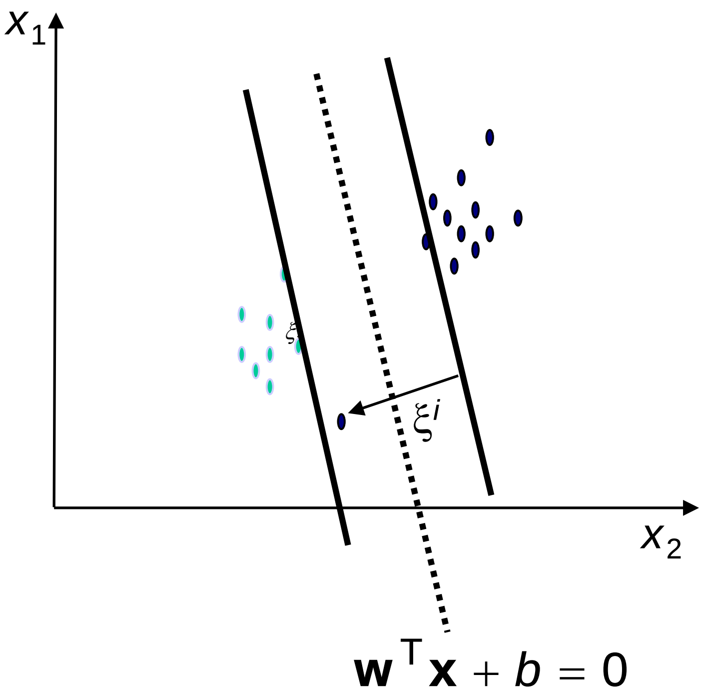
Linear, soft-margin SVMs

$$\min_{\mathbf{w},b} \|\mathbf{w}\|^2 + C\sum_i \xi^i \qquad\qquad y^i\left(\mathbf{w}^{\mathrm{T}}\mathbf{x}^i + b\right) \geq 1 - \xi^i, \forall \mathbf{x}^i$$

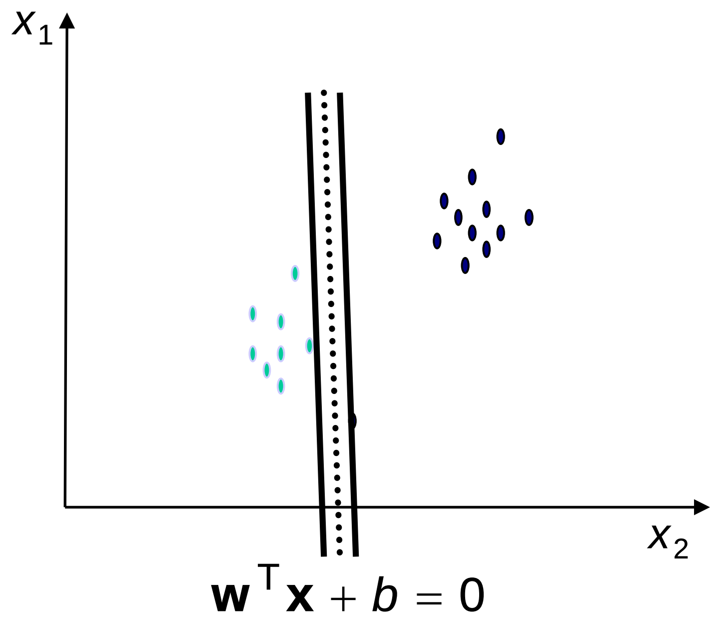
$$\xi^i \geq 0$$

- Algorithm tries to maintain ξ_i to zero while maximizing margin
- Notice: algorithm does not minimize the *number* of misclassifications (NP-complete problem) but the sum of distances from the margin hyperplanes
- Other formulations use ξ_i^2 instead
- As $C \to \infty$, we get closer to the hard-margin solution

Robustness of Soft vs Hard Margin SVMs



Soft Margin SVN



Hard Margin SVN

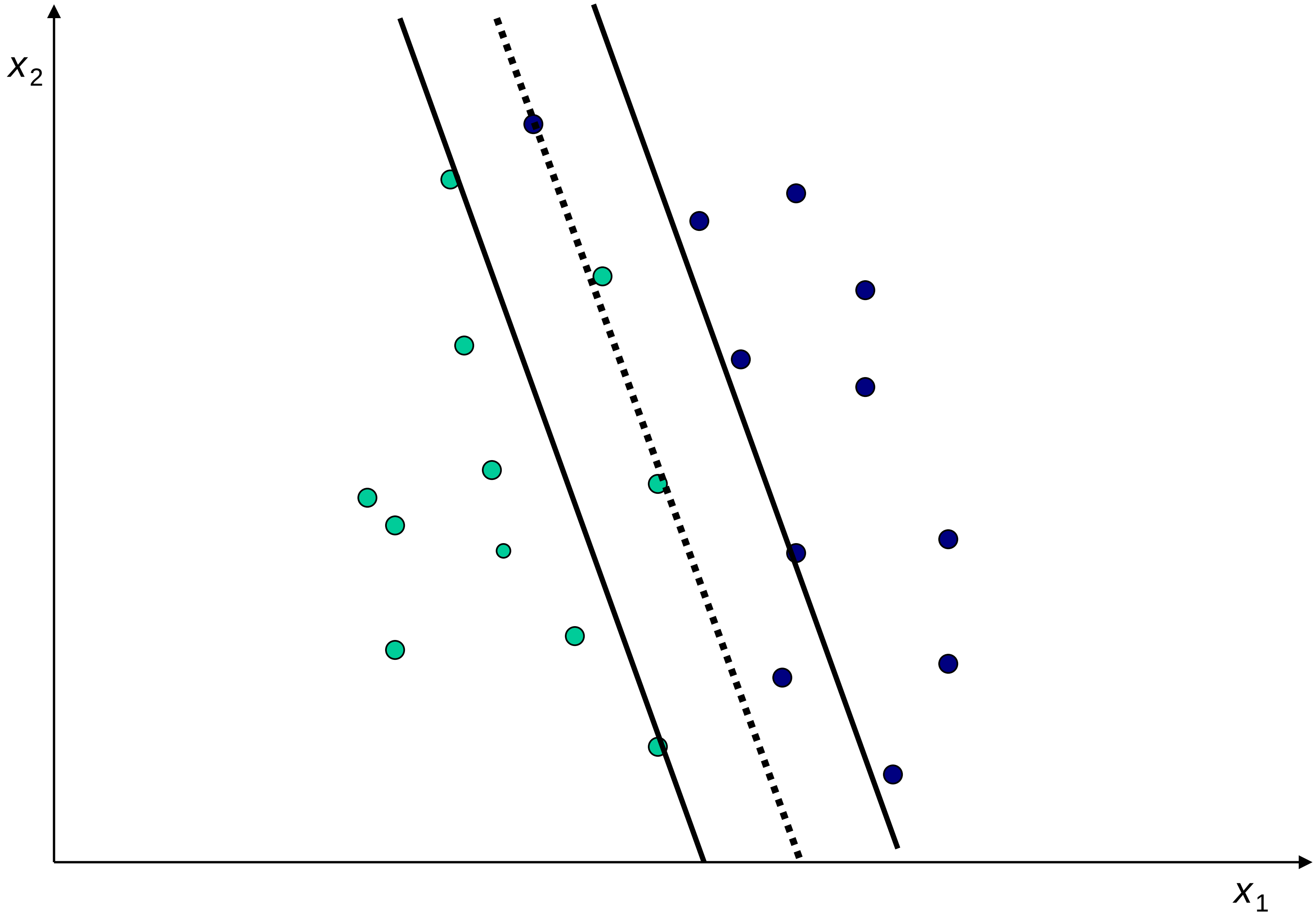
Soft vs Hard Margin SVM

- Soft-Margin always have a solution
- Soft-Margin is more robust to outliers
  - Smoother surfaces (in the non-linear case)
- Hard-Margin does not require to guess the cost parameter (requires no parameters at all)

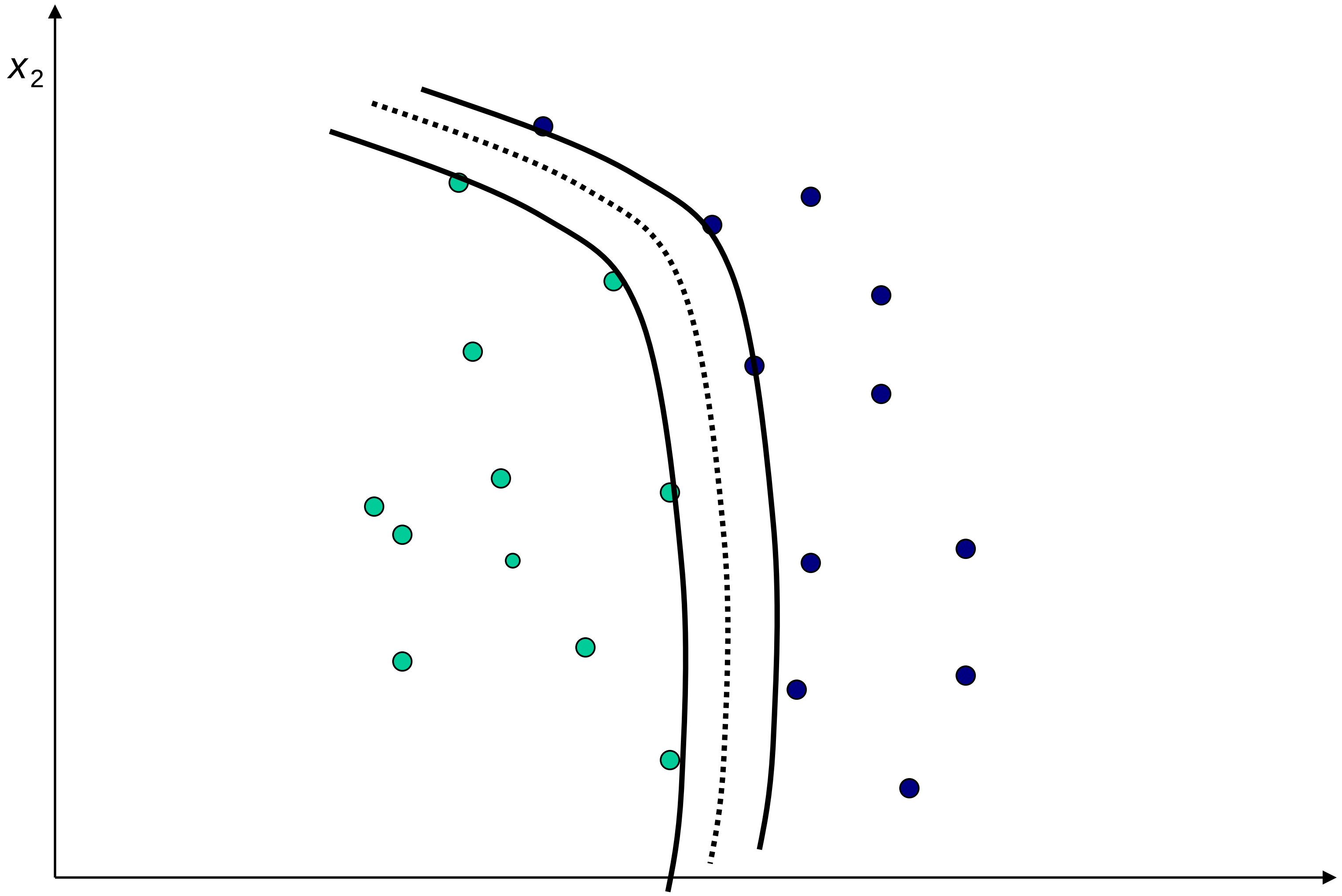
Outline

- Introduction
- Large-margin classifier
  - Motivation
  - Optimization problem
- Soft margin classifier
  - Motivation
  - Optimization problem
- Mapping to feature space
  - Motivation
  - Dual SVM problem and the kernel trick
  - Kernels and Mercer’s condition
- Other kernel methods
- Discussion and conclusions

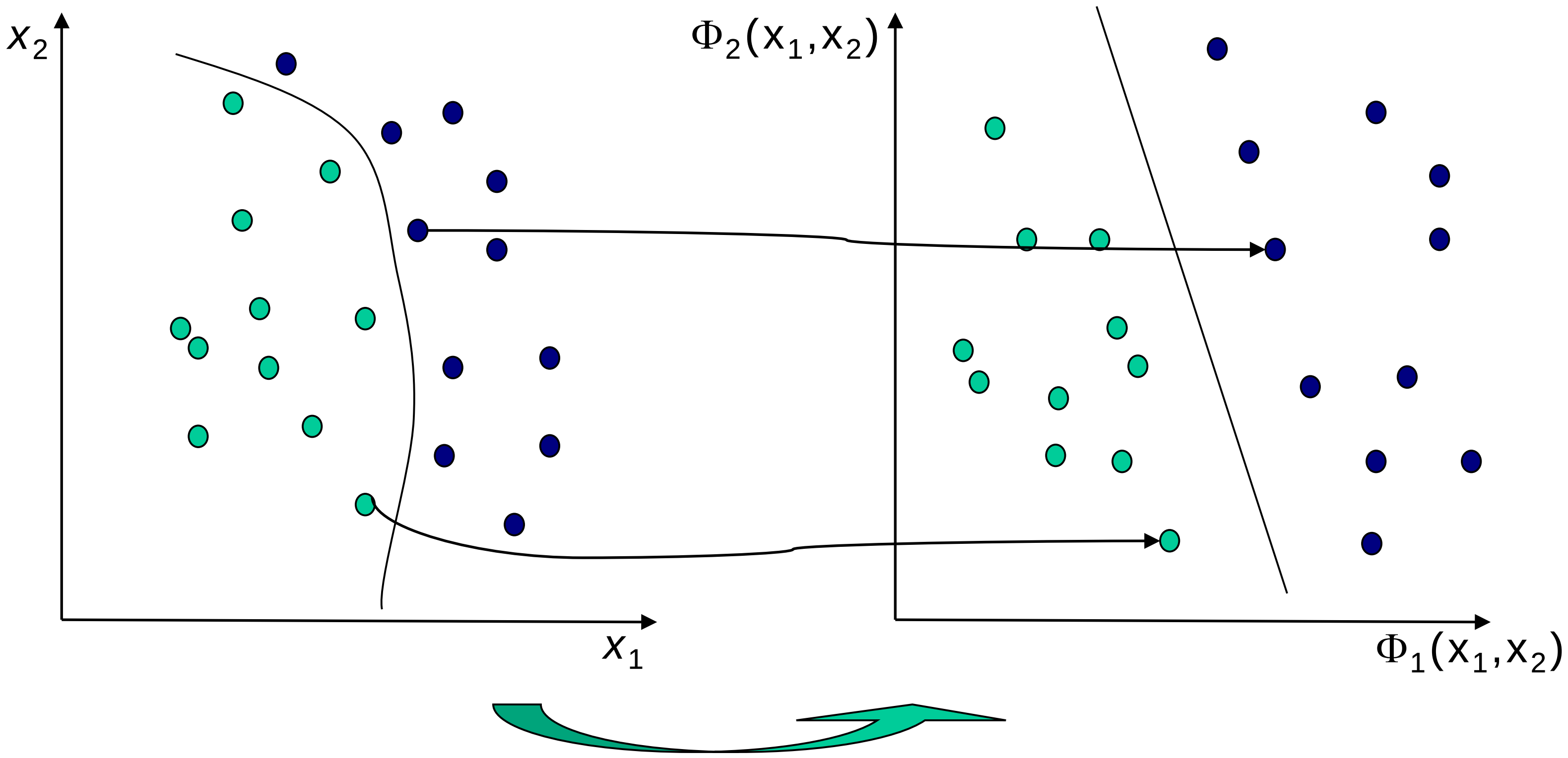
Limitations of linear decision surfaces



Advantages of nonlinear surfaces



Linear classifieers in high-dimensional spaces



Find function $\Phi(\mathbf{x})$ to map
to a different space

Mapping Data to a High-Dimensional Space

- Find function $\Phi(\mathbf{x})$ to map to a different space, then SVM formulation becomes:

$$\min_{\mathbf{w},b} \|\mathbf{w}\|^2 + C\sum_i \xi^i \qquad y^i\left(\mathbf{w}^{\mathrm{T}}\Phi\left(\mathbf{x}^i\right) + b\right) \geq 1 - \xi^i, \forall \mathbf{x}^i$$

$$\xi^i \geq 0$$

- Data appear as $\Phi(\mathbf{x})$, weights $\mathbf{w}$ are now weights in the new space
- Explicit mapping expensive if $\Phi(\mathbf{x})$ is very high dimensional
- Solving the problem without explicitly mapping the data is desirable

The Dual of the SVM Formulation

- Original SVM formulation
  - N inequality constraints
  - *N positivity constraints*
  - N number of ξ variables
  - $D+1$ parameters $\mathbf{w}, b$

$$\min_{\mathbf{w},b} \|\mathbf{w}\|^2 + C\sum_i \xi^i$$

$$y^i\left(\mathbf{w}^{\mathrm{T}}\Phi\left(\mathbf{x}^i\right) + b\right) \geq 1 - \xi^i, \forall \mathbf{x}^i$$

$$\xi^i \geq 0$$

- The (Wolfe) dual of this problem
  - one equality constraint
  - N positivity constraints
  - N number of α variables (Lagrange multipliers)
  - Objective function more complicated

$$\min_{\alpha^i} \frac{1}{2}\sum_{i,j} \alpha^i \alpha^j y^i y^j \left(\Phi\left(\mathbf{x}^i\right)^{\mathrm{T}} \Phi\left(\mathbf{x}^i\right)\right) - \sum_i \alpha^i$$

$$\text{s.t. } 0 \leq \alpha^i \leq C, \forall \mathbf{x}^i$$

$$\sum_i \alpha^i y^i = 0$$

- NOTICE: Data only appear as $\Phi(\mathbf{x}_i)^{\mathrm{T}}\,\Phi(\mathbf{x}_j)$

The Kernel trick

- $\Phi(\mathbf{x}_i)^T \Phi(\mathbf{x}_j)$: means, map data into new space, then take the inner product of the new vectors
- We can find a function such that: $K(\mathbf{x}_i , \mathbf{x}_j) = \Phi(\mathbf{x}_i)^T \Phi(\mathbf{x}_j)$, i.e., the function evaluates the inner product of the images of the data
- Then, we do not need to explicitly map the data into the high-dimensional space to solve the optimization problem (for training)

The Kernel trick and new instances

- How do we classify without explicitly mapping the new instances? It turns out that

$$\mathrm{sgn}\left(\mathbf{w}^{\mathrm{T}}\Phi(\mathbf{x}) + b\right) = \mathrm{sgn}\left(\sum_i \alpha^i y^i K\left(\mathbf{x}^i, \mathbf{x}\right) + b\right)$$

- b can be extracted by solving

$$\alpha^j\left(y^j \sum_i \alpha^i y^i K\left(\mathbf{x}^i, \mathbf{x}^j\right) + b - 1\right) = 0$$

for any j with $\alpha^j \neq 0$

Polynomial kernel

- Consider we have two variables x_1 and x_2 at disposal.
- We build the mapping

$$\Phi : \Re^2 \to \Re^6 : \Phi(x_1, x_2) = \left(x_1^2, x_2^2, \sqrt{2}x_1 x_2, x_1, x_2, 1\right)$$

- Let us define the kernel

$$K(\mathbf{x}, \mathbf{z}) = \left(\mathbf{x}^T \mathbf{z} + 1\right)^2$$

- We have

$$\begin{aligned}\Phi(\mathbf{x})^T \Phi(\mathbf{z}) &= x_1^2 z_1^2 + x_2^2 z_2^2 + 2 x_1 x_2 z_1 z_2 + x_1 z_1 + x_2 z_2 + 1 \\ &= (x_1 z_1 + x_2 z_2 + 1)^2 \\ &= K(\mathbf{x}, \mathbf{z})\end{aligned}$$

Polynomial kernel

- $K(\mathbf{x},\mathbf{z}) = \left(\mathbf{x}^T\mathbf{z} + 1\right)^p$

  is called the polynomial kernel of degree p.

- For $p=2$ and $D=1000$
  - building explicitly the mappings $\Phi(\mathbf{x})$ and $\Phi(\mathbf{z})$ then calculating the inner product between $\Phi(\mathbf{x})$ and $\Phi(\mathbf{z})$ means
    - to calculate around 10^6 new features, and
    - to take the inner product of two 10^6–dimensional vectors
  - Using the kernel means
    - to take the inner product of two 10^3–dimensional vectors
    - To take the square of the result
- In general, using the Kernel trick provides huge computational savings over explicit mapping!

Gaussian kernel

- The Gaussian kernel

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

  is widely used
- There is a hyperparameter (σ)

- In theory, it maps instances to an infinite-dimensional space
  - Quite difficult to write/compute $\Phi(\mathbf{x})$ explicitly…
- In practice, the dimension of the features space is the number of instances

Kernels and Mercer condition

- Kernels must be symmetric (obviously)

- Is there a mapping $\Phi(\mathbf{x})$ for any symmetric function $K(\mathbf{x},\mathbf{z})$? No.
- The SVM dual formulation requires calculation $K(\mathbf{x}^i, \mathbf{x}^j)$ for each pair of training instances. The array $G^{ij} = K(\mathbf{x}^i, \mathbf{x}^j)$ is called the Gram matrix
- Mercer condition: there is a feature space $\Phi(\mathbf{x})$ when the Kernel is such that G is always semi-positive definite

- How to build kernels? If $K_1(\mathbf{x},\mathbf{z})$ and $K_2(\mathbf{x},\mathbf{z})$ are kernels, and $p(.)$ is a polynomial, then

$$\left.\begin{array}{l} aK_1(\mathbf{x},\mathbf{z}) + bK_2(\mathbf{x},\mathbf{z}) \\ K_1(\mathbf{x},\mathbf{z})K_2(\mathbf{x},\mathbf{z}) \\ p(K_1(\mathbf{x},\mathbf{z})) \\ \exp(K_1(\mathbf{x},\mathbf{z})) \\ \text{etc.} \end{array}\right\} \text{ are kernels too}$$

Outline

- Introduction
- Large-margin classifier
  - Motivation
  - Optimization problem
- Soft margin classifier
  - Motivation
  - Optimization problem
- Mapping to feature space
  - Motivation
  - Dual SVM problem and the kernel trick
  - Kernels and Mercer’s condition
- Other kernel methods
- Discussion and conclusions

Other types of kernel methods

- Multi-class SVMs
- SVMs that perform regression
- SVMs that perform clustering
- Kernels suitable for sequences of strings, or other specialized kernels
  - Very useful when instances $\mathbf{x}$ are not standard vectors: strings (genomics), functions (infinite-dimensional objects, time series), etc.
  - No need to have the instances $\mathbf{x}$: the knowledge of the “distance” kernel $K(\mathbf{x}^i, \mathbf{x}^j)$ for each pair of instances is sufficient!

- Basically all data analysis methods that can be expressed in terms of $\mathbf{x}^T\mathbf{z}$ can be “kernelized”:
  - Principal Component Analysis
  - Partial Least Squares
  - Self-Organizing Maps
  - Etc.

Multi-class SVMs

- One-versus-all
  - Train n binary classifiers, one for each class against all other classes.
  - Predicted class is the class of the most confident classifier
- One-versus-one
  - Train $n(n-1)/2$ classifiers, each discriminating between a pair of classes
  - Several strategies for selecting the final classification based on the output of the binary SVMs
- Truly MultiClass SVMs
  - Generalize the SVM formulation to multiple categories

Variable selection with SVMs

- Recursive Feature Elimination
  - Train a linear SVM
  - Remove the variables with the lowest weights (those variables affect classification the least), e.g., remove the lowest 50% of variables
  - Retrain the SVM with remaining variables and repeat until classification is reduced
- Very successful
- Other formulations exist where minimizing the number of variables is folded into the optimization problem
- Similar algorithms exist for non-linear SVMs
- Some of the best and most efficient variable selection methods

Outline

Comparison with multi-layer perceptrons

MLP

- Hidden Layers map to moderate-dimensional spaces (higher or lower)
- Search space has multiple local minima
- Training is expensive
- Classification extremely efficient
- Requires number of hidden units and layers
- Very good accuracy in typical domains

SVM

- Kernel maps to a very-high dimensional space
- Search space has a unique minimum
- Training is extremely efficient
- Classification extremely efficient
- Requires kernel, kernel hyperparameters and regularization constant *C*
- Very good accuracy in typical domains
- Extremely robust

Why do SVM generalize?

- Mapping to a very high-dimensional space: risk of high number of parameters, thus overfitting?

- Not really:
  - Model in feature space is very constrained (strong bias in that space)
  - Number of parameters limited by # instances (solution has to be a linear combination of the training instances)

- Large theory on Structural Risk Minimization (Vapnik, ...) providing bounds on the error of an SVM
- Typically the error bounds too loose to be of practical use
  - Except, tentatively, to compare models (choice of kernel, hyperparameters, ...)

Conclusions

- SVMs express learning as a mathematical problem taking advantage of the rich theory in optimization
  - quadratic optimization problem
  - # unknowns = # data
    (advantageous in high-dimensional spaces, moderate sample size)
  - SVM includes many models (flexibility on the choice of the kernel)

- SVM uses the kernel trick to map indirectly to extremely high dimensional spaces

- SVMs are extremely successful, robust, efficient, and versatile while there are good theoretical indications as to why they generalize well

Sources and references

- Sources
  - Most of these slides come from (or are largely inspired by) MEDINFO 2004, tutorial on Machine Learning Methods for Decision Support and Discovery, by Constantin F. Aliferis & Ioannis Tsamardinos

- Further readings

  - An introduction to Support Vector Machines and other kernel-based learning methods, N. Cristianini, J. Shawe-Taylor, Cambridge University Press, 2002.
  - Learning with Kernels, Support Vector Machines, Regularization, Optimization and Beyond, B. Schölkopf, A.J. Smola, MIT Press, 2002
  - http://www.kernel-machines.org/tutorial.html
  - C. J. C. Burges. A Tutorial on Support Vector Machines for Pattern Recognition. Knowledge Discovery and Data Mining, 2(2), 1998.
  - Hastie, Tibshirani, Friedman, The Elements of Statistical Learning, Springer 2001