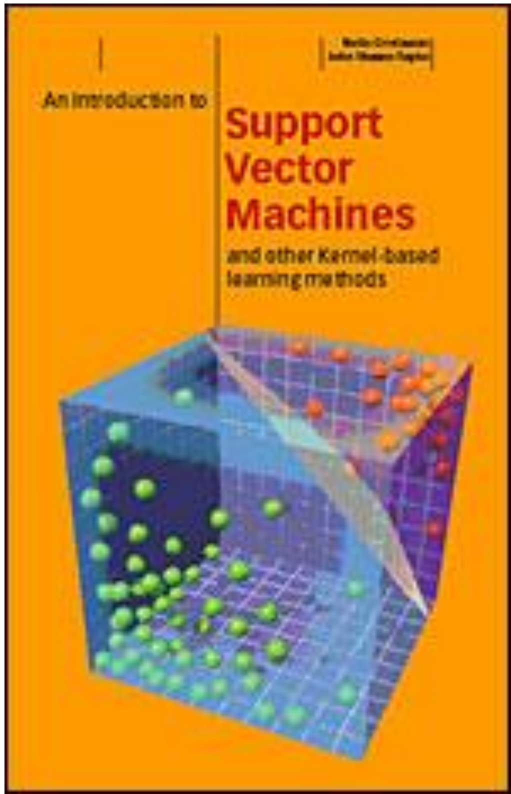
Support Vector Machine Classifiers

Outline

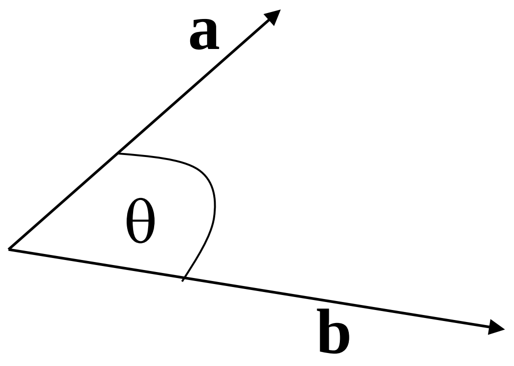
- Support Vector Machines for Classification
  - Linear Discrimination
  - Nonlinear Discrimination
- SVM Mathematically
- Extensions
- Data Classification
- Kernel Functions

Definition

- 'Support Vector Machine is a system for efficiently training linear learning machines in kernel-induced feature spaces, while respecting the insights of generalisation theory and exploiting optimisation theory.'
  - **AN INTRODUCTION TO SUPPORT VECTOR MACHINES (**and other kernel-based learning methods)
    N. Cristianini and J. Shawe-Taylor
    Cambridge University Press
    2000 ISBN: 0 521 78019 5
  - Kernel Methods for Pattern Analysis
    John Shawe-Taylor & Nello Cristianini
    Cambridge University Press, 2004



The Scalar Product



$$\mathbf{a} \cdot \mathbf{b} = |\mathbf{a}||\mathbf{b}| \cos\theta$$

The scalar or dot product is, in some sense, a measure of Similarity

Decision Function for binary classification

$$f(x) \in \mathbf{R}$$

$$f(x_i) \geq 0 \Rightarrow y_i = 1$$
$$f(x_i) < 0 \Rightarrow y_i = -1$$

Support Vector Machines

- SVMs pick best separating hyperplane according to some criterion
  - e.g. maximum margin
- Training process is an optimisation
- Training set is effectively reduced to a relatively small number of support vectors

Feature Spaces

- We may separate data by mapping to a higher-dimensional feature space
  - The feature space may even have an infinite number of dimensions!
- We need not explicitly construct the new feature space

Kernels

- We may use Kernel functions to implicitly map to a new feature space
- Kernel fn:

$$K\left(\mathbf{x}_1, \mathbf{x}_2\right) \in \mathbf{R}$$

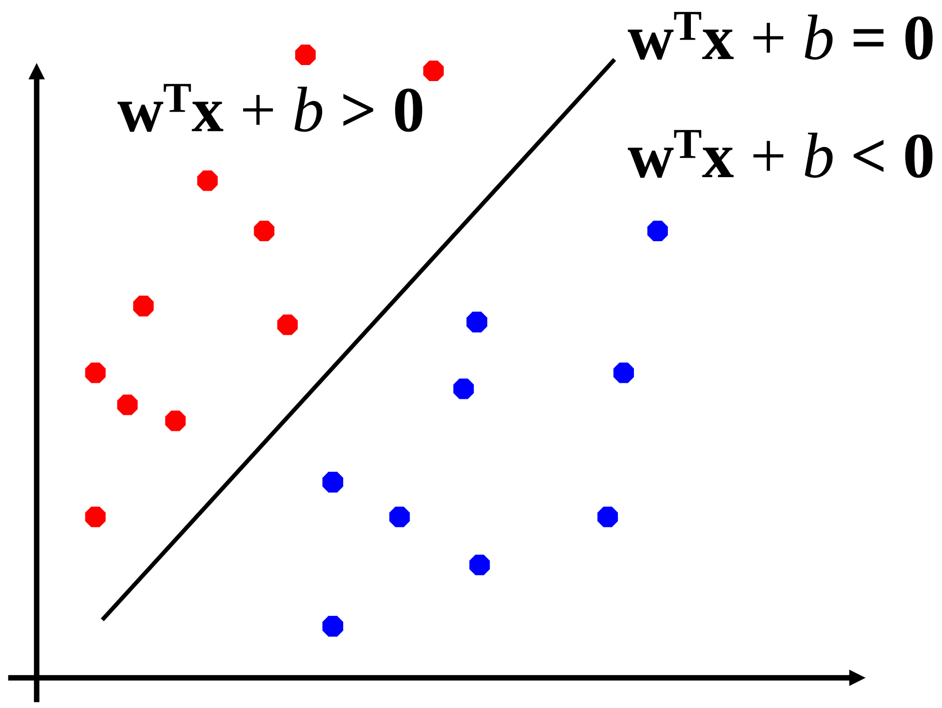
- Kernel must be equivalent to an inner product in some feature space

Example Kernels

Linear:	$\langle \mathbf{x} \cdot \mathbf{z} \rangle$
Polynomial:	$P(\langle \mathbf{x} \cdot \mathbf{z} \rangle)$
Gaussian:	$\exp\left(-\lVert \mathbf{x} - \mathbf{z} \rVert^2 / \sigma^2\right)$

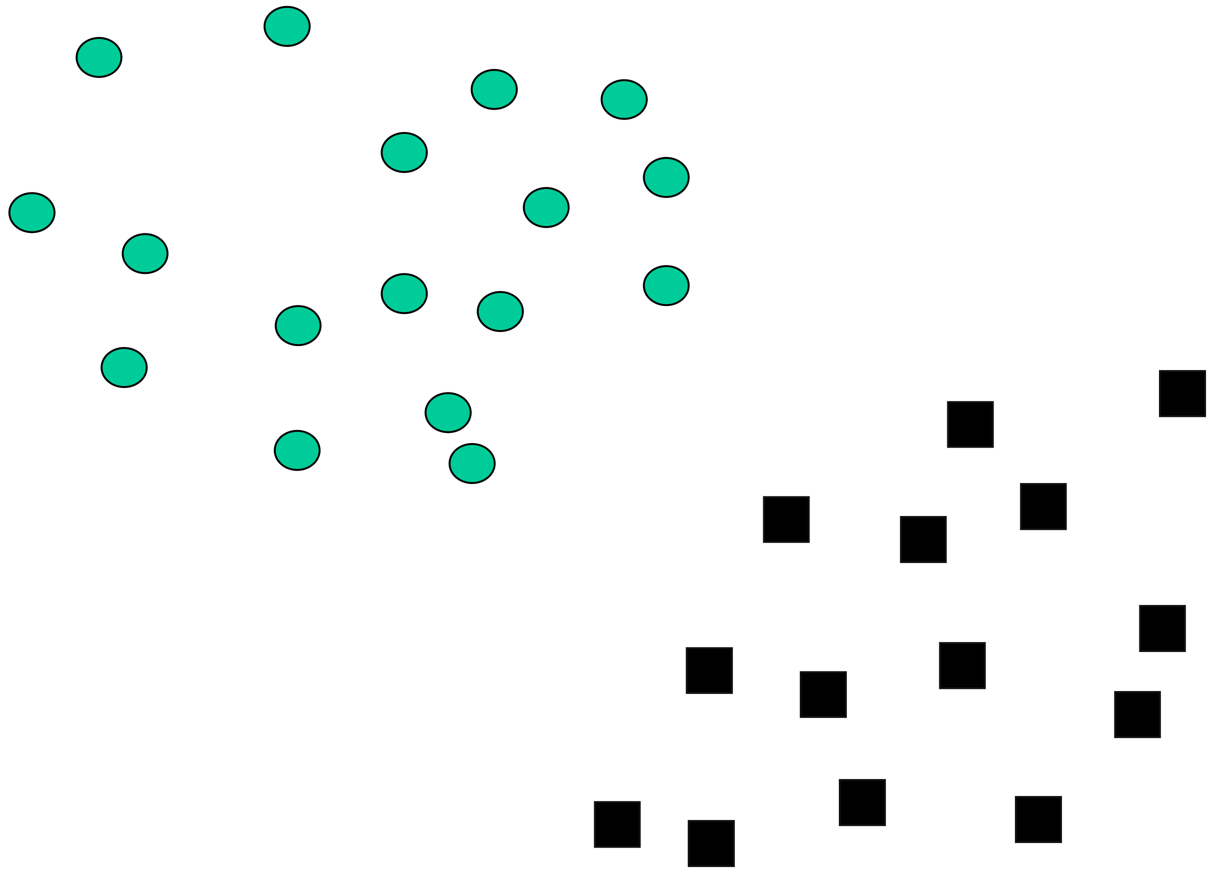
Perceptron Revisited: Linear Separators

- Binary classification can be viewed as the task of separating classes in feature space:

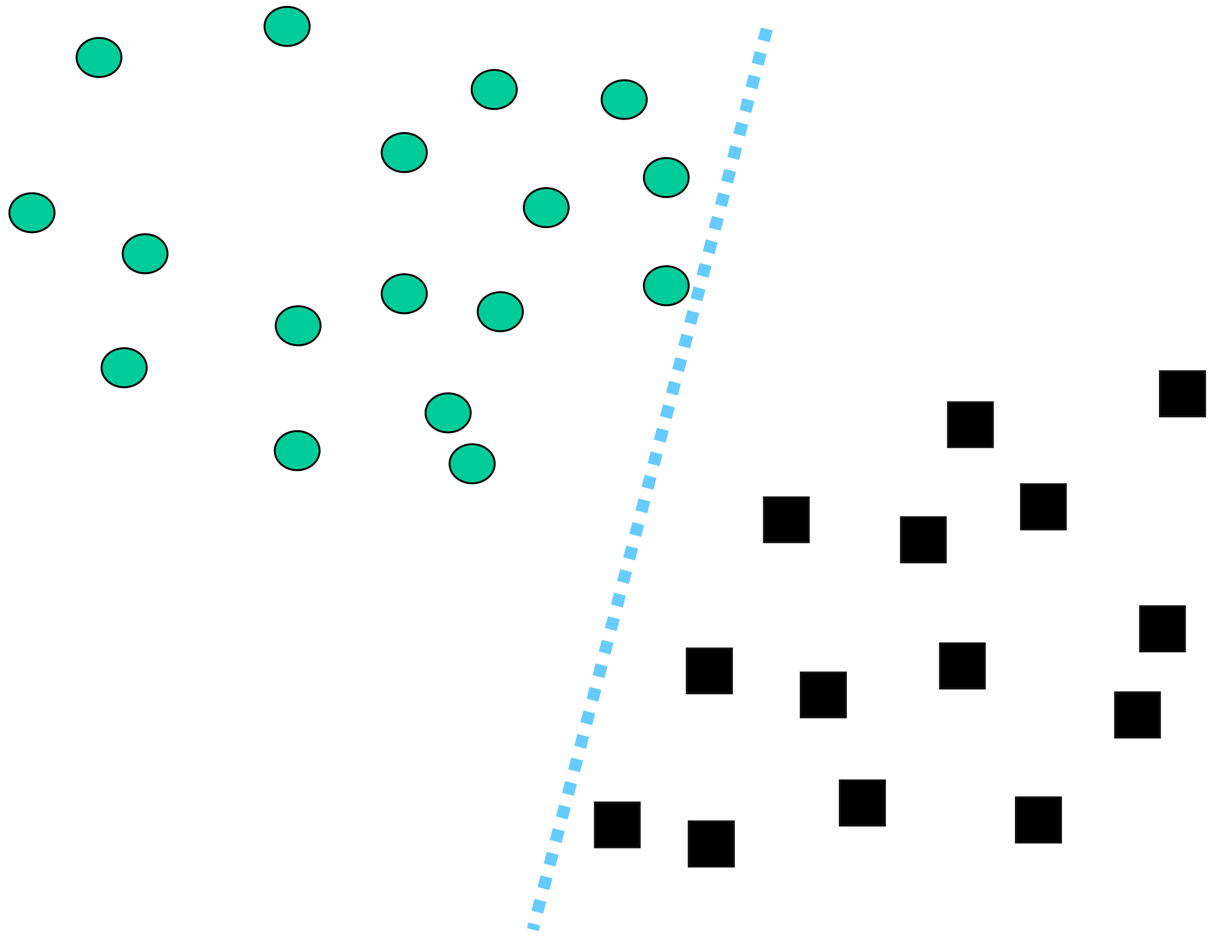

$\mathbf{w}^T\mathbf{x} + b = \mathbf{0}$
$\mathbf{w}^T\mathbf{x} + b > \mathbf{0}$
$\mathbf{w}^T\mathbf{x} + b < \mathbf{0}$

$f(\mathbf{x}) = \text{sign}(\mathbf{w}^T\mathbf{x} + b)$

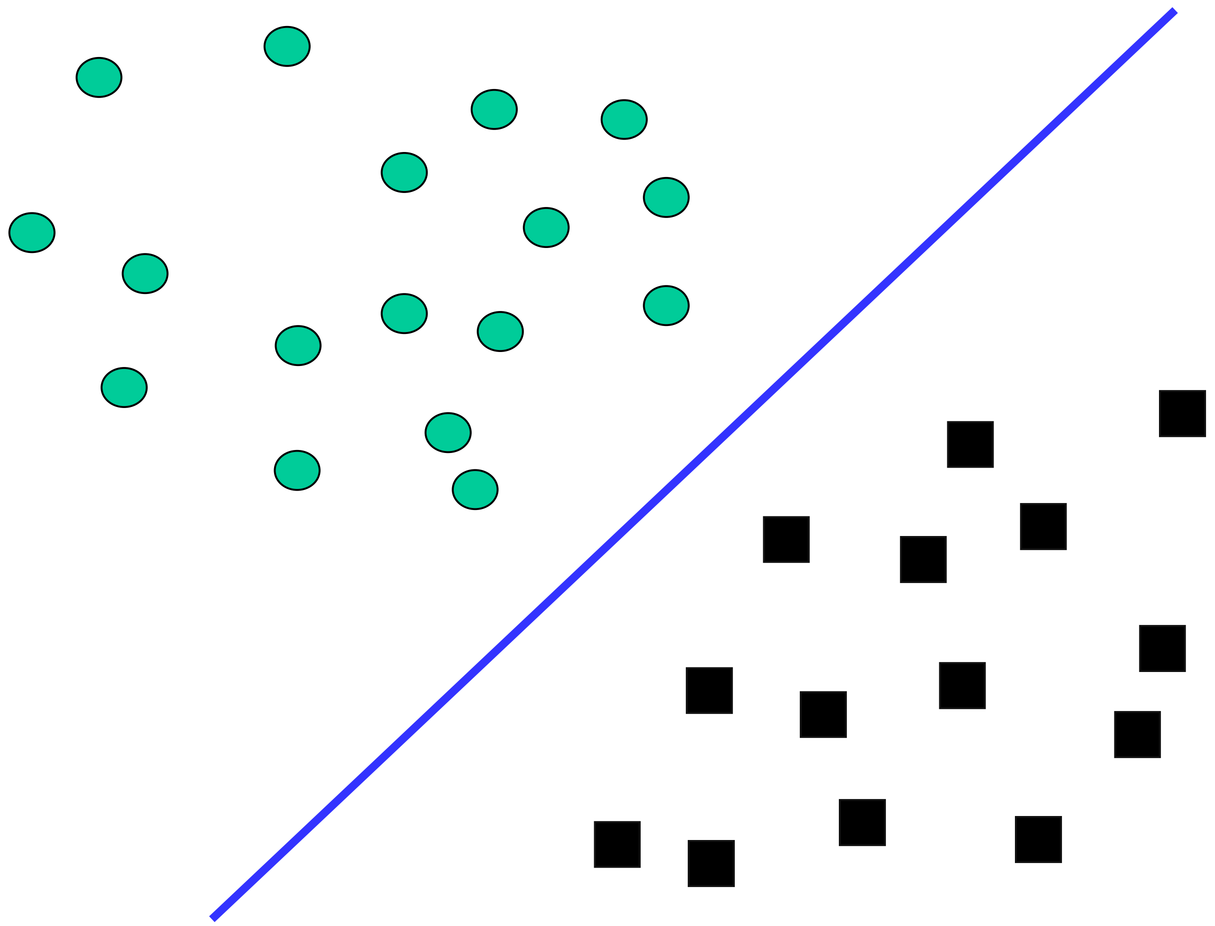
Which of the linear separators is optimal?



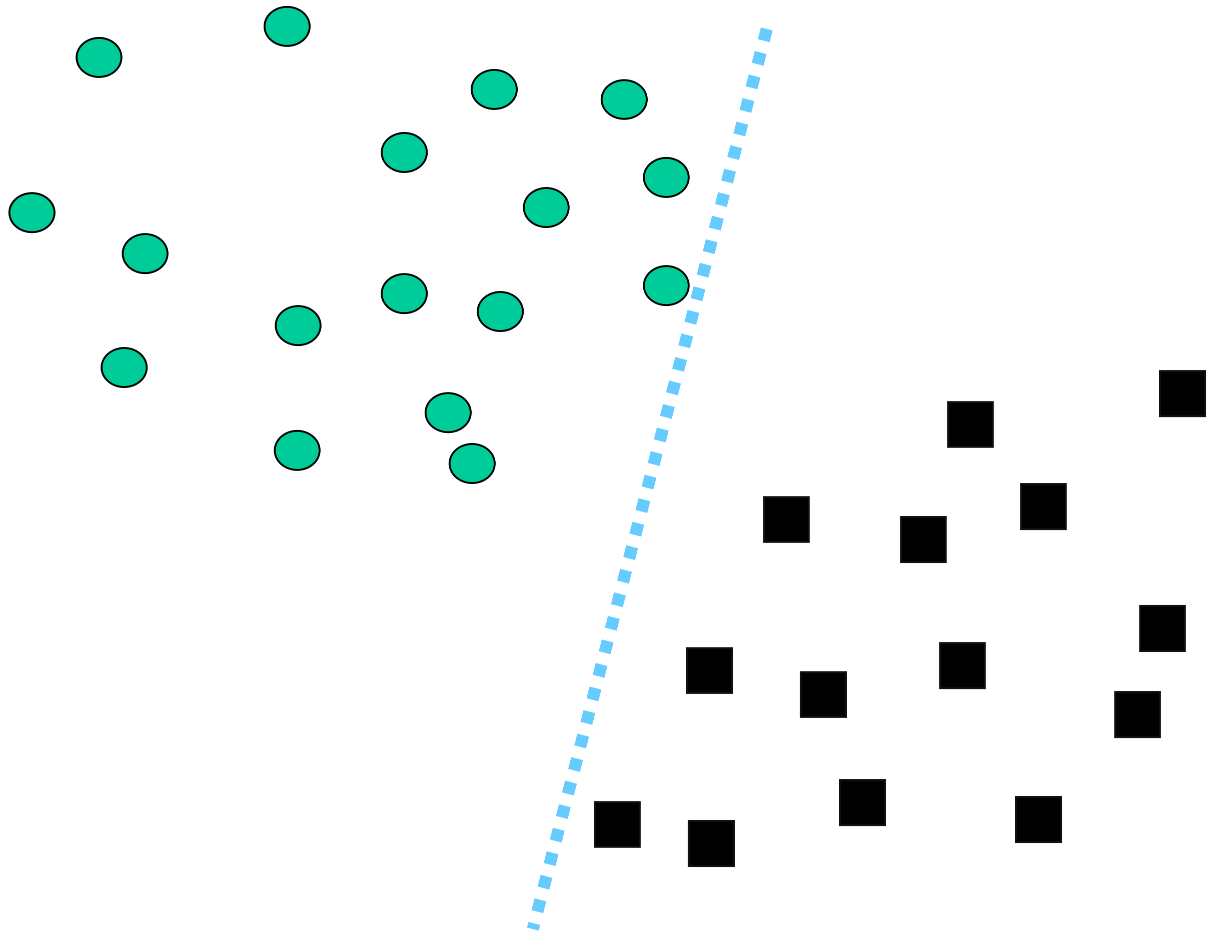
Best Linear Separator?



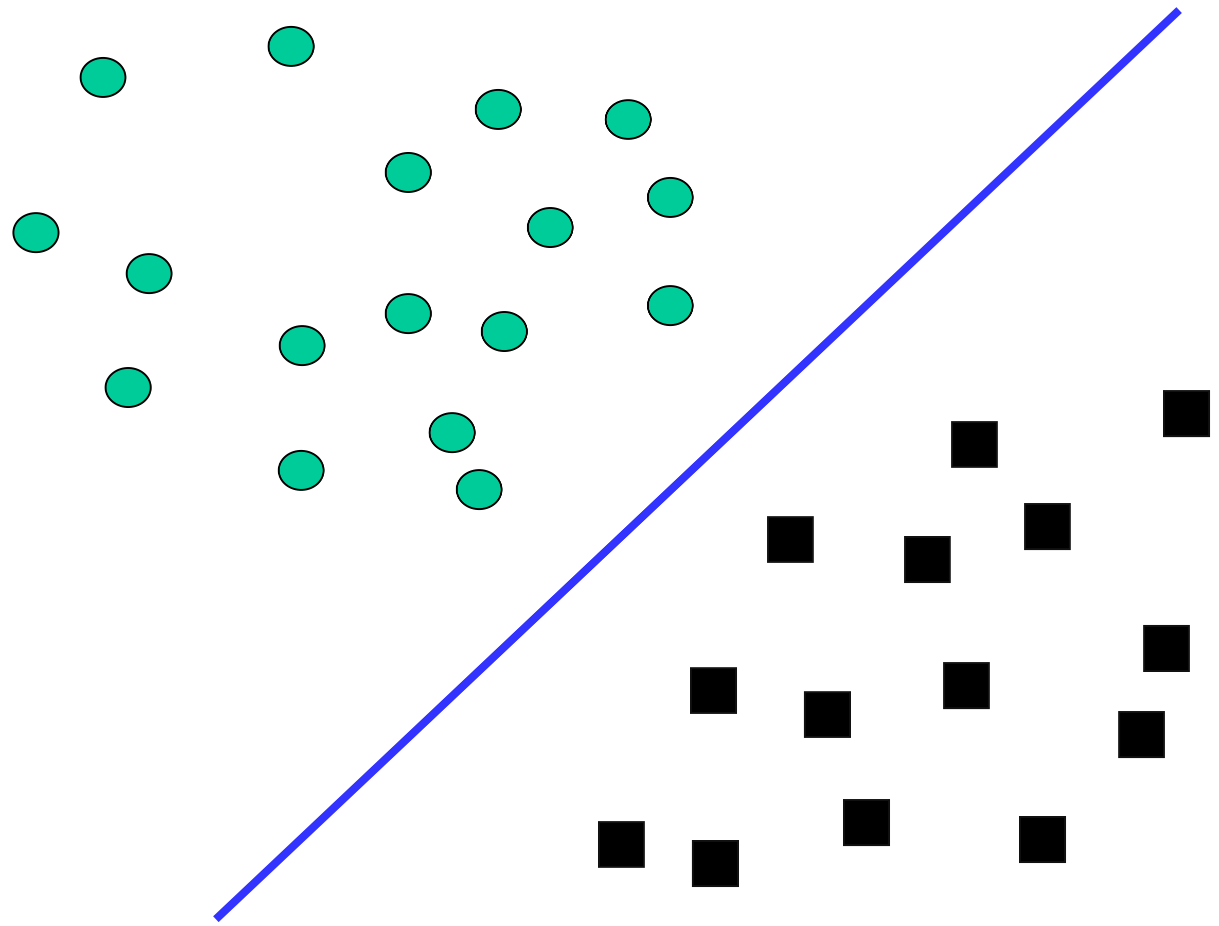
Best Linear Separator?



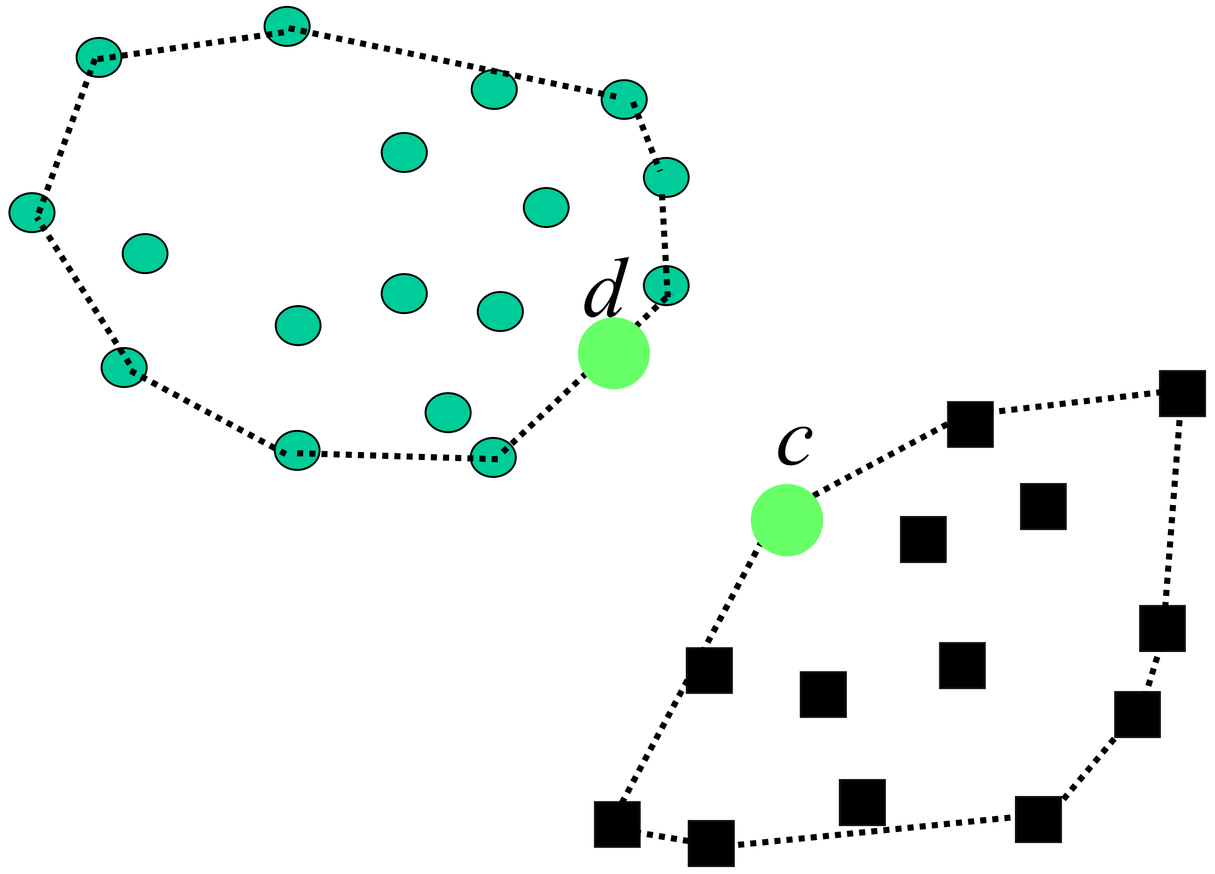
Best Linear Separator?



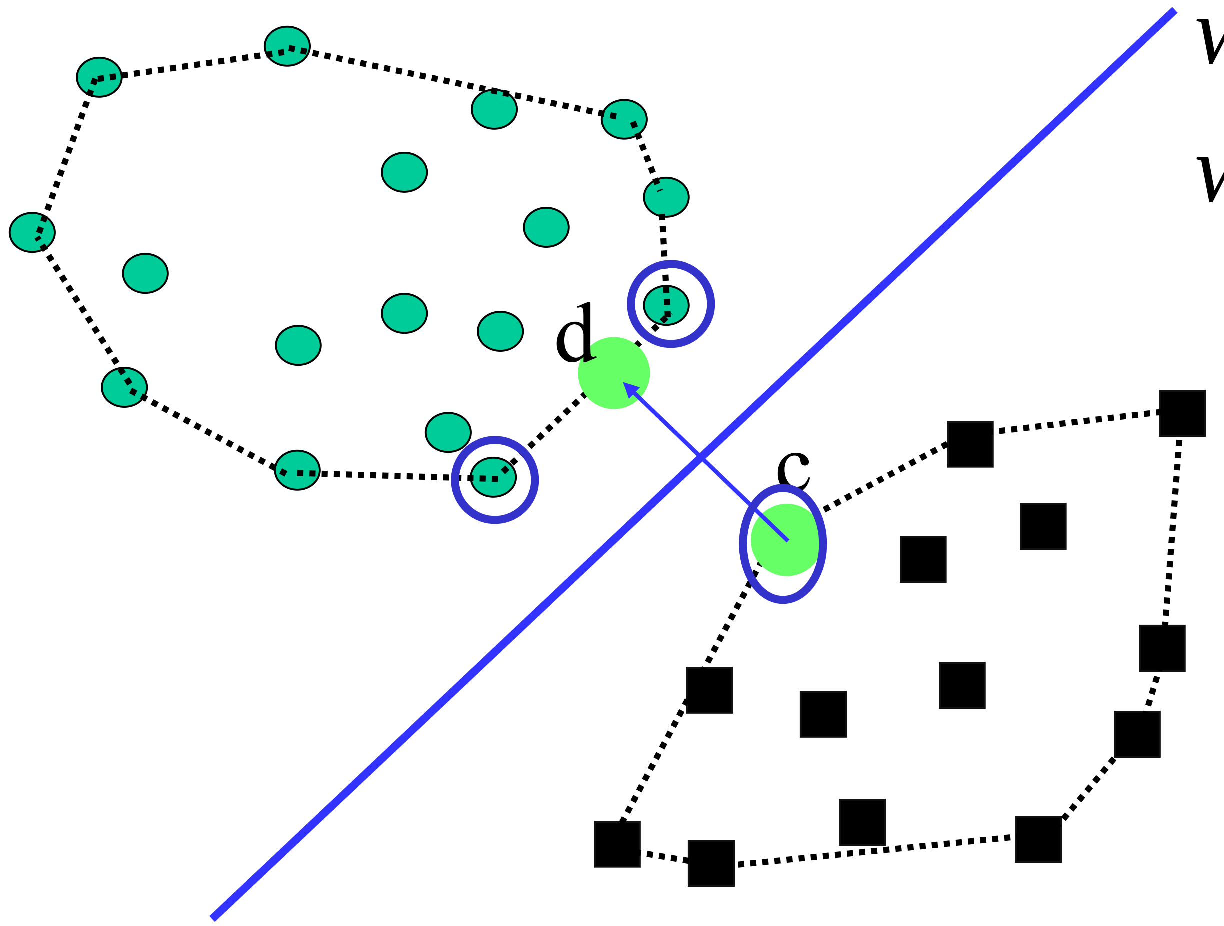
Best Linear Separator?



Find Closest Points in Convex Hulls



Plane Bisect Closest Points

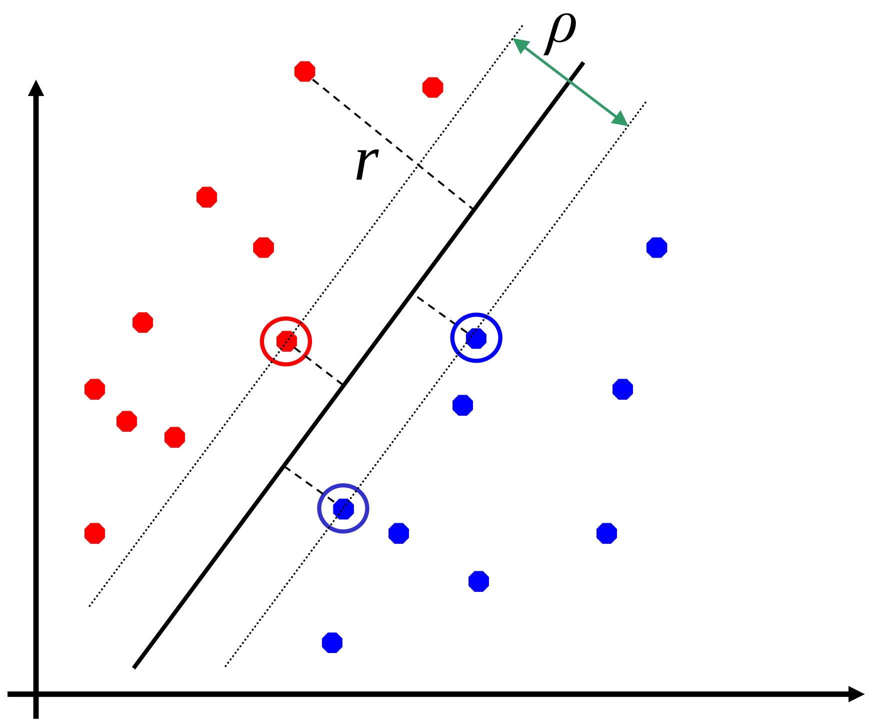


$$w^T x + b = 0$$

$$w = d - c$$

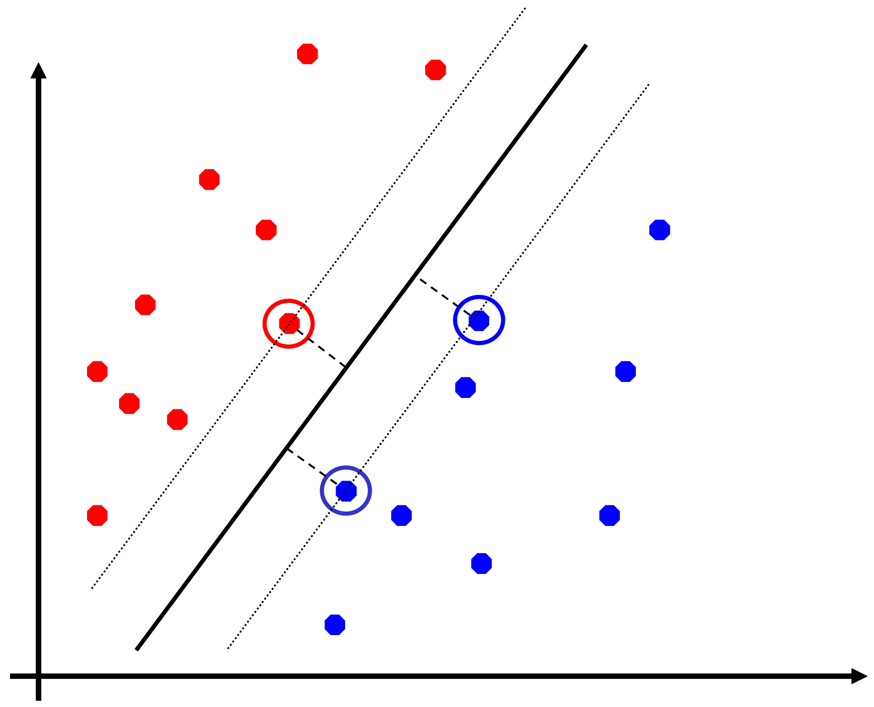
Classification Margin

- Distance from example data to the separator is $r = \dfrac{\mathbf{w}^T\mathbf{x}+b}{\|\mathbf{w}\|}$
- Data closest to the hyperplane are ***support vectors***.
- ***Margin*** ρ of the separator is the width of separation between classes.



Maximum Margin Classification

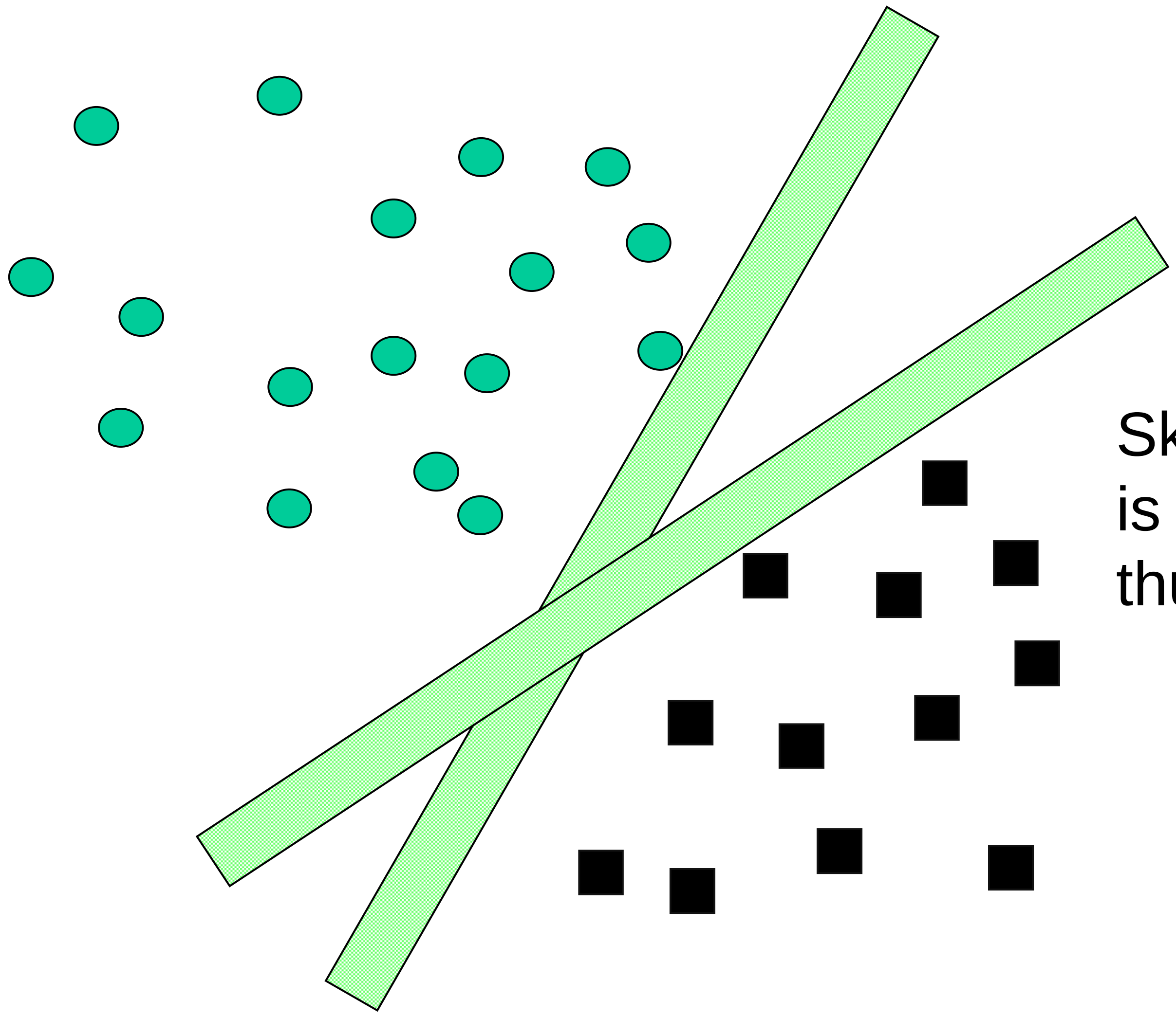
- Maximizing the margin is good according to intuition and theory.
- Implies that only support vectors are important; other training examples are ignorable.



Statistical Learning Theory

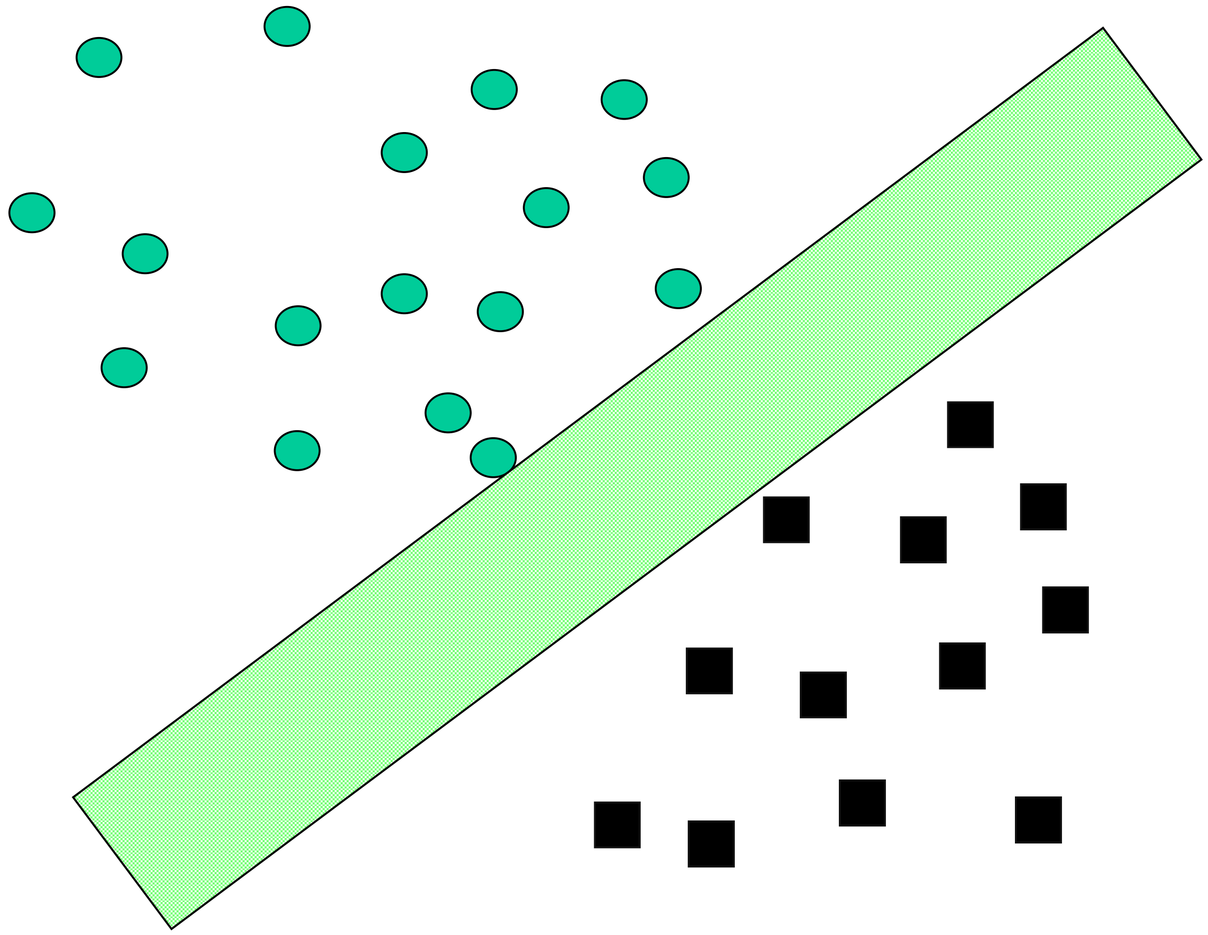
- Misclassification error and the function complexity bound generalization error.
- Maximizing margins minimizes complexity.
- "Eliminates" overfitting.
- Solution depends only on *Support Vectors* not number of attributes.

Margins and Complexity



Skinny margin
is more flexible
thus more complex.

Margins and Complexity



Fat margin
is less complex.

Linear SVM Mathematically

- Assuming all data is at distance larger than 1 from the hyperplane, the following two constraints follow for a training set $\{(\mathbf{x_i}, y_i)\}$

$$\mathbf{w^T x_i} + b \geq 1 \quad \text{if } y_i = 1$$

$$\mathbf{w^T x_i} + b \leq -1 \quad \text{if } y_i = -1$$

- For support vectors, the inequality becomes an equality; then, since each example's distance from the
- hyperplane is $r = \dfrac{\mathbf{w}^T\mathbf{x} + b}{\|\mathbf{w}\|}$ the margin is: $\rho = \dfrac{2}{\|\mathbf{w}\|}$

Linear SVMs Mathematically (cont.)

- Then we can formulate the *quadratic optimization problem:*

Find $\mathbf{w}$ and b such that

$\rho = \frac{2}{\|\mathbf{w}\|}$ is maximized and for all $\{(\mathbf{x_i}, y_i)\}$

$\mathbf{w}^T\mathbf{x_i} + b \geq 1$ if $y_i=1$; $\mathbf{w}^T\mathbf{x_i} + b \leq -1$ if $y_i = -1$

A better formulation:

Find $\mathbf{w}$ and b such that

$\mathbf{\Phi}(\mathbf{w}) = \frac{1}{2}\,\mathbf{w}^T\mathbf{w}$ is minimized and for all $\{(\mathbf{x_i}, y_i)\}$

$y_i\,(\mathbf{w}^T\mathbf{x_i} + b) \geq 1$

Solving the Optimization Problem

Find **w** and b such that

$\mathbf{\Phi}(\mathbf{w}) = ½\, \mathbf{w}^T\mathbf{w}$ is minimized and for all $\{(\mathbf{x_i}, y_i)\}$

$y_i (\mathbf{w}^T\mathbf{x_i} + b) \geq 1$

- Need to optimize a *quadratic* function subject to *linear* constraints.
- Quadratic optimization problems are a well-known class of mathematical programming problems, and many (rather intricate) algorithms exist for solving them.
- The solution involves constructing a *dual problem* where a *Lagrange multiplier* α_i is associated with every constraint in the primary problem:

Find $\alpha_1 \dots \alpha_N$ such that

$\mathbf{Q}(\boldsymbol{\alpha}) = \Sigma\alpha_i - ½\Sigma\Sigma\alpha_i\alpha_j y_i y_j \mathbf{x_i}^T\mathbf{x_j}$ is maximized and

(1) $\Sigma\alpha_i y_i = 0$

(2) $\alpha_i \geq 0$ for all α_i

The Optimization Problem Solution

- The solution has the form:

$$\boxed{\mathbf{w} = \Sigma\alpha_i y_i \mathbf{x_i} \qquad b = y_k - \mathbf{w}^\mathbf{T}\mathbf{x_k} \text{ for any } \mathbf{x_k} \text{ such that } \alpha_k \neq 0}$$

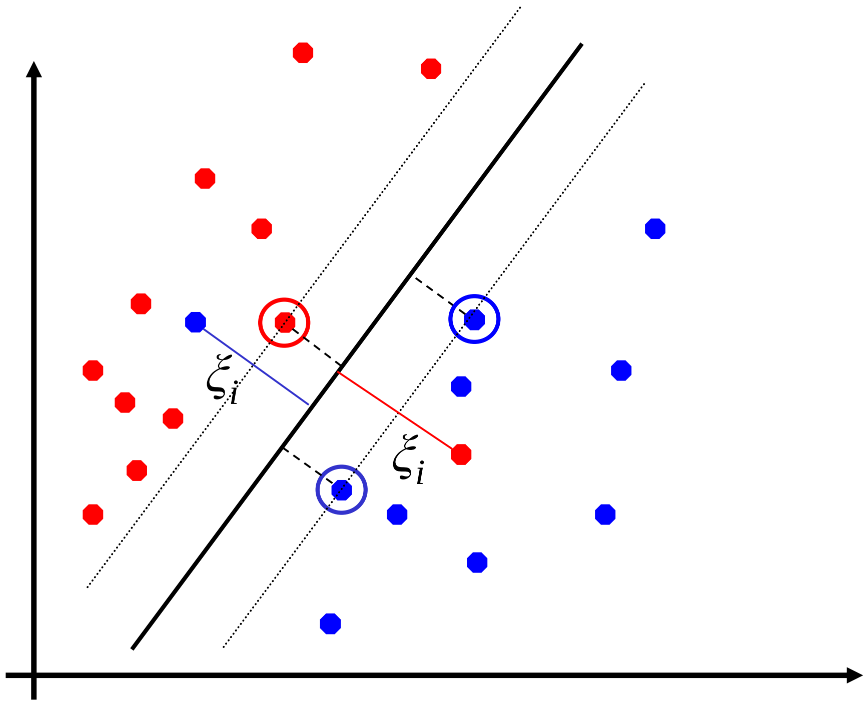
- Each non-zero α_i indicates that corresponding $\mathbf{x_i}$ is a support vector.
- Then the classifying function will have the form:

$$\boxed{f(\mathbf{x}) = \Sigma\alpha_i y_i \mathbf{x_i}^\mathbf{T}\mathbf{x} + b}$$

- Notice that it relies on an *inner product* between the test point $\mathbf{x}$ and the support vectors $\mathbf{x_i}$ – we will return to this later!
- Also keep in mind that solving the optimization problem involved computing the inner products $\mathbf{x_i}^\mathbf{T}\mathbf{x_j}$ between all training points!

Soft Margin Classification

- What if the training set is not linearly separable?
- *Slack variables* ξ_i can be added to allow misclassification of difficult or noisy examples.



Soft Margin Classification Mathematically

- The old formulation:

Find **w** and *b* such that
$\mathbf{\Phi}(\mathbf{w}) = \frac{1}{2}\, \mathbf{w}^T\mathbf{w}$ is minimized and for all $\{(\mathbf{x_i}, y_i)\}$
$y_i (\mathbf{w^T x_i} + b) \geq 1$

- The new formulation incorporating slack variables:

Find **w** and *b* such that
$\mathbf{\Phi}(\mathbf{w}) = \frac{1}{2}\, \mathbf{w}^T\mathbf{w} + C\Sigma\xi_i$ is minimized and for all $\{(\mathbf{x_i}, y_i)\}$
$y_i (\mathbf{w^T x_i} + b) \geq 1 - \xi_i$ and $\xi_i \geq 0$ for all i

- Parameter *C* can be viewed as a way to control overfitting.

Soft Margin Classification – Solution

- The dual problem for soft margin classification:

  Find $\alpha_1 \ldots \alpha_N$ such that
  $\mathbf{Q}(\boldsymbol{\alpha}) = \Sigma\alpha_i - \frac{1}{2}\Sigma\Sigma\alpha_i\alpha_j y_i y_j \mathbf{x_i}^{\mathbf{T}}\mathbf{x_j}$ is maximized and
  (1) $\Sigma\alpha_i y_i = 0$
  (2) $0 \leq \alpha_i \leq C$ for all α_i

- Neither slack variables ξ_i nor their Lagrange multipliers appear in the dual problem!
- Again, $\mathbf{x_i}$ with non-zero α_i will be support vectors.
- Solution to the dual problem is:

$\mathbf{w} = \Sigma\alpha_i y_i \mathbf{x_i}$
$b = y_k(1 - \xi_k) - \mathbf{w}^{\mathbf{T}}\mathbf{x_k}$ where $k = \underset{k}{\operatorname{argmax}}\ \alpha_k$

But neither $\mathbf{w}$ nor b are needed explicitly for classification!

$f(\mathbf{x}) = \Sigma\alpha_i y_i \mathbf{x_i}^{\mathbf{T}}\mathbf{x} + b$

Theoretical Justification for Maximum Margins

- Vapnik has proved the following:

  The class of optimal linear separators has VC dimension h bounded from above as

$$h \le \min\left\{\left\lceil \frac{D^2}{\rho^2} \right\rceil, m_0\right\} + 1$$

  where ρ is the margin, D is the diameter of the smallest sphere that can enclose all of the training examples, and m_0 is the dimensionality.

- Intuitively, this implies that regardless of dimensionality m_0 we can minimize the VC dimension by maximizing the margin ρ.

- Thus, complexity of the classifier is kept small regardless of dimensionality.

Linear SVMs: Overview

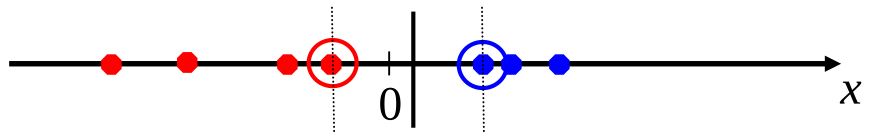
- The classifier is a *separating hyperplane.*
- Most "important" training points are support vectors; they define the hyperplane.
- Quadratic optimization algorithms can identify which training points $\mathbf{x_i}$ are support vectors with non-zero Lagrangian multipliers α_i.
- Both in the dual formulation of the problem and in the solution training points appear only inside inner products:

$$\boxed{\begin{array}{l}\text{Find } \alpha_1 \ldots \alpha_N \text{ such that} \\ \mathbf{Q}(\boldsymbol{\alpha}) = \Sigma\alpha_i - \tfrac{1}{2}\Sigma\Sigma\alpha_i\alpha_j y_i y_j \boxed{\mathbf{x_i}^\mathbf{T}\mathbf{x_j}} \text{ is maximized and} \\ (1)\ \ \Sigma\alpha_i y_i = 0 \\ (2)\ \ 0 \leq \alpha_i \leq C \text{ for all } \alpha_i \end{array}}$$

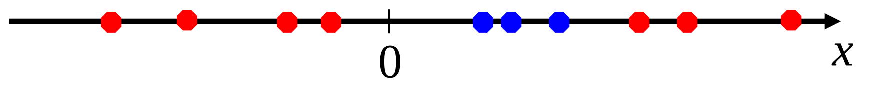
$$\boxed{f(\mathbf{x}) = \Sigma\alpha_i y_i \boxed{\mathbf{x_i}^\mathbf{T}\mathbf{x}} + b}$$

Non-linear SVMs

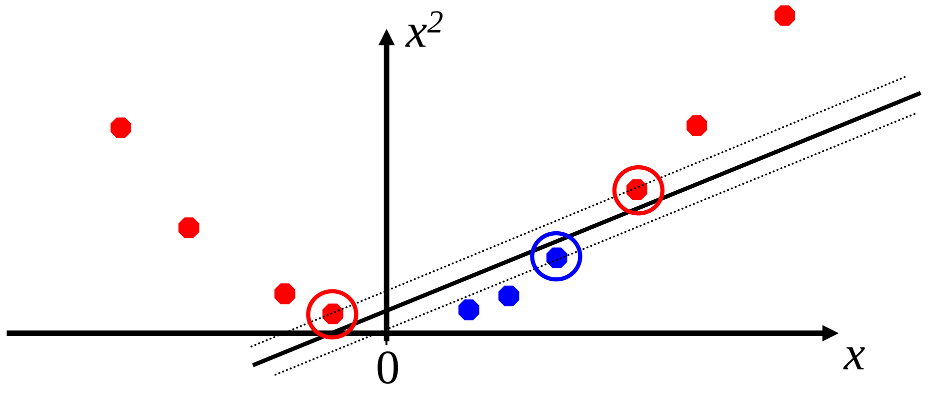
- Datasets that are linearly separable with some noise work out great:



- But what are we going to do if the dataset is just too hard?



- How about… mapping data to a higher-dimensional space:



Nonlinear Classification

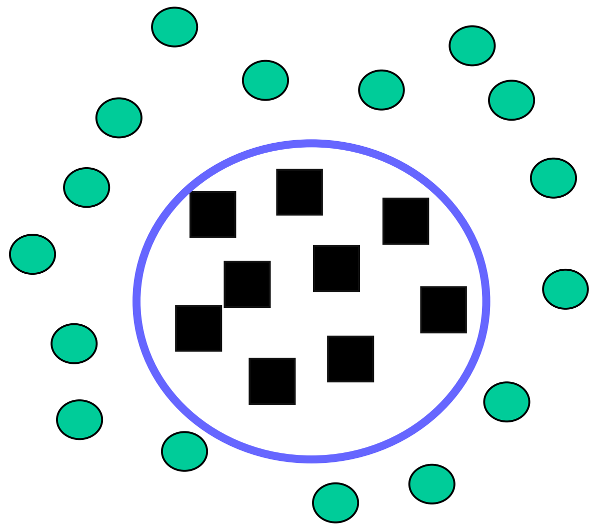
$$x = [a, b]$$

$$x \cdot w = w_1 a + w_2 b$$

$$\downarrow$$

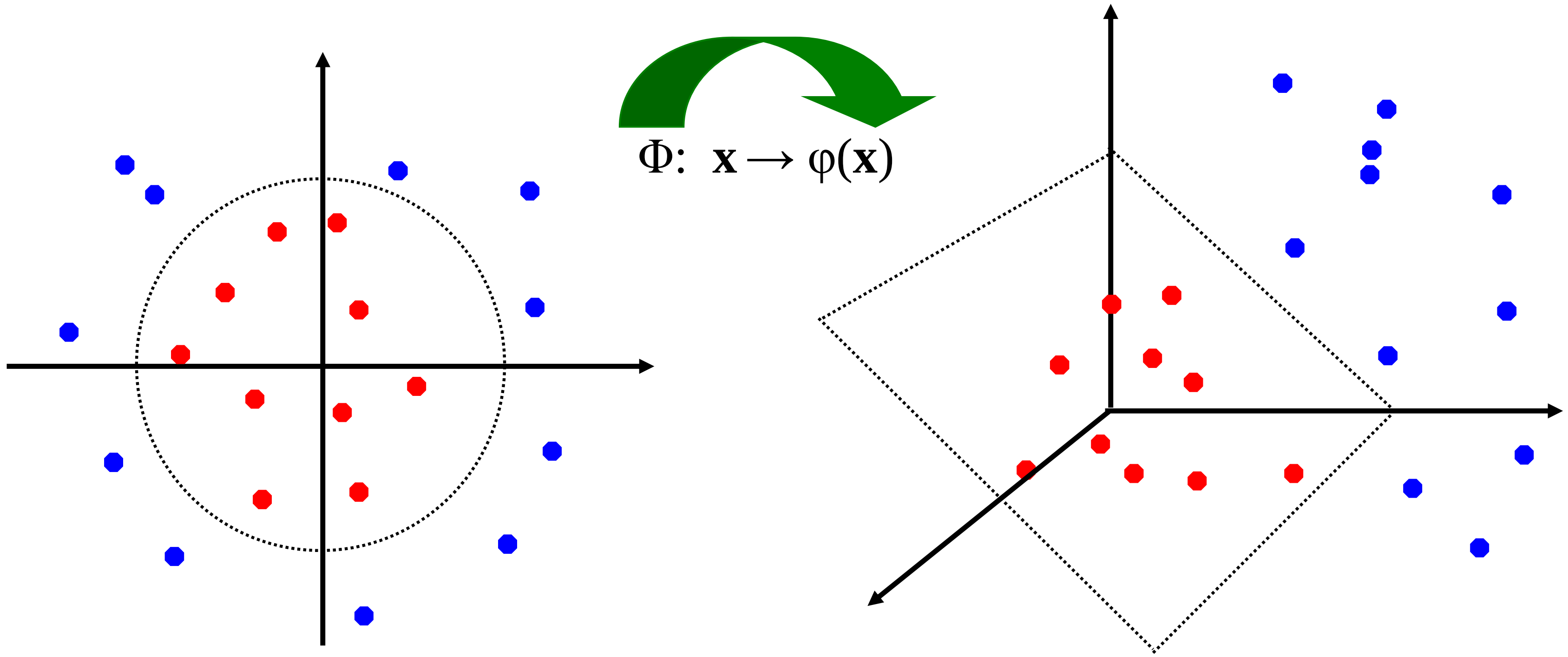
$$\theta(x) = [a, b, ab, a^2, b^2]$$

$$\theta(x) \cdot w = w_1 a + w_2 b + w_3 ab + w_4 a^2 + w_5 b^2$$



Non-linear SVMs: Feature spaces

- General idea: the original feature space can always be mapped to some higher-dimensional feature space where the training set is separable:



The “Kernel Trick”

- The linear classifier relies on inner product between vectors $K(\mathbf{x_i},\mathbf{x_j})=\mathbf{x_i}^{\mathbf{T}}\mathbf{x_j}$
- If every datapoint is mapped into high-dimensional space via some transformation Φ: $\mathbf{x} \rightarrow \varphi(\mathbf{x})$, the inner product becomes:

$$K(\mathbf{x_i},\mathbf{x_j})= \varphi(\mathbf{x_i})^{\mathbf{T}}\varphi(\mathbf{x_j})$$

- A *kernel function* is some function that corresponds to an inner product into some feature space.
- Example:

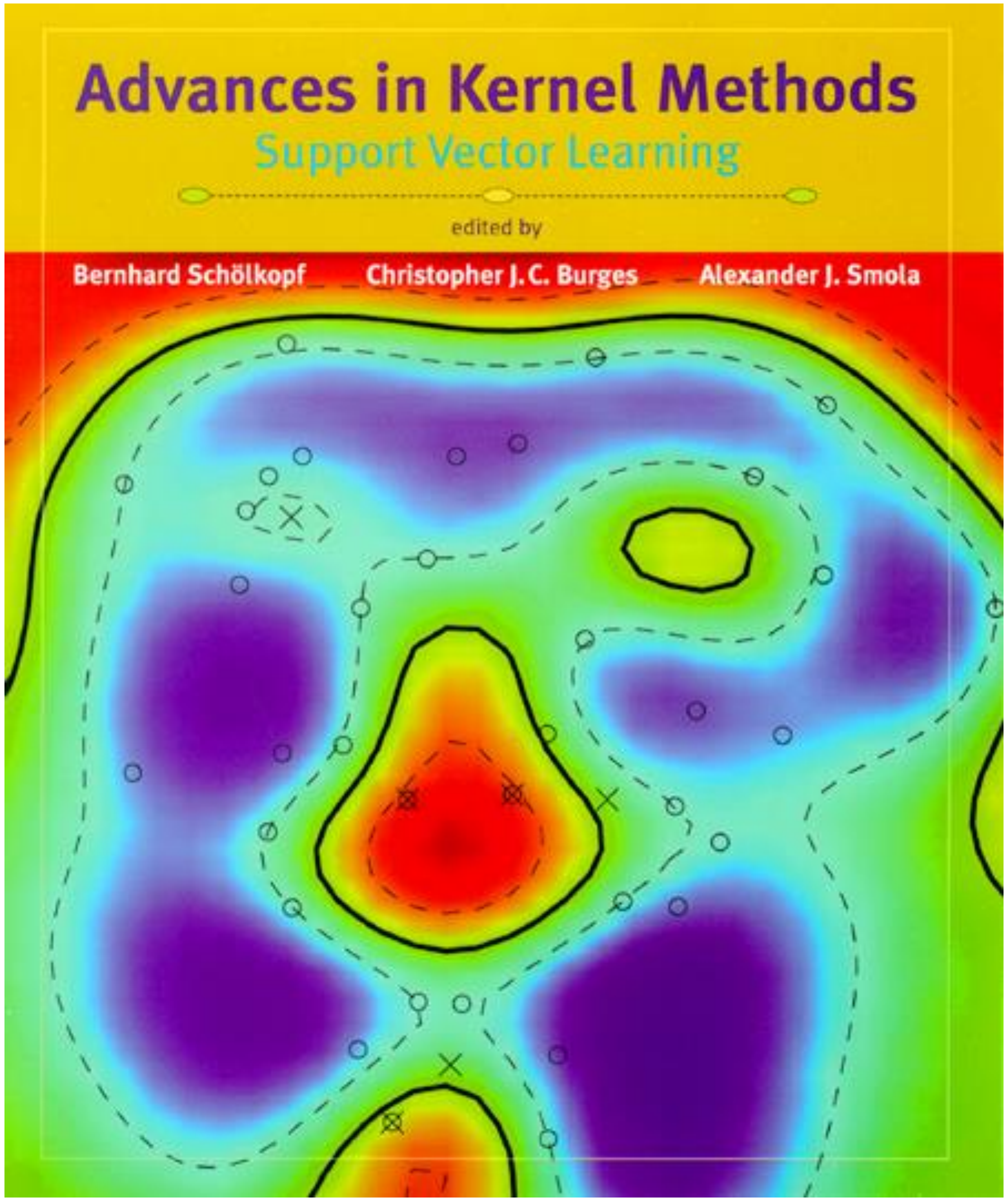
2-dimensional vectors $\mathbf{x}=[x_1 \;\; x_2]$; let $K(\mathbf{x_i},\mathbf{x_j})=(1 + \mathbf{x_i}^{\mathbf{T}}\mathbf{x_j})^2$,

Need to show that $K(\mathbf{x_i},\mathbf{x_j})= \varphi(\mathbf{x_i})^{\mathbf{T}}\varphi(\mathbf{x_j})$:

$$K(\mathbf{x_i},\mathbf{x_j})=(1 + \mathbf{x_i}^{\mathbf{T}}\mathbf{x_j})^2 = 1+ x_{i1}^2x_{j1}^2 + 2\, x_{i1}x_{j1}\, x_{i2}x_{j2}+ x_{i2}^2x_{j2}^2 + 2x_{i1}x_{j1} + 2x_{i2}x_{j2}=$$

$$= [1 \;\; x_{i1}^2 \;\; \sqrt{2}\, x_{i1}x_{i2} \;\; x_{i2}^2 \;\; \sqrt{2}x_{i1} \;\; \sqrt{2}x_{i2}]^{\mathbf{T}} [1 \;\; x_{j1}^2 \;\; \sqrt{2}\, x_{j1}x_{j2} \;\; x_{j2}^2 \;\; \sqrt{2}x_{j1} \;\; \sqrt{2}x_{j2}] =$$

$$= \varphi(\mathbf{x_i})^{\mathbf{T}}\varphi(\mathbf{x_j}), \quad \text{where } \varphi(\mathbf{x}) = [1 \;\; x_1^2 \;\; \sqrt{2}\, x_1x_2 \;\; x_2^2 \;\; \sqrt{2}x_1 \;\; \sqrt{2}x_2]$$



Advances in Kernel Methods

Support Vector Learning

edited by

Bernhard Schölkopf Christopher J.C. Burges Alexander J. Smola

Positive Definite Matrices

A square matrix A is *positive definite if* $x^TAx>0$ for all nonzero column vectors x.

It is negative definite if $x^TAx < 0$ for all nonzero x.

It is *positive semi-definite* if $x^TAx \geq 0$.

And *negative semi-definite* if $x^TAx \leq 0$ for all x.

What Functions are Kernels?

- For some functions $K(\mathbf{x}_i,\mathbf{x}_j)$ checking that $K(\mathbf{x}_i,\mathbf{x}_j)= \varphi(\mathbf{x}_i)^T\varphi(\mathbf{x}_j)$ can be cumbersome.
- Mercer's theorem:

  Every semi-positive definite symmetric function is a kernel

- Semi-positive definite symmetric functions correspond to a semi-positive definite symmetric Gram matrix:

K=

$K(\mathbf{x}_1,\mathbf{x}_1)$	$K(\mathbf{x}_1,\mathbf{x}_2)$	$K(\mathbf{x}_1,\mathbf{x}_3)$	…	$K(\mathbf{x}_1,\mathbf{x}_N)$
$K(\mathbf{x}_2,\mathbf{x}_1)$	$K(\mathbf{x}_2,\mathbf{x}_2)$	$K(\mathbf{x}_2,\mathbf{x}_3)$		$K(\mathbf{x}_2,\mathbf{x}_N)$
…	…	…	…	…
$K(\mathbf{x}_N,\mathbf{x}_1)$	$K(\mathbf{x}_N,\mathbf{x}_2)$	$K(\mathbf{x}_N,\mathbf{x}_3)$	…	$K(\mathbf{x}_N,\mathbf{x}_N)$

Examples of Kernel Functions

- Linear: $K(\mathbf{x_i},\mathbf{x_j}) = \mathbf{x_i}^{\mathbf{T}}\mathbf{x_j}$

- Polynomial of power p: $K(\mathbf{x_i},\mathbf{x_j}) = (1 + \mathbf{x_i}^{\mathbf{T}}\mathbf{x_j})^p$

- Gaussian (radial-basis function network): $K(\mathbf{x_i},\mathbf{x_j}) = e^{-\frac{\|\mathbf{x_i} - \mathbf{x_j}\|^2}{2\sigma^2}}$

- Two-layer perceptron: $K(\mathbf{x_i},\mathbf{x_j}) = \tanh(\beta_0 \mathbf{x_i}^{\mathbf{T}}\mathbf{x_j} + \beta_1)$

Non-linear SVMs Mathematically

- Dual problem formulation:

$$\boxed{\begin{array}{l}\text{Find } \alpha_1 \ldots \alpha_N \text{ such that} \\ \mathbf{Q}(\boldsymbol{\alpha}) = \Sigma\alpha_i - \tfrac{1}{2}\Sigma\Sigma\alpha_i\alpha_j y_i y_j K(\mathbf{x_i}, \mathbf{x_j}) \text{ is maximized and} \\ (1)\ \ \Sigma\alpha_i y_i = 0 \\ (2)\ \alpha_i \geq 0 \text{ for all } \alpha_i\end{array}}$$

- The solution is:

$$\boxed{f(\mathbf{x}) = \Sigma\alpha_i y_i K(\mathbf{x_i}, \mathbf{x_j}) + b}$$

- Optimization techniques for finding α_i's remain the same!

SVM applications

- SVMs were originally proposed by Boser, Guyon and Vapnik in 1992 and gained increasing popularity in late 1990s.
- SVMs are currently among the best performers for a number of classification tasks ranging from text to genomic data.
- SVM techniques have been extended to a number of tasks such as regression [Vapnik *et al.* ’97], principal component analysis [Schölkopf *et al.* ’99], etc.
- Most popular optimization algorithms for SVMs are SMO [Platt ’99] and SVM^{light} [Joachims’ 99], both use *decomposition* to hill-climb over a subset of α_i’s at a time.
- Tuning SVMs remains a black art: selecting a specific kernel and parameters is usually done in a try-and-see manner.

SVM Extensions

- Regression
- Variable Selection
- Boosting
- Density Estimation
- Unsupervised Learning
  - Novelty/Outlier Detection
  - Feature Detection
  - Clustering

Support Vector Machine Resources

- SVM Application List
  - http://www.clopinet.com/isabelle/Projects/SVM/applist.html
- Kernel machines
  - http://www.kernel-machines.org/
- Pattern Classification and Machine Learning
  - http://clopinet.com/isabelle/#projects
- R a GUI language for statistical computing and graphics
  - http://www.r-project.org/
- Kernel Methods for Pattern Analysis – 2004
  - http://www.kernel-methods.net/
- An Introduction to Support Vector Machines (and other kernel-based learning methods)
  - http://www.support-vector.net/
- Kristin P. Bennett web page
  - http://www.rpi.edu/~bennek
- Isabelle Guyon's home page
  - http://clopinet.com/isabelle

Support Vector Machine **References**

- Duda R.O. and Hart P.E.; *Patter Classification and Scene Analysis*. Wiley, 1973.
- T.M. Cover. Geometrical and statistical properties of systems of linear inequalities with applications in pattern recognition. IEEE Transactions on Electronic Computers}, 14:326--334, 1965.
- V.Vapnik and A.Lerner. Pattern recognition using generalized portrait method. Automation and Remote Control}, 24, 1963.
- V.Vapnik and A.Chervonenkis. A note on one class of perceptrons. Automation and Remote Control}, 25, 1964.
- J.K. Anlauf and M.Biehl. The adatron: an adaptive perceptron algorithm. Europhysics Letters, 10:687--692, 1989.
- N.Aronszajn. Theory of reproducing kernels. *Transactions of the American Mathematical Society*, 68:337--404, 1950.
- M.Aizerman, E.Braverman, and L.Rozonoer.Theoretical foundations of the potential function method in pattern recognition learning. *Automation and Remote Control* 25:821--837, 1964.
- O. L. Mangasarian. Linear and nonlinear separation of patterns by linear programming. *Operations Research* , 13:444--452, 1965.
- F. W. Smith. Pattern classifier design by linear programming. *IEEE Transactions on Computers* , C-17:367--372, 1968.
- C.Cortes and V.Vapnik. Support vector networks. Machine Learning, 20:273--297, 1995.V.Vapnik. The Nature of Statistical Learning Theory}. Springer Verlag, 1995.
- V.Vapnik. Statistical Learning Theory}. Wiley, 1998.A.N. Tikhonov and V.Y. Arsenin. Solutions of Ill-posed Problems. W. H. Winston, 1977.
- B.Schoelkopf, C.J.C. Burges, and A.J. Smola, Advances in kernel methods - support vector learning, MIT Press, Cambridge, MA, 1999.
- A.J. Smola, P.Bartlett, B.Schoelkopf, and C.Schuurmans, Advances in large margin classifiers, MIT Press, Cambridge, MA, 1999.