Applied Machine Learning: Algorithms, Practice and Theory

Logistic Regression and SVM

Koby Crammer
Technion – Israel Institute of Technology

Outline

- Naïve Bayes:
  - Generative (joint) probabilistic linear model
- Logistic regression:
  - Discriminative (conditional) probabilistic linear model
  - Log-loss
- Support Vector Machines:
  - Discriminative linear model
  - Hinge-loss
  - Large Margin, Regularization, Kernels
- Models, Inference, Learning

Review of Setting

Formal Setting – Data

- Instances $\mathbf{x} \in \mathcal{X}$
  - Images, Sentences
- Labels $y \in \mathcal{Y} = \{-1 \; ; \; 1\}$
  - Binary
- Statistical Assumption $(X, Y) \sim P$
  - Used for training and evaluation
- I.I.D. Sample $S = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^{m}$

  $(\boldsymbol{x}_i, y_i) \sim P$

Formal Setting - Predictions

- Continuous predictions : $f : \mathcal{X} \rightarrow \mathbb{R}$
  - Label $\text{sign}(f(\mathbf{x}))$
  - Confidence $|f(\mathbf{x})|$
- Notation: $f(\mathbf{x}) = \hat{y}$

Formal Setting - Evaluation

- Loss
  - No. of mistakes $\ell(f(\boldsymbol{x}), y) \in \mathbb{R}_+$
- Expected Loss

$$\mathrm{E}_{(X,Y)\sim P}\left[\ell(f(X), Y)\right]$$

- Averaged Loss

$$\mathrm{E}_{(X,Y)\sim S}\left[\ell(f(X), Y)\right]$$

$$= \frac{1}{m}\sum_{i=1}^{m} \ell(f(\boldsymbol{x}_i), y_i)$$

Empirical Risk Minimization

- Goal: Minimize classification error

$$\mathrm{E}_{(X,Y)\sim P}\left[1_{f(X)\neq Y}\right]$$

- Approximate using Sample:

$$\frac{1}{m}\sum_{i=1}^{m} 1_{f(\boldsymbol{x}_i)\neq y_i}$$

- Bound with “nice” loss

$$\frac{1}{m}\sum_{i=1}^{m} \ell\left(f(\boldsymbol{x}_i), y_i\right)$$

Loss Functions

- Natural Loss:
  - Zero-One loss:

$$\ell\left(f(\boldsymbol{x}), y\right) = \begin{cases} 0 & y = \operatorname{sign}(f(\boldsymbol{x})) \\ 1 & y \neq \operatorname{sign}(f(\boldsymbol{x})) \end{cases}$$

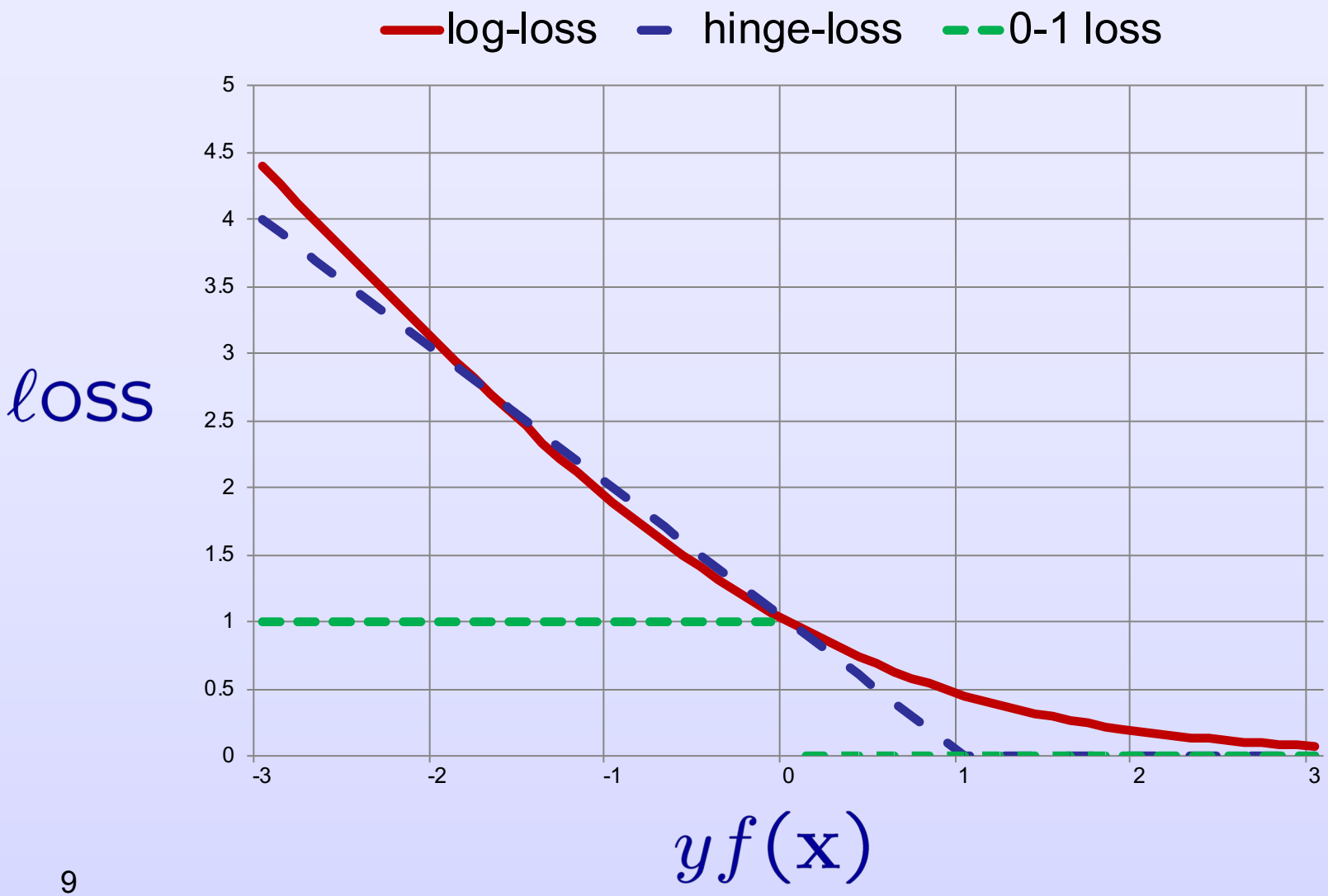
- Real-valued-predictions loss:
  - Hinge loss:

$$\ell\left(f(\boldsymbol{x}), y\right) = \max\left(0, 1 - yf(\boldsymbol{x})\right)$$

  - Log loss (Max Entropy, Boosting)

$$\ell\left(f(\boldsymbol{x}), y\right) = \log\left(1 + \exp\left(-yf(\boldsymbol{x})\right)\right)$$

Loss Functions



Logistic Regression: Model

- Goal:
  - Design a discriminative parametric model for the conditional distribution

$$P(Y|X = \boldsymbol{x}) \in [0, 1]$$

Binary Classification

$$Y \in \{-1, +1\}$$

Vector Representation of a Document

Logistic Regression: Model

- Attempt I: $P(Y = +1|\boldsymbol{x}) = 2\boldsymbol{w} \cdot \boldsymbol{x}$
  - Problem: $P(Y = +1|\boldsymbol{x}) \in [0, 1]$
    $\boldsymbol{w} \cdot \boldsymbol{x} \in (-\infty, \infty)$
- Attempt II: $\log P(Y = +1|\boldsymbol{x}) = 2\boldsymbol{w} \cdot \boldsymbol{x}$
  - Problem: $\log P(Y = +1|\boldsymbol{x}) \in (-\infty, 0]$
    $\boldsymbol{w} \cdot \boldsymbol{x} \in (-\infty, \infty)$

Logistic Regression

Logistic Regression: Model

- Attempt III:

$$\log \frac{P(Y=+1|\boldsymbol{x})}{P(Y=-1|\boldsymbol{x})} = \log \frac{P(Y=+1|\boldsymbol{x})}{1-P(Y=+1|\boldsymbol{x})}$$

$$= 2\boldsymbol{w} \cdot \boldsymbol{x}$$

$\in (-\infty, \infty)$

$\in [0, \infty)$

Logistic Regression: Model

- Model:

$$\log \frac{P(Y=+1|\boldsymbol{x})}{1-P(Y=+1|\boldsymbol{x})} = 2\boldsymbol{w}\cdot\boldsymbol{x}$$

- Probability of label :

$$P(Y=+1|\boldsymbol{x}) = \frac{e^{2\boldsymbol{w}\cdot\boldsymbol{x}}}{1+e^{2\boldsymbol{w}\cdot\boldsymbol{x}}} = \frac{e^{\boldsymbol{w}\cdot\boldsymbol{x}}}{e^{-\boldsymbol{w}\cdot\boldsymbol{x}}+e^{\boldsymbol{w}\cdot\boldsymbol{x}}}$$

- Compact form:

$$P(y|\boldsymbol{x}) = \frac{e^{y\boldsymbol{w}\cdot\boldsymbol{x}}}{e^{-\boldsymbol{w}\cdot\boldsymbol{x}}+e^{\boldsymbol{w}\cdot\boldsymbol{x}}}$$

Logistic Regression: Prediction

- Pretend it is the correct model, use Bayes-optimal

$$\hat{y} = \arg\max_{z} P(Y = z|\boldsymbol{x})$$

- Maximum a-posteriori

Logistic Regression: Prediction

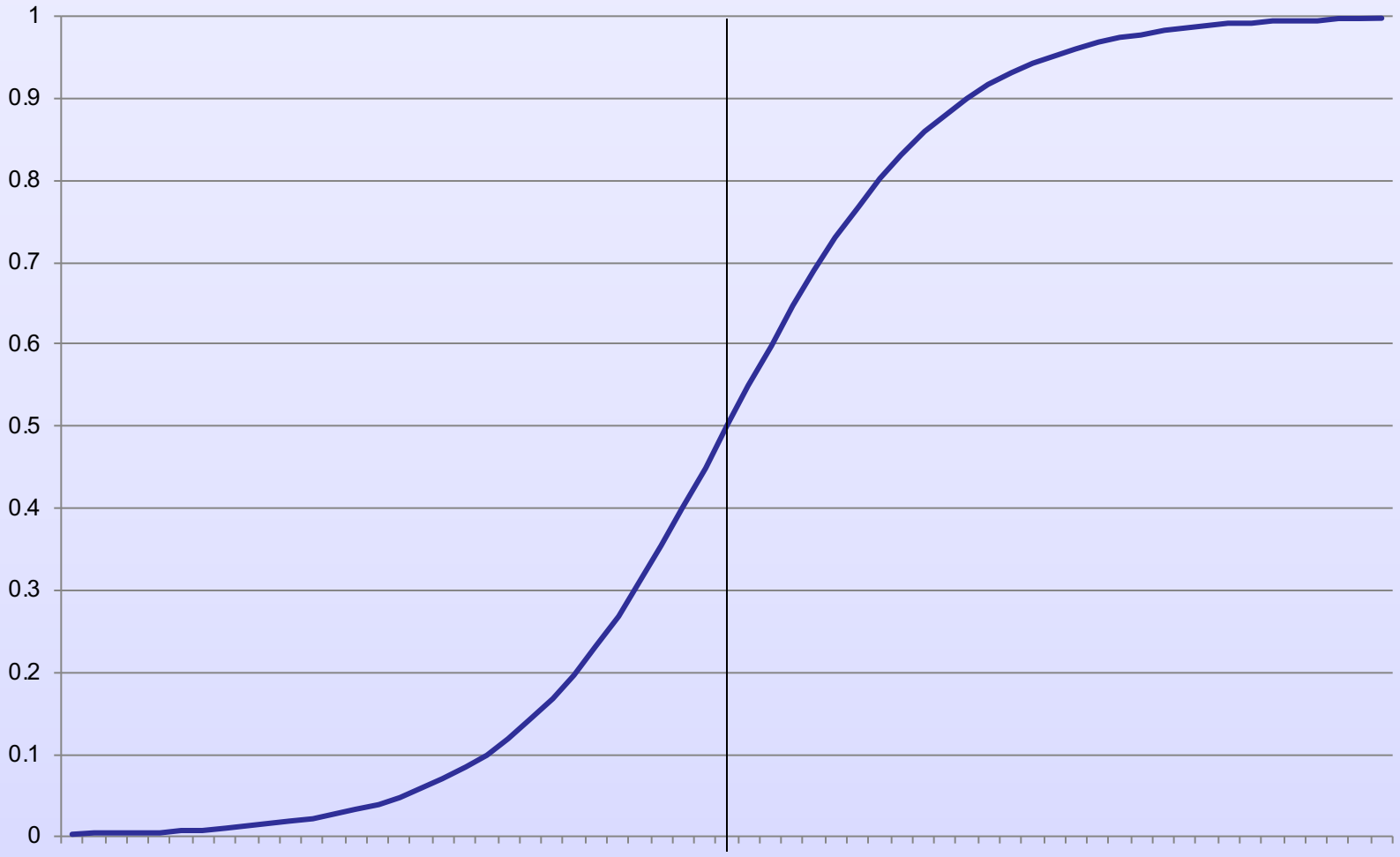
- Predict class +1 iff

Maximum a-posteriori prediction is linear classification

$$\hat{y} = \text{sign}\left(\boldsymbol{w} \cdot \boldsymbol{x}\right)$$

$$\boldsymbol{w} \cdot \boldsymbol{x} > 0$$

Logistic Regression: Prediction



Logistic Regression: Learning

- Given training set

$$S = \{(\boldsymbol{x}_t, y_t)\}_{i=t}^{m} \qquad \boldsymbol{x}_t \in \mathbb{R}^Q \qquad y_t \in \{\pm 1\}$$

- Find a good parameters-vector: $\boldsymbol{w} \in \mathbb{R}^Q$

Hard: not continuous, nor convex

- How we define what is “good”?
  - Minimize training error

$$\frac{1}{m} \sum_{t=1}^{m} 1_{y_t(\boldsymbol{w} \cdot \boldsymbol{x}_t) > 0}$$

Logistic Regression: Learning

- We have a probabilistic model, use it!
- Maximize labels-probability of sample

$$\max_{\boldsymbol{w}} P\left(y_1 \dots y_m | \boldsymbol{x}_1 \dots \boldsymbol{x}_m\right) = \prod_t P\left(y_t | \boldsymbol{x}_t\right)$$

- Use monotonic function “log:

$$\log \prod_t P\left(y_t | \boldsymbol{x}_t\right) = \sum_t \log P\left(y_t | \boldsymbol{x}_t\right)$$

Logistic Regression: Learning

- Substitute model: $$P(y|\boldsymbol{x}) = \frac{e^{y\boldsymbol{w}\cdot\boldsymbol{x}}}{e^{-\boldsymbol{w}\cdot\boldsymbol{x}} + e^{\boldsymbol{w}\cdot\boldsymbol{x}}}$$

$$\sum_t \log P\left(y_t|\boldsymbol{x}_t\right)$$

Logistic Regression: Learning

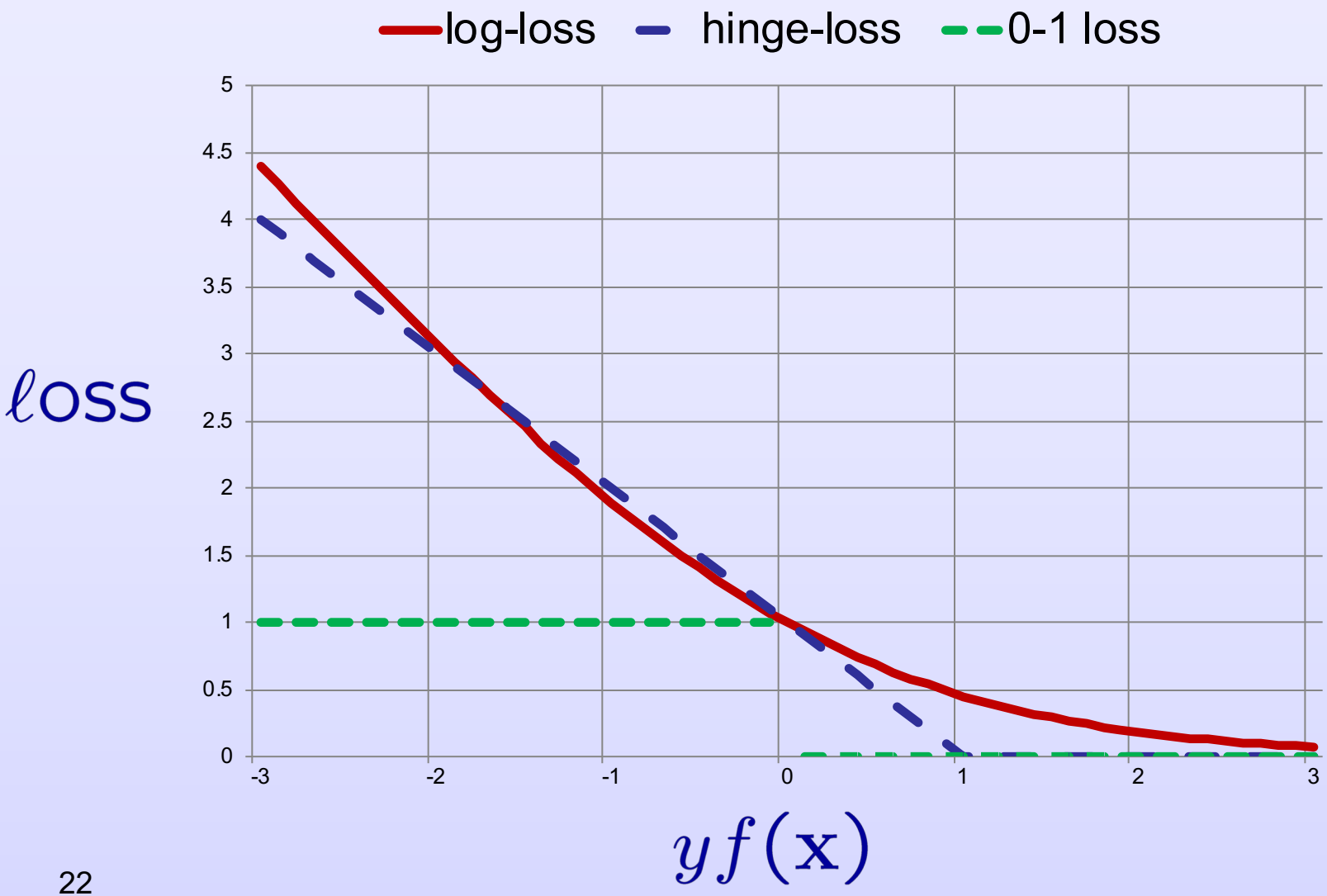
- Substitute model:

Logistic Regression is minimizing
a sum of the log loss

$$\sum_t \log\left(e^{-2y_t \boldsymbol{w}\cdot\boldsymbol{x}_t} + 1\right)$$

$$= -\sum_t \log\left(e^{-2y_t \boldsymbol{w}\cdot\boldsymbol{x}_t} + 1\right)$$

Loss Functions



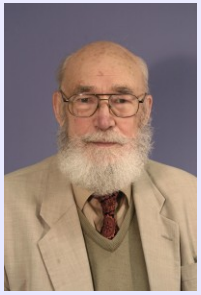
Logistic Regression: Comments

- Equivalent to maximum conditional-entropy
- Many algorithms to solve:
  - e.g. gradient descent
- Easley generalized to multi-class problems

History

- Regression
  - Adrien-Marie Legendre, 1805
  - Johann Carl Friedrich Gauss, 1809
  - John Nelder and Robert Wedderburn, 1972





- Logistic Regression
  - Pierre Francois Verhulst, 1838
  - Pearl & Reed, 1920
  - Joseph Berkson, 1944
  - E.T. Jaynes, ~1950



Linear Models

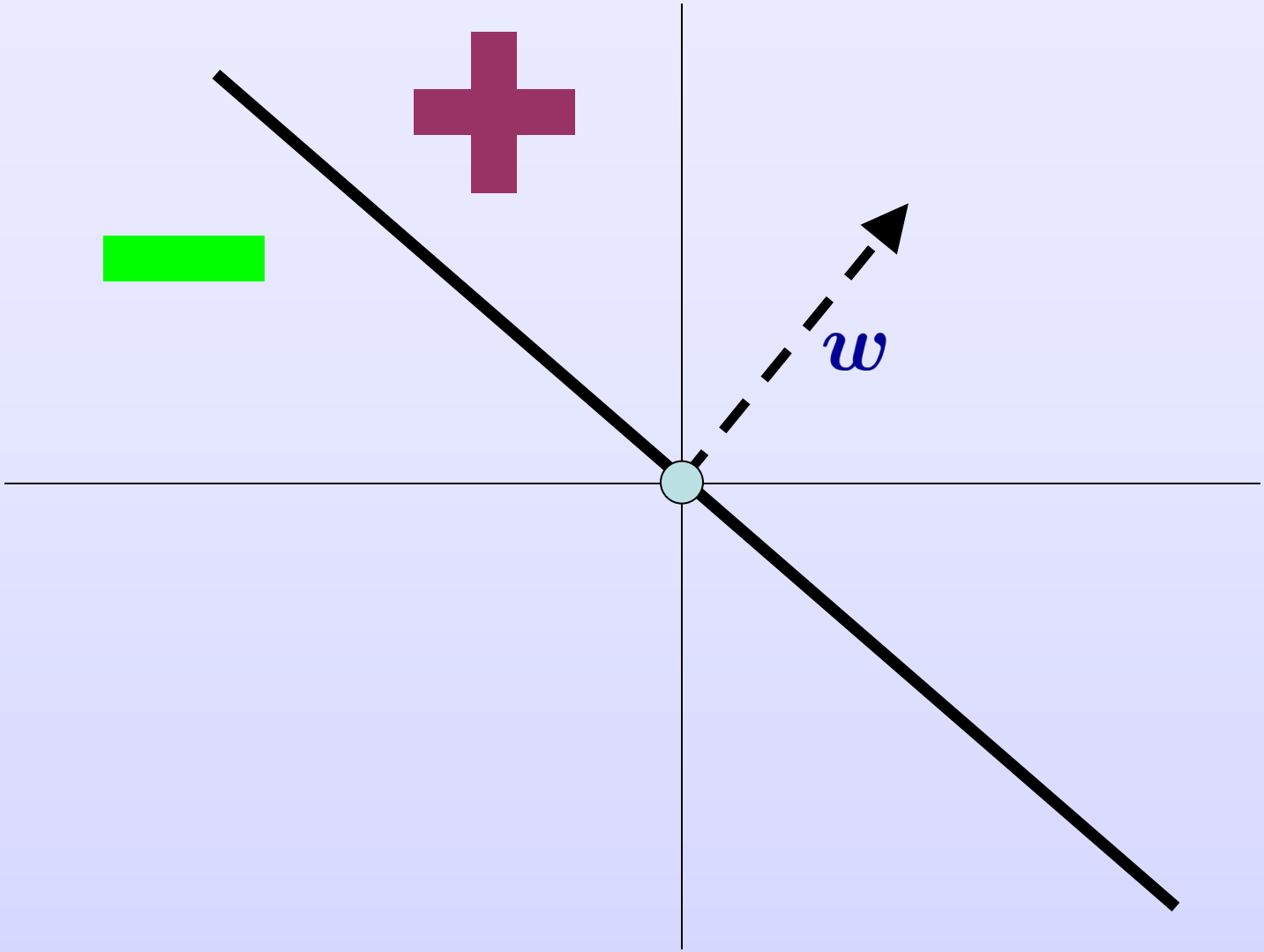
Why compute probability

if

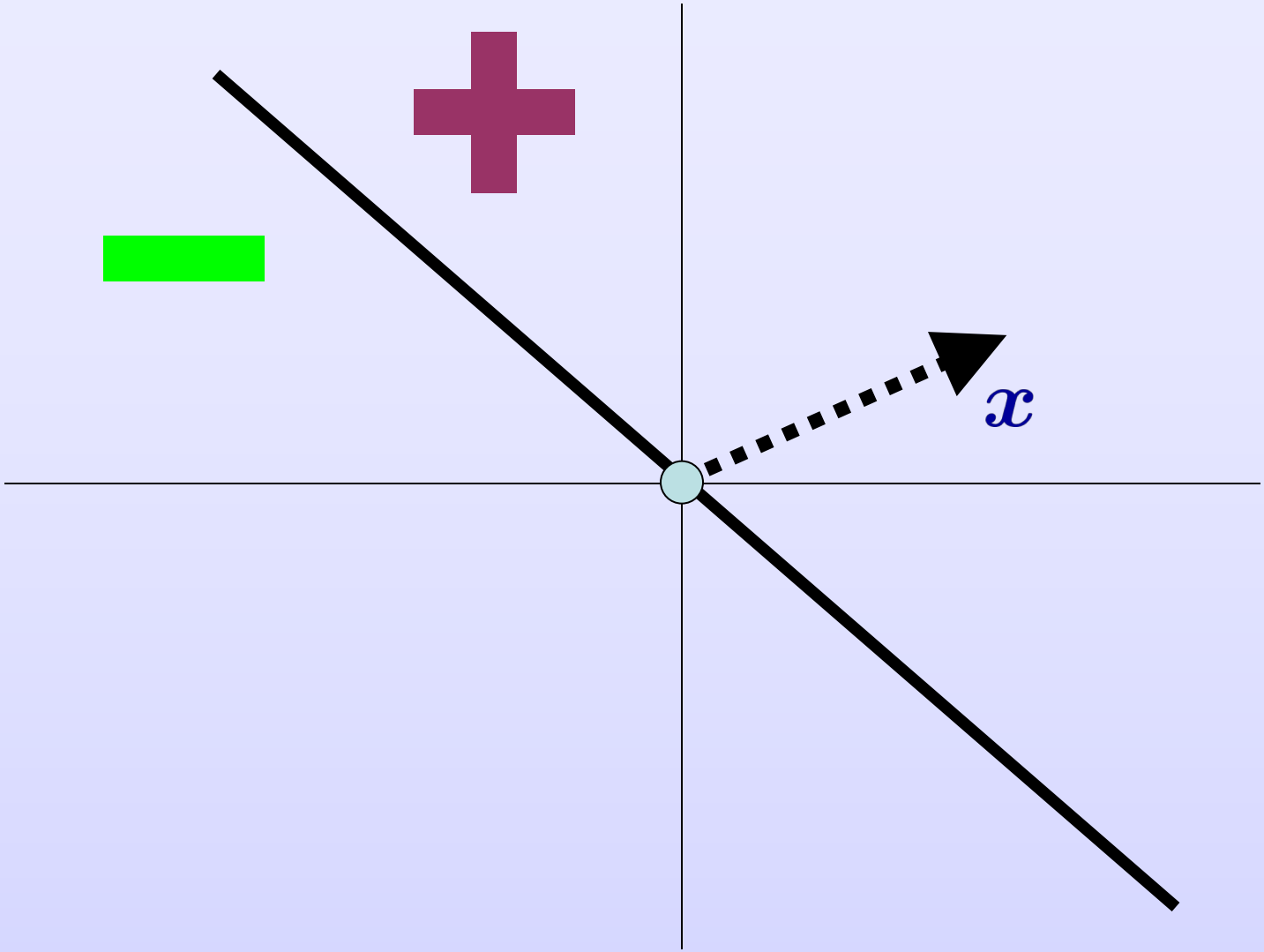
only want to make predictions?

- Do not solve an estimation problem of interest by solving a more general (harder) problem as an intermediate step
- Only care about decision boundaries

Linear Classifiers



Linear Classifiers



Linear Classifiers

- Prediction :

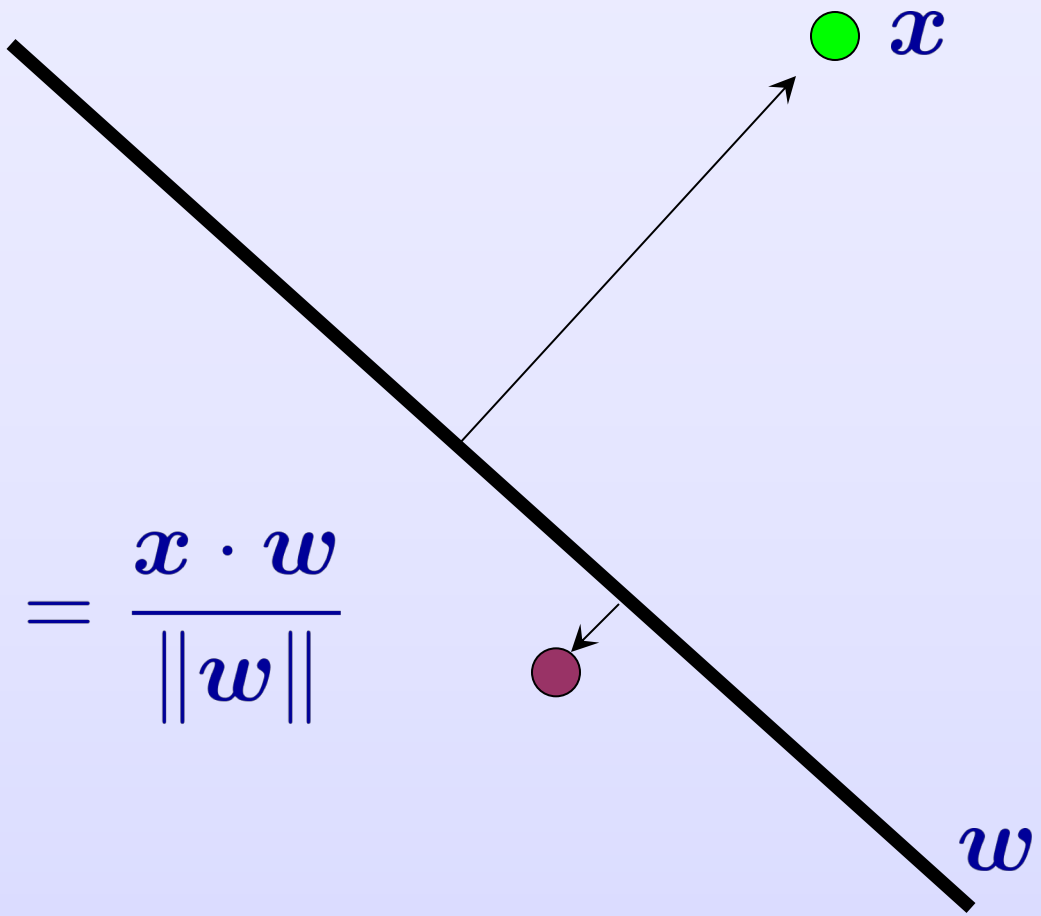
$$\widehat{y} = \text{sign}(f(\mathbf{x}))$$

$$= \text{sign}(\mathbf{w} \cdot \mathbf{x})$$

- Confidence in prediction:

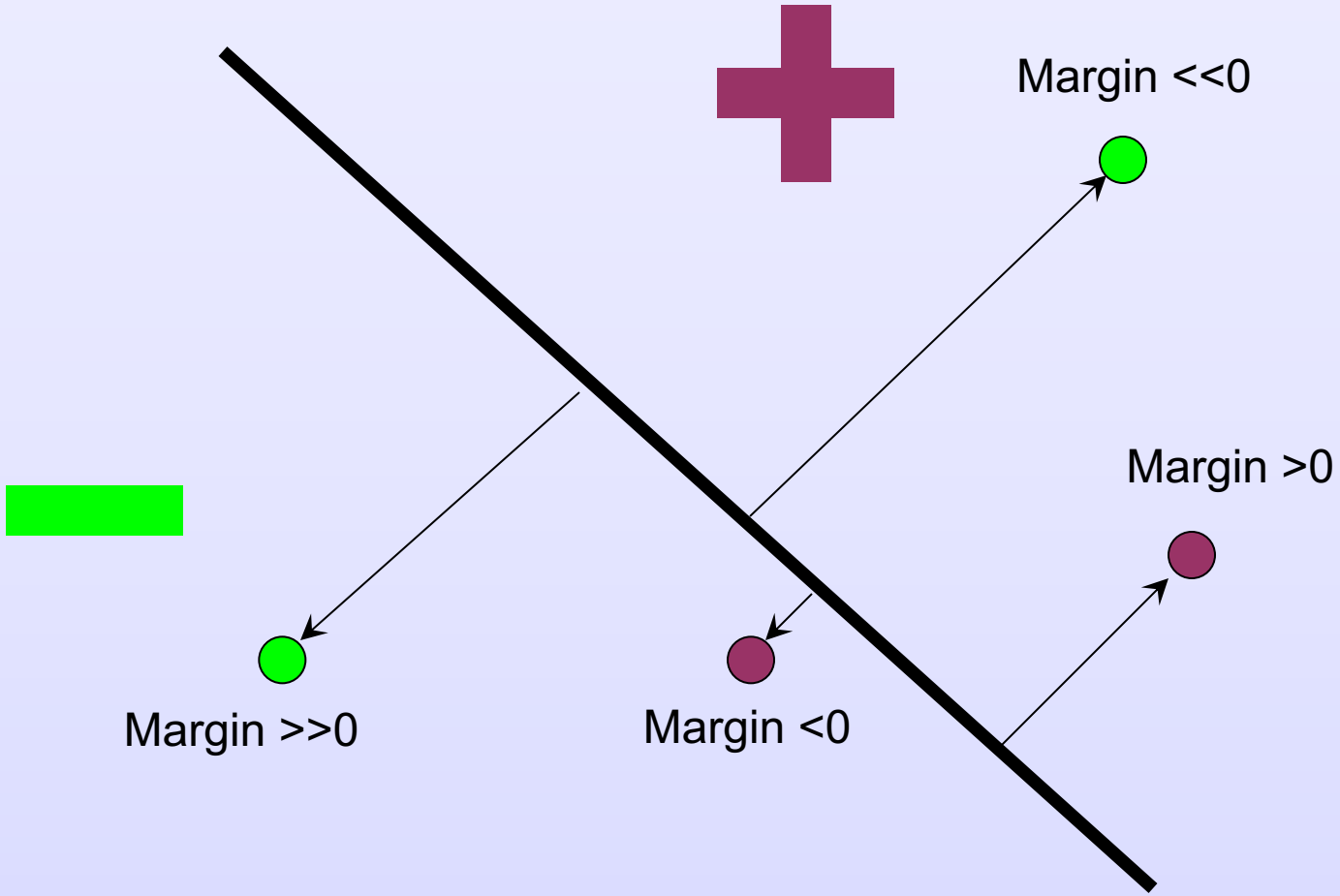
$$|f(\mathbf{x})| = |\mathbf{w} \cdot \mathbf{x}|$$

Signed Distance



$$d(\boldsymbol{x}; \boldsymbol{w}) = \frac{\boldsymbol{x} \cdot \boldsymbol{w}}{\|\boldsymbol{w}\|}$$

Margin

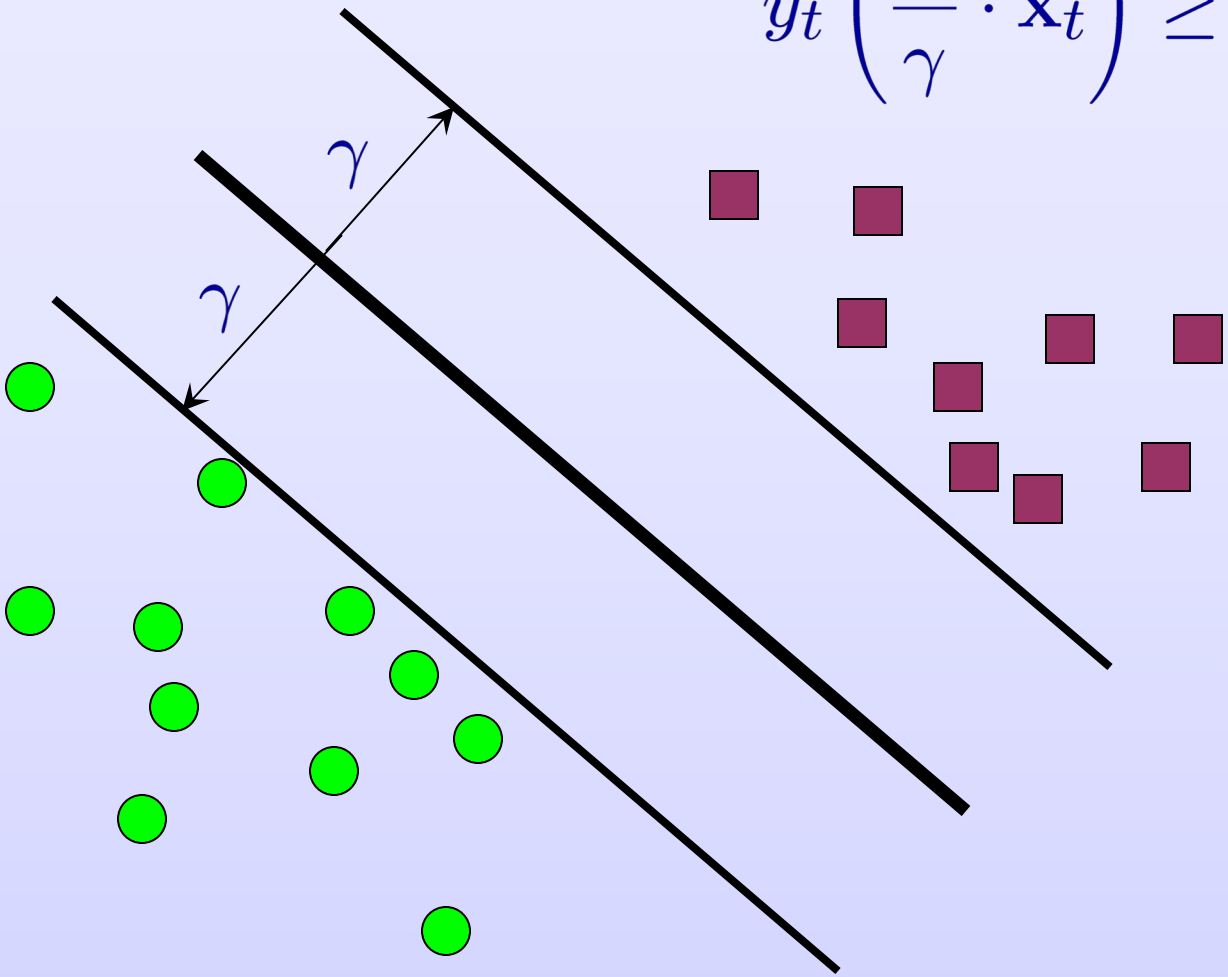


$$d\left(y\boldsymbol{x};\boldsymbol{w}\right)=\frac{y\boldsymbol{x}\cdot\boldsymbol{w}}{\|\boldsymbol{w}\|}$$

Separable Set

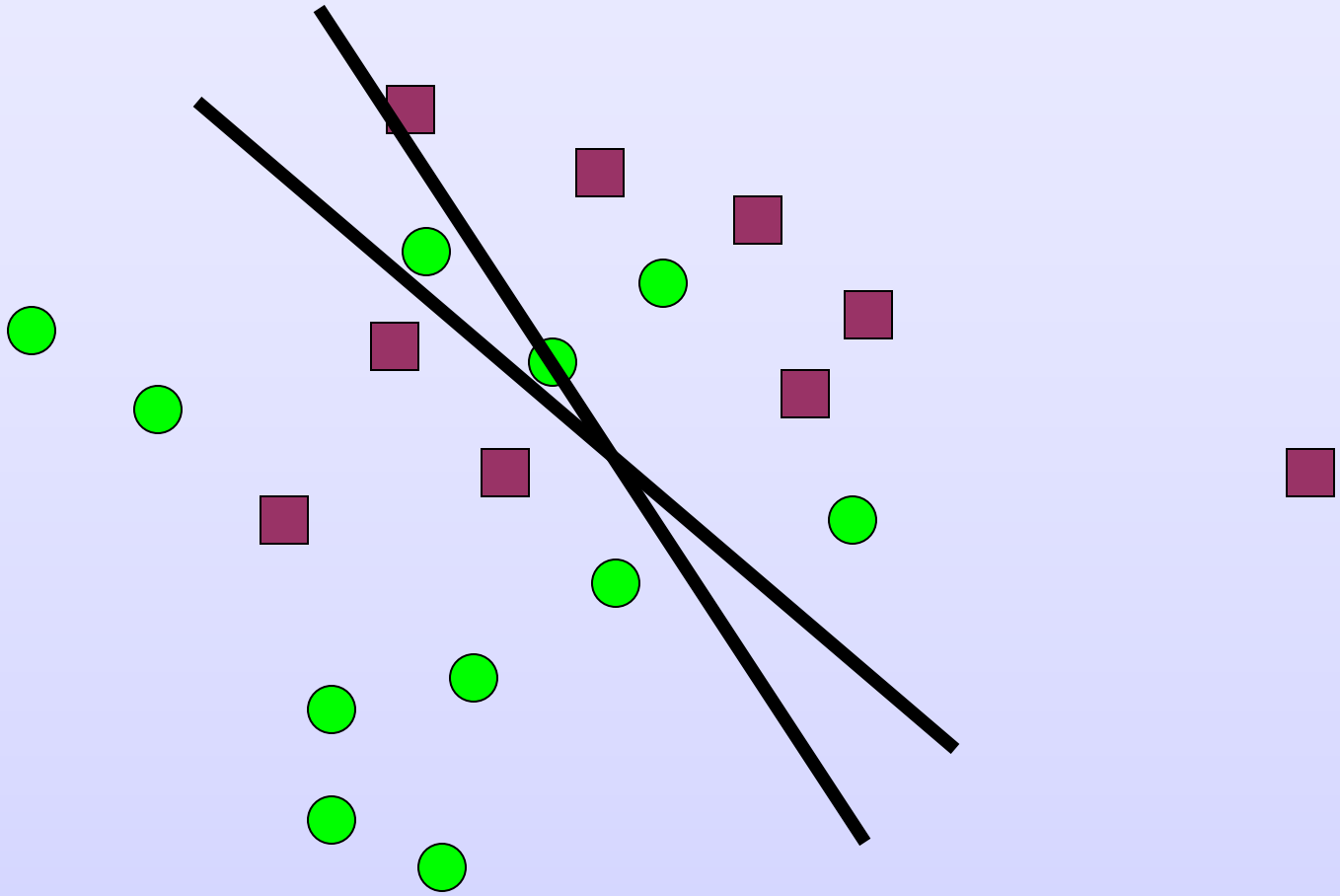
$\exists \mathbf{w}$ such that $\forall t$ $y_t(\mathbf{w} \cdot \mathbf{x}_t) \geq \gamma > 0$

$$y_t \left(\frac{\mathbf{w}}{\gamma} \cdot \mathbf{x}_t \right) \geq 1$$

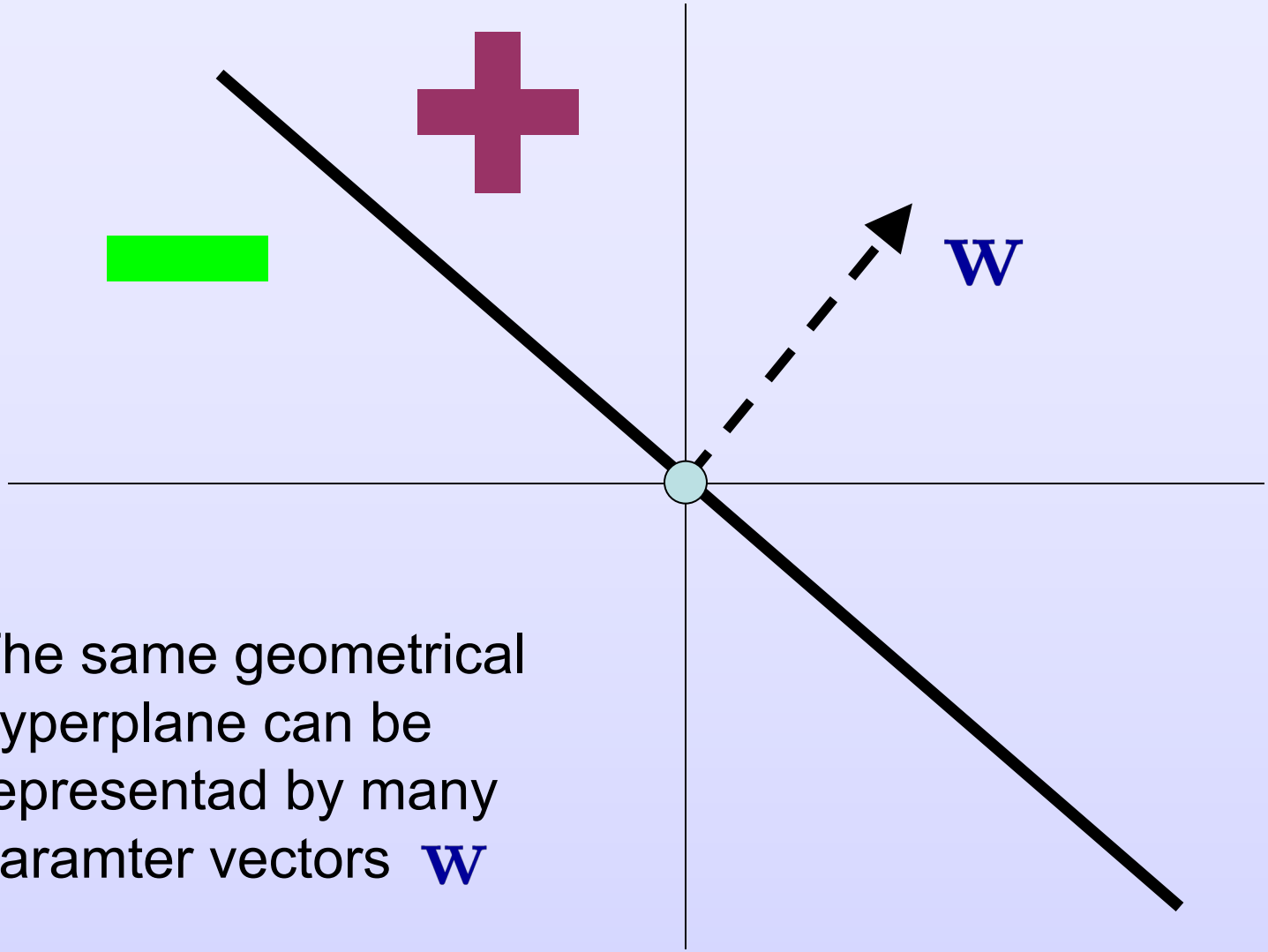


Inseparable Sets

$\forall \mathbf{w}$ there exists t such that $y_t(\mathbf{w} \cdot \mathbf{x}_t) < 0$

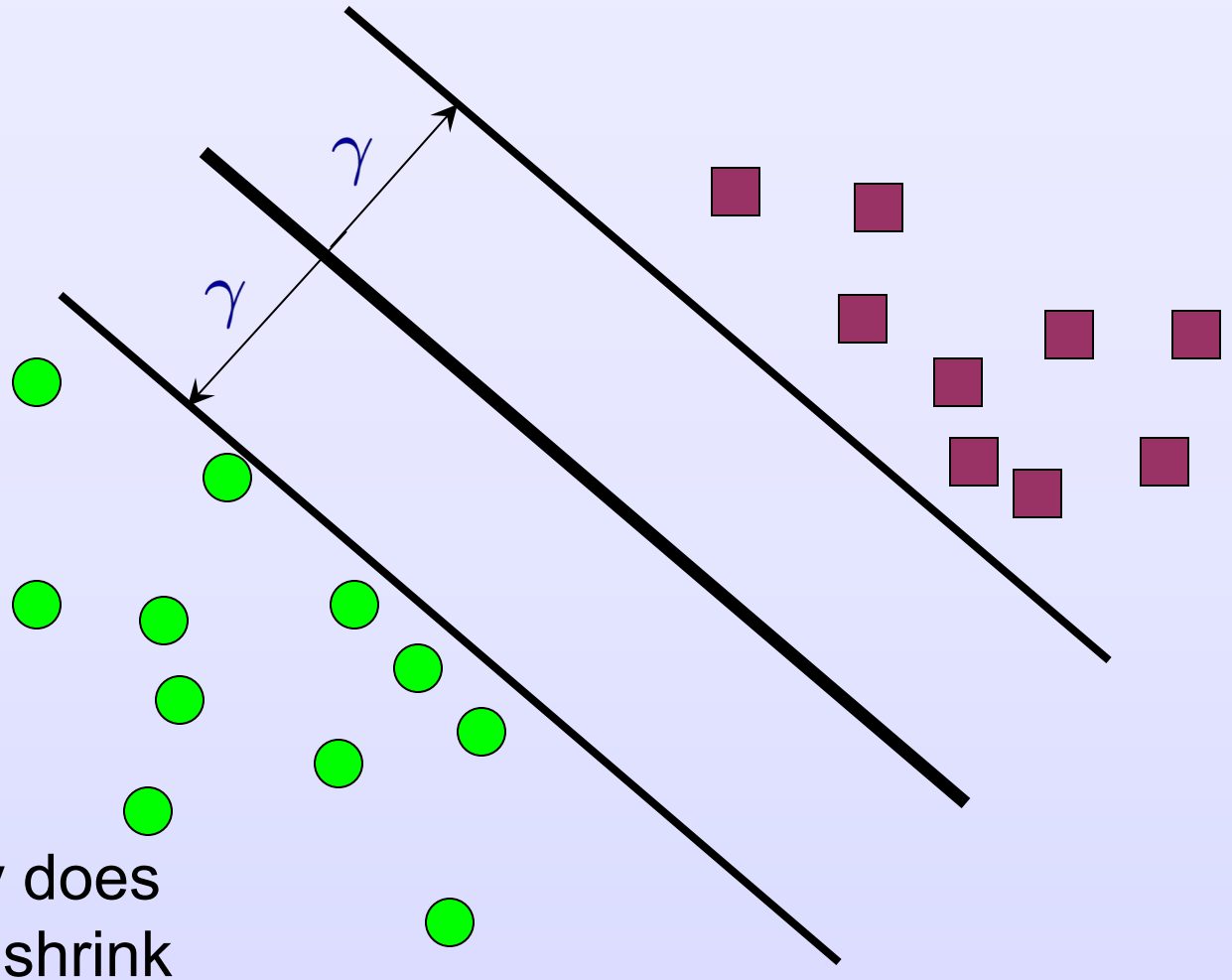


Degree of Freadom - I



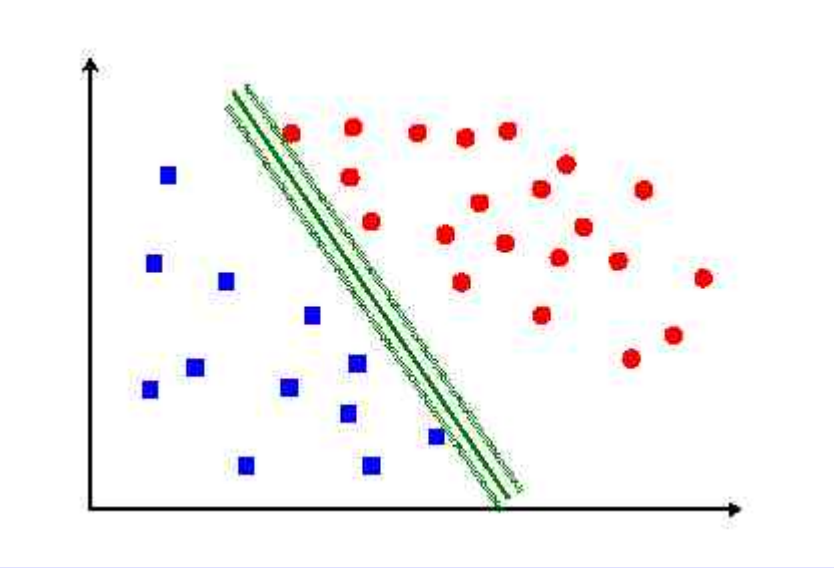
The same geometrical hyperplane can be representad by many paramter vectors $\mathbf{w}$

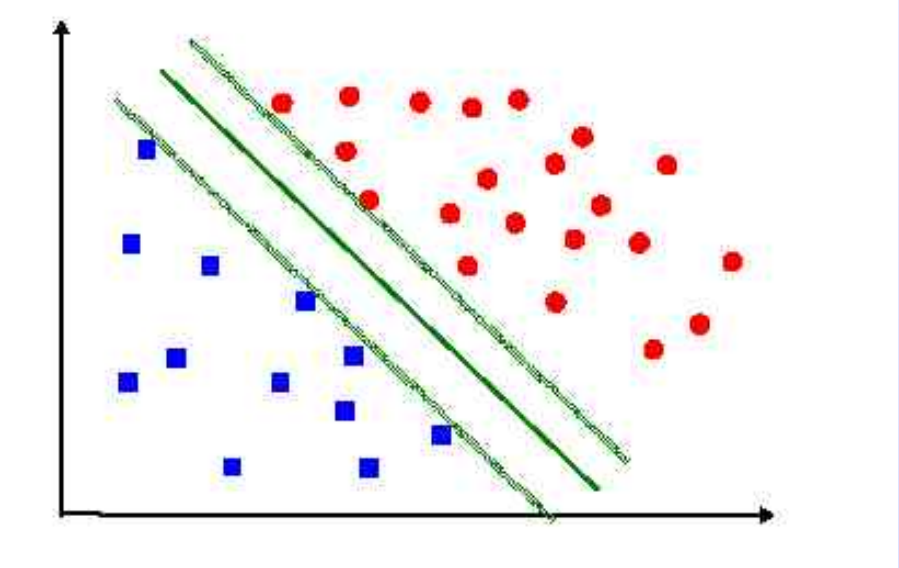
Degree of Freadom - II



Problem difficulty does not change if we shrink or expand the input space

Large Margin Classification





Support Vector Machines

Learning to maximize the margin

Support Vector Machines

- Assume input data is separable
- Find hyperplane with maximal (sample-)margin
- Equivalent, find hyperplane such that
  - Closest point to the hyperplane …
  - … will be as far as possible from it

$$\boldsymbol{w}^* = \arg\max_{\boldsymbol{w}} \left(\min_{t=1} d\left(y_t \boldsymbol{x}_t; \boldsymbol{w}\right) \right)$$

$$= \arg\max_{\boldsymbol{w}} \left(\min_{t=1} \frac{y_t \boldsymbol{x}_t \cdot \boldsymbol{w}}{\|\boldsymbol{w}\|} \right)$$

Support Vector Machines

$$\boldsymbol{w}^* = \arg\max_{\boldsymbol{w}} \left(\min_{t=1} \frac{y_t \boldsymbol{x}_t \cdot \boldsymbol{w}}{\|\boldsymbol{w}\|} \right)$$

$$= \arg\max_{\boldsymbol{w}} \frac{1}{\|\boldsymbol{w}\|} \left(\min_{t=1} y_t \boldsymbol{x}_t \cdot \boldsymbol{w} \right)$$

- Use first degree of freedom

$$\boldsymbol{w}^* = \arg\max_{\boldsymbol{w}} \frac{1}{\|\boldsymbol{w}\|}$$

$$\text{s.t.} \ \min_{t=1} y_t \boldsymbol{x}_t \cdot \boldsymbol{w} \geq 1$$

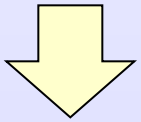
Support Vector Machines

- Algebra

Primal Separable SVM

$$\boldsymbol{w}^* = \arg\max_{\boldsymbol{w}} \frac{1}{\|\boldsymbol{w}\|}$$

$$\text{s.t.} \ \min_{t=1} y_t \boldsymbol{x}_t \cdot \boldsymbol{w} \geq 1$$



$$\boldsymbol{w}^* = \arg\min_{\boldsymbol{w}} \frac{1}{2}\|\boldsymbol{w}\|^2$$

$$\text{s.t.} \ \ y_t \boldsymbol{x}_t \cdot \boldsymbol{w} \geq 1 \ \ \forall t$$

Lagrange Theory

- Problem:

$$\min_x \; f(x)$$

$$\text{s.t.} \; g_t(x) \leq 0 \quad \forall t$$

- Min-Max game:

$$\max_\alpha \min_x f(x) + \sum_t \alpha_t g_t(x)$$

$$\text{s.t.} \quad \alpha_t \geq 0 \quad \forall t$$

- KKT conditions:

$$\alpha_t^* g_t(x^*) = 0$$

Lagrangian

$$\max_{\alpha} \min_{\boldsymbol{w}} \; \frac{1}{2}\|\boldsymbol{w}\|^2 + \sum_t \alpha_t \left(1 - y_t \boldsymbol{x}_t \cdot \boldsymbol{w}\right)$$

$$\text{s.t. } \alpha_t \geq 0 \;\; \forall t$$

- Taking the derivative with respect to $\boldsymbol{w}$

$$\frac{\partial}{\partial \boldsymbol{w}} \rightarrow \boldsymbol{w} - \sum_t \alpha_t y_t \boldsymbol{x}_t$$

- Optimal solution

$$\boldsymbol{w} = \sum_t \alpha_t y_t \boldsymbol{x}_t$$

Support Vector Machine

- Optimal Solution:

$$\boldsymbol{w} = \sum_t \alpha_t y_t \boldsymbol{x}_t$$

- KKT conditions:

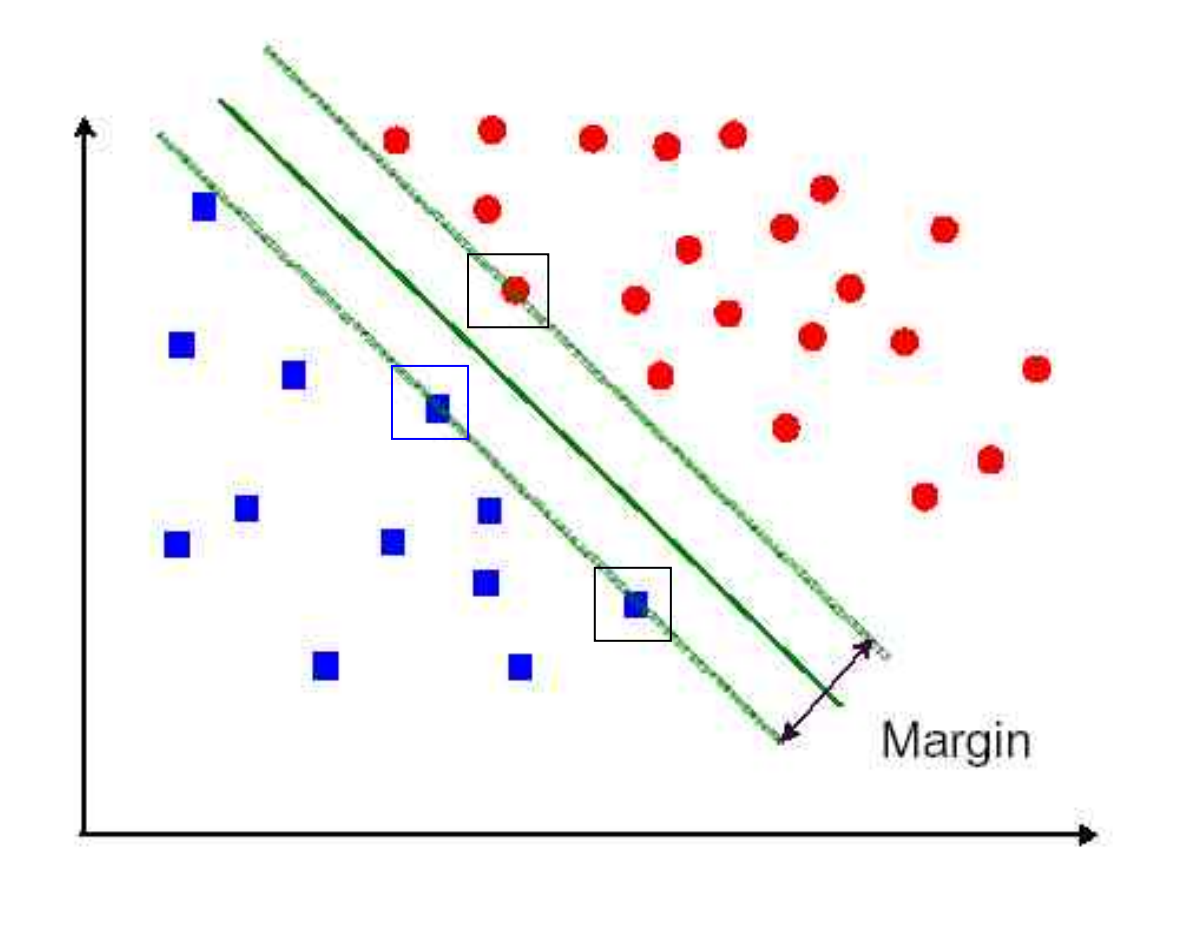
$$\alpha_t(1 - y_t \boldsymbol{w} \cdot \boldsymbol{x}_t) = 0$$

- For a specific example:

$$1 < y_t \boldsymbol{w} \cdot \boldsymbol{x}_t \quad \Rightarrow \quad \alpha_t = 0$$

- Only examples that lie on the boundary affect the classifier

Support Vectors



Dual Formulation

- Rewriting the Lagrangian

$$\frac{1}{2}\|\boldsymbol{w}\|^2 + \sum_t \alpha_t \left(1 - y_t \boldsymbol{x}_t \cdot \boldsymbol{w}\right)$$

$$= -\frac{1}{2}\|\boldsymbol{w}\|^2 + \sum_t \alpha_t - \sum_t \alpha_t y_t \boldsymbol{x}_t \cdot \boldsymbol{w} + \|\boldsymbol{w}\|^2$$

- Substituting back in the Lagrangian

$$\boldsymbol{w} = \sum_t \alpha_t y_t \boldsymbol{x}_t$$

Dual Formulation

- Rewriting the Lagrangian

$$\frac{1}{2}\|\boldsymbol{w}\|^2 + \sum_t \alpha_t \left(1 - y_t \boldsymbol{x}_t \cdot \boldsymbol{w}\right)$$

$$= -\frac{1}{2}\|\boldsymbol{w}\|^2 + \sum_t \alpha_t - \boldsymbol{w} \cdot \boldsymbol{w} + \|\boldsymbol{w}\|^2$$

- Substituting back in the Lagrangian

$$\boldsymbol{w} = \sum_t \alpha_t y_t \boldsymbol{x}_t$$

Dual Formulation

- Rewriting the Lagrangian

$$\frac{1}{2}\|\boldsymbol{w}\|^2 + \sum_t \alpha_t \left(1 - y_t \boldsymbol{x}_t \cdot \boldsymbol{w}\right)$$

$$= -\frac{1}{2}\|\boldsymbol{w}\|^2 + \sum_t \alpha_t$$

- Substituting back in the Lagrangian

$$\boldsymbol{w} = \sum_t \alpha_t y_t \boldsymbol{x}_t$$

Dual Formulation

- Rewriting the Lagrangian

$$\frac{1}{2}\|\boldsymbol{w}\|^2 + \sum_t \alpha_t \left(1 - y_t \boldsymbol{x}_t \cdot \boldsymbol{w}\right)$$

$$= -\frac{1}{2}\sum_{t,s} \alpha_t \alpha_s y_t y_s \left(\boldsymbol{x}_t \cdot \boldsymbol{x}_s\right) + \sum_t \alpha_t$$

- Substituting back in the Lagrangian

$$\boldsymbol{w} = \sum_t \alpha_t y_t \boldsymbol{x}_t$$

Dual Formulations

From features to kernels

Dual Formulation

$$\max_{\alpha} -\frac{1}{2}\sum_{t,s}\alpha_t\alpha_s y_t y_s\left(\boldsymbol{x}_t\cdot\boldsymbol{x}_s\right)+\sum_t\alpha_t$$

$$\text{s.t. } \alpha_t\geq 0 \quad \forall t$$

- Making Predictions:

$$\boldsymbol{w}=\sum_t\alpha_t y_t\boldsymbol{x}_t$$

$$f(\boldsymbol{x})=\text{sign}\left(\boldsymbol{w}\cdot\boldsymbol{x}\right)$$

$$=\text{sign}\left(\sum_t\alpha_t y_t\left(\boldsymbol{x}\cdot\boldsymbol{x}_t\right)\right)$$

Inner-Product

$$\max_{\alpha} -\frac{1}{2}\sum_{t,s}\alpha_t\alpha_s y_t y_s\left(\boldsymbol{x}_t\cdot\boldsymbol{x}_s\right)+\sum_t\alpha_t$$

$$\text{s.t. } \alpha_t\geq 0 \quad \forall t$$

$$f(\boldsymbol{x})=\text{sign}\left(\sum_t\alpha_t y_t\left(\boldsymbol{x}\cdot\boldsymbol{x}_t\right)\right)$$

- Learning and Inferring/predicting can be computed via inner-products between inputs
- No need to know inputs explicitly

Feature Mapping

- Use feature mapping

$$\Phi : \mathbb{R}^d \rightarrow \mathbb{R}^D \quad D >> d$$

- Learning and inference become:

$$\max_{\alpha} -\frac{1}{2}\sum_{t,s}\alpha_t\alpha_s y_t y_s \left(\Phi(\boldsymbol{x}_t)\cdot\Phi(\boldsymbol{x}_s)\right) + \sum_t \alpha_t$$

$$\text{s.t. } \alpha_t \geq 0 \quad \forall t$$

$$f(\boldsymbol{x}) = \text{sign}\left(\sum_t \alpha_t y_t \left(\Phi(\boldsymbol{x})\cdot\Phi(\boldsymbol{x}_t)\right)\right)$$

From Features to Kernels

- Problem:
  - Computing features can infeasble

$$\Phi : \mathbb{R}^d \rightarrow \mathbb{R}^D \quad D >> d$$

- Wish:
  - Efficiently compute the inner-product directly

$$K(\boldsymbol{x}, \boldsymbol{z}) = (\Phi(\boldsymbol{x}) \cdot \Phi(\boldsymbol{z}))$$

From Features to Kernels

- Learning:

$$\max_{\alpha} -\frac{1}{2}\sum_{t,s}\alpha_t\alpha_s y_t y_s K\left(\boldsymbol{x}_t,\boldsymbol{x}_s\right)+\sum_t \alpha_t$$

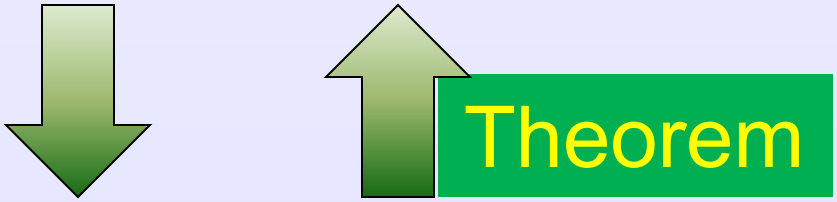
$$\text{s.t. } \alpha_t \geq 0 \quad \forall t$$

- Inference:

$$f(\boldsymbol{x}) = \text{sign}\left(\sum_t \alpha_t y_t K\left(\boldsymbol{x},\boldsymbol{x}_t\right)\right)$$

Kernels

- Let $K(\boldsymbol{x}, \boldsymbol{z}) = (\Phi(\boldsymbol{x}) \cdot \Phi(\boldsymbol{z}))$



- Properties:
  - Symmetric $K(\boldsymbol{x}, \boldsymbol{z}) = K(\boldsymbol{z}, \boldsymbol{x})$
  - Given a finite set of inputs $\boldsymbol{x}_1 \ldots \boldsymbol{x}_n$
    define a matrix $\kappa_{s,t} = K(\boldsymbol{x}_t, \boldsymbol{x}_s)$
    then the matrix κ is positive semi-definite

Example

- Consider 2-d inputs, define:

$$\Phi(x_1, x_2) = \left(x_1^2, x_2^2, \sqrt{2}x_1x_2\right)$$

- Then:

$$\Phi(x_1, x_2) \cdot \Phi(z_1, z_2)$$

$$= \left(x_1^2, x_2^2, \sqrt{2}x_1x_2\right) \cdot \left(z_1^2, z_2^2, \sqrt{2}z_1z_2\right)$$

$$= x_1^2z_1^2 + x_2^2z_2^2 + 2x_1x_2z_1z_2$$

$$= (x_1z_1 + x_2z_2)^2$$

$$= (\boldsymbol{x} \cdot \boldsymbol{z})^2$$

Example

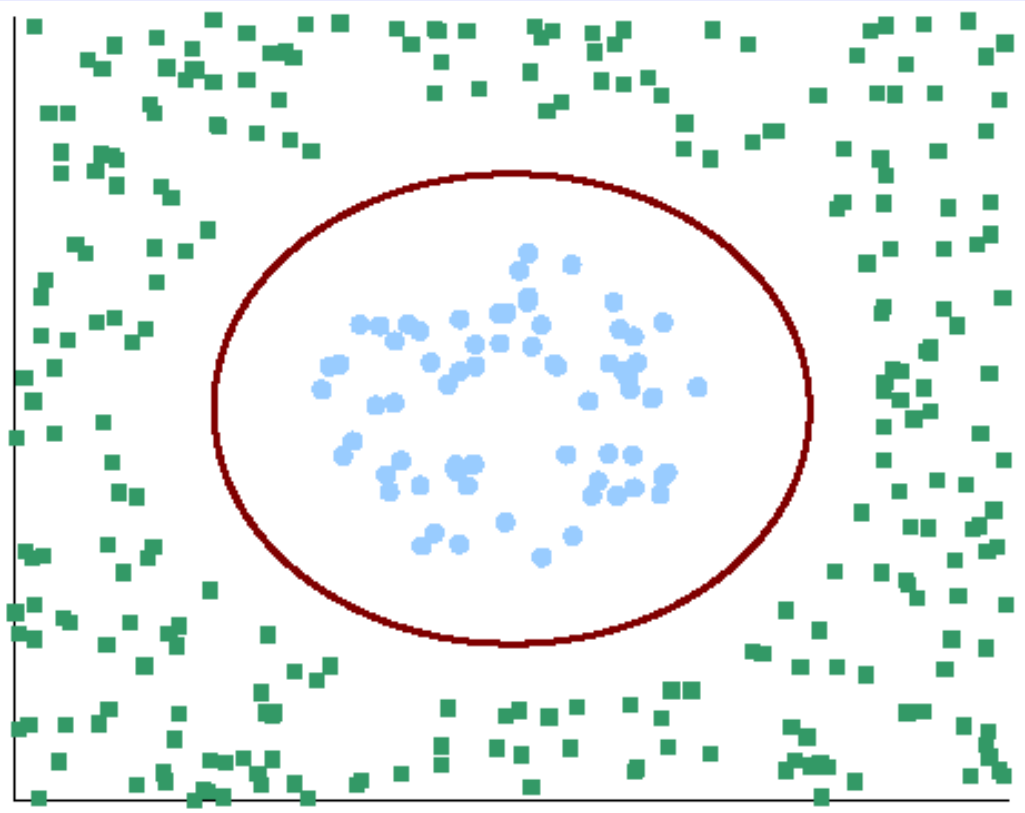
- Consider 2-d inputs, define:

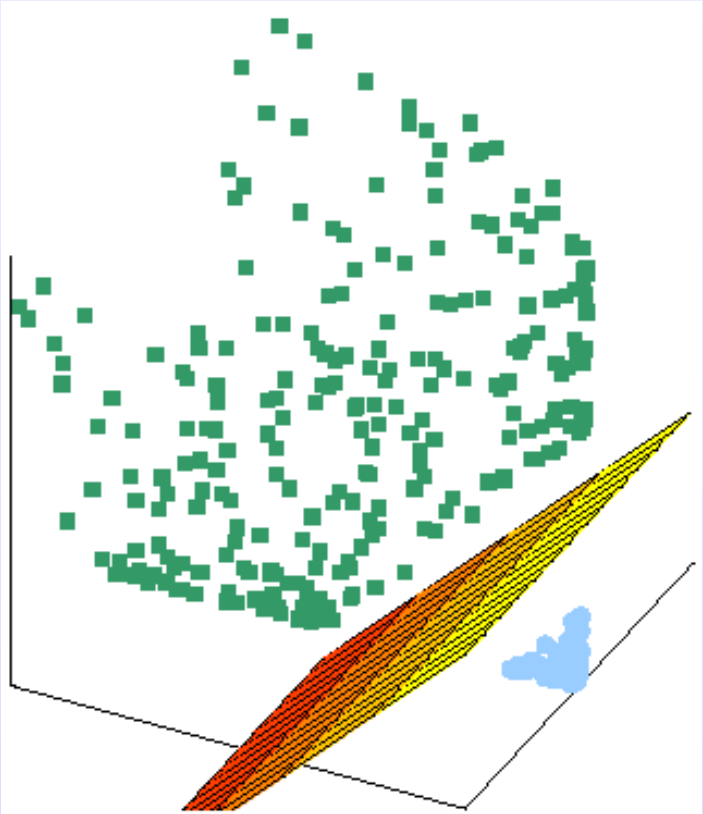
$$\Phi(x_1, x_2) = \left(x_1^2, x_2^2, \sqrt{2}x_1 x_2\right)$$

- Then:

$$\begin{aligned}\Phi(x_1, x_2) \cdot \Phi(z_1, z_2) &= (\boldsymbol{x} \cdot \boldsymbol{z})^2 \\ &= K(\boldsymbol{x}, \boldsymbol{z})\end{aligned}$$

Kernels





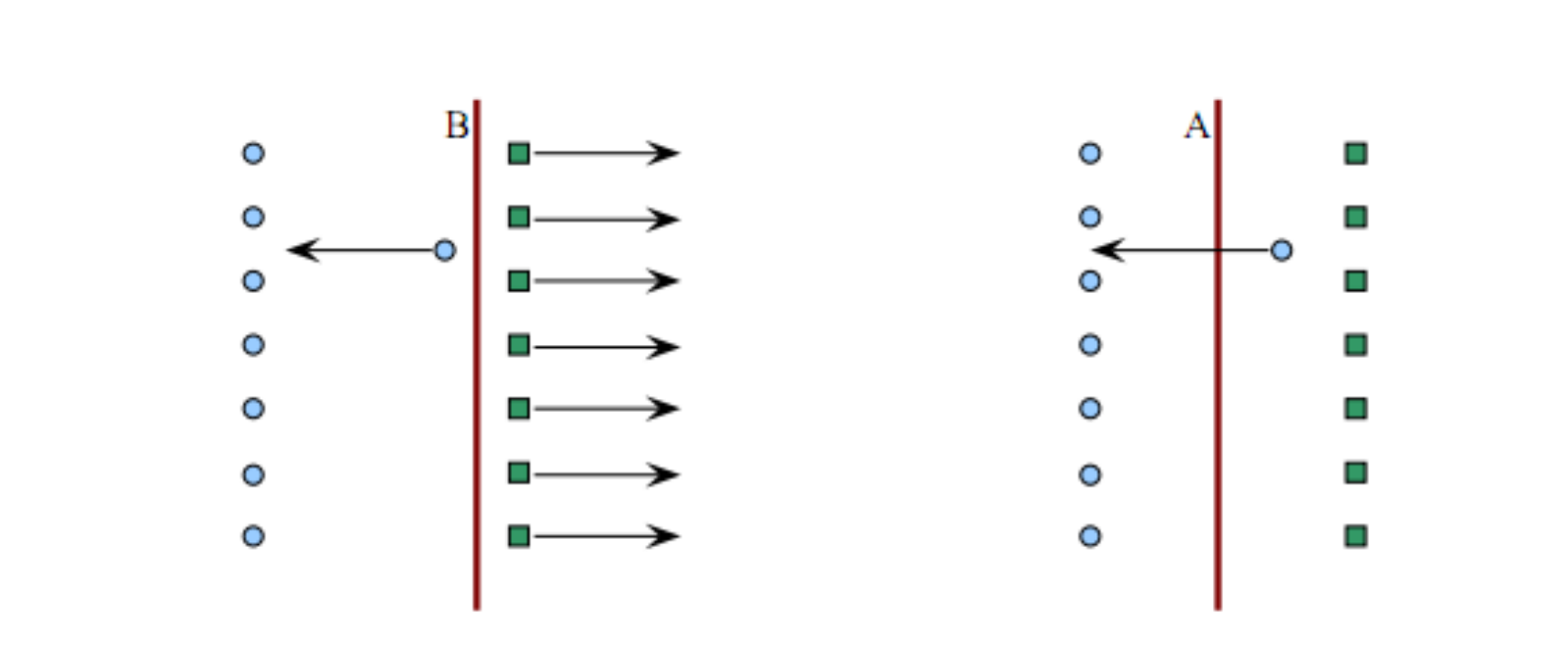
Kernels

- Common:
  - Polynomial $K(\boldsymbol{x}, \boldsymbol{z}) = (a + \boldsymbol{x} \cdot \boldsymbol{z})^b$
  - RBF $K(\boldsymbol{x}, \boldsymbol{z}) = \exp -a\|\boldsymbol{x} - \boldsymbol{z}\|^2$
- Crafted:
  - e.g. String kernels, tree kernels
- Learned:
  - using SDP, boosting ...

Non-Separable Formulation

Hinge Loss + Regulrization

What is better?



Support Vector Machines

- Add s

Primal Non-Separable SVM

$$\boldsymbol{w}^* = \arg\min_{\boldsymbol{w}} \frac{1}{2}\|\boldsymbol{w}\|^2$$

$$\text{s.t. } y_t \boldsymbol{x}_t \cdot \boldsymbol{w} \geq 1 \quad \forall t$$

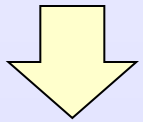
$$\boldsymbol{w}^* = \arg\min_{\boldsymbol{w}} \frac{1}{2}\|\boldsymbol{w}\|^2 + C\sum_t \xi_t$$

$$\text{s.t. } y_t \boldsymbol{x}_t \cdot \boldsymbol{w} \geq 1 - \xi_t \,, \quad \xi_t \geq 0 \quad \forall t$$

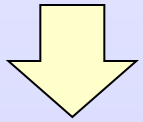
Hinge Loss

- Rewrite constraints

$$y_t \boldsymbol{x}_t \cdot \boldsymbol{w} \geq 1 - \xi_t \quad , \quad \xi_t \geq 0 \quad \forall t$$



$$\xi_t \geq 1 - y_t \boldsymbol{x}_t \cdot \boldsymbol{w} \quad , \quad \xi_t \geq 0 \quad \forall t$$



$$\xi_t \geq \max\{0, 1 - y_t \boldsymbol{x}_t \cdot \boldsymbol{w}\} \quad \forall t$$

Hinge Loss

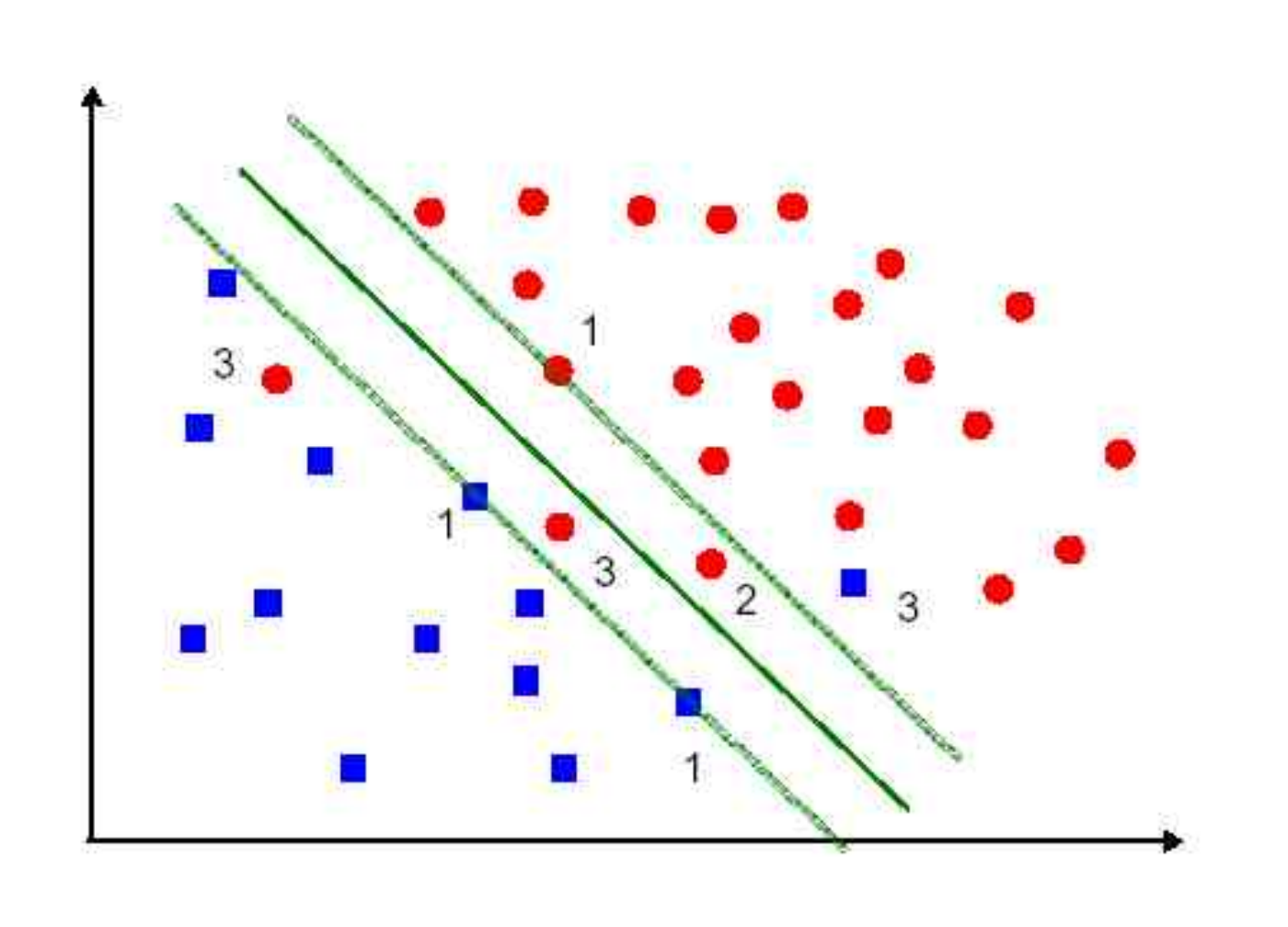
Regularization

Tradeoff Parameter

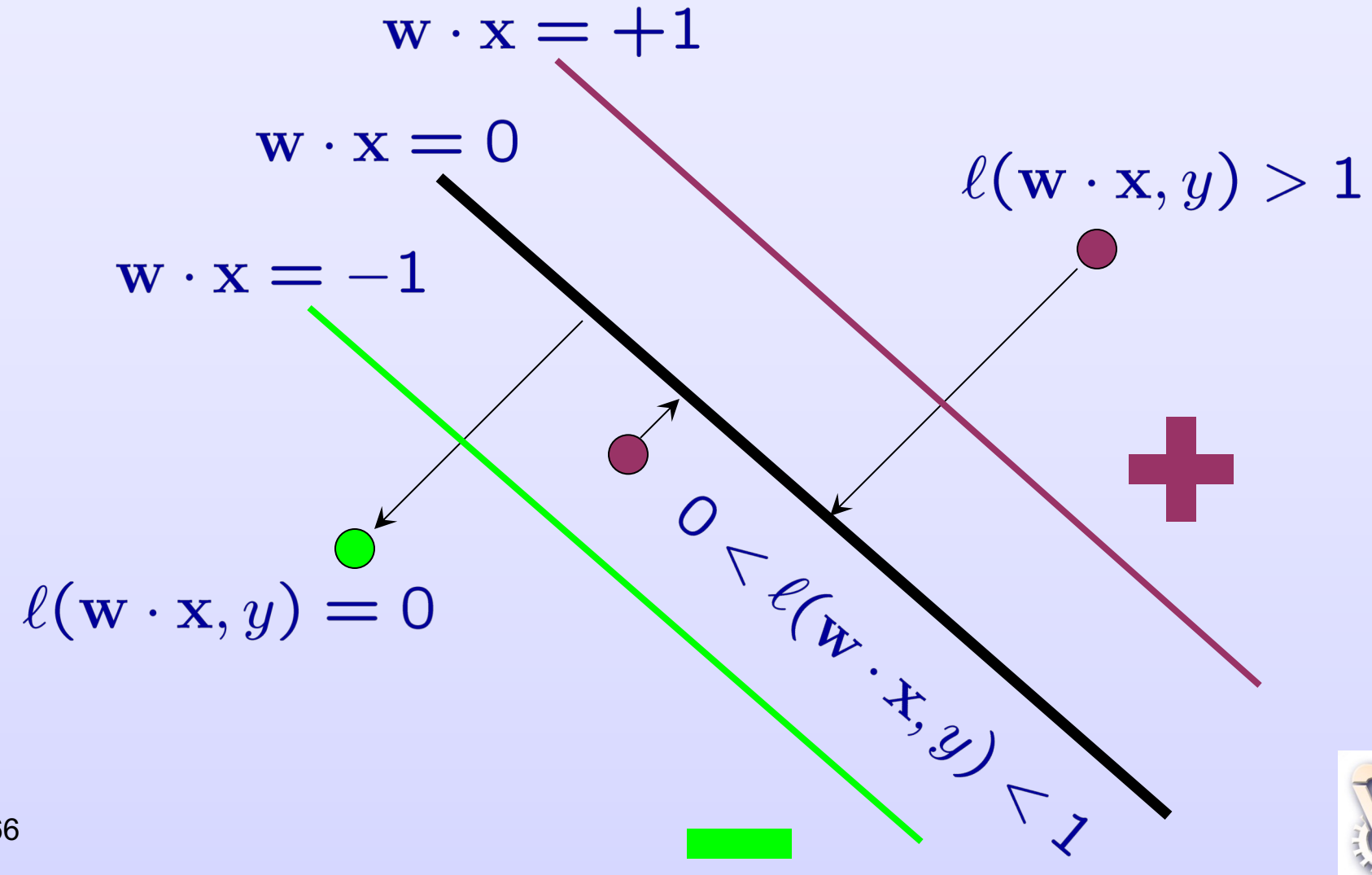
Hinge Loss

$$\boldsymbol{w}^* = \arg\min_{\boldsymbol{w}} \frac{1}{2}\|\boldsymbol{w}\|^2 + C\sum_t \max\{0, 1 - y_t \boldsymbol{x}_t \cdot \boldsymbol{w}\}$$

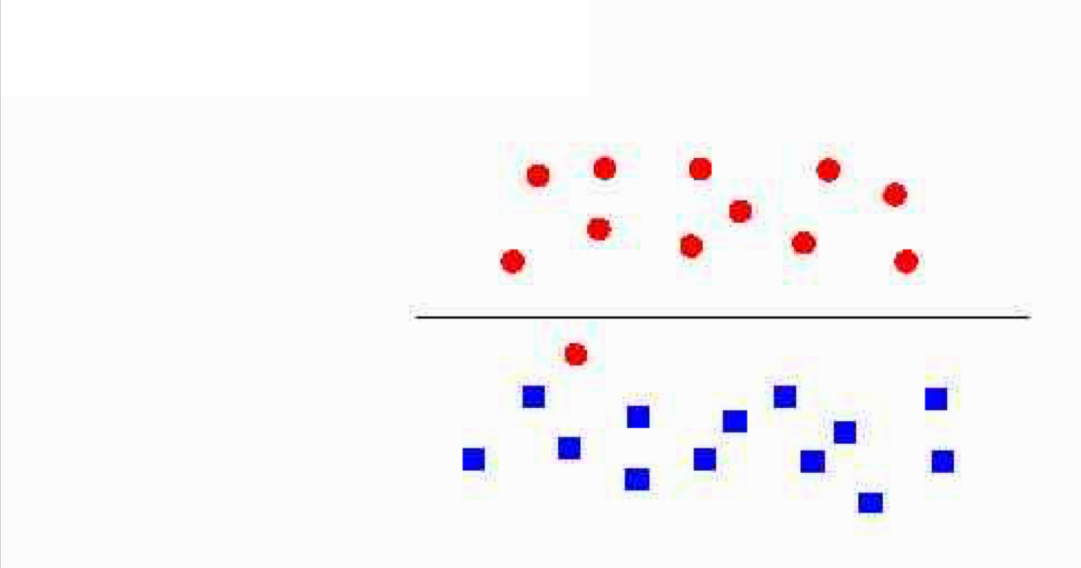
Support Vectors

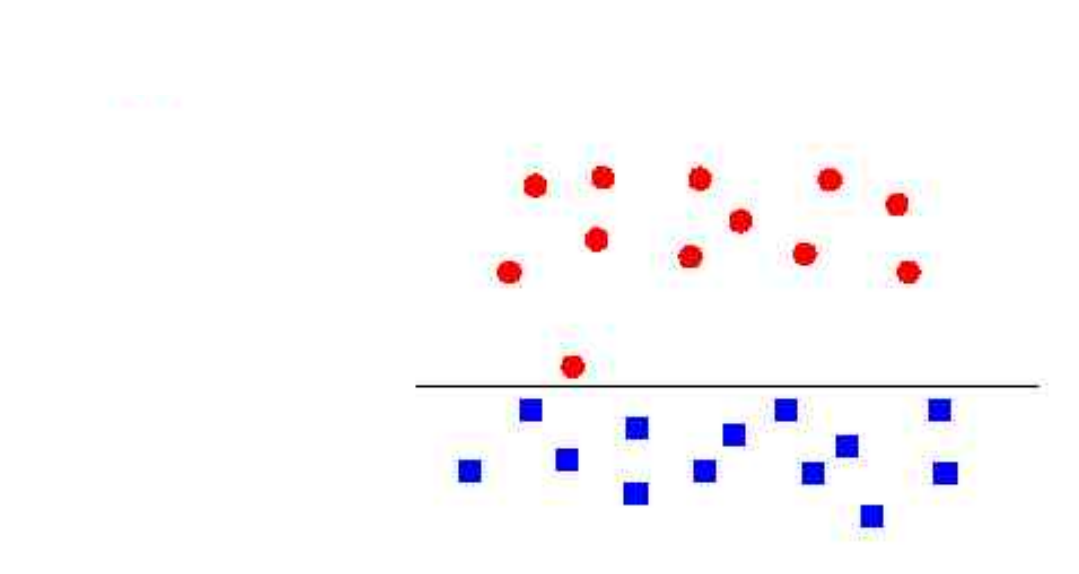


Hinge Loss


$\mathbf{w} \cdot \mathbf{x} = +1$
$\mathbf{w} \cdot \mathbf{x} = 0$
$\mathbf{w} \cdot \mathbf{x} = -1$
$\ell(\mathbf{w} \cdot \mathbf{x}, y) > 1$
$0 < \ell(\mathbf{w} \cdot \mathbf{x}, y) < 1$
$\ell(\mathbf{w} \cdot \mathbf{x}, y) = 0$

Non-Separable Formulation





Dual Non-Separable

- Learning:

$$\max_{\alpha} -\frac{1}{2}\sum_{t,s}\alpha_t\alpha_s y_t y_s K\left(\boldsymbol{x}_t,\boldsymbol{x}_s\right)+\sum_t \alpha_t$$

$$\text{s.t. } 0\leq \alpha_t \leq C \;\forall t$$

- Inference:

$$f(\boldsymbol{x}) = \text{sign}\left(\sum_t \alpha_t y_t K\left(\boldsymbol{x},\boldsymbol{x}_t\right)\right)$$

Comments

- SVM: quadratic objective + linear constraints
- Principle of large margin:
  - Not only be correct, be also confident
- Kernels
- Bias term – class prior
- Many many algorithms to solve optimization
- Variants for:
  - regression, ranking, multi-class, multi-label, structured prediction …

The Common and Different

- Naïve Bayes, Logistic Regression and SVMs employ linear models
- Generative vs. Discriminative
- Probabilistic output?
- Minimization of Regularization + Loss
  - Log loss, Hinge loss, Exp loss, Huber loss …
  - Squared L2, L1, L-infinity, …
- Many algorithms can be combined with kernels, including logistic-regression
- Generalization theory