CSI 436/536
Introduction to Machine Learning

SVM theory

Professor Siwei Lyu
Computer Science
University at Albany, State University of New York

Support vector machines

- Support vector machines (SVM) is one of the most widely used ML algorithms today
  - Theoretical foundation (statistical learning theory) developed in 1960s by Vapnik & Chervonenkis
  - Algorithm first introduced by Vapnik et.al. in 1992
  - Aims to replace NN as a more provable method to alleviate overfitting
  - Many successful applications (computer vision, text, bioinformatics)

Key components of SVM

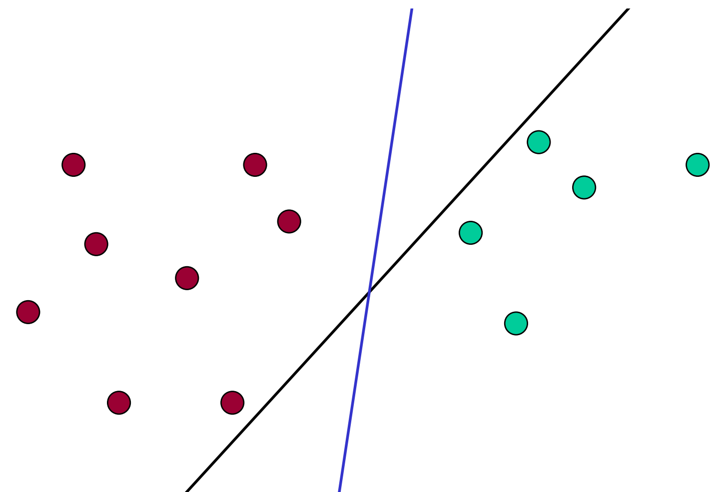
- Large-margin learning
  - theoretical guarantee of good performance in generalization
  - Efficiency in model: reducing training data to SVs
- Quadratic programming optimization
  - efficient optimization and unique global solution
- The "Kernel tricks"
  - extension to nonlinear prediction functions and models without explicit feature mapping

SVM for binary classification

- Characteristics
  - training to maximize classification ***margin***
  - decision function specified by a subset of training examples known as the **support vectors**
- we study the following cases
  - Linear SVM: separable case
  - Linear SVM: non-separable case
  - Nonlinear SVM

Linear SVM: separable case

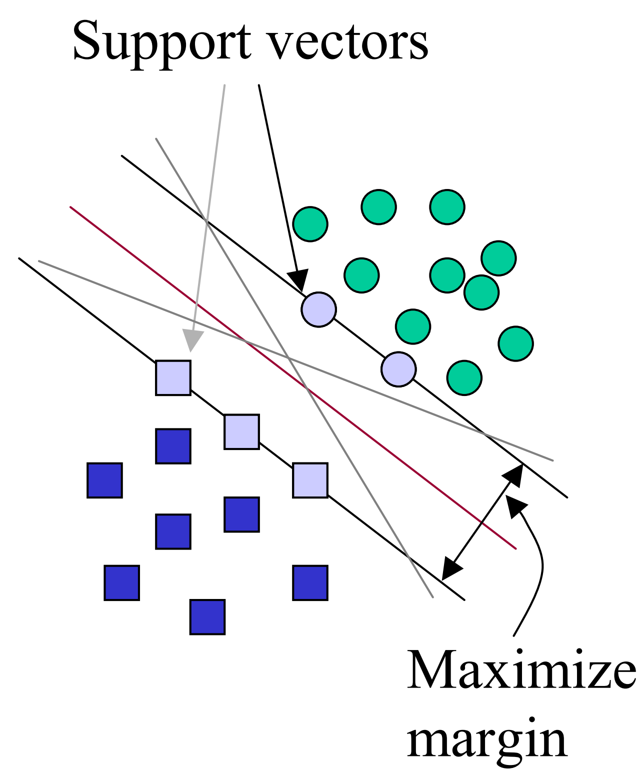
- Uses linear prediction function $f(x, \{w, b\}) = \text{sign}(w \cdot x + b).$
- Assume separable data:
  - There exist a linear function that can perfectly separate the two classes of data
  - If there is one linear function that can separate the two classes of data, then there are ***infinite*** number of linear functions that can do the same (**Hausdorff separation theorem**)



- The question is: which one is the optimal
  - This is an ill-posed problem

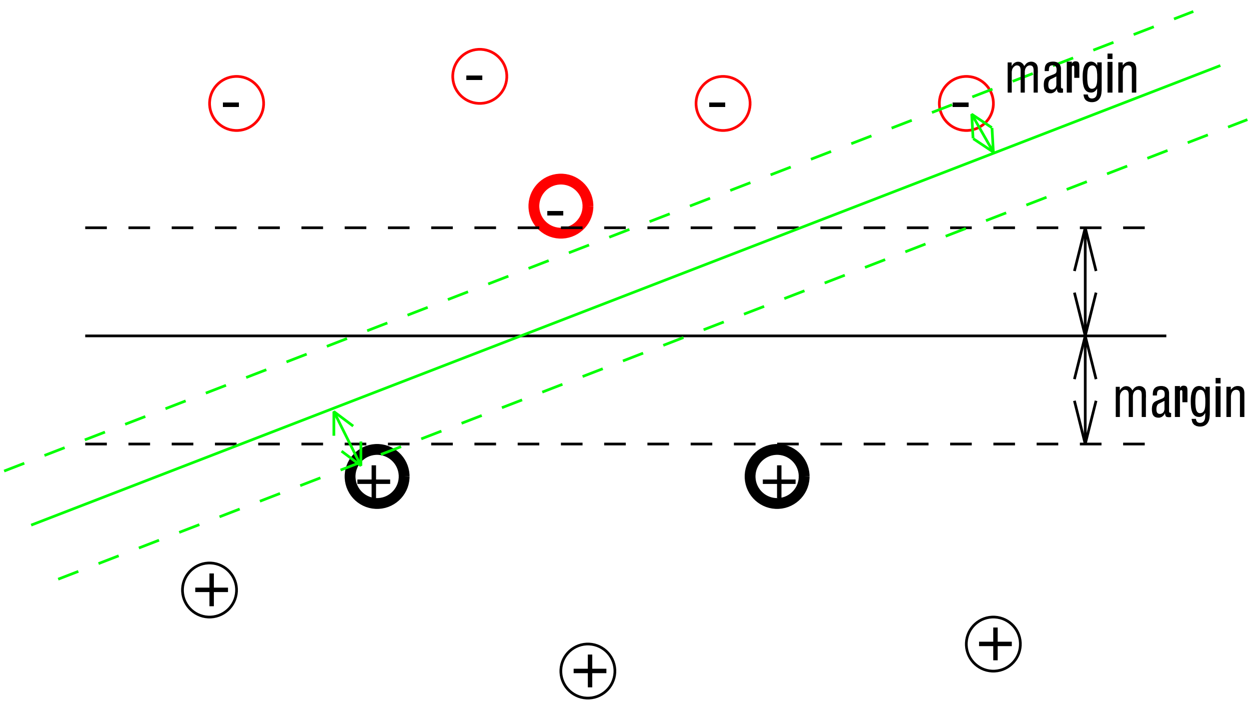
Linear SVM: separable case

- choosing an optimal linear classifier for separable training data is an ill-posed problem
  - Extra requirement: the classifier needs to generalize to unseen data
- Idea of SVM
  - Find linear classifier with the maximal classification margin
  - Margin measures the size of the open space between the two classes given a classifier



Why seek larger-margin?

- Large margin gives less chance for future errors
- Large margin guarantees generalization of the learned model



Large margin and generalization

- We can try to learn $f(x, \alpha)$ by choosing a function that performs well on training data:

$$R_{emp}(\alpha) = \frac{1}{m}\sum_{i=1}^{m} \ell(f(x_i, \alpha), y_i) = \text{Training Error}$$

where ℓ is the zero-one *loss function*, $\ell(y, \hat{y}) = 1$, if $y \neq \hat{y}$, and 0 otherwise. R_{emp} is called the *empirical risk*.

- By doing this we are trying to minimize the overall risk:

$$R(\alpha) = \int \ell(f(x, \alpha), y) dP(x, y) = \text{Test Error}$$

where P(x,y) is the (unknown) joint distribution function of x and y.

No free lunch theorem

- training data alone are not enough to choose which function is better
- if f(x) allows all function from X to {±1}

Training set $(x_1, y_1), \ldots, (x_m, y_m) \in \mathcal{X} \times \{\pm 1\}$

Test set $\bar{x}_1, \ldots, \bar{x}_{\bar{m}} \in \mathcal{X}$,

such that the two sets do not intersect.

For any f there exists f^*:

1. $f^*(x_i) = f(x_i)$ for all i
2. $f^*(x_j) \neq f(x_j)$ for all j

Controlling the flexibility

- NFL theorem says that we cannot use the whole function family for learning as it will easily lead to overfitting
- When all things equal, we should choose a model family that is not "too flexible"
- How do we quantify a model family's flexibility

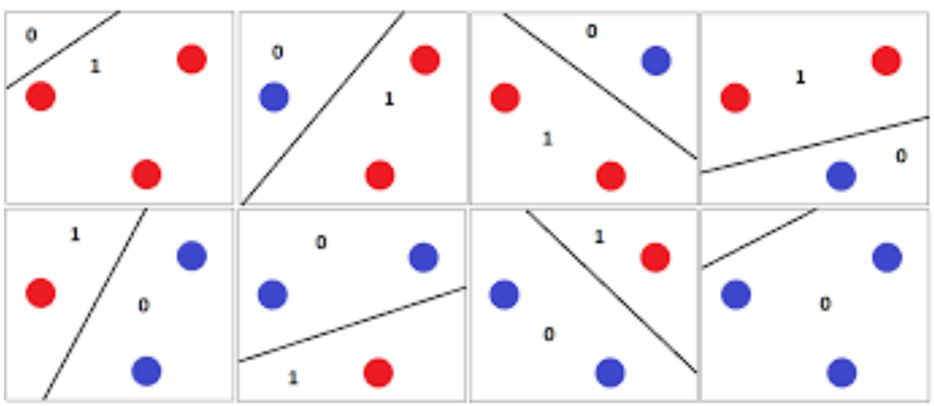
Shattering

- A decision function mapping X → {-1,+1} limited to a training set of m samples is equivalent to a complete bipartite graph
  - One set of nodes correspond to m training data
  - The other correspond to {-1,+1} label

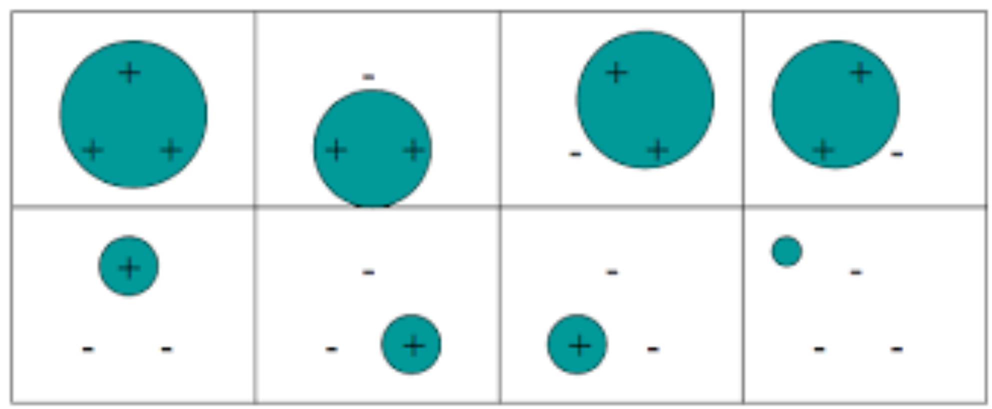
  - The total number of such mapping is 2^m
- A function family shatters a data set means that all such mappings can be obtained from one member from that family

The VC dimension

- The Vapnik-Chervonenkis (VC) dimension
  - A combinatorial entity controlling the flexibility of a function family, the more phenomena explained by f, the higher the VC-dim
- It is the ***maximum number*** of points that can be shattered in all possible ways by a member of the function family



Lines



Circles

Some VC-dims

- For a finite family, VC-dim(H) $\leq \log_2|H|$
- hyper-plane in d-dims space has VC-dim d+1
- Neural network with n nodes and E edges has VC-dim O(nE)
- Norm-limited hyper-planes

Consider hyperplanes $(w \cdot x) = 0$ *where* w *is normalized w.r.t a set of points* X^* *such that:* $\min_i |w \cdot x_i| = 1$.

The set of decision functions $f_w(x) = sign(w \cdot x)$ *defined on* X^* *such that* $||w|| \leq A$ *has a VC dimension satisfying*

$$h \leq R^2 A^2.$$

where R *is the radius of the smallest sphere around the origin containing* X^*.

VC-dim and generalization

- Vapnik & Chervonenkis in the 1960s showed that
  - For any function family with VC-dim <= h
  - For any training set of size m
  - For any number $\eta \in (0,1)$, with probability larger than 1-η, we have

$$R(\alpha) \leq R_{emp}(\alpha) + \sqrt{\frac{h(log(\frac{2m}{h}+1) - log(\frac{\eta}{4})}{m}}$$

or simply, with high probability,

Test Error $\leq$ Training Error + Complexity of set of Models

Interpreting the inequality

- It is probabilistic: so there is a chance, albeit small, that it does not hold true
- It is a bound: so even we minimize the RHS, the true risk may still be large
- It is for a family of functions, so it is not really that useful for individual model
- It works for all data distributions, so it may not give the best on the data we interested
- Its asymptotic behavior is good but not useful

How to understand this

- With high probability, we have

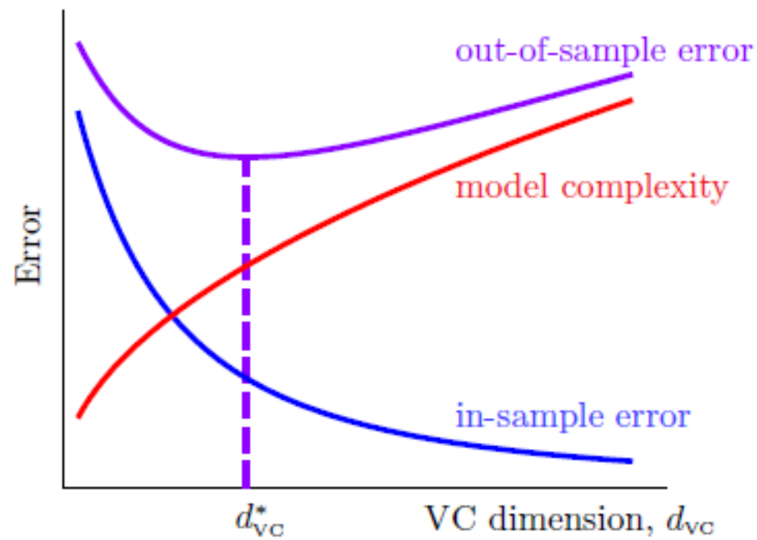
  test error ≤ **training error** + **model complexity**

  a high capacity set of functions get low training error but may ”**overfit**”

  - a simple set of models have low complexity, but will get high training error ”**under-fit**”

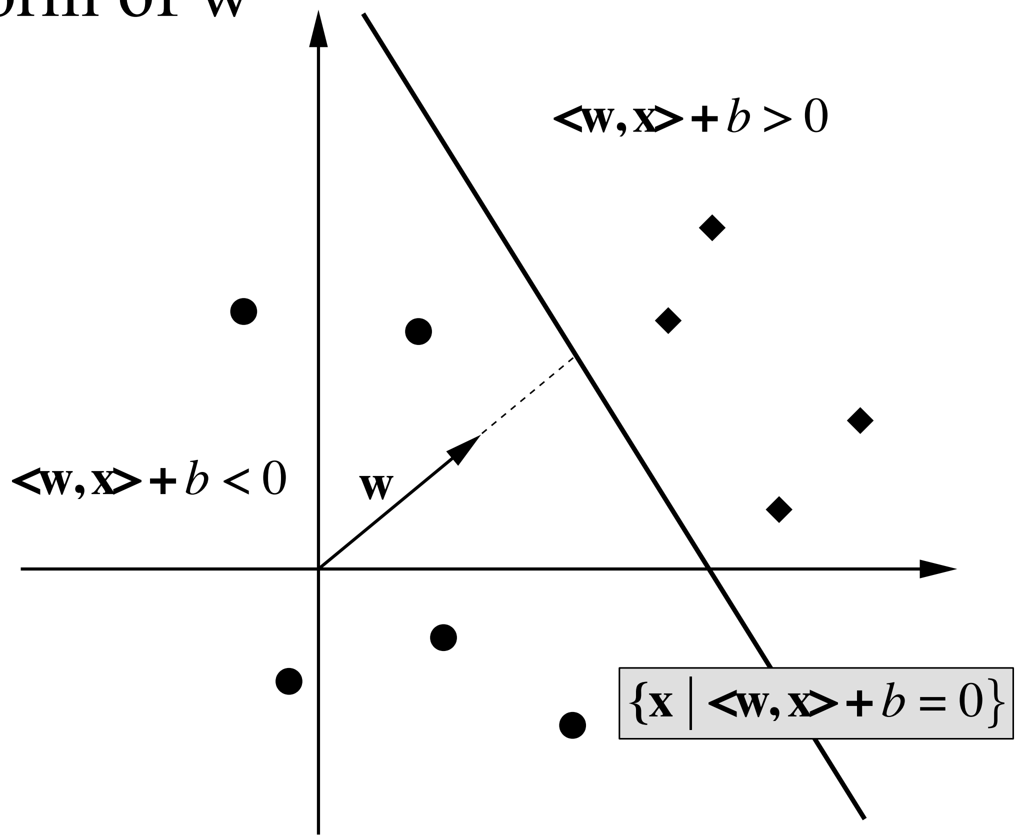
- We can understand it as

  test error ≤ **training error** + VC-dimension



Large margin -> low VC dim

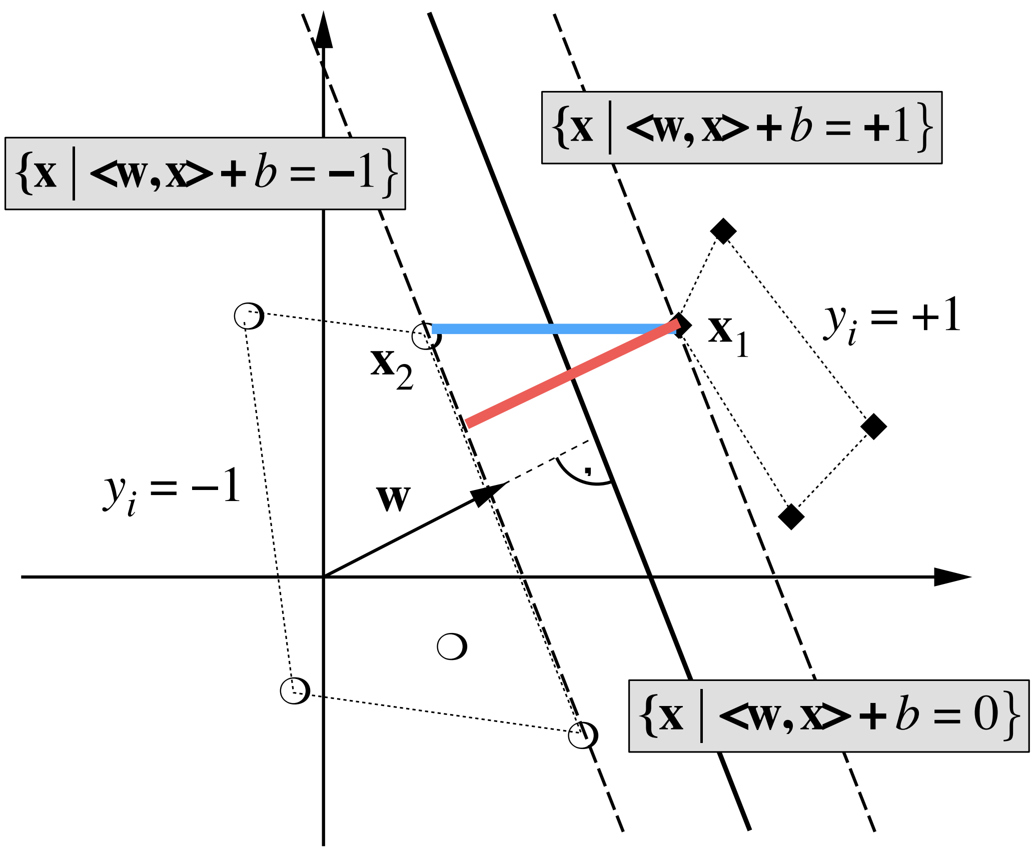
- For most models, we cannot compute VC-dim, but for linear classifiers $w^T x$ we can bound its VC-dim with the norm of w



- The norm of w is related with classification margin

Large margin -> low VC dim

- Large margin -> upper bound norm of w -> related with the VC dim of norm bounded linear functions



Note:

$$\langle \mathbf{w}, \mathbf{x}_1 \rangle + b = +1$$
$$\langle \mathbf{w}, \mathbf{x}_2 \rangle + b = -1$$

$$\Rightarrow \quad \langle \mathbf{w}, (\mathbf{x}_1 - \mathbf{x}_2) \rangle = 2$$

$$\Rightarrow \quad \left\langle \frac{\mathbf{w}}{\|\mathbf{w}\|}, (\mathbf{x}_1 - \mathbf{x}_2) \right\rangle = \frac{2}{\|\mathbf{w}\|}$$

From low complexity to larger margin

- Large margin -> upper bound norm of w -> related with the VC dim of norm bounded linear functions
- Using the VC-inequality, we would like to minimize the upper-bound of test error
  test error ≤ **training error** + **VC-dimension**
- For linear model, we use the result that for the family of linear functions determined by w, $f(x) = w^T x + b$ (varying b), VC-dim < O(||w||), so for linear model, we have (roughly)
  test error ≤ **training error** + $||w||^2$

Linear SVM: separable case

That function before was a little difficult to minimize because of the step function in $\ell(y, \hat{y})$ (either 1 or 0).

Let's assume we can separate the data perfectly. Then we can optimize the following:

Minimize $||w||^2$, subject to:

$$(w \cdot x_i + b) \geq 1, \text{ if } y_i = 1$$

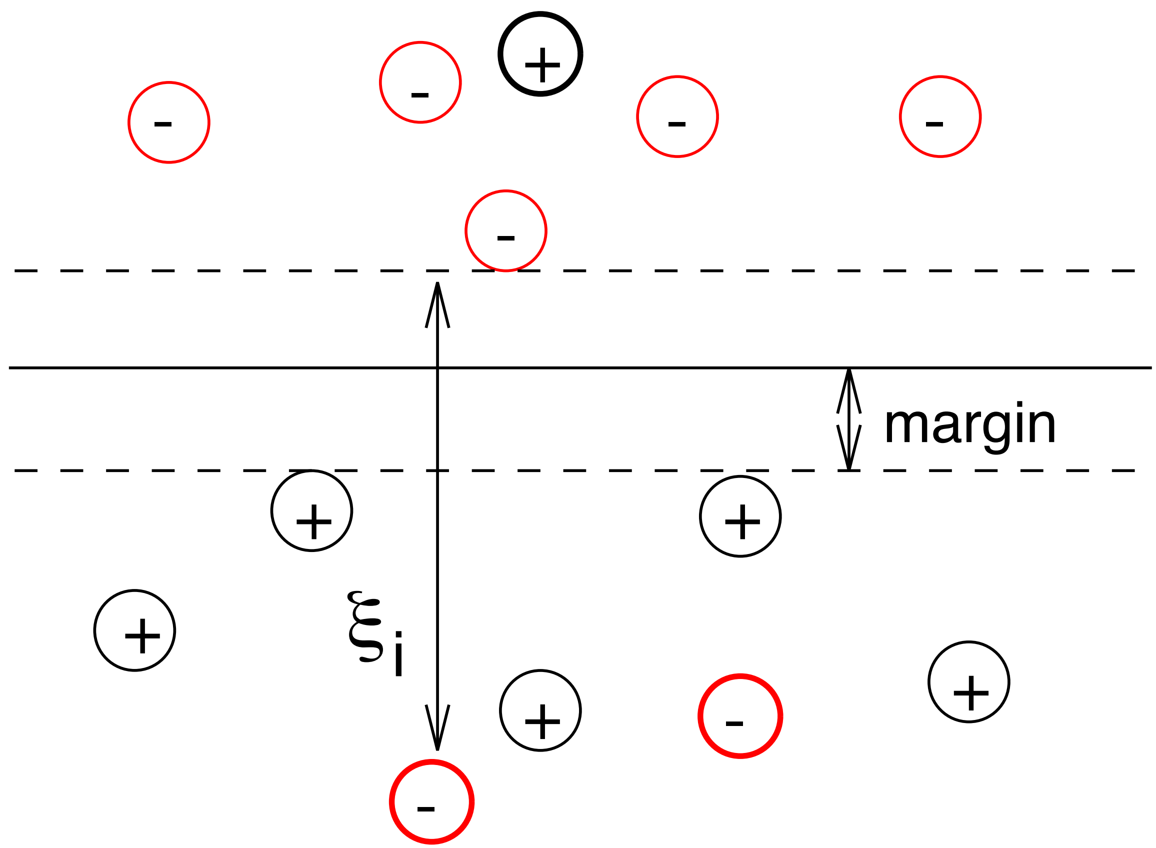
$$(w \cdot x_i + b) \leq -1, \text{ if } y_i = -1$$

The last two constraints can be compacted to:

$$y_i(w \cdot x_i + b) \geq 1$$

Linear SVM: non-separable case

- Introducing slack variables to measure the error



- SVs are those data points that support the hyperplane and in the margin area

Linear SVM: non-separable case

Minimize: w and b

$$||w||^2 + C \sum_{i=1}^{m} \xi_i$$

subject to:

$$y_i(w \cdot x_i + b) \geq 1 - \xi_i, \quad \xi_i \geq 0$$

This is just the same as the original objective:

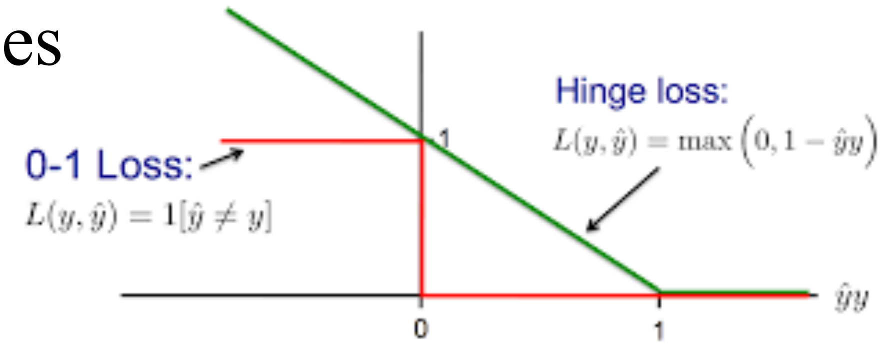
$$\text{C} \ \frac{1}{m} \sum_{i=1}^{m} \ell(w \cdot x_i + b, y_i) + ||w||^2$$

except ℓ is no longer the zero-one loss, but is called the "hinge-loss": $\ell(y, \hat{y}) = \max(0, 1 - y\hat{y})$. This is still a quadratic program!

Why hinge loss

- We can use other types of losses

$$C\,\frac{1}{m}\sum_{i=1}^{m}\ell(w\cdot x_i+b,y_i)+||w||^2$$



- If we use least squares loss, this is Tikhonov-regularized binary classification
- We can also use logistic loss, then it is Tikhonov-regularized logistic regression
- Hinge loss gives sparsity
  - Optimal solution is going to be a linear combination of training data, hinge loss makes sure we only need a small set of them
  - Important for nonlinear SVM