Machine Learning

CS 539

Worcester Polytechnic Institute
Department of Computer Science
Instructor: Prof. Kyumin Lee

Upcoming Schedule

- HW4
  - https://canvas.wpi.edu/courses/58900/assignments/357384
  - Due date is July 16

- Online Quiz3 will be taken on July 9
  - Coverage: from neural network to SVM

What if Data Are Not Linearly Separable? (Soft Margin linear SVM Classifier)

- Cannot find $\boldsymbol{\theta}$ that satisfies $y_i(\boldsymbol{\theta}^\mathsf{T}\mathbf{x}_i) \geq 1 \quad \forall i$

- Introduce slack variables ξ_i

$$y_i(\boldsymbol{\theta}^\mathsf{T}\mathbf{x}_i) \geq 1 - \xi_i \quad \forall i$$

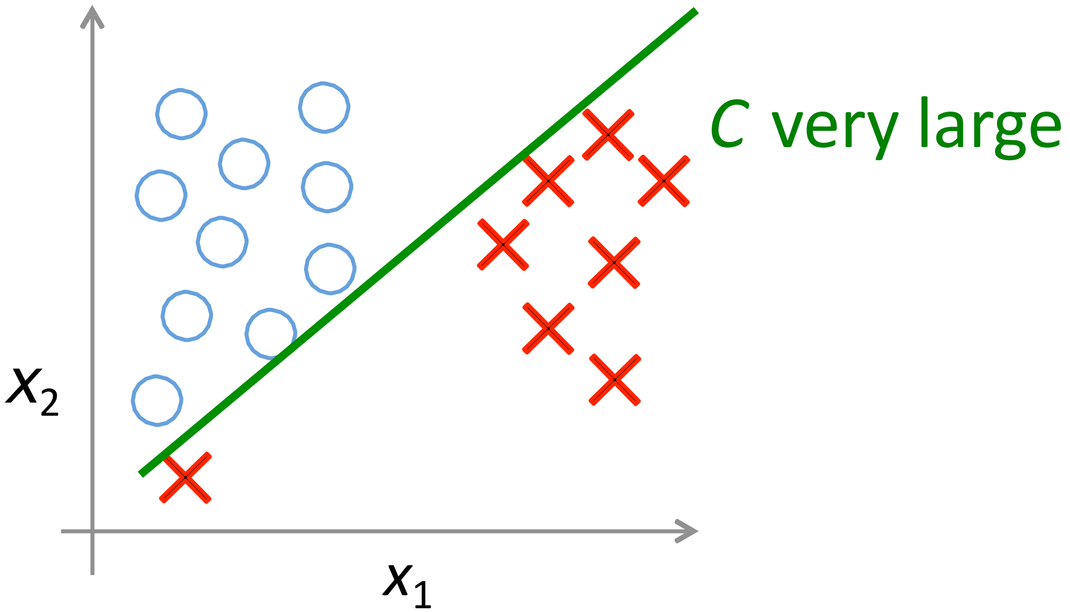
- New problem:

$$\min_{\boldsymbol{\theta}} \frac{1}{2}\sum_{j=1}^{d}\theta_j^2 + C\sum_i \xi_i$$

$$\text{s.t. } y_i(\boldsymbol{\theta}^\mathsf{T}\mathbf{x}_i) \geq 1 - \xi_i \quad \forall i$$

Refer to Flach: 7.3: pages 218 - 219

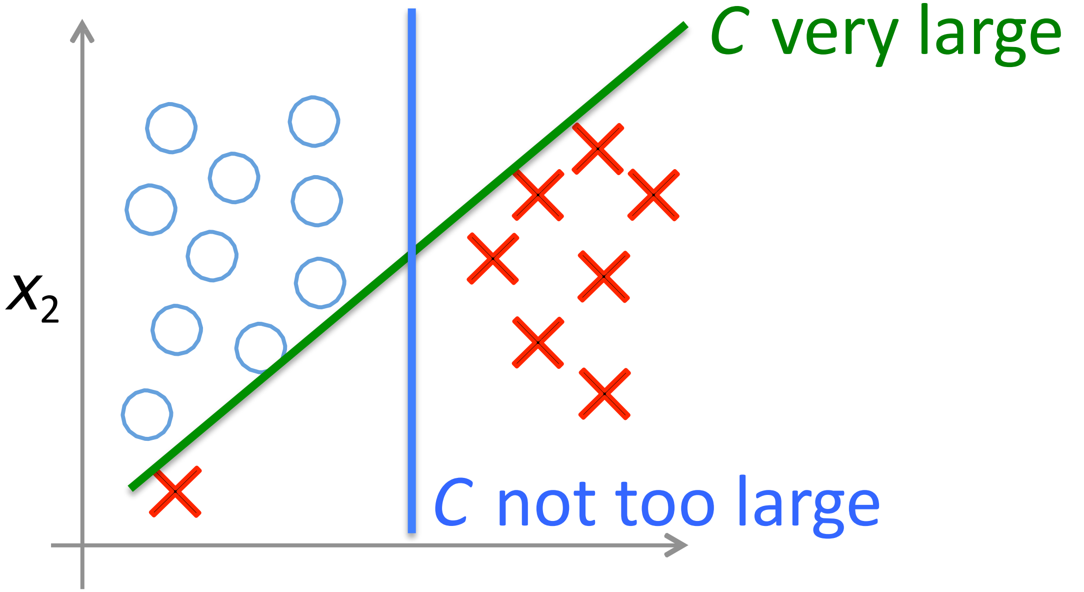
Large Margin Classifier in Presence of Outliers



$$\min_{\boldsymbol{\theta}} \frac{1}{2} \sum_{j=1}^{d} \theta_j^2 + C \sum_i \xi_i$$

$$\text{s.t. } \; y_i(\boldsymbol{\theta}^\intercal \mathbf{x}_i) \geq 1 - \xi_i \quad \forall i$$

Large Margin Classifier in Presence of Outliers



$$\min_{\boldsymbol{\theta}} \frac{1}{2}\sum_{j=1}^{d}\theta_j^2 + C\sum_i \xi_i$$

$$\text{s.t. } y_i(\boldsymbol{\theta}^\intercal \mathbf{x}_i) \geq 1 - \xi_i \quad \forall i$$

You can think of C as similar to $\frac{1}{\lambda}$

SVM Dual Representation

Maximize $J(\boldsymbol{\alpha}) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j \langle \mathbf{x}_i, \mathbf{x}_j \rangle$

$$\text{s.t.} \quad \boxed{0 \leq \alpha_i \leq \mathrm{C}} \quad \forall i$$

$$\sum_{i} \alpha_i y_i = 0$$

$\alpha_i = 0$ these are outside or on the margin;

$0 < \alpha_i < C$ these are the support vectors on the margin;

$\alpha_i = C$ these are on or inside the margin.

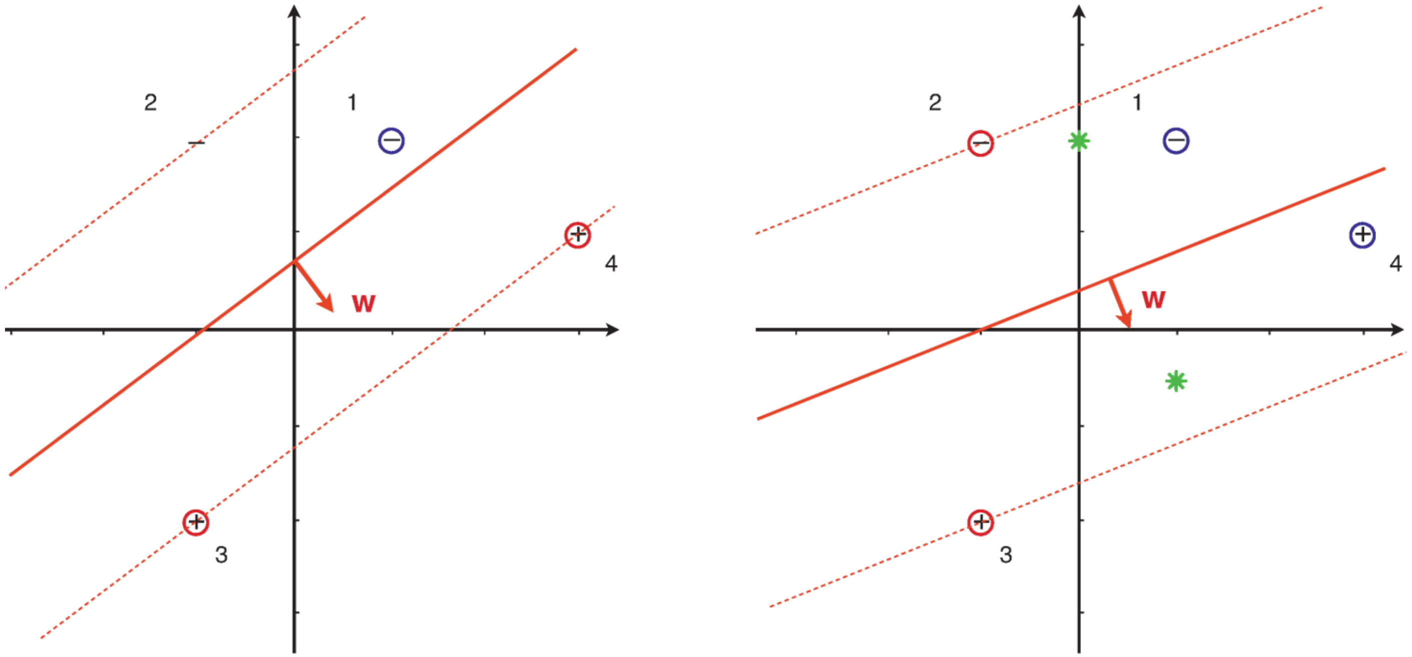
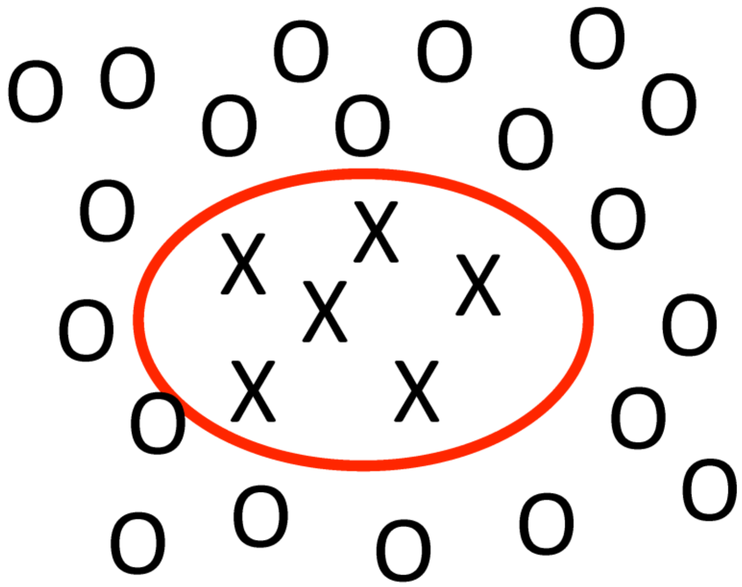
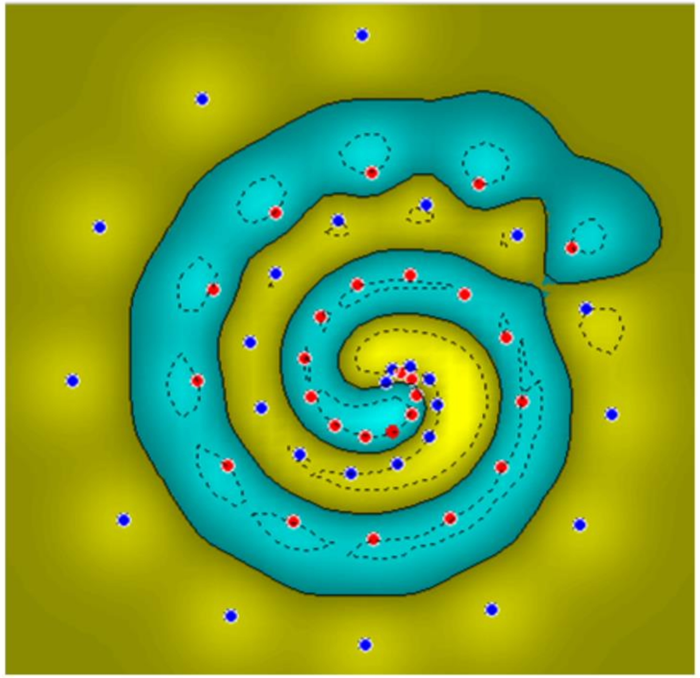


Figure 7.9. **(left)** The soft margin classifier learned with $C = 5/16$, at which point $\mathbf{x}_2$ is about to become a support vector. **(right)** The soft margin classifier learned with $C = 1/10$: all examples contribute equally to the weight vector. The asterisks denote the class means, and the decision boundary is parallel to the one learned by the basic linear classifier.

The figure was adapted from Flach 7.3: pages 218 - 219

What if Surface is Non-Linear?

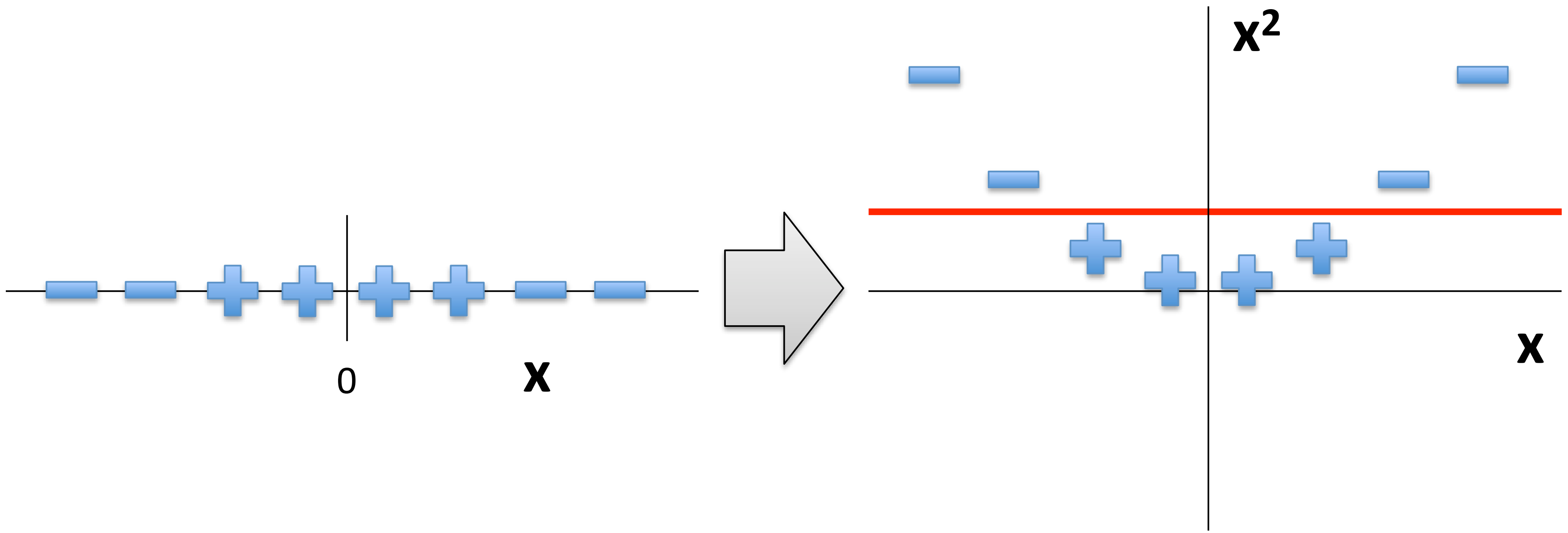




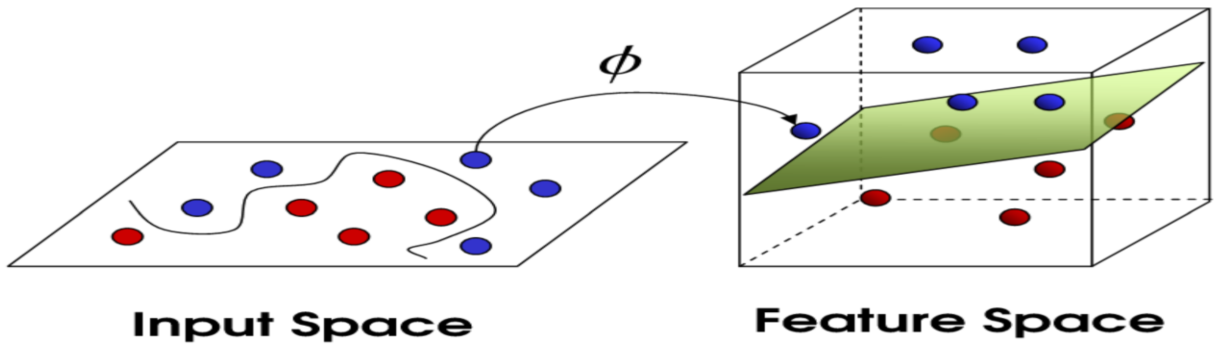
Kernel Methods

Making the Non-Linear Linear

When Linear Separators Fail



Mapping into a New Feature Space



$$\Phi : \mathcal{X} \mapsto \hat{\mathcal{X}} = \Phi(\mathbf{x})$$

- For example, with $\mathbf{x}_i \in \mathbb{R}^2$

$$\Phi([x_{i1}, x_{i2}]) = [x_{i1}, x_{i2}, x_{i1}x_{i2}, x_{i1}^2, x_{i2}^2]$$

- Rather than run SVM on $\mathbf{x}_i$, run it on $\Phi(\mathbf{x}_i)$
  - Find non-linear separator in input space
- What if $\Phi(\mathbf{x}_i)$ is really big?
  - Use kernels to compute it implicitly!

Kernels

- Find kernel K such that

$$K(\mathbf{x}_i, \mathbf{x}_j) = \langle \Phi(\mathbf{x}_i), \Phi(\mathbf{x}_j) \rangle$$

- Computing $K(\mathbf{x}_i, \mathbf{x}_j)$ should be efficient, much more so than computing $\Phi(\mathbf{x}_i)$ and $\Phi(\mathbf{x}_j)$

- Use $K(\mathbf{x}_i, \mathbf{x}_j)$ in SVM algorithm rather than $\langle \mathbf{x}_i, \mathbf{x}_j \rangle$
- Remarkably, this is possible!

$$J(\boldsymbol{\alpha}) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j \langle \mathbf{x}_i, \mathbf{x}_j \rangle$$

The Polynomial Kernel

Let $\mathbf{x}_i = [x_{i1}, x_{i2}]$ and $\mathbf{x}_j = [x_{j1}, x_{j2}]$

Consider the following function:

$$\begin{aligned} K(\mathbf{x}_i, \mathbf{x}_j) &= \langle \mathbf{x}_i, \mathbf{x}_j \rangle^2 \\ &= (x_{i1}x_{j1} + x_{i2}x_{j2})^2 \\ &= (x_{i1}^2 x_{j1}^2 + x_{i2}^2 x_{j2}^2 + 2x_{i1}x_{i2}x_{j1}x_{j2}) \\ &= \langle \Phi(\mathbf{x}_i), \Phi(\mathbf{x}_j) \rangle \end{aligned}$$

where

$$\Phi(\mathbf{x}_i) = [x_{i1}^2, x_{i2}^2, \sqrt{2}x_{i1}x_{i2}]$$

$$\Phi(\mathbf{x}_j) = [x_{j1}^2, x_{j2}^2, \sqrt{2}x_{j1}x_{j2}]$$

Incorporating Kernels into SVM

$$J(\boldsymbol{\alpha}) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j \cancel{\langle \mathbf{x}_i, \mathbf{x}_j \rangle}$$

$$J(\boldsymbol{\alpha}) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j \boxed{K(\mathbf{x}_i, \mathbf{x}_j)}$$

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

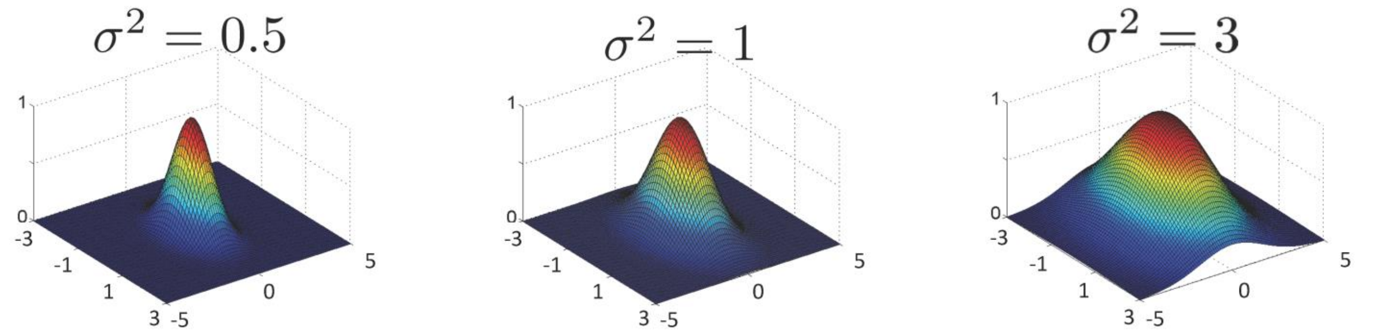
$$\sum_i \alpha_i y_i = 0$$

The Gaussian Kernel

- Also called Radial Basis Function (RBF) kernel

$$K(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|_2^2}{2\sigma^2}\right)$$

  - Has value 1 when $\mathbf{x}_i = \mathbf{x}_j$
  - Value falls off to 0 with increasing distance
  - Note: Need to do feature scaling <u>before</u> using Gaussian Kernel



A Few Good Kernels…

- Linear Kernel $K(\mathbf{x}_i, \mathbf{x}_j) = \langle \mathbf{x}_i, \mathbf{x}_j \rangle$
- Polynomial kernel $K(\mathbf{x}_i, \mathbf{x}_j) = (\langle \mathbf{x}_i, \mathbf{x}_j \rangle + c)^d$
  - $c \geq 0$ trades off influence of lower order terms
- Gaussian kernel $K(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|_2^2}{2\sigma^2}\right)$
- Sigmoid kernel $K(\mathbf{x}_i, \mathbf{x}_j) = \tanh\left(\alpha \mathbf{x}_i^\intercal \mathbf{x}_j + c\right)$

Many more…

- Cosine similarity kernel
- Chi-squared kernel
- String/tree/graph/wavelet/etc kernels

Practical Advice for Applying SVMs

- Use SVM software package to solve for parameters
  - e.g., SVMlight, libsvm, cvx (fast!), etc.

    scikit-learn internally use libsvm and liblinear

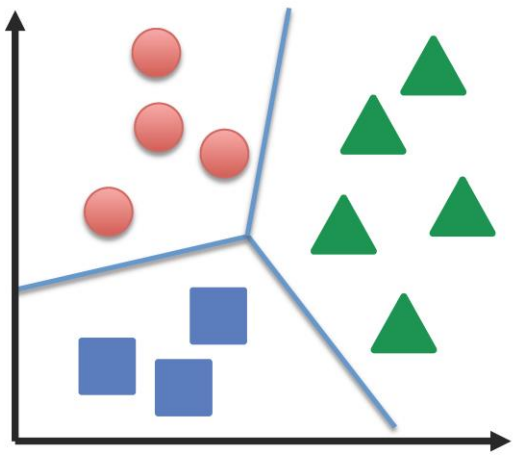
- Need to specify:
  - Choice of parameter C
  - Choice of kernel function
    - Associated kernel parameters

e.g., $K(\mathbf{x}_i, \mathbf{x}_j) = (\langle \mathbf{x}_i, \mathbf{x}_j \rangle + c)^d$

$$K(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|_2^2}{2\sigma^2}\right)$$

http://scikit-learn.org/stable/modules/svm.html

Multi-Class Classification with SVMs



$y \in \{1, \ldots, K\}$

- Many SVM packages already have multi-class classification built in
- Otherwise, use one-vs-rest
  - Train K SVMs, each picks out one class from rest, yielding $\boldsymbol{\theta}^{(1)}, \ldots, \boldsymbol{\theta}^{(K)}$
  - Predict class i with largest $(\boldsymbol{\theta}^{(i)})^\intercal \mathbf{x}$

Going back to Soft Margin linear SVM Classifier….

Quadratic Solver may not be scalable (or slow) to handle a large dataset

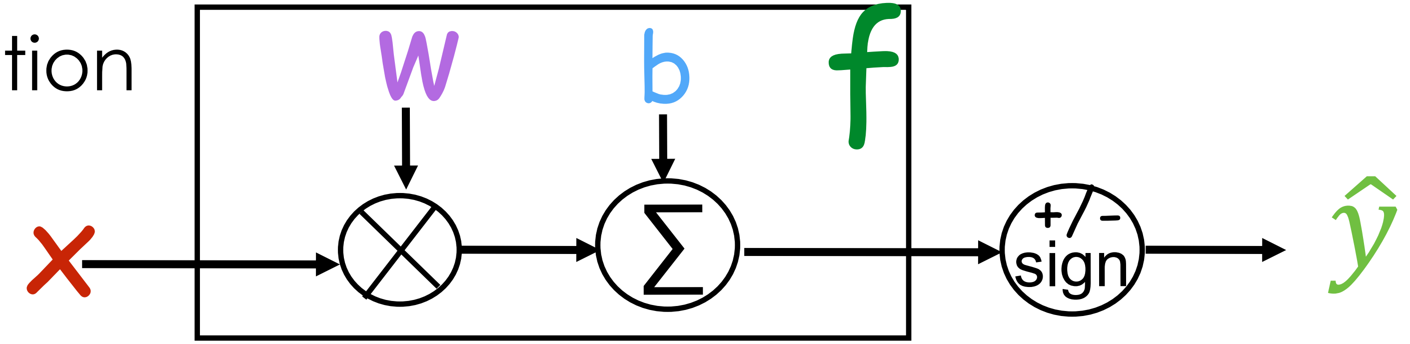
- New problem:

$$\min_{\boldsymbol{\theta}} \frac{1}{2}\sum_{j=1}^{d}\theta_j^2 + C\sum_i \xi_i$$

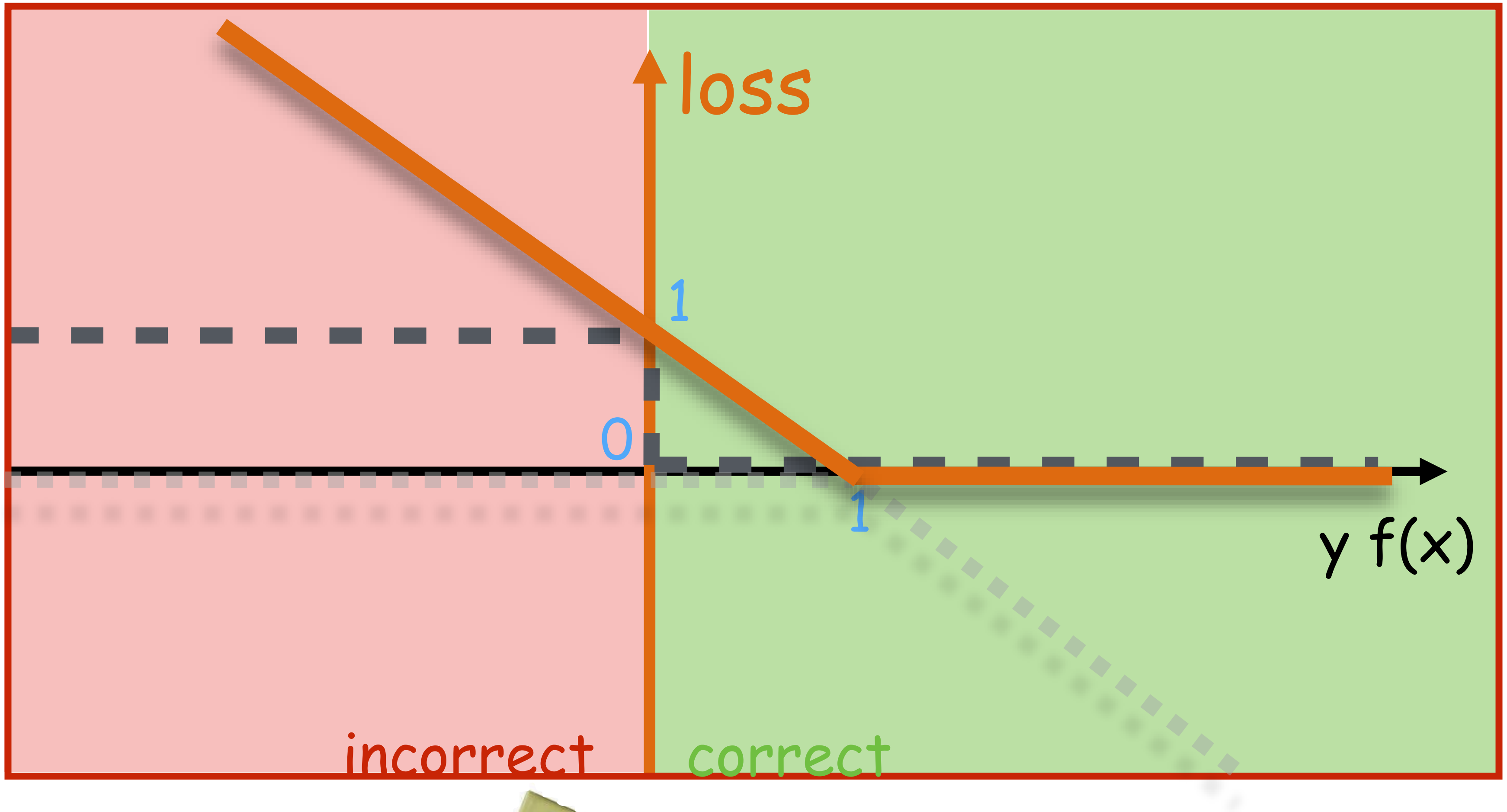
$$\text{s.t. } y_i(\boldsymbol{\theta}^\intercal \mathbf{x}_i) \geq 1 - \xi_i \quad \forall i$$

$$\boxed{\underset{w,\ b}{\text{minimize}} \ \frac{1}{2}\|w\|_2^2 + C\sum_{i=1}^{n} \max(0,\ 1 - y_i f(x_i))}$$

Prediction



max(0 , 1 - y f(x))



Convex

easy to optimize

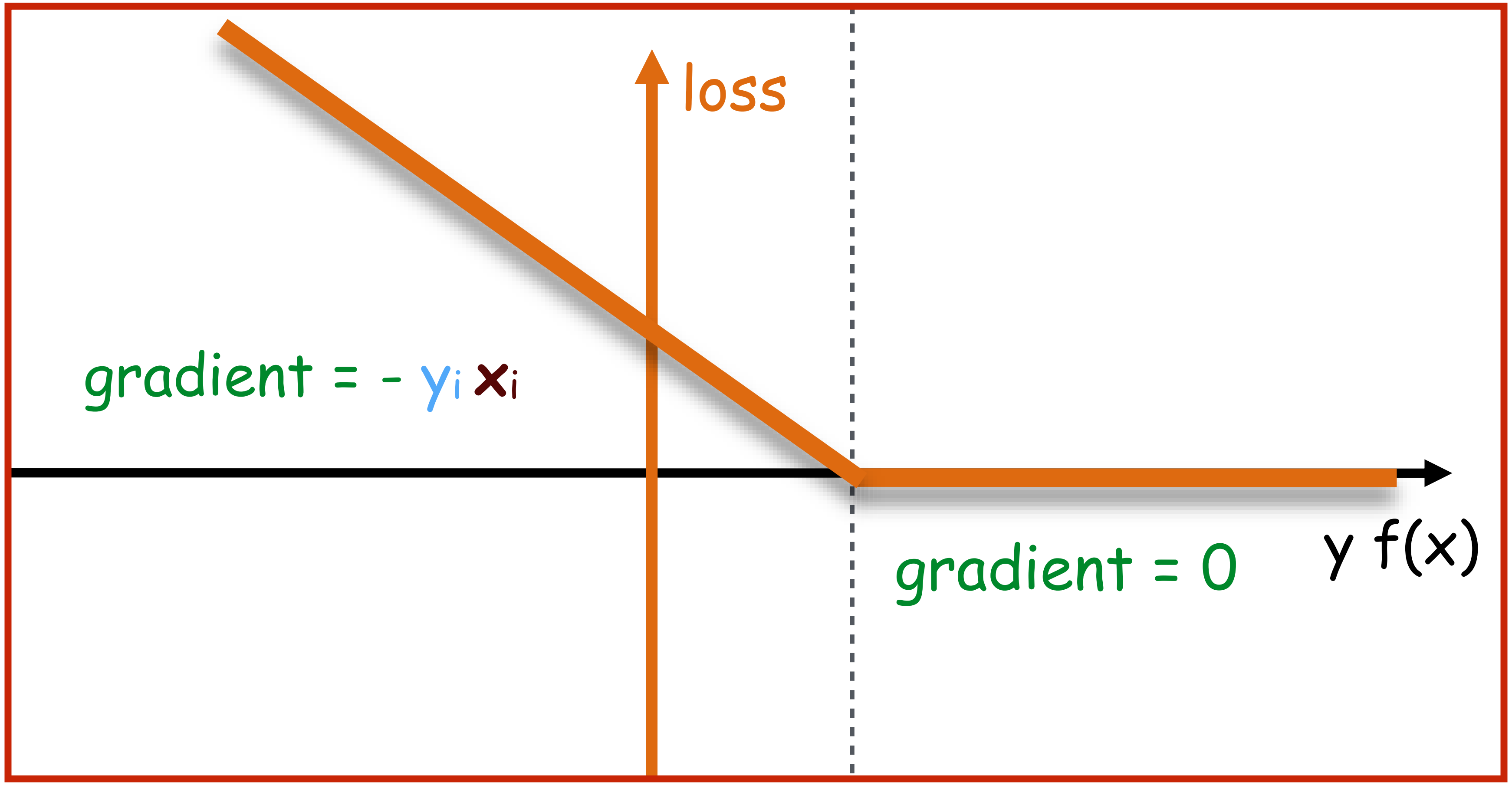
"Hinge" Loss

Gradient of Loss Function ?

$$\underset{w,\ b}{\text{minimize}} \quad \frac{1}{2}\|w\|_2^2 \quad + \quad C \sum_{i=1}^{n} \max(0,\ 1 - y_i f(x_i))$$

Maximize Margin — Minimize Hinge Loss

$\max(0, 1 - y f(x))$ Convex but ...



Sub-Gradient

Subgradient (on one data point)

$$L = \sum_i \left(\frac{\lambda}{2} \|w\|_2^2 + \max(0,\ 1 - y_i f(x_i)) \right) \quad \text{let } \lambda = \frac{1}{nC}$$

on one data point

if $1 - y_i f(x_i) > 0$	if $1 - y_i f(x_i) \leq 0$
$\frac{\partial L}{\partial w} = \lambda w - y_i x_i$	$\frac{\partial L}{\partial w} = \lambda w$
$\frac{\partial L}{\partial b} = -y_i$	$\frac{\partial L}{\partial b} = 0$

Linear SVM (training)

initialize $\mathbf{w}$ and b

Loop for n_epoch iterations:

Loop for each training instance ($\mathbf{x}$, y) in training set

compute subgradients

$$\frac{\partial L}{\partial \mathbf{w}} \quad \frac{\partial L}{\partial b}$$

update the parameters $\mathbf{w}$ and b

$$w \longleftarrow w - a \frac{\partial L}{\partial \mathbf{w}}$$

$$b \longleftarrow b - a \frac{\partial L}{\partial b}$$

a is a constant ("Learning Rate")

https://scikit-learn.org/stable/modules/generated/sklearn.svm.LinearSVC.html#sklearn.svm.LinearSVC

Other SVM Variations

- nu SVM
  - nu parameter controls:
    - Fraction of support vectors (lower bound) and misclassification rate (upper bound)
    - E.g., $\nu = 0.05$ guarantees that ≥ 5% of training points are SVs and training error rate is ≤ 5%
  - Harder to optimize than C-SVM and not as scalable
- SVMs for regression
- SVMs for clustering

...

http://scikit-learn.org/stable/modules/svm.html

Conclusion

- SVMs find optimal linear separator
- The kernel trick makes SVMs learn non-linear decision surfaces

- Strength of SVMs:
  - Good theoretical and empirical performance
  - Supports many types of kernels

- Disadvantage of SVMs:
  - Need to choose the kernel (and tune its parameters)

Quiz3

- Quiz3 will be taken on July 9 - it will be available only during the day

- The coverage will be from Neural Network (lecture 4-2) to SVM (lecture 7-2)

Ensemble Learning

Ensemble Learning

Consider a set of classifiers $h_1, \ldots, h_L$ <= called ensemble

Idea: construct a classifier $H(\mathbf{x})$ that combines the individual decisions of $h_1, \ldots, h_L$

- e.g., could have the member classifiers vote, or
- e.g., could use different members for different regions of the instance space
- Works well if the members each have low error rates

Successful ensembles require **diversity**

- Classifiers should make different mistakes
- Can have different types of base learners

Practical Application: Netflix Prize

Goal: predict how a user will rate a movie

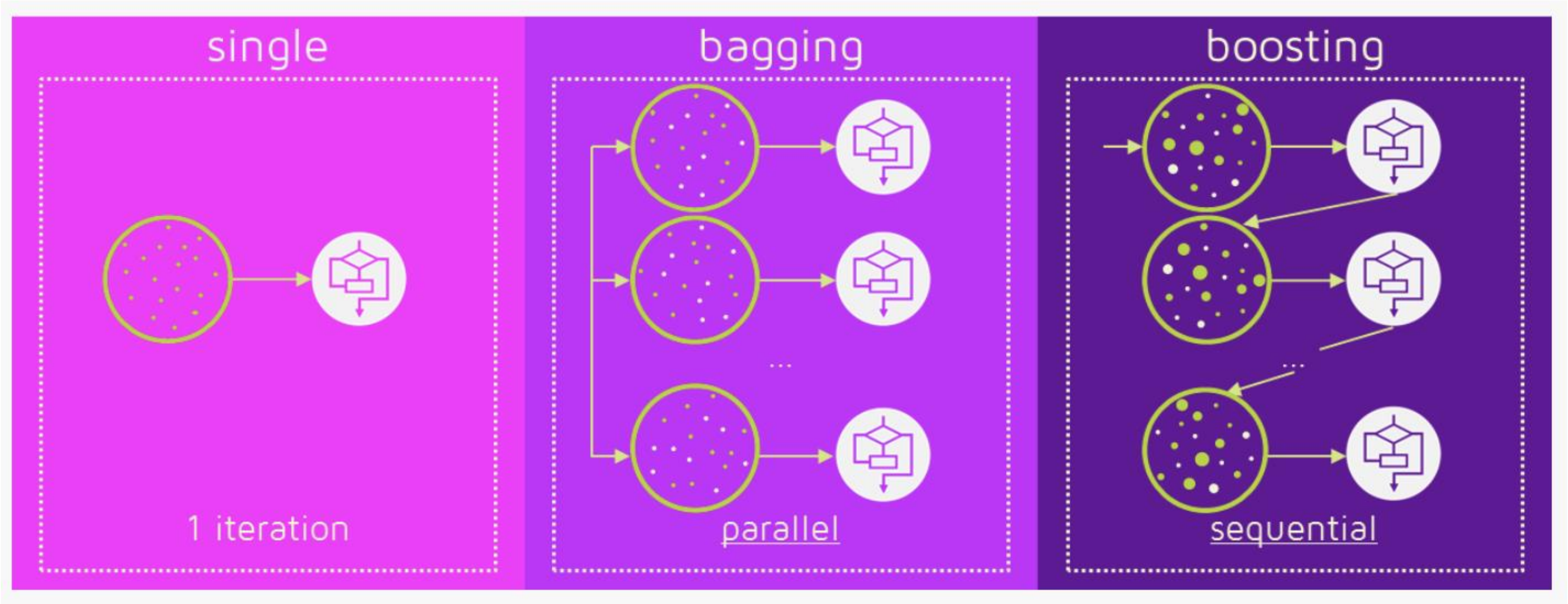
- Based on the user's ratings for other movies and other peoples' ratings
- with no other information about the movies

This application is called "collaborative filtering"

Neflix Prize: $1M to the first team to do 10% better than Netflix' system (2007-2009)

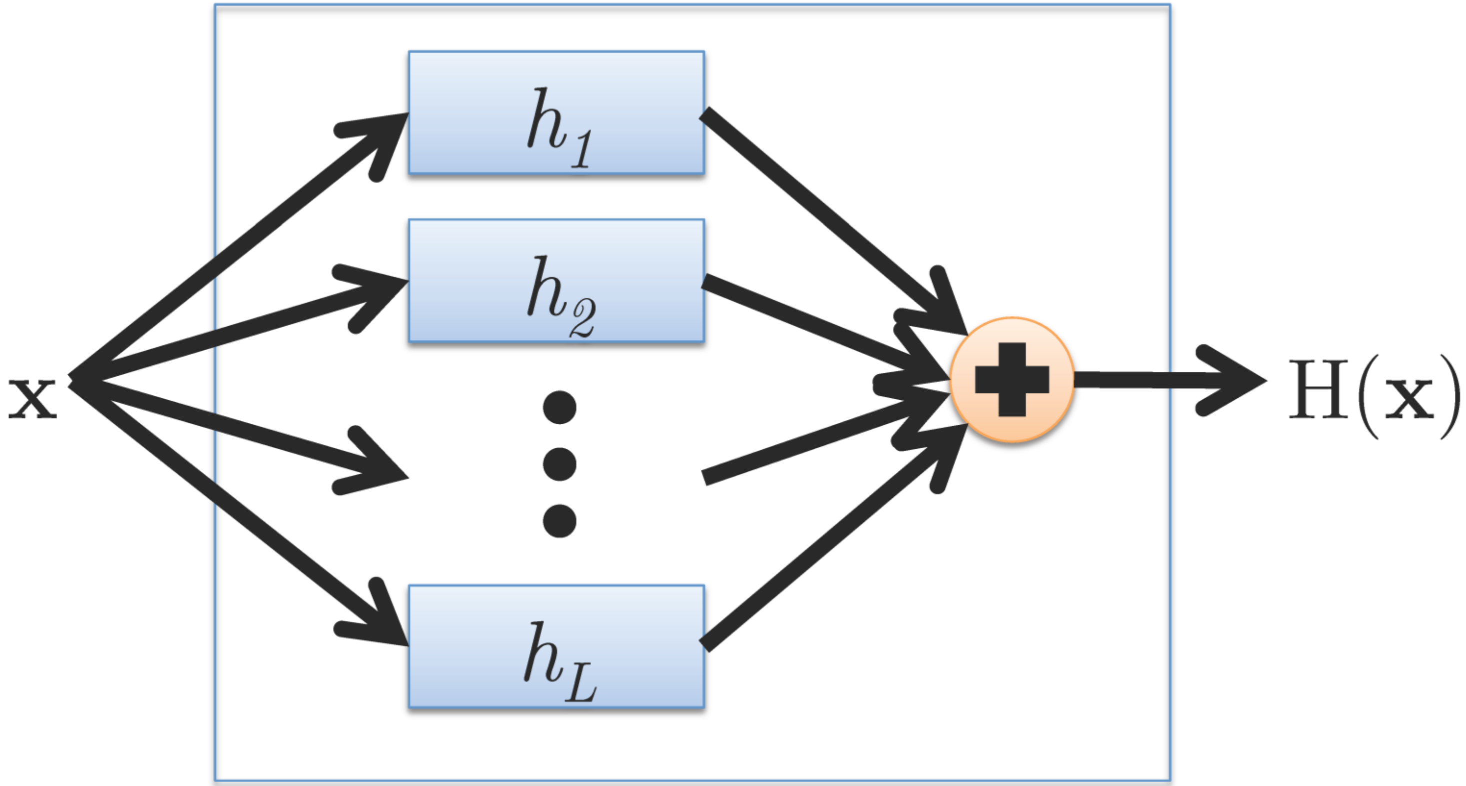
Winner: BellKor's Pragmatic Chaos – an ensemble of more than 800 rating systems

Two Types of Ensemble Methods



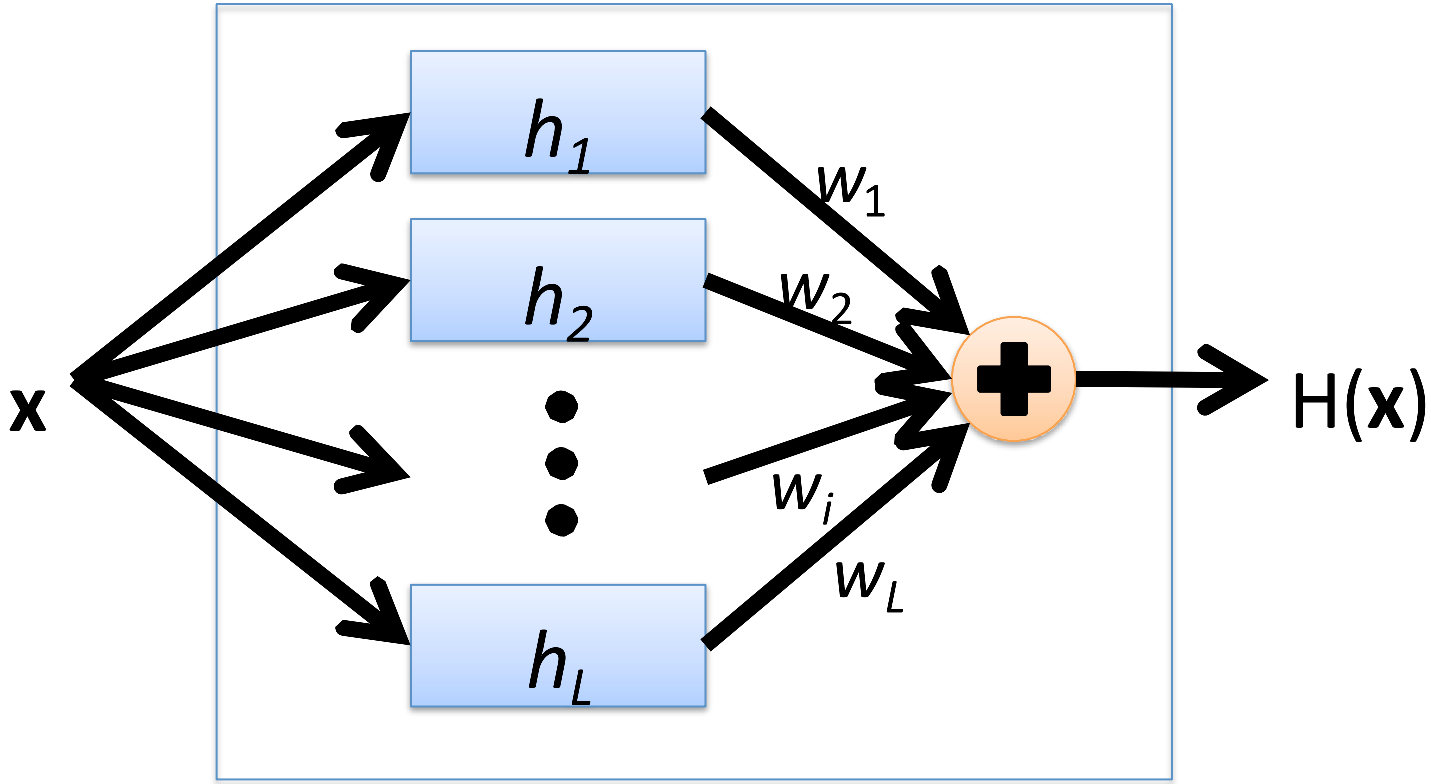
- Bagging: Build independent classifiers
- Boosting: Build sequential classifiers

Combining Classifiers: Averaging



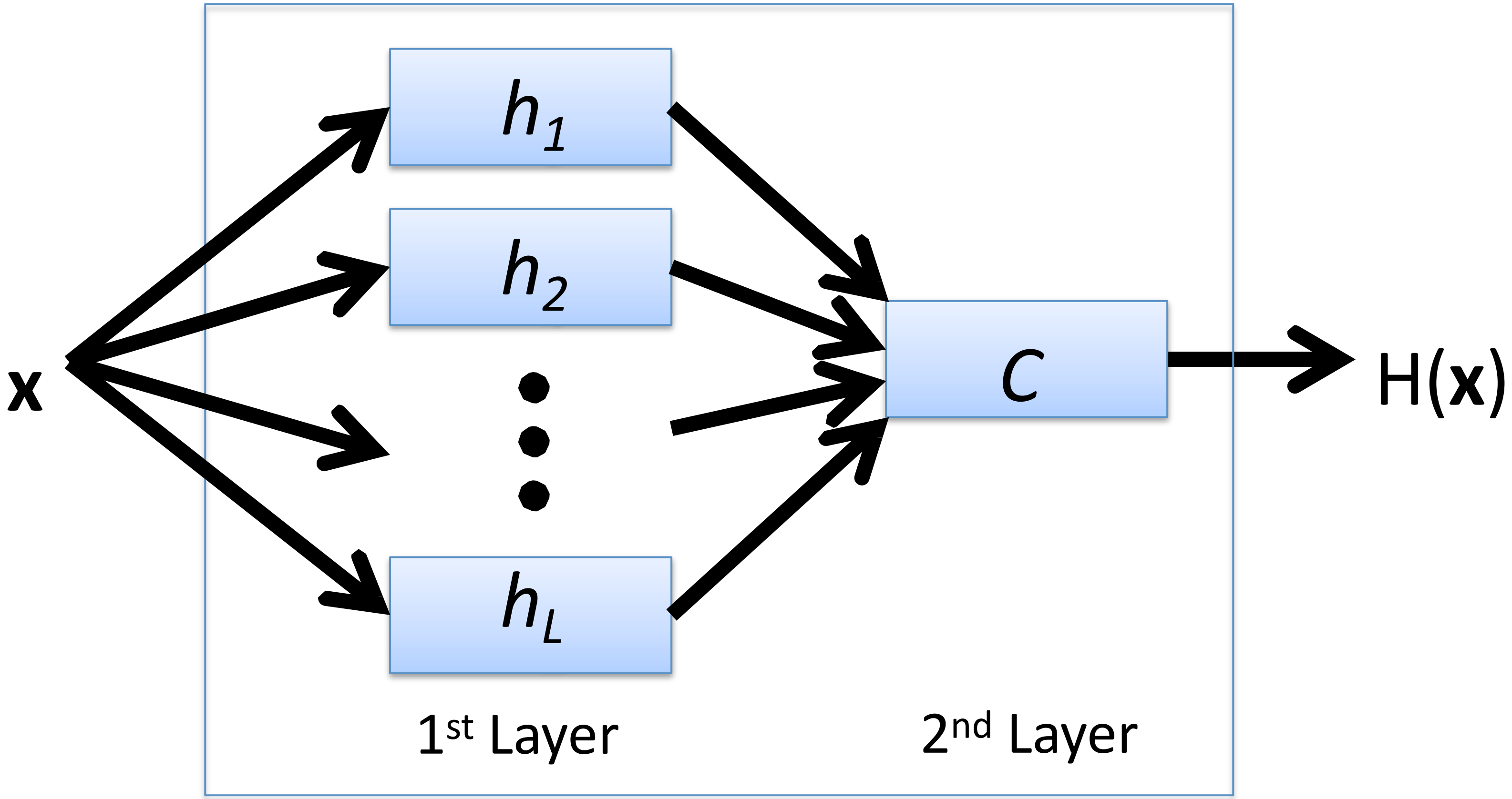
- Final hypothesis is a simple vote of the members

Combining Classifiers: Weighted Average



- Coefficients of individual members are trained using a validation set

Combining Classifiers: Stacking



- Predictions of 1st layer used as input to 2nd layer
- Train 2nd layer on validation set

How to Achieve Diversity

Cause of the Mistake	Diversification Strategy
Overfitting	Vary the training sets
Some features are noisy	Vary the set of input features

Manipulating the Training Data

- **Bagging:**
  - Create bootstrap replicates of training set
    - **Bootstrap replication:** Given *n* training examples, construct a new training set by sampling *n* instances with replacement
  - Train a classifier (e.g., a decision tree) for each replicate
  - Estimate classifier performance using out-of-bootstrap
  - Average output of all classifiers

Manipulating the Features

Random Forests

- Construct decision trees on bootstrap replicas
  - Restrict the node decisions to a small subset of features picked randomly for each node
- Do not prune the trees
  - Estimate tree performance on out-of-bootstrap data
- Average the output of all trees