Machine Learning 10-601

Tom M. Mitchell
Machine Learning Department
Carnegie Mellon University

December 4, 2017

Today:

Review for final exam

Recommended reading:

- Mitchell: “Key Ideas in ML”

Loss Functions

AdaBoost Algorithm

Given: $(x_1, y_1), \ldots, (x_m, y_m)$ where $x_i \in X$, $y_i \in Y = \{-1, +1\}$
Initialize $D_1(i) = 1/m$.
For $t = 1, \ldots, T$:

- Train weak learner using distribution D_t.
- Get weak hypothesis $h_t : X \to \{-1, +1\}$ with error

$$\epsilon_t = \Pr_{i \sim D_t}\left[h_t(x_i) \neq y_i\right].$$

- Choose $\alpha_t = \frac{1}{2} \ln\left(\frac{1 - \epsilon_t}{\epsilon_t}\right)$.
- Update:

$$\begin{aligned} D_{t+1}(i) &= \frac{D_t(i)}{Z_t} \times \begin{cases} e^{-\alpha_t} & \text{if } h_t(x_i) = y_i \\ e^{\alpha_t} & \text{if } h_t(x_i) \neq y_i \end{cases} \\ &= \frac{D_t(i) \exp(-\alpha_t y_i h_t(x_i))}{Z_t} \end{aligned}$$

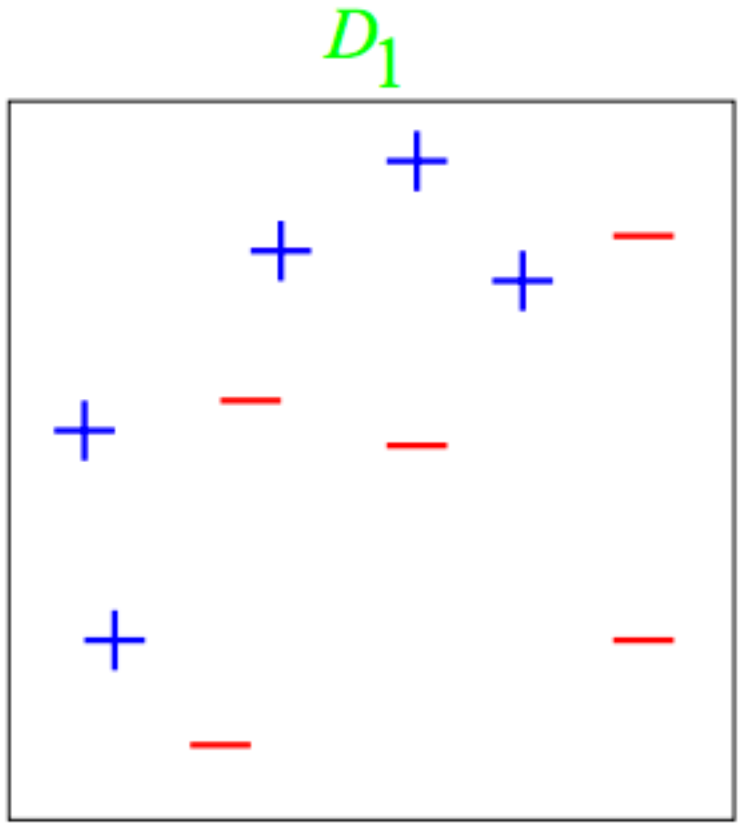
where Z_t is a normalization factor (chosen so that D_{t+1} will be a distribution).

Output the final hypothesis:

$$H(x) = \operatorname{sign}\left(\sum_{t=1}^{T} \alpha_t h_t(x)\right).$$

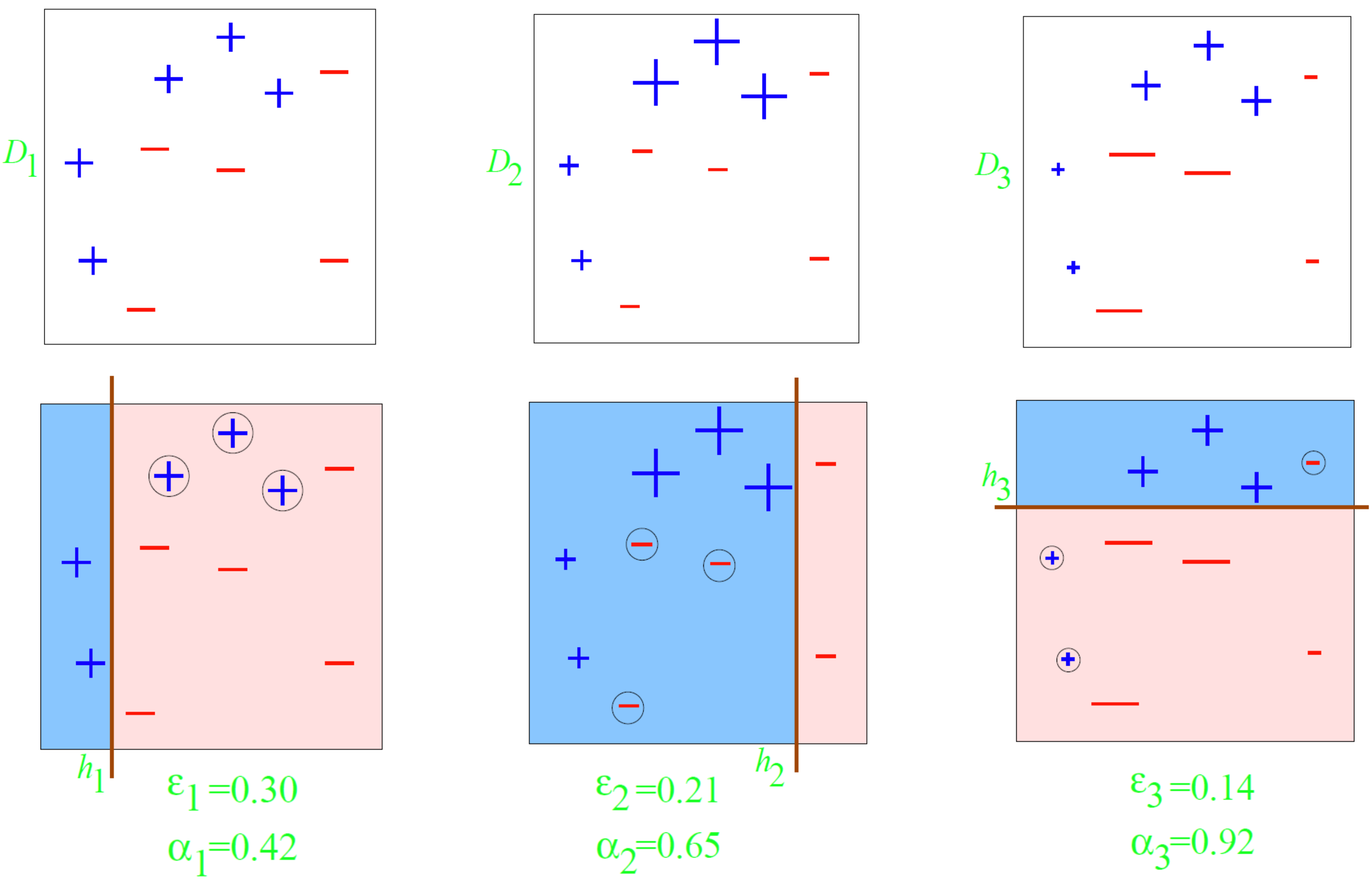
AdaBoost: A toy example

Weak classifiers: vertical or horizontal half-planes (a.k.a. decision stumps)

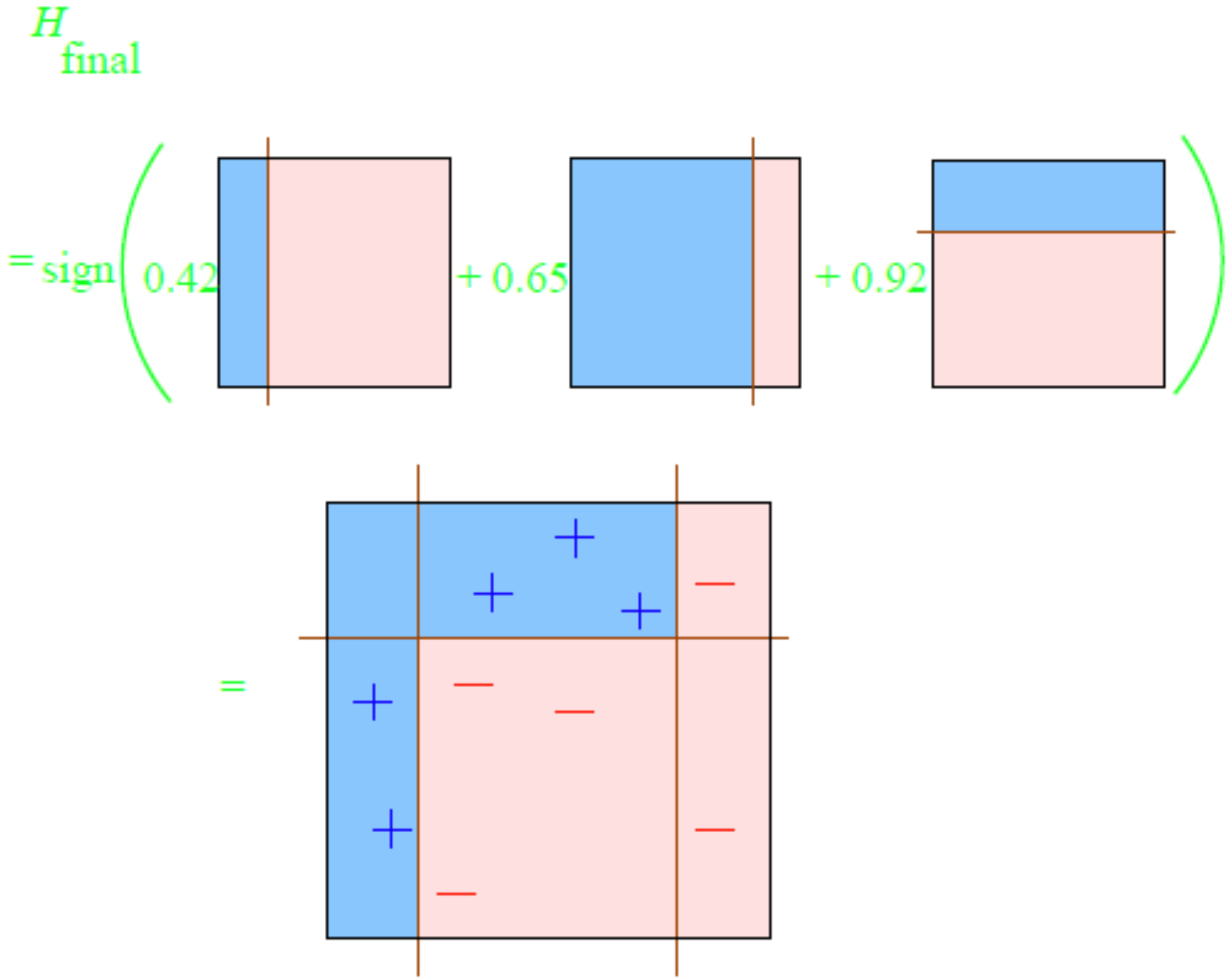


[Rob Schapire]

AdaBoost: A toy example



AdaBoost: A toy example



[Rob Schapire]

Theoretical Result 1: Training Error

We can minimize this bound by choosing α_t on each iteration to minimize Z_t.

$$Z_t = \sum_{i=1}^{m} D_t(i) \exp(-\alpha_t y_i h_t(x_i))$$

For boolean target function, this is accomplished by [Freund & Schapire '97]:

$$\boxed{\alpha_t = \frac{1}{2} \ln \left(\frac{1 - \epsilon_t}{\epsilon_t} \right)}$$

Where:

$$\boxed{\epsilon_t = \sum_{i=1}^{m} D_t(i) \delta(h_t(x_i) \neq y_i)}$$

Logistic Regression

Logistic regression assumes:

$$P(Y=1|X=\mathbf{x}) = \frac{1}{1+\exp(-(\mathbf{wx}+b))}$$

Equivalently if $y \in \{-1,+1\}$:

$$P(Y=y|X=\mathbf{x}) = \frac{1}{1+\exp(-y(\mathbf{wx}+b))}$$

And trains to minimize log loss:

$$loss = \sum_{j=1}^{m} \ln(1+\exp(-y_j(\mathbf{wx_j}+b)))$$

Logistic regression and Boosting

Logistic regression:

- Minimize loss fn

$$\sum_{i=1}^{m} \ln(1 + \exp(-y_i f(x_i)))$$

- Define

$$f(x) = \sum_k w_k x_k + b$$

where x_j predefined

Boosting:

- Minimize loss fn

$$\sum_{i=1}^{m} \exp(-y_i f(x_i))$$

- Define

$$f(x) = \sum_t \alpha_t h_t(x)$$

where $h_t(x_i)$ defined dynamically to fit data

- Weights α_j learned incrementally

Boosting and Logistic Regression

Logistic regression equivalent to minimizing log loss

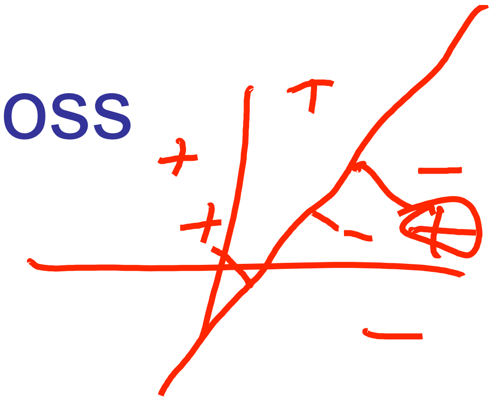
$$\sum_{i=1}^{m} \ln(1 + \exp(-y_i f(x_i)))$$

Boosting minimizes similar loss function!!

$$\frac{1}{m}\sum_{i} \exp(-y_i f(x_i)) = \prod_{t} Z_t$$

Both smooth approximations of 0/1 loss!

Slack variables – Hinge loss



Complexity penalization

$$\xi_j = \mathrm{loss}(f(x_j), y_j)$$

$$f(x_j) = \mathrm{sgn}(\mathbf{w} \cdot \mathrm{x_j} + \mathrm{b})$$

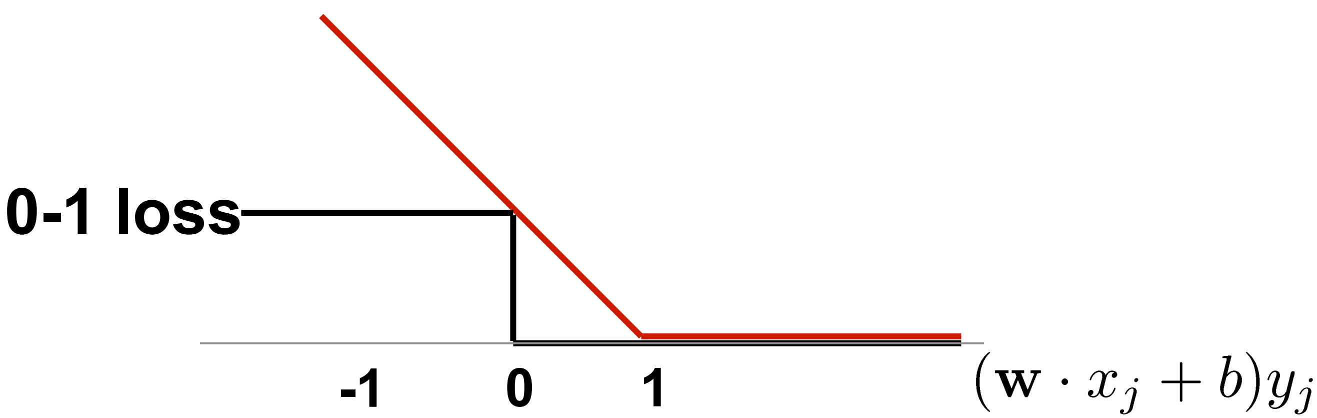
$$\min_{\mathbf{w},b} \ \mathbf{w}^T\mathbf{w} + C \sum_j \xi_j$$

$$\text{s.t. } (\mathbf{w}^T\mathbf{x}_j + b)\, y_j \geq 1 - \xi_j \quad \forall j$$

$$\xi_j \geq 0 \quad \forall j$$

$$\xi_j = (1 - (\mathbf{w} \cdot x_j + b)y_j))_+$$

Hinge loss



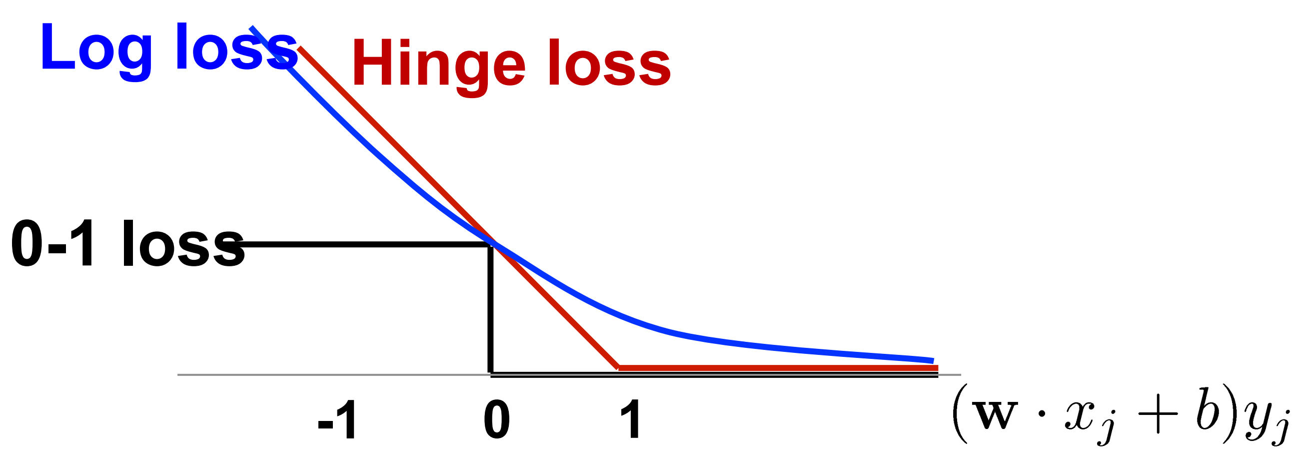
SVM vs. Logistic Regression

SVM : **Hinge loss**

$$\operatorname{loss}(f(x_j), y_j) = (1 - (\mathbf{w} \cdot x_j + b) y_j)_+$$

Logistic Regression : **Log loss** (negative log conditional likelihood)

$$\operatorname{loss}(f(x_j), y_j) = -\log P(y_j \mid x_j, \mathbf{w}, b) = \log(1 + e^{-(\mathbf{w} \cdot x_j + b) y_j})$$



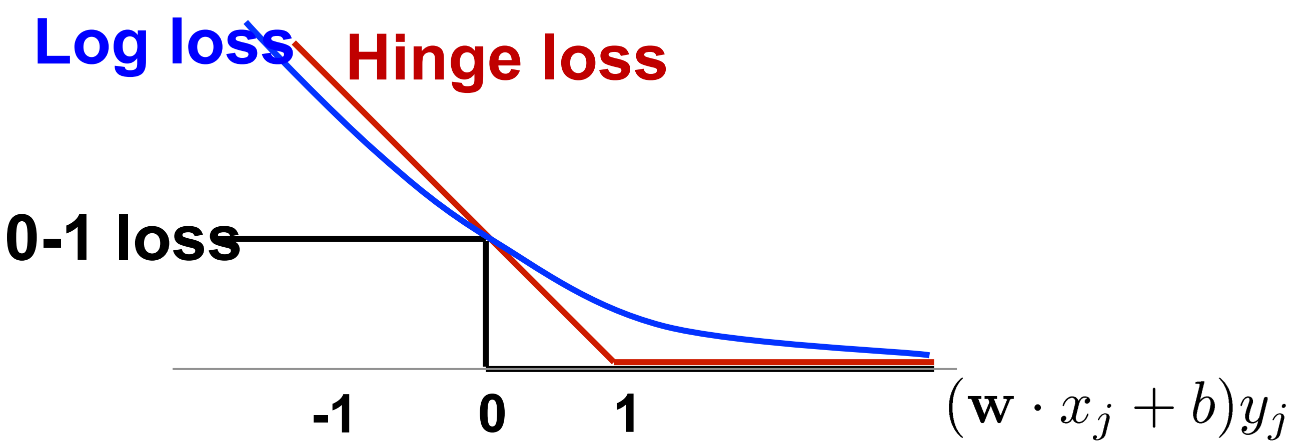
Boosting : $loss(\mathbf{x_j}, y_j) = \exp(-(y_j \sum_t \alpha_t h_t(\mathbf{x_j})))$

SVM : **Hinge loss**

$$loss(\mathbf{x_j}, y_j) = (1 - (\mathbf{w} \cdot x_j + b) y_j))_+$$

Logistic Regression : **Log loss** (negative log conditional likelihood)

$$loss(\mathbf{x_j}, y_j) = -\log P(y_j \mid x_j, \mathbf{w}, b) = \log(1 + e^{-(\mathbf{w} \cdot x_j + b) y_j})$$



Loss Functions

- MLE, MAP
- log loss = negative log likelihood
  - minimize log loss = minimize misassigned probability mass
- regression: sum of squared errors
- regression: sum of squared errors plus square of weights
- SVM Hinge loss (maximize margin with slack variables)
- AdaBoost
- Bayes nets
- Logistic regression

What You Should Know

- Sample complexity varies with the learning setting
  - Learner actively queries trainer
  - Examples arrive at random
  - …
- Within the PAC learning setting, we can bound the probability that learner will output hypothesis with given error
  - For ANY consistent learner (case where $c \in H$)
  - For ANY "best fit" hypothesis (agnostic learning, where perhaps c not in H)
- VC dimension as measure of complexity of H
- Mistake bounds
- Conference on Learning Theory: http://www.learningtheory.org
- ML Department course on Machine Learning Theory

Kernels : Key Points

- Many learning tasks are framed as optimization problems
- Primal and Dual formulations of optimization problems
- Dual version framed in terms of dot products between x's
- Kernel functions $k(x,y)$ allow calculating dot products $\langle \Phi(x),\Phi(y) \rangle$ without bothering to project x into $\Phi(x)$
- Leads to major efficiencies, and ability to use very high dimensional (virtual) feature spaces

What you should know

Primal and Dual optimization problems

Kernel functions

Support Vector Machines

- Maximizing margin
- Kernel SVM’s
- Noise, slack variables and hinge loss
- Relationship between SVMs and logistic regression
  - 0/1 loss
  - Hinge loss
  - Log loss

Theory shows overfitting, mistakes depends on margin size

What you should know

1. Using unlabeled data to reweight labeled examples gives better approximation to true error
   - If we assume examples drawn from fixed P(X)

2. Unlabeled can help EM learn Bayes nets for P(X,Y), and thus P(Y|X)
   - If we assume the Bayes net structure reflects cond. independencies

2.5. Transductive SVM's
   - If we assume maximizing margin captures relationship between P(X) and f: X→Y

3. Jointly train multiple classifiers, coupled by consistency constraints that can be evaluated using unlabeled data
   - optimize both the fit to labeled examples, and satisfaction of the consistency constraints

What You Should Know

- Ensemble methods
- Weighted Majority
  - Learns weights for a given pool of hypotheses
  - Mistake bound relative to best hypothesis in the pool
  - …
- Boosting
  - Learns weigths and hypotheses
  - Theory: training error, true error, correspondence to Log. Regression
  - Practice: Boosted decision trees (and stumps) very popular!
- Many variants of ensemble methods
  - Resample training data to generate variety
  - Randomize learning algorithm to generate variety
  - Active learning – choose examples where vote is closest to tie

Representation Learning: What you should know

- Unsupervised dimension reduction using all features
  - Principle Components Analysis
    - Minimize reconstruction error
  - Singular Value Decomposition
    - Efficient PCA
  - Independent components analysis
  - Canonical correlation analysis
  - Probabilistic models with latent variables
- Supervised dimension reduction
  - Hidden layers of Neural Networks
    - Most flexible, local minima issues
- LOTS of ways of combining discovery of latent features with classification tasks

Bias, Variance, Unavoidable Error

Bias in stats bias $\hat{\theta}$ is $E[\hat{\theta}] - \theta$

take over $P(x)$

Bias in ML involves estimating f_n

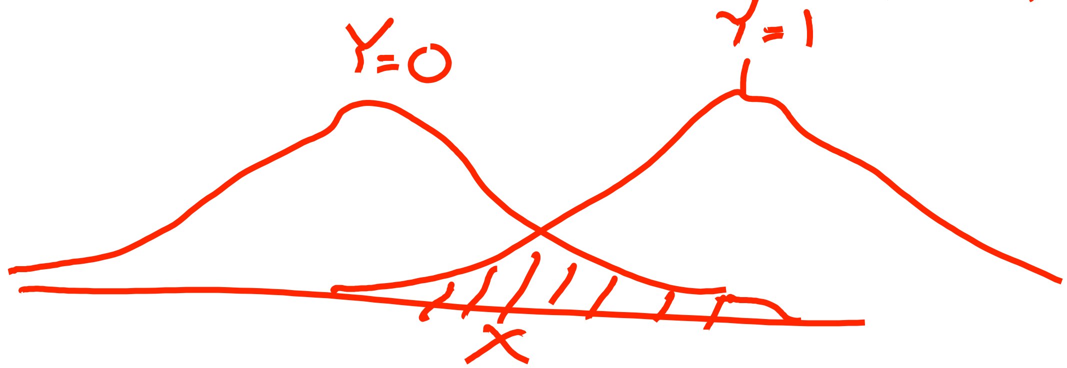
incomplete H

alg choosing $h \in H$ has biased pref (eg. short trees)

Variance – due to finite samples of training data

statistical anomalies, unrepresentative data

Unavoidable error for nondeterministic fns.



Transforming Inputs, Then Classifying

- Neural net, kernel SVM, AdaBoost, WeightedMajority

Perspectives on ML

- Optimization
- Probabilistic modeling
- Parameterized programs
- Evolution

optimal policy

$$Q^{\pi^*}(s,a) = r(s,a) + \gamma V(s') = r(s,a) + \gamma \max_{a'} Q^{\pi^*}(s',a')$$

next state

π : St $\Rightarrow$ Action

$$f : X \to Y$$

f : State $\to$ action

V^{π} : State $\to \mathbb{R}$

sum of discounted future rewards using π from this state

Q^{π} : state $\times$ action $\to \mathbb{R}$

Key Results

- No free lunch
- Sources of error: bias, variance, unavoidable error
- Overfitting
- Bayes nets
- Deep neural nets
- PAC learning theory
- Semi-supervised learning
- Learning ensembles of functions
- Representation learning
- Kernel methods
- Reinforcement learning