Caltech

Machine Learning & Data Mining

CS/CNS/EE 155

Lecture 3:

SVM, Logistic Regression, Neural Nets, Evaluation Metrics

Announcements

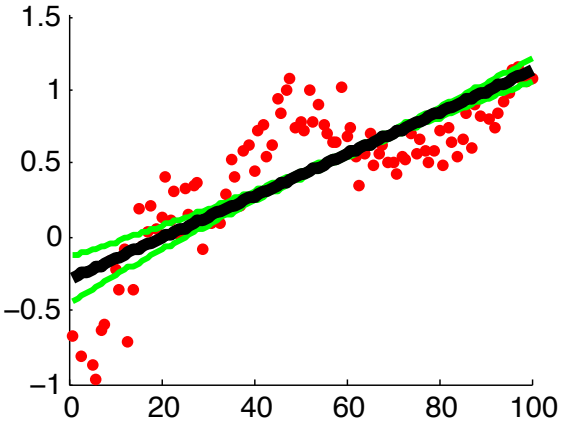
- HW1 Due
  - Will be graded in about a week
- HW2 released
  - Due Jan 19th at 2pm
  - Via Moodle

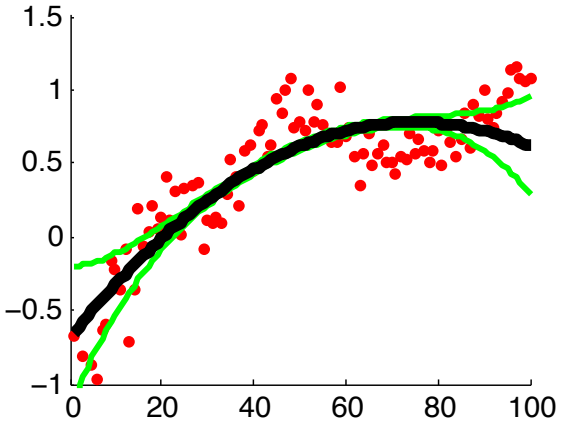
Recap: Basic Recipe

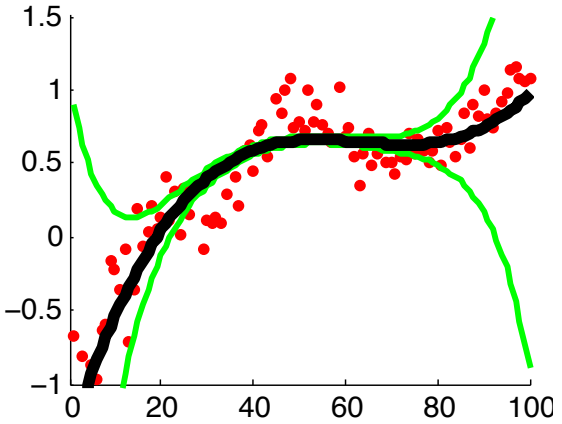
- Training Data: $S = \{(x_i, y_i)\}_{i=1}^{N}$ $x \in R^D$ $y \in \{-1, +1\}$
- Model Class: $f(x \mid w, b) = w^T x - b$ **Linear Models**
- Loss Function: $L(a, b) = (a - b)^2$ **Squared Loss**
- Learning Objective: $\underset{w,b}{\operatorname{argmin}} \sum_{i=1}^{N} L\left(y_i, f(x_i \mid w, b)\right)$

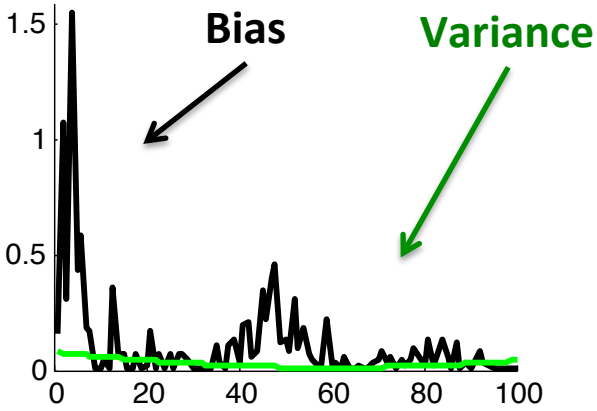
Optimization Problem

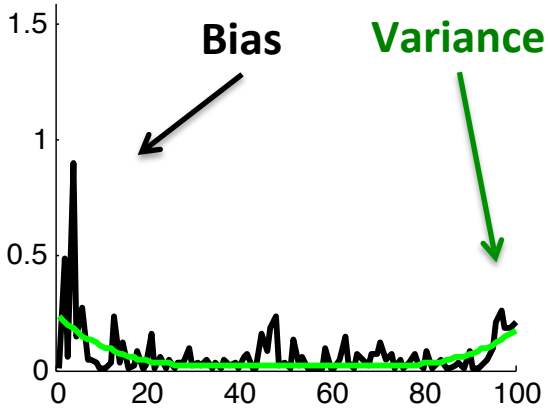
Recap: Bias-Variance Trade-off

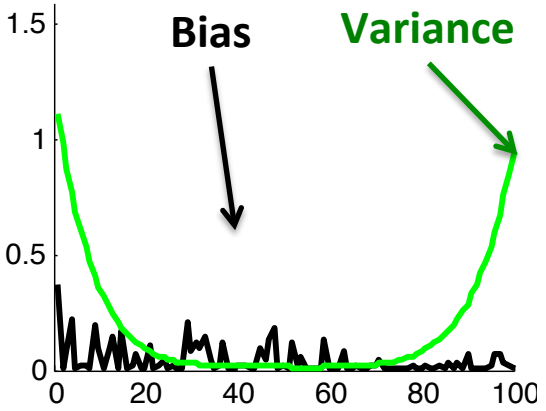












Recap: Complete Pipeline

$$S = \{(x_i, y_i)\}_{i=1}^{N}$$

Training Data

$$f(x \mid w, b) = w^T x - b$$

Model Class(es)

$$L(a, b) = (a - b)^2$$

Loss Function

$$\underset{w,b}{\operatorname{argmin}} \sum_{i=1}^{N} L\left(y_i, f(x_i \mid w, b)\right)$$

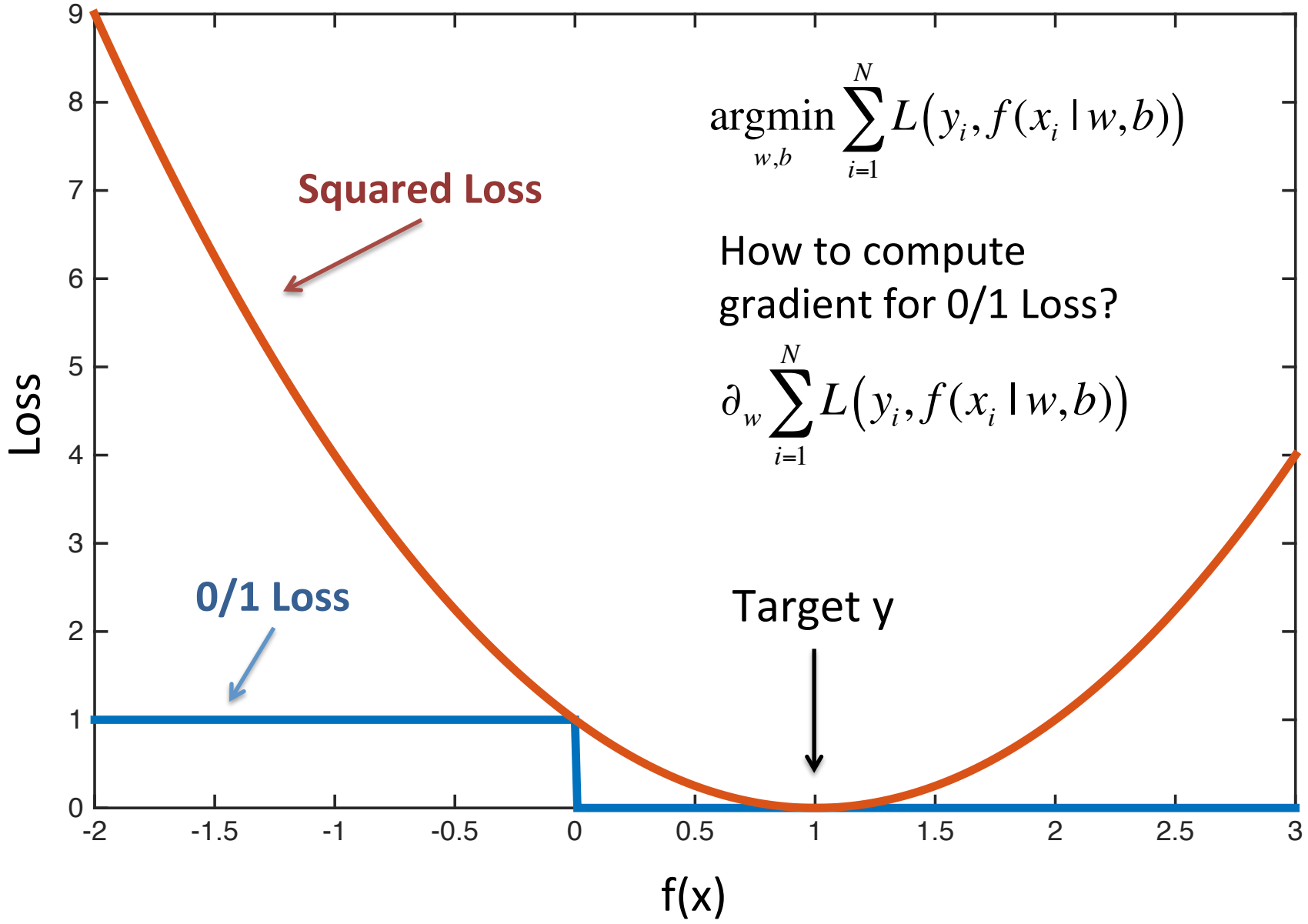
SGD!

Cross Validation & Model Selection

Profit!

Today

- **Beyond Basic Linear Models**
  - **Support Vector Machines**
  - **Logistic Regression**
  - **Feed-forward Neural Networks**
  - **Different ways to interpret models**
- Different Evaluation Metrics



$$\underset{w,b}{\operatorname{argmin}} \sum_{i=1}^{N} L\left(y_i, f(x_i \mid w,b)\right)$$

How to compute gradient for 0/1 Loss?

$$\partial_w \sum_{i=1}^{N} L\left(y_i, f(x_i \mid w,b)\right)$$

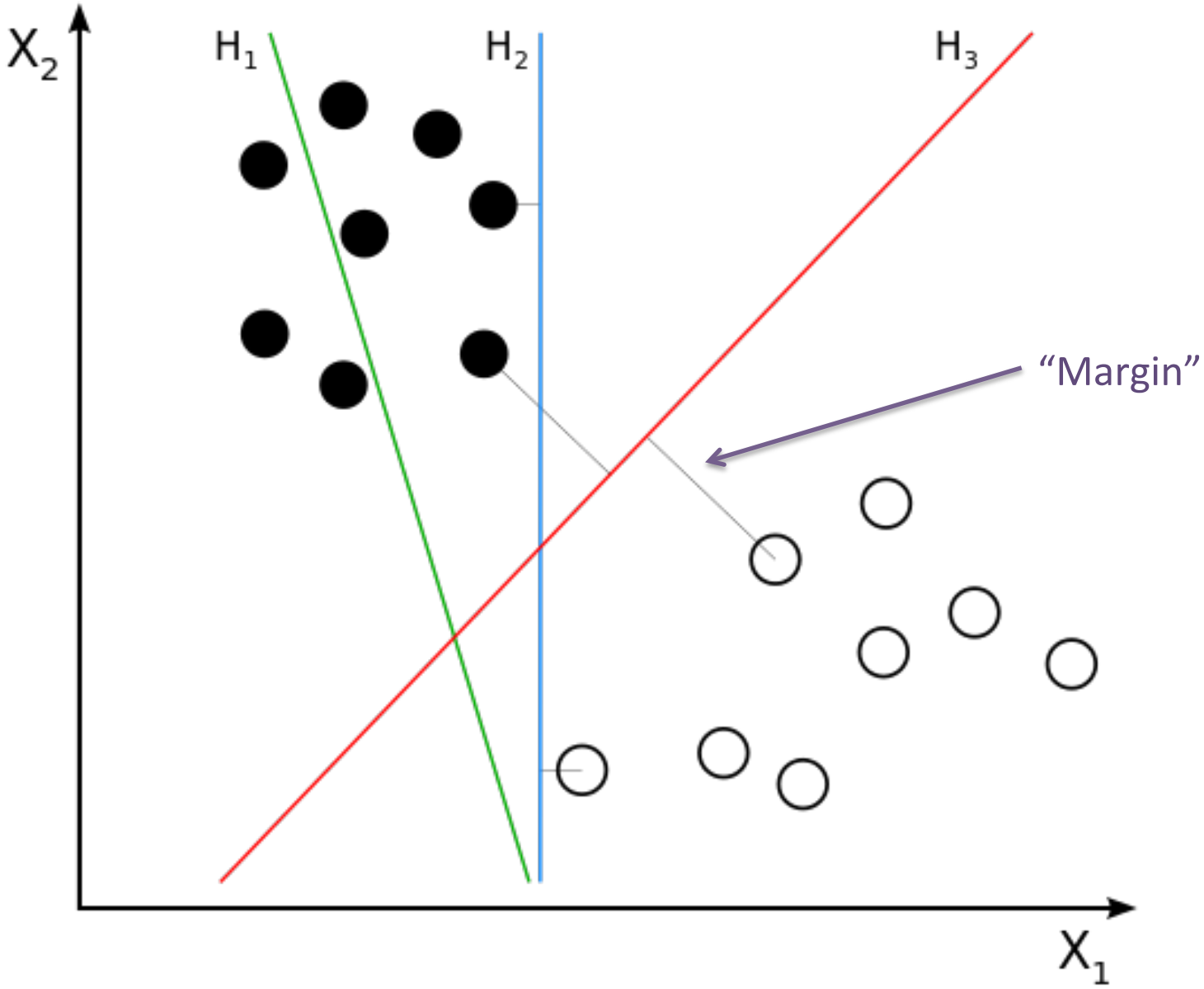
Recap: 0/1 Loss is Intractable

- 0/1 Loss is flat or discontinuous everywhere
- VERY difficult to optimize using gradient descent
- **Solution:** Optimize surrogate Loss
  - **Today: Hinge Loss (…eventually)**

Support Vector Machines

aka Max-Margin Classifiers

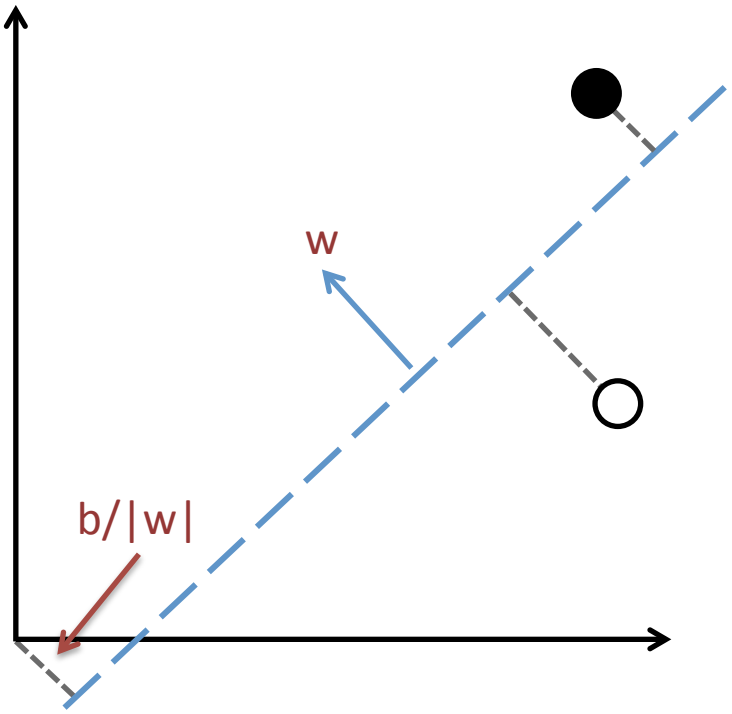
Which Line is the Best Classifier?



Source: http://en.wikipedia.org/wiki/Support_vector_machine

Recall: Hyperplane Distance

- Line is a 1D, Plane is 2D
- Hyperplane is many D
  - Includes Line and Plane
- Defined by (w,b)
- Distance: $\frac{|w^T x - b|}{\|w\|}$
- Signed Distance: $\frac{w^T x - b}{\|w\|}$

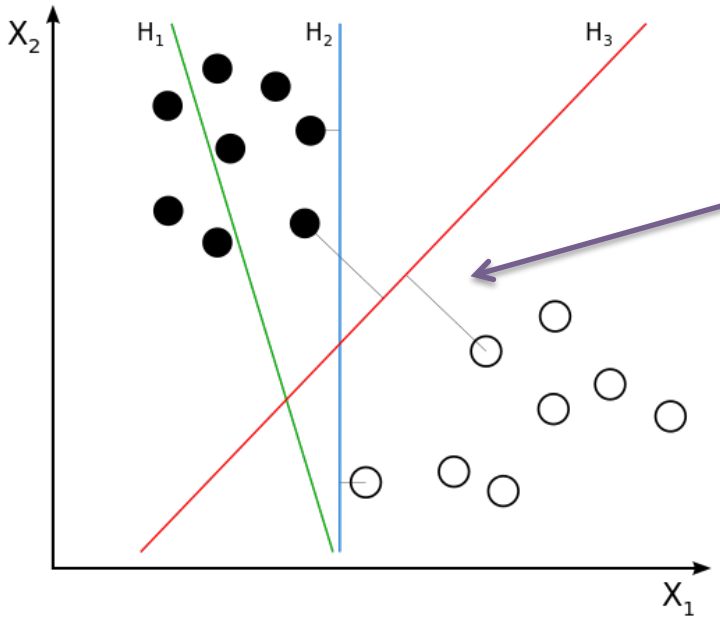


Linear Model = un-normalized signed distance!

How to Maximize Margin?
(Assume Linearly Separable)

Choose w that maximizes:

$$\underset{w,b}{\operatorname{argmax}}\left[\min_{(x,y)}\frac{y\left(w^T x-b\right)}{\|w\|}\right]$$



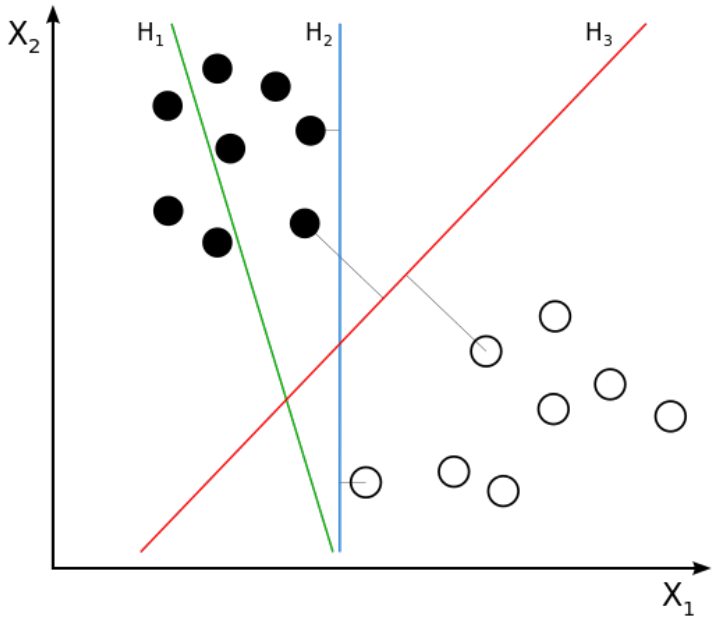
Margin

How to Maximize Margin?

(Assume Linearly Separable)

$$\underset{w,b}{\operatorname{argmax}}\left[\min_{(x,y)} \frac{y\left(w^T x - b\right)}{\|w\|}\right]$$

$$\equiv \underset{w,b:\,\|w\|=1}{\operatorname{argmax}}\left[\min_{(x,y)} y\left(w^T x - b\right)\right]$$



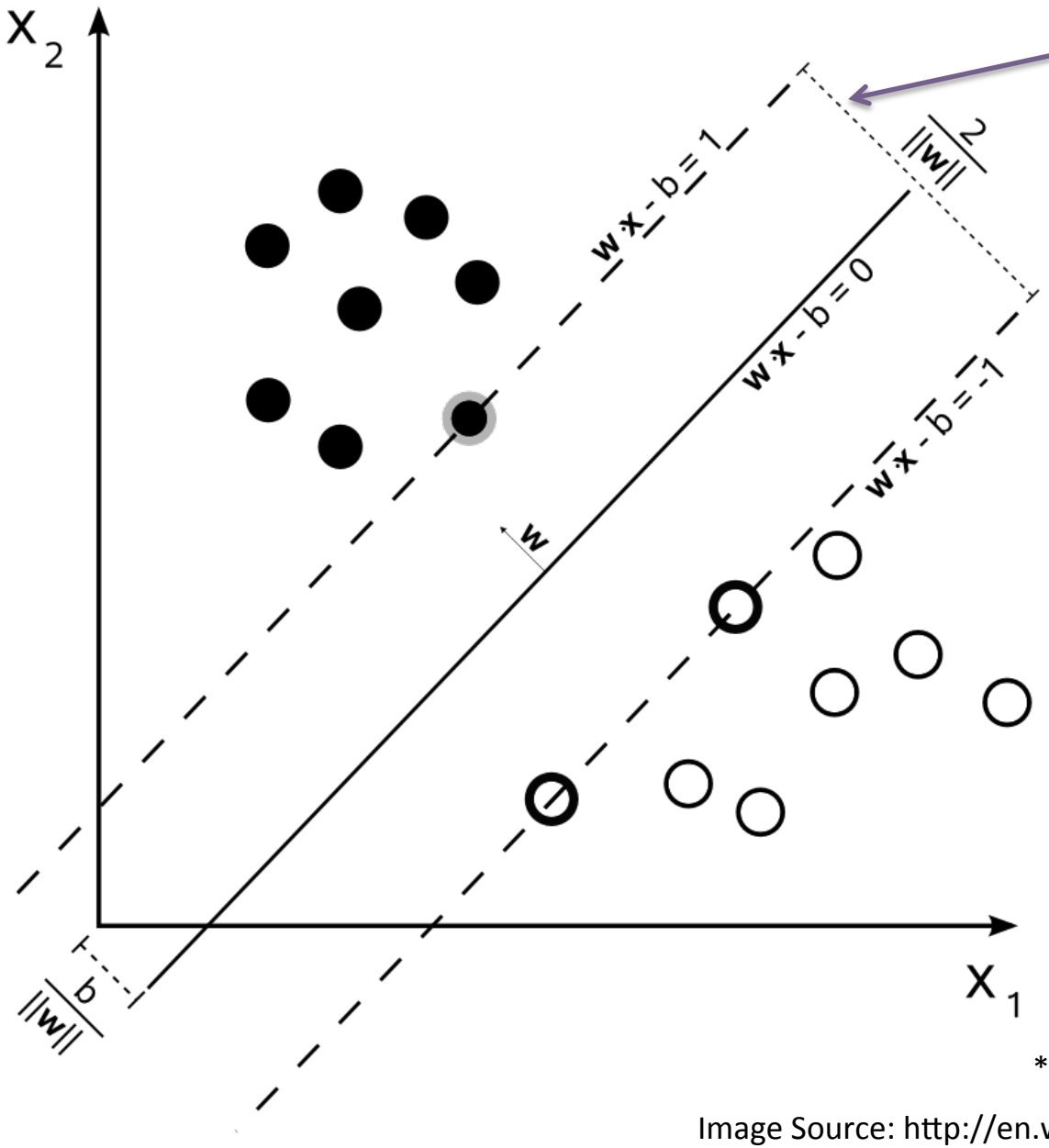
Suppose we enforce:

$$\min_{(x,y)} y\left(w^T x - b\right) = 1$$

Then:

$$= \underset{w,b}{\operatorname{argmin}}\|w\| \equiv \underset{w,b}{\operatorname{argmin}}\|w\|^2$$

Max Margin Classifier (Support Vector Machine)



$$\underset{w,b}{\operatorname{argmin}} \frac{1}{2} w^T w \equiv \frac{1}{2}\|w\|^2$$

$$\forall i : y_i \left(w^T x_i - b \right) \geq 1$$

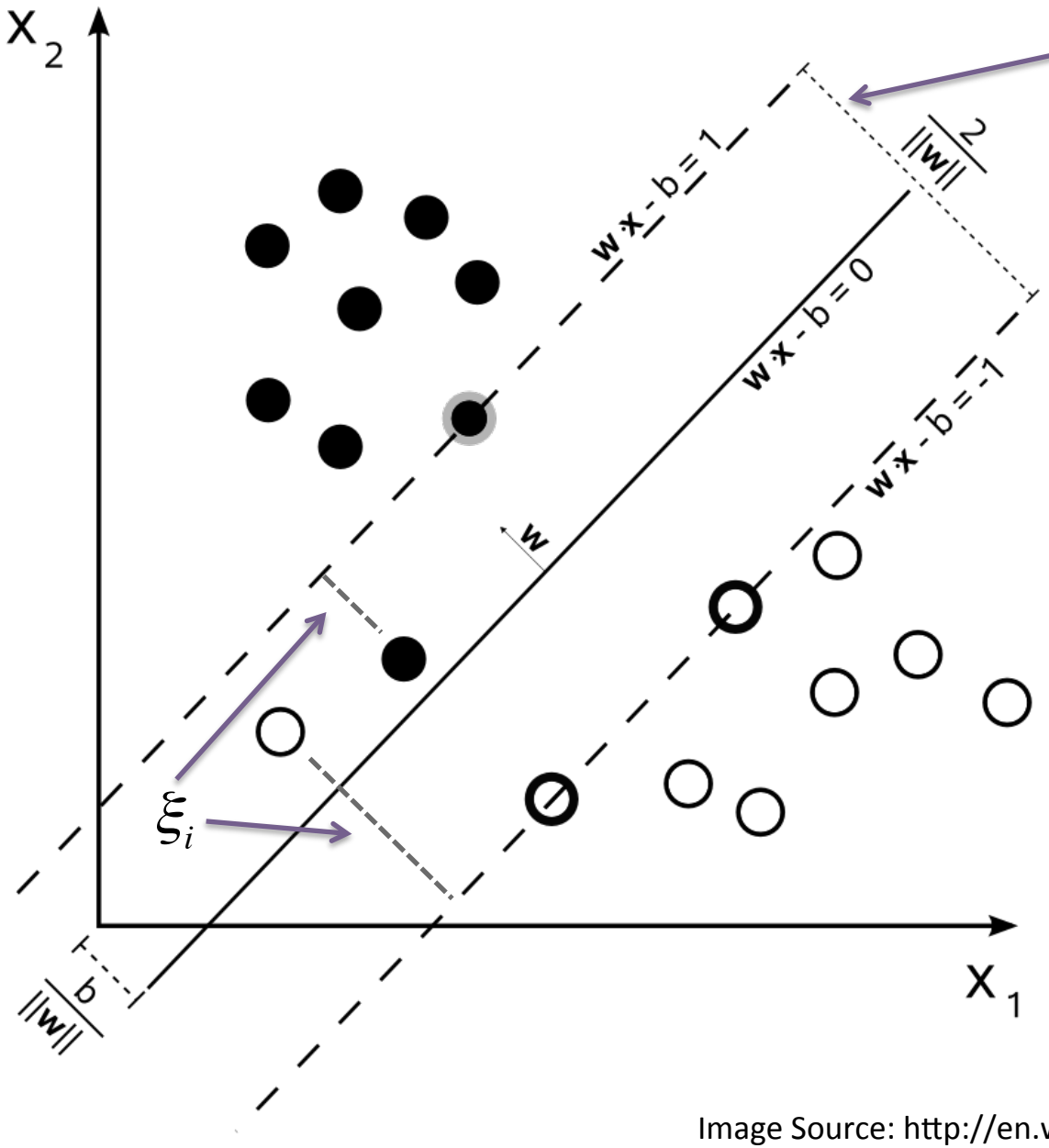
Better generalization
to unseen test examples
(beyond scope of course*)
(only training data on
margin matter)

"Linearly Separable"

*http://olivier.chapelle.cc/pub/span_lmc.pdf

Image Source: http://en.wikipedia.org/wiki/Support_vector_machine

Soft-Margin Support Vector Machine



"Margin"

$$\underset{w,b,\xi}{\operatorname{argmin}} \frac{1}{2} w^T w + \frac{C}{N} \sum_i \xi_i$$

$$\forall i : y_i \left(w^T x_i - b \right) \geq 1 - \xi_i$$

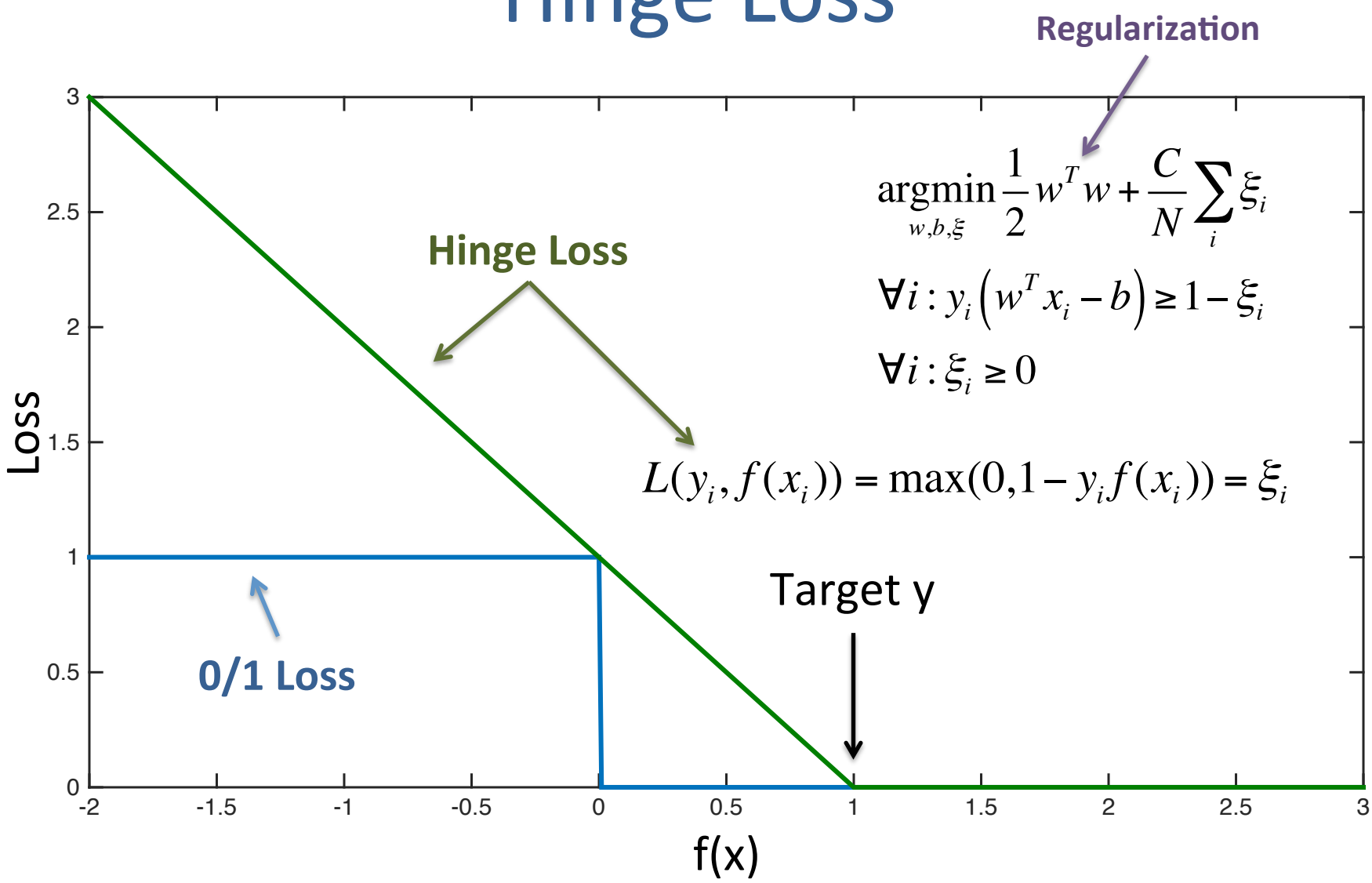
$$\forall i : \xi_i \geq 0$$

"Slack"

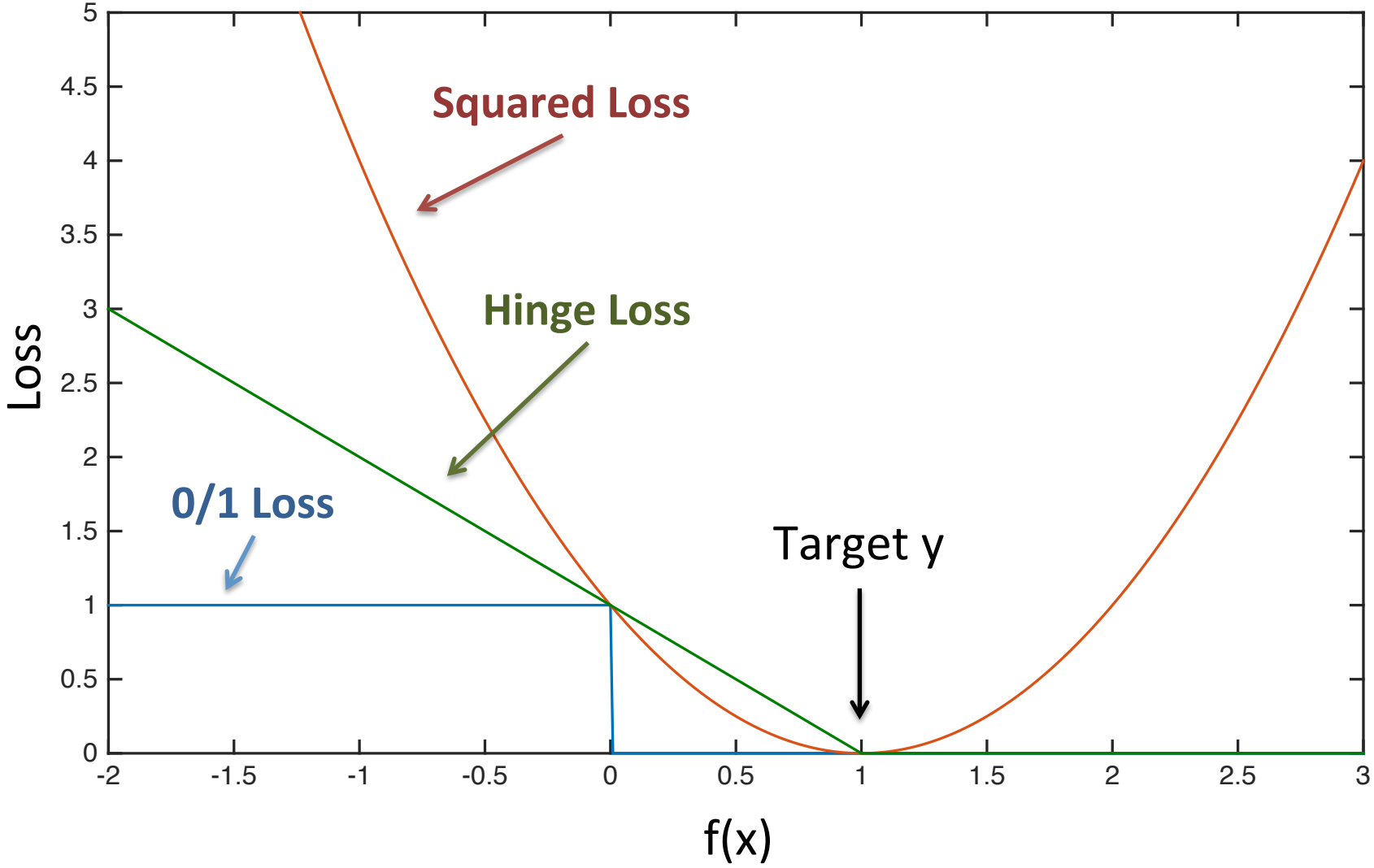
Size of Margin
vs
Size of Margin Violations
(C controls trade-off)

Image Source: http://en.wikipedia.org/wiki/Support_vector_machine

Hinge Loss


Regularization
$\underset{w,b,\xi}{\operatorname{argmin}} \frac{1}{2} w^T w + \frac{C}{N} \sum_i \xi_i$
$\forall i : y_i \left(w^T x_i - b \right) \geq 1 - \xi_i$
$\forall i : \xi_i \geq 0$
Hinge Loss
$L(y_i, f(x_i)) = \max(0, 1 - y_i f(x_i)) = \xi_i$
Target y
0/1 Loss
Loss
f(x)
3
2.5
2
1.5
1
0.5
0
-2
-1.5
-1
-0.5
0
0.5
1
1.5
2
2.5
3

Hinge Loss vs Squared Loss



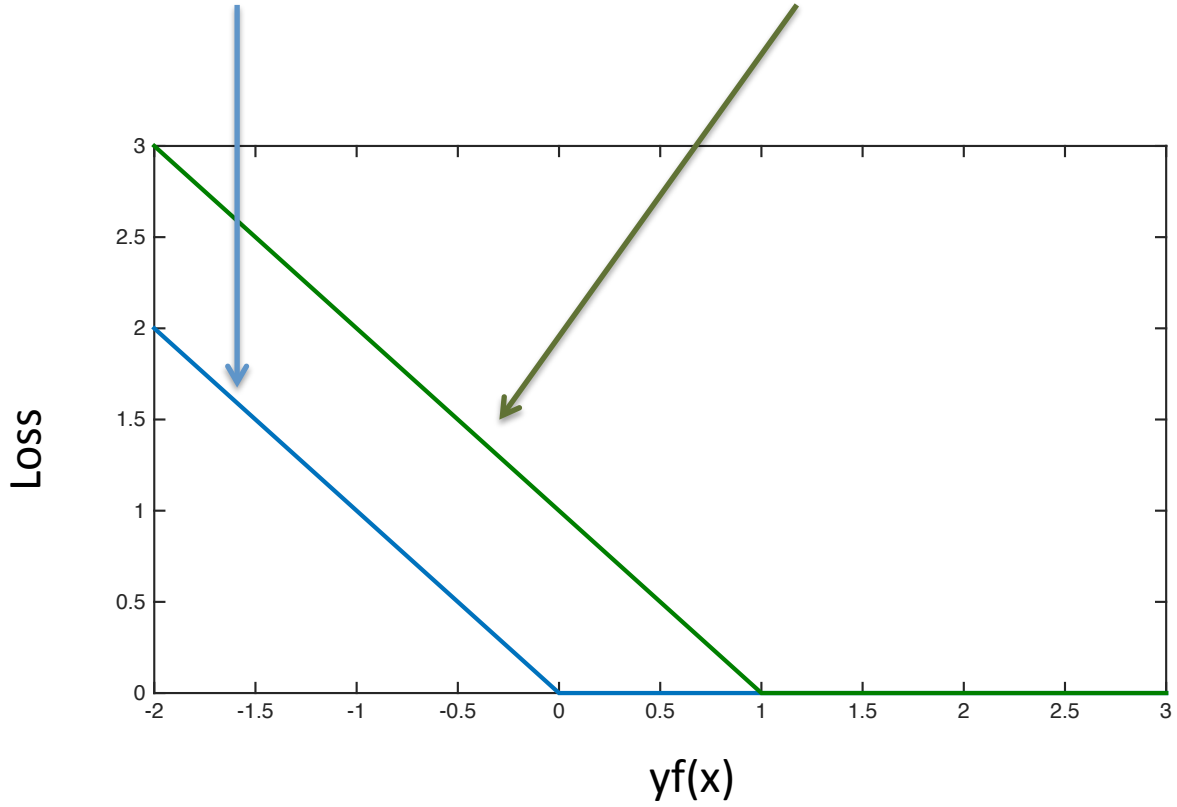
Comparison with Perceptron

Perceptron

$$\max\{0, -y_i f(x_i \mid w, b)\}$$

SVM/Hinge

$$\max\{0, 1 - y_i f(x_i \mid w, b)\}$$



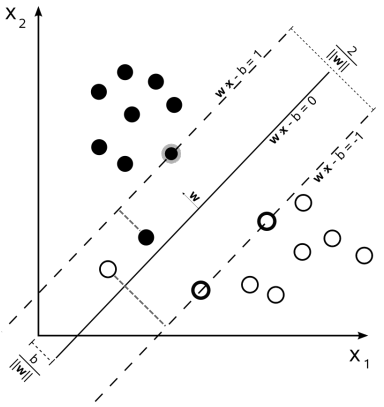
Support Vector Machine

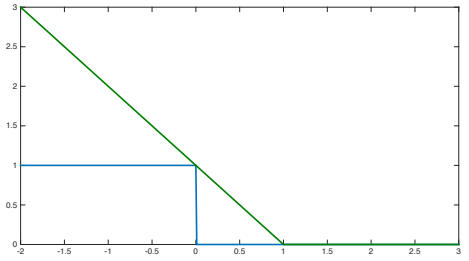
- 2 Interpretations
- Geometric
  - Margin vs Margin Violations
- Loss Minimization
  - Model complexity vs Hinge Loss
  - (Will discuss in depth next lecture)
- **Equivalent!**

$$\underset{w,b,\xi}{\operatorname{argmin}} \frac{1}{2} w^T w + \frac{C}{N} \sum_i \xi_i$$

$$\forall i : y_i \left(w^T x_i - b \right) \geq 1 - \xi_i$$

$$\forall i : \xi_i \geq 0$$





Comment on Optimization

- Hinge Loss is not smooth
  - Not differentiable

$$\underset{w,b,\xi}{\operatorname{argmin}} \frac{1}{2} w^T w + \frac{C}{N} \sum_i \xi_i$$

$$\forall i : y_i \left(w^T x_i - b \right) \geq 1 - \xi_i$$

$$\forall i : \xi_i \geq 0$$

- How to optimize?

- **Stochastic (Sub-)Gradient Descent still works!**
  - More sophisticated methods in future recitation
  - (Non-smooth convex optimization)

https://en.wikipedia.org/wiki/Subgradient_method

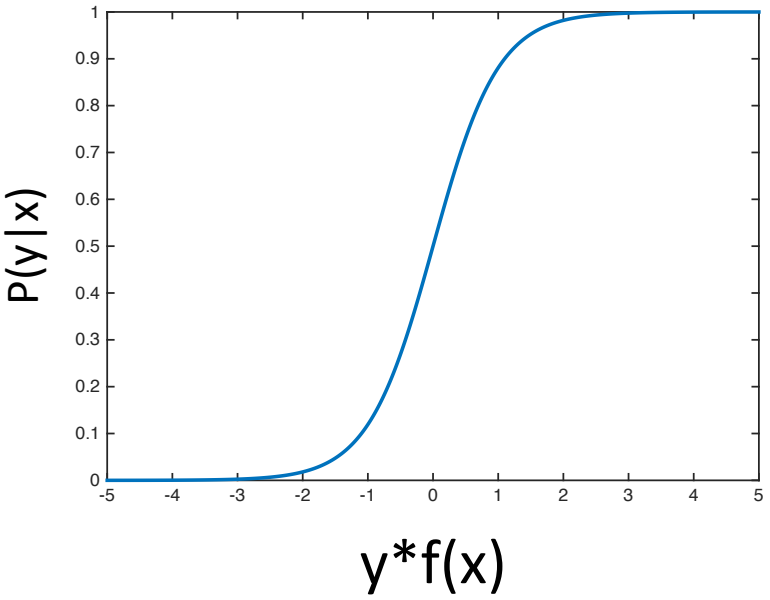
Logistic Regression

aka “Log-Linear” Models

Logistic Regression

$$P(y \mid x, w, b) = \frac{e^{\frac{1}{2}y\left(w^T x - b\right)}}{e^{\frac{1}{2}y\left(w^T x - b\right)} + e^{-\frac{1}{2}y\left(w^T x - b\right)}}$$

$$P(y \mid x, w, b) = \frac{1}{1 + e^{-y(w^T x - b)}}$$



"Log-Linear" Model

Also known as sigmoid function: $\sigma(a) = \frac{e^a}{1 + e^a}$

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

Maximum Likelihood Training

- Training set: $S = \{(x_i, y_i)\}_{i=1}^{N} \quad x \in R^D \quad y \in \{-1, +1\}$
- Maximum Likelihood: $\underset{w,b}{\operatorname{argmax}} \prod_i P(y_i \mid x_i, w, b)$
  - **(Why?)**
- Each (x,y) in S sampled independently!
  - Discussed further in Recitation Next Thursday (1/21)

Why Use Logistic Regression?

- SVMs often better at classification
  - At least if there is a margin…
- Calibrated Probabilities?
- Increase in SVM score….
  - ...similar increase in P(y=+1|x)?
  - **Not well calibrated!**
- **Logistic Regression!**

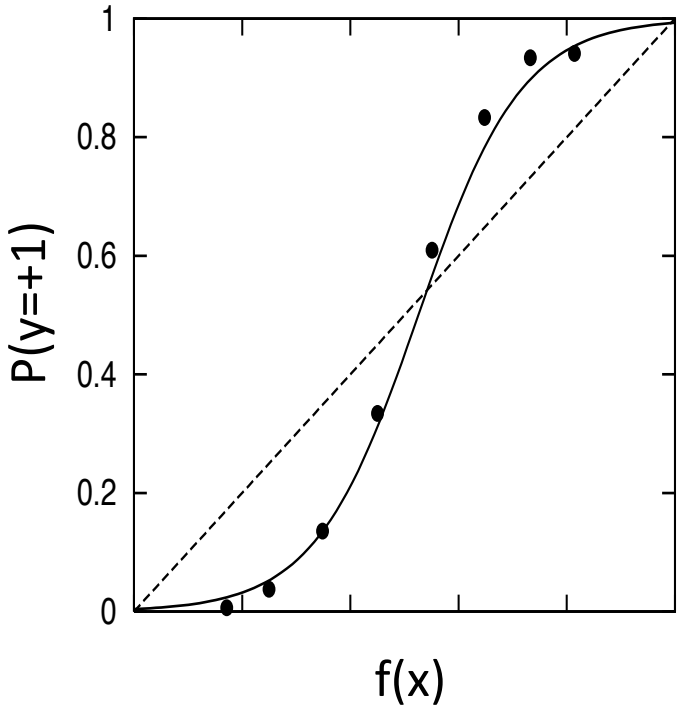


*Figure above is for Boosted Decision Trees (SVMs have similar effect)

Image Source: http://machinelearning.org/proceedings/icml2005/papers/079_GoodProbabilities_NiculescuMizilCaruana.pdf

Log Loss

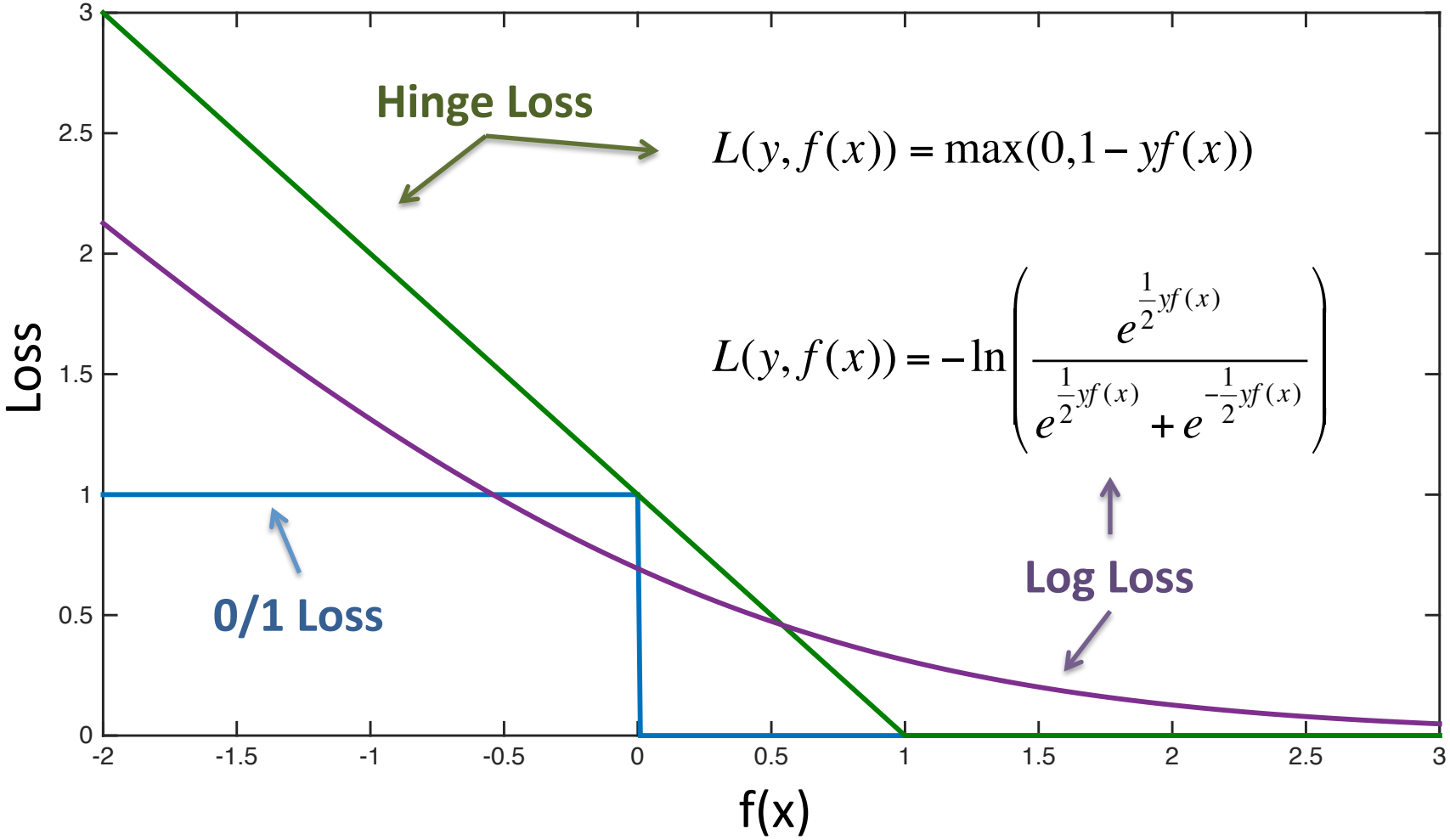
$$P(y \mid x, w, b) = \frac{e^{\frac{1}{2}y\left(w^T x - b\right)}}{e^{\frac{1}{2}y\left(w^T x - b\right)} + e^{-\frac{1}{2}y\left(w^T x - b\right)}} = \frac{e^{\frac{1}{2}yf(x|w,b)}}{e^{\frac{1}{2}yf(x|w,b)} + e^{-\frac{1}{2}yf(x|w,b)}}$$

$$\underset{w,b}{\operatorname{argmax}} \prod_i P(y_i \mid x_i, w, b) = \underset{w,b}{\operatorname{argmin}} \sum_i \underbrace{-\ln P(y_i \mid x_i, w, b)}_{\text{Log Loss}}$$

$$L(y, f(x)) = -\ln\left(\frac{e^{\frac{1}{2}yf(x)}}{e^{\frac{1}{2}yf(x)} + e^{-\frac{1}{2}yf(x)}}\right)$$

Solve using
Gradient Descent

Log Loss vs Hinge Loss



Log-Loss Gradient
(For One Example)

$$\partial_w - \ln P(y_i \mid x_i) = -\partial_w \left(\frac{1}{2} y_i f(x_i \mid w, b) - \ln \left(e^{\frac{1}{2} y_i f(x_i \mid w, b)} + e^{-\frac{1}{2} y_i f(x_i \mid w, b)} \right) \right)$$

$$= -\frac{1}{2} y_i x_i + \partial_w \ln \left(e^{\frac{1}{2} y_i f(x_i \mid w, b)} + e^{-\frac{1}{2} y_i f(x_i \mid w, b)} \right)$$

$$= -\frac{1}{2} y_i x_i + \frac{1}{e^{\frac{1}{2} y_i f(x_i \mid w, b)} + e^{-\frac{1}{2} y_i f(x_i \mid w, b)}} \partial_w \left(e^{\frac{1}{2} y_i f(x_i \mid w, b)} + e^{-\frac{1}{2} y_i f(x_i \mid w, b)} \right)$$

$$= \left(-1 + \frac{1}{e^{\frac{1}{2} y_i f(x_i \mid w, b)} + e^{-\frac{1}{2} y_i f(x_i \mid w, b)}} \left(e^{\frac{1}{2} y_i f(x_i \mid w, b)} - e^{-\frac{1}{2} y_i f(x_i \mid w, b)} \right) \right) \frac{1}{2} y_i x_i$$

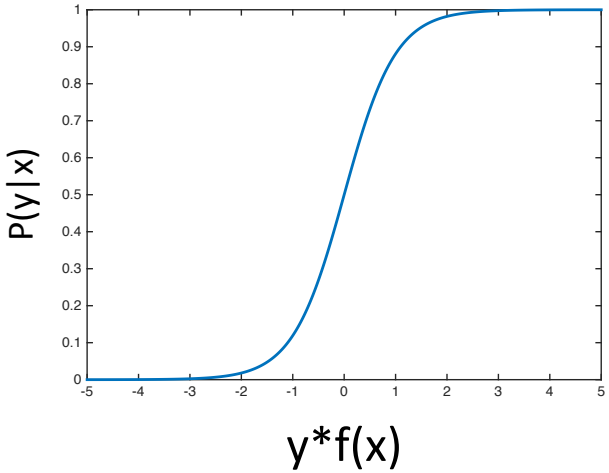
$$= \left(-1 + P(y_i \mid x_i) - P(-y_i \mid x_i) \right) \frac{1}{2} y_i x_i$$

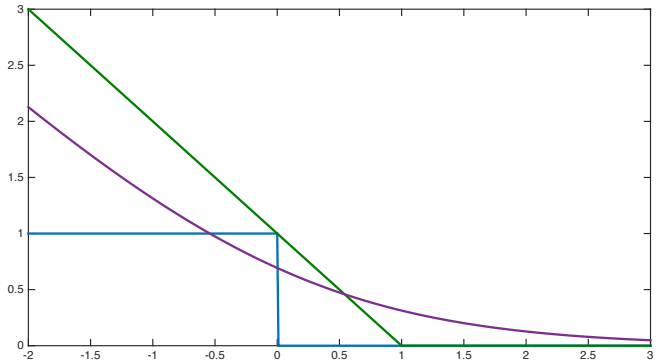
$$= -P(-y_i \mid x_i) y_i x_i = -\left(1 - P(y_i \mid x_i) \right) y_i x$$

$$P(y \mid x, w, b) = \frac{e^{\frac{1}{2} y f(x \mid w, b)}}{e^{\frac{1}{2} y f(x \mid w, b)} + e^{-\frac{1}{2} y f(x \mid w, b)}}$$

Logistic Regression

- Two Interpretations
- Maximizing Likelihood
- Minimizing Log Loss
- **Equivalent!**



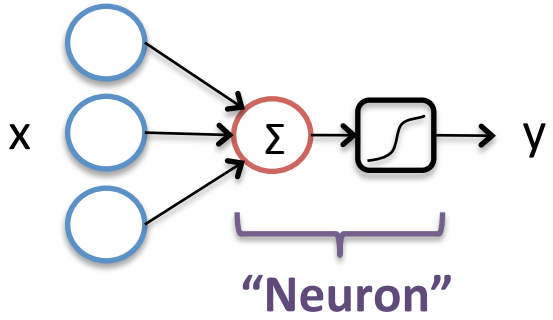


Feed-Forward Neural Networks

aka Not Quite Deep Learning

1 Layer Neural Network

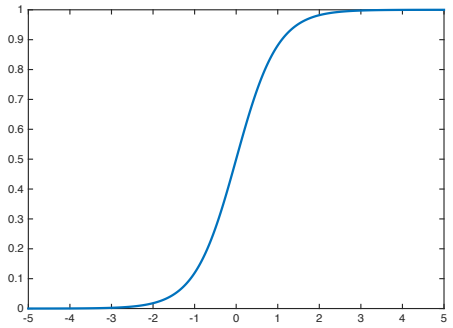
- 1 Neuron
  - Takes input x
  - Outputs y



$$f(x|w,b) = w^T x - b$$
$$= w_1 * x_1 + w_2 * x_2 + w_3 * x_3 - b$$

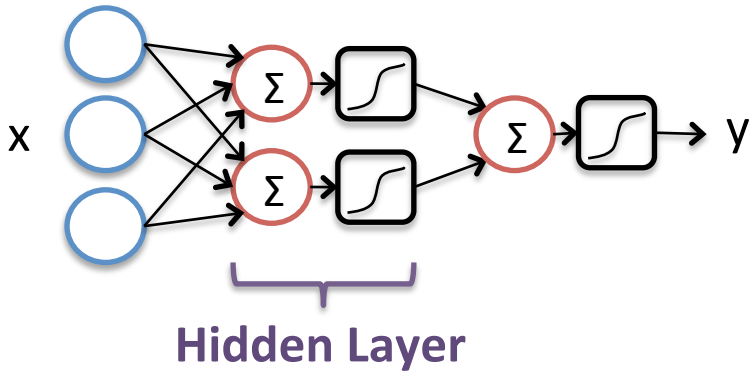
$$\longrightarrow \quad y = \sigma(f(x))$$

- **~Logistic Regression!**
  - Gradient Descent



sigmoid
tanh
rectilinear

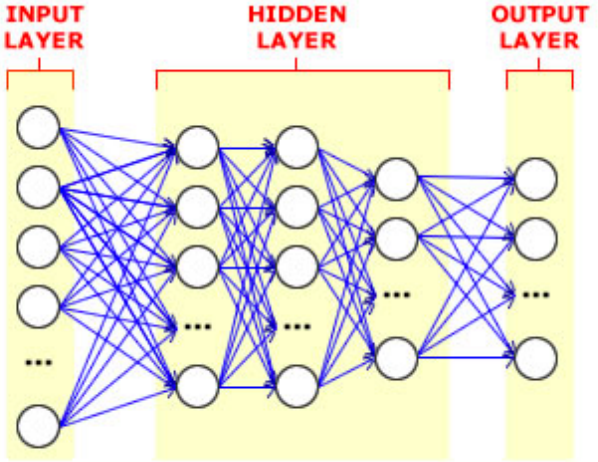
2 Layer Neural Network



- 2 Layers of Neurons
  - 1st Layer takes input x
  - 2nd Layer takes output of 1st layer
- Can approximate arbitrary functions
  - Provided hidden layer is large enough
  - "fat" 2-Layer Network

Non-Linear!

Aside: Deep Neural Networks

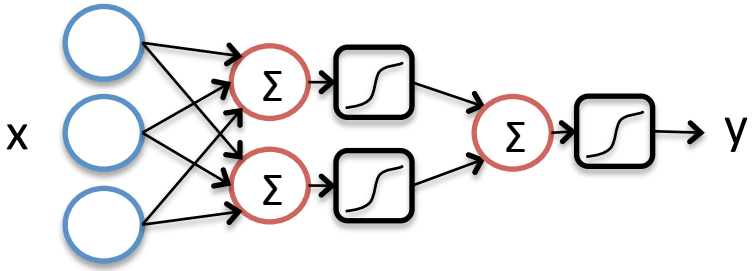


- Why prefer Deep over a "Fat" 2-Layer?
  - Compact Model
    - (exponentially large "fat" model)
  - Easier to train?

Image Source: http://blog.peltarion.com/2014/06/22/deep-learning-and-deep-neural-networks-in-synapse/

Training Neural Networks

- Gradient Descent!
  - Even for Deep Networks*
- Parameters:
  - $(w_{11},b_{11},w_{12},b_{12},w_2,b_2)$



$$f(x|w,b) = w^T x - b \quad y = \sigma(f(x))$$

$$\partial_{w_2} \sum_{i=1}^{N} L(y_i,\sigma_2) = \sum_{i=1}^{N} \partial_{w_2} L(y_i,\sigma_2) = \sum_{i=1}^{N} \partial_{\sigma_2} L(y_i,\sigma_2) \partial_{w_2} \sigma_2 = \sum_{i=1}^{N} \partial_{\sigma_2} L(y_i,\sigma_2) \partial_{f_2} \sigma_2 \partial_{w_2} f_2$$

$$\partial_{w_{1m}} \sum_{i=1}^{N} L(y_i,\sigma_2) = \sum_{i=1}^{N} \partial_{\sigma_2} L(y_i,\sigma_2) \partial_{f_2} \sigma_2 \partial_{w_1} f_2 = \sum_{i=1}^{N} \partial_{\sigma_2} L(y_i,\sigma_2) \partial_{f_2} \sigma_2 \partial_{\sigma_{1m}} f_2 \partial_{f_{1m}} \sigma_{1m} \partial_{w_{1m}} f_{1m}$$

*more complicated

Backpropagation = Gradient Descent (lots of chain rules)

Today

- Beyond Basic Linear Models
  - Support Vector Machines
  - Logistic Regression
  - Feed-forward Neural Networks
  - Different ways to interpret models
- **Different Evaluation Metrics**

Evaluation

- 0/1 Loss (Classification)
- Squared Loss (Regression)
- Anything Else?

Example: Cancer Prediction

		Patient	
Model	**Loss Function**	Has Cancer	Doesn't Have Cancer
	Predicts Cancer	Low	Medium
	Predicts No Cancer	**OMG Panic!**	Low

- Value Positives & Negatives Differently
  - Care much more about positives
- "Cost Matrix"
  - 0/1 Loss is Special Case

Optimizing for Cost-Sensitive Loss

- There is no universally accepted way.

Simplest Approach (Cost Balancing):

$$\underset{w,b}{\operatorname{argmin}}\left(1000\sum_{i:y_i=1}L\left(y_i,f(x_i\mid w,b)\right)+\sum_{i:y_i=-1}L\left(y_i,f(x_i\mid w,b)\right)\right)$$

Loss Function	Has Cancer	Doesn't Have Cancer
Predicts Cancer	0	1
Predicts No Cancer	1000	0

Precision & Recall

- **Precision** = TP/(TP + FP)
- **Recall** = TP/(TP + FN)

F1 = 2/(1/P+ 1/R)

Care More About Positives!

		Patient	
Model	**Counts**	Has Cancer	Doesn't Have Cancer
	Predicts Cancer	20	30
	Predicts No Cancer	5	70

- TP = True Positive, TN = True Negative
- FP = False Positive, FN = False Negative

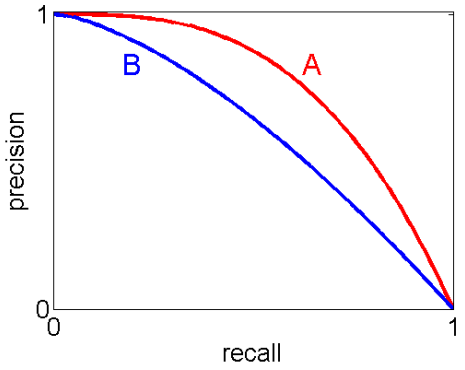


Image Source: http://pmtk3.googlecode.com/svn-history/r785/trunk/docs/demos/Decision_theory/PRhand.html

Example: Search Query

- Rank webpages by relevance

Ranking Measures

- Predict a Ranking (of webpages)
  - Users only look at top 4
  - Sort by f(x|w,b)
- Precision @4 =1/2
  - Fraction of top 4 relevant
- Recall @4 =2/3
  - Fraction of relevant in top 4
- Top of Ranking Only!

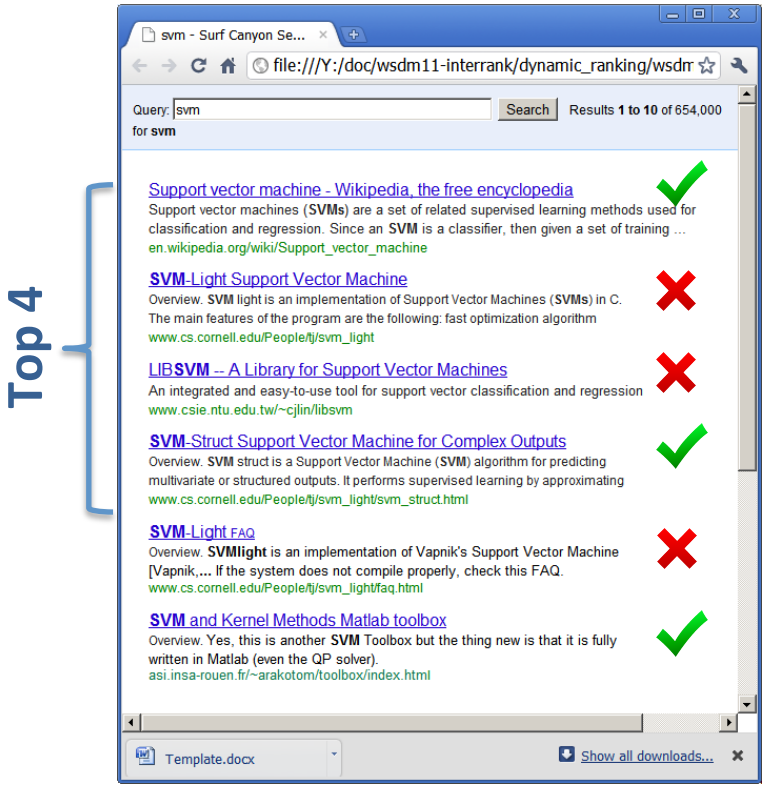
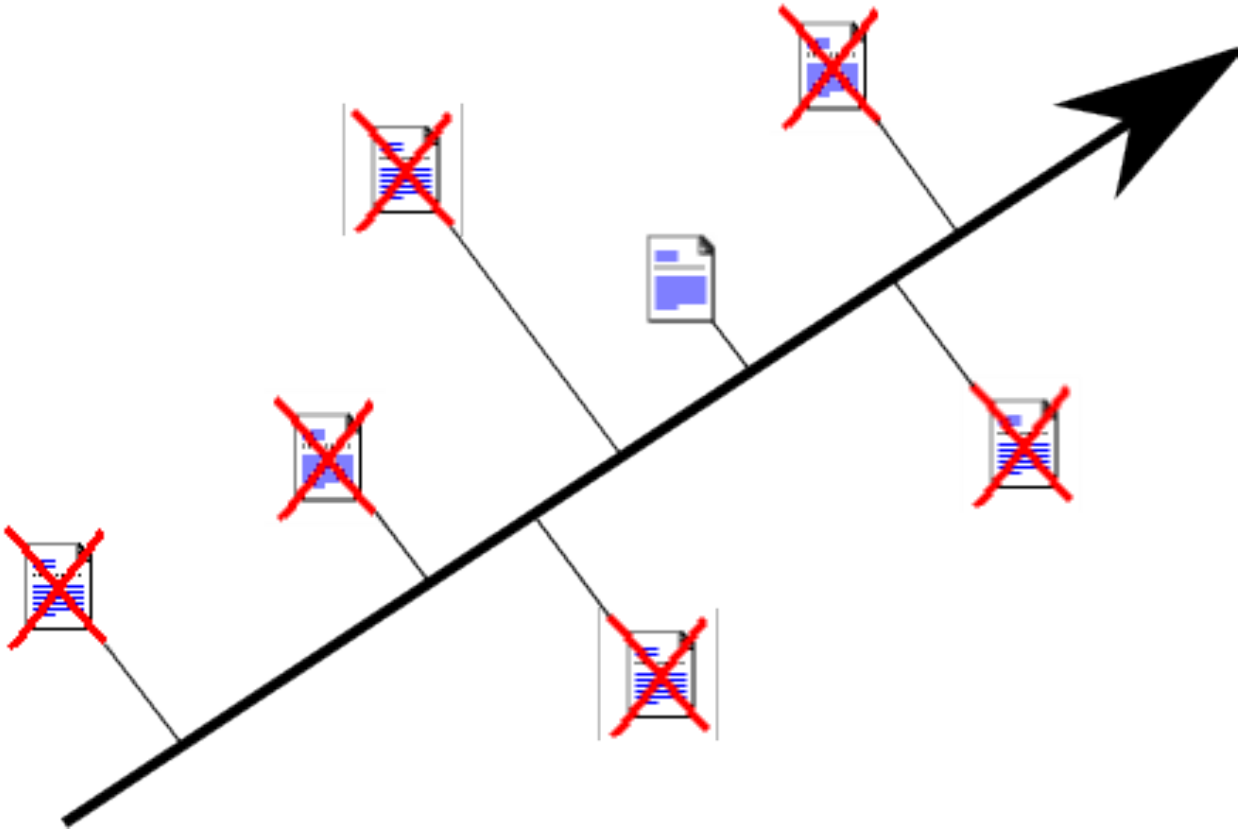


Image Source: http://pmtk3.googlecode.com/svn-history/r785/trunk/docs/demos/Decision_theory/PRhand.html

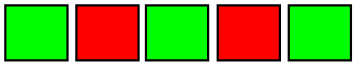
Pairwise Preferences



2 Pairwise Disagreements
4 Pairwise Agreements

ROC-Area

- ROC-Area
  - Area under ROC Curve
  - Fraction pairwise agreements

- Example: 

ROC-Area: 0.5
#Pairwise Preferences = 6
#Agreements = 3

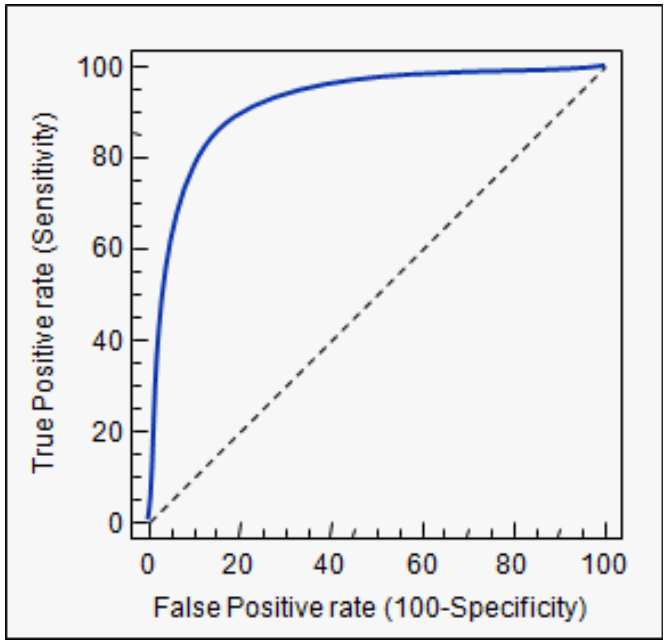
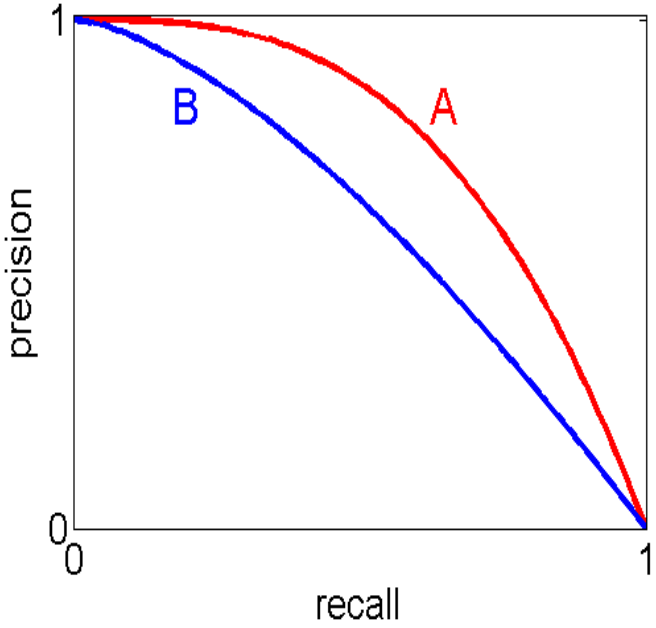


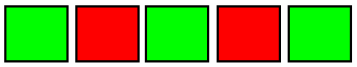
Image Source: http://www.medcalc.org/manual/roc-curves.php

Average Precision

- Average Precision
  - Area under P-R Curve
  - P@K for each positive



- Example:



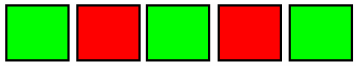
AP: $\frac{1}{3} \cdot \left(\frac{1}{1} + \frac{2}{3} + \frac{3}{5} \right) \approx 0.76$

Precision at Rank Location of Each Positive Example

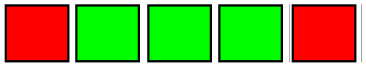
Image Source: http://pmtk3.googlecode.com/svn-history/r785/trunk/docs/demos/Decision_theory/PRhand.html

ROC-Area versus Average Precision

- ROC-Area Cares about every pairwise preference equally
- Average Precision cares more about top of ranking



ROC-Area: 0.5 Average Precision: 0.76



ROC-Area: 0.5 Average Precision: 0.64

Summary: Evaluation Measures

- Different Evaluations Measures
  - Different Scenarios
- Large focus on getting positives
  - Large cost of mis-predicting cancer
  - Relevant webpages are rare

Next Lecture

- Regularization
- Lasso
- **Thursday:**
  - **Recitation on Matrix Linear Algebra**