

Machine Learning & Data Mining

CS/CNS/EE 155

Lecture 3:
SVM, Logistic Regression, Neural Nets,
Evaluation Metrics

Announcements

- HW1 Due Tomorrow
 - Will be graded in about a week
- HW2 Released Tomorrow
 - Due Jan 22rd at 9pm
- Recitation Thursday
 - Linear Algebra (& Vector Calculus)
 - 7:30pm in Annenberg 105

Recap: Basic Recipe

- Training Data:

$$S = \{(x_i, y_i)\}_{i=1}^N \quad \begin{aligned} x &\in R^D \\ y &\in \{-1, +1\} \end{aligned}$$

- Model Class:

$$f(x | w, b) = w^T x - b \quad \text{Linear Models}$$

- Loss Function:

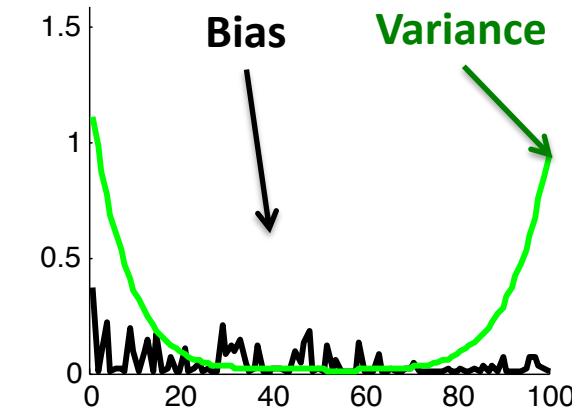
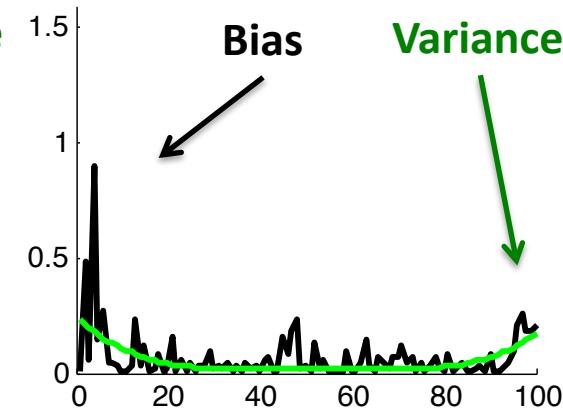
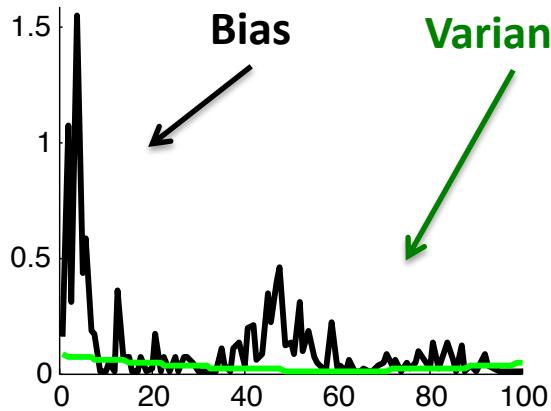
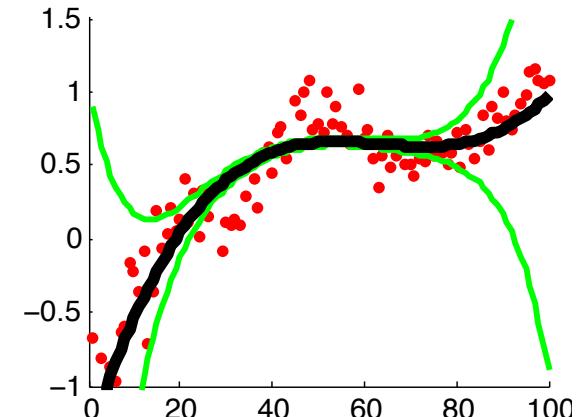
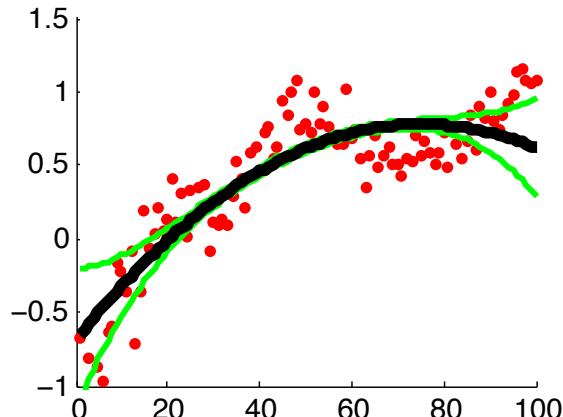
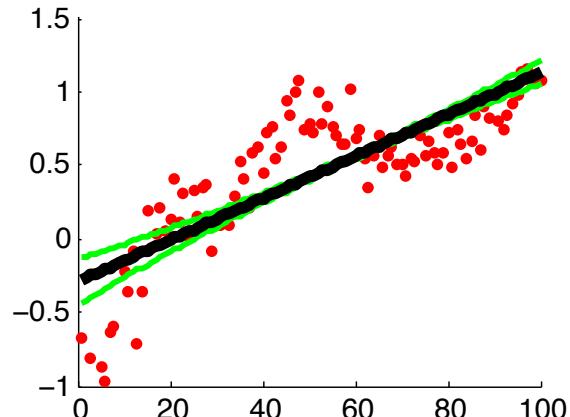
$$L(a, b) = (a - b)^2 \quad \text{Squared Loss}$$

- Learning Objective:

$$\operatorname{argmin}_{w, b} \sum_{i=1}^N L(y_i, f(x_i | w, b))$$

Optimization Problem

Recap: Bias-Variance Trade-off



Recap: Complete Pipeline

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

Training Data

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

Model Class(es)

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

Loss Function



$$\operatorname{argmin}_{w,b} \sum_{i=1}^N L(y_i, f(x_i | w, b)) \quad \text{SGD!}$$

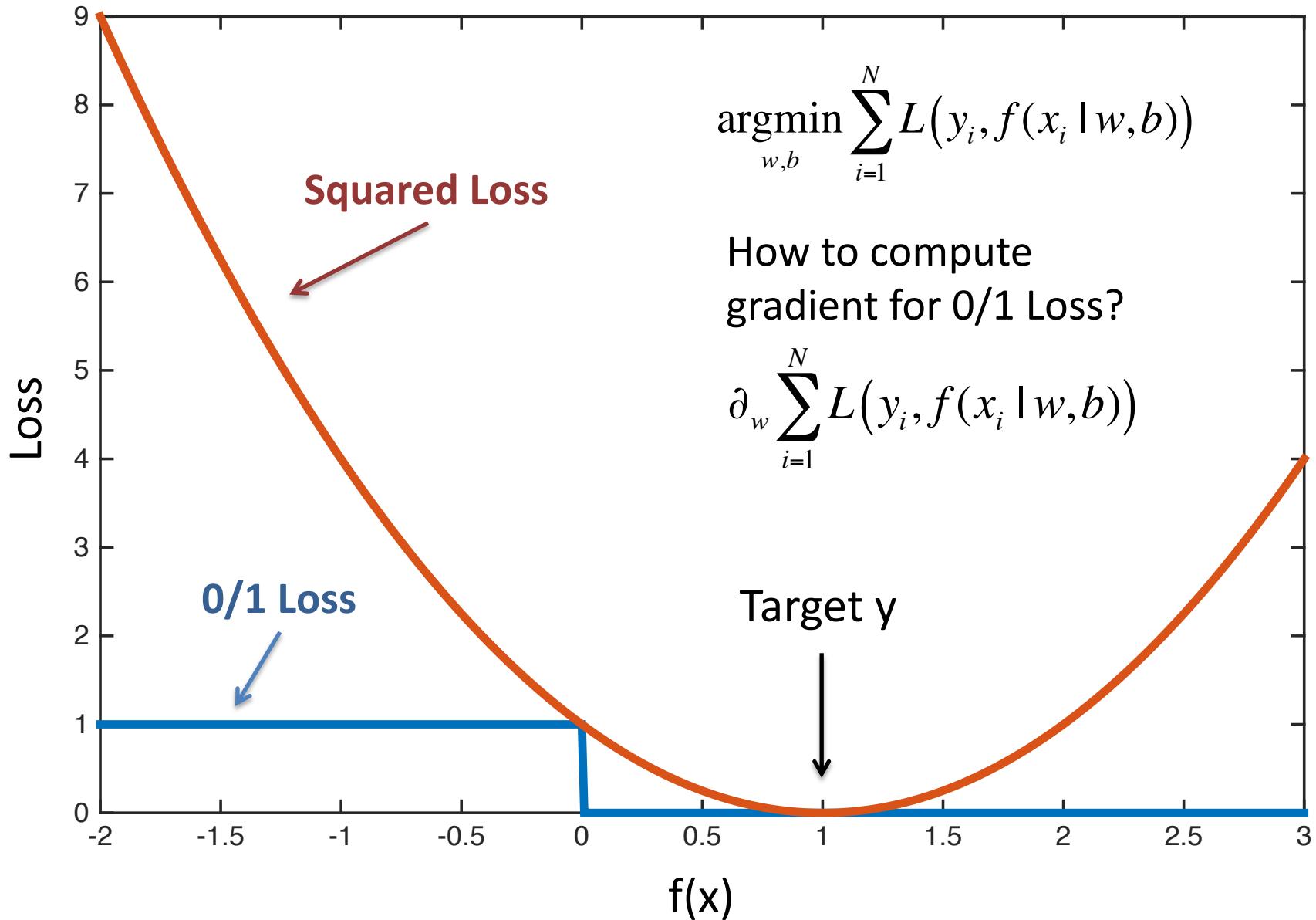
Cross Validation & Model Selection



Profit!

Today

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



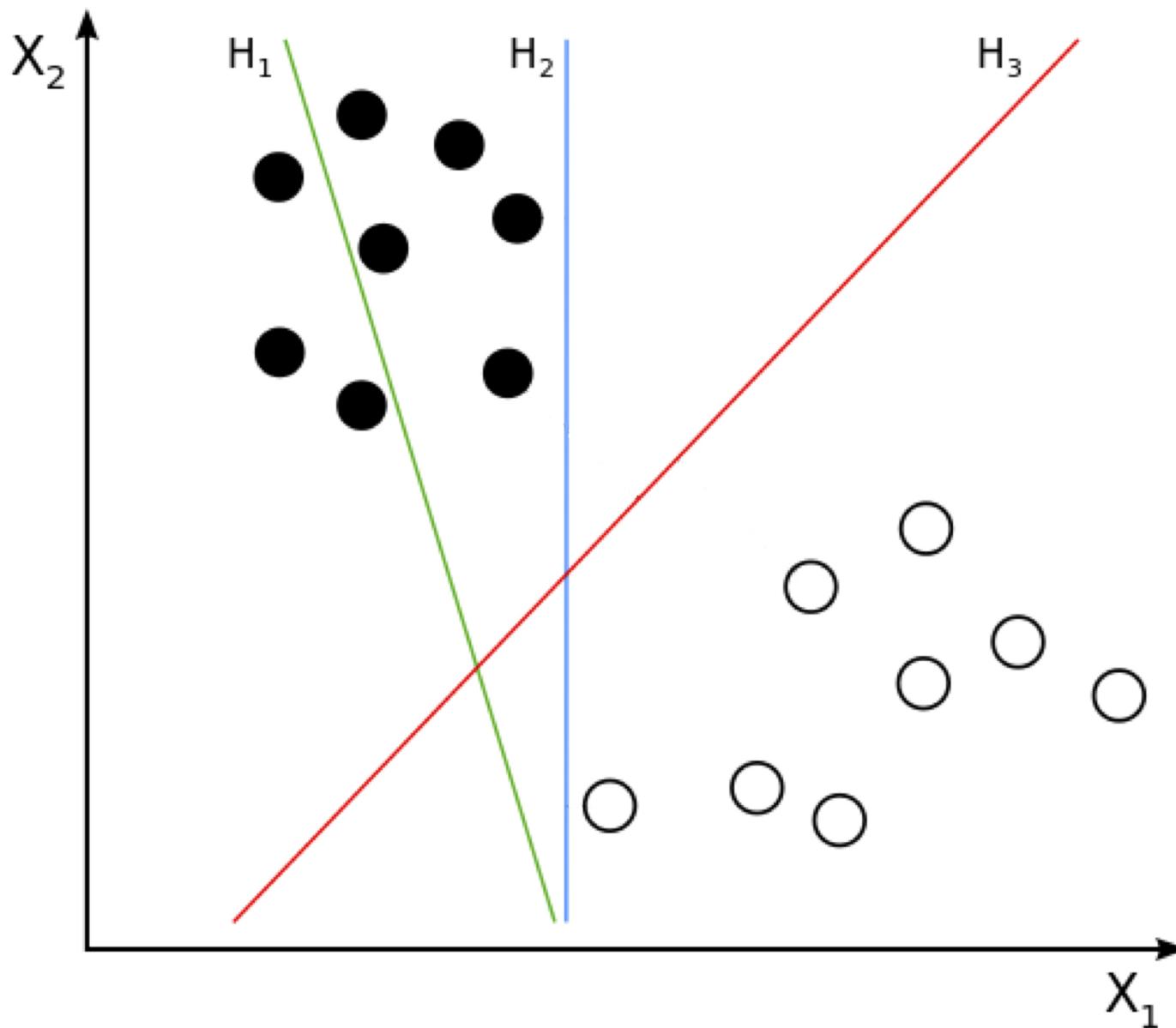
Recap: 0/1 Loss is Intractable

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

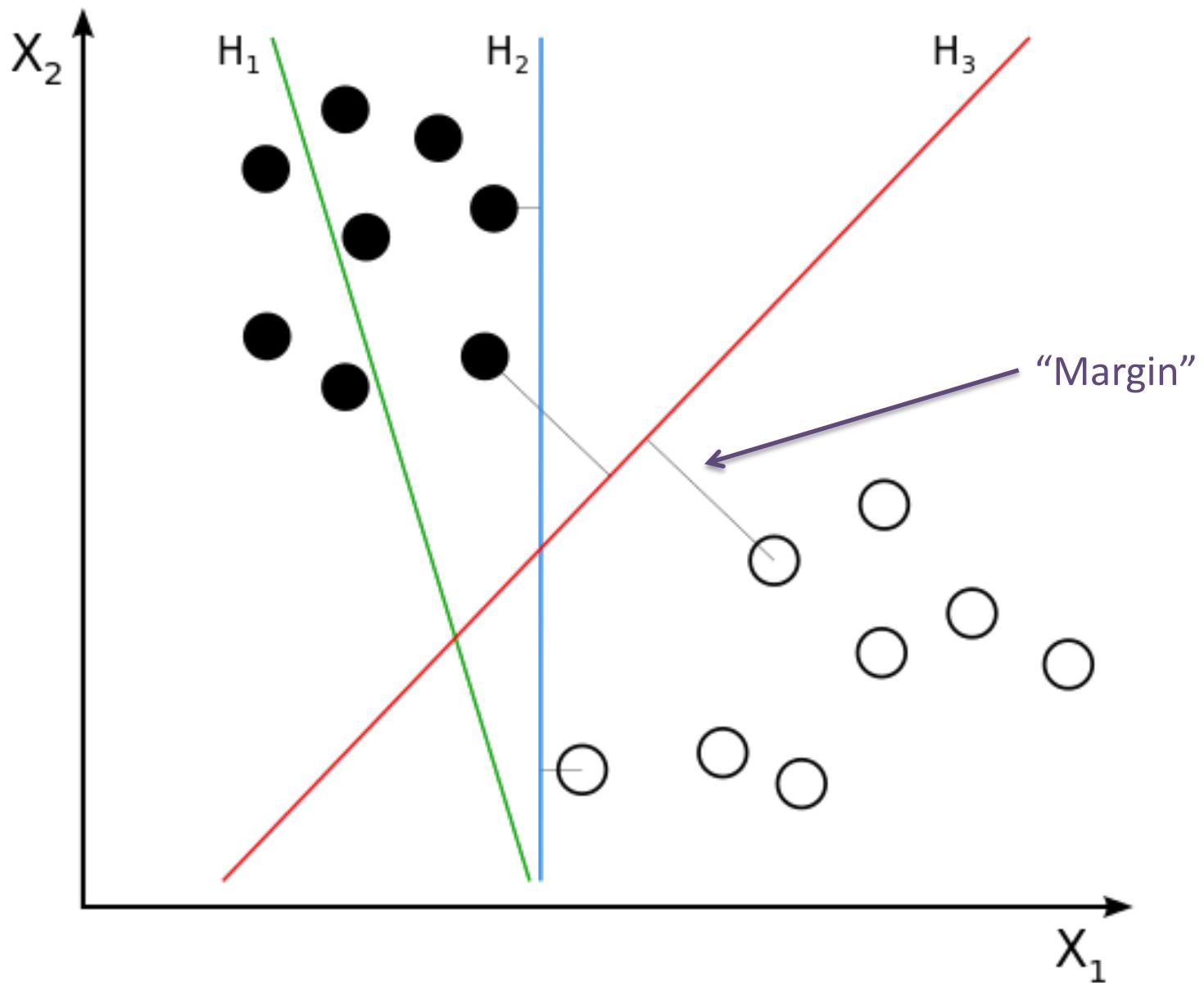
Support Vector Machines

aka Max-Margin Classifiers

Which Line is the Best Classifier?



Which Line is the Best Classifier?



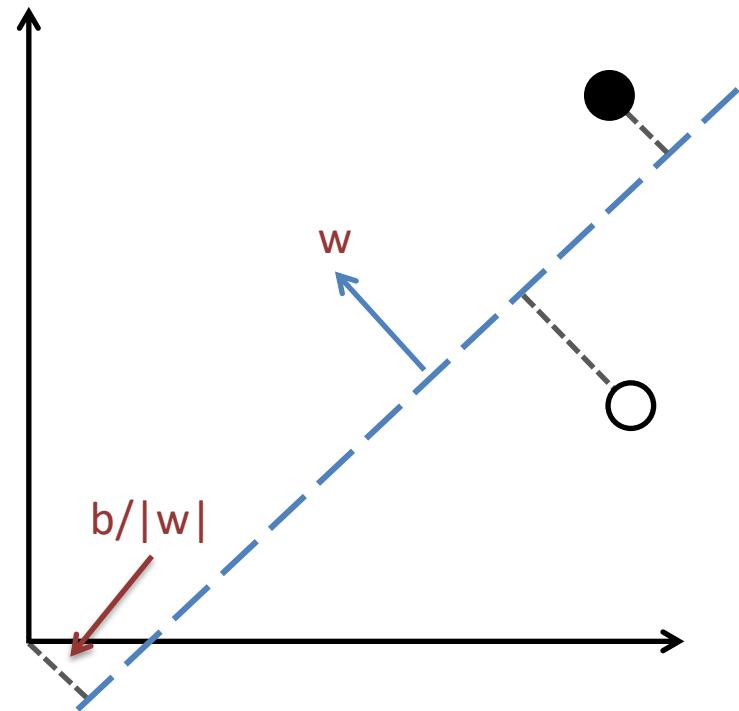
Recall: Hyperplane Distance

- Line is a 1D, Plane is 2D
- Hyperplane is many D
 - Includes Line and Plane
- Defined by (w, b)

- Distance:

$$\frac{|w^T x - b|}{\|w\|}$$

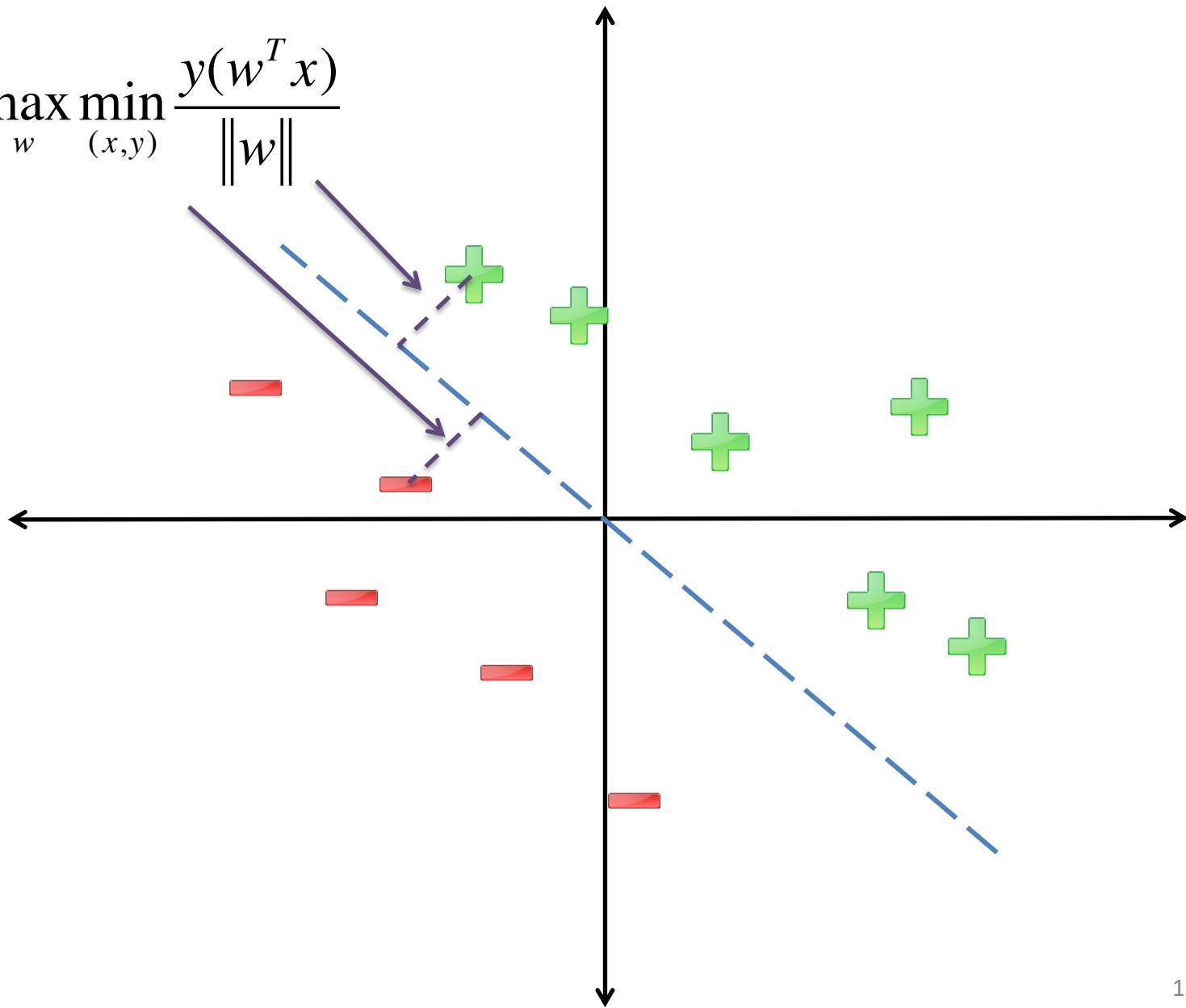
- Signed Distance: $\frac{w^T x - b}{\|w\|}$



Linear Model = un-normalized signed distance!

Recall: Margin

$$\gamma = \max_w \min_{(x,y)} \frac{y(w^T x)}{\|w\|}$$

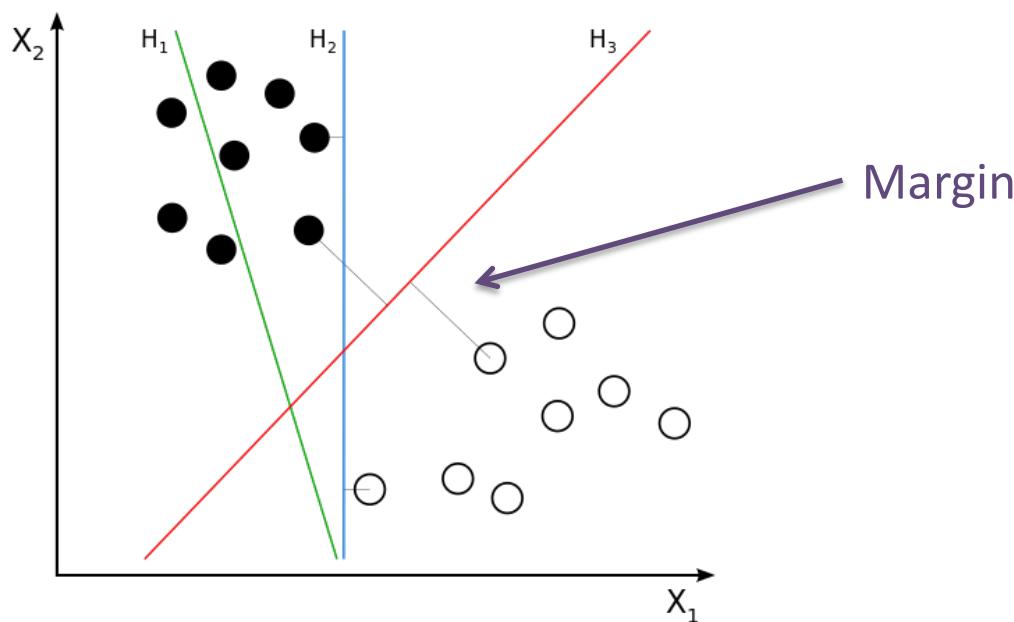


How to Maximize Margin?

(Assume Linearly Separable)

Choose w that maximizes:

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

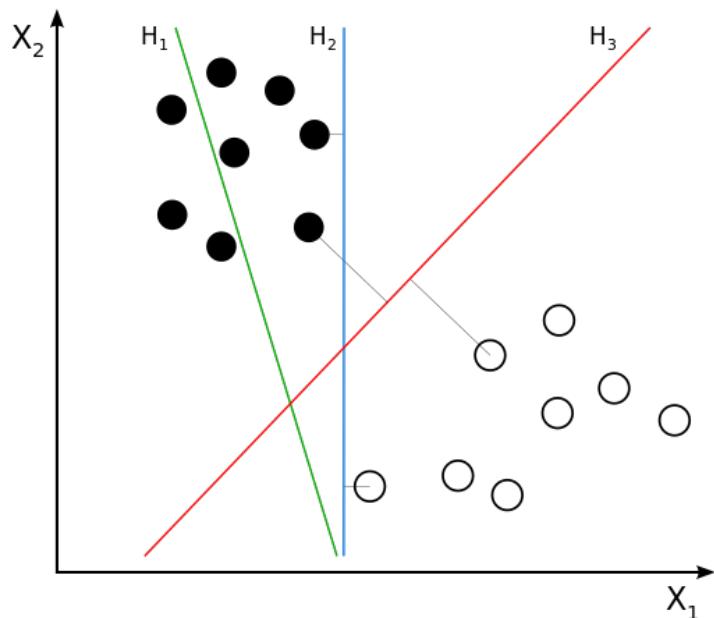


How to Maximize Margin?

(Assume Linearly Separable)

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

Hold Denominator Fixed



Suppose we instead enforce:

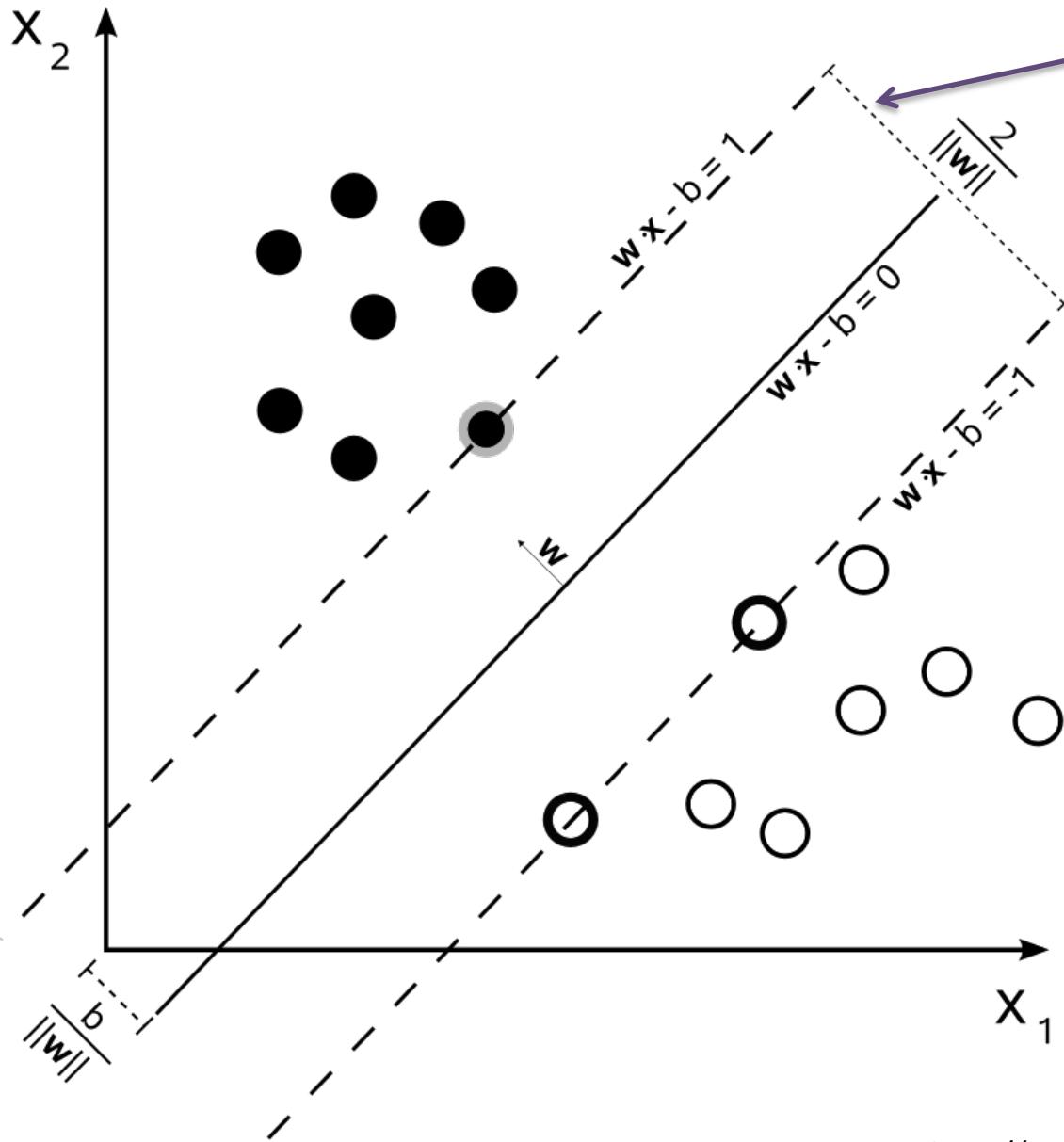
$$\min_{(x,y)} y(w^T x - b) = 1$$

Hold Numerator Fixed

Then:

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

Max Margin Classifier (Support Vector Machine)



“Margin”

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

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

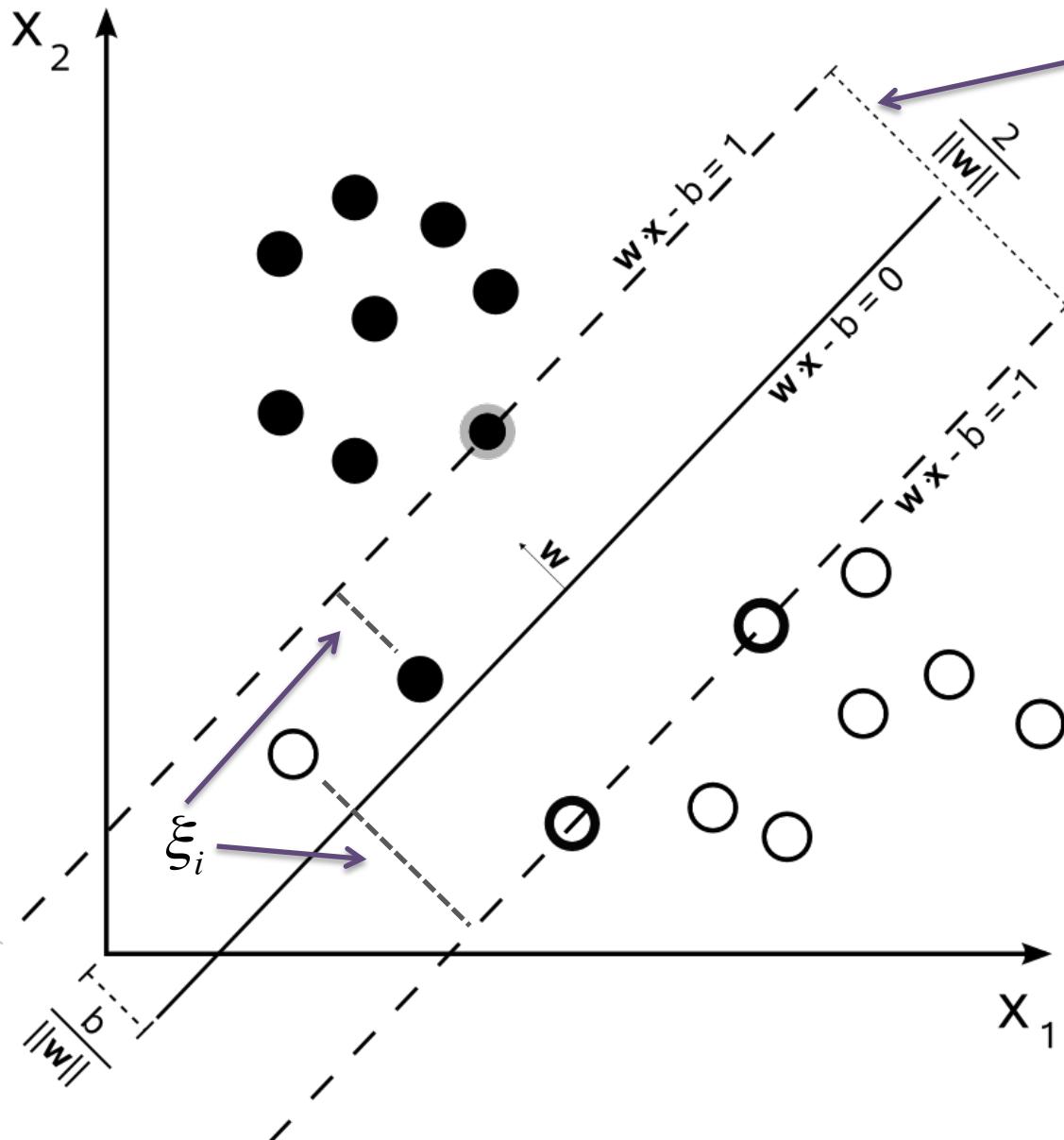
Better generalization
to unseen test examples
(beyond scope of course*)
(only training data on
margin matter)

“Linearly Separable”

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

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

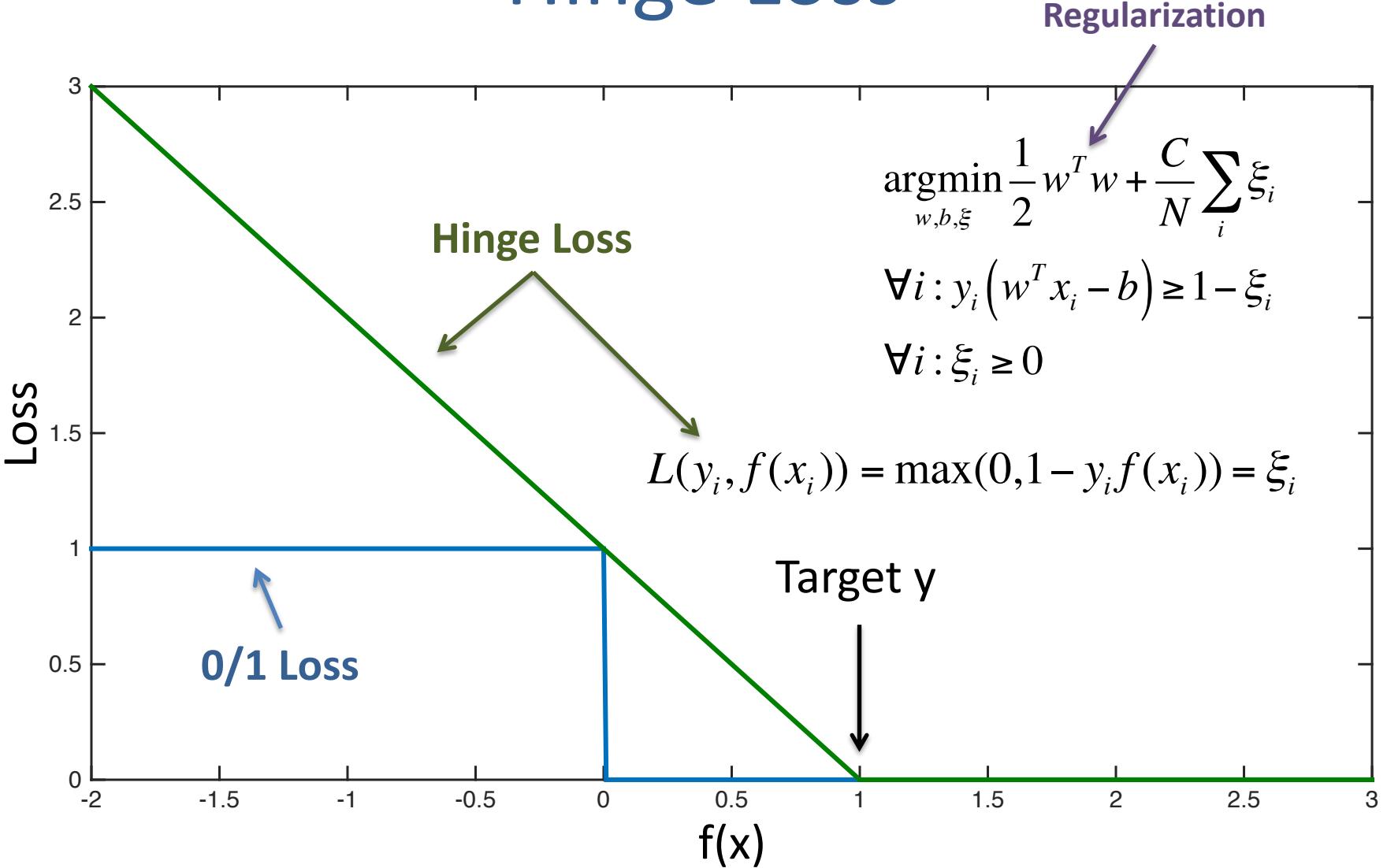
Soft-Margin Support Vector Machine



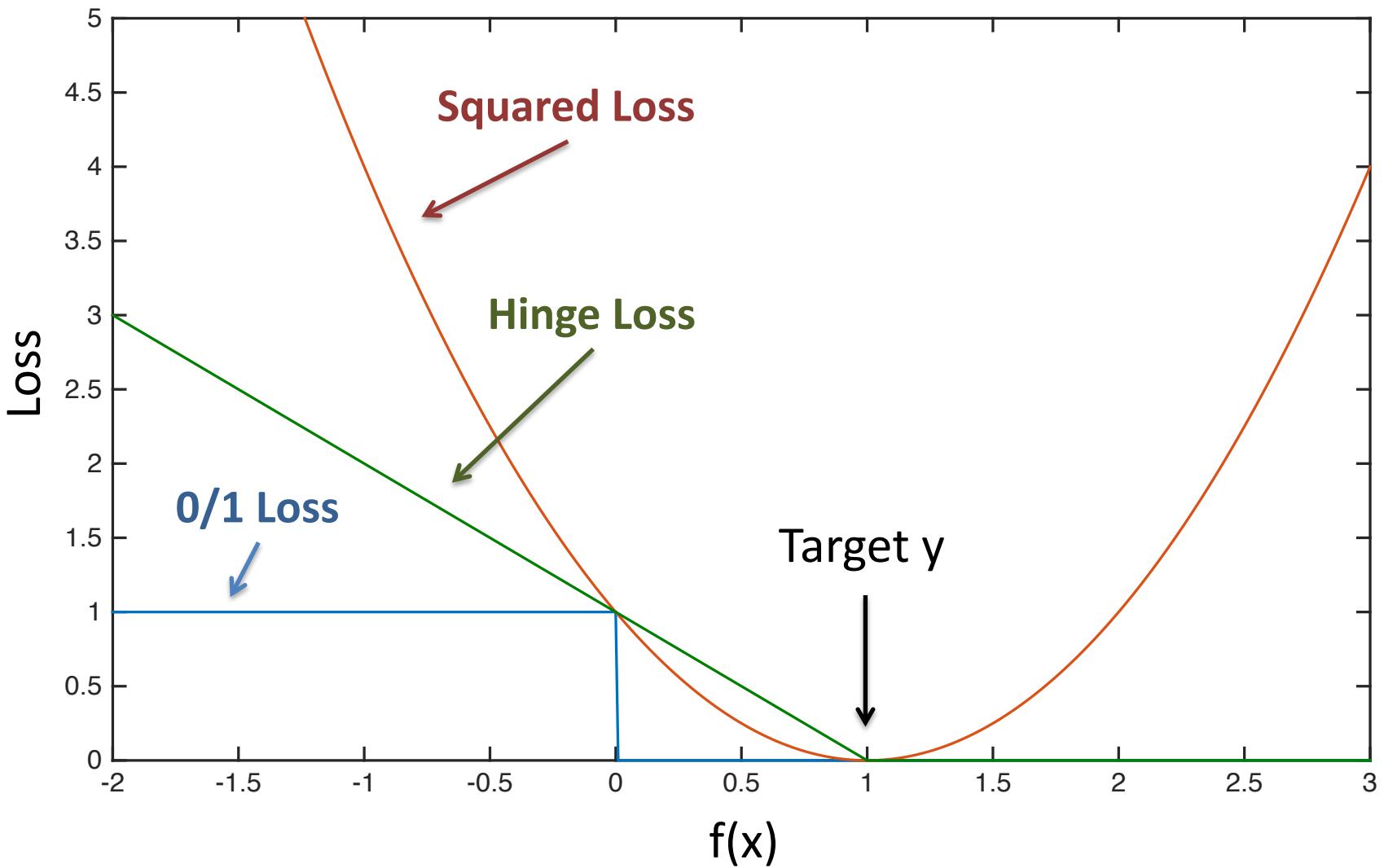
$$\underset{w,b,\xi}{\operatorname{argmin}} \frac{1}{2} w^T w + \frac{C}{N} \sum_i \xi_i$$
$$\forall i : y_i (w^T x_i - b) \geq 1 - \xi_i$$
$$\forall i : \xi_i \geq 0$$

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

Hinge Loss



Hinge Loss vs Squared Loss



Recall: Perceptron Learning Algorithm (Linear Classification Model)

- $w^1 = 0, b^1 = 0$

$$f(x | w) = \text{sign}(w^T x - b)$$

- For $t = 1 \dots$

- Receive example (x, y)

- If $f(x | w^t, b^t) = y$

- $[w^{t+1}, b^{t+1}] = [w^t, b^t]$

- Else

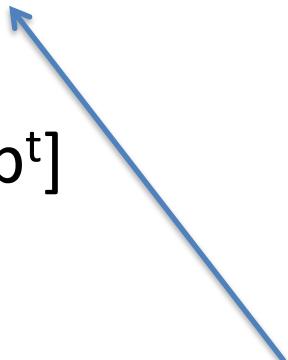
- $w^{t+1} = w^t + yx$

- $b^{t+1} = b^t - y$

Training Set:

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

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

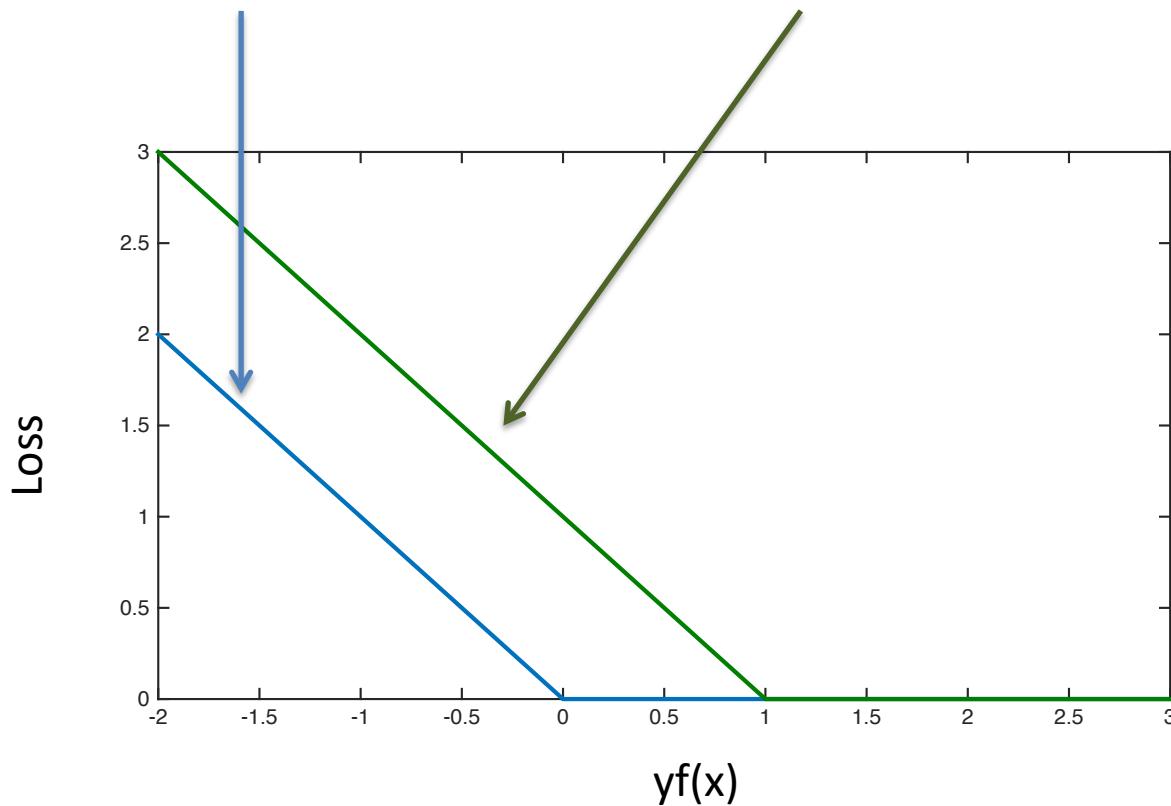


Go through training set
in arbitrary order
(e.g., randomly)

Comparison with Perceptron “Loss”

Perceptron

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



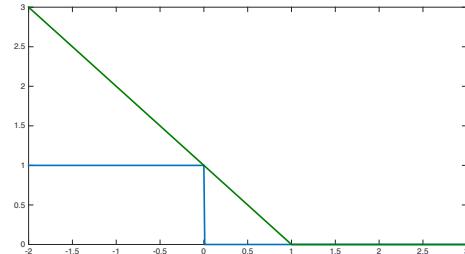
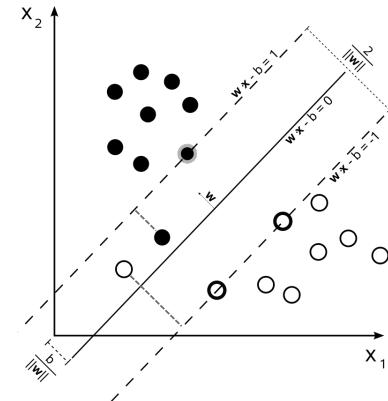
SVM/Hinge

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

Support Vector Machine

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

$$\underset{w,b,\xi}{\operatorname{argmin}} \frac{1}{2} w^T w + \frac{C}{N} \sum_i \xi_i$$
$$\forall i : y_i (w^T x_i - b) \geq 1 - \xi_i$$
$$\forall i : \xi_i \geq 0$$



Comment on Optimization

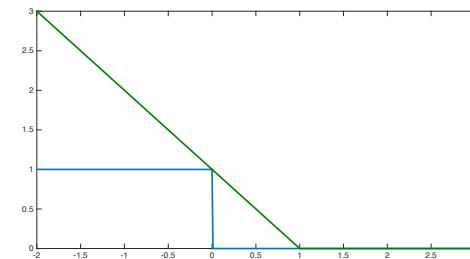
- Hinge Loss is not smooth
 - Not differentiable

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

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

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

- How to optimize?



- **Stochastic (Sub-)Gradient Descent still works!**
 - Sub-gradients discussed next lecture

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

Logistic Regression

aka “Log-Linear” Models

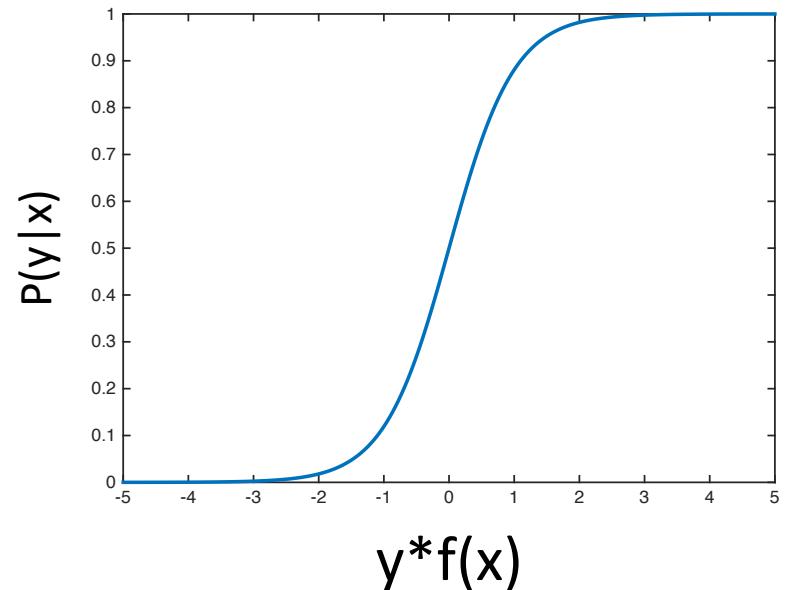
Logistic Regression

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

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

“Log-Linear” Model

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



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

Maximum Likelihood Training

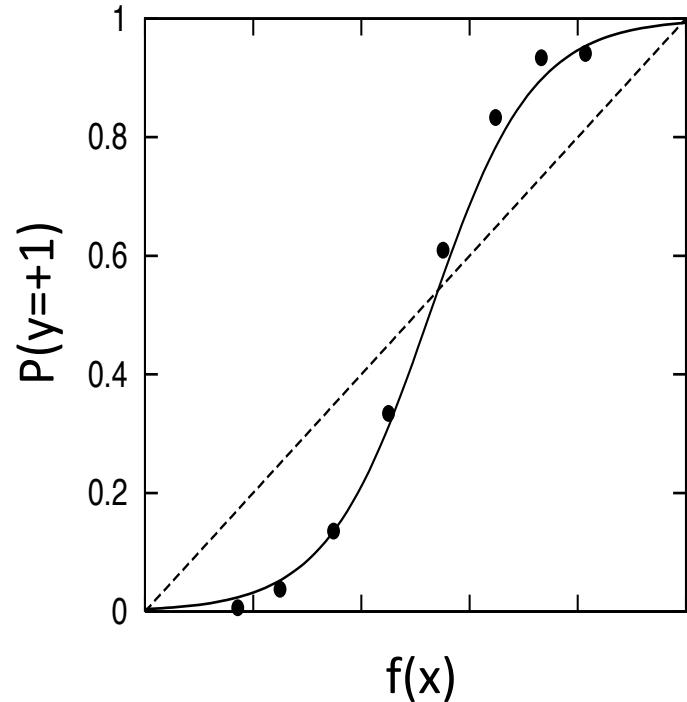
- Training set:

$$S = \{(x_i, y_i)\}_{i=1}^N \quad \begin{matrix} x \in R^D \\ y \in \{-1, +1\} \end{matrix}$$

- Maximum Likelihood: $\underset{w,b}{\operatorname{argmax}} \prod_i P(y_i | x_i, w, b)$
 - **(Why?)**
- Each (x,y) in S sampled independently!
 - Discussed further in Probably Recitation

Why Use Logistic Regression?

- SVMs often better at classification
 - Assuming margin exists...
- 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)

Log Loss

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

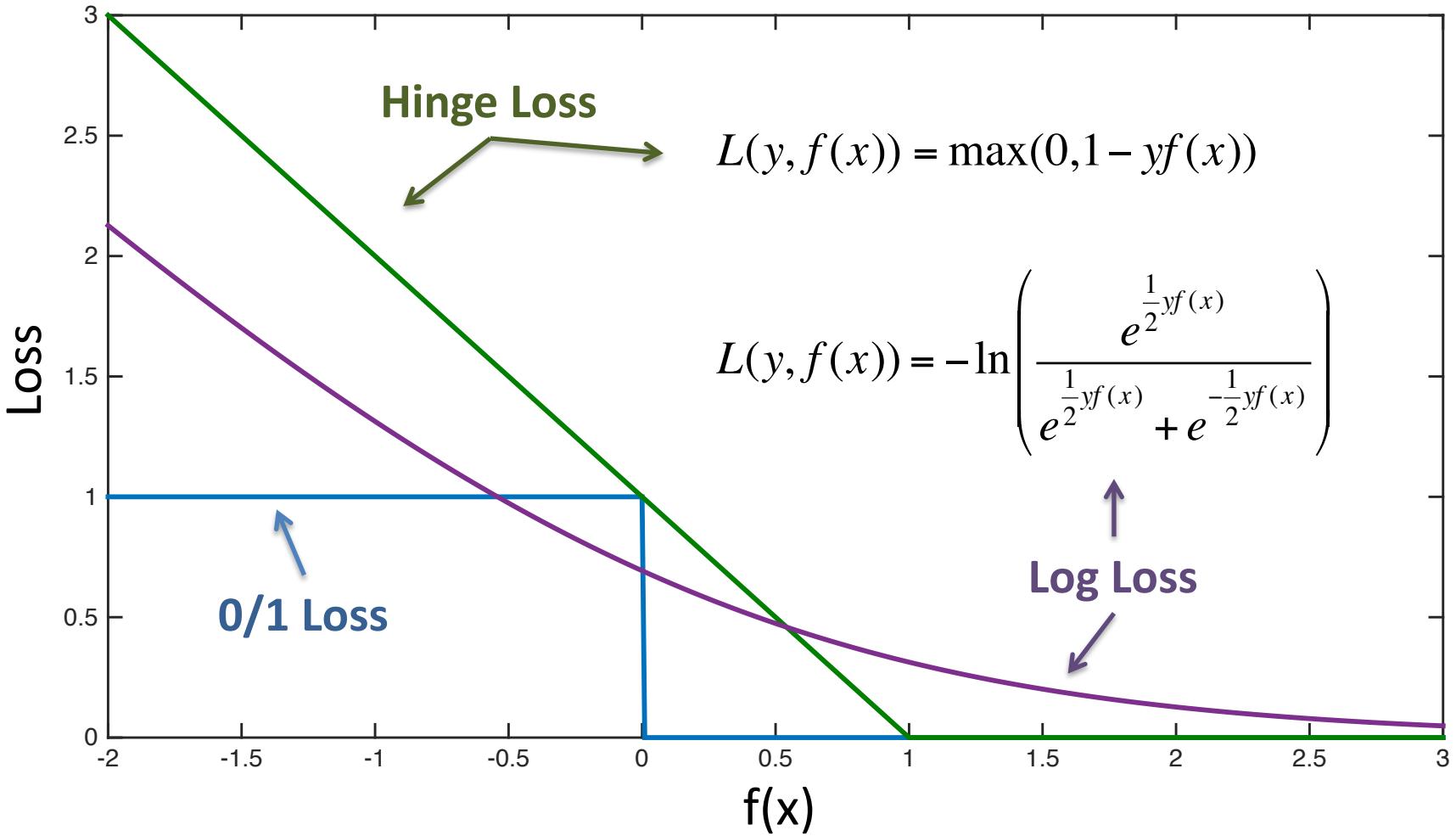
$$\underset{w,b}{\operatorname{argmax}} \prod_i P(y_i \mid x_i, w, b) = \underset{w,b}{\operatorname{argmin}} \sum_i -\ln P(y_i \mid x_i, w, b)$$


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
(Stoch.) Gradient Descent

Log Loss vs Hinge Loss



Log-Loss Gradient

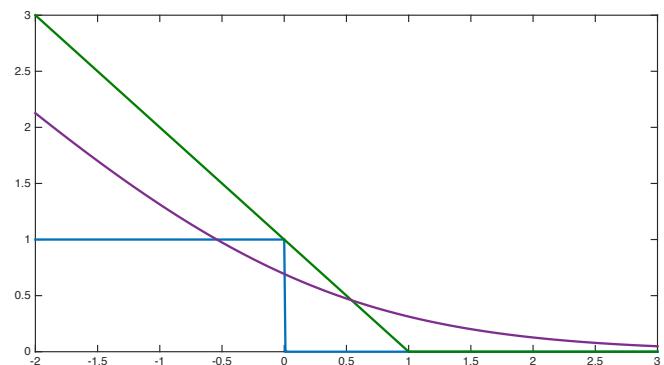
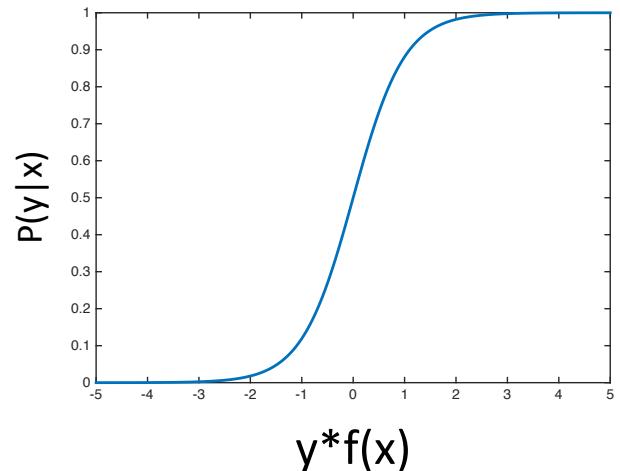
(For One Example)

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

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

Logistic Regression

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

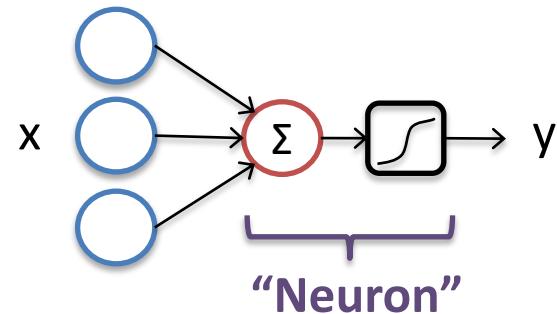


Feed-Forward Neural Networks

aka Not Quite Deep Learning

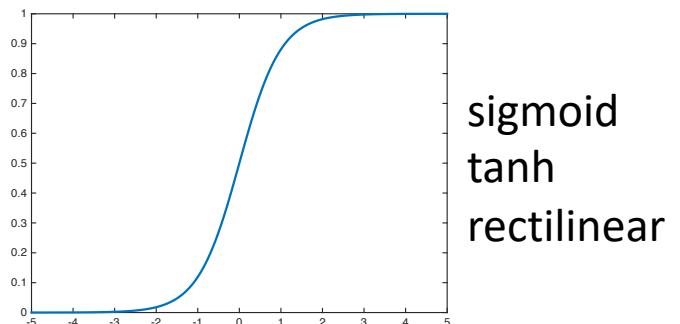
1 Layer Neural Network

- 1 Neuron
 - Takes input x
 - Outputs y

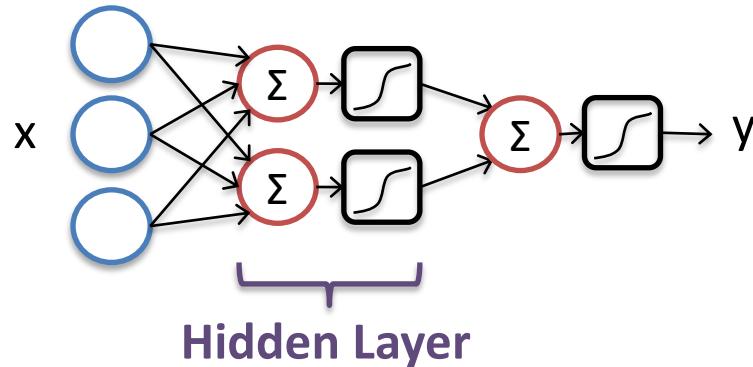


$$\begin{aligned} f(x|w,b) &= w^T x - b \\ &= w_1 * x_1 + w_2 * x_2 + w_3 * x_3 - b \end{aligned} \qquad \longrightarrow y = \sigma(f(x))$$

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



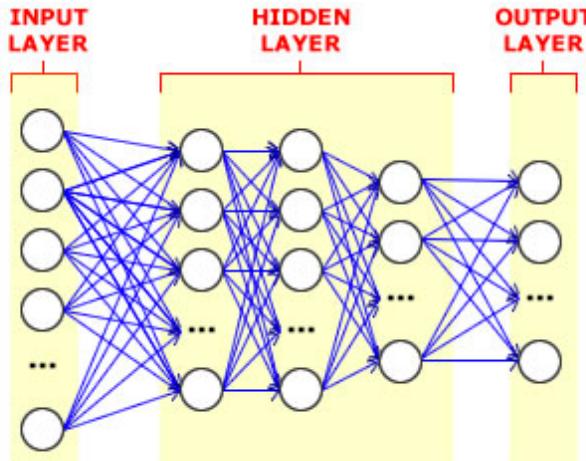
2 Layer Neural Network



- 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!

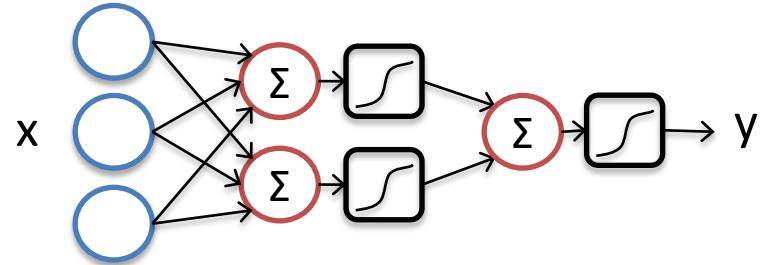
Aside: Deep Neural Networks



- Why prefer Deep over a “Fat” 2-Layer?
 - Compact model
 - (exponentially large “fat” model)
 - Easier to train?
 - Discussed further in deep learning lectures

Training Neural Networks

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



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

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

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

**Backpropagation = Gradient Descent
(lots of chain rules)**

*more complicated

Story So Far

- Different Loss Functions
 - Hinge Loss
 - Log Loss
 - Can be derived from different interpretations
- Non-Linear Model Classes
 - Neural Nets
 - Composable with different loss functions
- No closed-form solution for training
 - Must use some form of gradient descent

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	
Loss Function		Has Cancer	Doesn't Have Cancer
Model	Predicts Cancer		
	Predicts No Cancer		

- 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):

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

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

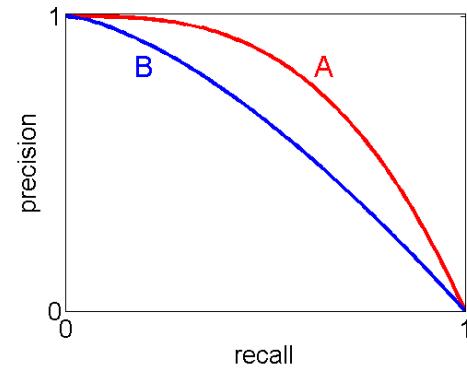
Precision & Recall

- **Precision** = $TP/(TP + FP)$ $F1 = 2/(1/P+ 1/R)$
- **Recall** = $TP/(TP + FN)$

Care More About Positives!

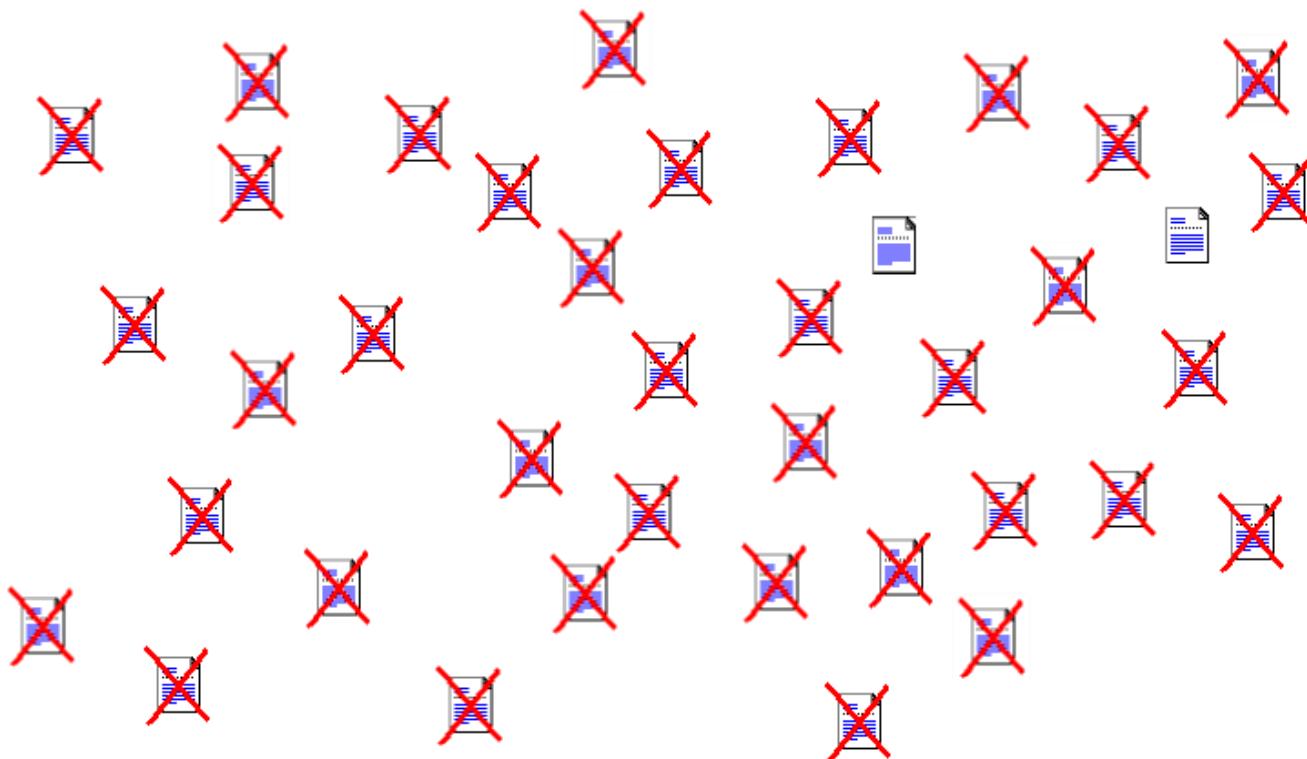
		Patient	
Counts		Has Cancer	Doesn't Have Cancer
Model	Predicts Cancer	20 (TP)	30 (FP)
	Predicts No Cancer	5 (FN)	70 (TN)

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



Example: Search Query

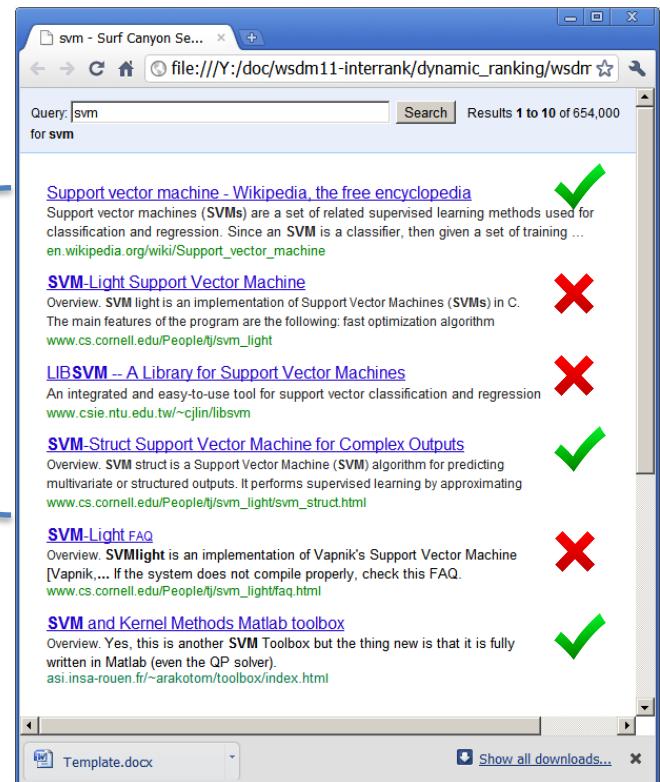
- Rank webpages by relevance



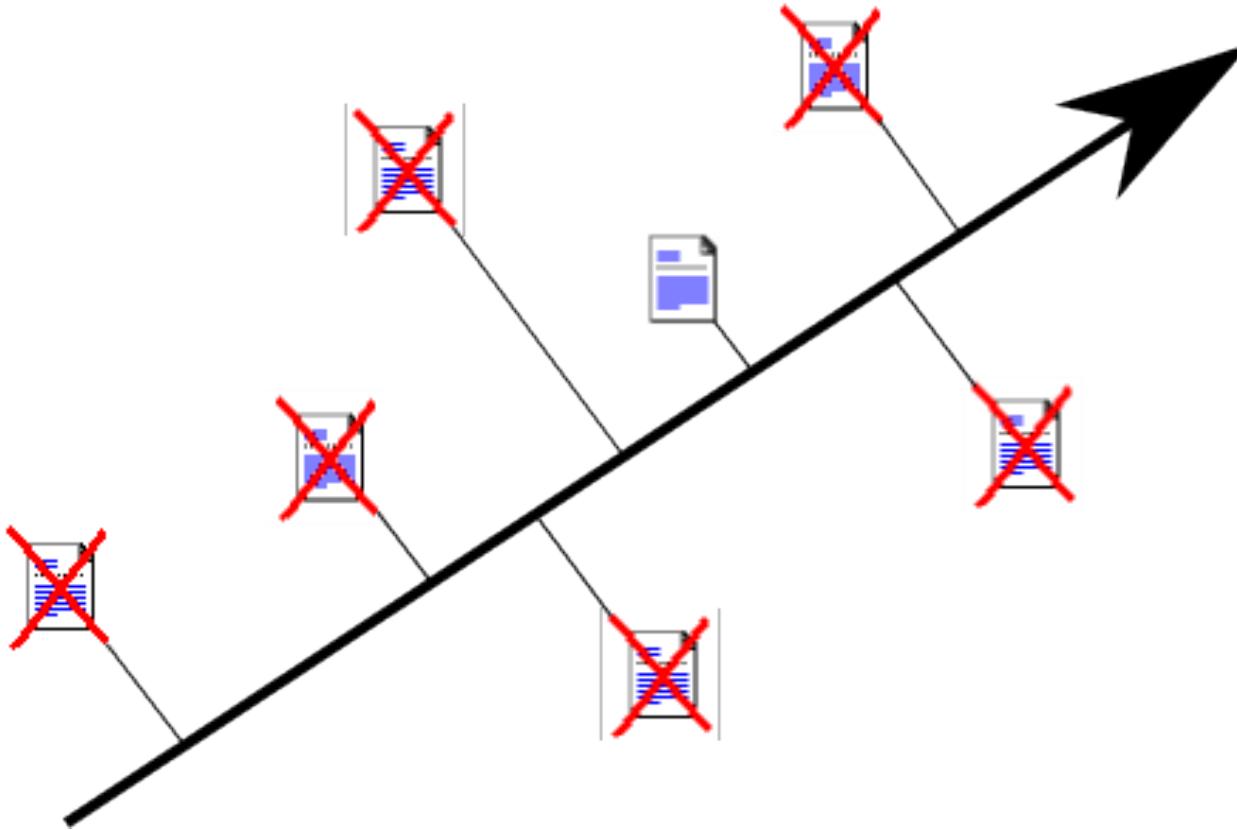
Ranking Measures

- 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!

Top 4



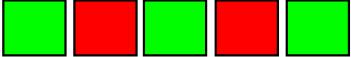
Pairwise Preferences



2 Pairwise Disagreements

4 Pairwise Agreements

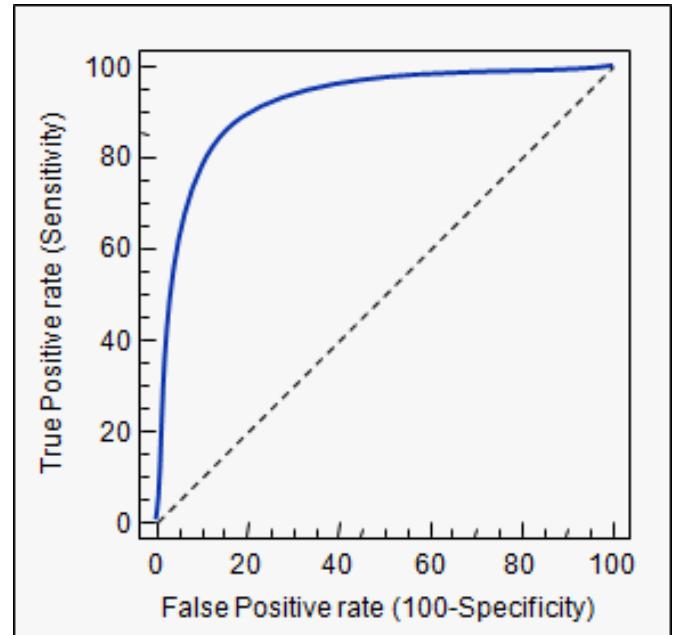
ROC-Area

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

ROC-Area: 0.5

#Pairwise Preferences = 6

#Agreements = 3



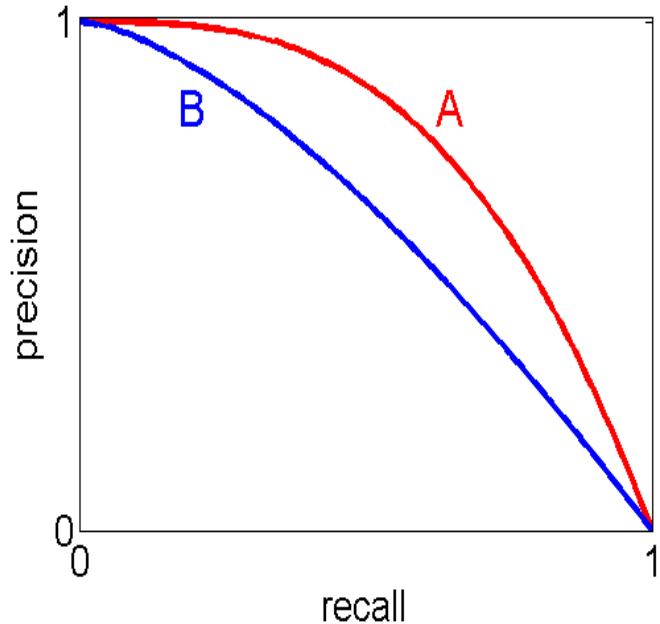
Average Precision

- 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



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

Other Challenges



- “Correct” if overlap is large enough
- How to define large enough?
- Duplicate detections?
- What is learning objective?
- Similar challenges in videos:
 - E.g., temporal bounding box around “running” activity
 - Duplicate predictions: break into two separate running activities
- Other examples: heart-rate monitoring

Summary: Evaluation Measures

- Different Evaluations Measures
 - Different Scenarios
- Large focus on getting positives
 - Large cost of mis-predicting cancer
 - Relevant webpages are rare
 - Aka “Class Imbalance”
- Other challenges:
 - localization in continuous domain

Next Lecture

- Regularization
- Lasso
- Thursday:
 - Recitation on Matrix Linear Algebra (& Calculus)