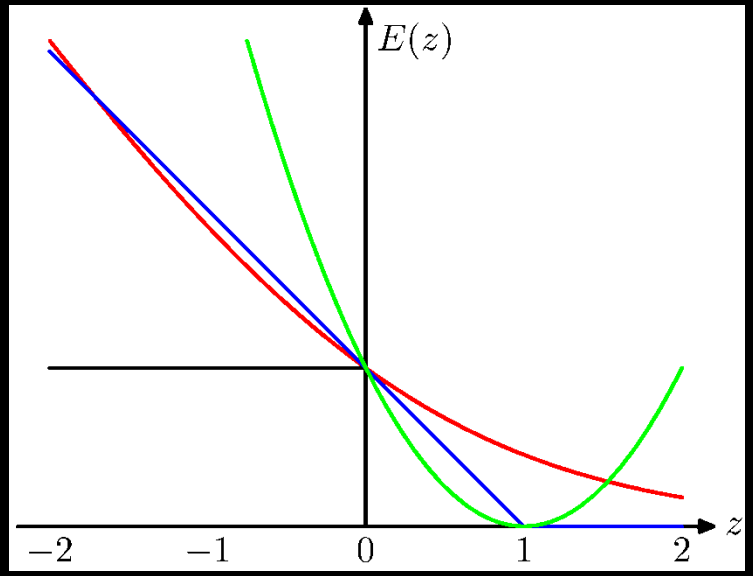
Machine Learning for Computer Vision

3 October, 2013
MVA – ENS Cachan



Lecture 2: Logistic regression & intro to MIL

Iasonas Kokkinos

Iasonas.kokkinos@ecp.fr

Department of Applied Mathematics
Ecole Centrale Paris

Galen Group
INRIA-Saclay

Administrative details

- http://vision.mas.ecp.fr/vision/Personnel/iasonas/teaching.html

- Make sure you have all received my email from last week

Lecture outline

Recap & problems of linear regression

Logistic Regression

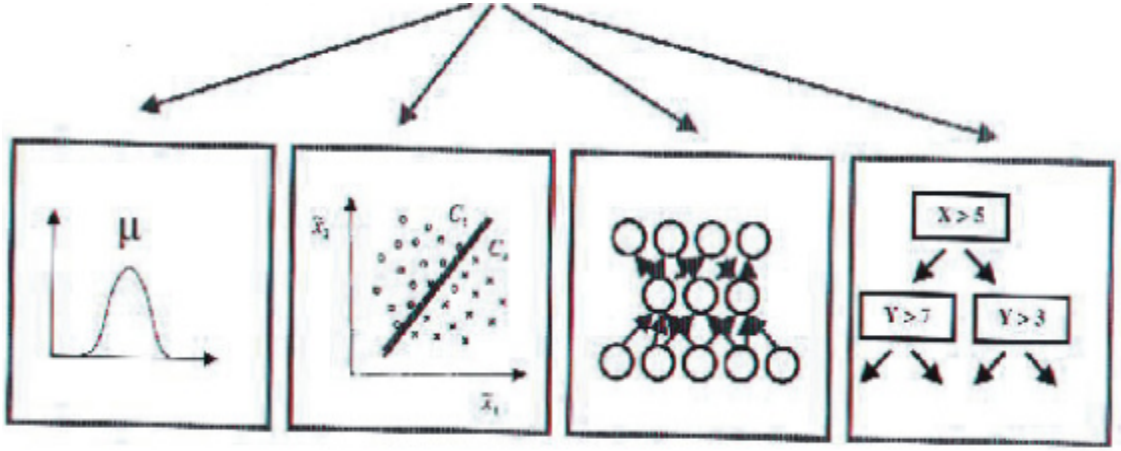
Introduction to Multiple Instance Learning

Learning problem

- Recover input-output mapping

  - Output: y
  - Input: x
  - Method: f
  - Parameters: w

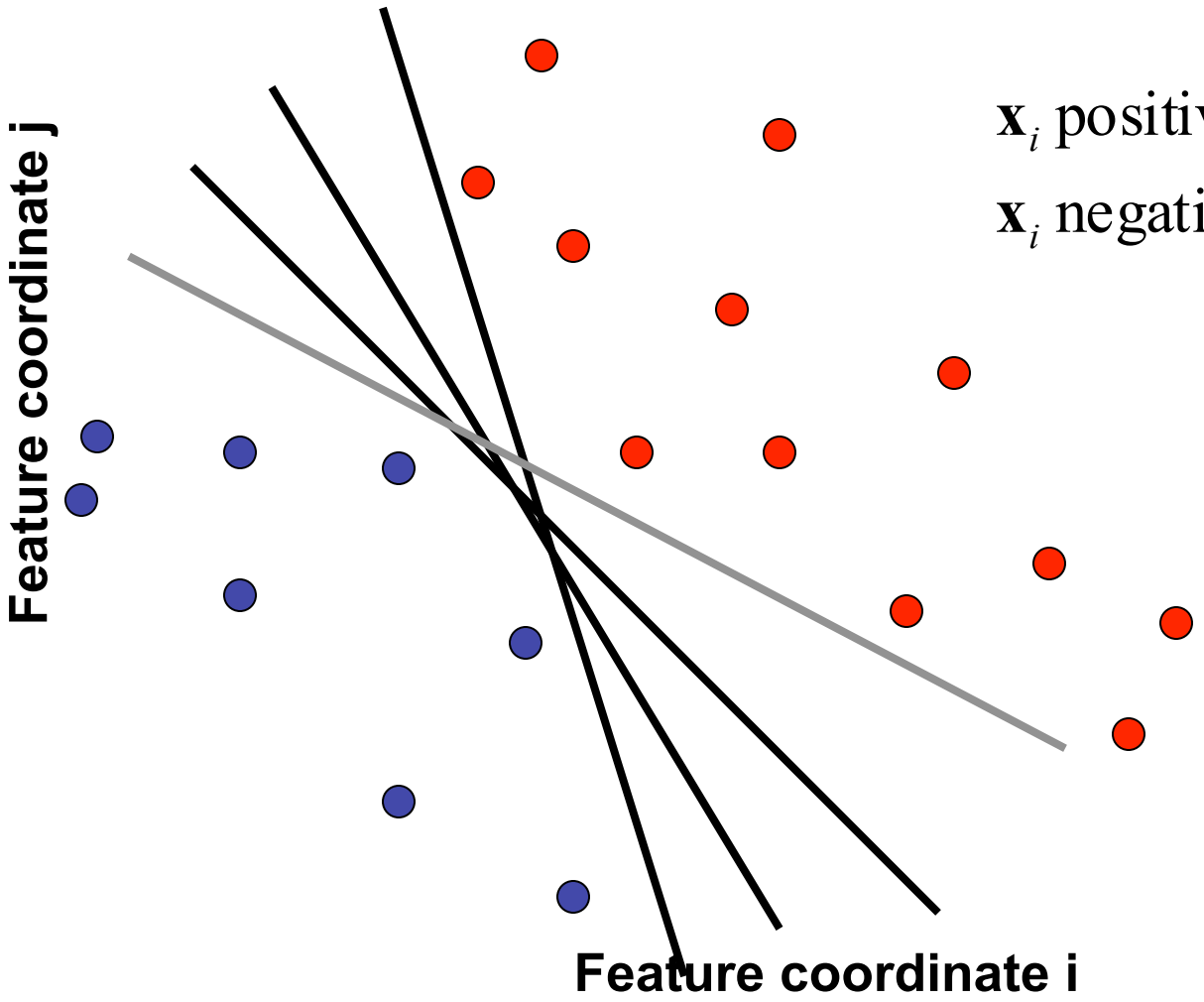
$$y = f_w(x) \quad (= f(x, w))$$



- Aspects of the learning problem
  - Identify methods that fit the problem setting
  - Determine parameters that properly classify the training set
  - Measure and control the complexity of these functions

Linear Classifiers

- Find linear expression (*hyperplane*) to separate positive and negative examples



$$\mathbf{x}_i \text{ positive}: \quad \mathbf{x}_i \cdot \mathbf{w} + b \geq 0$$

$$\mathbf{x}_i \text{ negative}: \quad \mathbf{x}_i \cdot \mathbf{w} + b < 0$$

Each data point has a class label:

$$y_t = \begin{cases} +1 & (\bullet) \\ -1 & (\circ) \end{cases}$$

Learning problem formulation

Given: Training set of feature-label pairs $S = \{(x^i, y^i)\}\ i = 1, \ldots, N$

$$y^i \in \{0, 1\} \quad x^i \in \mathcal{X}$$

Wanted: simple $f : \mathcal{X} \rightarrow \{0, 1\}$ that works well for S

simple: penalizing function complexity, to guarantee generalization

works well: quantified by loss criterion $L(S, f)$

Linear regression

Linear $f : R^K \rightarrow R$ $\quad y = f_w(x) =< x, w >= \sum_{k=1}^{K} x_k w_k$

Loss function: quantify appropriateness of w for $S = \{(x^i, y^i)\}\ i = 1, \ldots, N$

$$L(w) = \sum_{i=1}^{N} (y^i - < x^i, w >)^2$$

Introduce vector notation

$$\mathbf{y} = \begin{bmatrix} y_1 \\ \vdots \\ y_N \end{bmatrix} \quad \mathbf{X} = \begin{bmatrix} x_1^1 & \ldots & x_k^1 & \ldots & x_K^1 \\ & & \vdots & & \\ x_1^N & \ldots & x_k^N & \ldots & x_K^N \end{bmatrix} \quad \mathbf{w} = \begin{bmatrix} w_1 \\ \vdots \\ w_k \\ \vdots \\ w_K \end{bmatrix}$$

$$\mathbf{e} = \mathbf{y} - \mathbf{X}\mathbf{w}$$

$$L(\mathbf{w}) = \mathbf{e}^T\mathbf{e} \qquad \mathbf{w}^* = \left(\mathbf{X}^T\mathbf{X}\right)^{-1}\mathbf{X}^T\mathbf{y}$$

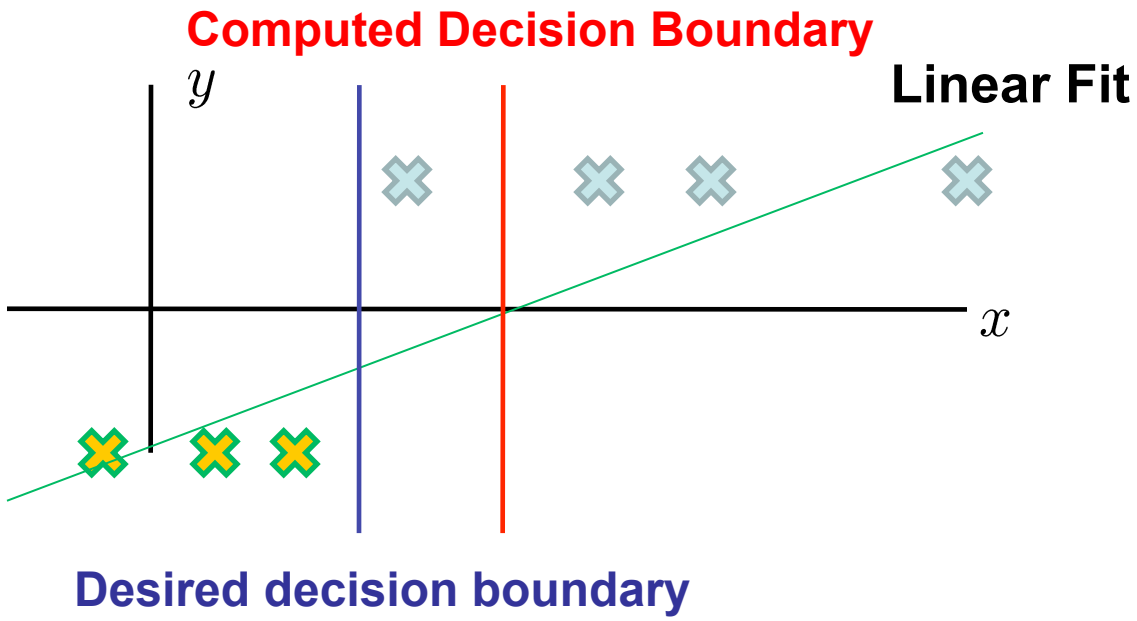
$$L(\mathbf{w}) = \mathbf{e}^T\mathbf{e} + \lambda\mathbf{w}^T\mathbf{w} \qquad \mathbf{w}^* = \left(\mathbf{X}^T\mathbf{X} + \lambda\mathbf{I}\right)^{-1}\mathbf{X}^T\mathbf{y}$$

Inappropriateness of quadratic loss

We chose the quadratic cost function for convenience

Single, global minimum & closed form expression

But does it indicate classification performance?

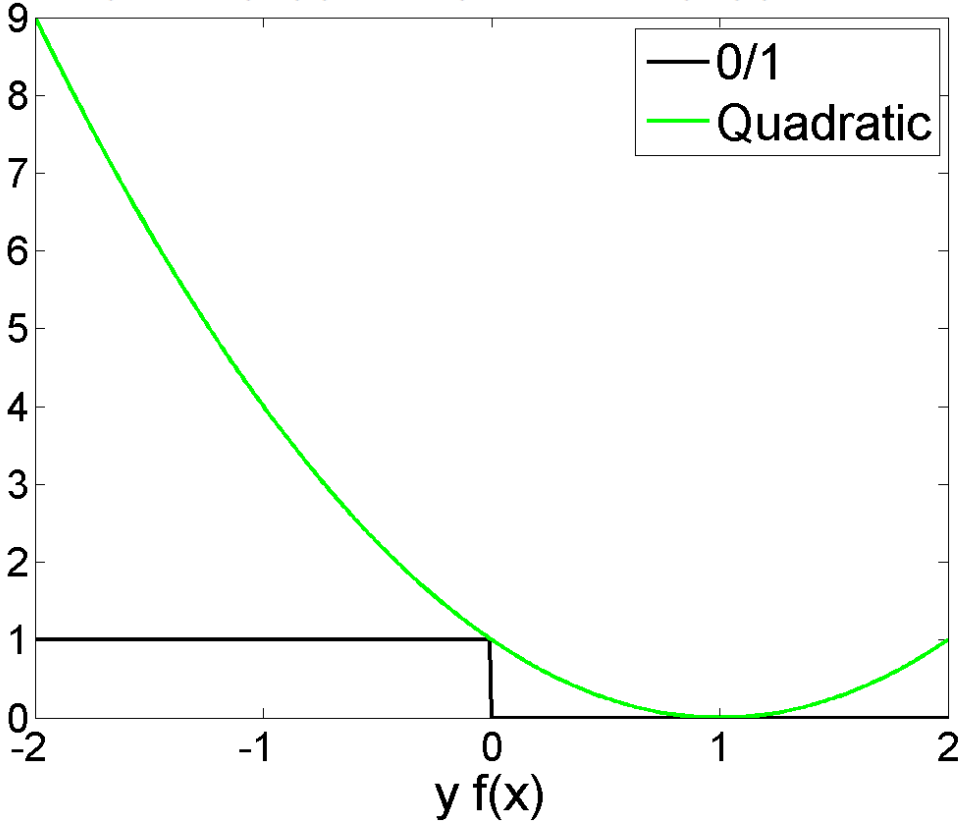


Inappropriateness of quadratic loss

Consider transformation: $y_\pm = 2y_b - 1 \qquad y_b \in \{0,1\}\ y_\pm \in \{-1,1\}$

$$l(y, f(x)) = (1 - yf(x))^2$$

$$\begin{aligned} l(y, f(x)) &= (y - f(x))^2 \\ &\overset{y^2=1}{=} y^2(y - f(x))^2 \\ &= (y^2 - yf(x))^2 \\ &\overset{y^2=1}{=} (1 - yf(x))^2 \end{aligned}$$



Quadratic loss is not robust to outliers and penalizes outputs that are `too good'

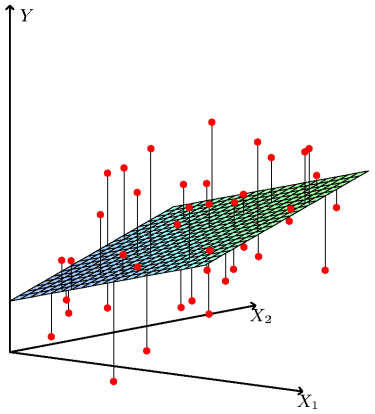
Lecture outline

- Recap & problems of linear regression
- Logistic Regression
  - Training criterion formulation
  - Optimization
  - Interpretation
  - Application to boundary detection
- Introduction to Multiple Instance Learning

Probabilistic interpretation of least squares

Outputs: continuous random variables

linear function of inputs + zero mean Gaussian r.v.



$$P(y|\mathbf{x},\mathbf{w}) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(y- < \mathbf{x},\mathbf{w} >)^2}{2\sigma^2}\right)$$

Optimal w: maximizes likelihood of observed outputs

$$C(\mathbf{w}) = \log P(\mathbf{y}|\mathbf{X},\mathbf{w}) = \sum_{i=1}^{N} \log P(y^i|\mathbf{x}^i,\mathbf{w})$$

$\alpha = -\frac{1}{2\sigma^2}$ $c = -N/2\log(2\pi\sigma^2)$

$$= c + a\sum_{i=1}^{N}(y^i - \mathbf{x}^i\mathbf{w})^2 = c + a\,(\mathbf{y}-\mathbf{X}\mathbf{w})^T(\mathbf{y}-\mathbf{X}\mathbf{w})$$

$$C(\mathbf{w}) = -|\alpha|L(\mathbf{w}) + c$$

But the labels are discrete!

A probabilistic criterion for training a classifier

Training set: $\{(\mathbf{x}^1, y^1), \ldots, (\mathbf{x}^N, y^N)\}, \quad \mathbf{x} \in R^M, \quad y \in \{0, 1\}$

y: discrete observations: model as samples from Bernoulli distribution

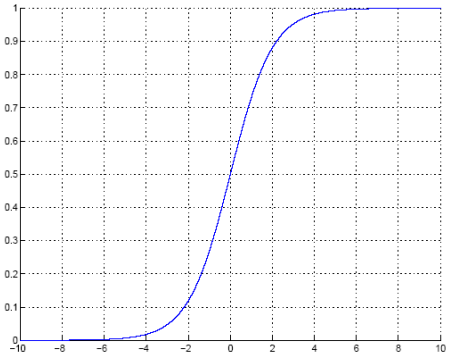
$$P(y = 1|\mathbf{x}, \mathbf{w}) = f(\mathbf{x}, \mathbf{w})$$
$$P(y = 0|\mathbf{x}, \mathbf{w}) = 1 - f(\mathbf{x}, \mathbf{w})$$
$$P(y|\mathbf{x}) = (f(\mathbf{x}, \mathbf{w}))^y (1 - f(\mathbf{x}, \mathbf{w}))^{1-y}$$

Find w that maximizes the likelihood of labels in the training set

$$-L(\mathbf{w}) = C(\mathbf{w}) = \log P(\mathbf{y}|\mathbf{X}, \mathbf{w}) = \sum_{i=1}^{N} \log P(y^i|\mathbf{x}^i, \mathbf{w})$$
$$= \sum_i y^i \log f(\mathbf{x}^i, \mathbf{w}) + (1 - y^i) \log(1 - f(\mathbf{x}^i, \mathbf{w}))$$

Sigmoidal function & logistic regression

$$P(y=1|\mathbf{x},\mathbf{w}) = f(\mathbf{x},\mathbf{w}) = g(<\mathbf{x},\mathbf{w}>) = \frac{1}{1+\exp(-<\mathbf{x},\mathbf{w}>)}$$



$$g(a) = \frac{1}{1+\exp(-a)}$$ sigmoidal

Training criterion from previous slide:

$$C(\mathbf{w}) = \sum_{i=1}^{N} y^i \log g(<\mathbf{x}^i,\mathbf{w}>) + (1-y^i)\log\left(1-g(<\mathbf{x}^i,\mathbf{w}>)\right)$$

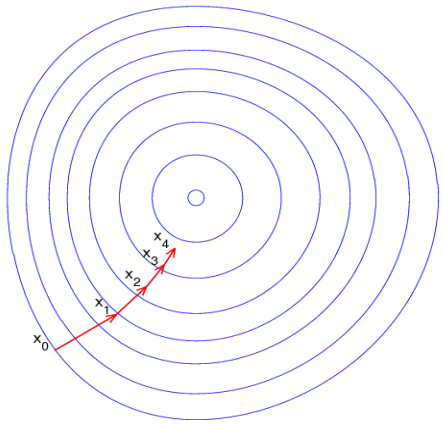
How do we optimize it with respect to w?

What does it mean?

Lecture outline

- Recap & problems of linear regression
- Logistic Regression
  - Training criterion formulation
  - Optimization
  - Interpretation
  - Application to boundary detection
- Introduction to Multiple Instance Learning

Maximization by gradient ascent



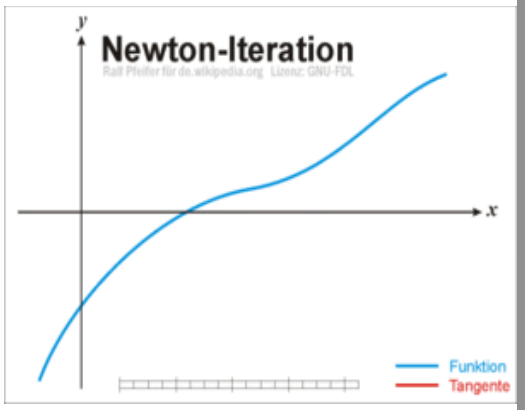
Gradient ascent: $$w_k' = w_k + \alpha \frac{\partial C(\mathbf{w})}{\partial w_k}$$

$$C(\mathbf{w}) = \sum_i y^i \log g(\mathbf{x}^i\mathbf{w}) + (1-y^i)\log(1-g(\mathbf{x}^i\mathbf{w}))$$

$$\begin{aligned}
\frac{\partial C(\mathbf{w})}{\partial w_k} &= \sum_{i=1}^{N}\left[y^i \frac{1}{g(\mathbf{x}^i\mathbf{w})}\frac{\partial g(\mathbf{x}^i\mathbf{w})}{\partial w_k} + (1-y^i)\frac{1}{1-g(\mathbf{x}^i\mathbf{w})}(-\frac{\partial g(\mathbf{x}^i\mathbf{w})}{\partial w_k})\right] \\
&= \sum_{i=1}^{N}\left[y^i \frac{1}{g(\mathbf{x}^i\mathbf{w})} - (1-y^i)\frac{1}{1-g(\mathbf{x}^i\mathbf{w})}\right] g(\mathbf{x}^i\mathbf{w})(1-g(\mathbf{x}^i\mathbf{w}))\frac{\partial \mathbf{x}^i\mathbf{w}}{\partial w_k} \\
&= \sum_{i=1}^{N}\left[y^i(1-g(\mathbf{x}^i\mathbf{w})) - (1-y^i)g(\mathbf{x}^i\mathbf{w})\right]\mathbf{x}_k^i \\
&= \sum_{i=1}^{N}\left[y^i - g(\mathbf{x}^i\mathbf{w})\right]\mathbf{x}_k^i
\end{aligned}$$

$$\frac{dg}{da} = g(a)(1-g(a))$$

Maximization with Newton-Raphson



Newton method for finding roots of 1D function $f(x)$:

$$f(x) = f'(x_0)(x - x_0) + f(x_0) \rightarrow x = x_0 - \frac{f(x_0)}{f'(x_0)}$$

Newton method for finding maxima of 1D function $f(x)$:

$$x = x_0 - \frac{f'(x_0)}{f''(x_0)} \qquad \text{condition} \qquad f''(x) < 0$$

Newton-Raphson method for finding maxima of N-D function $f(\mathbf{x})$:

$$\mathbf{x} = \mathbf{x_0} - [H(\mathbf{x_0})]^{-1} J(\mathbf{x_0}) \qquad H(\mathbf{x}) \prec 0$$

Newton-Raphson for Logistic Regression

Jacobian: $\underbrace{J(\mathbf{w})}_{M\times 1} = \frac{dC(\mathbf{w})}{d\mathbf{w}} = \mathbf{X}^T \underbrace{[\mathbf{y} - \mathbf{g}]}_{N\times 1}$ $\quad \mathbf{X} = \begin{bmatrix} x^1 \\ \vdots \\ x^N \end{bmatrix} \quad N \times M$

$$\frac{\partial C(\mathbf{w})}{\partial w_k} = \sum_{i=1}^{N} \mathbf{x}_k^i \left[y^i - g(\mathbf{x}^i \mathbf{w}) \right]$$

Hessian:

$$H(\mathbf{w})(k,j) = \frac{\partial^2 C(\mathbf{w})}{\partial w_k \partial w_j} = \frac{\partial \sum_{i=1}^{N} \left[y^i - g(\mathbf{x}^i \mathbf{w}) \right] \mathbf{x}_k^i}{\partial w_j}$$

$$= -\sum_{i=1}^{N} \frac{\partial g(\mathbf{x}^i \mathbf{w})}{\partial w_j} \mathbf{x}_k^i$$

$$H(\mathbf{w}) = -\mathbf{X}^T \mathbf{R} \mathbf{X}, \quad R_{i,i} = g(\mathbf{x}^i \mathbf{w})(1 - g(\mathbf{x}^i \mathbf{w}))$$

In statistics: Iteratively Reweighted Least Squares (IRLS)

Lecture outline

- Recap & problems of linear regression
- Logistic Regression
  - Training criterion formulation
  - Optimization
  - Interpretation
  - Application to boundary detection
- Introduction to Support Vector Machines

A compact expression for the loss

Using $h_{\mathbf{w}}(\mathbf{x}) = < \mathbf{x}, \mathbf{w} >$, $y_{\pm} = 2y_b - 1$:

$$P(y = 1|h_{\mathbf{w}}(\mathbf{x})) = \frac{1}{1 + \exp(-h_{\mathbf{w}}(\mathbf{x}))}$$

$$P(y = -1|h_{\mathbf{w}}(\mathbf{x})) = 1 - P(y = 1|h_{\mathbf{w}}(\mathbf{x})) = 1 - \frac{1}{1 + \exp(-h_{\mathbf{w}}(\mathbf{x}))}$$

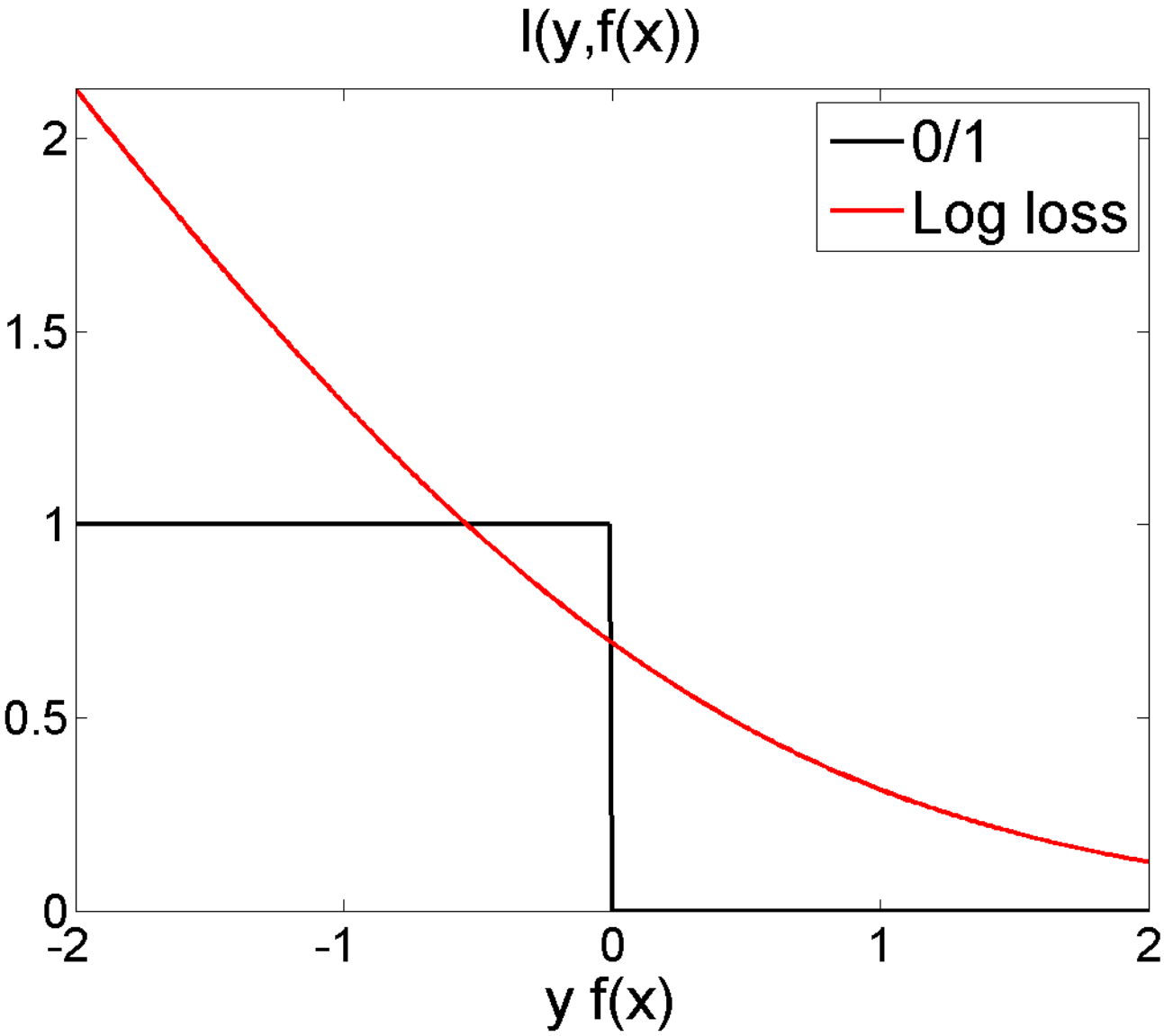
$$= \frac{\exp(-h_{\mathbf{w}}(\mathbf{x}))}{1 + \exp(-h_{\mathbf{w}}(\mathbf{x}))} = \frac{1}{\exp(h_{\mathbf{w}}(\mathbf{x})) + 1}$$

$$P(y = y^i|h_{\mathbf{w}}(\mathbf{x}^i)) = \frac{1}{1 + \exp(-y^i h_{\mathbf{w}}(\mathbf{x}^i))}$$

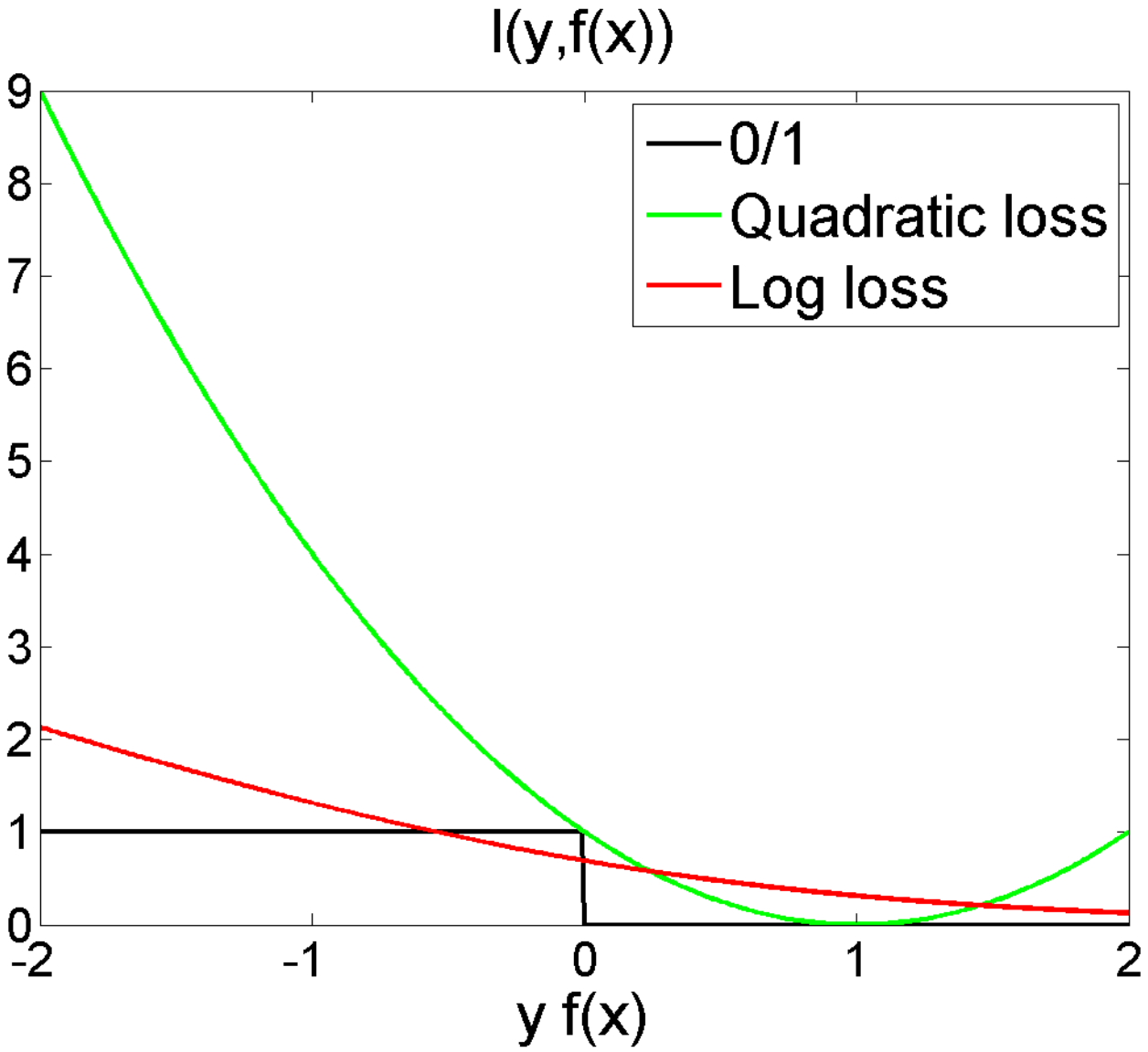
$$L(\mathbf{w}) = -C(\mathbf{w}) = \sum_{i=1}^{N} -\log P(y = y^i|h_{\mathbf{w}}(\mathbf{x}^i))$$

$$= \sum_{i=1}^{N} \log(1 + \exp(-y^i h_{\mathbf{w}}(\mathbf{x}^i)))$$

Log loss: $l(y, f(x)) = \log(1 + \exp(-yf(x)))$

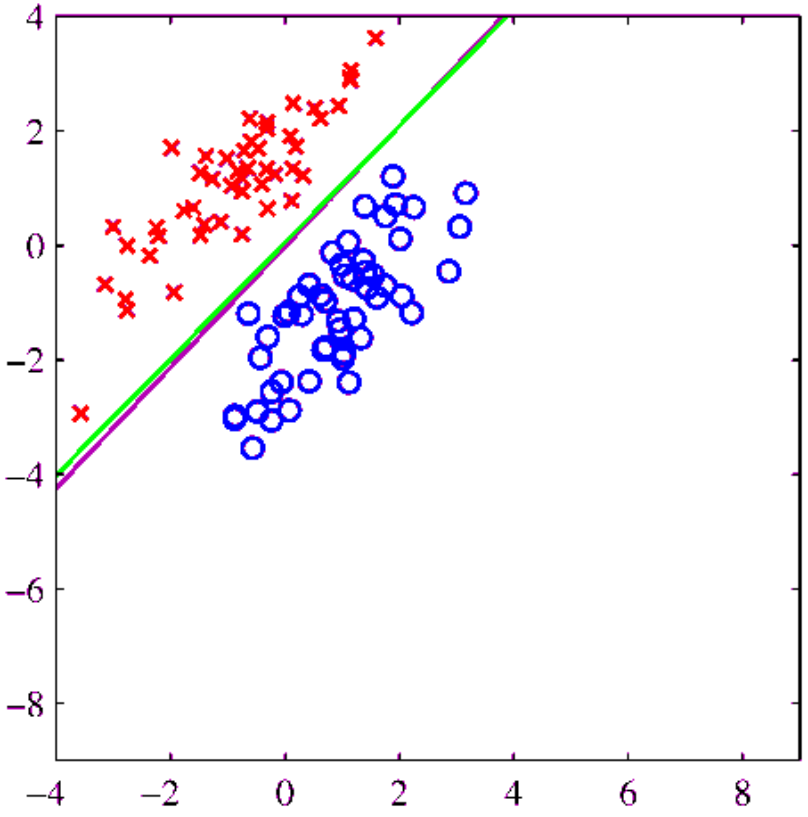


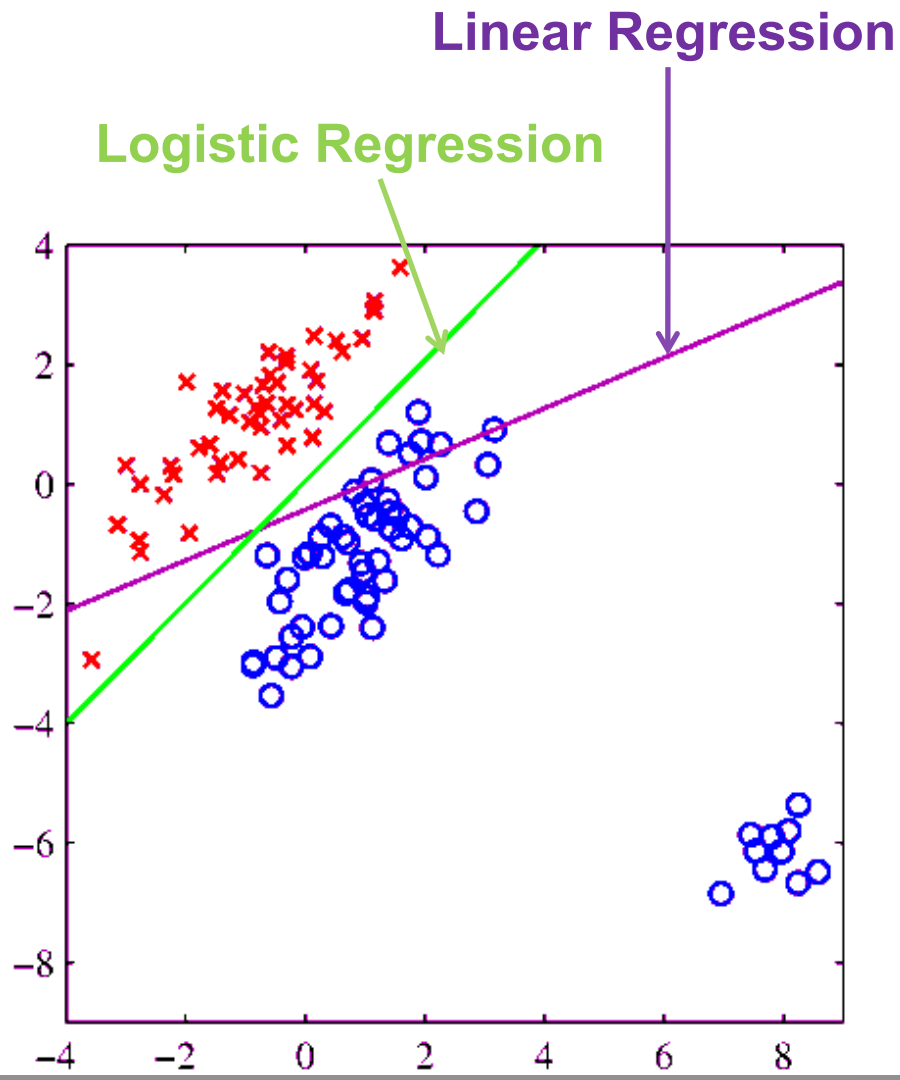
Log loss vs. quadratic loss



Logistic vs Linear Regression

Logistic regression is more robust





Lecture outline

Recap & problems of linear regression

Logistic Regression

Training criterion formulation

Optimization

Interpretation

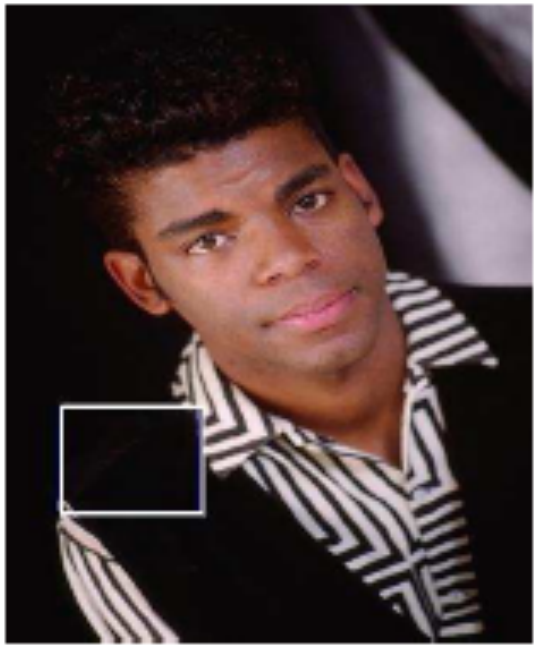
Application to boundary detection

Introduction to Multiple Instance Learning

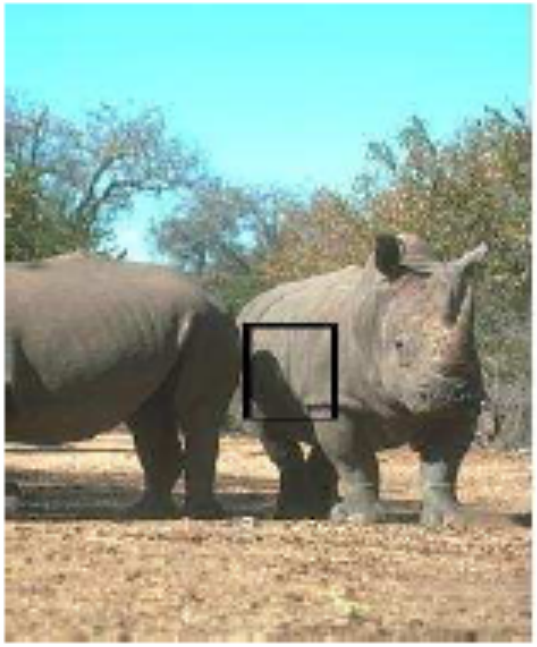
Boundary detection problem

Object/Surface Boundaries

Signal-level challenges



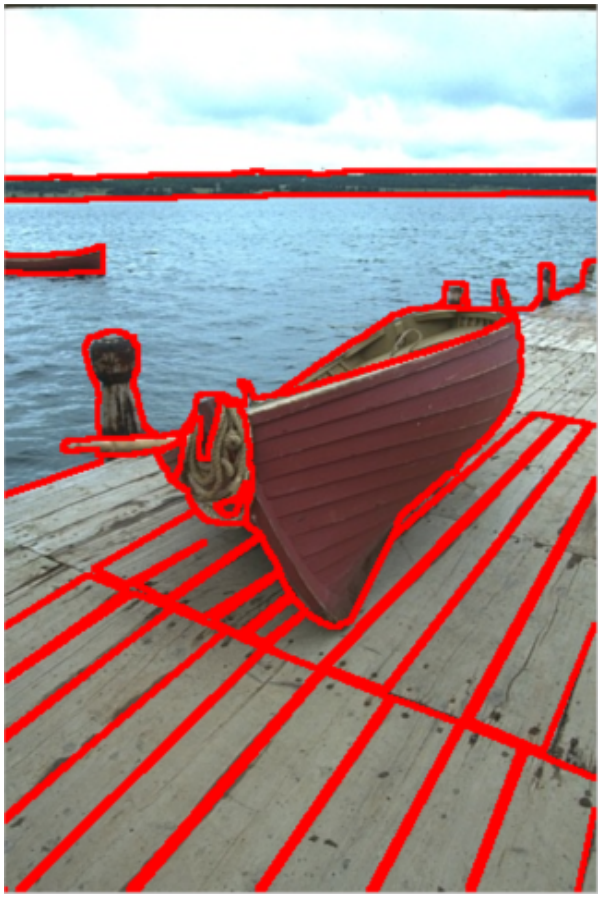
Poor contrast



Shadows

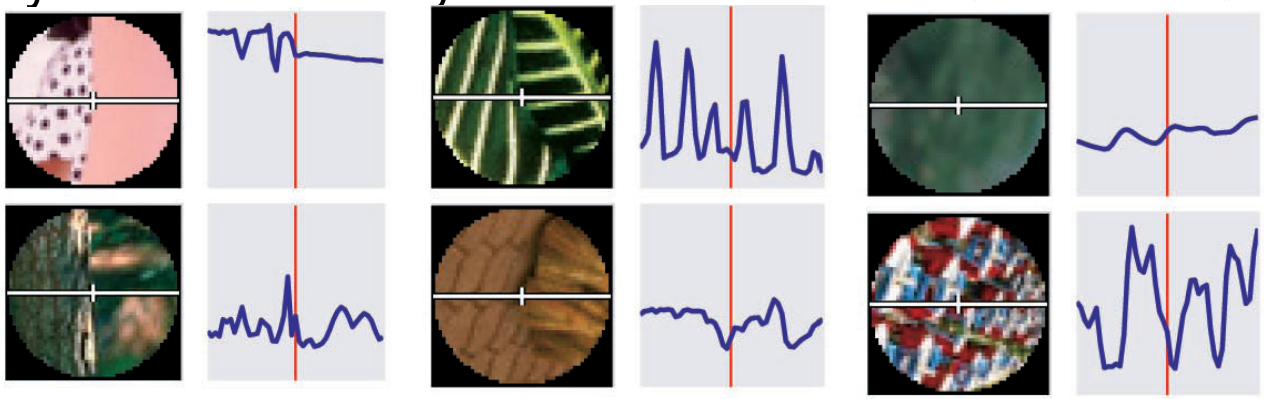
Texture

Fundamental challenges: can humans do it?

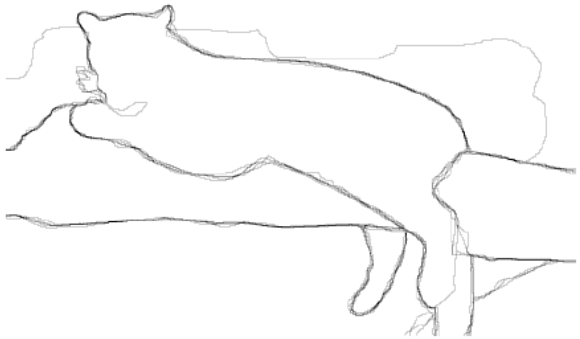


Learning-based approaches

Boundary or non-boundary?



Use human-annotated segmentations



Use any visual cue as input to the decision function.
Use decision trees/logisitic regression/boosting/… and *learn* to combine the individual inputs.

S. Konishi, A.Yuille, J. Coughlan, S.C. Zhu, “Statistical Edge Detection: Learning and Evaluating Edge Cues”, IEEE PAMI, 2003

D. Martin, C. Fowlkes, J. Malik. "Learning to Detect Natural Image Boundaries Using Local Brightness, Color and Texture Cues", IEEE PAMI, 2004

Evaluation protocol

Threshold detector's output at certain level

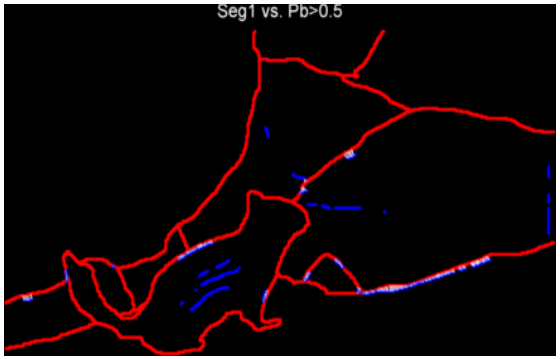
Match outputs to human ground-truth

Count true/false positives (t/f)positives, misses (m)

From precision, p and recall, r

$$p = \frac{\#\text{true positives}}{\#\text{detected}} = \frac{t}{t+f}$$

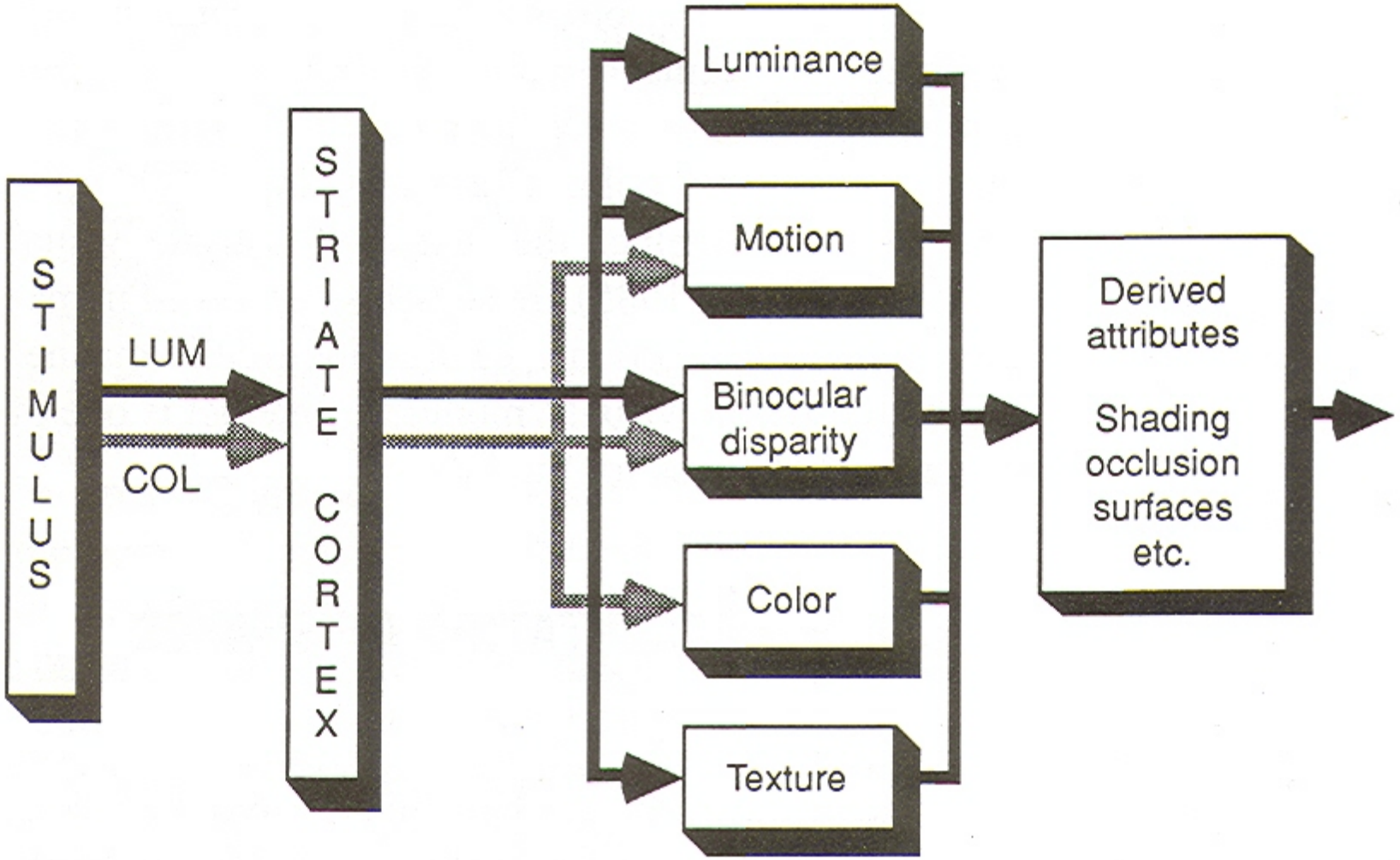
$$r = \frac{\#\text{true positives}}{\#\text{true}} = \frac{t}{t+m}$$



Quantify performance in terms of F-measure

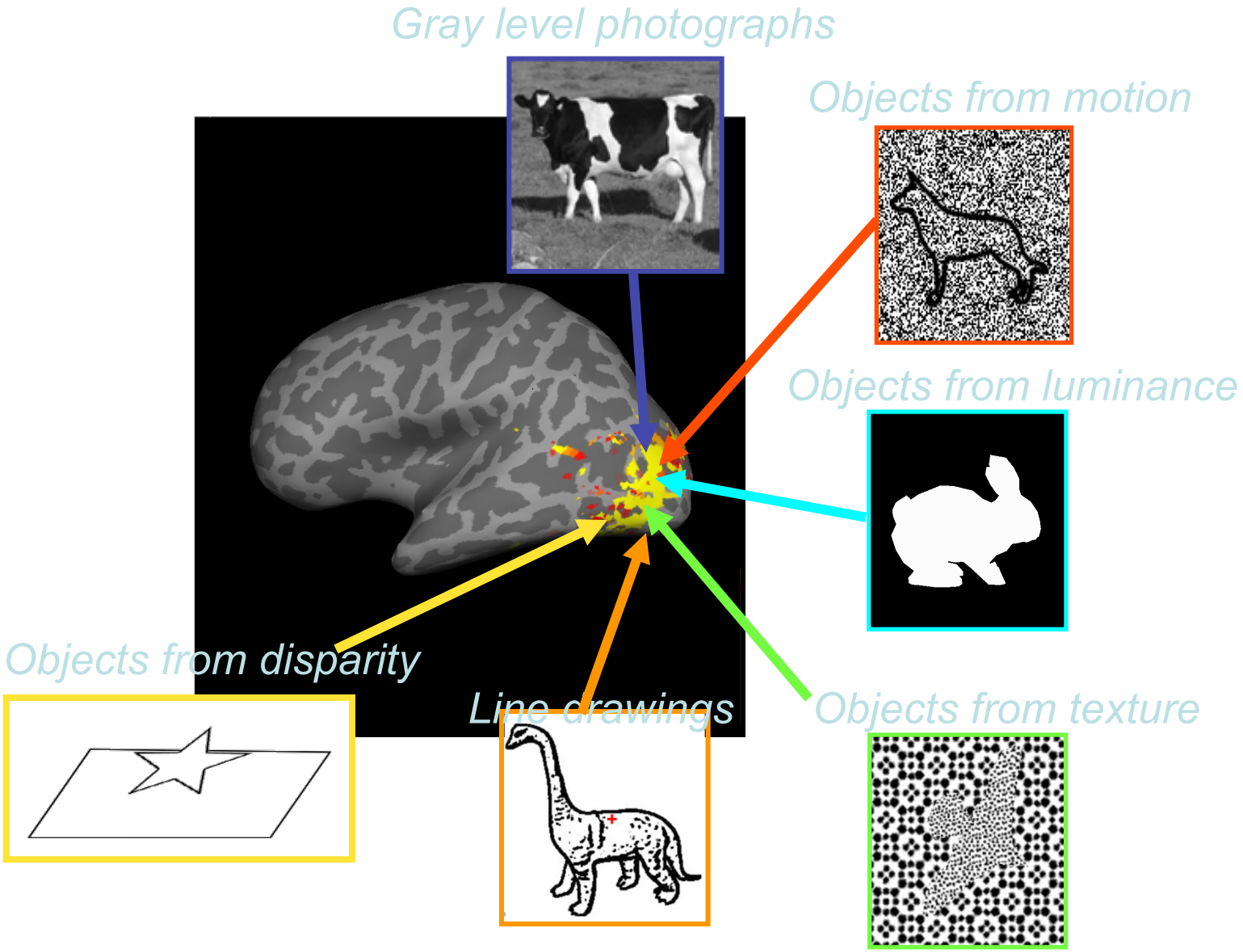
$$\mathcal{F} = \frac{2}{\frac{1}{p} + \frac{1}{r}} = \frac{pr}{p+r}$$

Contours can be defined by any of a number of cues (P. Cavanagh)



Slide credit: J. Malik, M. Maire

Cue-localization

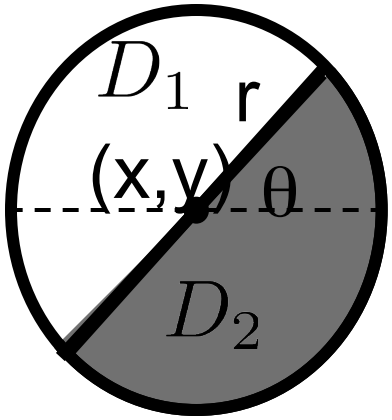


Slide credit: J. Malik, M. Maire

Grill-Spector et al. , Neuron 1998

Local boundary cues

Separate features per candidate orientation

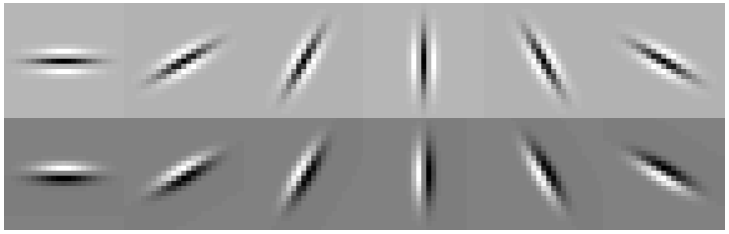


$$f = d\left(H(D_1), H(D_2)\right)$$

In specific: $f^{C,r}(x, y, \theta)$

$$C \in \{L, U, V, T\}, r \in \{1, 2, 3\}$$

- 1976 CIE L*a*b* colorspace
- Brightness Gradient BG(x,y,r,θ)
  - Difference of L* distributions
- Color Gradient CG(x,y,r,θ)
  - Difference of a*b* distributions
- Texture Gradient TG(x,y,r,θ)
  - Difference of distributions of V1-like filter responses



Slide credit: J. Malik, M. Maire

Boundary cues (horizontal)

Brightness

Color (a, b channels)

Texture

Boundary cues @ π/4

Brightness

Color (a, b channels)

Texture

Brightness cues @ π/2

Brightness

Color (a, b channels)

Texture

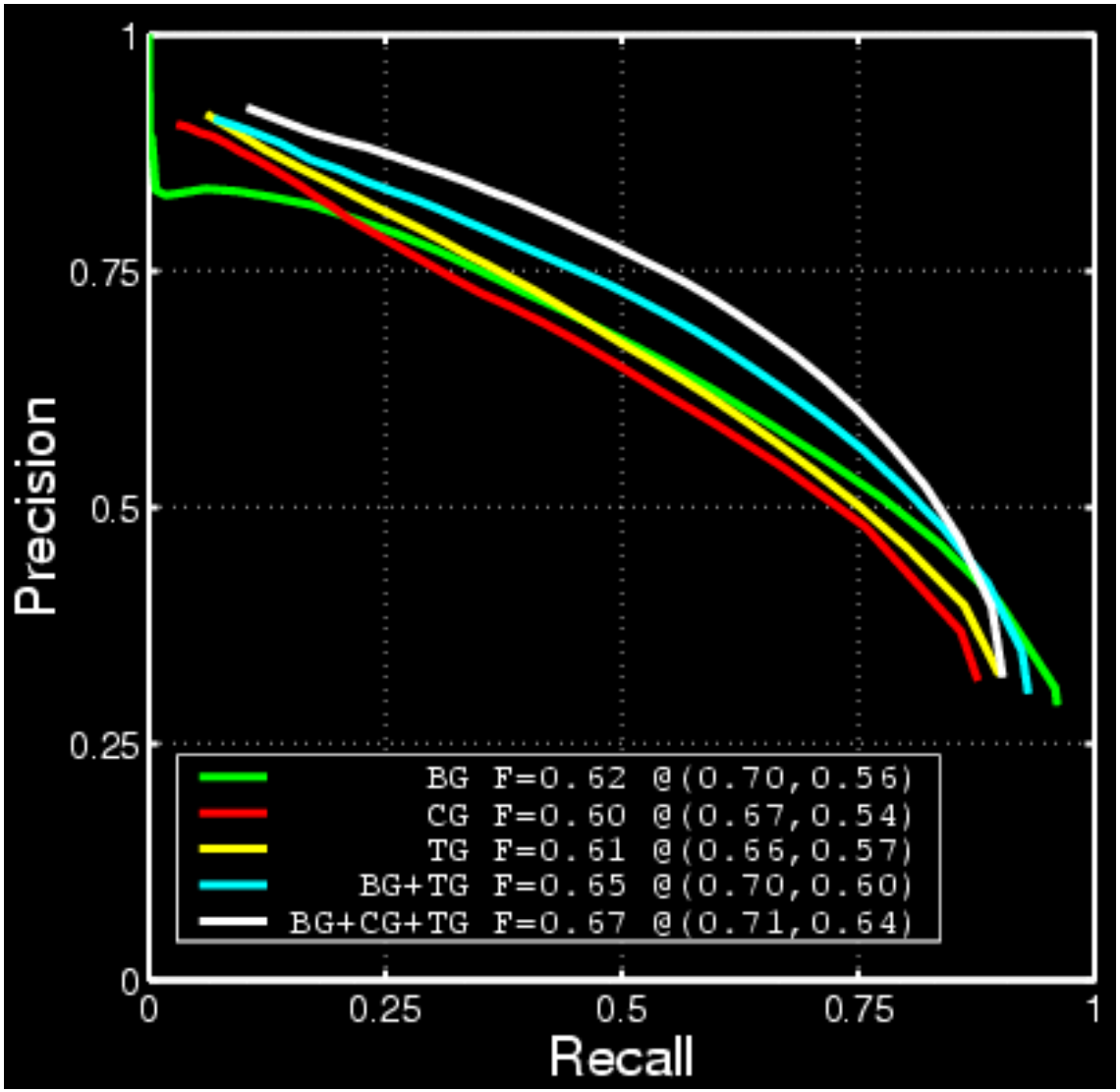
Brightness cues @ 3π/4

Brightness

Color (a, b channels)

Texture

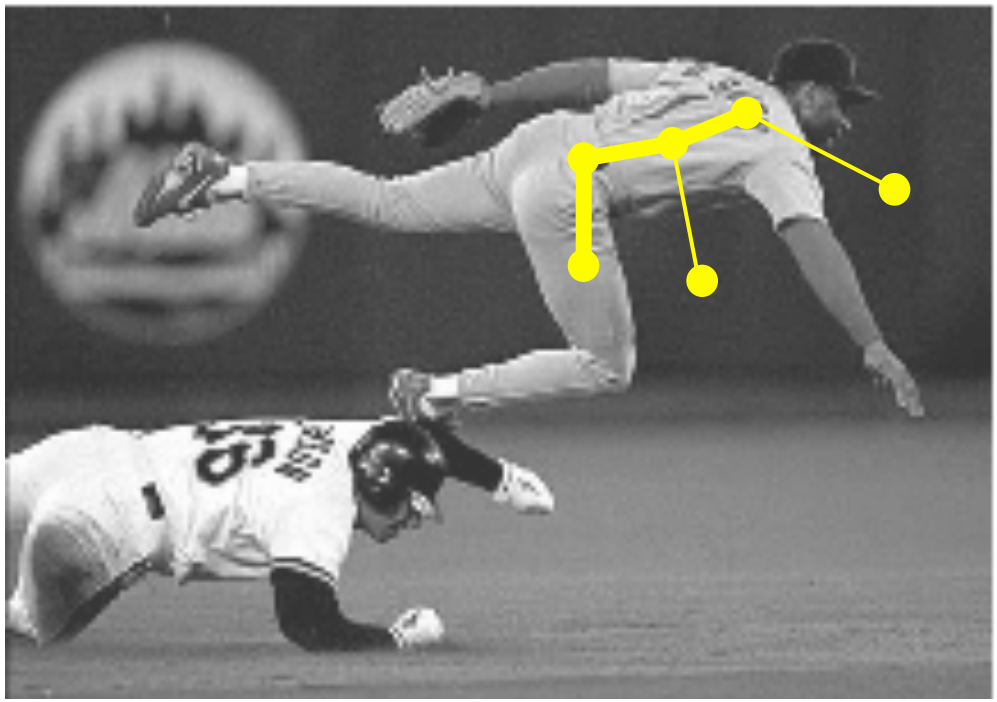
Cue combinations with logistic regression



Slide credit: J. Malik, M. Maire

Exploiting global constraints: image Segmentation as Graph Partitioning

Build a weighted graph G=(V,E) from image



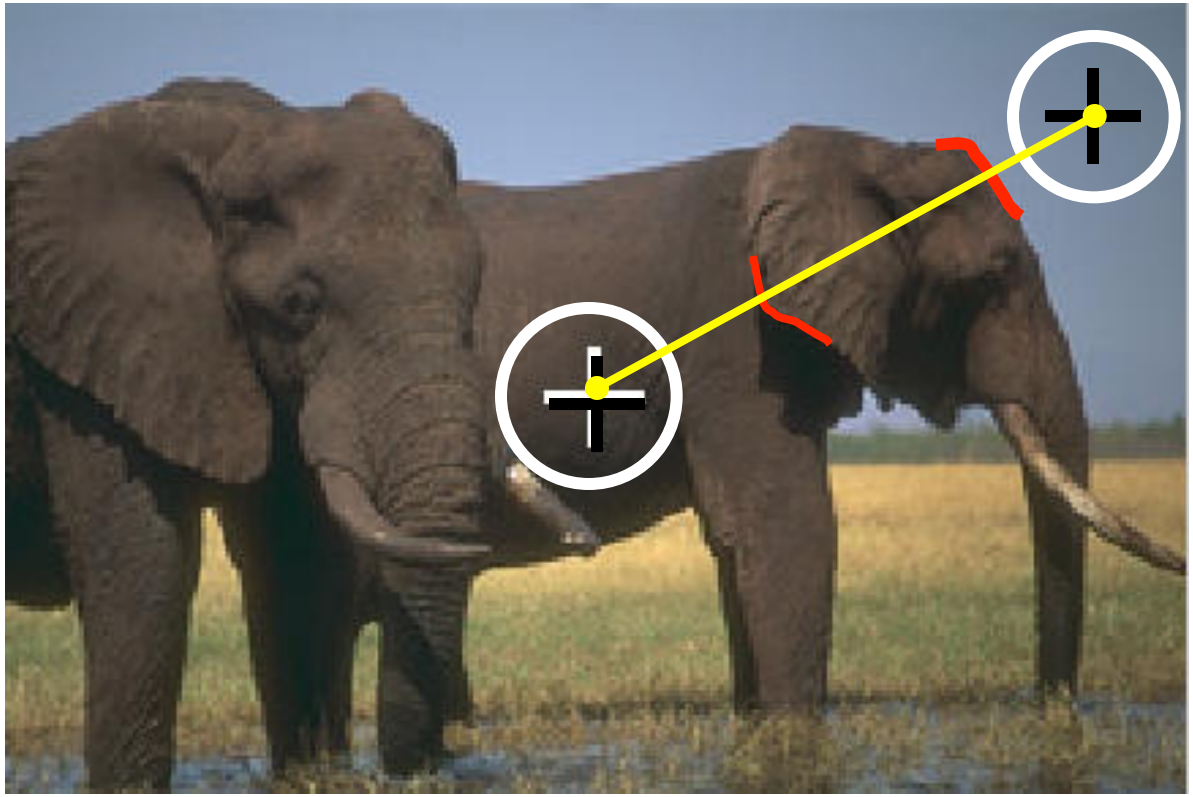
V: image pixels

E: connections between pairs of nearby pixels

Partition graph so that similarity within group is large and similarity between groups is small -- ***Normalized Cuts*** [Shi & Malik 97]

Slide credit: J. Malik, M. Maire

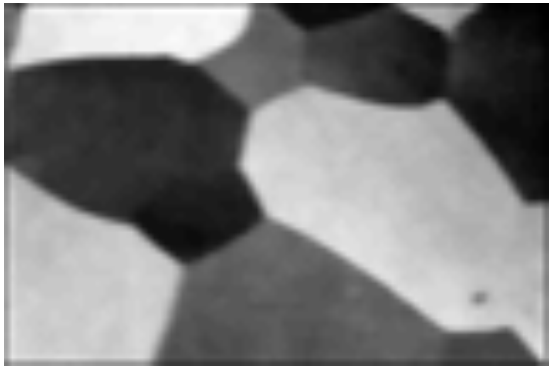
Wij small when intervening contour strong, strong when weak..

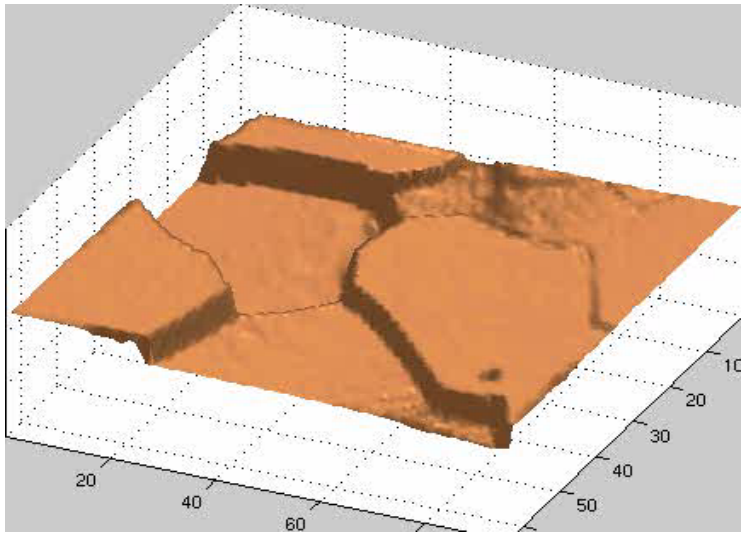


Slide credit: J. Malik, M. Maire

Normalized Cuts as a Spring-Mass system

- Each pixel is a point mass; each connection is a spring:





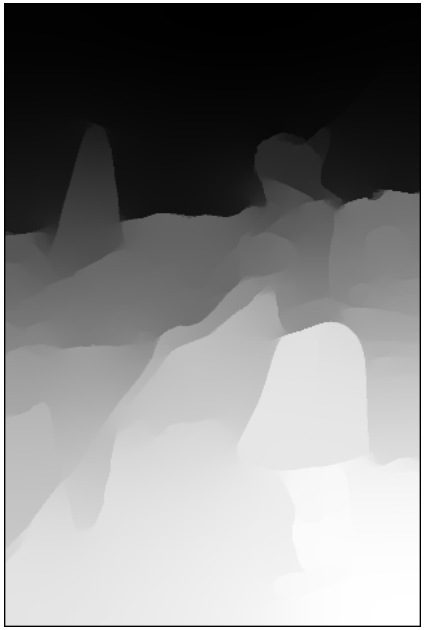
$$(D - W)y = \lambda Dy$$

- Fundamental modes are generalized eigenvectors of

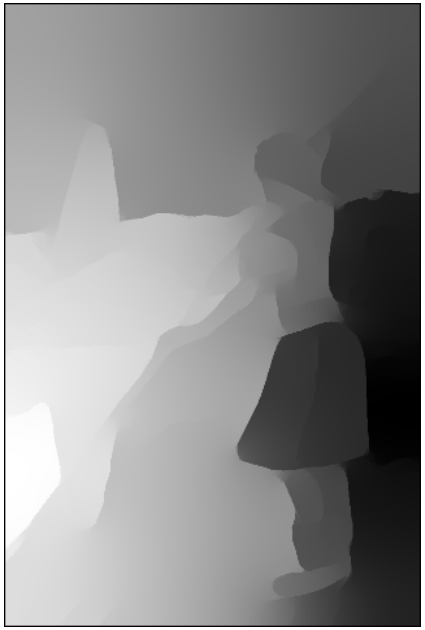
$$(D - W)\, x = \lambda D x$$

Slide credit: J. Malik, M. Maire

Contour information from eigenvectors



(λ_1, W_1)



(λ_8, W_8)

$\max_\theta SPb(x, y, \theta)$

$$SPb(x, y, \theta) = \sum_{k=1}^{8} \frac{1}{\sqrt{\lambda_k}} |G_\theta(x, y) * W_k(x, y)|$$

Slide credit: J. Malik, M. Maire

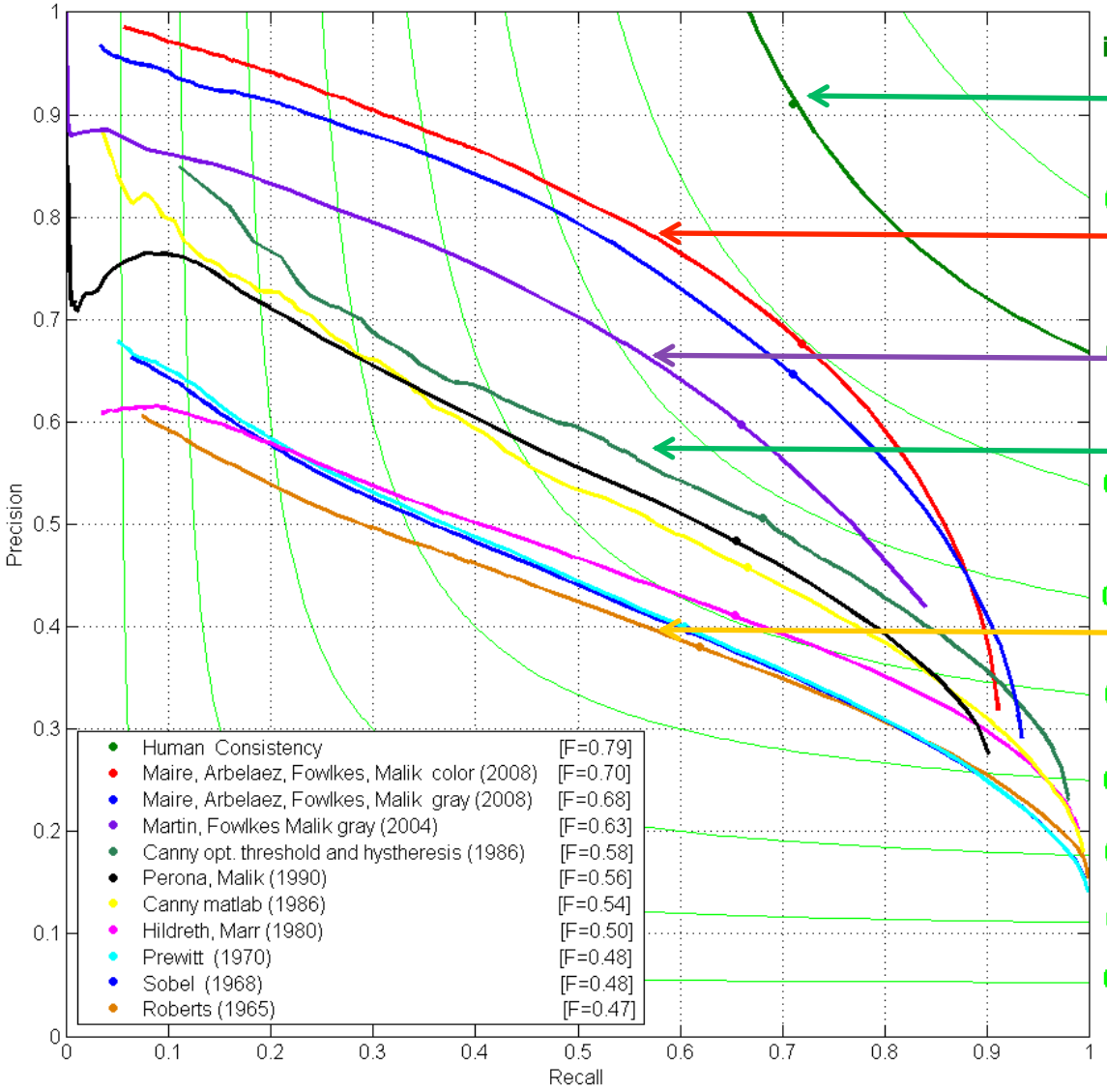
Classifier

Logistic regression

$$P(x, y, \theta) = \frac{1}{1 + \exp(-\sum_{i=1}^{13} w_i f_i(x, y, \theta))}$$

$$P(x, y) = \max_{\theta} P(x, y, \theta)$$

Progress during the last 40 years



Humans

Berkeley gPb, ‘08

Berkeley PB, ‘04

Canny+ Hysteresis (’85)

Prewitt, 1965

1 good feature= 5 years of work

Lecture outline

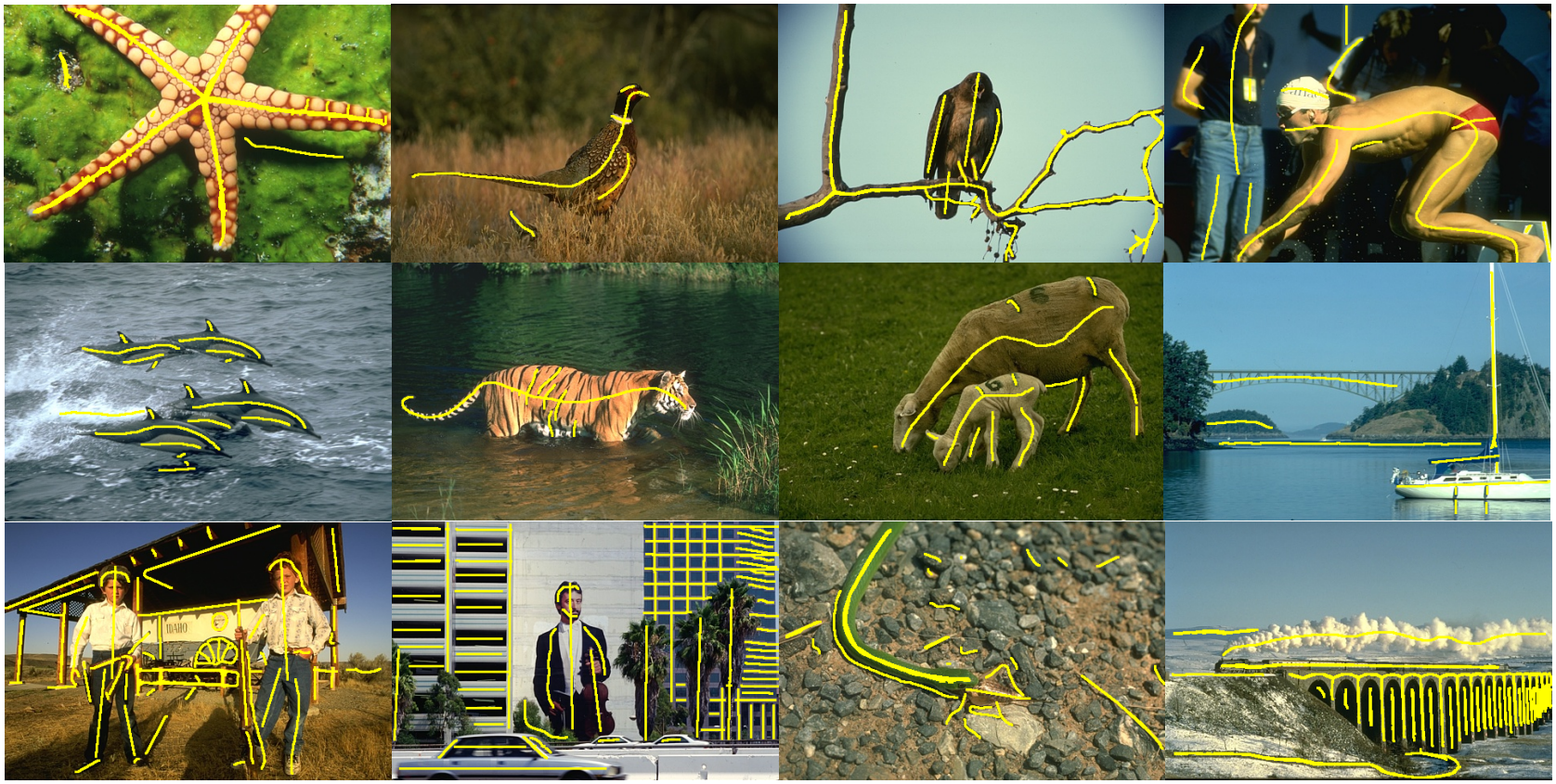
Recap & problems of linear regression

Logistic Regression

- Training criterion formulation
- Optimization
- Interpretation
- Application to boundary detection

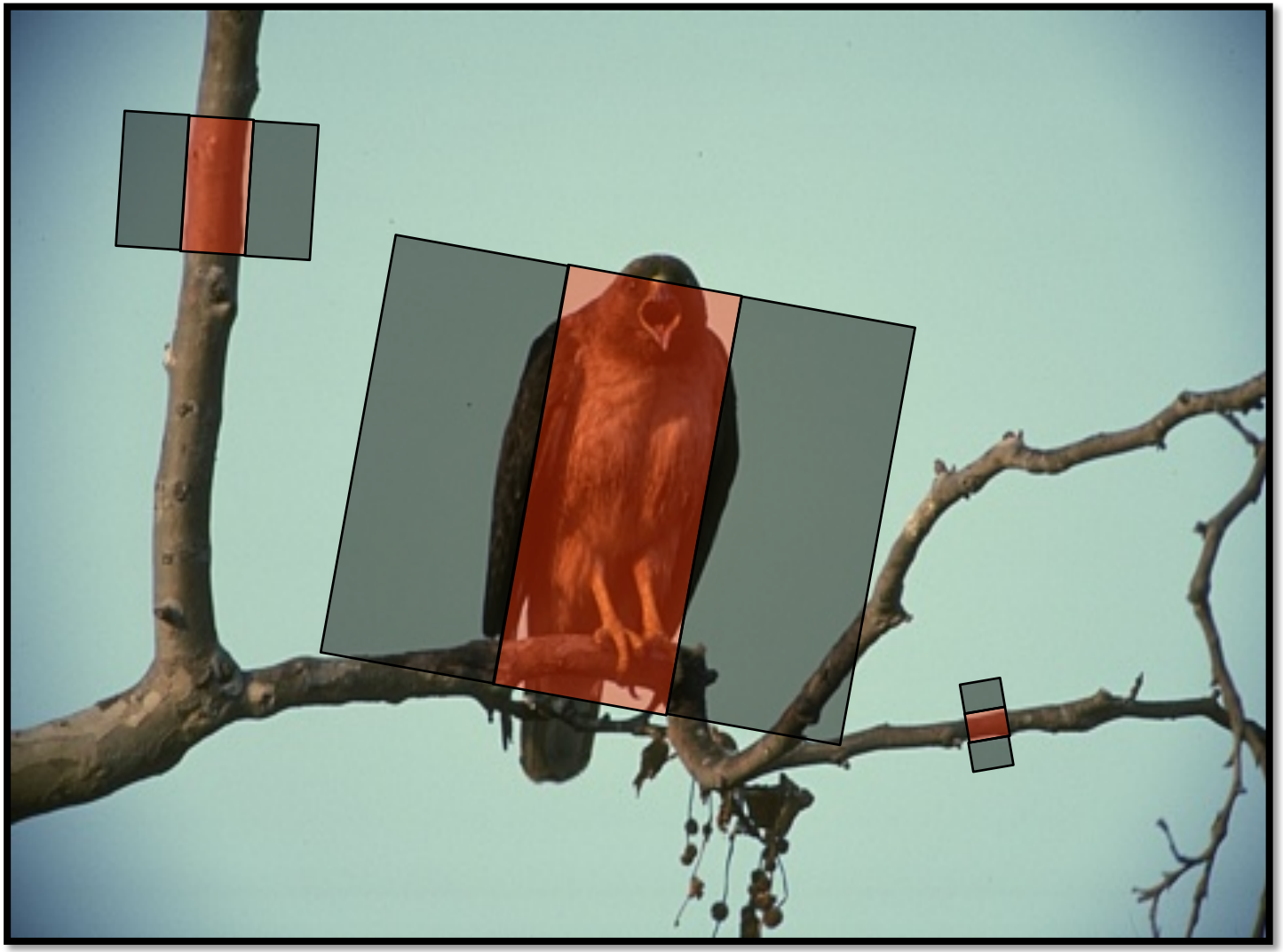
Introduction to Multiple Instance Learning

Learning symmetry detection



S. Tsogkas & I.K., Learning-based symmetry detection in natural images, ECCV 2012

Multi-scale and multi-orientation features



S. Tsogkas & I.K., Learning-based symmetry detection in natural images, ECCV 2012

Learning boundary detection

Problem I: inconsistent orientation information

Problem II: inconsistent location information

Sneaking into the fun room

mom's keychain grandma's keychain dad's keychain

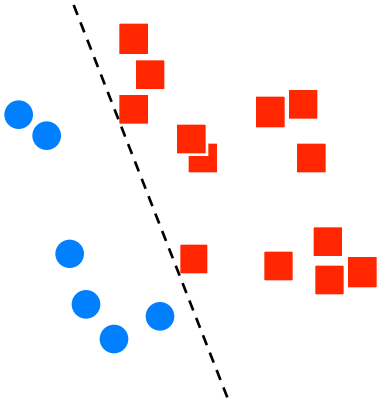


We know that dad cannot enter the fun room, either

Which key should we try?

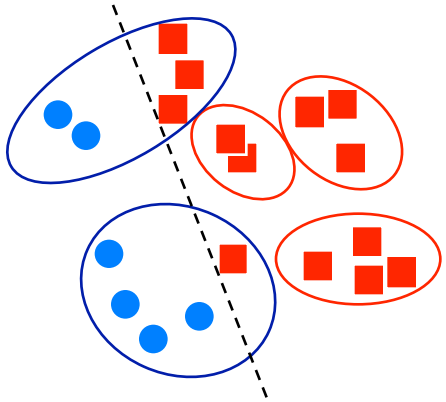
Slide Credit: B. Babenko/T. Dietterich

Multiple Instance Learning



Typical Learning

$$S = \{(x^i, y^i)\}$$



Multiple Instance Learning

$$S = \{(\underbrace{\{x^{i,1}, \ldots, x^{i,|B_i|}\}}_{B_i}, y^i)\}$$

$$y^i \in \{0, 1\} \; x^i \in \mathcal{X}$$

$$F : \mathcal{X} \rightarrow \{0, 1\}$$

Positive bag: at least one instance should be positive

Negative bag: no instance should be positive

$$\max_{x \in B_i} F(x) = y^i$$

Slide Credit: K. Grauman

Noisy-or classifier combination

N classifier responses, $\{p_1, \ldots, p_N\}$

p_i: probability of positive label (classifier i)

$1 - p_i$: probability of negative (classifier i)

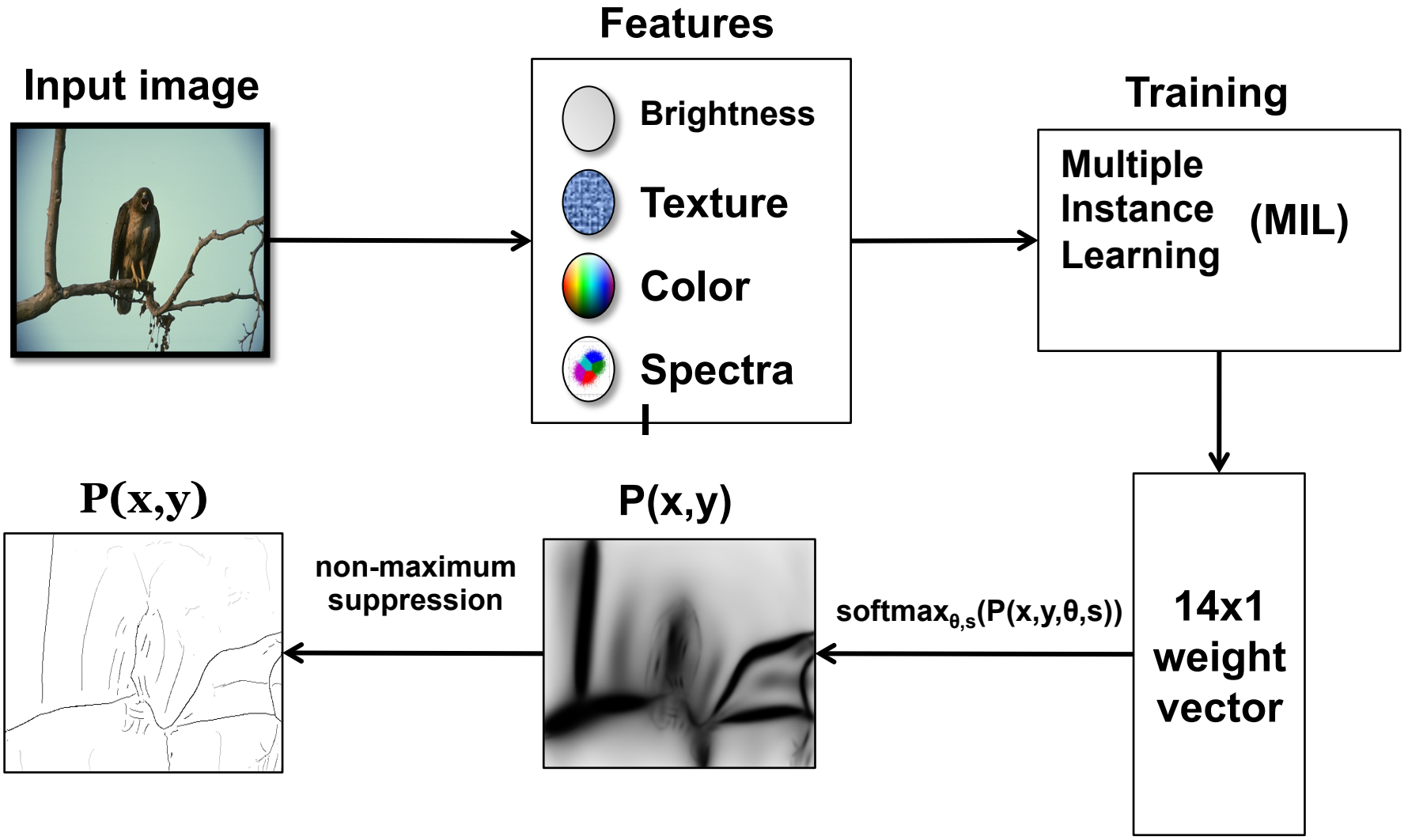
Negative bag: all instance classifiers are negative

$$\prod_{i=1}^{N}(1 - p_i)$$

Probability of bag being positive:

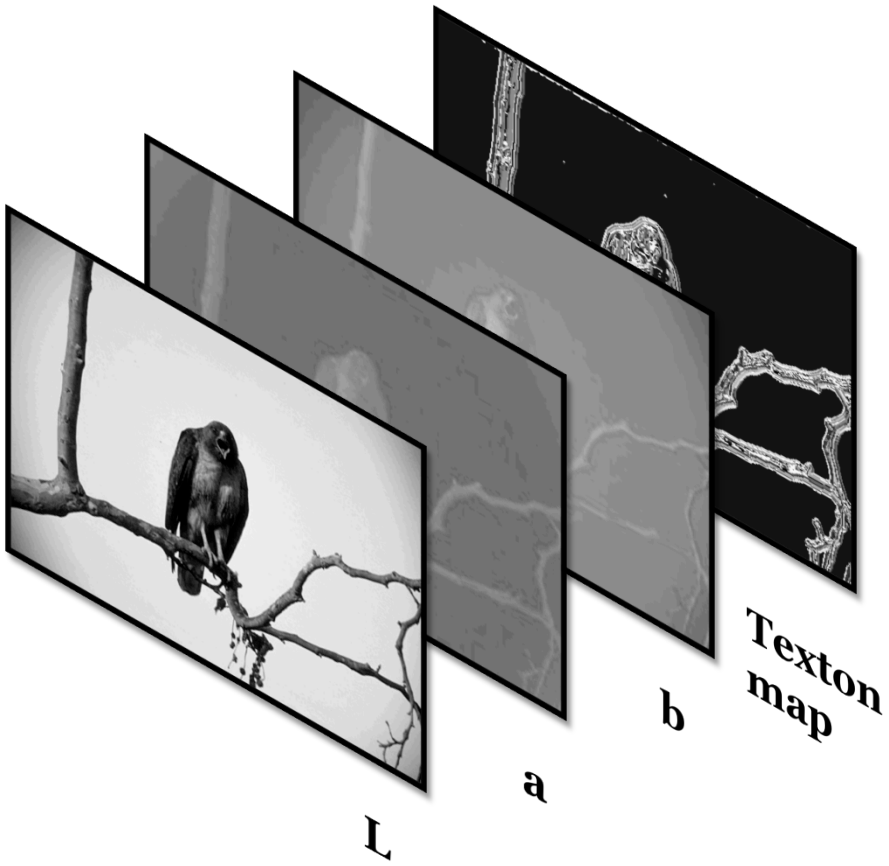
$$p = 1 - \prod_{i=1}^{N}(1 - p_i)$$

Algorithm pipeline



S. Tsogkas & I.K., Learning-based symmetry detection in natural images, ECCV 2012

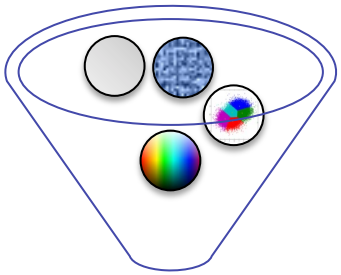
Feature extraction



- Color: CIE Lab color space.
- Texture: texton map.
- Hard binning (32 bins for 3 color channels, 64 textons).
- Rectangle filters extract features at multiple scales and orientations.
- Differences of histograms (“gradients”) of color and texture content → symmetry indication.
- χ^2 – distance → dissimilarity of feature content between adjacent rectangles.
- Integral images for fast extraction.

S. Tsogkas & I.K., Learning-based symmetry detection in natural images, ECCV 2012

MIL Training

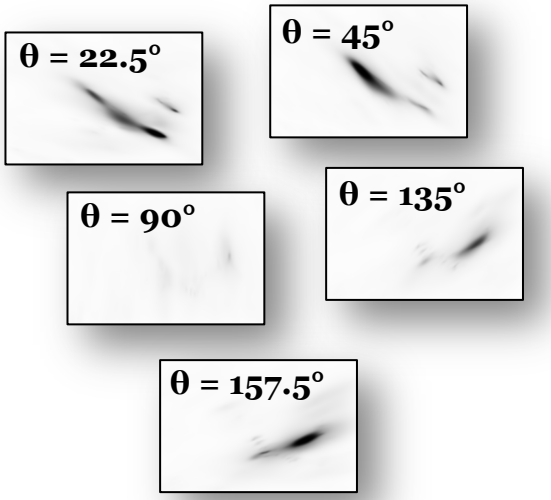


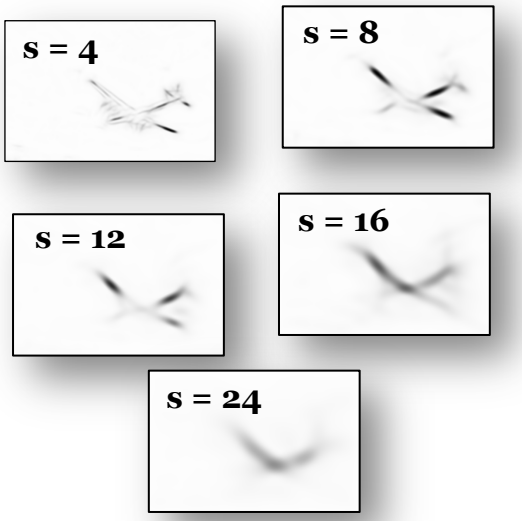
MIL training

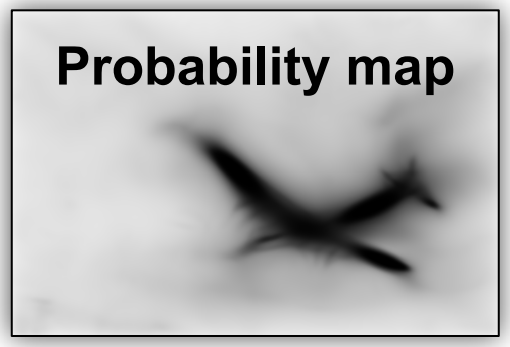
$$\mathbf{w} = \begin{pmatrix} w_1 \\ w_2 \\ \vdots \\ w_{14} \end{pmatrix}$$

Varying angle

Varying scale

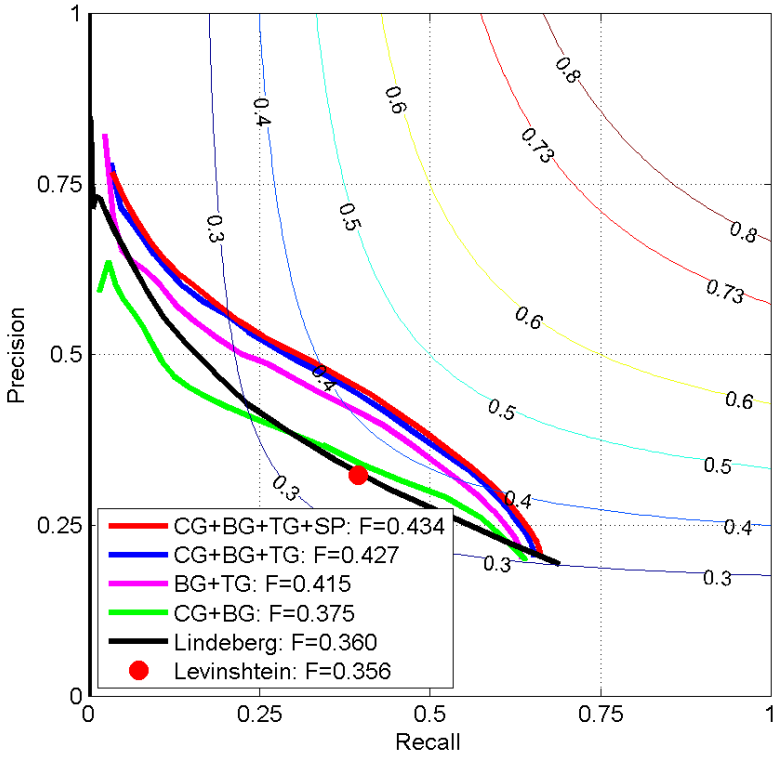






S. Tsogkas & I.K., Learning-based symmetry detection in natural images, ECCV 2012

Results



- State-of-the-art performance
- Large boost from color, texture and spectral cues.
- Long contours (region proposals)

S. Tsogkas & I.K., Learning-based symmetry detection in natural images, ECCV 2012

Some more results…

Lindeberg | Levinshtein | This work | Ground-truth

Lecture recap

- Logistic Regression
  - Training criterion
  - Optimization
  - Application to boundary detection
- Introduction to MIL

Logistic regression training cost

$$L(\mathbf{w}) = -C(\mathbf{w}) = \sum_{i=1}^{N} -\log P(y = y^i | h_{\mathbf{w}}(\mathbf{x}^i))$$

$$P(y = 1|\mathbf{x}, \mathbf{w}) = \frac{1}{1 + \exp(- < \mathbf{x}, \mathbf{w} >)}$$

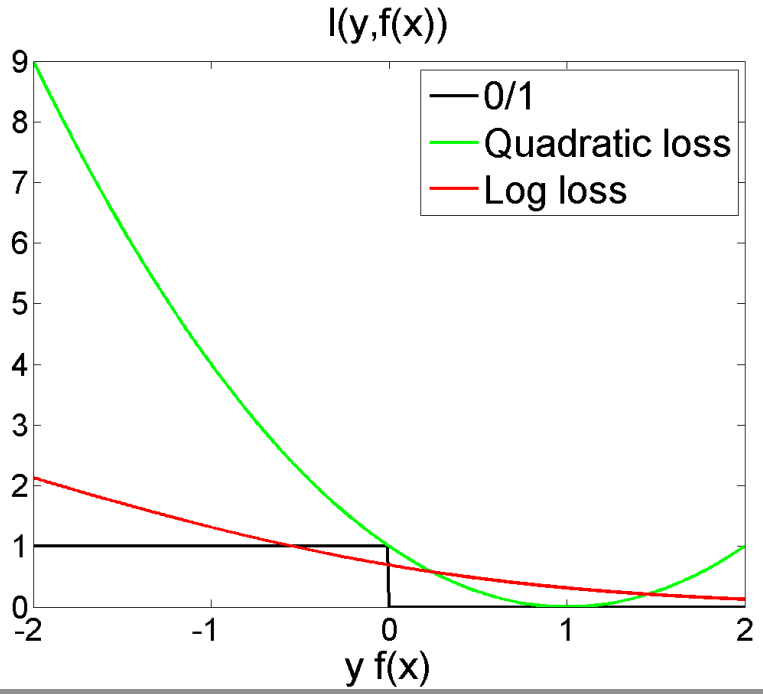
$$= \sum_{i=1}^{N} \underbrace{\log(1 + \exp(-y^i h_{\mathbf{w}}(\mathbf{x}^i)))}_{l(y^i, h_{\mathbf{w}}(\mathbf{x}^i))}$$

Log loss

$$l(y, f(x)) = \log(1 + \exp(-yf(x)))$$

Quadratic loss

$$l(y, f(x)) = (1 - yf(x))^2$$



Lecture recap

- Logistic Regression
  - Training criterion
  - Optimization
  - Application to boundary detection
- Introduction to MIL

Maximization with Newton-Raphson

Newton-Raphson method for finding maxima of N-D function:

$$\mathbf{w}_{t+1} = \mathbf{w}_t - \left[H(\mathbf{w}_t)\right]^{-1} J(\mathbf{w}_t)$$

Condition for maximum:

$$H(\mathbf{w}_t) \prec 0$$

Jacobian: $J(\mathbf{w}) = \mathbf{X}^T \left[\mathbf{y} - g\right]$ $\quad \mathbf{X} = \begin{bmatrix} x^1 \\ \vdots \\ x^N \end{bmatrix} \quad N \times M$

Hessian: $H(\mathbf{w}) = -\mathbf{X}^T \mathbf{R} \mathbf{X}, \quad R_{i,i} = g(\mathbf{x}^i \mathbf{w})(1 - g(\mathbf{x}^i \mathbf{w}))$

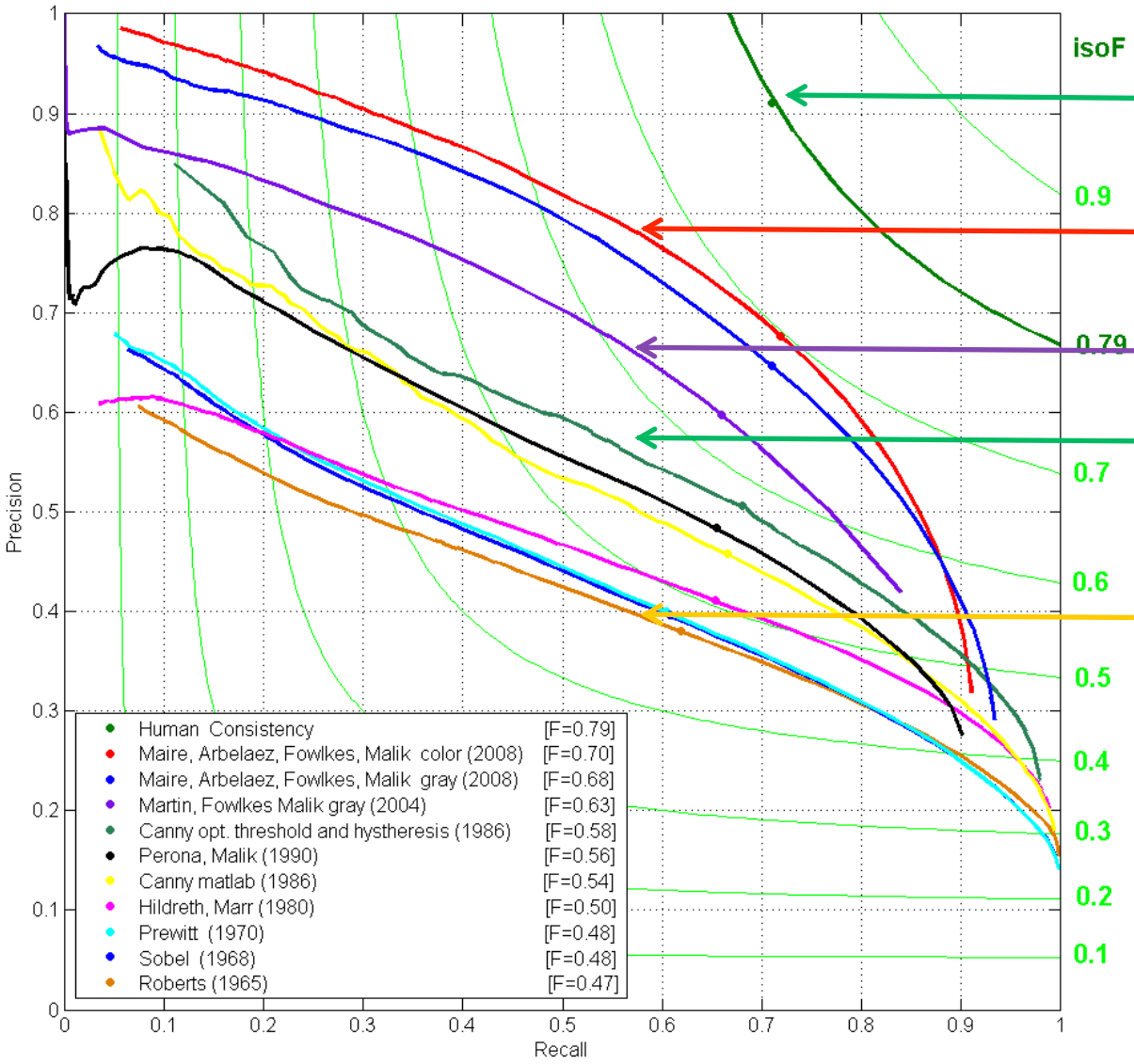
Lecture recap

Logistic Regression

- Training criterion
- Optimization
- Application to boundary detection

Introduction to MIL

Progress during the last 40 years



Humans

Berkeley gPb, ‘08

Berkeley PB, ‘04

Canny+ Hysteresis (’85)

Prewitt, 1965

1 good feature= 5 years of work

Lecture outline

Recap & problems of linear regression

Logistic Regression

- Training criterion formulation
- Optimization
- Interpretation
- Application to boundary detection

Introduction to Multiple Instance Learning

Appendix

Newton Raphson = Iteratively Reweighted Least Squares

Masking problem in multi-class linear regression & softmax

Newton Raphson = Iteratively Reweighted Least Squares

- Newton Raphson: $\mathbf{x} = \mathbf{x_0} - \left[H(\mathbf{x_0})\right]^{-1} J(\mathbf{x_0})$
- Hessian:

$$\begin{aligned} H(\mathbf{w})(k,j) &= \frac{\partial^2 C(\mathbf{w})}{\partial w_k \partial w_j} = \frac{\partial \sum_{i=1}^{N} \left[y^i - g(\mathbf{x}^i\mathbf{w})\right] \mathbf{x}_k^i}{\partial w_j} \\ &= -\sum_{i=1}^{N} \frac{\partial g(\mathbf{x}^i\mathbf{w})}{\partial w_j} \mathbf{x}_k^i = -\sum_{i=1}^{N} g(\mathbf{x}^i\mathbf{w})(1 - g(\mathbf{x}^i\mathbf{w}))\mathbf{x}_j^i\mathbf{x}_k^i \end{aligned}$$

- Rewrite $H(\mathbf{w}) = -\mathbf{X}^T R \mathbf{X}, \quad R_{i,i} = g(\mathbf{x}^i\mathbf{w})(1 - g(\mathbf{x}^i\mathbf{w}))$
- Update:

$$\begin{aligned} \mathbf{w}' &= \mathbf{w}^0 + \left(\mathbf{X}^T\mathbf{R}\mathbf{X}\right)^{-1} \mathbf{X}^T (\mathbf{y} - \mathbf{g}) \\ &= \left(\mathbf{X}^T\mathbf{R}\mathbf{X}\right)^{-1} \left[\left(\mathbf{X}^T\mathbf{R}\mathbf{X}\right) \mathbf{w}^0 + \mathbf{X}^T(\mathbf{y} - \mathbf{g})\right] \\ &= \left(\mathbf{X}^T\mathbf{R}\mathbf{X}\right)^{-1} \mathbf{X}^T\mathbf{R} \left[\mathbf{X}\mathbf{w}^0 + \mathbf{R}^{-1}(\mathbf{y} - \mathbf{g})\right] \\ &= \left(\mathbf{X}^T\mathbf{R}\mathbf{X}\right)^{-1} \mathbf{X}^T\mathbf{R}\mathbf{z} \end{aligned}$$

- Transform: $\mathbf{z}_i = \mathbf{x}^i\mathbf{w}^0 + \frac{1}{g(\mathbf{x}^i\mathbf{w}^0)(1 - g(\mathbf{x}^i\mathbf{w}^0))}(y^i - g(\mathbf{x}^i\mathbf{w}^0))$
- Weighted Least Squares Fit: $\mathbf{w}' = \text{argmin}_{\mathbf{w}} \sum_{i=1}^{N} R_{i,i}(z_i - \mathbf{x}^i\mathbf{w})^2$

Multiple classes & linear regression

K classes: one-of-k coding

4 classes, i-th sample is in 3rd class: $\mathbf{y}^i = (0, 0, 1, 0)$

Matrix notation:

$$\mathbf{Y} = \begin{bmatrix} y^1 \\ \vdots \\ y^N \end{bmatrix} (N \times K) \qquad \mathbf{X} = \begin{bmatrix} x^1 \\ \vdots \\ x^N \end{bmatrix} (N \times M)$$

$$\mathbf{W} = \left[\mathbf{w}^1 | \mathbf{w}^2 | \ldots | \mathbf{w}^K\right] (M \times K)$$

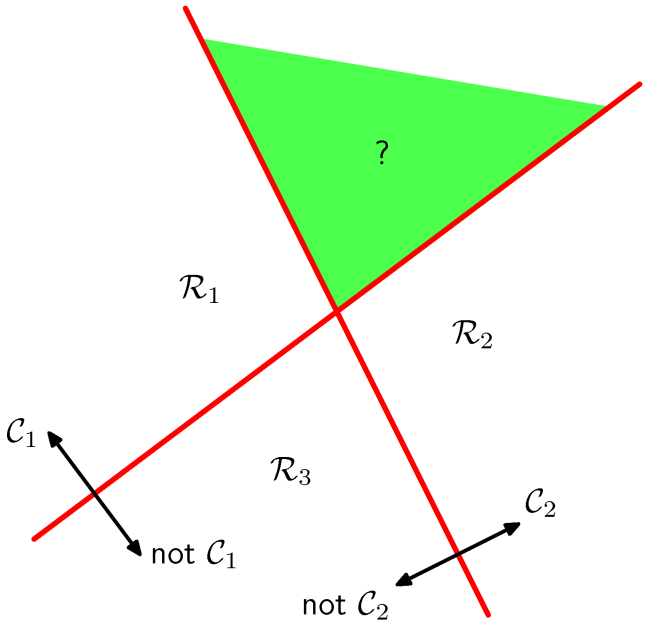
Loss function: $$RSS(\mathbf{W}) = \mathrm{Trace}\left[(\mathbf{Y} - \mathbf{XW})^T (\mathbf{Y} - \mathbf{XW})\right]$$

Least squares fit: $$\hat{\mathbf{W}} = (\mathbf{X^T X})^{-1}(\mathbf{X^T Y})$$

Multiple classes & linear regression

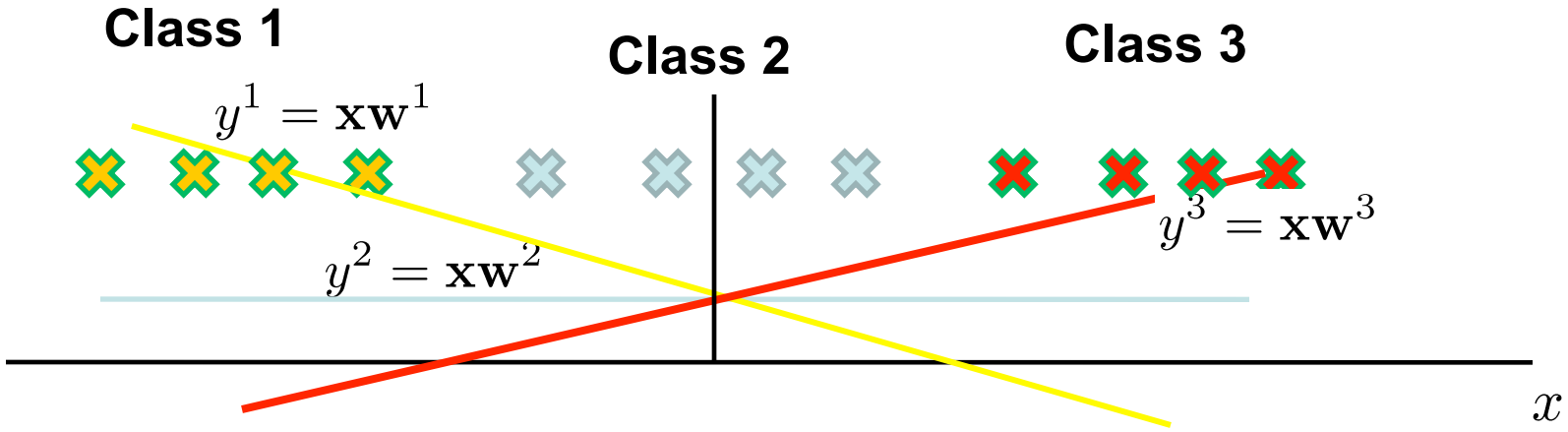
One linear discriminant per class: $y^k = \sum_{i=1}^{M} x_i w_i^k = \mathbf{x}\mathbf{w}^k$

Problem: ambiguous regions



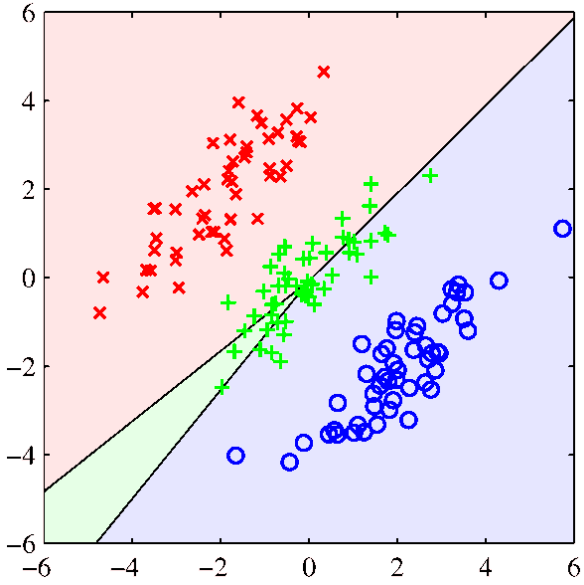
Solution: assign to discriminant with largest score

Masking Problem in linear regression



Nothing ever gets assigned to class 2!

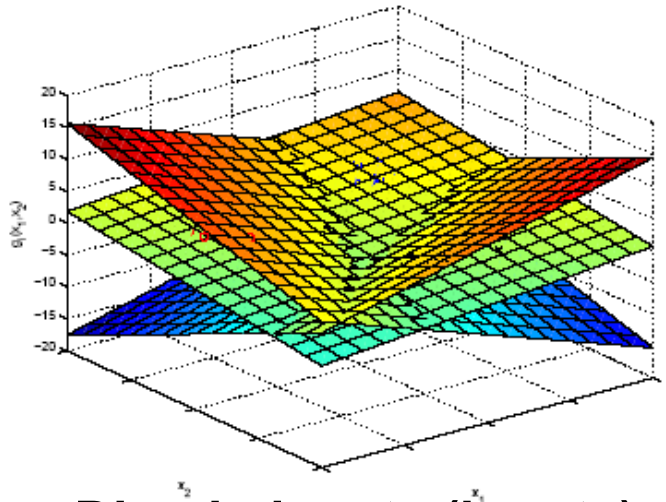
2D version:



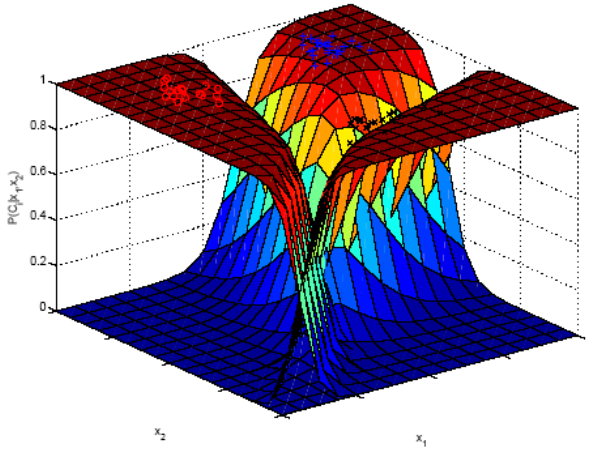
Multiple classes & logistic regression

Soft maximum of competing classes:

$$P(y = k|\mathbf{x}) = \frac{\exp(\mathbf{x}\mathbf{w_k})}{\sum_{n=1}^{K} \exp(\mathbf{x}\mathbf{w_n})},$$



Discriminants (inputs)



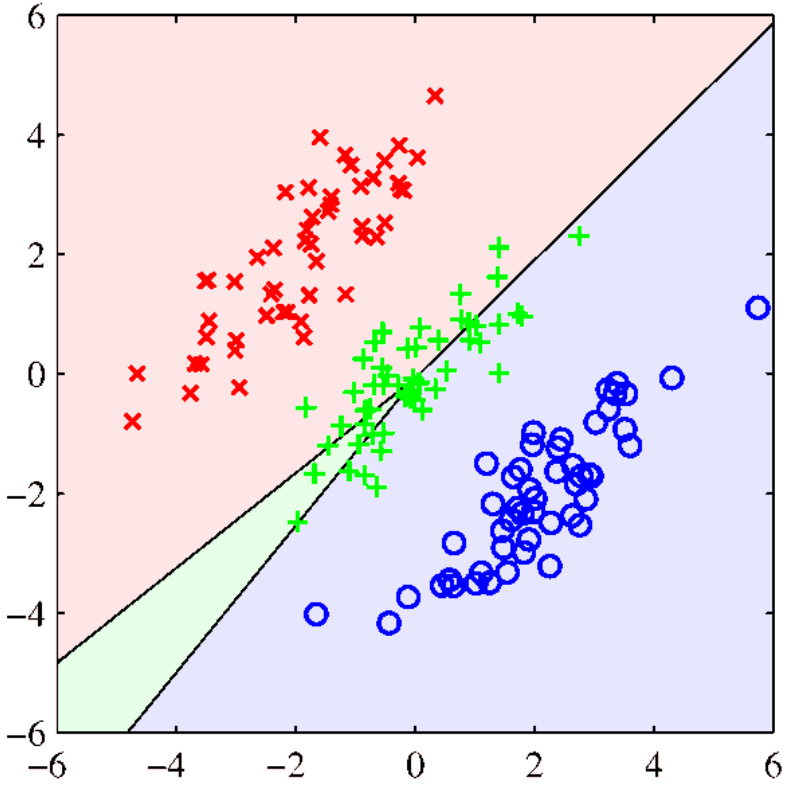
Softmax (outputs)

Label probability: $P(y^i|\mathbf{x}^i, \mathbf{w}) = \prod_{k=1}^{K} g_k(\mathbf{x}\mathbf{w})^{[y^i=k]}$

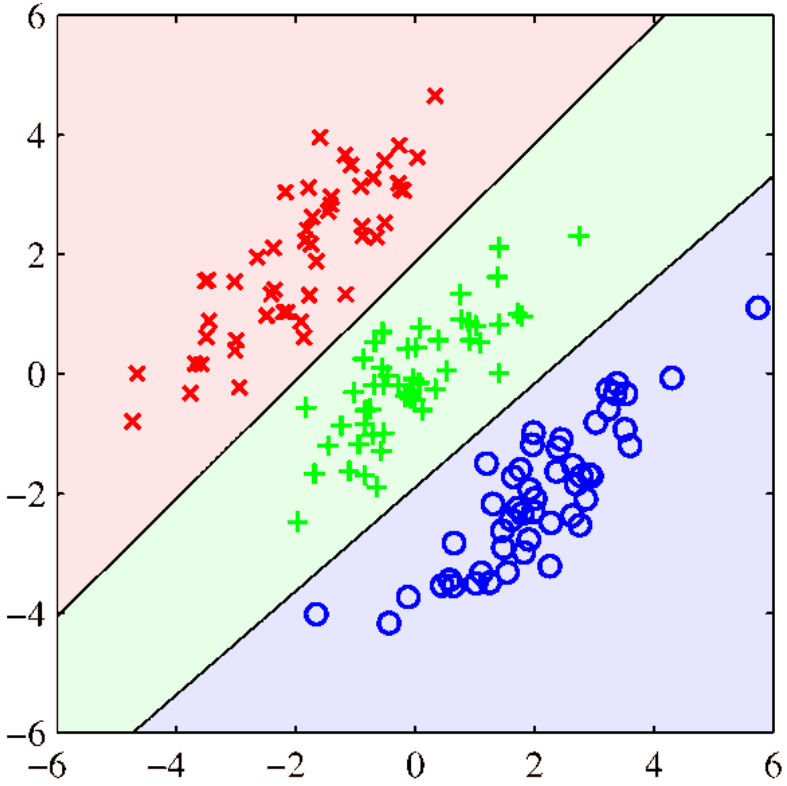
Similar steps for parameter estimation (Bishop's book)

Logistic vs Linear Regression, n>2 classes

Linear regression



Logistic regression



Logistic regression does not exhibit the masking problem