Object Detection

Ali Farhadi

CSE 576

We have talked about

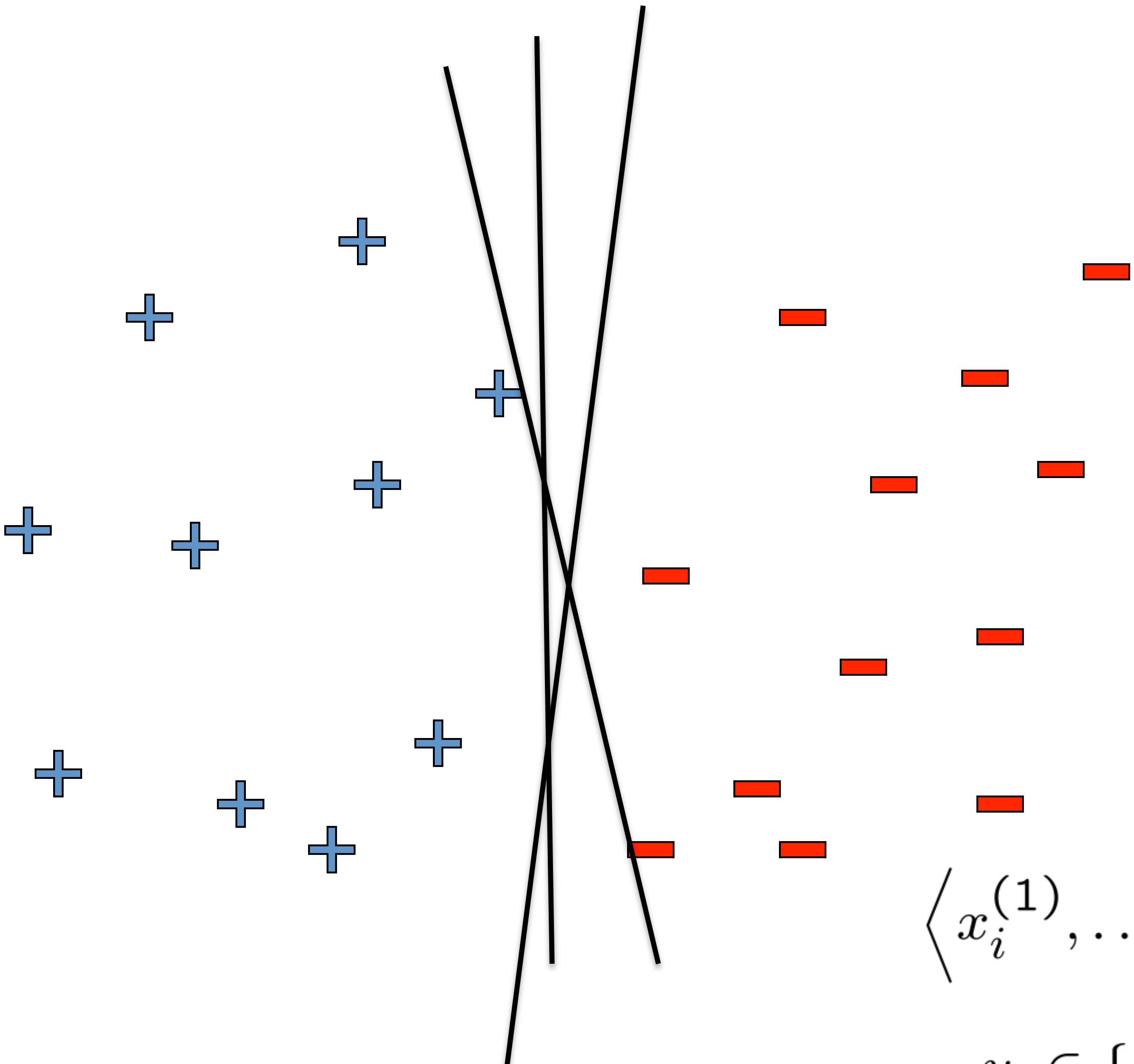
- Nearest Neighbor
- Naïve Bayes
- Logistic Regression
- Boosting

- We saw face detection

Support Vector Machines

Linear classifiers – Which line is better?

$\mathbf{w}. = \sum_j w^{(j)} x^{(j)}$



Data

$$\left\langle x_1^{(1)}, \ldots, x_1^{(m)}, y_1 \right\rangle$$

$$\vdots$$

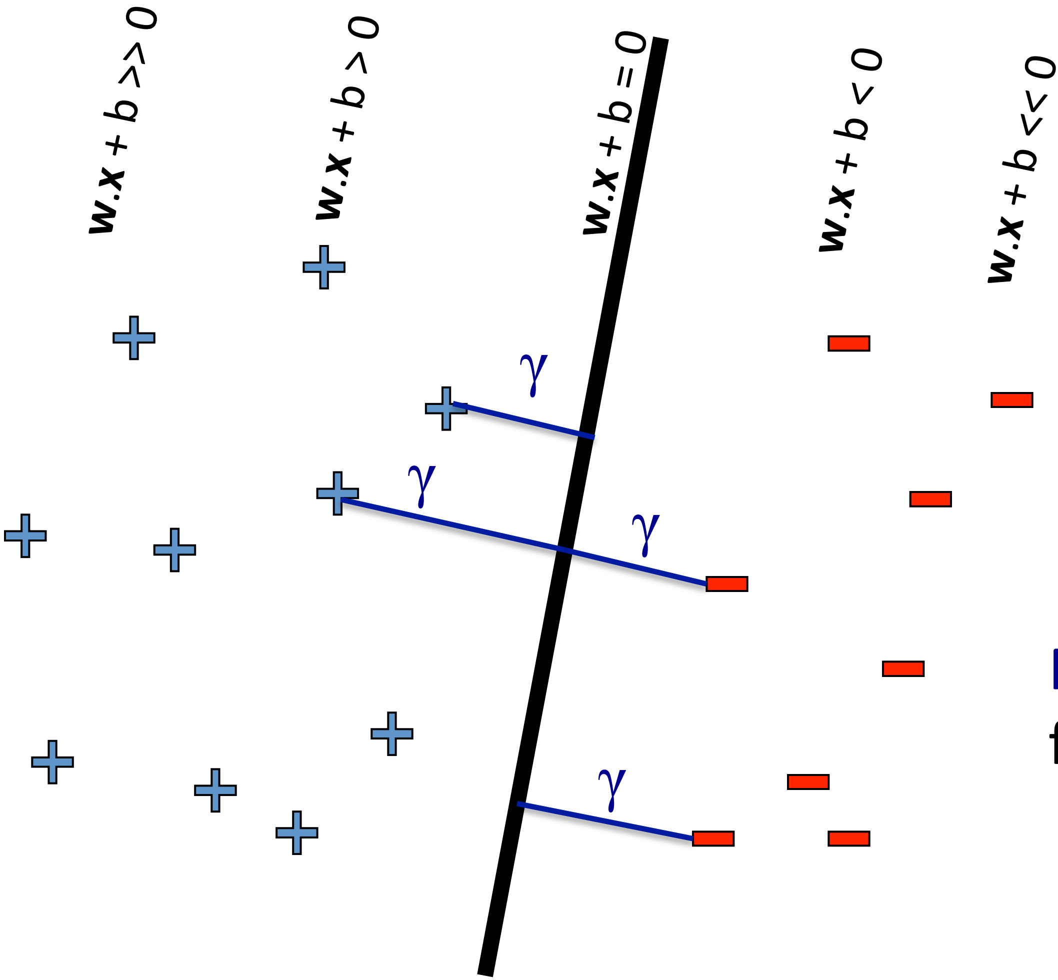
$$\left\langle x_n^{(1)}, \ldots, x_n^{(m)}, y_n \right\rangle$$

Example i

$\left\langle x_i^{(1)}, \ldots, x_i^{(m)} \right\rangle$ — m features

$y_i \in \{-1, +1\}$ — class

Pick the one with the largest margin!



$\mathbf{w.x} = \sum_j w^{(j)} x^{(j)}$

Margin: measures height of w.x+b plane at each point, increases with distance

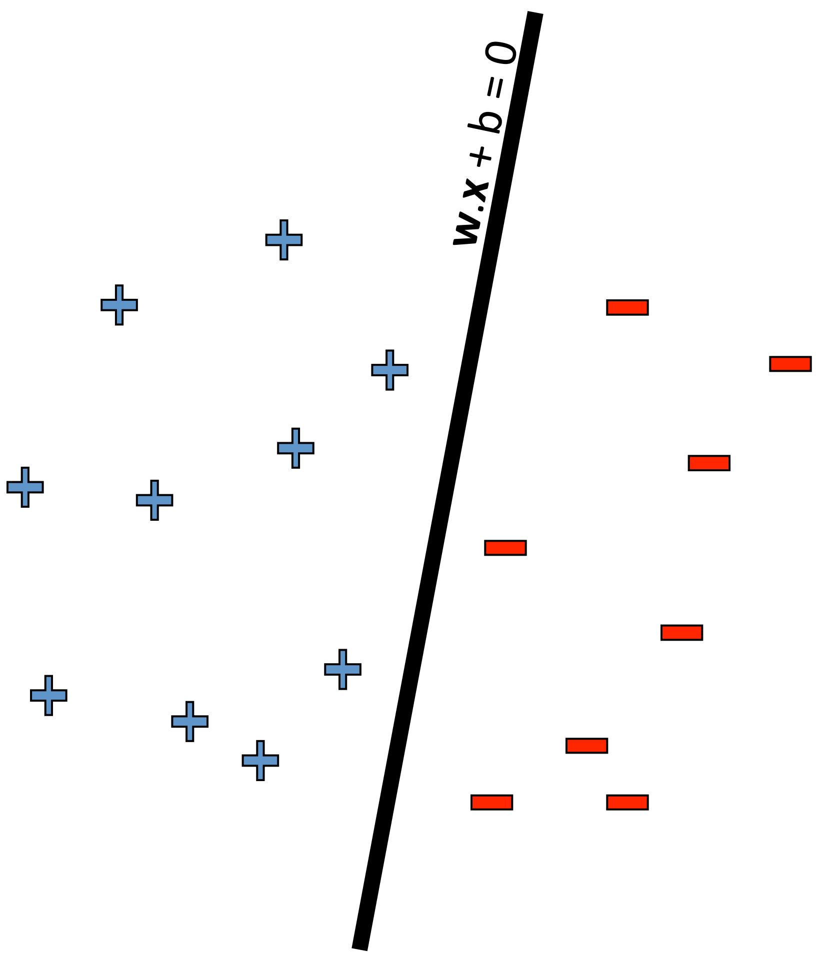
$$\gamma_j = (w.x_j + b)y_j$$

Max Margin: two equivalent forms

(1) $$\max_{w,b} \min_j \gamma_j$$

(2) $$\max_{\gamma,w,b} \gamma$$
$$\forall j \quad (w.x_j + b)y_j > \gamma$$

How many possible solutions?



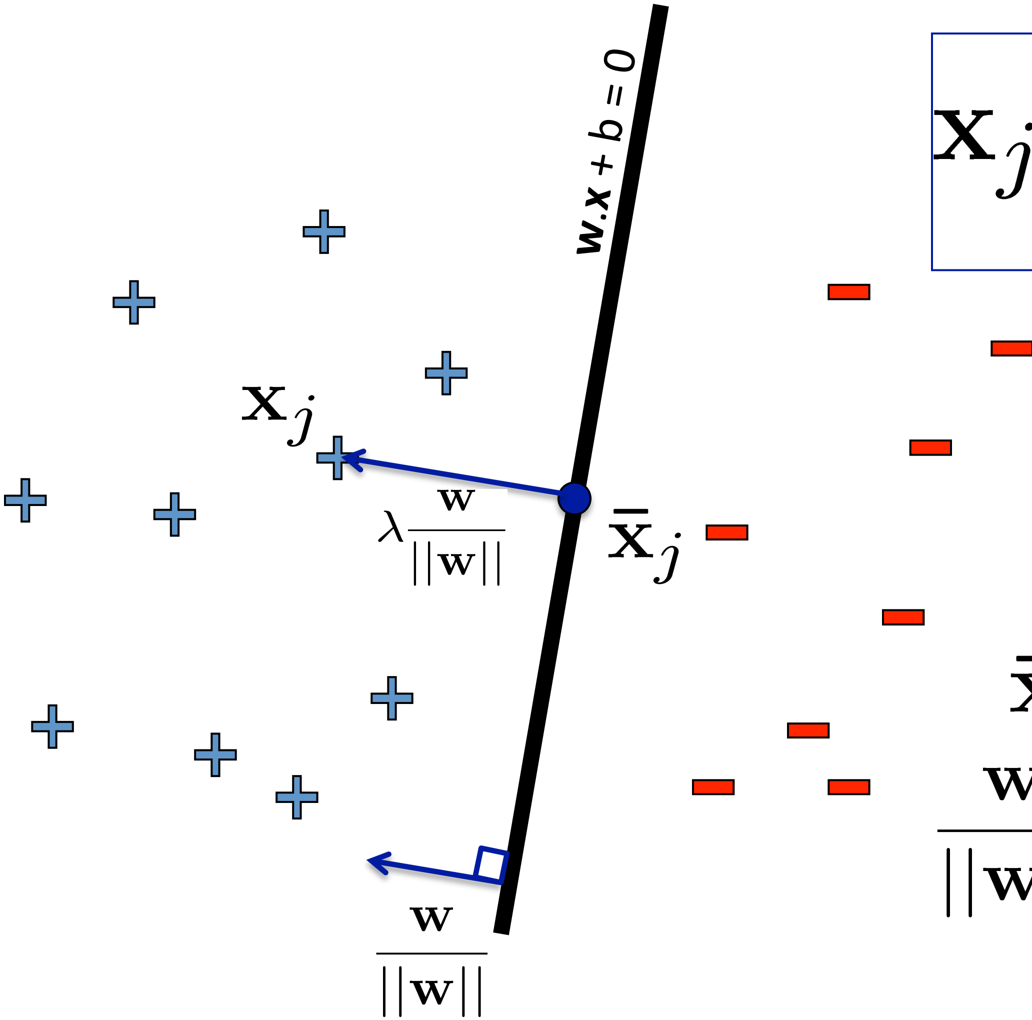
$$\max_{\gamma,w,b} \gamma$$
$$\forall j \quad (w.x_j + b)y_j > \gamma$$

Any other ways of writing the same dividing line?

- **w.x** + b = 0
- 2**w.x** + 2b = 0
- 1000**w.x** + 1000b = 0
- ....
- Any constant scaling has the same intersection with z=0 plane, so same dividing line!

Do we really want to max $_{\gamma,w,b}$?

Review: Normal to a plane



$$\mathbf{x}_j = \bar{\mathbf{x}}_j + \lambda \frac{\mathbf{w}}{||\mathbf{w}||}$$

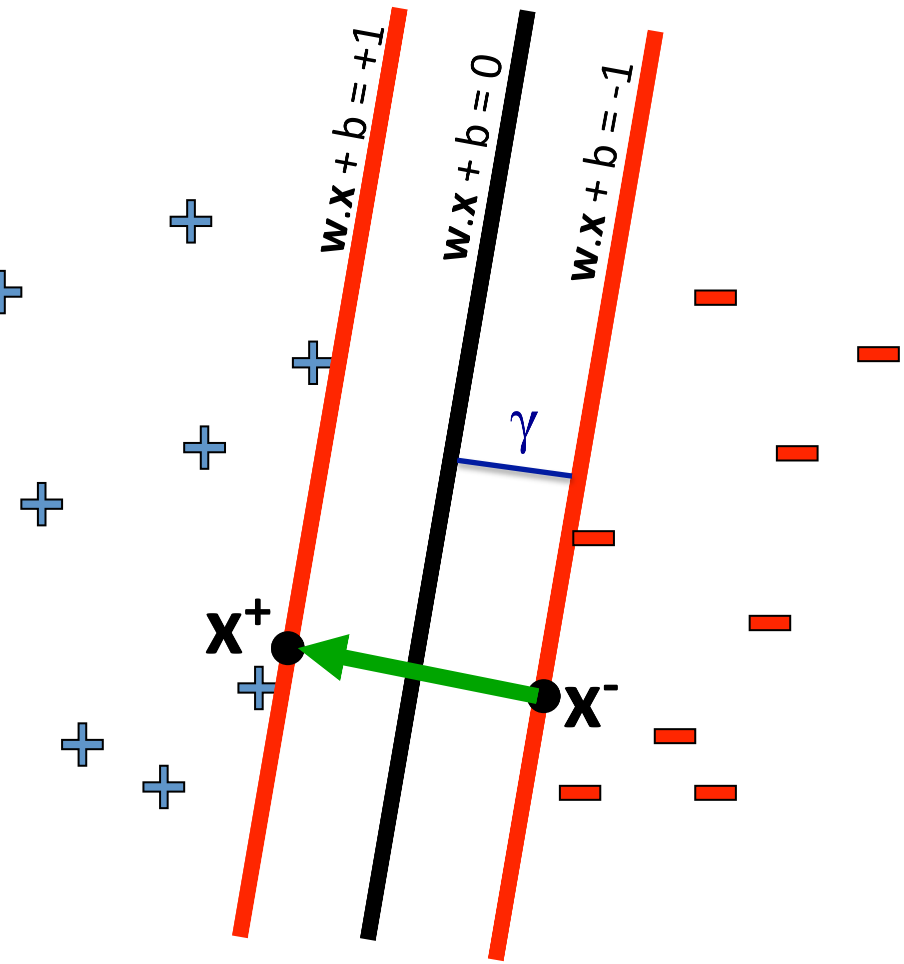
Key Terms

$\bar{\mathbf{x}}_j$ -- projection of x_j onto w

$\frac{\mathbf{w}}{||\mathbf{w}||}$ -- unit vector normal to w

Idea: *constrained* margin

$$\mathbf{x}_j = \bar{\mathbf{x}}_j + \lambda \frac{\mathbf{w}}{||\mathbf{w}||}$$



Generally:

$$x^+ = x^- + 2\gamma \frac{w}{||w||}$$

Assume: x^+ on positive line, x^- on negative

$$w.x^+ + b = 1$$

$$w.\left(x^- + 2\gamma \frac{w}{||w||}\right) + b = 1$$

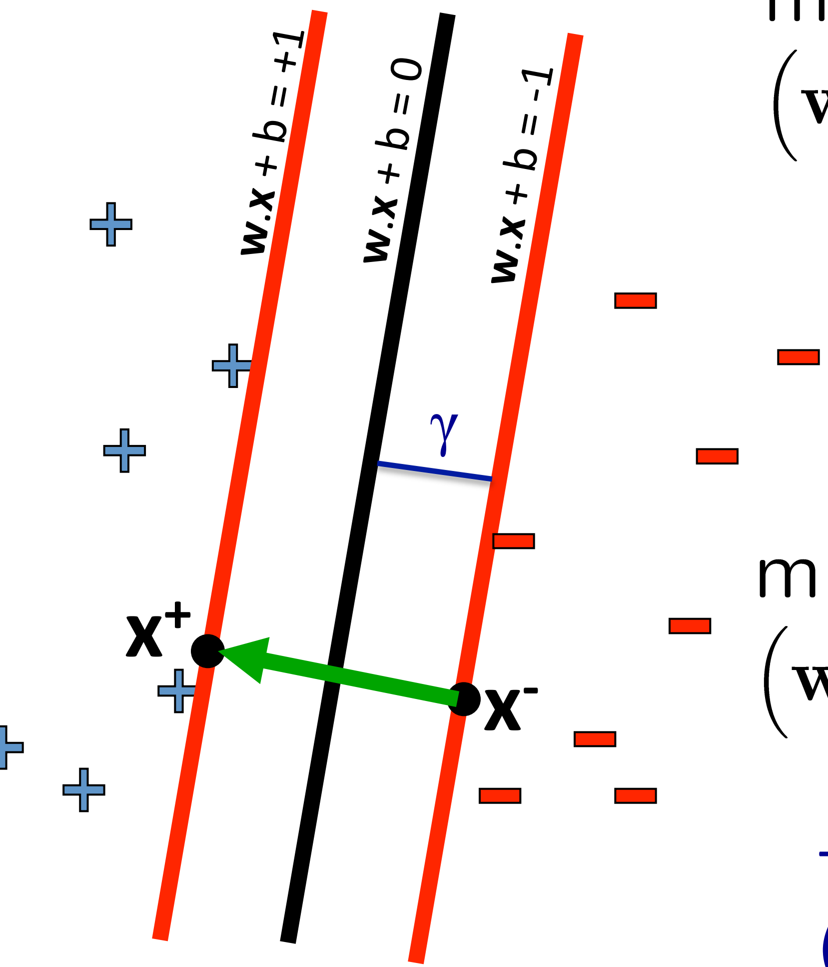
$$w.x^- + b + 2\gamma \frac{w.w}{||w||} = 1$$

$$\gamma \frac{w.w}{||w||} = 1$$

$$\gamma = \frac{||w||}{w.w} = \frac{1}{\sqrt{w.w}}$$

Final result: can maximize constrained margin by minimizing $||w||_2$!!!

Max margin using canonical hyperplanes



$$\text{maximize}_{\gamma,\mathbf{w},b} \quad \gamma$$

$$\left(\mathbf{w}.\mathbf{x}_j + b\right) y_j \geq \gamma, \ \forall j \in \text{Dataset}$$

$$\gamma = \frac{1}{\sqrt{\mathbf{w}.\mathbf{w}}}$$

$$\text{minimize}_{\mathbf{w},b} \quad \mathbf{w}.\mathbf{w}$$

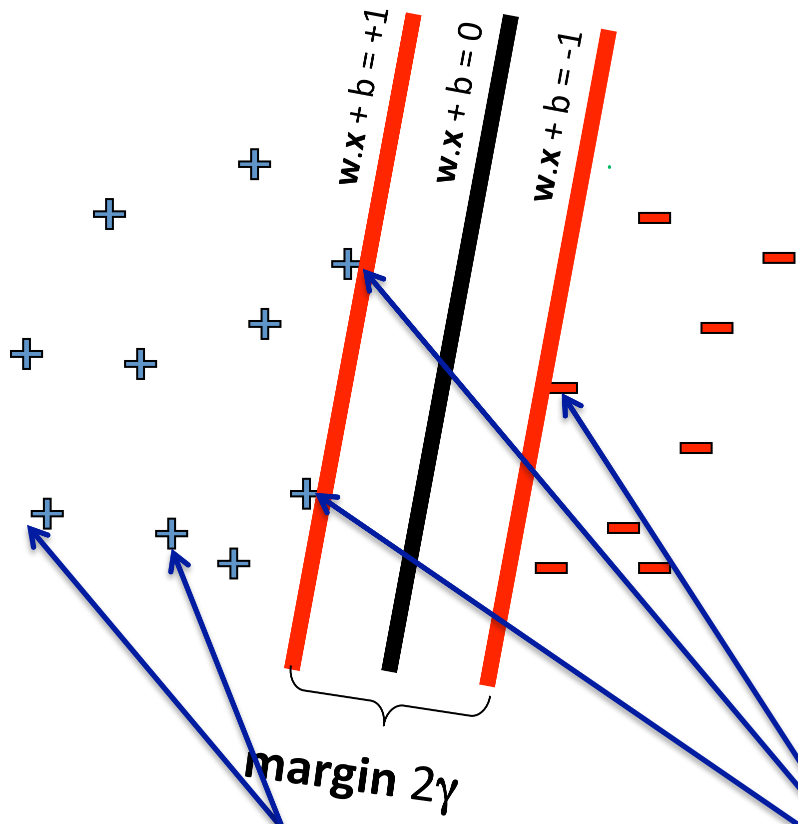
$$\left(\mathbf{w}.\mathbf{x}_j + b\right) y_j \geq 1, \ \forall j \in \text{Dataset}$$

The assumption of canonical hyperplanes (at +1 and -1) changes the objective and the constraints!

Support vector machines (SVMs)

$$\text{minimize}_{\mathbf{w},b} \quad \mathbf{w}.\mathbf{w}$$

$$\left(\mathbf{w}.\mathbf{x}_j + b\right) y_j \geq 1, \ \forall j$$



- Solve efficiently by quadratic programming (QP)
  – Well-studied solution algorithms
  – Not simple gradient ascent, but close
- Hyperplane defined by support vectors
  – Could use them as a lower-dimension basis to write down line, although we haven't seen how yet
  – More on this later

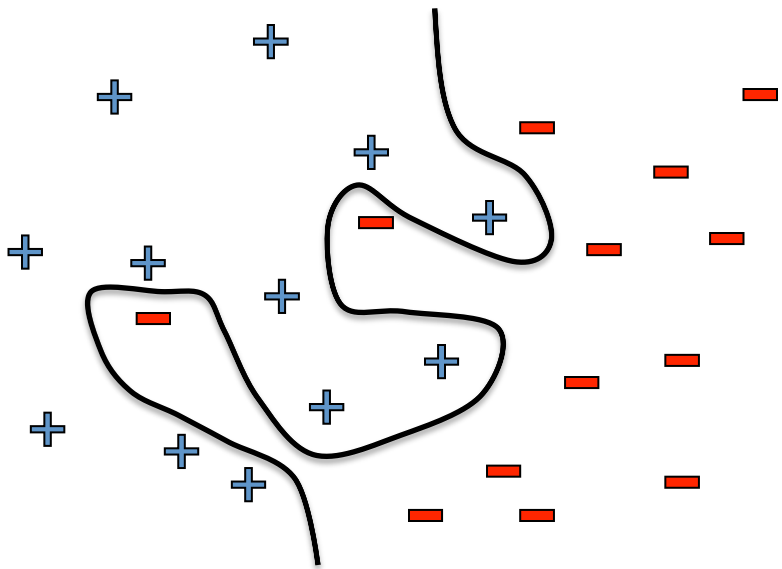
Non-support Vectors:
- everything else
- moving them will not change w

Support Vectors:
- data points on the canonical lines

What if the data is not linearly separable?

$$\left\langle x_i^{(1)}, \ldots, x_i^{(m)} \right\rangle \text{ — } m \text{ features}$$

$$y_i \in \{-1, +1\} \text{ — class}$$



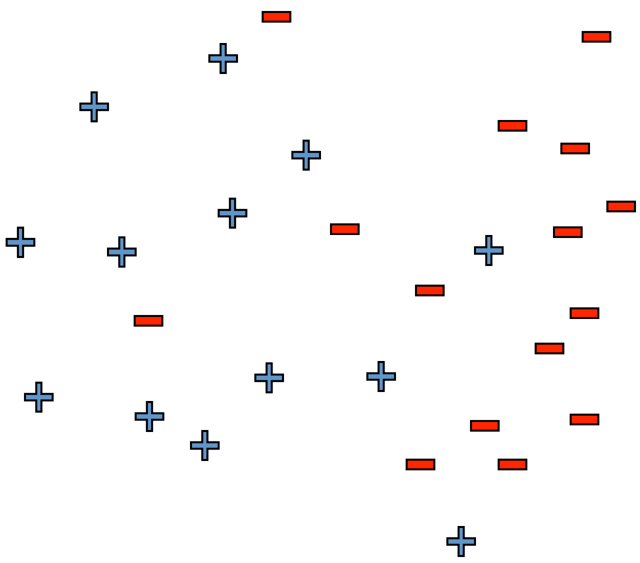
Add More Features!!!

$$\phi(x) = \begin{pmatrix} x^{(1)} \\ \ldots \\ x^{(n)} \\ x^{(1)}x^{(2)} \\ x^{(1)}x^{(3)} \\ \ldots \\ e^{x^{(1)}} \\ \ldots \end{pmatrix}$$

What about overfitting?

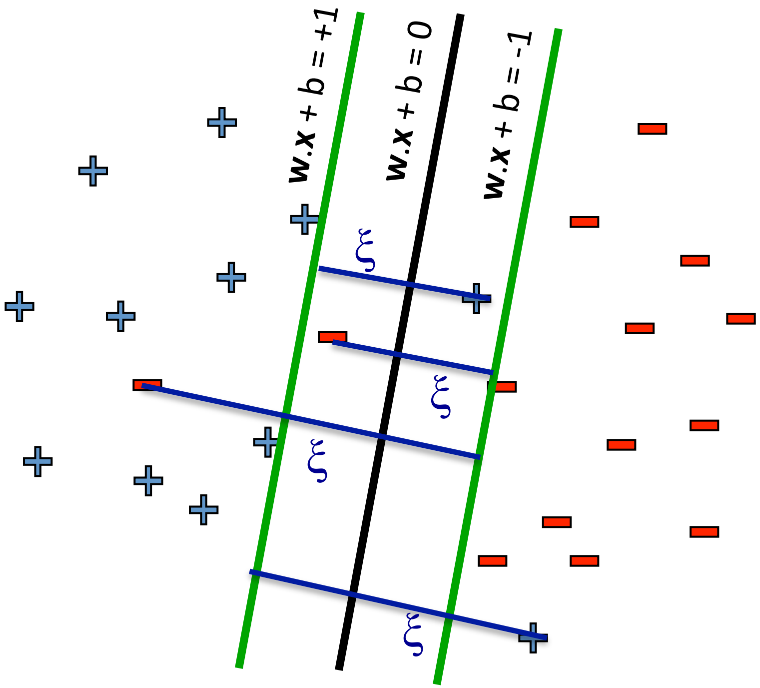
What if the data is still not linearly separable?

$$\text{minimize}_{\mathbf{w},b} \quad \mathbf{w}.\mathbf{w} \; + \text{C \#(mistakes)}$$
$$\left(\mathbf{w}.\mathbf{x}_j + b\right) y_j \geq 1 \qquad , \forall j$$



- First Idea: Jointly minimize **w.w** and number of training mistakes
  - How to tradeoff two criteria?
  - Pick C on development / cross validation
- Tradeoff #(mistakes) and **w.w**
  - 0/1 loss
  - Not QP anymore
  - Also doesn't distinguish near misses and really bad mistakes
  - NP hard to find optimal solution!!!

Slack variables – Hinge loss



$$\text{minimize}_{\mathbf{w},b} \quad \mathbf{w}.\mathbf{w} + C \Sigma_j \xi_j$$

$$\left(\mathbf{w}.\mathbf{x}_j + b\right) y_j \geq 1 - \xi_j \quad , \forall j \quad \xi_j \geq 0$$

For each data point:

- If margin ≥ 1, don't care
- If margin < 1, pay linear penalty

Slack Penalty $C > 0$:

- $C=\infty$ → have to separate the data!
- $C=0$ → ignore data entirely!
- Select on dev. set, etc.

Side Note: Different Losses

Logistic regression:

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

Boosting :

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

SVM:

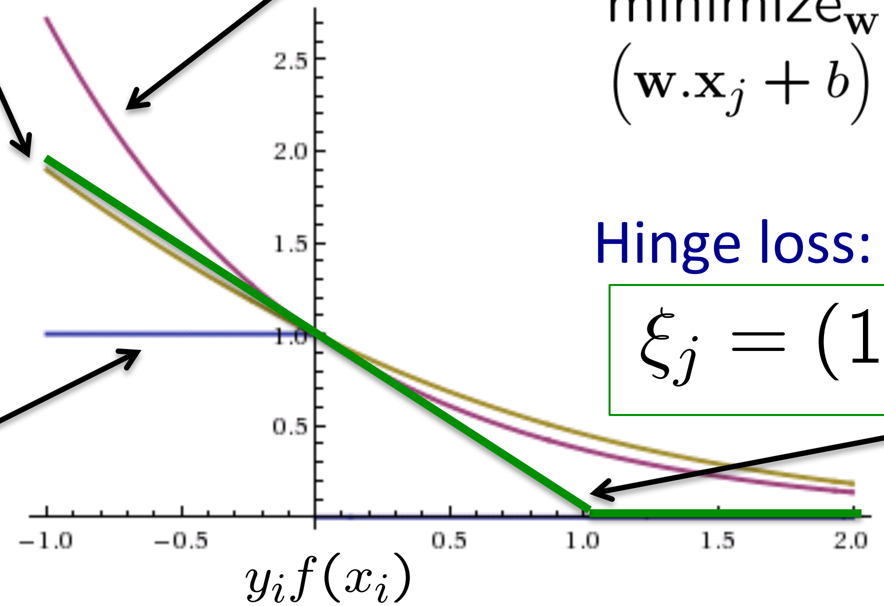
$$\text{minimize}_{\mathbf{w},b} \quad \mathbf{w}.\mathbf{w} + C\sum_j \xi_j$$
$$\left(\mathbf{w}.\mathbf{x}_j + b\right) y_j \geq 1 - \xi_j, \ \forall j$$
$$\xi_j \geq 0, \ \forall j$$

Hinge loss:

$$\xi_j = (1 - f(x_i)y_i)_+$$

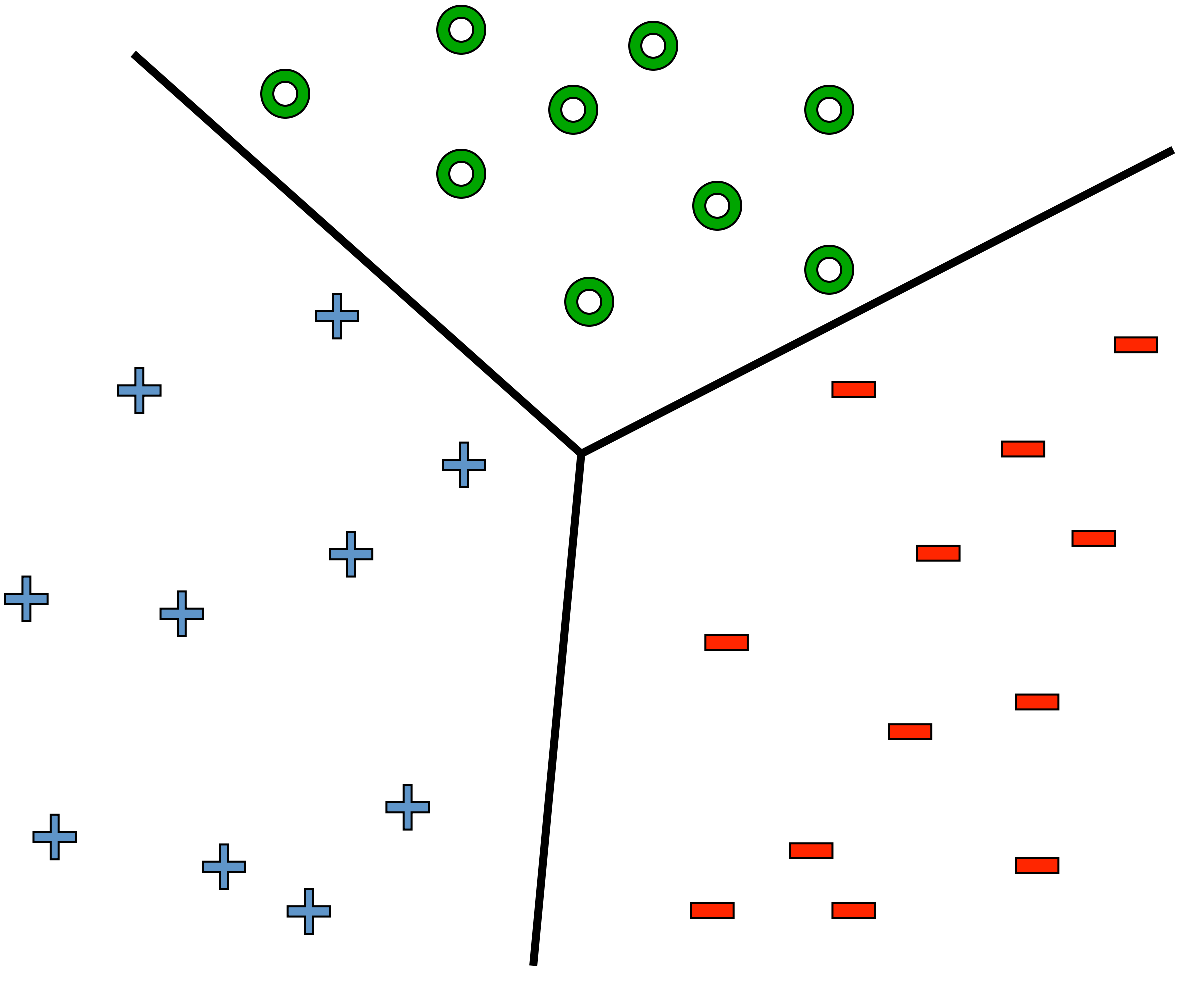
0-1 Loss:

$$\delta(H(x_i) \neq y_i)$$

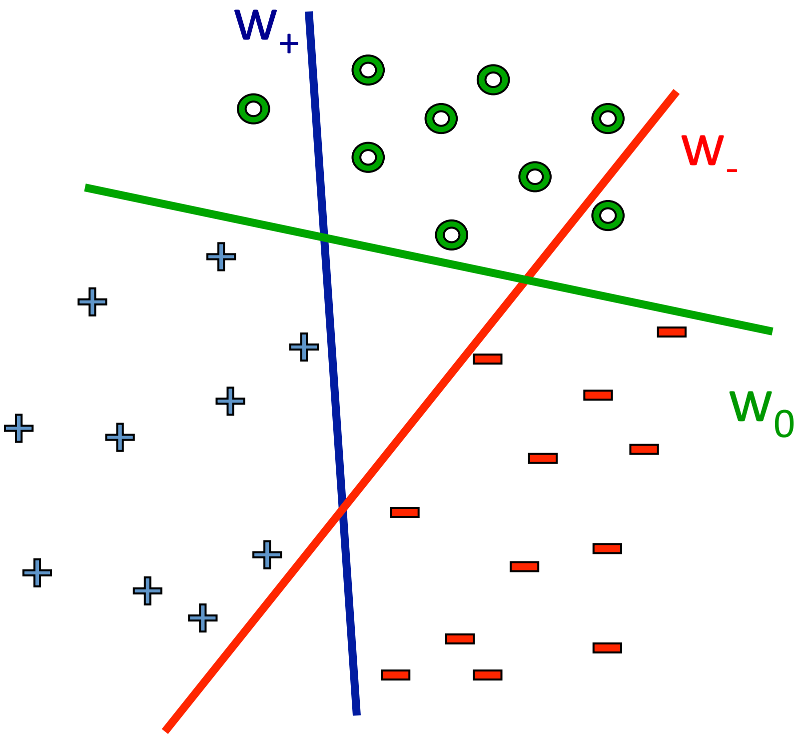


All our new losses approximate 0/1 loss!

What about multiple classes?



One against All



Learn 3 classifiers:

- + vs {0,-}, weights w_+
- - vs {0,+}, weights w_-
- 0 vs {+,-}, weights w_0

Output for x:

$y = \text{argmax}_i \, w_i.x$

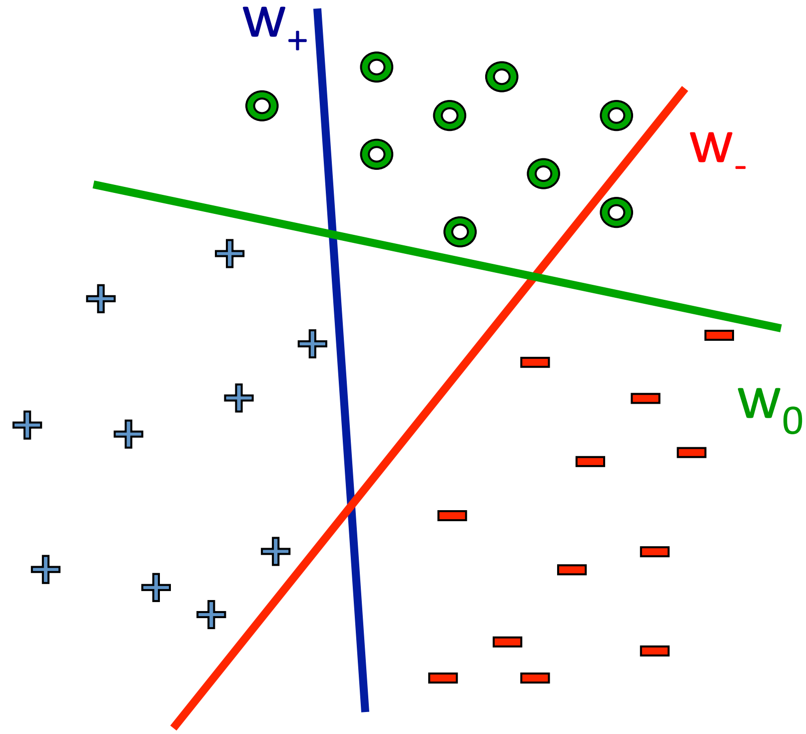
Any other way?

Any problems?

Learn 1 classifier: Multiclass SVM

Simultaneously learn 3 sets of weights:

- How do we guarantee the correct labels?
- Need new constraints!



For j possible classes:

$$\mathbf{w}^{(y_j)}.\mathbf{x}_j + b^{(y_j)} \geq \mathbf{w}^{(y')}.\mathbf{x}_j + b^{(y')} + 1, \ \forall y' \neq y_j, \ \forall j$$

Learn 1 classifier: Multiclass SVM

Also, can introduce slack variables, as before:

$$\text{minimize}_{\mathbf{w},b} \quad \sum_y \mathbf{w}^{(y)}.\mathbf{w}^{(y)} + C\sum_j \xi_j$$

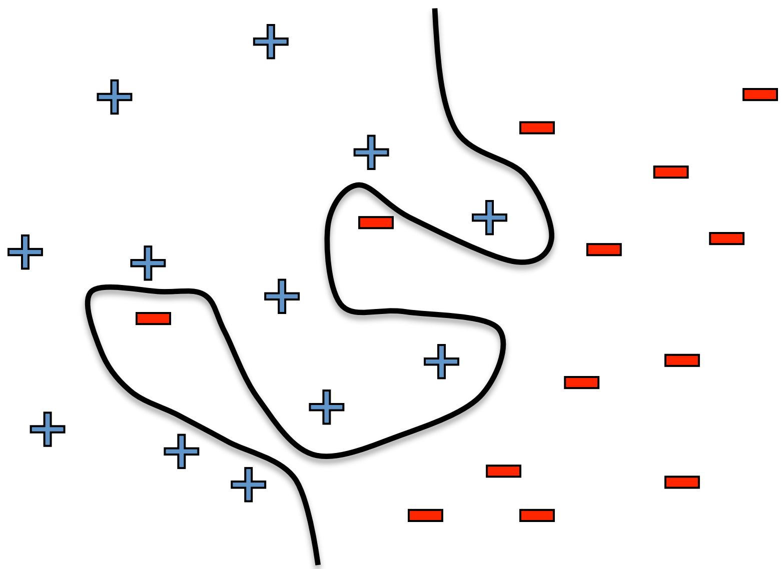
$$\mathbf{w}^{(y_j)}.\mathbf{x}_j + b^{(y_j)} \geq \mathbf{w}^{(y')}.\mathbf{x}_j + b^{(y')} + 1 - \xi_j, \;\; \forall y' \neq y_j, \;\; \forall j$$

$$\xi_j \geq 0, \;\; \forall j$$

What if the data is not linearly separable?

$$\left\langle x_i^{(1)}, \ldots, x_i^{(m)} \right\rangle \text{ — } m \text{ features}$$

$$y_i \in \{-1, +1\} \text{ — class}$$



Add More Features!!!

$$\phi(x) = \begin{pmatrix} x^{(1)} \\ \ldots \\ x^{(n)} \\ x^{(1)} x^{(2)} \\ x^{(1)} x^{(3)} \\ \ldots \\ e^{x^{(1)}} \\ \ldots \end{pmatrix}$$

SVM with a polynomial Kernel visualization

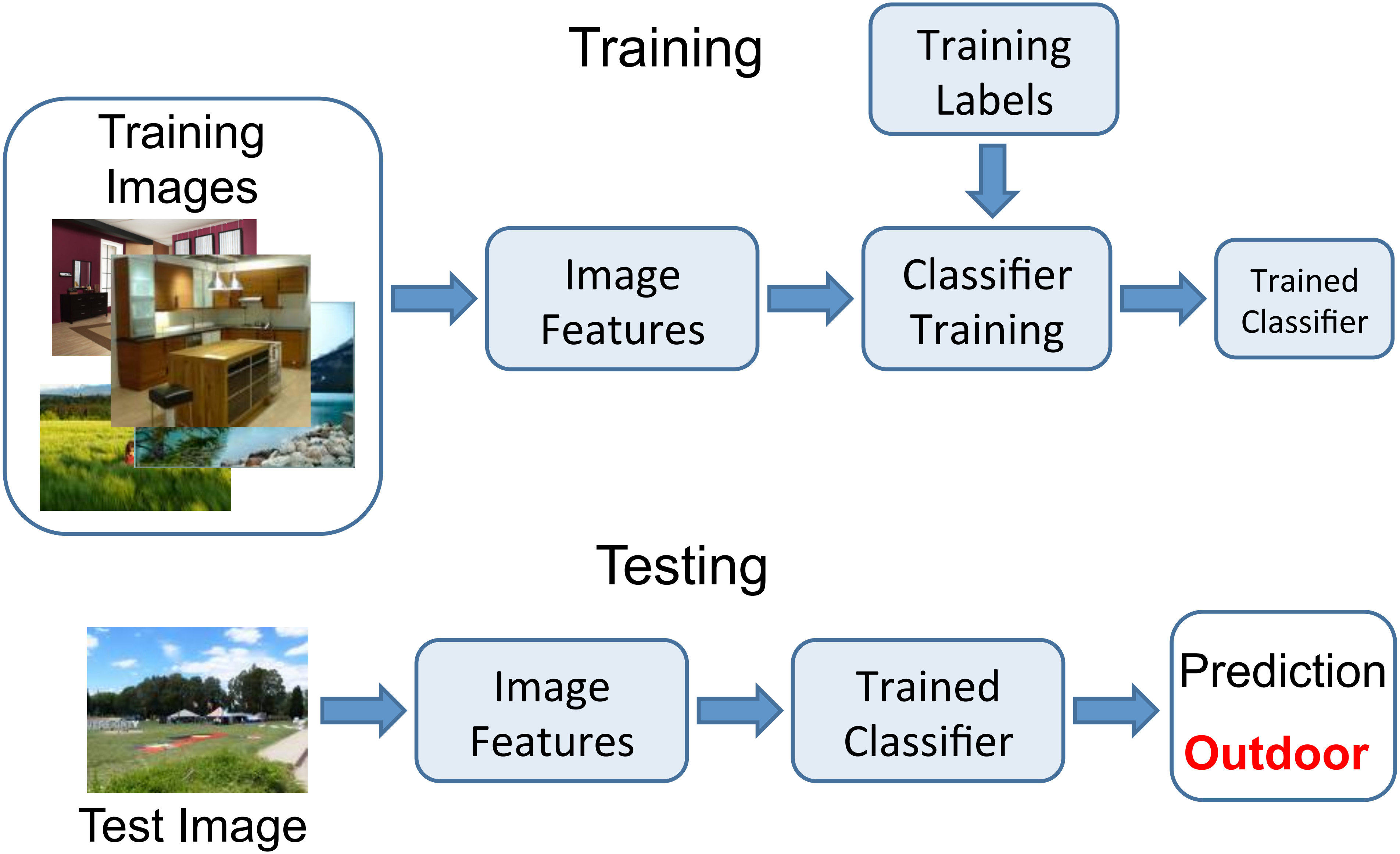
Created by:
Udi Aharoni

Comparison

assuming x in {0 1}

	Learning Objective	Training	Inference
Naïve Bayes	$\text{maximize} \sum_i \left[\sum_j \log P(x_{ij} \mid y_i; \theta_j) + \log P(y_i; \theta_0) \right]$	$\theta_{kj} = \frac{\sum_i \delta(x_{ij} = 1 \wedge y_i = k) + r}{\sum_i \delta(y_i = k) + Kr}$	$\boldsymbol{\theta}_1^T \mathbf{x} + \boldsymbol{\theta}_0^T (1 - \mathbf{x}) > 0$ where $\theta_{1j} = \log \frac{P(x_j = 1 \mid y = 1)}{P(x_j = 1 \mid y = 0)}$, $\theta_{0j} = \log \frac{P(x_j = 0 \mid y = 1)}{P(x_j = 0 \mid y = 0)}$
Logistic Regression	$\text{maximize} \sum_i \log(P(y_i \mid \mathbf{x}, \boldsymbol{\theta})) + \lambda \lVert \boldsymbol{\theta} \rVert$ where $P(y_i \mid \mathbf{x}, \boldsymbol{\theta}) = 1 / (1 + \exp(-y_i \boldsymbol{\theta}^T \mathbf{x}))$	Gradient ascent	$\boldsymbol{\theta}^T \mathbf{x} > t$
Linear SVM	$\text{minimize } \lambda \sum_i \xi_i + \frac{1}{2} \lVert \boldsymbol{\theta} \rVert$ such that $y_i \boldsymbol{\theta}^T \mathbf{x} \geq 1 - \xi_i \quad \forall i, \ \xi_i \geq 0$	Quadratic programming or subgradient opt.	$\boldsymbol{\theta}^T \mathbf{x} > t$
Kernelized SVM	complicated to write	Quadratic programming	$\sum_i y_i \alpha_i K(\hat{\mathbf{x}}_i, \mathbf{x}) > 0$
Nearest Neighbor	most similar features → same label	Record data	y_i where $i = \underset{i}{\text{argmin}} \ K(\hat{\mathbf{x}}_i, \mathbf{x})$

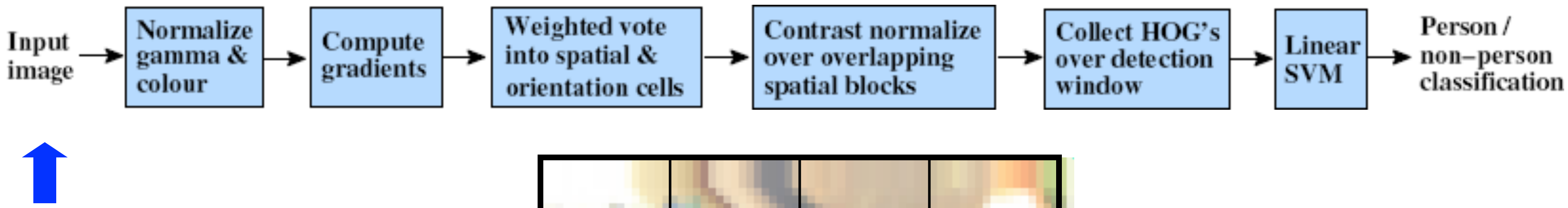
Image Categorization

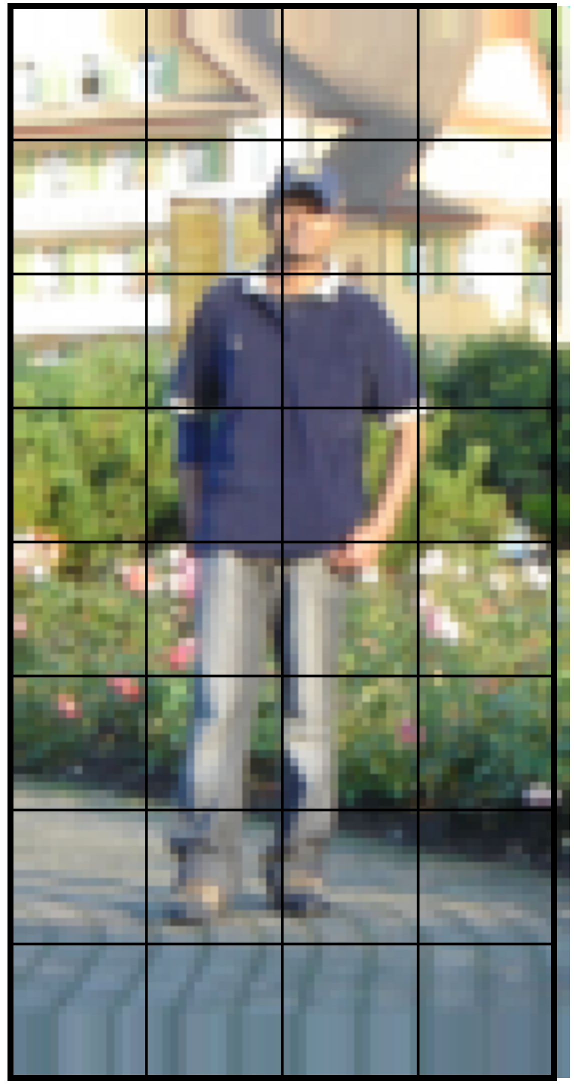


Example: Dalal-Triggs pedestrian

1. Extract fixed-sized (64x128 pixel) window at each position and scale
2. Compute HOG (histogram of gradient) features within each window
3. Score the window with a linear SVM classifier
4. Perform non-maxima suppression to remove overlapping detections with lower scores

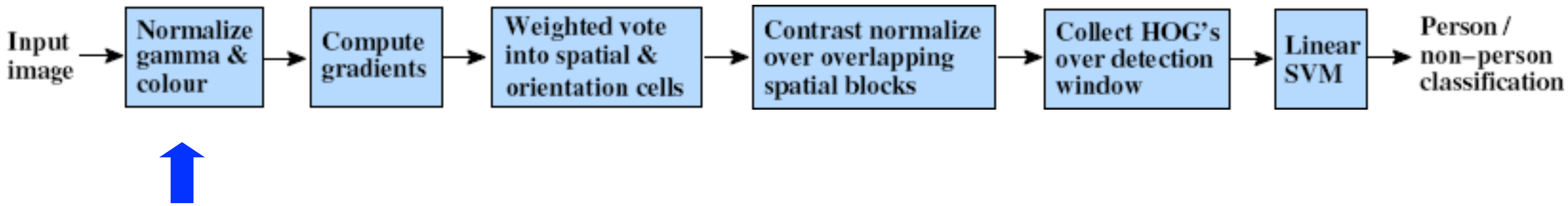
Navneet Dalal and Bill Triggs, Histograms of Oriented Gradients for Human Detection, CVPR05





Slides by Pete Barnum

Navneet Dalal and Bill Triggs, Histograms of Oriented Gradients for Human Detection, CVPR05



- Tested with
  - RGB
  - LAB
  - Grayscale

Slightly better performance vs. grayscale

Outperforms

-1	0	1

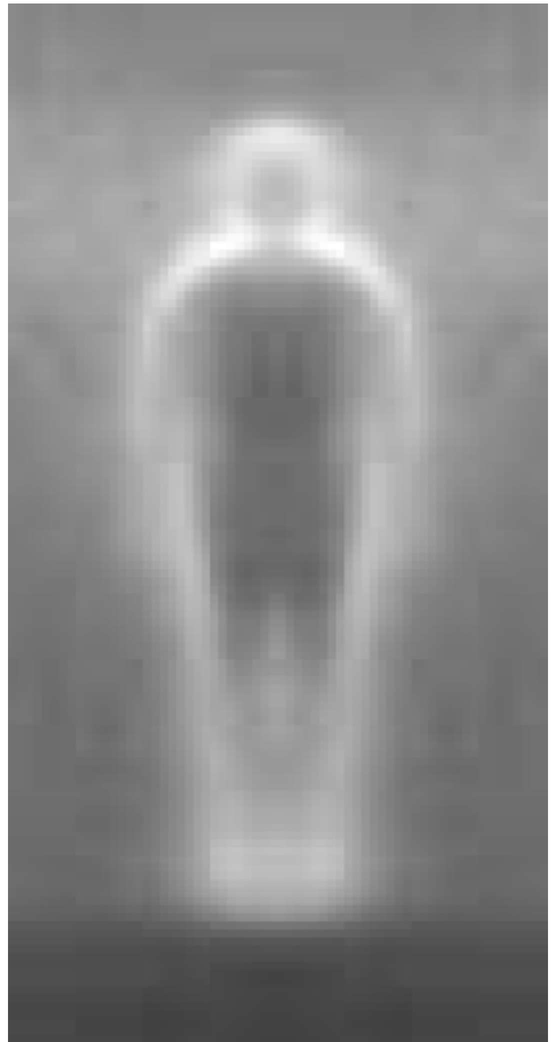
centered

-1	1

uncentered

1	-8	0	8	-1

cubic-corrected



0	1
-1	0

diagonal

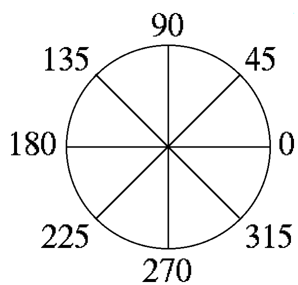
-1	0	1
-2	0	2
-1	0	1

Sobel

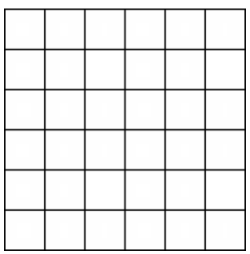
Navneet Dalal and Bill Triggs, Histograms of Oriented Gradients for Human Detection, CVPR05

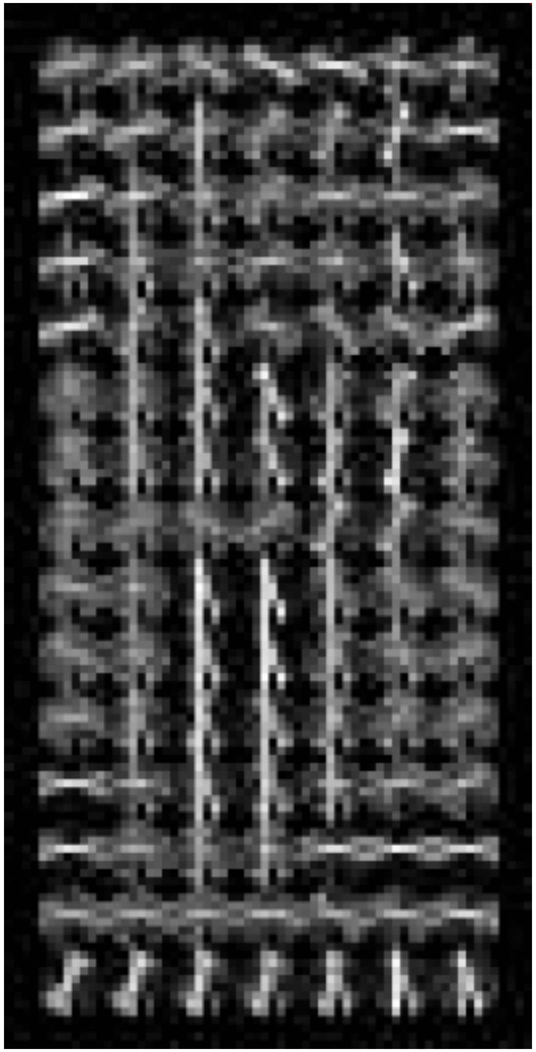
- Histogram of gradient orientations

Orientation: 9 bins (for unsigned angles)



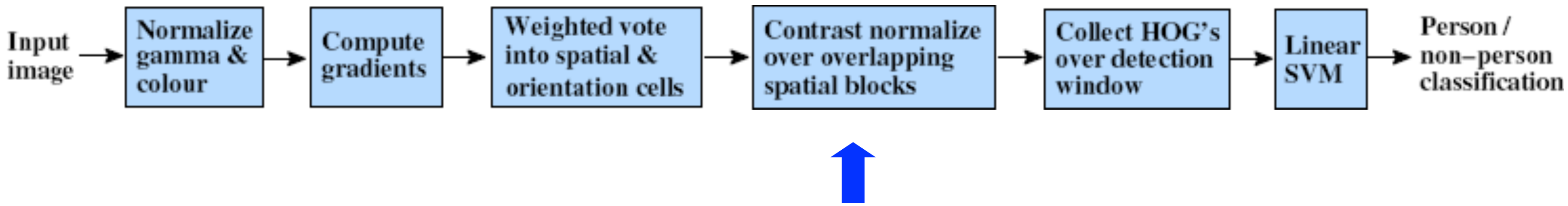
Histograms in 8x8 pixel cells



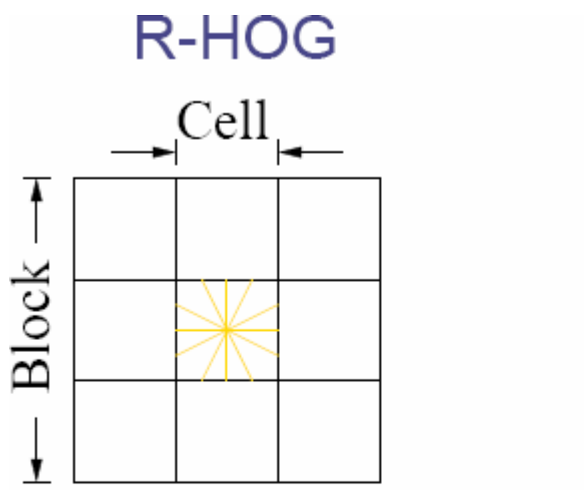


  - Votes weighted by magnitude
  - Bilinear interpolation between cells

Navneet Dalal and Bill Triggs, Histograms of Oriented Gradients for Human Detection, CVPR05

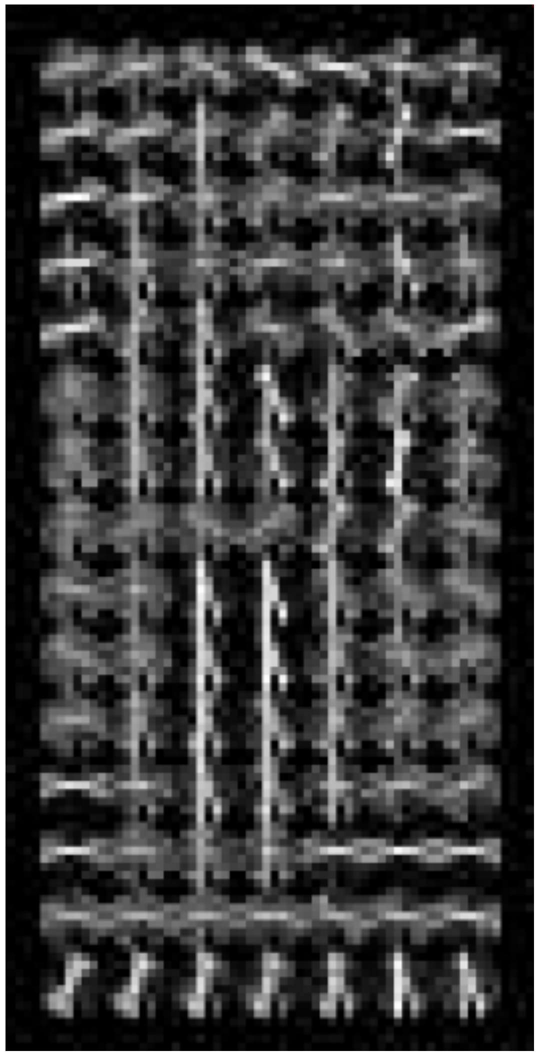


Normalize with respect to surrounding cells



$$L2-norm : v \longrightarrow v/\sqrt{||v||_2^2 + \epsilon^2}$$

X=



orientations

features = 15 x 7 x 9 x 4 = 3780

cells

normalizations by neighboring cells

Slides by Pete Barnum

Navneet Dalal and Bill Triggs, Histograms of Oriented Gradients for Human Detection, CVPR05

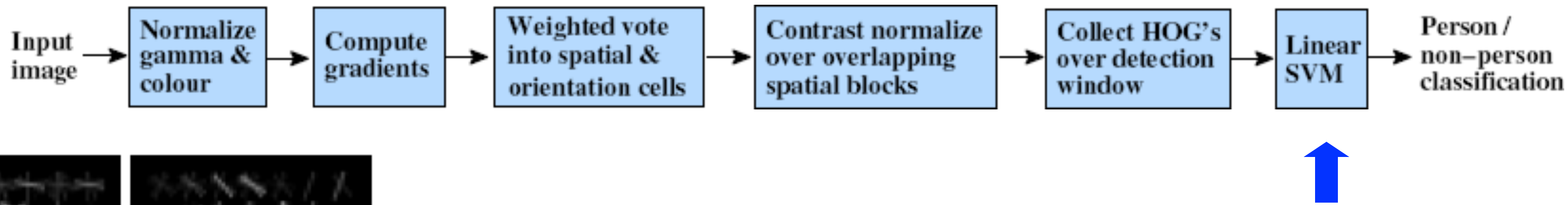
Training set

+

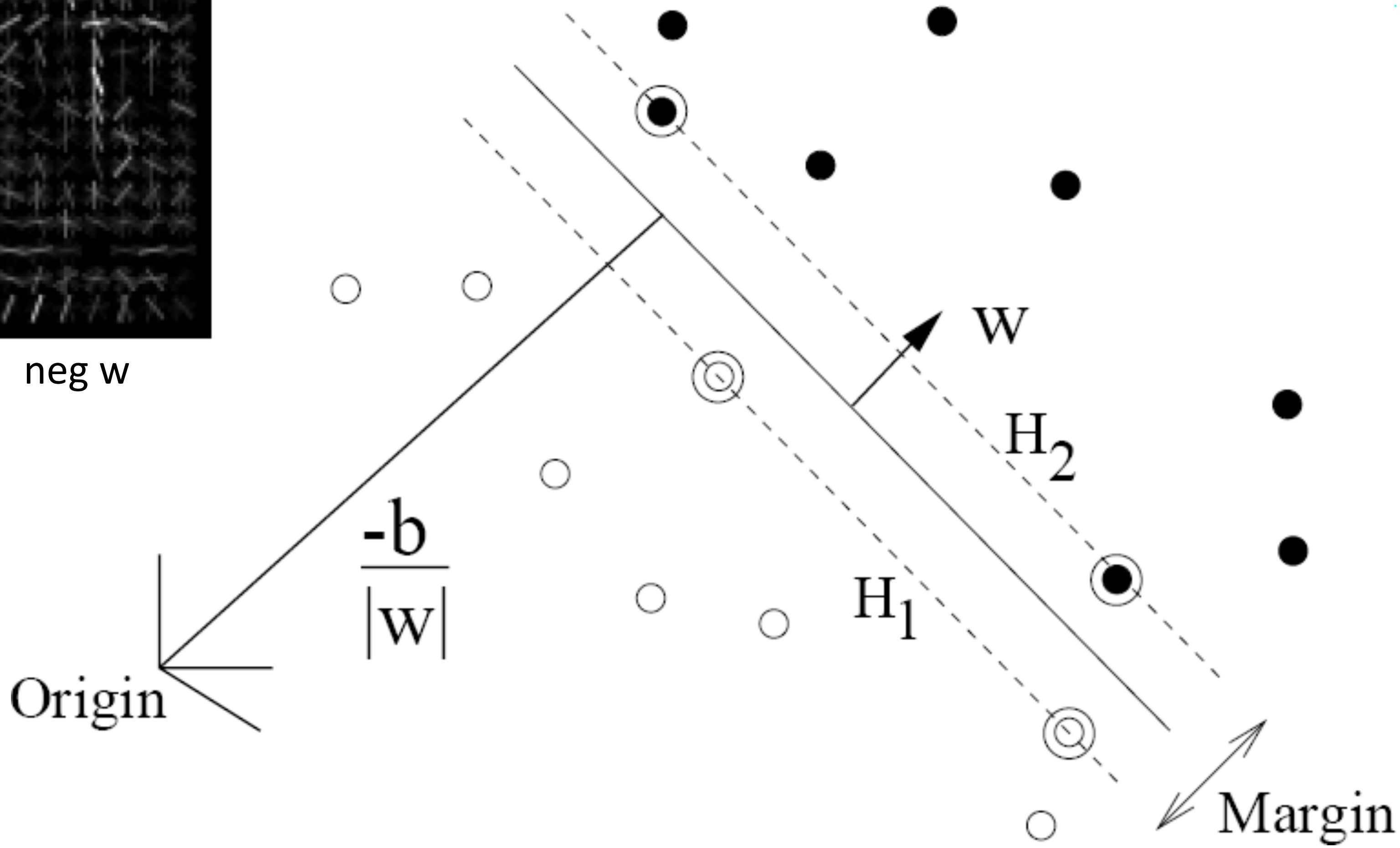
−





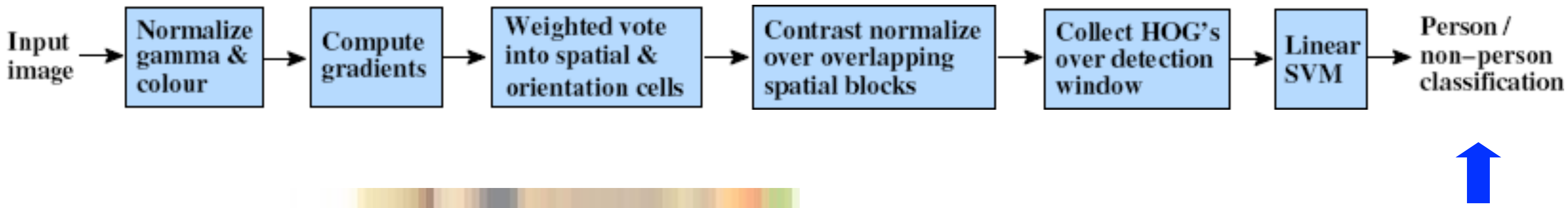



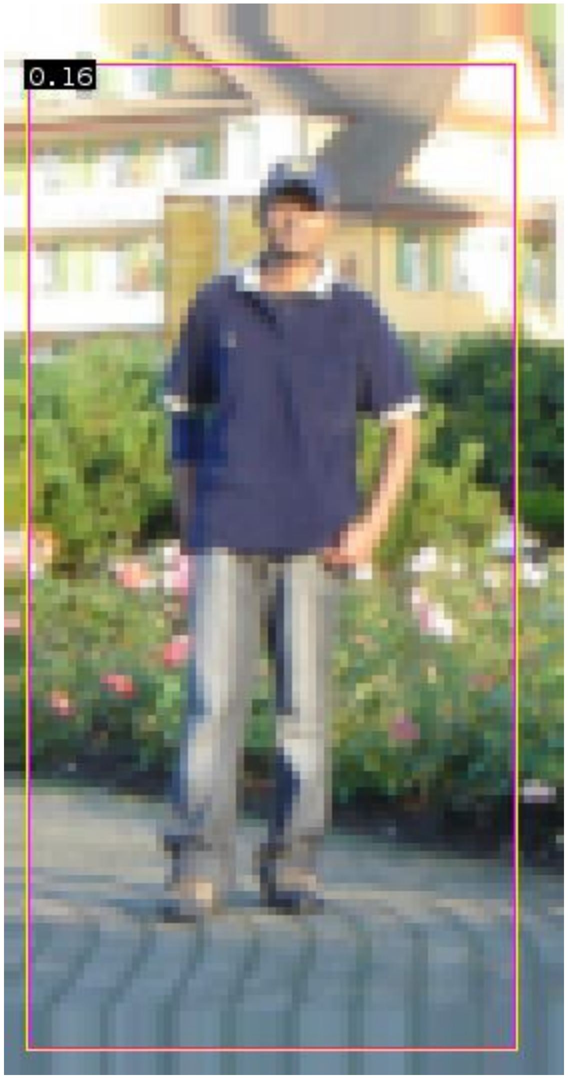
pos w   neg w



Slides by Pete Barnum

Navneet Dalal and Bill Triggs, Histograms of Oriented Gradients for Human Detection, CVPR05





$$0.16 = w^T x - b$$

$$sign(0.16) = 1$$

$=>$ pedestrian

Slides by Pete Barnum

Navneet Dalal and Bill Triggs, Histograms of Oriented Gradients for Human Detection, CVPR05

Detection examples

Each window is separately classified