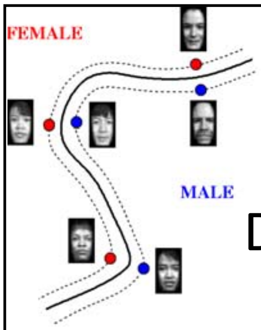


Discriminative classifiers for object classification

Thursday, Nov 12

Kristen Grauman

UT-Austin

Last time

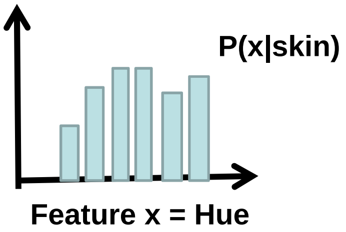
- – Supervised classification
  - Loss and risk, Bayes rule
  - Skin color detection example
- – Sliding window detection
  - Classifiers, boosting algorithm, cascades
  - Face detection example

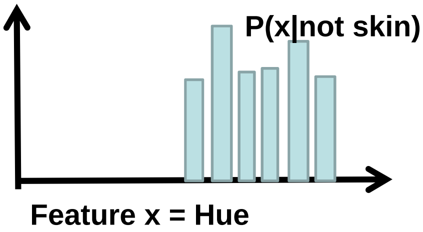
- – Limitations of a global appearance description
- – Limitations of sliding window detectors

Example: learning skin colors

- We can represent a class-conditional density using a histogram (a “non-parametric” distribution)

Now we get a new image, and want to label each pixel as skin or non-skin.





Bayes rule

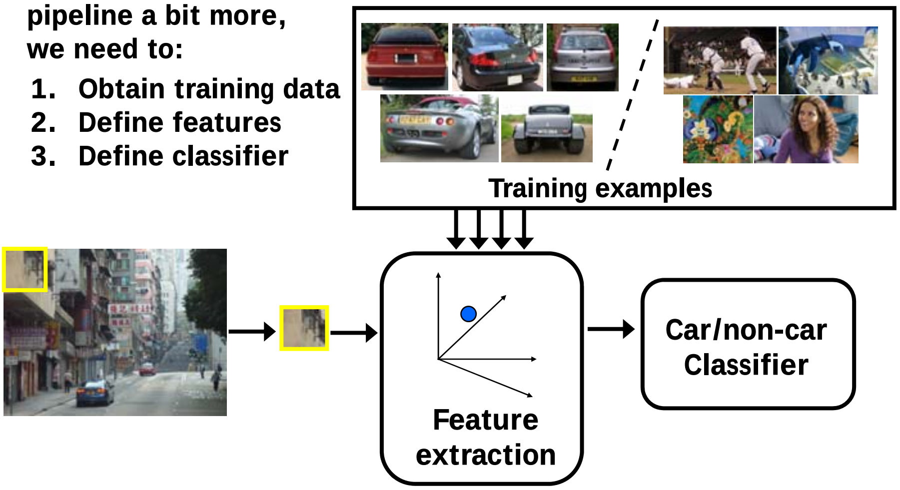
$$\overbrace{P(skin\,|\,x)}^{\text{posterior}} = \frac{\overbrace{P(x\,|\,skin)}^{\text{likelihood}}\overbrace{P(skin)}^{\text{prior}}}{P(x)}$$

$$P(skin\,|\,x)\ \alpha\ P(x\,|\,skin)P(skin)$$

Detection via classification: Main idea

Fleshing out this pipeline a bit more, we need to:

1. Obtain training data
2. Define features
3. Define classifier

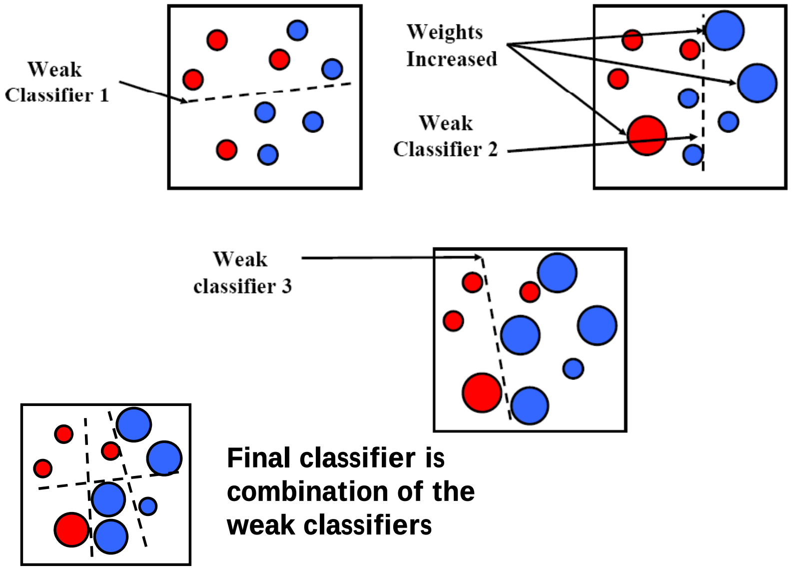


Training examples

Feature extraction

Car/non-car Classifier

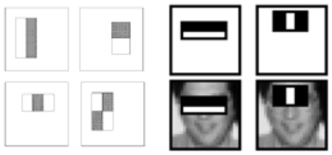
AdaBoost: Intuition



Final classifier is combination of the weak classifiers

Feature extraction: rectangular filters

"Rectangular" filters

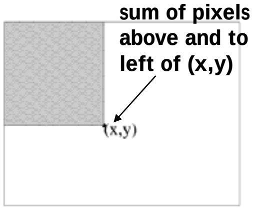


Feature output is difference between adjacent regions

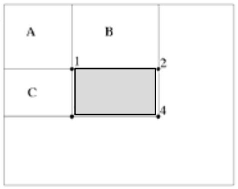
Efficiently computable with integral image: any sum can be computed in constant time

Avoid scaling images → scale features directly for same cost

Value at (x,y) is sum of pixels above and to the left of (x,y)



Integral image

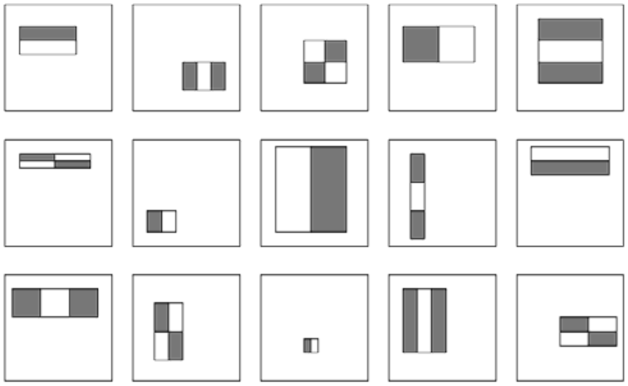


$$D = 1 + 4 - (2 + 3)$$
$$= A + (A + B + C + D) - (A + C + A + B)$$
$$= D$$

Viola & Jones, CVPR 2001

Visual Object Recognition Tutorial

Feature extraction: filter library



Considering all possible filter parameters: position, scale, and type:

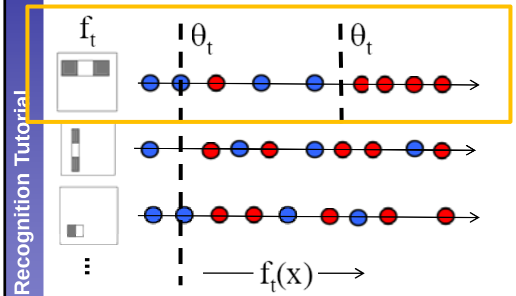
180,000+ possible features associated with each 24 x 24 window

Use AdaBoost both to select the informative features and to form the classifier

Visual Object Recognition Tutorial

AdaBoost for feature+classifier selection

- **Want to select the single rectangle feature and threshold that best separates** **positive** **(faces) and** **negative** **(non-faces) training examples, in terms of** ***weighted*** **error.**



Outputs of a possible rectangle feature on faces and non-faces.

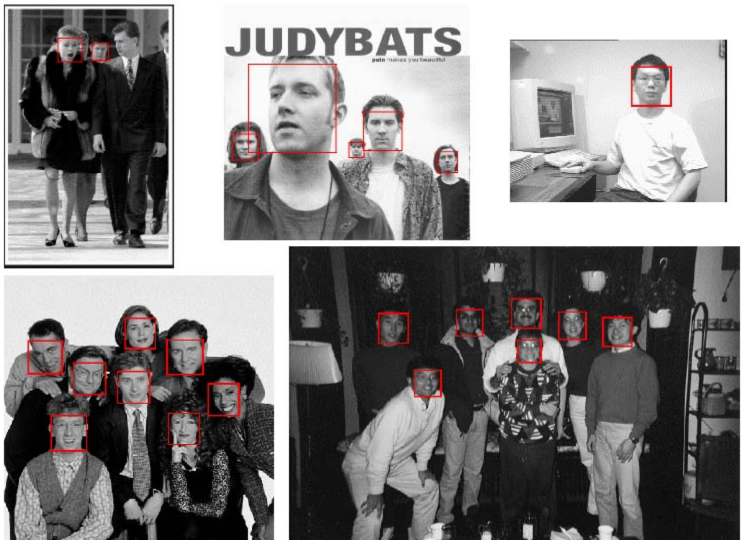
Resulting weak classifier:

$$h_t(x) = \begin{cases} +1 & \text{if } f_t(x) > \theta_t \\ -1 & \text{otherwise} \end{cases}$$

For next round, reweight the examples according to errors, choose another filter/threshold combo.

Visual Object Recognition Tutorial

Viola-Jones Face Detector: Results



Visual Object Recognition Tutorial

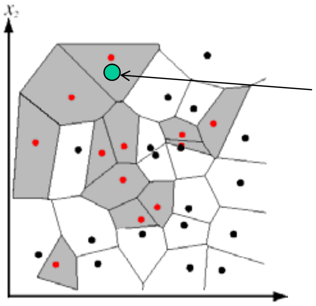
Outline

- Discriminative classifiers
  - Boosting (last time)
  - Nearest neighbors
  - Support vector machines
    - Application to pedestrian detection
    - Application to gender classification

Nearest Neighbor classification

- Assign label of nearest training data point to each test data point

Black = negative
Red = positive



from Duda *et al.*

Novel test example

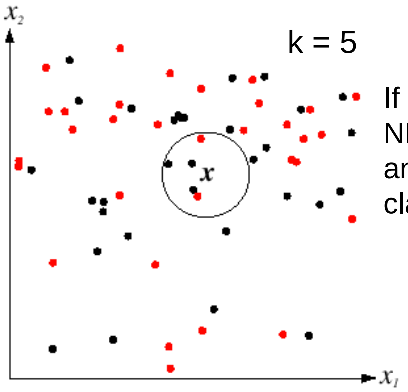
Closest to a positive example from the training set, so classify it as positive.

Voronoi partitioning of feature space
for 2-category 2D data

K-Nearest Neighbors classification

- For a new point, find the k closest points from training data
- Labels of the k points “vote” to classify

Black = negative
Red = positive



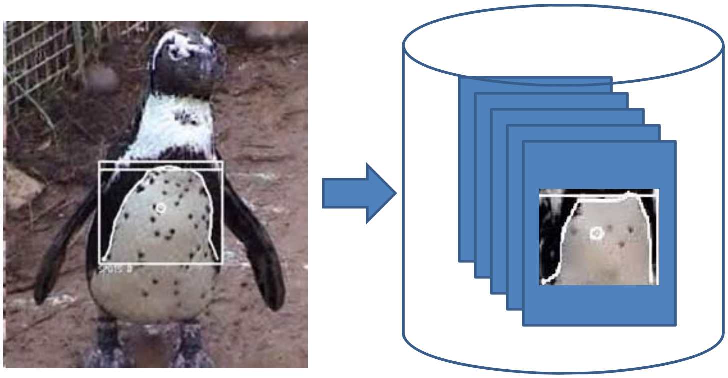
k = 5

If query lands here, the 5 NN consist of 3 negatives and 2 positives, so we classify it as negative.

Source: D. Lowe

Example: nearest neighbor classification

- We could identify the penguin in the new view based on the distance between its chest spot pattern and all the stored penguins’ patterns.



Labeled database of known penguin examples

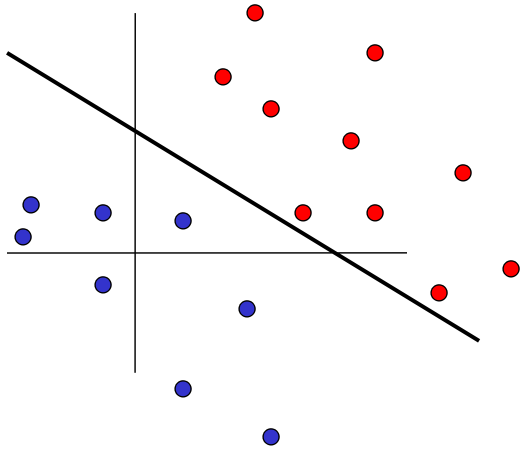
Nearest neighbors: pros and cons

- Pros:
  - Simple to implement
  - Flexible to feature / distance choices
  - Naturally handles multi-class cases
  - Can do well in practice with enough representative data
- Cons:
  - Large search problem to find nearest neighbors
  - Storage of data
  - Must know we have a meaningful distance function

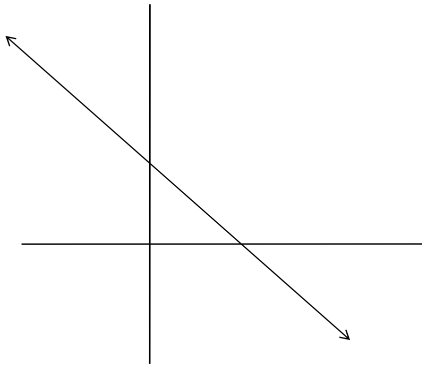
Outline

- Discriminative classifiers
  - Boosting (last time)
  - Nearest neighbors
  - <u>Support vector machines</u>
    - Application to pedestrian detection
    - Application to gender classification

Linear classifiers



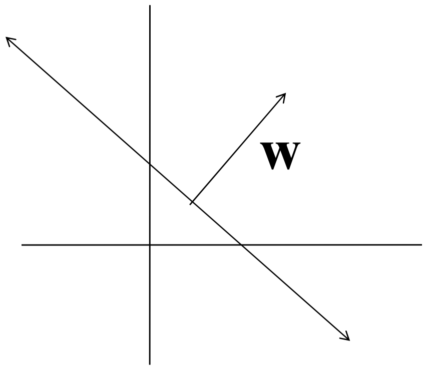
Lines in R^2



Let $\mathbf{w} = \begin{bmatrix} a \\ c \end{bmatrix}$ $\mathbf{x} = \begin{bmatrix} x \\ y \end{bmatrix}$

$$ax + cy + b = 0$$

Lines in R^2



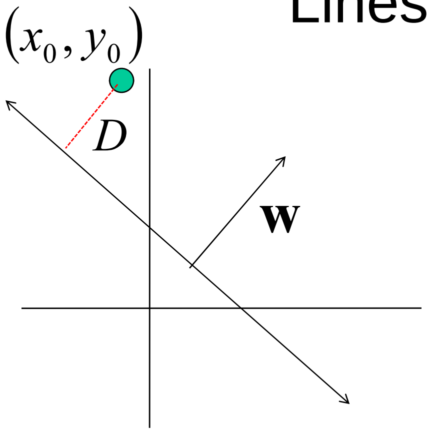
Let $\mathbf{w}=\begin{bmatrix} a \\ c \end{bmatrix}$ $\mathbf{x}=\begin{bmatrix} x \\ y \end{bmatrix}$

$$ax + cy + b = 0$$

$$\updownarrow$$

$$\mathbf{w} \cdot \mathbf{x} + b = 0$$

Lines in R^2



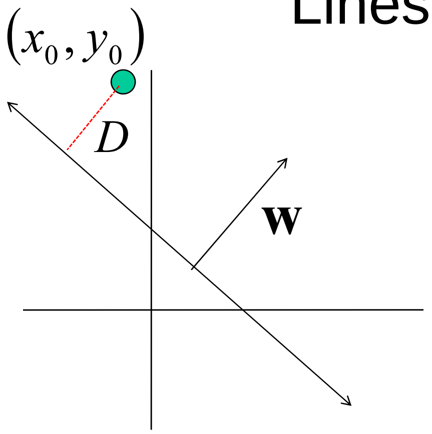
Let $\mathbf{w}=\begin{bmatrix} a \\ c \end{bmatrix}$ $\mathbf{x}=\begin{bmatrix} x \\ y \end{bmatrix}$

$$ax + cy + b = 0$$

$$\updownarrow$$

$$\mathbf{w} \cdot \mathbf{x} + b = 0$$

Lines in R^2



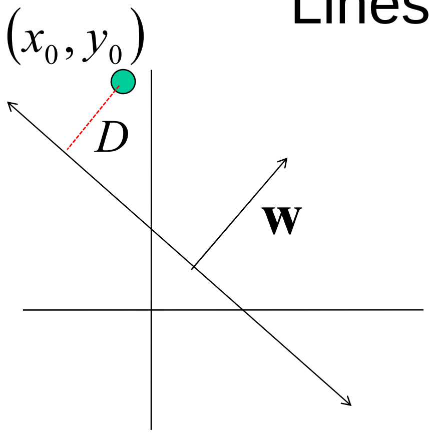
Let $\mathbf{w}=\begin{bmatrix} a \\ c \end{bmatrix}$ $\mathbf{x}=\begin{bmatrix} x \\ y \end{bmatrix}$

$$ax + cy + b = 0$$
$$\Updownarrow$$
$$\mathbf{w} \cdot \mathbf{x} + b = 0$$

$$D = \frac{|ax_0 + cy_0 + b|}{\sqrt{a^2 + c^2}}$$

distance from
point to line

Lines in R^2

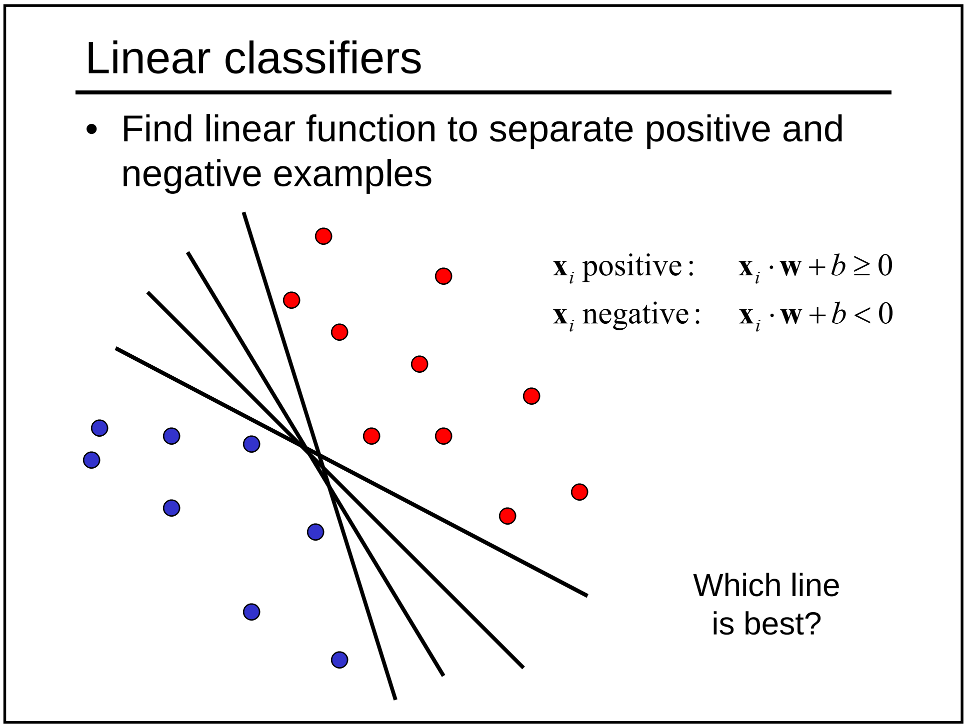


Let $\mathbf{w}=\begin{bmatrix} a \\ c \end{bmatrix}$ $\mathbf{x}=\begin{bmatrix} x \\ y \end{bmatrix}$

$$ax + cy + b = 0$$
$$\Updownarrow$$
$$\mathbf{w} \cdot \mathbf{x} + b = 0$$

$$D = \frac{|ax_0 + cy_0 + b|}{\sqrt{a^2 + c^2}} = \frac{\mathbf{w}^{\mathrm{T}}\mathbf{x} + b}{\|\mathbf{w}\|}$$

distance from
point to line



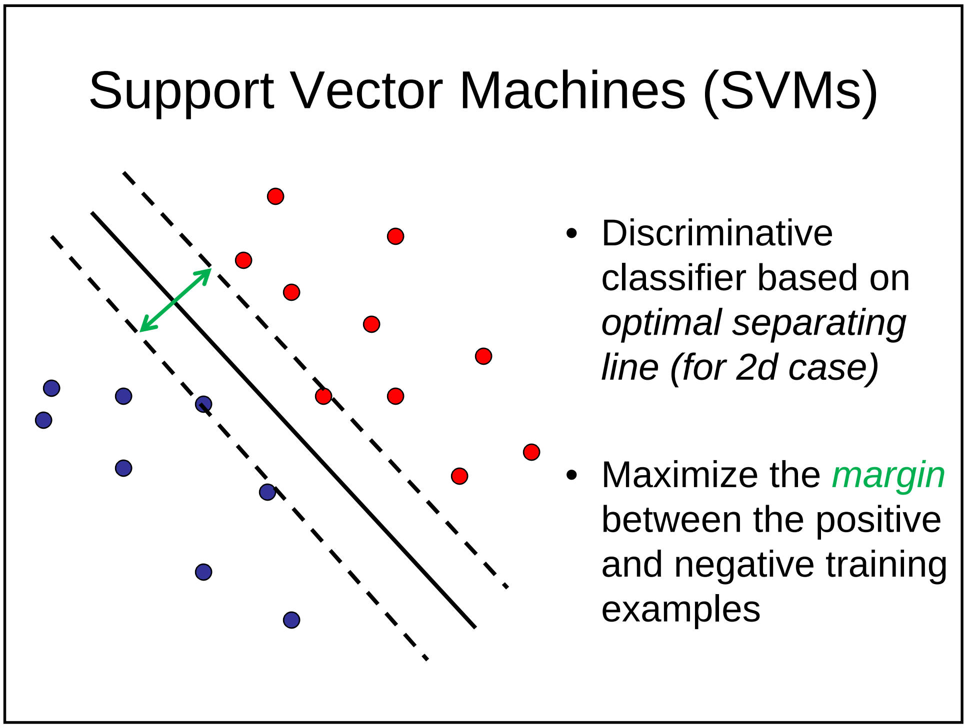
Linear classifiers

- Find linear function to separate positive and negative examples

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

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

Which line is best?

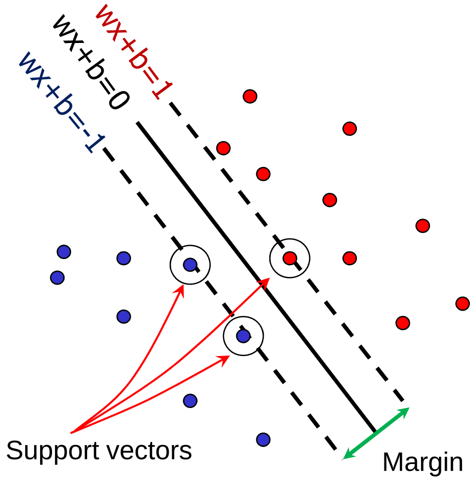


Support Vector Machines (SVMs)

- Discriminative classifier based on *optimal separating line (for 2d case)*
- Maximize the *margin* between the positive and negative training examples

Support vector machines

- Want line that maximizes the margin.



$\mathbf{x}_i$ positive $(y_i = 1)$: $\quad \mathbf{x}_i \cdot \mathbf{w} + b \geq 1$

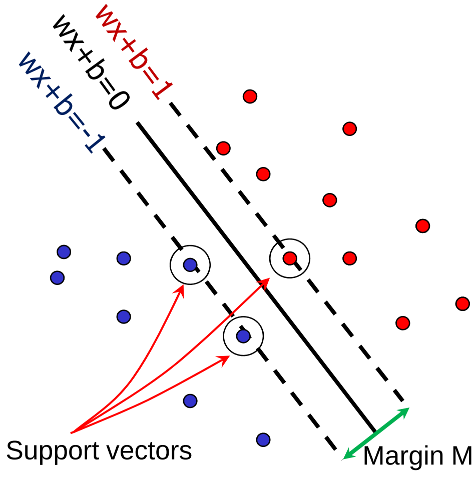
$\mathbf{x}_i$ negative $(y_i = -1)$: $\quad \mathbf{x}_i \cdot \mathbf{w} + b \leq -1$

For support, vectors, $\quad \mathbf{x}_i \cdot \mathbf{w} + b = \pm 1$

C. Burges, A Tutorial on Support Vector Machines for Pattern Recognition, Data Mining and Knowledge Discovery, 1998

Support vector machines

- Want line that maximizes the margin.



$\mathbf{x}_i$ positive $(y_i = 1)$: $\quad \mathbf{x}_i \cdot \mathbf{w} + b \geq 1$

$\mathbf{x}_i$ negative $(y_i = -1)$: $\quad \mathbf{x}_i \cdot \mathbf{w} + b \leq -1$

For support, vectors, $\quad \mathbf{x}_i \cdot \mathbf{w} + b = \pm 1$

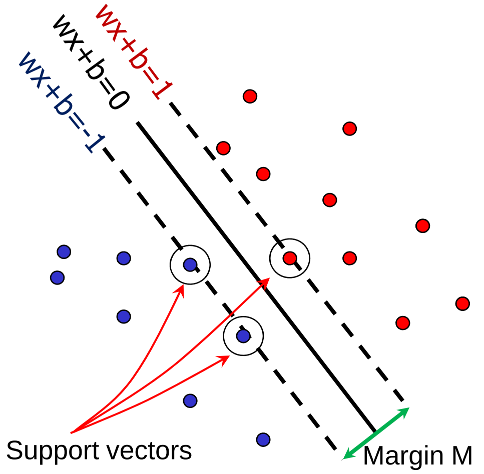
Distance between point and line: $\quad \dfrac{|\mathbf{x}_i \cdot \mathbf{w} + b|}{\|\mathbf{w}\|}$

For support vectors:

$$\frac{\mathbf{w}^T\mathbf{x} + b}{\|\mathbf{w}\|} = \frac{\pm 1}{\|\mathbf{w}\|} \qquad M = \left| \frac{1}{\|\mathbf{w}\|} - \frac{-1}{\|\mathbf{w}\|} \right| = \frac{2}{\|\mathbf{w}\|}$$

Support vector machines

- Want line that maximizes the margin.



$\mathbf{x}_i$ positive $(y_i = 1)$: $\mathbf{x}_i \cdot \mathbf{w} + b \geq 1$

$\mathbf{x}_i$ negative $(y_i = -1)$: $\mathbf{x}_i \cdot \mathbf{w} + b \leq -1$

For support, vectors, $\mathbf{x}_i \cdot \mathbf{w} + b = \pm 1$

Distance between point and line: $\frac{|\mathbf{x}_i \cdot \mathbf{w} + b|}{\|\mathbf{w}\|}$

Therefore, the margin is $2 / \|\mathbf{w}\|$

Finding the maximum margin line

1. Maximize margin $2/\|\mathbf{w}\|$
2. Correctly classify all training data points:

$\mathbf{x}_i$ positive $(y_i = 1)$: $\mathbf{x}_i \cdot \mathbf{w} + b \geq 1$

$\mathbf{x}_i$ negative $(y_i = -1)$: $\mathbf{x}_i \cdot \mathbf{w} + b \leq -1$

Quadratic optimization problem:

Minimize $\frac{1}{2}\mathbf{w}^T\mathbf{w}$

Subject to $y_i(\mathbf{w}\cdot\boldsymbol{x}_i+b) \geq 1$

One constraint for each training point.

Note sign trick.

C. Burges, A Tutorial on Support Vector Machines for Pattern Recognition, Data Mining and Knowledge Discovery, 1

Finding the maximum margin line

- Solution: $\mathbf{w} = \sum_i \underbrace{\alpha_i}_{\text{learned weight}} y_i \underbrace{\mathbf{x}_i}_{\text{Support vector}}$

Finding the maximum margin line

- Solution: $\mathbf{w} = \sum_i \alpha_i y_i \mathbf{x}_i$

  $b = y_i - \mathbf{w}\cdot\mathbf{x}_i$ (for any support vector)

  $$\mathbf{w}\cdot\mathbf{x} + b = \sum_i \alpha_i y_i \mathbf{x}_i \cdot \mathbf{x} + b$$

- Classification function:

$$f(x) = \text{sign}\,(\mathbf{w}\cdot\mathbf{x} + \text{b})$$
$$= \text{sign}\left(\sum_{\text{i}} \alpha_{\text{i}} \boxed{\mathbf{x}_i \cdot \mathbf{x}} + b\right)$$

If f(x) < 0, classify as negative, if f(x) > 0, classify as positive

- Notice that it relies on an *inner product* between the test point ***x*** and the support vectors $\boldsymbol{x}_i$
- (Solving the optimization problem also involves computing the inner products $\boldsymbol{x}_i \cdot \boldsymbol{x}_j$ between all pairs of training points)

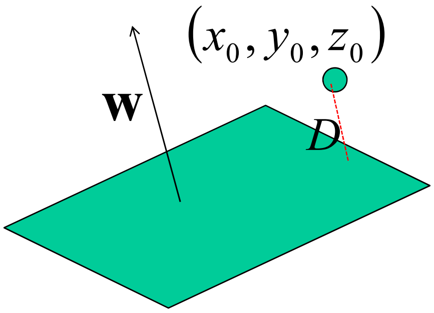
Questions

- How is the SVM objective different from the boosting objective?
- What if the features are not 2d?
- What if the data is not linearly separable?
- What if we have more than just two categories?

Questions

- How is the SVM objective different from the boosting objective?
- What if the features are not 2d?
    - – Generalizes to d-dimensions – replace line with “hyperplane”
- What if the data is not linearly separable?
- What if we have more than just two categories?

Planes in R^3



Let $\mathbf{w} = \begin{bmatrix} a \\ b \\ c \end{bmatrix}$ $\mathbf{x} = \begin{bmatrix} x \\ y \\ z \end{bmatrix}$

$$ax + by + cz + d = 0$$

$$\updownarrow$$

$$\mathbf{w} \cdot \mathbf{x} + d = 0$$

$$D = \frac{|ax_0 + by_0 + cz_0 + d|}{\sqrt{a^2 + b^2 + c^2}} = \frac{\mathbf{w}^{\mathrm{T}}\mathbf{x} + d}{\|\mathbf{w}\|}$$

distance from point to plane

Hyperplanes in R^n

Hyperplane *H* is set of all vectors $\mathbf{x} \in R^n$ which satisfy:

$$w_1 x_1 + w_2 x_2 + \ldots + w_n x_n + b = 0$$

$$\updownarrow$$

$$\mathbf{w}^{\mathrm{T}}\mathbf{x} + b = 0$$

$$D(H, \mathbf{x}) = \frac{\mathbf{w}^{\mathrm{T}}\mathbf{x} + b}{\|\mathbf{w}\|}$$

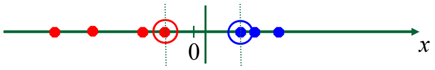
distance from point to hyperplane

Questions

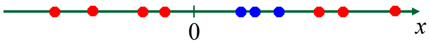
- What if the features are not 2d?
- **What if the data is not linearly separable?**
- What if we have more than just two categories?

Non-linear SVMs

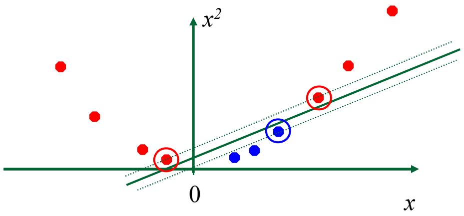
- Datasets that are linearly separable with some noise work out great:



- But what are we going to do if the dataset is just too hard?

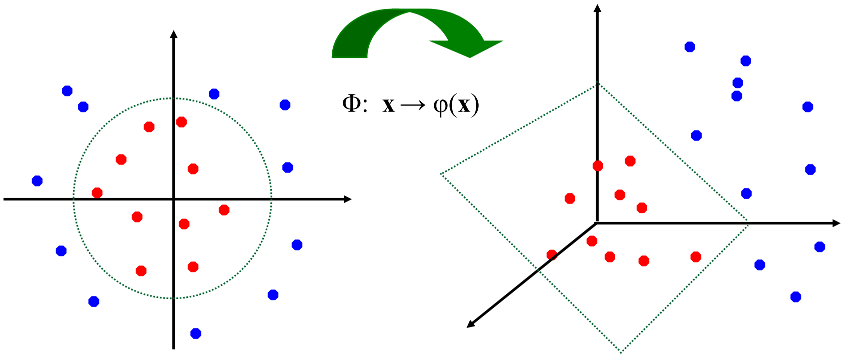


- How about… mapping data to a higher-dimensional space:



Non-linear SVMs: feature spaces

- General idea: the original input space can be mapped to some higher-dimensional feature space where the training set is separable:



Slide from Andrew Moore's tutorial: http://www.autonlab.org/tutorials/svm.html

---

Nonlinear SVMs

- *The kernel trick*: instead of explicitly computing the lifting transformation $\boldsymbol{\varphi}(\mathbf{x})$, define a kernel function K such that

$$K(\mathbf{x}_i, \mathbf{x}_j) = \boldsymbol{\varphi}(\mathbf{x}_i) \cdot \boldsymbol{\varphi}(\mathbf{x}_j)$$

- This gives a nonlinear decision boundary in the original feature space:

$$\sum_i \alpha_i y_i K(\mathbf{x}_i, \mathbf{x}) + b$$

Examples of kernel functions

- Linear: $K(x_i, x_j) = x_i^T x_j$
- Gaussian RBF: $K(x_i,x_j) = \exp(-\frac{\left\|x_i - x_j\right\|^2}{2\sigma^2})$
- Histogram intersection:

$$K(x_i, x_j) = \sum_k \min(x_i(k), x_j(k))$$

Questions

- What if the features are not 2d?
- What if the data is not linearly separable?
- **What if we have more than just two categories?**

Multi-class SVMs

- Achieve multi-class classifier by combining a number of binary classifiers

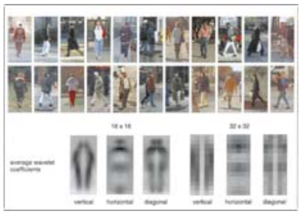
- **<u>One vs. all</u>**
  - Training: learn an SVM for each class vs. the rest
  - Testing: apply each SVM to test example and assign to it the class of the SVM that returns the highest decision value
- **<u>One vs. one</u>**
  - Training: learn an SVM for each pair of classes
  - Testing: each learned SVM "votes" for a class to assign to the test example

SVMs for recognition

1. Define your representation for each example.
2. Select a kernel function.
3. Compute pairwise kernel values between labeled examples
4. Give this "kernel matrix" to SVM optimization software to identify support vectors & weights.
5. To classify a new example: compute kernel values between new input and support vectors, apply weights, check sign of output.

Pedestrian detection

- Detecting upright, walking humans also possible using sliding window's appearance/texture; e.g.,

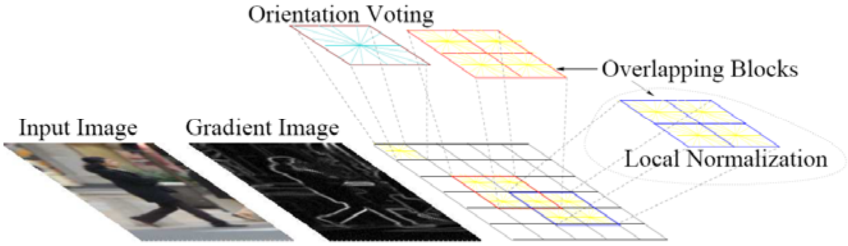


SVM with Haar wavelets [Papageorgiou & Poggio, IJCV 2000]

Space-time rectangle features [Viola, Jones & Snow, ICCV 2003]

SVM with HoGs [Dalal & Triggs, CVPR 2005]

Example: pedestrian detection with HoG's and SVM's



- Map each grid cell in the input window to a histogram counting the gradients per orientation.
- Train a linear SVM using training set of pedestrian vs. non-pedestrian windows.

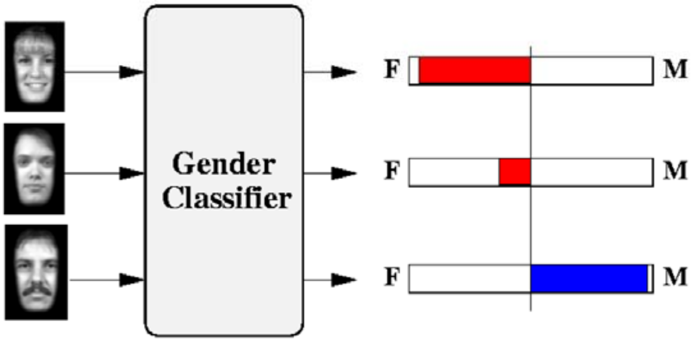
Dalal & Triggs, CVPR 2005

Code available: http://pascal.inrialpes.fr/soft/olt/

Pedestrian detection with HoG's & SVM's

- **Histograms of Oriented Gradients for Human Detection, Navneet Dalal, Bill Triggs, International Conference on Computer Vision & Pattern Recognition - June 2005**
- **http://lear.inrialpes.fr/pubs/2005/DT05/**

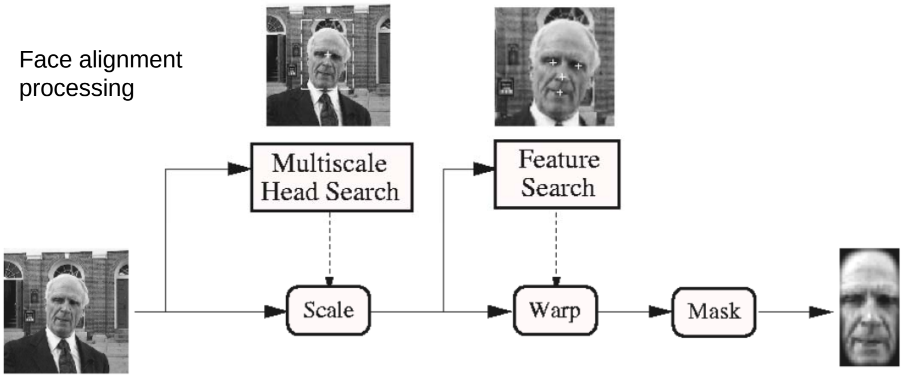
Example: learning gender with SVMs



Moghaddam and Yang, Learning Gender with Support Faces, TPAMI 2002.

Moghaddam and Yang, Face & Gesture 2000.

Face alignment processing



Processed faces

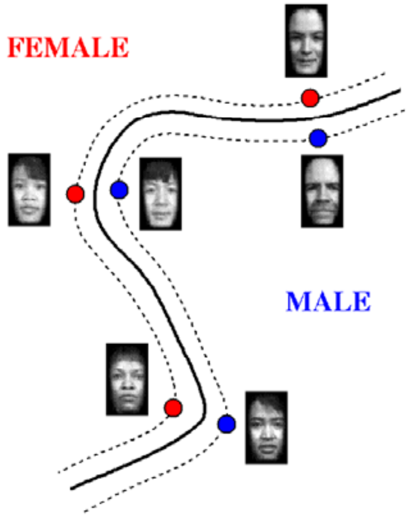
Moghaddam and Yang, Learning Gender with Support Faces, TPAMI 2002.

Learning gender with SVMs

- Training examples:
  - 1044 males
  - 713 females
- Experiment with various kernels, select Gaussian RBF

$$K(\mathbf{x_i}, \mathbf{x_j}) = \exp(-\frac{\left\|\mathbf{x_i} - \mathbf{x_j}\right\|^2}{2\sigma^2})$$

Support Faces



Moghaddam and Yang, Learning Gender with Support Faces, TPAMI 2002.

Classifier Performance

Classifier	Error Rate		
	Overall	Male	Female
SVM with RBF kernel	**3.38%**	**2.05%**	**4.79%**
SVM with cubic polynomial kernel	**4.88%**	**4.21%**	**5.59%**
Large Ensemble of RBF	5.54%	4.59%	6.55%
Classical RBF	7.79%	6.89%	8.75%
Quadratic classifier	10.63%	9.44%	11.88%
Fisher linear discriminant	13.03%	12.31%	13.78%
Nearest neighbor	27.16%	26.53%	28.04%
Linear classifier	58.95%	58.47%	59.45%

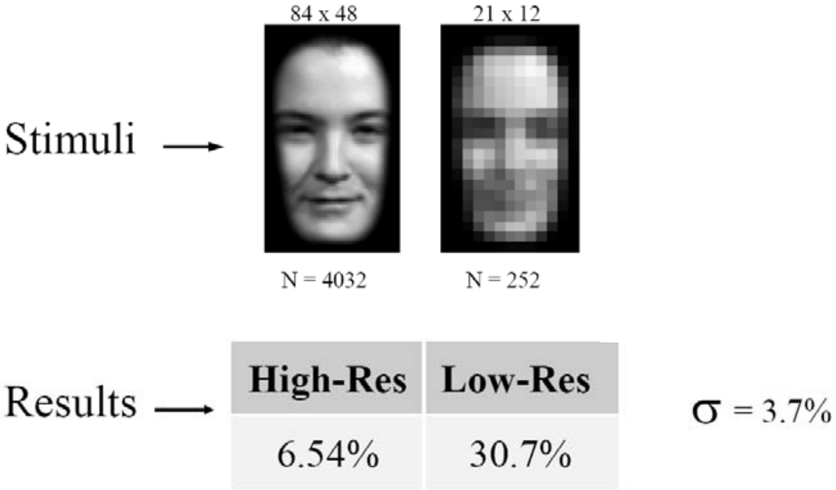
Moghaddam and Yang, Learning Gender with Support Faces, TPAMI 2002.

Gender perception experiment: How well can humans do?

- Subjects:
  - 30 people (22 male, 8 female)
  - Ages mid-20’s to mid-40’s
- Test data:
  - 254 face images (6 males, 4 females)
  - Low res and high res versions
- Task:
  - Classify as male or female, forced choice
  - No time limit

Moghaddam and Yang, Face & Gesture 2000.

Gender perception experiment: How well can humans do?



High-Res	Low-Res
6.54%	30.7%
Error	Error

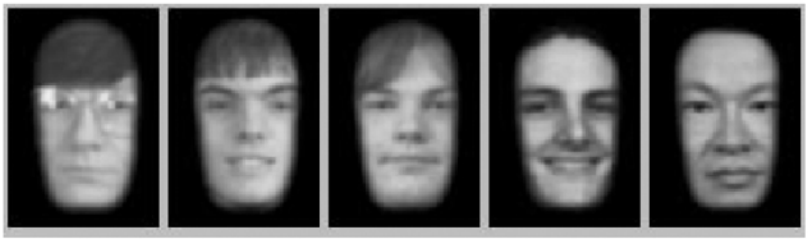
Moghaddam and Yang, Face & Gesture 2000.

Human vs. Machine

Figure 6. SVM vs. Human performance

- SVMs performed better than any single human test subject, at either resolution

Hardest examples for humans



Top five human misclassifications

Moghaddam and Yang, Face & Gesture 2000.

SVMs: Pros and cons

- Pros
  - Many publicly available SVM packages: http://www.kernel-machines.org/software
    http://www.csie.ntu.edu.tw/~cjlin/libsvm/
  - Kernel-based framework is very powerful, flexible
  - Often a sparse set of support vectors – compact at test time
  - Work very well in practice, even with very small training sample sizes
- Cons
  - No "direct" multi-class SVM, must combine two-class SVMs
  - Can be tricky to select best kernel function for a problem
  - Computation, memory
    - During training time, must compute matrix of kernel values for every pair of examples
    - Learning can take a very long time for large-scale problems

Adapted from Lana Lazebnik

Summary

- Discriminative classifiers applied to object detection / categorization problems.
  - Boosting (last time)
  - Nearest neighbors
  - Support vector machines
    - Application to pedestrian detection
    - Application to gender classification