

Lecture 8

Recognition and Classification (continued)

COS 429: Computer Vision



Review: Image Classification Steps

Training

Training Images



Image Features

Training Labels

Training

Learned model

Testing



Test Image

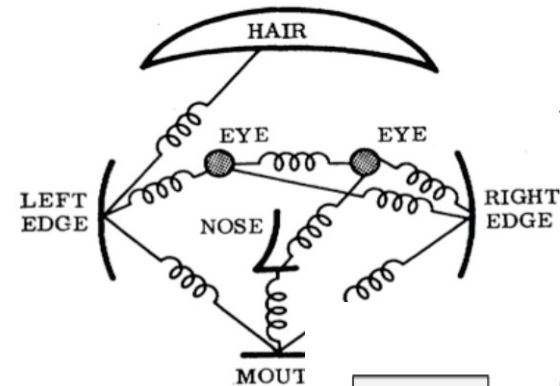
Image Features

Learned model

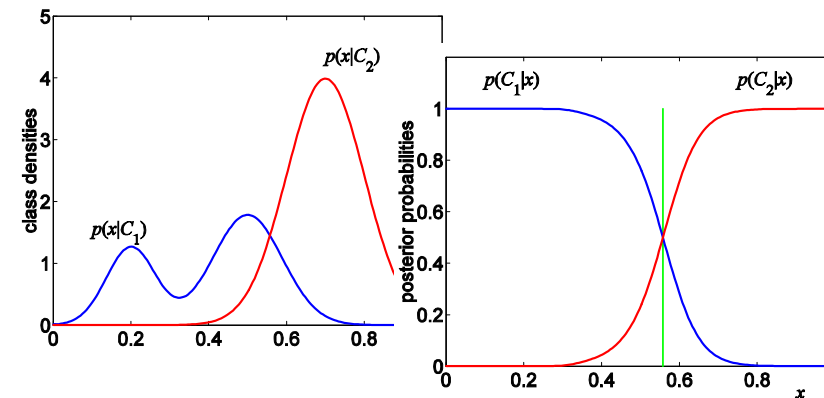
Prediction

Review: Object Detection Typical Components

- **Hypothesis** generation
 - Sliding window, Segmentation, feature point detection, random, search
- **Encoding** of (local) image data
 - Colors, Edges, Corners, Histogram of Oriented Gradients, Wavelets, Convolution Filters
- **Relationship** of different parts to each other
 - Blur or histogram, Tree/Star, Pairwise/Covariance
- **Learning** from (labeled) examples
 - Selecting representative examples (templates), Clustering, Building a cascade
 - Classifiers: Bayes, Logistic regression, SVM, AdaBoost, Neural Network...
 - Generative vs. Discriminative
- **Verification** - removing redundant, overlapping, incompatible examples
 - Non-Max Suppression, context priors, geometry



Exemplar Summary



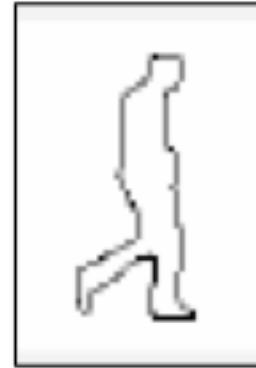
Example 1: Chamfer matching (Pedestrian Detection)



Input Image



Edge Detection



Template



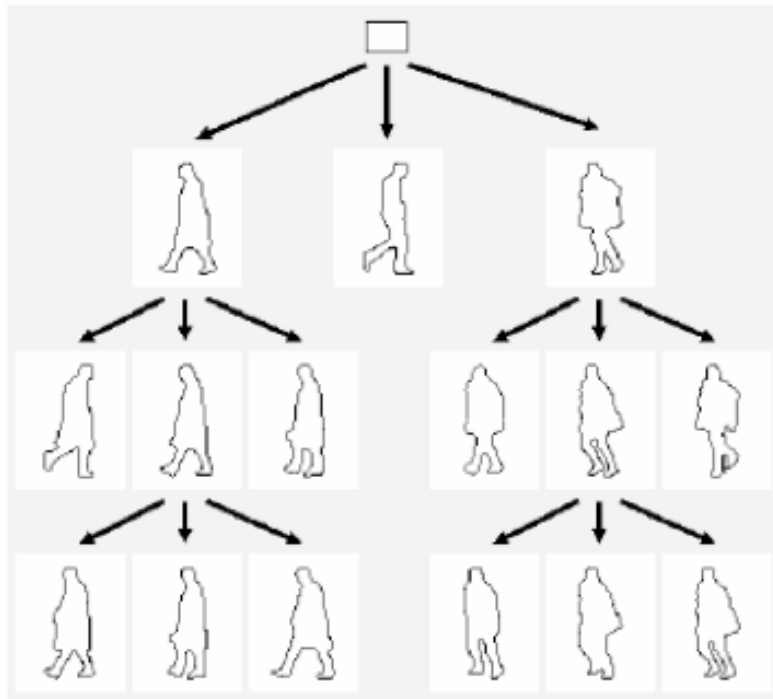
Find Best Match

$$D_{chamfer}(T, I) \equiv \frac{1}{|T|} \sum_{t \in T} d_I(t)$$

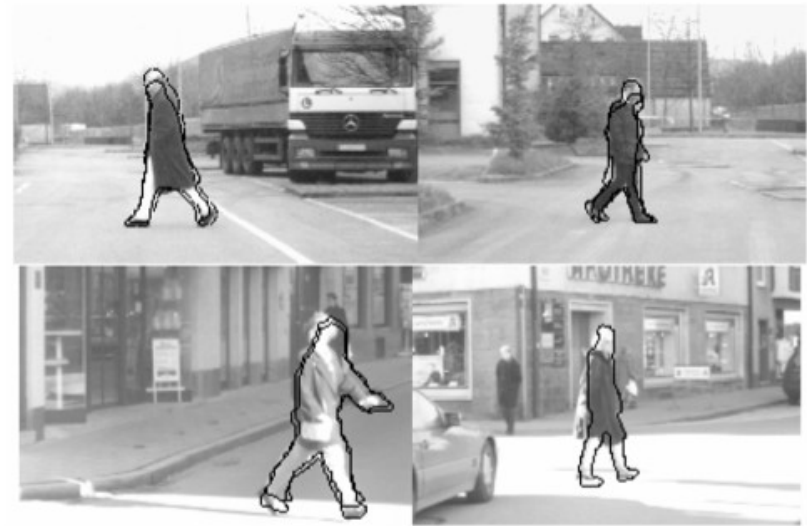


Distance Transform

Example 1: Chamfer matching (Pedestrian Detection)



Hierarchy of templates



Clustering Strategies

- K-means
 - Iteratively re-assign points to the nearest cluster center
- Mean-shift clustering
 - Estimate modes of pdf
- Agglomerative clustering
 - Start with each point as its own cluster and iteratively merge the closest clusters
- Spectral clustering
 - Split the nodes in a graph based on assigned links with similarity weights

As we go down this chart, the clustering strategies have more tendency to transitively group points even if they are not nearby in feature space

Example 2: Viola/Jones (Face Detection)

Robust Realtime Face Detection, IJCV 2004, Viola and Jones

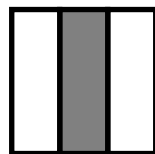
Features: “Haar-like Rectangle filters”

- Differences between sums of pixels in adjacent rectangles

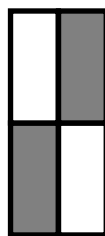
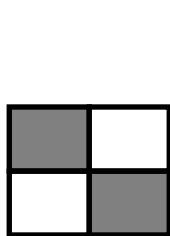
-1 +1



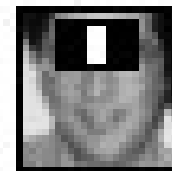
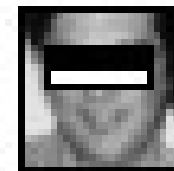
2-rectangle features



3-rectangle features



4-rectangles features

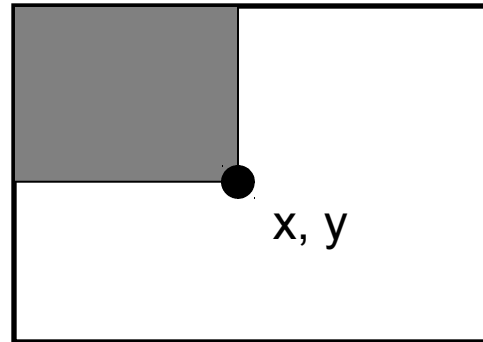


$60,000 \times 100 = 6,000,000$

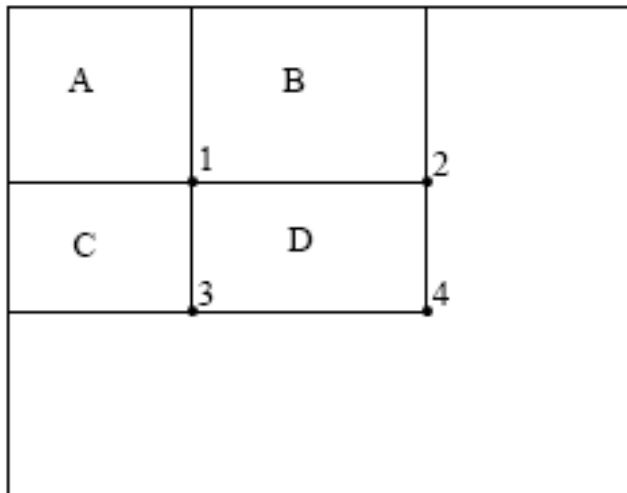
Unique Features

Example 2: Viola/Jones - Integral Images

- $ii = \text{cumsum}(\text{cumsum}(im, 1), 2)$



$ii(x,y)$ = Sum of the values in the grey region



How to compute $B-A$?

How to compute $A+D-B-C$?

Example 2: Feature selection with Adaboost

1. Create a large pool of features
2. Select features that are discriminative and work well together:
 - “Weak learner” = feature + threshold + parity
 - Choose weak learner that minimizes error on the weighted training set
 - Reweight

Trained Classifier
(for each stage of cascade)

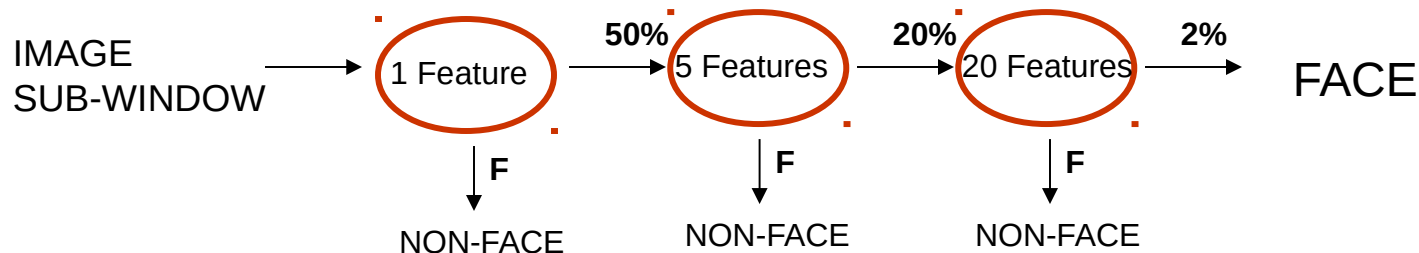
$$y_t(x) = \begin{cases} +1 & \text{if } h_t(x) > \theta_t \\ -1 & \text{otherwise} \end{cases}$$

$$Y(x) = \sum \alpha_t y_t(x)$$

$$\text{Decision} = \begin{cases} \text{face,} & \text{if } Y(x) > 0 \\ \text{non-face,} & \text{otherwise} \end{cases}$$

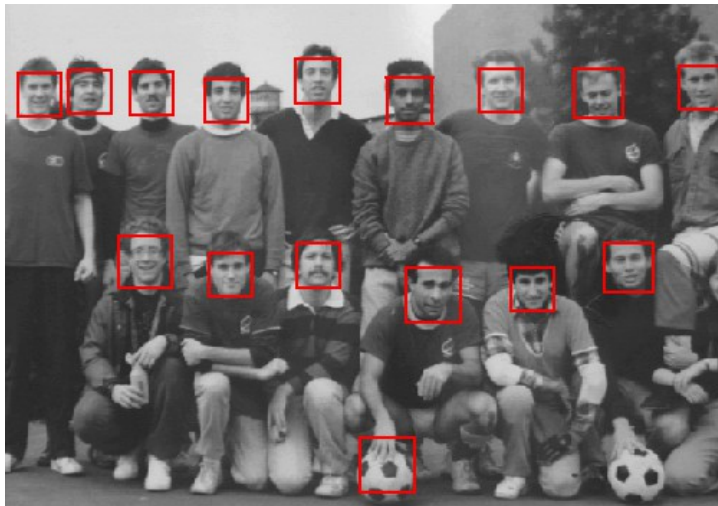
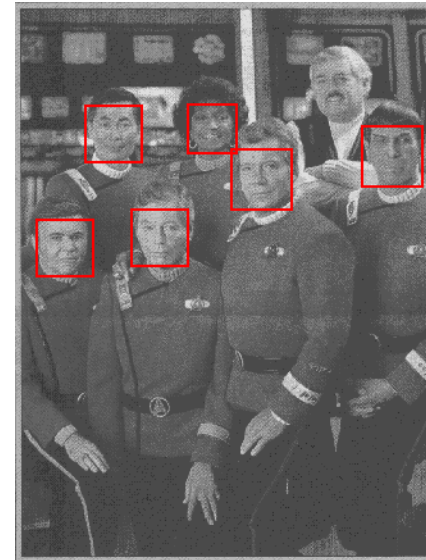
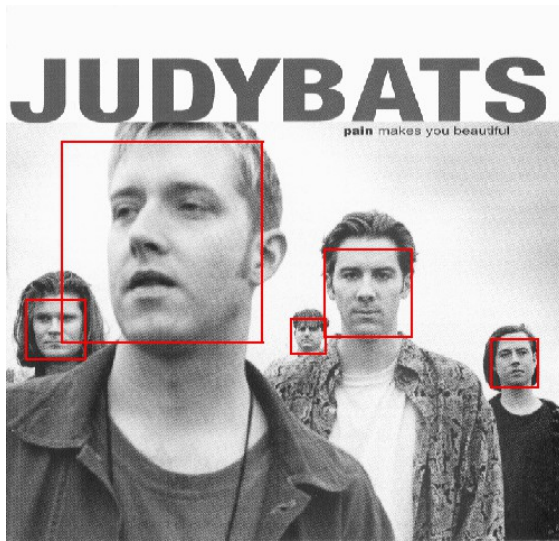
(More on AdaBoost next time...)

Example 2: Viola/Jones Cascaded Classifier



- first classifier: 100% detection, 50% false positives.
- second classifier: 100% detection, 40% false positives (20% cumulative)
using data from previous stage.
- third classifier: 100% detection, 10% false positive rate (2% cumulative)
- Put cheaper classifiers up front

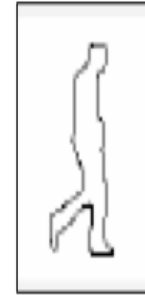
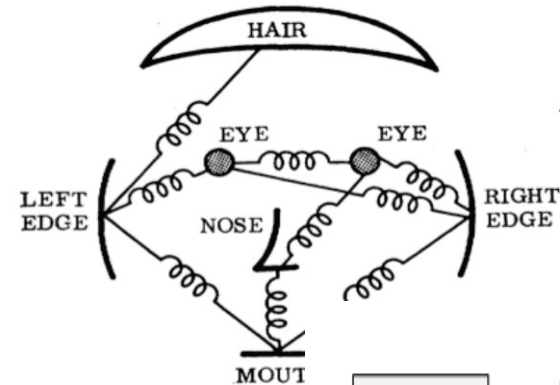
Example 2: Viola/Jones results



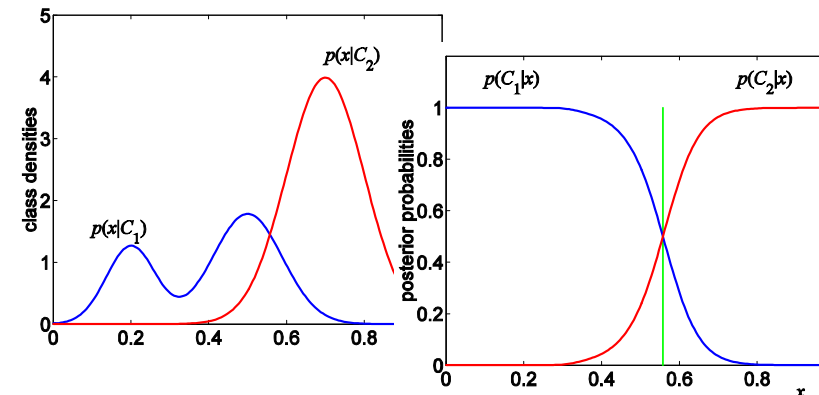
Run-time: 15fps (384x288 pixel image on a 700 Mhz Pentium III)

Typical Components

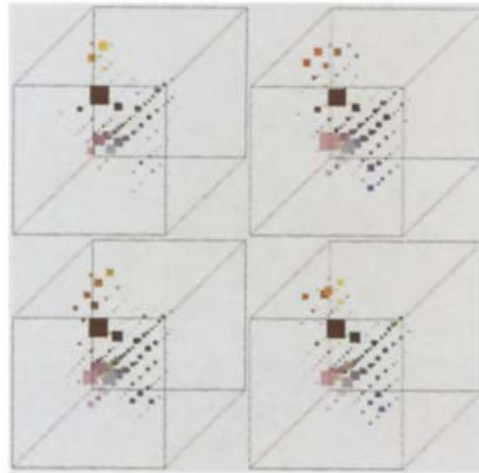
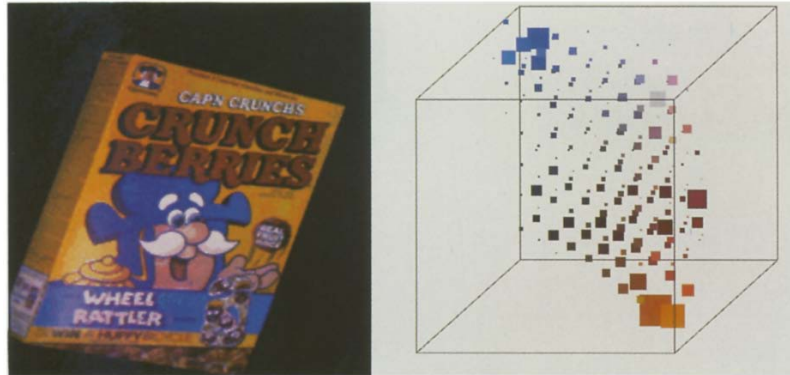
- **Hypothesis** generation
 - Whole image, Sliding window, Segmentation, Feature point detection, Search...
- **Encoding** of (local) image data
 - Colors, Edges, Corners, Histogram of Oriented Gradients, Wavelets, Convolution Filters...
- **Relationship** of different parts to each other
 - Blur, Histogram, Tree/Star, Pairwise/Covariance...
- **Learning** from labeled examples
 - Selecting representative examples (templates), Clustering, Building a cascade
 - Classifiers: Bayes, Logistic regression, SVM, AdaBoost, ...
 - Generative vs. Discriminative
- **Verification** - removing redundant, overlapping, incompatible examples
 - Non-Max Suppression, context priors, geometry



Exemplar Summary



(No Geometry) Example: Color Histograms



Swain and Ballard, [Color Indexing](#), IJCV 1991.

(No Geometry) Example: Bag of Words

Object



**Bag of
'words'**



Objects as texture

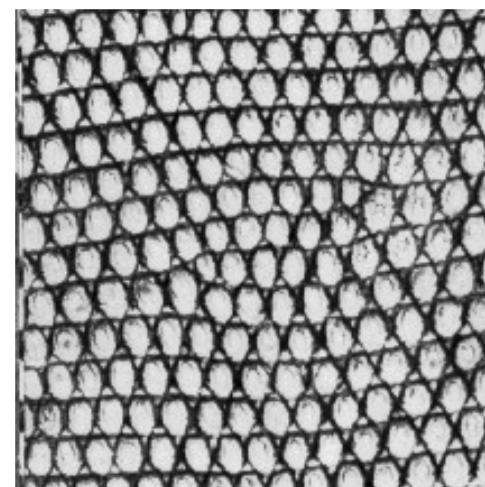
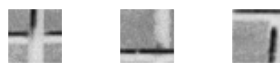
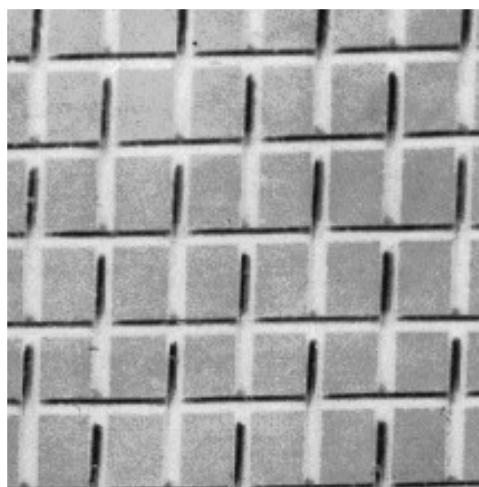
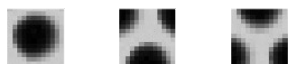
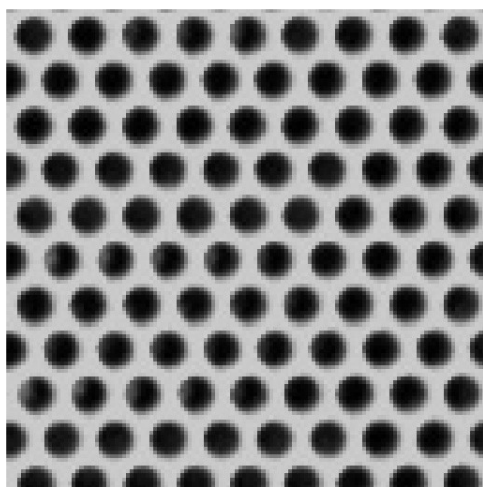
- All of these are treated as being the same



- No distinction between foreground and background: scene recognition?

• Origin 1: Texture recognition

- Texture is characterized by the repetition of basic elements or *textons*
- For stochastic textures, it is the identity of the textons, not their spatial arrangement, that matters



Julesz, 1981; Cula & Dana, 2001; Leung & Malik 2001; Mori, Belongie & Malik, 2001; Schmid 2001; Varma & Zisserman, 2002, 2003; Lazebnik, Schmid & Ponce, 2003

• Origin 2: Bag-of-words models

2007-01-23: State of the Union Address

George W. Bush (2001-)

abandon accountable affordable afghanistan africa aided ally anbar armed army **baghdad** bless challenges chamber chaos
choices civilians coalition commanders **commitment** confident confront congressman constitution corps debates deduction

deficit deliver
expand extr

1962-10-22: Soviet Missiles in Cuba

John F. Kennedy (1961-63)

insurgents ira
palestinian pay

abandon achieving adversaries aggression agricultural appropriate armaments **arms** assessments atlantic ballistic berlin
buildup burdens cargo college commitment communist constitution consumers cooperation crisis **cuba** dangers
declined **defensive** **economic**

september sh
violence vio

elimination emerge

halt hazards hen

modernization neglect

recession rejection r

surveillance tax te

1941-12-08: Request for a Declaration of War

Franklin D. Roosevelt (1933-45)

abandoning acknowledge aggression aggressors airplanes armaments **armed army** assault assembly authorizations bombing
britain british cheerfully claiming constitution curtail december defeats defending delays democratic dictators disclose

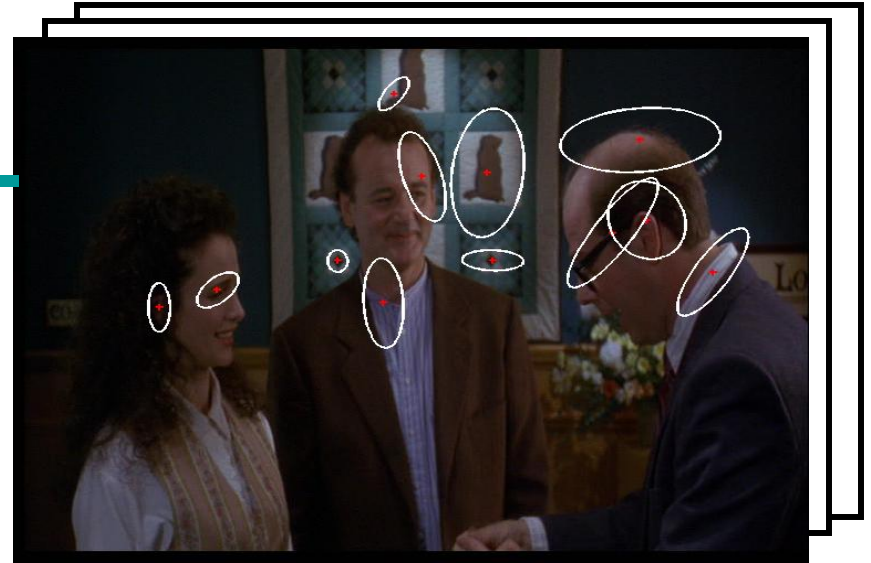
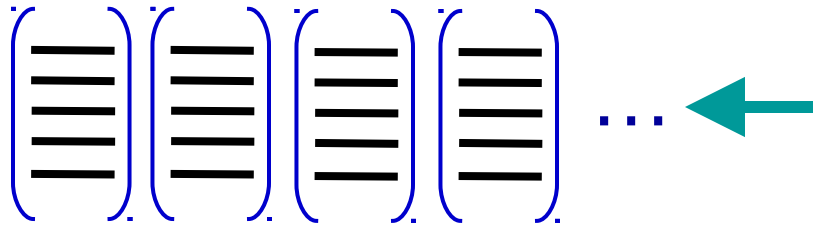
economic empire endanger **facts** false forgotten fortunes france **freedom** fulfilled fullness fundamental gangsters
german germany **god** guam harbor hawaii **hemisphere** hint hitler hostilities immune improving indies innumerable

invasion **islands** isolate **japanese** labor metals midst midway **navy** nazis obligation offensive
officially **pacific** partisanship patriotism pearl peril perpetrated perpetual philippine preservation privilege reject
repaired **resisting** retain revealing rumors seas soldiers speaks speedy stamina **strength** sunday sunk supremacy tanks taxes

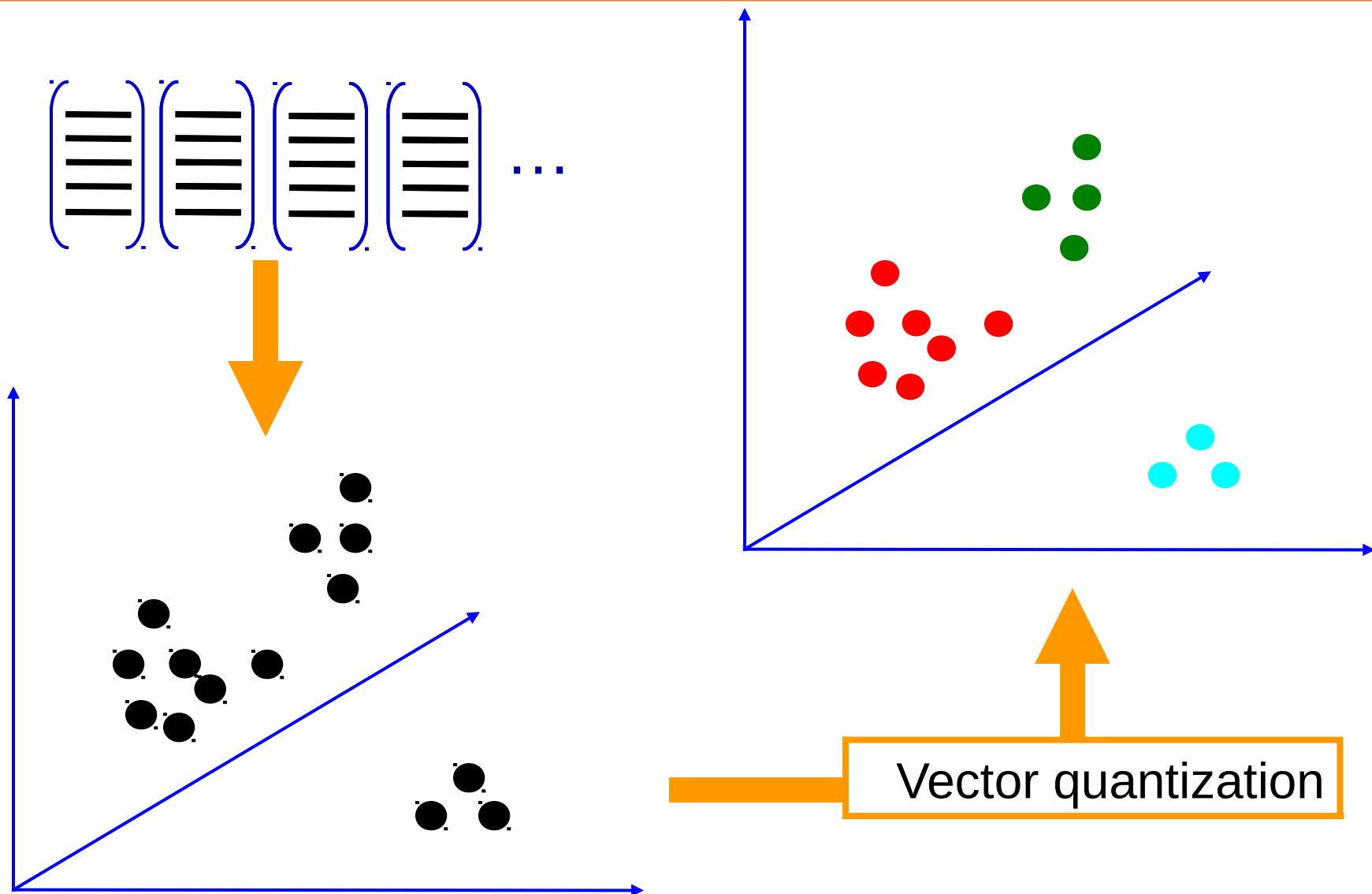
treachery true tyranny undertaken victory **war** wartime washington

- Orderless document representation:
frequencies of words from a dictionary Salton &

- Interest Point Features



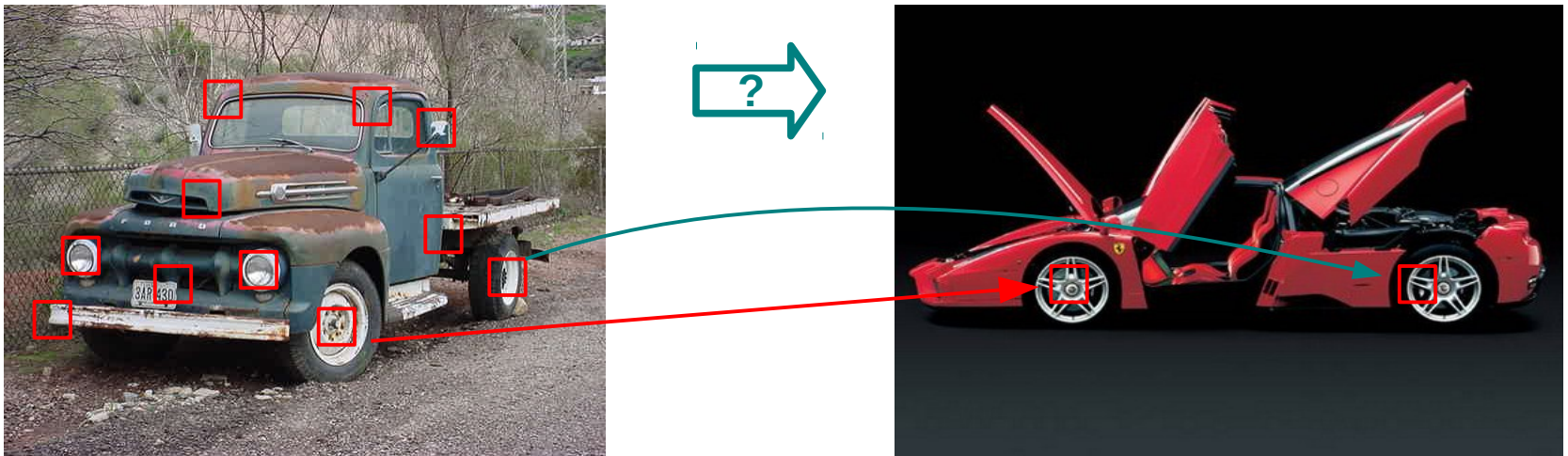
Clustering



Vector quantization

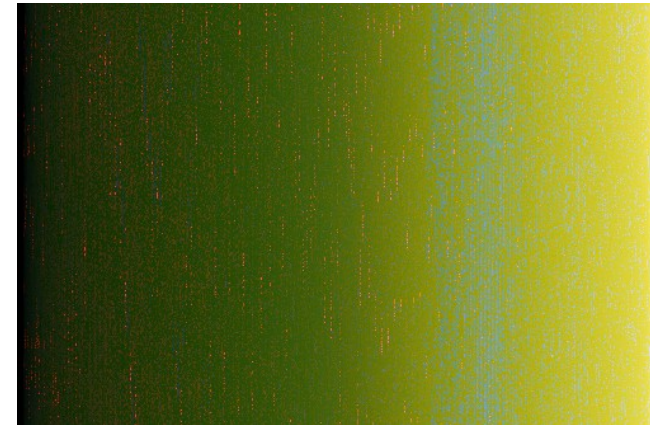
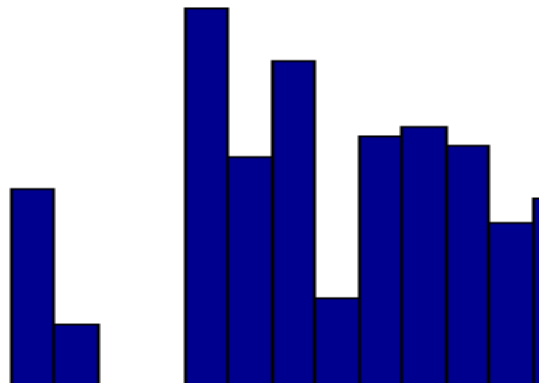
Recall: SIFT Matching with RANSAC

SIFT + RANSAC transformation matching to exemplars was not likely to work



How is SIFT-based classification through Bag of Words different?

The (obvious) problem with ignoring Geometry



All of these images have the same color histogram

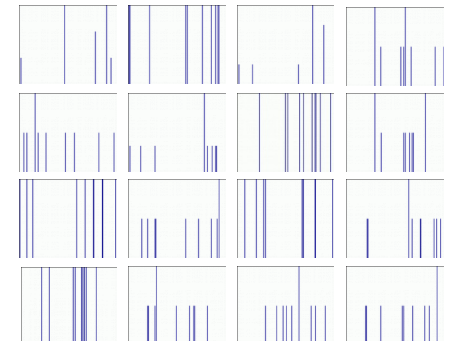
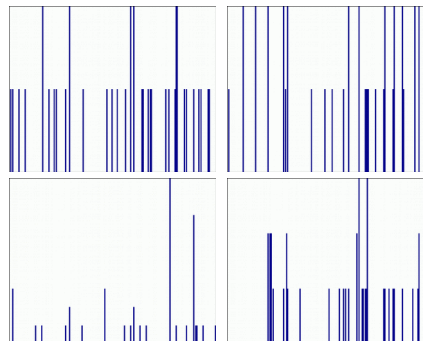
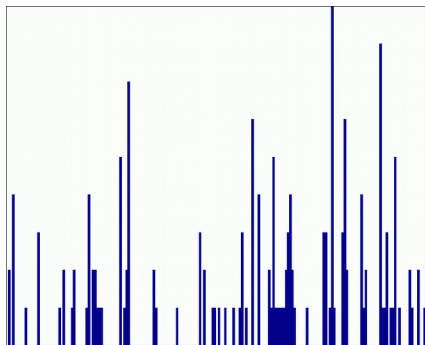
Adding Geometry back: Spatial pyramid



Compute histogram in each spatial bin

Spatial pyramid pooling

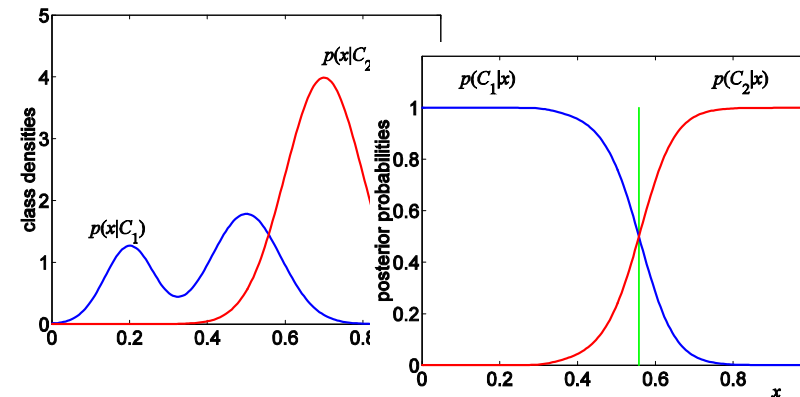
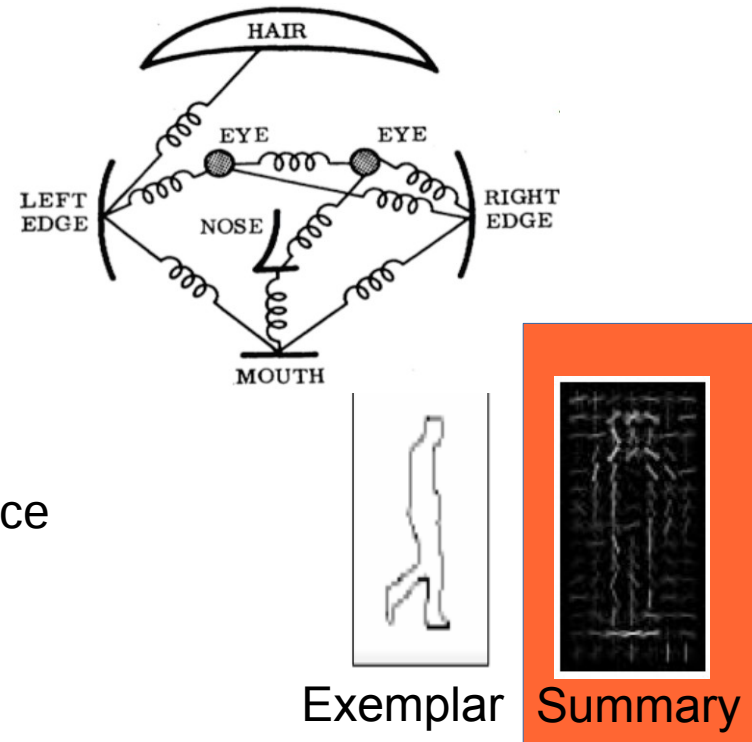
- Extension of a bag of features
- Locally orderless representation at several levels of resolution



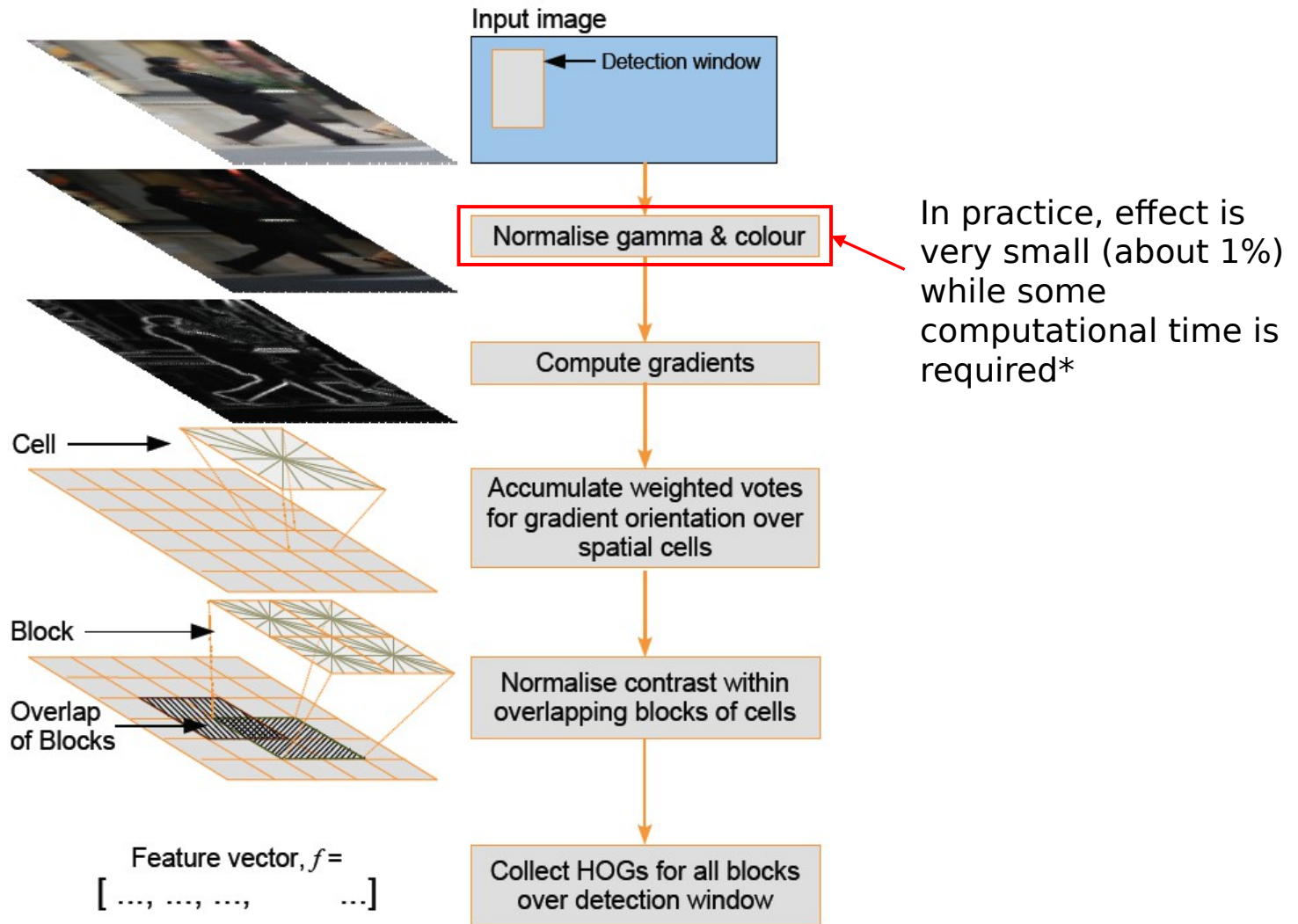
Lazebnik, Schmid & Ponce (CVPR 2006)

Object Classification technique: Delal & Triggs

- **Hypothesis** generation
 - **Sliding window**, Segmentation, feature point detection, random, search
- **Encoding** of (local) image data
 - Colors, Edges, Corners, **Histogram of Oriented Gradients**, Wavelets, Convolution Filters
- **Relationship** of different parts to each other
 - Blur or **histogram**, Tree/Star, Pairwise/Covariance
- **Learning** from labeled examples
 - Selecting representative examples (templates), Clustering, Building a cascade
 - Classifiers: K-NN, **Logistic regression**, **SVM**, **AdaBoost**, **Bayes ...**
 - Generative vs. Discriminative
- **Verification** - removing redundant, overlapping, incompatible examples
 - Non-Max Suppression, context priors, geometry

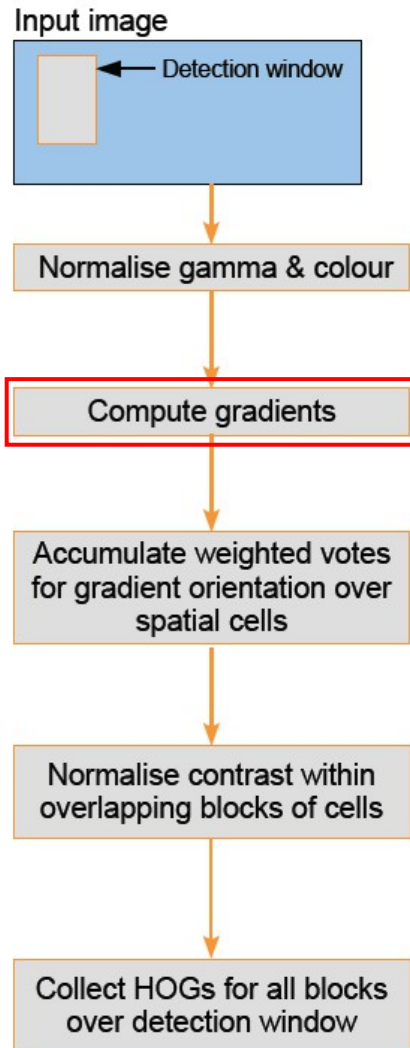


Dalal and Trigs Pedestrian Detector



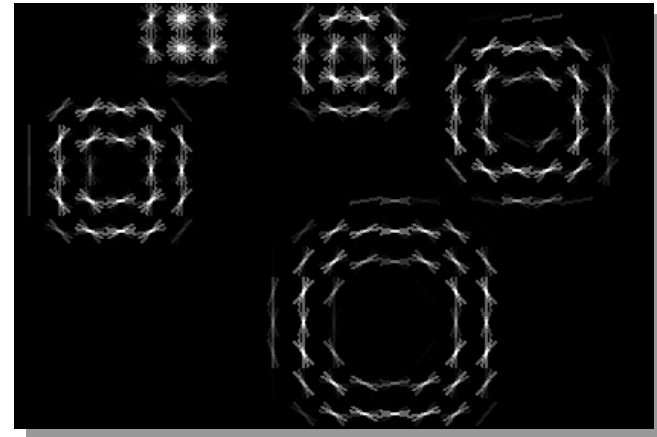
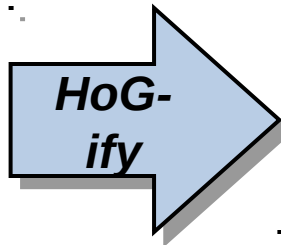
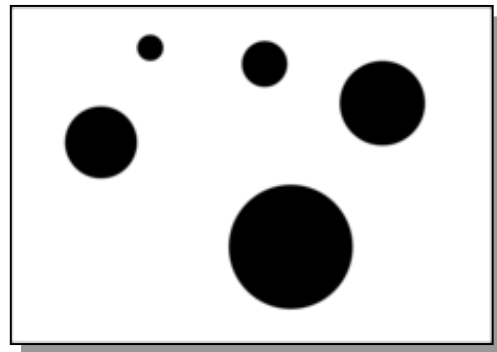
*Navneet Dalal and Bill Triggs. Histograms of Oriented Gradients for Human Detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, SanDiego, USA, June 2005. Vol. II, pp. 886-893.

Computing gradients

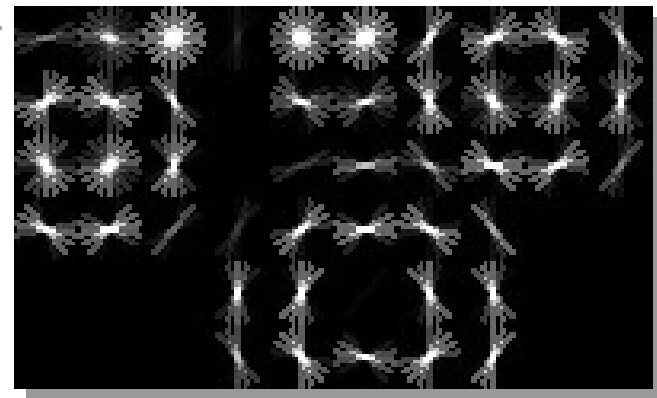


Mask Type	1D centered	1D uncentered	1D cubic,corrected	2x2 diagonal	3x3 Sobel
Operator	$\begin{bmatrix} -1, 0, \\ 1 \end{bmatrix}$	$[-1, 1]$	$[1, -8, 0, 8, -1]$	$\begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}$ $\begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix}$	$\begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}$ $\begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}$
Miss rate at 10^{-4} FPPW	11%	12.5%	12%	12.5%	14%

Histogram of Oriented Gradients



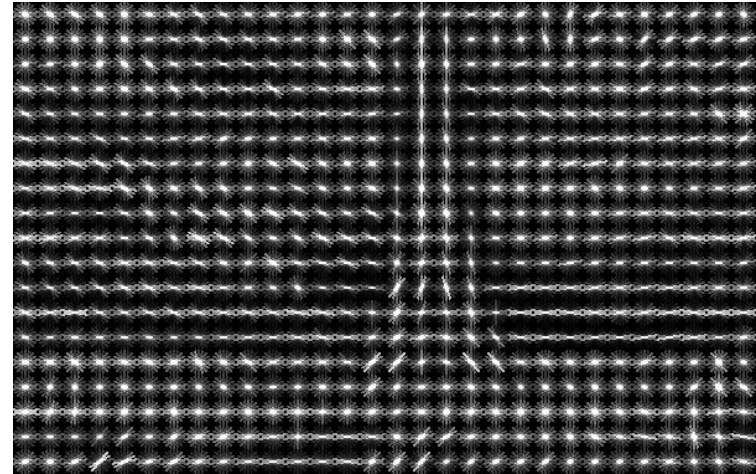
10x10 cells



20x20
cells

[Dalal and Triggs, CVPR 2005]

Histogram of Oriented Gradients



HOG feature vector for one block



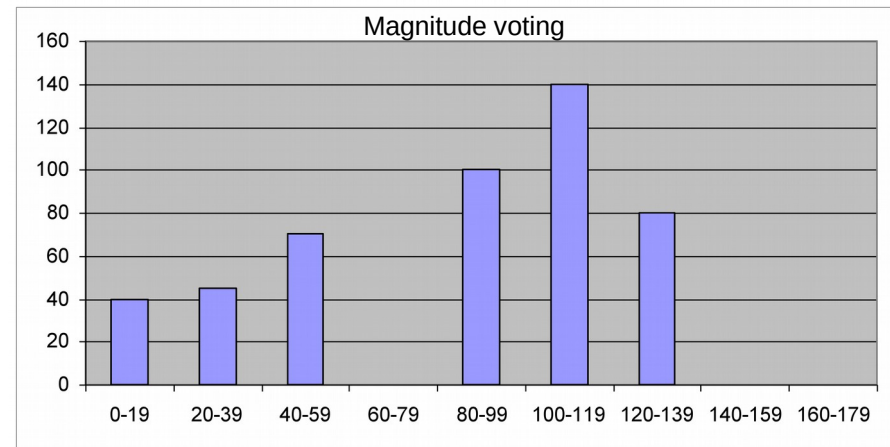
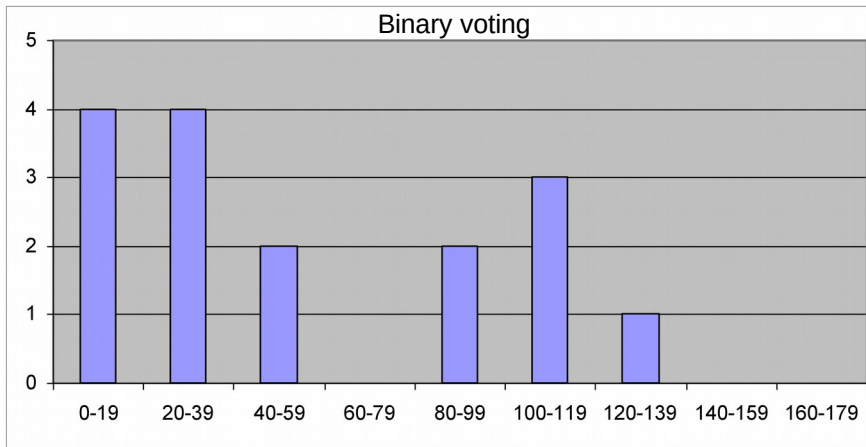
$$f = (h_1^1, \dots, h_9^1, \boxed{h_1^2, \dots, h_9^2}, h_1^3, \dots, h_9^3, h_1^4, \dots, h_9^4)$$

Angle

0	15	25	25
10	15	25	30
45	95	101	110
47	97	101	120

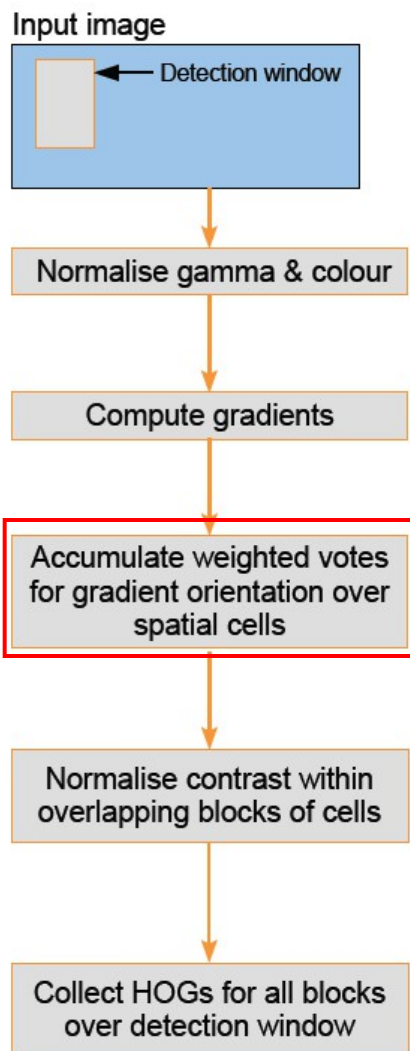
Magnitude

5	20	20	10
5	10	10	5
20	30	30	40
50	70	70	80

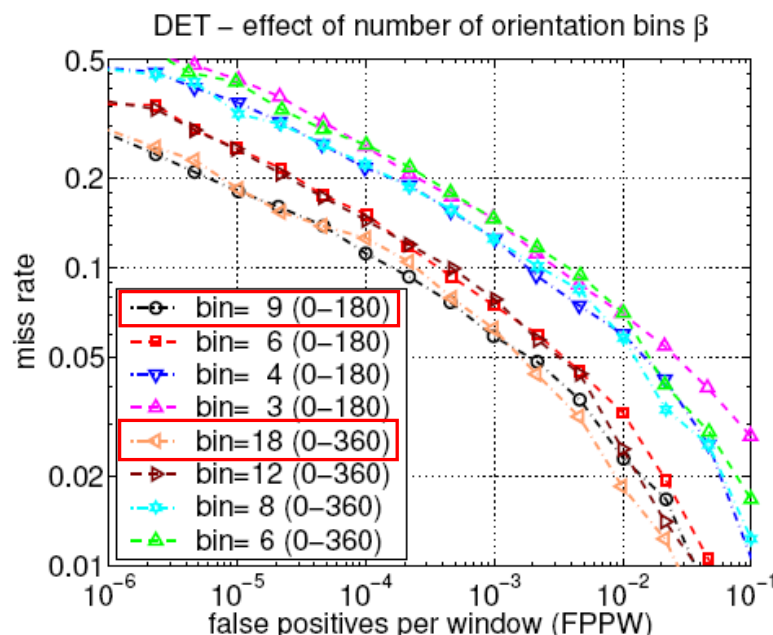


Feature vector extends while window moves

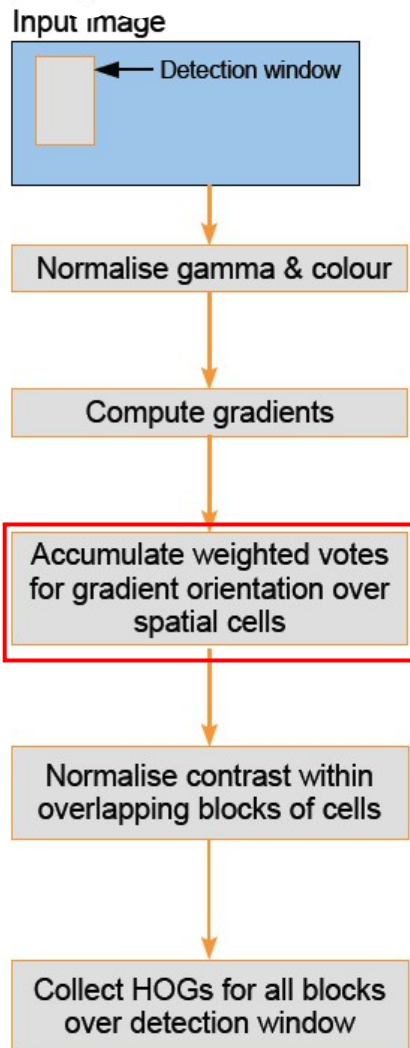
Accumulate weight votes over spatial cells



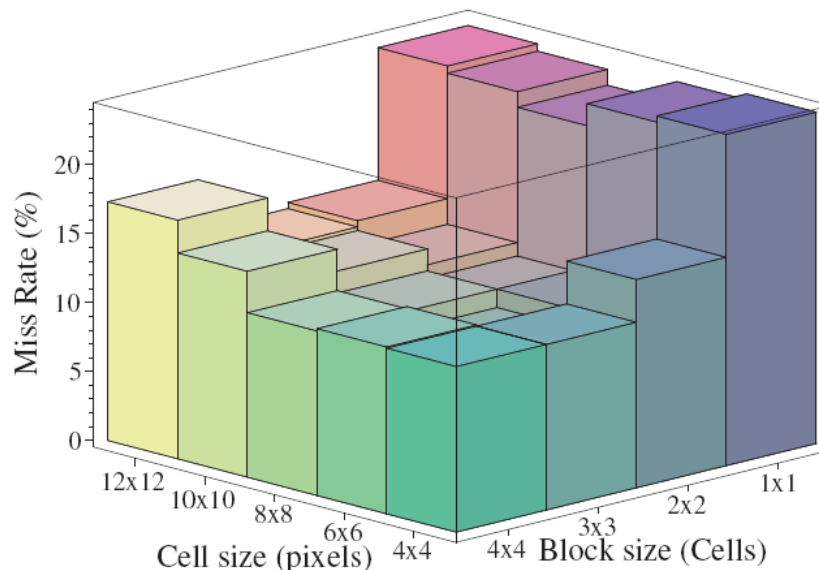
- How many bins should be in histogram?
- Should we use oriented or non-oriented gradients?
- How to select weights?
- Should we use overlapped blocks or not? If yes, then how big should be the overlap?
- What block size should we use?



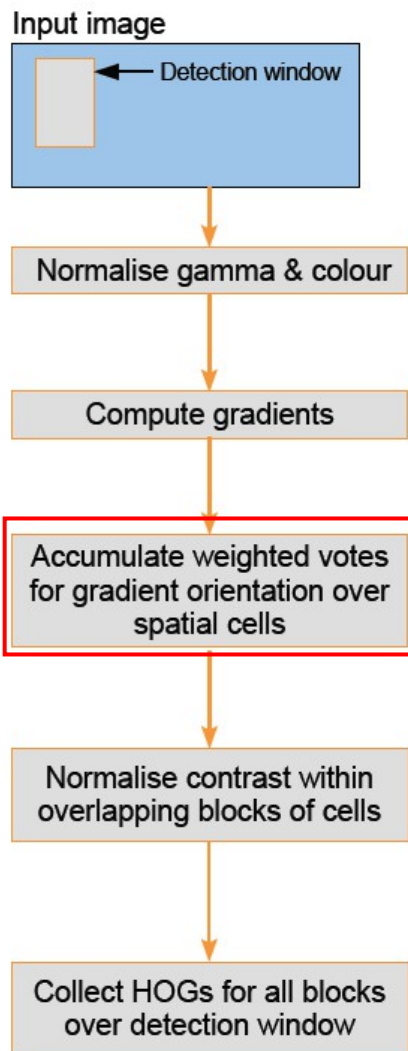
Accumulate weight votes over spatial cells



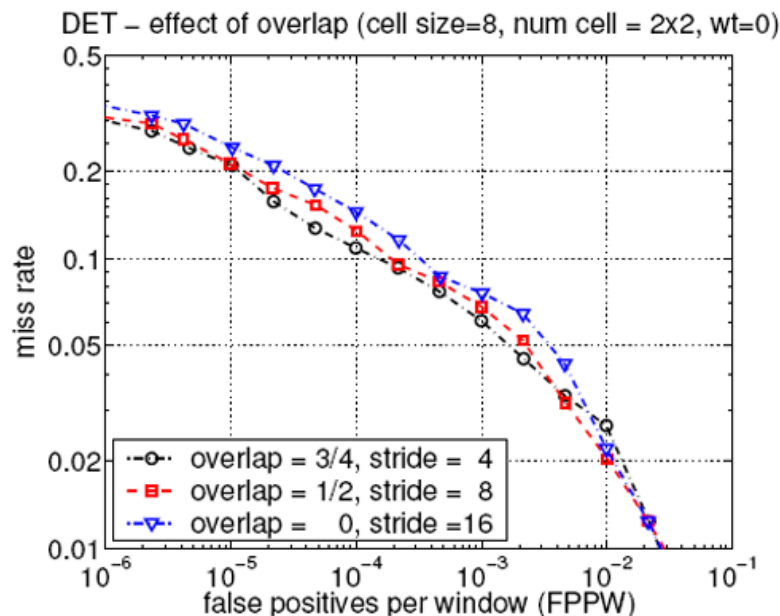
- How many bins should be in histogram?
- Should we use oriented or non-oriented gradients?
- How to select weights?
- Should we use overlapped blocks or not? If yes, then how big should be the overlap?
- What block size should we use?



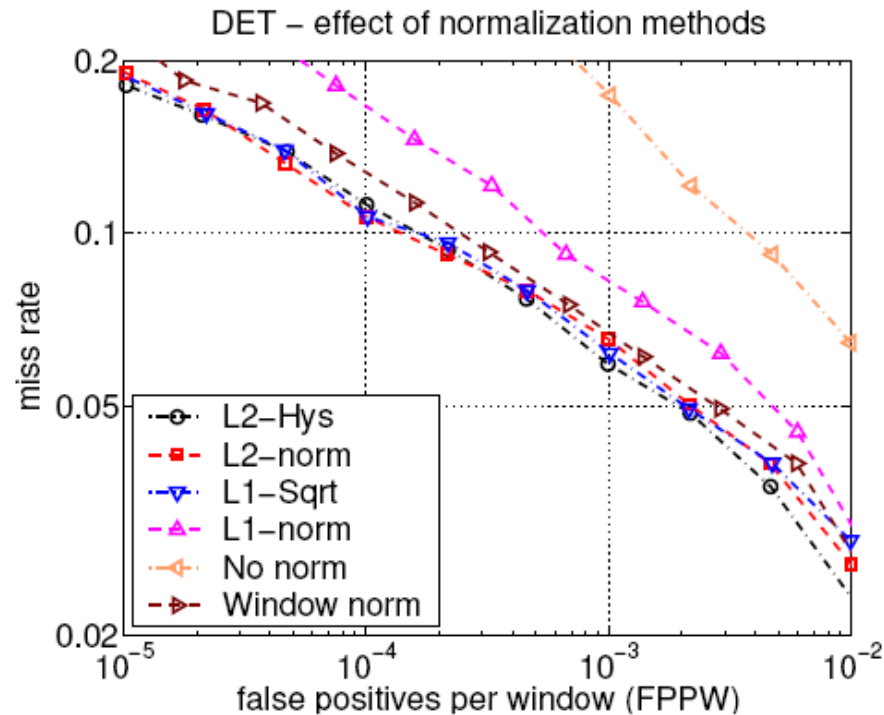
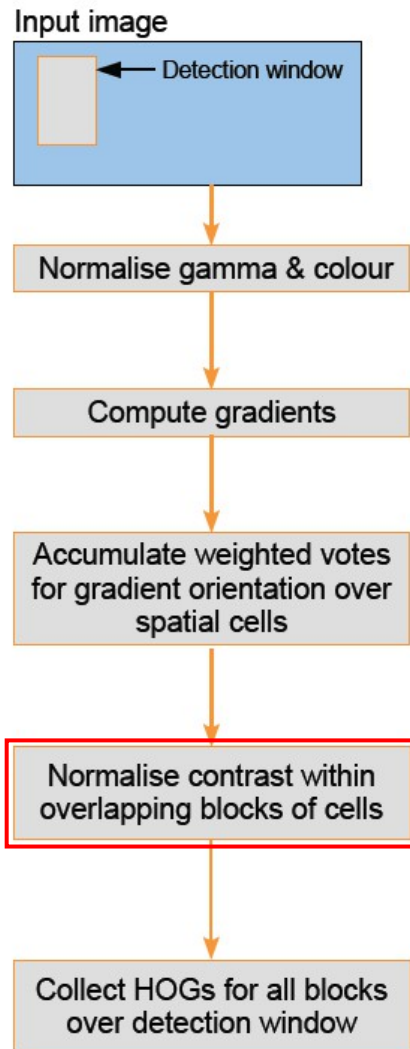
Accumulate weight votes over spatial



- How many bins should be in histogram?
- Should we use oriented or non-oriented gradients?
- How to select weights?
- Should we use overlapped blocks or not? If yes, then how big should be the overlap?
- What block size should we use?



Contrast normalization



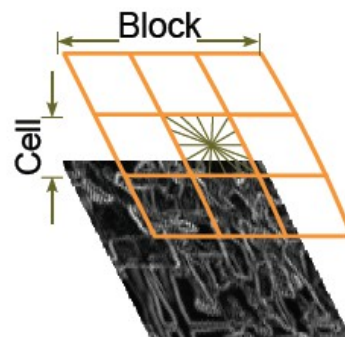
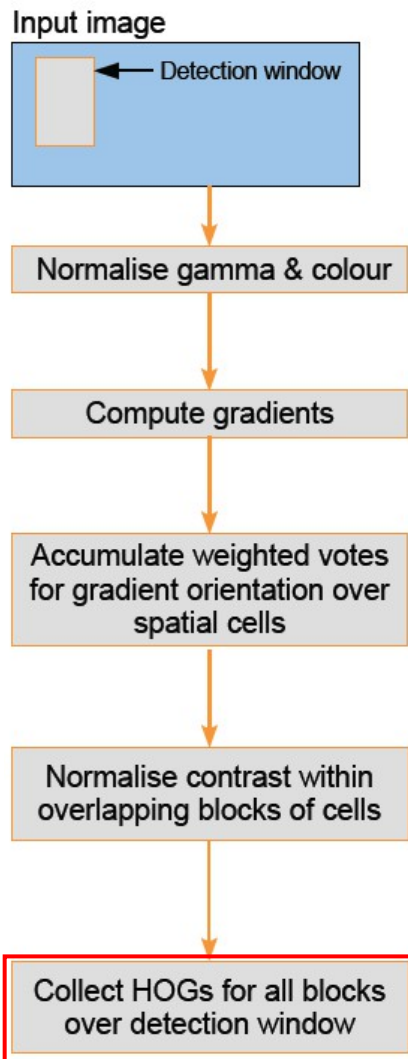
$$L1 - norm = \frac{v}{\|v\|_1 + \epsilon}$$

$$L1 - sqrt = \sqrt{\frac{v}{\|v\|_1 + \epsilon}}$$

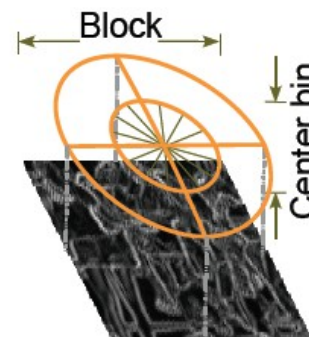
$$L2 - norm = \frac{v}{\sqrt{\|v\|_2^2 + \epsilon}}$$

L2 - Hys - L2-norm followed by clipping (limiting the maximum values of v to 0.2) and renormalising

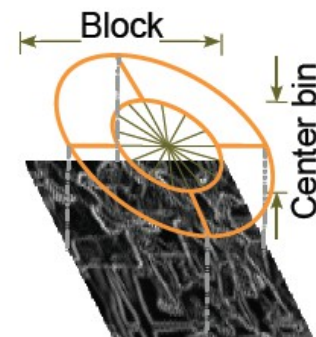
Making the feature vector



(a) R-HOG / SIFT



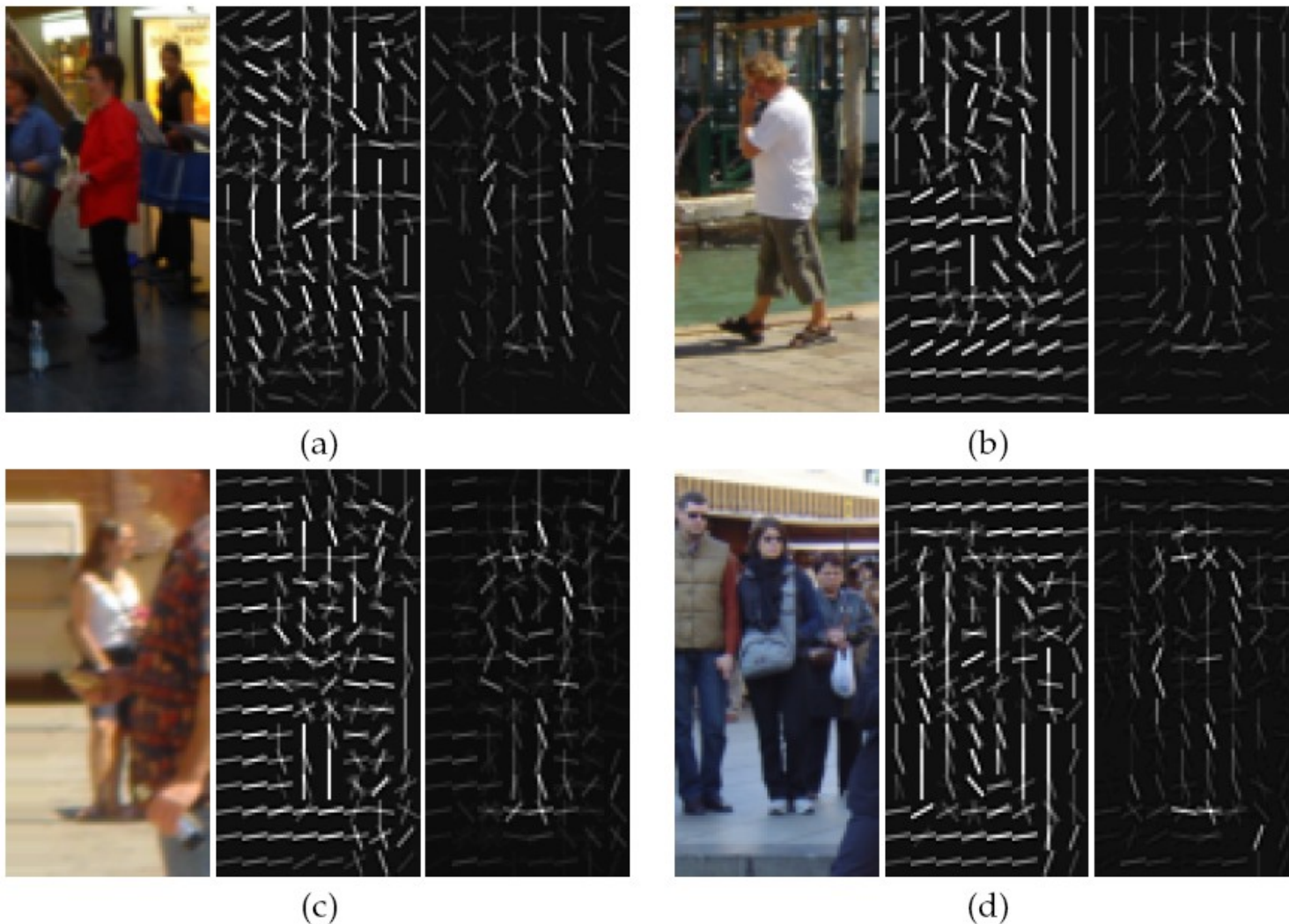
(b) C-HOG



(c) Single centre C-HOG

Variants of HOG descriptors. (a) A rectangular HOG (R-HOG) descriptor with 3×3 blocks of cells. (b) Circular HOG (C-HOG) descriptor with the central cell divided into angular sectors as in shape contexts. (c) A C-HOG descriptor with a single central cell.

HOG example



In each triplet: (1) the input image, (2) the corresponding R-HOG feature vector (only the dominant orientation of each cell is shown), (3) the dominant orientations selected by the SVM (obtained by multiplying the feature vector by the corresponding weights from the linear SVM).

Feature Vector → Classification Result

	<i>Supervised Learning</i>	<i>Unsupervised Learning</i>
<i>Discrete</i>	<div>classification or categorization</div>	clustering
<i>Continuous</i>	regression	dimensionality reduction

The machine learning framework

$$\hat{y} = f(\mathbf{x})$$

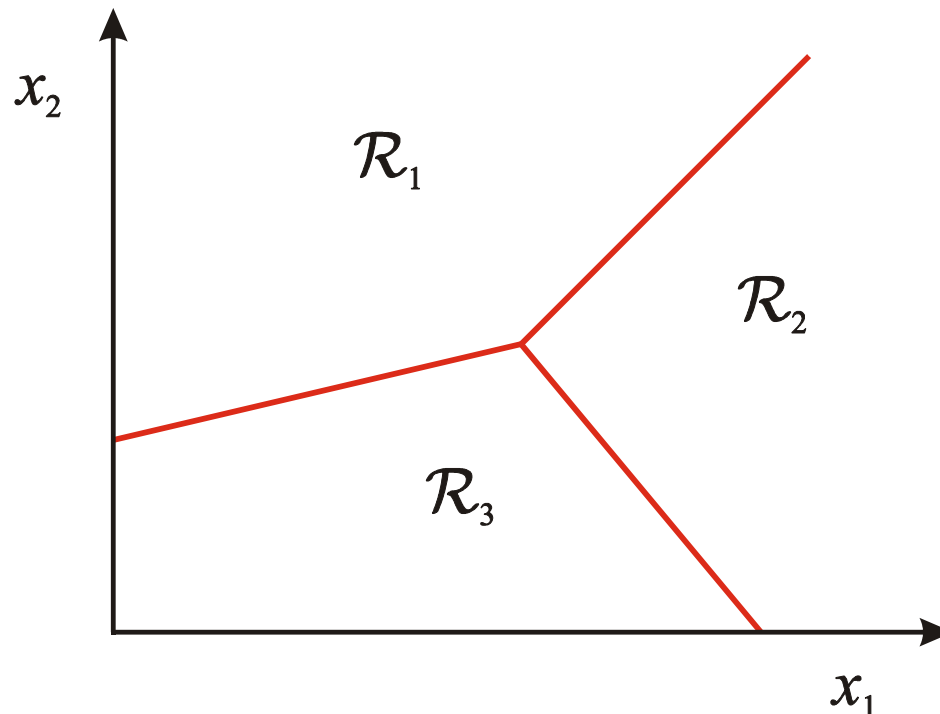
output prediction function Image features

The diagram illustrates the machine learning framework equation $\hat{y} = f(\mathbf{x})$. Three red arrows point from labels below to components of the equation: one from 'output' to \hat{y} , one from 'prediction function' to f , and one from 'Image features' to \mathbf{x} .

- **Training:** given a *training set* of labeled examples $\{(\mathbf{x}_1, y_1), \dots, (\mathbf{x}_N, y_N)\}$, estimate the prediction function f by minimizing the prediction error on the training set
- **Testing:** apply f to a never before seen *test example* \mathbf{x} and output the predicted value $y = f(\mathbf{x})$

Classification

- Assign input vector to one of two or more classes
- Any decision rule divides input space into *decision regions* separated by *decision boundaries*

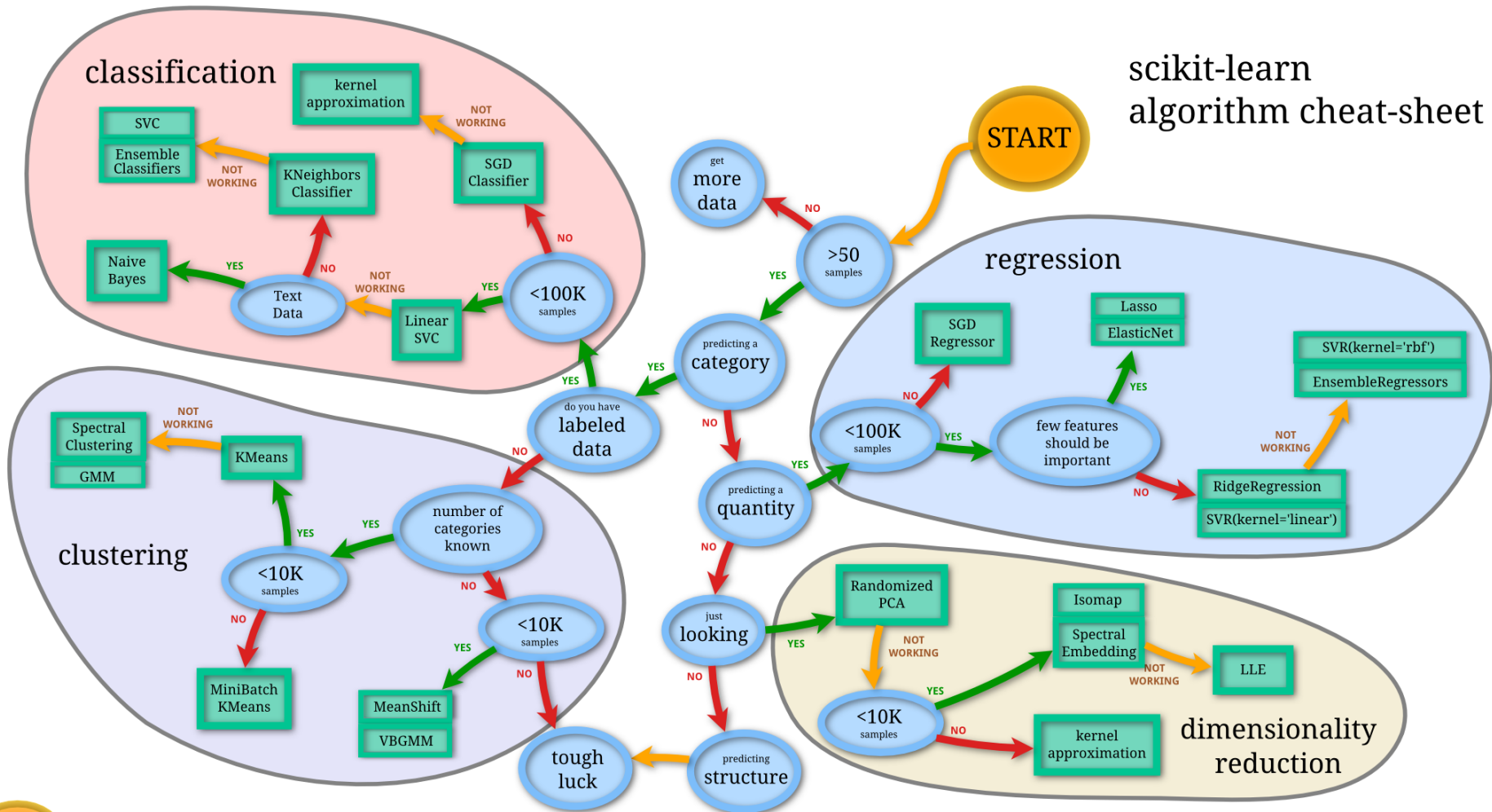


Many classifiers to choose from

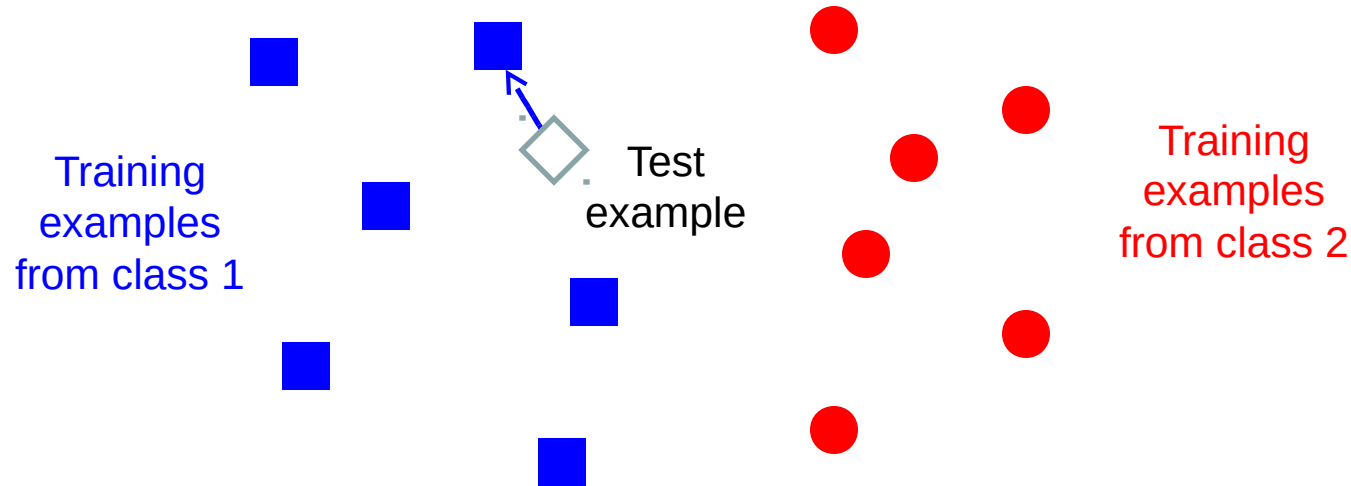
- SVM
- Neural networks
- Naïve Bayes
- Bayesian network
- Logistic regression
- Randomized Forests
- Boosted Decision Trees
- K-nearest neighbor
- RBMs
- Etc.

Which is the best one?

Which Algorithm to use?



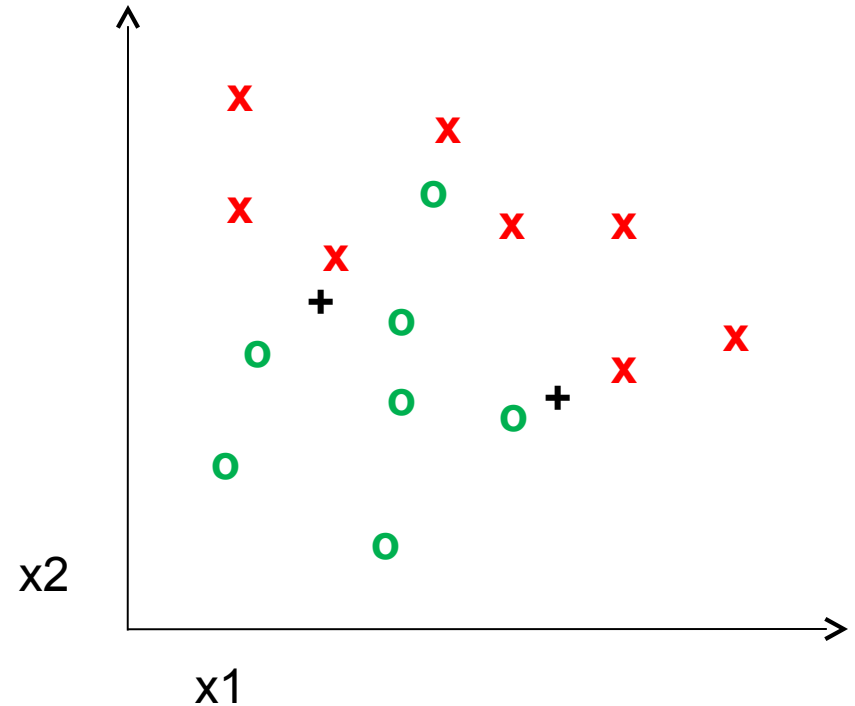
(1-) Nearest Neighbor



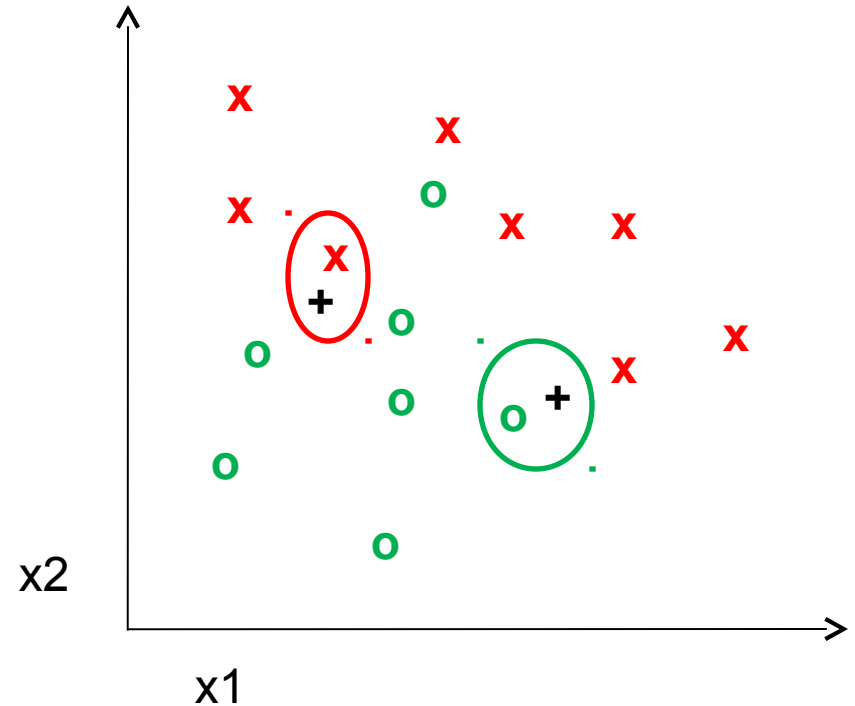
$f(\mathbf{x}) = \text{label of the training example nearest to } \mathbf{x}$

- All we need is a distance function for our inputs
- No training required!

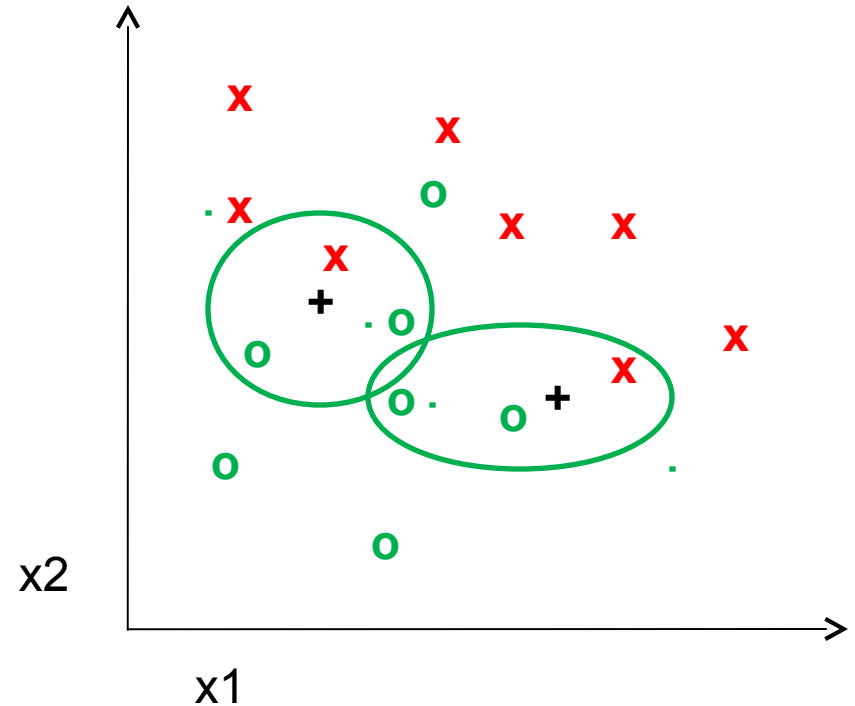
K-nearest neighbor



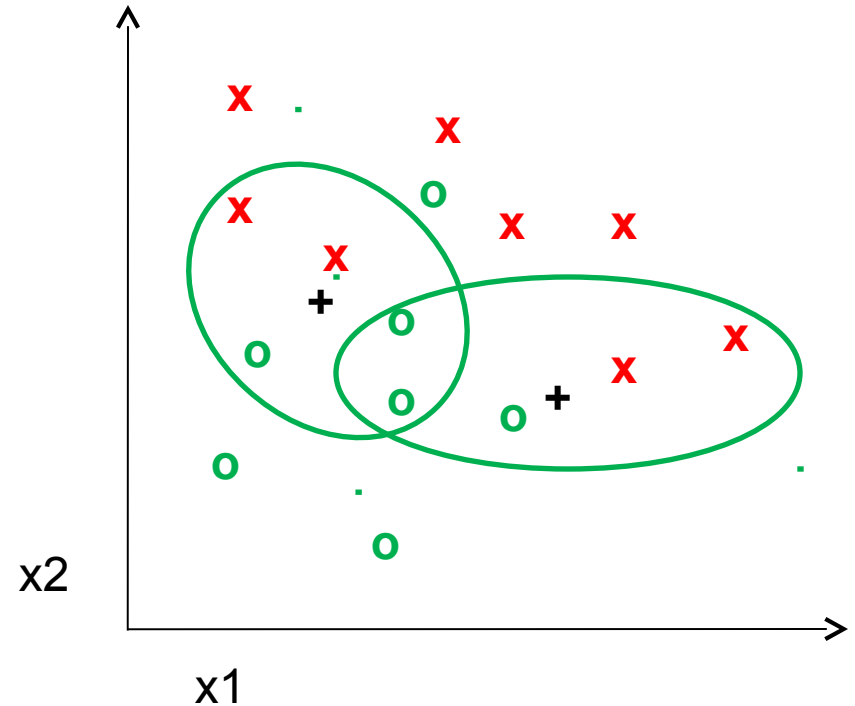
1-nearest neighbor



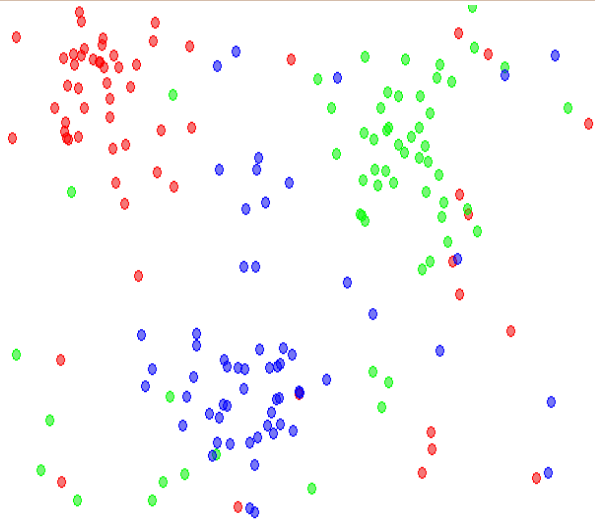
3-nearest neighbor



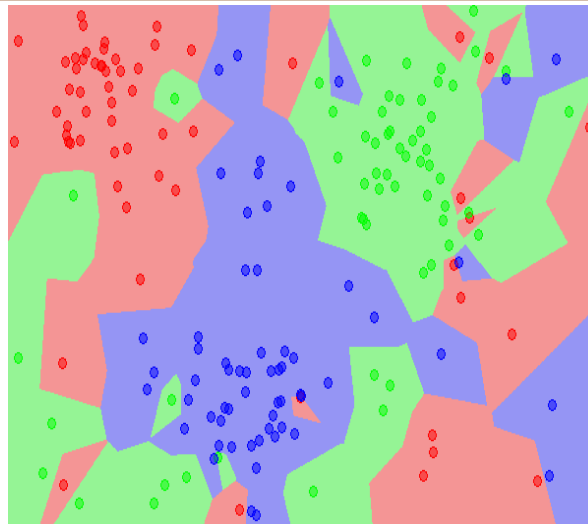
5-nearest neighbor



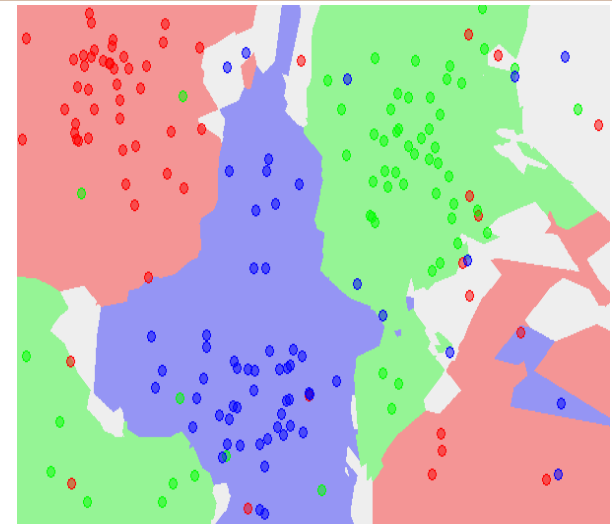
K-NN Classifiers



the data



1-NN classifier

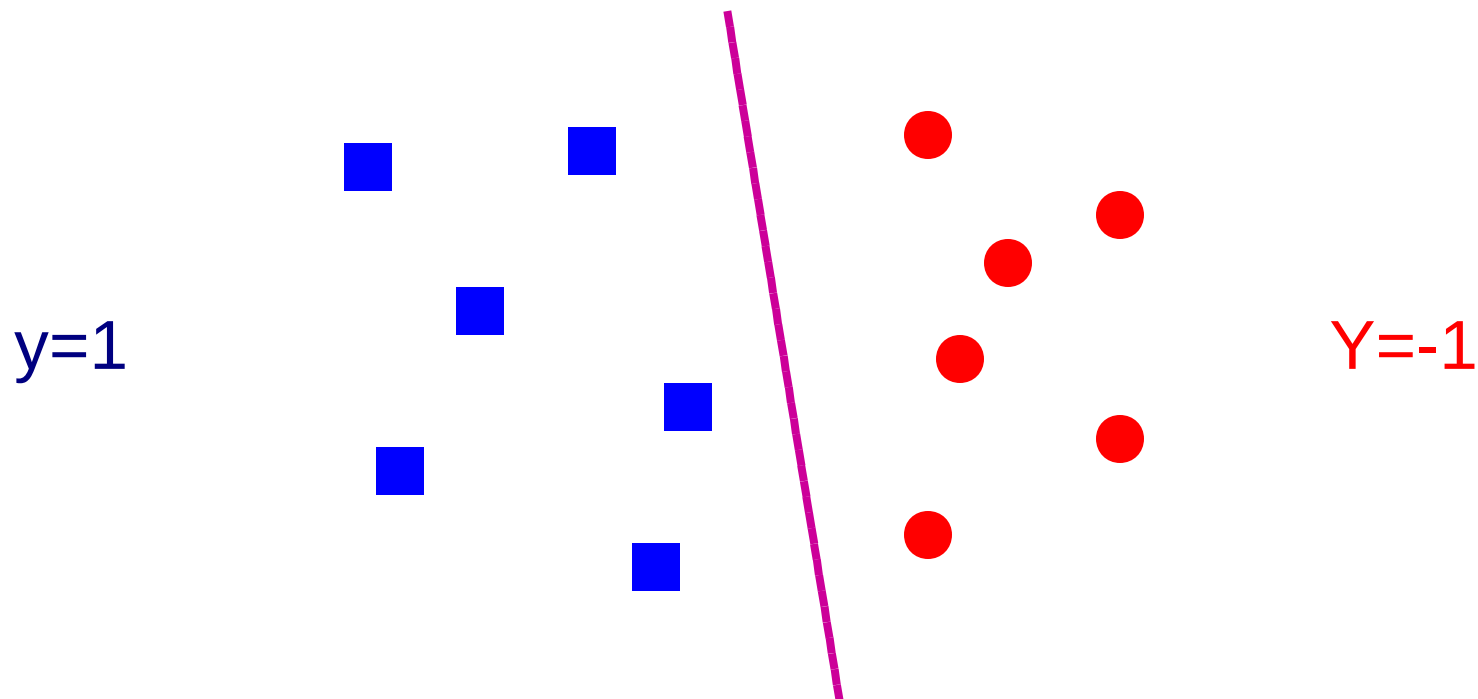


5-NN classifier

Questions:

- What distance function to use L1, L2?
- What is the accuracy of the 1-NN classifier on the training data?
- What is the accuracy of the 5-NN classifier on the training data?
- Which one do expect to do better on the test data?

Classifiers: Linear

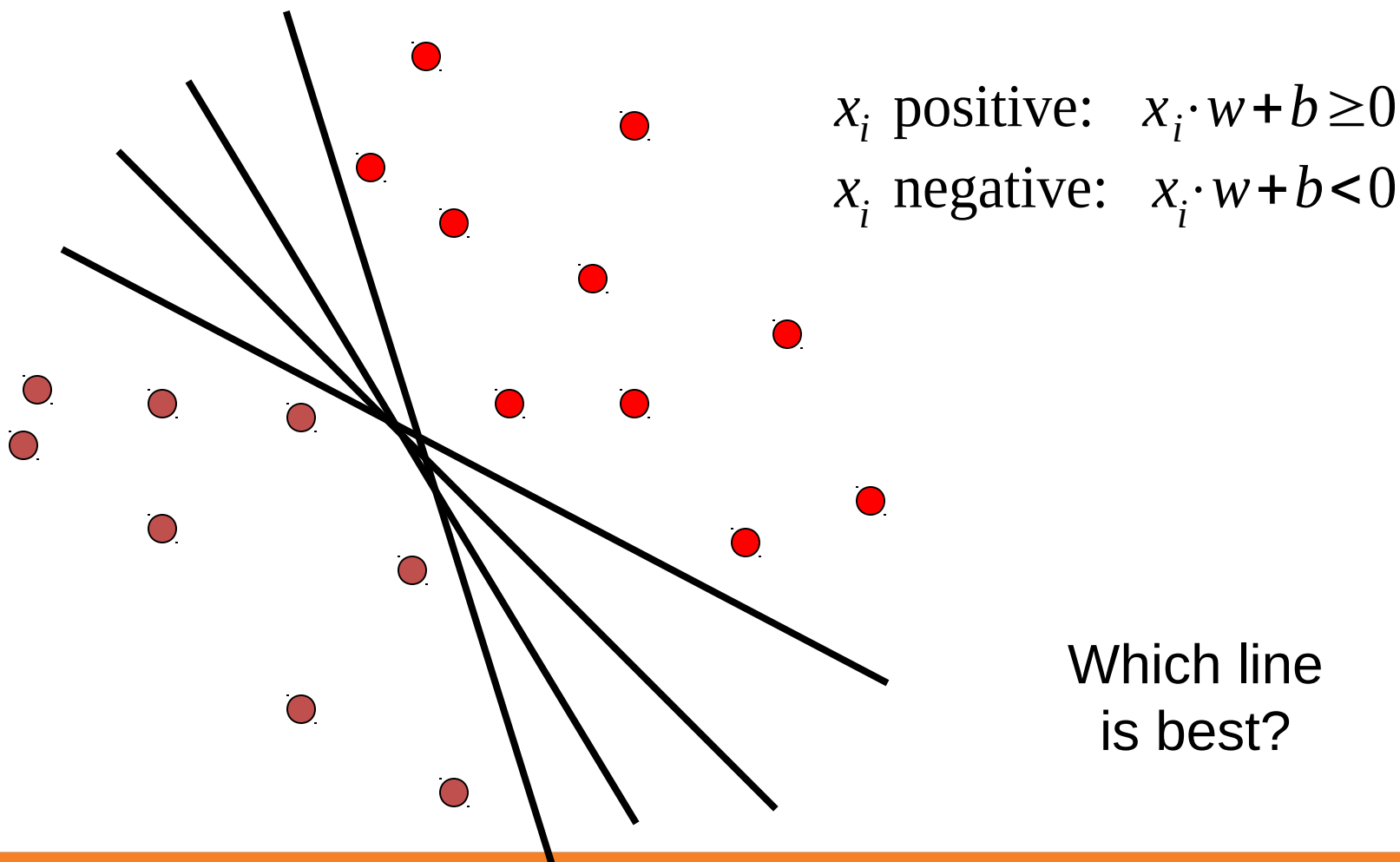


- Find a *linear function* to separate the classes:

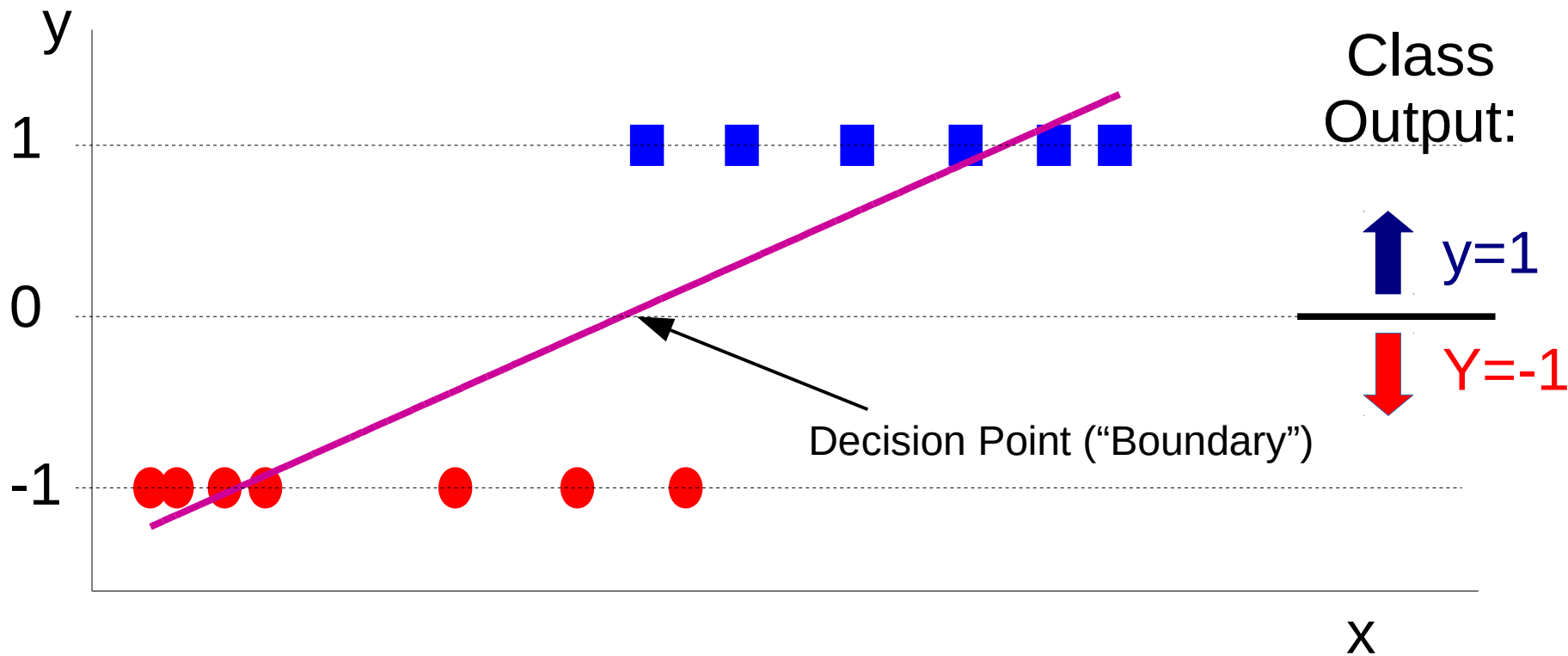
$$\hat{y}=f(\mathbf{x}) = \text{sgn}(\mathbf{w} \cdot \mathbf{x} + b)$$

Linear classifiers

- Find linear function to separate positive and negative examples



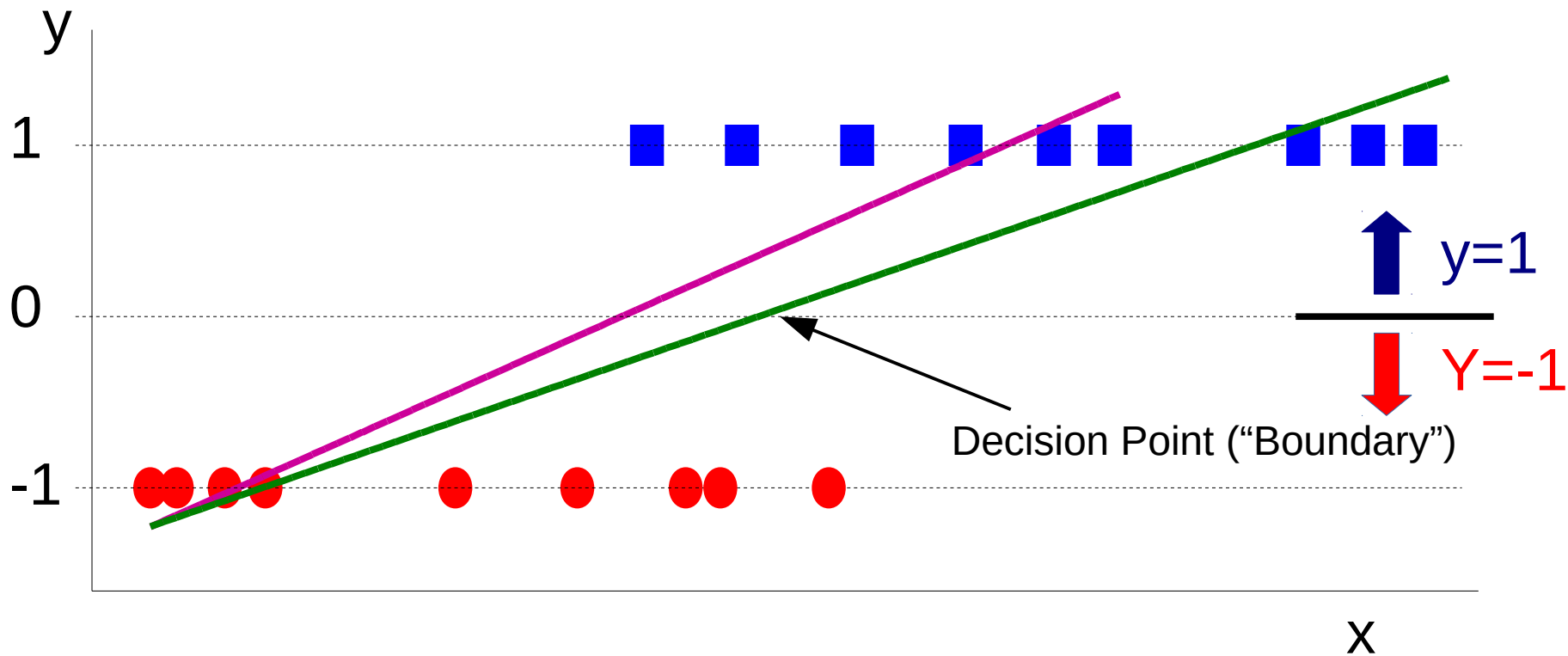
Using Least Squares for Classification



- Find a *linear function* to separate the classes:

$$\hat{y}=f(\mathbf{x}) = \text{sgn}(\mathbf{w} \cdot \mathbf{x} + b)$$

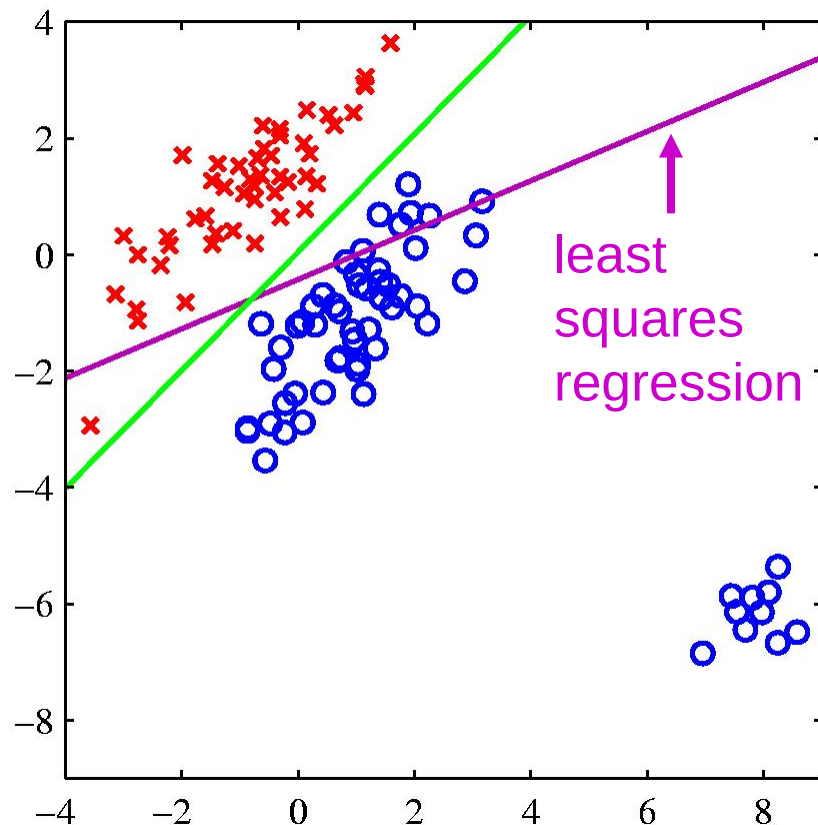
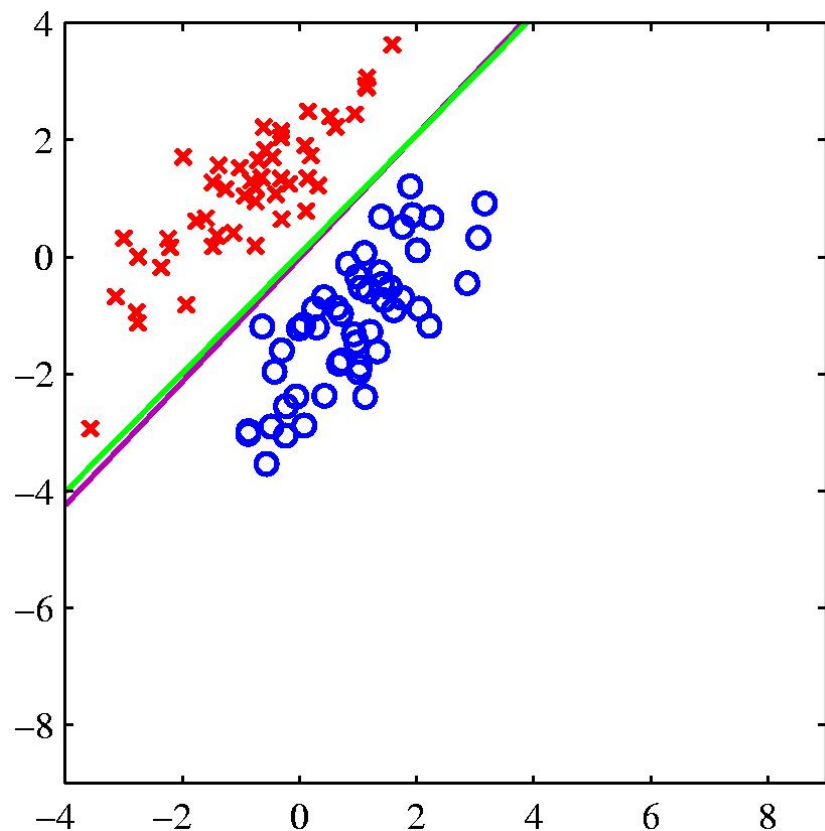
Using Least Squares for Classification



- Find a *linear function* to separate the classes:

$$\hat{y} = f(\mathbf{x}) = \text{sgn}(\mathbf{w} \cdot \mathbf{x} + b)$$

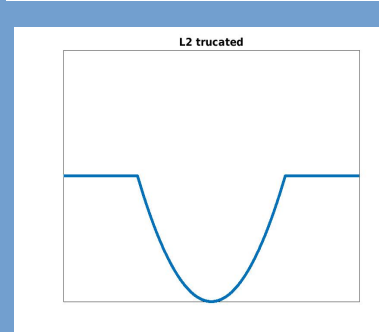
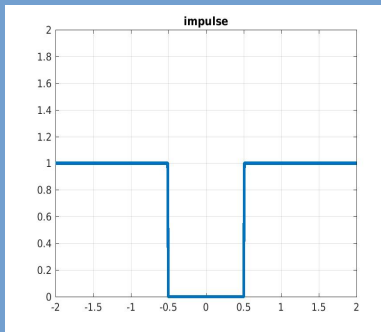
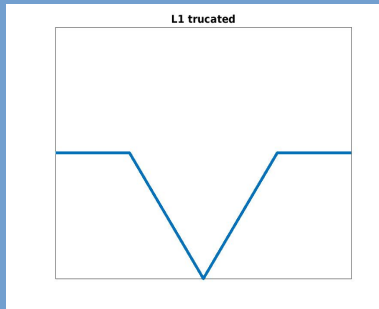
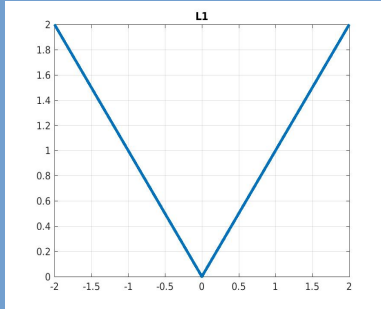
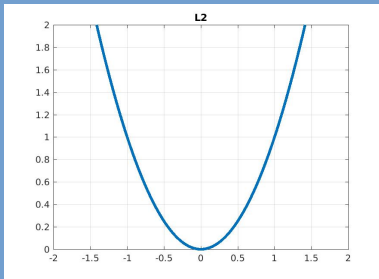
Using least squares for classification



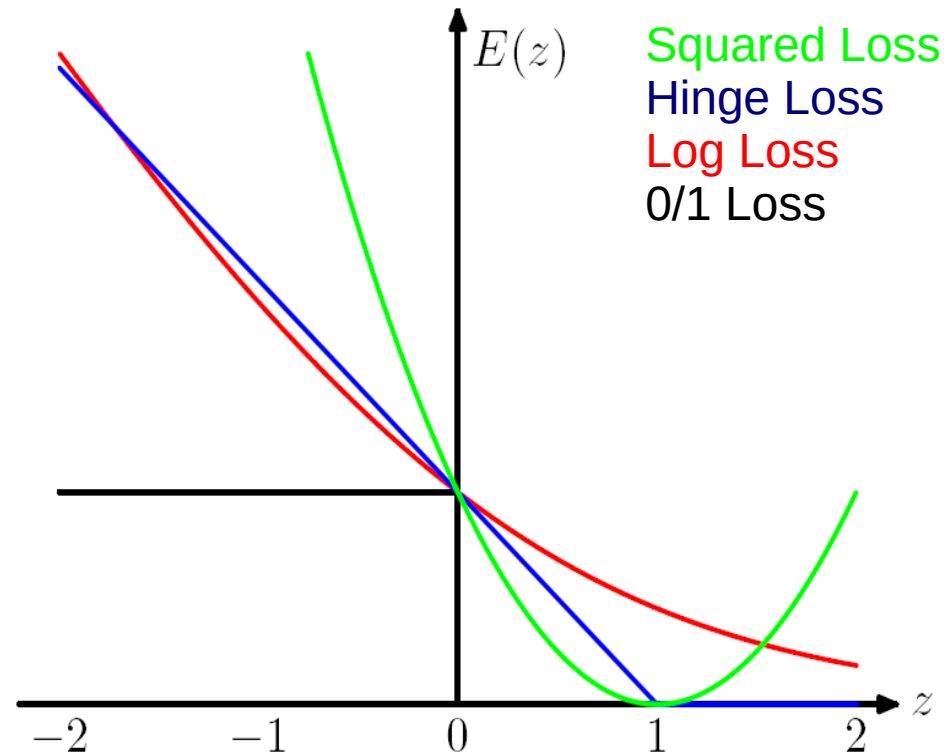
If the right answer is 1 and the model says 1.5, it loses, so it changes the boundary to avoid being “too correct”

The Problem: Loss Function

Recall: Regression Loss Functions



Some Classification Loss Functions



Sigmoid

We model the probability of a label Y to be equal $y \in \{-1, 1\}$, given a data point $x \in \mathbf{R}^n$, as:

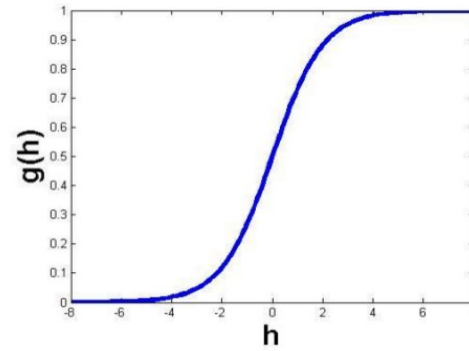
$$P(Y = y \mid x) = \frac{1}{1 + \exp(-y(w^T x + b))}.$$

This amounts to modeling the *log-odds ratio* as a linear function of X :

$$\log \frac{P(Y = 1 \mid x)}{P(Y = -1 \mid x)} = w^T x + b.$$

$$\frac{e^{-h}}{1 + e^{w^T x}}$$

binary



- ▶ The decision boundary $P(Y = 1 \mid x) = P(Y = -1 \mid x)$ is the hyperplane with equation $w^T x + b = 0$.
- ▶ The region $P(Y = 1 \mid x) \geq P(Y = -1 \mid x)$ (i.e., $w^T x + b \geq 0$) corresponds to points with predicted label $\hat{y} = +1$.

Log Loss

The likelihood function is

$$l(w, b) = \prod_{i=1}^m \frac{1}{1 + e^{-y_i(w^T x_i + b)}}.$$

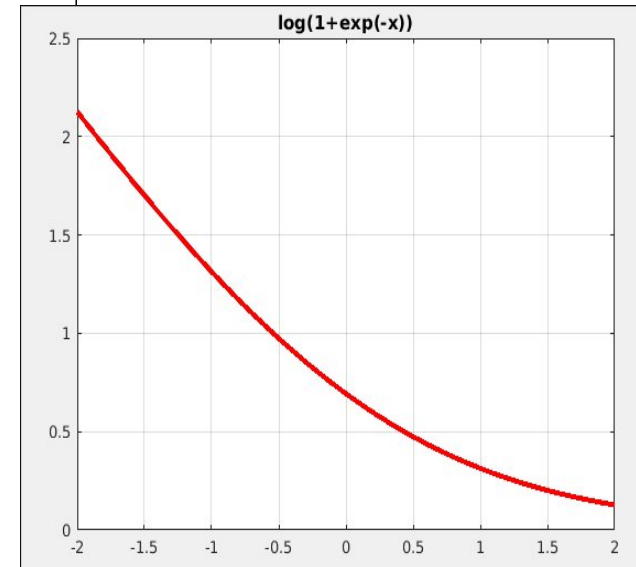
Now maximize the log-likelihood:

$$\max_{w, b} L(w, b) := - \sum_{i=1}^m \log(1 + e^{-y_i(w^T x_i + b)})$$

In practice, we may consider adding a regularization term

$$\max_{w, b} L(w, b) + \lambda r(w),$$

with $r(w) = \|w\|_2^2$ or $r(x) = \|w\|_1$.



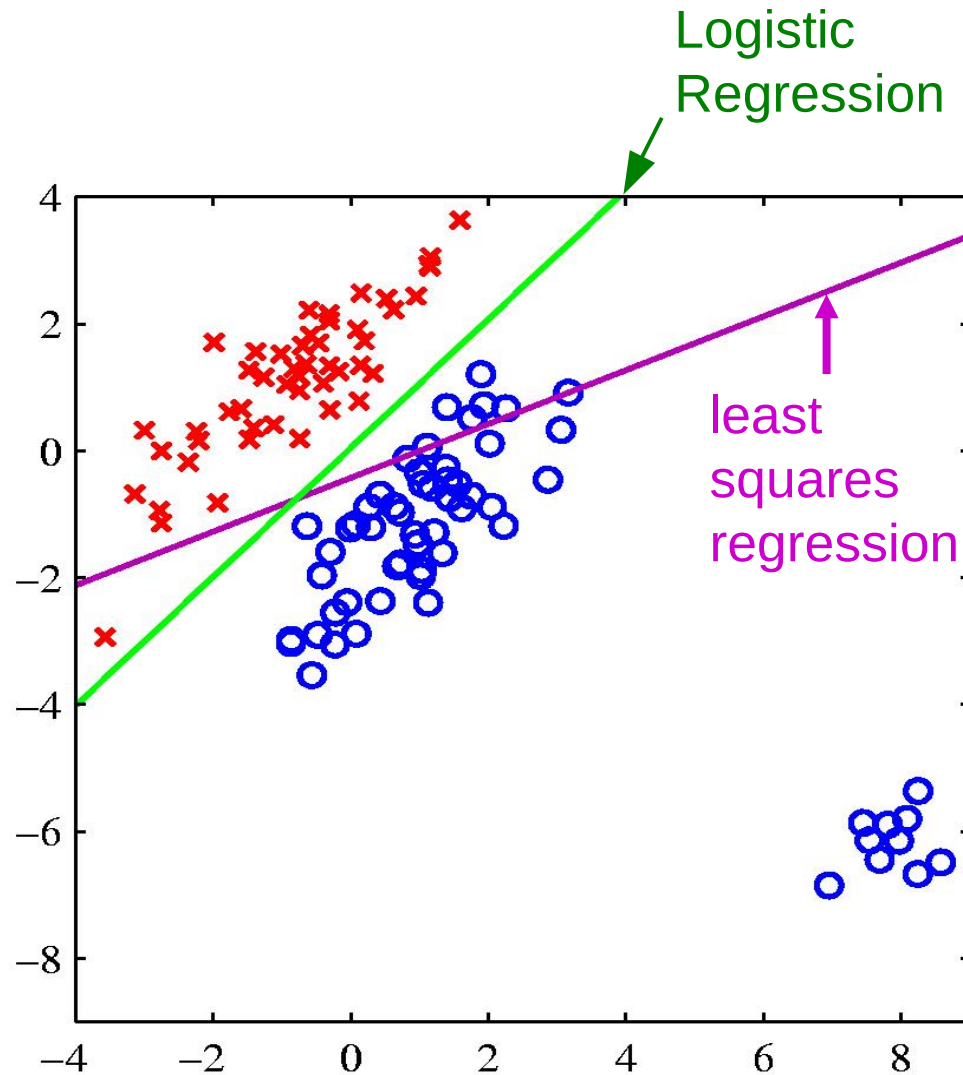
The Logistic Loss:

$$\log(1 + \exp(-z))$$

Where

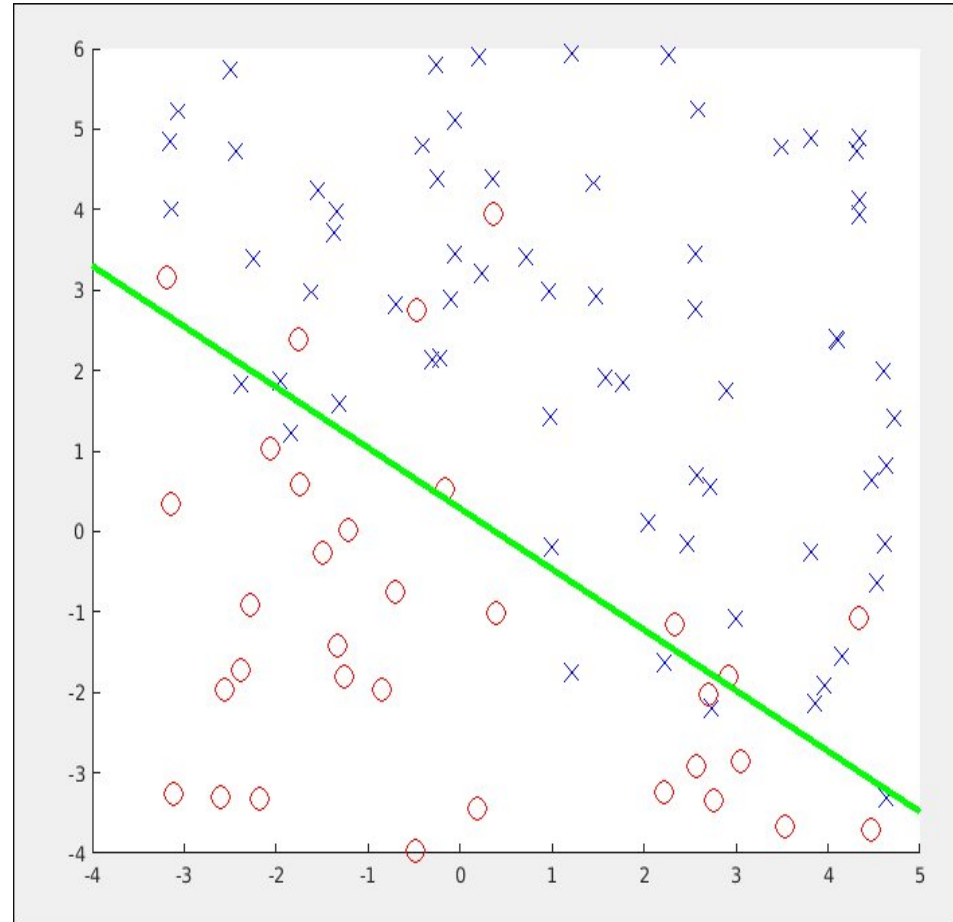
$$z = y^*(w^T x + b)$$

Logistic Result

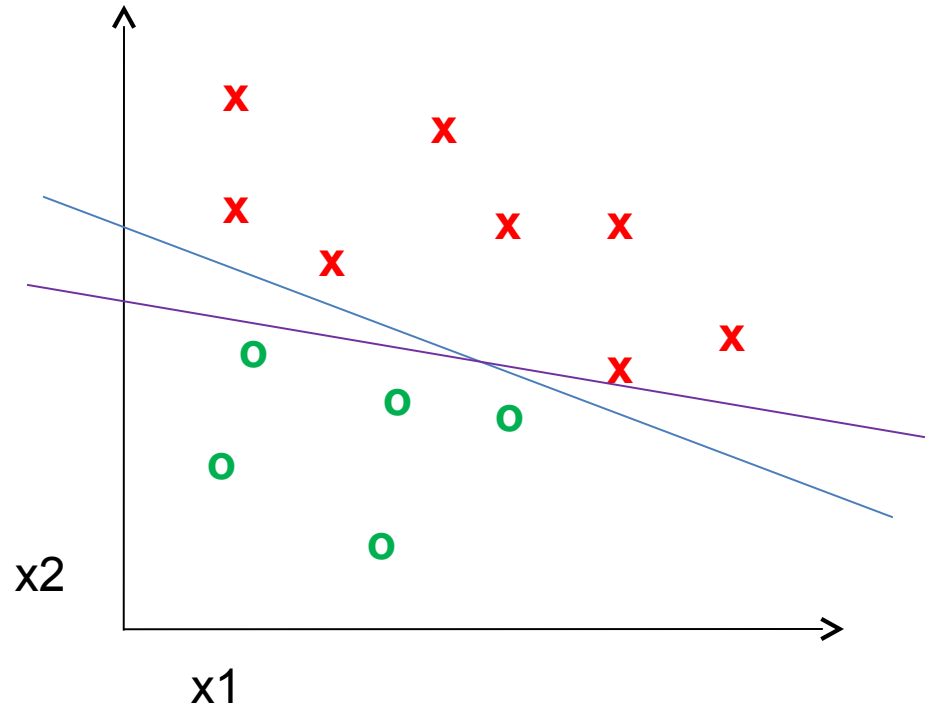


Using Logistic Regression

- Quick, simple classifier (try it first)
- Outputs a probabilistic label confidence
- Use L2 or L1 regularization
 - L1 does feature selection and is robust to irrelevant features but slower to train



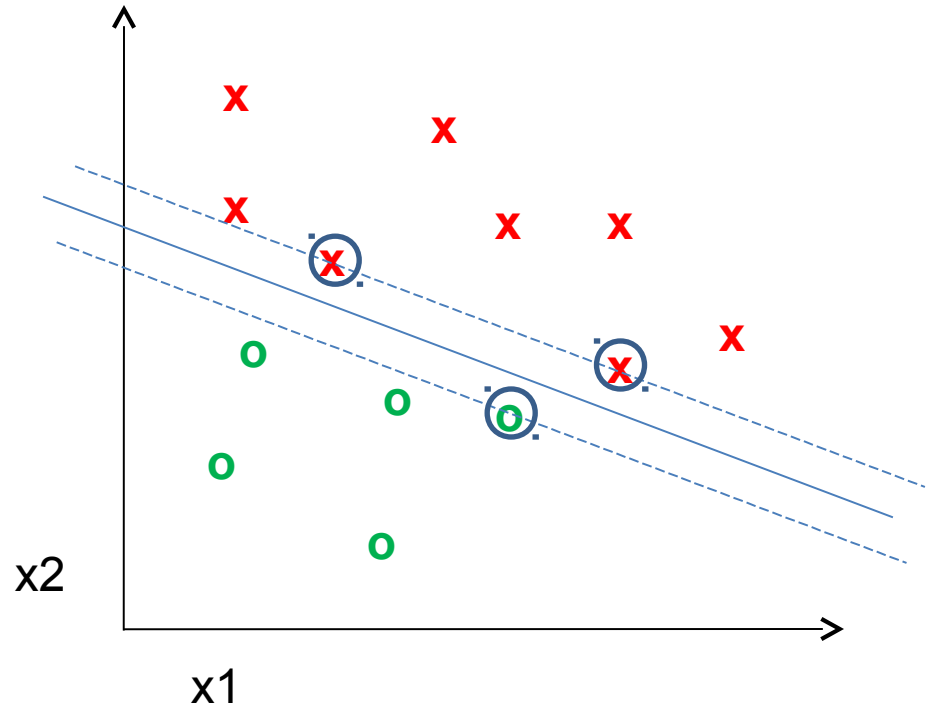
Classifiers: Linear SVM



- Find a *linear function* to separate the classes:

$$f(\mathbf{x}) = \text{sgn}(\mathbf{w} \cdot \mathbf{x} + b)$$

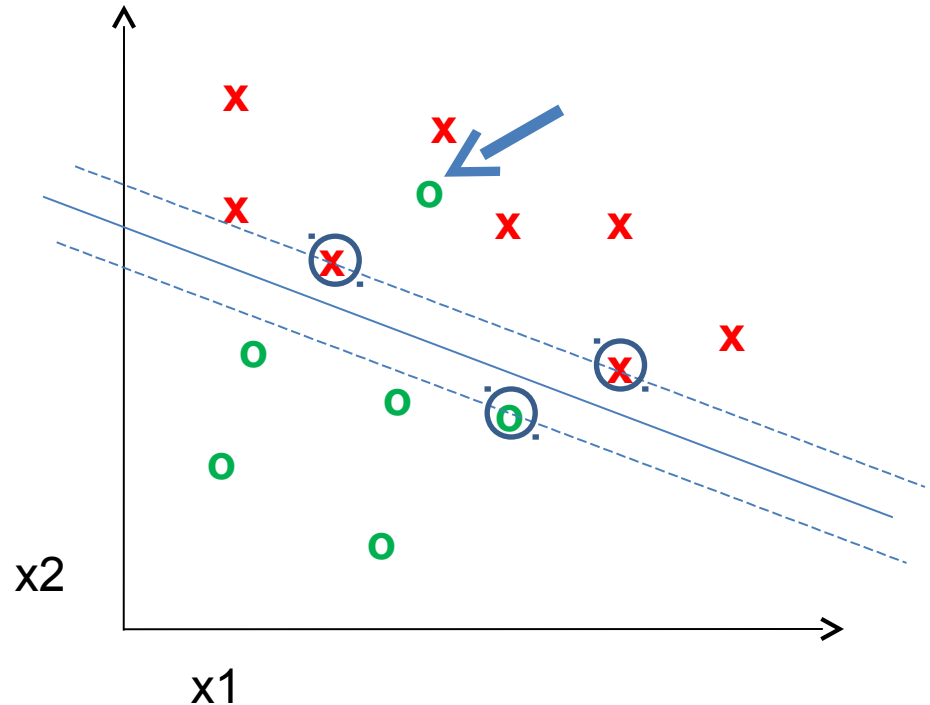
Classifiers: Linear SVM



- Find a *linear function* to separate the classes:

$$f(\mathbf{x}) = \text{sgn}(\mathbf{w} \cdot \mathbf{x} + b)$$

Classifiers: Linear SVM

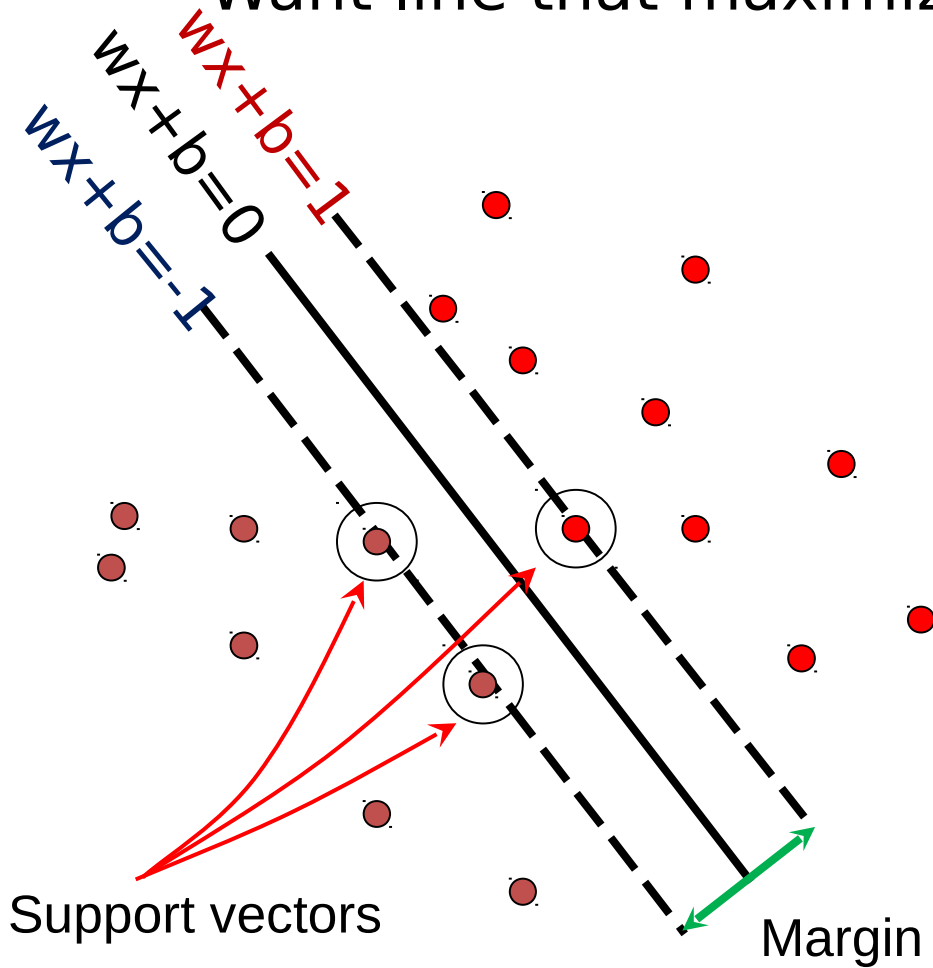


- Find a *linear function* to separate the classes:

$$f(\mathbf{x}) = \text{sgn}(\mathbf{w} \cdot \mathbf{x} + b)$$

Support vector machines: Margin

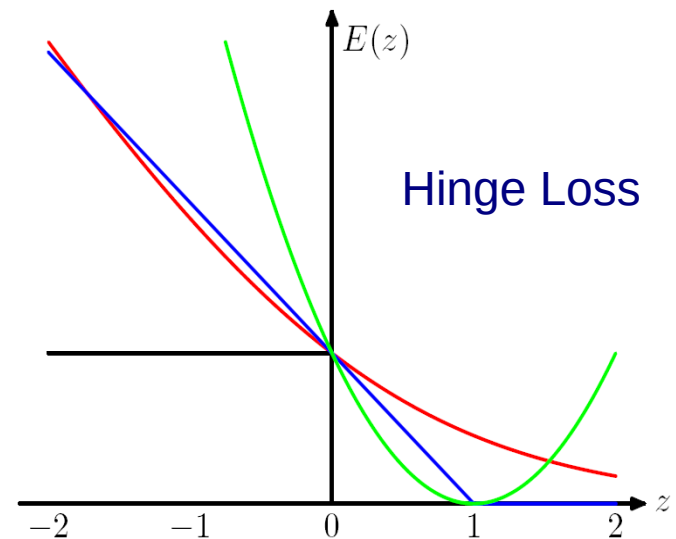
- Want line that maximizes the margin.



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

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

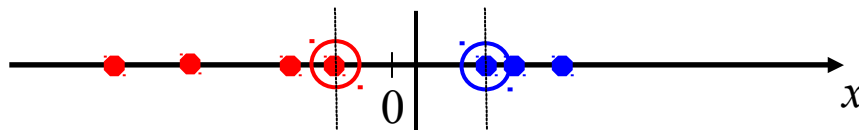
For support vectors, $\mathbf{x}_i \cdot \mathbf{w} + b = \pm 1$



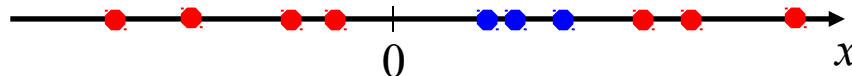
$$L(y, f(x)) = \max(0, 1 - y \cdot f(x))$$

Nonlinear SVMs

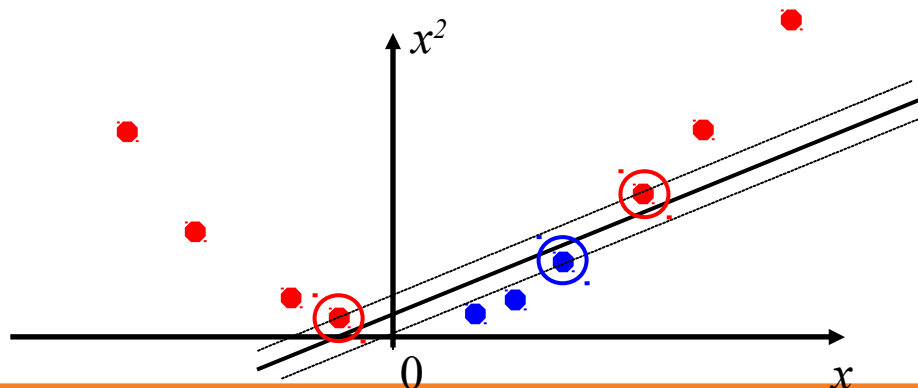
- Datasets that are linearly separable work out great:



- But what if the dataset is just too hard?

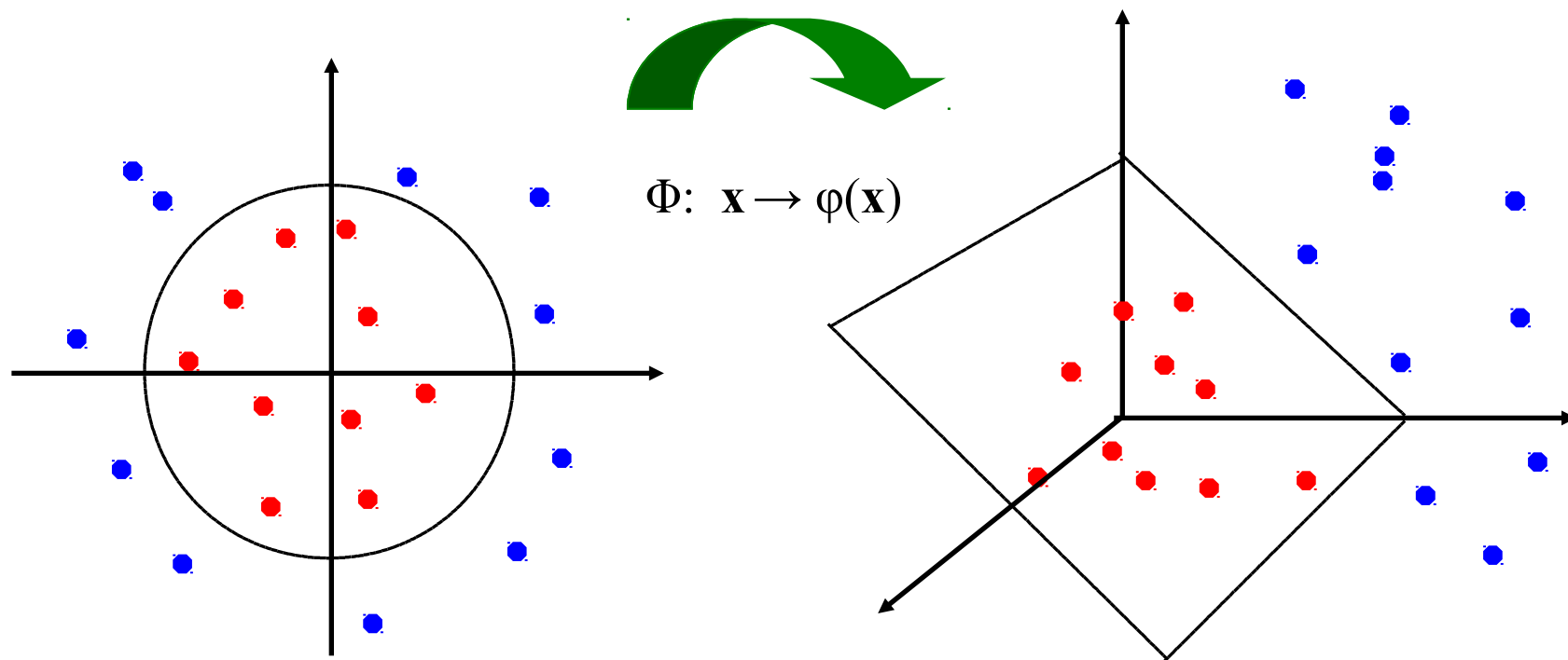


- We can map it to a higher-dimensional space:



Nonlinear SVMs

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



Nonlinear SVMs

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

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

(to be valid, the kernel function must satisfy *Mercer's condition*)

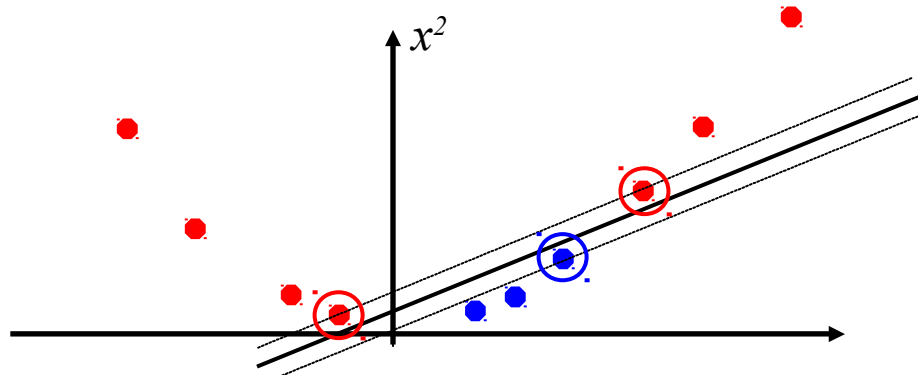
- This gives a nonlinear decision boundary in the original feature space:

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

C. Burges, [A Tutorial on Support Vector Machines for Pattern Recognition](#), Data Mining and Knowledge Discovery, 1998

Nonlinear kernel: Example

- Consider the mapping $\phi(x) = (x, x^2)$



$$\phi(x) \cdot \phi(y) = (x, x^2) \cdot (y, y^2) = xy + x^2 y^2$$

$$K(x, y) = xy + x^2 y^2$$

Kernels for bags of features

- Histogram intersection kernel:

$$I(h_1, h_2) = \sum_{i=1}^N \min(h_1(i), h_2(i))$$

- Generalized Gaussian kernel:

$$K(h_1, h_2) = \exp\left[-\frac{1}{A} D(h_1, h_2)^2\right]$$

- D can be (inverse) L1 distance, Euclidean distance, χ^2 distance, etc.

What about multi-class SVMs?

- Unfortunately, there is no “definitive” multi-class SVM formulation
- In practice, we have to obtain a multi-class SVM by combining multiple two-class SVMs
- One vs. others
 - Training: learn an SVM for each class vs. the others
 - Testing: apply each SVM to test example and assign to it the class of the SVM that returns the highest decision value
- One vs. one
 - Training: learn an SVM for each pair of classes
 - Testing: each learned SVM “votes” for a class to assign to the test example

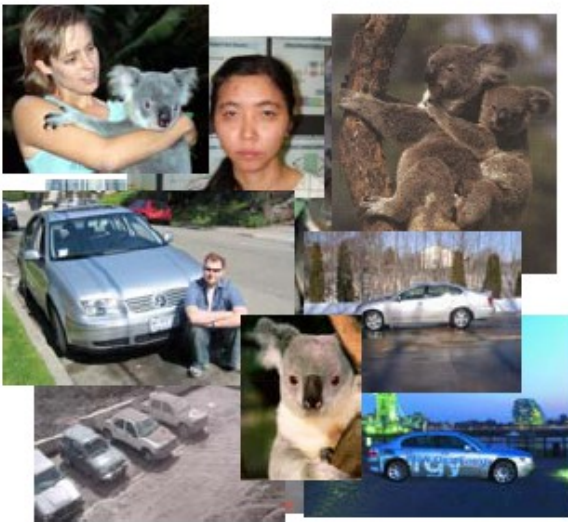
SVMs: Pros and cons

- Pros
 - Many publicly available SVM packages:
<http://www.kernel-machines.org/software>
 - Kernel-based framework is very powerful, flexible
 - SVMs work very well in practice, even with very small training sample sizes
- Cons
 - No “direct” multi-class SVM, must combine two-class SVMs
 - 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

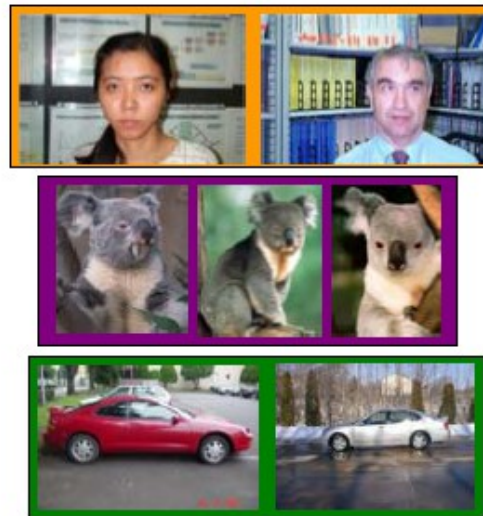
Spectrum of supervision

Less

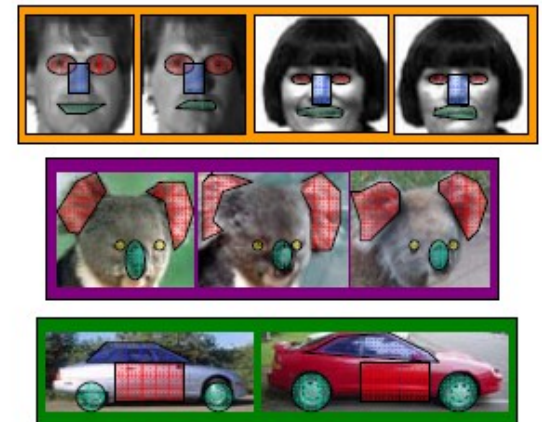
More



Unsupervised



“Weakly” supervised



Fully supervised

Definition depends on task

Generalization



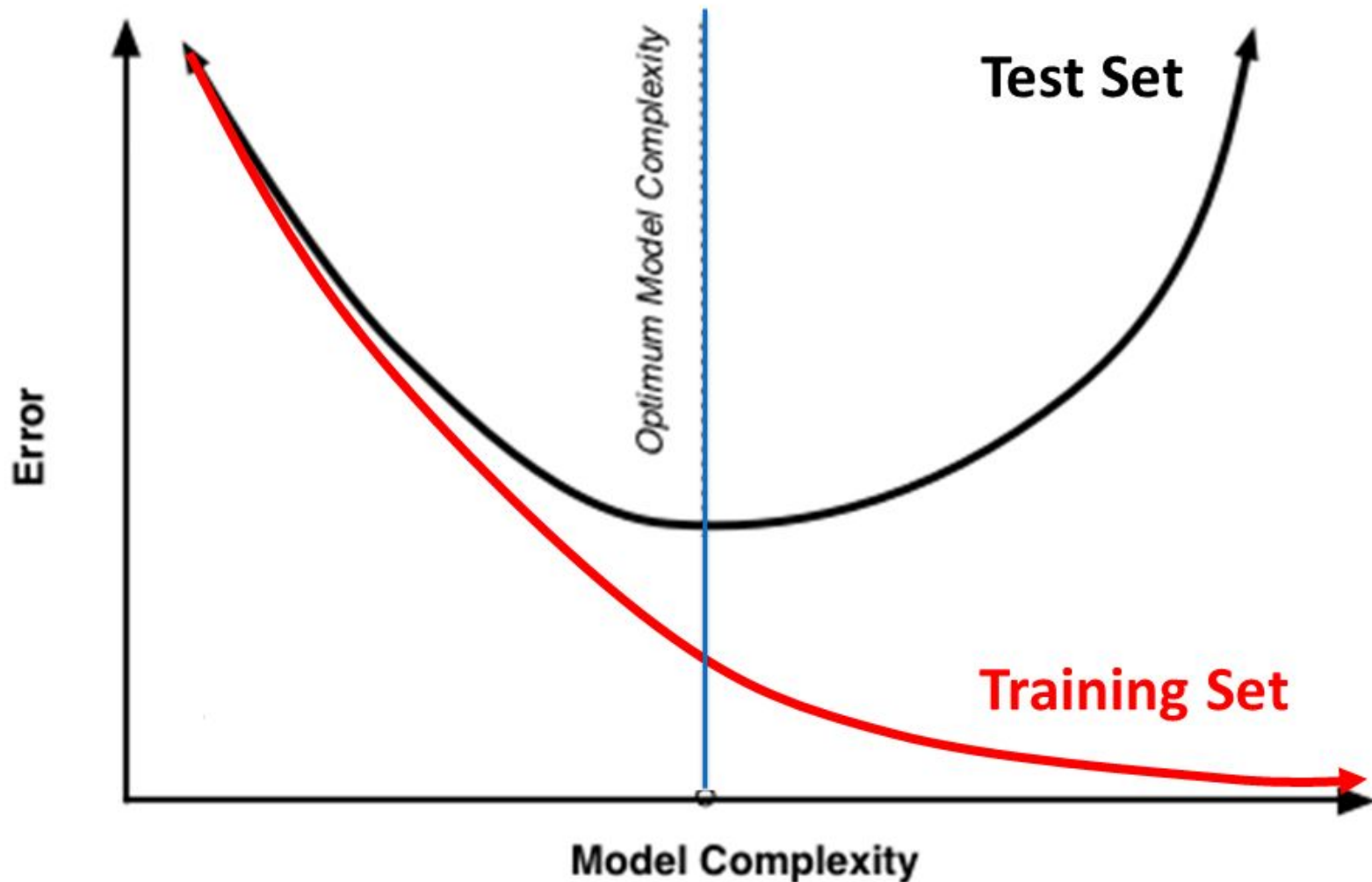
Training set (labels known)



Test set (labels unknown)

- How well does a learned model generalize from the data it was trained on to a new test set?

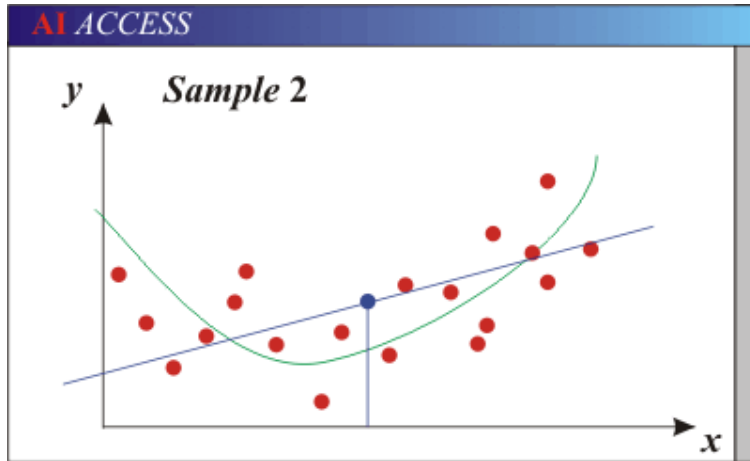
Train vs. Test Accuracy



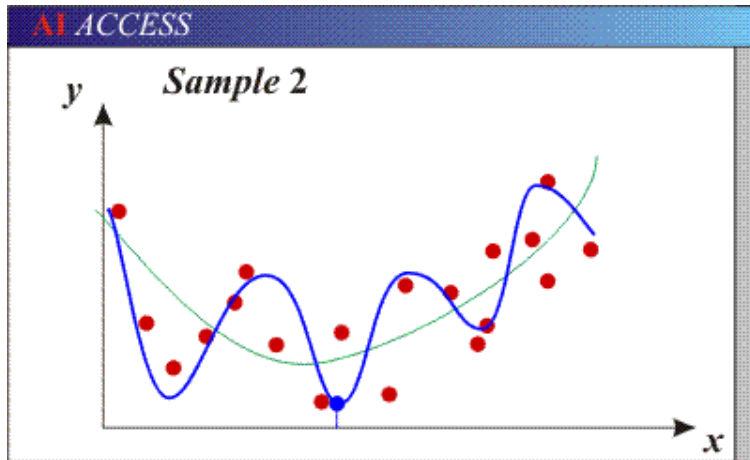
Generalization

- Components of generalization error
 - **Bias:** how much the average model over all training sets differ from the true model?
 - Error due to inaccurate assumptions/simplifications made by the model
 - **Variance:** how much models estimated from different training sets differ from each other
- **Underfitting:** model is too “simple” to represent all the relevant class characteristics
 - High bias and low variance
 - High training error and high test error
- **Overfitting:** model is too “complex” and fits irrelevant characteristics (noise) in the data
 - Low bias and high variance
 - Low training error and high test error

Bias-Variance Trade-off



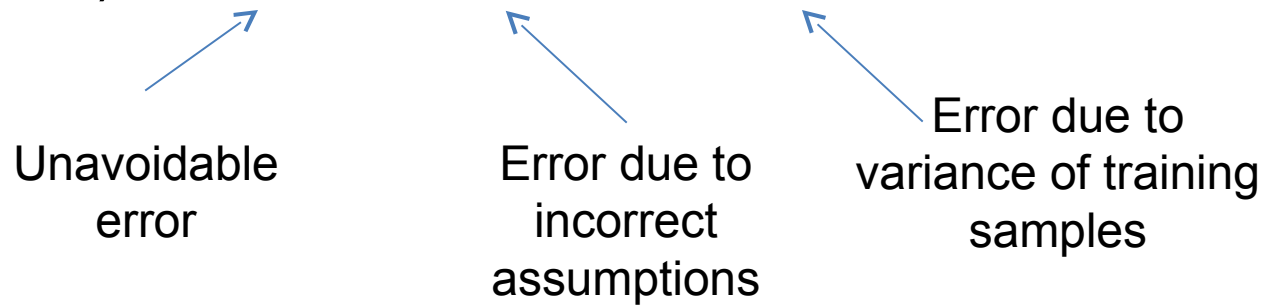
- Models with too few parameters are inaccurate because of a large bias (not enough flexibility).



- Models with too many parameters are inaccurate because of a large variance (too much sensitivity to the sample).

Bias-Variance Trade-off

$$E(\text{MSE}) = \text{noise}^2 + \text{bias}^2 + \text{variance}$$



Unavoidable
error

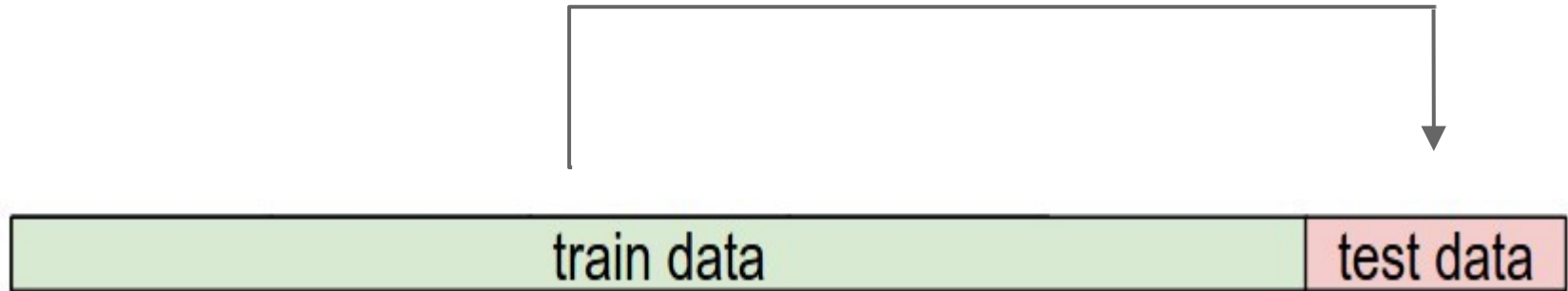
Error due to
incorrect
assumptions

Error due to
variance of training
samples

See the following for explanations of bias-variance (also Bishop's "Neural Networks" book):

- <http://www.inf.ed.ac.uk/teaching/courses/mlsc/Notes/Lecture4/BiasVariance.pdf>

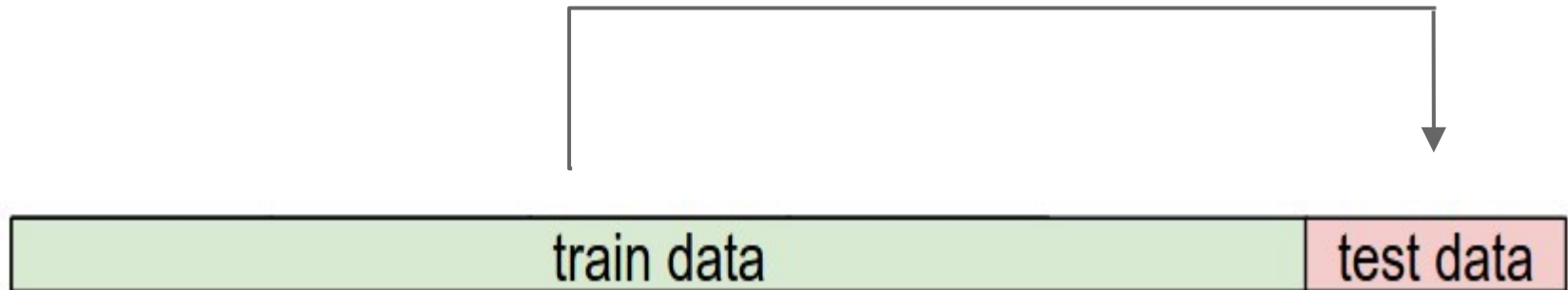
Try out what hyperparameters work best on test set.

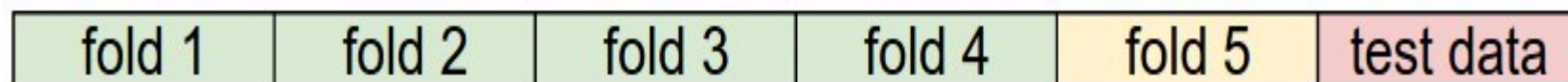
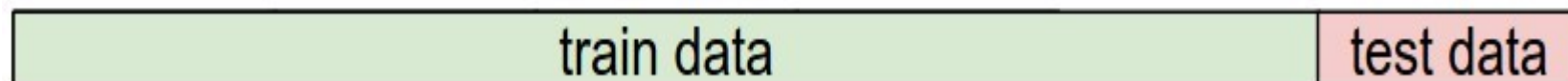


Trying out what hyperparameters work best on test set:

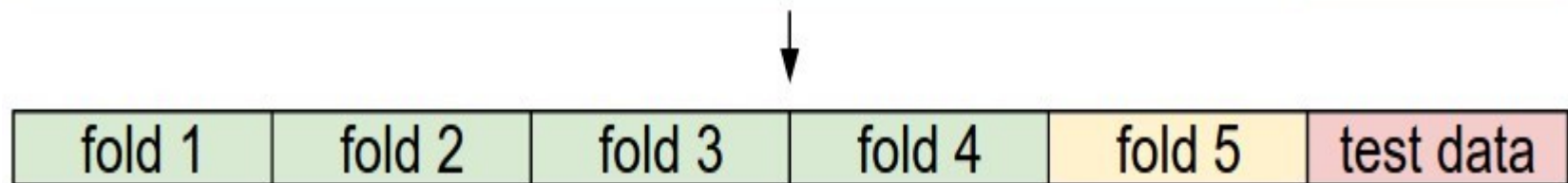
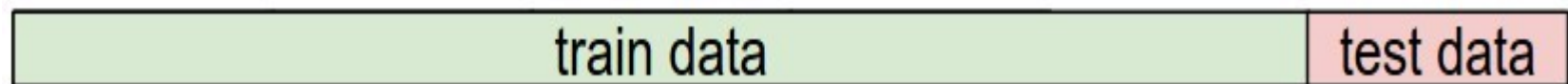
Very bad idea. The test set is a proxy for the generalization performance!

Use only **VERY SPARINGLY**, at the end.





Validation data
use to tune hyperparameters



Cross-validation

cycle through the choice of which fold is the validation fold, average results.

Remember...

- No classifier is inherently better than any other: you need to make assumptions to generalize
- Three kinds of error
 - Inherent: unavoidable
 - Bias: due to over-simplifications
 - Variance: due to inability to perfectly estimate parameters from limited data



What to remember about classifiers

- Machine learning algorithms are tools, not dogmas
- Try simple classifiers first
- Better to have smart features and simple classifiers than simple features and smart classifiers
- Use increasingly powerful classifiers with more training data (bias-variance tradeoff)

How to reduce variance?

- Choose a simpler classifier
- Regularize the parameters
- Get more training data

Generative vs. Discriminative Classifiers

Generative Models

- Represent both the data and the labels
- Often, makes use of conditional independence and priors
- Examples
 - Naïve Bayes classifier
 - Bayesian network
- Models of data may apply to future prediction problems

Discriminative Models

- Learn to directly predict the labels from the data
- Often, assume a simple boundary (e.g., linear)
- Examples
 - Logistic regression
 - SVM
 - Boosted decision trees
- Often easier to predict a label from the data than to model the data

Some Machine Learning

- General

- Tom Mitchell, *Machine Learning*, McGraw Hill, 1997
- Christopher Bishop, *Neural Networks for Pattern Recognition*, Oxford University Press, 1995

- Adaboost

- Friedman, Hastie, and Tibshirani, “Additive logistic regression: a statistical view of boosting”, *Annals of Statistics*, 2000

- SVMs

- <http://www.support-vector.net/icml-tutorial.pdf>