Lecture 8
Classification &
Dalal & Trigs

COS 429: Computer Vision

PRINCETON UNIVERSITY

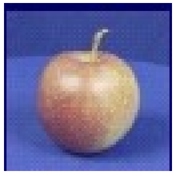
Review: Image Classification Steps

Training

Training Images

Image Features → Training (← Training Labels) → Learned model

Testing



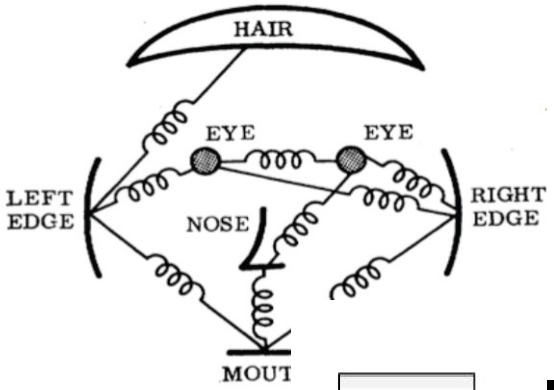
Test Image

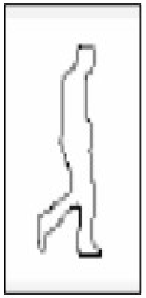
Image Features → Learned model → Prediction

Slide Credit: D. Hoiem and L. Lazebnik

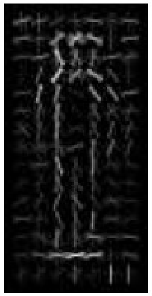
Review: Typical Components

- **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, ...
  - Generative vs. Discriminative
- **Verification -** removing redundant, overlaping, incompatible examples
  - Non-Max Suppression, context priors, geometry

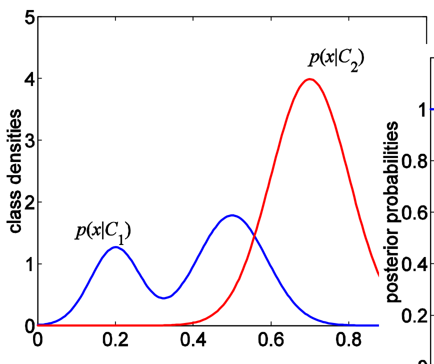


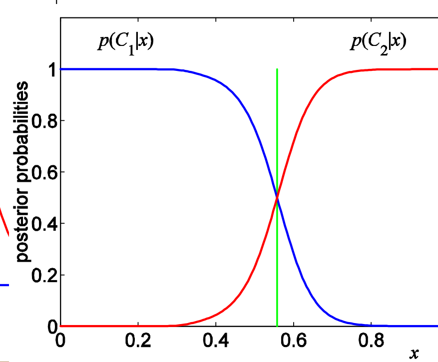


Exemplar



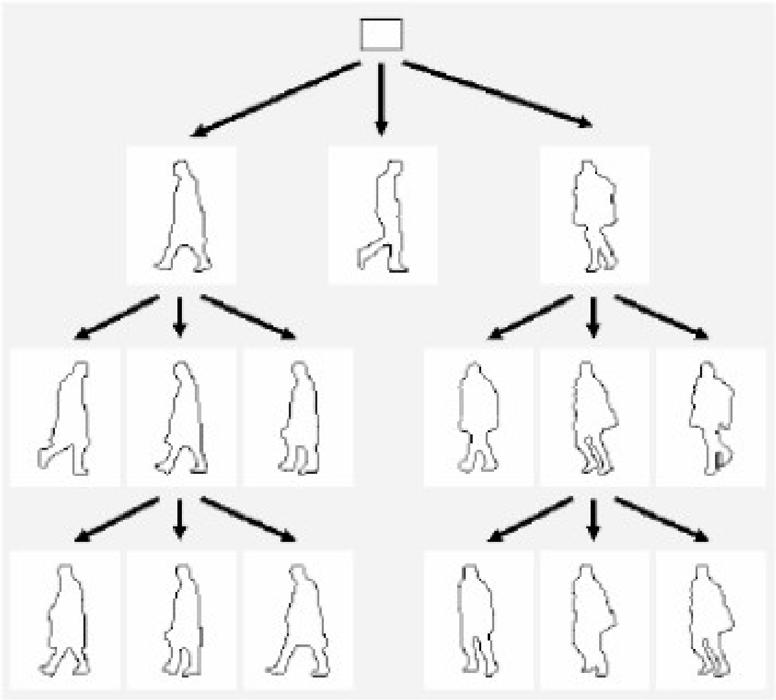
Summary



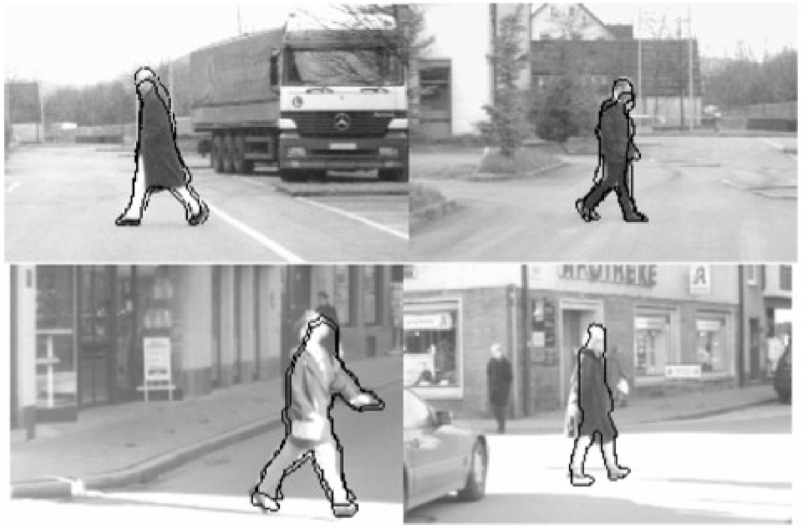


Slide Credit:

Example 1: Chamfer matching (Pedestrian Detection)



Hierarchy of templates



Gavrila & Philomin ICCV 1999

Slides from K. Grauman and B. Leibe

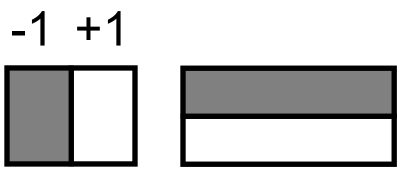
Slide Credit:

Example 2: Viola/Jones (Face Detection)

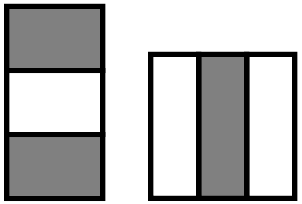
Robust Realtime Face Dection, IJCV 2004, Viola and Jones

Features: “Haar-like Rectangle filters”

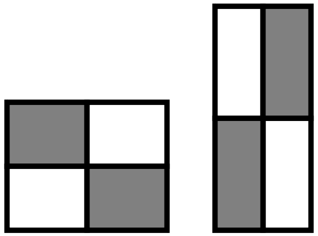
- Differences between sums of pixels in adjacent rectangles



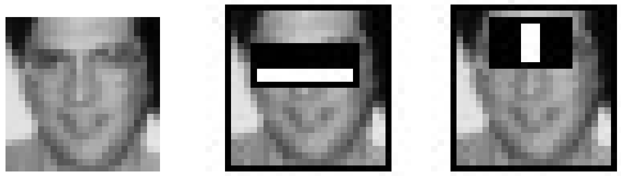
2-rectangle features



3-rectangle features

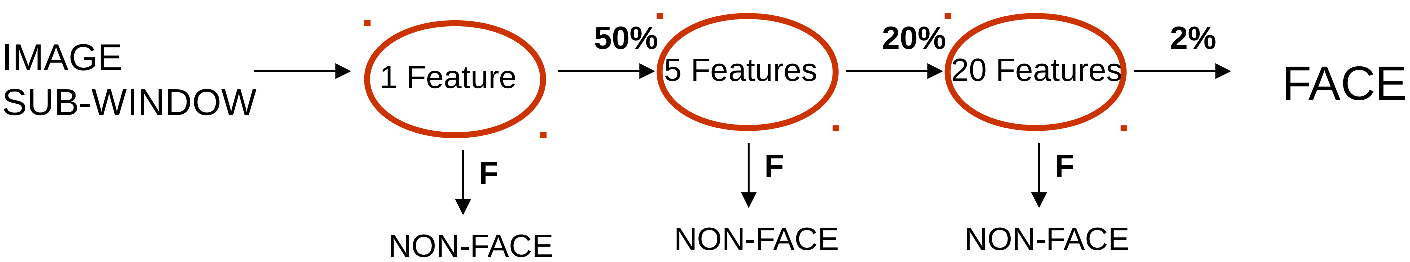


4-rectangles features



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

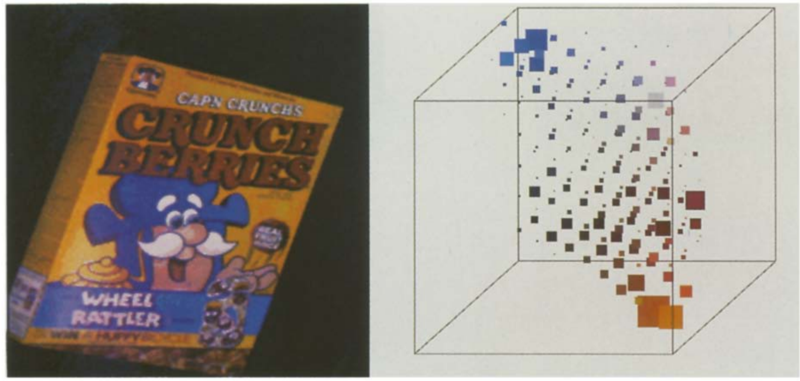
Unique Features

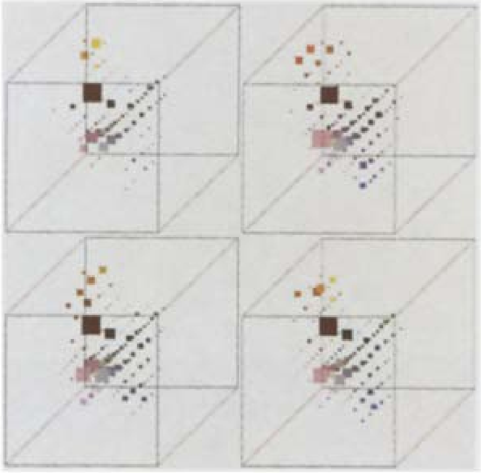


Slide from: Derek Hoiem

Slide Credit:

(No Geometry) Example: Color Histograms





Swain and Ballard, Color Indexing, IJCV 1991.

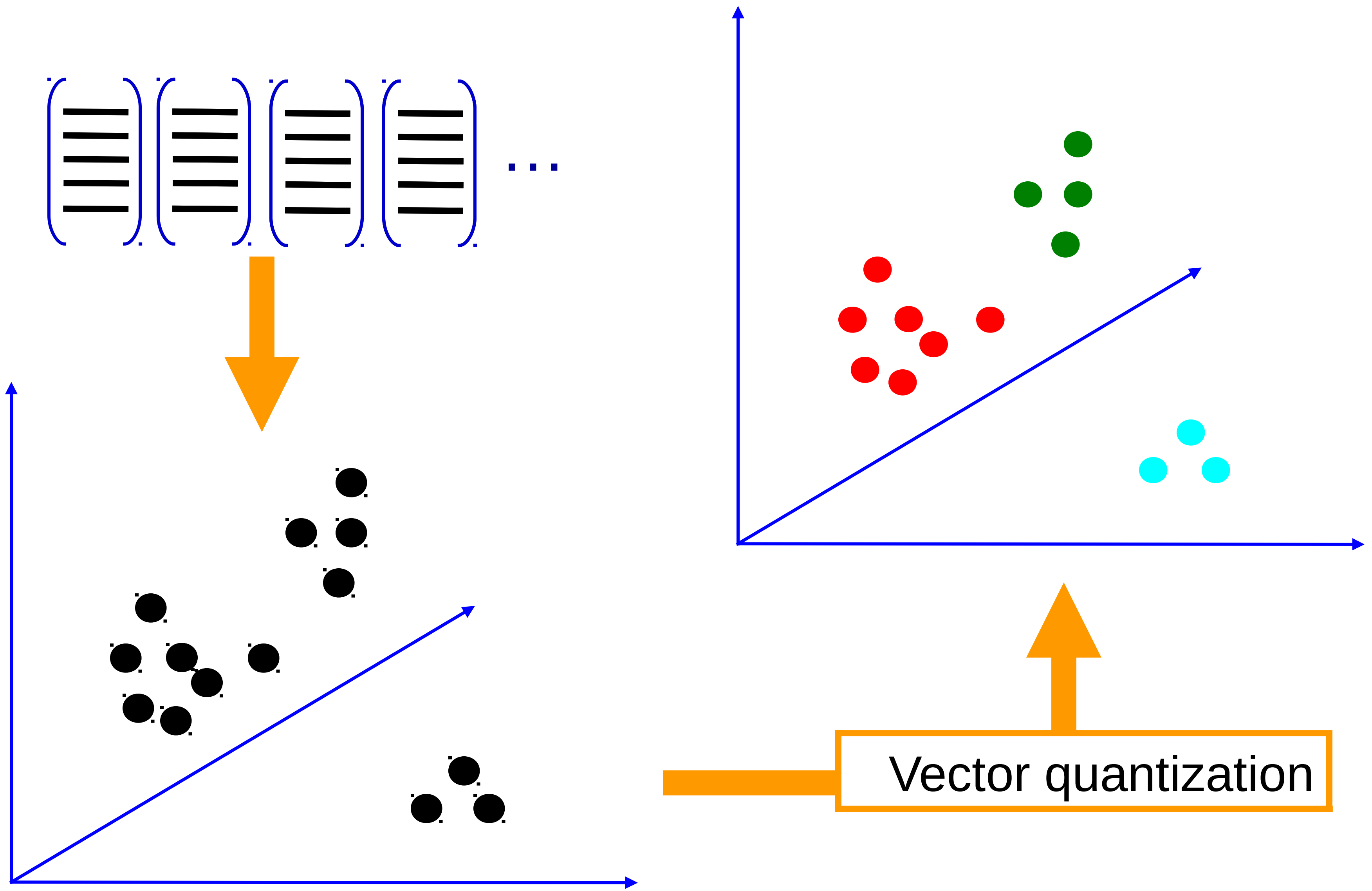
 Slide Credit: Svetlana Lazebnik

(No Geometry) Example: Bad of Words

Object → **Bag of ‘words’**

Slide Credit: Svetlana Lazebnik

Clustering (usually k-means)



Slide Credit: Josef Sivic

Machine Learning Problems

	Supervised Learning	*Unsupervised Learning*
Discrete	classification or categorization	clustering
Continuous	regression	dimensionality reduction

Slide Credit: James Hayes

Clustering Strategies

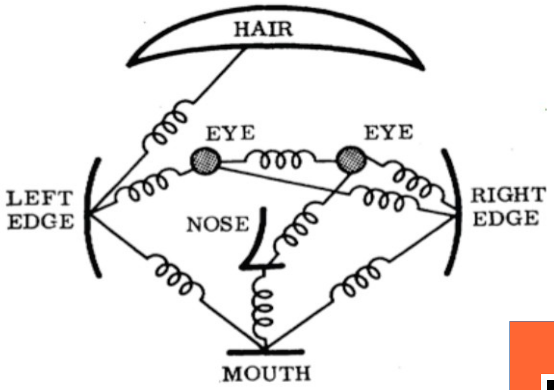
- K-means
  - Iteratively re-assign points to the nearest cluster center
- Agglomerative clustering
  - Start with each point as its own cluster and iteratively merge the closest clusters
- Mean-shift clustering
  - Estimate modes of pdf
- Spectral clustering
  - Split the nodes in a graph based on assigned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

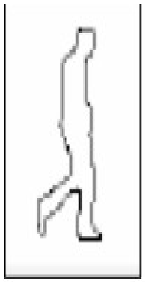
As we go down this chart, the clustering strategies have more tendency to transitively group points even if they are not nearby in feature space

Slide Credit:

Today

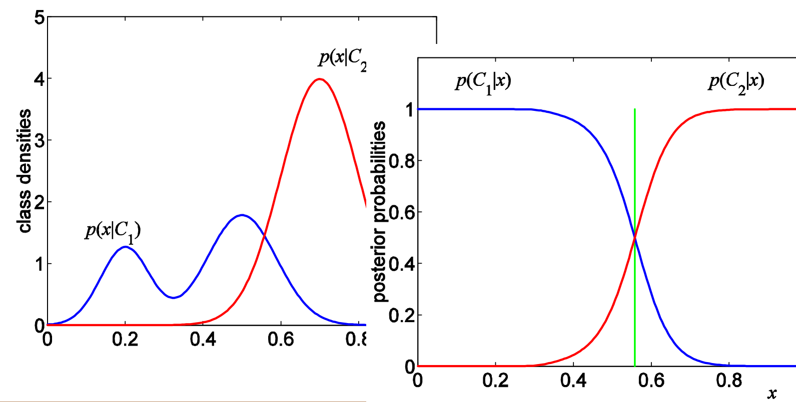
- **Hypothesis** generation
  - Sliding window, Segmentation, feature point detection, random, search
- **Encoding** of (local) image data
  - Colors, Edges, Corners, Histogram of Oriented Gradients, Wavelets, Convolution Filters
- **Relationship** of different parts to each other
  - Blur or histogram, 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, overlaping, incompatible examples
  - Non-Max Suppression, context priors, geometry





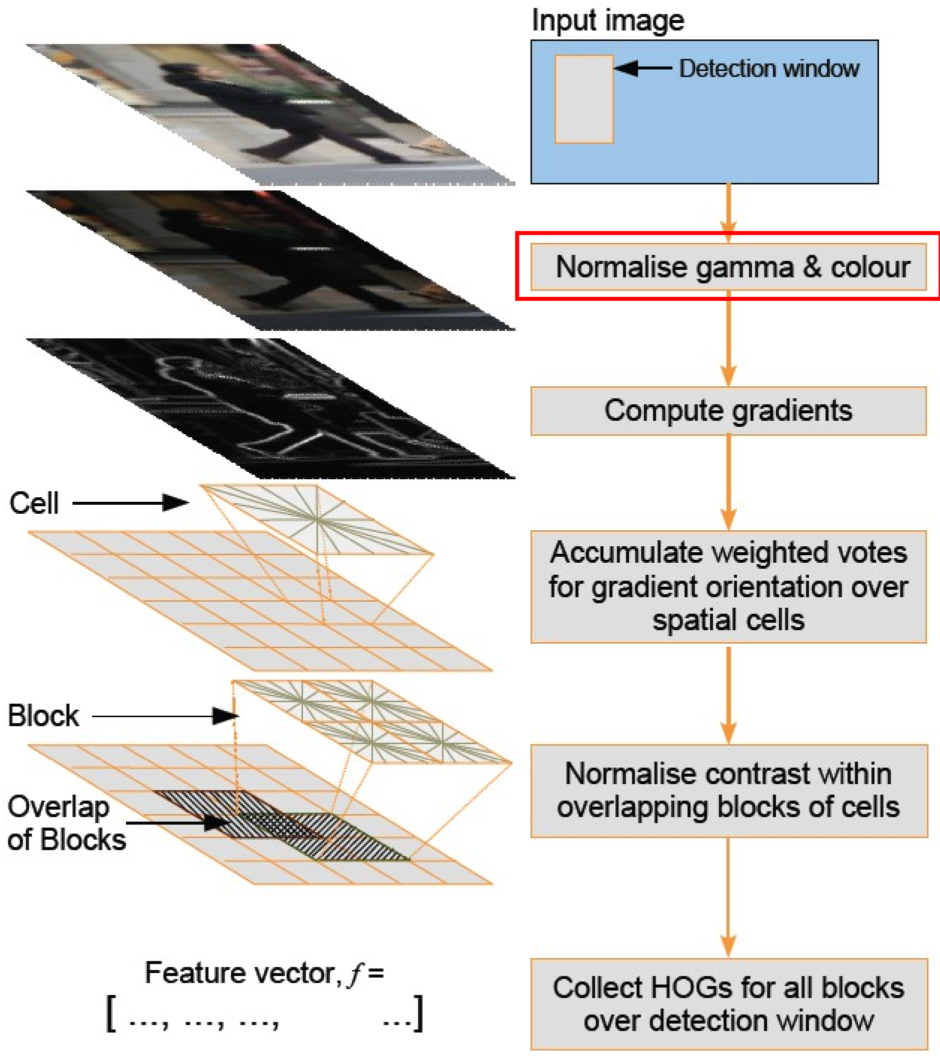
Exemplar

Summary



Slide Credit:

Dalal and Trigs Pedestrian Detector

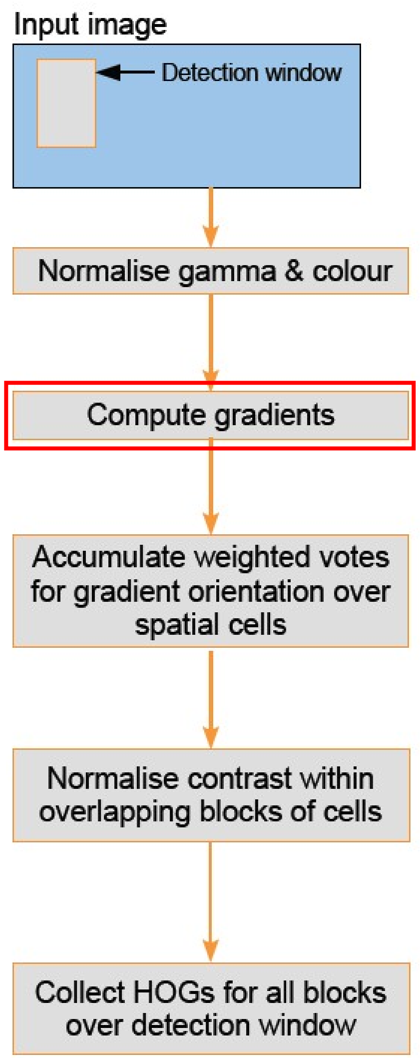


In practice, effect is very small (about 1%) while some computational time is required*

*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.

Slide Credit: *Qing J.Wang &Ru B. Zhang*

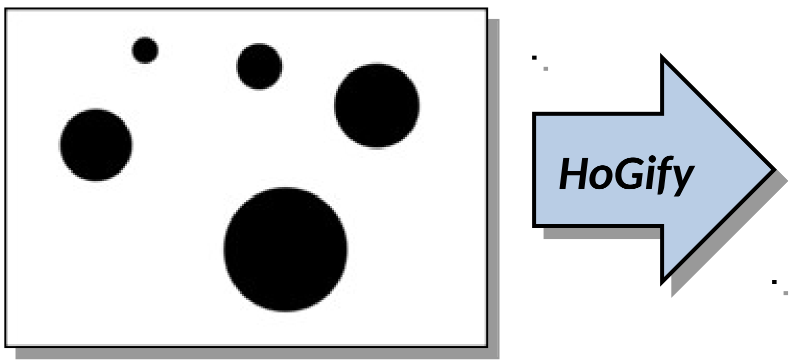
Computing gradients

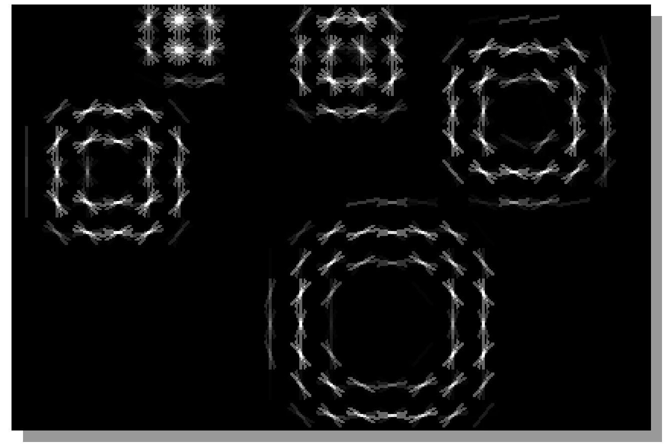


Mask Type	1D centered	1D uncentered	1D cubic-corrected	2x2 diagonal	3x3 Sobel
Operator	[-1, 0, 1]	[-1, 1]	[1, -8, 0, 8, -1]	$\begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}$ $\begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix}$	$\begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}$ $\begin{bmatrix}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

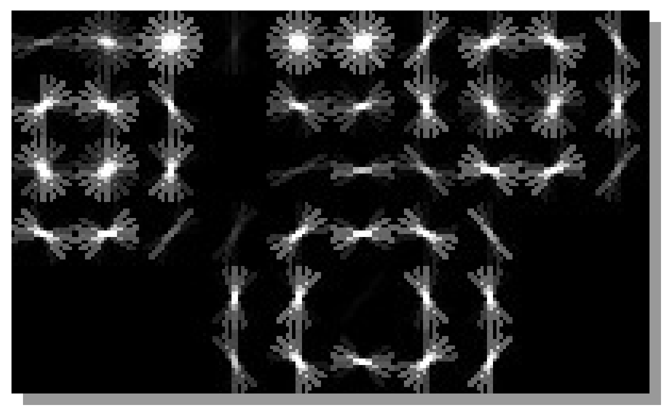
Slide Credit: *Qing J.Wang &Ru B. Zhang*

Histogram of Oriented Gradients





10x10 cells

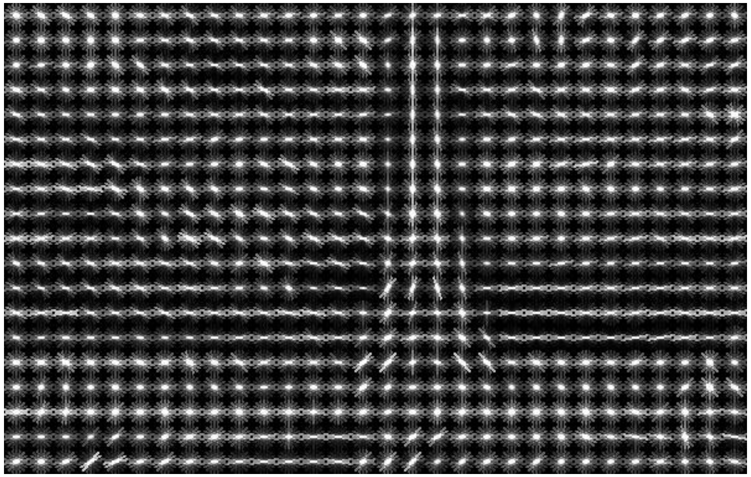


20x20 cells

[Dalal and Triggs, CVPR 2005]

Slide Credit: Noah Snavely

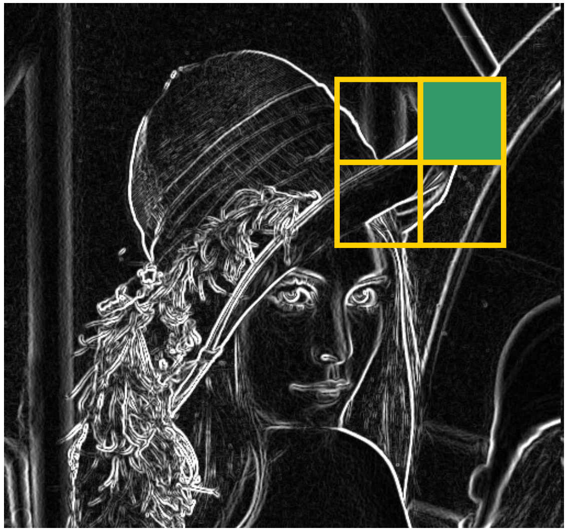
Histogram of Oriented Gradients



Slide Credit:

HOG feature vector for one block

$$f = \left(h_1^1, \ldots, h_9^1, h_1^2, \ldots, h_9^2, h_1^3, \ldots, h_9^3, h_1^4, \ldots, h_9^4\right)$$

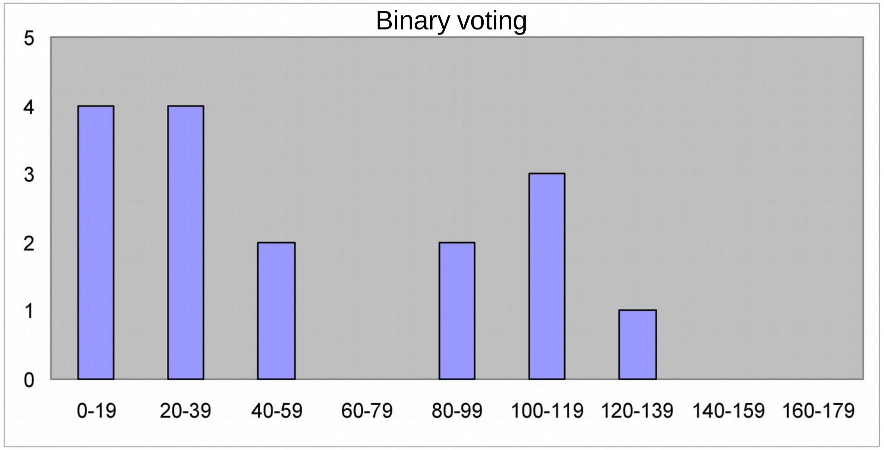


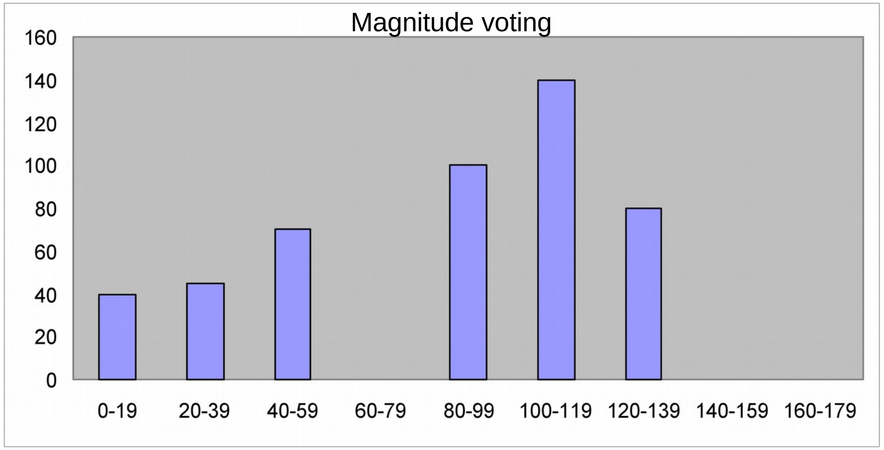
Angle

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

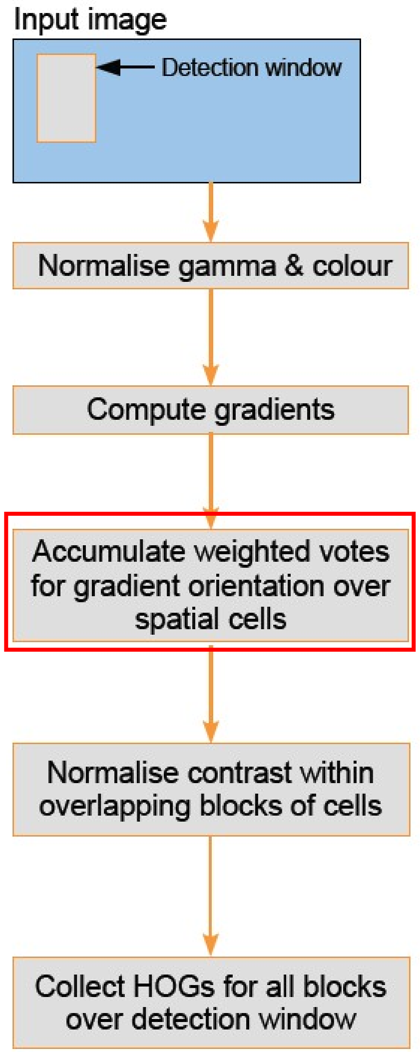




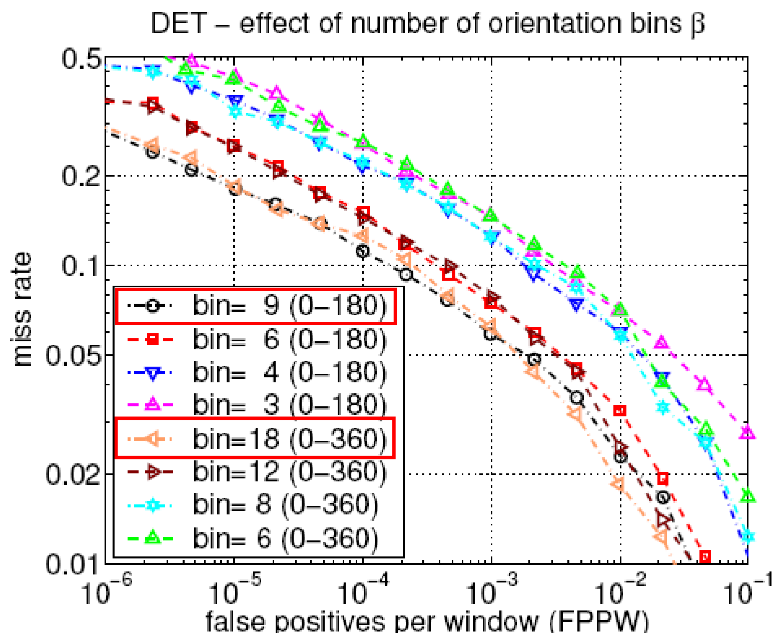
Feature vector extends while window moves

Slide Credit: *Qing J.Wang &Ru B. Zhang*

Accumulate weight votes over spatial cells

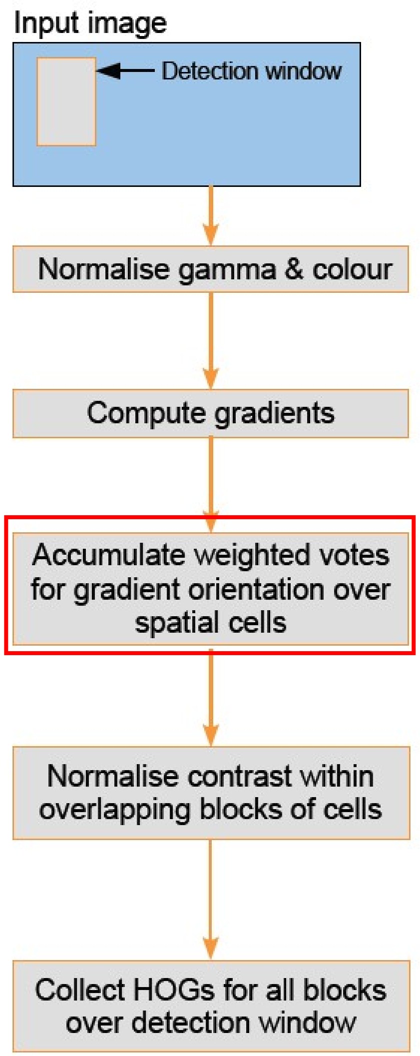


- How many bins should be in histogram?
- Should we use oriented or non-oriented gradients?
- How to select weights?
- Should we use overlapped blocks or not? If yes, then how big should be the overlap?
- What block size should we use?

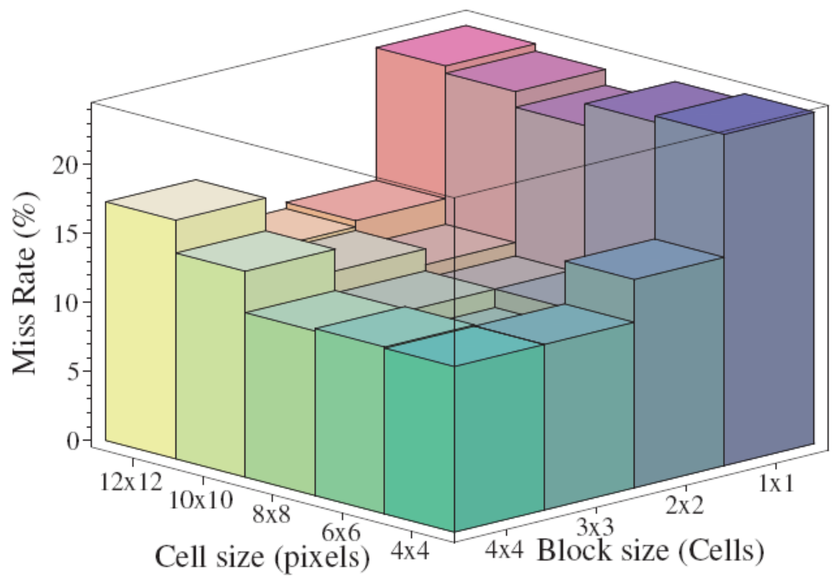


Slide Credit: *Qing J.Wang &Ru B. Zhang*

Accumulate weight votes over spatial cells

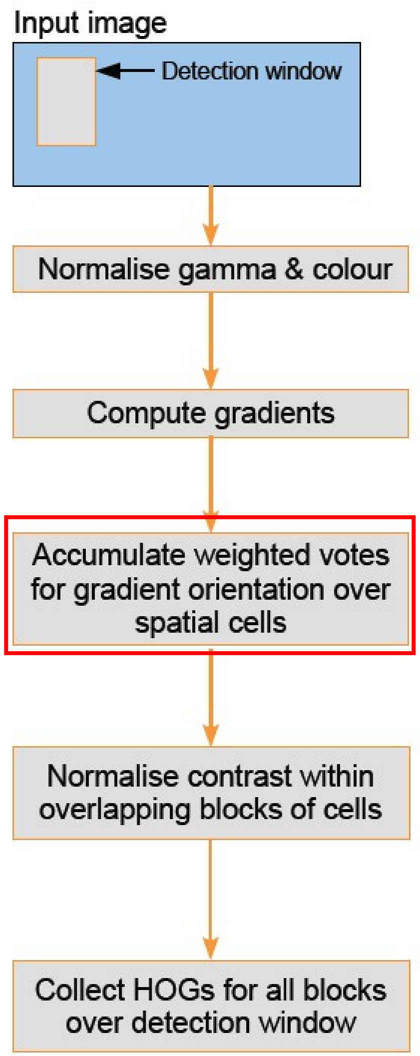


- How many bins should be in histogram?
- Should we use oriented or non-oriented gradients?
- How to select weights?
- Should we use overlapped blocks or not? If yes, then how big should be the overlap?
- What block size should we use?

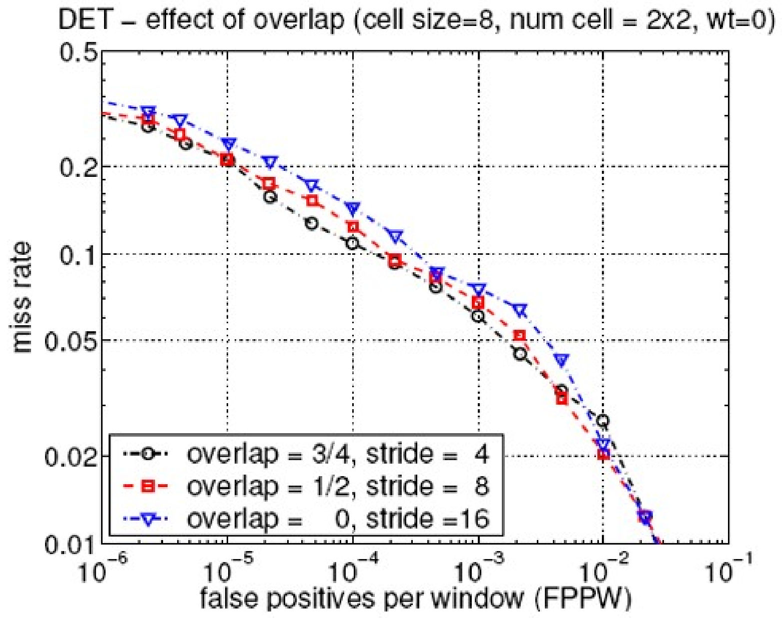


Slide Credit: *Qing J.Wang &Ru B. Zhang*

Accumulate weight votes over spatial cells

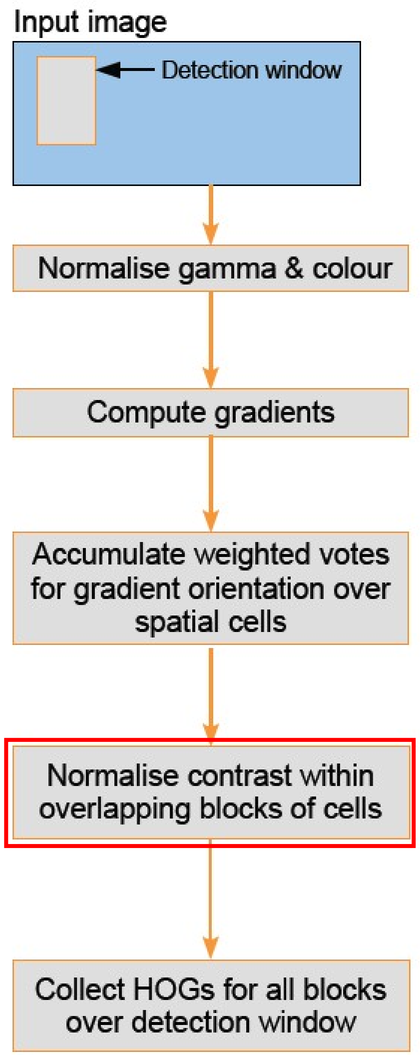


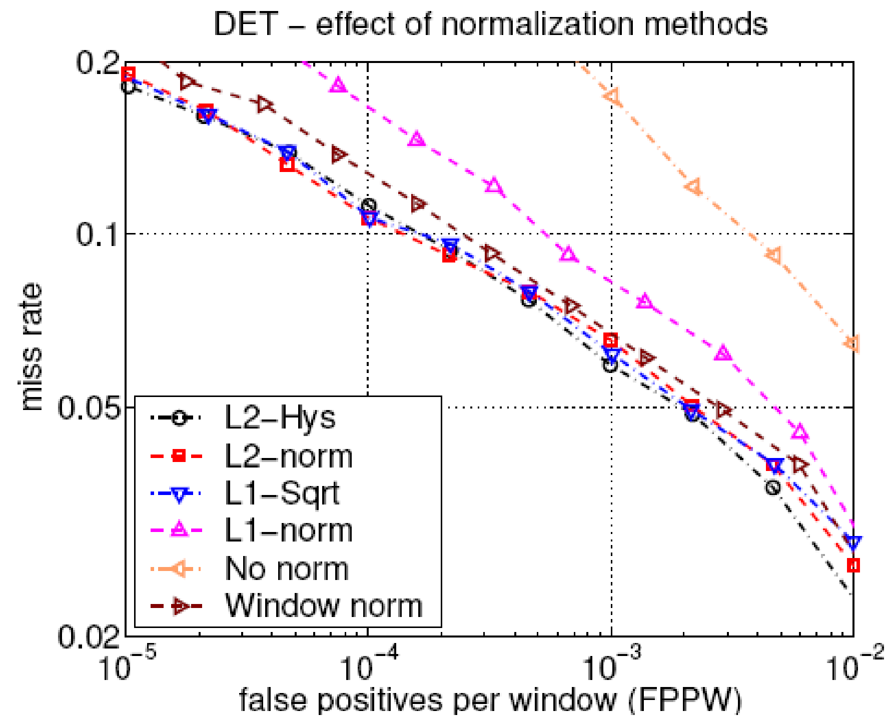
- How many bins should be in histogram?
- Should we use oriented or non-oriented gradients?
- How to select weights?
- Should we use overlapped blocks or not? If yes, then how big should be the overlap?
- What block size should we use?



Slide Credit: *Qing J.Wang &Ru B. Zhang*

Contrast normalization



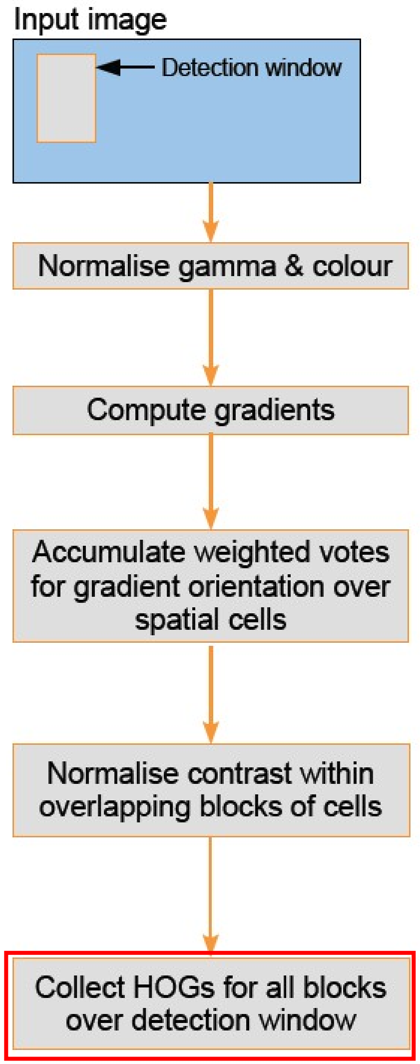


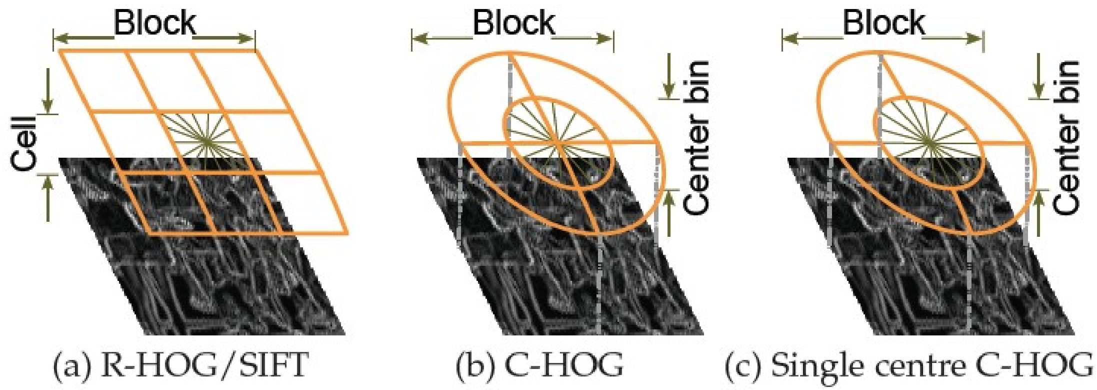
$$L1\text{-}norm = \frac{v}{\|v\|_1 + \varepsilon} \qquad L1\text{-}sqrt = \sqrt{\frac{v}{\|v\|_1 + \varepsilon}} \qquad L2\text{-}norm = \frac{v}{\sqrt{\|v\|_2^2 + \varepsilon}}$$

$L2\text{-}Hys$ - L2-norm followed by clipping (limiting the maximum values of v to 0.2) and renormalising

Slide Credit: *Qing J.Wang &Ru B. Zhang*

Making feature vector

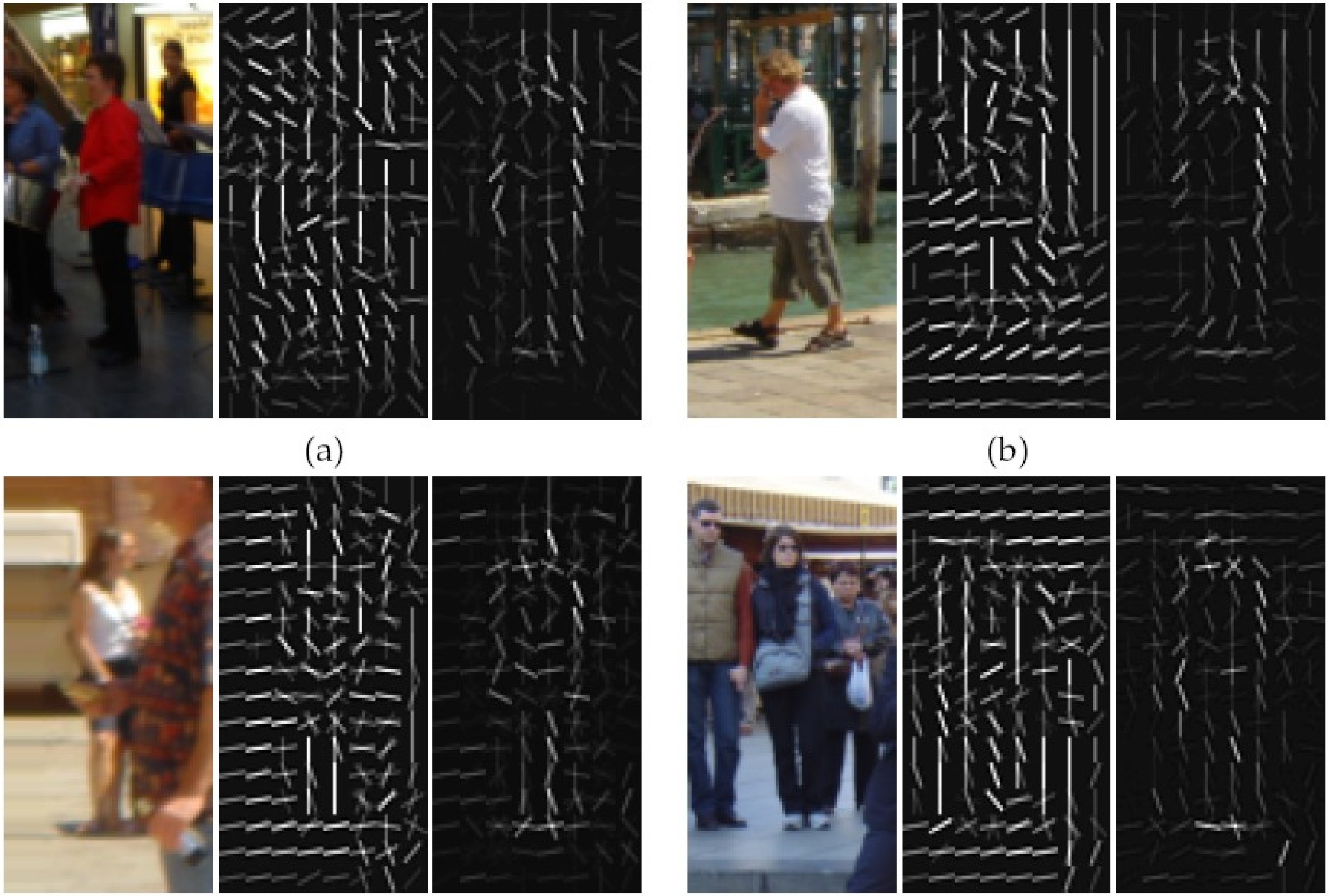




Variants of HOG descriptors. (a) A rectangular HOG (R-HOG) descriptor with 3 × 3 blocks of cells. (b) Circular HOG (C-HOG) descriptor with the central cell divided into angular sectors as in shape contexts. (c) A C-HOG descriptor with a single central cell.

Slide Credit: *Qing J.Wang &Ru B. Zhang*

HOG example

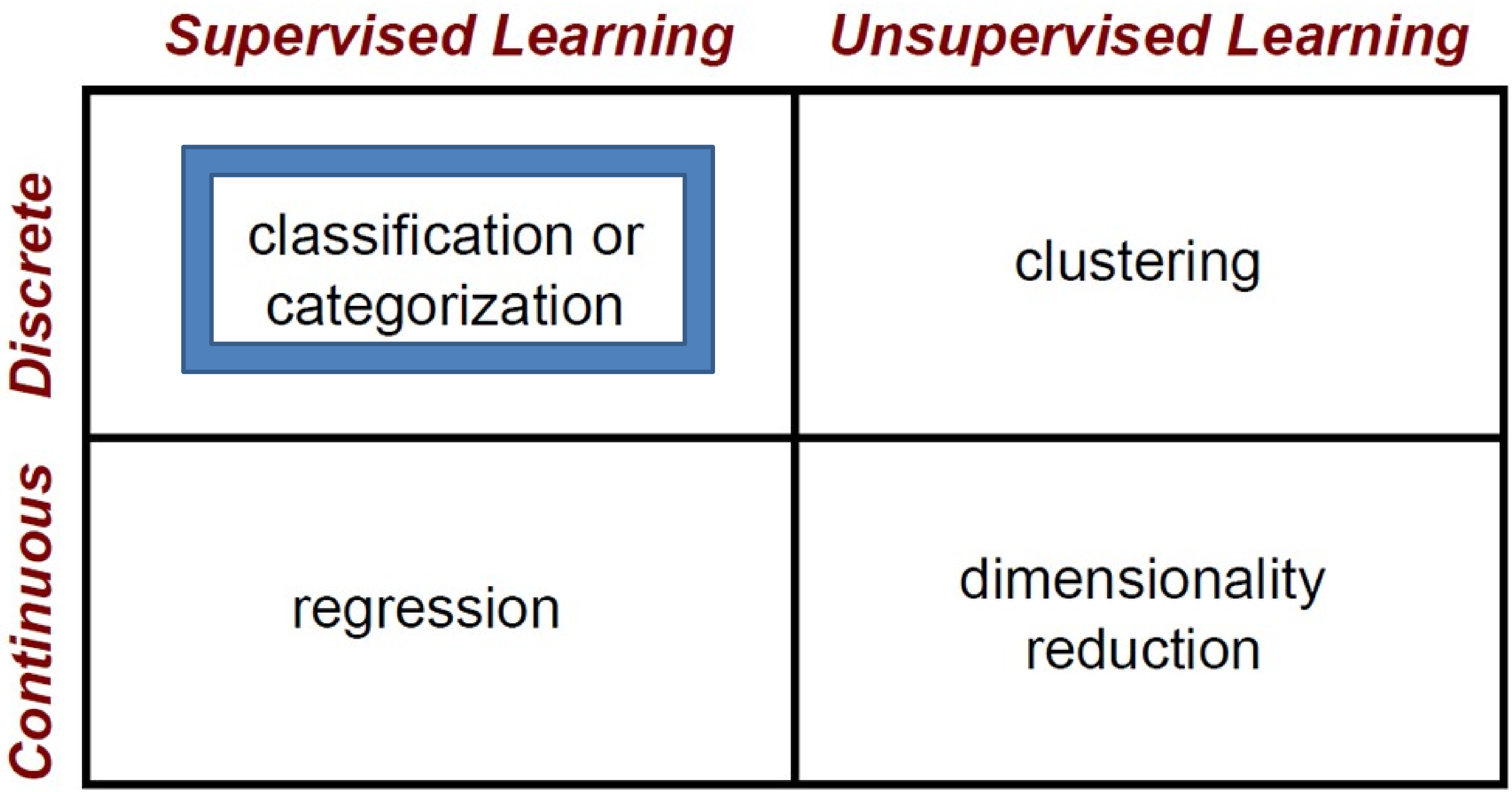


(a) (b) (c) (d)

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).

Slide Credit: *Qing J.Wang &Ru B. Zhang*

Feature Vector → Classification Result



Slide Credit: James Hayes

The machine learning framework

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

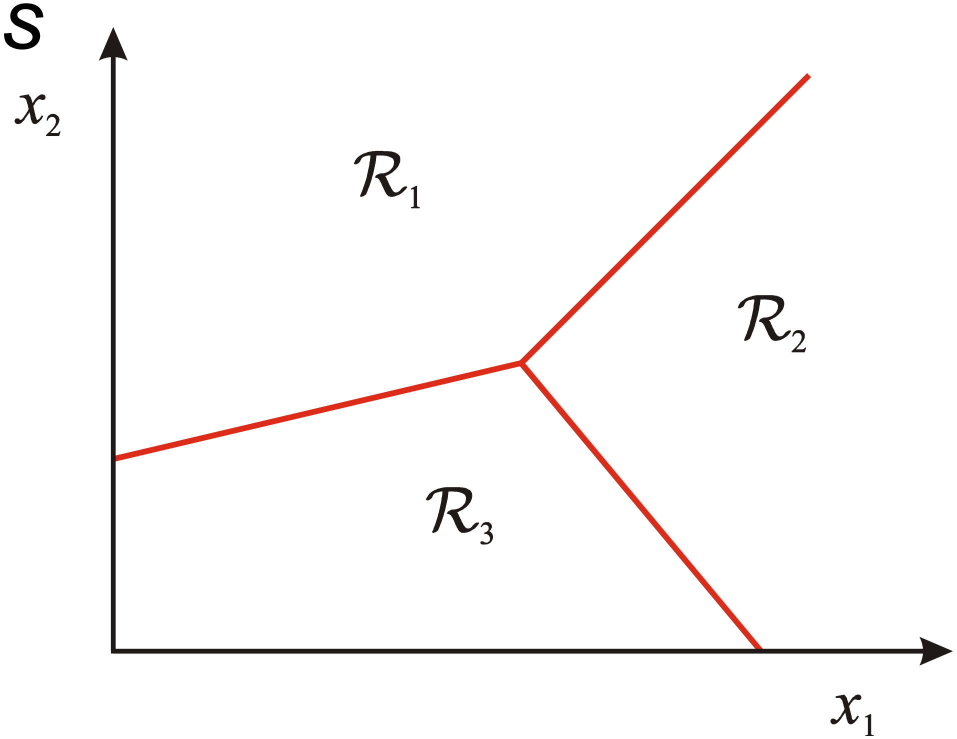
output — prediction function — Image features

- **Training:** given a *training set* of labeled examples $\{(\mathbf{x}_1,y_1), \ldots, (\mathbf{x}_N,y_N)\}$, 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 $y = f(\mathbf{x})$

Slide Credit: L. Lazebnik

Classification

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



Slide credit: L. Lazebnik

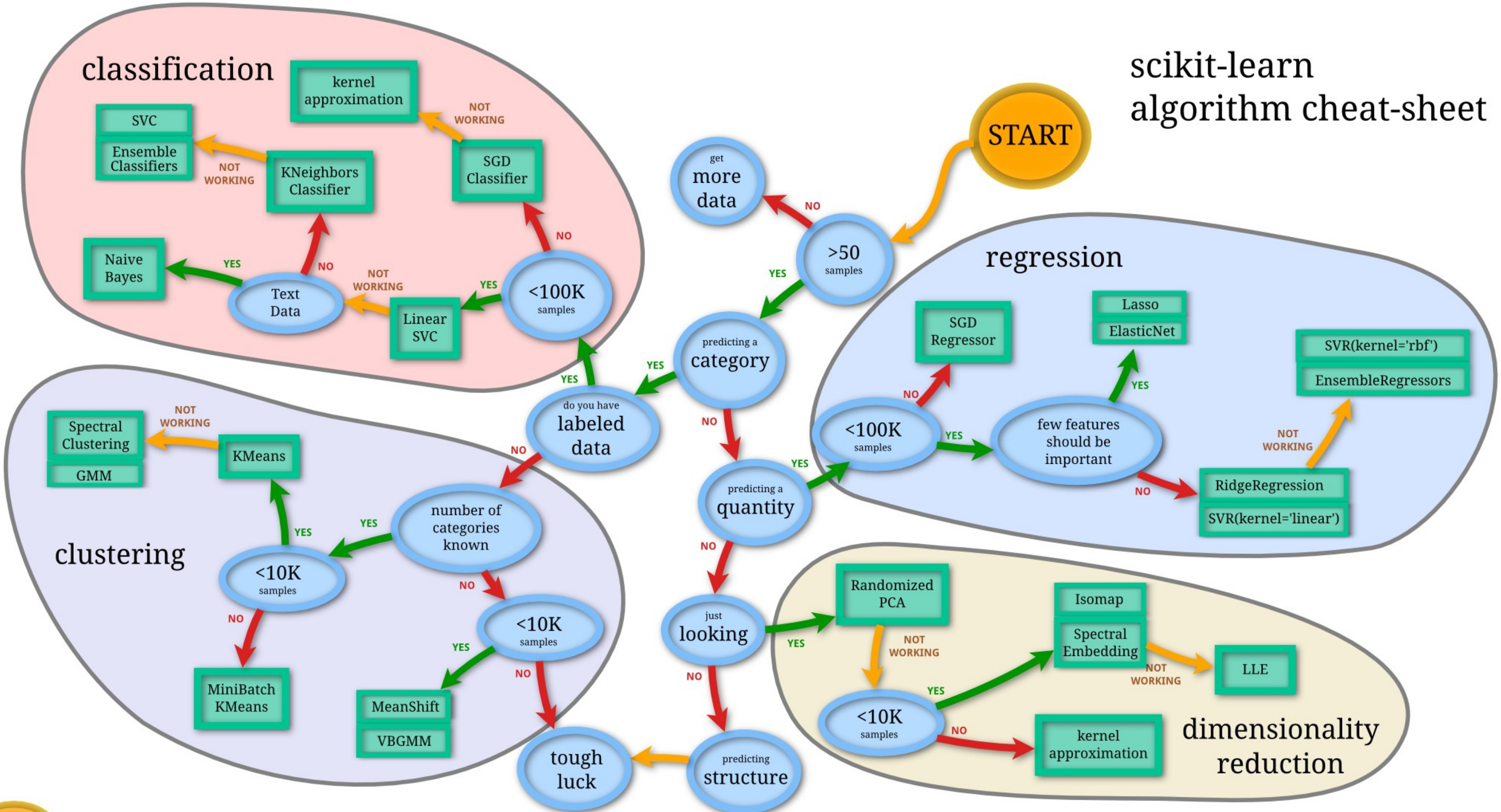
Many classifiers to choose from

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

Which is the best one?

Slide Credit: D. Hoiem

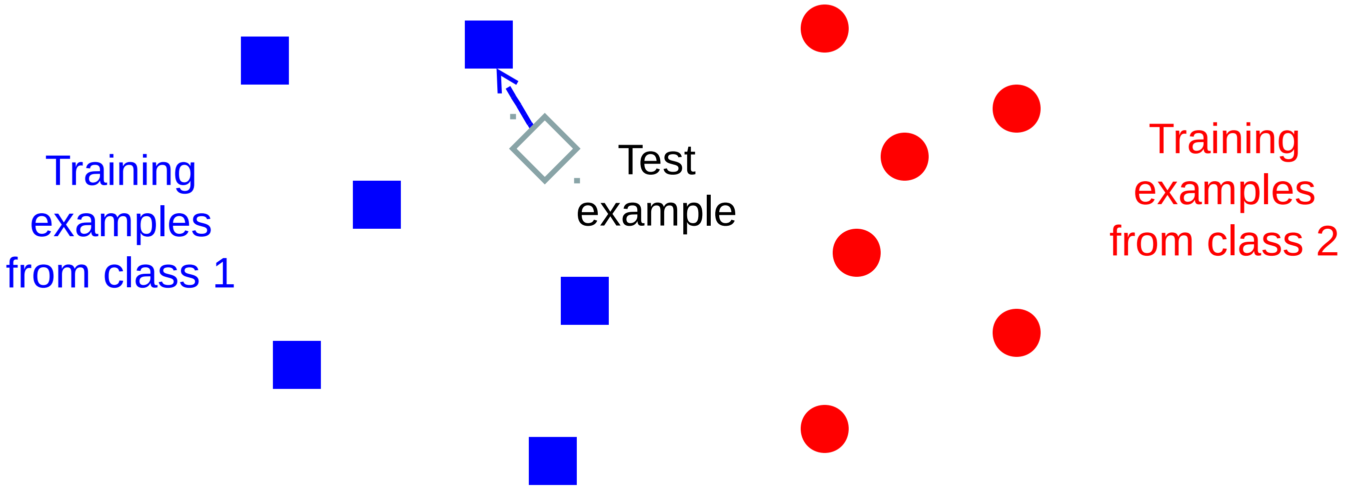
Which Algorithm to use?



Back

Slide Credit:

(1-) Nearest Neighbor

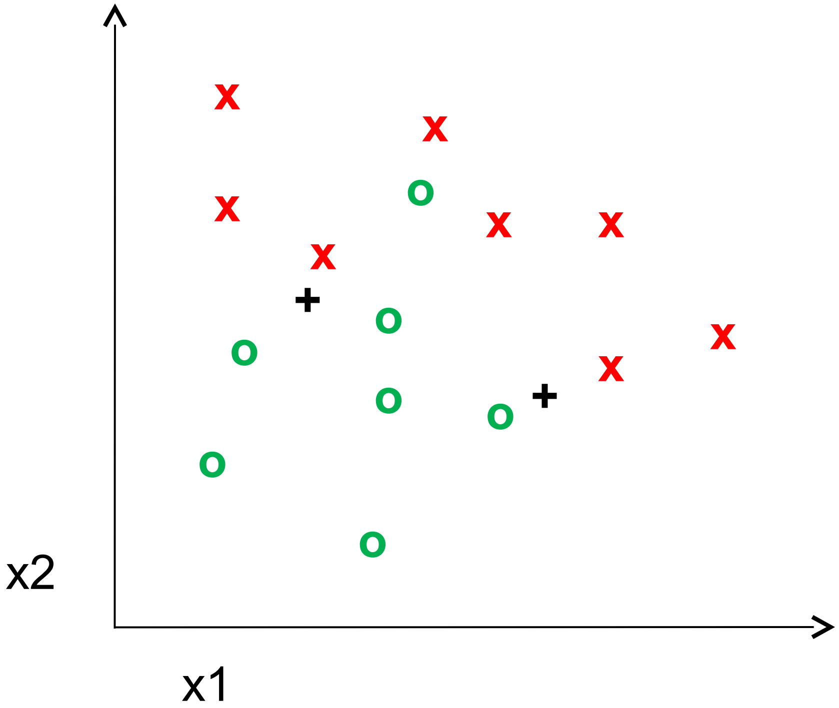


f(**x**) = label of the training example nearest to **x**

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

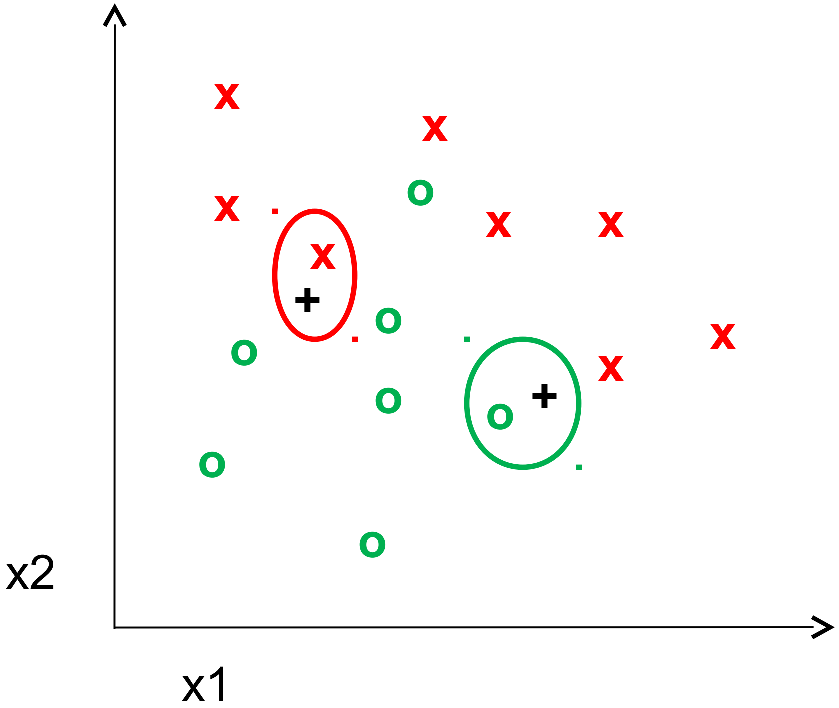
 Slide Credit: L. Lazebnik

K-nearest neighbor



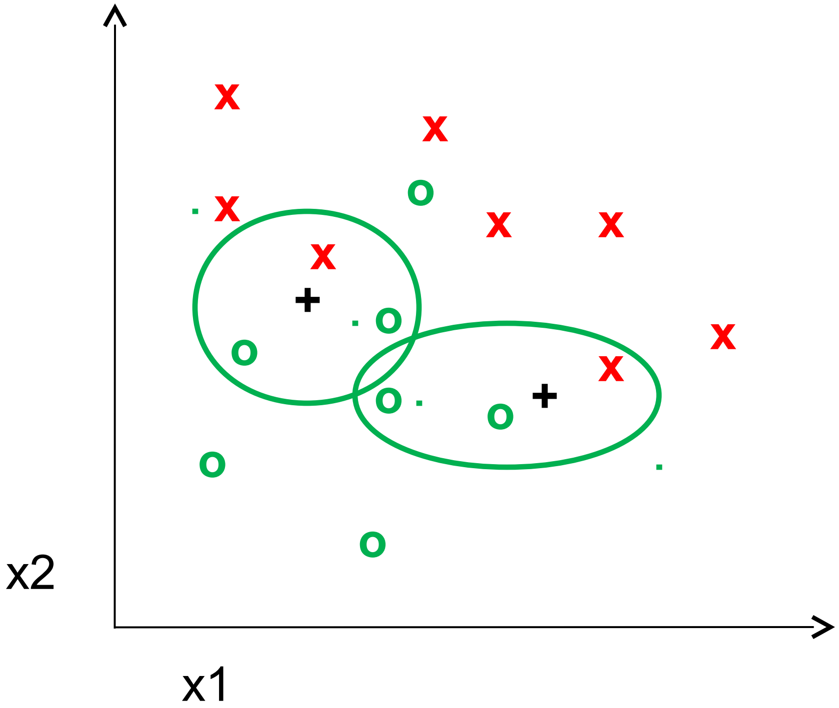
Slide Credit: D. Hoiem

1-nearest neighbor



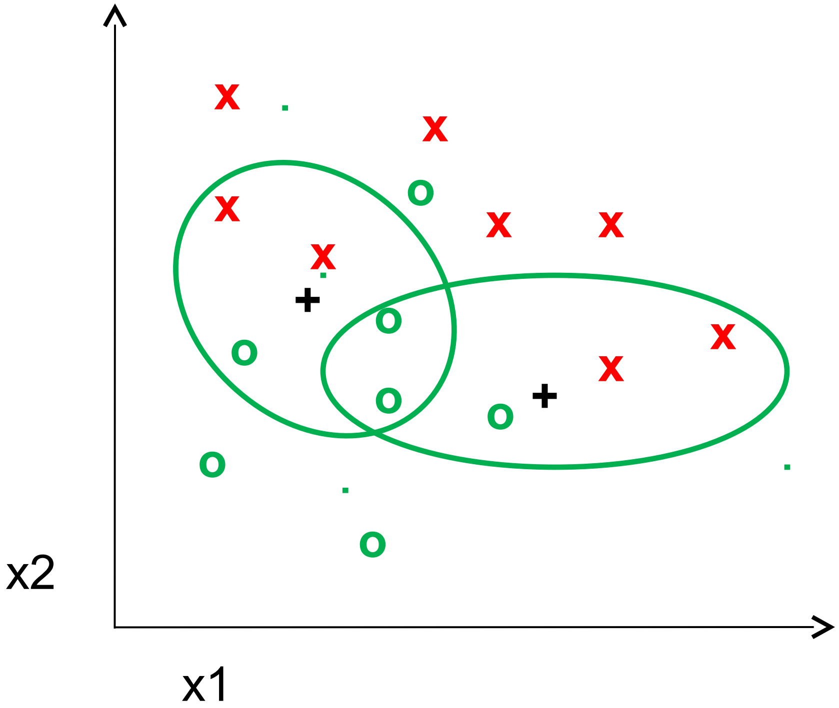
Slide Credit: D. Hoiem

3-nearest neighbor



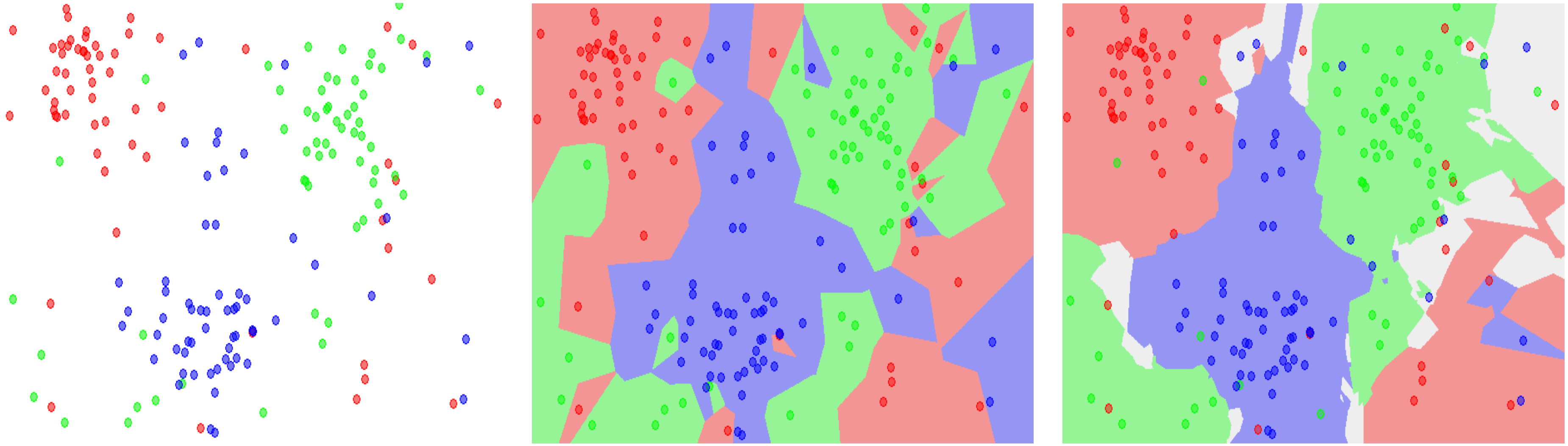
Slide Credit: D. Hoiem

5-nearest neighbor



Slide Credit: D. Hoiem

K-NN Classifiers

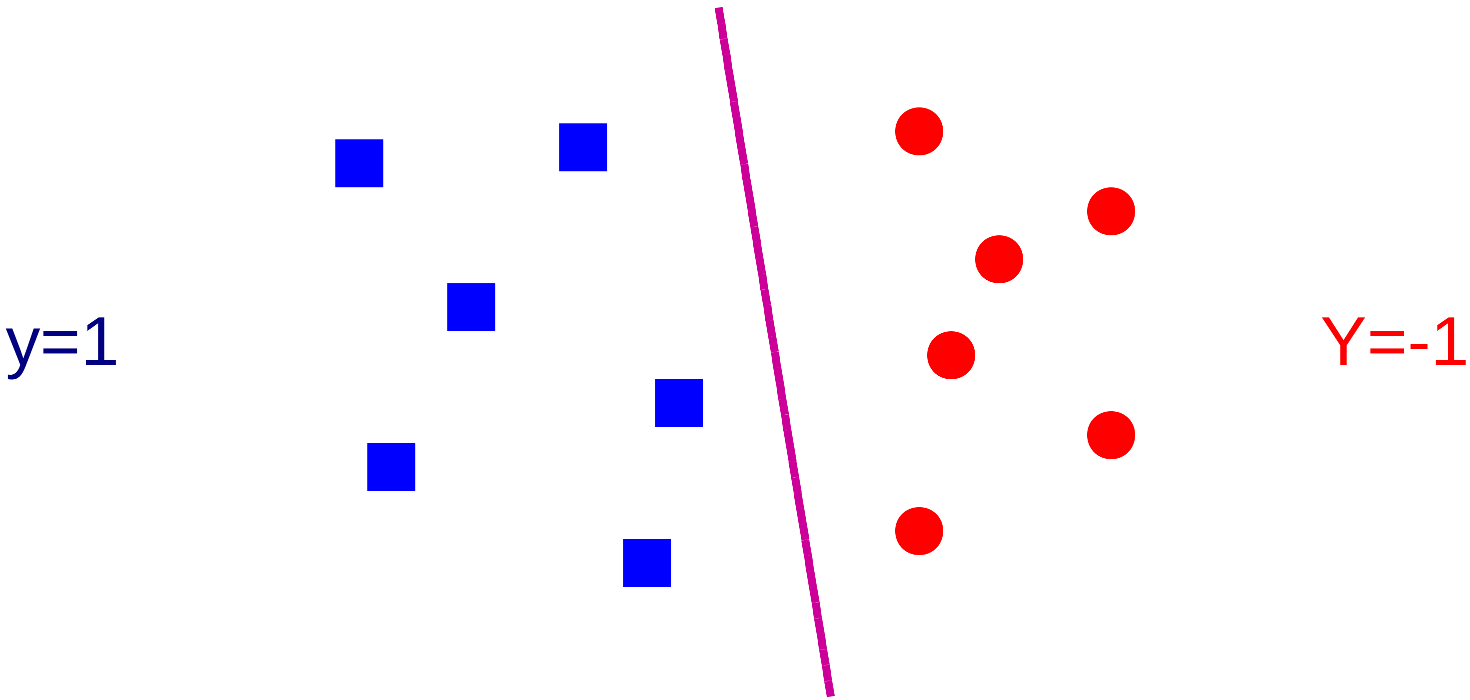


the data 1-NN classifier 5-NN classifier

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



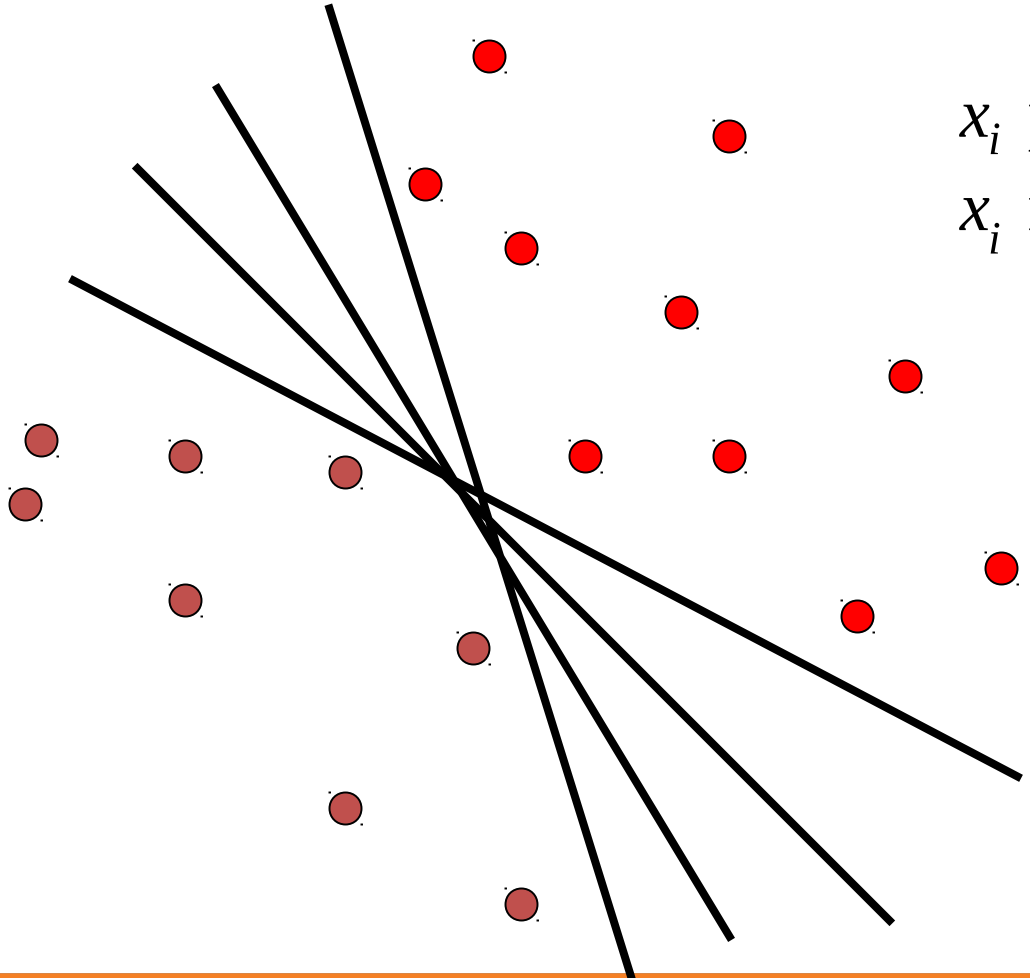
- Find a *linear function* to separate the classes:

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

 Slide Credit: Slide credit: L. Lazebnik

Linear classifiers

- Find linear function to separate positive and negative examples



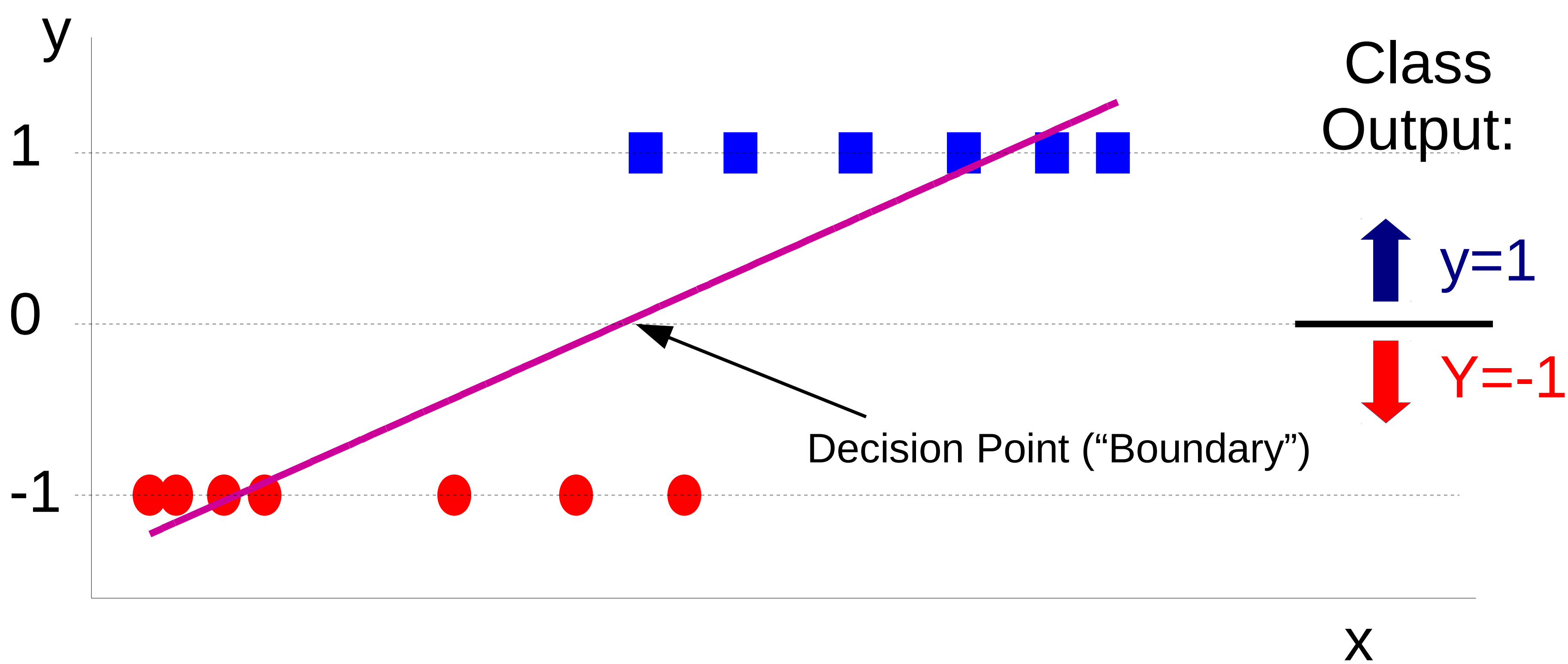
x_i positive: $x_i \cdot w + b \geq 0$

x_i negative: $x_i \cdot w + b < 0$

Which line is best?

Slide Credit: Kristin Grauman

Using Least Squares for Classification

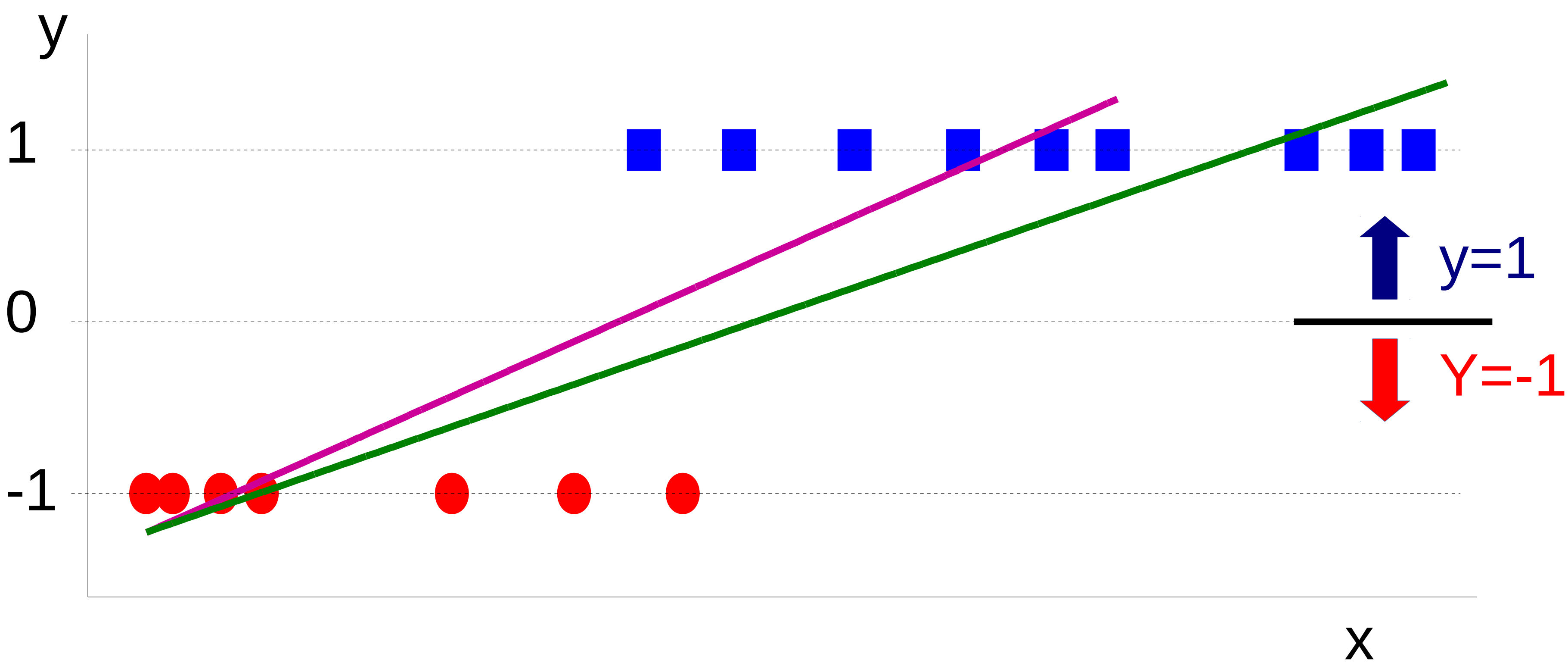


- Find a *linear function* to separate the classes:

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

Slide Credit: Slide credit: L. Lazebnik

Using Least Squares for Classification

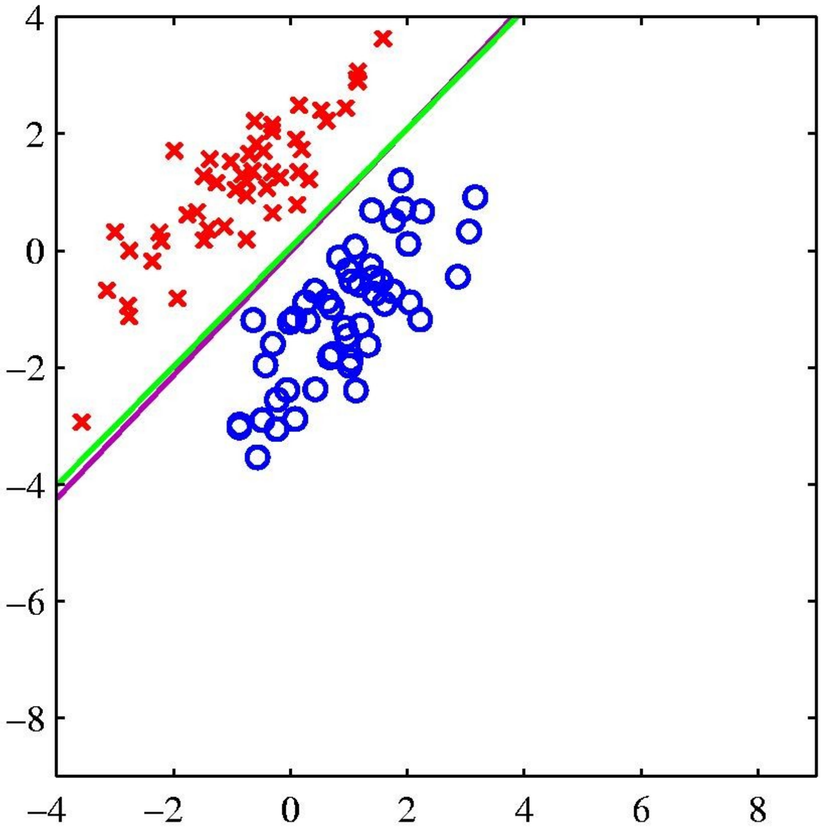


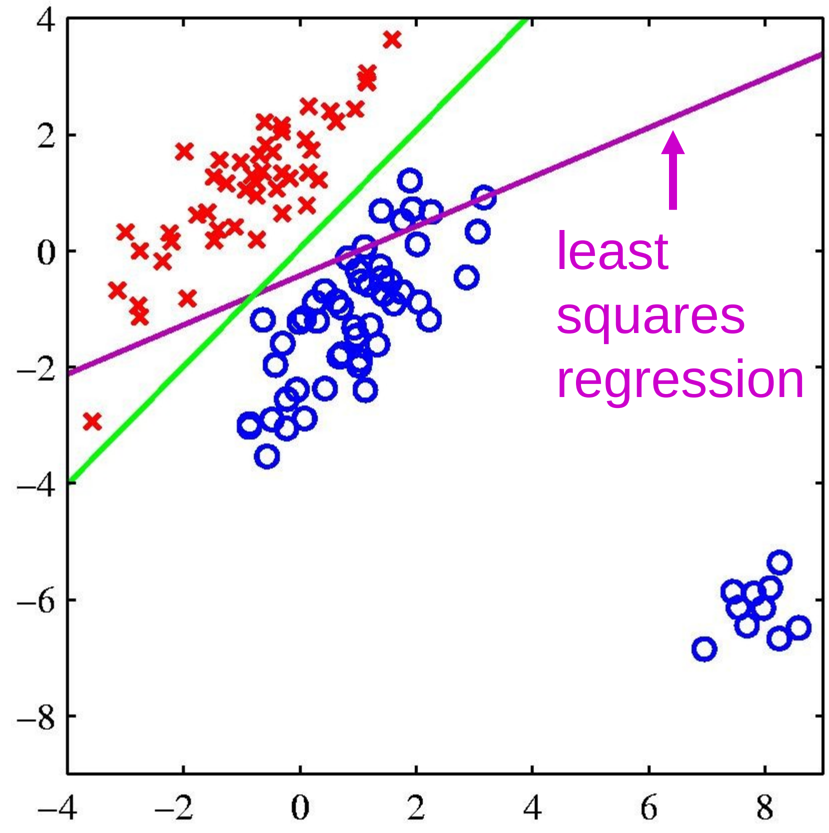
- Find a *linear function* to separate the classes:

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

Slide Credit: Slide credit: L. Lazebnik

Using least squares for classification



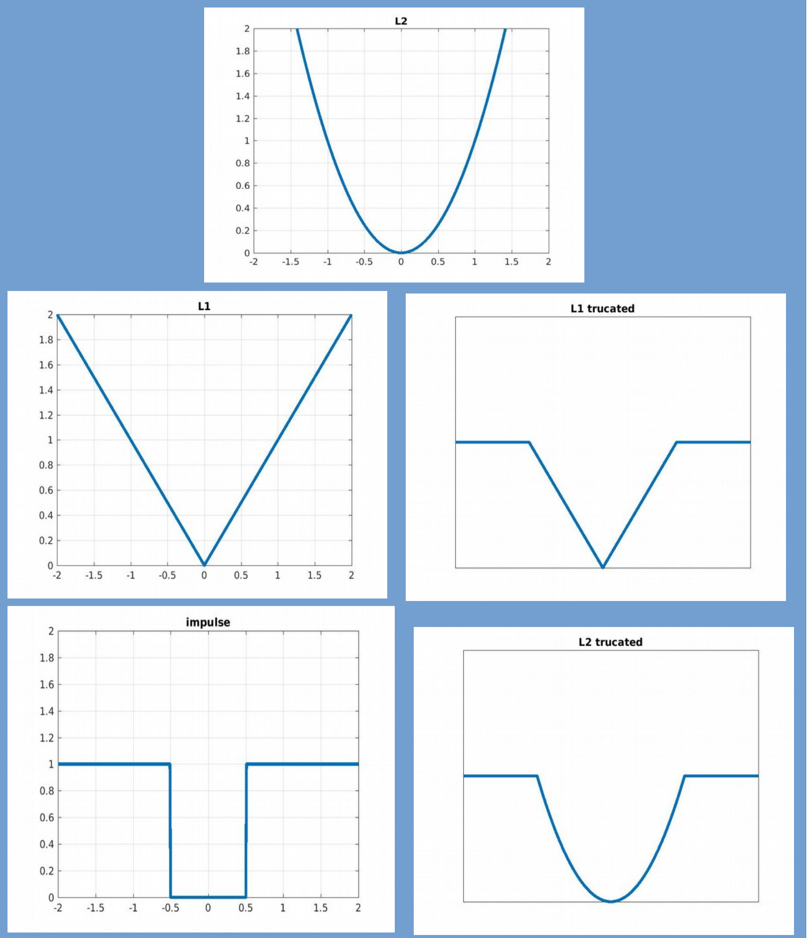


If the right answer is 1 and the model says 1.5, it loses, so it changes the boundary to avoid being “too correct”

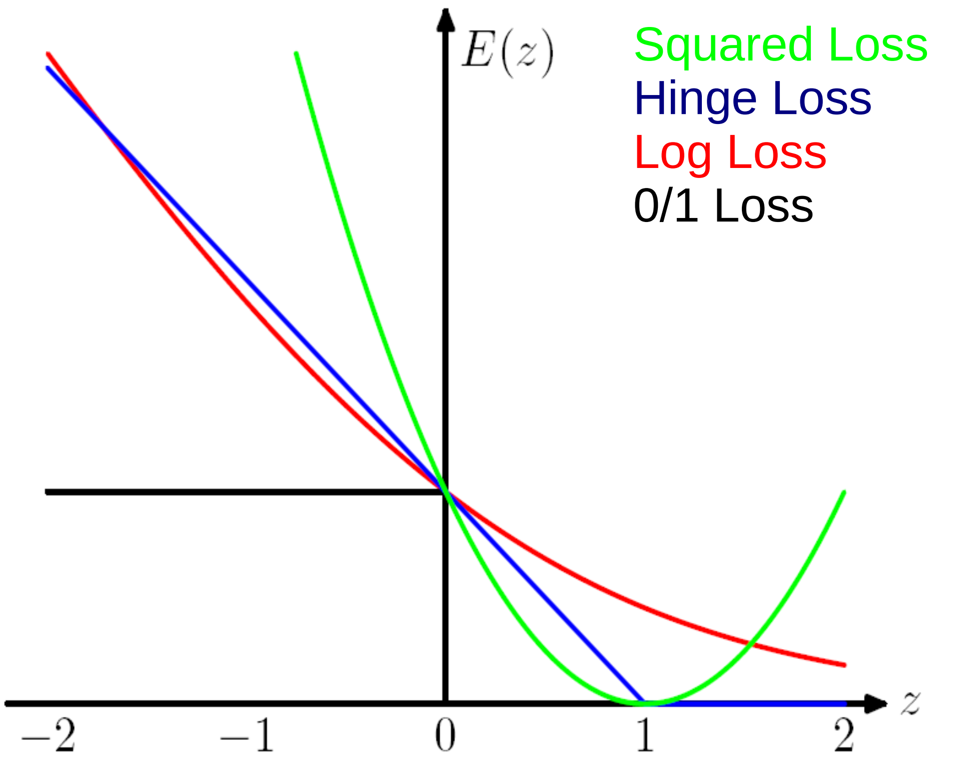
Slide Credit: J. Hinton

The Problem: Loss Function

Recall: Regression Loss Functions



Some Classification Loss Functions



Slide Credit:

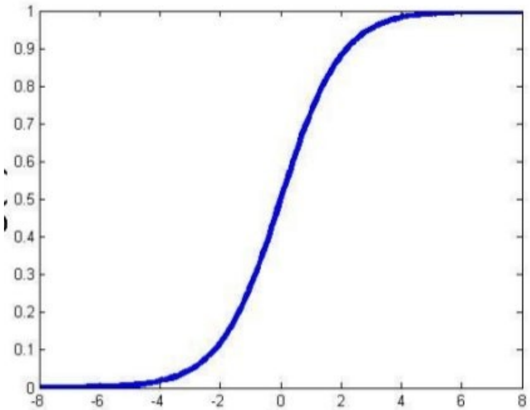
Sigmoid

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:

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



- 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 $\hat{y} = +1$.

Slide Credit: Ziv-Bar Joseph

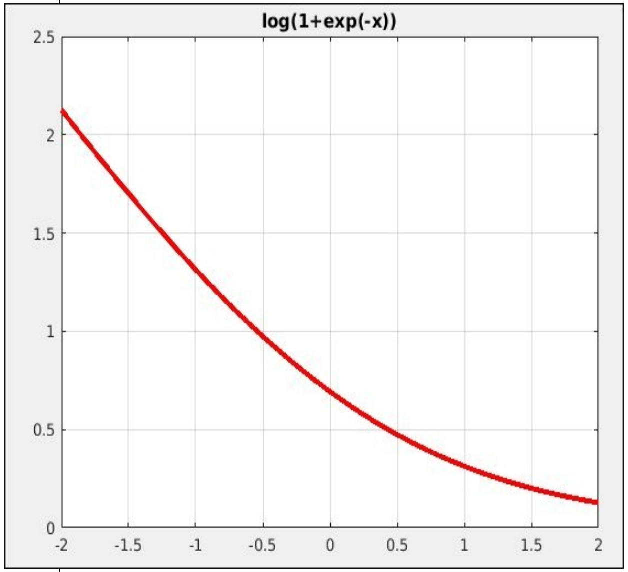
Log Loss

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:

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



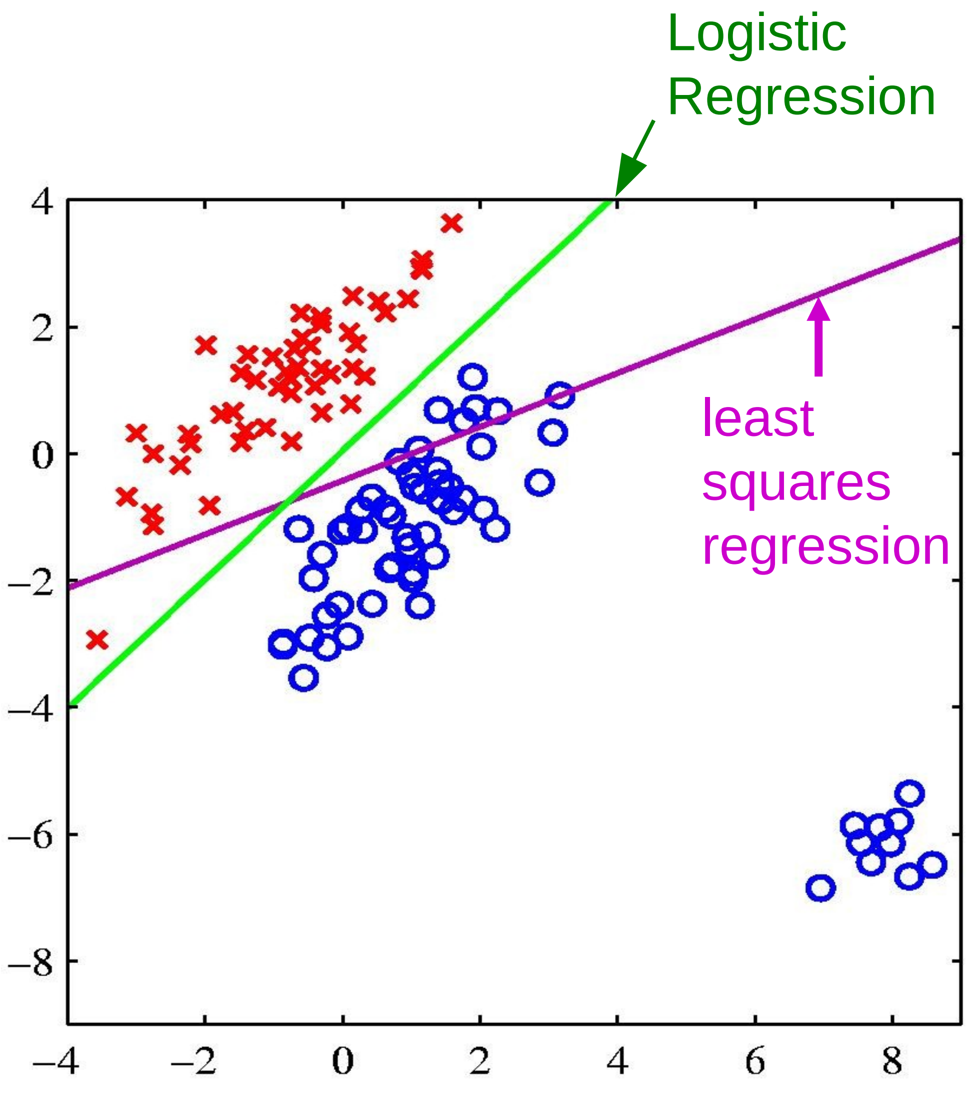
In practice, we may consider adding a regularization term

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

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

Slide Credit: Ziv-Bar Joseph

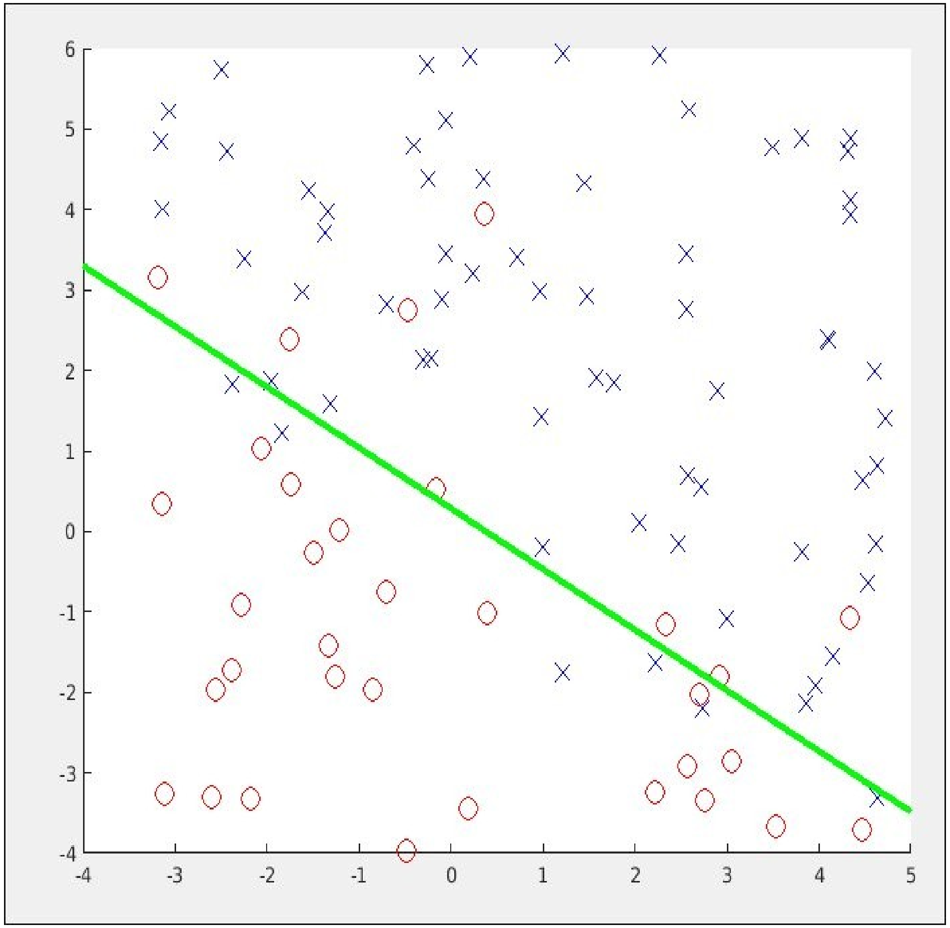
Logistic Result



Slide Credit:

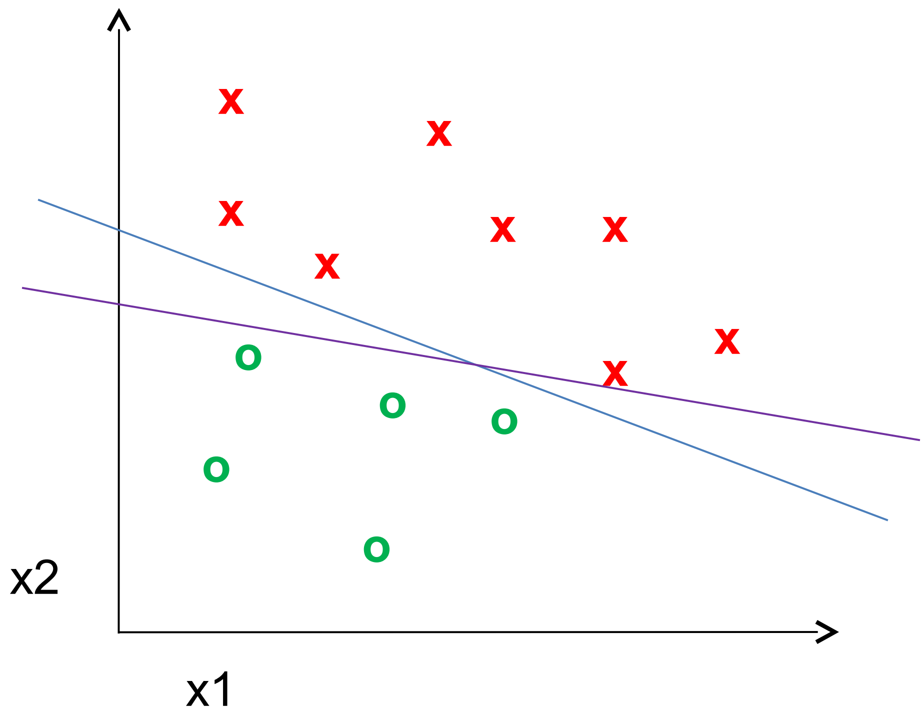
Using Logistic Regression

- 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



Slide Credit:

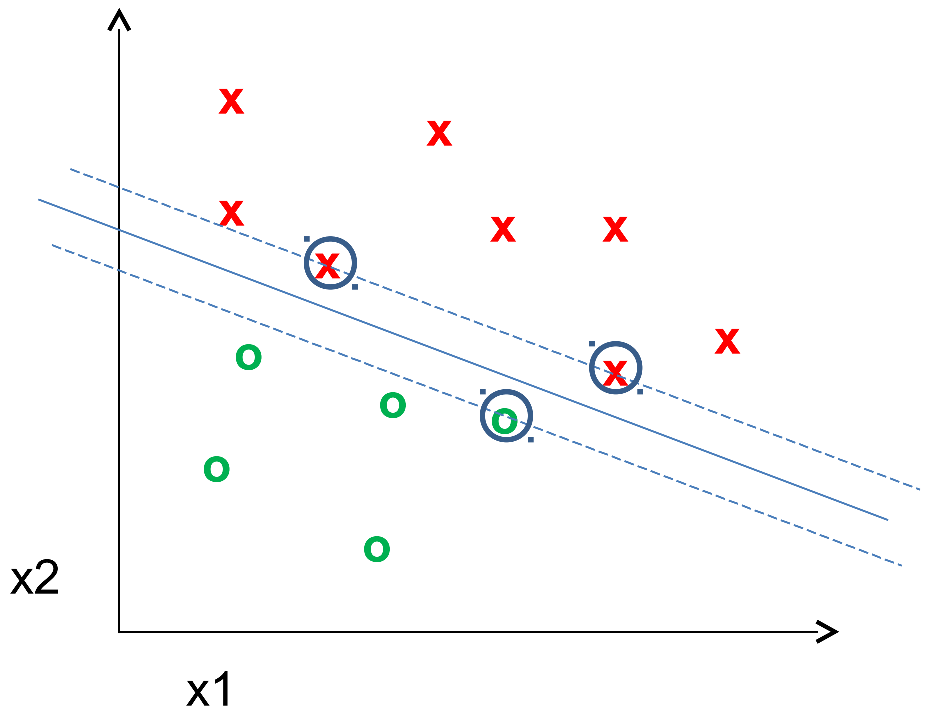
Classifiers: Linear SVM



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

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

Classifiers: Linear SVM

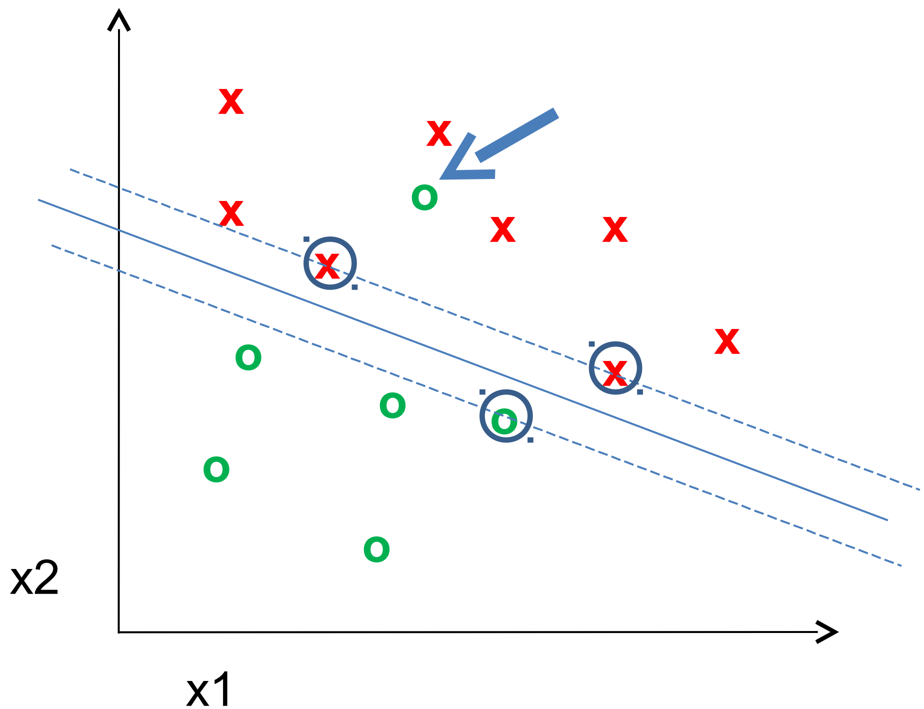


- Find a *linear function* to separate the classes:

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

Slide Credit:

Classifiers: Linear SVM



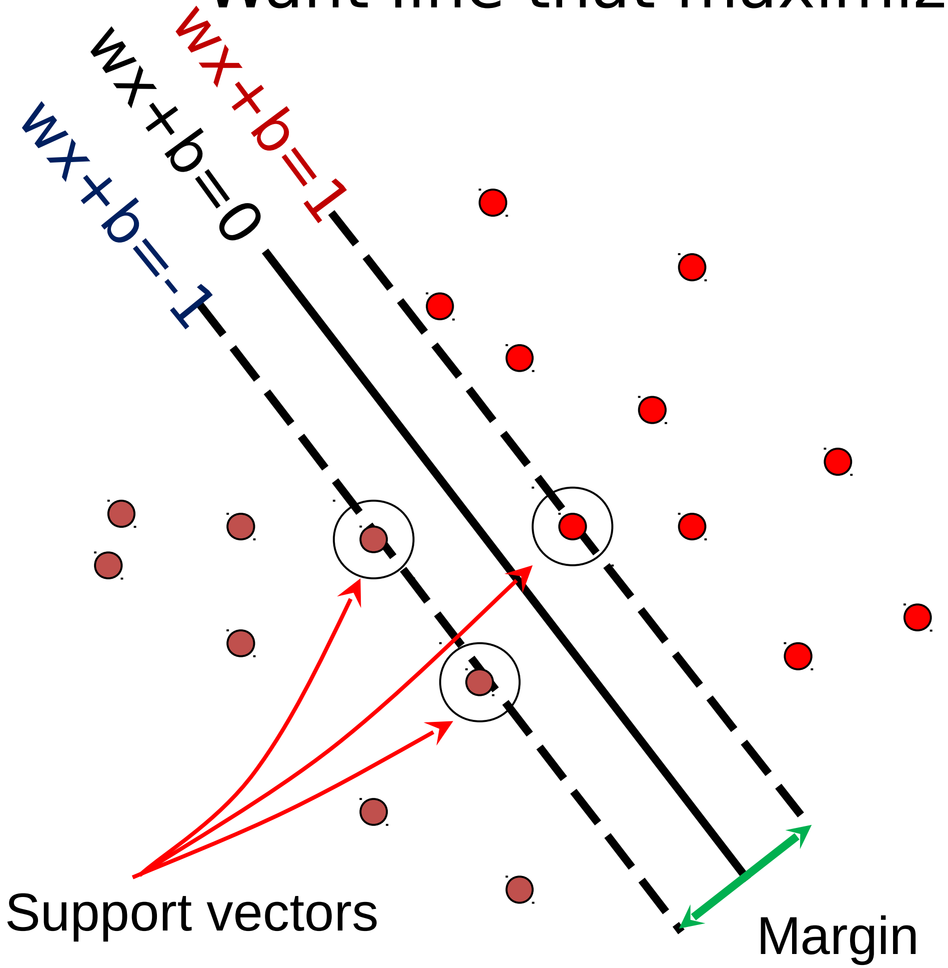
- Find a *linear function* to separate the classes:

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

Slide Credit:

Support vector machines: Margin

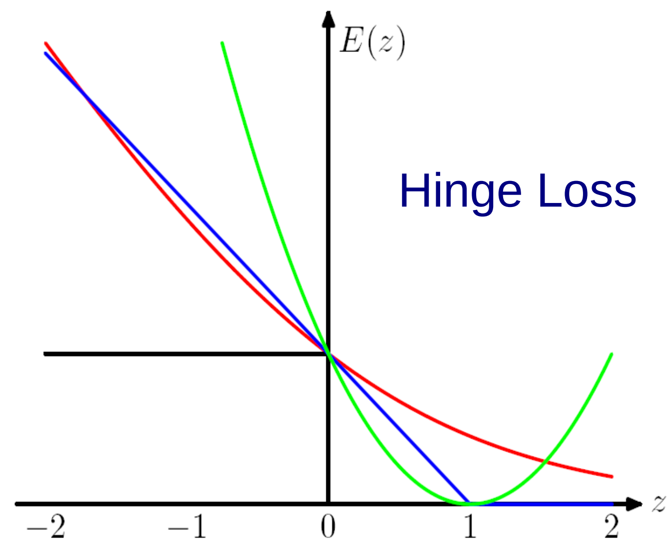
- Want line that maximizes the margin.



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



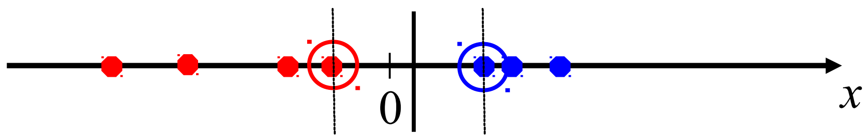
L(y,f(x)) = max(0,1−y·f(x))

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

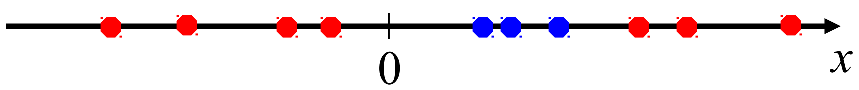
Slide Credit: Kristin Grauman

Nonlinear SVMs

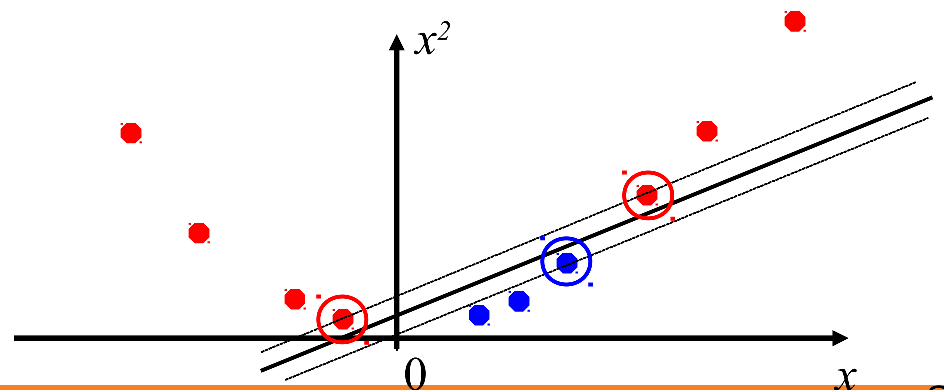
- Datasets that are linearly separable work out great:



- But what if the dataset is just too hard?



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

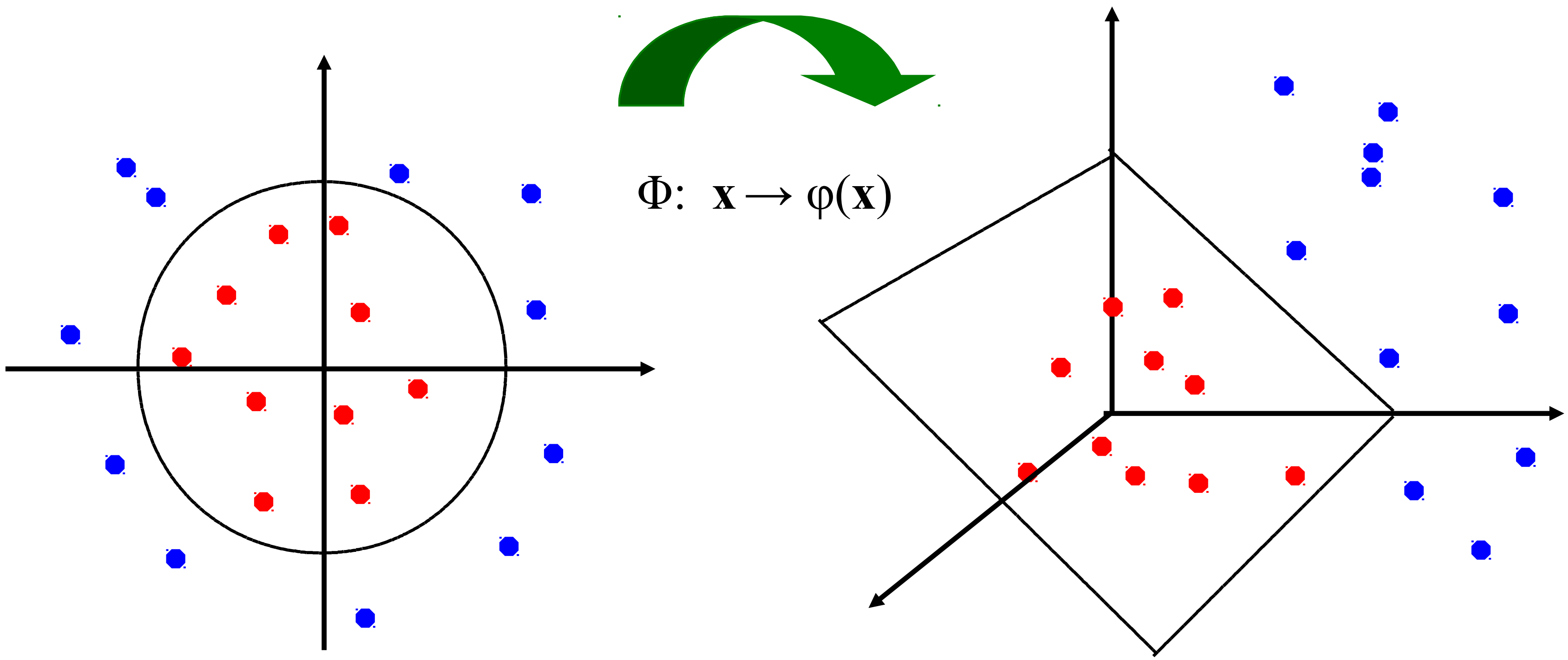


Slide credit: Andrew Moore

Slide Credit:

Nonlinear SVMs

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



Slide credit: Andrew Moore

Slide Credit:

Nonlinear SVMs

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

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

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

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

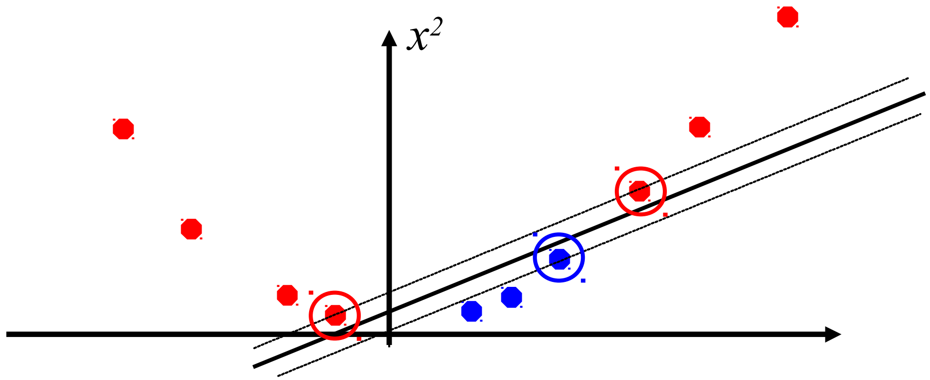
$$\sum_i \alpha_i y_i \phi(\boldsymbol{x}_i) \cdot \phi(\boldsymbol{x}) + b = \sum_i \alpha_i y_i K(\boldsymbol{x}_i, \boldsymbol{x}) + b$$

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

Slide Credit:

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.

J. Zhang, M. Marszalek, S. Lazebnik, and C. Schmid, Local Features and Kernels for Classifcation of Texture and Object Categories: A Comprehensive Study, IJCV 2007

Slide Credit:

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
  - Traning: 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

Slide Credit: Slide credit: L. Lazebnik

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