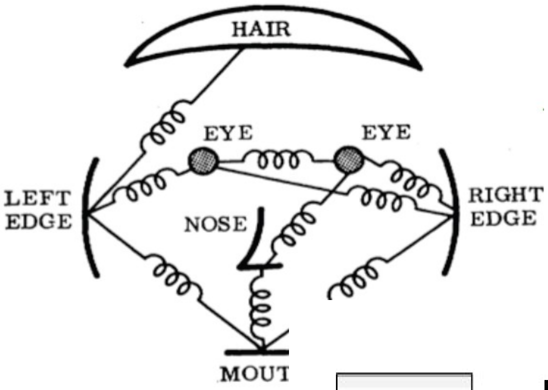
Lecture 8
Classification & Part Models

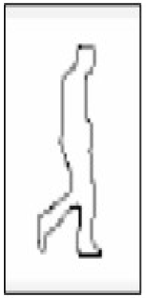
COS 429: Computer Vision

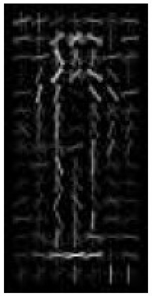
PRINCETON UNIVERSITY

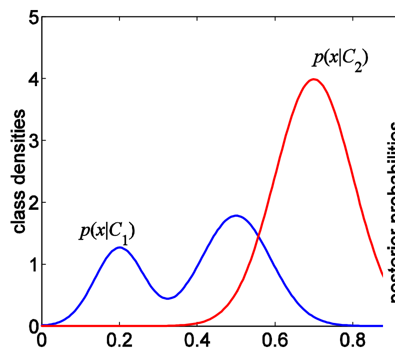
Review: Typical Components

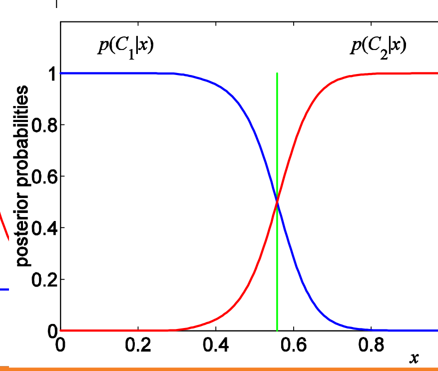
- **Hypothesis** generation
  - Sliding window, Segmentation, feature point detection, random, search
- **Encoding** of (local) image data
  - Colors, Edges, Corners, Histogram of Oriented Gradients, Wavelets, Convolution Filters
- **Relationship** of different parts to each other
  - Blur or histogram,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




Exemplar


Summary



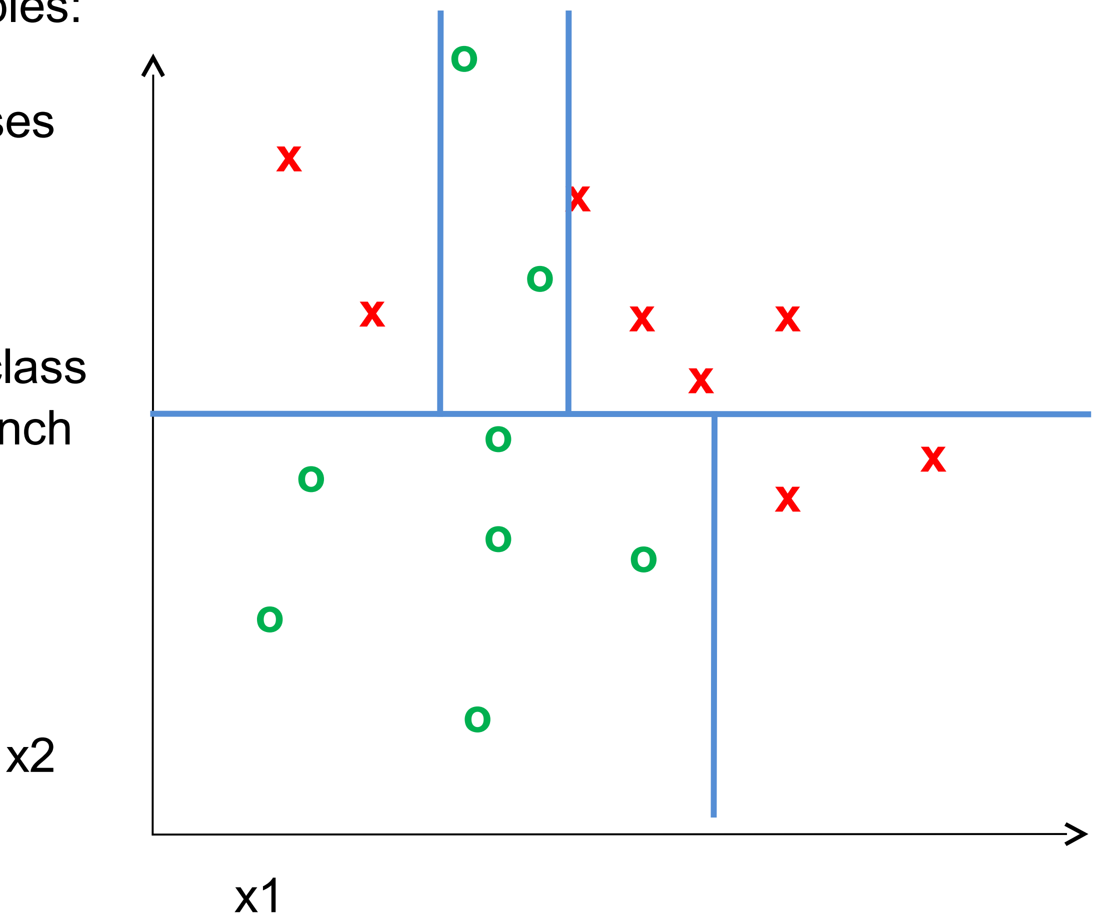


Slide Credit:

Classifiers: Decision Trees

Given (weighted) labeled examples:

- Select best single feature & threshold that separates classes
- For each branch, recurse
- Stop when
  - some depth is reached
  - Branch is (close to) single-class
  - too few examples left in branch



Ensemble Methods: Boosting

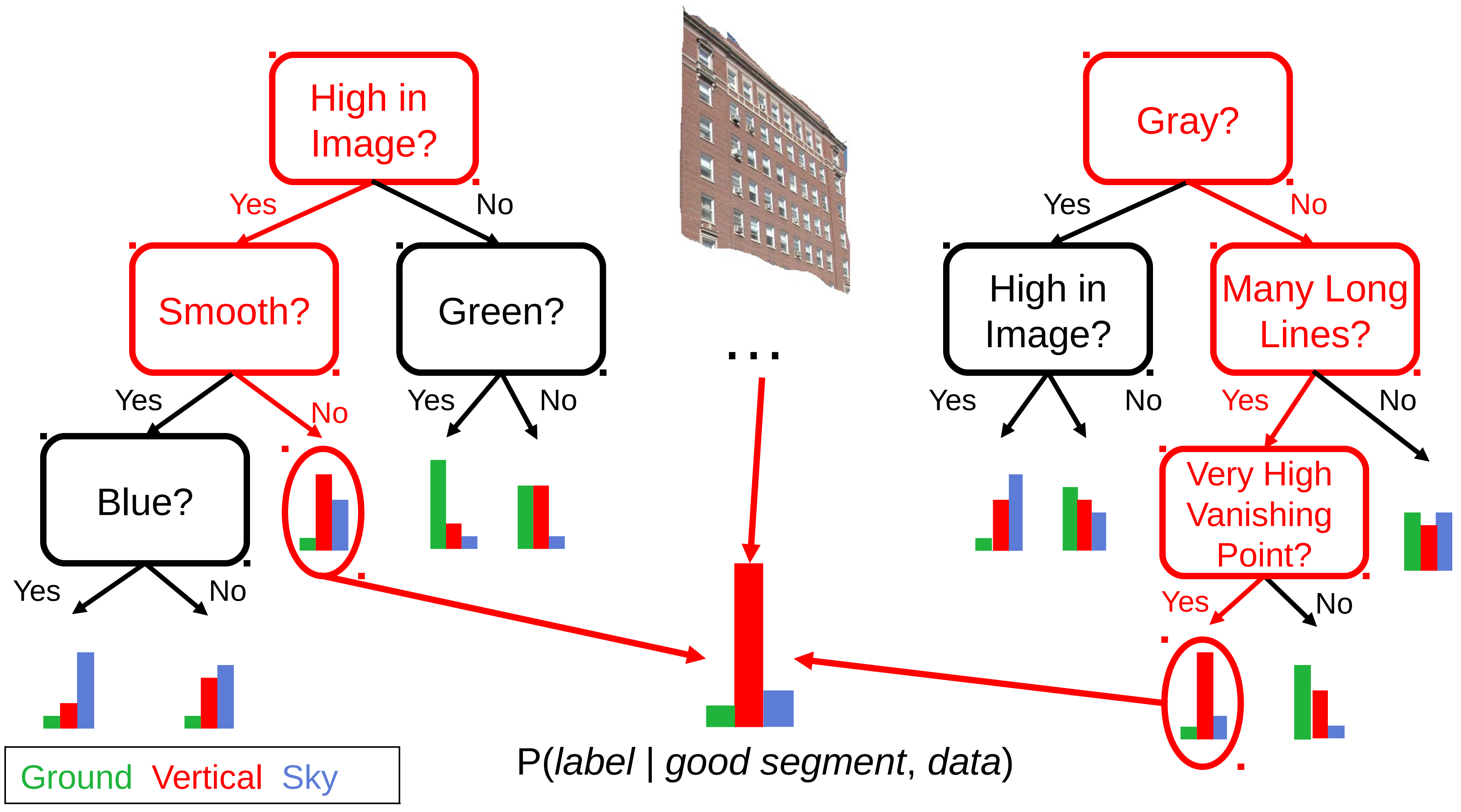
Discrete AdaBoost(Freund & Schapire 1996b)

1. Start with weights $w_i = 1/N$, $i = 1, \ldots, N$.
2. Repeat for $m = 1, 2, \ldots, M$:
   (a) Fit the classifier $f_m(x) \in \{-1, 1\}$ using weights w_i on the training data.
   (b) Compute $\text{err}_m = E_w[1_{(y \neq f_m(x))}]$, $c_m = \log((1 - \text{err}_m)/\text{err}_m)$.
   (c) Set $w_i \leftarrow w_i \exp[c_m \cdot 1_{(y_i \neq f_m(x_i))}]$, $i = 1, 2, \ldots N$, and renormalize so that $\sum_i w_i = 1$.
3. Output the classifier $\text{sign}[\sum_{m=1}^{M} c_m f_m(x)]$

figure from Friedman et al. 2000

Slide Credit:

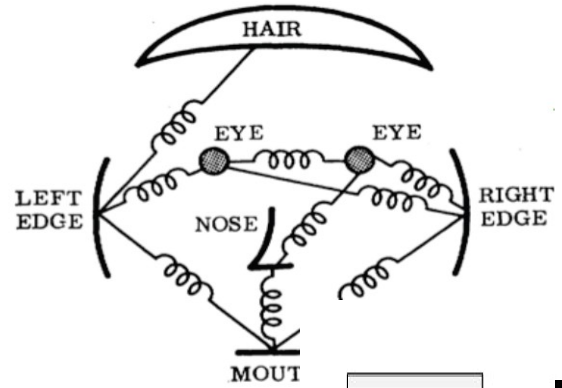
Boosted Decision Trees

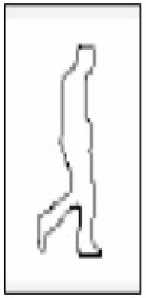


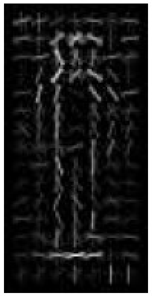
 Slide Credit: [Collins et al. 2002]

Review: Typical Components

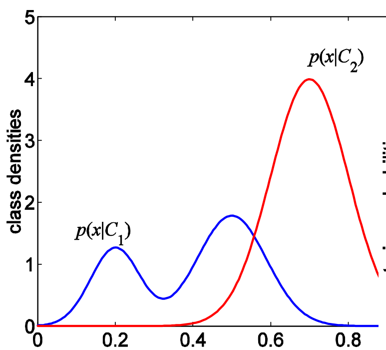
- **Hypothesis** generation
  - Sliding window, Segmentation, feature point detection, random, search
- **Encoding** of (local) image data
  - Colors, Edges, Corners, Histogram of Oriented Gradients, Wavelets, Convolution Filters
- **Relationship** of different parts to each other
  - Blur or histogram,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

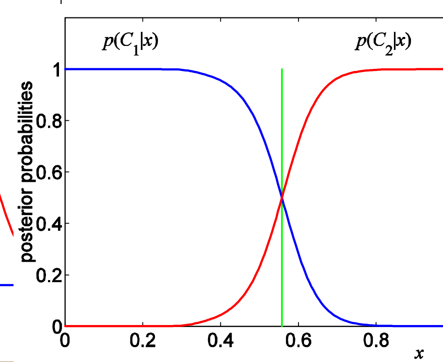






Exemplar Summary





Slide Credit:

Discriminative vs. Generative Classifiers

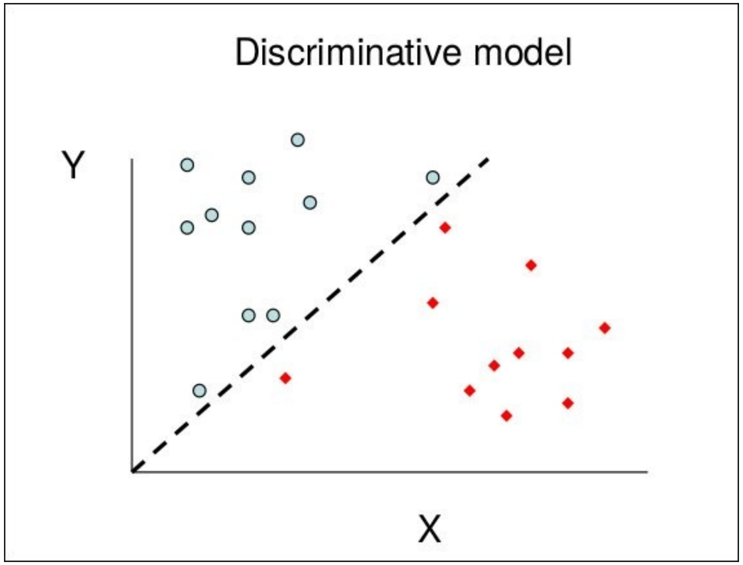
Training classifiers involves estimating f: X → Y, or P(Y|X) "Y given X"

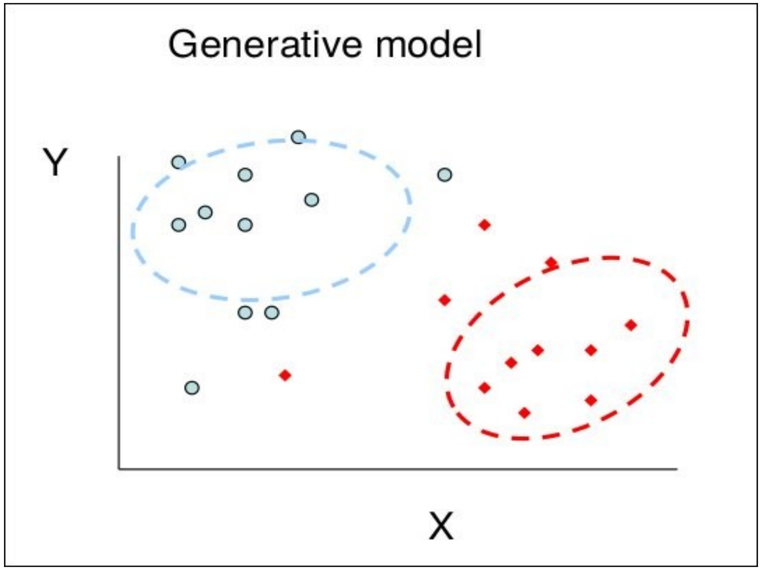
Discriminative Classification:
Find the boundary between classes

1. Assume some functional form for P(Y|X)
2. Estimate parameters of P(Y|X) directly from training data

Generative Classification:
Model each class & see which fits better

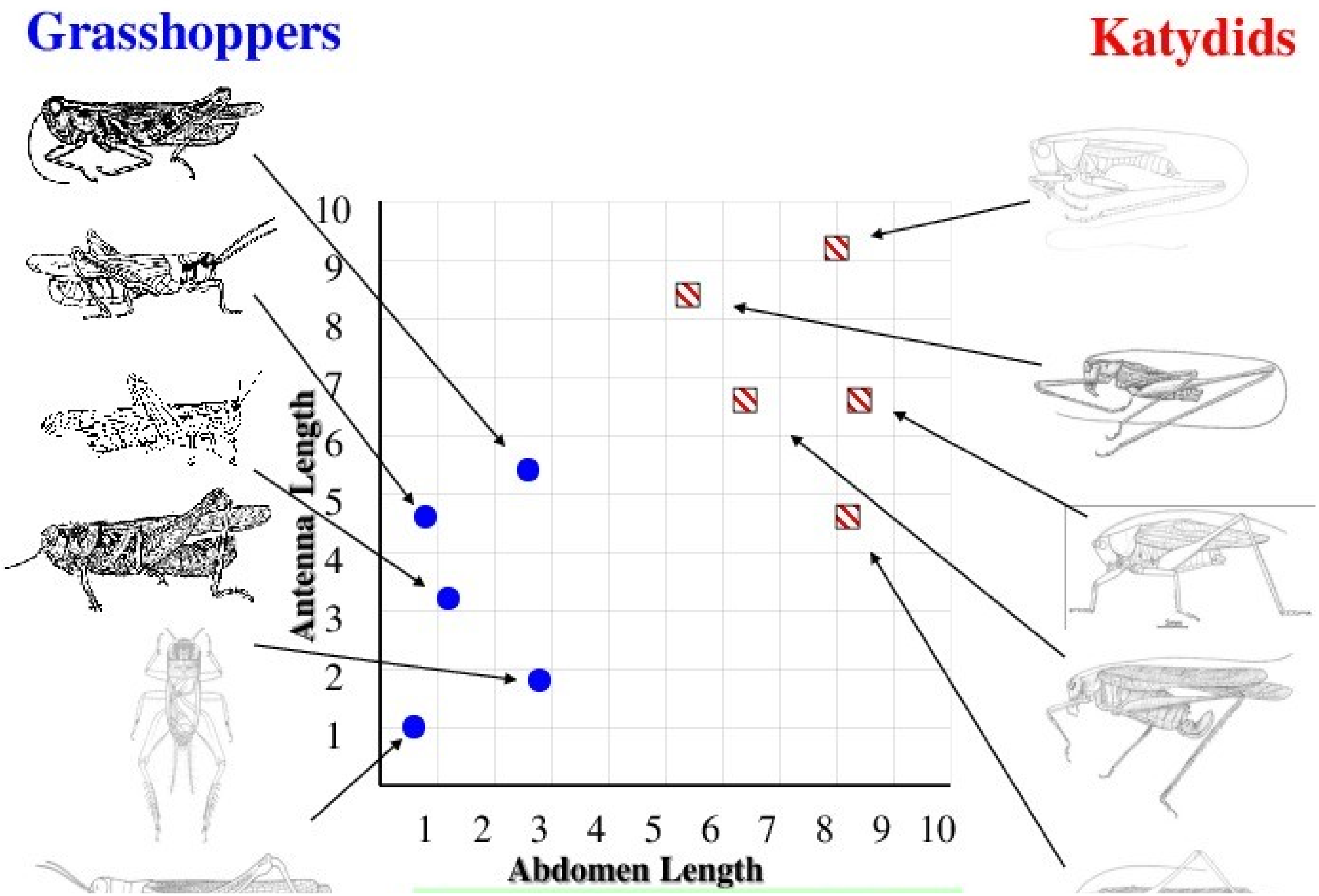
1. Assume some functional form for P(X|Y), P(X)
2. Estimate parameters of P(X|Y), P(X) directly from training data
3. Use Bayes rule to calculate $P(Y|X= x_i)$





Slide Credit:

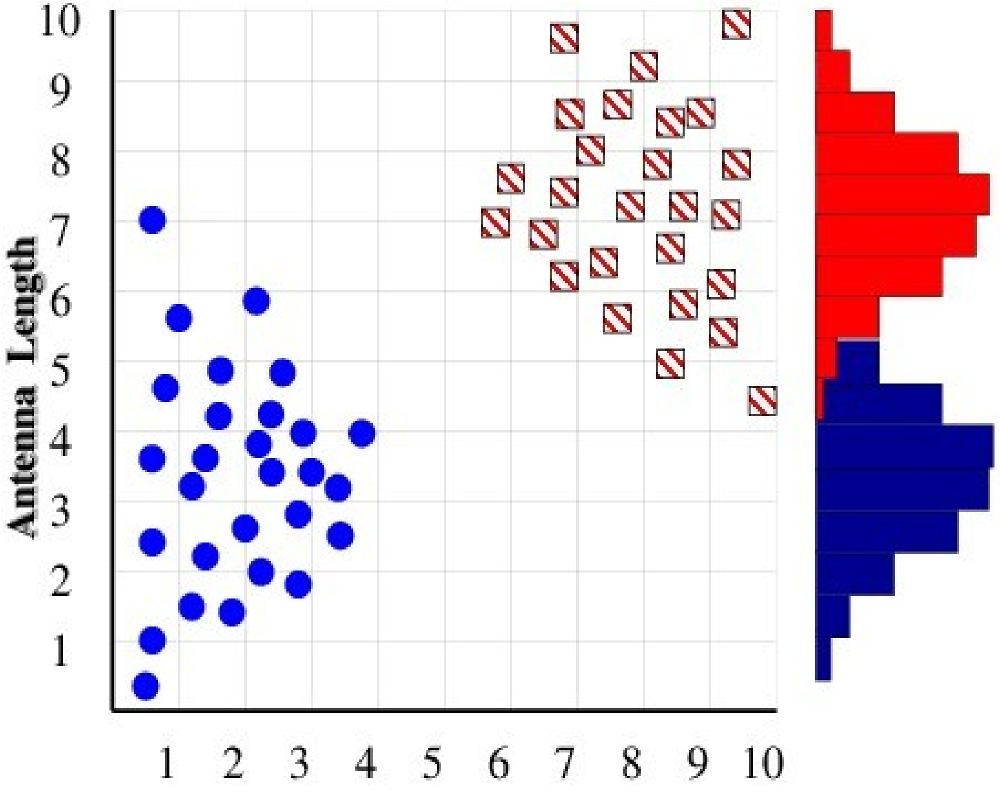
Bayesian Classification (Generative Model)



Slide Credit: Eamonn Keogh

Bayesian Classification

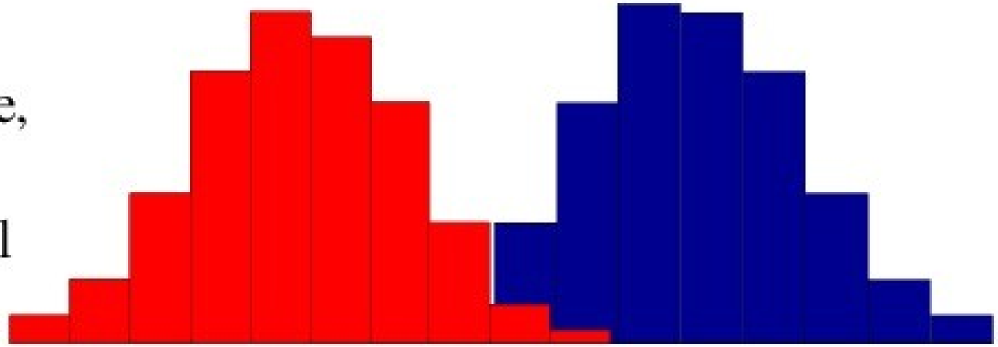
With a lot of data, we can build a histogram. Let us just build one for “Antenna Length” for now….



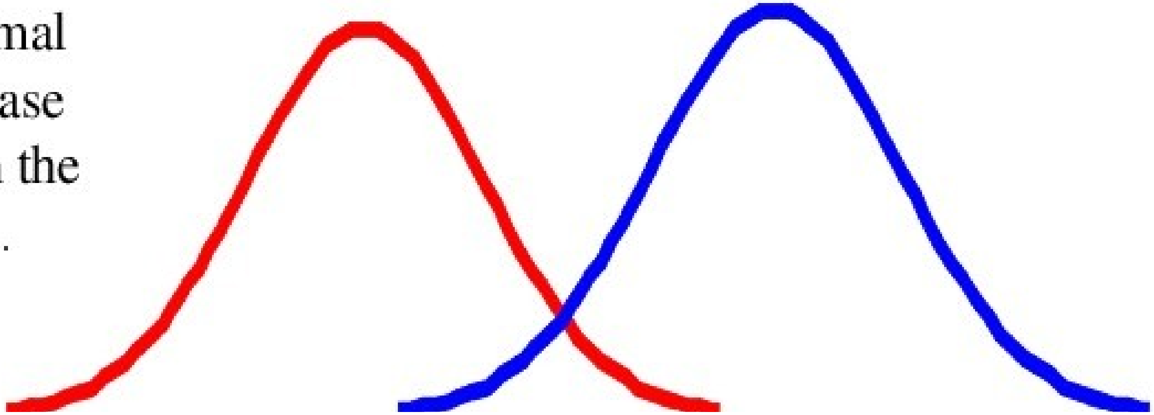
Slide Credit: Eamonn Keogh

Bayesian Classification

We can leave the histograms as they are, or we can summarize them with two normal distributions.



Let us us two normal distributions for ease of visualization in the following slides…



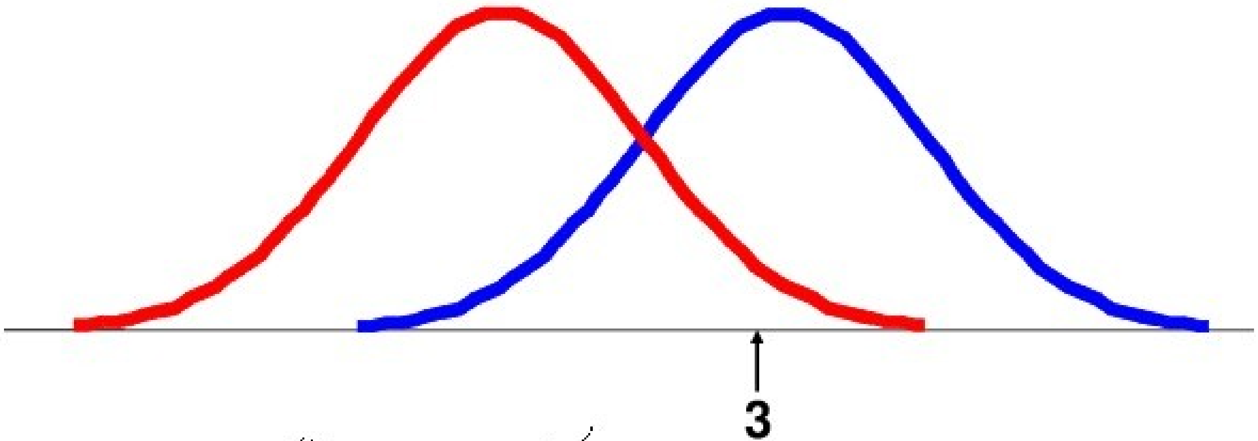
Slide Credit: Eamonn Keogh

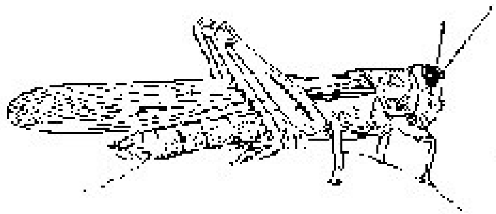
Bayesian Classification

- We want to classify an insect we have found. Its antennae are 3 units long. How can we classify it?

- We can just ask ourselves, give the distributions of antennae lengths we have seen, is it more probable that our insect is a **Grasshopper** or a **Katydid**.
- There is a formal way to discuss the most probable classification…

$p(c_j \mid d)$ = probability of class c_j, given that we have observed d





Antennae length is **3**

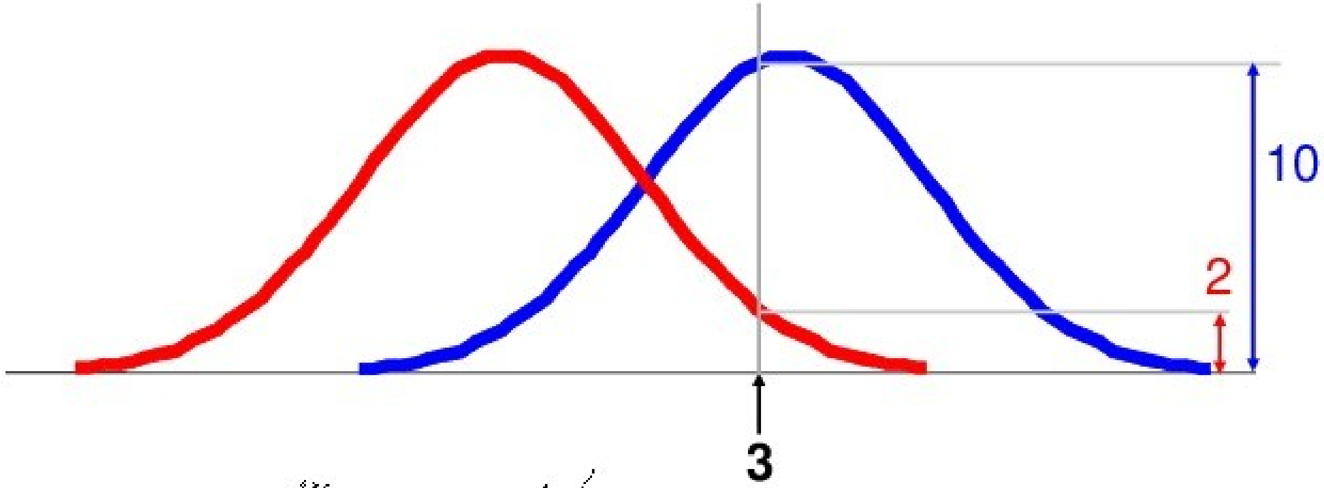
Slide Credit: Eamonn Keogh

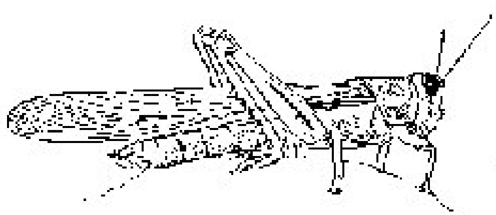
Bayesian Classification

$p(c_j \mid d)$ = probability of class c_j, given that we have observed d

P(**Grasshopper** | **3**) = 10 / (10 + 2) = 0.833

P(**Katydid** | **3**) = 2 / (10 + 2) = 0.166





Antennae length is **3**

Slide Credit: Eamonn Keogh

Bayesian Classification

- Bayesian classifiers use **Bayes theorem**, which says

$$p(c_j \mid d) = \frac{p(d \mid c_j)\, p(c_j)}{p(d)}$$

- $p(c_j \mid d)$ = probability of instance d being in class c_j,
  This is what we are trying to compute
- $p(d \mid c_j)$ = probability of generating instance d given class c_j,
  We can imagine that being in class c_j, causes you to have feature d with some probability
- $p(c_j)$ = probability of occurrence of class c_j,
  This is just how frequent the class c_j, is in our database
- $p(d)$ = probability of instance d occurring

  This can actually be ignored, since it is the same for all classes

 Slide Credit: Eamonn Keogh

Naïve Bayes Classification

- To simplify the task, **naïve Bayesian classifiers** assume attributes have independent distributions, and thereby estimate

$$p(d|c_j) = p(d_1|c_j) * p(d_2|c_j) * \dots * p(d_n|c_j)$$

The probability of class c_j generating instance d, equals….

The probability of class c_j generating the observed value for feature 1, multiplied by..

The probability of class c_j generating the observed value for feature 2, multiplied by..

Slide Credit: Eamonn Keogh

Naïve Bayes, Odds ratio, and Logit

Assuming independence of features d1 and d2, we can classify between 2 classes {C1, C2}, by compute the ratio:

$$\frac{P(C_1|d_1,d_2)}{P(C_2|d_1,d_2)} = \frac{P(d_1|C_1)\,P(d_2|C_1)\,P(C_1)}{P(d_1|C_2)\,P(d_2|C_2)\,P(C_2)} < 1$$

This is called the Odds ratio.

It is often easier to take the log of this, called the Log Odds or Logit:

$$\log\frac{P(C_1|d_1,d_2)}{P(C_2|d_1,d_2)}$$

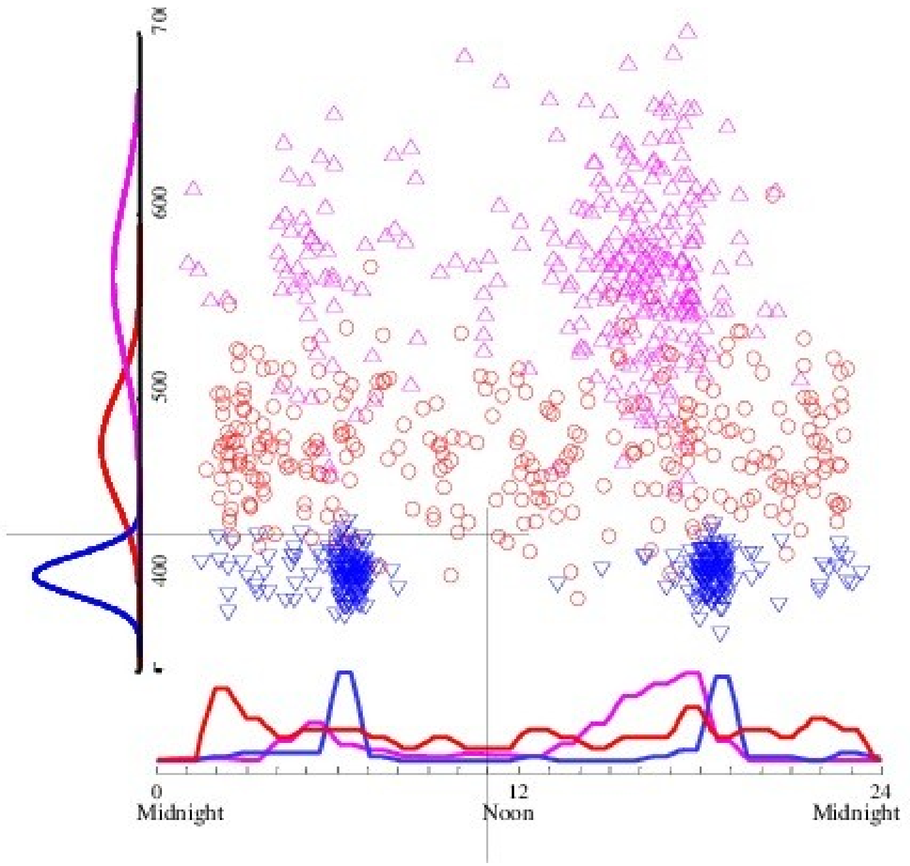
$$= \log\frac{P(d_1|C_1)}{P(d_1|C_2)} + \log\frac{P(d_2|C_1)}{P(d_2|C_2)} + \log\frac{P(C_1)}{P(C_2)} < 0$$

Slide Credit:

Naïve Bayes

Suppose I observe an insect with a wingbeat frequency of 420 at 11:00am

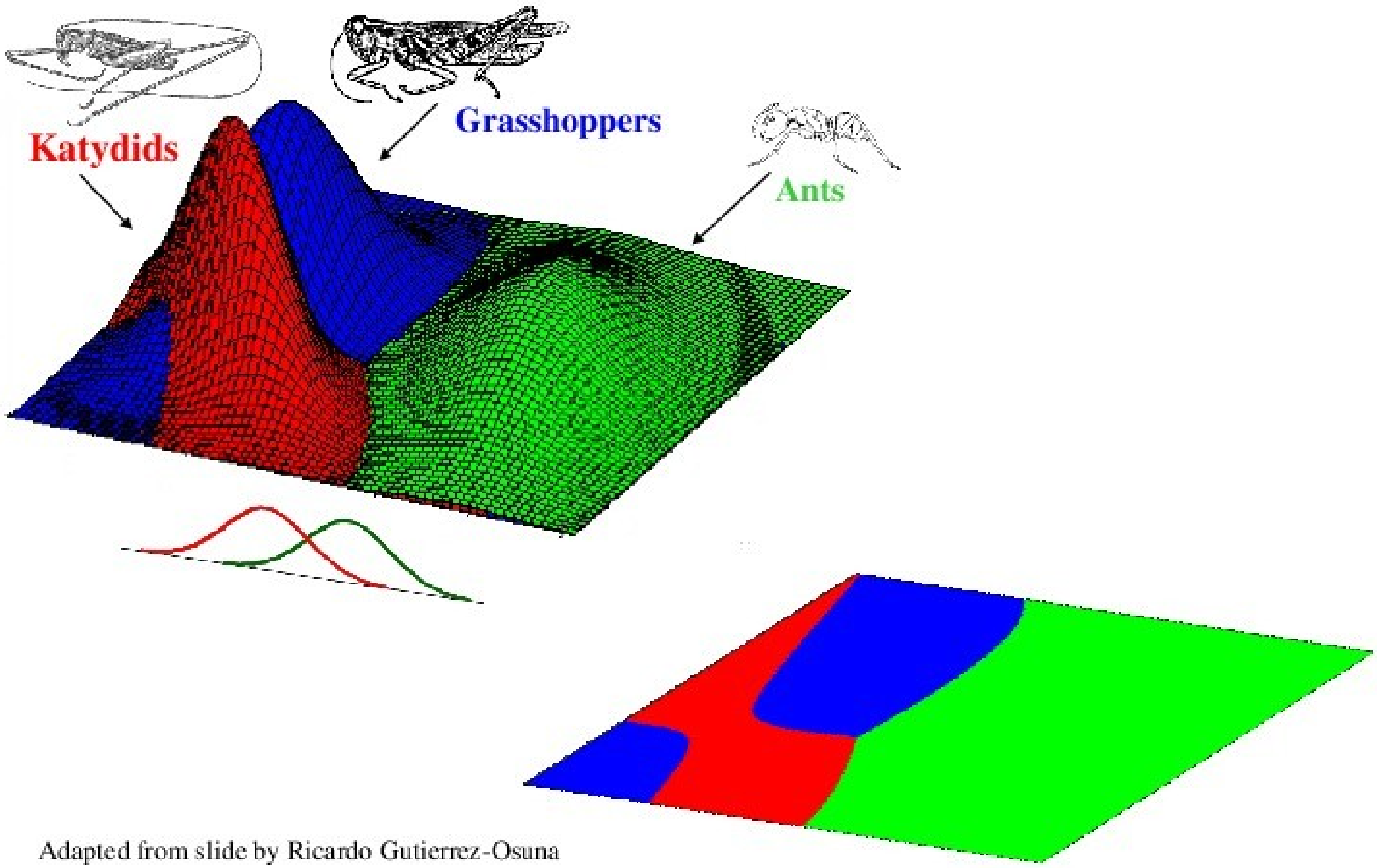
What is it?



Slide Credit: Eamonn Keogh

Naïve Bayes

Naive Bayes with Gaussian densities have piecewise quadratic decision boundary.



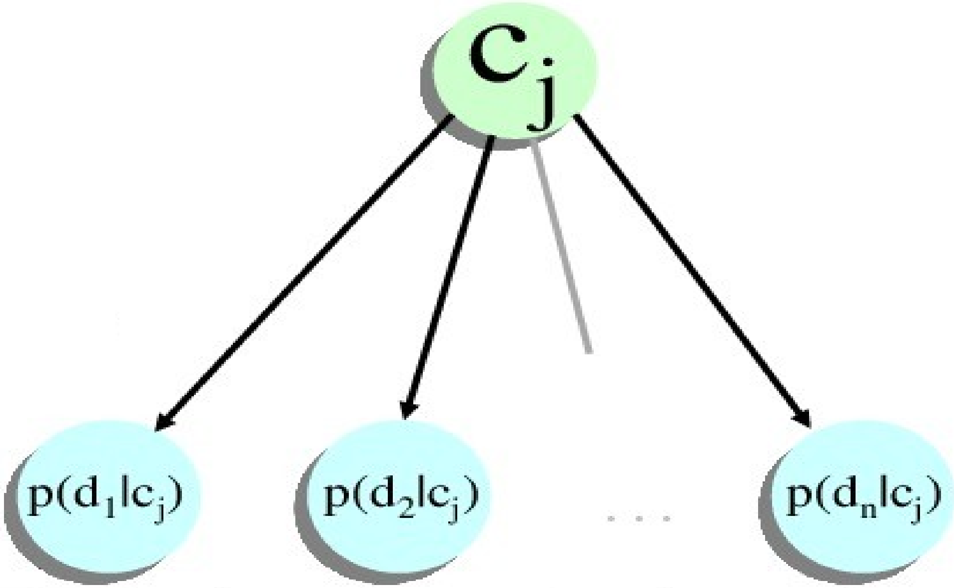
Slide Credit: Eamonn Keogh

Graphical Models

Independence assumption of Naive Bayes is the simplest assumption. Often much more complicated relationships needs to be represented.

A good way to do this is a Graphical Model
(aka. Byesian Network)

Naive Bayes assumes independence of d1,d2,... conditioned on the class c



Animal	Mass >10_{kg}	
Cat	Yes	0.15
	No	0.85
Dog	Yes	0.91
	No	0.09
Pig	Yes	0.99
	No	0.01

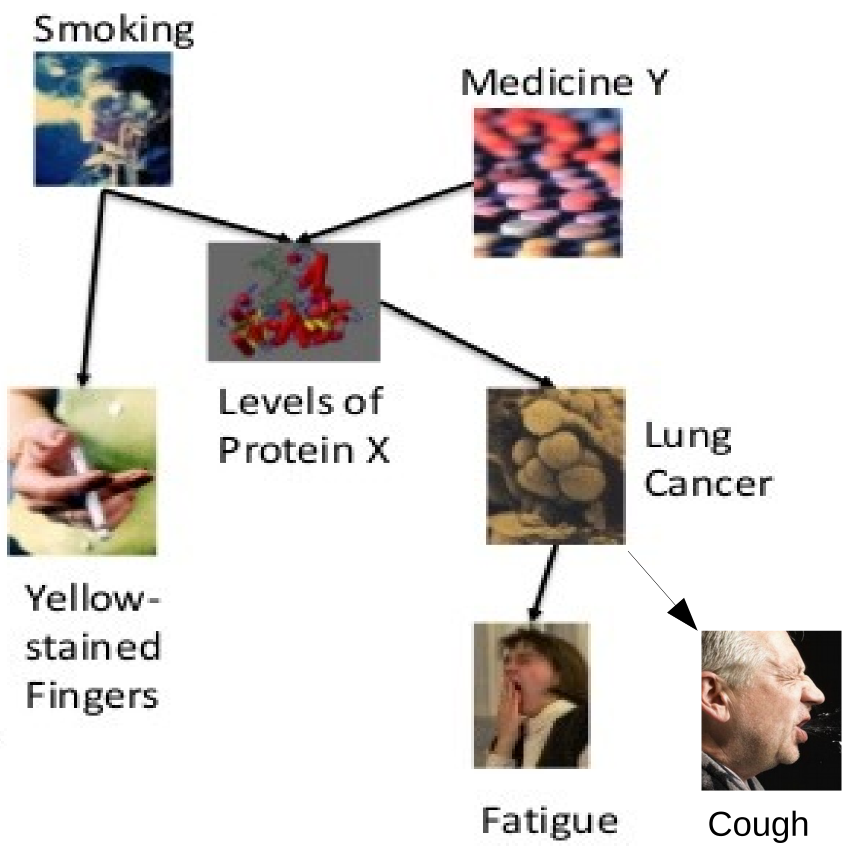
Animal	Color	
Cat	Black	0.33
	White	0.23
	Brown	0.44
Dog	Black	0.97
	White	0.03
	Brown	0.90
Pig	Black	0.04
	White	0.01

Animal
Cat
Dog
Pig

 Slide Credit: Eamonn Keogh

Graphical Models

More complicated models consider the dependence between different variables.



 Slide Credit: Ioannis Tsamardinos

Discriminative vs. Generative Classifiers

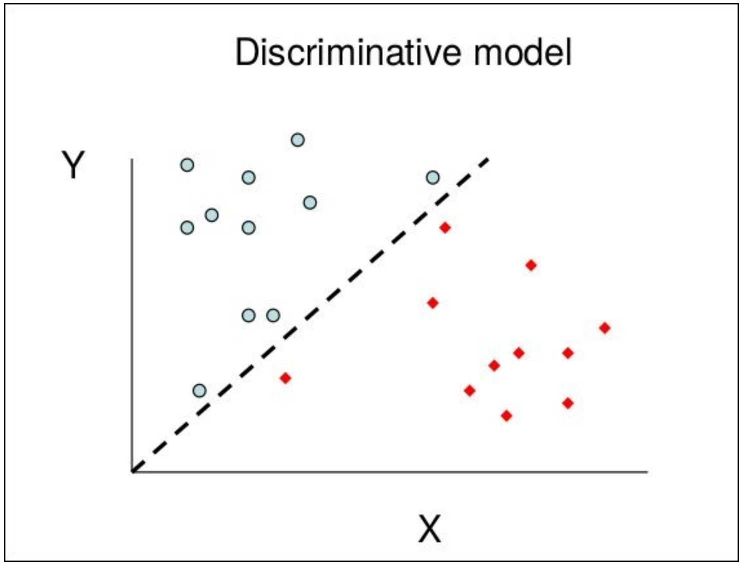
Training classifiers involves estimating f: X → Y, or P(Y|X) “Y given X”

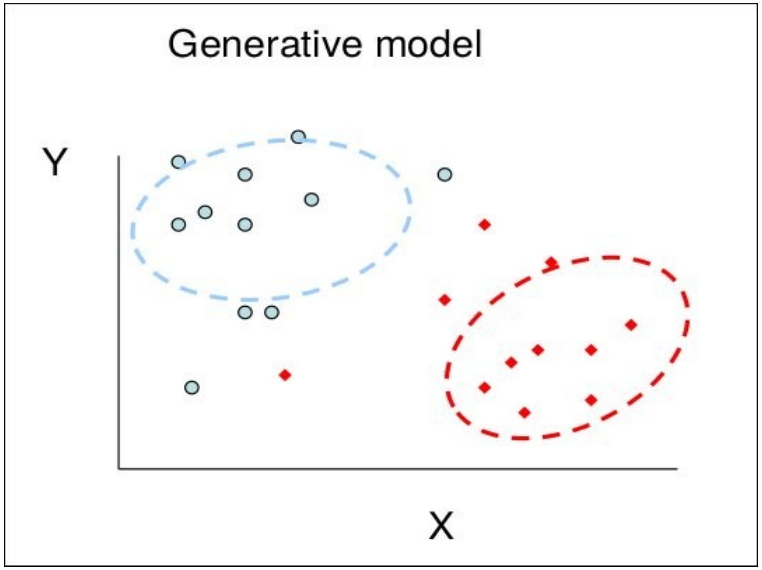
Discriminative Classification:
Find the boundary between classes

1. Assume some functional form for P(Y|X)
2. Estimate parameters of P(Y|X) directly from training data

Generative Classification:
Model each class & see which fits better

1. Assume some functional form for P(X|Y), P(X)
2. Estimate parameters of P(X|Y), P(X) directly from training data
3. Use Bayes rule to calculate P(Y|X= x_i)



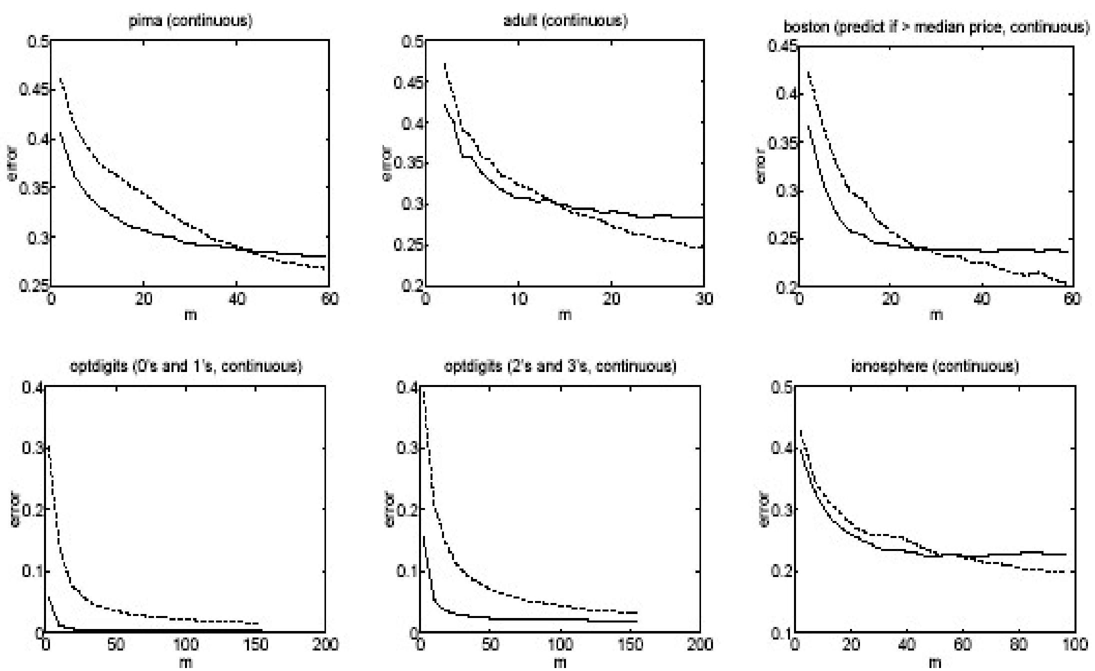


Slide Credit:

Discriminative vs. Generative Classifiers

Discriminative:
+ Model directly what you care about
+ With many examples usually more accurate
+ Often faster to evaluate, can scale well to many examples & classes

Generative:
+ Allows more flexibility to model relationships between variables
+ Can handle compositionality, missing & occluded parts (more “object oriented”)
+ Often needs less labeled examples

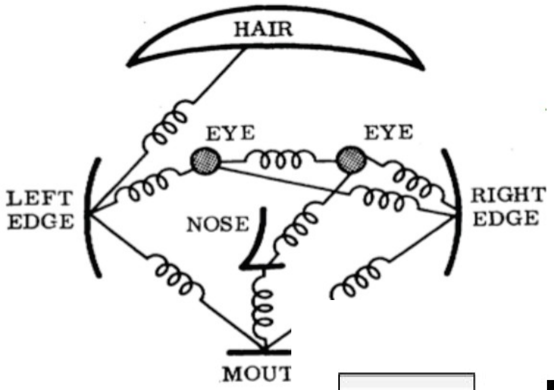


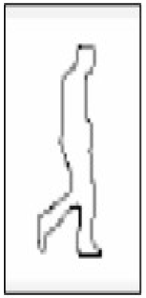
“On Discriminative vs. Generative classifiers: A comparison of logistic regression and naïve Bayes,” A. Ng and M. Jordan, NIPS 2002.

Slide Credit:

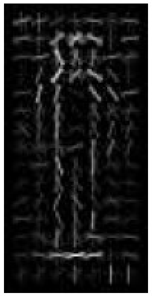
Review: Typical Components

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

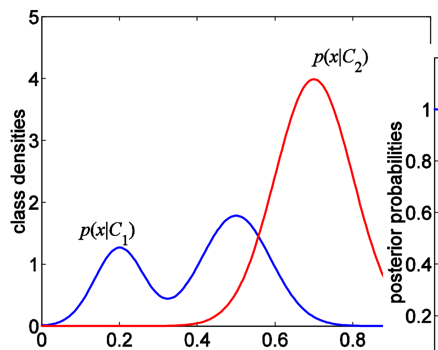


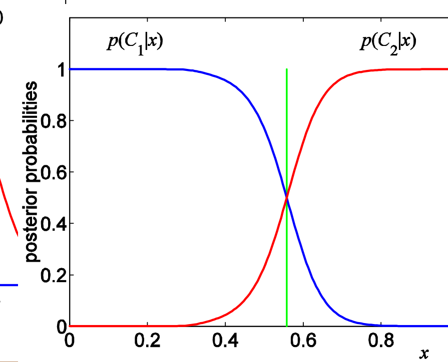


Exemplar



Summary

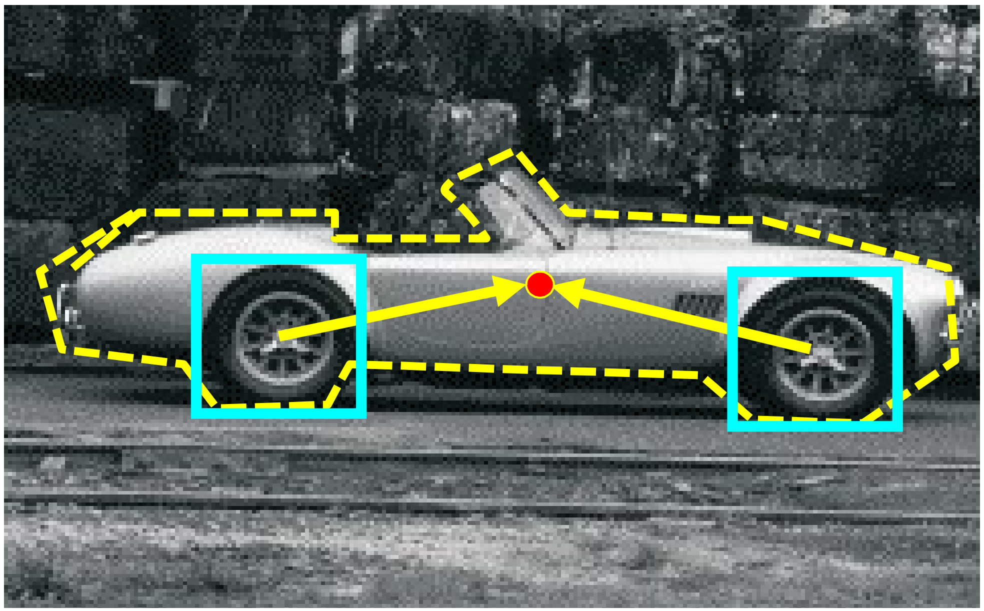


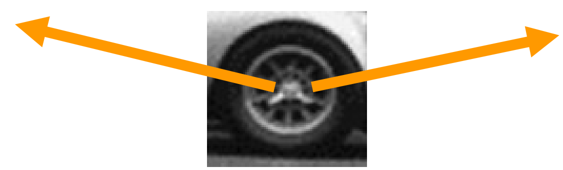


Slide Credit:

Implicit shape models

- Visual codebook is used to index votes for object position





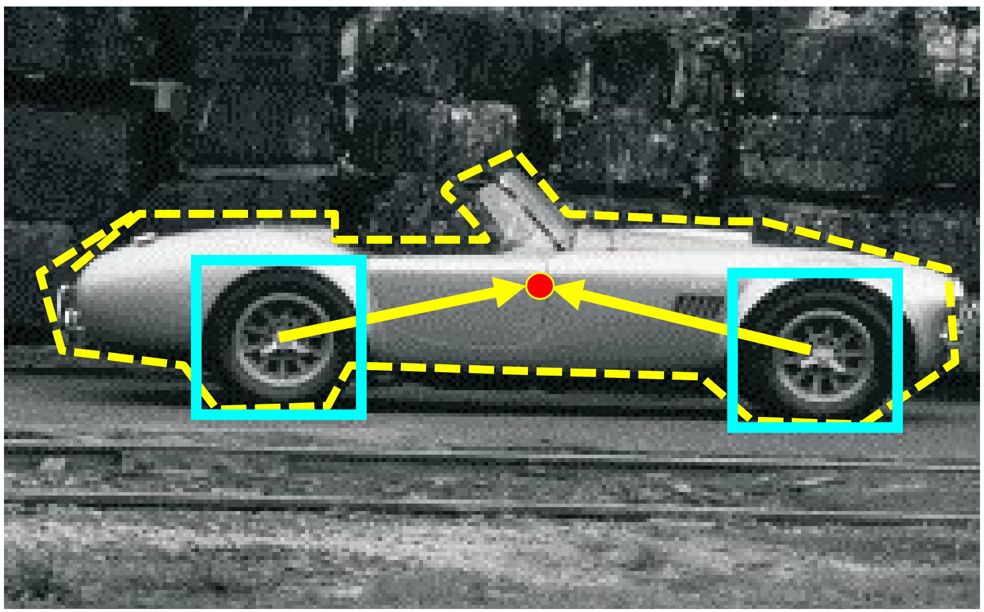
visual codeword with displacement vectors

training image annotated with object localization info

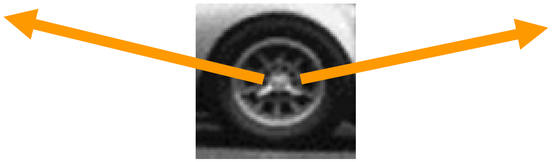
B. Leibe, A. Leonardis, and B. Schiele, Combined Object Categorization and Segmentation with an Implicit Shape Model, ECCV Workshop on Statistical Learning in Computer Vision 2004

Implicit shape models

- Visual codebook is used to index votes for object position



training image annotated with object localization info



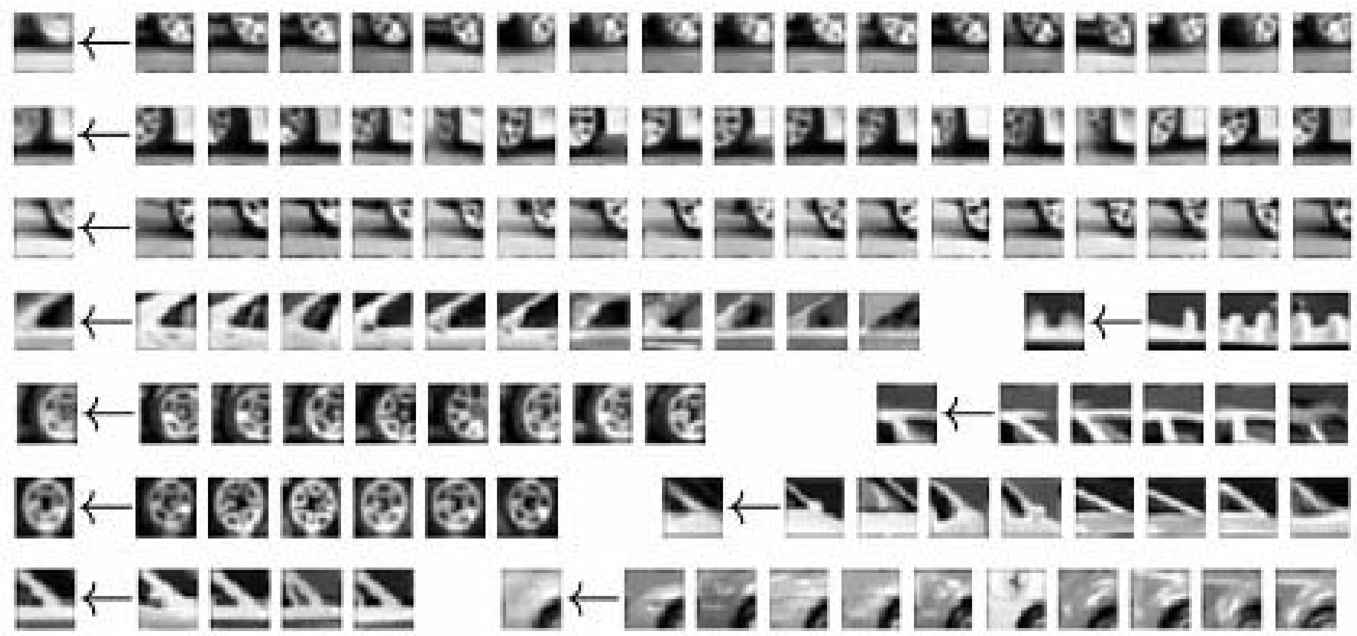
visual codeword with displacement vectors

B. Leibe, A. Leonardis, and B. Schiele, Combined Object Categorization and Segmentation with an Implicit Shape Model, ECCV Workshop on Statistical Learning in Computer Vision 2004

Slide Credit:

Implicit shape models: Training

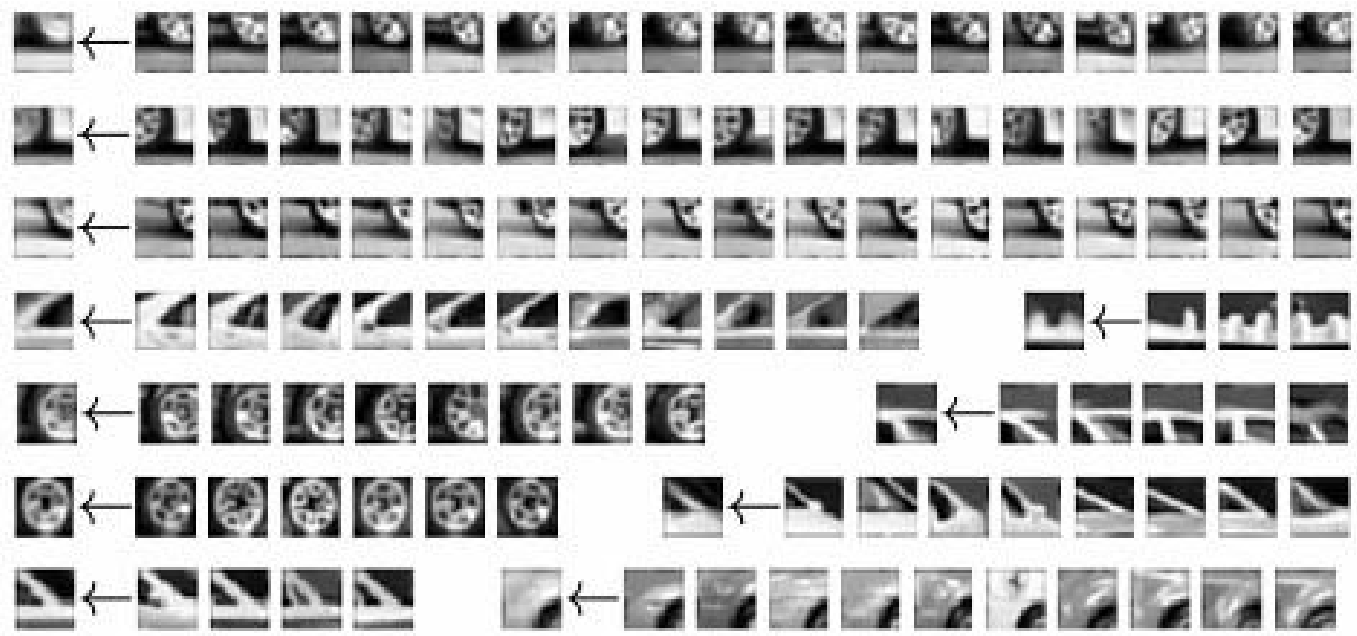
1. Build codebook of patches around extracted interest points using clustering



Slide Credit:

Implicit shape models: Training

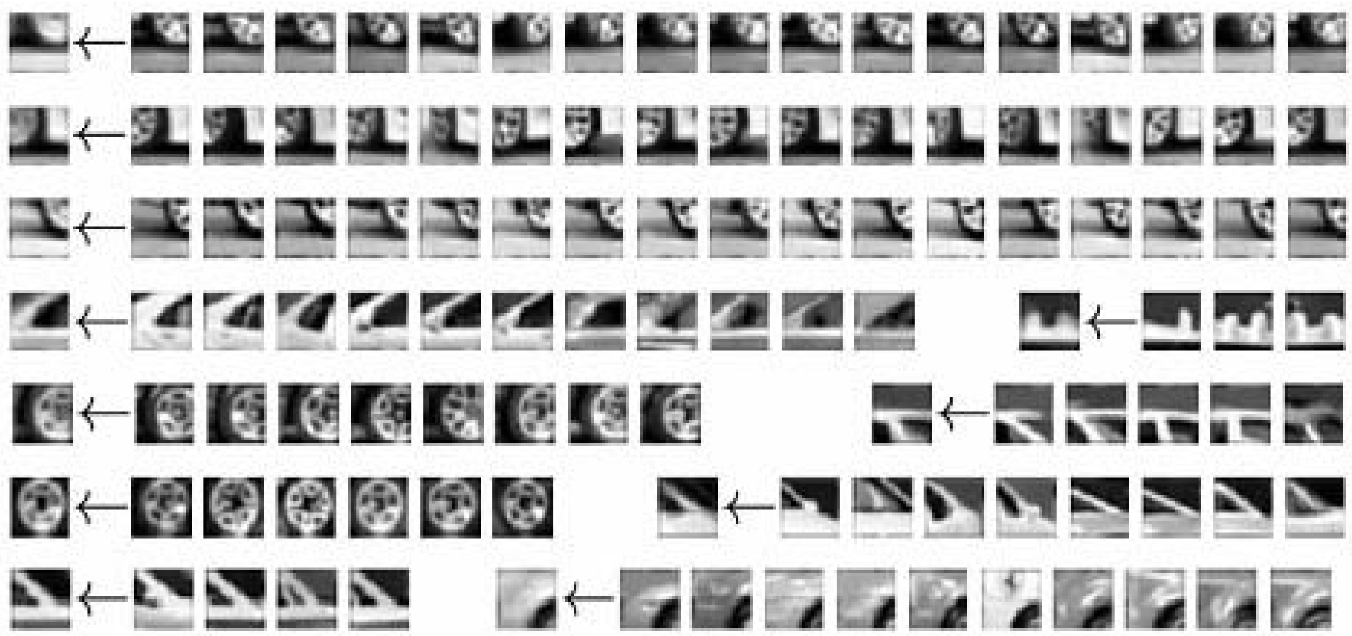
1. Build codebook of patches around extracted interest points using clustering



Slide Credit:

Implicit shape models: Training

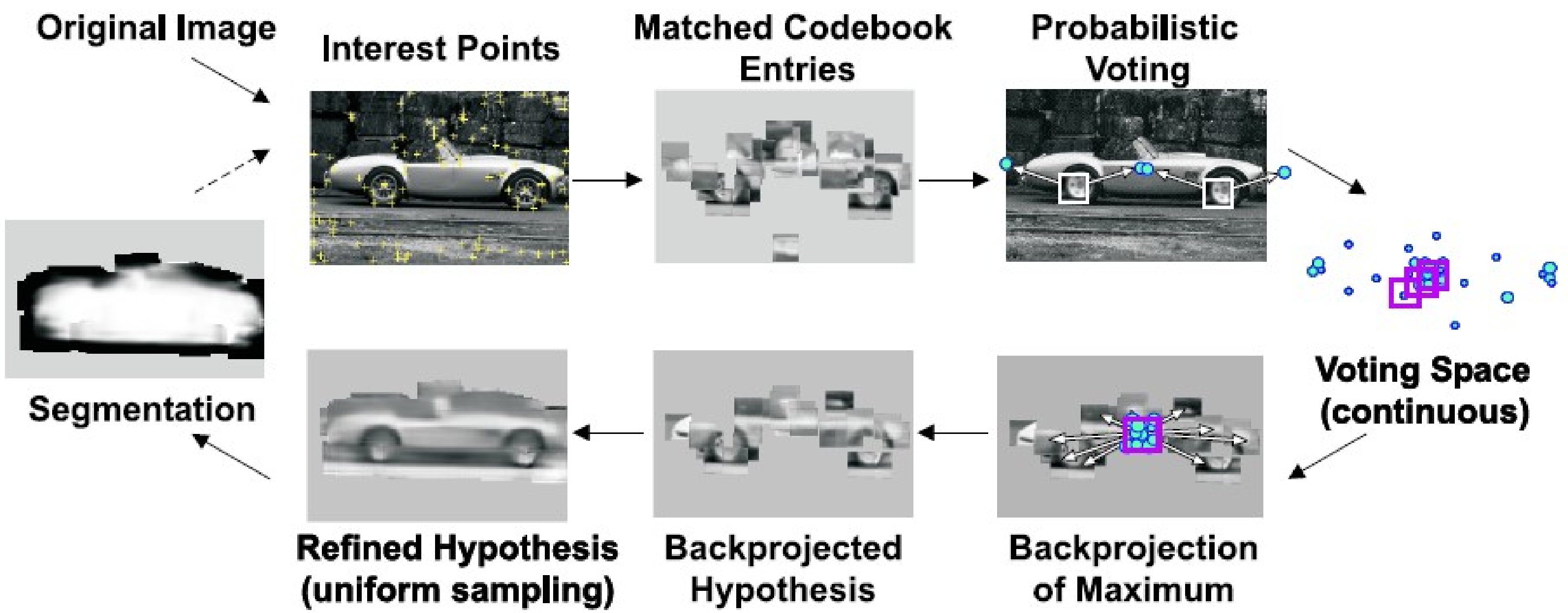
1. Build codebook of patches around extracted interest points using clustering



 Slide Credit:

Implicit shape models: Testing

1. Given test image, extract patches, match to codebook entry
2. Cast votes for possible positions of object center
3. Search for maxima in voting space
4. Extract weighted segmentation mask based on stored masks for the codebook occurrences



Slide Credit:

Example: Results on Cows

Original image

Slide Credit: Source: B. Leibe

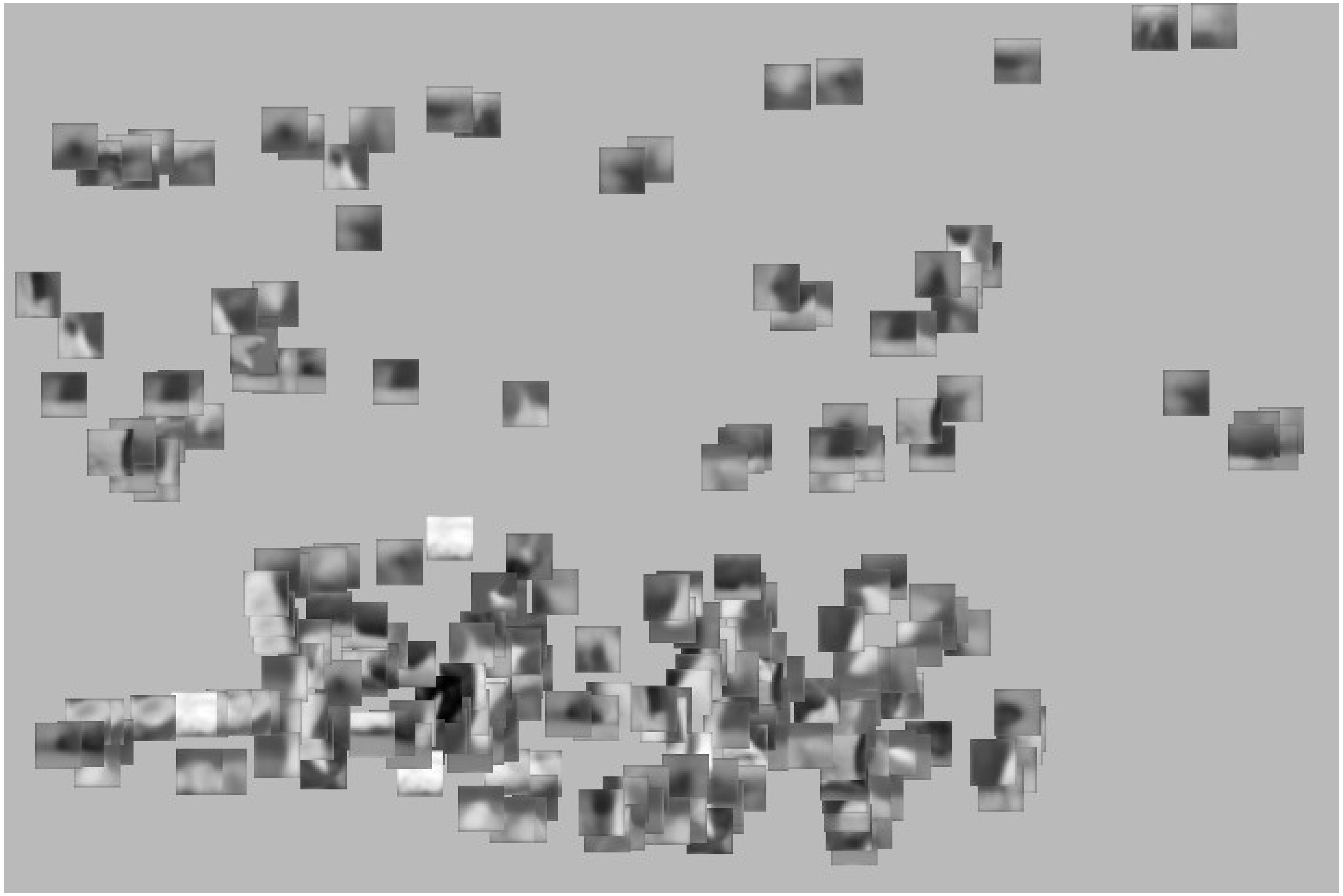
Example: Results on Cows

Interest points

Source: B. Leibe

Slide Credit:

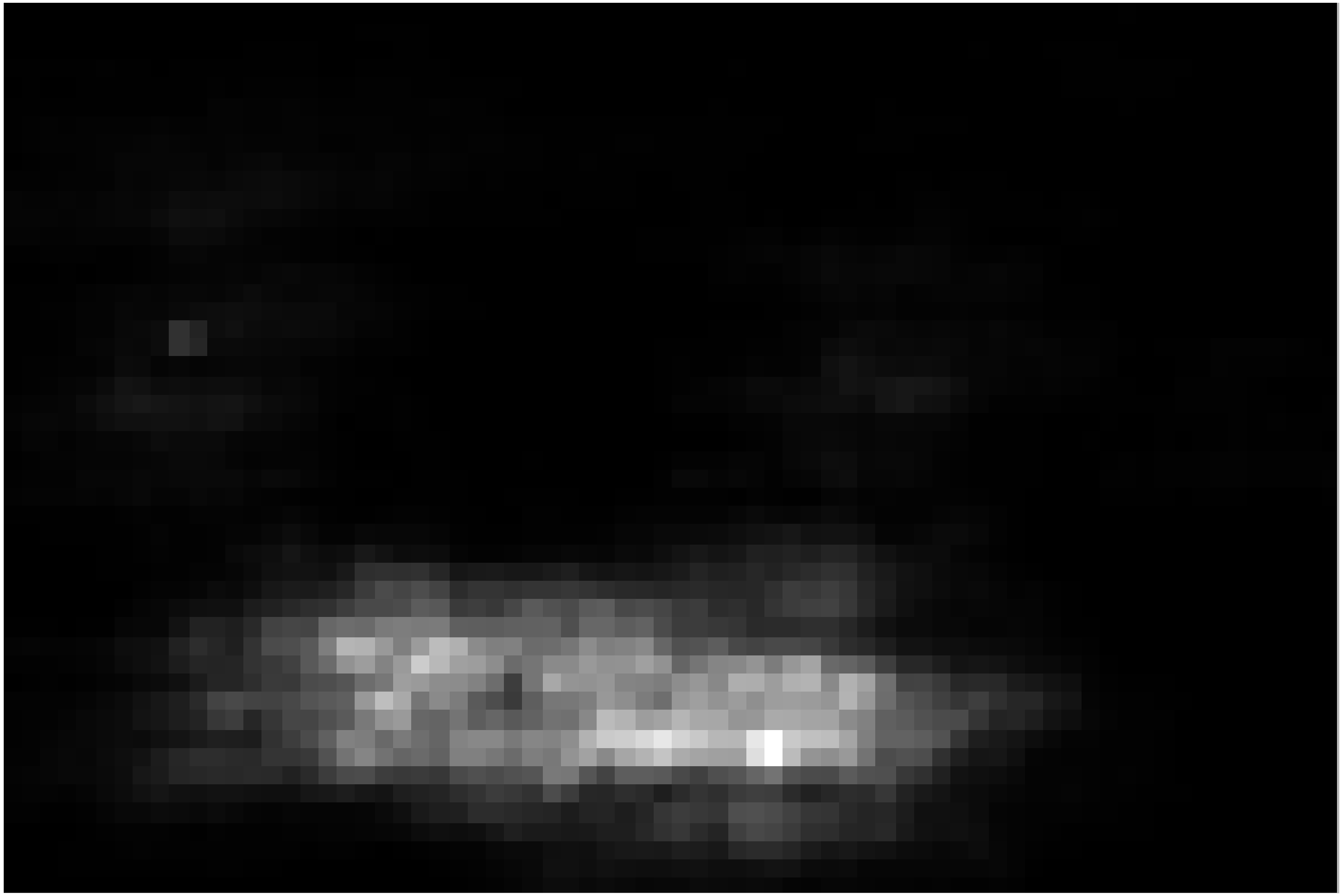
Example: Results on Cows



Matched patches

Slide Credit: Source: B. Leibe

Example: Results on Cows



Probabilistic votes

Slide Credit: Source: B. Leibe

Example: Results on Cows

Hypothesis 1

Slide Credit: Source: B. Leibe

Example: Results on Cows

Hypothesis 1

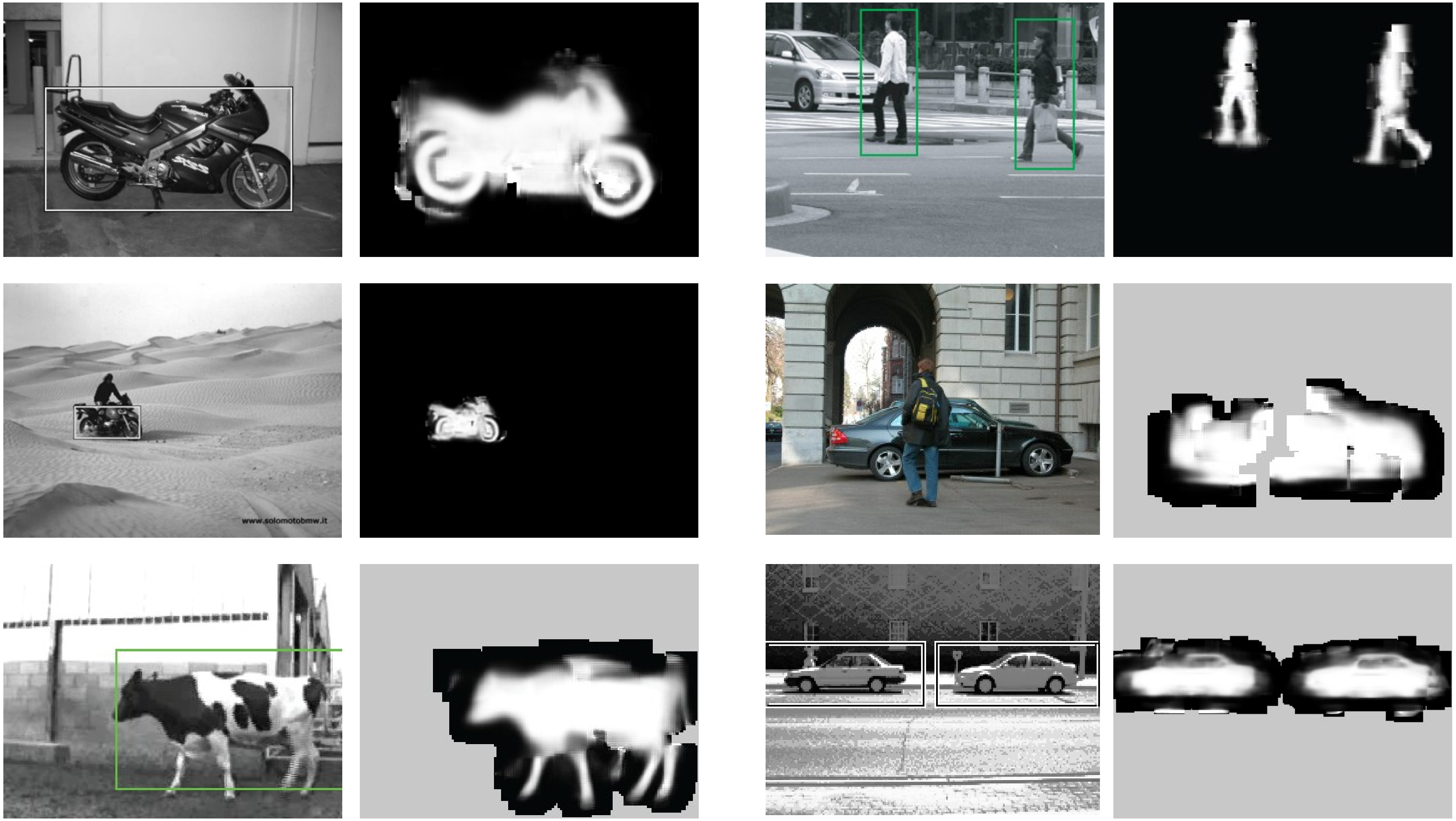
Slide Credit: Source: B. Leibe

Example: Results on Cows

Hypothesis 3

Slide Credit: Source: B. Leibe

Additional examples



B. Leibe, A. Leonardis, and B. Schiele,
Robust Object Detection with Interleaved Categorization and Segmentation, IJCV 77 (1-3), pp. 259-289, 2008

Slide Credit:

Generative part-based models

R. Fergus, P. Perona and A. Zisserman,
Object Class Recognition by Unsupervised Scale-Invariant Learning, CVPR 2003

Slide Credit:

Probabilistic model

$$P(image \mid object) = P(appearance, shape \mid object)$$

Part descriptors

Part locations

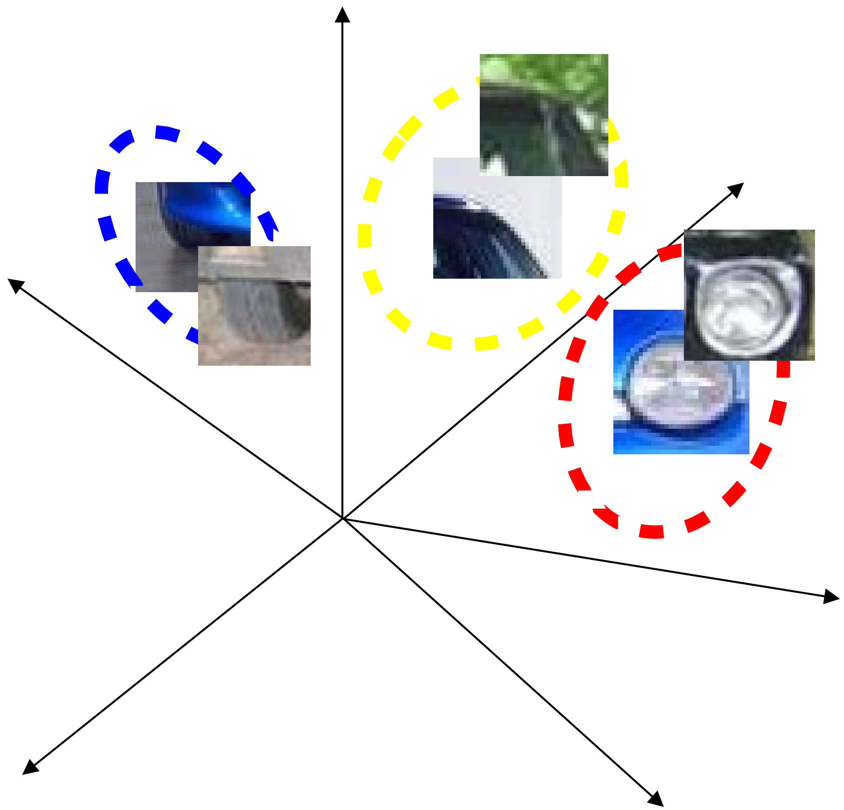
Candidate parts

Slide Credit:

Probabilistic model

$$P(image \mid object) = P(appearance, shape \mid object)$$

$$= \max_h \boxed{P(appearance \mid h, object)}\, p(shape \mid h, object)\, p(h \mid object)$$



Distribution over patch descriptors

High-dimensional appearance space

Slide Credit:

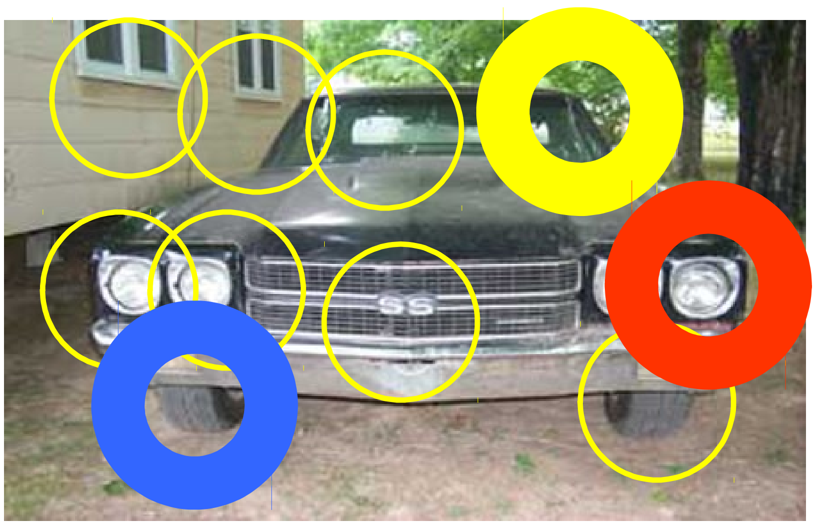
Probabilistic model

$$P(image \mid object) = P(appearance, shape \mid object)$$

$$= \max_h P(appearance \mid h, object)\, p(shape \mid h, object)\, p(h \mid object)$$

h: assignment of features to parts

Part 1



Part 3

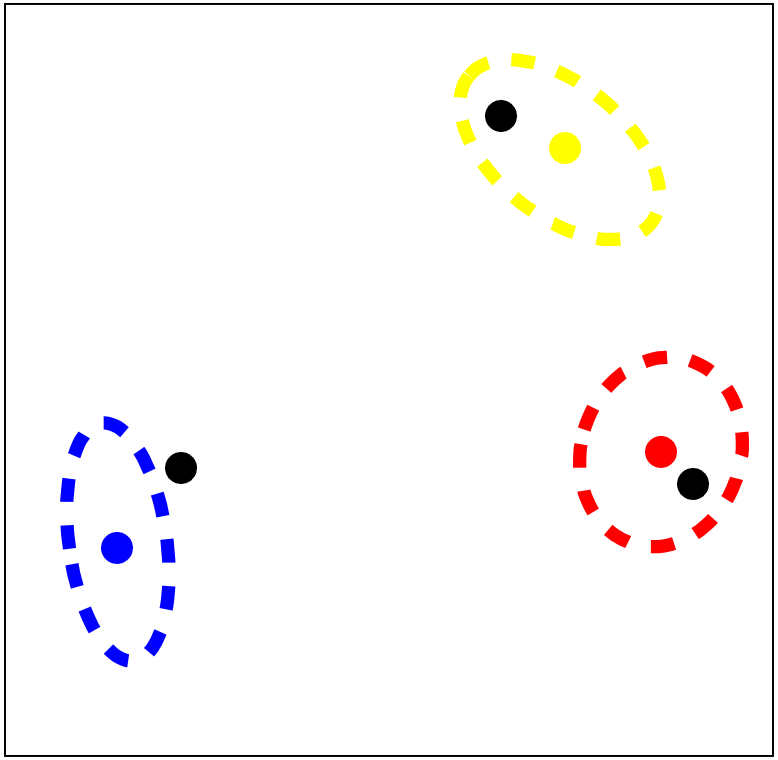
Part 2

Slide Credit:

Probabilistic model

$$P(image \mid object) = P(appearance, shape \mid object)$$

$$= \max_h P(appearance \mid h, object) \boxed{p(shape \mid h, object)} p(h \mid object)$$



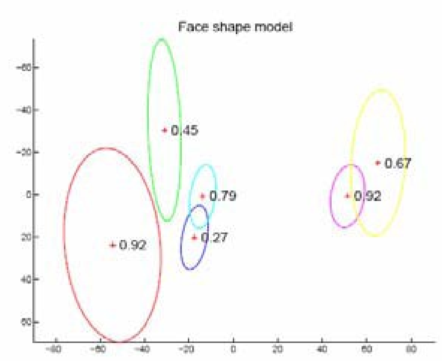
2D image space

Distribution
over joint
part positions

Slide Credit:

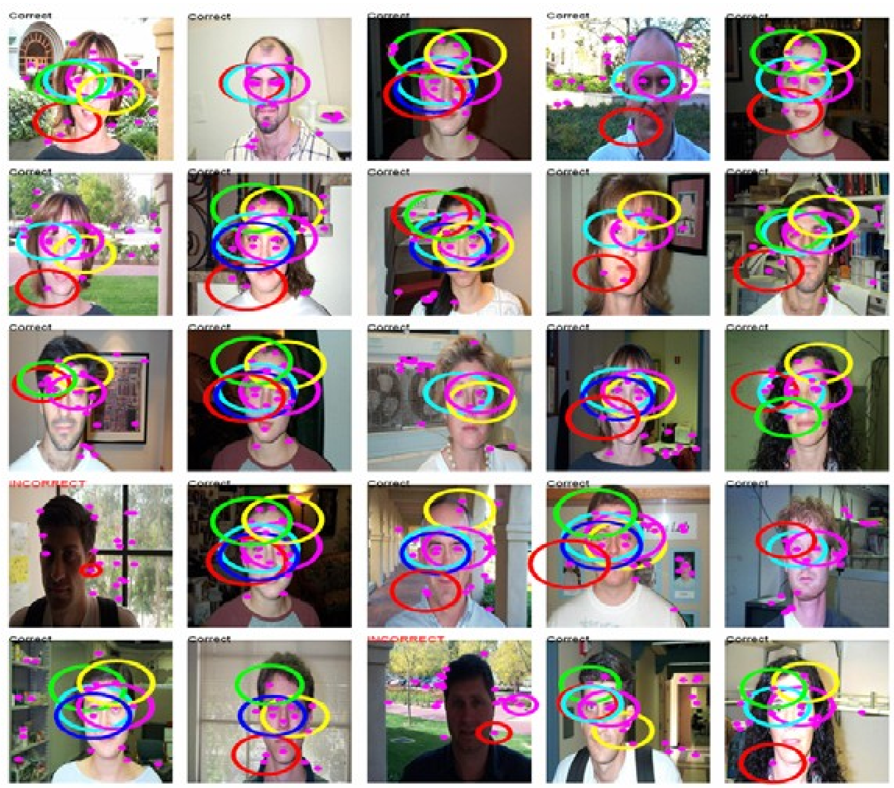
Results: Faces

Face shape model



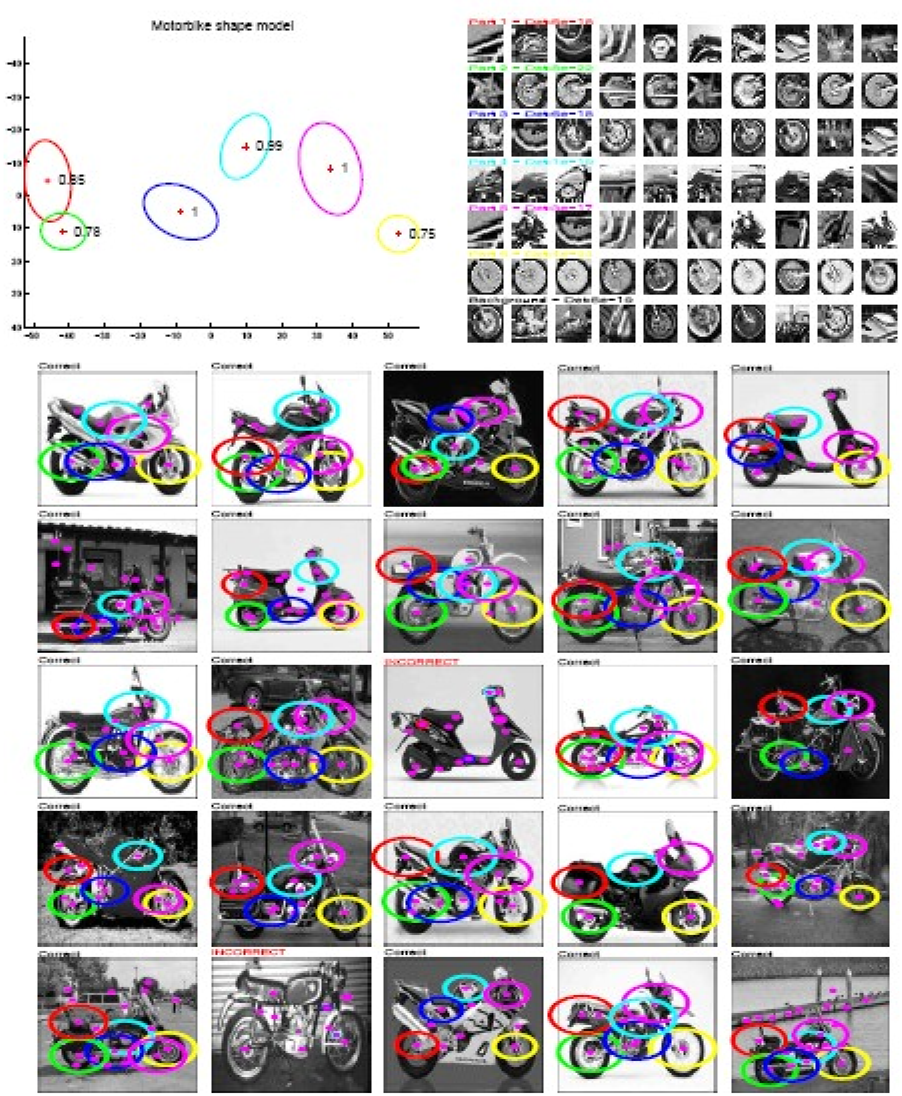
Patch appearance model

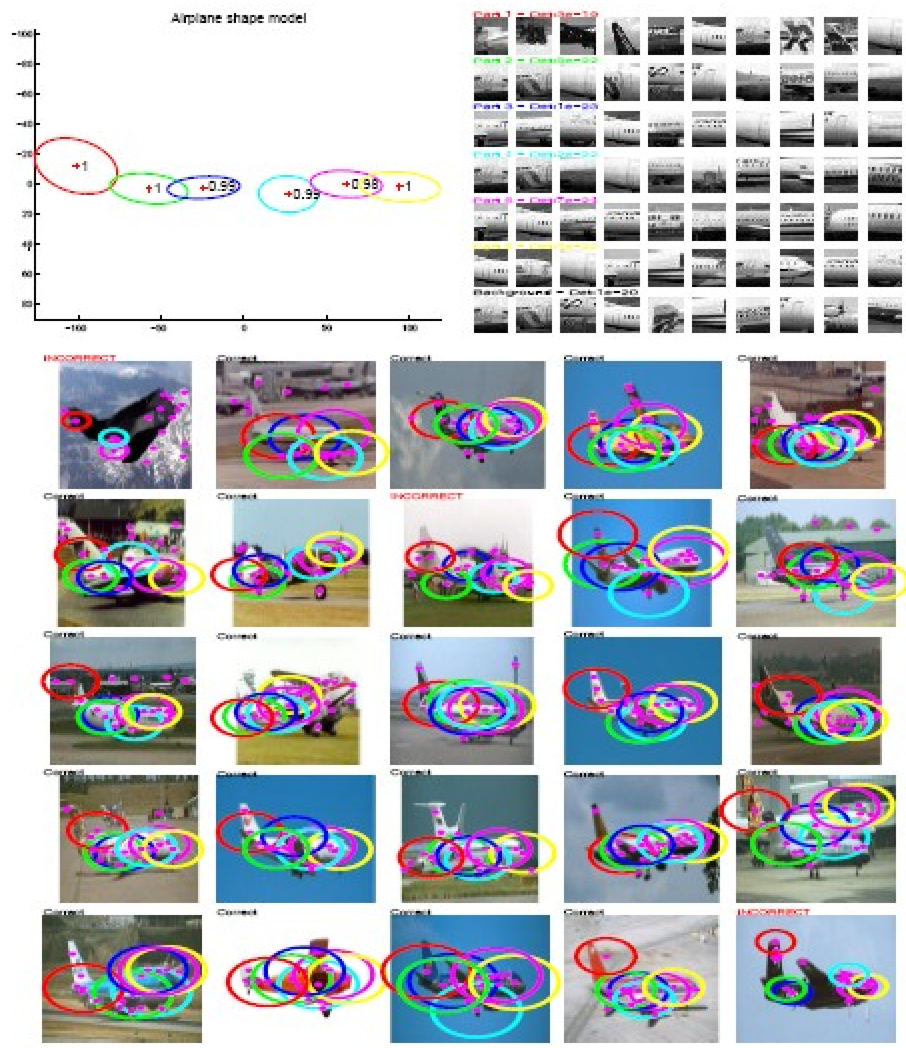
Recognition results



Slide Credit:

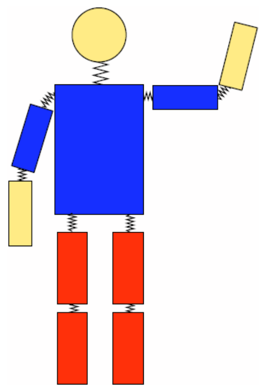
Results: Motorbikes and airplanes



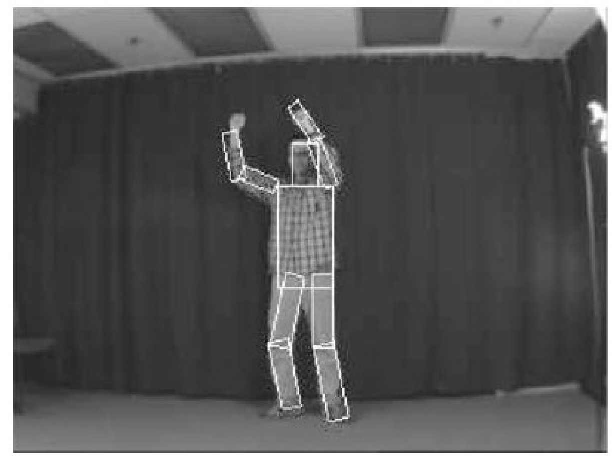


Slide Credit:

Pictorial structures



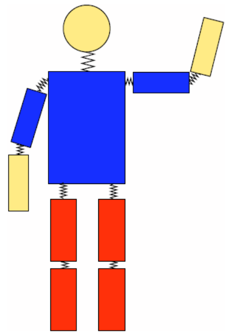
- Set of parts (oriented rectangles) connected by edges
- Recognition problem: find the most probable part layout $l_1, \ldots, l_n$ in the image

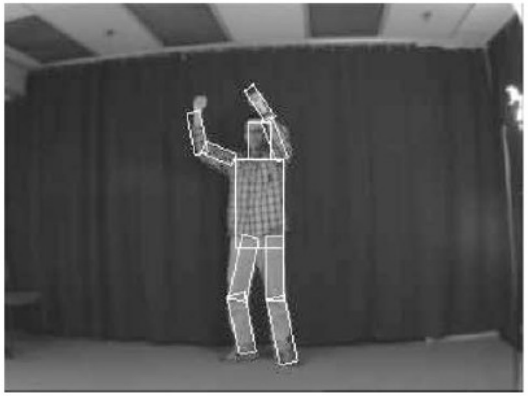


P. Felzenszwalb and D. Huttenlocher, **Pictorial Structures for Object Recognition**, IJCV 61(1), 2005

Slide Credit: Felzenszwalb

Pictorial structures





- MAP formulation: maximize posterior

$$P(l_1,\dots,l_n \mid \mathrm{Im}) \propto P(\mathrm{Im} \mid l_1,\dots,l_n)P(l_1,\dots,l_n) = \underbrace{\prod_i P(\mathrm{Im}(l_i))}_{\text{Appearance}} \underbrace{\prod_{i,j\in E} P(l_i \mid l_j)}_{\text{Geometry}}$$

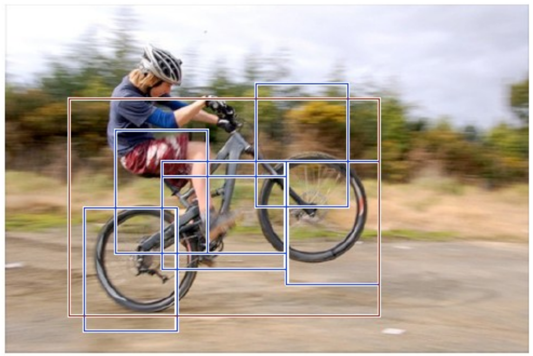
- Energy-based formulation: minimize minus the log of probability:

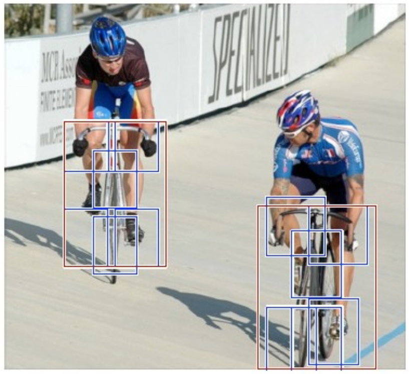
$$E(l_1,\dots,l_n) = \underbrace{\sum_i m_i(l_i)}_{\text{Matching cost}} + \underbrace{\sum_{i,j} d_{ij}(l_i,l_j)}_{\text{Deformation cost}}$$

Slide Credit: Felzenszwalb

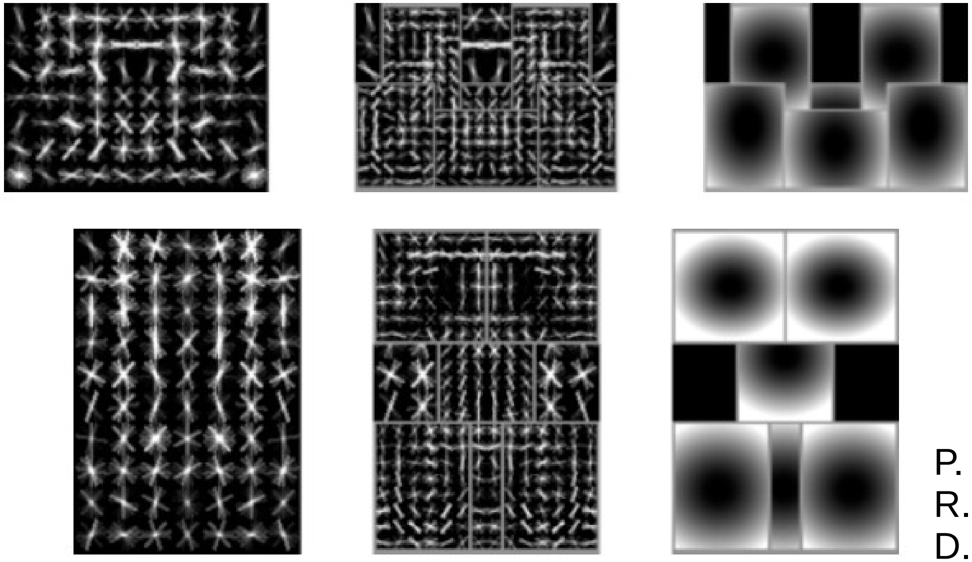
Deformable Parts Model

Detections





Template Visualization



root filters coarse resolution

part filters finer resolution

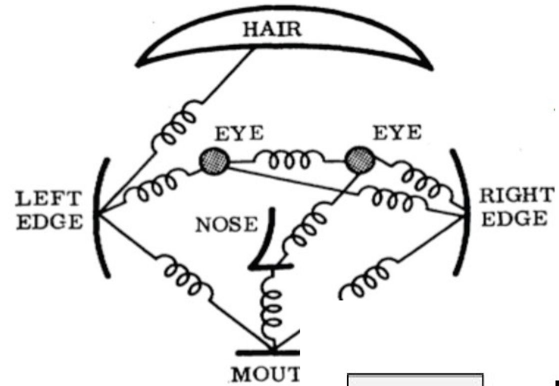
deformation models

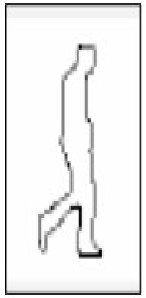
P. Felzenszwalb,
R. Girshick,
D. McAllester,
D. Ramanan
"Object Detection with Discriminatively Trained Part Based Models"

Slide Credit: Felzenszwalb

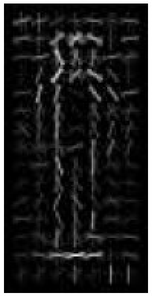
So many options... How to choose?

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

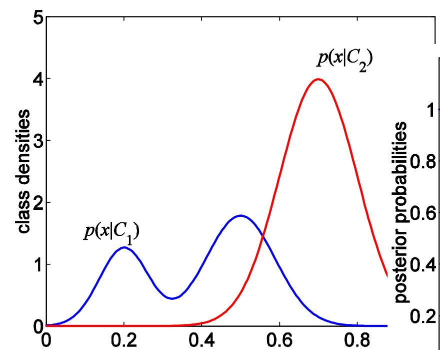


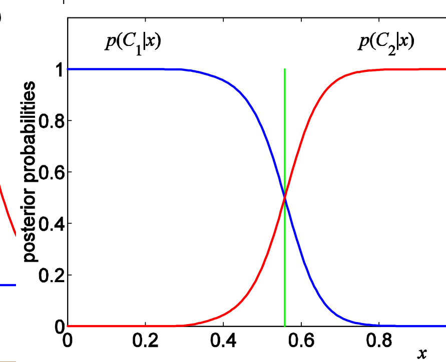


Exemplar



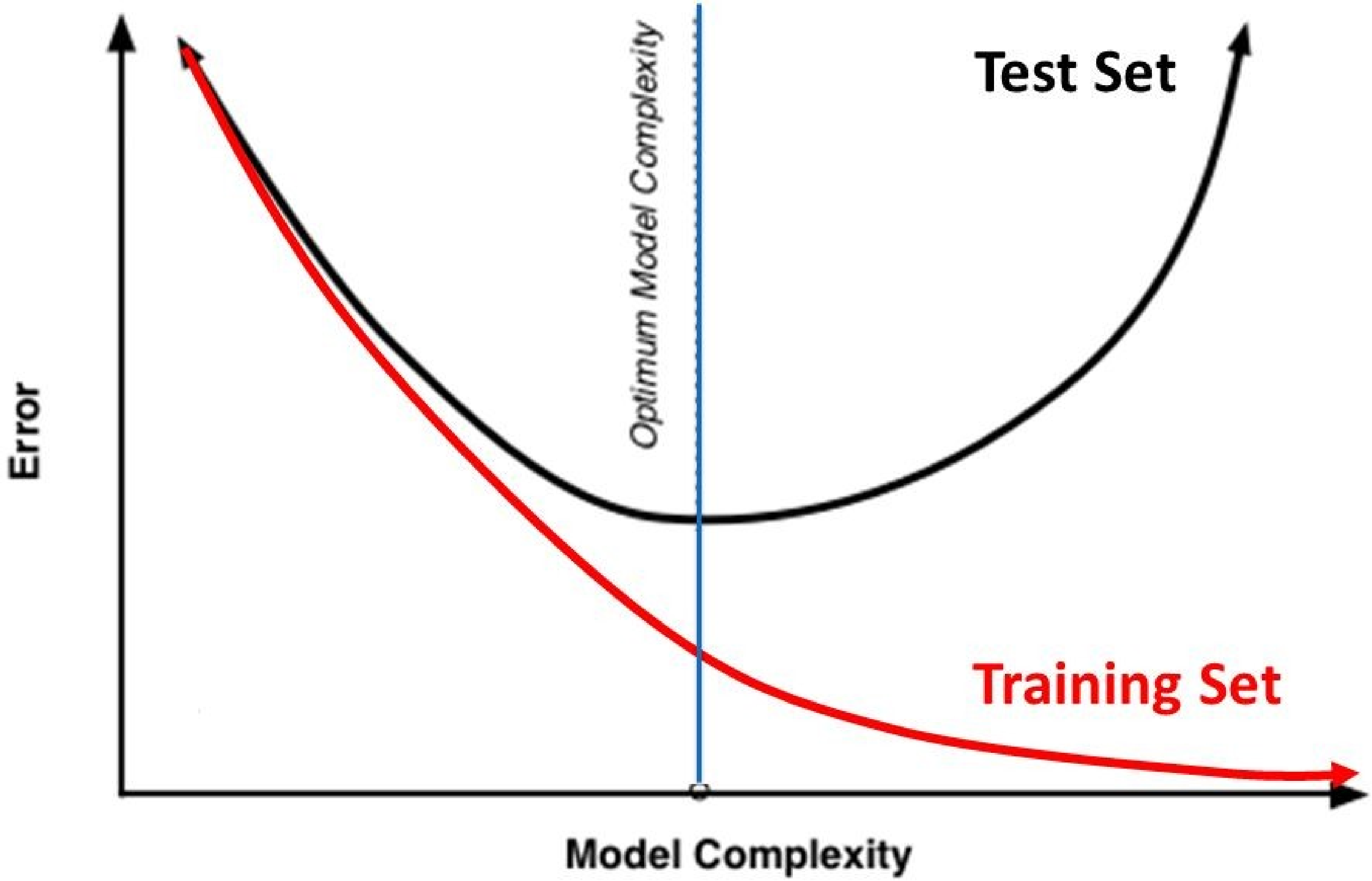
Summary





Slide Credit:

Train vs. Test Accuracy



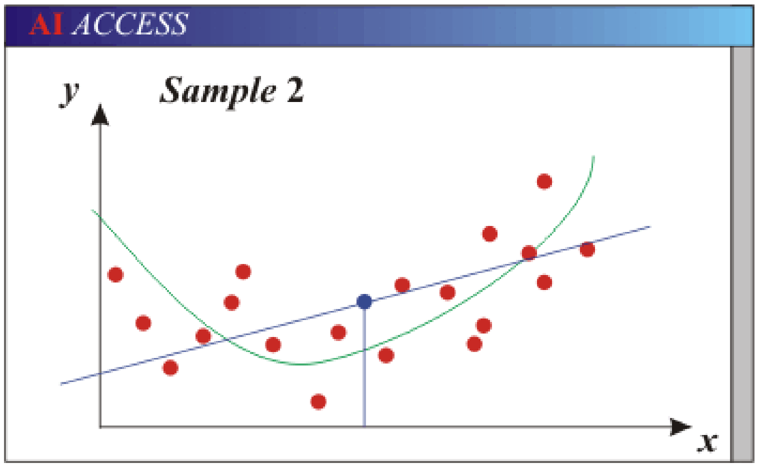
Slide Credit:

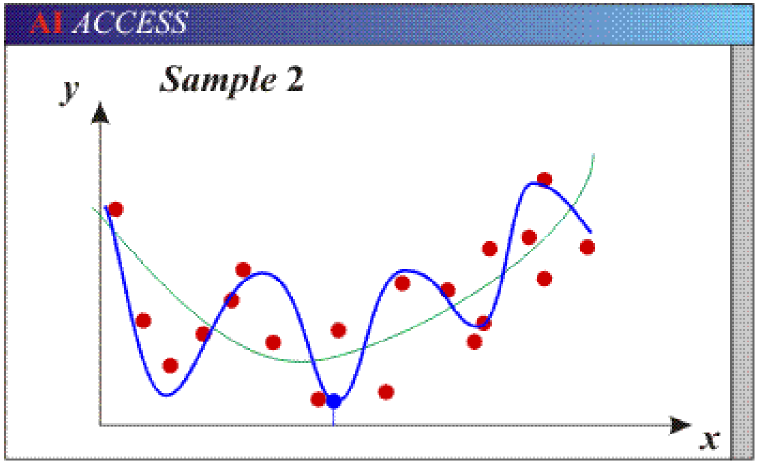
Generalization

- 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

Slide Credit: Slide credit: L. Lazebnik

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).

Slide credit: D. Hoiem

Bias-Variance Trade-off

$$E(MSE) = noise^2 + bias^2 + variance$$

Unavoidable error

Error due to incorrect assumptions

Error due to variance of training samples

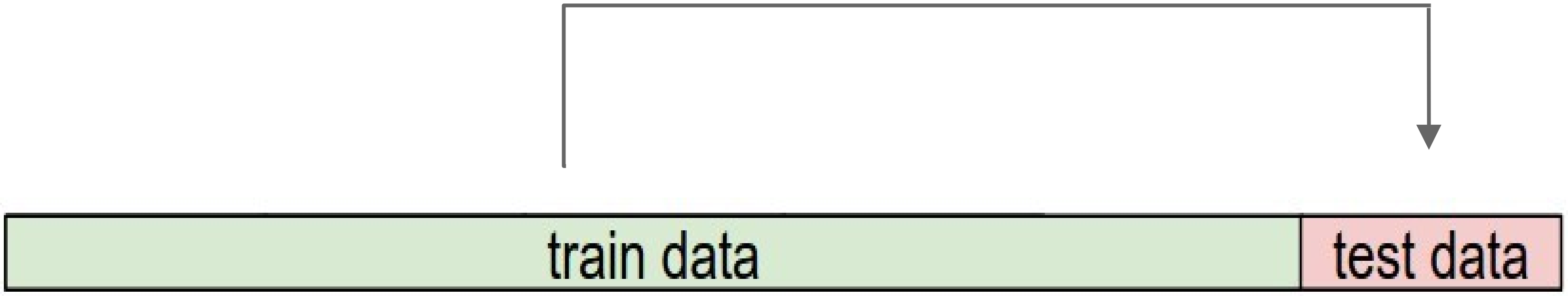
See the following for explanations of bias-variance (also Bishop's "Neural Networks" book):

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

Slide credit: D. Hoiem

Cross Validation

Try out what hyperparameters work best on test set.



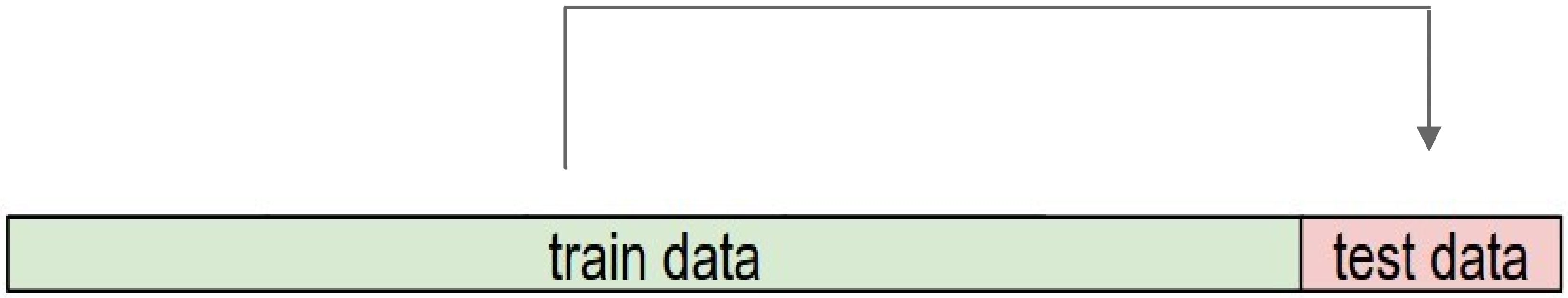
 Slide Credit:

Cross Validation

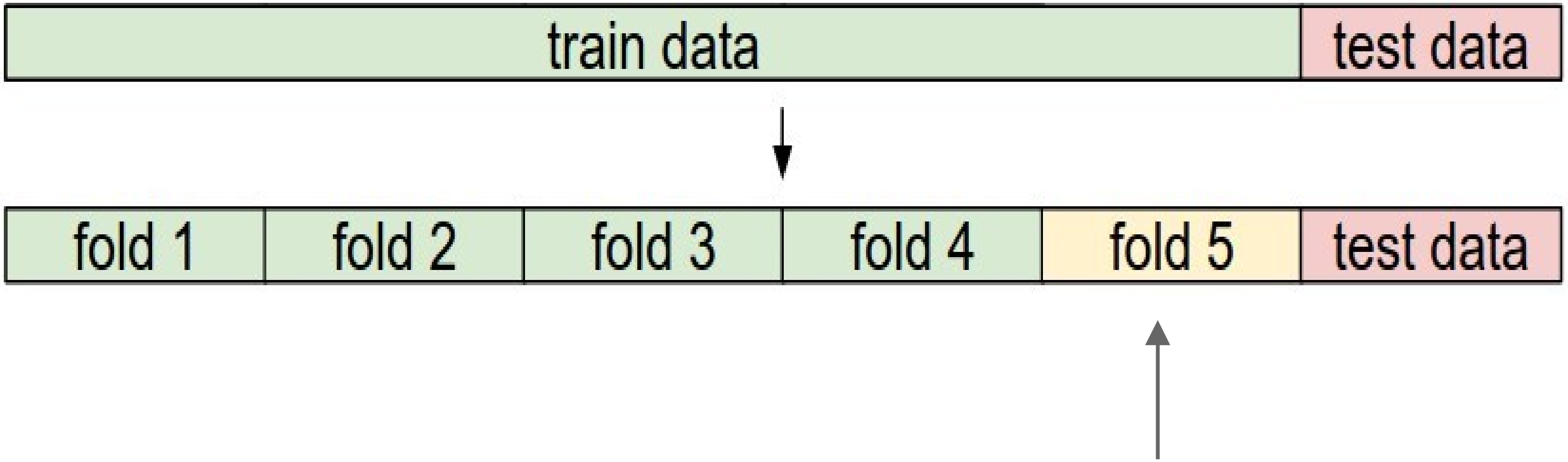
Trying out what hyperparameters work best on test set:

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

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



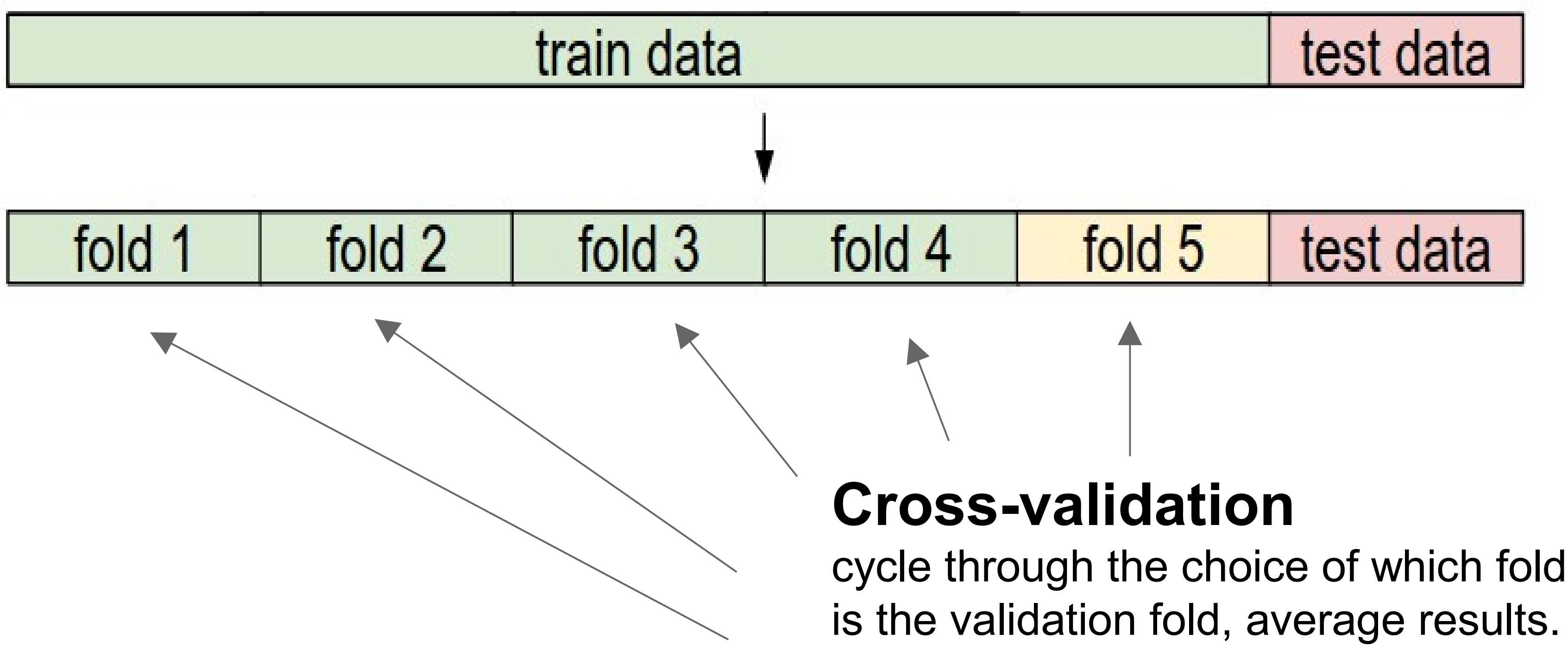
Cross Validation



Validation data
use to tune hyperparameters

Slide Credit:

Cross Validation



Slide Credit:

So…

- 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

Slide Credit: Slide credit: D. Hoiem

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)

 Slide Credit: Slide credit: D. Hoiem

How to reduce variance?

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

 Slide Credit: Slide credit: D. Hoiem