Machine Learning Basics (II)

簡韶逸 Shao-Yi Chien
Department of Electrical Engineering
National Taiwan University

Spring 2021

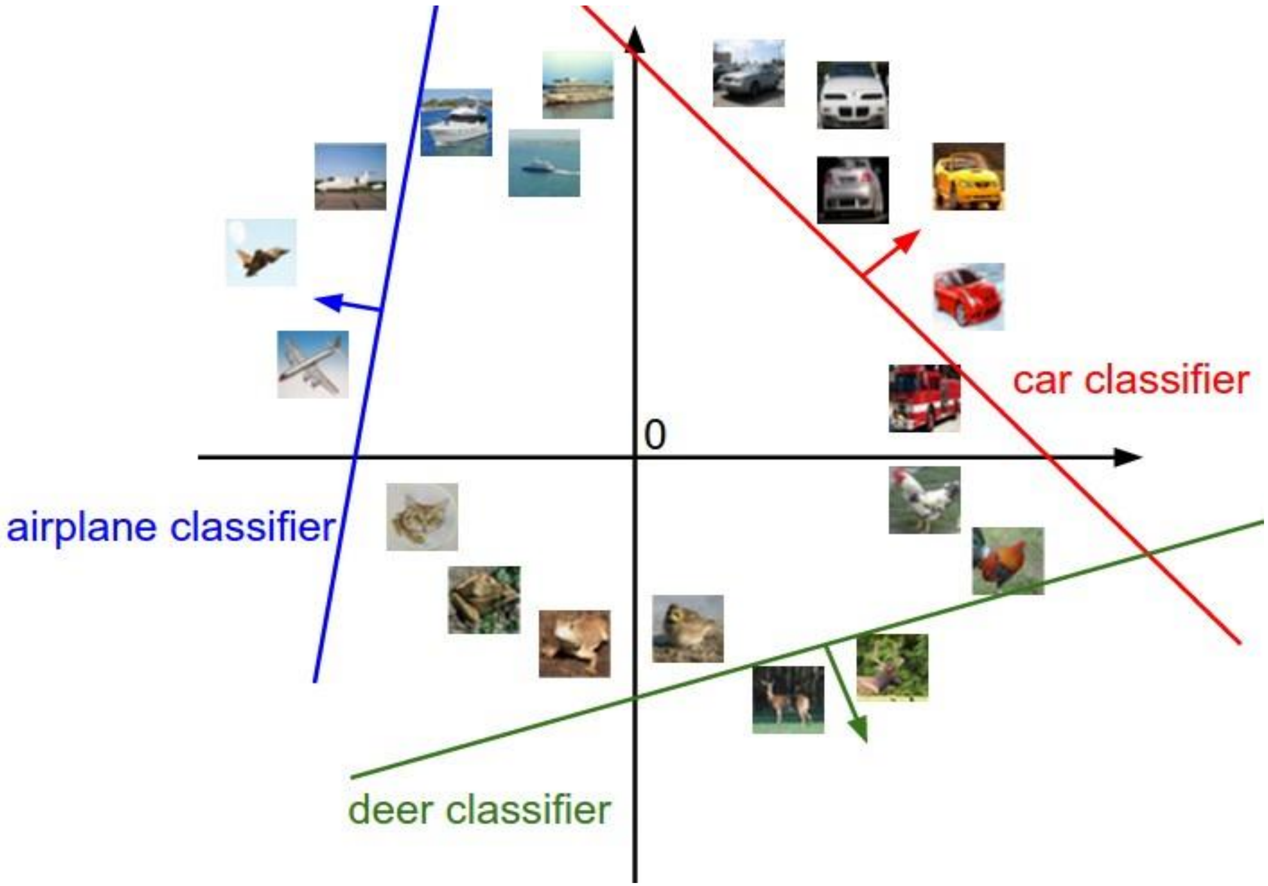
References and Slide Credits

- Slides from *Deep Learning for Computer Vision*, Prof. Yu-Chiang Frank Wang, EE, National Taiwan University
- Slides from *CE 5554 / ECE 4554: Computer Vision*, Prof. J.-B. Huang, Virginia Tech
- Slides from *CSE 576 Computer Vision*, Prof. Steve Seitz and Prof. Rick Szeliski, U. Washington
- Slides from EECS 498-007/598-005 Deep Learning for Computer Vision, Prof. Justin Johnson
- Slides from CS291A Introduction to Pattern Recognition, Artificial Neural Networks, and Machine Learning, Prof. Professor Yuan-Fang Wang, UCSB
- Slides from Introduction to Machine Learning Course, Prof. Sargur Srihari, University at Buffalo
- Duda et al., *Pattern Classification*
- Bishop*, Pattern Recognition and Machine Learning*
- Prof. Chih-Jen Lin (林智仁), CSIE, NTU, SVM library (LIBSVM), http://www.csie.ntu.edu.tw/~cjlin
- Reference papers

Outline

- Overview of recognition/classification pipeline
- Overview of machine learning
- From probability to Bayes decision rule
- Nonparametric techniques: Parzen window and nearest neighbor
- Unsupervised learning and supervised learning
- Unsupervised learning
  - Clustering: k-means
  - Dimension reduction: PCA and LDA
- Training, testing, & validation
- Supervised learning
  - Linear classification: support vector machine (SVM)
  - Combining models: decision tree, boosting
- Examples

Linear Classifier



CIFAR10

50,000 training images
each image is **32x32x3**

10,000 test images.

Parametric Approach

Image

Array of **32x32x3** numbers
(3072 numbers total)

f(**x**,**W**)

10 numbers giving
class scores

W
parameters
or weights

Parametric Approach: Linear Classifier

(3072,)

$$\boxed{f(x,W)} = \boxed{W}\boxed{x}$$

(10,) (10, 3072)

Image

Array of **32x32x3** numbers
(3072 numbers total)

$f(x, W)$

10 numbers giving class scores

W

parameters
or weights

Parametric Approach: Linear Classifier

(3072,)

$$\underbrace{\boxed{f(x,W)}}_{(10,)} = \underbrace{\boxed{W}}_{(10,\ 3072)}\overbrace{\boxed{x}}^{(3072,)} + \boxed{b} \quad (10,)$$

Image

Array of **32x32x3** numbers
(3072 numbers total)

f(**x**,**W**)

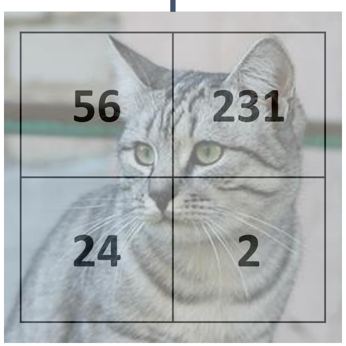
10 numbers giving
class scores

W
parameters
or weights

Example for 2x2 image, 3 classes (cat/dog/ship)

Stretch pixels into column

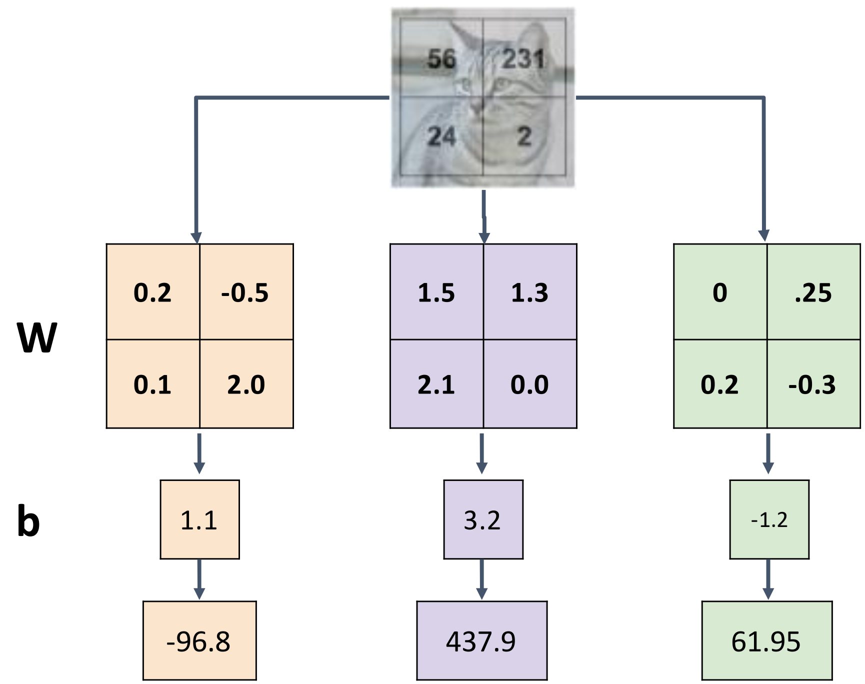
$f(x,W) = Wx + b$



Input image
(2, 2)

$$\underbrace{\begin{bmatrix} 0.2 & -0.5 & 0.1 & 2.0 \\ 1.5 & 1.3 & 2.1 & 0.0 \\ 0 & 0.25 & 0.2 & -0.3 \end{bmatrix}}_{W\ (3,\,4)} \underbrace{\begin{bmatrix} 56 \\ 231 \\ 24 \\ 2 \end{bmatrix}}_{(4,)} + \underbrace{\begin{bmatrix} 1.1 \\ 3.2 \\ -1.2 \end{bmatrix}}_{b\ (3,)} = \underbrace{\begin{bmatrix} -96.8 \\ 437.9 \\ 61.95 \end{bmatrix}}_{(3,)}$$

Interpreting an Linear Classifier: Visual Viewpoint

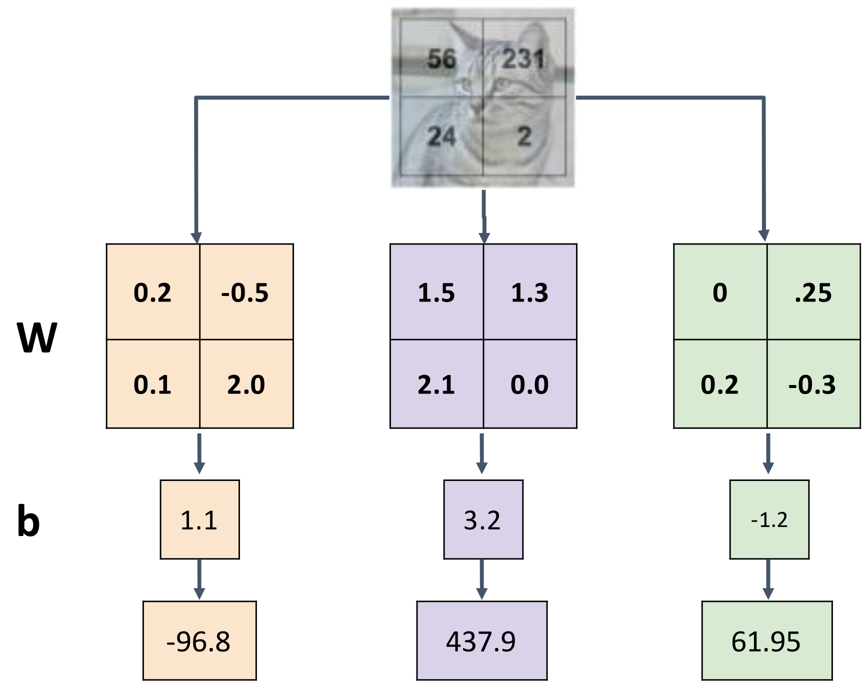


Interpreting an Linear Classifier: <u>Visual Viewpoint</u>

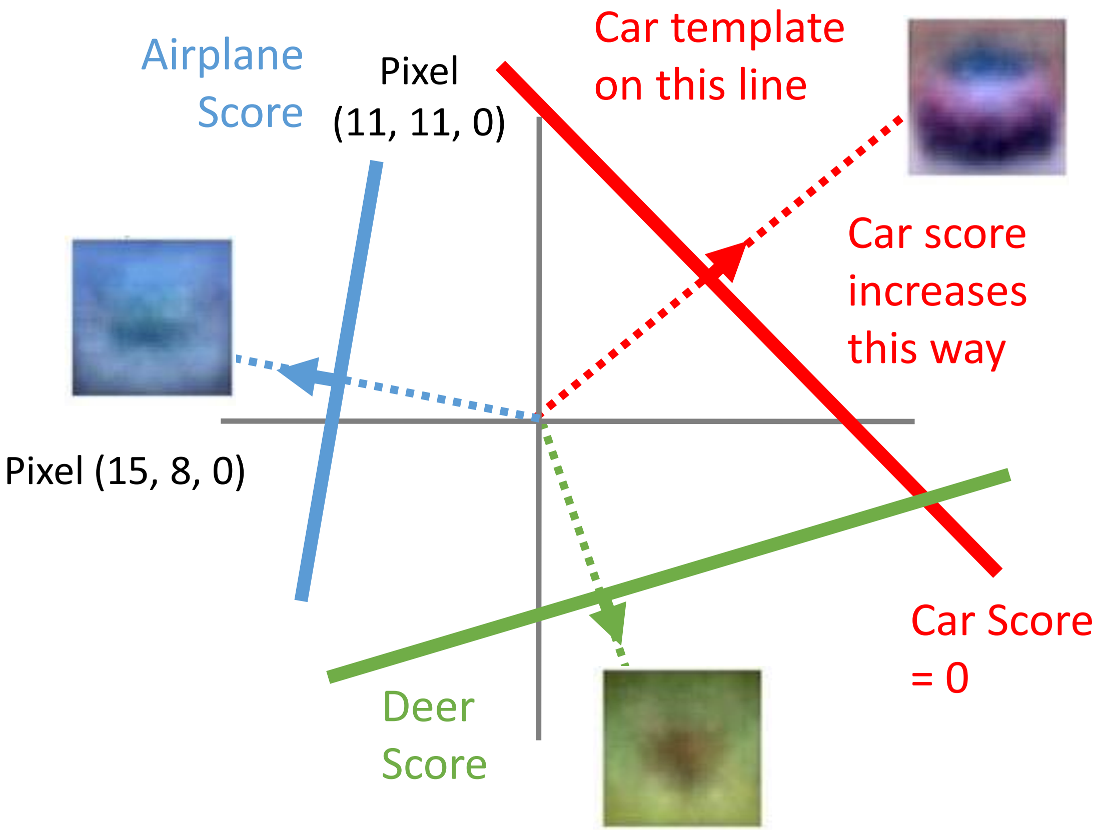
Linear classifier has one "template" per category

A single template cannot capture multiple modes of the data

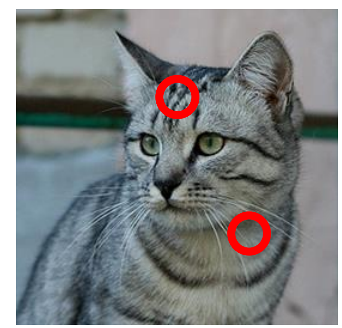
e.g. horse template has 2 heads!



Interpreting a Linear Classifier: Geometric Viewpoint

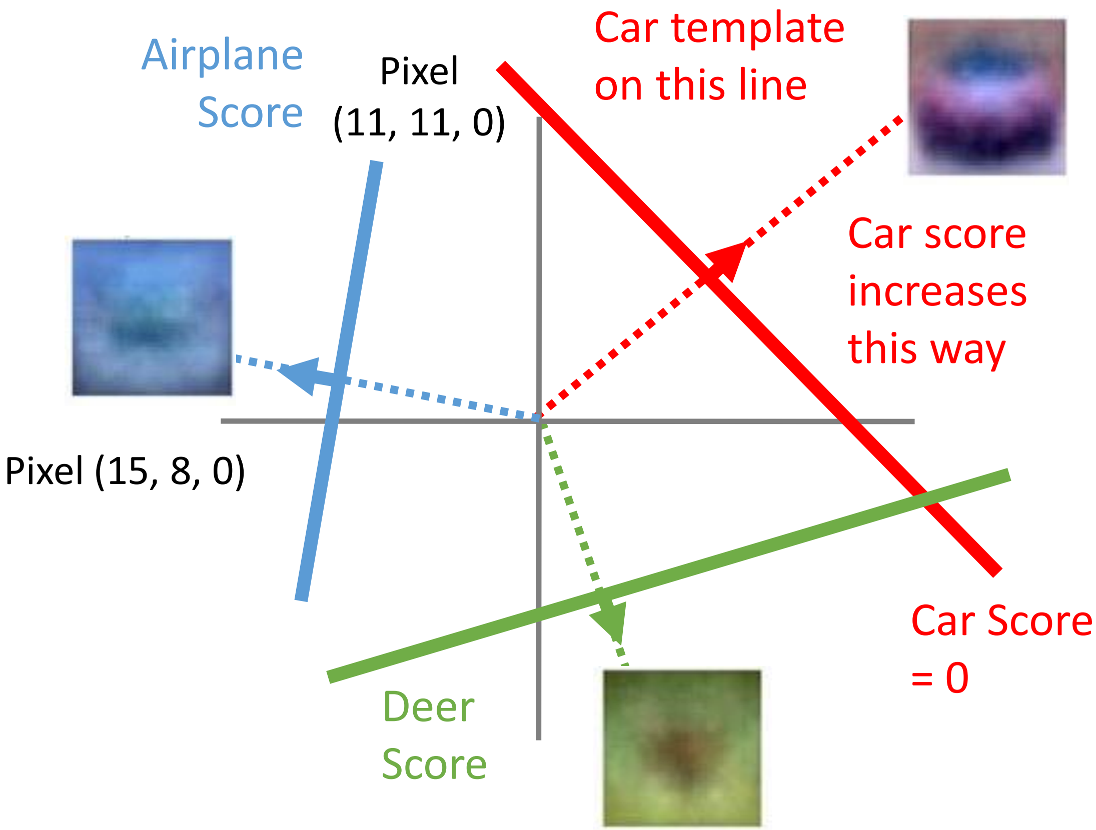


$$f(x,W) = Wx + b$$

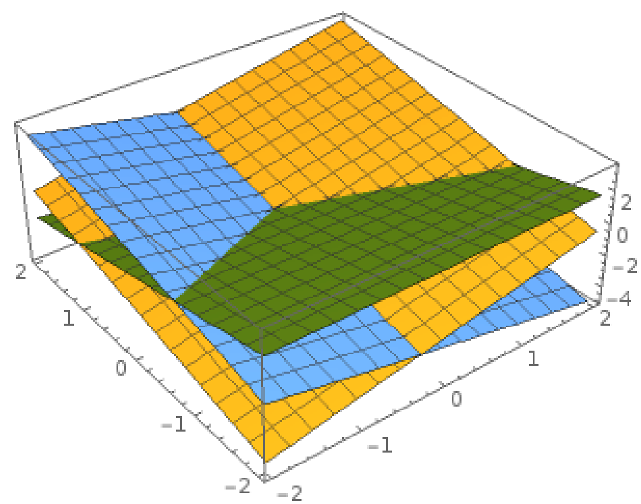


Array of **32x32x3** numbers
(3072 numbers total)

Interpreting a Linear Classifier: <u>Geometric Viewpoint</u>



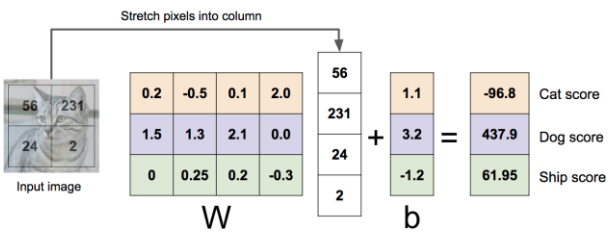
Hyperplanes carving up a high-dimensional space



Linear Classifier: Three Viewpoints

<u>Algebraic Viewpoint</u>

$f(x,W) = Wx$

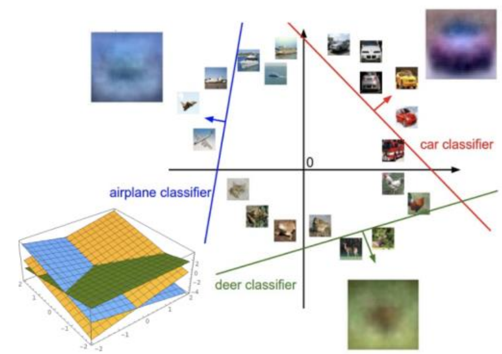


<u>Visual Viewpoint</u>

One template
per class

<u>Geometric Viewpoint</u>

Hyperplanes
cutting up space



Loss Function

A **loss function** tells how good our current classifier is

Low loss = good classifier
High loss = bad classifier

(Also called: **objective function**; **cost function**)

Negative loss function sometimes called **reward function**, **profit function**, **utility function**, **fitness function**, etc

Given a dataset of examples

$$\{(x_i, y_i)\}_{i=1}^N$$

Where x_i is image and
y_i is (integer) label

Loss for a single example is

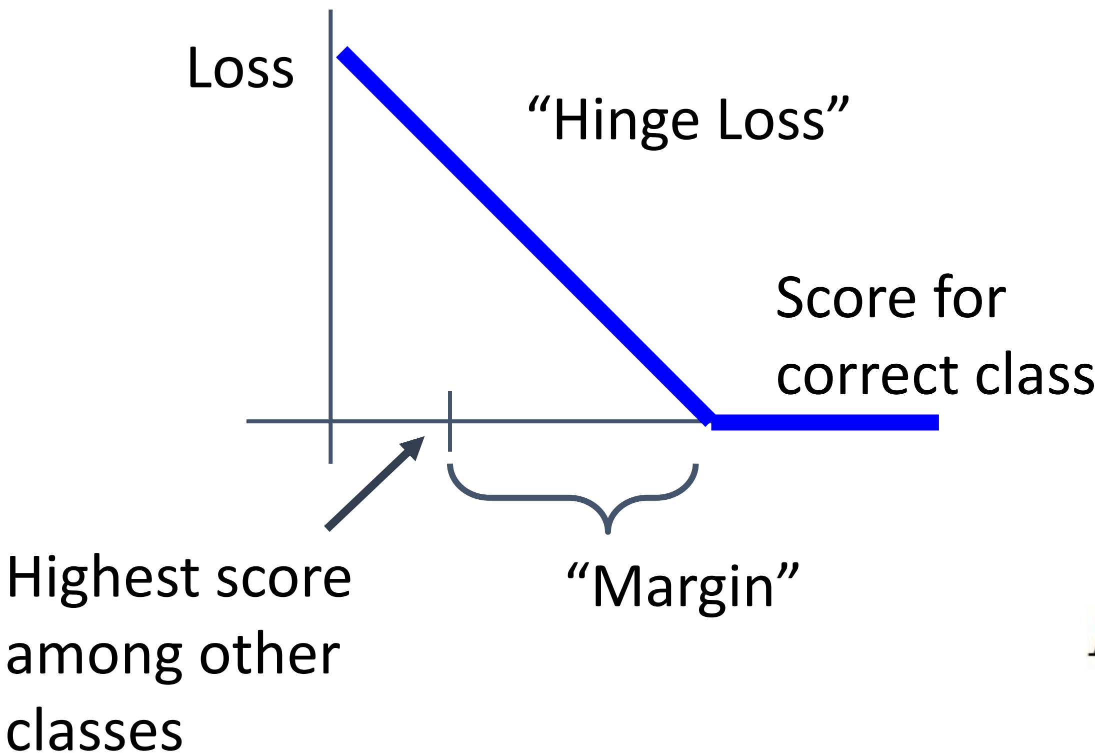
$$L_i(f(x_i, W), y_i)$$

Loss for the dataset is average of per-example losses:

$$L = \frac{1}{N}\sum_i L_i(f(x_i, W), y_i)$$

Multiclass SVM Loss

”The score of the correct class should be higher than all the other scores”



Given an example (x_i, y_i)
(x_i is image, y_i is label)

Let $s = f(x_i, W)$ be scores

Then the SVM loss has the form:

$$L_i = \sum_{j \neq y_i} \max(0, s_j - s_{y_i} + 1)$$

Multiclass SVM Loss

cat	**3.2**	1.3	2.2
car	5.1	**4.9**	2.5
frog	-1.7	2.0	**-3.1**
Loss	2.9		

Given an example (x_i, y_i)
(x_i is image, y_i is label)

Let $s = f(x_i, W)$ be scores

Then the SVM loss has the form:

$$L_i = \sum_{j \neq y_i} \max(0, s_j - s_{y_i} + 1)$$

= max(0, 5.1 - 3.2 + 1)
+ max(0, -1.7 - 3.2 + 1)
= max(0, 2.9) + max(0, -3.9)
= 2.9 + 0
= 2.9

Multiclass SVM Loss

cat	**3.2**	1.3	2.2
car	5.1	**4.9**	2.5
frog	-1.7	2.0	**-3.1**
Loss	2.9	0	

Given an example (x_i, y_i)
(x_i is image, y_i is label)

Let $s = f(x_i, W)$ be scores

Then the SVM loss has the form:

$$L_i = \sum_{j \neq y_i} \max(0, s_j - s_{y_i} + 1)$$

= max(0, 1.3 - 4.9 + 1)
+max(0, 2.0 - 4.9 + 1)
= max(0, -2.6) + max(0, -1.9)
= 0 + 0
= 0

Multiclass SVM Loss

cat	**3.2**	1.3	2.2
car	5.1	**4.9**	2.5
frog	-1.7	2.0	**-3.1**
Loss	2.9	0	12.9

Given an example (x_i, y_i)
(x_i is image, y_i is label)

Let $s = f(x_i, W)$ be scores

Then the SVM loss has the form:

$$L_i = \sum_{j \neq y_i} \max(0, s_j - s_{y_i} + 1)$$

= max(0, 2.2 - (-3.1) + 1)
+max(0, 2.5 - (-3.1) + 1)
= max(0, 6.3) + max(0, 6.6)
= 6.3 + 6.6
= 12.9

Multiclass SVM Loss

cat	**3.2**	1.3	2.2
car	5.1	**4.9**	2.5
frog	-1.7	2.0	**-3.1**
Loss	2.9	0	12.9

Given an example (x_i, y_i)
(x_i is image, y_i is label)

Let $s = f(x_i, W)$ be scores

Then the SVM loss has the form:

$$L_i = \sum_{j \neq y_i} \max(0, s_j - s_{y_i} + 1)$$

Loss over the dataset is:

L = (2.9 + 0.0 + 12.9) / 3
= 5.27

Cross-Entropy Loss
(Multinomial Logistic Regression)

Want to interpret raw classifier scores as **probabilities**

$$s = f(x_i; W)$$

$$P(Y = k | X = x_i) = \frac{e^{s_k}}{\sum_j e^{s_j}}$$ Softmax function

$$L_i = -\log P(Y = y_i | X = x_i)$$

Probabilities must be >= 0

Probabilities must sum to 1

	Unnormalized log-probabilities / logits		unnormalized probabilities		probabilities		Correct probs
cat	**3.2**	exp →	**24.5**	normalize →	**0.13**	→ Compare ←	**1.00**
car	5.1		164.0		0.87	*Cross Entropy*	0.00
frog	-1.7		0.18		0.00	$H(P, Q) = -\sum_i p_i \log(q_i)$	0.00

Regularization: Beyond Training Error

$$L(W) = \underbrace{\frac{1}{N}\sum_{i=1}^{N} L_i(f(x_i, W), y_i)}_{} + \underbrace{\lambda R(W)}_{}$$

λ = regularization strength (hyperparameter)

Data loss: Model predictions should match training data

Regularization: Prevent the model from doing *too* well on training data **(overfitting)**

Simple examples

<u>L2 regularization:</u> $R(W) = \sum_k \sum_l W_{k,l}^2$

L1 regularization: $R(W) = \sum_k \sum_l |W_{k,l}|$

Elastic net (L1 + L2): $R(W) = \sum_k \sum_l \beta W_{k,l}^2 + |W_{k,l}|$

More complex:

Dropout

Batch normalization

Cutout, Mixup, Stochastic depth, etc…

Loss Functions

- We have some dataset of (x, y)
- We have a **score function:**
- We have a **loss function**:

$$s = f(x; W) = Wx$$

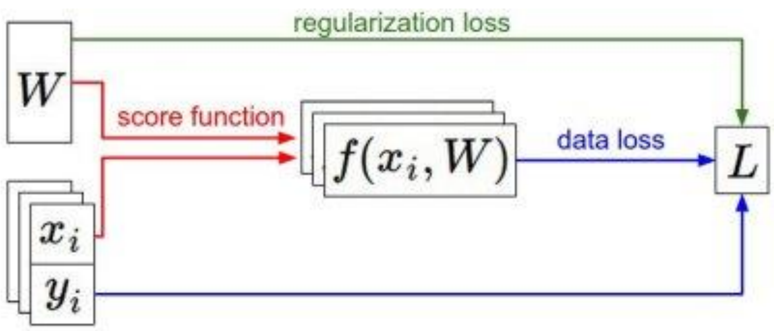
Linear classifier

$$L_i = -\log\left(\frac{e^{s_{y_i}}}{\sum_j e^{s_j}}\right)$$ Softmax

SVM

$$L_i = \sum_{j \neq y_i} \max(0, s_j - s_{y_i} + 1)$$

$$L = \frac{1}{N} \sum_{i=1}^{N} L_i + R(W)$$ Full loss

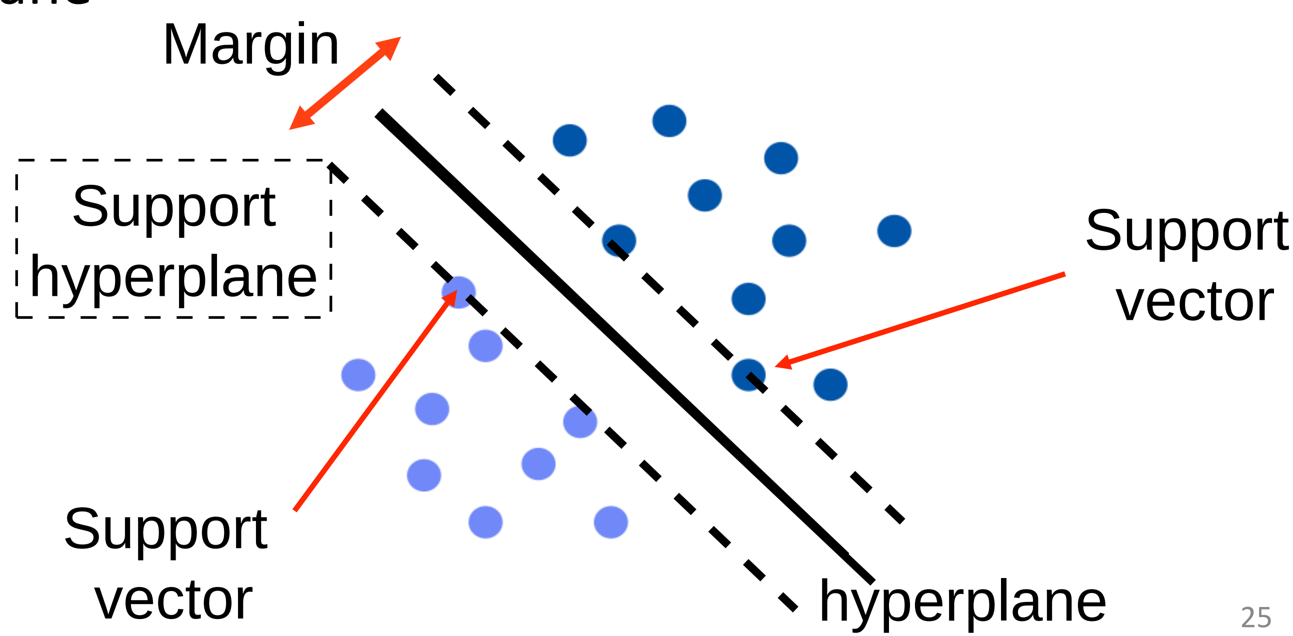


Support Vector Machine (SVM)

- A supervised learning method used for classification and regression
- Used in many applications : face detection, text recognition, video annotation, stock analysis ...
- SVM is recently developed during 1992--1995
- Prof. Chih-Jen Lin provides “libsvm” is a very useful and simple tool to use SVM for any applications in the world

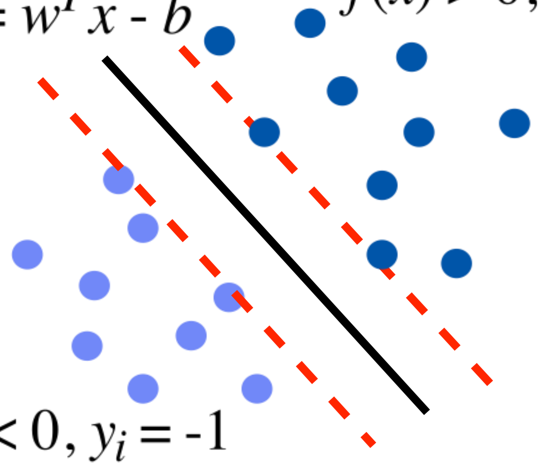
Support Vector Machine (SVM)

- The main effort is to find the hyperplane
  - Support hyperplane -- two parallel hyperplane that maximize the margin
  - Support vector -- the vector point locate on support hyperplane



Support Vector Machine (SVM)

$$Suppose\ \{x_i, y_i\}, i = 1,\ldots,n\ and\ x_i \in R^d, y_i \in \{+1, -1\}$$



Two support hyperplanes

$$\begin{aligned} w^T x &= b + \delta \\ w^T x &= b - \delta \end{aligned} \quad \Rightarrow \quad \begin{aligned} w^T x &= b + 1 \\ w^T x &= b - 1 \end{aligned}$$

So, rewrite the inequality

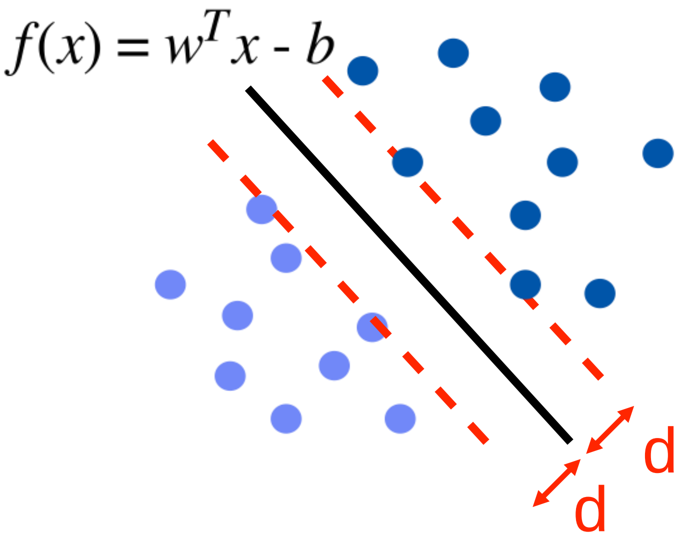
$$\begin{aligned} w^T x - b &\geq +1 \ \in y_i = +1 \\ w^T x - b &\leq -1 \ \in y_i = -1 \end{aligned}$$

The inequality

$$\Rightarrow \quad y_i (w^T x - b) - 1 \geq 0$$

Support Vector Machine (SVM)

$$Suppose\ \{x_i, y_i\}, i = 1,\ldots,n\ and\ x_i \in R^d, y_i \in \{+1, -1\}$$



The separating margin

$$d = \frac{(||b+1|| - ||b||)}{||w||} = \frac{1}{||w||}$$

$$d = \frac{(||b|| - ||b-1||)}{||w||} = \frac{1}{||w||}$$

Maximize separating margin

$$\frac{2}{||w||}$$

Minimize

$$\frac{1}{2}||w||^2$$

Support Vector Machine (SVM)

The primal problem of SVM

Minimize $\frac{1}{2}||w||^2$

Subject to $y_i\left(w^T x - b\right) - 1 \geq 0$

Lagrange Multiplier Method translate the above two equation and find out w, b, a that minimum L(w,b,a)

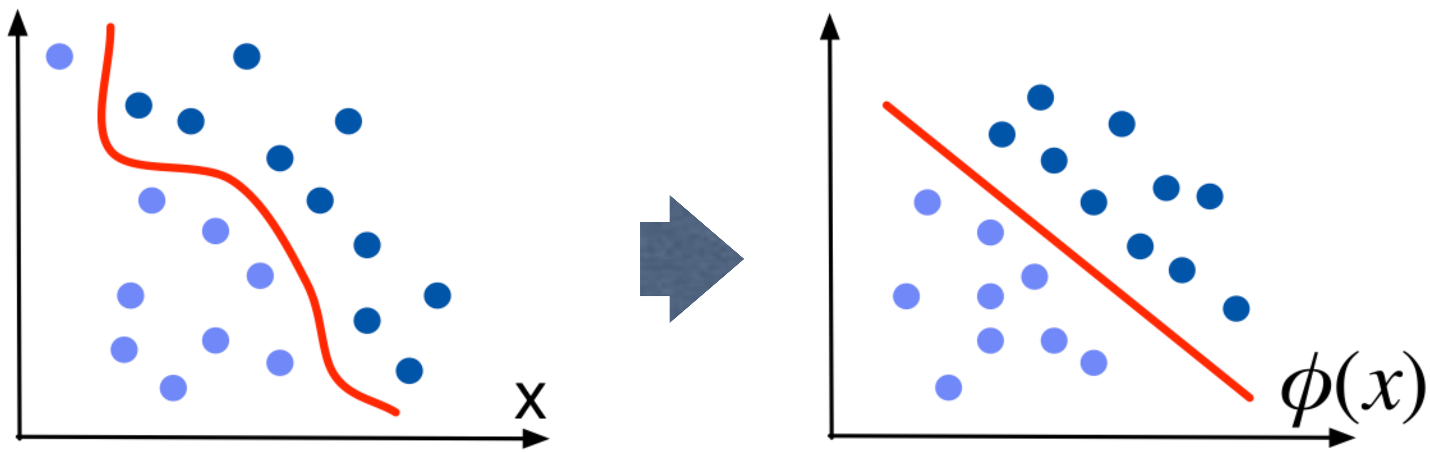
$$L(w, b, \alpha) = \frac{1}{2}|w|^2 - \sum_{i=1}^{N} \alpha_i \left[y_i \left(w^T x_i - b \right) - 1 \right]$$

Decision function:

$$\operatorname{sgn}\left(\boldsymbol{w}^T \phi(\boldsymbol{x}) + b\right) = \operatorname{sgn}\left(\sum_{i=1}^{N} y_i \alpha_i K(\boldsymbol{x}_i, \boldsymbol{x}) + b\right)$$

Non-linear Classification

- If the data is not linear, we can translate these vectors into higher-dimensional feature space



Kernel – RBF (Radial Basis Function)

- The RBF kernel nonlinearly maps samples into a higher dimensional space
- The RBF kernel can handle the case when the relation between class labels and attributes is nonlinear

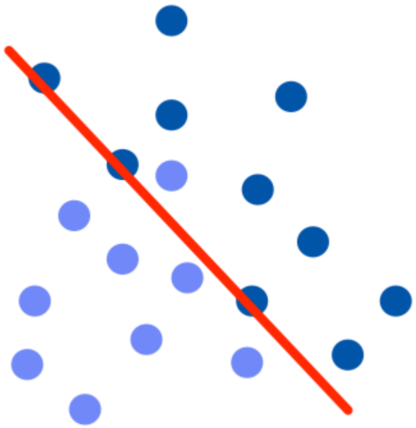
Linear Kernel $$K(x_i, x_j) = x_i^T x_j$$

RBF Kernel $$K(x_i, x_j) = e^{-\gamma |x_i - x_j|^2}$$

γ is a variable which plays an important role in SVM

Non-separable Case

- In real world, it is hard to find a hyperplane completely separating these data



So, rewrite the inequality

$$w^T x - b \geq +1 - \xi_i \quad \in y_i = +1$$

$$w^T x - b \leq -1 + \xi_i \quad \in y_i = -1$$

$$\xi_i \geq 0 \quad for\ i$$

Cost for penalty

$$Cost = \mathrm{C} \left(\sum_i \xi_i \right)^k$$

Non-separable Case

- Rewrite the equation of hyperplane with cost function

New primal problem of SVM

Minimize $\frac{1}{2}\|w\|^2 + C\sum_i \xi_i$

Subject to $y_i(w^T x - b) - 1 + \xi_i \geq 0 \quad for\ i$

$\xi_i \geq 0 \quad for\ i$

C is a penalty weighting for cost function and also plays an important role in SVM

https://www.csie.ntu.edu.tw/~cjlin/libsvm/#GUI

Combining Models

- Many models available in machine learning for classification and regression
- Instead of using one model in isolation improved performance can be obtained by combining different models

Two Methods of Combining

- 1. Committee: train L different models
  - Make predictions using average of the predictions
  - Boosting is a variant of Committee
    - Train multiple models in sequence
      - Error function used to train a model depends on performance of previous models
- 2. Select one of L models to make the prediction
  - Choice of model is a function of input variables
    - Different models become responsible for different regions of input space
  - Decision tree is an example

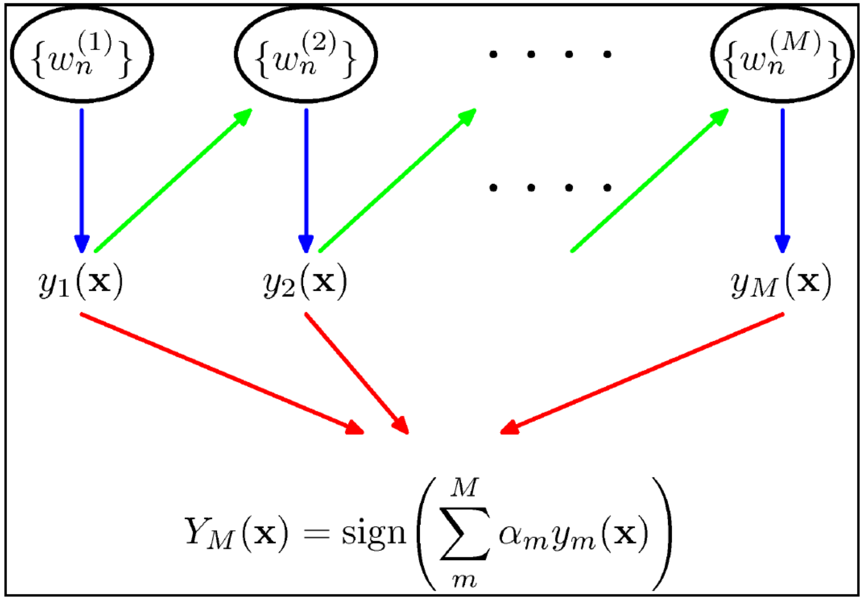
Boosting

- Incrementally adding models to the ensemble
- After a weak learner is added, the data are reweighted:
  - examples that are misclassified gain weight and examples that are classified correctly lose weight

AdaBoost (Adaptive Boosting)

- Most widely used form of boosting is the AdaBoost algorithm
- Boosting can give good results even if base classifiers have performance, only slightly better than random
  - Hence base learners are called weak learners
- Boosting can be extended to regression

Boosting Framework



- Each base classifier $y_m(x)$ is trained on a weighted form of the training set.
- Weights $w^{(m)}_n$ depend on the performance of the previous base classifier $y_{m-1}(x)$
- Once all base classifiers are trained, they are combined to give the final classifier Y_M (x)

Adaboost Algorithm

1. Initialize $w^{(1)}{}_n = 1/N$ for $n = 1, ..N$
2. For $m = 1, ..M$
   a) Fit a classifier $y_m(\mathbf{x})$ by minimizing weighted error

      $J_m = \sum_{n=1}^{N} w_n^{(m)} I(y_m(x_n) \neq t_n)$

      where $I(y_m(x_n) \neq t_n)$ is the indicator function and equals 1 when $I(y_m(x_n) \neq t_n)$ and 0 otherwise

   b) Evaluate the quantities $\epsilon_m = \frac{J_m}{\sum_{n=1}^{N} w_n^{(m)}}$

      and then use these to evaluate $\alpha_m = \ln\left\{\frac{1-\epsilon_m}{\epsilon_m}\right\}$

   c) Update the data weighting coefficients

      $w_n^{(m+1)} = w_n^{(m)} \exp\{\alpha_m I(y_m(x_n) \neq t_n)\}$

3. Make predictions using final model

   $Y_M(x) = sign(\sum_{m=1}^{M} \alpha_m y_m(x))$

ϵ_m: weighted error rate of classifier

$\boldsymbol{\alpha}_m$: weighting coeffts give greater weight to more accurate classifiers

$w_n^{(m+1)}$: increase weight of misclassified data with exponential error

Illustration of Boosting

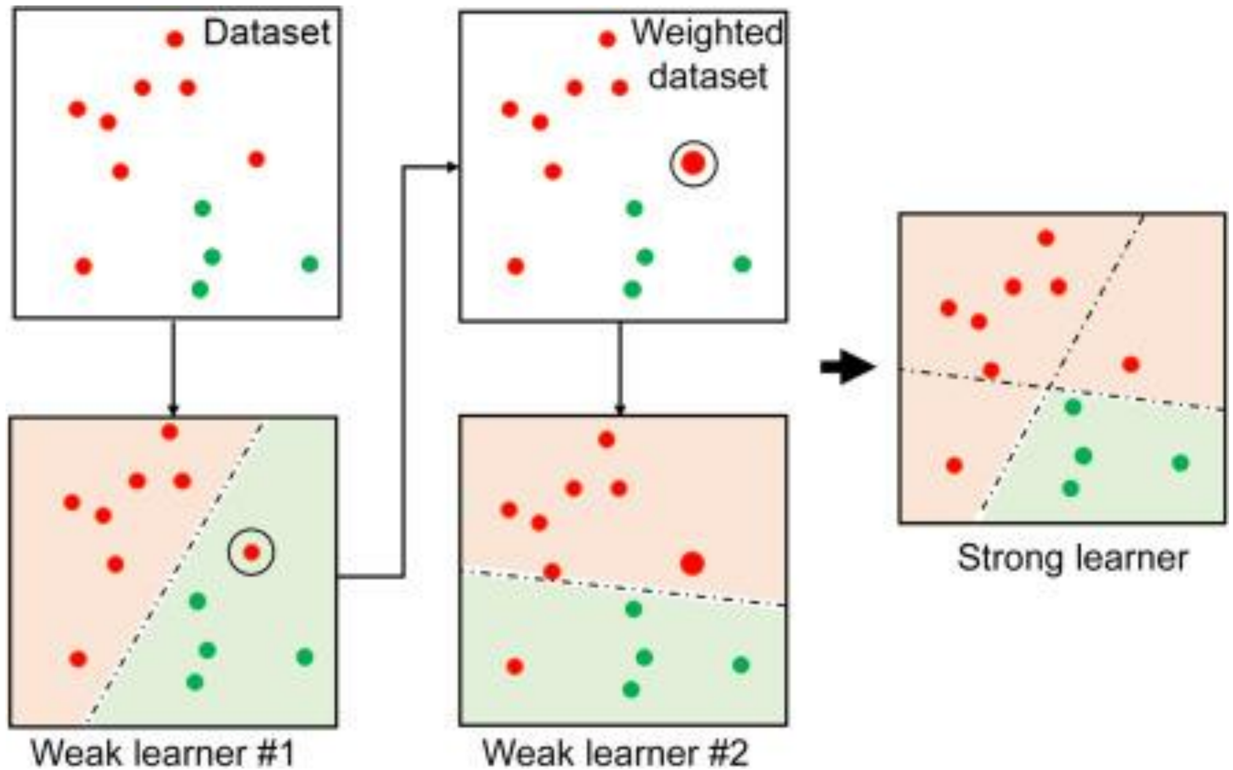


Illustration of Boosting

AdaBoost in Action

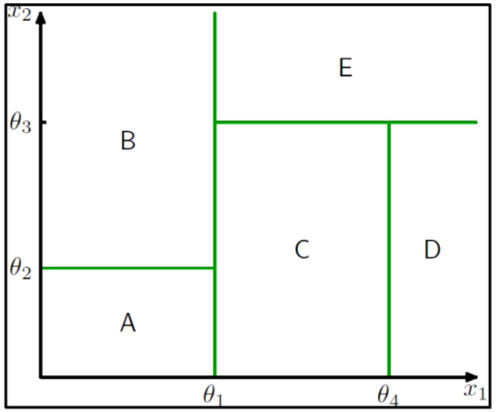
Kai O. Arras
Social Robotics Lab, University of Freiburg

Nov 2009 Social Robotics Laboratory

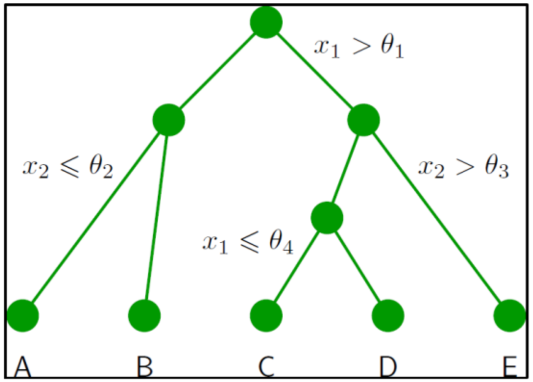
Decision Tree

- Choice of model is function of input variables
  - Different models become responsible for making predictions in different regions of input space
  - A sequence of binary selections corresponding to traversal of a tree structure

2-D input space $\boldsymbol{x}=[x_1,x_2]$
Partitioned into five regions
using axis aligned boundaries



Binary Tree corresponding to
Partitioning of input space

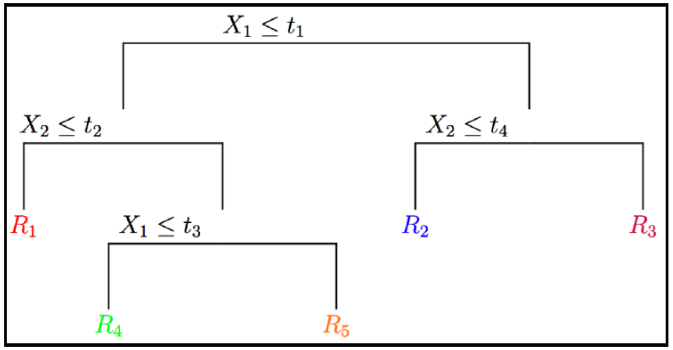


Use of Tree Models

- Can be used for both regression and classification
- So they are called Classification and Regression Trees (CART)
- Within each region there is a separate model to predict a target variable
  - In regression, we might simply predict a constant over each region
  - In classification, we might assign each region to a specific class
- Key property: they are interpretable by humans
  - To predict a disease, is temperature greater than a threshold?, is BP less than a threshold? Each leaf is a diagnosis

Regression Tree with 2 Inputs

Two input attributes (x_1, x_2), a real output



First node asks if $x_1 \leq t_1$

If yes, ask if $x_2 \leq t_2$

If yes, we are at quadrant R_1

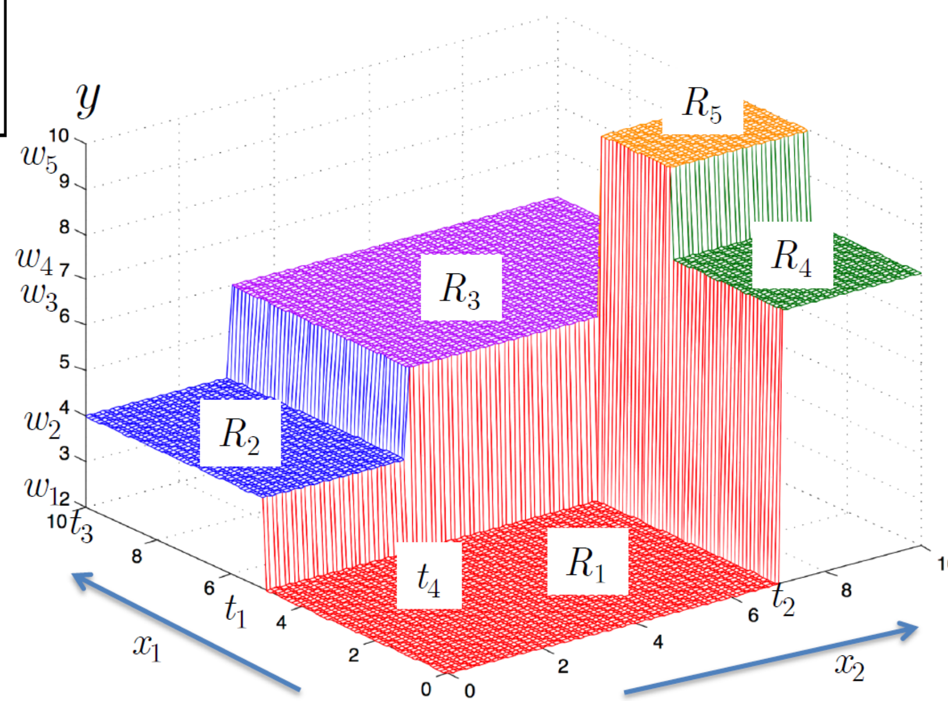
Associate a value of y with each region

Result of axis-parallel splits:

2-d space partitioned into 5 regions

Each region is associated with a mean response

Result: piecewise constant function

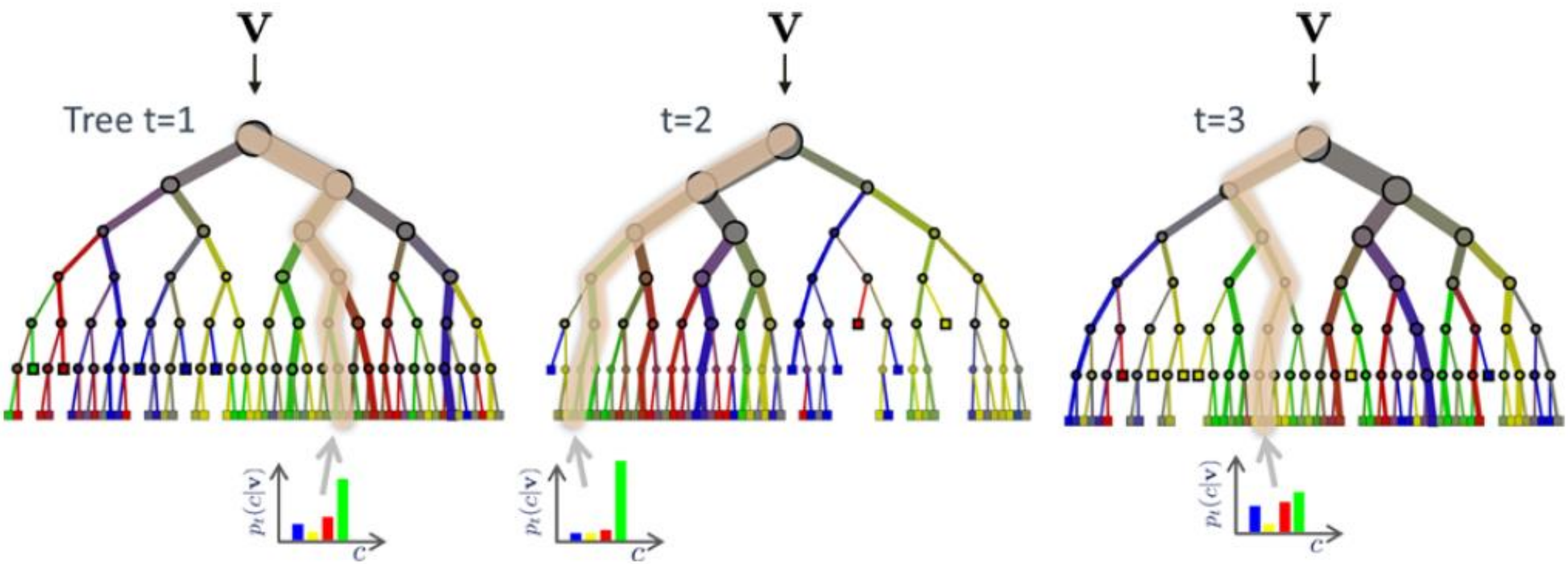


K. Murphy, Machine Learning, 2012

Model can be written as:

$$f(\boldsymbol{x}) = E[y \mid x] = \sum_{m=1}^{M} w_m I(\boldsymbol{x} \in R_m) = \sum_{m=1}^{M} w_m \phi(\boldsymbol{x}; \boldsymbol{v}_m)$$

Random Forest

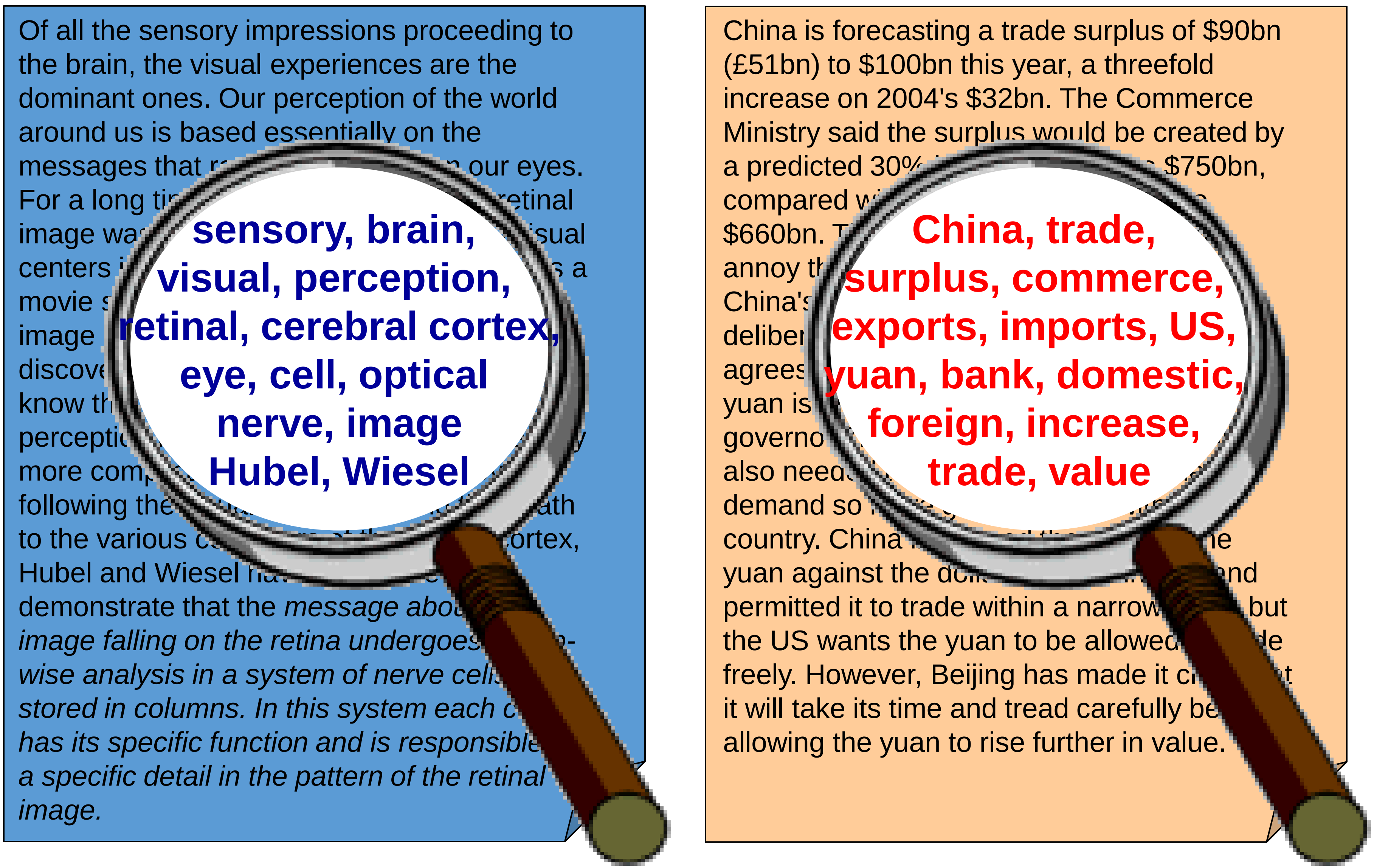


T = # trees

Examples

- Bag of Word
- Viola and Jones
- Pedestrian detection based on HoG

Analogy to documents



ICCV 2005 short course, L. Fei-Fei

Bag of *visual words*

- Image patches
- BoW histogram
- Codewords

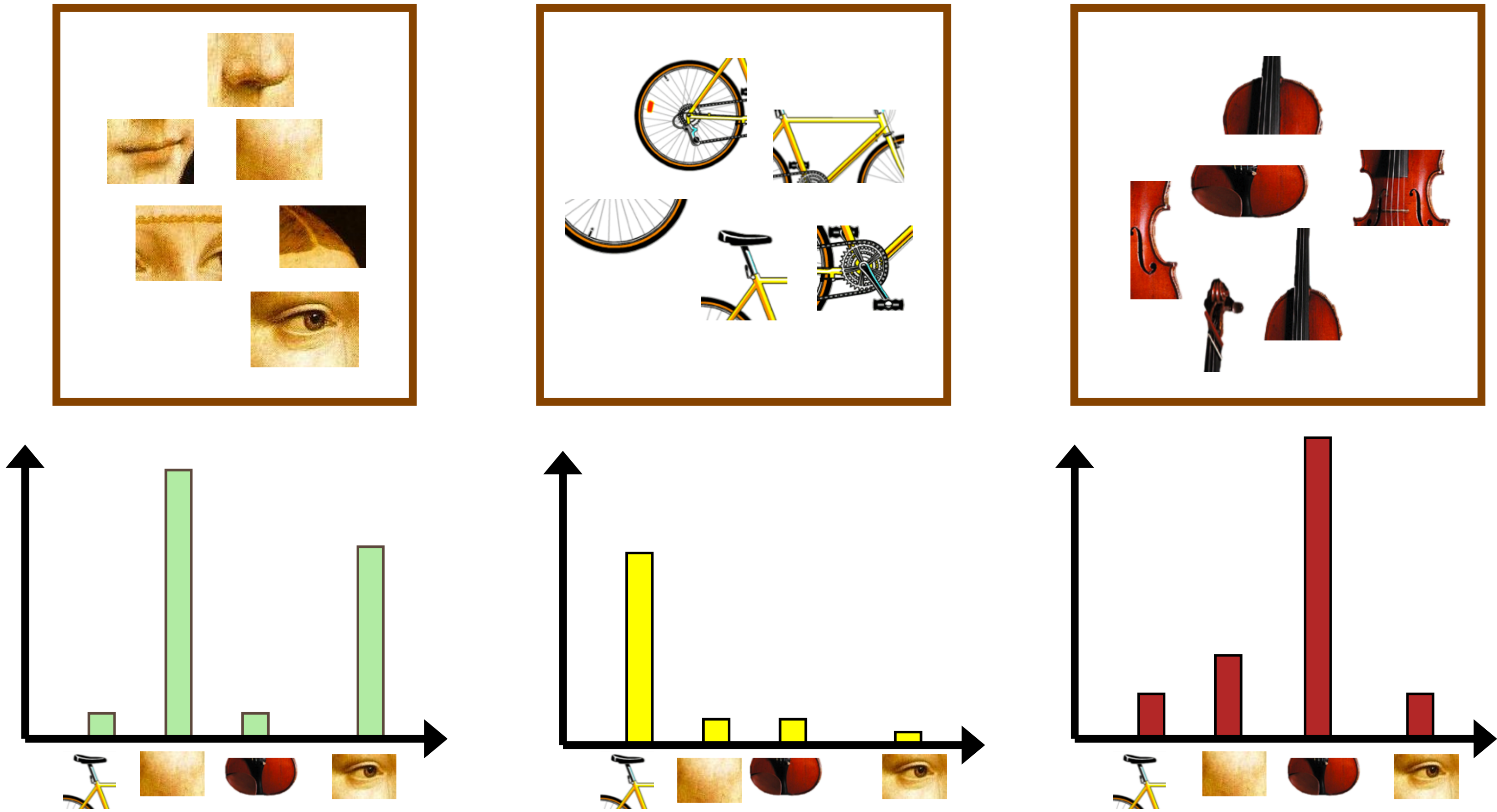


Image categorization with bag of words

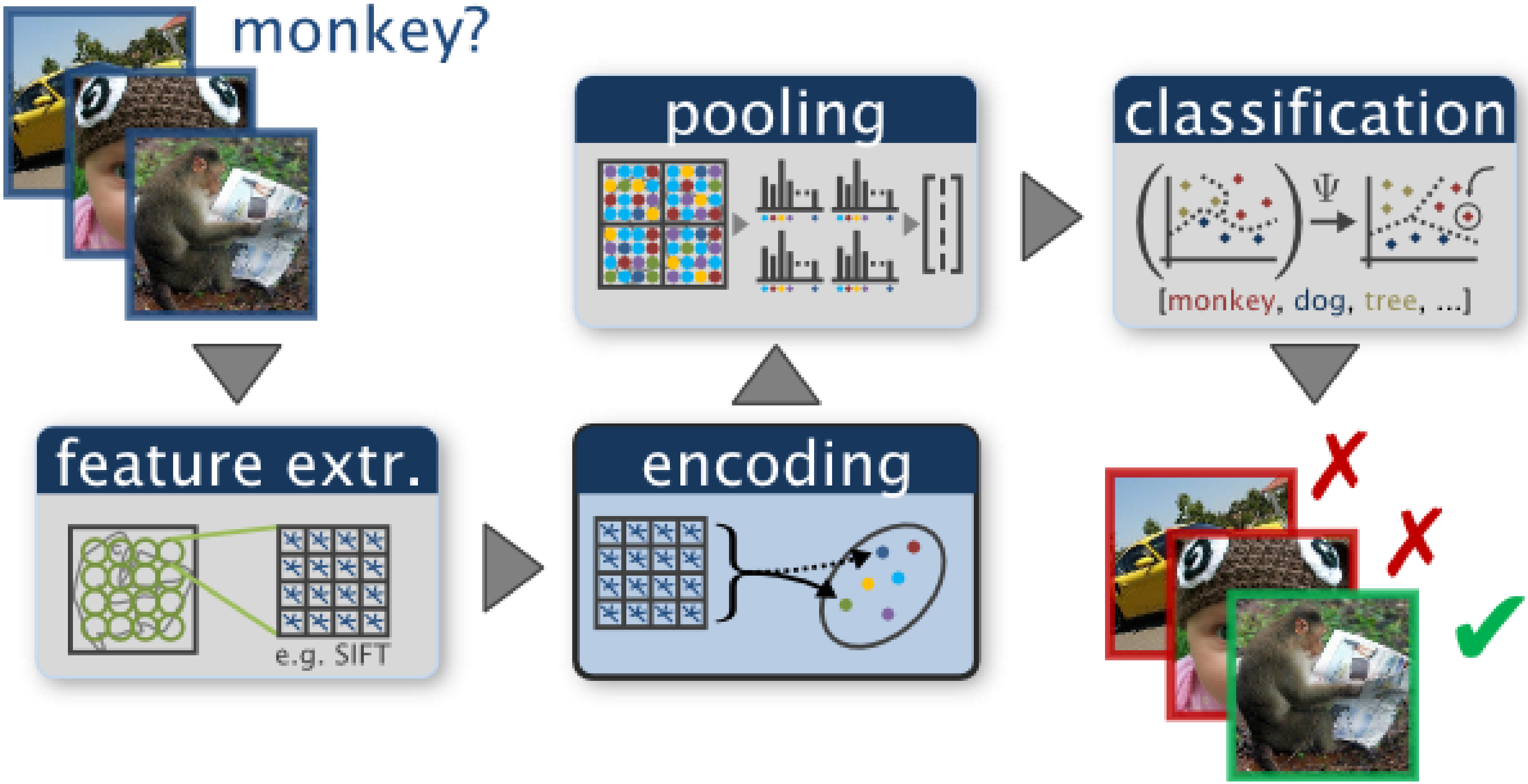
Training

1. Extract keypoints and descriptors for all training images
2. Cluster descriptors
3. Quantize descriptors using cluster centers to get "visual words"
4. Represent each image by normalized counts of "visual words"
5. Train classifier on labeled examples using histogram values as features

Testing

1. Extract keypoints/descriptors and quantize into visual words
2. Compute visual word histogram
3. Compute label or confidence using classifier

Bag of visual words image classification

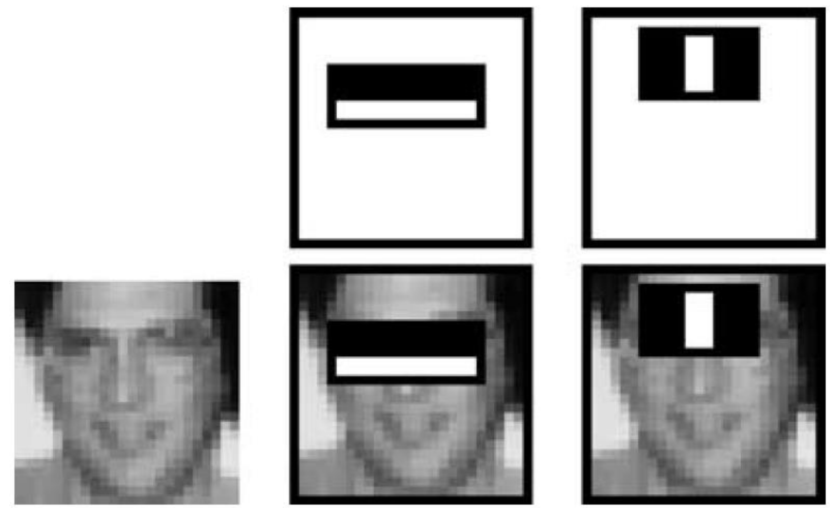


[Chatfieldet al. BMVC 2011]

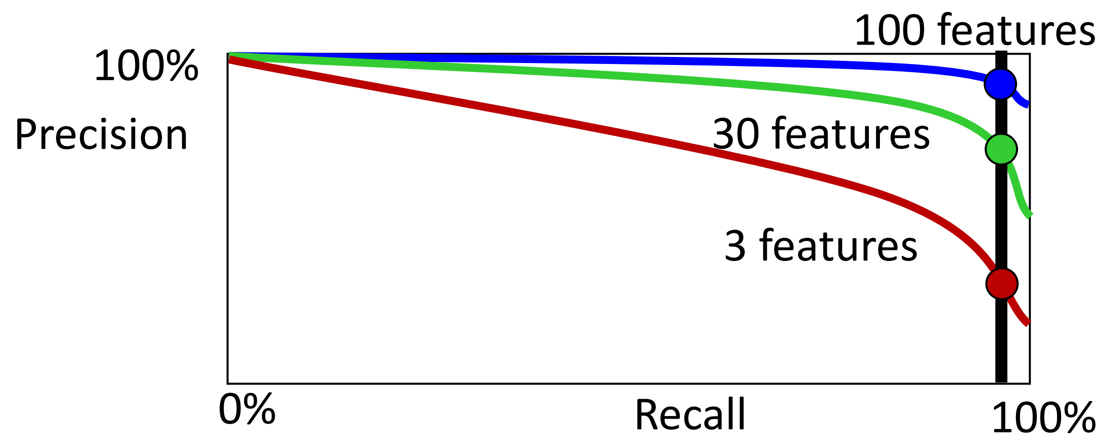
Viola & Jones Face Detection

P. Viola and M. J. Jones, "Robust Real-Time Face Detection," IJCV, 2004.

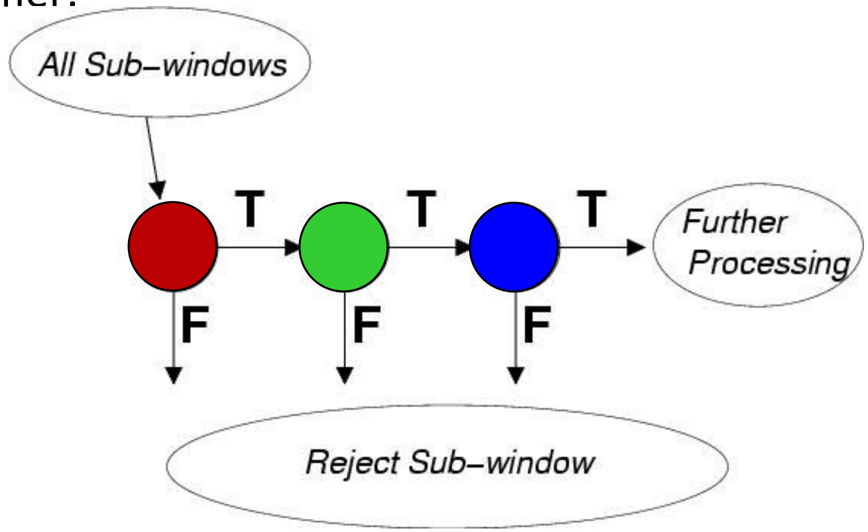
- Feature: Haar feature
  - Can be calculated efficiently with integral image technique
- Classifier
  - Adaboost
  - Cascade classifier (degenerate decision tree)



Cascade of Classifiers



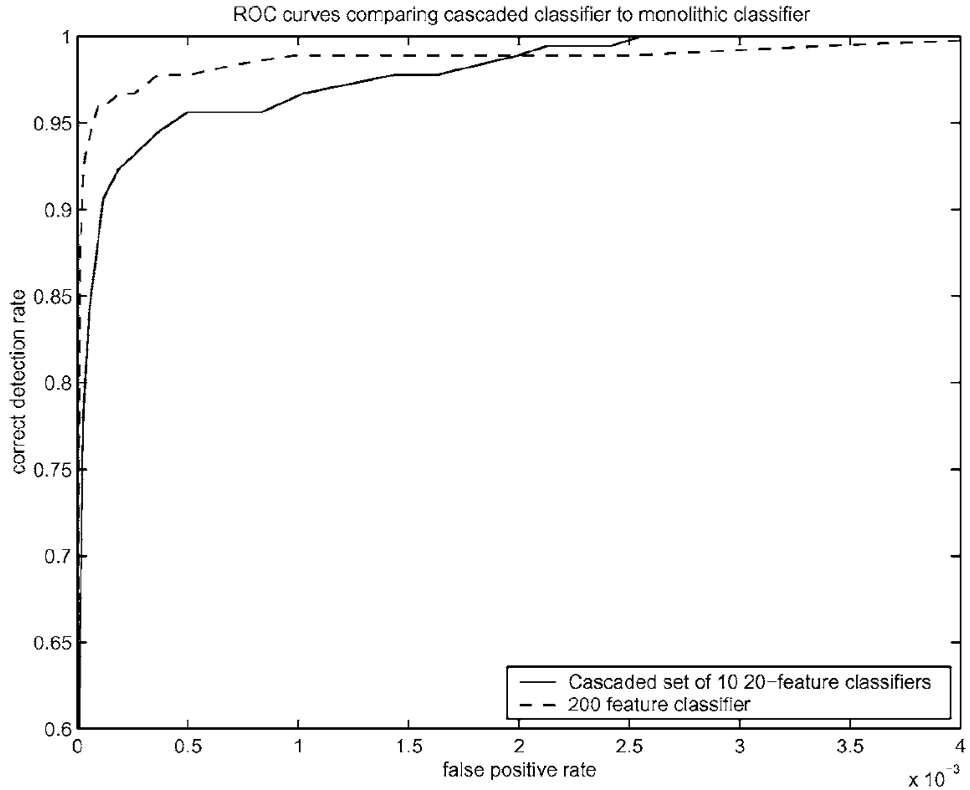
We want the complexity of the 3 features classifier with the performance of the 100 features classifier:

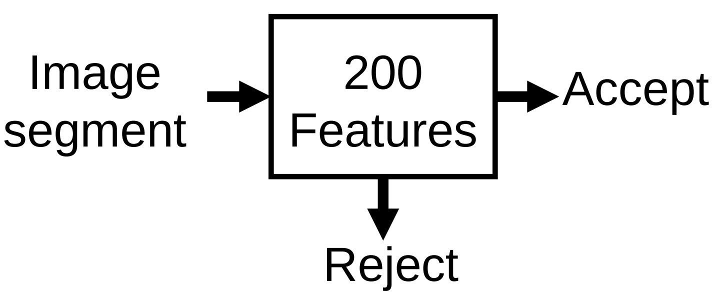


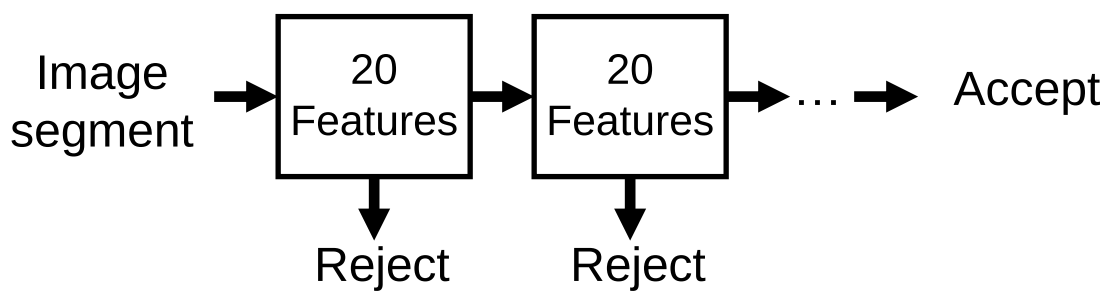
Select a threshold with high recall for each stage.

We increase precision using the cascade

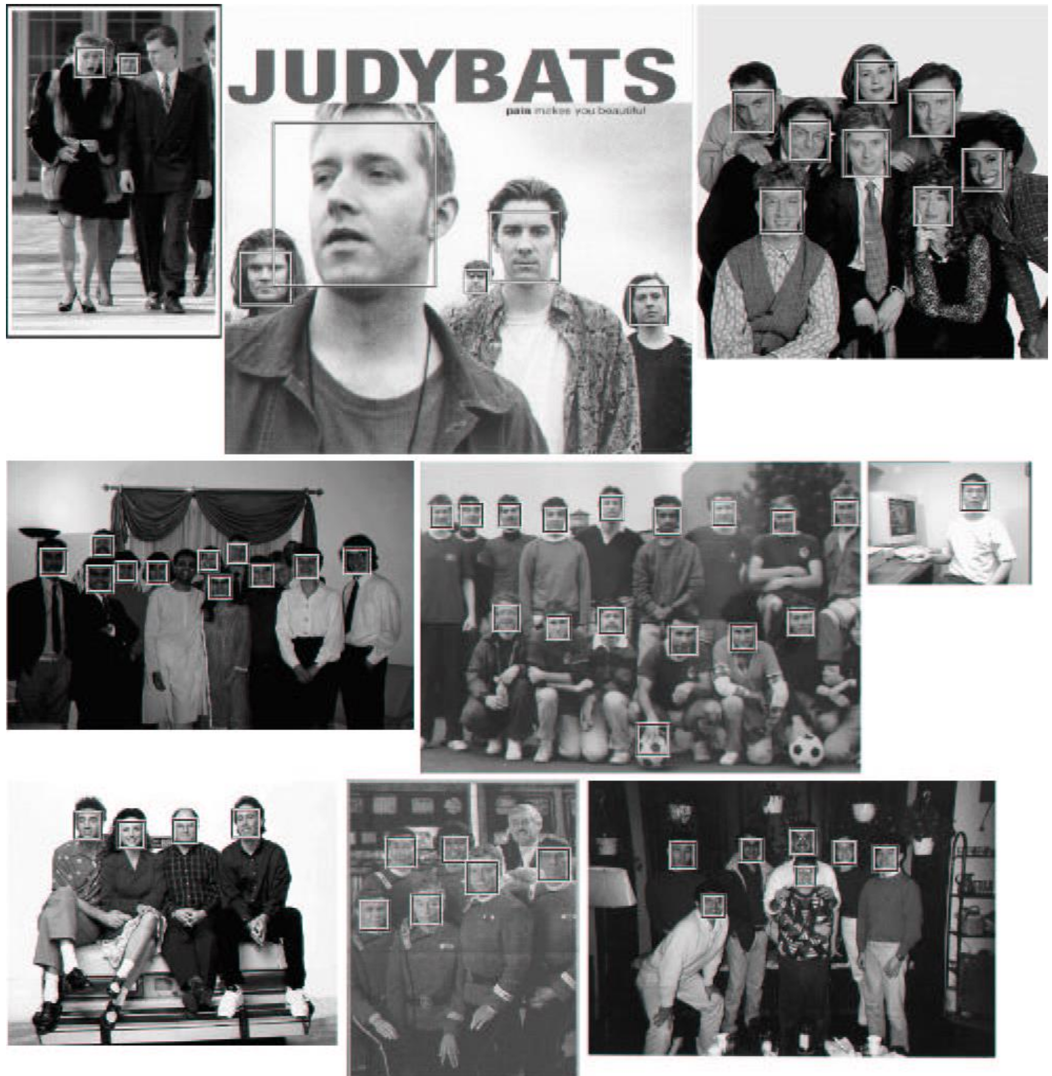
Viola & Jones Face Detection







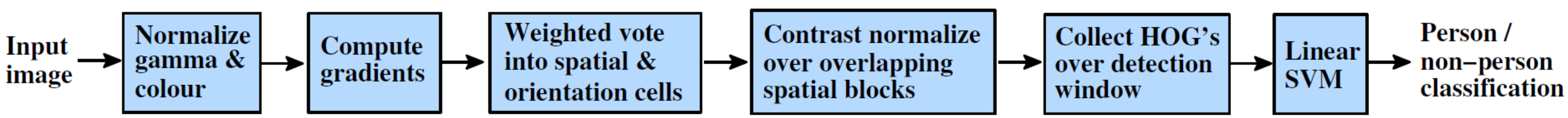
Viola & Jones Face Detection

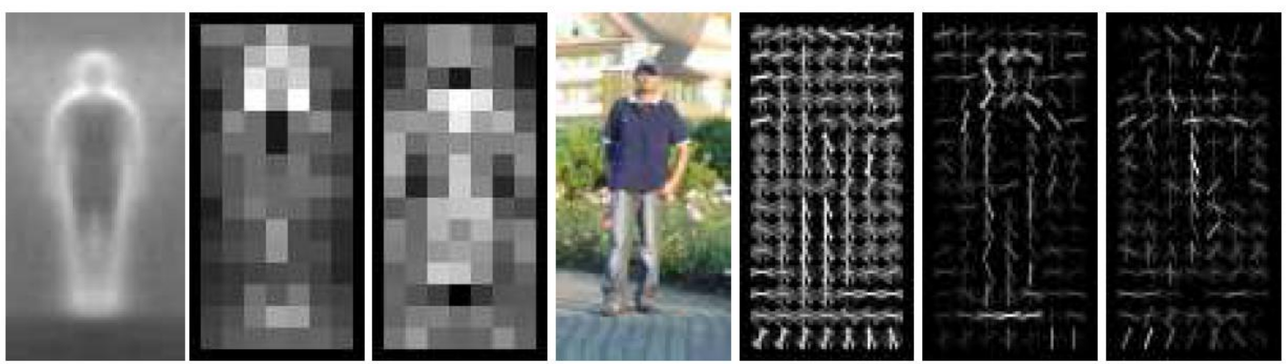


Pedestrian Detection with HoG

Navneet Dalal, Bill Triggs, “Histograms of Oriented Gradients for Human Detection,” CVPR 2005.

- Feature: Histogram of Gradient (HoG)
- Classifier: SVM





(a) (b) (c) (d) (e) (f) (g)

Figure 6. Our HOG detectors cue mainly on silhouette contours (especially the head, shoulders and feet). The most active blocks are centred on the image background just *outside* the contour. (a) The average gradient image over the training examples. (b) Each “pixel” shows the maximum positive SVM weight in the block centred on the pixel. (c) Likewise for the negative SVM weights. (d) A test image. (e) It’s computed R-HOG descriptor. (f,g) The R-HOG descriptor weighted by respectively the positive and the negative SVM weights.