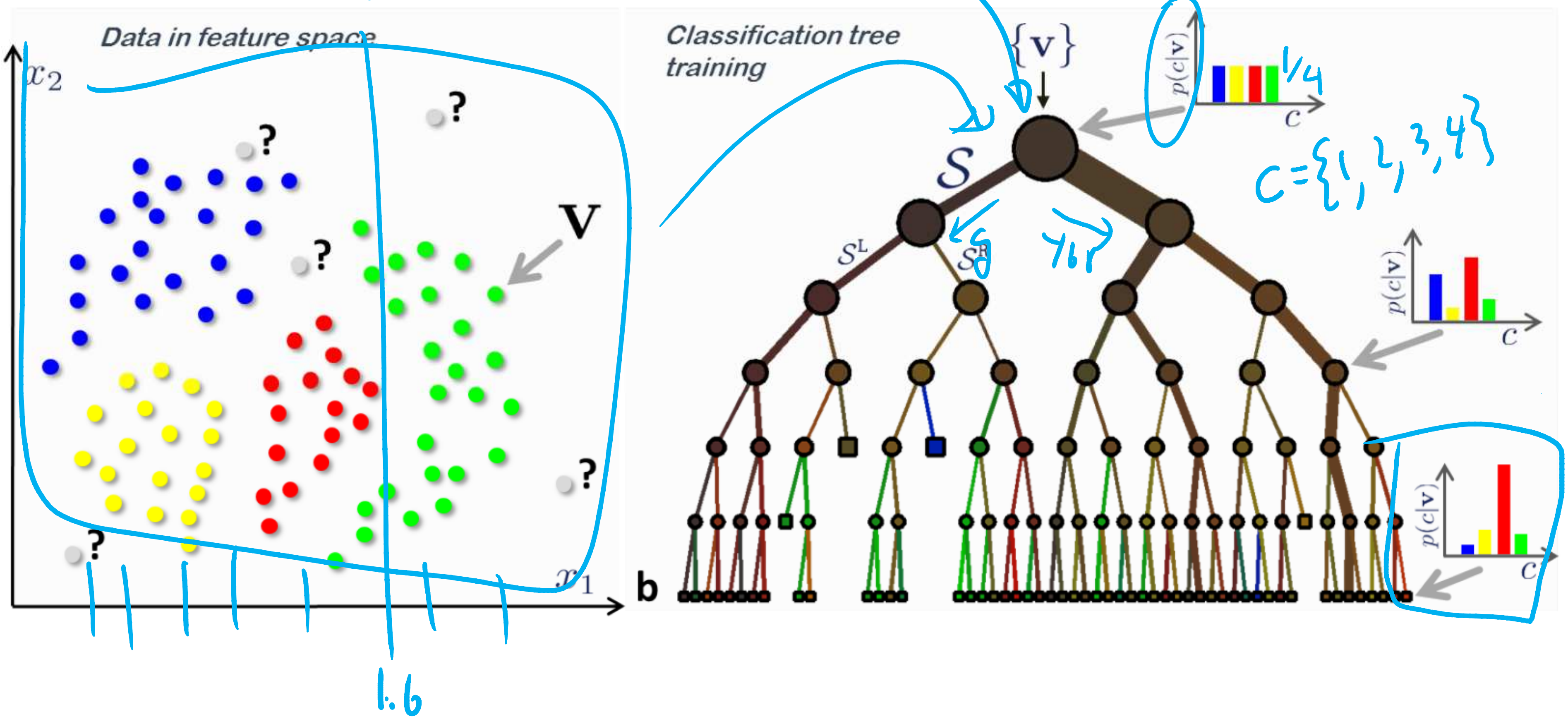
Outline of the lecture

This lecture discusses classification trees and how to incorporate then into an ensemble (random forest). It discusses:

- ❑ Random trees
- ❑ Random forests
- ❑ Object detection
- ❑ Kinect

Classification tree

$v = (x_1, x_2)$ Is $x_1 > 1.6$ $p(c|v)$

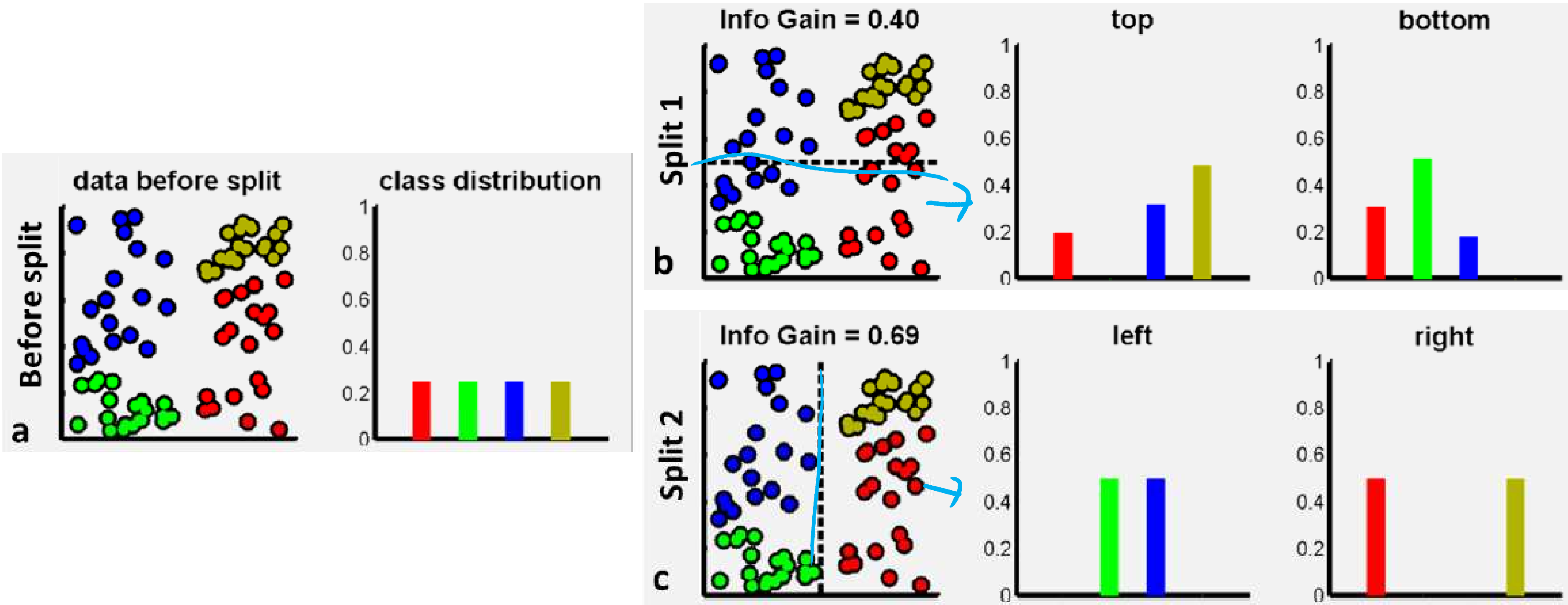


A generic data point is denoted by a vector $\mathbf{v} = (x_1, x_2, \cdots, x_d)$

$$\mathcal{S}_j = \mathcal{S}_j^{\mathrm{L}} \cup \mathcal{S}_j^{\mathrm{R}}$$

[Criminisi et al, 2011]

Use information gain to decide splits



$$I_j = H(\mathcal{S}_j) - \sum_{i \in \{L,R\}} \frac{|\mathcal{S}_j^i|}{|\mathcal{S}_j|} H(\mathcal{S}_j^i)$$

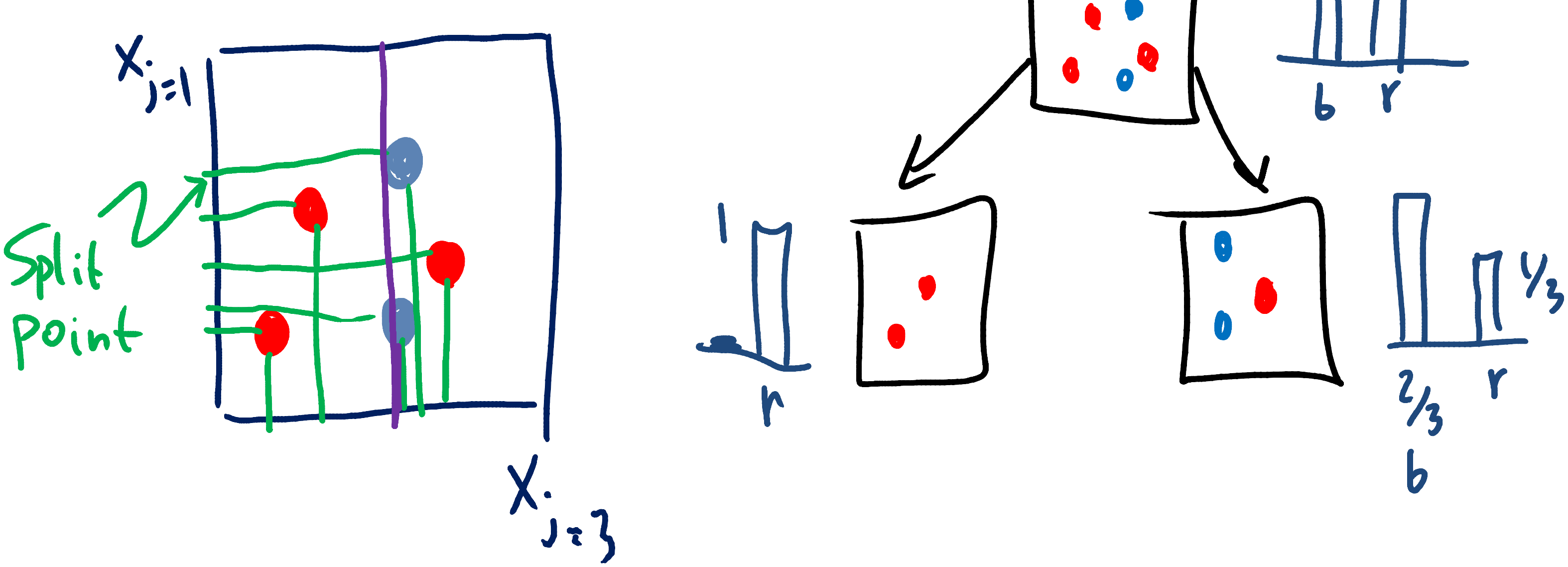
[Criminisi et al, 2011]

$d=3$ features
$n=5$ data

Building a random tree

$$X = \begin{array}{c|ccccc} & i=1 & i=2 & & & i=n=5 \\ \hline j=1 & 1 & 3 & 0 & 8 & 5 \\ j=2 & 0 & 6 & 2 & 9 & 5 \\ j=3 & 2 & 1 & 4 & 0 & 1 \end{array} \qquad Y = \begin{array}{ccccc} i=1 & i=2 & & & i=5 \\ \hline 0 & 1 & 0 & 1 & 0 \end{array}$$

Pick 2 features at random.



Random Forests algorithm

$\text{data} = \{(x_1, y_1), \ldots (x_N, y_N)\}$

1. For $b = 1$ to B: trees
   (a) Draw a bootstrap sample $\mathbf{Z}^*$ of size N from the training data. ↔ bagging
   (b) Grow a random-forest tree T_b to the bootstrapped data, by recursively repeating the following steps for each terminal node of the tree, until the minimum node size n_{min} is reached.
      i. Select m variables at random from the p variables. ←
      ii. Pick the best variable/split-point among the m. ←
      iii. Split the node into two daughter nodes. ←
2. Output the ensemble of trees $\{T_b\}_1^B$.

[From the book of Hastie, Friedman and Tibshirani]

Randomization

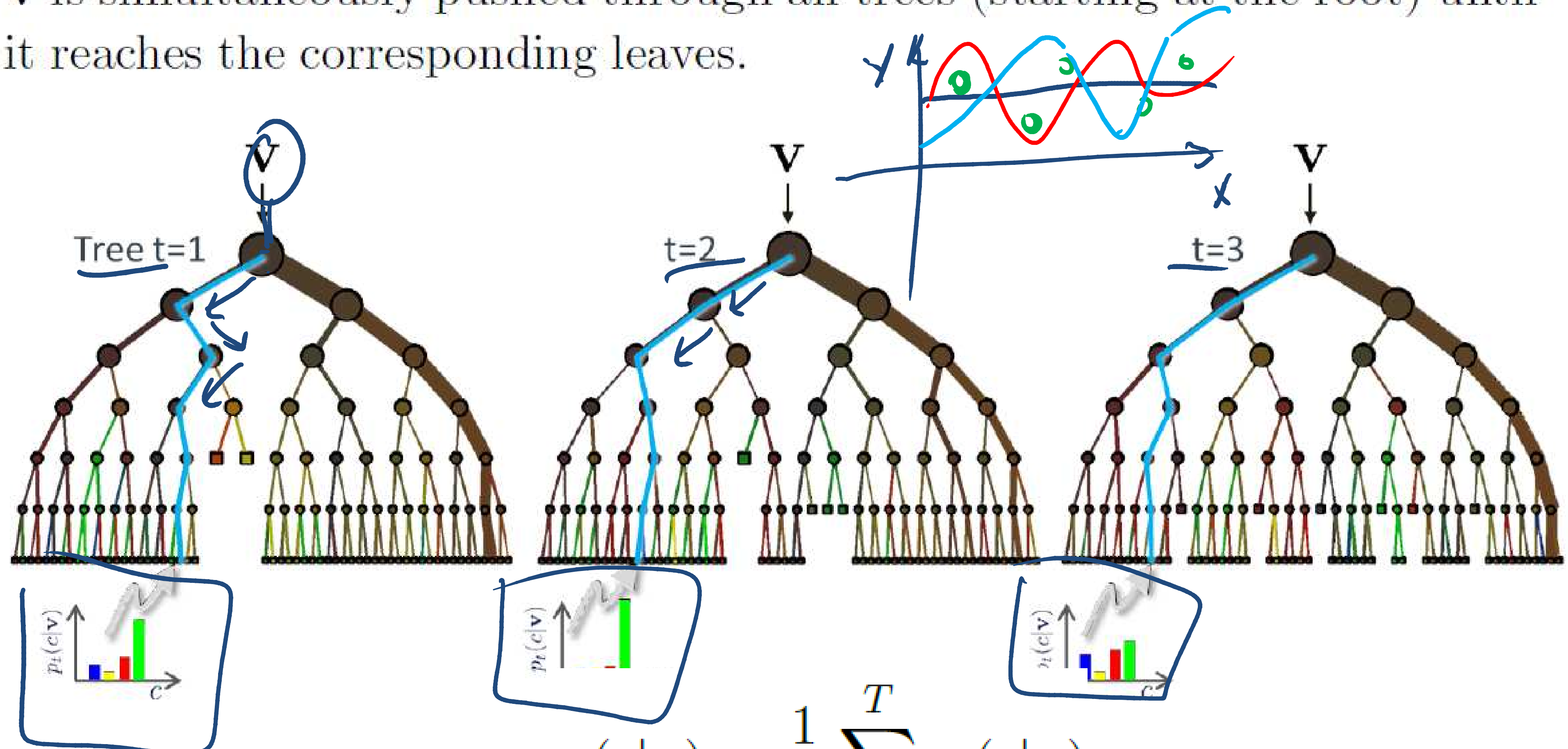
Randomized node optimization. If $\mathcal{T}$ is the entire set of all possible parameters $\boldsymbol{\theta}$ then when training the j^{th} node we only make available a small subset $\mathcal{T}_j \subset \mathcal{T}$ of such values.

$$\boldsymbol{\theta}_j^* = \arg \max_{\boldsymbol{\theta}_j \in \mathcal{T}_j} I_j.$$

[Criminisi et al, 2011]

Building a forest (ensemble)

In a forest with T trees we have $t \in \{1, \cdots, T\}$. All trees are trained independently (and possibly in parallel). During testing, each test point $\mathbf{v}$ is simultaneously pushed through all trees (starting at the root) until it reaches the corresponding leaves.

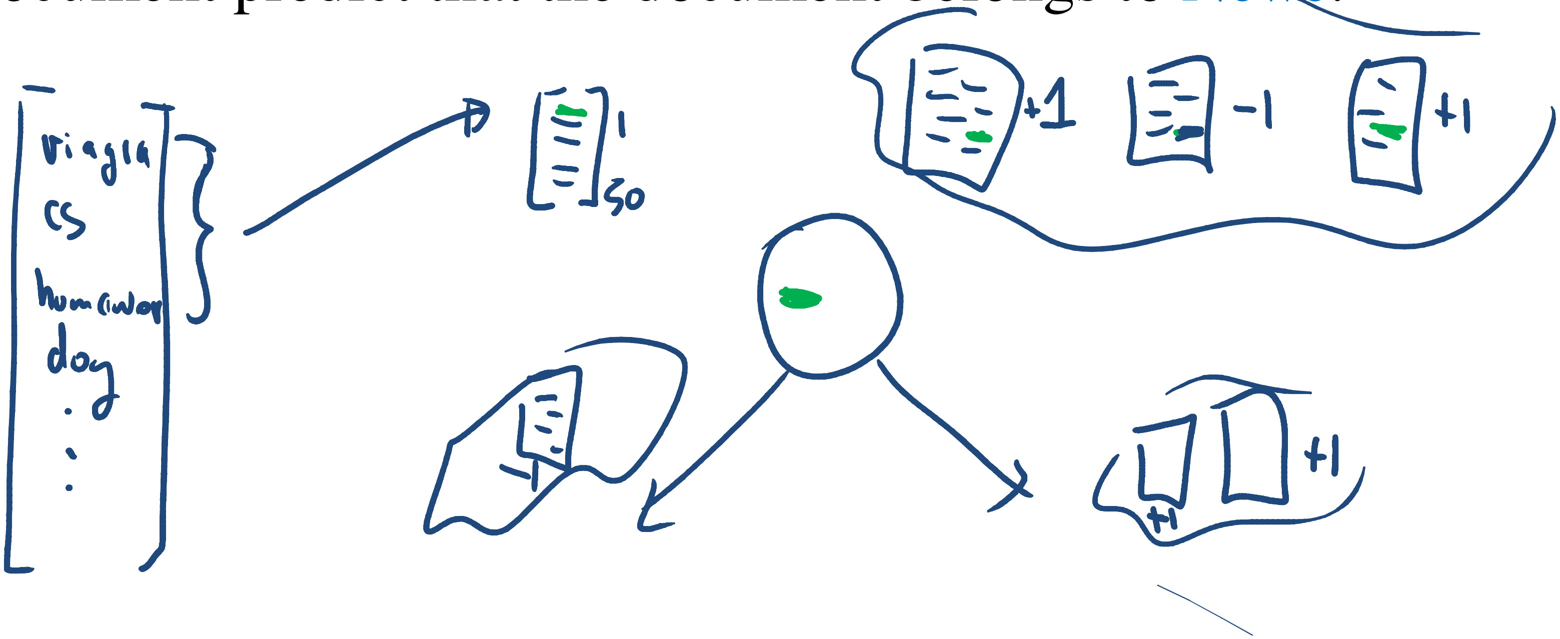


$$p(c|\mathbf{v}) = \frac{1}{T}\sum_{t=1}^{T} p_t(c|\mathbf{v})$$

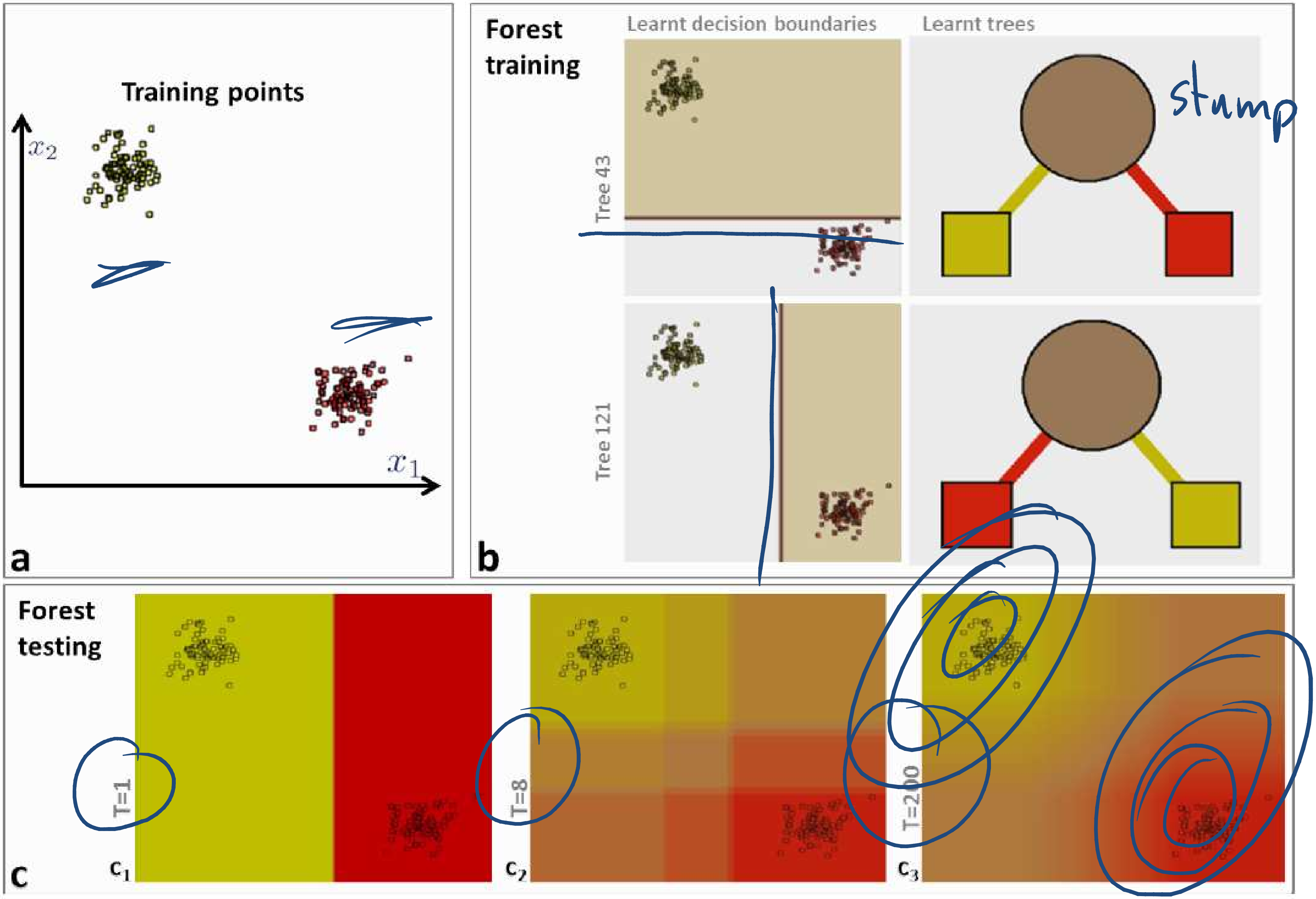
[Criminisi et al, 2011]

Text classification example

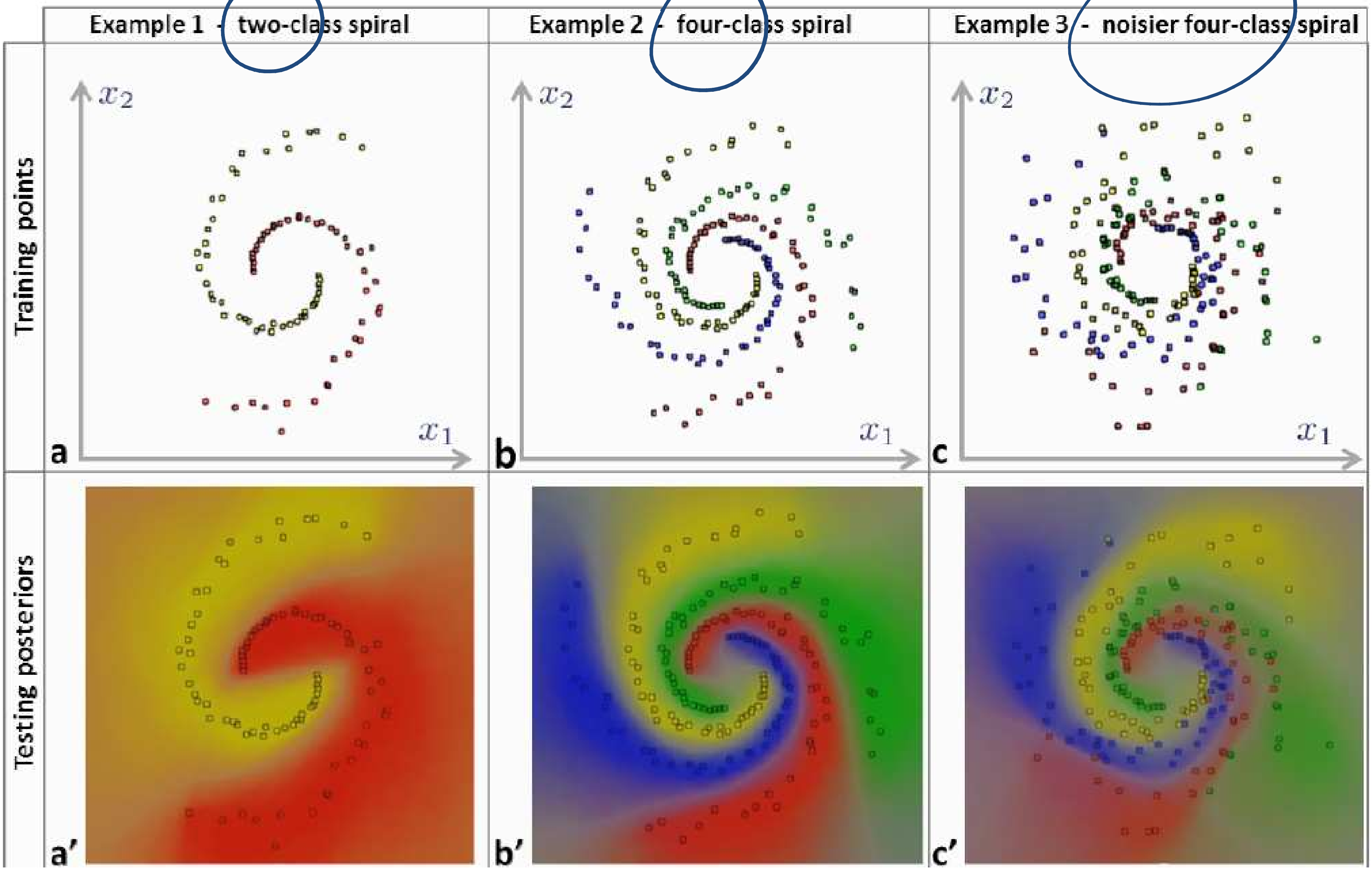
In news categorization, a possible term is *Bill Clinton*. A corresponding **weak learner (node)** is: If the term *Bill Clinton* appears in the document predict that the document belongs to **News**.



Effect of forest size

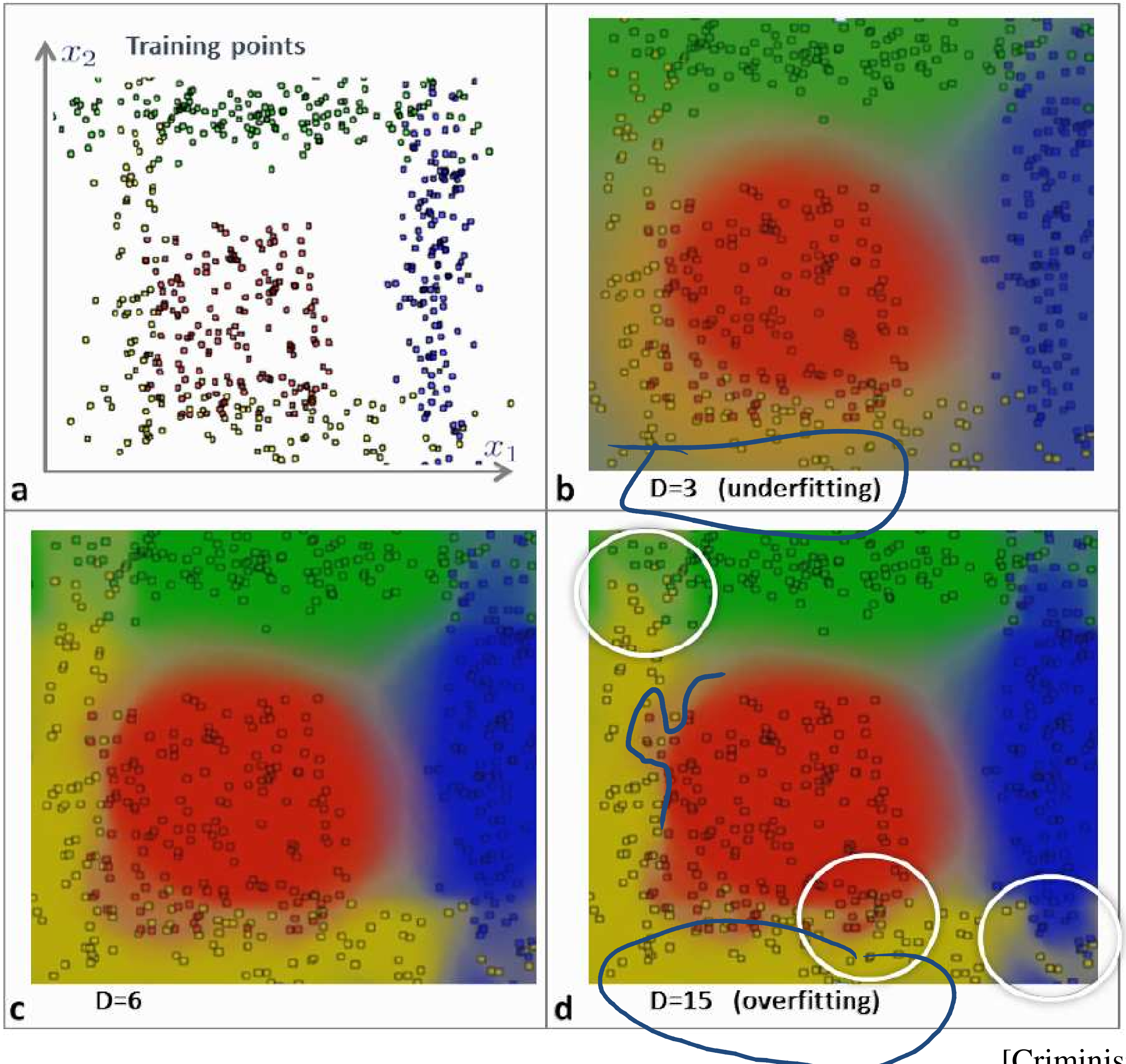


Effect of more classes and noise



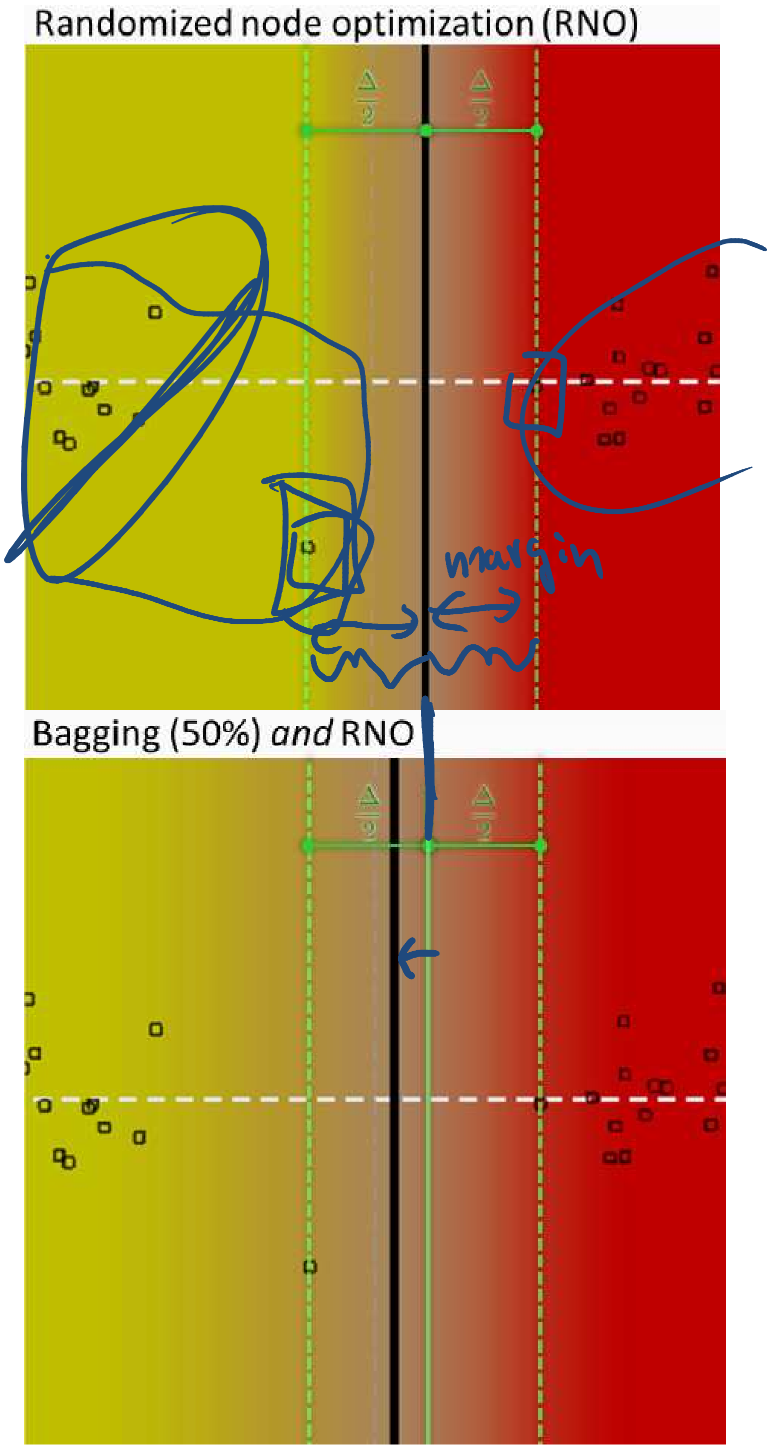
[Criminisi et al, 2011]

Effect of tree depth (D)



[Criminisi et al, 2011]

Effect of bagging



No bagging

max margin

[Criminisi et al, 2011]

Application to face detection

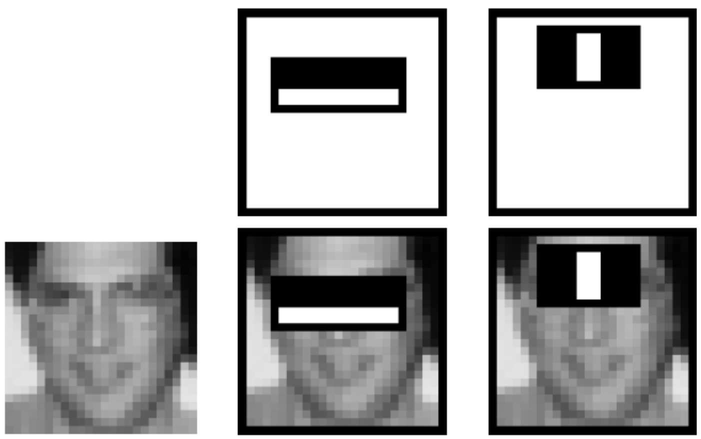
Training Data

- 5000 faces
  - All frontal
- 300 million non faces
  - 9400 non-face images

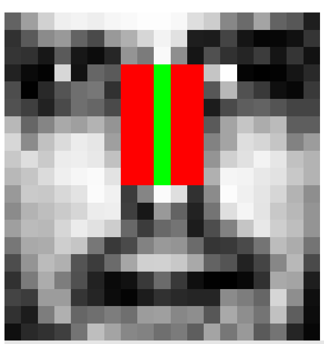
[Viola and Jones, 2001]

Object detection

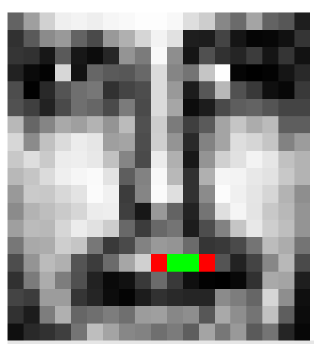
Idea: Extract simple features from all 24 by 24 pixel patches x_i. E.g., the value of a *two-rectangle feature is the difference between the sum* of the pixels within two rectangular regions. Then compare the level of activation (value of the feature f) with respect to a threshold (theta).



$$h_t(x_i) = \begin{cases} 1 & \text{if } f_t(x_i) > \theta_t \\ 0 & \text{otherwise} \end{cases}$$

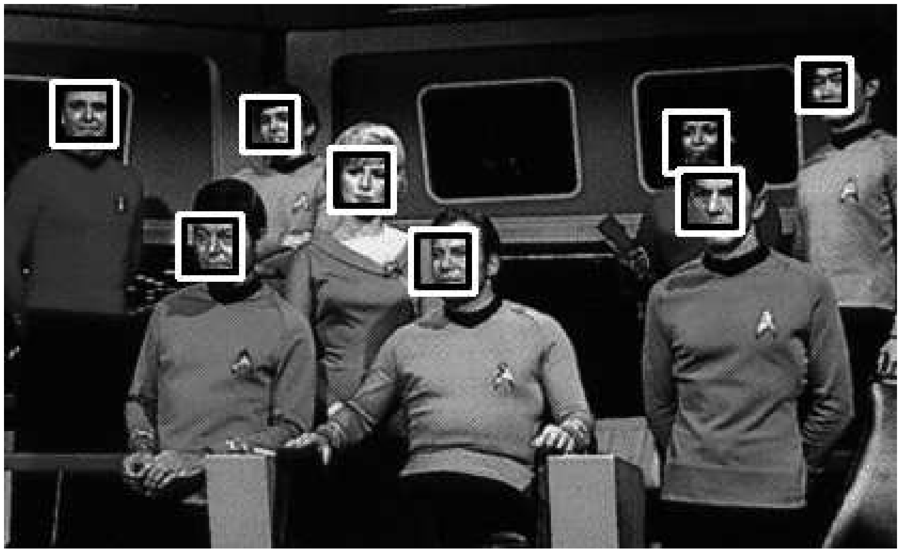


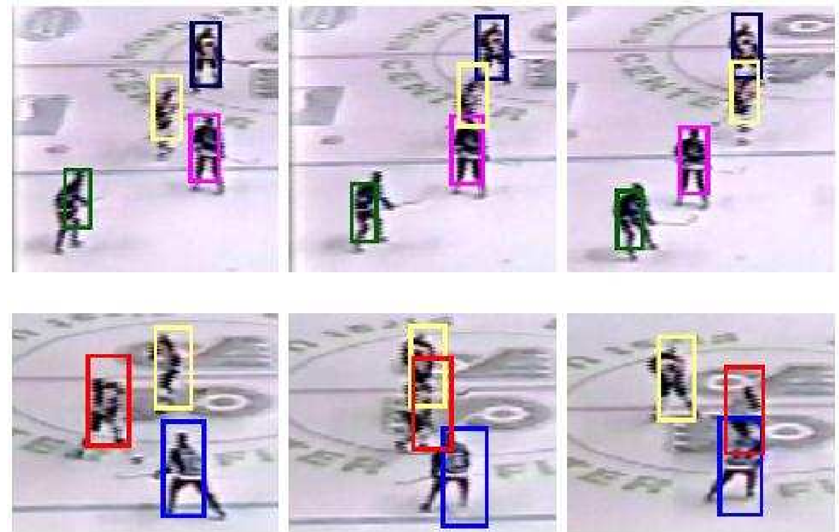
Relevant feature

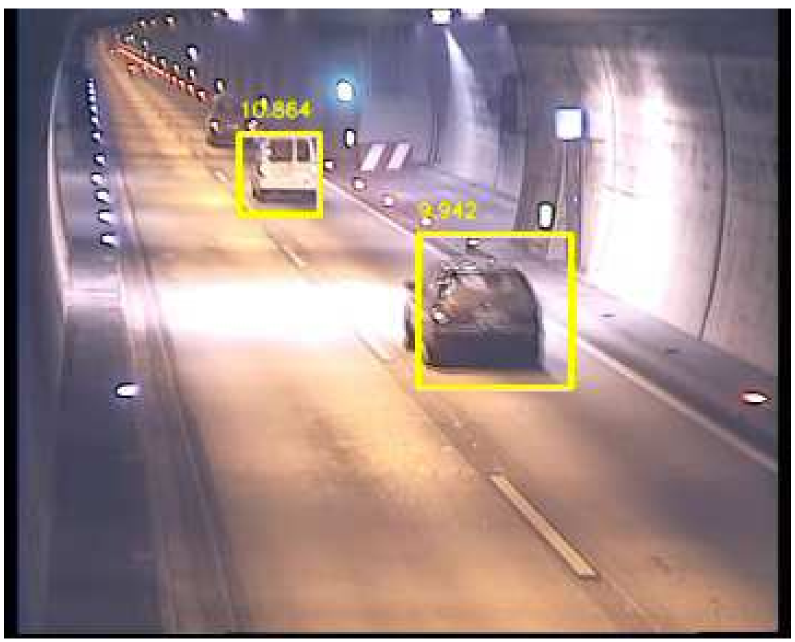


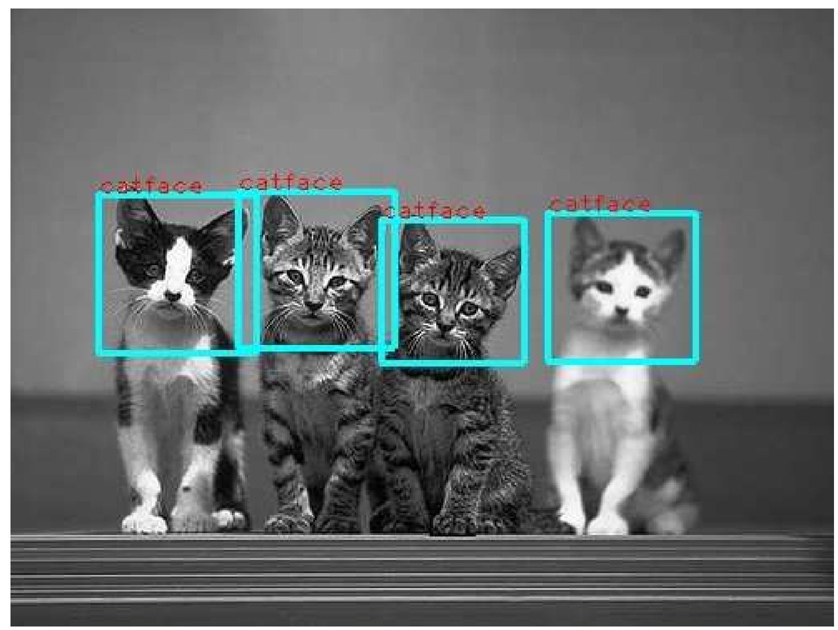
Irrelevant feature

Object detection

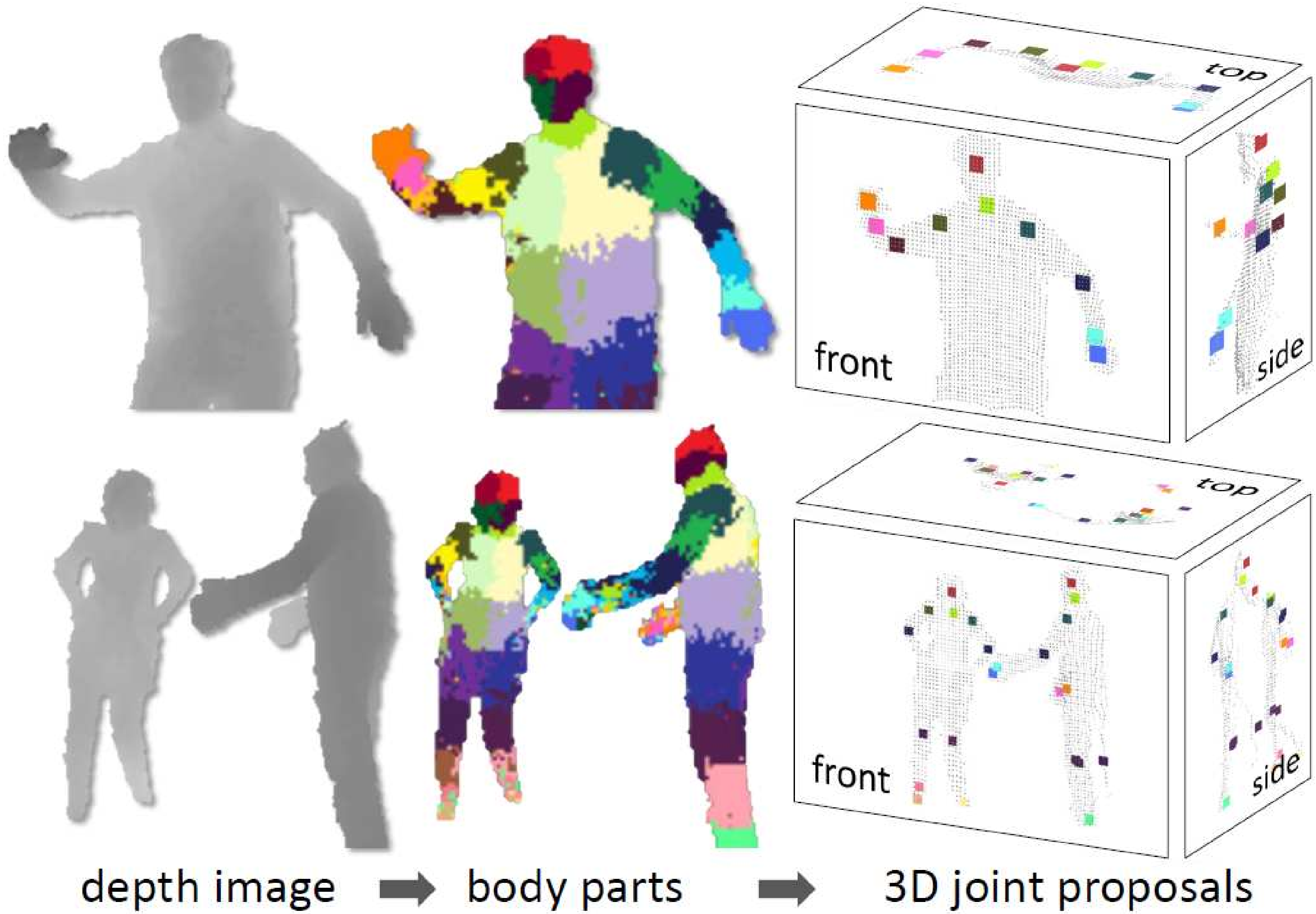








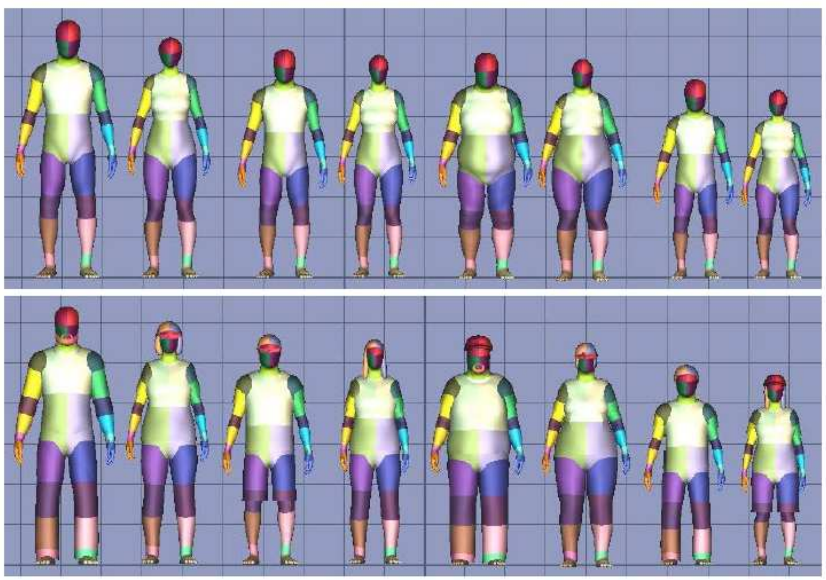
Random Forests and the Kinect

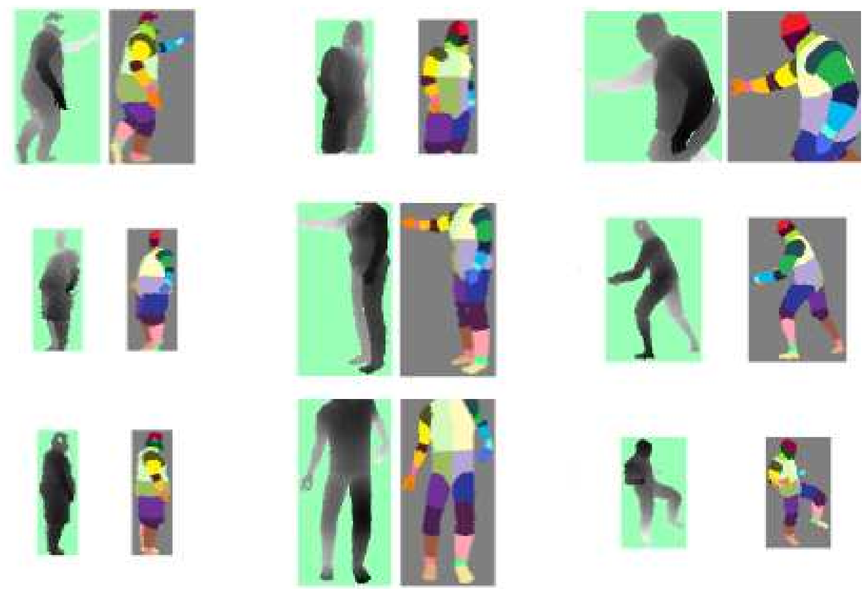


[Jamie Shotton et al 2011]

Random Forests and the Kinect

Lesson 1: Use computer graphics to generate plenty of data.

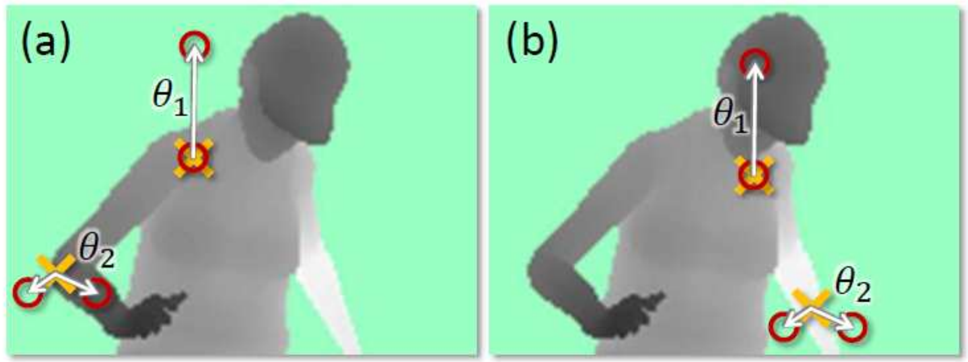




[Jamie Shotton et al 2011]

Random Forests and the Kinect

Lesson 2: Use simple depth features within random forests algorithm.



❑For each pixel x, compute the feature:

$$f_\theta(I,\mathbf{x}) = d_I\left(\mathbf{x} + \frac{\mathbf{u}}{d_I(\mathbf{x})}\right) - d_I\left(\mathbf{x} + \frac{\mathbf{v}}{d_I(\mathbf{x})}\right)$$

$d_I(x)$ is the depth at pixel x in image I

Parameters $\theta = (u; v)$ describe offsets u and v.

❑The normalization of the offsets ensures the features are depth invariant: At a given point on the body, a fixed world space offset will result whether the pixel is close or far from the camera.

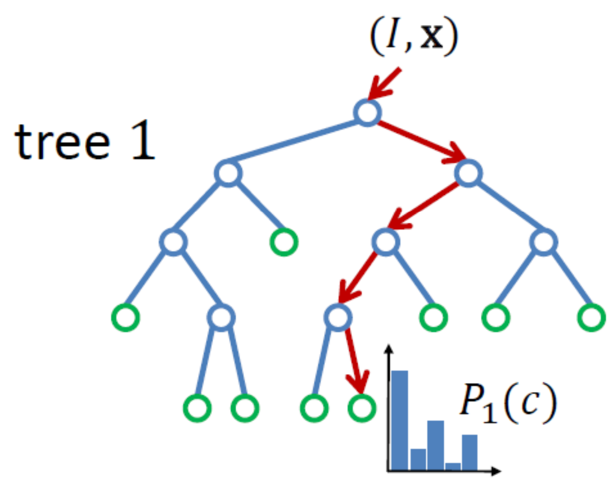
[Jamie Shotton et al 2011]

Tree algorithm

1. Randomly propose a set of splitting candidates $\phi = (\theta, \tau)$ (feature parameters θ and thresholds τ).

2. Partition the set of examples $Q = \{(I, \mathbf{x})\}$ into left and right subsets by each ϕ:

$$
\begin{aligned}
Q_{\mathrm{l}}(\phi) &= \{ (I, \mathbf{x}) \mid f_\theta(I, \mathbf{x}) < \tau \} \\
Q_{\mathrm{r}}(\phi) &= Q \setminus Q_{\mathrm{l}}(\phi)
\end{aligned}
$$



3. Compute the ϕ giving the largest gain in information:

$$
\begin{aligned}
\phi^\star &= \underset{\phi}{\operatorname{argmax}}\, G(\phi) \\
G(\phi) &= H(Q) - \sum_{s \in \{\mathrm{l}, \mathrm{r}\}} \frac{|Q_s(\phi)|}{|Q|} H(Q_s(\phi))
\end{aligned}
$$

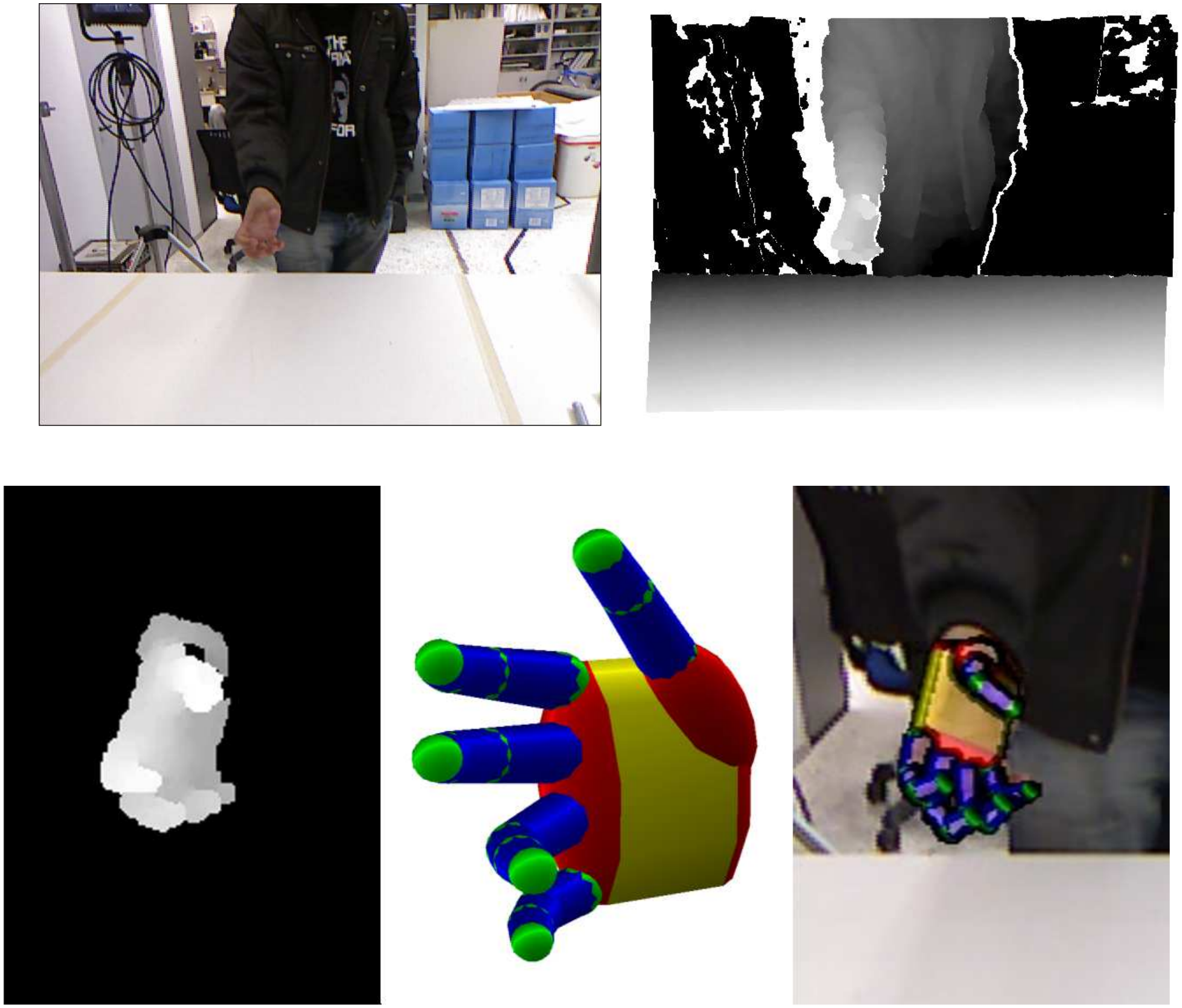
4. If the largest gain $G(\phi^\star)$ is sufficient, and the depth in the tree is below a maximum, then recurse for left and right subsets $Q_{\mathrm{l}}(\phi^\star)$ and $Q_{\mathrm{r}}(\phi^\star)$.

[Jamie Shotton et al 2011]

Performance on train and test data

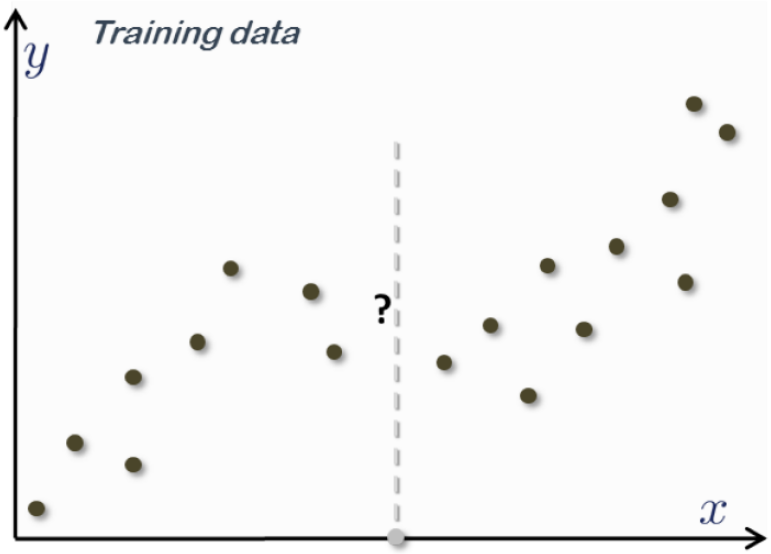
[Jamie Shotton et al 2011]

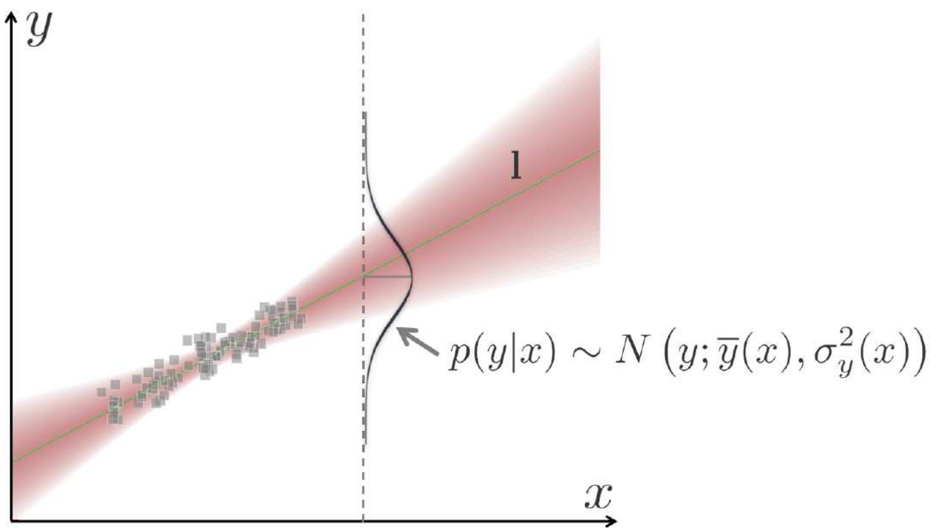
Applications: Interfaces

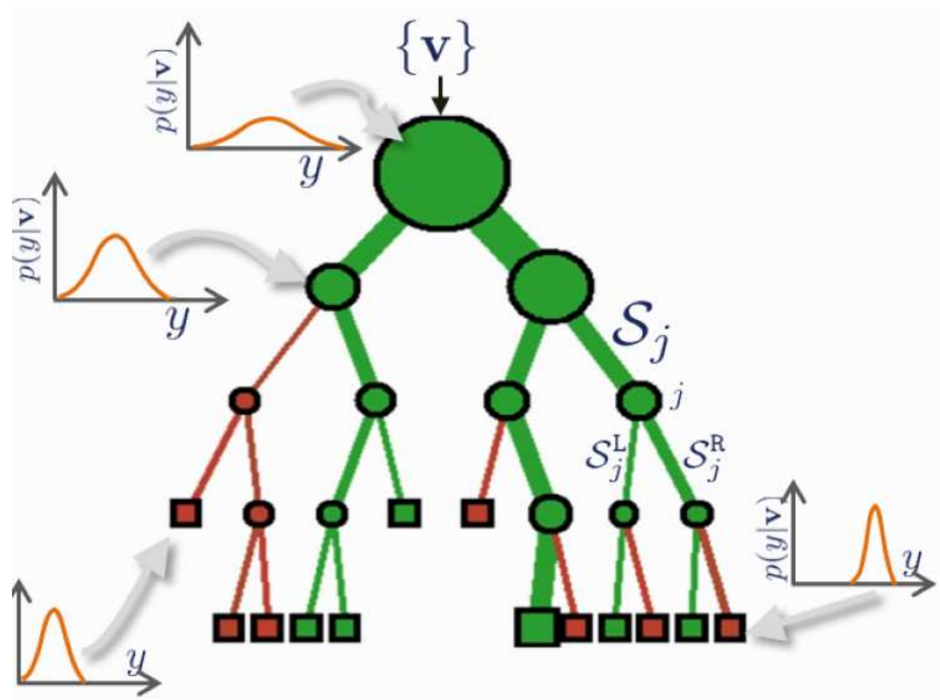


[Iason Oikonomidis et al 2011]

Trees for regression

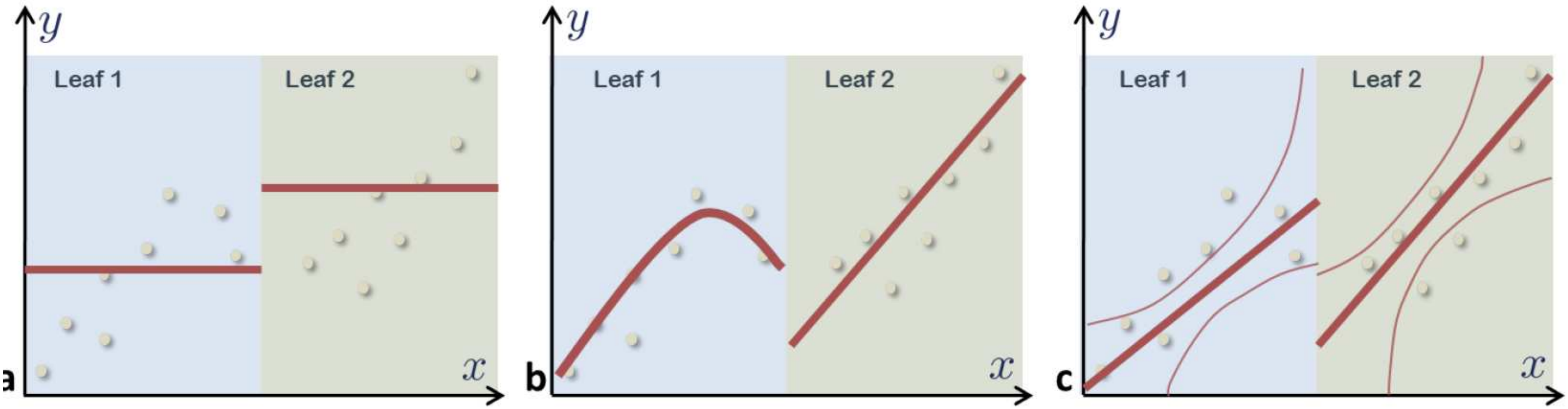






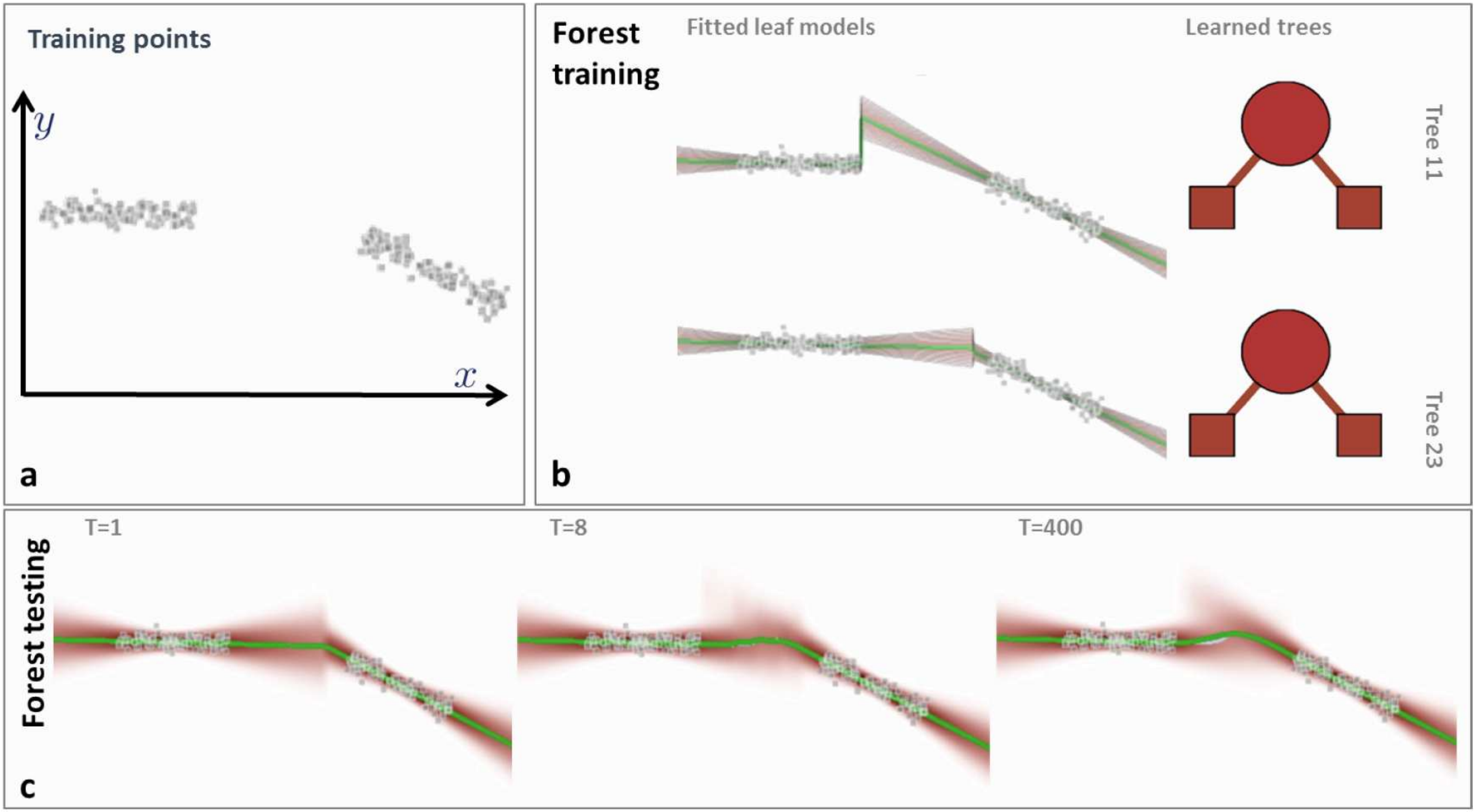
[Criminisi et al, 2011]

Regression trees



[Criminisi et al, 2011]

Regression forests



[Criminisi et al, 2011]