CS 2750 Machine Learning
Lecture 24

I. Decision trees
II. Ensamble methods: Mixtures of experts

Milos Hauskrecht
milos@cs.pitt.edu
5329 Sennott Square

Schedule

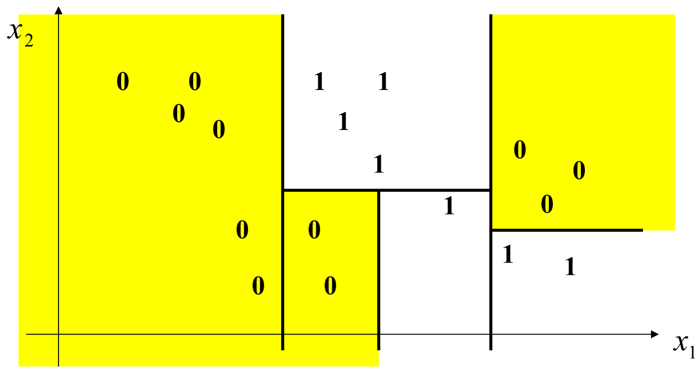
Exam:
- April 18, 2007

Term projects & project presentations:
- April 25, 2007
- At 1:00-4:00pm in SNSQ 5313

No class:
- on April 23, 2007

Decision trees

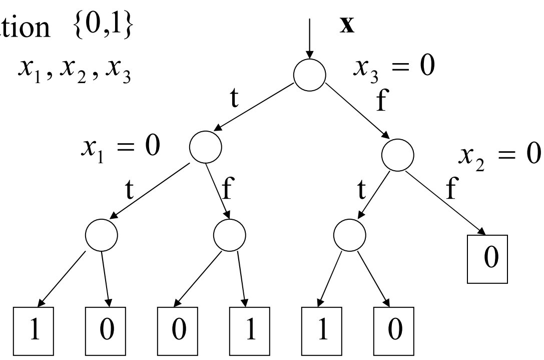
- An alternative approach to classification:
  - **Partition the input space to regions**
  - **Classify independently in every region**



Decision trees

- The partitioning idea is used in the **decision tree model**:
  - Split the space recursively according to inputs in $\mathbf{x}$
  - Classify (assign class label) at the bottom of the tree

Example:
Binary classification $\{0,1\}$
Binary attributes x_1, x_2, x_3



Decision trees

How to construct the decision tree?

- **Top-bottom algorithm**:
  - Find the best split condition (quantified based on the impurity measure) on the training set
  - Stops when no improvement possible
- **Impurity measure**:
  - Measures how well are the two classes separated
  - Ideally we would like to separate all 0s and 1
- Splits of **finite vs. continuous value attributes**

  Continuous value attributes conditions: $x_3 \leq 0.5$

Impurity measure

Let $|D|$ - Total number of data entries

$|D_i|$ - Number of data entries classified as *i*

$p_i = \frac{|D_i|}{|D|}$ - ratio of instances classified as *i*

- **Impurity measure** defines how well are the classes in the training dataset separated
- In general the impurity measure should satisfy:
  - Largest when data are split evenly to classes

    $$p_i = \frac{1}{\text{number of classes}}$$

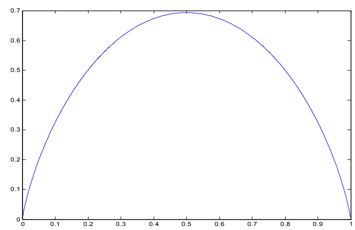
  - Should be 0 when all data belong to the same class

Impurity measures

- There are various impurity measures used in the literature
  - **Entropy based measure (Quinlan, C4.5)**

$$I(D) = Entropy\ (D) = -\sum_{i=1}^{k} p_i \log\ p_i$$

Example for k=2



  - **Gini measure (Breiman, CART)**

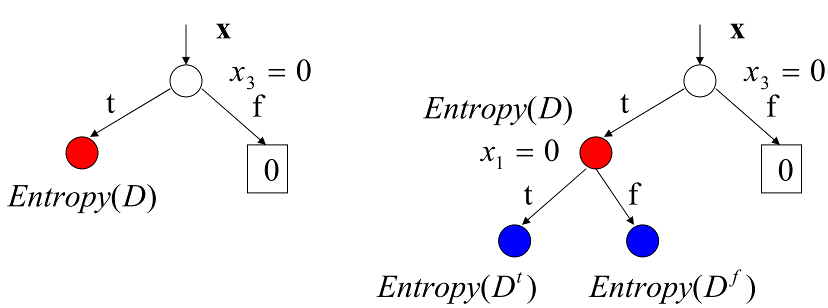
$$I(D) = Gini\ (D) = 1 - \sum_{i=1}^{k} p_i^{\ 2}$$

Decision-tree building

- **Gain due to split** – expected reduction in the impurity measure (entropy example)

$$Gain\ (D, A) = Entropy\ (D) - \sum_{v \in Values\ (A)} \frac{|\ D^v\ |}{|\ D\ |} Entropy\ (D^v)$$

$|\ D^v\ |$ - a partition of D with the value of attribute A = v



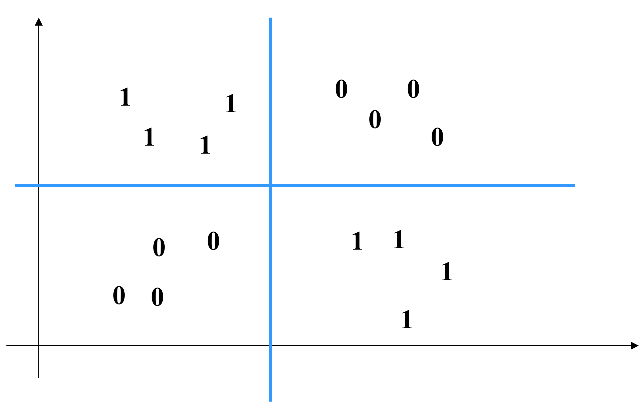
Decision tree learning

- **Greedy learning algorithm**:
  Repeat until no or small improvement in the purity
  - Find the attribute with the highest gain
  - Add the attribute to the tree and split the set accordingly

- Builds the tree in the top-down fashion
  - Gradually expands the leaves of the partially built tree
- The method is greedy
  - It looks at a single attribute and gain in each step
  - May fail when the combination of attributes is needed to improve the purity (parity functions)

Decision tree learning

- **Limitations of greedy methods**
  Cases in which a combination of two or more attributes improves the impurity



Decision tree learning

By reducing the impurity measure we can grow **very large trees**

Problem: Overfitting

- We may split and classify very well the training set, but we may do worse in terms of the generalization error

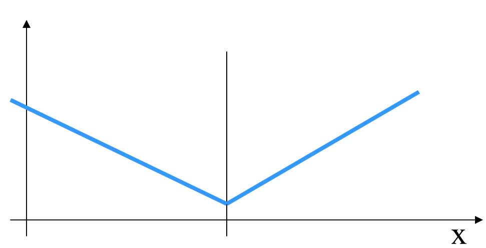
Solutions to the overfitting problem:

- **Solution 1.**
  - Prune branches of the tree built in the first phase
  - Use internal validation set to test for the overfit
- **Solution 2**.
  - Test for the overfit in the tree building phase
  - Stop building the tree when performance on the validation set deteriorates

Mixture of experts model

- **Ensamble methods:**
  - Use a combination of simpler learners to improve predictions
- **Mixture of expert model:**
  - Different input regions covered with different learners
  - A “soft” switching between learners
- **Mixture of experts**

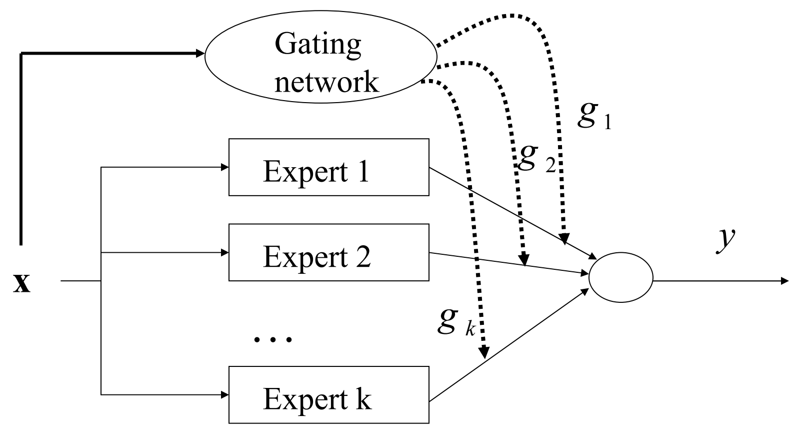
  Expert = learner



Mixture of experts model

- **Gating network** : decides what expert to use

  $g_1, g_2, \dots g_k$ - gating functions



Learning mixture of experts

- **Learning consists of two tasks:**
  - Learn the parameters of individual expert networks
  - Learn the parameters of the gating network
    - Decides where to make a split
- **Assume:** gating functions give probabilities

$$0 \leq g_1(\mathbf{x}), g_2(\mathbf{x}), \dots g_k(\mathbf{x}) \leq 1 \qquad \sum_{u=1}^{k} g_u(\mathbf{x}) = 1$$

- Based on the probability we partition the space
  - partitions belongs to different experts
- How to model the gating network?
  - **A multiway classifier model:**
    - **softmax model**
    - **a generative classifier model**

Learning mixture of experts

- Assume we have a **set of linear experts**

$$\mu_i = \boldsymbol{\theta}_i^T \mathbf{x} \qquad \text{(Note: bias terms are hidden in x)}$$

- Assume a **softmax gating network**

$$g_i(\mathbf{x}) = \frac{\exp(\boldsymbol{\eta}_i^T \mathbf{x})}{\sum_{u=1}^{k} \exp(\boldsymbol{\eta}_u^T \mathbf{x})} \approx p(\omega_i \mid \mathbf{x}, \boldsymbol{\eta})$$

- Likelihood of y (assumed that errors for different experts are normally distributed with the same variance)

$$P(y \mid \mathbf{x}, \Theta, \boldsymbol{\eta}) = \sum_{i=1}^{k} P(\omega_i \mid \mathbf{x}, \boldsymbol{\eta}) p(y \mid \mathbf{x}, \omega_i, \boldsymbol{\Theta})$$

$$= \sum_{i=1}^{k} \left[\frac{\exp(\boldsymbol{\eta}_i^T \mathbf{x})}{\sum_{j=1}^{k} \exp(\boldsymbol{\eta}_j^T \mathbf{x})} \right] \left[\frac{1}{\sqrt{2\pi}\,\sigma} \exp\left(-\frac{\|y - \mu_i\|^2}{2\sigma^2} \right) \right]$$

Learning mixture of experts

Gradient learning.

On-line update rule for parameters $\boldsymbol{\theta}_i$ **of expert** i

– If we know the expert that is responsible for **x**

$$\theta_{ij} \leftarrow \theta_{ij} + \alpha_{ij}(y - \mu_i) x_j$$

– If we do not know the expert

$$\theta_{ij} \leftarrow \theta_{ij} + \alpha_{ij} h_i (y - \mu_i) x_j$$

h_i - **responsibility of the** i**th expert** = a kind of posterior

$$h_i(\mathbf{x}, y) = \frac{g_i(\mathbf{x}) p(y \mid \mathbf{x}, \omega_i, \boldsymbol{\theta})}{\sum_{u=1}^{k} g_u(\mathbf{x}) p(y \mid \mathbf{x}, \omega_u, \boldsymbol{\theta})} = \frac{g_i(\mathbf{x}) \exp\left(-1/2\|y - \mu_i\|^2\right)}{\sum_{u=1}^{k} g_u(\mathbf{x}) \exp\left(-1/2\|y - \mu_u\|^2\right)}$$

$g_i(\mathbf{x})$ - a prior $\qquad$ $\exp(\ldots)$ - a likelihood

Learning mixtures of experts

Gradient methods

- **On-line learning of gating network parameters $\boldsymbol{\eta}_i$**

$$\eta_{ij} \leftarrow \eta_{ij} + \beta_{ij}(h_i(\mathbf{x}, y) - g_i(\mathbf{x}))x_j$$

- The learning with conditioned mixtures can be extended to learning of parameters of an **arbitrary expert network**
  – e.g. logistic regression, multilayer neural network

$$\theta_{ij} \leftarrow \theta_{ij} + \beta_{ij} \frac{\partial l}{\partial \theta_{ij}}$$

$$\frac{\partial l}{\partial \theta_{ij}} = \frac{\partial l}{\partial \mu_i} \frac{\partial \mu_i}{\partial \theta_{ij}} = h_i \frac{\partial \mu_i}{\partial \theta_{ij}}$$

Learning mixture of experts

EM algorithm offers an alternative way to learn the mixture

Algorithm:

Initialize parameters Θ

Repeat

Set $\Theta' = \Theta$

1. **Expectation step**

$$Q(\Theta \mid \Theta') = E_{H|\mathbf{X},\mathbf{Y},\mathbf{\Theta}'} \log P(\mathbf{H}, \mathbf{Y} \mid \mathbf{X}, \Theta, \xi)$$

2. **Maximization step**

$$\Theta = \arg\max_{\Theta} Q(\Theta \mid \Theta')$$

until no or small improvement in $Q(\Theta \mid \Theta')$

– **Hidden variables are identities of expert networks responsible for (x,y) data points**