Data-driven Intelligent Systems

Lecture 7
Decision Trees and Classification

KNOWLEDGE
TECHNOLOGY

http://www.informatik.uni-hamburg.de/WTM/

From Data to Knowledge

Medical Data by Dr. X, Tokyo Med. & Dent. Univ., 38 attributes:

10, M, 0, 10, 10, 0, 0, 0, SUBACUTE, 37, 2, 1, 0,15, -, -, 6000, 2, 0, abnormal, abnormal,
-, 2852, 2148, 712, 97, 49, F, -, multiple, , 2137, negative, n, n, ABSCESS, ***VIRUS***

12, M, 0, 5, 5, 0, 0, 0, ACUTE, 38.5, 2, 1, 0,15, -, -, 10700, 4, 0, normal, abnormal,
+, 1080, 680, 400, 71, 59, F, -, ABPC+CZX, , 70, negative, n, n, n, BACTERIA, ***BACTERIA***

15, M, 0, 3, 2, 3, 0, 0, ACUTE, 39.3, 3, 1, 0,15, -, -, 6000, 0,0, normal, abnormal,
+, 1124, 622, 502, 47, 63, F, -, FMOX+AMK, , 48, negative, n, n, n, BACTE(E), ***BACTERIA***

16, M, 0, 32, 32, 0, 0, 0, SUBACUTE, 38, 2, 0, 0, 15, -, +, 12600, 4, 0, abnormal, abnormal,
+, 41, 39, 2, 44, 57, F, -, ABPC+CZX, ?, ?, negative, ?, n, n, ABSCESS, ***VIRUS***

Numerical attribute   Categorical attribute   Missing values   Class labels

IF cell_poly <= 220 AND Risk = n
AND Loc_dat = + AND Nausea > 15
THEN Prediction = VIRUS [87,5%]

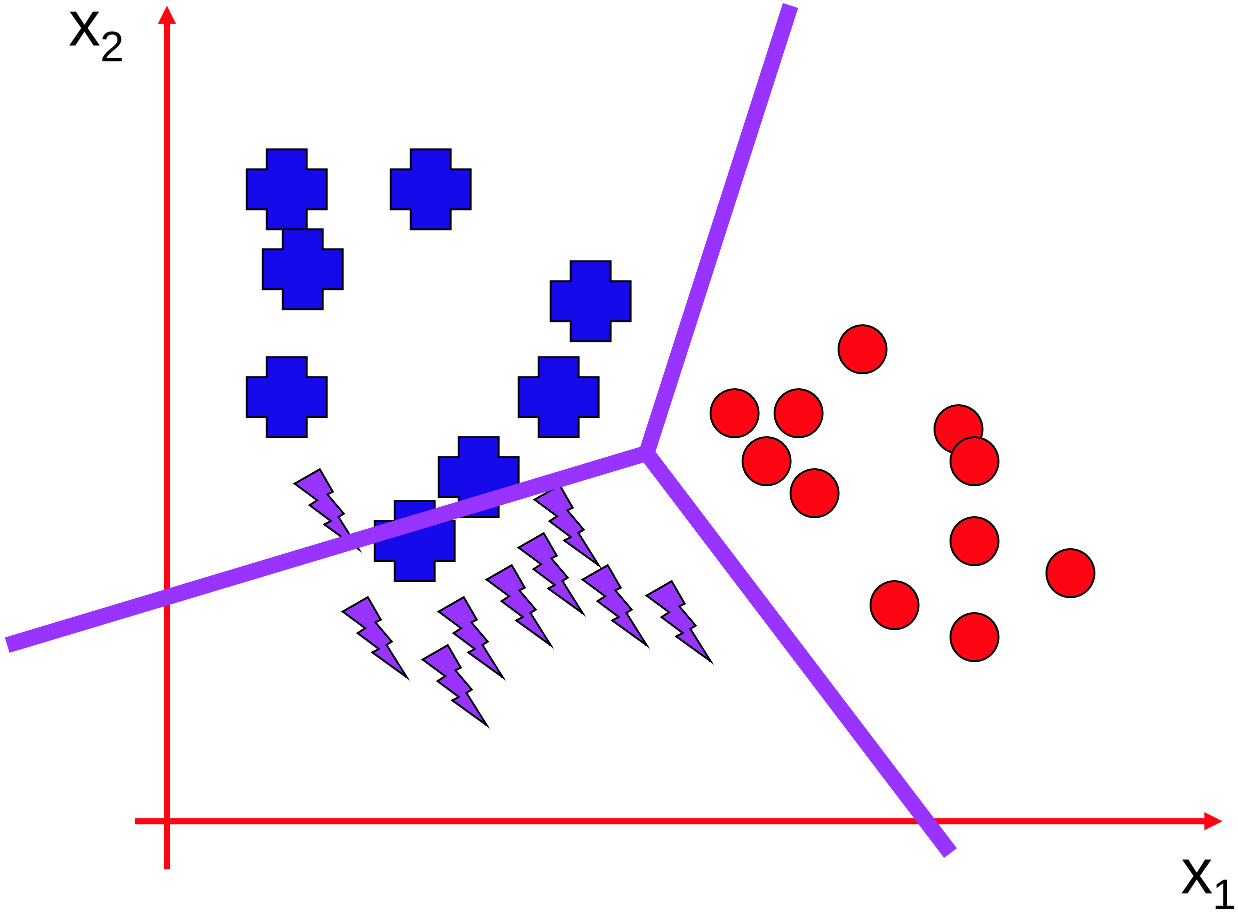
Predictive accuracy

Overview

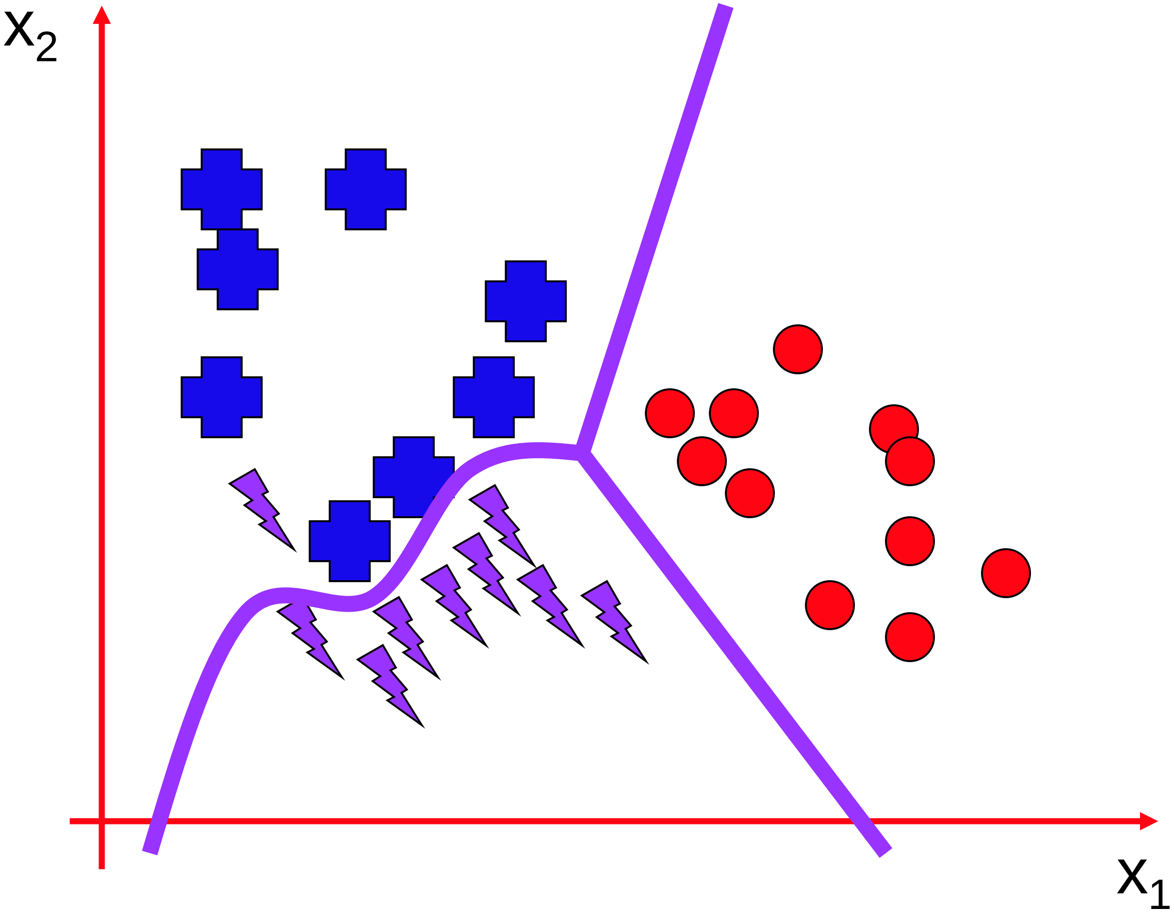
- Decision tree induction
- Criteria for attribute split
  - Information Gain
  - Gini Impurity
- Advanced decision trees
  - Continuous attributes
  - Gain ratio
  - Missing values
  - Pruning
  - Rule extraction
- Limitations of decision trees

next week

Decision Boundaries



Decision Boundaries



History of Decision Trees

- 1966: Hunt, colleagues in psychology used full search decision tree methods to model human concept learning
- 1977: Breiman, Friedman, colleagues in statistics develop simultaneous Classification And Regression Trees (CART)
- 1986: Quinlan's landmark paper on ID3
- Late 1980s: Various improvements, i.e: coping with noise, continuous attributes, missing data, non-axis-parallel DTs
- 1993: Quinlan's updated algorithm, C4.5
- Towards 2000: Quinlan: More pruning, overfitting control heuristics (C5.0, etc.); combining DTs; incremental learning

Supervised vs. Unsupervised Learning (continuous outputs)

- **Supervised Learning: Regress*ion***
  - The training data (observations, measurements, etc.) are accompanied by ***continuous output values***
  - For new data where output values are missing, “predict” the most likely output values based on the training
- **Unsupervised Learning: *general case***
  - Given a set of measurements, observations, etc. of which the class labels are unknown
  - Aim: represent the data in another form
    - E.g. compressed via PCA, …

Supervised vs. Unsupervised Learning (discrete outputs)

today

- **Supervised Learning: *Classification***
  - The training data (observations, measurements, etc.) are accompanied by ***categorical class labels*** (discrete or nominal)
  - New data is classified based on the training
- **Unsupervised Learning: *Clustering***
  - Given a set of measurements, observations, etc. of which the class labels are unknown
  - Aim: establish the existence of clusters in the data

Decision Trees

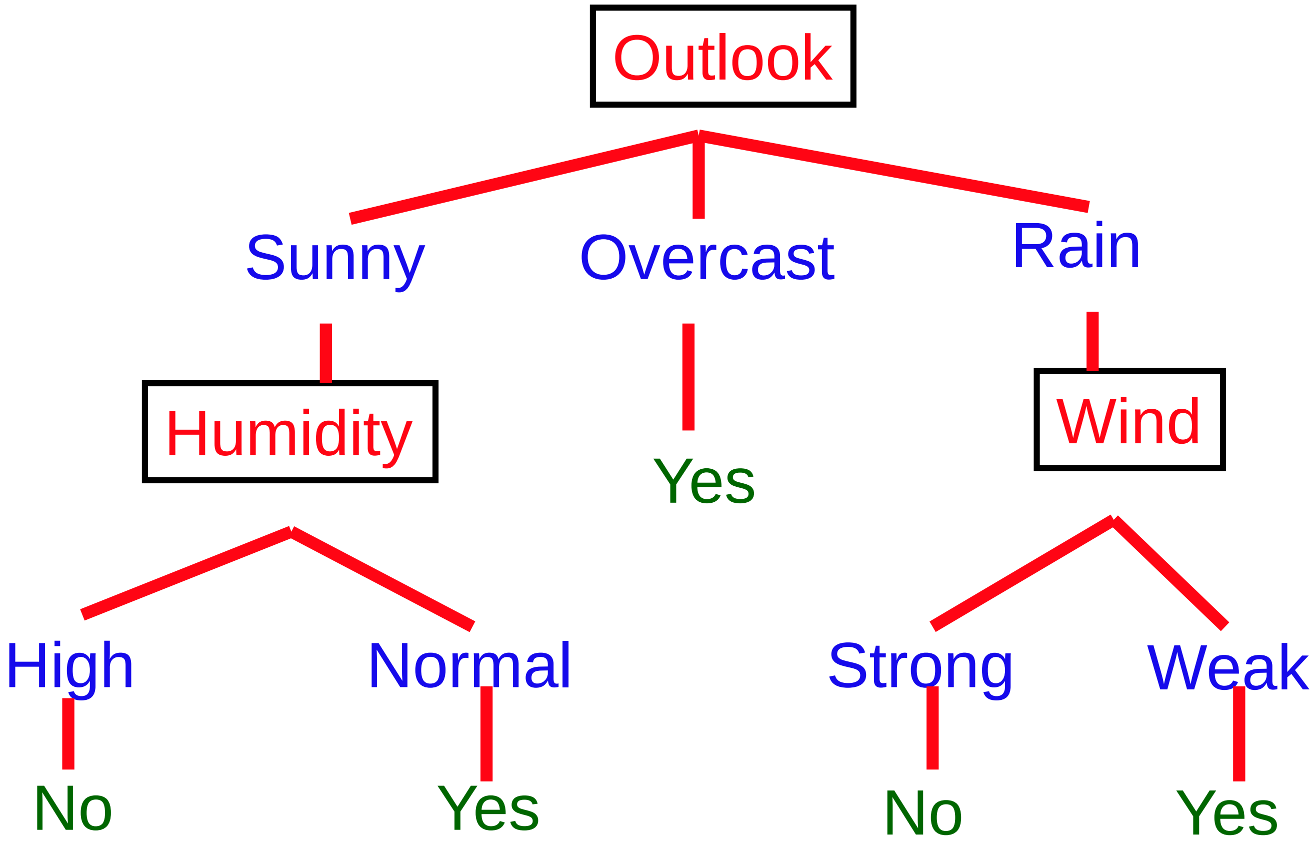
- Split ***classification*** into a series of choices about features in turn

- Lay them out in a tree

- Progress down the tree to the leaves

Example: Anyone for Tennis?

Decide whether to play tennis based on the weather



Rules and Decision Trees

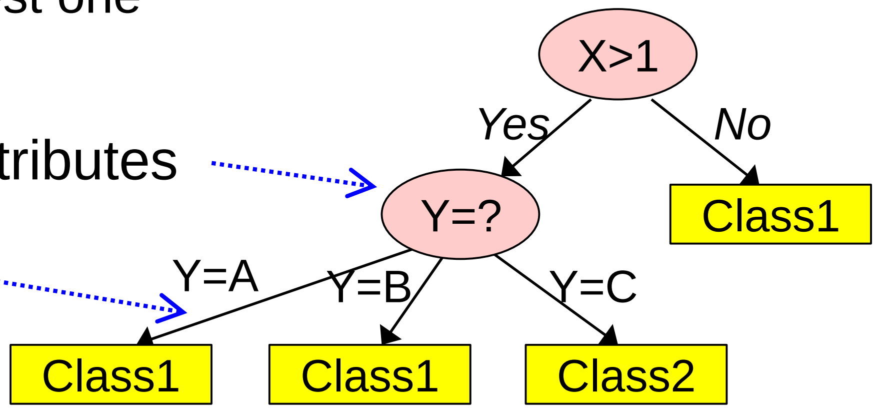
- Tree can be turned into a set of rules:
  - if (outlook = sunny & humidity = normal) | (outlook = overcast)
    | (outlook = rain & wind = weak)

    then

    play tennis
- How do we generate the trees?
  - Need to choose features / attributes
  - Need to choose order of features / attributes

Decision Trees

- Efficient method for producing ***classifiers*** from data
  - ***Supervised learning*** methods that construct decision trees from a set of input-output samples
  - Guarantees that a simple tree is found
    - but not necessarily the simplest one
- Consists of
  - ***Nodes*** that are tests on the attributes
  - Outgoing ***branches*** of a node correspond to all the possible outcomes of the test at the node
  - ***Leaves*** that are sets of samples belonging to the same class



Applications of Decision Trees

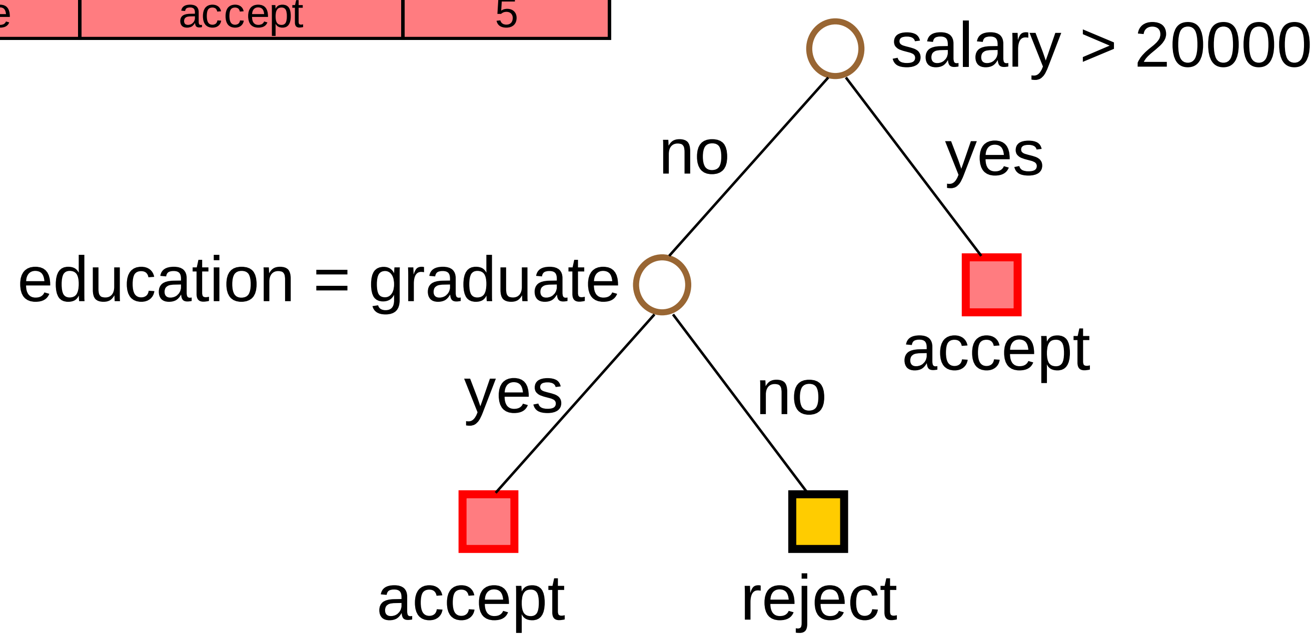
- Astronomy: star/galaxy classification
- Aircraft: uncover flaws in the manufacturing process
- Medical: classify if a tumor is cancerous,
  detect microcalcifications in mammography
- Vision: recognize 3D objects in high-level vision
- Pharmacology: classification in drug analysis
- Physics: detection of particles
- Plant disease: assess the hazard of mortality to pine trees
- Power systems: security assessment; stability prediction
- Medical text classification
- …

http://www.cbcb.umd.edu/~salzberg/docs/murthy_thesis/survey/node32.html

Example of Decision Tree for Credit Approval

Credit Analysis

salary	education	label	#
10000	high school	reject	1
40000	under graduate	accept	2
18000	under graduate	reject	3
75000	graduate	accept	4
15000	graduate	accept	5



Decision Tree for Classification

- Given:
  - Database of ***samples***, each assigned a ***class label***.

- Task: Develop a ***model***/profile for each class:
  - **Example profile** (good credit):

    (salary > 20k) or (salary <= 20k and education = graduate) => Credit = Good (approved)

Overview

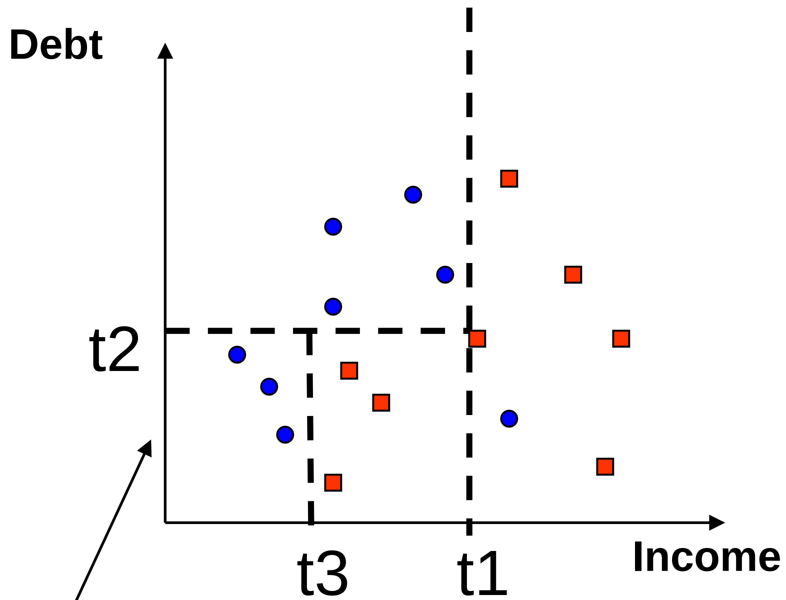
- Decision tree induction
- Criteria for attribute split
  - Information Gain
  - Gini Impurity

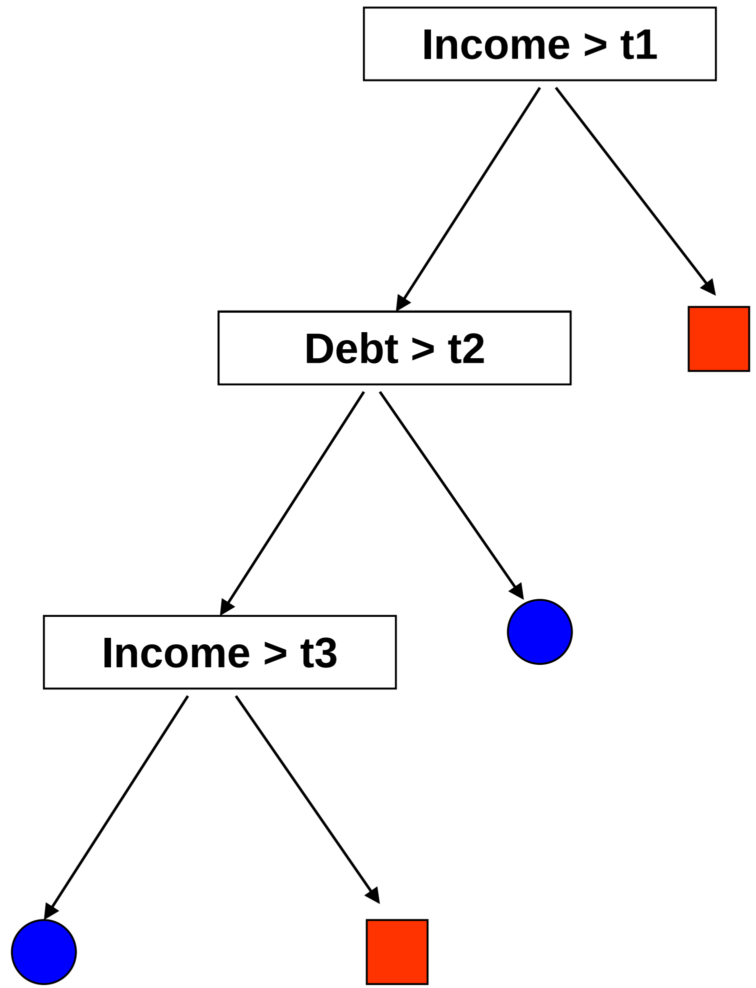
Classification by Decision Tree Induction

- Decision tree ***generation*** consists of two phases:
  1. Tree ***construction***:
     - At start, all the training examples are at the root.
     - Partition the examples recursively based on selected attributes.
  2. Tree ***pruning***:
     - Identify and remove branches that reflect noise or outliers.

- Decision tree ***use***: Classifying an unknown sample
  - Test the attribute values of the sample against the decision tree

Decision Tree: Example





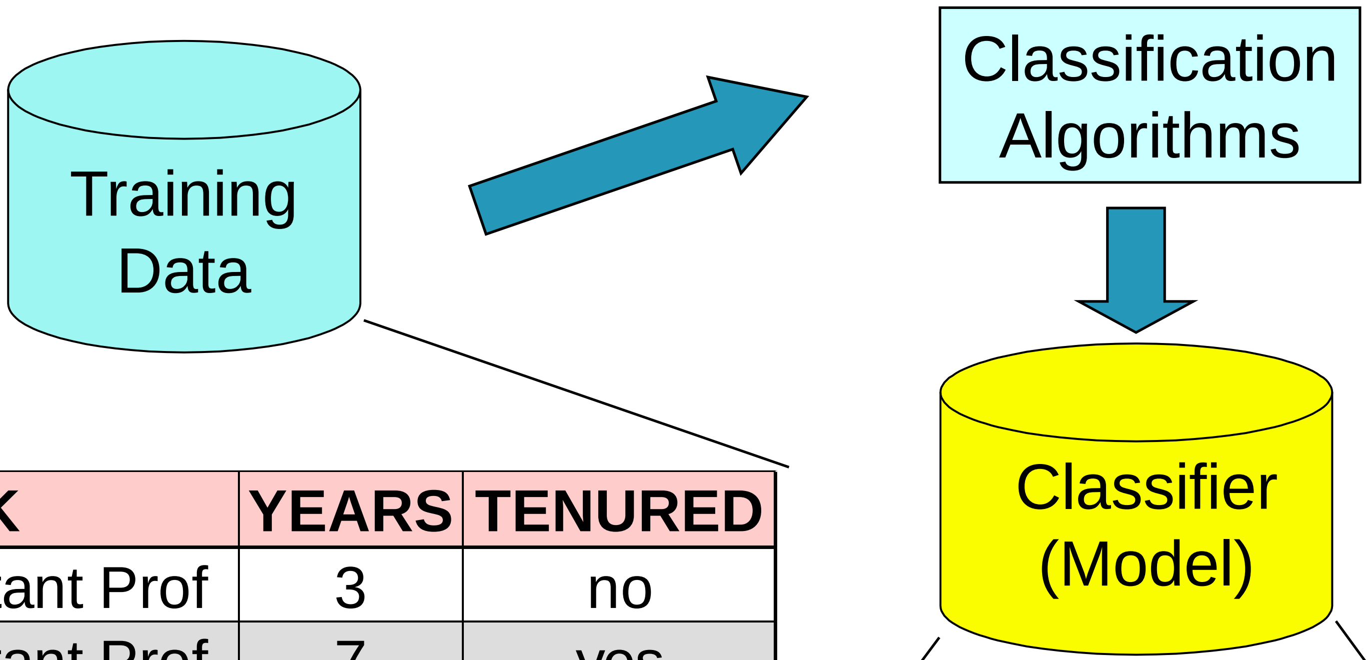
Note: tree boundaries are piecewise linear and axis-parallel

Are all correctly classified?

Classification: Model Construction & Usage

- ***Model construction***: describing a set of predetermined classes
  - The set of tuples used for model construction is the ***training set***
  - Each tuple/sample is assumed to belong to a predefined class, as determined by the ***class label attribute***
  - The ***model*** is represented as decision tree or as classification rules (cf. neural network: as mathematical formulae)
- ***Model usage***: for classifying future or unknown objects
  - Evaluate the model with a ***test set*** (independent of training set)
    - Compare ***known labels*** of test samples with the classification result of the model
    - E.g., ***accuracy*** rate is the percentage of test set samples that are correctly classified by the model
  - If the accuracy is acceptable, use the model to ***classify data*** tuples whose class labels are not known

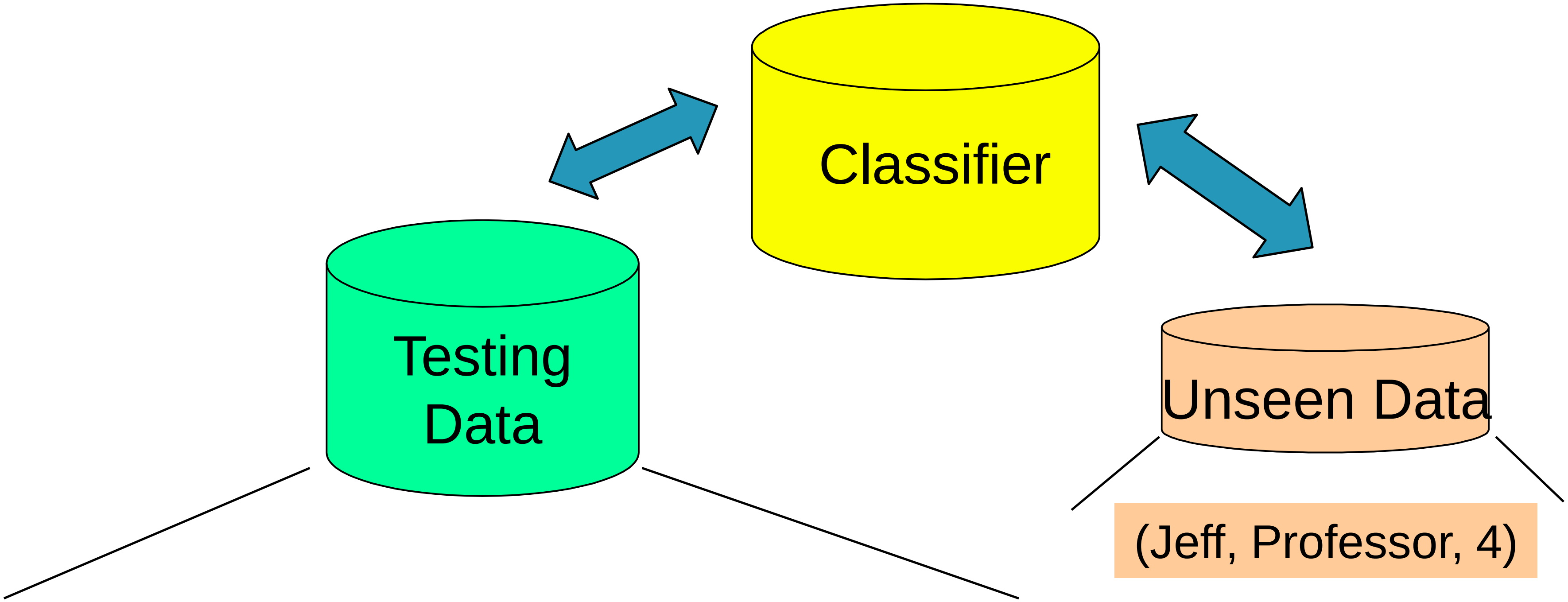
Process (1): Model Construction



NAME	RANK	YEARS	TENURED
Mike	Assistant Prof	3	no
Mary	Assistant Prof	7	yes
Bill	Professor	2	yes
Jim	Associate Prof	7	yes
Dave	Assistant Prof	6	no
Anne	Associate Prof	3	no

IF rank = ‘professor’
OR years > 6
THEN tenured = ‘yes’

Process (2): Using the Model



NAME	RANK	YEARS	TENURED
Tom	Assistant Prof	2	no ✓
Merlisa	Associate Prof	7	yes ✓
George	Professor	5	yes ✓
Joseph	Assistant Prof	7	yes ✓

Tenured?

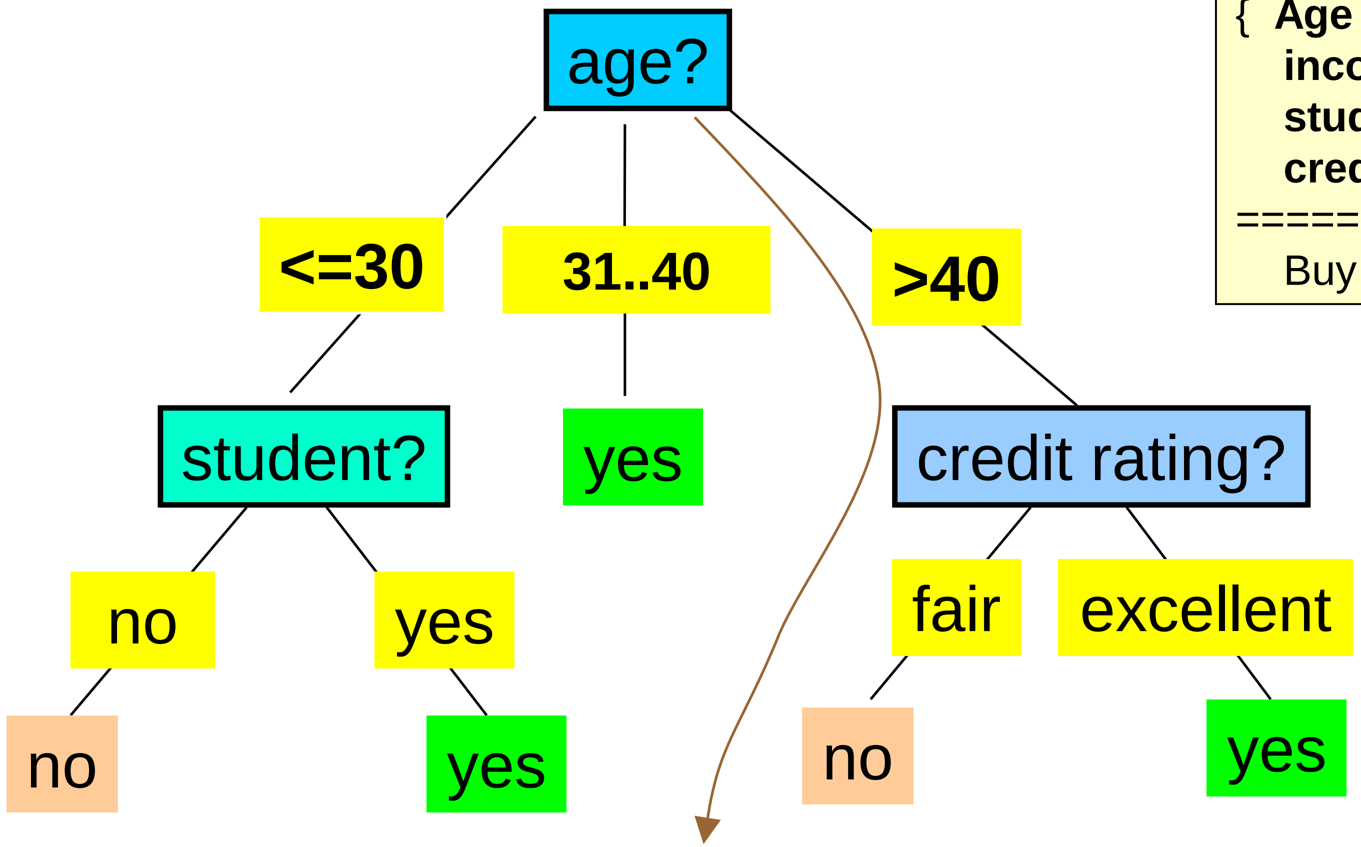
Yes!

Decision Tree Induction: Training Dataset

This follows an example of Quinlan's ID3

age	income	student	credit_rating	buys_computer
<=30	high	no	fair	no
<=30	high	no	excellent	no
31…40	high	no	fair	yes
>40	medium	no	fair	yes
>40	low	yes	fair	yes
>40	low	yes	excellent	no
31…40	low	yes	excellent	yes
<=30	medium	no	fair	no
<=30	low	yes	fair	yes
>40	medium	yes	fair	yes
<=30	medium	yes	excellent	yes
31…40	medium	no	excellent	yes
31…40	high	yes	fair	yes
>40	medium	no	excellent	no

Output: A Decision Tree for “*buys_computer*”



Classify new sample:

{ **Age = 45** and
income = low and
student = no and
credit = fair}
=================
Buy computer = **?**

Answer (class):
NO

Decision Tree

- Requirements for a Decision Tree algorithm:
  1. Consistent attribute-value description for all data
  2. Predefined classes
  3. Discrete classes
  4. Sufficient data
  5. “Logical” classification (not weighted decisions)

- Pros
  - ***Fast*** execution time
  - Generated trees (rules) are ***easy to interpret*** by humans
  - ***Scale well*** for large data sets
  - Can handle high-dim. data

- Cons
  - Cannot capture ***correlations*** among attributes
  - Consider only ***axis-parallel*** cuts

Many Decision Tree Algorithms

- Classifiers from machine learning and statistical community:
  - CART (as an advance in applied statistics)
  - ID3
  - C4.5 [Quinlan 93] $\rightarrow$ C5.0
- Classifiers for large databases:
  - SLIQ, SPRINT
  - SONAR
  - Rainforest
- Aspects are quality of the tree, scalability, and memory use.

Overview

- Decision tree induction
- Criteria for attribute split
  - Information Gain
  - Gini Impurity

Algorithm for Decision Tree Induction

- Basic algorithm (a greedy algorithm)
  - Tree is constructed in a ***top-down recursive divide-and-conquer manner***
  - At start, all the training examples are at the root
  - Attributes are categorical (if continuous-valued, they are discretized in advance)
  - Examples are partitioned recursively based on selected attributes
  - Test attributes are selected on the basis of a heuristic or statistical measure (e.g., ***information gain***)
- Conditions for **stopping partitioning**
  - All samples for a given node belong to the same class
  - There are no remaining attributes for further partitioning
  - Heuristics: maximum depth of tree; minimum # data at leaf, etc.
  - ***Majority voting*** is employed for classifying the leaf

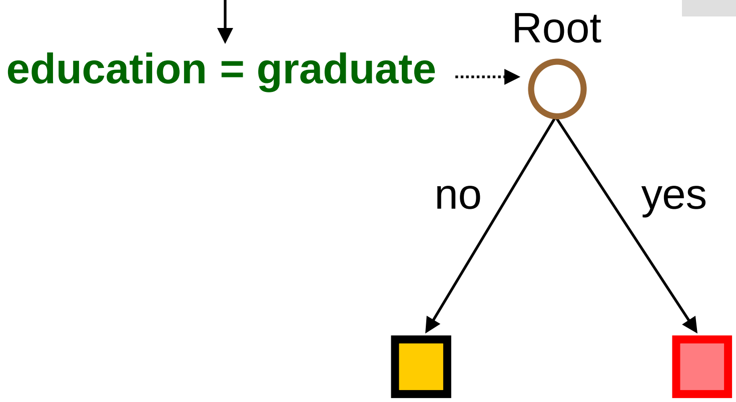
Decision Tree Algorithms: First Splitting

education

high school	reject	1
under graduate	accept	2
under graduate	reject	3
graduate	accept	4
graduate	accept	5

salary

10000	reject	1
15000	accept	5
18000	reject	3
40000	accept	2
75000	accept	4



high-school	10000	reject	1
under-graduate	40000	accept	2
under-graduate	18000	reject	3

graduate	75000	accept	4
graduate	15000	accept	5

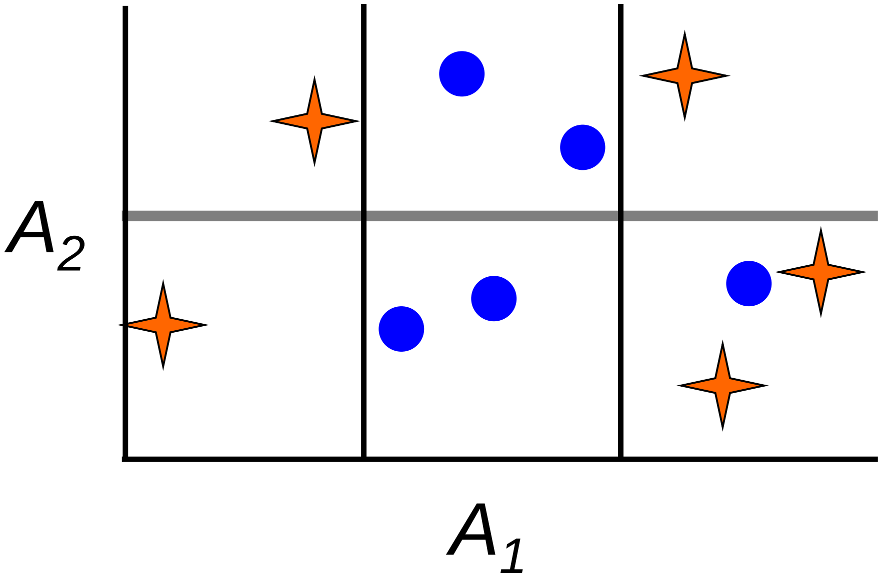
we did not explain how we selected “education” attribute for splitting

Overview

- Decision tree induction
- Criteria for attribute split
    - Information Gain
    - Gini Impurity

Select the Attribute with highest Information Gain

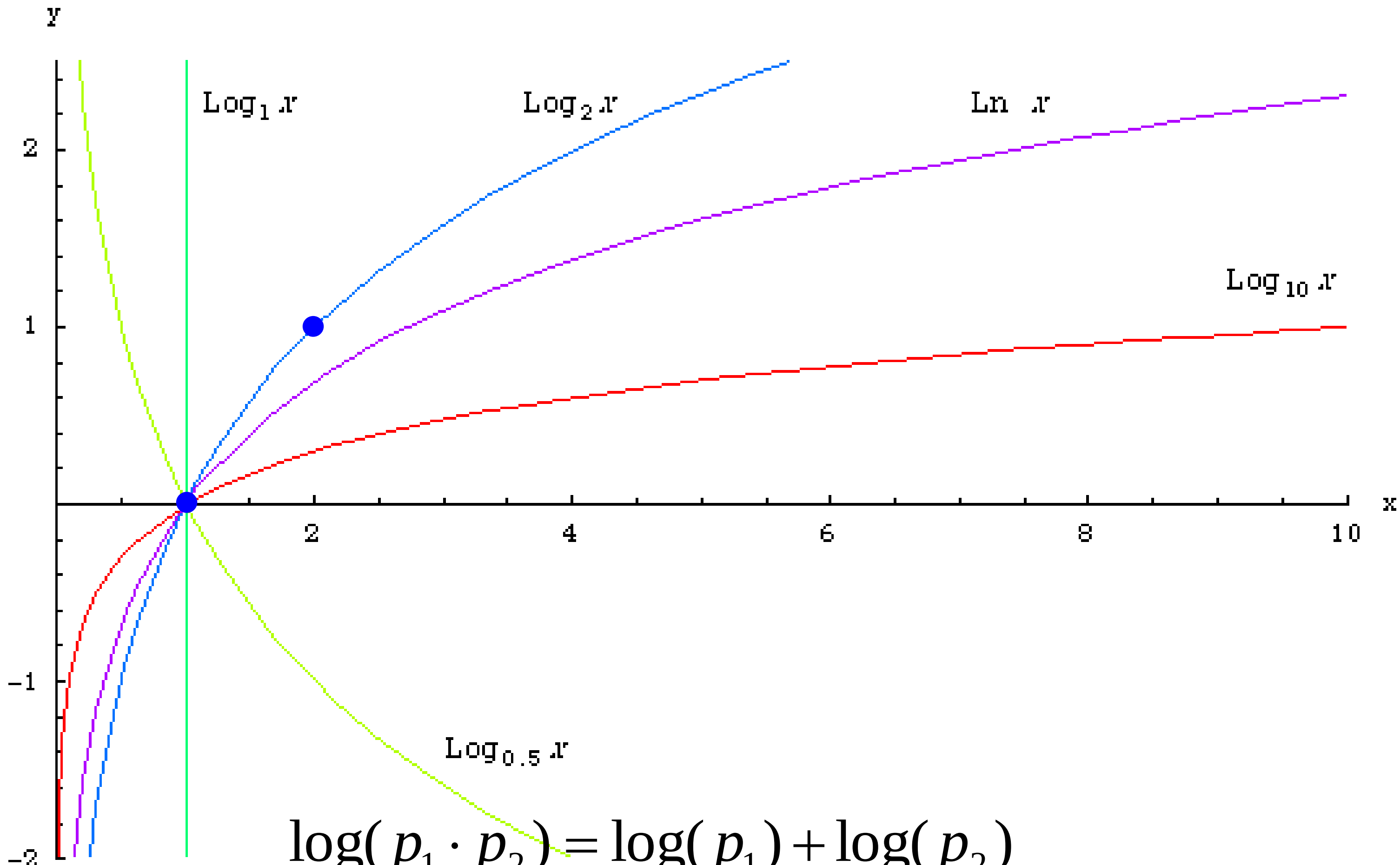
Entropy *H(D)* over all data is large, because both classes are equally frequent



Split over A_2 would not reduce the entropy. This attribute is **not** suited for splitting.

Consider split over attribute A_1: Entropy $H_{A1}(D)$ over the three partitions is low, because classes get separated well.

Reminder…$\log_2 p$



$$\log(p_1 \cdot p_2) = \log(p_1) + \log(p_2)$$

$$\log(\frac{1}{p_1} \cdot \frac{1}{p_2}) = -\log(p_1) - \log(p_2)$$

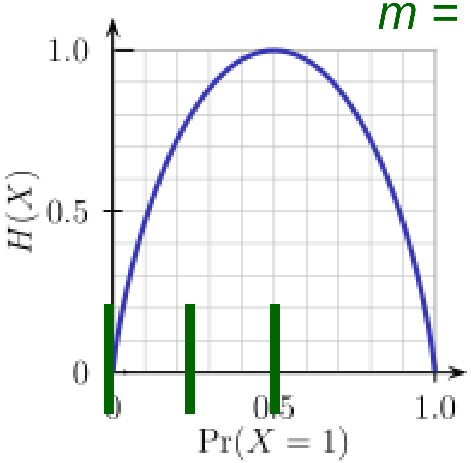
Brief Review of Entropy

- Entropy (Information Theory)
  - ***Measure of uncertainty*** associated with a random variable
    - Higher entropy ⇨ higher uncertainty
    - Lower entropy ⇨ lower uncertainty
  - Calculation: For a discrete random variable Y taking *m* distinct values $\{y_1,\ldots,y_m\}$, where $p_i=P(Y=y_i)$

$$H(Y) = \underbrace{\sum_{i=1}^{m} p_i}_{\textit{weighted average over the classes}} \cdot \log(\underbrace{\frac{1}{p_i}}_{\textit{surprise}}) = -\sum_{i=1}^{m} p_i \cdot \log(p_i)$$

Brief Review of Entropy

for $m = 2$



$$H(Y) = \sum_{i=1}^{m} p_i \cdot \log(\frac{1}{p_i}) = -\sum_{i=1}^{m} p_i \cdot \log(p_i)$$

Examples for m=2 classes:

- p_1=0, p_2=1

$$H(Y) = -0 \cdot \log(0) - 1 \cdot \log(1) = -0 \cdot (-\infty_{small}) - 1 \cdot 0 = 0$$

- p_1=0.25, p_2=0.75

$$H(Y) = -0.25 \cdot \log(0.25) - 0.75 \cdot \log(0.75) = -0.25 \cdot (-2) - 0.75 \cdot (-0.415)$$
$$= 0.811$$

- p_1=0.5, p_2=0.5

$$H(Y) = -0.5 \cdot \log(0.5) - 0.5 \cdot \log(0.5)$$
$$= -0.5 \cdot (-1) - 0.5 \cdot (-1) = 1$$

Select the Attribute with highest Information Gain (ID3/C4.5)

- Let p_i be the probability that an arbitrary tuple in *D* belongs to class C_i, (*m* classes) estimated by $|C_{i,D}|/|D|$
- ***Information*** (entropy) to classify a tuple ***in D***:

$$Info(D) = -\sum_{i=1}^{m} p_i \log_2(p_i)$$

- ***Average information*** needed, after using ***attribute A*** to split *D* into *k* partitions:

$$Info_A(D) = \underbrace{\sum_{j=1}^{k} \frac{|D_j|}{|D|}}_{\text{weighted average over the partitions}} \cdot \underbrace{Info(D_j)}_{\text{entropy in partition}}$$

- ***Information gained*** by branching on attribute *A*:

$$Gain(A) = Info(D) - Info_A(D)$$

Information Gain – Example

- Class “buys_computer =yes” (9x)
- Class “buys_computer =no” (5x)

$$Info(D) = I(9,5) = -\frac{9}{14}\log_2\left(\frac{9}{14}\right) - \frac{5}{14}\log_2\left(\frac{5}{14}\right) = 0.94$$

$$Info_{age}(D) = \boxed{\frac{5}{14}I(2,3)} + \underbrace{\frac{4}{14}I(4,0)}_{0} + \frac{5}{14}I(3,2) = 0.694$$

“age <=30” has 5 out of 14 samples, with 2 “yes” and 3 “no”

age	yes_i	no_i	$I(yes_i, no_i)$
<=30	2	3	0,971
31…40	4	0	0
>40	3	2	0,971

age	income	student	credit_rating	buys_computer
<=30	high	no	fair	no
<=30	high	no	excellent	no
31…40	high	no	fair	yes
>40	medium	no	fair	yes
>40	low	yes	fair	yes
>40	low	yes	excellent	no
31…40	low	yes	excellent	yes
<=30	medium	no	fair	no
<=30	low	yes	fair	yes
>40	medium	yes	fair	yes
<=30	medium	yes	excellent	yes
31…40	medium	no	excellent	yes
31…40	high	yes	fair	yes
>40	medium	no	excellent	no

$$Gain(age) = Info(D) - Info_{age}(D) = \underline{0.246}$$

- Information gains for other splits:

$$Gain(income) = 0.029$$

$$Gain(student) = 0.151$$

$$Gain(credit\_rating) = 0.048$$

Decision Tree Algorithm

- Key idea: ***Recursive Partitioning***
  - Take all of your data.
  - Consider *all* possible values of *all* variables.
  - Select the variable A and value(s) X_A that produces the greatest ***separation*** in the target.
    - For example: $(X_A=t_1)$ is called a "split".
    - If $X < t_1$ then send data point to the "left" branch, otherwise, send data point to the "right" branch.
  - Now repeat same process on the new emerging "nodes" using the data belonging to each node (data subset)

→ You get a "tree"

Overview

- Decision tree induction
- Criteria for attribute split
  - Information Gain
  - Gini Impurity

Attribute Selection Measure Comparison

- ***Information gain*** (ID3/C4.5)
  - All attributes are assumed to be categorical.
    - Can be modified for continuous-valued attributes.

- ***Gini impurity*** (IBM Intelligent Miner, CART, SLIQ, SPRINT)
  - All attributes are assumed continuous-valued.
    - Can be modified for categorical attributes.
  - Assume there exist several possible split values for each attribute.

Gini

- Corrado Gini, Italian statistician, 1884-1965
- ***Gini coefficient***
  - Used to show inequality of income distribution in a population
  - Large if unequal incomes,
    small if equal incomes

our interest

- ***Gini impurity***
  - A measure for the distribution of labels in a set
  - Large if many equally distributed labels,
    small if large probability only for few labels
- Hence, Gini coefficient ≠ Gini impurity
  - Attention: Both sometimes referred to as “Gini index”

Attribute Selection using *Gini Impurity*

- A data set *D* contains examples from *m* classes, and
- p_j is the relative frequency of class *j* in *D.*
- Then Gini impurity, *Gini*(*D*), is defined as:

$$Gini(D) = \underbrace{\sum_{j=1}^{m} p_j}_{\text{weighted average over the classes}} \underbrace{(1 - p_j)}_{\text{probability of incorrect labeling}} = 1 - \sum_{j=1}^{m} p_j^{\,2}$$

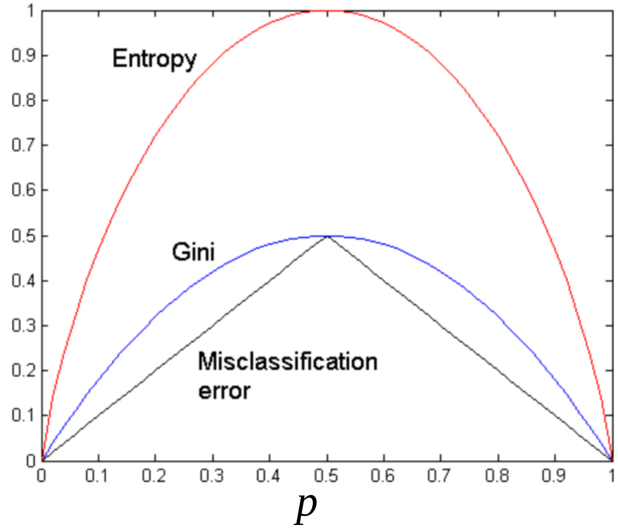
- Gini measures how often a randomly chosen element from the set would be incorrectly labeled if it was randomly labeled according to the distribution of labels in the subset.
- Should be minimized!
- Note: while $\Sigma_i\, p_i = 1$ always, not so $\Sigma_i\, p_i^2$

Attribute Selection using *Gini Impurity*

$$Gini(D) = \sum_{j=1}^{m} p_j(1 - p_j) = 1 - \sum_{j=1}^{m} p_j^{\;2}$$

- Minimum (1-1=0) when all records belong to one class
  $\rightarrow$ best in terms of information requirement
- Maximum (1 - 1/*m*) when records are *equally distributed* among all classes
  $\rightarrow$ worst in terms of information requirement

Gini impurity for *m=2* classes



Splitting Based on *Gini* Impurity

- When a node *p* is split into *k* partitions, the ***quality*** of split is computed as

$$Gini_{\text{split}} = \sum_{i=1}^{k} \frac{n_i}{n} Gini(i)$$

where: n_i = number of records at child i,
n = number of records at node p.

- Interpretation: weighted sum of *Gini* impurity for subsets i of samples caused by splitting
- Example: if a data set D is split into two subsets D_1 and D_2 with sizes n_1 and n_2 respectively, the *Gini* impurity $Gini_{split}(D)$ is:

$$Gini_{\text{split}}(D) = \frac{n_1}{n} Gini(D_1) + \frac{n_2}{n} Gini(D_2)$$

Splitting Based on *Gini* Impurity

- Need to enumerate all possible splitting points for each attribute
- The attribute that provides the smallest $Gini_{split}(D)$ is chosen to split the node

Gini Impurity – Example

- Class “buys_computer =yes” (9x)
- Class “buys_computer =no” (5x)

$$Gini(D) = Gini(9{,}5) = 1 - \left(\frac{9}{14}\right)^2 - \left(\frac{5}{14}\right)^2$$

$$= 0.459$$

$$Gini_{age}(D) = \boxed{\frac{5}{14} Gini(2{,}3)} + \frac{4}{14} \underbrace{Gini(4{,}0)}_{0} + \frac{5}{14} Gini(3{,}2)$$

$$= 0.343$$

age	yes_i	no_i	$Gini(yes_i,no_i)$
<=30	2	3	0,48
31…40	4	0	0
>40	3	2	0,48

“age <=30” has 5 out of 14 samples, with 2 “yes” and 3 “no”

age	income	student	credit_rating	buys_computer
<=30	high	no	fair	no
<=30	high	no	excellent	no
31…40	high	no	fair	yes
>40	medium	no	fair	yes
>40	low	yes	fair	yes
>40	low	yes	excellent	no
31…40	low	yes	excellent	yes
<=30	medium	no	fair	no
<=30	low	yes	fair	yes
>40	medium	yes	fair	yes
<=30	medium	yes	excellent	yes
31…40	medium	no	excellent	yes
31…40	high	yes	fair	yes
>40	medium	no	excellent	no

- Compute Gini for other splits:

$$Gini(income) = \ldots$$

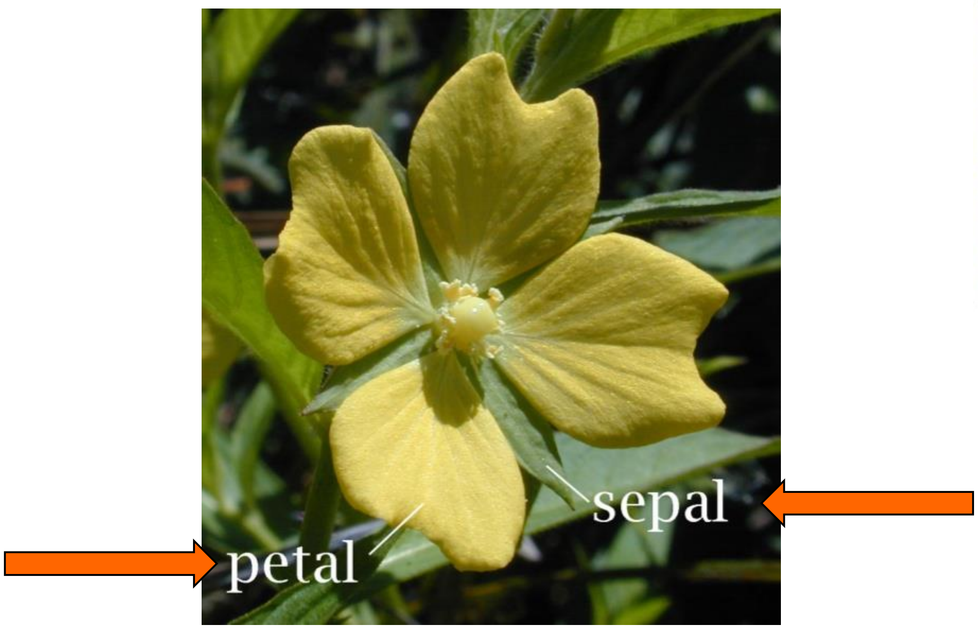
$$Gini(student) = \ldots$$

$$Gini(credit\_rating) = \ldots$$

- Considering all other splits:
- Split at lowest value

Matlab Demo of Decision Tree

- In Matlab, t = classregtree(X,y,'*Name*',*value*)   creates a decision tree.
- **Example**: Create a classification tree for Fisher's iris data, a typical test case for many classification techniques.



Iris setosa

Iris versicolor

Iris virginica

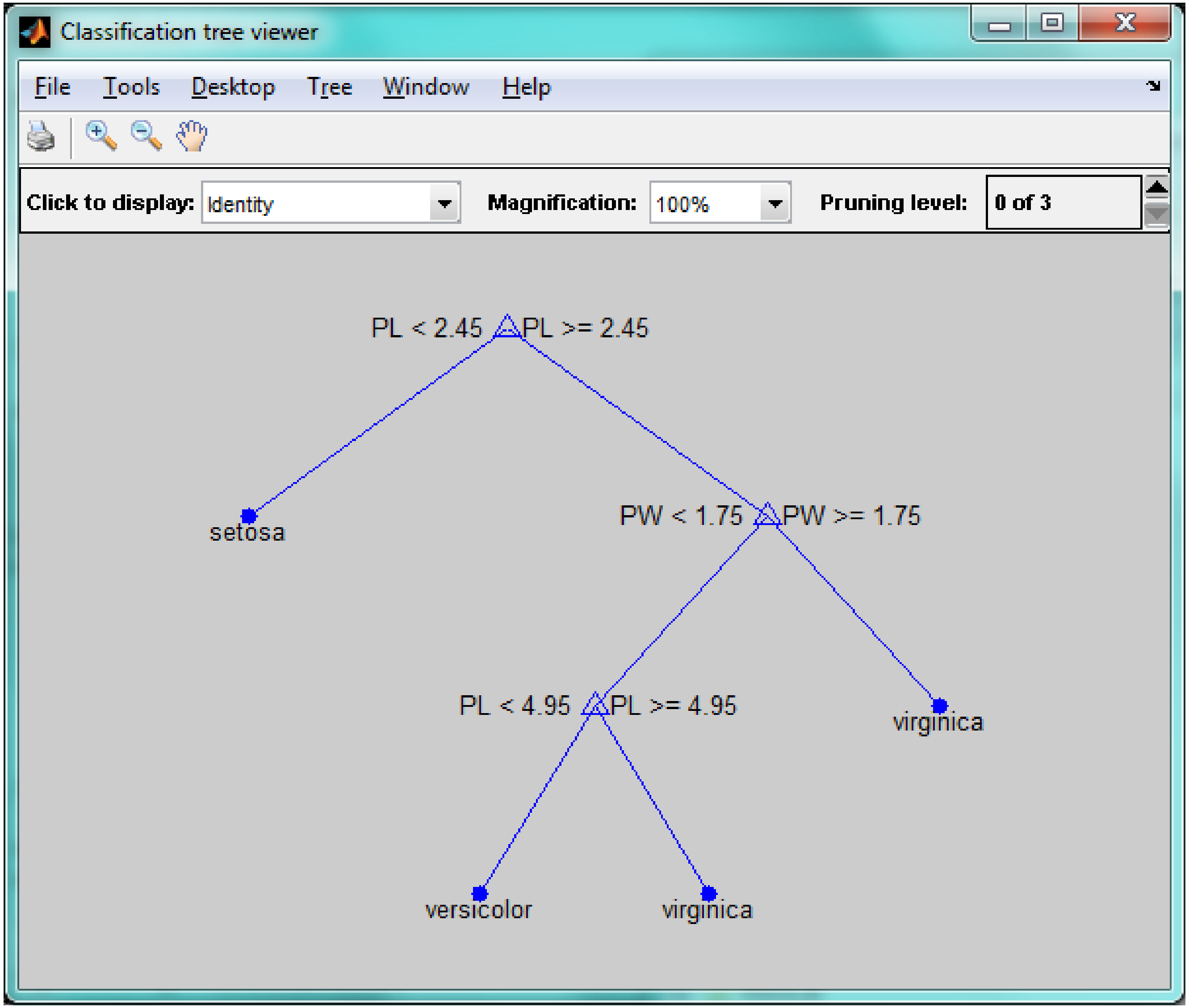
Matlab Demo of Decision Tree

- In Matlab, t = classregtree(X,y,'*Name*',*value*) creates a decision tree.
- **Example**: Create a classification tree for Fisher's iris data, a typical test case for many classification techniques.
  - In this data set, four attributes (Sepal Length, Sepal Width, Petal Length and Petal Width) are considered in order to distinguish three species of flowers (*Iris setosa*, *Iris virginica* and *Iris versicolor*).
  - Commands:

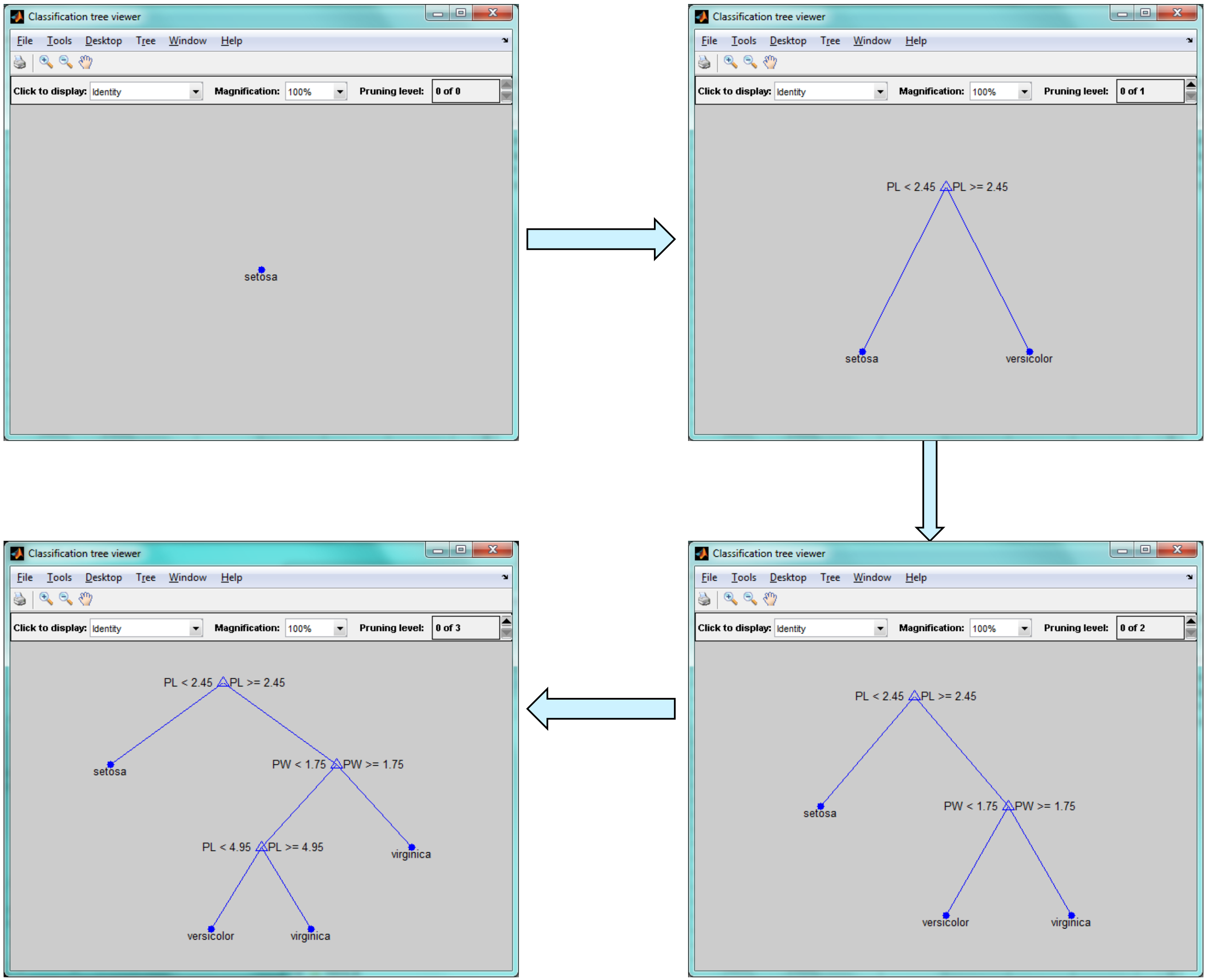
```
load fisheriris;
t = classregtree(meas,species,... 'names',{'SL'
'SW' 'PL' 'PW'});
```

  - Program generates a decision tree based on the data set.

Matlab Demo of Decision Tree



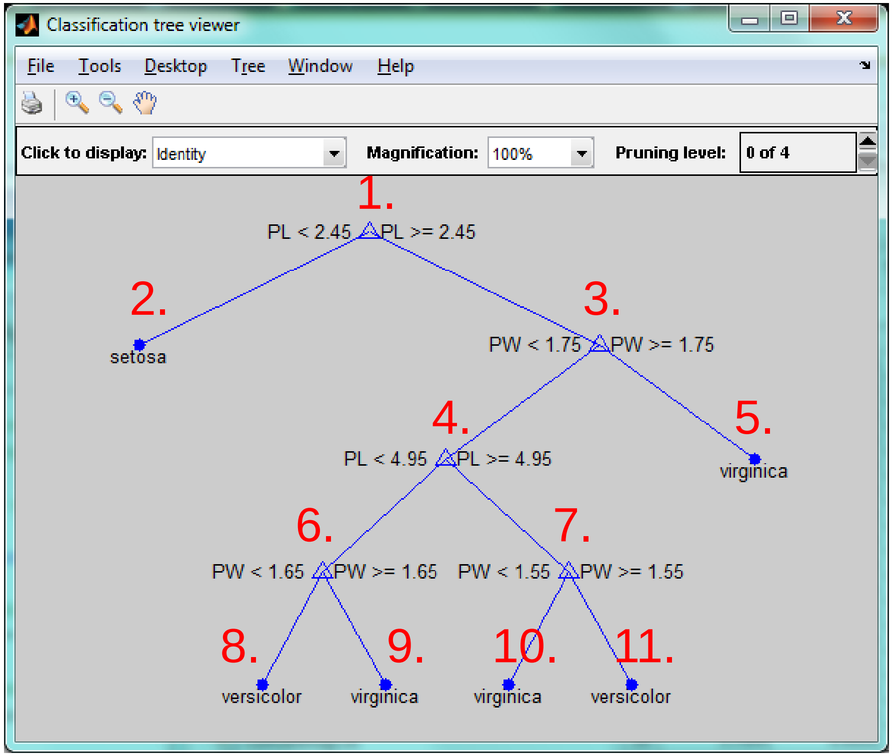
Matlab Demo of Decision Tree



Matlab Demo of Decision Tree

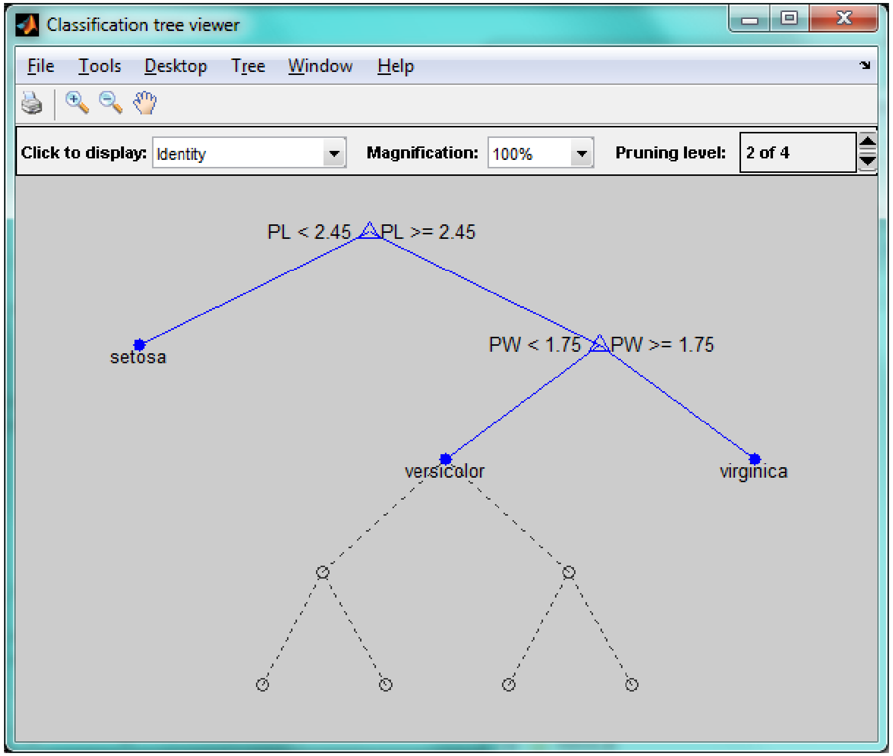
Final Decision tree for classification

1. if PL<2.45 then node 2 elseif
   PL>=2.45 then node 3
2. class = setosa
3. if PW<1.75 then node 4 elseif
   PW>=1.75 then node 5
4. if PL<4.95 then node 6 elseif
   PL>=4.95 then node 7
5. class = virginica
6. if PW<1.65 then node 8 elseif
   PW>=1.65 then node 9
7. if PW<1.55 then node 10 elseif
   PW>=1.55 then node 11
8. class = versicolor
9. class = virginica
10. class = virginica
11. class = versicolor



Matlab Demo of Decision Tree

- We can also prune the tree to avoid overfitting
- tt = prune(t,'level',2)



Decision Trees (Summary)

- Advantages
  - Automatically create tree representations from data
  - Trees can be converted to rules, can discover “new” rules
  - Identify most discriminating attribute first
    - Using Information Gain (Ratio) or Gini Impurity
  - Tree handle discrete (or discretized) attributes
    - Next lecture: deal with continuous, mixed, and missing attributes
- Disadvantages
  - Examines attributes individually, but not inter-attribute relationships
  - Future splits not known when splitting → not globally optimal tree
  - Trees can get large and difficult to understand
  - Can produce counter-intuitive rules
  - Tree induction has no direct relation to training objective, i.e. minimizing the classification error

News: HiWi Position at WTM Available

Studentische Hilfskraft (40 Std./Monat) im Bereich:

"Hardwarenahe Hilfstätigkeiten im Bereich Lehre und Forschung„

Aufgaben:

1) Mithilfe bei der technischen Vorbereitung des Lehr- und Forschungsbetriebes
2) Mithilfe bei der Wartung und Ausgabe von Geräten
3) Mithilfe bei der Sammlung von Forschungsergebnissen

Anforderungsprofil: Gute theoretische und praktische Kenntnisse von PC-Hardware (Standard-Hardware, Grafikkarten); Gute Kenntnisse im Linux Bereich, vorzugsweise Ubuntu; Kenntnisse in Windows 10; Gute Deutsch- und Englischkenntnisse in Wort und Schrift; Organisationstalent, Ordnungssinn und Fähigkeit des selbstständigen Einarbeitens; Teamfähigkeit & zuverlässige Email-Kommunikation; Sorgfältige Arbeitsweise; Interesse an einer längerfristigen Beschäftigung; noch mehrere Semester verbleibende Studienzeit

Kontakt:

Erik Strahl, F-232, strahl@informatik.uni-hamburg.de

https://www.inf.uni-hamburg.de/en/inst/ab/wtm/people/strahl.html