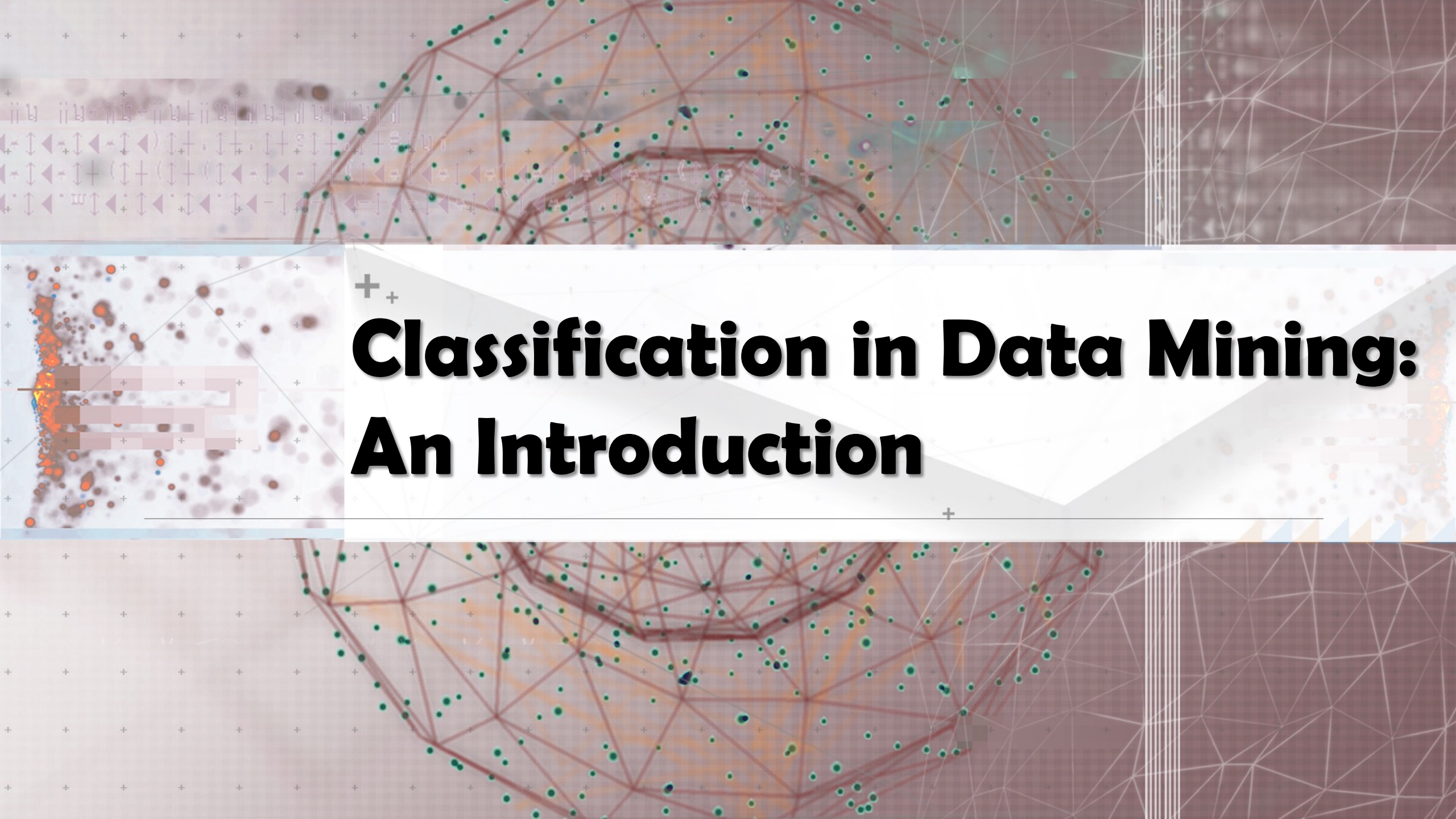


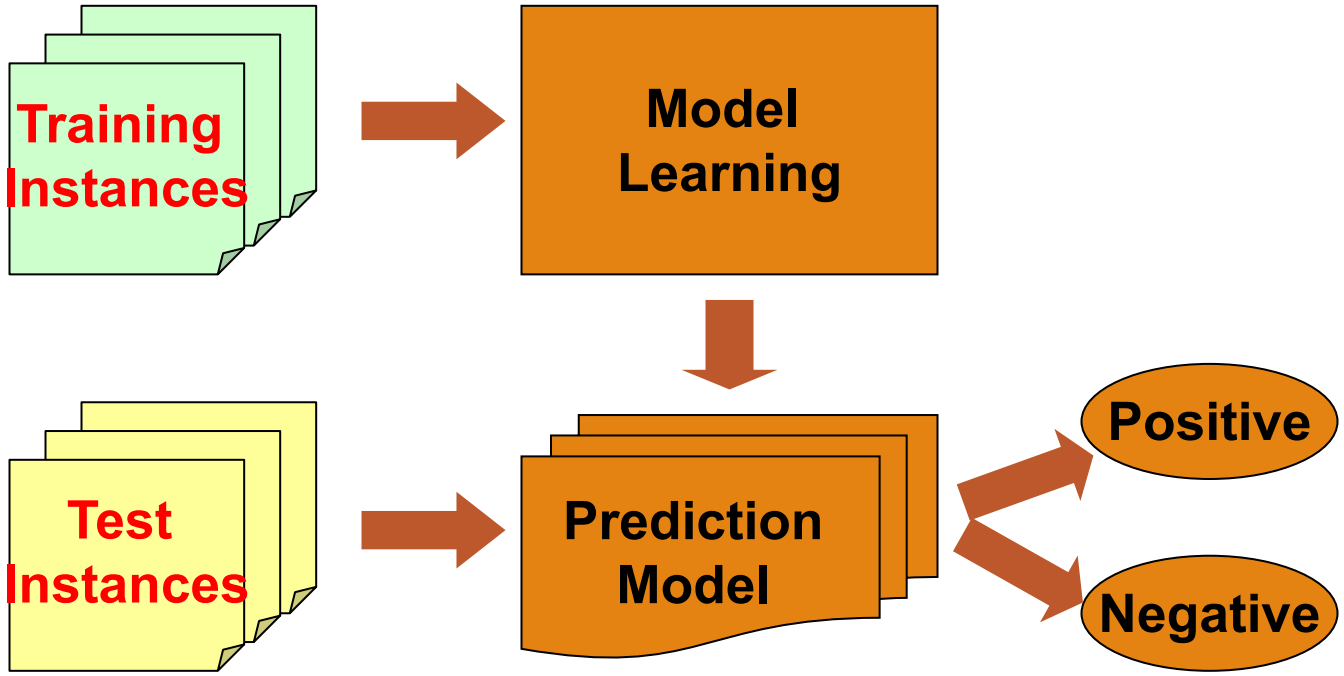
Classification in Data Mining: An Introduction

Supervised vs. Unsupervised Learning (1)

- Supervised learning (classification)
  - Supervision: The training data such as observations or measurements are accompanied by **labels** indicating the classes which they belong to
  - New data is classified based on the models built from the training set

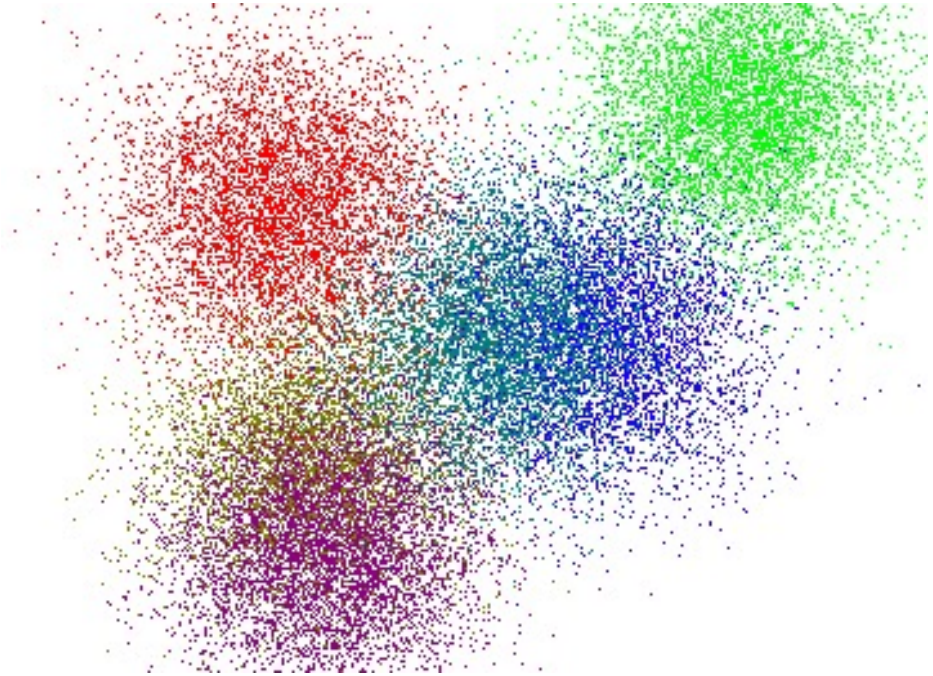
Training Data with class label:

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



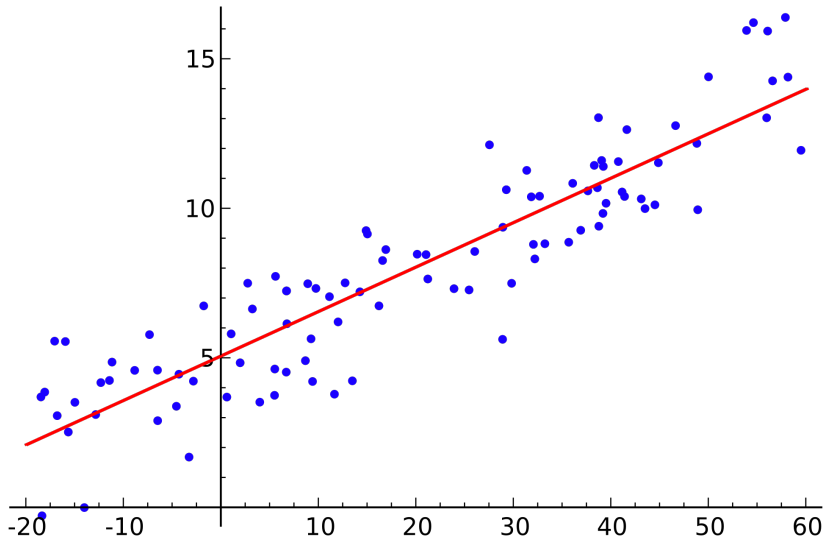
Supervised vs. Unsupervised Learning (2)

- Unsupervised learning (clustering)
  - The class labels of training data are unknown
  - Given a set of observations or measurements, establish the possible existence of classes or clusters in the data



Prediction Problems: Classification vs. Numeric Prediction

- Classification
  - Predict categorical class labels (discrete or nominal)
  - Construct a model based on the training set and the **class labels** (the values in a classifying attribute) and use it in classifying new data
- Numeric prediction
  - Model continuous-valued functions (i.e., predict unknown or missing values)
- Typical applications of classification
  - Credit/loan approval
  - Medical diagnosis: if a tumor is cancerous or benign
  - Fraud detection: if a transaction is fraudulent
  - Web page categorization: which category it is



Classification—Model Construction, Validation and Testing

- **Model construction**
  - Each sample is assumed to belong to a predefined class (shown by the **class label**)
  - The set of samples used for model construction is **training set**
  - **Model**: Represented as decision trees, rules, mathematical formulas, or other forms
- **Model Validation and Testing**:
  - **Test:** Estimate accuracy of the model
    - The known label of test sample is compared with the classified result from the model
    - *Accuracy:* % of test set samples that are correctly classified by the model
    - Test set is independent of training set
  - **Validation**: If *the test set* is used to select or refine models, it is called **validation** (or development) **(test) set**
- **Model Deployment:** If the accuracy is acceptable, use the model to classify new data

Major Reference Readings for the Course

- **Textbook**
  - Han, J., Pei, J., & Tong, H. (2022). *Data mining: Concepts and Techniques (4th ed.)*. Morgan Kaufmann
  - **Chapters most related to the course**
    - Chapter 6: Classification: Basic Concepts
    - Chapter 7: Classification: Advanced Methods
    - Chapter 10: Deep Learning

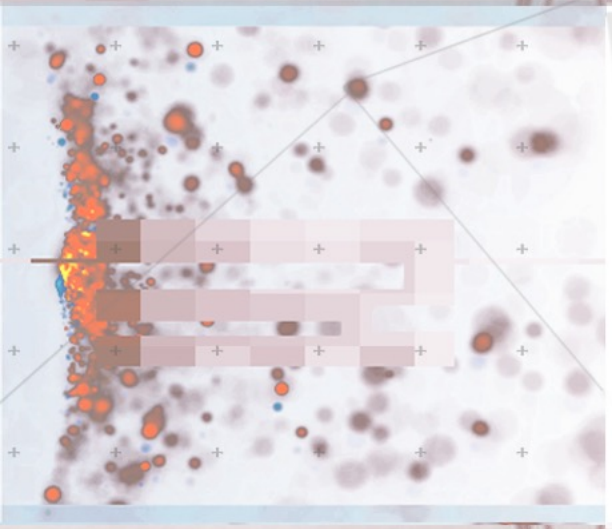
    Please use the most recent PDFs shared with the class
- Other references will be listed at the end of each lecture video

Course Structure

- Lesson 0: Classification in Data Mining: An Introduction
- Lesson 1: Decision Tree Induction
- Lesson 2: Bayes Classifier and Bayesian Networks
- Lesson 3: Model Evaluation, Selection, and Improvements
- Lesson 4: Classification with Weak Supervision
- Lesson 5: Linear Classifier and Support Vector Machines
- Lesson 6: Neural Networks and Deep Learning
- Lesson 7: Pattern-Based Classification and K-Nearest Neighbors Algorithm

Recommended Readings

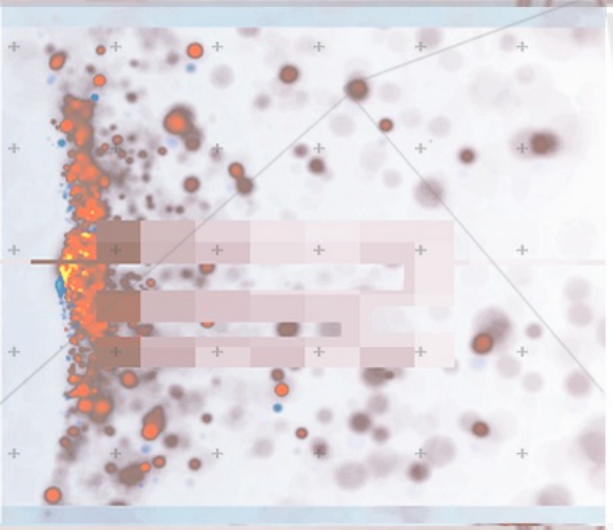
- C. C. Aggarwal, Data Mining: The Textbook, Springer, 2015
- R. O. Duda, P. E. Hart, and D. G. Stork. Pattern Classification, 2nd ed. John Wiley, 2001
- T. Hastie, R. Tibshirani, and J. Friedman, The Elements of Statistical Learning: Data Mining, Inference, and Prediction, 2nd ed., Springer, 2009
- T. M. Mitchell. Machine Learning. McGraw Hill, 1997
- P.-N. Tan, M. Steinbach, A. Karpatne, V. Kumar, Introduction to Data Mining, 2nd ed., Addison Wesley, 2013
- S. M. Weiss and C. A. Kulikowski. Computer Systems that Learn: Classification and Prediction Methods from Statistics, Neural Nets, Machine Learning, and Expert Systems. Morgan Kaufman, 1991
- I. H. Witten and E. Frank. Data Mining: Practical Machine Learning Tools and Techniques, 2ed. Morgan Kaufmann, 2005
- M. J. Zaki and W. Meira Jr., Data Mining and Analysis: Fundamental Concepts and Algorithms, Cambridge University Press, 2014



Decision Tree Induction

Outline

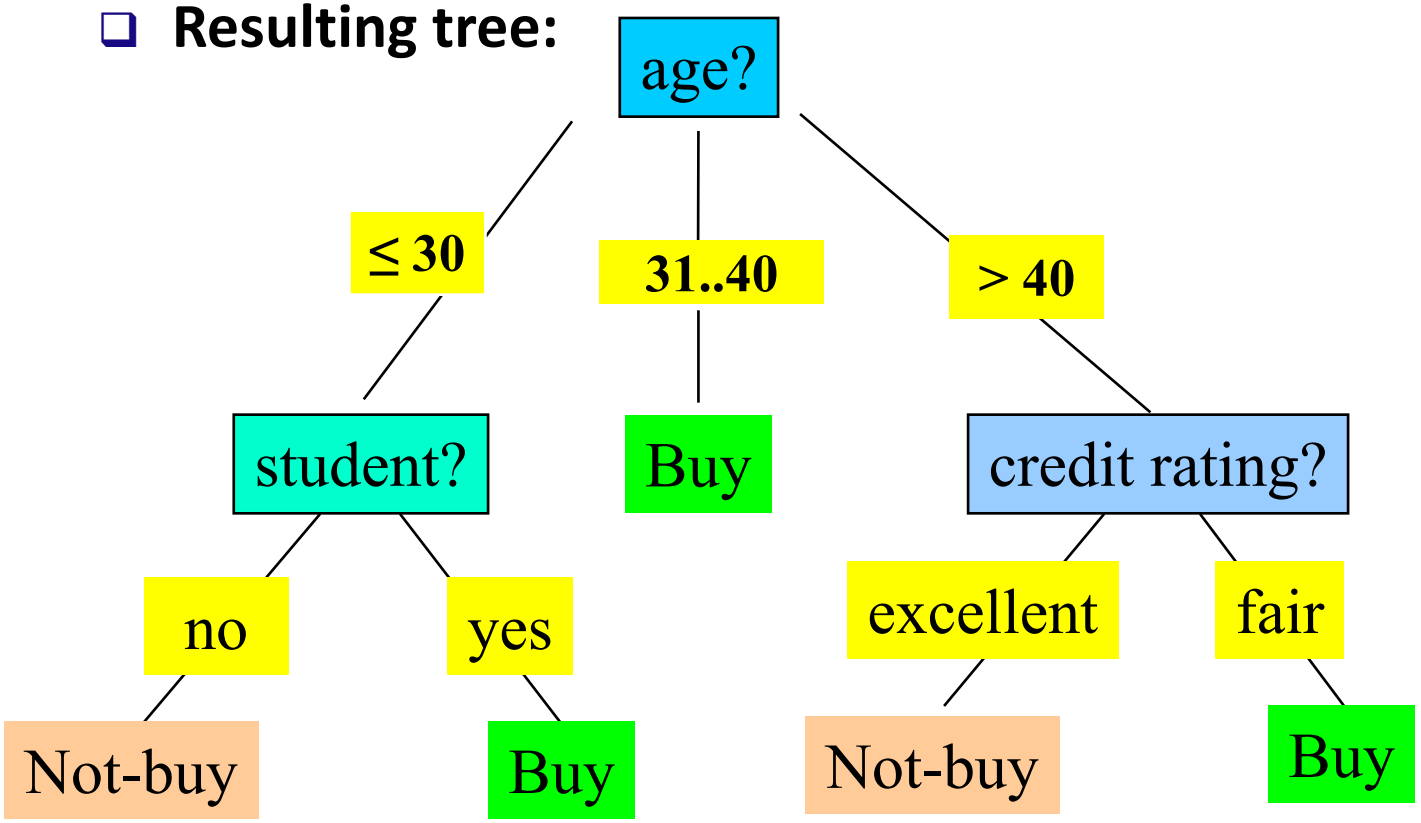
- Decision Tree Induction: Basic Idea and Algorithm
- Alternative Attribute Selection Measures in Decision Tree Induction
- Overfitting and Tree Pruning
- Decision Tree Construction in Large Datasets
- Visualization of Decision Trees and Tree Construction by Visual Data Mining



Decision Tree Induction: Basic Idea and Algorithm

Decision Tree Induction: An Example

- **Decision tree construction**:
  - A top-down, recursive, divide-and-conquer process
- **Resulting tree:**



Training data set: Who buys computer?

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

Note: The data set is adapted from "Playing Tennis" example of R. Quinlan

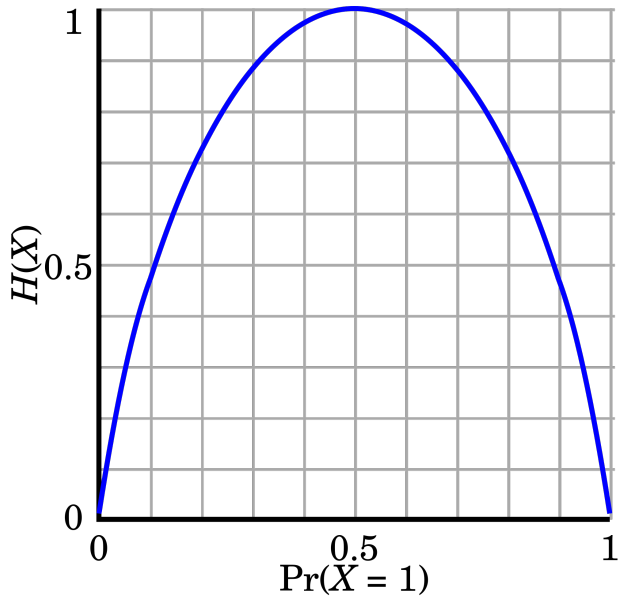
From Entropy to Info Gain: A Brief Review of Entropy

- Entropy (Information Theory)
  - A measure of uncertainty associated with a random number
  - Calculation: For a discrete random variable Y taking m distinct values $\{y_1, y_2, \ldots, y_m\}$

$$H(Y) = -\sum_{i=1}^{m} p_i \log(p_i) \quad where\ p_i = P(Y = y_i)$$

  - Interpretation
    - Higher entropy → higher uncertainty
    - Lower entropy → lower uncertainty
- Conditional entropy

$$H(Y|X) = \sum_{x} p(x) H(Y|X = x)$$



m = 2

Information Gain: An Attribute Selection Measure

- Select the attribute with the highest information gain (used in typical decision tree induction algorithm: ID3)
- Let p_i be the probability that an arbitrary tuple in D belongs to class C_i, estimated by $|C_{i,D}|/|D|$
- Expected information (entropy) needed to classify a tuple in D:

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

- Information needed (after using A to split D into v partitions) to classify D:

$$Info_A(D) = \sum_{j=1}^{v} \frac{|D_j|}{|D|} \times Info(D_j)$$

- Information gained by branching on attribute A

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

Example: Attribute Selection with Information Gain

- Class P: buys_computer = "yes"
- Class N: buys_computer = "no"

$$Info(D) = I(9,5) = -\frac{9}{14}\log_2(\frac{9}{14}) - \frac{5}{14}\log_2(\frac{5}{14}) = 0.940$$

age	p_i	n_i	$I(p_i, n_i)$
<=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

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

$\frac{5}{14}I(2,3)$ means "age <=30" has 5 out of 14 samples, with 2 yes'es and 3 no's.

Hence

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

Similarly, we can get

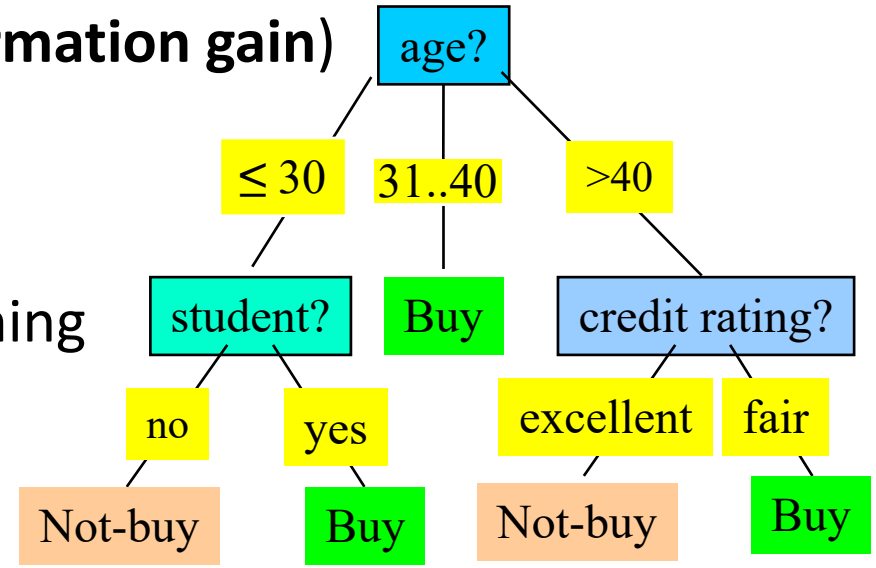
$$Gain(income) = 0.029$$

$$Gain(student) = 0.151$$

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

Decision Tree Induction: Algorithm

- Basic algorithm
  - Tree is constructed in a **top-down, recursive, divide-and-conquer manner**
  - At start, all the training examples are at the root
  - Examples are partitioned recursively based on selected attributes
  - On each node, attributes are selected based on the training examples on that node, and 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
  - There are no samples left
- Prediction
  - **Majority voting** is employed for classifying the leaf



How to Handle Continuous-Valued Attributes?

- Method 1: Discretize continuous values and treat them as categorical values
  - E.g., age: < 20, 20..30, 30..40, 40..50, > 50
- Method 2: Determine the ***best split point*** for continuous-valued attribute A
  - Sort the value A in increasing order:, e.g., 15, 18, 21, 22, 24, 25, 29, 31, …
  - *Possible split point:* the midpoint between *each pair of adjacent values*
    - $(a_i+a_{i+1})/2$ is the midpoint between the values of a_i and a_{i+1}
    - e.g., (15+18)/2 = 16.5, 19.5, 21.5, 23, 24.5, 27, 30, …
  - The point with the *maximum information gain* for A is selected as the **split-point** for A
- Split: Based on split point P
  - The set of tuples in D satisfying A ≤ P vs. those with A > P

Alternative Attribute Selection Measures in Decision Tree Induction

Gain Ratio: A Refined Measure for Attribute Selection

- Information gain measure is biased towards attributes with a large number of values
- Gain ratio: Overcomes the problem (as a normalization to information gain)

$$SplitInfo_A(D) = -\sum_{j=1}^{v} \frac{|D_j|}{|D|} \times \log_2\left(\frac{|D_j|}{|D|}\right)$$

  - GainRatio(A) = Gain(A)/SplitInfo(A)
  - The attribute with the maximum gain ratio is selected as the splitting attribute
  - Gain ratio is used in a popular algorithm C4.5 (a successor of ID3) by R. Quinlan
- Example
  - $\text{SplitInfo}_{\text{income}}(D) = -\frac{4}{14}\log_2\frac{4}{14} - \frac{6}{14}\log_2\frac{6}{14} - \frac{4}{14}\log_2\frac{4}{14} = 1.557$
  - GainRatio(income) = 0.029/1.557 = 0.019

Another Measure: Gini Index

- Gini index: Used in CART, and also in IBM IntelligentMiner
- If a data set D contains examples from n classes, gini index, $gini(D)$ is defined as
  - $gini(D) = 1 - \sum_{j=1}^{n} p_j^2$
    - p_j is the relative frequency of class j in D
- If a data set D is split on A into two subsets D_1 and D_2, the $gini$ index $gini(D)$ is defined as
  - $gini_A(D) = \frac{|D_1|}{|D|} gini(D_1) + \frac{|D_2|}{|D|} gini(D_2)$
- Reduction in Impurity:
  - $\Delta gini(A) = gini(D) - gini_A(D)$
- The attribute which provides the smallest $gini_{split}(D)$ (or the largest reduction in impurity) is chosen to split the node (*need to enumerate all the possible splitting points for each attribute*)

Computation of Gini Index

- Example: D has 9 tuples in buys_computer = "yes" and 5 in "no"

$$gini(D) = 1 - \left(\frac{9}{14}\right)^2 - \left(\frac{5}{14}\right)^2 = 0.459$$

- Suppose the attribute income partitions D into 10 in D_1: {low, medium} and 4 in D_2
  - $gini_{income \in \{low,medium\}}(D) = \frac{10}{14} gini(D_1) + \frac{4}{14} gini(D_2)$

$$= \frac{10}{14}\left(1 - \left(\frac{7}{10}\right)^2 - \left(\frac{3}{10}\right)^2\right) + \frac{4}{14}\left(1 - \left(\frac{2}{4}\right)^2 - \left(\frac{2}{4}\right)^2\right) = 0.443$$

$$= Gini_{income \in \{high\}}(D)$$

  - $Gini_{\{low, high\}}$ is 0.458; $Gini_{\{medium, high\}}$ is 0.450
  - Thus, split on $income \in$ {low, medium} (i.e., also $\{high\}$) has the lowest Gini index

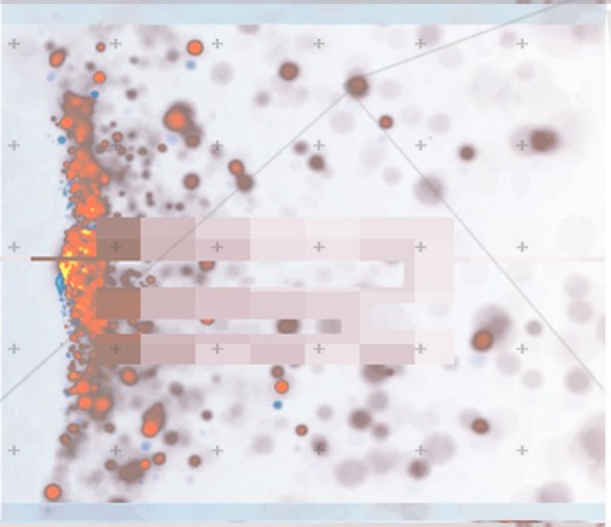
- The attributes discussed above assume categorical attributes
- The algorithm can also be adapted to continuous-valued attributes
  - One may need other tools, e.g., clustering, to get the possible split values

Comparing Three Attribute Selection Measures

- The three measures, in general, return good results but
  - **Information gain**:
    - is biased towards multivalued attributes
  - **Gain ratio**:
    - tends to prefer unbalanced splits in which one partition is much smaller than the others
  - **Gini index**:
    - is biased to multivalued attributes
    - has difficulty when # of classes is large
    - tends to favor tests that result in equal-sized partitions and purity in both partitions

Other Attribute Selection Measures

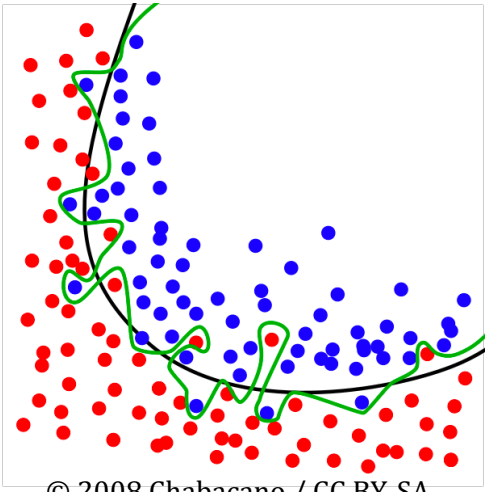
- Minimal Description Length (MDL) principle
  - Philosophy: The simplest solution is preferred
  - The best tree as the one that requires the fewest # of bits to (1) encode the tree, and (2) encode the exceptions to the tree
- CHAID: a popular decision tree algorithm, measure based on χ^2 test for independence
- Multivariate splits (partition based on multiple variable combinations)
  - CART: finds multivariate splits based on a linear combination of attributes
- There are many other measures proposed in research and applications
  - E.g., G-statistics, C-SEP
- Which attribute selection measure is the best?
  - Most give good results, none is significantly superior than others

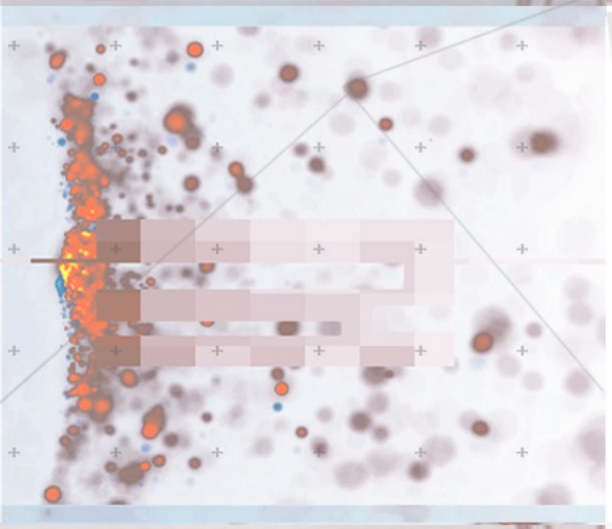


Overfitting and Tree Pruning

Overfitting and Tree Pruning

- Overfitting: An induced tree may overfit the training data
  - Too many branches, some may reflect anomalies due to noise or outliers
  - Poor accuracy for unseen samples
- Two approaches to avoid overfitting
  - Prepruning: *Halt tree construction early*—do not split a node if this would result in the goodness measure falling below a threshold
    - Difficult to choose an appropriate threshold
  - Postpruning: *Remove branches* from a "fully grown" tree—get a sequence of progressively pruned trees
    - Use a set of data different from the training data to decide which is the "best pruned tree"





Decision Tree Construction in Large Datasets

Classification in Large Databases

- Why is decision tree induction popular?
  - Relatively fast learning speed
  - Convertible to simple and easy to understand classification rules
  - Easy to be adapted to database system implementations (e.g., using SQL)
  - Comparable classification accuracy with other methods
- Scalability: Classifying data sets with millions of examples and hundreds of attributes with reasonable speed
- **RainForest** (VLDB'98—Gehrke, Ramakrishnan & Ganti)
  - Builds an AVC-list (attribute, value, class label)

RainForest: A Scalable Classification Framework

- The criteria that determine the quality of the tree can be computed separately
  - Builds an AVC-list: **AVC (Attribute, Value, Class_label)**
- **AVC-set** (of an attribute *X*)
  - Projection of training dataset onto the attribute *X* and class label where counts of individual class label are aggregated
- **AVC-group** (of a node *n*)
  - Set of AVC-sets of all predictor attributes at node *n*
- Then read the data again & do it similarly to generate the next level of the tree

age	income	student	credit_rating	com
<=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

The Training Data

AVC-set on *Age*

Age	Buy_Computer	
	yes	no
<=30	2	3
31..40	4	0
>40	3	2

AVC-set on *Income*

income	Buy_Computer	
	yes	no
high	2	2
medium	4	2
low	3	1

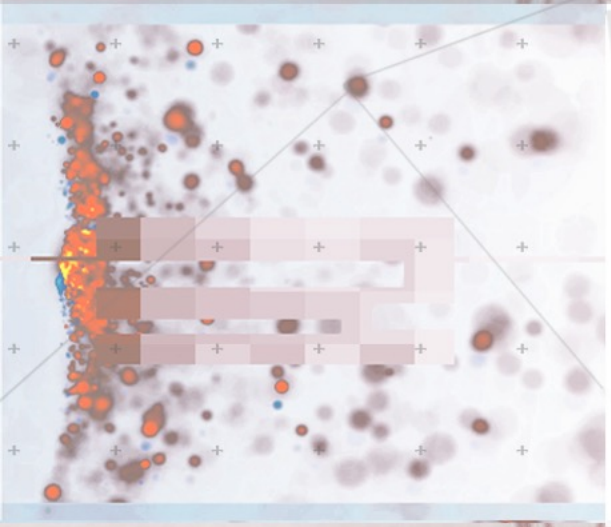
AVC-set on *Student*

student	Buy_Computer	
	yes	no
yes	6	1
no	3	4

AVC-set on *Credit_Rating*

Credit rating	Buy_Computer	
	yes	no
fair	6	2
excellent	3	3

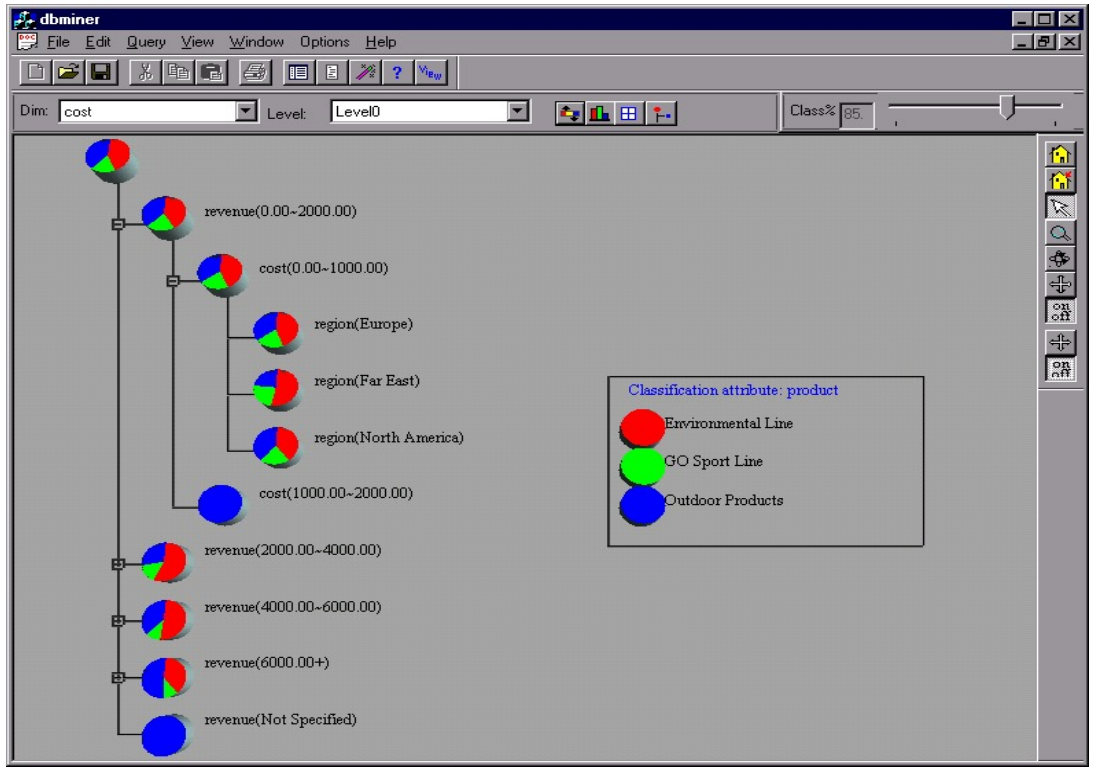
Its AVC Sets

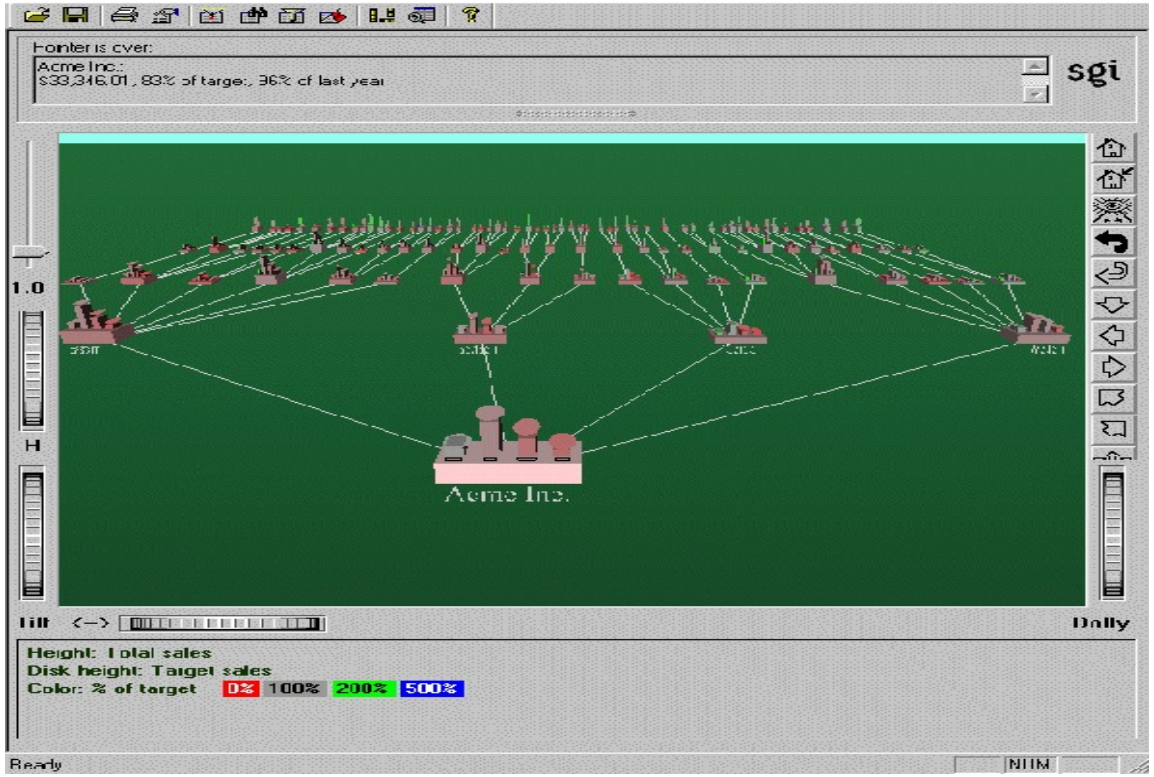


Visualization of Decision Trees and Tree Construction by Visual Data Mining

Visualization of a Decision Tree

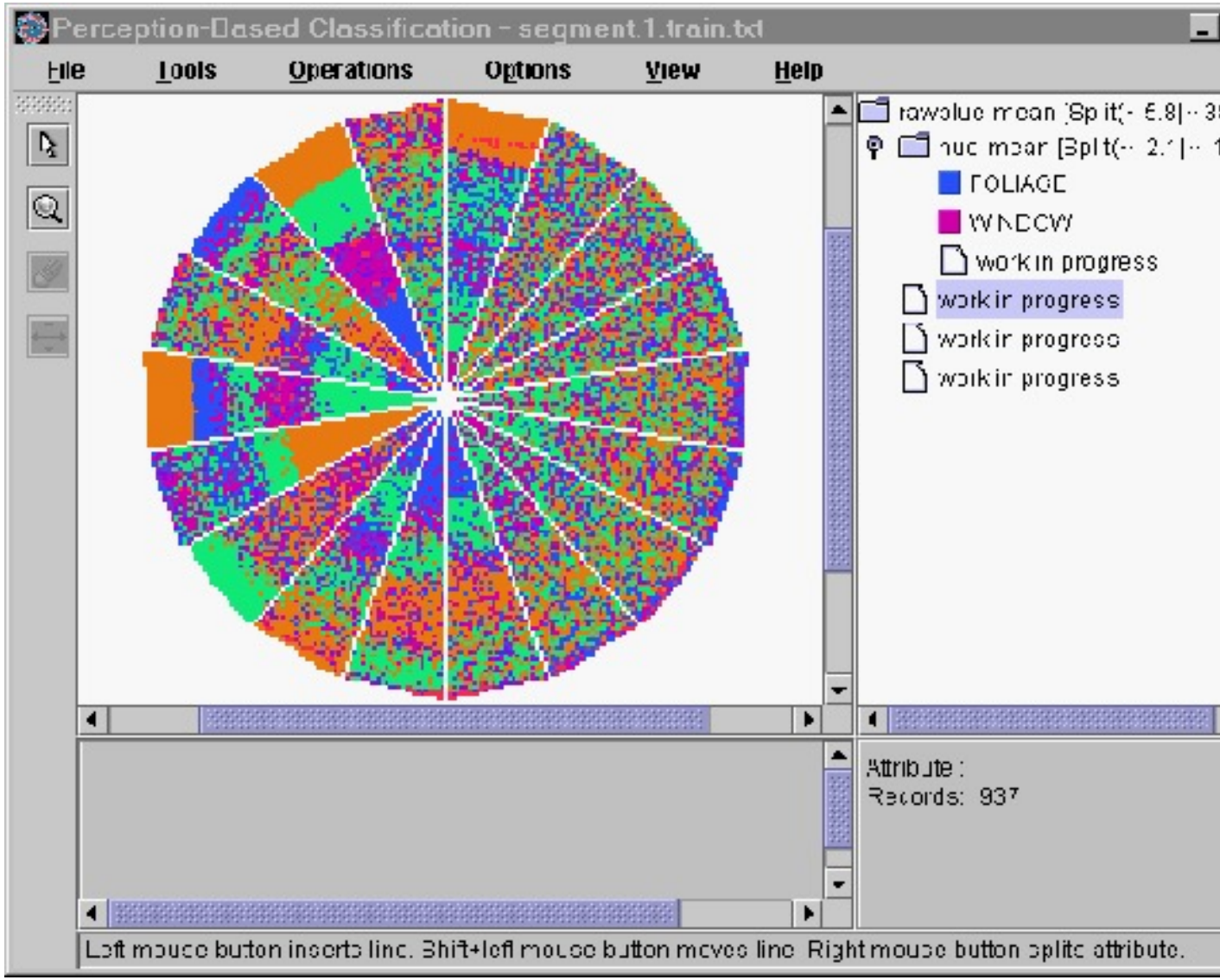
- A decision tree can be visualized with various interactive visualization tools
- The left figure is a partial tree visualization with three classes in DBMiner
- The right figure is an interactive tree visualization tool in SGI/MineSet 3.0

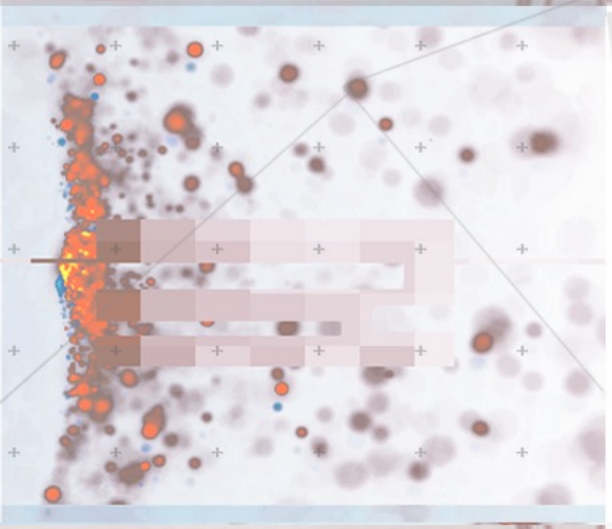




Interactive Visual Mining by Perception-Based Classification

- Perception-based classifier (PCB): developed at Univ. of Munich (1999)
- One color represents one class label
- One pie represents one attribute (or variable)
- The pie with random spread implies weak classification power
- The pie with clearly partitioned color strips implies good classification power
- One can select a good attribute and regenerate new pie charts for classification at the subsequent levels





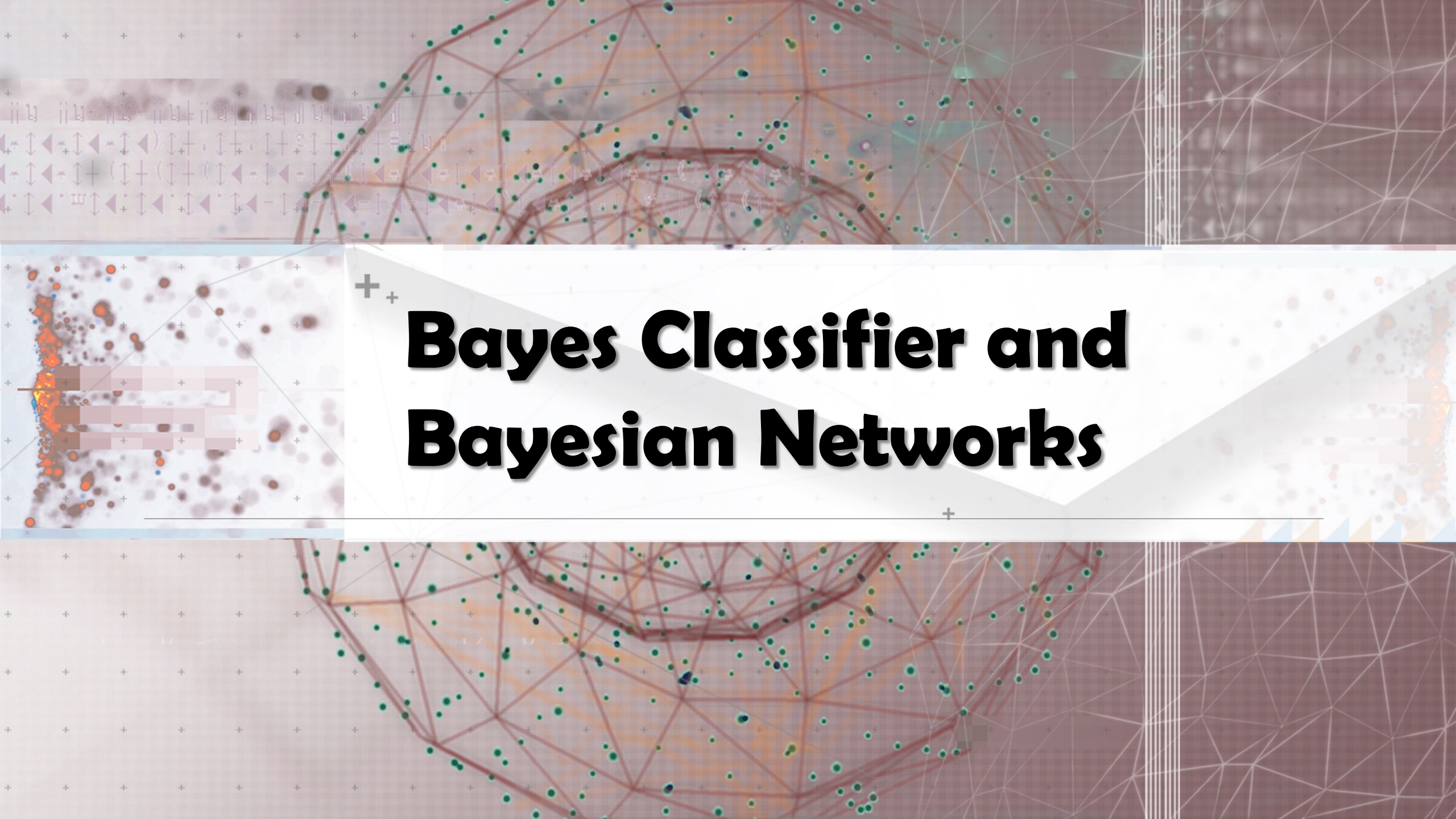
Summary

Summary

- Decision Tree Induction: Basic Idea and Algorithm
- Alternative Attribute Selection Measures in Decision Tree Induction
- Overfitting and Tree Pruning
- Decision Tree Construction in Large Datasets
- Visualization of Decision Trees and Tree Construction by Visual Data Mining

Recommended Readings

- M. Ankerst, C. Elsen, M. Ester, H.P. Kriegel, Visual Classification: An Interactive Approach to Decision Tree Construction. KDD 1999
- J. Gehrke, R. Ramakrishnan, V. Ganti. Rainforest: A Framework for Fast Decision Tree Construction of Large Datasets. Proc. of VLDB, 1998
- T.-S. Lim, W.-Y. Loh, Y.-S. Shih. A Comparison of Prediction Accuracy, Complexity, and Training Time of Thirty-three Old and New Classification Algorithms. Machine Learning, 2000
- J. Mingers. An Empirical Comparison of Pruning Methods for Decision Tree Induction, Machine Learning, 4(2): 227-243, 1989
- J. R. Quinlan. Induction of Decision Trees. *Machine Learning*, 1:81-106, 1986
- J. R. Quinlan. C4.5: Programs for Machine Learning. Morgan Kaufmann, 1993



Bayes Classifier and Bayesian Networks

Outline

- Bayes Classifier
- Bayesian Networks

Bayes Classifier

What Is Bayes Classifier?

- A statistical classifier
  - Perform *probabilistic prediction* (*i.e.,* predict the probability of a class membership)
- Foundation—Based on Bayes' Theorem
- Performance: A simple Bayes classifier, *naïve Bayes classifier*, has comparable performance with decision tree and many other classification methods
- Incremental
  - Each training example can incrementally increase/decrease the probability that a hypothesis is correct: Prior knowledge can be combined with observed data
- **Theoretical Standard**
  - Even when Bayesian methods are computationally intractable, they can provide a standard of optimal decision making against which other methods can be measured

Bayes' Theorem: Basics

- Total probability Theorem:

$$p(B) = \sum_i p(B|A_i)p(A_i)$$

- Bayes' Theorem:

$$p(H|\mathbf{X}) = \frac{p(\mathbf{X}|H)P(H)}{p(\mathbf{X})} \propto p(\mathbf{X}|H)\,P(H)$$

posteriori probability — What we should choose

likelihood — What we just see

prior probability — What we knew previously

- **X:** a data sample ("*evidence*")
- H: X belongs to class C

Prediction can be done based on Bayes' Theorem:

Classification is to derive the maximum posteriori

Naïve Bayes Classifier: Making a Naïve Assumption

- Practical difficulty of Bayes classifier: It requires initial knowledge of many probabilities, which may not be available or involving significant computational cost
- A Naïve Special Case
  - Make an additional assumption to simplify the model, but achieve comparable performance

Attributes are conditionally independent
(i.e., no dependence relation between attributes)

$$p(X|C_i) = \prod_k p(x_k|C_i) = p(x_1|C_i) \cdot p(x_2|C_i) \cdots\cdots p(x_n|C_i)$$

  - Only need to count the class distribution w.r.t. features
- Naive Bayes classifier: combines the independent feature model with a decision rule
  - The *MAP* (*maximum a posteriori*) decision rule: Pick the hypothesis that is most probable

Naïve Bayes Classifier: Categorical vs. Continuous Valued Features

- If feature x_k is categorical, $p(x_k = v_k|C_i)$ is the # of tuples in C_i with $x_k = v_k$, divided by $|C_{i,D}|$ (# of tuples of C_i in D)

$$p(X|C_i) = \prod_k p(x_k|C_i) = p(x_1|C_i) \cdot p(x_2|C_i) \cdots\cdots p(x_n|C_i)$$

- If feature x_k is continuous-valued, $p(x_k = v_k|C_i)$ is usually computed based on Gaussian distribution with a mean μ and standard deviation σ

$$p(x_k = v_k|C_i) = N\left(x_k|\mu_{C_i}, \sigma_{C_i}\right) = \frac{1}{\sqrt{2\pi}\sigma_{C_i}} e^{-\frac{\left(x-\mu_{C_i}\right)^2}{2\sigma^2}}$$

Naïve Bayes Classifier: Training Dataset

Class:
C1:buys_computer = ‘yes’
C2:buys_computer = ‘no’

Data to be classified:
X = (age <=30, Income = medium,
Student = yes, Credit_rating = Fair)

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

Naïve Bayes Classifier: An Example

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

- $P(C_i)$: P(buys_computer = "yes") = 9/14 = 0.643
  P(buys_computer = "no") = 5/14= 0.357
- Compute $P(X|C_i)$ for each class
  P(age = "<=30"|buys_computer = "yes") = 2/9 = 0.222
  P(age = "<= 30"|buys_computer = "no") = 3/5 = 0.6
  P(income = "medium" | buys_computer = "yes") = 4/9 = 0.444
  P(income = "medium" | buys_computer = "no") = 2/5 = 0.4
  P(student = "yes" | buys_computer = "yes) = 6/9 = 0.667
  P(student = "yes" | buys_computer = "no") = 1/5 = 0.2
  P(credit_rating = "fair" | buys_computer = "yes") = 6/9 = 0.667
  P(credit_rating = "fair" | buys_computer = "no") = 2/5 = 0.4
- **X = (age <= 30 , income = medium, student = yes, credit_rating = fair)**

$P(X|C_i)$: P(X|buys_computer = "yes") = 0.222 x 0.444 x 0.667 x 0.667 = 0.044
P(X|buys_computer = "no") = 0.6 x 0.4 x 0.2 x 0.4 = 0.019

$P(X|C_i)*P(C_i)$: P(X|buys_computer = "yes") * P(buys_computer = "yes") = 0.028
P(X|buys_computer = "no") * P(buys_computer = "no") = 0.007

Therefore, X belongs to class ("buys_computer = yes")

Avoiding the Zero-Probability Problem

- Naïve Bayesian prediction requires each conditional probability be **non-zero**
  - Otherwise, the predicted probability will be zero

$$\mathrm{p}(\mathrm{X}|C_i) = \prod_k p(x_k|C_i) = p(x_1|C_i) \cdot p(x_2|C_i) \cdots\cdots p(x_n|C_i)$$

- Example. Suppose a dataset with 1000 tuples:

  income = low (0), income= medium (990), and income = high (10)

- Use **Laplacian correction** (or Laplacian estimator)
  - *Adding 1 to each case*

    Prob(income = low) = 1/(1000 + 3)

    Prob(income = medium) = (990 + 1)/(1000 + 3)

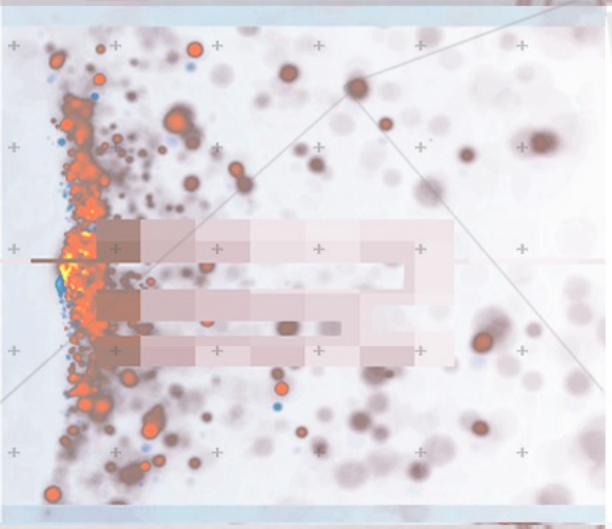
    Prob(income = high) = (10 + 1)/(1000 + 3)
  - The "corrected" probability estimates are close to their "uncorrected" counterparts

Naïve Bayes Classifier: Strength vs. Weakness

- Strength
  - Easy to implement
  - Good results obtained in most of the cases
- Weakness
  - Assumption: attributes conditional independence, therefore loss of accuracy
  - Practically, dependencies exist among variables
    - E.g., Patients: Profile: age, family history, etc.

      Symptoms: fever, cough etc.

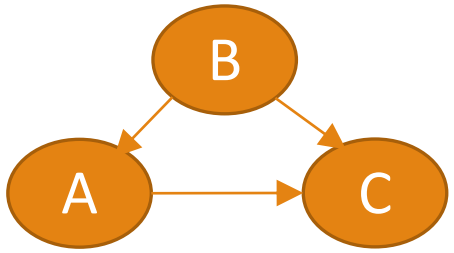
      Disease: lung cancer, diabetes, etc.
    - Dependencies among these cannot be modeled by Naïve Bayes Classifier
- How to deal with these dependencies?
  - Use Bayesian Belief Networks



Bayesian Networks

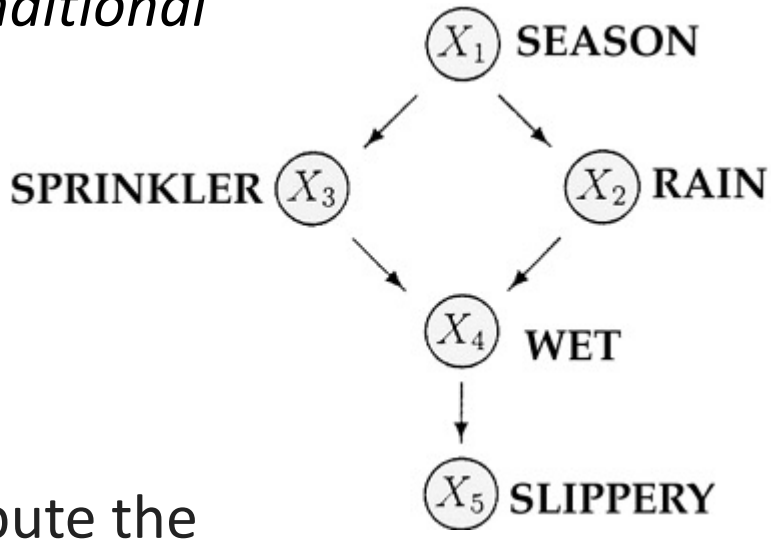
From Naïve Bayes to Bayesian Network

- A naïve Bayes classifier assumes that the value of a particular feature is independent of the value of any other feature, given the class variable
  - This assumption is often too simple to model the real world well
- Bayesian network (or Bayes network, belief network, Bayesian model or probabilistic directed acyclic graphical model) is a probabilistic **graphical model**
  - Represented by a set of *random variables* and *their conditional dependencies* via a *directed acyclic graph* (DAG)



directed **acyclic** graphical model

$$p(A, B, C) = p(B) \cdot p(A|B) \cdot p(C|A, B)$$

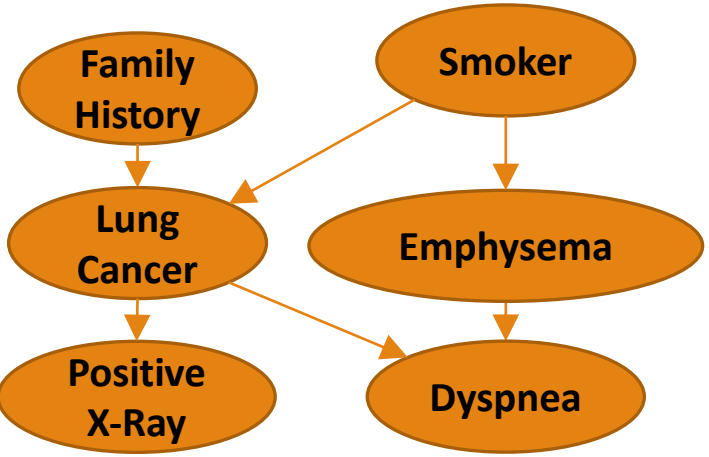


- Ex. Given symptoms, the network can be used to compute the probabilities of the presence of various diseases

Bayesian Network

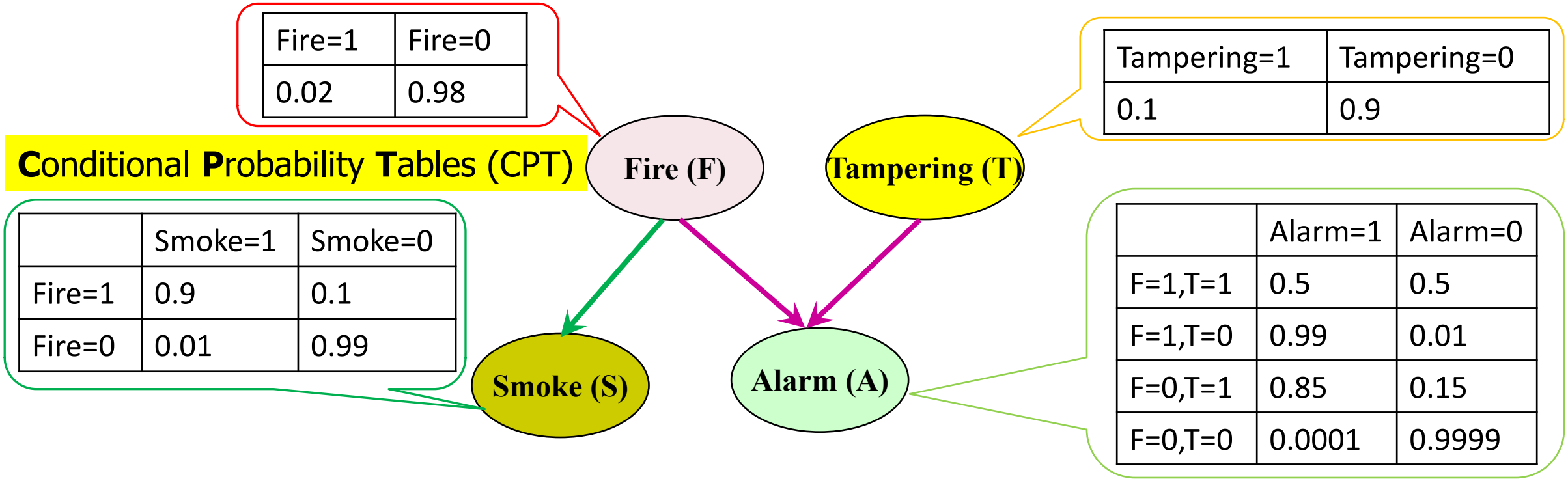
- **Bayesian network** (or **Bayesian belief network, probabilistic network**):
  - Allows *class conditional independencies* between *subsets* of variables
  - Represents the joint distribution compactly in a factorized way
  - Can be described by a generative process
- Two components:
  - A *directed acyclic graph* (called a structure)
    - The **nodes** represent random variables, observed or hidden
    - The **edges** represent direct dependency between variables
  - A set of *conditional probability tables* (CPTs)
    - CPTs are attached to the nodes

Nodes: random variables Links: dependency



	FH, S	*FH, ~S*	*~FH, S*	*~FH, ~S*
LC	0.8	0.5	0.7	0.1
~LC	0.2	0.5	0.3	0.9

Representing Joint Distribution with Bayesian Network



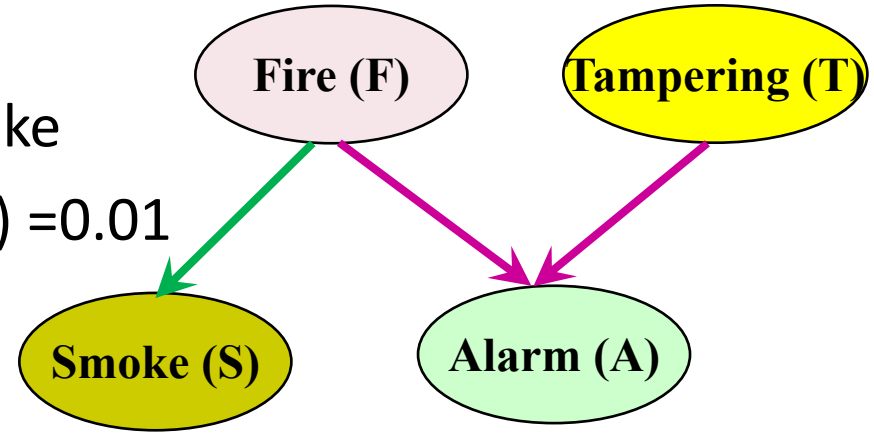
From the Bayesian network we can read the factorization of the joint distribution:

$$p(F, S, A, T) = p(F) \cdot p(T) \cdot p(S|F) \cdot p(A|F, T)$$

In general, we have the chain rule for Bayesian networks:

$$p(X) = \prod_k p(x_k|\text{Parents}(x_k))$$

Bayesian Network: An Example

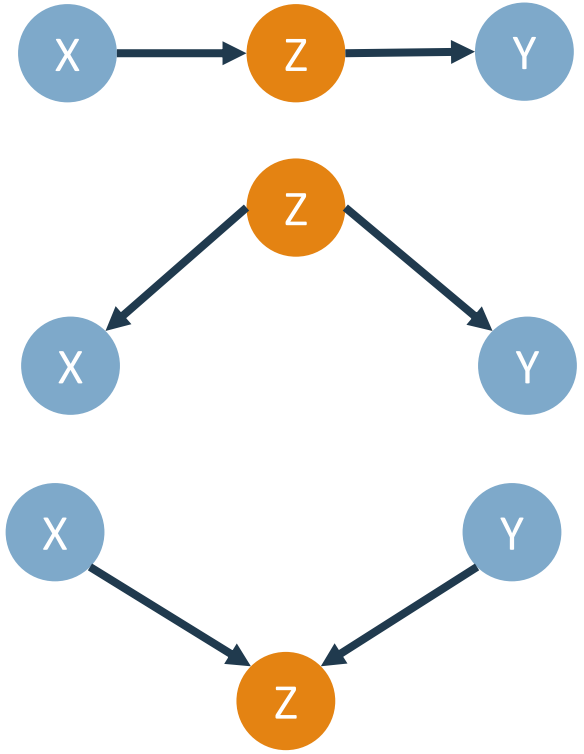


- Causal Reasoning:
  - The value of variable Fire influences variable Smoke
    - If F=1 → p(S=1|F=1) =0.9; if F=0 → p(S=1|F=0) =0.01
- Evidential Reasoning:
  - The value of variable Smoke also influences Fire
    - If S=1 → p(F=1|S=1) = $\frac{p(S=1|F=1)*p(F=1)}{p(S=1)}$ = 0.647; if S=0, p(F=1|S=0)=0.002
- Intercausal Reasoning:
  - The value of variable Fire does not influence Tampering
    - p(T|F) = p(T), F and T are independent
  - However, observing Alarm makes Fire and Tampering coupled
    - p(T|F, A) = $\frac{p(T,A,F)}{p(A|F)p(F)} = \frac{p(A|F,T)p(F)p(T)}{p(F)\sum_T p(A,T|F)} = \frac{p(A|F,T)p(T)}{\sum_T p(A|F,T)p(T)}$
    - p(T=1|F=1, A=1) = 0.053; p(T=1|F=0, A=1) ≈ 1.000

Dependencies in Bayesian Network

- Markovian assumption
  - Each variable becomes independent of its non-effects once its direct causes are known

- Consider the local structure of 3 variables
  - Cascade
    - Given the middle node Z, X and Y are independent
  - Common parent
    - Given the parent node Z, X and Y are independent
  - V-structure (common child):
    - Given the child node Z, X and Y are coupled
      - also called X explains away Y w.r.t. Z



Three Major Tasks on Bayesian Network

- **Representation (Construction)**
  - How to construct a Bayesian network so that the probability distribution reflects real-world phenomenon?
- **Inference**
  - Given a Bayesian network, how can we answer questions about events?
  - Marginal inference: What is the probability distribution of a particular variable given observed evidence?
  - MAP inference: What is the most likely assignment of variables given observed evidence?
- **Learning (Training)**
  - How can we fit a Bayesian network to data?
  - In some cases, the network structure needs to be learned as well

How Are Bayesian Networks Constructed?

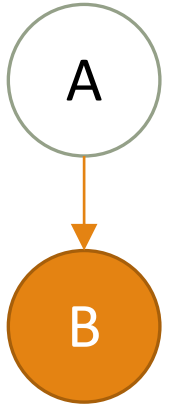
- **Subjective construction**
  - Human identification of (direct) causal structure
  - People are quite good at identifying direct causes from a given set of variables & whether the set contains all relevant direct causes
- **Synthesis from other specifications**
  - E.g., from a formal system design: block diagrams & information flow
- **Learning from data** (e.g., from medical records or student admission record)
  - Learn parameters given its structure or learn both structure and parameters
  - Maximum likelihood principle: favors Bayesian networks that maximize the probability of observing the given data set

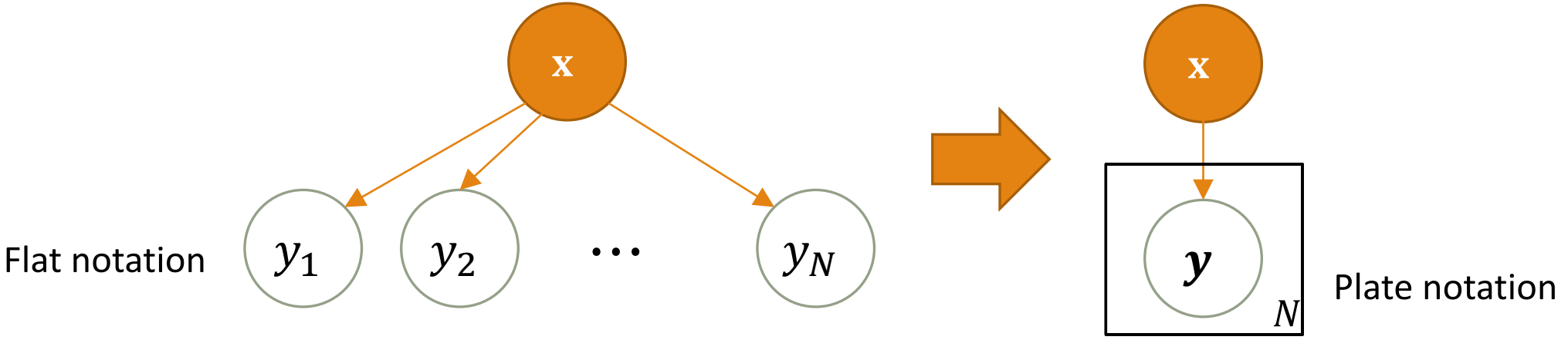
Training Bayesian Networks: Several Scenarios

- Scenario 1: Given both the network structure and all variables observable
  - *Compute only the CPT entries*
- Scenario 2: Network structure known, some variables hidden
  - Use *gradient descent* (a greedy hill-climbing method), i.e., search for a solution along the steepest descent of a criterion function
- Scenario 3: Network structure unknown, all variables observable
  - Search through the model space to *reconstruct network topology*
- Scenario 4: Unknown structure, all hidden variables
  - No good algorithms known for this purpose
- D. Heckerman. A Tutorial on Learning with Bayesian Networks. In *Learning in Graphical Models,* M. Jordan, ed. MIT Press, 1999

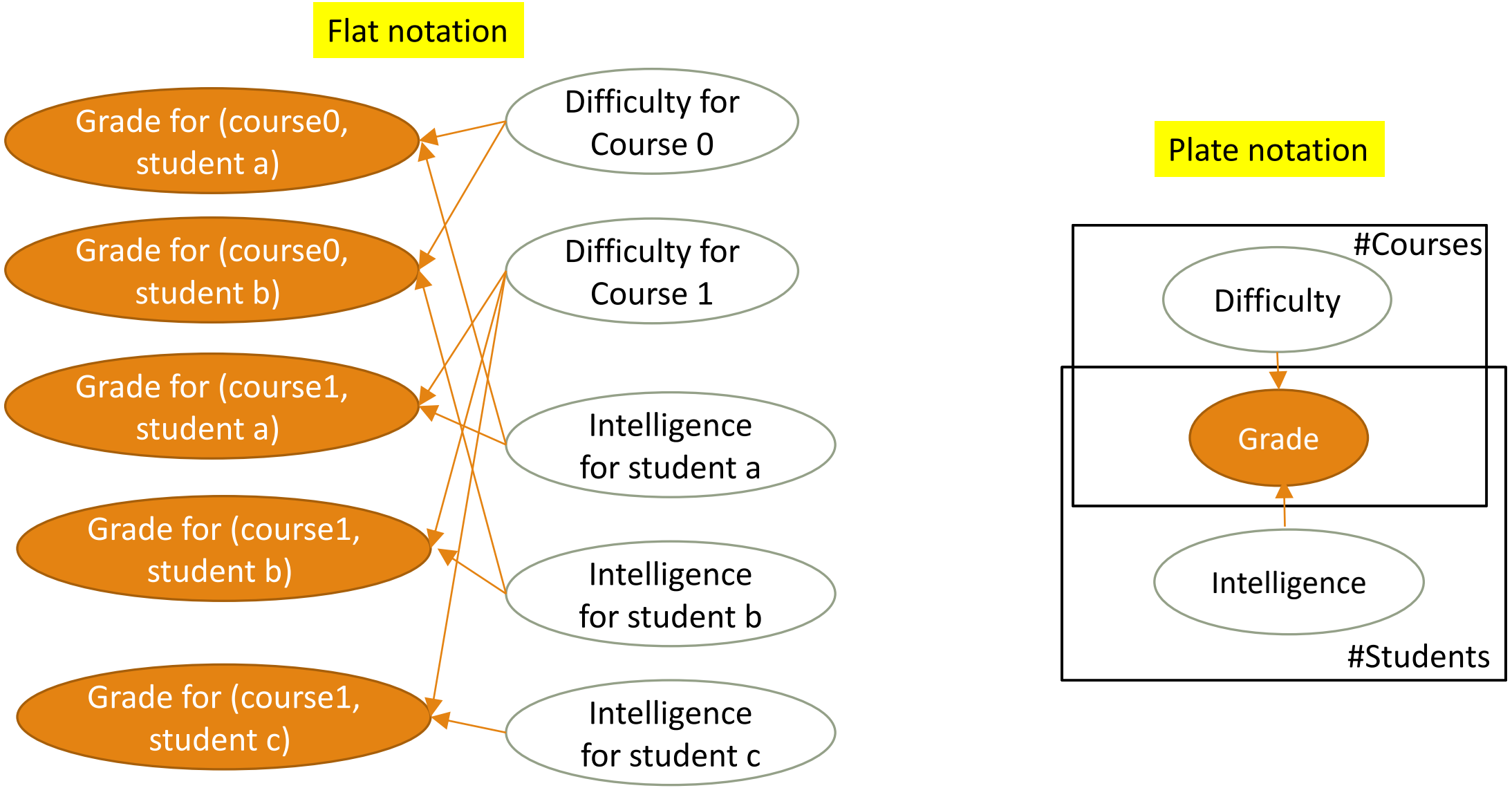
Probabilistic Graphic Model: Plate Notation

- How to represent variables and their dependencies concisely in a graphical model?
- Node (or **circle**) representing variables
  - A solid (or shaded) circle means the corresponding variable is *observed*; otherwise it is *hidden*
- **Directed edge** representing dependency among variables:
  - A Directed Acyclic Graphical (DAG) model
- Using plate notation instead of flat notation
  - The variables in the same plate share the same **conditional probability table**





An Example of Plate Notation



Summary

Summary

- Bayes Classifier
- Bayesian Networks

Recommended Readings

- D. Heckerman. A Tutorial on Learning with Bayesian Networks. In *Learning in Graphical Models,* M. Jordan, ed. MIT Press, 1999
- D. D. Lewis, "Naive (Bayes) at Forty: The Independence Assumption in Information Retrieval", ECML, 1998
- T.-S. Lim, W.-Y. Loh, Y.-S. Shih. A Comparison of Prediction Accuracy, Complexity, and Training Time of Thirty-three Old And New Classification Algorithms. Machine Learning, 2000
- K. P. Murphy, Machine Learning: A Probabilistic Perspective, MIT Press, 2012
- A. Ng and M. Jordan, On Discriminative vs. Generative Classifiers: A Comparison of Logistic Regression and Naive Bayes. *NIPS, 2002*
- J. Pearl, *Causality: Models, Reasoning, and Inference. Cambridge Univ. Press. 2000*
- S. Russell and P. Norvig, *Artificial Intelligence: A Modern Approach* (2nd ed.). Prentice Hall, 2003

Appendix: Detailed Computation of Some Probabilities on Slide #16 "Bayesian Network: An Example"

Thanks to Zoey (Sha) Li @ UIUC for working out the detailed computation

$$
\begin{aligned}
s(F=1|S=1) &= \frac{p(S=1,F=1)}{p(S=1)} \\
&= \frac{p(S=1|F=1)\times p(F=1)}{\sum_f p(S=1,F)} \\
&= \frac{p(S=1|F=1)\times p(F=1)}{p(S=1|F=1)p(F=1)+p(S=1|F=0)p(F=0)} \\
&= \frac{0.9\times 0.02}{0.9\times 0.02+0.01\times 0.98} \\
&= 0.647
\end{aligned}
$$

$$
\begin{aligned}
p(T=1|F=1,A=1) &= \frac{p(T=1,F=1,A=1)}{p(A=1|F=1)p(F=1)} \\
&= \frac{p(A=1|F=1,T=1)p(F=1)p(T=1)}{p(A=1|F=1)p(F=1)} \\
&= \frac{p(A=1|F=1,T=1)p(T=1)}{\sum_T p(A=1,T|F=1)} \\
&= \frac{0.5\times 0.1}{0.5\times 0.1+0.99\times 0.9} \\
&= 0.053
\end{aligned}
$$

$$
\begin{aligned}
p(T=1|F=0,A=1) &= \frac{p(A=1|T=1,F=0)p(T=1)}{\sum_T p(A=1,T|F=0)} \\
&= \frac{0.85\times 0.1}{0.85\times 0.1+0.001\times 0.9} \\
&= 0.9989\sim 1.000
\end{aligned}
$$