Data Mining

Practical Machine Learning Tools and Techniques

Slides for Chapter 6 of *Data Mining* by I. H. Witten, E. Frank and M. A. Hall

Part 2

Implementation:
Real machine learning schemes

- 6.8 Clustering: hierarchical, incremental, probabilistic
  - Hierarchical, incremental, probabilistic, Bayesian
- 6.9 Semisupervised learning
  - Clustering for classification, co-training

Clustering: how many clusters?

- How to choose k in k-means? Possibilities:
  - Choose k that minimizes cross-validated squared distance to cluster centers
  - Use penalized squared distance on the training data (eg. using an MDL criterion)
  - Apply k-means recursively with $k = 2$ and use stopping criterion (eg. based on MDL)
    - Seeds for subclusters can be chosen by seeding along direction of greatest variance in cluster
(one standard deviation away in each direction from cluster center of parent cluster)
    - Implemented in algorithm called X-means (using Bayesian Information Criterion instead of MDL)

Hierarchical clustering

- Recursively splitting clusters produces a hierarchy that can be represented as a *dendogram*
  - Could also be represented as a Venn diagram of sets and subsets (without intersections)
  - Height of each node in the dendogram can be made proportional to the dissimilarity between its children

Agglomerative clustering

- Bottom-up approach
- Simple algorithm
  - Requires a distance/similarity measure
  - Start by considering each instance to be a cluster
  - Find the two closest clusters and merge them
  - Continue merging until only one cluster is left
  - The record of mergings forms a hierarchical clustering structure – a *binary dendogram*

Distance measures

- *Single-linkage*
  - Minimum distance between the two clusters
  - Distance between the clusters closest two members
  - Can be sensitive to outliers
- *Complete-linkage*
  - Maximum distance between the two clusters
  - Two clusters are considered close only if all instances in their union are relatively similar
  - Also sensitive to outliers
  - Seeks compact clusters

Distance measures cont.

- Compromise between the extremes of minimum and maximum distance
- Represent clusters by their centroid, and use distance between centroids – *centroid linkage*
  - Works well for instances in multidimensional Euclidean space
  - Not so good if all we have is pairwise similarity between instances
- Calculate average distance between each pair of members of the two clusters – *average-linkage*
- Technical deficiency of both: results depend on the numerical scale on which distances are measured

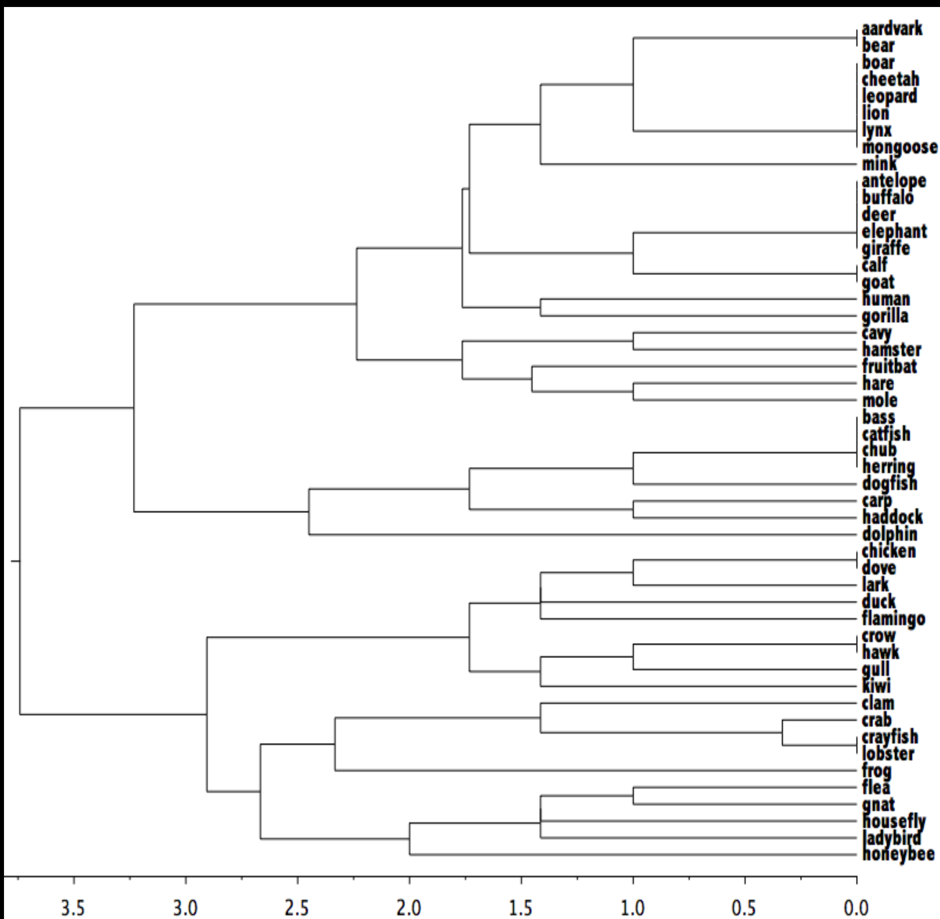
More distance measures

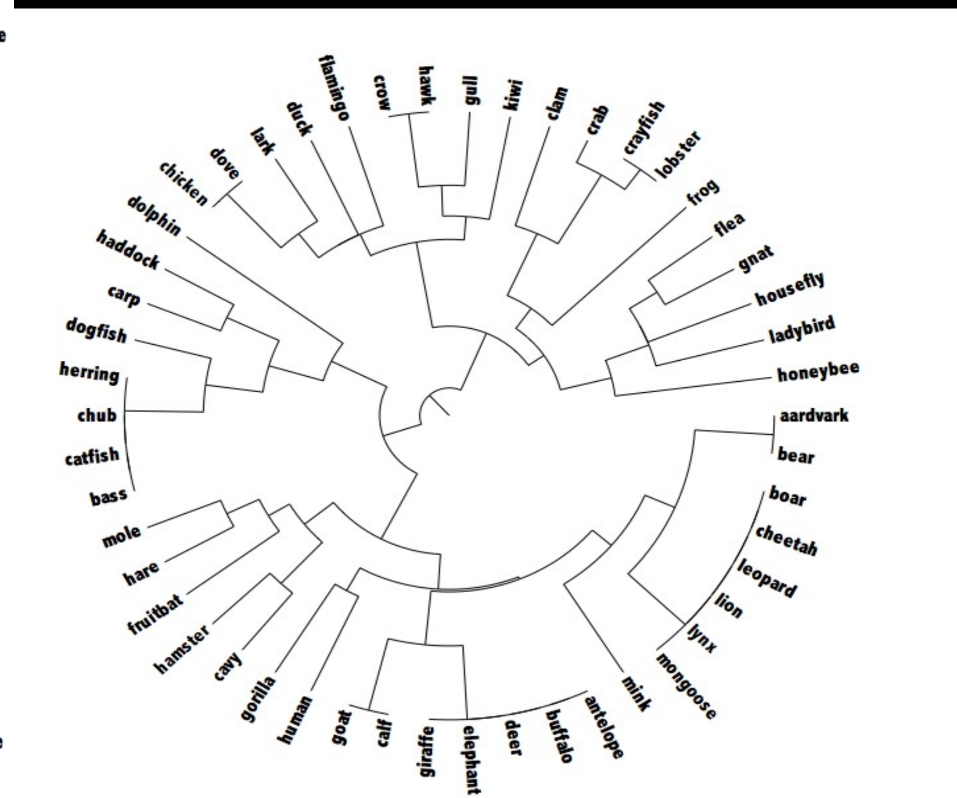
- *Group-average* clustering
  - Uses the average distance between all members of the merged cluster
  - Differs from average-linkage because it includes pairs from the same original cluster
- *Ward's* clustering method
  - Calculates the increase in the sum of squares of the distances of the instances from the centroid before and after fusing two clusters
  - Minimize the increase in this squared distance at each clustering step
- **All** measures will produce the same result if the clusters are compact and well separated

Example hierarchical clustering

- 50 examples of different creatures from the zoo data

Complete-linkage

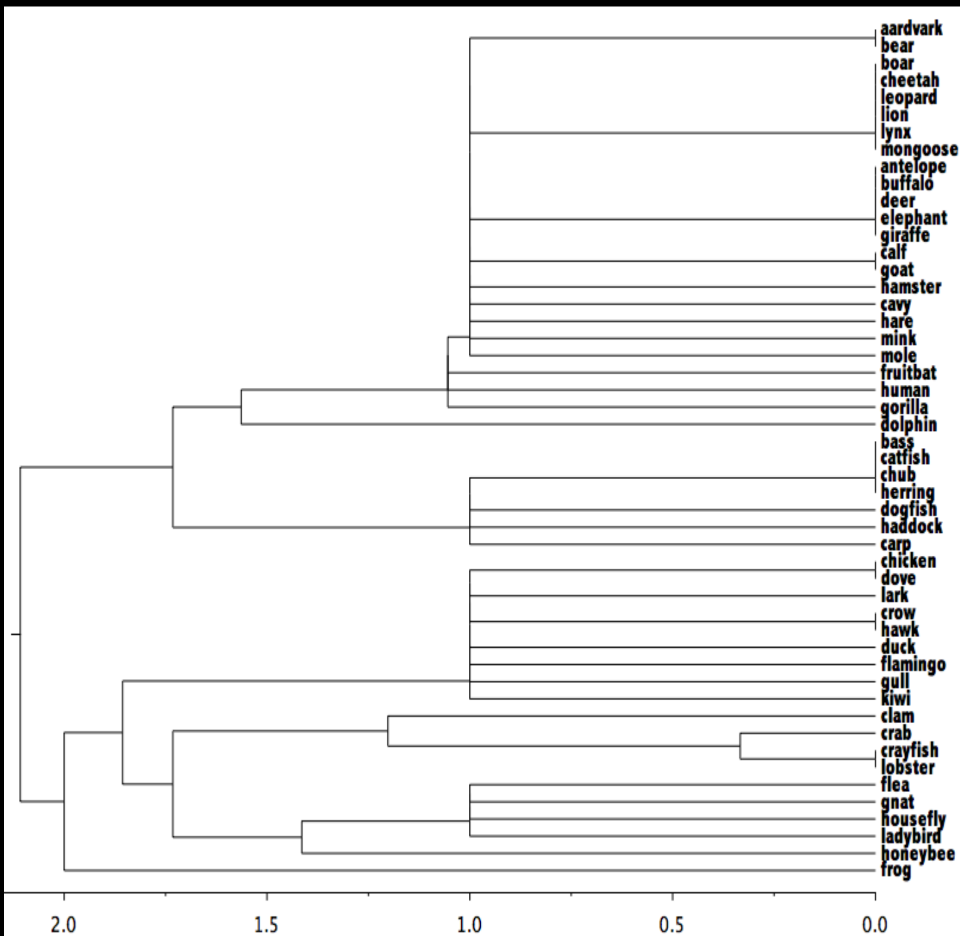




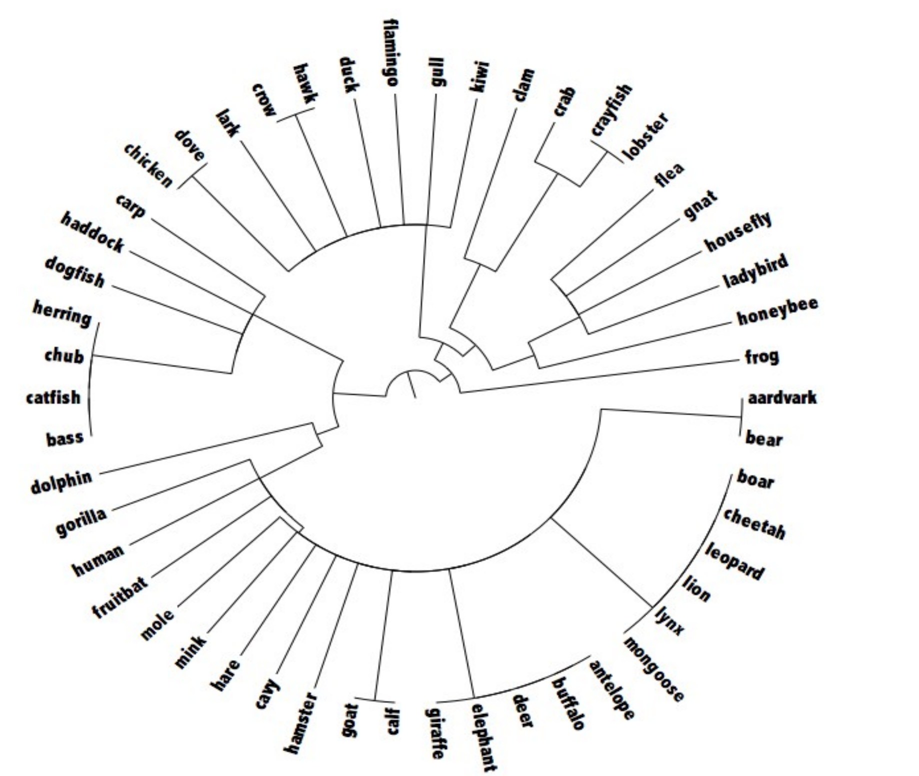
Dendogram

Polar plot

Example hierarchical clustering 2



Single-linkage



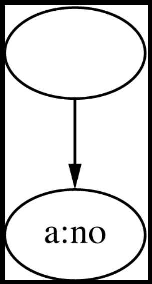
Incremental clustering

- Heuristic approach (COBWEB/CLASSIT)
- Form a hierarchy of clusters incrementally
- Start:
  - tree consists of empty root node
- Then:
  - add instances one by one
  - update tree appropriately at each stage
  - to update, find the right leaf for an instance
  - May involve restructuring the tree
- Base update decisions on *category utility*

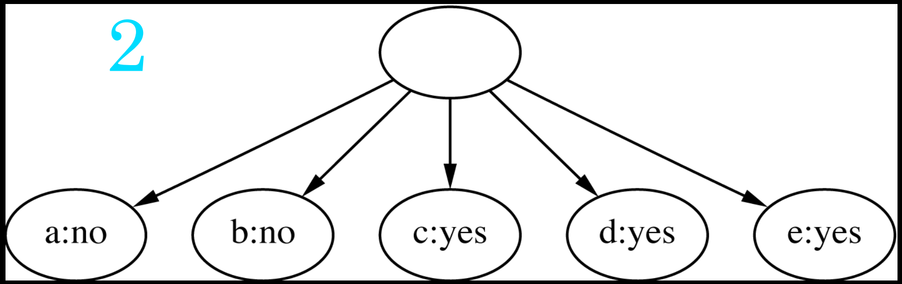
Clustering weather data

ID	Outlook	Temp.	Humidity	Windy
A	Sunny	Hot	High	False
B	Sunny	Hot	High	True
C	Overcast	Hot	High	False
D	Rainy	Mild	High	False
E	Rainy	Cool	Normal	False
F	Rainy	Cool	Normal	True
G	Overcast	Cool	Normal	True
H	Sunny	Mild	High	False
I	Sunny	Cool	Normal	False
J	Rainy	Mild	Normal	False
K	Sunny	Mild	Normal	True
L	Overcast	Mild	High	True
M	Overcast	Hot	Normal	False
N	Rainy	Mild	High	True

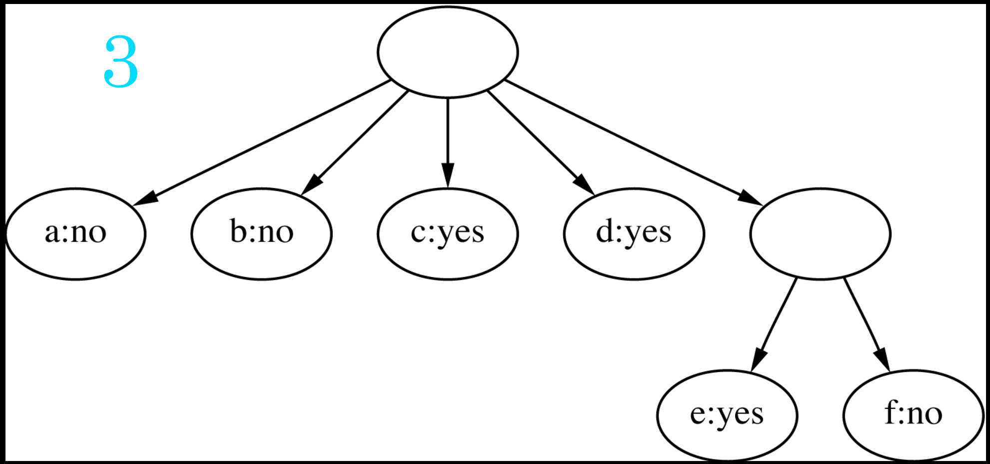
1



2



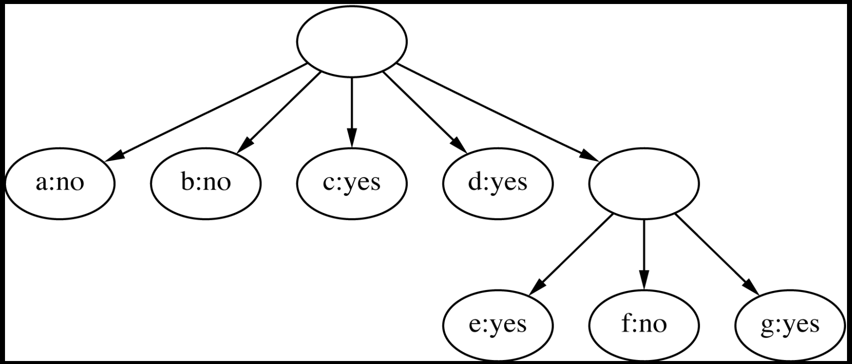
3



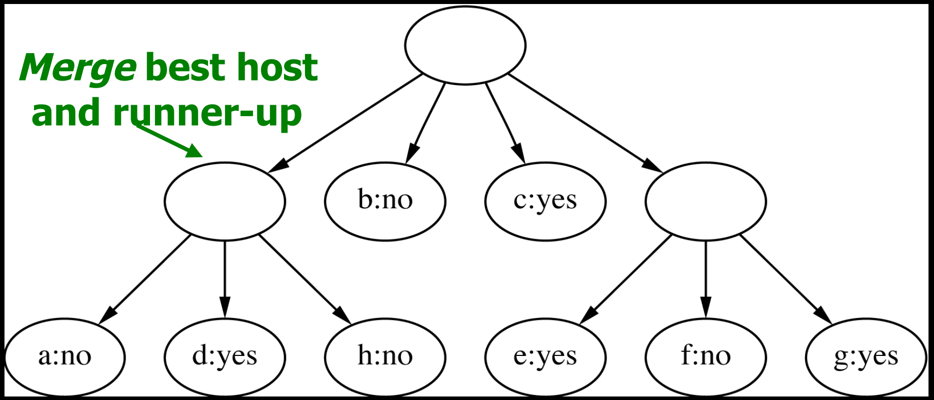
Clustering weather data

ID	Outlook	Temp.	Humidity	Windy
A	Sunny	Hot	High	False
B	Sunny	Hot	High	True
C	Overcast	Hot	High	False
D	Rainy	Mild	High	False
E	Rainy	Cool	Normal	False
F	Rainy	Cool	Normal	True
G	Overcast	Cool	Normal	True
H	Sunny	Mild	High	False
I	Sunny	Cool	Normal	False
J	Rainy	Mild	Normal	False
K	Sunny	Mild	Normal	True
L	Overcast	Mild	High	True
M	Overcast	Hot	Normal	False
N	Rainy	Mild	High	True

4

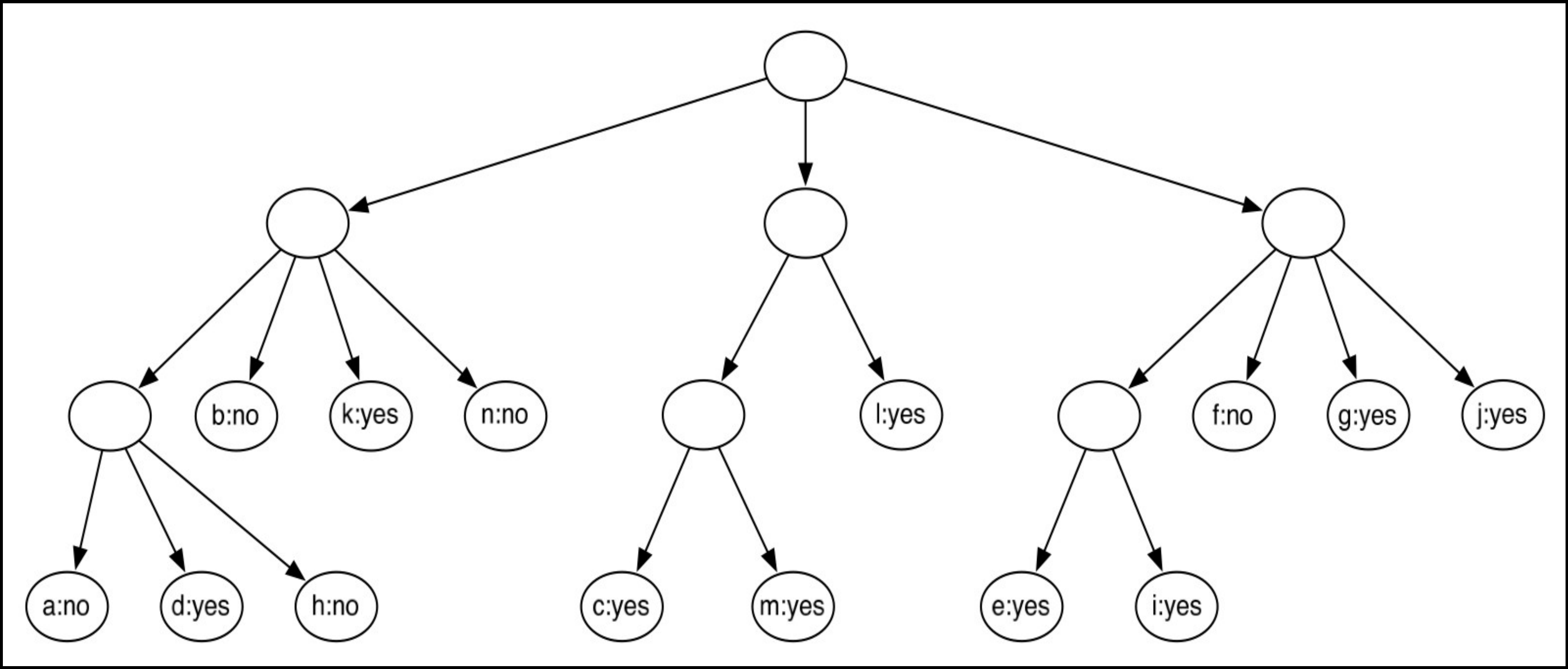


5

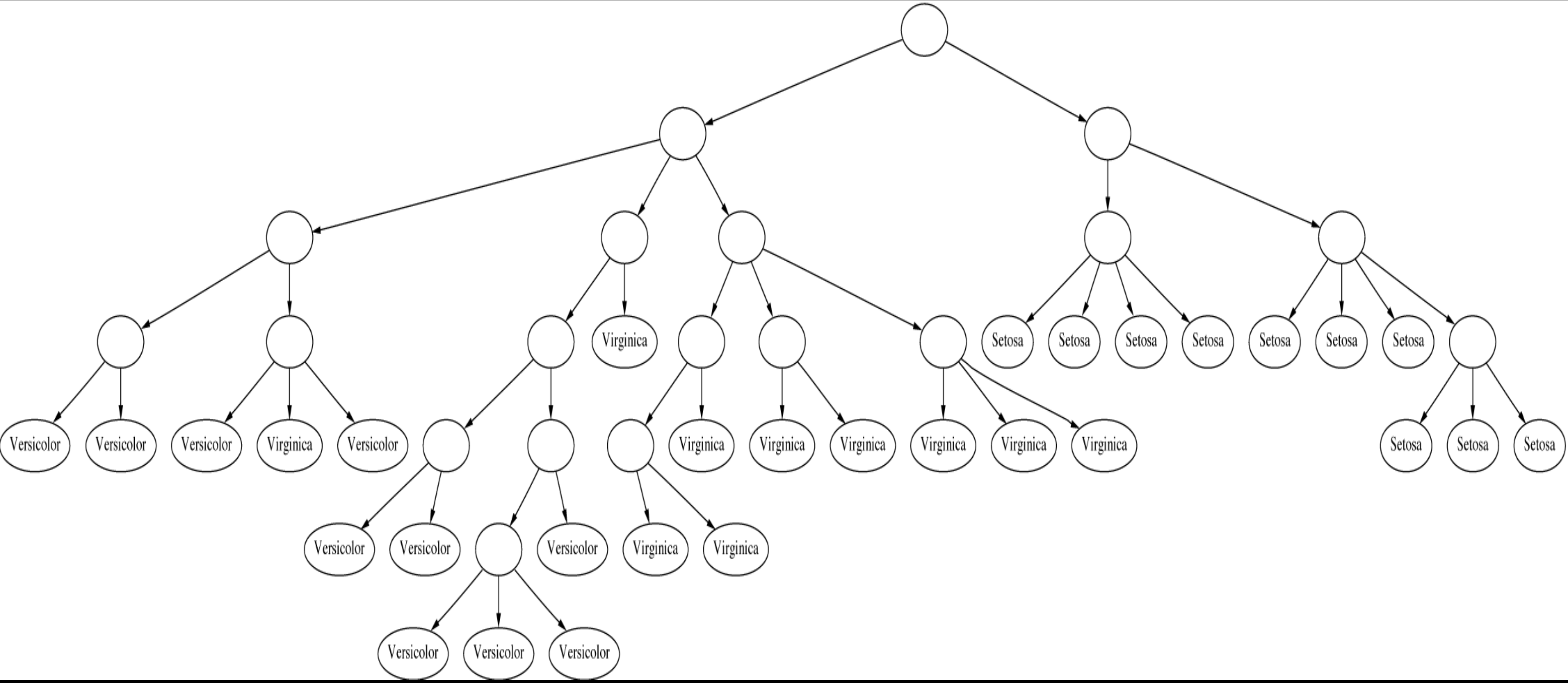


Consider *splitting* the best host if merging doesn't help

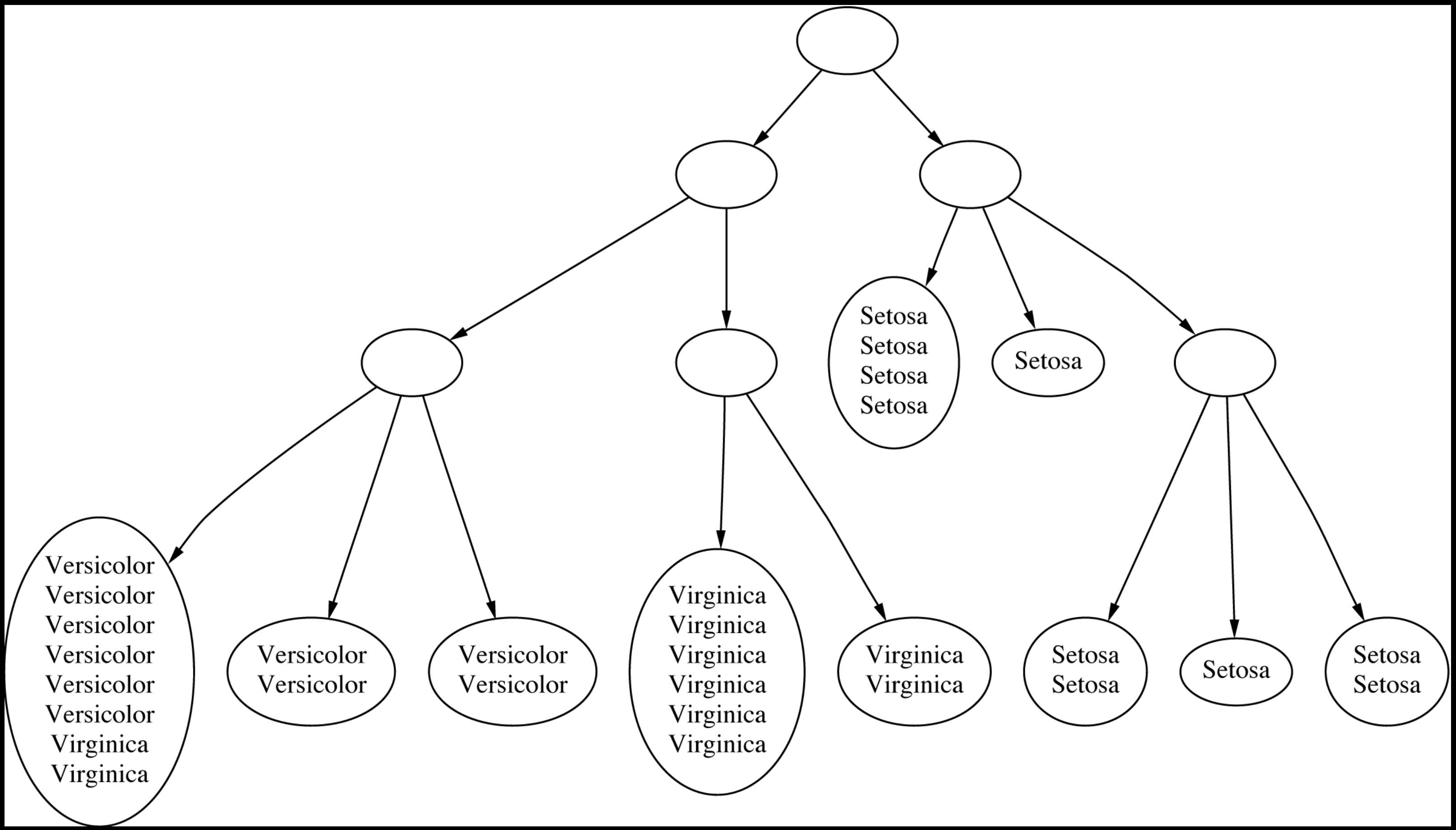
Final hierarchy



Example: the iris data (subset)



Clustering with cutoff



Category utility

- Category utility: quadratic loss function defined on conditional probabilities:

$$CU(C_1, C_2, \ldots, C_k) = \frac{\sum_l Pr[C_l] \sum_i \sum_j (Pr[a_i = v_{ij} | C_l]^2 - Pr[a_i = v_{ij}]^2)}{k}$$

- Every instance in different category $\Rightarrow$ numerator becomes

$$n - \sum_i \sum_j Pr[a_i = v_{ij}]^2 \longleftarrow \textit{maximum}$$

number of attributes

Numeric attributes

- Assume normal distribution:

$$f(a) = \frac{1}{\sqrt{(2\pi)}\sigma} \exp\left(-\frac{(a-\mu)^2}{2\sigma^2}\right)$$

- Then:

$$\sum_j Pr[a_i = v_{ij}]^2 \equiv \int f(a_i)^2 \, da_i = \frac{1}{2\sqrt{\pi}\sigma_i}$$

- Thus

$$CU(C_1, C_2, \ldots, C_k) = \frac{\sum_l Pr[C_l] \sum_i \sum_j (Pr[a_i = v_{ij} | C_l]^2 - Pr[a_i = v_{ij}]^2)}{k}$$

becomes

$$CU(C_1, C_2, \ldots, C_k) = \frac{\sum_l Pr[C_l] \frac{1}{2\sqrt{\pi}} \sum_i \left(\frac{1}{\sigma_{il}} - \frac{1}{\sigma_i}\right)}{k}$$

- Prespecified minimum variance
  - *acuity* parameter

Probability-based clustering

- Problems with heuristic approach:
  - Division by *k*?
  - Order of examples?
  - Are restructuring operations sufficient?
  - Is result at least *local* minimum of category utility?
- Probabilistic perspective ⇒
  seek the *most likely* clusters given the data
- Also: instance belongs to a particular cluster
  with a certain probability

Finite mixtures

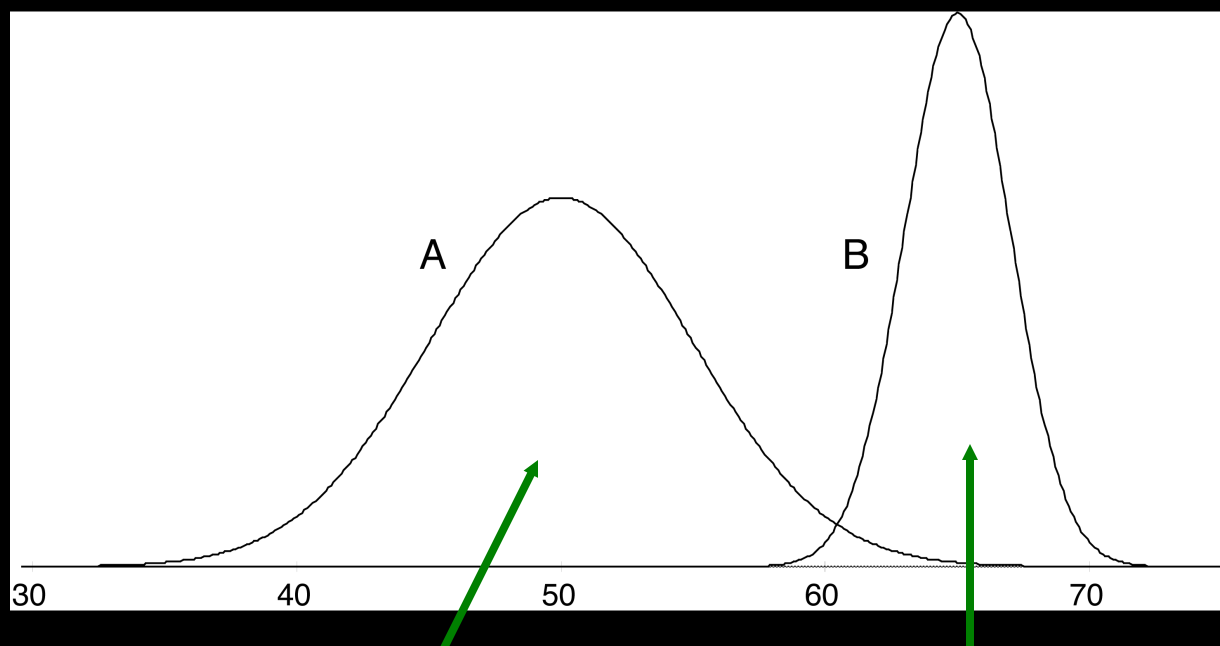
- Model data using a *mixture* of distributions
- One cluster, one distribution
  - governs probabilities of attribute values in that cluster
- *Finite mixtures* : finite number of clusters
- Individual distributions are normal (usually)
- Combine distributions using cluster weights

Two-class mixture model

data

A	51	B	62	B	64	A	48	A	39	A	51
A	43	A	47	A	51	B	64	B	62	A	48
B	62	A	52	A	52	A	51	B	64	B	64
B	64	B	64	B	62	B	63	A	52	A	42
A	45	A	51	A	49	A	43	B	63	A	48
A	42	B	65	A	48	B	65	B	64	A	41
A	46	A	48	B	62	B	66	A	48		
A	45	A	49	A	43	B	65	B	64		
A	45	A	46	A	40	A	46	A	48		

model



$\mu_A=50,\ \sigma_A=5,\ p_A=0.6$ $\mu_B=65,\ \sigma_B=2,\ p_B=0.4$

Using the mixture model

- Probability that instance x belongs to cluster A:

$$Pr[A|x]=\frac{Pr[x|A]Pr[A]}{Pr[x]}=\frac{f(x;\mu_A,\sigma_A)p_A}{Pr[x]}$$

with

$$f(x;\mu,\sigma)=\frac{1}{\sqrt{(2\pi)}\sigma}\exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)$$

- Probability of an instance given the clusters:

$$Pr[x|\text{the\_clusters}]=\sum_i Pr[x|\text{cluster}_i]Pr[cluster_i]$$

Learning the clusters

- Assume:
  - we know there are *k* clusters
- Learn the clusters ⇒
  - determine their parameters
  - I.e. means and standard deviations
- Performance criterion:
  - *probability of training data given the clusters*
- EM algorithm
  - finds a local maximum of the likelihood

EM algorithm

- EM = Expectation-Maximization
  - Generalize *k*-means to probabilistic setting
- Iterative procedure:
  - E "expectation" step:
    Calculate cluster probability for each instance
  - M "maximization" step:
    Estimate distribution parameters from cluster probabilities
- Store cluster probabilities as instance weights
- Stop when improvement is negligible

More on EM

- Estimate parameters from weighted instances

$$\mu_A = \frac{w_1 x_1 + w_2 x_2 + \ldots + w_n x_n}{w_1 + w_2 + \ldots + w_n}$$

$$\sigma_A = \frac{w_1 (x_1 - \mu)^2 + w_2 (x_2 - \mu)^2 + \ldots + w_n (x_n - \mu)^2}{w_1 + w_2 + \ldots + w_n}$$

- Stop when log-likelihood saturates

- Log-likelihood:

$$\sum_i \log \left(p_A Pr[x_i | A] + p_B Pr[x_i | B] \right)$$

Extending the mixture model

- More then two distributions: easy
- Several attributes: easy—assuming independence!
- Correlated attributes: difficult
  - Joint model: bivariate normal distribution with a (symmetric) covariance matrix
  - n attributes: need to estimate $n + n\,(n+1)/2$ parameters

More mixture model extensions

- Nominal attributes: easy if independent
- Correlated nominal attributes: difficult
  - Two correlated attributes $\Rightarrow$ $v_1 v_2$ parameters
- Missing values: easy
- Can use other distributions than normal:
  - "log-normal" if predetermined minimum is given
  - "log-odds" if bounded from above and below
  - Poisson for attributes that are integer counts
- Use cross-validation to estimate k !

Bayesian clustering

- Problem: many parameters $\Rightarrow$ EM overfits
- *Bayesian approach* : give every parameter a prior probability distribution
  - Incorporate prior into overall likelihood figure
  - Penalizes introduction of parameters
- Eg: Laplace estimator for nominal attributes
- Can also have prior on number of clusters!
- Implementation: NASA's AUTOCLASS

Discussion

- Can interpret clusters by using supervised learning
  - post-processing step
- Decrease dependence between attributes?
  - pre-processing step
  - E.g. use *principal component analysis*
- Can be used to fill in missing values
- Key advantage of probabilistic clustering:
  - Can estimate likelihood of data
  - Use it to compare different models objectively

Semisupervised learning

- *Semisupervised learning*: attempts to use unlabeled data as well as labeled data
  - The aim is to improve classification performance
- Why try to do this? Unlabeled data is often plentiful and labeling data can be expensive
  - Web mining: classifying web pages
  - Text mining: identifying names in text
  - Video mining: classifying people in the news
- Leveraging the large pool of unlabeled examples would be very attractive

Clustering for classification

- Idea: use naïve Bayes on labeled examples and then apply EM
  - First, build naïve Bayes model on labeled data
  - Second, label unlabeled data based on class probabilities (“expectation” step)
  - Third, train new naïve Bayes model based on all the data (“maximization” step)
  - Fourth, repeat 2^{nd} and 3^{rd} step until convergence
- Essentially the same as EM for clustering with fixed cluster membership probabilities for labeled data and #clusters = #classes

Comments

- Has been applied successfully to document classification
  - Certain phrases are indicative of classes
  - Some of these phrases occur only in the unlabeled data, some in both sets
  - EM can generalize the model by taking advantage of co-occurrence of these phrases
- Refinement 1: reduce weight of unlabeled data
- Refinement 2: allow multiple clusters per class

Co-training

- Method for learning from *multiple views* (multiple sets of attributes), eg:
  - First set of attributes describes content of web page
  - Second set of attributes describes links that link to the web page
- Step 1: build model from each view
- Step 2: use models to assign labels to unlabeled data
- Step 3: select those unlabeled examples that were most confidently predicted (ideally, preserving ratio of classes)
- Step 4: add those examples to the training set
- Step 5: go to Step 1 until data exhausted
- Assumption: views are independent

EM and co-training

- Like EM for semisupervised learning, but view is switched in each iteration of EM
  - Uses all the unlabeled data (probabilistically labeled) for training
- Has also been used successfully with support vector machines
  - Using logistic models fit to output of SVMs
- Co-training also seems to work when views are chosen randomly!
  - Why? Possibly because co-trained classifier is more robust