INTRODUCTION TO DATA MINING

Daniel Rodríguez, University of Alcalá

Universidad de Alcalá

OXFORD BROOKES UNIVERSITY

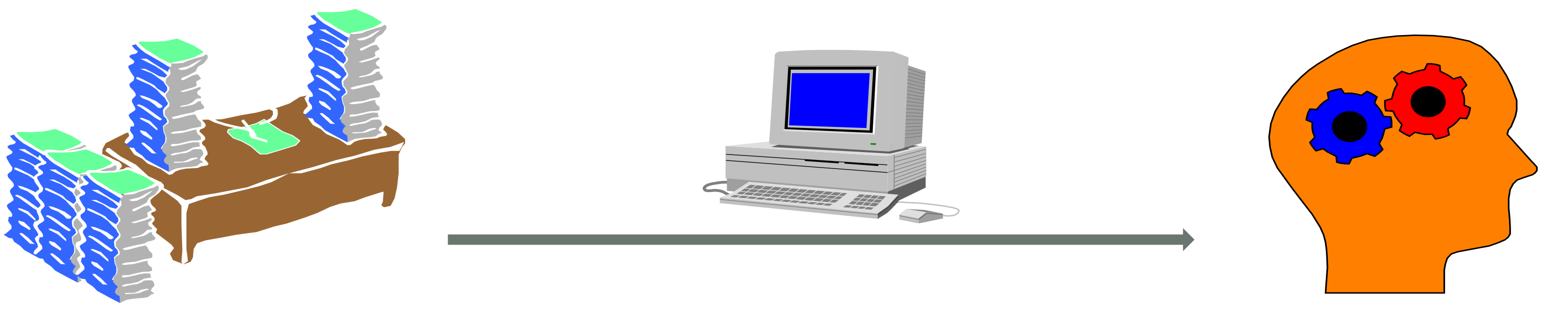
Outline

- Knowledge Discovery in Datasets
- Model Representation
  - Types of models
    - Supervised
    - Unsupervised
- Evaluation

- (Acknowledgement: Jesús Aguilar-Ruiz for some slides)

Knowledge Discovery in Datasets

KDD is the automatic extraction of non-obvious, hidden knowledge from large volumes of data.



Large amount of data

Data mining algorithms?

What knowledge? How to represent and use it?

Knowledge Discovery in Datasets

The non-trivial process of identifying valid, novel, potentially useful, and ultimately understandable patterns in data -

Fayyad, Piatetsky-Shapiro, Smyth (1996)

non-trivial process — multiple process

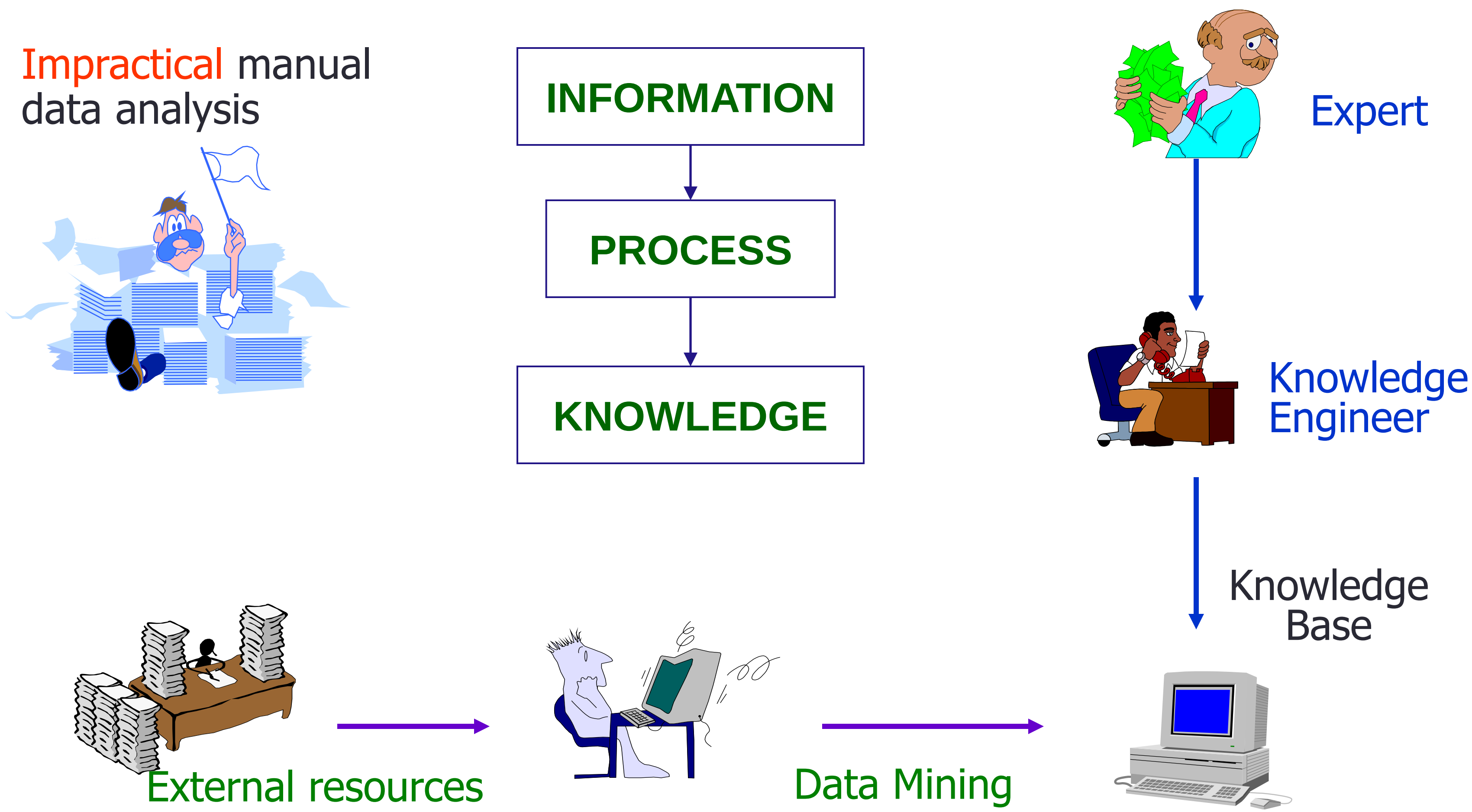
valid — Justified patterns/models

novel — Previously unknown

useful — Can be used

understandable — by human and machine

Knowledge Discovery in Datasets



KDD: Potential Applications

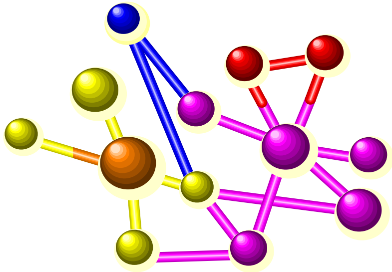
Business information

- Marketing and sales data analysis
- Investment analysis
- Loan approval
- Fraud detection
- etc.

Manufacturing information

- Controlling and scheduling
- Network management
- Experiment result analysis
- etc.

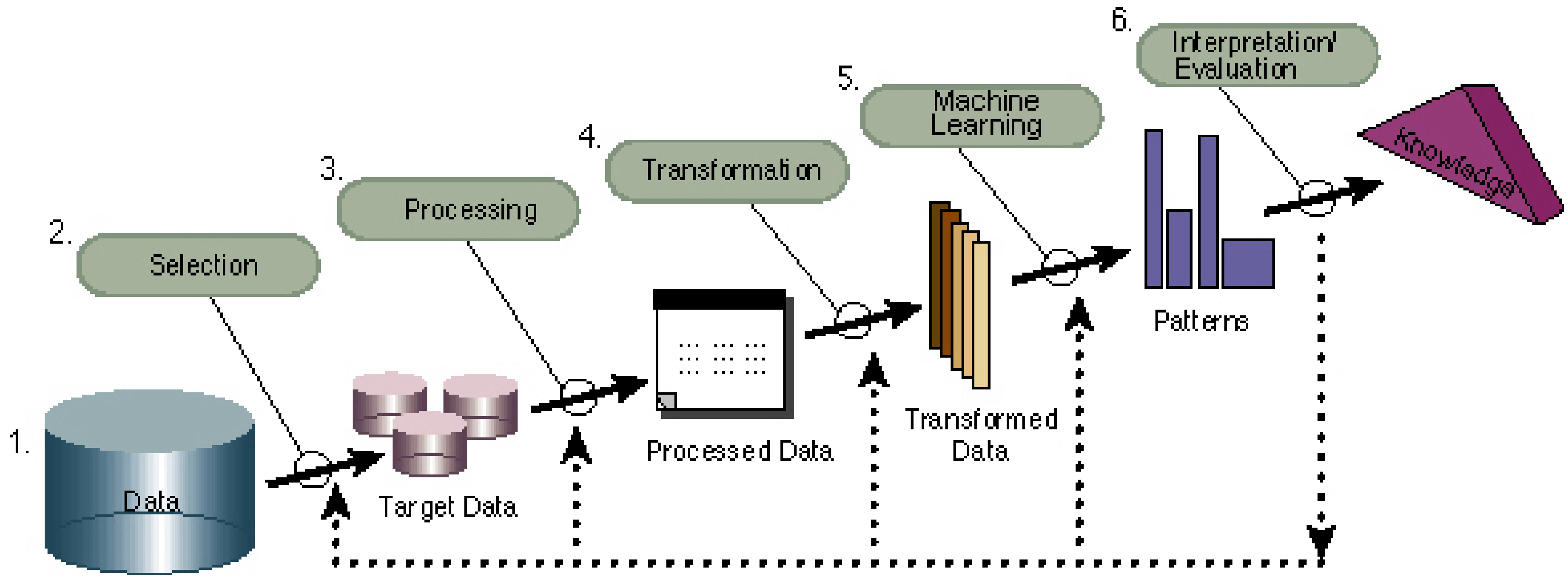
Scientific information



- Sky survey cataloging
- Biosequence Databases
- Geosciences: Quakefinder
- etc.

Personal information

KDD Process



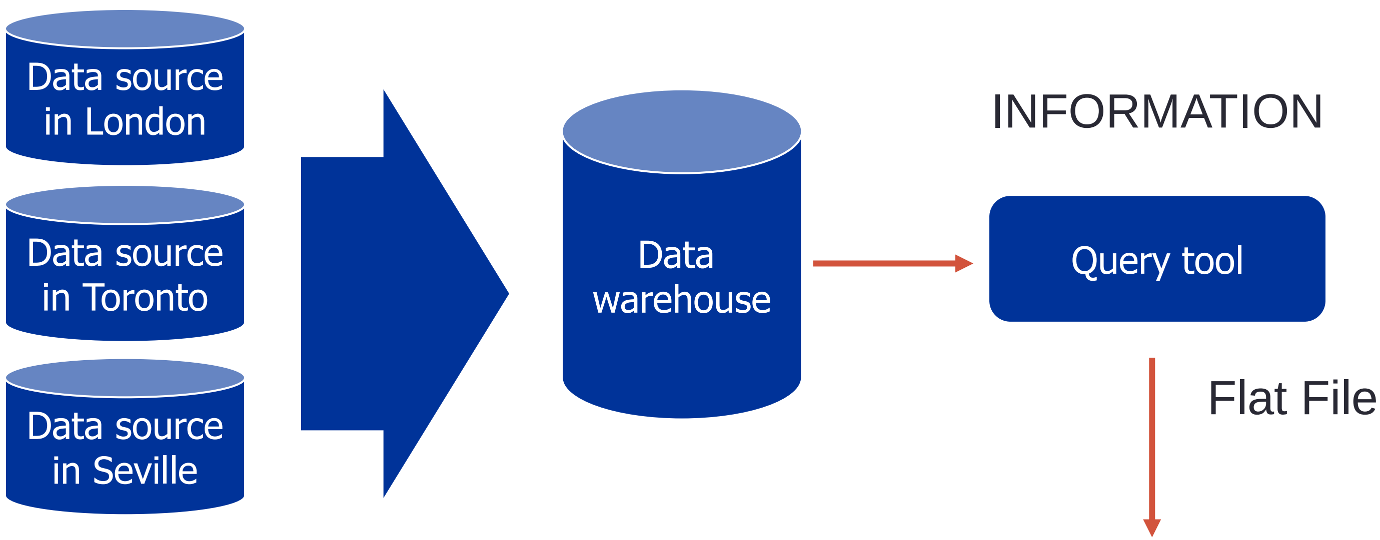
An Overview of the Steps That Compose the KDD Process

KDD Process

1. Data integration and recopilation
2. Selection, cleaning and transformation
   - Data
3. Machine Learning (data mining)
   - Patterns (e.g., classifiers, rules, etc.)
4. Evaluation and interpretation
   - Knowledge
5. Decision making

KDD Process: Integration & Recopilation

- Data warehousing, databases, data repositories from different and heterogeneous sources.
- We usually deal with a plain and flat matrix of data



customer

Cust-ID	name	address	age	income	credit-info	.
C1	Smith, Sandy	5463 E Hasting, Burnaby BC V5A 459, Canada	21	$27000	1	…

KDD Process: Selection, cleaning and transformation

- Removing Outliers
- Data Sampling (if there are too much data)
- Missing Values
- Removing redundant and irrelevant attributes (feature selection)
- Derive new attributes from existing ones
  - E.g., Population density from population and area
- Discretisation, normalisation, etc.

KDD Model Classification

- DM algorithms are traditionally divided into
  - Supervised learning which aims to discover knowledge for classification or prediction (predictive)
  - Unsupervised learning which refers to the induction to extract interesting knowledge from data (descriptive)

- New approaches also consider semisupervised learning: goal is classification but the input contains both unlabeled and labeled data.
  - Subgroup Discovery approaches generate descriptive rules are also half way between descriptive and predictive techniques.

Supervised learning - classifiers

- A classifier resembles a function in the sense that it attaches a value (or a range or a description) to a set of attribute values.
- Induce a classification model, given a database with
  - *m* instances (samples) characterized by
  - *n* predicted attributes, $A_1, \ldots, A_n$,
  - and the class variable, *C* (it is possible to have more than one class).

A_1	…	A_n	C
$a_{1,1}$	…	$a_{1,n}$	c_1
…	…	…	…
$a_{m,1}$		$a_{m,1}$	c_m

Supervised Techniques

- **Decision trees** are trees where each leaf indicates a class and internal nodes specifies some test to be carried out.
  - There are many tree-building algorithms such as C4.5 (Quilan) or ID3.
- **Rule induction**
  - **If** condition **then** $\text{class}_{\text{label}}$
  - **If** … **then** … **else if** … (hierarchical)
- **Lazy techniques** store previous instances and search similar ones when performing classification with new instances
  - **k-nearest neighbour** (k-NN) is a method for classifying objects based on closest training examples in the feature space.
- **Numeric prediction**
  - Regression Techniques
- **Neural Networks** are composed by a set of nodes (units, neurons, processing elements) where each node has input and output and performs a simple computation by its node function.
- **Statistical Techniques**
  - **Bayesian networks classifiers** assign a set of attributes $A_1, A_2,..., A_n$ to a class C_j such that $P(C_j \mid A_1, A_2,..., A_n)$ is maximal.
- **Meta-techniques**

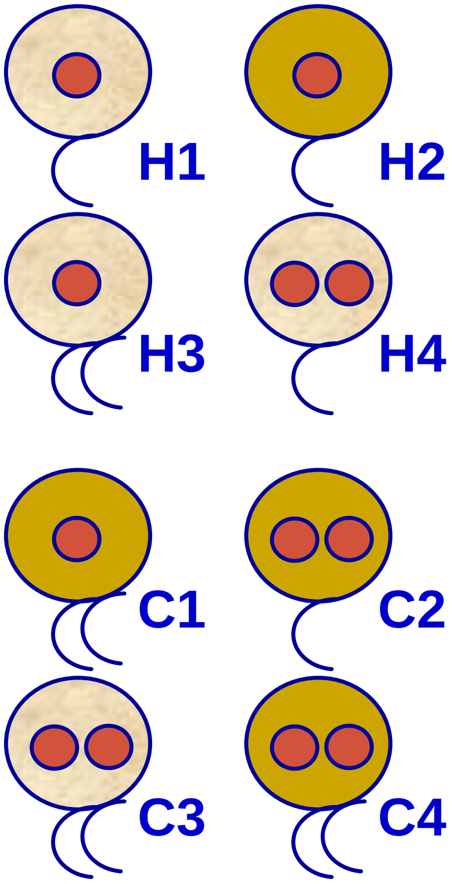
Unsupervised Techniques

- Association
  - Association Rules, APRIORI
    - E.g., rules among supermarket items
- Clustering
  - Tree clustering: join together objects (e.g., animals) into successively larger clusters, using some measure of similarity or distance.
  - Algorithms: K-Means, EM (Expectation Maximization)
- There is no ‘class’ attribute or class considered as another attribute

A_1	…	A_n
$a_{1,1}$	…	$a_{1,n}$
…	…	…
$a_{m,1}$		$a_{m,1}$

KDD Representation (Models)

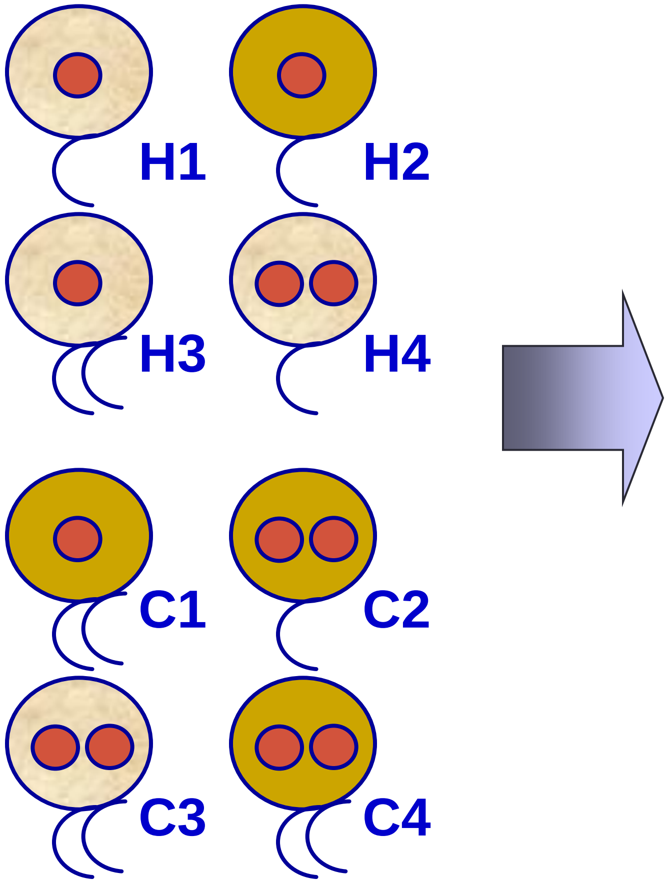
Dataset: Cancerous and Healthy Cells



	colour	#nuclei	#tails	class
H1	light	1	1	healthy
H2	dark	1	1	healthy
H3	light	1	2	healthy
H4	light	2	1	healthy
C1	dark	1	2	cancerous
C2	dark	2	1	cancerous
C3	light	2	2	cancerous
C4	dark	2	2	cancerous

Decision Rules

Mining with Decision Rules



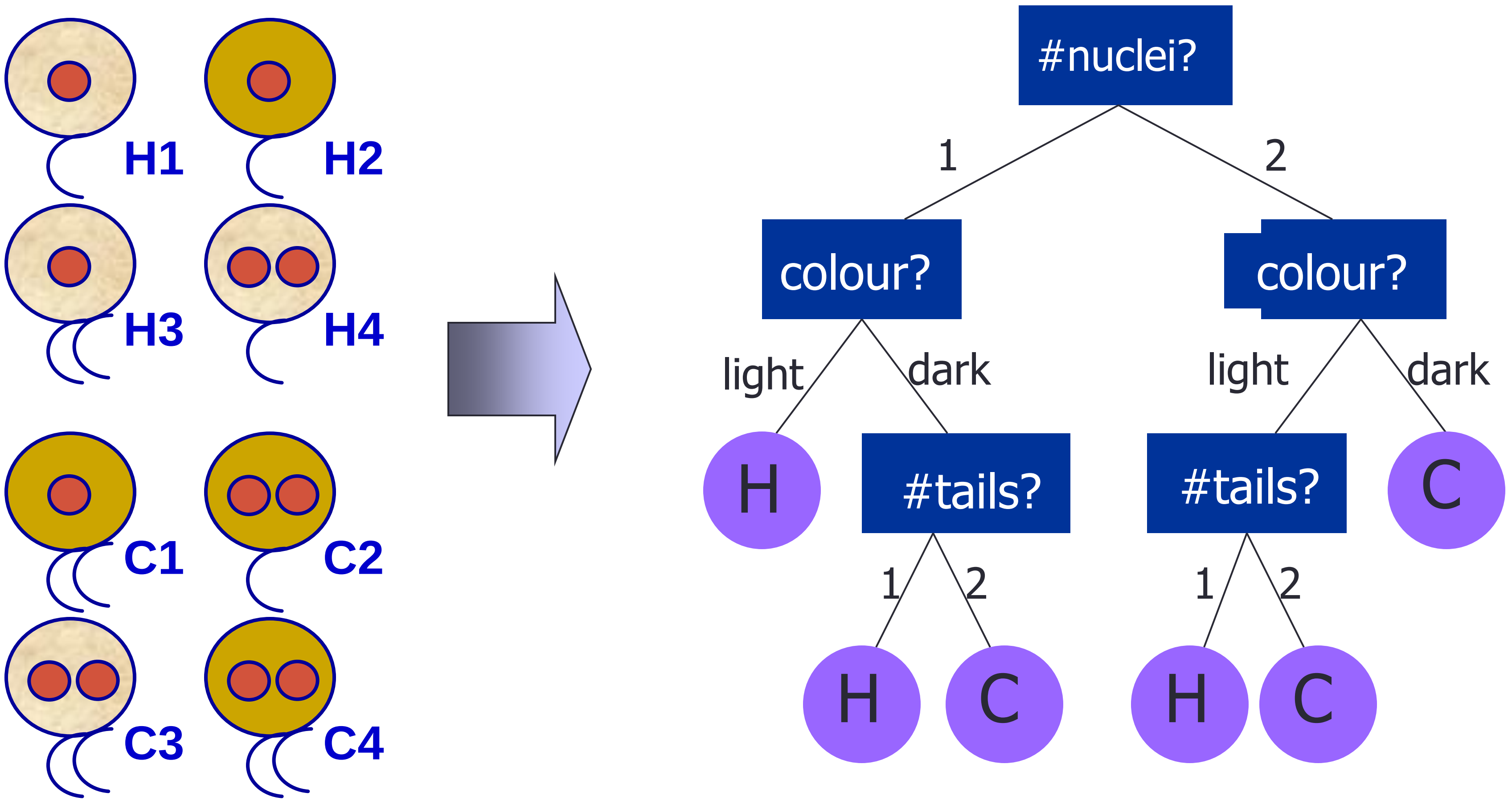
If colour = light **and** # nuclei = 1
Then cell = healthy

If #nuclei = 2 **and** colour = dark
Then cell = cancerours

(and 4 rules more)

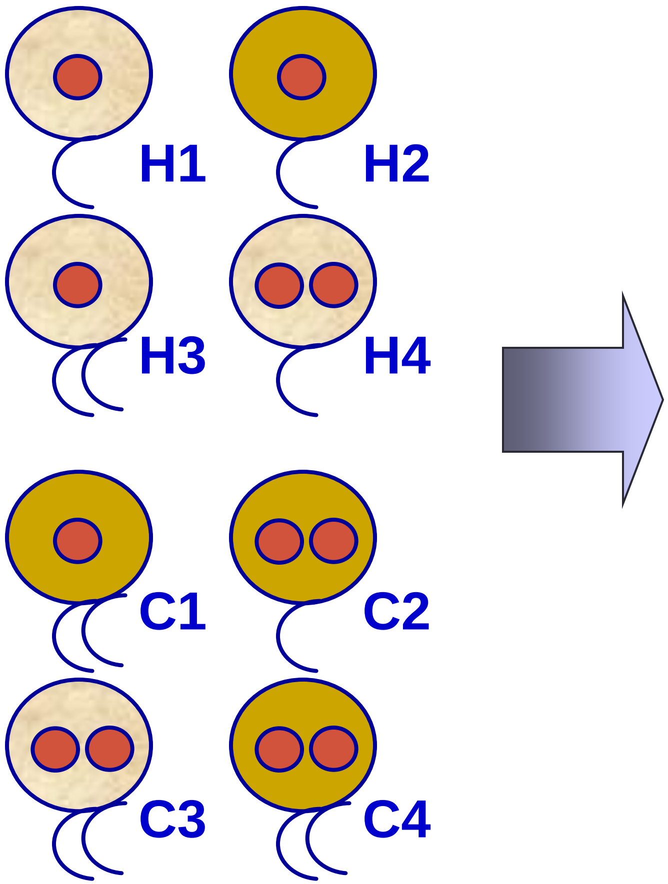
Decision Trees

Mining with Decision Trees



Hierarchical Decision Rules

Mining with Hierarchical Decision Rules



If colour = light **and** # nuclei = 1
Then cell = healthy

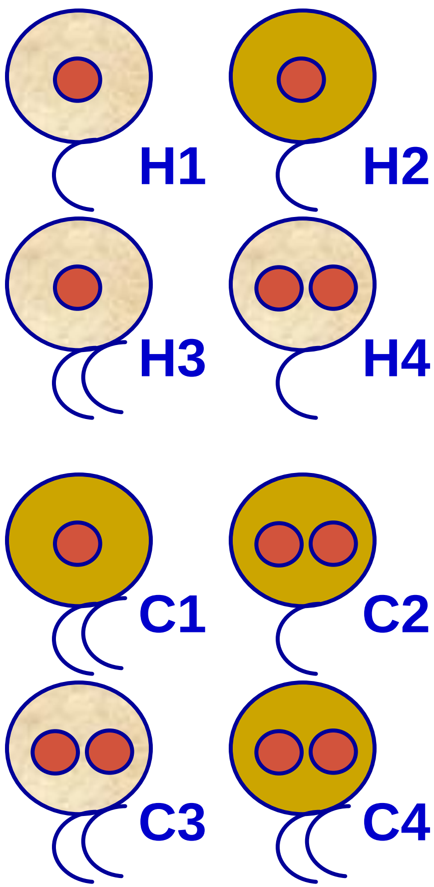
Else
  If #nuclei = 2 **and** colour = dark
  Then cell = cancerous

  Else
    If #tails = 1
    Then cell = healthy

    Else cell = cancerous

Association Rules

Mining with Association Rules



If colour = light
and # nuclei = 1
Then # tails = 1
(support = 25%; confidence = 50%)

association among A and B means that the presence of A in a record implies the presence of B in the same record

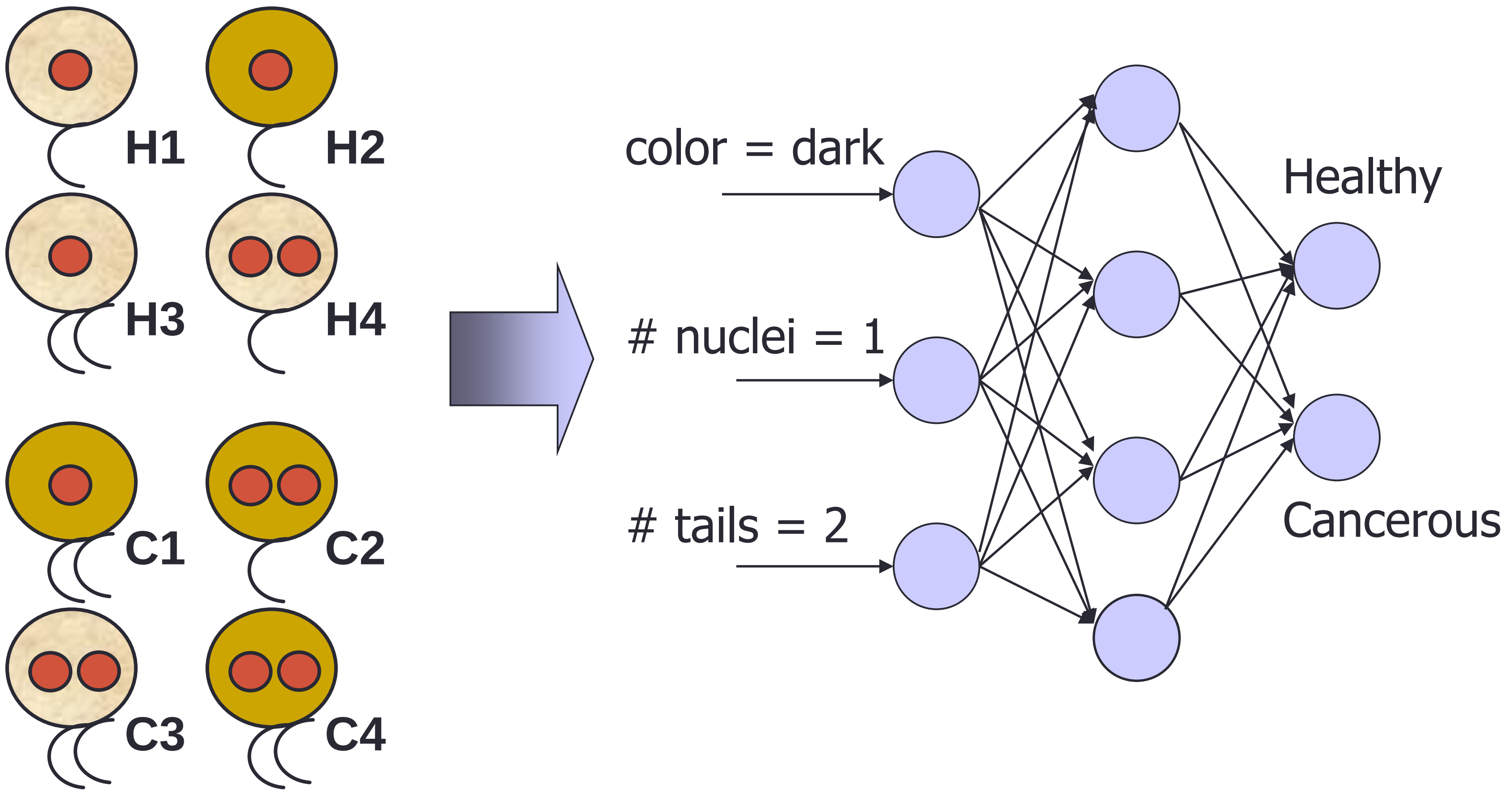
If # nuclei = 2
and cell = cancerous
Then # tails = 2
(support = 3/8%; confidence = 2/3%)

Support: the proportion of times that the rule applies.
Confidence: the proportion of times that the rule is correct

Apriori algorithm, Agrawal 1993

Neural Networks

Mining with Neural Networks

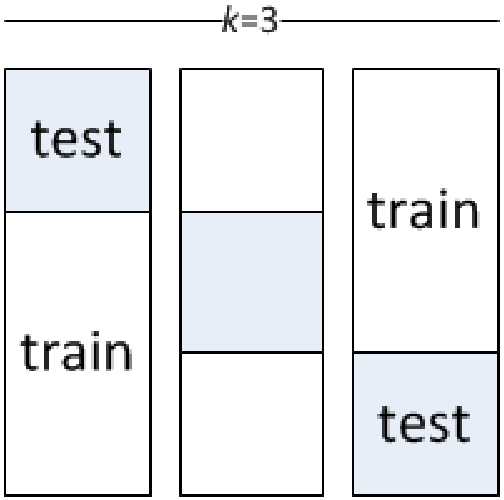


Evaluation

- Once we obtain the model with the training data, we need to evaluate it with some new data (testing data)
  - We cannnot use the the same data for training and testing
    - E.g., Evaluating a student with the exercises previouly solved
      - Student ‘s marks will be “optimistic” and we don’t know about student capability to generalise the learned concepts.
- Holdout approach consist of dividing the dataset into training (approx. 2/3 of the data) and testing (approx 1/3 of the data).
  - Problems: Data can be skewed, missing classes, etc. if randomly divided
- Stratification ensures that each class is represented with approximately equal proportions
  - E.g., if data contains aprox 45% of positive cases, the training and testing datasets should mantain similar proportion of positive cases.
- Holdout estimate can be made more reliable by repeating the process with different subsamples (*repeated holdout* method)
  - The error rates on the different iterations are averaged (overall error rate)

Cross-Validation

- *Cross-validation (CV)* avoids overlapping test sets
  - First step: split dataset (D) into k subsets of equal size $C_1, \ldots, C_k$
  - Second step: we construct a dataset $D_i = D - C_i$ used for training and test the accuracy of the classifier f_{D_i} on C_i subset for testing
    - Having done this for all k we estimate the accuracy of the method by averaging the accuracy over the k cross-validation trials
- Called *k-fold cross-validation*
  - Usually k=10
  - Subsets are generally stratified before the CV is performed
  - The error estimates are averaged to yield an overall error estimate



Evaluation Measures

- From the confusion matrix, we can obtain multiple measures about the goodness of a classifier.
- E.g. for a binary problem:

		Predicted		
		Positive	Negative	
Actual	Positive	**TP True Positive**	**FN False Negative (Type II error)**	TPrate=TP/(TP+FN) (Sensitivity, Recall)
	Negative	**FP False Positive (Type I error)**	**TN True Negative**	TNrate=TN/(FP+TN) (Specificity)
		PPV=TP/(TP+FP) Positive Predictive Value (Confidence, Precision)	NPV=TN/(FN+TN) Negative Predicted Value	Accuracy= TP+FP/(TP+TN+FP+FN)

Evaluation Measures

- Many times we need to combine the TP and FP to estimate the goodness of a classifier
  - For example, with imbalance data, the accuracy of a classifer needs to improve the percentage of the mayority class. In a binay problem and 50/50 distribution, we need improve accuracy over 50%. However if the distribution is 90/10, accuracy needs to be over 90%
- f −*measure* is an harmonic median of these proportions:

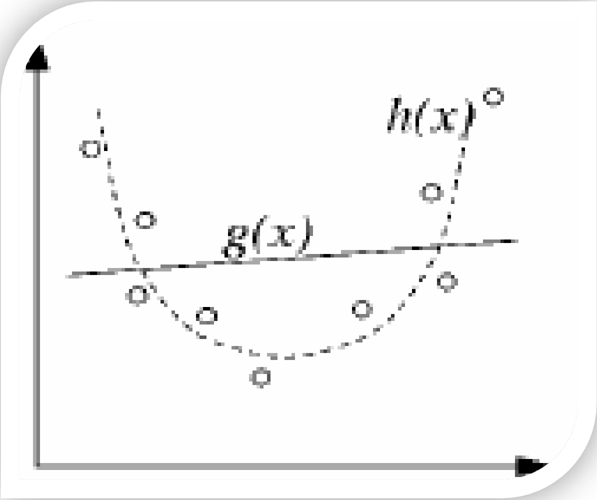
$$f - measure = \frac{2\,TP}{2\,TP + FP + FN}$$

- AUC (Area under the ROC)

Evaluation: Underfitting vs. Overfitting

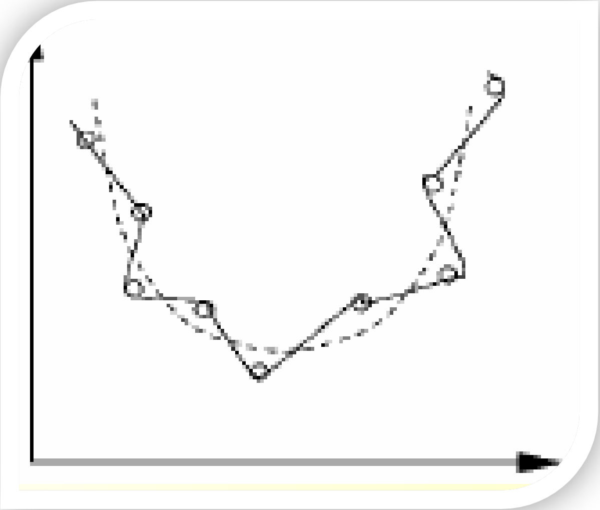
Underfitting

- Too simple model



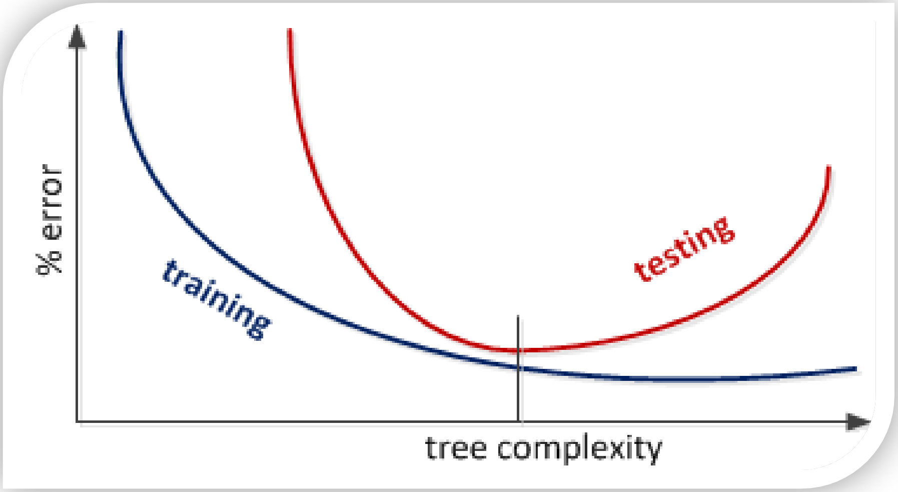
Overfitting

- Too complex model



Overfitting – Decision trees example

- Increasing the tree size, decreases the training and testing errors. However, at some point after (tree complexity), training error keeps decreasing but testing error increases
  - Many algorithms have parameters to determine the model complexity (e.g., in decision trees is the prunning parameter)



Evaluation: Cost

- In many cases to fail in one way is not the same as failing in the other one (Type I error vs. Type II error).
- Cost can be considered when inducing the model from the data using the cost data.
  - Cost sensitive classifiers
  - Metacost (Dominos, 1999)
- Ej. Cost matrices
  - by default –left- and considering cost –right-:

		Predicted	
		Positive	Negative
Actual	Positive	**0**	**1**
	Negative	**1**	**0**

		Predicted	
		Positive	Negative
Actual	Positive	**0**	**15**
	Negative	**1**	**0**

Thanks!

- Questions?