Complex Data Modeling Using Machine Learning

C.F. Aliferis M.D., Ph.D.

3-13-02, 3-14-02

Discovery Systems Laboratory,
Department of Biomedical Informatics,
Vanderbilt University

Goals & Overview

Goals (conceptual understanding)

To be able to answer the following questions:

- What is *Machine Learning* (ML)? How is it different than *Statistics* and *Data Mining*?
- What are example applications of ML in: drug development, bioinformatics, and clinical problem areas?
- What is the difference between *supervised* and *unsupervised* ML methods?
- What is the theoretical basis of supervised Inductive ML?
- What is the role of search, optimization, and knowledge representation in ML?
- How can ML methods fail?

Goals (conceptual understanding) CNT'D

To be able to answer the following questions:

- What is and how *Decision Tree Induction* works?
- What is and how *K-Nearest Neighbors* work?
- What is and how *Genetic Algorithms* work?
- What is and how *Artificial Neural Networks* work?
- What is and how *Clustering* works?
- What is and how *Causal probabilistic Network Induction* works?

Goals (practical experience)

To get exposure to the following tools:

- Decision Tree Induction (See5) for classification and hypothesis generation
- Neural Networks (NevProp) for classification
- Causal Probabilistic Network Induction (Tetrad3 and/or BN Power Constructor) for hypothesis generation

Generalities

What is *Machine Learning* (ML)? How is it different than *Statistics* and *Data Mining*?

- Machine Learning is the branch of Computer Science (Artificial Intelligence in particular) that studies systems that learn.
- Systems that learn = systems that improve their performance in some problem solving tasks without having been programmed by humans how to solve the tasks.
- ML has practically replaced Knowledge Acquisition for building Decision Support ("Expert") Systems.

What is *Machine Learning* (ML)? How is it different than *Statistics* and *Data Mining*?

- Typical tasks:
  - ✓ image recognition,
  - ✓ medical diagnosis, medical prognosis, treatment recommendation,
  - ✓ elicitation of possible causal structure of problem domain,
  - ✓ game playing,
  - ✓ various optimization tasks,
  - ✓ prediction of structure or function of biomolecules,
  - ✓ text categorization,
  - ✓ identification of relevant variables, etc.

What is *Machine Learning* (ML)? How is it different than *Statistics* and *Data Mining*?

- Broadly speaking ML, DM, and Statistics have similar goals (modeling for classification and hypothesis generation or testing).

- **Statistics** has traditionally emphasized models that can be solved analytically (for example various versions of the Generalized Linear Model – GLM). To achieve this both restrictions in the expressive power of models and their parametric distributions are heavily used.
- **Data Mining** emphasizes very large-scale data analysis, trading off soundness for problem size.
- **Machine Learning** seeks to use computationally powerful approaches to learn very complex non- or quasi-parametric models of the data. Some of these models are closer to human representations of the problem domain per se (or of problem solving in the domain)

Indicative Example applications of ML

1. Drug R&D
   - Debouck C, Goodfellow PN. DNA microarrays in drug discovery and development.
     Nat Genet. 1999 Jan;21(1 Suppl):48-50. Review.
     PMID: 9915501
   - Dean PM, Zanders ED, Bailey DS.Industrial-scale, genomics-based drug design and discovery. Trends Biotechnol. 2001 Aug;19(8):288-92. PMID: 11451470
   - Zuegge J, Schneider G, Coassolo P, Lave T. Prediction of hepatic metabolic clearance: comparison and assessment of prediction models. Clin Pharmacokinet. 2001;40(7):553-63. Review. PMID: 11510631
   - Pintore M, Taboureau O, Ros F, Chretien JR. Database mining applied to central nervous system (CNS) activity. Eur J Med Chem. 2001 Apr;36(4):349-59.
     PMID: 11461760
   - Coulter DM, Bate A, Meyboom RH, Lindquist M, Edwards IR.
     Antipsychotic drugs and heart muscle disorder in international pharmacovigilance: data mining study. BMJ. 2001 May 19;322(7296):1207-9.
     PMID: 11358771

Indicative Example applications of ML

1. Drug R&D (CONT'D)
   - Peh KK, Lim CP, Quek SS, Khoh KH. Use of artificial neural networks to predict drug dissolution profiles and evaluation of network performance using similarity factor.
     Pharm Res. 2000 Nov;17(11):1384-8.
     PMID: 11205731
   - Dowell JA, Hussain A, Devane J, Young D.
     Artificial neural networks applied to the in vitro-in vivo correlation of an extended-release formulation: initial trials and experience.
     J Pharm Sci. 1999 Jan;88(1):154-60.
     PMID: 9874718
   - Bate A, Lindquist M, Edwards IR, Olsson S, Orre R, Lansner A, De Freitas RM.
     A Bayesian neural network method for adverse drug reaction signal generation.
     Eur J Clin Pharmacol. 1998 Jun;54(4):315-21. PMID: 9696956
   - Kurth C, Wegerer V, Degner D, Sperling W, Kornhuber J, Paulus W, Bleich S.
     Risk assessment of alcohol withdrawal seizures with a Kohonen feature map.
     Neuroreport. 2001 May 8;12(6):1235-8. PMID: 11338198
   - McGuire WL, Tandon AK, Allred DC, Chamness GC, Ravdin PM, Clark GM.
     Treatment decisions in axillary node-negative breast cancer patients. J Natl Cancer Inst Monogr. 1992;(11):173-80.
     PMID: 1627425

Indicative Example applications of ML

2. Bioinformatics
   - Prediction of Protein Secondary Structure
   - Prediction of Signal Peptides
   - Gene Finding and Intron/Exon Splice Site Prediction
   - Diagnosis using cDNA and oligonucleotide array gene expression data
   - Identification of molecular subtypes of patients with various forms of cancer

3. Clinical problem areas
   - Survival after Pneumonia (CAP)
   - Survival after Syncope
   - Diagnosis of Acute M.I.
   - Diagnosis of Prostate Cancer
   - Diagnosis of Breast Cancer
   - Prescription and monitoring in hemodialysis
   - Prediction of renal transplant graft failure

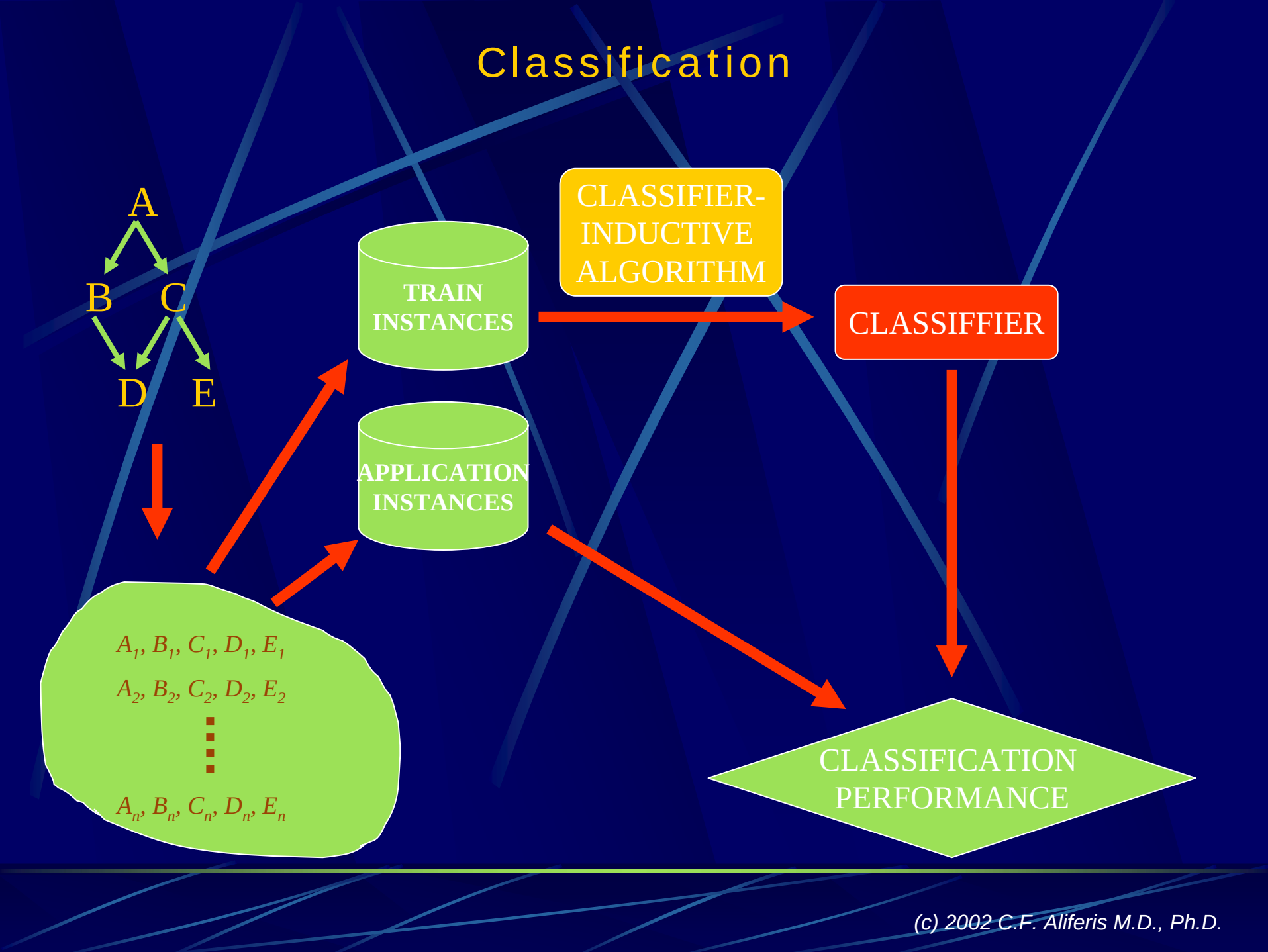
What is the difference between *supervised* and *unsupervised* ML methods?

- Supervised learning:
  - Give to the learning algorithm several instances of input-output pairs; the algorithm learns to predict the correct output that corresponds to some inputs (not only previously seen but also previously *unseen* ones ("generalization")).
  - Example: show to learning algorithm patient cases (i.e., findings vector and a correct diagnosis for each case); then the algorithm induces a classifier that can classify a previously unseen patient to the correct diagnostic category given the findings observed in that patient)

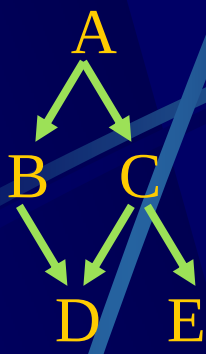
What is the difference between *supervised* and *unsupervised* ML methods?

- Unsupervised learning:
  - Discover the categories (or other structural properties of the domain)
  - Example: give the learning algorithm gene expression measurements of patients with Lung Cancer; the algorithm finds sub-types (“molecular profiles”) of patients that are very similar to each other, and different to the rest of the types. Or another algorithm may discover how various genes interact among themselves to determine development of cancer.

Classification

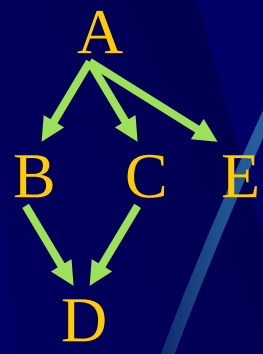


Discovery



TRAIN INSTANCES

STRUCTURE-INDUCTION ALGORITHM



A_1, B_1, C_1, D_1, E_1

A_2, B_2, C_2, D_2, E_2

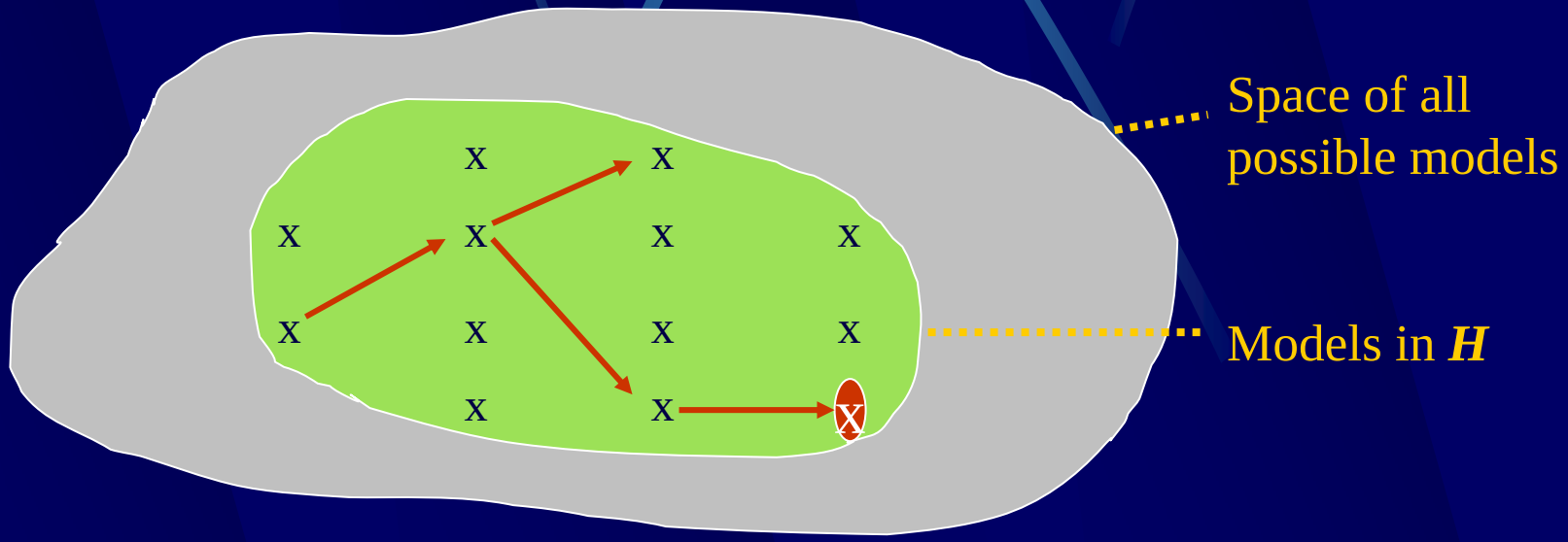
$\vdots$

A_n, B_n, C_n, D_n, E_n

PERFORMANCE

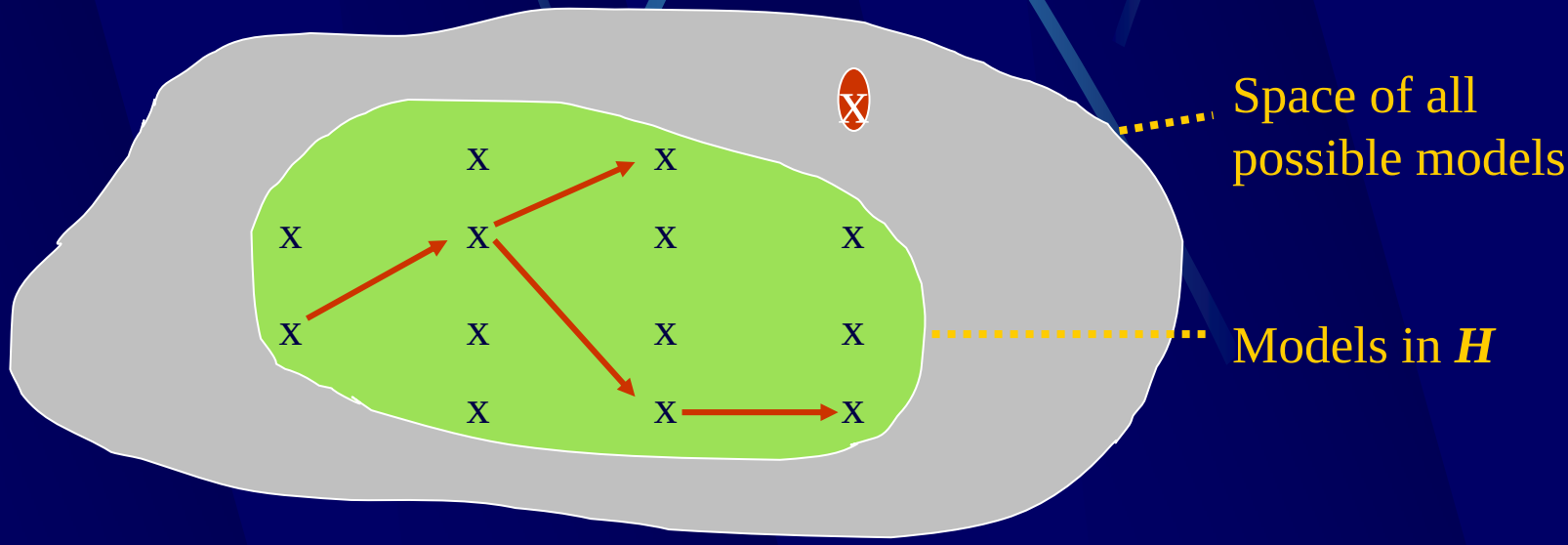
What is the theoretical basis of supervised Inductive ML?

- Inductive Machine Learning algorithms can be designed and analyzed using the following framework:
  - A language ***L*** in which we express models. The set of all possible models expressible in L constitutes our hypothesis space ***H***
  - A scoring metric ***M*** tells us how good is a particular model
  - A search procedure ***S*** helps us identify the best model in ***H***



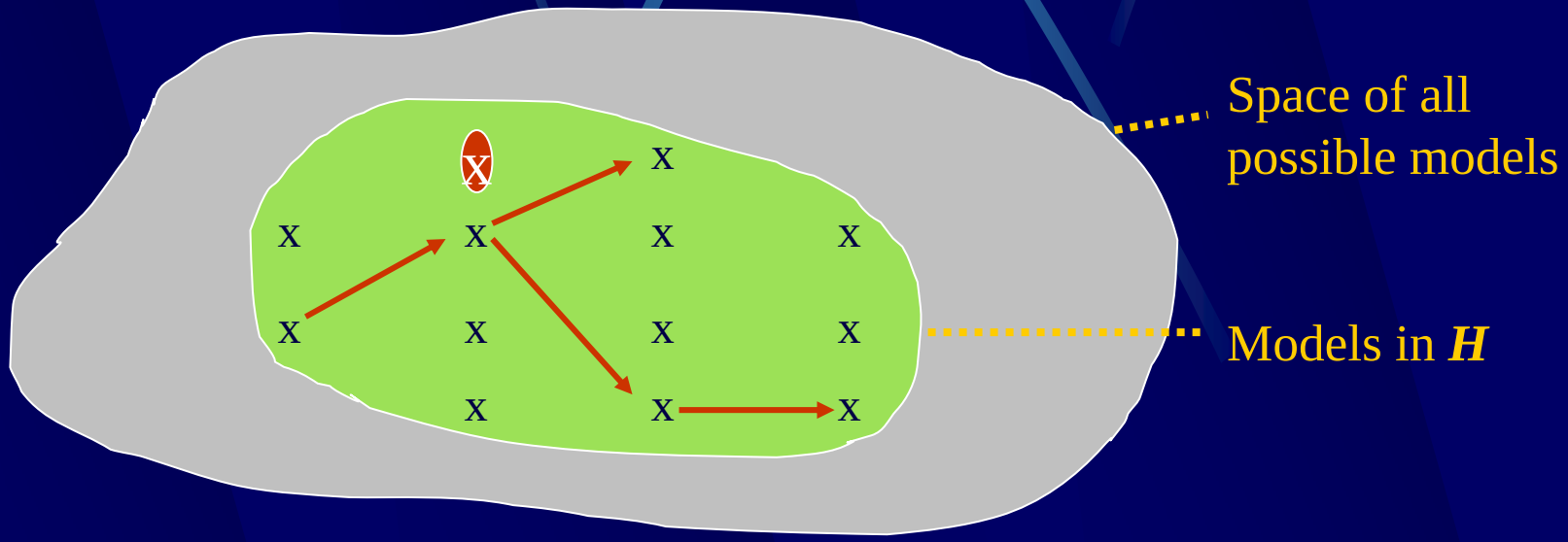
How can ML methods fail?

- Wrong language Bias: best model is not in *H*
- Example: we look for linear models, and the domain is non-linear



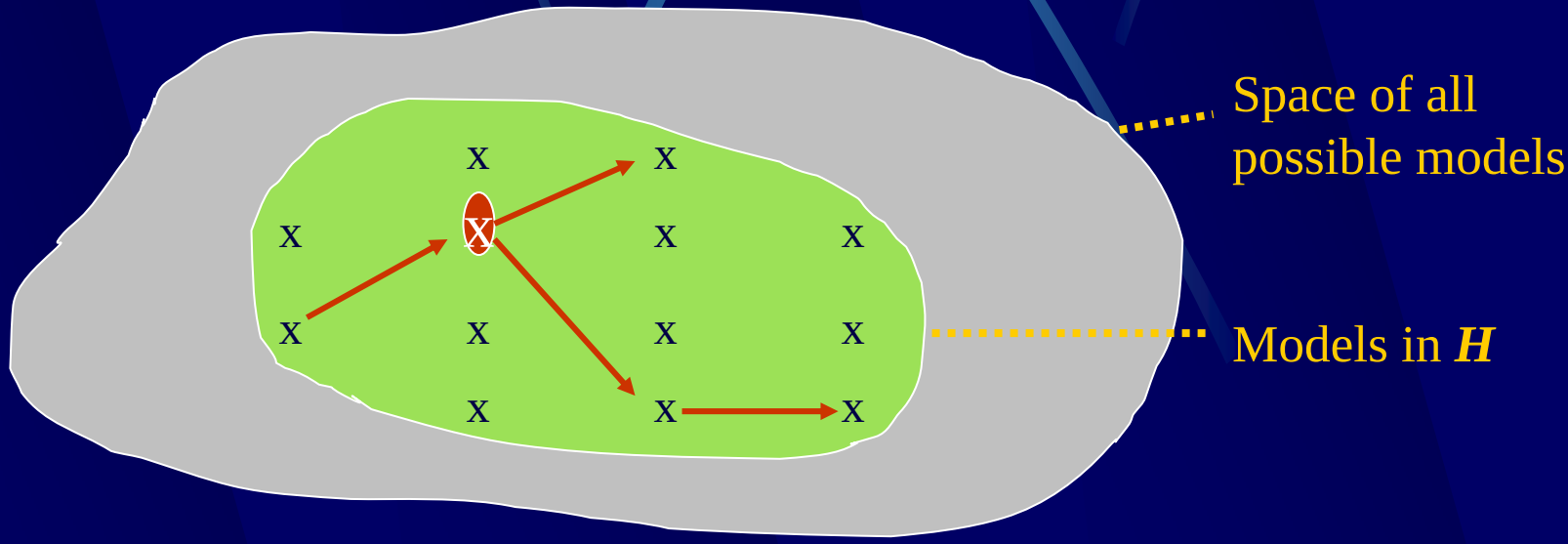
How can ML methods fail?

- Search Failure: best model is in *H* but search fails to examine it
- Example we use a steepest-ascent search in a multi-modal fitness landscape



How can ML methods fail?

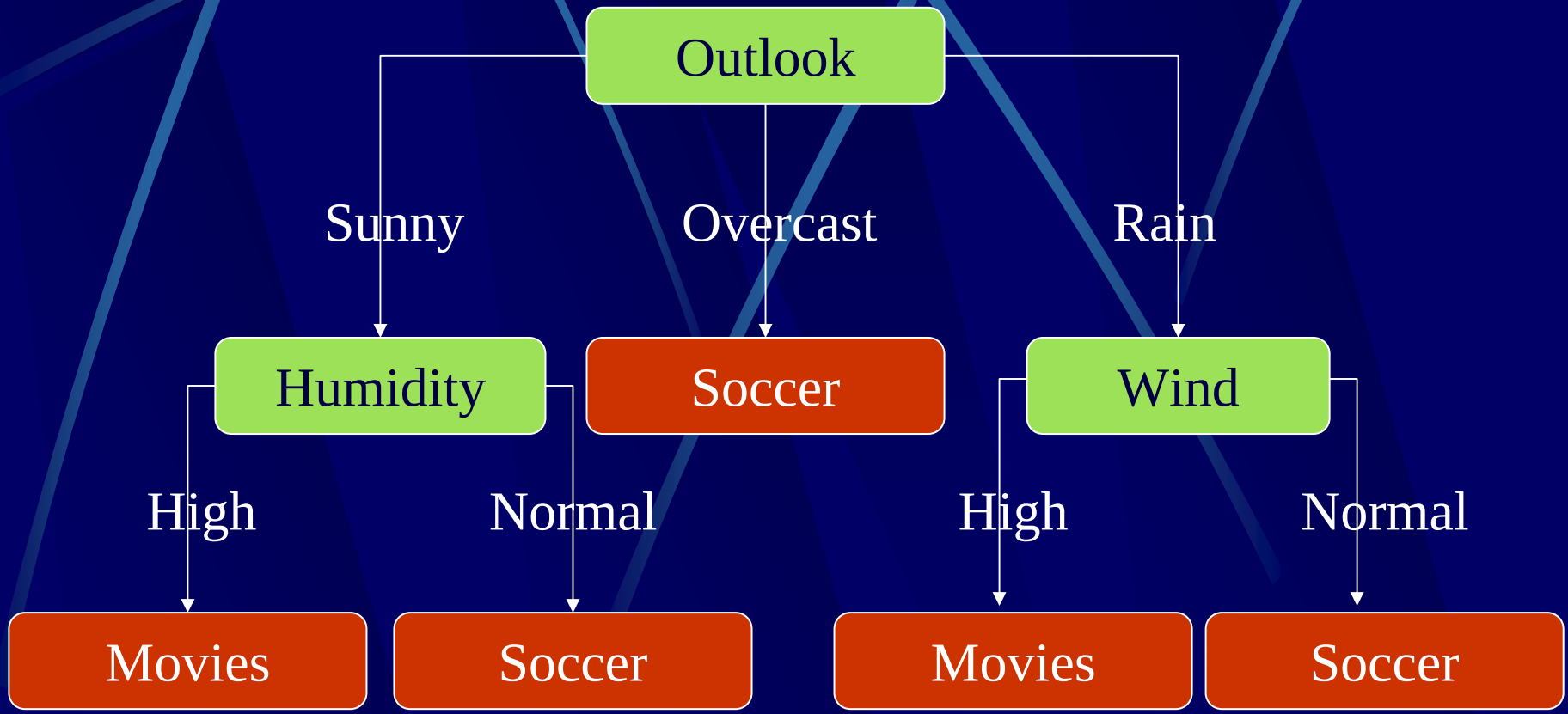
- Metric Failure: best model is in *H,* search finds it but is deemed to be worse than an inferior model
- Example: we do not incorporate prior information for models in *M*



Decision Tree Induction

Decision Tree Induction

- An example decision tree to solve the problem of how to spend my free time (play soccer or go to the movies?)



Decision Tree Induction

- The Decision Tree can be, alternatively, thought as a collection of rules:

R1: Outlook=*sunny,* Humidity=*high* → Decision=*Movies*

R2: Outlook=*sunny,* Humidity=*normal* → Decision=*Soccer*

R3: Outlook=*overcast* → Decision=*Soccer*

R4: Outlook=*rain,* Wind=*strong* → Decision=*Movies*

R5: Outlook=*rain,* Wind=*normal* → Decision=*Soccer*

Decision Tree Induction

- The Decision Tree can yet be thought of as *concept learning*: For example the concept “good day for soccer” is the disjunction of the following conjunctions:

(Outlook=*sunny* **and** Humidity=*normal)* **or**

(Outlook=*overcast*) **or**

(Outlook=*rain* **and** Wind=*normal*)

Decision Tree Induction

- Decision Trees can be automatically learned from data using either information-theoretic criteria or a measure of classification performance.
- The induction procedure is very simple in principle:

1. Start with an empty tree
2. Put at the root of the tree the variable that best classifies the training examples
3. Create branches under the variable corresponding to its values
4. Under each branch repeat the process with the remaining variables

Notes:

- "best classifies" can be determined on the basis of maximizing homogeneity of outcome in the resulting subgroups, cross-validated accuracy, best-fit of some linear regressor, etc.

Decision Tree Induction

Notes (CNT'D):

- DTI is best for:
  - Discrete domains
  - Target function has discrete outputs
  - Disjunctive/conjunctive descriptions required
  - Training data may be noisy
  - Training data may have missing values
- DTI can represent any finite discrete-valued function
- Extensions for continuous variables do exist
- Search is clearly greedy and thus can be trapped in local minima
- DTI is very sensitive to high feature-to-sample ratios; when many features contribute a little to calssification DTI does not do well
- DT models are highly intuitive, and easy to explain and use even without computing equipment available

K-Nearest Neighbors

K-Nearest Neighbors

- Assume we wish to model patient response to treatment; suppose we have seen the following cases:

Patient#	Treatment type	Genotype	Survival
1	1	1	1
2	1	2	2
3	1	1	1
4	1	2	2
5	2	1	2
6	2	2	1
7	2	1	2
8	2	2	1

(Notice the very strong interaction between treatment and genotype in determining survival)

K-Nearest Neighbors

- Say we want to predict outcome for a patient *i* that received treatment 1 and is of genotype class 2. KNN searches for the K most similar cases in the training data base (using Euclidean Distance or other similarity metric):

$$ED(\mathbf{x_i},\mathbf{x_j}) = \sqrt{\sum_k (x_{i,k} - x_{j,k})^2}$$

- For example patient #1 and the new patient have ED=

Patient#	Treatment type	Genotype	Survival
1	1	1	1
i	1	2	**?**

$$ED = \sqrt{((1-1)^2 + (1-2)^2)} = 1$$

K-Nearest Neighbors

- Similarly the distances of case *i* to all training cases are:

Patient#	ED(Patient#, P*i*)	Survival
1	1	1
2	0	2
3	1	1
4	0	2
5	1.4	2
6	1	1
7	1.4	2
8	1	1

- Now let's rank training cases according to distance to case *i*

K-Nearest Neighbors

Patient#	ED(Patient#, P*i*)	Survival
2	0	2
4	0	2
3	1	1
1	1	1
6	1	1
8	1	1
5	1.4	2
7	1.4	2

As we can see the training case most similar to *i* has outcome 2. The 2 training cases most similar to *i* have a median outcome 2. The 3 training cases most similar to *i* have a median outcome 2, and so on. We say that for K=1 the KNN predicted value is 2, for K=2 the predicted value is 2, and so on.

K-Nearest Neighbors

To summarize:

- KNN is based on a case-based reasoning framework.
- It has good asymptotic properties for K>1.
- It is straightforward to implement (of course care has to be given to variable encoding, variable relevance, and distance metric); efficient encoding is not easy since it requires specialized data structures.
- It is used in practice as:
  - a baseline comparison for new methods
  - component algorithm for “wrapper” feature selection methods
  - Non-parametric density estimator

Genetic Algorithms

Genetic Algorithms

- Evolutionary Computation (Genetic Algorithms & Genetic Programming) is motivated by the success of evolution as a robust method for adaptation found in nature*
- The main algorithm is very simple:

1. Generate randomly a population P of p hypotheses
2. Compute the fitness of each member of P, h_i
3. Repeat
   a. Create a random sample Ps from P by choosing each h_i with probability proportional to the relative fitness of h_i to the total fitness of all h_j
   b. Augment Ps with cross-over offspring of the remaining hypotheses chosen with same probability as in step #4
   c. Change members of Ps at random by bit-mutations
   d. Replace P by Ps and compute new fitness of each member of P
4. Until enough generations have been created or a good enough hypothesis have been generated
5. Return best hypothesis

Genetic Algorithms

- The population size, cross-over rate, and mutation rate are parameters that are set empirically
- Representation of hypotheses in GAs is typically a bitstring so that the mutation and cross-over operations can be achieved easily.
- E.g., consider encoding clinical decision-making rules:

variable1: fever {yes, no}

variable2: x_ray {positive, negative}

variable3: diagnosis {flu, pneumonia}

Rule1: fever=yes and x_ray=positive → diagnosis=pneumonia

Rule2: fever=no and x_ray=negative → diagnosis= flu or pneumonia

Bitstring representation:

R1:	1 0	1 0	0 1
R2:	1 0	0 1	1 1

(note: we can constrain this representation by using less bits, the fitness function, and syntactic checks)

Genetic Algorithms

Let's cross-over these rules at a random point:

R1:	1 0	‖	1 0	0 1
R2:	1 0	‖	0 1	1 1

Gives:

R1':	1 0	0 1	1 1
R2':	1 0	1 0	0 1

And mutation at two random bits may give:

R1'':	1 0	0 **0**	1 1
R2'':	1 **1**	1 0	0 1

Which is interpreted as:

Rule1'': fever=yes and x_ray=negative → diagnosis=flu or pneumonia

Rule2'': fever=unknown and x_ray=positive → diagnosis=pneumonia

Genetic Algorithms

Notes:

- There exist many variations of how to do cross over, how to select hypotheses for mutation/cross-over, how to isolate subpopulations, etc.
- Although it may appear at first that the process of finding better hypotheses relies totally on chance, this is not the case. Several theoretical results (most famous one being the "Schema Theorem" prove that exponentially more better-fit hypotheses are being considered than worse-fit ones (to the number of generations).
- Furthermore, due to the discrete nature of optimization local minima will trap the algorithm less, but also it becomes more difficult to find the global optimum.
- It has been shown that GA perform an implicit parallel search in hypotheses templates without explicitly generating them ("Implicit Paralellism".

Genetic Algorithms

Notes:

- Serious caveat: GAs are “black box” optimizers; sometimes they are applied appropriately to learn models when no better alternative can be reasonably found, and when they do have a chance for finding a good solution. There exist cases however when much faster and provably sound algorithms can (and should) be used, as well as cases where heuristic search is provably not capable of finding a good solution (and thus should not be used).

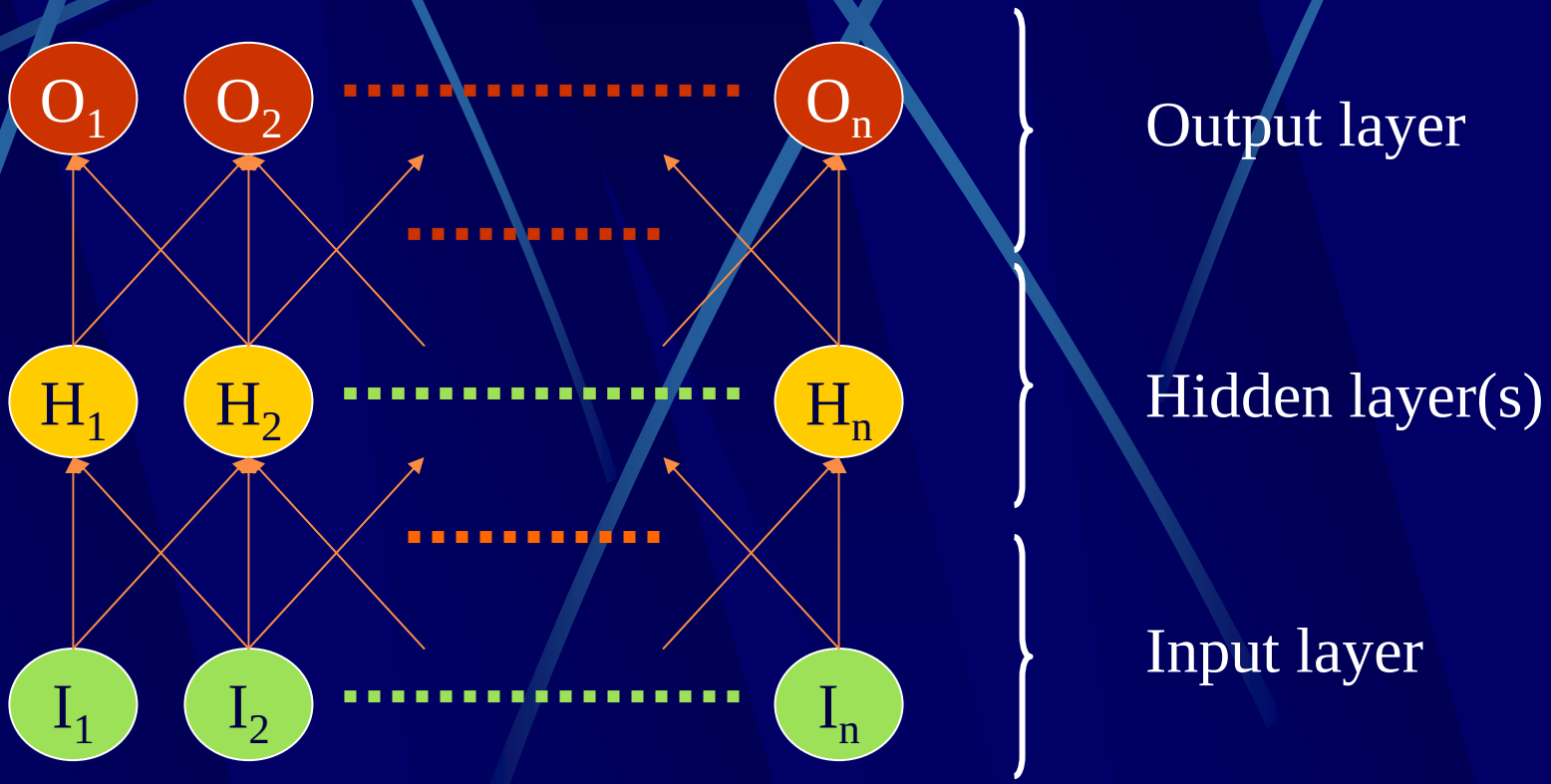
  Consider, for example, the problems of finding the shortest path between two cities in a map, of sorting numbers, of solving a linear program, of fitting a linear model, etc. for all these cases better and faster special-purpose algorithms exist and should be used instead.

Artificial Neural Networks

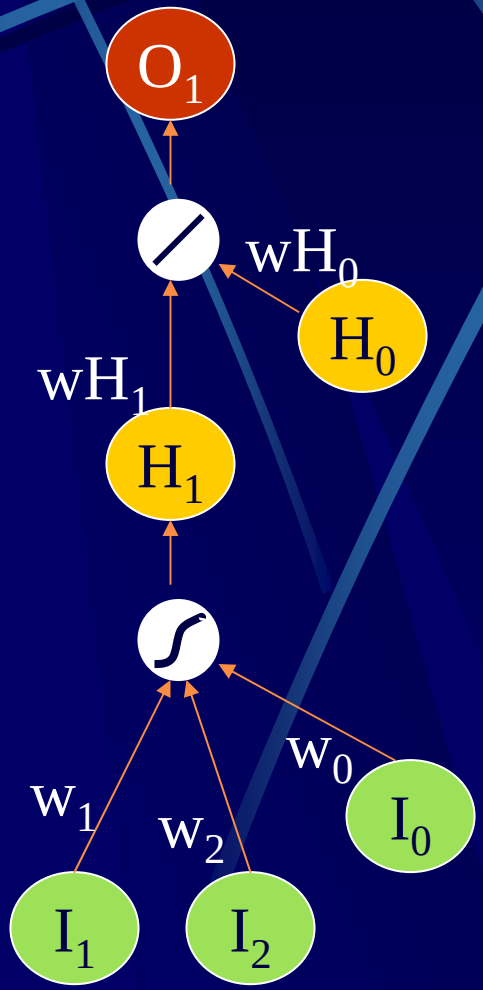
Artificial Neural Networks

- A very large, family of supervised and unsupervised learning methods motivated by the way biological neural systems are organized*.
- Among the most effective and robust learning methods yet devised
- They have excellent documented results in a wide variety of problem areas including: image recognition, text categorization, medical diagnosis, structural genomic and proteomic classification, dynamic control, drug-dosage adjustment, cluster discovery, speech recognition, etc.
- On the other hand it is almost impossible to understand why a network performs the way it does (i.e., *what* it has learned) and *why* it gives the output that it does.
- We will primarily discuss here a very important and widely-used class of ANNs: multi-layer feed-forward (acyclic) networks trained with gradient descent (Backpropagation algorithm)

Artificial Neural Networks



Artificial Neural Networks



$$O_1 = wH_0 + wH_1 \times H_1$$

$$H_1 = 1/(1 + e^{-(w_0 + w_1 \times I_1 + w_2 \times I_2)})$$

Artificial Neural Networks

- Backpropagation algorithm:

1. Initialize all weights to small random numbers
2. Repeat
3. Propagate input forward:
   a. Compute output of every unit given an input vector
4. Propagate errors backward:
   a. Calculate error for each output unit
   b. Calculate error for each hidden layer unit
   c. Update each network weight
5. Until error small enough or other stopping criterion (independent test set error, number of train iterations, etc.)

Artificial Neural Networks

- Notes:
  - The choice of a smooth transfer function allows differentiation of the error function and thus derivation of an easy to compute stochastic gradient descent optimization rule employed in BP
  - BP may be trapped in local minima; in practice this is not a major problem (because of stochastic gradient descent, momentum, and very high-dimensional spaces)
  - A ANN with three layers of units can represent any function to arbitrary accuracy
  - ANNs have very good generalization behavior
  - Overfitting can be avoided by empirical (generalization error, weight decay) or theoretical means
  - 1-of-n encoding preferred over 1-unit encoding for multiple-category output

Clustering

Clustering

- Unsupervised class of methods
- Basic idea: group similar items together and different items apart
- Countless variations:
  - of what constitutes “similarity” (may be distance in feature space,may be other measures of association),
  - of what will be clustered (patients, features, time series, cell-lines, combinations thereoff, etc.)
  - of whether clusters are “hard” (no multi-membership) or “fuzzy”
  - of how clusters will be build and organized (partitional, agglomerative, non-hierarchical methods)
- Uses:
  - Taxonomy (e.g., identify molecular subtypes of disease)
  - Classification (e.g., classify patients according to genomic information)
  - Hypothesis generation (e.g., if genes are highly “co-expressed” then this may suggest they are in same pathway)

Clustering

- **K-means clustering**: We want to partition the data into *k* most-similar groups

1. Choose *k* cluster centers ("centroids") to coincide with *k* randomly chosen patterns (or arbitrarily-chosen points in the pattern space)
2. Repeat
3. Assign each pattern in data to cluster with the closest centroid
4. Recompute new centroids
5. Until convergence (i.e., few or no re-assignments or small decrease in error function such as total sum of squared errors of each pattern in a cluster from centroid of that cluster)

Variations:

- selection of good initial partitions
- Allow splitting/merging of resulting clusters
- Various similarlity measures and convergence criteria

Clustering (k-means)

e.g., (*K*=2)

A B	C D	E F
2 3	9 10	11 12

Step 1: (arbitrarily)

[A B	C D]	[E F]

Centroid1=6, centroid2=11.5

Step 2:

[A B]	[C D	E F]

Centroid1=2.5, centroid2=10.25

-------(algorithm stops)--------

Clustering

Agglomerative Single Link:

1. Start with each pattern belonging to its own cluster
2. Repeat
3. Join these two clusters that have the smallest pair-wise distance
4. Until all patterns are in one cluster

Note:

- Inter-cluster distance between clusters A and B is computed as the minimum distance of all pattern pairs (a,b) s.t. a belongs to A and b to B

Clustering (ASL)

e.g.,

A	B	C	D
1	2	5	7

Step 1:

[A] [B] [C] [D] smallest distance [A] [B]=1

Step 2:

[A B] [C] [D] smallest distance [C] [D]=2

Step 3:

[A B] [C D] smallest distance [A B] [C D]=3

Step 4:

[A B C D] smallest distance [C][D]=2

-------(algorithm stops)--------

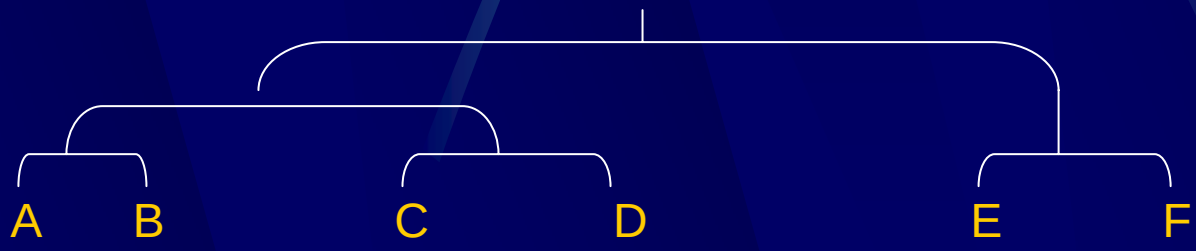
Clustering (ASL)

e.g.,

A	B	C	D	E	F
1	2	5	7	11	12

Step 1:	[A] [B]	[C]	[D]	[E]	[F]	smallest distance [A] [B]=1 OR [E] [F]=1
Step 2:	[A B]	[C]	[D]	[E]	[F]	smallest distance [E] [F]=1
Step 3:	[A B]	[C]	[D]	[E F]		smallest distance [C] [D]=2
Step 4:	[A B]	[C D]		[E F]		smallest distance [A B] [C D]=3
Step 5:	[A B C D]			[E F]		smallest distance [A B C D] [E F]=4
Step 6:	[A B C D E F]					-------(algorithm stops)--------

Schematic representation via the “dendrogram”:



Clustering

Agglomerative Complete Link:

1. Start with each pattern belonging to its own cluster
2. Repeat
3. Join these two clusters that have the smallest pair-wise distance
4. Until all patterns are in one cluster

Note:

- Inter-cluster distance between clusters A and B is computed as the maximum distance of all pattern pairs (a,b) s.t. a belongs to A and b to B

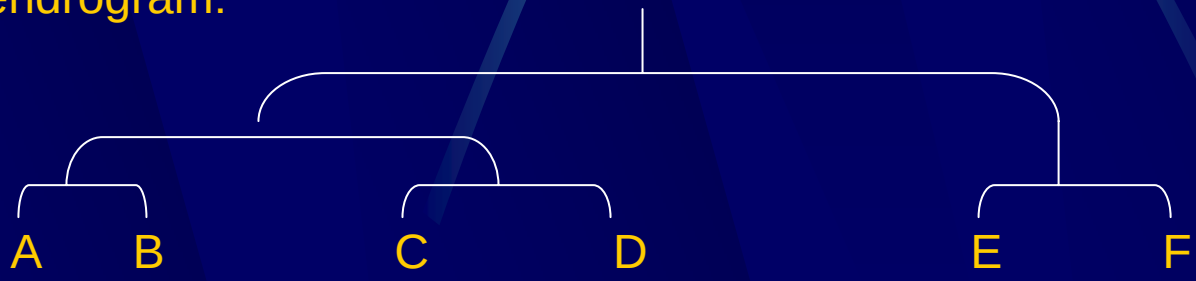
Clustering (ACL)

e.g.,

A	B	C	D	E	F
1	2	5	7	11	12

Step 1:	[A]	[B]	[C]	[D]	[E]	[F]	smallest distance [A] [B]=1 OR [E] [F]=1
Step 2:	[A	B]	[C]	[D]	[E]	[F]	smallest distance [E] [F]=1
Step 3:	[A	B]	[C]	[D]	[E	F]	smallest distance [C] [D]=2
Step 4:	[A	B]	[C	D]	[E	F]	smallest distance [A B] [C D]=6
Step 5:	[A	B	C	D]	[E	F]	smallest distance [A B C D] [E F]=11
Step 6:	[A	B	C	D	E	F]	-------(algorithm stops)--------

With dendrogram:



Clustering

- Clustering has been very prevalent so far in bioinformatics:

  Papers with a Mesh indexing keyword of statistics account for 5.9% of all pubmed articles. In oligo array papers this jumps to 13.3%. Cluster analysis accounts for 26% of statistics-related papers on oligo arrays, and 16.7% of genetic network-related papers. In Nature Genetics cluster analysis is used 71.4% on all statistics-related papers. In CAMDA 2000, cluster analysis was used in 27% of all papers.

- Caveats:

a. There does not exist a good understanding on how to translate from “A and B cluster together” to: “A and B are dependent/independent causally/non-causally”

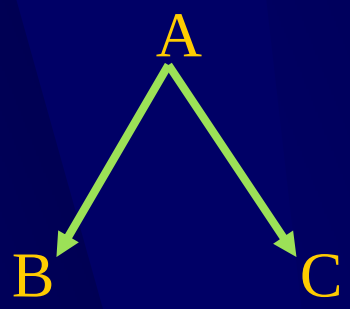
b. There do not exist good studies outlining what can be learned or cannot be learned with clustering methods (learnability), how reliably (validity, stability), with what sample (sample complexity). Such analyses exist for a variety of other methods

c. Other objections: visual appeal, familiarity, small sample, no explicit assumptions to check, accessibility, tractability.

Causal Probabilistic Network Induction

Causal Probabilistic (Bayesian) Network Induction

- Causal Probabilistic Networks provide a simple but very powerful language for representing causal processes
- Causal Bayesian Network = Graph (Variables (nodes), dependencies (arcs)) + Joint Probability Distribution + Markov Property
- Graph has to be DAG (directed acyclic) in the standard BN model



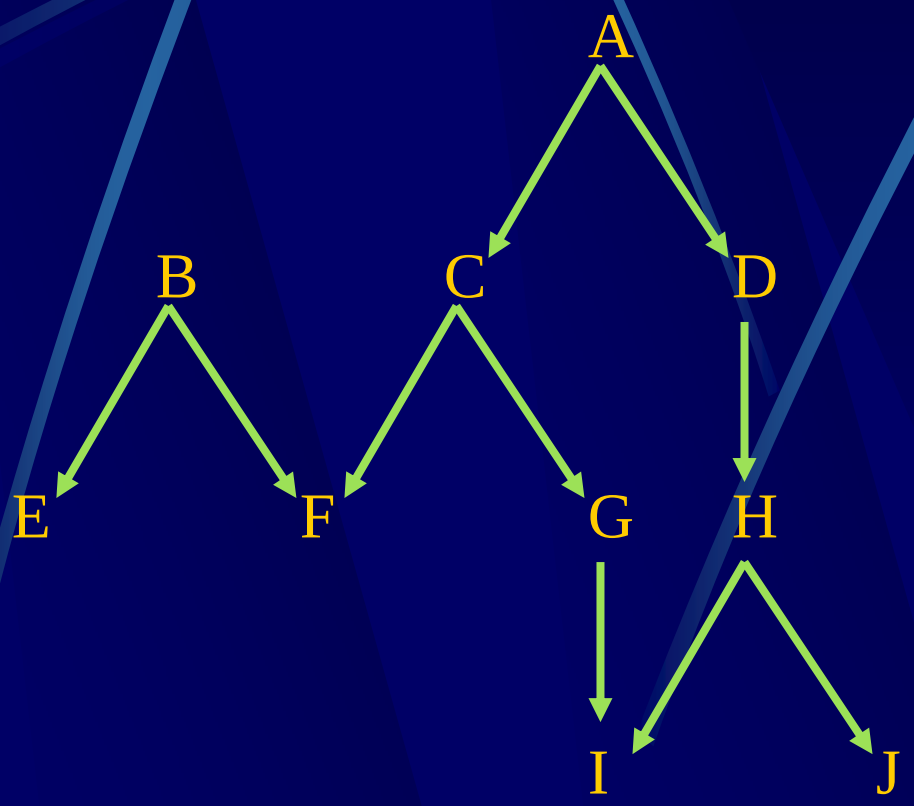
JPD

```
P(A+, B+, C+)=0.006
P(A+, B+, C-)=0.014
P(A+, B-, C+)=0.054
P(A+, B-, C-)=0.126
P(A-, B+, C+)=0.240
P(A-, B+, C-)=0.160
P(A-, B-, C+)=0.240
P(A-, B-, C-)=0.160
```

- Theorem 1 (Neapolitan): any JPD can be represented in BN form

Causal Probabilistic (Bayesian) Network Induction

- Markov Property: the probability distribution of any node ***N*** given its parents ***P*** is independent of any subset of the non-descendent nodes ***W*** of ***N***



e.g., :

D ⊥ {B, C, E, F, G | A}

F ⊥ {A, D, E, F, G, H, I, J | B, C }

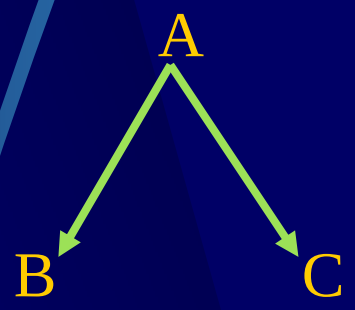
Causal Probabilistic (Bayesian) Network Induction

- Theorem 2 (Pearl): the Markov property enables us to decompose (factor) the joint probability distribution into a product of prior and conditional probability distributions

$$P(V_1, V_2, \ldots, V_n) = \prod_i p\,(V_i \mid \text{Parents}(V_i))$$

Causal Probabilistic (Bayesian) Network Induction

- Theorem 3 (Pearl): A DAG and set of conditional probabilities of each node given its parents defines a BN with a unique and valid jpd.



$$P(\boldsymbol{V}) = \prod_{i} p(V_i|Pa(V_i))$$

The original JPD:

```
P(A+, B+, C+)=0.006
P(A+, B+, C-)=0.014
P(A+, B-, C+)=0.054
P(A+, B-, C-)=0.126
P(A-, B+, C+)=0.240
P(A-, B+, C-)=0.160
P(A-, B-, C+)=0.240
P(A-, B-, C-)=0.160
```

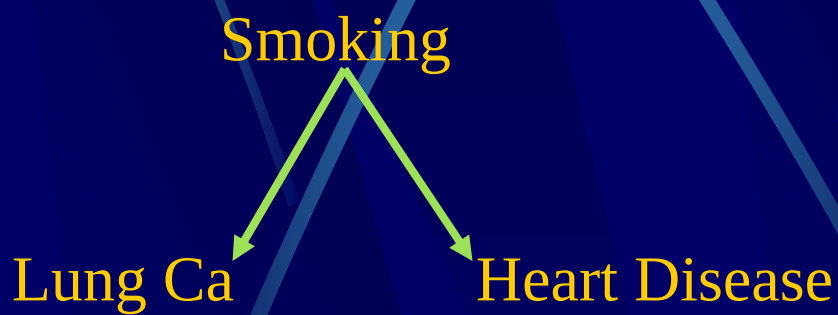
Becomes:

```
P(A+)=0.8
P(B+ | A+)=0.1
P(B+ | A-)=0.5
P(C+ | A+)=0.3
P(C+ | A-)=0.6
```

Up to Exponential Saving in Number of Parameters!

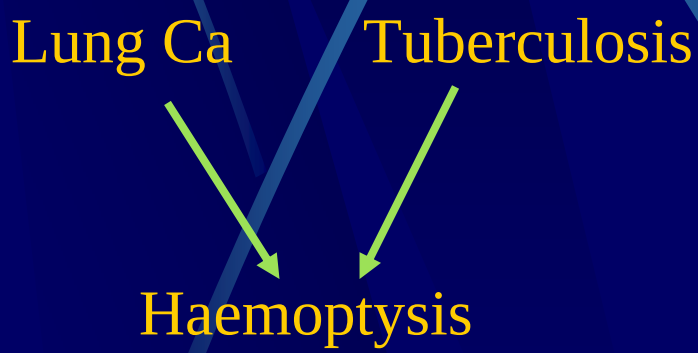
Causal Probabilistic (Bayesian) Network Induction

- The Markov property captures causality:
  - Confounders



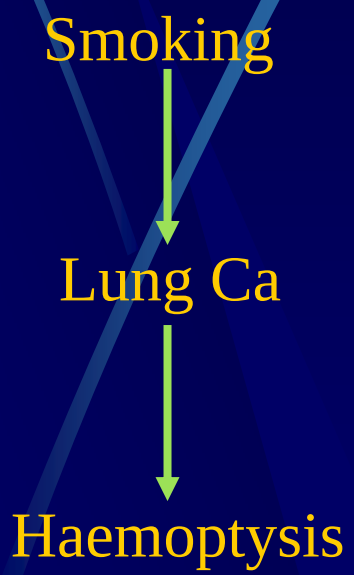
Causal Probabilistic (Bayesian) Network Induction

- The Markov property captures causality:
  - Modeling “explaining away”
  - Modeling/understanding selection bias



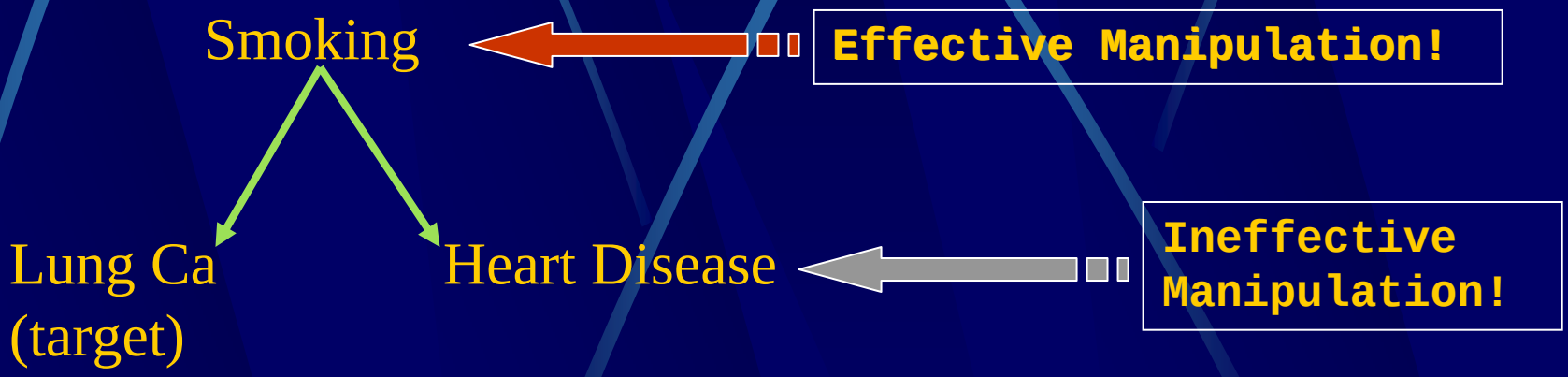
Causal Probabilistic (Bayesian) Network Induction

- The Markov property captures causality:
  - Modeling causal pathways



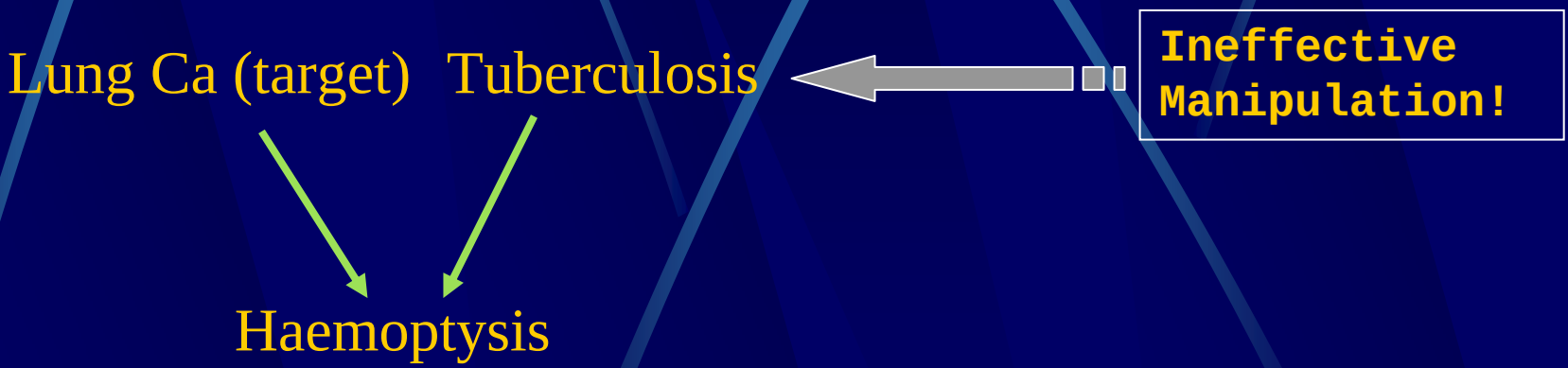
Causal Probabilistic (Bayesian) Network Induction

- The Markov property captures causality:
  - Manipulation in the presence of confounders



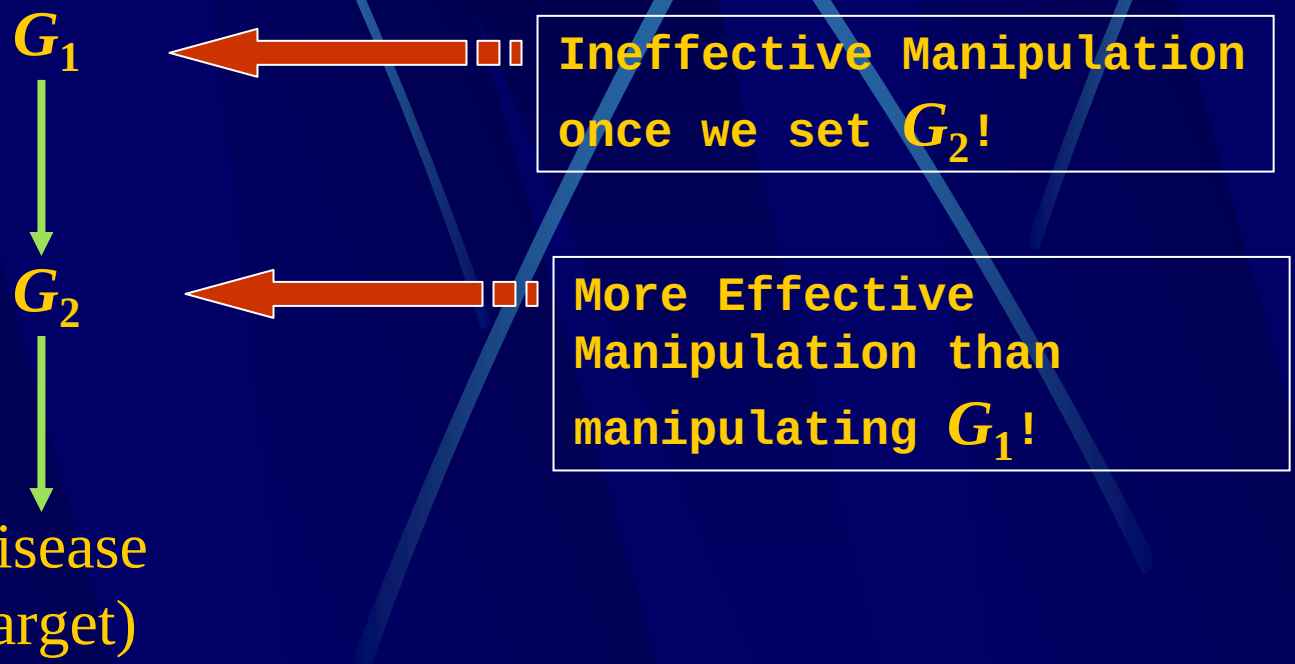
Causal Probabilistic (Bayesian) Network Induction

- The Markov property captures causality:
  - Manipulation in the presence of selection bias



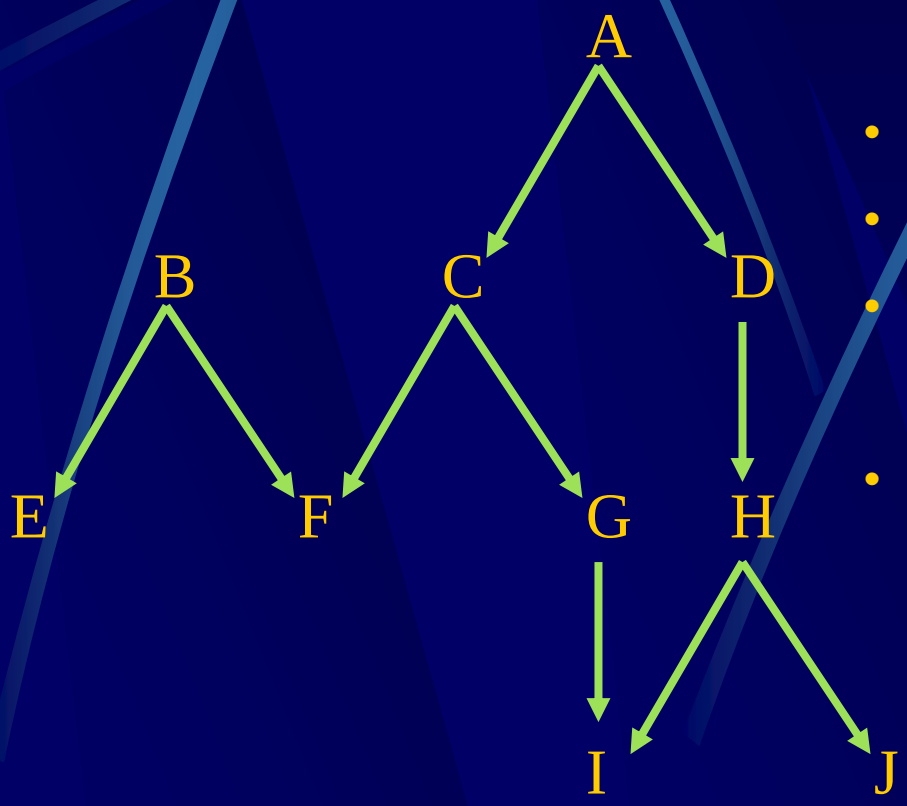
Causal Probabilistic (Bayesian) Network Induction

- The Markov property captures causality:
  - Identifying targets for manipulation in causal chains



Causal Probabilistic (Bayesian) Network Induction

- Inference: Once we have a BN model of some domain we can ask questions:

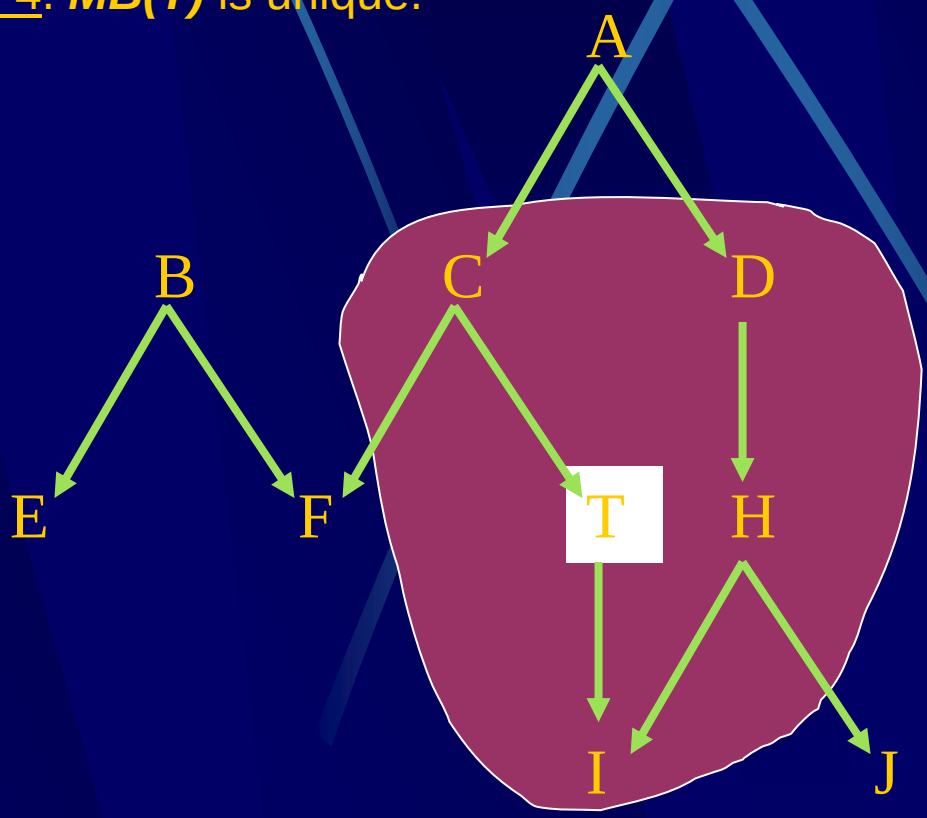


- **Forward: P(D+,I-| A+)=?**
- **Backward: P(A+| C+, D+)=?**
- **Forward & Backward:**

  P(D+,C-| I+, E+)=?
- **Arbitrary abstraction/Arbitrary predictors/predicted variables**

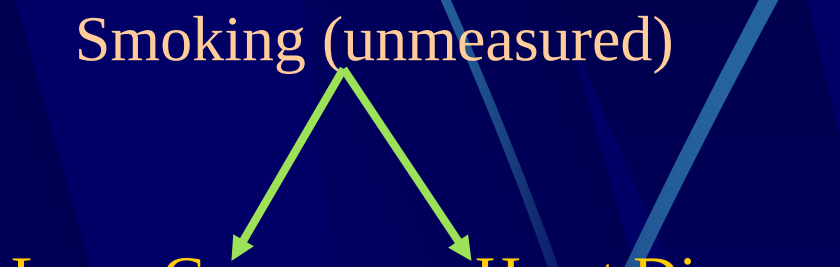
Causal Probabilistic (Bayesian) Network Induction

- Markov Blanket of a feature *T*: The smallest feature subset conditioned on which all other features are independent of *T*.
- Theorem 4. ***MB(T)*** is unique.



Causal Probabilistic (Bayesian) Network Induction

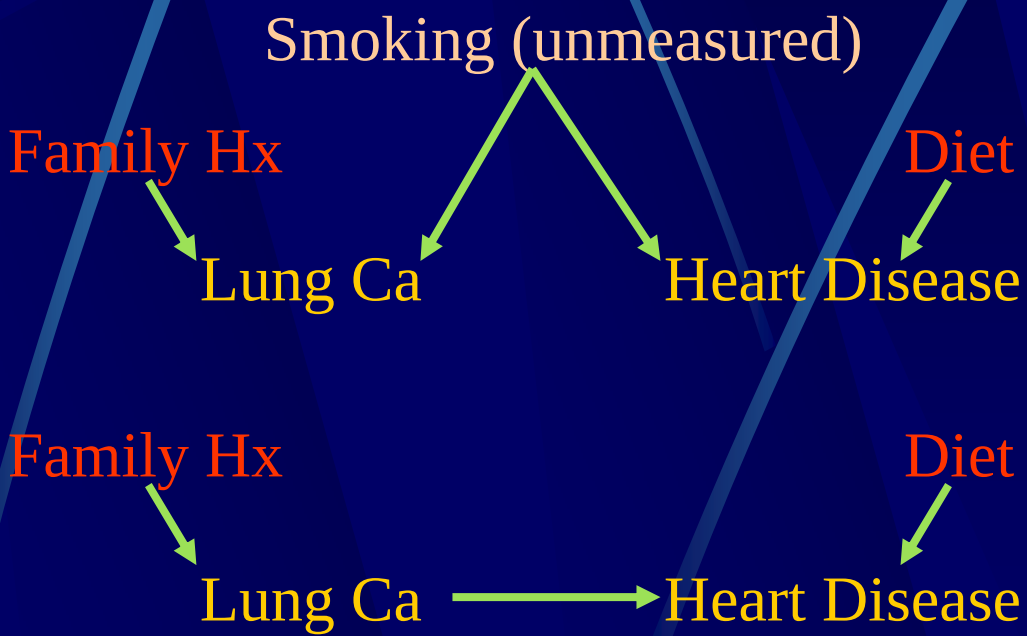
- CPNs can help us learn causal relationships without doing experiments!



But Fisher says these two causal graphs are not distinguishable without doing an experiment (!?)

Causal Probabilistic (Bayesian) Network Induction

- CPNs can help us learn causal relationships without doing experiments!



Fisher is right of course; however if we know a cause of each variable of interest then, in many cases, we can derive causal associations without an experiment

Causal Probabilistic (Bayesian) Network Induction

- Faithfulness. The graph *G* of some CPN *C* is faithful to a joint probability distribution *J* over feature set *V* if and only if every conditional dependence *P* entailed by *J* is also entailed by *G* . (Note that every conditional dependence *P* entailed by *G* is also already entailed by *J* because of the Markov Condition).
- We say that a data-generating process *K* is faithfully represented by *C'*, if *K* in the sample limit produces data with joint probability distribution *D,* and *C'* is faithful to *D*.
- Causal Sufficiency. For every pair of measured variables in the training data, all their common parents are also measured. (Note algorithms for non-causally sufficient datasets also exist, however their theoretical basis is much more complex and will not be discussed here)

Causal Probabilistic (Bayesian) Network Induction

- Causal Discovery with BNs: Assume faithfulness of the data-generating process to the data; assume causal sufficiency; assume data is random sample from all instances produced by the process. Then an algorithm that generates the correct causal network given the data is:

PC Algorithm (Outline)

Phase I: find direct edges by using the criterion that A has a direct edge to B iff for all subsets of features there is no subset S, s.t. independent(A, B | S).
Phase II: orient edges in "collider" triplets (i.e., of the type: A->B<-C) using the criterion that if there are direct edges between A, B and between B,C, but not between A, C, and there is no subset containing B s.t. independent (A,C |B) then A->B<-B
Phase III: enter a constraint-propagation loop for orienting edges further by adding orientations until no further orientations can be produced using the two following criteria: (a) if A->B…->C and A-C then A->C also, and (b) if A->B-C then B->C

For More...

T.Mitchell, “Machine Learning”, McGraw Hill, 1997.