Bayesian Networks continued

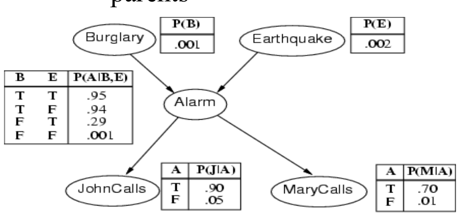
Louis Oliphant
oliphant@cs.wisc.edu
CS540 section 2

Announcements

- Read:
  - Chapter 14 sections 14.1-14.5
  - Chapter 20 section 20.1 and 20.2
- Homework 4 out today
- Project Web pages due next Thursday

Recall Structure

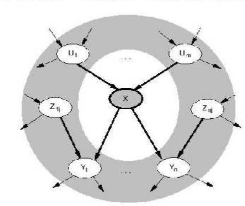
- Directed Acyclic Graph (DAG)
  - each node is a Random Variable
  - RV can be discrete or continuous (we will deal with discrete only)
- Conditional Probability Tables
  - P(Node|Parents(Node)) for each possible setting of parents



Recall Semantics

- Represents the full joint distribution
- Conditional Independence captured in topology of network: Markov blanket

Each node is conditionally independent of all others given its Markov blanket: parents + children + children's parents



Recall Inference in Bayes Nets

- Exact Methods
  - Inference by Enumeration

    $P(X|\mathbf{e})=\alpha P(X,\mathbf{e})=\alpha \sum_{y} P(X,\mathbf{e},\mathbf{y})$
  - Variable Elimination
    - multiplying factors
    - summing out variables in factors
- Approximate Methods
  - Inference is #P-Hard (harder than NP-Hard)
  - Sampling Methods (today)

Learning in Bayes Nets

- Learning Parameters of Bayes Nets
  - Given a dataset, create the CPTs for network (today)
- Learning Structure of Bayes Nets
  - Given a dataset, create the topology of the network (today)

Inference with Sampling Methods

- Direct Sampling
- Rejection Sampling
- Likelihood Weighting
- Markov Chain Monte Carlo

Background: Sampling from a Distribution

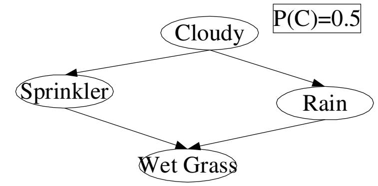
- Suppose you have a probability distribution over a discrete random variable:
  - <green, blue, red, yellow>
  - <0.4, 0.3, 0.2, 0.1>
- How can you generate an assignment for the RV according to the probability distribution and all you have is a way of generating random numbers?

Background: Sampling from a Distribution

- Suppose you have a probability distribution over a discrete random variable:
  - <green, blue, red, yellow>
  - <0.4, 0.3, 0.2, 0.1>
- How can you generate an assignment for the RV according to the probability distribution and all you have is a way of generating random numbers?
  - Convert the probability distribution to a cummulative distribution
    - <0.4, 0.7, 0.9, 1.0>
  - Generate a random number, x:
    - if x<0.4 guess green, if x<0.7 guess blue, if x< 0.9 guess red and if x<1.0 guess yellow

Running Example

P(C)=0.5



C	P(S)
t	.10
f	.50

C	P(R)
t	.80
f	.20

S	R	P(W)
t	t	.99
t	f	.90
f	t	.90
f	f	.00

Direct Sampling

- Only used when no evidence nodes are in network
- Sample each variable in topological order
  - *Cloudy, Sprinkler, Rain, Wet Grass*
- The probability distribution that is used is determined by the settings for the parents
  - Sample P(Cloudy) from distribution <0.5, 0.5>
    - say returns true
  - Sample P(Sprinkler|Cloudy=true) from <0.1, 0.9>
    - say returns false
  - Sample P(Rain|Cloudy=true) from <0.8, 0.2>
    - say returns true
  - Sample P(WetGrass|Sprinkler=false, Rain=true) from <0.9,0.1>
    - say returns true
- A new data point is generated with the given settings:
  - (Cloudy, ¬Sprinkler, Rain, Wet Grass)

Direct Sampling

- Do this lots of times (say N times)
  - Generates lots of data
  - Now you can use your data to estimate the probability of any query
- P(*Wet Grass*=true) is just the number of examples in the data that have *Wet Grass*=true divided by the total number of examples
- This technique can estimate probabilities for any set of query variables
- Probability estimate gets closer to actual probability as N gets large

Rejection Sampling

- Used when you have evidence nodes in network
- Generate Examples using Direct Sampling, Ignoring your evidence
- Throw out any example that does not match the evidence

Rejection Sampling

- Estimate P(Rain|Sprinkler=true)
  - Generate 100 examples
    - 73 have Sprinkler=false, 27 have Sprinkler=true
      - of the 27, Rain=true in 8, Rain=false in 19
    - Throw out the 73 examples, and calculate probabilities just using the 27 remaining
  - P(Rain|Sprinkler=true) = < 8/27, 19/27 >
- Problem with Rejection Sampling is a that a lot of examples can be rejected
  - so your confidence in the estimate will be small

Likelihood Weighting

- Only generates examples that are consistent with the evidence
- Weights examples by how likely they are
- No examples are thrown out, just weighted less if they are less likely

Likelihood Weighting

- Algorithm returns an example and a weight for the example
- weight, w, is initialized to 1
- Visit nodes in topological order
  - non-evidence nodes: sample as in direct sampling
  - evidence nodes: $w \leftarrow w * P(\text{evidence node}|\text{Parents})$

Likelihood Weighting

- Generate an example to answer the query P(*Rain*| *Sprinkler*=true, *WetGrass*=true)
  - visit *Cloudy*: Sample P(*Cloudy*) from <0.5, 0.5>; suppose returns true
  - visit *Sprinkler*: evidence node so weight is updated to be 1.0*P(*Sprinkler*=true|*Cloudy*=true)=0.1
  - visit Rain: Sample P(*Rain*|*Cloudy*=true); suppose returns true
  - visit *WetGrass*: evidence node so weight is updated to be 0.1*P(*WetGrass*=true|*Sprinkler*=true, *Rain*=true)= 0.099
- returns example
  - <cloudy,sprinkler,rain,wetGrass> weight=0.099

Markov Chain Monte Carlo

- A local Search Algorithm
- Starts with some initial setting for each Random Variable
- Randomly sample a value for one of the non-evidence nodes, *conditioned on the current settings for that nodes markov blanket*
- Do this repeatedly
- Keep track of the setting for the query nodes to calculate probabilities
- Guaranteed to converge in the limit

Learning Bayesian Networks

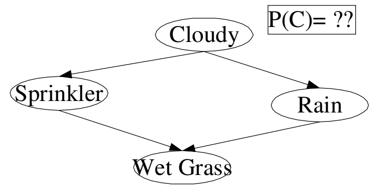
- Learning Parameters
  Given a dataset, create the CPT's for each node
- Learning Structure
  Given a dataset, create the topology of the network

Learning Parameters

- Two types of learning:
  - Maximum Likelihood (ML)
    - set the parameters of the model so that the likelihood of the data will be maximized
  - Maximum A-Posteriori (MAP)
    - You have some prior belief for the hypothesis (what the parameters should be) $\mathbf{P(h_i)}$
    - You also have some new data from which you can calculate the likelihood $\mathbf{P(d|h_i)}$
    - Update the parameters using both the prior and the likelihood $\mathbf{P(h_i|d)=P(d|h_i)P(h_i)}$

Maximum Likelihood

- Visit each node and for each possible entry in the CPT
  - Calculate the fraction of the data that falls into that slot



C	P(S)
t	??
f	??

C	P(R)
t	??
f	??

S	R	P(W)
t	t	??
t	f	??
f	t	??
f	f	??

Sample Data set

Cloudy,	Sprinkler,	Rain,	Wet Grass
t	t	t	t
t	f	f	f
f	f	f	f
t	f	t	t
f	t	f	t

...

Issues

- You need to be concerned about situations that never occur in the data set. Their probabilities will be zero.
- Use Laplacian Priors

Learning Topology

- Search for a good model
  - Start with a model with no links
  - Or start with a model with best guess
- Modify Model
  - Try adding one link at a time
  - Try changing one link at a time (reverse, add, or delete)
- Fit Parameters
  - Use a dataset to fit the CPT's of the model
- Measure Accuracy
  - Using another dataset find the Accuracy of the model for the dataset
  - Keep the changes that maximize the likelihood of the data (while penalizing complexity of the model)

Conclusion

- Sampling Methods
  - Direct Sampling
  - Rejection Sampling
  - Likelihood Weighting
  - Markov Chain Monte Carlo (MCMC)
- Parameter Learning
  - Maximum Likelihood (ML)
  - Maximum A-Posteriori (MAP)
- Structure Learning
  - Search