STATISTICAL METHODS IN AI/ML

Vibhav Gogate
University of Texas, Dallas

LEARNING: Lecture 3

THE UNIVERSITY OF TEXAS AT DALLAS
Erik Jonsson School of Engineering and Computer Science

What we will cover in the next two lectures?

- Parameter Learning in Bayesian networks (FOD and POD)
  - Bayesian Approach
- Structure Learning in Bayesian Networks (FOD and POD)
  - Maximum Likelihood and Bayesian Approach

Bayesian Learning

- Remember our biased coin example. What is the MLE of $P(x = h) = \theta$ when:
  - Coin tossed 10 times and we get 6 heads
  - Coin tossed 100 times and we get 53 heads
  - Coin tossed 1000 times and we get 520 heads
- Which answer will you believe?
- Relying on data may be problematic especially when you have access to only a few examples.
- A better strategy for learning: **Combine data with prior knowledge**.
  - Believe prior knowledge especially when you feel strongly about the knowledge when you don't have enough data.
- **Bayesian Approach:** A formal approach that implements this strategy

Bayes Theorem and Bayesian Statistics

Dataset $\mathcal{D}$ and Parameter θ

- **MLE:** Find a value for θ such that the likelihood of the data is maximized. Namely $\max_\theta P(\mathcal{D}|\theta)$
- **Bayesian approach:** Given a prior distribution $P(\theta)$ on θ, find a posterior distribution over θ given the data. Namely, find $P(\theta|\mathcal{D})$.
- How to find the Posterior? (Remember: Bayes rule)

$$P(\theta|\mathcal{D}) = \frac{P(\mathcal{D}|\theta)P(\theta)}{P(\mathcal{D})} \propto P(\mathcal{D}|\theta)P(\theta)$$

- In other words, Posterior $\propto$ Likelihood $\times$ Prior
- **Maximum-a-Posteriori Estimate (MAP):** Find a value for θ such that the posterior is maximized. Namely $\max_\theta P(\theta|\mathcal{D})$. If $P(\theta)$ is uniform, MAP=MLE. Thus, MAP generalizes MLE.

Bayesian Learning: Point Estimation Example (Discrete Priors)

What is the MLE of $P(x = h) = \theta$:

- 6 heads out of 10. MLE $= 6/10$=0.6
- 53 heads out of 100. MLE $= 53/100$=0.53
- 520 heads out of 1000. MLE=520/1000=0.52

Prior Knowledge:

Domain of θ: $\{0.49, 0.52, 0.55\}$.

The distribution $P(\theta)$:$(0.45, 0.39, 0.16)$.

What is the MAP estimate of θ:

- 6 heads out of 10. MAP solution $= 0.49$.
- 53 heads out of 100. MAP solution $= 0.52$
- 520 heads out of 1000. MAP solution $= 0.52$

Posterior calculations:

Posterior $\propto$ Likelihood $\times$ Prior

Dataset 1: 6 heads out of 10:

$P(\theta = 0.49|Data) \propto 0.49^6(1 - 0.49)^4 \times 0.45$

$P(\theta = 0.52|Data) \propto 0.52^6(1 - 0.52)^4 \times 0.39$

$P(\theta = 0.55|Data) \propto 0.55^6(1 - 0.55)^4 \times 0.16$

Dataset 2: 53 heads out of 100:

$P(\theta = 0.49|Data) \propto 0.49^{53}(1 - 0.49)^{47} \times 0.45$

$P(\theta = 0.52|Data) \propto 0.52^{53}(1 - 0.52)^{47} \times 0.39$

$P(\theta = 0.55|Data) \propto 0.55^{53}(1 - 0.55)^{47} \times 0.16$

Dataset 3: 520 heads out of 1000:

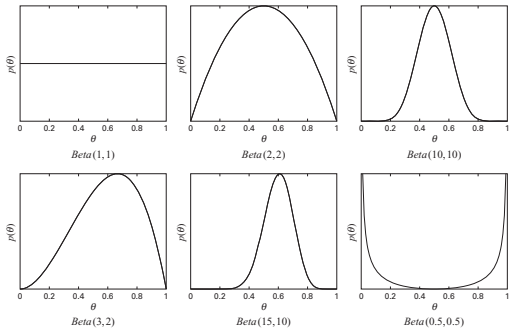
$P(\theta = 0.49|Data) \propto 0.49^{520}(1 - 0.49)^{480} \times 0.45$

$P(\theta = 0.52|Data) \propto 0.52^{520}(1 - 0.52)^{480} \times 0.39$

$P(\theta = 0.55|Data) \propto 0.55^{520}(1 - 0.55)^{480} \times 0.16$

Bayesian Learning: Point Estimation Example (Continuous Priors)

- Likelihood Function is Binomial: $\theta^{\#heads}(1-\theta)^{\#tails}$.
- Use a conjugate prior. $P(\theta)$ is given by the Beta Distribution. Given two real numbers $\alpha > 0$ and $\beta > 0$, the Beta PDF is proportional to $\theta^{\alpha-1}(1-\theta)^{\beta-1}$.



- **Why Conjugate Prior[1]? So that the posterior which is a product of the likelihood and the prior is also a Beta Distribution.**
- Posterior = Likelihood $\times$ Prior = Beta$(\alpha + \#heads, \beta + \#tails)$
- MAP estimate = $\frac{\alpha+\#heads-1}{\alpha+\#heads+\beta+\#tails-2}$. Compare with MLE = $\frac{\#heads}{\#heads+\#tails}$

[1] when the posterior distribution is in the same family as the prior

Bayesian Learning: Some Terms

- The **marginal likelihood** is the distribution of the Data marginalized over the parameter(s), i.e. $P(\mathcal{D}) = \int P(\mathcal{D} \mid \theta) P(\theta) \, \mathrm{d}\theta$.
- The **posterior predictive distribution** is the distribution of a new data point, marginalized over the posterior: $P(x \mid \mathcal{D}) = \int P(x \mid \theta) P(\theta \mid \mathcal{D}) \, \mathrm{d}\theta$
- The **prior predictive distribution** is the distribution of a new data point, marginalized over the posterior: $P(x) = \int P(x \mid \theta) P(\theta) \, \mathrm{d}\theta$
- Conjugate Priors for parameter of binary variables: **Beta Distribution**.
- Conjugate Priors for parameter of variables having more than two values: **Dirichlet distribution**.

Bayesian Learning: From One Parameter to Bayesian Networks (FOD)

- Assume that each parameter has a separate prior and the priors on the parameters are independent. Namely, given a set of parameters $\Theta = \{\theta_1, \ldots, \theta_m\}$, we have:

$$P(\Theta) = \prod_{i=1}^{m} P(\theta_i)$$

Under this assumption, each parameter can be estimated independently. For example given the Prior Distribution $P(\theta_{x|\mathbf{u}}) = Beta(\alpha_{x|\mathbf{u}}, \beta_{x|\mathbf{u}})$ for a parameter $\theta_{x|\mathbf{u}}$

- Posterior of $\theta_{x|\mathbf{u}}$:

$$P(\theta_{x|\mathbf{u}}|\mathcal{D}) = Beta(\#[x, \mathbf{u}] + \alpha_{x|\mathbf{u}}, \#[\bar{x}, \mathbf{u}] + \beta_{x|\mathbf{u}})$$

- MAP Estimate:

$$\theta^{MAP}_{x|\mathbf{u}} = \frac{\#[x, \mathbf{u}] + \alpha_{x|\mathbf{u}} - 1}{\#[x, \mathbf{u}] + \alpha_{x|\mathbf{u}} + \#[\bar{x}, \mathbf{u}] + \beta_{x|\mathbf{u}} - 2}$$

- Compare to MLE:

$$\theta^{ML}_{x|\mathbf{u}} = \frac{\#[x, \mathbf{u}]}{\#[x, \mathbf{u}] + \#[\bar{x}, \mathbf{u}]}$$

Bayesian Approach for Parameter Learning in Bayesian Networks: POD

- Similar to the Likelihood, Posterior will generally be highly complex and multi-modal in the POD case.
- Bayesian inference would require a complex integration procedure, which generally has no analytic solution.
- However, if we use MAP estimation, then we can use EM and Gradient Ascent with minor modifications

$$\max_{\Theta} \log P(\Theta|\mathcal{D}) = \max_{\Theta} \log\{P(\mathcal{D}|\Theta)P(\Theta)\} = \max_{\Theta} \{\log P(\mathcal{D}|\Theta) + \log(P(\Theta))\}$$

- Perform Full Bayesian Inference via Approximate Inference techniques such as sampling and variational inference.

Bayesian Approach: Summary

- Put a prior on the parameters. Namely, set the parameters by leveraging prior/background knowledge.
- Use the data to improve the prior yielding a posterior.
- As the number of examples tend to infinity, the effect of the prior disappears. However, for a finite data size, the learned model depends on the prior.
- Yields a distribution on the parameters instead of a value.
- Generally harder than MLE. Use MAP Estimates to approximate full Bayesian Inference.

Some Jokes: Bayesians vs Frequentists

- A Bayesian is one who, vaguely expecting a horse, and catching a glimpse of a donkey, strongly believes he has seen a mule.
- A Bayesian and a Frequentist were to be executed. The judge asked them what were their last wishes. The Bayesian replied that he would like to give the Frequentist one more lecture. The judge granted the Bayesian's wish and then turned to the Frequentist for his last wish. The Frequentist quickly responded that he wished to hear the lecture again and again and again and again

Next Up

- Structure Learning in Bayesian Networks
  - FOD vs POD
  - MLE vs Bayesian Approach

Some of you asked me about too many mentions of "Bayesian."

- The word "**Bayesian Inference**" comes from Statistics because we use the Bayes rule to estimate the posterior.
- In my opinion, "**Bayesian networks**" is a misnomer. As we saw, it uses the chain rule and conditional independence. We should call them sparse chain networks or suggest a better name!

Learning the Structure of Bayesian Networks given Data: FOD and MLE

- Goal is to find a maximum likelihood structure (and parameters)
- Goal is problematic: a complete DAG is provably the best structure.

Theorem

If a DAG G^* is the result of adding edges to G, then $LL(G^*|\mathcal{D}) \geq LL(G|\mathcal{D})$.

- Overfitting/Underfitting, Bias versus Variance tradeoff

Underfitting **High Bias/Low Variance**	**Overfitting** **Low Bias/High Variance**
Small # of parents Sparse DAG	Large # of parents Dense DAG

- **Prefer Sparser structures (simpler models) even if the model is unlikely to capture the data generating distribution accurately. Sparse structures are likely to yield better generalization in limited data settings.**

Learning the Structure of Bayesian Networks given Data: Solving the Overfitting/Underfitting Problems

- Bias the model: Put Constraints on the type of Bayesian network that is induced by the learning algorithm
  - Treewidth is Bounded by a Constant.
  - Use a scoring function which penalizes models having large number of edges or parents (soft, sparsity constraints)
- Use Latent Variables to eliminate underfitting issues or learn diverse models (nowadays called deep architectures)
- Start with the most Biased Model, reduce the bias gradually and use a validation set (or cross validation) to choose a model

Learning Bounded Treewidth Models

- Good news: Models with treewidth 1 can be learned optimally in polynomial time! Recall that such Bayesian networks are called Singly-connected Bayesian networks and their primal graph is a tree.
- Bad news: Models having treewidth > 1 are NP-hard to learn.

Naive Approach to Learning Optimal Tree Bayesian Networks for FOD

Algorithm

- BestLL $= -\infty$;
- BestTree is initialized to NULL.
- Generate all possible Singly-Connected Tree structures and store them in a set $\mathcal{T}$
- **For each** tree T in $\mathcal{T}$ do
  - Learn parameters of T (recall: closed form solution for the FOD case)
  - **If** $LL(T) >$ Best LL **Then**
    - BestLL $= LL(T)$
    - BestTree $=$ T (with its parameters)
- **Return** BestTree.

Chow-Liu Algorithm: Learning optimal trees for FOD

- Chow and Liu (1968) proved that for trees (does not apply for graphs):

$$argmax_G LL(G|\mathcal{D}) = argmax_G \sum_{U \to X} MI(X, U)$$

Mutual Information

$$MI(X, U) = \sum_{x,u} \Pr(x, u) \log \left[\frac{\Pr(x, u)}{\Pr(x) \Pr(u)} \right]$$

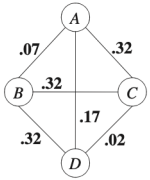
  - measures the dependence between the variables.
  - Higher the value, higher the dependence.
  - is zero only when X and U are independent

- What this means? Given $MI(X, U)$ for all pairs (X, U), find a spanning tree having the maximum $\sum_{U \to X} MI(X, U)$

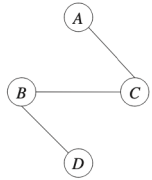
Learning Bounded Treewidth Bayesian Networks

- NP-hard Task. No efficient algorithm exists.
- Best known approach by Elidan and Gould, JMLR 2008.
- In practice, adding Latent Variables with Trees works best.

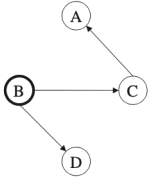
Chow-Liu Algorithm: Learning optimal trees given complete data



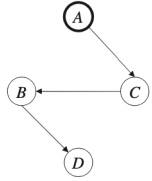
(a) Mutual information graph



(b) Maximum spanning tree



(c) Maximum likelihood tree



(d) Maximum likelihood tree

Log likelihood = -12.1

- Construct the complete graph with edges weighted by mutual information
- Find a maximal spanning tree (e.g., using Kruskal's algorithm)
- Convert the tree to a rooted tree (think: root at the top) and orient the arcs from the top to bottom.
- Learn the parameters of the tree Bayesian network using the FOD closed form solution.
- Complexity: $O(n^2 M + n^2 \log n)$

Chow-Liu Trees with Latent Variables

- Mixtures of Tree Bayesian networks. Add one latent variable with k values

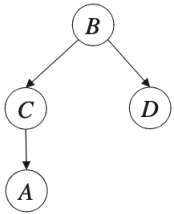
$$P(\mathbf{X}) = \sum_{i=1}^{k} p_i Q_i(\mathbf{X}; \Theta_i)$$

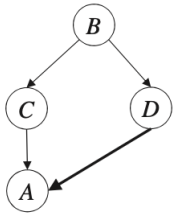
  where $p_1, \ldots, p_k$ are called mixture weights; $p_i \geq 0$ and $\sum_{i=1}^{k} p_i = 1$; and $Q_i(\mathbf{X}; \Theta_i)$ is a tree Bayesian Network.
- How to Learn these networks? The EM Algorithm
  - Begin with a random model (randomly choose trees and parameters for each tree as well as randomly choose p_i's)
  - Repeat until convergence
    - Complete the data using current model
    - Estimate the parameters using the completed data
- More sophisticated deep network of latent variables (Cutset networks, Sum Product Networks, Probabilistic Sentential Decision Diagrams)

Score-Based Methods

- Using Likelihood is problematic! Therefore, add a penalty term that will prefer simpler models over complicated, highly connected ones.
- Score of model = $\mathrm{LL}(G|D)$ + Penalty
- $\mathit{Penalty} = -\phi(M) \times \mathit{Dim}(G)$
  - $\phi(M)$ is a function of the number of data points
  - $\mathit{Dim}(G)$ is the number of parameters in the Bayesian network
- Akaike Information criteria: $\phi(M) = \mathit{constant}$
- Minimum description length (Bayesian information criteria): $\phi(M) = \frac{\log_2(M)}{2}$

Likelihood vs MDL scores





Likelihoods from left to right: -12.1 and -10.1

MDL or BIC Penalty $= -\frac{\log_2(5)}{2}(7) = -8.1$

MDL or BIC Penalty $= -\frac{\log_2(5)}{2}(9) = -10.4$

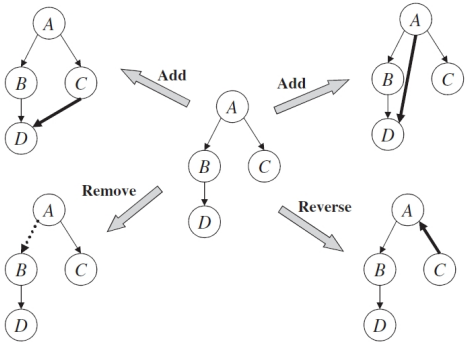
- Which one will you prefer if you use just likelihood?
- Which one will you prefer if you use MDL?

Searching for network structure that optimizes a given score

- Expensive because we have to consider a large number of network structures
- Greedy algorithms, local search style algorithms
- Score should be decomposable over the families for fast search

$$Score(G|\mathcal{D}) = \sum_{X\mathbf{U}} Score(X, \mathbf{U}|\mathcal{D})$$

Local search



Score after deleting $A \rightarrow B$ is

$$Current\ Score - Score(B, A) + Score(B)$$

Remove changes one family
Add changes one family
Reverse changes two families

Constraining the Search Space

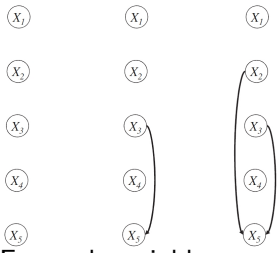
Similar to the algorithm for generating I-maps

- Assume a total ordering of variables
- For each variable find a set of previous variables $\mathbf{U}_i$ that maximize the corresponding local score $Score(X_i, \mathbf{U}_i | \mathcal{D})$.

Problems

- Everything depends upon the ordering used

Greedy Search: The K3 algorithm



For X_5

- X_3 yields the best score
- X_3, X_2 yields the best score
- No extension of X_3, X_2 improves the score. Therefore Stop.

For each variable

- Start with empty set of parents
- Successively add variables (as parents) until the score does not increase. At each step, add the variable that increases the score the most.

Branch and Bound Search, Structural EM etc.

- Branch and Bound search. Complicated. Seldom used in practice. Homework, read chapter 17 from the book by Adnan Darwiche.
- Learning Bayesian Networks in the Presence of POD (Structural EM: Not covered). Read chapter 19 from the book by Koller & Friedman.

Summary

- Why Structure Learning using LL score is problematic?
- Learning Bounded Treewidth Networks: The Chow-Liu algorithm
- Using Latent variables and Mixtures
- Score-Based Methods: Greedy and Local Search methods.