CSE 573: Artificial Intelligence
Spring 2012

Structure Learning, EM, Cotraining

Dan Weld

Slides adapted from Carlos Guestrin, Krzysztof Gajos, Dan Klein, Stuart Russell, Andrew Moore & Luke Zettlemoyer

Some Typical Biases

- Occam's razor

We are to admit no more causes of natural things than such as are both true and sufficient to explain their appearances
– William of Ockham (1288-1348)

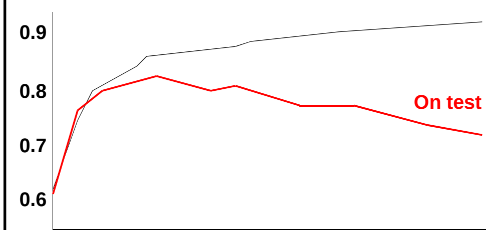
© Daniel S. Weld 2

Some Typical Biases

- Occam's razor
- MDL – Minimum description length
- Concepts can be approximated by
  - ... **conjunctions** of predicates,
  - ... **linear** functions
  - ... **short** decision trees
- Maximal conditional independence
- Minimum cross-validation error
- Minimum number of features
- Etc..

© Daniel S. Weld 3

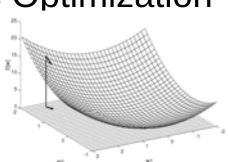
Overfitting



© Daniel S. Weld 4

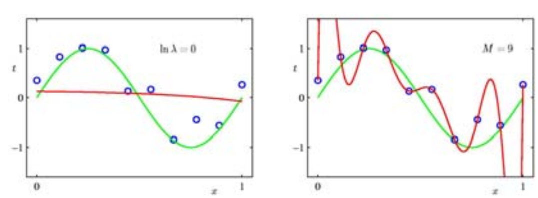
Learning as Optimization

- Methods
  - Closed form
  - Greedy search
  - Gradient ascent
- Loss Function (preference bias)
  - Minimize ***loss*** over training data (test data)
  - Loss(h,data) = error(h, data) + complexity(h)



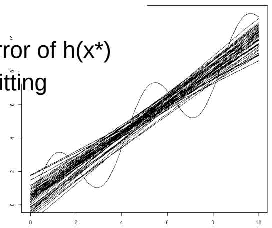
Regularization term E.g., $\lambda\ ||w||^2$

Effect of Regularization



Bias / Variance Tradeoff

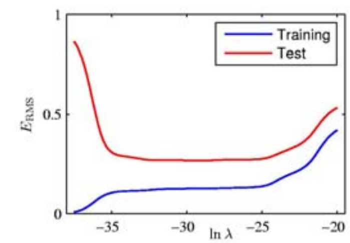
- Variance: **E[(h(x*) – h(x*))²]**
  How much h(x*) varies between training sets
  Reducing variance risks underfitting
- Bias: **[h(x*) – f(x*)]**
  Describes the ***average*** error of h(x*)
  Reducing bias risks overfitting



Note: **inductive bias** *vs* **estimator bias**

Slide from T Dietterich

Bias-Variance Tradeoff



High Variance High Bias

Topics

- Learning Parameters for a Bayesian Network
  - Fully observable
  - Hidden variables (EM algorithm)
- Learning Structure of Bayesian Networks
- Cool Stuff
  - Learning Ensembles
  - Cotraining

© Daniel S. Weld

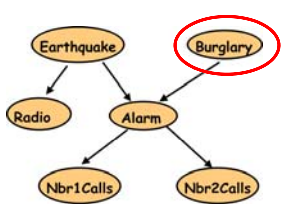
Summary

	Prior	Hypothesis
Maximum Likelihood Estimate	Uniform	The most likely
Maximum A Posteriori Estimate	Any	The most likely
Bayesian Estimate	Any	Weighted combination

Minimizes error
Great when data is scarce
Potentially much harder to compute

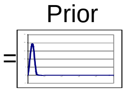
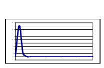
Beta & Dirichlet

Parameter Estimation and Bayesian Networks



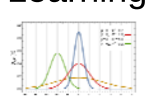
B
F
F
T
F
T

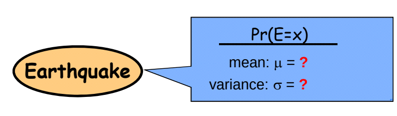
Prior

P(B) =  + data = 

Now compute either MAP or Bayesian estimate

Learning with Continuous Variables





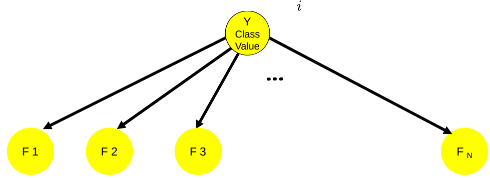
$$\hat{\mu}_{MLE} = \frac{1}{N}\sum_{i=1}^{N} x_i$$

$$\hat{\sigma}^2_{MLE} = \frac{1}{N}\sum_{i=1}^{N} (x_i - \hat{\mu})^2$$

© Daniel S. Weld

A Popular Structure: Naïve Bayes

$$P(Y, F_1 \dots F_n) = P(Y)\prod_i P(F_i|Y)$$

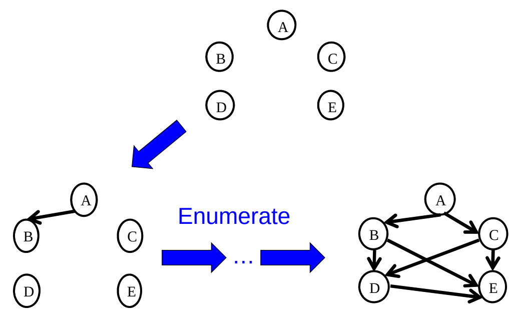


Assume that features are conditionally independent given class variable
Works surprisingly well for ***classification*** (predicting the right class)
But forces probabilities towards 0 and 1

What if we ***don't*** know structure?

Learning The Structure of Bayesian Networks

- Search thru the space…
  - of possible network structures!
  - (for now still assume can observe all values)
- For each structure, learn parameters
  - As just shown…
- Pick the one that fits observed data best
  - Calculate P(data)

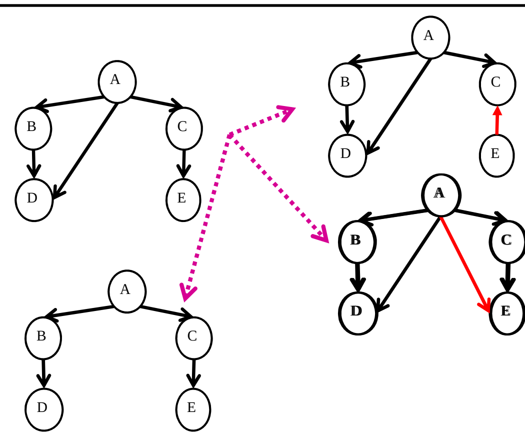


Two problems:
- Fully connected graph will be most probable
- Exponential number of structures

Learning The Structure of Bayesian Networks

- Search thru the space…
  - of possible network structures!
- For each structure, learn parameters
  - As just shown…
- Pick the one that fits observed data best
  - Calculate P(data)

Two problems:
- Fully connected will be most probable
  - Add penalty term (regularization) ∝ model complexity
- Exponential number of structures
  - Local search



Score Functions

- Bayesian Information Criterion (BIC)
  - P(D | BN) – penalty
  - Penalty = ½ (# parameters) Log (# data points)
- MAP score
  - P(BN | D) = P(D | BN) P(BN)
  - P(BN) must decay exponentially with # of parameters for this to work well
- Loss(h,data) = error(h, data) + complexity(h)

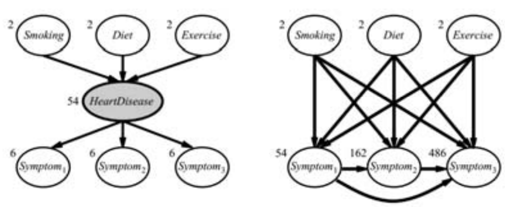
© Daniel S. Weld

Topics

- Learning Parameters for a Bayesian Network
  - Fully observable
  - Hidden variables (EM algorithm)
- Learning Structure of Bayesian Networks
- Cool Stuff
  - Learning Ensembles
  - Cotraining

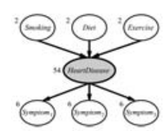
© Daniel S. Weld

Why Learn Hidden Variables?



Chicken & Egg Problem

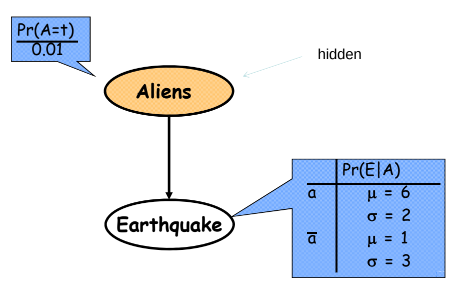
- If we knew whether patient had disease
  - It would be easy to learn CPTs
  - But we can't observe states, so we don't!



- If we knew CPTs
  - It would be easy to predict if patient had disease
  - But we don't, so we can't!

Slide by Daniel S. Weld

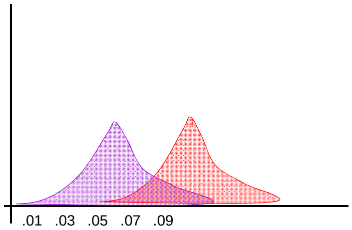
Continuous Variables



© Daniel S. Weld

Simplest Version

- Mixture of two distributions

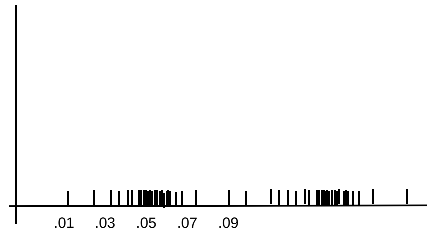


- Know: form of distribution & variance, σ =1
- Just need *mean* of each distribution

Slide by Daniel S. Weld

26

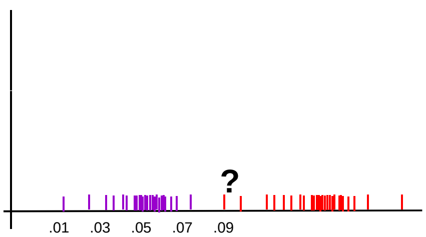
Input Looks Like



Slide by Daniel S. Weld

27

We Want to Predict

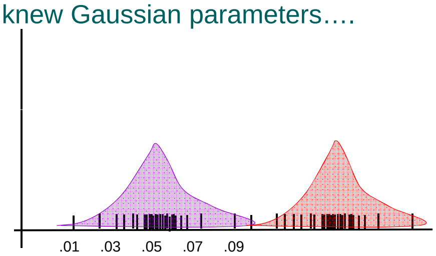


Slide by Daniel S. Weld

28

Chicken & Egg

Note that coloring instances would be easy *if* we knew Gaussian parameters….

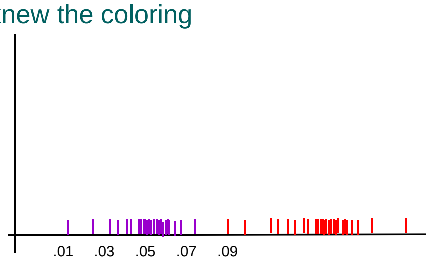


Slide by Daniel S. Weld

29

Chicken & Egg

And finding the Gaussians would be easy *if* we knew the coloring

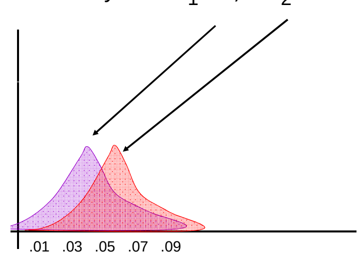


Slide by Daniel S. Weld

30

Expectation Maximization (EM)

- Pretend we *do* know the parameters
  - Initialize randomly: set θ_1=?; θ_2=?

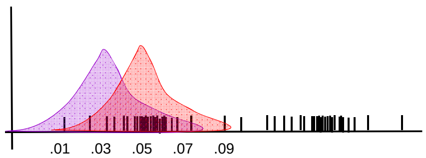


Slide by Daniel S. Weld

31

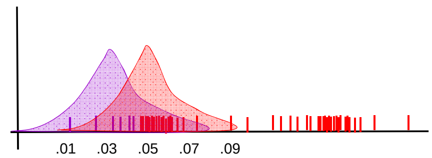
Expectation Maximization (EM)

- Pretend we *do* know the parameters
  - Initialize randomly
- [E step] Compute probability of instance having each possible value of the hidden variable



Expectation Maximization (EM)

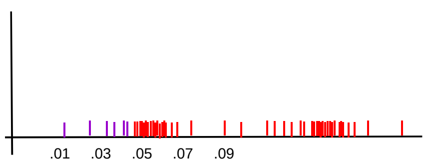
- Pretend we *do* know the parameters
  - Initialize randomly
- [E step] Compute probability of instance having each possible value of the hidden variable



Expectation Maximization (EM)

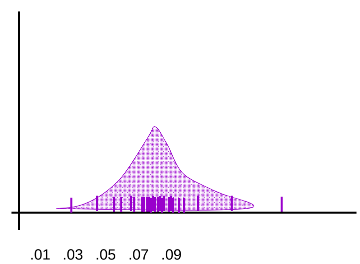
- Pretend we *do* know the parameters
  - Initialize randomly
- [E step] Compute probability of instance having each possible value of the hidden variable

[M step] Treating each instance as ***fractionally*** having **both** values compute the new parameter values



ML Mean of Single Gaussian

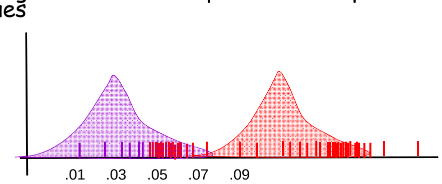
$U_{ml} = \text{argmin}_u \sum_i (x_i - u)^2$



Expectation Maximization (EM)

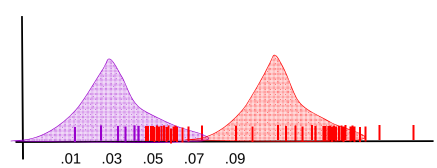
-

[M step] Treating each instance as fractionally having **both** values compute the new parameter values



Expectation Maximization (EM)

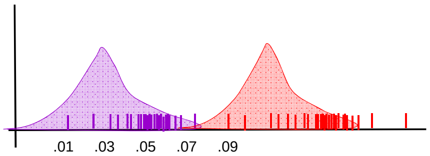
- [E step] Compute probability of instance having each possible value of the hidden variable



Expectation Maximization (EM)

- [E step] Compute probability of instance having each possible value of the hidden variable

[M step] Treating each instance as fractionally having both values compute the new parameter values



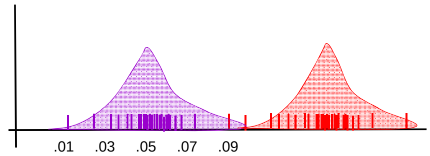
Slide by Daniel S. Weld

38

Expectation Maximization (EM)

- [E step] Compute probability of instance having each possible value of the hidden variable

[M step] Treating each instance as fractionally having both values compute the new parameter values



Slide by Daniel S. Weld

39

EM

- Works for multiple hidden variables
  & other parametric forms
  - E.g., Baum-Welch algorithm for HMMs
- Optimality?
- Complexity?
- Search?

40

Topics

- Learning Parameters for a Bayesian Network
  - Fully observable
  - Hidden variables (EM algorithm)
- Learning Structure of Bayesian Networks
- Cool Stuff
  - Learning Ensembles
  - Cotraining

© Daniel S. Weld

Ensembles of Classifiers

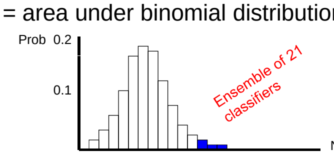
- Traditional approach: Use one classifier
- Can one do better?
- Approaches:
  - Cross-validated committees
  - Bagging
  - Boosting
  - Stacking

© Daniel S. Weld

Ensembles of Classifiers

- Assume
  - Errors are independent (suppose 30% error)
  - Majority vote
- Probability that majority is wrong…

= area under binomial distribution

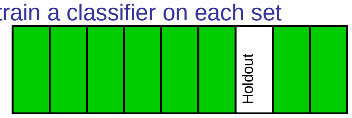


- If individual area is 0.3
- Area under curve for ≥11 wrong is 0.026
- Order of magnitude improvement!

© Daniel S. Weld

Constructing Ensembles Cross-validated committees

- Partition examples into *k* disjoint equiv classes
- Now create *k* training sets
  - Each set is union of all equiv classes ***except one***
  - So each set has (k-1)/k of the original training data
- Now train a classifier on each set



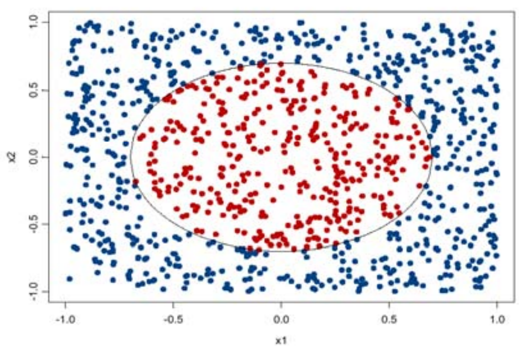
© Daniel S. Weld

Ensemble Construction II Bagging

- Generate k sets of training examples
- For each set
  - Draw m examples randomly (with replacement)
  - From the original set of m examples
- Each training set corresponds to
  - 63.2% of original (+ duplicates)
- Now train classifier on each set
- Intuition: Sampling helps algorithm become more robust to noise/outliers in the data

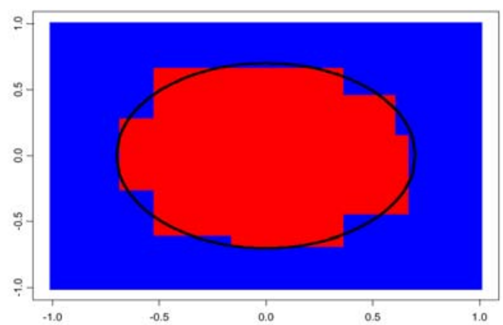
© Daniel S. Weld

Bagging Example

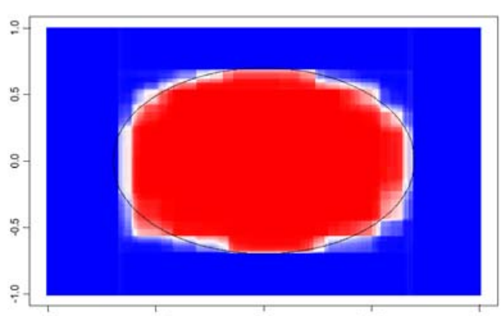


decision tree learning algorithm; very similar to ID3

CART decision boundary



100 bagged trees



shades of blue/red indicate strength of vote for particular classification

Boosting [Schapire, 1989]

- Idea: run weak learner multiple times on (reweighted!) training data; weight learned classifiers ∝ their accuracy
- On each iteration *t*:
  - Learn a hypothesis, h_t, using distribution to weight examples
  - Compute a strength for this hypothesis – α_t
  - Reweight training examples by how well they were classified
- Final classifier:

$$h(x) = \text{sign}\left(\sum_i \alpha_i h_i(x)\right)$$

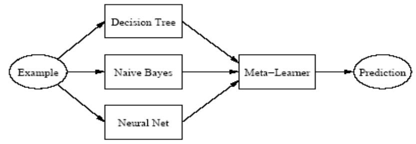
- **Practically useful**
- **Theoretically interesting**

Bagging vs Boosting

Slide from T Dietterich 52

Ensemble Creation IV
Stacking

- Train several base learners
- Next train meta-learner
  - Learns when base learners are right / wrong
  - Now meta learner arbitrates



Train using cross validated committees
- Meta-L inputs = base learner predictions
- Training examples = 'test set' from cross validation

53

© Daniel S. Weld

Topics

- Learning Parameters for a Bayesian Network
  - Fully observable
  - Hidden variables (EM algorithm)
- Learning Structure of Bayesian Networks
- Cool Stuff
  - Learning Ensembles
  - Semi-supervised learning (Cotraining)

© Daniel S. Weld