CSE 473: Artificial Intelligence
Spring 2012

Bayesian Networks - Learning

Dan Weld

Slides adapted from Jack Breese, Dan Klein, Daphne Koller, Stuart Russell, Andrew Moore & Luke Zettlemoyer

Search thru a
Problem Space / State Space

• Input:
  - Set of states
  - Operators [and costs]
  - Start state
  - Goal state [test]

• Output:
  - Path: start ⇒ a state satisfying goal test
  - [May require shortest path]
  - [Sometimes just need state passing test]

Graduation?

• Getting a BS in CSE as a search problem?
  (don’t think too hard)

• Space of States
• Operators
• Initial State
• Goal State

Topics

• Some Useful Bayes Nets
  - Hybrid Discrete / Continuous
  - Naïve Bayes

• Learning Parameters for a Bayesian Network
  - Fully observable
  - Maximum Likelihood (ML),
  - Maximum A Posteriori (MAP)
  - Bayesian
  - Hidden variables (EM algorithm)

• Learning Structure of Bayesian Networks

Bayes Nets

Continuous Variables

So far: assuming variables have discrete values
Could also allow continuous values, E ∈ R,
And specify probabilities using a continuous distribution, such as a Gaussian

\[ P(x | \mu, \sigma) = \frac{1}{\sigma \sqrt{2\pi}} e^{-\frac{(x - \mu)^2}{2\sigma^2}} \]
Continuous Variables

So far: assuming variables have discrete values
Could also allow continuous values, \( E \in \mathbb{R} \)
And specify probabilities using a continuous distribution, such as a Gaussian

\[
P(x | \mu, \sigma) = \frac{1}{\sigma \sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}
\]

Earthquake

Pr(E=x)

mean: \( \mu = 6 \)
variance: \( \sigma = 2 \)

Continuous Variables

Aliens

Pr(A=t) Pr(A=f)

0.01 0.99

Earthquake

Pr(E|A)

a

\( \mu = 6 \)
\( \sigma = 2 \)

Bayesian Learning

Use Bayes rule:

\[
P(Y | X) = \frac{P(X | Y) P(Y)}{P(X)}
\]

Or equivalently: \( P(Y | X) \propto P(X | Y) P(Y) \)

Parameter Estimation and Bayesian Networks

We have:
- Bayes Net structure and observations
- We need: Bayes Net parameters

Parameter Estimation and Bayesian Networks

Now compute either MAP or Bayesian estimate
What Prior to Use?

- Prev, you knew: it was one of only three coins
  - Now more complicated...
  - The following are two common priors
    - Binary variable Beta
      - Posterior distribution is binomial
      - Easy to compute posterior
    - Discrete variable Dirichlet
      - Posterior distribution is multinomial
      - Easy to compute posterior

Beta Distribution

- Example: Flip coin with Beta distribution as prior over p [prob(heads)]
  1. Parameterized by two positive numbers: a, b
  2. Mode of distribution (E[p]) is \( a/(a+b) \)
  3. Specify our prior belief for \( p = a/(a+b) \)
  4. Specify confidence in this belief with high initial values for a and b
- Updating our prior belief based on data
  - incrementing \( a \) for every heads outcome
  - incrementing \( b \) for every tails outcome
- So after \( h \) heads out of \( n \) flips, our posterior distribution says \( P(\text{head}) = (a+h)/(a+b+n) \)

One Prior: Beta Distribution

The Beta distribution is
\[
\beta_{a,b}(x) = \frac{\Gamma(a + b)}{\Gamma(a)\Gamma(b)} x^{a-1} (1-x)^{b-1},
\]
where \( 0 \leq x \leq 1 \) and \( a, b > 0 \)

Parameter Estimation and Bayesian Networks

Prior
\[
P(B|\text{data}) = \text{Beta}(1,4) \quad \text{"+ data"} = (3,7)
\]
Prior \( P(B) = 1/(1+4) = 20\% \) with equivalent sample size 5

Parameter Estimation and Bayesian Networks

\[
P(A|E,B) = ?
P(A|E,\neg B) = ?
P(A|\neg E,B) = ?
P(A|\neg E,\neg B) = ?
\]
Parameter Estimation and Bayesian Networks

Prior

P(A|E,B) = ?
P(A|E,¬B) = ?
P(A|¬E,B) = ?
P(A|¬E,¬B) = ?

Prior + data = Beta(2,3)

Output of Learning

Did Learning Work Well?

Can easily calculate P(data) for learned parameters

Learning with Continuous Variables

Bayes Nets for Classification

- One method of classification:
  - Use a probabilistic model!
  - Features are observed random variables F_i
  - Y is the query variable
  - Use probabilistic inference to compute most likely Y
    \[ y = \arg \max_y P(y|f_1 \ldots f_n) \]
  - You already know how to do this inference
A Popular Structure: Naïve Bayes

\[ P(Y, F_1 \ldots F_n) = P(Y) \prod_i P(F_i | Y) \]

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

Naïve Bayes

- Naïve Bayes assumption:
  - Features are independent given class:
  \[ P(X_1, X_2 | Y) = P(X_1 | X_2, Y) P(X_2 | Y) \]
  \[ = P(X_1 | Y) P(X_2 | Y) \]
  More generally:
  \[ P(X_1 \ldots X_n | Y) = \prod_i P(X_i | Y) \]

- How many parameters?
  - Suppose \( X \) is composed of \( n \) binary features

A Spam Filter

- Naïve Bayes spam filter
- Data:
  - Collection of emails, labeled spam or ham
  - Note: someone has to hand label all this data!
  - Split into training, held-out, test sets
- Classifiers
  - Learn on the training set
  - Tune it on a held-out set
  - Test it on new emails

Dear Sir,

First, I must solicit your confidence in this transaction, this is by virtue of its nature as being utterly confidential and top secret...

TO BE REMOVED FROM FUTURE MAILING, SIMPLY REPLY TO THIS MESSAGE AND PUT "REMOVE" IN THE SUBJECT.

Ok, before this is Matterly OFF but the beginning to go insane. Had an old Dell Dimension XPS sitting in the corner and decided to put it to use, I knew it was working pre being stuck in the corner, but when I plugged it in, hit the power nothing happened.

Example: Spam Filtering

- Model: \( P(C, W_1 \ldots W_n) = P(C) \prod_i P(W_i | C) \)
- What are the parameters?

\[
\begin{align*}
  P(C) & : \text{ham} : 0.66, \text{spam} : 0.33 \\
  P(W|\text{spam}) & : \text{the} : 0.0156, \text{to} : 0.013, \text{and} : 0.0119, \text{of} : 0.0095, \text{you} : 0.0093, \text{a} : 0.0086, \text{with} : 0.0086, \text{from} : 0.0075, \ldots \\
  P(W|\text{ham}) & : \text{the} : 0.0210, \text{to} : 0.0133, \text{of} : 0.0119, \text{2002} : 0.0110, \text{with} : 0.0108, \text{from} : 0.0107, \text{and} : 0.0105, \text{a} : 0.0100, \ldots 
\end{align*}
\]

Where do these come from?

Example: Overfitting

- Posterior determined by relative probabilities (odds ratios):

\[
\begin{align*}
  P(W|\text{ham}) & : \text{south-west} : \text{inf}, \text{nation} : \text{inf}, \text{morally} : \text{inf}, \text{nicely} : \text{inf}, \text{seriously} : \text{inf}, \ldots \\
  P(W|\text{spam}) & : \text{screens} : \text{inf}, \text{minute} : \text{inf}, \text{guaranteed} : \text{inf}, \text{\$205.00} : \text{inf}, \text{delivery} : \text{inf}, \text{signature} : \text{inf}, \ldots 
\end{align*}
\]

What went wrong here?
Generalization and Overfitting

- Relative frequency parameters will overfit the training data!
- Unlikely that every occurrence of "money" is 100% spam
- Unlikely that every occurrence of "office" is 100% ham
- What about all the words that don’t occur in the training set at all?
- In general, we can’t go around giving unseen events zero probability

- As an extreme case, imagine using the entire email as the only feature
  - Would get the training data perfect (if deterministic labeling)
  - Wouldn’t generalize at all
  - Just making the bag-of-words assumption gives some generalization, but not enough

- To generalize better: we need to smooth or regularize the estimates

Estimation: Smoothing

- Problems with maximum likelihood estimates:
  - If I flip a coin once, and it’s heads, what’s the estimate for P(heads)?
  - What if I flip 10 times with 8 heads?
  - What if I flip 10M times with 8M heads?

- Basic idea:
  - We have some prior expectation about parameters (here, the probability of heads)
  - Given little evidence, we should skew towards our prior
  - Given a lot of evidence, we should listen to the data

- Relative frequencies are the maximum likelihood estimates

  \[
  \theta_{ML} = \arg \max_{\theta} P(X|\theta) = \frac{\text{count}(x)}{\text{total samples}}
  \]

- In Bayesian statistics, we think of the parameters as just another random variable, with its own distribution

  \[
  \theta_{MAP} = \arg \max_{\theta} P(\theta|X) = \arg \max_{\theta} \frac{P(X|\theta)P(\theta)}{P(X)}
  \]

Estimation: Laplace Smoothing

- Laplace’s estimate:
  - Pretend you saw every outcome once more than you actually did

  \[
  P_{\text{LAP}}(x) = \frac{c(x) + 1}{N + k|X|}
  \]

  - Laplace’s estimate (extended):
    - Pretend you saw every outcome k extra times

  \[
  P_{\text{LAP}}(x|y) = \frac{c(x,y) + k}{c(y) + k|X|}
  \]

  - What’s Laplace with k = 0?
  - k is the strength of the prior

  - Laplace for conditionals:
    - Smooth each condition independently:

- For real classification problems, smoothing is critical

  - New odds ratios:

    |    | helvetica | 11.4 | seems | 10.8 | group | 10.2 | areas | 8.3 | ...
    |    | Credit | 28.8 | ORDER | 27.2 | <POST> | 26.9 | money | 26.5 | ...

  - Do these make more sense?
NB with Bag of Words for text classification

- **Learning phase:**
  - Prior $P(Y)$
    - Count how many documents from each topic (prior)
  - $P(X|Y)$
    - For each of $m$ topics, count how many times you saw word $X_i$ in documents of this topic (+ $k$ for prior)
    - Divide by number of times you saw the word (+ $k$ | words)

- **Test phase:**
  - For each document
    - Use naïve Bayes decision rule
      $$h_{NB}(x) = \arg\max_y P(y)^{\text{LengthDoc}} \prod_{i=1}^{\text{LengthDoc}} P(x_i|y)$$

**Probabilities: Important Detail!**

- $P(\text{spam} | X_1 \ldots X_n) = \prod_i P(\text{spam} | X_i)$

**Any more potential problems here?**

- We are multiplying lots of small numbers
  - Danger of underflow!
    - $0.5^{57} \approx 7 \times 10^{-18}$

- Solution? Use logs and add!
  - $p_1 * p_2 = e^{\log(p_1) + \log(p_2)}$
  - Always keep in log form

---

**Naïve Bayes**

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

---

**Example Bayes’ Net: Car**

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

---

**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 $P(\text{data})$

---

What if we don’t know structure?
Learning The Structure of Bayesian Networks

- Search thru the space
- For each structure, learn parameters
- Pick the one that fits observed data best
- As just shown...
- Calculate P(data)

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

Learning The Structure of Bayesian Networks

- Search thru the space
- For each structure, learn parameters
- Pick the one that fits observed data best
- As just shown...
- Calculate P(data)

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

Structure Learning as Search

- Local Search
  1. Start with some network structure
  2. Try to make a change (add or delete or reverse edge)
  3. See if the new network is any better
- What should the initial state be?
  - Uniform prior over random networks?
  - Based on prior knowledge?
  - Empty network?
- How do we evaluate networks?

Score Functions

- Bayesian Information Criteion (BIC)
  - \( P(D \mid BN) \) — penalty
  - Penalty = \( \frac{1}{2} \) (# parameters) Log (# data points)

- MAP score
  - \( P(BN \mid D) = P(D \mid BN) P(BN) \)
  - \( P(BN) \) must decay exponentially with # of parameters for this to work well
Topics

- Some Useful Bayes Nets
  - Hybrid Discrete / Continuous
  - Naïve Bayes
- Learning Parameters for a Bayesian Network
  - Fully observable
    - Maximum Likelihood (ML),
    - Maximum A Posteriori (MAP)
  - Bayesian
    - Hidden variables (EM algorithm)
- Learning Structure of Bayesian Networks

Tuning on Held-Out Data

- Now we’ve got two kinds of unknowns
  - Parameters: the probabilities $P(Y|X), P(Y)$
  - Hyperparameters, like the amount of smoothing to do: $k$
- Where to learn?
  - Learn parameters from training data
  - Must tune hyperparameters on different data
  - Why?
    - For each value of the hyperparameters, train and test on the held-out data
    - Choose the best value and do a final test on the test data

Baselines

- First step: get a baseline
  - Baselines are very simple “straw man” procedures
  - Help determine how hard the task is
  - Help know what a “good” accuracy is
- Weak baseline: most frequent label classifier
  - Gives all test instances whatever label was most common in the training set
  - E.g. for spam filtering, might label everything as ham
  - Accuracy might be very high if the problem is skewed
  - E.g. calling everything “ham” gets 66%, so a classifier that gets 70% isn’t very good...
- For real research, usually use previous work as a (strong) baseline

Confidences from a Classifier

- The confidence of a probabilistic classifier:
  - Posterior over the top label
    \[ \text{confidence}(x) = \max_y P(y|x) \]
  - Represents how sure the classifier is of the classification
  - Any probabilistic model will have confidences
  - No guarantee confidence is correct
- Calibration
  - Weak calibration: higher confidences mean higher accuracy
  - Strong calibration: confidence predicts accuracy rate
  - What’s the value of calibration?

Precision vs. Recall

- Let’s say we want to classify web pages as homepages or not
  - In a test set of 1K pages, there are 3 homepages
  - Our classifier says they are all non-homepages
  - 99.7 accuracy!
  - Need new measures for rare positive events
- Precision: fraction of guessed positives which were actually positive
- Recall: fraction of actual positives which were guessed as positive
- Say we detect 5 spam emails, of which 2 were actually spam, and we missed one
  - Precision: 2 correct / 5 guessed = 0.4
  - Recall: 2 correct / 3 true = 0.67
- Which is more important in customer support email automation?
- Which is more important in airport face recognition?
**Precision vs. Recall**

- **Precision/recall tradeoff**
  - Often, you can trade off precision and recall
  - Only works well with weakly calibrated classifiers

- To summarize the tradeoff:
  - **Break-even point**: precision value when \( p = r \)
  - **F-measure**: harmonic mean of \( p \) and \( r \):
    \[
    F_1 = \frac{2}{1/p + 1/r}
    \]

**Errors, and What to Do**

- **Examples of errors**

**What to Do About Errors?**

- Need more features—words aren’t enough!
  - Have you emailed the sender before?
  - Have 1K other people just gotten the same email?
  - Is the sending information consistent?
  - Is the email in ALL CAPS?
  - Do inline URLs point where they say they point?
  - Does the email address you by (your) name?

- Can add these information sources as new variables in the NB model

- Next class we’ll talk about classifiers which let you easily add arbitrary features more easily

**Summary**

- **Bayes rule lets us do diagnostic queries with causal probabilities**

- **The naïve Bayes assumption takes all features to be independent given the class label**

- **We can build classifiers out of a naïve Bayes model using training data**

- **Smoothing estimates is important in real systems**

- **Classifier confidences are useful, when you can get them**

**What to Do About Errors?**

- Need more features—words aren’t enough!
  - Have you emailed the sender before?
  - Have 1K other people just gotten the same email?
  - Is the sending information consistent?
  - Is the email in ALL CAPS?
  - Do inline URLs point where they say they point?
  - Does the email address you by (your) name?

- Can add these information sources as new variables in the NB model

- Next class we’ll talk about classifiers which let you easily add arbitrary features more easily
Summary

- Bayes rule lets us do diagnostic queries with causal probabilities
- The naïve Bayes assumption takes all features to be independent given the class label
- We can build classifiers out of a naïve Bayes model using training data
- Smoothing estimates is important in real systems
- Classifier confidences are useful, when you can get them