15-388/688 - Practical Data Science: Probabilistic modeling

J. Zico Kolter
Carnegie Mellon University
Fall 2016

Outline

Probabilisitic graphical models

Probabilistic inference

Bayesian modeling

Probabilistic programming

Outline

Probabilisitic graphical models

Probabilistic inference

Bayesian modeling

Probabilistic programming

Probabilistic graphical models

Probabilistic graphical models are all about representing distributions

$$p(X)$$

where X represents some large *set* of random variables

Example: suppose $X \in \{0,1\}^n$ (n-dimensional random variable), would take $2^n - 1$ parameters to describe the full joint distribution

Graphical models offer a way to represent these same distributions more compactly, by exploiting *conditional independencies* in the distribution

Note: I'm going to use "probabilistic graphical model" and "Bayesian network" interchangeably, even though there are differences

Bayesian networks

A Bayesian network is defined by

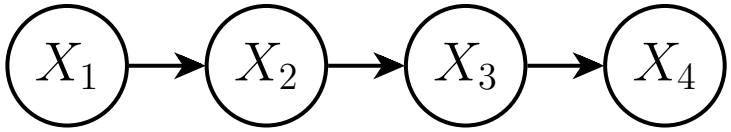
1. A directed acyclic graph, $G = \{V = \{X_1, \ldots, X_n\}, E\}$
2. A set of conditional distributions $p(X_i|\text{Parents}(X_i))$

Defines the joint probability distribution

$$p(X) = \prod_{i=1}^{n} p(X_i|\text{Parents}(X_i))$$

Equivalently: each node is conditionally independent of all non-descendants given its parents

Example Bayesian network



Conditional independencies let us simply the joint distribution:

$$p(X_1, X_2, X_3, X_4) = p(X_1)p(X_2|X_1)p(X_3|X_1, X_2)p(X_4|X_1, X_2, X_3)$$

$2^4 - 1 = 15$ parameters (assuming binary variables)

$$= p(X_1)p(X_2|X_1)p(X_3|X_2)p(X_4|X_3)$$

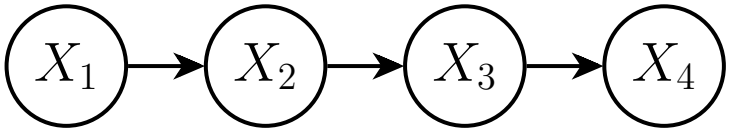
1 parameter

2 parameters

7 parameters

Generative model

Can also describe the probabilistic distribution as a sequential "story", this is called a *generative model*



$$
\begin{aligned}
X_1 &\sim \text{Bernoulli}\left(\phi^{(1)}\right) \\
X_2 | \; X_1 = x_1 &\sim \text{Bernoulli}\left(\phi^{(2)}_{x_1}\right) \\
X_3 | \; X_2 = x_2 &\sim \text{Bernoulli}\left(\phi^{(3)}_{x_2}\right) \\
X_4 | \; X_3 = x_3 &\sim \text{Bernoulli}\left(\phi^{(3)}_{x_3}\right)
\end{aligned}
$$

"First sample X_1 from a Bernoulli distribution with parameter $\phi^{(1)}$, then sample X_2 from a Bernoulli distribution with parameter $\phi^{(2)}_{x_1}$, where x_1 is the value we sampled for X_1, then sample X_3 from a Bernoulli …"

More general generative models

This notion of a “sequential story” (generative model) is extremely powerful for describing very general distributions

Naive Bayes:

$$Y \sim \text{Bernoulli}(\phi)$$
$$X_i | Y = y \sim \text{Categorical}\left(\phi_y^{(i)}\right)$$

Gaussian mixture model:

$$Z \sim \text{Categorical}(\phi)$$
$$X | Z = z \sim \mathcal{N}(\mu_z, \Sigma_z)$$

More general generative models

Linear regression:

$$Y|X = x \sim \mathcal{N}(\theta^T x, \sigma^2)$$

Changepoint model:

$$X \sim \text{Uniform}(0,1)$$

$$Y|X = x \sim \begin{cases} \mathcal{N}(\mu_1, \sigma^2) \text{ if } x < t \\ \mathcal{N}(\mu_2, \sigma^2) \text{ if } x \geq t \end{cases}$$

Latent Dirichlet Allocation: M documents, K topics, N_i words/document

$$\theta_i \sim \text{Dirichlet}(\alpha) \text{ (topic distributions per document)}$$
$$\phi_k \sim \text{Dirichlet}(\beta) \text{ (word distributions per topic)}$$
$$z_{i,j} \sim \text{Categorical}(\theta_i) \text{ (topic of } i\text{th word in document)}$$
$$w_{i,j} \sim \text{Categorical}(\phi_{z_i,j}) \text{ (}i\text{th word in document)}$$

Outline

Probabilisitic graphical models

Probabilistic inference

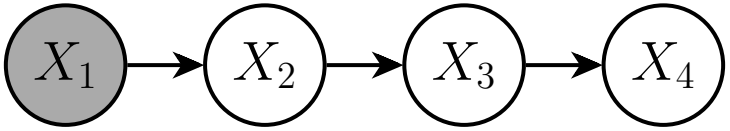
Bayesian modeling

Probabilistic programming

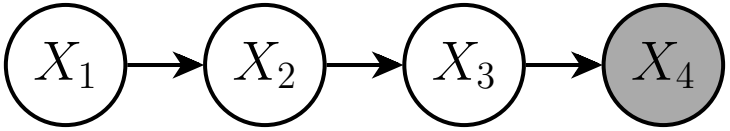
The inference problem

Given observations (i.e., knowing the value of some of the variables in a model), what is the distribution over the other (hidden) variables?

A relatively "easy" problem if we observe variables at the "beginning" of chains in a Bayesian network:



If we observe the value of X_1, then X_2, X_3, X_4 have the same distribution as before, just with X_1 "fixed"



But if we observe X_4 what is the distribution over X_1, X_2, X_3?

Approaches to inference

There are three categories of common approaches to inference (more exist, but these are most common)

1. Exact methods: Bayes' rule or variable elimination methods

2. Approximate variational approaches: approximate distributions over hidden variables using "simple" distributions, minimizing the difference between these distributions and the true distributions

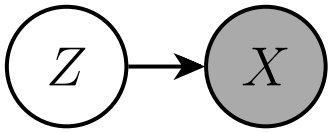
3. Sampling approaches: draw samples from the the distribution over hidden variables, without construction them explicitly

Exact inference example

Mixture of Gaussians model:

$$Z \sim \text{Categorical}(\phi)$$
$$X|Z = z \sim \mathcal{N}(\mu_z, \Sigma_z)$$

Expectation step: compute $p(Z|x)$



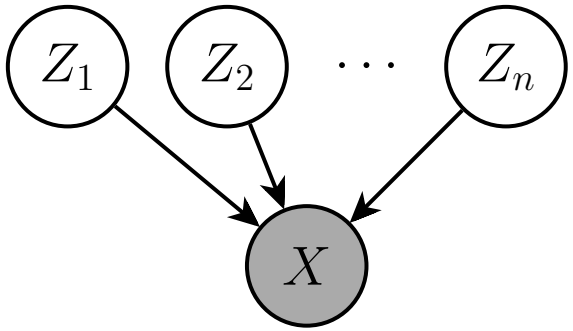
In this case, we can solve inference exactly with Bayes' rule:

$$p(Z|x) = \frac{p(x|Z)p(Z)}{\sum_z p(x|z)p(z)}$$

Need for approximate inference

In most cases, the exact distribution over hidden variables cannot be computed, would require representing an exponentially large distribution over hidden variables (or infinite, in continuous case)

$$Z_i \sim \text{Bernoulli}(\phi_i), \qquad i = 1, \ldots, n$$
$$X|Z = z \sim \mathcal{N}(\theta^T z, \sigma^2)$$

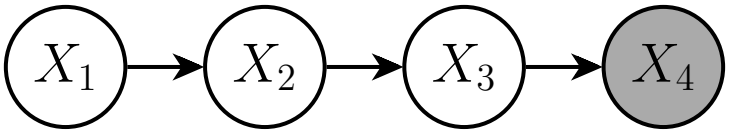


Distribution $P(Z|x)$ is a full distribution over n binary random variables

Sample-based inference

A naive strategy (rejection sampling): draw samples from the generative model until we find one that matches the observed data, distribution over other variables will be samples of the hidden variables given observed variables

As we get more complex models, and more observed variables, probability that we see our exact observations goes to zero



Markov chain Monte Carlo

Markov chain Monte Carlo (MCMC) refers to a class of methods that approximately draw samples from over the hidden variables

The techniques work by iteratively sampling from some of the hidden variables (we'll denote them Z_i) conditioned on others (both other hidden variables Z_i and observed variables X)

Gibbs sampling:

$$\text{Repeat: sample } Z_i \sim P(Z_i | X, Z_{j:j \neq i})$$

Metropolis:

$$\text{Repeat: sample } Z_i' \sim Q(Z_i' | Z_i), u \sim \text{Uniform}[0,1]$$

$$\text{Set: } Z_i \leftarrow Z_i' \text{ if } u < \frac{Q(Z_i | Z_i') P(Z_i' | X, Z_{j:j \neq i})}{Q(Z_i' | Z_i) P(Z_i | X, Z_{j:j \neq i})}$$

Outline

Probabilisitic graphical models

Probabilistic inference

Bayesian modeling

Probabilistic programming

Maximum likelihood estimation

Our discussion of probabilistic modeling thus far has maintained a separation between *variables* and *parameters*

Roughly speaking: variables are the things we take expectations over (or sample), and parameters are the things we optimize

E.g. maximum likelihood estimation required that we solve the problem (given observed data $x^{(i)}$):

$$\underset{\theta}{\text{maximize}} \sum_{i=1}^{m} \log p(x^{(i)}; \theta)$$

Bayesian statistics

In Bayesian statistics, everything (including "parameters" θ) is a random variable, we write likelihoods now as

$$p\left(x^{(i)}\middle|\theta\right)$$

In order for these probabilities to be well-defined, we need to define ***prior distribution*** $p(\theta;\alpha)$ on the "parameters" themselves, where α are hyperparameters (typically fixed and not estimated at all)

Instead of finding a point estimate of θ, in Bayesian statistics we try to quantify the *distribution* of $\theta|X$ (θ given the observed data), called the ***posterior distribution***

$$p(\theta|X) = \frac{p(X|\theta)p(\theta;\alpha)}{\int p(X|\theta)p(\theta;\alpha)d\theta}$$

Bayesian linear regression

Bayesian linear regression model

$$\theta \sim \mathcal{N}(0, \rho I)$$
$$Y|\theta, x \sim \mathcal{N}(\theta^T x, \sigma^2)$$

Without proof, I'll claim that the posterior distribution is given by

$$\theta|x^{(1:m)}, y^{(1:m)} \sim \mathcal{N}(\mu, \Sigma)$$
$$\Sigma = \rho I + \sigma^2 X^T X$$
$$\mu = \sigma^2 \Sigma^{-1} X^T y$$

where X and y and the normal matrix/vector of inputs/outputs

Key point: *posterior* distribution over θ is also Gaussian

Conjugate priors

You may hear this term if you read about Bayesian statistics

All this is saying is the following: suppose

$$\theta \sim F(\alpha) \quad (F \text{ is some distribution})$$
$$X|\theta \sim G(\theta) \ (G \text{ some other distribution})$$

Then if F is a ***conjugate prior*** for G

$$\theta|X \sim F(\alpha')$$

i.e., the posterior has the same type of distribution as the prior

This is quite useful, as it represents just about the only case where we can represent the posterior distribution exactly

Conjugate priors and limitations

Example: Normal distribution is conjugate for mean parameter of Normal (see Bayesian linear regression), Inverse Gamma is conjugate for variance parameter

Example: Beta distribution is conjugate prior for Bernoulli, Dirichlet is conjugate for categorical

In the vast majority of cases, you won't use exact conjugate priors, meaning you can't come up with a closed form distribution for the parameters given the data

Need to resort to approximate inference methods, often sampling

(Simplified) Bayesian changepoint detection

Changepoint detection:

$$X \sim \text{Uniform}(0,1)$$

$$Y|x \sim \begin{cases} \mathcal{N}(\mu_1, \sigma^2) \text{ if } x < t \\ \mathcal{N}(\mu_2, \sigma^2) \text{ if } x \geq t \end{cases}$$

Bayesian changepoint detection:

$$t \sim \text{Uniform}(0,1)$$

$$\mu_1, \mu_2 \sim \mathcal{N}(0, \nu^2)$$

$$\sigma^2 \sim \text{InverseGamma}(\alpha, \beta)$$

$$Y|x \sim \begin{cases} \mathcal{N}(\mu_1, \sigma^2) \text{ if } x < t \\ \mathcal{N}(\mu_2, \sigma^2) \text{ if } x \geq t \end{cases}$$

Outline

Probabilisitic graphical models

Probabilistic inference

Bayesian modeling

Probabilistic programming

Probabilistic programming

In recent years, there has been substantial effort to “automate” the specification of probabilistic models and inference within these models

In probabilistic programming languages, users specify the model similar to writing code, specify the observed variables (if any), and then perform inference (usually sampling-based) to compute posterior

The PyMC framework (https://pymc-devs.github.io/pymc/) is one such language/framework for Python

(Bayesian) Changepoint detection in PyMC

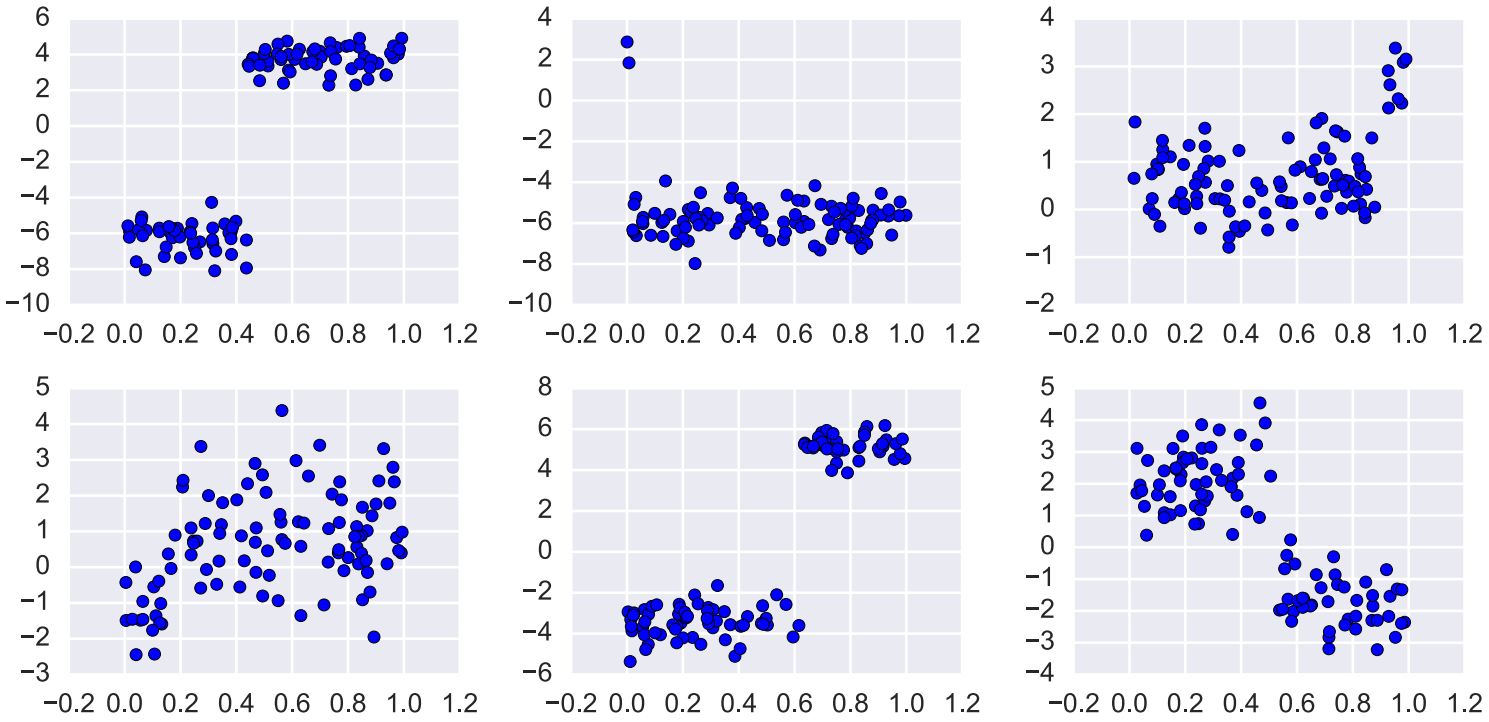
Model of changepoint detection generative model in PyMC:

```
N = 100
t = pm.Uniform("t", 0, 1)
mu1 = pm.Normal("mu1", 0, 0.1)
mu2 = pm.Normal("mu2", 0, 0.1)
tau = pm.Gamma("tau", 2.0, 1.0)

x = pm.Container([pm.Uniform("x_{}".format(i), 0, 1) for i in range(N)])
y = pm.Container([pm.Normal("y_{}".format(i),
                            (x[i]<t)*mu1 + (x[i]>=t)*mu2, tau)
                  for i in range(N)])

model = pm.Model([t,mu1,mu2,tau,x,y])
```

Run MCMC to generate samples:

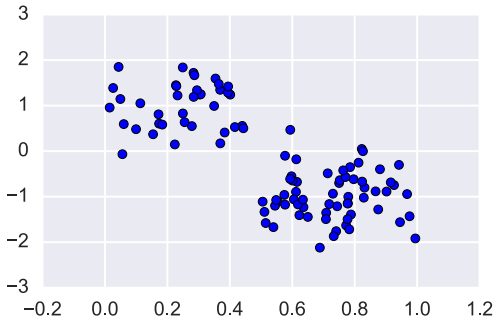
```
mcmc = pm.MCMC(model)
mcmc.sample(100)
```

Samples from Generative model



Adding observations

Suppose we see the following values for x,y



Add observed values in PyMC

```
...
x = pm.Container([pm.Uniform("x_{}".format(i), 0, 1,
                             observed=True, value=x0[i])
                  for i in range(N)])
y = pm.Container([pm.Normal("y_{}".format(i),
                            (x[i]<t)*mu1 + (x[i]>=t)*mu2, tau,
                            observed=True, value=y0[i])
                  for i in range(N)])
...
mcmc.sample(10000, burn=100) # throw away first 100 samples
```

Posterior distributions