CPSC 440: Advanced Machine Learning

More Monte Carlo

Mark Schmidt

University of British Columbia

Winter 2021

Last Time: Monte Carlo Methods

- Given density estimator, we often want to make probabilistic inferences:
  - Marginals: what is the probability that $x_j = c$?
    - What is the probability we're in industry 10 years after graduation?
  - Conditionals: what is the probability that $x_j = c$ given $x_{j'} = c'$?
    - What is the probability of industry after 10 years, if we immediately go to grad school?

- A basic Monte Carlo method for estimating probabilities of events:
  1. Generate a large number of samples x^i from the model,

$$X = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{bmatrix}.$$

     - Last time we discussed inverse transform and ancestral sampling.
  2. Compute frequency that the event happened in the samples,

$$p(x_2 = 1) \approx 3/4,$$
$$p(x_3 = 0) \approx 0/4.$$

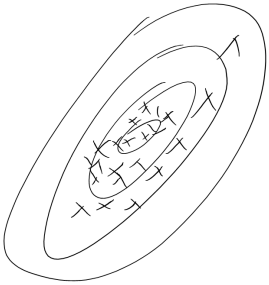
Monte Carlo Method for Inequalities

- Monte Carlo estimate of probability that variable is above threshold:
  - Compute fraction of examples where sample is above threshold.

Monte Carlo Method for Mean

- A Monte Carlo approximation of the mean:
  - Approximate the mean by average of samples.

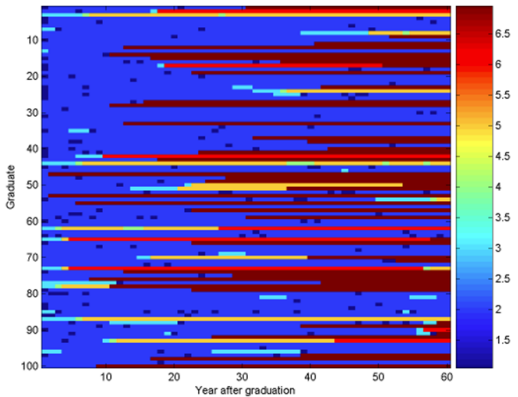
$$E[x] \approx \frac{1}{n} \sum_{i=1}^{n} x^i.$$



- Visual demo of Monte Carlo approximation of mean and vairance:
  - `http://students.brown.edu/seeing-theory/basic-probability/index.html`

Monte Carlo for Markov Chains

- Our samples from the CS grad student Markov chain:



- We can estimate probabilities by looking at frequencies in samples.
  - In how many out of the 100 chains did we have $x_{10} =$ "industry"?
- This works for continuous states too (for inequalities and expectations).

Monte Carlo Methods for Markov Chains

- Some Monte Carlo approximations of inference tasks in Markov chains:
  - Marginal $p(x_j = c)$ is the number of chains that were in state c at time j.
  - Average value at time j, $E[x_j]$, is approximated by average of x_j in the samples.
  - $p(5 \leq x_j \leq 10)$ is approximate by frequency of x_j being between 5 and 10.
    - This makes more sense for continuous states than evaluating equalities.
  - $p(x_j \leq 10, x_{j+1} \geq 10)$ is approximated by number of chains where both happen.

Monte Carlo Methods: General Form

- Monte Carlo methods approximate expectations of random functions,

$$\underbrace{\mathbb{E}[g(x)] = \sum_{x \in \mathcal{X}} g(x)p(x)}_{\text{discrete } x} \quad \text{or} \quad \underbrace{\mathbb{E}[g(x)] = \int_{x \in \mathcal{X}} g(x)p(x)dx}_{\text{continuous } x}.$$

- Computing mean is the special case of $g(x) = x$.
- Computing probability of any event A is also a special case:
  - Set $g(x) = \mathcal{I}[\text{“}A \text{ happened in sample } x^i\text{”}]$, indicator function for event A.

- To approximate expectation, generate n samples x^i from $p(x)$ and use:

$$\mathbb{E}[g(x)] \approx \frac{1}{n} \sum_{i=1}^{n} g(x^i).$$

Unbiasedness of Monte Carlo Methods

- Let $\mu = \mathbb{E}[g(x)]$ be the value we want to approximate (not necessarily mean).
- The Monte Carlo estimate is an unbiased approximation of μ,

$$\begin{aligned}
\mathbb{E}\left[\frac{1}{n}\sum_{i=1}^{n} g(x^i)\right] &= \frac{1}{n}\mathbb{E}\left[\sum_{i=1}^{n} g(x^i)\right] && \text{(linearity of } \mathbb{E}\text{)} \\
&= \frac{1}{n}\sum_{i=1}^{n} \mathbb{E}[g(x^i)] && \text{(linearity of } \mathbb{E}\text{)} \\
&= \frac{1}{n}\sum_{i=1}^{n} \mu && (x^i \text{ is IID with mean } \mu) \\
&= \mu.
\end{aligned}$$

- The law of large numbers says that:
  - Unbiased approximators "converge" (probabilistically) to expectation as $n \to \infty$.
  - So the more samples you get, the closer to the true value you expect to get.

Rate of Convergence of Monte Carlo Methods

- Let f be the squared error in a 1D Monte Carlo approximation,

$$f(x^1, x^2, \ldots, x^n) = \left(\frac{1}{n} \sum_{i=1}^{n} g(x^i) - \mu \right)^2 .$$

- If variance is bounded, error with n samples is $O(1/n)$,

$$\begin{aligned}
\mathbb{E}\left[\left(\frac{1}{n} \sum_{i=1}^{n} g(x^i) - \mu \right)^2 \right] &= \mathsf{Var}\left[\frac{1}{n} \sum_{i=1}^{n} g(x^i) \right] && \text{(unbiased and def'n of variance)} \\
&= \frac{1}{n^2} \mathsf{Var}\left[\sum_{i=1}^{n} g(x^i) \right] && (\mathsf{Var}(\alpha x) = \alpha^2 \mathsf{Var}(x)) \\
&= \frac{1}{n^2} \sum_{i=1}^{n} \mathsf{Var}[g(x^i)] && \text{(IID)} \\
&= \frac{1}{n^2} \sum_{i=1}^{n} \sigma^2 = \frac{\sigma^2}{n}. && (x^i \text{ is IID with var } \sigma^2)
\end{aligned}$$

- Similar $O(1/n)$ argument holds for $d > 1$ (notice that faster for small σ^2).

Conditional Probabilities with Monte Carlo

- We often want to compute conditional probabilities in Markov chains.
  - We can ask "what lead to $x_{10} = 4$?" with queries like $p(x_1 \mid x_{10} = 4)$.
  - We can ask "where does $x_{10} = 4$ lead?" with queries like $p(x_d \mid x_{10} = 4)$.

- Monte Carlo approach to estimating $p(x_j \mid x_{j'})$:
  1. Generate a large number of samples from the Markov chain, $x^i \sim p(x_1, x_2, \ldots, x_d)$.
  2. Use Monte Carlo estimates of $p(x_j = c, x_{j'} = c')$ and $p(x_{j'} = c')$ to give

$$p(x_j = c \mid x_{j'} = c') = \frac{p(x_j = c, x_{j'} = c')}{p(x_{j'} = c')} \approx \frac{\sum_{i=1}^{n} I[x_j^i = c, x_{j'}^i = c']}{\sum_{i=1}^{n} I[x_{j'}^i = c']},$$

  frequency of first event in samples consistent with second event.

- This is a special case of rejection sampling (we'll see general case later).
  - Unfortunately, if $x_{j'} = c'$ is rare then most samples are "rejected" (ignored).

http://students.brown.edu/seeing-theory/compound-probability/index.html

Outline

Exact Marginal Calculation

- In typical settings Monte Carlo has slow convergence like stochastic gradient.
  - $O(1/t)$ convergence rate where constant is variance of samples.
    - If all samples look the same, it converges quickly.
    - If samples look very different, it can be painfully slow.

- For discrete-state Markov chains, we can actually compute marginals directly:
  - We're given initial probabilities $p(x_1 = s)$ for all s as part of the definition.
  - We can use transition probabilities to compute $p(x_2 = s)$ for all s:

$$p(x_2) = \underbrace{\sum_{x_1=1}^{k} p(x_2, x_1)}_{\text{marginalization rule}} = \sum_{x_1=1}^{k} \underbrace{p(x_2 \mid x_1)p(x_1)}_{\text{product rule}}.$$

Exact Marginal Calculation

- We can do a similar calculation to compute $p(x_3)$:

$$\begin{aligned} p(x_3 = s) &= \sum_{x_2=1}^{k} \sum_{x_1=1}^{k} p(x_1, x_2, x_3) \\ &= \sum_{x_2=1}^{k} \sum_{x_1=1}^{k} p(x_3 \mid x_2) p(x_2 \mid x_1) p(x) \\ &= \sum_{x_2=1}^{k} p(x_3 \mid x_2) \sum_{x_1=1}^{k} p(x_2 \mid x_1) p(x) \\ &= \sum_{x_2=1}^{k} p(x_3 \mid x_2) p(x_2). \end{aligned}$$

- We can also derive this recursively,

$$p(x_3) = \underbrace{\sum_{x_2=1}^{k} p(x_3, x_2)}_{\text{marginalization rule}} = \sum_{x_2=1}^{k} \underbrace{p(x_3 \mid x_2) p(x_2)}_{\text{product rule}},$$

which is simpler but more-complicated scenarios won't yield a simple recursion.

Exact Marginal Calculation

- Recursive formula for maginals at time j:

$$p(x_j) = \sum_{x_{j-1}=1}^{k} p(x_j \mid x_{j-1}) p(x_{j-1}),$$

  called the Chapman-Kolmogorov (CK) equations.

- The CK equations can be implemented as matrix-vector multiplication:
  - Define π^j as a vector containing the marginals at time t:

$$\pi_c^j = p(x_j = c).$$

  - Define T^j as a matrix cotaining the transition probabilities:

$$T_{cc'}^j = p(x_j = c \mid x_{j-1} = c').$$

Exact Marginal Calculation

- Implementing the CK equations as a matrix multiplications:

$$T^j \pi^{j-1} = \begin{bmatrix} p(x_j = 1|x_{j-1} = 1) & p(x_j = 1|x_{j-1} = 2) & \dots & p(x_j = 1|x_{j-1} = k) \\ p(x_j = 2|x_{j-1} = 1) & p(x_j = 2|x_{j-1} = 2) & \dots & p(x_j = 2|x_{j-1} = k) \\ p(x_j = k|x_{j-1} = 1) & p(x_j = k|x_{j-1} = 2) & \dots & p(x_j = k|x_{j-1} = k) \end{bmatrix} \begin{bmatrix} p(x_{j-1} = 1) \\ p(x_{j-1} = 2) \\ \vdots \\ p(x_{j-1} = k) \end{bmatrix}$$

$$= \begin{bmatrix} \sum_{c=1}^k p(x_j = 1 \mid x_{j-1} = c) p(x_{j-1} = c) \\ \sum_{c=1}^k p(x_j = 2 \mid x_{j-1} = c) p(x_{j-1} = c) \\ \vdots \\ \sum_{c=1}^k p(x_j = k \mid x_{j-1} = c) p(x_{j-1} = c) \end{bmatrix} = \begin{bmatrix} p(x_j = 1) \\ p(x_j = 2) \\ \vdots \\ p(x_j = k) \end{bmatrix} = \pi^j.$$

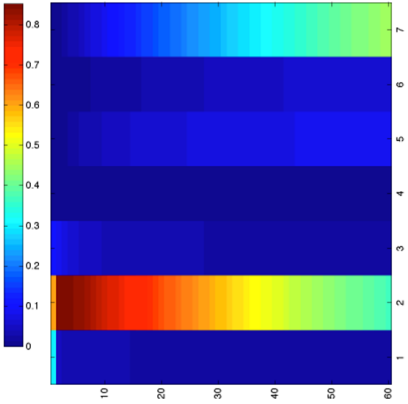
- Cost of multiplying a vector by a $k \times k$ matrix is $O(k^2)$.

- So cost to compute marginals up to time d is $O(dk^2)$.
  - This is fast considering that last step sums over all k^d possible paths.

$$p(x_d) = \sum_{x_1=1}^{k} \sum_{x_2=1}^{k} \cdots \sum_{x_{j-1}=1}^{k} \sum_{x_{j+1}=1}^{k} \cdots \sum_{x_{d-1}=1}^{k} p(x_1, x_2, \dots, x_d).$$

Marginals in CS Grad Career

- CK equations can give all marginals $p(x_j = c)$ from CS grad Markov chain:



- Each row j is a state and each column c is a year.

Continuous-State Markov Chains

- The CK equations also apply if we have continuous states:

$$p(x_j) = \int_{x_{j-1}} p(x_j \mid x_{j-1}) p(x_{j-1}) dx_{j-1},$$

  but this integral may not have a closed-form solution.

- Gaussian probabilities are an important special case:
  - If $p(x_{j-1})$ and $p(x_j \mid x_{j-1})$ are Gaussian, then $p(x_j)$ is Gaussian.
    - Joint distribution is a product of Gaussians.
  - So we can write $p(x_j)$ in closed-form in terms of mean and variance.

- If the probabilities are non-Gaussian, usually can't represent $p(x_j)$ distribution.
  - You are stuck using Monte Carlo or other approximations.

Stationary Distribution

- A stationary distribution of a homogeneous Markov chain is a vector π satisfying

$$\pi(c) = \sum_{c'} p(x_j = c \mid x_{j-1} = c')\pi(c').$$

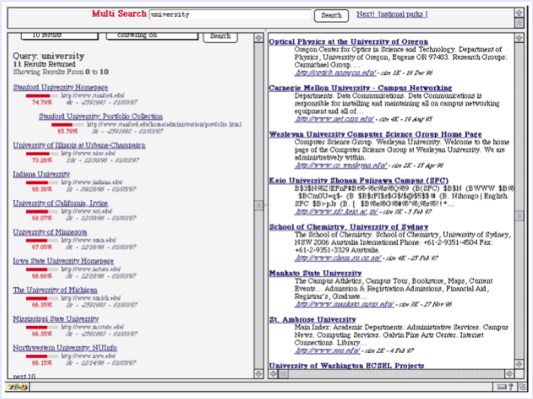
- "Probabilities don't change across time" (also called "invariant" distribution).
  - Here we are talking about the "marginal" probabilities $p(x_j)$, not the "transition" probabilities $p(x_j \mid x_{j-1})$.

- Under certain conditions, marginals converge to a stationary distribution.
  - $p(x_j = c) \to \pi(c)$ as j goes to ∞.
  - If we fit a Markov chain to the rain example, we have $\pi(\text{"rain"}) = 0.41$.
  - In the CS grad student example, we have $\pi(\text{"dead"}) = 1$.

- Stationary distribution is basis for Google's PageRank algorithm.

Application: PageRank

- Web search before Google:

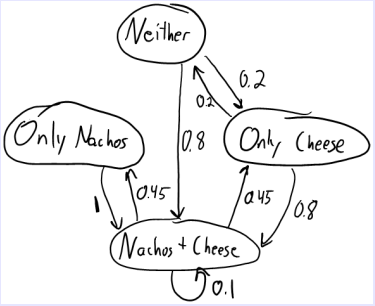


http://ilpubs.stanford.edu:8090/422/1/1999-66.pdf

- It was also easy to fool search engines by copying popular websites.

State Transition Diagram

- State transition diagrams are common for visualizing homogenous Markov chains:



$$P = \begin{bmatrix} 0 & 0 & 0.2 & 0.8 \\ 0 & 0 & 0 & 1 \\ 0.2 & 0 & 0 & 0.8 \\ 0 & 0.45 & 0.45 & 0.1 \end{bmatrix}$$

- Each node is a state, each edge is a non-zero transition probability.
  - For web-search, each node will be a webpage.
- Cost of CK equations is only $O(z)$ instead of $O(k^2)$ if you have only z edges.

Application: PageRank

- Wikipedia's cartoon illustration of Google's PageRank:
  - Large face means higher rank.

https://en.wikipedia.org/wiki/PageRank

- "Important webpages are linked from other important webpages".
- "Link is more meaningful if a webpage has few links".

Application: PageRank

- Google's PageRank algorithm for measuring the importance of a website:
  - Stationary probability in "random surfer" Markov chain:
    - With probability α, surfer clicks on a random link on the current webpage.
    - Otherwise, surfer goes to a completely random webpage.
- To compute the stationary distribution, they use the power method:
  - Repeatedly apply the CK equations.
  - Iterations are faster than $O(k^2)$ due to sparsity of links.
    - Transition matrix is "sparse plus rank-1" which allows fast multiplication.
  - Can be easily parallelized.

Application: Game of Thrones

- PageRank can be used in other applications.
- "Who is the main character in the Game of Thrones books?"

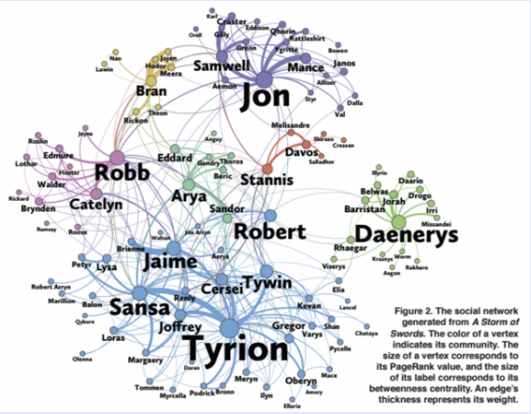


Figure 2. The social network generated from *A Storm of Swords*. The color of a vertex indicates its community. The size of a vertex corresponds to its PageRank value, and the size of its label corresponds to its betweenness centrality. An edge's thickness represents its weight.

http://qz.com/650796/mathematicians-mapped-out-every-game-of-thrones-relationship-to-find-the-main-character

Existence/Uniqueness of Stationary Distribution

- Does a stationary distribution π exist and is it unique?
- A sufficient condition for existence/uniqueness is that all $p(x_j = c \mid x_{j'} = c') > 0$.
  - PageRank satisfies this by adding probability $(1 - \alpha)$ of jumping to a random page.
- Weaker sufficient conditions for existence and uniqueness ("ergodic"):
  1. "Irreducible" (doesn't get stuck in part of the graph).
  2. "Aperiodic" (probability of returning to state isn't on fixed intervals).

Summary

- Monte Carlo samples to approximate expectations of random functions.
  - Tke average of the function applied to each sample.
- Chapman-Kolmogorov equations compute exact univariate marginals.
  - For discrete or Gaussian Markov chains.
- Stationary distribution of homogenous Markov chain.
  - Marginals as time goes to ∞.
  - Basis of Google's PageRank method.
- Next time: voice Photoshop.

Monte Carlo as a Stochastic Gradient Method

- Consider case of using Monte Caro method to estimate mean $\mu = \mathbb{E}[x]$,

$$\mu \approx \frac{1}{n}\sum_{i=1}^{n} x^i.$$

- We can write this as minimizing the 1-strongly convex

$$f(w) = \frac{1}{2}\|w - \mu\|^2.$$

- The gradient is $\nabla f(w) = (w - \mu)$.
- Consider applying stochastic gradient descent on f using

$$\nabla f_i(w^k) = w^k - x^{k+1},$$

which is unbiased since each x^i is unbiased μ approximation.

- Monte Carlo method is a stochastic gradient method with this approximation.

Monte Carlo as a Stochastic Gradient Method

- Monte Carlo approximation as a stochastic gradient method with $\alpha_i = 1/(i+1)$,

$$\begin{aligned}
w^n &= w^{n-1} - \alpha_{n-1}(w^{n-1} - x^i) \\
&= (1 - \alpha_{n-1})w^{n-1} + \alpha_{n-1}x^i \\
&= \frac{n-1}{n}w^{n-1} + \frac{1}{n}x^i \\
&= \frac{n-1}{n}\left(\frac{n-2}{n-1}w^{n-2} + \frac{1}{n-1}x^{i-1}\right) + \frac{1}{n}x^i \\
&= \frac{n-2}{n}w^{n-2} + \frac{1}{n}\left(x^{i-1} + x^i\right) \\
&= \frac{n-3}{n}w^{n-3} + \frac{1}{n}\left(x^{i-2} + x^{i-1} + x^i\right) \\
&= \frac{1}{n}\sum_{i=1}^{n} x^i.
\end{aligned}$$

- We know the rate of stochastic gradient for strongly-convex is $O(1/n)$.

Law of the Unconscious Statistician

- We use these identities to define the expectation of a function g applied to a random variable x,

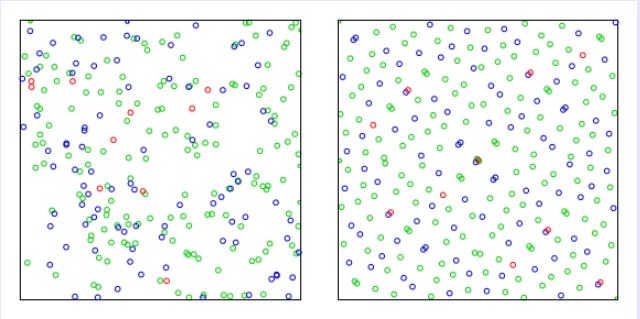
$$\underbrace{\mathbb{E}[g(x)] = \sum_{x \in \mathcal{X}} g(x)p(x)}_{\text{discrete } x} \quad \text{or} \quad \underbrace{\mathbb{E}[g(x)] = \int_{x \in \mathcal{X}} g(x)p(x)dx}_{\text{continuous } x}.$$

- The transformation from expectation to sum/integral is known as the "law of the unconsciuos statistician".
  - It's usually taken as being true, but it's proof is a bit of a pain.

Accelerated Monte Carlo: Quasi Monte Carlo

- Unlike stochastic gradient, there are some "accelerated" Monte Carlo methods.

- Quasi Monte Carlo methods achieve an accelerated rate of $O(1/n^2)$.
  - Key idea: fill the space strategically with a deterministic "low-discrepancy sequence".
  - Uniform random vs. deterministic low-discrepancy:



https://en.wikipedia.org/wiki/Quasi-Monte_Carlo_method

Label Propagation as a Markov Chain Problem

- Semi-supervised label propagation method has a Markov chain interpretation.
  - We have $n + t$ states, one for each [un]labeled example.
- Monte Carlo approach to label propagation ("adsorption"):
  - At time $t = 0$, set the state to the node you want to label.
  - At time $t > 0$ and on a labeled node, output the label.
    - Labeled nodes are absorbing states.
  - At time $t > 0$ and on an unlabeled node i:
    - Move to neighbour j with probability proportional w_{ij} (or $\bar{w}_{ij}$).
- Final predictions are probabilities of outputting each label.
  - Nice if you only need to label one example at a time (slow if labels are rare).
  - Common hack is to limit random walk time to bound runtime.