More categorical variables and Monte Carlo

CPSC 440/550: Advanced Machine Learning

`cs.ubc.ca/~dsuth/440/23w2`

University of British Columbia, on unceded Musqueam land

2023-24 Winter Term 2 (Jan–Apr 2024)

Outline

Motivation: probabilistic inference

- Given a general model, we often want to make inferences
  - Marginals: what's the probability that $X_i = c$?
  - Conditionals: what's the probability that $X_i = c$, given that $X_{i'} = c'$?

- This has been simple for the models we've seen so far
  - For Bernoulli/categorical, computing probabilities is straightforward
  - For product of Bernoullis (or categoricals), assumed everything is independent

- For many models, inference has no closed form or might be NP-hard
- In these cases, we'll often use Monte Carlo approximations

Monte Carlo: marginalization by sampling

- A basic Monte Carlo method for estimating probabilities of events:
- Step 1: Generate a lot of samples $x^{(i)}$ from our model

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

- Step 2: Count how often the event occurred in the samples

$$\Pr(X_2 = 1) \approx \frac{3}{4} \qquad \Pr(X_3 = 0) \approx 0$$

- This very simple idea is one of the most important algorithms in ML/statistics
- Modern versions developed to build better nuclear weapons :/
  - "Sample" from a physics simulator, see how often it leads to a chain reaction

Monte Carlo to approximate probabilities

- Monte Carlo estimate of the probability of an event A:

$$\frac{\text{number of samples where } A \text{ happened}}{\text{number of samples}} = \frac{1}{n}\sum_{i=1}^{n} \mathbb{1}(A \text{ happened in } x^{(i)})$$

- You can think of this as the MLE of a binary variable $\mathbb{1}(A \text{ happened})$
- Approximating probability of a pair of independent dice adding to 7:
  - Roll two dice, check if they add to 7
  - Roll two dice, check if they add to 7
  - Roll two dice, check if they add to 7
  - Roll two dice, check if they add to 7
  - Roll two dice, check if they add to 7
  - Roll two dice, check if they add to 7
  - ...
  - Monte Carlo estimate: fraction where they add to 7

Monte Carlo to approximate probabilities

- Recall the problem of modeling (Lib, CPC, NDP, GRN, PPC)
- From 100 samples, what's the probability that $n_{\texttt{Lib}} > \max(n_{\texttt{CPC}}, n_{\texttt{NDP}}, \ldots)$?

- Can answer this in closed form with math . . . or think less and do Monte Carlo
  - Generate 100 samples, check who won
  - Generate 100 samples, check who won
  - . . .
  - Approximate probability by fraction of times they won

- Another example: probability that $\mathrm{Beta}(\alpha, \beta)$ is above 0.7

Monte Carlo to estimate the mean

- A Monte Carlo estimate for the mean: the mean of the samples

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

- A Monte Carlo approximation of the expected value of X^2:

$$\mathbb{E}[X^2] \approx \frac{1}{n}\sum_{i=1}^{n} \left(x^{(i)}\right)^2$$

- A Monte Carlo approximation of the expected value of $g(X)$:

$$\mathbb{E}[g(X)] \approx \frac{1}{n}\sum_{i=1}^{n} g\left(x^{(i)}\right) \qquad \mathbb{E}[g(X)] = \sum_{x\in\mathcal{X}} p(x)g(x) \text{ or } \int_{x\in\mathcal{X}} p(x)g(x)\mathrm{d}x$$

  - Most general form: $g(x) = x$, $g(x) = x^2$, $g(x) = \mathbb{1}(A \text{ happens on } x)$

$$\mathbb{E}[\mathbb{1}(A \text{ happens on } x)] = \int_{x\in\mathcal{X}} p(x)\,\mathbb{1}(A \text{ happens on } x)\mathrm{d}x = \int_{x:A \text{ happens}} p(x)\mathrm{d}x = \Pr(A)$$

Monte Carlo: theory

- Let $\mu = \mathbb{E}[g(X)]$ be the value we want to compute, $\hat{\mu}$ our estimate
- Assume $\sigma^2 = \mathrm{Var}[g(X)]$ exists and is bounded ("not infinite")

- With iid samples, Monte Carlo gives an unbiased estimate of μ
  - Expected value of $\hat{\mu}$, over samples we might draw, is exactly μ

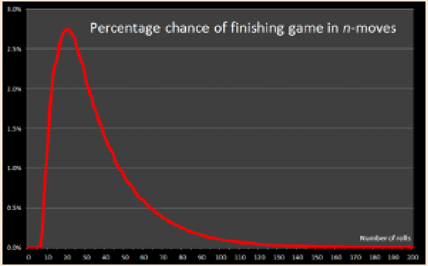
- Monte Carlo estimate "converges to μ" as $n \to \infty$
  - Estimate gets arbitrarily close to μ as n increases: (strong) law of large numbers

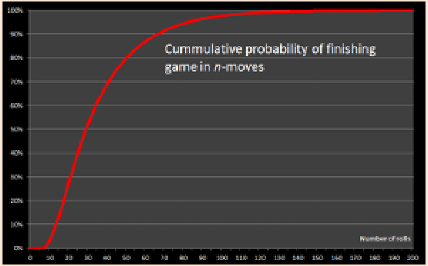
- Expected squared error is exactly $\mathbb{E}(\hat{\mu} - \mu)^2 = \frac{\sigma^2}{n}$
- $\hat{\mu}$ is *approximately* normal with mean μ and variance $\frac{\sigma^2}{n}$ (central limit theorem)

Example application: Snakes and Ladders

bonus!

- Kid's game "Snakes and Ladders":
  - Start at 1, roll die, move the marker, follow snake/ladder
  - Absolutely no decision-making: can simulate the game
- How long does this game go for?
  - Run the game lots of times, see how many turns it took





https://www.datagenetics.com/blog/november12011/

Conditional probabilities with Monte Carlo

- "How much loooonger will this game go?"
  - Just simulate starting from current game state

- "What's the probability the game will go >100 turns, if it's already gone 50?"
- One approach:

$$\Pr(A \mid B) = \frac{\Pr(A \cap B)}{\Pr(B)} \approx \frac{\frac{1}{n}\sum_{i=1}^{n} \mathbb{1}(A \text{ and } B \text{ happened on } x^{(i)})}{\frac{1}{n}\sum_{i=1}^{n} \mathbb{1}(B \text{ happened on } x^{(i)})}$$

- This is one instance of rejection sampling (more later)
- If B is rare, most samples are wasted

Outline

MLE for categorical distribution

- How do we learn a categorical model?

$$\mathbf{X} = \begin{bmatrix} \texttt{NDP} \\ \texttt{Lib} \\ \texttt{Lib} \\ \texttt{CPC} \\ \vdots \end{bmatrix} \xrightarrow{\text{density estimator}} \boldsymbol{\theta} = \begin{bmatrix} \Pr(X = \texttt{Lib}) = 0.404 \\ \Pr(X = \texttt{NDP}) = 0.307 \\ \Pr(X = \texttt{CPC}) = 0.216 \\ \Pr(X = \texttt{Grn}) = 0.039 \\ \Pr(X = \texttt{PPC}) = 0.032 \end{bmatrix}$$

- Like before, start with maximum likelihood estimation (MLE):

$$\hat{\boldsymbol{\theta}} \in \arg\max_{\theta} p(\mathbf{X} \mid \boldsymbol{\theta})$$

- Like before, MLE will be $\theta_c = \frac{n_c}{n}$ (the portion of cs in the data)
- Like before, derivation is more complicated than the result

Derivation of the MLE that doesn't work

- We showed last time that the likelihood is

$$p(\mathbf{X} \mid \boldsymbol{\theta}) = \theta_1^{n_1} \cdots \theta_k^{n_k}$$

- So, the log-likelihood is

$$\log p(\mathbf{X} \mid \boldsymbol{\theta}) = n_1 \log \theta_1 + \cdots + n_k \log \theta_k$$

- Take the derivative for a particular θ_c:

$$\frac{\partial}{\partial \theta_c} \log p(\mathbf{X} \mid \boldsymbol{\theta}) = \frac{n_c}{\theta_c}$$

- Set the derivative to zero:

$$\frac{n_c}{\theta_c} = 0$$

- ... huh?

Fixing the derivation

- Setting the derivative to zero doesn't work
  - Ignores the constraint that $\sum_c \theta_c = 1$
- Some ways to enforce constraints (see e.g. this StackExchange thread):
  - Use "Lagrange multipliers," find stationary point of the "Lagrangian"
  - Define $\theta_k = 1 - \sum_{c=1}^{k-1} \theta_c$, replace in the objective function
- We'll take a different way:
  - Use a different parameterization $\tilde{\theta}_c$ that doesn't have this constraint
  - Compute the MLE for the $\tilde{\theta}_c$ by setting derivative to zero
  - Convert from the $\tilde{\theta}_c$ to θ_c

Unnormalized parameterization

- Let's have $\tilde{\theta}_c$ be unnormalized:

$$\Pr(X = c \mid \tilde{\theta}_1, \ldots, \tilde{\theta}_k) \propto \tilde{\theta}_c$$

  - Still need each $\tilde{\theta}_c \geq 0$
- Can then find

$$p(c \mid \tilde{\boldsymbol{\theta}}) = \frac{\tilde{\theta}_c}{\sum_{i=1}^{k} \tilde{\theta}_c} = \frac{\tilde{\theta}_c}{Z_{\tilde{\boldsymbol{\theta}}}}$$

- The "normalizing constant" $Z_{\tilde{\boldsymbol{\theta}}}$ makes the total probability 1
  - Don't need the explicit sum-to-1 constraint anymore
  - Note: constant for different x; **not** constant for different $\boldsymbol{\theta}$
- To convert from unnormalized to normalized: $\theta_c = \tilde{\theta}_c / Z_{\tilde{\boldsymbol{\theta}}}$

Derivation of the MLE that does work

- The likelihood in terms of the unnormalized parameters is

$$p(\mathbf{X} \mid \tilde{\boldsymbol{\theta}}) = \left(\frac{\tilde{\theta}_1}{Z_{\tilde{\boldsymbol{\theta}}}}\right)^{n_1} \cdots \left(\frac{\tilde{\theta}_k}{Z_{\tilde{\boldsymbol{\theta}}}}\right)^{n_k} = \frac{1}{Z_{\tilde{\boldsymbol{\theta}}}^n} \tilde{\theta}_1^{n_1} \cdots \tilde{\theta}_k^{n_k}$$

- So, the log-likelihood is

$$\log p(\mathbf{X} \mid \tilde{\boldsymbol{\theta}}) = n_1 \log \tilde{\theta}_1 + \cdots + n_k \log \tilde{\theta}_k - n \log Z_{\tilde{\boldsymbol{\theta}}}$$

- Take the derivative for a particular $\tilde{\theta}_c$:

$$\frac{\partial}{\partial \tilde{\theta}_c} \log p(\mathbf{X} \mid \tilde{\boldsymbol{\theta}}) = \frac{n_c}{\tilde{\theta}_c} - \frac{n}{Z_{\tilde{\boldsymbol{\theta}}}} \frac{\partial Z_{\tilde{\boldsymbol{\theta}}}}{\partial \tilde{\boldsymbol{\theta}}_c} = \frac{n_c}{\tilde{\theta}_c} - \frac{n}{Z_{\tilde{\boldsymbol{\theta}}}} \qquad \text{since } \frac{\partial}{\partial \tilde{\theta}_c}\left(\tilde{\theta}_1 + \cdots + \tilde{\theta}_k\right) = 1$$

- Set the derivative to zero:

$$\frac{n_c}{\tilde{\theta}_c} = \frac{n}{Z_{\tilde{\boldsymbol{\theta}}}} \quad \text{so} \quad \frac{\tilde{\theta}_c}{Z_{\tilde{\boldsymbol{\theta}}}} = \frac{n_c}{n}$$

  - Can check this objective is concave, so this is a max
  - Many solutions, but all the same after normalizing

MAP estimate, Dirichlet prior

- As before, might prefer MAP estimate over MLE
- Often becomes more important for large k: lots of parameters!

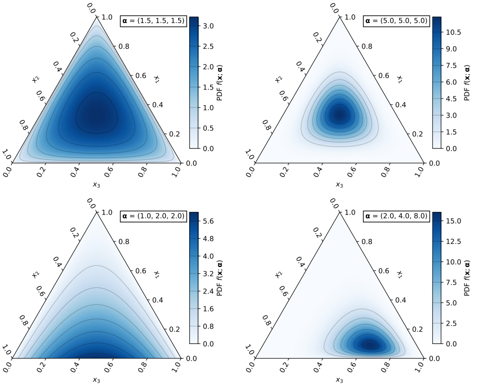
- Most common prior is the Dirichlet distribution:

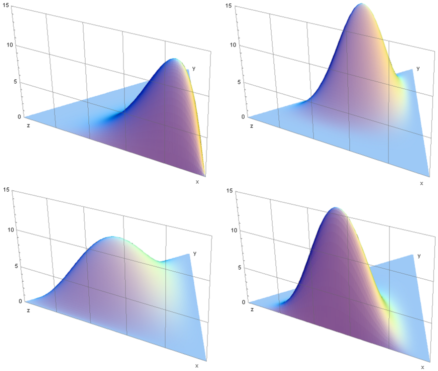
$$p(\theta_1, \ldots, \theta_k \mid \alpha_1, \ldots, \alpha_k) \propto \theta_1^{\alpha_1 - 1} \cdots \theta_k^{\alpha_k - 1}$$

  - Generalization of the beta distribution to k classes
  - Requires each $\alpha_c > 0$
- This is a distribution over $\boldsymbol{\theta}$
  - Probability distribution over possible (categorical) probability distributions

Dirichlet distribution

- Wikipedia's visualizations for $k = 3$:





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

MAP estimate for Dirichlet-Categorical

- Reason to use the Dirichlet: again because posterior is simple

$$p(\boldsymbol{\theta} \mid \mathbf{X}, \boldsymbol{\alpha}) \propto p(\mathbf{X} \mid \boldsymbol{\theta}) p(\boldsymbol{\theta} \mid \boldsymbol{\alpha}) \propto \theta_1^{n_1} \cdots \theta_k^{n_k} \, \theta_1^{\alpha_1 - 1} \cdots \theta_k^{\alpha_k - 1}$$
$$= \theta_1^{(n_1 + \alpha_1) - 1} \cdots \theta_k^{(n_k + \alpha_k) - 1}$$

  i.e. it's Dirichlet again with parameters $\tilde{\alpha}_c = n_c + \alpha_c$

- A few more steps show MAP for a categorical with Dirichlet prior is

$$\hat{\theta}_c = \frac{n_c + \alpha_c - 1}{\sum_{c'=1}^{k} (n_{c'} + \alpha_{c'} - 1)}$$

- Dirichlet has k hyper-parameters α_c
  - Often use $\alpha_c = \alpha$ for some $\alpha \in \mathbb{R}$: one hyperparameter
  - Makes the MLE $\hat{\theta}_c = \dfrac{n_c + \alpha - 1}{n + k(\alpha - 1)}$
  - $\alpha = 2$ gives Laplace smoothing (add 1 “fake” count for each class)

Conjugate priors

- This is our second example where prior and posterior have the same form
  - Beta prior + Bernoulli likelihood gives a Beta posterior
    - Also happens with binomial, geometric, ... likelihoods
  - Dirichlet prior + categorical likelihood gives a Dirichlet posterior
    - Also happens with multinomial likelihood

- When this happens, we say prior is conjugate to the likelihood
- Prior and posterior come from the same "family" of distributions

$$X \sim L(\theta) \quad \theta \sim P(\lambda) \qquad \text{implies} \quad \theta \mid X \sim P(\lambda')$$

  - Updated parameters λ will depend on the data

- Many computations become easier if we have a conjugate prior
- But not all distributions have conjugate priors
  - And even when one exists, might not be convenient

Outline

Multi-class classification

- Often have classification with categorical labels and/or features

$$\mathbf{X} = \begin{bmatrix} \texttt{Cough} & \texttt{Low fever} & \texttt{Normal breathing} \\ \texttt{Cough} & \texttt{High fever} & \texttt{Shortness of breath} \\ \texttt{No cough} & \texttt{High fever} & \texttt{Normal breathing} \\ \texttt{No cough} & \texttt{Low fever} & \texttt{Normal breathing} \\ \texttt{Cough} & \texttt{Medium fever} & \texttt{Normal breathing} \end{bmatrix} \qquad \mathbf{y} = \begin{bmatrix} \texttt{Cold} \\ \texttt{Pneumonia} \\ \texttt{Covid} \\ \texttt{Covid} \\ \texttt{Cold} \end{bmatrix}$$

- Can adapt all of our previous binary classification methods:
  - Naïve Bayes
  - Tabular probabilities
  - Logistic regression / neural nets

Product of categoricals, multi-class Naïve Bayes

- Start: multivariate categorical density estimation
  - Input: n iid samples of categorical vectors $x^{(1)}, \ldots, x^{(n)}$
  - Output: model giving probability for any assignment of values $x_1, \ldots, x_d$

$$\mathbf{X} = \begin{bmatrix} \mathtt{A} & \mathtt{C} & \mathtt{C} & \mathtt{T} & \mathtt{T} & \mathtt{T} & \mathtt{A} & \mathtt{G} & \mathtt{C} \\ \mathtt{A} & \mathtt{C} & \mathtt{C} & \mathtt{G} & \mathtt{T} & \mathtt{T} & \mathtt{A} & \mathtt{G} & \mathtt{G} \\ \mathtt{A} & \mathtt{C} & \mathtt{C} & \mathtt{T} & \mathtt{T} & \mathtt{T} & \mathtt{A} & \mathtt{G} & \mathtt{C} \\ \mathtt{A} & \mathtt{A} & \mathtt{C} & \mathtt{T} & \mathtt{T} & \mathtt{T} & \mathtt{C} & \mathtt{G} & \mathtt{G} \end{bmatrix} \xrightarrow{\text{density estimator}} \begin{matrix} \vdots \\ \Pr(X_1 = \mathtt{A}, X_2 = \mathtt{C}, \ldots, X_9 = \mathtt{C}) = 0.11 \\ (4^9 \text{ possible values}) \end{matrix}$$

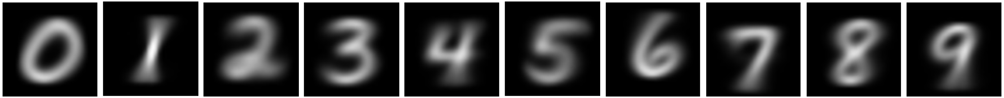
- Like for product of Bernoullis, we could use product of categoricals
  - Assumes X_j are mutually independent: strong assumption that makes things easy

$$\Pr(X_1 = c_1, \ldots, X_d = c_d) = \Pr(X_1 = c_1) \ldots \Pr(X_d = c_d) = \theta_{1,c} \cdots \theta_{d,c_d}$$

  - Parameter $\theta_{j,c}$ is probability that jth entry is in cth class
- Like before, could use product of categoricals conditional on Y to get categorical naïve Bayes

Multi-class naïve Bayes on MNIST

- Binarized MNIST: label is categorical, but images are still product of Bernoullis
- Parameter of the Bernoulli for each class:



- One sample from each class:

Tabular probabilities for categorical data

- Can use a tabular parameterization: with two binary features, three-way label,

$$
\begin{array}{lll}
\Pr(Y=1 \mid X_1=0, X_2=0) = \theta_{1|00} & \Pr(Y=2 \mid X_1=0, X_2=0) = \theta_{2|00} & \Pr(Y=3 \mid X_1=0, X_2=0) = \theta_{3|00} \\
\Pr(Y=1 \mid X_1=0, X_2=1) = \theta_{1|01} & \Pr(Y=2 \mid X_1=0, X_2=1) = \theta_{2|01} & \Pr(Y=3 \mid X_1=0, X_2=1) = \theta_{3|01} \\
\Pr(Y=1 \mid X_1=1, X_2=0) = \theta_{1|10} & \Pr(Y=2 \mid X_1=1, X_2=0) = \theta_{2|10} & \Pr(Y=3 \mid X_1=1, X_2=0) = \theta_{3|10} \\
\Pr(Y=1 \mid X_1=1, X_2=1) = \theta_{1|11} & \Pr(Y=2 \mid X_1=1, X_2=1) = \theta_{2|11} & \Pr(Y=3 \mid X_1=1, X_2=1) = \theta_{3|11}
\end{array}
$$

  - Don't necessarily need $\theta_{3|x}$; can use $\theta_{3|x} = 1 - \theta_{1|x} - \theta_{2|x}$
- MLE has simple closed form: $\hat{\theta}_{y|x} = n_{y|x}/n_x$
  - Just the categorical MLE for each condition
- Can use a Dirichlet (or whatever other) prior and do MAP
- Will overfit unless you have small number of distinct x

Parameterizing conditionals

- Tabular treats each $\theta_{y|x}$ totally separately
- Could instead share information for "similar" x
  - Can no longer express every possible distribution, potentially computationally harder
  - Statistically much easier to fit

- One choice: weight w_c for **each** class, get $z_c = w_c^\mathsf{T} x$ for each c
- Need to turn the z_c into parameters of a categorical distribution
- Binary data: mapped one z into $(0, 1)$ with sigmoid $f(z) = 1/(1 + \exp(-z))$
  - But using $\theta_c = f(z_c)$ won't sum to one
- Softmax function first makes nonnegative by taking $\exp$, then normalizes:

$$\theta_c = [\operatorname{softmax}(\mathbf{z})]_c = \frac{\exp(z_c)}{\sum_{c'=1}^{k} \exp(z_{c'})} \propto \exp(z_c)$$

  - Don't *have* to use softmax, other options exist, but this is default

Categorical features as inputs

- How do we use categorical data in the features x?
- Usually convert to set of binary features ("one-hot" / "one of k" encoding)

Age	City	Income
23	Van	26,000
25	Sur	67,000
19	Bur	16,500
43	Sur	183,000

$\rightarrow$

Age	Van	Bur	Sur	Income
23	1	0	0	26,000
25	0	0	1	67,000
19	0	1	0	16,500
43	0	0	1	183,000

- If you see a new category in test data: usually, just set *all* of them to zero

Softmax and binary logistic regression

- With two categories: using a "dummy" value $z_2 = 0$

$$[\mathrm{softmax}\left((z, 0)\right)]_1 = \frac{\exp(z)}{\exp(z) + \exp(0)} \times \frac{\exp(-z)}{\exp(-z)} = \frac{1}{1 + \exp(-z)}$$

- Two-class softmax regression with one weight frozen at zero is logistic regression

Softmax loss

review

- Taking the negative log-likelihood:

$$-\log p(\mathbf{y} \mid W, \mathbf{X}) = -\sum_{i=1}^{n} \log p(y^{(i)} \mid W, x^{(i)}) = -\sum_{i=1}^{n} \left[\log \left(\frac{\exp\left(w_{y^{(i)}}^{\mathsf{T}} x^{(i)}\right)}{\sum_{c=1}^{k} \exp(w_c^{\mathsf{T}} x^{(i)})} \right) \right]$$

$$= \sum_{i=1}^{n} \left[-w_{y^{(i)}}^{\mathsf{T}} x^{(i)} + \log \left(\sum_{c=1}^{k} \exp(w_c^{\mathsf{T}} x^{(i)}) \right) \right]$$

    - Convex (note log-sum-exp is convex), differentiable: can use gradient descent
    - Often add a regularizer (i.e. a prior on W)
- Gradient has a nice form:

$$\frac{\partial}{\partial w_c}[-\log p(\mathbf{y} \mid W, \mathbf{X})] = -\sum_{i=1}^{n} \mathbb{1}(y^{(i)} = c) x^{(i)} + \sum_{i=1}^{n} \underbrace{\frac{\exp(w_c^{\mathsf{T}} x^{(i)})}{\sum_{c'=1}^{k} \exp(w_{c'}^{\mathsf{T}} x^{(i)})}}_{p(y^{(i)}=c \mid x^{(i)}, W)} x^{(i)}$$

Multi-label versus multi-class classification

- Before: we saw multi-label classification, where y is a binary vector of length k
  - "This image has a chair and a person, but no frog"
- Multi-class, e.g. with softmax loss: y has exactly one of k discrete labels
  - "This is an image of a frog"
- Could have multiple categorical labels (some of which might be binary)
  - "This paper's arXiv primary class is `stat.ML`, and the first author is a student"

Summary

- Monte Carlo is a general way to estimate expectations when you can sample
  - Including probabilities: expectations of indicators

- Next time: everything is regularization

Law of the Unconscious Statistician

bonus!

- These inequalities sometimes called "Law of the Unconscious Statistician":

$$\mathbb{E}[g(X)] = \sum_{x \in \mathcal{X}} g(x)p(x) \qquad \mathbb{E}[g(X)] = \int_{x \in \mathcal{X}} g(x)p(x)\mathrm{d}x$$

- Two explanations I've heard for "unconscious":
  - You can compute expectations without thinking
  - Or: people don't realize this is actually a theorem to prove, not a definition

$$Y = g(X)$$

$$\mathbb{E}[Y] = \sum_y y \Pr(Y = y) = \sum_y y \sum_{x : g(x) = y} p(x) = \sum_x g(x)p(x)$$

Mean and Variance of Monte Carlo

bonus!

- $\hat{\mu} = \frac{1}{n}\sum_{i=1}^{n} g(x^{(i)})$ is an unbiased estimate of $\mu = \mathbb{E}[g(x)]$:

$$\mathbb{E}\left[\frac{1}{n}\sum_{i=1}^{n} g(x^{(i)})\right] = \frac{1}{n}\sum_{i=1}^{n}\mathbb{E}\left[g(x^{(i)})\right] = \frac{1}{n}\sum_{i=1}^{n}\mu = \mu$$

- If $\mathrm{Var}(g(x^{(i)})) = \sigma^2$ for some finite σ, then

$$\mathrm{Var}\left(\frac{1}{n}\sum_{i=1}^{n} g(x^{(i)})\right) = \frac{1}{n^2}\sum_{i=1}^{n}\underbrace{\mathrm{Var}\left(g(x^{(i)})\right)}_{\sigma^2} - \frac{2}{n^2}\sum_{i\neq j}\underbrace{\mathrm{Cov}\left(g(x^{(i)}), g(x^{(j)})\right)}_{0}$$

$$= \frac{1}{n^2}n\sigma^2 = \frac{\sigma^2}{n}$$

- Expected squared error is σ^2/n:

$$\mathbb{E}\left[\left(\frac{1}{n}\sum_{i=1}^{n} g(x^{(i)}) - \mu\right)^2\right] = \left(\mathbb{E}\frac{1}{n}\sum_{i=1}^{n} g(x^{(i)}) - \mu\right)^2 + \mathrm{Var}\left(\frac{1}{n}\sum_{i=1}^{n} g(x^{(i)})\right) = 0 + \frac{\sigma^2}{n}$$

Monte Carlo as a stochastic gradient method

bonus!

Can view as SGD on $f(\hat{\mu}) = \frac{1}{n}\|\hat{\mu} - \mu\|^2$ with learning rate $\frac{1}{i+1}$:

$$\begin{aligned}
\hat{\mu}_n &= \hat{\mu}_{n-1} - \frac{1}{n}\left(\hat{\mu}_{n-1} - x^{(i)}\right) \\
&= \left(1 - \frac{1}{n}\right)\hat{\mu}_{n-1} + \frac{1}{n}x^{(i)} \\
&= \frac{n-1}{n}\left(\frac{1}{n-1}\sum_{i=1}^{n-1} x^{(i)}\right) + \frac{1}{n}x^{(i)} \\
&= \frac{1}{n}\sum_{i=1}^{n-1} x^{(i)} + \frac{1}{n}x^{(i)} = \frac{1}{n}\sum_{i=1}^{n} x^{(i)}
\end{aligned}$$