

Aula 9

Aula passada

- Método da rejeição (*rejection sampling*)
- Exemplos
- *Importance Sampling*
- Exemplos
- Generalização

Aula de hoje

- *Self-normalized Importance Sampling*
- Gerando amostras complicadas
- Variância amostral
- Simulação

Importance Sampling (IS)

- Calcular valor esperado da função h , onde X tem distribuição dada por f

$$\mu_f = E_f[h(X)] = \sum_i^N h(i)f(i)$$

- **Problemas:** N é muito grande; f é difícil de gerar amostras; $h(i)$ não “combina bem” com $f(i)$
- **Solução:** usar outra distribuição g para amostrar
- Seja g distribuição para v.a. X , tal que $f(i) > 0 \rightarrow h(i) > 0$

$$\mu_f = \sum_i \frac{h(i)f(i)}{g(i)} g(i) = E_g \left[\frac{h(X)f(X)}{g(X)} \right]$$

- Podemos estimar E_g usando MC para aproximar μ_f

Algoritmo

- Monte Carlo para estimar $\mu_f = E_g[h(x)f(x)/g(x)]$:

$S = 0;$

para $i = 1, \dots, n$

 Gerar amostra x com distribuição g ;

$S = S + h(x)f(x)/g(x);$

retorne S/n

- Se g for bem escolhida, variância do estimador com IS pode ser menor que estimador original!
 - outra razão para usar IS

Distribuições Normalizadas

- Em muitos casos, função de probabilidade f é da forma

$$f(i) = \frac{z(i)}{C} \quad , \text{ onde } C = \sum_i^N z(i)$$

- C é chamada de constante de normalização

- ex. distribuição Zeta(α) , $C_\alpha = \sum_i^\infty 1/i^\alpha$

- Muito comum em distribuições empíricas

- seja X um perfil no FB, $f_X(i)$ tem probabilidade proporcional ao número de amigos do perfil i
- $z(i)$ = número de amigos do perfil i , N é o total de perfis

Constante de Normalização

- **Problema:** em muitos casos, não sabemos a constante de normalização C
 - no FB, para calcular C precisaria passar por todos os 2 bilhões de perfis!
- Como estimar um valor esperado que utiliza f ?
 - Seja h uma função qualquer, queremos

$$\mu_f = E_f[h(X)] = \sum_i^N h(i)f(i) = 1/C \sum_i^N h(i)z(i)$$

Importance Sampling

Self-Normalized Importance Sampling

- Seja $f(i) = z(i)/C$ para uma constante C

$$\mu_f = \frac{1/C \sum_i h(i) z(i)}{1/C \sum_i z(i)} =$$

$$\frac{\frac{1/C \sum_i h(i) z(i)}{g(i)} g(i)}{\frac{1/C \sum_i z(i)}{g(i)} g(i)} = \frac{E_g \left[\frac{h(X) z(X)}{g(X)} \right]}{E_g \left[\frac{z(X)}{g(X)} \right]} = \frac{E_g [h(X) w(X)]}{E_g [w(X)]}$$

$$, \text{ onde } w(i) = \frac{z(i)}{g(i)}$$

Self-Normalized Importance Sampling

- w é chamada de função de peso, relação entre $z(i)$ e $g(i)$
- Não precisamos conhecer C – *incrível!*
- Podemos estimar μ_f usando estimadores para seguintes valores esperados

$$E_g[h(X)w(X)] \quad E_g[w(X)]$$

- Algoritmo

para $i = 1, \dots, n$

Gerar amostra x com distribuição g

$$S1 = S1 + h(x)z(x)/g(x)$$

$$S2 = S2 + z(x)/g(x)$$

$$\text{returne } (S1/n)/(S2/n) = S1/S2$$

Qualidade do Estimador

- Temos os seguintes estimadores

$$\hat{\mu}_{hw} = \frac{1}{n} \sum_i^n h(X_i) w(X_i) \quad \hat{u}_w = \frac{1}{n} \sum_i^n w(X_i) \longrightarrow \hat{u}_f = \frac{\hat{u}_{hw}}{\hat{u}_w}$$

- Qual é o valor esperado do estimador?

$$E[\hat{u}_f] = E\left[\frac{\hat{u}_{hw}}{\hat{u}_w}\right] \neq \frac{E[\hat{u}_{hw}]}{E[\hat{u}_w]} \quad \text{Que é o que nós temos!}$$

- Ou seja, nosso estimador não tem como valor esperado a quantia que desejamos estimar!
 - chamado de estimador enviesado (*biased estimator*)
- Boa notícia: no limite as diferenças somem
 - chamado de estimador consistente (*consistent estimator*)

Qualidade do Estimador

- Temos o seguinte

$$\mu_f = E_f[h(X)]$$

$$\hat{u}_f^n = \frac{1/n \sum_i^n h(X_i) w(X_i)}{1/n \sum_i^n w(X_i)}$$

- Teorema:

$$P(\lim_{n \rightarrow \infty} \hat{\mu}_f^n = \mu_f) = 1$$

- Prova

- amostras X_i com distribuição g , trazer C de volta
- numerador: converge para μ_f (lei dos grandes números)
- denominador: converge para 1 (lei dos grds números)
- Estimador é assintoticamente sem viés (*asymptotically unbiased*)

Gerando Amostras Complicadas

- Até agora, assumimos conhecimento da função de probabilidade, $f(i)$
 - ou ao menos de sua constante de normalização
- **Problema:** em muitos casos, não temos esta função
 - Máquina caça níquel no cassino
 - Você chega com 20 reais
 - Cada rodada custa 1 real, e possivelmente dá recompensa
 - Quantas rodadas de jogo até você perder tudo (inclusive o que ganhou)?
 - Sim, você vai perder tudo se jogar o suficiente!



Amostras no Cassino

- Seja T v.a. que denota o número de rodadas até você perder tudo quando você chega com 20 reais
 - queremos calcular $E[T]$
- **Problema:** não temos função de probabilidade de T , ou seja $f(i) = P[T = i]$, para $i = 20, 21, \dots$
 - $f(20) = P[T = 20] =$ perder todas as vezes consecutivas
 - $f(40) = P[T = 40] = ?$
- Seja R v.a. que denota o retorno da máquina (possivelmente zero), com função de probabilidade g , ou seja $g(j) = P[R = j]$
- Seja S o valor total remanescente
- A cada rodada, valor total diminui de um e aumenta de R
- Rodadas são independentes

Amostras no Cassino

- Seja m o valor inicial (no caso, 20 reais)
- Podemos definir a soma iterativamente

$$S_k = m + \sum_{j=1}^k (R_j - 1) \quad , \text{ onde } R_j \text{ é a recompensa na rodada } j$$

- E definir o valor para nossa v.a. T

$$T = \min \left\{ k \mid m + \sum_{j=1}^k (R_j - 1) \leq 0 \right\}$$

- primeira rodada onde a soma é menor ou igual a zero
- Como estimar $E[T]$?

Simular as rodadas!

- sabemos gerar amostras para R , pois g é dada

Algoritmo

- Estimador de $E[T]$ usando a média amostral

$T = 0;$

para $i = 1, \dots, n$

$S = m; t = 0;$

enquanto ($S > 0$)

 Gerar amostra r com distribuição g

$S = S + r - 1; t = t + 1;$

$T = T + t;$

retorna T/n

- Convergência depende da variância de T

- necessário para definir n , número de amostras

- Quanto vale $\text{Var}[T]$?

Variância do Estimador

- Sabemos que estimador via média amostral tem variância

$$\hat{\mu}_T^n = 1/n \sum_{i=1}^n T_i \quad \text{Var}[\hat{\mu}_T^n] = \frac{\sigma_T^2}{n}$$

- Não sabemos σ_T^2 , mesmo se soubermos σ_R^2
- Podemos estimar σ_T^2 - *Monte Carlo to the rescue!*
- Definição

$$\sigma_T^2 = \text{Var}[T] = E[(X - \mu_T)^2]$$

- Estimador (ingênuo, veremos)

$$\hat{\sigma}_{T,n}^2 = 1/n \sum_{i=1}^n (T_i - \mu_T)^2 \quad \leftarrow$$

Mas não temos μ_T , mas podemos usar seu estimador!

Estimando a Variância

- Estimador (ingênuo, veremos)

$$\hat{\sigma}_{T,n}^2 = 1/n \sum_{i=1}^n (T_i - \hat{\mu}_T^n)^2 \quad \xleftarrow{\text{Usando o estimador para valor esperado}}$$

- Qualidade do estimador: se for sem viés, seu valor esperado é igual ao valor sendo estimado para todo n , no caso σ_T^2

$$\begin{aligned} E[\hat{\sigma}_{T,n}^2] &= E[1/n \sum_{i=1}^n (T_i - \hat{\mu}_T^n)^2] = 1/n \sum_{i=1}^n E[(T_i - \hat{\mu}_T^n)^2] \\ &= 1/n \sum_{i=1}^n E[(T_i^2 - 2T_i \hat{\mu}_T^n + (\hat{\mu}_T^n)^2)] = \dots = \frac{n-1}{n} \sigma_T^2 \end{aligned}$$

- Este estimador é *enviesado*! Mas viés é de fácil correção
 - fator multiplicativo de $n/(n-1)$ corrige o viés

Variância Amostral

- Estimador para variância sem viés (*sample variance*)

$$\hat{S}_{T,n}^2 = \frac{n}{n-1} \hat{\sigma}_{T,n}^2 = \frac{1}{n-1} \sum_{i=1}^n (T_i - \hat{\mu}_T^n)^2$$

- Que pode ser calculado mais facilmente como

$$M_1 = \sum_{i=1}^n T_i \quad M_2 = \sum_{i=1}^n (T_i)^2$$

$$\hat{S}_{T,n}^2 = \frac{M_2 - M_1^2/n}{n-1} \quad \hat{\mu}_T^n = \frac{M_1}{n}$$

- Variância amostral de T pode ser usada para calcular variância do estimador $\hat{\mu}_T^n$, e com isto definir n

Gerando Amostras Complicadas

- Primeiro lance na sinuca
- Ver vídeo:
<https://www.youtube.com/watch?v=iR8BClwp5fE>
- Aleatoriedade: posição da bola branca, local de contato, ângulo do taco, velocidade do taco
- Todo o resto é física Newtoniana (determinístico)
- Quantas bolas são encaçapadas? Valor esperado?



**Simular o sistema
(até bolas pararem)**

Simulação de Sistemas

- **Simulador:** gerador sofisticado de amostras que são difíceis de gerar!
 - simular comportamento de sistema complicado a partir de seus componentes básicos
- Método de Monte Carlo é a teoria que sustenta simulação
 - Geradores de variáveis aleatórias, estimadores baseados em média amostral e variância amostral
- Técnica amplamente utilizada para avaliar sistemas com eventos aleatórios
 - lógica depende do simulador depende do domínio, não é muito fácil generalizar