

Переборные методы: сэмплирование

И. Куралёнок, Н. Поваров

Яндекс

СПб, 2013

Содержание

- 1 Понятие сэмплирования
 - Процесс сэмплирования
 - Основные методы сэмплирования
- 2 Переборные методы в ML
 - Полный перебор
- 3 Hill climbing
- 4 Сэмплирование марковскими цепями
 - Metropolis-Hastings алгоритм
 - Алгоритм Гиббса
- 5 Построение вероятностного пространства для максимизации

Понятие сэмплирования

Сэмплирование — метод исследования множества путём анализа его подмножеств.

Применяется когда:

- множество слишком велико для перебора;
- каждое дополнительное измерение дорого;
- предварительный анализ.

Алгоритм сэмплирования

- 1 Понять какое множество мы изучаем
- 2 Осознать что из этого множества мы можем измерить
- 3 Определить количество измерений
- 4 Разработать план сэмплирования
- 5 Провести сэмплирование

Типы сэмплирования

- Вероятностное сэмплирование:

$$p(x), \forall x : p(x) > 0$$

Например: попробуем посчитать соотношение мужчин/женщин

- Невероятностное сэмплирование:

$$p(x), \exists x : p(x) = 0$$

Например: “по результатам опроса superjob.ru, 100% россиян пользуются интернетом”

- Без возвратов
- С возвратами

Виды сэмплирования

- Вероятностное сэмплирование
 - Простое вероятностное
 - Систематическое
 - Стратифицированное
 - Пропорциональное
 - Кластерное
- Невероятностное сэмплирование
 - Опрос ближайших
 - Панельное сэмплирование

Как выбрать нужное?

Надо учитывать:

- природа и размер возможного сэмпла;
- наличие дополнительной информации об элементах;
- необходимая точность измерений;
- точность отдельных измерений в сэмпле;
- стоимость измерений.

Возвращаясь к ML

$$F_0 = \arg \max_F p(F|X)$$

- + если известны вероятности можно попробовать посэмплировать решения;
- не определено пространство F ;
- неясно как устроить обход.

Иногда все просто

$$F_0 = \arg \max_{F \in \{f_i\}_{i=1}^n} p(F|X)$$

- 1 введём порядок обхода;
- 2 переберём все возможные решения;
- 3 составим взвешенное решение/выберем лучшее.

Но чаще всё непросто

$$F_0 = \arg \max_{F \in \{f_i\}_{i=1}^{\infty}} p(F|X)$$

- 1 введём порядок обхода;
- 2 применим систематическое сэмплирование;
- 3 составим взвешенное решение/выберем лучшее.

Случайное блуждание I

Чтобы построить порядок обхода можно воспользоваться такой схемой:

$$\begin{aligned}F &= F(x, \lambda), \lambda \in \mathbb{R}^n \\F_t &= F(x, \lambda_t) \\ \lambda_{t+1} &= \lambda_t + \xi \quad C(\lambda_{t+1} | \{\lambda_i\}_0^t)\end{aligned}$$

- будем блуждать по пространству параметров;
- необходимо определить:
 - 1 способ сделать шаг;
 - 2 условие принятия этого шага.

Случайное блуждание II

На что стоит обратить внимание при построении блуждания:

- размерность λ может быть меньше чем кажется;
- ограничения на λ существенно осложняют процедуру.

Некоторые виды случайного блуждания

- множество фиксированных шагов $\xi \sim U(\{\xi_i\}_1^m)$;
- гауссовское $\xi_t \sim N(\mu, \sigma^2)$;
- самозависимое;
- etc.

Simple hill climbing

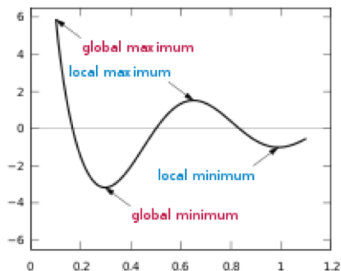
$$\xi \sim U(\{\xi_i\}_1^{2n}), \xi_{i\frac{i}{2}} = -1^{i \bmod 2} \omega, \xi_{ij} = 0, j \neq \frac{i}{2},$$

$$C(\lambda_{t+1}|\lambda_t) = \frac{p(F(\lambda_{t+1})|X)}{p(F(\lambda_t)|X)} > 1$$

Свойства:

- простой;
- быстро сходится;
- зависим от выбора начальной точки.
- etc.

Random-restart (shotgun) hill climbing



Проблемы:

- сходится в локальный максимум;
- может долго сходиться, если начало далеко от максимума;
- аллеи.

⇒ Можно рестартить hill climbing из разных начальных точек

Интуиция

Мы бы хотели получить сэмплирование, а для этого:

- хорошо бы обойти всё пространство;
- нельзя всегда ходить "по шерсти";
- скорость движения должна меняться в зависимости от плотности.

⇒ Markov Chain Monte-Carlo (MCMC)

Metropolis-Hastings алгоритм

Введем $p(\lambda_1|\lambda_2)$, отвечающую за локальность.

$$\alpha = \frac{p(F_{\lambda_{t+1}}|X)p(\lambda_{t+1}|\lambda_t)}{p(F_{\lambda_t}|X)p(\lambda_t|\lambda_{t+1})}$$

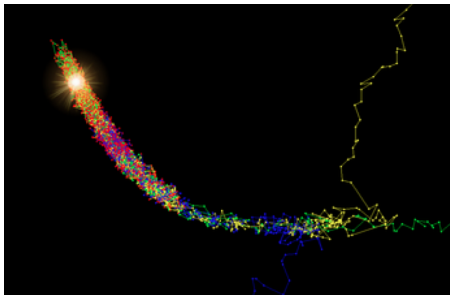
$$\psi \sim U(0, 1)$$

$$C(\lambda_{t+1}|\lambda_t) = \begin{cases} 1, & \alpha \geq \psi \\ 0 & \end{cases}$$

Например, $p(\lambda_{t+1}|\lambda_t) \sim N(\lambda_t, \sigma^2 E)$

Если $p(\lambda_1|\lambda_2) = p(\lambda_2|\lambda_1)$ — это Metropolis

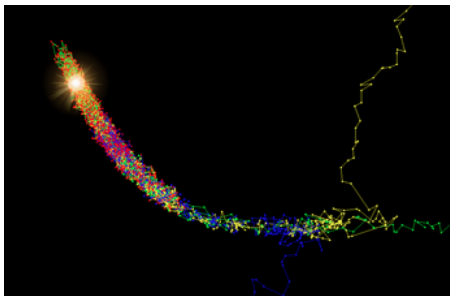
Свойства



- + Обходит всё пространство
- + Это действительно взвешенное сэмплирование
- Последовательные сэмплы похожи друг на друга
- На этапе разогрева показывает что-то странное

⇒ *Точно придём в максимум!*

Свойства



- + Обходит всё пространство
- + Это действительно взвешенное сэмплирование
- Последовательные сэмплы похожи друг на друга
- На этапе разогрева показывает что-то странное

⇒ *Точно придём в максимум!*

Проблема только с тем, что придём за бесконечное время

Сложности в использовании

- Сходимость зависит от выбора $p(\lambda_{t+1}|\lambda_t)$
- Если хотим использовать разумное распределение, оно многомерное \Rightarrow его сложно реализовывать

Пример с дартс I

Вася и Петя повесели на стене мишень и поиграли в дартс. На следующий день пришел их бригадир Юра и задался вопросом кто из подчиненных будет заделывать дырки в стенах.

- Будет честнее, чтобы “косой” платил больше
- Дырки все одинаковые
- Каждый говорит: “да я токо разок кинул!”
- Один признался, что: “ну вот это — моя дырка”
- Играли без употребления, так что можно считать, что кидали $\sim N(m_i, \Sigma_i)$, $i \in \{\text{Вася, Петя}\}$ и параметры не менялись во времени

Надо помочь Юре!

Пример с дартс II

При фиксированных параметрах вероятность увидеть дырки $\{x_j\}$, $x_j \in \mathbb{R}^2$

$$LL = \sum_j \log(\pi N(m_A, \Sigma_A)(x_j) + (1 - \pi)N(m_B, \Sigma_B)(x_j))$$

где A — Вася, B — Петя, π — доля Васиных бросков, m_i — сбитость прицела, Σ_i — разброс стрелка.

Максимизировать такое добро трудно из-за суммы под логарифмом. Было бы классно точно знать, что тот или иной бросок точно сделал Вася $A_t \in \{0, 1\}$:

$$LL = \sum_t A_t \log(N(m_A, \Sigma_A)(x_j)) + (1 - A_t) \log(N(m_B, \Sigma_B)(x_j))$$

В таком варианте все совсем просто считается (если мы еще не забыли формулу плотности нормального многомерного распределения :))

Пример с дартс III

Если чуть более формально, то мы ввели скрытую/ненаблюдаемую переменную $I(A)$ (такое называется data augmentation) и хотим:

- 1 Прикинуть начальные m_i, Σ_i, π
- 2 Посчитать ожидание скрытого параметра $A_j = \mu((m_i, \Sigma_i, \pi))$
- 3 В условиях найденных A_j максимизировать LL
- 4 Перейти к второму шагу до сходимости

Expectation maximization алгоритм

Фокус, который мы проделали называется *EM*-алгоритм. Пускай данные у нас состоят из видимой части L и скрытой Z , $X = (L, Z)$. Мы хотим найти оптимальные параметры λ :

$$LL(p(L|\lambda)) = LL(p(X|\lambda))$$

- 1 Возьмем какой-то λ_0
- 2 Посчитаем ожидание (expectation step):

$$E(\lambda|\lambda_t) = \mu(LL(\lambda, X)|Z, \lambda_t)$$

- 3 Проведем оптимизацию (maximization step):

$$\lambda_{t+1} = \arg \max_{\lambda} E(\lambda|\lambda_t)$$

- 4 Перейти к второму шагу до сходимости

Алгоритм Гиббса

В Метрополисе есть проблема: многомерное распределение $p(F|X) = p(\lambda|X)$. Можно попробовать рассматривать его по частям.

- 1 Начнем с какого-то λ
- 2 Сгенерируем λ_{t+1} по правилам

$$p(\lambda_{t+1,i}|X, \lambda_{t+1,1}, \dots, \lambda_{t+1,i-1}, \lambda_{t,i+1}, \dots, \lambda_{t,n})$$

- 3 Будем бегать по $i = 1 \dots n$, пока не сойдется
- 4 Полученная λ — следующая точка сэмплирования
- 5 Пока хочется идем в п.2

Пример с дартс IV

На языке товарищей Геман:

- 1 Прикинуть начальные m_i, Σ_i, π
- 2 Для каждого сэмпла сгенерировать A_j из текущего π
- 3 В условиях найденных A_j максимизировать LL
- 4 Перейти к второму шагу до сходимости того момента пока распределение параметров не перестанет меняться

Как можно построить $p(F|X)$ для RMSE

$$p(\lambda|X) = \frac{e^{-c\|F(\lambda|X)-Y\|_2}}{Z}$$

$$Z = \int_{\lambda} e^{-c\|F(\lambda|X)-Y\|_2} d\lambda$$

Если максимизируем, то надеемся задрать Y так, чтобы Z был определён.

Заключение

Сэмплирование очень полезная вещь:

- собирать данные;
- брать интегралы;
- обучаться.

Посмотрите на `bayesian inference` и библиотеку Гиббса для него: `BUGS`.

Задание на дом

- Датасет, как обычно, в svn
- Суть - учимся предсказывать размер аудитории сервиса
- $N(t) = \left(\frac{1}{1+e^{at+b}}\right)N_0$
- Ищем a , b , N_0
- Можно прямо hill climbing
- Можно подумать и воспользоваться алгоритмом Гиббса