Уменьшение размерности

Сергей Николенко

Академический Университет, 2012

Outline

Задача PCA

- Рассмотрим данные, лежащие в пространстве очень большой размерности.
- Как с ними работать? Часто оказывается, что «на самом деле» данные имеют меньшую размерность, чем кажется.
- Метод главных компонент – попытка найти оптимальное *линейное* сокращение размерности:
  1. найти проекцию (на пространство заданной размерности), в которой максимизируется дисперсия;
  2. найти проекцию минимальной цены (т.е. минимального суммарного расстояния до проекций всех точек).

Задача PCA

- Рассмотрим $\{\mathbf{x}_n\}_1^N$, $\mathbf{x}_n$ имеет размерность D. Хотим спроецировать в размерность $M < D$.
- Начнём с $M = 1$: надо найти вектор $\mathbf{u}_1$, $\mathbf{u}_1^\top \mathbf{u}_1 = 1$, для которого максимизируется дисперсия в проекции

$$\frac{1}{N}\sum_{n=1}^{N}\left(\mathbf{u}_1^\top \mathbf{x}_n - \mathbf{u}_1^\top \bar{\mathbf{x}}\right)^2 = \mathbf{u}_1^\top \mathbf{S}\mathbf{u}_1, \text{ где}$$

$$\bar{\mathbf{x}} = \frac{1}{N}\sum_{n=1}^{N}\mathbf{x}_n, \quad \mathbf{S} = \frac{1}{N}\sum_{n=1}^{N}\left(\mathbf{x}_n - \bar{\mathbf{x}}\right)\left(\mathbf{x}_n - \bar{\mathbf{x}}\right)^\top.$$

- Т.е. задача – максимизировать $\mathbf{u}_1^\top \mathbf{S}\mathbf{u}_1$ с ограничением $\mathbf{u}_1^\top \mathbf{u}_1 = 1$.

Задача PCA

- Задача – максимизировать $\mathbf{u}_1^\top \mathbf{S}\mathbf{u}_1$ с ограничением $\mathbf{u}_1^\top \mathbf{u}_1 = 1$.
- Добавляем множитель Лагранжа, максимизируем

$$\mathbf{u}_1^\top \mathbf{S}\mathbf{u}_1 + \lambda_1(1 - \mathbf{u}_1^\top \mathbf{u}_1).$$

- Получается, что максимум достигается, когда

$$\mathbf{S}\mathbf{u}_1 = \lambda_1 \mathbf{u}_1, \text{ т.е. } \lambda_1 = \mathbf{u}_1^\top \mathbf{S}\mathbf{u}_1.$$

- В итоге мы получили, что $\mathbf{u}_1$ – собственный вектор $\mathbf{S}$ с максимальным собственным числом λ_1.
- И дальше то же самое: $\mathbf{u}_2$ – второй собственный вектор и т.д.

Задача PCA

- Теперь с другой стороны – будем минимизировать ошибку.
- Введём ортонормированный базис $\{\mathbf{u}_j\}$, $\mathbf{u}_i^\top \mathbf{u}_j = \delta_{ij}$.
- Векторы раскладываются тогда как

$$\mathbf{x}_n = \sum_{i=1}^{D} \left(\mathbf{x}_n^\top \mathbf{u}_i\right) \mathbf{u}_i,$$

а хотим мы аппроксимировать как

$$\tilde{\mathbf{x}}_n = \sum_{i=1}^{M} z_{ni}\mathbf{u}_i + \sum_{i=M+1}^{D} b_i \mathbf{u}_i,$$

где b_i для всех одинаковые (поворот и смещение подпространства размерности M).

Задача PCA

- Аппроксимируем

$$\tilde{\mathbf{x}}_n = \sum_{i=1}^{M} z_{ni}\mathbf{u}_i + \sum_{i=M+1}^{D} b_i\mathbf{u}_i,$$

оптимизируем в итоге

$$J = \frac{1}{N}\sum_{n=1}^{N} \|\mathbf{x}_n - \tilde{\mathbf{x}}_n\|^2.$$

- Взяв производные по z_{nj}, получим

$$z_{nj} = \mathbf{x}_n^\top \mathbf{u}_j.$$

- По b_j:

$$b_j = \bar{\mathbf{x}}^\top \mathbf{u}_j.$$

Задача PCA

- Взяв производные по z_{nj}, получим $z_{nj} = \mathbf{x}_n^\top \mathbf{u}_j$, а по b_j – $b_j = \bar{\mathbf{x}}^\top \mathbf{u}_j$.
- Итого получаем

$$J = \frac{1}{N} \sum_{n=1}^{N} \sum_{i=M+1}^{D} \left(\mathbf{x}_n^\top \mathbf{u}_i - \bar{\mathbf{x}}_n^\top \mathbf{u}_i \right)^2 = \sum_{i=M+1}^{D} \mathbf{u}_i^\top \mathbf{S} \mathbf{u}_i.$$

- И опять получается, что $\mathbf{u}_i$ должны быть собственными векторами $\mathbf{S}$, то же самое.

Применения PCA

- Зачем всё это нужно?
  1. Сжать данные, уменьшить размерность.
  2. Сделать preprocessing – после PCA могут быть лучше видны характерные особенности; например, можно декоррелировать данные: если записать собственные векторы как $\mathbf{SU} = \mathbf{UL}$, где $\mathbf{L}$ – диагональная матрица из λ_i, $\mathbf{U}$ – ортогональная матрица из $\mathbf{u}_i$, и сделать преобразование

$$\mathbf{y}_n = \mathbf{L}^{-1/2}\mathbf{U}^\top(\mathbf{x}_n - \bar{\mathbf{x}}),$$

получим данные с нулевым средним и ковариациями

$$\frac{1}{N}\sum \mathbf{y}_n\mathbf{y}_n^\top = \frac{1}{N}\sum_{n=1}^{N}\mathbf{L}^{-1/2}\mathbf{U}^\top(\mathbf{x}_n-\bar{\mathbf{x}})(\mathbf{x}_n-\bar{\mathbf{x}})^\top\mathbf{U}\mathbf{L}^{-1/2} =$$
$$= \mathbf{L}^{-1/2}\mathbf{U}^\top\mathbf{S}\mathbf{U}\mathbf{L}^{-1/2} = \mathbf{L}^{-1/2}\mathbf{L}\mathbf{L}^{-1/2} = \mathbf{I}.$$

- В чём отличие от дискриминанта Фишера?

Вероятностный PCA

- А как это всё по-нашему, по-байесовски?
- Введём скрытую переменную $\mathbf{z}$, соответствующую подпространству главных компонент; базовая модель простая:

$$\mathbf{x} = \mathbf{W}\mathbf{z} + \mu + \epsilon.$$

- Априорное распределение $p(\mathbf{z}) = \mathcal{N}(\mathbf{z} \mid \mathbf{0}, \mathbf{I})$ (можно и вокруг нуля, не важно).
- Будем считать, что наблюдаемая – опять с нормальным шумом:

$$p(\mathbf{x} \mid \mathbf{z}) = \mathcal{N}(\mathbf{x} \mid \mathbf{W}\mathbf{z} + \boldsymbol{\mu}, \sigma^2\mathbf{I}).$$

- Тут всё раскладывается по компонентам $\mathbf{x}$, т.е. это наивный байес по сути.

Вероятностный PCA

- Как теперь найти $\mathbf{W}$, $\boldsymbol{\mu}$, σ^2? Правдоподобие (упражнение):

$$p(\mathbf{x}) = \int p(\mathbf{x} \mid \mathbf{z}) p(\mathbf{z}) d\mathbf{z} = \mathcal{N}(\mathbf{x} \mid \boldsymbol{\mu}, \mathbf{C}), \text{ где } \mathbf{C} = \mathbf{W}\mathbf{W}^\top + \sigma^2 \mathbf{I}.$$

- Можно обращать $\mathbf{C}$ быстрее (не как $D \times D$ матрицу) как

$$\mathbf{C}^{-1} = \frac{1}{\sigma}\mathbf{I} - \frac{1}{\sigma^2}\mathbf{W}\mathbf{M}^{-1}\mathbf{W}^\top, \text{ где } \mathbf{M} = \mathbf{W}^\top\mathbf{W} + \sigma^2\mathbf{I}.$$

- Апостериорное распределение получается

$$p(\mathbf{z} \mid \mathbf{x}) = \mathcal{N}\left(\mathbf{z} \mid \mathbf{M}^{-1}\mathbf{W}^\top(\mathbf{x} - \boldsymbol{\mu}), \frac{1}{\sigma}^2 \mathbf{M}\right).$$

Вероятностный PCA

- Чтобы максимизировать правдоподобие, считаем:

$$\ln p(\mathbf{X} \mid \boldsymbol{\mu}, \mathbf{W}, \sigma^2) =$$

$$= -\frac{ND}{2}\ln(2\pi) - \frac{N}{2}\ln|\mathbf{C}| - \frac{1}{2}\sum_{n=1}^{N}(\mathbf{x}_n - \boldsymbol{\mu})^\top \mathbf{C}^{-1}(\mathbf{x}_n - \boldsymbol{\mu}) =$$

$$= -\frac{N}{2}\left(D\ln(2\pi) + \ln|\mathbf{C}| + \mathrm{Tr}(\mathbf{C}^{-1}\mathbf{S})\right)$$

(мы подставили $\mu = \bar{x}$, что легко получить из исходной формулы).

Вероятностный PCA

- Более сложный результат (без док-ва): нули производной по $\mathbf{W}$ будут в

$$\mathbf{W}_{ML} = \mathbf{U}_M(\mathbf{L}_m - \sigma^2\mathbf{I})^{1/2}\boldsymbol{R},$$

где $\mathbf{U}_M$ – $D \times M$ матрица из подмножества собственных векторов, $\mathbf{L}_M$ – диагональная из λ_i, $\boldsymbol{R}$ – произвольная ортогональная.

- А максимум достигается, если взять M наибольших собственных векторов, т.е. всё то же самое.

Вероятностный PCA

- В пространствах большой размерности может быть сложно работать непосредственно с матрицами (обращать надо).
- Поэтому для PCA часто используют EM-алгоритм: на E-шаге считают параметры нормального распределения $\mathbf{z}_n$, а на M-шаге, фиксируя $\mathbf{z}_n$, максимизируют по $\mathbf{W}$ и σ^2.
- Упражнение: получите формулы для EM-алгоритма.
- Стандартный PCA получается из вероятностного в пределе при $\sigma^2 \to 0$; EM-алгоритм всё равно будет работать даже в пределе.
- Есть и байесовская версия – вводим prior на $\mathbf{W}$ (независимые гауссианы), пересчитываем гиперпараметры $\boldsymbol{\alpha}$, максимизируя (приближённо)

$$p(\mathbf{X} \mid \boldsymbol{\alpha}, \boldsymbol{\mu}, \sigma^2) = \int p(\mathbf{X} \mid \mathbf{W}, \boldsymbol{\mu}, \sigma^2) p(\mathbf{W} \mid \boldsymbol{\alpha}) d\mathbf{W}.$$

Outline

Kernel PCA

- Но что если многообразие, на котором лежат данные, нелинейное?
- Для PCA работает трюк с ядрами:

$$\mathbf{S} = \frac{1}{N} \sum_{n=1}^{N} \mathbf{x}_n \mathbf{x}_n^\top,$$

  и здесь участвуют только скалярные произведения.
- Значит, можно делать стандартный PCA в пространстве большой размерности $\phi(\mathbf{x})$.

Анализ независимых компонент

- ICA – предположим, что распределение раскладывается: $p(\mathbf{z}) = \prod_{j=1}^{M} p(z_j)$.
- Мы тогда рассматриваем наблюдения как линейные комбинации скрытых (пример: blind source separation).
- Шум можно к наблюдаемым не добавлять, т.к. скрытых переменных столько же, сколько видимых, и шум можно спрятать в $p(z_j)$.

Автоассоциативные нейронные сети

- Интересная идея: автоассоциативная нейронная сеть.
- Выход считается равным входу, а на (маленьком) среднем уровне обучается сжатое представление данных.
- Autoencoders – поговорим об этом ещё в контексте deep learning.

Обзор

1. Обучение с подкреплением:
   1. бандиты, exploration vs. exploitation;
   2. марковские процессы принятия решений, TD-обучение;
   3. индексы Гиттинса;
   4. оценки на regret, стратегия UCB1, логарифмическая оценка;
   5. Dynamic Gamma-Poisson.

Обзор

2. Графические вероятностные модели:
   1. направленные и ненаправленные, фактор-графы;
   2. задача маргинализации, маргинализация в линейной цепи;
   3. алгоритм передачи сообщений;
   4. проблема с циклами, loopy BP.

Обзор

3. Приближённый вывод:
   1. вариационные методы: основы;
   2. вариационный байесовский вывод для гауссиана;
   3. сэмплирование: выборка по значимости, выборка с отклонением;
   4. методы MCMC, сэмплирование по Гиббсу;
   5. Expectation Propagation.

Обзор

4. Примеры и применения:
   1. topic modeling, LDA (вариационный вывод);
   2. рейтинг-системы, TrueSkill (Expectation Propagation);
   3. рекомендательные системы.
5. Плюс ещё глубокое обучение...

Thank you!

Спасибо за внимание!