Методы машинного обучения. Обучение без учителя: векторизация данных

Воронцов Константин Вячеславович
www.MachineLearning.ru/wiki?title=User:Vokov
вопросы к лектору: k.v.vorontsov@yandex.ru

материалы курса:
github.com/MSU-ML-COURSE/ML-COURSE-23-24
орг.вопросы по курсу: ml.cmc@mail.ru

ВМК МГУ • 20 февраля 2024

Методы обучения без учителя (unsupervised learning)

Выявление структуры данных на основе сходства:

- кластеризация (clustering) и квантизация (quantization)
- оценивание плотности распределения (density estimation)
- одноклассовая классификация (anomaly detection)

Преобразование признакового пространства:

- метод главных компонент (principal components analysis)
- автокодировщики (autoencoders)
- многомерное шкалирование (multidimensional scaling)
- матричные разложения (matrix factorization)

Поиск взаимосвязей в данных или синтез учителя:

- частичное обучение (semi-supervised learning)
- поиск ассоциативных правил (association rule learning)
- самостоятельное обучение (self-supervised learning)

Содержание

Постановка задачи кластеризации и квантизации

Дано:
$X^\ell = \{x_i\}_{i=1}^{\ell}$ — обучающая выборка объектов, $x_i \in \mathbb{R}^n$
$\rho^2(x, w) = \|x - w\|^2$ — евклидова метрика в $\mathbb{R}^n$

Найти:
центры кластеров $w_y \in \mathbb{R}^n$, $y \in Y$; модель кластеризации «правило жёсткой конкуренции» (WTA, Winner Takes All):

$$a(x) = \arg\min_{y \in Y} \rho(x, w_y)$$

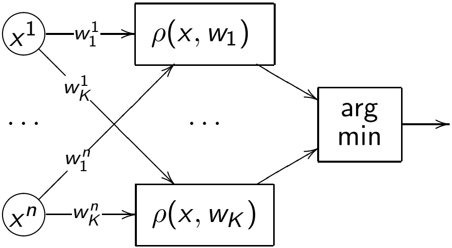
Критерий: среднее внутрикластерное расстояние

$$Q(w; X^\ell) = \sum_{i=1}^{\ell} \rho^2(x_i, w_{a(x_i)}) \to \min_{w_y :\, y \in Y}$$

Квантизация данных — замена x_i на ближайший центр $w_{a(x_i)}$

Сеть Кохонена (сеть с конкурентным обучением)

Структура модели — (якобы) двухслойная нейронная сеть:



$$a(x) = \arg\min_{y \in Y} \rho(x, w_y)$$

Градиентный шаг в методе SG: для выбранного $x_i \in X^\ell$

$$w_y := w_y + \eta(x_i - w_y)\bigl[a(x_i) = y\bigr]$$

Если x_i относится к кластеру y, то w_y сдвигается в сторону x_i

T.Kohonen. Self-organized formation of topologically correct feature maps. 1982.

Алгоритм SG (Stochastic Gradient)

Вход: выборка X^ℓ; темп обучения η; параметр λ;
Выход: центры кластеров $w_y \in \mathbb{R}^n$, $y \in Y$;

инициализировать центры w_y, $y \in Y$;
инициализировать текущую оценку функционала:
$Q := \sum\limits_{i=1}^{\ell} \rho^2(x_i, w_{a(x_i)})$;
повторять
- выбрать объект x_i из X^ℓ (например, случайно);
- найти ближайший центр: $y := \arg\min\limits_{y \in Y} \rho(x_i, w_y)$;
- градиентный шаг: $w_y := w_y + \eta(x_i - w_y)$;
- оценить значение функционала:
  $Q := (1 - \lambda)Q + \lambda\rho^2(x_i, w_y)$;

пока значение Q и/или веса w не стабилизируются;

Жёсткая и мягкая конкуренция

Правило жёсткой конкуренции WTA (winner takes all):

$$w_y := w_y + \eta(x_i - w_y)\big[a(x_i) = y\big], \quad y \in Y$$

Недостатки правила WTA:

- медленная скорость сходимости
- некоторые w_y могут никогда не выбираться

Правило мягкой конкуренции WTM (winner takes most):

$$w_y := w_y + \eta(x_i - w_y)\, K\big(\rho^2(x_i, w_y)\big), \quad y \in Y$$

где ядро $K(\rho^2)$ — неотрицательная невозрастающая функция

Теперь центры *всех* кластеров смещаются в сторону x_i,
но чем дальше от x_i, тем меньше величина смещения

Обоснование правила мягкой конкуренции

Жёсткая кластеризация WTA:

$$a(x) = \arg\min_{y \in Y} \rho(x, w_y)$$

Мягкая кластеризация WTM:

объект x «размазывается» по всем кластерам,

$$a_y(x) = K\bigl(\rho^2(x, w_y)\bigr) \quad y \in Y$$

Минимизация среднего внутрикластерного расстояния:

$$Q(w; X^\ell) = \frac{1}{2} \sum_{i=1}^{\ell} \sum_{y \in Y} a_y(x_i)\, \rho^2(x_i, w_y) \to \min_w;$$

$$\frac{\partial Q(w)}{\partial w_y} = \sum_{i=1}^{\ell} (w_y - x_i)\, a_y(x_i)\, \underbrace{\left(1 + \frac{\rho^2 K'(\rho^2)}{K(\rho^2)}\right)}$$

скалярный множитель-поправка к градиентному шагу η

Задача классификации LVQ (Learning Vector Quantization)

Дано:

$X^\ell = \{x_i, y_i\}_{i=1}^{\ell}$ — объекты $x_i \in \mathbb{R}^n$ с метками классов $y_i \in Y$

C — множество кластеров, $y(c) \in Y$ — класс кластера $c \in C$

Найти:

центры кластеров $w_c \in \mathbb{R}^n$, $c \in C$ в модели кластеризации WTA

$$c(x) = \arg\min_{c \in C} \rho(x, w_c)$$

и модель классификации $a(x) = y(c(x))$

Критерий:

min внутрикластерных расстояний в своём классе,

max внутрикластерных расстояний с чужими классами:

$$Q = \sum_{i=1}^{\ell} \rho^2(x_i, w_{c(x_i)}) \Big([y(c(x_i)) = y_i] - [y(c(x_i)) \neq y_i] \Big) \to \min_{w_c\colon c \in C}$$

Алгоритм SG (Stochastic Gradient)

Вход: выборка X^ℓ; темп обучения η; параметр λ;
Выход: центры кластеров $w_c \in \mathbb{R}^n$, $c \in C$;

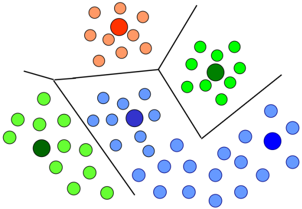
инициализировать центры w_c и задать $y(c)$, $c \in C$;
инициализировать текущую оценку функционала Q;
повторять

- выбрать объект x_i из X^ℓ (например, случайно);
- вычислить кластеризацию: $c := \arg\min\limits_{c \in C} \rho(x_i, w_c)$;
- вычислить классификацию: $a := y(c)$;
- $w_c := w_c + \eta(x_i - w_c)$, если $a = y_i$;
- $w_c := w_c - \eta(x_i - w_c)$, если $a \neq y_i$;
- оценить значение функционала:
  $$Q := (1 - \lambda)Q + \lambda\rho^2(x_i, w_y)\Bigl([a{=}y_i] - [a{\neq}y_i]\Bigr);$$

пока значение Q и/или веса w не стабилизируются;

Обучаемое векторное квантование

- кластеризация, реализуемая (якобы) нейронной сетью
- классификация путём разбиения каждого класса на заданное число кластеров
- позволяет моделировать классы сложной формы
- похоже на отбор эталонов (prototype selection)
- похоже на байесовский классификатор с GMM-классами



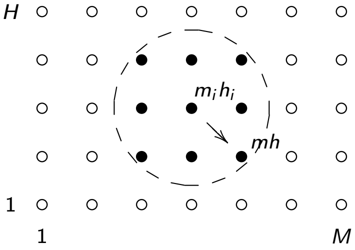
Teuvo Kohonen. Improved versions of learning vector quantization. 1990

Карта Кохонена (Self Organizing Map, SOM)

$Y = \{1,\dots,M\} \times \{1,\dots,H\}$ — прямоугольная сетка кластеров
Каждому узлу (m,h) приписан нейрон Кохонена $w_{mh} \in \mathbb{R}^n$
Наряду с метрикой $\rho(x_i, x)$ на X вводится метрика на сетке Y:

$$r\bigl((m_i,h_i),(m,h)\bigr) = \sqrt{(m-m_i)^2 + (h-h_i)^2}$$

Окрестность(m_i, h_i):



Teuvo Kohonen. Self-Organizing Maps. 2001.

Обучение карты Кохонена

Вход: X^ℓ — обучающая выборка; η — темп обучения;

Выход: $w_{mh} \in \mathbb{R}^n$ — векторы весов, $m = 1..M$, $h = 1..H$;

$w_{mh} := \text{random}\left(-\frac{1}{2MH}, \frac{1}{2MH}\right)$ — инициализация весов;

повторять

- выбрать объект x_i из X^ℓ случайным образом;
- WTA: вычислить координаты кластера:
- $(m_i, h_i) := a(x_i) \equiv \arg\min\limits_{(m,h)\in Y} \rho(x_i, w_{mh})$;
- **для всех** $(m, h) \in \text{Окрестность}(m_i, h_i)$
  - WTM: сделать шаг градиентного спуска:
  - $w_{mh} := w_{mh} + \eta(x_i - w_{mh})\, K\big(r((m_i, h_i), (m, h))\big)$;

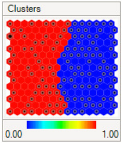
пока кластеризация не стабилизируется;

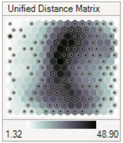
Интерпретация карт Кохонена

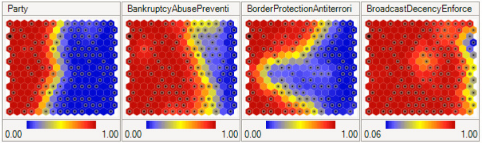
Два типа графиков — цветных карт $M \times H$:

- Цвет узла (m, h) — локальная плотность в точке (m, h) — среднее расстояние до k ближайших точек выборки
- По одной карте на каждый признак:
  цвет узла (m, h) — значение j-й компоненты вектора $w_{m,h}$

Пример: задача UCI house-votes (US Congress voting patterns)
Объекты — конгрессмены
Признаки — результаты голосования по различным вопросам
Есть целевой признак «партия» $\in$ {демократ, республиканец}

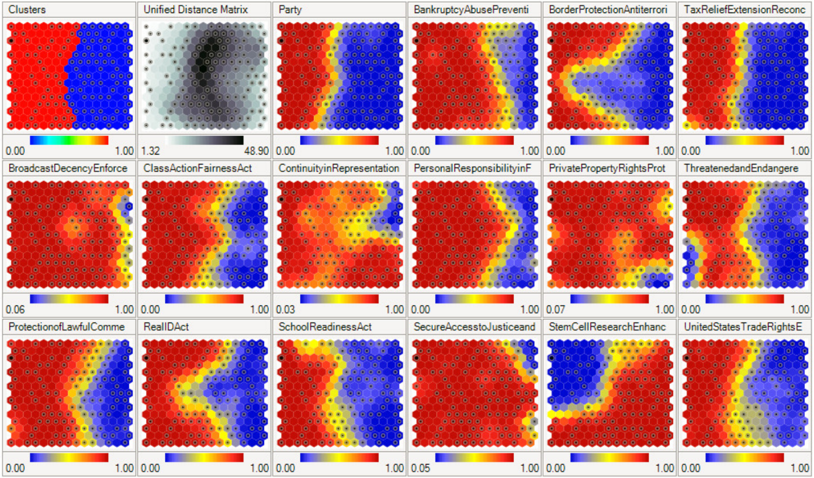






Интерпретация карт Кохонена (продолжение примера)

Пример: задача UCI house-votes (US Congress voting patterns)



Достоинства и недостатки карт Кохонена

Достоинства:

- Возможность визуального анализа многомерных данных
- Квантование выборки по кластерам,
  с автоматическим определением числа непустых кластеров

Недостатки:

- **Субъективность.** Карта отражает не только кластерную структуру данных, но также зависит от...
  — свойств сглаживающего ядра;
  — (случайной) инициализации;
  — (случайного) выбора x_i в ходе итераций.
- **Искажения.** Близкие объекты исходного пространства могут переходить в далёкие точки на карте, и наоборот.

Рекомендуется только для разведочного анализа данных.

Построение автокодировщика — задача обучения без учителя

$X^\ell = \{x_1, \dots, x_\ell\}$ — обучающая выборка
$f\colon X \to Z$ — кодировщик (encoder), кодовый вектор $z = f(x, \alpha)$
$g\colon Z \to X$ — декодировщик (decoder), реконструкция $\hat{x} = g(z, \beta)$

Суперпозиция $\hat{x} = g(f(x))$ должна восстанавливать исходные x_i:

$$\mathscr{L}_{\text{AE}}(\alpha, \beta) = \sum_{i=1}^{\ell} \mathscr{L}\bigl(g(f(x_i, \alpha), \beta), x_i\bigr) \to \min_{\alpha, \beta}$$

Квадратичная функция потерь: $\mathscr{L}(\hat{x}, x) = \|\hat{x} - x\|^2$

Пример 1. Линейный автокодировщик: $x \in \mathbb{R}^n$, $z \in \mathbb{R}^m$

$$f(x, A) = \underset{m \times n}{A}\, x, \qquad g(z, B) = \underset{n \times m}{B}\, z$$

Пример 2. Двухслойная сеть с функциями активации σ_f, σ_g:

$$f(x, A) = \sigma_f(Ax + a), \qquad g(z, B) = \sigma_g(Bz + b)$$

Обучение и использование автокодировщиков

Метод обучения:

- Стохастический градиент (SG) по параметрам (α, β)
- Одновременно обучаются две модели — $f(x, \alpha)$ и $g(z, \beta)$

Способы использования:

- Сжатие данных с минимальной потерей информации
- Векторизация данных:
  — понижение размерности (dimensionality reduction)
  — синтез более удачных признаков (feature generation)
- Обучение с учителем в новом пространстве признаков
- Генерация синтетических объектов, похожих на реальные

Rumelhart, Hinton, Williams. Learning internal representations by error propagation. 1986.

David Charte et al. A practical tutorial on autoencoders for nonlinear feature fusion: taxonomy, models, software and guidelines. 2018.

Линейный автокодировщик и метод главных компонент

Линейный автокодировщик: $f(x, A) = Ax$, $g(z, B) = Bz$,

$$\mathscr{L}_{\text{AE}}(A, B) = \sum_{i=1}^{\ell} \|BAx_i - x_i\|^2 \to \min_{A,B}$$

Метод главных компонент: $f(x, U) = U^{\mathsf{T}}x$, $g(z, U) = Uz$,
в матричных обозначениях $F = (x_1 \dots x_\ell)^{\mathsf{T}}$, $U^{\mathsf{T}}U = I_m$, $G = FU$,

$$\|F - GU^{\mathsf{T}}\|^2 = \sum_{i=1}^{\ell} \|UU^{\mathsf{T}}x_i - x_i\|^2 \to \min_{U}$$

Автокодировщик обобщает метод главных компонент:

- не обязательно $B = A^{\mathsf{T}}$ (хотя часто именно так и делают)
- произвольные A, B вместо ортогональных
- нелинейные модели $f(x, \alpha)$, $g(z, \beta)$ вместо Ax, Bz
- произвольная функция потерь $\mathscr{L}$ вместо квадратичной
- SG оптимизация вместо сингулярного разложения SVD

Разреживающие автокодировщики (Sparse AE)

Применение L_1 или L_2-регуляризации к векторам весов α, β:

$$\mathscr{L}_{\text{AE}}(\alpha, \beta) + \lambda\|\alpha\| + \lambda\|\beta\| \to \min_{\alpha,\beta}$$

Применение L_1-регуляризации к кодовым векторам z_i:

$$\mathscr{L}_{\text{AE}}(\alpha, \beta) + \lambda \sum_{i=1}^{\ell} \sum_{j=1}^{m} |f_j(x_i, \alpha)| \to \min_{\alpha,\beta}$$

Энтропийная регуляризация для случая $f_j \in [0, 1]$:

$$\mathscr{L}_{\text{AE}}(\alpha, \beta) + \lambda \sum_{j=1}^{m} \text{KL}(\varepsilon\|\bar{f}_j\,) \to \min_{\alpha,\beta},$$

где $\bar{f}_j = \frac{1}{\ell} \sum\limits_{i=1}^{\ell} f_j(x_i, \alpha)$; $\quad \varepsilon \in (0, 1)$ — близкий к нулю параметр,

$\text{KL}(\varepsilon\|\rho) = \varepsilon \log \frac{\varepsilon}{\rho} + (1-\varepsilon) \log \frac{1-\varepsilon}{1-\rho}$ — KL-дивергенция.

D.Arpit et al. Why regularized auto-encoders learn sparse representation? 2015.

Шумоподавляющий автокодировщик (Denoising AE)

Устойчивость кодовых векторов z_i относительно шума в x_i:

$$\mathscr{L}_{\text{DAE}}(\alpha, \beta) = \sum_{i=1}^{\ell} \mathsf{E}_{\tilde{x} \sim q(\tilde{x}|x_i)} \mathscr{L}\bigl(g(f(\tilde{x}, \alpha), \beta), x_i\bigr) \to \min_{\alpha, \beta}$$

Вместо вычисления $\mathsf{E}_{\tilde{x}}$ в методе SG объекты x_i сэмплируются и зашумляются по одному: $\tilde{x} \sim q(\tilde{x}|x_i)$.

Варианты зашумления $q(\tilde{x}|x_i)$:

- $\tilde{x} \sim \mathcal{N}(x_i, \sigma^2 I)$ — добавление гауссовского шума
- обнуление компонент вектора x_i с вероятностью p_0
- такие искажения $x_i \to \tilde{x}$, относительно которых реконструкция $\hat{x}_i$ должна быть устойчивой

P. Vincent, H. Larochelle, Y. Bengio, P.-A. Manzagol. Extracting and composing robust features with denoising autoencoders. ICML-2008.

Реляционный автокодировщик (Relational AE)

Наряду с потерями реконструкции объектов минимизируем потери реконструкции отношений между объектами:

$$\mathscr{L}_{\mathsf{AE}}(\alpha, \beta) + \lambda \sum_{i<j} \mathscr{L}\bigl(\sigma(\hat{x}_i^{\mathsf{T}} \hat{x}_j), \sigma(x_i^{\mathsf{T}} x_j)\bigr) \to \min_{\alpha, \beta}$$

где $\hat{x}_i = g(f(x_i, \alpha), \beta)$ — реконструкция объекта x_i,
$x_i^{\mathsf{T}} x_j$ — скалярное произведение (близость) пары объектов,
$\sigma(s) = (s - s_0)_+$ — пороговая функция с параметром s_0
(если векторы не близки, то неважно, насколько),
$\mathscr{L}(\hat{s}, s)$ — функция потерь, например, $(\hat{s} - s)^2$.

Эксперимент: улучшается качество классификации изображений с помощью кодовых векторов на задачах MNIST, CIFAR-10

Qinxue Meng et al. Relational autoencoder for feature extraction. 2018.

Вариационный автокодировщик (Variational AE)

Задача: построить декодировщик, способный генерировать «фейковые» объекты x, похожие на объекты выборки $x_1, \dots, x_\ell$

$q_\alpha(z|x)$ — вероятностный кодировщик с параметром α
$p_\beta(\hat{x}|z)$ — вероятностный декодировщик с параметром β

Максимизация нижней оценки log-правдоподобия:

$$\mathscr{L}_{\text{VAE}}(\alpha, \beta) = \sum_{i=1}^{\ell} \log p(x_i) = \sum_{i=1}^{\ell} \log \int q_\alpha(z|x_i) \frac{p_\beta(x_i|z)p(z)}{q_\alpha(z|x_i)} dz \geqslant$$

$$\geqslant \sum_{i=1}^{\ell} \int q_\alpha(z|x_i) \log \frac{p_\beta(x_i|z)p(z)}{q_\alpha(z|x_i)} dz =$$

$$= \sum_{i=1}^{\ell} \int q_\alpha(z|x_i) \log p_\beta(x_i|z) dz - \text{KL}\bigl(q_\alpha(z|x_i) \,\|\, p(z)\bigr) \to \max_{\alpha,\beta}$$

D.P.Kingma, M.Welling. Auto-encoding Variational Bayes. 2013.
C.Doersch. Tutorial on variational autoencoders. 2016.

Вариационный автокодировщик (Variational AE)

Оптимизационная задача для вариационного автокодировщика:

$$\sum_{i=1}^{\ell} \underbrace{\mathsf{E}_{z\sim q_\alpha(z|x_i)} \log p_\beta(x_i|z)}_{\substack{\text{качество реконструкции}\\ \approx \log p_\beta(x_i|z),\ z\sim q_\alpha(z|x_i)}} - \underbrace{\mathrm{KL}\bigl(q_\alpha(z|x_i) \,\|\, p(z)\bigr)}_{\text{регуляризатор по } \alpha} \to \max_{\alpha,\beta}$$

где $p(z)$ — априорное распределение, обычно $\mathcal{N}(0, \sigma^2 I)$

Репараметризация $q_\alpha(z|x_i)$: $\quad z = f(x_i, \alpha, \varepsilon)$, $\varepsilon \sim \mathcal{N}(0, I)$

Метод стохастического градиента:

- сэмплировать $x_i \sim X^\ell$, $\;\varepsilon \sim \mathcal{N}(0, I)$, $\;z = f(x_i, \alpha, \varepsilon)$
- градиентный шаг:
  $\alpha := \alpha + h\nabla_\alpha \bigl[\log p_\beta\bigl(x_i|f(x_i, \alpha, \varepsilon)\bigr) - \mathrm{KL}\bigl(q_\alpha(z|x_i)\|p(z)\bigr)\bigr]$;
  $\beta := \beta + h\nabla_\beta \bigl[\log p_\beta(x_i|z)\bigr]$;

Генерация похожих объектов: $\;x \sim p_\beta\bigl(x|f(x_i, \alpha, \varepsilon)\bigr)$, $\;\varepsilon \sim \mathcal{N}(0, I)$

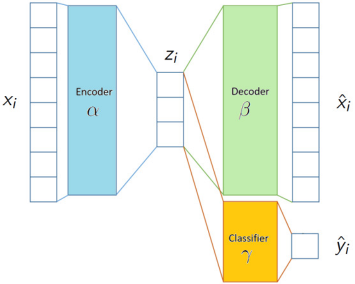
Автокодировщик с частичным обучением

Данные: размеченные $(x_i, y_i)_{i=1}^k$, неразмеченные $(x_i)_{i=k+1}^\ell$

Совместное обучение кодировщика, декодировщика и предсказательной модели (классификации, регрессии или др.):

$$\sum_{i=1}^{\ell} \mathscr{L}\bigl(g(f(x_i,\alpha),\beta),x_i\bigr) + \lambda \sum_{i=1}^{k} \tilde{\mathscr{L}}\bigl(\hat{y}(f(x_i,\alpha),\gamma),y_i\bigr) \to \min_{\alpha,\beta,\gamma}$$

$z_i = f(x_i, \alpha)$ — кодировщик
$\hat{x}_i = g(z_i, \beta)$ — декодировщик
$\hat{y}_i = \hat{y}(z_i, \gamma)$ — предиктор



Функции потерь:
$\mathscr{L}(\hat{x}_i, x_i)$ — реконструкция
$\tilde{\mathscr{L}}(\hat{y}_i, y_i)$ — предсказание

Dor Bank, Noam Koenigstein, Raja Giryes. Autoencoders. 2020

Резюме

Обучение без учителя — выявление структур в данных

- Сети Кохонена — никакие не сети, просто кластеризация
- LVQ — кластеризации для классификации
- Карты Кохонена — кластеризация для 2D-визуализации

Разновидности векторизации данных:

- *Квантизация* — сокращение объёма выборки, замена объектов ближайшими центрами кластеров
- *Автокодировщики* — синтез векторных представлений (эмбедингов) объектов, обычно с понижением размерности
- *Метод главных компонент* — частный случай линейного автокодировщика

Методы обучения — на основе SG