МО2

Горлевич Даниил

2021

Contents

Part I
Лекции

1 Ядровые методы

Данные: $x = (x_1, ...x_m)$
Базисные функции: $\phi(x_1, ...)$
Модель принимает вид: $a(x) = \sum_{j=1}^{m} w_j \phi_j(x)$
Для хорошего качества нужно много базисных функций → Ядровые методы позволяют не перебирать большое количество базисных функций

- Быстрое обучение

Ядровые методы

1. Двойственное представление для линейной регрессии

   $Q(w) = \frac{1}{2} \sum_{i=1}^{l} (\sum_{j=1}^{m} (w_j \phi_j(x_i) - y_i)^2 + \frac{\lambda}{2} ||w||_2^2 = \frac{1}{2} ||\Phi w - y||_2^2 + \frac{\lambda}{2} ||w||_2^2$

   $$\Phi = \begin{pmatrix} \phi_1(x_1) & ... & \phi_m(x_1) \\ ... & ... & ... \\ \phi_1(x_l) & ... & \phi_m(x_l) \end{pmatrix}$$

   $\nabla_w Q = \Phi^T(\Phi w - y) + \lambda w \to w = -\frac{1}{\lambda} \Phi^T(\Phi w - y) \to w = \Phi^T a$

   w является линейной комбинацией строк Φ → Решение можно искать из $w = \Phi^T a$

   $Q(a) = \frac{1}{2} ||\Phi\Phi^T a - y|| + \frac{\lambda}{2} a^T \Phi\Phi^T a \to min_a$

$ФФ^T$ - матрица Грама (попарных скалярных произведений объектов)

Можно записать Q(w) так, что он зависит только от скалярных произведений объектов

2. SVM
$$\begin{cases} \sum_{i=1}^{l} \lambda_i - \frac{1}{2} \sum_{i,j=1}^{l} \lambda_i \lambda_j y_i y_j < x_i, x_j > \to \max_\lambda \\ 0 \geq \lambda_i \leq C \\ \sum_{i=1}^{l} \lambda_i y_i = 0 \end{cases}$$
Такая формулировка задачи зависит от скалярных произведений объектов

3. Алгоритм
   (a) Добавляем новые признаки
   (b) $x, z \in X$
   (c) Делаем это так, что $< \phi(x), \phi(z) >$ выражается через $< x, z >$
   (d) Используем метод, который использует скалярные произведения объектов
   (e) В этом методе $< x, z > \to < \phi(x), \phi(z) >$ (*Kernel trick*)

4. **Ядро** - функция $K(x, z) =< \phi(x), \phi(z) >$, где $\phi : X \to H$
   (a) H - спрямляющее пространство
   (b) ϕ - спрямляющее отображение

5. **Теорема Мерсера**
   (a) $K(x, z)$ - ядро $\leftrightarrow \begin{cases} K(x, z) = K(z, x) \\ K \text{ неотрицательно определенная} \end{cases}$
   (b) НО $\to \forall l, \forall (x_1, ..., x_l) \in R^d \to (K(x_i, x_j))_{i,j=1}^{l}$ НО
   (c) *На практике теорема Мерсера слишком сложна для применения*

6. **Теорема 1**
   (a) Если
      i. $K_1(x, z), K_2(x, z)$ - ядра, $x, z \in X$
      ii. $f^{(x)}$ - вещественная функция на X
      iii. $\phi : X \to R^n$

iv. K_3 - ядро заданное на R^n

(b) То *следующие функции являюися ядрами:*

i. $K(x,z) = K_1(x,z) + K_2(x,z)$
ii. $K(x,z) = \alpha K_1(x,z)$
iii. $K(x,z) = K_1 K_2$
iv. $K(x,z) = f^{(x)} f^{(z)}$
v. $K(x,z) = K(\phi(x), \phi(z))$

7. **Теорема 2**

(a) Если:

i. $K_1(x,z), K_2(x,z), ...$ - последовательность ядер
ii. $\exists K(x,z) = \lim_{n\to\infty} K_n(x,z), \forall x,z$

(b) То:

i. K - ядро

8. **Полиномиальные ядра**

(a) p(v) - многочлен с неотриц. коэфф

(b) $K(x,z) = w_0 + w_1 < x,z > + w_2 < x,z >^2 + ...$

(c) Является ядром по теореме 1

(d) $K(x,z) = (< x,z > + R)^m = \sum_{i=0}^{m} C_m^i R^{m-i} < x,z >^i$

i. Если расписать все $< x,z >^i$, то получим все мономы степени i от исходных признаков
ii. Зачем R? $\rightarrow$ коэффициент при мономе $= \sqrt{C_m^i R^{m-i}}$
iii. Сравним веса при мономах 1 и (m - 1) $\sqrt{\frac{C_m^{m-1} R}{C_m^1 R^{m-1}}} = \sqrt{\frac{1}{R^{m-2}}}$
iv. R больше - мономы высоких степеней имеют низкий вклад
v. Конечномерное спрямляющее пространство, но можно сделать линейно разделимое пространство

9. **Гауссовы ядра**

(a) Позволяет перевести в бесконечномерное спрямляющее пространство

(b) $\boxed{K(x,z) = exp\left(-\frac{||x-z||^2}{2\sigma^2}\right)}$

i. $exp(< x,z >) = \sum_{k=0}^{\infty} \frac{<x,z>^k}{k}, \forall x,z = \lim_{n\to\infty} \sum_{k=0}^{\infty} \frac{<x,z>^k}{k}$
A. Разложение через ряд Тейлора

B. Ядро, как последовательность ядер

ii. $\frac{exp(<x,z>)}{2\sigma^2}$ - ядро, аналогично

iii. $exp\left(-\frac{||x-z||^2}{2\sigma^2}\right) = exp\left(-\frac{<x-z,x-z>}{2\sigma^2}\right) = exp\left(-\frac{<x,x>-<z,z>,+<x,z>}{2\sigma^2}\right) = \frac{exp(<x,z>/\sigma^2}{exp(||x||^2/\sigma^2)exp(||z||^2/\sigma^2}$)

iv. $exp(< x, z > /\sigma^2) = K(x,z) =< \phi(x), \phi(z) >$

v. $\tilde{\phi(x)} = \frac{\phi(x)}{||\phi(x)||} = \frac{\phi(x)}{\sqrt{K(x,x)}}$

vi. $< \tilde{\phi(x)}, \tilde{\phi(z)} >= \frac{<\phi(x),\phi(z)>}{\sqrt{K(x,x)K(z,z)}}$

(c) Какое спрямляющее пространство? - бесконечная сумма всех мономов

(d) *Утверждение:* $x_1, ..., x_l$ - различные векторы из $\mathbb{R}^d$
Тогда:
$G = (exp\left(-\frac{||x-z||^2}{2\sigma^2}\right))_{i,j=1}^{l}$ - невырожденная при $\sigma^2 > 0$

(e) $x_1, ..., x_l \in \mathbb{R}^d$ - их матрица Грамма невырождена $\to \phi(x_1, ..., x_l)$ ЛНЗ $\to$ бесконечное количество ЛНЗ векторов $\to$ бесконечномерное пространство

10. **Ядровой SVM**

(a) $\begin{cases} \frac{1}{2}||w||^2 + C\sum_{i=1}^{l}\xi_i \to min_{w,b,\xi} \\ y_i(< w, x_i > +b) \geq 1 - \xi_i \\ \xi_i \geq 0 \end{cases}$

$$L(w,b,\xi,\lambda,\mu) = \frac{1}{2}||w||^2 + C\sum_{i=1}^{d}\xi_i - \sum_{i=1}^{l}\lambda_i(y_i(< w, x_i > +b) - 1 + \xi_i) - \sum_{i=1}^{l}\mu_i\xi_i$$

В точке оптимума $\nabla_w L = 0$

$$\nabla_w L = w - \sum_{i=1}^{l}\lambda y_i x_i = 0 \to w = \sum_{i=1}^{l}\lambda_i y_i x_i$$

$$\nabla_b L = \sum_{i=1}^{l}\lambda_i y_i = 0$$

$$\nabla_{\xi_i} L = C - \lambda_i - \mu_i = 0 \to \lambda_i + \mu_i = C$$

Условие дополняющей нежесткости:

$\lambda_i(y_i(< w, x_i > +b)-1+\xi_i) = 0 \to \lambda_i = 0$ или $(y_i(< w, x_i > +b)-1+\xi_i) = 0$

$$\mu_i\xi_i = 0 \to \mu_i = 0 \text{ или } \xi_i = 0$$

Свойства лагранжиана:

$$\lambda \geq 0, \mu \geq 0$$

(b) Типы объектов

i. $\lambda_i = 0 \to \mu_i = C \to \xi_i = 0 \to x_i$ лежит с правильной стороны от разделяющей гиперплоскости и на достаточном расстоянии от нее. $w = \sum_{i=1}^{l} \lambda y_i x_i \to$ объект не влияет на веса. Называется **периферийный.**

ii. $0 < \lambda_i < 1 \to \mu \neq 0 \to \xi_0 = 0$. x_i не залезает на разделяющую полосу, но $y_i(< w, x_i > +b) = 1 \to x_i$ лежит прямо на границе. Дает вклад в w. x_i - **опорный граничный.**

iii. $\lambda_i = C \to \xi_i > 0$. x_i дает вклад в w. $\xi_i > 0 \to x_i$ нарушает границу - **Опорные нарушители**.

(c) Подставляем $w = \sum_{i=1}^{l} \lambda y_i x_i$ в лагранжиан, учтем ограничения $\sum_{i=1}^{l} \lambda_i y_i = 0$ и $C - \lambda_i - \mu_i = 0$

Двойственная задача SVM

$$\begin{cases} L = \sum_{i=1}^{l} \lambda_i - \frac{1}{2} \sum_{i,j=1}^{l} \lambda_i \lambda_j y_i y_j < x_i, x_j > \to \max_\lambda \\ \sum_{i=1}^{l} \lambda_i y_i = 0 \\ 0 \leq \lambda_i \leq C \end{cases}$$

(d) Если λ - решение, то $w = \sum_{i=1}^{l} \lambda_i y_i x_i$ - решение исходной задачи

(e) Задача зависит от объектов только через скалярное произведение $\to$ можно заменить его на ядро

(f) Находим b Берем $x_i : 0 < \lambda_i < C \to \xi_i = 0 \to y_i(< w, x_i > +b) = 1 \to b = y_i - < w, x_i >$

(g) Минусы ядрового SVM

i. Сложно контролировать переобучение

ii. Необходимо хранить в памяти матрицу Грамма

iii. Нельзя менять функцию потерь

11. **Применение ядерной модели**

(a) $a(x) = sign(< w, x > +b) = sign(< \sum_{i=1}^{l} \lambda y_i x_i, x > +b) = sign(\sum_{i=1}^{l} \lambda_i y_i < x_i, x > +b)$

2 Аппроксимации ядер, EM алгоритм

Скалярные произведения тяжело хранить из-за размера матрицы.
Есть ли возможность построить $\tilde{\phi}(x) \to < \tilde{\phi}(x_i), \tilde{\phi}(x_j) > \approx K(x_i, x_j)$

2.1 Метод случайных признаков Фурье

$$K(x, z) = K(x - z)$$

K - непрерывная функция
Теорема Бохнера

$$K(x - z) \to \exists p(w) \to K(x - z) = \int_{R^d} p(w) e^{iw^T(x-z)} dw$$

Используем:

$$K(x-z) = \int_{R^d} p(w) e^{iw^T(x-z)} dw \xrightarrow{\text{Формула Эйлера } [1]} \int_{R^d} p(w) cos(w^T(x-z)) + i \int_{R^d} p(w) cos(w^T(x-z))$$

$$\xrightarrow{K(x-z) \text{ - веществ.}} \text{Комплексная часть} = 0 \to K(x-z) = \int_{R^d} p(w) cos(w^T(x-z)) dw$$

$$\xrightarrow{\text{Монте-Карло } [2]} K(x - z) \approx \{w_j \sim p(w)\} : \frac{1}{n} \sum_{i=1}^{n} cosw_j^T(x - z)$$

$$= \frac{1}{n} \sum_{i=1}^{n} cosw_j^T x cosw_j^T z + sinw_j^T x sinw_j^T z$$

$$\tilde{\phi}(x) = \frac{1}{n}(cosw_1^T x, \ldots, cosw_n^T x, sinw_1^T x, \ldots, sinw_n^T x)$$

$$K(x - z) = < \tilde{\phi}(x), \tilde{\phi}(z) >$$

Для гауссова ядра:

$$p(w) = \mathcal{N}(0, 1)$$

[1] $e^{ix} = cosx + isinx$
[2] $\int_a^b f(x)dx = \frac{b-a}{n} \sum_{i=1}^{N} f(u_i)$

2.2 EM алгоритм

Смесь распределений:

$$\begin{cases} p(x) = \sum_{k=0}^{K} \pi_k p_k(x) \\ \sum \pi_k = 1 \end{cases}$$

Вероятностный эксперимент:
Выбираем K из $[\pi_1, \ldots, \pi_K]$, выбираем x из $pi_k(x)$
Z - скрытые переменные

$$Z = \{0,1\}^K, \sum Z_k = 1$$

$$p(Z_k = 1) = \pi_k$$

$$p(z) = \prod_{k=1}^{K} \pi_k^{Z_k}$$

$$p(x \mid Z_k = 1) = p_k(x)$$

$$p(x \mid z) = \prod_{k=1}^{K} (p_k(x)^{Z_k})$$

$$p(x,z) = p(x \mid z)p(z) = \prod_{k=1}^{K} (\pi_k p_k(x))^{Z_k}$$

$$p(x) = \sum_{k=1}^{K} p(x, z = k) = \sum_{k=1}^{K} \pi_k p_k(x)$$

Вероятностная кластеризация:
$p_k(x)$ - распределение k-го кластера

$$x \to (p_1(x), \ldots, p_k(x))$$

Хотим описать X смесью распределений

$$p(x) = \sum_{k=1}^{K} \pi_k \phi(x \mid \theta_k), \phi(x \mid \theta_k) \sim \mathcal{N}(\mu, \Sigma), \theta = (\mu, \Sigma)$$

Неполное правдоподобие:

$$ln(P(X \mid \Theta)) = \sum_{i=1}^{l} log \sum_{k=1}^{K} \pi_k \phi(x_i \mid \theta_k) \to max_\theta$$

Логарифм многооптимальная функция - просто оптимизировать ее сложно
Используем функцию полного правдоподобия

$$log(P, X \mid \Theta) = \sum_{i=1}^{l} log \sum_{k=1}^{K} (\pi_k \phi(x_i \mid \theta_k))^{Z_k}$$

$$\sum_{i=1}^{l} \sum_{k=1}^{K} Z_{ik}(log\pi_k + log\phi(x_i \mid \theta_k)) \to \max_{\Theta}$$

Известно аналитическое решение для нормального распределения.
Не знаем Z_ik

$$\Theta = (\pi_1, \ldots, \pi_k, \theta_1, \ldots, \theta_k)$$

Используем метод ALS для поиска Z, Θ

1. Оптимизация по скрытым переменным

   Апостериорное распределение: $p(Z \mid X, \Theta) = \frac{P(X,Z|\Theta)}{p(X|\Theta)}$

   $$Z^\star = \arg\max_Z p(Z \mid X, \Theta)$$

2. Оптимизировать по Θ

   $$logp(X, Z^\star \mid \Theta) \to \max_{\Theta}$$

3. Повторять до сходимости

Можно лучше. Не гарантирует сходимости
EM-алгоритм - метод обучения моделей со скрытыми переменными
EM-алгоритм

1. E-шаг - вычисляем $p(Z \mid X, \Theta)$ и запоминаем
2. M-шаг

$$E_{Z \sim p(Z|X,\Theta)} logp(X, Z \mid \Theta) = \sum_{Z} p(Z \mid X, \Theta) logp(X, Z \mid \Theta) \to \max_{\Theta}$$

Вывод EM-алгоритма

$$logp(X \mid \Theta) = Z(q, \Theta) + KL(q \mid\mid p)$$

$$L(q, \Theta) = \sum_Z q(Z) log \frac{p(X, Z \mid \Theta)}{q(Z)}$$

$$KL(q \mid\mid p) = -\sum_Z q(Z) log \frac{p(Z \mid X, \Theta)}{q(Z)}$$

$$\forall q(Z)$$

$L(q, \Theta)$ - нижняя оценка
Берем $q(Z) = p(Z \mid X, \Theta)$ - получаем E-шаг
$L(q, \Theta) = \sum_{Z \sim q(Z)} p(Z) log(...)$ - M-шаг
EM-алгоритм дает гарантии на рост правдоподобия

3 EM алгоритм 2

Свойства

1. $logP(X \mid \Theta^{new}) \geq logP(X \mid \Theta^{old})$
2. Если Θ_i не является станционарной точкой l, то $\Theta_{i+1} \neq \Theta_i$

$$\nabla l(\Theta_i) \neq 0$$
$$logP(X \mid \Theta_i) = L(q \mid \Theta_i) + KL(q(\Theta_i) \mid\mid p)$$
$$KL = 0 \to \nabla_\Theta KL(q(\mid\mid p) = 0 \to \nabla L(q \mid \Theta_i) \neq 0 \to$$
На M шаге точно сдвинемся и поменяем Θ

Теорема

$$Q(\Theta, \Theta^{Old}) = E_{z \sim p(Z \mid X, \Theta^{Old})} logP(Z, X \mid \Theta^{Old})$$

Пусть Q непрерывна по обоим аргументам
Тогда:

1. Все предельные точки последовательности Θ являются станционарными точками $logP(X \mid \Theta)$

2. $logP(X \mid \Theta)$ монотонно сходится к $logP(X \mid \Theta^\star)$ - одной из станционарных точек

Отвлеченная штука:
X - обучающая выборка
Хотим подогнать под нее распределение $p(x \mid \theta)$
Эмпирическое распределение:

$$\hat{p}(x \mid X) = \frac{1}{l}\sum_{i=1}^{l}[x = x_i]$$

Минимизировать KL-дивергенцию между эмпирическим и параметрическим распределением.

$$KL(\hat{p}(x \mid X) \mid\mid p(x \mid \theta)) \to min_\theta$$

$$= \sum_{i=1}^{l}\frac{1}{l}log\frac{1/l}{p(x_i \mid \theta)} = \sum_{i=1}^{l}\frac{1}{l}log(1/l) - logp(x_i \mid \theta) \to$$

$$\sum_{i=1}^{l} llogP(x_i \mid \theta) \to \max_\theta$$

4 Поиск аномалий

В обучении есть только один класс - неаномальный, надо научится отделять от него аномалии
Несбалансированная классификация

1. (Under/over)sampling - взвешенный функционал ошибки
2. Синтетические объекты
   (a) SMOTE
      i. Выбираем объекты X_1 из минорного класса, выбираем случаный объект из k ближайших соседей тоже из минорного класса X_2
      ii. Новый объект: $X = \alpha X_1 + (1-\alpha)X_2, \alpha \sim U(0,1)$
      iii. Предполагает существование объектов между X_1, X_2
   (b) Аугментации

Одноклассовая классификация
Бенчмарк: Классификация X на нормальные и аномальные, стандартные метрики

1. Статистический подход - описываем плотностью $p(x)$ для новых объектов смотрим на вероятность - $p(x)$ - novelty score.

   Откуда брать p

   (a) Параметрический подход:

   $$\sum_{i=1}^{l} logP(x \mid \theta) \to \max_{\theta}$$

   (b) Непараметрический подход:

   i. d = 1:

   $$p(x) = \lim_{h \to 0} \frac{1}{2h} P(\xi \in [x-h, x+h])$$

   $$\hat{p}(x) = \frac{1}{2h} \frac{1}{l} \sum_{i=1}^{l} \mid [x_i - x \mid < h] =$$

   $$= \frac{1}{lh} \sum_{i=1}^{l} \frac{1}{2} [\frac{\mid x_i - x \mid}{h} < 1]$$

   Можно заменить на более гладкую плотность:

   $$\frac{1}{lh} \sum_{i=1}^{l} \frac{1}{2} K(\frac{x_i - x}{h})]$$

   A. $K(z) = K(-z)$
   B. $\int_{\mathcal{R}} K(z)dz = 1$
   C. $K(z) \geq 0$
   D. Не возрастает при $Z > 0$

   ii. d > 1:

   $$\hat{p}(x) = \frac{1}{lV(h)} \sum_{i=1}^{l} K(\frac{\rho(x_i, x)}{h})$$

   $$V(h) = \int K(\frac{\rho(x_i, x)}{h}) dx$$

   h - гиперпараметр

2. Метрический подход

   x - аномалия, если он далеко от других объектов

   Смотреть на количество объектов в ϵ-окрестности?

   Плохой подход:

   Надо смотреть не на единую окрестность, а смотреть на плотность объектов в отдельной точке и на основе нее оценивать окрестность

   Определения:

   (a) $\rho_k(x)$ - такое минимальное число n, что:
   Для $\geq$ k объектов из $X/\{x\}$ выполнено $\rho(x,z) \leq n$
   Для $\leq$ k-1 объектов выполнено $\rho(x,z) < n$
   По сути: расстоение до k-го ближайшего соседа

   (b) K-окрестность:

$$\mathcal{N}_k(x) = \{z \in X/\{x\}\} : \rho(x,z) \leq \rho_k(x)$$

   (c) Reachibility Distance:

$$rd_k(x,z) = max(\rho_k(z), \rho(x,z))$$

   Позволяет сгладить расстояние между объектами

   (d) Local Reachibility Distance

$$lrd_k(x) = \frac{1}{\frac{1}{|\mathcal{N}_k(x)|} \sum_{z \in \mathcal{N}_k(x)} rd_k(x,z)}$$

   Обращенное среднее расстояние от x до ближайших соседей

   (e) Local Outlier Factor

$$LOF_k(x) = \frac{\frac{1}{|\mathcal{N}_k(x)|} \sum_{z \in \mathcal{N}_k(x)} lrd_k(z)}{lrd_k(x)}$$

   Отлавливаем объекты у которых соседи находятся в плотных областях, но сами они находятся далеко от соседей

3. Model-based AD

   (a) Есть примеры нормальных объектов

(b) Хотим найти наименьшую область, содержащую все объекты

$$a(x) = sign(< w, x > -\rho)$$

Идея:

Отделяем X от начала координат с помощью a()

$$\begin{cases} \frac{1}{2} \mid\mid w \mid\mid^2 + \frac{1}{\nu\ell} \sum \xi_i - \rho \to \min_{w,\xi,\rho} \\ < w, x_i > \geq \rho - \xi_i, \forall i \\ \xi_i \geq 0 \end{cases}$$

ν - гиперпараметр

$\sum [a(x) = -1] \leq \nu$

Требования к решению:

i. Отделить как можно больше объектов от 0. За это отвечает $\sum \xi_i$
ii. Максимизировать отступ. За это отвечает $\mid\mid w \mid\mid^2$
iii. Гиперплоскость как можно дальше от нуля. За это отвечает ρ

$$a(x) = sign(< w, x > -\rho)$$

Можно записать двойственную задачу:

$$\begin{cases} \frac{1}{2} \sum \lambda_i \lambda_j K(x_i, x_j) \to \min_\lambda \\ 0 < \lambda_i \leq \frac{1}{\nu\ell} \\ \sum \lambda_i = 1 \end{cases}$$

С помощью ядра получаем компактную область

4. Random projections

(a) Isolation Forest

Строим жадное дерево со случайными предикатами по случайным признакам

Если в каком-то листе оказывается 1 объект - прекращаем разбиение

Аномальные объекты рано получают свой лист

Обучение:

Строим лес из N деревьев, в каждом случайные предикаты.
Максимальная глубина $D = log_2 \ell$
Применение:
$h_n(x)$ - оценка аномальности x с точки зрения n дерева
$K_n(x)$ - глубина листа в который попадает x в n дереве
Нужно сделать поправку на количество объектов в листе

$$h_n(x) = K_n(x) + C(m_n(x))$$

$$C(m) = 2H(m-1) - 2\frac{m-1}{m}$$

$$H(i) \approx lni + 0.577$$

Можно использовать и $log_2(m)$

$$a(x) = 2^{-\frac{\frac{1}{N}\sum_{n=1}^{N} h_n(x)}{C(l)}}$$

$C(l)$ - средняя длина пути

(b) Extra Random Trees
Берем индикатор попадания в листья и строим линейную модель

Как измерять качество:

1. Anomaly detection

   Есть примеры аномалий, но мало данных

   Смотрим какое количество аномалий модель угадывает

2. Novelty detection

   Аномалии не даны, качество модели оценивается глазами

5 Обучение без учителя

1. Кластеризация: DBScan, Спектральная классификация, Affinity Propagation
2. Внешние метрики качества кластеризации
3. Тематическое моделирование

K-means:
Основная проблема - ищет сферические кластеры

5.1 DBScan

Типы объектов:

1. Ядровые:

   В ϵ-окрестности находится N объектов

2. Погранченые объекты:

   Достижимы из ядровых, находится в ϵ-окрестности ядрового

3. Выбросы:

   Все остальные

Псевдокод:

```
K = 0 #Num clasters
rho #metric
epsilon, N #hyperparam
for i = 1 ... l:
    if label(x[i]) != 0:
        continue
    #point neighborhood
    U = {x in X | rho(x[i], x[j] <= epsilon)}
    if |U| < N:
        label(x[i]) = noise
        continue
    K++ # found new claster if |U| > N
    label(x[i]) = K
    U = U \ {x[i]}
    for x[j] in U:
        if label(x[j]) = noise:
            label(x[j]) = K
        if label(x[j]) != 0:
            continue
        label(x[j]) = K
        #point neighborhood
        R = {xm in X | rho(x[m], x[j]) < epsilon}
        if |R| >= N:
            #new core object neighborhood included in U
            U = U & R
```

Преимущества:

1. Находит кластеры сложной формы
2. Находит выбросы
3. Быстрый
4. Не надо задавать число кластеров

Недостатки:

1. Проблемы если кластеры разной плотности
2. Проблемы с точками на краях
3. Не работает если кластеры характеризуются неплотность
4. Не параллелится

5.2 Иерархическая кластеризация

Цель: Найти кластерную структуру
Визуализировать разную структуру кластеров при разном их количестве
Агломеративная кластеризация
Начинаем с того, что каждый объект является кластером
Псевдокод:

```
C = {{x[1]}, {x[2]}, ..., {x[l]}}
for n = 2, ..., l:
    G, H = argmin rho(G, H) #find nearest clasters
    C = (C \ {G, H}) U {G U H}
```

Функция расстояния между кластерами ρ:

1. Single Linkage:
$$\rho_{sl}(G, H) = \min_{x_i \in G, x_j \in H} \rho(x_i, x_j)$$

   Чувствителен к выбросам

   Главная проблема: Chaining

   Алгоритм подцепляет отдельные объекты, а не кластеры

   Дендрограмма - картинка присоединения объектов

2. Complete linkage

$$\rho_{cl}(G,H) = \max_{x_i \in G, x_j \in H} \rho(x_i, x_j)$$

Кластеры не будут компактными

3. Group Average

$$\rho_{ga}(G,H) = \frac{1}{\mid G \mid\mid H \mid} \sum_{x_i \in G, x_j \in H} \rho(x_i, x_j)$$

5.3 Графовая кластеризация

G = (X, E)
E - ребра:

1. Полный граф - все вершины связаны

   $w_{ij} = exp(-\frac{\|x_i - x_j\|^2}{2\sigma^2})$

2. KNN-граф:

   $w_i j \neq 0 \Leftrightarrow x_i, x_j$ ближайшие соседи

3. ϵ-граф:

   $w_i j \neq 0 \Leftrightarrow \rho(x_i, x_j) \leq \epsilon$

Поиск решение

1. Найти связные компоненты (для 3его варианта)

   Тупой метод

2. Минимальное остовное дерево (Алгоритм Краскала)

   (a) Начинаем с отдельных вершин
   (b) Сливаем два кластера с максимальным ребром между ними
   (c) Повторить пока не будет K кластеров
   (d) Это агломеративная кластеризация с sl
   (e) Решает задачу:

$$\max \min_{x_i \in G, x_j in H} \rho(x_i, x_j)$$

3. Спектральная кластеризация

$$A, B \subset X, A \cap B = \emptyset$$
$$W(A, B) = \sum_{x_i \in A, x_j \in B} w_i j$$
$$X = A_1 \cup A_2 \cup \dots A_k$$

Ошибка кластеризации:

$$\text{Ratio Cut}(A_1, \dots, A_k) = \frac{1}{2} \sum_{i=1}^{K} \frac{w(A_i, \bar{A}_i)}{\mid A_i \mid} \to \min_{A_i, \dots, A_k} (\star)$$
$$\bar{A}_i = X \setminus A_i$$

Хотим, чтобы ребра между кластерами были как можно менее значимы → Каждый кластер должен быть изолированным

$K = 2 \to$ Задача поиска максимального потока

$K > 2 \to$ NP-полная задача

$G = (X, E)$

$d_i = \sum_{j=1} dw_i j$ - сумма ребер, которые с ней связаны

$D = diag(d_1, \dots, d_l)$

$L = D - W$, W - матрица смежности, L - **лаплассиан**

Свойства лаплассиана

(a)
$$f \in R^d$$
$$f^T L f = \frac{1}{2} \sum_{i,j=1}^{l} w_{ij} (f_i - f_j)^2$$

(b) L - симметричная

(c) L - неотрицательно определенная

Теорема

(a) Кратность $\lambda = 0$ у L равна числу компонент связности графа
Кратность:

i. Собственные значения: $Ax = \lambda x$

ii. $det(A - \lambda I) = 0 \to \lambda_i$

iii. λ_i - решение

iv. $\lambda_i, \forall i$ - спектр графа

v. Характеристическое уравнение выражаем в виде характеристического многочлена и раскладываем его в виде решений

$$P_A(\lambda) = (-1)^n \prod_{i=1}^{n} (\lambda - \lambda_i) = (-1)^n (\lambda - \lambda_1)^{k_1} \dots$$

k_1 называется кратностью для λ_1

(b) $A_1, \dots, A_k$

Вектор индикатор: $f_i = ([x_j \in A_i])_{j=1}^{\ell}$

$f_1, \dots, f_k$ - собственные векторы для $\lambda = 0$

Доказательство:

K = 1:

(a) Является ли $\lambda = 0$ собственным значением

$$f = (1, \dots, 1)$$

$$Lf = Df - Wf = \begin{pmatrix} d_1 \\ \vdots \\ d_\ell \end{pmatrix} - \begin{pmatrix} \sum w_{1,j} \\ \vdots \\ \sum w_{\ell,j} \end{pmatrix} = 0$$

(b) Кратность $\lambda = 0 = 1 \to$ нет других собственных векторов

Допустим:

$$\exists f' \in R^\ell : \exists p \neq q, f'p \neq f'q, Lf' = 0$$

$$(f')^T L f' = \frac{1}{2} \sum_{i,j=1}^{\ell} w_{ij} (f'_i - f'_j)^2 = 0$$

$$\forall i, j : \begin{bmatrix} w_{ij} = 0 \text{ Нет ребра} \\ f'_i = f'_j \end{bmatrix}$$

Граф G связный $\to$ Существует путь из p в q $\to$

Путь: $w \to i_1 \to \dots \to p$

$w_{pi_1} \neq 0 \to f'p = f'_{i_1} \to w_{i_1 \dots} \neq 0 \to \dots \to f'_p = f'_{i_1} = \dots = f'_q$

$\perp$

K $>$ 1:

Можно упорядочить объекты так, чтобы L был блочно-диагональным

$$L = \begin{pmatrix} L_1 & 0 & 0 \\ 0 & L_2 & 0 \\ & \ddots & \\ 0 & 0 & L_K \end{pmatrix}$$

Спектр блочно-диагональной матрицы = объединение спектров отдельных блоков

$L_i \to f_i = ([x_j \in A_i])_{j=1}^{\ell}$

Кратность $\lambda = 0$ равна K

Гипотеза: x_i, x_j - похожие объекты $\Rightarrow$ у собственного вектора f_i для $\lambda_i \approx 0$, будет $f_ij \approx f_ik$

Для связанных графов не берем $\lambda = 0$, т.к. тогда будет одна компонетна

Алгоритм:

(a) Строим лапласиан

(b) Находим m (гиперпараметр) нормированных собственных векторов $u_1, \ldots, u_m$, соотв. наименьшим собственным значениям.
Сложность: $O(l^3)$

(c) $U = (u_1 \mid \ldots \mid u_m) \in R^{\ell \times m}$

(d) Новые признаки близки для объектов в одной плотной области

(e) K-means

Как это связано с задачей $(\star)$**:**

Если эту задачу релаксировать и искать не жесткое приписывание к классам, а распределение, то ее решение U.

6 Метрики качества классификации, тематическое моделирование

6.1 Affinity Propagation

Цель - найти типовые объекты и на их основе выделять кластеры

Сходство вершин:

$$s(i,k) = -\|x_i - x_k\|^2$$

$r(i,k)$ - Насколько x_k является типовым объектом для x_i
$a(i,k)$ - Насколько у x_i важный голос для типового объекта
Инициализируем 0 и итеративно рассчитываем показатели:

$$r(i,k) = s(i,k) - \max_{k' \neq k}(a(i,k') + s(i,k'))$$

Если рядом есть более близкие объекты, чем k, то k не очень хороший представитель.

$$a(i,k) = \min(0, r(k,k) + \sum_{i' \neq i, i' \neq k} max(r(i',u)))$$

$$\alpha(x_i) = \arg\max_{k \in \{1,\ldots,\ell\}}(r(i,k) + a(i,k))$$

6.2 Оценка качества кластеризации

1. Внутренние
   (a) Без использование лейблов
   (b) Внутрикластерное расстояние
   (c) Межкластерное расстояние
2. Внешние
   (a) Знаем истинные номера кластеров $y_1, \ldots, y_\ell$
   (b) Номера кластеров нельзя сравнивать с истинными
   (c) Посчитать K! перестановок и найти классы?
   (d) **Требования к метрике**
      i. Гомогенность
      Значение метрики качества должно уменьшаться при объединении в один кластер двух эталонных
      ii. Полнота
      Значение метрики качества должно уменьшаться при разделении эталонного кластера на части

iii. Rag bag
Значение метрики качества должно быть выше у той версии кластеризации, которая помещает новый нерелевантный обоим кластерам элемент в шумный кластер, по сравнению с версией, которая помещает этот элемент в чистый кластер

iv. Cluster size vs quantity
Значительное ухудшение кластеризации большого числа небольших кластеров должно обходиться дороже небольшого ухудшения кластеризации в крупном кластере.

(e) **Метрики**

i. Bcubed
$y(x)$ - номер кластера в истинной разметке
$a(x)$ - выход кластеризации
Correctness:

$$C(x_i, x_j) = \begin{cases} 1, y(x_i) = y(x_j) \quad a(x_i) = a(x_j) \\ 0, otw \end{cases}$$

$$\text{Precision-Bcubed} = Avg_{x_i}(Avg_{x_j, a(x_i)=a(x_j)}(C(x_i, x_j)))$$

$$\text{Recall-Bcubed} = Avg_{x_i}(Avg_{x_j, y(x_j)=y(x_i)}C(x_i, x_j))$$

$$F_{\text{Bcubed}} = 2\frac{\text{PrecisionRecall}}{\text{Precision} + \text{Recall}}$$

6.3 Подбор метрик для продукта

Вводим набор ухудшений

7 Тематическое моделирование

Методы кластеризации для текстов
Есть T тематик
x_d - документ
$\theta_d \in R^T$ - распределение тематик для документа
$\phi_t in R^W$ - тема описывается распределением на словах
W размер словаря

7.1 LSA (Latent Semantic Analysis)

$$X \in R^{d \times W}$$

X_{dw} - сколько раз слово w входит в документ d

$$X = \Theta \times \Phi, \Theta \in R^{d \times T}, \Phi in R^{T \times W}$$

$$x_{dw} = \sum_{t=1}^{T} \theta_{dt} \times \phi_{wt}$$

Делаем SVD для разложения

7.2 PLSA

Хотим ввести вероятности

$$p(t \mid d) = \theta_{td}$$
$$p(w \mid t) = \phi_{wt}$$

Генерация текста x_d

1. Сэмплируем тему $t \sim p(t \mid d)$
2. Сэмплируем слово $w \sim p(w \mid t)$
3. Добавляем слово в текст
4. Повторяем до нужной длины

θ_{dt}, ϕ_{tw} - параметры модели
Неполное правдоподобие данных:

$$\begin{cases} \sum_{d=1}^{D} \sum_{t=1}^{T} \sum_{w=1}^{W} [t_{dj} = t] log \phi_{w_{d_j t}} \theta_{td} \\ \theta_{td}, \phi_{wt} \geq 0 \\ \sum \theta, \sum \phi = 1 \end{cases}$$

Тема является скрытой переменной
Можно построить EM-алгоритм по этой задаче:
E-шаг: $p(t_{dj} \mid d, w_{dj})$
M-шаг: $\phi_{wt}, \theta_{td} = ?$

7.3 LDA (Latent Dirichlet Allocation)

В PLSA не требуем невырожденности распределений
Дополнительно вводим распределения параметров:
$\Phi_t = Dir(\alpha)$
$\Theta_d = Dir(\beta)$

$$Dir(x_1, \ldots, x_n, \alpha) = \frac{\Gamma(\alpha n)}{\Gamma^n(\alpha)} \prod_{i=1}^{n} x_i^{\alpha-1}$$

Распределение на дискретных распределениях с n исходами

8 Частичное обучение (semisupervised)

Используется в случае:
Есть обучающая выборка $X^l = \{(x_i, y_i)\}_{i=1}^{\ell}$
Есть неразмеченная выборка: $X^u = \{(x_i)\}_{i=\ell+1}^{n}$
Самая большая ценность состоит в необычных объектах
Мотивация: Неразмеченные данные собрать проще, чем размеченные
Категории задачи:

1. Semisupervised learning: $X_\ell \cup X_u \to a(x)$
2. Transductive learning $X_\ell \cup X_u \to$ Найти метки для X_u

Методы:

1. Self-training
   (a) Обучить a(x) на X^ℓ
   (b) Применить на X^u
   (c) Добавить $(x_i, a(x_i))$ в X^ℓ объекты на которых модель наиболее уверена
   Критерий:
      i. Для классификации: самые большие уверенности
      ii. Брать по порогу расстояния к обучающим
      iii. Всю X^u с весами на основе уверенности модели
      Взвешиваем лосс
   (d) Повторить

2. Генеративные модели

   Описываем каждый класс нормальный распределением

   Если бы были только X^ℓ:

   Правдоподобие и максимизация по параметрам:

$$\sum_{i=1}^{\ell} logP(y_i \mid \theta)P(x_i \mid y_i, \theta) \to \max_\theta$$

   Если X^ℓ и X^u:

   Неразмеченные данные хотим описать как смесь распределений классов

$$\sum_{i=1}^{\ell} logP(y_i \mid \theta)P(x_i \mid y_i, \theta) + \sum_{i=\ell+1}^{n} log \sum_{y=1}^{K} p(y \mid \theta)p(x_i \mid y, \theta) \to \max_\theta$$

   Используем EM-алгоритм для поиска θ

   Второе слагаемое может перевесить - нужно добавить λ

3. Упрощенная версия: Cluster-and-label:

   Обучаем алгоритм кластеризации и помечаем в каждом кластере объекты самым популярным классом

4. Методы на основе моделей

   (a) Логит $\to$ Expectation Regularization

   (b) Semi-Supervised SVM = S3VM

   Безусловная задача оптимизации SVM:

$$\|w\|^2 + C \sum_{i=1}^{\ell} \max(0, 1 - y_i < w, x_i >) \to \min_w$$

   Лосс > 0 когда расстояние до гиперплоскости отрицательное

   Для unsupervised: чего требовать?

   Если объект близко к гиперплоскости - штрафуем, если далеко - не штрафуем

   Задача:

$$\|w\|^2 + C_1 \sum_{i=1}^{\ell} \max(0, 1 - y_i < w, x_i >) + C_2 \sum_{i=\ell+1}^{n} \max(0, 1 - |< w, x_i >|) \to \min_w$$

Может подобрать гиперплоскость, которая просто лежит далеко от данных и не разделяет их

Можно потребовать, чтобы баланс классов был такой же, как и на размеченных данных

$$\frac{1}{n-\ell} \sum_{i=\ell+1}^{n} a(x_i) = \frac{1}{\ell} \sum_{i=1}^{\ell} y_i$$

Тогда гиперплоскость не может просто не разбивать классы

Очень сложная задача с точки зрения оптимизации- максимум, модуль, ограничения

CCCP: ConCave Convex Procedure

Метод для оптимизации суммы выпуклой и вогнутой функции

5. Графовые методы

$$a(x) = \infty \sum_{i=1}^{\ell} (a(x_i) - y_i)^2 + \sum_{i,j=1}^{n} w_{ij}(a(x_i) - a(x_j))$$

$$w_{ij} = exp\left(-\frac{\|x_i - x_j\|^2}{2\sigma^2}\right)$$

Согласованные метки на соседних объектах для неразмеченных

Бесконечно более важно оптимизировать первый элемент

Упрощение задачи (Manifold Regularization):

$$\sum_{i=1}^{\ell} L(y_i, a(x_i)) + \lambda_1 R(a) + \lambda_2 \sum_{i,j=1}^{n} w_{ij}(a(x_i) - a(x_j))$$

$$\sum_{i,j=1}^{n} w_{ij}(a(x_i) - a(x_j)) = a^T L a$$

9 Метрические методы

Case-based reasoning - не очень зависит от параметров
$\rho : X \times X \to (0, +\infty)$ - функция расстояния
Обучение: запоминаем X
Применение:
u - новый объект
Строим вариационный ряд:

$$\rho(u, x_1) \leq \rho(u, x_2) \dots$$

Классификация:

$$a(u) = \arg\max_{y \in Y} \sum_{k=1}^{K} w(K, u, x_k)[y_k = y]$$

Веса учитывают расстояния до точки:

$$w(K, u, x_k) = K(\frac{\rho(u, x_k)}{h})$$

Регрессия (Формула Надарая-Ватсона):

$$a(u) = \frac{\sum_{k=1}^{K} w y_k}{\sum_{k=1}^{K} w}$$

Зачем нужен kNN?

1. Если легко задать расстояния, но сложно придумать признаки
2. Если задача решается через сходство
3. Мало представителей каждого класса

Оптимальность kNN:

$Y = \{-1, +1\}$
$p(y = +1 \mid x)$
u - хотим классифицировать
x_u - ближайший сосед
p^*_{bayes} - вероятность ошибки опт. Байесовского класс
$p_{1nn} \leq 2p_{bayes}, if l \to \infty$

Пример хитрой метрики
Хотим сделать расстояние на текстах
$C(i, j)$ - расстояние между представлениями i и j слов
t_{ij} - количество смысла, перетекающего из x_i в z_j
x_i - число вхождений некоторого слова i из словаря в текст x
$\sum_{j=1}^{d} t_{ij} = x_i$
$\sum_{i=1}^{d} t_{ij} = z_j$
$t_{ij} \geq 0$
Стоимость переноса смысла
$\sum_{i,j=1}^{d} t_{ij} C(i, j) = \mu(x, z) \to min_{t_{ij}}$
Для оптимального t_{ij}: $\mu(x, z) = \rho(x, z)$
Задача оптимизации: min cost max flow

9.1 Быстрый поисх ближайших соседей

Зачем?

1. Задачи retrieval
2. Рекомендации
3. ...

9.1.1 Точные методы

1. KD-деревья и другие структуры

   При росте d сложность приближается к линейной

9.1.2 Приближенные решения

1. LSH - locality sensitive hashing

   (a) Определение:
   Семейство функций $\mathcal{F} = \{f : X \to H\}$ с распределением P(f) называется (d_1, d_2, p_1, p_2) чувствительным, если

   i. $p(x, z) \leq d_1 \Rightarrow P_{f \in \mathcal{F}}[f(x) = f(z)] \geq p_1$
   ii. $p(x, z) \geq d_1 \Rightarrow P_{f \in \mathcal{F}}[f(x) = f(z)] \leq p_2$

Пример: MinHash

Объекты - множества, $x \subset U = \{u_1, \ldots, u_n\}$

π - перестановка на множестве U

$f_\pi(x) = \min\{\pi(i) \mid u_i \in A\}$

Утверждение:

$$P_\pi[f_\pi(A) = f_\pi(B)] = \frac{\mid A \cap B \mid}{\mid A \cup B \mid}$$

Док-во:

Три категории $u \in U$:

i. $u \in A, u \in B$

ii. $u \in A, u \notin B$
или
$u \notin A, u \in B$

iii. $u \notin A, u \notin B$
$\pi = \begin{pmatrix} u_1 \\ u_2 \\ u_3 \end{pmatrix}$
Одинаковые хэши для A и B?
u из первой группы должны иметь меньший хэш
Какова доля перестановок, где хотя бы 1 элемент первого типа идет раньше всех второго типа:
$\frac{p}{p+q} = \frac{|A \cap B|}{|A \cup B|}$
$\rho(A, B) = 1 - \frac{|A \cap B|}{|A \cup B|}$
$\rho(A, B) \leq d_1 \Rightarrow P_{f \in \mathcal{F}}[f(x) = f(z)] = \frac{|A \cap B|}{|A \cup B|} = 1 - \rho(A, B) \geq 1 - d_1$
$\rho(A, B) \geq d_2 \Rightarrow P \leq 1 - d_1$
MinHash - $(d_1, d_2, 1 - d_1, 1 - d_2)$ чувствителен

(b) Для косинусного расстояния:
$\rho(x, y) = \arccos \frac{<x,y>}{\|x\|\|y\|}$
$\mathcal{F} = \{f_w(x) = sign < w, x > \mid w \in R^d\}$
Геометрически: Накидываем случайные гиперплоскости w и смотрим с какой стороны от них находятся точки
Если между точками угла маленький - вероятность их рассечения плоскостью мала

(c) Для евклидова расстояния:
$\mathcal{F} = \{f_{w,b}(x) = \frac{<w,x>+b}{r} \mid w \in R^d, b \in [0, r)\}$
Геометрически: проводим случайную прямую, разбиваем на отрезки длины r. Значение хэш функции - номер отрезка
$w \sim \mathcal{N}(0, I)$ $b \sim U[0, r)$
Если варьировать распределения, то можно получить другие метрики:
Метрики Минковского - $\rho \in (0, 2]$
Манхэттенская метрика ($\rho = 1$): $w \sim Cauchy$

(d) Проблема этих методов

i. Не очень устойчиво из-за вероятностного подхода
ii. При большом расстоянии все равно есть вероятность совпадения хэшей
iii. Хотим форму сигмоиды, а не линейно убывающую вероятность равенства хэшей

(e) Композиция хэш-функций:
$g(x) = (f_1(x), \ldots, f_m(x))$
Алгоритм: $x \to g(x) \to \{x_i \in X \mid g(x_i) = g(x)\} \to$ Ищем ближайших соседей среди N(x)
$d = \rho(x, z)$
$P(f_1(x) = f_1(z)) = p$
$P(g(x) = g(z)) = p^m$
Степенная функция это не то, что нам нужно
Модифицируем:
$g_1(x) = (f_{11}(x), \ldots, f_{1m}(x))$
$\vdots$
$g_L(x) = (f_{L1}(x), \ldots, f_{Lm}(x))$
$P[[g_1(x) = g_1(z)] \,||\, \ldots \,||\, [g_L(x) = g_L(z)]] = 1 - (1 - p^m)^L$

(f) Теория:
Алгоритм решает задачу поиска c-ближайшего соседа, если для нового объекта u с вероятностью $1-\epsilon$, алгоритм возвращает объект выборки $z \in X : \rho(u, z) \leq C\rho(u, x_\star)$, где $x_\star$ - ближайший сосед u
Для LSH:
$\exists L, m : O(d\ell^n log\ell)$ - время поиска в LSH

2. **NSW** (Navigable Small World)

   (a) Small world graph - если сгенерировать случайный граф - среднее расстояние между двумя парами вершин очень близко

   (b) Представляем выборку в виде графа, где у каждой вершины небольшая степень, но выполняется свойство малого мира

   (c) $G = (X, E)$
   Задаем алгоритм жадного поиска:

      i. u - новый объект
      ii. Берем случайную вершину v в G
      iii. В цикле: Среди всех соседей v ищем вершину $v' : \rho(v', u) < \rho(v, u)$
      iv. Если такой сосед нашелся - переходим в него
      v. Используем мультистарт
      Находим множество результатов C_u, можно расширить это множестве окрестностями C_u - выбираем ближайшие

   (d) Добавление вершины u:

      i. Мультистарт - $C(u)$
      ii. $D(u) = C(u) \cup$ окрестности вершины
      iii. Соединяем u с k ближайшими соседями из D(u)

   (e) Особенность метода:

      i. В графе есть области плотности и связующие цепочки
      ii. Свойство малого мира достигается за счет связующих цепочек - вершин с очень высокой степенью

3. **HNSW** (hierarchical NSW):

   (a) На следующий уровень пропускам только log от числа вершин с некотрой вероятностью

   (b) Начинаем с самого верхнего уровня графа: находим ближайшего соседа

   (c) Начинаем из этой точки на более низком уровне, ищем новую точку

   (d) ...

10 Задача ранжирования

$X = \{x_1, \ldots, x_l\} \subset X$
$(i, j) \in R \subset \{1, \ldots, \ell\}^2 \Rightarrow a(x_i) > a(x_j)$
$\{1, \ldots, \ell\}^2$ - множество всех пар
Обычно так:
$x = (q, d)$ - q - запрос, документ
в R - пары $x_i = (q_i, d_i), x_j = (q_j, d_j)$, где $q_i = q_j$
Метрики качества ранжирования

1. Наивный подход:
   (a) $R \to y_1, \ldots, y_\ell : (i, j) \in R \Rightarrow y_i > y_j$
   (b) Обучаем $a(x_i) \approx y_i$, метрика - точность прогнозов
   (c) Модель может правильно ранжировать, но при этом выдавать лейблы далекие от исходных и качество будет плохим при хорошем ранжировании
2. Дефектные пары:
   $\frac{1}{|R|} \sum_{(i,j)\in R} [a(x_i) \leq a(x_j)]$
3. precision@k:
   Работает, когда таргет является разметкой релевантности документов под запрос $y \in \{0, 1\}$
   $\sum_{i=1}^{K} [y(i) = 1]$
   Проблема:
   Не учитывает порядок выдачи в топ k документов
4. Average Precision@k(q)
   $AP@k = \sum_{i=1}^{K} \frac{y_i}{\sum_{j=1}^{K} y_j} \times precision@i$
5. MAP@k
   Q - множество запросов
   $MAP@k = \frac{1}{|Q|} \sum_{q\in Q} AP@k(q)$
6. DCG@k (для небинарных меток)
   $DCG@k(q) = \sum_{i=1}^{K} g(y_i) d(i)$
   $g(y) 2^y - 1$
   $d(i) = \frac{1}{log(i+1)}$

7. nDCG@k(q)

   $nDCG@k(q) = \frac{DCG@k(q)}{maxDCG@k(q)}$

8. Каскадные метрики

   pFound:

   Пытается промоделировать поведение пользователей

   $y \in [0, 1]$ - вероятность найти ответ в документе

   p_i - вероятность, что пользователь дойдет до i-ой позиции в выдаче

   $p_i = 1$

   $p_{i+1} = p_i(1 - y_i)(1 - p_{out})$

   $p_o ut$ - вероятность, что пользователь уйдет

   $pFound@k(q) = \sum_{i=1}^{K} p_i \times y_i$

Признаки для моделей ранжирования

1. Запросные признаки
   (a) Эмбеддинг запроса
   (b) Популярность
   (c) Категория
   (d) Персонализация
   (e) Признаки про пользователя
2. Статические - только про документ
   (a) Эмбеддинг документа
   (b) Категория документа
   (c) PageRank

   $PR(d) = \frac{1-\delta}{|D|} + \delta \sum_{c \in D_d^{in}} \frac{PR(c)}{|D_c^{out}|}$

   D_d^{in} - мн-во документов, ссылающихся на d

   D_c^{out} - мн-во док., на которые ссылается c
3. Динамические признаки - про пару/запрос документ
   (a) Косинусное расстояние между эмбеддингами документа и запроса

(b) BM25

$q = q_1, q_2, \ldots, q_n$ - слова

$BM25(q, d) = \sum_{i=1}^{n} IDF(q_i) \times \frac{TF(q_i,d)\times(K_1+1)}{TF(q_i,d)+K_1(1+b+b\frac{|D|}{n_d})}$

Методы ранжирования

1. Pointwise - поточечные методы

   $q : (d_1, y_1), \ldots, (d_{n_q}, y_{n_q})$

   $\sum_{q\in Q} \sum_{i=1}^{n_q} L(y_i, a(q, d_i)) \to \min_a$

2. Попарные методы

   Главное, чтобы пары были правильно расположены относительно друг друга

   $(i, j) \in R \Rightarrow a(x_i) > a(x_j)$

   $\frac{1}{|R|} \sum_{(i,j)\in R} [a(x_i) < a(x_j)] rightarrow \min_a$

   $\frac{1}{|R|} \sum [a(x_i) - a(x_j) < 0] \leq$

   $\leq \frac{1}{|R|} \sum_{(i,j\in R)} \tilde{L}(a(x_i) - a(x_j)) \; rightarrow \min_a$

   $\frac{1}{|R|} \sum log(1 + e^{a(x_j)-a(x_i)}) \to \min_a$

   Частный случай (RankNet):

   $a(x) =< w, x >$

   $\tilde{L}(z) = log(1 + e^{-\sigma z})$

   $w^t = w^{t-1} + \eta \frac{\sigma}{1+exp(\sigma<x_j-x_i,w>}(x_j - x_i)$

   Используем метрику

   δF_{ij} - как изменится метрика качества, если поменять $x_i x_j$ местами

   LambdaRank:

   $w^t = w^{t-1} + \eta \frac{\sigma}{1+exp(\sigma<x_j-x_i,w>} \mid \delta F_{ij} \mid (x_j - x_i)$

3. Списочные методы

   Напрямую оптимизируем метрики качества:

   ListNet

$q_1, \ldots, q_m$

$q : d_1, \ldots, d_{n_q}$

$a(q, d_1) = z_1, \ldots, a(q, d_{n_q}) = z_{n_q}$

$$\frac{1}{m} \sum_{i=1}^{m} nDCG@k(q_i) \to \max$$

$$nDCG@k(q) = \sum_{i=1}^{K} \frac{2^y(i)}{log(i+1)}$$

Параметры модели зашиты в порядке ранжирования

$$nDCG@k(q, \pi(a)) = \sum_{i=1}^{K} \frac{2^{y(\pi(i)}}{log(i+1)} \downarrow$$

$$E_\pi nDCG@k(q, \pi) = \sum_{\pi \in Sym(q_1, \ldots, q_{nq})} p(\pi) nDCG@k(q, \pi)$$

ϕ - неубывающая строго + функция

$$p_z(\pi) = \prod_{j=1}^{n_q} \frac{\phi(z_{\pi(j)})}{\sum_{k=j}^{n_q} \phi(z_{\pi(k)})}$$

Свойства этого распределения:

(a) Распределение на мн-ве всех перестановок n_q документов

(b) $\pi_i : d_i$ выше $d_j, z_i > z_j$
π' : как π_i только d_i ниже d_j
$p_z(\pi) > p_z(\pi')$

(c) Максимальную вероятность имеет перестановка с отсортированной по модели выборке

Можно посчитать $\frac{\partial p_z(\pi)}{\partial a}$

Задача: $E_\pi nDCG@k(q, \pi) \to \max_a$

$\sum_{\pi \in Sym(q_1, \ldots, q_{n_q})} p(\pi) nDCG@k(q, \pi) \to \max_a$

Очень много $(!n_q)$ слагаемых

В ListNet перестановки делаются иначе:

Вероятность, что j документ попадет на первое место в перестановке:

$p_z(j) = \frac{\phi(z_j)}{\sum_{i=1}^{n} \phi(x_i)}$

$Q(y, z) = -\sum_{j=1}^{n_q} p_y(j) log p_z(j) \to \min_a$

11 Рекомендательные системы

Задача, где любые две сущности надо сопоставлять друг с другом
Определения:
Множество пользователей:
$U = \{u_1, \dots, u_n\}$
Множество айтемов:
$I = \{i_1, \dots, i_m\}$
Для некоторых (u, i) $\exists r_{ui}$
$R = \{(u, i) \mid \exists r_{ui}\}$
Нужно найти $a(u, i)$
$U \xrightarrow{\text{Запрос}}$ Отбор кандидатов $\rightarrow$ Ранжирование $\rightarrow$ Переранжирование с учетом бизнес-требований $\rightarrow$ выдача

11.1 Методы

1. Коллаборационная фильтрация:

   Используем только информацию о взаимодействиях

   $R = \begin{pmatrix} r_{u_1 i_1} \dots r_{un} \\ r_{u_2 i_1} \dots\dots \end{pmatrix}$

   (a) Memory-based методы

   Просто используем эвристики - находим похожих пользователей и рекомендуем то, что понравилось им

   (b) Модели со скрытыми переменными

   $p_u \in R^d$ - вектор пользователя

   $q_i \in R^d$ - вектор айтема

   Обучаем так, чтобы:

   $r_{ui} \approx < p_u, q_i >$

   $(\star) \sum_{u,i \in R} (r_{ui} - w_u - w_i - < p_u, q_i >) \rightarrow \min\limits_{\substack{p_u, q_i \\ w_u, w_i}}$

   Наблюдение 1:

   $R \in R^{n \times m}$

   $P = (p_1 \mid \dots \mid p_n) \in R^{d \times n}$

   $Q = (q_1 \mid \dots \mid q_n) \in R^{d \times m}$

   $(P^T Q)_{ui} = < p_u, q_i >$

   $\left\| R - P^T Q \right\|_F \rightarrow \min_{P,Q}$

   Если матрица R известна, то это задача низкорангового приближения R - SVD

PureSVD: Заполняем все пропуски нулями и применяем обычный SVD

Наблюдение 2:

Если знаем все q_i

Рассмотрим конкретного пользователя:

$\sum_{i:\exists r_{ui}}(r_{ui} - w_u - w_i - < p_u, q_i >)^2 \to \min_{p_u \in R^d}$

Задача сводится к линейной регрессииU

Где это может пригодится?

i. Обучили (*), получили P, Q - матрицу Q можно хранить на серверах
   Когда приходит u решаем задачу линейной регрессии, обучая p_u
   Не нужно хранить p
   Можно учесть самые свежие действия пользователя
ii. Можно фиксировать все q_i на основе контента айтема i

Обучение LFM:

i. SGD
   Задача невыпуклая - можно попасть в локальный минимум
ii. ALS (alternating least squares)
   Фиксируем P, находим Q $\to$ Фиксируем Q, находим P $\to$ ...

Наблюдение 3:

Если нашли идеальное решение $R = P^T Q$

$\|r_i\|$ - насколько всем пользователям нравится этот айтем

Даже если много единиц, то норма все равно будет большой $\to$ популярность айтема i

$r_i = P^T q_i$

$\sigma_{min}(P) \times \|q_i\| \geq \|r_i\| \leq \sigma_{max}(P) \times \|q_i\|$

Сингулярное разложение

Пусть дана матрица $F_{n\times m}$. Тогда F можно представить в следующем виде:

$$F_{n\times m} = U_{n\times n}\Sigma_{n\times m}V^T_{m\times m}$$

Основные свойства сингулярного разложения:

i. $n \times n$-матрица $U = (u_1, \ldots, u_n)$ ортогональна, $U^T U = I_n$, столбцы u_j — собственные векторы матрицы FF^T

ii. $m \times m$-матрица $V = (v_1, \ldots, v_n)$ ортогональна, $V^T V = I_m$, столбцы v_j — собственные векторы матрицы $F^T F$

iii. $n \times m$-матрица $\Sigma_{n \times m}$ диагональная, $\Sigma_{n \times m} = diag(\sqrt{\lambda_1}, \ldots, \sqrt{\lambda_n})$, $\lambda_j \geq 0$ — собственные значения матриц $F^T F$ и FF^T $\sqrt{\lambda_i}$ - сингулярные числа

Пусть i популярен - $\|q_i\| \gg 0 \rightarrow$ для многих пользователей $< p_u, q_i > \gg 0$

Хак: Заменяем a(x) $= < p_u, q_i >$ на $a(x) = \frac{<p_u, q_i>}{\|p_u\| \|q_i\|}$

Наблюдение 3': i: 5 показов, 4 успешных $\rightarrow$ высокий CTR $\rightarrow$ норма становится высокой, но большой CTR может быть случайностью на низком количестве показов

Надо регуляризировать с учетом норм q_i, p_u

Наблюдение 4:

LFM: $(\mid U \mid + \mid I \mid)(d + 1)$ параметров

Увеличиваем количество параметров с помощью Neural CF:

i. Берем эмбеддинги p_u, q_i

ii. Конкатенируем эмбедингы в один вектор

iii. Наворачиваем сверху полносвязные слои

iv. В конце они должны предсказывать r_{ui}
Для задачи лучше использовать не полносвязные слои, чтобы не застревать в локальных минимумах

Factorization Machines - обобщение LFM

11.2 Работа с неявным фидбэком

Явный фидбэк: пользователь непосредственно поставил оценку
Неявный фидбэк: факт покупки, факт просмотра, ...

iALS

$S_{ui} = \begin{cases} 1, \exists r_{ui} \\ 0, \nexists r_{ui} \end{cases}$

$C_{ui} = 1 + [\exists r_{ui}] \alpha r_{ui}$

α может принимать различные значения в зависимости от градации позитивности неявного фидбэка

Модель:

$$\sum_{\substack{u\in U\\ i\in I}} C_{ui}(S_{ui} - w_u - w_i - < p_u, q_i >)^2 + \lambda \dots$$

11.3 Контентные рекомендации

Подходы

1. $q_i = f(i)$ - считаем q_i по контенту
   Обучаем (a_i)
2. DSSM, Siameese Nets
   $i \to$ контент $\to$ превращаем в вектор q_i
   $u \to$ история $(i_1, i_2, \dots) \to$ превращаем в вектор p_u
   Требуем, чтобы $corr(\rho(p_u, q_i), r_{ui}) \uparrow$
   Для обучения используем триплетную функцию потерь

11.4 Как это все используется

1. Отбор кандидатов
   (a) Эвристики
   (b) Легкая модель
   (c) Поиск ближайших соседей
   (d) Графовые методы
      i. Двудольный граф с пользователями и айтемами
      ii. Запускаем случайное блуждание
2. Ранжирование $(u, i) \to$ признаки

12 AutoML

12.1 Что такое AutoML

1. Автоматизация некоторого этапа ML
2. Система, которая способна полностью решать бизнес задача

Уровни AutoML

1. Сами алгоритмы
2. API к алгоритмам
3. Автоматическая оптимизация гиперпараметров / подбор ансамблей
4. Автоматическая генерация признаков, аугментаций, отбор признаков, визуализация
   (a) Стратегии обучения + управление бюджетом
   (b) Простое Meta обучение
5. Автоматическое определение домена, объединение табличек без знания структуры базы данных, спецификация под задачу
6. Полная оптимизация, работает лучше, чем люди

Перспективные направления

1. Продвинутое Meta learning
2. Domain Specific Language
3. Базы знаний

Бывают проприетарные и открытые AutoML, так же есть исследовательские и индстриальные

12.2 Зачем нужен AutoML?

1. ML выгоден, AutoML быстрый - не нужно таких затрат на работу
2. Автоматическое решение обходят только специалисты и для этого нужно время

12.3 Элементы AutoML

1. Данные
   (a) Нет предобработки
   (b) Разные источники и форматы
   (c) Структурированные и неструктурированные
2. Black Box

   (a) Препроцессинг
   (b) Генерация признаков
   (c) Выбор гиперпараметров
   (d) Обучение модели / ансамбля - оптимизация целевой метрики
3. Предсказания

Экспертная система:

1. K-раз прогоняем препроцессинг с учетом уже построенных ранее моделей
2. Генерация признаков + выбор гиперпараметров
3. Обучение модели

Нелинейная связь между элементами - результат каждого этапа может влиять на предыдущий

12.4 Существующие решения

1. AutoSklearn
   (a) Умеет работать в сжатые сроки
   (b) Оптимизируется байсовским оптимизатором
   (c) Модели на каждой итерации байесовского оптимизатора сохраняются
   (d) Для каждого датасета нашли оптимальный пайплайн и построили метамодель - использование 25 оптимальных кандидатов и постройка ансамбля для похожих датасетов
2. AutoSklearn 2.0
   (a) Увеличили размер метадатасета - разносторонние пайплайны
3. Oboe
   (a) На основе метамоделей позволяет эффективно строить модели
   (b) Признаки основаны на качестве модели, а не статистиках датасета
   (c) Восстанавливаем матрицу ошибок простыми моделям
4. TensorOboe

   (a) Вместо матрицы ошибок тензор ошибок

5. TPOT
   (a) Строят дерево и оптимизируют его генетическим алгоритмом
6. AutoGluon
   (a) Используют многоуровневые сети и LightGBM
   (b) Делают бэггинг и K-fold валидации
   (c) Дамп моделей занимает 200 гб

12.5 Анализ и выводы

Слабые пайплайны:

1. Простые / неэффективные модели
2. Наивный препроцессинг и генерация признаков

Мета-алгоритмы:

1. Маленькие наборы датасетов
2. Синтетические, игрушечные и странные датасеты
3. Слишком широкая сетка гиперпараметров
4. Вычислительно дорогая оптимизация параметров

12.6 Бэнчмарки

1. OpenML
2. AutoCV, AutoNLP, AutoTS, AutoSignal, ..., AutoDL

13 Рекомендательные системы 2

Подход:

1. Есть user, есть item
2. Прогоняем через сеть и получаем эмбеддинг для item
3. Для item, которые понравились пользователю строим эмбединги и превращаем их вектор user
4. Верхним слоем подгоняем выход из этих двух эмбеддингов к рейтингу

13.1 Холодный старт

Новый пользователь:

1. Показывать популярное
2. Факторы по гео/соц/дем
3. Модерируемые холодные подборки
4. Опросить пользователя о его интересах

Новый айтем:

1. Найти пользователей, которые смотрели похожие айтемы
2. Подписчикам канала
3. Гарантировать новому айтему некоторое количество рандомных показов

13.2 Метрики качества рекомендаций

1. Оффлайн
   (a) Исторические данные: $u_1 : i_1, i_2, \ldots, i_n$ $u_2 : \ldots$ $\vdots$ $u_m : \ldots$
   (b) Как разбивать на обучение и тест:
      i. Важно: разбивать по времени
      ii. Интересы и запросы могут сильно изменятся во времени - надо смотреть качество на онлайне
2. Онлайн

(a) A/B тестирование

(b) Бизнес-метрики

(c) Метрики про качество угадывания

(d) По пользовательским сигналам отбирать сомнительный контент

(e) Разнообразие - вероятности, которые мы предсказуем может быть скоррелирована по разным выданным айтемам

(f) Serendipity - успешные рекомендации редких или непохожих на историю пользователя айтемов

14 Нейросетевые методы для табличных данных

Inductive bias: нужно корректировать архитектуры с учетом специфики данных: картинки - Conv, текст - RNN
Бустинг:
Композиция неглубоких деревьев
Учитывает особенности данных:

1. Учитывает разные распределения столбцов
2. Моделирует зависимости ответов от небольших комбинаций признаков. Нейросеть же учит взаимосвязи между всеми признаками.
3. Можем контролировать переобучение

14.1 DeepFM

$FM : a_{FM}(x) = w_0 + < w, x > + \sum_{j_1=1}^{d} \sum_{j_2=j_1+1}^{d} < v_{j_1}, v_{j_2} > \times x_{j_1}, x_{j_2}$
Статья была написана для задачи предсказывания кликов на рекламу
Много разреженных категориальных признаков
Для каждого категориального признака строим векторное представление
Сверху этих векторов строим FC-слои и предсказываем ответ
$y \approx \sigma(a_{FM}(x) + a_{DNN}(x))$

14.2 AutoInt

Использование self-attention

Вход:

$$x_1, \dots, x_n$$
$$q_1, \dots, q_n$$
$$K_1, \dots, K_n$$
$$v_1, \dots, v_n$$

$W_{ij} = \frac{exp(\frac{<q_j, K_i>}{\sqrt{d}})}{\sum_{p=1}^{n} exp(\frac{<q_j, K_p>}{\sqrt{d}})}$
Вычисляет важность слова x_i для нового представления слова x_j
Выходы: $z_j = \sum_{p=1}^{n} w_{pj} v_p$
Как набор признаков превратить в набор векторов?

1. Если x_j - категориальный, то обучаем свой вектор для каждой категории
2. Если x_j - числовой, то $x_j \to x_j \times v_j$

На большой размерности векторов работает с трудом

14.3 NODE

Neural Oblivious Decision Ensembles
$\sum_{j_1=0}^{1} \sum_{j_n=0}^{1} C_{j_1,\dots,j_n} [x_{i_1} = j_1][x_{i_2} = j_2] \dots$
n - глубина дерева
$C_{j_1,\dots,j_n}$ - прогноз
$[x_{i_1} = j_1]$ - индикатор попадания в лист
$f_i(x) = \sum_{j=1}^{d} x_j entmax(F_{i1}, \dots, F_{id})$

$$z \in R^k \xrightarrow{entmax} z' \in R^k :$$
$$\sum t'_i = 1$$
$$z'_i \geq 0$$

$p_i(x) = \sigma_\alpha(\frac{f_i(x) - b_i}{\tau_i})$
$\sigma_\alpha(z) = entmax([z, 0])$
$\tilde{b}(x) = \sum_{j_1=0}^{1} \dots \sum_{j_n=0}^{1} C_{j_1,\dots,j_n} (p_1(x))_1^j (1-p_1(x))^{1-j_1} \times (p_n(x))^{j_n} (1-p_n(x))^{1-j_n}$

14.4 TabNet

Идея: self-supervision для табличных данных
Берем данные и пытаемся обучить модель восстанавливать пропуски

Part II
Семинары

15 Семинар: Задачи условной оптимизации

Учебник: Boyd, Convex Optimization

$$\begin{cases} f_0(x) \to min_{x \in R^d} \\ f_i(x) \leq 0, i = 1, \dots, m \\ h_i(x) = 0, i = 1, \dots, p \end{cases}$$

$$G(x) = f_0(x) + \sum_{i=1}^{m} I_-(f_i(x)) + \sum_{i=1}^{p} I_0(h_i(x)) \to min$$

Штрафы за нарушение ограничений:

$$I_-(z) = \begin{cases} 0, z \leq 0 \\ +\infty, z > 0 \end{cases}$$

$$I_0 = \begin{cases} 0, z = 0 \\ +\infty, z \neq 0 \end{cases}$$

$G(x) \to \infty$ в точках где не выполняется условие
Проблема: Недифференцируема
Заменяем функции на их аппроксимации ($\hat{I}_- = ax$)
Лагранжиан:

$$L(x, \lambda, \nu) = f_0(x) + \sum_{i=1}^{m} \lambda_i f_i(x) + \sum_{i=1}^{p} \nu_i h_i(x)$$

$$\lambda_i \geq 0$$

x - прямые (primal) переменные
λ, ν - двойственные переменные
Двойственная функция

$$g(\lambda, \nu) = \inf_x L(x, \lambda, \nu)$$

- Двойственная функция всегда вогнутая
- Дает нижнюю оценку на минимум функции в прямой задаче
  x' - допустимая точка (все условия выполнены)

$$L(x', \lambda, \nu) = f_0(x') + \sum_{i=1}^{m} \lambda_i f_i(x') + \sum_{i=1}^{p} \nu_i h_i(x')$$
$$f_i(x) \leq 0, h_i(x) = 0 \to L(x', \lambda, \nu) \leq f_0(x')$$
$$\inf_x L(x, \lambda, \nu) \leq \inf_{x'} L(x', \lambda, \nu) \leq \inf_{x'} f_0(x')$$

  $\uparrow$ - это и есть решение исходной задачи

$$g(\lambda, \nu) \leq f_0(x_\star)$$

$$g(\lambda, \nu) \to \max_{\lambda,\nu}, \lambda_i \geq 0$$

  $\lambda^\star, \nu^\star$ - решение двойственной задачи
  $g(\lambda^\star, \nu^\star) \leq f_0(x_*)$ - слабая двойственность
  $g(\lambda^\star, \nu^\star) = f_0(x_*)$ - сильная двойственность
  Достаточное условие сильной двойственности (Условие Слейтера)

  - Задача выпуклая:
    $f_0, f_1, \ldots, f_m$ - выпуклые
    $h_1, \ldots, h_p$ - линейные
  - $\exists x'$, что все ограничения выполнены строго

Пусть имеет место сильная двойственность:

$$g(\lambda^\star, \nu^\star) = f_0(x_*)$$

$$g(\lambda^\star, \nu^\star) = \inf_x (f_0(x) + \sum \lambda^\star f_i(x) + \sum \nu^\star h_i(x)) \leq f_0(x_\star) + \sum \lambda^\star f_i(x_\star) + \sum \nu^\star h_i(x_\star) \leq f_0(x_\star)$$

Все неравенства являются равенствами:

- Если решить безусловную задачу при подставлении $\lambda^\star, \nu^\star$, то получим решение прямой задачи
- $\lambda_i^\star f_i(x^\star) = 0$ - условие дополняющей нежесткости

Теорема Куна-Такера
Необходимые условия для

$$\begin{cases} \nabla_x L(x_\star, \lambda^\star, \nu^\star) = 0 \\ f_i(x) \leq 0 \\ h_i(x) = 0 \\ \lambda_i \geq 0 \\ \lambda_i f_i(x_\star) = 0 \\ \text{Сильная двойственность} \end{cases} \leftrightarrow x_\star, \lambda^\star, \nu^\star \text{решения}$$

16 Семинар 3: EM алгоритм

На M-шаге:

$$\Theta = \arg\max_\Theta E_q logp(X, Z \mid \Theta)$$

$$logp(X \mid \Theta_{i+1}) > logp(X \mid \Theta_i)$$

Задача: **Шумная разметка изображений 100 экспертами**
i - изображение, j - эксперт: $l_{ij} \in \{0, 1\}$
Истинный класс для картинки $Z_i \in \{0, 1\}$
Дополнительные параметры:

$$\beta_i \in (0, +\infty), \alpha_j \in \mathcal{R}$$

β - сложность изображения, α - уровень эксперта

$$p(l_{ij} = Z_i \mid Z_i, \alpha_j, \beta_i) = \sigma(\alpha_j \beta_i) = \frac{1}{1 + e^{-\alpha_j \beta_i}}$$

$$p(Z_i, l_i \mid \alpha, \beta) = p(Z_i) \prod_j p(l_i j \mid Z_i, \alpha_j, \beta_i)$$

$p(Z_i)$?

1. Задать как $1/2$, т.к. имеем два класса

2. Задать как баланс классов

3. Найти как параметр $p(1) = \pi$

$$p(Z, l \mid \alpha, \beta) = \prod_i Z_i \prod_j p(l_i j \mid Z_i, \alpha_j, \beta_i)$$

Необходимо свести вероятность $l_{ij} = Z_i$ к вероятности l_{ij}

$$p(l_{ij} = Z_i \mid \ldots) = \sigma(\alpha\beta)$$

$$p(l_{ij} \neq Z_i \mid \ldots) = 1 - \sigma(\alpha\beta)$$

Бернулли:

$$p(l \mid \ldots) = p(l = Z \mid \ldots)^{[l=Z]} \times p(l \neq Z \mid \ldots)^{[l \neq Z]} = \sigma(\alpha\beta)^{[l=Z]}\sigma(-\alpha\beta)^{[l \neq Z]}$$

$$p(Z_i, l_{ij} \mid \ldots) = p(Z_i) \prod_j \sigma(\alpha\beta)^{[l=Z]}\sigma(-\alpha\beta)^{[l \neq Z]}$$

E-шаг:

$$q^\star(Z_i) = p(Z_i \mid l_{ij}, \alpha_j, \beta_i) \xrightarrow{\text{Теорема Байеса}} \frac{p(Z_i, l_{ij} \mid \alpha, \beta)}{p(l_{ij} \mid \alpha, \beta)} = \frac{p(Z_i, l_{ij} \mid \alpha, \beta)}{\sum_t p(t, l_{ij} \mid \alpha, \beta)}$$

$$q^\star(Z) = \frac{p(Z_i) \prod_j \sigma(\alpha\beta)^{[l=Z]}\sigma(-\alpha\beta)^{[l \neq Z]}}{\sum_{t \in \{0,1\}} p(t_i) \prod_j \sigma(\alpha\beta)^{[l=t]}\sigma(-\alpha\beta)^{[l \neq t]}} = \frac{\gamma_i^{Z_i}}{\gamma_i^0 + \gamma_i^1} = \frac{e^{log\gamma_i^{Z_i}}}{e^{log\gamma_i^0 + \gamma_i^1}}$$

M-шаг:

$$E_{q^\star} logp(Z, l \mid \alpha, \beta) \to \max_{\alpha,\beta}$$

$$E_{q^\star} log \prod_i p(Z, l \mid \alpha, \beta) = \sum_i E_{q_i^\star} logp(Z, l \mid \alpha, \beta) =$$

$$= \sum_i E_{q_i^\star}[logp(Z_i) + \sum_j [l_{ij} = Z_i] log\sigma(\alpha\beta) + [l_{ij} \neq Z_i] log\sigma(-\alpha\beta)] \to \max_{\alpha,\beta}$$

$$\sum_i \sum_j \sum_{t \in \{0,1\}} q_i^\star(t)[[l_{ij} = t] log\sigma(\alpha\beta) + [l_{ij} \neq t] log\sigma(-\alpha\beta)]$$

Оптимизируем:

$$\frac{\partial}{\partial x} log\sigma(x) = \sigma(-x)$$

$$\frac{\partial}{\partial \alpha} log\sigma(\alpha\beta) = \beta\sigma(-\alpha\beta)$$

$$\frac{\partial}{\partial \alpha} \log \sigma(-\alpha\beta) = -\beta\sigma(\alpha\beta)$$

$$\frac{\partial}{\partial \alpha} E_{q^\star} logp(Z, l \mid \alpha, \beta) = \sum_i \sum_t q_i^\star \beta([l = t]\sigma(-\alpha\beta)) - [l \neq t]\sigma(\alpha\beta))$$

По β аналогично

17 Семинар 4: Основы байсовских методов

Существует распределение $p(x, y)$
Интересует распределение: $p(y \mid x)$
Формула Байеса

$$p(y \mid x) = \frac{p(x \mid y)p(y)}{p(x)}$$

$p(x \mid y)$ - правдоподобие X, распределение объектов для некоторого класса
$p(y)$ - априорное распределение, доли классов в обучающей выборке
$p(x)$ - нормировочная константа
Функционал среднего риска

$$R(a) = \int_Y \int_X L(y, a(x))p(x, y)dxdy$$

$$E_{y,x} L(y, a(x))$$

Как использовать оптимальное распределение, когда оно найдено?

$$L(y, a) = [y \neq a]$$

Функционал среднего риска:

$$R(a) = \int_Y \int_X [y \neq a(x)]p(x,y)dxdy = \sum_{y=1}^{K} \int_X [y \neq a(x)]p(x,y)dxdy =$$

$$= \int_X \sum_{y \neq a(x)} p(x,y)dxdy = \int_X (1 - \sum_{y=a(x)} p(x,y))dxdy =$$

$$1 - \int_X p(x, a(x))dxdy \to \min \Rightarrow a_\star(x) = \arg\max_{y \in Y} P(y \mid x)$$

Для регрессии:

$$L(y,a) = (y-a)^2$$
$$a_\star(x) = E(y \mid x)$$

Как найти $p(y \mid x)$
В классификации:

$$a_\star(x) = \arg\max_{y \in Y} p(y \mid x) = \arg\max_{y \in Y} \frac{p(x \mid y)p(y)}{p(x)} =$$
$$= \arg\max_{y \in Y} p(x \mid y)p(y)$$

$p(y)$ задается исходя из распределения y
$p(x \mid y, \theta)$ находим θ ММП

Пример:

$$p(y \mid x, w) = \mathcal{N}(< w, x >, \sigma^2)$$

Правдоподобие:

$$\prod_{i=1}^{\ell} \frac{1}{\sqrt{2\pi\sigma^2}} exp(-\frac{(y_i - < w, x_i >)^2}{2\sigma^2}) \to \max_w$$

$$logL = -\ell log\sqrt{2\pi\sigma^2} - \frac{1}{2\sigma^2} \sum_{i=1}^{l} (y_i - < w, x_i >)^2 \to \max_w \Rightarrow$$

$$\Rightarrow \sum_{i=1}^{l} (y_i - < w, x_i >)^2 \to \min_w$$

Классификация:
Нужно найти $p(x \mid y, \theta)$ для всех классов

$$p(x \mid y, \theta) = \mathcal{N}(\mu_y, \Sigma_y)$$

Можем найти μ_y, Σ_y по ММП
Если параметры распределены нормально - Нормальный дискриминантный анализ
Если $\Sigma_y = \Sigma$, метод называется линейный дискриминант Фишера
Разделяющая поверхность:

$$p(y = +1 \mid x, \theta) = p(y = -1 \mid x, \theta)$$

$\Sigma_{-1} \neq \Sigma_{+1} \Rightarrow$ квадратичная поверхность
$\Sigma_{-1} = \Sigma_{+1} \Rightarrow$ Линейная поверхность

Больше распределений:

$$p(w \mid y, x) = \frac{p(y \mid x, w)p(w)}{p(x, y)}$$

$p(w) \sim \mathcal{N}(0, \sigma^2 I)$ - запрещаем модели большие веса

$$logP(w \mid y, x) = -\frac{1}{2\sigma_2}\sum_{i=1}^{\ell}(y_i - < w, x_i >)^2 - \frac{\ell}{2\alpha^2}\sum_{j=1}^{\alpha} w_j^2 \to \max_w$$

Фактически: MSE с регуляризацией L^2

$$\lambda = \frac{\ell\sigma^2}{\alpha^2}$$

Что если $w_j \sim \mathcal{N}(0, \alpha_j^2)$?
Отдельный коэффициент регуляризации для каждого параметра - такое не особо выводится в классическом машинном обучении $\to$ RVM

Наивный Байесовский алгоритм
Исходя из предположения о независимости признаков:

$$p(x \mid y) = \prod_{j=1}^{d} p(x_j \mid y)$$

$$a(x) = \arg\max_{y \in Y} p(y \mid x) = \arg\max_y (lnP(y) + \sum_{j=1}^{d} lnP(x_j \mid y))$$

18 Семинар 5: Спектральная кластеризация

Алгоритм:

1. $L = D - W, D = diag(d_1, \ldots, d_\ell), d_i = \sum_{j=1}^{\ell} w_{ij}$
2. $u_1, \ldots, u_m$ - собственные векторы, соотв. минимальным собственным значениям L
3. $U = (u_1 \mid \ldots \mid u_m) \in R^{l \times m}$
4. K-means над U

Почему не делать кластеризацию t-SNE или Umap?

1. Оптимизирует положение точек, а не расположение кластеров
2. В t-SNE ошибка может быть неограничено большой $\rightarrow$ зашумленные представления объектов
3. Есть шанс, что PSA будет лучше, чем t-SNE

Задача кластеризации:

$$W(A, B) = \sum_{i \in A, j \in B} w_{ij}$$

$$A, B \subset X$$

$$A \cap B = \emptyset$$

$$\bar{A} = X \setminus A$$

$$\text{Ratio Cut}(A_1, \ldots, A_k) = \frac{1}{2} \sum_{i=1}^{K} \frac{W(A_i, \bar{A}_i)}{\mid A_i \mid} \rightarrow \min_{A_1, \ldots, A_K}$$

K = 2:

$$\text{Ratio Cut}(A, \bar{A}) \rightarrow \min_{A \subset X}$$

Задача поиска минимального разреза

$$X = A \cup \bar{A}$$

$$f : f_i = \begin{cases} \sqrt{\frac{|\bar{A}|}{|A|}}, x_i \in A \\ -\sqrt{\frac{|A|}{|\bar{A}|}}, x_i \in \bar{A} \end{cases}$$

Квадратичная форма:

$$f^T L f = \frac{1}{2}\sum_{i,j=1}^{\ell} w_{ij}(f_i - f_j)^2 =$$

$$= \frac{1}{2}\left(\sum_{x_i\in A, x_j\in \bar{A}} w_{ij}\sqrt{\frac{|\ \bar{A}\ |}{|\ A\ |}} + \sqrt{\frac{|\ A\ |}{|\ \bar{A}\ |}}\right)^2 + \frac{1}{2}\left(\sum_{x_i\in \bar{A}, x_j\in A} w_{ij}\sqrt{-\frac{|\ A\ |}{|\ \bar{A}\ |}} + \sqrt{-\frac{|\ \bar{A}\ |}{|\ A\ |}}\right)^2$$

$$= \frac{1}{2}\left(\sqrt{\frac{|\ \bar{A}\ |}{|\ A\ |}} + \sqrt{\frac{|\ A\ |}{|\ \bar{A}\ |}}\right)^2 (W(A,\bar{A}) + W(\bar{A}, A)) \Rightarrow$$

$$\Rightarrow \left(\sqrt{\frac{|\ \bar{A}\ |}{|\ A\ |}} + \sqrt{\frac{|\ A\ |}{|\ \bar{A}\ |}}\right)^2 = \left(\frac{|\ \bar{A}\ | + |\ A\ |}{|\ A\ |} + \frac{|\ A\ | + |\ \bar{A}\ |}{|\ \bar{A}\ |}\right) = \ell(\frac{1}{|\ A\ |} + \frac{1}{|\ \bar{A}\ |});$$

$$(W(A,\bar{A}) + W(\bar{A}, A)) = 2W(A,\bar{A}) \Rightarrow$$

$$f^T L f = \ell(\frac{1}{|\ A\ |} + \frac{1}{|\ \bar{A}\ |})W(A,\bar{A}) = 2\ell\text{Ratio Cut}(A,\bar{A}) \propto \text{Ratio Cut}(A,\bar{A})$$

$$\sum_{i=1}^{\ell} f_i = |\ A\ |\sqrt{\frac{|\ \bar{A}\ |}{|\ A\ |}} - |\ \bar{A}\ |\sqrt{\frac{|\ A\ |}{|\ \bar{A}\ |}} = \sqrt{|\ A\ ||\ \bar{A}\ |} - \sqrt{|\ A\ ||\ \bar{A}\ |} = 0$$

$$\sum_{i=1}^{\ell} f_i^2 = |\ A\ |\frac{|\ \bar{A}\ |}{|\ A\ |} + |\ \bar{A}\ |\frac{|\ A\ |}{|\ \bar{A}\ |} = \ell$$

Переписываем оптимизационную задачу:

$$\begin{cases} f^T L f \to \min_{f_i\in\{\ldots\}} \\ < f, \vec{1} >= 0 \\ \|f\|^2 = \ell \end{cases}$$

Релаксация: $f_i \in R$

$$\begin{cases} f^T L f \to \min_{f\in R^\ell} \\ < f, \vec{1} >= 0 \\ \|f\|^2 = \ell \end{cases}$$

Лагранжиан:

$$\mathcal{L} = f^T L f + \lambda_1 < f, \vec{1} > + \lambda_2(\|f\|_2 - \sqrt{\ell})$$

$$\nabla_f \mathcal{L} = 2L_f + \lambda_1 \vec{1} + \lambda_2 \frac{1}{\|f\|} f = 0 \mid \times \vec{1}^T$$

$$2\vec{1}^T L f + \lambda_1 \ell + \lambda_2 \frac{1}{\|f\|} < \vec{1}, f >= 0 \Rightarrow$$

$$L1 = 0 \Rightarrow f^T L 1 = 0 \Rightarrow \lambda_1 \ell = 0$$

$$\lambda_1 = 0$$

$$2Lf + \lambda_2 \frac{1}{\|f\|} f = 0$$

$$Lf = -\frac{\lambda_2}{2\|f\|} f \Rightarrow$$

f - собственный вектор L, соотв. с.з.μ

$$f^T L f = \mu f^T f = \|f\|_2 \mu = \ell \mu$$

Новая задача: $\begin{cases} \mu \to \min_{\mu \text{ - с.з L}, f\text{- с.в.}} \\ < f, \vec{1} >= 0 \\ \|f\| = \sqrt{\ell} \end{cases}$

Если G-связный, то с.в. соотв. 0 с.з. не подходит из-за невыполнения первого ограничения, в неполном графе может подходить
Решение - это с.в., соотв. второму собств. знач.
Находим f $\to$ запускам K-means

19 Семинар 6: Отбор признаков

19.1 Deep Clustering

1. Прогоняем объекты через нейросеть
2. По векторным описаниям строим псевдоразметку с помощью KMeans
3. Обучаем на псевдоразметке
4. Повторяем каждую эпоху

Даже необученная сеть не супер плохо размечает
Проблемы:

1. Несбалансированные выборки - использование взвешенных лоссов

2. Наличие пустых кластеров - берем случайный центр другого кластера и добавляем шум
3. Делаем PCA перед кластеризацией, L2 нормализацию
4. Сбрасываем линейный слой на каждой эпохе

Есть возможности для улучшений:

1. Использовать не PCA, а MLP
2. Инициализируем матрицу классификатора в виде центроидов

19.2 Positional Encoding

Более простая задача:

1. Подаем сети координату точки
2. Она восстанавливает цвет пикселя с координатами
3. Можно воссоздавать 3D сцены (NERF)

19.3 Спектральный анализ

Берем картинку и парсим ее на частоты с помощью преобразования Фурье
При высоких частотах - быстрые изменения цвета
Обычный персептрон не умеет передавать высокие частоты

19.4 Positional encoding

Подаем не только саму картинку, но и гармоники - сеть сможет извлекать высокие частоты и обучаться на них

19.5 Feature extraction

Методы:

1. Filter
   (a) Relevancy - удаляем близкие фичи
   (b) Redundancy - используем Mutual Info classifier
   (c) MRMR classifier
2. Wrapper - переучиваем модель на подмножествах фичей

20 Работа с признаками

1. Придумывание признаков
2. Feature selection
3. Понижение размерности

20.1 Отбор признаков

1. Методы фильтрации
   (a) Корреляция x_j с y - не учитывает нелинейность и попарную корреляцию
   (b) Для корреляции: t-score

$$R_j = \frac{| \mu_{-1} - \mu_{+1} |}{\sqrt{\frac{\sigma_{-1}^2}{n_{-1}} + \frac{\sigma_{+1}^2}{n_{+1}}}}$$

   (c) Для многоклассовой f-score
2. Методы обертки
   (a) Ищем подмножество признаков, при котором ошибка модели на валидации поменьше
   (b) Жадное удаление / добавление
   (c) Генетические алгоритмы
   $\beta \in \{0,1\}^\alpha$ - вхождение признака в подмножество признаков
   Итерация:
      i. Популяция: $B = \{\beta_1, \ldots\}$
      ii. Скрещивание: $\beta_j = \beta' \times \beta''$

$$\beta_j = \begin{cases} \beta', p = \frac{1}{2} \\ \beta'', p = \frac{1}{2} \end{cases}$$

      iii. Мутация: $\sim \beta' \to \beta_j = \begin{cases} \beta'_j, p \\ 1 - \beta'_j, 1 - p \end{cases}$
      iv. Новая популяция: $B' = \{\sim \beta' \times \beta''\}$ для какого-то числа пар из B
      v. Делаем селекцию: Оставляем n лучших организмов
   (d) Отбор признаков на основе моделей: Лассо, Out of bag

20.2 Понижение размерности

Метод главных компонент (PCA)

$X \in R^{1\times D}$

$u_1, \ldots, u_D \in R^D$ - главные компоненты, если

(1) :$< u_i, u_j >= 0, \forall i \neq j$

(2) : $\|u_j\|^2 = 1$

(3) : При проецировании выборки X на $u_1, \ldots, u_d : var \to \max$

Поиск первой компоненты:

$$\begin{cases} u_1^T X^T X u_1 \to \max \\ \|u_1\|^2 = 1 \end{cases}$$

Лагранжиан:

$$2X^T X u_1 + 2\lambda u_1 = 0 \Rightarrow \lambda \to \max$$

u_1 - собств. вектор $X^T X$ соотв. наибольшему с.з.
Постановка 2:

$X \in R^{\ell\times D}$

$Z \in R^{\ell\times d}$

$U \in R^{d\times D}$

Задача:

$\left\|X - ZU^T\right\|_F^2 \to min$

Решается с помощью сингулярного разложения

21 Метод k ближайших соседей

$X = \{(x_i, y_i)\}_{i=1}^{\ell}$

$\rho : X \times X \to (0, +\infty)$

$U : \rho(u, x_1) \leq \ldots \leq \rho(u_\ell)$

$a(u) = \arg\max_{y\in Y} \sum_{i=1}^{K} w_i[y_i = y]$

Особенности метода:

1. Шумовые признаки

   Очень чувствителен к шумовым признакам, т.к. использует все признаки для подсчета расстояния

2. Проклятие размерности

   Все объекты находятся по краям гиперкуба - трудно быстро искать близких соседей

3. Функции расстояния

   (a) Метрика Минковского: $\rho_p(x,z) = (\sum \mid x_j - z_j \mid^p)^{\frac{1}{p}}$
   $\rho_\infty(x,z) = max \mid x_j - z_j \mid$
   $\rho_0(x,z) = \sum_{j=1}^{d}[x_j \neq z_j]$
   Можно добавить веса для отдельных признаков:
   Веса можно подбирать покоординатным спуском

   (b) Расстояние Махаланобиса
   $\rho(x,z) = \sqrt{(x-z)^T S^{-1}(x-z)}$
   S - симметричная, положительно определенная матрица

   (c) Косинусная мера
   $\rho_{cos}(x,z) = arccos(\frac{<x,z>}{\|x\|\|z\|})$

22 Метрические методы 2

22.1 Расстояния на категориальных признаках

Один категориальный признак
Как измерить расстояния:

1. $\rho(x,z) = [x \neq z]$
2. $\rho(x,z) = \alpha[x \neq z] + \beta[x = z]$
3. Сделать α, β зависимыми от признака
4. $\rho(x,z) = [x \neq z] log(f(x)+1) log(f(z)+1)$
   f(x) - сколько раз в обучающей выборке встречается категория x
5. $\rho(x,z) = [x \neq z] + [x = z] \times \sum_{q:p(q)\leq p(x)} p_j^2(q)$
   $p(x)$ - частота
   $p_j^2(x)$ - вероятность, что у пары объектов категория x

22.2 Обучение метрик

Зачем:

1. Подобрать метрику для улучшения kNN
2. Когда необходимо разносить разные объекты по дальности

Самая параметризованная метрика: Метрика Махаланобиса
$\rho(x, z) = \|Ax - Az\|^2 = (x - z)^T A^T A(x - z)$
Выучиваем матрицу $A \in R^{n \times d}$
Методы обучения:

1. NCA - neighborhood component analysis

   $x_i \to$ выбираем x_j из некоторого распределения $\to$ относим x_i к y_j

   $$p_{ij} = \begin{cases} \frac{exp(-\|Ax_i - Ax_j\|^2)}{\sum_{i \neq j} exp(-\|Ax_i - Ax_j\|^2)}, i \neq j \\ 0, i = j \end{cases}$$

   Вероятность отнесения к правильному классу:

   $C_i = \{j \mid y_j = y_i\}$

   $p_i = \sum_{j \in C_i} p_{ij}$

   $Q(A) = \sum_{i=1}^{\ell} p_i \to \max_A$

2. LMNN - Large Margin NN

   Используем триплетный лосс:

   Берем для x_i положительные объекты и отрицательные объекты

   $\eta_{ij} \in \{0, 1\}$ - является ли x_j целевым объектом для x_i (входит в k соседей)

   Целевые объекты должны быть близки:

   $$\sum_{i \neq j} \xi_{ij} \|Ax_i - Ax_j\|^2 + C \sum_{i=1}^{\ell} \sum_{j \neq i} \sum_{\substack{m \neq i \\ m \neq j}} \xi_{ij}[y_m \neq y_i] \times$$
   $$\times \max(0, \alpha + \|Ax_i - Ax_j\|^2 - \|Ax_i - Ax_m\|^2) \to \min_A$$

3. ITML

   $p(x \mid A) = \frac{1}{z} exp(-\frac{1}{2} \|Ax - A\mu\|^2)$

   z - нормировочная константа

   μ - центр распределения

   S - мн-во пар, которые похожи

   D - мн-во пар, которые не похожи

   A_0 - априорная матрица для расстояния Махаланобиса:

   Можно ввеси на основе выборочной ков. матрицы

   Можно ввести как диагональную

$$\begin{cases} KL(p(x \mid A_0) \mid\mid p(x \mid A)) \to \min_A \\ \rho_A(x_i, x_j) \leq u, (i,j) \in S \\ \rho_A(x_i, x_j) \geq L, (i,j) \in D \end{cases}$$

4. MCML (Maximally collapsing metric learning)

   $p_A(j \mid i) = \frac{exp(-\|Ax_i - Ax_j\|^2)}{\sum_{k \neq i} exp(-\|Ax_i - Ax_k\|^2)}$

   $p_0(j \mid i) \propto \begin{cases} 1, y_i = y_i \\ 0, y_j \neq y_i \end{cases}$

$$\sum_{i=1}^{\ell} KL(p_0(\dot{\mid} i) \mid\mid p_A(\dot{\mid} i)) \to \min_A$$

5. Ядровой переход

   $K, \phi : X \to H$

   $L : H \to R^n$

   $\rho(x, z) = \|L\phi(x) - L\phi(z)\|^2$

   Ищем L:

   Из функционального анализа:

   $L(h) = (< h, w_1 >, \ldots, < h, w_n >), w_1, \ldots, w_n \in H$

   $w_i = \sum_{j=1}^{\ell} \alpha_{ij} \phi(x_j)$

23 Multilabel classification

Задача: Объект относится сразу к нескольким классам
Можно решать задачу бинарной классификации, отдельно для каждой метки, но тогда мы теряем информацию, которую несет принадлежность к предыдущим меткам.
Используем определение условной вероятности

$$p(y_1, y_2, \dots, y_D \mid x) = \prod_{i=1}^{D} p(y_i \mid x, y_1, y_2, \dots)$$

В таком подходе не можем получить предсказания отдельно по меткам:

1. Жадный подход - последовательно применяем $\arg\max_{y_i}$, получаем быстрое, но далекое от оптимального решение

2. Оптимальный, но долгий подход - расчет всех вероятностей $p(y_1, y_2, \dots, y_D \mid x)$ с последующим применением arg max к совместному распределению. Возможно использовать только При маленьком количестве классов из-за экспоненциальной сложности

3. Beam Search - ищем наиболее вероятный путь с помощью использования нескольких наиболее вероятных объектов

4. Монте-Карло - сэмлпируем по какой ветви пойти дальше

23.1 Label powerset

Представляем множество меток, как двоичное число и распределяем на основе этого их по классам.
Может быть проблематично обучать из-за того, что близкие объекты разносятся по разным классам
Для борьбы с большой размерность меток используем метод RAkEL:
Для каждого подмножества меток решается отдельная задача многоклассовой классификации, отбор меток производится голосованием классификатором
В реальных задачах можно провести кластеризацию на k кластеров, производить первичную классификацию на кластеры, а потом внутри кластеров классифицировать на изначальные категории

23.2 Отображение в пространство более низкой размерности

Используем сингулярное разложение матрицы меток
$Y = U\Sigma V^T$
Берем первые m столбцов, соответствующих наибольшим собственным числам матрицы $U - \hat{U}$
$\hat{U}$ - наилучшая проекционная матрица в пространство m
Перейдем в новое пространство: $\hat{Y} = \hat{U}^T Y$
Предсказания отображаем обратно, умножая слева на $\hat{U}$

23.3 Использование известных методов

1. Метод k ближайших соседей. $p(y_i = 1 \mid x) = \frac{1}{k} \sum_{j \in N_k} y_i^(j)$
2. Нейронные сети
3. Решающие деревья - модифицировать, чтобы критерий информативности соответствовал multi-label задаче и в листьях делалось предсказание для каждой из меток

23.4 Подбор порога бинаризации

1. Общий порог 0.5 для всех меток
2. Подбор порога таким образом, чтобы доля меток на трейне и тесте совпадали
3. Подбор порога для каждой метки с учетом некоторой функции потерь

23.5 Focal Loss

Модифицируем CE loss таким образом, что "легкие" для модели объекты не перевешивают "трудные"

$$FL(p_t) = -(1 - p_t)^\gamma log p_t, \gamma \in [0, 5]$$

24 Попарные методы ранжирования

Зачастую для ранжирования объекты представлены взаимодействиями между объектами.

Для некоторого набора объектов $\{x_i\}_{i=1}^n$, обучающая выборка представлена случайным подмножеством случайных соотношений $\{x_{i_k} > x_{j_k}\}_{k=1}^\ell$. Может быть известен контекст для соотношений.
Рассматриваем модели для попарных соотношений в контексте матча между двумя игроками x_i, x_j

1. Признаки объектов - для каждого игрока известно признаковое описание $x_i \in R^d$
2. Признаки контекста - признаковое описание матча $z_k \in R^D$
3. Набор соотношений - история матча в виде $\{(x_i, x_j, z_k)\}$, где на первом месте стоит победитель

24.1 Rote learning

Модель зазубривания: самая простая, но требует много памяти
Запоминаем матрицу вероятностей для каждой пары игроков (a, b)

$$P(x_i \text{ победит } x_j) = \frac{\sum_{k=1}^{\ell}[x_{i_k} = x_i][x_{j_k} = x_j] + 1}{\sum_{k=1}^{\ell}[x_{i_k} = x_i][x_{j_k} = x_j] + [x_{j_k} = x_i][x_{i_k} = x_j] + 2}$$

Недостатки:

1. Нужно много данных
2. Не учитывает характеристики игроков
3. Не может строить прогнозы для игроков которые раньше не встречались

24.2 Модель Бредли-Терри

Сопоставляем каждому игроку его силу γ_i
Тогда:

$$P(x_i \text{ победит } x_j) = \frac{exp(\gamma_i)}{exp(\gamma_i) + exp(\gamma_j)} = \frac{1}{1 + exp(\gamma_i - \gamma_j)} = \sigma(M(x_i, x_j))$$

$M(x_i, x_j) = \gamma_i - \gamma_j$ - функция матча
Функцию матча можно задать произвольным образом, но должны выполнятся условия:

1. $M(x_i, x_j) \in R$
2. $M(x_i, x_j) \to +\infty : P(x_i > x_j) \to 1$
3. $M(x_i, x_j) = -M(x_j, x_i)$

Плюсы:

1. Обладает небольшим числом параметров
2. Дает оценки на исход матча между игроками, которые не встречались раньше

Минусы:

1. Не использует информацию об игроках

24.3 Pairwise logistic regression model

$$\gamma_i = w^T x_i$$

Не учитывает контекст матча

24.4 Blade-chest model

Обучаем для каждого игрока два вектора x^{blade}, x^{chest}
Чем ближе x_i^{blade} к x_j^{chest}, тем более вероятно выиграет x_i

$$M(x_i, x_j) = \left\|x_j^{blade} - x_i^{chest}\right\|^2 - \left\|x_i^{blade} - x_j^{chest}\right\|^2$$

Также, можно записать через скалярное произведение
Кроме того, помимо линейной комбинации признаков игроков, можно также использовать любую дифференцируемую функцию для оценки этих векторов.

$$\begin{aligned} x_i^{blade} &= f_{blade}(Bx_i) \\ x_j^{blade} &= f_{blade}(Bx_j) \\ x_i^{chest} &= f_{chest}(Cx_i) \\ x_j^{chest} &= f_{chest}(Cx_j) \end{aligned}$$

Можно добавить z_k двумя способами:

1. Добавление признаков игры к описанию каждого игрока

$$x_i^{blade} = f_{blade}\left(B\begin{bmatrix} x_i \\ z_k \end{bmatrix}\right)$$

2. Поэлементное домножение векторов представлений на векторы представлений матча

$$x_i^{blade} = f_{blade}(Bx_i) \times f_{match}(B'z_k)$$

25 Factorization Machines

25.1 Вывод ALS и HALS

Задача:

$$\sum_{u,i}(r_{ui} - < p_u, q_i >)^2 \to \min_{P,Q}$$

25.1.1 ALS

Фиксируем одну из матриц:

$$\nabla_{p_u}\left[\sum_{u,i}(r_{ui} - < p_u, q_i >)^2\right] = \sum_i 2(r_{ui} - < p_u, q_i >)^2 q_i = 0$$

$$\sum_i r_{ui}q_i - \sum_i q_i q_i^T p_u = 0$$

Тогда:

$$p_u = \left(\sum_i q_i q_i^T\right)^{-1} \sum_i r_{ui}q_i \forall u$$

$$q_i = \left(\sum_u p_u p_u^T\right)^{-1} \sum_u r_{ui}p_u \forall i$$

25.1.2 HALS

На каждой итерации решаем задачу оптимизации относительно одной строчки матриц P и Q

$$\frac{\partial}{\partial P_{ku}}\left[\sum_{u',i}(r_{u'i} - < p_{u'}, q_i >)^2\right] = \sum_i 2(r_{ui} - < p_{u'}, q_i >)Q_{ki} = 0$$

$$\sum_i (r_{ui} - \sum_{s\neq k} P_{su}Q_{si})Q_{ki} - P_{ku}\sum_i Q_{ki}^2 = 0$$

$$P_{ku} = \frac{\sum_i (r_{ui} - \sum_{s\neq k} P_{su}Q_{si})Q_{ki}}{\sum_i Q_{ki}^2}$$

25.2 Neural Colaborative Filtering

Сопоставляем каждому пользователю u one-hot вектор α_u, у которого на u-ом месте стоит 1, а остальные координаты заполнены нулями, аналогично определим вектор β_1 для товара 1
Испольуем нейросеть с двумя полносвязными слоями на входе - один для пользователя, второй для товара.
После входных слоев конкатенируем полученные векторы и передаем в следующие полносвязные слои
На выходе получаем $\hat{r}_{ui}$, которое сравниваем с r_{ui}
Достоинства модели:

1. Можем легко обобщить модель на случай, когда есть признаки пользователя и товаров, добавив их в a_u и β_i соответственно

2. В зависимости от размеров выборки и природы данных мы можем адаптировать ахритектуру под свои нужды

25.3 Факторизационные машины

Пусть обучающая выборка представлена матрицей $X \in R^{\ell\times d}$, где ℓ - кол-во объектов, а d - кол-во признаков. Объектами являются пары пользователь-товар для которых известны оценки r_{ui} степени заинтересованности в товаре
Факторизационные машины учитывают взаимодействия признаков вплоть до некоторой степени

$$a(x) = w_0 + \sum_{j=1}^{d} w_j x_j + \frac{1}{2} \sum_{j=1}^{d} \sum_{k=1, k \neq j}^{d} < v_j, v_k > x_j x_k$$

Где $\vec{w} \in R, \vec{v_1} \in R^r$ - параметры модели, $\vec{x}$ - признаковое описание объекта
Матрица всех весов при слагаемых второго порядка представима в виде:

$$W = VV^T$$

Факторизационные машины используются в случаях, когда количество признаков очень велико

25.3.1 Методы обучения

Пусть имеется обучающая выборка $X = \{(x_i, y_i)\}_{i=1}^{\ell}$
Параметрами модели являются $\Theta = (w_0, w_1, \ldots, w_d, v_1, \ldots, v_d)$
Задача обучения:

$$Q(a, X) = \sum_{i=1}^{\ell} L(a(x_i; \Theta), y_i) + \sum_{j=1}^{|\Theta|} \lambda_j \theta_j^2 \to \min_{\Theta}$$

SGD

...

ALS

...

Markov Chain Monte Carlo

Параметры сэмплируются из апостериорного распределения

25.3.2 Частные случаи

Matrix Factorization

Естественный путь описания пары $(u, i) \in U \times I$ - бинарный вектор, содержащий две единицы. Первая стоит на u-ом месте. Вторая стоит на (|U| + i)-ом месте.
Факторизационная машина будет выглядеть следующим образом:

$$a(x) = w_0 + w_u + w_{|u|+i} + < v_u, v_{|U|+i} >$$

Данная модель совпадает с моделью в подходе, использующем матричное разложение

Pairwise Interaction Tensor Factorization

Допустим появилась информация о товарах - помимо множества пользователей U и множества товаров I теперь имеется множество тэгов T. Естественный

способ описания тройки $(u, i, l) \in U \times I$ - бинарный вектор, состоящий из нулей и содержащий три единицы

$$a(x) = w_0 + w_u + w_{|U|+i} + w_{|U|+|I|+l} + < v_u, v_{|U|+i} > + < v_u, v_{|U|+|I|+l} > + < v_{|U|+1}, v_{|U|+|I|+l} >$$

SVD++
Добавляется множественнозначный признак - товар может входить в несколько категорий.

$$a(x) = w_0 + w_u + w_i + < v_u, v_i > + \frac{1}{m} \sum_{j=1}^{k} < v_i, v_{l_j} > + \frac{1}{m} \sum_{j=1}^{m} w_{l_j} + \frac{1}{m} \sum_{j=1}^{m} < v_u, v_{l_j} > + \frac{1}{m^2} \sum_{j=1}^{m} \sum_{j'}^{m} < v_{l_j}, v_{l'_j} >$$

26 Интерпретируемость моделей

26.1 Интерпретация основанная на особенностях модели

Линейная модель
Естественная важность - веса после масштабирования. По абсолютному значению судим о силе влияния, по знаку на направление. Могут быть искажения из-за коррелированности признаков.
Решающие деревья
При переходе к композициям уже сложно визуализировать
Нейронные сети
Карты уверенности предсказаний:
Закрываем некоторый элемент исходного изображения и смотрим на уверенность предсказания по сравнению с исходным изображением. Чем сильнее уменьшается уверенность, тем важнее для предсказания эти пиксели

26.2 LIME

Идея: Интерпертировать предсказания некоторой объясняемой модели a(x) для заданного объекта $x^\star$ в его окрестности
Для каждого объекта наряду с признаковым описанием вводится его интерпретируемое описание - результатом работы метода является объясняющая модель a(x), строящая предсказания для интерпретируемых представлений.

В качестве интерпретируемых представлений обычно рассматривают достаточно простые бинарные признаковые описания объектов.

1. Для текста: Мешок слов с ограничением на количество слово или N-грамм

2. Для изображений: суперпиксели - непрерывные области похожих пикселей

Построение модели:
Составляется суррогатная выборка x_i:
Создаем объект x_i путем обнуления случайного количества единиц в интерпретируемом представлении x'
Переходим от представления x_i нового объекта к признаковому описанию x_i в исходном признаковом пространстве и положим $y_i = a(x_i)$. Таким образом, мы создали искусственную выборку в окрестности интерпретируемого представления x'
Итоговая модель может быть найдена, как решение задачи:

$$a = \arg\min_{b \in B} \sum_{x_i \in X^l_{x^\star}} \pi_{x^\star}(x_i)(a(x_i) - b(\bar{x}_i))^2 + \Omega(b)$$

$x^\star$ - объект, для которого происходит поиск интерпертации
B - семейство возможных объясняющих моделей
$\pi_{x^\star}(x)$ - вес объекта в функционале ошибки
$\Omega(b)$ - сложность модели
При использовании квадратичной функции потерь с экспоненицальным ядром в качестве функции π и семейства B линейных моделей получаем метод *разреженных линейных пространств*

26.3 Influential Instances

26.3.1 Диагностика через удаление

Оценка влияния объектов обучающей выборки на итоговую модель
Сравниваем модели с помощью *расстояния Кука*
Для задачи регрессии влияние объекта x_i обучающей выборки X^ℓ можно оценить с помощью:

$$D_i = \frac{\sum_{j=1}^{\ell}(a(x_j) - a^{-i}(x_j))^2}{d \times MSE(a, X^\ell)}$$

26.3.2 Функции влияния

Альтернативный вариант предлагает для моделей с дифференцируемой функцией потерь исследовать влияние изменения веса для конкретного примера обучающей выборки на параметры модели через влияние на функцию потерь
$\hat{\theta}$ - вектор параметров модели, обученной на выборке X^ℓ
$\hat{\theta}_{\epsilon,x_i}$ - вектор параметров при увеличении веса объекта x_i на ϵ

$$\hat{\theta}_{\epsilon,x_i} = \arg\min_\theta \frac{1}{l}\sum_{i=1}^{l} L(x_i,\theta) + \epsilon L(x_i,\theta)$$

$\hat{\theta}_{\epsilon,x_i} = \hat{\theta}$
Тогда: влияние увеличения веса на объекте x_i на ошибку на новом объекте x можно вычислить следующим образом:

$$l_{up,loss}(x_i,x) = \frac{\partial L(x,\hat{\theta}_{\epsilon,x_i})}{\partial\epsilon} \mid_{\epsilon=0} = \nabla_\theta L(x,\hat{\theta}_{0,x_i})^T \frac{d\hat{\theta_{\epsilon,x_i}}}{d\epsilon} \mid_{\epsilon=0} = -\nabla_\theta L(x,\hat{\theta})^T H_{\hat{\theta}}^{-}1 \nabla_\theta L(x,\hat{\theta})$$

Где $H_{\hat{\theta}} = \frac{1}{l}\sum_1^l \nabla_\theta^2 L(x_i,\hat{\theta})$
Использовать это можно для:

1. Сравнивать модели между собой
2. Детектировать несовпадение доменов между обучающим и тестовым множеством
3. Корректировать обучающие данные

26.4 Состязательные атаки

Состязательными объектами называются объекты, предсказание на которых меняются при малом изменении (наложение шума)
Методы делятся на:

1. White-box - с доступом к градиентам модели
2. Black-box - без доступа к градиентам модели

1. Ненаправленные - объект должен попасть в любой другой класс
2. Направленные - объект должен попасть в конкретный другой класс

В рамках атаки можно генерировать объект, решая задачу:

$$L(a(x^{adv}), y_{target}) + \lambda \left\| x^{adv} - x \right\| \to \min_{x^{adv}}$$

Пусть x - исходное изображение, y_{true} - его истинная метка.
Методы white-box атак:

1. Fast gradient sign method

   Изменяем объект в сторону градиента функции потерь для истинной метки

   $$x^{adv} = x + \epsilon sign(\nabla_x L(x, y_{true}))$$

2. Targeted fast gradient sign method

   Объект изменяется в сторону антиградиента функции потерь для целевой метки

   $$x^{adv} = x - \epsilon sign(\nabla_x L(x, y_{target}))$$

3. Iterative fast gradient sign method

   Вместо одного шага длины ϵ делается T шагов длины $\alpha = \frac{\epsilon}{T}$