Повороты признаков в алгоритме AdaBoost

А. Гой
Научный руководитель: В. В. Китов

Кафедра Математических Методов Прогнозирования
Факультет Вычислительной Математики и Кибернетики
Московский Государственный Университет имени М.В. Ломоносова

Технологии Баз Данных, 2016

План

1. Введение
   - Метод главных компонент

Постановка задачи

- **Дано:** $\mathbf{x}_1, \ldots, \mathbf{x}_N$ — множество точек (объектов) из $\mathbb{R}^D$.
- **Задача:** найти представления $\mathbf{y}_1, \ldots, \mathbf{y}_N$ из $\mathbb{R}^d$, где $d < D$.
- **Что хотелось бы получить (неформально):** необходимо найти такие представления, чтобы максимально сохранить всю информацию, которая содержится в $\mathbf{x}_1, \ldots, \mathbf{x}_N$
- **Метод главных компонент:** использует линейное преобразование, сохраняя максимум информации через максимизацию дисперсии:

$$\mathbf{XR} = \mathbf{Y}, \qquad (1)$$

где $\mathbf{X} \in \mathbb{R}^{N \times D}$, $\mathbf{Y} \in \mathbb{R}^{N \times d}$ — матрицы строками которых являются объекты, а $\mathbf{R} \in \mathbb{R}^{D \times d}$ — матрица столбцами которой являются ортонормированные вектора (главные компоненты) $\mathbf{r}_1, \ldots, \mathbf{r}_d$, образующие базис в $\mathbb{R}^d$

Оптимизационная задача

Как найти такие $\mathbf{r}_1, \ldots, \mathbf{r}_d$, которые

1. образуют ортанормированный базис;
2. переход к этому базис сохраняет максимальное количество дисперсии.

Оптимизационная задача:

$$\begin{cases} \|\mathbf{X}\mathbf{r}_k\|^2 \to \max_{\mathbf{r}_k} \\ \|\mathbf{r}_k\| = 1 \\ \mathbf{r}_k^T \mathbf{r}_1 = \cdots = \mathbf{r}_k^T \mathbf{r}_{k-1} = 0, \end{cases} \tag{2}$$

где норма $\|\mathbf{X}\mathbf{r}_k\|^2$ есть дисперсия проекций $\mathbf{x}_1, \ldots, \mathbf{x}_N$ на главную компонету $\mathbf{r}_k$

Визуализация

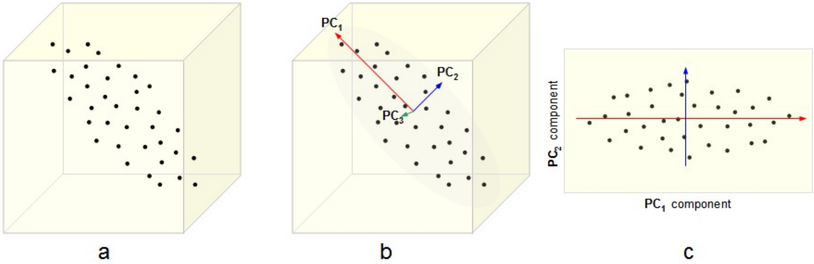


Рис.: a — исходная трехмерная выборка, b — главные компоненты отвечающие направлениям с максимальной дисперсией, c — спроецированная на двумерную плоскость выборка

Связь метода главных компонент и поворотов

- Рассмотрим случай $d = D$: вычисляем все главные компонеты $\mathbf{a}_1, \ldots, \mathbf{a}_D$ — новый базис в $\mathbb{R}^D$
- $\mathbf{R} = [\mathbf{r}_1; \ldots; \mathbf{r}_D]$ — матрица поворота ($\mathbf{R}^{-1} = \mathbf{R}^T$)

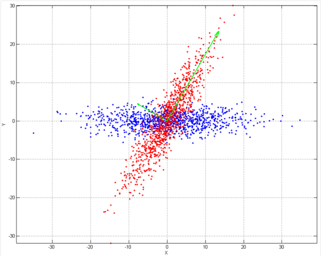


Рис.: Красным цветом обозначены исходные точки. Зеленые стрелки показывают направление векторов $\mathbf{r}_1, \mathbf{r}_2$. Синие точки полученные после умножения матрицы объектов $\mathbf{X}$ на матрицу вращения $\mathbf{R}$.

План

2. Применение повротов в AdaBoost
    - AdaBoost $\rightarrow$ Rotation AdaBoost

Схема работы алгоритма AdaBoost

Решающее правило:

$$y(\mathbf{x}) = \arg\max_t \sum_{l=1}^{L} \alpha_l \, [y_l(\mathbf{x}) = t]$$

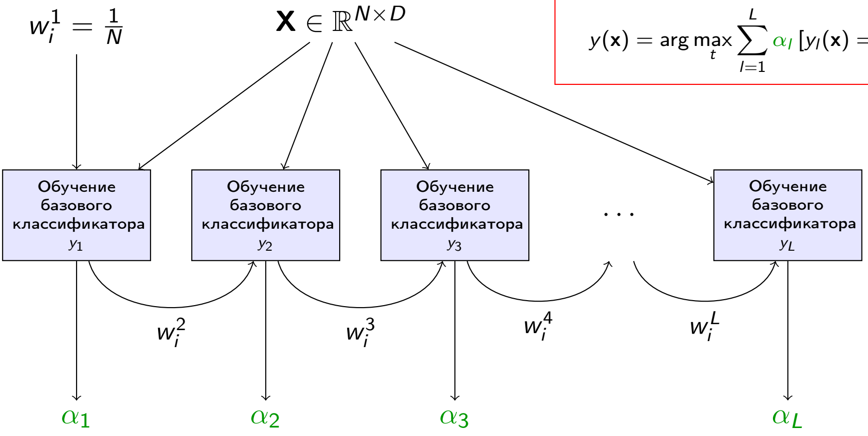
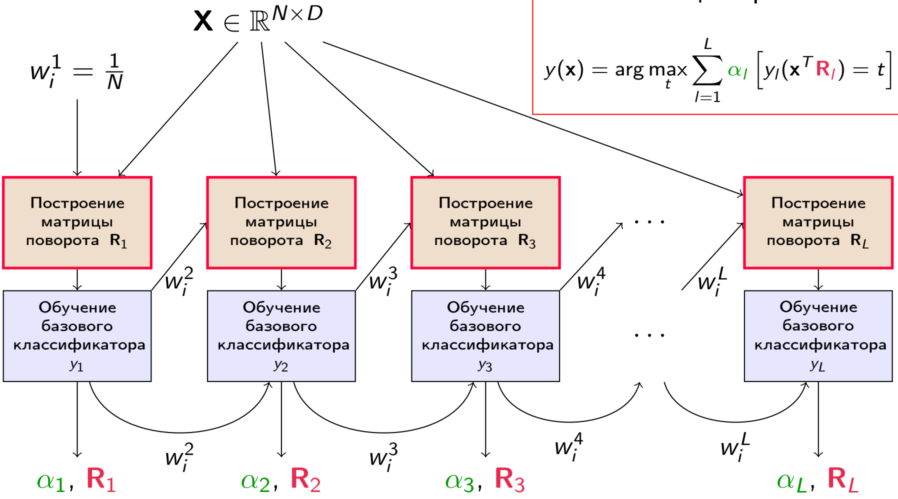


Схема работы алгоритма Rotation AdaBoost

Решающее правило:

$$y(\mathbf{x}) = \arg\max_t \sum_{l=1}^{L} \alpha_l \left[y_l(\mathbf{x}^T \mathbf{R}_l) = t \right]$$



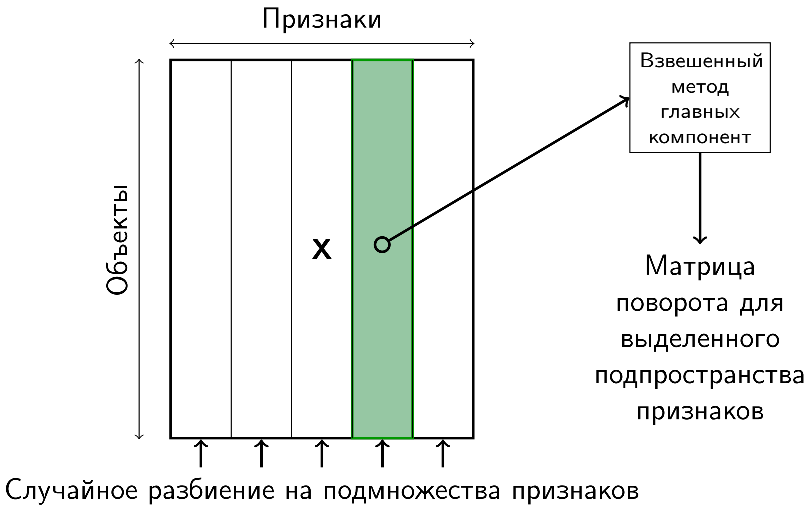
Псевдокод алгоритма AdaBoost

1. Проинициализируем веса: $w_i^1 = \frac{1}{N}$, $i = 1, \ldots, N$.
2. Для каждого $l = 1, \ldots, L$:
   a. Обучим алгоритм y_l на $(\mathbf{X}, \mathbf{t})$ при помощи весов w_i^l.
   b. Вычислим взвешенную ошибку ε_l алгоритма y_l:
   $$\varepsilon_l = \sum_{i=1}^{N} w_i^l \left[y_l(\mathbf{x}_i) \neq t_i \right].$$
   c. Вычислим вес нового классификатора:
   $$\alpha_l = \ln \frac{1 - \varepsilon_l}{\varepsilon_l} + \ln(K + 1).$$
   d. Пересчитаем и нормализуем веса:
   $$\overline{w}_n^{l+1} = w_i^l \cdot \exp\left(\alpha_l \cdot [y_l(\mathbf{x}_i) \neq t_i] \right), \; i = 1, \ldots, N,$$
   $$w_i^{l+1} = \frac{\overline{w}_i^{l+1}}{\sum_{i=1}^{N} \overline{w}_i^{l+1}}, \; i = 1, \ldots, N.$$

Псевдокод алгоритма Rotation AdaBoost

1. Проинициализируем веса: $w_i^1 = \frac{1}{N}$, $i = 1, \ldots, N$.
2. Для каждого $l = 1, \ldots, L$:
   a. Спец. образом построим матрицу $\mathbf{R}_l$ при помощи весов w_n^l.
   b. Обучим алгоритм y_l на $(\mathbf{X}\mathbf{R}_l, \mathbf{t})$ при помощи весов w_n^l.
   c. Вычислим взвешенную ошибку ε_l алгоритма $\mathbf{y}_l$:
   $$\varepsilon_l = \sum_{i=1}^{N} w_i^l \left[y_l(\mathbf{x}_i^\top \mathbf{R}_l) \neq t_i \right].$$
   d. Вычислим вес нового классификатора:
   $$\alpha_l = \ln \frac{1 - \varepsilon_l}{\varepsilon_l} + \ln(K + 1).$$
   e. Пересчитаем и нормализуем веса:
   $$\overline{w}_i^{l+1} = w_i^l \cdot \exp\left(\alpha_l \cdot \left[y_l(\mathbf{x}_i^\top \mathbf{R}_l) \neq t_i \right] \right), \; i = 1, \ldots, N,$$
   $$w_i^{l+1} = \frac{\overline{w}_i^{l+1}}{\sum_{i=1}^{N} \overline{w}_i^{l+1}}, \; i = 1, \ldots, N.$$

Детали поворотов



План

Условия эксперимента

- **Цель:** сравнить AdaBoost, Rotation Forest и Rotation AdaBoost
- **Данные:** 25 выборок из UCI Machine Learning Repository
- **Минимальная предобработка:**
  1. удаление объектов с пропущенными значенями;
  2. преобразование категориальных признаков при помощи бинарного кодирования (one-hot encoding);
  3. нормализация признаков к нулевому среднему и единичной дисперсии.
- **Метрика качества:** процент верно угаданых меток классов (точность, accuracy) на кросс-валидации по 10 блокам

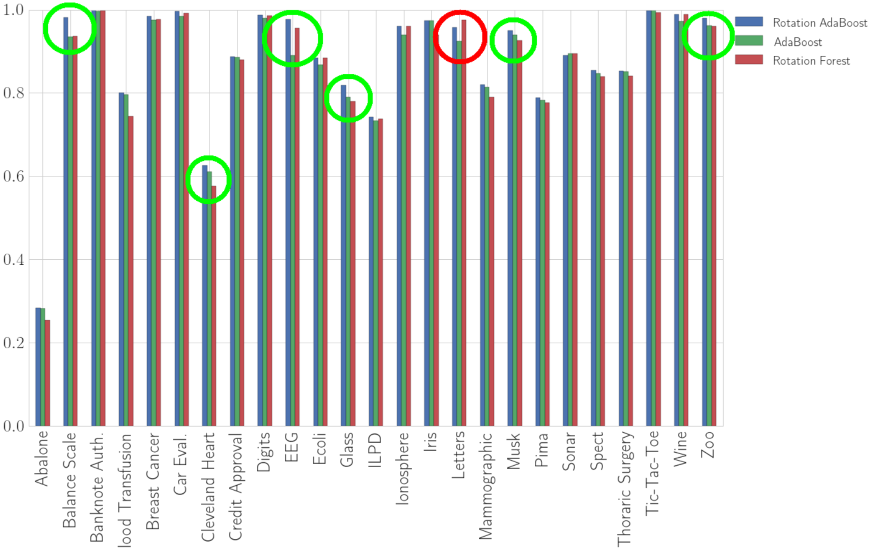
Гиперпараметры алгоритмов

- **Проблема:** наиболее честное сравнение — перебор всех гиперпараметров. Очень дорого.
- **Решение:** выберем только наиболее важные параметры, которые имеют ключевое влияние на качество предсказаний алгоритма.
- Точность алгоритма на выборке — лучшая точность показанная одной из его версий.

Гиперпараметры алгоритмов (продолжение)

- **AdaBoost:** *(фиксированный learning rate: 0.1)*
  1. максимальная глубина дерева решений: $\{1, 2, 3, 4, 5, 6\}$;
  2. количество деревьев в ансамбле: $\{1, 2, 3, \dots, 2500\}$;
- **Rotation Forest:** *(не ограничиваем глубину деревьев)*
  1. количество деревьев в ансамбле: $\{1, 2, 3, \dots, 2500\}$;
  2. максимальный размер случайных подмножеств признаков:
     - $f(x) = \lfloor \log_2(x) \rfloor$;
     - $f(x) = \lfloor \sqrt{x} \rfloor$;
     - $f(x) = x$.
- **Rotation AdaBoost:** *(фиксированный learning rate: 0.1)*
  1. максимальная глубина дерева решений: $\{1, 2, 3, 4, 5, 6\}$;
  2. количество деревьев в ансамбле: $\{1, 2, 3, \dots, 2500\}$;
  3. максимальный размер случайных подмножеств признаков.

Результаты


1.0
0.8
0.6
0.4
0.2
0.0
Abalone
Balance Scale
Banknote Auth.
lood Transfusion
Breast Cancer
Car Eval.
Cleveland Heart
Credit Approval
Digits
EEG
Ecoli
Glass
ILPD
Ionosphere
Iris
Letters
Mammographic
Musk
Pima
Sonar
Spect
Thoraric Surgery
Tic-Tac-Toe
Wine
Zoo
Rotation AdaBoost
AdaBoost
Rotation Forest

Результаты (продолжение)

	Rotation AdaBoost		*Rotation Forest*		*AdaBoost*	
Abalone	0.284	•	0.253		0.282	
Balance Scale	0.980	•	0.936		0.934	
Banknote Auth.	1.000	•	0.997		0.997	
Blood Transfusion	0.800	•	0.743		0.795	
Breast Cancer	0.984	•	0.977		0.975	
Car Eval.	0.996	•	0.991		0.983	
Cleveland Heart	0.626	•	0.576		0.611	
Credit Approval	0.886	•	0.879		0.885	
Digits	0.986	•	0.985		0.980	
EEG	0.976	•	0.955		0.889	
Ecoli	0.884		0.883		0.867	•
Glass	0.818	•	0.780		0.789	
ILPD	0.742	•	0.737		0.733	
Ionosphere	0.960	•	0.960		0.940	
Iris	0.973	•	0.940		0.973	•
Letters	0.957		0.974	•	0.925	
Mammographic	0.819	•	0.790		0.813	
Musk	0.949	•	0.926		0.938	
Pima	0.789	•	0.777		0.782	
Sonar	0.890		0.894		0.894	•
Spect	0.855	•	0.839		0.847	
Thoraric Surgery	0.853	•	0.840		0.851	
Tic-Tac-Toe	0.997		0.992		0.998	•
Wine	0.988	•	0.988	•	0.972	
Zoo	0.980	•	0.960		0.962	

Таблица: Наилучшие результаты рассматриваемых алгоритомов на каждой выборке. Символом • обозначен лучший результат в строке.

Размер случайных подмножеств

- Что лучше повороты в случайных подпространствах или поворот признакового пространства целиком?
- Логично предположить, что случайные повороты на каждой итерации могут дестабилизировать процедуру бустинга.

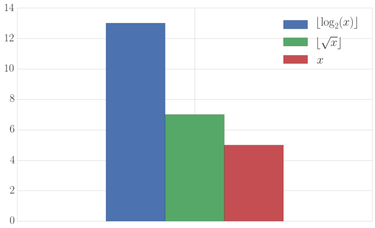


Рис.: Количество «побед» версий Rotation AdaBoost

Вывод: повороты в случайных подпространствах действильно важны и позволяют повысить точность классификатора

План

Направления дальнейших исследований

- Применить вышеописанную идею поворотов к градиентному бустингу.
- Процедура поворота при помощи метода главных компонет — очень вычислительно затратная операция (необходимо выполнять на каждой итерации SVD-разложение):
    - **Идея 1:** не обязательно находить точный поворот, а достаточно лишь знать приближенный ответ (метод главных компонет через EM-алгоритм);
    - **Идея 2:** преобразования методом главных компонент не на каждом шаге бустинга, а лишь на некоторых итерациях.
- Использовать интеллектуальный отбор признаков для очередного поворота.

Спасибо за внимание!