Методы машинного обучения. Линейные ансамбли

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

материалы курса:
github.com/MSU-ML-COURSE/ML-COURSE-24-25
орг.вопросы по курсу: ml.cmc@mail.ru

ВМК МГУ • 26 ноября 2024

Содержание

Определение ансамбля

$X^\ell = (x_i, y_i)_{i=1}^{\ell} \subset X \times Y$ — обучающая выборка, $y_i = y^*(x_i)$
$a_t\colon X \to Y,\ \ t = 1, \dots, T$ — обучаемые *базовые алгоритмы*

Идея ансамблирования (Ю.И.Журавлёв): как из множества по отдельности плохих алгоритмов a_t построить один хороший?

Декомпозиция базовых алгоритмов $a_t(x) = C(b_t(x))$
$a_t\colon X \xrightarrow{b_t} R \xrightarrow{C} Y$, где R — удобное *пространство оценок*,
b_t — *алгоритмические операторы*, C — *решающее правило*

Ансамбль (композиция) базовых алгоритмов $a_1, \dots, a_T$,
$F\colon R^T \to R$ — *корректирующая* (агрегирующая) *операция*

$$a(x) = C\bigl(F(b_1(x), \dots, b_T(x))\bigr),$$

Ю.И.Журавлёв. Об алгебраическом подходе к решению задач распознавания или классификации. Проблемы кибернетики, 1978.
M.Kearns, L.G.Valiant. Cryptographic limitations on learning Boolean formulae and finite automata. 1989.

Пространства оценок и решающие правила

- **Пример 1:** классификация, Y — конечное множество, $R = Y$, $C(b) \equiv b$ — решающее правило не используется.
- **Пример 2:** классификация на 2 класса, $Y = \{-1, +1\}$,
$$a(x) = \operatorname{sign}\bigl(b(x)\bigr),$$
$R = \mathbb{R}$, $b\colon X \to \mathbb{R}$ — *real-valued classifier*, $C(b) \equiv \operatorname{sign}(b)$.
- **Пример 3:** классификация на M классов $Y = \{1, \dots, M\}$,
$$a(x) = \arg\max_{y \in Y} b_y(x),$$
$R = \mathbb{R}^M$, $b\colon X \to \mathbb{R}^M$, $C(b_1, \dots, b_M) \equiv \arg\max\limits_{y \in Y} b_y$.
- **Пример 4:** регрессия, $Y = R = \mathbb{R}$, $C(b) \equiv b$ — решающее правило не используется.

Агрегирующие (корректирующие) функции

Общие требования к агрегирующей функции:

- $F(b_1, \dots, b_T, x) \in \bigl[\min_t b_t, \max_t b_t\bigr]$ — среднее по Коши $\forall x$
- $F(b_1, \dots, b_T, x)$ монотонно не убывает по всем b_t

Примеры агрегирующих функций:

- простое голосование (simple voting):

  $$F(b_1, \dots, b_T) = \frac{1}{T} \sum_{t=1}^{T} b_t$$

- взвешенное голосование (weighted voting):

  $$F(b_1, \dots, b_T) = \sum_{t=1}^{T} \alpha_t b_t, \quad \sum_{t=1}^{T} \alpha_t = 1, \quad \alpha_t \geqslant 0$$

- смесь алгоритмов (mixture of experts)
  с функциями компетентности (gating function) $g_t \colon X \to \mathbb{R}$

  $$F(b_1, \dots, b_T, x) = \sum_{t=1}^{T} g_t(x) b_t(x)$$

Проблема разнообразия (diversity) базовых алгоритмов

Измерение с.в. ξ по независимым наблюдениям $\{\xi_t\}$:

- $\mathsf{E}\frac{1}{T}(\xi_1 + \cdots + \xi_T) = \mathsf{E}\xi$ — матожидание среднего
- $\mathsf{D}\frac{1}{T}(\xi_1 + \cdots + \xi_T) = \frac{1}{T}\mathsf{D}\xi$ — дисперсия $\to 0$ при $T \to \infty$

Но базовые алгоритмы не являются независимыми с.в.:

- решают одну и ту же задачу
- настраиваются на один целевой вектор (y_i)
- обычно выбираются из одной и той же модели

Способы повышения *разнообразия* базовых алгоритмов:

- обучение по различным (случайным) подвыборкам
- обучение по различным (случайным) наборам признаков
- обучение из разных параметрических моделей
- обучение с использованием рандомизации
- (иногда даже) обучение по зашумлённым данным

Методы стохастического ансамблирования

Способы повышения разнообразия с помощью рандомизации:

- bagging (bootstrap aggregating) — подвыборки обучающей выборки «с возвращением», в каждую выборку попадает $1 - \left(1 - \frac{1}{\ell}\right)^{\ell} \to 1 - \frac{1}{e} \approx 63.2\%$ объектов, при $\ell \to \infty$
- pasting — случайные обучающие подвыборки
- random subspaces — случайные подмножества признаков
- random patches — случ. подмн-ва и объектов, и признаков
- cross-validated committees — выборка разбивается на k блоков (k-fold) и делается k обучений без одного блока

Пусть $\mu\colon (G, U) \mapsto b$ — *метод обучения* по подвыборке $U \subseteq X^{\ell}$, использующий только признаки из $G \subseteq F^n = \{f_1, \dots, f_n\}$

Tin Kam Ho. The random subspace method for constructing decision forests. 1998.
Leo Breiman. Bagging predictors // Machine Learning. 1996.

Методы стохастического ансамблирования в одном псевдо-коде

Вход: обучающая выборка X^ℓ; параметры: T,
- ℓ' — объём обучающих подвыборок,
- n' — размерность признаковых подпространств,
- ε_1 — порог качества базовых алгоритмов на обучении,
- ε_2 — порог качества базовых алгоритмов на контроле;

Выход: базовые алгоритмы b_t, $t = 1, \dots, T$;

для всех $t = 1, \dots, T$:
- $U_t :=$ случайная подвыборка объёма ℓ' из X^ℓ;
- $G_t :=$ случайное подмножество мощности n' из F^n;
- $b_t := \mu(G_t, U_t)$;
- **если** $Q(b_t, U_t) > \varepsilon_1$ **то** не включать b_t в ансамбль;
- **если** $Q(b_t, X^\ell \backslash U_t) > \varepsilon_2$ **то** не включать b_t в ансамбль;

Ансамбль — простое голосование: $b(x) = \frac{1}{T} \sum\limits_{t=1}^{T} b_t(x)$

Несмещённая оценка ошибок

Out-of-bag — несмещённая оценка ансамбля на объекте:

$$\mathrm{OOB}(x_i) = \frac{1}{|T_i|} \sum_{t \in T_i} b_t(x_i), \qquad T_i = \{t\colon x_i \notin U_t\}$$

Несмещённая оценка ошибки ансамбля на обучающей выборке:

$$\mathrm{OOB}(X^\ell) = \sum_{i=1}^{\ell} \mathscr{L}\bigl(\mathrm{OOB}(x_i), y_i\bigr),$$

где $\mathscr{L}\bigl(b(x_i), y_i\bigr)$ — значение функции потерь на объекте x_i.

Оценивание важности признаков $f_j,\ \ j = 1, \ldots, n$:

$$\mathrm{importance}_j = \frac{\mathrm{OOB}^j(X^\ell) - \mathrm{OOB}(X^\ell)}{\mathrm{OOB}(X^\ell)} \cdot 100\%,$$

где при вычислении $b_t(x_i)$ для OOB^j значения признака f_j случайным образом перемешиваются на всех объектах $x_i \notin U_t$.

Преобразование простого голосования во взвешенное

- **Линейная модель** над готовыми признаками $b_t(x)$:
$$b(x) = \sum_{t=1}^{T} \alpha_t b_t(x)$$
- **Обучение:** МНК для регрессии, LR для классификации:
$$Q(\alpha, X^\ell) = \sum_{i=1}^{\ell} \mathscr{L}\bigl(b(x_i), y_i\bigr) \to \min_{\alpha}.$$
Регуляризация: $\alpha_t \geqslant 0$ либо LASSO: $\sum_{t=1}^{T} |\alpha_t| \leqslant \varkappa$.
- **Наивный байесовский классификатор** предполагает независимость с.в. $b_t(x)$ и даёт аналитическое решение:
$$\alpha_t = \ln \frac{1 - p_t}{p_t}, \quad t = 1, \dots, T,$$
p_t — оценка вероятности ошибки базового алгоритма b_t.

Случайный лес (Random Forest)

Грубое обучение деревьев для случайного леса:

- бэггинг над решающими деревьями, без pruning
- признак в каждой вершине дерева выбирается из случайного подмножества k из n признаков. По умолчанию $k = \lfloor n/3 \rfloor$ для регрессии, $k = \lfloor \sqrt{n} \rfloor$ для классификации

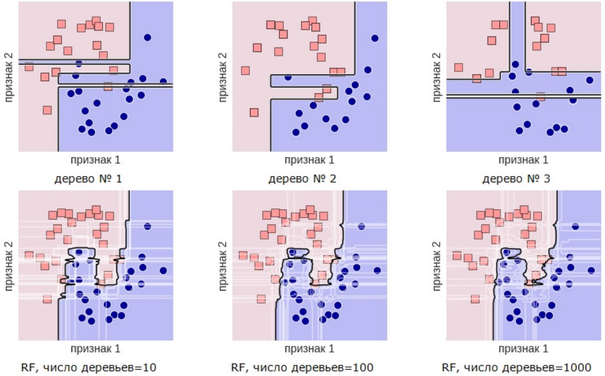
Параметры, которые можно настраивать (в частности, по OOB):

- число T деревьев
- число k случайно выбираемых признаков
- максимальная глубина деревьев
- минимальное число объектов в расщепляемой подвыборке
- минимальное число объектов в листьях
- критерий расщепления: MSE для регрессии, энтропийный или Джини для классификации

Breiman L. Random Forests. Machine Learning, 2001.

Постепенное сглаживание разделяющей поверхности

Пример разделения выборки с помощью отдельных деревьев (показаны соответствующие бутстреп-подвыборки) и случайного леса с числом деревьев 10, 100, 1000:



https://dyakonov.org/2019/04/19/ансамбли-в-машинном-обучении

Разновидности решающих лесов

- Случайный лес (Random Forest)
- Использование большого числа простых решающих деревьев в качестве признаков, в любом классификаторе.
- Oblique Random Forest, Rotation Forest
  $f_v(x)$ — линейные комбинации признаков, выбираемые по энтропийному критерию информативности.
- Решающий список из решающих деревьев:
  — при образовании статистически ненадёжного листа этот лист заменяется переходом к следующему дереву;
  — следующее дерево строится по объединению подвыборок, прошедших через ненадёжные листы предыдущего дерева.

https://en.wikipedia.org/wiki/Random_forest

Преимущества и ограничения стохастического ансамблирования

Преимущества:

- метод-обёртка (envelop) над базовым методом обучения
- подходит для классификации, регрессии и других задач
- простая реализация и простое распараллеливание
- возможность получения несмещённых оценок OOB
- возможность оценивания важности признаков
- RF — один из лучших универсальных методов в ML

Ограничения:

- требуется оооооочень много базовых алгоритмов
- трудно агрегировать устойчивые базовые методы обучения

https://dyakonov.org/2016/11/14/случайный-лес-random-forest

Бустинг для задачи классификации с двумя классами

Возьмём $Y = \{\pm 1\}$, $b_t \colon X \to \{-1, 0, +1\}$, $C(b) = \operatorname{sign}(b)$.
$b_t(x) = 0$ — отказ (лучше промолчать, чем соврать).

Взвешенное голосование:

$$a(x) = \operatorname{sign}\left(\sum_{t=1}^{T} \alpha_t b_t(x)\right), \quad x \in X.$$

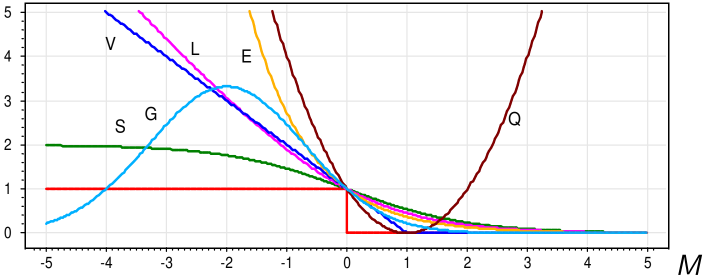
Функционал качества композиции — число ошибок на X^ℓ:

$$Q_T = \sum_{i=1}^{\ell} \left[y_i \sum_{t=1}^{T} \alpha_t b_t(x_i) < 0 \right].$$

Две основные эвристики бустинга:

- фиксация $\alpha_1 b_1(x), \ldots, \alpha_{t-1} b_{t-1}(x)$ при добавлении $\alpha_t b_t(x)$;
- гладкая аппроксимация пороговой функции потерь $[M \leqslant 0]$.

Гладкие аппроксимации пороговой функции потерь $[M < 0]$



$E(M) = e^{-M}$ — экспоненциальная (AdaBoost);
$L(M) = \log_2(1 + e^{-M})$ — логарифмическая (LogitBoost);
$Q(M) = (1 - M)^2$ — квадратичная (GentleBoost);
$G(M) = \exp\bigl(-cM(M + s)\bigr)$ — гауссовская (BrownBoost);
$S(M) = 2(1 + e^{M})^{-1}$ — сигмоидная;
$V(M) = (1 - M)_+$ — кусочно-линейная (из SVM);

Экспоненциальная аппроксимация пороговой функции потерь

Оценка функционала качества Q_T сверху:

$$Q_T \leqslant \widetilde{Q}_T = \sum_{i=1}^{\ell} \underbrace{\exp\Bigl(-y_i \sum_{t=1}^{T-1} \alpha_t b_t(x_i)\Bigr)}_{w_i} \exp\bigl(-y_i \alpha_T b_T(x_i)\bigr)$$

Нормированные веса: $\widetilde{W}^\ell = (\tilde{w}_1, \dots, \tilde{w}_\ell)$, $\tilde{w}_i = w_i \,/\, \sum_{j=1}^{\ell} w_j$.

Взвешенная доля ошибочных (negative) и правильных (positive) классификаций с нормированными весами $U^\ell = (u_1, \dots, u_\ell)$:

$$N(b, U^\ell) = \sum_{i=1}^{\ell} u_i \bigl[b(x_i) = -y_i\bigr]; \quad P(b, U^\ell) = \sum_{i=1}^{\ell} u_i \bigl[b(x_i) = y_i\bigr].$$

$1 - N - P$ — взвешенная доля отказов от классификации.

Y.Freund, R.E.Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. 1995

Основная теорема бустинга (для AdaBoost)

Пусть $\mathscr{B}$ — *достаточно богатое* семейство базовых алгоритмов.

Теорема (Freund, Schapire, 1996)

Пусть для любого нормированного вектора весов U^ℓ существует алгоритм $b \in \mathscr{B}$, классифицирующий выборку хотя бы немного лучше, чем наугад: $P(b; U^\ell) > N(b; U^\ell)$.

Тогда минимум функционала $\widetilde{Q}_T$ достигается при

$$b_T = \arg\max_{b \in \mathscr{B}} \sqrt{P(b; \widetilde{W}^\ell)} - \sqrt{N(b; \widetilde{W}^\ell)}.$$

$$\alpha_T = \frac{1}{2} \ln \frac{P(b_T; \widetilde{W}^\ell)}{N(b_T; \widetilde{W}^\ell)}.$$

R.E.Schapire, Y.Singer. Improved boosting using confidence-rated predictions. 1999

Доказательство (шаг 1 из 2)

Воспользуемся тождеством $\forall\alpha\in\mathbb{R},\ \ \forall b\in\{-1,0,+1\}$:
$e^{-\alpha b} = e^{-\alpha}[b=1] + e^{\alpha}[b=-1] + [b=0]$.

Положим для краткости $\alpha = \alpha_T$ и $b_i = b_T(x_i)$. Тогда

$$\widetilde{Q}_T = \Biggl(e^{-\alpha}\underbrace{\sum_{i=1}^{\ell}\tilde{w}_i[b_i=y_i]}_{P} + e^{\alpha}\underbrace{\sum_{i=1}^{\ell}\tilde{w}_i[b_i=-y_i]}_{N} + \underbrace{\sum_{i=1}^{\ell}\tilde{w}_i[b_i=0]}_{1-P-N}\Biggr)\underbrace{\sum_{i=1}^{\ell}w_i}_{\widetilde{Q}_{T-1}}$$

$$= \bigl(e^{-\alpha}P + e^{\alpha}N + (1-P-N)\bigr)\,\widetilde{Q}_{T-1} \to \min_{\alpha,b}.$$

$$\tfrac{\partial}{\partial\alpha}\widetilde{Q}_T = \bigl(-e^{-\alpha}P + e^{\alpha}N\bigr)\widetilde{Q}_{T-1} = 0 \ \Rightarrow\ e^{-\alpha}P = e^{\alpha}N \ \Rightarrow\ e^{2\alpha} = \tfrac{P}{N}.$$

Получили требуемое: $\boxed{\alpha_T = \tfrac{1}{2}\ln\tfrac{P}{N}}$.

Доказательство (шаг 2 из 2)

Подставим оптимальное значение $\alpha = \frac{1}{2}\ln\frac{P}{N}$ обратно в $\widetilde{Q}_T$:

$$\begin{aligned}\widetilde{Q}_T &= \bigl(e^{-\alpha}P + e^{\alpha}N + (1 - P - N)\bigr)\,\widetilde{Q}_{T-1} = \\ &= \Bigl(1 + \sqrt{\tfrac{N}{P}}P + \sqrt{\tfrac{P}{N}}N - P - N\Bigr)\,\widetilde{Q}_{T-1} = \\ &= \Bigl(1 - \bigl(\sqrt{P} - \sqrt{N}\bigr)^2\Bigr)\,\widetilde{Q}_{T-1} \to \min_b .\end{aligned}$$

Поскольку $\widetilde{Q}_{T-1}$ не зависит от α_T и b_T, минимизация $\widetilde{Q}_T$ эквивалентна либо максимизации $\sqrt{P} - \sqrt{N}$ при $P > N$, либо максимизации $\sqrt{N} - \sqrt{P}$ при $P < N$, однако второй случай исключён условием теоремы.

Получили $\boxed{b_T = \arg\max_b \sqrt{P} - \sqrt{N}}$. Теорема доказана.

Следствие 1. Сходимость

Теорема

Если на каждом шаге семейство $\mathscr{B}$ и метод обучения обеспечивают построение базового алгоритма b_t такого, что

$$\sqrt{P(b_t; \widetilde{W}^\ell)} - \sqrt{N(b_t; \widetilde{W}^\ell)} = \gamma_t > \gamma$$

при некотором $\gamma > 0$, то за конечное число шагов будет построен корректный алгоритм $a(x)$.

Доказательство. Q_T сходится к нулю со скоростью геометрической прогрессии:

$$Q_{T+1} \leqslant \widetilde{Q}_{T+1} = \widetilde{Q}_T(1-\gamma^2) \leqslant \cdots \leqslant \widetilde{Q}_1(1-\gamma^2)^T.$$

Наступит момент, когда $\widetilde{Q}_T < 1$.
Но тогда $Q_T = 0$, поскольку $Q_T \in \{0, 1, \ldots, \ell\}$.

Следствие 2. Исходный (классический) вариант AdaBoost

Пусть отказов нет, $b_t\colon X \to \{\pm 1\}$. Тогда $P = 1 - N$.

Теорема (Freund, Schapire, 1995)

Пусть для любого нормированного вектора весов U^ℓ существует алгоритм $b \in \mathscr{B}$, классифицирующий выборку хотя бы немного лучше, чем наугад: $N(b; U^\ell) < \frac{1}{2}$.

Тогда минимум функционала $\widetilde{Q}_T$ достигается при

$$b_T = \arg\min_{b \in \mathscr{B}} N(b; \widetilde{W}^\ell).$$

$$\alpha_T = \frac{1}{2} \ln \frac{1 - N(b_T; \widetilde{W}^\ell)}{N(b_T; \widetilde{W}^\ell)}.$$

Y.Freund, R.E.Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. 1995

Алгоритм AdaBoost (в исходном варианте)

Вход: обучающая выборка X^ℓ; параметр T;
Выход: базовые алгоритмы и их веса $\alpha_t b_t$, $t = 1, \dots, T$;

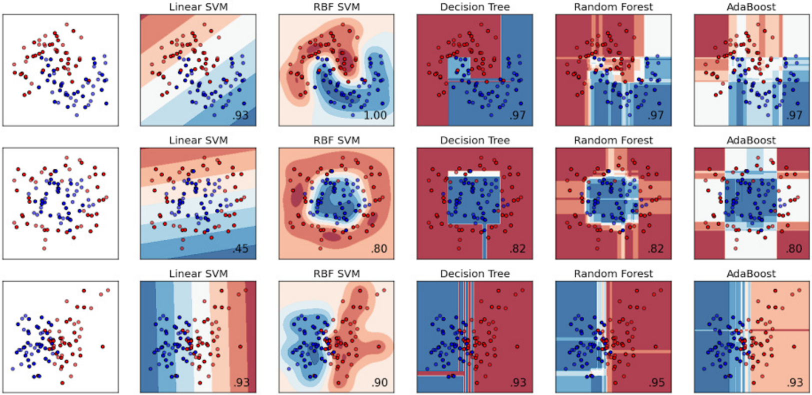
инициализировать веса объектов: $w_i := 1/\ell$, $i = 1, \dots, \ell$;
для всех $t = 1, \dots, T$:
- обучить базовый алгоритм:
$$b_t := \arg\min_b N(b; W^\ell);$$
$$\alpha_t := \frac{1}{2} \ln \frac{1 - N(b_t; W^\ell)}{N(b_t; W^\ell)};$$
- обновить веса объектов:
$$w_i := w_i \exp\bigl(-\alpha_t y_i b_t(x_i)\bigr), \quad i = 1, \dots, \ell;$$
- нормировать веса объектов:
$$w_0 := \textstyle\sum_{j=1}^\ell w_j;$$
$$w_i := w_i / w_0, \quad i = 1, \dots, \ell;$$

Эвристики и рекомендации

- **Базовые классификаторы (weak classifiers):**
  — решающие деревья — используются чаще всего;
  — пороговые правила, т.н. «решающие пни» (data stumps)
  $$\mathscr{B} = \Big\{ b(x) = \big[f_j(x) \lessgtr \theta\big] \;\Big|\; j = 1, \dots, n,\ \theta \in \mathbb{R} \Big\};$$
  — для SVM бустинг не эффективен.
- **Отсев шума:** отбросить объекты с наибольшими w_i.
- **Модификация формулы для α_t на случай $N = 0$:**
  $$\alpha_t := \frac{1}{2} \ln \frac{1 - N(b_t; W^\ell) + \frac{1}{\ell}}{N(b_t; W^\ell) + \frac{1}{\ell}};$$
- **Дополнительный критерий остановки:**
  увеличение частоты ошибок на контрольной выборке.

Случайный лес и бустинг в сравнении с другими методами

Эксперименты на трёх двумерных модельных выборках:

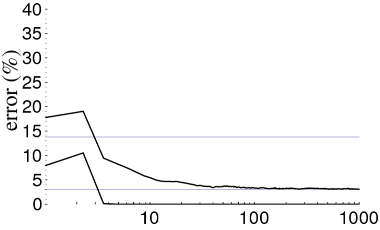


Решения могут выглядеть странно... тем не менее, RF и бустинг — одни из самых сильных универсальных методов в ML

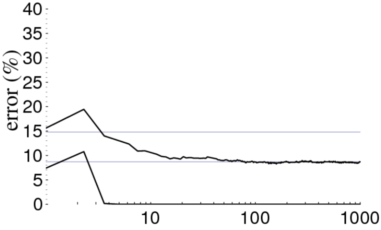
Эксперименты с алгоритмом классификации AdaBoost

Удивительное отсутствие переобучения вплоть до $T = 1000$ (нижняя кривая — обучение, верхняя – тест):

Задача UCI:letter



Задача UCI:satimage

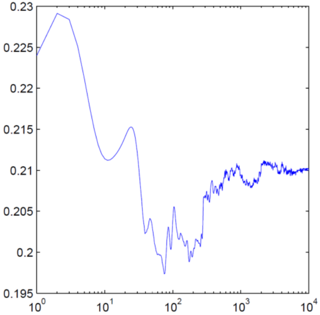


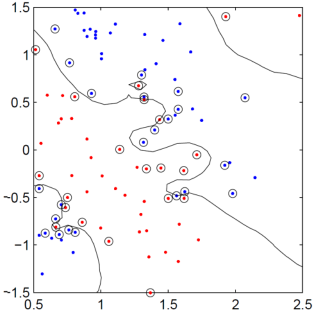
До этих экспериментов считалось, что увеличение числа параметров неизбежно приводит к переобучению

R.E.Schapire, Y.Freund, Wee Sun Lee, P.Bartlett. Boosting the margin: a new explanation for the effectiveness of voting methods. Annals of Statistics, 1998.

Иногда AdaBoost всё же переобучается...

... но не сильно, и на тысячах базовых классификаторах.
Слева: зависимость ошибки на тестовой выборке от $|T|$.
Справа: разделяющая поверхность при переобучении.





G.Rätsch, T.Onoda, K.R.Müller. An improvement of AdaBoost to avoid overfitting. 1998.

Недостатки AdaBoost

- Чрезмерная чувствительность к выбросам из-за e^{-M}
- Неинтерпретируемое нагромождение из сотен алгоритмов
- Не удаётся строить короткие композиции из «сильных» алгоритмов типа SVM (только длинные из «слабых»)
- Требуются достаточно большие обучающие выборки (бэггинг обходится более короткими)

Способы устранения:

- Отсев выбросов по критерию увеличения веса w_i
- Обучение b_t по случайным подвыборкам как в бэггинге
- Градиентный бустинг с произвольными функциями потерь
- Явная оптимизация распределения отступов

Резюме

- Ансамбли позволяют решать сложные задачи, которые плохо решаются отдельными базовыми алгоритмами
- Обычно ансамбль строится *алгоритмом-обёрткой* (envelop): базовые алгоритмы обучаются готовыми методами
- Базовые алгоритмы: компромисс качество/различность
- Две основные эвристики бустинга (и не только AdaBoost):
  — обучать базовые алгоритмы по одному
  — использовать гладкую замену пороговой функции потерь
- Важное открытие середины 90-х: обобщающая способность бустинга не ухудшается с ростом сложности T
- Практическое сравнение бустинга и бэггинга:
  — бустинг лучше для классов с границами сложной формы
  — бэггинг лучше для коротких обучающих выборок
  — бэггинг легче распараллеливается