Точные оценки вероятности переобучения

К. В. Воронцов (vokov@forecsys.ru)

27 сентября 2009 г.

Содержание

1 Вероятность переобучения

Пусть задано конечное множество объектов $\mathbb{X} = \{x_1, \ldots, x_L\}$, называемое *генеральной* выборкой, и множество A, элементы которого называются *алгоритмами*. Существует бинарная функция $I\colon A \times \mathbb{X} \to \{0, 1\}$, называемая *индикатором ошибки*. Если $I(a, x) = 1$, то говорят, что алгоритм a допускает ошибку на объекте x.

В задачах классификации или регрессии алгоритмами являются функции на множестве объектов $\mathbb{X}$. Однако для целей данного исследования нет необходимости конкретизировать понятие «алгоритма»; достаточно считать алгоритмы элементами некоторого абстрактного множества A, предполагая лишь, что существует способ определить, допускает ли алгоритм a ошибку на объекте x. Такое понимание «алгоритма» позволяет расширить класс рассматриваемых задач.

Вектором ошибок алгоритма a будем называть L-мерный бинарный вектор $\vec{a} = \big(I(a, x_i)\big)_{i=1}^{L}$. Числом ошибок алгоритма a на выборке $X \subseteq \mathbb{X}$ называется величина

$$n(a, X) = \sum_{x \in X} I(a, x).$$

Частотой ошибок или *эмпирическим риском* алгоритма a на выборке $X \subseteq \mathbb{X}$ называется величина $\nu(a, X) = \frac{1}{|X|} n(a, X)$.

Алгоритм $a \in A$ называется *корректным* на выборке $X \subseteq \mathbb{X}$, если $n(a, X) = 0$.

Пусть ℓ — фиксированное натуральное число, $\ell < L$. Обозначим через $[\mathbb{X}]^\ell$ множество всех ℓ-элементных подмножеств генеральной выборки $\mathbb{X}$. Мощность $[\mathbb{X}]^\ell$, очевидно, равна C_L^ℓ.

Методом обучения называется отображение $\mu\colon [\mathbb{X}]^\ell \to A$, которое произвольной *обучающей выборке* $X \in [\mathbb{X}]^\ell$ ставит в соответствие некоторый алгоритм $a = \mu X$ из A. Метод обучения μ называется методом *минимизации эмпирического риска*, если

$$\mu X = \arg\min_{a \in A} n(a, X). \tag{1.1}$$

Уклонением частот ошибок алгоритма a на двух выборках X и $\bar{X} = \mathbb{X} \backslash X$ называется разность частот $\delta(a, X) = \nu(a, \bar{X}) - \nu(a, X)$. *Переобученностью* метода μ на выборке X будем называть уклонение частот ошибок алгоритма $a = \mu X$:

$$\delta_\mu(X) = \delta\big(\mu X, X\big) = \nu\big(\mu X, \bar{X}\big) - \nu\big(\mu X, X\big).$$

Будем говорить, что метод μ *переобучен* на выборке X, если $\delta_\mu(X) \geqslant \varepsilon$, где ε — положительный вещественный параметр, называемый *порогом переобучения*.

Обычно термин «переобучение» вводится неформально и обозначает нежелательное явление, когда алгоритм, настроенный по обучающей выборке, допускает слишком много ошибок на контрольных данных. Здесь этому термину придаётся более строгий формальный смысл. Нашей основной задачей будет получение точных оценок *вероятности переобучения*, то есть вероятности того, что $\delta_\mu(X) \geqslant \varepsilon$.

2 Слабая вероятностная аксиоматика

Теория статистического обучения основана на предположении, что выборка $\mathbb{X}$ — простая, то есть что её элементы выбраны из некоторой (как правило, бесконечной) генеральной совокупности случайно, независимо, согласно одной и той же (как правило, неизвестной) вероятностной мере.

Мы будем придерживаться более слабой вероятностной аксиоматики [18].

Пусть $\mathbb{X}$ — произвольное конечное множество объектов. Предполагается, что все его C_L^ℓ разбиений на наблюдаемую обучающую выборку X длины ℓ и скрытую контрольную выборку $\bar{X}$ длины $k = L - \ell$ реализуются с равной вероятностью. Данное предположение фактически эквивалентно стандартной гипотезе о независимости элементов выборки $\mathbb{X}$. Однако существование вероятностной меры на всём пространстве объектов не предполагается, и даже само это пространство не вводится.

В слабой аксиоматике событиями являются подмножества разбиений выборки $\mathbb{X}$. Точнее, для произвольного предиката $\beta\colon [\mathbb{X}]^\ell \to \{0,1\}$ вероятность события $\beta(X)$ определяется как доля разбиений, при которых $\beta(X) = 1$:

$$\mathsf{P}\,\beta(X) = \frac{1}{C_L^\ell} \sum_{X \in [\mathbb{X}]^\ell} \beta(X).$$

Цель данной работы — получение точных оценок *вероятности переобучения* для метода минимизации эмпирического риска μ:

$$Q_\varepsilon \equiv Q_\varepsilon(\mu, \mathbb{X}) = \mathsf{P}\left[\delta_\mu(X) \geqslant \varepsilon\right]. \tag{2.1}$$

Здесь и далее квадратные скобки означают преобразование логического выражения в число 0 или 1 по правилам $[\textit{истина}] = 1$, $[\textit{ложь}] = 0$.

Введение слабой аксиоматики мотивируется следующими соображениями.

Во-первых, в задачах анализа данных выборки могут быть только конечными, будь то уже известные наблюдаемые данные, или скрытые данные, которые станут известны в будущем. В некоторых задачах число предсказаний k настолько мало́, что вводить «вероятность ошибки» как предел частоты ошибок при $k \to \infty$ просто некорректно. Слабая аксиоматика позволяет получать точные оценки, справедливые при любых конечных ℓ и k, не прибегая к асимптотическим методам. Понятие «вероятность ошибки» в слабой аксиоматике вообще не определяется. Качество алгоритмов характеризуется частотой их ошибок на конечных выборках. Понятие переобученности определяется как отклонение частоты ошибок в двух подвыборках, а не как отклонение частоты ошибок от её вероятности. Заметим, что такой подход не является новым в теории статистического обучения. Фундаментальные работы Вапника и Червоненкиса [3, 4] также основывались на оценках уклонения частот в двух подвыборках.

Во-вторых, вероятности, определяемые через «долю разбиений выборки», легко оценивать эмпирически. Для этого можно применять, в частности, метод Монте-Карло, заменяя среднее по всем разбиениям средним по случайному подмножеству

разбиений. Эта методика напоминает скользящий контроль [8, 10], но отличается от него тем, что оценивается не эмпирическое среднее частоты ошибок на контроле, а эмпирическое распределение переобученности δ_μ. Это позволило в [18] выделить и численно сравнить четыре основных фактора завышенности классических оценок Вапника-Червоненкиса. Вообще, в слабой аксиоматике яснее прослеживается связь теоретических оценок с эмпирическими методиками типа перестановочных тестов или скользящего контроля.

В-третьих, оценки вида $Q_\varepsilon(\mu, \mathbb{X}) \leqslant \eta(\varepsilon)$ при необходимости легко переносятся из слабой аксиоматики в сильную (колмогоровскую). Для этого достаточно взять математическое ожидание по полной выборке $\mathbb{X}$ от обеих частей неравенства:

$$\mathsf{P}_{\mathbb{X}}\big\{\delta_\mu(X) \geqslant \varepsilon\big\} = \mathsf{E}_{\mathbb{X}} Q_\varepsilon(\mu, \mathbb{X}) \leqslant \mathsf{E}_{\mathbb{X}}\eta(\varepsilon),$$

разумеется, дополнительно предположив, что выборка $\mathbb{X}$ — простая. Если оценка $\eta(\varepsilon)$ не зависит от полной выборки $\mathbb{X}$, то она непосредственно переносится из слабой аксиоматики в сильную. Если оценка зависит от некоторой функции полной выборки $T(\mathbb{X})$, то значение этой функции надо либо интерпретировать как априорное знание, либо оценивать по наблюдаемой части выборки. Во всех этих случаях вид результирующей оценки не меняется при переходе от слабой аксиоматики к сильной. Поэтому вполне допустимо оставаться в рамках слабой аксиоматики.

В слабой аксиоматике особую роль играет *гипергеометрическая функция вероятности*

$$h_L^{\ell,m}(s) = C_m^s C_{L-m}^{\ell-s} / C_L^\ell,$$

выражающая долю способов, которыми можно выбрать ℓ объектов без возвращений из генеральной выборки длины L, содержащей m ошибок, получив при этом ровно s ошибок. Функция $h_L^{\ell,m}(s)$ имеет три целочисленных параметра L, $\ell = 0, \ldots, L$, $m = 0, \ldots, L$ и аргумент s, пробегающий целые значения от $s_0 = \max\{0, m - k\}$ до $s_1 = \min\{m, \ell\}$. При всех других целых значениях m и s договоримся доопределять биномиальные коэффициенты C_m^s и функцию $h_L^{\ell,m}(s)$ нулём.

Введём стандартным образом *гипергеометрическую функцию распределения*

$$H_L^{\ell,m}(z) = \sum_{s=s_0}^{\lfloor z \rfloor} h_L^{\ell,m}(s). \tag{2.2}$$

Следующее утверждение является переформулировкой хорошо известных классических фактов в терминах слабой аксиоматики.

Лемма 2.1. *Пусть алгоритм a допускает m ошибок на генеральной выборке, $n(a, \mathbb{X}) = m$. Тогда вероятность того, что алгоритм a допускает s ошибок на выборке X, описывается гипергеометрической функцией вероятности:*

$$\mathsf{P}\big[n(a, X) = s\big] = \mathsf{P}\big[n(a, \bar{X}) = m - s\big] = h_L^{\ell,m}(s),$$

а вероятность большого уклонения частот ошибок алгоритма a описывается гипергеометрической функцией распределения:

$$\mathsf{P}\big[\delta(a,X) \geqslant \varepsilon\big] = H_L^{\ell,m}\left(\tfrac{\ell}{L}(m-\varepsilon k)\right). \tag{2.3}$$

Замечание 2.1. Неравенство $\delta(a,X) = \frac{m-s}{k} - \frac{s}{\ell} \geqslant \varepsilon$ равносильно $s \leqslant \frac{\ell}{L}(m-\varepsilon k)$, следовательно, значение $\left\lfloor \frac{\ell}{L}(m-\varepsilon k)\right\rfloor$ равно наибольшему числу ошибок алгоритма a на наблюдаемой выборке, при котором уклонение частот $\delta(a,X)$ превышает ε.

Замечание 2.2. Лемма 2.1 является аналогом закона больших чисел в слабой аксиоматике, поскольку правая часть (2.3) стремится к нулю при $\ell, k \to \infty$. Известные верхние оценки скорости сходимости в законе больших чисел — неравенства Чебышёва, Хёффдинга, Чернова и др. [13] можно рассматривать как завышенные асимптотические оценки точного равенства (2.3).

Замечание 2.3. В правой части (2.3) стоит величина $m = n(a,X) + n(a,\bar{X})$, зависящая от числа ошибок в скрытой выборке $n(a,\bar{X})$, которое как раз и требуется предсказать, зная число ошибок в наблюдаемой выборке. Этот «порочный круг» разрывается следующим образом. Переформулируем оценку (2.3) в эквивалентном виде: с вероятностью $\eta(\varepsilon) = H_L^{\ell,m}\left(\frac{\ell}{L}(m-\varepsilon k)\right)$ справедливо неравенство $\delta(a,X) \geqslant \varepsilon(\eta)$, где $\varepsilon(\eta)$ — функция, обратная к $\eta(\varepsilon)$. Записав это неравенство как уравнение относительно $n(a,\bar{X})$ и решив его численно, получаем точную верхнюю оценку либо для $n(a,\bar{X})$, либо для уклонения частот $\delta(a,X)$. Аналогичная техника обращения активно использовалась в [12, 11] при получении оценок обобщающей способности, основанных на биномиальном распределении.

3 Предшествующие результаты и эксперименты

Классические оценки Вапника-Червоненкиса [16] также легко переформулируются в слабой аксиоматике. По сути дела, в исходных работах [2, 3, 4, 1] они именно так и выводились[1]. В явном виде переформулировка была сделана в [18]:

$$\begin{aligned} Q_\varepsilon(\mu,\mathbb{X}) &\leqslant \sum_{m=\lceil \varepsilon k\rceil}^{L} \Delta_m(\mu,\mathbb{X})\, H_L^{\ell,m}\left(\tfrac{\ell}{L}(m-\varepsilon k)\right) \leqslant \\ &\leqslant \Delta(\mu,\mathbb{X}) \max_{m=0,\ldots,L} H_L^{\ell,m}\left(\tfrac{\ell}{L}(m-\varepsilon k)\right), \end{aligned} \tag{3.1}$$

[1] Дальнейшее развитие теории статистического обучения пошло по пути отказа от комбинаторной техники вывода оценок в пользу более продвинутого математического аппарата теории вероятностей: изопериметрических неравенств, теории эмпирических процессов, рademахеровской сложности. При этом проблема завышенности оценок «заметалась под ковёр»: количество промежуточных оценок возрастало, и становилось всё труднее проследить, на каком именно из знаков $\leqslant$ происходит наибольшая потеря точности. Эта тенденция хорошо видна по последним наиболее заметным работам, в том числе обзорного характера [17, 7, 11, 6, 14].

где $\Delta(\mu, \mathbb{X})$ — *коэффициент разнообразия* (shattering coefficient), равный числу различных векторов ошибок, порождаемых алгоритмами $a = \mu X$, когда X пробегает всё множество $[\mathbb{X}]^\ell$; $\Delta_m(\mu, \mathbb{X})$ — коэффициент разнообразия m-го слоя — множества алгоритмов $a = \mu X$, допускающих ровно m ошибок на генеральной выборке. Очевидно, коэффициент разнообразия разлагается по слоям: $\Delta(\mu, \mathbb{X}) = \sum_{m=0}^{L} \Delta_m(\mu, \mathbb{X})$.

Величина $\Delta(\mu, \mathbb{X})$ не превосходит числа разбиений C_L^ℓ, но может оказаться строго меньше него по двум причинам. Во-первых, метод μ может строить по различным подвыборкам X совпадающие алгоритмы. Во-вторых, различные алгоритмы могут порождать совпадающие векторы ошибок. Легко показать, что оценка (3.1) нетривиальна (меньше единицы) лишь в тех случаях, когда совпадений образуется достаточно много и $\Delta(\mu, \mathbb{X})$ становится существенно меньше числа разбиений.

Известно, что на практике оценка (3.1) сильно завышена. Основной причиной завышенности принято считать применение *неравенства Буля* (union bound) при выводе этой оценки. Большинство оценок, различными способами улучшающих данный результат, также не обходятся без неравенства Буля [17, 11, 9, 14].

Эксперименты [18] на семи реальных задачах классификации из репозитория UCI позволили количественно оценить основные факторы завышенности. Значения функционала Q_ε и некоторых его верхних оценок, взятых с промежуточных шагов вывода оценки (3.1), измерялись методом Монте-Карло по случайному подмножеству разбиений. В качестве метода обучения μ использовался алгоритм индукции логических закономерностей (rule induction). Были выделены четыре фактора завышенности, два из которых оказались наиболее существенными — пренебрежение эффектами расслоения и сходства алгоритмов.

Эффект расслоения. В практических ситуациях множество алгоритмов, как правило, *расслаивается* по уровням частоты ошибок $\nu(a, \mathbb{X})$, причём основная масса алгоритмов концентрируется в области наихудших частот (около 50%). Лишь малая доля алгоритмов имеют низкие частоты и, соответственно, высокие шансы быть полученными в результате обучения. Остальные алгоритмы семейства фактически не задействуются. Это связано с универсальностью применяемых семейств алгоритмов. Для решения конкретной задачи с фиксированной функцией ошибки I и выборкой $\mathbb{X}$ подходит лишь малая доля алгоритмов из A. Подавляющее большинство алгоритмов «предназначены» для других задач, и в конкретной задаче практически не задействуются методом обучения μ. В то же время, понятие VC-размерности и другие распространённые меры сложности основаны на подсчёте числа всех алгоритмов в семействе (точнее, числа попарно различных векторов ошибок), без учёта вероятностей их получения. Эксперименты [18] показали, что пренебрежение эффектом расслоения может ухудшать оценку Q_ε в 10^2–10^5 раз.

Эффект сходства. На практике часто применяются *связные семейства* алгоритмов, в которых для каждого алгоритма $a \in A$ найдутся другие алгоритмы $a' \in A$ такие, что векторы ошибок алгоритмов a и a' отличаются только на одном объекте [15]. Связные семейства порождаются методами классификации с непрерывной по параметрам разделяющей поверхностью. Это линейные классификаторы, машины

опорных векторов с непрерывными ядрами, нейронные сети с непрерывными функциями активации, решающие деревья с пороговыми условиями ветвления, и многие другие. Чем больше в семействе схожих алгоритмов, тем сильнее завышено неравенство Буля, используемое при выводе VC-оценки (3.1). Эксперименты [18] показали, что пренебрежение эффектом сходства может ухудшать оценку Q_ε в 10^3–10^4 раз.

Остальные факторы вместе ухудшали оценку Q_ε в 10^1–10^2 раз.

Явление расслоения и связанные с ним оценки переобучения (shell bounds) изучались в [12, 11]. Они существенно точнее VC-оценок, но всё ещё сильно завышены. Для их вычисления приходится строить эмпирическое распределение алгоритмов по уровням ошибок, что требует ресурсоёмкого стохастического моделирования методом Монте-Карло. Другой подход связан с введением *алгоритмической функции везения* (algorithmic luckiness function), с помощью которой все алгоритмы семейства ранжируются по их предпочтительности относительно заданной выборки; затем, следуя VC-теории, применяется неравенство Буля и оцениваются мощности покрытия (covering numbers) [9], что также ведёт к завышенности. Влияние сходства алгоритмов на вероятность переобучения почти не изучалось, за исключением [5, 15], где также не удалось добиться радикального улучшения оценок. Все эти результаты свидетельствуют о том, что учёт расслоения и сходства по отдельности не даёт существенного выигрыша в точности оценок.

В экспериментах [19] влияние расслоения и сходства на вероятность переобучения удалось оценить как совместно, так и по отдельности. Для этого рассматривалась *цепочка алгоритмов* — последовательность векторов ошибок, в которой каждый последующий вектор отличается от предыдущего только на одном объекте. Цепочка является простейшим частным случаем связного семейства алгоритмов. Цепочки могут порождаться, в частности, при непрерывном изменении одного из параметров, или всего вектора параметров вдоль некоторой непрерывной траектории.

Эксперименты [19] проводились на модельных цепочках двух типов. Генерация *цепочки с расслоением* начинается с вектора ошибок $\vec{a}_0$ с заданным числом ошибок $m = n(a_0, \mathbb{X})$. Каждый следующий вектор ошибок $\vec{a}_d$, $d = 1, \ldots, D$, генерируется из $\vec{a}_{d-1}$ путём инверсии одной случайно выбранной координаты. В *цепочке без расслоения* число ошибок $n(a_d, \mathbb{X})$, чередуясь, принимает значения m и $m + 1$.

Для каждой цепочки строилась соответствующая ей *не-цепочка* из векторов $\vec{a}'_d$ с таким же числом ошибок, $n(a'_d, \mathbb{X}) = n(a_d, \mathbb{X})$, но случайным образом перепутанными координатами; тем самым разрушалась связность цепочки.

Итого, строилось четыре последовательности векторов ошибок с одинаковыми параметрами D и m. Их сопоставление позволило раздельно оценить влияние связности и расслоения на вероятность переобучения.

Оценки Q_ε вычислялись методом Монте-Карло по 1000 случайных разбиений при $\ell = k = 100$, $m \in \{10, 50\}$, $\varepsilon = 0.05$. Зависимости Q_ε от длины цепочки D показаны на рис. 1. Видно, что связность понижает темп роста этой зависимости, а расслоение опускает уровень горизонтальной асимптоты, особенно для «лёгкой задачи» (левый график). Только при наличии у семейства обоих свойств достигаются приемлемые численные значения вероятности переобучения. Для «трудной задачи»

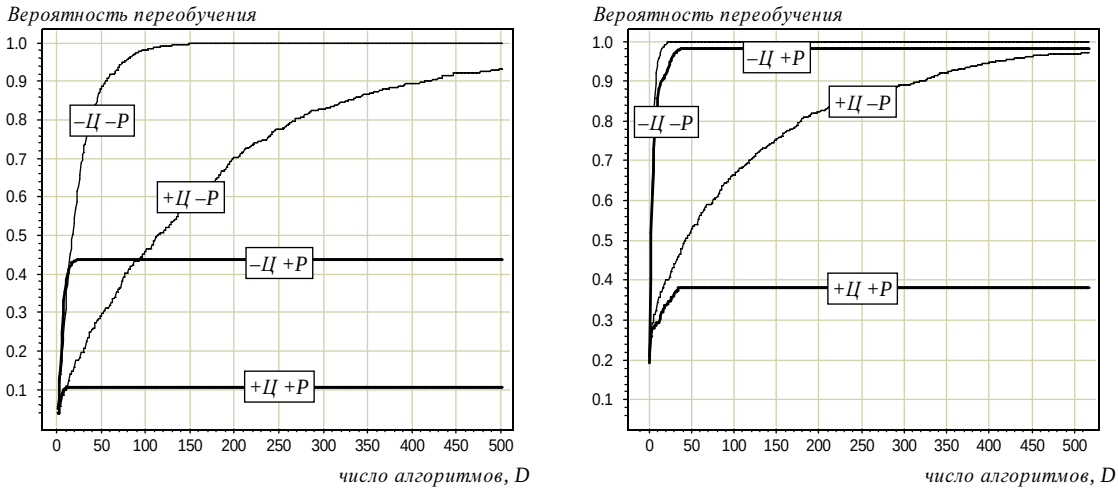

Рис. 1. Зависимость вероятности переобучения Q_ε от числа алгоритмов D при $m = 10$ («лёгкая задача», левый график) и $m = 50$ («трудная задача», правый график). Условные обозначения: $+Ц$ — наличие цепочки, $-Ц$ — отсутствие цепочки, $+Р$ — наличие расслоения, $-Р$ — отсутствие расслоения.

(правый график) свойство расслоения в отдельности уже практически не влияет на оценку, однако в совокупности со свойством связности, опять-таки, приводит к существенному уменьшению Q_ε. Заметим, что согласно VC-теории зависимость $Q_\varepsilon(D)$ линейно возрастает и вообще не имеет горизонтальной асимптоты.

Известные в теории статистического обучения подходы не дают точных оценок вероятности переобучения для цепочек с расслоением. В то же время, такие оценки могут быть получены комбинаторными методами. Их обобщение и привело к основному результату, описанному в следующем разделе.

4 Общие оценки вероятности переобучения

В данной работе приводятся точные оценки вероятности переобучения, основанные на том, что для каждого алгоритма $a \in A$ можно в явном виде записать условия, при которых данный алгоритм является результатом обучения: $\mu(X) = a$.

Будем полагать, что A — конечное множество, и все алгоритмы имеют попарно различные векторы ошибок.

Гипотеза 4.1. *Пусть множество A, выборка $\mathbb{X}$ и метод μ таковы, что для каждого алгоритма $a \in A$ можно указать пару непересекающихся подмножеств $X_a \subset \mathbb{X}$ и $X'_a \subset \mathbb{X}$, удовлетворяющую условию*

$$\big[\mu X{=}a\big] = \big[X_a \subseteq X\big]\big[X'_a \subseteq \bar{X}\big], \quad \forall X \in [\mathbb{X}]^\ell. \tag{4.1}$$

Множество X_a будем называть *порождающим*, множество X'_a — *запрещающим* алгоритм a. Гипотеза 4.1 означает, что метод μ выбирает алгоритм a тогда и только тогда, когда в обучающей выборке X находятся все порождающие объекты и ни

одного запрещающего. Все остальные объекты $\mathbb{X}\backslash X_a\backslash X'_a$ будем называть *нейтральными* для алгоритма a. Наличие или отсутствие нейтральных объектов в обучающей выборке не влияет на результат обучения. В следующих разделах будут приведены нетривиальные примеры семейств, для которых гипотеза 4.1 выполняется.

Лемма 4.1. *Для любой выборки X справедливо тождество*

$$\sum_{a\in A}\big[\mu X{=}a\big] = 1. \tag{4.2}$$

Доказательство с очевидностью вытекает из того, что для любой выборки X метод μ выбирает один и только один алгоритм. Тождество (4.2) может использоваться для проверки того, что условия в правой части (4.1) сформулированы корректно.

Для произвольного $a \in A$ обозначим через L_a число нейтральных объектов, через ℓ_a — число нейтральных объектов, попадающих в обучающую выборку:

$$L_a = L - |X_a| - |X'_a|;$$
$$\ell_a = \ell - |X_a|.$$

Лемма 4.2. *Если гипотеза 4.1 справедлива, то вероятность получить в результате обучения алгоритм a равна*

$$P(a) = \mathsf{P}\big[\mu X{=}a\big] = \frac{C^{\ell_a}_{L_a}}{C^{\ell}_{L}}.$$

Доказательство. Согласно гипотезе 4.1

$$\mathsf{P}\big[\mu X{=}a\big] = \mathsf{P}\big[X_a \subseteq X\big]\big[X'_a \subseteq \bar{X}\big].$$

Это есть доля разбиений генеральной выборки $\mathbb{X} = X \sqcup \bar{X}$ таких, что множество объектов X_a целиком лежит в X, а множество объектов X'_a целиком лежит в $\bar{X}$. Число таких разбиений равно числу способов отобрать ℓ_a из L_a нейтральных объектов в обучающую подвыборку $X\backslash X_a$, которое, очевидно, равно $C^{\ell_a}_{L_a}$. Общее число разбиений равно C^{ℓ}_{L}, а их отношение как раз и есть $P(a)$. ■

Вероятность переобучения Q_ε выражается по формуле полной вероятности, если для каждого алгоритма a из A известна вероятность $P(a)$ получить его в результате обучения и условная вероятность большого уклонения частот $\mathsf{P}\big(\delta(a,X) \geqslant \varepsilon \mid a\big)$ при условии, что получен алгоритм a:

$$Q_\varepsilon = \sum_{a\in A} P(a)\,\mathsf{P}\big(\delta(a,X) \geqslant \varepsilon \mid a\big).$$

Условная вероятность даётся леммой 2.1, если учесть, что при фиксированном алгоритме a подмножества X_a и X'_a не участвуют в разбиениях. Рассматривая L_a нейтральных объектов и всевозможные их разбиения на ℓ_a обучающих и $L_a - \ell_a$ контрольных, получим:

$$\mathsf{P}\big(\delta(a,X) \geqslant \varepsilon \mid a\big) = H^{\ell_a, m_a}_{L_a}\left(s_a(\varepsilon)\right),$$

где m_a — число ошибок алгоритма a на нейтральных объектах; $s_a(\varepsilon)$ — наибольшее число ошибок алгоритма a на нейтральных обучающих объектах $X\setminus X_a$, при котором имеет место большое уклонение частот ошибок, $\delta(a,X) \geqslant \varepsilon$:

$$m_a = n(a, \mathbb{X}\setminus X_a\setminus X'_a);$$
$$s_a(\varepsilon) = \tfrac{\ell}{L}\big(n(a,\mathbb{X}) - \varepsilon k\big) - n(a, X_a).$$

Более строгий комбинаторный вывод точной оценки Q_ε представлен ниже.

Теорема 4.3. *Если гипотеза 4.1 справедлива, то вероятность переобучения вычисляется по формуле*

$$Q_\varepsilon = \sum_{a\in A} P(a) H_{L_a}^{\ell_a, m_a}\left(s_a(\varepsilon)\right).$$

Доказательство. Рассмотрим функционал Q_ε. Введём в (2.1) под знак суммирования по X ещё два вспомогательных суммирования: первый — по всем алгоритмам a из A при условии $\mu X = a$, второй — по всем значениям s числа ошибок алгоритма a на подвыборке $X\setminus X_a$. Очевидно, значение Q_ε от этого не изменится:

$$Q_\varepsilon = \mathsf{P}\big[\delta_\mu(X) \geqslant \varepsilon\big] = \mathsf{P}\sum_{a\in A}\big[\mu X{=}a\big]\sum_{s=0}^{\ell_a}\big[n(a, X\setminus X_a) = s\big]\big[\delta(a,X)\geqslant\varepsilon\big]. \tag{4.3}$$

Число ошибок алгоритма a на обучающей подвыборке X равно $s + n(a, X_a)$, поэтому уклонение частот ошибок выражается в виде

$$\delta(a,X) = \frac{n(a,\mathbb{X}) - s - n(a,X_a)}{k} - \frac{s + n(a,X_a)}{\ell},$$

следовательно,

$$\big[\delta(a,X)\geqslant\varepsilon\big] = \big[s \leqslant \tfrac{\ell}{L}\big(n(a,\mathbb{X}) - \varepsilon k\big) - n(a,X_a)\big] = \big[s\leqslant s_a(\varepsilon)\big].$$

Подставим полученное выражение в (4.3), затем заменим $\big[\mu X = a\big]$ правой частью равенства (4.1) и переставим знаки суммирования (очевидно, P также можно рассматривать как суммирование):

$$Q_\varepsilon = \sum_{a\in A}\sum_{s=0}^{\ell_a} \underbrace{\mathsf{P}\big[X_a\subseteq X\big]\big[X'_a\subseteq \bar{X}\big]\big[n(a,X\setminus X_a) = s\big]}_{N(a)}\big[s\leqslant s_a(\varepsilon)\big]. \tag{4.4}$$

Выделенное в данной формуле выражение $N(a)$ есть доля разбиений генеральной выборки $\mathbb{X} = X\sqcup\bar{X}$ таких, что множество объектов X_a целиком лежит в X, множество объектов X'_a целиком лежит в $\bar{X}$, и в подвыборку $X\setminus X_a$ длины ℓ_a попадает ровно s объектов, на которых алгоритм a допускает ошибку.

Для наглядности представим вектор ошибок a разбитым на шесть блоков:

$$a = \Big(\underbrace{X_a; \overbrace{1,\ldots,1}^{s}; \underbrace{0,\ldots,0}_{X\setminus X_a}}_{X}; \underbrace{X'_a; \overbrace{1,\ldots,1}^{m_a-s}; \underbrace{0,\ldots,0}_{\bar{X}\setminus X'_a}}_{\bar{X}}\Big).$$

Число ошибок алгоритма a на объектах, не попадающих ни в X_a, ни в X'_a, равно m_a. Существует $C^s_{m_a}$ способов выбрать из них s объектов, которые попадут в $X\backslash X_a$. Для каждого из этих способов имеется ровно $C^{\ell_a-s}_{L_a-m_a}$ способов выбрать $\ell_a - s$ объектов, на которых алгоритм a не допускает ошибку, и которые также попадут в $X\backslash X_a$. Тем самым однозначно определяется состав выборки $X\backslash X_a$, а, значит, и состав выборки $\bar{X}\backslash X'_a$. Таким образом, $N(a) = C^s_{m_a} C^{\ell_a-s}_{L_a-m_a}/C^\ell_L$. Подставим это выражение в (4.4) и выделим в нём формулу гипергеометрической функции вероятности:

$$Q_\varepsilon = \sum_{a\in A} \frac{C^{\ell_a}_{L_a}}{C^\ell_L} \sum_{s=s_0}^{\ell_a} \big[s \leqslant s_a(\varepsilon)\big] \frac{C^s_{m_a} C^{\ell_a-s}_{L_a-m_a}}{C^{\ell_a}_{L_a}} = \sum_{a\in A} P(a) H^{\ell_a,m_a}_{L_a}\left(s_a(\varepsilon)\right).$$

Теорема доказана. ■

Гипотеза 4.1 накладывает слишком сильные ограничения на выборку $\mathbb{X}$, семейство A и метод μ. Поэтому теорему 4.3 удаётся применять лишь в некоторых специальных случаях. Рассмотрим естественное обобщение гипотезы 4.1. Предположим, что для каждого алгоритма a существуют различные варианты выделения порождающих и запрещающих множеств.

Гипотеза 4.2. *Пусть множество A, выборка $\mathbb{X}$ и метод μ таковы, что для каждого алгоритма $a \in A$ можно указать конечное множество индексов V_a, и для каждого индекса $v \in V_a$ можно указать порождающее множество $X_{av} \subset \mathbb{X}$, запрещающее множество $X'_{av} \subset \mathbb{X}$ и коэффициент $c_{av} \in \mathbb{R}$, удовлетворяющие условиям*

$$\big[\mu X{=}a\big] = \sum_{v\in V_a} c_{av} \big[X_{av} \subseteq X\big] \big[X'_{av} \subseteq \bar{X}\big], \quad \forall X \in [\mathbb{X}]^\ell. \tag{4.5}$$

Гипотеза 4.1 является частным случаем гипотезы 4.2, когда все множества V_a одноэлементные и $c_{av} = 1$. Примеры задач, удовлетворяющих гипотезам 4.1 или 4.2, будут рассмотрены в следующих параграфах.

Следующая теорема утверждает, что гипотеза 4.2 верна всегда.

Теорема 4.4. *Для любых $\mathbb{X}$, A и μ существуют множества V_a, X_{av}, X'_{av}, при которых справедливо представление* (4.5)*, причём $c_{av} = 1$ для всех $a \in A$, $v \in V_a$.*

Доказательство. Зафиксируем произвольный алгоритм $a \in A$. Возьмём в качестве индексного множества V_a множество всех подвыборок $v \in [\mathbb{X}]^\ell$, при которых $\mu v = a$. Для каждого $v \in V_a$ положим $X_{av} = v$, $X'_{av} = \mathbb{X} \setminus v$, $c_{av} = 1$. Тогда для любого $X \in [\mathbb{X}]^\ell$ справедливо представление, имеющее вид (4.5):

$$\big[\mu X{=}a\big] = \sum_{v\in V_a} \big[v{=}X\big] = \sum_{v\in V_a} \big[v{=}X\big] \big[\mathbb{X}\backslash v{=}\mathbb{X}\backslash X\big] = \sum_{v\in V_a} \big[v \subseteq X\big] \big[\mathbb{X}\backslash v \subseteq \bar{X}\big],$$

причём, если $\mu X = a$, то ровно одно слагаемое в этой сумме равно единице, остальные равны нулю; если же $\mu X \neq a$, то все слагаемые равны нулю. ■

Теорема 4.4 является типичной теоремой существования. Использованный при её доказательстве способ построения индексных множеств V_a требует явного перебора всех разбиений выборки, что приводит к вычислительно неэффективным оценкам

вероятности переобучения. Однако представление (4.5) в общем случае не единственно. Отдельной проблемой является поиск такого представления, в котором мощности множеств $|V_a|$, $|X_{av}|$, $|X'_{av}|$ были бы как можно меньше. Эффективные вычислительные алгоритмы, эксплуатирующие свойства расслоения и сходства в семействах алгоритмов, рассматриваются в следующих параграфах.

Хотя гипотеза 4.2 верна всегда, мы будем продолжать называть её «гипотезой», имея в виду предположение, что существует некоторое, более эффективное, представление вида (4.5), отличное от использованного в теореме 4.4.

Введём для каждого алгоритма $a \in A$ и каждого индекса $v \in V_a$ обозначения:

$$\begin{aligned} L_{av} &= L - |X_{av}| - |X'_{av}|; \\ \ell_{av} &= \ell - |X_{av}|; \\ m_{av} &= n(a,\mathbb{X}) - n(a,X_{av}) - n(a,X'_{av}); \\ s_{av}(\varepsilon) &= \tfrac{\ell}{L}\big(n(a,\mathbb{X}) - \varepsilon k\big) - n(a,X_{av}). \end{aligned}$$

В условиях гипотезы 4.2 справедливы соответствующие обобщения леммы о вероятностях получения алгоритмов и теоремы о вероятности переобучения.

Лемма 4.5. *Если гипотеза 4.2 справедлива, то для всех $a \in A$ вероятность получить в результате обучения алгоритм a равна*

$$P(a) = \mathsf{P}\big[\mu X{=}a\big] = \sum_{v \in V_a} c_{av} P_{av}; \tag{4.6}$$

$$P_{av} = \mathsf{P}\big[X_{av} \subseteq X\big]\big[X'_{av} \subseteq \bar{X}\big] = \frac{C^{\ell_{av}}_{L_{av}}}{C^{\ell}_{L}}. \tag{4.7}$$

Доказательство. Достаточно применить операцию P к левой и правой частям (4.5). Дальнейшие рассуждения аналогичны доказательству леммы 4.2. ■

Теорема 4.6. *Если гипотеза 4.2 справедлива, то вероятность переобучения вычисляется по формуле*

$$Q_\varepsilon = \sum_{a \in A} \sum_{v \in V_a} c_{av} P_{av} H^{\ell_{av}, m_{av}}_{L_{av}}\left(s_{av}(\varepsilon)\right). \tag{4.8}$$

Доказательство. Аналогично доказательству теоремы 4.3, вероятность переобучения приводится к выражению, которое отличается от (4.4) появлением знака суммирования по v, коэффициентов c_{av} и двойных индексов ${}_{av}$ вместо одинарных ${}_a$:

$$Q_\varepsilon = \sum_{a \in A} \sum_{v \in V_a} \sum_{s=0}^{\ell} c_{av}\, \mathsf{P}\big[X_{av} \subseteq X\big]\big[X'_{av} \subseteq \bar{X}\big]\big[n(a, X \backslash X_{av}) = s\big]\big[s \leqslant s_{av}(\varepsilon)\big],$$

В остальном доказательство аналогично доказательству теоремы 4.3. ■

Рассмотрим один частный случай, когда формула (4.8) существенно упрощается.

Теорема 4.7. *Пусть гипотеза 4.2 справедлива, μ — метод минимизации эмпирического риска, множество A содержит алгоритм, корректный на генеральной выборке $\mathbb{X}$. Тогда вероятность переобучения принимает более простой вид:*

$$Q_\varepsilon = \sum_{a \in A} \big[n(a, \mathbb{X}) \geqslant \varepsilon k\big] P(a). \tag{4.9}$$

Доказательство. Рассмотрим произвольный алгоритм $a \in A$ и произвольный индекс $v \in V_a$. Если некоторый объект, на котором a допускает ошибку, содержится в обучающей выборке X, то метод μ не сможет выбрать данный алгоритм, так как существует корректный алгоритм, не допускающий ошибок на X. Следовательно, множество объектов, на которых алгоритм a допускает ошибку, целиком содержится в X'_{av}. Значит, алгоритм a не допускает ошибок на нейтральных объектах и $m_{av} = 0$. В этом случае гипергеометрическая функция $H^{\ell_{av},0}_{L_{av}}(s_{av}(\varepsilon))$ является вырожденной: при $s_{av}(\varepsilon) \geqslant 0$ она представляет собой сумму из одного слагаемого, равного 1; при $s_{av}(\varepsilon) < 0$ число слагаемых равно нулю и вся сумма равна нулю:

$$H^{\ell_{av},0}_{L_{av}}(s_{av}(\varepsilon)) = \big[s_{av}(\varepsilon) \geqslant 0\big] = \big[\tfrac{\ell}{L}\big(n(a, \mathbb{X}) - \varepsilon k\big) - n(a, X_{av}) \geqslant 0\big] = \big[n(a, \mathbb{X}) \geqslant \varepsilon k\big].$$

Подставляя это выражение в (4.8), получаем (4.9). ■

5 Блочное вычисление вероятности переобучения

Допустим, что векторы ошибок всех алгоритмов из множества A известны и попарно различны.

Опр. 5.1. *Метод минимизации эмпирического риска μ будем называть пессимистичным, если в случаях неоднозначности, когда минимум в* (1.1) *достигается на нескольких алгоритмах, выбирается алгоритм с бóльшим числом ошибок на контрольной выборке $\bar{X}$. Если же и таких алгоритмов несколько, то предполагается, что метод μ выбирает алгоритм с бóльшим порядковым номером.*

Пессимистичная минимизация эмпирического риска на практике нереализуема, так как метод μ не может использовать контрольную выборку. Теоретически этот «наихудший случай» интересен тем, что он даёт верхние оценки Q_ε. При любых других стратегиях устранения неоднозначности вероятность переобучения Q_ε может оказаться только меньше.

Пусть множество A состоит из D алгоритмов, $A = \{a_1, \ldots, a_D\}$. Значения $I(a_d, x_i)$ образуют бинарную $L \times D$-матрицу ошибок, столбцы которой являются векторами ошибок алгоритмов, а строки соответствуют объектам. Обозначим через $b = (b_1, \ldots, b_D)$ произвольный бинарный вектор размерности D. Выборка $\mathbb{X}$ разбивается на непересекающиеся *блоки* $U_b \subseteq \mathbb{X}$ так, что всем объектам в блоке соответствует одна и та же строка b в матрице ошибок:

$$U_b = \big\{x_i \in \mathbb{X} \;\big|\; I(a_d, x_i) = b_d,\ d = 1, \ldots, D\big\}.$$

Обозначим через B множество бинарных векторов b, которым соответствуют непустые блоки U_b. Очевидно, $|B| \leqslant \min\{L, 2^D\}$. Обозначим $m_b = |U_b|$.

Каждой обучающей выборке $X \in [\mathbb{X}]^\ell$ поставим в соответствие целочисленный вектор $(s_b)_{b\in B}$ такой, что $s_b = |X \cap U_b|$ — число объектов из блока U_b, попадающих в обучающую выборку. Множество всех таких векторов, соответствующих всевозможным обучающим выборкам, обозначим через S. Очевидно, S можно также определить и другим способом:

$$S = \Big\{ s = (s_b)_{b\in B} \;\Big|\; s_b = 0, \ldots, m_b,\ \sum_{b\in B} s_b = \ell \Big\}.$$

Запишем число ошибок алгоритма a_d на выборке X в виде суммы по блокам:

$$n(a_d, X) = \sum_{b\in B} b_d \, |X \cap U_b| = \sum_{b\in B} b_d s_b.$$

Таким образом, выбор алгоритма методом μ зависит только от того, сколько объектов s_b из каждого блока попадёт в обучающую выборку, но не зависит от того, какие именно это будут объекты. Определим функцию $d^*\colon S \to \{1, \ldots, D\}$ как номер алгоритма, выбранного методом μ по обучающей выборке:

$$d^*(s) = \max\Big(\operatorname*{Arg\,min}_{d=1,\ldots,D} \sum_{b\in B} b_d s_b \Big), \tag{5.1}$$

где через $\operatorname*{Arg\,min}_{d=1,\ldots,D} f(d)$ обозначается множество значений d, при которых функция $f(d)$ достигает минимального значения.

Теорема 5.1. *Пусть μ — пессимистичная минимизация эмпирического риска, векторы ошибок всех алгоритмов $a \in A$ попарно различны. Тогда вероятность получить алгоритм a_d в результате обучения:*

$$\mathsf{P}\big[\mu X{=}a_d\big] = \frac{1}{C_L^\ell} \sum_{s\in S} \Big(\prod_{b\in B} C_{m_b}^{s_b} \Big) \big[\, d^*(s){=}d \,\big]; \tag{5.2}$$

вероятность переобучения:

$$Q_\varepsilon = \frac{1}{C_L^\ell} \sum_{s\in S} \Big(\prod_{b\in B} C_{m_b}^{s_b} \Big) \Big[\sum_{b\in B} b_{d^*(s)} \big(m_b\ell - s_b L \big) \geqslant \varepsilon k \ell \Big]. \tag{5.3}$$

Доказательство.

Произвольному набору значений $(s_b)_{b\in B}$ из S соответствует множество выборок $X \in [\mathbb{X}]^\ell$ таких, что $|X \cap U_b| = s_b$. Число таких выборок равно произведению $\prod_{b\in B} C_{m_b}^{s_b}$, так как для каждого блока U_b существует $C_{m_b}^{s_b}$ способов отобрать s_b объектов в подвыборку $X \cap U_b$.

Поскольку условия $\mu X = a_d$ и $d^*(s) = d$ равносильны, вероятность получить алгоритм a_d в результате обучения выражается в следующем виде:

$$\mathsf{P}\big[\mu X{=}a_d\big] = \frac{1}{C_L^\ell} \sum_{X\in[\mathbb{X}]^\ell} \big[\mu X{=}a_d\big] =$$
$$= \frac{1}{C_L^\ell} \sum_{X\in[\mathbb{X}]^\ell} \big[d^*(s){=}d\big] = \frac{1}{C_L^\ell} \sum_{s\in S} \Big(\prod_{b\in B} C_{m_b}^{s_b}\Big) \big[\, d^*(s){=}d\,\big].$$

Теперь запишем вероятность переобучения:

$$Q_\varepsilon = \mathsf{P}\big[\delta_\mu(X) \geqslant \varepsilon\big] = \mathsf{P} \sum_{d=1}^{D} \big[\mu X{=}a_d\big]\big[\delta(a_d, X) \geqslant \varepsilon\big].$$

Распишем уклонение частот ошибок алгоритма a_d в виде суммы по блокам:

$$\delta(a_d, X) = \frac{1}{k} \sum_{b\in B} b_d(m_b - s_b) - \frac{1}{\ell} \sum_{b\in B} b_d s_b = \frac{1}{\ell k} \sum_{b\in B} b_d\big(m_b\ell - s_b L\big).$$

Тогда выражение для вероятности переобучения примет вид:

$$Q_\varepsilon = \frac{1}{C_L^\ell} \sum_{s\in S} \Big(\prod_{b\in B} C_{m_b}^{s_b}\Big) \sum_{d=1}^{D} \big[\, d^*(s){=}d\,\big] \Big[\sum_{b\in B} b_d\big(m_b\ell - s_b L\big) \geqslant \varepsilon\ell k\Big].$$

Отсюда немедленно вытекает требуемое выражение (5.3). ■

Замечание 5.1. Если множество B содержит векторы b, соответствующие пустым блокам U_b, то формулы (5.2) и (5.3) остаются в силе, поскольку тогда $m_b = s_b = 0$.

Теорема 5.1 позволяет, зная векторы ошибок всех алгоритмов, выписать вероятности получения каждого из алгоритмов и вероятность переобучения Q_ε. Однако непосредственные вычисления по формулам (5.2) и (5.3) могут потребовать затрат времени, экспоненциальных по длине выборки L. В худшем случае, когда все блоки U_b одноэлементные, множество S состоит из всевозможных булевых векторов длины L, содержащих ровно ℓ единиц. При этом число слагаемых в (5.2) и (5.3) равно C_L^ℓ.

Вычисления по теореме 5.1 могут быть достаточно эффективны только когда число блоков $|B|$ невелико, в частности, при малом числе алгоритмов.

Рассмотрим в качестве примера «экстремальный» частный случай, когда семейство состоит только из двух алгоритмов, $A = \{a_1, a_2\}$.

Положим $B = (11, 10, 01, 00)$.

Пусть в выборке $\mathbb{X}$ имеется m_{11} объектов, на которых оба алгоритма допускают ошибку; m_{10} объектов, на которых только a_1 допускает ошибку; m_{01} объектов, на которых только a_2 допускает ошибку; $m_{00} = L - m_{11} - m_{10} - m_{01}$ объектов, на которых оба алгоритма дают верный ответ:

$$a_1 = (\,1,\ldots,1,\ 1,\ldots,1,\ 0,\ldots,0,\ 0,\ldots,0\,);$$
$$a_2 = (\underbrace{1,\ldots,1}_{m_{11}},\ \underbrace{0,\ldots,0}_{m_{10}},\ \underbrace{1,\ldots,1}_{m_{01}},\ \underbrace{0,\ldots,0}_{m_{00}}).$$

Теорема 5.2. *Пусть μ — пессимистичная минимизация эмпирического риска, и семейство состоит из двух алгоритмов $A = \{a_1, a_2\}$. Тогда при любом $\varepsilon \in [0,1)$ справедлива точная оценка:*

$$Q_\varepsilon = \sum_{s_{11}=0}^{m_{11}} \sum_{s_{10}=0}^{m_{10}} \sum_{s_{01}=0}^{m_{01}} \frac{C_{m_{11}}^{s_{11}} C_{m_{10}}^{s_{10}} C_{m_{01}}^{s_{01}} C_{L-m_{11}-m_{10}-m_{01}}^{\ell-s_{11}-s_{10}-s_{01}}}{C_L^\ell} \times$$
$$\times \Big(\big[s_{10} < s_{01}\big] \big[s_{11} + s_{10} \leqslant \tfrac{\ell}{L}(m_{11} + m_{10} - \varepsilon k)\big] +$$
$$+ \big[s_{10} \geqslant s_{01}\big] \big[s_{11} + s_{01} \leqslant \tfrac{\ell}{L}(m_{11} + m_{01} - \varepsilon k)\big] \Big). \tag{5.4}$$

Доказательство. Воспользуемся теоремой 5.1.

Множество S состоит из целочисленных векторов $s = (s_{11}, s_{10}, s_{01}, s_{00})$, для которых $s_{11} + s_{10} + s_{01} + s_{00} = \ell$. Поэтому сумма $\sum_{s \in S}$ преобразуется в тройную сумму $\sum_{s_{11}=0}^{m_{11}} \sum_{s_{10}=0}^{m_{10}} \sum_{s_{01}=0}^{m_{01}}$, при этом s_{00} выражается через остальные компоненты вектора s.

Номер $d^*(s)$ алгоритма, выбранного методом μ по обучающей выборке, равен 1 при $s_{10} < s_{01}$ и 2 при $s_{10} \geqslant s_{01}$.

Теперь подставим значения m_b, s_b, $d^*(s)$ в (5.3):

$$\Big[\sum_{b \in B} b_{d^*(s)} \big(m_b \ell - s_b L\big) \geqslant \varepsilon \ell k \Big] = \big[d^*(s) = 1\big] \big[(m_{10} + m_{11})\ell - (s_{10} - s_{11})L \geqslant \varepsilon \ell k\big] +$$
$$+ \big[d^*(s) = 2\big] \big[(m_{01} + m_{11})\ell - (s_{01} - s_{11})L \geqslant \varepsilon \ell k\big].$$

Отсюда следует требуемое выражение (10.1). ■

Точная оценка вероятности переобучения для двухэлементного семейства была представлена в [19] без доказательства. Результаты численных экспериментов показывают, что уже в этом простейшем случае возникает переобучение и проявляются эффекты расслоения и сходства, снижающие вероятность переобучения.

6 Рекуррентное вычисление вероятности переобучения

Допустим, что векторы ошибок всех алгоритмов из множества A известны, попарно различны, и в A есть корректный на $\mathbb{X}$ алгоритм. Пусть μ — пессимистичный метод минимизации эмпирического риска. Задача заключается в том, чтобы для каждого алгоритма $a \in A$ найти всю информацию, необходимую для вычисления вероятности переобучения Q_ε по теореме 4.6:

$$\mathfrak{I}(a) = \langle X_{av}, X'_{av}, c_{av} \rangle_{v \in V_a},$$

где V_a — индексное множество, X_{av} — множество порождающих объектов, X'_{av} — множество запрещающих объектов, $c_{av} \in \mathbb{R}$.

Перенумеруем алгоритмы в порядке неубывания $n(a, \mathbb{X})$ — числа ошибок на полной выборке: $A = \{a_0, \ldots, a_D\}$. Очевидно, $n(a_0, \mathbb{X}) = 0$.

Обозначим через μ_d метод обучения, удовлетворяющий определению 5.1 и выбирающий алгоритмы только из подмножества $A_d = \{a_0, \ldots, a_d\}$. Рассмотрим процедуру последовательного добавления алгоритмов, на каждом шаге которой осуществляется переход от метода μ_{d-1} к методу μ_d.

Допустим, что для всех алгоритмов a_t, $t < d$, информация $\mathfrak{I}(a_t)$ относительно метода μ_{d-1} уже вычислена. Зададимся целью вычислить информацию $\mathfrak{I}(a_d)$ и скорректировать информацию $\mathfrak{I}(a_t)$, $t < d$, относительно метода μ_d. Заметим, что такая коррекция в общем случае необходима, поскольку алгоритм a_d может отбирать некоторые разбиения у каждого из предыдущих алгоритмов a_t.

Лемма 6.1. *Метод μ_d выбирает алгоритм a_d тогда и только тогда, когда все объекты, на которых a_d допускает ошибку, попадают в контрольную выборку:*

$$\big[\mu_d X{=}a_d\big] = \big[X'_d \subseteq \bar{X}\big], \quad X'_d = \big\{x_i \in \mathbb{X}\colon I(a_d, x_i) = 1\big\}.$$

Доказательство. Если хотя бы один объект, на котором a_d допускает ошибку, окажется в обучающей выборке X, то метод μ_d выберет алгоритм с меньшим числом ошибок на X. Такой алгоритм точно есть, например, a_0. Итак, условие $X'_d \subseteq \bar{X}$ является необходимым для того, чтобы метод μ_d выбрал алгоритм a_d. Покажем, что оно является также и достаточным. Для этого достаточно показать, что если алгоритмов из A_d, не допускающих ошибок на обучающей выборке X, окажется несколько, то из них будет выбран именно a_d. В силу упорядоченности множества A_d алгоритм a_d допускает максимальное число ошибок на $\mathbb{X}$, а среди алгоритмов с таким же числом ошибок на $\mathbb{X}$ имеет максимальный номер. Поэтому, согласно определению 5.1, алгоритм a_d будет выбран методом μ_d из A_d всегда, когда $X'_d \subseteq \bar{X}$. ■

Допустим, что непосредственно перед добавлением алгоритма a_d условия выбора каждого из предыдущих алгоритмов a_t были записаны в виде (4.5):

$$\big[\mu_{d-1} X{=}a_t\big] = \sum_{v \in V_t} c_{tv} \underbrace{\big[X_{tv} \subseteq X\big]\big[X'_{tv} \subseteq \bar{X}\big]}_{J_{tv}(d-1)}, \quad t < d.$$

После добавления алгоритма a_d эти условия изменятся. К ним добавится требование, чтобы множество X'_d не лежало целиком в контрольной выборке $\bar{X}$; иначе вместо алгоритма a_t метод μ_d выберет алгоритм a_d:

$$\begin{aligned}\big[\mu_d X{=}a_t\big] &= \big[\mu_{d-1} X{=}a_t\big]\big[X'_d \not\subseteq \bar{X}\big] = \\ &= \sum_{v \in V_t} c_{tv} \underbrace{\big[X_{tv} \subseteq X\big]\big[X'_{tv} \subseteq \bar{X}\big]\big[X'_d \not\subseteq \bar{X}\big]}_{J_{tv}(d)}, \quad t < d. \end{aligned} \tag{6.1}$$

Индукцией по d легко доказывается, что тождество (4.2) выполняется на каждом шаге, то есть что условия (6.1) определены корректно. Действительно, предполагая

$$\sum_{t=0}^{d-1} \big[\mu_{d-1} X{=}a_t\big] = 1,$$

получаем

$$\sum_{t=0}^{d}\big[\mu_d X{=}a_t\big] = \sum_{t=0}^{d-1}\big[\mu_{d-1}X{=}a_t\big]\big[X'_d \not\subseteq \bar{X}\big] + \big[X'_d \subseteq \bar{X}\big] = 1.$$

Чтобы получить правила корректировки информации $\mathfrak{I}(a_t)$, достаточно привести выражение (6.1) к виду (4.5), что и будет проделано в следующей лемме.

Лемма 6.2. *Корректировка информации $\mathfrak{I}(a_t)$, $t < d$ при добавлении алгоритма a_d сводится к проверке для каждого $v \in V_t$ такого, что $X_{tv} \cap X'_d = \varnothing$, трёх условий:*

1) если $X'_d \setminus X'_{tv} = \{x_i\}$ — одноэлементное множество, то x_i присоединяется к X_{tv};

2) если $|X'_d \setminus X'_{tv}| > 1$, то множество индексов V_t пополняется ещё одним элементом (обозначим его w) и полагается $c_{tw} = -c_{tv}$, $X_{tw} = X_{tv}$, $X'_{tw} = X'_{tv} \cup X'_d$;

3) если $|X'_d \setminus X'_{tv}| = 0$, то из множества индексов V_t удаляется индекс v; соответственно, из $\mathfrak{I}(a_t)$ удаляется вся тройка $\langle X_{tv}, X'_{tv}, c_{tv}\rangle$.

Доказательство. Если $X_{tv} \cap X'_d \neq \varnothing$, то из $X_{tv} \subseteq X$ следует, что множество X'_d не лежит целиком в контрольной выборке $\bar{X}$. Следовательно, никакая корректировка для тройки $\langle X_{tv}, X'_{tv}, c_{tv}\rangle$ не нужна:

$$J_{tv}(d) = \big[X_{tv} \subseteq X\big]\big[X'_{tv} \subseteq \bar{X}\big] = J_{tv}(d-1).$$

Если всё-таки $X_{tv} \cap X'_d = \varnothing$, то возможны три случая, в зависимости от мощности множества $X'_d \setminus X'_{tv}$.

Первый случай: $X'_d \setminus X'_{tv} = \{x_i\}$ — одноэлементное множество. Тогда справедлива цепочка равенств $[X'_d \not\subseteq \bar{X}] = [x_i \notin \bar{X}] = [x_i \in X]$. Подставляя в (6.1), получим

$$J_{tv}(d) = \big[X_{tv} \sqcup \{x_i\} \subseteq X\big]\big[X'_{tv} \subseteq \bar{X}\big].$$

Второй случай: $|X'_d \setminus X'_{tv}| > 1$. Тогда

$$\begin{aligned} J_{tv}(d) &= \big[X_{tv} \subseteq X\big]\big[X'_{tv} \subseteq \bar{X}\big]\big(1 - [X'_d \subseteq \bar{X}]\big) = \\ &= J_{tv}(d-1) - \big[X_{tv} \subseteq X\big]\big[X'_{tv} \cup X'_d \subseteq \bar{X}\big]. \end{aligned} \tag{6.2}$$

Таким образом, в выражении для $\big[\mu_d X{=}a_t\big]$ появится ещё одно слагаемое, что равносильно добавлению во множество V_t ещё одного индекса (обозначим его w), для которого полагается $c_{tw} = -c_{tv}$, $X_{tw} = X_{tv}$, $X'_{tw} = X'_{tv} \cup X'_d$.

Наконец, третий случай: $|X'_d \setminus X'_{tv}| = 0$. Тогда представление (6.2) остаётся в силе, однако разность $J_{tv}(d)$ оказывается равной нулю, поскольку $X'_{tv} \cup X'_d = X'_{tv}$. Обнуление $J_{tv}(d)$ равносильно удалению индекса v из множества индексов V_t вместе с удалением соответствующей тройки $\langle X_{tv}, X'_{tv}, c_{tv}\rangle$ из информации $\mathfrak{I}(a_t)$. ■

Леммы 6.1, 6.2 и теорема 4.7 позволяют рекуррентно вычислять вероятность переобучения Q_ε. На каждом d-м шаге, $d = 0, \ldots, D$, добавляется алгоритм a_d, вычисляется информация $\mathfrak{I}(a_d)$; затем для каждого $t = 0, \ldots, d-1$ корректируется информация $\mathfrak{I}(a_t)$ и вероятности P_{tv}. На основе скорректированной информации обновляется текущая оценка Q_ε. По окончании последнего D-го шага текущая оценка Q_ε

Алгоритм 6.1. Рекуррентное вычисление вероятности переобучения

Вход:
матрица ошибок $I(a_d, x_i)$, $d = 0, \ldots, D$, $i = 1, \ldots, L$;

Выход:
информация $\mathfrak{I}(a_d) = \langle X_{dv}, X'_{dv}, c_{dv} \rangle_{v \in V_d}$ для всех $d = 0, \ldots, D$,
вероятность переобучения Q_ε;

1: $Q_\varepsilon := 0$; упорядочить $A = \{a_0, \ldots, a_D\}$ по возрастанию $n(a_d, \mathbb{X})$;
2: **для всех** $d := 1, \ldots, D$
3: добавить алгоритм a_d:
$V_d := \{\varnothing\}$; $X_d := \varnothing$; $X'_d := \{x \in \mathbb{X} \colon I(a_d, x) = 1\}$; $c_d := 1$; $P_d := \prod_{j=0}^{|X'_d|-1} \left(\frac{k-j}{L-j}\right)$;
если $n(a_d, \mathbb{X}) \geqslant \varepsilon k$ **то** $Q_\varepsilon := Q_\varepsilon + P_d$;
4: **для всех** $t := 0, \ldots, d-1$
5: **для всех** $v \in V_t$ таких, что $X_{tv} \cap X'_d = \varnothing$
6: $\Delta := |X'_d \setminus X'_{tv}|$;
7: **если** $\Delta = 1$ **то**
8: скорректировать множество X_{tv} и вероятность P_{tv}:
$X_{tv} := X_{tv} \sqcup (X'_d \setminus X'_{tv})$; $P'_{tv} := P_{tv}$; $P_{tv} := P_{tv}\ell_{tv}/L_{tv}$;
если $n(a_t, \mathbb{X}) \geqslant \varepsilon k$ **то** $Q_\varepsilon := Q_\varepsilon + c_{tv}(P_{tv} - P'_{tv})$;
9: **иначе если** $\Delta > 1$ **то**
10: добавить в V_t новый индекс w;
$X_{tw} := X_{tv}$; $X'_{tw} := X'_{tv} \cup X'_d$; $c_{tw} := -c_{tv}$; $P_{tw} := P_{tv} \prod_{j=0}^{\Delta-1} \left(1 - \frac{\ell_{tv}}{L_{tv}-j}\right)$;
если $n(a_t, \mathbb{X}) \geqslant \varepsilon k$ **то** $Q_\varepsilon := Q_\varepsilon + c_{tw}P_{tw}$;
11: **иначе**
12: удалить из V_t индекс v;
если $n(a_t, \mathbb{X}) \geqslant \varepsilon k$ **то** $Q_\varepsilon := Q_\varepsilon - c_{tv}P_{tv}$;

должна совпадать с точным значением вероятности переобучения. Эта рекуррентная вычислительная процедура подробно записана в виде псевдокода Алгоритма 6.1.

Некоторые обозначения, использованные в Алгоритме 6.1, требуют дополнительных пояснений.

Во-первых, вместо парного индекса $_{a_t v}$ для алгоритма a_t всюду используется сокращённое обозначение $_{tv}$.

Во-вторых, предполагается $L_{tv} = L - |X_{tv}| - |X'_{tv}|$, $\ell_{tv} = \ell - |X_{tv}|$, и вычисление величин L_{tv} и ℓ_{tv} в явном виде не записывается, чтобы не перегружать псевдокод.

В-третьих, при добавлении алгоритма a_d на шаге 3 создаётся тройка $\langle X_d, X'_d, c_d \rangle$ и вычисляется вероятность P_d получения алгоритма a_d методом μ_d. При этом в индексное множество заносится «пустой элемент», обозначаемый $\varnothing$. Это позволяет в дальнейшем сокращать запись индексов, полагая, что если индекс не двойной, а одинарный, то имеется в виду первый элемент индексного множества V_d, то есть $X_d \equiv X_{d\varnothing}$, $X'_d \equiv X'_{d\varnothing}$, $c_d \equiv c_{d\varnothing}$, $P_d \equiv P_{d\varnothing}$.

Вычисление вероятностей $P_{tw} = \mathsf{P}\big[X_{tw} \subseteq X\big]\big[X'_{tw} \subseteq \bar{X}\big]$ по формуле (4.7) также требует пояснений.

На шаге 3 вычисляется вероятность получения алгоритма a_d методом μ_d:

$$P_d = \frac{C^{\ell}_{L-|X'_d|}}{C^{\ell}_{L}} = \prod_{j=0}^{|X'_d|-1} \left(\frac{k-j}{L-j}\right).$$

Значения P_d одинаковы для всех алгоритмов с равным числом ошибок $n(a_d, \mathbb{X})$. Поэтому их можно вычислить один раз заранее для всех $n = 0, \ldots, L$.

На шаге 8 к порождающему множеству X_{tv} добавляется один объект, что приводит к очень простой корректировке предыдущего значения вероятности P'_{tv}:

$$P_{tv} = \frac{C^{\ell_{tv}-1}_{L_{tv}-1}}{C^{\ell}_{L}} = \frac{C^{\ell_{tv}}_{L_{tv}}}{C^{\ell}_{L}} \frac{\ell_{tv}}{L_{tv}} = P'_{tv} \frac{\ell_{tv}}{L_{tv}}.$$

На шаге 10 из тройки $\langle X_{tv}, X'_{tv}, c_{tv}\rangle$ формируется новая тройка $\langle X_{tw}, X'_{tw}, c_{tw}\rangle$ путём добавления Δ элементов к запрещающему множеству X'_{tw}. Следовательно,

$$P_{tw} = \frac{C^{\ell_{tv}}_{L_{tv}-\Delta}}{C^{\ell}_{L}} = \frac{C^{\ell_{tv}}_{L_{tv}}}{C^{\ell}_{L}} \frac{(L_{tv} - \ell_{tv} - \Delta + 1)\cdots(L_{tv} - \ell_{tv})}{(L_{tv} - \Delta + 1)\cdots L_{tv}} = P_{tv} \prod_{j=0}^{\Delta-1} \left(1 - \frac{\ell_{tv}}{L_{tv} - j}\right).$$

Алгоритм 6.1 может оказаться вычислительно неэффективным, если шаг 10 будет выполняться слишком часто. Каждое его выполнение приводит к добавлению ещё одного слагаемого в сумму (4.8). Следующая теорема позволяет сокращать время вычисления за счёт понижения точности верхней оценки Q_ε.

Теорема 6.3. *Если в Алгоритме 6.1 иногда пропускать шаг 10 при $c_{tv} = 1$, то вычисляемое в результате значение Q_ε будет верхней оценкой вероятности переобучения.*

Доказательство. Выполнение шага 10 при $c_{tv} = 1$, $c_{tw} = -1$ приводит к уменьшению текущего вычисленного значения Q_ε на величину $P_{tw} \geqslant 0$. Соответственно, невыполнение шага 10 приводит к исключению из суммы (4.9) отрицательного слагаемого $-P_{tw}$, и, возможно, ещё некоторого числа положительных и отрицательных слагаемых, которые появятся в этой сумме в результате последующих корректировок тройки $\langle X_{tw}, X'_{tw}, c_{tw}\rangle$ при выполнении шагов 10 и 12. Каждая такая корректировка возникает в результате добавления некоторого алгоритма a_d, $d > t$, который «отнимает» часть разбиений у алгоритма a_t, уменьшая слагаемое P_{tw} до величины $\tilde{P}_{tw}$:

$$\tilde{P}_{tw} = \mathsf{P}\big[X_{tw} \subseteq X\big]\big[X'_{tw} \subseteq \bar{X}\big]\big[X'_d \not\subseteq \bar{X}\big] \leqslant P_{tw}.$$

Исключение из суммы (4.9) отрицательного слагаемого $-P_{tw}$ вместе со всеми последующими слагаемыми, корректирующими тройку $\langle X_{tw}, X'_{tw}, c_{tw}\rangle$, может только увеличить значение Q_ε, вычисляемое Алгоритмом 6.1. ■

Рассмотрим *упрощённую рекуррентную оценку* вероятности переобучения, которая получится, если из Алгоритма 6.1 убрать шаги 9–12. В этом случае шаг 10 будет пропускаться всегда, и тройки $\langle X_{tw}, X'_{tw}, c_{tw}\rangle$ с отрицательными значениями c_{tw}

никогда не будут создаваться. В результате каждому алгоритму a_d будет соответствовать только одна тройка, то есть все индексные множества V_d, $d = 0, \ldots, D$, будут одноэлементными. Согласно теореме 6.3, значение Q_ε, вычисляемое упрощённым рекуррентным Алгоритмом 6.1, будет верхней оценкой вероятности переобучения. Чтобы выписать эту оценку в явном виде, нам потребуется ввести несколько новых обозначений.

Для каждого алгоритма $a \in A$ обозначим через E_a множество объектов генеральной выборки $\mathbb{X}$, на которых он допускает ошибку: $E_a = \{x_i \in \mathbb{X} \colon I(a, x_i) = 1\}$. Очевидно, $n(a, \mathbb{X}) = |E_a|$.

Подмножество алгоритмов $A_m = \{a \in A \colon n(a, \mathbb{X}) = m\}$ называется *m-м слоем* множества A. Разбиение множества A на слои $A = \bigsqcup_{m=0}^{L} A_m$ называется *расслоением* множества алгоритмов A.

Опр. 6.1. *Связностью $q(a)$ алгоритма $a \in A$ будем называть число алгоритмов в следующем слое, допускающих ошибки на тех же объектах, что и a:*

$$q(a) = \#\{a' \in A_{n(a,\mathbb{X})+1} \colon I(a, x) \leqslant I(a', x),\ x \in \mathbb{X}\}.$$

Образно говоря, связность $q(a)$ — это число способов, которыми вектор ошибок алгоритма a может быть «испорчен» ещё на одном каком-то объекте, если рассматривать всевозможные векторы ошибок алгоритмов множества A.

Графом связности или просто графом множества алгоритмов A будем называть направленный граф, вершины которого соответствуют алгоритмам, а рёбрами (a, a') соединяются пары алгоритмов, для которых $E_{a'} \setminus E_a = 1$. Тогда связность $q(a)$ алгоритма a — это число рёбер графа, исходящих из вершины a.

Пример 6.1. На рис. 2 слева показана двумерная линейно разделимая выборка длины $L = 10$, состоящая из объектов двух классов, по 5 объектов в каждом классе. Справа построен граф связности множества линейных алгоритмов классификации для данной выборки. По вертикальной оси отложены номера слоёв m. Единственная точка на графе при $m = 0$ соответствует алгоритму, разделяющему объекты на два класса без ошибок; следующий слой $m = 1$ содержит всевозможные алгоритмы, разделяющие выборку на два класса с одной ошибкой (для данной выборки их оказалось ровно 5); слой $m = 2$ содержит уже 8 алгоритмов, и т. д.

Теорема 6.4. *Пусть векторы ошибок всех алгоритмов множества A попарно различны, в A есть корректный на $\mathbb{X}$ алгоритм, Δ_{mq} — число алгоритмов в m-м слое со связностью q. Тогда справедлива верхняя оценка вероятности переобучения*

$$Q_\varepsilon \leqslant \sum_{m=\lceil \varepsilon k \rceil}^{L} \sum_{q=0}^{L} \Delta_{mq} \frac{C_{L-m-q}^{\ell-q}}{C_L^\ell}. \tag{6.3}$$

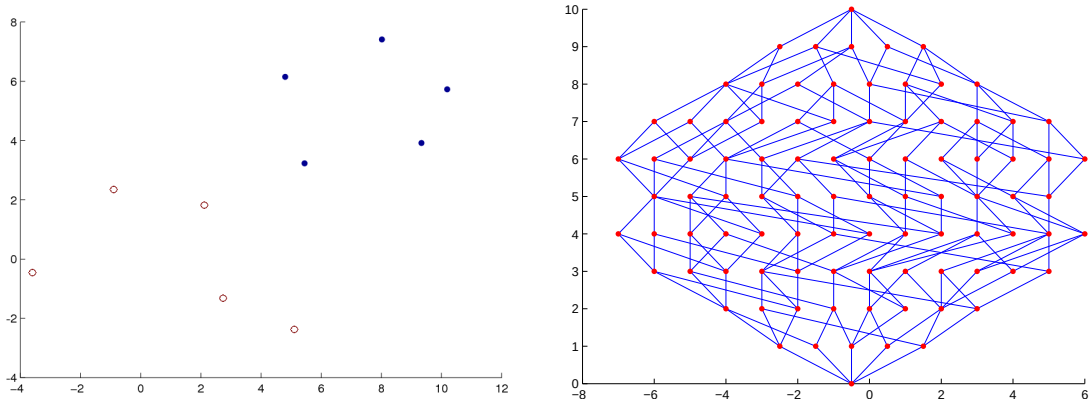

Рис. 2. Исходная выборка и граф связности множества линейных алгоритмов классификации.

Доказательство. Рассмотрим упрощённый рекуррентный Алгоритм 6.1, из которого удалены шаги 9–12. Он даёт верхнюю оценку вероятности переобучения. Для каждого алгоритма $a \in A$ он строит единственную тройку $\langle X_a, X'_a, 1\rangle$, в которой запрещающее множество X'_a совпадает с E_a, а порождающее множество состоит из всех объектов, добавленных на шаге 8. Это те и только те объекты x_i, для которых существует алгоритм $a' \in A$, допускающий на одну ошибку больше, чем a. Очевидно, при этом $X'_{a'} \setminus X'_a = E_{a'} \setminus E_a = \{x_i\}$ — одноэлементное множество. Число таких объектов x_i совпадает со значением связности $q(a)$. Таким образом, $|X'_a| = n(a, \mathbb{X})$, $|X_a| = q(a)$ для произвольного алгоритма $a \in A$. Следовательно, оценка (4.9) принимает вид:

$$Q_\varepsilon \leqslant \sum_{a\in A} \big[n(a,\mathbb{X}) \geqslant \varepsilon k\big] \frac{C^{\ell_a}_{L_a}}{C^\ell_L} = \sum_{a\in A} \big[n(a,\mathbb{X}) \geqslant \varepsilon k\big] \frac{C^{\ell-q(a)}_{L-n(a,\mathbb{X})-q(a)}}{C^\ell_L} =$$

$$= \sum_{m=\lceil \varepsilon k\rceil}^{L} \sum_{q=0}^{L} \underbrace{\sum_{a\in A} \big[n(a,\mathbb{X}) = m\big]\big[q(a) = q\big]}_{\Delta_{mq}} \frac{C^{\ell-q}_{L-m-q}}{C^\ell_L}.$$

■

Согласно оценке (6.3) наибольший вклад в вероятность переобучения вносят алгоритмы с малым числом ошибок, начиная от $m = \lceil \varepsilon k \rceil$. По мере увеличения m комбинаторный множитель $\frac{C^{\ell-q}_{L-m-q}}{C^\ell_L}$ убывает экспоненциально.

Второй вывод состоит в том, что увеличение связности q только улучшает оценку. В экспериментах с линейными алгоритмами классификации среднее значение связности q с высокой точностью совпадало с размерностью пространства параметров. При увеличении размерности пространства возникают два противонаправленных эффекта: с одной стороны, увеличивается число алгоритмов в каждом слое, что приводит к росту Q_ε; с другой стороны, увеличивается связность q, что приводит к уменьшению Q_ε.

Предварительные эксперименты показали также, что *профиль расслоения и связности* Δ_{mq} для некоторых семейств алгоритмов с высокой точностью является сепарабельным: $\Delta_{mq} \lessapprox \Delta_m \lambda_q$, где Δ_m — коэффициент разнообразия m-го слоя, λ_q — доля алгоритмов m-го слоя, имеющих связность q.

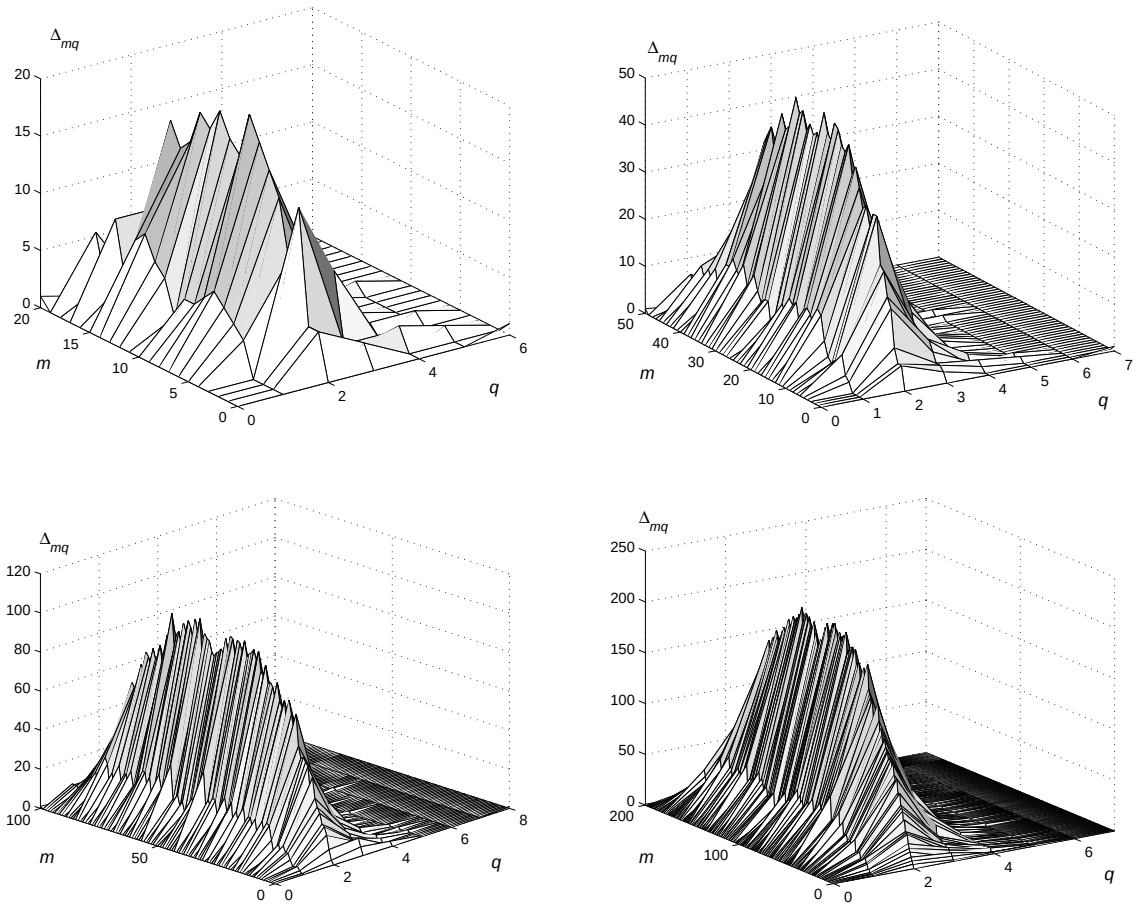

Рис. 3. Профили расслоения и связности для двумерных выборок длины $L = 20,\ 50,\ 100,\ 200$. Профиль Δ_{mq} — это количество алгоритмов с числом ошибок m на генеральной выборке и связностью q.

Вектор $(\Delta_m)_{m=0}^{L}$ логично называть *профилем расслоения*, а вектор $(\lambda_q)_{q=0}^{L}$ — *профилем связности* множества алгоритмов A. Очевидно, профиль связности удовлетворяет условию нормировки $\sum\limits_{q=0}^{L} \lambda_q = 1$.

На Рис. 3 показаны графики зависимости Δ_{mq} от m и q для множества линейных алгоритмов классификации и линейно разделимых двумерных выборок длины $L = 20,\ 50,\ 100$. Хорошо видно, что профиль связности концентрируется в точке $q = 2$, что совпадает с размерностью пространства. С увеличением длины выборки доминирование данной компоненты профиля усиливается[2].

[2]Вычислительные эксперименты, представленные на Рис. 2 и Рис. 3, выполнены Ильёй Решетняком, студентом ВМиК МГУ.

В терминах профилей расслоения и связности слегка ухудшенная оценка (6.3) принимает следующий вид:

$$Q_\varepsilon \leqslant \underbrace{\sum_{m=\lceil \varepsilon k \rceil}^{k} \Delta_m \frac{C_{L-m}^{\ell}}{C_L^{\ell}}}_{\text{VC-оценка}} \underbrace{\sum_{q=0}^{L} \lambda_q \left(\frac{\ell}{L-m} \right)^q}_{\text{поправка на связность}}.$$

Первая часть этой оценки представляет собой в точности VC-оценку (3.1), выраженную через профиль расслоения для частного случая, когда множество A содержит корректный алгоритм, $n(a_0, \mathbb{X}) = 0$.

Вторая часть представляет собой «поправку на связность». Она быстро убывает с ростом q, что делает оценку существенно более точной, чем классическая VC-оценка (3.1) и чем оценка, учитывающая только профиль расслоения.

При известной $L \times D$-матрице ошибок вычисление по формуле (6.3) занимает $O(D)$ операций, тогда как упрощённый Алгоритм 6.1 является более ресурсоёмким и требует $O(D^2)$ операций. Если *профиль расслоения и связности* Δ_{mq} каким-то образом удалось оценить заранее, то вычисления займут $O(L^2)$ операций, что в реальных ситуациях существенно меньше, чем $O(D)$. Если же профиль Δ_{mq} представлен в виде разложения $\Delta_m \lambda_q$, то вычисления займут $O(L)$ операций, что уже совершенно приемлемо для практических приложений.

7 Монотонная цепочка алгоритмов

Монотонная цепочка алгоритмов — это простейшая модель однопараметрического связного семейства алгоритмов, предполагающая, что при непрерывном удалении некоторого параметра от оптимального значения число ошибок на полной выборке только увеличивается.

Определим расстояние между алгоритмами как расстояние Хэмминга между их векторами ошибок:

$$\rho(a, a') = \sum_{i=1}^{L} \big| I(a, x_i) - I(a', x_i) \big|, \quad \forall a, a' \in A.$$

Опр. 7.1. *Множество алгоритмов $A = \{a_0, a_1, \ldots, a_D\}$ называется цепочкой алгоритмов, если $\rho(a_{d-1}, a_d) = 1$ для всех $d = 1, \ldots, D$.*

В экспериментах [19] было показано, что вероятность переобучения цепочки может оказаться существенно ниже, чем у произвольного несвязного множества алгоритмов с такими же частотами ошибок $\nu(a_d, \mathbb{X})$. Ниже выводятся точные оценки вероятности переобучения для некоторых специальных видов цепочек.

Опр. 7.2. *Цепочка алгоритмов $A = \{a_0, a_1, \ldots, a_D\}$ называется монотонной, если $n(a_d, \mathbb{X}) = m + d$ при некотором $m \geqslant 0$. Алгоритм a_0 называется лучшим в цепочке.*

Пример 7.1. Пусть $\mathbb{X}$ — множество точек в $\mathbb{R}^n$; A — семейство линейных алгоритмов классификации — параметрических отображений из $\mathbb{X}$ в $\{-1,+1\}$ вида

$$a(x,w) = \operatorname{sign}(x_1 w_1 + \cdots + x_n w_n), \quad x = (x_1,\ldots,x_n) \in \mathbb{R}^n$$

с параметром $w \in \mathbb{R}^n$. Пусть функция потерь имеет вид $I(a,x) = \big[a(x,w) \neq y(x)\big]$, где $y(x)$ — истинная классификация объекта x, и множество объектов $\mathbb{X}$ линейно разделимо, т. е. существует $w^* \in \mathbb{R}^n$, при котором алгоритм $a(x,w^*)$ не допускает ошибок на $\mathbb{X}$. Тогда, при некоторых дополнительных предположениях технического характера, множество алгоритмов $\big\{a(x,w^*+t\delta)\colon t \in [0,+\infty)\big\}$ образует монотонную цепочку для любого направляющего вектора $\delta \in \mathbb{R}^n$, за исключением некоторого конечного множества векторов. При этом $m = 0$.

Теорема 7.1. *Пусть $A = \{a_0, a_1, \ldots, a_D\}$ — монотонная цепочка; $L \geqslant m + D$. Тогда:*

1) в случае $D \geqslant k$

$$Q_\varepsilon = \sum_{d=0}^{k} P_d H_{L-d-1}^{\ell-1,m}(s_d(\varepsilon)); \qquad P_d = \frac{C_{L-d-1}^{\ell-1}}{C_L^\ell};$$

2) в случае $D < k$

$$Q_\varepsilon = \sum_{d=0}^{D-1} P_d H_{L-d-1}^{\ell-1,m}(s_d(\varepsilon)) \;+\; P_D H_{L-D}^{\ell,m}(s_D(\varepsilon));$$

$$P_d = \frac{C_{L-d-1}^{\ell-1}}{C_L^\ell},\ d = 0,\ldots,D-1; \qquad P_D = \frac{C_{L-D}^{\ell}}{C_L^\ell},$$

где P_d — вероятность получить алгоритм a_d методом μ; $s_d(\varepsilon) = \frac{\ell}{L}(m + d - \varepsilon k)$.

Доказательство. Перенумеруем объекты таким образом, чтобы каждый из алгоритмов a_d, $d = 1,\ldots,D$ допускал ошибку на объектах $x_1,\ldots,x_d$. Очевидно, лучший алгоритм a_0 не допускает ошибку ни на одном из этих объектов. Нумерация остальных объектов не имеет значения, так как алгоритмы не различимы на них.

Для наглядности представим выборку $\mathbb{X}$ разбитой на три блока:

$$\begin{array}{rccccccc}
 & x_1 & x_2 & x_3 & & x_D & & \overbrace{\qquad}^{m} \\
a_0 = (& 0, & 0, & 0, & \ldots & 0, & 0,\ldots,0, & 1,\ldots,1 \ ); \\
a_1 = (& 1, & 0, & 0, & \ldots & 0, & 0,\ldots,0, & 1,\ldots,1 \ ); \\
a_2 = (& 1, & 1, & 0, & \ldots & 0, & 0,\ldots,0, & 1,\ldots,1 \ ); \\
a_3 = (& 1, & 1, & 1, & \ldots & 0, & 0,\ldots,0, & 1,\ldots,1 \ ); \\
\ldots & & & & \ldots & & \ldots & \ldots \\
a_D = (& 1, & 1, & 1, & \ldots & 1, & 0,\ldots,0, & 1,\ldots,1 \ );
\end{array}$$

При рассмотрении алгоритма a_d возможны три случая.

1. Если $k < d$, то число ошибок алгоритма a_d на объектах $\{x_1,\ldots,x_d\}$ превышает длину контрольной выборки. Часть ошибок обязательно окажется в обучающей подвыборке X, и метод μ выберет другой алгоритм. В этом случае

$$[\mu X{=}a_d] = 0.$$

2. Если $d = D < k$, то метод μ выберет наихудший алгоритм в цепочке a_D тогда и только тогда, когда все объекты $\{x_1, \ldots, x_D\}$ будут находиться в контрольной подвыборке $\bar{X}$. В этом случае

$$[\mu X{=}a_d] = [x_1, \ldots, x_D \in \bar{X}].$$

3. Во всех остальных случаях метод μ выберет алгоритм a_d, если только все объекты $\{x_1, \ldots, x_d\}$ будут находиться в контрольной подвыборке $\bar{X}$, а объект x_{d+1} — в обучающей подвыборке X. В этом случае

$$[\mu X{=}a_d] = [x_{d+1} \in X][x_1, \ldots, x_d \in \bar{X}].$$

Теперь можно применить теорему 4.3.

Если $D \geqslant k$, то алгоритму a_d соответствуют следующие значения параметров (для упрощения обозначений вместо двойных индексов L_{a_d} будем использовать одинарные L_d): $L_d = L - d - 1$, $\ell_d = \ell - 1$, $m_d = m + d - d = m$, $s_d(\varepsilon) = \frac{\ell}{L}(m + d - \varepsilon k)$. Отсюда получаем утверждение теоремы для случая $D \geqslant k$.

Если $D < k$, то алгоритмам $a_0, \ldots, a_{D-1}$ соответствуют те же значения параметров, что и при $D \geqslant k$. Для наихудшего алгоритма a_D отличается только параметр $\ell_D = \ell$. Отсюда получаем утверждение теоремы для случая $D < k$. ■

Замечание 7.1. В ходе доказательства полезно проверить, что вероятности P_d вычислены корректно и в сумме дают единицу. Для случая $D \geqslant k$ проверка сводится к применению известного комбинаторного тождества:

$$\sum_{d=0}^{D} P_d = \sum_{d=0}^{k} P_d + \sum_{d=k+1}^{D} 0 = \frac{1}{C_L^\ell}\left(C_{L-1}^{\ell-1} + C_{L-2}^{\ell-1} + \cdots + C_{\ell-1}^{\ell-1}\right) = 1.$$

Для случая $D < k$ то же самое тождество приходится применить дважды, заметив, что $C_{L-D}^{\ell} = C_{L-D-1}^{\ell-1} + \cdots + C_{\ell-1}^{\ell-1}$:

$$\sum_{d=0}^{D} P_d = \frac{1}{C_L^\ell}\left(C_{L-1}^{\ell-1} + \cdots + C_{L-D}^{\ell-1} + C_{L-D}^{\ell}\right) = 1.$$

8 Унимодальная цепочка алгоритмов

Унимодальная цепочка является более реалистичной моделью однопараметрического связного семейства, по сравнению с монотонной цепочкой. Если мы имеем лучший алгоритм a_0 с оптимальным значением некоторого вещественного параметра, то отклонение значения этого параметра как в бóльшую, так и в меньшую, сторону будет приводить, как правило, к увеличению числа ошибок.

Опр. 8.1. *Множество алгоритмов $A = \{a_0, a_1, \ldots, a_D, a'_1, \ldots, a'_{D'}\}$ называется унимодальной, если левая ветвь $a_0, a_1, \ldots, a_D$ и правая ветвь $a_0, a'_1, \ldots, a'_{D'}$ являются монотонными цепочками. Алгоритм a_0 называется лучшим в цепочке.*

Пример 8.1 (продолжение примера 7.1). Пусть множество объектов $\mathbb{X} \subset \mathbb{R}^n$ линейно разделимо, т. е. существует линейный алгоритм классификации $a(x, w^*)$ с параметром $w^* \in \mathbb{R}^n$, не допускающий ошибок на $\mathbb{X}$. Тогда множество алгоритмов $\{a(x, w^* + t\delta) \colon t \in \mathbb{R}\}$ образует унимодальную цепочку для почти любого направляющего вектора $\delta \in \mathbb{R}^n$.

Обозначим через $m = n(a_0, \mathbb{X})$ число ошибок лучшего алгоритма.

Рассмотрим унимодальную цепочку с ветвями равной длины, $D = D'$. Перенумеруем объекты так, чтобы каждый из алгоритмов a_d, $d = 1, \ldots, D$ допускал ошибку на объектах $x_1, \ldots, x_d$; а каждый из алгоритмов a'_d, $d = 1, \ldots, D$ допускал ошибку на объектах $x'_1, \ldots, x'_d$. Будем предполагать, что множества объектов $\{x_1, \ldots, x_D\}$ и $\{x'_1, \ldots, x'_D\}$ не пересекаются. Очевидно, лучший алгоритм a_0 не допускает ошибку ни на одном из этих объектов. Нумерация остальных объектов не имеет значения, так как алгоритмы не различимы на них.

Для наглядности представим выборку $\mathbb{X}$ разбитой на четыре блока:

	x_1	x_2	x_3		x_D	x'_1	x'_2	x'_3		x'_D		$\overbrace{\quad}^{m}$	
$a_0 = ($	0,	0,	0,	$\ldots$	0,	0,	0,	0,	$\ldots$	0,	$0, \ldots, 0,$	$1, \ldots, 1$	$);$
$a_1 = ($	1,	0,	0,	$\ldots$	0,	0,	0,	0,	$\ldots$	0,	$0, \ldots, 0,$	$1, \ldots, 1$	$);$
$a_2 = ($	1,	1,	0,	$\ldots$	0,	0,	0,	0,	$\ldots$	0,	$0, \ldots, 0,$	$1, \ldots, 1$	$);$
$a_3 = ($	1,	1,	1,	$\ldots$	0,	0,	0,	0,	$\ldots$	0,	$0, \ldots, 0,$	$1, \ldots, 1$	$);$
$\ldots$				$\ldots$					$\ldots$		$\ldots$	$\ldots$	
$a_D = ($	1,	1,	1,	$\ldots$	1,	0,	0,	0,	$\ldots$	0,	$0, \ldots, 0,$	$1, \ldots, 1$	$);$
$a'_1 = ($	0,	0,	0,	$\ldots$	0,	1,	0,	0,	$\ldots$	0,	$0, \ldots, 0,$	$1, \ldots, 1$	$);$
$a'_2 = ($	0,	0,	0,	$\ldots$	0,	1,	1,	0,	$\ldots$	0,	$0, \ldots, 0,$	$1, \ldots, 1$	$);$
$a'_3 = ($	0,	0,	0,	$\ldots$	0,	1,	1,	1,	$\ldots$	0,	$0, \ldots, 0,$	$1, \ldots, 1$	$);$
$\ldots$				$\ldots$					$\ldots$		$\ldots$	$\ldots$	
$a'_D = ($	0,	0,	0,	$\ldots$	0,	1,	1,	1,	$\ldots$	1,	$0, \ldots, 0,$	$1, \ldots, 1$	$);$

Будем полагать, что если минимум (1.1) достигается на нескольких алгоритмах с одинаковым числом ошибок как на обучающей, так и на генеральной выборке, то метод μ выбирает алгоритм из левой ветви.

Теорема 8.1. *Пусть $A = \{a_0, a_1, \ldots, a_D, a'_1, \ldots, a'_D\}$ — унимодальная цепочка, $k \leqslant D$ и $2D + m \leqslant L$. Тогда вероятность получить каждый из алгоритмов цепочки в результате обучения есть*

$$P_0 = \mathsf{P}\big[\mu X{=}a_0\big] = \frac{C_{L-2}^{\ell-2}}{C_L^\ell};$$

$$P_d = \mathsf{P}\big[\mu X{=}a_d\big] = \frac{C_{L-d-1}^{\ell-1} - C_{L-2d-2}^{\ell-1}}{C_L^\ell};$$

$$P'_d = \mathsf{P}\big[\mu X{=}a'_d\big] = \frac{C_{L-d-1}^{\ell-1} - C_{L-2d-1}^{\ell-1}}{C_L^\ell};$$

вероятность переобучения при $s_d(\varepsilon) = \frac{\ell}{L}(m + d - \varepsilon k)$ *выражается в виде*

$$Q_\varepsilon = \frac{C_{L-2}^{\ell-2}}{C_L^\ell} H_{L-2}^{\ell-2,m}\left(s_0(\varepsilon)\right) + \sum_{d=1}^{k}\Bigg(2\frac{C_{L-d-1}^{\ell-1}}{C_L^\ell} H_{L-d-1}^{\ell-1,m}\left(s_d(\varepsilon)\right) - \\ - \frac{C_{L-2d-2}^{\ell-1}}{C_L^\ell} H_{L-2d-2}^{\ell-1,m}\left(s_d(\varepsilon)\right) - \frac{C_{L-2d-1}^{\ell-1}}{C_L^\ell} H_{L-2d-1}^{\ell-1,m}\left(s_d(\varepsilon)\right)\Bigg).$$

Доказательство. Введём вспомогательные переменные:

$$\begin{aligned}
&\beta_d = [x_{d+1} \in X][x_1, \ldots, x_d \in \bar{X}], \quad d = 1, \ldots, D-1;\\
&\beta_D = [x_1, \ldots, x_D \in \bar{X}];\\
&\beta'_d = [x'_{d+1} \in X][x'_1, \ldots, x'_d \in \bar{X}], \quad d = 1, \ldots, D-1;\\
&\beta'_D = [x'_1, \ldots, x'_D \in \bar{X}].
\end{aligned}$$

Условия $\beta_1, \ldots, \beta_D$ несовместны, причём одно из них выполнено тогда и только тогда, когда $x_1 \in \bar{X}$. Следовательно,

$$[x_1 \in X] + \beta_1 + \cdots + \beta_D = 1.$$

Аналогично,

$$[x'_1 \in X] + \beta'_1 + \cdots + \beta'_D = 1.$$

Если бы левая и правая ветви рассматривать как отдельные монотонные цепочки, то можно было бы утверждать, что $[\mu X{=}a_d] = \beta_d$ и $[\mu X{=}a'_d] = \beta'_d$. Однако в случае унимодальной цепочки условия получения алгоритмов a_d, a'_d приобретают более сложный вид. Если выполнено условие β_d и одновременно одно из условий $\beta'_{d+1}, \ldots, \beta'_D$, то метод μ выберет один из алгоритмов $a'_{d+1}, \ldots, a'_D$ из правой ветви, согласно договорённости, что выбирается наихудший алгоритм из всех, допускающих одинаковое наименьшее число ошибок на X. Аналогично, если выполнено условие β'_d и одновременно одно из условий $\beta_d, \ldots, \beta_D$, то метод μ выберет один из алгоритмов $a_d, \ldots, a_D$ из левой ветви. Обратим внимание, что алгоритмы левой ветви имеют приоритет. Таким образом, условия получения всех алгоритмов унимодальной цепочки выражаются через вспомогательные переменные:

$$\begin{aligned}
&[\mu X{=}a_0] = [x_1, x'_1 \in X] = \big(1 - \beta_1 - \cdots - \beta_D\big)\big(1 - \beta'_1 - \cdots - \beta'_D\big);\\
&[\mu X{=}a_d] = \beta_d\big(1 - \beta'_{d+1} - \cdots - \beta'_D\big), \quad d = 1, \ldots, D-1;\\
&[\mu X{=}a'_d] = \beta'_d\big(1 - \beta_d - \cdots - \beta_D\big), \quad d = 1, \ldots, D-1;\\
&[\mu X{=}a_D] = \beta_D;\\
&[\mu X{=}a'_D] = \beta'_D\big(1 - \beta_D\big).
\end{aligned}$$

Непосредственной подстановкой нетрудно убедиться, что тождество (4.2) выполнено, следовательно, совокупность условий $[\mu X{=}a]$ определена корректно.

Найдём вероятности всех алгоритмов цепочки, применив лемму 4.5.

$$P_0 = \mathsf{P}\big[\mu X{=}a_0\big] = \mathsf{P}[x_1, x'_1 \in X] = \frac{C_{L-2}^{\ell-2}}{C_L^\ell};$$

$$\begin{aligned} P_d = \mathsf{P}\big[\mu X{=}a_d\big] &= \mathsf{P}[x_{d+1} \in X][x_1, \ldots, x_d \in \bar{X}] - \\ &- \sum_{t=d+1}^{k-d} \mathsf{P}[x_{d+1}, x'_{t+1} \in X][x_1, \ldots, x_d, x'_1, \ldots, x'_t \in \bar{X}] = \\ &= \frac{1}{C_L^\ell}\left(C_{L-d-1}^{\ell-1} - \sum_{t=d+1}^{k-d} C_{L-d-t-2}^{\ell-2}\right) = \frac{C_{L-d-1}^{\ell-1} - C_{L-2d-2}^{\ell-1}}{C_L^\ell}; \end{aligned}$$

$$\begin{aligned} P'_d = \mathsf{P}\big[\mu X{=}a'_d\big] &= \mathsf{P}[x'_{d+1} \in X][x'_1, \ldots, x'_d \in \bar{X}] - \\ &- \sum_{t=d}^{k-d} \mathsf{P}[x'_{d+1}, x_{t+1} \in X][x'_1, \ldots, x'_d, x_1, \ldots, x_t \in \bar{X}] = \\ &= \frac{1}{C_L^\ell}\left(C_{L-d-1}^{\ell-1} - \sum_{t=d}^{k-d} C_{L-d-t-2}^{\ell-2}\right) = \frac{C_{L-d-1}^{\ell-1} - C_{L-2d-1}^{\ell-1}}{C_L^\ell}. \end{aligned}$$

Теперь запишем вероятность переобучения, применив теорему 4.6.

$$\begin{aligned} Q_\varepsilon = &\frac{C_{L-2}^{\ell-2}}{C_L^\ell} H_{L-2}^{\ell-2,m}\left(s_0(\varepsilon)\right) + \\ &+ \sum_{d=1}^{k}\left(\frac{C_{L-d-1}^{\ell-1}}{C_L^\ell} H_{L-d-1}^{\ell-1,m}\left(s_d(\varepsilon)\right) - \sum_{t=d+1}^{k-d} \frac{C_{L-d-t-2}^{\ell-2}}{C_L^\ell} H_{L-d-t-2}^{\ell-2,m}\left(s_d(\varepsilon)\right)\right) + \\ &+ \sum_{d=1}^{k}\left(\frac{C_{L-d-1}^{\ell-1}}{C_L^\ell} H_{L-d-1}^{\ell-1,m}\left(s_d(\varepsilon)\right) - \sum_{t=d}^{k-d} \frac{C_{L-d-t-2}^{\ell-2}}{C_L^\ell} H_{L-d-t-2}^{\ell-2,m}\left(s_d(\varepsilon)\right)\right). \end{aligned}$$

Полученное выражение можно упростить, заметив, что

$$\sum_{t=d+1}^{k-d} \frac{C_{L-d-t-2}^{\ell-2}}{C_L^\ell} H_{L-d-t-2}^{\ell-2,m}\left(s_d(\varepsilon)\right) = \frac{C_{L-2d-2}^{\ell-1}}{C_L^\ell} H_{L-2d-2}^{\ell-1,m}\left(s_d(\varepsilon)\right);$$

$$\sum_{t=d}^{k-d} \frac{C_{L-d-t-2}^{\ell-2}}{C_L^\ell} H_{L-d-t-2}^{\ell-2,m}\left(s_d(\varepsilon)\right) = \frac{C_{L-2d-1}^{\ell-1}}{C_L^\ell} H_{L-2d-1}^{\ell-1,m}\left(s_d(\varepsilon)\right);$$

Подставляя эти выражения в формулу для Q_ε, получим требуемую оценку. ■

Замечание 8.1. Нетрудно убедиться, что вероятности P_d, P'_d найдены корректно:

$$P_0 + \sum_{d=1}^{D}\big(P_d + P'_d\big) = \frac{C_{L-2}^{\ell-2}}{C_L^\ell} + \sum_{d=1}^{D}\frac{C_{L-d-1}^{\ell-1} - C_{L-2d-2}^{\ell-1}}{C_L^\ell} + \frac{C_{L-d-1}^{\ell-1} - C_{L-2d-1}^{\ell-1}}{C_L^\ell} =$$

$$= \frac{C_{L-2}^{\ell-2}}{C_L^\ell} + \frac{1}{C_L^\ell}\sum_{d=1}^{D}\Big(2C_{L-d-1}^{\ell-1} - C_{L-2d-2}^{\ell-1} - C_{L-2d-1}^{\ell-1}\Big) =$$

$$= \frac{1}{C_L^\ell}\Big(C_{L-2}^{\ell-2} + 2\big(C_{L-2}^{\ell-1} + \cdots + C_{\ell-1}^{\ell-1}\big) - \big(C_{L-3}^{\ell-1} + \cdots + C_{\ell-1}^{\ell-1}\big)\Big) =$$

$$= \frac{1}{C_L^\ell}\Big(C_{L-2}^{\ell-2} + 2C_{L-1}^{\ell} - C_{L-2}^{\ell}\Big) = 1.$$

9 Единичная окрестность лучшего алгоритма

Другим примером связного семейства является единичная окрестность лучшего алгоритма. Это искусственная постановка задачи, но она интересна по двум причинам. Во-первых, это «экстремальный» случай, когда алгоритмы максимально близки друг к другу, и классические оценки, основанные на неравенстве Буля, максимально завышены. Во-вторых, это первый шаг на пути к общим точным оценкам вероятности переобучения. Следующим шагом должно стать увеличение радиуса окрестности.

Опр. 9.1. *Множество алгоритмов* $A = \{a_0, a_1, \ldots, a_D\}$ *называется единичной окрестностью алгоритма* a_0, *если все векторы ошибок* a_d *попарно различны,* $n(a_d, \mathbb{X}) = n(a_0, \mathbb{X}) + 1$ *и* $\rho(a_0, a_d) = 1$ *для всех* $d = 1, \ldots, D$. *Алгоритм* a_0 *называется лучшим в окрестности или центром окрестности.*

Будем полагать, что если минимум (1.1) достигается на нескольких алгоритмах с одинаковым числом ошибок как на обучающей, так и на генеральной выборке, то метод μ выбирает алгоритм с меньшим номером.

Теорема 9.1. *Пусть* $A = \{a_0, a_1, \ldots, a_D\}$ *— единичная окрестность алгоритма* a_0; $m = n(a_0, \mathbb{X})$; $L \geqslant m + D$. *Тогда*

$$Q_\varepsilon = P_0 H_{L-D}^{\ell-D,m}\left(\tfrac{\ell}{L}(m - \varepsilon k)\right) + \sum_{d=1}^{D} P_d H_{L-d}^{\ell-d+1,m}\left(\tfrac{\ell}{L}(m + 1 - \varepsilon k)\right);$$

$$P_0 = \frac{C_{L-D}^{k}}{C_L^k}; \quad P_d = \frac{C_{L-d}^{k-1}}{C_L^k},\ d = 1, \ldots, D;$$

где P_d *— вероятность получить алгоритм* a_d *в результате обучения.*

Доказательство. Перенумеруем объекты таким образом, чтобы каждый из алгоритмов a_d, $d = 1, \ldots, D$ допускал ошибку на объекте x_d. Очевидно, лучший алгоритм a_0 не допускает ошибку ни на одном из этих объектов. Нумерация остальных объектов не имеет значения, так как алгоритмы не различимы на них.

Для наглядности представим выборку $\mathbb{X}$ разбитой на три блока:

$$\begin{array}{rccccccc}
 & x_1 & x_2 & x_3 & & x_D & & \overbrace{\qquad}^{m} \\
a_0 = (& 0, & 0, & 0, & \ldots & 0, & 0,\ldots,0, & 1,\ldots,1 \ ); \\
a_1 = (& 1, & 0, & 0, & \ldots & 0, & 0,\ldots,0, & 1,\ldots,1 \ ); \\
a_2 = (& 0, & 1, & 0, & \ldots & 0, & 0,\ldots,0, & 1,\ldots,1 \ ); \\
a_3 = (& 0, & 0, & 1, & \ldots & 0, & 0,\ldots,0, & 1,\ldots,1 \ ); \\
\ldots & & & & \ldots & & \ldots & \ldots \\
a_D = (& 0, & 0, & 0, & \ldots & 1, & 0,\ldots,0, & 1,\ldots,1 \ );
\end{array}$$

Нетрудно видеть, что множество разбиений, при которых метод μ выбирает алгоритм a_d, представляется в следующем виде:

$$\begin{aligned}
&\big[\mu X{=}a_0\big] = \big[x_1,\ldots,x_D \in X\big]; \\
&\big[\mu X{=}a_d\big] = \big[x_1,\ldots,x_{d-1} \in X\big]\big[x_d \in \bar{X}\big], \quad d = 1,\ldots,D.
\end{aligned}$$

Параметры для подстановки в формулу теоремы 4.3:

$$\begin{aligned}
&L_0 = L - D; \quad \ell_0 = \ell - D; \quad m_0 = m; \quad s_0(\varepsilon) = \tfrac{\ell}{L}(m - \varepsilon k); \\
&L_d = L - d; \quad \ell_d = \ell - d + 1; \quad m_d = m; \quad s_d(\varepsilon) = \tfrac{\ell}{L}(m + 1 - \varepsilon k); \quad d = 1,\ldots,D.
\end{aligned}$$

Подставляя эти параметры в формулу теоремы 4.3, получаем требуемое. ■

Замечание 9.1. Нетрудно убедиться, что вероятности P_d найдены корректно:

$$\sum_{d=0}^{D} P_d = \frac{1}{C_L^k}\big(C_{L-D}^k + \underbrace{C_{L-D}^{k-1} + \cdots + C_{L-1}^{k-1}}_{C_L^k - C_{L-D}^k}\big) = 1.$$

10 Пара алгоритмов

Рассмотрим ещё один «экстремальный» частный случай — когда семейство состоит только из двух алгоритмов, $\mathbb{A} = \{a_1, a_2\}$. Уже в этом простейшем случае возникает переобучение и проявляются эффекты расслоения и сходства, снижающие вероятность переобучения.

Точная оценка вероятности переобучения для двухэлементного семейства была представлена в [19] без доказательства. Ниже эта оценка выводится как формальное следствие теоремы 4.6.

Теорема 10.1. *Пусть в выборке $\mathbb{X}$ имеется m_0 объектов, на которых оба алгоритма допускают ошибку; m_1 объектов, на которых только a_1 допускает ошибку; m_2 объектов, на которых только a_2 допускает ошибку; и для определённости $m_1 \geqslant m_2$:*

$$\begin{aligned}
a_1 &= (\,1,\ldots,1,\ 1,\ldots,1,\ 0,\ldots,0,\ 0,\ldots,0\,); \\
a_2 &= (\underbrace{1,\ldots,1}_{m_0},\ \underbrace{0,\ldots,0}_{m_1},\ \underbrace{1,\ldots,1}_{m_2},\ 0,\ldots,0\,).
\end{aligned}$$

Тогда для любого $\varepsilon \in [0,1)$ справедлива точная оценка:

$$Q_\varepsilon = \sum_{s_0=0}^{m_0}\sum_{s_1=0}^{m_1}\sum_{s_2=0}^{m_2} \frac{C_{m_0}^{s_0} C_{m_1}^{s_1} C_{m_2}^{s_2} C_{L-m_0-m_1-m_2}^{\ell-s_0-s_1-s_2}}{C_L^\ell} \times$$
$$\times\Big(\big[s_1 \leqslant s_2\big]\big[s_0+s_1 \leqslant \tfrac{\ell}{L}(m_0+m_1-\varepsilon k)\big] +$$
$$+ \big[s_1 > s_2\big]\big[s_0+s_2 \leqslant \tfrac{\ell}{L}(m_0+m_2-\varepsilon k)\big]\Big). \tag{10.1}$$

Доказательство. Введём множества объектов U_1 и U_2, на которых только один из алгоритмов допускает ошибку:

$$U_1 = \big\{x \in \mathbb{X}\colon I(a_1,x)=1,\ I(a_2,x)=0\big\}, \quad |U_1| = m_1;$$
$$U_2 = \big\{x \in \mathbb{X}\colon I(a_1,x)=0,\ I(a_2,x)=1\big\}, \quad |U_2| = m_2.$$

Если в обучающую подвыборку X попадает больше объектов из U_1, чем из U_2, то метод μ выбирает алгоритм a_1. Если в X попадает одинаковое число объектов из U_1 и U_2, то также выбирается алгоритм a_1. Иначе выбирается a_2. Обозначим через X_1 произвольное подмножество U_1, через X_2 — произвольное подмножество U_2.

$$[\mu X{=}a_1] = \sum_{(X_1,X_2)\in V_1} \big[X_1 \cup X_2 \subseteq X\big]\big[(U_1\backslash X_1)\cup(U_2\backslash X_2) \subseteq \bar{X}\big];$$
$$[\mu X{=}a_2] = \sum_{(X_1,X_2)\in V_2} \big[X_1 \cup X_2 \subseteq X\big]\big[(U_1\backslash X_1)\cup(U_2\backslash X_2) \subseteq \bar{X}\big];$$

где V_1 и V_2 — индексные множества, соответственно, для алгоритмов a_1 и a_2:

$$V_1 = \big\{v=(X_1,X_2)\colon X_1 \subseteq U_1,\ X_2 \subseteq U_2,\ |X_1| \leqslant |X_2|\big\};$$
$$V_2 = \big\{v=(X_1,X_2)\colon X_1 \subseteq U_1,\ X_2 \subseteq U_2,\ |X_1| > |X_2|\big\}.$$

Отсюда находятся все параметры для подстановки в формулу теоремы 4.6:

$$X_{1v} = X_{2v} = X_1 \cup X_2;$$
$$X'_{1v} = X'_{2v} = (U_1\backslash X_1)\cup(U_2\backslash X_2);$$
$$L_{1v} = L_{2v} = L - m_1 - m_2;$$
$$\ell_{1v} = \ell_{2v} = \ell - |X_1| - |X_2|;$$
$$m_{1v} = m_0 + m_1 - |X_1| - |U_1\backslash X_1| = m_0;$$
$$m_{2v} = m_0 + m_2 - |X_2| - |U_2\backslash X_2| = m_0;$$
$$s_{1v}(\varepsilon) = \tfrac{\ell}{L}(m_0+m_1-\varepsilon k) - |X_1|;$$
$$s_{2v}(\varepsilon) = \tfrac{\ell}{L}(m_0+m_2-\varepsilon k) - |X_2|;$$

Запишем вероятность получить алгоритм a_1 согласно теореме 4.6 и, вводя обозначения $s_1 = |X_1|$ и $s_2 = |X_2|$, заменим суммирование по подмножествам X_1, X_2 суммированием по значениям мощности этих подмножеств:

$$P_1 = \sum_{(X_1,X_2)\in V_1} \frac{C_{L-m_1-m_2}^{\ell-|X_1|-|X_2|}}{C_L^\ell} = \sum_{s_1=0}^{m_1}\sum_{s_2=0}^{m_2} \big[s_1 \leqslant s_2\big] C_{m_1}^{s_1} C_{m_2}^{s_2} \frac{C_{L-m_1-m_2}^{\ell-s_1-s_2}}{C_L^\ell}.$$

Вероятность P_2 получается из P_1 заменой $[s_1 \leqslant s_2]$ на $[s_1 > s_2]$.

Запишем вероятность переобучения согласно теореме 4.6, снова заменяя суммирование по подмножествам X_1, X_2 суммированием по s_1, s_2:

$$Q_\varepsilon = \sum_{s_1=0}^{m_1}\sum_{s_2=0}^{m_2}\big[s_1 \leqslant s_2\big]C_{m_1}^{s_1}C_{m_2}^{s_2}\frac{C_{L-m_1-m_2}^{\ell-s_1-s_2}}{C_L^\ell}H_{L-m_1-m_2}^{\ell-s_1-s_2,m_0}\left(\tfrac{\ell}{L}(m_0+m_1-\varepsilon k)-s_1\right)+$$
$$+\sum_{s_1=0}^{m_1}\sum_{s_2=0}^{m_2}\big[s_1 > s_2\big]C_{m_1}^{s_1}C_{m_2}^{s_2}\frac{C_{L-m_1-m_2}^{\ell-s_1-s_2}}{C_L^\ell}H_{L-m_1-m_2}^{\ell-s_1-s_2,m_0}\left(\tfrac{\ell}{L}(m_0+m_2-\varepsilon k)-s_2\right).$$

Подставляя сюда гипергеометрическую функцию распределения как сумму

$$H_{L-m_1-m_2}^{\ell-s_1-s_2,m_0}(z) = \sum_{s_0=0}^{m_0}[s_0 \leqslant z]\frac{C_{m_0}^{s_0}C_{L-m_0-m_1-m_2}^{\ell-s_0-s_1-s_2}}{C_{L-m_1-m_2}^{\ell-s_1-s_2}}, \tag{10.2}$$

и упрощая полученное выражение, получаем утверждение теоремы. ■

Численные эксперименты с данной оценкой для пары алгоритмов уже были выполнены в [19]. Основной вывод заключался в том, что даже в этом простейшем случае возникает переобучение и проявляются эффекты расслоения и сходства, снижающие вероятность переобучения.

Литература

[1] *Вапник В. Н.* Восстановление зависимостей по эмпирическим данным. — М.: Наука, 1979.

[2] *Вапник В. Н., Червоненкис А. Я.* О равномерной сходимости частот появления событий к их вероятностям // *ДАН СССР.* — 1968. — Т. 181, № 4. — С. 781–784.

[3] *Вапник В. Н., Червоненкис А. Я.* О равномерной сходимости частот появления событий к их вероятностям // *Теория вероятностей и ее применения.* — 1971. — Т. 16, № 2. — С. 264–280.

[4] *Вапник В. Н., Червоненкис А. Я.* Теория распознавания образов. — М.: Наука, 1974.

[5] *Bax E. T.* Similar classifiers and VC error bounds: Tech. Rep. CalTech-CS-TR97-14: 1997. `http://citeseer.ist.psu.edu/bax97similar.html`.

[6] *Boucheron S., Bousquet O., Lugosi G.* Theory of classification: A survey of some recent advances // *ESAIM: Probability and Statistics.* — 2005. — no. 9. — Pp. 323–375. `http://www.econ.upf.edu/~lugosi/esaimsurvey.pdf`.

[7] *Bousquet O.* Concentration Inequalities and Empirical Processes Theory Applied to the Analysis of Learning Algorithms: Ph.D. thesis / Ecole Polytechnique, France. — 2002. `http://www.kyb.mpg.de/publications/pss/ps1444.ps`.

[8] *Efron B.* The Jackknife, the Bootstrap, and Other Resampling Plans. — SIAM, Philadelphia, 1982.

[9] *Herbrich R., Williamson R.* Algorithmic luckiness // *Journal of Machine Learning Research.* — 2002. — no. 3. — Pp. 175–212. `http://citeseer.ist.psu.edu/article/herbrich02algorithmic.html`.

[10] *Kohavi R.* A study of cross-validation and bootstrap for accuracy estimation and model selection // 14th International Joint Conference on Artificial Intelligence, Palais de Congres Montreal, Quebec, Canada. — 1995. — Pp. 1137–1145.
http://citeseer.ist.psu.edu/kohavi95study.html.

[11] *Langford J.* Quantitatively Tight Sample Complexity Bounds: Ph.D. thesis / Carnegie Mellon Thesis. — 2002.
http://citeseer.ist.psu.edu/langford02quantitatively.html.

[12] *Langford J., McAllester D.* Computable shell decomposition bounds // Proc. 13th Annu. Conference on Comput. Learning Theory. — Morgan Kaufmann, San Francisco, 2000. — Pp. 25–34.
http://citeseer.ist.psu.edu/langford00computable.html.

[13] *Lugosi G.* On concentration-of-measure inequalities. — Machine Learning Summer School, Australian National University, Canberra. — 2003.
http://citeseer.ist.psu.edu/lugosi98concentrationmeasure.html.

[14] *Philips P.* Data-Dependent Analysis of Learning Algorithms: Ph.D. thesis / The Australian National University, Canberra. — 2005.
http://infoeng.rsise.anu.edu.au/files/petra_philips_thesis.pdf.

[15] *Sill J.* Monotonicity and connectedness in learning systems: Ph.D. thesis / California Institute of Technology. — 1998.
http://etd.caltech.edu/etd/available/etd-09222005-110351/.

[16] *Vapnik V.* Statistical Learning Theory. — Wiley, New York, 1998.

[17] *Vayatis N., Azencott R.* Distribution-dependent Vapnik-Chervonenkis bounds // *Lecture Notes in Computer Science*. — 1999. — Vol. 1572. — Pp. 230–240.
http://citeseer.ist.psu.edu/vayatis99distributiondependent.html.

[18] *Vorontsov K. V.* Combinatorial probability and the tightness of generalization bounds // *Pattern Recognition and Image Analysis*. — 2008. — Vol. 18, no. 2. — Pp. 243–259.
http://www.springerlink.com/content/78537p01838123u7/.

[19] *Vorontsov K. V.* On the influence of similarity of classifiers on the probability of overfitting // Pattern Recognition and Image Analysis: new information technologies (PRIA-9). — Vol. 2. — Nizhni Novgorod, Russian Federation, 2008. — Pp. 303–306.
http://www.ccas.ru/frc/papers/voron08pria-conf-eng.pdf.