МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ
МОСКОВСКИЙ ФИЗИКО-ТЕХНИЧЕСКИЙ ИНСТИТУТ (государственный университет)
ФАКУЛЬТЕТ УПРАВЛЕНИЯ И ПРИКЛАДНОЙ МАТЕМАТИКИ
ВЫЧИСЛИТЕЛЬНЫЙ ЦЕНТР ИМ. А. А. ДОРОДНИЦЫНА РАН
КАФЕДРА «ИНТЕЛЛЕКТУАЛЬНЫЕ СИСТЕМЫ»

Минаев Павел Юрьевич

Методика тестирования алгоритмов классификации в системе Полигон и её обоснование

511656 — Математические и информационные технологии

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА МАГИСТРА

Научный руководитель:
с.н.с. ВЦ РАН, д.ф.-м.н. Воронцов К.В.

Москва 2011 г.

Оглавление

Аннотация

При решении задач классификации возникает проблема выбора наилучшего алгоритма для данной конкретной задачи. Существует большое количество методов тестирования алгоритмов классификации, которые были предложены различными авторами. В данной работе были исследованы существующие методы, после чего они были стандартизованы и обобщены в единую методику тестирования алгоритмов классификации.

Данная методика тестирования была реализована в научном проекте «Полигон», который был запущен в 2009 году. В рамках проекта было проведено множество практических тестирований алгоритмов классификации на реальных задачах. Эти эксперименты позволяют сделать вывод о важности использования единой методики при оценивании качества алгоритмов классификации.

Важным вопросом при реализации данного проекта стало обоснование использования методики тестирования. Были проведены серии экспериментов, с помощью которых получены практические рекомендации по использованию данной методики тестирования в системе «Полигон».

Введение

Существует большое количество методов сравнения алгоритмов классификации на задачах, которые предлагались различными авторами в разные годы. До текущего момента не было попыток объединить данные методы в единую методику тестирования алгоритмов классификации.

Целью данного исследования является разработка и реализация методики тестирования алгоритмов классификации для системы «Полигон», которая позволит проводить тестирование различных алгоритмов на различных задачах, используя web-интерфейс.

К методике были предъявлены следующие требования:

- Стандартизация методов,
- Воспроизводимость результатов тестирования.

Стандартизация методов достигается за счёт того, что все методы рассчитываются одинаковым образом. Это позволяет сравнивать результаты работы различных алгоритмов на различных задачах.

Воспроизводимость результатов достигается за счёт идентичности реализации алгоритма, идентичности методики тестирования, а также идентичности исходных данных. В системе «Полигон» реализованы репозитирий задач и репозиторий алгоритмов, кроме этого в базе данных хранятся результаты работы всех алгоритмов классификации на всех задачах. Всё вместе это позволяет воспроизводить результаты тестирования, и, следовательно, иметь возможность использовать данную систему как доказательство хорошего качества классификации и для ссылок в научных статьях

В предлагаемой методике тестирования сведены вместе основные методы анализа, предложенные ранее авторами. Данная обобщённая методика позволяет лучше оценивать качество классификации, а также лучше изучить задачу, на которой проводится тестирование.

Методика тестирования алгоритмов классификации была реализована в рамках научного проекта «Полигон», который был запущен как web-интерфейс.

В ходе практического использования методики с помощью системы «Полигон» возникли вопросы, связанные с её обоснованием. Вторая часть данной работы посвящена экспериментальному исследованию вопроса обоснования выбранной методики тестирования. Было проведено большое количество экспериментов, которые позволили дать практические рекомендации для использования в системе «Полигон».

Методика тестирования в системе Полигон

Класс решаемых задач

Пусть X – множество объектов, Y – конечное множество классов, $X^L = \{(x_i, y_i)\}_{i=1}^L \subset X \times Y$ – выборка длины L. Объекты x из X описываются признаками $f_1(x), \ldots, f_n(x)$, возможно разнотипными. Задача классификации задаётся $L \times n$ - матрицей данных $F = [f_{ij}] = f_j(x_i)$ и целевым вектором $[y_i]$. Дополнительно может быть задана матрица потерь $[C_{yy'}]$, где $[C_{yy'}]$ – штраф за отнесение объекта класса y к классу y', а также некоторая априорная информация о признаках. Матрица данных может содержать пропуски.

Стандартная постановка задачи классификации:

Требуется построить алгоритм классификации $a: X \to Y$, аппроксимирующий неизвестную целевую зависимость $y(x)$ на всём множестве X.

Метод обучения μ по обучающей выборке $X^L \subseteq X$ строит алгоритм классификации $a = \mu(X^L)$.

Рассмотрим алгоритм классификации с точки зрения Полигона. На входе он получает обучающую выборку в виде матрицы объекты-признаки F и целевого вектора y и контрольную выборку F' в виде матрицы объекты-признаки. На выходе он выдаёт вектор классификации и матрицу апостериорной вероятности классов для контрольных объектов.

Качество алгоритма a на конечной выборке U характеризуется частотой ошибок:

$$\nu(a, U) = \frac{1}{U} \sum_{x \in U} [a(x) \neq y(x)].$$

Скользящий контроль

Скользящий контроль (cross-validation, CV) — процедура эмпирического оценивания обобщающей способности алгоритмов классификации.

Фиксируется некоторое множество разбиений исходной генеральной выборки на две подвыборки: обучающую и контрольную. Для каждого разбиения выполняется настройка алгоритма по обучающей выборке, затем оценивается его средняя ошибка на объектах контрольной выборки. Оценкой скользящего контроля называется средняя по всем разбиениям величина ошибки на контрольных выборках.

Если выборка независима, то средняя ошибка скользящего контроля даёт несмещённую оценку вероятности ошибки.

Процедура скользящего контроля

Выборка X^L разбивается N различными способами на две непересекающиеся выборки: $X^L = X_n^l \cup X_n^k$, где X_n^l — обучающая выборка длины l, X_n^k — контрольная выборка длины $k = L - l$, $n = 1, \dots, N$ — номер разбиения.

Для каждого разбиения n строится алгоритм $a_n = \mu(X_n^l)$ и вычисляется значение функционала качества $Q_n = Q(a_n,\ X_n^k)$. Среднее арифметическое значений Q_n по всем разбиениям называется оценкой скользящего контроля:

$$CV(\mu, X^L) = \frac{1}{N}\sum_{n=1}^{N} Q(\mu(X_n^m), X_n^k).$$

Различные варианты скользящего контроля отличаются видами функционала качества и способами разбиения выборки.

Стратификация выборки

Стратификация выборки — это способ уменьшить разброс (дисперсию) оценок скользящего контроля, в результате чего получаются более узкие доверительные интервалы.

Стратификация заключается в том, чтобы заранее поделить выборку на части (страты), и при разбиении на обучение длины l и контроль длины k гарантировать, что каждая страта будет поделена между обучением и контролем в той же пропорции $l:k$.

Стратификация классов в задачах классификации означает, что каждый класс делится между обучением и контролем в пропорции $l:k$.

Разновидности скользящего контроля

Полный скользящий контроль (complete CV)

Оценка скользящего контроля строится по всем $N = C_L^k$ разбиениям.

При $k > 2$ количество разбиений становится слишком, что затрудняет практическое применение данного метода.

Случайные разбиения

Разбиения $n = 1, \dots, N$ выбираются случайно, независимо и равновероятно из множества всех C_L^k разбиений.

Контроль по отдельным объектам (leave-one-out CV)

Является частным случаем полного скользящего контроля при $l = 1$, соотвественно, $N = L$. LOO - самый распространённый вариант скользящего контроля.

Преимущества LOO в том, что каждый объект ровно один раз участвует в контроле, а длина обучающих выборок лишь на единицу меньше длины полной выборки.

Недостатком LOO является большая ресурсоёмкость, так как обучаться приходится L раз. Некоторые методы обучения позволяют достаточно быстро перенастраивать внутренние параметры алгоритма при замене одного обучающего объекта другим. В этих случаях вычисление LOO удаётся заметно ускорить.

Скользящий контроль по q блокам (q-fold CV)

Выборка случайным образом разбивается на q непересекающихся блоков одинаковой длины $k_1, \dots, k_q$:

$$X^L = X_1^{k_1} \cup \dots \cup X_q^{k_q},$$

$k_1 + \dots + k_q$. Каждый блок по очереди становится контрольной выборкой, при этом обучение производится по остальным $q - 1$ блокам. Критерий определяется как средняя ошибка на контрольной выборке:

$$CV(\mu, X^L) = \frac{1}{q} \sum_{n=1}^{q} Q\left(\mu\left(X^L \backslash X_n^{k_n}\right), X_n^{k_n}\right)$$

Это компромисс между LOO и случайными разбиениями. С одной стороны, обучение производится только q раз вместо L. С другой стороны, длина обучающих выборок, равная $\frac{q-1}{q} L$ с точностью до округления, не сильно отличается от длины полной выборки L. Обычно выборку разбивают случайным образом на 5 или 20 блоков.

Контроль по t×q блокам (t×q-fold CV)

Контроль по q блокам (q-fold CV) повторяется t раз. Каждый раз выборка случайным образом разбивается на q непересекающихся блоков. Этот способ наследует все преимущества q-fold CV, при этом появляется дополнительная возможность увеличивать число разбиений.

Данный вариант скользящего контроля, со стратификацией классов, является стандартной методикой тестирования и сравнения алгоритмов классификации. В частности, он применяется в системах WEKA и «Полигон алгоритмов».

Процедура t*q-кратного скользящего контроля в системе Полигон

Процедура скользящего контроля является основной в системе Полигон. Производится $\boldsymbol{N}$ разбиений выборки $\boldsymbol{X^L}$ на обучающую подвыборку длины $\boldsymbol{l}$ и контрольную длины $\boldsymbol{k}$, $\boldsymbol{X^L = X_n^l \cup X_n^k}$, $\boldsymbol{L = l + k}$, $\boldsymbol{n = 1, \dots, N}$. Оценка скользящего контроля для функции $\boldsymbol{\varphi}: \{\boldsymbol{1, \dots, N}\} \to \boldsymbol{R}$ определяется как среднее

$$\widehat{\boldsymbol{E}}\boldsymbol{\varphi} = \frac{\mathbf{1}}{\boldsymbol{N}}\sum_{\boldsymbol{n}=\mathbf{1}}^{\boldsymbol{N}}\boldsymbol{\varphi}(\boldsymbol{n}).$$

Разбиения строятся по стандартной методике $\boldsymbol{t} \times \boldsymbol{q}$ – кратного скользящего контроля (в системе Полигон $\boldsymbol{t} = \mathbf{10}, \boldsymbol{q} = \mathbf{5}$). Генерируется $\boldsymbol{t}$ случайных разбиений выборки $\boldsymbol{X^L}$ на $\boldsymbol{q}$ блоков примерно равной длины и равными долями классов, и каждый блок поочерёдно становится контрольной выборкой, а обучение происходит по остальным блокам. Таким образом, количество разбиений равно произведению количества случайных разбиений $\boldsymbol{t}$ на количество блоков в каждом разбиении $\boldsymbol{q}$: $\boldsymbol{N} = \boldsymbol{tq}$, а длина контрольной выборки будет равна длине одного блока: $\boldsymbol{k} = \frac{\boldsymbol{L}}{\boldsymbol{q}}$ с точностью до округления. Именно с помощью процедуры скользящего контроля в системе “Полигон” будут вычисляться различные методики.

Качество классификации на n-ом разбиении характеризуется частотой ошибок на обучении $\nu_n^l = \nu\left(a_n, X_n^l\right)$ и на контроле $\nu_n^k = \nu\left(a_n, X_n^k\right)$. Обобщающая способность метода μ на выборке X^L характеризуется одной из оценок скользящего контроля

$$CV\left(\mu, X^L\right) = \widehat{E}\nu_n^k; \qquad CV\left(\mu, X^L\right) = \widehat{E}[\nu_n^k - \nu_n^l > \varepsilon].$$

Статистики в системе Полигон

Разложение ошибки на смещение и вариацию

Разложение ошибки на смещение и вариацию является мощным инструментом анализа работы алгоритма классификации. При заданной функции потерь и выборке средние ошибки алгоритма классификации можно разложить на три компоненты:

- *Смещение* – показывает степень несогласованности данных задачи с моделью обучения.
- *Вариация* – показывает степень изменчивости результата алгоритма классификации при варьировании состава обучающей выборки.
- *Шум* – минимальная неустранимая средняя ошибка для любого алгоритма.

Разложение ошибки и оценка компонент позволяет дать оценку качеству алгоритма классификации. Большое значение смещения говорит о том, что выбранная модель обучения не подходит для решения задачи. Большое значение вариации означает сильный разброс в результатах классификации в зависимости от состава обучающей выборки и, следовательно, неустойчивость алгоритма классификации.

Теория

Пусть нам дана выборка $\{(\boldsymbol{x_1},\ \boldsymbol{y_1}), \dots, (\boldsymbol{x_n},\ \boldsymbol{y_n})\}$, $\boldsymbol{f}(\boldsymbol{x})$ - обученный на этой выборке алгоритм классификации. На объекте $\boldsymbol{x_i}$ этот алгоритм предсказывает ответ $\boldsymbol{y} = \boldsymbol{f}(\boldsymbol{x_i})$, а $\boldsymbol{y_i}$ является истинным значение класса, к которому принадлежит объект $\boldsymbol{x_i}$. Функция потерь $\boldsymbol{L}(\boldsymbol{y_i}, \boldsymbol{y})$ измеряет стоимость предсказания $\boldsymbol{y}$, когда истинным значением является $\boldsymbol{y_i}$.

Введём ***оптимальное предсказание*** $\boldsymbol{y_*}$ для объекта $\boldsymbol{x_i}$ как тот ответ, который минимизирует $\boldsymbol{L}(\boldsymbol{y_i}, \boldsymbol{y_*})$. Оптимальным является тот алгоритм, который выдаёт ответ $\boldsymbol{f}(\boldsymbol{x}) = \boldsymbol{y_*}$ для каждого $\boldsymbol{x}$.

Рассмотрим объект $\boldsymbol{x_i}$, принадлежащий классу $\boldsymbol{y_i}$. Так как метод обучения создаёт различные алгоритмы классификации при различных обучающих выборках, то функция потерь $\boldsymbol{L}(\boldsymbol{y_i}, \boldsymbol{y})$ будет также являться функцией от обучающей выборки. Чтобы устранить эту зависимость следует делать усреднение по всем данным выборкам. Пусть $\boldsymbol{D}$ – это набор выборок, тогда ожидаемыми потерями, усреднёнными по набору выборок $\boldsymbol{D}$, будут $\boldsymbol{E_D}[\boldsymbol{L}(\boldsymbol{y_i}, \boldsymbol{y})]$. Таким образом, средние ошибки раскладываются на три величины: смещение, вариацию и шум. При этом алгоритм стремится к минимальным потерям, то есть минимизирует ожидаемые потери $\boldsymbol{E_D}[\boldsymbol{L}(\boldsymbol{y_i}, \boldsymbol{y})]$.

Теперь определим ***главное предсказание*** для функции потерь $\boldsymbol{L(y_i, y)}$ относительно объекта $\boldsymbol{x_i}$ и набора выборок $\boldsymbol{D}$ как

$$\boldsymbol{y_{min}(x_i) = argmin_{y'}\, E_D[L(y', y(x_i))]}$$

Пусть $\boldsymbol{Y}$ является набором всех предсказаний, которые выдаёт алгоритм на выборках из $\boldsymbol{D}$, тогда главное предсказание – это значение $\boldsymbol{y'}$, имеющее минимальные средние потери относительно всех предсказаний из $\boldsymbol{Y}$.

Определим ***смещение*** алгоритма классификации на объекте $\boldsymbol{x_i}$, как

$$\boldsymbol{B(x_i) = L(y_*, y_{min})}$$

Другими словами, это потери, которые являются следствием несовпадения главного предсказания, обеспечивающего минимальные средние потери в данном алгоритме классификации, и оптимального предсказаний.

Определим ***вариацию*** алгоритма классификации на объекте $\boldsymbol{x_i}$, как

$$\boldsymbol{V(x_i) = E_D[L(y_{min}, y)]},$$

где усреднение происходит по переменной $\boldsymbol{y}$, которая пробегает по всем значениям выборки $\boldsymbol{D}$. Другими словами, вариация - это средние потери, связанные с разницей между всеми возможными предсказаниями в модели и главным предсказанием.

Определим *шум* на объекте $\boldsymbol{x_i}$, принадлежащему классу $\boldsymbol{y_i}$, как

$$\boldsymbol{N(x_i) = L(y_i, y_*)}.$$

Другими словами, это неизбежная компонента потерь, встречающаяся вне зависимости от алгоритма обучения.

Разложение ошибки на вариацию и смещение без учёта шума показано на следующем рисунке.

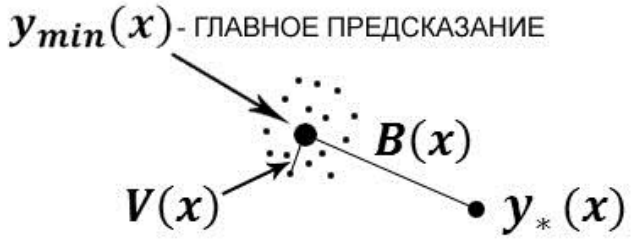


Рисунок 1. Разложение ошибки на смещение и вариацию

Рассмотрим объект x_i, принадлежащий классу y_i, и алгоритм классификации, который предсказал y на обучающей выборке D. Для функции потерь L, получается следующее разложение:

$$E_D[L(y_i, y)] = c_1 L(y_i, y_*) + L(y_*, y_{min}) + c_2 E_D[L(y_{min}, y)] =$$

$$= c_1 N(x_i) + B(x_i) + c_2 V(x_i)$$

c_1 и c_2 являются коэффициентами, которые зависят от вида функции потерь L. В частности для квадратичной функции потерь получается $c_1 = c_2 = 1$, если рассматривать $y_* = y_i$ и $y_{min} = E_D[y]$:

$$E_D\left[(y_i - y)^2\right] = (y_i - E_D[y])^2 + E_D\left[(E_D[y] - y)^2\right]$$

Введём понятия смещённого и несмещённого объектов. ***Несмещёнными*** называются те объекты, для которых $y_{min} = y_*$ и соответственно $B(x) = 0$, а ***смещёнными*** - $y_{min} \neq y_*$, а значит и $B(x) \neq 0$.

Таким образом, все объекты выборки X^L разделяются на две группы – смещённые (biased) и несмещённые (unbiased). Разделение объектов выборки на две таких группы играет важную роль. Будем отдельно считать вариацию для этих двух групп, для смещённых и несмещённых объектов соответственно: V_b и V_u. При этом увеличение вариации на несмещённых объектах увеличивает средние потери, а увеличение вариации на смещённых объектах, наоборот, уменьшает средние потери алгоритма. Следовательно, вариация будет определяться как разница между вариацией на несмещённых объектах и вариацией на смещённых объектах: $V = V_u - V_b$.

Рассмотрим двухклассовую задачу и любую действительно значимую функцию потерь, для которой $\forall_y L(y, y) = 0$ и $\forall_{y_1 \neq y_2} L(y_1, y_2) \neq 0$. Тогда коэффициенты выразятся через вероятность предсказания оптимального класса на объекте x - $P_D(y = y_*)$:

$$c_1 = P_D(y = y_*) - \frac{L(y_*, y)}{L(y, y_*)} P_D(y \neq y_*), \qquad c_2 = \begin{cases} 1, & \text{если } y_{min} = y_*; \\ -\frac{L(y_*, y_{min})}{L(y_{min}, y_*)}, & \text{если } y_{min} \neq y_*. \end{cases}$$

Для функции потерь 0-1 коэффициенты получаются:

$$c_1 = 1, \qquad c_2 = \begin{cases} 1, & \text{если } y_{min} = y_*; \\ -1, & \text{если } y_{min} \neq y_*. \end{cases}$$

Разложение функции потерь будет выглядеть следующим образом:

$$E_D[L(y_i, y)] = N(x) + B(x) + V_u(x) - V_b(x).$$

В случае многоклассовой задачи для функции потерь 0-1 коэффициенты получаются:

$$c_1 = P_D(y = y_*) - P_D(y \neq y_*) P_{y_i}(y = y_i | y_* \neq y_i),$$

$$c_2 = \begin{cases} 1, & \text{если } y_{min} = y_*; \\ -P_D(y = y_* | y \neq y_{min}), & \text{если } y_{min} \neq y_*. \end{cases}$$

Средние потери по всей выборки разделяются на шум, среднее смещение и среднюю вариацию:

$$E_{D,x_i}[L(y_i, y)] = E_{x_i}[c_1 N(x_i)] + E_{x_i}[B(x_i)] + E_{x_i}[c_2 V(x_i)].$$

Скалярная статистика - Вариация

Средняя вариация характеризует степень изменчивости результата обучения на данном объекте при варьировании состава обучающей выборки. Чем меньше средняя вариация, тем устойчивее алгоритм классификации.

Скалярная статистика - Смещение

Среднее смещение характеризует степень несогласованности данного объекта с методом обучения. Чем меньше смещение, тем лучше алгоритм подходит для решения конкретной задачи.

Полигон

В качестве функции потерь была взята функция 0-1. Так как оценка уровня шума представляет собой довольно трудоёмкую вычислительную задачу, было принято решение принять, следуя работам Kohavi, Wolpert (1996) [8] и Domingos (2000) [4], что уровень шума равен нулю $\boldsymbol{N}(\boldsymbol{x}) = \boldsymbol{0}$. Данное предположение не ограничивает общности исследования, поскольку основной интерес представляет собой разложение ошибки на вариацию и смещение.

Теорема П. Домингеса [4] о разложение ошибки на вариацию и смещение:

$$\boldsymbol{CV}\left(\mu, \boldsymbol{X^L}\right) = \frac{\boldsymbol{1}}{\boldsymbol{L}} \sum_{\boldsymbol{x} \in \boldsymbol{X^L}} \boldsymbol{B}(\boldsymbol{x}) + \frac{\boldsymbol{1}}{\boldsymbol{L}} \sum_{\substack{\boldsymbol{x} \in \boldsymbol{X^L} \\ \boldsymbol{B}(\boldsymbol{x})=\boldsymbol{0}}} \boldsymbol{V}(\boldsymbol{x}) - \frac{\boldsymbol{1}}{\boldsymbol{L}} \sum_{\substack{\boldsymbol{x} \in \boldsymbol{X^L} \\ \boldsymbol{B}(\boldsymbol{x})\neq\boldsymbol{0}}} \boldsymbol{V}(\boldsymbol{x})$$

С помощью процедуры скользящего контроля для каждого объекта выборки были подсчитаны средние по всем разбиениям значения ошибки, смещения и вариации. Затем объекты были отсортированы по увеличению средней ошибки.

Значение смещения на объекте может быть либо 0, либо 1, так как каждый объект либо правильно классифицируется большинством алгоритмов, либо на нём совершается ошибка. В результате все объекты были разделены на две группы: смещённые и несмещённые. Смещённые – те объекты, которые неправильно классифицировались большинством алгоритмов, а несмещённые – наоборот, классифицировались правильно.

На рисунке 2 показан график разложения ошибки на вариацию и смещение. На нём видно, что на несмещённых объектах ошибка состоит только из вариации, а на смещённых объектах ошибка состоит из разности между смещением и вариацией. Таким образом, вариация на смещённых объектах уменьшает среднюю ошибку алгоритма классификации. Это хорошо согласуется с теорией, описанной в статье P. Domingos (2000) [4].

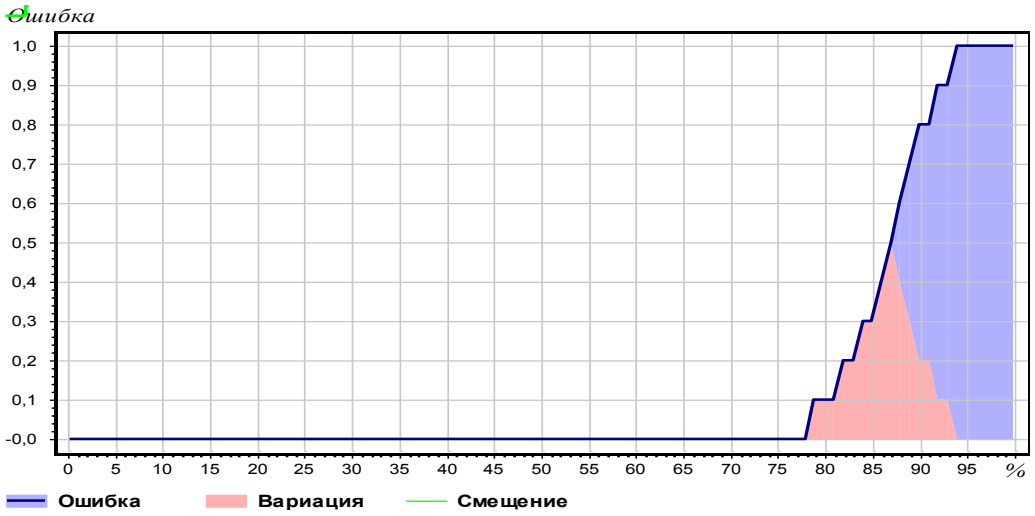


Рисунок 2. Разложение ошибки на смещение и вариацию

Описание графика:

- По оси абсцисс отложены объекты, отсортированные по возрастанию ошибки на них.
- По оси ординат отложены ошибки на объектах.
- Синим цветом с тёмно-синей окантовкой отмечены средние ошибки на объектах.
- Красным цветом отмечен модуль вариации на объектах. При этом при смещении равном 0 вариация положительна, а при смещении равном 1 вариация будет отрицательной.
- Зелёным цветом отмечено значение смещения на объектах.

Анализ графика:

1. *Оценка эффективности метода обучения.*

Большое количество смещённых объектов говорит о плохом качестве классификации. Смещённые объекты на графике – это те объекты, для которых смещение (линия зелёного цвета) равно 1. Несмещённые объекты – это те, для которых смещение равно 0.

2. *Оценка ширины зоны неустойчивой классификации (то есть количество пограничных объектов).*

Пограничный объект лежит на границе между классами и поэтому при разных обучающих выборках относится то к одному классу, то к другому. Пограничные объекты на графике – это те объекты, для которых вариация близка к значению 0,5 (закрашена красным цветом).

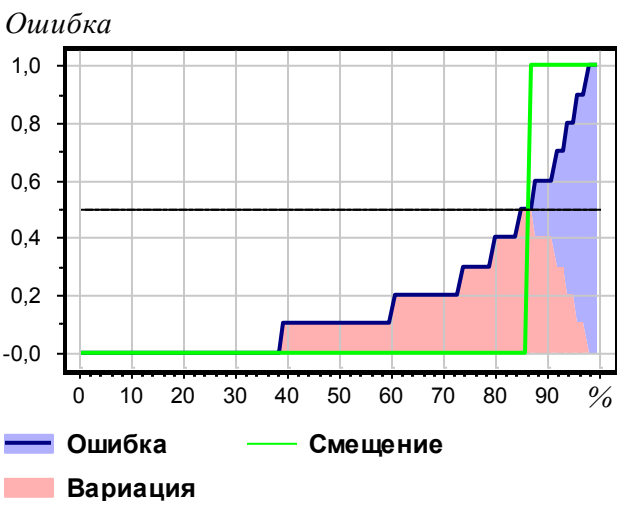
Чем больше пограничных объектов, тем хуже разделяются классы в задаче и тем шире зона неустойчивой классификации.

В многомерных признаковых пространствах достаточно трудно оценить, какие объекты являются пограничными, в том время как данная методика справляется с этой задачей и позволяет оценить ширину пограничной зоны.

Эксперимент

С помощью системы Полигон были проведены следующие эксперименты на задачах из репозитория UCI [14] и алгоритмах из системы Weka [15].

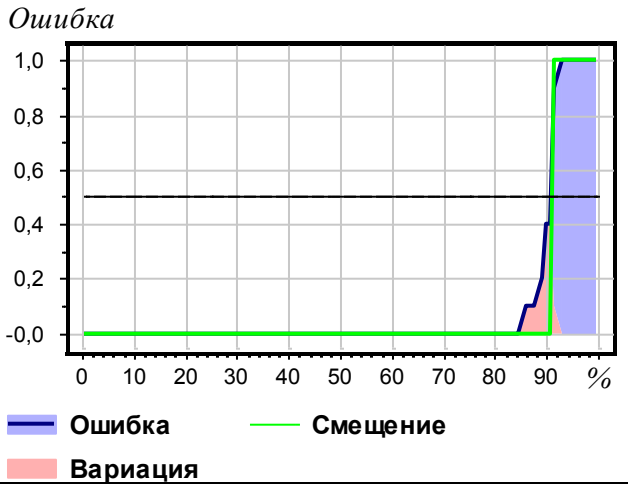
Результаты:



Метод Обучения: trees.RandomTree
Задача: Australian
Выборка: Все классы

1. Достаточно хорошее качество классификации, чуть менее 15% объектов являются смещёнными

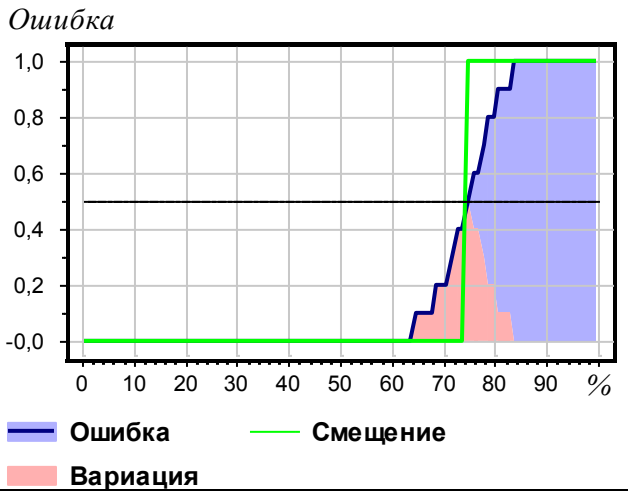
2. Зона неустойчивой классификации достаточно широкая (около 20%), поэтому можно говорить о неустойчивости метода обучения



Метод Обучения: rules.JRip
Задача: Echocardiogram
Выборка: Все классы

1. Хорошее качество классификации, менее 10% объектов являются смещёнными

2. Метод обучения достаточно устойчив, так как зона неустойчивости очень узкая (менее 3%)



Метод Обучения: functions.SMO
Задача: German
Выборка: Все классы

1. Плохое качество классификации, более 25% объектов являются смещёнными

2. Метод обучения достаточно устойчив, зона неустойчивости около 7%

Карта ошибок

Карта ошибок позволяет оценить устойчивость алгоритма классификации в зависимости от состава обучающей выборки.

Метод обучения это функция, на вход которой подаётся обучающая выборка, а на выходе получается алгоритм классификации. С помощью скользящего контроля для метода обучения получается несколько алгоритмов классификации. Для каждого из алгоритмов подсчитываются ошибки на обучающей выборке и на контрольной выборке.

На карте ошибок по оси абсцисс отложена ошибка алгоритма на обучающей выборке, а по оси ординат ошибка на контрольной выборке. Точками обозначены алгоритмы классификации.

Полигон

Воспользовавшись процедурой скользящего контроля, система Полигон генерирует *t*q* алгоритмов классификации после обучения на различных обучающих выборках. Для каждого такого алгоритма подсчитывается ошибка на контрольной выборке и на обучающей выборке.

Карта ошибок получается следующим образом, по оси абсцисс откладывается ошибка, которую допустил алгоритм на обучающей выборке, а по оси ординат – ошибка на контрольной выборке.

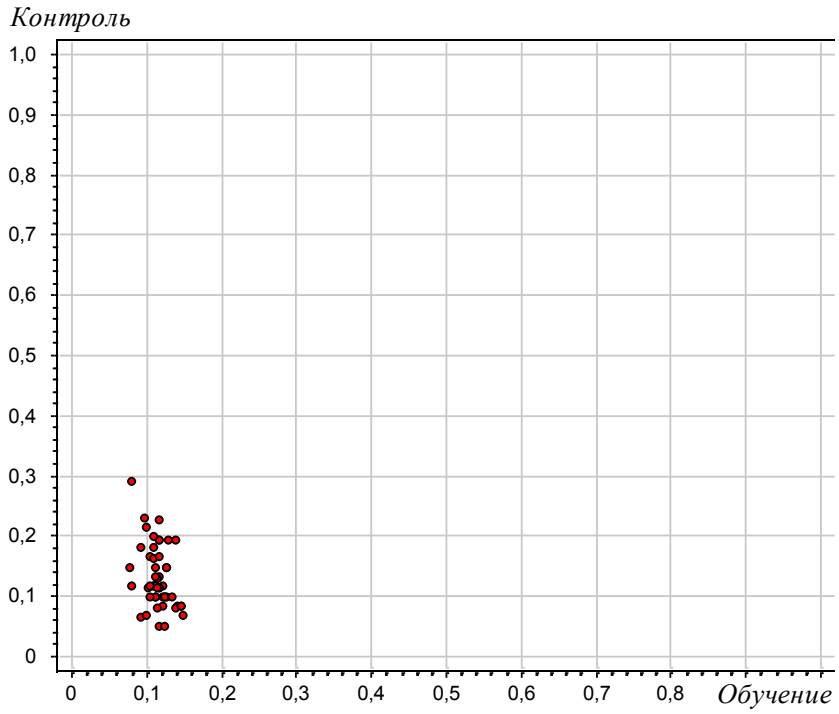


Рисунок 3. Карта ошибок

Анализ графика:

1. *Оценка устойчивости метода обучения в зависимости от состава обучающей выборки.*

Если алгоритмы на графике расположены достаточно кучно, то можно говорить о том, что метод обучения устойчив к составу обучающей выборки. То есть, изменения в обучающей выборке не существенно меняют ответы алгоритма. С другой стороны, если на графике виден большой разброс, то это говорит о неустойчивости метода обучения.

2. *Оценка переобучения алгоритма.*

Переобучение – нежелательное явление, которое возникает, когда средняя ошибки алгоритма на контрольной выборке существенно выше, чем средняя ошибка на обучающей выборке. Переобучение обычно возникает при использовании избыточно сложных моделей.

Для оценки переобучения необходимо обратить внимание на то, сколько алгоритмов расположены выше диагонали (в этом случае ошибка на контрольной выборке больше, чем ошибка на обучающей выборке). Если большинство алгоритмов лежит выше диагонали, то это говорит о большой вероятности переобучения.

3. *Оценка эффективности метода обучения.*

При хорошем качестве классификации метода обучения все алгоритмы должны обладать малыми ошибками как на контрольной выборке, так и на обучающей выборке, а, следовательно, лежать вблизи начала координат на карте ошибок.

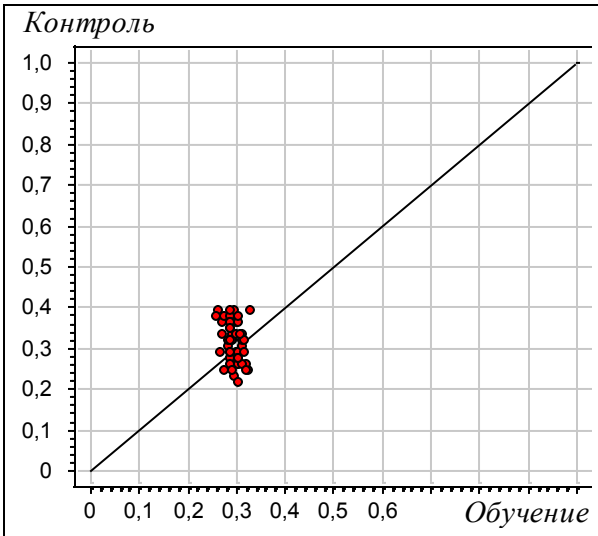
4. *Оценка отделимости классов.*

Если построить карту ошибок отдельно по классам, то можно оценить качество классификации каждого класса и отделимость класса от всех остальных.

Эксперимент

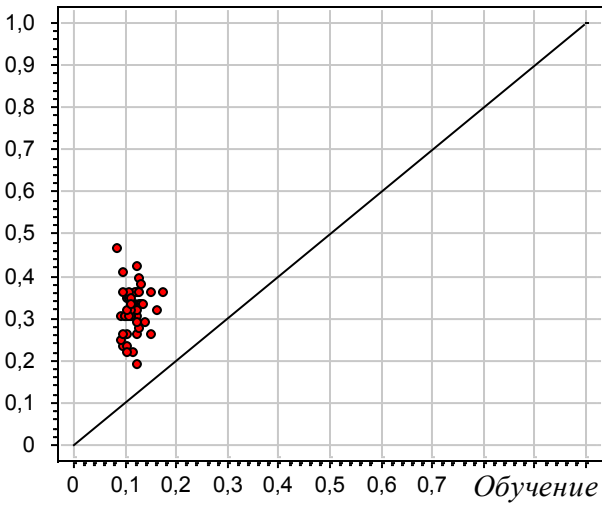
С помощью системы Полигон были проведены следующие эксперименты на задачах из репозитория UCI [14] и алгоритмах из системы Weka [15].

Результаты:



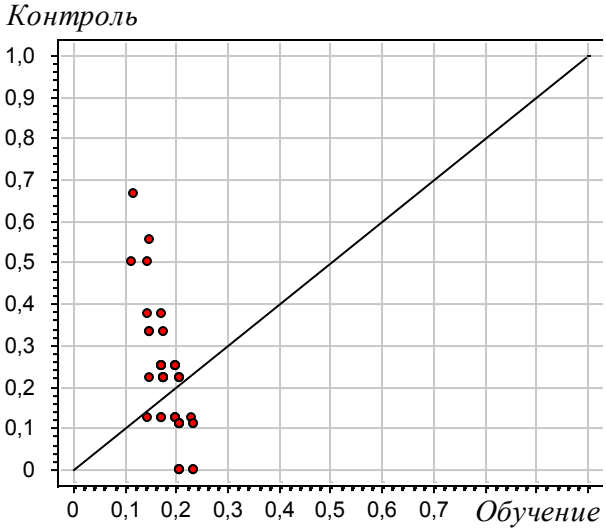
Метод Обучения: functions.Logistic
Задача: Liver_Disorders
Выборка: Все классы

1. Устойчивый алгоритм, малая зависимость от состава обучающей выборки
2. Эффект переобучения не наблюдается
3. Алгоритм обладает плохим качеством классификации. Средняя ошибка около 30%



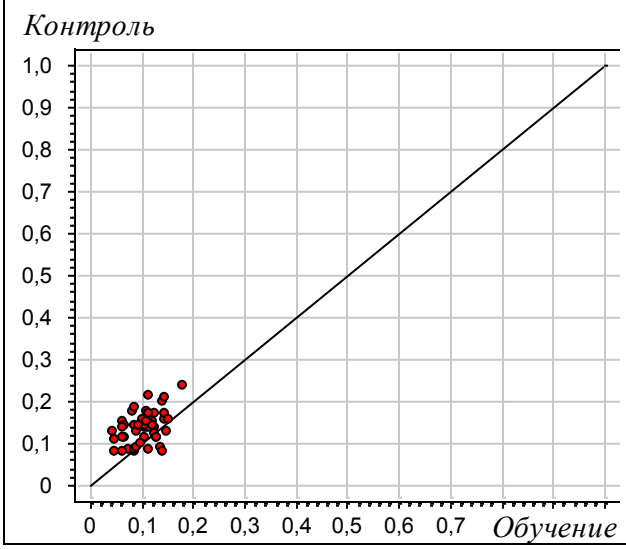
Метод Обучения: meta.Bagging
Задача: Liver_Disorders
Выборка: Все классы

1. Достаточно устойчивый метод обучения, малая зависимость от состава обучающей выборки
2. Наблюдается эффект переобучения, так как все алгоритмы расположены выше диагонали (ошибка на контроле больше ошибки на обучении)
3. Достаточно плохое качество классификации. Не смотря на малую ошибку на обучении, ошибка алгоритма на контроле составляет более 30%



Метод Обучения: bayes.NaiveBayes
Задача: Echocardiogram
Выборка: Класс 1

1. Неустойчивый метод обучения, сильная зависимость от состава обучающей выборки
2. Наблюдается эффект переобучения, так как большинство алгоритмов лежат выше диагонали
3. Плохое качество классификации. Несмотря на малую ошибку на обучении, ошибка алгоритма на контроле доходит до 60%



Метод Обучения: rules.JRip
Задача: German
Выборка: Класс 0

1. Достаточно устойчивый метод обучения, малая зависимость от состава обучающей выборки

2. Малая вероятность эффекта переобучения

3. Хорошее качество классификации. Средние ошибки на контрольной выборке и на обучающей выборке не превосходят 15%

ROC-кривая

Roc-кривая (Receiver Operator Characteristic) – кривая, которую наиболее часто используют для представления результатов бинарной классификации в машинном обучении. ROC-кривая - очень полезный инструмент для визуализации и оценки качества моделей обучения.

Рассмотрим задачу с двумя классами. Назовём один из классов классом с положительными исходами, а второй – с отрицательными исходами. При этом будем считать, что у классификатора есть некий параметр – порог, варьируя который мы сможем получать различные разбиения на два класса. Полученные данные будем выводить в качестве графика как зависимость правильно классифицируемых положительных объектов от ошибок на положительных объектах.

Roc-кривые также используют для многоклассовых задач. В этом случае можно построить n разных Roc-кривых для каждого класса. Для каждой roc-кривой в качестве положительного класса берётся один из множества классов, а все остальные объединяются как отрицательные.

Теория

Введём следующие термины:

- *Истинно положительные примеры (True Positives, TP)* – верно классифицированные положительные примеры.
- *Истинно отрицательные примеры (True Negatives, TN)* – верно классифицированные отрицательные примеры.
- *Ложно отрицательные примеры (False Negatives, FN)* – положительные примеры, классифицированные как отрицательные (ошибка 1ого рода). “Ложный пропуск” – когда интересующее нас событие ошибочно не обнаруживается.
- *Ложно положительные примеры (False Positives, FP)* – отрицательные примеры, классифицированные как положительные (ошибка 2ого рода). “Ложное обнаружение” – когда при отсутствии события, оно ошибочно обнаруживается.

При анализе roc-кривых оперируют не абсолютными, а относительными величинами:

$$\boldsymbol{TPR}\ (\boldsymbol{True\ Positives\ Rate}) = \frac{\boldsymbol{TP}}{\boldsymbol{TP} + \boldsymbol{FN}}\ \boldsymbol{100\%}$$

$$\boldsymbol{FPR}\ (\boldsymbol{False\ Positives\ Rate}) = \frac{\boldsymbol{FP}}{\boldsymbol{FP} + \boldsymbol{TN}}\ \boldsymbol{100\%}$$

Введём два главных параметра, с чьей помощью можно будет оценивать эффективность алгоритмов: *чувствительность (sensitivity)* и *специфичность (specificity).*

$$\boldsymbol{Se(Sensitivity) = TPR = \frac{TP}{TP + FN}\ 100\%}$$

$$\boldsymbol{Sp(Specificity) = 100 - FPR = \frac{TN}{FP + TN}\ 100\%}$$

Модель с высокой чувствительностью часто даёт истинный результат при наличии положительного исхода, то есть обнаруживает положительные примеры.

В то время как, модель с высокой специфичностью чаще даёт истинный результат при наличии отрицательного исхода, то есть обнаруживает отрицательные примеры.

ROC-кривая строится следующим образом:

1. Для каждого значения порога отсечения, которое меняется от 0 до 1 с некоторым шагом, рассчитываются значения чувствительность Se и специфичности Sp.
2. Строится график зависимости Se (чувствительности) по оси ординат от 1 - Sp (1 - специфичность) по оси абсцисс.

В результате получается ROC кривая:

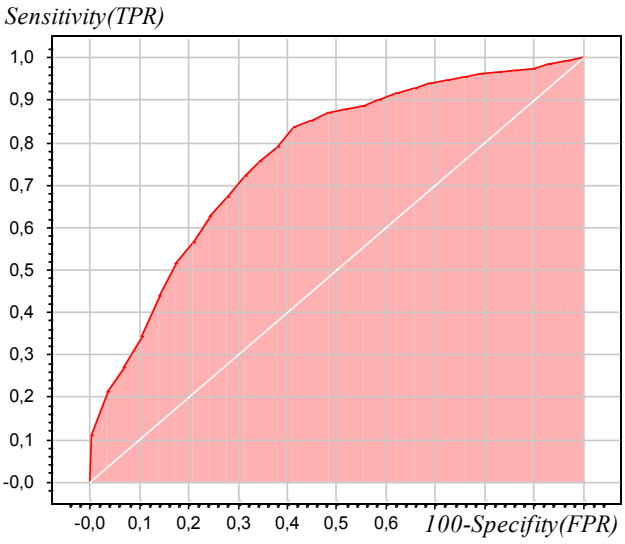


Рисунок 4. ROC-кривая

Обычно графики ROC-кривых дополняют прямой $\boldsymbol{y = x}$, которая характеризует “бесполезный” классификатор, который совсем не различает классы.

ROC-кривая для идеального классификатора проходит через левый верхний угол, где доля истинно положительных случаев составляет 100%, а доля ложно положительных случаев равна нулю. Поэтому чем ближе полученная кривая к верхнему левому углу, тем больше предсказательная способность модели.

Есть несколько способов сравнения алгоритмов с помощью ROC-кривых. Первый заключается в визуальном сравнении двух моделей, для которого необходимо наложить масштабированные ROC-кривые друг на друга. После этого можно анализировать их взаимное расположение. Кривая, расположенная выше и левее, соответствует более эффективной модели с лучшей предсказательной способностью.

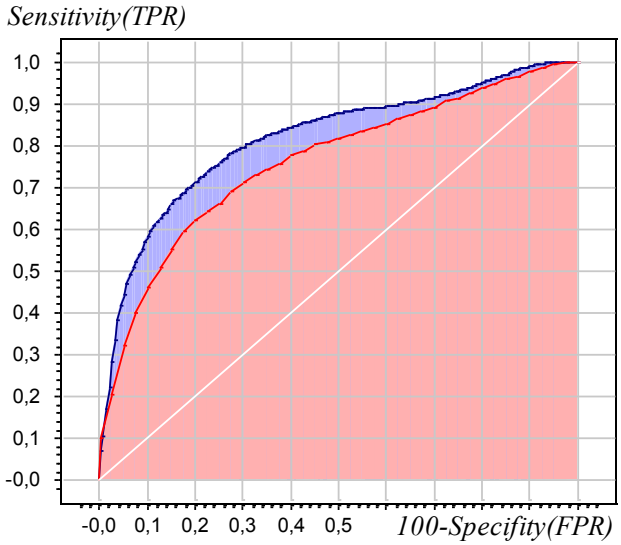


Рисунок 5. Сравнение ROC-кривых

Однако визуальное сравнение ROC-кривых не всегда позволяет дать ответ на вопрос, какая из них является более эффективной.

Скалярная статистика - Площадь под roc-кривой (AUC)

В качестве другого способа сравнения можно использовать оценку площади под кривой *(Area Under Curve, **AUC**)*. Площадь изменяется от 0 до 1.0, но на самом деле в качестве нижней границы можно взять 0.5 , как для “бесполезного” классификатора и в качестве верхней 1.0 , как для “идеального” классификатора.

$$\boldsymbol{AUC} = \int \boldsymbol{f(x)dx}$$

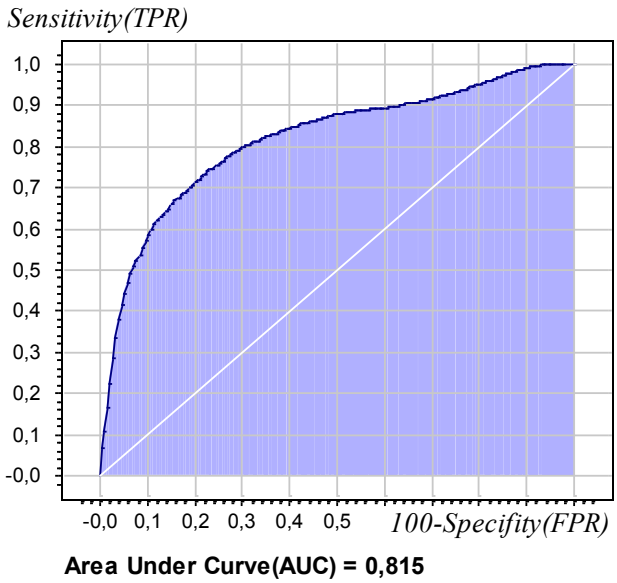


Рисунок 6. Площадь под ROC-кривой

Показатель ***AUC*** предназначен для сравнения моделей, и чем он больше, тем больше предсказательная способность модели.

В литературе приводится следующая таблица для оценки эффективности модели через значения AUC:

Интервал AUC	Качество модели
0.9 – 1.0	Отличное
0.8 – 0.9	Очень хорошее
0.7 – 0.8	Хорошее
0.6 – 0.7	Среднее
0.5 – 0.6	Неудовлетворительное

Несбалансированность классов

ROC-кривые обладают очень привлекательным свойством: они не чувствительны к изменениям в распределении классов. Это происходит из-за того, что ROC-кривые основаны на чувствительности и специфичности, которые являются относительными величинами. В случае несбалансированных классов (так называемое *class skew*), когда объектов одного класса во много раз больше, чем объектов другого класса, ROC-кривая будет выглядеть, также как и в случае со сбалансированными классами.

Усреднение

Важным фактором в ROC-анализе является усреднение нескольких кривых. Пусть у нас есть n контрольных выборок $\boldsymbol{T_1}, \boldsymbol{T_2}, \ldots, \boldsymbol{T_n}$, для каждой из этих выборок мы строим свою ROC-кривую. Теперь у нас есть n ROC-кривых и нам надо их усреднить. Можно объединить все контрольные выборки и вычислить комбинированную ROC-кривую. Но более наглядным будет использовать пороговое усреднение (threshold averaging). Этот метод генерирует набор порогов. Для каждого порога и для каждой выборки находится соответствующая точка на ROC-кривой. Затем происходит усреднение точек, соответствующих одному порогу. На график наносятся усреднённые точки, соединив которые мы получаем усреднённую ROC-кривую. График дополняется доверительными интервалами по обеим осям.

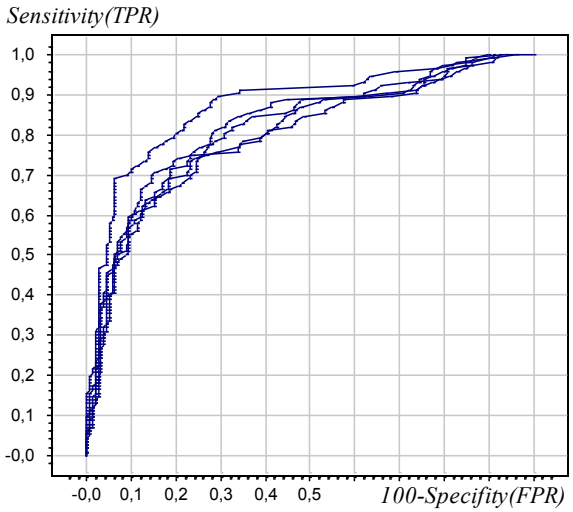


Рисунок 7. ROC-кривые пяти различных выборок.

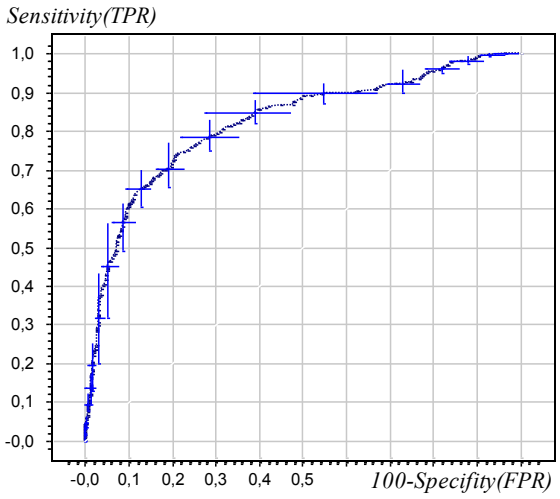


Рисунок 8. Усреднённая ROC-кривая.

Многоклассовая задача

Для задач с количеством классов большим двух вычисления будут очень сложными, и будет потеряна всякая возможность для восприятия данных, а визуализация является главным достоинством ROC-

кривых. Одним из методов анализа ROC-кривой для n классов является создание n разных ROC-кривых для каждого из классов. Пусть $\boldsymbol{Y}$– это набор всех классов, для i-ой ROC-кривой определим $\boldsymbol{y_i}$ как положительный класс, а все остальные – как отрицательный, то есть:

$$\boldsymbol{P_i} = \boldsymbol{y_i}$$

$$\boldsymbol{N_i} = \cup_{\boldsymbol{j \neq i}} \boldsymbol{y_i} \in \boldsymbol{Y}$$

Для многоклассовых задач также можно вычислить $\boldsymbol{AUC_{total}}$, используя метод, предложенный Провостом и Домингесом в 2001 году в своей работе про вероятностные оценки деревьев [10]. Для этого для каждого класса $\boldsymbol{y_i} \in \boldsymbol{Y}$ вычислим $\boldsymbol{AUC}(\boldsymbol{y_i})$, потом посчитаем взвешенную сумму площадей под кривыми по априорной вероятности распределения по классам в данных.

$$\boldsymbol{AUC^1_{total}} = \sum_{\boldsymbol{y_i \in Y}} \boldsymbol{AUC}(\boldsymbol{y_i}) \cdot \boldsymbol{p}(\boldsymbol{y_i})$$

Этот способ разбиения классов удобен, однако он имеет недостаток: чувствительность к изменению в распределении классов и стоимостях ошибок, поэтому $\boldsymbol{AUC^1_{total}}$ имеет тот же недостаток. Несмотря на этот недостаток, данный метод может работать хорошо и обеспечивать разумную гибкость результатов.

Хэнд и Тилл в 2001 году предложили другой подход к подсчёту $\boldsymbol{AUC_{total}}$ [7], стараясь избежать недостатков, которым подвергнут подход Провоста и Домингеса. Их подход основан на том факте, что AUC эквивалентен вероятности того, что классификатор будет больше ценить случайно выбранный положительный пример, чем отрицательный. Из этого вероятностного подхода они вывели формулу, которая измеряет попарную различимость классов.

$$\boldsymbol{AUC^2_{total}} = \frac{\boldsymbol{2}}{\boldsymbol{|Y|(|Y|-1)}} \sum_{\boldsymbol{\{y_i, y_j\} \in Y}} \boldsymbol{AUC}(\boldsymbol{y_i}, \boldsymbol{y_j})$$

Хотя эта формулировка нечувствительна к изменениям в распределении классов, но очень сложно визуализировать поверхность, площадь под которой вычисляется.

Полигон

Воспользовавшись процедурой скользящего контроля, система Полигон вычисляет roc-кривые для каждого из $t*q$ алгоритмов на контрольных выборках, на обучающих выборках для каждого класса $\boldsymbol{y_i} \in \boldsymbol{Y}$. Применив пороговое усреднение, мы получили для каждого класса усреднённую roc -кривую на обучении и на контроле. Кроме усреднённых roc-кривых также вычисляются вариационные кривые, которые играют роль доверительных интервалов.

Стоит отметить, что в случае двухклассовой задачи roc-кривые для двух классов различаются только отражением относительно диагонали $\boldsymbol{y} = -\boldsymbol{x}$. Действительно, ошибка первого рода для нулевого класса является ошибкой второго рода для первого класса и, наоборот, для ошибки второго рода.

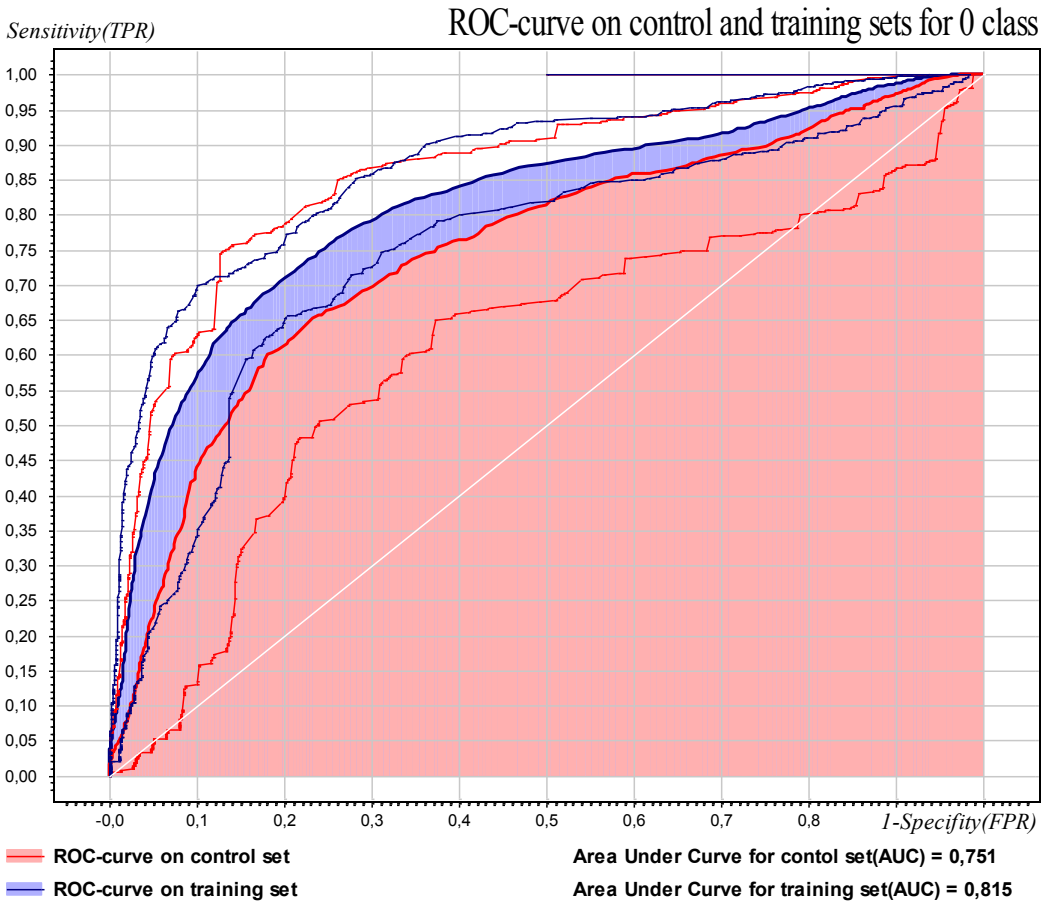


Рисунок 9. ROC-кривая на контрольной и на обучающей выборках
с доверительными интервалами для нулевого класса

Описание графика:

- По оси абсцисс отложена *ошибка второго рода* (FPR, 1 – специфичность).
- По оси ординат отложена *1- ошибка первого рода* (TPR, чувствительность).

- Тёмно-красная линия с красной заливкой – ROC-кривая на контроле.
- Тёмно-синяя линия с синей заливкой – ROC-кривая на обучении.
- Внизу под графиком приведено значение площади под кривой (AUC) для обучающей выборки и контрольной выборки.
- Тонкая белая линия обозначает диагональ (“бесполезный классификатор”).
- Тонкими красными линиями показаны доверительные интервалы на контроле.
- Тонкими синими линиями показаны доверительные интервалы на обучении.

Анализ графика:

1. *Оценка эффективности метода обучения.*

Чем больше площадь под roc -кривой на контроле и чем ближе кривая проходит к левому верхнему углу, тем лучше данный метод обучения подходит для решения данной задачи.

2. *Оценка переобучения алгоритма.*

При большой разнице между кривыми по обучающей выборке и по контрольной выборке можно говорить об эффекте переобучения.

На графике тонкие линии показывают доверительные интервалы, красного цвета - для контрольной выборки и синего цвета - для обучающей выборки. Если между нижней синей кривой и верхней красной кривой есть промежуток, то можно говорить, что в данном случае имеет место эффект переобучения.

3. *Сравнение roc -кривых для разных классов позволяет оценить отделимость классов.*

У надёжно отделимого класса roc -кривая на контрольной выборке расположена ближе к левому верхнему углу. В этом случае почти все объекты из этого класса распознаются верно, и совершается минимальное количество ложных обнаружений (когда объекты не из данного класса приписываются к нему).

Плохо отделимый класс имеет кривую близкую к диагонали (это “бесполезный” классификатор, который не видит разницу между классами).

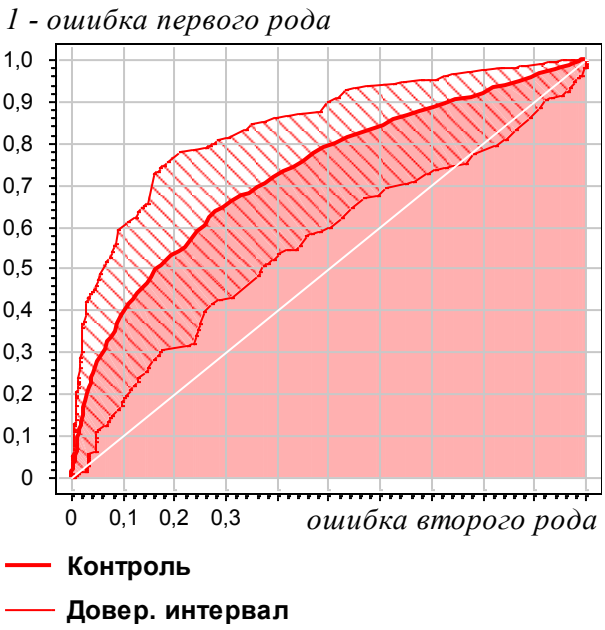
4. *Доверительные интервалы для roc-кривой на контроле позволяют оценить устойчивость метода обучения.*

На графике тонкие красные линии показывают доверительные интервалы для roc-кривой на контрольной выборке. Если доверительные интервалы достаточно велики, то это говорит о неустойчивости метода обучения, то есть о сильной зависимости от состава обучающей выборки.

Эксперименты

С помощью системы Полигон были проведены следующие эксперименты на задачах из репозитория UCI [14] и алгоритмах из системы Weka [15].

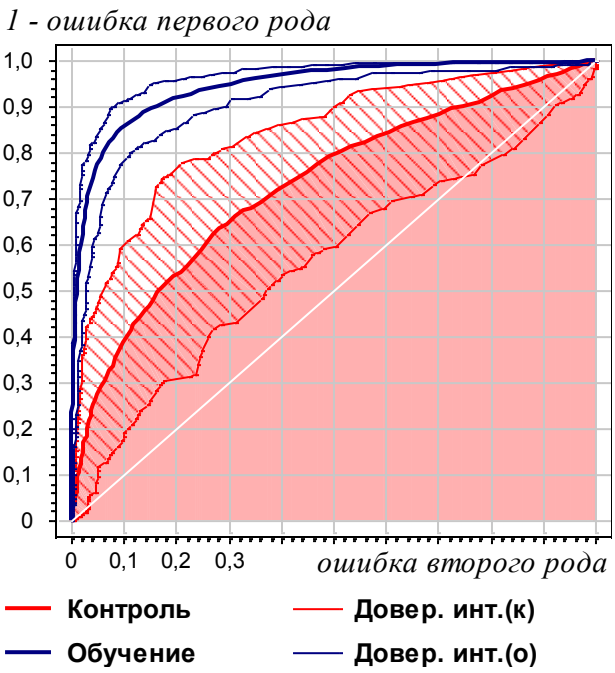
Результаты:



Метод Обучения: meta.Bagging
Задача: Liver_Disorders
Выборка: Класс 0

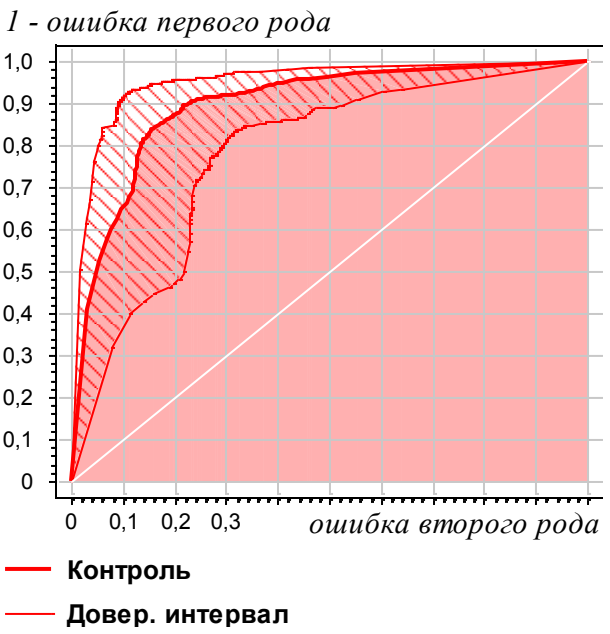
1. Среднее качество классификации, площадь под кривой (AUC) равна 0,754

2. Неустойчивый метод обучения, так как roc-кривые на контрольной выборке, а, следовательно, и качество классификации сильно зависят от обучающей выборки



Метод Обучения: meta.Bagging
Задача: Liver_Disorders
Выборка: Класс 0

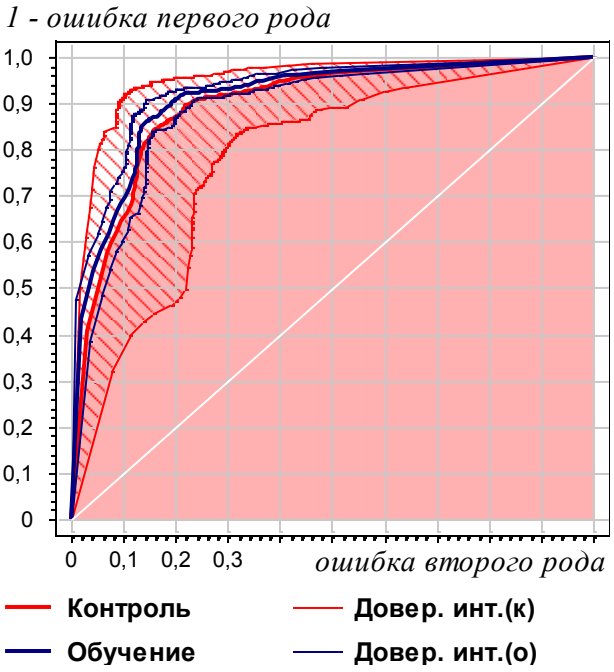
1. Наблюдается эффект переобучения, так как кривая на обучении намного ближе к левому верхнему углу, чем кривая на контроле. Разница между площадью под кривой на обучении (0,950) и на контроле (0,754) очень большая (0,196)



Метод Обучения: bayes.NaiveBayes
Задача: Heart Disease
Выборка: Класс 0

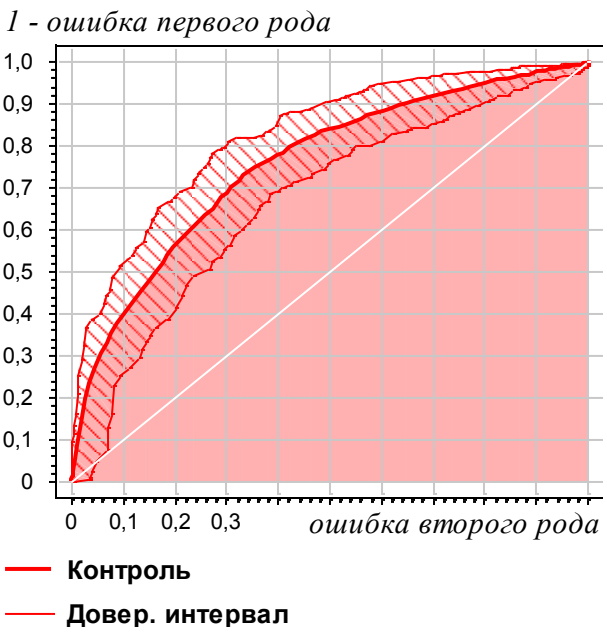
1. Хорошее качество классификации, площадь под кривой (AUC) равна 0,906

2. Достаточно устойчивый метод обучения, так как доверительный интервал для кривой достаточно узкий



Метод Обучения: bayes.NaiveBayes
Задача: Heart Disease
Выборка: Класс 0

1. Эффект переобучения отсутствует. Roc-кривые на обучении и на контроле расположены близко к друг другу. Разница между площадью под кривой на обучении (0, 914) и на контроле (0, 906) очень малая (0,008)



Метод Обучения: meta.Bagging
Задача: German
Выборка: Класс 0

1. Среднее качество классификации, площадь под кривой (AUC) равна 0,772

2. Устойчивый метод обучения, так как доверительный интервал достаточно узкий

Распределение отступов

Отступ показывает насколько близко объект подходит к границе своего класса. Это даёт информацию для анализа объектов из выборки. Объекты не являются равноценными, и их можно разделить на несколько категорий относительно каждого класса:

- *Шумовые* – объекты, лежащие за границей данного класса и имеющие большой отрицательный отступ.

- *Эталонные* – типичные представители класса, то есть объекты, лежащие глубоко в своём классе и имеющие большой положительный отступ.

- *Пограничные* – объекты, лежащие в зоне неуверенной классификации и имеющие отступ примерно равный нулю.

- *Остальные* – объекты, не попавшие ни в одну из вышеперечисленных категорий.

Если классифицируемый объект близок к эталонному объекту, то, скорее всего, он принадлежит тому же классу, что и эталонный объект. Шумовые объекты лучше исключить из выборки для улучшения качества классификации. Если же размер выборки велик и для улучшения работы алгоритма требуется уменьшить размер выборки, то следует уменьшить количество эталонных объектов в выборке. При этом необходимо следить, чтобы в каждом классе оставалось достаточное количество эталонных объектов. Распределение отступов на обучающих и контрольных выборках показывают, насколько надёжно данный алгоритм разделяет классы.

Теория

Воспользоваться распределением отступов можно, если алгоритм возвращает матрицу апостериорных вероятностей классов. Если же алгоритм осуществляет “жёсткую” классификацию, то график вырождается в константу.

Пусть $\Gamma_y(\boldsymbol{x})$ – оценка принадлежности объекта $\boldsymbol{x}$ к классу $\boldsymbol{y}$ и пусть алгоритм имеет вид:

$$\boldsymbol{a}(\boldsymbol{x}) = \boldsymbol{arg}\,\boldsymbol{\max_{y \in Y}} \Gamma_y(\boldsymbol{x})$$

Отступом объекта $\boldsymbol{x_i} \in \boldsymbol{X^L}$ называется величина

$$\boldsymbol{M}(\boldsymbol{a}, \boldsymbol{x_i}) = \Gamma_{y_i}(\boldsymbol{x_i}) - \max_{y \in Y \setminus y_i} \Gamma_y(\boldsymbol{x_i})$$

Отступ показывает, насколько близко объект x_i подходит к границе своего класса y_i. Если объект оказывается за границей класса, то отступ будет отрицателен и алгоритм на данном объекте допускает ошибку. Чем больше отступы, тем лучше качество классификации.

Пусть дана выборка X^L, рассмотрим класс $y \in Y$ и все объекты из выборки X^L, принадлежащие этому классу. Для каждого объекта из данного класса вычислим отступ $M(a, x_i)$. Таким образом, было получено множество значений отступов на объектах из класса y $\{M(a, x_i)\}_{i=1}^{L}$. После этого множество было упорядочено по возрастанию и теперь можно строить график зависимости отступа от объекта для класса y.

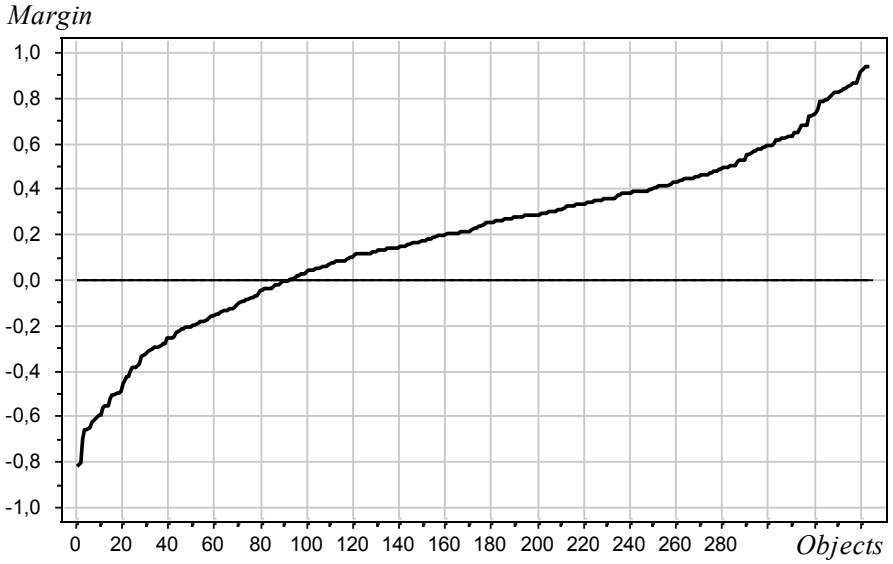


Рисунок 10. Распределение отступов

Правее всех находятся объекты с самыми большими отступами, и они являются эталонными объектами. Левее в отрицательной области по значениям отступа находятся шумовые объекты. Шумовые объекты лучше исключить из обучающей выборки для улучшения качества классификации.

Рисунок 11. Пример шумового объекта

Переобучение

Распределение отступов на обучающей и на контрольной выборке позволяет анализировать эффект переобучения. В случае если обучение происходило на шумовых или пограничных объектах, то алгоритм может хорошо настроиться по ним и давать на обучении малую ошибку и, соответственно, большой положительный отступ. Однако при этом на контроле алгоритм будет часто ошибаться и у многих объектов из контрольной выборки будет большой отрицательный отступ

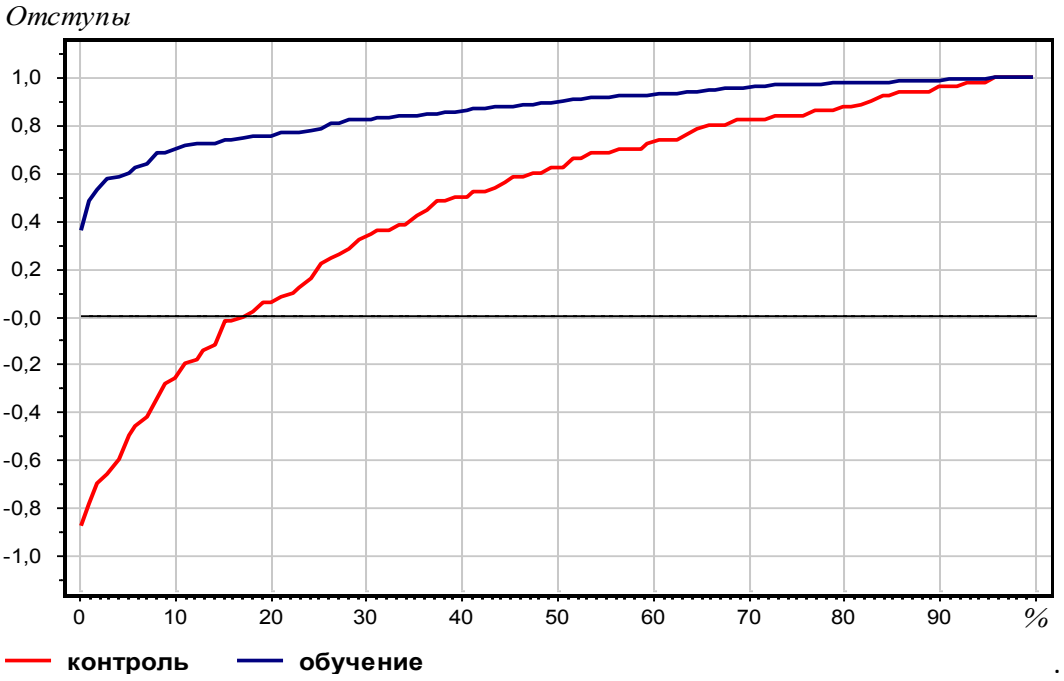


Рисунок 12. Эффект переобучения.
Синяя кривая – распределение отступов на обучении, красная – на контроле

Скалярная статистика – Доля пограничных объектов

Пограничным называется объект, который разными алгоритмами, полученными после обучения на разных обучающих выборках, классифицируется в разные классы, то есть данный объект принадлежит к зоне неуверенной классификации.

В системе Полигон пограничный объект - это тот объект, у которого нулевой отступ входит в доверительный интервал.

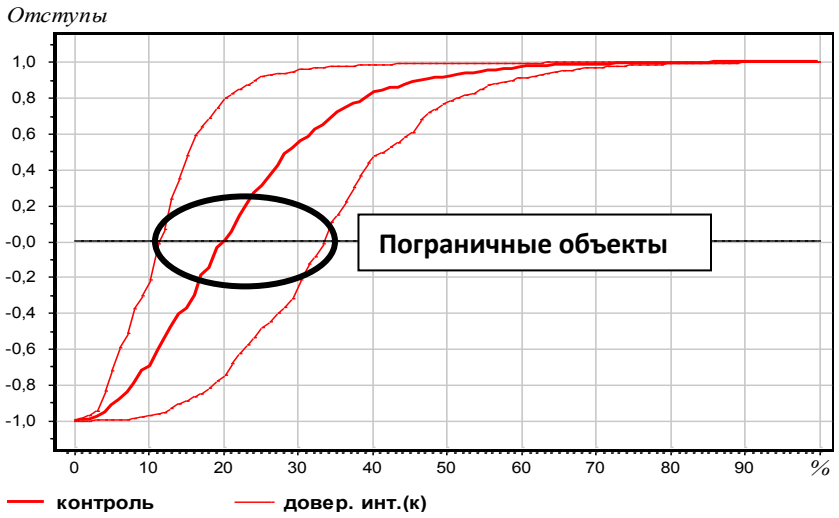


Рисунок 13. Распределение отступов, пограничные объекты

Скалярная статистика – Доля шумовых объектов

Шумовым называется объект, который лежит среди объектов другого класса, то есть за границей своего класса.

В системе Полигон шумовой объект - это тот объект, у которого весь доверительный интервал лежит меньше нуля. Чем меньше шумовых объектов, тем лучше работает алгоритм классификации и (или) тем лучше сама выборка (содержит мало выбросов).

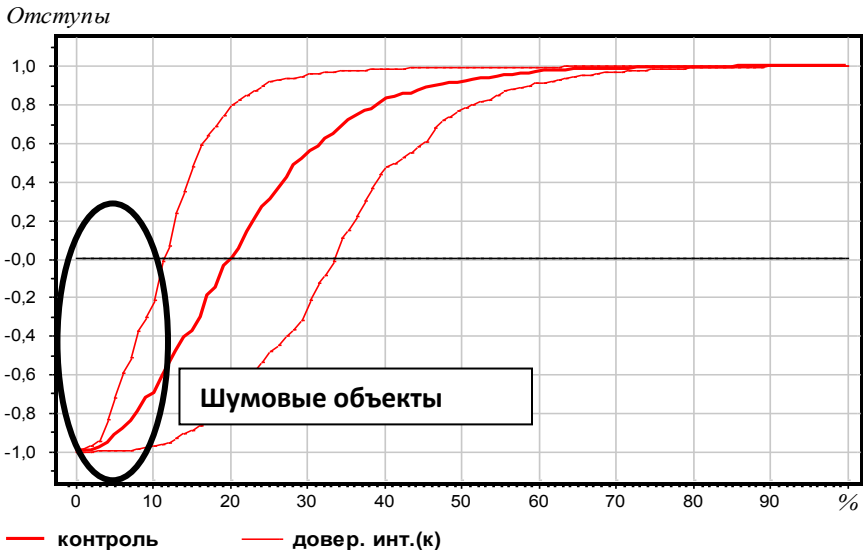


Рисунок 14. Распределение отступов, шумовые объекты

Скалярная статистика – Доля эталонных объектов

Эталонным называется объект, у которого отступ находится рядом с единицей, то есть данный объект почти всегда правильно классифицируется алгоритмом.

В системе Полигон эталонный объект - это тот объект, у которого весь доверительный интервал лежит около единицы.

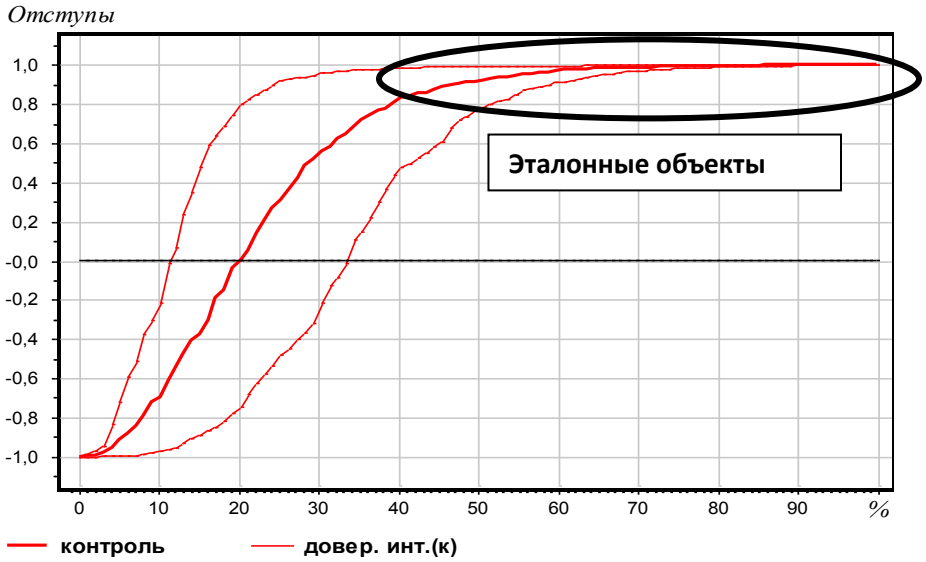


Рисунок 15. Распределение отступов, эталонные объекты

Полигон

Система Полигон для каждого из *t*q* алгоритмов, полученных при скользящем контроле, вычисляет кривую распределения отступов на контроле и на обучении отдельно для каждого класса $y_i \in Y$ и на всех классах вместе. После усреднения по всем алгоритмам получается распределение отступов и доверительные интервалы в виде двух кривых (Рисунок 11 и 12).

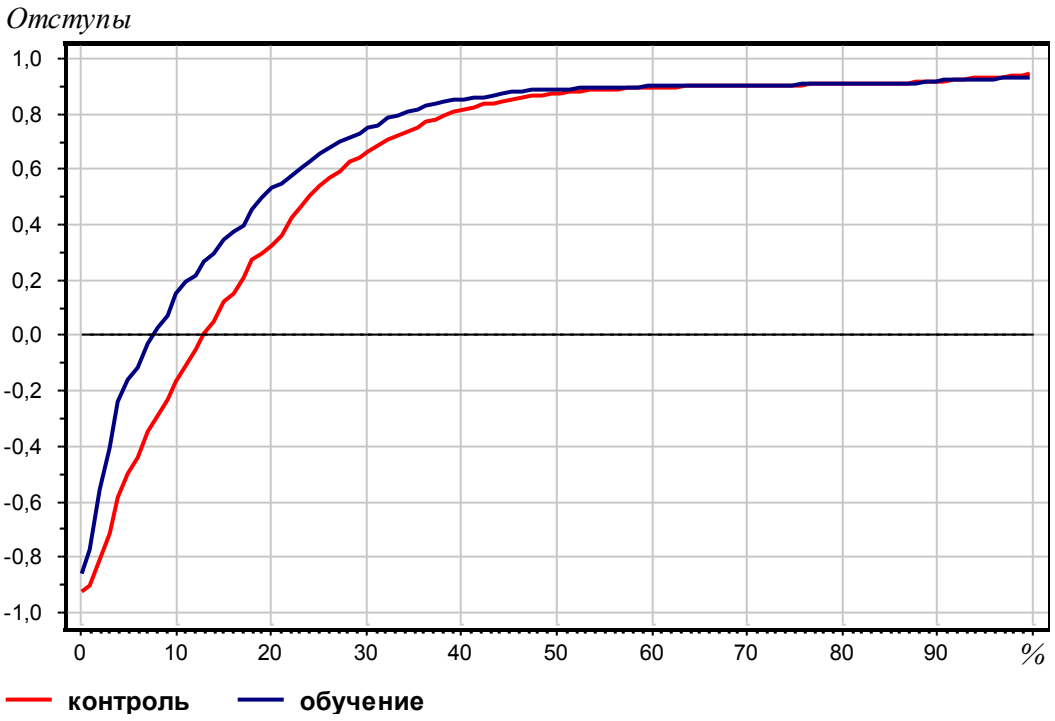


Рисунок 16. Распределение отступов на контроле и на обучении

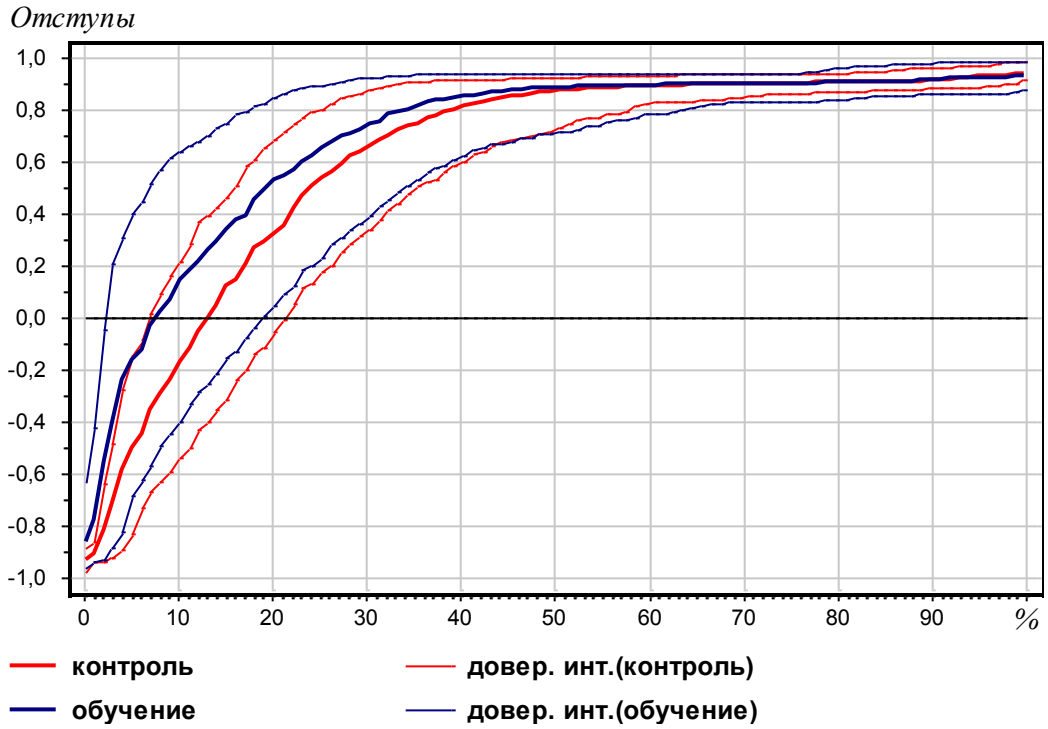


Рисунок 17. Распределение отступов на контроле и на обучении
с доверительными интервалами

Описание графиков:

- По оси абсцисс отложен номер объекта в процентах от количества объектов в выборке.
- По оси ординат отложен отступ данного объекта.
- Красная линия – кривая распределения отступов на контроле.
- Синяя линия – кривая распределения отступов на обучении.
- Тонкая чёрная линия соответствует нулевому отступу.
- Тонкими красными линиями показаны доверительные интервалы на контроле.
- Тонкими синими линиями показаны доверительные интервалы на обучении.

Анализ графиков:

1. *Оценка эффективности метода обучения.*

Малая доля объектов с отрицательными отступами (шумовые объекты) говорит о хорошем качестве классификации. Если же шумовых объектов достаточно много, то данный метод обучения не позволяет достичь хорошего качества классификации.

2. *Оценка переобучения алгоритма.*

При большой разнице между кривыми на контроле и на обучении можно говорить об эффекте переобучения.

На графике тонкие линии показывают доверительные интервалы, красного цвета - для контроля и синего цвета - для обучения. Если между нижней синей кривой и верхней красной кривой есть промежуток, то вероятность переобучения достаточно велика.

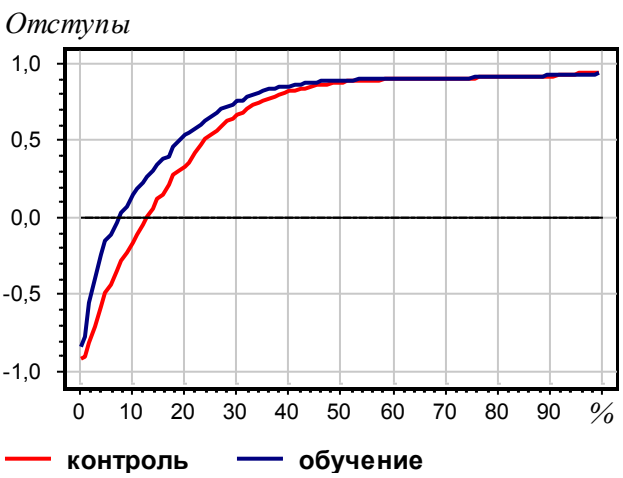
3. *Сравнение ROC-кривых на контроле для разных классов позволяет оценить отделимость классов.*

Класс, у которого большое количество шумовых объектов, является трудно отделимым. Класс, у которого шумовых и пограничных объектов мало, является надёжно отделимым.

Эксперименты

С помощью системы Полигон были проведены следующие эксперименты на задачах из репозитория UCI [14] и алгоритмах из системы Weka [15].

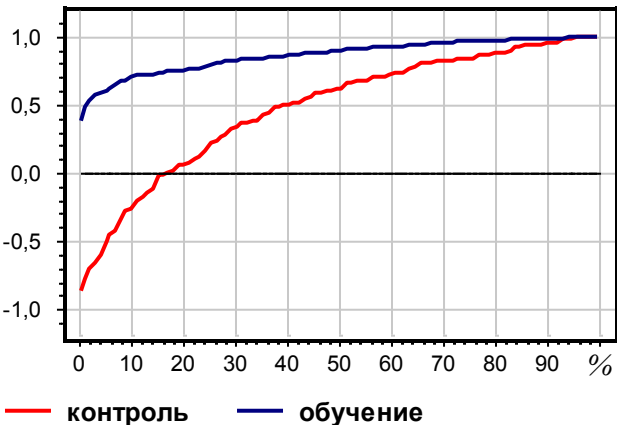
Результаты:



Метод Обучения: rules.JRip
Задача: Australian
Выборка: Все классы

1. Хорошее качество классификации, доля шумовых объектов в обучающей выборке менее 10%, а в контрольной менее 15%. Более 60% объектов являются эталонными

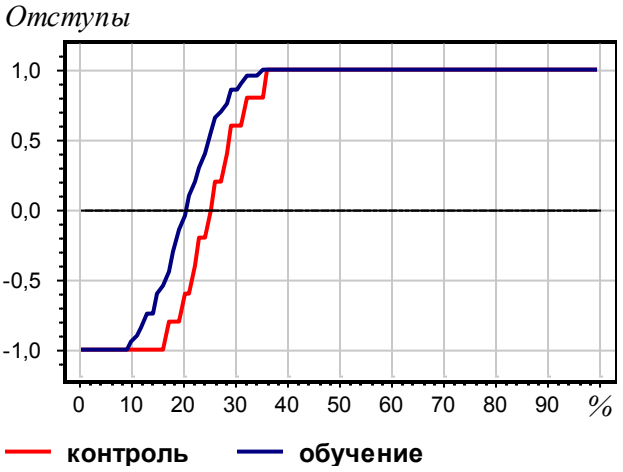
2. Кривые распределения отступов на обучении и на контроле лежат близко друг от друга, поэтому вероятность переобучения достаточно мала



Метод Обучения: trees.RandomForest
Задача: Heart Disease
Выборка: Все классы

1. Среднее качество классификации, на контроле 15% объектов являются шумовыми, а доля эталонных объектов менее 30%

2. Кривые распределения отступов на обучении и на контроле лежат далеко друг от друга, поэтому вероятность переобучения очень велика



Метод Обучения: functions.SMO
Задача: German
Выборка: Все классы

1. Среднее качество классификации, на контроле и на обучении более 20% объектов являются шумовыми

2. Кривые распределения отступов на обучении и на контроле расположены близко друг к другу, поэтому вероятность переобучения очень мала

Кривая обучения

Кривая обучения позволяет оценить необходимую длину обучающей выборки для максимизации качества работы алгоритма. То есть для полного использования информации, содержащейся в генеральной выборке.

Кривая обучения – это зависимость средней ошибки, которую допускает алгоритм на контрольной выборке, от длины обучающей выборки.

Полигон

Система Полигон генерирует специальный набор обучающих выборок разной длины: от 10% до 90% от длины генеральной выборки. Для уменьшения зависимости от конкретной выборки разбиения повторяются *t* раз.

На каждой выборке подсчитываются средние ошибки на контроле и на обучении. После чего для кривой обучения строится график зависимости ошибки на контрольной выборке от длины обучающей выборки. Длина обучающей выборки выводится в процентах от генеральной выборки.

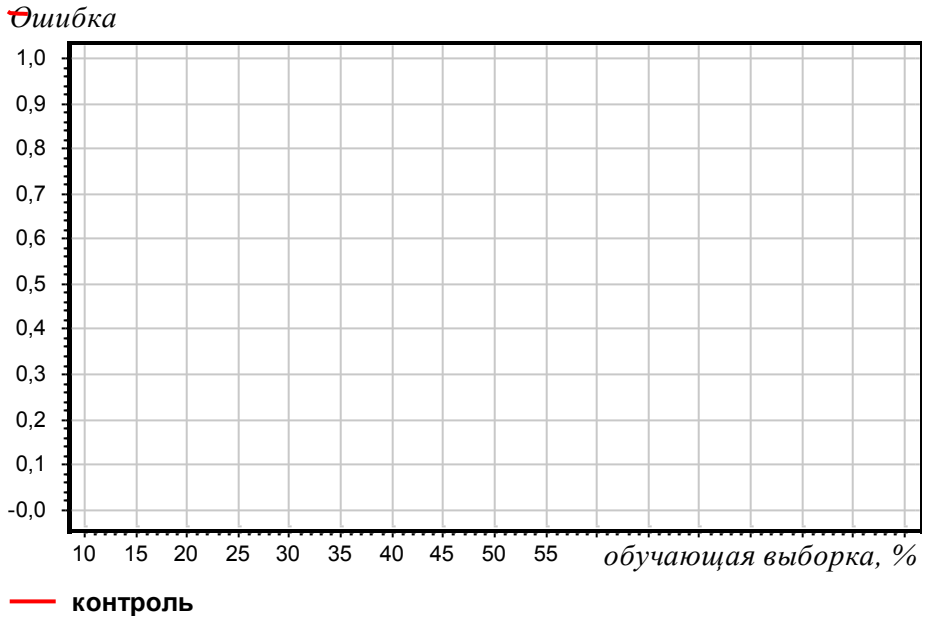


Рисунок 18. Кривая обучения

Описание графика:

- По оси абсцисс отложена длина обучающей выборки в процентах от длины генеральной выборки.
- По оси ординат отложена средняя ошибка алгоритмов на выборке данной длины.
- Красная линия – кривая обучения на контрольной выборке.
- Синяя линия – кривая обучения на обучающей выборке.

Анализ графика:

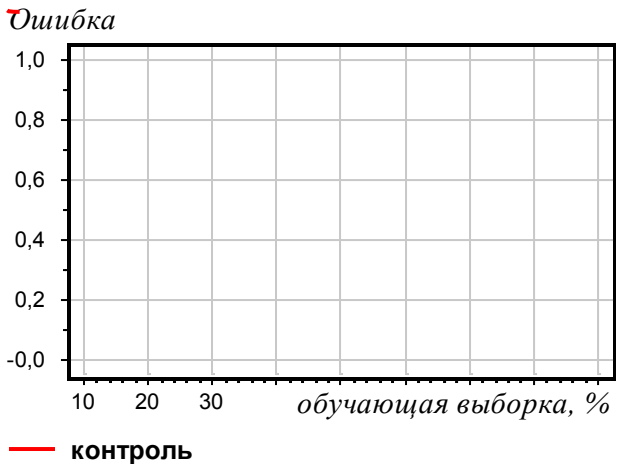
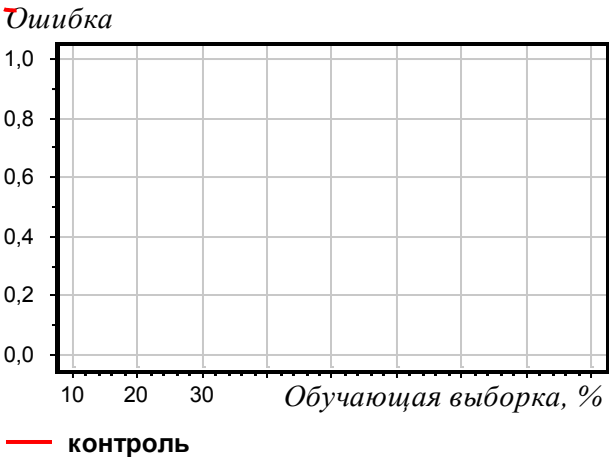
1. *Оценка минимальной длины обучающей выборки, необходимой для полного использования информации, содержащейся в генеральной выборке.*

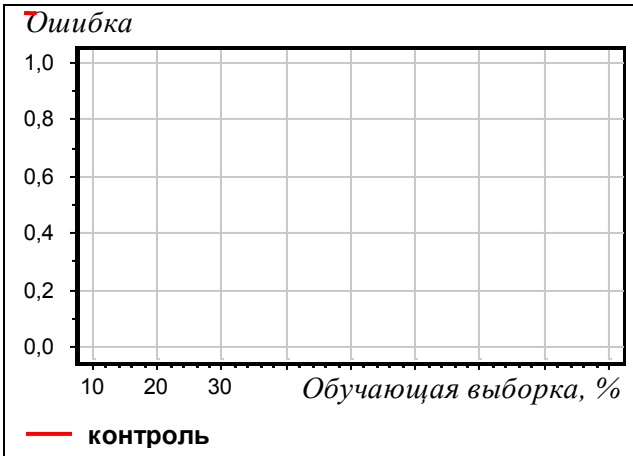
Чем меньше длина обучающей выборки, тем больше ошибок на контрольной выборке совершает алгоритм. Из графика можно определить длину обучающей выборке, когда её увеличение не будет способствовать улучшению качества классификации.

Эксперимент

С помощью системы Полигон были проведены следующие эксперименты на задачах из репозитория UCI [14] и алгоритмах из системы Weka [15].

Результаты:

	**Метод Обучения:** rules.JRip **Задача:** Iris **Выборка:** Класс 0 1. При длине обучающей выборки менее 10% средняя ошибка на контрольной выборке будет более 20%. При длине более 50% - ошибка менее 2%. Таким образом, 50% генеральной выборки - это минимальная длина обучения для достижения ошибки менее 2%.
	**Метод Обучения:** rules.JRip **Задача:** Iris **Выборка:** Все классы 1. При увеличении длины обучения средняя ошибка на контроле монотонно уменьшается до длины обучения 70%, после чего остаётся почти постоянной. Отсюда можно сделать вывод, что минимальная длина обучения – 70% от генеральной выборки.



Метод Обучения: lazy.KStar
Задача: Echocardiogram
Выборка: Все классы

1. При увеличении длины обучения средняя ошибка на контроле остаётся постоянной. Отсюда можно сделать вывод, что минимальная длина обучения – 10% от генеральной выборки.

Распределение стандартной ошибки

Распределение стандартной ошибки позволяет оценить качество классификации алгоритма, а также устойчивость алгоритма при варьировании обучающей выборки.

Пусть стандартная ошибка – случайная величина с неизвестным распределением. Построим её эмпирическую функцию распределения (Рисунок 6).

Полигон

Система Полигон для каждого из *t*q* алгоритмов, полученных при скользящем контроле, вычисляет эмпирическую функцию распределения ошибки. Далее происходит усреднение по всем алгоритмам и в результате получается функция распределения стандартной ошибки.

Вычисления проводятся отдельно для каждого класса, и отдельно на контроле и на обучении.

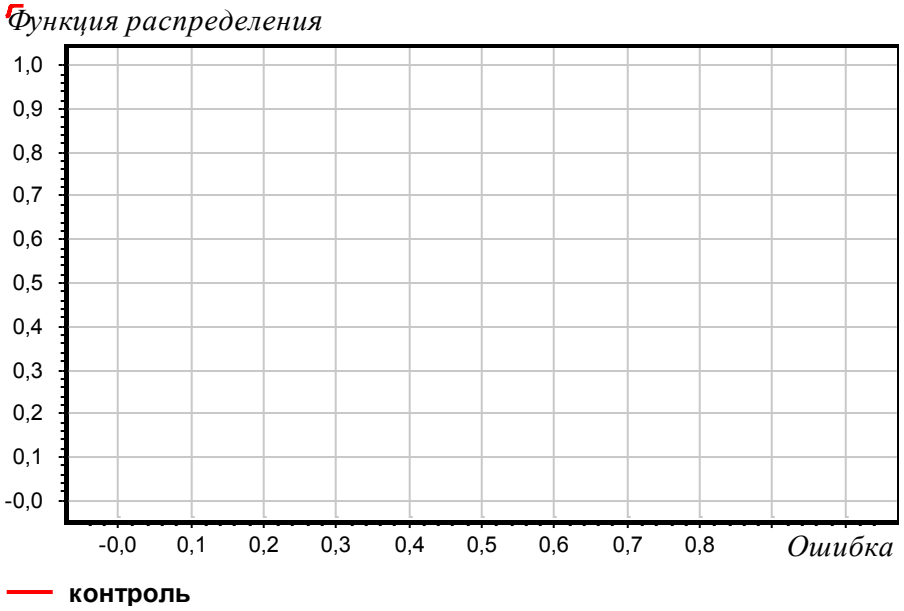


Рисунок 19. Функция распределения стандартной ошибки

Описание графика:

- По оси абсцисс отложена средняя ошибка алгоритмов на контрольной иди обучающей выборке.
- По оси ординат отложена эмпирическая функция распределения средней ошибки как случайной величины.
- Красная линия – функция распределения стандартной ошибки на контрольной выборке.
- Синяя линия – функция распределения стандартной ошибки на обучающей выборке.

Анализ графика:

1. *Оценка качества классификации метода обучения*

По графику можно оценить математическое ожидание ошибки на контрольной или обучающей выборке. Чем меньше средняя ошибка, тем лучше качество классификации.

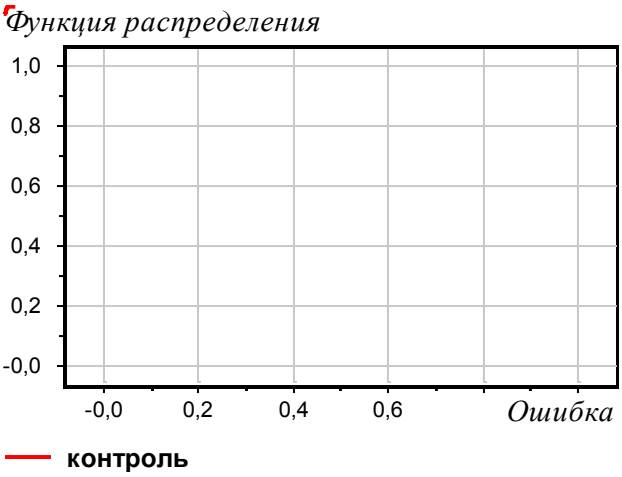
2. Оценка устойчивости метода обучения в зависимости от состава обучающей выборки

Быстрота роста функции распределения стандартной ошибки позволяет оценить устойчивость метода обучения. Чем быстрее растёт функция распределения, тем меньше средняя ошибка на выборке меняется при изменении обучающей выборки и, следовательно, устойчивее метод обучения.

Эксперимент

С помощью системы Полигон были проведены следующие эксперименты на задачах из репозитория UCI [14] и алгоритмах из системы Weka [15].

Результаты:

	**Метод Обучения:** rules.JRip **Задача:** Heart Disease **Выборка:** Все классы 1. Среднее качество классификации. Средние ошибки различных алгоритмов находятся в интервале от 0,1 до 0,3, с математическим ожиданием около 0,21 2. Плохая устойчивость метода обучения. Слишком большой разброс средних значений от 0,1 до 0,3
	**Метод Обучения:** rules.JRip **Задача:** Iris **Выборка:** Все классы 1. Плохое качество классификации. Средние ошибки различных алгоритмов находятся в интервале от 0,2 до 0,3, с математическим ожиданием около 0,26. 2. Хорошая устойчивость метода обучения. Малый разброс средних значений от 0,2 до 0,3

Метод Обучения: functions.Logistic
Задача: Australian
Выборка: Все классы

1. Хорошее качество классификации. Средние ошибки различных алгоритмов находятся в интервале от 0,07 до 0,16, с математическим ожиданием около 0,12.

2. Хорошая устойчивость метода обучения. Малый разброс средних значений от 0,07 до 0,16

Обоснования методики тестирования Полигона

В системе Полигон методика тестирования алгоритмов классификации основана на принципе t*q – кратного скользящего контроля со стратификацией классов. В связи с этим возникают следующие вопросы, связанные с обоснованием использования выбранной методики:

- Способ разбиения выборки на обучение и контроль
  - Случайным образом
  - Скользящий контроль
- Количество повторений t разбиений выборки на блоки при использовании скользящего контроля для построения
- Выбор оптимальных параметров t и q для скользящего контроля

Для того чтобы ответить на эти вопросы была проведена серия экспериментов, целью которых было обоснование принципа скользящего контроля, используемой в методике тестирования в системе Полигон.

Скользящий контроль является стандартным и общепринятым принципом для оценки методики тестирования, но он обладает несколькими недостатками:

- Задачу обучения приходится решать N раз, что сопряжено со значительными вычислительными затратами.
- Оценка скользящего контроля предполагает, что метод обучения уже задан. Она ничего не говорит о том, какими свойствами должны обладать «хорошие» алгоритмы обучения, и как их строить.
- Попытка использовать скользящий контроль для обучения в роли оптимизируемого критерия, приводит к тому, что он утрачивает свойство несмещённости, и снова возникает риск переобучения.
- Скользящий контроль дает несмещенную точечную оценку статистики, но при этом доверительный интервал получается зауженным. В настоящее время не существует методов построения доверительных интервалов статистик на основе скользящего контроля.

Эксперимент 1

Главным недостатком скользящего контроля с точки зрения системы Полигон является невозможность получения точных оценок для доверительного интервала. Это происходит из-за того, что при скользящем контроле обучение происходит на схожих выборках, что влечёт за собой схожие ответы алгоритмов и, следовательно, занижение ширины доверительного интервала.

Цель эксперимента: оценить, насколько зауженными получаются оценки доверительного интервала, полученные с помощью скользящего контроля**.**

При скользящем контроле выборка разбивается на q блоков, из которых каждый блок по очереди становится контрольной выборкой, а остальные q-1 блоков - обучающей выборкой. Таким образом, обучение происходит по схожим выборкам, которые содержат по $\frac{q-2}{q-1}L$ одинаковых объектов. В

таблице представлена зависимость перекрытия выборок (выраженная в процентах от длины выборки) от количества блоков, на которые делится выборка.

q	3	4	5	6	7	8	9	10	15	20
Перекрытие выборок	50,0%	66,7%	75,0%	80,0%	83,3%	85,7%	87,5%	88,9%	92,9%	94,7%

При настройке метода обучения на вход подаётся обучающая выборка, а на выходе получается алгоритм классификации. Когда на вход подаются выборки с большим перекрытием, может произойти лишняя перенастройка метода обучения: на выходе получатся похожие алгоритмы, которые будут давать схожие ответы на объектах контрольной выборки. Это может привести к занижению оценки доверительного интервала статистики.

Когда обучение происходит на независимых обучающих выборках без пересечения, то алгоритм будет давать менее схожие ответы и, следовательно, оценка доверительного интервала не будет заниженной.

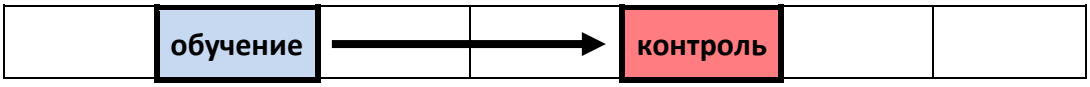
Методология эксперимента

Возьмём большую генеральную выборку длины N, поделим её на r блоков равной длины $n = \frac{N}{r}$ с одинаковым распределением классов в каждом блоке. Обучающая выборка из n объектов должна быть достаточна для хорошего качества классификации алгоритма.

1	2	3	…			r

Первый способ разбиения выборки

Рассмотрим первый способ разбиения выборки на обучение и контроль. Один блок по очереди берётся в качестве обучающей выборки и подаётся на вход методу обучения, на выходе получается алгоритм классификации.

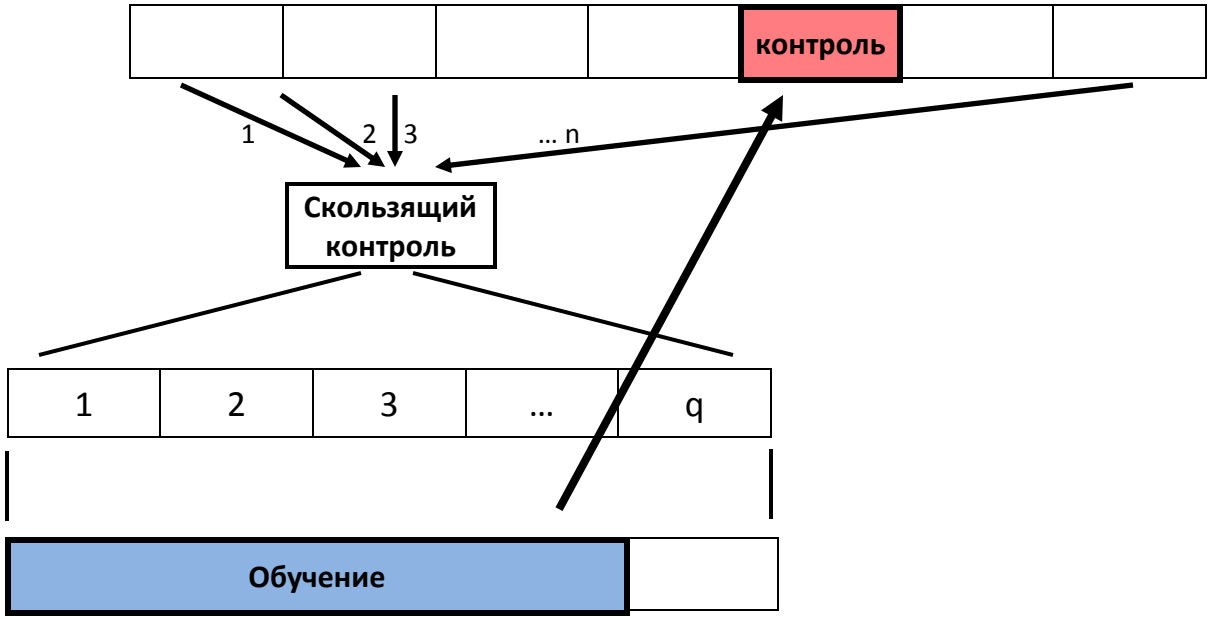


В качестве контрольной выборки берётся один из оставшихся блоков. После работы алгоритма классификации подсчитывается частота ошибок на контрольной выборке. Процедура повторяется для каждого из r блоков, а затем повторяется для каждого из t разбиений на блоки. После этого для частоты ошибок на контрольной выборке вычисляется математическое ожидание и строится доверительный интервал.

Второй способ разбиения выборки

Теперь рассмотрим второй способ разбиения выборки на обучение и контроль. Из генеральной выборки случайным образом выбирается n объектов. Эти объекты объединяются в подвыборку, которая будет подаваться на вход системе Полигон для настройки метода обучения. К этой выборке будет применена методика t*q-кратного скользящего контроля, используемого в системе Полигон.

Выборка делится на q блоков, метод обучения настраивается по очереди по q-1 блокам. В качестве контрольной выборки берётся не оставшийся блок из скользящего контроля, а контрольная выборка из первого способа разбиения генеральной выборки, то есть блок длины n.



После работы алгоритма классификации подсчитывается частота ошибок на контрольной выборке. Затем процедура обучения повторяется для каждого из q блоков и для t различных разбиений на блоки. Для всех алгоритмов подсчитываются частоты ошибок на контрольных выборках. После этого вычисляется математическое ожидание частоты ошибок на контрольной выборке и строится доверительный интервал.

Из-за того, что разбиения на q блоков происходит t раз, среднее значения перекрытия обучающих выборок будут отличаться от теоретических перекрытий, вычисленных по формуле $\frac{q-2}{q-1}$. Ниже приведён пример реальных перекрытий в зависимости от q.

q	4	5	8	10
Теоретическое перекрытие	66,7%	75,0%	85,7%	88,9%
Экспериментальное перекрытие	74,4%	79,6%	87,3%	88,9%

Доверительный интервал

Построение доверительного интервала для случайной величины ϑ – частоты ошибок на контроле.

Выборочное среднее:

$$\overline{\vartheta} = \frac{1}{n}\sum_{i=1}^{n}\vartheta_i$$

- несмещённая оценка математического ожидания случайной величины.

Обозначим $\hat{S}$ стандартное несмещённое выборочное квадратичное отклонение:

$$\widehat{S} = \sqrt{\widehat{S}^2} = \sqrt{\frac{1}{n-1}\sum_{i=1}^{n}(\vartheta_i - \overline{\vartheta})^2}\,.$$

Пусть α - уровень достоверности. Обозначим за Z_α – квантиль стандартного нормального распределения:

$$\alpha = \mathrm{P}(|\vartheta| > Z_\alpha) = 2\int_{Z_\alpha}^{\infty}\vartheta\, dx$$

Получаем доверительный интервал для частоты ошибок на контроле вида:

$$\left(\overline{\boldsymbol{\vartheta}} - \boldsymbol{Z_\alpha}\cdot\widehat{\boldsymbol{S}} < \boldsymbol{\vartheta} < \overline{\boldsymbol{\vartheta}} + \boldsymbol{Z_\alpha}\cdot\widehat{\boldsymbol{S}}\right)$$

где α - уровень достоверности.

Выбор задачи

Для проведения эксперимента задача должна была подходить по следующим требованиям. В ней должно было быть большое количество объектов, около 5000. Большее количество объектов в выборке приведёт к сильно завышенному времени работы системы. При этом алгоритм, который обучается на выборке длиной около 500 объектов, должен давать хорошее качество классификации.

N - длина тестовой выборки, n – общая длина контрольной и обучающей выборок (по этой части будет происходить обучение и настройка алгоритмов с помощью скользящего контроля).

$N \sim 5000$ объектов, $n \sim 500$ объектов, $N \gg n$.

Задача Nursery

Из репозитория UCI[14] была взята задача «*Nursery Hand Dataset*», Vl. Rajkovic. Задача имеет 12960 объектов, которые принадлежат одному из следующих 5 классов (стоит ли принимать ребёнка в школу).

Распределение объектов по классам:

Класс	Количество объектов	Количество объектов [%]
Не рекомендовать	4320	33,333%
Рекомендовать	2	0,015%
Сильно рекомендовать	328	2,531%
Приоритетные	4266	32,917%
Специальный приоритет	4044	31,204%

Для упрощения обучения было оставлено только три самых распространённых класса: не рекомендовать, приоритетные, специальный приоритет. Объекты, принадлежащие двум остальным классам, были удалены из выборки. Также выборка была сокращена до 4500 объектов.

Распределение объектов по классам:

Класс	Количество объектов	Количество объектов [%]
Не рекомендовать	1580	35,11%
Приоритетные	2162	48,04%
Специальный приоритет	758	16,85%
	4500	

Задача Chess

Из репозитория UCI[14] была взята задача «Chess End-Game: King+Rook versus King+Pawn», Vl. Rajkovic [12]. Задача имеет 3196 объектов, которые принадлежат одному из следующих 2 классов (кто победит в партии).

Распределение объектов по классам:

Класс	Количество объектов	Количество объектов [%]
Победа белых	1669	52%
Поражение белых	1527	48%
	3196	

Модельная задача #1

Кроме реальных задач было решено смоделировать задачу с необходимыми характеристиками: способную обучаться на маленьких выборках и имеющую большое количество объектов.

В смоделированной задаче 5000 объектов, 2 класса и 2 признака. Объекты равномерно разбросаны по двум пересекающимся прямоугольникам. Площадь пересечения прямоугольников – 4,878%.

Класс	Количество объектов	Количество объектов [%]
0	1950	39,0%
1	3050	61,0%
	5000	

Визуально задача представлена на рисунке:

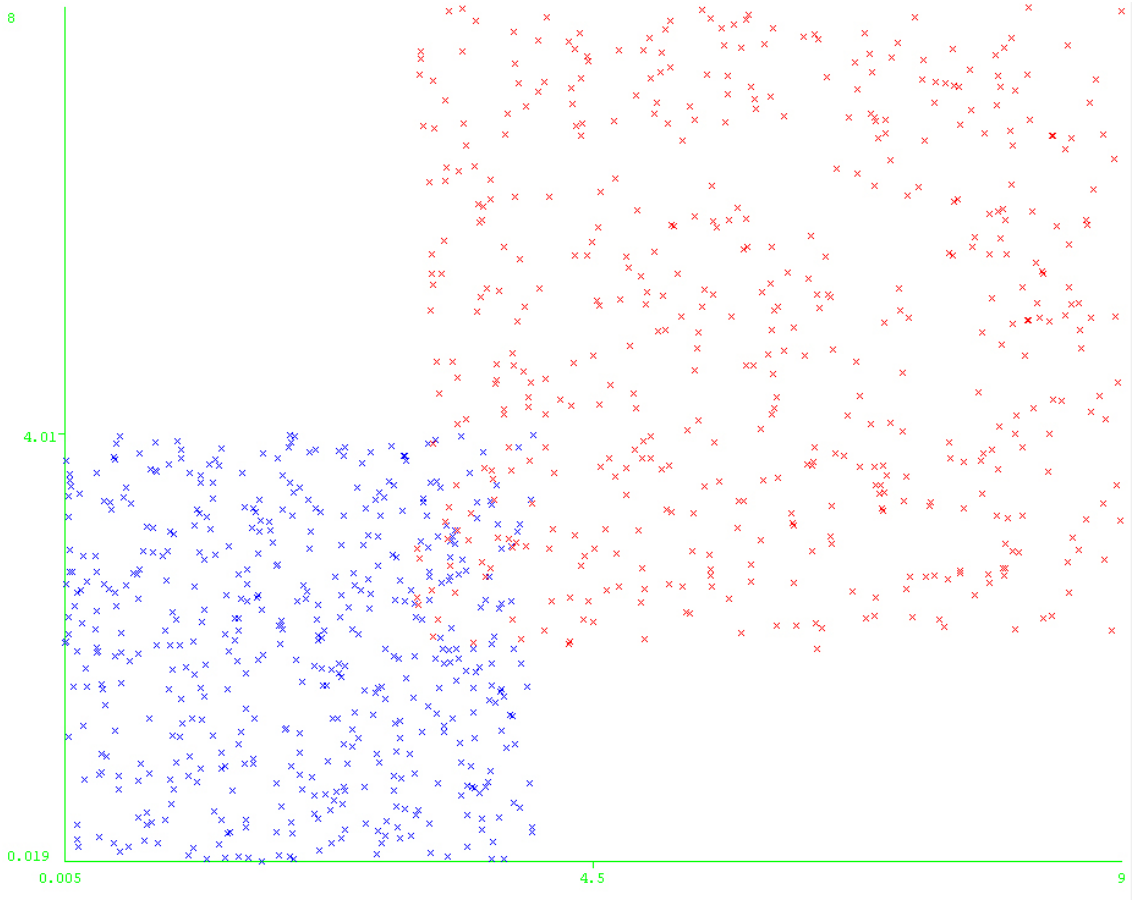


Рисунок 21. Смоделированная задача

Результаты эксперимента

С помощью системы Полигон были проведены эксперименты на алгоритмах functions.SMO(SVM), lazy.KStar(kNN), meta.Bagging из системы Weka[14] и задачах Nursery, Chess из репозитория UCI[15] и смоделированной задачи Model. Для этих задач были построены графики зависимости частоты ошибки и доверительного интервала от перекрытия обучающих выборок.

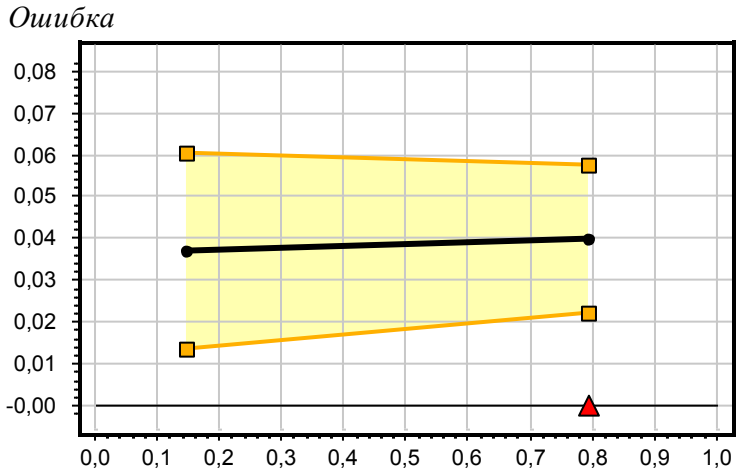
На графиках, представленных ниже:

- По оси абсцисс отложено перекрытие обучающих выборок в процентах от дины обучающей выборки
- По оси ординат отложены средняя частота ошибок и доверительные интервалы
- Чёрная линия – средняя частота ошибок на контроле
- Жёлтые линии – доверительные интервалы

На графиках слева отложена частота ошибок и доверительный интервал при первом способе разбиения выборки (минимальное перекрытие обучающих выборок), справа – при втором способе разбиения выборки (перекрытие, получающееся при использовании скользящего контроля).

Красным треугольником на рисунках обозначено перекрытие обучающих выборок, которое получается при использовании скользящего контроля с текущими параметрами системы Полигон t = 10 и q = 5.

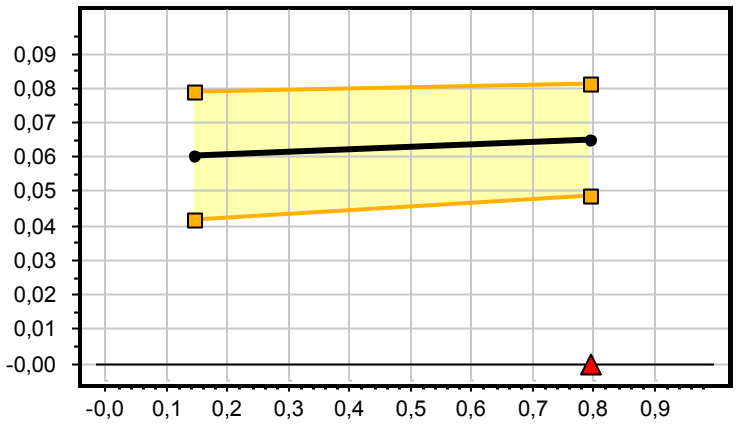
Серия экспериментов: t*q-кратный скользящий контроль с параметрами t = 10, q = 5.



t = 10, q = 5
Метод Обучения: functions.SMO
Задача: Nursery

Номер	Перекрытие	Ширина доверительного интервала
1	14,7%	0,0334
2	79,6%	0,0407

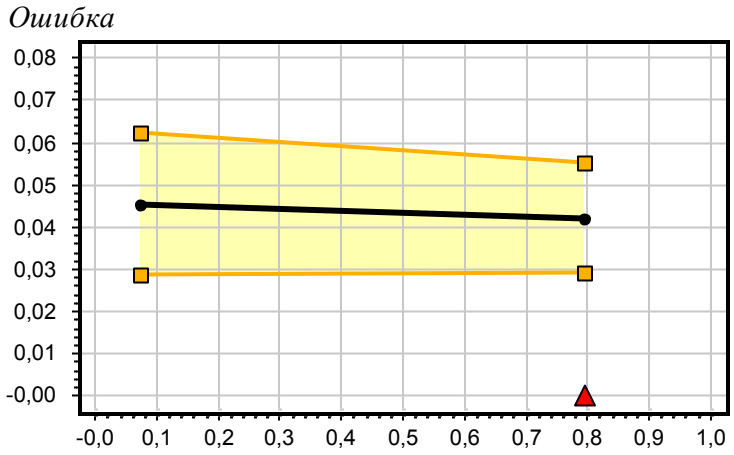
Занижение оценки доверительного интервала: 22,2%



t = 10, q = 5
Метод Обучения: functions.SMO
Задача: Chess

Номер	Перекрытие	Ширина доверительного интервала
1	7,3%	0,0371
2	79,6%	0,0324

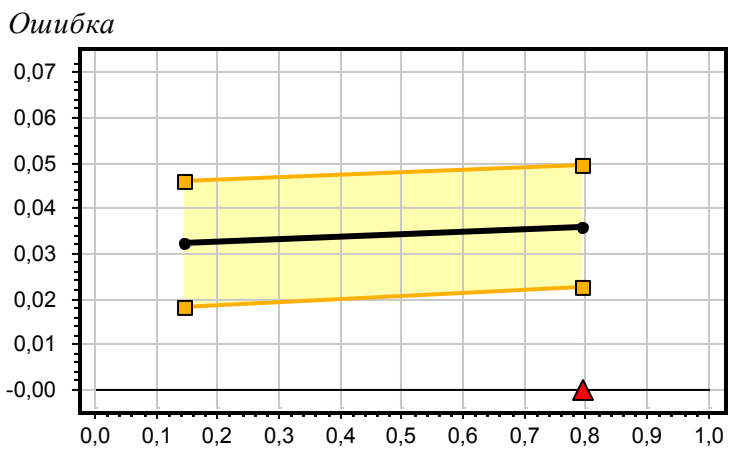
Занижение оценки доверительного интервала: 14,4%



t = 10, q = 5
Метод Обучения: functions.SMO
Задача: Model

Номер	Перекрытие	Ширина доверительного интервала
1	14,7%	0,0337
2	79,6%	0,0261

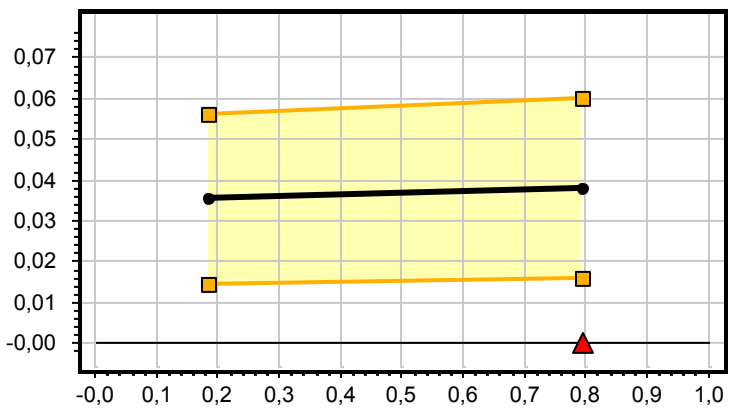
Занижение оценки доверительного интервала: 29,1%



t = 10, q = 5
Метод Обучения: meta.Bagging
Задача: Nursery

Номер	Перекрытие	Ширина доверительного интервала
1	14,7%	0,0279
2	79,6%	0,0271

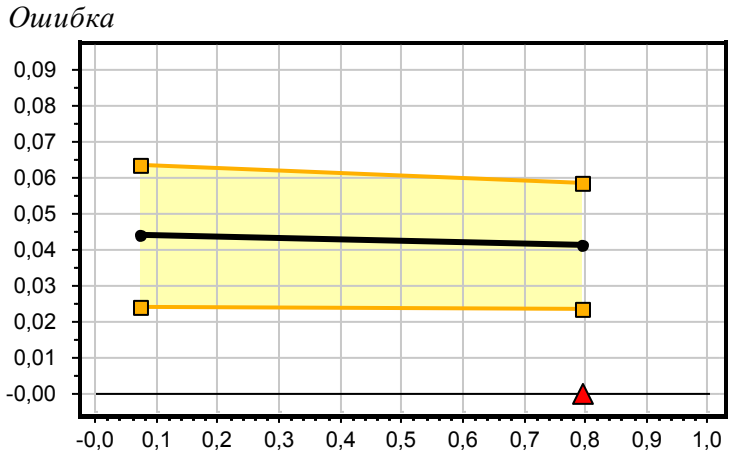
Занижение оценки доверительного интервала: 3,1%



t = 10, q = 5
Метод Обучения: meta.Bagging
Задача: chess

Номер	Перекрытие	Ширина доверительного интервала
1	7,3%	0,0416
2	79,6%	0,0442

Занижение оценки доверительного интервала: -6,0%

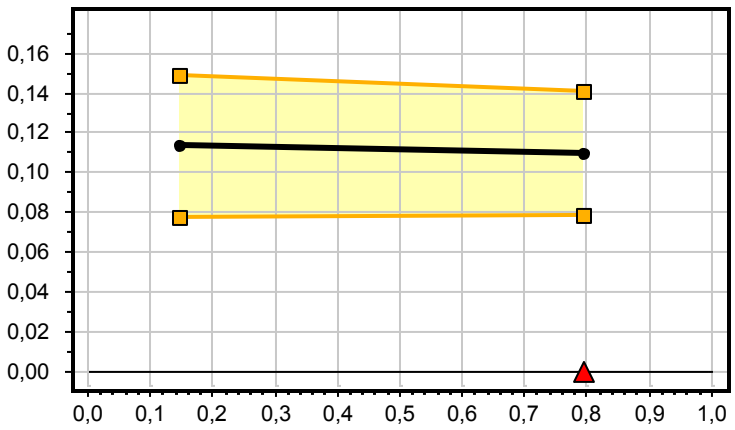


t = 10, q = 5
Метод Обучения: meta.Bagging
Задача: Model

Номер	Перекрытие	Ширина доверительного интервала
1	14,7%	0,0390
2	79,6%	0,0345

Занижение оценки доверительного интервала: 13,1%

Ошибка

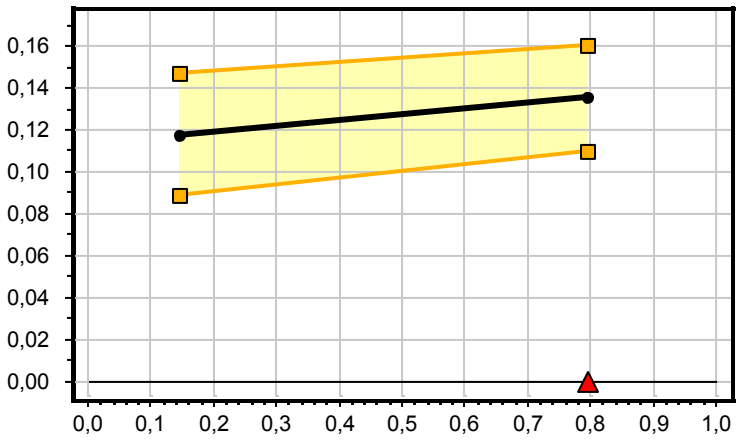


t = 10, q = 5
Метод Обучения: lazy.KStar
Задача: Nursery

Номер	Перекрытие	Ширина доверительного интервала
1	14,7%	0,0718
2	79,6%	0,0628

Занижение оценки доверительного интервала: 14,4%

Ошибка

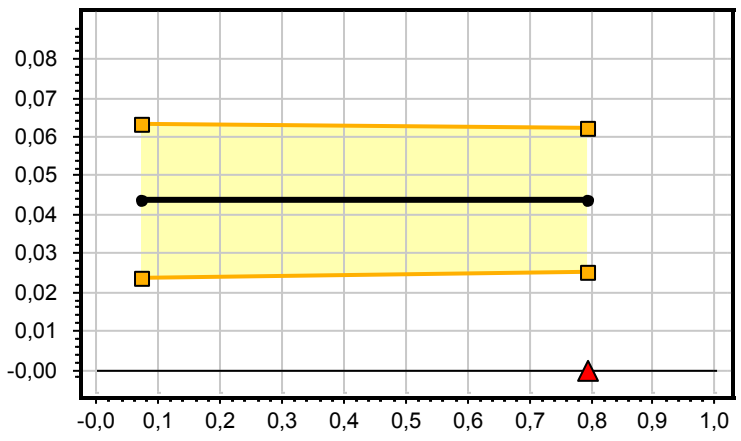


t = 10, q = 5
Метод Обучения: lazy.KStar
Задача: Chess

Номер	Перекрытие	Ширина доверительного интервала
1	7,3%	0,0587
2	79,6%	0,0504

Занижение оценки доверительного интервала: 16,4%

Ошибка



t = 10, q = 5
Метод Обучения: lazy.KStar
Задача: Model

Номер	Перекрытие	Ширина доверительного интервала
1	14,7%	0,0399
2	79,6%	0,0372

Занижение оценки доверительного интервала: 7,1%

Аналогичные эксперименты были проведены при значениях параметрах q: 4, 6, 7, 8, 10.

В таблицах приведены величина занижения доверительного интервала (в процентах) для каждой пары «алгоритм – задача» для параметра q равного 4, 5, 6, 7, 8, 10.

		Алгоритмы			
	q=4	functions.SMO	lazy.KStar	meta.Bagging	Среднее по задаче
Задачи	chess	16,4%	4,6%	-9,5%	3,8%
	nursery	0,3%	-3,6%	5,9%	0,9%
	model	11,1%	14,8%	22,9%	16,3%
Среднее по алгоритму		9,3%	5,3%	6,4%	7,0%

		Алгоритмы			
	q=5	functions.SMO	lazy.KStar	meta.Bagging	Среднее по задаче
Задачи	chess	14,4%	16,4%	-6,0%	8,3%
	nursery	33,0%	14,4%	3,1%	16,8%
	model	29,1%	7,1%	13,1%	16,4%
Среднее по алгоритму		25,5%	12,6%	3,4%	13,8%

		Алгоритмы			
	q=6	functions.SMO	lazy.KStar	meta.Bagging	Среднее по задаче
Задачи	chess	27,7%	-4,7%	24,3%	15,8%
	nursery	35,4%	15,1%	8,3%	19,6%
	model	9,1%	13,2%	11,1%	11,2%
Среднее по алгоритму		24,1%	7,9%	14,6%	15,5%

		Алгоритмы			
	q=7	functions.SMO	lazy.KStar	meta.Bagging	Среднее по задаче
Задачи	chess	28,3%	10,5%	-25,3%	4,5%
	nursery	3,8%	53,6%	27,4%	28,3%
	model	-6,8%	-13,5%	23,9%	1,2%
Среднее по алгоритму		8,4%	16,9%	8,6%	11,3%

		Алгоритмы			
	q=8	functions.SMO	lazy.KStar	meta.Bagging	Среднее по задаче
Задачи	chess	16,3%	13,5%	52,0%	27,3%
	nursery	-0,9%	19,2%	6,6%	8,3%
	model	11,6%	10,9%	18,9%	13,8%
Среднее по алгоритму		9,0%	14,5%	25,8%	16,5%

		Алгоритмы			
	q=10	functions.SMO	lazy.KStar	meta.Bagging	Среднее по задаче
Задачи	chess	41,1%	29,5%	36,0%	35,5%
	nursery	22,2%	31,6%	6,3%	20,0%
	model	1,2%	26,5%	39,3%	22,4%
Среднее по алгоритму		21,5%	29,2%	27,2%	26,0%

Зависимость среднего по всем алгоритмам и задачам занижения оценки доверительного интервала от параметра q (количества блоков в скользящем контроле) отображена на следующем графике.

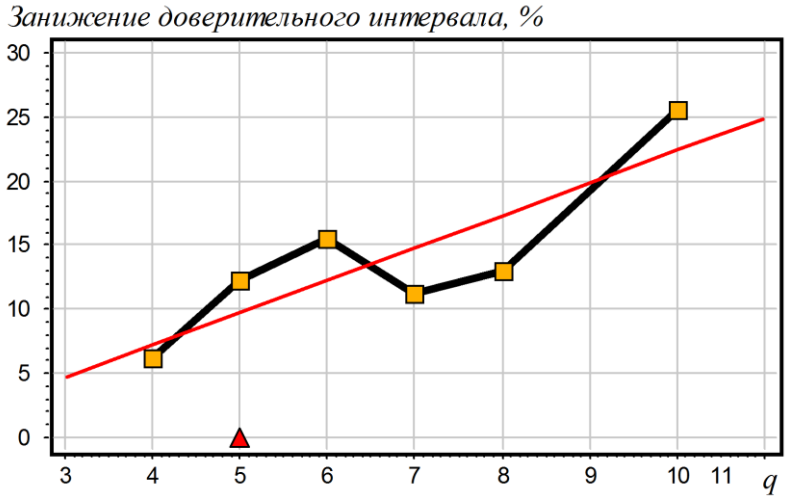


Рисунок 22. Зависимость занижения доверительного интервала от количества блоков в скользящем контроле

На графике видно, что при увеличении количества блоков, на которые разбивается генеральная выборка при скользящем контроле, увеличивается занижение оценки доверительного интервала. Таким образом, если перекрытие обучающих выборок увеличивается, то увеличивается вероятность того, что оценка доверительного интервала, полученная с помощью скользящего контроля, будет сильно заниженной, то есть не будет достоверной.

Также полученные данные позволяют сделать вывод о том, что достоверность оценки доверительного интервала зависит от алгоритма классификации. У алгоритма function.SMO среднее занижение оценки доверительно интервала равно 16,3%, а у алгоритмов lazy.KStar и meta.Bagging 14,4%.

Эксперимент 2

Цель эксперимента: определение оптимального параметра t в t*q-кратном скользящем контроле.

Методология эксперимента

Была проведена серия экспериментов, в которых алгоритм классификации и задача подавались на вход системе Полигон, которая использует t*q-кратный скользящий контроль. На выходе подсчитывалась частота ошибок на контроле для каждого разбиения генеральной выборки на обучение и контроль. После этого для частоты ошибок на контрольной выборке была построена эмпирическая функция распределения.

Пусть p – вероятность совершения ошибки алгоритмом на объекте. При фиксированной обучающей выборке количество ошибок на контрольной выборке подчиняется биномиальному распределению.

Если обучающая выборка изменяется, частота ошибок на контрольной выборке подчиняется уже не биномиальному распределению, а смеси биномиальных распределений. Но общий вид распределения должен быть схожим с биномиальным распределением.

Биномиальное распределение

Биномиальное распределение – распределение количества успехов в последовательности из n независимых испытаний Бернулли с вероятностью успеха в каждом равном p.

Пусть $X_1, X_2, \dots, X_n$ - конечная последовательность независимых случайных величин с распределением Бернулли.

$$X_i = \begin{cases} 1, & p \\ 0, & 1-p \end{cases}, i = 1, \dots, n$$

Построим случайную величину Y:

$$Y = \sum_{i=1}^{n} X_i$$

Тогда Y - число успехов в последовательности $X_1, X_2, \dots, X_n$, имеет биномиальное распределение с n степенями свободы и вероятностью успеха p.

Функция плотности вероятности биномиального распределения $Bin(p, k, n)$:

$$\mathbb{P}(Y = k) = \binom{n}{k} p^k (1-p)^{n-k} \quad , k = 0, \dots, n,$$

где $\binom{n}{k} = \frac{n!}{(n-k)!k!}$ - биномиальный коэффициент.

Результаты эксперимента

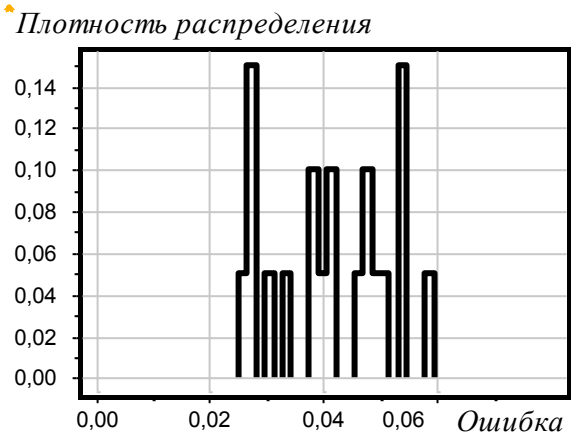
С помощью системы Полигон были проведены эксперименты на алгоритмах functions.SMO(SVM), lazy.KStar(kNN), meta.Bagging из системы Weka[14] и задачах Nursery, Chess из репозитория UCI[15] и смоделированной задачи Model.

Для алгоритмов и задач были построены графики эмпирической плотности распределения частоты ошибок на контроле при различных значениях параметра t (количество повторений разбиений генеральной выборки на блоки): 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 15, 20.

На графиках, приведённых ниже:

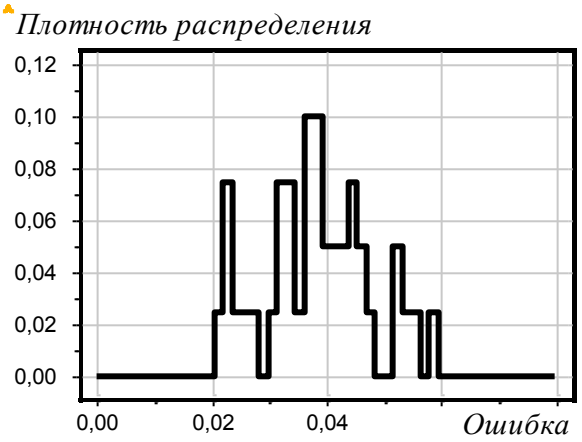
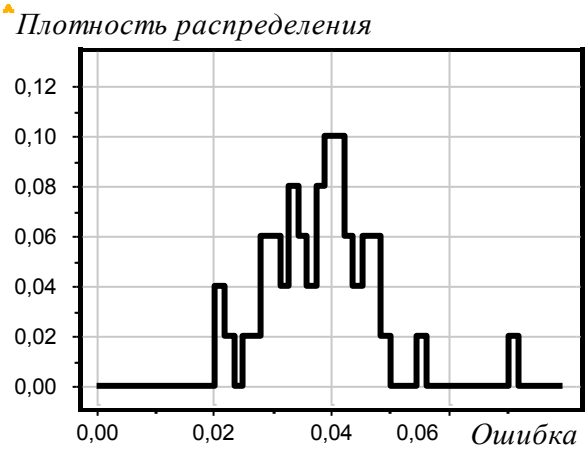
- По оси абсцисс отложена частота ошибок на контрольной выборке
- По оси ординат отложена плотность распределения
- Чёрная линия – эмпирическая функция плотности распределения
- Жёлтые линии – биномиальная функция плотности распределения
- Жёлтые вертикальные линии – границы доверительного интервала

На графиках отложены эмпирическая функция плотности распределения и биномиальная функция плотности распределения частоты ошибок на контрольной выборке для алгоритма meta.Bagging, задачи Chess, q = 5 и различных значениях параметра t: 4, 6, 8, 10.



Метод Обучения: meta.Bagging
Задача: Chess
Скользящий контроль: CV_5_t

Параметр t:
t = 4

	**Метод Обучения:** meta.Bagging **Задача:** Chess **Скользящий контроль: CV_5_t** **Параметр t:** **t = 6**
	**Метод Обучения:** meta.Bagging **Задача:** Chess **Скользящий контроль: CV_5_t** **Параметр t:** **t = 8**
	**Метод Обучения:** meta.Bagging **Задача:** Chess **Скользящий контроль: CV_5_t** **Параметр t:** **t = 10**

На графиках видно, что при увеличении параметра t (количества повторений разбиения генеральной выборки) эмпирическая функция плотности распределения становится менее похожей на равномерную функцию распределения и более похожей на биномиальную функцию распределения.

Практические рекомендации для системы Полигон

Рекомендация 1

Увеличение перекрытия обучающих выборок приводит к занижению оценки доверительного интервала для статистики, и эта оценка становится менее достоверной. Следовательно, чем больше будет количество блоков q в скользящем контроле, тем менее достоверной будет оценка доверительного интервала. Поэтому в качестве практических рекомендаций для системы Полигон параметр q следует брать равным 4 или 5, в случае, если задача имеет выборку большой длины. Иначе следует выбирать параметр q равный 7 или 8, для того, чтобы полностью использовать информацию, содержащуюся в генеральной выборке задачи.

Рекомендация 2

Увеличение параметра t приводит к более надёжным данным для построения доверительного интервала, однако оно также приводит к увеличению вычислительной сложности эксперимента. Поэтому оптимальными параметрами для системы Полигон являются t в интервале от 8 до 10.

Результаты

Разработана методика тестирования алгоритмов классификации.
Данная методика удовлетворяет требованиям стандартизации и воспроизводимости результатов, а, следовательно, позволяет сравнивать качество алгоритмов классификации на задачах и подтверждать полученные результаты.

Методика тестирования была внедрена и протестирована в рамках системы «Полигон».
Данная методика тестирования алгоритмов классификации была реализована в рамках научного проекта «Полигон», который является совместной разработкой компании ЗАО «Форексис» и ВЦ РАН им. Дородницына. В рамках проекта было проведено множество практических тестирований алгоритмов классификации на реальных задачах. Эти эксперименты позволяют сделать вывод о важности использования единой методики при оценивании качества алгоритмов классификации.

Проведено обоснования использования методики для системы Полигон.
В системе «Полигон» проведены эксперименты на реальных алгоритмах и реальных задачах, которые позволили дать практические рекомендации по использованию методики тестирования.

Список литературы

[1] Воронцов К.В., Инякин А.С., Лисица А.В. (2007). Система эмпирического измерения качества алгоритмов классификации. ММРО-13 (pp. 577-581).

[2] Лисица А.И., Воронцов К.В., Ивахненко А.А., Инякин А.С., Синцова В.В. (2010). Системы тестирования алгоритмов машинного обучения MLcomp, TunedIt и Полигон. ИОИ-8.

[3] Паклин Н. (2006). Логистическая регрессия и ROC-анализ – математический аппарат. http://www.basegroup.ru/library/analysis/regression/logistic/.

[4] Asuncion A., Newman D.J. (2007). UCI Machine Learning Repository – University of California, Irvine. – www.ics.uci.edu/~mlearn/MLRepository.html.

[5] Domingos P. (2000). A Unified Bias-Variance Decomposition and its Applications. 17th ICML (pp. 231-238). Department of Computer Science and Engineering, University of Washington, Seattle, WA.

[6] Fawcett T. (2005). An introduction to ROC analysis. Institute for the Study of Learning and Expertise. Institute for the Study of Learning and Expertise, Palo Alto, USA.

[7] Garg A., Roth D. (2003). Margin Distribution and Learning Algorithms. ICML'03 (pp. 210-217). Washington, DC USA.

[8] Hand D., Till R. (2001). A simple generalization of area under the ROC curve for multiple class classification problems. Machine Learning, 45, 171-186.

[9] Kohavi R., Wolpert D. H. (1996). Bias plus variance decomposition for zero-one loss functions. Proceedings of the Thirteenth International Conference on Machine Learning (pp. 275-283). Bari, Italy: Morgan Kaufmann.

[10] Kong E.B., Dietterich T.G. (1995). Error-Correcting Output Coding Corrects Bias and Variance. Proceedings of the Thirteenth International Conference on Machine Learning (pp. 313-321). Tahoe city, CA: Morgan Kaufmann.

[11] Perlich Cl., Provost F., Simonoff J. S. (2003). Tree Induction vs. Logistic Regression: A Learning-Curve Analysis. Leonard N. Stern School of Business, New York University, NY, NY10012.

[12] Provost F., Domingos P. (2001). Well-trained PETs: Improving probability estimation trees, CeDER Working Paper #IS-00-04, Stern School of Business, New York University, NY, NY 10012.

[13] Tsuyuguchi M., Uehara K. (2002). Bias-Variance Decomposition of Zero-One Loss in Average-Case Model. Kobe University, Nada, Japan.

[14] The UCI Machine Learning Repository (Репозиторий задач машинного обучения), http://archive.ics.uci.edu/ml/

[15] WEKA - Data Mining Software in Java (Weka – программа для машинного обучения), http://www.cs.waikato.ac.nz/ml/weka/

[16] Воронцов К. В., Инякин А. С., Лисица А. В., Романов М. Ю., Стрижов В. В., Хачай М. Ю., Чехович Ю. В. Распределенная вычислительная система «полигон алгоритмов классификации» // Интеллектуализация обработки информации (ИОИ-2008): Тезисы докл. — Симферополь: КНЦ НАН Украины, 2008. С. 54–56.

[17] Воронцов К. В., Ивахненко А. А., Инякин А. С., Лисица А. В., Минаев П. Ю. «Полигон» — распределённая система для эмпирического анализа задач и алгоритмов классификации // Математические методы распознавания образов: 14-ая Всеросс. конф.: Докл. М.: МАКС Пресс, 2009. С. 503–506.