УНИВЕРСИТЕТ ИТМО

И.Р. Баймуратов

МЕТОДЫ АВТОМАТИЗАЦИИ МАШИННОГО ОБУЧЕНИЯ

Санкт-Петербург
2020

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

УНИВЕРСИТЕТ ИТМО

И.Р. Баймуратов

МЕТОДЫ АВТОМАТИЗАЦИИ МАШИННОГО ОБУЧЕНИЯ

УЧЕБНОЕ ПОСОБИЕ

РЕКОМЕНДОВАНО К ИСПОЛЬЗОВАНИЮ В УНИВЕРСИТЕТЕ ИТМО
по направлению подготовки 09.04.04 Программная инженерия
в качестве учебного пособия для реализации основных профессиональных образовательных программ высшего образования магистратуры,

Санкт-Петербург
2020

Баймуратов И.Р., Методы автоматизации машинного обучения– СПб: Университет ИТМО, 2020. – 40 с.

Рецензент(ы):
Муромцев Дмитрий Ильич, кандидат технических наук, доцент, доцент (квалификационная категория "ординарный доцент") факультета программной инженерии и компьютерной техники, Университета ИТМО.

В учебном пособии изложены методы автоматизации машинного обучения: методы конструирования признаков, выбора моделей обучения и оптимизации гиперпараметров алгоритмов, оценки результатов обучения, а также актуальные проблемы.

Университет ИТМО – ведущий вуз России в области информационных и фотонных технологий, один из немногих российских вузов, получивших в 2009 году статус национального исследовательского университета. С 2013 года Университет ИТМО – участник программы повышения конкурентоспособности российских университетов среди ведущих мировых научно-образовательных центров, известной как проект «5 в 100». Цель Университета ИТМО – становление исследовательского университета мирового уровня, предпринимательского по типу, ориентированного на интернационализацию всех направлений деятельности.

Оглавление

Введение

Учебное пособие предназначено для студентов, изучающих курс «Методы машинного обучения» по направлению подготовки 09.04.04 «Программная инженерия» и другие курсы, связанные с машинным обучением. Для усвоения учебного материала при самостоятельной работе студентов в пособии изложены систематичные и теоретически обоснованные знания, связанные с общими задачами применения методов машинного обучения. Также пособие содержит указания к выполнению лабораторных работ, позволяющих получить практические навыки автоматизации машинного обучения. В качестве программного обеспечения для выполнения лабораторных работ предлагается использовать библиотеку «Scikit-learn» языка «Python», но указанные задания могут быть выполнены и в других программных комплексах.

Предполагается, что алгоритмы машинного обучения должны работать «из коробки», но современное состояние технологии далеко до достижения этой цели. При проектировании процесса машинного обучения пользователю приходится осуществлять предварительную обработку данных, выбирать конкретный алгоритм и настраивать его гиперпараметры. Решения, принимаемые на этих этапах, существенно влияют на качество результатов применения того либо иного метода. Проектирование процесса машинного обучения в большинстве случаев осуществляется вручную, на основе опыта либо интуиции пользователя с применением эвристических, недетерминированных методов. Ручное проектирование имеет ряд очевидных недостатков. Во-первых, непрерывно появляются новые методы и алгоритмы машинного обучения, а существующие развиваются и усложняются, в результате чего для применения актуальных технологий машинного обучения пользователю требуется значительная теоретическая

и практическая подготовка. Во-вторых, ручной поиск оптимального решения требует больших человеческих трудозатрат. С одной стороны, производится большое количество итераций обучения и оценки результатов, что занимает значительное время, с другой, требуются интеллектуальные усилия для разработки эвристических методов решения конкретных задач. Наконец, ручное проектирование может привести к субоптимальным результатам. Это может быть обусловлено либо ограниченностью человеческих вычислительных способностей, либо неэффективностью примененного эвристического метода. Таким образом, автоматизация машинного обучения является актуальным направлением исследований. Оно включает в себя автоматизацию следующих задач [34]:

- предварительная обработка данных;
- конструирование признаков;
- выбор модели обучения;
- оптимизация гиперпараметров;
- конструирование конвейеров;
- анализ полученных результатов и др.

Существующие методы автоматизации уже показали большую эффективность в сравнении с работой экспертов. На сегодняшний день существуют следующие системы автоматизированного машинного обучения:

- TransmogrifAI [61];
- H2O-AutoML [30];
- Darwin [18];

- DataRobot [19];
- GoogleAutoML [29];
- Auto-sklearn [26];
- MLjar [42];
- Auto_ML [5];
- TPOT [46];
- Auto-keras [35];
- Ludwig [41];
- Auto-WEKA [38];
- Azure ML [4];
- H2O-DriverlessAI [31];
- ATM [57];
- RECIPE [22];
- Auto-MEKA [23];
- ML-Plan [43] и др.

В первом разделе данного пособия представлены формальные модели машинного обучения, которые представляют собой теоретическое основания для построения методов автоматизации. Теоретический раздел содержит список вопросов для самоконтроля. Дальнейшие разделы соответствуют основным задачам автоматизации машинного обучения: второй раздел посвящен автоматизации конструирования признаков, третий – выбору модели и оптимизации гиперпараметров, четвертый – оценке результатов машинного обучения. В пятом разделе представлены некоторые

проблемы автоматизации машинного обучения, актуальные на сегодняшний день. Для получения практических навыков пособие содержит две лабораторные работы: первая направлена на получение навыка автоматизации конструирования признаков, вторая – выбора модели и оптимизации гиперпараметров.

1. Формальные модели машинного обучения

Теории машинного обучения формулируют ответы на такие вопросы, как что такое обучение, какова цель обучения и какими средствами она достигается [44]. Ответы на эти вопросы предполагают некоторую формальную модель обучения. Рассмотрим две существующие теории машинного обучения: теорию вероятностно приблизительно корректного обучения и теорию оккамовского обучения [36].

1.1. Вероятностно приблизительно корректное обучение

На сегодняшний день наиболее распространенной является теория вероятностно приблизительно корректного обучения [63] (ВПК). Соответствующая ей модель обучения включает в себя:

- множество входных данных X;
- множество выходных значений Y;
- множество гипотез $H: X \to Y$;
- выборку $S \subset X \times Y$;
- функцию потерь $L(h)$.

Согласно парадигме ВПК обучения, алгоритм обучения A на основе выборки S осуществляет выбор гипотезы $h \in H$, т.е. $A: S \to H$. Выбор осуществляется посредством минимизации функции потерь $L(h, S)$:

$$h_S = argmin_{h \in H} L(h, S).$$

Центральной проблемой ВПК обучения является сложность обучающей выборки – определение размера m выборки S, необходимого для выбора гипотезы h_S из множества гипотез H, которая с заданной вероятностью δ

имеет значение функции потерь $L(h, S)$, не превышающее заданное значение ϵ. Нижняя граница размера выборки m задается следующей формулой:

$$m \geq \frac{\log(|H|/\delta)}{\epsilon}.$$

Если m – конечное число, говорят, что алгоритм *обучаемый*, если m растет полиномиально, то алгоритм ВПК-обучаемый.

1.2. Оккамовское обучение

В оккамовском обучении [9] не рассматриваются вопросы, связанные с репрезентативностью выборки объектов, и оценка гипотез осуществляется не на основе точности предсказаний, а на основе того, насколько компактно она описывает данные. Идея оккамовского обучения заключается в том, что алгоритм, который во всех случаях находит компактную гипотезу, также имеет предсказательную силу.

Модель оккамовского обучения включает в себя:

- множество входных данных X;
- множество выходных значений Y;
- множество гипотез $H: X \to Y$;
- выборку $S = X \times Y$.

Как и в ВПК обучении, задачей оккамовского алгоритма обучения A является выбор гипотезы $h \in H$ на основе выборки S. Выбор гипотезы h, представленной как бинарная последовательность, осуществляется на основе ее размера $size(h)$. Пусть $size(y)$ обозначает размер кратчайшего представления $y \in Y$, $|X| = m$ и $|Y| = n$, тогда должно выполняться условие:

$$size(h) \leq \left(n\, size(y)\right)^{\alpha} m^{\beta}.$$

Для некоторых заданных $\alpha \geq 0$ и $0 \leq \beta < 1$.

Контрольные вопросы

1. Перечислите задачи автоматизации машинного обучения.
2. Перечислите основные теории машинного обучения.
3. Определите формальные модель ВПК обучения.
4. Определите формальную модель оккамовского обучения.

2. Конструирование признаков

Задачу конструирования признаков можно рассматривать как поиск оптимального представления набора данных. Конструирование признаков направлено на достижение различных целей:

- сокращение вычислительных затрат при последующем обучении;
- повышение качества результатов последующего обучения;
- визуализация данных;
- извлечение новых признаков и др.

Рассмотрим основные методы определения оптимальной структуры признакового пространства.

2.1. Правило Кайзера

Правило Кайзера [65] – один из самых простых и интуитивно понятных критериев отбора компонент. Значимы те компоненты, которые превосходят среднее значение собственных значений $\bar{\lambda}$. Правило Кайзера упрощается, если данные были изначально нормированы на единичную выборочную дисперсию по осям, в этом случае значимы компоненты, для которых $\lambda_i > 1$. Правило Кайзера эффективно только в простых случаях, когда несколько собственных значений λ_i значительно превосходят среднее значение $\bar{\lambda}$, в то время как остальные значительно меньше него, и неэффективно в ситуации, когда собственные значения компонент близки к $\bar{\lambda}$. Например, согласно правилу Кайзера, если собственное число $\lambda_i = 1{,}01$, то компонента информативна, а если $\lambda_i = 0{,}99$, то нет.

2.2. Правило сломанной трости

Одним из эвристических подходов к оценке оптимального числа компонент m является правило сломанной трости [13]. Собственные значения компонент нормируются на единичную сумму и сравниваются с распределением длин обломков трости единичной длины, которую сломали в $m-1$ случайно выбранных точках. Пусть L_i – длины полученных частей трости в порядке убывания, и λ_i – нормированные собственные значения компонент, тогда нормированное значение λ_i' определяется формулой:

$$\lambda_i' = \frac{\lambda_i}{\sum_i \lambda_i}.$$

Определим математическое ожидание L_i как l_i:

$$l_i = \frac{1}{n}\sum_i \frac{1}{L_i}.$$

По правилу сломанной трости следует оставлять только те компоненты, для которых выполняется условие $\lambda_\text{i}' > l_i$.

2.3. Объясненная дисперсия

При использовании метода главных компонент [48] (МГК) используется оценка на основе объясненной дисперсии. Задача метода главных компонент состоит в проецировании набора данных на новый ортонормированный базис. Векторы нового базиса называются главными компонентами, и собственное значение λ_i описывает дисперсию исходных данных вдоль i-ой главной компоненты. Пусть S – объясненная дисперсия выбранного числа компонент d из множества всех компонент m, опишем полученную формулу:

$$S = \frac{\sum_{i=i}^{d} \lambda_i}{\sum_{i=i}^{m} \lambda_i}.$$

При выборе всех компонент, т.е. при $d = m$, получим набор данных, равный исходному. Пороговое значение для объясненной дисперсии выбирается

исследователем после анализа исходных данных и собственных векторов и значений, полученных в результате применения МГК. При низком значении порога представление данных будет неполным, с другой стороны, избыточное число главных компонент приведет к ситуации, когда моделируется шум вместо содержательной информации. Стандартными значениями порога являются 80% и 95%.

2.4. «Метод локтя»

При определении оптимального набора компонент применим так называемый «Метод локтя» [59]. Суть этого метода применительно к задаче конструирования признаков заключается в рассмотрении объясненной дисперсии как целевой функции, зависящей от числа компонент. Оптимальное число компонент при этом определяется как число, при достижении которого прирост объясненной дисперсии становится незначительным. Если изобразить эту зависимость на графике, то в соответствующей оптимальному числу будет изгиб, напоминающий локоть, отсюда и название метода. Недостаток метода заключается в том, что «локоть» далеко не всегда явно выражен и может быть однозначно идентифицирован.

3. Выбор модели обучения и оптимизация гиперпараметров алгоритмов

В последние годы проблеме выбора модели и оптимизации гиперпараметров посвящается множество исследовательских работ. Существуют различные ее формулировки: проблема оптимизации гиперпараметров (hyperparameter optimization, HPO) [37], проблема полного выбора модели (Full Model Selection, FMS) [25] или проблема совместного выбора алгоритма и гиперпараметров (Combined Algorithm Selectionand Hyperparameter, CASH) [60] [26].

Рассмотрим формальную постановку проблемы. Пространство моделей состоит из комбинации пространства алгоритмов $A = \{A^1, \dots, A^k\}$ и пространства соответствующих им гиперпараметров $\Lambda = \{\Lambda^1, \dots, \Lambda^k\}$. Выбор алгоритма $A^j \in A$ и гиперпараметров $\lambda^j \in \Lambda$ осуществляется на основе функции потерь $L\left(A^j_{\lambda_j}, D_{train}, D_{test}\right)$ на обучающей выборке D_{train} и тестовой выборке D_{test} при k-блочной кросс-валидации:

$$A^*_{\lambda*} = \text{argmin}_{A^j \in A, \lambda \in \Lambda^j} \frac{1}{k} \sum_{i=1}^{k} L\left(A^j_{\lambda}, D^i_{train}, D^i_{test}\right).$$

На сегодняшний день наиболее распространенным подходом является оптимизация по типу «черный ящик», которая осуществляется следующим образом: вводится некоторая мера качества результатов, и алгоритм машинного обучения рассматривается как функция, отображающая гиперпараметры в качество результатов. Оптимальным решением является комбинация алгоритма и его гиперпараметров, при которых результат обучения обладает максимальным качеством. Проблема оптимизации гиперпараметров алгоритмов машинного обучения включает в себя следующие подпроблемы [17]:

- минимизация сложности оценки результатов применения алгоритма;
- достижение устойчивость оценки результатов применения алгоритма;
- оптимизация гиперпараметров со сложной структурой.

3.1. Методы выбора модели и оптимизации гиперпараметров

- *Поиск по решетке.* При поиске по решетке [8] по пространству моделей используется некоторая целевая функция для оценки качества результатов и кросс-валидация для устойчивости этой оценки. Так как гиперпараметры могут иметь значения непрерывного типа, могут потребоваться некоторые эвристические ограничения или дискретизация. Трудозатратность полного перебора растет с объемом и размерностью данных (экспоненциально), но легко распараллеливается.
- *Случайный поиск.* При случайном поиске [8] оценивается заданное число случайно сгенерированных наборов значений гиперпараметров. В отличие от поиска по решетке, не требует дискретизации и требует меньших трудозатрат. Также производительность случайного поиска можно повышать с помощью априорных знаний, используя то либо иное распределение.
- *Байесовская оптимизация.* В ходе байесовской оптимизации [33] [55] разрабатывается вероятностная модель зависимости целевой функции от значений гиперпараметров. При каждой итерации обучения применяются наиболее предпочтительные значения гиперпараметров, и затем вероятностная модель обновляется на основе результатов обучения. Байесовская оптимизация позволяет достичь оптимума быстрее, чем при поиске по решетке или случайном поиске, так как

позволяет предсказывать качество результата до выполнения обучения.

- *Оптимизация на основе градиентов.* Для некоторых алгоритмов машинного обучения, таких как метод опорных векторов или логистическая регрессия, значения гиперпараметров можно представить в виде градиента и оптимизировать их с помощью градиентного спуска [15] [16] [28]. Градиентный спуск имеет меньшую вычислительную сложность, чем поиск по решетке, и при этом не уступает в качестве случайному поиску.
- *Эволюционная оптимизация.* При оптимизации на основе эволюционных алгоритмов [7] генерируется случайная популяция наборов значений гиперпараметров, затем эти популяции итеративно изменяются и отбираются на основе оценки результатов, пока она не перестанет увеличиваться либо пока не достигнет желаемой величины.
- *Метаобучение.* При метаобучении [26] рассматривается множество образцовых наборов данных и анализируются их метапризнаки. Затем для нового набора данных вычисляется его сходство с образцовыми и выбираются алгоритмы машинного обучения, ранее оцененные как наиболее производительные на наиболее схожих образцовых наборах. Таким образом можно ограничить множество рассматриваемых алгоритмов машинного обучения до выполнения обучения.
- *Метод ансамблей* заключается в обучении на одном наборе данных нескольких алгоритмов с целью получения комбинированной модели, дающей более качественные результаты [47] [50] [52]. Качество комбинированной модели тем лучше, чем более разнообразные модели в нем задействованы [40]. Вычислительные затраты при построении

ансамбля зависят от количества и сложности входящих в него алгоритмов. Исследования показали, что при классификации наилучшие результаты получаются при числе компонент ансамбля, равном числу классов [10] [11].

3.2. Критерии принятия решений в условиях неопределенности

Проблему оптимизации гиперпараметров алгоритмов машинного обучения можно рассматривать как проблему принятия решений в условиях неопределенности. В теории принятия решений при поиске оптимального решения используются различные критерии. Использование того или иного критерия зависит от различных условий, таких как знание вероятностного распределения P состояний природы П, заданной степени оптимизма γ, заданной степени доверия ν к распределению P и т.д. Рассмотрим следующие критерии [1]:

- *Критерий Байеса.* Если известно вероятностное распределение P состояний природы П, то эффективность $b(\chi_i)$ стратегии χ_i, согласно критерию Байеса, можно определить как средневзвешенный выигрыш $u(\chi_i, \pi_j)$ с весами $P_1, \dots, P_m$:

$$b(\chi_i) = \sum_j P_j \, I(\chi_i, \pi_j).$$

  Тогда оптимальной стратегией по Байесу b^* будет стратегия с максимальной эффективностью:

$$b^* = \max_i b(\chi_i) = \max_i \sum_j P_j \, u(\pi_j, \chi_i).$$

- *Критерий Лапласа.* Если вероятностное распределение P неизвестно, то можно предположить, что состояния природы π_j равновероятны, тогда эффективность $l(\chi_i)$ стратегии χ_i, согласно критерию Лапласа, определяется формулой:

$$l(\chi_i) = \frac{1}{m}\sum_j u(\chi_i, \pi_j),$$

где m – количество состояний природы. Оптимальной стратегией по Лапласу *l** будет стратегия с максимальной эффективностью:

$$l^* = \max_i l(\chi_i) = \max_i \sum_j \frac{1}{m} u(\chi_i, \pi_j).$$

- *Критерий Вальда.* Если вероятностное распределение *P* неизвестно, то игрок может стремиться минимизировать свои риски. Тогда эффективность $v(\chi_i)$ стратегии χ_i, согласно критерию Вальда, определяется формулой:

$$v(\chi_i) = \min_j u(\chi_i, \pi_j),$$

а оптимальной стратегией по Вальду v^* будет стратегия с максимальной эффективностью:

$$v^* = \max_i v(\chi_i) = \max_i \min_j u(\chi_i, \pi_j).$$

4. Оценка результатов машинного обучения

Задача анализа полученных результатов тесно связана с задачей нахождения баланса между недообучением и переобучением, т.е. оптимальной сложности модели обучения. Рассмотрим основные методы нахождения оптимальной сложности модели обучения.

4.1. Регуляризация

Одним из методов предотвращения переобучения является регуляризация. Регуляризация – это метод, который заключается в добавлении штрафа за сложность модели [45]. Пусть имеется некоторая регрессионная модель $f(x) = y$ и функция потерь $L(f(x), y)$, тогда к функции потерь прибавляется регуляризатор $R(f)$:

$$\min_f \sum_i L(f(x_i), y_i) + \lambda R(f),$$

где λ – коэффициент регуляризации. Конкретное значение регуляризатора $R(f)$ зависит от нормы векторного пространства. Пусть $f(x) = wx$, тогда возможны следущие вариации:

- L_1-регуляризация (манхэттенское расстояние), или lasso-регрессия:

$$\min_w \sum_i L(wx_i, y_i) + \lambda \left|\left|w\right|\right|_1;$$

- L_2-регуляризация (евклидовое расстояние), или ridge-регрессия:

$$\min_w \sum_i L(wx_i, y_i) + \lambda \left|\left|w\right|\right|_2^2.$$

4.2. Информационные критерии

Информационный критерий Акаике [3] (AIC) является функцией максимизированного логарифмического правдоподобия l и числа параметров модели K:

$$AIC = -2l + 2K.$$

Другими словами, AIC включает в себя асимптотически несмещенную оценку ожидаемой относительной информации или расстояния Кульбака-Лейблера (K-L), которое представляет собой количество потерянной информации, когда мы используем модель g для приближения модели f. AIC выбирает модель-кандидата с наименьшим ожидаемым расстоянием K-L. Переобучение предотвращается за счет того, что при увеличении количества параметров первый член становится меньше, тогда как второй, штрафной член, становится больше. AIC основан на предположении, что данные имеют нормальное вероятностное распределение.

Байесовский информационный критерий [53] (BIC) или критерий Шварца также основан на функции правдоподобия и тесно связан с информационным критерием Акаике. Пусть n – размер выборки, k – количество параметров и L – функция правдоподобия, тогда:

$$BIC = -2\ln L + k\ln n.$$

Модель с более низким значением BIC является предпочтительной. BIC основан на предположении, что распределение данных относится к экспоненциальному семейству.

4.3. Оценка кластеризации

Рассмотрим некоторые внутренние метрики кластеризации [24]:

- Индекс Calinski-Harabasz [12] зависит от внутрикластерного расстояния $WGSS$, дисперсии внутри кластеров, и от межкластерного расстояния $BGSS$, дисперсии центров тяжести кластеров:

$$C = \frac{n-k}{k-1}\frac{BGSS}{WGSS}.$$

Наилучшее распределение n объектов по k кластерам будет иметь максимальное значение данного критерия.

- Индекс Davies-Bouldin [21] зависит от внутрикластерного и межкластерного расстояния:

$$C = \frac{1}{k}\sum_k \max_{k \neq k'} \frac{\delta_k + \delta_{k'}}{\Delta_{kk'}},$$

где δ_k – среднее расстояние внутри кластера k:

$$\delta_k = \frac{1}{n_k}\sum_i \left| |A_i - G| \right|,$$

и $\Delta_{kk'}$ – расстояние между двумя кластерами:

$$\Delta_{kk'} = \left| |G_k - G_{k'}| \right|.$$

Чувствителен к выбросам в кластере и не зависит от абсолютных значений координат точек в кластере. Чем меньше значение критерия, тем лучше структура кластеров.

- Индекс Silhouette [49]:

$$C = \frac{1}{n}\sum_i s(i),$$

где $s(I)$ – силуэт точки A_i:

$$s(i) = \frac{\beta(i) - \alpha(i)}{\max \alpha(i), \beta(i)},$$

такой, что $\alpha(i)$ – среднее расстояние от точки A_i до точек внутри кластера:

$$\alpha(i) = \frac{1}{n_k - 1}\sum_{ij} d\left(A_i, A_j\right),$$

и $\beta(i)$ – среднее расстояние от точки A_i до точек других кластеров:

$$\beta(i) = \frac{1}{n_k}\sum_{ij} d\left(A_i, A_j'\right).$$

4.4. Меры информативности

Также оценка результатов машинного обучения осуществляется на основе мер информативности. Рассмотрим существующие меры информативности:

- Информация Фишера [27]: количество информации $I(\theta)$, которое случайная величина X содержит о зависимой величине θ. Пусть$f(x, \theta)$ – некоторая функция правдоподобия величины θ. Если f имеет резкие скачки, это значит, что соответствующие скачкам значения X содержат много информации о величине θ. Эту зависимость и отражает информация Фишера $I(\theta)$, которая представляет собой дисперсию производной функции правдоподобия $f(x, \theta)$:

$$I(\theta) = \int \left(\frac{\delta}{\delta\theta} \log f\,(x, \theta)\right)^2 f(x, \theta) dx.$$

- Информация Шэннона [54]: количество информации $I(x)$, которое содержит некоторое значение x случайной величины X. Чем меньше вероятность $P(x)$ некоторого значения x, т.е. чем более оно «неожиданное», тем больше информации оно содержит:

$$I(x) = -\log P\,(x).$$

  На основе информации Шэннона разработано множество производных мер [58]:

  - Информационная энтропия случайной величины $H(X)$ – это математическое ожидание собственной информации $I(x)$ ее значений:

$$H(X) = -\sum_i P(x_i) \log P\,(x_i).$$

- Условная энтропия дискретных случайных величин X и Y с распределением вероятностей $P(x_i)$ и $P(y_j)$ и совместной вероятностью $P(x_i|y_j)$:

$$H(Y|X) = -\sum_i \sum_j P(x_i)P(y_j|x_i) \log P\,(y_j|x_i).$$

- Относительная энтропия (расстояние Кульбака-Лейблера [39]) дискретных случайных величин X и Y с n значениями:

$$\mathrm{D(X||}Y) = \sum_i P(x_i) \log \frac{P(x_i)}{P(y_i)}.$$

- Взаимная информация $I(X, Y)$ дискретных случайных величин X и Y:

$$I(X, Y) = H(Y) - H(Y|X).$$

- Колмогоровская сложность [56] [2] [14]: характеризует количество вычислительных ресурсов, необходимых для воспроизведения объекта y на основе описания x. Определяется как длина кратчайшей программы p, входными данными которой является x, а результатом выполнения – y:

$$K(y|x) = \min(l(p): p(x) = y).$$

- Семантическая информация: информация $inf(s)$ определяется как содержание высказывания s и измеряется на основе его логической вероятности $q(s)$ [6]:

$$inf(s) = \log \frac{1}{q(s)}.$$

- Комбинаторная информация: обычно рассматривается в рамках теории информации Шэннона, однако Колмогоров [2] выделил ее как отдельный подход. Для комбинаторного подхода существенна независимость от каких-либо вероятностных допущений.

Комбинаторная информативность позволяет определить количество знаков, необходимое для кодирования в m-значном коде сообщения x длиной l в алфавите, состоящем из N символов:

$$H(x) = l[\log_m N].$$

Представленный обзор позволяет сделать вывод о существовании множества различных, не связанных друг с другом определений информации, что делает проблематичной полную и непротиворечивую оценку информативности результатов машинного обучения.

5. Актуальные проблемы

5.1. Выбор модели обучения без учителя

На сегодняшний день технологии автоматизации машинного обучения имеют значительные ограничения. В частности, они применяются только к методам обучения с учителем [34]. При обучении с учителем оценка результатов обучения является достаточно тривиальной задачей: оценка формируется на основе точности значений, выданных алгоритмом, в сравнении со значениями, заданными в обучающей выборке. Такой способ оценки не зависит от структуры моделей обучения и, следовательно, позволяет сравнивать результаты применения различных алгоритмов с различными значениями гиперпараметров. Но такой тип обучения позволяет охватить далеко не все типы практических задач. На практике методы обучения с учителем зачастую оказываются неприменимыми в силу того, что обучающей выборки нет либо ее недостаточно для построения эффективной модели.

Выявлять признаки, зависимости и образы без обучающей выборки позволяют методы обучения без учителя, но автоматизация машинного обучения без учителя влечет ряд сложностей. В частности, при обучении без учителя на данный момент единственным способом оценки результатов обучения является оценка на основе структурных свойств полученной модели обучения, например, при кластеризации - на основе отделимости кластеров. При таком подходе критерий качества обучения вытекает из структуры модели, следовательно, сравнение результатов обучения с помощью алгоритмов, использующих различные модели, становится невозможным. Таким образом, на сегодняшний день не существует методов автоматизации машинного обучения без учителя.

Рассмотрим эту проблему на примере кластеризации. Сгенерируем методом «make_moons» модуля библиотеки sklearn.datasets набор данных размером $n = 50$ и с уровнем шума 0,05. Произведем кластеризацию двумя алгоритмами различного типа: алгоритмом K-means, основанном на центроидной модели обучения, и алгоритмом иерархической кластеризации на основе минимальной связи, реализации «sklearn.cluster.KMeans» и «sklearn.cluster.AgglomerativeClustering» соответственно. Затем произведем оценку результатов обучения метрикой Silhouette. Результат представлен на рисунках 1 и 2.

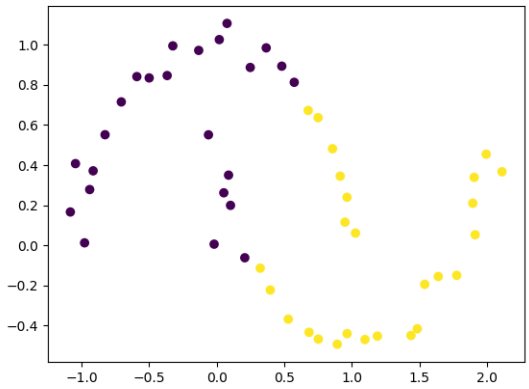

Рис. 1 – K-means, Silhouette=0,48

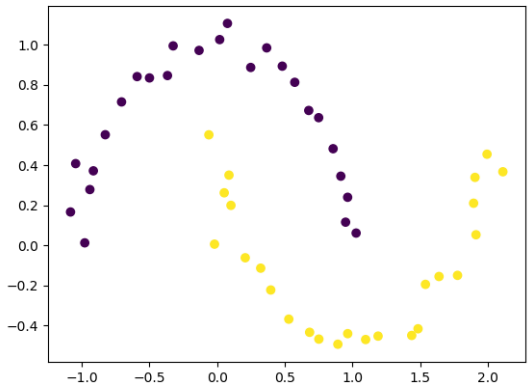

Рис. 2 – Single linkage, Silhouette=0,28

Как видно, большее значение метрики имеет результат применения алгоритма K-means, тогда как интуитивно более «правильным» является результат применения алгоритма Single linkage, так как результат кластеризации соответствует модели, заложенной при генерации набора данных. Это объясняется тем, что метрика Silhouette основана на центроидной модели кластеризации, следовательно, оценка на ее основе имеет смещение в сторону алгоритмов, которые также используют центроидную модель при обучении. Таким образом, существующие

внутренние метрики кластеризации зависят от модели обучения, что не позволяет осуществлять выбор между моделями обучения.

5.2. Устойчивость оценки результатов обучения без учителя

Другим недостатком внутренних метрик кластеризации является то, что они не всегда имеют явно выраженный глобальный или хотя бы локальный максимум или минимум, отличные от тривиальных значений количества кластеров, тем самым приводя к недообучению или переобучению. Для примера рассмотрим набор данных «Ирисы Фишера» [62], который имеет размер n = 150, размерность d = 4 и количество классов k = 3. Произведем кластеризацию алгоритмом K-means при k = 2,…,n и оценим результат метриками Calinski-Harabasz, Davies-Bouldin и Silhouette. Результаты представлены на Рисунках 3, 4 и 5. Оптимальное количество кластеров k^* для каждой метрики выделено на графике красной точкой.

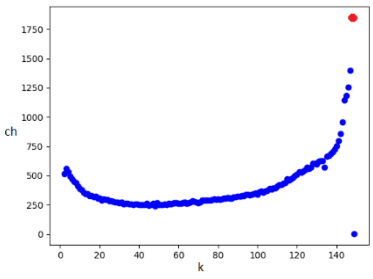

Рис. 3 – Calinski-Harabasz, k* = 149

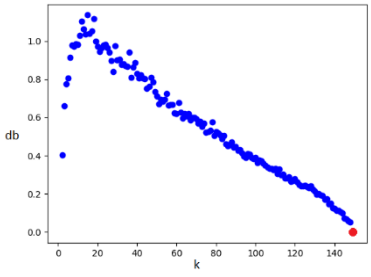

Рис. 4 – Davies-Bouldin, k* = 150

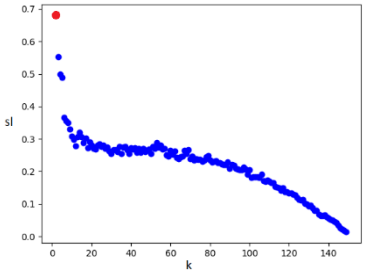

Рис. 5 – Silhouette, k* = 2

Как видно, оптимальное количество кластеров по метрикам Calinski-Harabasz и Davies-Bouldin практически совпадает с максимальным количеством кластеров, что можно рассматривать как переобучение, а оптимальное количество кластеров по метрике Silhouette совпадает с минимальным количеством, что можно рассматривать как недообучение. Таким образом, существующие внутренние метрики кластеризации

неустойчивы, и это не позволяет осуществлять оптимизацию гиперпараметров алгоритмов обучения без учителя.

Один из методов решения проблемы неустойчивости метрик кластеризации заключается в добавлении некоторого объективного априорного распределения, основанного на комбинаторных моделях, с целью придать больший вес более вероятным распределениям объектов по кластерам и меньший — менее вероятным. Устойчивость при этом достигается за счет того, что кластеризации с тривиальным количеством кластеров являются менее вероятными. Такой подход получил название «поправка на случайность» [adjustment for chance]. На данный момент существуют такие метрики с поправкой на случайность, как скорректированный индекс Рэнда [51] (adjusted Rand index) и скорректированная взаимная информация [64] (adjusted mutual information).

Для примера рассмотрим скорректированную взаимную информацию. Идея скорректированной взаимной информации заключается в корректировке взаимной информации двух случайных величин с учетом вероятностного распределения их совместных разбиений. Пусть имеется N точек, два разбиения U и V и количество точек $a_i = |U_i|$ для $U_i \subseteq U$, i=1...R и $b_j = |V_j|$ для $V_j \in V$, j=1...C, тогда обозначим общее количество способов распределить множество N по двум разбиениям U и V как Ω:

$$\Omega = \frac{(N!)^2}{\prod_i a_i! \prod_j b_j!}.$$

Каждые два совместных разбиения U и V могут быть представлены как таблица сопряженности M:

$$M = \left[n_{ij}\right]_{j=1\ldots C}^{i=1\ldots R}.$$

Пусть имеется некоторая таблица сопряженности M, тогда существуют w различных способов распределить точки так, чтобы получилась данная M:

$$w = \frac{N!}{\prod_i \prod_j n_{ij}!}.$$

Таким образом, вероятность $P(M|a,b)$ для некоторой M по отношению к множеству $\mathcal{M}$ всех возможных таблиц сопряженности определяется формулой:

$$P(M|a,b) = \frac{w}{\Omega}.$$

Пусть совместная информация *I(M)* таблицы сопряженности M определяется формулой:

$$I(M) = \sum_i \sum_j \frac{n_{ij}}{N} \log \frac{N n_{ij}}{a_i b_j}.$$

тогда средняя взаимная информация всех возможных взаимных разбиений случайных величин X и Y определяется как ожидание взаимной информации $E(I(M)|a,b)$:

$$E\big(I(X,Y)\big) = E(I(M)|a,b) = \sum_{M \in \mathcal{M}} I(M) P(M|a,b).$$

Тогда скорректированная взаимная информация $AI(X, Y)$ определяется следующим образом:

$$AI(X,Y) = \frac{I(X,Y) - E\big(I(X,Y)\big)}{\max\big(H(X),H(Y)\big) - E\big(I(X,Y)\big)}.$$

Проблема в том, что скорректированная взаимная информация применяется для оценки сходства полученного разбиения с заданным, т.е. это внешний критерий, следовательно, он неприменим при обучении без учителя. Однако метод достижения устойчивости с помощью объективных априорных распределений представляется перспективным, так как является полностью детерминированным и не зависит от экспертных оценок.

5.3. Вычислительная сложность автоматизации

Оптимизация гиперпараметров при больших размерах наборов данных либо при сложной структуре модели обучения требует значительных вычислительных затрат, она может занимать от нескольких часов до нескольких дней. Для ускорения этого процесса применяются различные эвристические методики:

- обучение на меньшей выборке из набора данных;
- сокращение количества итераций при обучении;
- использование только части признаков;
- сокращение количества итераций при кросс-валидации.

Такие методики снижают вычислительную сложность обучения, жертвуя качеством. Согласно теореме «об отсутствии бесплатных обедов» [66], для проблемы оптимизации не существует метода, позволяющего достичь того же уровня качества результатов при меньшей вычислительной сложности.

Оценим вычислительную сложность автоматизации машинного обучения, исходя из формальной постановки проблемы оптимизации модели обучения. При оптимизации производится поиск по следующим множествам:

- множество алгоритмов A;
- множество гиперпараметров Λ;
- множество частей кросс-валидации k;

Также каждая итерация включает в себя

- выполнение алгоритма A;
- расчет оценки результата L.

Обозначим временную сложность нахождения оптимального алгоритма A^* и его оптимальных гиперпараметров λ^* как $T(A^*_{\lambda^*})$, тогда ее можно описать следующей формулой:

$$T(A^*_{\lambda^*}) = k|A||\Lambda|T(A)T(L).$$

Рассмотрим пример. Допустим, имеется d-мерный набор данных из n объектов, и мы хотим произвести кластеризацию с помощью центроидной модели. Как правило, используется 10-частная кросс-валидация, следовательно, k = 10. Далее, к алгоритмам центроидной кластеризации относятся такие алгоритмы, как k-means, k-medoids, k-medians. Таким образом, пусть $|A|$ = 3. Ключевым гиперпараметром каждого из центроидных алгоритмов является количество кластеров k [32], которое может иметь значение от 1 до n. Следовательно, $|\Lambda| = n$. Пусть для оценки результата кластеризации используется индекс Calinski–Harabasz, который зависит от числа кластеров k и количества объектов n, следовательно, $T(L) = O(kn)$. Наконец, сложность самого алгоритма $T(A) = O(nkdi)$, где d – размерность набора данных и i – количество итераций, необходимых для сходимости, которое в худшем случае равно $i = 2^{\Omega(\sqrt{n})}$ [20]. Таким образом, сложность данной задачи нахождения оптимальной модели обучения $A^*_{\lambda^*}$ описывается формулой:

$$T(A^*_{\lambda^*}) = O\left(2^{\Omega(\sqrt{n})}k^2n^3d\right).$$

Таким образом, оптимизация обучения приводит к сверхполиномиальному увеличению вычислительной сложности.

Лабораторные работы

Лабораторная работа 1: «Автоматизация конструирования признаков»

Цель работы: приобрести практический опыт автоматизации конструирования признаков.

Задание: найти оптимальное представление набора данных с помощью метода главных компонент (МГК) и критериев отбора компонент.

Шаги:

1. Сгенерировать набор данных произвольной размерности d, например, методом «sklearn.datasets.make_blobs».
2. Получить дисперсии значений компонент. Это можно сделать с помощью класса «sklearn.decomposition.PCA» и атрибута «explained_variance_».
3. Определить пороговые значения дисперсии компонент с помощью:
    - правила Кайзера;
    - правила сломанной трости;
    - «метода локтя».
4. Построить диаграммы, например, с помощью метода «matplotlib.pyplot.bar», отметить на диаграмме пороговые значения по каждому критерию.
5. Применить МГК с количеством компонент, полученным на основе одного из критериев.

Отчет должен содержать следующие файлы:

1. Сгенерированный набор данных в формате .csv.
2. Диаграмма, полученная на шаге 4.
3. Преобразованный набор данных, полученный на шаге 5, в формате .csv.

Лабораторная работа 2: «Автоматизация выбора модели и оптимизации гиперпараметров»

Цель работы: приобрести практический опыт автоматизации классификации.

Задание: разработать автоматизированный классификатор.

Шаги:

1. Сгенерировать набор данных для классификации, например, методом «sklearn.datasets.make_blobs».
2. Выбрать не менее 3 алгоритмов классификации, например, из библиотеки «sklearn».
3. Выбрать целевую функцию. Если была использована библиотека «sklearn», то функции можно выбрать из модулю «sklearn.metrics».
4. Для каждого выбранного классификатора реализовать автоматическую оптимизацию гиперпараметров методами поиска по решетке и случайного поиска с кросс-валидацией. При использовании «sklearn» можно воспользоваться методами «model_selection.GridSearchCV» и «model_selection.RandomizedSearchCV».
5. Реализовать автоматизированный выбор классификатора из сформированного набора с оптимизированными гиперпарамтрами на основе выбранной целевой функции.
6. Применить выбранный классификатор.

Отчет должен содержать следующие файлы:

1. Сгенерированный набор данных в формате .csv.
2. Текстовый файл, в котором указаны:
    - выбранные классификаторы;
    - выбранная целевая функция;

- список полученных в результате оптимизации значений гиперпараметров для каждого классификатора и значения целевой функции.

3. Результат классификации в формате .csv.

Список литературы

1. Блягоз З.У., Попова А.Ю. Принятие решений в условиях риска и неопределенности // Вестник Адыгейского государственного университета. 2006. №4.
2. Колмогоров А.Н. Три подхода к определению понятия «количество информации» // Проблемы передачи информации, 1965, 1:1, 3—11.
3. Akaike H. A new look at the statistical model identification // IEEE Transactions on Automatic Control, 1974, Vol. 19, Pp. 716—723.
4. Automated machine learning with azureml. https://github.com/Azure/MachineLearningNotebooks/tree/master/how-to-use-azureml/automated-machine-learning, accessed: 2019-04-10.
5. Auto-ml: Automated machine learning for production and analytics. https://github.com/ClimbsRocks/*automl*, accessed: 2019-04-10.
6. Bar-Hillel Y., Carnap R. An Outline of a Theory of Semantic Information // Technical Report No. 247, October 27, 1952, Research Laboratory of Electronics. – 49.
7. Bergstra J., Bardenet R., Bengio Y., Kegl B. Algorithms for hyperparameter optimization // Proceedings of the 24th International Conference on Neural Information Processing Systems, 2011, 2546-2554.
8. Bergstra J., Bengio Y. Random search for hyper-parameter optimization // Journal of Machine Learning Research, 2012, 13, 281–305.
9. Blumer A., Ehrenfeucht A., Haussler D., Warmuth M.K. Occam's razor // Information processing letters, 1987, 24(6), 377-380.
10. Bonab H.R., Can F. A Theoretical Framework on the Ideal Number of Classifiers for Online Ensembles in Data Streams // 25th Conference on Information and Knowledge Management, 2016.

11. Bonab H.R., Can F. Less Is More: A Comprehensive Framework for the Number of Components of Ensemble Classifiers // IEEE Transactions on Neural Networks and Learning Systems, 2017.
12. Cali´nski T., Harabasz J. A dendrite method for cluster analysis //Communications in Statistics-theory and Methods, 1974, 3: 1-27.
13. Cangelosi R., Goriely A. Component retention in principal component analysis with application to cDNA microarray data // Biology Direct, 2007.
14. Chaitin G.J. On the Length of Programs for Computing Finite Binary Sequences // Journal of Association for Computing Machinery, 1966, v. 13, No. 4, pp. 547–569.
15. Chapelle O., Vapnik V., Bousquet O., Mukherjee S. Choosing multiple parameters for support vector machines // Machine Learning, 2002, 46: 131–159.
16. Chuong B., Chuan-Sheng F., Andrew Y.Ng. Efficient multiple hyperparameter learning for log-linear models // Advances in Neural Information Processing Systems, 2008, 20, 377-384.
17. Claesen M., De Moor B. Hyperparameter Search in Machine Learning //arXiv:1502.02127, 2015.
18. Darwin-spark cognition. https://github.com/sparkcognition/darwin-sdk, accessed: 2019-04-10.
19. Datarobot usage examples. https://github.com/datarobot/datarobot-sagemaker-examples, accessed:2019-04-10.
20. David A., Vassilvitskii S. How Slow is the k-means Method? // Proceedings of the 21nd Annual Symposium on Computational Geometry, 2006, 144–153.

21. Davies D.L., Bouldin D.W. A Cluster Separation Measure // IEEE Transactions on Pattern Analysis and Machine Intelligence, 1979, PAMI-1 (2): 224–227.
22. de S´a A.G.C., Pinto W.J.G.S., Otavio L.V.B.O., Pappa G.L. RECIPE: A Grammar-Based Framework for Automatically Evolving Classification Pipelines // Lecture Notes in Computer Science, 2017, 10196, 246–261.
23. de S´a, A.G.C., Freitas A.A., Pappa G.L. Automated Selection and Configuration of Multi-Label Classification Algorithms with Grammar-Based Genetic Programming // Parallel Problem Solving from Nature – PPSN XV. Lecture Notes in Computer Science. Springer International Publishing, 2018, 11102: 308–320.
24. Desgraupes B. Clustering indices, University of Paris Ouest-Lab Modal'X. vol.1, 2013, pp. 1-34.
25. Escalante H., Montes M., Sucar E. Particle Swarm Model Selection // Journal of Machine Learning Research, 2009, 10, 405–440.
26. Feurer M., Klein A., Eggensperger K., Springenberg J., Blum M., Hutter F. Efficient and Robust Automated Machine Learning // Advances in Neural Information Processing Systems 28 (NIPS 2015). — 2015.
27. Fisher R.A. Theory of statistical estimation // Proceedings Cambridge Philosophical Society, 1925, 22(5), pp. 700–725.
28. Franceschi L., Donini M., Frasconi P., Pontil M. Forward and Reverse Gradient-Based Hyperparameter Optimization // Proceedings of the 34thInternational Conference on Machine Learning, 2017, 70, 6-11.
29. Google cloud automl. https://cloud.google.com/automl/, accessed: 2019-04-10.
30. H2o.ai automl github. https://github.com/h2oai/h2o-3, accessed: 2019-04-10.

31. H2o-driverlessai. http://docs.h2o.ai/driverless-ai/latest-stable/docs/userguide/index.html, accessed: 2019-04-10.
32. Hamerly G., Elkan Ch. Learning the K in K-Means // Advances in Neural Information Processing Systems, 2004, 17.
33. Hutter F., Hoos H., Leyton-Brown K. Sequential model-based optimization for general algorithm configuration // Learning and Intelligent Optimization, Lecture Notes in Computer Science, 2011, 6683: 507–523.
34. Hutter F., Kotthoff L., Vanschoren J. Automatic machine learning: methods, systems, challenges // Challenges in Machine Learning, New York: Springer,2019.
35. Jin H., Song Q., Hu X. Auto-keras: An efficient neural architecture search system // arXiv: 1806.10282, 2018.
36. Kearns M.J., Vazirani U.V. An introduction to computational learning theory, chapter 2. MIT press, 1994.
37. Komer B., Bergstra J., Eliasmith C. Hyperopt-sklearn: Automatic hyperparameter configuration for scikit-learn // ICML workshop on Automated Machine Learning, 2014.
38. Kotthoff L., Thornton C., Hoos H.H., Hutter F., Leyton-Brown K. Auto-WEKA 2.0: Automatic model selection and hyperparameter optimization in WEKA // Journal of Machine Learning Research. — 2017.
39. Kullback S., Leibler R.A. On information and sufficiency // The Annals of Mathematical Statistics. 22(1), 1951. pp. 79-86.
40. Kuncheva L., Whitaker C. Measures of diversity in classifier ensembles and Their Relationship with the Ensemble Accuracy // Machine Learning, 2003,51:2, 181–207.
41. Ludwig. https://github.com/uber/ludwig, accessed: 2019-04-10.

42. Mljar. https://github.com/mljar/mljar-api-python, accessed: 2019-04-10.
43. Mohr F., Wever M., Hullermeier E. ML-Plan: Automated machine learning via hierarchical planning // Machine Learning, 2018, 107 (8–10): 1495–1515.
44. Mohri M., Rostamizadeh A., Talwalkar A. Foundations of Machine Learning. MIT Press, Second Edition, 2018.
45. Neumaier A. Solving ill-conditioned and singular linear systems: A tutorial on regularization // SIAM Review 40, 1998, pp. 636–666.
46. Olson R.S., Bartley N., Urbanowicz R.J., Moore J.H. Evaluation of a tree-based pipeline optimization tool for automating data science // Proceedings of the Genetic and Evolutionary Computation Conference (GECCO), 2016,485–492.
47. Opitz D., Maclin R. Popular ensemble methods: An empirical study // Journal of Artificial Intelligence Research, 1999, 11, 169–198.
48. Pearson K. On Lines and Planes of Closest Fit to Systems of Points in Space// Philosophical Magazine, 1901, 2 (11): 559–572.
49. Peter J.R. Silhouettes: a Graphical Aid to the Interpretation and Validation of Cluster Analysis // Computational and Applied Mathematics, 1987, 20: pp.53–65.
50. Polikar R. Ensemble based systems in decision making // IEEE Circuits and Systems Magazine, 2006, 6:3, 21–45.
51. Rand W.M. Objective criteria for the evaluation of clustering methods// Journal of the American Statistical Association. American Statistical Association, 1971, 66 (336): 846–850.
52. Rokach L. Ensemble-based classifiers // Artificial Intelligence Review, 2010,33:1-2, 1–39.

53. Schwarz G.E. Estimating the dimension of a model // Annals of Statistics,1978, 6(2): 461–464.
54. Shannon C.E.A Mathematical Theory of Communication // Bell System Technical Journal, 1948. Vol. 27, pp. 379-423.
55. Snoek J., Larochelle H., Adams R. Practical Bayesian Optimization of Machine Learning Algorithms // Advances in Neural Information Processing Systems,2012, 2, 2951-2959.
56. Solomonoff R.J. A Formal Theory of Inductive Inference // Information and Control, 1964, v. 7, No. 1, pp. 1–22; No.2, pp. 224–254.
57. Swearingen Th., Drevo W., Cyphers B., Cuesta-Infante A., Ross A., Veeramachaneni K. ATM: A distributed, collaborative, scalable system for automated machine learning // 2017 IEEE International Conference on Big Data (Big Data), 2017, IEEE: 151–162.
58. Thomas M.C., Joy A.T. Elements of Information Theory, 2nd Edition. Wiley-Interscience, 2006. pp. 792.
59. Thorndike R.L. Who Belongs in the Family? // Psychometrika, 1953, 18 (4):267–276.
60. Thornton C., Hutter F., Hoos H.H., Leyton-Brown K. Auto-WEKA: Combined selection and hyperparameter optimization of classification algorithms // 19th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD'13), 2013.
61. Transmogrifai. https://github.com/salesforce/TransmogrifAI, accessed: 2019-04-10.
62. UCI Machine Learning Repository: Iris Data Set. http://archive.ics.uci.edu/ml/datasets/Iris (дата обращения: 18.11.2019).
63. Valiant L. A theory of the learnable // Communications of the ACM, 27, 1984.

64. Vinh N.X., Epps J., Bailey J. Information theoretic measures for clusterings comparison // Proceedings of the 26th Annual International Conference on Machine Learning - ICML, 2009, pp. 1073-1080.
65. Warne R.T., Larsen R. Evaluating a proposed modification of the Guttmanrule for determining the number of factors in an exploratory factor analysis // Psychological Test and Assessment Modeling, 2014, 56: 104–123.
66. Wolpert D.H., Macready, W.G. No Free Lunch Theorems for Optimization //IEEE Transactions on Evolutionary Computation, 1997, 1(67): 67-82.

Баймуратов Ильдар Раисович

Методы автоматизации машинного обучения

Учебное пособие

В авторской редакции
Редакционно-издательский отдел Университета ИТМО
Зав. РИО Н.Ф. Гусарова
Подписано к печати
Заказ №
Тираж
Отпечатано на ризографе

Редакционно-издательский отдел
Университета ИТМО
197101, Санкт-Петербург, Кронверский пр., 49