Активная стратегия совместного выбора алгоритма классификации и его гиперпараметров

Ефимова В.А., мл. науч. сотр., МНЛ КТ Университета ИТМО.
efimova@rain.ifmo.ru

Аннотация

Существует множество алгоритмов машинного обучения, но для обработки конкретного набора данных нужно не только выбрать один из них, но еще настроить его гиперпараметры. В этой работе рассматривается задача одновременного и автоматического выбора алгоритма классификации и настройки его гиперпараметров.

Введение

Во многих областях современной жизни возникают задачи, в которых на основе обучающей выборки нужно некоторым образом охарактеризовать данные. Для их решения создано множество алгоритмов, но обычно каждый алгоритм эффективен не для всех возникающих задач, а на некотором их подмножестве. Алгоритмы обладают гиперпараметрами, выбор которых сильно влияет на качество решения, получаемого в результате работы этих алгоритмов. Одновременный выбор алгоритма и его гиперпараметров для определенного вида данных является комплексной задачей, не получившей решения. На сегодняшний день она разбивается на две подзадачи, решаемые независимо: выбор алгоритма из ограниченного множества и оптимизация гиперпараметров фиксированного алгоритма.

Первая подзадача обычно решается с помощью мета-обучения [2], а для решения второй разработано несколько алгоритмов настройки гиперпараметров, основные из которых мы рассмотрим далее. Кроме того, существует библиотека Auto-WEKA [3], в которой подбор алгоритмов и их гиперпараметров осуществляется полным перебором, но это является не самым эффективным методом. Цель данной работы — предложить алгоритм одновременного выбора алгоритма классификации и его гиперпараметров, основанный на обучении с подкреплением, а также оптимизировать время его работы.

Основные определения

В данной работе рассматривается обучение с учителем: нахождение функции $f : X \to Y$, где X — множество признаков, а Y — конечное множество меток. Алгоритм обучения с учителем A сопоставляет обучающей выборке $D_{train} = \{d_1, d_2, \ldots, d_n\}$, где $d_i = (x_i, y_i) \in X \times Y$, функцию вида f. Полученная в ходе обучения алгоритма функция f применяется для предсказания метки объекта [1].

Конкретный вид алгоритма A задается с помощью вектора гиперпараметров $\lambda = \{\lambda_1, \lambda_2, \ldots, \lambda_n\}$, выбираемых из пространства гиперпараметров Λ_A. Гиперпараметры влияют на работу алгоритма A, поэтому гиперпараметризованный алгоритм будем далее обозначать A_λ.

Рассмотрим задачу выбора алгоритма. Дано некоторое множество алгоритмов $\mathcal{A} = \{A^1, A^2, \ldots A^m\}$ и обучающая выборка $D = \{d_1, d_2, \ldots d_n\}$, где $d_i = (x_i, y_i) \in X \times Y$. Нужно выбрать алгоритм $A^* \in \mathcal{A}$, который окажется наиболее эффективным. Эффективность оценивается с помощью k-стратового скользящего контроля (k-fold cross validation), а ее критерием является эмпирический риск $Q(A, D)$.

Тогда проблему выбора алгоритма можно записать в виде требования минимизации эмпирического риска:

$$A^* \in \operatorname*{argmin}_{A_j \in \mathcal{A}} Q(A^j, D).$$

Однако описанная методология предполагает запуск всех алгоритмов и последующее сравнение, что требует значительного времени. С этим помогает справиться мета-обучение — подход, целью которого является решение задачи выбора алгоритмов без непосредственного применения каждого из них. Есть множество наборов данных $\mathcal{D}$, для каждого из них вычисляется некоторый вектор мета-признаков. Каждый алгоритм запускается на каждом наборе данных из $\mathcal{D}$, после чего вычисляется эмпирический риск, на основе которого формируются метки классов. Затем мета-классификатор обучается на полученных результатах. В качестве x_i выступает вектор мета-признаков, а в качестве y_i — самый эффективный алгоритм.

Оптимизация гиперпараметров заключается в подборе таких $\lambda^* \in \Lambda$, при которых заданный алгоритм A будет наиболее эффективен. Это можно записать записать в виде формулы:

$$\lambda^* \in \operatorname*{argmin}_{\lambda \in \Lambda} Q(A_\lambda, D).$$

Изначально гиперпараметры часто настраивались вручную, но это требовало много времени или же приводило к неоптимальному результату. Теперь же разработаны алгоритмы, автоматически решающие данную задачу. Существуют подходы к решению задачи оптимизации гиперпараметров: Grid search [8], Random Search [5], Стохастический градиентный спуск [6], Bayesian Optimization [7] (его частный случай — Sequental Model-Based Optimization, SMBO), [4]) Tree-structured Parzen estimator (TPE) [12].

Особо стоит выделить работу [3], в которой предложено настраивать гиперпараметры с помощью алгоритма SMBO [4]. Этот алгоритм затрагивает проблему exploration/exploitation. В [11] предложен Sequential model-based algorithm configuration (SMAC), работающий на основе SMBO и позволяющий решить поставленную в этой работе задачу.

Ключевая идея SMAC'а заключается в достижении наибольшей эффективности, жертвуя точностью в обмен на сокращение времени вычислений. Чтобы оценить, насколько новый вектор гиперпараметров улучшит эффективность, при его рассмотрении приходится запускать не только алгоритм, параметризованный этим вектором, но и алгоритм, параметризованный лучшим вектором на текущий момент. Сначала алгоритм запускается на одной страте (fold), затем на двух и так далее, до числа запусков алгоритма на предыдущем наборе. Если набор гиперпараметров выдерживает такое сравнение, то он запускается на еще одной страте. Таким образом, заведомо плохой вектор гиперпараметров может быть исключен уже после одного запуска.

SMAC и TPE реализованы в открытой библиотеке Auto-WEKA, позволяющей автоматически выбирать лучший из 27 базовых алгоритмов, 10 мета-алгоритмов и 2 ансамблевых алгоритмов лучший, одновременно настраивая его гиперпараметры [3].

Существуют и другие алгоритмы, но они применяются для решения более узких задач, так что их рассматривать не будем.

Совместный выбор алгоритма и его гиперпараметров

Мы уже рассмотрели выбор алгоритма и оптимизацию гиперпараметров, в этой работе предлагается одновременно решать эти задачи. Запишем формально.

Дан набор алгоритмов $\mathcal{A} = \{A^1, A^2, \ldots, A^k\}$, с каждым из которых

связано пространство гиперпараметров $\Lambda^1, \Lambda^2, \ldots, \Lambda^k$. Также даны обучающая и тестовая выборки. Нужно найти такой алгоритм $A^*_{\lambda^*}$, что:

$$A^*_{\lambda^*} \in \underset{A^j \in \mathcal{A}, \lambda \in \Lambda^j}{\operatorname{argmin}} Q(A^j_\lambda, D).$$

Несмотря на то, что проблема одновременного выбора алгоритма и гиперпараметров важна на практике, существует мало работ на эту тему. И рассмотрены в них только частные случаи решения. Например, в работе [9] предложено около трехсот комбинаций алгоритмов и их гиперпараметров для шести изученных алгоритмов. В работе [10] рассмотрена работа двадцати мета-классификаторов на четыреста шестидесяти шести наборах данных. Можно запускать алгоритмы с какими-то фиксированными параметрами, а потом упорядочивать их с помощью скользящего контроля (cross-validation), но такой подход неэффективен, так как можно легко упустить наилучшее решение.

Auto-WEKA позволяет одновременно выбирать алгоритм машинного обучения и настраивать его гиперпараметры. Но сделано это практически полным перебором, что не эффективно по времени. Создатели бибиотеки Auto-WEKA предполагают, что для фиксированного набора данных поиск приемлемой конфигурации на четырехъядерном процессоре займет более 30 часов [3]. В данной работе описан метод, который позволит ускорить поиск оптимальной конфигурации.

Представим задачу выбора наиболее эффективного алгоритма и его гиперпараметров как задачу активного обучения, которая сводится к задаче о многоруком контекстном бандите.

Активное обучение — подход, применимый к задачам машинного обучения, в которых можно разметить лишь малую часть данных. Алгоритму активного обучения разрешается обращаться к оракулу $\mathcal{O}$, который определит класс указанного объекта, но нужно снизить количество обращений, постаравшись максимизировать информативность полученных ответов.

Выбор объектов для разметки может быть представлен как поиск компромисса между исследованием и использованием (exploration/exploitation). С одной стороны, алгоритм активного обучения, который будет пытаться размечать те данные, которые дадут наибольший прирост качества работы итогового алгоритма, рискует оказаться в ситуации, что никогда не запросит метки у устойчиво классифицируемых объектов, которая, однако, отличается от результата работы классификатора. С другой стороны, алгоритм, который не будет мало ориентироваться на ожидаемую полезность

запроса метки объекта, рискует сделать много запросов, совершенно не влияющих на работу классификатора. Поиск такого компромисса решается через сведение задачи к задаче о многоруком бандите [13].

Вернемся к поставленной задаче. Дано множество алгоритмов $\mathcal{A} = \{A^1, A^2, \dots A^k\}$, каждому из которых соответсвует пространство гиперпараметров $\Lambda^1, \Lambda^2, \dots, \Lambda^k$. Но чтобы точно оценить результат работы алгоритма $A^j_\lambda, \lambda \in \Lambda^j$, его нужно запустить с гиперпараметрами λ. Запуск позволит улучшить точность ответа, но займет некоторое время. Для поиска наилучшего решения данной задачи воспользуемся алгоритмом обучения с подкреплением, а именно решением задачи о многоруком бандите (multi-armed bandit).

В этой задаче на каждой итерации агент выбирает одну из N рук. В каждый момент времени t агент выбирает руку a_i и получает выигрыш $r(i,t)$. Обозначим за a^*_t руку, выбор которой в момент t приведет к максимальному выигрышу, то есть

$$a^*_t = \underset{i=1..N}{\operatorname{argmax}}\, r(i,t).$$

Пусть $\delta_i(t) = r(a^*_t, t) - r(i,t)$ — разница между оптимальным и полученным выигрышем в момент времени t. А $regret(t) = \delta_{a_t}(t)$ — потери при выборе руки a_t. Цель агента — минимизировать суммарные потери за конечное время T:

$$R(T) = \sum_{t=1}^{T} regret(t).$$

Такая задача возникает при извлечении информации и при создании рекомендательных систем [14]. Очевидно, что если выбирать все руки по очереди некоторое число раз, то потери будут достаточно велики. Так что для решения данной задачи было разработано несколько алгоритмов, например Epsilon-greedy, UCB1, Softmax [15].

Теперь сведем задачу одновременного выбора алгоритма и его гиперпараметров к задаче о многоруком бандите. В качестве рук будем рассматривать пары из алгоритма классификации и алгоритма настройки гиперпараметров (A, A_{tune}). После выбора руки будем запускать один или все алгоритмы оптимизации гиперпараметров на соответсвующих им алгоритмах классификации с ограничением по времени.

Время между алгоритмами можно распределять двумя способами:

1. На каждой итерации основного алгоритма разыгрывать временной отрезок и запускаеть один алгоритм.

Algorithm 1 Contextual Multi-Armed Bandits for Conbined Selection and Hyperparameter Optimization of Classification Algorithms

Require: $M, t, Algorighms = \{A^i, A^i_{tune}\}, i = 1..N$ ▷ t — time budget for one iteration

for $i = 1, \ldots, N$ **do**
  $e_i \leftarrow Execute(A^i, A^i_{tune}, t)$
end for
$History = \emptyset$
$best\_err = \min_{i=1..N}(e_i)$
for $j = 1, \ldots, M$ **do**
  $i \leftarrow MAB(Algorithms, History)$
  $e_i \leftarrow Execute(A^i, A^i_{tune}, t)$
  $History.add(i, e_i)$
  $best\_err \leftarrow min(best\_err, e_i)$
end for

2. На каждой итерации основного алгоритма пересчитывать вероятности улучшения текущего минимального эмпирического риска и в соответствии с ними распределять части фиксированного временного отрезка. Тогда запускаются все алгоритмы.

Запустив алгоритм, мы получим эмпирический риск, достигаемый при последней найденной конфигурации. С его помощью мы определим полученную награду: как разницу между оптимальным эмпирическим риском, найденный в ходе предыдущих итераций, и текущим эмпирическим риском.

Псевдокод приведен в листинге 1. Здесь рассматривается простейший случай: число итераций ограничено константой M и на каждой из них разыгрывается фиксированный отрезок времени t.

Заключение

Был рассмотрен алгоритм, решающий задачу одновременного выбора алгоритма и его гиперпараметров. Он позволяет ускорить существующее решение данной задачи, так как работает гарантированно быстрее, чем полный перебор.

Предложенный алгоритм можно улучшить, изначально отсортировав пары алгоритмов с помощью мета-обучения. Кроме того, можно

ввести контекст и воспользоваться алгоритмом для решения задачи о многоруком контекстном бандите. В качестве контекста тогда будем использовать эмпирический риск, полученный при запуске алгоритма классификации на наборах данных, отобранных с помощью метаобучения.

Литература

[1] Hastie T., Tibshirani R., Friedman J. Unsupervised Learning. NY: Springer, 2009.

[2] Giraud-Carrier C., Vilalta R., Brazdil P. Introduction to the Special Issue on Meta-Learning // Machine Learning. 2004. Vol. 54(3). P. 187-193.

[3] Thornton C., Hoos H. H., Hutter F., Leyton-Brown K. Auto-WEKA: Automated Selection and Hyper-Parameter Optimization of Classification Algorithms // CoRR, abs/1208.3719. 2012.

[4] Hutter F., Hoos H. H., Leyton-Brown K. Sequential Model-Based Optimization for General Algorithm Configuration // Technical Report TR-2010-10, University of British Columbia, Computer Science. 2010.

[5] Bergstra J., Bengio Y. Random search for hyper-parameter optimization // Journal of Machine Learning Research. 2012. Vol 13. P. 281–305.

[6] Bottou L. On-line Learning in Neural Networks, ch. On-line Learning and Stochastic Approximations, p 9–42. Cambridge University Press, 1998.

[7] Snoek J., Larochelle H., Adams R. P. Practical bayesian optimization of machine learning algorithms // Annual Conference on Neural Information Processing Systems. 2012. P. 2960–2968.

[8] Friedman J., Hastie T., Tibshirani R. The Elements of Statistical Learning: Data Mining, Inference, and Prediction. Springer Series in Statistics, 2009.

[9] Leite R., Brazdil P., Vanschoren J. Selecting classification algorithms with active testing // The International Conference on Machine Learning and Data Mining. 2012. P. 117–131.

[10] Sun Q., Pfahringer B. Pairwise meta-rules for better meta-learning-based algorithm ranking // Machine Learning. 2013. Vol 93(1). P. 141–161.

[11] Hutter F., Hoos H., Leyton-Brown K. Sequential model-based optimization for general algorithm configuration // Learning and Intelligent Optimization Conference. 2011. P. 507–523.

[12] Bergstra J., Bardenet R., Bengio Y., Kegl B. Algorithms for Hyper-Parameter Optimization // Annual Conference on Neural Information Processing Systems, 2011. P. 2546–2554.

[13] Ganti R., Gray A. G. Building bridges: Viewing active learning from the multi-armed bandit lens // arXiv preprint arXiv:1309.6830. 2013.

[14] Bouneffouf D., Bouzeghoub A., Gancarski A. L. A contextual-bandit algorithm for mobile context-aware recommender system // ICONIP (3). 2012. P. 324-331.

[15] Sutton R.S.,Barto A. Reinforcement learning: An introduction. MIT press, 1998.