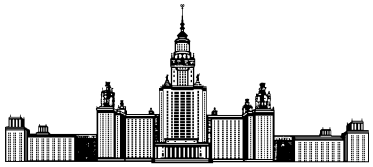


Московский государственный университет имени М. В. Ломоносова

Факультет вычислительной математики и кибернетики

Кафедра математических методов прогнозирования

Амир Мирас Сабыргалиулы

Оптимизация алгоритма бустинга под решающие деревья с повышенной субоптимальностью

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА

Научный руководитель:

к.ф-м.н.

В.В. Китов

Москва, 2017

Содержание

Аннотация

В данной работае рассмотрены семейтва заглядывающих вперед деревьев решений. Они были модифицированы специально для алгоритма градиентного бустинга стохастических путем. Проведено экспериментальное сравнение классического градиентного бустинга с градиентным бустингом над заглядывающими вперед решающими деревьями. По результатом эксперимента на большинстве реальных задач наилучшие результаты показал предложенный алгоритм градиентного бустинга.

1 Введение

Градиентный бустинг [4] представляет собой мощное семейство методов машинного обучения, которое показало значительный успех в широком диапазоне практических применений. Например, один из представителей данного семейства xgboost набрал большую популярность среди команд-победителей ряда конкурсов по анализу данных.

Основная идея градиентного бустинга заключается в последовательном построении композиции алгоритмов машинного обучения, когда каждый следующий алгоритм стремится компенсировать недостатки композиции всех предыдущих алгоритмов. Высокая гибкость алгоритма позволяет вводить различные изменения в дизайн метода, таким образом, делая метод подходящим для многих задач машинного обучения.

Обычно в качестве базовых алгоритмов используются так называемые «слабые» модели. К их числу относятся неглубокие деревья решений, строящиеся жадным способом. Одним из главных недостатков которого является необратимость ущерба причиненного неправильным решением: как только был выбран признак для разделения объектов в определенном узле, не существует способа для возврата и выбора другого признака, то есть метод сходится к локально оптимальному решению в каждом узле. Кроме того, жадные алгоритмы требуют определенного количества времени и если имеется возможность задействовать простаивающий вычислительный ресурс, не в состоянии произвести лучшее дерево. Для преодоления этой проблемы, существует другое семейство методов, которые пытаются предсказать полезность разделения в узле путем оценки его влияния на более глубокие узлы дерева.

Первые попытки использовать данную идею были осуществлены в работах [9] и [7]. Оба исследования пришли к одинаковому выводу: заглядывающие вперед алгоритмы создают деревья хуже жадных алгоритмов. Однако, эти работы не рассматривают способы использования заглядывающих вперед алгоритмов в качестве базовых моделей в бустинге. Мы попробуем обобщить данный алгоритм для задачи регрессии и применить в качестве базовых моделей в градиентном бустинге.

Показано [7], что рост глубины k, на которую заглядывает дерево, приводит к переобучению. Предполагается, что это является побочным эффектом чрезмерного

поиска критерия дробления: локально оптимальное дробление в узле не обязательно улучшает точность глобального дерева. Поэтому в данной работе разработаны стохастические варианты данных методов SCAT-k и ECAT-k для выбора лучшего критерия. Методы умеют заглядывать на k шагов вперед, тем самым учитывают информацию из более глубоких узлов дерева, но для оптимизацию критерия дробления используют только подмножество доступных ему параметров. Это сделано для избежания переобучения.

В работе [6] показано, что для большинства наборов данных высокая точность предсказания достигается с заглядывающими на два шага вперед деревьями. Это означает, что небольшое расширение локальной области поиска одноуровневых поддеревьев обычно достаточно для нахождения наиболее точной модели. Поэтому мы рассмотрим алгоритмы SCAT-k и ECAT-k при $k = 2$.

Проведено сравнение алгоритмов бустинга над заглядывающими вперед деревьями с выбором лучшего критерия жадным способом, SCAT-2 и ECAT-2. Показано, что на большинстве рассмотренных в работе задач предложенные методы работает лучше жадных алгоритмов.

1.1 Градиентный бустинг

В данной секции мы определим метод градиентного бустинга и введем обозначения.

Пусть $X = (x_1, ..., x_n)$ – обучающая выборка, где каждый объект описывается множеством признаков $F = (f_1, ..., f_d)$. Предположим, что каждый признак вещественный. $Y = (y_1, ..., y_n)$ – вектор ответов обучающей выборки. Будем искать зависимость $a_T(x) : X \to Y$ в виде взвешенной суммы базовых алгоритмов:

$$a_T(x) = \sum_{t=0}^{T} \gamma_t b_t(x), \tag{1}$$

где каждый b_t из некоторого множества базовых алгоритмов $\mathcal{A}$.

Композицию будем строить путем «жадного наращивания», поэтапно увеличивая количество базовых алгоритмов. На каждом шаге $1 \leqslant t \leqslant T$ будем предполагать, что уже построен алгоритм $a_{t-1}(x)$, и хотим выбрать следующий базовый алгоритм $b_t(x)$ так, чтобы как можно сильнее уменьшить функционал ошибки Q:

$$a_t(x) = a_{t-1}(x) + \gamma_t b_t(x) \tag{2}$$

$$Q(a_t, X) = \sum_{i=1}^{n} L(y_i, a_{t-1}(x_i) + \gamma_t b_t(x_i)) \to \min, \tag{3}$$

где $L(y, z)$ – некоторая дифференцируемая функция потерь, которая выбирается в зависимости от типа задачи: в задачах регрессии это обычно квадратичная, а в случае классификации логистическая функция потерь. В итоге нам необходимо решить задачу минимизации в n-ом пространстве для вещественного функционала Q, которая зависит от точек $\{a_T(x_i)\}_{i=1}^n$. Для решения сделаем один шаг градиентного спуска, двигаясь в сторону антиградиента $\{-g_t(x_i)\}_{i=1}^n$:

$$g_t(x) = \left[\frac{\partial L(y, f(x))}{\partial f(x)}\right]_{f(x)=a_{t-1}(x)}. \tag{4}$$

Таким образом, нашу оптимизационную задачу можно заменить на классическую задачу обучения по прецедентом с обучающей выборкой $\{(x_i, -g_t(x_i))\}_{i=1}^n$, которую можно решить с помощью метода наименьших квадратов:

$$b_t(x) = \operatorname*{argmin}_{b \in \mathcal{A}} \sum_{i=1}^{n} [-g_t(x_i) - b(x_i)]^2 \tag{5}$$

После того, как нашли новый алгоритм, линейным поиском можно подобрать коэффициент при нем:

$$\gamma_n = \underset{\gamma \in \mathbb{R}}{\operatorname{argmin}} \sum_{i=1}^{T} L(y_i, a_{t-1}(x_i) + \gamma b_t(x_i)). \tag{6}$$

1.2 Алгоритм построения решающего дерева

В качестве базовых моделей в градиентном бустинге используются решающие деревья регрессии. Рассмотрим простой жадный алгоритм его построения, основанный на принципе «разделяй и властвуй» [8]. Идея алгоритма заключается в последовательном дроблении выборки на две части до тех пор, пока в листе дерева количество объектов не окажется меньше или равно заданного параметра msl. Псевдокод алгоритма решающего дерева показан в Алгоритм 1.

Алгоритм 1: $DecisionTreeRegressor(X, Y)$

параметры: F – множество признаков, msl – минимальное количество объектов в листе.

если $|X| \leq msl$ **тогда**
  вернуть $ЛИСТ(\frac{1}{|X|}\sum_{i=1}^{|X|} y_i)$;
конец
$f, v, I = CAT(X, Y)$ // (Choose Attribute and Theshold);
$X_{\leq}, Y_{\leq} = \{x \in X, y \in Y \mid f(x) \leq v\}$;
$X_{>}, Y_{>} = \{x \in X, y \in Y \mid f(x) > v\}$;
$S_{\leq} = DecisionTreeRegressor(X_{\leq}, Y_{\leq}, F)$;
$S_{>} = DecisionTreeRegressor(X_{>}, Y_{>}, F)$;
вернуть $УЗЕЛ(f, \{S_{\leq}, S_{>}\})$;

2 Методы

2.1 CAT-k

Ключевым моментом построения решающего дерева является выбор признака и соответствующего порога, которые задают максимально информативное ветвление узла. Этот шаг описывается в методе CAT(X, Y). Для этого необходимо задать критерий информативности, на основе которого осуществляется разбиение выборки.

Рассмотрим определенный узел дерева t. Разделим обучающую выборку X на две части по признаку $f \in F$ с порогом $v \in V_f$, где V_f – множество значений признака f. Получаем множества $X_{\leq}$ и $X_{>}$. Тогда критерий информативности $Q(X, Y, f, v)$ записывается в следующем виде:

$$Q(X, Y, f, v) = I(X, Y) - \frac{|X_{\leq}|}{|X|} \cdot I(X_{\leq}, Y_{\leq}) - \frac{|X_{>}|}{|X|} \cdot I(X_{>}, Y_{>}), \tag{7}$$

где $I(X, Y)$ – функция информативности, которая оценивает разброс ответов в узле дерева. Основной задачей является поиск параметров f и v, при которых достигается максимум функционала $Q(X, Y, f, v)$.

Как видно из формулы 7, критерий определятся через функцию информативности $I(X, Y)$. Попробуем обобщить его для заглядывающих вперед алгоритмов. Определим новую функцию, которая представляет следующую взвешенную сумму:

$$I_1(X, Y, f, v) = \frac{|X_{\leq}|}{|X|} \cdot I(X_{\leq}, Y_{\leq}) + \frac{|X_{>}|}{|X|} \cdot I(X_{>}, Y_{>}). \tag{8}$$

Тогда критерий информативности можно переписать в следующем виде:

$$Q_1(X, Y, f, v) = I(X, Y) - I_1(X, Y, f, v). \tag{9}$$

Индекс «1» обозначает, что критерий был посчитан с учетом узлов на одном уровне ниже текущего, то есть дерево решений заглядывает на один шаг вперед. Тогда мы можем рекурсивно обобщить данный критерий для деревьев заглядывающих на k шагов вперед:

$$I_k(X, Y, f, v) = \frac{|X_{\leq}|}{|X|} \min_{g \in F, u \in V_g} [I_{k-1}(X_{\leq}, Y_{\leq}, g, u)] + \frac{|X_{>}|}{|X|} \min_{g \in F, u \in V_g} [I_{k-1}(X_{>}, Y_{>}, g, u)]. \tag{10}$$

$$Q_k(X, Y, f, v) = I(X, Y) - I_k(X, Y, f, v). \tag{11}$$

Псевдокод алгоритма выбора признака и порога с описанным критерием информативности показан в Алгоритм 2. Функция $CAT_k(X, Y)$ помимо признака и порога возвращает оптимальное значения информативности. Это сделано для удобства реализации метода.

Алгоритм 2: CAT-k(X, Y)

параметры: F – множество признаков.

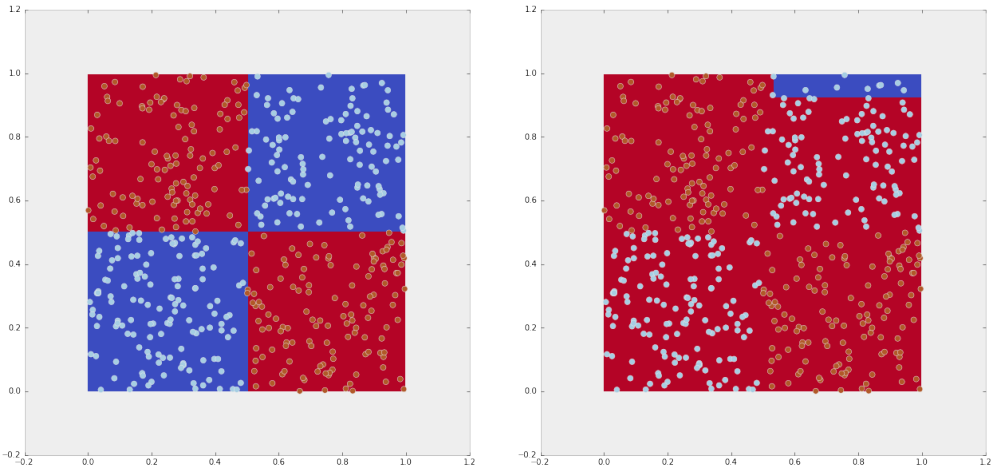
если $k{=}0$ **тогда**
  вернуть $None, None, I(X, Y)$;
конец
$f_{opt} = None$, $v_{opt} = None$, $Q_{opt} = -\inf$;
для каждого $f \in F$ **выполнять**
  $V_f = множество\ значений(f)$;
  для каждого $v \in V_f$ **выполнять**
    $X_{\leq}, Y_{\leq} = \{x \in X, y \in Y \mid f(x) \leq v\}$;
    $X_{>}, Y_{>} = \{x \in X, y \in Y \mid f(x) \geq v\}$;
    $f_{\leq}, v_{\leq}, I_{\leq}$ =CAT-k-1$(X_{\leq}, Y_{\leq})$;
    $f_{>}, v_{>}, I_{>}$ =CAT-k-1$(X_{>}, Y_{>})$;
    $I = \frac{|X_{\leq}|}{|X|} \cdot I_{\leq} + \frac{|X_{>}|}{|X|} \cdot I_{>}$;
    если $Q_{opt} < I(X, Y) - I$ **тогда**
      $f_{opt} = f$, $v_{opt} = v$, $I_{opt} = I$;
    конец
  конец
конец
вернуть $f_{opt}, v_{opt}, I_{opt}$;

Рассмотрим сложность алгоритма CAT-k. Пусть количество объектов равно $n = |X|$, а признаков $d = |F|$. Тогда сложность выбора признака и порога $T(\text{CAT-k}) = O(n \cdot d) \cdot T(\text{CAT-k-1})$ и $T(\text{CAT-0}) = O(n)$. Получаем, что асимптотическая сложность CAT-k равна $O(n \cdot (n \cdot d)^k)$.

В работе [2] рассмотрен алгоритм LSID3. Данный метод использует энтропийный критерий информативности, который обобщен для заглядывающих на k шагов

вперед решающих деревьев. Работоспособность алгоритма проверим на задаче про XOR – выборку. Пусть у двумерной выборки имеются два класса, где целевая зависимость имеет следующий вид: $y(x_1, x_2) = [(x_1 - 0.5)(x_2 - 0.5) > 0]$.

Заглядывающий вперед дерево с параметром $k = 2$ строит «идеальное» решение. Жадный алгоритм не сможет построить такое дерево, так как правильное ветвление в корневой вершине не увеличивает информативность, следовательно, алгоритм ID3 никогда не выберет его, и весь дальнейший процесс построения дерева пойдёт не оптимальным образом. Результат показан на Рис. 1.



a) lookahead дерево b) жадное дерево;

Рис. 1: Результаты алгоритмов на XOR – выборке

2.2 SCAT-k (Stochastic Choose an Attribute and Threshold)

Время выполнения рассмотренного метода CAT-k сильно зависит от параметра k. Несмотря на способности использовать дополнительные вычислительные ресурсы, применимость данного метода является проблематичной.

Во-первых, сложность работы CAT-k экспоненциально растет с ростом k, ограничивая гибкость алгоритма. Следующая проблема этого подхода заключается в том, что в некоторых ситуациях заглядывания вперед на глубину k недостаточно, чтобы определить полезность признака в узле дерева. Рассмотрим задачу n-XOR: $y(x_1, \ldots, x_n) = x_1 \oplus \cdots \oplus x_n$, чтобы проиллюстрировать эту проблему. ID3-k с $k < n$ не сможет определить полезность признаков $x_1, ., , , x_n$. Тем не менее, дерево заглядывающее вперед на глубину n найдет оптимальное решение.

Алгоритм 3: SCAT-k(X, Y)

параметры: F – множество признаков, s – доля порогов на каждом шаге.

если $k{=}0$ **тогда**
  вернуть $None, None, I(X, Y)$;
конец
$f_{opt} = None$, $v_{opt} = None$, $Q_{opt} = -\inf$;
для каждого $f \in F$ **выполнять**
  $V_f =$ *множество значений*(f);
  $n_s = |V_f| \cdot s$;
  $V_f^s =$ *выбрать случайно* n_s *элементов*(V);
  для каждого $v \in V^s$ **выполнять**
    $X_{\leq}, Y_{\leq} = \{x \in X, y \in Y \mid f(x) \leq v\}$;
    $X_{>}, Y_{>} = \{x \in X, y \in Y \mid f(x) \geq v\}$;
    $f_{\leq}, v_{\leq}, I_{\leq}$ =CAT-k-1$(X_{\leq}, Y_{\leq})$;
    $f_{>}, v_{>}, I_{>}$ =CAT-k-1$(X_{>}, Y_{>})$;
    $I = \frac{|X_{\leq}|}{|X|} \cdot I_{\leq} + \frac{|X_{>}|}{|X|} \cdot I_{>}$;
    если $Q_{opt} < I(X, Y) - I$ **тогда**
      $f_{opt} = f$, $v_{opt} = v$, $I_{opt} = I$;
    конец
  конец
конец
вернуть $f_{opt}, v_{opt}, I_{opt}$;

Рассмотрим методы стохастически понижающие сложность алгоритма CAT-k. В работе [2] предложен метод LSID3 (Lookahead by Stochastic ID3), который является модификацией алгоритма ID3-k. Обобщим его для произвольного критерия информативности.

Пусть $V_f = \{v_1, \ldots, v_n\}$ множество всевозможных значений признака f. В CAT-k мы пробегаемся по всем этим значениям, что требует огромных временных усилий, так как значение данного признака для всех объектов выборки может быть разным, а в методе SCAT-k будем выбирать случайное подмножество V_f^s множества V_f, количе-

ство объектов которого равно $n_s = s \cdot |V_f|$, где $s \leq 1$. Тогда функция информативности 11 меняется следующим образом:

$$I_k(X, Y, f, v) = \frac{|X_{\leq}|}{|X|} \min_{g \in F, u \in V_g^s} [I_{k-1}(X_{\leq}, Y_{\leq}, g, u)] + \frac{|X_{>}|}{|X|} \min_{g \in F, u \in V_g^s} [I_{k-1}(X_{>}, Y_{>}, g, u)]. \tag{12}$$

Асимптотическая сложность алгоритма уменьшится до $O(n \cdot (n_s \cdot d)^k)$. Псевдокод алгоритма показан в Алгоритм 3.

2.3 ECAT-k (Extremely Randomized Choose an Attribute and Threshold)

Попробуем использовать стохастический метод, который применяется в алгоритме ExtraTrees. В CAT-k для разделения выборки на две части в каждом узле дерева выбираются признак и значение этого признака, осуществляя полный перебор с двойным циклом.

В алгоритме ECAT-k на каждом шаге случайным образом будем генерировать случайное подмножества U^p множества всевозможных пар <признак, значение> $U = \{(f, v) | \forall f \in F, \forall v \in V_f\}$ с количеством элементов $l_p = p \cdot |U|$, где $p \leq 1$. Таким образом, двойной цикл по всем признакам и их значениям заменим на один цикл по парам <признак, значение>. Функция информативности:

$$I_k(X, Y, f, v) = \frac{|X_{\leq}|}{|X|} \min_{(g,u) \in U^p} [I_{k-1}(X_{\leq}, Y_{\leq}, g, u)] + \tag{13}$$

$$+ \frac{|X_{>}|}{|X|} \min_{(g,u) \in U^p} [I_{k-1}(X_{>}, Y_{>}, g, u)]. \tag{14}$$

Асимптотическая сложность алгоритма уменьшится до $O(n \cdot (l_p)^k))$.

2.4 Оценка числа операций

Было показано, что асимптотическая сложность методов CAT - k, SCAT - k и ECAT - k растет экспоненциально с увеличением глубины k. Однако, при вычислении данной сложности не учитывалась информация о убывании количества объектов после каждого дробления. В данной секции будет выведена более точная оценка вычислительной сложности каждого из перечисленных методов.

Алгоритм 4: ECAT-k(X, Y)

параметры: F – множество признаков, p – доля пар <признак, значение> каждом шаге.

если $k{=}0$ **тогда**
  вернуть $None, None, I(X, Y)$;
конец
$f_{opt} = None$, $v_{opt} = None$, $Q_{opt} = -\inf$;
$l_p = p \cdot n \cdot d$;
$U = \{(f, v) | \forall f \in F, \forall v \in V_f,$ *где* $V_f =$ *множество значений*$(f)\}$;
$U_p =$ *выбрать случайно* l_p *элементов*(U);
для каждого $(f, v) \in U^p$ **выполнять**
  $X_{\leq}, Y_{\leq} = \{x \in X, y \in Y \mid f(x) \leq v\}$;
  $X_{>}, Y_{>} = \{x \in X, y \in Y \mid f(x) \geq v\}$;
  $f_{\leq}, v_{\leq}, I_{\leq}$ =CAT-k-1$(X_{\leq}, Y_{\leq})$;
  $f_{>}, v_{>}, I_{>}$ =CAT-k-1$(X_{>}, Y_{>})$;
  $I = \frac{|X_{\leq}|}{|X|} \cdot I_{\leq} + \frac{|X_{>}|}{|X|} \cdot I_{>}$;
  если $Q_{opt} < I(X, Y) - I$ **тогда**
    $f_{opt} = f$, $v_{opt} = v$, $I_{opt} = I$;
  конец
конец
вернуть $f_{opt}, v_{opt}, I_{opt}$;

Рассмотрим определенный узел дерева t с объектами обучающей выборки $X = (x_1, ..., x_n)$ и множеством признаков $F = (f_1, ..., f_d)$. Будем предполагать, что каждый признак $f \in F$ имеет вещественное множество значений V_f, уникальных для каждого объекта $x \in X$. Пусть $X_{\leq}$ и $X_{>}$ – множества, которые получаются при разделении X по признаку f с порогом $v \in V_f$.

CAT-k. Аппроксимацию количества арифметических действий метода CAT-k, выполняемых в узле t обозначим через $C_k(n,d)$. Тогда из алгоритма 2 имеем:

$$C_k(n,d) = \sum_{\forall f \in F} \sum_{\forall v \in V_f} (C_{k-1}(|X_{\leq}|,d) + C_{k-1}(|X_{>}|,d)) = \tag{15}$$

$$= d \cdot \sum_{i_k=1}^{n} (C_{k-1}(i_k,d) + C_{k-1}(n-i_k,d)) = 2 \cdot d \sum_{i_k=1}^{n} C_{k-1}(i_k,d), \tag{16}$$

Соотношение 16 позволяет рекурентно выразить $C_k(n,d)$:

$$C_k(n,d) = 2 \cdot d \sum_{i_k=1}^{n} C_{k-1}(i_k,d) = 2 \cdot d \sum_{i_k=1}^{n} \left[2 \cdot d \sum_{i_{k-1}=1}^{i_k} C_{k-2}(i_{k-1},d) \right] = \tag{17}$$

$$= 4 \cdot d^2 \sum_{i_k=1}^{n} \sum_{i_{k-1}=1}^{i_k} C_{k-2}(i_{k-1},d) = 2^k \cdot d^k \sum_{i_k=1}^{n} \sum_{i_{k-1}=1}^{i_k} \cdots \sum_{i_2=1}^{i_3} \sum_{i_1=1}^{i_2} C_0(i_1,d). \tag{18}$$

При $k = 0$ имеем лист дерева, где усуществляется i_1 операций для подсчета функции информативности. Тогда, положив условие $C_0(i_1,d) = i_1$ в 18, получаем:

$$C_k(n,d) = 2^k \cdot d^k \sum_{i_k=1}^{n} \sum_{i_{k-1}=1}^{i_k} \cdots \sum_{i_2=1}^{i_3} \sum_{i_1=1}^{i_2} i_1. \tag{19}$$

Для вычисления суммы 19 воспользуемся следующей формулой, найденная Якобом Бернулли [1]:

$$\sum_{i=1}^{n} i^m = \frac{1}{m+1} \sum_{j=0}^{m} C_{m+1}^{j} B_j n^{m+1-j}, \tag{20}$$

где $B_0, \cdots, B_m$ – числа Бернулли. Учитывая, что $B_0 = 1$ найдем аппроксимацию формулы 20:

$$\sum_{i=1}^{n} i^m \approx \frac{n^{m+1}}{m+1}. \tag{21}$$

Найденная аппроксимация позволяет оценить 19:

$$C_k(n,d) \approx 2^k \cdot d^k \sum_{i_k=1}^{n} \sum_{i_{k-1}=1}^{i_k} \cdots \sum_{i_2=1}^{i_3} \frac{i_2}{2} \approx \cdots \approx 2^k \cdot d^k \frac{n^{k+1}}{2 \cdot 3 \cdot \cdots \cdot (k+1)} \tag{22}$$

Упростив выражение 22, получаем аппроксимацию числа операций для CAT-k:

$$C_k(n,d) \approx \frac{n \cdot (2 \cdot d \cdot n)^k}{(k+1)!} \tag{23}$$

SCAT-k. Из алгоритма 3 известно, что в методе SCAT-k значение порога v выбирается из множества V_f^s, где $|V_f^s| = s \cdot |V_f|$. Тогда, обозначив количество операций метода SCAT-k через $S_k(n, d, s)$, имеем следующее:

$$S_k(n, d, s) = \sum_{\forall f \in F} \sum_{\forall v \in V_f^s} (S_{k-1}(|X_{\leq}|, d, s) + S_{k-1}(|X_{>}|, d, s)) = \tag{24}$$

$$= d \cdot \sum_{\forall v \in V_f^s} (S_{k-1}(|X_{\leq}|, d, s) + S_{k-1}(|X_{>}|, d, s)) \,. \tag{25}$$

Предположим, что v – случайная величина из равномерного распределения множества значений V_f. Тогда вероятность того, что порог v окажется в V_f^s равен s. Величину 25 можем оценить в среднем как следующее математическое ожидание:

$$S_k(n, d, s) = d \cdot \sum_{i_k=1}^{n} s \cdot (S_{k-1}(i_k, d, s) + S_{k-1}(n - i_k, d, s)) = \tag{26}$$

$$= d \cdot s \sum_{i_k=1}^{n} \cdot (S_{k-1}(i_k, d, s) + S_{k-1}(n - i_k, d, s)) = \tag{27}$$

$$= 2 \cdot d \cdot s \sum_{i_k=1}^{n} S_{k-1}(i_k, d, s). \tag{28}$$

Дальше, проведя аналогичные действия 17 - 23, получаем аппроксимацию числа операций для SCAT-k:

$$S_k(n, d, s) \approx \frac{n \cdot (2 \cdot d \cdot n \cdot s)^k}{(k+1)!}. \tag{29}$$

ECAT-k. Из алгоритма 4 известно, что в методе ECAT-k случайным образом выбирается множество U^p из множества всевозможных пар <признак, значение> U, где $|U^p| = |U| \cdot d$. Теперь, обозначив количество операций метода ECAT-k через $E_k(n, d, p)$, из алгоритма 4 имеем:

$$E_k(n, d, p) = \sum_{\forall (f,v) \in U^p} (E_{k-1}(|X_{\leq}|, d, p) + E_{k-1}(|X_{>}|, d, p)) \,. \tag{30}$$

Учитывая, что вероятность попадания пары (f, v) в U^p равен p, оценим 30 как следующее математическое ожидание:

$$E_k(n, d, p) = \sum_{\forall (f,v) \in U} p \cdot (E_{k-1}(|X_{\leq}|, d, p) + E_{k-1}(|X_{>}|, d, p)) = \tag{31}$$

$$= \sum_{\forall f \in F} \sum_{\forall v \in V} p \cdot (E_{k-1}(|X_{\leq}|, d, p) + E_{k-1}(|X_{>}|, d, p)) = \tag{32}$$

$$= d \cdot p \sum_{i_k=1}^{n} \cdot (E_{k-1}(i_k, d, p) + E_{k-1}(n - i_k, d, p)) = \tag{33}$$

$$= 2 \cdot d \cdot p \sum_{i_k=1}^{n} E_{k-1}(i_k, d, p). \tag{34}$$

Дальше как и в методе CAT-k получаем следующую аппроксимацию для ECAT-k:

$$E_k(n, d, p) \approx \frac{n \cdot (2 \cdot d \cdot n \cdot p)^k}{(k+1)!} \tag{35}$$

3 Вычислительные эксперименты

3.1 Исходные данные

Эксперименты были проведены на 12 датасетах из UCI Machine Learning Repository с задачами классификации и регрессии. Подробное описание датасетов можно увидеть в таблице 1.

	#объектов	#классов	#признаков
EEG Eye State	7000	2	14
Pima Indians diabetes	768	2	8
ILPD	579	2	11
Credit approval	653	2	46
Breast Cancer	284	2	30
Mammographic mass	831	2	14
Digits	1797	10	64
Car Eval	1728	4	21
Forest Type	523	4	27
Facebook metrics	495	регрессия	12
Wine quality	4898	регрессия	11
Concrete Compressive Strength	1030	регрессия	8

Таблица 1: Описание датасетов

Предварительная обработка каждого датасета включала следующие операции:

1. Категориальные признаки были преобразованы с помощью метода one-hot-encoding.

2. Вещественные признаки масштабированы так, что они имеют нулевое среднее и единичное стандартное отклонение.

3. Удалены признаки с пропущенные значениями.

3.2 Условия эксперимента

Эксперименты будут проводится над градиентным бустингом с различными методами подбора признака и порога в базовой модели.

Для избежания переобучения будем использовать бустинг над деревьями с глубиной 3. Максимальное количество базовых моделей в бустинге $T = 2000$ и скорость обучения $\gamma = 0.1$. Информативность разделения в узле дерева оцениваем через функцию MSE:

$$I(Y) = \frac{1}{n}\sum_{i=1}^{n}(y_i - \overline{y})^2, \text{ где } \overline{y} = \frac{1}{n}\sum_{i=1}^{n} y_i \tag{36}$$

Каждый датасет был разбит на три части, таким образом, что алгоритмы обучались на 50%, качество считалась на 25%, а на остальной части подбиралось оптимальное количество деревьев в бустинге: если в течении 200 итераций алгоритму не удалось уменьшить функционал ошибки, бустинг останавливается.

Для оценки качества задачи классификации использовалась точность:

$$accuracy(Y, \widetilde{Y}) = \frac{1}{n}\sum_{i=1}^{n}[y_i = \widetilde{y}_i], \text{ где } \widetilde{y}_i \text{– ответ бустинга для объекта } x_i, \tag{37}$$

а для регрессии MSE:

$$MSE(Y, \widetilde{Y}) = \frac{1}{n}\sum_{i=1}^{n}(y_i - \widetilde{y}_i)^2. \tag{38}$$

Методы SCAT-k и ECAT-k имеют влияние случайности при выборе множества порогов и признаков. Поэтому алгоритмы были запущены 3 раза и качество усреднялось по всем запускам.

Сравнение алгоритмов проводилось по суммарному баллу, который считался по следующему принципу:

1. Каждый датасет имеет 1 балл.

2. Балл за датасет получает тот алгоритм, который показал наивысшее качество на данном датасете. Если таких алгоритмов несколько, тогда балл делится между алгоритмами.

3.3 Эксперимент

Целью данного эксперимента является сравнение градиентного бустинга с методами подбора параметров SCAT-2 (GradBoostSCAT-2), ECAT-2 (GradBoostECAT-2) и CAT-1 (GradBoost). Следует отметить, что CAT-1 является жадным алгоритмом построения дерева, заглядывающий только на один шаг вперед. Именно этот метод используется в качестве базовой модели во многих реализациях бустинга.

	GradBoostSCAT-2	GradBoostECAT-2	GradBoost
EEG Eye State	0.932±0.001	0.961±0.001 •	0.951
Pima Indians diabetes	0.733±0.005 •	0.731±0.016	0.7248
ILPD	0.674±0.003	0.687±0.025 •	0.683
Credit approval	0.892±0.012 •	0.885±0.006	0.847
Breast	0.948±0.003	0.967±0.003 •	0.944
Mammographic mass	0.788±0.0	0.792±0.005 •	0.769
Digits	0.966±0.002 •	0.966±0.001 •	0.924
Car Eval	0.975±0.001	0.981±0.002 •	0.979
Forest Type	0.908±0.016	0.927±0.004 •	0.893
Facebook metrics	58526.1±825 •	74308.0±3886	88500.1
Wine quality	0.464±0.004	0.441±0.004 •	0.491
Concrete Compressive Strength	45.481±0.674	40.509±0.352 •	42.048
Суммарный балл	3.5	8.5 •	0

Таблица 2: Качество алгоритмов

Было показано, что для построения заглядывающих вперед методов требуется значительно больше времени по сравнению с жадным алгоритмом. Поэтому подберем такие значения параметров s и p, при которых все методы будут иметь одинаковую

сложность работы:

$$C_1(n,d) = S_2(n,d,s) = E_2(n,d,p) \tag{39}$$

$$n^2 \cdot d = \frac{n \cdot (2 \cdot d \cdot n \cdot s)^2}{3!} = \frac{n \cdot (2 \cdot d \cdot n \cdot p)^2}{3!} \tag{40}$$

$$n \cdot d = \frac{(2 \cdot d \cdot n \cdot s)^2}{6} = \frac{(2 \cdot d \cdot n \cdot p)^2}{6} \tag{41}$$

$$\sqrt{\frac{3}{2 \cdot n \cdot d}} = s = p \tag{42}$$

В итоге методы SCAT-2 и ECAT-2 будут запускаться с параметрами $s = \sqrt{\frac{3}{2 \cdot n \cdot d}}$ и $p = \sqrt{\frac{3}{2 \cdot n \cdot d}}$.

В таблице 2 приведены результаты качества алгоритмов. В таблице 3 показано среднее время работы обучения одного дерева в бустинге.

	GradBoostSCAT-2	GradBoostECAT-2	GradBoost
EEG Eye State	15.744	20.626	11.972
Pima Indians diabetes	0.388	0.496	0.446
ILPD	0.342	0.484	0.415
Credit approval	0.258	0.801	0.809
Breast	0.956	1.697	1.3
Mammographic mass	0.116	0.201	0.383
Digits	6.981	9.368	7.884
Car Eval	0.5011	0.585	0.986
Forest Type	0.406	0.781	0.504
Facebook metrics	0.11	0.179	0.31
Wine quality	6.789	11.809	6.052
Concrete Compressive Strength	0.364	0.486	0.472

Таблица 3: Среднее время (в секундах) обучения одной модели в бустинге

4 Заключение

Заглядывающие вперед алгоритмы построения решающих деревьев представляет собой достаточно сильную модель. В данной работе метод был модифицирован специально для алгоритма градиентного бустинга стохастических путем:

1. Были предложены оптимизации lookahead деревьев SCAT-k и ECAT-k.

2. Данные алгоритмы были реализованы на языке **python** и представлены в открытый доступ.

3. Алгоритмы градиентного бустинга над решающими деревьями с выбором параметров SCAT-2 (GradBoostSCAT-2), ECAT-2 (GradBoostECAT-2) и классический градиентный бустинг были сравнены на реальных задачах при условии, что методы используют в среднем одинаковое количество вычислительных операций. По результатом эксперимента на большинстве реальных задач наилучшие результаты показал модифицированный градиентный бустинг с GradBoostECAT-2.

.

Список литературы

[1] Laura Elizabeth S Coen. Sums of powers and the bernoulli numbers. Master's thesis, Eastern Illinois University, 1996.

[2] Saher Esmeir and Shaul Markovitch. Lookahead-based algorithms for anytime induction of decision trees. In *Proceedings of the Twenty-first International Conference on Machine Learning*, ICML '04, pages 33–, New York, NY, USA, 2004. ACM.

[3] Saher Esmeir and Shaul Markovitch. Anytime learning of decision trees. *Journal of Machine Learning Research*, 8:891–933, May 2007.

[4] Jerome H. Friedman. Greedy function approximation: A gradient boosting machine. *Annals of Statistics*, 29:1189–1232, 2000.

[5] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning*. Springer Series in Statistics. Springer New York Inc., New York, NY, USA, 2001.

[6] Mark Last and Michael Roizman. Avoiding the look-ahead pathology of decision tree learning. *Int. J. Intell. Syst.*, 28(10):974–987, 2013.

[7] Sreerama Murthy and Steven Salzberg. Lookahead and pathology in decision tree induction. In *Proceedings of the 14th International Joint Conference on Artificial Intelligence - Volume 2*, IJCAI'95, pages 1025–1031, San Francisco, CA, USA, 1995. Morgan Kaufmann Publishers Inc.

[8] J. R. Quinlan. Induction of decision trees. *Mach. Learn.*, 1(1):81–106, March 1986.

[9] J.A. Sonquist, E.L. Baker, and J.N. Morgan. *Searching for structure: an approach to analysis of substantial bodies of micro-data and documentation for a computer program*. Survey Research Center, University of Michigan, 1974.