ЦИФРОВАЯ КУЛЬТУРА
УНИВЕРСИТЕТ ИТМО

Машинное обучение

Лекция 6. Решающие деревья

Автор: Рустам Азимов

Санкт-Петербург, 2023г.

Линейные методы

- Мы рассматривали линейные методы, которые обладают рядом важных достоинств
  - быстро обучаются
  - способны работать с большим количеством объектов и признаков
  - имеют небольшое количество параметров
  - легко регуляризуются
- При этом у них есть и серьёзный недостаток — они могут восстанавливать только линейные зависимости
- Конечно, можно добавлять в выборку новые признаки, которые нелинейно зависят от исходных, но
  - этот подход является чисто эвристическим
  - требует выбора типа нелинейности
  - всё равно ограничивает сложность модели сложностью признаков

Decision Trees

- Теперь рассмотрим **решающие деревья (decision trees)** — семейство моделей, которые позволяют восстанавливать нелинейные зависимости произвольной сложности
- Решающие деревья хорошо описывают процесс принятия решения во многих ситуациях, например при выдачи кредита в банке
  1. Какой возраст у клиента? Если меньше 18, то отказываем в кредите, иначе продолжаем.
  2. Какая зарплата у клиента? Если больше 50 тысяч рублей, то переходим к шагу 3, иначе к шагу 4.
  3. Какой стаж у клиента? Если меньше 5 лет, то не выдаем кредит, иначе выдаем.
  4. Есть ли у клиента другие кредиты? Если есть, то отказываем, иначе выдаем.

Решающие деревья

- Начиная с 60-х годов им уделяется большое внимание
- Они обладают высокими интерпретируемостью и выразительной способностью
- Но деревья крайне трудны для оптимизации из-за свой дискретной структуры — дерево нельзя продифференцировать по параметрам и найти с помощью градиентного спуска хотя бы локальный оптимум
- Даже число параметров у них не является постоянным и может меняться в зависимости от глубины, выбора критериев дробления и прочих деталей

Решающие деревья

- Из-за этого все методы построения решающих деревьев являются жадными и эвристичными
- На сегодняшний день решающие деревья не очень часто используются как отдельные методы классификации или регрессии
- В то же время, как оказалось, они очень хорошо объединяются в композиции — решающие леса, которые являются одними из наиболее сильных и универсальных моделей

Решающие деревья: определение

- Рассмотрим бинарное дерево, в котором:
  - Каждой внутренней вершине v приписана функция (или предикат) $\beta_v : \mathbb{X} \to \{0, 1\}$
  - Каждой листовой вершине v приписан прогноз $c_v \in \mathbb{Y}$ (в случае с классификацией листу также может быть приписан вектор вероятностей)
- Алгоритм $a(x)$ стартует из корневой вершины v_0 и вычисляет значение функции β_{v_0}
- Если оно равно нулю, то алгоритм переходит в левую дочернюю вершину, иначе в правую
- Процесс продолжается, пока не будет достигнута листовая вершина; алгоритм возвращает тот класс, который приписан этой вершине
- Такой алгоритм называется **бинарным решающим деревом**

Решающие деревья

- На практике в большинстве случаев используются одномерные предикаты β_v, которые сравнивают значение одного из признаков с порогом:

$$\beta_v(x; j, t) = [x_j < t]$$

- Существуют и многомерные предикаты, например:
  - Линейные $\beta_v(x) = [\langle w, x \rangle < t]$
  - Метрические $\beta_v(x) = [\rho(x, x_v) < t]$, где точка x_v является одним из объектов выборки или любой точкой пространства признаков
- Многомерные предикаты позволяют строить ещё более сложные разделяющие поверхности, но очень редко используются на практике
- Например, из-за того, что усиливают и без того выдающиеся способности деревьев к переобучению
- Поэтому далее мы будем говорить только об одномерных предикатах

Построение деревьев

- Легко убедиться, что для любой выборки можно построить решающее дерево, не допускающее на ней ни одной ошибки
- Но тривиальное дерево, в каждом листе которого находится ровно по одному объекту выборки, будет переобученным
- Можно поставить задачу поиска дерева, которое является минимальным (с точки зрения количества листьев) среди всех деревьев, не допускающих ошибок на обучении и надеяться на наличие у дерева обобщающей способности
- К сожалению, эта задача является NP-полной, и поэтому приходится ограничиваться жадными алгоритмами построения дерева

Базовый жадный алгоритм построения бинарного решающего дерева

- Начнем со всей обучающей выборки X и найдем наилучшее ее разбиение на две части $R_1(j,t) = \{x \mid x_j < t\}$ и $R_2(j,t) = \{x \mid x_j \geq t\}$ с точки зрения заранее заданного функционала качества $Q(X,j,t)$
- Найдя наилучшие значения j и t, создадим корневую вершину дерева, поставив ей в соответствие предикат $[x_j < t]$
- Объекты разобьются на две части — одни попадут в левое поддерево, другие в правое
- Для каждой из этих подвыборок рекурсивно повторим процедуру, построив дочерние вершины для корневой, и так далее
- В каждой вершине мы проверяем, не выполнилось ли некоторое условие останова — и если выполнилось, то прекращаем рекурсию и объявляем эту вершину листом

Базовый жадный алгоритм построения бинарного решающего дерева

- Когда дерево построено, каждому листу ставится в соответствие ответ
- В случае с классификацией это может быть класс, к которому относится больше всего объектов в листе, или вектор вероятностей (скажем, вероятность класса может быть равна доле его объектов в листе)
- Для регрессии это может быть среднее значение, медиана или другая функция от целевых переменных объектов в листе
- Выбор конкретной функции зависит от функционала качества в исходной задаче

Базовый жадный алгоритм построения бинарного решающего дерева

- Решающие деревья могут обрабатывать пропущенные значения — ситуации, в которых для некоторых объектов неизвестны значения одного или нескольких признаков
- Для этого необходимо модифицировать процедуру разбиения выборки в вершине, что можно сделать несколькими способами
- После того, как дерево построено, можно провести его стрижку (pruning) — удаление некоторых вершин с целью понижения сложности и повышения обобщающей способности

Критерии информативности

- При построении дерева необходимо задать функционал качества, на основе которого осуществляется разбиение выборки на каждом шаге
- Обозначим через R_m множество объектов, попавших в вершину, разбиваемую на данном шаге, а через R_l и R_r — объекты, попадающие в левое и правое поддерево соответственно при заданном предикате
- Мы будем использовать функционалы следующего вида:

$$Q(R_m, j, s) = H(R_m) - \frac{|R_l|}{|R_m|} H(R_l) - \frac{|R_r|}{|R_m|} H(R_r)$$

Критерии информативности

- Здесь $H(R)$ — это критерий информативности (impurity criterion), который оценивает качество распределения целевой переменной среди объектов множества R
- Чем меньше разнообразие целевой переменной, тем меньше должно быть значение критерия информативности — и, соответственно, мы будем пытаться минимизировать его значение
- Функционал качества $Q(R_m, j, t)$ мы при этом будем максимизировать

Критерии информативности

- Как уже обсуждалось, в каждом листе дерево будет выдавать константу — вещественное число, вероятность или класс
- Исходя из этого, можно предложить оценивать качество множества объектов R тем, насколько хорошо их целевые переменные предсказываются константой (при оптимальном выборе этой константы):

$$H(R) = \min_{c \in \mathbb{Y}} \frac{1}{|R|} \sum_{(x_i, y_i) \in R} L(y_i, c)$$

- Где $L(y, c)$ — некоторая функция потерь

Критерии информативности: регрессия

- Как обычно, в регрессии выберем квадрат отклонения в качестве функции потерь

$$H(R) = \min_{c \in \mathbb{Y}} \frac{1}{|R|} \sum_{(x_i, y_i) \in R} (y_i - c)^2$$

- Как известно, минимум в этом выражении будет достигаться на среднем значении целевой переменной

$$H(R) = \frac{1}{|R|} \sum_{(x_i, y_i) \in R} (y_i - \frac{1}{|R|} \sum_{(x_j, y_j) \in R} y_j)^2$$

- Мы получили, что информативность вершины измеряется её дисперсией — чем ниже разброс целевой переменной, тем лучше вершина
- Можно использовать другие функции ошибки L — например, при выборе абсолютного отклонения мы получим в качестве критерия среднее абсолютное отклонение от медианы

Критерии информативности: классификация

- Рассмотрим индикатор ошибки как функцию потерь

$$H(R) = \min_{c \in \mathbb{Y}} \frac{1}{|R|} \sum_{(x_i, y_i) \in R} [y_i \neq c]$$

- Легко видеть, что оптимальным предсказанием тут будет наиболее популярный класс k^* — значит, критерий будет равен следующей доле ошибок

$$H(R) = \frac{1}{|R|} \sum_{(x_i, y_i) \in R} [y_i \neq k^*] = 1 - p_{k^*}$$

- Данный критерий является достаточно грубым, поскольку учитывает частоту p_{k^*} лишь одного класса

Критерии информативности: классификация

- Рассмотрим ситуацию, в которой мы выдаём в вершине не один класс, а распределение на всех классах $c = (c_1, \ldots, c_K)$

$$H(R) = \min_{\sum_k c_k = 1} \frac{1}{|R|} \sum_{(x_i, y_i) \in R} \sum_{k=1}^{K} (c_k - [y_i = k])^2$$

- Оптимальный вектор вероятностей состоит из долей классов $c_* = (p_1, \ldots, p_K)$
- Отсюда получается критерий Джини:

$$H(R) = \sum_{k=1}^{K} p_k (1 - p_k)$$

Критерии останова

- Можно придумать большое количестве критериев останова
- Перечислим некоторые ограничения и критерии:
  - Ограничение максимальной глубины дерева
  - Ограничение минимального числа объектов в листе
  - Ограничение максимального количества листьев в дереве
  - Останов в случае, если все объекты в листе относятся к одному классу
  - Требование, что функционал качества при дроблении улучшался как минимум на s процентов
- С помощью грамотного выбора подобных критериев и их параметров можно существенно повлиять на качество дерева
- Тем не менее, такой подбор является трудозатратным и требует проведения кросс-валидации

Методы стрижки дерева

- Стрижка дерева является альтернативой критериям останова
- При использовании стрижки сначала строится переобученное дерево (например, до тех пор, пока в каждом листе не окажется по одному объекту), а затем производится оптимизация его структуры с целью улучшения обобщающей способности
- Существует ряд исследований, показывающих, что стрижка позволяет достичь лучшего качества по сравнению с ранним остановом построения дерева
- Но на данный момент методы стрижки редко используются и не реализованы в большинстве библиотек для анализа данных
- Причина заключается в том, что деревья сами по себе являются слабыми алгоритмами и не представляют интереса, а при использовании в композициях они либо должны быть переобучены (в случайных лесах), либо должны иметь очень небольшую глубину (в бустинге) и стрижка не нужна

Методы стрижки дерева

- Одним из методов стрижки является cost-complexity pruning
- Обозначим дерево, полученное в результате работы жадного алгоритма, через T_0
- Как и при регуляризации боремся с переобучением вводом штрафа за размер дерева (здесь |T| — число листьев)

$$R_\alpha(T) = R(T) + \alpha|T|$$

- Можно показать, что существует последовательность вложенных деревьев с одинаковыми корнями далее из нее выбирается оптимальное дерево с помощью кросс-валидации:

$$T_K \subset T_{K-1} \subset \ldots \subset T_0$$

- Здесь $0 = \alpha_0 < \alpha_1 < \ldots < \alpha_K < \infty$

Дополнительные источники

- https://habr.com/ru/company/ods/blog/322534/
- machinelearning.ru
- scikit-learn.org
- kaggle