Деревья решений и их применение. Алгоритм CART

1.Введение

1.1.Определение понятия дерева решений

Деревья решений это семейство алгоритмов, которое очень сильно отличается от линейных моделей, но в то же время играет важную роль в машинном обучении. Классические алгоритмы на деревья решений были известны сравнительно давно, а некоторые их вариации (например, случайный лес) вообще являются очень мощным инструментом в своей области. Первые идеи создания деревьев решений восходят к работам Ховленда (Hoveland) и Ханта(Hunt) конца 50-х годов XX века. Однако, основополагающей работой, давшей импульс для развития этого направления, явилась книга Ханта (Hunt, E.B.), Мэрина (Marin J.) и Стоуна (Stone, P.J) "Experiments in Induction", увидевшая свет в 1966 г. Помимо этих работ, известны также более поздние статьи, например “Classification and Regression Trees” (Leo Breiman, Jerome Friedman, Charles J. Stone, R.A. Olshen (1984)), “Induction of Decision Trees” (Quinlan (1986)).

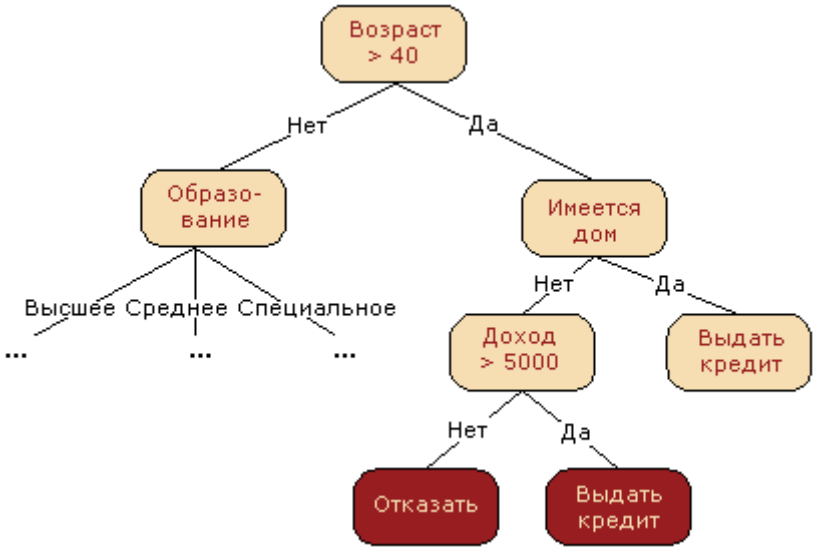
1.2. Базовое определение алгоритма

В машинном обучении дерево решений представляет модель, которая, как ни удивительно, имеет структуру дерева. Оно может быть как бинарным (когда все вершины, кроме листьев, делятся на не более чем 2 поддерева), так и N-арным (когда максимальная степень разбиения промежуточных вершин равна N).

Теперь представим, что в каждой вершине заключено некое разделяющее условие. Если мы подадим на вершину дерева исходные данные, а затем будем идти по вершинам в зависимости от значения фильтрующего условия, то в лист мы сможем прийти с некоторой подвыборкой. Таким образом, если мы построим дерево с некоторыми условиями в каждой вершине (например, дающие ответ да/нет), а в листьях заключим ответы, то мы можем давать с помощью обхода дерева из вершины в листья выдать по исходной

выборке заготовленный ответ. Это базовый принцип работы алгоритма дерева решений.

Для примера можно рассмотреть следующую диаграмму, в которой определяется возможность выдачи кредитов заявленному лицу:



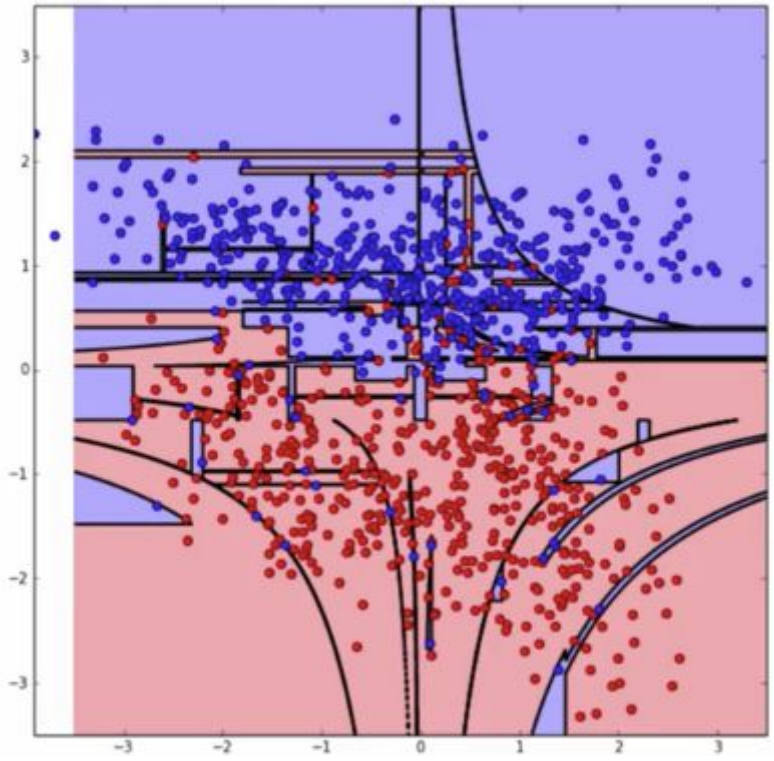
Говоря более формально, можно дать следующее определение условию в вершине:

$$[x_j \leq t]$$

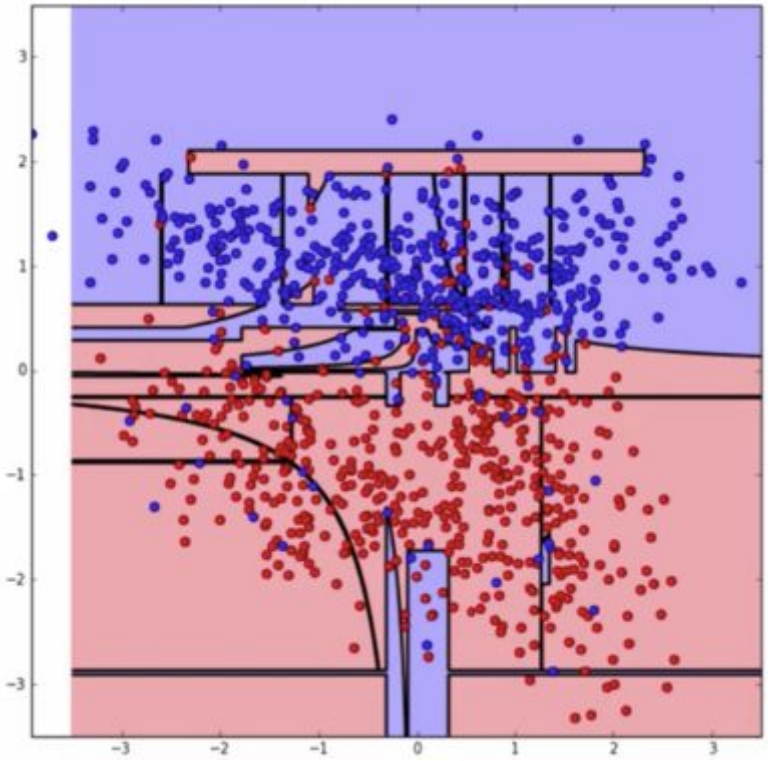
Здесь x_j - значение `j` признака, `t` - заданный порог. Условия выбираются очень простыми, но даже несмотря на это, оно помогает решать многие сложные задачи. Стоит отметить, что значения в листьях могу быть как вещественным числом (если решается задача регрессии), так и класс или распределение вероятностей классов (если решается задача классификации).

Рассмотрим немного примеров.

1. Задача классификации. Исходя из определения дерева, плоскость с признаками с помощью вершин-условий отсекается некоторыми кусками. На этом рисунке модель скорее переобучилась, поскольку много встречается маленьких областей сложной формы. Это может образоваться вследствие нескольких признаков модели, о которых речь пойдет дальше.

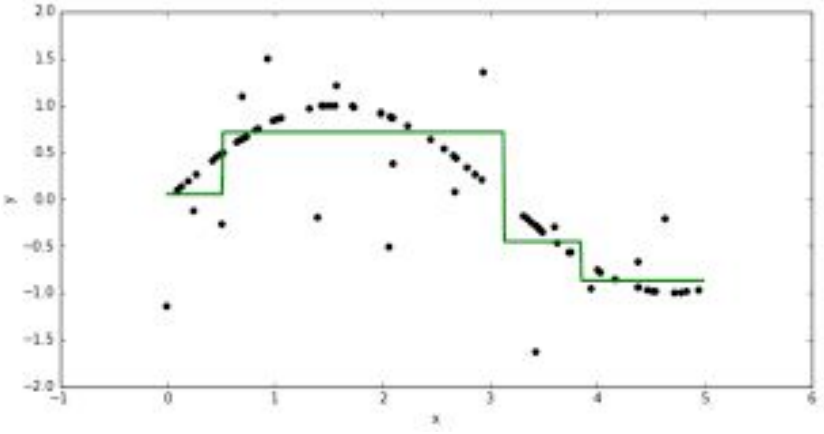


В то время как на следующем рисунке таких областей значительно меньше.



2. Задача регрессии. Пусть решается задача регрессии с одним признаком, по которому можно восстановить значение

целевой переменной. Тогда не очень глубокое дерево восстанавливает зависимость примерно так:



Восстановленная зависимость будет кусочно-постоянной, но в целом будет иметь неплохое качество.

2. Обучение

2.1. Жадный способ построения

В машинном обучении применяется жадный способ построения решающего дерева от корня к листьям. Сначала выбирается корень, который разбивает выборку на две. Затем разбивается каждый из потомков этого корня и так далее. Дерево ветвится до тех пор, пока этого не будет достаточно.

Остается уточнить способ разбиения каждого потомка. Как было сказано ранее, в качестве условия в каждой вершине строящегося дерева будет использоваться простейшее условие: значение одного из признаков будет сравниваться с некоторым порогом.

Пусть в вершину m попало множество X_m объектов из обучающей выборки. Параметры в условии $[x_j \leq t]$ будут выбраны так, чтобы минимизировать данный критерий ошибки $Q(X_m, j, t)$, зависящий от этих параметров:

$$Q(X_m, j, t) \rightarrow min$$

Параметры `j` и `t` можно подбирать перебором. Действительно, признаков конечное число, а из всех возможных значений порога `t` можно рассматривать только те, при которых получаются различные разбиения. Можно показать, что таких

значений параметра t столько, сколько различных значений признака x_j на обучающей выборке.

После того, как параметры были выбраны, множество X_m объектов из обучающей выборки разбивается на два множества

$$X_l = \{x \in X_m \mid [x_j \leq t]\},\ X_r = \{x \in X_m \mid [x_j > t\ ]\}$$

каждое из которых соответствует своей дочерней вершине.

Предложенную процедуру можно продолжить для каждой из дочерних вершин: в этом случае дерево будет все больше и больше углубляться. Такой процесс рано или поздно должен остановиться, и очередная дочерняя вершина будет объявлена листком, а не разделена пополам. Этот момент определяется критерием остановки. Существует много различных вариантов критерия остановки, среди которых самые очевидные:

- Если в вершину попал только один объект обучающей выборки или все объекты принадлежат одному классу (в задачах классификации), дальше разбивать не имеет смысла.
- Можно также останавливать разбиение, если глубина дерева достигла определенного значения.

Если какая-то вершина не была поделена, а была объявлена листом, нужно определить прогноз, который будет содержаться в данном листе. В этот лист попала некоторая подвыборка X_m исходной обучающей выборки и требуется выбрать такой прогноз, который будет оптимален для данной подвыборки.

В задаче регрессии, если функционал — среднеквадратичная ошибка, оптимально давать средний ответ по этой подвыборке:

$$a_m = \frac{1}{|X_m|} \sum_{i \in X_m} y_i$$

В задаче классификации оптимально возвращать тот класс, который наиболее популярен среди объектов в Xm:

$$a_m = argmax_{y \in Y} \sum_{i \in X_m} [y_i = y]$$

Кроме того, можно указать и вероятности классов как долю их объектов в Xm:

$$a_{mk} = \frac{1}{|X_m|} \sum_{i \in X_m} [y_i = k]$$

3.Критерии останова

3.1. Стрижка деревьев

Критерий останова используется, чтобы принять решение: разбивать вершину дальше или сделать листовой. Худший случай решающего дерева — такое, в котором каждый лист соответствует своему объекту обуча- ющей выборки. В этом случае дерево будет максимально переобученным и не будет обобщать информацию, полученную из обучающей выборки. Грамотно подобранный критерия останова позволяет бороться с переобучением.

Самый простой критерий останова проверяет, все ли объекты в вершине относятся к одному классу. Однако такой критерий останова может быть использован только в случае простых выборок, так как для сложных он остановится только тогда, когда в каждом листе останется примерно по одному объекту.

Гораздо более устойчивый и полезный критерий проверяет, сколько объектов оказалось в вершине, и разбиение продолжается, если это число больше, чем некоторое выбранное n. Соответственно, если в вершину попало ≤ n объектов, она становится листовой. Параметр n нужно подбирать.

Случай n = 1 является худшим случаем, описанным выше. При этом выбирать n нужно так, чтобы по n объектам, которые попали в вершину, можно было устойчиво построить прогноз. Существует рекомендация, что n нужно брать равным 5.

Еще один критерий, гораздо более грубый, заключается в ограничении на глубину дерева. Этот критерий хорошо себя зарекомендовал при построении композиций, когда много решающих деревьев объединяют в один сложный алгоритм. Об этом пойдет речь позже.

Существует и другой подход к борьбе с переобучением деревьев — стрижка. Он заключается в том, что сначала строится решающее дерево максимальной сложности и глубины, до тех пор,

пока в каждой вершине не окажется по 1 объекту обучающей выборки.

После этого начинается «стрижка», то есть удаление листьев в этом дереве по определенному критерию. Например, можно стричь до тех пор, пока улучшается качество некоторой отложенной выборки.

Существует мнение, и это подкреплено многими экспериментами, что стрижка работает гораздо лучше, чем простые критерии, о которых говорилось раньше. Но стрижка — очень ресурсоёмкая процедура, так как, например, может потребоваться вычисление качества дерева на некоторой валидационной выборке на каждом шаге.

На самом деле, сами по себе деревья на сегодняшний день почти не используются, они бывают нужны только для построения композиции и объединения большого числа деревьев в один алгоритм. В случае с композициями такие сложные подходы к борьбе с переобучением уже не нужны, так как достаточно простых критериев останова, ограничения на глубину дерева или на число объектов в листе.

4. Свойства решающих деревьев

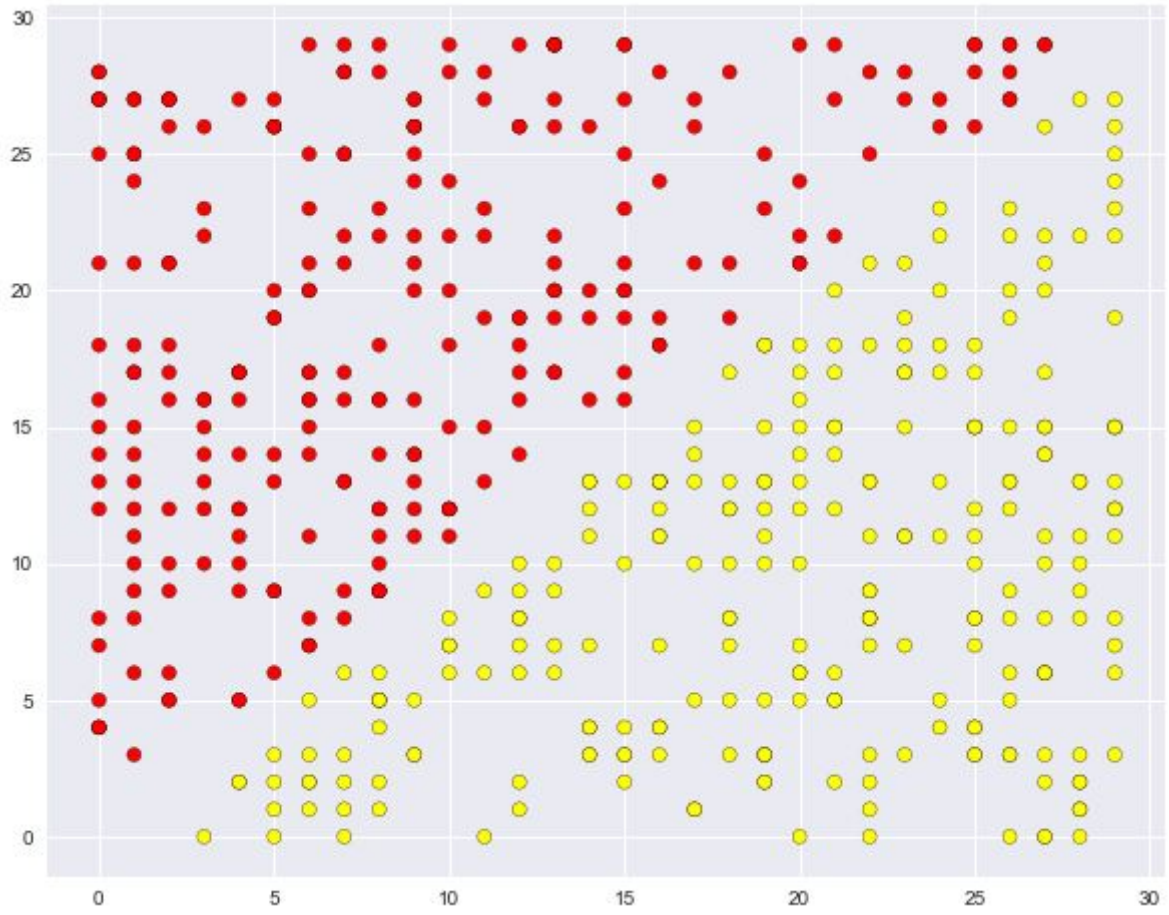
4.1. Преимущества

1. Простота - рассмотренный алгоритм действительно простой, в нем нет очень большой теории, выражается на естественном языке.
2. Интерпретируемость - большинство библиотек поддерживают методы “распечатки” построенного дерева. Любое обученное дерево можно легко визуализировать по пороговым значениями в каждой вершине.
3. Встроенный отбор признаков - заранее нет необходимости проводить анализ на информативность признаков, дерево с помощью стрижки само узнает, что наиболее информативно, а что нет.

4. Работает с различными распределениями, с дискретными/непрерывными признаками - вытекает из определения дерева.
    a. Не нужна нормализация

4.2. Недостатки

1. Жадность не всегда приносит глобальный успех - когда разделяется выборка в некоторой вершине, то решение о разбиении по тому или иному условию определяется лишь локально для конкретного признака, невозможно сказать, насколько хорошо будет идти процесс ветвления дальше. То есть полным перебором можно было бы найти более компактное дерево, решающее исходную задачу, возможно, и с меньшим числом ошибок.
2. Разделяющая прямая может оказаться слишком сложной - самый простой наглядный пример:



3. Для новых данных требуется перестройка дерева - потому что по новым данным нельзя сказать, в какую вершину можно поместить их, ведь нарушится все ее поддерево

Литература

1. Деревья решений - общие принципы работы. https://basegroup.ru/community/articles/description
2. Курс “Введение в машинное обучение” https://www.coursera.org/lecture/vvedenie-mashinnoe-obuchenie/rieshaiushchiie-dieriev-ia-kVz6T
3. Classification and regression trees for machine learning (CART) https://machinelearningmastery.com/classification-and-regression-trees-for-machine-learning/
4. Introduction to classification and regression trees for machine learning (CART) https://www.datasciencecentral.com/profiles/blogs/introduction-to-classification-regression-trees-cart
5. Открытый курс машинного обучения. Тема 3. Классификация, деревья решений и метод ближайших соседей https://habr.com/company/ods/blog/322534/
6. History of Decision Tree https://stats.stackexchange.com/questions/257537/who-invented-the-decision-tree
7. Induction of Decision Trees http://hunch.net/~coms-4771/quinlan.pdf