Chapter 6

Decision Tree
Деревья принятия решений

Подобно методам опорных векторов деревья принятия решений (decision tree) являются универсальными алгоритмами машинного обучения, которые могут заниматься задачами классификации и регрессии, включая даже многовыходовые задачи.

Одним из многих качеств деревьев принятия решений является то, что они требуют совсем небольшой подготовки данных. В частности, для них вообще не нужно масштабирование или центрирование признаков.

Алгоритм обучения CART

Для обучения деревьев принятия решений (также называемых "растущими" деревьями) библиотека Scikit-Learn использует алгоритм дерева классификации и регрессии (***Classification And Regression Tree -CART***). Идея на самом деле довольно проста: алгоритм сначала расщепляет обучающий набор на два поднабора с применением единственного признака k и порога t_k (скажем, "длина лепестка:::; 2.45 см"). Как он выбирает k и t_k? Алгоритм ищет пару (k, t_k), которая производит самые чистые поднаборы (взвешенные по их размеру). Функция издержек, которую он пытается минимизировать, имеет вид, представленный в уравнении 6.2.

Уравнение 6.2. Функция издержек CART для классификации

$$J(k, t_k) = \frac{m_{\text{левый}}}{m} G_{\text{левый}} + \frac{m_{\text{правый}}}{m} G_{\text{правый}},$$

где $\begin{cases} G_{\text{левый/правый}} & \text{измеряет загрязненность левого/правого поднабора,} \\ m_{\text{левый/правый}} & \text{— количество образцов в левом/правом поднаборе.} \end{cases}$

После того как алгоритм успешно расщепил обучающий набор на два поднабора, он расщепляет эти поднаборы, используя ту же самую логику, затем рекурсивно расщепляет подподнаборы и т.д. Алгоритм останавливает рекурсию по достижении максимальной глубины (определенной гиперпараметром max_depth) или когда не может найти расщепление, которое сократило бы загрязненность.

Вычислительная сложность

Вырабатывание прогнозов требует обхода дерева принятия решений от корня до какого-то листа. Деревья принятия решений обычно близки к сбалансированным, так что обход дерева принятия решений требует прохождения через приблизительно ***O(log2(m))*** узлов. Поскольку каждый

узел требует проверки значения лишь одного признака, общая сложность прогноза составляет только ***O(log2(m))*** независимо от количества признаков. Таким образом, прогнозы будут очень быстрыми даже при работе с крупными обучающими наборами.

Загрязненность Джини или энтропия?

По умолчанию применяется мера загрязненности Джини (Gini impurity; также называемая неоднородностью Джини - примеч. пер.), но вместо нее вы можете выбрать меру энтропии загрязненности, установив гиперпараметр `criterion в "entropy".`

Уравнение 6.3. Энтропия

$$H_i = -\sum_{\substack{k=1 \\ p_{i,k} \neq 0}}^{n} p_{i,k} \log\left(p_{i,k}\right)$$

По правде говоря, большую часть времени особой разницы нет: они приводят к похожим деревьям. Загрязненность Джини слегка быстрее подсчитывать, поэтому она является хорошим вариантом по умолчанию.

Гиперпараметры регуляризации

Деревья принятия решений выдвигают очень мало предположений об обучающих данных.

Оставленная не связанной ограничениями, древовидная структура будет адаптировать себя к обучающим данным, очень близко подгоняясь к ним и, скорее всего, допуская переобучение. Такая модель часто называется непараметрической моделью `(nonparametric node` , но не из-за отсутствия каких-либо параметров (они имеются и нередко в изобилии), а по той причине, что количество параметров перед обучением не определено, оттого структура модели вольна тесно привязываться к данным.

В противоположность этому параметрическая модель (`parametric node` , подобная линейной модели, имеет предопределенное количество параметров, так что ее степень свободы ограничивается, сокращая риск переобучения (но увеличивая риск недообучения).

Параметры регуляризации:

- `max_depth -` максимальная глубина (по умолч. `None).` Уменьшение `max_depth` будет регуляризировать модель и соответственно сокращать риск переобучения.
- `min_samples_split -` минимальное число образцов, которые должны присутствовать в узле, прежде чем его можно будет расщепить.

- `min_samples_leaf` - минимальное количество образцов, которое должен иметь листовой узел .
- `min_weight_fraction_leaf` – то же, что и `min_samples_leaf`, но выраженное в виде доли от общего числа взвешенных образцов.
- `max_leaf_nodes` - максимальное количество листовых узлов.
- `max_features` - максимальное число признаков, которые оцениваются при расщеплении каждого узла.

`DecisionTreeClassifier`

Регрессия

`DecisionTreeRegressor`

Ответы на вопросы:

1. **Какой будет приблизительная глубина дерева принятия решений, обученного (без ограничений) на обучающем наборе с 1 миллионом образцов?**

$$log_2(\text{кол} - \text{во образцов}) = \text{кол} - \text{во листьев}$$

$log_2(1\ 000\ 000) \approx 20$ *(листьев)*

2. **Является ли загрязненность Джини узла обычно меньше или больше, чем у его родительского узла? Она обычно меньше/больше или всегда меньше/больше?**

Загрязненность Джини узла обычно ниже, чем у его родителя. Причиной является функция издержек алгоритма обучения CART, которая расщепляет каждый узел способом, сводящим к минимуму.

3. **Если дерево принятия решений переобучается обучающим набором, то хорошей ли идеей будет уменьшение `max_depth`?**

Если дерево принятия решений переобучается обучающим набором, то уменьшение `max_depth` может быть хорошей идеей, так как это регуляризует модель.

4. **Если дерево принятия решений недообучается на обучающем наборе, то хорошей ли идеей будет масштабирование входных признаков?**

Деревья принятия решений не заботятся о том, масштабированы или центрированы обучающие данные; это один из связанных с ними приятных моментов. Следовательно, если дерево принятия решений недообучается на обучающем наборе, тогда масштабирование будет пустой тратой времени.

5. **Если обучение дерева принятия решений на обучающем наборе, содержащем 1 миллион образцов, занимает один час, то сколько примерно времени потребуется для обучения другого дерева принятия решений на обучающем наборе, содержащем 10 миллионов образцов?**

Вычислительная сложность обучения дерева принятия решений составляет $O(n \times m\ log(m))$. Таким образом, если вы умножите размер обучающего набора на 10, то время обучения будет умножено на

$$K = \big(n \times 10m\ log(10m)\big)/(n \times m\ log(m) = 10\log(10m)/\log(m) \approx 11.7$$

K = 11,7 ч.

6. **Если ваш обучающий набор содержит 100 000 образцов, то ускорит ли процесс обучения установка `presort=True`?**

Предварительная сортировка обучающего набора ускоряет обучение, только если набор данных содержит менее нескольких тысяч образцов. В случае 100 000 образцов сортировка значительно замедлит обучение.