ЛЕКЦИЯ 8. Интеллектуальный анализ данных: задачи классификации.

Проблема переобучения

Основной недостаток алгоритма ID3 тенденция к переобучению. Например, если данные содержат шум, то число уникальных значений атрибутов увеличится, а в крайнем случае для каждого примера обучающего множества значения атрибутов окажутся уникальными. Следуя логике ID3, можно предположить, что при разбиении по такому атрибуту будет создано количество узлов, равное числу примеров, так как в каждом узле окажется по одному примеру. После этого каждый узел будет объявлен листом и дерево даст число правил, равное числу примеров обучающего набора, С точки зрения классифицирующей способности такая модель бесполезна.

В качестве примера описанной ситуации возьмем случай, когда атрибут A_1 в выборке, представленной в табл. 1, содержит не три значения A, B и C, а 14 значений от A до N. Тогда в результате разбиения будет сформировано 14 узлов, в каждый из которых попадет только один пример, и на этом построение дерева закончится. Дополнительно усложняет проблему тот факт, что критерий прироста информации всегда будет приводить к выбору атрибутов с наибольшим количеством уникальных значений. Действительно, если в результате разбиения будут получены множества, содержащие по одному объекту, образующему один класс, то частота появления этого класса окажется равной числу примеров, то есть 1. Таким образом, в выражении для оценки количества информации появится $log_2(1) = 0$. Следовательно, и все выражение обратится в ноль, то есть $Info_S(T) = 0$. Всегда будет обеспечен максимальный прирост информации $Gain(S) = max$, что приведет к выбору алгоритмом соответствующего атрибута.

Данная проблема решается введением нормировки. В рассмотрение включается дополнительный показатель, который представляет собой оценку потенциальной информации, созданной при разбиении множества T на n подмножеств T_i:

$$Split - Info(S) = -\sum_{i=1}^{n}\left(\left(\frac{|T_i|}{|T|}\right) log_2\left(\frac{|T_i|}{|T|}\right)\right)$$

С помощью этого показателя можно модифицировать критерий прироста информации, перейдя к отношению:

$$Split - Info(S) = \frac{Gain(T)}{Split - Info(T)}$$

Новый критерий позволяет оценить долю информации, полученной при разбиении, которая является полезной, то есть способствует улучшению

классификации. Использование данного отношения обычно приводит к выбору более удачного атрибута, чем обычный критерий прироста. Вычисление отношения прироста проиллюстрируем с помощью при м ера из табл. 1.

Найдем прирост информации для разбиения S_1:

$$Split - Info(S_1) = -\frac{5}{14} log_2\left(\frac{5}{14}\right) - \frac{4}{14} log_2\left(\frac{4}{14}\right) - \frac{5}{14} log_2\left(\frac{5}{14}\right) = 1{,}577$$

Тогда отношение прироста будет:

$$Split - Info(S_1) = \frac{0{,}246}{1{,}577} = 0{,}156$$

Аналогичная процедура должна быть выполнена для остальных разбиений в дереве решений.

Смысл этой модификации прост. T_i/T – отношение числа примеров в i-м подмножестве, полученном в результате разбиения S, к числу примеров в родительском множестве T. Если в результате разбиения получается большое число подмножеств с небольшим числом примеров, что характерно для переобучения, то параметр $Split - Info$ растет. Поскольку он стоит в знаменателе выражения, то $Gain - ratio$ есть обычный прирост информации, «оштрафованный» с помощью параметра $Split - Info$. Благодаря этому атрибут, для которого параметр $Split - Info$ растет, имеет меньше шансов быть выбранным для разбиений, чем при использовании обычного $Gain$-критерия.

Неизвестные значения атрибутов

Рассматривая алгоритмы, мы предполагали, что все значения атрибутов, используемых при разбиении, известны. Но типичной проблемой реальных наборов является отсутствие значений атрибутов в отдельных наблюдениях. Если в наборе данных, для которого строится дерево решений, имеют место отсутствующие значения, то можно выбрать один из вариантов решения данной проблемы:

- исключить все наблюдения, содержащие пропущенные значения;
- создать новый или модифицировать существующий алгоритм таким образом, чтобы он мог работать с пропусками в данных.

Первое решение простое, но оно часто неприемлемо из-за того, что в наборе данных может присутствовать большое количество записей с пропущенными значениями. И если все такие записи будут исключены из рассмотрения, то оставшихся записей окажется недостаточно для формирования обучающего множества.

Различные классификационные алгоритмы обычно основаны на заполнении пропусков наиболее вероятными значениями.

В алгоритме C4.5 предполагается, что наблюдения с неизвестными значениями имеют статистическое распределение соответствующего атрибута согласно относительной частоте появления известных значений. Например, рассмотрим множество наблюдений из табл. 1, в котором отсутствует значение атрибута A_1 в строке 6.

Введем в рассмотрение параметр F, который представляет собой число наблюдений в наборе данных с известным значением данного атрибута, отнесенное к общему числу наблюдений. Тогда модифицированный для работы с пропущенными значениями критерий прироста информации будет иметь вид:

$$Gain(S) = F(Info(T) - Info_S(T))$$

Аналогично параметр $Split - Info(S)$ может быть видоизменен путем отнесения числа наблюдений, содержащих пропущенные значения, к отдельным дополнительно создаваемым подмножествам. Если разбиение S дает n подмножеств, его параметр $Split - Info(S)$ вычисляется, как если бы в результате разбиения было получено $n + 1$ подмножество.

Рассмотрим работу алгоритма C4.5, модифицированного для обработки пропущенных значений, на примере из табл. 1.

Таблица 1. Набор данных для иллюстрации работы алгоритма C4.5

N п/п	A_1	A_2	A_3	C
1	A	70	Да	C_1
2	A	90	Да	C_2
3	A	85	Нет	C_2
4	A	95	Нет	C_2
5	A	70	Нет	C_1
6		90	Да	C_1
7	B	78	Нет	C_1
8	B	65	Да	C_1
9	B	75	Нет	C_1
10	C	80	Да	C_2
11	C	70	Да	C_2
12	C	80	Нет	C_1
13	C	80	Нет	C_1
14	C	96	Нет	C_1

Из 13 наблюдений, в которых значения атрибута A_1 известны, 8 относятся к классу C_1 и 5 к классу C_2. Поэтому в результате разбиения по атрибуту A_1 получим:

$$Info_{S_1}(T) = -\left(\frac{8}{13}\right)log_2\left(\frac{8}{13}\right) - \left(\frac{5}{13}\right)log_2\left(\frac{5}{13}\right) = 0{,}961 \text{ бит}$$

После разделения исходного множества T с помощью атрибута A_1 на подмножества в соответствии со значениями атрибута A, B и C результирующая информация определяется следующим образом:

$$Info_{S_1}(T) = \frac{5}{13}\left(-\frac{2}{5}log_2\left(\frac{2}{5}\right) - \frac{3}{5}log_2\left(\frac{3}{5}\right)\right) + \frac{3}{13}\left(-\frac{3}{3}log_2\left(\frac{3}{3}\right) - \frac{0}{3}log_2\left(\frac{0}{3}\right)\right) + \frac{5}{14}\left(-\frac{3}{5}log_2\left(\frac{3}{5}\right) - \frac{2}{5}log_2\left(\frac{2}{5}\right)\right) = 0{,}747 \text{ бит}$$

Прирост информации, полученный в результате данного разбиения, корректируется с помощью фактора $F = 13/14 = 0{,}93$. Тогда:

$$Gain(S_1) = F(0{,}961 - 0{,}747) = 0{,}93(0{,}961 - 0{,}747) = 0{,}199 \text{ бит}$$

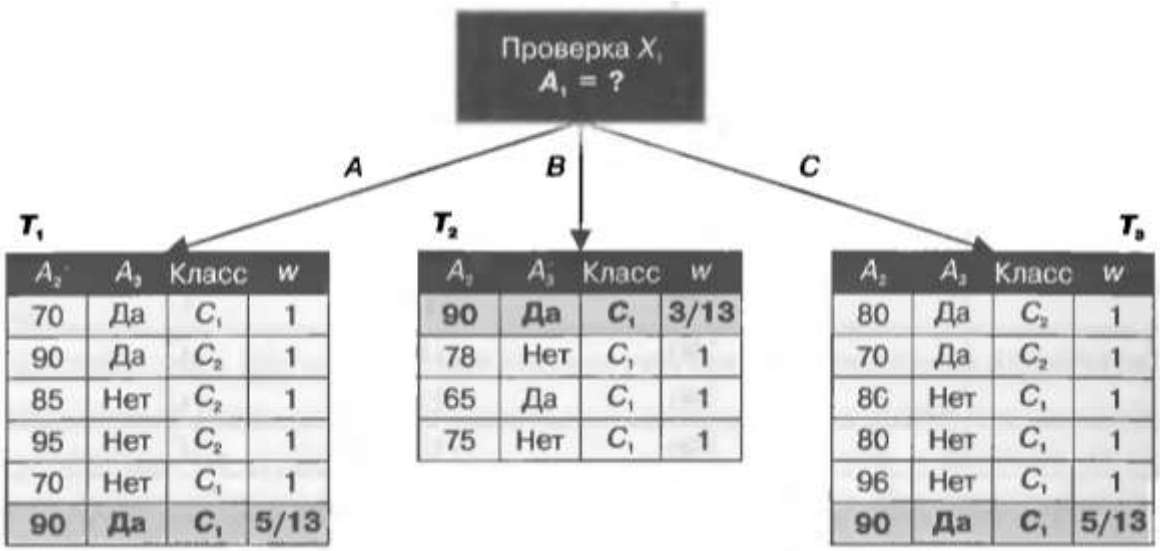
Прирост, полученный в результате данного разбиения, несколько ниже, чем для множества из табл. 1 (то есть без пропусков), где он составлял 0,246:

$$Split - Info(S_1) = -\left(\frac{5}{13}\right)log_2\left(\frac{5}{13}\right) - \left(\frac{3}{13}\right)log_2\left(\frac{3}{13}\right) - \left(\frac{5}{13}\right)log_2\left(\frac{5}{13}\right) - \left(\frac{1}{13}\right)log_2\left(\frac{1}{13}\right) = 1{,}876$$

Когда наблюдение с известным значением атрибута переходит из множества T в подмножество T_i вероятность его отношения к T_i, равна 1, а ко всем остальным подмножествам – 0. Если в примере имеется пропущенное значение, то относительно данного примера может быть сделано только некоторое вероятностное предположение. При этом очевидно, что вероятность принадлежности примера ко всем подмножествам, полученным в результате разбиения, будет равна 1 (то есть к какому-либо из них он будет отнесен обязательно).

После разбиения множества T на подмножества по атрибуту A_1 запись, содержащая пустое значение, будет распределена во все три полученных подмножества. Данная ситуация проиллюстрирована на рис. 4.

Веса w_i для примеров с пропусками будут равны вероятностям $5/13, 3/13$ и $5/13$ соответственно. Теперь T_i может быть интерпретировано алгоритмом не как число записей в соответствующем подмножестве, а как сумма весов w для данного подмножества.



A_2	A_3	Класс	w
70	Да	C_1	1
90	Да	C_2	1
85	Нет	C_2	1
95	Нет	C_2	1
70	Нет	C_1	1
90	**Да**	C_1	**5/13**

A_2	A_3	Класс	w
90	**Да**	C_1	**3/13**
78	Нет	C_1	1
65	Да	C_1	1
75	Нет	C_1	1

A_2	A_3	Класс	w
80	Да	C_2	1
70	Да	C_2	1
80	Нет	C_1	1
80	Нет	C_1	1
96	Нет	C_1	1
90	**Да**	C_1	**5/13**

Рис. 4. Пример разбиения множества

Алгоритм CART

CART (Classification and Regression Tree) популярный алгоритм построения деревьев решений, предложенный в 1984 г. Деревья решений, построенные с помощью CART, являются бинарными, то есть содержат только два потомка в каждом узле.

Пусть задано обучающее множество, содержащее K примеров и N классов. Введем в рассмотрение показатель, который позволит оценить эффективность разбиения, полученного на основе конкретного атрибута. Обозначим его $Q(s|t)$, где s – идентификатор разбиения, t – идентификатор узла. Тогда можно записать:

$$Q(s|t) = 2P_L P_R \sum_{j=1}^{N} \left(P(j|t_L) - P(j|t_R)\right) \quad (1)$$

где t_L и t_R – левый и правый потомки узла t соответственно;

$P_L = K_L/K$ – отношение числа примеров в левом потомке узла t к общему числу примеров;

$P_R = K_R/K$ – отношение числа примеров в правом потомке узла t к общему числу примеров;

$P(j|t_L) = K_j^L/K_t$ – отношение числа примеров j-го класса в t_L к общему числу примеров в t_L;

$P(j|t_R) = K_j^R/K_t$ – отношение числа примеров j-го класса в t_R к общему числу примеров в t_R.

Тогда наилучшим разбиением в узле t будет то, которое максимизирует показатель $Q(s|t)$. Рассмотрим ту же задачу предсказания кредитного риска потенциального клиента банка, что и в примере на алгоритм С4,5 (табл. 2.),

Таблица 2. Кредитные истории клиентов

№ клиента	Сбережения	Другие активы (недвижимость, автомобиль и т.д.)	Годовой доход, тыс. у. е.	Кредитный риск
1	Средние	Высокие	75	Низкий
2	Низкие	Низкие	50	Высокий
3	Высокие	Средние	25	Высокий
4	Средние	Средние	50	Низкий
5	Низкие	Средние	100	Низкий
6	Высокие	Высокие	25	Низкий
7	Низкие	Низкие	25	Высокий
8	Средние	Средние	75	Низкий

Все восемь примеров обучающего множества поступают в корневой узел дерева. Поскольку алгоритм CART создает только бинарные разбиения, возможные кандидаты, которые будут оцениваться на начальном этапе, представлены в табл. 3. Непрерывный атрибут Доход был предварительно квантован с использованием порогов 25, 50 и 75 тыс.

Таблица 3. Потенциальные разбиения

Разбиение	Левый потомок, t_L	Правый потомок, t_R
1	Сбережения = Низкие	Сбережения ∈ {Высокие, Средние}
2	Сбережения = Средние	Сбережения ∈ {Высокие, Низкие}
3	Сбережения = Высокие	Сбережения ∈ {Средние, Низкие}
4	Активы = Низкие	Активы ∈ {Высокие, Средние}
5	Активы = Средние	Активы ∈ {Высокие, Низкие}
6	Активы = Высокие	Активы ∈ { Средние, Низкие}
7	Доход ≤ 25	Доход > 25
8	Доход ≤ 50	Доход > 50
9	Доход ≤ 75	Доход > 75

Для каждого потенциального разбиения вычислим значения составляющих выражения (1) и исследуем их влияние на значение всего показателя Q.

Можно увидеть, что $Q(s|t)$ увеличивается, когда оба сомножителя в выражении 1: $2P_L P_R$ и $\sum_{j=1}^{N}\left(P(j|t_L) - P(j|t_R)\right)$ – также увеличиваются.

Обозначим сумму в выражении 1 через $W(s|t)$, то есть:

$$W(s|t) = \sum_{j=1}^{N}\left(P(j|t_L) - P(j|t_R)\right)$$

Компонент $W(s|t)$ будет расти при увеличении разности в скобках, Данная сумма максимальна, когда количество примеров, относящихся к одному классу, в обоих потомках максимально различается. Следовательно, значение окажется наибольшим тогда, когда оба потомка вообще не будут содержать примеров одинаковых классов. В частности, если классов только два, то оба потомка будут чистыми. Теоретически максимальное значение $W(s|t)$ равно числу классов в обучающем множестве, Поскольку в нашем примере только два класса – *Высокий* и *Низкий*, максимальное значение $W(s|t)$ равно 2. В табл. 4 приведены результаты расчета компонентов выражения (1).

Таблица 4. Расчет значений меры Q для потенциальных разбиений

№	P_L	P_R	$P(j\|t_L)$		$P(j\|t_R)$		$2P_LP_R$	$W(s\|t)$	$Q(s\|t)$
			Низкий	Высокий	Низкий	Высокий			
1	0,375	0,625	0,333	0,667	0,8	0,2	0,46875	0,934	0,4378
2	0,375	0,625	1	0	0,4	0,6	0,46875	1,2	0,5625
3	0,25	0,75	0,5	0,5	0,667	0,333	0,375	0,334	0,1253
4	0,25	0,75	0	1	0,833	0,167	0,375	1,667	0,6248
5	0,5	0,5	0,75	0,25	0,5	0,5	0,5	0,5	0,25
6	0,25	0,75	1	0	0,5	0,5	0,375	1	0,375
7	0,375	0,625	0,333	0,667	0,8	0,2	0,46875	0,934	0,4378
8	0,625	0,375	0,4	0,6	1	0	0,46875	1,2	0,5625
9	0,875	0,125	0,571	0,429	1	0	0,21875	0,858	0,1877

Рассмотрим расчет P_L. Смотрим 2 столбец 1 строки таб. 3 – *Сбережения = Низкие*. В табл. 2 в столбце 2 (Сбережения) существует 8 значений из них 3 (2, 5, 7) – *Низкие*. $P_{L1} = K_{L1}/K = \frac{3}{8} = 0{,}375$. Аналогично, смотрим 2 столбец 2 строки таб. 3 – *Сбережения = Средние*. В табл. 2 в столбце 2 (Сбережения) существует 8 значений из них 3 (1, 4, 8) – *Средние*. $P_{L1} = K_{L1}/K = \frac{3}{8} = 0{,}375$ и т.д.

Рассмотрим расчет P_R. Смотрим 3 столбец 1 строки таб. 3 – Сбережения $\in$ {Высокие, Средние}. В табл. 2 в столбце 2 (Сбережения) существует 8 значений из них 3 (1, 4, 8) – *Средние* и 2 (3,6) – *Высокие*. $P_{R1} = K_{R1\text{Средние}} + K_{R1\text{Высокие}}/K = \frac{3+2}{8} = 0{,}625$. Соответственно, можно рассчитывать $P_{R1} = 1 - P_{L1} = 1 - 0{,}375 = 0{,}625$. Аналогично, рассчитываем остальные значения.

Рассмотрим расчет $P(j|t_L)$ низкий. Смотрим 2 столбец 1 строки таб. 3 – *Сбережения = Низкие*. В таб. 3 выделим 3 столбца (№ клиента, Сбережения, Кредитный риск) и 3 строки (2, 5, 7) с показателем *Сбережения – Низкие* (табл. 5). Исходя из табл. 5 существует 3 значения сбережения *Низкие* из них 1 кредитный риск *Низкий*. Тогда $P(1|t_L)_{\text{низкий}} = K^L_{1\,\text{низкий}}/K_{\text{сбережения низкие}} = 1/3 = 0{,}333$

Таблица 5. Расчет $P(j|t_L)$

№ клиента	Сбережения	Кредитный риск
2	Низкие	Высокий
5	Низкие	Низкий
7	Низкие	Высокий

Аналогично, исходя из табл. 5 существует 3 значения сбережения *Низкие* из них 2 кредитный риск *Высокий*. Тогда $P(1|t_L)_{\text{высокий}} = K^L_{1\text{ высокий}}/K_{\text{сбережения низкие}} = 2/3 = 0{,}667$. И так далее для $P(j|t_L)$ и $P(j|t_R)$ – *Низкий* и *Высокий*.

Заполняется столбец $2P_LP_R$. Тогда для первой строки $2P_{L1}P_{R1} = 2(0{,}375 \times 0{,}625) = 0{,}46875$.

Заполняется столбец $W(s|t)$. $W(s|t)_1 = (P(1|t_L)_{\text{низкий}} - P(1|t_R)_{\text{низкий}}) + (P(1|t_L)_{\text{высокий}} - P(1|t_R)_{\text{высокий}}) = (0{,}8 - 0{,}333) + (0{,}667 - 0{,}2) = 0{,}934$

Заполняется столбец $Q(s|t)$. $Q(s|t)_1 = 2P_{L1}P_{R1} \times W(s|t)_1 = 0{,}46875 \times 0{,}934 = 0{,}4378$

Произведение $P_L \times P_R$ возрастает с увеличением значений сомножителей. Это происходит, когда доли записей одного класса в левом и правом потомках оказываются равны. Следовательно, мера $Q(s|t)$ имеет тенденцию давать сбалансированные разбиения, которые будут делить исходное множество на подмножества, содержащие примерно одинаковое количество записей, Теоретически максимальное значение $2P_LP_R = 2 \times 0{,}5 \times 0{,}5 = 0{,}5$.

В нашем примере только потенциальное разбиение 5 (таб.4) дает произведение $P_L \times P_R$, достигающее теоретического максимума 0,5, поскольку в результате записи были разделены на две равные группы по четыре в каждой,

Наибольшее из наблюдаемых значений $Q(s|t)$ было получено для разбиения 4 (табл. 9.11), где $Q(s|t) = 0{,}6248$. Значит, для начального разбиения CART выберет условие, являются ли активы клиента низкими. Тогда в результате разбиения будут созданы два потомка: в одном окажутся записи, в которых атрибут *Активы* принимает значение *Низкие*, во втором записи, в которых этот же атрибут принимает значения *Высокие* и *Средние*. Полученное в результате данного разбиения дерево представлено на рис. 5.

Обе записи, в которых активы клиента являются низкими и по этой причине оказавшиеся в левом узле, содержат одну и ту же целевую переменную, указывающую на высокий кредитный риск Таким образом, узел является чистым. Узел будет объявлен листом, и дальнейшее разбиение по данной ветви производиться не будет. Записи в правом узле относятся к различным классам.

Рис. 5. Дерево после первого разбиения

Потребуется дальнейшее разбиение, поэтому мы снова рассчитаем $Q(s|t)$ и представим полученные результаты в табл. 6. Обратим внимание на то, что ранее использованное разбиение 4 больше не рассматривается и № клиента 2 и 7 удаляются из таблицы 2.

Таблица 6. Второй этап расчета значений меры Q

№	P_L	P_R	$P(j\|t_L)$		$P(j\|t_R)$		$2P_LP_R$	$W(s\|t)$	$Q(s\|t)$
			Низкий	Высокий	Низкий	Высокий			
1	0,167	0,833	1	0	0,8	0,2	0,2782	0,4	0,1112
2	0,5	0,5	1	0	0,667	0,333	0,5	0,6666	0,3333
3	0,333	0,667	0,5	0,5	1	0	0,4444	1	0,4444
4	0,667	0,333	0,75	0,25	1	0	0,4444	0,5	0,2222
5	0,333	0,667	1	0	0,75	0,25	0,4444	0,5	0,2222
6	0,333	0,667	0,5	0,5	1	0	0,4444	1	0,4444
7	0,5	0,5	0,667	0,333	1	0	0,5	0,6666	0,3333
8	0,167	0,833	0,8	0,2	1	0	0,2782	0,4	0,1112

Наибольшее значение меры $Q(s|t) = 0{,}4444$ было получено для разбиений 3 и 6. Произвольным образом выберем разбиение 3 *Сбережения = Высокие*. В результате дерево будет дополнено двумя новыми узлами, В левом потомке окажутся записи, в которых атрибут *Сбережения* принимает значение *Высокие*. Таких записей в исходном множестве всего две (табл. 2, записи 3 и 6). Однако они имеют разное значение целевой переменной: для записи 3 кредитный риск высокий, а для записи 7 – низкий. Следовательно, данный узел содержит два класса и в нем возможно дальнейшее ветвление. Во второй узел этого разбиения будут отобраны оставшиеся записи – 1,4,5 и 8. Как можно увидеть из табл. 2, все они имеют одну и ту же метку класса, указывающую на низкий риск кредитования заемщиков. Поскольку записи, попавшие в данный узел, относятся к одному классу, узел объявляется листом. Результирующее дерево представлено на рис. 6.

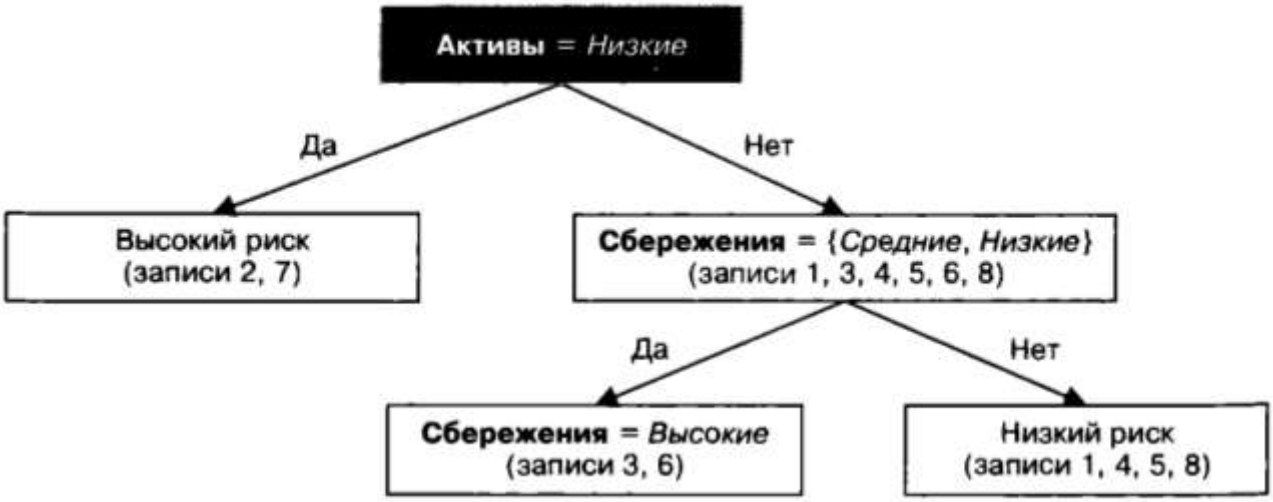


Рис. 6. Дерево после второго разбиения

Новое разбиение возможно только для узла, содержащего записи 3 и 6, относящиеся к различным классам. Для разбиения можно использовать два ранее не применявшихся атрибута – *Доход* и *Другие активы*. Однако, поскольку в обеих записях сумма дохода одна и та же (25 000 у. е.), это ничего не даст. В то же время значения атрибута *Другие активы* различаются: для записи 3 – *Средние*, а для записи 6 – *Высокие*, поэтому его можно использовать для разбиения. Результирующее дерево представлено на рис. 7.

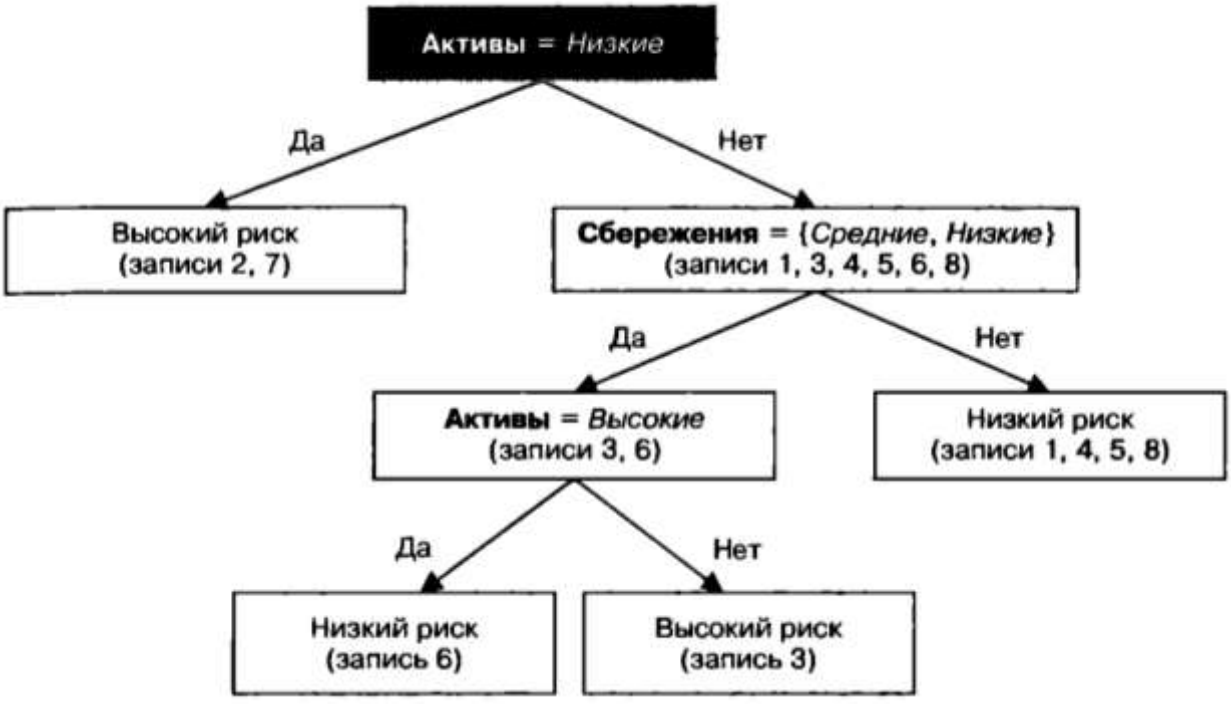


Рис. 7. Дерево после третьего разбиения

В общем случае алгоритм будет рекурсивно продолжаться, пока в дереве остаются узлы, содержащие примеры, которые относятся к различным классам. Когда узлов, для которых можно выполнить разбиение, не останется (то есть все подветви будут заканчиваться листьями), будет построено полное дерево. Но при

работе с реальными наборами данных часто возникает ситуация, когда получить абсолютно чистые узлы не удается, что ведет к возникновению ошибки классификации. Рассмотрим набор данных, представленный в табл. 7.

Таблица 7. Набор данных

№ клиента	Сбережения	Другие активы (недвижимость, автомобиль и т.д.)	Годовой доход, тыс. у. е.	Кредитный риск
1	Высокие	Низкие	< 30	Низкий
2	Высокие	Низкие	< 30	Низкий
3	Высокие	Низкие	< 30	Высокий
4	Высокие	Низкие	< 30	Высокий
5	Высокие	Низкие	< 30	Высокий

Визуальный анализ табл. 7. показывает, что все представленные в ней потенциальные заемщики характеризуются высокими сбережениями, но низкими (менее 30 тыс, у. е,) доходами. В то же время, несмотря на одинаковые значения атрибутов, целевые переменные для идентичных записей окажутся разными, Как известно, записи, где одному и тому же набору значений входных переменных соответствуют разные значения выходных, называются противоречивыми и от них стремятся избавиться путем очистки данных. На практике подобная ситуация может сложиться, если атрибуты *Сбережения* и *Другие активы* получены путем квантования соответствующих непрерывных атрибутов.

Очевидно, что в данном случае невозможно получить чистые узлы, так как одинаковые значения независимых атрибутов не позволят разделить объекты по классам. В результате в листьях окажутся «смеси» нескольких классов. В такой ситуации можно отнести всех клиентов к категории *Высокого* кредитного риска. Тогда вероятность того, что случайно выбранная из данного набора запись будет отнесена к соответствующему классу, составит $3/5 = 0{,}6$, или 60%. В то же время вероятность неправильной классификации составит 0,4, или 40 %. Данное значение называется ошибкой классификации. Если узлов, в которых допущена ошибка классификации, несколько, то полная ошибка дерева есть средневзвешенная ошибка по всем узлам. При этом в качестве весов используются доли записей в каждом листе относительно общего количества записей в обучающем множестве.

Регрессионною дерево решений

Как следует из названия алгоритма CART классификационные и регрессионные деревья решений, он позволяет строить не только классификационные, но и регрессионные модели. В основном процесс построения

регрессионного дерева аналогичен процессу построения классификационного, но вместо меток классов в листьях будут расположены числовые значения. Фактически регрессионные деревья реализуют кусочно-постоянную функцию входных переменных (рис. 8.).

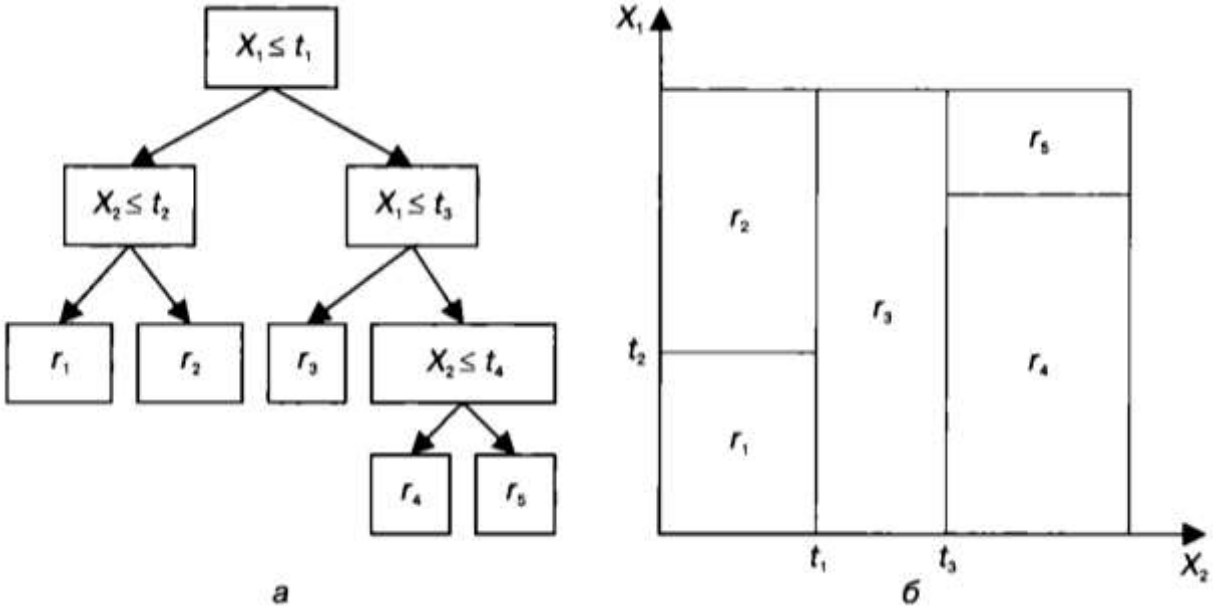


Рис. 8. Регрессионное дерево решений

На рис. 8а представлено дерево решений, построенное для двух входных переменных X_1 и X_2 и содержащее 5 листьев, На рис. 8б показано разбиение (кусочно-постоянная функция двух переменных), которое реализует данное дерево.

В результате построения регрессионного дерева в каждом листе должны оказаться примеры с близкими значениями выходной переменной, Чем ближе будут эти значения, тем меньше их дисперсия. Поэтому дисперсия является хорошей мерой чистоты узла.

Для минимизации квадратичной ошибки на обучающем множестве результат в листе определяется как среднее значений выходных переменных обучающих примеров, распределенных в данный лист. При этом мера «загрязнённости» листа I пропорциональна дисперсии D_y значений выходной переменной примеров в узле:

$$I = D_y = E\left\{\left(y_n - E_y\right)^2\right\}$$

Тогда наилучшим разбиением в узле будет то, которое обеспечит максимальное уменьшение дисперсии выходной переменной. Следует отметить, что для регрессионных деревьев решений упрощение важнее, чем для классификационных. Это связано с тем, что регрессионные деревья, как правило, получаются более сложными, чем классификационные, поскольку количество

значений непрерывной целевой переменной намного разнообразнее, чем категориальной. Например, если значения непрерывной целевой переменной уникальны для каждого примера обучающего множества, то полное дерево будет содержать число листьев, равное числу примеров, Процедура упрощения дерева путем отсечения ветвей основана на анализе квадратичной ошибки на тестовом множестве: отсекаются все узлы, удаление которых не приводит к росту ошибки, превышающему заданное значение.

Критерии оптимизации деревьев решений

Существуют два основных подхода к выбору оптимальной сложности дерева решений.

1. Принудительная остановка алгоритма с помощью условия, при выполнении которого рост дерева автоматически остановится. Этот метод называется ранней остановкой (рис. 9.). Здесь могут использоваться такие параметры, ограничивающие рост дерева, как минимальное количество примеров в узле глубина дерева, статистическая значимость разбиений и т. д,

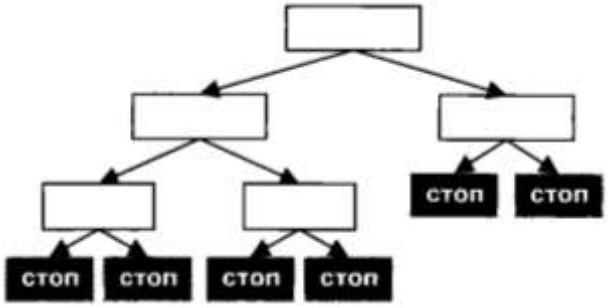


Рис. 9. Ранняя остановка построения дерева решений

2. Сначала строится полное дерево, затем производится его упрощение путем отсечения ветвей (рис. 10), В данном случае создается последовательность поддеревьев в порядке увеличения их сложности, После этого при помощи определённого критерия выбирается лучшее поддерево. Обычно в качестве такого критерия используется эффективность работы дерева на валидационном множестве.

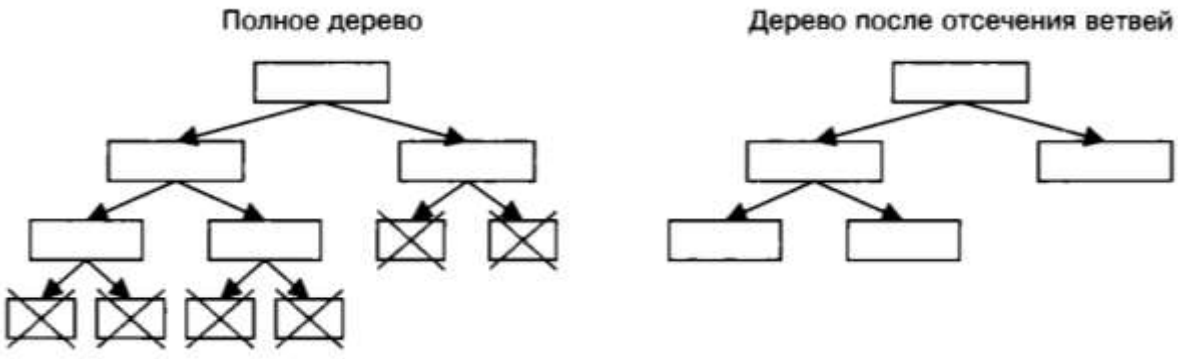


Рис. 10. Отсечение ветвей

Обычно процесс построения дерева решений делится на две фазы: фазу роста и фазу упрощения. Если упрощение производится первым способом, то говорят, что имеет место ранняя остановка. Для упрощения вторым способом используют термин «отсечение ветвей».

Предварительная остановка менее затратная в вычислительном плане, но возникает риск потери хороших разбиений, которые могут следовать за плохими. Поэтому отсечение ветвей, как правило, позволяет получить лучшие результаты и пользуется большей популярностью.

Алгоритм дерева решений строит лучшее разбиение в корневом узле, где присутствует вся выборка записей, На более низких уровнях, по мере того как записи распределяются по узлам, узлы становятся меньше и в них преобладают примеры с близкими значениями атрибутов. С уменьшением числа примеров в узлах падает ценность связанных с ними правил. И если на не котором уровне разбиение дало узлы, содержащие один два примера, то такие разбиения не имеют смысла, поскольку обобщающая способность модели оказывается слабой.

Данный фактор обычно связывают с переобучением – адаптацией модели к частным случаям, нетипичным примерам, шумам в данных и т, д. Вследствие переобучения модель оказывается неустойчивой, а ее поведение при работе с реальными данными непредсказуемым. Способом решения проблемы является удаление неустойчивых раз биений путем объединения узлов с небольшим числом примеров.

Упрощение деревьев решений преследует такие цели:

- уменьшить сложность дерева и извлеченных из него правил, повысить интерпретируемость классификационной модели;
- повысить устойчивость и обобщающую способность модели;
- сократить вычислительные затраты, связанные с работой модели,