Лекция 4

ЗАДАЧА КЛАССИФИКАЦИИ. МЕТОД ДЕРЕВЬЕВ РЕШЕНИЙ

Основные положения метода

Метод деревьев решений (***decision tree***) для задачи классификации состоит в том, чтобы осуществлять процесс деления исходных данные на группы, пока не будут получены однородные (или почти однородные) их множества. Совокупность правил, которые дают такое разбиение (***partition***), позволят затем делать прогноз (определять наиболее вероятный номер класса) для новых данных.

Метод деревьев решений применим для решения задач классификации, возникающих в самых разных областях, и считается одним из самых эффективных.

Примеры практических задач классификации:

- *скоринговые модели* кредитования;
- *маркетинговые исследования*, направленные на выявление предпочтений клиента или степени его удовлетворённости;
- *диагностика* (медицинская или техническая).

Основные понятия

Дерево решений *– это модель, представляющая собой совокупность правил для принятия решений*.

Графически её можно представить в виде древовидной структуры, где моменты принятия решений соответствуют так называемым ***узлам*** (***decision nodes***).

В узлах происходит **ветвление** процесса (***branching***), т.е. деление его на так называемые **ветви** (***branches***) в зависимости от сделанного выбора.

Конечные (или, терминальные) узлы называют ***листьями*** (***leafs***, ***leaf nodes***), каждый лист – это конечный результат последовательного принятия решений.

Данные, подлежащие классификации, находятся в так называемом «***корне***» дерева (***root***).

В зависимости от решения, принимаемого в узлах, процесс в конце концов останавливается в одном из листьев, где переменной отклика (искомому номеру класса) присваивается то или иное значение.

Идея метода

Метод деревьев решений реализует принцип так называемого «***рекурсивного деления***» (recursive partitioning).

Эта стратегия также называется «***Разделяй и властвуй***» («***Divide and conquer***»).

В узлах, начиная с корневого, выбирается признак, значение которого используется для разбиения всех данных на 2 класса.

Процесс продолжается до тех пор, пока не выполнится *критерий остановки*:

- Все (или почти все) данные данного узла принадлежат одному и тому же классу;
- Не осталось признаков, по которым можно построить новое разбиение;
- Дерево превысило заранее заданный «лимит роста» (если таковой был заранее установлен).

Пример

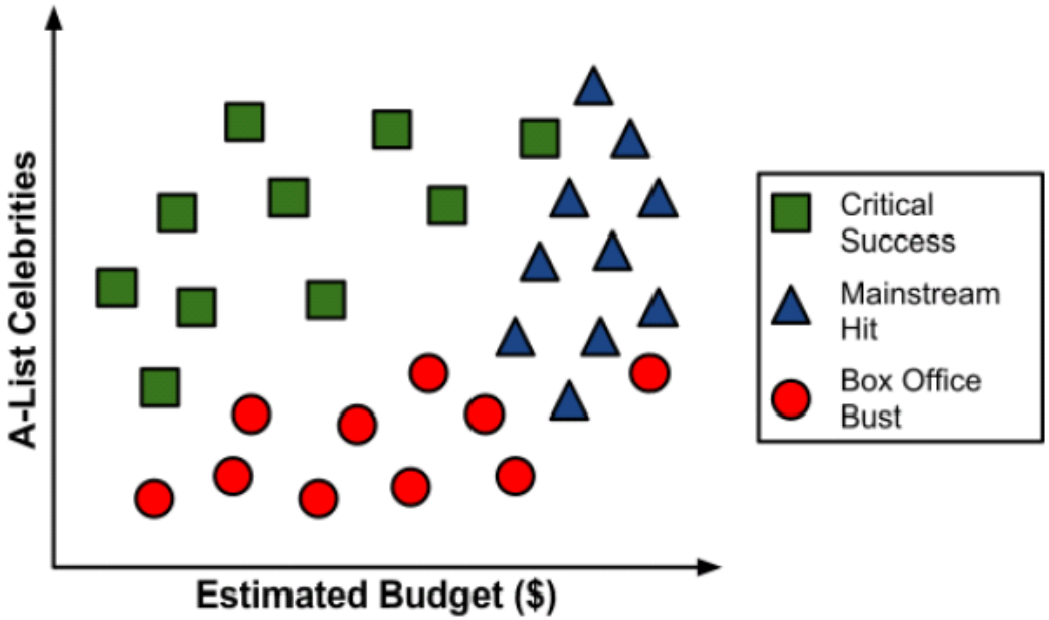
В кинокомпании стол редактора завален сценариями кинофильмов, нужно разложить их по трём ящикам:

- Популярные («mainstream hits»);
- Не популярные у зрителей, но получившие высокую оценку критиков;
- Не имеющие успеха.

Не прочитывая каждый сценарий нужно разработать алгоритм классификации сценариев по трем классам.

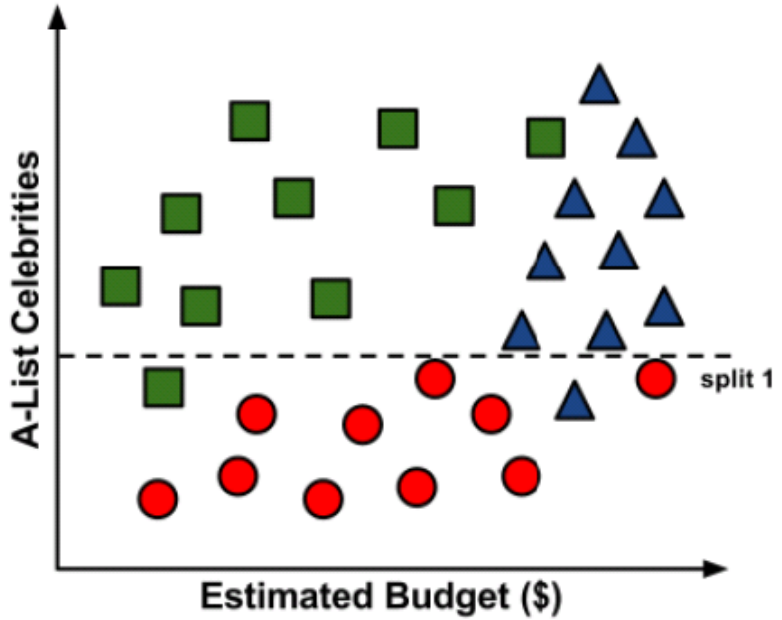
В архиве киностудии собраны данные о киносценариях за последние 10 лет.

После просмотра 30 киносценариев замечено, что фильмы с высоким бюджетом привлекают больше звёзд.



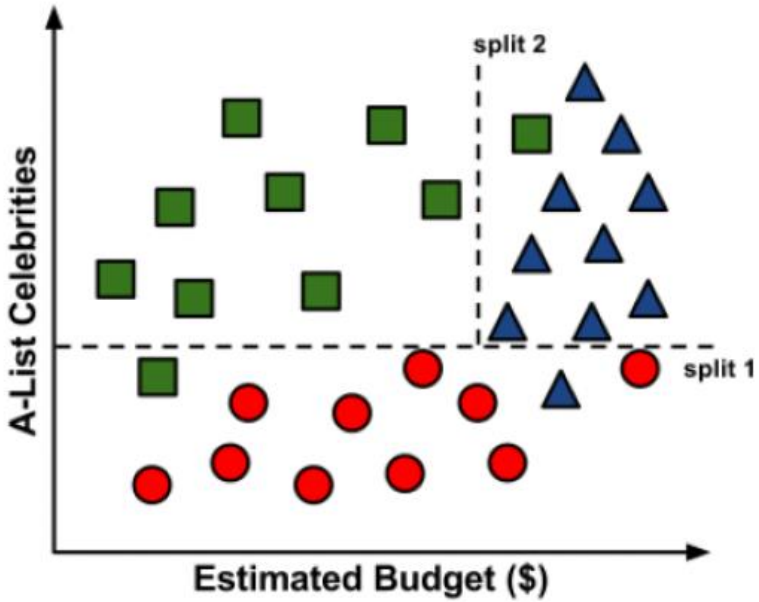
Пример

1) Количество снимавшихся в фильме звёзд как первый из признаков, по которому производится разбиение данных



2) Рассмотрим группу сценариев с большим числом задействованных звёзд.

Разобьём её на 2 подгруппы по признаку «размер бюджета»



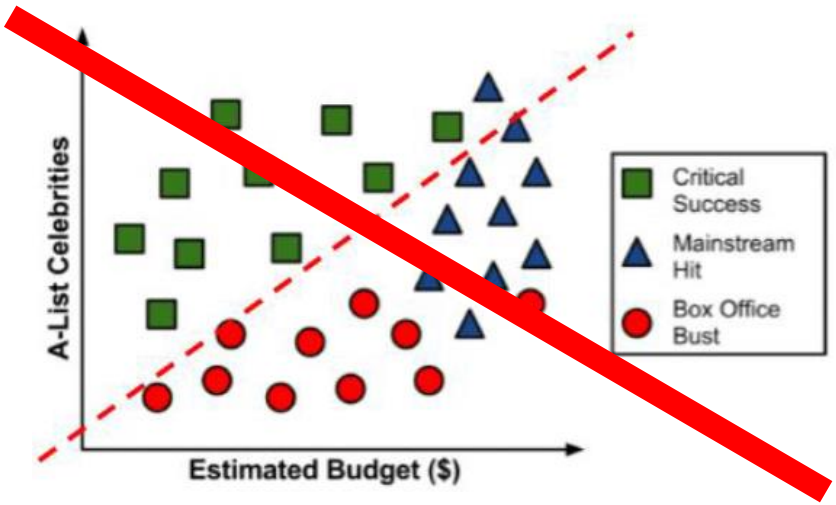
Левая верхняя группа - фильмы, которые были высоко оценены критиками (большое число занятых звёзд, но относительно небольшой бюджет).
Группа в правом верхнем углу – фильмы с большим бюджетом и большим числом звёзд;
Третья группа – состоит преимущественно из фильмов, не имевших успеха (число звёзд, занятых в них, невелико, а их бюджет варьируется от очень малого до очень большого).

Пример

Продолжать процесс разделения данных можно и дальше, пока не получим очень «мелкое» разделение (может оказаться, что каждая группа будет содержать лишь по одному элементу), однако понятно, что смысла в такой классификации нет.

Ограничим ветвление дерева – например, остановим процесс, когда *каждая группа хотя бы на 80% будет состоять из элементов одного и того же класса*.

Заметим, что в данном случае мы говорим лишь о разбиениях данных (точек в Евклидовом пространстве) прямыми (в общем случае – гиперплоскостями), параллельными осям координат.



Метод деревьев решений не позволяет строить такие разбиения, поскольку для разделения данных используются условия вида:

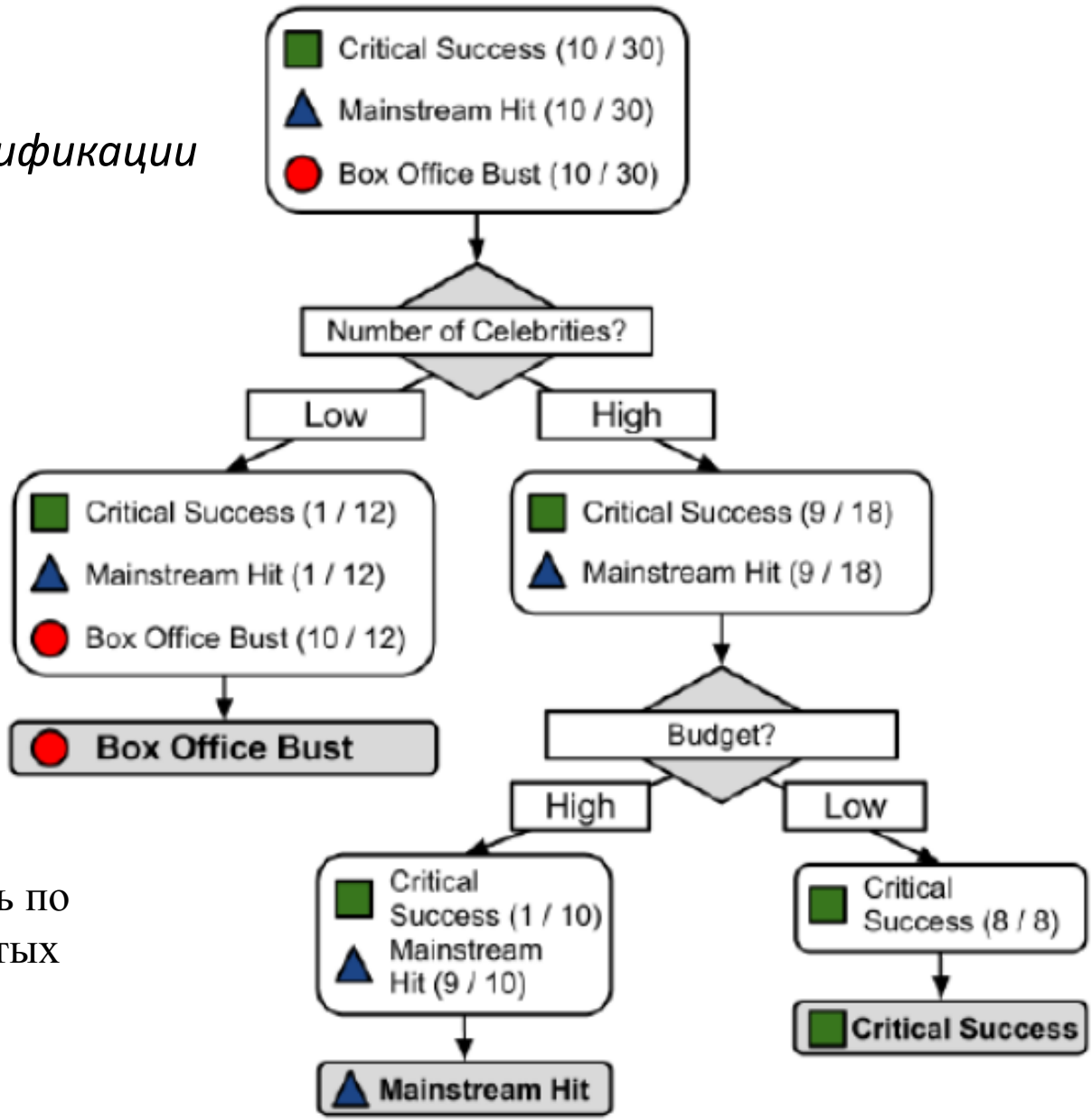
$$\{x < a\}, \{x > a\},$$

где x – значение фактора, a – некоторое фиксированное число.

«***axis-parallel splits***», т.е. «**разбиения, параллельные осям**».

Пример

Дерево решений для классификации киносценариев



Классификация сценариев лишь по 2-м факторам (количество занятых звёзд и бюджет) - дерево получилось небольшим.

Численные алгоритмы метода деревьев решений, допускающие компьютерную реализацию

Существуют различные численные алгоритмы построения деревьев решений: ***CART*, C4.5, CHAID, CN2, NewId, ITrule** и другие.

Алгоритм CART (***Classification and Regression Tree***) очевидно решает задачи классификации и регрессии.

Разработан в 1974-1984 годах четырьмя профессорами статистики - *Leo* Breiman (Berkeley), Jerry Friedman (Stanford), Charles Stone (Berkeley) и Richard Olshen (Stanford).

Атрибуты набора данных могут иметь как дискретное, так и числовое значение.

Алгоритм *CART* предназначен для построения *бинарного дерева решений*.

Другие особенности алгоритма *CART*:

- функция оценки качества разбиения;
- механизм отсечения дерева;
- алгоритм обработки пропущенных значений;
- построение деревьев регрессии.

Алгоритм CART

Функция оценки качества разбиения, которая используется для выбора оптимального *правила*, - индекс *Gini* . Данная оценочная функция основана на идее уменьшения неопределенности в узле:

$$gini(T) = 1 - \sum_{j=1}^{n} p_j^2$$

где T - текущий узел, p_j - вероятность класса j в узле T, n - количество классов.

Допустим, есть узел, и он разбит на два класса. Максимальная неопределенность в узле будет достигнута при разбиении его на два подмножества по 50 примеров, а максимальная определенность - при разбиении на 100 и 0 примеров.

Правила разбиения. В каждом узле разбиение может идти только по одному атрибуту. Если атрибут является числовым, то во *внутреннем узле* формируется *правило* вида *xi* <= *c*.

Значение *c* в большинстве случаев - среднее арифметическое двух соседних упорядоченных значений переменной *xi* обучающего набора данных.

Если же атрибут относится к категориальному типу, то во внутреннем узле формируется правило xi V(xi), где V(xi) - некоторое непустое подмножество множества значений переменной xi в обучающем наборе данных.

Алгоритм CART

Механизм отсечения - *minimal cost-complexity tree pruning*, алгоритм *CART* принципиально отличается от других алгоритмов конструирования деревьев решений.

В рассматриваемом алгоритме отсечение - это компромисс между получением дерева "подходящего размера" и получением наиболее точной оценки классификации.

Метод заключается в получении последовательности уменьшающихся деревьев, но деревья рассматриваются не все, а только "лучшие представители".

Перекрестная проверка (*V-fold cross-validation*) является наиболее сложной и одновременно оригинальной частью алгоритма *CART* - путь выбора окончательного дерева, при условии, что набор данных имеет небольшой объем или же записи набора данных настолько специфические, что разделить набор на обучающую и тестовую выборку не представляется возможным.

Алгоритм C4.5 разработан **Джоном Квинланом** (*John Ross Quinlan*).
C4.5 является усовершенствованной версией алгоритма ID3 того же автора.

Алгоритм C4.5 строит дерево решений с неограниченным количеством *ветвей* у узла. Данный алгоритм может работать только с дискретным зависимым атрибутом и поэтому может решать только задачи классификации.

C4.5 считается одним из самых известных и широко используемых алгоритмов построения деревьев классификации.

Для работы алгоритма C4.5 необходимо соблюдение следующих требований:

- Каждая запись набора данных должна быть ассоциирована с одним из предопределенных классов, т.е. один из атрибутов набора данных должен являться меткой класса.
- Классы должны быть дискретными. Каждый пример должен однозначно относиться к одному из классов.
- Количество классов должно быть значительно меньше количества записей в исследуемом наборе данных.

Версия алгоритма - алгоритм C4.8 - реализована в инструменте Weka как J4.8 (Java). Коммерческая реализация метода: C5.0, разработчик RuleQuest, Австралия.

Алгоритм C4.5 медленно работает на сверхбольших и зашумленных наборах данных.

Алгоритм (C5.0) автоматизированного построения дерева решений

Фактически алгоритм C5.0 представляет собой стандарт процедуры построения деревьев решений.

Эта программа реализуется на коммерческой основе (http://www.rulequest.com/), но версия, встроенная в пакет R (и некоторые другие пакеты) доступны бесплатно.

Выбор признака, по которому будет осуществляться разбиение: ищем такой признак (для построения разбиения по нему), который позволил бы получить как можно более чистые группы.

Группы, полностью состоящие из элементов одного класса называют «***чистыми***» («***pure***»).

Для измерения степени чистоты группы существует несколько способов. Алгоритм C5.0 использует в качестве меры чистоты группы понятие ***энтропии***:

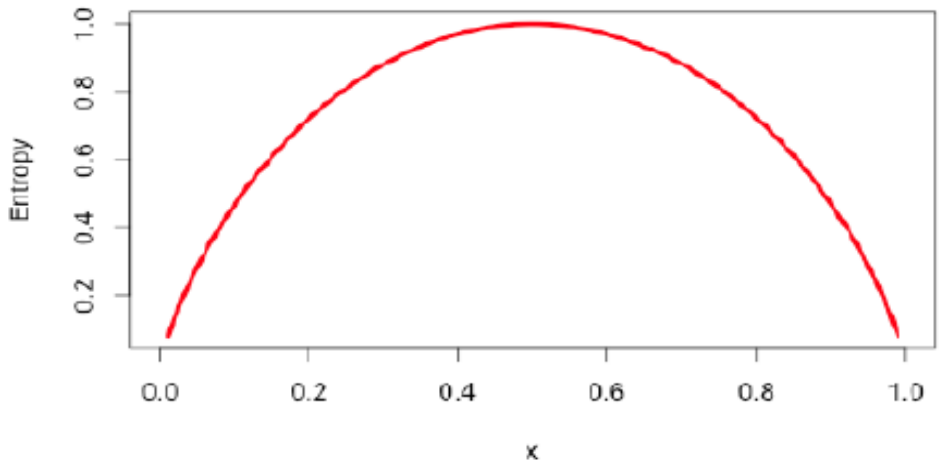
$$\text{Entropy}(S) = \sum_{i=1}^{c} -p_i \log_2(p_i)$$

- для системы, допускающей c возможных состояний.

Здесь pi – вероятность нахождения системы в состоянии i.

Энтропия как мера чистоты групп

Если у системы всего 2 возможных состояния, то её энтропия – функция одной переменной *p* , график которой имеет вид:




```
> curve(-x * log2(x) - (1 - x) * log2(1 - x),
        col="red", xlab = "x", ylab = "Entropy", lwd=4)
```

Энтропия максимальна, когда *p*=0,5. Нетрудно доказать аналитически, что для любого числа состояний максимум энтропии достигается в том случае, когда все они равновероятны.

Нулевая энтропия означает отсутствие неопределённости (применительно к деревьям решений это означает, что группа является полностью однородной, т.е. чистой).

Единичная энтропия означает полную неопределённость (в нашем случае – неопределённость относительно состава группы).

Алгоритм (C5.0) автоматизированного построения дерева решений

Алгоритм может выбрать тот признак, разбиение по которому даст самую чистую группу (т.е. группу, имеющую наименьшую энтропию). Эти вычисления называются «*information gain*» (буквально «усиление информации»).

Этот признак определяется методом перебора. Для каждого признака F («feature» – признак, свойство, характеристика) значение *information gain* вычисляется как разность энтропий группы до разбиения и после него:

$$\mathrm{InfoGain}(F) = \mathrm{Entropy}(S_1) - \mathrm{Entropy}(S_2)$$

Здесь Entropy(S2) – суммарная энтропия групп, полученных в результате разбиения.

Чем больше значение *information gain* для выбранного признака F, тем лучше этот признак F подходит для того, чтобы провести по нему разбиение.
Если для выбранного признака F значение *information gain* = 0, это означает, что разбиение по этому признаку бесперспективно, т.к. оно не приведёт к снижению энтропии.
С другой стороны, максимально возможное значение величины *information gain* = величине энтропии до разбиения (энтропия после разбиения будет = 0, т.е. полученные в результате разбиения группы будут полностью однородными - чистыми).

«Обрезка» дерева решений

Может возникнуть ситуация, когда группы окажутся слишком мелкими, а точек ветвления будет слишком много – в этом случае говорят, что модель «*is overfitted*», т.е. переопределена.

Пользоваться такими деревьями решений на практике бывает неудобно. Чтобы избежать этого, осуществляют так называемую «обрезку» (***pruning***) дерева решений. Результат «обрезки» - уменьшение размера дерева решений.

«***Pre-pruning***» - «*предварительная обрезка*»: состоит в том, чтобы *заранее* условиться о том, что процесс ветвления будет остановлен по достижении заданного числа решений. Как любой метод без обратной связи, он чреват тем, что будут пропущены важные, но трудно обнаруживаемые данные.

«***Post-pruning***» - «последующая обрезка»: сначала строится всё дерево решений, а потом по определённым правилам производится его «обрезка», т.е. отсечение лишних ветвей. Этот подход считается более надёжным - он гарантирует, что никакая важная информация не будет потеряна.

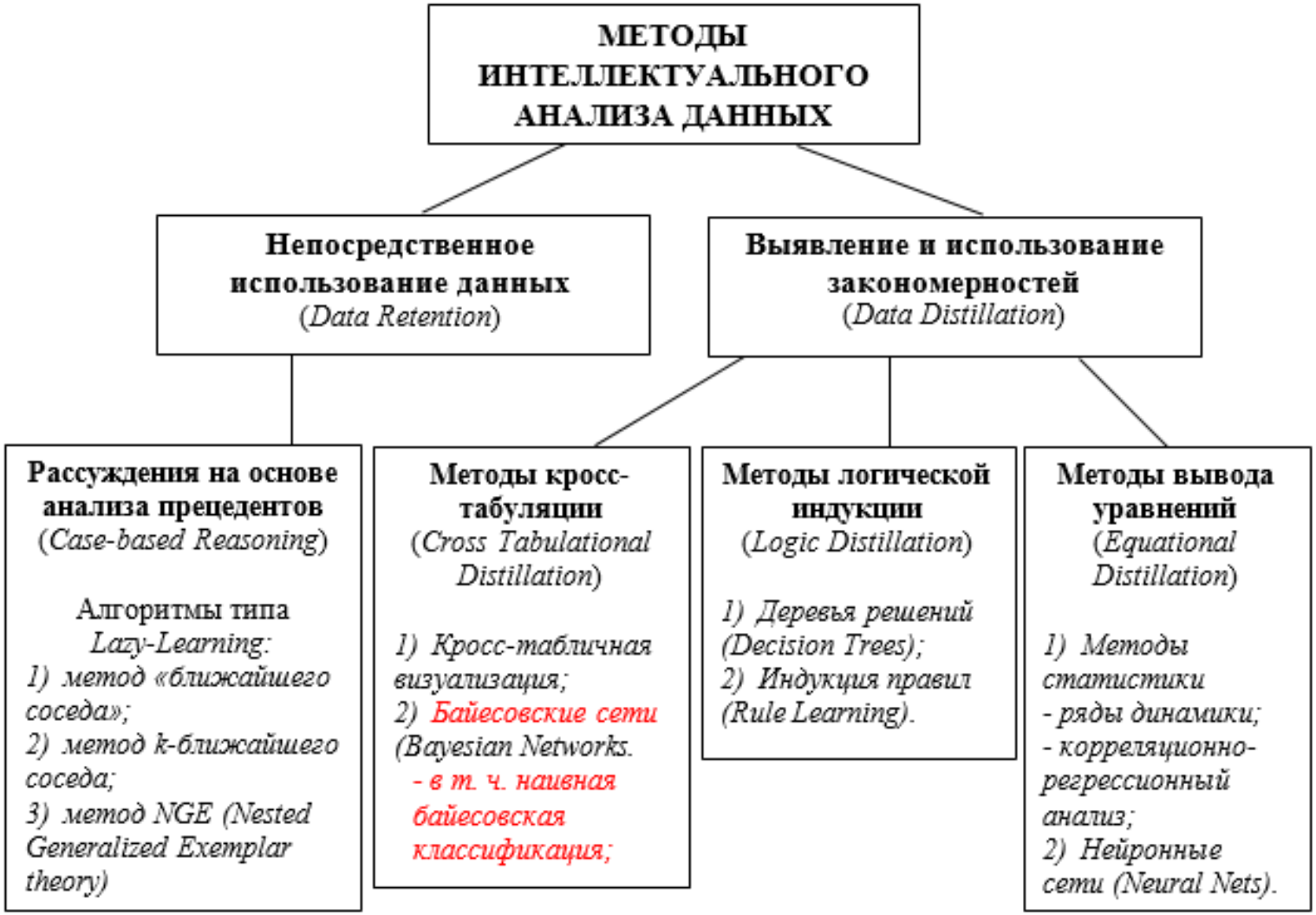
Иногда вместо того, чтобы «отсекать» ветви, их переносят выше по дереву или же заменяют более простыми - «subtree raising» («подъём поддерева») и «subtree replacement» («замена поддерева»).

Сильные стороны алгоритма C5.0	Слабые стороны алгоритма C5.0
Алгоритм C5.0 является универсальным, хорошо решает задачи классификации из разных областей.	Алгоритм C5.0 «тяготеет» к разбиениям по признакам с большим количеством уровней (Decision tree models are often biased toward splits on features having a large number of levels)
Алгоритм C5.0 может применяться для анализа не только числовых, но и номинальных данных; обеспечивает обработку пропущенных данных.	Модель (т.е. дерево решений) может оказаться как недоопределённой, так и переопредёлённой (It is easy to overfit or underfit the model)
Алгоритм C5.0 для построения дерева решений использует только самые важные признаки объектов (выбирает из множества факторов только те, которые сильно влияют на результат классификации).	Неточности классификации могут возникнуть из-за того, что используются только «параллельные осям» разбиения (axis-parallel splits)
Алгоритм C5.0 требует относительно небольшого объёма обучающей выборки.	Алгоритм C5.0 очень чувствителен к обучающей выборке – даже небольшие изменения в ней могут сильно повлиять на результат.
Интерпретация результатов работы алгоритма C5.0 не требует специальной математической подготовки.	Деревья решений иногда получаются очень большими, что затрудняет их интуитивное понимание.
Алгоритм C5.0 считается более эффективным чем другие алгоритмы классификации.	

Лекция 5

БАЙЕСОВСКИЕ МЕТОДЫ КЛАССИФИКАЦИИ

Байесовские методы. Байесовская классификация



Классификация технологических методов ИАД [1]

Байесовские методы. Байесовская классификация

Суть методов кросс-табуляции

Кросс-табуляция является простой формой анализа, широко используемой в генерации отчетов средствами систем оперативной аналитической обработки (OLAP).

Двумерная кросс-таблица представляет собой матрицу значений, каждая ячейка которой лежит на пересечении значений атрибутов.

Расширение идеи кросс-табличного представления на случай гиперкубической информационной модели является основой многомерного анализа данных, поэтому эта группа методов рассматривается как симбиоз многомерного оперативного анализа и интеллектуального анализа данных.

К методам ИАД группы кросс-табуляции относится также использование байесовских сетей (***Bayesian Networks***).

Байесовские методы. Байесовские сети

Байесовская сеть (или байесова сеть, байесовская сеть доверия) - ***графическая вероятностная модель, представляющая собой множество переменных и их вероятностных зависимостей.***

Математический аппарат байесовых сетей создан американским ученым *Джудой Перлом*, лауреатом Премии Тьюринга (2011 г.).

Формально: ***байесовская сеть*** - это направленный ациклический граф, каждой вершине которого соответствует случайная переменная, а дуги графа кодируют отношения условной независимости между этими переменными.

Вершины могут представлять переменные любых типов, быть взвешенными параметрами, скрытыми переменными или гипотезами.

Байесовские методы. Байесовские сети

Байесовская сеть - это полная модель для переменных и их отношений

Следовательно, она может быть использована для того, чтобы давать ответы на вероятностные вопросы.

Этот процесс вычисления апостериорного распределения переменных по переменным-свидетельствам называют ***вероятностным выводом***, что дает универсальную оценку для приложений, где нужно выбрать значения подмножества переменных, которое минимизирует функцию потерь, например, вероятность ошибочного решения.

Байесовская сеть позволяет получить ответы на следующие типы вероятностных запросов:

1) нахождение вероятности свидетельства,

2) определение априорных маргинальных вероятностей,

3) определение апостериорных маргинальных вероятностей, включая:

- *прогнозирование*, или прямой вывод - определение вероятности события при наблюдаемых причинах),
- *диагностирование*, или обратный вывод (абдукция), - определение вероятности причины при наблюдаемых следствиях,
- *межпричинный* (смешанный) *вывод* или трансдукция, - определение вероятности одной из причин наступившего события при условии наступления одной или нескольких других причин этого события.

4) вычисление наиболее вероятного объяснения наблюдаемого события,

5) вычисление апостериорного максимума.

Байесовские методы. Байесовская классификация

- Ранее байесовская классификация использовалась для формализации знаний экспертов в экспертных системах.
- В настоящее время байесовская классификация применяется и в качестве одного из методов Data Mining.
- Байесовские методы получили достаточно широкое распространение и активно используются в самых различных областях знаний.

История вопроса: *формула Байеса* была опубликована в 1763 году спустя 2 года после смерти *Томаса Байеса*.
Методы, использующие ее, получили широкое распространение только к концу 20 века - расчеты требуют определенных вычислительных затрат, и они стали возможны лишь с развитием информационных технологий.

Байесовские методы. Байесовская классификация

Байесовский метод опирается на теорему о том, что если плотности распределения классов известны, то алгоритм классификации, имеющий минимальную вероятность ошибок, можно выписать в явном виде.

Для оценивания плотностей классов по выборке применяются различные подходы (в частности, параметрический, непараметрический и оценивание смесей распределений).

В ***байесовских классификаторах*** используется *критерий, минимизирующий вероятность принятия ошибочного решения*, поэтому байесовские алгоритмы являются статистически оптимальными.

Но при этом алгоритмы требуют в идеале полного знания многомерных функций распределения наблюдаемых признаков для каждого класса. Необходимость такого знания обусловлена использованием ***формулы Байеса***, которая лежит в основе ***байесовских методов*** принятия решения.

Байесовский подход основан на предположении, что задача выбора решения сформулирована в терминах теории вероятностей и известны все представляющие интерес вероятностные величины. В основе байесовской классификации лежит ***правило Байеса.***

Байесовские методы. Байесовская классификация

Теорема Байеса позволяет рассчитать апостериорную вероятность $P(c|x)$ на основе $P(c)$, $P(x)$ и $P(x|c)$:

$$P(c|x) = \frac{P(x|c) \cdot P(c)}{P(x)}$$

$P(c|x)$ – апостериорная вероятность данного класса c

(т.е. данного значения целевой переменной) при данном значении признака x.

$P(c)$ – априорная вероятность данного класса.

$P(x|c)$ – правдоподобие, т.е. вероятность данного значения признака при данном классе.

$P(x)$ – априорная вероятность данного значения признака.

Байесовская классификация. Постановка задачи

Рассмотрим обучающую выборку из n объектов, каждый из которых принадлежит одному из K классов и характеризуется набором m числовых признаков a_1, a_2,... a_m.

Пусть имеется n_k объектов k-ого класса, так что

$$N=\Sigma n_k K_k=1.$$

Значение j-ого признака i-ого объекта из k-ого класса обозначим x_{ijk}.

Тогда этот объект можно охарактеризовать вектором-строкой $x_{ik}=(x_{i1k},...,x_{ijk},...,x_{imk})$. Эту строку будем рассматривать как i-ю реализацию векторной случайной величины ξ_k, подчиняющейся распределению вероятностей с плотностью $p(x_1,...,x_m|k)$, своей для каждого класса k.

Пусть теперь наблюдается объект, для которого необходимо определить, к какому классу он относится. Объект характеризуется только набором m числовых признаков $x_1,...,x_m$.

Байесовская классификация. Общая структура байесовского классификатора

В основе классификатора лежит следующее ***правило***. Классификатор вычисляет ***апостериорную вероятность*** $P(k|x)$ каждого класса k, которому может принадлежать испытуемый объект, и относит этот объект к апостериорно наиболее вероятному классу ˆ:

$$\hat{k} = \arg\max_k \ln P(k|x_1, \dots, x_m)$$

Апостериорная вероятность вычисляется по ***формуле Байеса***:

$$P(k|x_1,\dots,x_m)=P(k)p(x_1,\dots,x_m|k)/p(k),$$

где $P(k)$ – априорная вероятность того, что объект относится к k-ому классу, $p(k)$ и $p(x_1,\dots,x_m|k)$ - безусловная и условная многомерные плотности распределения вектора признаков, компоненты которого обычно статистически зависимы.

Таким образом, байесовский классификатор предполагает, что многомерная совместная плотность распределения признаков известна для всех классов.

Байесовская классификация. Общая структура байесовского классификатора

Аналитическое представление многомерной плотности вероятности известно только для нормального распределения.

При этом многомерная нормальная плотность распределения дает подходящую модель для одного важного случая, а именно когда значения векторов признаков x для данного класса k представляются непрерывнозначными, слегка искаженными версиями единственного типичного вектора, или вектора-прототипа, μ_k.

Именно этого ожидают, когда классификатор выбирается так, чтобы выделять те признаки, которые, будучи различными для образов, принадлежащих различным классам, были бы, возможно, более схожи для образов из одного и того же класса.

Байесовская классификация. Общая структура байесовского классификатора

Многомерная нормальная плотность распределения в общем виде представляется выражением

$$p(x) = \frac{1}{(2\pi)^{\frac{m}{2}} \det R^{\frac{1}{2}}} e^{-\frac{1}{2}(x-\mu)^T R^{-1}(x-\mu)}$$

где μ – m-компонентный вектор среднего значения,
R – ковариационная матрица размера $m \times m$,
T – знак транспонирования.

Если все недиагональные элементы равны нулю, то $p(x)$ сводится к произведению одномерных нормальных плотностей компонент вектора x).

Байесовская классификация. Общая структура байесовского классификатора

Для многомерного нормального распределения удаётся выразить в аналитически замкнутой форме (с точностью до несущественных слагаемых) алгоритм байесовской классификации:

$$\hat{k} = \arg\max_k \left(\ln P(k) - \frac{1}{2} \ln \det R_k - \frac{1}{2} (x_k - \mu_k) R_k^{-1} (x_k - \mu_k)^T \right)$$

где μ_k – m-вектор-строка математических ожиданий значений признаков объектов класса k,
R_k – $m \times m$-матрица ковариаций векторов признаков класса k.

Диагональные элементы матрицы образуют m-вектор D_k дисперсий признаков объектов класса k.

Алгоритм байесовской классификации с обучением состоит из двух этапов:
1) этап обучения;
2) этап классификации

Байесовские методы. Байесовская классификация

Достоинства байесовских сетей как метода Data Mining

- в модели определяются зависимости между всеми переменными, это позволяет легко обрабатывать ситуации, в которых значения некоторых переменных неизвестны;
- байесовские сети достаточно просто интерпретируются и позволяют на этапе прогностического моделирования легко проводить анализ по сценарию «что, если»;
- байесовский метод позволяет естественным образом совмещать закономерности, выведенные из данных, и, например, экспертные знания, полученные в явном виде;
- использование байесовских сетей позволяет избежать проблемы переучивания (overfitting), то есть избыточного усложнения модели, что является слабой стороной многих методов (например, деревьев решений и нейронных сетей).

Байесовские методы. Наивно-байесовский подход (Naive-Bayes Approach)

Суть метода наивно-байесовской классификации

В наивном байесовском классификаторе делается предположение о независимости признаков объекта. Если пренебречь статистическими связями между компонентами вектора признаков, тогда матрица R_k будет диагональной с вектором D_k на главной диагонали и классификатор станет ***наивным байесовским классификатором***.

Также предполагается, что ***маргинальная плотность распределения*** $p(xj|k)$ любого признака является нормальной для любого класса.

Но на практике так бывает далеко не всегда, то есть наблюдаемые данные не подчиняются нормальному закону распределения (в общем случае закон вообще неизвестен) и имеет место статистическая зависимость, поэтому область применения классификатора сужается.

Байесовские методы. Наивно-байесовский подход (Naive-Bayes Approach)

Наивно-байесовский подход имеет следующие недостатки:

- перемножать условные вероятности корректно только тогда, когда все входные переменные действительно статистически независимы;
- хотя часто данный метод показывает достаточно хорошие результаты при несоблюдении условия статистической независимости, но теоретически такая ситуация должна обрабатываться более сложными методами, основанными на обучении байесовских сетей;
- невозможна непосредственная обработка непрерывных переменных - требуется их преобразование к интервальной шкале, чтобы атрибуты были дискретными; однако такие преобразования иногда могут приводить к потере значимых закономерностей;
- на результат классификации в наивно-байесовском подходе влияют только индивидуальные значения входных переменных, комбинированное влияние пар или троек значений разных атрибутов здесь не учитывается.

Оптимальный байесовский классификатор

Так как ***оптимальный байесовский классификатор*** является модификацией наивного байесовского классификатора, то в качестве решающего правила также берётся рассмотренная ранее формула.

Одна из идей оптимизации наивно-байесовского классификатора состоит в том, чтобы, максимально используя обучающую выборку и гауссову копула-функцию, обойти два «наивных» предположения.

Модификация позволяет:

1) учесть статистические связи между наблюдаемыми признаками;

2) адаптировать классификатор к неизвестному действительному распределению путем приведения сглаженных маргинальных функций распределения признаков к нормальному виду.

Другими словами, с помощью нелинейных гауссовых копула-функций негауссовы данные преобразуются в гауссовы, которые можно подавать на вход классификатору.

Литература

1. Л. В. Щавелёв Способы аналитической обработки данных для поддержки принятия решений. (СУБД. - 1998. - № 4-5)
2. Воронцов К.В. Математические методы обучения по прецедентам (теория обучения машин) URL: http://www.ccas.ru/voron
3. Chickering D, Geiger D., Heckerman D. Learning Bayesian networks: The combination of knowledge and statistical data / Machine Learning. 1995. 20. P. 197-243
4. Heckerman D Bayesian Networks for Data Mining / Data Mining and Knowledge Discovery. 1997. № 1. P. 79-119
5. etc, Friedman N., Geiger D., Goldszmidt M. Bayesian Network Classifiers / Machine Learning. 1997. 29. P. 131-165
6. Brand E., Gerritsen R / Naive-Bayes and Nearest Neighbor DBMS. 1998. № 7