# Энтропия и деревья принятия решений

Деревья принятия решений являются удобным инструментом в тех случаях, когда требуется не просто классифицировать данные, но ещё и объяснить почему тот или иной объект отнесён к какому-либо классу.

Давайте сначала, для полноты картины, рассмотрим природу энтропии и некоторые её свойства. Затем, на простом примере, увидим каким образом использование энтропии помогает при создании классификаторов. После чего, в общих чертах сформулируем алгоритм построения дерева принятия решений и его особенности.

**Комбинаторная энтропия**

Рассмотрим множество разноцветных шариков: *2* красных, *5* зеленых и *3* желтых. Перемешаем их и расположим в ряд. Назовём эту операцию *перестановкой*:



Давайте посчитаем количество различных перестановок, учитывая, что шарики одного цвета — неразличимы. Если бы каждый шарик имел уникальный цвет, то количество перестановок было бы *10!*, но если два шарика одинакового цвета поменять местами — новой перестановки не получится. Таким образом, нужно исключить *5!* перестановок зеленых шариков между собой (а также, *3!* желтых и *2!* красных). Поэтому, в данном случае, решением будет:

https://habrastorage.org/r/w1560/storage2/716/2a4/29f/7162a429fe60a31ddd45a96b91c34c33.png

[Мультиномиальний коэффициент](http://en.wikipedia.org/wiki/Multinomial_theorem) позволяет рассчитать количество перестановок в общем случае данной задачи:

https://habrastorage.org/r/w1560/storage2/114/180/e94/114180e94f0d212e4cf2361c12a7832a.png

(*Ni* — количество одинаковых шариков каждого цвета).

Все перестановки можно пронумеровать числами от *0* до *(W — 1)*. Следовательно, строка из *log2(W)* бит однозначно кодирует каждую из перестановок.

Поскольку перестановка состоит из *N* шариков, то среднее количество бит, приходящихся на один элемент перестановки можно выразить как:

https://habrastorage.org/r/w1560/storage2/671/11d/f5c/67111df5c3f76c9b4d3039454a2539f3.png

Эта величина называется *комбинаторной энтропией*:

https://habrastorage.org/r/w1560/storage2/e7d/576/199/e7d5761997da90a8c6273cb7375a4fd3.png

Чем более однородно множество (преобладают шарики какого-то одного цвета) — тем меньше его комбинаторная энтропия, и наоборот — чем больше различных элементов в множестве, тем выше его энтропия.

**Энтропия Шеннона**

Давайте рассмотрим подробнее описанное выше выражение для энтропии:

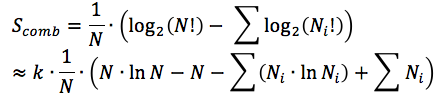
https://habrastorage.org/r/w1560/storage2/0db/aee/594/0dbaee594c66c3f006f439a03e91a5fc.png

Учитывая свойства логарифмов, преобразуем формулу следующим образом:

https://habrastorage.org/r/w1560/storage2/d45/12a/806/d4512a806eb37b3b851b39bcd0fe22f5.pngПредположим, что количество шариков достаточно велико для того чтобы воспользоваться [формулой Стирлинга](http://ru.wikipedia.org/wiki/%D0%A4%D0%BE%D1%80%D0%BC%D1%83%D0%BB%D0%B0_%D0%A1%D1%82%D0%B8%D1%80%D0%BB%D0%B8%D0%BD%D0%B3%D0%B0):

https://habrastorage.org/r/w1560/storage2/f6f/695/5cb/f6f6955cb871479409defcfe958f4407.png

Применив формулу Стирлинга, получаем:

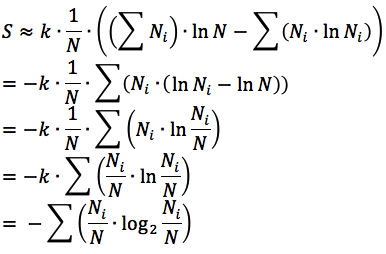
¶

,где *k* — коэффициент перехода к натуральным логарифмам.

Учитывая что

https://habrastorage.org/r/w1560/storage2/a4f/a73/0ce/a4fa730ce84b863fec45039eb3f4c019.png

выражение можно преобразовать:



Поскольку общее количество шариков *N*, а количество шариков *i-*го цвета — *Ni*, то вероятность того, что случайно выбранный шарик будет именно этого цвета является:

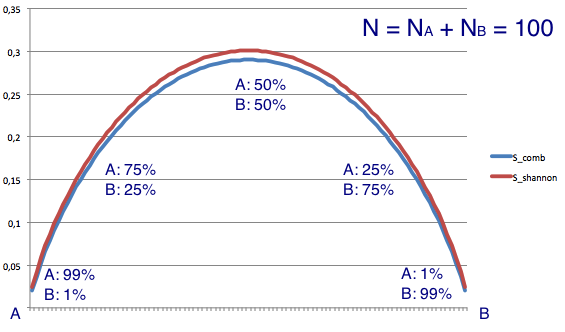
https://habrastorage.org/r/w1560/storage2/20e/08e/97f/20e08e97f3347c954162c867e9777a1e.png.

Исходя из этого, формула для энтропии примет вид:

https://habrastorage.org/r/w1560/storage2/37f/edc/8ae/37fedc8ae95e569960e7a8c9d6e1a8a8.png

Данное выражение является *энтропией Шенонна*.¶¶При более тщательном выводе можно показать, что энтропия Шенонна является пределом для комбинаторной энтропии, поэтому её значение всегда несколько больше значения комбинаторной энтропии.

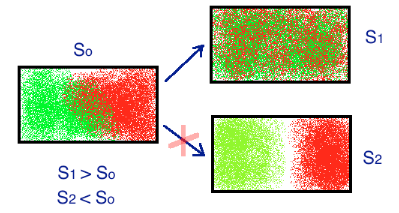
Сравнение двух энтропий представлено на следующем рисунке, который рассчитан для множеств, содержащих два типа объектов — *А* и *В* (суммарное количество элементов в каждом множестве — 100):



**Термодинамика**

¶Аналогичные выражения для энтропии можно получить в термодинамике:

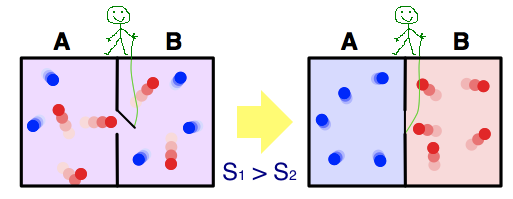
* Исходя из микроскопических свойств веществ: опираясь на постулаты статистической термодинамики (в данном случае оперируют неразличимыми частицами, которые находятся в разных энергетических состояниях).
* Исходя из макроскопических свойств веществ: путем анализа работы тепловых машин.



Понятие энтропии играет фундаментальную роль в термодинамике, фигурируя в формулировке [Второго начала термодинамики](http://ru.wikipedia.org/wiki/%D0%92%D1%82%D0%BE%D1%80%D0%BE%D0%B5_%D0%BD%D0%B0%D1%87%D0%B0%D0%BB%D0%BE_%D1%82%D0%B5%D1%80%D0%BC%D0%BE%D0%B4%D0%B8%D0%BD%D0%B0%D0%BC%D0%B8%D0%BA%D0%B8): *если изолированная макроскопическая система находится в неравновесном состоянии, то наиболее вероятным является её самочинный переход в состояние с большим значением энтропии*:

**Демон Максвелла**

Чтобы подчеркнуть статистическую природу Второго начала термодинамики в 1867 году Джеймс Максвелл предложил мысленный эксперимент: «Представим сосуд, заполненный газом определённой температуры, сосуд разделен перегородкой с заслонкой, которую демон открывает чтобы пропускать быстрые частицы в одну сторону, а медленные — в другую. Следовательно, спустя некоторое время, в одной части сосуда сконцентрируются быстрые частицы, а в другой — медленные. Таким образом, вопреки Второму началу термодинамики, *демон Максвелла может уменьшать энтропию замкнутой системы»*:



Позже, Лео Сциллард разрешил парадокс, но это обсуждение несколько выходит за рамки данной статьи.

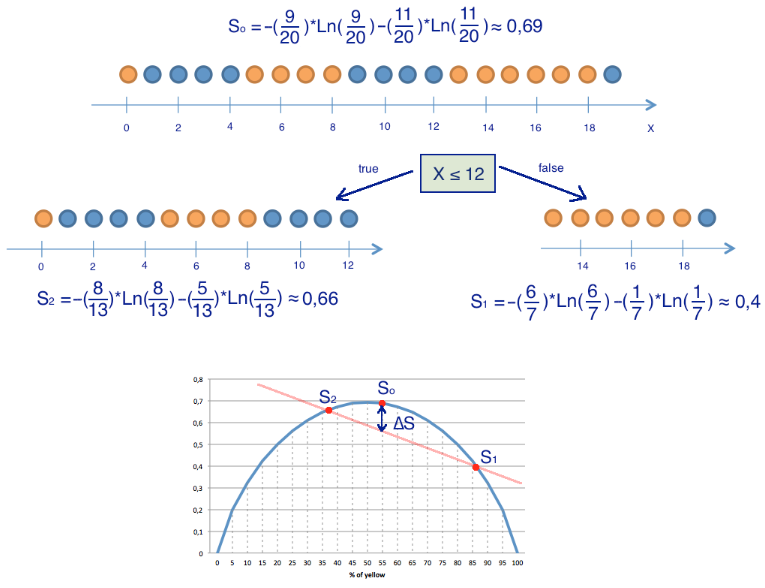
**Демон Максвелла == Классификатор**

Если вместо «быстрых» и «медленных» частиц рассматривать объекты, принадлежащие к различным классам, тогда демона Максвелла можно рассматривать в качестве своеобразного классификатора.

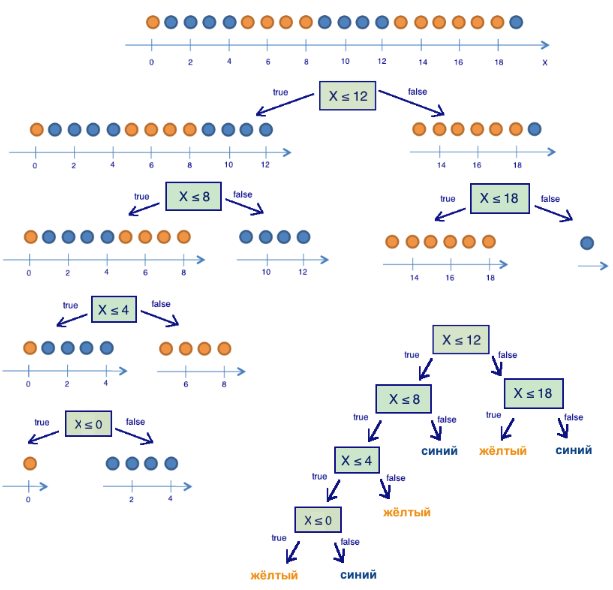
Сама формулировка парадокса подсказывает алгоритм обучения: *нужно находить правила (предикаты), на основе которых разбивать тренировочный набор данных, таким образом, чтобы уменьшалось среднее значение энтропии*. Процесс деления множества данных на части, приводящий к уменьшению энтропии, можно рассматривать как *производство информации*.

Разбив исходный набор данных на две части по некому предикату, можно рассчитать энтропию каждого подмножества, после чего рассчитать среднее значение энтропии — если оно окажется меньшим чем энтропия исходного множества, значит предикат содержит некую обобщающую информацию о данных.

Для примера, рассмотрим множество двухцветных шариков, в котором цвет шарика зависит только от координаты *х*:¶(из практических соображений, при расчётах удобно использовать энтропию Шеннона)



Из рисунка видно что если разделить множество на две части, при условии что одна часть будет содержать все элементы с координатой *х ≤ 12*, а другая часть — все элементы, у которых *х > 12*, то средняя энтропия будет меньше исходной на *∆S*. Это значит, что данный предикат обобщает некоторую информацию о данных (легко заметить, что при *х > 12* — почти все шарики жёлтые).Если использовать относительно простые предикаты («больше», «меньше», «равно» и т.п.) — то, скорее всего, одного правила будет недостаточно для создания полноценного классификатора. Но процедуру поиска предикатов можно повторять рекурсивно для каждого подмножества. Критерием остановки является нулевое (или очень маленькое) значение энтропии. В результате получается древовидный набор условий, который называется *Деревом принятия решений*:



Листьями дерева принятия решений являются классы. Чтобы классифицировать объект при помощи дерева принятия решений — нужно последовательно спускаться по дереву (выбирая направление основываясь на значениях предикатов применяемых к классифицируемому объекту). Путь от корня дерева до листьев можно трактовать как объяснение того, почему тот или иной объект отнесён к какому-либо классу.

В рассмотренном примере, для упрощения, все объекты характеризуются только одним атрибутом — координатой *х*, но точно такой же подход можно применить и к объектам со множественными атрибутами.

Также, не накладывается ограничений на значения атрибутов объекта — они могут иметь как категориальную, так и числовую или логическую природу. Нужно только определить предикаты, которые умеют правильно обрабатывать значения атрибутов (например, вряд ли есть смысл использовать предикаты «больше» или «меньше» для атрибутов с логическими значениями).

**Алгоритм построения дерева принятия решений**

В общих чертах, алгоритм построения дерева принятия решений можно описать следующим образом:¶(мне кажется, что алгоритм описанный «человеческим языком» легче для восприятия)

*s0 = вычисляем энтропию исходного множества*

*Если s0 == 0 значит:*

*Все объекты исходного набора, принадлежат к одному классу*

*Сохраняем этот класс в качестве листа дерева*

*Если s0 != 0 значит:*

*Ищем предикат, который разбивает исходное множество таким образом, чтобы уменьшилось среднее значение энтропии*

*Найденный предикат является частью дерева принятия решений, сохраняем его*

*Разбиваем исходное множество на подмножества, согласно предикату*

*Повторяем данную процедуру рекурсивно для каждого подмножества*

Что значит «*ищем предикат*»? Как вариант, можно считать, что на основе каждого элемента исходного множества можно построить предикат, который разбивает множество на две части. Следовательно, алгоритм можно переформулировать:

*s0 = вычисляем энтропию исходного множества*

*Если s0 == 0 значит:*

*Все объекты исходного набора, принадлежат к одному классу*

*Сохраняем этот класс в качестве листа дерева*

*Если s0 != 0 значит:*

*Перебираем все элементы исходного множества:*

*На основе каждого элемента генерируем предикат, который разбивает исходное множество на два подмножества*

*Рассчитываем среднее значение энтропии*

*Вычисляем ∆S*

*Нас интересует предикат, с наибольшим значением ∆S*

*Найденный предикат является частью дерева принятия решений, сохраняем его*

*Разбиваем исходное множество на подмножества, согласно предикату*

*Повторяем данную процедуру рекурсивно для каждого подмножества*

Как можно *«на основе каждого элемента множества генерировать предикат»*?

В самом простом случае, можно использовать предикаты, которые относятся только к значению какого-нибудь атрибута (например, *«x ≥ 12»*, или *«цвет == жёлтый»* и т.п.). Следовательно, алгоритм примет вид:

*s0 = вычисляем энтропию исходного множества*

*Если s0 == 0 значит:*

*Все объекты исходного набора, принадлежат к одному классу*

*Сохраняем этот класс в качестве листа дерева*

*Если s0 != 0 значит:*

*Перебираем все элементы исходного множества:*

*Для каждого элемента перебираем все его атрибуты:*

*На основе каждого атрибута генерируем предикат, который разбивает исходное множество на два подмножества*

*Рассчитываем среднее значение энтропии*

*Вычисляем ∆S*

*Нас интересует предикат, с наибольшим значением ∆S*

*Найденный предикат является частью дерева принятия решений, сохраняем его*

*Разбиваем исходное множество на подмножества, согласно предикату*

*Повторяем данную процедуру рекурсивно для каждого подмножества*

На самом деле, если рассматривать классифицируемые объекты как точки в многомерном пространстве, то можно увидеть, что предикаты, разделяющие множество данных на подмножества, являются гиперплоскостями, а процедура обучения классификатора является поиском ограничивающих объёмов (в общем, как и для любого другого вида классификаторов).

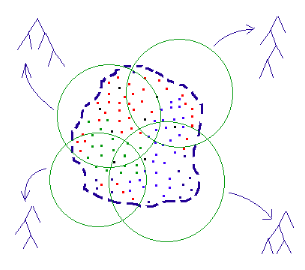
Главным достоинством является, получаемая в результате, древовидная структура предикатов, которая позволяет интерпретировать результаты классификации (хотя в силу своей «жадности», описанный алгоритм, не всегда позволяет обеспечить оптимальность дерева в целом).

Одним из краеугольных камней описанного алгоритма является критерий остановки при построении дерева. В описанных выше псевдокодах, я прекращал построение дерева только при достижении множества, в котором все элементы принадлежат к одному классу (*энтропия == 0*). Такой подход позволяет полностью подогнать дерево принятия решений под обучающую выборку данных, но это не всегда эффективно с практической точки зрения (полученное дерево является *переобученным*).

Одним из возможных критериев остановки может быть небольшое значение *∆S*. Но при таком подходе, всё же, невозможно дать универсальный совет: при каких значениях *∆S* следует прекращать построение дерева.

**Random forest**

Чтобы не заморачиваться над критерием остановки при построении дерева, можно поступить следующим образом: *выбирать случайные подмножества из обучающей выборки данных, и для каждого подмножества строить своё дерево принятия решений (в принципе, даже не важно какой критерий остановки будет использоваться)*:



Полученный в результате ансамбль деревьев (упрощённая версия [Random forest](http://ru.wikipedia.org/wiki/Random_forest)) можно использовать для классификации, прогоняя классифицируемый объект через все деревья. Каждое дерево как будто «голосует» за принадлежность объекта к определённому классу. Таким образом, на основе того, какая часть деревьев проголосовала за тот или иной класс — можно заключить с какой вероятностью объект принадлежит к какому-либо классу.

Данный метод позволяет достаточно адекватно обрабатывать пограничные области данных:

Можно заметить, что единичное дерево принятия решений описывает область, которая полностью содержит красные точки, в то время как ансамбль деревьев описывает фигуру, которая более близка к окружности.

**Если есть желание поэкспериментировать**

Я создал небольшое приложение, для сравнения дерева принятия решений и random forest. При каждом запуске приложения создаётся случайный набор данных, соответствующий красному кругу на зелёном фоне, а в результате выполнения приложения получается картинка, типа той, которая изображена выше.

* У Вас должна быть установлена [среда выполнения Java](http://www.java.com/en/download/help/download_options.xml)
* Загрузите бинарник *dec\_tree\_demo.jar*
* Для запуска приложения наберите в командной строке: java -jar dec\_tree\_demo.jar out.png

Исходники [есть](https://github.com/lagodiuk/decision-tree) на гитхабе.

**Вместо заключения**

Деревья принятия являются неплохой альтернативой, в тех случаях, когда надоедает подстраивать абстрактные веса и коэффициенты в других алгоритмах классификации, либо, когда приходится обрабатывать данные со смешанными (категориальными и числовыми) атрибутами.

**Дополнительная информация**

1. *Яцимирский В.К. Физическая химия* (здесь довольно неплохо описано понятие энтропии, а также рассматриваются некоторые философские аспекты данного феномена)
2. [Toby Segaran, Programming Collective Intelligence](http://shop.oreilly.com/product/9780596529321.do) (в данной книге есть глава посвящённая деревьям принятия решений, и вообще, если Вы ещё не читали этой книги — советую обязательно заглянуть туда :-)
3. Такие библиотеки как [Weka](http://en.wikipedia.org/wiki/Weka_%28machine_learning%29) и [Apache Mahout](http://en.wikipedia.org/wiki/Apache_Mahout) содержат реализацию деревьев принятия решений.