KARPOV.COURSES >>>
КОНСПЕКТ

Конспект > 17 урок > Решающее дерево: постановка задачи регрессии/классификации и гиперпараметры модели

> Введение в решающие деревья

Решающие деревья представляют собой невероятно мощное семейство моделей в машинном обучении. Они хоть и тесно связаны линейными моделями, таковыми интуитивно не являются.

Дадим мотивацию перед тем, как перейдем к основной концепции.

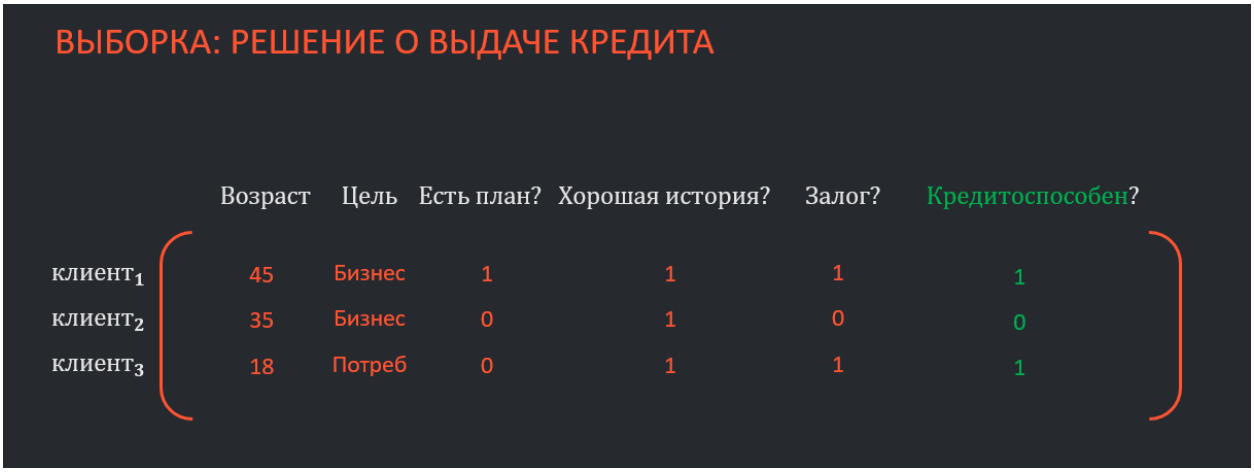
Как в реальном мире принимаются решения?

Зачастую через проверку некоторых логических правил, например:

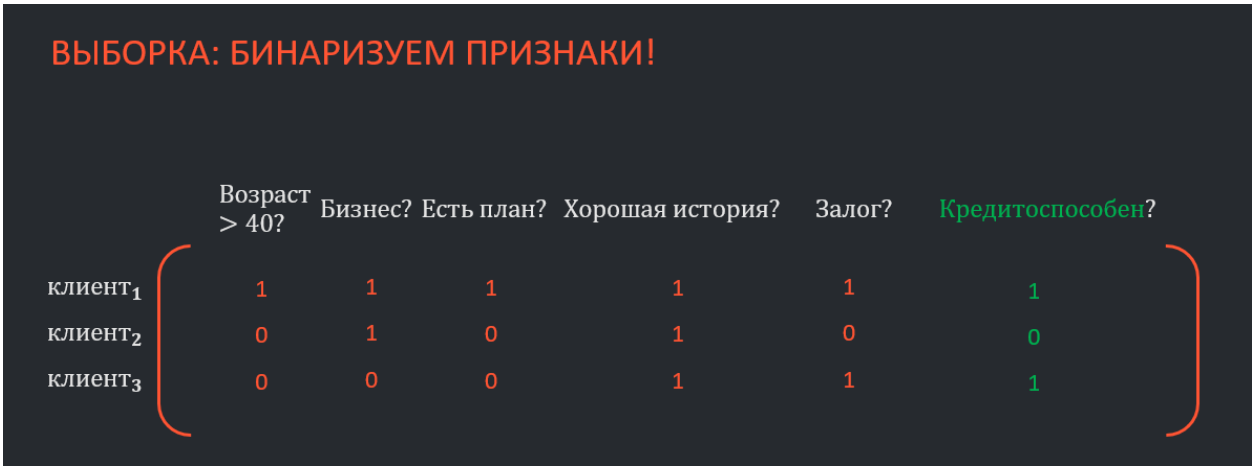
1. на приеме у доктора приходится отклонять или принимать симптоматику
2. во время экзамена успешно или безуспешно отвечать на ряд вопросов

Итоговое решение принимается сквозь призму реализовавшихся комбинаций установленных правил.

Давайте посмотрим на пример ниже, где мы пытаемся решить задачу классификации на примере некой банковской организации, которой нужно принять решение о выдаче кредита.



Следующим шагом на картинке ниже мы переводим признаки в бинарные. Теперь у нас каждый клиент описан некой комбинацией, некой цепочкой, реализацией логических признаков, т.е. таких, которые позволяют отвечать на вопрос «да» или «нет».



На основании цепочек полученных признаков можно попробовать установить между ними и таргетом зависимость. Например :

[Возраст > 40] & [Цель = Потреб] & [Хорошая история] --> Выдать кредит

[Возраст < 40] & [Цель = Бизнес] & [Нет бизнес-плана] --> Отказать

Такой алгоритм можно изобразить в виде дерева, как изображено на рисунке ниже:

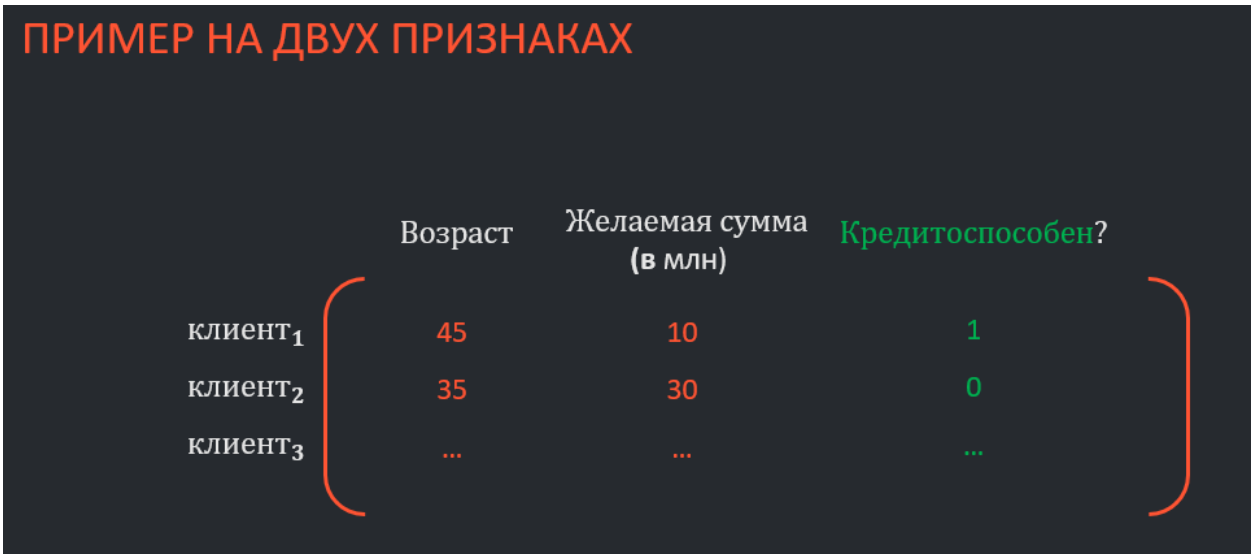
Корень дерева — это та самая верхушка дерева, то правило, с которого обычно начинается исследование объекта.

Внутренняя вершина (предикат) — проверяет правило.

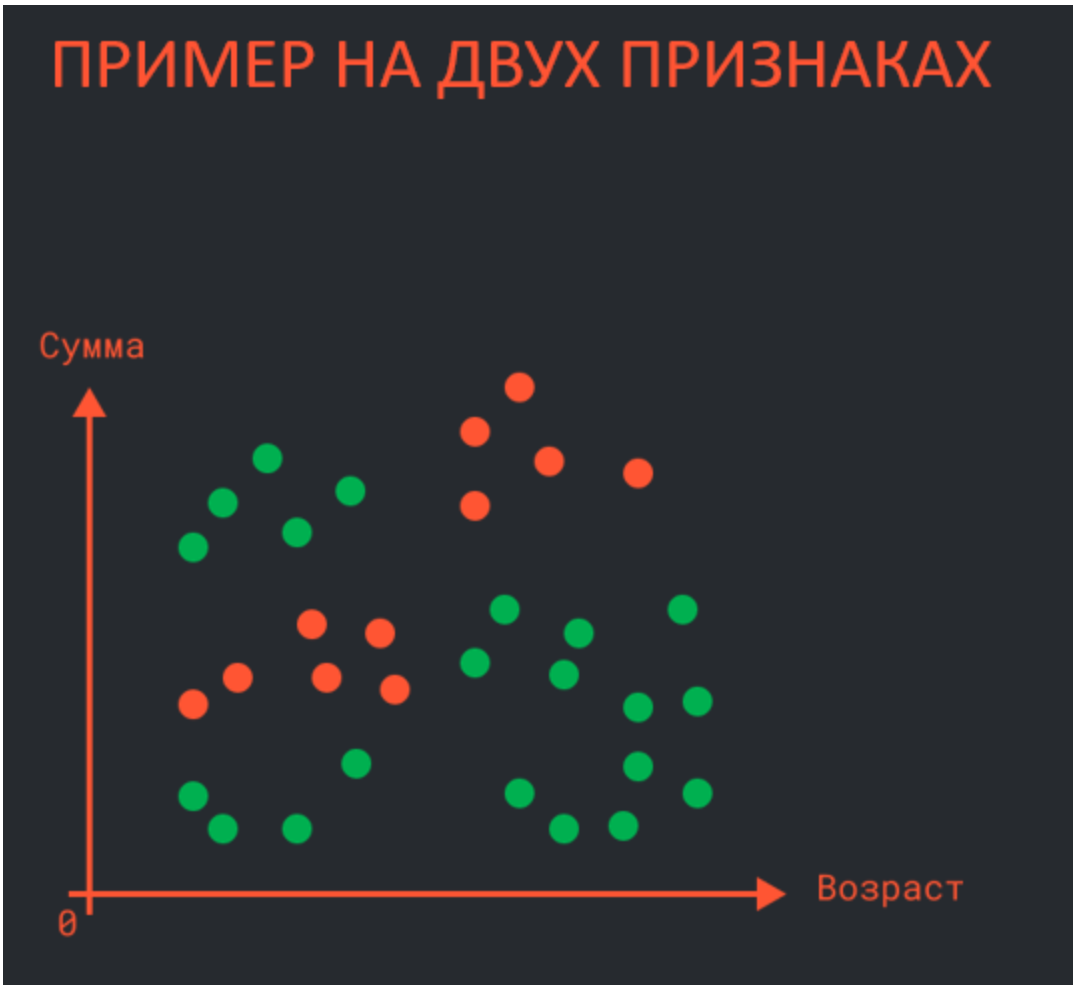
Листовая вершина (решение) — в ней находится решение.

Это дерево можно назвать бинарным, так как мы можем посмотреть на количество потомков каждой вершины и увидеть, что их число равняется двум, т.е. мы можем ответить да или нет, третьего варианта не дано.

Теперь давайте представим, что мы имеем модель, которая описывается деревом выше. Давайте посмотрим визуально на наше признаковое пространство, когда мы пытаемся построить некоторое дерево. Чтобы ответить на данный вопрос, давайте немного сузим наш пример до двух признаком на картинках ниже.

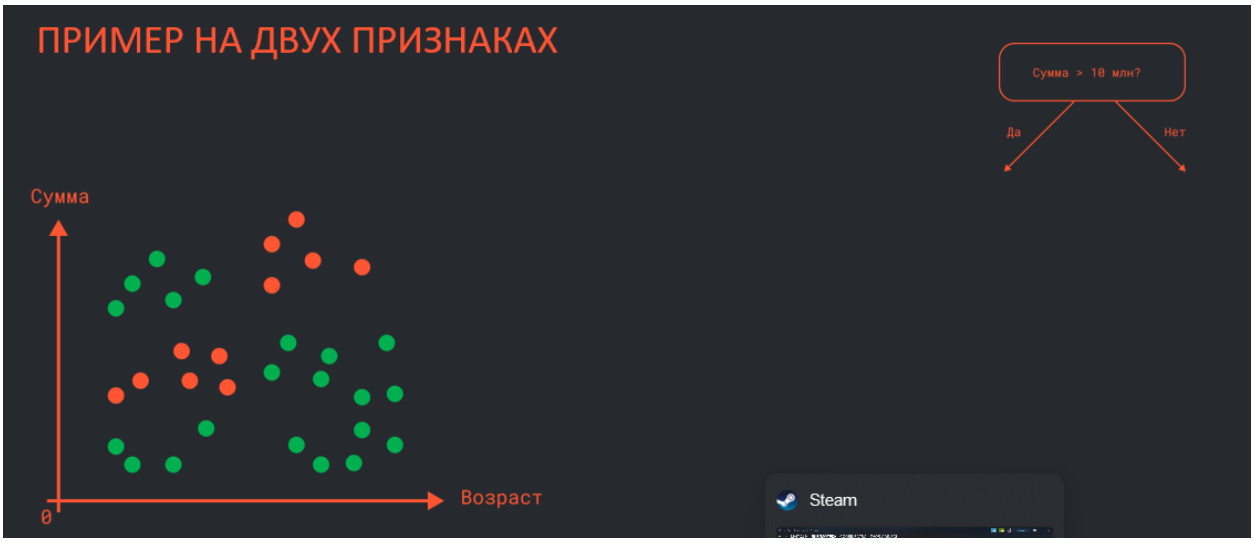


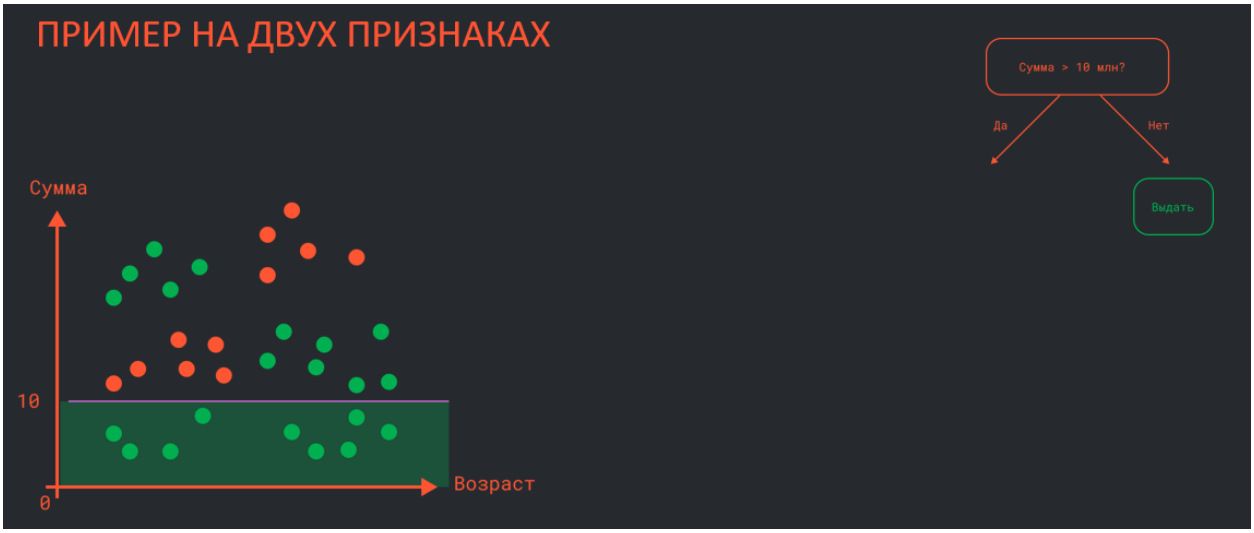
Отображение клиентов в виде точек на признаковом пространстве. Зеленым отображаются клиенты со значением таргета 1 — кредитоспособные, красным со значением 0 — некредитоспособные.

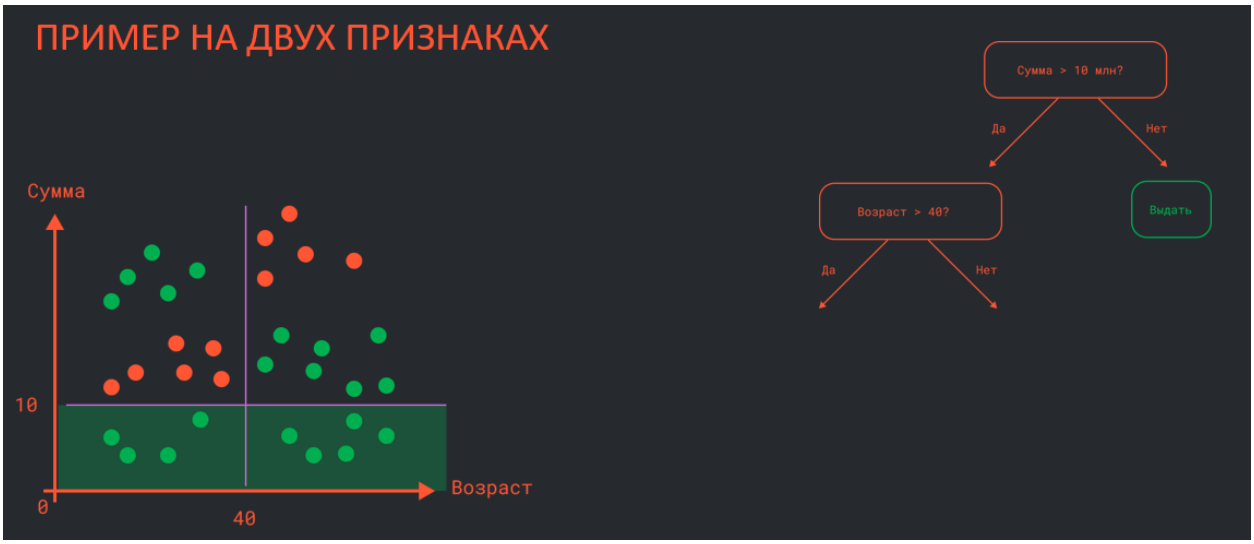


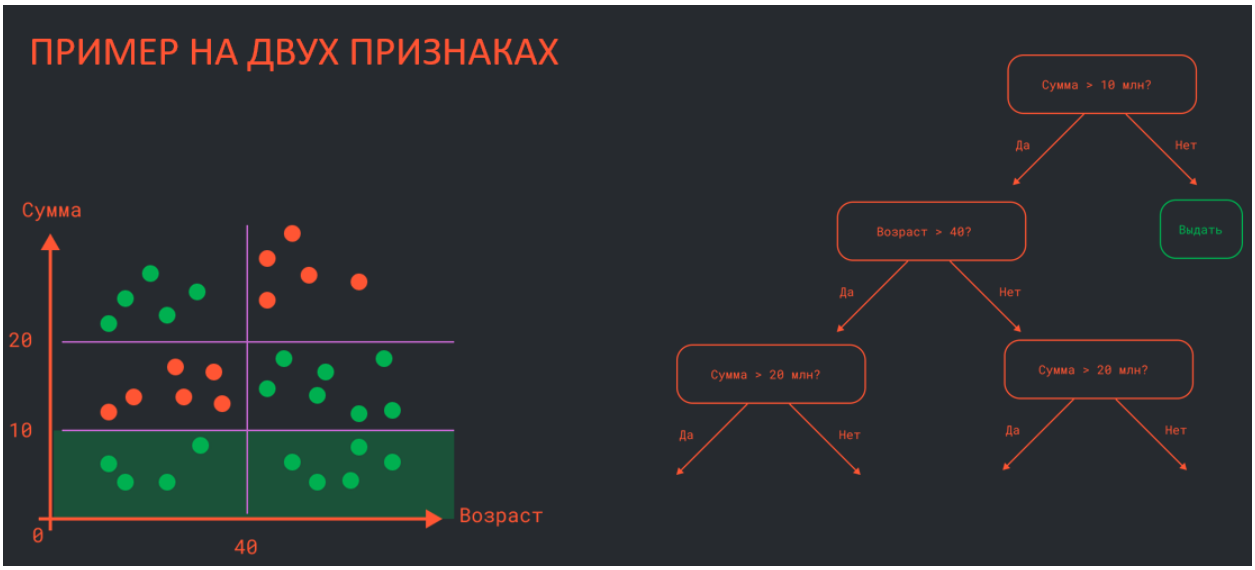
Допустим, что у нас есть некоторое правило. Эти правила мы будем называть предикатами, т.е. ту логику, которая содержится внутри наших вершин, и на которую нужно давать ответ «да» или «нет».

Далее давайте посмотрим на картинки с деревом и предикатами ниже.

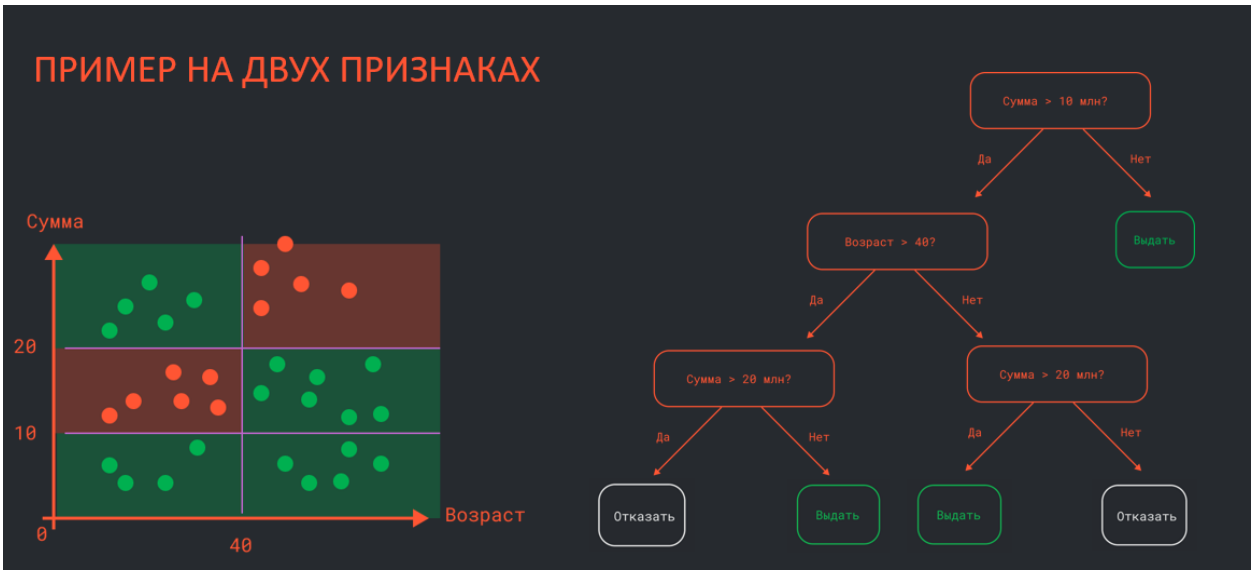

ПРИМЕР НА ДВУХ ПРИЗНАКАХ
Сумма > 10 млн?
Да
Нет
Сумма
Возраст
0
Steam


ПРИМЕР НА ДВУХ ПРИЗНАКАХ
Сумма > 10 млн?
Да
Нет
Выдать
Сумма
10
0
Возраст


ПРИМЕР НА ДВУХ ПРИЗНАКАХ
Сумма > 10 млн?
Да
Нет
Возраст > 40?
Выдать
Да
Нет
Сумма
10
0
40
Возраст



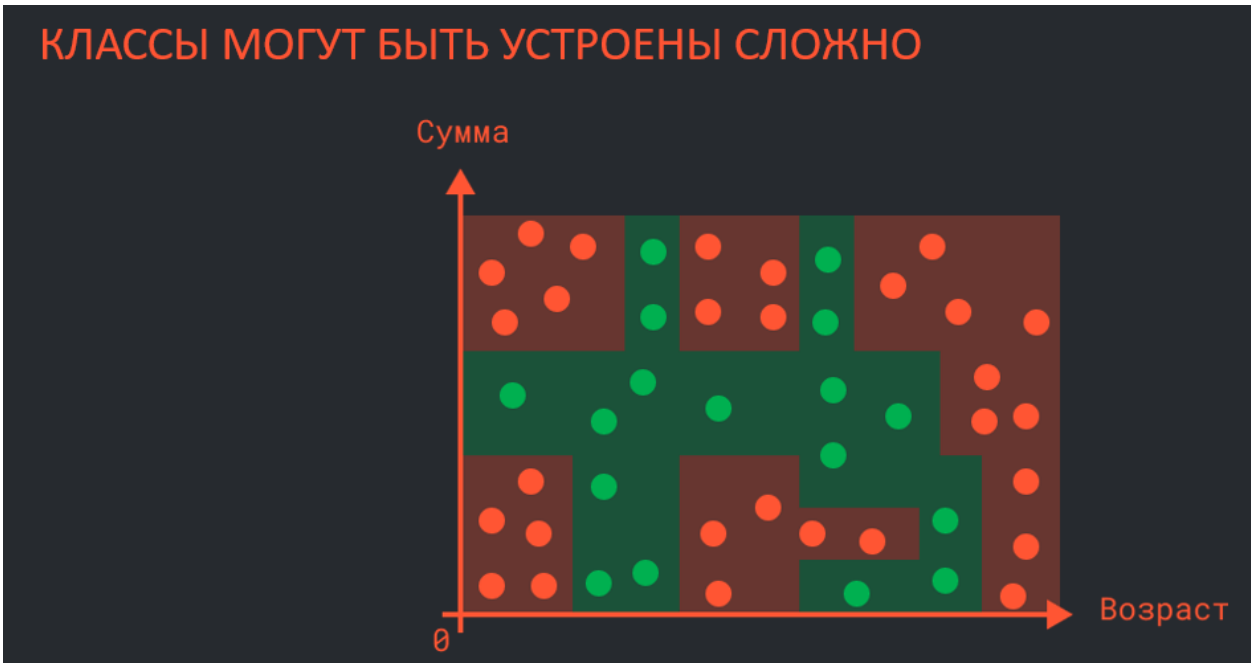
А теперь давайте посмотрим на картинке ниже на дерево с листовыми вершинами, в которых уже содержится какой-то прогноз.



Исходя из данных, которые содержатся в картинках выше, мы можем сделать вывод. При построении дерева мы подбираем некоторое правило, которое хорошо разделяет наши объекты. В идеале оно работает так: когда мы строим горизонтальную или вертикальную линию - по одну из ее сторон оказываются объекты исключительно одного класса. Когда это произошло в соответствующей вершине мы можем давать прогноз, сделать лист, если по какой-то из сторон объекты все еще перемешаны, мы продолжаем строить наше дерево, подбирать

какие-то новые предикаты, и у нас конечном итоге рано или поздно окажется такая ситуация, что абсолютно для каждого объекта из нашей выборки мы сможем дать какой-то ответ.

Пример на двух признаках был несложным, но часто на практике данные могут быть устроены намного сложнее, как на картинке ниже. И исходя из этого нам придется строить более сложные деревья и делить признаковое пространство на большее количество частей.



Выше мы рассматривали пример задачи классификации, а теперь давайте посмотрим, что будет происходить при рассмотрении задачи регрессии. В листьях деревьев, которые решают задачу регрессии, просто лежат константные числа, и чтобы дать какой-то прогноз для того или иного объекта нашей модели достаточно ответить на ряд вопросов, на ряд предикатов, и когда мы попадаем в какую-то листовую вершину — смотрим, какое число там лежит и даем соответствующий ответ.

Пример задачи регрессии на картинке ниже.

> Критерии качества и информативности

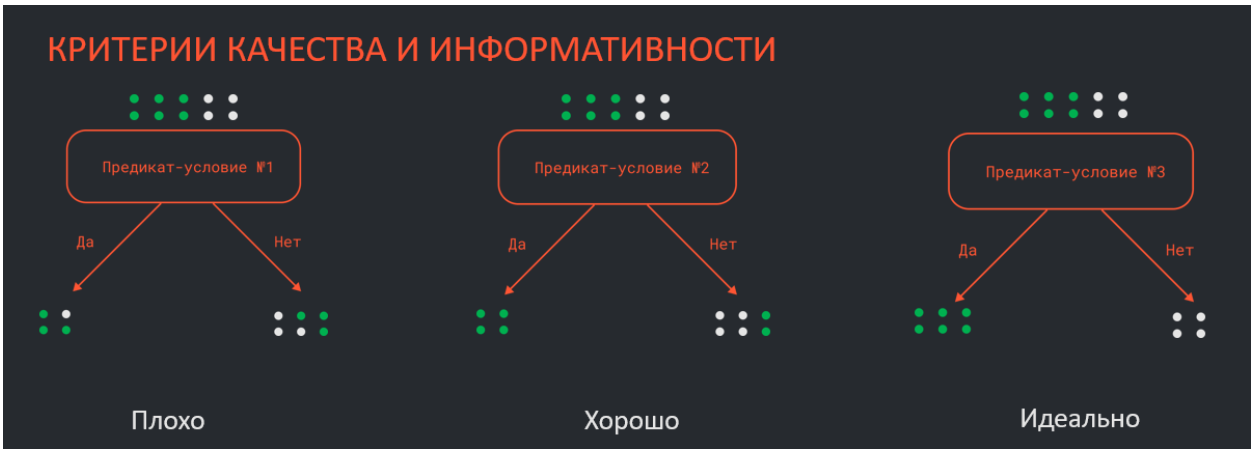
Каждый предикат в нашем дереве состоит из двух компонент и соответственно ими и описывается. Во-первых это признак *j,* по которому мы выбираем условие, а во-вторых порог *t* — так называемый trashhold, который показывает, относительного какой величины мы будем отсекать признак.

Выходит, что чисто математически каждый предикат можно записать в виде такого индикатора:

$$[dj \leq t]$$

Если какое *j-ый* признак реализуется меньше, чем какое-то число *t*, тогда данное выражение, индикатор вернет нам 1, а когда больше, чем это число, индикатор возвращает 0. Соответственно 1 и 0 можно воспринимать как ответы «да» или «нет». В зависимости от размера выборки число таких предикатов может быть достаточно большим — например, у нас 1 млн объектов и у них тысяча признаков, понятное дело, что эти объекты можно отсекать по каждому из признаков, причем с разными порогами. Нужно научиться сравнивать, какие предикаты хорошие, какие плохие, а для этого нам понадобятся так называемые критерии информативности и критерии качества.

Представим, что мы уже находимся в вершине дерева, может быть в корневой, может во внутренней, и у нас есть в этой вершине некоторое разбиение, т. е. какое-то количество объектов, которое осталось от предыдущих, как показано на примерах на картинке ниже.



На картинке выше мы можем интуитивно заметить, какое разбиение лучшее, а какое худшее. Но давайте введем какой-то формальный критерий, который будет нам математически — в виде какого-то числа определять хороший предикат. Логика следующая: в чем состоит наша задача, когда мы смотрим на какую-то подвыборку, на какую-то ячейку? Мы пытаемся понять, под каким углом и по какому признаку лучше данную выборку разделить. Наша цель — это некоторая определенность в новых вершинах. Мы хотим хаотичность внутри каждой из ячеек уменьшить. Данные у нас изначально были разбиты достаточно хаотично в пропорции 60 на 40, то, чего мы хотим добиться вообще в идеале — это того, чтобы в каждой из полученных вершин соотношение было 100 к 0. Например слева были все зеленые, справа все белые. И как раз некоторую функцию, которая оценивает хаотичность в какой-то вершине, мы будем называть критерием информативности.

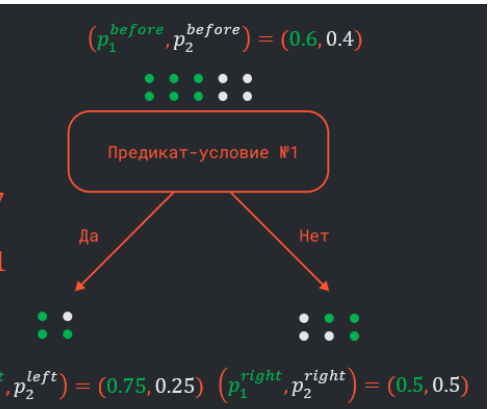
Для задачи классификации одним из популярных критериев является энтропия.

Формула энтропии:

$$H(p1, p2) = -(p1logp1 + p2logp2)$$

Под $p1$ и $p2$ мы можем понимать доли зеленых и белых объектов. Теперь давайте посчитаем энтропию на наших предикатах, что мы выделяли раньше на картинке выше. На картинках ниже мы можем увидеть результаты расчетов энтропии на разных предикатах и посмотреть, как происходят изменения в критерии информативности.

КРИТЕРИИ КАЧЕСТВА И ИНФОРМАТИВНОСТИ



— $H(p_1^{before}, p_2^{before}) = -(0.6 \cdot \log 0.6 + 0.4 \cdot \log 0.4) \approx 0.97$

— $H(p_1^{left}, p_2^{left}) = -(0.75 \cdot \log 0.75 + 0.25 \cdot \log 0.25) \approx 0.81$

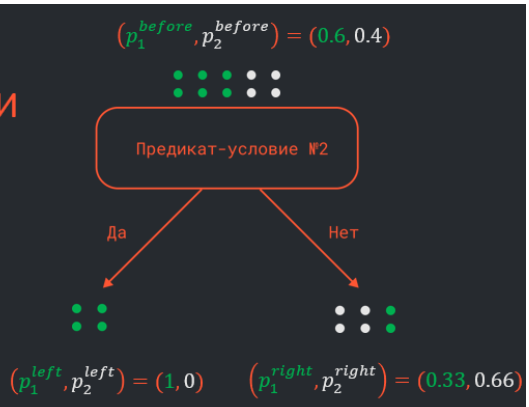
— $H\left(p_1^{right}, p_2^{right}\right) = -(0.5 \cdot \log 0.5 + 0.5 \cdot \log 0.5) = 1$

— Как изменилась хаотичность?

— Критерий качества – изменение в критерии информативности!

— $Q(d^j, t) = H(p_1^{before}, p_2^{before}) - \frac{1}{2} H(p_1^{left}, p_2^{left}) - \frac{1}{2} H\left(p_1^{right}, p_2^{right}\right)$

КРИТЕРИИ КАЧЕСТВА И ИНФОРМАТИВНОСТИ



— $H(p_1^{before}, p_2^{before}) \approx 0.97$

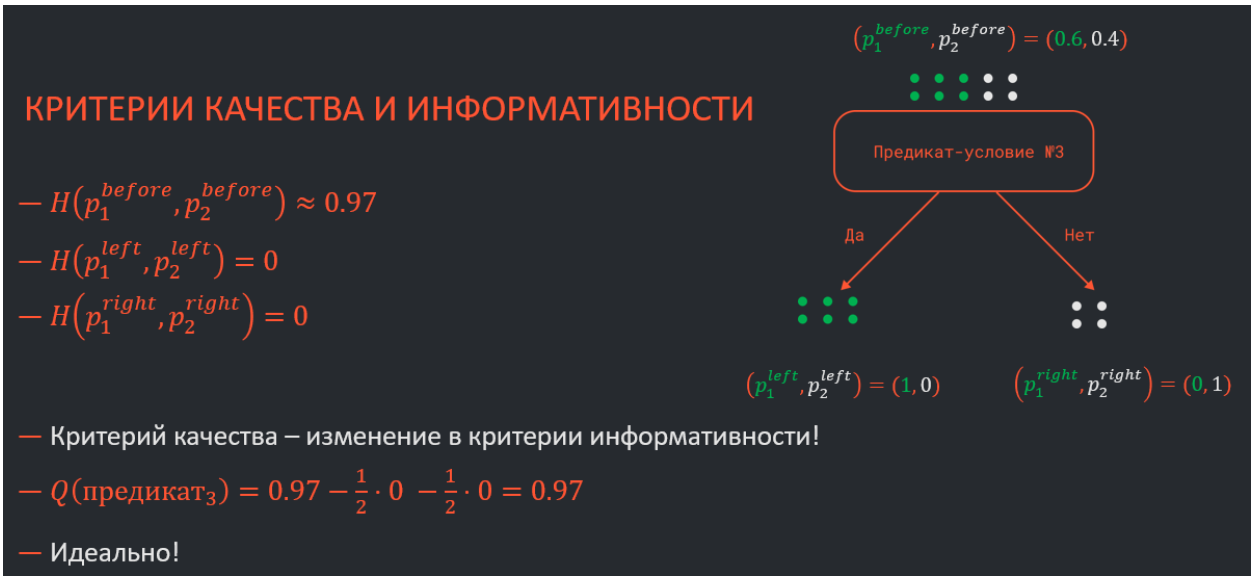
— $H(p_1^{left}, p_2^{left}) = 0$

— $H\left(p_1^{right}, p_2^{right}\right) \approx 0.92$

— Критерий качества – изменение в критерии информативности!

— $Q(\text{предикат}_2) = 0.97 - \frac{1}{2} \cdot 0 \ - \frac{1}{2} \cdot 0.92 = 0.51$

— Уже куда лучше!



Исходя из рассмотренного изменения в критерии информативности мы можем выбрать лучший предикат, максимизируя дельту критерия информативности.

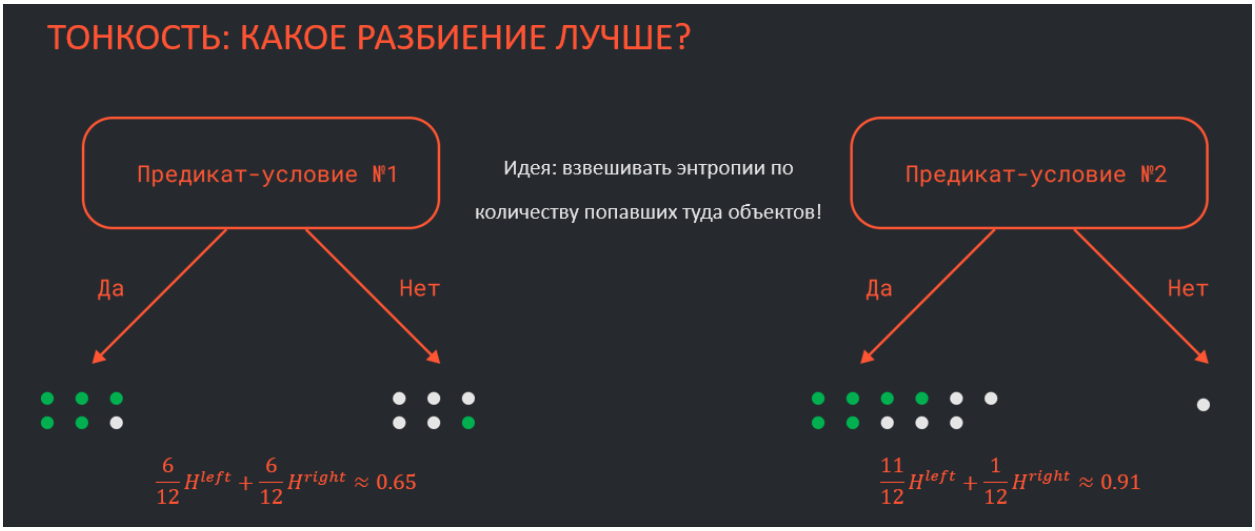
Для этого еще используется так называемый критерий Джини, очень похожий по свойствам на энтропию.

Формула критерия Джини:

$$H(p1, p2) = -(p1(1-p1) + p2(1-p2))$$

Понятно, что критериев информативности в машинном обучении достаточно много, но нам пока хватит этих двух. Конечно, в зависимости от того, по какому критерию мы будем сравнивать между собой предикаты, у нас итоговое решение может меняться, поэтому критерий информативности можно назвать гиперпараметром, который настраивается заранее еще до обучения, и в зависимости от того, как мы будем измерять качество предикатов, могут меняться наши финальные разбиения или финальные прогнозы. У данного сюжета есть небольшая тонкость, давайте ее обсудим, посмотрим на картинку внизу и попробуем снова применить интуицию.

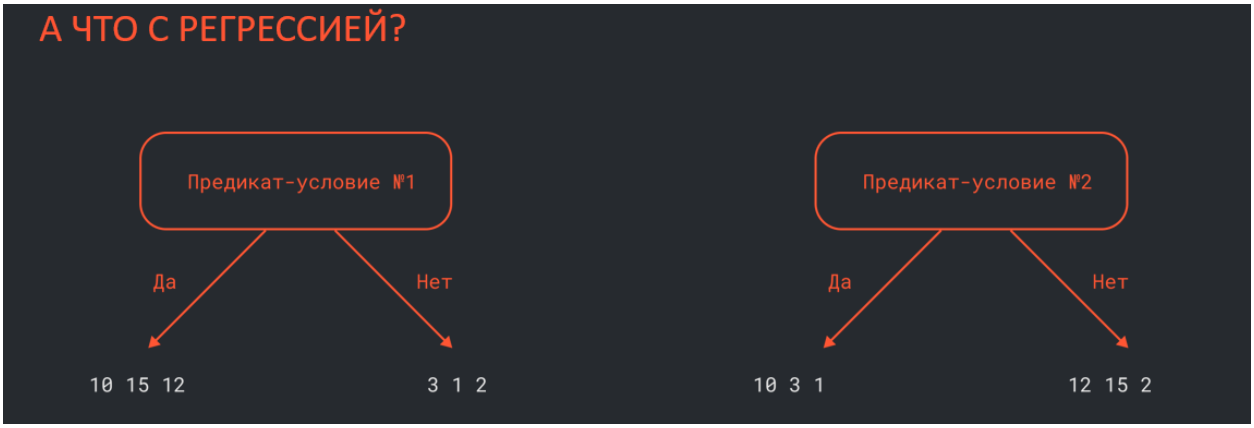
Из того, что мы видим на картинке, кажется, что предикат слева лучше. Но давайте посмотрим на результат вычисления энтропии слева и справа. Получается, что правый предикат лучше, но это совсем не бьется с нашей интуицией. Дело в том, что мы энтропию справа и слева на втором предикате взвешиваем одинаково, т.е. они равный вклад вносят в нашу формулу. Но является ли это справедливым, так как справа оказалось 11 объектов, а слева 1 объект. Можно предложить идею взвешивать энтропию по количеству попавших туда объектов. Тогда посмотрим на картинку ниже, где вычисляем энтропию согласно предложению.



Теперь, когда мы взвесили энтропию справа и слева, мы видим, что показатель энтропии и нашей интуиции начинает соотноситься. Левое правило показалось

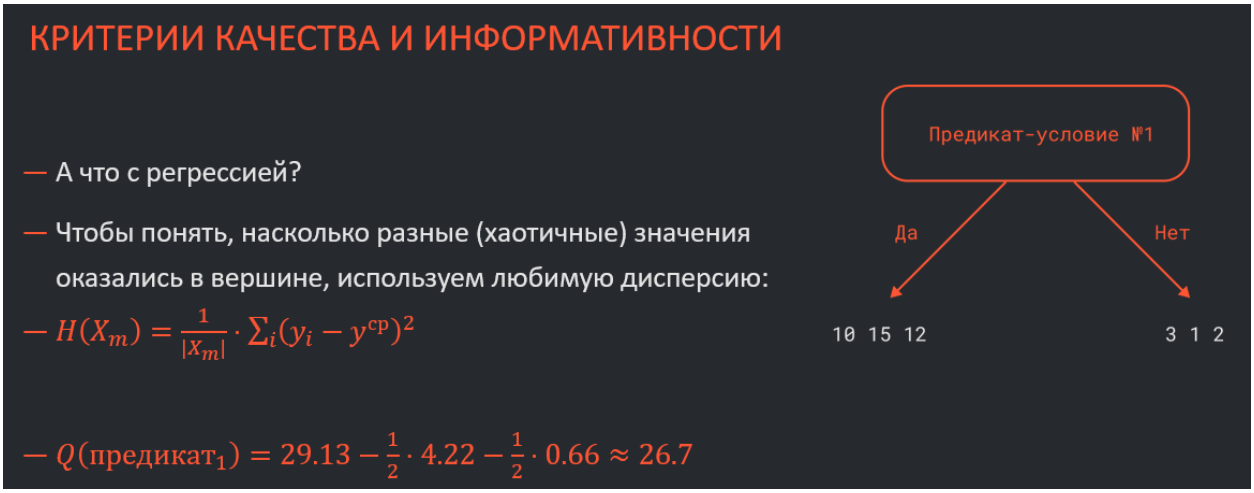
нам лучше, у него в итоге энтропия оказалась меньше, т.е. хаотичность в данных меньше, и он с большей уверенностью отличат классы друг от друга.

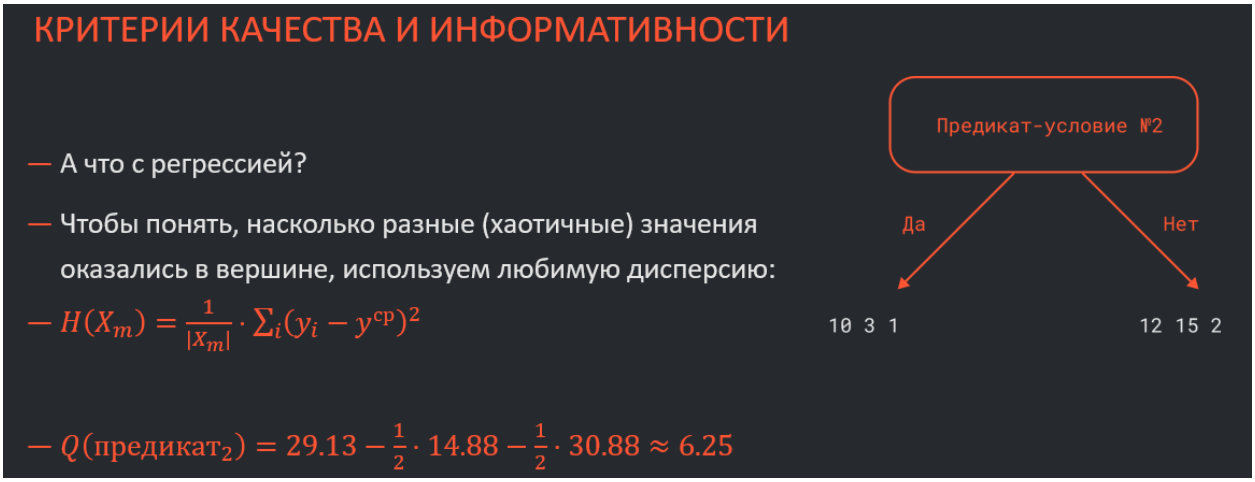
Мы рассматривали задачу классификации, а что с регрессией? Там все то же самое, нам нужно научиться измерять хаотичность в данных и сравнивать предикаты по некоторому критерию. Посмотрим на картинку внизу и снова воспользуемся интуицией, попробуем сравнить картинку слева и справа.



Когда мы решаем задачу регрессии и находимся в некоторой листовой вершине, чтобы дать прогноз, мы смотрим на объекты из обучающей выборки и усредняемся по ним. По факту, когда мы выбираем лучший предикат, что для задачи регрессии, что для задачи классификации, наша задача сделать такое разбиение объектов на две части, чтобы в этих частях оказалось как можно больше похожих друг на друга объектов. Кажется, что слева, правило №1 справляется с этой задачей значительно лучше, чем правило №2.

Давайте теперь формально научимся оценивать хаотичность в разбиении для задачи регрессией. С этим достаточно хорошо справляется дисперсия. Ведь она оценивает то, как в среднем числа отклоняются от своего среднего значения, иначе говоря, матожидания в квадрате, т.е. чем данные меньше отклоняются от среднего, тем менее хаотичны, если больше, то более хаотичны. Давайте при помощи дисперсии замерим критерий качества, для этого нам нужно взять начальную дисперсию и подсчитать взвешенную после разбиения.



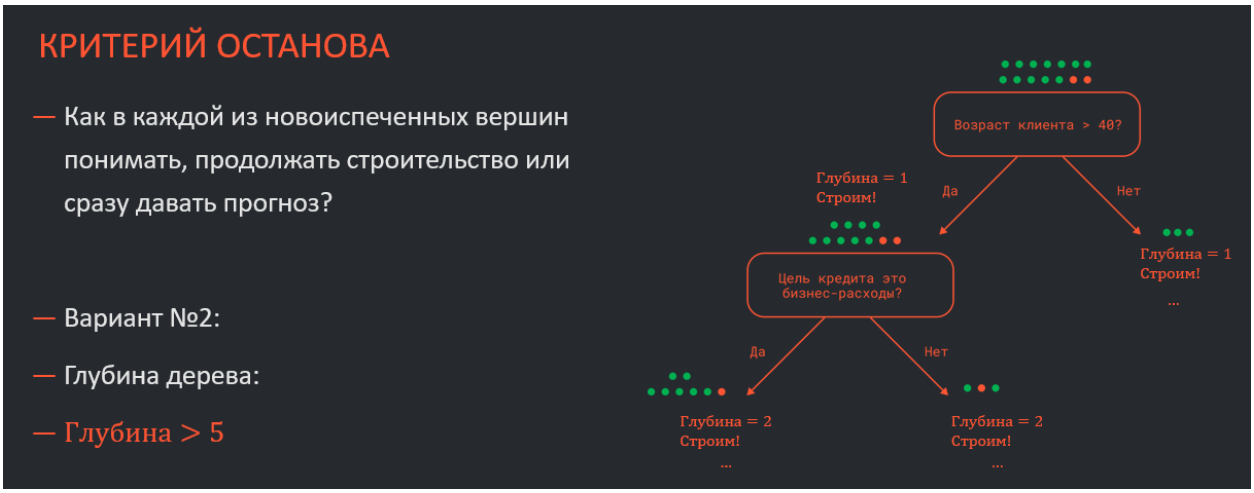


Исходя из картинок выше, мы выбираем первый предикат, который соотносится с нашей интуицией. На основании вышеизложенного мы можем вывести общий алгоритм выбора лучшего предиката. Пусть мы находимся в некой вершине (корень или некоторая внутренняя вершина), в данной вершине мы можем замерить критерий информативности, ну допустим для задачи классификации это будет критерий энтропии или критерий Джини, и далее мы перебираем все возможные пороговые значения t, всевозможные признаки d_j таким образом, что полученный предикат максимизировал разницу между начальной хаотичностью и той, которая получается после разбиения.

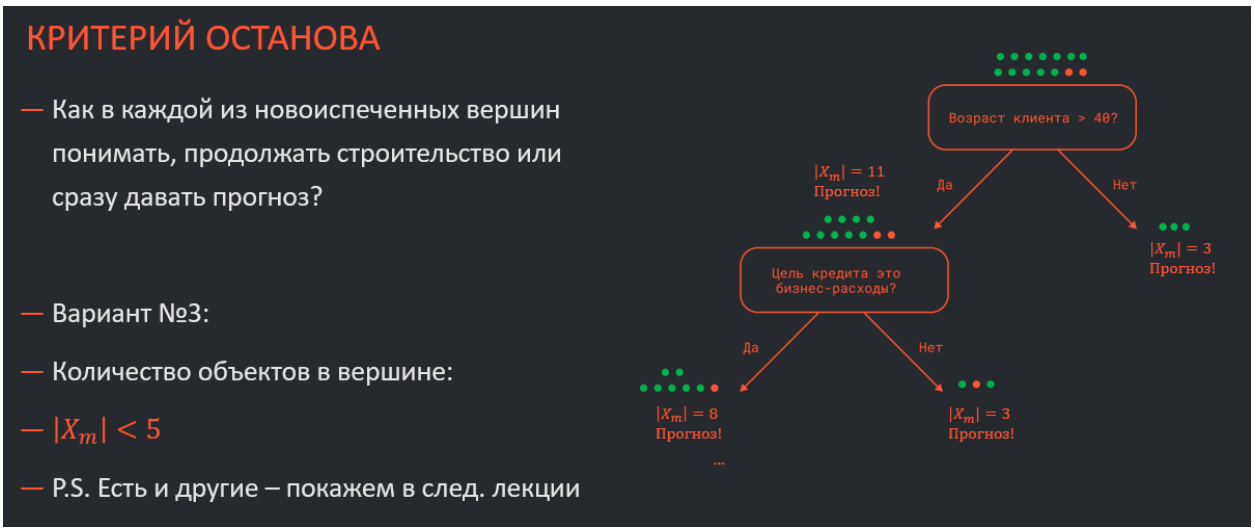
> Критерии останова

Когда мы находимся в какой-то вершине и выбрали какой-то лучший предикат, нам нужно понять следующее: вот мы в правую и левую вершину от него идем, надо ли нам дальше строить дерево или уже в какой-то из этих вершин можно давать прогноз. Нам нужны какие-то формальные критерии останова, которые будут сигнализировать нашему дереву о том, что можно прекратить дальнейшее построение. Как раз в качестве такого критерия останова может выступать уровень хаотичности, если он нас внутри вершины устраивает, тогда мы построение заканчиваем, а если оказывается достаточно большим, то продолжаем. Пример можно посмотреть на картинке ниже.

Второй вариант остановки построения дерева — это проверять глубину, на которой находится вершина, например, если глубина больше пяти, то мы останавливаемся. Пример можно посмотреть на картинке ниже.



Еще один очень популярный критерий останова — листовой, который сравнивает количество объектов, которое оказалось в вершине. Если объектов, меньше, чем пять, тогда вершину мы объявляем листовой, если больше, тогда объявляем внутренней и продолжаем строительство. Пример можно посмотреть на картинке ниже.



Есть, конечно, часть и других критериев останова, часть других мы рассмотрим на следующем занятии.

> Жадный алгоритм

Необходимо объединить воедино весь пройденный материал и придумать какой-то алгоритм, который поможет от начала и до конца построить дерево. Самый

популярный алгоритм — это так называемый жадный алгоритм. Пример построения можно посмотреть на картинке ниже.

Это рекурсивный алгоритм, который помогает нам построить дерево от начала и до конца, при этом для каждой вершины находя лучший предикат или сразу объявляя эту вершину листовой, если в ней выполняется критерий останова. Почему данный алгоритм называется жадным? Идея следующая: пусть мы на каждом этапе выбирали лучший предикат, но можно согласиться, что это ни каким образом не свидетельствует о том, что наше дерево идеально, что оно самое лучшее. Каждая вершина влияет на последующую, и может быть стоило начать с другого предиката. Буквально на каждом шаге алгоритма мы жадничаем, выбираем самый лучший предикат, но прогноз нашего дерева — это не конкретно какое-то правило, выбранное для нашего дерева, а последовательность, но при этом на практике это оказывается достаточным для того, чтобы улавливать какие-либо сложные зависимости и хорошо делить наше признаковое пространство.