

Решающие деревья

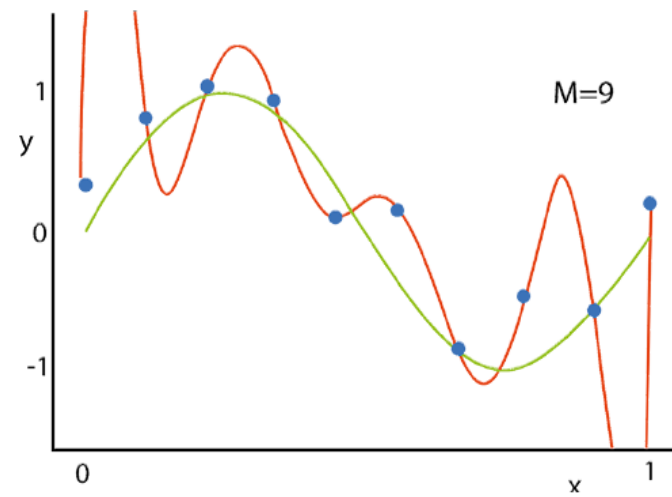
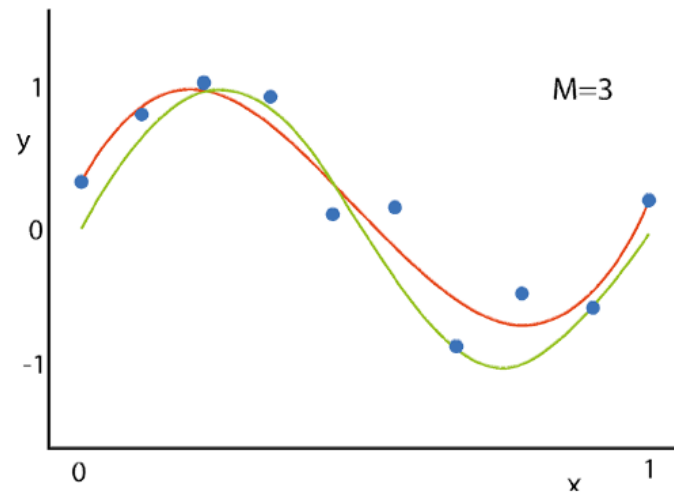
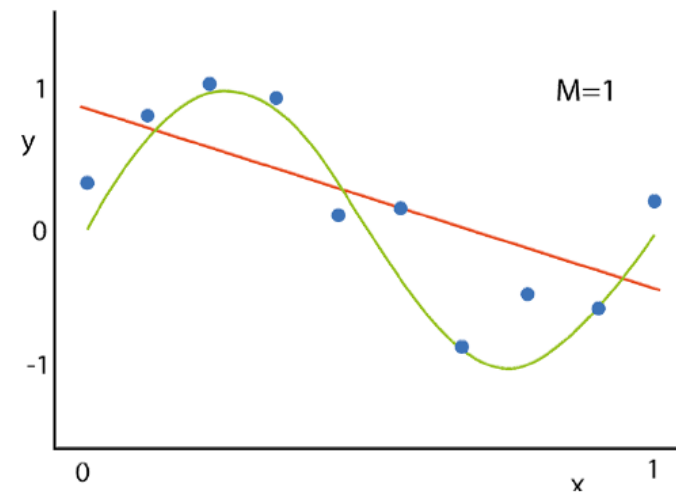
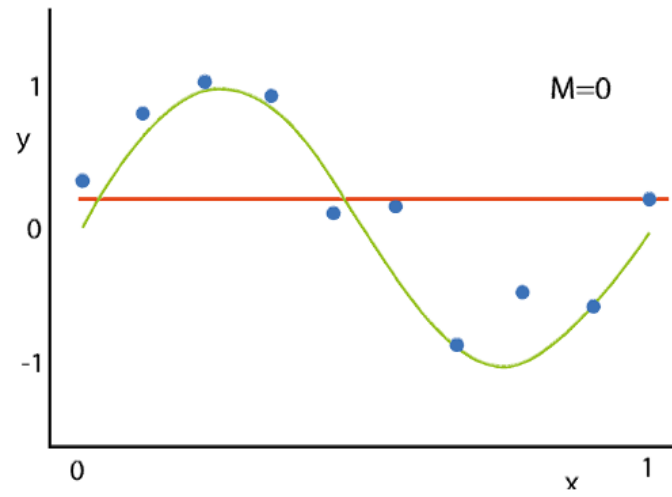
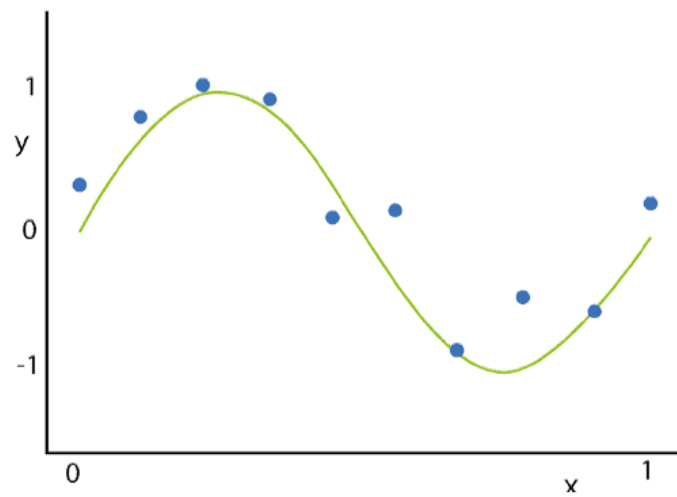
Занятие 2

Проблемы переобучения

- Простой пример
- Представим что у нас есть какая то модель, например – линейный классификатор
- Обучили ее на выборке, и измерили долю ошибок.
- Доля ошибок – 0.2
- Алгоритм обучился?

Проблема переобучения

- На новых данных получили долю неверных ответов – 0.9
- Из этого следует вывод, что алгоритм не обладает обобщающей способностью
- Но при этом он показал хорошее качество на обучении.
- Такая ситуация называется *переобучением модели*.



Вывод

- Недообучение – **плохое** качество на обучении и на новых данных
- Переобучение – **хорошее** качество на обучении и **плохое** на новых данных

Как выявить переобучение?

- Хороший алгоритм – хорошее качество на обучении
- Переобученный алгоритм – хорошее качество на обучении
- Вывод?

Способы выявить переобучение

- Отложенная выборка
- Кросс-валидация

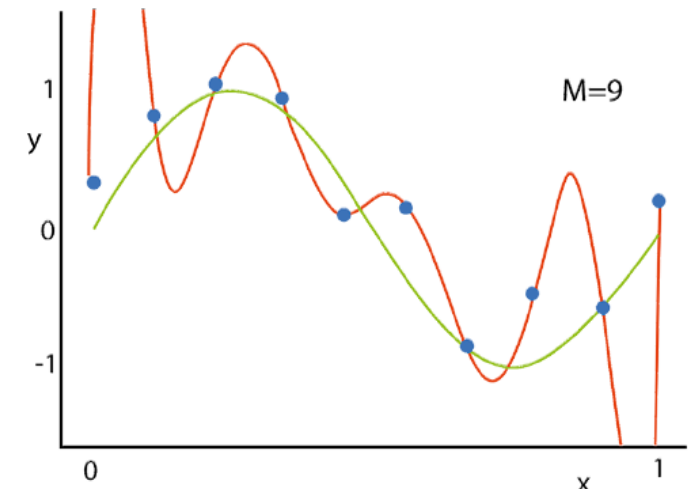
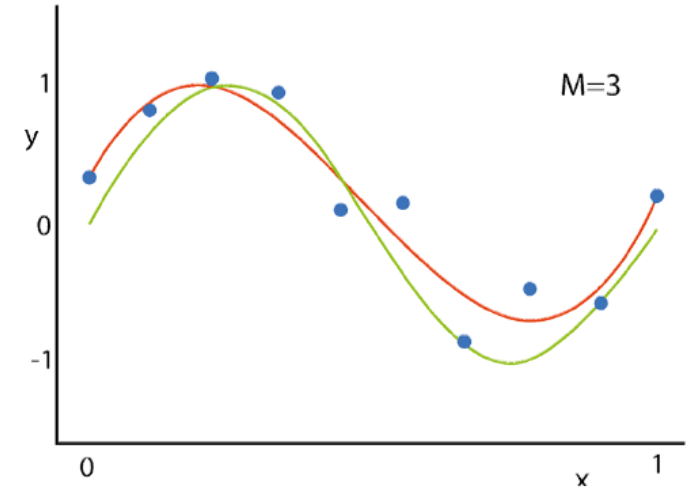
Пример переобучения по весам

$$a(x) = w_0 + w_1 x + w_2 x^2 + \dots + w_9 x^9$$

$$a(x) = 0.4 + 8x - 23x^2 + 19x^3$$

$$a(x) = 0.5 + 13458922x + 43983740x^2 + \dots + 2740x^9$$

Большие веса – признак **переобученности** модели
С большими весами можно бороться - **регуляризация**



Регуляризация

- $Q(a, x) = \frac{1}{l} \sum_{i=1}^l (a(x_i) - y_i)^2$ - функционал ошибки на обучении регрессии
- $Q(w, x) = \frac{1}{l} \sum_{i=1}^l (w_0 + \sum_{j=1}^d (w_j x_{ij}) - y_i)^2$ - функционал ошибки линейной модели на обучении регрессии
- Квадратичный регуляризатор

$$\|w\|^2 = \sum_{j=1}^d w_j^2$$

- Новый функционал:
- $Q(w, x) + \lambda \|w\|^2 \rightarrow \min$

Регуляризация

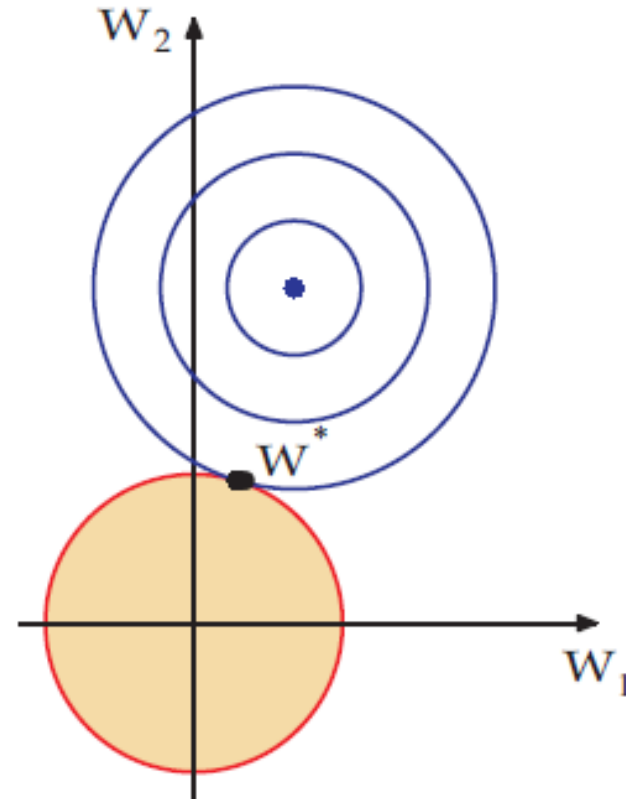
- Новый функционал:
- $Q(w, x) + \lambda \|w\|^2 \rightarrow \min$
- Чем больше λ , тем ниже сложность модели
- Чем меньше λ , тем выше риск переобучения
- Нужно подобрать оптимальное значение
- Как?
- Кросс-валидация

Смысл регуляризации

- $Q(w, x) + \lambda \|w\|^2 \rightarrow \min$

- Эквивалентная задача

- $$\begin{cases} Q(w, x) \rightarrow \min \\ \|w\|^2 < C \end{cases}$$



Виды регуляризаторов

- L2 – регуляризатор
- $\|w\|^2 = \sum_{j=1}^d w_j^2$
- Штрафует модель за сложность
- Гладкий и выпуклый

L1 – регуляризатор

$$\|w\|_1 = \sum_{j=1}^d |w_j|$$

Негладкий

Зануляет признаки

Позволяет отбирать признаки

Способы выявить переобучение

- Нужные дополнительные данные для выявления переобучения
- Методы:
- Отложенная выборка
- Кросс-валидация

Отложенная выборка

1. Разбиваем выборку на две части
2. На одной обучаем
3. На другой измеряем качество
Доля ошибок, MSE и т.д.

Маленькая отложенная часть

- Обучающая выборка репрезентативная
- Оценка качества ненадежная

Большая отложенная часть

- Оценка качества надежная
- Оценка качества смещенная

Обычно: 70/30, 80/20, 0.632/0.368

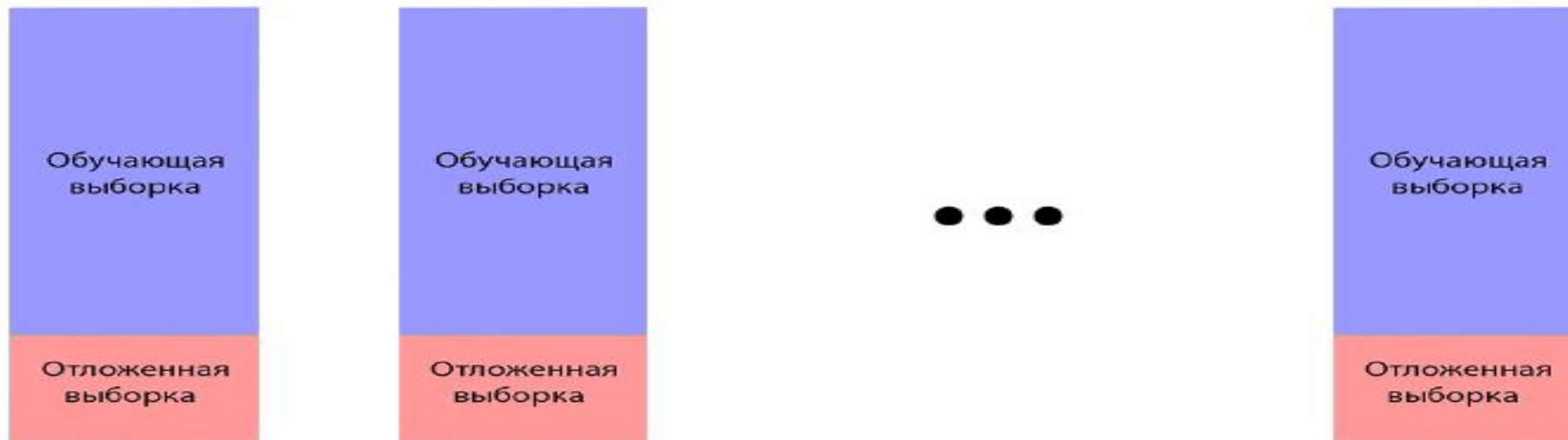


Особенности:

- Обучать нужно один раз
- Зависит от разбиения
- Подходит для большой выборки

Много отложенных выборок

- Разбиваем выборку случайным образом на две части с разными пропорциями n раз
- Усредняем оценку качества
- Нет гарантий, что каждый объект побывает в обучающей выборке



Кросс валидация

- Разбиваем выборку на k блоков
- Каждый блок по очереди выступает в качестве тестового
- Получим k оценок качества
- Усредним их и получим оценку качества по кросс валидации

Мало блоков

- Надежные оценки
- Смещенные оценки

Много блоков

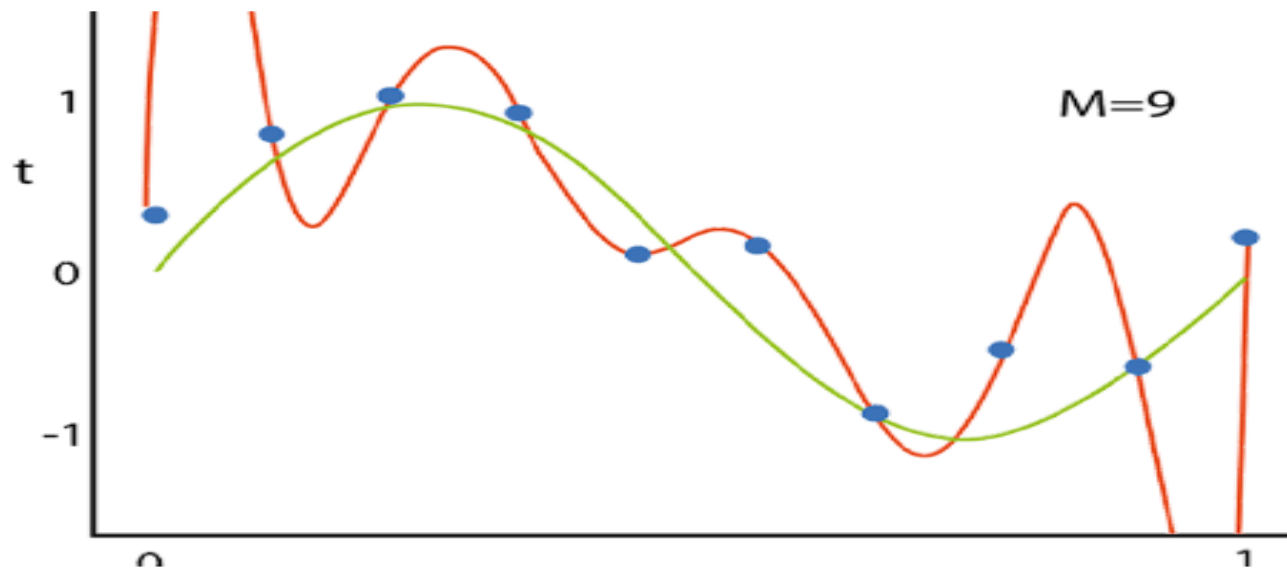
- Ненадежные оценки
- Несмещенные оценки

- Обычно $k = 3, 5, 10$
- Чем больше выборка тем меньше k
- Чем больше k , тем больше раз надо обучать алгоритм



Сравнение алгоритмов и выбор гиперпараметров

- Существуют параметры, которые нельзя настроить на обучающей выборке
- Примеры: параметр регуляризации
- Примеры: степень полинома



Примеры: что лучше линейная модель или решающее дерево
Примеры: какой функционал лучше – MSE или MAE

Как выбрать наилучшие параметры?

- Сравним 1000 алгоритмов
- Выбираем лучший на отложенной выборке
- Отложенная выборка превращается в обучающую
- Легко переобучиться

Более хитрая схема


Обучение

Валидация

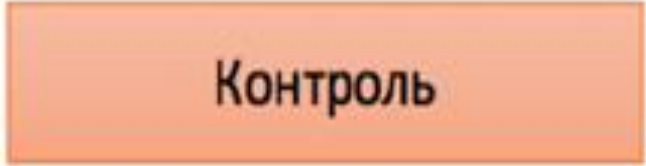
Контроль

- Настраиваем все алгоритмы на обучении
- Сравниваем на валидации
- Лучший проверяем на контроле (тесте)

Либо кросс валидация



Кросс-валидация



Контроль

- Обучаем и сравниваем алгоритмы с помощью кросс-валидации
- Лучший проверяем на контроле

Метрики качества в задачах регрессии

- Для задания функционала ошибки (обучение)
- Для подбора гиперпараметров (кросс – валидация)
- Для оценивания итоговой модели (бизнес, медицина)

Метрики

- $MSE = \frac{1}{l} \sum_{i=1}^l (a(x_i) - y_i)^2$
- $MAE = \frac{1}{l} \sum_{i=1}^l |a(x_i) - y_i|$
- $R^2(a, X) = 1 - \frac{\sum_{i=1}^l (a(x_i) - y_i)^2}{\sum_{i=1}^l (y_i - \bar{y})^2}$ где
- $\bar{y} = \frac{1}{l} \sum_{i=1}^l y_i$ – средний ответ
- Коэффициент детерминации – доля дисперсии, объяснённая моделью, в общей дисперсии ответов
- Значение можно интерпретировать

Коэффициент детерминации

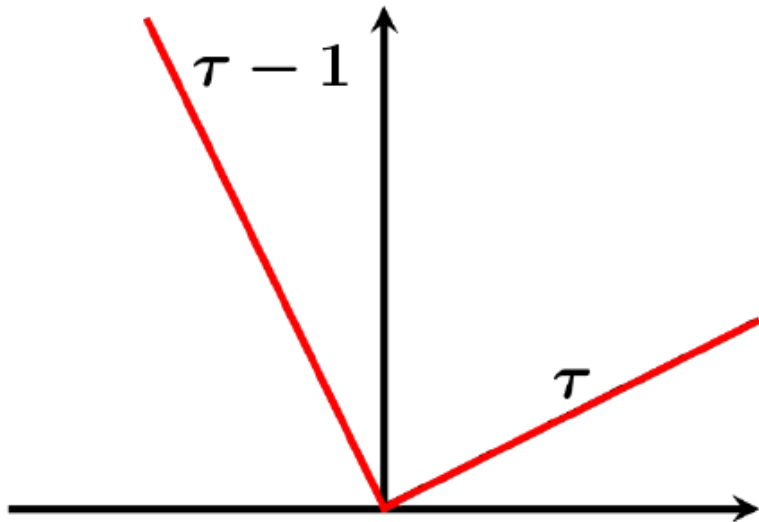
- $R^2(a, X) = 1 - \frac{\sum_{i=1}^l (a(x_i) - y_i)^2}{\sum_{i=1}^l (y_i - \bar{y})^2}$
- $0 \leq R^2 \leq 1$ (для разумных моделей)
- $R^2 = 1$ – идеально
- $R^2 = 0$ – модель на уровне константной
- $R^2 < 0$ – модель хуже константной

Несимметричные потери

- Пример: прогнозирование спроса на ноутбуки
- Заниженный прогноз – потеря лояльности и потенциальной прибыли
- Завышенный прогноз – расходы на хранение

Квантильная ошибка

$$\rho_{\tau}(a, X) = \frac{1}{l} \sum_{i=1}^l \left((\tau - 1)[y_i < a(x_i)] + \tau[y_i \geq a(x_i)] \right) (y_i - a(x_i))$$



Качество классификации

- Доля неправильных ответов:

$$\frac{1}{\ell} \sum_{i=1}^{\ell} [a(x_i) \neq y_i]$$

Доля правильных ответов:

$$\text{accuracy}(a, X) = \frac{1}{\ell} \sum_{i=1}^{\ell} [a(x_i) = y_i]$$

Несбалансированные выборки

- Пример
- Класс -1 : 950 объектов
- Класс +1 : 50 объектов
- $A(x) = -1$
- Доля правильны ответов: 0.95
- Надежно?

Цены ошибок

- Пример: кредитный скоринг
- Модель 1:
 - 80 кредитов вернули
 - 20 кредитов не вернули
- Модель 2:
 - 48 кредитов вернули
 - 2 кредита не вернули
- Кто лучше?

Точность и полнота

- Что хуже?
- Выдать кредит плохому клиенту
- Не выдать кредит хорошему клиенту
- Доля верных ответов не учитывает цены ошибок

Матрица ошибок классификации

Категория i		Экспертная оценка	
		Положительная	Отрицательная
Оценка системы	Положительная	TP	FP
	Отрицательная	FN	TN

Модель $a_1(x)$

	$y = 1$	$y = -1$
$a(x) = 1$	80	20
$a(x) = -1$	20	80

Модель $a_2(x)$

	$y = 1$	$y = -1$
$a(x) = 1$	48	2
$a(x) = -1$	52	98

Точность (Precision)

- Можно ли доверять классификатору при $a(x) = 1$?

$$\text{precision}(a, X) = \frac{TP}{TP + FP}$$

Модель $a_1(x)$

	$y = 1$	$y = -1$
$a(x) = 1$	80	20
$a(x) = -1$	20	80

Модель $a_2(x)$

	$y = 1$	$y = -1$
$a(x) = 1$	48	2
$a(x) = -1$	52	98

Попробуйте посчитать Precision для двух моделей

Модель $a_1(x)$

	$y = 1$	$y = -1$
$a(x) = 1$	80	20
$a(x) = -1$	20	80

Модель $a_2(x)$

	$y = 1$	$y = -1$
$a(x) = 1$	48	2
$a(x) = -1$	52	98

» $\text{precision}(a_1, X) = 0.8$ » $\text{precision}(a_2, X) = 0.96$

Полнота (Recall)

- Как много положительных объектов находит классификатор?

$$\text{recall}(a, X) = \frac{TP}{TP + FN}$$

Полнота(Recall)

	$y = 1$	$y = -1$
$a(x) = 1$	80	20
$a(x) = -1$	20	80

	$y = 1$	$y = -1$
$a(x) = 1$	48	2
$a(x) = -1$	52	98

Попробуйте посчитать Recall для двух моделей

Полнота(Recall)

	$y = 1$	$y = -1$
$a(x) = 1$	80	20
$a(x) = -1$	20	80

	$y = 1$	$y = -1$
$a(x) = 1$	48	2
$a(x) = -1$	52	98

» $\text{recall}(a_1, X) = 0.8$

» $\text{recall}(a_2, X) = 0.48$

Кредитный скоринг

- Неудачных кредитов должно быть не больше 5%
- Ограничение: $\text{precision} \geq 0.95$
- Максимизируем полноту

Медицинская диагностика

- Надо найти не менее 80% больных
- Ограничение: $\text{recall} \geq 0.8$
- Максимизируем точность

Объединение

$$A = \frac{1}{2}(\text{precision} + \text{recall})$$

$$M = \min(\text{precision}, \text{recall})$$

$$F = \frac{2 \cdot \text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}$$

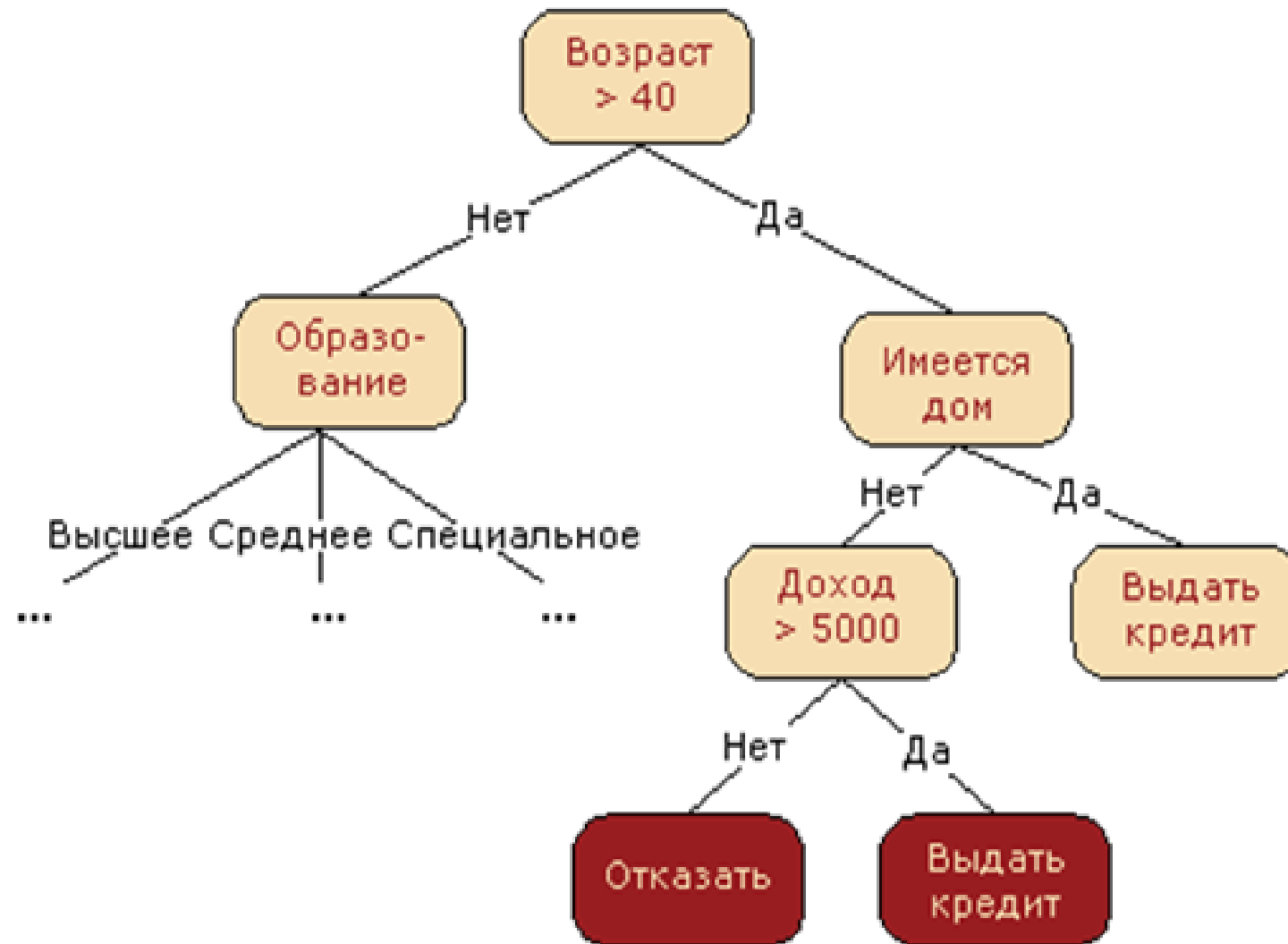
$$F_{\beta} = (1 + \beta^2) \frac{\text{precision} \cdot \text{recall}}{\beta^2 \cdot \text{precision} + \text{recall}}$$

$\beta = 0.5$, важнее полнота

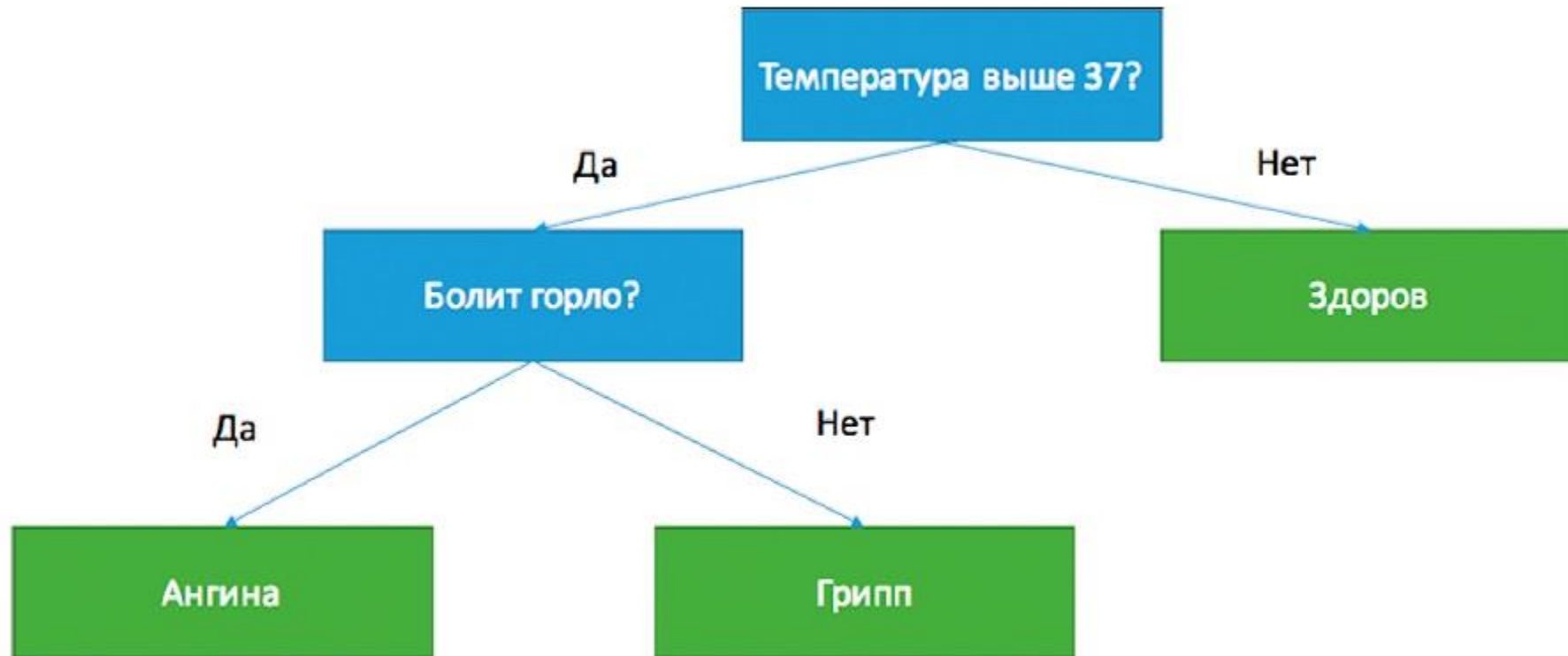
$\beta = 2$, важнее точность

Перейдем к деревьям

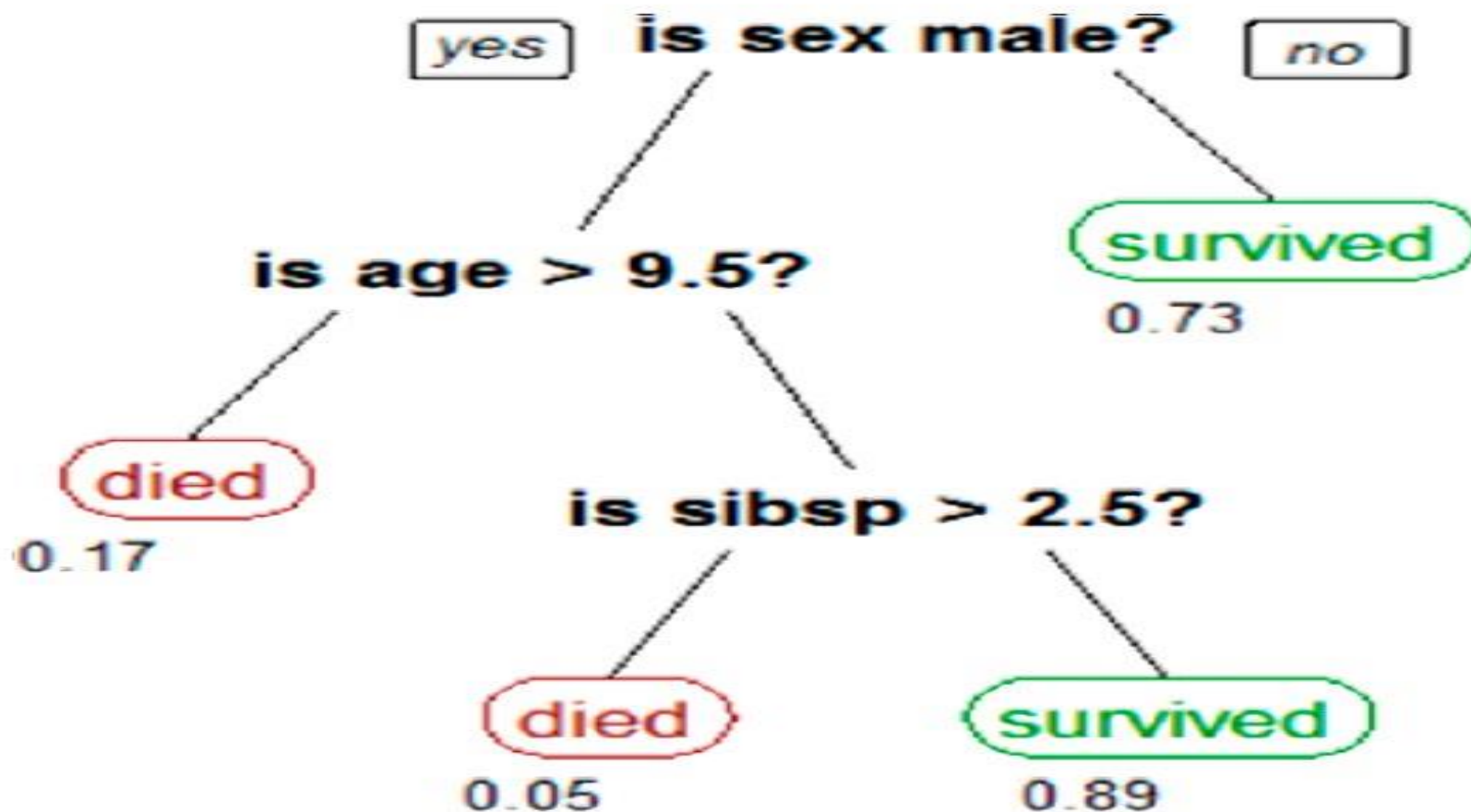
Кредитный скоринг



Медицинская диагностика



Прогнозирование смертей пассажиров Титаника



Критерия разбиения

- 1. Какой возраст у клиента? Если меньше 18, то отказываем в кредите, иначе продолжаем.
- 2. Какая зарплата у клиента? Если меньше 50 тыс. рублей то переходим к шагу 3, иначе к шагу 4.
- 3. Какой стаж у клиента? Если меньше 10 лет, то не выдаем кредит, иначе выдаем.
- 4. Есть ли у клиента другие кредиты? Если есть, то отказываем, иначе выдаем.

Популярность

- Решающие деревья не часто используются как отдельные алгоритмы
- Хорошо работают в композиции – пример: решающие леса
- Композиции из деревьев, одни из наиболее сильных и универсальных моделей

Определение

Рассмотрим бинарное дерево :

- Каждой внутренней вершине v приписана функция (или предикат) $\beta_v: X \rightarrow \{0, 1\}$;
- Каждой листовой вершине v приписан прогноз $c_v \in Y$ (в случае с классификацией — вектор вероятностей)

Рассмотрим алгоритм $a(x)$,

1. Стартуем из корневой вершины v_0 и вычисляем значение функции β_{v_0} .
2. Если оно равно нулю, то алгоритм переходит в левую дочернюю вершину, иначе в правую
3. Вычисляем предикат в новой вершине, и переходим в левое или правое и т.д. пока не достигнем листа
4. Алгоритм возвращает класс, который приписан данному листу

Такой алгоритм называют *бинарным решающим деревом*.

Определение(Коротко)

- Бинарное дерево (не обязательно)
- В каждой внутренней вершине записано условие
- В каждом листе записан прогноз

Вопрос

- Какая основная проблема решающих деревьев?

Предикаты

1. Одномерные предикаты(сравнивают значение одного из признаков с порогом)

$$\beta_v(x; j, t) = [x_j < t]$$

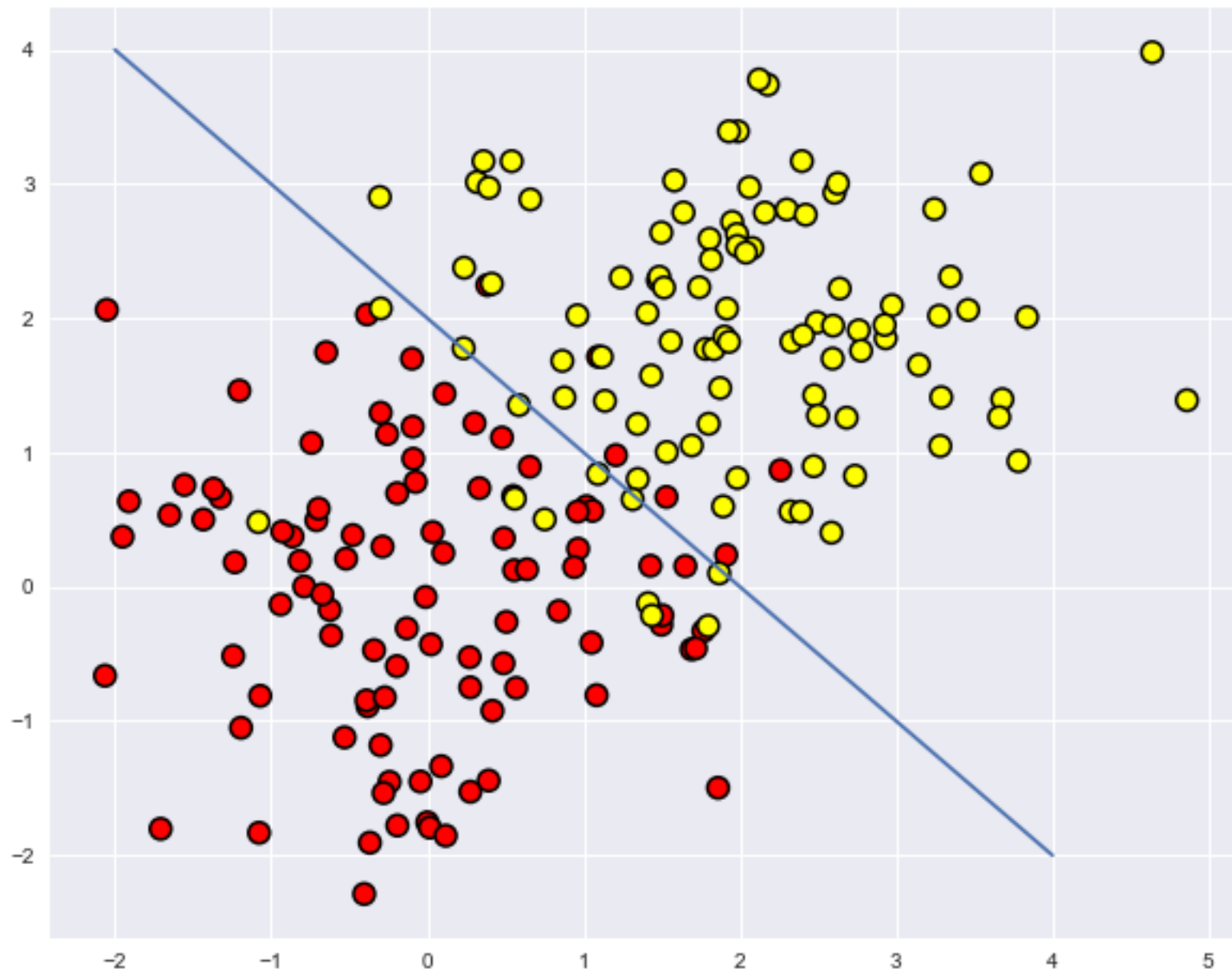
2. Многомерные предикаты

- Линейные $\beta_v(x) = [\langle w, x \rangle < t]$;
- Метрические $\beta_v(x) = [\rho(x, x_v) < t]$

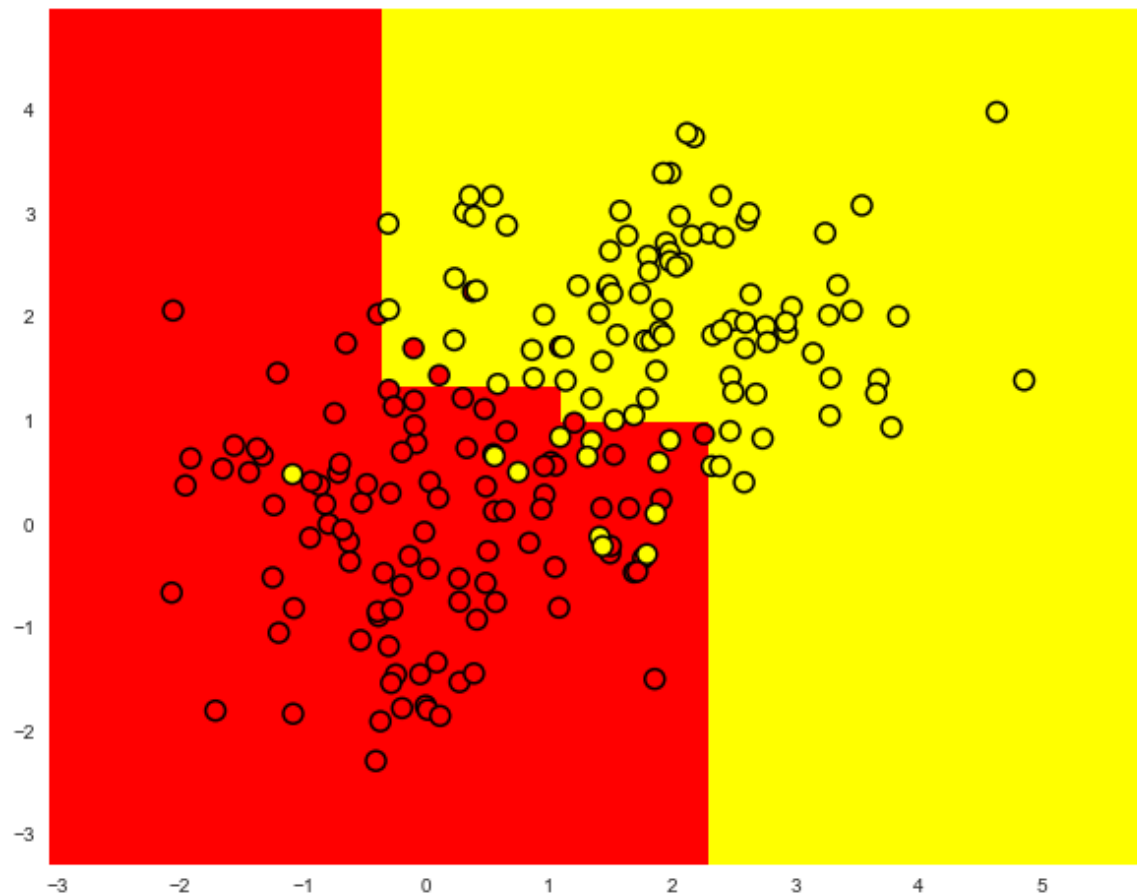
Многомерные предикаты помогают строить более сложные модели, не приводят к одной распространенной проблеме решающих деревьев – переобучение.

Пример

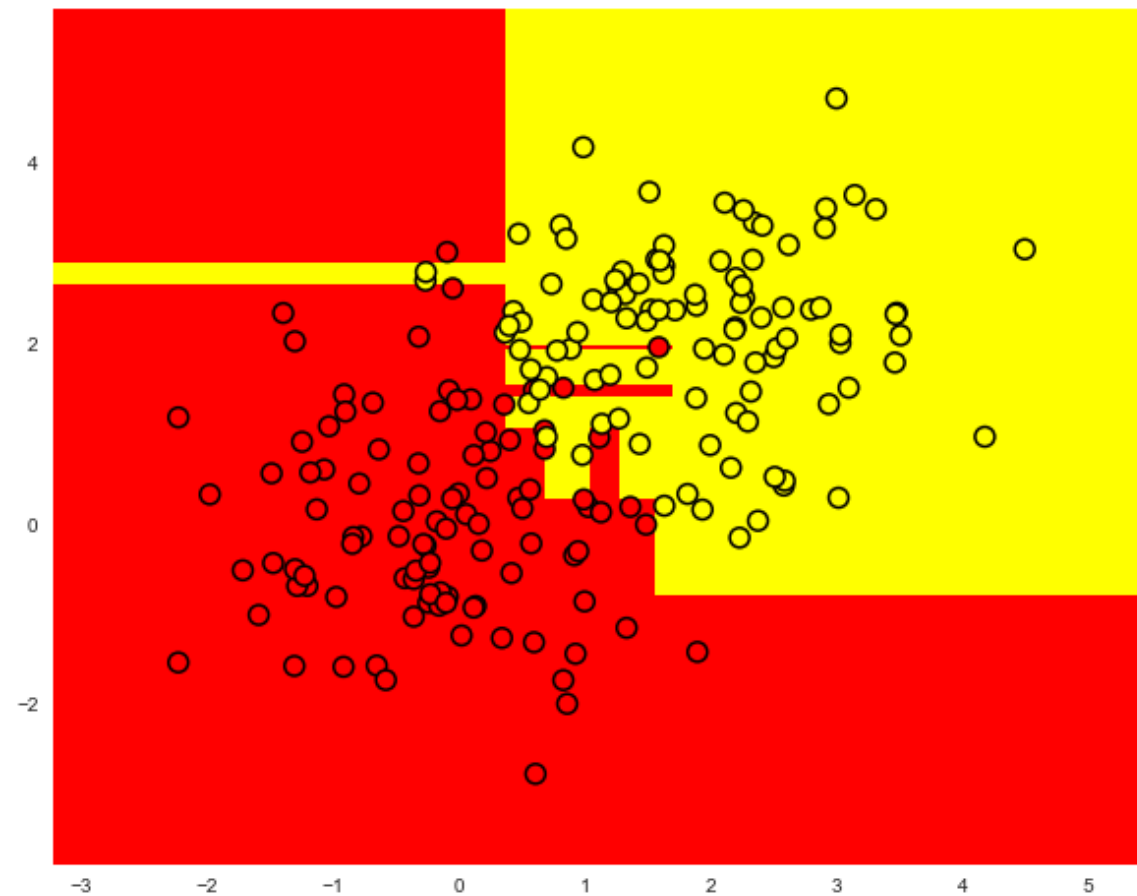
- Сгенерируем данные.
- Два класса будут сгенерированы из двух нормальных распределений с разными средними



Пример



Дерево с глубиной 3



Неограниченная глубина

Дерево не допускает ошибок

- Для любой выборки можно построить решающие дерево, не допускающее на ней ни одной ошибки
- Такое дерево будет переобученным, в каждом листе которого находится ровно один объект выборки
- Дерево не сможет показать качество на новых данных.
- Можно поставить задачу поиска дерева, которое является минимальным(листья) среди всех деревьев, не допускающих ошибок на обучении. В таком случае можно надеяться на обобщающую способность.
- NP – полная задача, пользуемся жадными алгоритмами.

Построение дерева

- 1. Возьмем обучающую выборку X и найдем наилучшее ее разбиение на две части

$$R_1(j, t) = \{x | x_j < t\} \text{ и } R_2(j, t) = \{x | x_j \geq t\}$$

С точки зрения заранее заданного функционала качества $Q(X, j, t)$

- 2. Найдя наилучшие значения j и t , создадим корневую вершину дерева, поставив ей в соответствие предикат $[x_j < t]$. Объекты разобьются на две части – одни попадут в левое поддерево, другие в правое.
- 3. Для этих поддеревьев рекурсивно повторим операцию.
- 4. В каждой вершине мы проверяем, не выполнилось ли некоторое условие останова
- 5. Если выполнилось, прекращаем рекурсию и объявляем вершину листом

Пример



Ответ дерева

- Когда дерево построено, каждому листу ставится в соответствие ответ.
- Классификация.
В случае с классификацией это может быть класс, к которому относится больше всего объектов в листе, или вектор вероятностей(вероятность класса может быть равна доле его объектов в листе)
- Регрессия
Для регрессии это может быть среднее значение, медиана или другая функция(выбор зависит от функционала качества)

Факты о дереве решений

- Решающие деревья последовательно проверяют простые условия
- Интерпретируемые
- Позволяют восстанавливать нелинейные зависимости
- Легко переобучаются

Пропущенные значения

- Решающие деревья могут обрабатывать пропущенные значения
- Ситуации, в которых для некоторых объектов неизвестны значения одного или нескольких признаков
- Для этого нужно модифицировать процедуру разбиения выборки в вершине.

Стрижка деревьев

- После того, как дерево построено, можно провести его стрижку(pruning)
- Удаление некоторых вершин с целью понижения сложности и повышения обобщающей способности



Summary

Конкретный метод построения решающего дерева определяется

1. Видом предикатов в вершинах
2. Функционалом качества $Q(X, j, t)$
3. Критерием останова
4. Методом обработки пропущенных значений
5. Методом стрижки

Критерий информативности

- При построении дерева необходимо задать функционал качества, на основе которого осуществляется разбиение выборки на каждом шаге
- Обозначим через X_m множество объектов, попавших в вершину, разбиваемую на данном шаге
- А через X_l и X_r - объекты, попадающие в левое и правое поддереву соответственно при заданном предикате
- Будем использовать функционалы следующего вида:
 - $Q(X_m, j, t) = H(X_m) - \frac{|X_l|}{|X_m|} H(X_l) - \frac{|X_r|}{|X_m|} H(X_r).$
- Где $H(X)$ – это *критерий информативности (impurity criterion)* – который оценивает качество распределения целевой переменной среди объектов множества X
- Такой функционал также называют *приростом информации*
- Критерий информативности минимизируется, функционал качества максимизируется.

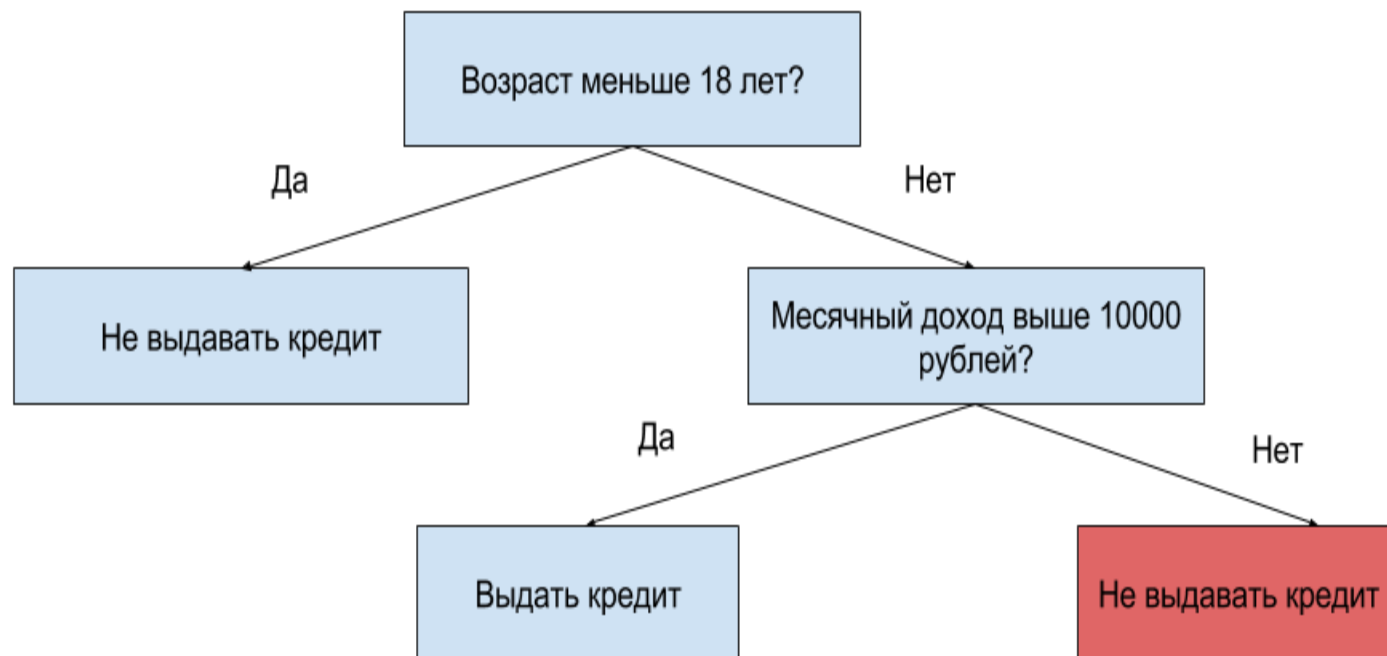
Критерий информативности

- В каждом листе дерево выдает константу – вещественное число, вероятность или класс.
- Исходя из этого можно предложить оценивать качество множества объектов X , насколько их целевые переменные предсказываются константой (при оптимальном выборе константы)
- $H(X) = \min_{c \in Y} \frac{1}{|X|} \sum_{(x_i, y_i) \in X} L(y_i, c)$
- Где $L(y, c)$ – некоторая функция потерь

Упражнение 1

Пусть дана следующая выборка из пяти объектов(первый признак-возраст, второй – месячный доход):

[20, 8000]
[15, 15000]
[28, 9500]
[24, 30000]
[30, 20000]



Упражнение 2

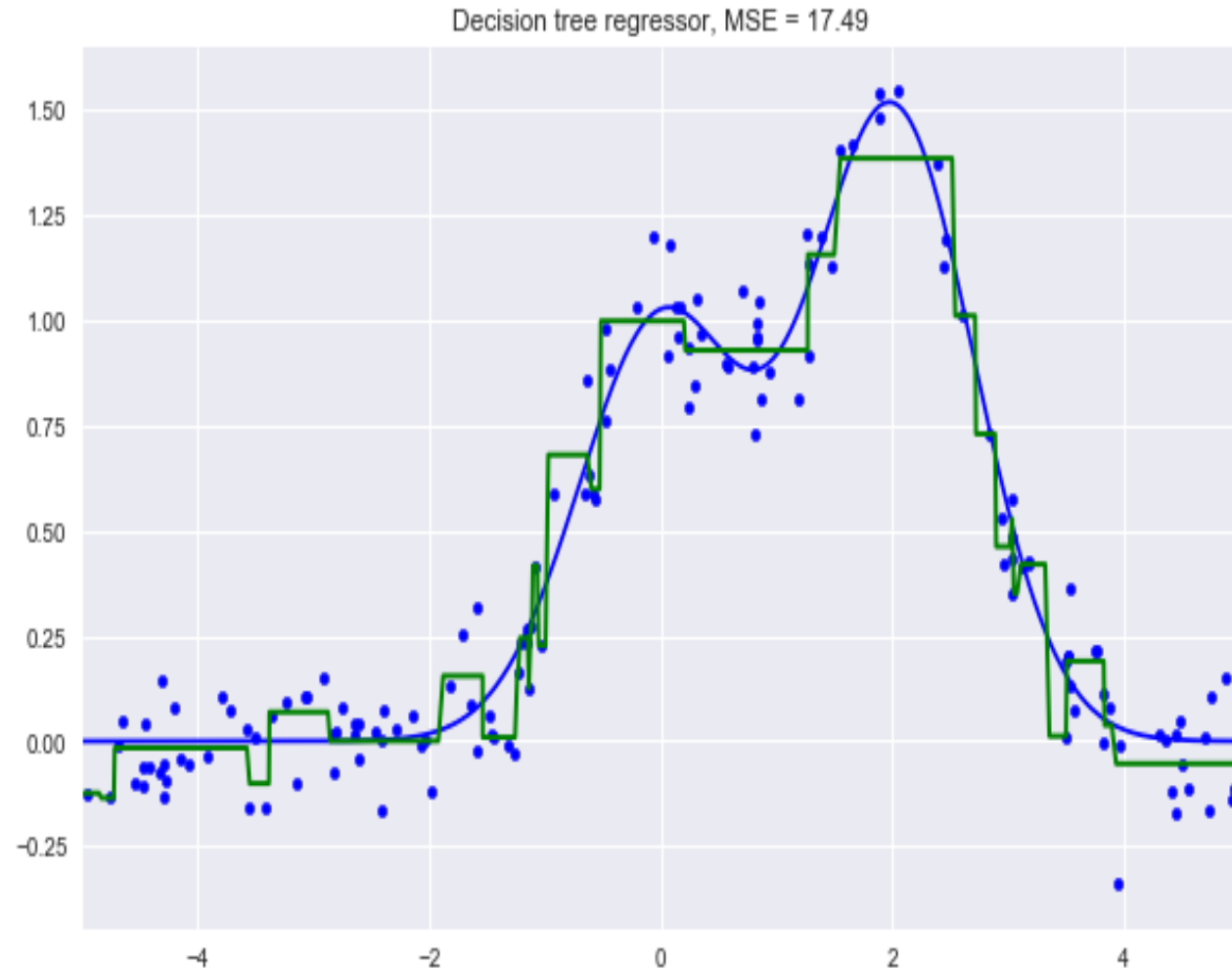
- Мы решаем задачу классификации с помощью решающего дерева.
- Ниже приведены разные варианты распределения классов в вершине ($[c1, c2, c3]$ означает, что в вершине $c1$ объектов первого класса, $c2$ объектов второго класса и $c3$ объектов третьего класса). Какой из них должен получить меньшее значение критерия информативности?
 1. $[95, 3, 2]$
 2. $[45, 45, 10]$
 3. $[33, 34, 33]$

Регрессия

- В регрессии в качестве функции потерь использует квадрат отклонения. В этом случае критерий информативности будет выглядеть так:
- $$H(X) = \min_{c \in Y} \frac{1}{|X|} \sum_{(x_i, y_i) \in X} (y_i - c)^2$$
- Минимум в этом выражении будет достигаться на среднем значении целевой переменной. Значит, критерий можно переписать в следующем виде:
- $$H(X) = \frac{1}{|X|} \sum_{(x_i, y_i) \in X} \left(y_i - \frac{1}{|X|} \sum_{(x_i, y_i) \in X} y_j \right)^2$$
- Таким образом, информативность вершины измеряется ее дисперсией – чем ниже разброс целевой переменной, тем лучше вершина
- Можно использовать и другие функции ошибки L.
- При выборе абсолютного отклонения мы получим в качестве критерия среднее абсолютное отклонение от медианы

Пример

- Сгенерируем данные, распределенные вокруг функции $f(x) = e^{-x^2} + 1.5 * e^{-(x-2)^2}$ с некоторым шумом
- Обучим на них дерево решений и изобразим прогнозы
- Дерево решений аппроксимирует зависимость в данных кусочно-постоянной функцией.



Классификация

Ошибка классификация

- Обозначим через p_k долю объектов класса $k \in \{1, \dots, K\}$, попавших в вершину X :
- $p_k = \frac{1}{|X|} \sum_{(x_i, y_i) \in X} [y_i = k]$
- Через k_* обозначим класс, чьих представителей оказалось больше всего среди объектов, попавших в данную вершину:
- $k_* = \underset{k}{\operatorname{argmax}} p_k$
- Рассмотрим индикатор ошибки как функцию потерь:
- $H(X) = \min_{c \in Y} \frac{1}{|X|} \sum_{(x_i, y_i) \in X} [y_i \neq c]$
- Оптимальным предсказанием тут будет наиболее популярный класс k_* -значит, критерий будет равен следующей доле ошибок:
- $H(X) = \min_{c \in Y} \frac{1}{|X|} \sum_{(x_i, y_i) \in X} [y_i \neq k_*] = 1 - p_{k_*}$

Критерий Джинни

$$H(X) = \sum_{k=1}^K p_k(1 - p_k)$$

Если $p_1 = 1, p_2 = \dots = p_k = 0$, то $H(X) = 0$

Энтропийный критерий

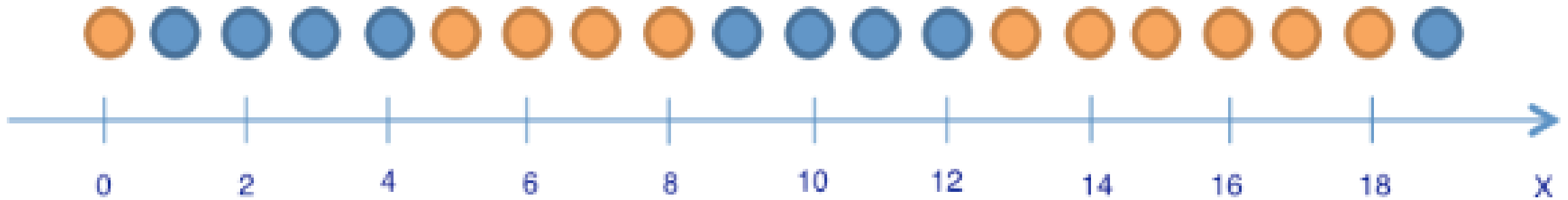
- Здесь можно вспомнить игру "20 вопросов".
- Один человек загадывает знаменитость, а второй пытается отгадать, задавая только вопросы, на которые можно ответить "Да" или "Нет" (опустим варианты "не знаю" и "не могу сказать").
- Какой вопрос отгадывающий задаст первым делом?
- Конечно, такой, который сильнее всего уменьшит количество оставшихся вариантов.
- К примеру, вопрос "Это Анджелина Джоли?" в случае отрицательного ответа оставит более 6 миллиардов вариантов для дальнейшего перебора (конечно, поменьше, не каждый человек – знаменитость, но все равно немало),
- а вот вопрос "Это женщина?" отсекает уже около половины знаменитостей.
- То есть, признак "пол" намного лучше разделяет выборку людей, чем признак "это Анджелина Джоли", "национальность-испанец" или "любит футбол".
- Это интуитивно соответствует понятию прироста информации, основанного на энтропии.

Энтропийный критерий

$$H(X) = - \sum_{k=1}^K p_k \ln p_k$$

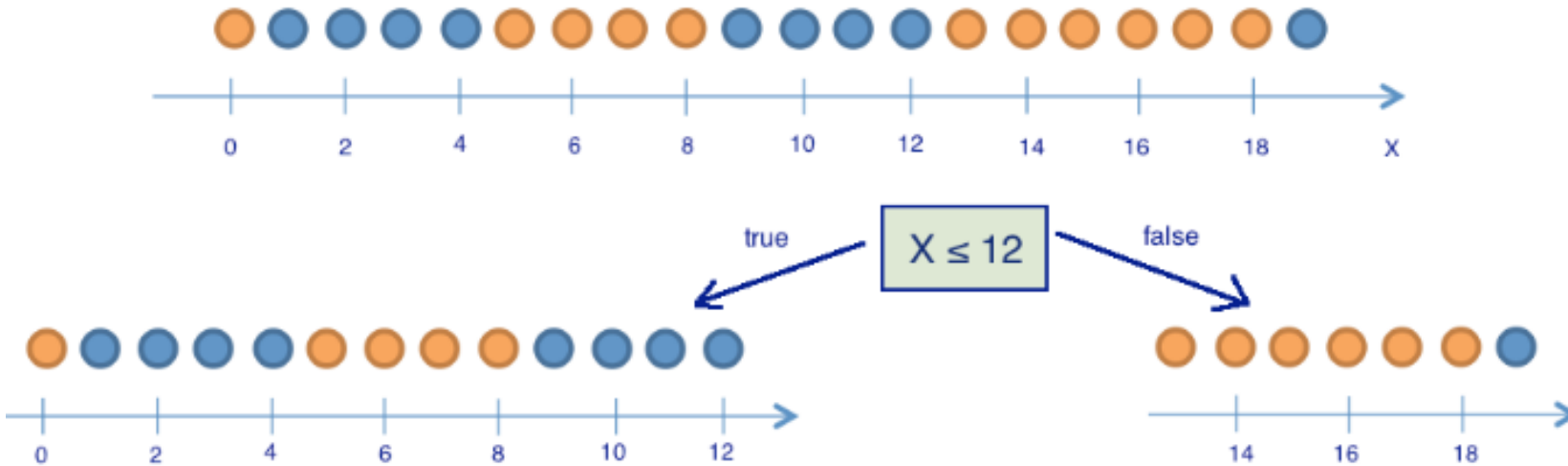
- Считаем, что $0 \ln 0 = 0$
- Если $p_1 = 1, p_2 = \dots = p_k = 0$, то $H(X) = 0$
- p_i — вероятность нахождения системы в i — ом состоянии.
- Интуитивно энтропия означает степень хаоса в системе, чем больше энтропия, тем больше хаос.

Энтропийный критерий



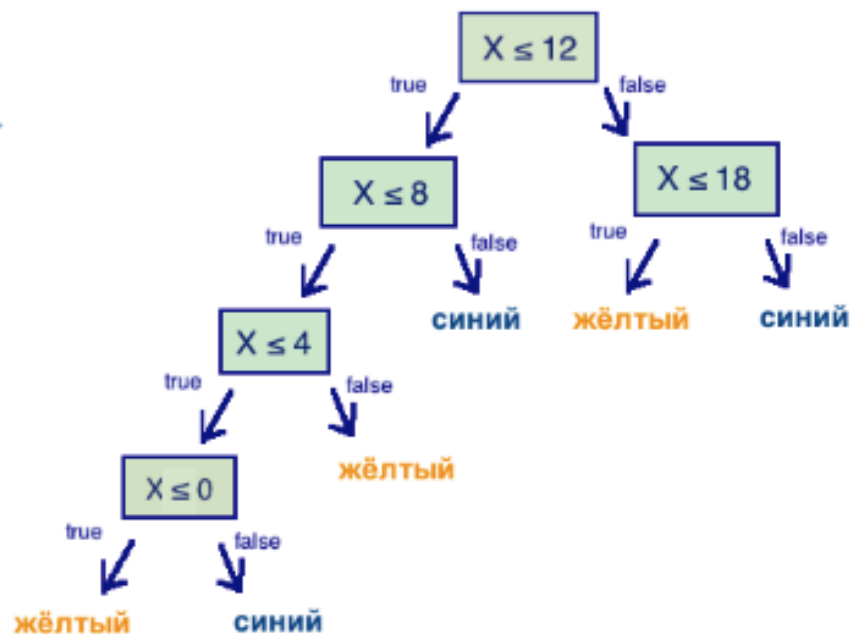
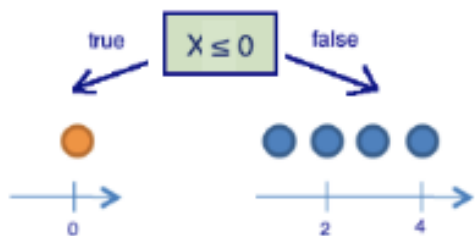
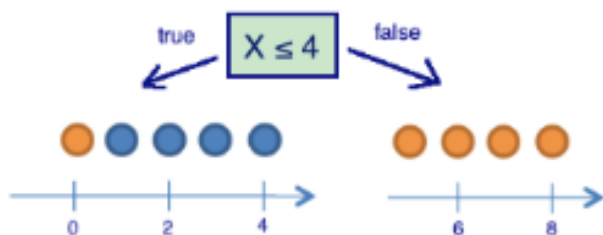
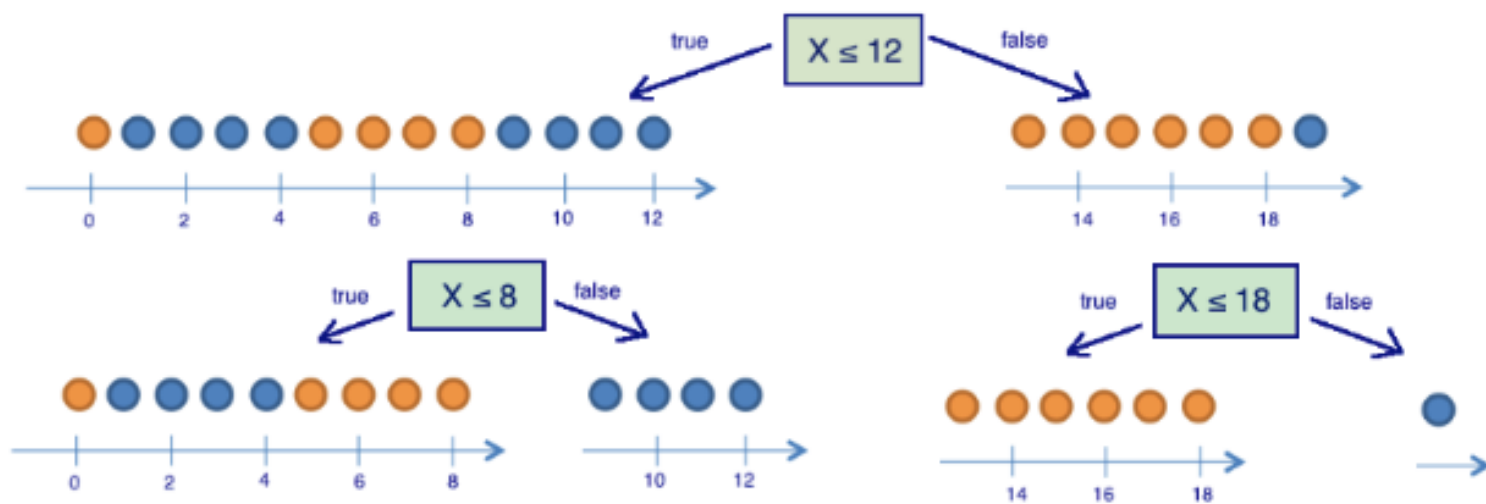
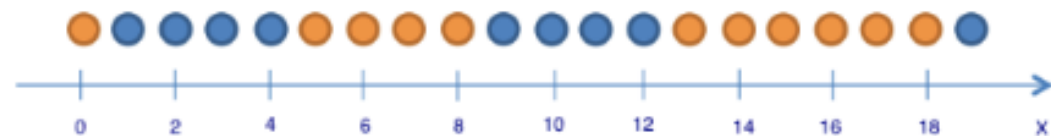
$$S_0 = -\frac{9}{20} \log_2 \frac{9}{20} - \frac{11}{20} \log_2 \frac{11}{20} \approx 1$$

Энтропийный критерий



$$S_1 = -\frac{5}{13} \log_2 \frac{5}{13} - \frac{8}{13} \log_2 \frac{8}{13} \approx 0.96$$

$$S_2 = -\frac{1}{7} \log_2 \frac{1}{7} - \frac{6}{7} \log_2 \frac{6}{7} \approx 0.6$$



Выводы

- Критерий ошибки измеряет разброс ответов после разбиения
- Выражается через критерий информативности
- В регрессии: MSE
- В классификации: критерий Джинни и энтропийный критерий

Критерий останова

- Как понять, разбивать вершину или делать листовой
- Способ борьбы с переобучением

Критерий Останова

- Все объекты в вершине относятся к одному классу
- Хороший критерий?

Критерий останова

- Чуть усовершенствуем
- В вершину попало $\leq n$ объектов
- При $n = 1$ получаем максимально переобученные деревья
- n должно быть достаточно, чтобы построить надежный прогноз
- Рекомендация: $n = 5$
- Очень грубый критерий, но хорошо работает в композициях.

Критерии останова

- Ограничение максимальной глубины дерева
- Ограничение минимального числа объектов в листе
- Ограничение максимального количества листьев в дереве
- Останов в случае, если все объекты в листе относятся к одному классу
- Требование, что функционал качества при дроблении улучшался как минимум на s процентов.

Стрижка деревьев

- Строим максимальное переобученное дерево
- Удаляем листья по некоторому критерию
- Пример: удаляем, пока улучшается ошибка
- Считается, что работает лучше критериев останова
- Трудоемкая процедура
- Имеет смысл только при использовании одного дерева
- В композициях деревьев достаточно простых критериев останова

Решающие деревья и категориальные признаки

$$\bullet [x^j \leq t]$$

- Только для вещественных и бинарных признаков!

N-АРНЫЕ ДЕРЕВЬЯ

Нужно сделать разбиение вершины m

Для бинарных или вещественных признаков

$$[x^j \leq t]$$

N-АРНЫЕ ДЕРЕВЬЯ

- Нужно сделать разбиение вершины m
- Для категориального признака x^j с n значениями $\{c_1, \dots, c_n\}$
- Пробуем разбить на n вершин
- В i -ю дочернюю вершину идут объекты с $x^j = c_i$

N-АРНЫЕ ДЕРЕВЬЯ

- Разбили X_m на n частей: X_1, \dots, X_n
- Критерий ошибки:
- $Q(X_m, j) = \sum_{i=1}^n \frac{|X_i|}{|X_m|} H(X_i) \rightarrow \min_j$
- Выбираем из всех разбиений то на котором меньше всего $Q(X_m, j)$ или $Q(X_m, j, t)$
- Высокий риск переобучения
- Подходит для очень больших выборок

Бинарные деревья

- Нужно сделать разбиение вершины m
- Для категориального признака $f(x^j)$ с n значениями $C = \{c_1, \dots, c_n\}$
- Разобьём множество значений на две части:
- $C = C_1 \cup C_2$
- Разбиение $[x^j \in C_1]$

Бинарные деревья

- Как разбить S ?
- Отсортируем значения
- Заменяем $c_{(1)}, \dots, c_{(n)}$ на $1, \dots, n$
- Будем работать как с вещественным признаком.

Сортировка значений

- Для бинарной классификации

$$\frac{\sum_{i \in X_m} [x_i^j = c_{(1)}][y_i = +1]}{\sum_{i \in X_m} [x_i^j = c_{(1)}]} \leq \dots \leq \frac{\sum_{i \in X_m} [x_i^j = c_{(n)}][y_i = +1]}{\sum_{i \in X_m} [x_i^j = c_{(n)}]}$$

Сортировка значений

- Для регрессии

$$\frac{\sum_{i \in X_m} [x_i^j = c_{(1)}] y_i}{\sum_{i \in X_m} [x_i^j = c_{(1)}]} \leq \dots \leq \frac{\sum_{i \in X_m} [x_i^j = c_{(n)}] y_i}{\sum_{i \in X_m} [x_i^j = c_{(n)}]}$$

Как быть с количественным признаком?

- Может быть так, что количественного признака очень много значений, и проверять каждое уникальное будет очень дорого?
- Как поступить?

Метод ближайших соседей

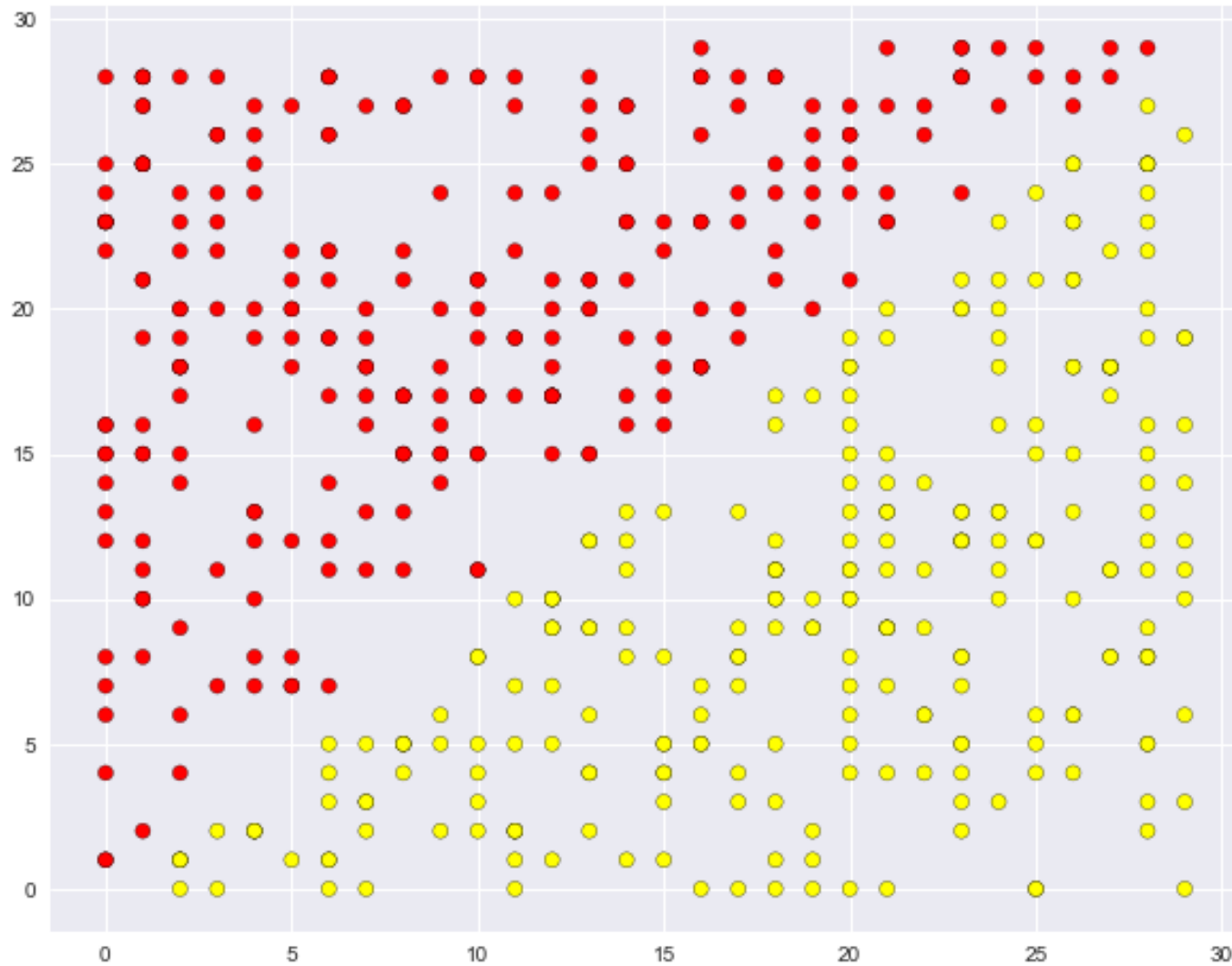


Интуитивно: посмотри на соседей, какие преобладают, таков и ты. Формально основой метода является гипотезой компактности: если метрика расстояния между примерами введена достаточно удачно, то схожие примеры гораздо чаще лежат в одном классе, чем в разных.

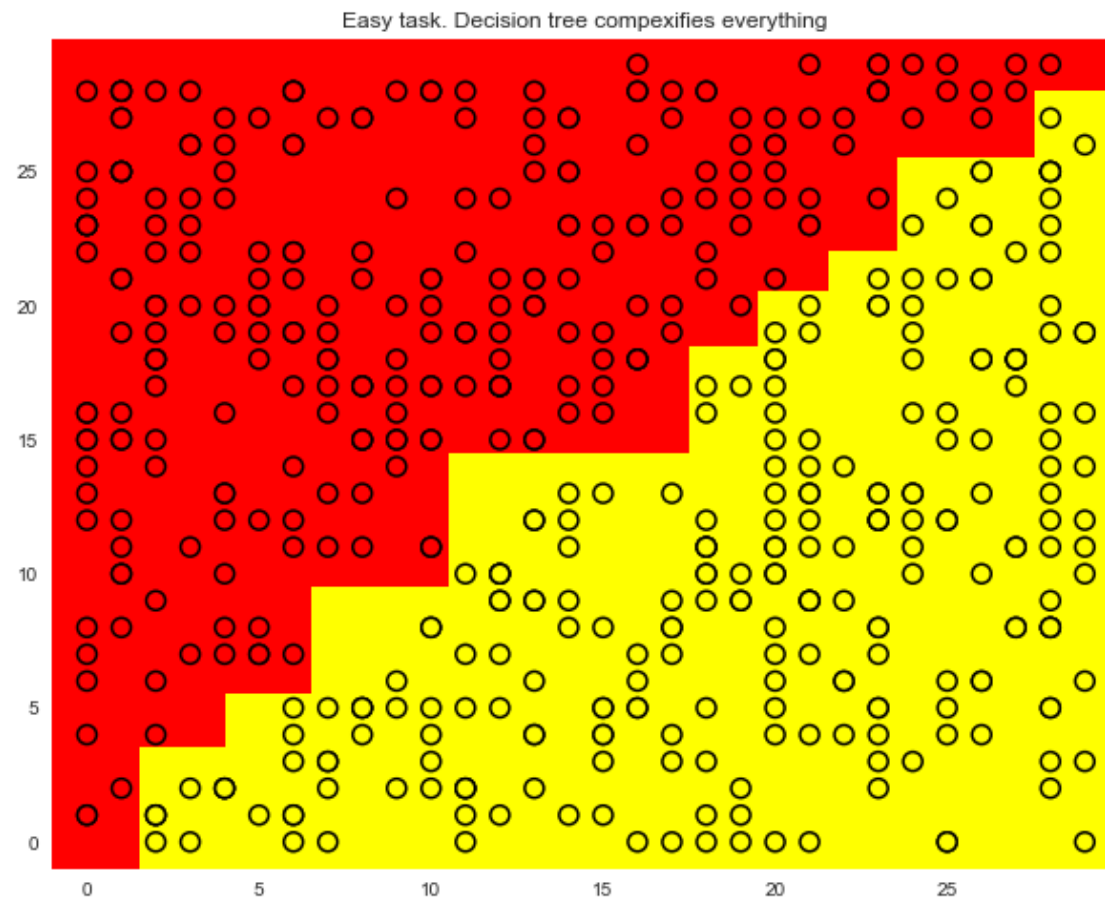
Алгоритм

- Для классификации каждого из объектов тестовой выборки необходимо последовательно выполнить следующие операции:
- Вычислить расстояние до каждого из объектов обучающей выборки
- Отобрать k объектов обучающей выборки, расстояние до которых минимально
- Класс классифицируемого объекта – это класс, наиболее часто встречающийся среди k ближайших соседей.

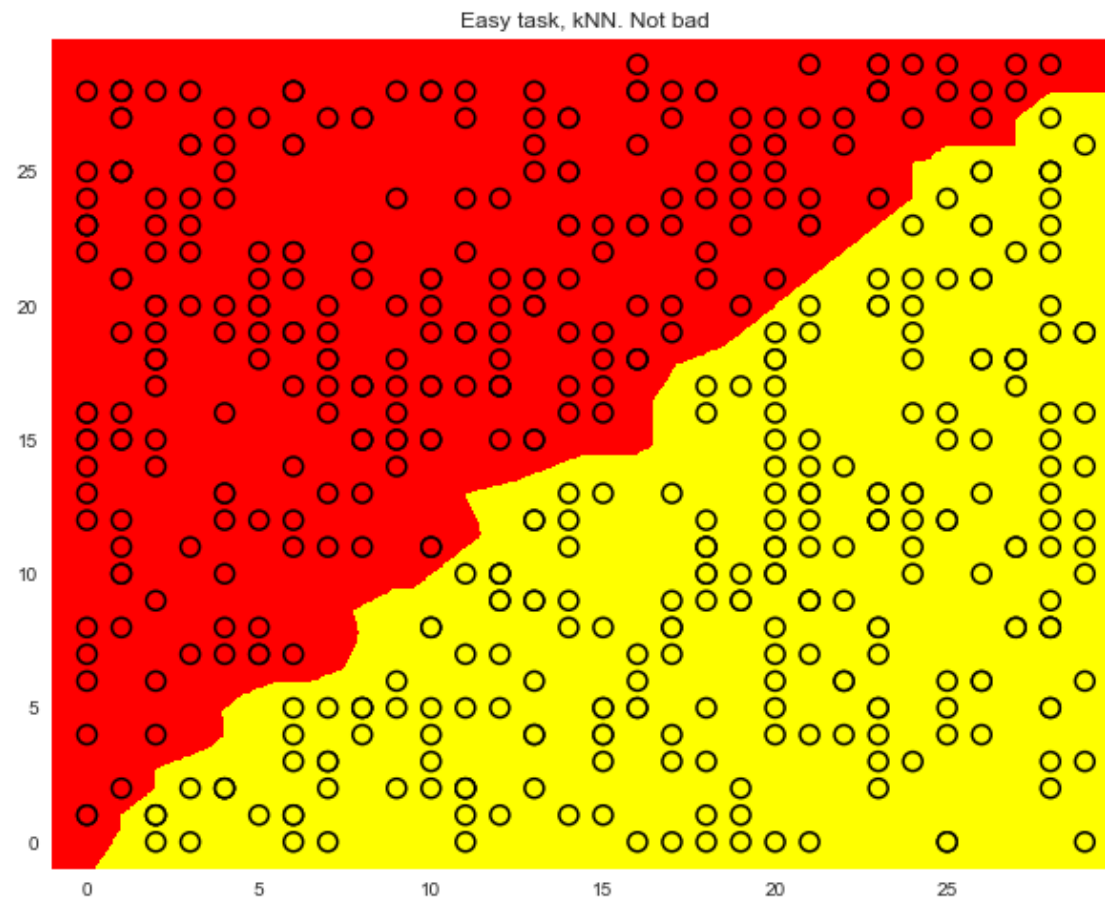
Пример, где алгоритмы делают все не оптимально



Какой алгоритм?



Метод ближайших соседей



Плюсы дерева решений

- Порождение четких правил классификации, понятных человеку, например, "если возраст < 25 и интерес к мотоциклам, то отказать в кредите". Это свойство называют интерпретируемостью модели;
- Деревья решений могут легко визуализироваться, то есть может "интерпретироваться" (строгое определения я не видел) как сама модель (дерево), так и прогноз для отдельного взятого тестового объекта (путь в дереве);
- Быстрые процессы обучения и прогнозирования;
- Малое число параметров модели;
- Поддержка и числовых, и категориальных признаков.

Минусы дерева решений

- У порождения четких правил классификации есть и другая сторона: деревья очень чувствительны к шумам во входных данных, вся модель может кардинально измениться, если немного изменится обучающая выборка (например, если убрать один из признаков или добавить несколько объектов), поэтому и правила классификации могут сильно изменяться, что ухудшает интерпретируемость модели;
- Разделяющая граница, построенная деревом решений, имеет свои ограничения (состоит из гиперплоскостей, перпендикулярных какой-то из координатной оси), и на практике дерево решений по качеству классификации уступает некоторым другим методам;
- Необходимость отсекаать ветви дерева (pruning) или устанавливать минимальное число элементов в листьях дерева или максимальную глубину дерева для борьбы с переобучением. Впрочем, переобучение - проблема всех методов машинного обучения;
- Нестабильность. Небольшие изменения в данных могут существенно изменять построенное дерево решений. С этой проблемой борются с помощью ансамблей деревьев решений (рассмотрим далее);

Минусы дерева решений

- Проблема поиска оптимального дерева решений (минимального по размеру и способного без ошибок классифицировать выборку) NP-полна, поэтому на практике используются эвристики типа жадного поиска признака с максимальным приростом информации, которые не гарантируют нахождения глобально оптимального дерева;
- Модель умеет только интерполировать, но не экстраполировать (это же верно и для леса и бустинга на деревьях). То есть дерево решений делает константный прогноз для объектов, находящихся в признаковом пространстве вне параллелепипеда, охватывающего все объекты обучающей выборки. В нашем примере с желтыми и синими шариками это значит, что модель дает одинаковый прогноз для всех шариков с координатой > 19 или < 0 .

Плюсы метода ближайших соседей

- Простая реализация
- Можно адаптировать под нужную задачу выбором метрики или ядра (в двух словах: ядро может задавать операцию сходства для сложных объектов типа графов, а сам подход kNN остается тем же).
- Неплохая интерпретация, можно объяснить, почему тестовый пример был классифицирован именно так. Хотя этот аргумент можно атаковать: если число соседей большое, то интерпретация ухудшается (условно: "мы не дали ему кредит, потому что он похож на 350 клиентов, из которых 70 – плохие, что на 12% больше, чем в среднем по выборке").

Минусы метода ближайших соседей

- Метод считается быстрым в сравнении, например, с композициями алгоритмов, но в реальных задачах, как правило, число соседей, используемых для классификации, будет большим (100-150), и в таком случае алгоритм будет работать не так быстро, как дерево решений;
- Если в наборе данных много признаков, то трудно подобрать подходящие веса и определить, какие признаки не важны для классификации/регрессии;
- Зависимость от выбранной метрики расстояния между примерами. Выбор по умолчанию евклидова расстояния чаще всего ничем не обоснован. Можно отыскать хорошее решение перебором параметров, но для большого набора данных это отнимает много времени;
- Нет теоретических оснований выбора определенного числа соседей - только перебор (впрочем, чаще всего это верно для всех гиперпараметров всех моделей). В случае малого числа соседей метод чувствителен к выбросам, то есть склонен переобучаться;
- Как правило, плохо работает, когда признаков много, из-за "проклятия размерности".