Машинное обучение

Лекция 8. Логические алгоритмы классификации.

Катя Тузова

Разбор летучки

Обзор уже известных подходов

1. Метрический
   - K ближайших соседей
   - Кластеризация
2. Линейный
   - Градиентный спуск
   - Метод опорных векторов

Логические закономерности

$X^l = (x_i, y_i)_{i=1}^l$ - обучающая выборка.

Логическая закономерность (правило) – предикат $\beta : X \to \{0, 1\}$, который удовлетворяет двум требованиям:

- Интерпретируемость
- Информативность относительно одного из классов $c \in Y$

Алгоритм: $a(X, \beta) \to y$

Интерпретируемость

- Записывается на естественном языке
- Зависит от небольшого числа признаков

Интерпретируемость

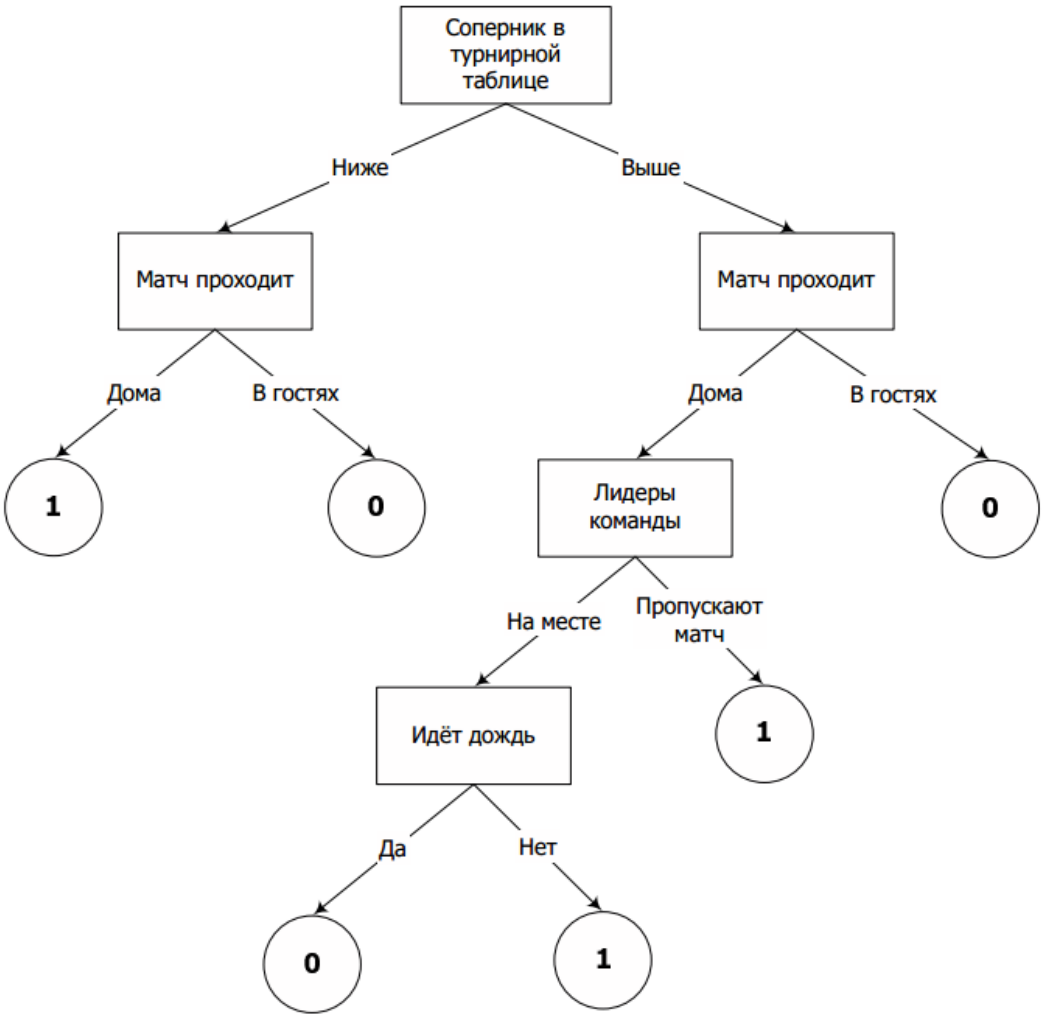
Медицинская диагностика:

- Температура выше n°
- Есть кашель

Кредитный скоринг:

- Заработная плата выше, чем n руб.
- Кредит не больше, чем m руб.

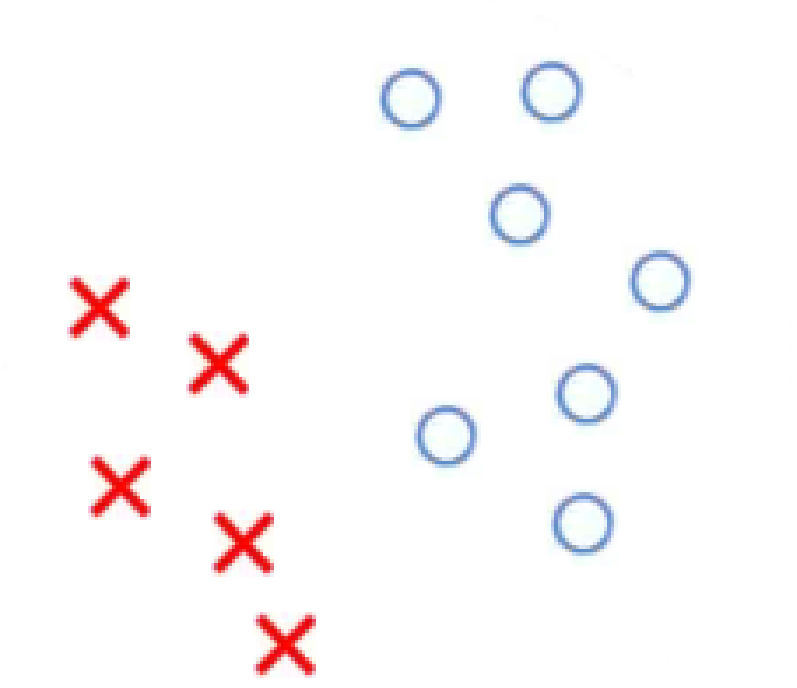
Интерпретируемость



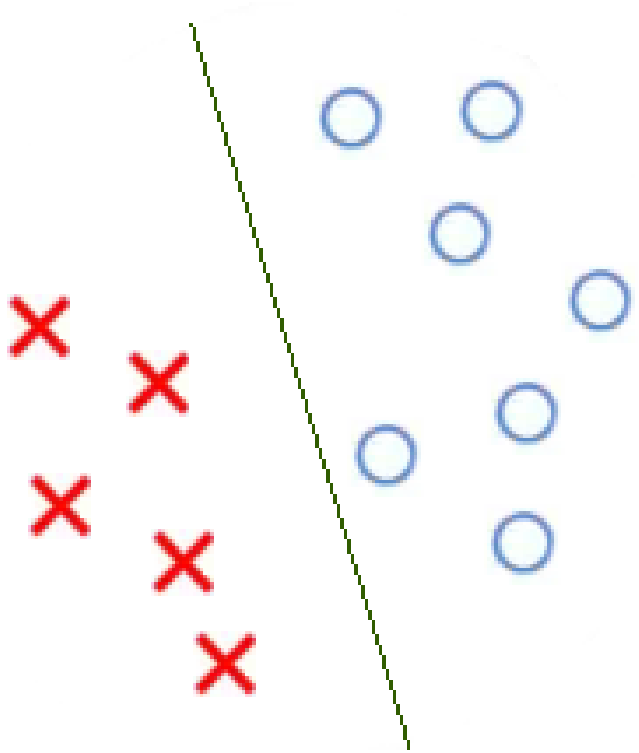
Информативность

- Максимизировать количество правильно распознанных объектов класса c
  $p_c(\beta) = \#\{x_i : \beta(x_i) = 1, y_i = c\} \to \max$
- Минимизировать количество объектов, ошибочно классифицированных как класс c
  $n_c(\beta) = \#\{x_i : \beta(x_i) = 1, y_i \neq c\} \to \min$

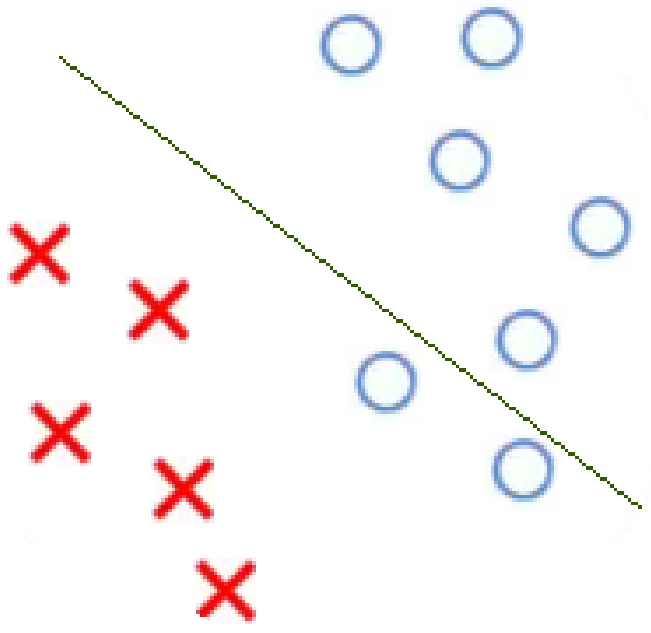
Информативность



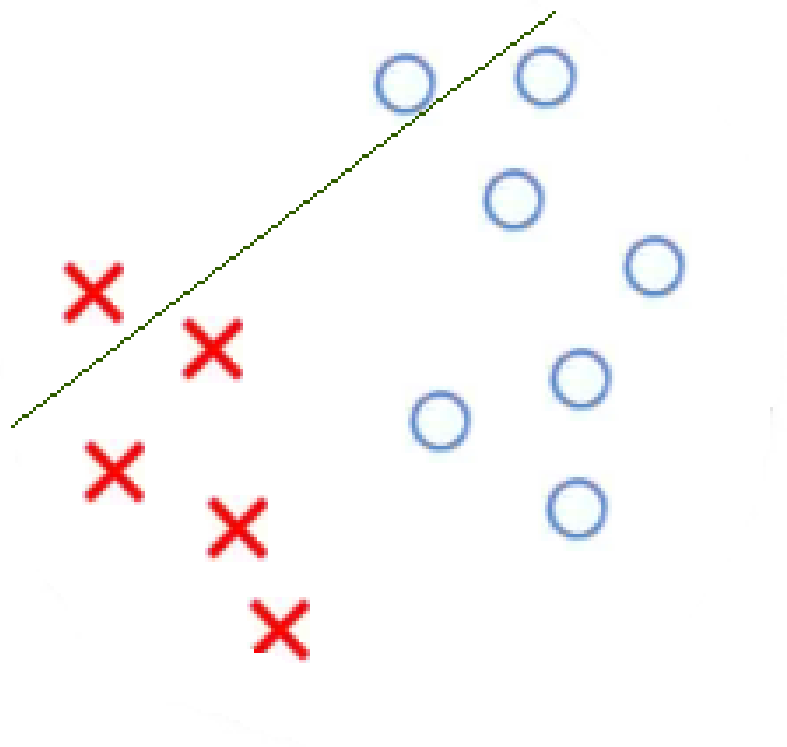
Информативность



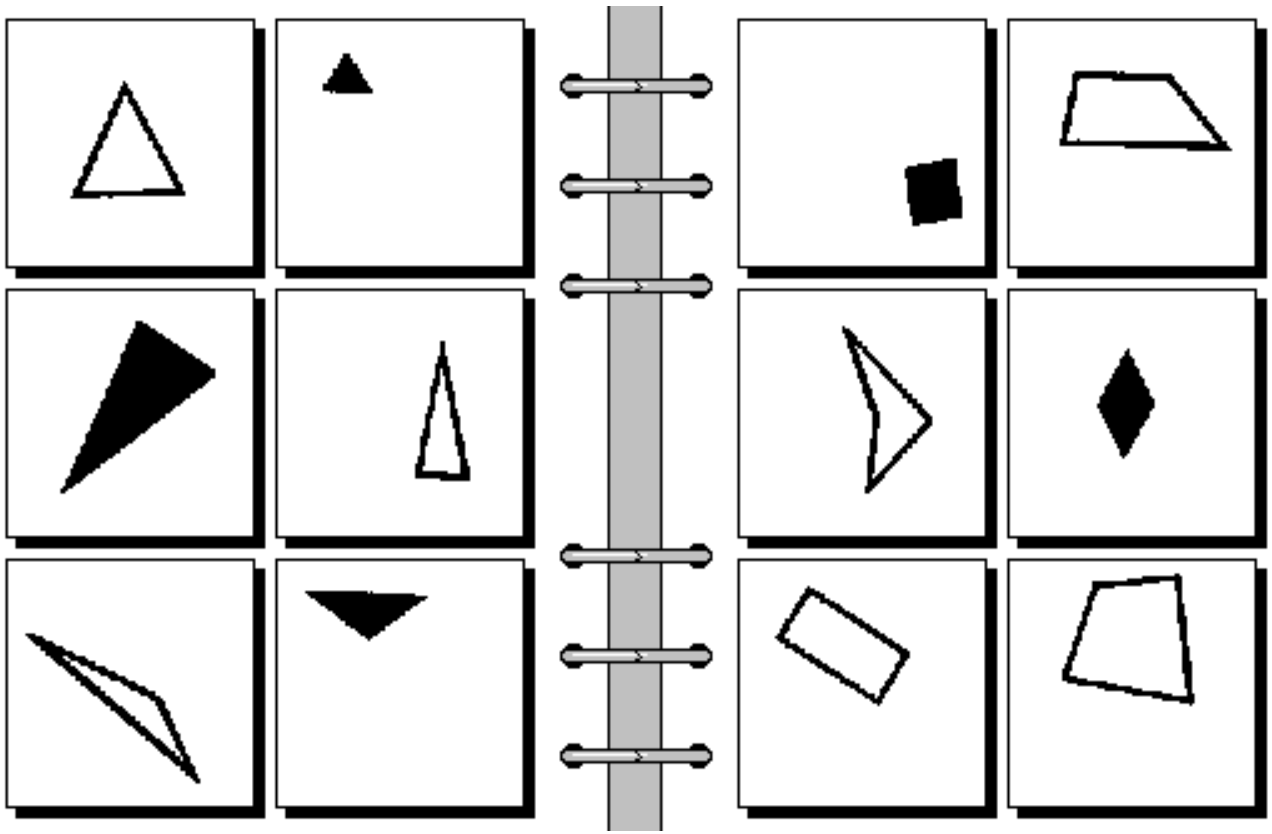
Информативность



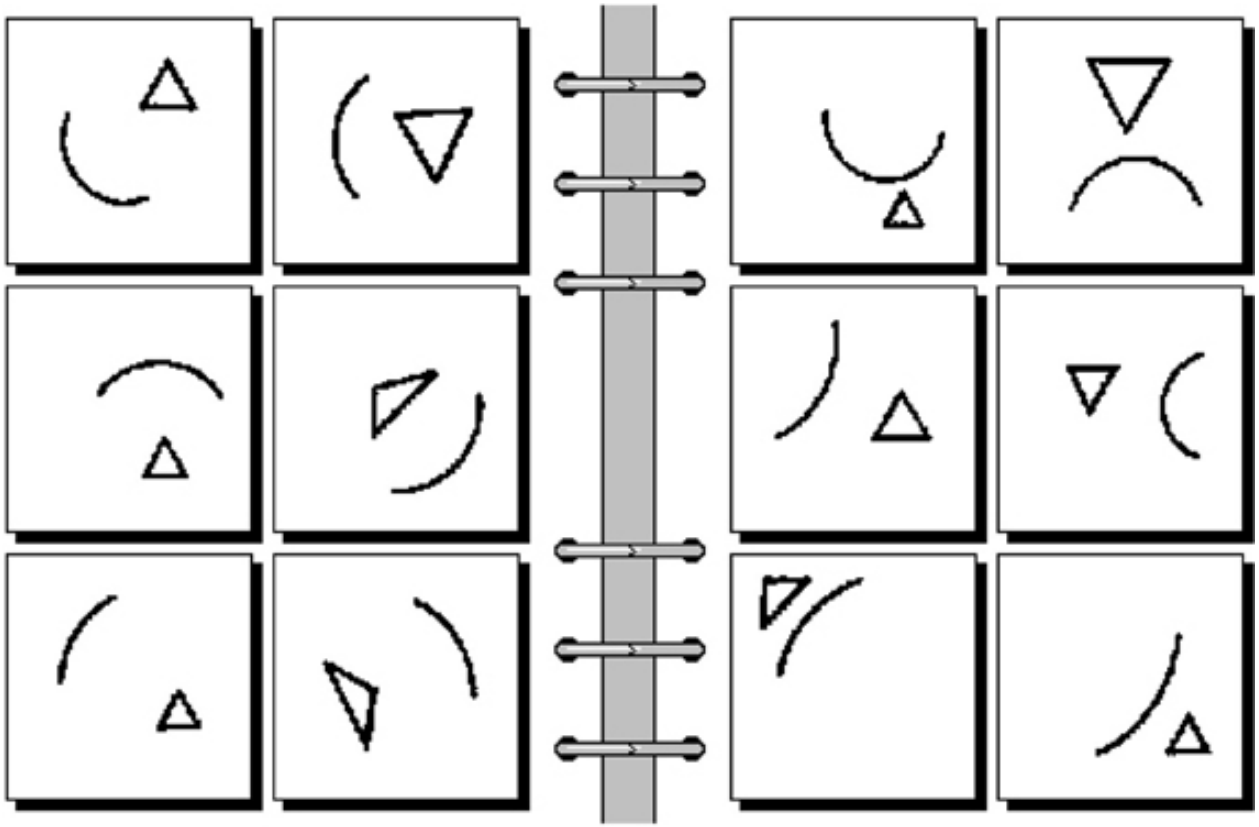
Информативность



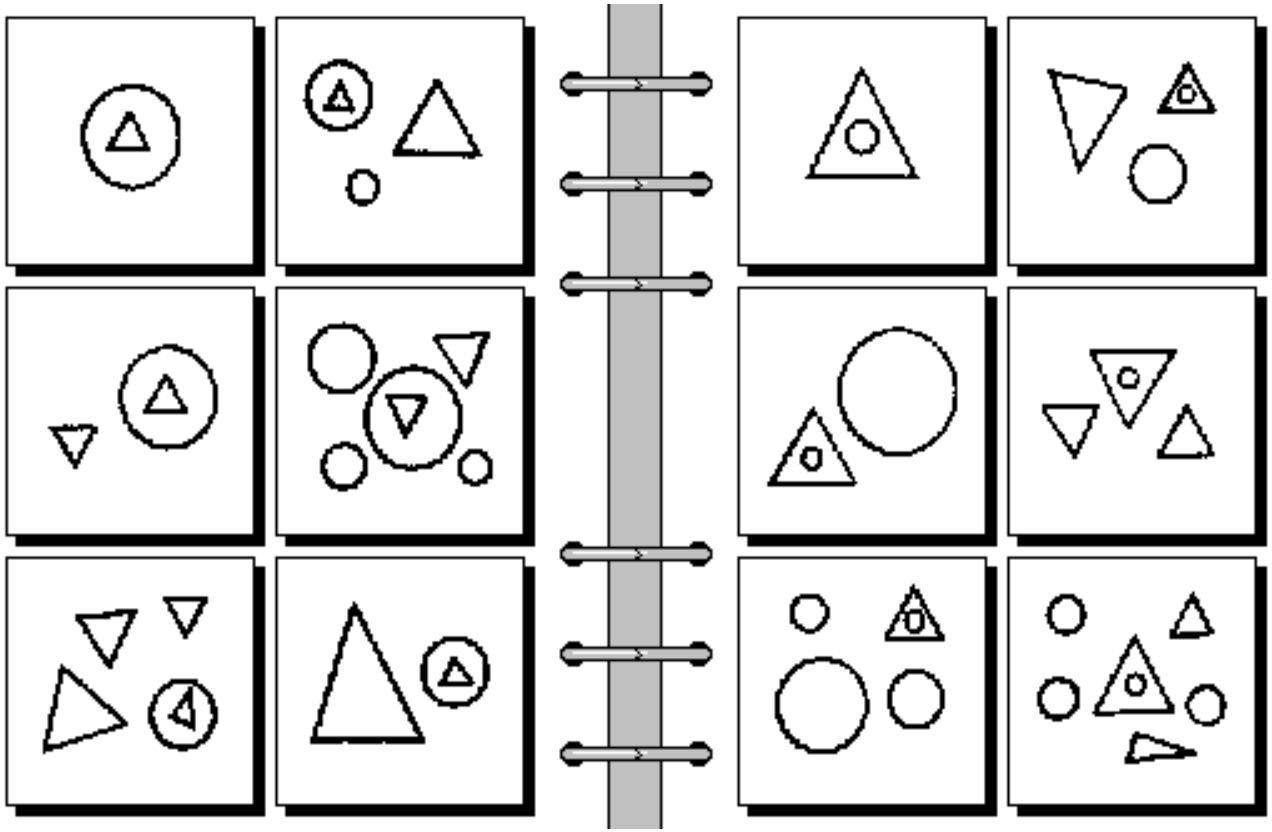
Поиск закономерности



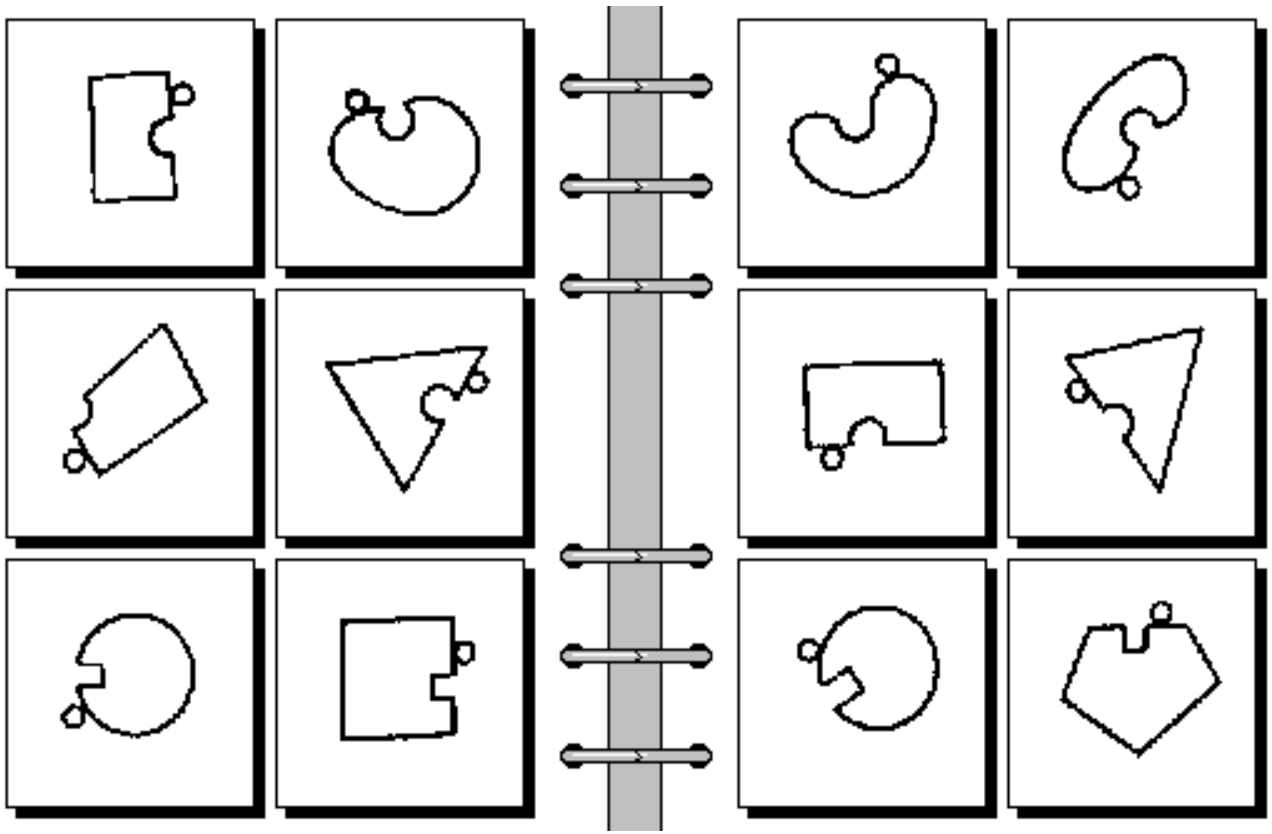
Поиск закономерности



Поиск закономерности



Поиск закономерности



Основные вопросы

- Как изобретать признаки?
- Какого вида закономерности $\beta(x)$ нужны?
- Как определять информативность?
- Как выбирать закономерности?
- Как объединять закономерности в алгоритм?

Как изобретать признаки

Какие признаки выбрать для задачи кредитного скоринга?

Виды правил

Виды правил

- Пороговое условие (decision stump)
  $\beta(x) = [f_j(x) \le a_j]$ или $[a_j \le f_j(x) \le b_j]$
- Конъюнкция J пороговых условий
  $\beta(x) = \bigwedge\limits_{j \in J} [a_j \le f_j(x) \le b_j]$
- Синдром – выполнение не менее d условий из J
  $\beta(x) = \left[\sum\limits_{j \in J} [a_j \le f_j(x) \le b_j] \ge d \right]$

Оценивание информативности

Идея: Хотим получить один критерий из двух:

$$tp(\beta) = \#\{x_i : \beta(x_i) = 1, y_i = c\} \to \max$$
$$fp(\beta) = \#\{x_i : \beta(x_i) = 1, y_i \neq c\} \to \min$$

Очевидные свертки:

- $I(tp, fp) = \frac{tp}{tp+fp} \to \max$
- $I(tp, fp) = tp - fp \to \max$
- $I(tp, fp) = tp - Cfp \to \max$

Пример свертки двух критериев

Пусть число примеров искомого класса 200 и число остальных объектов 100

tp	fp	$tp - fp$	$tp - 5fp$	$\frac{tp}{tp+fp}$
50	0	50	50	1
100	50	50	-150	0.6
50	9	41	5	0.84
5	0	5	5	1

Используемые критерии информативности

- Энтропийный информационный критерий $IGain(p,n) = h(\frac{P}{l}) - \frac{tp+tp}{l} h(\frac{tp}{tp+fp}) - \frac{l-tp-fp}{l} h(\frac{P-tp}{l-tp-fp}) \to \max$
  где $h(q) = -q \log_2 q - (1-q) \log_2 (1-q)$
- Fisher's Exact Test
  $IStat(p,n) = -\frac{1}{l} \log_2 \frac{C_P^{tp} C_N^{fp}}{C_{P+N}^{tp+fp}} \to \max$
- Критерий Джини
  $I(\beta, X^l) = \#\{(x_i, x_j) : \beta(x_i) = \beta(x_j), y_i = y_j\}$
- Критерий Донского
  $I(\beta, X^l) = \#\{(x_i, x_j) : \beta(x_i) \neq \beta(x_j), y_i \neq y_j\}$

Поиск информативных закономерностей

Input: Обучающая выборка X^l
Output: Множество закономерностей $\mathcal{B}$

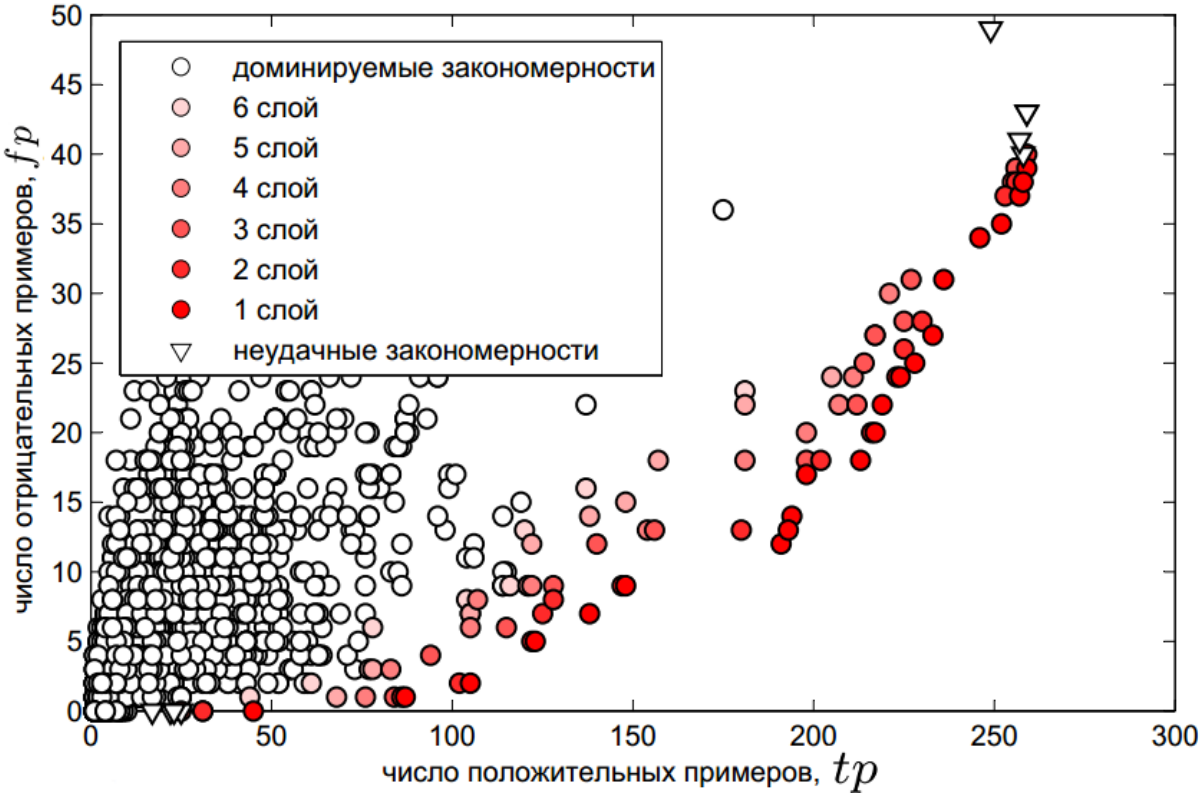
Выбрать начальное множество $\mathcal{B}$
Повторять пока правила улучшаются:
$\mathcal{B}' =$ множество модификаций правил $\beta \in \mathcal{B}$
Удалить слишком похожие правила из $\mathcal{B} \cup \mathcal{B}'$
Оценить инфоматативность правил $\beta \in \mathcal{B}'$
$\mathcal{B} =$ наиболее информативные правила из $\mathcal{B} \cup \mathcal{B}'$

Примеры:

- Генетические алгоритмы
- Метод ветвей и границ
- Стохастический локальный поиск

Критерий Парето

Парето-фронт – множество недоминируемых закономерностей (правее и ниже точек нет)



Картинка с machinelearning.ru

Вопрос

Как собрать классификатор из закономерностей?

Решающий список

Множество классов $c_1, c_2, \ldots, c_T \in Y$

Идея:
Возьмем $\beta_1(x), \beta_2(x), \ldots, \beta_T(x)$ закономерностей и будем по порядку применять на объекте. Как только предикат β_i сработал – вернем соответствующий класс c_i.

Решающий список

Множество классов $c_1, c_2, \ldots, c_T \in Y$

Идея:
Возьмем $\beta_1(x), \beta_2(x), \ldots, \beta_T(x)$ закономерностей и будем по порядку применять на объекте. Как только предикат β_i сработал – вернем соответствующий класс c_i.

$E(\beta_i, X^l) = \frac{fp(\beta_i)}{fp(\beta_i)+tp(\beta_i)} \to \min$ – доля ошибок β_i на X^l

Т.к. каждое правило принимает окончательное решение $\Rightarrow$ ошибка правила равна ошибке всего алгоритма

Жадный алгоритм для решающего списка

Input: X^l, семейство предикатов $\mathcal{B}$, T_{max}, I_{min}, E_{max}, l_0
Output: Решающий список $\{\beta_i, c_i\}_{i=1}^{T}$

$U = X^l$
Для всех $i = 1, 2, \ldots, T_{max}$
Выбрать класс c_i
$\beta_i = \max\limits_{E(\beta,U) \leq E_{max}} I(\beta, U)$
Если $I(\beta_i, U) < I_{min}$ продолжить
$U = \{x \in U : \beta_i(x) = 0\}$
Если $|U| \leq l_0$ выйти

Наблюдения

- Низкое качество классификации
- Разные стратегии выбора класса c_i
- Подбор параметров

Наблюдения

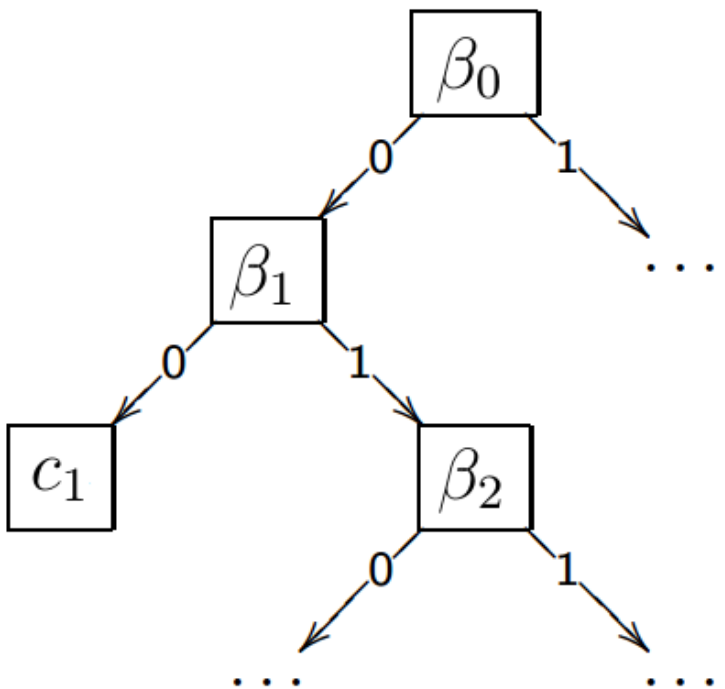
- Низкое качество классификации
- Разные стратегии выбора класса c_i
- Подбор параметров

Параметр E_{max} влияет на сложность списка:
$E_{max} \downarrow \Rightarrow tp(\beta_i) \downarrow, T \uparrow$
$E_{max} \uparrow$, $T_{max} \uparrow \Rightarrow$ переобучение

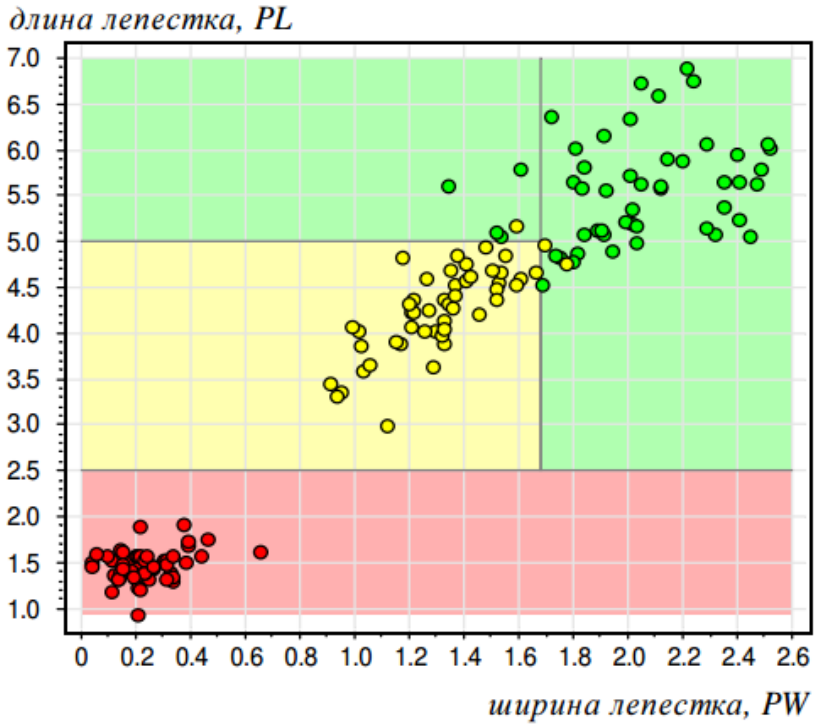
Бинарное решающее дерево

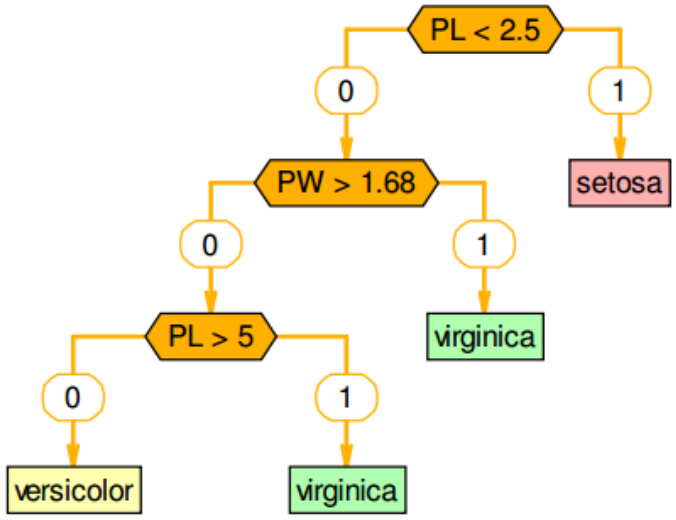
Бинарное решающее дерево – алгоритм классификации a(x), задающийся бинарным деревом:

- $\forall v \in V_{inner} \to \beta_v : X \to \{0, 1\}$, $\beta \in \mathcal{B}$
- $\forall v \in V_{leaf} \to$ имя класса $c_v \in Y$



Пример решающего дерева

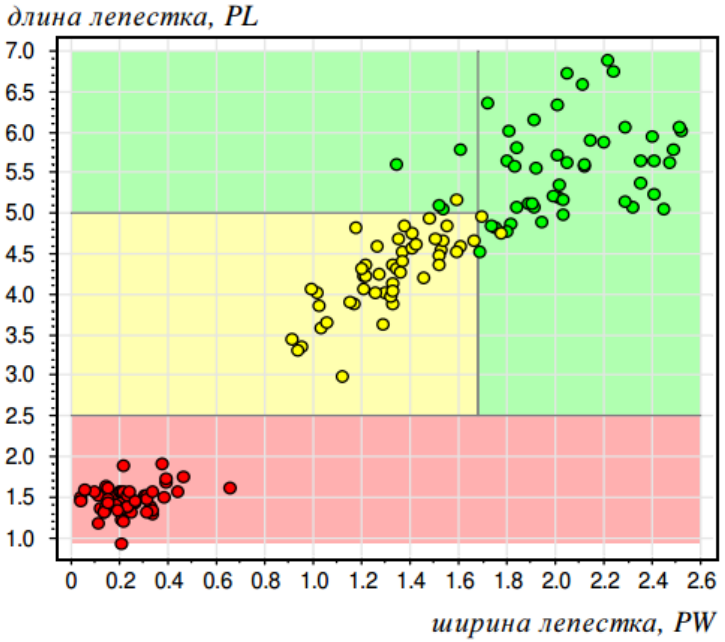


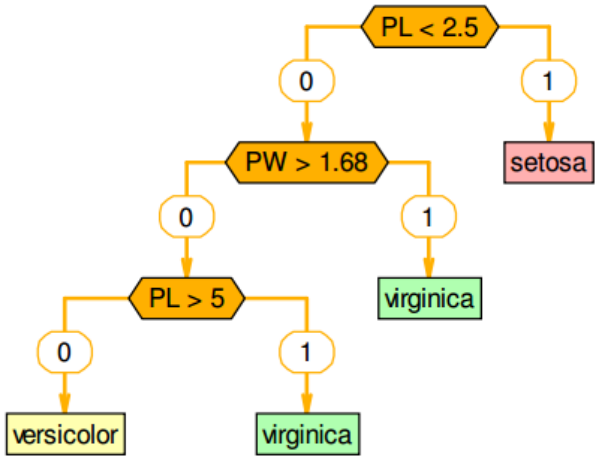


В осях двух самых информативных признаков (из 4) два класса разделились без ошибок, на третьем 3 ошибки.

Картинка с machinelearning.ru

Набор конъюнкций





setosa	$r_1(x) = [PL \leqslant 2.5]$
virginica	$r_2(x) = [PL > 2.5] \wedge [PW > 1.68]$
virginica	$r_3(x) = [PL > 5] \wedge [PW \leqslant 1.68]$
versicolor	$r_4(x) = [PL > 2.5] \wedge [PL \leqslant 5] \wedge [PW < 1.68]$

Картинка с machinelearning.ru

Алгоритм построения ID3

LearnID3 (U, $\mathcal{B}$):

Если все объекты из U лежат в одном классе $c \in Y$:

Вернуть новый лист v, $c_v = c$

$$\beta^* = \max_{\beta \in \mathcal{B}} I(\beta, U)$$

$$U_0 = \{x \in U : \beta(x) = 0\}$$
$$U_1 = \{x \in U : \beta(x) = 1\}$$

Если $U_0 = \varnothing$ или $U_1 = \varnothing$:

Вернуть v, c_v = Majority(U)

Создать новую внутреннюю вершину v: $\beta_v = \beta$

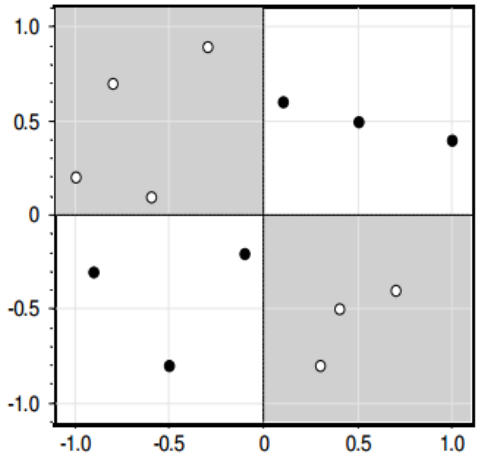
L_v = LearnID3 (U_0, $\mathcal{B}$)
R_v = LearnID3 (U_1, $\mathcal{B}$)
Вернуть v

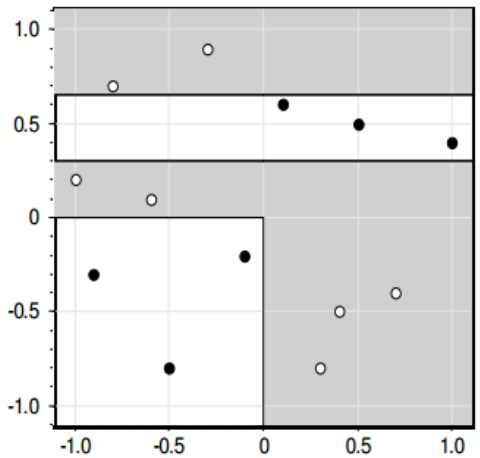
Плюсы и минусы

+ Интерпретируемость и простота классификации
+ Допустимы разнотипные данные и данные с пропусками
+ Не бывает отказов от классификации
+ Трудоёмкость линейна по длине выборки
+ Можно варьировать множество $\mathcal{B}$

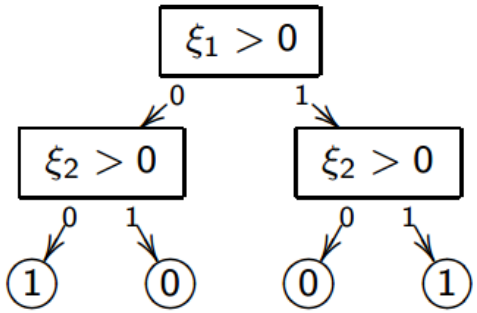
- Жадный ID3 сильно переобучается
- Высокая чувствительность к шуму, к составу выборки, к критерию информативности
- Чем дальше v от корня, тем меньше надёжность выбора β_v , c_v

Переобучение

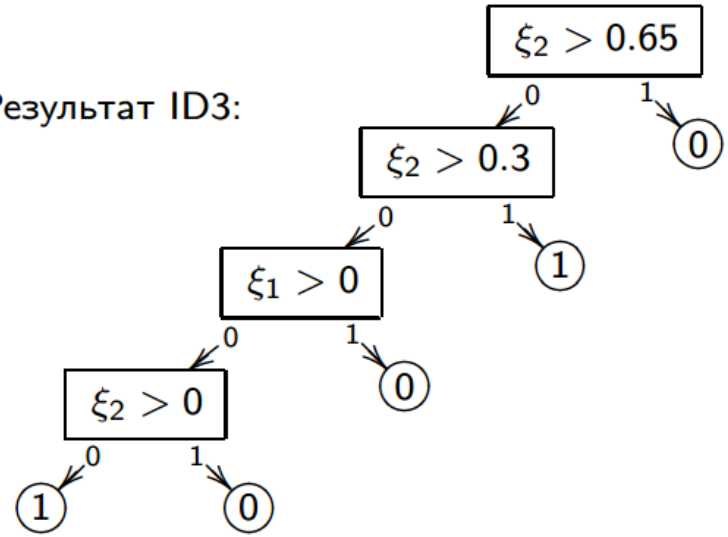




Оптимальное дерево для задачи XOR:



Результат ID3:



Картинка с machinelearning.ru

Подрезание дерева C4.5

X^k – независимая контрольная выборка, $k \approx 0.5l$

Для всех $v \in V_{inner}$:

S_v = подмножество объектов X^k, дошедших до v

Если $S_v = \varnothing$:

Вернуть новый лист v, c_v = Majority(U)

Вычислить число ошибок четырьмя способами:

$r(v)$ – поддеревом, растущим из вершины v

$r_L(v)$ – левой дочерней вершины L_v

$r_R(v)$ – правой дочерней вершины R_v

$r_c(v)$ – к классу $c \in Y$

В зависимости от того, какое из них минимально:

Сохранить поддерево v

Заменить поддерево v поддеревом L_v

Заменить поддерево v поддеревом R_v

Заменить v листом $c_v = \min\limits_{c \in Y} r_c(v)$

Композиции алгоритмов

- Можно использовать результаты нескольких алгоритмов, а не одного.
- В подавляющем большинстве случаев композиция алгоритмов даёт лучший результат, нежели какой-то один

Вопрос

Почему это работает?

Почему это работает

- Ошибки алгоритмов взаимно компенсируются
- На одних объектах хорошо работают одни алгоритмы, на других – другие

Композиции алгоритмов

- Желательно использовать принципиально разные алгоритмы, наборы признаков
- Не всегда два оптимальных алгоритма в смеси дадут оптимальный результат
- Чем больше параметров – тем больше шанс переобучиться

Bootstrapping

Выборка X^l

Сгенерируем много датасетов так: будем из X выбирать l точек с замещением, т.е. в новом датасете некоторые точки будут повторяться

Баггинг

Выборка X^l

Сделаем путем bootstrapping M датасетов размера l, потом обучим M моделей, а потом усредним

$y(x) = \frac{1}{M} \sum_{i=1}^{l} y_i(x)$

Бустинг

- Предположим, что у нас есть возможность обучать какую-нибудь простую модель на подмножестве данных.
- Обучили модель, посмотрели, где она хорошо работает, обучили следующую модель на том подмножестве, где она работает плохо, повторили.

Random forest

Голосование деревьев классификации, $Y = \{-1, +1\}$

$$a(t) = \text{sign}\, \frac{1}{T} \sum_{t=1}^{T} b_t(x)$$

- Каждое дерево $b_t(x)$ обучается по случайной выборке с повторениями
- В каждой вершине признак выбирается из случайного подмножества $\sqrt{n}$ признаков
- Признаки и пороги выбираются по критерию Джини

На следующей лекции

– Байесовские методы классификации