

Машинное обучение

Лекция 8. Логические алгоритмы классификации.

Катя Тузова

Обзор уже известных подходов

1. Метрический

- К 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 $\beta : X \rightarrow \{0, 1\}$, который удовлетворяет двум требованиям:

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

Алгоритм: $a(X, \beta) \rightarrow y$

Интерпретируемость

- Записывается на естественном языке
- Зависит от небольшого числа признаков

Интерпретируемость

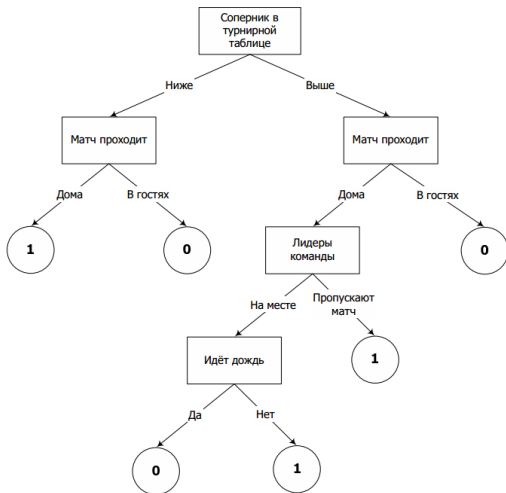
Медицинская диагностика:

- Температура выше n°
- Есть кашель

Кредитный скоринг:

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

Интерпретируемость



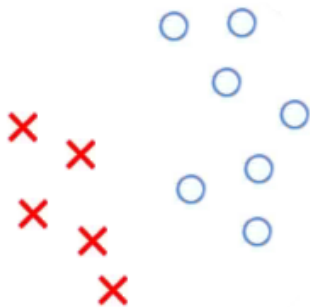
- Максимизировать количество правильно распознанных объектов класса c

$$tp(\beta) = \# \{x_i : \beta(x_i) = 1, y_i = c\} \rightarrow \max$$

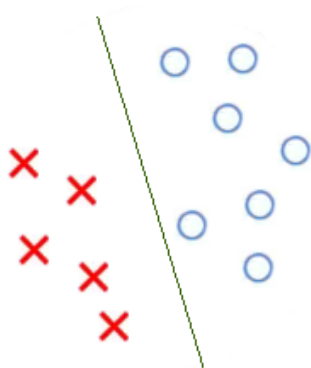
- Минимизировать количество объектов, ошибочно классифицированных как класс c

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

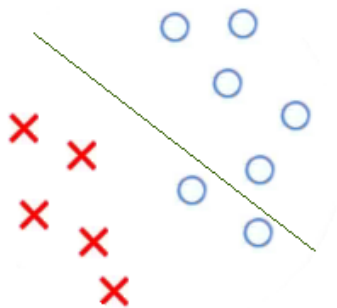
Информативность



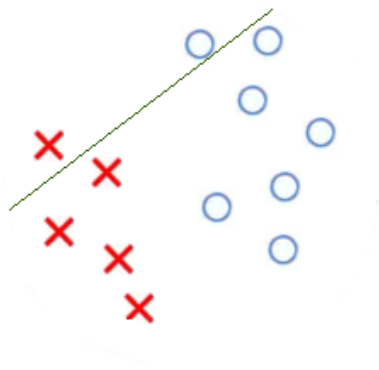
Информативность



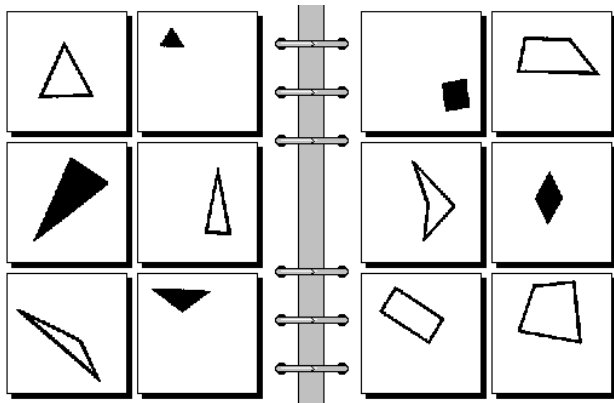
Информативность



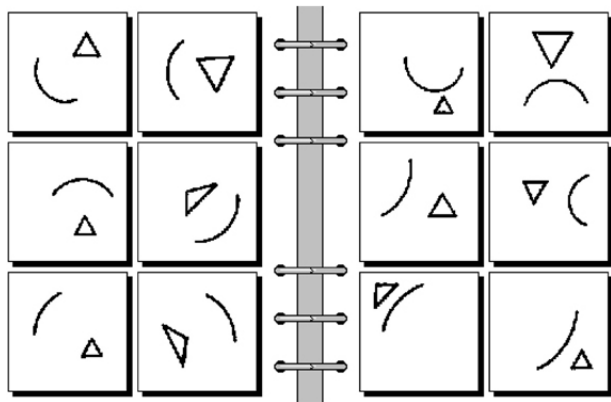
Информативность



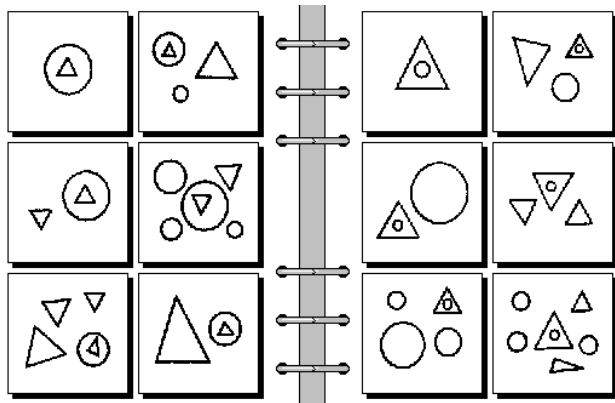
Поиск закономерности



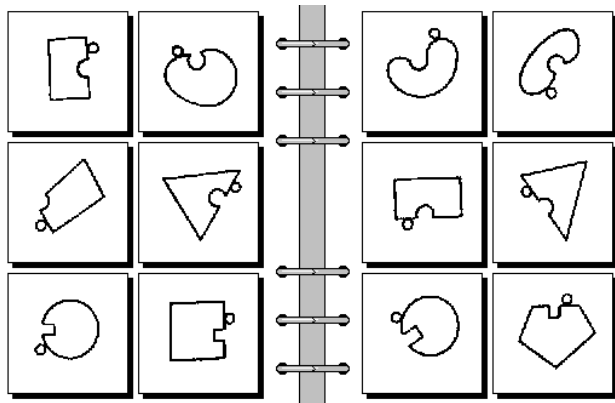
Поиск закономерности



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)

$$\beta(x) = [f_j(x) \leq a_j] \text{ или } [a_j \leq f_j(x) \leq b_j]$$

- Конъюнкция J пороговых условий

$$\beta(x) = \bigwedge_{j \in J} [a_j \leq f_j(x) \leq b_j]$$

- Синдром – выполнение не менее d условий из J

$$\beta(x) = \left[\sum_{j \in J} [a_j \leq f_j(x) \leq b_j] \geq d \right]$$

Оценивание информативности

Идея: Хотим получить один критерий из двух:

$$tp(\beta) = \# \{x_i : \beta(x_i) = 1, y_i = c\} \rightarrow \max$$

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

Очевидные свертки:

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

$$- I(tp, fp) = tp - fp \rightarrow \max$$

$$- I(tp, fp) = tp - Cfp \rightarrow \max$$

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

$$IGain(tp, fp) = h\left(\frac{P}{l}\right) - \frac{tp+fp}{l}h\left(\frac{tp}{tp+fp}\right) - \frac{l-tp-fp}{l}h\left(\frac{P-tp}{l-tp-fp}\right) \rightarrow \max$$

где $h(q) = -q \log_2 q - (1 - q) \log_2(1 - q)$

- Критерий Джини

$$IGini(tp, fp) = IGain(tp, fp), \text{ где } h(q) = 4q(1 - q)$$

- Fisher's Exact Test

$$IStat(tp, fp) = -\frac{1}{l} \log_2 \frac{C_P^{tp} C_N^{fp}}{C_{P+N}^{tp+fp}} \rightarrow \max$$

P - общее количество объектов искомого класса

N - количество остальных объектов

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 информативность правил $\beta \in \mathcal{B}'$

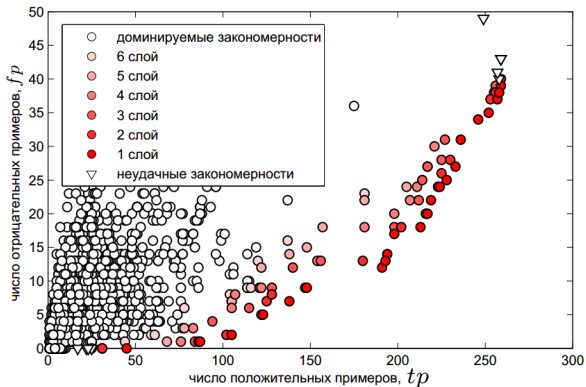
\mathcal{B} = наиболее информативные правила из $\mathcal{B} \cup \mathcal{B}'$

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

Как собрать классификатор из закономерностей?

Решающий список

Идея:

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

Решающий список

Идея:

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

$$E(\beta_i, X^l) = \frac{fp(\beta_i)}{fp(\beta_i)+tp(\beta_i)} \rightarrow \min - \text{доля ошибок } \beta_i \text{ на } X^l$$

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 $i = 1, 2, \dots, T_{max}$

Выбрать класс c_i

$$\beta_i = \max_{E(\beta, U) \leq E_{max}} I(\beta, U)$$

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:

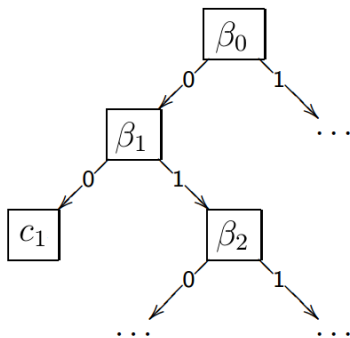
$$E_{max} \downarrow \Rightarrow tp(\beta_i) \downarrow, T \uparrow$$

$$E_{max} \uparrow \Rightarrow \text{underfitting}$$

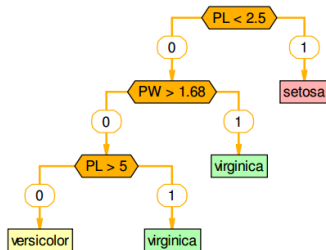
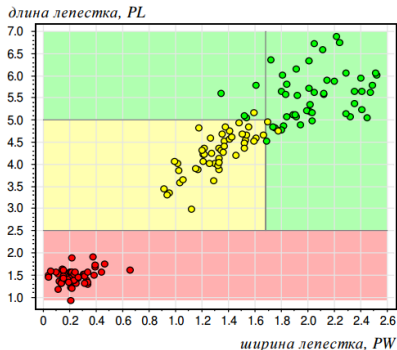
Бинарное решающее дерево

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

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



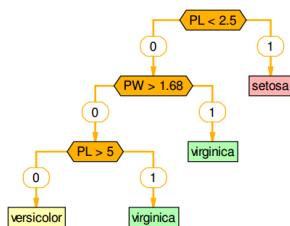
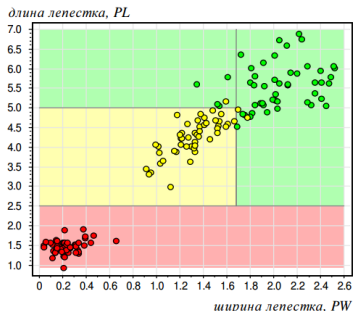
Пример решающего дерева



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

Картинка с machinelearning.ru

Набор конъюнкций



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

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 $U_0 = \emptyset$ или $U_1 = \emptyset$:

Вернуть v , $c_v = \text{Majority}(U)$

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

$$L_v = \text{LearnID3}(U_0, \mathcal{B})$$

$$R_v = \text{LearnID3}(U_1, \mathcal{B})$$

Вернуть v

Критерий информативности

- Критерий Джини

$$I(\beta, X^l) = \# \{(x_i, x_j) : \beta(x_i) = \beta(x_j), y_i = y_j\}$$

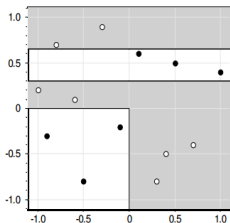
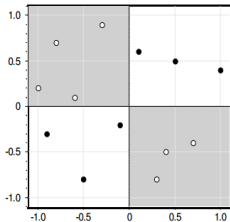
- Критерий Донского

$$I(\beta, X^l) = \# \{(x_i, x_j) : \beta(x_i) \neq \beta(x_j), y_i \neq y_j\}$$

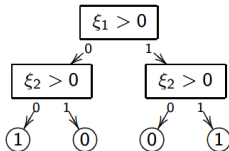
Плюсы и минусы

- + 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

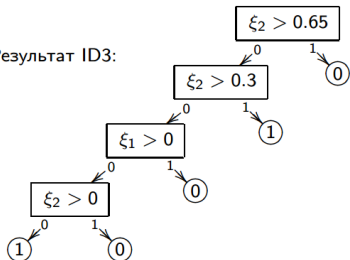
Переобучение



Оптимальное дерево для задачи XOR:



Результат ID3:



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 $S_v = \emptyset$:

Вернуть новый лист v , $c_v = \text{Majority}(U)$

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 $c_v = \min_{c \in Y} r_c(v)$

Композиции алгоритмов

- Можно использовать результаты нескольких алгоритмов, а не одного.
- Часто композиция алгоритмов может давать не худший или даже лучший результат

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 объектов с повторениями

Выборка X^l

Сделаем путем bootstrapping M датасетов размера l ,
потом обучим M моделей, а потом усредним
 $y(x) = \text{Majority}(y_i(x))$

Random forest

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

$$a(t) = \text{Majority}(b_t(x))$$

- Каждое дерево $b_t(x)$ обучается по случайной выборке с повторениями
- В каждой вершине предикат выбирается из случайного подмножества \sqrt{n} предикатов

- 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.

На следующей лекции

- Байесовские методы классификации