Data Science для решения задач информационной безопасности

Лекция 2. Классификация: основные понятия. Пример алгоритма классификации: Random Forest

---

Павел Владимирович Слипенчук

24 сентября 2019

Москва, МГТУ им.Бауманка,
каф.ИУ-8, КИБ

План лекции

1. Признак. Вектор признаков Классы. Обучающая и тестовая выборки. Задача классификации, классификатор(оценщик)
2. Тестовая выборка. Отсечка. tp, tn, fp, fn. Расчёт полноты и точность
3. Дерево решений. Лес решений
4. Построение случайного дерева решений
5. Случайный Лес (Random Forest)

Признак. Вектор признаков Классы. Обучающая и тестовая выборки. Задача классификации, классификатор(оценщик)

Признак, вектор, класс

Признак x_i– определенное значение. Категориальное, сравнимое, или числовое: целочисленное, булевое, или дробное.

Вектор признаков $\mathbf{x} = (x_1, x_2, \ldots x_n)$ – вектор, каждое значение которого является *признаком*.

Класс (метка) y – значение (как правило целочисленное), присваиваемое какому-либо вектору признаков: $y \mapsto \mathbf{x}$

А в чем физический смысл?

Пример

- x_1 – сумма транзакции [в рублях]
- x_2 – возраст клиента [в годах]
- x_3 – пол клиента [булевый: 1 – мужской, 0 – женский]
- x_4 – MCC код[1]
- $y = 1$ – операция мошенническая (фродовая); $y = 0$ – легитимная.

$$
\begin{array}{ll}
0 \mapsto (3234, 25, 1, 1731) & 0 \mapsto (2540, 55, 0, 1731) \\
1 \mapsto (18400, 45, 0, 3137) & 0 \mapsto (2540, 55, 0, 1731) \\
1 \mapsto (903, 19, 0, 4121) & 0 \mapsto (1875, 45, 0, 4121) \\
0 \mapsto (854, 21, 1, 4121) & 1 \mapsto (702, 21, 0, 4121) \\
1 \mapsto (903, 19, 0, 4121) & 0 \mapsto (1875, 45, 0, 4121) \\
0 \mapsto (28400, 41, 1, 3137) & 0 \mapsto (25040, 55, 0, 1731)
\end{array}
$$

[1] ***Merchant Category Code*** – номер деятельности компании при осуществлении безналичной оплаты. Например **1731** означает оплату за электроэнергию, **3137** – покупка авиабилетов, **4121** – такси

Замечание

В отличие от таблицы, представленной на слайде №3, в данных на реальных задачах *вектор признаков* может состоять из 200 и более признаков: $\mathbf{x} = (x_1, x_2, ..., x_{200}, ...)$

Замечание

Иногда можно встретить ещё два понятия: *характеристика* и *контрибутор*.

Характеристика – это какая-либо информация из которой можно получить один или более признаков.

Контрибутор (контрибьютер) – это совокупность признаков (возможно один), который вносит определённый вклад в скоринговую модель.

Однако можно считать эти термины устаревшими, так как *Feature Extraction* ввело понятие «*сырые данные*» (raw data) и убило понятие характеристика; а ансамбли сделали ненужным термин контрибутор.

Функция высшего порядка

Функция высшего порядка – это функция, принимающая в качестве аргументов хотя бы одну другую функцию и/или возвращающая в качестве выхода функцию.

ДЗ. см.так же:

1. *функциональное программирование*
2. *декоратор*, *фабрика декораторов*

Классификация. Постановка задачи

Определим *неупорядоченную совокупность*:

$$U_{fit} = \{y \mapsto \mathbf{x}\} \tag{1}$$

где y – это *класс*, а $\mathbf{x}$ – это *вектор признаков*. Множество (1) будем называть ***обучающей выборкой***.

Функцию, отображающую вектор $\mathbf{x}$ в значения на интервале $[0, 1]$ будем называть ***функцией скоринга***:

$$\hat{y} := score(\mathbf{x}); \hat{y} \in [0, 1] \tag{2}$$

Значение $\hat{y}$ называют ***откликом*** *вектора признаков* $\mathbf{x}$.

Обучением (в задаче классификации) будем называть процесс получения *функции скоринга*(2) из *обучающей выборки*(1). Таким образом можно определить *функцию высшего порядка fit*:

$$score := fit(U_{fit}) \tag{3}$$

Как правило множество U_{fit} задается парой:

$$U_{fit} := (\mathsf{X}, \mathsf{y}) \tag{4}$$

где X и y – это **упорядоченные** совокупности (списки) из *векторов признаков* $\mathbf{x}$ и соответствующих им *классов* y.

То есть:

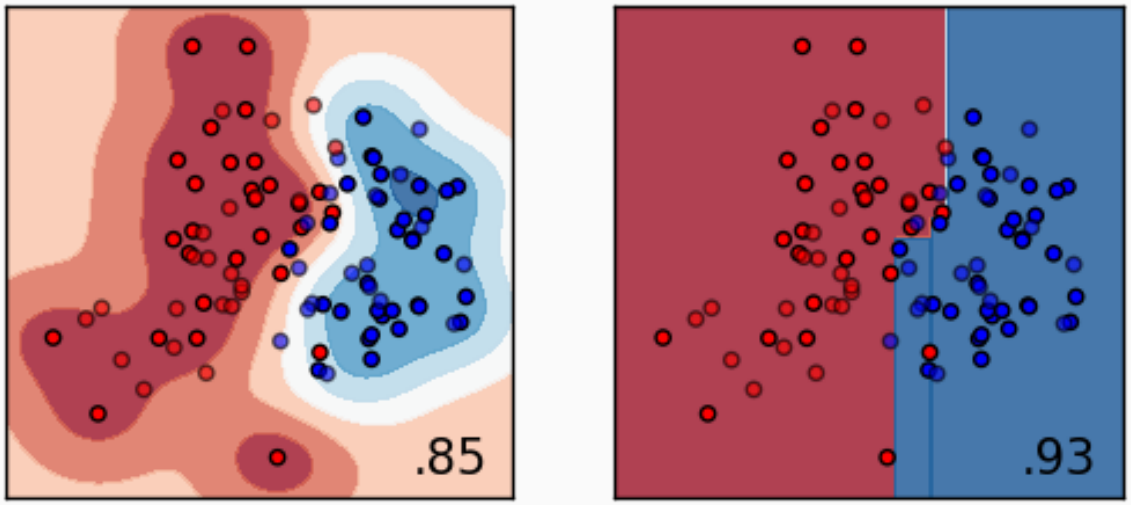
$$\mathsf{X} \stackrel{def}{=} \left((x_1^1, x_2^1, ..., x_n^1), (x_1^2, x_2^2, ..., x_n^2), ..., (x_1^m, x_2^m, ..., x_n^m)\right) \tag{5}$$

$$\mathsf{y} \stackrel{def}{=} (y^1, y^2, ..., y^m) \tag{6}$$

где верхние цифры – это индексы, а не степени.

Классификация

Функция *fit* (см. (3) на слайде №7) возвращает функцию *score*, которая размечает каждую точку пространства признаков на величину от 0 до 1. Чем ближе эта величина к 1, тем «более вероятно» что это класс 1, чем ближе к 0 – тем более вероятно что класс 0.



Замечание

Иногда для удобства fit возвращает не в диапазоне $[0, 1]$, а в диапазоне $[-1, 1]$.

Замечание

U_{fit} в общем случае это именно неупорядоченная совокупность, а не множество. Во множестве ни один элемент не может быть более одного раза, а в неупорядоченной совокупности – может. Для простоты в дальнейшем U_{fit} будем называть множеством.

Замечание

Иногда функция *fit* представляет собой *ленивое вычисление*[2] и выдаёт *score* почти мгновенно. И уже для каждого конкретного $\mathbf{x}$ функция $score(\mathbf{x})$ рассчитывает *отклик* $\hat{y}$ на основании *обучающей выборки* U_{fit}

[2] ***Ленивые вычисления*** (lazy evaluation, отложенные вычисления) – метод программной разработки функций, в которой вычисления откладывают до тех пор, пока не понадобится их результат.

Без использования алгоритмов машинного обучения, функцию *score* необходимо разрабатывать «вручную». Пример слайда №3:

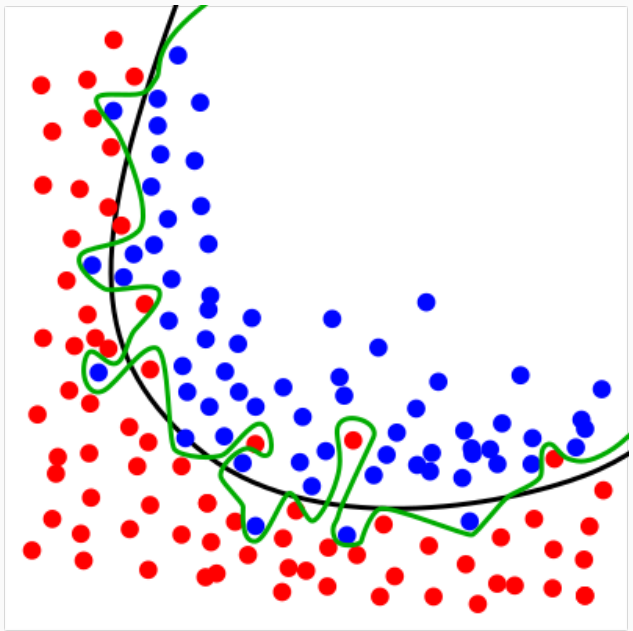
$$
\begin{array}{ll}
0 \mapsto (3234, 25, 1, 1731) & 0 \mapsto (2540, 55, 0, 1731) \\
1 \mapsto (18400, 45, 0, 3137) & 0 \mapsto (2540, 55, 0, 1731) \\
1 \mapsto (903, 19, 0, 4121) & 0 \mapsto (1875, 45, 0, 4121) \\
0 \mapsto (854, 21, 1, 4121) & 1 \mapsto (702, 21, 0, 4121) \\
1 \mapsto (903, 19, 0, 4121) & 0 \mapsto (1875, 45, 0, 4121) \\
0 \mapsto (28400, 41, 1, 3137) & 0 \mapsto (25040, 55, 0, 1731)
\end{array}
$$

Можно задать функцию *score* следующим способом:

$$
score := \begin{cases} 0, \text{ если } x_3 = 1 \vee x_4 = 1731 \\ 0.6667, \text{ если } x_3 = 0 \wedge x_4 \neq 1731 \end{cases} \tag{7}
$$

Переобучение (Переподгонка, Overfitting)

Переобучение – эффект построенной модели (функция *score*), при котором на тестовой выборке модель работает существенно хуже.



Это связано с тем, что при построении модели («в процессе обучения») в обучающей выборке обнаруживаются некоторые случайные закономерности, которые отсутствуют в генеральной совокупности.

Пример переобучения

В примере (7) у нас никогда не бывает мошенничества если клиент является мужчиной ($x_3 = 1$):

$$score := \begin{cases} 0, \text{ если } x_3 = 1 \vee x_4 = 1731 \\ 0.6667, \text{ если } x_3 = 0 \wedge x_4 \neq 1731 \end{cases}$$

Однако это абсурд! Просто у нас такая выборка:

$$\begin{array}{ll} 0 \mapsto (3234, 25, 1, 1731) & 0 \mapsto (2540, 55, 0, 1731) \\ 1 \mapsto (18400, 45, 0, 3137) & 0 \mapsto (2540, 55, 0, 1731) \\ 1 \mapsto (903, 19, 0, 4121) & 0 \mapsto (1875, 45, 0, 4121) \\ 0 \mapsto (854, 21, 1, 4121) & 1 \mapsto (702, 21, 0, 4121) \\ 1 \mapsto (903, 19, 0, 4121) & 0 \mapsto (1875, 45, 0, 4121) \\ 0 \mapsto (28400, 41, 1, 3137) & 0 \mapsto (25040, 55, 0, 1731) \end{array}$$

Обучение с подкреплением, reinforcement learning (Дообучение, Refitting)

Есть первоначальная обучающая выборка:

$$U_{fit} = \{y \mapsto \mathbf{x}\}$$

Первоначальное обучение:

$$score_1 := fit(U_{fit})$$

В дальнейшем при получении нового множества $\hat{U}_{fit}$ (возможно состоящее из одного элемента) мы дообучаем систему:

$$score_{i+1} := refit(\hat{U}_{fit}, score_i) \tag{8}$$

Таким образом функция $score_i$ заменяется на новую функцию $score_{i+1}$.

Тестовая выборка. Отсечка. tp, tn, fp, fn. Расчёт полноты и точность.

Итак, у нас построена *score* каким-либо алгоритмом машинного обучения. Пока оставим за скобками как этот алгоритм построен. Это «чёрный ящик».

Тестовая выборка

После того как мы задали функцию оценщика (*score*) необходимо как-то оценить качество этой функции.

Для этого мы выбираем другую выборку U_{test} и для каждого $\mathbf{x} \in U_{test}$ находим $\hat{y}$.

Таким образом можно задать совокупность пар:

$$\{(y, \hat{y})\} := \{(y, \hat{y}) \ \ \forall (\mathbf{x}, y) \in U_{test} : \hat{y} := score(\mathbf{x})\} \tag{9}$$

После того, как мы задали какую либо отсечку (*offset*), мы каждую пару $(y, \hat{y})$ можем отнести к одному из четырёх событий: **true positive (tp), true negative (tn), false positive (fp), false negative (fn)**.

Вспоминаем первую лекцию. Что такое полнота и точность?

Просмотрим для определённой отсечки *offset* множество $y, \hat{y}$. Если $\hat{y} \geqslant offset$ – то это **positive**, если $\hat{y} < offset$ – то это **negative**.

Если $y = 0$ и **negative** или $y = 1$ и **positive**, то это **true** negative/positive. Иначе **false** negative/positive.

Определим величины:

1. *ctp* (count true positive) – количество событий true positive (tp);
2. *cfp* (count false positive) – количество событий false positive (fp);
3. *ctn* (count true negative) – количество событий true positive (tn);
4. *cfn* (count false negative) – количество событий false negative (fn).

Вопрос на засыпку

Верно ли, что полноту (П) и точность (T) можно определить с помощью формул?:

$$П = \frac{ctp}{ctp + cfn}$$

$$T = \frac{ctp}{ctp + cfp}$$

Мнение зала?

Вопрос на засыпку

Верно ли, что полноту (П) и точность (T) можно определить с помощью формул?:

$$П = \frac{ctp}{ctp + cfn}$$

$$T = \frac{ctp}{ctp + cfp}$$

Запомните

Формула для полноты – верна, а для точности – нет!

Почему?

Вопрос на засыпку

Тестовая выборка U_{test} является лишь подмножеством всех событий.

Как правило мошенничество составляет пренебрежимо малую часть всех транзакций в системе, поэтому за определённый промежуток времени для U_{test} выбирают все события мошенничества.

Однако взять и проскорить все легитимные операции физически невозможно. (Напрмер в Сбербанк Онлайне **ежесекундно** совершается в среднем 80 транзакций). Поэтому для U_{test} выбирают лишь малую долю легитимных транзакций для проверки функции *score*.

Полнота и точность

Введём величину $q_{legitim}$ – это отношение количества легитимных транзакций во всей выборке, делённая на количество легитимных транзакций в U_{test}.

Для любой отсечки *offset* можем вычислить ***полноту*** (П, recall) и ***точность*** (T, precision):

$$\text{П} = \frac{ctp}{ctp + cfn} \tag{10}$$

$$T = \frac{ctp}{ctp + cfp \cdot q_{legitim}} \tag{11}$$

Дерево решений. Лес решений

Замечание

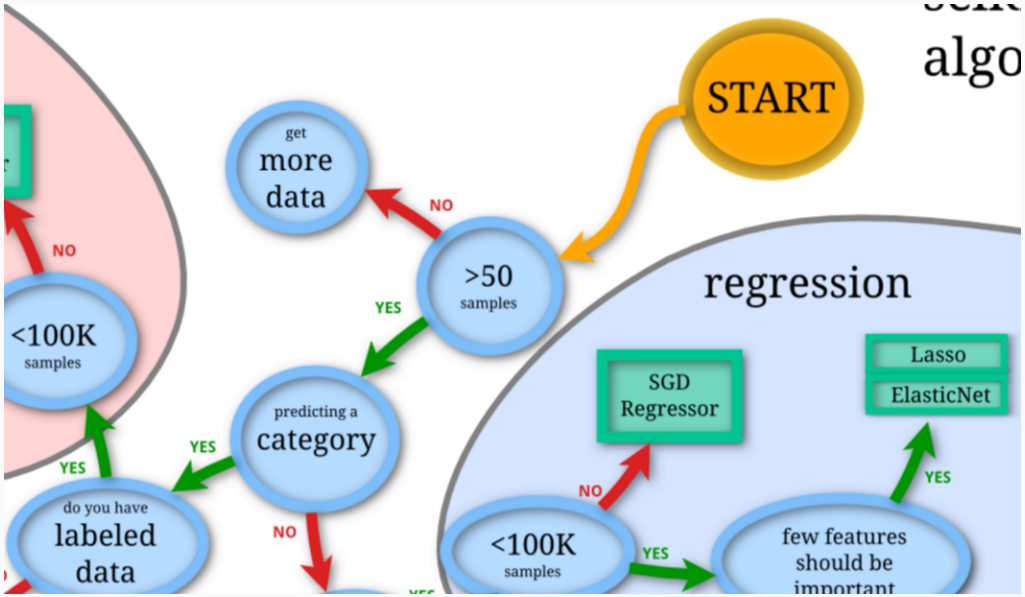
Алгоритмов машинного обучения очень много. Хороший Data Scientist должен взять себе в привычку время от времени изучать те или иные алгоритмы.

Цель курса – прикладная. Научить решать ИБ задачи с помощью Data Science методов.

Тем не менее хотя бы один алгоритм должен быть изучен.

Дерево решений

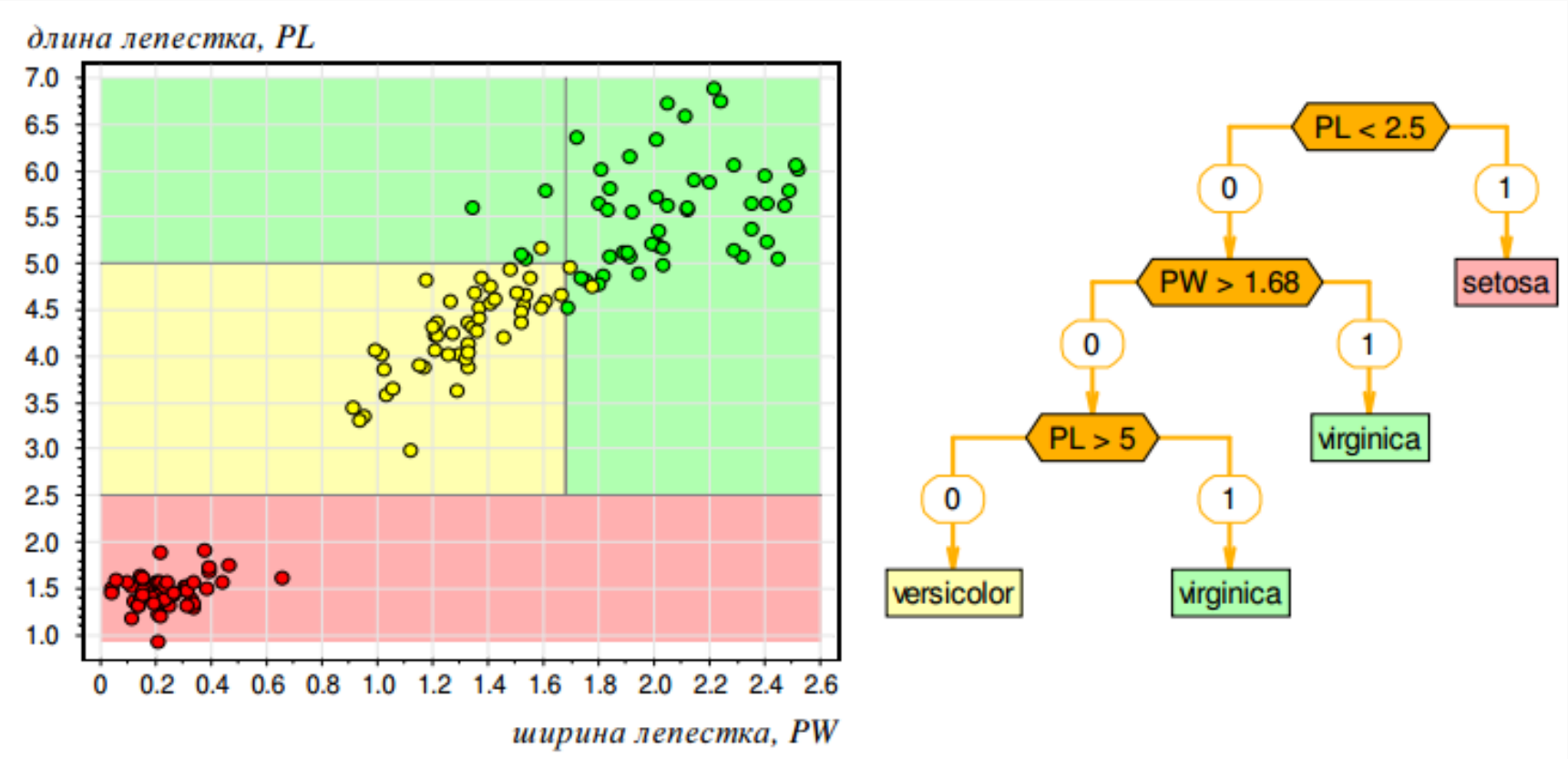
Отрывок из «Choosing the right estimator»:[3]



На каждом шаге мы отвечаем на вопрос и ”идем по дереву дальше”.

[3] https://scikit-learn.org/stable/tutorial/machine_learning_map/index.html

Пример дерева решений: «Ирисы Фишера»



Пример *экспертной системы*, являющийся решающим деревом.

Пример дерева решений: «Ирисы Фишера»

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	$r_4(x) = [PL > 2.5] \wedge [PL \leqslant 5] \wedge [PW < 1.68]$

Любое дерево решений можно представить в виде совокупности булевых выражений.

Формальное определение

Дерево – это связанный | ациклический | граф.

Дерево решений – это дерево, в терминальных вершинах которых определён *отклик*, в остальных листьях – функции[4] каждый выход из которых определяет метку какого-либо выходящего ребра.

[4] на практике – *булевые* функции, но в общем определении – не обязательно.

Лес решений.

Лес решений – это *ансамбль*, каждый классификатор которого является *деревом решений*.

Обычно лес решений – это среднее арифметическое всех его деревьев.

Например решаем задачу обнаружения мошенничества. Всего обучено 500 деревьев решений. На какой-либо транзакции 423 дерева определили что эта транзакция мошенническая, 42 – легитимная, остальные 35 деревьев – отказ от классификации.

Каков отклик данного леса?

Лес решений.

Лес решений – это *ансамбль*, каждый классификатор которого является *деревом решений*.

Обычно лес решений – это среднее арифметическое всех его деревьев.

Например решаем задачу обнаружения мошенничества. Всего обучено 500 деревьев решений. На какой-либо транзакции 423 дерева определили что эта транзакция мошенническая, 42 – легитимная, остальные 35 деревьев – отказ от классификации.

Тогда лес вернёт решение, что с *априорной вероятностью* $p = \frac{423}{423+42} = 0.9097$ данная операция является мошеннической.

Замечание.

Лес решений – это не обязательно среднее арифметическое *откликов*.

$$p = \frac{1}{N} \cdot \sum_{i=1}^{N} \hat{y}_i$$

где $\hat{y} = 1$, если подозрение на мошенничество или $\hat{y} = 0$ иначе.

Можно каждому i-му дереву присвоить вес $w_i \geqslant 0$ и по разному его взвешивать:

$$p = \frac{1}{\sum_{i=1}^{N} w_i} \cdot \sum_{i=1}^{N} \hat{y}_i \cdot w_i$$

Однако на практике в этом нет смысла.

Построение случайного дерева решений.

Замечание

Существует большое количество оптимизаций и улучшений. Данное описание построения показывает СУТЬ работы. Желающие понять тему до конца могут ознакомится с академическими работами по построению современных Random Forest систем.

Постановка задачи

Пока рассмотрим случай, в котором *вектор признаков* состоит всего из двух величин: (x_1, x_2).

Таким образом *обучающая выборка* U_{fit} имеет вид:

$$U_{fit} = \{y \mapsto (x_1, x_2)\} \tag{12}$$

Тогда требуется определить *функцию высшего порядка fit*, которая принимает на вход U_{fit} а на выходе возвращает функцию *score*:

$$score := fit(U_{fit}) \tag{13}$$

Функция *score* – это *дерево решений*.

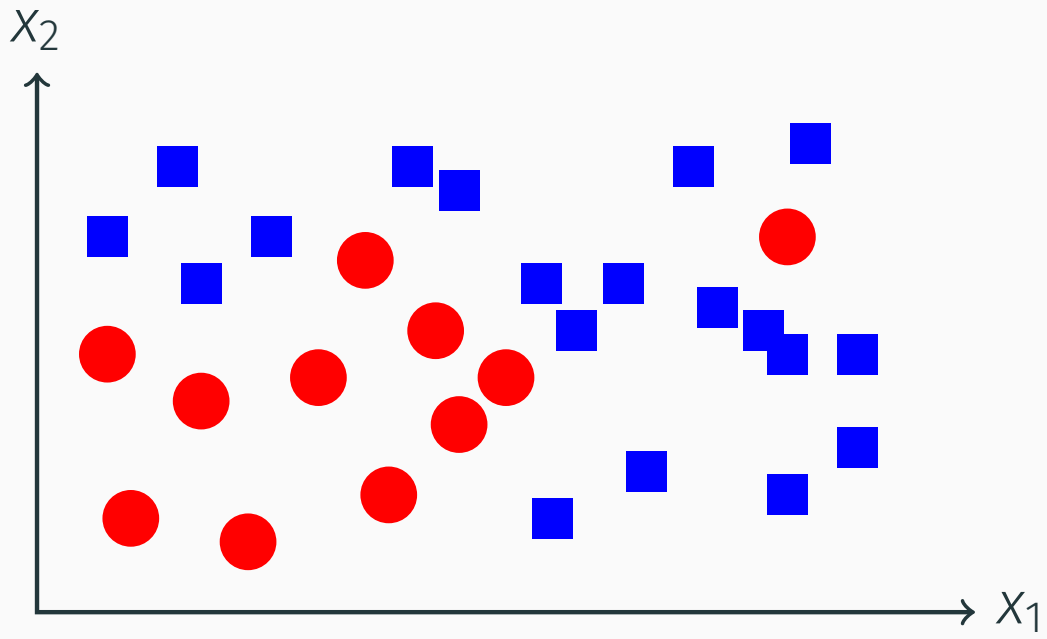
На выходе *score* возвращает $\hat{y} \in \{0, 1\}$:

$$\hat{y} := score(x_1, x_2) \tag{14}$$

Визуализация данных

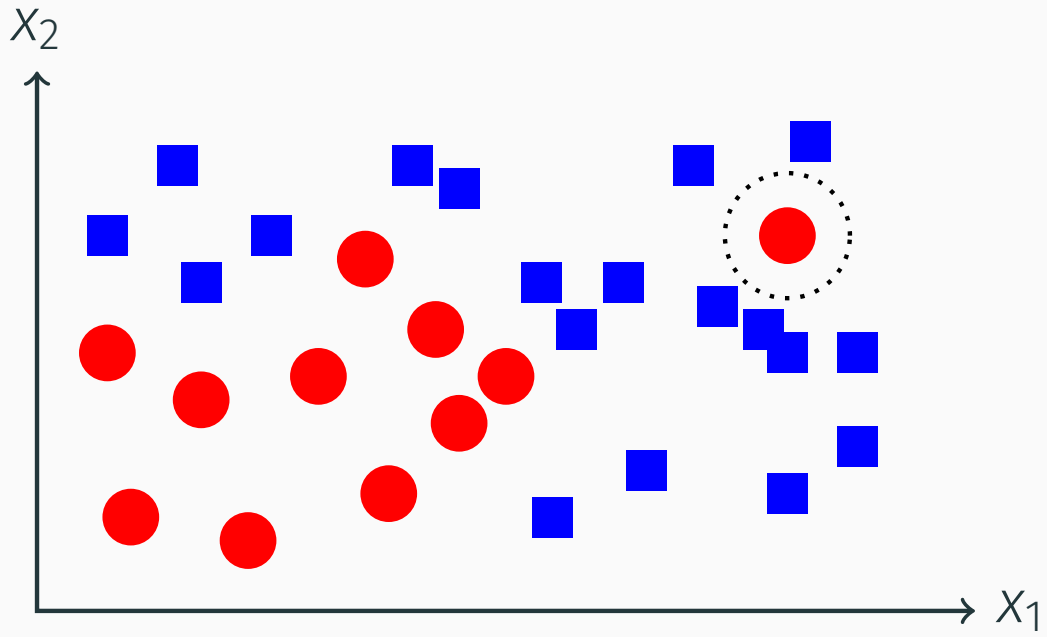
Для удобства обозначим $y = 1$ (мошенничество) красным кругом, $y = 0$ (легитимная операция) – синим квадратом.

Так как у нас вектор признаков $\mathbf{x}$ состоит из двух признаков (x_1, x_2), то каждую пару $y \mapsto \mathbf{x}$ можно визуализировать на плоскости:



Выброс

Выброс – объект обучающей выборки (пара $y \mapsto \mathbf{x}$), «выделяющееся из общей выборки»



Замечание.

Строгого определения термина *выброс*, не существует.

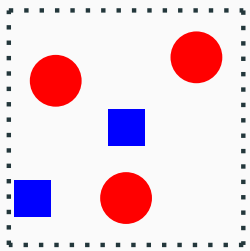
Индекс Джини

Пусть n – это количество классов. (В нашем примере $n = 2$).

Зафиксируем некое *замкнутое подпространство* признаков и назовём его бином[5] (*bin*).

Определим Δ_i – доля объектов класса i относительно всех объектов в данном бине:

$$\Delta_i \overset{def}{=} \frac{\| \{y \mapsto \mathbf{x} : y = y_i \wedge \mathbf{x} \in bin\} \|}{\| \{y \mapsto \mathbf{x} : \mathbf{x} \in bin\} \|} \tag{15}$$



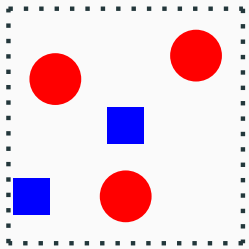
Пример: $\Delta_0 = \frac{2}{5}$; $\Delta_1 = \frac{3}{5}$

[5] от англ **bin [bɪn]** – контейнер, емкость, бак, урна.

Индекс Джини

Индекс (загрязненности) Джини (Gini impurity) некоего бина *bin* – это величина, вычисленная по формуле:

$$\delta(bin) = 1 - \sum_{i=0}^{n-1} \Delta_i^2 \tag{16}$$

Пример: 

$$\delta(bin) = 1 - \left(\frac{2}{5}\right)^2 - \left(\frac{3}{5}\right)^2 = 1 - \frac{13}{25} = \frac{12}{25}$$

Замечание

Не путайте *Индекс Джини* и *Коэфициент Джини*. Это разные понятия.

Альтернативы Индекса Джини

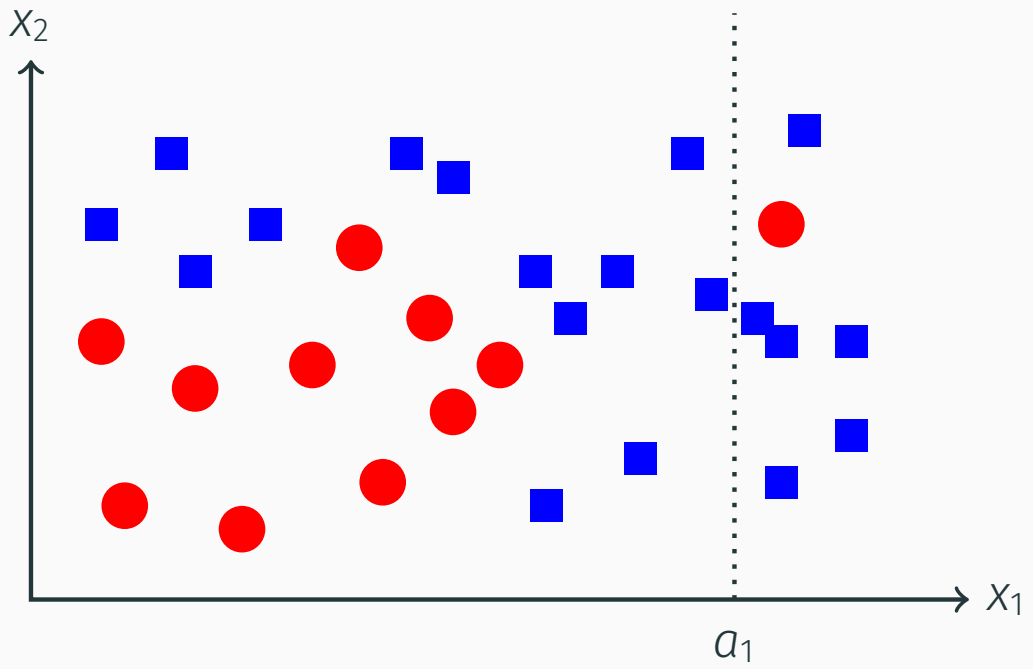
Более «классическим» критерием является ***(информационная) энтропия по Шеннону***:

$$H(bin) = -\sum_{i=0}^{n-1} \Delta_i \cdot log_n \Delta_i \tag{17}$$

Однако на практике её вычислять достаточно долго. Индекс Джини, это «математически кастрированная энтропия», дающая почти те же результаты за более быстрое время.

Шаг 0. Вручную определим критерий остановки: δ_{stop}. На практике $\delta_{stop} = 10^{-3}...10^{-6}$. В нашем примере пусть $\delta_{stop} = 0.25$.

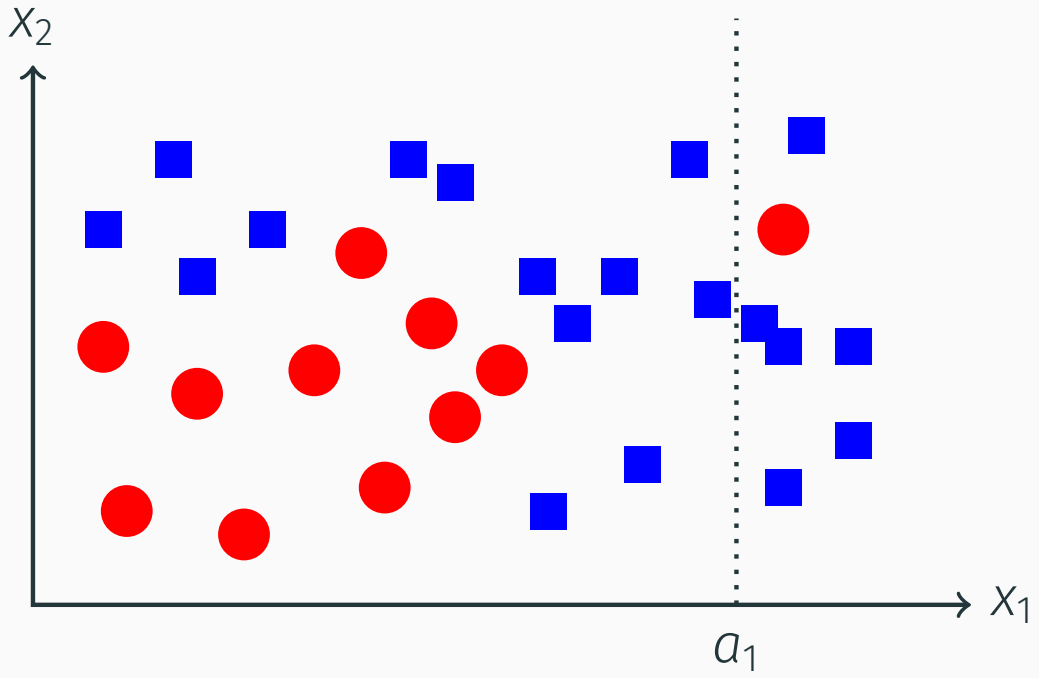
Шаг 1. Определим случайно один из признаков. Допустим это x_1. Определим случайно величину на отрезке x_1. Пусть это некое a_1. Начертим ось, перпендикулярную x_1 и проходящую через $(a_1, 0)$.



Таким образом мы разбили пространство признаков $R^n = R^2$ на два *бина*:

1. $r(x_1 = a_1)$ – бин, для которого $x_1 \geqslant a_1$;
2. $l(x_1 = a_1)$ – бин для которого $x_1 < a_1$.

Посчитаем индекс Джини для каждого бина и сравним его с δ_{stop}:



$$\delta\left(r(x_1 = a_1)\right) = 1 - \left(\frac{6}{7}\right)^2 - \left(\frac{1}{7}\right)^2 = \frac{12}{49} \leqslant 0.25 = \delta_{stop}$$

$$\delta\left(l(x_1 = a_1)\right) = 1 - \left(\frac{13}{23}\right)^2 - \left(\frac{10}{23}\right)^2 = \frac{260}{529} \approx 0,49149 > 0.25 = \delta_{stop}$$

Если индекс джини не больше δ_{stop}, то в рамках данного бина выносим вердикт. Иначе повторяем шаг 1, в рамках выбранного бина.

Пересечение бинов

Определим операцию $bin_1 \cap bin_2$ – означающую пересечение бинов.

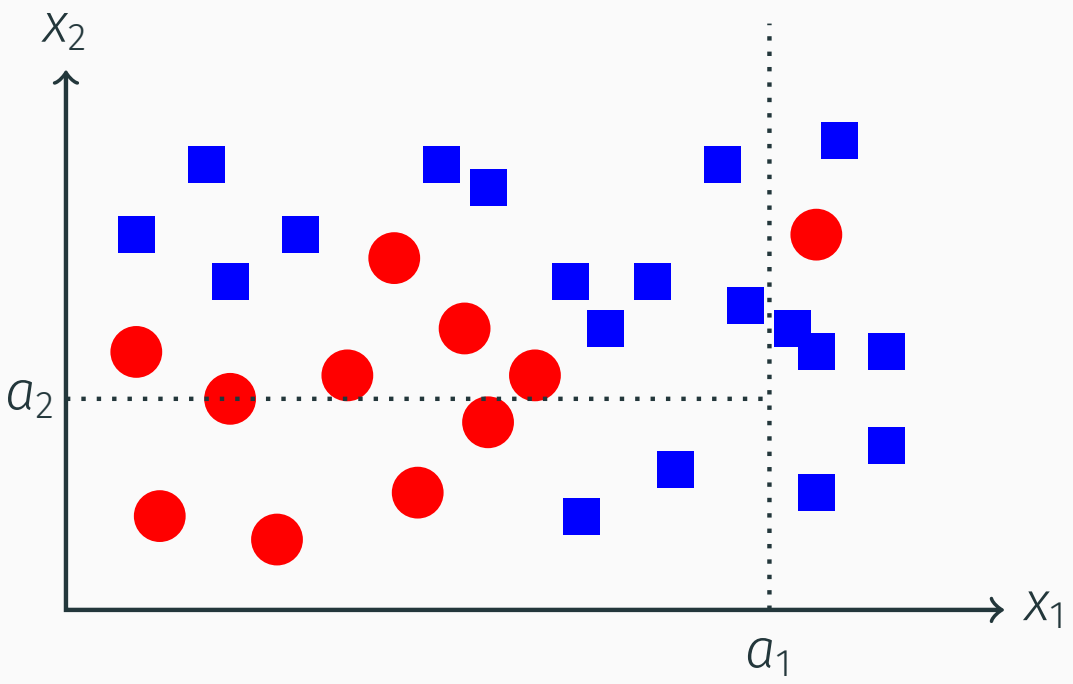
Например:

$$r(x_1 = a_1) \cap r(x_2 = a_2) \cap l(x_1 = a_3)$$

Означает:

$$\begin{cases} x_1 \geqslant a_1 \\ x_2 \geqslant a_2 \\ x_1 < a_3 \end{cases}$$

Шаг 2. Допустим что теперь случайным образом выбрали x_2. Определили a_2.



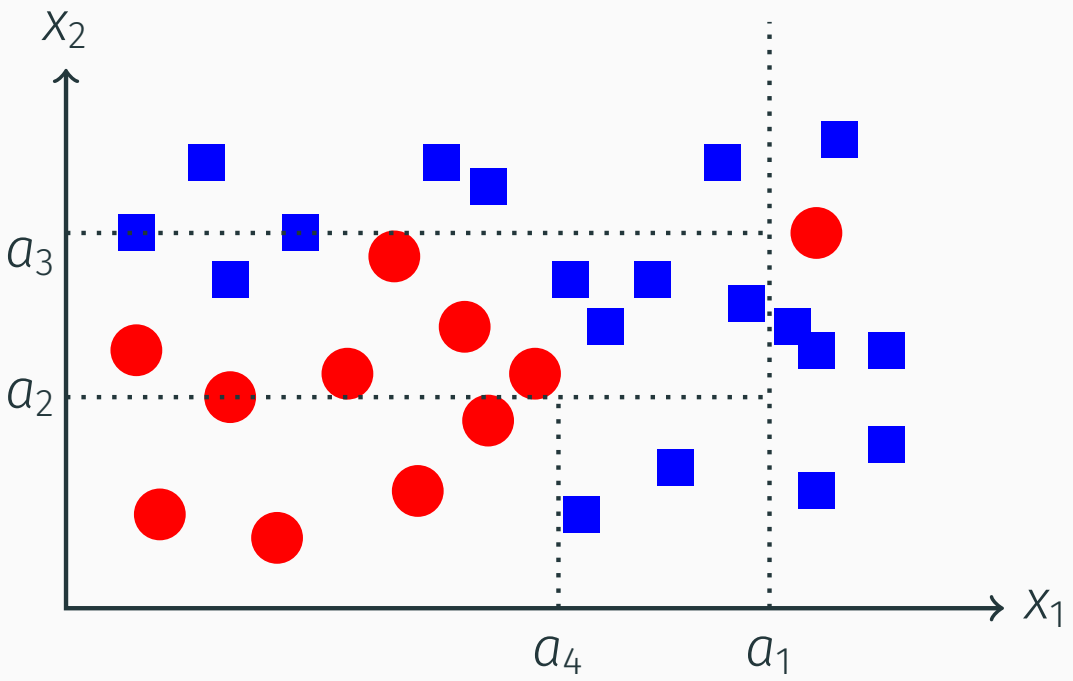
$$\delta\big(l(x_1 = a_1) \cap r(x_2 = a_2)\big) = 1 - \left(\frac{11}{17}\right)^2 - \left(\frac{6}{17}\right)^2 = \frac{132}{289} \approx 0,4567 > 0.25$$

$$\delta\big(l(x_1 = a_1) \cap l(x_2 = a_2)\big) = 1 - \left(\frac{2}{6}\right)^2 - \left(\frac{4}{6}\right)^2 = \frac{16}{36} \approx 0.4444 > 0.25$$

Вывод: в обоих винах строим дерево решений дальше. Нет остановки.

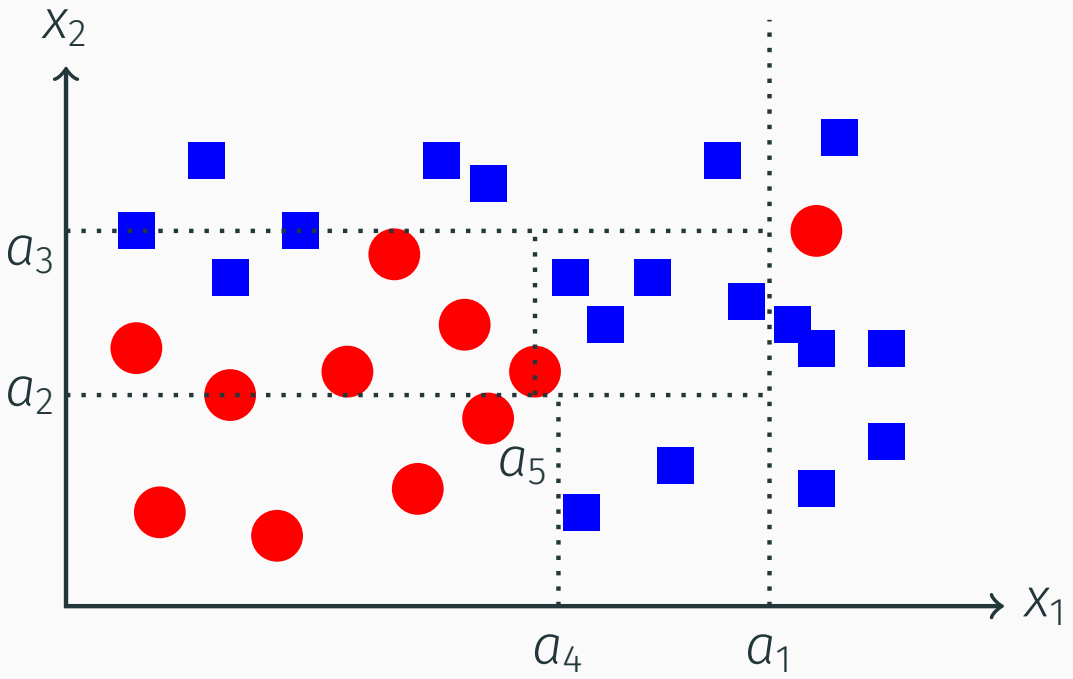
Шаг 3. В бине $\big(l(x_1 = a_1) \cap r(x_2 = a_2)\big)$ случайно выбрали x_2 и a_3.

В бине $\big(l(x_1 = a_1) \cap l(x_2 = a_2)\big)$ случайно выбрали x_1 и a_4.



Очень удачный шаг. Остался бин $l(x_1 = a_1) \cap r(x_2 = a_2) \cap l(x_2 = a_3)$.

Шаг 4.



Конец алгоритма. Индекс Джини всех бинов не больше чем δ_{stop}.

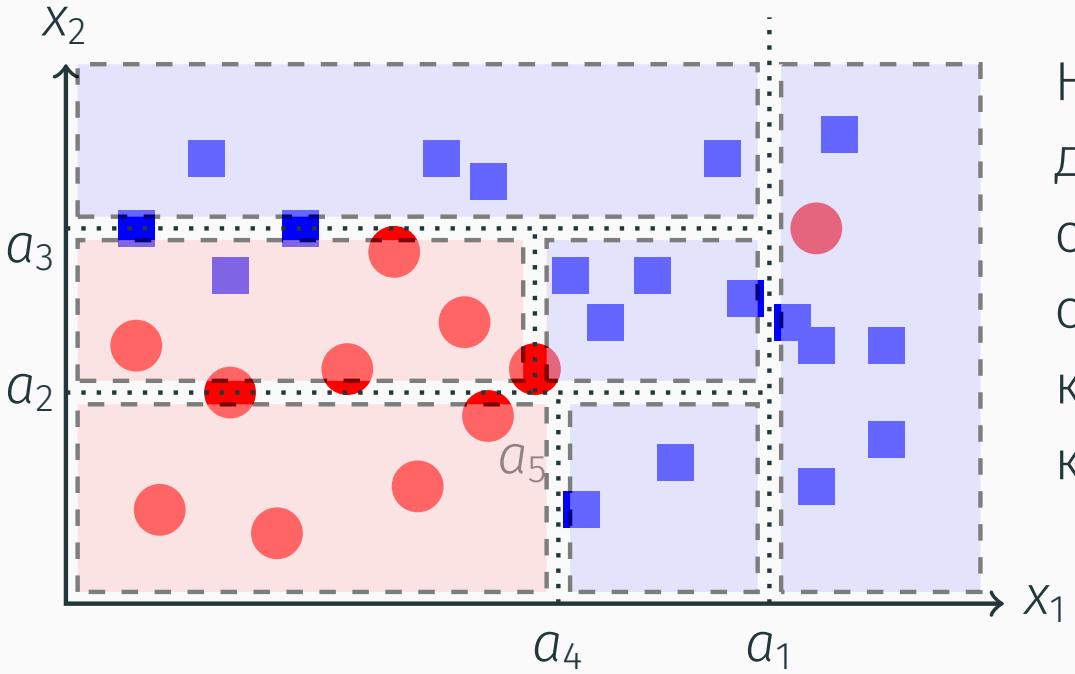
Замечание

Данное дерево было построено случайным образом. При повторном запуске алгоритма может получится другое дерево.

Случайное дерево решений.

Случайное дерево решений – это *бинарный* классификатор, возвращающий априорную вероятность:

- $p = 0$ – легитимная операция
- $p = 1$ – мошенническая операция



На рисунке бины, для которых $p = 0$ отмечены голубыми областями, а бины, для которых $p = 1$ отмечены красными областями.

Точки остановки

Помимо *индекса Джини* (параметр *min_impurity_split* в scikit-learn) можно использовать другие точки остановки:

- Количество данных (векторов признаков) в бине меньше *min_samples_split*;
- Глубина дерева достигла *max_depth*.

ДЗ. Посмотреть: `https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier`

Случайный Лес (Random Forest)

Изобретатели Случайного Леса (Random Forest):

- Павлов Юрий Леонидович (р.1949)
- Лео Брейнман (Leo Breiman) (1928 — 2005)

Случайный лес – это *ансамбль* случайных деревьев (на практике от 100 до 1000).

В общем случае случайным образом выбираются:

1. подмножество признаков, на котором строим дерево
2. подмножество обучающей выборки
3. выбор признака и значения разбиения.

Обычно берут функцию голосования. Всего N деревьев, n_1 – количество деревьев вернувших 1, n_0 – количество деревьев, вернувших 0, при этом $n_0 + n_1 \leqslant N$. (Если нет отказа от классификации, то $n_0 + n_1 = N$).

Тогда *априорная вероятность* мошенничества вычисляется по формуле:

$$p = \frac{n_1}{n_1 + n_0} \tag{18}$$

Замечание.

Вместо функции голосования, можно брать любое другое **среднее по Колмогорову** или другую функцию.

«Центральная эмпирическая «теорема» о Случайном Лесе»

«Ц.Э.Т»

«Если вы работаете в области, в которой совершенно некомпетентны и/или решаете задачу, про которую ничего не знаете – используйте случайный лес. Этот алгоритм М.О. – самый лучший!»

Это совершенно «не академично», но истинно.

Более того, т.к. в задачах ИБ, ваш противник – человек, то следует все усилия приложить на Feature Extraction, а в качестве алгоритма М.О. использовать Random Forest.

Выводы «Ц.Э.Т.»

Выводы:

1. Случайный лес – решение многих задач на вполне приемлемом уровне
2. Сравните свой классификатор со случайным лесом и вы поймете насколько ваш классификатор хорош
3. По настоящему интересные задачи – это те, которые не решаются случайным лесом.

Достоинства RF

1. очень просто устроен
2. устойчивость к переобучению.
3. понятные и легко настраиваемые параметры
4. либо RF работает, либо данные «сырые» и требуют предобработки (например выбор паттернов в изображении нейронной сетью)
5. уже реализован. Легко программируем с нуля (для прошивок)
6. быстрый. (Условия дерева решений if-then-else просты и быстрые.)
7. $\sim$масштабируемый.
8. RF является мощным средством против «активного противника»[6].
9. не требует определения расстояния Махаланобиса[7]

[6] в задачах с обратной связью изучаемого субъекта (хакера, мошенника)
[7] на след.лекциях будет

Список материалов

Случайные леса

Сергей Павлович Чистяков **«Случайные леса: обзор»**[8]

sklearn:

- sklearn.ensemble.RandomForestClassifier[9]
- sklearn.ensemble.RandomForestRegressor[10]
- sklearn.ensemble.IsolationForest[11]

---

[8] http://resources.krc.karelia.ru/transactions/doc/trudy2013/trudy_2013_1_117-136.pdf

[9] https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html

[10] https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestRegressor.html

[11] https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.IsolationForest.html

Вопросы для самопроверки

Признаки, вектор признаков, выборка

1. Что такое *признак*?
2. Что такое несравнимый (категориальный) признак? Чем он отличается от сравнимого?
3. Что такое булевый признак? Числовой?
4. Приведите пример булевого и несравнимого признака. Болевого и сравнимого.
5. Что такое *вектор признаков*? Есть два вектора $\mathbf{x}_a = (x_1, x_2)$ и $\mathbf{x}_b = (x_2, x_1)$. Являются ли они идентичными? Можете привести задачу, когда являются?
6. Что такое *обучающая выборка*? Почему формула (1) задаётся как **неупорядоченная** совокупность, а не как список (т.е. упорядоченная совокупность). Почему порядок перечесления данных в аргументе функции (3) не важен?

Обучающая выборка со слайда №3

Посмотрите на *выборку* со слайда №3.

1. Верно ли утверждение, что если совершается оплата электроэнергии, то данная операция всегда легитимная?
2. Есть ли корреляция между возрастом клиента и мошенничеством при покупке авиабилетов?
3. Можно ли предположить, что таксисты чаще обманывают молодых девушек?
4. Покупает ли молодёжь авиабилеты через данный банк?
5. Если совершают мошенничество в сфере автоперевозок, то обманывают мужчин или женщин?

Обучение и переобучение

1. Что такое обучение (fitting)? Что такое переобучение? (overfitting)?
2. Почему функции *fit* и *refit* – это функции высшего порядка?
3. Как с помощью ансамблирования уменьшить проблему переобучения (а для некоторых алгоритмов и вовсе его убрать) ?
4. можете предложить какую-либо меру измерить переобучение?
5. Верно ли утверждение, что чем больше и «разнообразней» выборка, тем меньше ошибок, вызванных переобучением?

Полнота, точность и q_{fraud}

Существуют задачи, в которых физически невозможно рассчитать скоринг для всех мошеннических операций на определённом достаточном для анализа качества системы промежутке времени (например неделя). К примеру – это задача распознавания спама. Спама очень много...

Тогда, для расчёта $\{y, \hat{y}\}$ выбирается лишь подмножество мошеннических операций для выборки U_{test}.

Таким образом, аналогично $q_{legitim}$ разумно определить коэффициент q_{fraud}.

Как тогда изменятся формулы (10) и (11)?

В банковской системе «ВашБанк Онлайн» введена система фрод-мониторинга. Она представляет собой лес решений из 700 деревьев решений. Функция принятия решений – среднее арифметическое. На определённой транзакции 573 деревьев определило транзакцию как мошенническую, 57 как легитимную, остальные – отказ от классификации.

1. каков отклик системы ФМ?
2. каков отклик системы ФМ, если отказ от классификации считать легитимными операциями?
3. каков отклик системы ФМ, если отказ от классификации считать подозрением на мошенничество?
4. каков отклик системы ФМ, если отказ от классификации считать подозрением на мошенничество, однако вклад брать не как 1, а как 0.7 ?

Индекс Джини и бины

1. Почему на практике используют индекс Джини, а не энтропию по Шеннону? Можете придумать ещё более простую «меру загрязнённости», чем индекс Джини?
2. При каких условиях индекс Джини равен нулю? При каких 1/2 ? При каких равен 1? При каких условиях не определён?
3. Докажите что для любого количества классов n, индекс Джини лежит в границах: $\delta \in [0, \frac{1}{2}]$.
4. Почему бины обязательно квадратные, а не круглые или треугольные? Какая разница в каком подпространстве признаков рассчитывать индекс Джини?

Случайное дерево. Случайный лес.

1. С помощью кванторов и слов напишите строгий академический алгоритм, описывающий построение случайного дерева над пространством признаков R^n.
2. Аналогично п.1, напишите строгий академический алгоритм, описывающий построение случайного леса
3. Объясните, почему RF устойчив к переобучению.
4. Объясните, почему RF очень легко реализуем на ПЛИС.
5. Пусть есть некий признако x_i принимающий одно из значений: «русский», «француз», «чукча», «арапеш». Как задать функции $r(x_i = a)$ и $l(x_i = a)$?
6. В RF количество деревьев N – параметр. Придумайте алгоритм автоматического[12] нахождения оптимального N для любой обучающей выборки.

[12] =без участния эксперта