



# Машинное обучение в науках о Земле

Михаил Криницкий

К.Т.Н., С.Н.С.

Институт океанологии РАН им. П.П. Ширшова

Лаборатория взаимодействия океана и атмосферы и  
мониторинга климатических изменений (ЛВОАМКИ)



# Непараметрические методы машинного обучения

Михаил Криницкий

К.Т.Н., С.Н.С.

Институт океанологии РАН им. П.П. Ширшова

Лаборатория взаимодействия океана и атмосферы и  
мониторинга климатических изменений (ЛВОАМКИ)

# Деревья решений



ОБУЧЕНИЕ ДЕРЕВЬЕВ ©

# ЗАДАЧА КЛАССИФИКАЦИИ

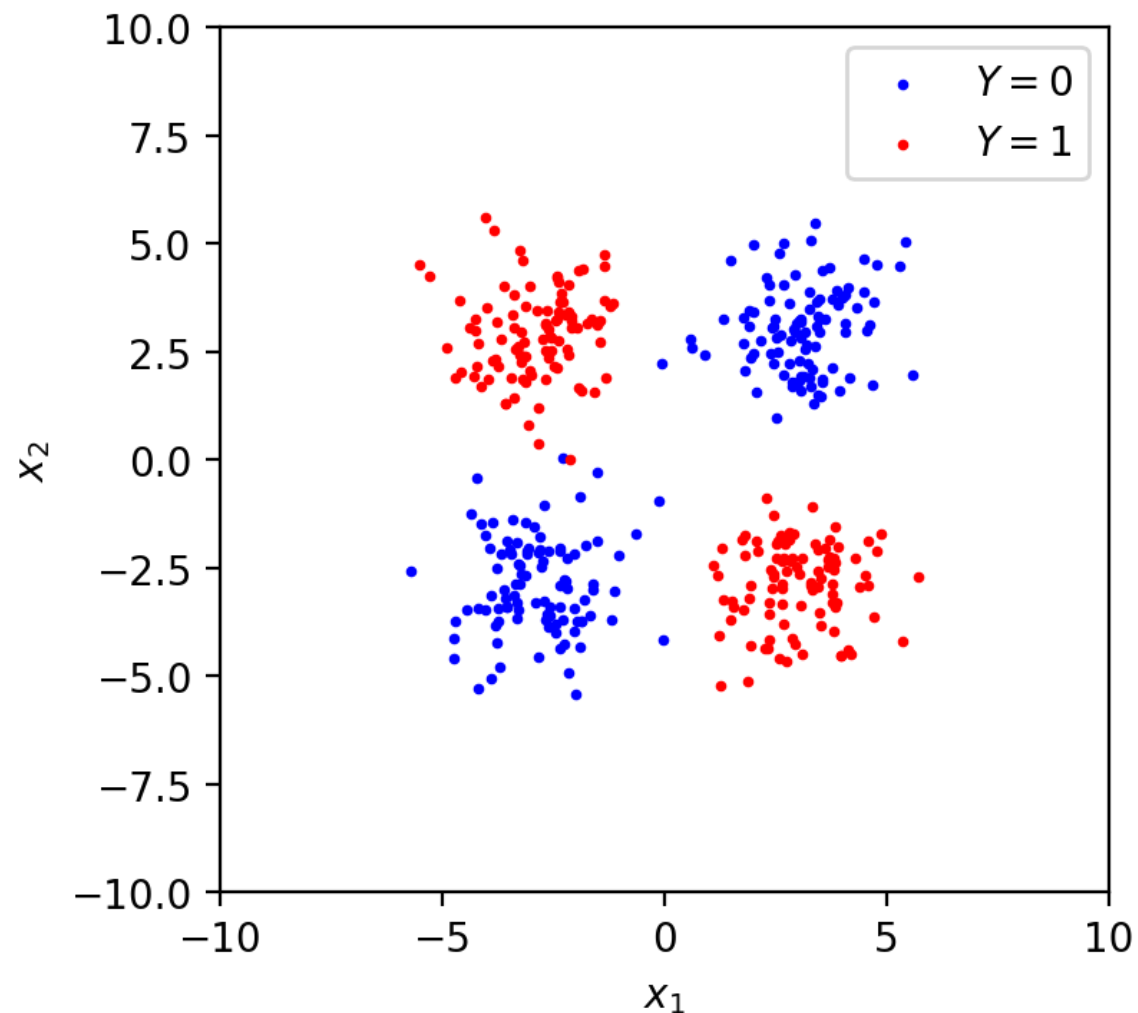
В качестве упрощенного примера рассмотрим задачу бинарной классификации, в которой объекты описываются двумерными векторами:

$$X \in \mathbb{R}^2$$
$$Y \in \{0, 1\}$$

Легко обнаружить, что линейные модели не смогут хорошо решить представленную задачу.

Варианты решения в рамках известных моделей:

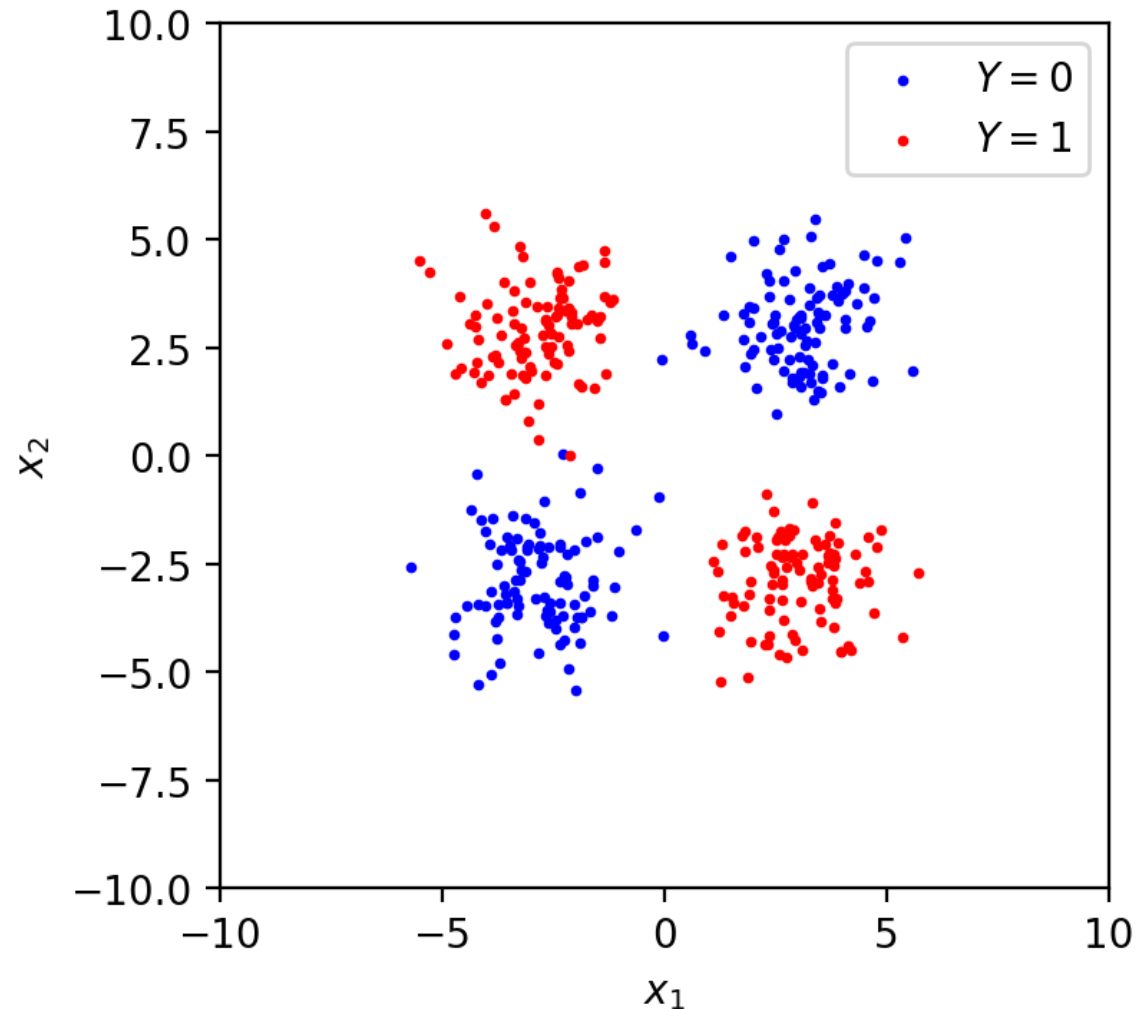
- GLM, GAM при условии расширения признакового пространства («вручную», согласно соображениям исследователя);
- SVM с применением kernel trick, виртуально расширяющим признаковое пространство;
- Искусственные нейронные сети



# ЗАДАЧА КЛАССИФИКАЦИИ

Варианты решения в рамках известных моделей:

- GLM, GAM при условии расширения признакового пространства («вручную», согласно соображениям исследователя);
  - В случае сложных данных с признаковым описанием высокой размерности решения исследователя могут быть субоптимальными;
  - GLM и GAM – модели со слабой выразительной способностью; в случае данных с повышенной сложностью этой выразительной способности может не хватать даже на новом пространстве признаков;
  - порождение «подходящих» признаков может оказаться непосильной задачей для исследователя;
- SVM с применением kernel trick, виртуально расширяющим признаковое пространство;
  - подбор ядра для хорошего решения задачи – искусство, задача может быть решена субоптимально при условии неверно выбранного ядра;
  - на больших выборках данных обучение SVM с ядром может занимать существенное время (см. лекцию 17); SVM может демонстрировать субоптимальное решение на шумных данных с пересекающимися классами, особенно при неверном подборе гиперпараметров метода;
- Искусственные нейронные сети
  - может быть слишком выразительным решением; оптимизация соотношения выразительной способности и сложности данных – искусство, на настоящий момент не автоматизируется;
  - существенные вычислительные затраты

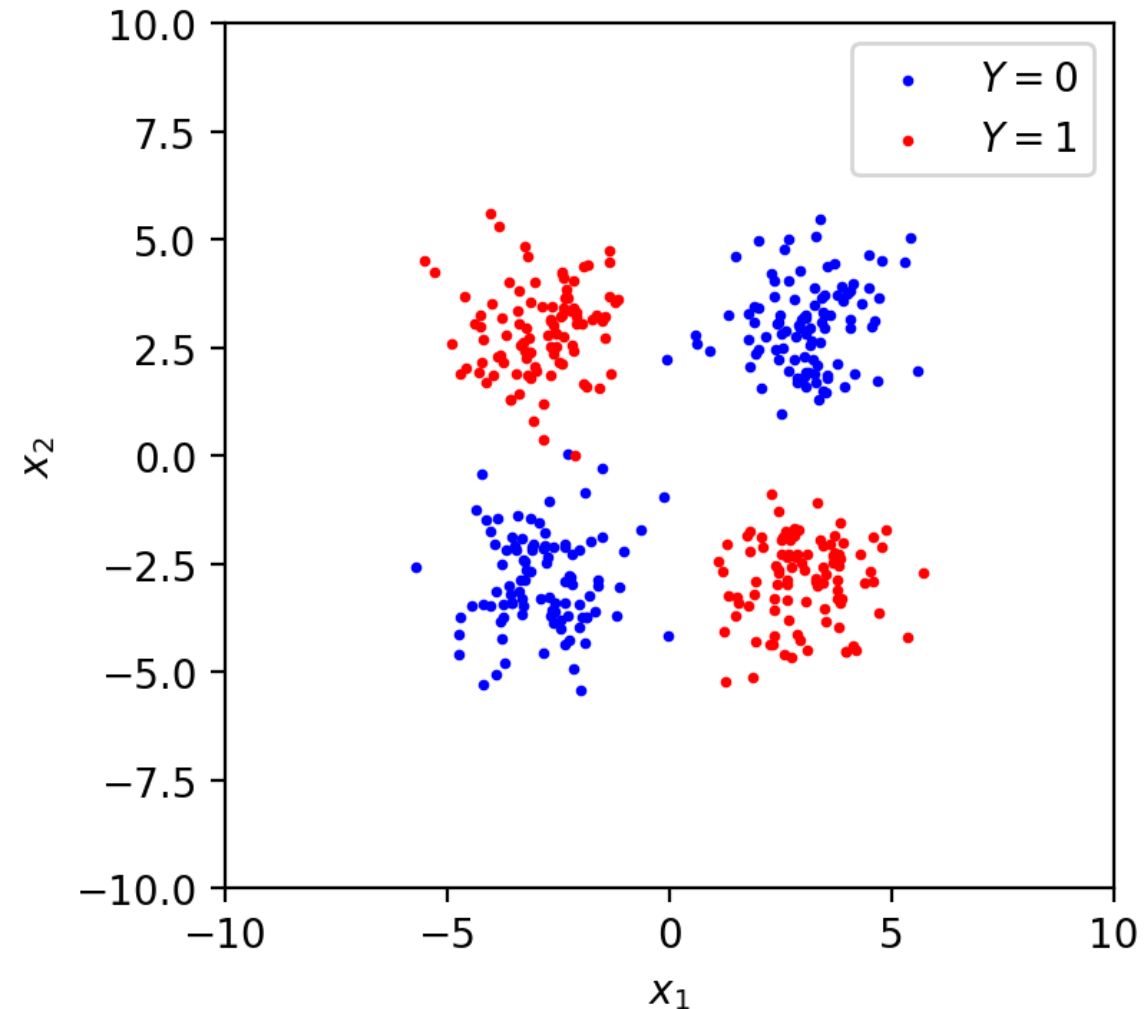


# Альтернатива - деревья решений (Decision Trees, DT)

Идея:

- строить разделяющую поверхность максимально просто, понятно и интерпретируемо;
- имитировать процесс принятия решений в сценарии «вопрос-решение».
- Пусть эта поверхность будет линейна. Даже пусть она будет кусочно-постоянна

Кусочно-постоянная разделяющая поверхность состоит из участков гиперплоскостей, каждая из которых разделяет пространство примеров только по одному из признаков.



# DT в режиме исполнения

Формализация в рекурсивной записи:

- пусть  $R_p^l$  ("parent" sample) - выборка тестовых примеров  $\{x_i, y_i\}_p^l$  на очередном  $l$ -том этапе ветвления;
- по определенному условию  $[x^{(j)} \geq t^{(l)}]$  на  $j_l$ -й признак производится деление выборки на  $R_{c1}$  и  $R_{c2}$  ("child" samples):

$$R_{c1}^l = \{X | x^{(j_l)} < t^{(l)}, x \in R_p^l\}$$

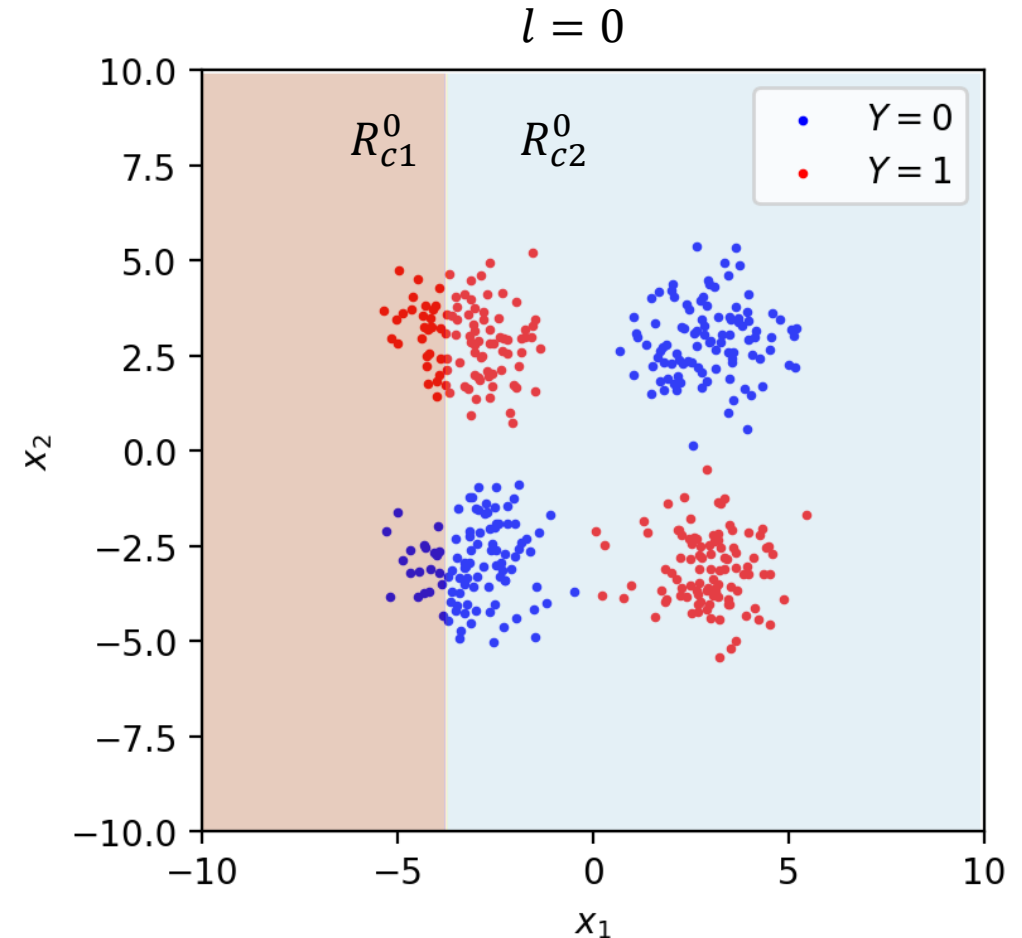
$$R_{c2}^l = \{X | x^{(j_l)} \geq t^{(l)}, x \in R_p^l\}$$

NOTE: деление применяется только к тому подмножеству примеров, которое содержится в «родительской» выборке  $R_p$ . Из этого следует, например, что каждый пример окажется в одной из выборок  $R_{c1}$  или  $R_{c2}$  и только одной из них.

- если на этом этапе деление останавливается, множества примеров в  $R_{c1}^l$  и в  $R_{c2}^l$  называют «**листьями**» ("leafs")

NOTE: в экстремальном случае построения на этапе обучения дерева «до конца» в каждом листе содержится только один элемент.

- после  $l$ -го деления каждая из выборок  $R_{c1}^l, R_{c2}^l$  становится «родительской» для очередного ветвления; процедура разделения повторяется для вновь образованных  $R_p^{l+1}$ .
- классификация: элементам в «листьях» присваивается (взвешенный) majority-класс (определяется на этапе обучения как класс, имеющий численное преимущество в этом листе)
- регрессия: элементам в «листьях» присваивается (взвешенное) среднее значение, определяемое на этапе обучения



На рисунке: деление выборки всех примеров  $R_p^0$  на  $R_{c1}^0, R_{c2}^0$ . Внимание: на рисунке – тестовая выборка, правила деления (номер признака  $j_l$  и пороговое значение  $t^{(l)}$ ) были определены во время обучения.

# DT в режиме исполнения

Вычисление целевой переменной  $\hat{y}$  для объектов в листе

- (!!!) Все объекты листа будут иметь одно и то же значение целевой переменной

Классификация:

$$\hat{c}_R = \operatorname{argmax}_{c \in \mathbb{Y}} p_c^{(R)}$$

$p_c^{(R)}$  - доля **обучающих** примеров класса  $c$  в листе  $R$   
(может вычисляться с учетом весов примеров  $\{w_i\}$ )

$p_c^{(R)}$  вычисляется на этапе обучения

Регрессия:

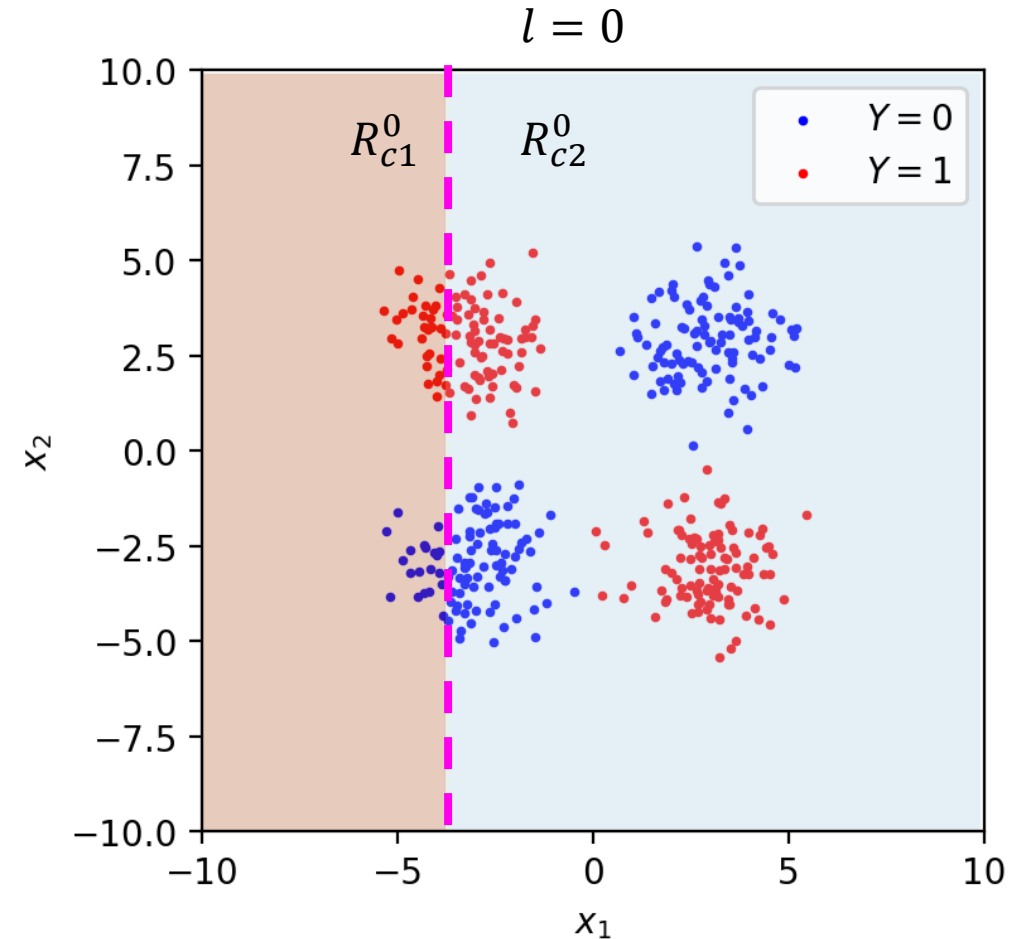
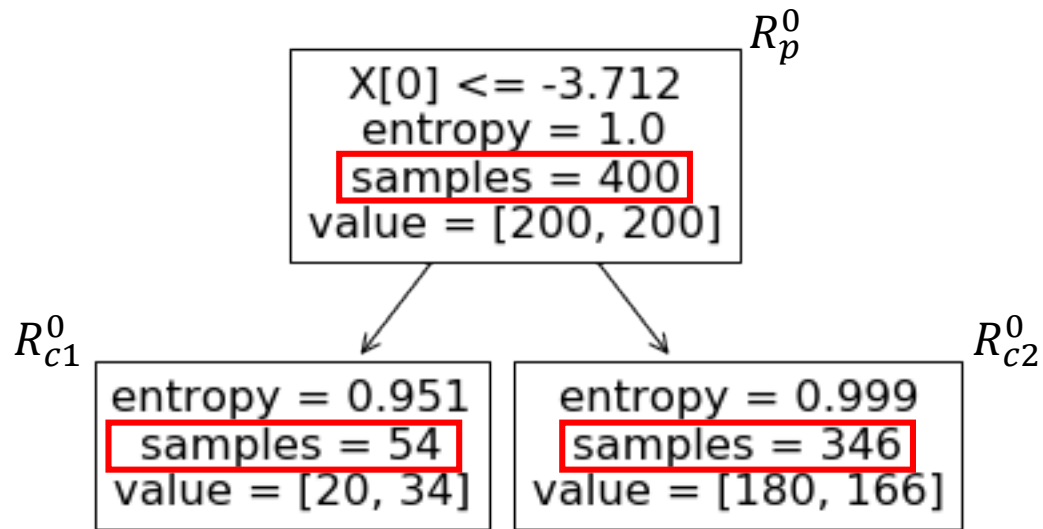
$$\hat{y}(R) = \frac{1}{\sum_{i=1}^{|R|} w_i} * \sum_{i=1}^{|R|} w_i * y_i^{(train,R)}$$

$$p_c^{(R)} = \frac{1}{\sum_{i=1}^{|R|} w_i} * \sum_{i=1}^{|R|} w_i * [y_i^{(train,R)} == c]$$



# DT в режиме исполнения

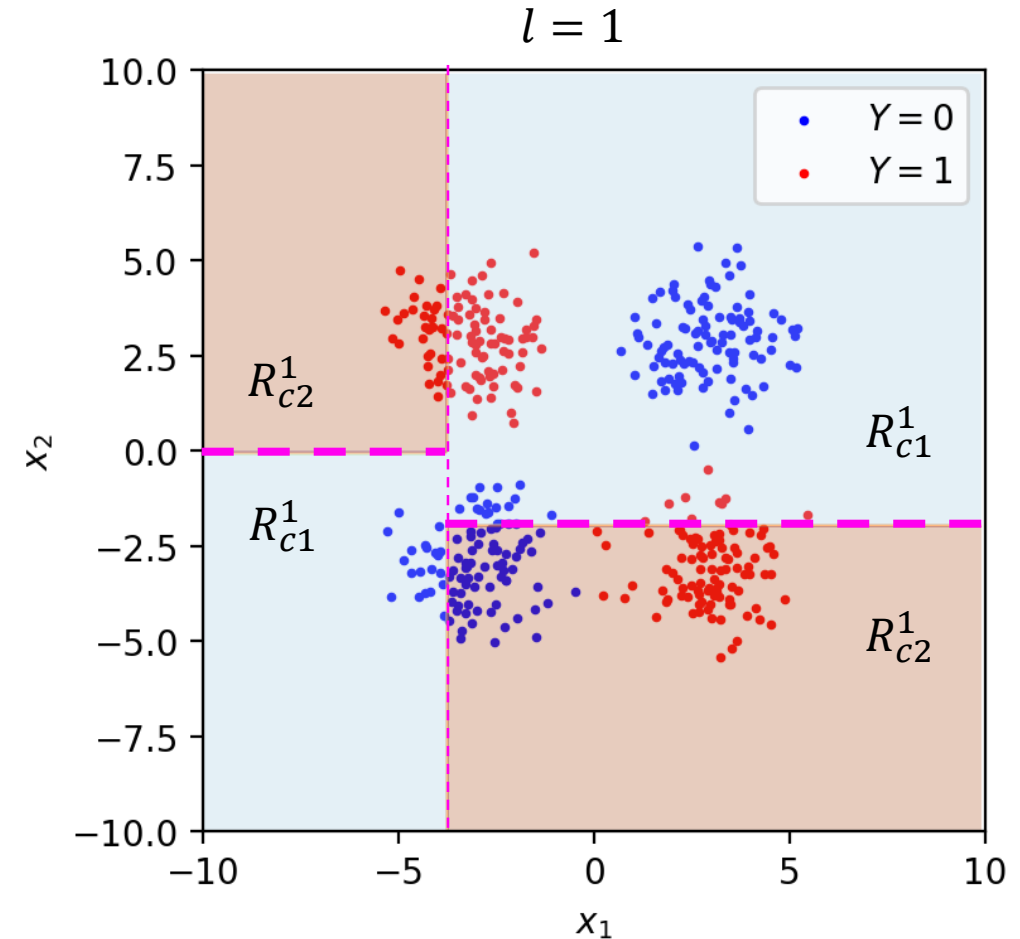
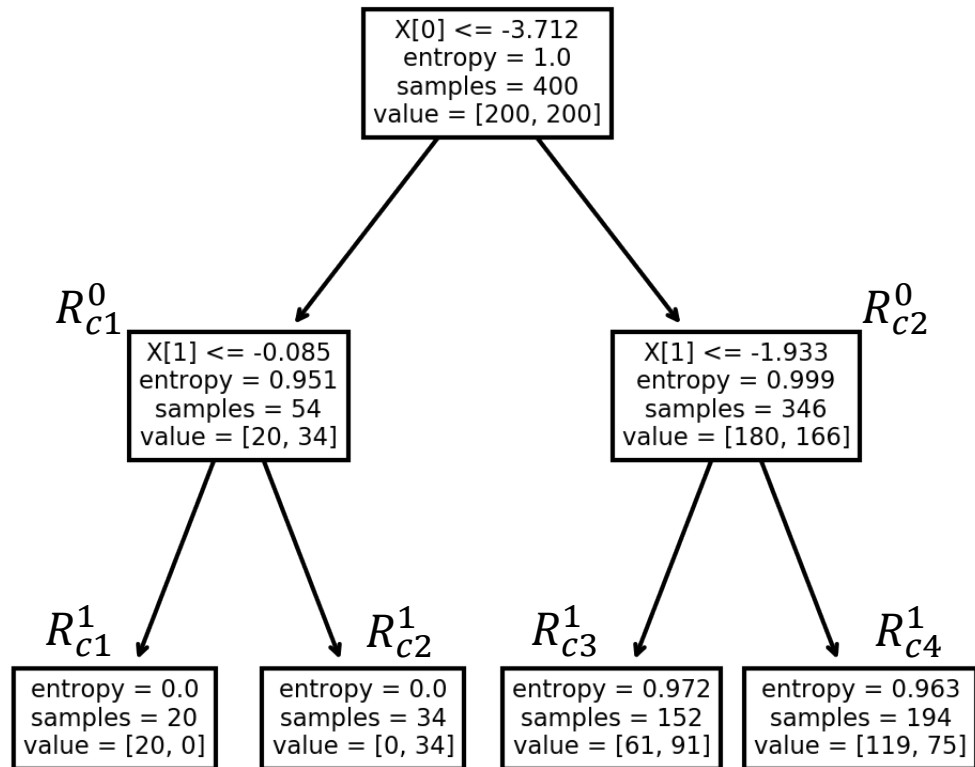
Схема ветвления на первом уровне ( $l = 0$ )



На рисунке: деление выборки всех примеров  $R_p^0$  на  $R_{c1}^0, R_{c2}^0$ .  
Внимание: на рисунке – тестовая выборка, правила деления (номер признака  $j_l$  и пороговое значение  $t^{(l)}$ ) были определены во время обучения.

# DT в режиме исполнения

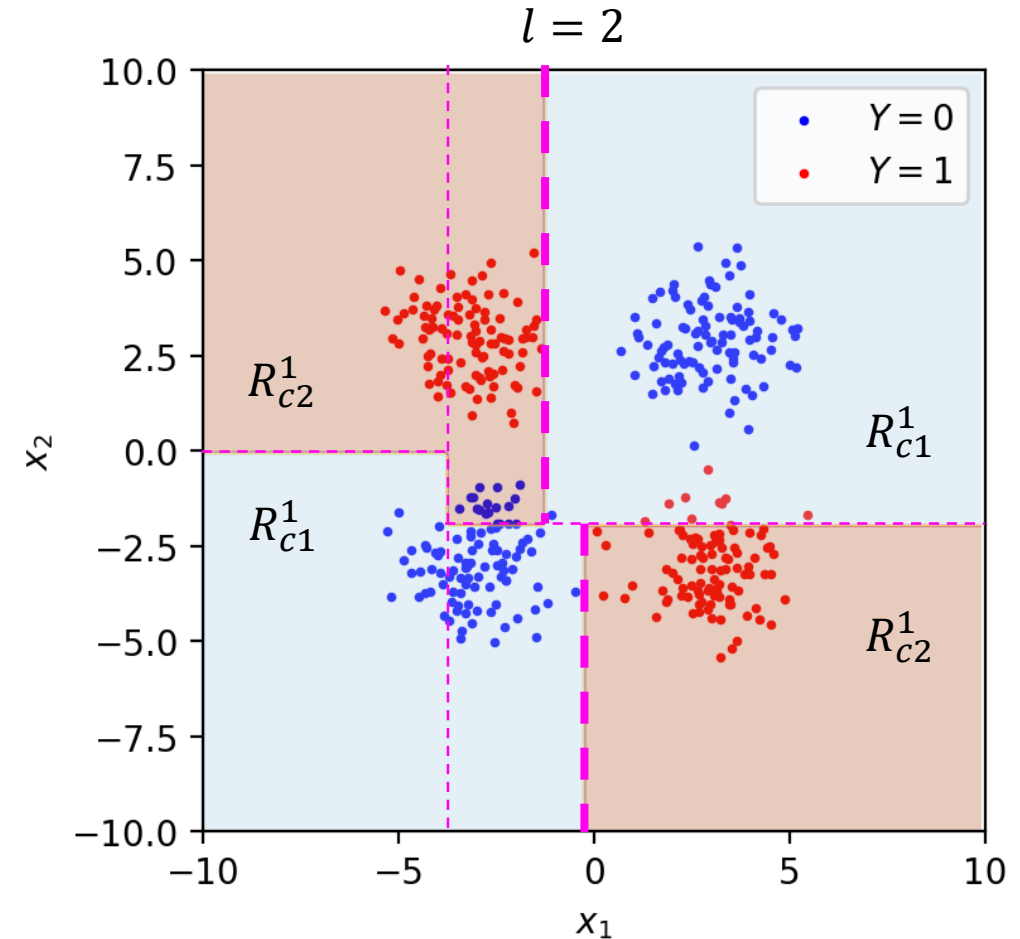
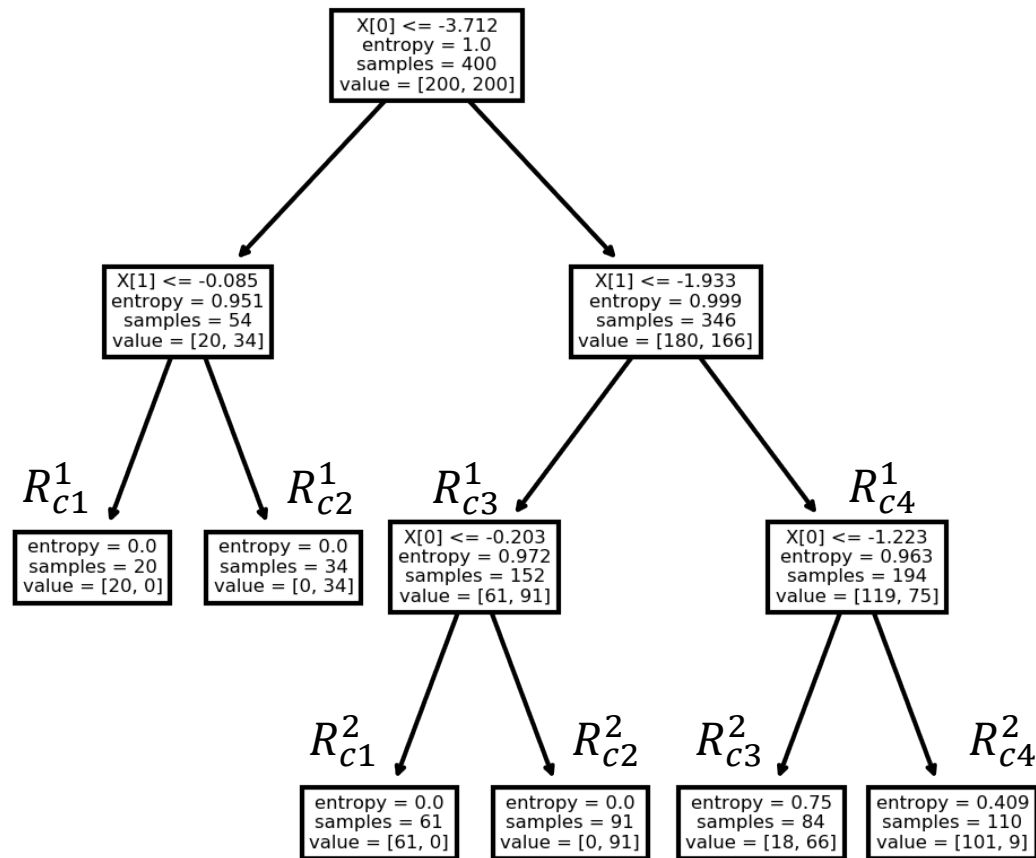
Схема ветвления на втором уровне ( $l = 1$ )



На рисунке: деление выборки всех примеров  $R_p^0$  на второй итерации ветвления. Внимание: на рисунке – тестовая выборка, правила деления (номер признака  $j_l$  и пороговое значение  $t^{(l)}$ ) были определены во время обучения.

# DT в режиме исполнения

Схема ветвления на третьем уровне ( $l = 2$ )



На рисунке: деление выборки всех примеров  $R_p^0$  на третьей итерации ветвления. Внимание: на рисунке – тестовая выборка, правила деления (номер признака  $j_l$  и пороговое значение  $t^{(l)}$ ) были определены во время обучения.

# DT в режиме обучения



ОБУЧЕНИЕ ДЕРЕВЬЕВ ©

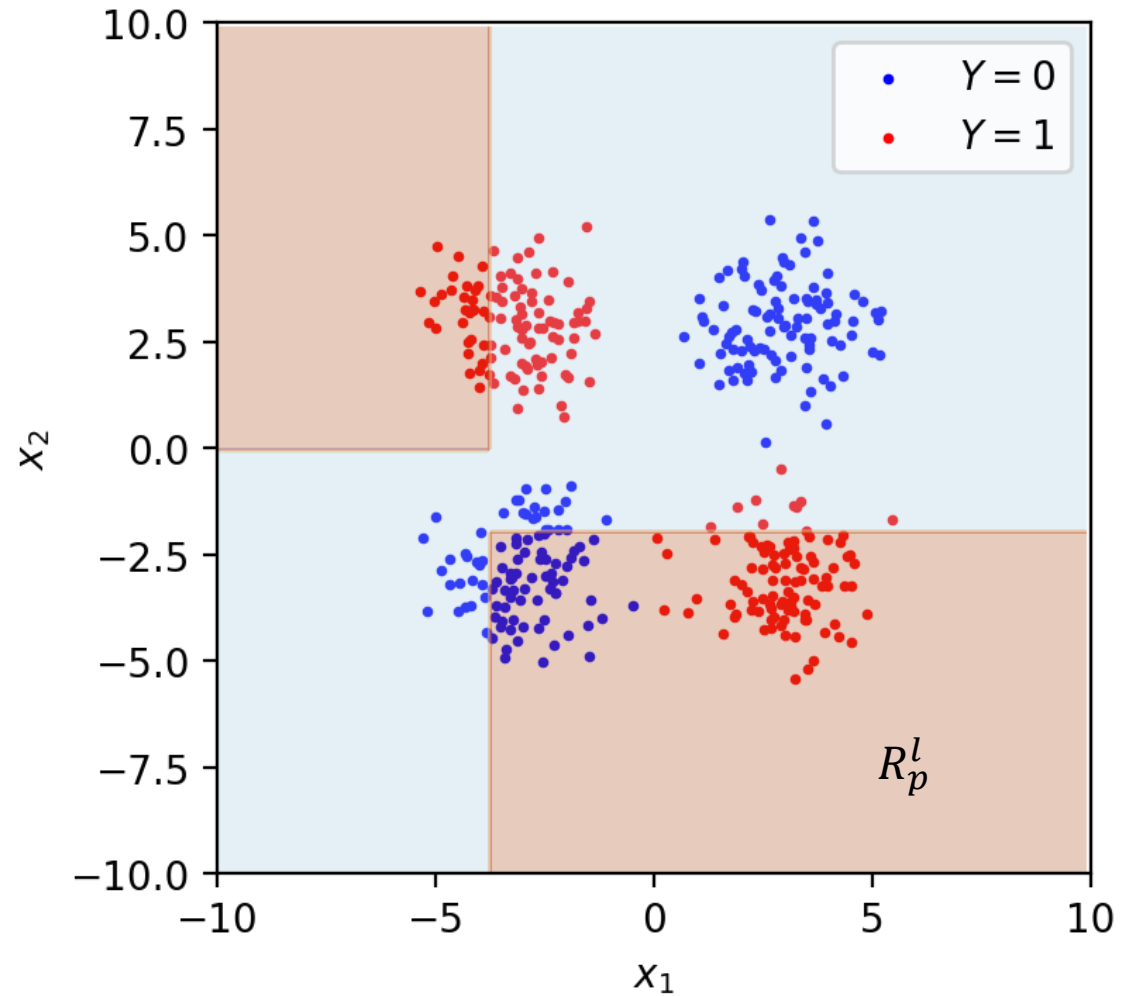
“Обучение деревьев” by Alexander Gavrikov

# DT в режиме обучения

На рисунке – результат после  $l$ -го ветвления тренировочной выборки.

Цель: осуществить деление подвыборки тренировочных примеров  $R_p^l$ .

- Если в  $R_p^l$  - примеры только одного класса, - нет смысла их делить: в обеих областях после разделения будут присваиваться те же самые метки, что и в  $R_p^l$ ; в этом случае разделение не производится, в текущей ветке останавливается ветвление.
- Для ветвления  $s(j_l, t^{(l)})$  следует выбрать номер признака  $j_l$  и пороговое значение  $t^{(l)}$ , исходя из каких-то соображений. КАКИХ?



# DT в режиме обучения

На рисунке – результат после  $l$ -го ветвления тренировочной выборки.

Цель: осуществить деление подвыборки тренировочных примеров  $R_p^l$ .

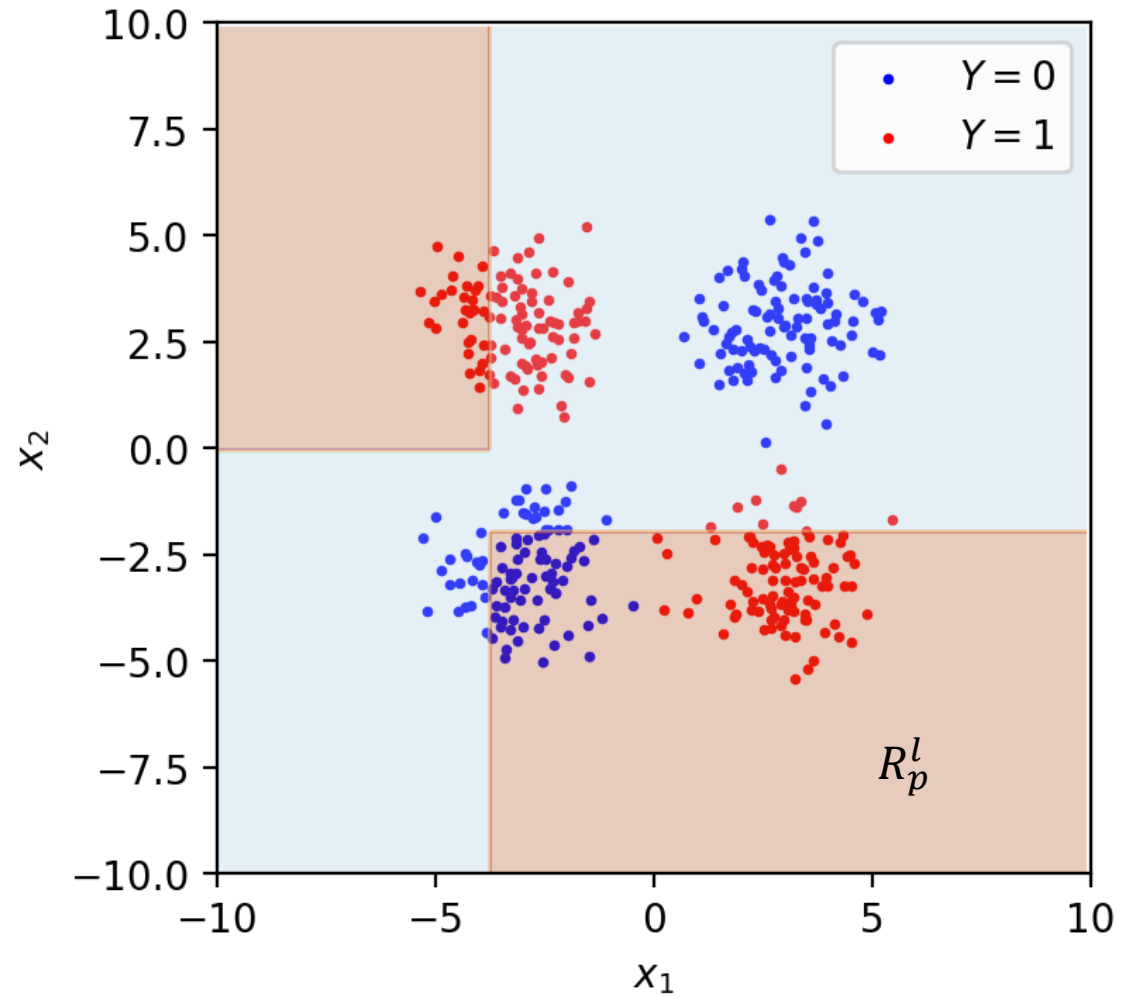
- Для ветвления  $s(j_l, t^{(l)})$  следует выбрать **номер признака  $j_l$**  и **пороговое значение  $t^{(l)}$** , исходя из оптимизации приращения функции потерь. В результате ветвления суммарная функция должна уменьшиться как можно сильнее.

**ИДЕЯ функции потерь:** это ф-я, которая должна характеризовать качество классификации в листе  $R$ . Напомним, что всем примерам, оказавшимся в листе, присваивается одинаковый класс, определяемый голосованием по классам тренировочных примеров в этом листе.

Обозначим:  $p_c^{(R)}$  - доля обучающих примеров класса  $c$  в листе  $R$

Тогда класс, присваиваемый примерам в этом листе на этапе исполнения:

$$\widehat{c}_R = \operatorname{argmax}_{\mathbb{Y}} p_c^{(R)}$$



# ДТ в режиме обучения

**ИДЕЯ функции потерь:** это ф-я, которая должна характеризовать качество классификации в листе  $R$ . Напомним, что всем примерам, оказавшимся в листе, присваивается одинаковый класс, определяемый голосованием по классам тренировочных примеров в этом листе.

Обозначим:  $p_c^{(R)}$  - доля обучающих примеров класса  $c$  в листе  $R$  (может вычисляться с учетом весов примеров  $\{w_i\}$ )

Тогда класс, присваиваемый примерам в этом листе на этапе исполнения:

$$\widehat{c}_R = \operatorname{argmax}_{c \in \mathbb{Y}} p_c^{(R)}$$

Варианты функции потерь:

- Доля неверно классифицированных обучающих примеров:

$$\mathcal{L}_{mc} = 1 - \max_{c \in \mathbb{Y}} p_c^{(R)}$$

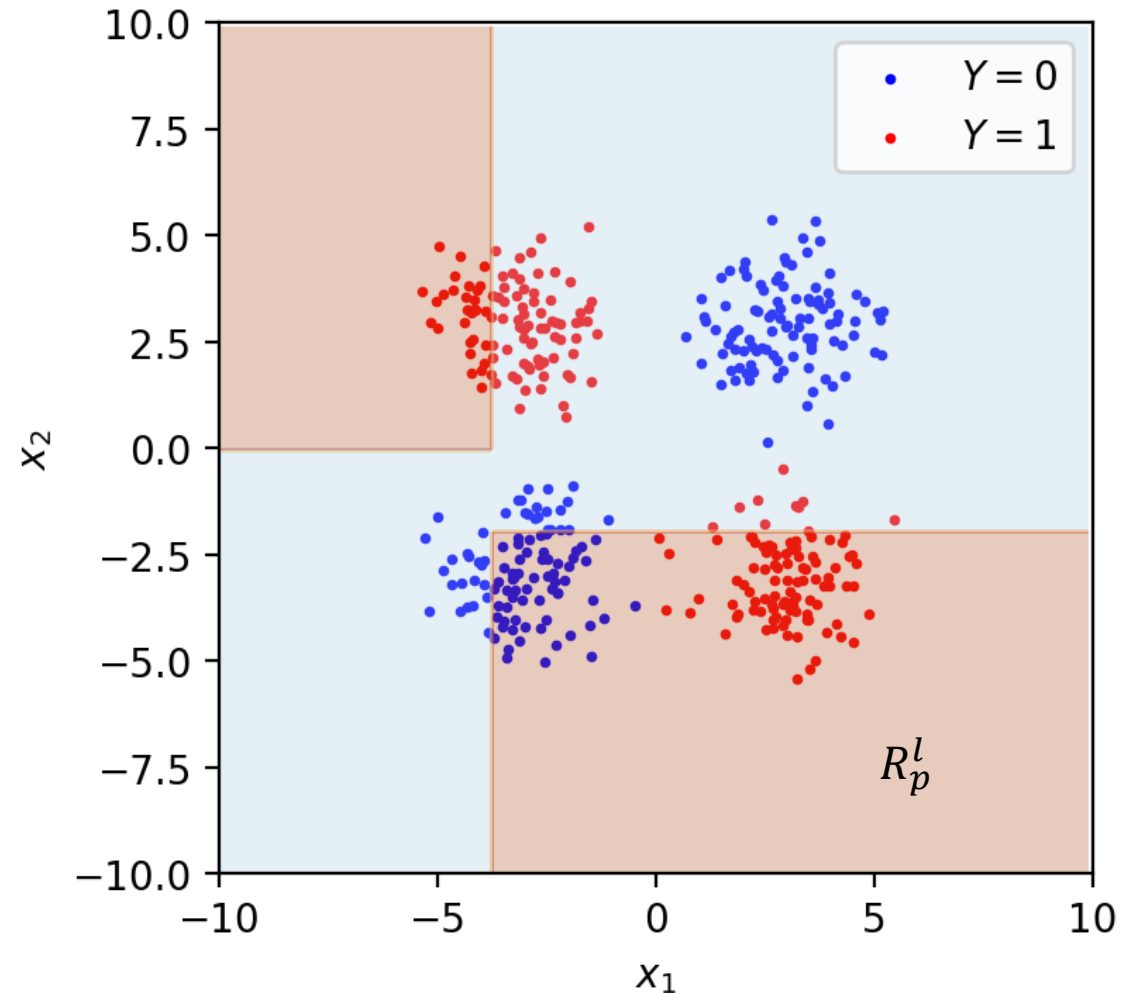
- Коэффициент Джини (отражает степень непохожести классов в  $R$ ):

$$\mathcal{L}_{gini} = \sum_{c \in \mathbb{Y}} p_c^{(R)} (1 - p_c^{(R)})$$

- Перекрестная энтропия:

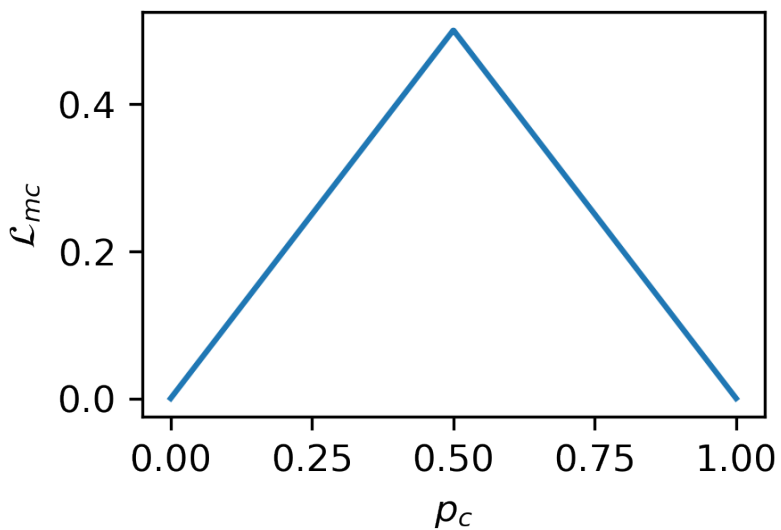
$$\mathcal{L}_{ce} = - \sum_{c \in \mathbb{Y}} p_c^{(R)} \log p_c^{(R)}$$

Общее в этих функциях: чем более однородна подвыборка  $R$  в смысле классов обучающих примеров, тем меньше значение функции. Альтернативно: чем больше доля класса, по которому определяется метка для всех примеров из  $R$ , тем меньше значение функции.

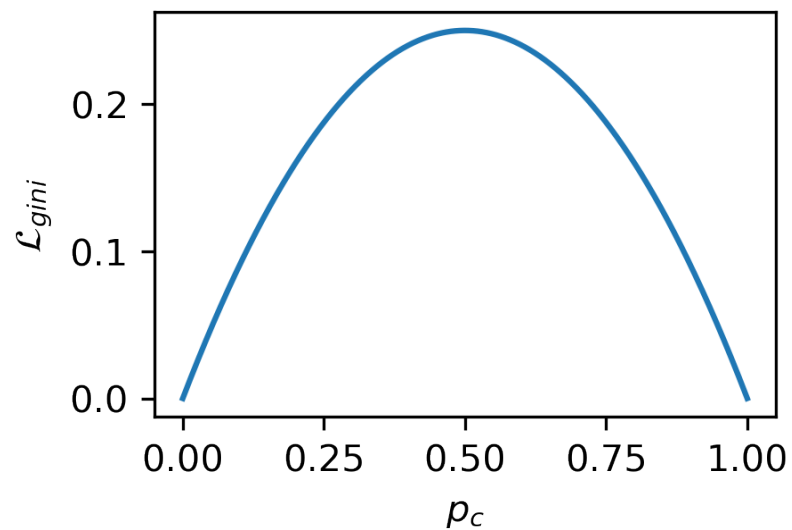


# DT в режиме обучения

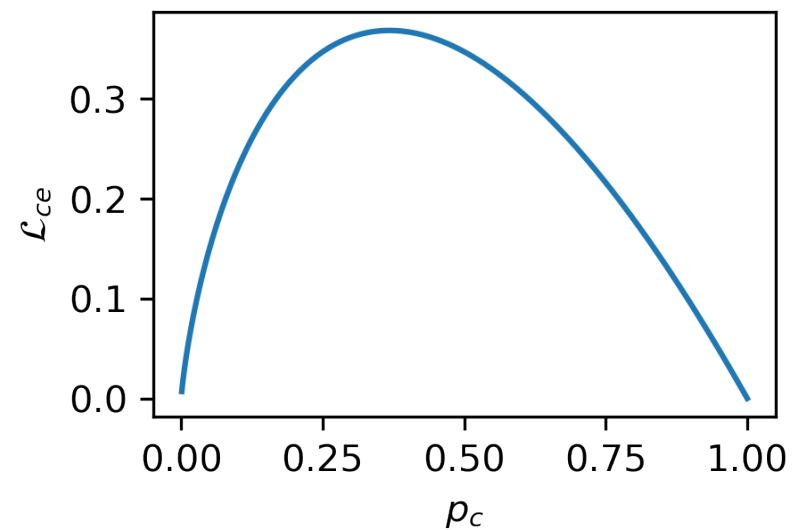
$$\mathcal{L}_{mc} = 1 - \max_{c \in \mathbb{Y}} p_c^{(R)}$$



$$\mathcal{L}_{gini} = \sum_{c \in \mathbb{Y}} p_c^{(R)} (1 - p_c^{(R)})$$



$$\mathcal{L}_{ce} = - \sum_{c \in \mathbb{Y}} p_c^{(R)} \log p_c^{(R)}$$

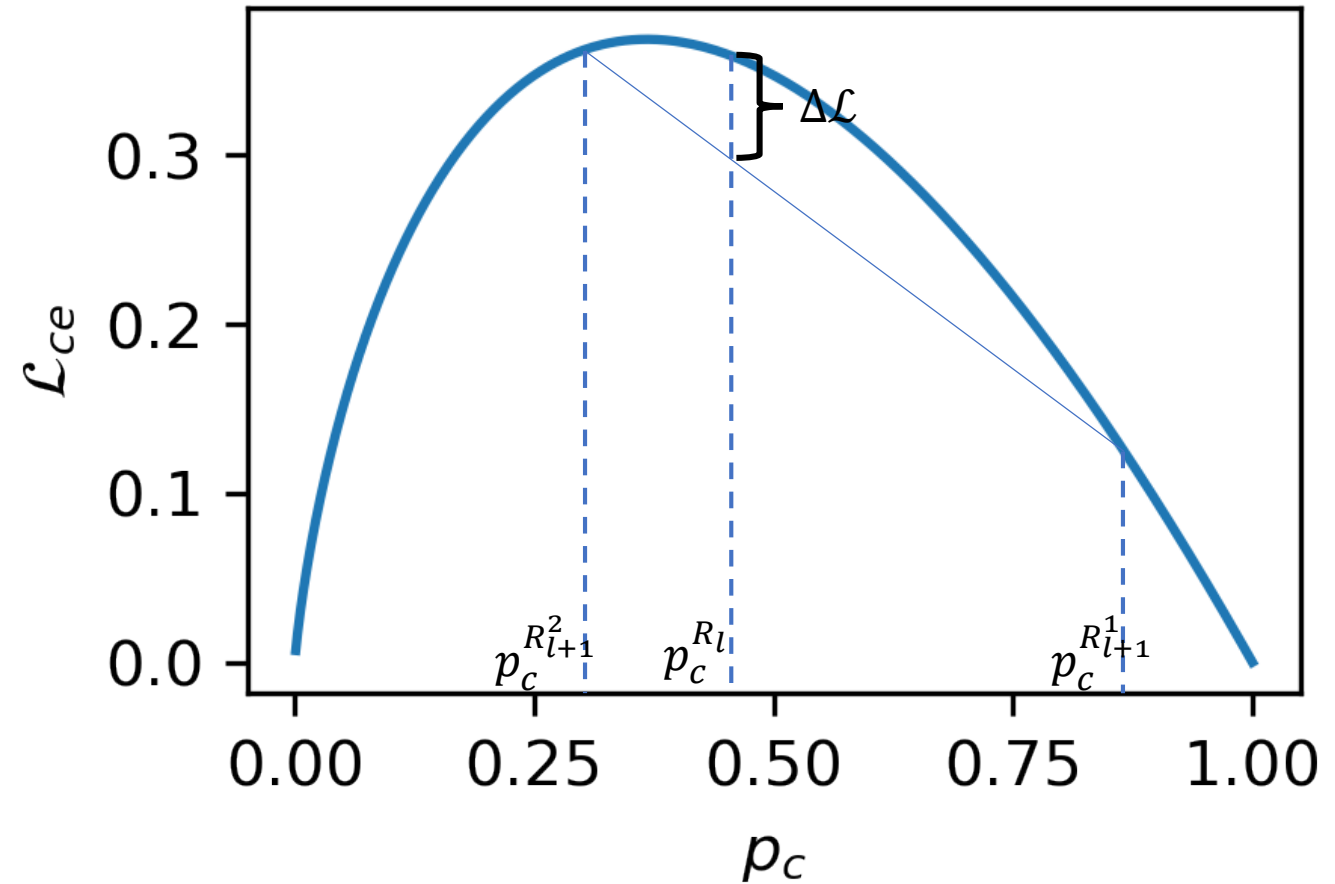




# DT в режиме обучения

Разделение обучающей подвыборки  $R_p^l$  приводит к тому, что суммарная функция потерь снижается на величину  $\Delta\mathcal{L}$ .

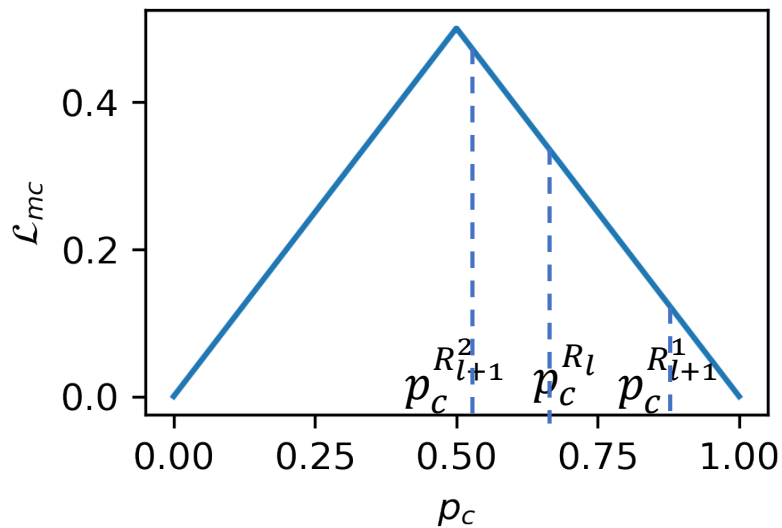
$$\mathcal{L}_{ce} = - \sum_{c \in \mathbb{Y}} p_c^{(R)} \log p_c^{(R)}$$



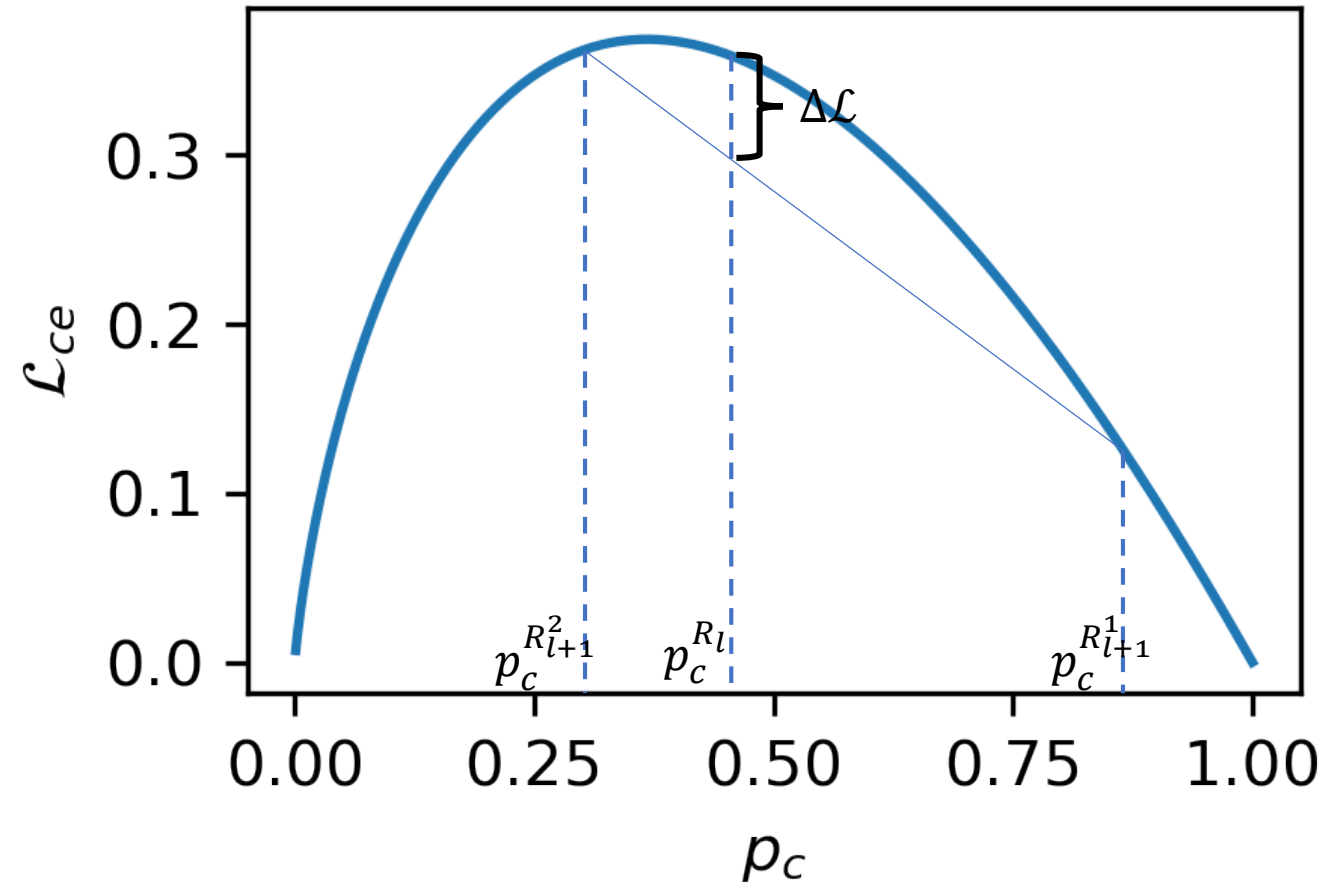
# DT в режиме обучения

Разделение обучающей подвыборки  $R_p^l$  приводит к тому, что суммарная функция потерь снижается на величину  $\Delta\mathcal{L}$ .

Заметим, что в случае функции потерь, характеризующей долю неверно классифицированных примеров, снижение суммарной функции потерь может быть нулевым  $\Rightarrow \mathcal{L}_{mc}$  - не лучший вариант функции потерь для настройки деревьев решений.



$$\mathcal{L}_{ce} = - \sum_{c \in \mathbb{Y}} p_c^{(R)} \log p_c^{(R)}$$



# DT в режиме обучения

На рисунке – результат после  $l$ -го ветвления тренировочной выборки.

Цель: осуществить деление подвыборки тренировочных примеров  $R_p^l$ .

- Для ветвления  $s(j_{l+1}, t^{(l+1)})$  следует выбрать **номер признака  $j_{l+1}$**  и **пороговое значение  $t^{(l+1)}$** , исходя из оптимизации приращения функции потерь. В результате ветвления суммарная функция должна уменьшиться как можно сильнее.

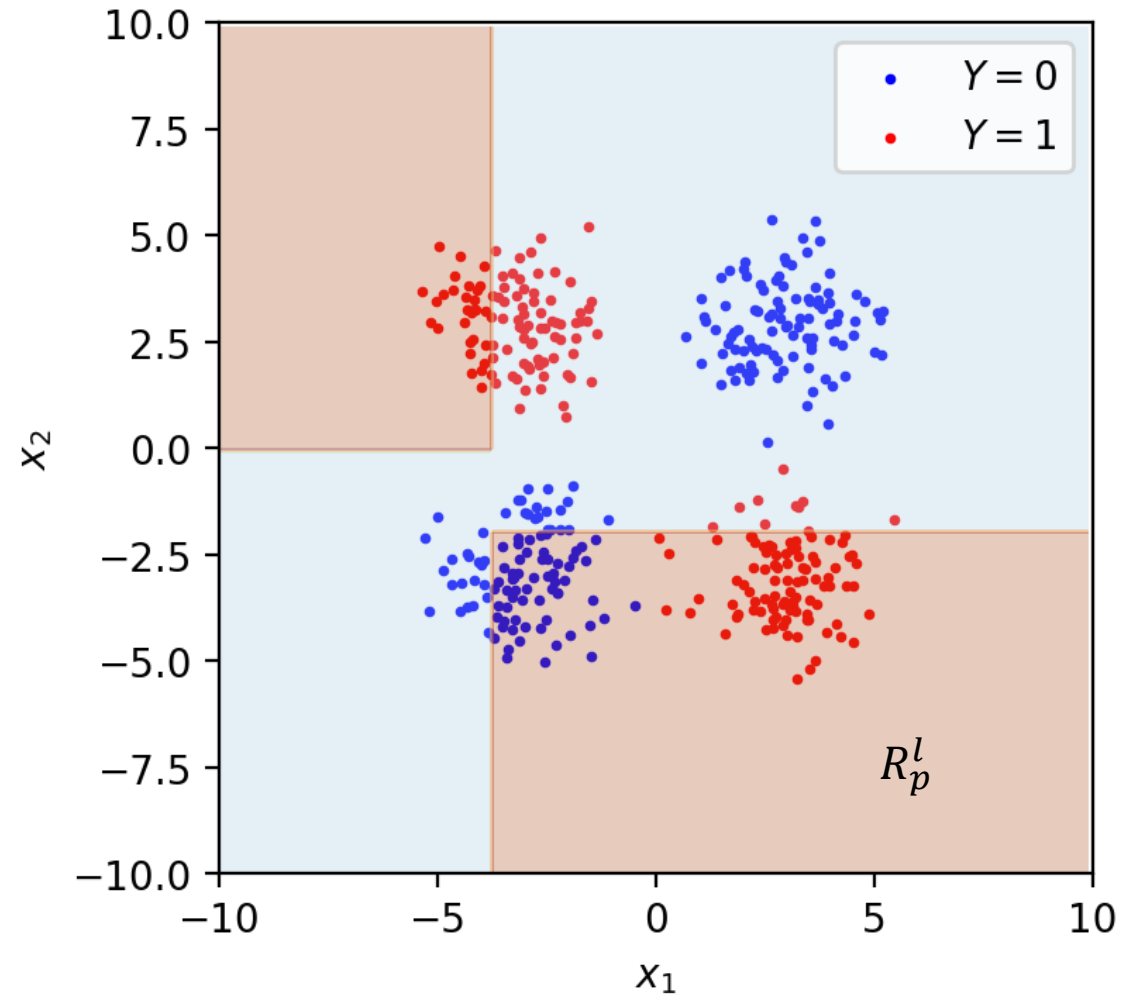
Разделение обучающей подвыборки  $R_p^l$  приводит к тому, что суммарная функция потерь снижается на величину  $\Delta\mathcal{L}$ .

Это означает, что можно искать разделение  $l + 1$  как решение задачи оптимизации:

$$j_{l+1}, t^{(l+1)} = \operatorname{argmax}_{j \in [1 \dots f], t_j \in \mathbb{X}_j} \Delta\mathcal{L}(R_p^l, R_{c1}^l, R_{c2}^l)$$

- $f$  – количество признаков признакового описания объектов
- пороговое значение  $t_j$  ищется среди всех возможных значений  $j$ -го признака

это – т.н. «жадный» (greedy) подход: получение локально оптимального решения на каждой итерации.



# DT в режиме обучения

Псевдоалгоритм построения дерева решений на основании выборки  $R = \{X, Y\}$ :

Начальное состояние:

Выборка  $R_p^l = R$ ;  $l = 0$

Выполнять до тех пор, пока не будет удовлетворено условие останова:

1. метка для присвоения элементам текущего подмножества:

$$\hat{c}(R_p^l) = \operatorname{argmax}_{c \in \mathbb{Y}} p_c^{(R_p^l)}$$

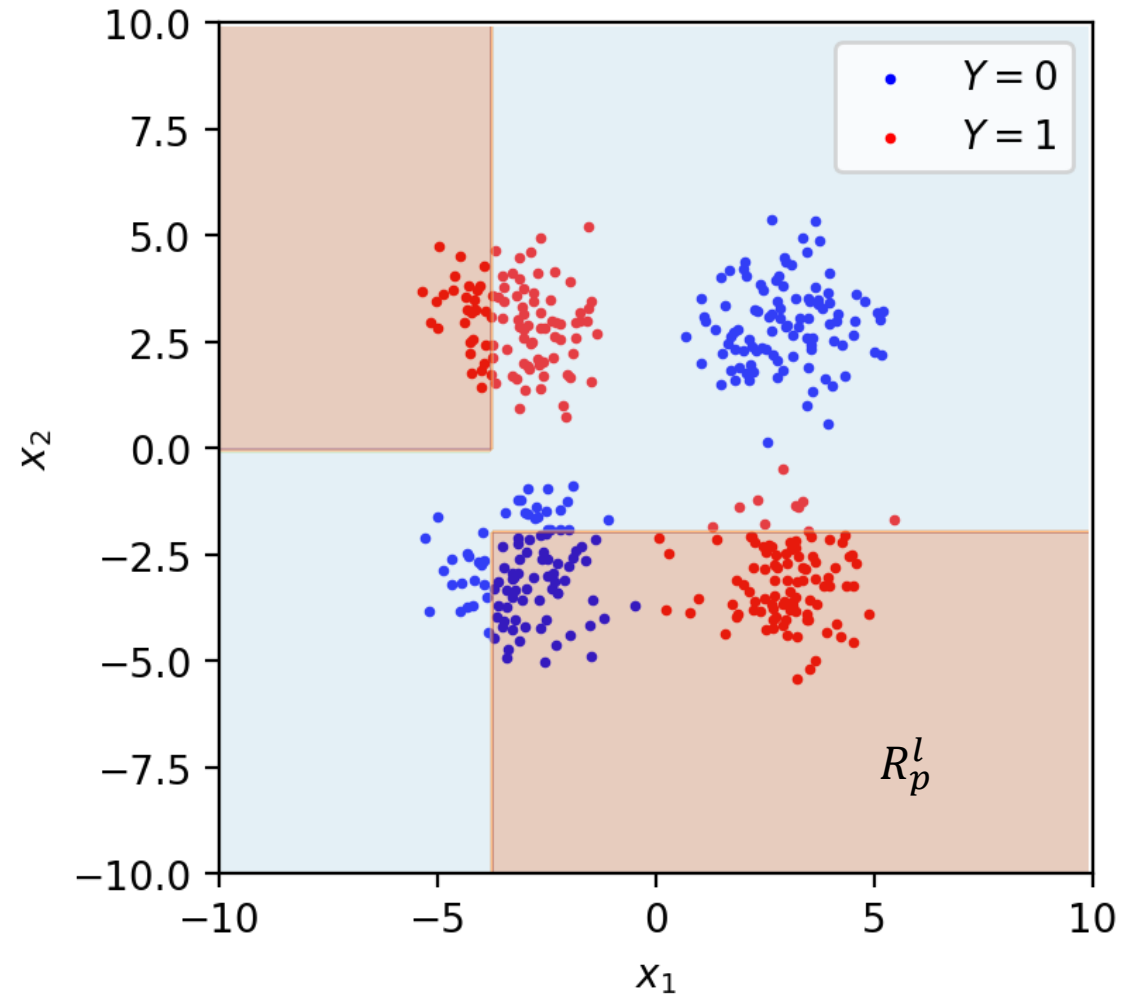
1.  $\mathcal{L} = \mathcal{L}_{ce}(R_p^l, \hat{c}(R_p^l))$
2. если  $\mathcal{L} = 0$  – останов в этой ветке дерева (такое может быть, когда в  $R_p^l$  - элементы обучающей выборки только одного класса)
3. если  $\mathcal{L} > 0$  – поиск признака и порогового значения по этому признаку в рамках оптимизационной задачи:

$$R_{c1}^l = \{X | x^{(j_l)} < t^{(l)}, x \in R_p^l\}$$

$$R_{c2}^l = \{X | x^{(j_l)} \geq t^{(l)}, x \in R_p^l\}$$

$$j_l, t^{(l)} = \operatorname{argmax}_{j \in [1 \dots f], t_j \in \mathbb{X}_j} \Delta \mathcal{L}(R_p^l, R_{c1}^l, R_{c2}^l)$$

4. повторять с п.1 для подвыборок  $R_{c1}^l, R_{c2}^l$  как «родительских».



# Деревья решений в задаче регрессии

## Regression trees

Меняются только:

- способ вычисления целевой переменной в листе:

$$\hat{y}(R) = \frac{1}{|R|} \sum_{i=1}^{|R|} y_i$$

- форма функции потерь:

$$\mathcal{L}(R) = MSE(\hat{y}(R), Y_R)$$

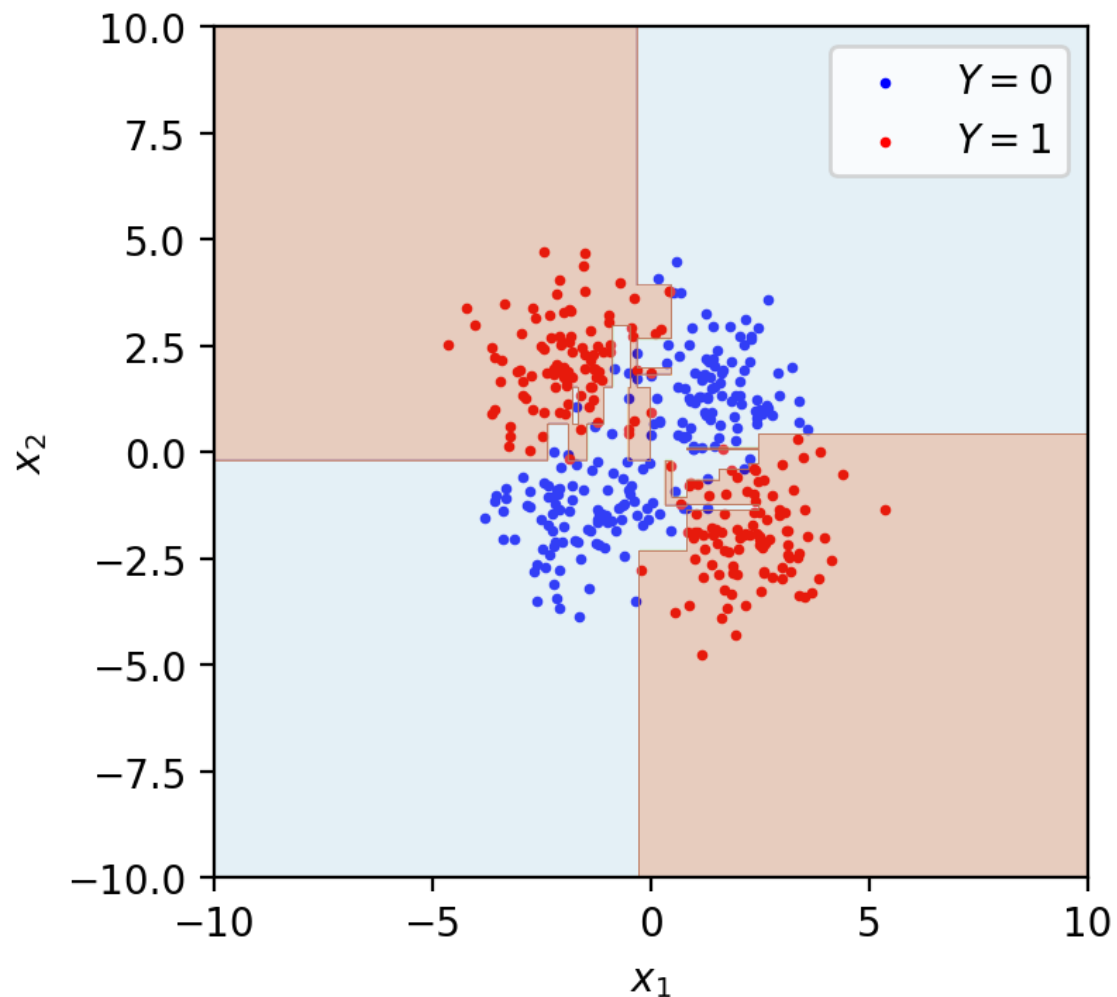
# Деревья решений

## Особенности метода

- алгоритм обучения быстрый и вычислительно недорогой
- в режиме применения метод интерпретируем (можно продемонстрировать последовательность вопросов и соответствующих решений в ветвлениях)
- ДР естественным образом учитывают категориальные признаки: правило ветвления логическое, поэтому для ветвлений можно ставить условие типа совпадения категорий:

$$R_{c1}^l = \{X | x^{(j_l)} == t^{(l)}, x \in R_p^l\}$$

$$R_{c2}^l = \{X | x^{(j_l)} \neq t^{(l)}, x \in R_p^l\}$$



# Деревья решений

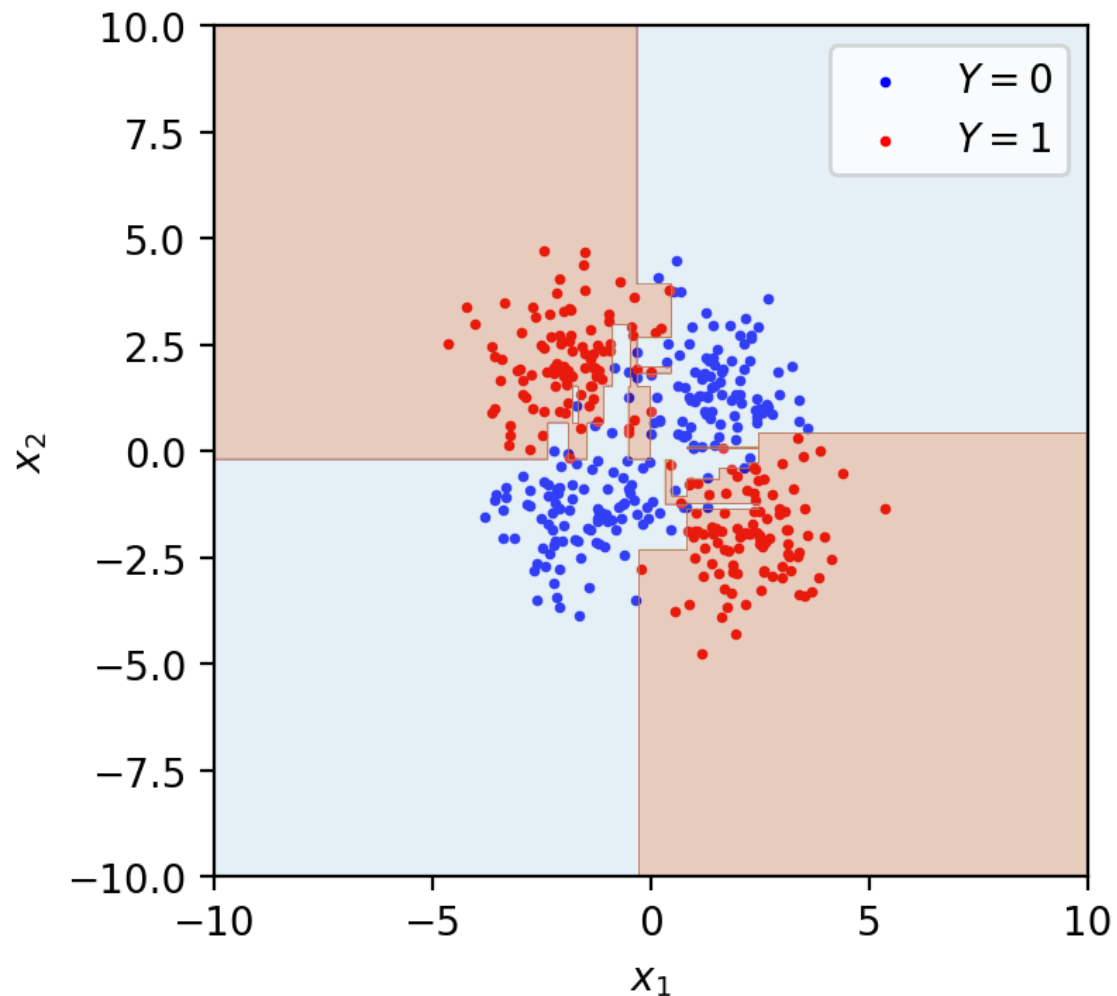
## Особенности метода

- деревья решений **сильно склонны к переобучению**; для борьбы с этим можно применять эвристические способы ограничения выразительной способности во время обучения:
  - ограничивать глубину дерева (`max_depth*`): если достигнута максимальная глубина, критерий останова ветвления считается выполненным;
  - ограничивать минимальный размер листа (`min_samples_leaf*`): если количество элементов в очередном листе не превышает этого значения, критерий останова ветвления в этой ветке считается выполненным;
  - ограничивать максимальное количество листьев дерева при его построении (`max_leaf_nodes*`): если количество листьев достигло определенного предела, обучение всего дерева прекращается;
  - ограничивать минимальное снижение функции потерь при ветвлении (`min_impurity_decrease*`): если максимально достижимое снижение функции потерь при ветвлении не превышает порогового, критерий останова ветвления в этой ветке считается выполненным

Эти гиперпараметры можно оптимизировать на основании меры качества на валидационной выборке, на выборках OOB в подходе bootstrap или в подходе скользящего контроля.

- альтернативно: ведется полное обучение всего дерева без ограничений, после чего выполняется обрезка (`pruning`) дерева: оставляются ветвления до определенной глубины (подбирается на основании меры качества на валидационной выборке).

**НИКОГДА НЕ ПРИМЕНЯЙТЕ** деревья решений как таковые!



\* приводятся имена параметров моделей `DecisionTreeClassifier` и `DecisionTreeRegressor` пакета `scikit-learn`