

# Лекция 12: Деревья решений

# Методы, основанные на деревьях решений

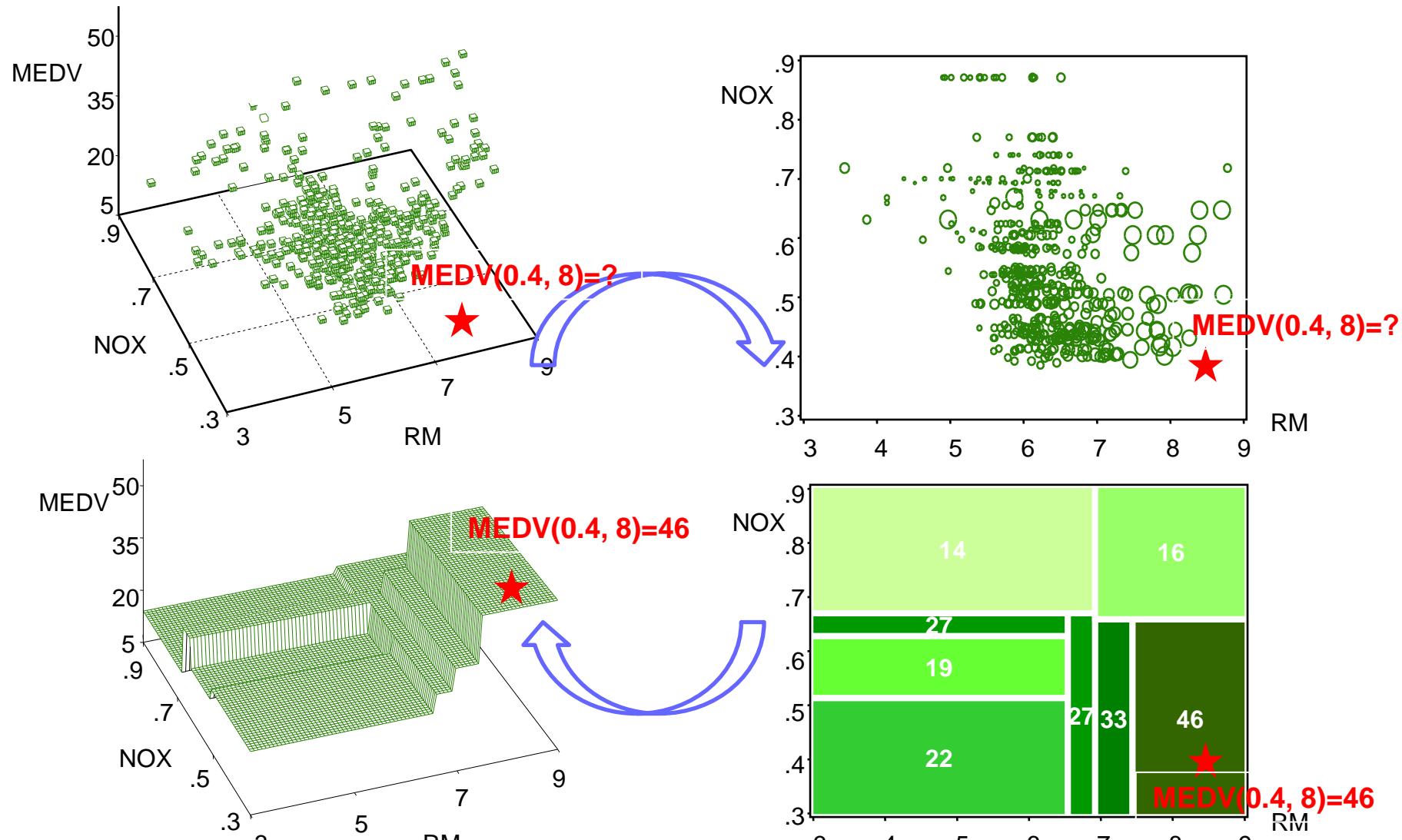
## ■ Ключевые особенности:

- Эти методы используют *стратификацию* или *сегментирование* пространства признаков на области.
- Для сегментации пространства признаков может использоваться *набор правил*, который можно представить в виде дерева
- Деревья решений могут применяться как к *задачам регрессии*, так и к *классификации*.
- Методы, основанные на деревьях, просты в *интерпретации*, при этом показывают достаточно хорошие результаты по точности прогнозирования.
- Нестабильные модели – это плюс для ансамблей *бэггинг*, *методы случайного леса* и *бустинг*. Эти методы строят множество деревьев, результаты прогнозирования которых потом объединяются для получения итогового прогноза.

# Деревья решений в задачах классификации и регрессии

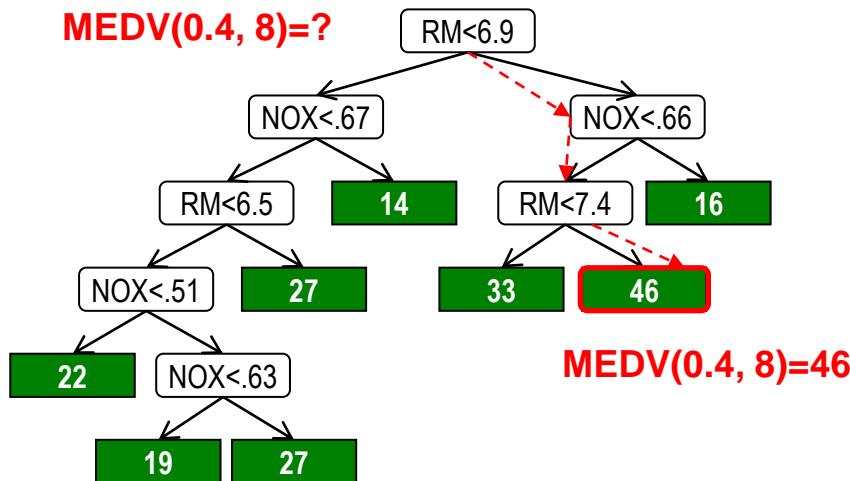
- Дерево решений - граф (древовидная структура), в котором:
  - Внутренние узлы – условия на атрибуты
  - Каждая исходящая ветка соответствует выходному значению условия, ветка целиком – альтернативное решение
  - В листьях метки классов (или распределение классов) или значения целевой переменной для регрессии
  - Каждому узлу соответствует область в пространстве признаков R
  - Области для листьев – финальные, не содержат внутри других областей
- Построение дерева обычно – 2 фазы
  - «рост» : в начале в корне все примеры, далее рекурсивное разбиение множества примеров по выбранному(ым) атрибуту(ам)
  - «отсечение» ветвей pruning - выявление и удаление ветвей (решений), приводящих к шуму или к выбросам
- Применение дерева решений для нового объекта
  - Проверка атрибутов – путь по ветви до листа. В листе отклик.

# Непрерывный отклик



# Непрерывный отклик

$$\text{MEDV}(0.4, 8) = ?$$



If RM ∈ {values} and NOX ∈ {values}, then MEDV=value.

<u>Leaf</u>	<u>RM</u>	<u>NOX</u>	<u>Прогноз MEDV</u>
1	<6.5	<.51	22
2	<6.5	[.51, .63)	19
3	<6.5	[.63, .67)	27
4	[6.5, 6.9)	<.67	27
5	<6.9	$\geq .67$	14
6	[6.9, 7.4)	<.66	33
7	$\geq 7.4$	<.66	46
8	$\geq 6.9$	$\geq .66$	16

## ■ Модели регрессии на основе деревьев решений:

- #### Решающая (регрессионная) функция кусочно-постоянная:

$a(x) = \sum_{m=1}^M c_m I(x \in R_m)$ , где  $c_m$  - константа,  $M$  – число регионов (листьев)

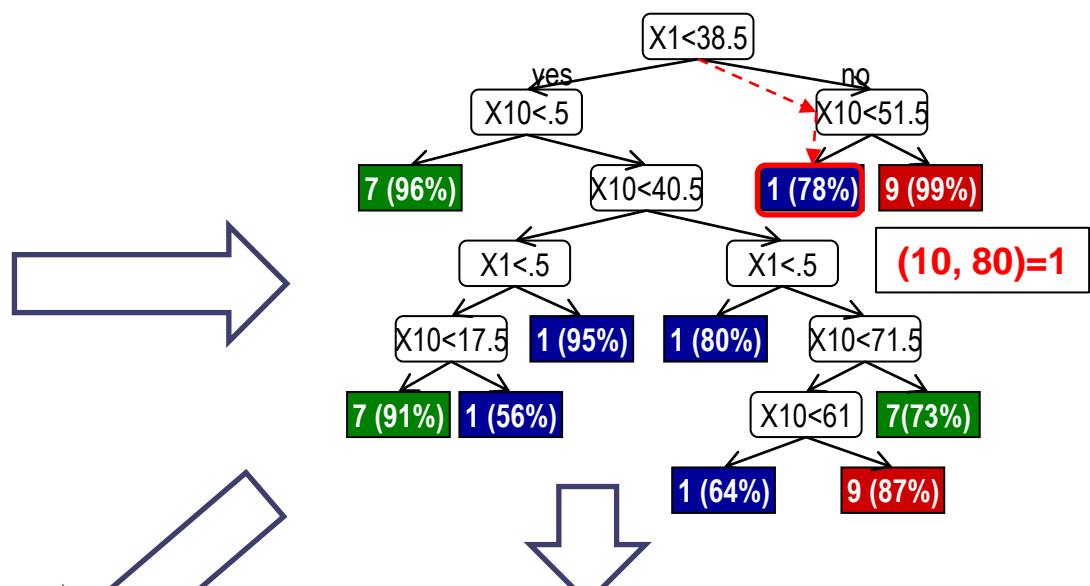
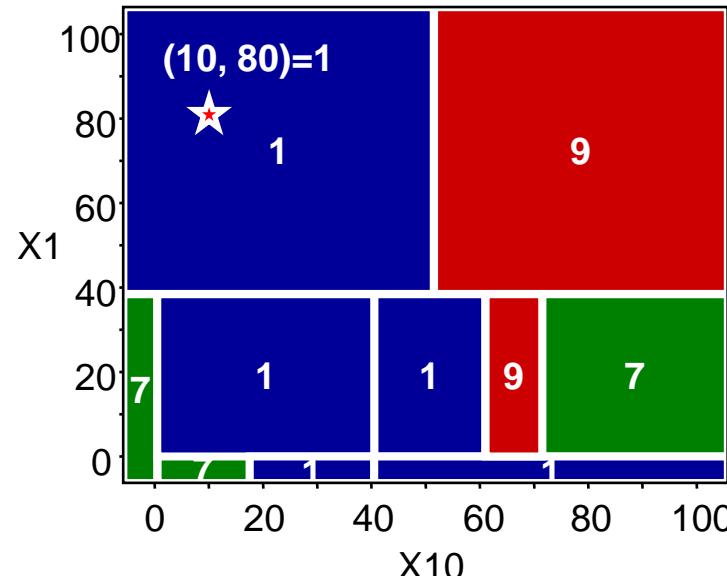
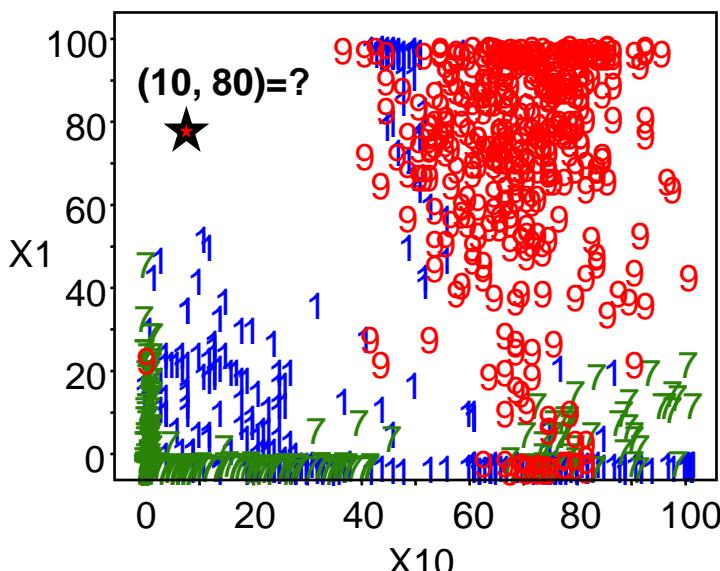
$R_m = \prod_{i=1}^d I(a_{mi} < x_i \leq b_{mi})$  или  $R_m = \Lambda_{i=1}^d [a_{mi} < x_i \leq b_{mi}]$  - «прямоугольный» регион (для числовых признаков),  $d$  – р

#### ■ Эмпирический риск (однородность регионов):

- один из вариантов, без регуляризации и с кв. функцией потерь:

$$Q(a(x), \{(x_i, y_i)\}_{i=1}^l) = \sum_{i=1}^l \sum_{m=1}^M (y_i - a(x))^2 I(x_i \in R_m) \rightarrow \min_{R_m, c_m} \Rightarrow c_m = \frac{1}{|R_m|} \sum_{i: x_i \in R_m} y_i$$

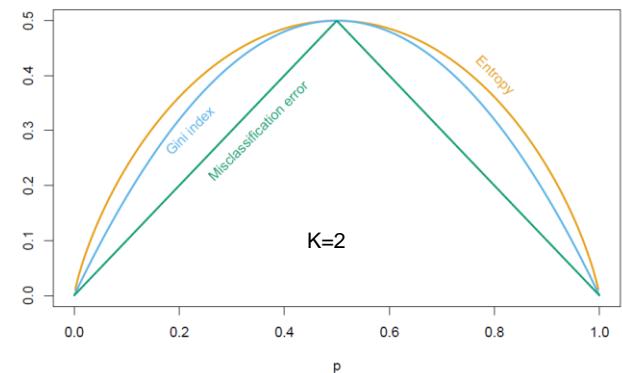
# Категориальный отклик



Leaf	$\Pr(1 x)$	$\Pr(7 x)$	$\Pr(9 x)$	Прогноз
1	.03	.96	.01	7
2	.09	.91	.00	7
3	.56	.44	.00	1
4	.95	.05	.00	1
5	.80	.10	.10	1
6	.64	.09	.27	1
7	.00	.13	.87	9
8	.10	.73	.17	7
9	.78	.01	.21	1

# Категориальный отклик

- Модели классификации на основе деревьев решений:
  - Классификатор  $a(x) = \operatorname{argmax}_{k,m} [p_{mk} I(x \in R_m)]$ , где  
 $p_{mk} = P(y = k | x \in R_m) = \frac{1}{|R_m|} \sum_{i:x_i \in R_m} (y_i = k)$  - оценка вероятности класса  $k$  в регионе  $m$ , если  $m(x)$  – индекс региона, куда попало наблюдение  $x$ , то  $p_{m(x)k}$  является дискриминантной функцией класса  $k$
- Эмпирический риск (однородность регионов):
  - Ошибка классификации (не гладкая):
$$Q_{miss} = \sum_{i=1}^l \sum_{m=1}^M I(x_i \in R_m) \frac{I(y_i \neq a(x_i))}{|R_m|} = \sum_{i=1}^l (1 - p_{m(x_i) a(x_i)})$$
  - Индекс Джини (гладкая, ограничивает  $Q_{miss}$  сверху):
$$Q_{Gini} = \sum_{m=1}^M \sum_{k=1}^K p_{mk} (1 - p_{mk})$$
  - Энтропия (ограничивает  $Q_{Gini}$  сверху):
$$Q_{KL} = - \sum_{m=1}^M \sum_{k=1}^K p_{mk} \log_2(p_{mk})$$



# Более сложные варианты деревьев

## ■ Типы регионов:

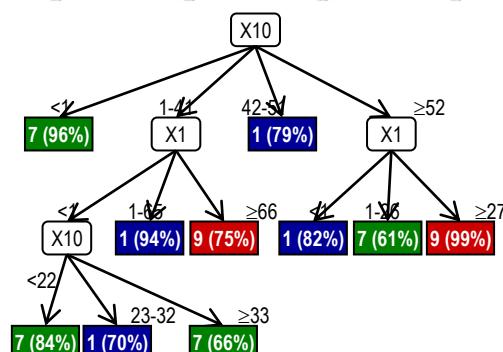
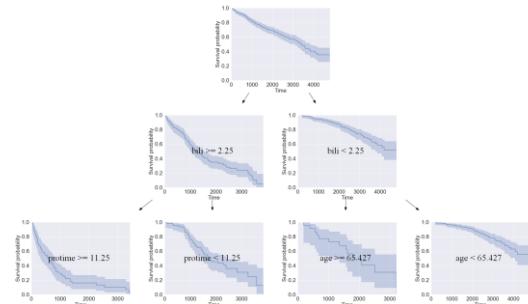
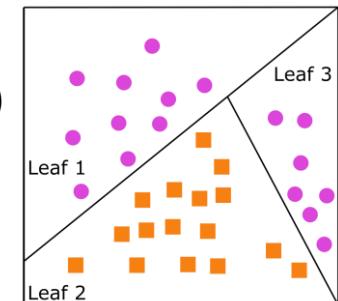
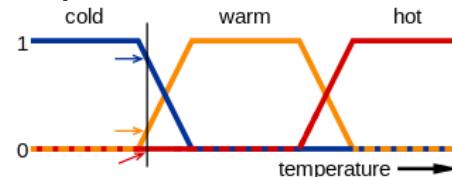
- Порядковые и числовые предикторы ( $a_{mi} < x_i \leq b_{mi}$ )
  - Категориальные предикторы ( $x_i \in S_{mi}$ )
  - «Многогранники» ( $\bigcup \sum_i w_{mi}x_i \leq b_m$ )
  - «Сфераы» ( $\sum_i (a_{mi} - x_i)^2 \leq b_m$ )

## ■ Стратифицированные модели :

- В регионах не константа, а ф-ция  $c_m(x)$ ,  
например, непараметрическая модель (сплайн)

## ■ Разбиение регионов:

- Бинарное – каждый регион делится на два
  - Множественное – много ветвей в дереве
  - Нечеткие правила – отдельная история



# Процесс построения деревьев решений – рекурсивное разбиение

- Цель:
  - найти непересекающиеся области  $R_1, \dots, R_M$ ,  $\forall i \neq j: R_i \cap R_j = \emptyset$ , покрывающие все пространство признаков  $X = \bigcup_m R_m$  так, чтобы поведение отклика внутри каждого региона было максимально однородным, т.е. минимизировать некий целевой критерий разбиения
$$Q(R_1, \dots, R_M) \rightarrow \min$$
- Подход на основе рекурсивного разбиения (**нисходящий, жадный**):
  - Вычислительно нецелесообразно (**NP-полная задача**) рассматривать все возможные разбиения пространства признаков, даже в рамках фиксированной структуры правил и критерия разбиения
  - **Нисходящий** - начинается в корне дерева, где один регион (все пространство признаков), затем последовательно *рекурсивно* разбиваются доступные регионы на более мелкие и каждое разбиение приводит к образованию новых ветвей, расположенных ниже по дереву.
  - **Жадный** – лучшее разбиение выбирается по критерию на каждом шаге, просмотра вглубь нет (иначе тоже NP-полная задача), не приведет к глобально лучшему дереву даже по выбранному критерию

# Рекурсивное разбиение

## ■ Алгоритм поиска разбиения для региона $R$

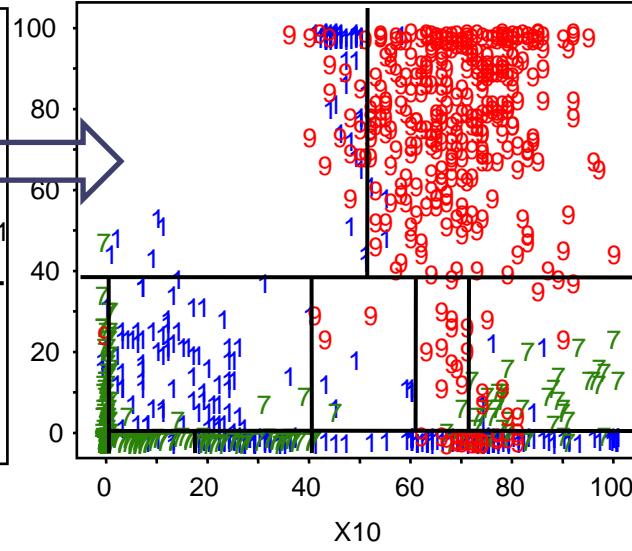
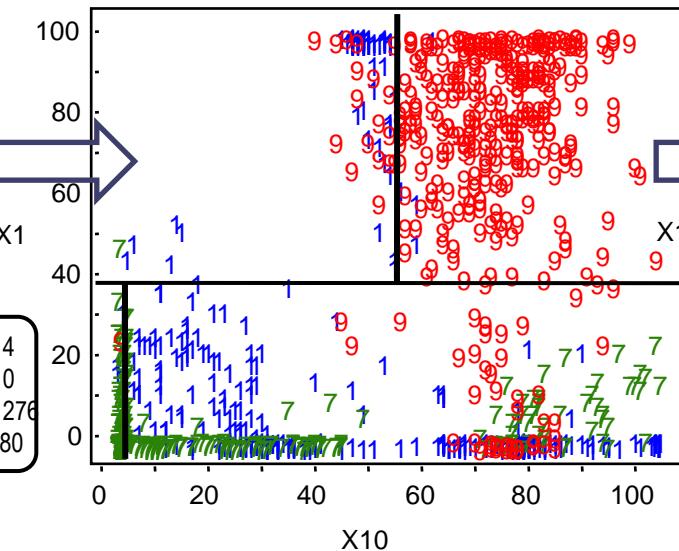
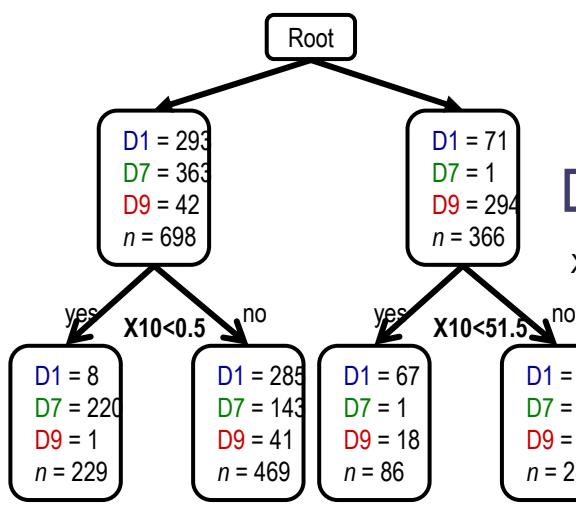
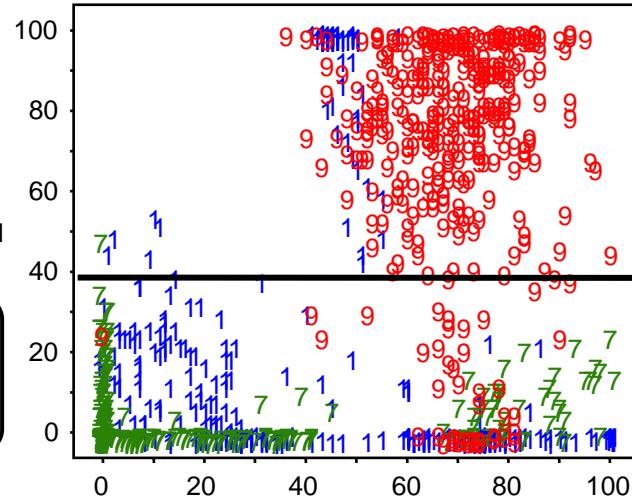
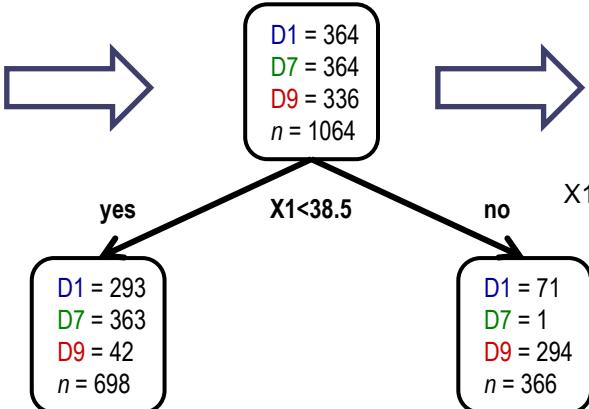
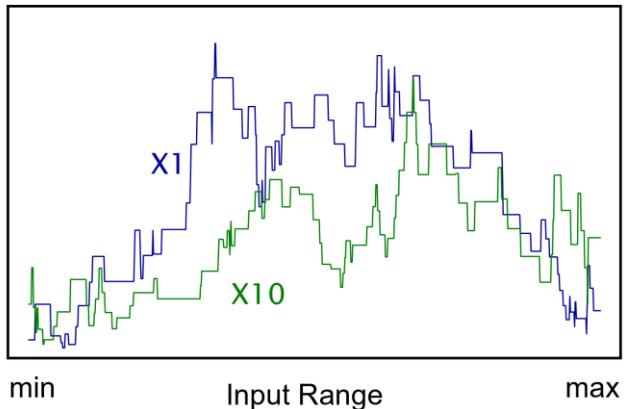
начинаем с корня, первый регион разбиения равен всему  $R=X$ .

1. Проверить условия остановки/роста дерева для данного региона
2. Сформировать множество гипотез  $\{f_i\}$  для разбиения, таких что  $f_i: R \rightarrow B$  разбивает «родительский» регион на  $B$  «дочерних» регионов ( $B$  – число ветвей), удовлетворяющих условиям остановки/роста
3. Рассчитать значение критерия разбиения  $Q(f_i)$  для каждой гипотезы и выбрать лучшую по критерию
4. Дорастить дерево (лист, соответствующий разбиваемому региону превращается во внутренний узел) новыми  $B$  ветвями, заменив «родительский» регион на  $B$  «дочерних»
5. Для каждого полученного региона (соответствующего новым листьям) применить Алгоритм поиска разбиения

## ■ Особенности (упрощения) для поиска прямоугольных регионов:

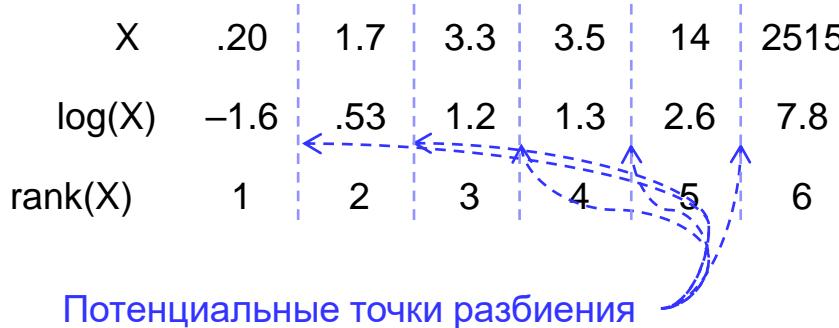
- Гипотезы разбиения (в виде порогов для порядковых/числовых и в виде подмножеств для категориальных) строятся по каждому предиктору отдельно, выбирается лучшая по предиктору
- Затем лучшие гипотезы сравниваются между предикторами

# Рекурсивное разбиение



# Гипотезы-кандидаты для поиска разбиения числового предиктора

- Рассмотрим прямоугольные регионы для числового предиктора  $x$ :
  - Разбиваем одномерный «родительский» регион ( $a < x \leq b$ ) с  $N$  различными значениями на  $B$  ветвей
  - Надо сформировать варианты разбиения (гипотезы), каждая задается порогами  $a < \theta_1 < \theta_2 < \dots < \theta_{B-2} < \theta_{B-1} < b$ , ветви задаются условиями ( $a < x \leq \theta_1$ ), ( $\theta_1 < x \leq \theta_2$ ), ..., ( $\theta_{B-1} < x \leq b$ )
- Варианты разбиения для числового или порядкового предиктора:
  - В общем случае:  $C_{B-1}^{N-1} = \frac{(N-1)!}{(B-1)! (N-B)!}$ , для бинарного разбиения:  $N - 1$  для всех ветвей от 2 до  $N$ :  $\sum_{b=2}^N C_{b-1}^{N-1} = 2^{N-1} - 1$ ,
  - Выбор вариантов – серединные точки

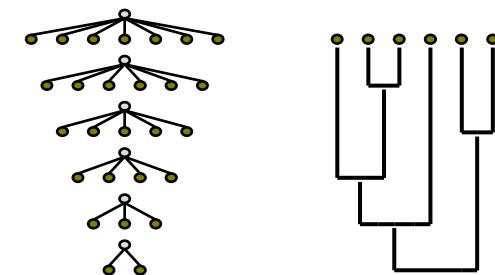


$$\begin{aligned} & 1-234 \quad \binom{3}{1} = 3 \\ & 12-34 \\ & 123-4 \\ \\ & 1-2-34 \quad \binom{3}{2} = 3 \\ & 1-23-4 \\ & 12-3-4 \\ \\ & 1-2-3-4 \quad \binom{3}{3} = 1 \end{aligned}$$

# Гипотезы-кандидаты для поиска разбиения категориального предиктора

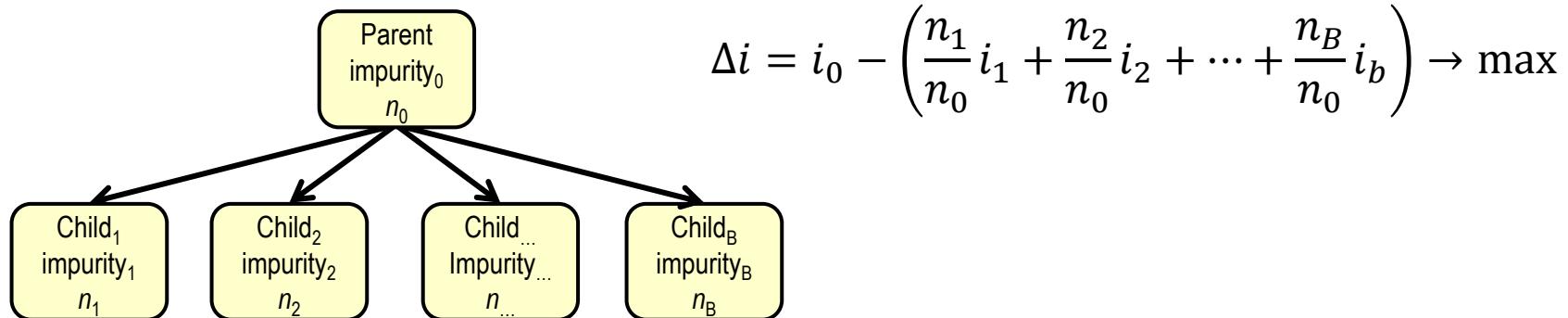
- Разбиваем множество категориальных значений предиктора из региона «родительского» узла  $S_m$ :
  - Ищем  $B$  ветвей:  $S_m = \bigcup_{b=1}^B S_{bm}$ :  $\forall i \neq j \Rightarrow S_{im} \cap S_{jm} = \emptyset$ , если  $|S_m| = N$ , то всего вариантов – число Стирлинга 2 порядка:
$$S(N, B) = B \cdot S(N - 1, B) + S(N - 1, B - 1)$$
  - Для бинарного дерева:  $2^{N-1} - 1$
- Сокращение числа гипотез:
  - Ограничение снизу на  $|S_m|$  или  $|S_{jm}|$
  - Эвристические, жадные алгоритмы
- Пример - иерархическая кластеризация гипотез (алгоритм Касса):
  - Строим  $N$  ветвей (каждая ветвь – значение)
  - Рассматриваем все варианты склейки двух
  - Выбираем лучшую склейку по критерию
  - Продолжаем, пока не «склеим» все в  $B$  ветвей

$N$	2	3	4	total
2	1			1
3	3	1		4
4	7	6	1	14
5	15	25	10	51
6	31	90	65	202
7	63	301	350	876
8	127	966	1701	4139
9	255	3025	7770	21146



# Критерии разбиения на основе однородности

- Сравнить гипотезы о разбиении (внутри одного предиктора):
  - на основе прироста однородности дочерних регионов по сравнению с родительским внутри :



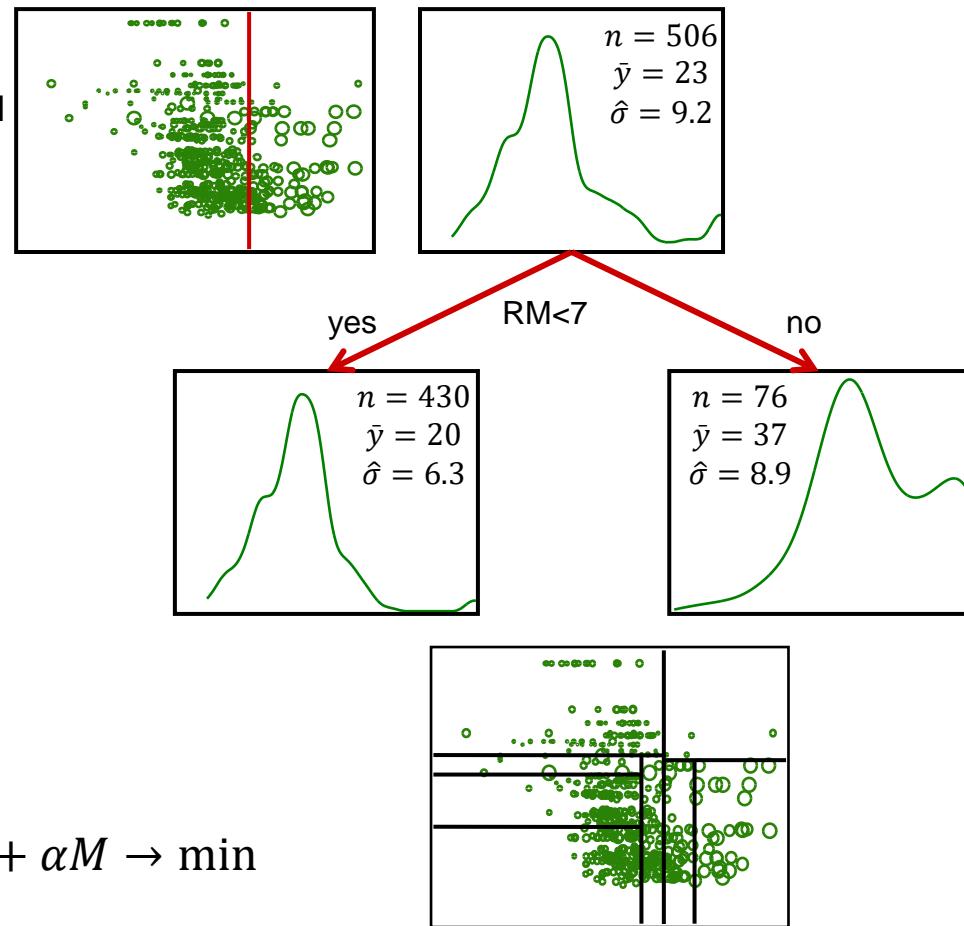
- Лучшие разбиения разных предикторов:
  - сравнивать либо по тому же критерию  $\Delta i(x_k) \vee \Delta i(x_j)$ , либо по нормированному  $\frac{\Delta i(x_j)}{i(x_j)} \vee \frac{\Delta i(x_k)}{i(x_k)}$

- Примеры критерия однородности:
  - Вариации (числовой отклик)
  - Энтропия, Джини, ошибка классификации (категориальный отклик)

# Критерии разбиения на основе уменьшения вариации

## ■ Для числового отклика:

- Выбираем гипотезу, для которой средняя взвешенная вариация (квадратичная ошибка) по дочерним узлам максимально уменьшается.
- В результате – «разбегаются» средние отклики по регионам и уменьшается дисперсия в них.
- Получаемый эмпирический риск квадратичной функции потерь и его можно регулировать (**по регионам**):



$$Q_\alpha(T) = \sum_{m=1}^M |R_m| \cdot Q_m + \alpha M \rightarrow \min$$

$$c_m = \frac{1}{|R_m|} \sum_{i:x_i \in R_m} y_i, Q_m = \frac{1}{|R_m|} \sum_{i:x_i \in R_m} (y_i - c_m)^2$$

# Критерии разбиения на основе уменьшения энтропии

- Мера неоднородности  $Q$  распределения классов в регионе  $R_m$ :
  - $p_{mk} = P(y = k | x \in R_m) = \frac{1}{|R_m|} \sum_{i:x_i \in R_m} (y_i = k)$
  - $Q$  максимальна в чистом регионе, т.е.  $\exists k: p_{mk} = 1, \forall k \neq k: p_{mk} = 0$
  - $Q$  минимальна, если классы равновероятны, т.е.  $\forall k: p_{mk} = 1/K$
  - Если отклик категориальный (не порядковый), то  $Q$  не зависит от порядка классов.
- Мера неоднородности выборки в регионе  $R_m$  на основе энтропии:
$$Q_{Entropy}(R_m) = H(Y|x \in R_m) = - \sum_{k=1}^K p_{mk} \log_2(p_{mk})$$
  - мера неопределенности (неоднородности) отклика  $Y$  в регионе  $R_m$
  - мат. ожидание (по классам) ф-ции потерь:  $L(p) = -\log_2(p)$
  - $-\log_2(p_{mk})$  KL-дивергенция для распределения с «чистым» классом  $k$  в регионе  $R_m$ ,  $(0, \dots, 1_k, \dots, 0)$ , насколько оно близко к  $p_{mk}$

# Information Gain (прирост информации)

- Энтропия в родительском узле - совместная:

$$H_p(y, x \in R_p) = - \sum_{k=1}^K P(Y = k, x \in R_p) \log_2 P(Y = k, x \in R_p)$$

- Энтропия в дочернем узле  $b$  – условная, неопределенность отклика, если знаем что  $x \in R_b$ :

$$H_b(y|x \in R_b) = - \sum_{k=1}^K P(Y = k|x \in R_b) \log_2 P(Y = k|x \in R_b)$$

- Ожидаемая условная энтропия по всем дочерним узлам  $1 \leq b \leq B$ , неопределенность при условии разбиения на  $R_p = R_1 \cup \dots \cup R_B$ :

$$H(y|R_1, \dots, R_B) = \sum_{b=1}^B P(x \in R_b) H_b(y|x \in R_b) = \sum_{b=1}^B \frac{|R_b|}{|R_p|} H_b(y|x \in R_b)$$

- Information Gain (как раз то, что мы максимизируем):

- уменьшение энтропии при заданном разбиении:

$$IG(y|R_p = R_1 \cup \dots \cup R_B) = H_p(y, x \in R_p) - H(y|R_1, \dots, R_B)$$

# Критерии разбиения на основе индекса Джини

## ■ Интерпретации индекса Gini:

- Изначально в экономике - оценка неравенства населения по доходам
- Модельный пример – вероятность вытащить (с возвратом) из закрытой корзины с цветными шарами два шара разного цвета:

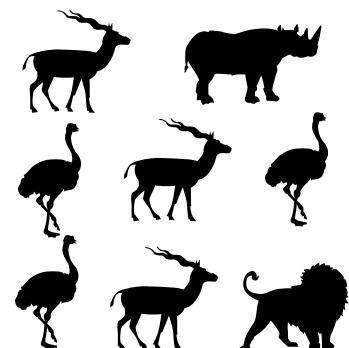
$$Gini(R_m) = 1 - \sum_{k=1}^K p_{mk}^2 = \sum_{j < k} 2p_{mk}p_{mj}$$

- Аналогично энтропии - мера неоднородности выборки в регионе и мат. ожидание (по классам) убывающей ф-ции потерь:  $L(p) = (1 - p)$ :

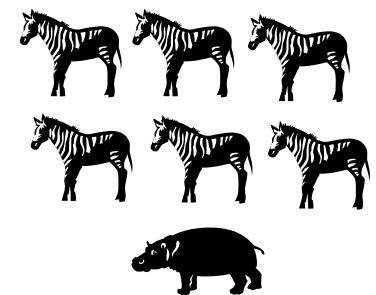
$$Q_{Gini}(R_m) = \sum_{k=1}^K p_{mk}(1 - p_{mk})$$

- Интересный факт: если мера  $Y = \{0,1\}$ , то индекс Джини совпадает с вариацией

$$Gini = 1 - 2\left(\frac{3}{8}\right)^2 - 2\left(\frac{1}{8}\right)^2 = 0.69$$



$$Gini = 1 - \left(\frac{6}{7}\right)^2 - \left(\frac{1}{7}\right)^2 = 0.24$$



# Оценка важности переменных

- Варианты оценки важности переменных (всегда на выборке):
  - Вариант 1: по каждому предиктору  $x_i$  суммирование прироста меры однородности  $\text{Gain}(x_i) = \sum_{\text{node}:x_i \in \text{node}} \Delta Q_{\text{node}}(x_i)$  (Джини или Энтропии для категориального отклика или вариации для числового) по всем вхождениям переменной в дерево, т.е. по всем внутренним узлам с условиями на переменную  $x_i$
  - Вариант 2: считаем качество полной модели (дерева) и модели, где все  $x_i = \text{missing}$ , сравниваем ухудшение:

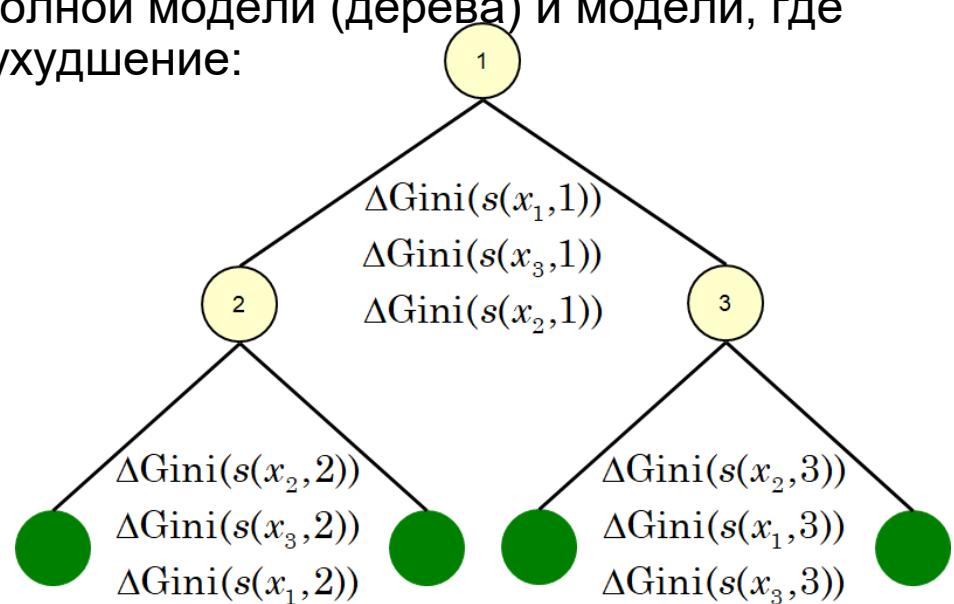
$$\text{Gain}(x_i) = \frac{Q(T) - Q(T|x_i=\text{missing})}{Q(T)}$$

- Нормировка:

$$\text{Importance}(x_i) = \frac{\text{Gain}(x_i)}{\max_j [\text{Gain}(x_j)]}$$

или

$$\text{Importance}(x_i) = \frac{\text{Gain}(x_i)}{\sum_j \text{Gain}(x_j)}$$



# Пример дерева

```
from sklearn.tree import DecisionTreeClassifier, DecisionTreeRegressor, plot_tree
```

```
from sklearn.datasets import load_iris
```

```
from sklearn.inspection import DecisionBoundaryDisplay
```

```
iris = load_iris()
```

```
X, y = iris.data, iris.target
```

```
X = X[:, :2]
```

```
X.shape, y.shape, iris.target_names
```

```
((150, 2), (150,), array(['setosa', 'versicolor', 'virginica'], dtype='|<U10'))
```

```
tree = DecisionTreeClassifier(criterion="gini", max_depth=5, min_samples_split=5, min_samples_leaf=3,  
                               ccp_alpha=0.0) # pruning parameter
```

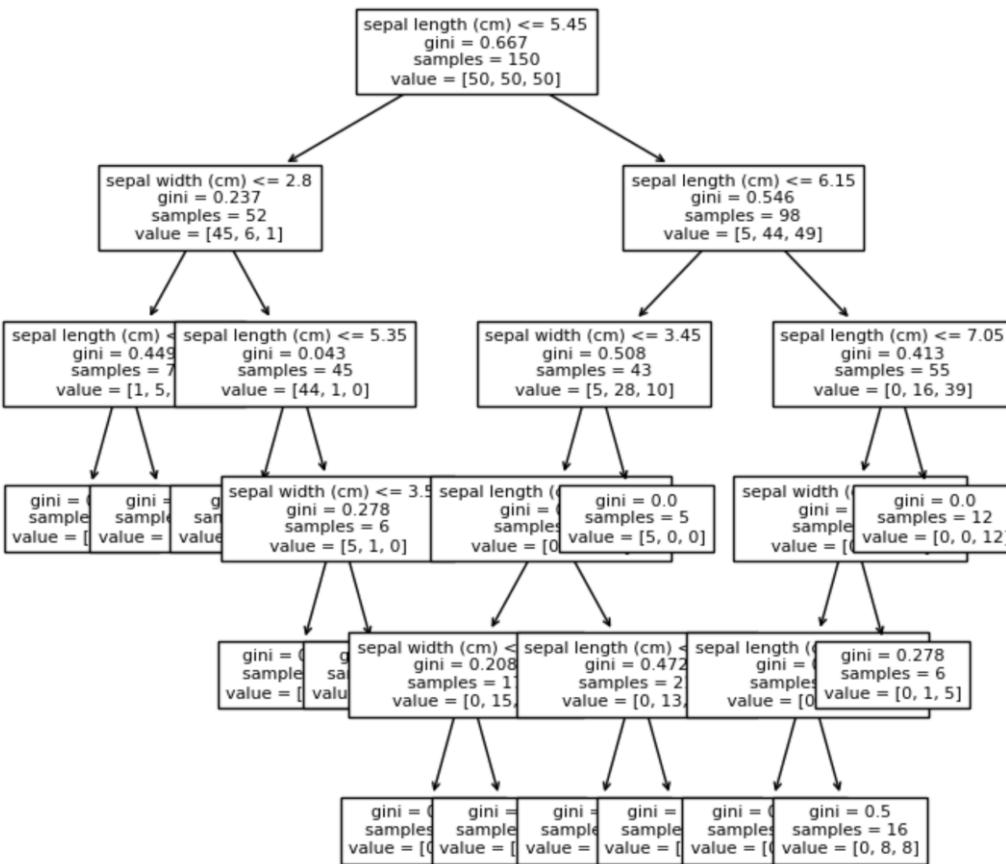
```
tree.fit(X, y)
```

```
DecisionTreeClassifier
```

```
DecisionTreeClassifier(max_depth=5, min_samples_leaf=3, min_samples_split=5)
```

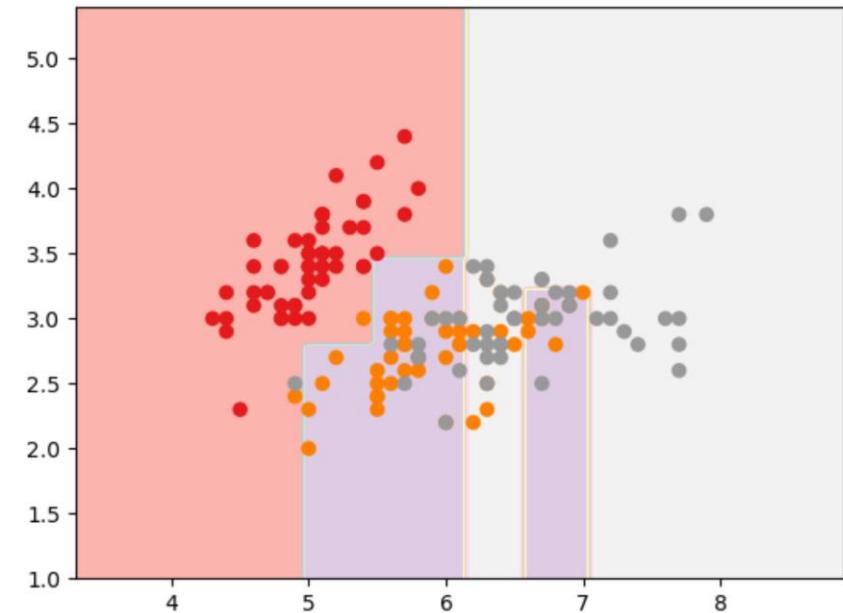
# Пример дерева

```
plot_tree(tree, fontsize=8, feature_names=iris.feature_names)  
plt.gcf().set_size_inches(8, 8)
```



```
DecisionBoundaryDisplay.from_estimator(tree, X, cmap="Pastel1")  
plt.scatter(*X.T, c=y, cmap="Set1")
```

```
<matplotlib.collections.PathCollection at 0x7fa63f56b1c0>
```



```
tree.feature_importances_
```

```
array([0.76047482, 0.23952518])
```

# Статистические критерии разбиения

- Недостатки критериев на основе оценки однородности выборки:
  - При сравнении предикторов с разной мощностью (число различных значений) тяготеют к выбору более мощного варианта
  - При сравнении вариантов разбиения с разным числом ветвей (больше 2) тяготеют к выбору большего числа ветвей
  - В общем случае не позволяют разумно задать порог на остановку роста (например, на минимально допустимое улучшение)
- Идея статистических критериев:
  - Оценивать как меняются распределения отклика в дочерних узлах по сравнению с родительским, чем больше меняются, тем лучше
  - Оценивать по p-value базовую гипотезу  $H_0$  о том, что распределение не изменилось, чем меньше p-value, тем более мы уверены, что разбиение полезно
  - Сравнивать гипотезы о разбиении по  $\text{logworth} = -\log_{10}(p_\alpha)$
  - Использовать порог для p-value для отбора гипотез и остановки роста
  - Использовать корректировку Бонферрони для множественного сравнения гипотез

# Критерий Фишера для числового отклика

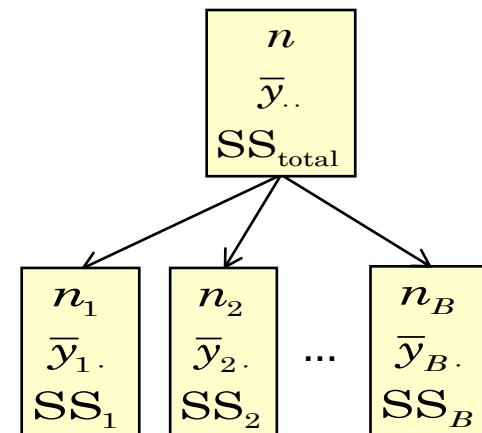
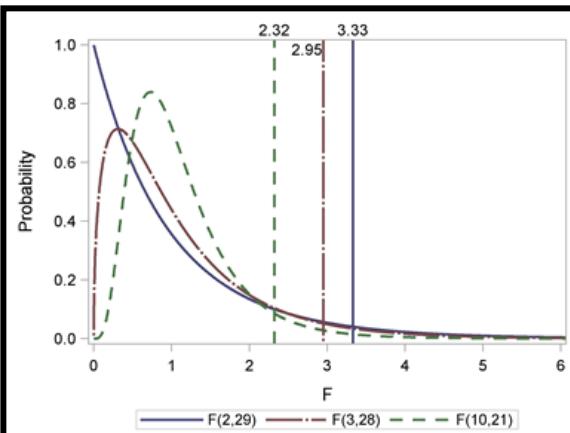
## ■ Идея из дисперсионного анализа:

- Гипотеза  $H_0$  - все групповые средние в  $B$  ветвях совпадают
- Считается статистика Фишера:

$$F = \left( \frac{SS_{model}}{SS_{error}} \right) \left( \frac{N-B}{B-1} \right) \sim F_{B-1, N-B}, \text{ где } SS_{total} = \sum_{i=1}^N (y_i - \bar{y})^2,$$

$$SS_{error} = \sum_{b=1}^B \sum_{i:x_i \in R_b} (y_i - \bar{y}_b)^2, \quad SS_{model} = SS_{total} - SS_{error}$$

- По распределению Фишера со степенями свободы  $B - 1$  и  $N - B$  находится p-value (уровень значимости) гипотезы  $H_0$ , чем он меньше, тем увереннее мы отклоняем  $H_0$



# Критерий $\chi^2$ для категориального отклика

## ■ Идея из анализа таблиц частот:

- Строим матрицу сопряженности для заданного разбиения (строки – ветви, столбцы – классы, ячейки – сколько наблюдений класса попало в соответствующую ветвь)
- Гипотеза  $H_0$  - распределение классов в  $B$  ветвях одинаковое и совпадает с родительским, считается статистика:

$$\chi^2 = \sum_{k=1}^K \sum_{b=1}^B \frac{(O_{bk} - E_{bk})^2}{E_{bk}} \sim \chi^2_{(B-1)(K-1)}, \text{ где}$$

$O_{bk}$  - сколько наблюдений из класса  $k$  попало в ветвь  $b$

$E_{bk} = P_k |R_b|$  - сколько бы попало, если  $H_0$  верна

- По распределению  $\chi^2_v$  со степенями свободы  $v = (B - 1)(K - 1)$  находится p-value (уровень значимости) гипотезы  $H_0$ , чем меньше тем лучше

Матрица  $O$

	<38.5	$\geq 38.5$	
1	293	71	.342
7	363	1	.342
9	42	294	.316
	.656	.344	$n=1064$

Матрица  $E$

239	125
239	125
225	116

Матрица  $\chi^2$

12	23
64	123
149	273

# Корректировка Бонферрони

- Корректируется p-value с учетом множественного сравнения гипотез:
  - Для серии  $m$  сравнений нескольких гипотез, каждая с уровнем значимости  $\alpha$ , уровень значимости всей серии  $\alpha_m \leq 1 - (1 - \alpha)^m$
  - Корректировка Бонферрони – домножаем уровень значимости  $\alpha$  на число сравнений  $m$ , что тоже самое, домножаем p-value на  $m$
  - Скорректированный на  $m$  сравнений критерий разбиения logworth:  $\text{logworth}_m(p_\alpha) = -\log(mp_\alpha) = -\log(p_\alpha) - \log(m) = \text{logworth}(p_\alpha) - \log(m)$
  - $\log(m)$  - штраф за мощность предиктора и/или число ветвей

X: 38.5		
1	293	71
7	363	1
9	42	294

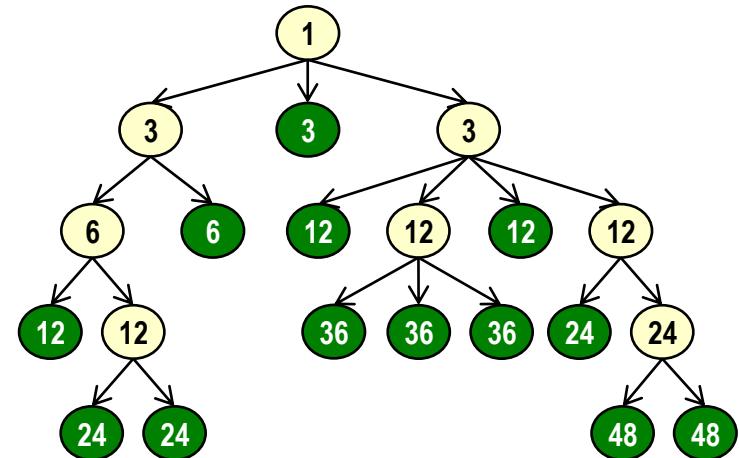
X: 17.5 36.5		
1	249	42
7	338	25
9	26	16
294		

$\chi^2_v$	$v$	$-\log_{10}(P)$	$m$	$-\log_{10}(mP)$
644	2	140	96	138
660	4	141	4560	137

# Множитель глубины

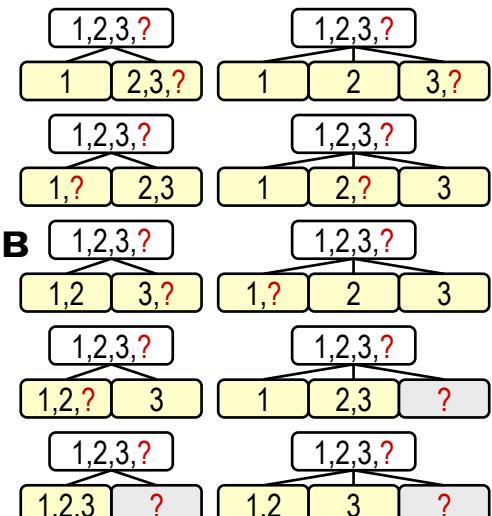
- В теории разбиение на глубине  $d$  также зависит от предыдущих разбиений, поэтому p-value можно корректировать по Бонферрони с учетом глубины и числа ветвей на уровнях выше

	$-\log_{10}(P)$	$m$	$-\log_{10}(mP)$	$d$	$-\log_{10}(2^d mP)$
	26.7	53	24.9	0	24.9
	3.12	14	1.97	1	1.67
	1.63	39	.039	1	-.26
	2.40	11	1.36	2	.76

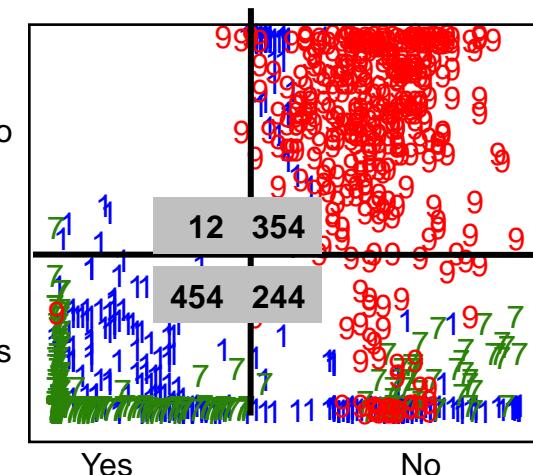


# Пропущенные значения

- Важное преимущество деревьев решений:
  - Могут работать с пропусками без подстановки
- Основные подходы:
  - Строим гипотезы о разбиении **без учета пропусков**
  - **Направляем пропуски по отдельной ветке** (если дерево не бинарное), по самой **большой** ветке, по самой **точной** ветке, по **всем веткам** одновременно пропорционально их размеру
  - **Расширяем множество гипотез** разбиения проверкой: что будет, если запустить пропуски по каждой ветке  $b$  (пример справа вверху)
  - **Суррогатные правила** (пример справа внизу): No для каждого лучшего разбиения по предиктору  $x_i$ , находим разбиение по  $x_{j \neq i}$ , максимально согласованное (максимальное пересечения регионов дочерних узлов) с исходным. Обычно строят несколько дублирующих правил

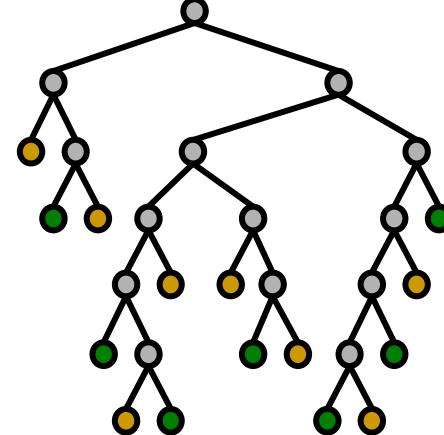


Уровень согласия=76%

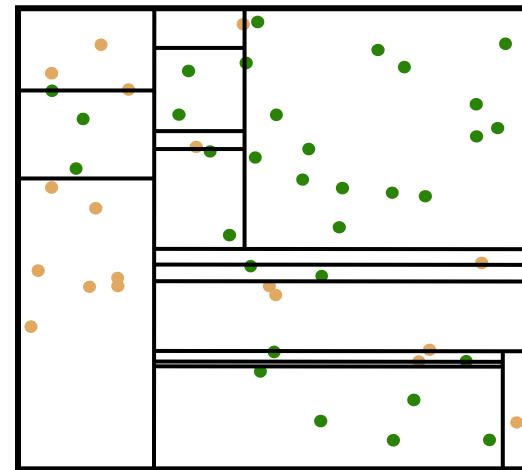


# Переобучение и сложность деревьев решений

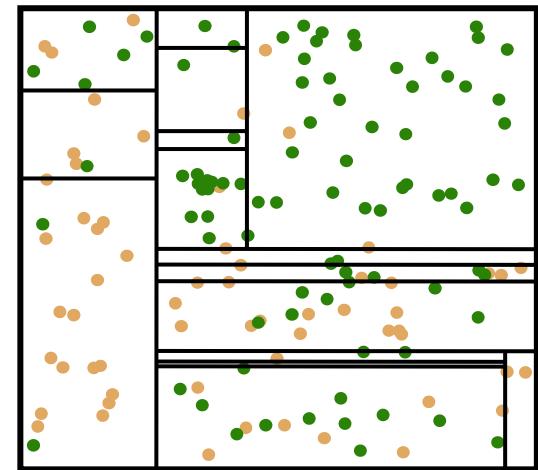
Максимальное дерево  
часто переобучено



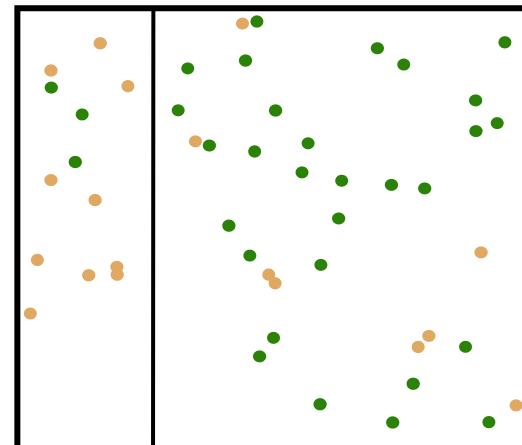
Тренировочный набор



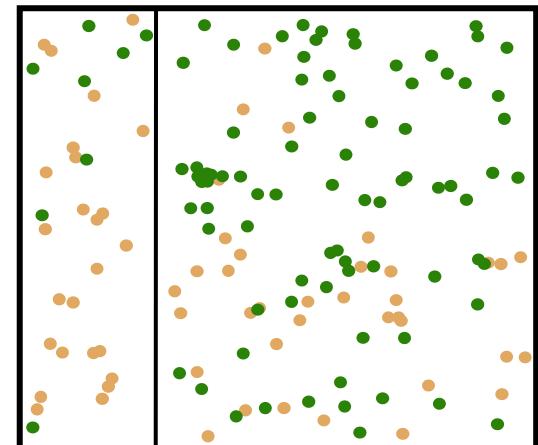
Новые данные



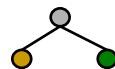
Тренировочный набор



Новые данные



Небольшое дерево  
часто недообучено

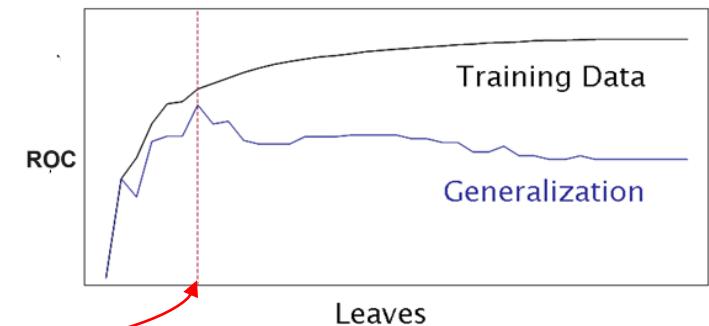
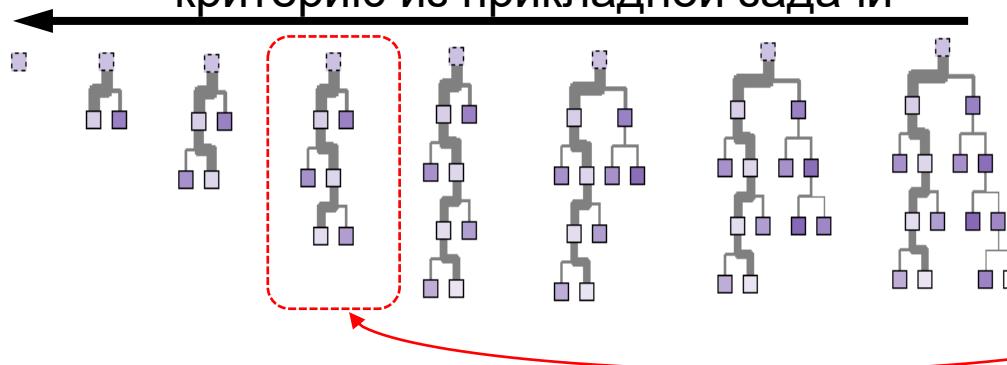


# Контроль сложности деревьев решений

- Сложность дерева:
  - Обычно оценивают по числу листьев
  - Как и у других моделей – рост сложности влечет рост дисперсии и уменьшение смещения, и наоборот.
  - Сложность можно контролировать: ограничением роста (pre-pruning) или упрощением максимального дерева – обрубанием ветвей (pruning)
- Параметры ограничение роста:
  - Максимально допустимая глубина дерева
  - Минимально допустимое число наблюдений в листе
  - Максимально допустимое число ветвей
  - Минимально допустимое число различных значений в предикторе для формирования по нему гипотез о разбиении
  - Порог останова на p-value или другой нормированный критерий
  - Корректировка порогов отсечения с учетом глубины или числа ветвей
- Обрубание ветвей - дальше

# Обрубание дерева

- Процедура обрубания ветвей (или удаление слабых связей):
  - построение большого дерева  $T_0$ , а затем выполнение *отсечения* для получения *поддерева* для *сокращения сложности*
- Простой подход – с использованием валидационного набора
  - Строим максимальное дерево и последовательно проверяем варианты обрубания листьев (из одного общего родителя) с оценкой качества поддерева на валидационной выборке
  - Получаем семейство поддеревьев, выбираем лучшее
  - Важно: **критерий обрубания может не совпадать с критерием роста**, например, строим дерево по IG, а упрощаем по ROC или критерию из прикладной задачи



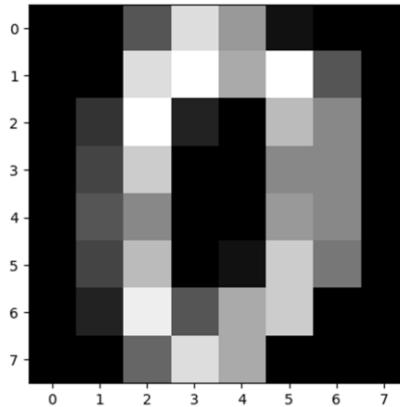
# Сложность дерева - пример

```
from sklearn.metrics import accuracy_score
from sklearn.datasets import load_digits
```

```
digits = load_digits()
X, y = digits.data, digits.target
X.shape, np.unique(y)
```

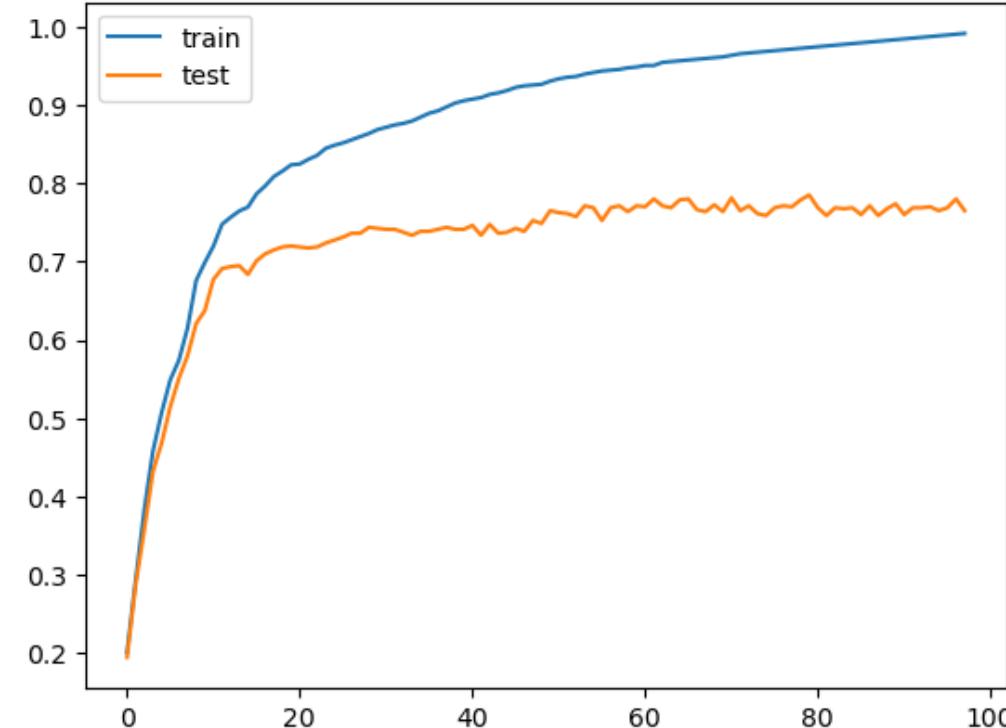
```
((1797, 64), array([0, 1, 2, 3, 4, 5, 6, 7, 8, 9]))
```

```
plt.imshow(X[0].reshape(8, 8), cmap="gray");
```



```
N = 1000 # Labels are scattered evenly enough
train_X, train_y = X[:N], y[:N]
test_X, test_y = X[N:], y[N:]
```

```
result = []
for max_leaf_nodes in range(2, 100):
    tree = DecisionTreeClassifier(max_leaf_nodes=max_leaf_nodes)
    tree.fit(train_X, train_y)
    score = {"train":accuracy_score(train_y, tree.predict(train_X)),
             "test":accuracy_score(test_y, tree.predict(test_X))}
    result.append(score)
pd.DataFrame(data=result).plot();
```



# Обрубание дерева с регуляризацией (cost-complexity/MDL)

## ■ Регуляризованный эмпирический риск:

$$Q_\alpha(T) = \sum_{m \in T} |T_m| \cdot Q_m(T) + \alpha |T|,$$

$Q_m$  - оценка неоднородности в листе  $m$ ,  $|T|$  - сложность дерева  $T$  (обычно, число листьев), мощность листа  $|T_m|$  - число наблюдений

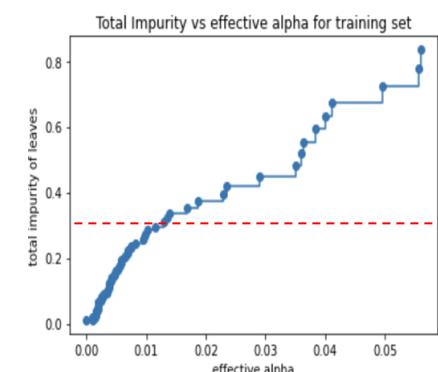
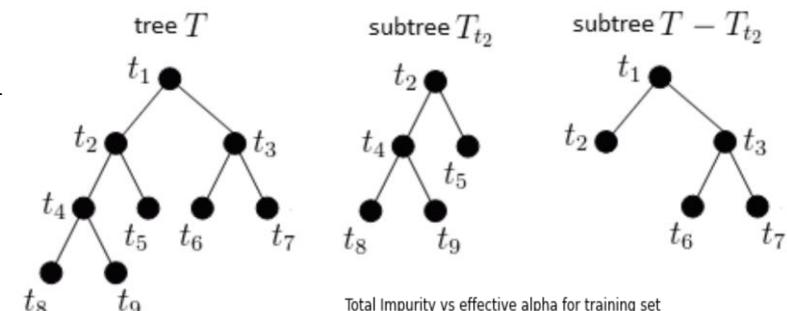
- Что обрубить? Если  $Q_\alpha(t) - Q_\alpha(T_t) \rightarrow 0$ , то  $\alpha_{eff}(t) = \frac{Q(t) - Q(T_t)}{|T_t| - 1}$ ,  $t$  – узел,  $T_t$  - его поддерево,  $\alpha_{eff}(t)$  – его параметр регуляризации

## ■ Процедура обрубания

- Инициализация  $T^{(1)} = T, \alpha_1 = 0, i = 1$
- Повторять: выбрать  $\min_{t \in T^{(i)}} \alpha_{eff}(t)$   
 $\alpha_{i+1} = \alpha_{eff}(t), T^{(i+1)} = T^{(i)} - T_t^{(i)}$

## ■ Результат:

- $0 = \alpha_1 \leq \alpha_2 \leq \dots \leq \alpha_k \Leftrightarrow$   
 $T = T^{(1)} \subset T^{(2)} \subset \dots \subset T^{(k)} = \{\text{root}\}$
- Можно построить «трассу» зависимости однородности  $Q_\alpha$  от  $\alpha_{eff}$  и подобрать порог кросс-валидацией или на тестовой выборке

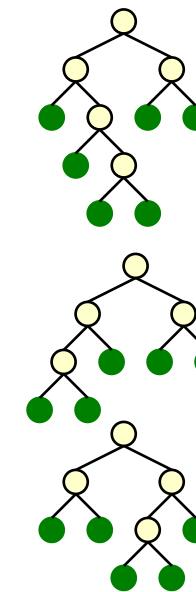


# Деревья решений как инструмент предобработки данных

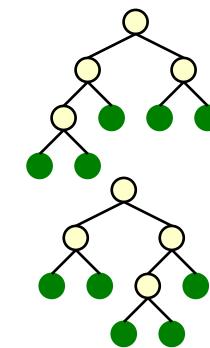
## ■ Подстановка пропусков

- на основе оценок  $x_i = F_{tree}(x_1, \dots, x_{i-1}, x_{i+1}, \dots)$
- много достоинств: любой отклик, работа с числовыми, категориальными и пропущенными значениями других признаков, произвольные зависимости, сохранение распределений
- недостатки: нестабильность и невысокая точность (но они тут не важны)

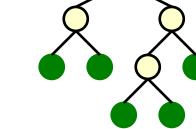
$x_1$	$x_2$	$x_3$
8.6	14	?
?	43	1.4
6.3	22	2.7
3.8	?	?
1.4	19	1.1
4.6	63	1.0
5.5	26	2.3
?	?	?
1.7	82	2.8
6.8	23	1.8
5.8	30	1.2



$$y = x_1 \\ \mathbf{x} = (x_2, x_3)$$



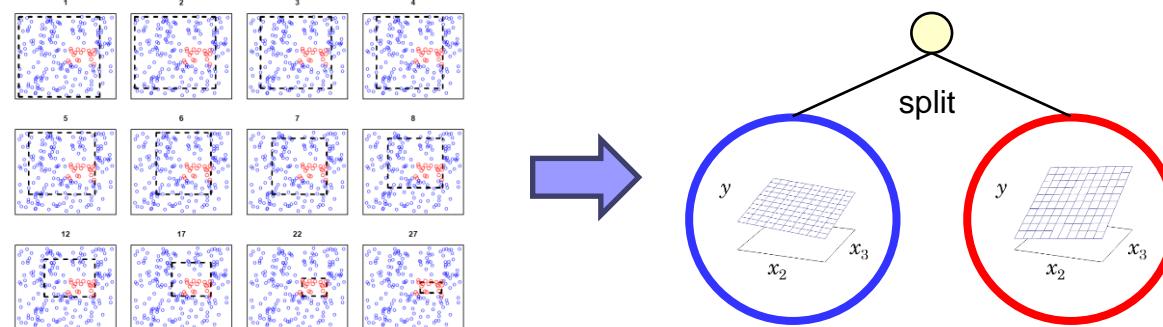
$$y = x_2 \\ \mathbf{x} = (x_1, x_3)$$



$$y = x_3 \\ \mathbf{x} = (x_1, x_2)$$

# Деревья решений как инструмент предобработки данных

- Выделение «чистых» регионов:
  - для задач классификации с большим дисбалансом классов (PNrule), пример алгоритма:
    1. Р-фаза: строим дерево решений, находим самый большой и «чистый» лист (пропорция целевого класса и размер выше заданных порогов), удаляем наблюдения найденного листа из выборки, повторяем Р-фазу
    2. N-фаза: «очищенный» набор не такой дисбалансный, имеет смешанные области со сложными границами, в них строим гибкие точные модели (например, ансамбли или нейросети)
  - для «негладких» регрессий (PRIM, bump hunting)



# Деревья решений как инструмент предобработки данных

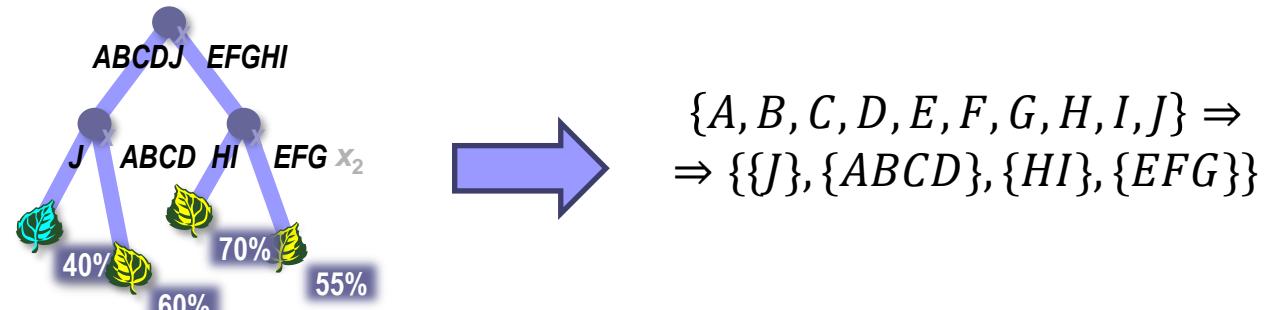
- Отбор значимых признаков:
  - Не нужно дополнительно предобращивать – деревья работают с пропусками, числовыми и категориальным признаками
  - Не важно насколько сложная нелинейная зависимость



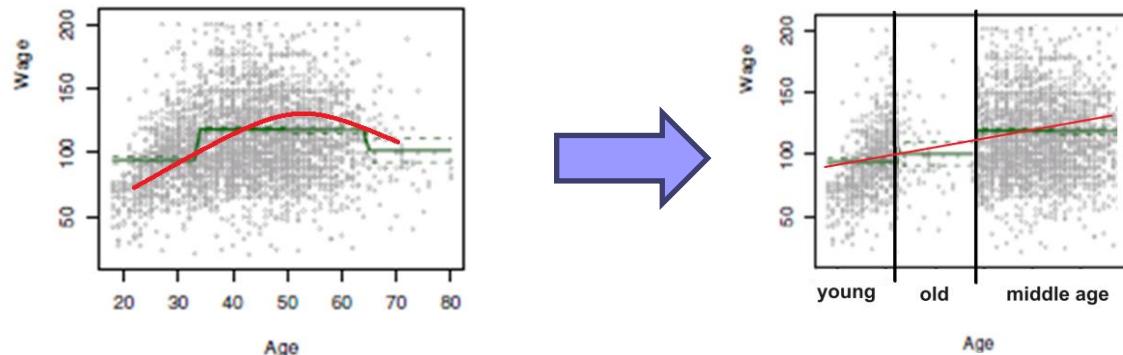
# Деревья решений как инструмент предобработки данных

## ■ Преобразование предикторов с учетом отклика:

- строится одномерное дерево (с одним входом)
- значения, попавшие в листья формируют подмножества группировки (для категориальных)

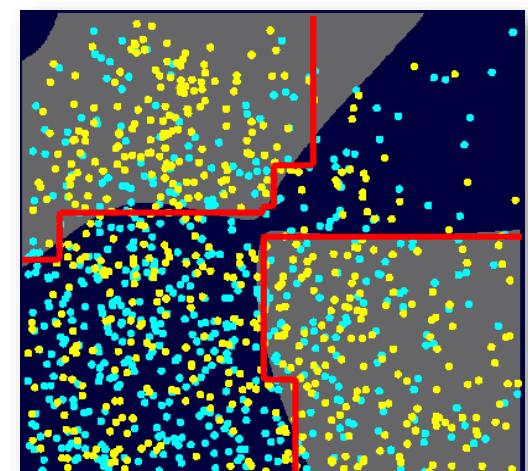
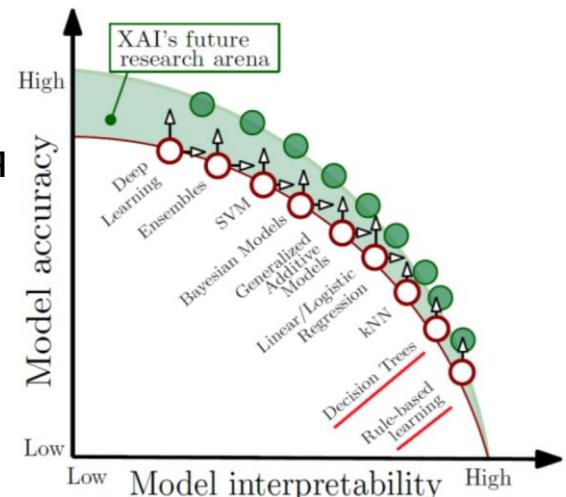


- отрезки дискретизации (для числовых предикторов)



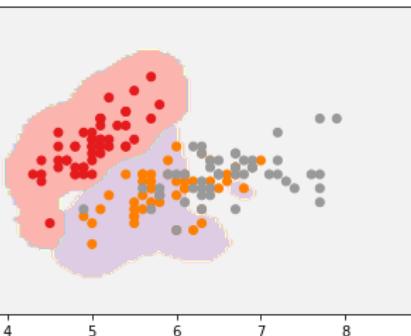
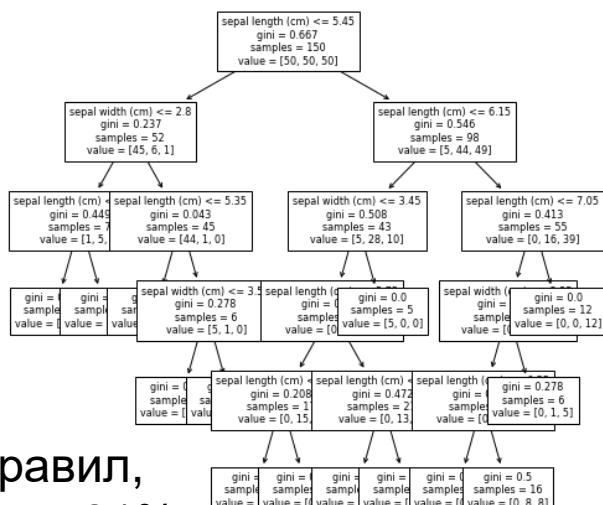
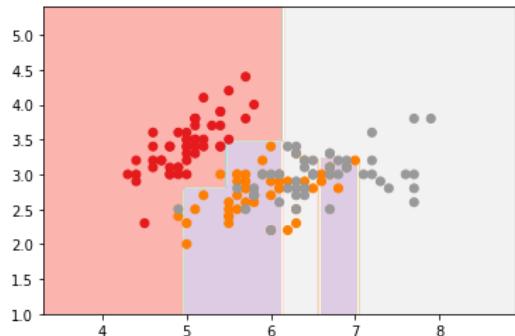
# Деревья решений как инструмент интерпретации сложных моделей

- Explainable AI:
  - Чем модель более сложная, тем более точная и менее понятная человеку (менее интерпретируемая)
  - Как получить описание не интерпретируемой модели?
- Суррогатные модели:
  - Строится сложная не интерпретируемая модель (например, нейросеть, сплайны или ансамбль)
  - На ее прогнозах (а не на реальных откликах) строится суррогатное «объясняющее» дерево (можно оценить уровень его согласованности или аппроксимации исходной модели)
  - На реальных откликах такое дерево не построить

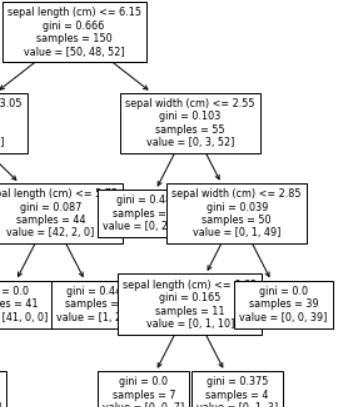
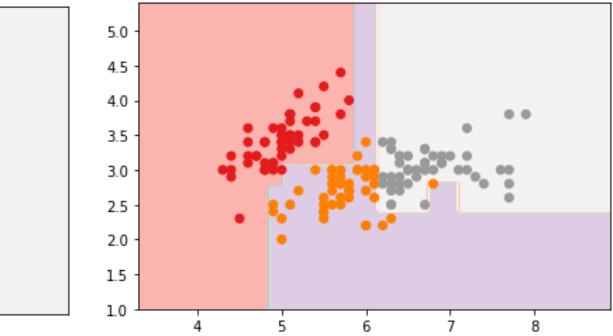


# Пример суррогатной модели

- Дерево на исходных данных, SVM с RBF, Суррогатное дерево



точность 84%



13 правил,  
точность 81%

8 правил,  
точность 84%

# Особенности классических алгоритмов построения деревьев решений

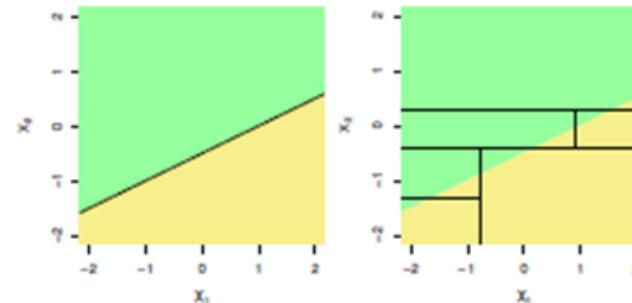
Свойства	CHAID (Kass)	CART (Breiman)	C4.5 (Quinlan)
Критерий для числ. отклика	Фишер	Вариация	нет
Критерий для кат. отклика	Хи-квадрат	Джини	Энтропия
Число ветвей	Больше или равно двум	Всегда две	Больше или равно двум
Работа с пропусками	Отдельная ветвь или перебор гипотез	Подстановка или суррогатные правила	Пропорция по веткам или подстановка
Особенности	Корректировка Бонферрони и глубина	Линейные комбинации при разбиении	Алгоритм логического «сокращения»
Обрубание вервей	Нет или по валидации	Cost-complexity	На основе ошибок

# Преимущества деревьев решений

- Деревья решений имеют самую высокую интерпретируемость, считается, что они отражают процесс принятия решений людьми
- Деревья могут обрабатывать разные типы входных переменных и откликов, пропуски, относительно не чувствительны к выбросам в признаках (но чувствительны в отклике)
- Деревья не делают предположений о виде и сложности зависимости
- Есть эффективные инструменты борьбы с переобучением
- Легко адаптируются к разным задачам машинного обучения
- Быстро обучаются и применяются
- Инструмент подготовки данных

# Недостатки деревьев решений

- Невысокое качество модели (особенно на гладких зависимостях, где растет сложность):



- Нестабильность модели (жадный алгоритм):

