Построение деревьев решений

ФИНАНСОВЫЙ УНИВЕРСИТЕТ
ПРИ ПРАВИТЕЛЬСТВЕ РОССИЙСКОЙ ФЕДЕРАЦИИ

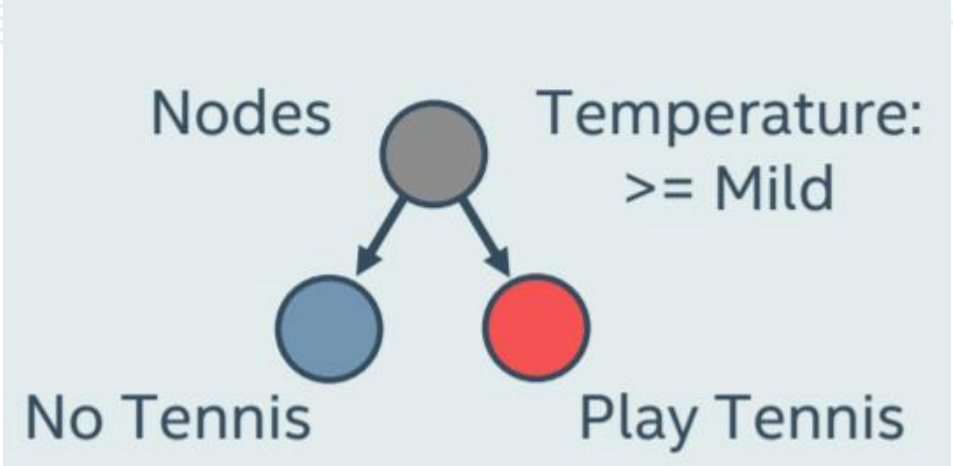
Введение в деревья решений

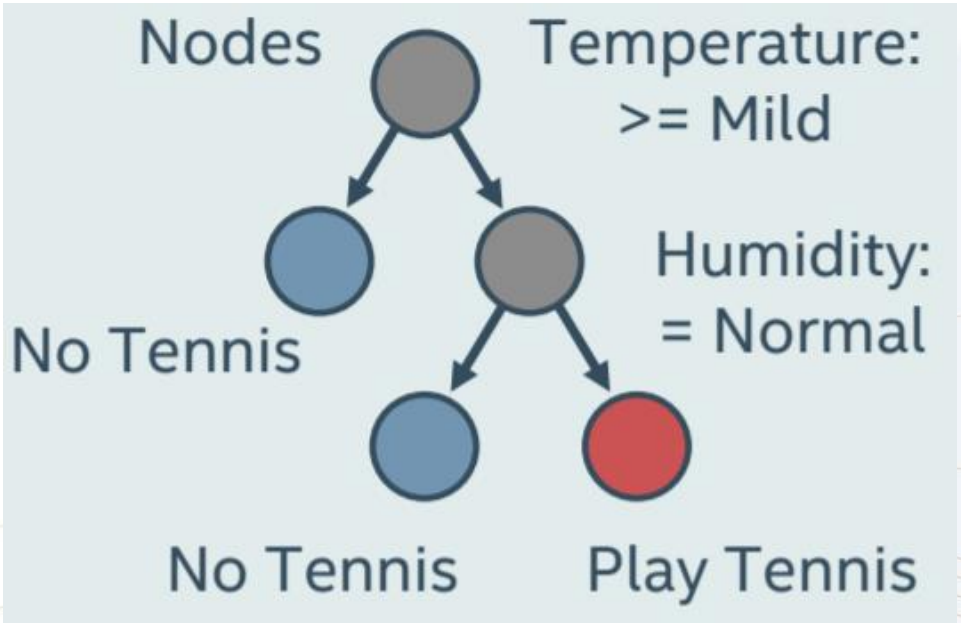
Day	Outlook	Temperature	Humidity	Wind	PlayTennis
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No

Давайте предскажем, стоит ли играть в теннис исходя из состояние атмосферы, температуры, влажности, ветра

Построение дерева решений

Разобьем данные на основе признаков, чтобы предсказать результат

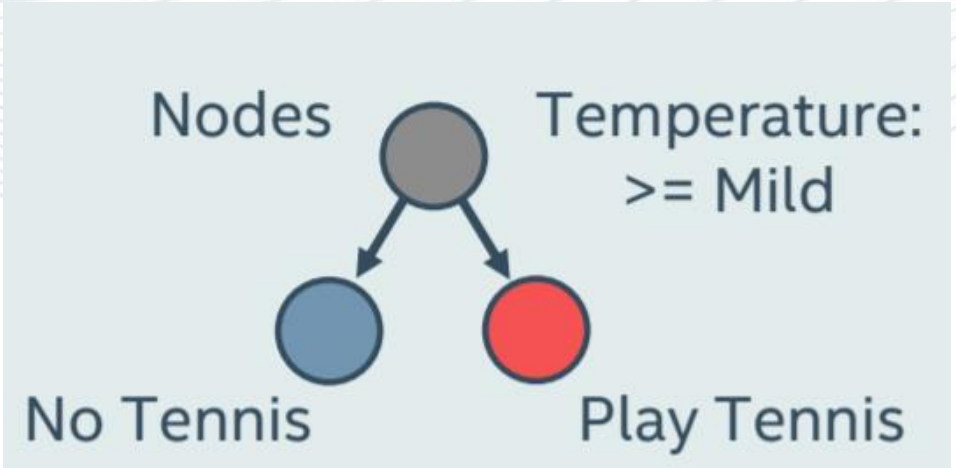


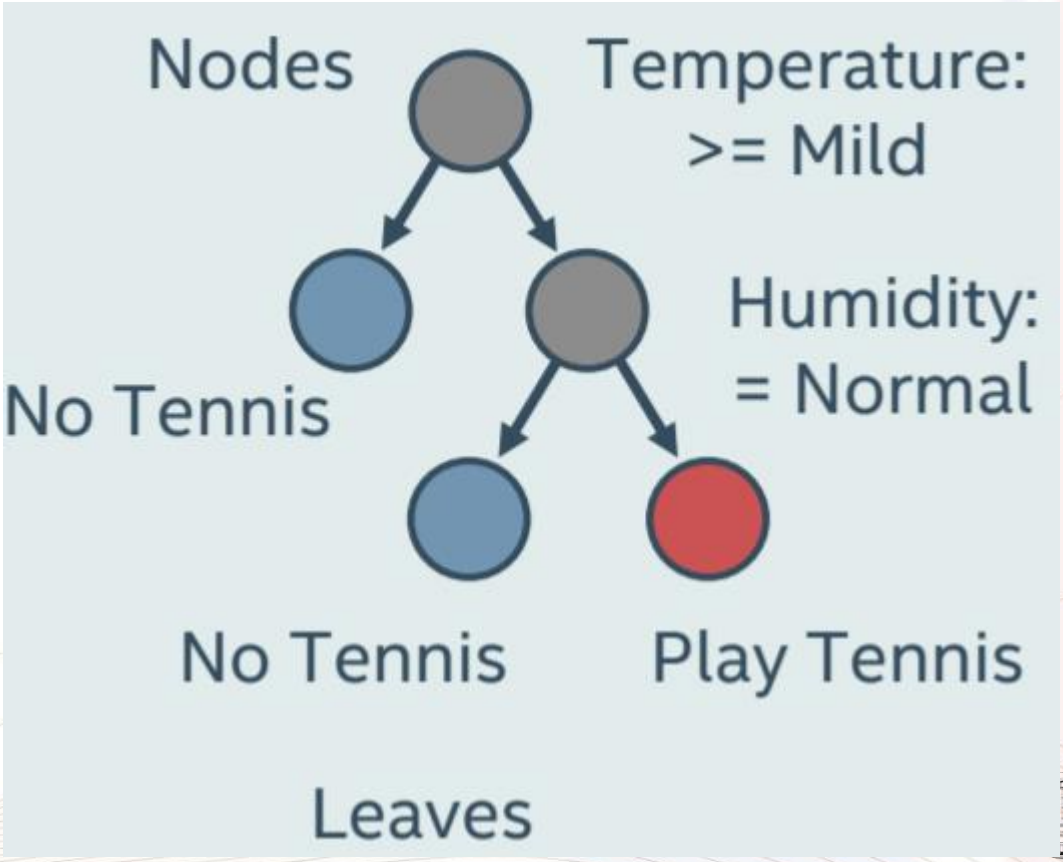


Построение дерева решений

Разделим данные на основе признаков, чтобы предсказать результат

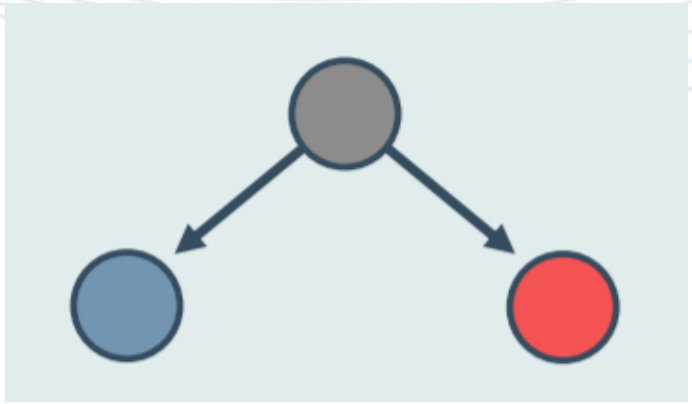
Деревья, которые предсказывают категориальные результаты, являются деревьями решений



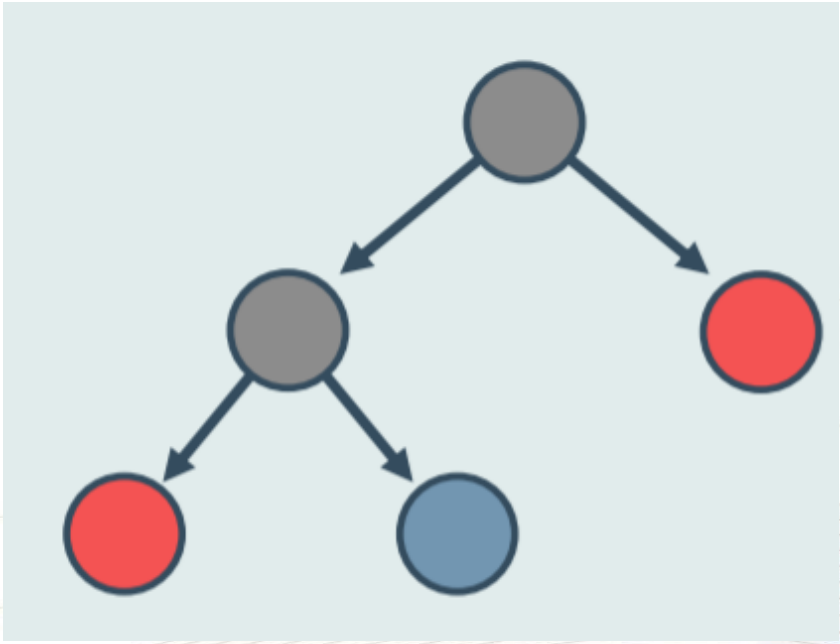


Построение дерева решений

Выберите объект и разбейте данные на двоичное дерево



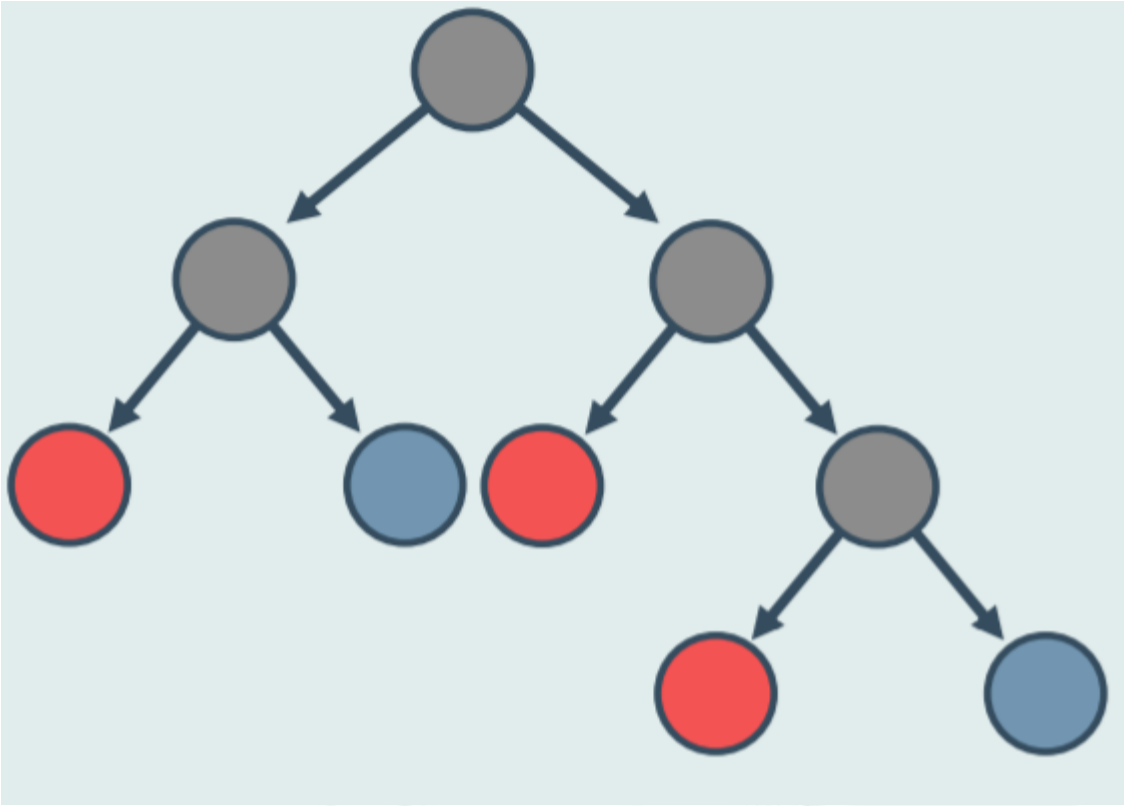
Продолжить разбиение с помощью доступных функций (признаков)



Построение дерева решений

Продолжаем до тех пор пока (критерий останова):

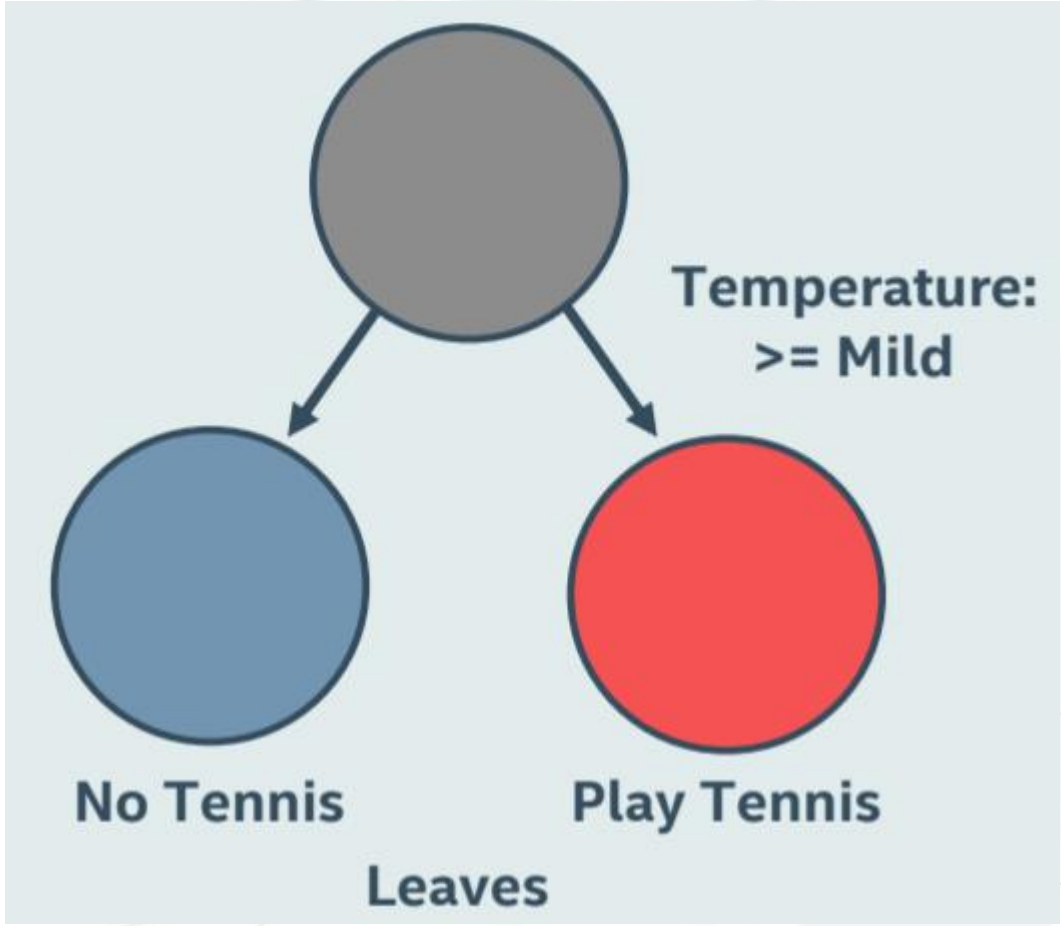
- Листовые узлы являются чистыми - остается только один класс
- Достигнута максимальная глубина
- Достигнут показатель качества



Построение дерева решений

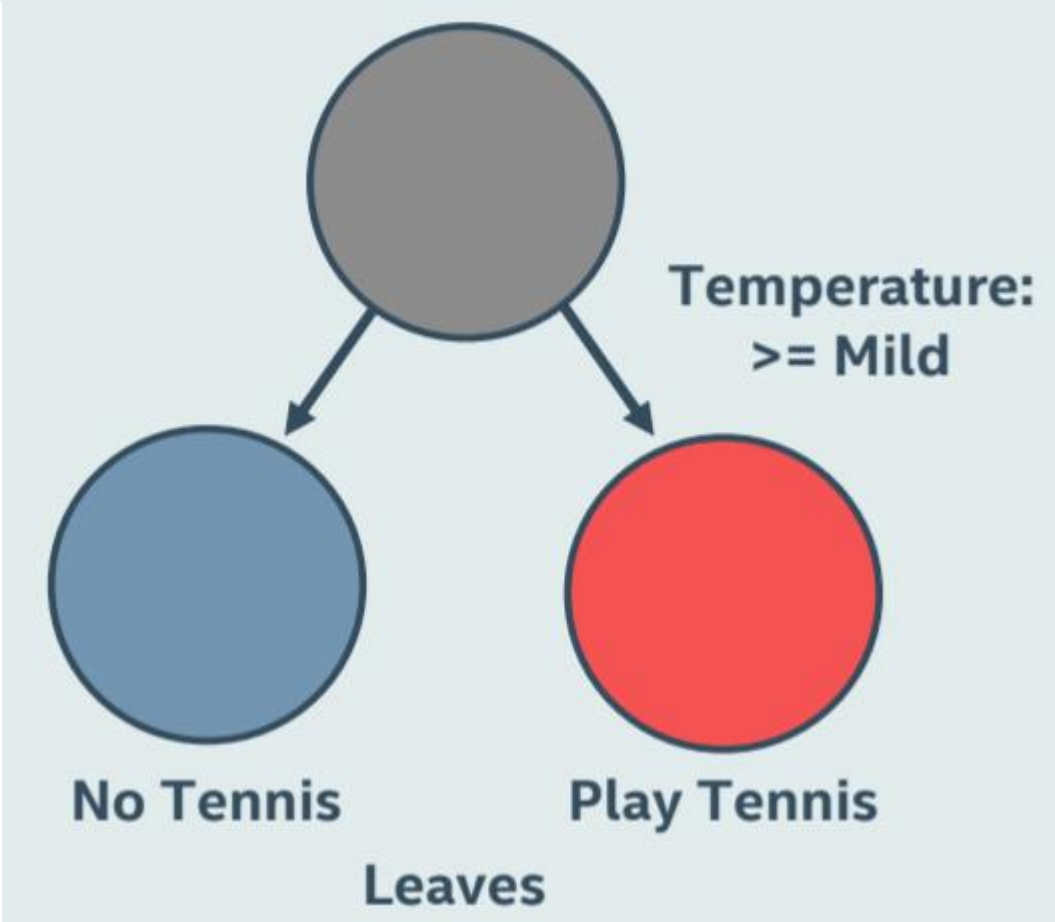
Будем использовать *жадный* поиск: найдем лучшее разбиение на каждом шаге

«Жадными» называются алгоритмы, которые на каждом шаге делают локально оптимальный выбор, допуская, что итоговое решение также окажется оптимальным.



Построение дерева решений

- Как определить лучший сплит (разбиение)?
- Тот, который максимизирует информацию, полученную в результате разбиения
- Как рассчитать количество полученной информации?



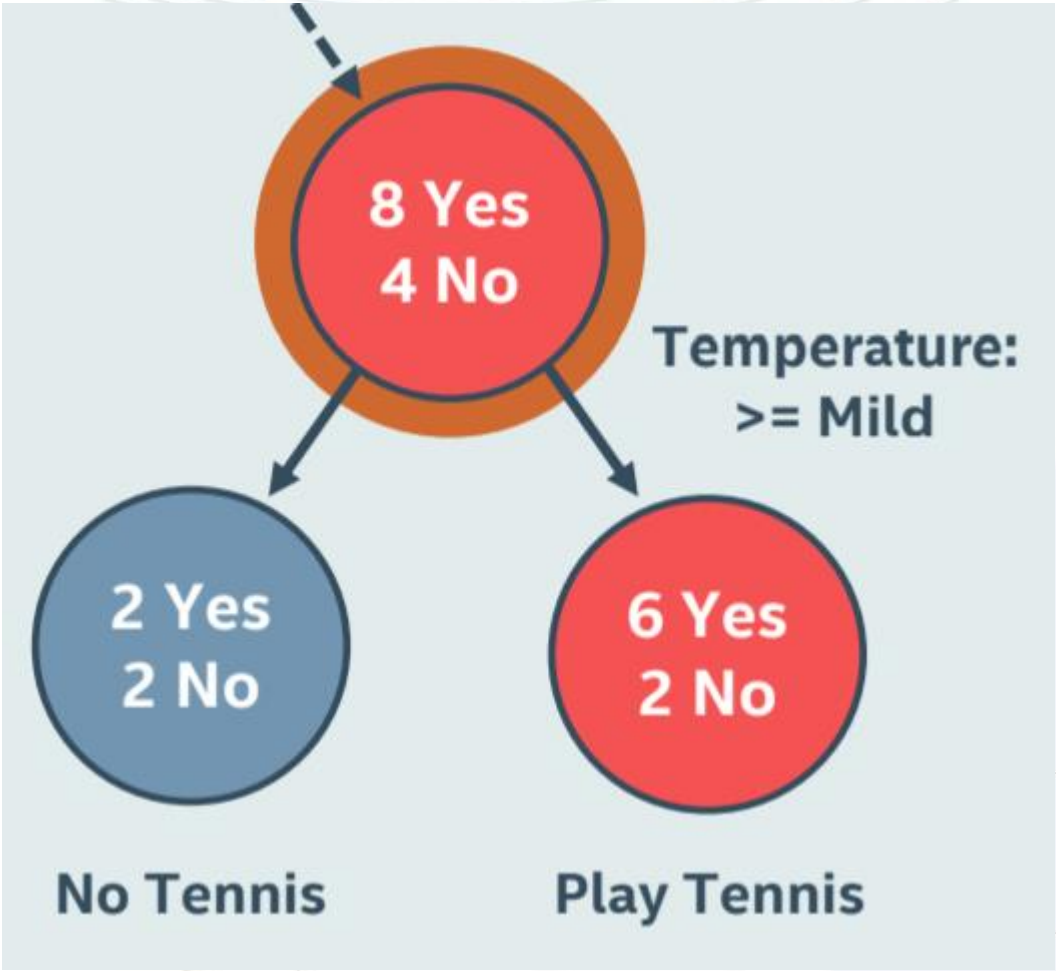
Разбиение на основе энтропии

Уравнение энтропии

$$H(t) = -\sum_{i=1}^{n} p(i|t) log_2[p(i|t)]$$

Энтропия до разбиения

$$-{}^{8}/_{12}\, log_2({}^{8}/_{12}) \; -{}^{4}/_{12}\, log_2({}^{4}/_{12}) = 0.9183$$



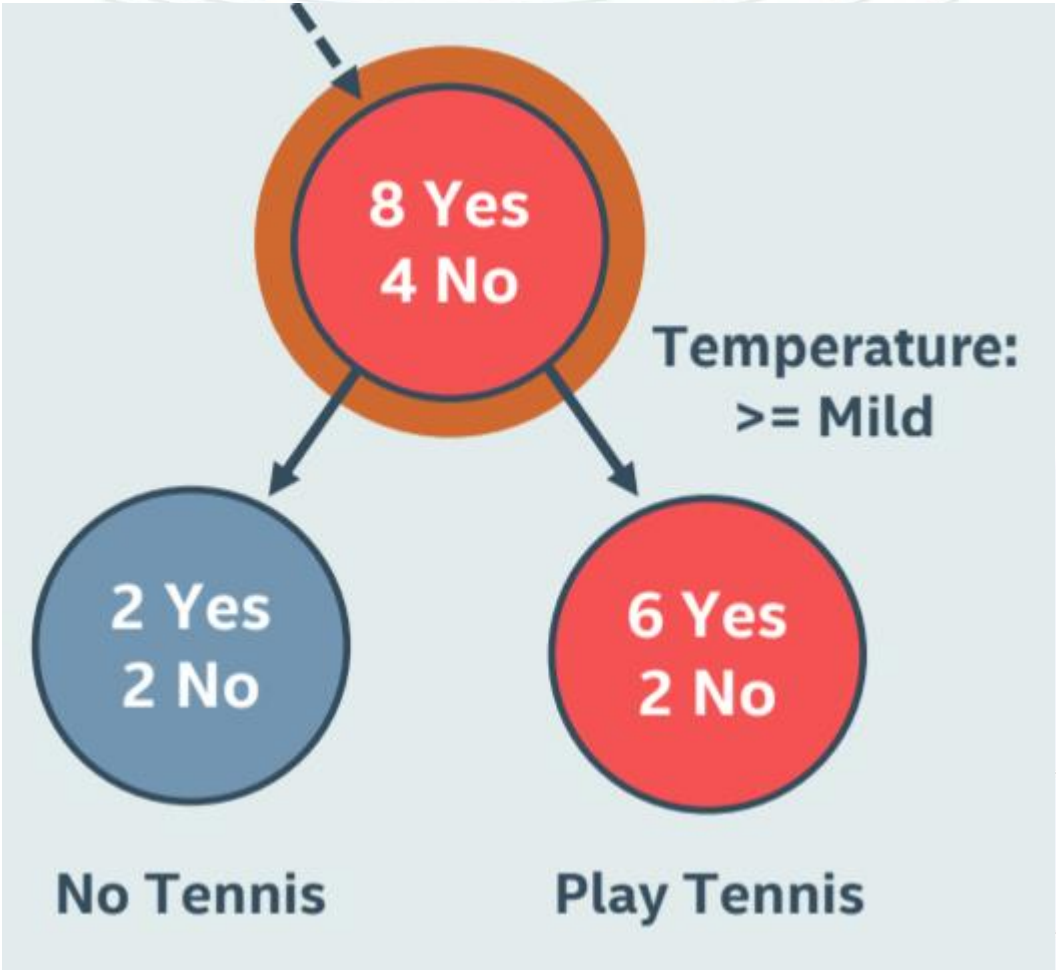
Разбиение на основе энтропии

Уравнение энтропии

$$H(t) = -\sum_{i=1}^{n} p(i|t) log_2[p(i|t)]$$

Энтропия в левой ветке

$$-{}^{2}/_{4} log_2({}^{2}/_{4}) \; -{}^{2}/_{4} log_2({}^{2}/_{4}) = 1.0000$$



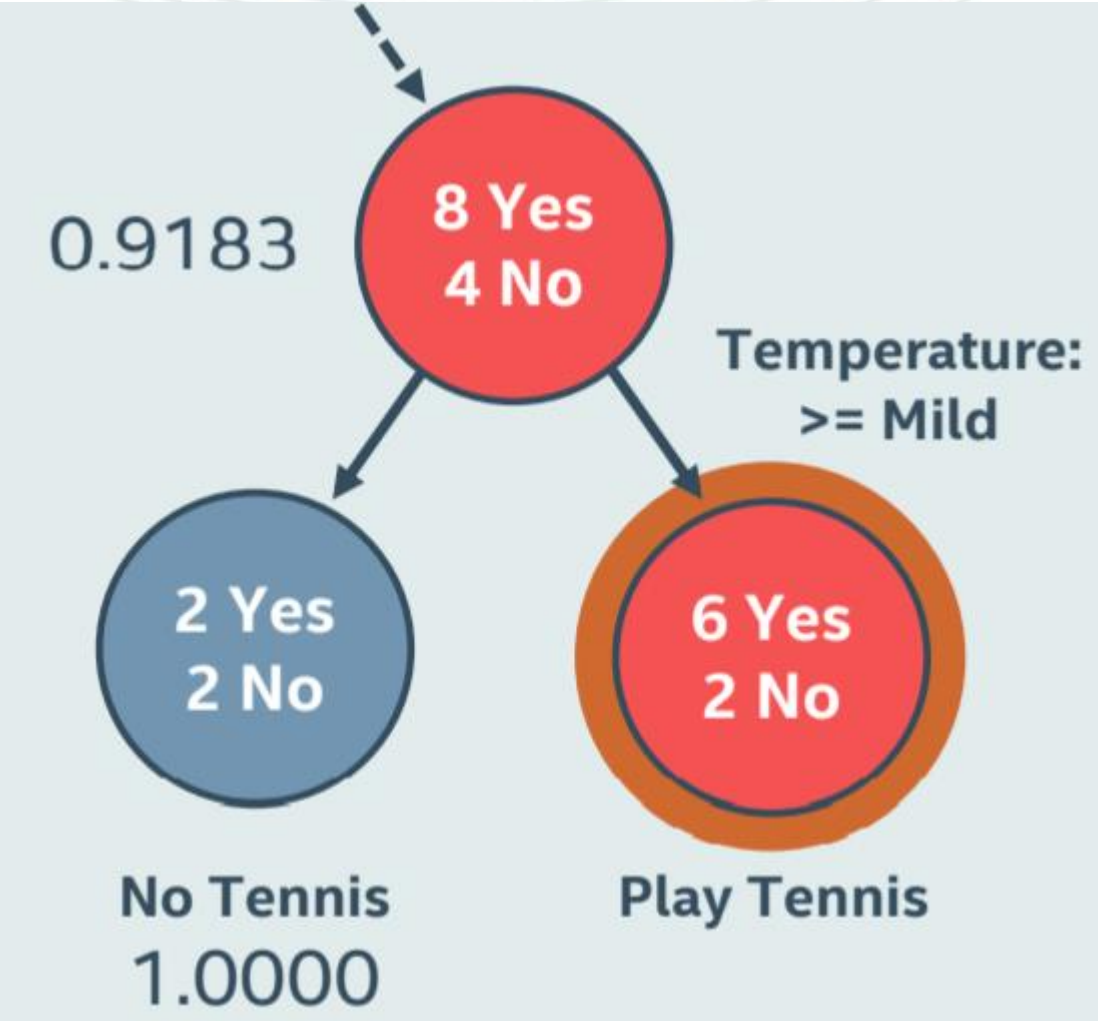
Разбиение на основе энтропии

Уравнение энтропии

$$H(t) = -\sum_{i=1}^{n} p(i|t) log_2[p(i|t)]$$

Энтропия в правой ветке ветке

$$-{}^{6}/_{8}\, log_2({}^{6}/_{8}) \; -{}^{2}/_{8}\, log_2({}^{2}/_{8}) = 0.8113$$



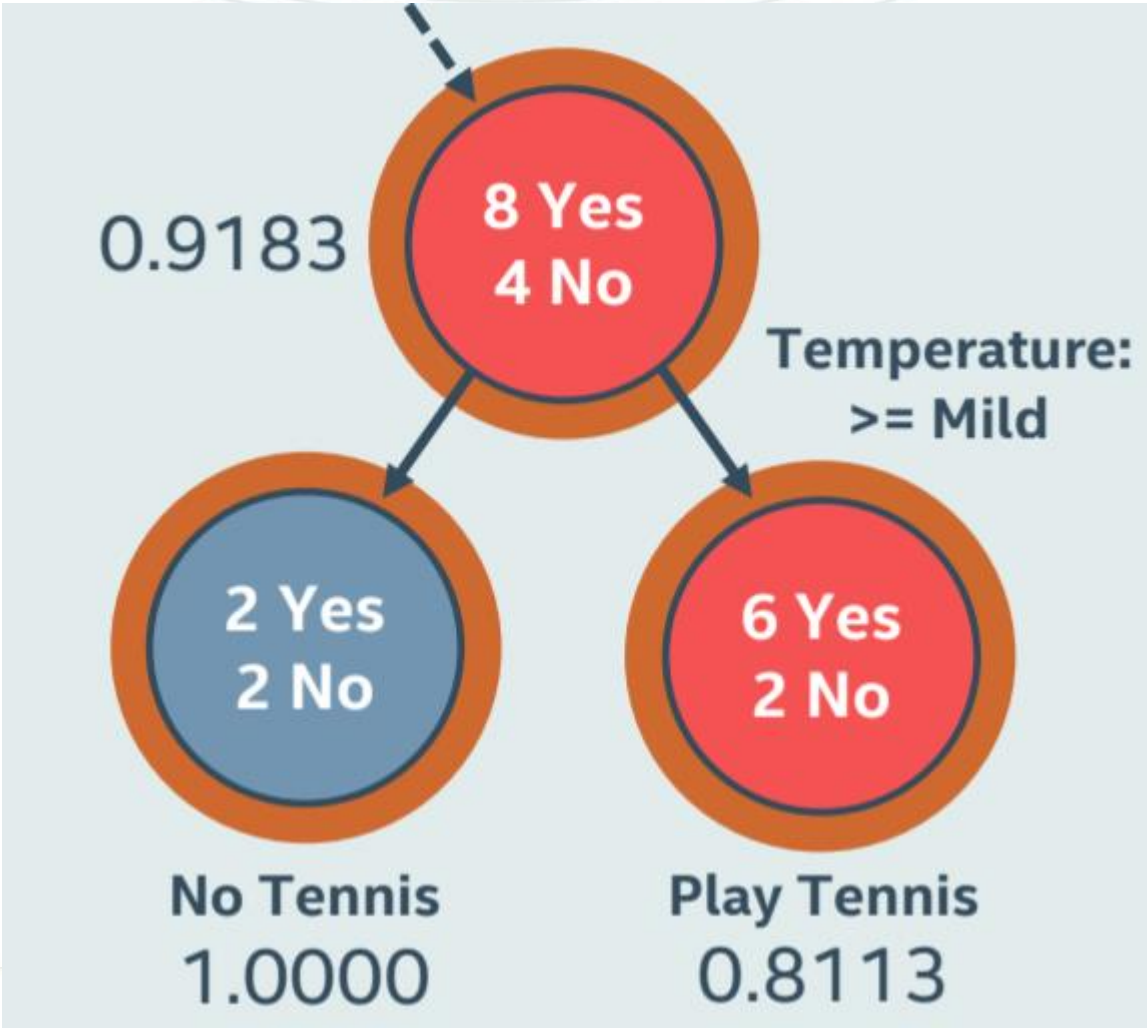
Разбиение на основе энтропии

Уравнение энтропии

$$H(t) = -\sum_{i=1}^{n} p(i|t) log_2[p(i|t)]$$

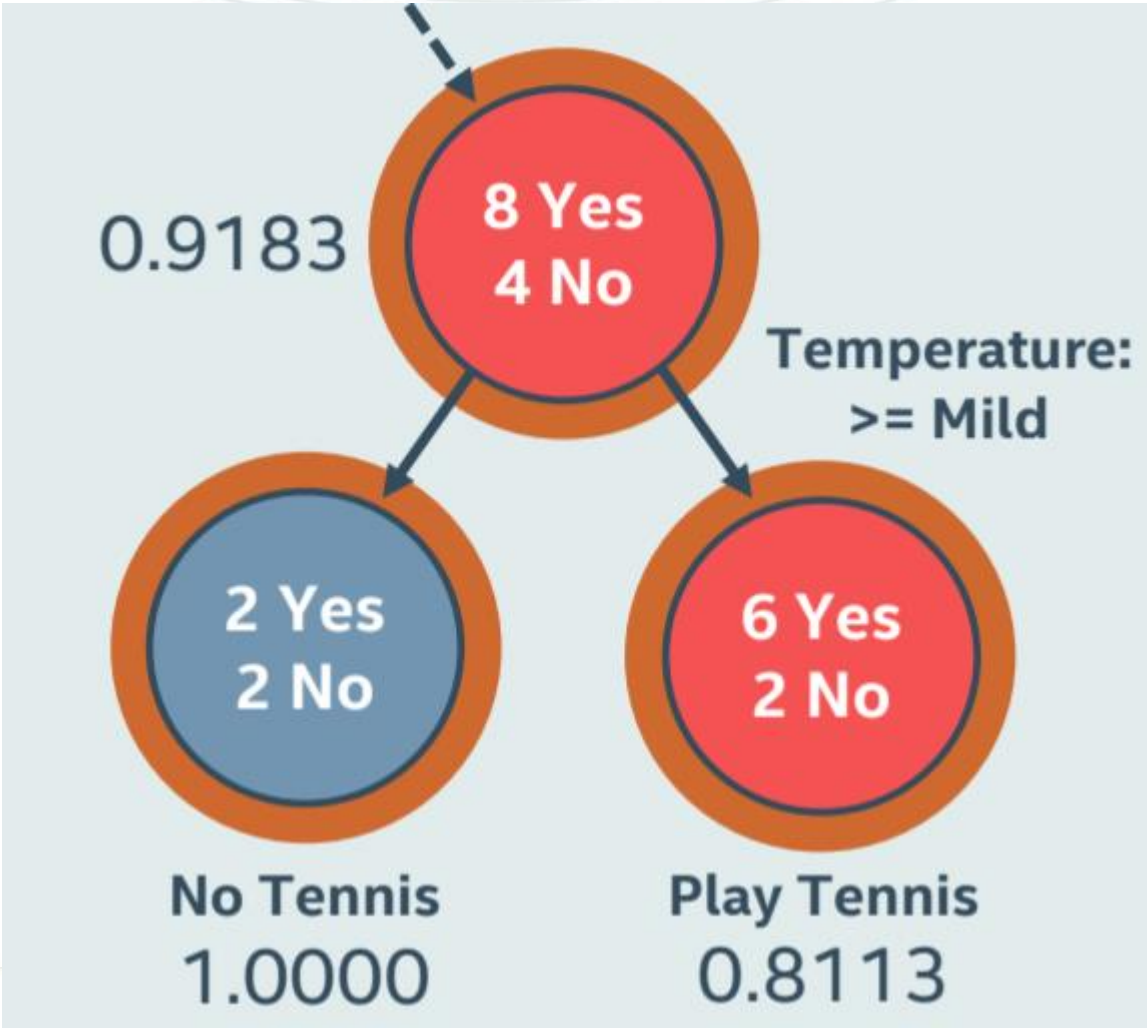
Прирост информации (уменьшение энтропии):

$$0.9183 - {}^{4}/_{12} * 1.0000 \; - {}^{8}/_{12} * 0.8113$$
$$= 0.0441$$



Разбиение на основе энтропии

- Разбиение на основе энтропии позволяет производить дальнейшее разбиение
- Может в конечном итоге достичь цели однородных узлов



Разбиение на основе неопределенности Джини (Gini impurity)

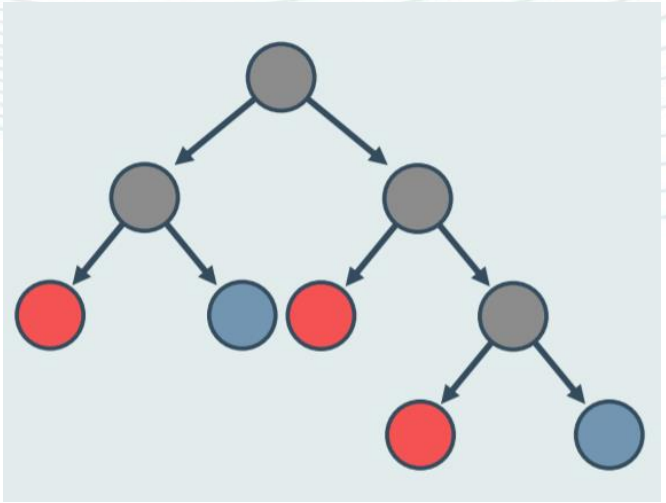
$$G(t) = 1 - \sum_{i=1}^{n} p(i|t)^2$$

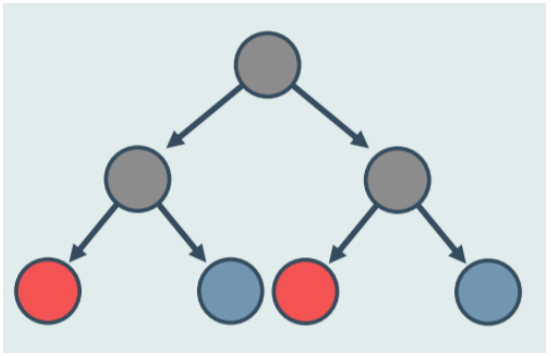
- На практике часто для разделения используется неопределенность Джини
- Не содержит логарифм
- Максимизацию этого критерия можно интерпретировать как максимизацию числа пар объектов одного класса, оказавшихся в одном поддереве

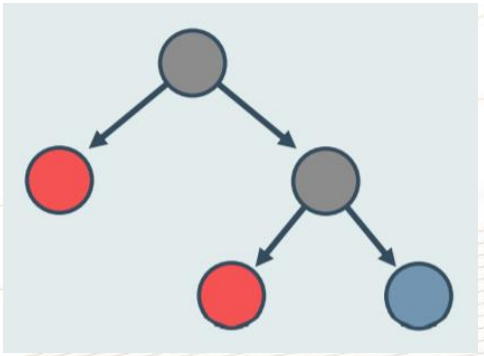
Обрезка деревьев решений

Проблема: деревья решений имеют тенденцию к переобучению

- Небольшие изменения в данных сильно влияют на прогнозирование - высокая дисперсия
- Одно из решений: обрезать деревья

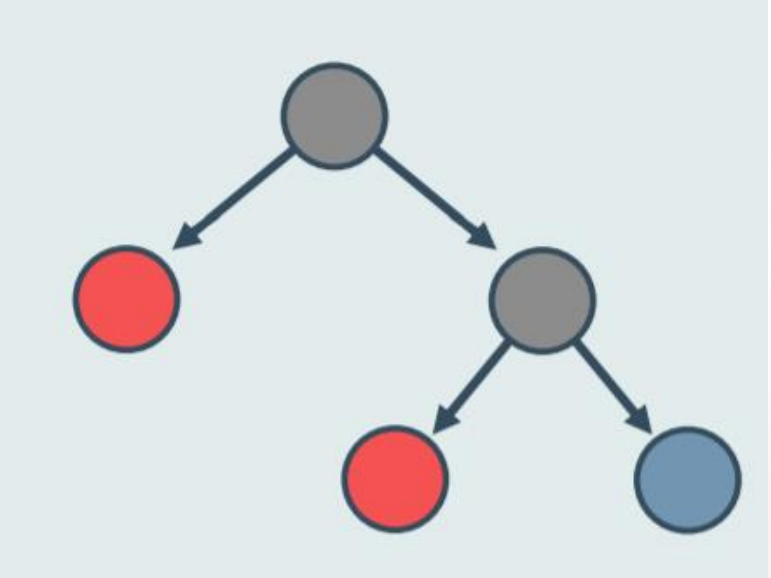






Сильные стороны деревьев решений

- Порождение четких правил классификации, понятных человеку, например, "если возраст < 25 и интерес к мотоциклам, то отказать в кредите". Это свойство называют интерпретируемостью модели
- Деревья решений могут легко визуализироваться, то есть может "интерпретироваться" (строгого определения я не видел) как сама модель (дерево), так и прогноз для отдельного взятого тестового объекта (путь в дереве)
- Быстрые процессы обучения и прогнозирования;
- Малое число параметров модели;
- Поддержка и числовых, и категориальных признаков



Слабые стороны деревьев решений

- У порождения четких правил классификации есть и другая сторона: деревья очень чувствительны к шумам во входных данных, вся модель может кардинально измениться, если немного изменится обучающая выборка (например, если убрать один из признаков или добавить несколько объектов), поэтому и правила классификации могут сильно изменяться, что ухудшает интерпретируемость модели
- Разделяющая граница, построенная деревом решений, имеет свои ограничения (состоит из гиперплоскостей, перпендикулярных какой-то из координатной оси), и на практике дерево решений по качеству классификации на определенных задачах уступает некоторым другим методам
- Необходимость отсекать ветви дерева (pruning) или устанавливать минимальное число элементов в листьях дерева или максимальную глубину дерева для борьбы с переобучением
- Нестабильность. Небольшие изменения в данных могут существенно изменять построенное дерево решений

Слабые стороны деревьев решений

- Проблема поиска оптимального дерева решений (минимального по размеру и способного без ошибок классифицировать выборку) требует полного перебора всех комбинаций, поэтому на практике используются эвристики типа жадного поиска признака с максимальным приростом информации, которые не гарантируют нахождения глобально оптимального дерева
- Сложно поддерживаются пропуски в данных
- Модель умеет только интерполировать, но не экстраполировать. То есть дерево решений делает константный прогноз для объектов, находящихся в признаковом пространстве вне параллелепипеда, охватывающего все объекты обучающей выборки

В принципе дерево решений можно построить до такой глубины, чтоб в каждом листе был ровно один объект. Но на практике это не делается (если строится только одно дерево) из-за того, что такое дерево будет переобученным – оно слишком настроится на обучающую выборку и будет плохо работать на прогноз на новых данных. Где-то внизу дерева, на большой глубине будут появляться разбиения по менее важным признакам.

Есть два исключения, когда деревья строятся до максимальной глубины:

- Случайный лес (композиция многих деревьев) усредняет ответы деревьев, построенных до максимальной глубины
- Стрижка дерева (pruning). При таком подходе дерево сначала строится до максимальной глубины, потом постепенно, снизу вверх, некоторые вершины дерева убираются за счет сравнения по качеству дерева с данным разбиением и без него (сравнение проводится с помощью кросс-валидации)

Основные способы борьбы с переобучением в случае деревьев решений:

- искусственное ограничение глубины или минимального числа объектов в листе - построение дерева просто в какой-то момент прекращается
- стрижка дерева

Класс DecisionTreeClassifier в Scikit-learn

Основные параметры класса sklearn.tree.DecisionTreeClassifier:

- max_depth – максимальная глубина дерева
- max_features — максимальное число признаков, по которым ищется лучшее разбиение в дереве (это нужно потому, что при большом количестве признаков будет "дорого" искать лучшее (по критерию типа прироста информации) разбиение среди всех признаков)
- min_samples_leaf – минимальное число объектов в листе. У этого параметра есть понятная интерпретация: скажем, если он равен 5, то дерево будет порождать только те классифицирующие правила, которые верны как минимум для 5 объектов. Если примеров, которые попадают в данный узел, будет меньше заданного, узел считается листом (т.е. дальнейшее ветвление прекращается). Чем больше этот параметр, тем менее ветвистым получается дерево.

Параметры дерева надо настраивать в зависимости от входных данных, и делается это обычно с помощью кросс-валидации

DecisionTreeClassifier: синтаксис

Импортируем класс, содержащий метод классификации

```
from sklearn.tree import DecisionTreeClassifier
```

Создадим экземпляр класса

```
DTC = DecisionTreeClassifier(criterion='gini',
                             max_features=10, max_depth=5)
```

tree parameters

DecisionTreeClassifier: синтаксис

Импортируем класс, содержащий метод классификации

```
from sklearn.tree import DecisionTreeClassifier
```

Создадим экземпляр класса

```
DTC = DecisionTreeClassifier(criterion='gini',
                             max_features=10, max_depth=5)
```

Обучим дерево на обучающем выборке, а затем прогнозируем ожидаемое значение на тестовой выборке

```
DTC = DTC.fit(X_train, y_train)
y_predict = DTC.predict(X_test)
```

Используйте DecisionTreeRegressor для регрессии.
Настройте параметры перекрестной проверки