Деревья решений (decision trees)

PIERIAN DATA

Деревья решений

- Следующие три раздела посвящены методам на основе деревьев.
- Три основных метода:
  - Деревья решений (decision trees)
  - Случайные леса (random forests)
  - Расширяемые деревья (boosted trees)

Деревья решений

- Все эти три метода основаны на базовом алгоритме деревьев решений.
- Мы пройдём все три метода, каждый в своём разделе курса, и в самом конце уже будут проверочные упражнения.

Деревья решений

История

PIERIAN DATA

Деревья решений

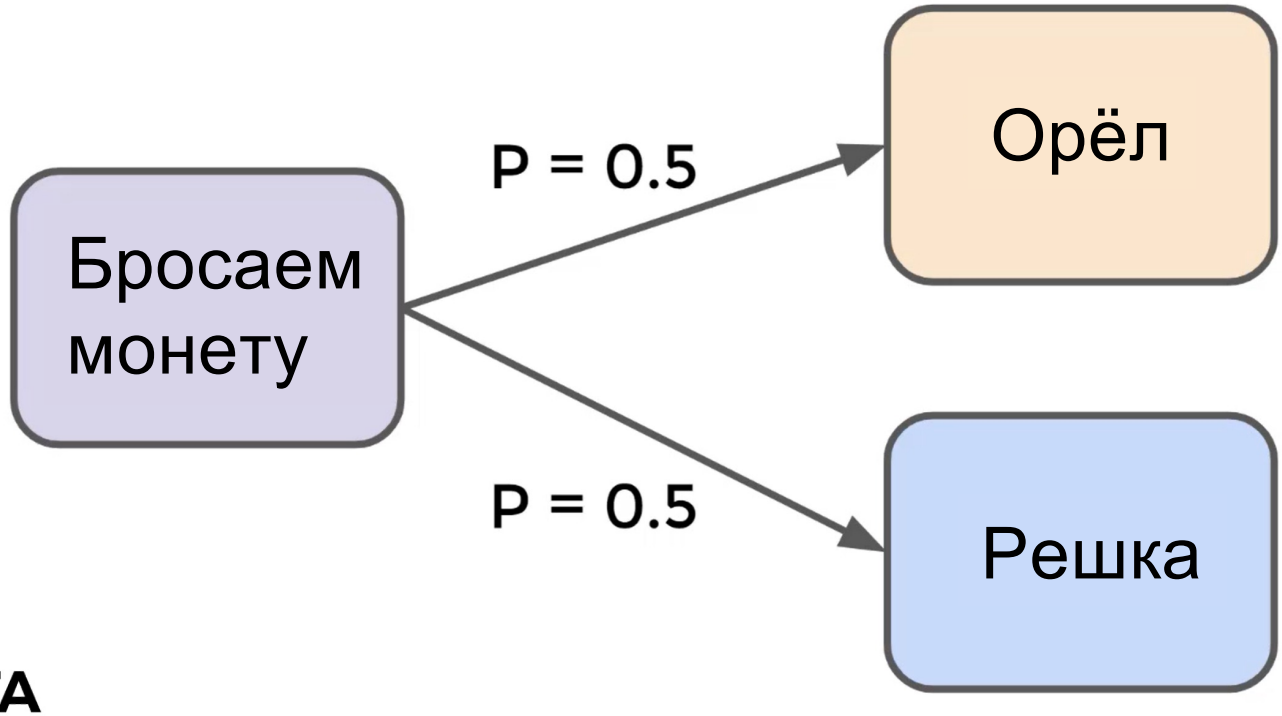
- Хотя базовые деревья решений использовались для принятия решений уже очень длительное время, статистические деревья решений были разработаны относительно недавно.
- Обратите внимание, это разные вещи.

Деревья решений

- Общий термин “дерево решений” можно описать как схему, отображающую выбор решения:

Деревья решений

- Общий термин “дерево решений” можно описать как схему, отображающую выбор решения:



Деревья решений

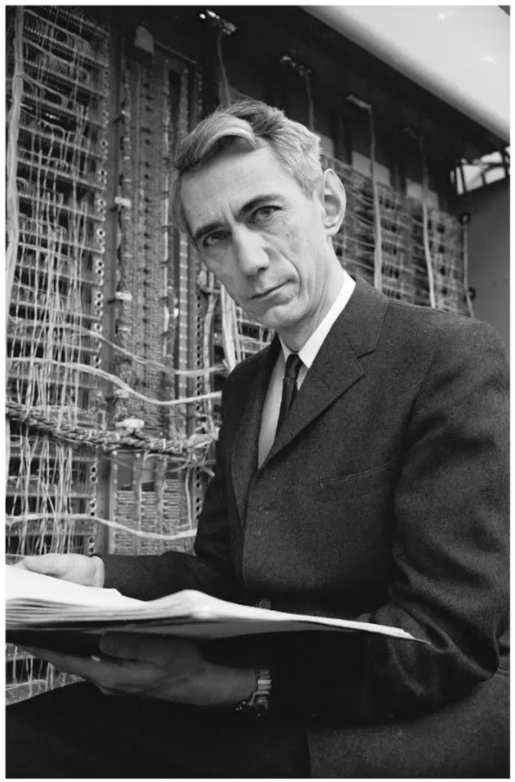
- Обучение деревьев решений – это статистическое моделирование с применением деревьев, где решения в каждом узле принимаются на основе некоторого условия (некоторой метрики).
- Давайте рассмотрим шаги, которые привели к возможности создавать предсказания на основе деревьев.

Деревья решений

- В двух словах – деревья решений и похожие методы разбивают данные на части на основе информации, содержащейся в признаках.
- Это значит, что нам нужно математическое определение термина информация, и способность измерить её.

Деревья решений

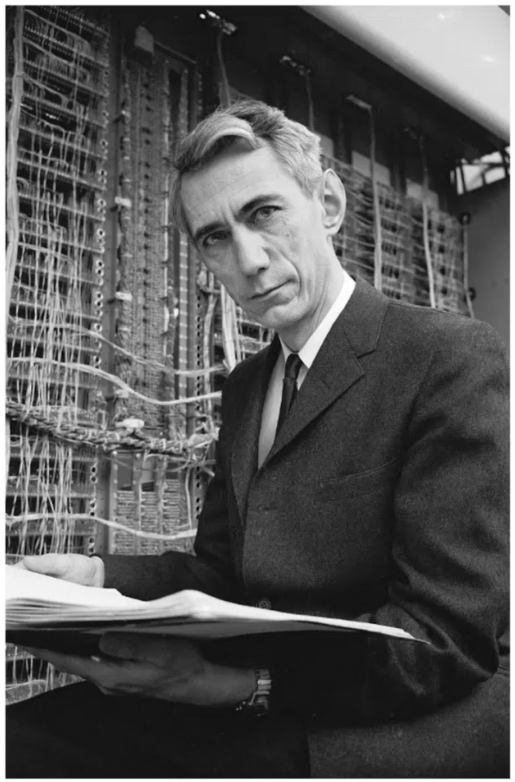
- Клод Шеннон считается отцом теории информации.
- В 1948 году в журнале Bell System Technical Journal опубликовал работу “Математическая теория коммуникаций”.



PIERIAN DATA

Деревья решений

- Работал в разных сферах:
  - Проектирование микросхем
  - Криптография
  - Носимые компьютеры
  - Искусственный интеллект

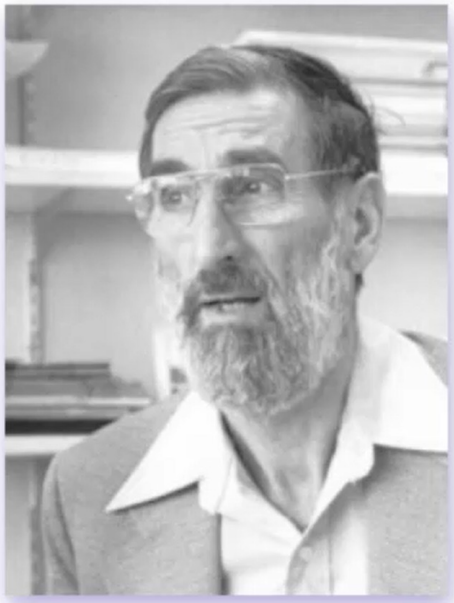


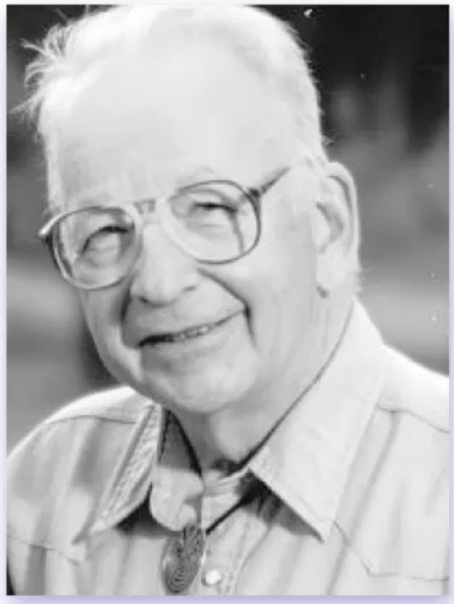
Деревья решений

- Способность измерить информацию станет важна, когда мы рассмотрим математические основы построения деревьев решений.
- Мы займёмся этим позже, а пока посмотрим на этапы развития деревьев решений.

Деревья решений

- 1963: Морган, Санквист - первая публикация регрессионного алгоритма деревьев





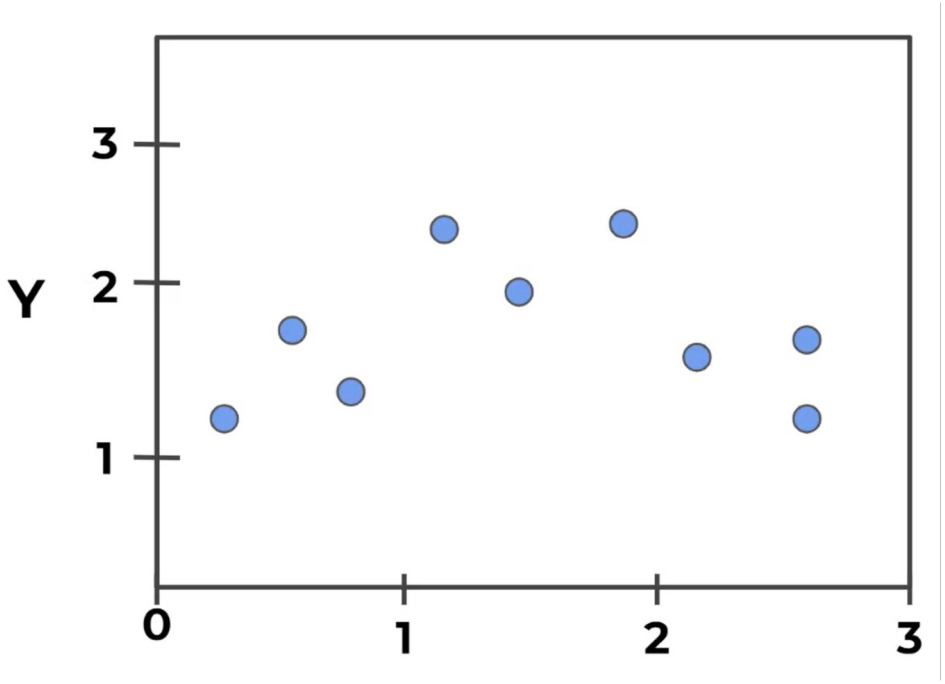
Деревья решений

- 1963: piecewise-constant regression tree

PIERIAN DATA

Деревья решений

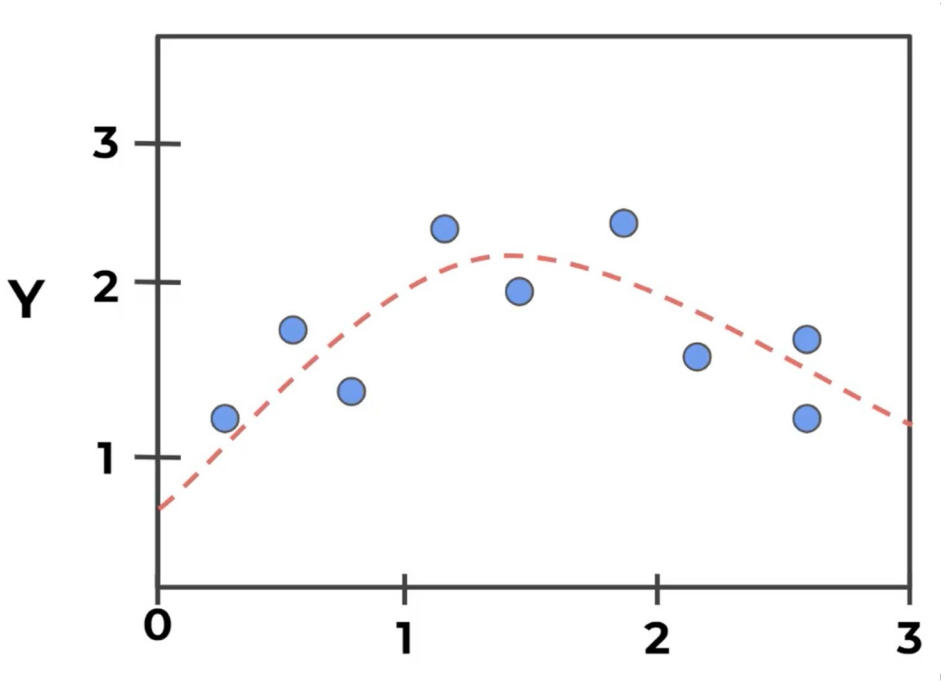
- 1963: piecewise-constant regression tree



PIERIAN DATA

Деревья решений

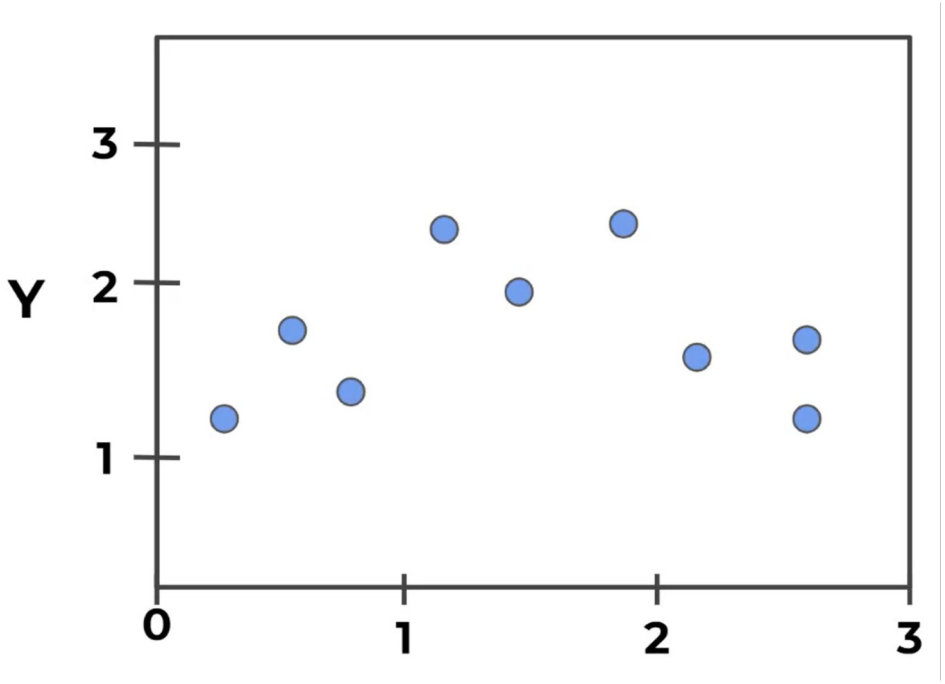
- 1963: piecewise-constant regression tree



PIERIAN DATA

Деревья решений

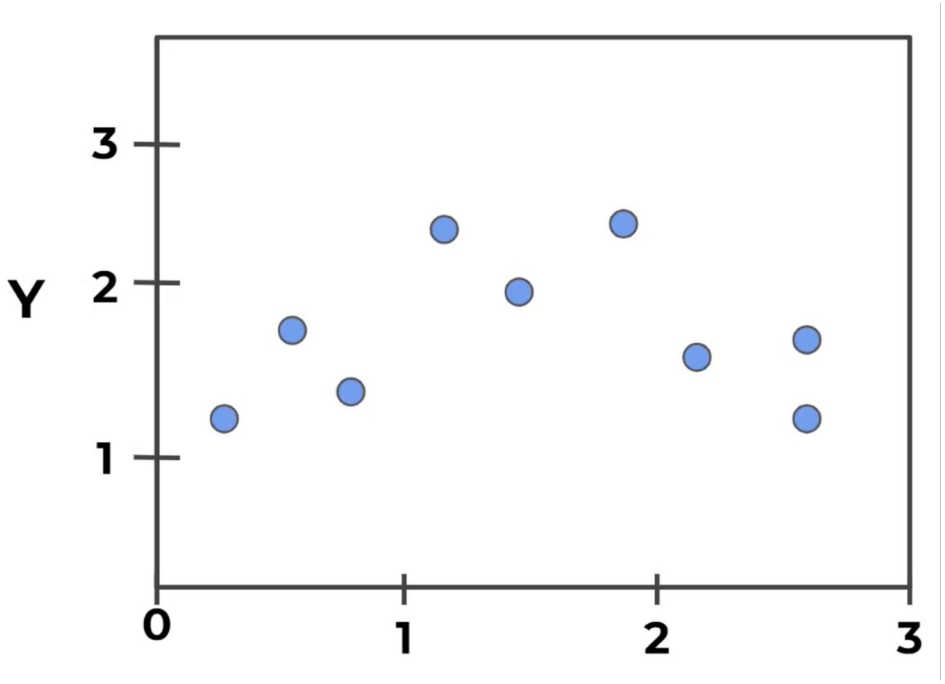
- 1963: piecewise-constant regression tree



PIERIAN DATA

Деревья решений

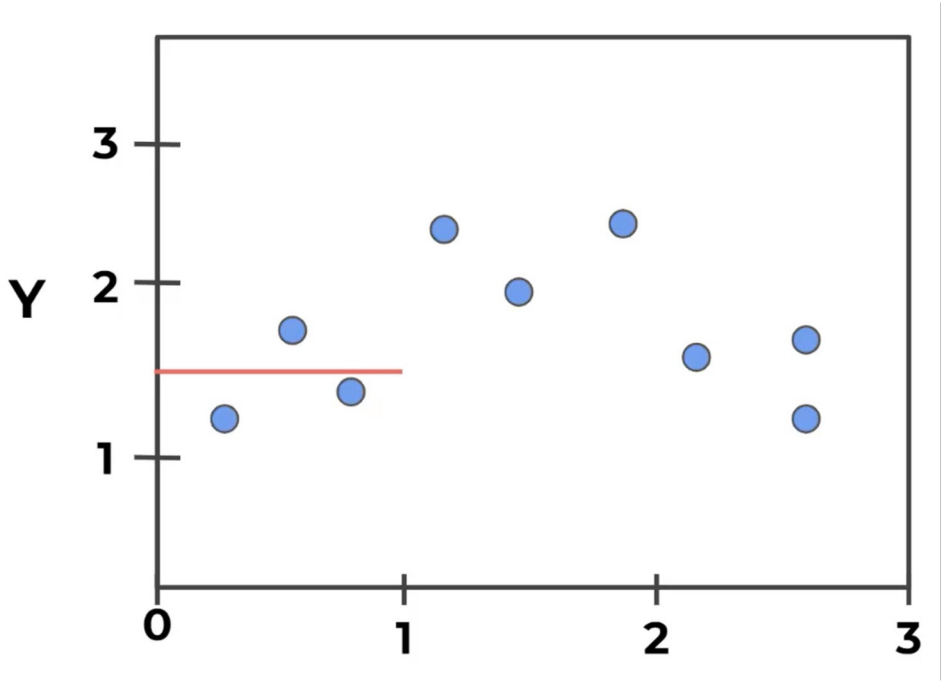
- 1963: piecewise-constant regression tree

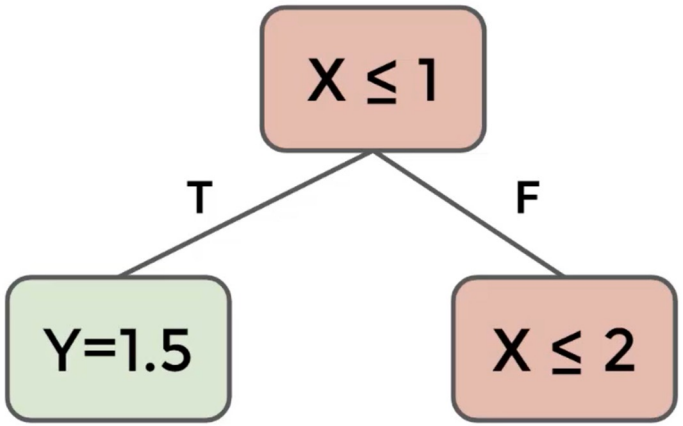


X < 1

Деревья решений

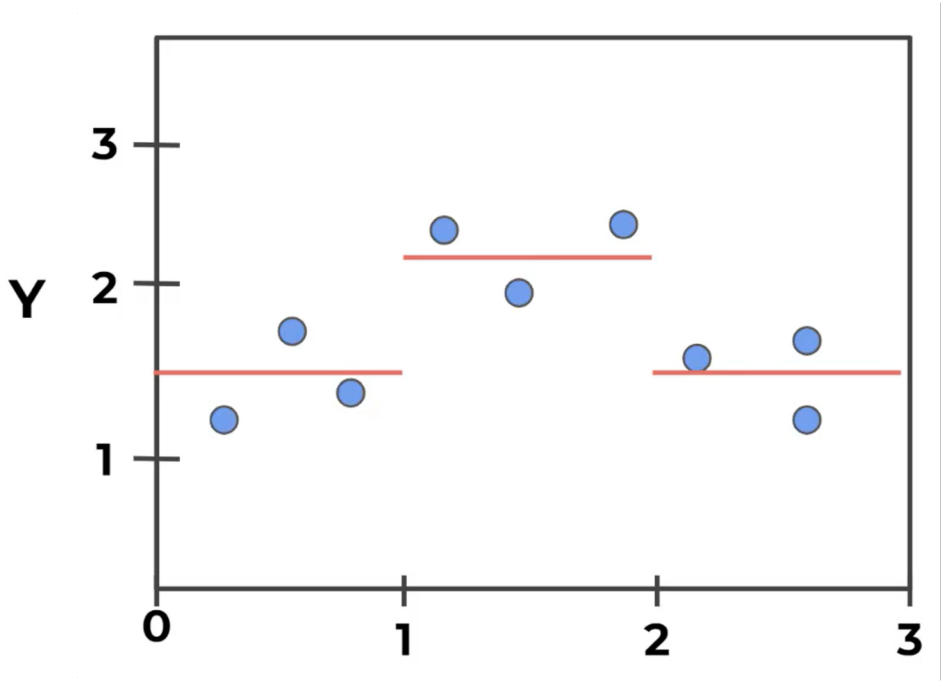
- 1963: piecewise-constant regression tree

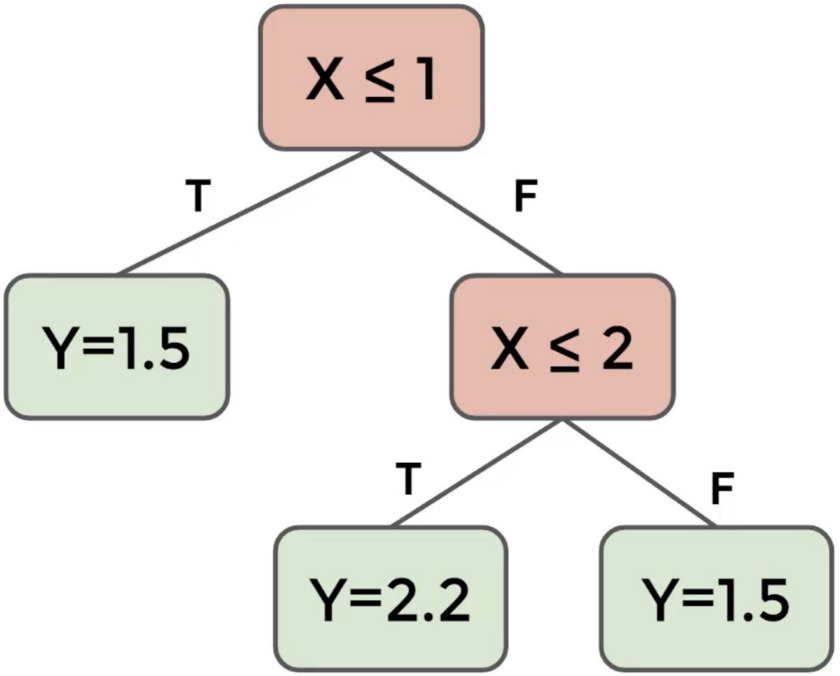




Деревья решений

- 1963: piecewise-constant regression tree

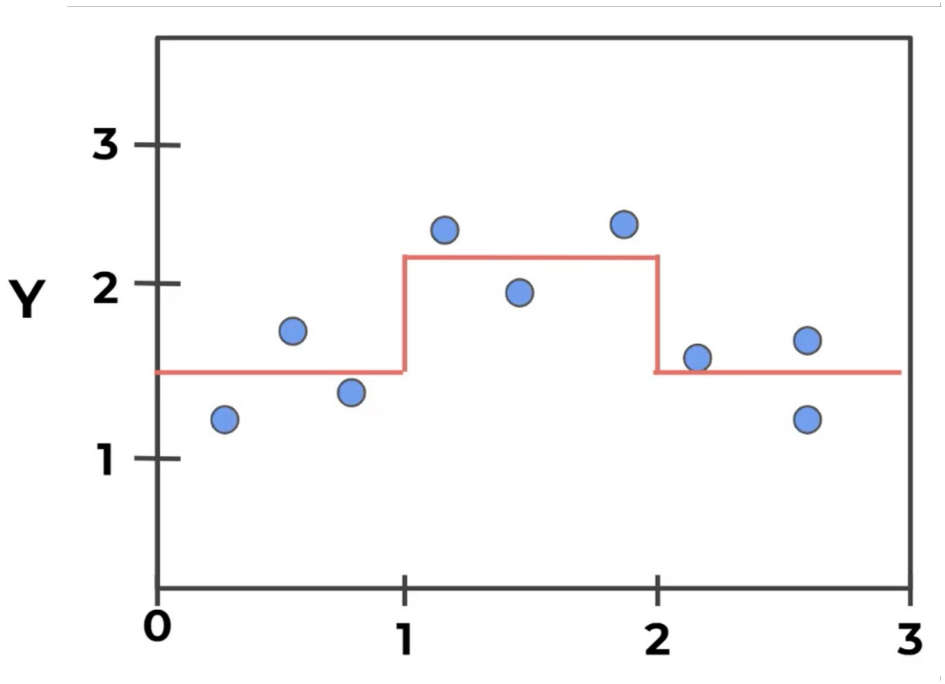


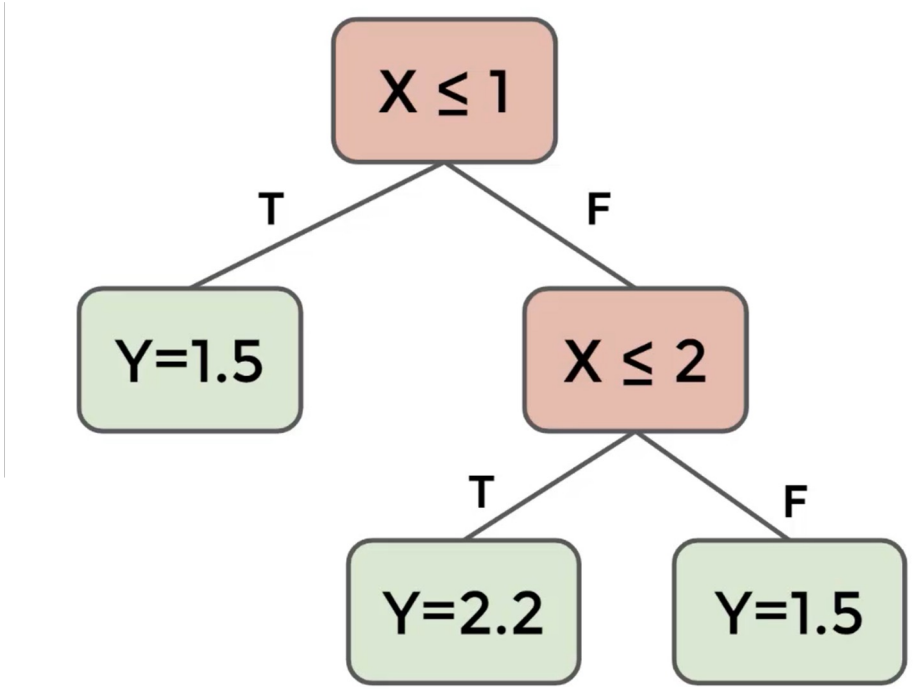


PIERIAN DATA

Деревья решений

- 1963: piecewise-constant regression tree





Деревья решений

- В статье 1963 года разбиения в каждом узле дерева основывались на “загрязнении узла” (node impurity), которое по сути определялось как метрика ошибки:

$$\phi(t) = \sum_{i \in t} (y_i - \bar{y})^2$$

Деревья решений

- 1972: Роберт Мессенджер и Льюис Манделл публикуют первый алгоритм классификации с применением деревьев.
- Условие разделения в узлах было названо “Theta Automatic Interaction Detection” (THAID)

Деревья решений

- 1972: Роберт Мессенджер и Льюис Манделл публикуют первый алгоритм классификации с применением деревьев.
- Условие разделения в узлах было названо “Theta Automatic Interaction Detection” (THAID)
- 1980: Гордон Кисс публикует развитие алгоритма CHAID – Chi-square automatic interaction detection.

Деревья решений

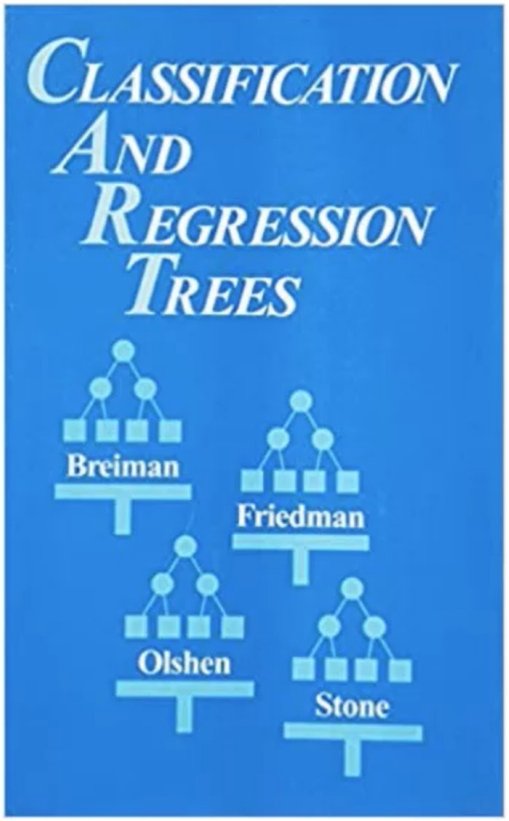
- 1970-е годы: Лео Брейман и Чарльз Стоун из Беркли, Джером Фридман и Ричард Ольшен из Стэнфорда начали разработку алгоритмов CART – Classification and Regression tree.

Деревья решений

- 1970-е годы: Лео Брейман и Чарльз Стоун из Беркли, Джером Фридман и Ричард Ольшен из Стэнфорда начали разработку алгоритмов CART – Classification and Regression tree.
- 1984: Они опубликовали книгу по алгоритмам CART, включая детали реализации на компьютере.
- Методы на основе CART стали стандартом (и в Scikit-Learn!)

Деревья решений

- В книге CART были предложены многие концепции:
  - Кросс-валидация деревьев
  - Усечение деревьев (pruning)
  - Суррогатные разбиения
  - Оценки важности переменных
  - Поиск линейных разделений



Деревья решений

- 1986: Джон Росс Квинлан разработал алгоритм деревьев решений ID3 на основе метрики “gain ratio”.
- 1990-е: улучшения ID3 – C4.5 (всё ещё популярен)
- 2000-е: выпущена оптимизированная коммерческая версия C5.0 с различными улучшениями.

Деревья решений

- Многие улучшения основного алгоритма были перенесены на другие методы с применением деревьев – случайные леса и расширяемые деревья.
- Давайте посмотрим на фундаментальную идею деревьев решений!

Деревья решений

Терминология

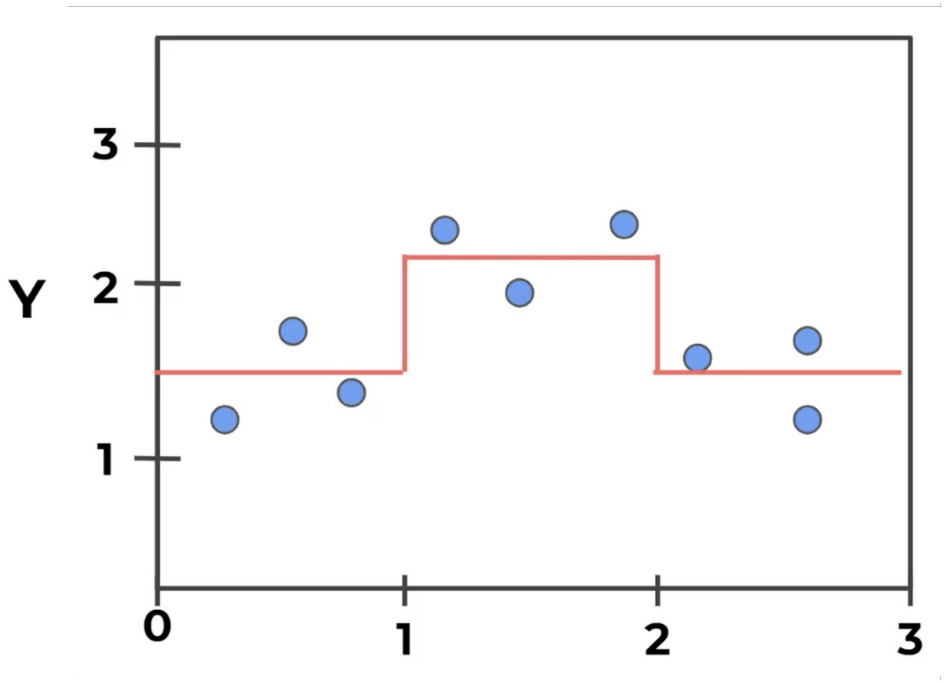
PIERIAN DATA

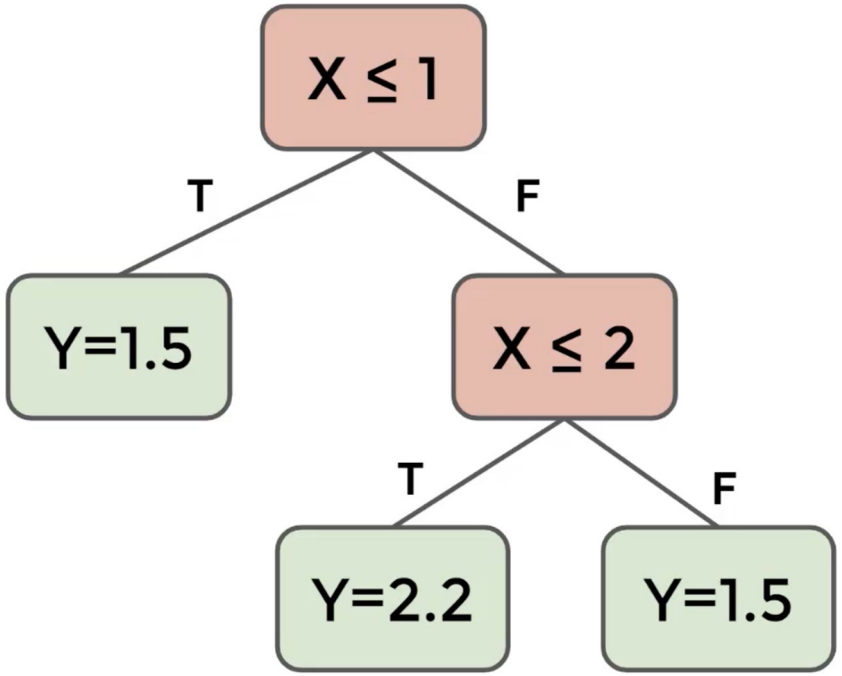
Деревья решений

- Для начала обсудим различные термины, которые встречаются при обсуждении деревьев...

Деревья решений

- Вспомним наше простое дерево:

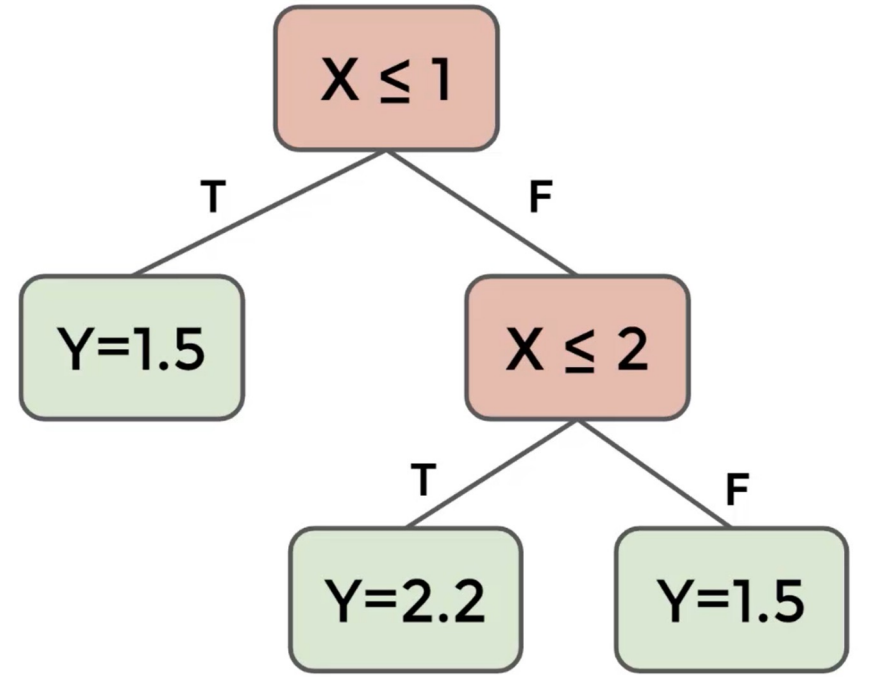




PIERIAN DATA

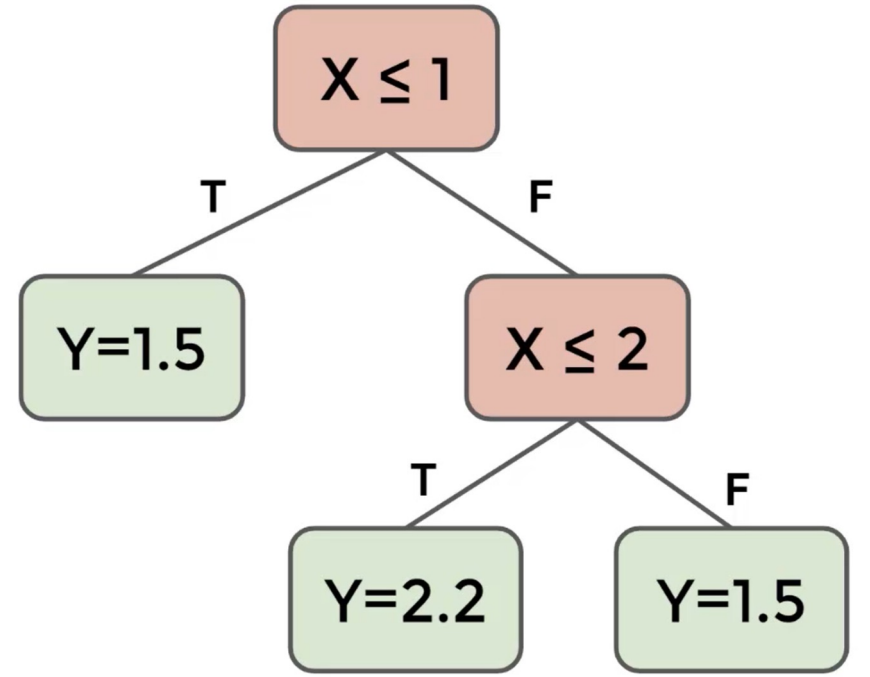
Деревья решений

- Вспомним наше простое дерево:



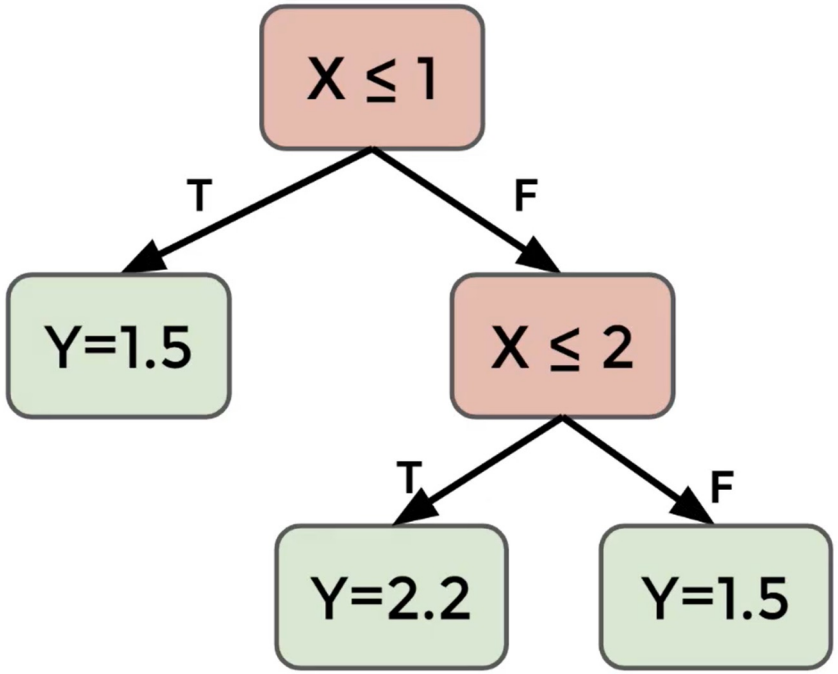
Деревья решений

- Разбиение (splitting):



Деревья решений

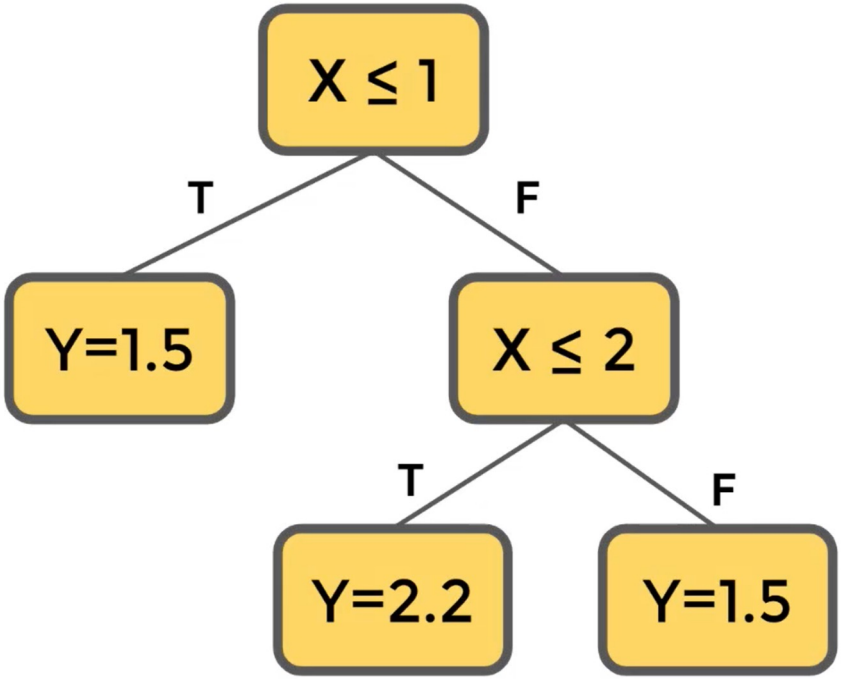
- Разбиение (splitting):



PIERIAN DATA

Деревья решений

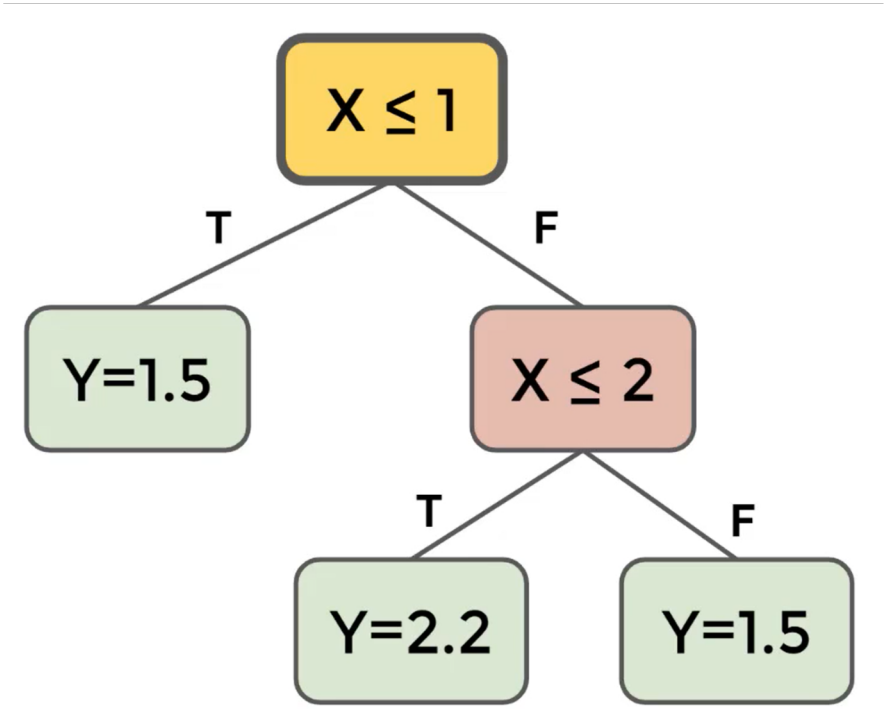
- Узлы (nodes):



PIERIAN DATA

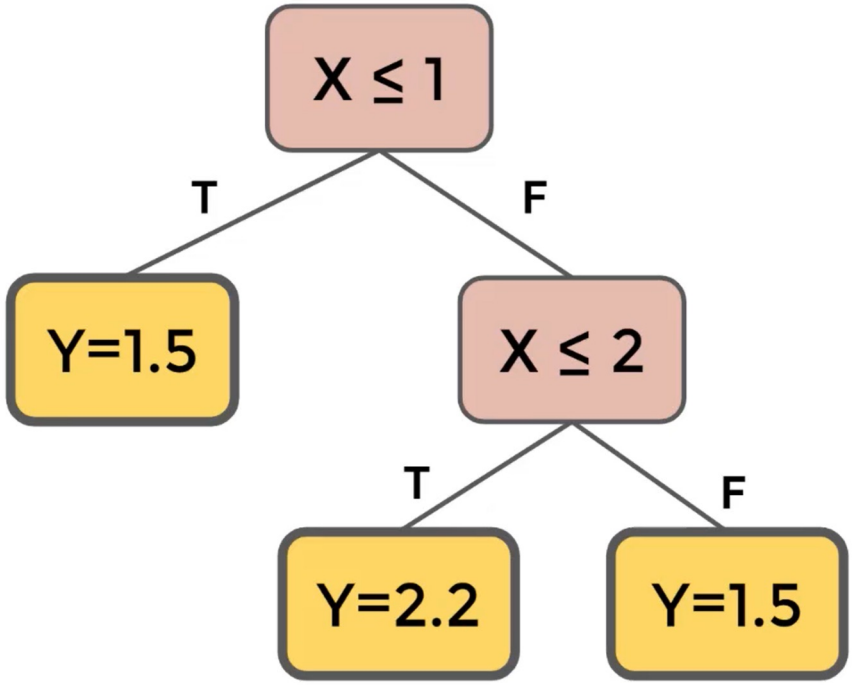
Деревья решений

- Корневой узел (root node):



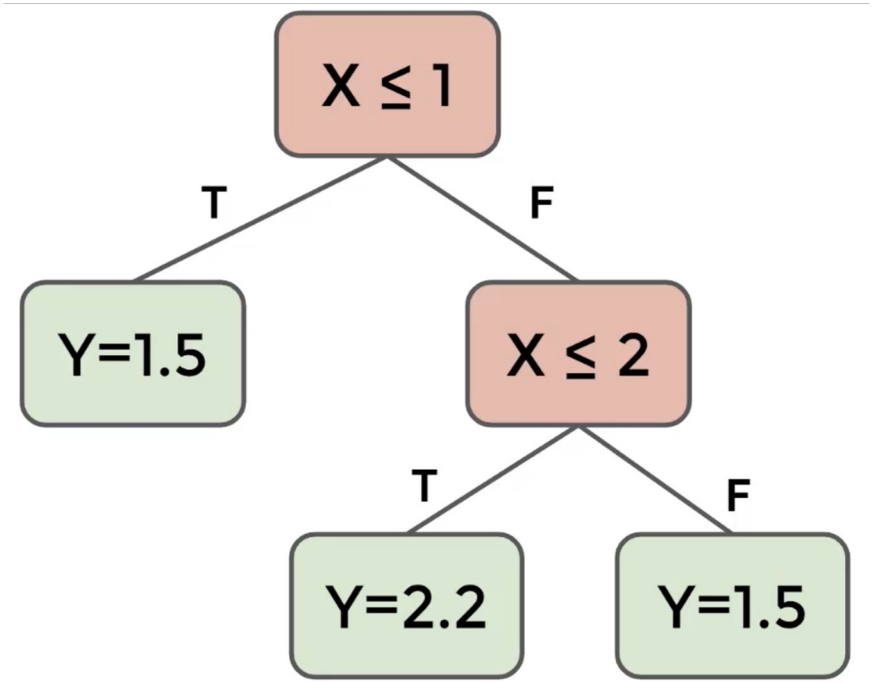
Деревья решений

- Листовые / конечные узлы (leaf / terminal nodes):



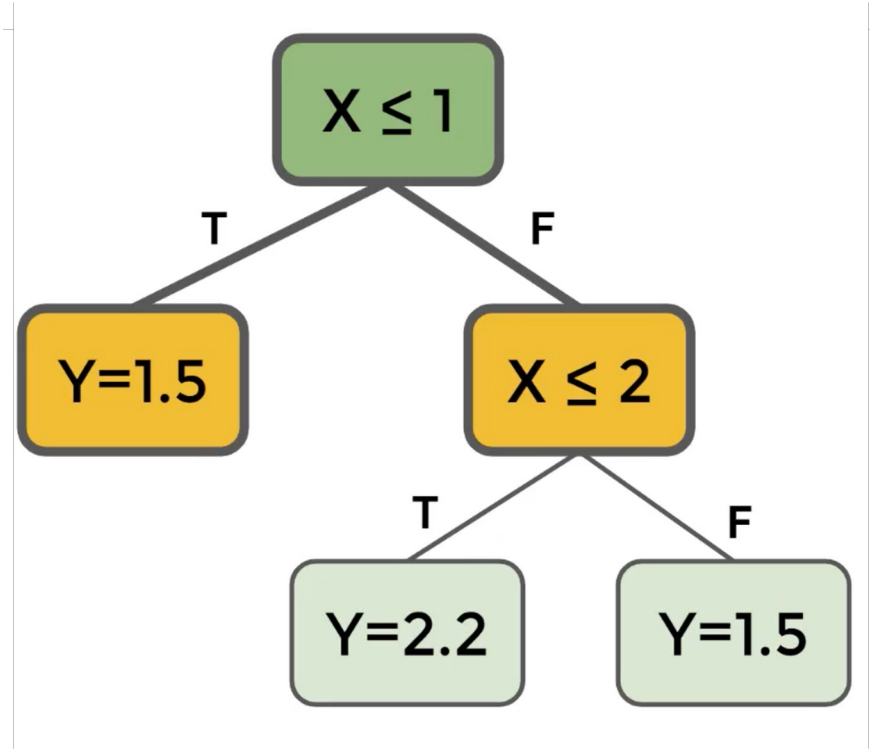
Деревья решений

- Родительские/дочерние узлы (parent/children nodes):



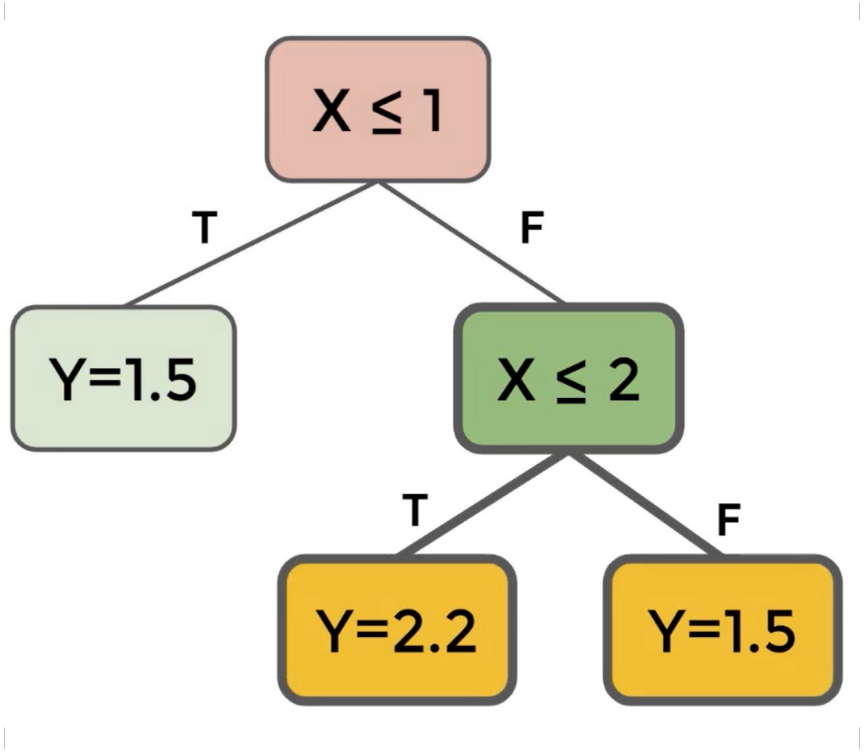
Деревья решений

- Родительские/дочерние узлы (parent/children nodes):



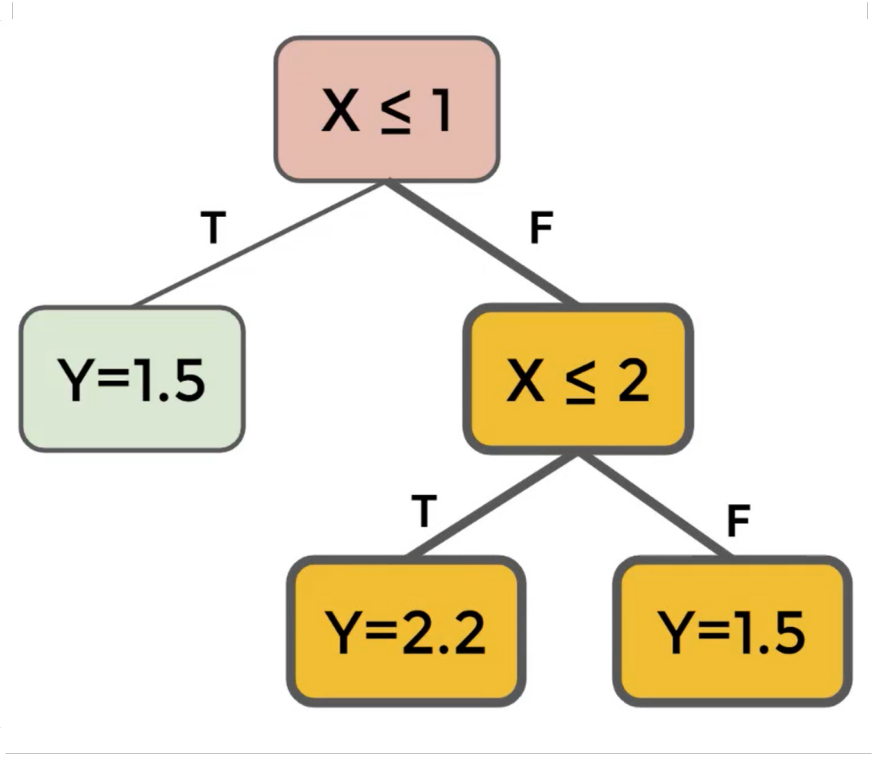
Деревья решений

- Родительские/дочерние узлы (parent/children nodes):



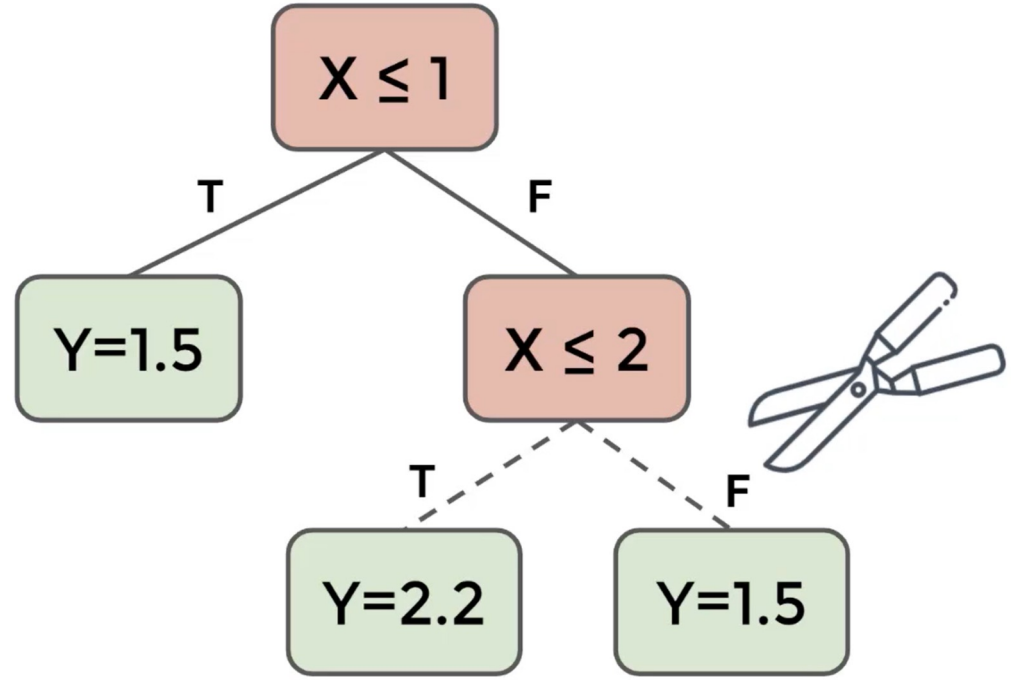
Деревья решений

- Ветви / поддеревья (tree branches / sub trees):



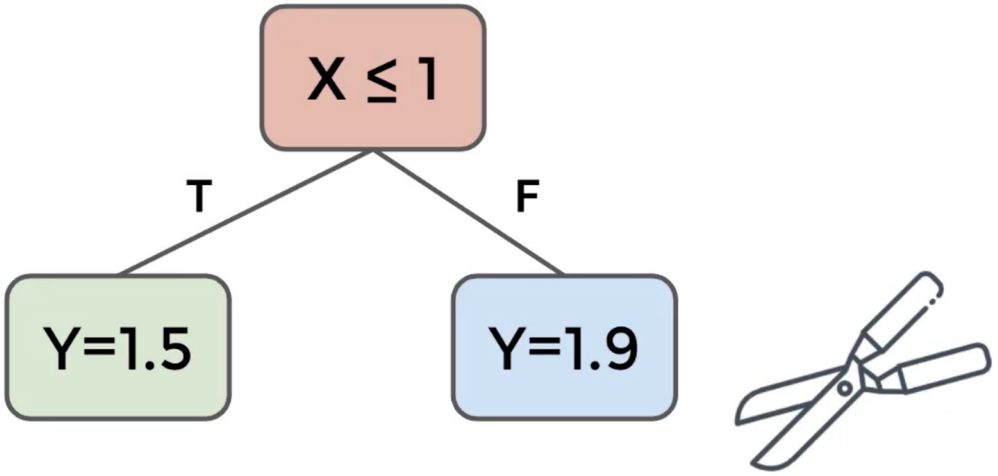
Деревья решений

- Усечение (pruning)



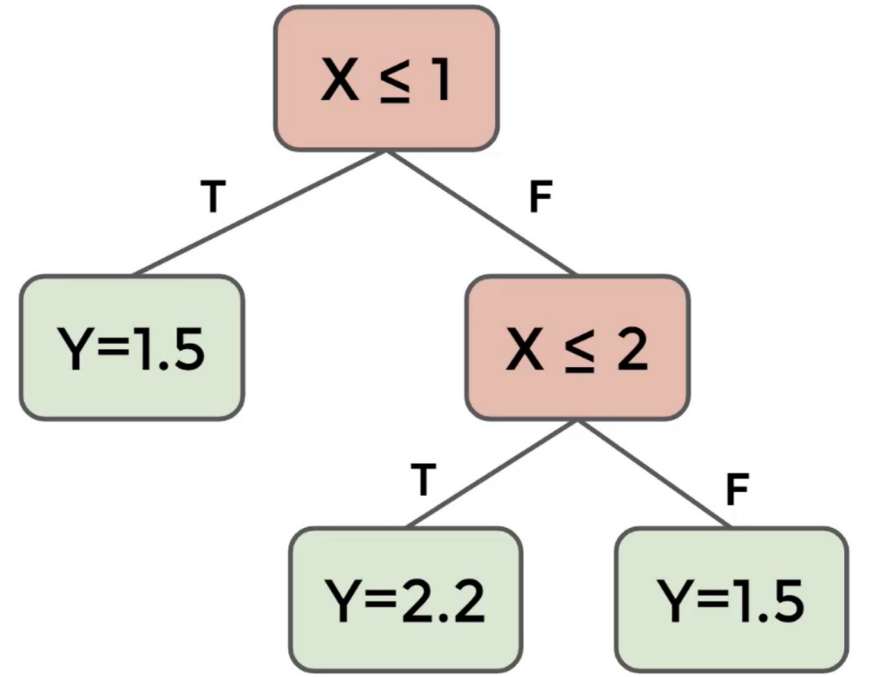
Деревья решений

- Усечение (pruning)



Деревья решений

- Далее мы перейдём к построению дерева!



Деревья решений

Gini impurity

PIERIAN DATA

Деревья решений

- Прежде чем мы узнаем, как при построении деревьев выбираются критерии разбиения, давайте взглянем на наиболее часто используемую метрику измерения информации для деревьев решений – gini impurity (загрязнение Джини, индекс Джини, примесь Джини).

Деревья решений

- Метрика “gini impurity” измеряет, насколько информация в наборе данных является чистой (“pure”).
- С точки зрения задач классификации, мы можем воспринимать это как однородность классов.
- Посмотрим как это выглядит на простейшем примере двух классов...

Деревья решений

- Метрика “gini impurity” для классификации
  - Для набора классов C и заданного набора данных Q:

$$G(Q) = \sum_{c \in C} p_c(1 - p_c)$$

Деревья решений

- Метрика “gini impurity” для классификации
  - Для набора классов C и заданного набора данных Q, $\mathbf{p_c}$ это вероятность класса **c.**

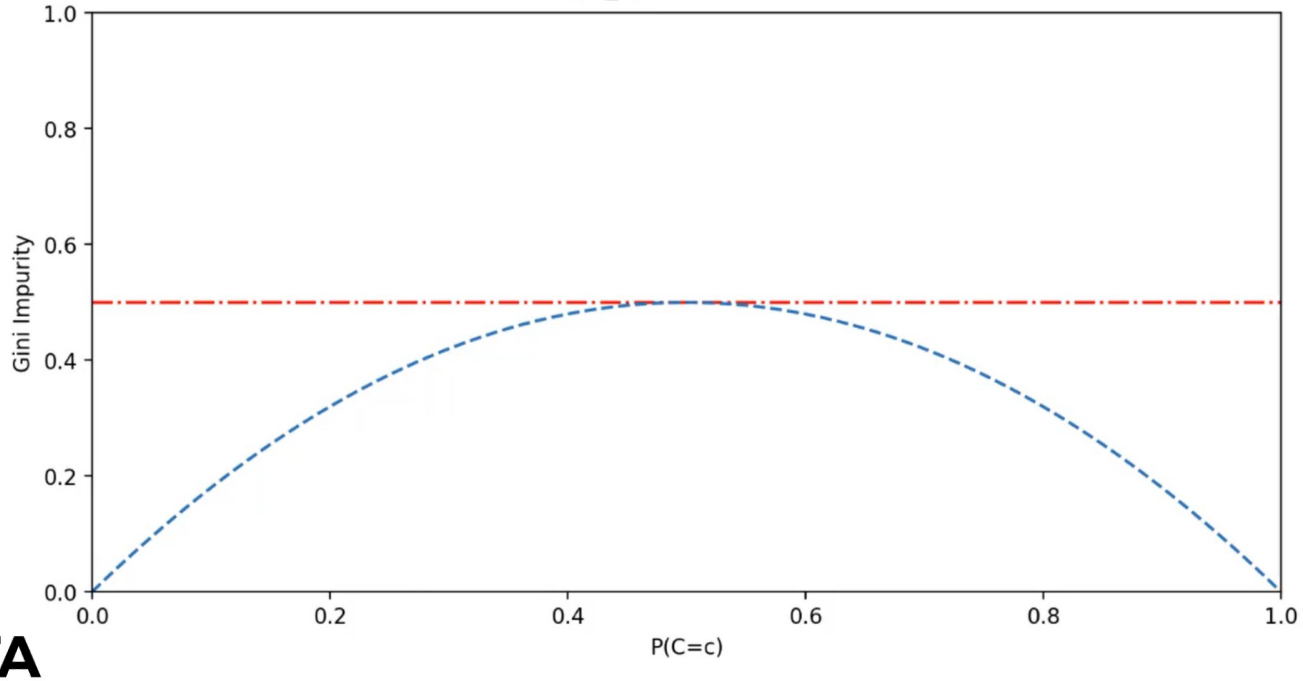
$$p_c = \frac{1}{N_Q} \sum_{x \in Q} \mathbb{1}(y_{class} = c)$$

$$G(Q) = \sum_{c \in C} p_c(1 - p_c)$$

Деревья решений

- Метрика “gini impurity” для классификации

$$G(Q) = \sum_{c \in C} p_c(1 - p_c)$$

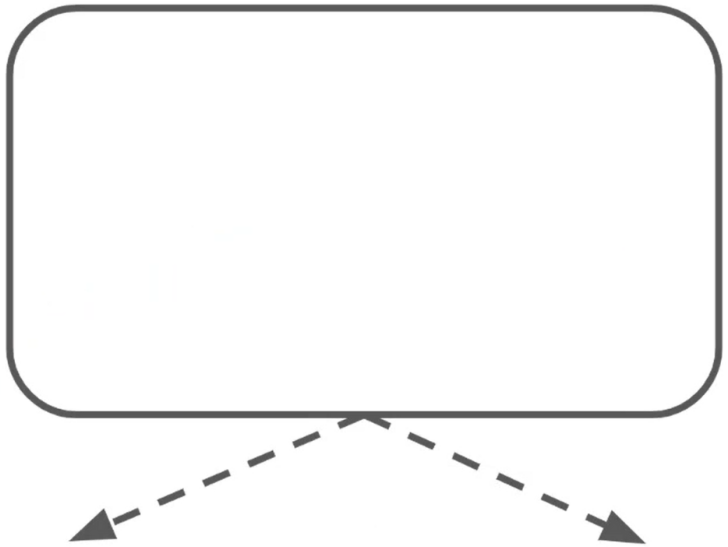


PIERIAN DATA

Деревья решений

- Метрика “gini impurity” для классификации

$$G(Q) = \sum_{c \in C} p_c(1 - p_c)$$

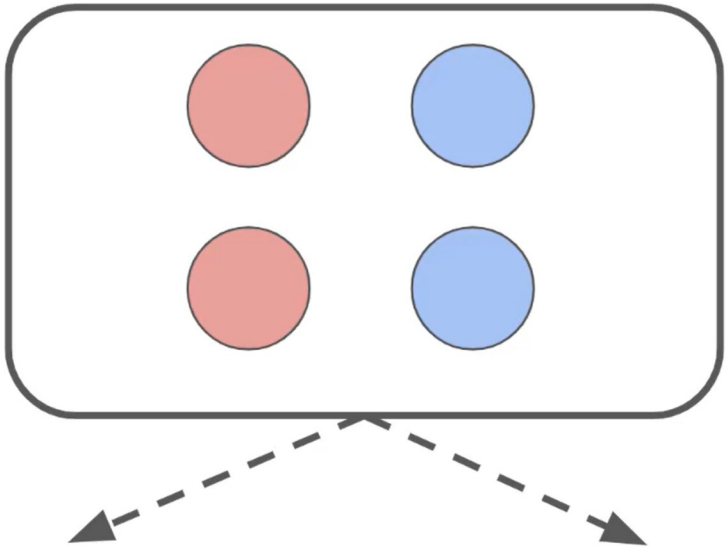


PIERIAN DATA

Деревья решений

- Метрика “gini impurity” для классификации

$$G(Q) = \sum_{c \in C} p_c(1 - p_c)$$



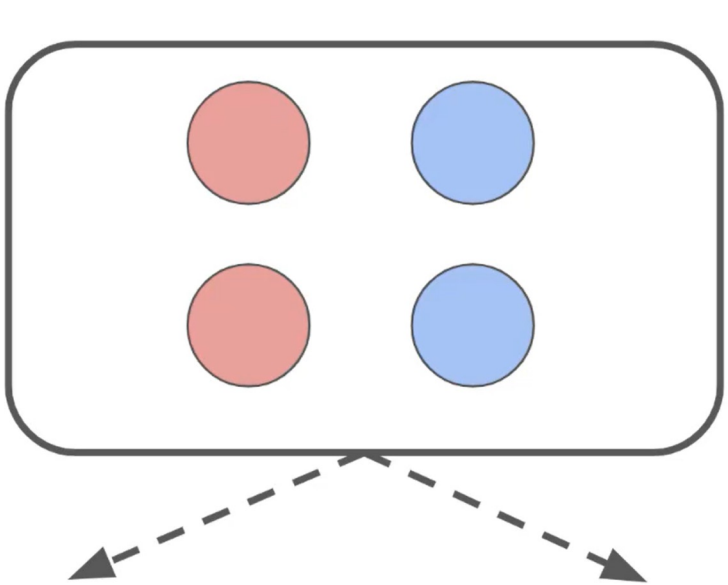
PIERIAN DATA

Деревья решений

- Метрика “gini impurity” для классификации

$$G(Q) = \sum_{c \in C} p_c(1 - p_c)$$

Класс “красный”
(2/4)(1 - 2/4) = 0.25

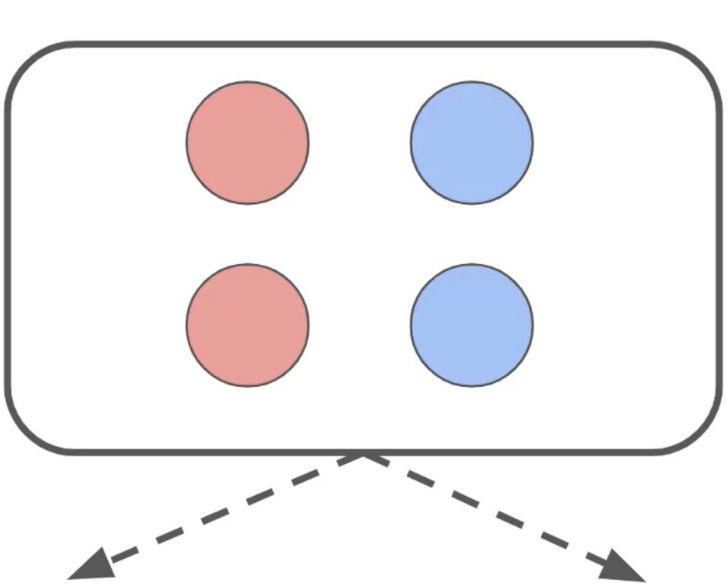


Деревья решений

- Метрика “gini impurity” для классификации

$$G(Q) = \sum_{c \in C} p_c(1 - p_c)$$

Класс “красный”
(2/4)(1 - 2/4) = 0.25

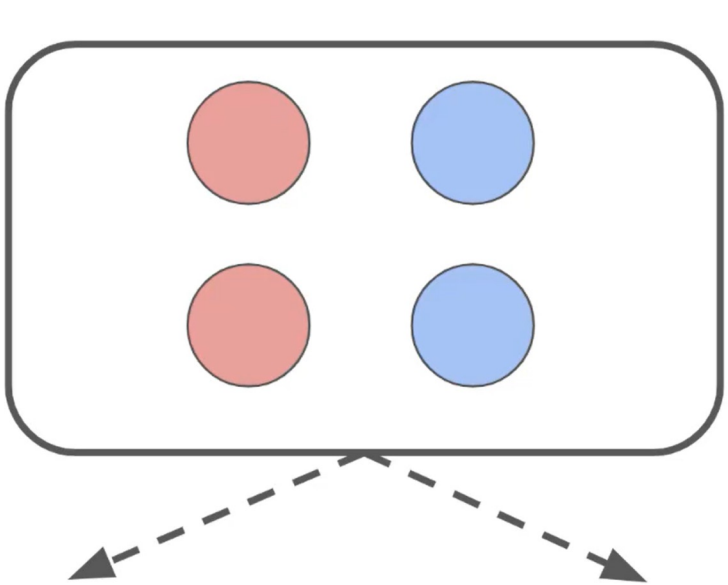


Класс “синий”
(2/4)(1 - 2/4) = 0.25

Деревья решений

- Метрика “gini impurity” для классификации

$$G(Q) = \sum_{c \in C} p_c(1 - p_c)$$



Класс “красный”
(2/4)(1 - 2/4) = 0.25

Класс “синий”
(2/4)(1 - 2/4) = 0.25

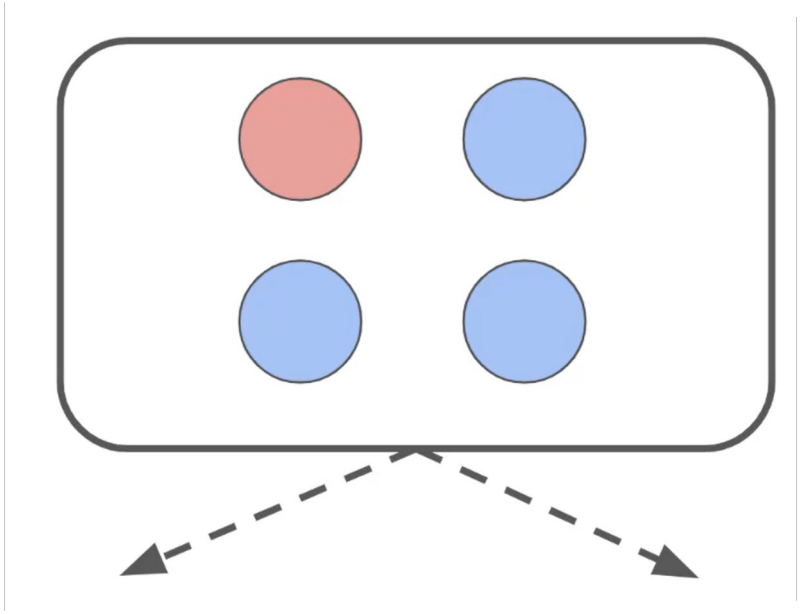
Gini Impurity
0.25 + 0.25 = 0.5

PIERIAN DATA

Деревья решений

- Данные более чистые (меньше “impurity”)

$$G(Q) = \sum_{c \in C} p_c(1 - p_c)$$



Класс “красный”
(1/4)(1 - 1/4) = 0.1875

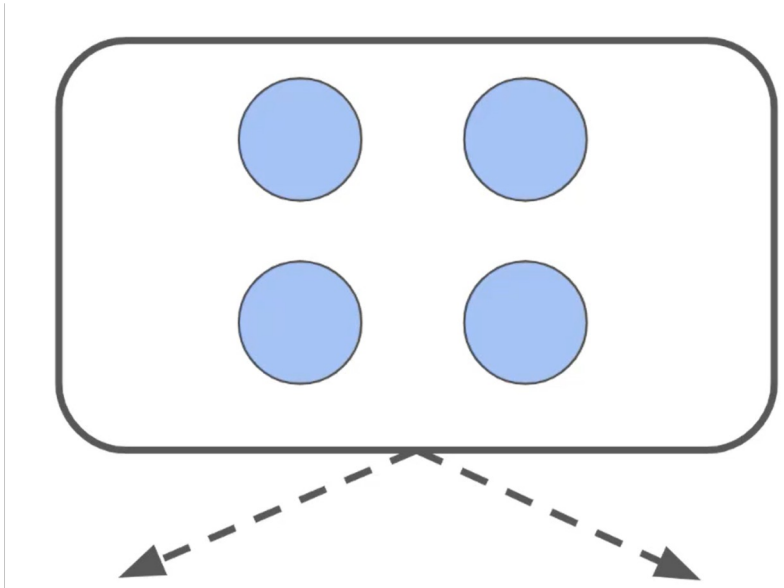
Класс “синий”
(3/4)(1 - 3/4) =0.1875

Gini Impurity
0.1875+0.1875 = 0.375

Деревья решений

- Данные полностью чистые (нулевой “impurity”)

$$G(Q) = \sum_{c \in C} p_c(1 - p_c)$$



Класс “красный”
(0/4)(1 - 0/4) = 0

+

Класс “синий”
(4/4)(1 - 4/4) = 0

⇩

Gini Impurity
0 + 0 = 0

Деревья решений

- Если цель дерева решений в том, чтобы отделить классы друг от друга, то мы можем выполнять разбиения данных на основе метрики gini impurity.
- Мы хотим минимизировать gini impurity на листьях.
- Если на листьях gini impurity минимально, то это будет означать, что мы удачно разделили классы.

Деревья решений

- В следующей лекции мы построим пример применения gini impurity, вычисляя его на основе набора данных.
- Далее мы рассмотрим различные типы разбиения признаков и определения того, какой признак будет корневым узлом.

Деревья решений

Gini impurity в деревьях – часть 1

PIERIAN DATA

Деревья решений

- Начнём разбираться в том, как лучше выбрать порядок узлов и как выполнять разбиение внутри дерева.
- Для начала посмотрим, как построить дерево на основе некоторого набора данных, используя метрику gini impurity.

Деревья решений

- Начиная строить дерево, нам нужно выбрать признак, который будет использоваться для корневого узла.
- Мы можем применить метрику gini impurity для сравнения информации, содержащейся внутри признаков в наборе данных.
- Рассмотрим этот момент подробнее...

Деревья решений

- Метрика “gini impurity” для классификации
  - Для набора классов C и заданного набора данных Q:

$$G(Q) = \sum_{c \in C} p_c(1 - p_c)$$

PIERIAN DATA

Деревья решений

- Метрика “gini impurity” для классификации
  - Для набора классов C и заданного набора данных Q, p_c это вероятность класса **c.**

$$p_c = \frac{1}{N_Q} \sum_{x \in Q} \mathbb{1}(y_{class} = c)$$

$$G(Q) = \sum_{c \in C} p_c(1 - p_c)$$

Деревья решений

- Метрика “gini impurity” для классификации
  - Для набора классов C и заданного набора данных Q, $\mathbf{p_c}$ это вероятность класса **c.**

$$p_c = \frac{1}{N_Q} \sum_{x \in Q} \mathbb{1}(y_{class} = c)$$

$$G(Q) = \sum_{c \in C} p_c (1 - p_c)$$

Деревья решений

- Рассмотрим следующий набор данных:

X - URL Link	Y-Spam
Yes	Yes
Yes	Yes
No	No
No	No
No	Yes
No	No
Yes	No

Деревья решений

- Попытаемся спрогнозировать – спам или нет:

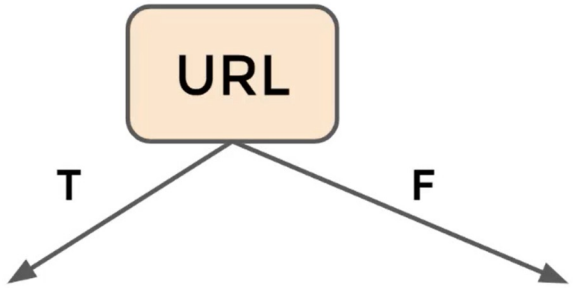
X - URL Link	Y-Spam
Yes	Yes
Yes	Yes
No	No
No	No
No	Yes
No	No
Yes	No

PIERIAN DATA

Деревья решений

- Является ли письмо спамом, если в нём есть URL:

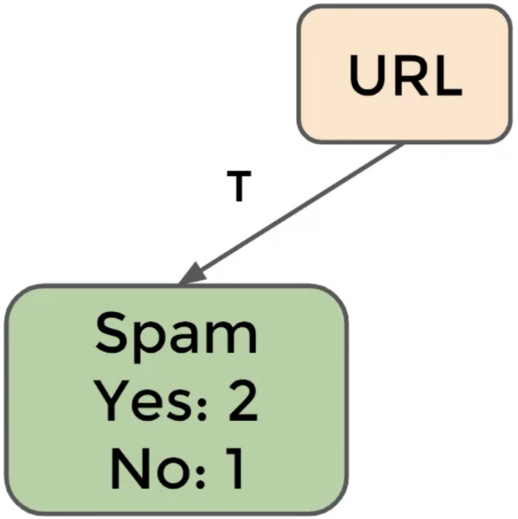
X - URL Link	Y-Spam
Yes	Yes
Yes	Yes
No	No
No	No
No	Yes
No	No
Yes	No



Деревья решений

- Является ли письмо спамом, если в нём есть URL:

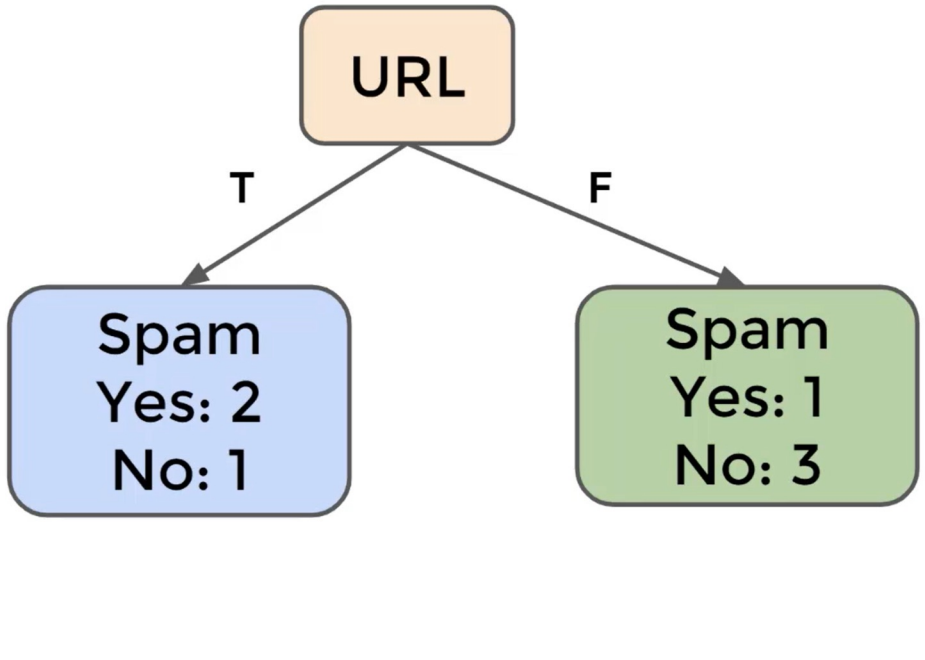
X - URL Link	Y-Spam
Yes	Yes
Yes	Yes
No	No
No	No
No	Yes
No	No
Yes	No



Деревья решений

- Является ли письмо спамом, если в нём есть URL:

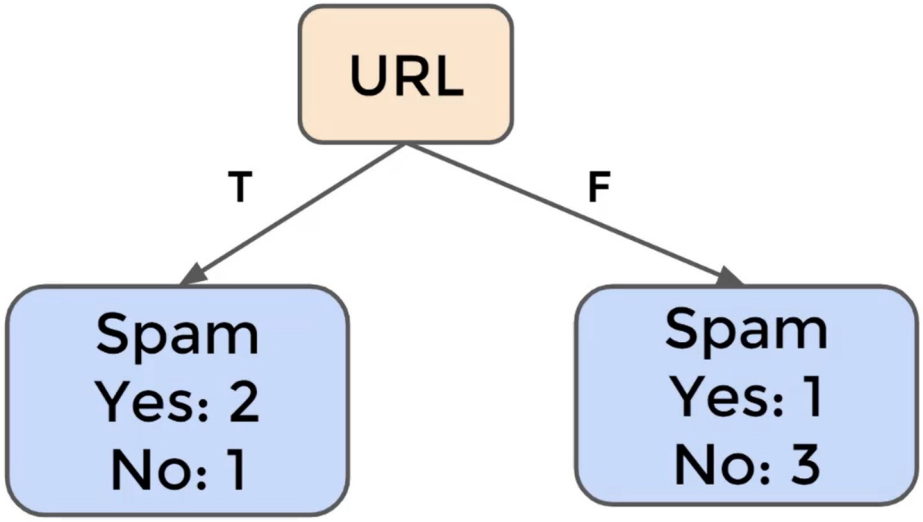
X - URL Link	Y-Spam
Yes	Yes
Yes	Yes
No	No
No	No
No	Yes
No	No
Yes	No



Деревья решений

- Является ли письмо спамом, если в нём есть URL:

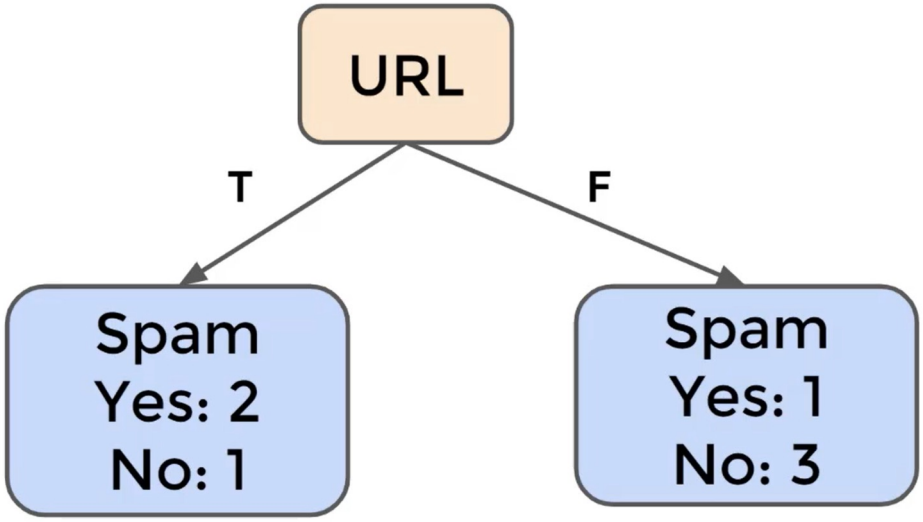
X - URL Link	Y-Spam
Yes	Yes
Yes	Yes
No	No
No	No
No	Yes
No	No
Yes	No



Деревья решений

- Вспомним формулу gini impurity:

X - URL Link	Y-Spam
Yes	Yes
Yes	Yes
No	No
No	No
No	Yes
No	No
Yes	No

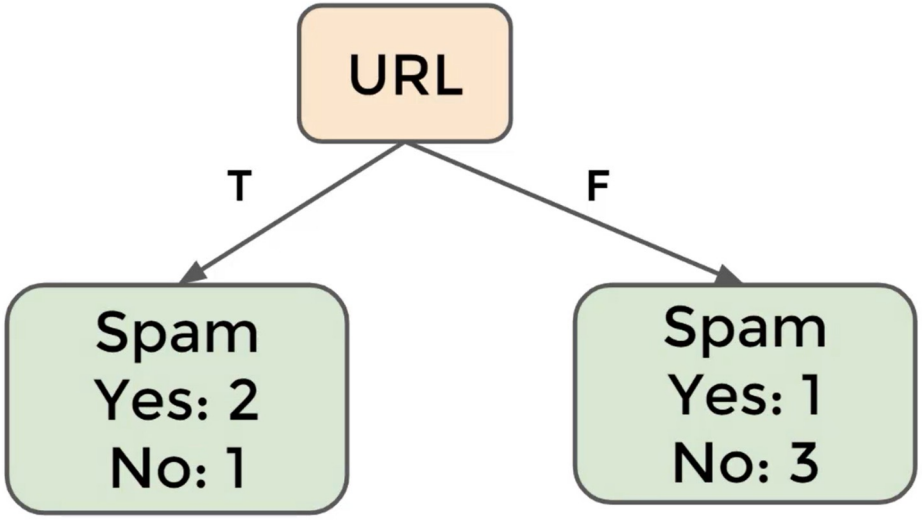


$$G(Q) = \sum_{c \in C} p_c(1 - p_c)$$

Деревья решений

- Пусть “спам” и “не-спам” будут классами C
- Левый листовой узел:

$(2/3)(1-2/3)$

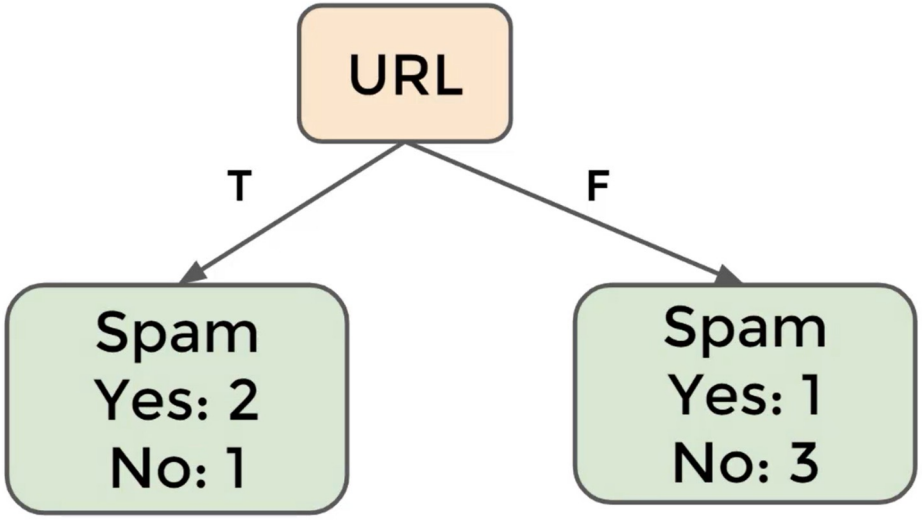


$$G(Q) = \sum_{c \in C} p_c(1 - p_c)$$

Деревья решений

- Пусть “спам” и “не-спам” будут классами C
- Левый листовой узел:

$$(2/3)(1-2/3) + (1/3)(1-1/3)$$



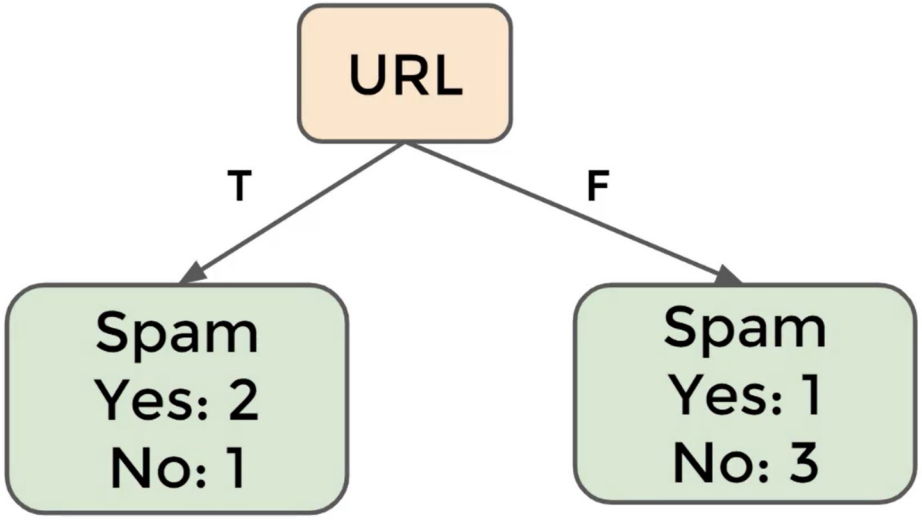
$$G(Q) = \sum_{c \in C} p_c(1 - p_c)$$

Деревья решений

- Пусть “спам” и “не-спам” будут классами C
- Левый листовой узел:

$$(2/3)(1-2/3) + (1/3)(1-1/3)$$

$$\text{Gini}=0.44$$



$$G(Q) = \sum_{c \in C} p_c(1 - p_c)$$

Деревья решений

- Пусть “спам” и “не-спам” будут классами C
- Левый листовой узел:

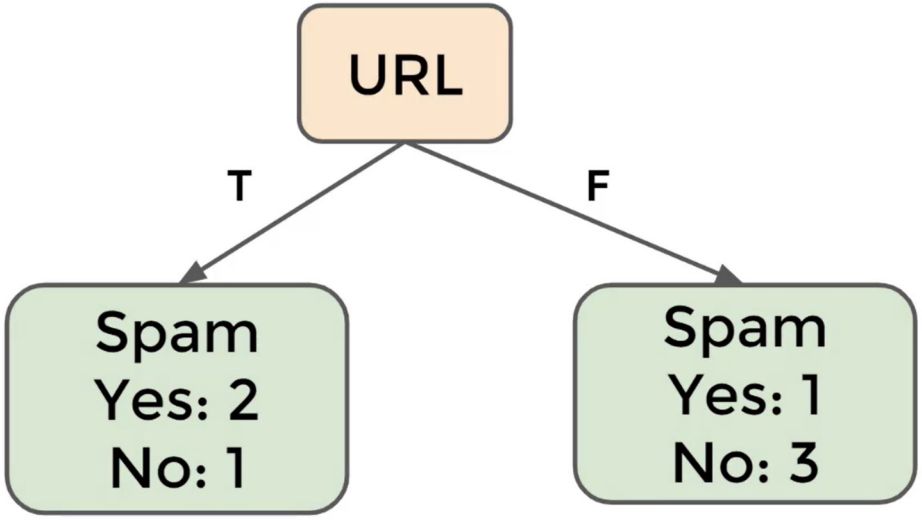
$$(2/3)(1-2/3) + (1/3)(1-1/3)$$

Gini=0.44

- Правый листовой узел:

$$(1/4)(1-1/4) + (3/4)(1-3/4)$$

Gini=0.375

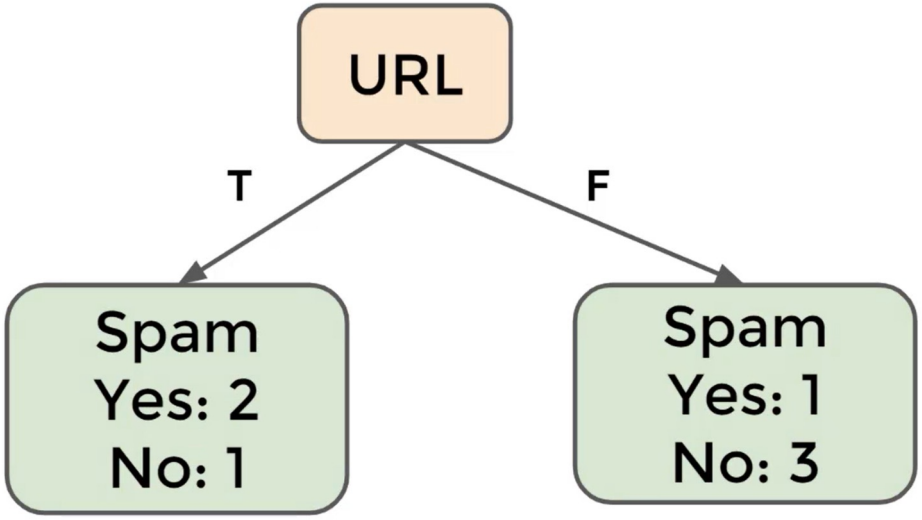


$$G(Q) = \sum_{c \in C} p_c(1 - p_c)$$

PIERIAN DATA

Деревья решений

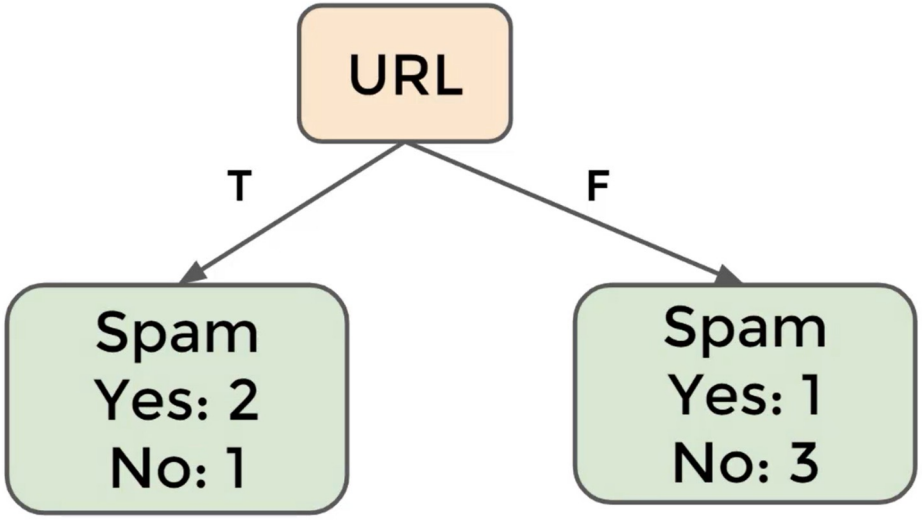
- Итак, для признака URL метрика gini impurity:
- Левый лист: Gini=0.44
- Правый лист: Gini=0.375



$$G(Q) = \sum_{c \in C} p_c(1 - p_c)$$

Деревья решений

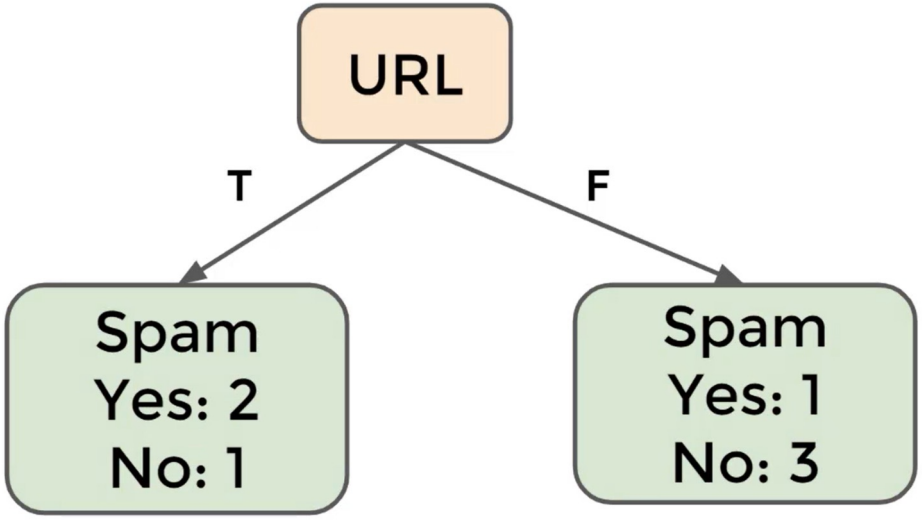
- Всего писем: (2+1) + (1+3) = 7
- Левый лист: Gini=0.44
- Правый лист: Gini=0.375



$$G(Q) = \sum_{c \in C} p_c(1 - p_c)$$

Деревья решений

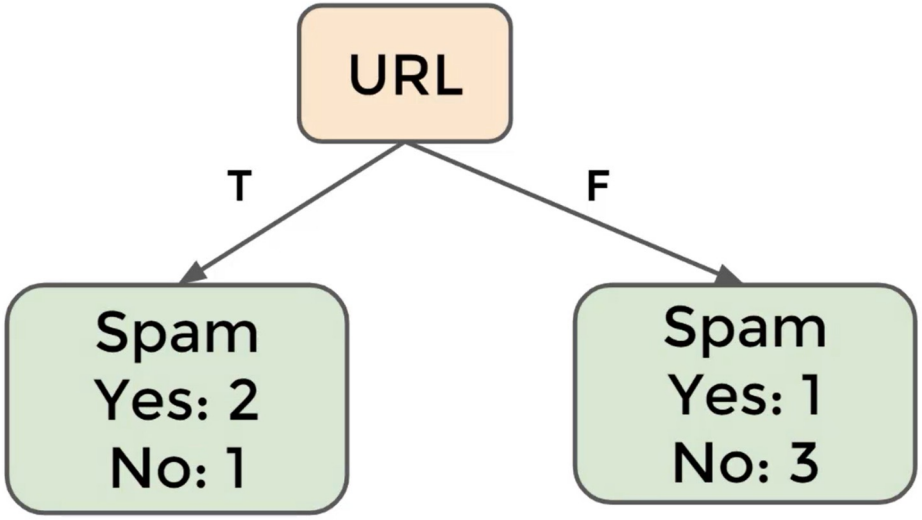
- Всего писем: (2+1) + (1+3) = 7
- Левый лист: Gini=0.44
- Правый лист: Gini=0.375
- Писем слева: 3
- Писем справа: 4



$$G(Q) = \sum_{c \in C} p_c(1 - p_c)$$

Деревья решений

- Всего писем: (2+1) + (1+3) = 7
- Левый лист: Gini=0.44
- Правый лист: Gini=0.375
- Писем слева: 3
- Писем справа: 4
- (3/7)*0.44+(4/7)*0.375
- Gini Impurity = 0.403

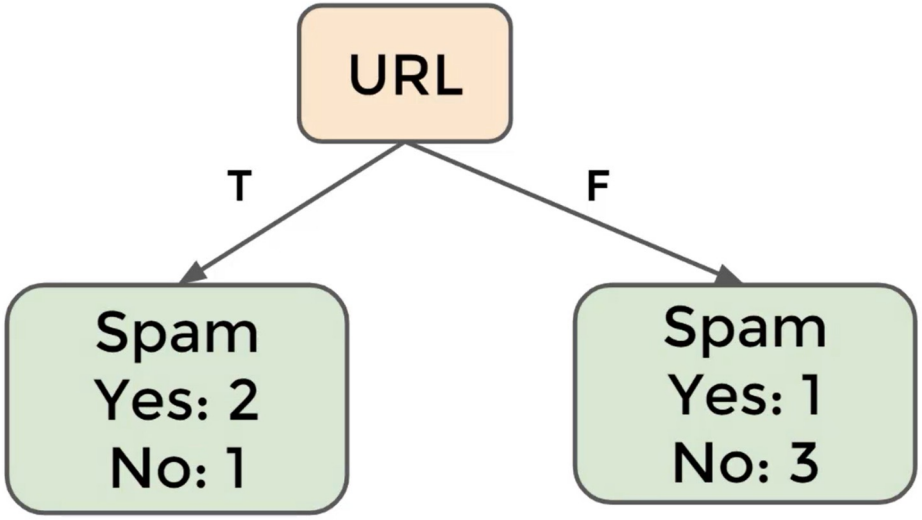


$$G(Q) = \sum_{c \in C} p_c(1 - p_c)$$

PIERIAN DATA

Деревья решений

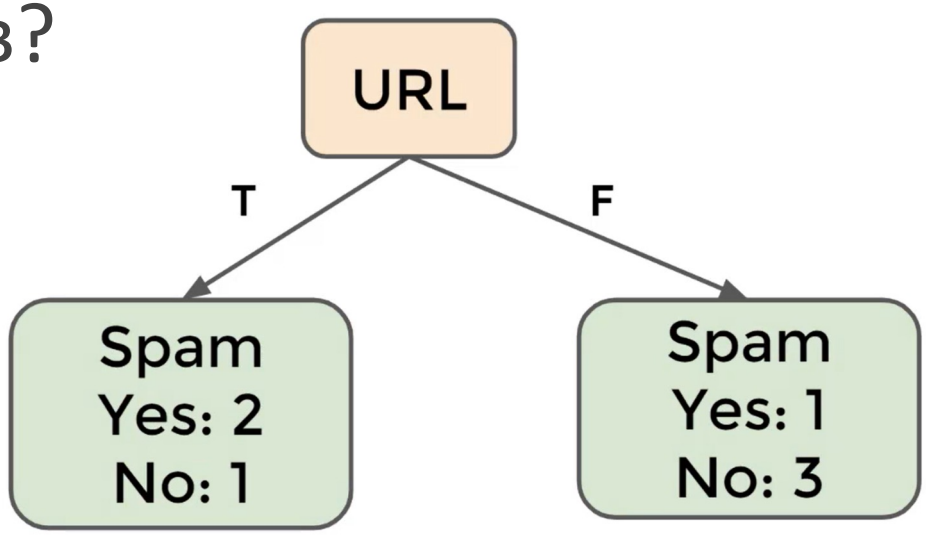
- Gini Impurity для признака URL равна 0.403



$$G(Q) = \sum_{c \in C} p_c(1 - p_c)$$

Деревья решений

- Gini Impurity для признака URL равна 0.403
- А если несколько признаков?



$$G(Q) = \sum_{c \in C} p_c(1 - p_c)$$

Деревья решений

- Нам нужно разобрать ещё несколько вопросов:
  - Несколько признаков
  - Непрерывные признаки
  - Мульти-категориальные признаки
- Мы можем применить gini impurity для каждого из этих случаев, чтобы найти наилучший корневой узел и наилучшие параметры разбиения для листьев.

Деревья решений

Gini impurity в деревьях – часть 2

PIERIAN DATA

Деревья решений

- Ранее мы узнали, как вычислять gini impurity для бинарного категориального признака (состоящего только из двух категорий).
- Нам нужно разобрать ещё несколько вопросов:
  - Непрерывные признаки
  - Мульти-категориальные признаки (N>2)
  - Как выбрать корневой узел

Деревья решений

- Представим себе непрерывный признак:

X - Words in Email	Y-Spam
10	Yes
40	No
20	Yes
50	No
30	No

Деревья решений

- Вычислим метрику gini impurity

X - Words in Email	Y-Spam
10	Yes
40	No
20	Yes
50	No
30	No

Деревья решений

- Сначала отсортируем данные

X - Words in Email	Y-Spam
10	Yes
40	No
20	Yes
50	No
30	No

Деревья решений

- Вычислим возможные значения для разделения

X - Words in Email	Y-Spam
10	Yes
20	Yes
30	No
40	No
50	No

Words ≤ N

Деревья решений

- Возьмём средние значения “между” строками

X - Words in Email	Y-Spam
10	Yes
15	
20	Yes
25	
30	No
35	
40	No
45	
50	No

Words ≤ N

Деревья решений

- Вычислим gini impurity для каждого разбиения

X - Words in Email	Y-Spam
10	Yes
20	Yes
30	No
40	No
50	No

15

Words ≤ 15

Деревья решений

- Вычислим gini impurity для каждого разбиения

X - Words in Email	Y-Spam
10	Yes
20	Yes
30	No
40	No
50	No

15

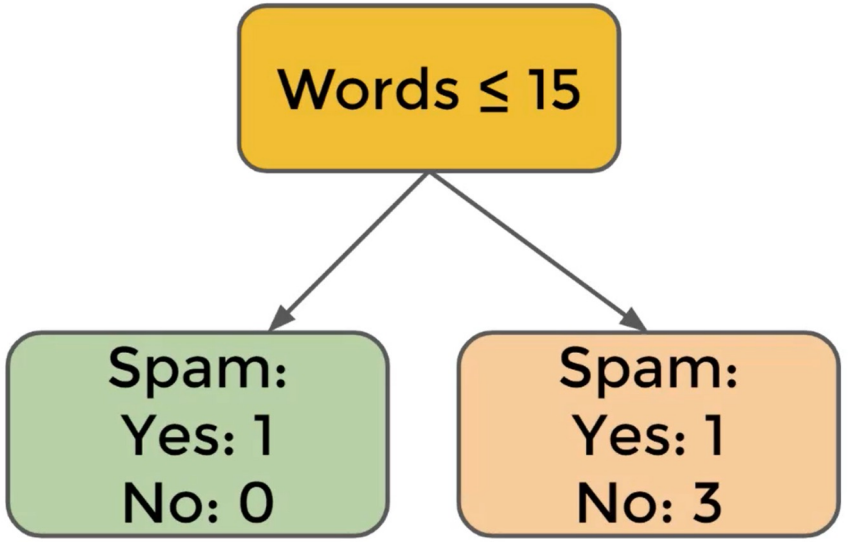
Words ≤ 15

Деревья решений

- Вычислим gini impurity для каждого разбиения

X - Words in Email	Y-Spam
10	Yes
20	Yes
30	No
40	No
50	No

15



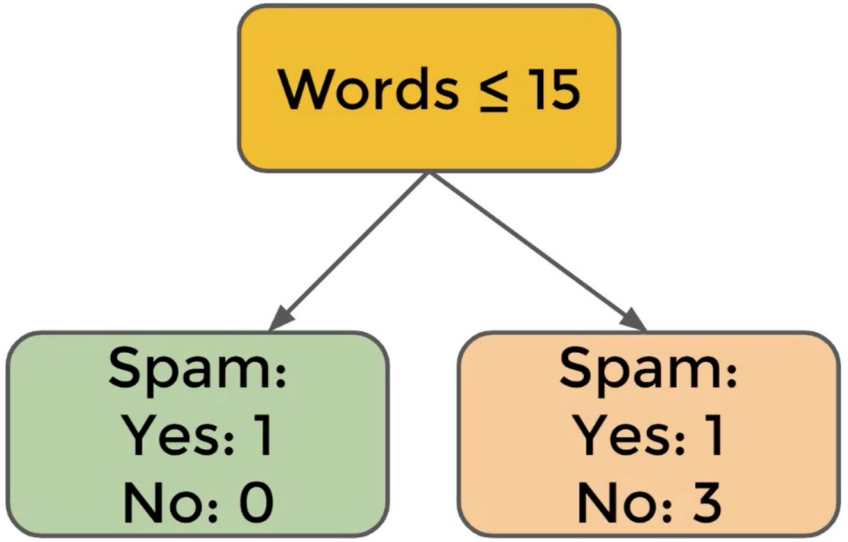
$$G(Q) = \sum_{c \in C} p_c(1 - p_c)$$

Деревья решений

- Вычислим gini impurity для каждого разбиения

X - Words in Email	Y-Spam
10	Yes
20	Yes
30	No
40	No
50	No

15



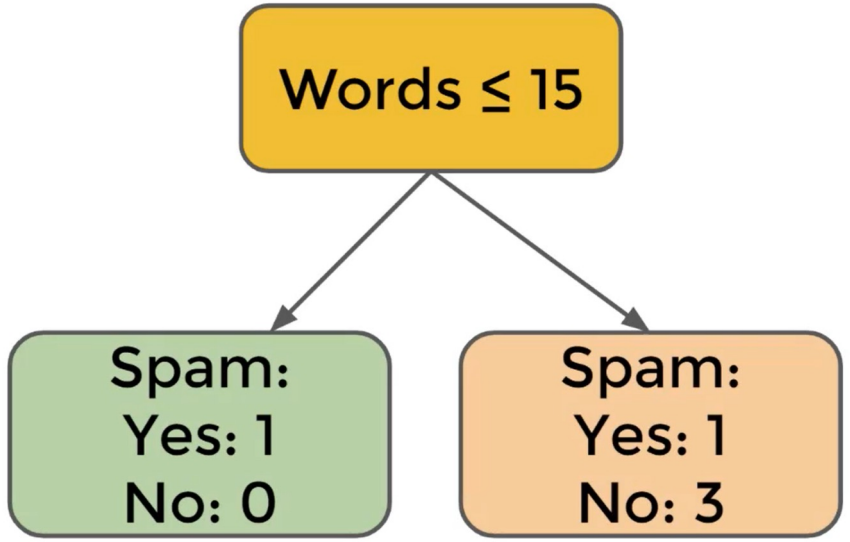
$$G(Q) = (1/5)(0+0) + (4/5)((1/4)(1-1/4)+(3/4)(1-3/4))$$

Деревья решений

- Вычислим gini impurity для каждого разбиения

X - Words in Email	Y-Spam
10	Yes
20	Yes
30	No
40	No
50	No

15



$$G(Q) = (1/5)(0+0) + (4/5)((1/4)(1-1/4)+(3/4)(1-3/4))$$

$$= 0.3$$

PIERIAN DATA

Деревья решений

- Вычислим gini impurity для каждого разбиения

X - Words in Email	Y-Spam
10	Yes
20	Yes
30	No
40	No
50	No

15 → Gini=0.3

PIERIAN DATA

Деревья решений

- Вычислим gini impurity для каждого разбиения

X - Words in Email	Y-Spam
10	Yes
20	Yes
30	No
40	No
50	No

15 → Gini=0.3

25 → Gini=0

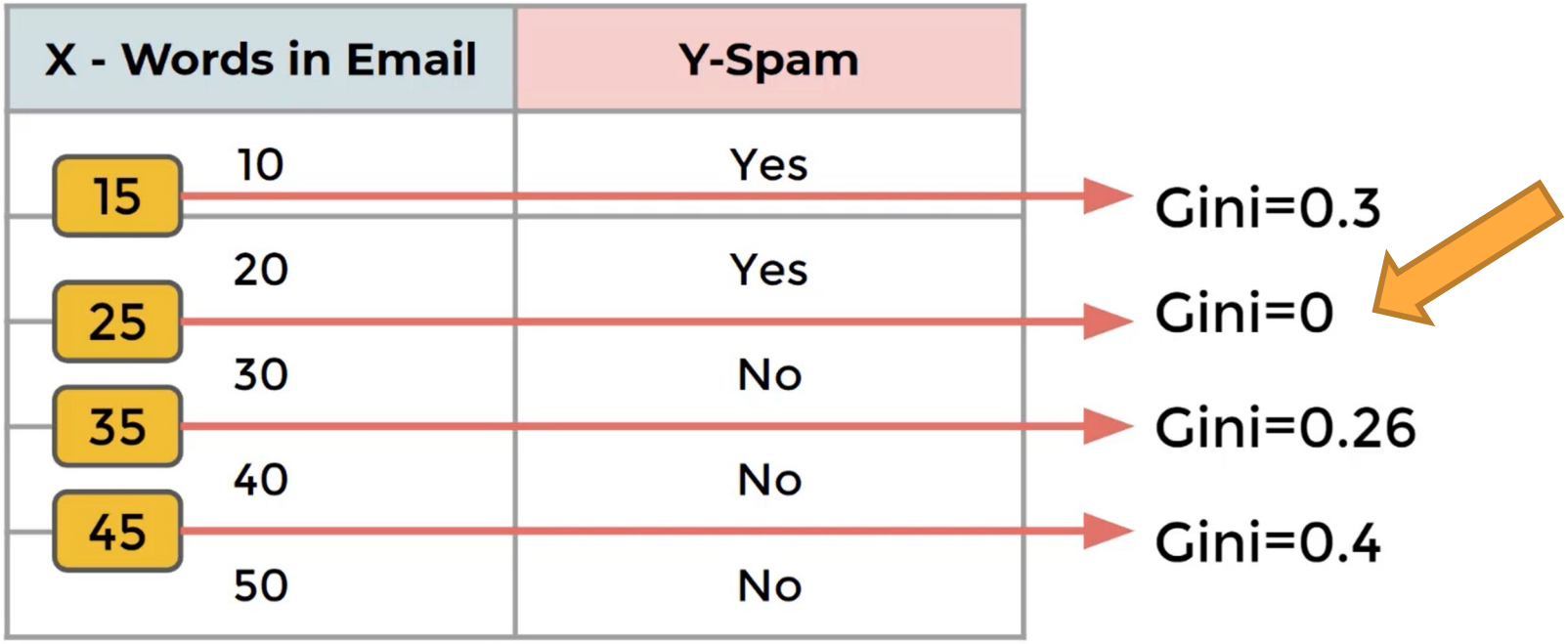
35 → Gini=0.26

45 → Gini=0.4

PIERIAN DATA

Деревья решений

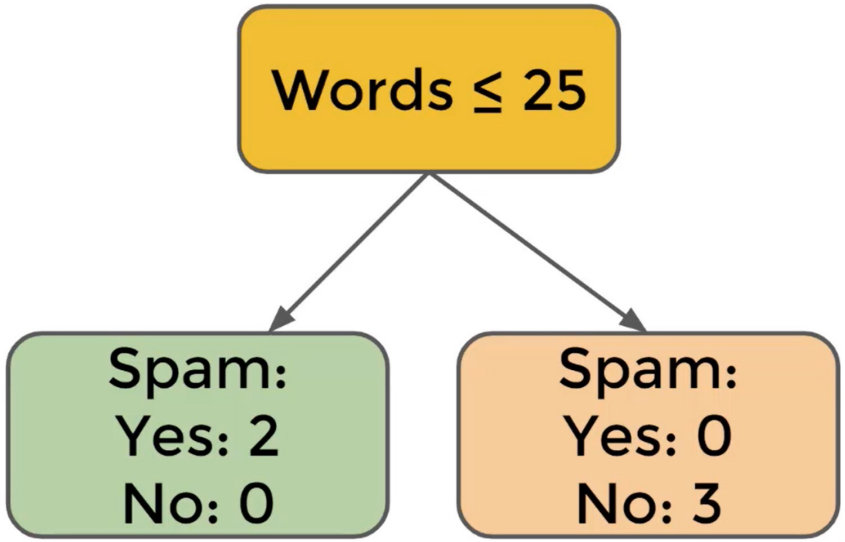
- Выбираем наименьший gini impurity



Деревья решений

- Это значение будет разделителем для узла

X - Words in Email	Y-Spam
10	Yes
20	Yes
25	
30	No
40	No
50	No



$$G(Q) = 0$$

Деревья решений

- Мы вычислили gini impurity для следующих случаев:
  - Бинарные признаки
  - Непрерывные признаки
- Далее давайте вычислим эту метрику для мульти-категориальных признаков

Деревья решений

- Мульти-категориальный признак:

X - Sender	Y-Spam
Abe	Yes
Bob	Yes
Claire	No
Abe	No
Bob	No

Деревья решений

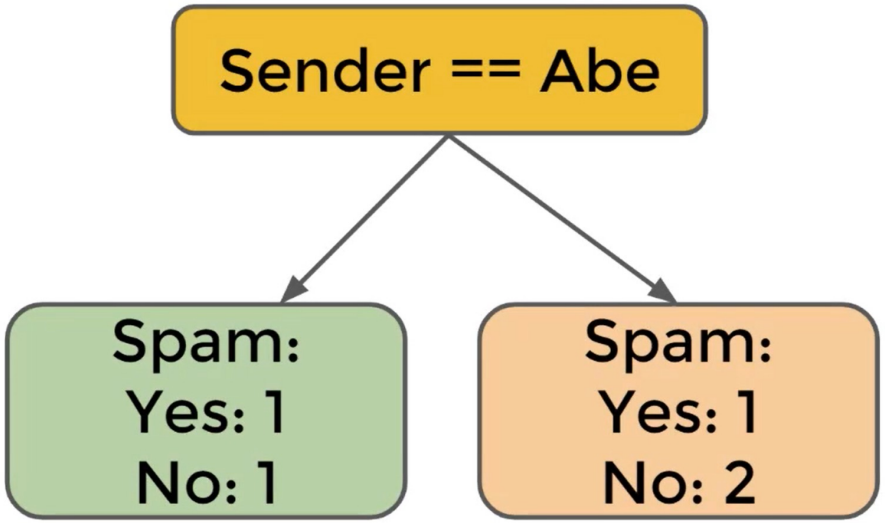
- Вычислим gini impurity для всех комбинаций

X - Sender	Y-Spam
Abe	Yes
Bob	Yes
Claire	No
Abe	No
Bob	No

Деревья решений

- Вычислим gini impurity для всех комбинаций

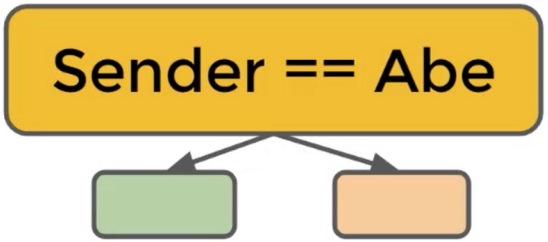
X - Sender	Y-Spam
Abe	Yes
Bob	Yes
Claire	No
Abe	No
Bob	No



Деревья решений

- Вычислим gini impurity для всех комбинаций

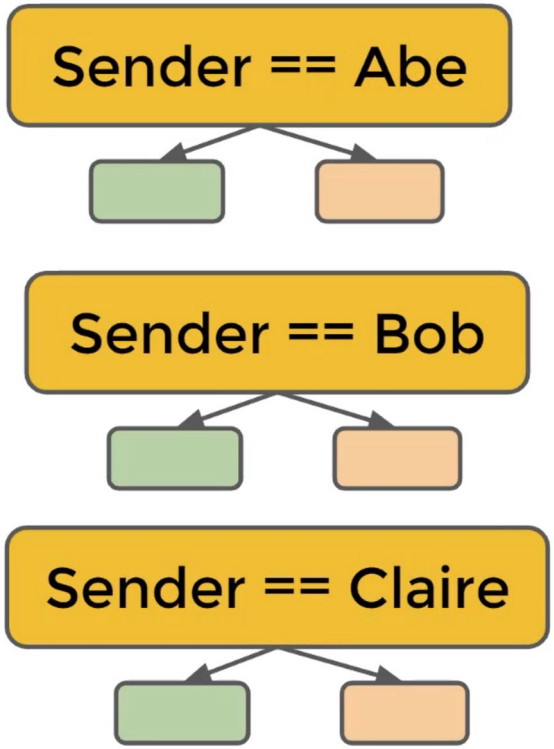
X - Sender	Y-Spam
Abe	Yes
Bob	Yes
Claire	No
Abe	No
Bob	No



Деревья решений

- Вычислим gini impurity для всех комбинаций

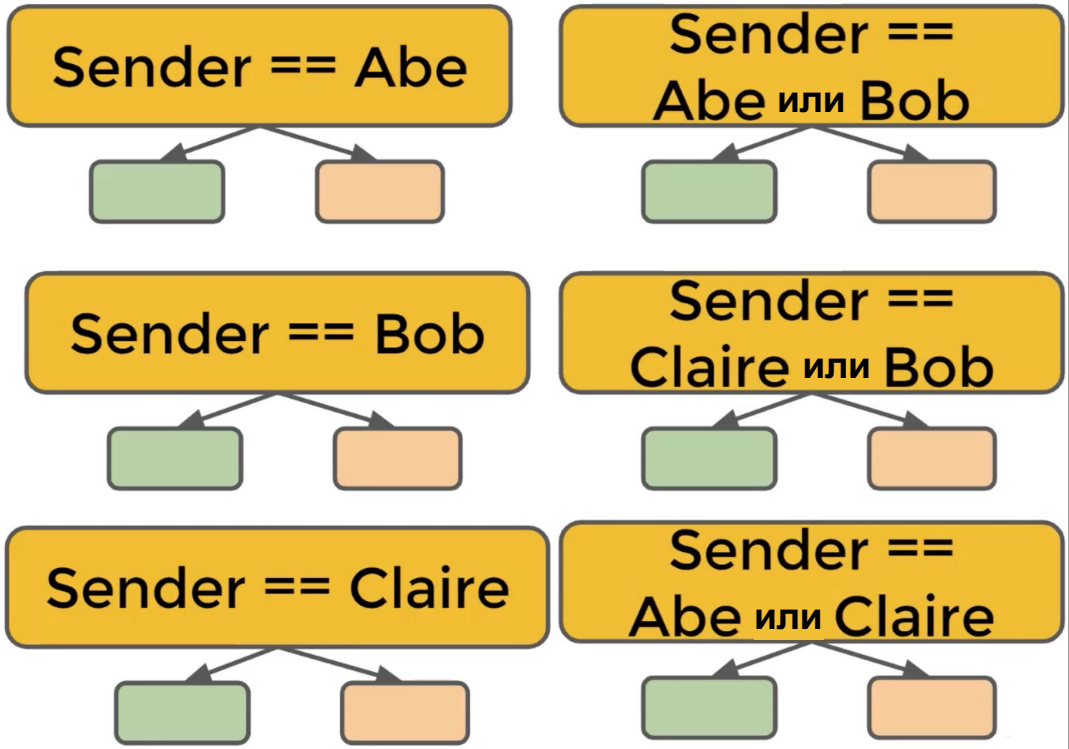
X - Sender	Y-Spam
Abe	Yes
Bob	Yes
Claire	No
Abe	No
Bob	No



Деревья решений

- Вычислим gini impurity для всех комбинаций

X - Sender	Y-Spam
Abe	Yes
Bob	Yes
Claire	No
Abe	No
Bob	No



PIERIAN DATA

Деревья решений

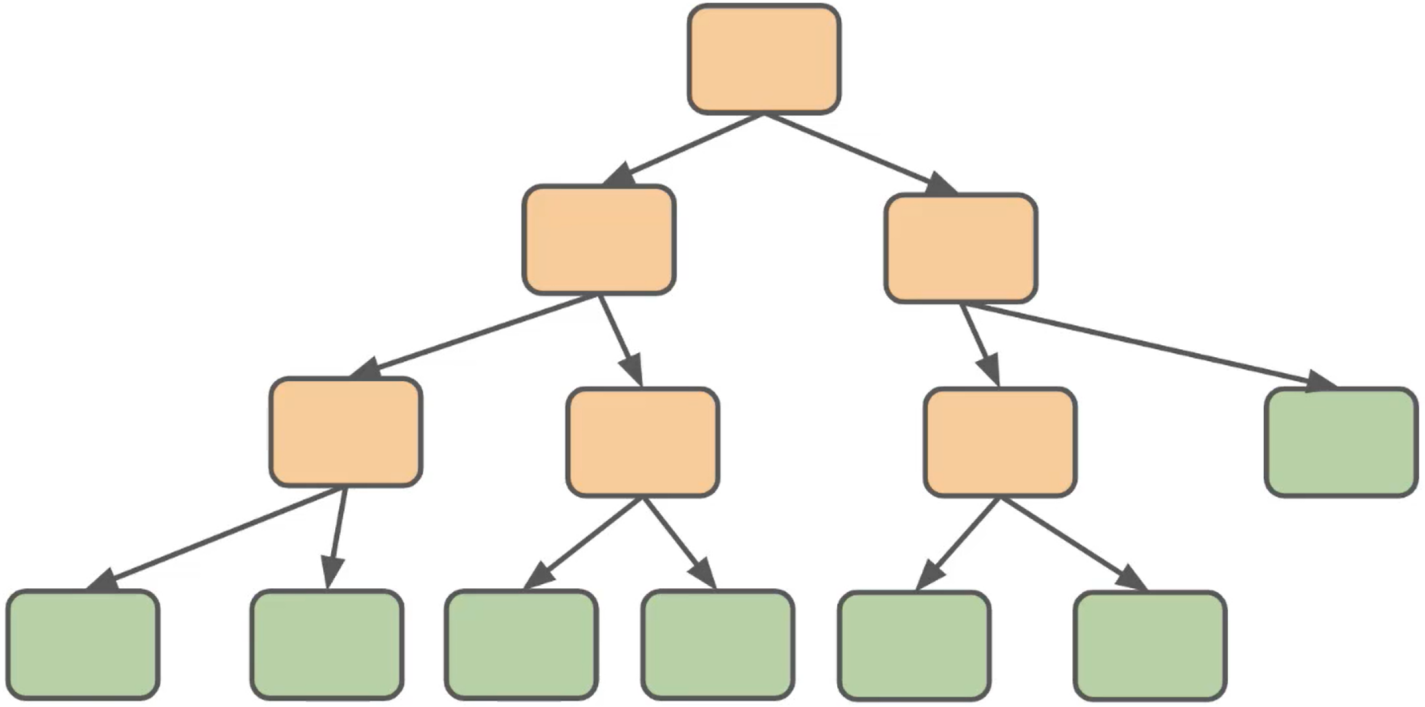
- Теперь мы умеем вычислить gini impurity для различных типов признаков.
- Но как выбрать признак для корневого узла, когда признаков несколько?
- Нужно вычислить gini impurity для каждого признака, выбрать тот где метрика наименьшая, и взять его первым.

Деревья решений

- Мы хотим минимизировать gini impurity на листьях.
- Оставляем листья, когда impurity меньше некоторого порогового значения.
- Выполняем разбиение только в тех случаях, когда impurity больше некоторого порогового значения.

Деревья решений

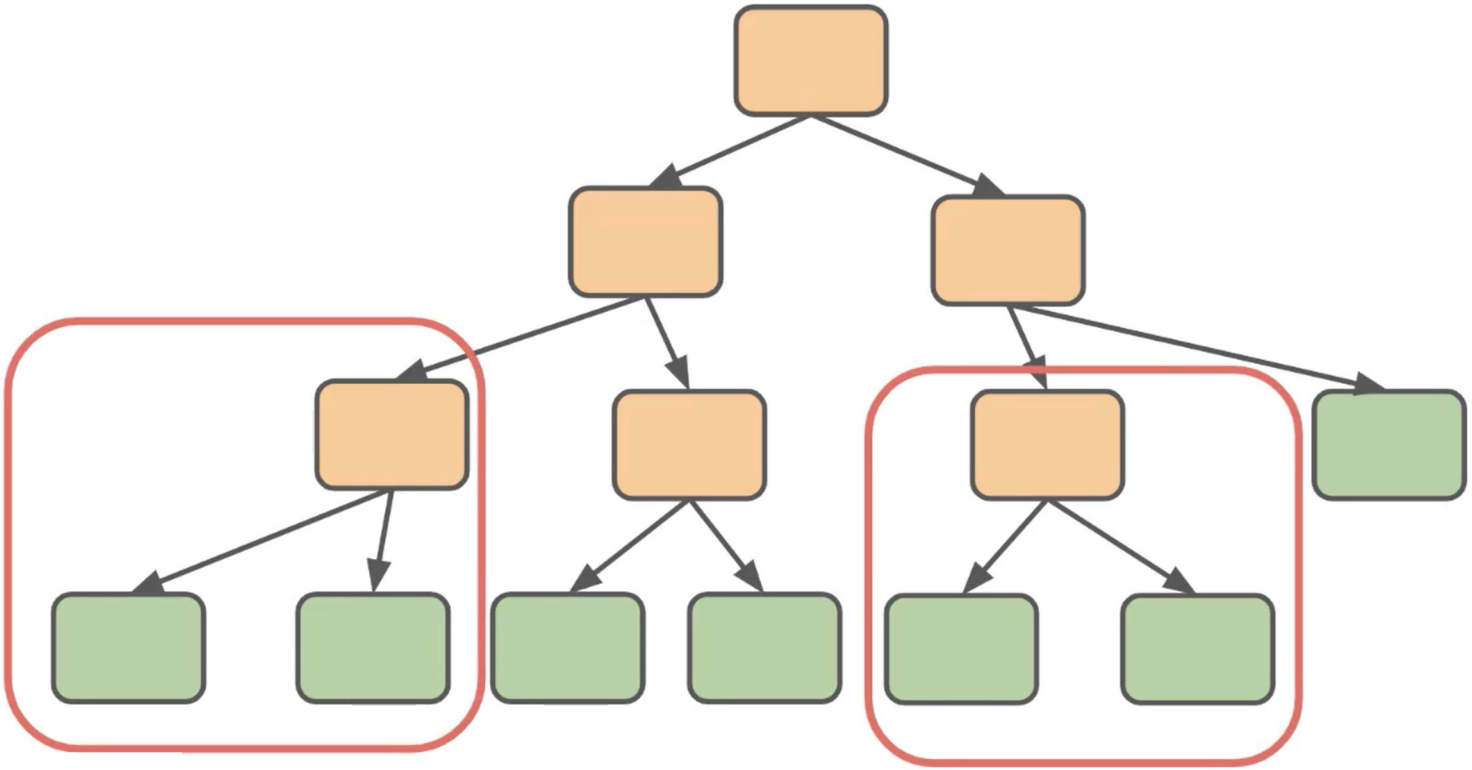
- Слишком большое дерево (“overfitted”):



PIERIAN DATA

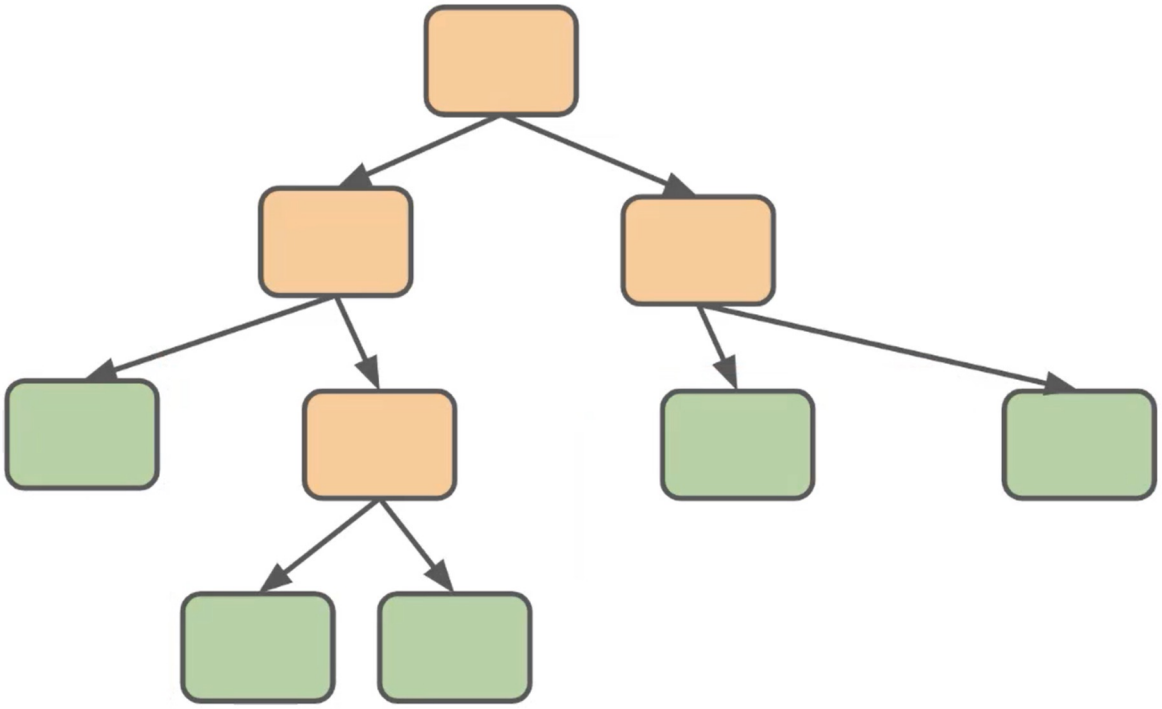
Деревья решений

- Добавляем порог уменьшения для gini impurity:



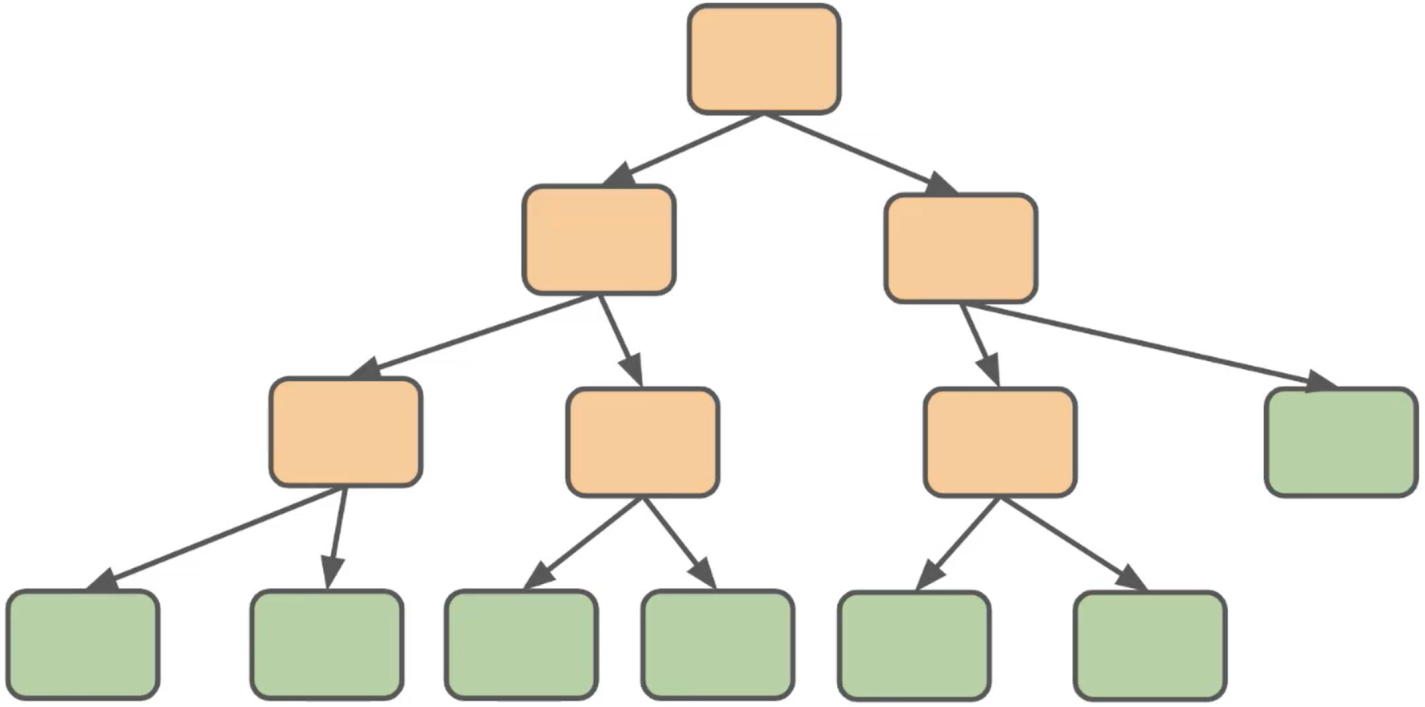
Деревья решений

- Добавляем порог уменьшения для gini impurity:



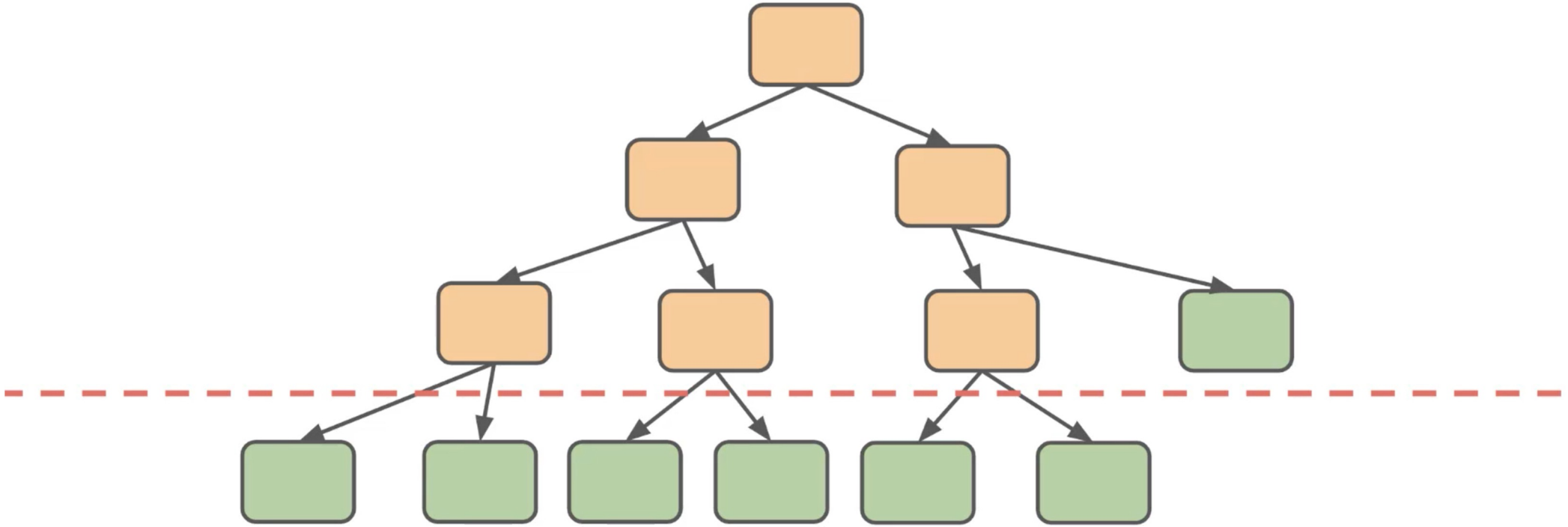
Деревья решений

- Ещё мы можем установить максимальную глубину



Деревья решений

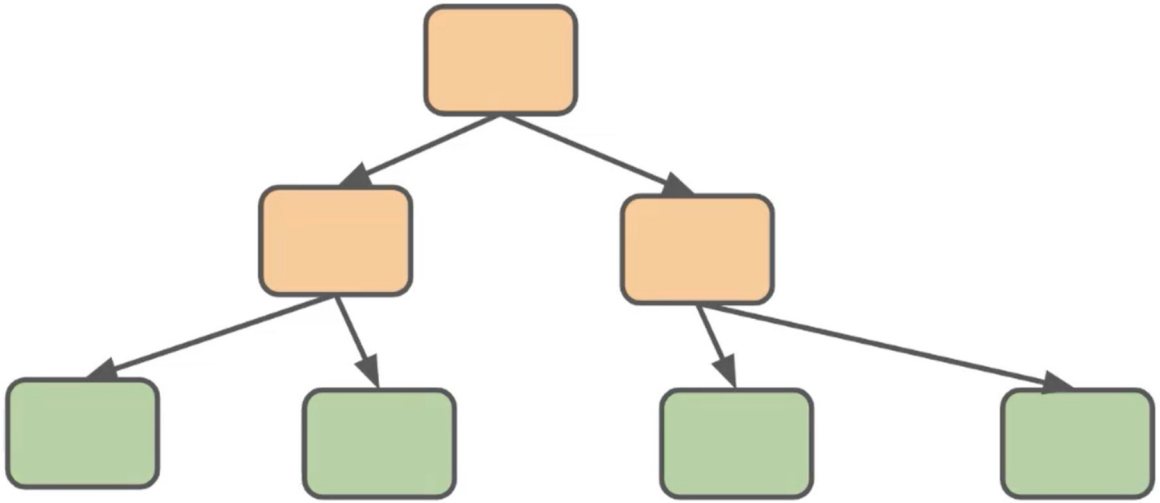
- Ещё мы можем установить максимальную глубину



PIERIAN DATA

Деревья решений

- Ещё мы можем установить максимальную глубину



Деревья решений

- Посмотрим различные гиперпараметры во время написания кода!

Деревья решений

Пишем код – часть 1 - данные

PIERIAN DATA

Деревья решений

Пишем код – часть 2 - модель

PIERIAN DATA