March 30, Kyiv

Data Science & Mathematical Modeling Bachelor Program

Course “Basics of Machine Learning”

Lecture 4: Decision trees

Oleg CHERTOV

Professor, Sc.D. (Doctor Habilitatus),
Head of the Applied Mathematics Department

Applied Mathematics Department
Igor Sikorsky Kyiv Polytechnic Institute
Ukraine

Гра «двадцять питань»

- Я задумаю визначну особистість, а ви повинні за 20 питань вгадати, кого я загадав

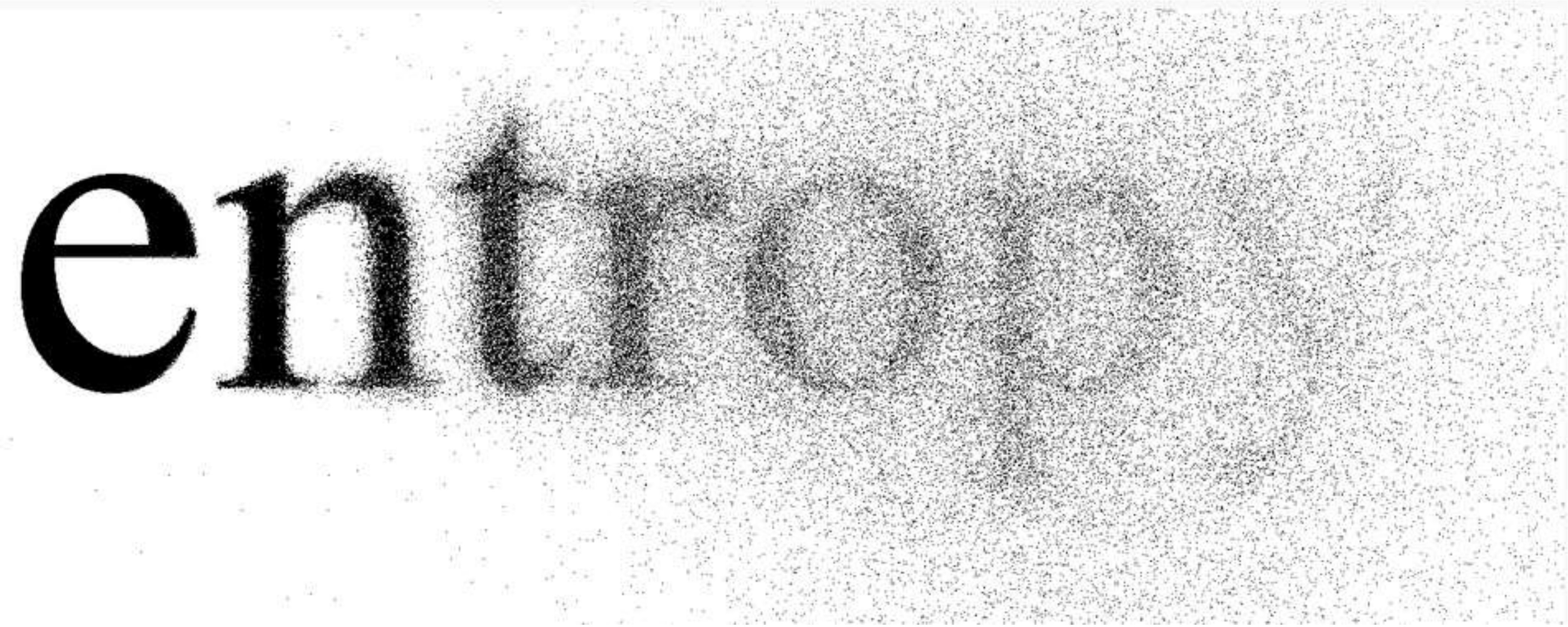
Гра «двадцять питань»

- Я задумаю визначну особистість, а ви повинні за 20 питань вгадати, кого я загадав

- Важлива здатність питання відсіяти якомога більше невірних відповідей
- Це інтуїтивно відповідає поняттю приросту інформації, яке базується на ентропії

How much information you're missing

- Entropy is how much information you're missing
- Ентропія — це те, як багато інформації вам не відомо про систему



Ентропія Шеннона

- Ентропія Шеннона визначається для системи з N можливими станами наступним чином:

$$S = -\sum_{i=1}^{N} p_i \log_2 p_i \,,$$

Зазвичай, нові поняття вводяться так, щоб вони приймали додатні значення.
А у нас у формулі для підрахунку ентропії стоїть мінус.
Чому?

де p_i – ймовірність знаходження системи в i-му стані

- Ентропія відповідає ступеню хаосу в системі
- Чим вище ентропія, тим менше впорядкована система і навпаки

Ентропія Шеннона

- Якщо всі записи належать одному класу, то
вектор ймовірностей: (1, 0, 0, …, 0) і

$$\boldsymbol{E(D) = \sum_{i=1}^{k}(-p_i \log_2 p_i) = -p_1 \log_2 p_1 = -1 * \log_2 1 = 0}$$

- Якщо всі записи рівномірно розподілені по всім k класам, то вектор ймовірностей: (1/k, 1/k, …, 1/k) і

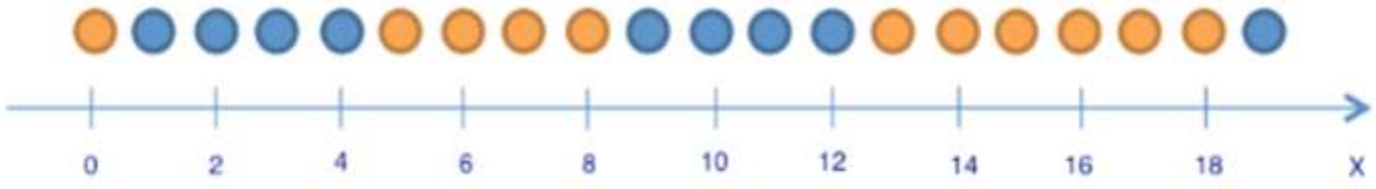
$$\boldsymbol{E(D) = \sum_{i=1}^{k} -(1/k) \log_2 (1/k) = -\log_2 (1/k) = -(0 - \log_2 k) = \log_2 k}$$

- $\boldsymbol{\log_2 k}$ – як відомо із теорії інформації, це максимальне значення ентропії

- Чим вище ентропія, тим більш неоднорідна та менш впорядкована система і навпаки

Дерево прийняття рішень. Іграшковий приклад - 1

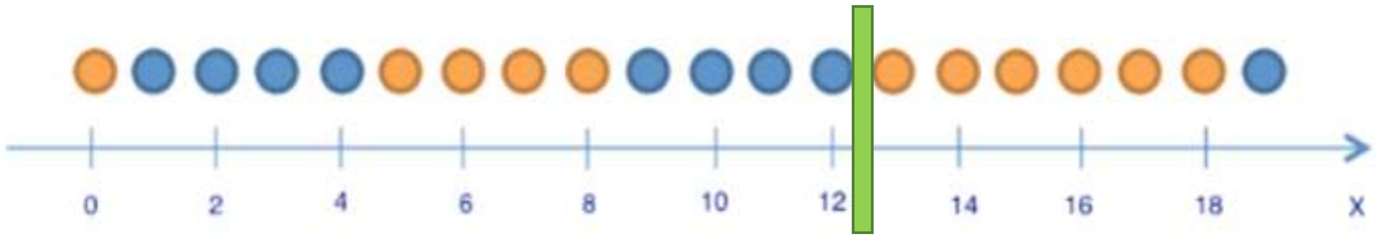
- Потрібно визначити – кулька якого кольору знаходиться на конкретному місці, задавши мінімальну кількість питань



- Якщо задавати «погані» питання – чи є кулька на *і*-му місці синьою, то потрібно 20 таких питань
- Потрібні питання типу – чоловік чи жінка, живий чи мертвий тощо
- Давайте спробуємо розділити цю послідовність кульок на дві підпослідовності – найбільш оптимальні для подальшої класифікації цих кульок
- Які варіанти розділення на дві частини здаються прийнятними?

Дерево прийняття рішень. Іграшковий приклад - 2

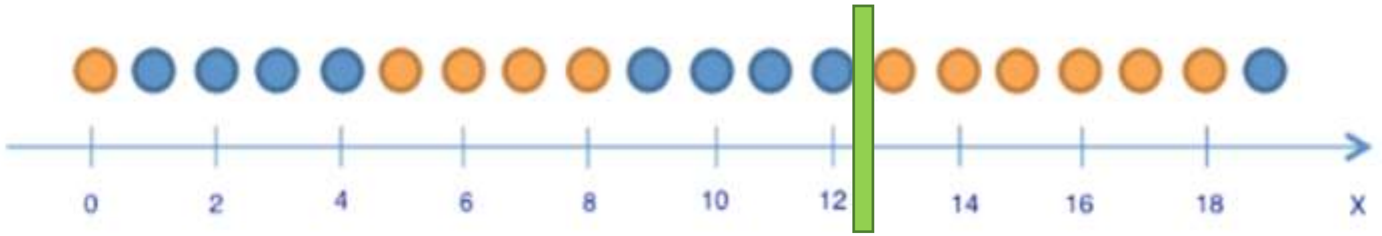
- Потрібно визначити – кулька якого кольору знаходиться на конкретному місці, задавши мінімальну кількість питань



- Які варіанти розділення на дві частини здаються прийнятними?

Дерево прийняття рішень. Іграшковий приклад - 3

- Потрібно визначити – кулька якого кольору знаходиться на конкретному місці, задавши мінімальну кількість питань

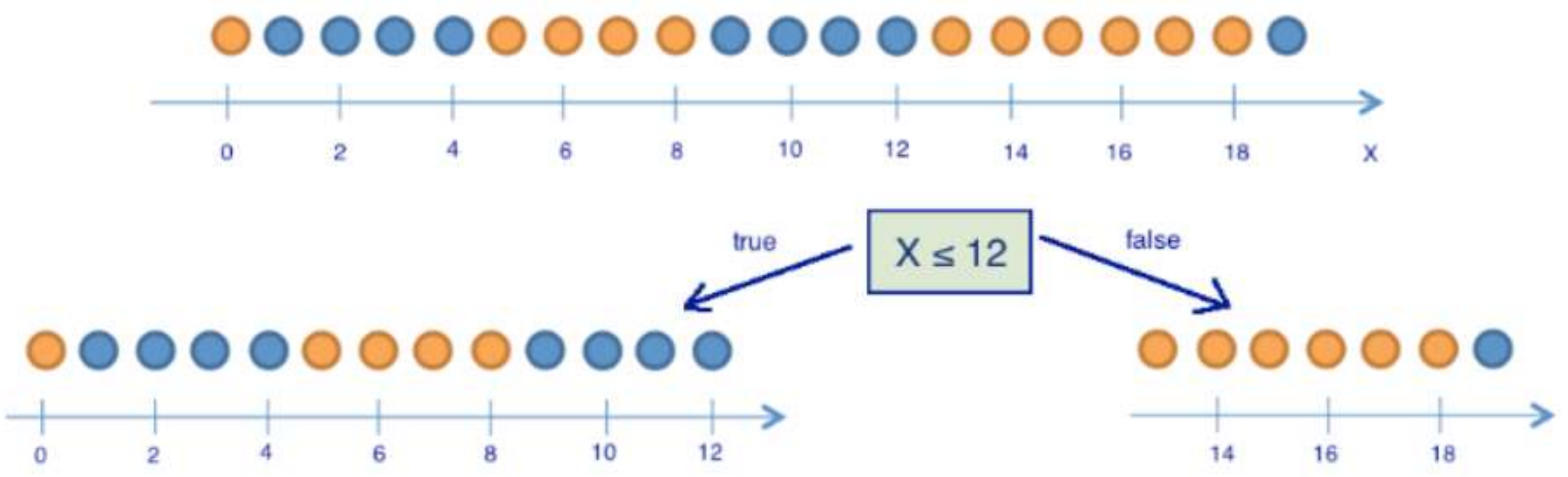


- Маємо 9 синіх кульок і 11 коричневих
- Якщо ми навмання виберемо кульку, то вона з ймовірністю $p_1 = \frac{9}{20}$ буде синьою та з ймовірністю $p_2 = \frac{11}{20}$ – коричневою
- За формулою Шеннона початкова ентропія нашої системи складе:

$$S_0 = -\frac{9}{20}\log_2\frac{9}{20} - \frac{11}{20}\log_2\frac{11}{20} \approx 1.00$$

- А як зміниться ентропія, якщо ми розіб'ємо кульки на дві частини: з координатою менше або дорівнює 12 і більше 12

Дерево прийняття рішень. Іграшковий приклад - 4



- У лівій частині маємо 13 кульок, із яких 8 синіх і 5 коричневих
- Ентропія лівої частини кульок складе:

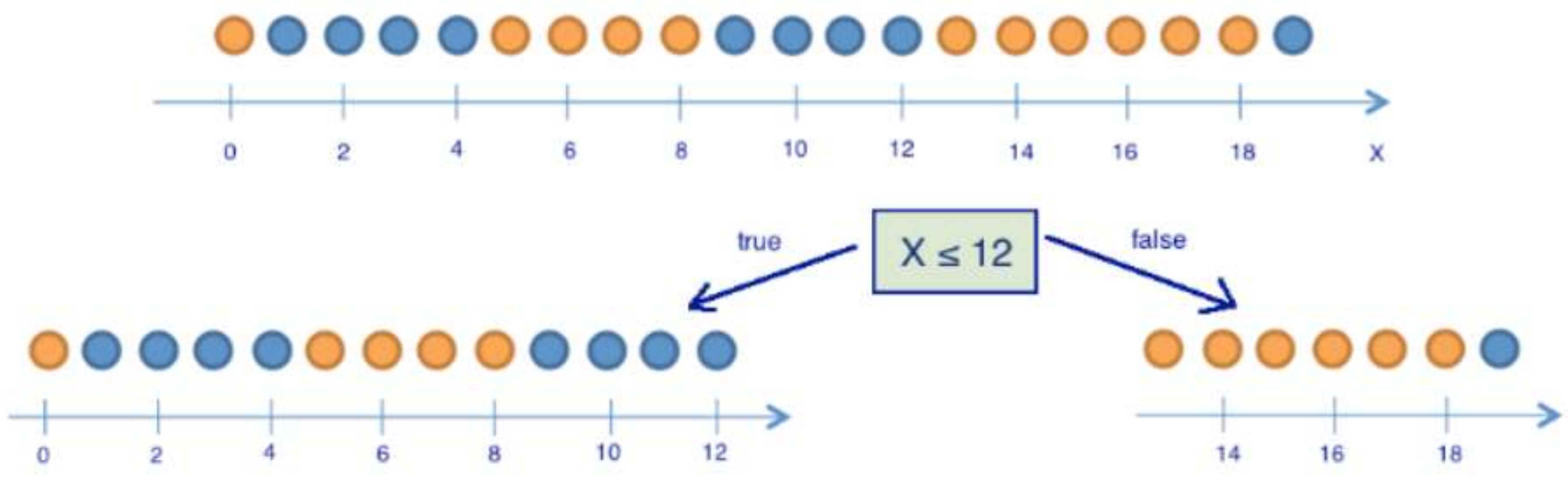
$$S_1 = -\frac{5}{13}\log_2\frac{5}{13} - \frac{8}{13}\log_2\frac{8}{13} \approx 0.96$$

- У правій частині маємо 7 кульок, із яких 1 синя і 6 коричневих
- Ентропія правої частини кульок складе:

$$S_2 = -\frac{1}{7}\log_2\frac{1}{7} - \frac{6}{7}\log_2\frac{6}{7} \approx 0.60$$

- Як порахувати ентропію системи із двох отриманих частин?
- Просто додати?

Дерево прийняття рішень. Іграшковий приклад - 5



- Як порахувати ентропію системи із двох отриманих частин?
- Просто додати?
- Ні, з ваговим коефіцієнтом величини відповідної частини!

- У лівій частині маємо 13 кульок, із яких 8 синіх і 5 коричневих
- Ентропія лівої частини кульок складе:

$$S_1 = -\frac{5}{13}\log_2\frac{5}{13} - \frac{8}{13}\log_2\frac{8}{13} \approx 0.96$$

- У правій частині маємо 7 кульок, із яких 1 синя і 6 коричневих
- Ентропія правої частини кульок складе:

$$S_2 = -\frac{1}{7}\log_2\frac{1}{7} - \frac{6}{7}\log_2\frac{6}{7} \approx 0.60$$

Дерево прийняття рішень. Іграшковий приклад - 6

- Оскільки ентропія – це ступінь невизначеності у системі, то її зменшення природно назвати **приростом інформації** (information gain, IG)
- При розділенні вибірки за ознакою Q (у нашому іграшковому прикладі це ознака "x≤12") приріст інформації визначається як

$$IG(Q) = S_0 - \sum_{i=1}^{q} \frac{Ni}{N} S_i,$$

де **q** – кількість частин (=груп) після розбиття,
N_i – число елементів вибірки, у яких ознака Q має *i*-е значення

- У нашому випадку після розділення маємо дві частини (q=2), одна з яких має 13 елементів (N_1=13), а друга – 7 (N_2=7)
- Приріст інформації складе

$$IG(x \le 12) = S_0 - \frac{13}{20} S_1 - \frac{7}{20} S_2 \approx 0.16$$

- Як порахувати ентропію системи із двох отриманих частин?
- Просто додати?

- Ні, з ваговим коефіцієнтом величини відповідної частини!

Дерево прийняття рішень. Іграшковий приклад - 6

- За формулою Шеннона початкова ентропія нашої системи склала:

$$S_0 = -\frac{9}{20}\log_2\frac{9}{20} - \frac{11}{20}\log_2\frac{11}{20} \approx 1.00$$

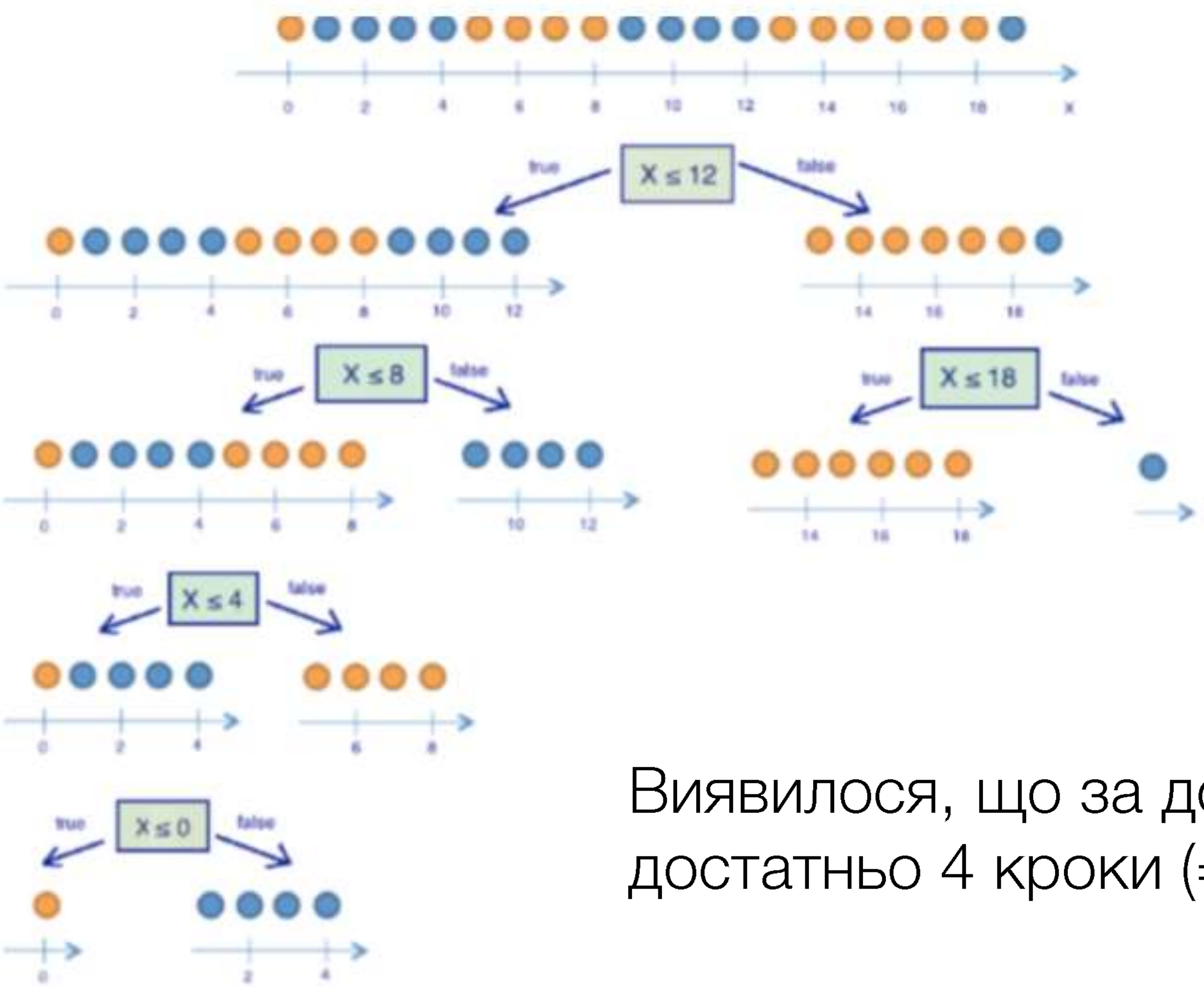
- У нашому випадку після розділення маємо дві частини (**q**=2), одна з яких має 13 елементів (N_1=13), а друга – 7 (N_2=7)
- Приріст інформації склав

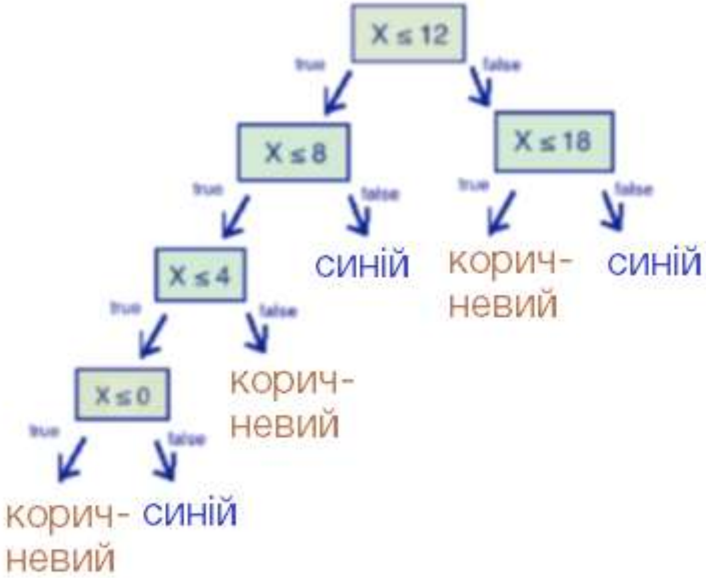
$$IG(x \leq 12) = S_0 - \frac{13}{20}S_1 - \frac{7}{20}S_2 \approx 0.16$$

- Виходить, що розбивши наші кульки на дві зазначені групи, ми зменшили ентропію, тобто отримали більш упорядковану систему, ніж спочатку

- Продовжимо розділення кульок на групи до тих пір, поки не отримаємо у кожній групі кульки однакового кольору

Дерево прийняття рішень.
Іграшковий приклад - 7





Виявилося, що за допомогою дерева прийняття рішень достатньо 4 кроки (=запитання), а не **20**!

Дерево прийняття рішень. Кредитний скорінг - 1

Дерево прийняття рішень. Кредитний скорінг - 2

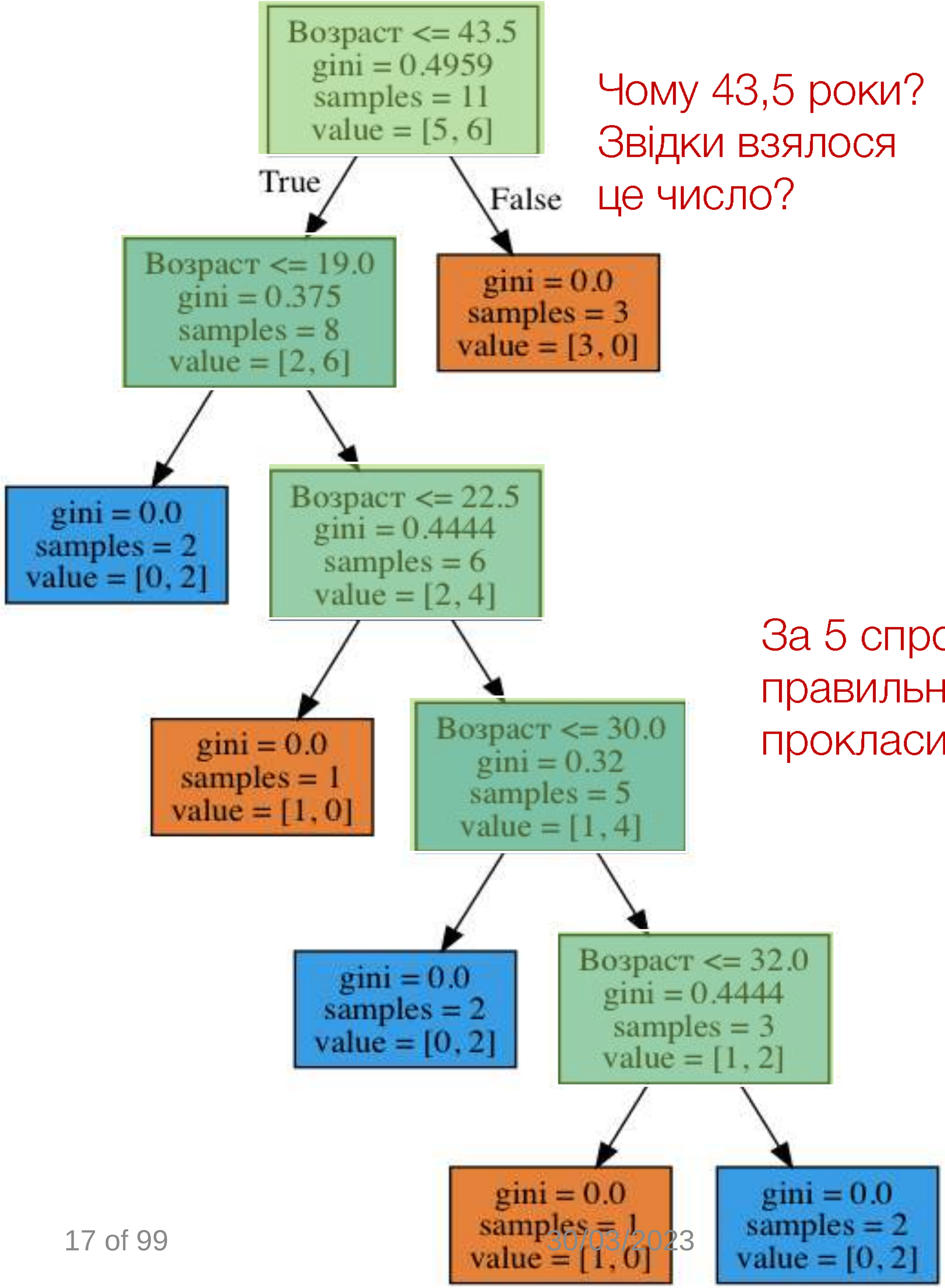
	Вік	Неповернення кредиту
0	17	1
2	18	1
3	20	0
7	25	1
8	29	1
9	31	0
10	33	1
4	38	1
5	49	0
6	55	0
1	64	0

- Початкові дані, впорядковані за віком
- *Крайній лівий стовпчик – початковий номер рядка даних*

- Як швидко розділити прохачів на дві категорії – тих, хто повертають кредити і не повертають?

Дерево прийняття рішень. Кредитний скорінг - 3

	Вік	Неповернення кредиту
0	17	1
2	18	1
3	20	0
7	25	1
8	29	1
9	31	0
10	33	1
4	38	1
5	49	0
6	55	0
1	64	0



Чому 43,5 роки? Звідки взялося це число?

За 5 спроб все правильно прокласифікуємо

Дерево прийняття рішень. Кредитний скорінг - 4

- Початкові дані, впорядковані за віком

	Вік	Зарплата	Неповернення кредиту
0	17	25	1
2	18	22	1
3	20	36	0
7	25	70	1
8	29	33	1
9	31	102	0
10	33	88	1
4	38	37	1
5	49	59	0
6	55	74	0
1	64	80	0

- Початкові дані, впорядковані за з/п

	Вік	Зарплата	Неповернення кредиту
2	18	22	1
0	17	25	1
8	29	33	1
3	20	36	0
4	38	37	1
5	49	59	0
7	25	70	1
6	55	74	0
1	64	80	0
10	33	88	1
9	31	102	0

- Якщо додасться нова ознака, то чи можна зменшити кількість спроб?

Дерево прийняття рішень. Кредитний скорінг - 5

- Початкові дані, впорядковані **за віком**

	Вік	Зарплата	Неповернення кредиту
0	17	25	1
2	18	22	1
3	20	36	0
7	25	70	1
8	29	33	1
9	31	102	0
10	33	88	1
4	38	37	1
5	49	59	0
6	55	74	0
1	64	80	0

- Початкові дані, впорядковані **за з/п**

	Вік	Зарплата	Неповернення кредиту
2	18	22	1
0	17	25	1
8	29	33	1
3	20	36	0
4	38	37	1
5	49	59	0
7	25	70	1
6	55	74	0
1	64	80	0
10	33	88	1
9	31	102	0

- Якщо додасться нова ознака, то чи можна зменшити кількість спроб?
- Є два кандидати, з яких почати розділення

Дерево прийняття рішень. Кредитний скорінг - 6

- Початкові дані, впорядковані **за віком**

	Вік	Зарплата	Неповернення кредиту
0	17	25	1
2	18	22	1
3	20	36	0
7	25	70	1
8	29	33	1
9	31	102	0
10	33	88	1
4	38	37	1
5	49	59	0
6	55	74	0
1	64	80	0

- Початкові дані, впорядковані **за з/п**

	Вік	Зарплата	Неповернення кредиту
2	18	22	1
0	17	25	1
8	29	33	1
3	20	36	0
4	38	37	1
~~5~~	~~49~~	~~59~~	~~0~~
7	25	70	1
~~6~~	~~55~~	~~74~~	~~0~~
~~1~~	~~64~~	~~80~~	~~0~~
10	33	88	1
9	31	102	0

- Якщо додасться нова ознака, то чи можна зменшити кількість спроб?

Дерево прийняття рішень. Кредитний скорінг - 7

- Початкові дані, впорядковані **за віком**

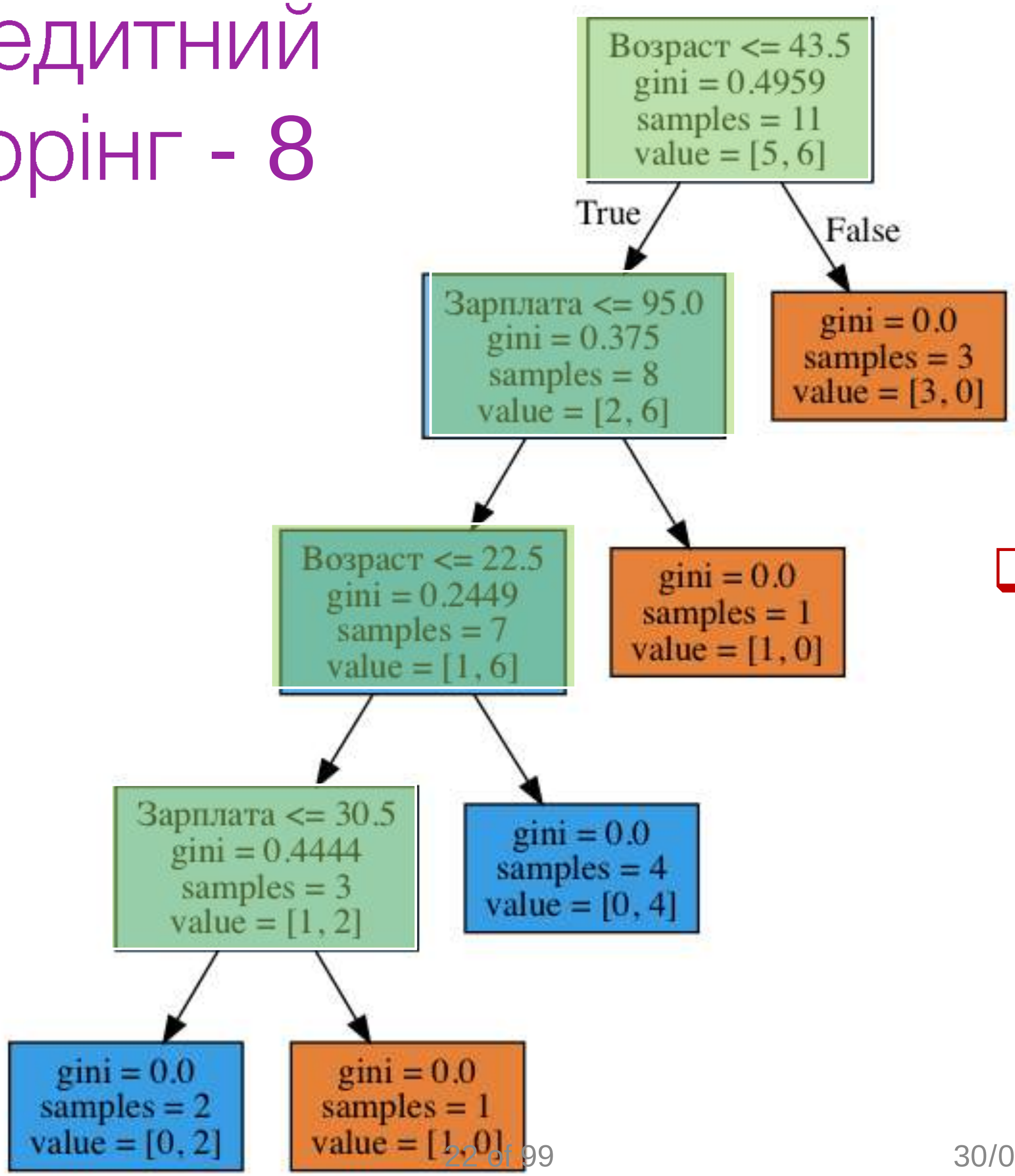
	Вік	Зарплата	Неповернення кредиту
0	17	25	1
2	18	22	1
3	20	36	0
7	25	70	1
8	29	33	1
9	31	102	0
10	33	88	1
4	38	37	1
5	49	59	0
6	55	74	0
1	64	80	0

- Початкові дані, впорядковані **за з/п**

	Вік	Зарплата	Неповернення кредиту
2	18	22	1
0	17	25	1
8	29	33	1
3	20	36	0
4	38	37	1
~~**5**~~	~~49~~	~~59~~	~~0~~
7	25	70	1
~~**6**~~	~~55~~	~~74~~	~~0~~
~~**1**~~	~~64~~	~~80~~	~~0~~
10	33	88	1
9	31	102	0

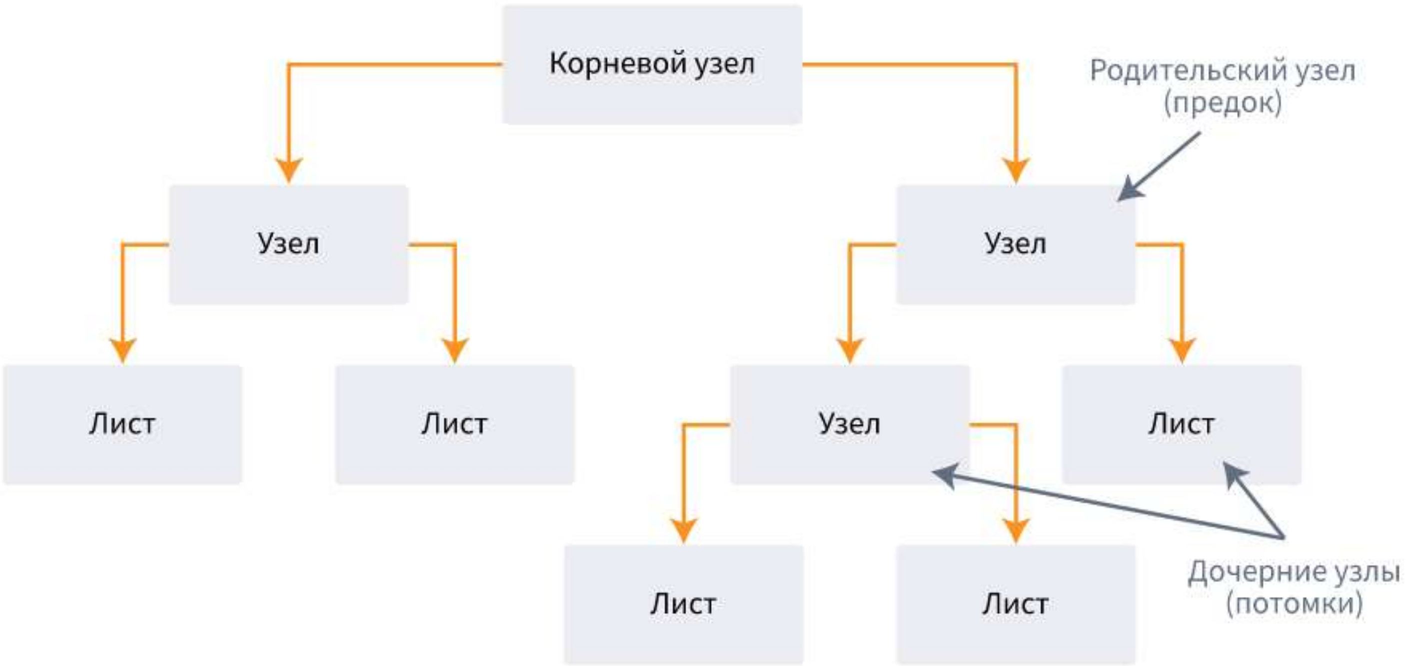
- Якщо додасться нова ознака, то чи можна зменшити кількість спроб?
- Для другого кроку тепер маємо дві близькі можливості

Дерево прийняття рішень. Кредитний скорінг - 8



- За 4 спроби все правильно прокласифікуємо

Дерево прийняття (= ухвалення) рішень - 1

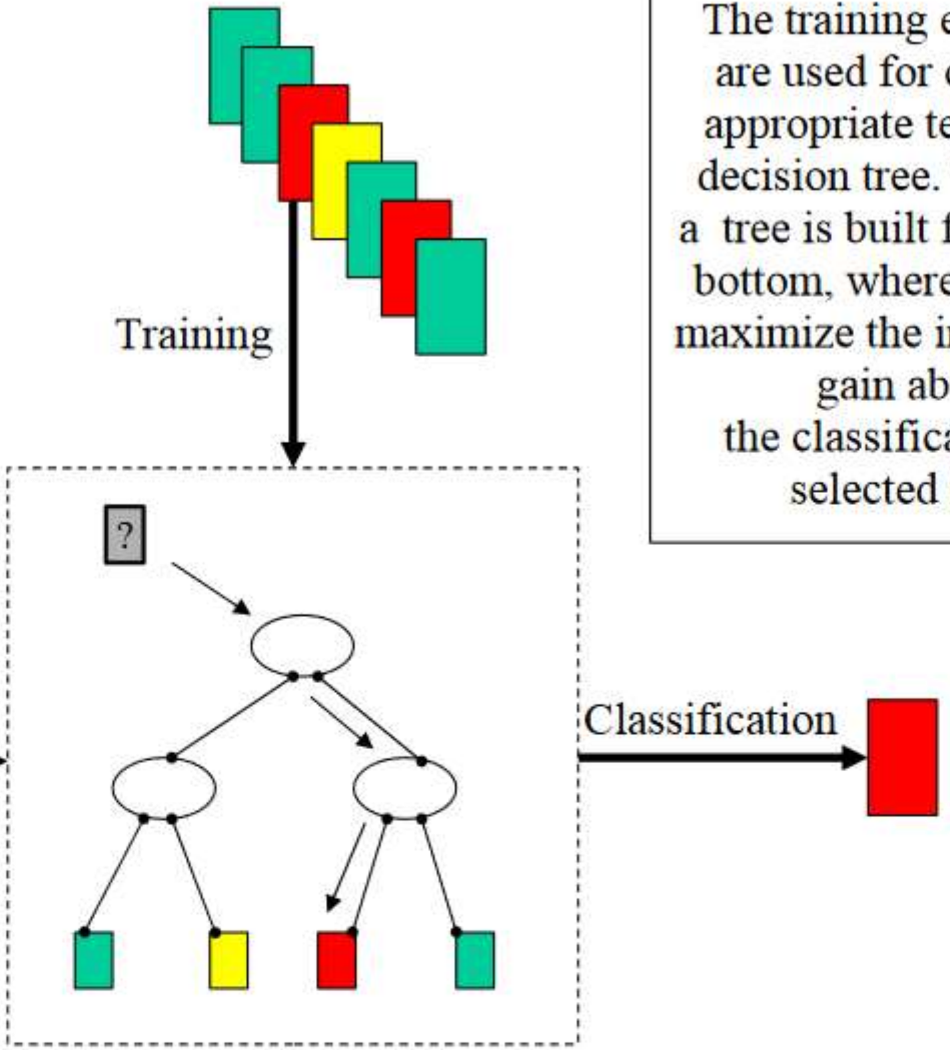


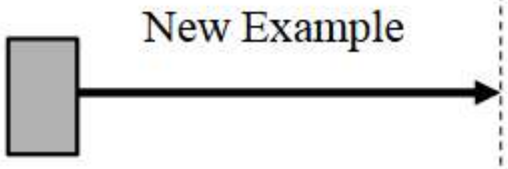
A decision tree consists of nodes and arcs which connect nodes, where

- The topmost node in a tree is the root node
- Each internal node (or decision node) denotes a test of the value of a given attribute
- Each leaf node (or terminal node) indicates predicted class, i.e. the value of the target attribute of examples
- Each branch represents an outcome of the test

Decision Tree Learning

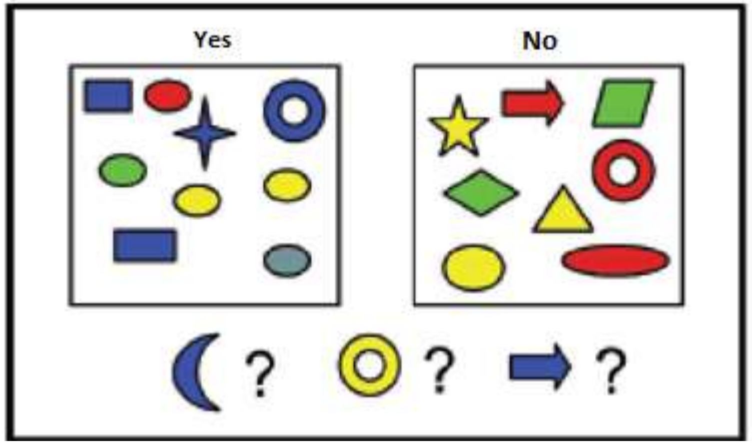
In ***Decision Tree Learning***, a new example is classified by submitting it to a series of tests that determine the class label of the example. These tests are organized in a hierarchical structure called a *decision tree*.





The training examples are used for choosing appropriate tests in the decision tree. Typically, a tree is built from top to bottom, where tests that maximize the information gain about the classification are selected first.

How are decision trees used for classification?



Features (attributes)

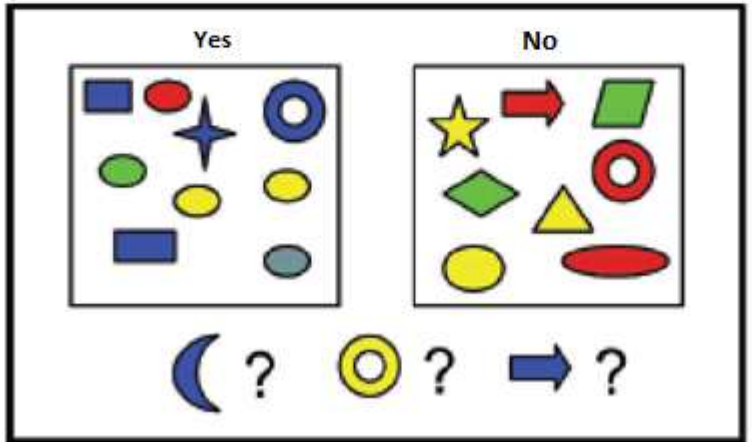
Cases

Color	Shape	Size (cm)	Label
Blue	Square	10	1
Red	Ellipse	2.4	1
Red	Ellipse	20.7	0

(Fig. from Kevin Murphy)

- What pattern we see in the figure above?
- Is it possible for us to depict it with a tree?

How are decision trees used for classification?



Features (attributes)

Cases

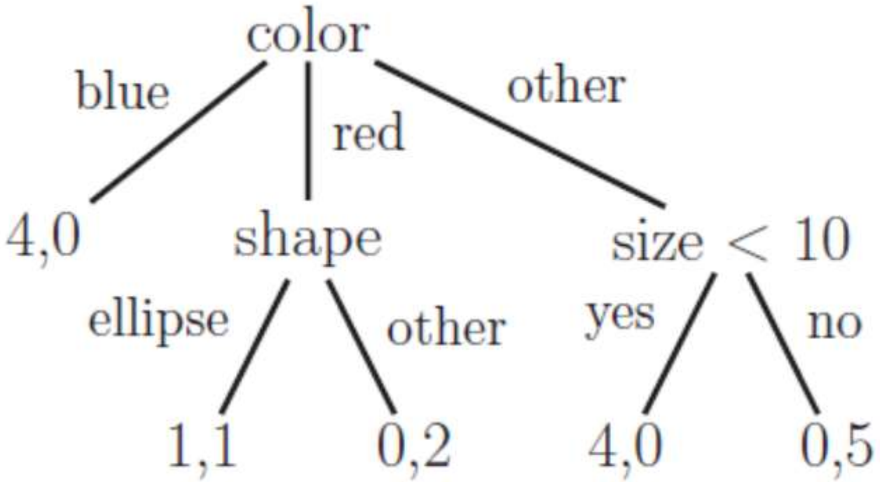
Color	Shape	Size (cm)
Blue	Square	10
Red	Ellipse	2.4
Red	Ellipse	20.7

Label
1
1
0

- ❑ (синя стрілка?): у нас не вистачає прикладів для правильної класифікації і ми можемо помилитися

  - якщо синю стрілку потрібно класифікувати праворуч, то у гілку по кольору потрібно додати ще одну перевірку, тобто чим більше ми враховуємо різних випадків, тим, зазвичай, розростається дерево, ми перенавчаємося!

  - а якщо ж це були помилки, то дерево сильно забруднюється, тому **потрібно своєчасно зупинятися**

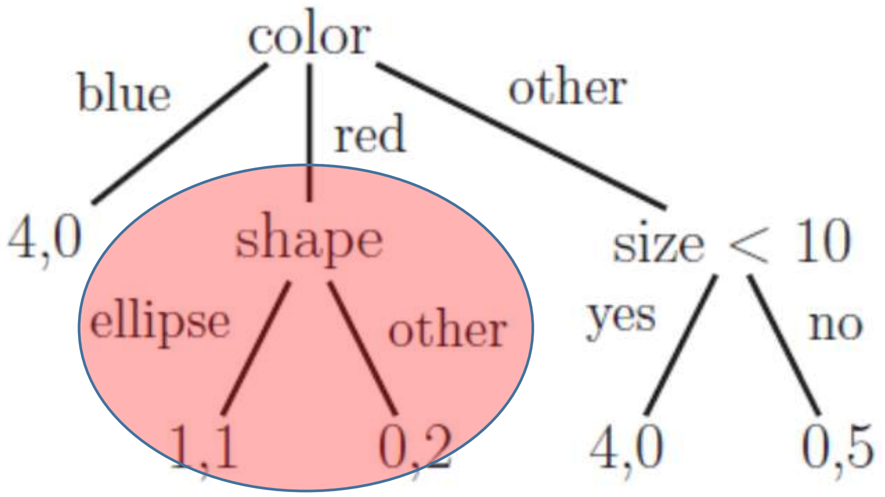


We need a compact tree for efficiency of computation and representation

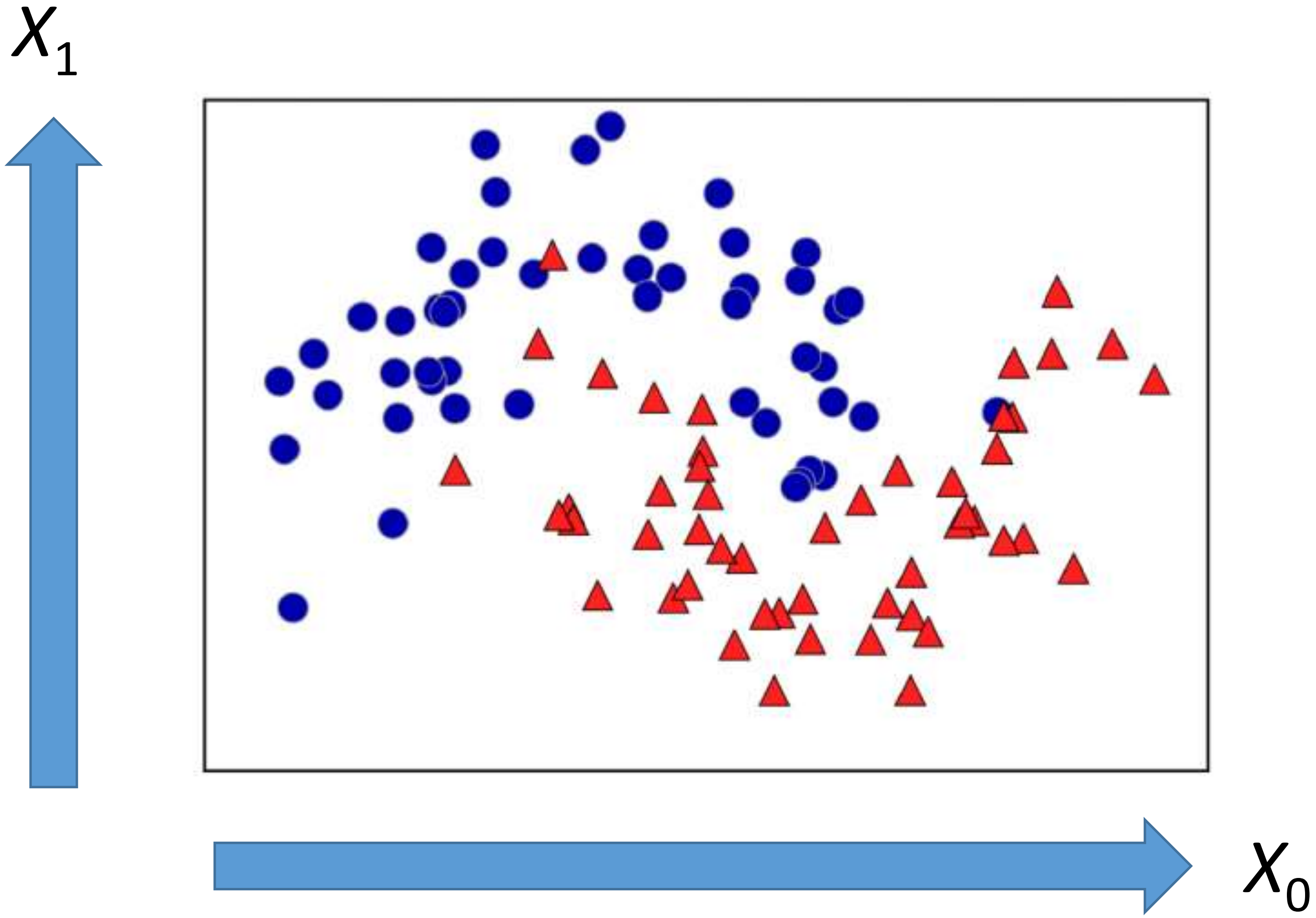
- The training dataset is finite (incomplete), may be noisy and hence generalization is needed

- Every decision is not affected by every variables

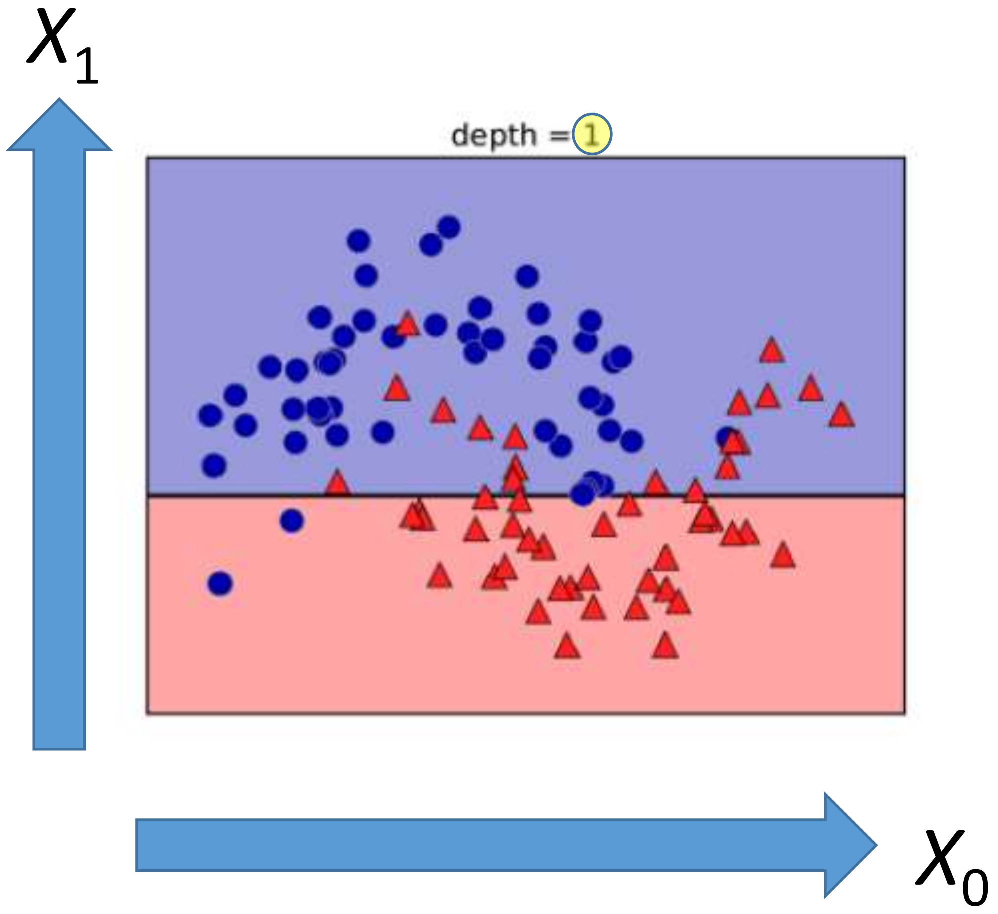
- Many variables become don't care and we can utilize this aspect to build a compact tree

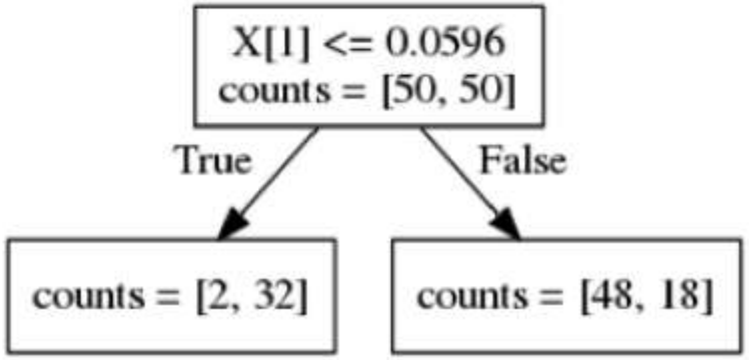


Classification with continuous attributes – 1

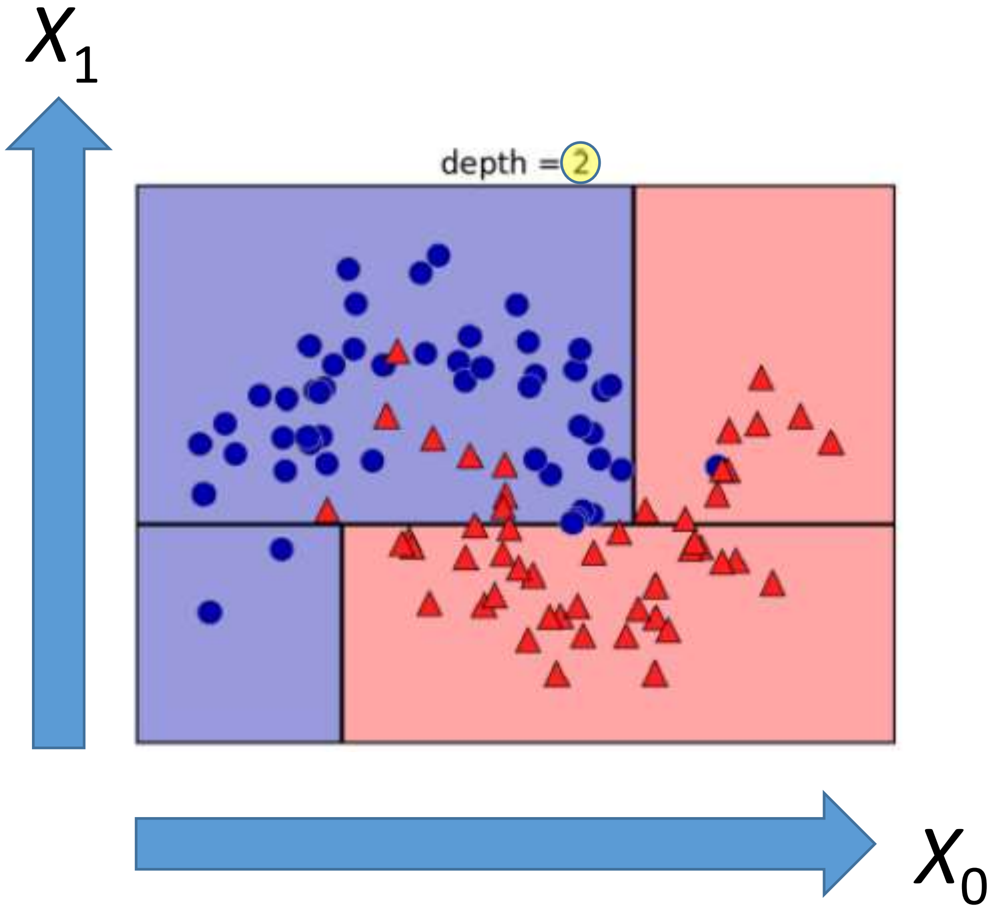


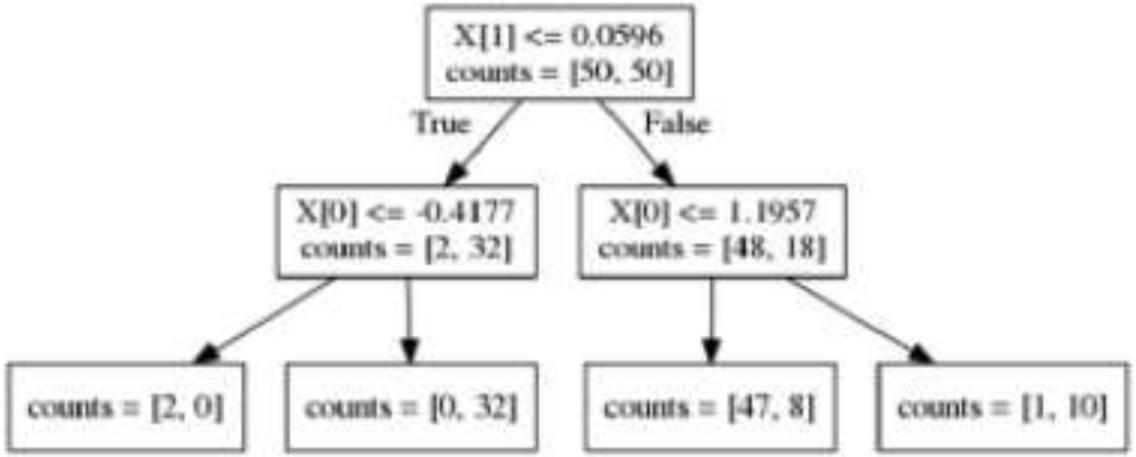
Classification with continuous attributes – 2



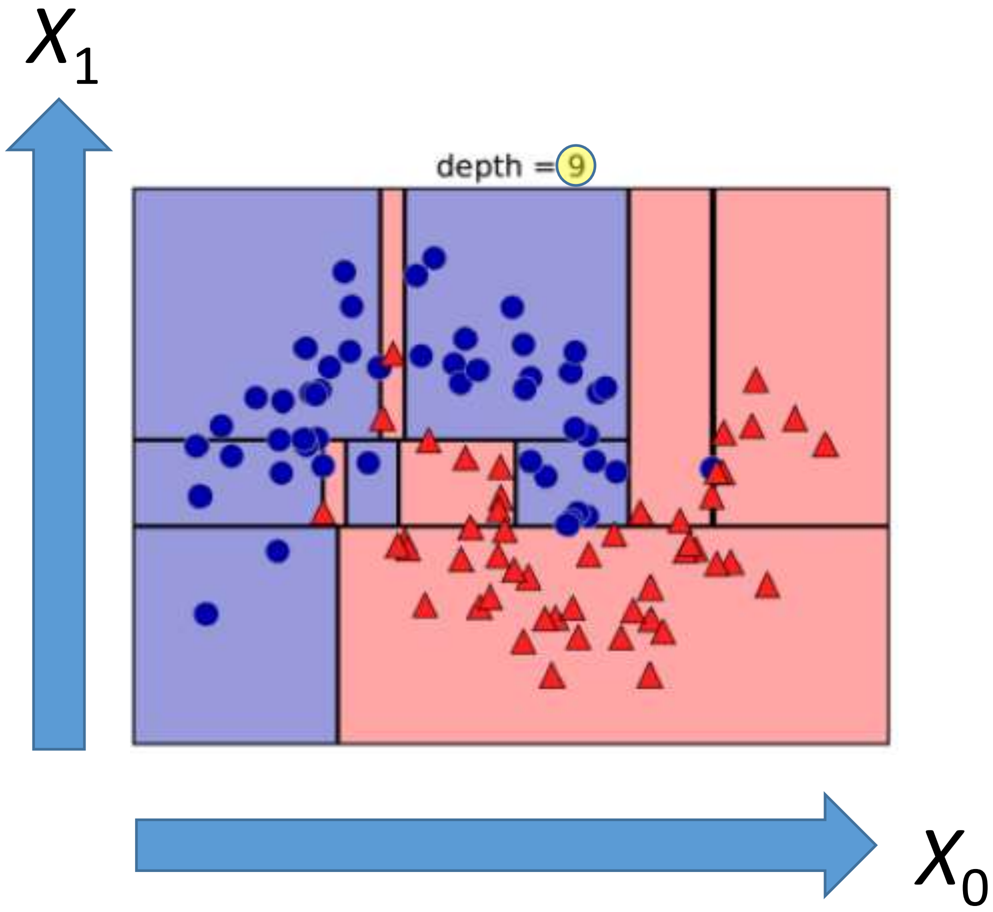


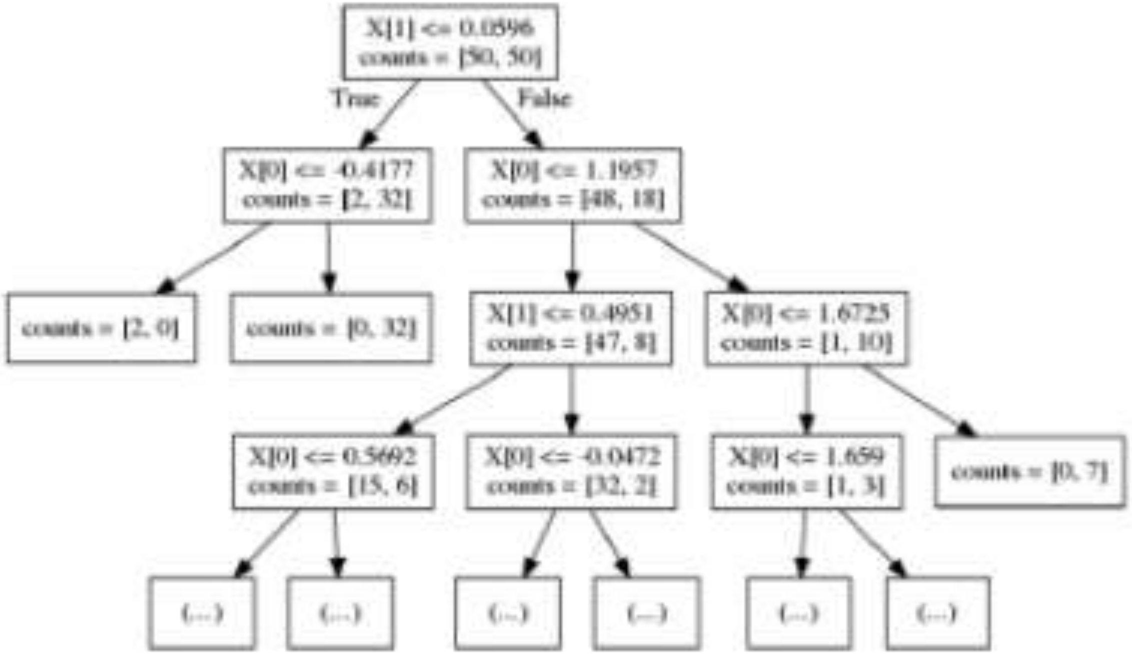
Classification with continuous attributes – 3



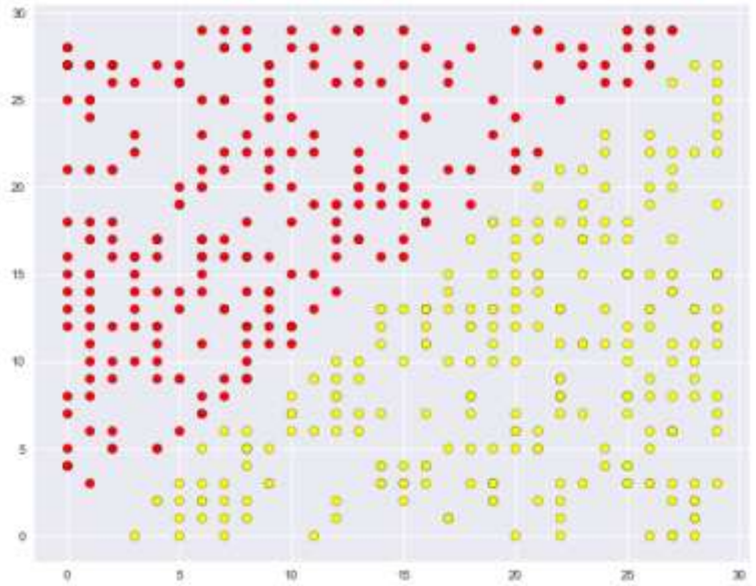


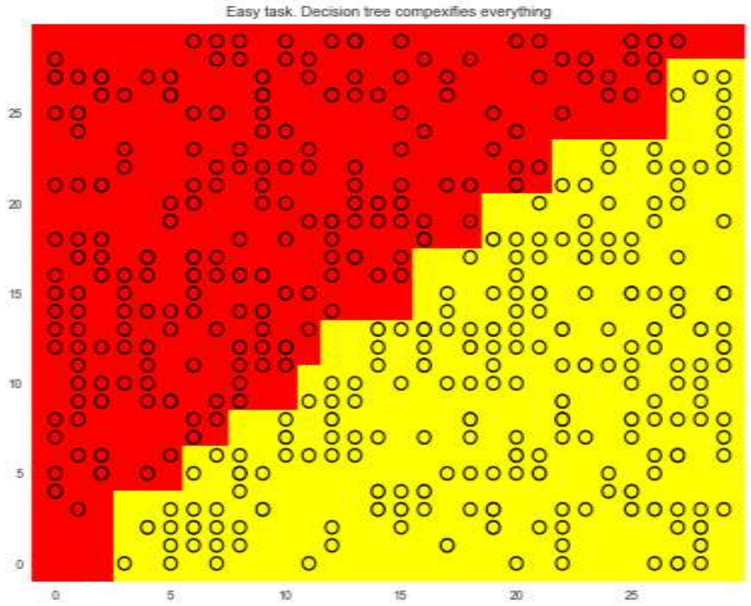
Classification with continuous attributes – 4





Дерева прийняття рішень не завжди даують гарний результат





Складові алгоритмів побудови дерев прийняття рішень

- «В ширину» чи «вглиб»?
- Binary (CART) or multiway (ID3, CHAID) splits?
- Splitting criterion (greedy algorithm):
  - Information Gain (приріст інформації)
  - (Information) Gain ratio (зважений приріст інформації)
  - Gini impurity (неоднорідність Джині)
  - Classification error (помилка класифікації)
  - Chi-square test statistic for independence (критерій Хі-квадрат)
- Обрізання дерев (prepruning, postpruning)
- Обробка пропущених значень, паралелізація

Greedy algorithm
(жадібний алгоритм)

- Всі відомі алгоритми побудови дерев прийняття рішень – це жадібні алгоритми
- **Жадібний алгоритм** – це алгоритм, який на кожному кроці робить локально найкращий вибір в надії, що підсумкове рішення буде оптимальним

- Наприклад, алгоритм Дейкстри знаходження найкоротшого шляху в графі є жадібним, оскільки на кожному кроці шукається вершина з найменшою вагою, в якій ми ще не бували, після чого оновлюємо значення інших вершин
- При цьому можна довести, що найкоротші шляхи, знайдені в вершинах, є оптимальними
- Але не завжди жадібні алгоритми дають гарний результат: взяття «жертви» у шахах може привести до поразки

Візьмемо як критерій розгалуження приріст інформації

- Оскільки ентропія – це ступінь невизначеності у системі, то її зменшення природно назвати **приростом інформації** (information gain, IG)
- При розділенні вибірки за ознакою Q приріст інформації визначається як

$$IG(Q) = S_0 - \sum_{i=1}^{q} \frac{Ni}{N} S_i,$$

де q – кількість частин (=груп) після розбиття,

N_i – число елементів вибірки, у яких ознака Q має *i*-е значення

Who is it?

Nadal and weather

Entropy

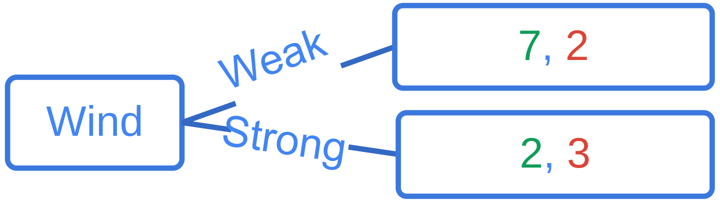
Action	Wind	Temp	Outlook	Humidity
Play (P)	Weak	Hot	Sunny	High
Play	Strong	Hot	Sunny	High
Stay (S)	Weak	Hot	Rain	High
Play	Weak	Mid	Overcast	High
Stay	Strong	Cold	Rain	Normal
Play	Weak	Cold	Overcast	Normal
Stay	Strong	Cold	Rain	Normal
Play	Weak	Mid	Sunny	Normal
Play	Weak	Cold	Sunny	Normal
Play	Strong	Mid	Overcast	Normal
Stay	Weak	Mid	Sunny	High
Stay	Strong	Mid	Rain	High
Play	Weak	Hot	Overcast	Normal
Play	Weak	Cold	Rain	High

Nadal training statistics with weather conditions

Will he play if:

Action	Wind	Temp	Outlook	Humidity
?	**Weak**	**Hot**	**Sunny**	**Normal**

$$E(D) = -\left(\frac{9}{14}\right)\log_2\left(\frac{9}{14}\right) - \left(\frac{5}{14}\right)\log_2\left(\frac{5}{14}\right) = 0.940$$



Simplification of computation of average entropy:

$$E(Weak) = -\left(\frac{7}{9}\right)\log_2\left(\frac{7}{9}\right) - \left(\frac{2}{9}\right)\log_2\left(\frac{2}{9}\right) =$$
$$= -\left(\frac{1}{9}\right)(7\log_2 7 + 2\log_2 2) + \log_2 9$$

$$E(Weak) = -\left(\frac{7}{9}\right)\log_2\left(\frac{7}{9}\right) - \left(\frac{2}{9}\right)\log_2\left(\frac{2}{9}\right) = 0.764$$

$$E(Strong) = -\left(\frac{2}{5}\right)\log_2\left(\frac{2}{5}\right) - \left(\frac{3}{5}\right)\log_2\left(\frac{3}{5}\right) = 0.970$$

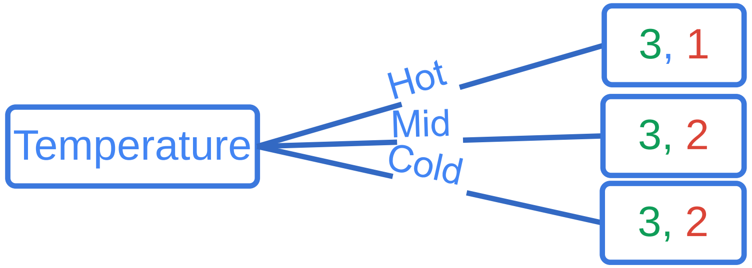
$$Gain(D, Wind) = E(D) - \left(\frac{9}{14}E(Weak) + \frac{5}{14}E(Strong)\right)$$
$$= 0.940 - (0.491 + 0.346) = 0.940 - 0.837 = 0.102$$

Entropy

Action	Wind	Temp	Outlook	Humidity
Play (P)	Weak	Hot	Sunny	High
Play	Strong	Hot	Sunny	High
Stay (S)	Weak	Hot	Rain	High
Play	Weak	Mid	Overcast	High
Stay	Strong	Cold	Rain	Normal
Play	Weak	Cold	Overcast	Normal
Stay	Strong	Cold	Rain	Normal
Play	Weak	Mid	Sunny	Normal
Play	Weak	Cold	Sunny	Normal
Play	Strong	Mid	Overcast	Normal
Stay	Weak	Mid	Sunny	High
Stay	Strong	Mid	Rain	High
Play	Weak	Hot	Overcast	Normal
Play	Weak	Cold	Rain	High

Nadal training statistics with weather conditions

Action	Wind	Temp	Outlook	Humidity
Gain	**0.102**			



$$E(Hot) = -\left(\frac{3}{4}\right)\log_2\left(\frac{3}{4}\right) - \left(\frac{1}{4}\right)\log_2\left(\frac{1}{4}\right) = 0.811$$

$$E(Mid) = E(Cold) = -\left(\frac{3}{5}\right)\log_2\left(\frac{3}{5}\right) - \left(\frac{2}{5}\right)\log_2\left(\frac{2}{5}\right) = 0.970$$

$$Gain(D, Temp) = E(D) - \left(\frac{4}{14}E(H) + \frac{5}{14}E(M) + \frac{5}{14}E(C)\right)$$

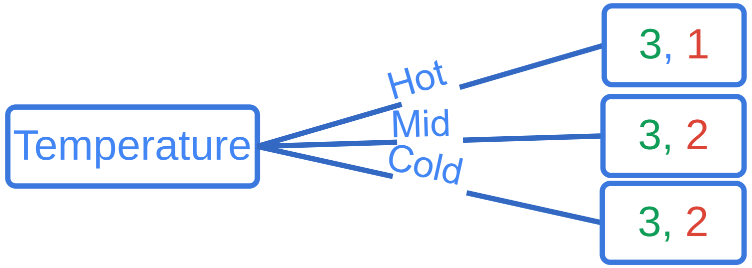
$$= 0.940 - (0.231 + 0.346 + 0.346) = 0.940 - 0.932 = 0.008$$

Entropy

Action	Wind	Temp	Outlook	Humidity
Play (P)	Weak	Hot	Sunny	High
Play	Strong	Hot	Sunny	High
Stay (S)	Weak	Hot	Rain	High
Play	Weak	Mid	Overcast	High
Stay	Strong	Cold	Rain	Normal
Play	Weak	Cold	Overcast	Normal
Stay	Strong	Cold	Rain	Normal
Play	Weak	Mid	Sunny	Normal
Play	Weak	Cold	Sunny	Normal
Play	Strong	Mid	Overcast	Normal
Stay	Weak	Mid	Sunny	High
Stay	Strong	Mid	Rain	High
Play	Weak	Hot	Overcast	Normal
Play	Weak	Cold	Rain	High

Nadal training statistics with weather conditions

Action	Wind	Temp	Outlook	Humidity
Gain	**0.102**			



$$E(Hot) = -\left(\frac{3}{4}\right)\log_2\left(\frac{3}{4}\right) - \left(\frac{1}{4}\right)\log_2\left(\frac{1}{4}\right) = 0.811$$

$$E(Mid) = E(Cold) = -\left(\frac{3}{5}\right)\log_2\left(\frac{3}{5}\right) - \left(\frac{2}{5}\right)\log_2\left(\frac{2}{5}\right) = 0.970$$

$$Gain(D, Temp) = E(D) - \left(\frac{4}{14}E(H) + \frac{5}{14}E(M) + \frac{5}{14}E(C)\right)$$

$$= 0.940 - (0.231 + 0.346 + 0.346) = 0.940 - 0.932 = 0.008$$

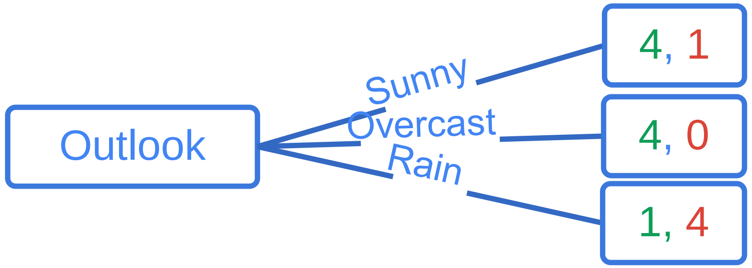
Зверніть увагу: маленький приріст інформації, бо майже рівномірний розподіл випадків для різних температур

Entropy

Action	Wind	Temp	Outlook	Humidity
Play (P)	Weak	Hot	Sunny	High
Play	Strong	Hot	Sunny	High
Stay (S)	Weak	Hot	Rain	High
Play	Weak	Mid	Overcast	High
Stay	Strong	Cold	Rain	Normal
Play	Weak	Cold	Overcast	Normal
Stay	Strong	Cold	Rain	Normal
Play	Weak	Mid	Sunny	Normal
Play	Weak	Cold	Sunny	Normal
Play	Strong	Mid	Overcast	Normal
Stay	Weak	Mid	Sunny	High
Stay	Strong	Mid	Rain	High
Play	Weak	Hot	Overcast	Normal
Play	Weak	Cold	Rain	High

Nadal training statistics with weather conditions

Action	Wind	Temp	Outlook	Humidity
Gain	**0.102**	**0.008**		



$E(Sunny) = 0.722$

$E(Overcast) = 0$

$E(Rain) = 0.722$

$$Gain(D, Outlook) = E(D) - \left(\frac{5}{14}E(S) + \frac{4}{14}E(O) + \frac{5}{14}E(R)\right)$$

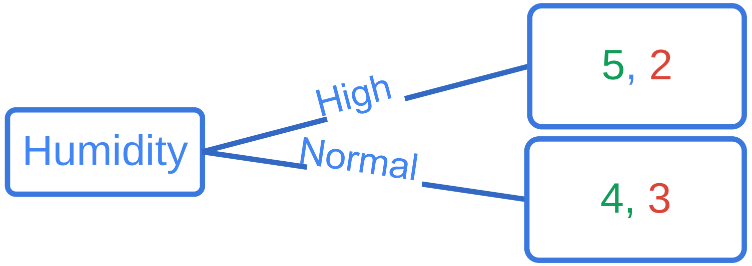
$$= 0.940 - (0.258 + 0 + 0.258) = 0.940 - 0.516 = 0.424$$

Entropy

Action	Wind	Temp	Outlook	Humidity
Play (P)	Weak	Hot	Sunny	High
Play	Strong	Hot	Sunny	High
Stay (S)	Weak	Hot	Rain	High
Play	Weak	Mid	Overcast	High
Stay	Strong	Cold	Rain	Normal
Play	Weak	Cold	Overcast	Normal
Stay	Strong	Cold	Rain	Normal
Play	Weak	Mid	Sunny	Normal
Play	Weak	Cold	Sunny	Normal
Play	Strong	Mid	Overcast	Normal
Stay	Weak	Mid	Sunny	High
Stay	Strong	Mid	Rain	High
Play	Weak	Hot	Overcast	Normal
Play	Weak	Cold	Rain	High

Nadal training statistics with weather conditions

Action	Wind	Temp	Outlook	Humidity
Gain	**0.102**	**0.008**	**0.424**	



$E(High) = 0.863$

$E(Normal) = 0.985$

$$Gain(D, Humidity) = E(D) - \left(\frac{7}{14}E(H) + \frac{7}{14}E(N)\right)$$

$$= 0.940 - (0.431 + 0.493) = 0.940 - 0.924 = 0.016$$

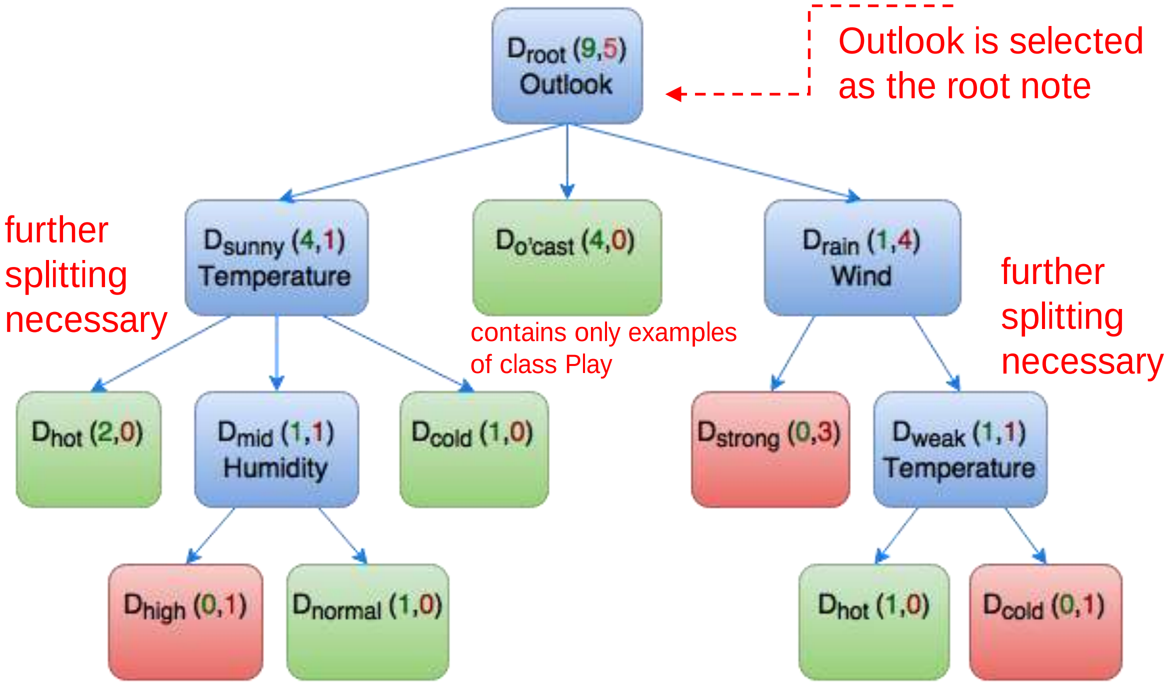
Entropy. Final decision tree

Action	Wind	Temp	Outlook	Humidity
Play (P)	Weak	Hot	Sunny	High
Play	Strong	Hot	Sunny	High
Stay (S)	Weak	Hot	Rain	High
Play	Weak	Mid	Overcast	High
Stay	Strong	Cold	Rain	Normal
Play	Weak	Cold	Overcast	Normal
Stay	Strong	Cold	Rain	Normal
Play	Weak	Mid	Sunny	Normal
Play	Weak	Cold	Sunny	Normal
Play	Strong	Mid	Overcast	Normal
Stay	Weak	Mid	Sunny	High
Stay	Strong	Mid	Rain	High
Play	Weak	Hot	Overcast	Normal
Play	Weak	Cold	Rain	High

Nadal training statistics with weather conditions

Information gains at root node are:

Action	Wind	Temp	Outlook	Humidity
Gain	0.102	0.008	**0.424**	0.016

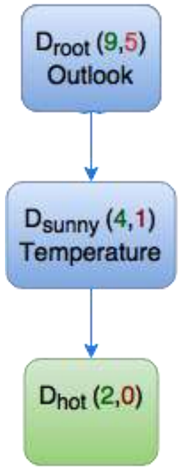


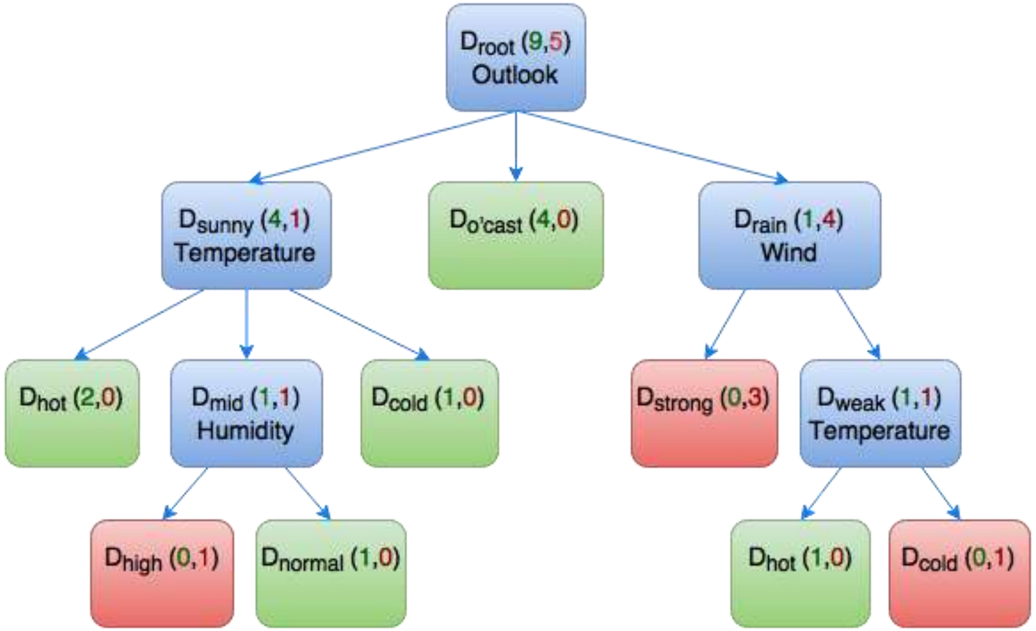
Pure leaves→ No further expansion necessary

Decision for Test dataset

Will he play if:

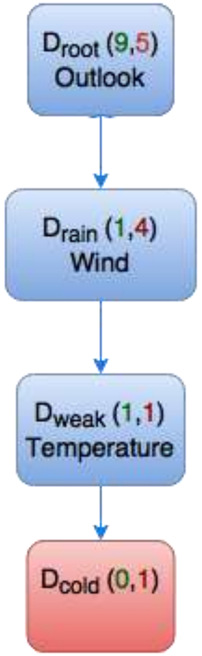
Action	Wind	Temp	Outlook	Humidity
?	Weak	Hot	Sunny	Normal





Will he play if:

Action	Wind	Temp	Outlook	Humidity
?	Weak	Cold	Rain	Normal



Decision Trees vs Neural Networks: rules! (white vs black box)

- The Decision Tree is one of the most powerful and popular classification and prediction algorithms in current use in machine learning
- The attractiveness of decision trees is due to the fact that, in contrast to neural networks, decision trees represent rules
- Rules can readily be expressed so that humans can understand them or even directly used in a database access language like SQL so that records falling into a particular category may be retrieved

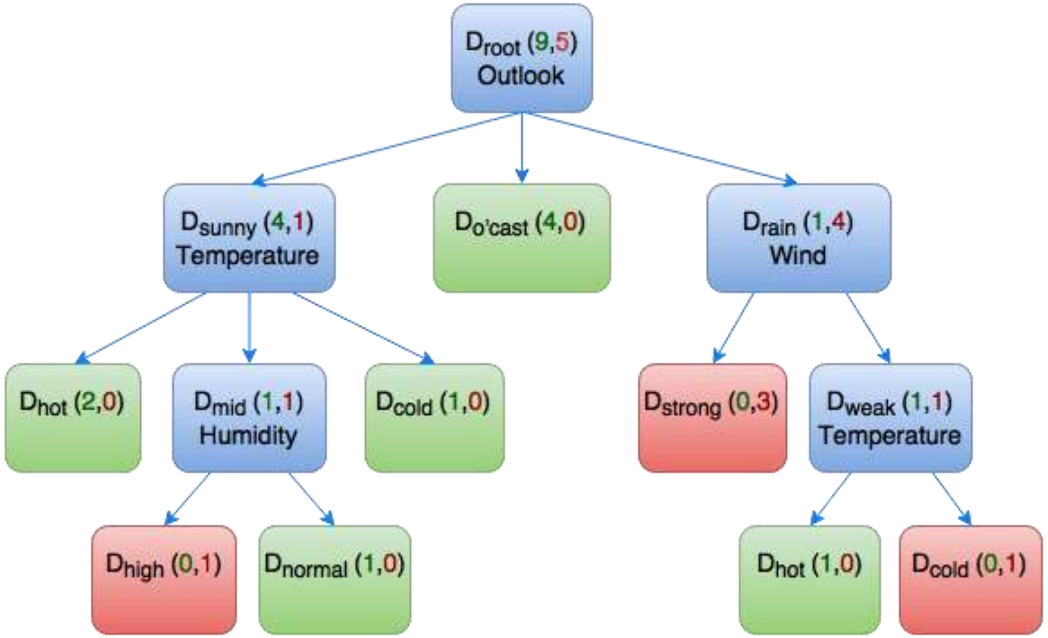
Rules Extraction (дерева прийняття рішень як логічний метод класифікації)

- We have 8 leaf nodes.

In a decision tree, each leaf node represents a rule

- We have the following rules corresponding to the tree given in figure

A decision tree produces a sequence of rules (or series of questions) that can be used to recognize the class



1. if it is SUNNY and Temperature is HOT then PLAY
2. if it is SUNNY and Temperature is MID and Humidity is HIGH then STAY
3. if it is SUNNY and Temperature is MID and Humidity is NORMAL then PLAY
4. if it is OVERCAST then PLAY
5. if it is RAIN and Wind is WEAK and Temperature is HOT then PLAY

…

1. (Outlook=SUNNY) & (Temperature=HOT) ⇒ PLAY
2. (Outlook=SUNNY) & (Temperature=MID) & (Humidity=HIGH) ⇒ STAY
3. (Outlook=SUNNY) & (Temperature=MID) & (Humidity= NORMAL) ⇒ PLAY
4. (Outlook=OVERCAST) ⇒ PLAY
5. (Outlook= RAIN) & (Wind= WEAK) & (Temperature= HOT) ⇒ PLAY

Highly-branching attributes

- Problematic: attributes with a large number of values
  - extreme case: each example has its own value
    - e.g. example ID; credit card number; telephone number; Day attribute in weather data

- Subsets are more likely to be pure if there is a large number of different attribute values
  - Information gain is biased towards choosing attributes with a large number of values

- This may cause several problems:
  - Overfitting
    - selection of an attribute that is non-optimal for prediction
  - Fragmentation
    - data are fragmented into (too) many small sets

If we add Data attribute (in our case "Nadal and weather")

- **Information gain is maximal for Data** (0.940 bits):

$$\boldsymbol{Gain(D,V) = E(D) - \sum \frac{N_v}{N} \times E(D_v)}$$

$$Gain(D,V) =$$

$$= E(D) - \left(\frac{1}{14}E(1st\ day) + \frac{1}{14}E(2nd\ day) + \cdots + \frac{1}{14}E(14th\ day)\right) =$$

$$= 0.940 - \left(\frac{1}{14}\left(-\frac{1}{1}\log_2\left(\frac{1}{1}\right)\right) + \frac{1}{14}\left(-\frac{1}{1}\log_2\left(\frac{1}{1}\right)\right) + \cdots + \frac{1}{14}\left(-\frac{1}{1}\log_2\left(\frac{1}{1}\right)\right)\right) =$$

$$= 0.940 - 0 = 0.940$$

$-1/1\ log2(1/1)$, бо в цей день він або грав, або не грав, інших варіантів не має: ймовірність $p = 1$

If we add Data attribute (in our case “Nadal and weather”)

- **Information gain is maximal for Data** (0.940 bits):

$$\boldsymbol{Gain(D,V) = E(D) - \sum \frac{N_v}{N} \times E(D_v)}$$

$$Gain(D,V) =$$

$$= E(D) - \left(\frac{1}{14}E(1st\ day) + \frac{1}{14}E(2nd\ day) + \cdots + \frac{1}{14}E(14th\ day)\right) =$$

$$= 0.940 - \left(\frac{1}{14}\left(-\frac{1}{1}\log_2\left(\frac{1}{1}\right)\right) + \frac{1}{14}\left(-\frac{1}{1}\log_2\left(\frac{1}{1}\right)\right) + \cdots + \frac{1}{14}\left(-\frac{1}{1}\log_2\left(\frac{1}{1}\right)\right)\right) =$$

$$= 0.940 - 0 = 0.940$$

Чому погано?
- ніякого прогнозу не побудуєш
- головне – не можна знайти закономірності

Чим характеризуються атрибути типу Data?

Intrinsic information of a feature (власна інформація атрибута)

- **Intrinsic information** of a split
  - entropy of distribution of instances into branches
  - i.e. how much information do we need to tell which branch an instance belongs to

$$SplitI(D,V) = -\sum \frac{N_v}{N} \log_2 \frac{N_v}{N}$$

Ця характеристика показує – умовно скільки нам перевірок потрібно зробити, щоб вибрати наступну гілку в дереві

- Example:
  - Intrinsic information of a split for Data attribute:

$$SplitI(D,Data) = -14 \times \left(\frac{1}{14} \log_2 \frac{1}{14}\right) = 3.807$$

- Observation about information of a split: attributes with higher intrinsic less useful

Information Gain Ratio
(зважений приріст інформації)

- Modification of the information gain that reduces its bias towards multi-valued attributes
- Takes number and size of branches into account when choosing an attribute
  - Corrects the information gain by taking the intrinsic information of a split into account
  - Definition of Information Gain Ratio:

$$\boldsymbol{GainRatio(D,V)=\frac{Gain(D,V)}{SplitI(D,V)}}$$

- Example: Information Gain Ratio of Data attribute:

$$\boldsymbol{GainRatio(D,Data)=\frac{0.940}{3.807}=\color{red}{0.246}}$$

Gain ratios for our case “Nadal and weather”

Outlook		Temperature	
Info =	0.516	Info =	0.932
Inf. Gain = 0.940 – 0.516 =	0.424	Inf. Gain = 0.940 – 0.932 =	0.008
Split info = info([5; 4; 5]) =	1.577	Split info = info([4; 5; 5]) =	1.577
Inf. Gain ratio = 0.424/1.577=	0.269	Inf. Gain ratio = 0.008/1.577=	0.005
Humidity		**Wind**	
Info =	0.924	Info =	0.837
Inf. Gain = 0.940 – 0.924 =	0.016	Inf. Gain = 0.940 – 0.837 =	0.103
Split info = info([7; 7]) =	1	Split info = info([9; 5]) =	0.940
Inf. Gain ratio = 0.016/1 =	0.016	Inf. Gain ratio = 0.103/0.940=	0.110

- ❑ Information Gain Ratio for Outlook: 0.269 > 0.246
- ❑ Gain ratio is more reliable than Information Gain

Порівняння критеріїв розгалуження (=мір неоднорідності)

- Ентропія:

$$I_H(t) = -\sum_{i=1}^{c} p(i|t)\log_2 p(i|t).$$

- Неоднорідність (індекс) Джині, Gini impurity (index):

$$I_G(t) = \sum_{i=1}^{c} p(i|t)(1-p(i|t)) = 1 - \sum_{i=1}^{c} p(i|t)^2.$$

- Помилка класифікації:

$$I_E(t) = 1 - \max\{p(i|t)\}.$$

- p(i|t) – доля випадків, яка належить класу *i* для окремого вузла t

Порівняння критеріїв розгалуження (=мір неоднорідності)

- Ентропія:

$$I_H(t) = -\sum_{i=1}^{c} p(i|t)\log_2 p(i|t).$$

- Помилка класифікації:

$$I_E(t) = 1 - \max\{p(i|t)\}.$$

- p(i|t) – доля випадків, яка належить класу *i* для окремого вузла t

Розглянемо ситуацію: p (i |t)=0.2 і 0.8:
2 білі кульки та 8 чорних.
Тоді будемо всі кульки вважати чорними
==> помилка класифікації = 1-0,8 = 0,2.

А якщо буде 4 білі кульки та 6 чорних?

Порівняння критеріїв розгалуження (=мір неоднорідності)

- Ентропія:

$$I_H(t) = -\sum_{i=1}^{c} p(i|t)\log_2 p(i|t).$$

- p(i|t) – доля випадків, яка належить класу *i* для окремого вузла t

Розглянемо ситуацію: p (i |t)=0.2 і 0.8:
2 білі кульки та 8 чорних.
Тоді будемо всі кульки вважати чорними
==> помилка класифікації = 1-0,8 = 0,2.
А якщо буде 4 білі кульки та 6 чорних?

Розщеплюємо там, де краще розрізняємо колір кульок, тобто менше помилимося, якщо будемо вважати, що всі кульки одного (=найпопулярнішого у цьому вузлу) кольору

- Помилка класифікації:

$$I_E(t) = 1 - \max\{p(i|t)\}.$$

Порівняння критеріїв розгалуження (=мір неоднорідності)

- Ентропія:

$$I_H(t) = -\sum_{i=1}^{c} p(i|t)\log_2 p(i|t).$$

- Неоднорідність (індекс) Джині, Gini impurity (index):

$$I_G(t) = \sum_{i=1}^{c} p(i|t)(1 - p(i|t)) = 1 - \sum_{i=1}^{c} p(i|t)^2.$$

- Помилка класифікації:

$$I_E(t) = 1 - \max\{p(i|t)\}.$$

- **Приклад:**

p (1|t) =0.2

p (2|t) =0.8

Розщепляємо там, де сукупна помилка буде менше

I = 0.2*0.8+
+0.8*0.2 =
=0.32

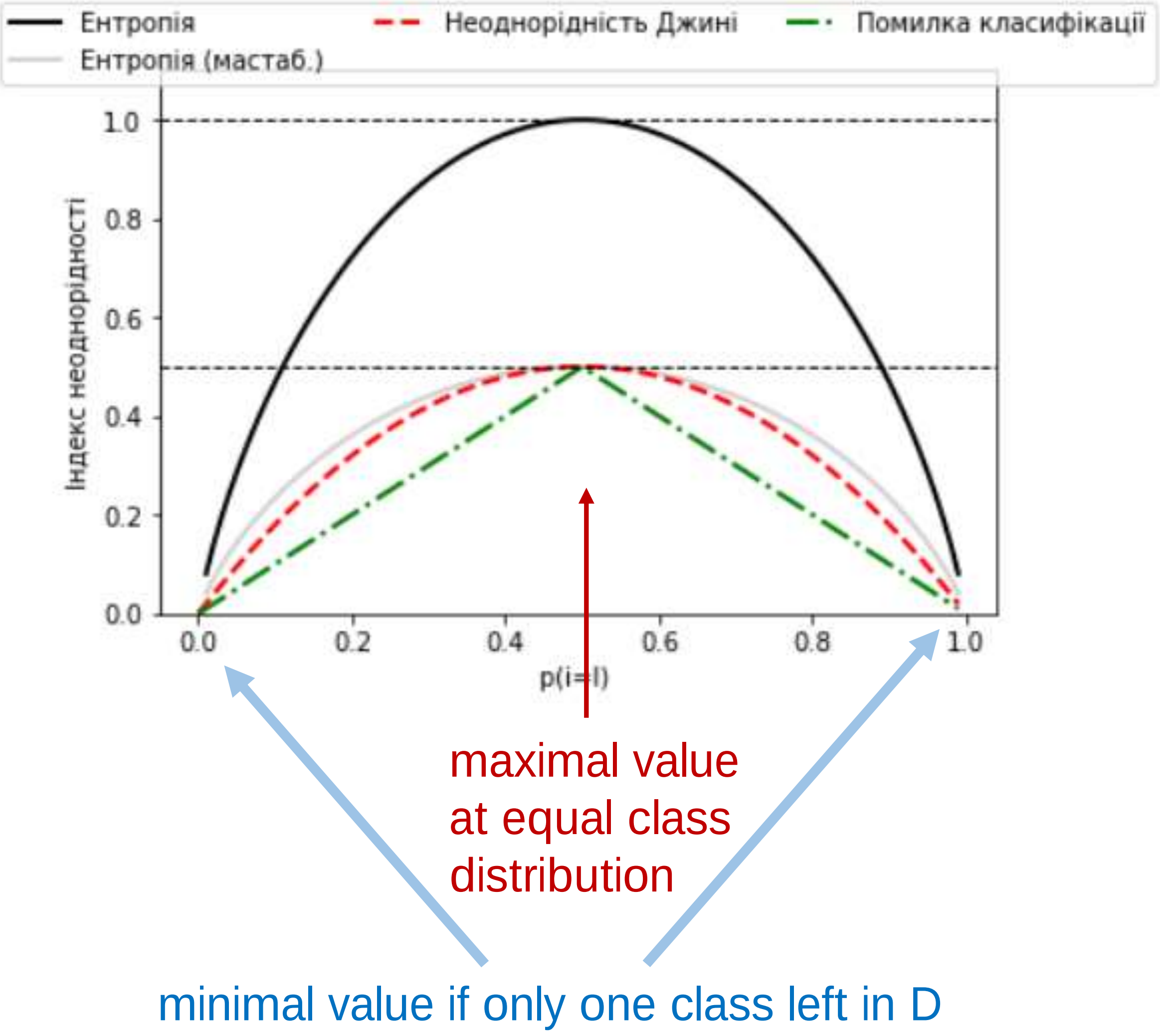
Розщепляємо там, де менше помилимося

I = 1-
- max(0.2;0.8) =
=0.2

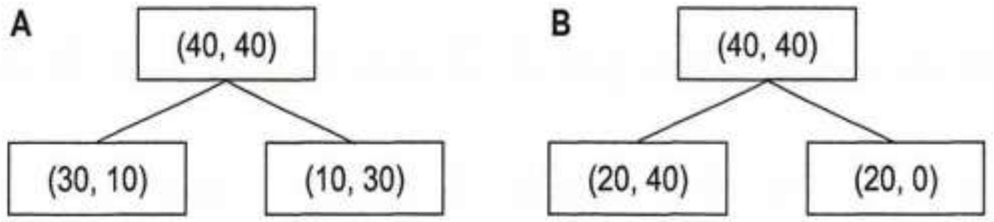
Порівняння критеріїв розгалуження для бінарного випадку

```python
import matplotlib.pyplot as plt
import numpy as np
def gini(p):
    return (p)*(1 - (p)) + (1 - p)*(1 - (1 - p))
def entropy(p):
    return - p*np.log2(p) - (1 - p)*np.log2((1 - p))
def error(p):
    return 1 - np.max([p, 1 - p])
x = np.arange (0.0, 1.0, 0.01)
ent = [entropy(p) if p != 0 else None for p in x]
sc_ent = [e*0.5 if e else None for e in ent]
err = [error(i) for i in x]
fig = plt.figure()
ax = plt.subplot(111)

for i, lab, ls, c, in zip ([ent, sc_ent, gini(x), err],
         ['Ентропія', 'Ентропія (мастаб.)',
          'Неоднорідність Джині', 'Помилка класифікації'],
         ['-', '-', '--', '-.'],
         ['black', 'lightgray', 'red', 'green', 'cyan']):
    line = ax.plot (x, i, label=lab, linestyle=ls, lw=2, color=c)
ax.legend(loc='upper center', bbox_to_anchor=(0.5, 1.15),
          ncol=3, fancybox=True, shadow=False)
ax.axhline(y=0.5, linewidth=1, color='k', linestyle='--')
ax.axhline(y=1.0 , linewidth=1, color='k', linestyle='--')
plt.ylim ([0, 1.1])
plt.xlabel ('p(i=l) ')
plt.ylabel('Індекс неоднорідності')
plt.show()
```



Помилка класифікації



$$I_E(t) = 1 - \max\{p(i|t)\}.$$

$$I_E(D_p) = 1 - 0.5 = 0.5;$$

$$A: I_E(D_{\text{левый}}) = 1 - \frac{3}{4} = 0.25;$$

$$A: I_E(D_{\text{правый}}) = 1 - \frac{3}{4} = 0.25;$$

$$A: IG_E = 0.5 - \frac{4}{8}0.25 - \frac{4}{8}0.25 = \boxed{0.25;}$$

$$B: I_E(D_{\text{левый}}) = 1 - \frac{4}{6} = \frac{1}{3};$$

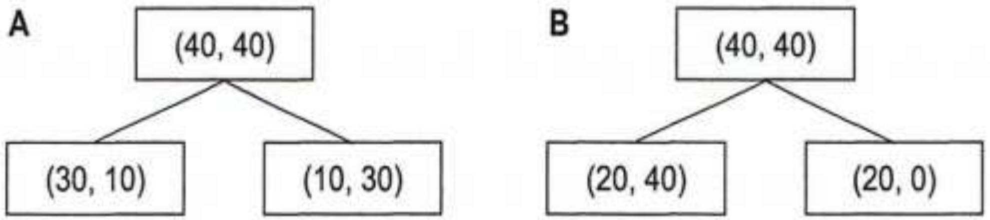
$$B: I_E(D_{\text{правый}}) = 1 - 1 = 0;$$

$$B: IG_E = 0.5 - \frac{6}{8} \times \frac{1}{3} - 0 = \boxed{0.25}$$

$$\boldsymbol{Gain(D,V) = I_E(D) - \sum \frac{N_v}{N} \times I_E(D_v)}$$

Погано –
не розрізняємо
ці дві ситуації

Ентропія



$$I_H(D_p) = -(0.5\log_2(0.5) + 0.5\log_2(0.5)) = 1;$$

$$A: I_H(D_{\text{левый}}) = -\left(\frac{3}{4}\log_2\left(\frac{3}{4}\right) + \frac{1}{4}\log_2\left(\frac{1}{4}\right)\right) = 0.81;$$

$$A: I_H(D_{\text{правый}}) = -\left(\frac{1}{4}\log_2\left(\frac{1}{4}\right) + \frac{3}{4}\log_2\left(\frac{3}{4}\right)\right) = 0.92;$$

$$A: IG_H = 1 - \frac{4}{8}0.81 - \frac{4}{8}0.81 = \boxed{0.19;}$$

$$B: I_H(D_{\text{левый}}) = -\left(\frac{2}{6}\log_2\left(\frac{2}{6}\right) + \frac{4}{6}\log_2\left(\frac{4}{6}\right)\right) = 0.92;$$

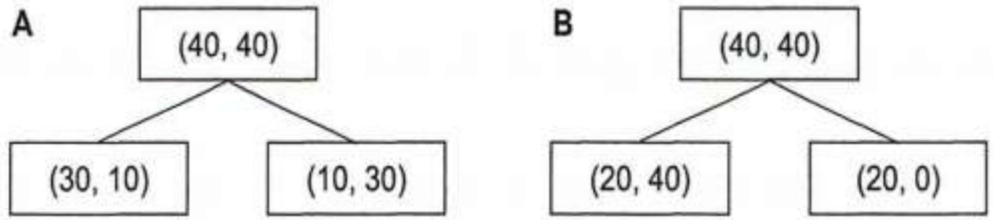
$$B: I_H(D_{\text{правый}}) = 0;$$

$$B: IG_H = 1 - \frac{6}{8}0.92 - 0 = \boxed{0.31}.$$

$$I_H(t) = -\sum_{i=1}^{c} p(i|t)\log_2 p(i|t).$$

$$\boldsymbol{Gain(D,V) = I_H(D) - \sum \frac{N_v}{N} \times I_H(D_v)}$$

Неоднорідність Джини



$$I_G(t)=\sum_{i=1}^{c}p(i|t)(1-p(i|t))=1-\sum_{i=1}^{c}p(i|t)^2.$$

$$\boldsymbol{Gain(D,V) = I_G(D) - \sum \frac{N_v}{N} \times I_G(D_v)}$$

$$I_G(D_p)=1-(0.5^2+0.5^2)=0.5;$$

$$A: I_G(D_{\text{левый}})=1-\left(\left(\frac{3}{4}\right)^2+\left(\frac{1}{4}\right)^2\right)=\frac{3}{8}=0.375;$$

$$A: I_G(D_{\text{правый}})=1-\left(\left(\frac{1}{4}\right)^2+\left(\frac{3}{4}\right)^2\right)=\frac{3}{8}=0.375;$$

$$A: IG_G=0.5-\frac{4}{8}0.375-\frac{4}{8}0.375=\boxed{0.125;}$$

$$B: I_G(D_{\text{левый}})=1-\left(\left(\frac{2}{6}\right)^2+\left(\frac{4}{6}\right)^2\right)=\frac{4}{9}=0.\overline{4};$$

$$B: I_G(D_{\text{правый}})=1-(1^2+0^2)=0;$$

$$B: IG_G=0.5-\frac{6}{8}0.\overline{4}-0=\boxed{0.1\overline{6}.}$$

Порівняння критеріїв розгалуження

Let us first recall that the Gini's coefficient is a particular case of the Havrda and Charvat's α-entropy (Havrda and Charvat (1967)). Let X_R be a discrete random variable with a probability distribution for its k values defined by $(p_1, p_2, ..., p_k)$. The α-entropy is defined by

$$H_\alpha(R) = \frac{1}{1-\alpha}\left(\sum_{i=1}^{k} p_i^\alpha - 1\right)$$

The case $\alpha = 2$ corresponds to the Gini's coefficient: $Gini(R) = 1 - \sum_i p_i^2$.

Moreover, when α tends toward 1, the limit of the α-entropy is the Shannon's entropy. Thus, the semantic of these two coefficients are close.

Interpretation 1. The Gini's coefficient can be interpreted as a variance of Bernoulli independent variables of respective parameters $p_1, p_2, ..., p_k$. Indeed, it is easy to show that $1 - \sum_i p_i^2 = \sum_i p_i(1-p_i)$.

Interpretation 2. The Gini's coefficient $1 - \sum_i p_i^2$ can be interpreted as a distance between the norms of two k-dimensional vectors: the components of the first vector are equal to 1 and those of the second one are equal to $p_1, p_2, ..., p_k$.

- Havrda, J.-H. and Charvat, F. (1967): Quantification methods of classification processes. Concepts of structural entropy - Kybernetika, 3, 30-37
- Gras R., Kuntz P. (2007). Reduction of Redundant Rules in Statistical Implicative Analysis. Selected Contributions in Data Analysis and Classification, pp. 367-376

Порівняння критеріїв розгалуження

- Generally, your performance will not change whether you use Gini impurity or Entropy
- Laura Elena Raileanu and Kilian Stoffel compared both in "Theoretical comparison between the Gini Index and Information Gain criteria“ (2004):
  - It only matters in 2% of the cases whether you use Gini impurity or entropy
  - Entropy might be a little slower to compute (because it makes use of the logarithm)
- Split criterion: Information Gain (ID3; C4.5)

  Gini index (CART; IBM IntelligentMiner)

Computing GINI Index

$$GINI(t) = 1 - \sum_j [p(j|t)]^2$$

- GINI Index for a given node t
- p(j|t) is the relative frequency of j at node t

C1	0
C2	6

P(C1) = 0/6 = 0, P(C2) = 6/6=1

GINI = 1 - $P(C1)^2$ - $P(C2)^2$ = 1 - 0 - 1 = 0

Minimum 0: all records in one class

C1	2
C2	4

P(C1) = 2/6 = 1/3, P(C2) = 4/6=2/3

GINI = 1 - $(1/3)^2$ - $(2/3)^2$ = 0.444

C1	3
C2	3

P(C1) = 3/6 = 1/2, P(C2) = 3/6 = 1/2

GINI = 1 - $(1/2)^2$ - $(1/2)^2$ = 1 - 1/4 - 1/4 = 0.5

Maximum = $1-n/n^2 = 1-1/n$:
records equally distributed in n classes

Finding the Best Split: GINI Index

When a node t is split into k partitions (children), the quality of split is computed as:

$$GINI_{split} = \sum_{i=1}^{k} \frac{n_i}{n} GINI(i)$$

n_i = number of records at child i,
n = number of records at node t

Computing GINI Index for Categorical Attributes (in Nadal case)

Nadal training statistics (GINI Index)

Step 1: Find the Gini for each predictor:

Wind?	Weak	Strong
Play	7	2
Stay	2	3

Day	Wind	Temp	Outlook	Humidity	Action
Day 1	Weak	Hot	Sunny	High	Play (P)
Day 2	Strong	Hot	Sunny	High	Play
Day 3	Weak	Hot	Rain	High	Stay (S)
Day 4	Weak	Mid	Overcast	High	Play
Day 5	Strong	Cold	Rain	Normal	Stay
Day 6	Weak	Cold	Overcast	Normal	Play
Day 7	Strong	Cold	Rain	Normal	Stay
Day 8	Weak	Mid	Sunny	Normal	Play
Day 9	Weak	Cold	Sunny	Normal	Play
Day 10	Strong	Mid	Overcast	Normal	Play
Day 11	Weak	Mid	Sunny	High	Stay
Day 12	Strong	Mid	Rain	High	Stay
Day 13	Weak	Hot	Overcast	Normal	Play
Day 14	Weak	Cold	Rain	High	Play

Computing GINI Index for Categorical Attributes (in Nadal case)

Nadal training statistics (GINI Index)

Step 1: Find the Gini for each predictor:

Gini Index For Wind

$$GINI(Weak) = 1 - P(play)^2 - P(stay)^2$$
$$= 1 - (7/9)^2 - (2/9)^2$$
$$= 0.346$$

Wind?	Weak	Strong
Play	7	2
Stay	2	3

Day	Wind	Temp	Outlook	Humidity	Action
Day 1	**Weak**	**Hot**	**Sunny**	**High**	**Play (P)**
Day 2	Strong	Hot	Sunny	High	Play
Day 3	**Weak**	**Hot**	**Rain**	**High**	**Stay (S)**
Day 4	**Weak**	**Mid**	**Overcast**	**High**	**Play**
Day 5	Strong	Cold	Rain	Normal	Stay
Day 6	**Weak**	**Cold**	**Overcast**	**Normal**	**Play**
Day 7	Strong	Cold	Rain	Normal	Stay
Day 8	**Weak**	**Mid**	**Sunny**	**Normal**	**Play**
Day 9	**Weak**	**Cold**	**Sunny**	**Normal**	**Play**
Day 10	Strong	Mid	Overcast	Normal	Play
Day 11	**Weak**	**Mid**	**Sunny**	**High**	**Stay**
Day 12	Strong	Mid	Rain	High	Stay
Day 13	**Weak**	**Hot**	**Overcast**	**Normal**	**Play**
Day 14	**Weak**	**Cold**	**Rain**	**High**	**Play**

GINI Index

Nadal training statistics (GINI Index)

Step 1: Find the Gini for each predictor:

Gini Index For Wind

$$\text{GINI(Weak)} = 1 - P(\text{play})^2 - P(\text{stay})^2$$
$$= 1 - (7/9)^2 - (2/9)^2$$
$$= 0.346$$

$$\text{GINI(Strong)} = 1 - P(\text{play})^2 - P(\text{stay})^2$$
$$= 1 - (2/5)^2 - (3/5)^2$$
$$= 0.480$$

Wind?	Weak	Strong
Play	7	2
Stay	2	3

Day	Wind	Temp	Outlook	Humidity	Action
Day 1	Weak	Hot	Sunny	High	Play (P)
Day 2	**Strong**	**Hot**	**Sunny**	**High**	**Play**
Day 3	Weak	Hot	Rain	High	Stay (S)
Day 4	Weak	Mid	Overcast	High	Play
Day 5	**Strong**	**Cold**	**Rain**	**Normal**	**Stay**
Day 6	Weak	Cold	Overcast	Normal	Play
Day 7	**Strong**	**Cold**	**Rain**	**Normal**	**Stay**
Day 8	Weak	Mid	Sunny	Normal	Play
Day 9	Weak	Cold	Sunny	Normal	Play
Day 10	**Strong**	**Mid**	**Overcast**	**Normal**	**Play**
Day 11	Weak	Mid	Sunny	High	Stay
Day 12	**Strong**	**Mid**	**Rain**	**High**	**Stay**
Day 13	Weak	Hot	Overcast	Normal	Play
Day 14	Weak	Cold	Rain	High	Play

GINI Index

Nadal training statistics (GINI Index)

Step 1: Find the Gini for each predictor:

Gini Index For Wind

GINI(Weak) = 1 - P(play)^2 - P(stay)^2

= 1 - (7/9)^2 - (2/9)^2

= 0.346

GINI(Strong) = 1 - P(play)^2 - P(stay)^2

= 1 - (2/5)^2 - (3/5)^2

= 0.480

$$GINI_{split} = \sum_{i=1}^{k} \frac{n_i}{n} GINI(i)$$

GINI-Split(Wind) = (9/14) * 0.346 + (5/14) * 0.480 =
= 0.393

Day	Wind	Temp	Outlook	Humidity	Action
Day 1	Weak	Hot	Sunny	High	Play (P)
Day 2	**Strong**	**Hot**	**Sunny**	**High**	**Play**
Day 3	Weak	Hot	Rain	High	Stay (S)
Day 4	Weak	Mid	Overcast	High	Play
Day 5	**Strong**	**Cold**	**Rain**	**Normal**	**Stay**
Day 6	Weak	Cold	Overcast	Normal	Play
Day 7	**Strong**	**Cold**	**Rain**	**Normal**	**Stay**
Day 8	Weak	Mid	Sunny	Normal	Play
Day 9	Weak	Cold	Sunny	Normal	Play
Day 10	**Strong**	**Mid**	**Overcast**	**Normal**	**Play**
Day 11	Weak	Mid	Sunny	High	Stay
Day 12	**Strong**	**Mid**	**Rain**	**High**	**Stay**
Day 13	Weak	Hot	Overcast	Normal	Play
Day 14	Weak	Cold	Rain	High	Play

GINI Index

Nadal training statistics (GINI Index)

Step 1: Find the Gini for each predictor:

Gini Index For Temp

$$GINI(Hot) = 1 - P(play)^2 - P(stay)^2$$

$$= 1 - (3/4)^2 - (1/4)^2$$

$$= 0.375$$

$$GINI(Mid) = 1 - (3/5)^2 - (2/5)^2$$

$$= 0.48$$

$$GINI(Cold) = 1 - (3/5)^2 - (2/5)^2$$

$$= 0.48$$

$$GINI\text{-}Split(Temp) = 0.375 * (4/14) + 0.48 * (5/14) + 0.48 * (5/14) = 0.450$$

Day	Wind	Temp	Outlook	Humidity	Action
Day 1	Weak	Hot	Sunny	High	Play (P)
Day 2	Strong	Hot	Sunny	High	Play
Day 3	Weak	Hot	Rain	High	Stay (S)
Day 4	Weak	Mid	Overcast	High	Play
Day 5	**Strong**	**Cold**	**Rain**	**Normal**	**Stay**
Day 6	**Weak**	**Cold**	**Overcast**	**Normal**	**Play**
Day 7	**Strong**	**Cold**	**Rain**	**Normal**	**Stay**
Day 8	Weak	Mid	Sunny	Normal	Play
Day 9	**Weak**	**Cold**	**Sunny**	**Normal**	**Play**
Day 10	Strong	Mid	Overcast	Normal	Play
Day 11	Weak	Mid	Sunny	High	Stay
Day 12	Strong	Mid	Rain	High	Stay
Day 13	Weak	Hot	Overcast	Normal	Play
Day 14	**Weak**	**Cold**	**Rain**	**High**	**Play**

GINI Index

Nadal training statistics (GINI Index)

Step 1: Find the Gini for each predictor:

Gini Index For Outlook

$$GINI(Sunny) = 1 - P(play)^2 - P(stay)^2$$
$$= 1 - (4/5)^2 - (1/5)^2$$
$$= 0.32$$

$$GINI(Rain) = 1 - (1/5)^2 - (4/5)^2$$
$$= 0.32$$

$$GINI(Over..) = 1 - (4/4)^2 - (0/4)^2$$
$$= 0$$

GINI-**Split(O'look)** = 0.320 * (5/14) + 0.32 * (5/14) + 0 *(4/14) = 0.229

Day	Wind	Temp	Outlook	Humidity	Action
Day 1	Weak	Hot	Sunny	High	Play (P)
Day 2	Strong	Hot	Sunny	High	Play
Day 3	Weak	Hot	Rain	High	Stay (S)
Day 4	**Weak**	**Mid**	**Overcast**	**High**	**Play**
Day 5	Strong	Cold	Rain	Normal	Stay
Day 6	**Weak**	**Cold**	**Overcast**	**Normal**	**Play**
Day 7	Strong	Cold	Rain	Normal	Stay
Day 8	Weak	Mid	Sunny	Normal	Play
Day 9	Weak	Cold	Sunny	Normal	Play
Day 10	**Strong**	**Mid**	**Overcast**	**Normal**	**Play**
Day 11	Weak	Mid	Sunny	High	Stay
Day 12	Strong	Mid	Rain	High	Stay
Day 13	**Weak**	**Hot**	**Overcast**	**Normal**	**Play**
Day 14	Weak	Cold	Rain	High	Play

GINI Index

Nadal training statistics (GINI Index)

Step 1: Find the Gini for each predictor:

Gini Index For Humidity

$$GINI(High) = 1 - P(play)^2 - P(stay)^2$$

$$= 1 - (4/7)^2 - (3/7)^2$$

$$= 0.490$$

$$GINI(Norm.) = 1 - (5/7)^2 - (2/7)^2$$

$$= 0.408$$

$$GINI\text{-}Split(Hum.) = 0.490 * (7/14) + +0.408 * (7/14) = 0.449$$

Day	Wind	Temp	Outlook	Humidity	Action
Day 1	Weak	Hot	Sunny	High	Play (P)
Day 2	Strong	Hot	Sunny	High	Play
Day 3	Weak	Hot	Rain	High	Stay (S)
Day 4	Weak	Mid	Overcast	High	Play
Day 5	**Strong**	**Cold**	**Rain**	**Normal**	**Stay**
Day 6	**Weak**	**Cold**	**Overcast**	**Normal**	**Play**
Day 7	**Strong**	**Cold**	**Rain**	**Normal**	**Stay**
Day 8	**Weak**	**Mid**	**Sunny**	**Normal**	**Play**
Day 9	**Weak**	**Cold**	**Sunny**	**Normal**	**Play**
Day 10	**Strong**	**Mid**	**Overcast**	**Normal**	**Play**
Day 11	Weak	Mid	Sunny	High	Stay
Day 12	Strong	Mid	Rain	High	Stay
Day 13	**Weak**	**Hot**	**Overcast**	**Normal**	**Play**
Day 14	Weak	Cold	Rain	High	Play

GINI Index

Step 1: Find the Gini for each predictor:

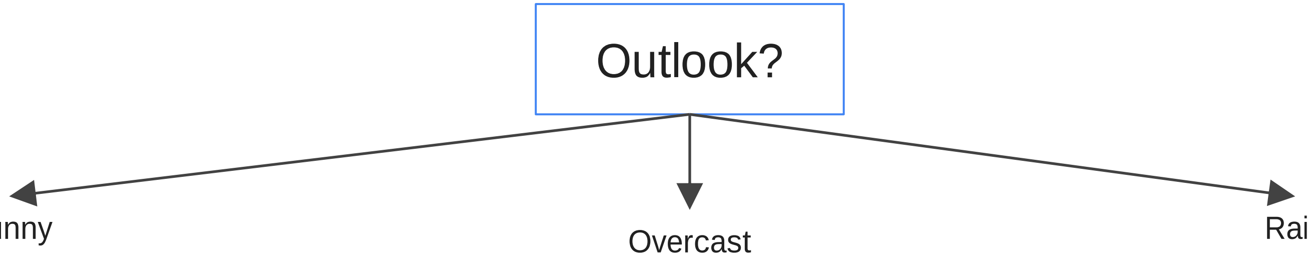
Step 2: Find the node's best split:

	Wind	Temp	Outlook	Humidity
Gini Split	0.394	0.450	0.229	0.449

The best split will use the outlook attribute

GINI Index

Step 3: Split the node



Sunny

Day	Wnd	Tmp	Hu.	
1	W	H	H	P
2	S	H	H	P
8	W	M	N	P
9	W	C	N	P
11	W	M	H	S

Impure Set repeat from Step 1

Overcast

Day	Wnd	Tmp	Hu.	
4	W	M	H	P
6	W	C	N	P
10	S	M	N	P
13	W	H	N	P

No further splitting (Pure set)

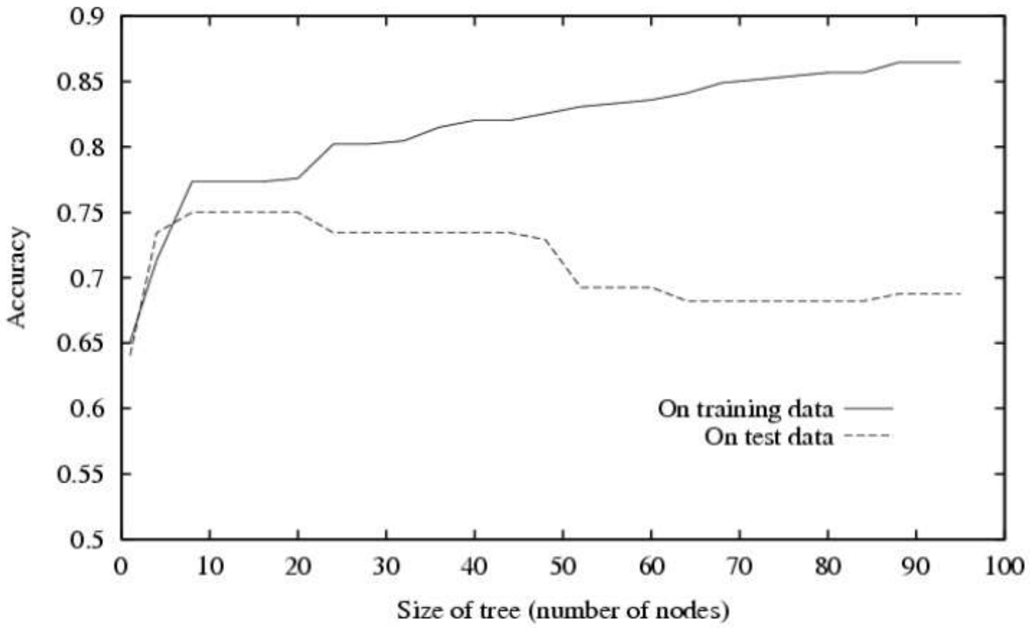
Rain

Day	Wnd	Tmp	Hu.	
3	W	H	H	S
5	S	C	N	S
7	S	C	N	S
12	S	M	H	S
14	W	C	H	P

Impure Set repeat from Step 1

Avoid overfitting in classification

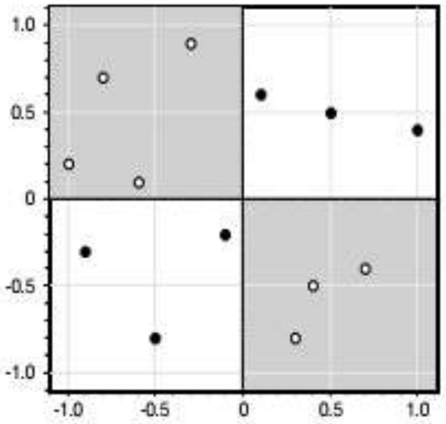
- Overfitting: An induced tree may overfit the training data
  - ❖ Too many branches, some may reflect anomalies due to noise or outliers
  - ❖ Poor accuracy for unseen samples

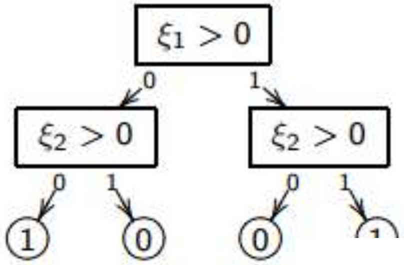


Жадібні алгоритми можуть невиправдано ускладнювати дерева

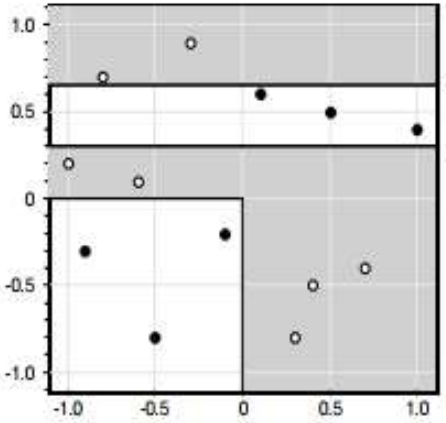
Задача виключного АБО (**XOR**)

Не розв'язується лінійним класифікатором (не може бути розділена прямою на два класи без помилок

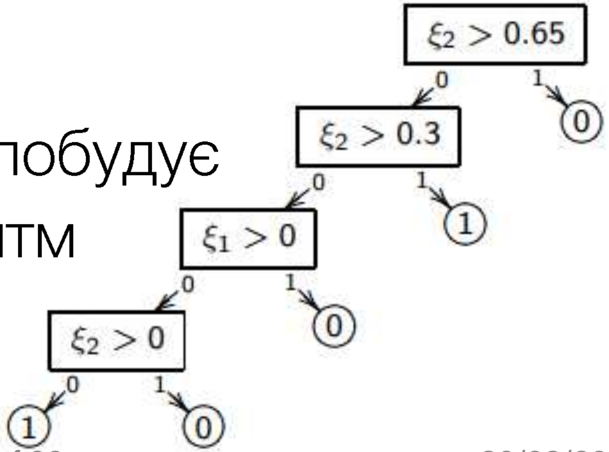




Оптимальне дерево



Дерево, яке побудує жадібний алгоритм



Як здогадатися, що спочатку потрібно провести лінію посередині?!

Avoid overfitting in classification

- Two approaches to avoid overfitting:
  - Prepruning: Halt tree construction early — do not split a node if this would result in the goodness measure falling below a threshold
    - Difficult to choose an appropriate threshold
  - Postpruning: Remove branches from a "fully grown" tree — get a sequence of progressively pruned trees
    - Use a set of data different from the training data to decide which is the "best pruned tree"
- Postpruning preferred in practice—prepruning can "stop early", but prepruning is faster than postpruning

Prepruning

- Based on statistical significance test
  - Stop growing the tree when there is no statistically significant association between any attribute and the class at a particular node
- Most popular test: chi-squared test
- ID3 used chi-squared test in addition to information gain
  - Only statistically significant attributes were allowed to be selected by information gain procedure

- *В статистиці вважається, що критерій хі-квадрат Пірсона можна застосовувати, якщо загальний розмір вибірки перевищує 50, а кількість об'єктів певного класу – більша 5*
- *Інакше можна застосувати корекцію Йетса* (Yates' correction):
  - *відкинувши низькочастотні події або об'єднавши їх з іншими подіями*

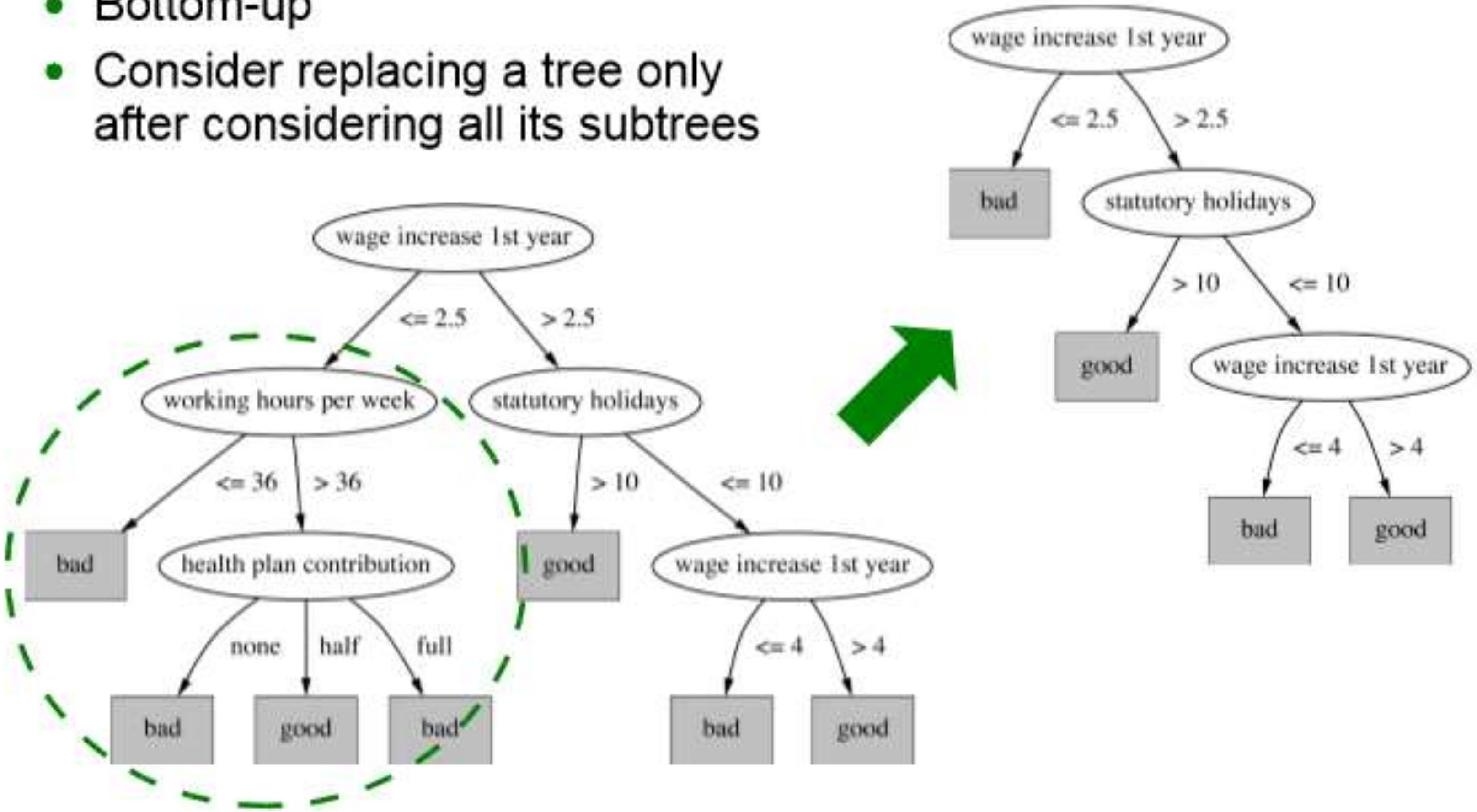
Прості способи попереднього обрізання дерев (**prepruning**)

- Обмеження глибини дерева. У цьому випадку побудова закінчується, якщо досягнуто задану глибину

- Визначення мінімальної кількості прикладів, які будуть міститися в кінцевих вузлах дерева. При цьому варіанті розгалуження тривають до того моменту, поки всі кінцеві вузли дерева не будуть чистими (=відноситимуться до одного класу) або будуть містити не більше, ніж задане число об'єктів

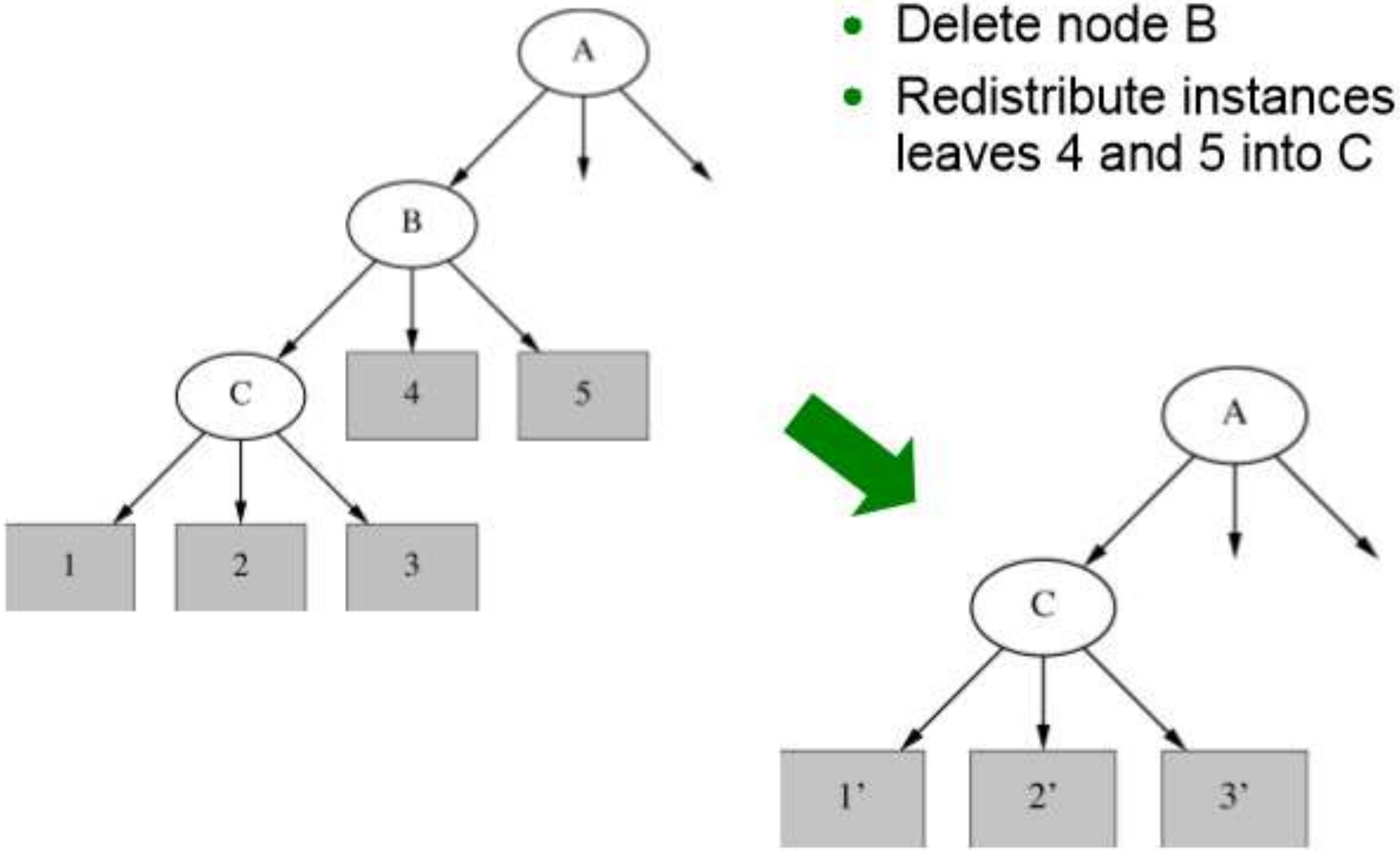
Postpruning: Subtree replacement

- Bottom-up
- Consider replacing a tree only after considering all its subtrees



Postpruning: Subtree raising

- Delete node B
- Redistribute instances of leaves 4 and 5 into C



Різні методи побудови дерев прийняття рішень

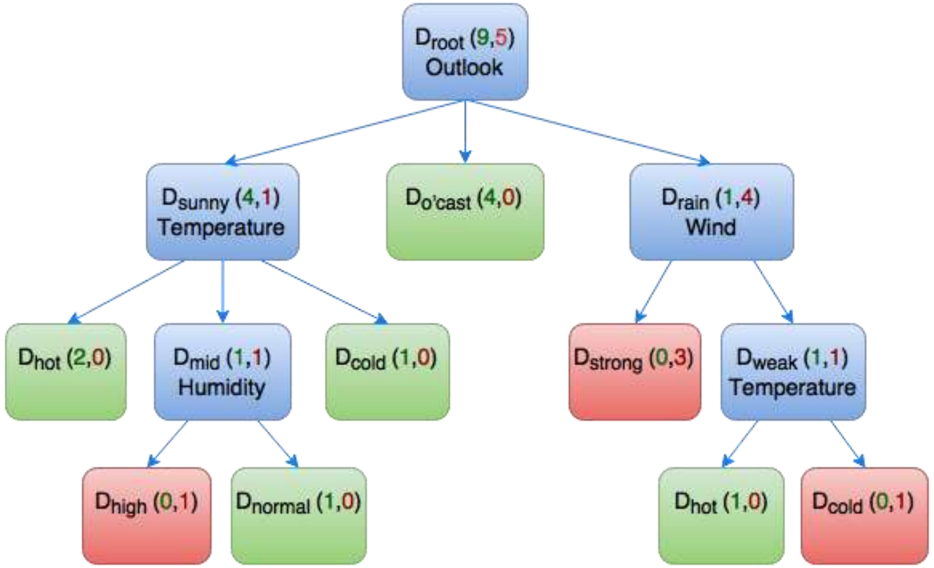
- ID3, C4.5, C5.0
- CART
- CHAID
- …

ID3 Algorithm

- ID3 (Iterative Dichotomiser 3) is an algorithm invented in 1986 by John Quinlan (Джоном Квінланом), Australian computer scientist, 1943-

- ID3 algorithm uses Information Gain methodology to create a tree:
  - This decision tree is used to classify new unseen test cases by working down the decision tree using the values of this test case to arrive at a terminal node that tells you what class this test case belongs to

ID3 Pseudo Code



Function ID3

- Input: Example set S
- Output: Decision Tree DT

If all examples in S belong to the same class c

- return a new leaf and label it with c

Else

- i. Select an attribute A according to some heuristic function

(If number of predicting attributes is empty, then return a new leaf with label = class that has the most values in the set S)

- ii. Generate a new node DT with A as test
- iii. For each value v_i of A
  - (a) Let S_i = all examples in S with A = v_i
  - (b) Use ID3 to construct a decision tree DT_i for example set S_i
  - (c) Generate an edge that connects DT and DT_i

Алгоритм C4.5 – не такий жадібний як ID3

- Автор алгоритму то й, що і у ID3 (Iterative Dichotomiser 3) – John Quinlan (Джон Квінлан):

J.R. Quinlan. C4.5: programs for machine learning. Morgan Kaufmann, 1993

- Є доступним його код:

https://www.rulequest.com/Personal/

- У подальшому відкрили доступ і до коду комерційного алгоритму C5.0:

https://www.rulequest.com/download.html

- Ідея алгоритму C4.5: якщо наша проблема полягає у перенавчанні, в ускладненні структури дерева, то на додатковій контрольній вибірці даних (test set) оцінимо узагальнюючу здатність побудованого дерева

Алгоритм C4.5 – не такий жадібний як ID3 (процедура підрізання дерев)

X^k — независимая контрольная выборка, $k \approx 0.5\ell$.

1: **для всех** $v \in V_{\text{внутр}}$
2: $S_v :=$ подмножество объектов X^k, дошедших до v;
3: **если** $S_v = \varnothing$ **то**
4: **вернуть** новый лист v, $c_v :=$ Мажоритарный класс(U);
5: число ошибок при классификации S_v четырьмя способами:
$r(v)$ — поддеревом, растущим из вершины v;
$r_L(v)$ — поддеревом левой дочерней вершины L_v;
$r_R(v)$ — поддеревом правой дочерней вершины R_v;
$r_c(v)$ — к классу $c \in Y$.
6: в зависимости от того, какое из них минимально:
сохранить поддерево v;
заменить поддерево v поддеревом L_v;
заменить поддерево v поддеревом R_v;
заменить поддерево v листом, $c_v := \arg\min_{c \in Y} r_c(v)$.

- Розіб'ємо навчальну вибірку на дві частини: навчальну та контрольну (зазвичай, роблять контрольну удвічі меншою за навчальну)
- По навчальній вибірці за допомогою ID3 будується дерево, а контрольну вибірку використовують, щоб це дерево спростити
- Порядок перегляду всіх вершин може бути різним, залежно від реалізації алгоритму: можна йти зверху вниз, знизу вгору або йти по внутрішнім вершин в якомусь випадковому порядку

Алгоритм CART (Classification and Regression Tree)

- Цей алгоритм реалізовано в scikit-learn

- Як алгоритм адаптується під розв'язання регресійної задачі?
- Якщо це лист дерева, то обираємо не самий поширений клас об'єктів із навчальної вибірки у цьому місті дерева (як в ID3 чи C4.5), а середньо арифметичне значення об'єктів із навчальної вибірки у цьому листі (відповідає оптимальному розв'язку методом найменших квадратів)
- Якщо це вузол, то критерій розгалуженості – це середньо квадратична помилка, тобто той самий критерій, що й в МНК

L. Breiman, J. Friedman, R. Olshen, and C. Stone. Classification and Regression Trees. Wadsworth, Belmont, CA, 1984

Алгоритм CART: критерій Minimal Cost-Complexity Pruning

- Середньо квадратична помилка зі штрафом за складність дерева:

$$C_\alpha = \sum_{x_i=1}^{\ell} (\hat{y}_i - y_i)^2 + \alpha |V_{\text{лист}}| \to \min$$

- При збільшенні коефіцієнта **α** (=константи регуляризації) дерево послідовно спрощується
- Причому послідовність вкладених дерев єдина
- З цієї послідовності вибирається дерево з мінімальною помилкою на тестовій вибірці (HoldOut)
- Для випадку класифікації використовується аналогічна стратегія обрізання (усічення), але з критерієм Джині

Обробка пропущених значень деревами прийняття рішень

- **На стадії навчання**

- **На стадії застосування**

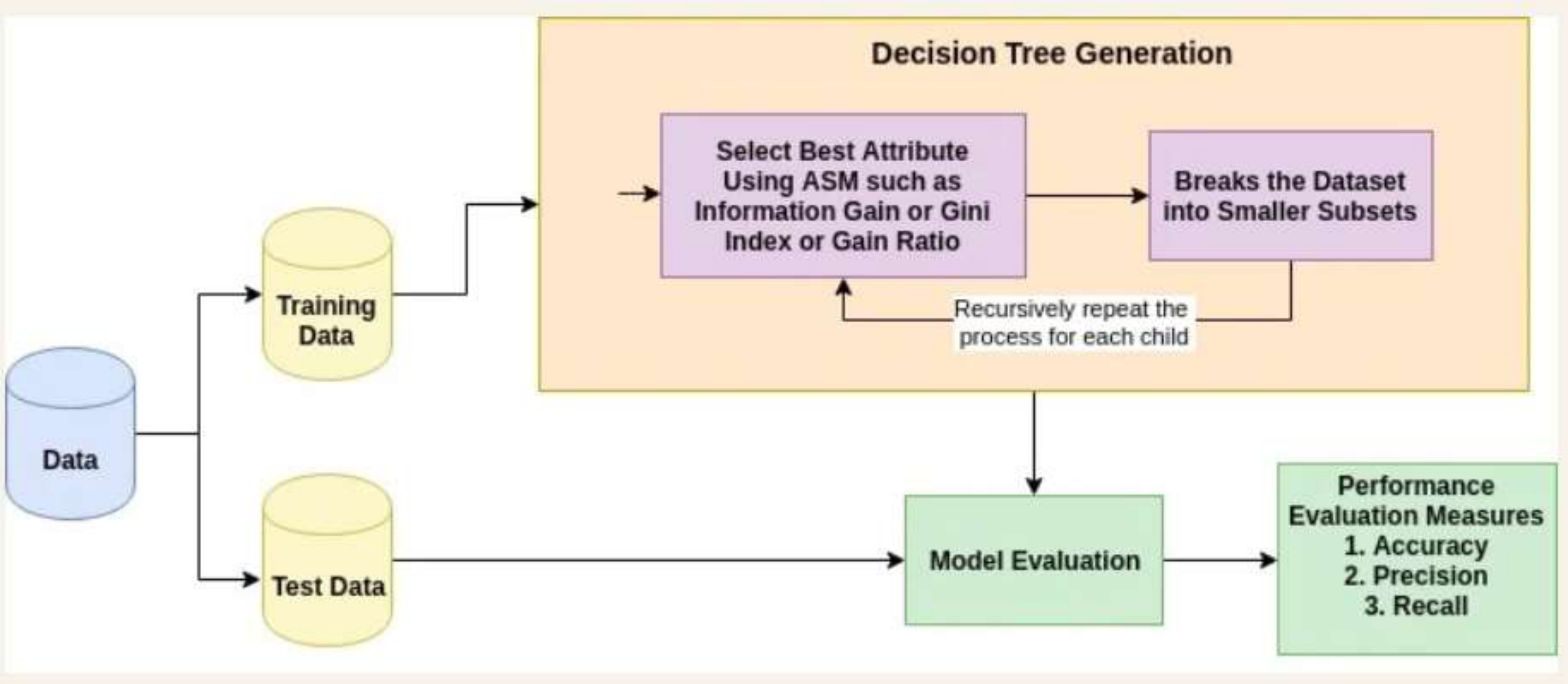
Переваги і недоліки дерев прийняття рішень

- Переваги:
  - Інтерпретовність
  - Дозволяються різні типи даних
  - Можливість обходу пропущених значень
  - Лінійна трудомісткість як по розміру навчальної вибірки, так і по кількості атрибутів
  - Відсутність відмови від класифікації (будь-який об’єкт буде класифіковано)

- Недоліки (із-за жадібності алгоритмів побудови дерев):
  - Перенавчання, що приводить до ускладненої структури дерева
  - Фрагментація вибірки (чим далі від кореня дерева, тим менш статистично надійним є вибір наступних вузлів, бо він робиться по малій вибірці)
  - Нестійкість до шуму, складу вибірки, критерію розгалуження (вилучення навіть одного об’єкта із навчальної вибірки може привести до побудови зовсім іншого дерева)

- Способи зниження впливу недоліків
  - Обрізання дерев
  - Композиція (=ліс) дерев

How does the Decision Tree algorithm work?



Validating the Data Model

- The main task of learning algorithms is their ability to generalize, that is, to work well on new data.

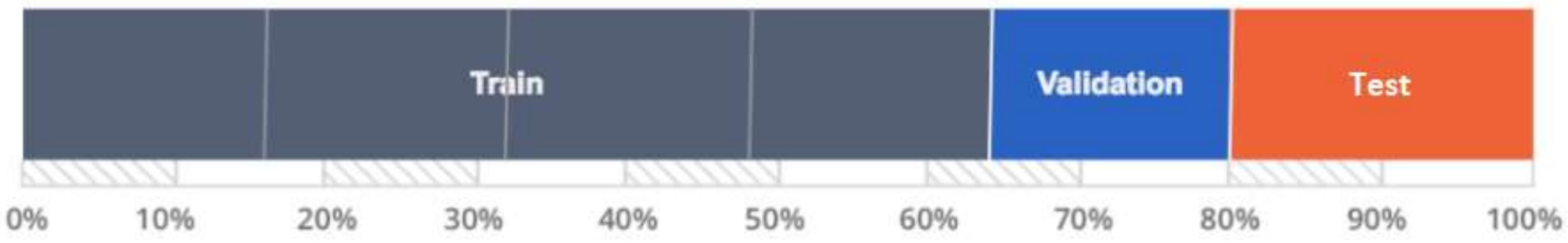
- Since we cannot immediately check the quality of the constructed model on the new data, we need to sacrifice a small portion of data in order to check the quality of the model on it.

- We want our model to generalize well without overfitting. We can ensure this by validating the model.

Partitioning Data

- The first step in developing a machine learning model is training and validation. In order to train and validate a model, you must first partition your dataset, which involves choosing what percentage of your data to use for the training, validation, and test (=holdout) sets.

- The following example shows a dataset with 64% training data, 16% validation data, and 20% test data.



Partitioning Data

❑ What is a Training Set? (for learning!)

A training set (навчальна вибірка) is the subsection of a dataset from which the machine learning algorithm uncovers, or “learns,” relationships between the features and the target variable. In supervised machine learning, training data is labeled with known outcomes.

❑ What is a Validation Set? (for tuning or selecting!)

A validation set (контрольна вибірка) is another subset of the input data to which we apply the machine learning algorithm to see how accurately it identifies relationships between the known outcomes for the target variable and the dataset’s other features.

❑ What is a Test Set? (for estimating!)

Sometimes referred to as “holdout” data (відкладена вибірка), a testing subset (тестова вибірка) provides a final estimate of the machine learning model’s performance after it has been trained and validated. Testing sets should never be used to make decisions about which algorithms to use or for improving or tuning algorithms.

Choosing model parameters and cross-validation

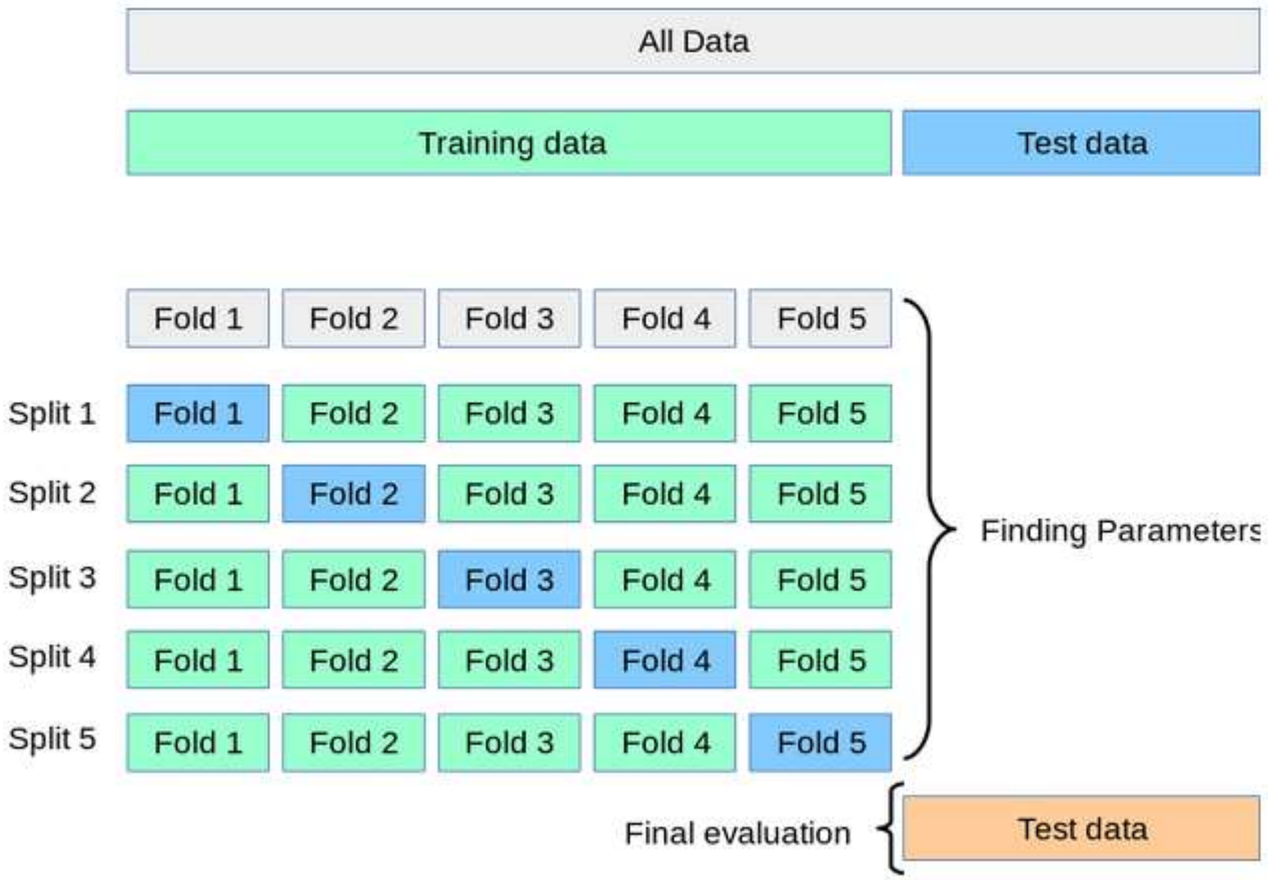
- This is most commonly done in one of 2 ways:

1. Hold-out set (**відкладена вибірка**). With this approach, we leave some fraction of the training sample (usually from 20% to 40%), train the model on the rest of the data (60-80% of the original sample) and calculate some metric of the model's quality on a held-out set:

```
# 70% training and 30% test
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=1)
```

However, by partitioning the available data into two or three sets, we drastically reduce the number of samples which can be used for learning the model, and the results can depend on a particular random choice for the pair of (train, validation) sets.

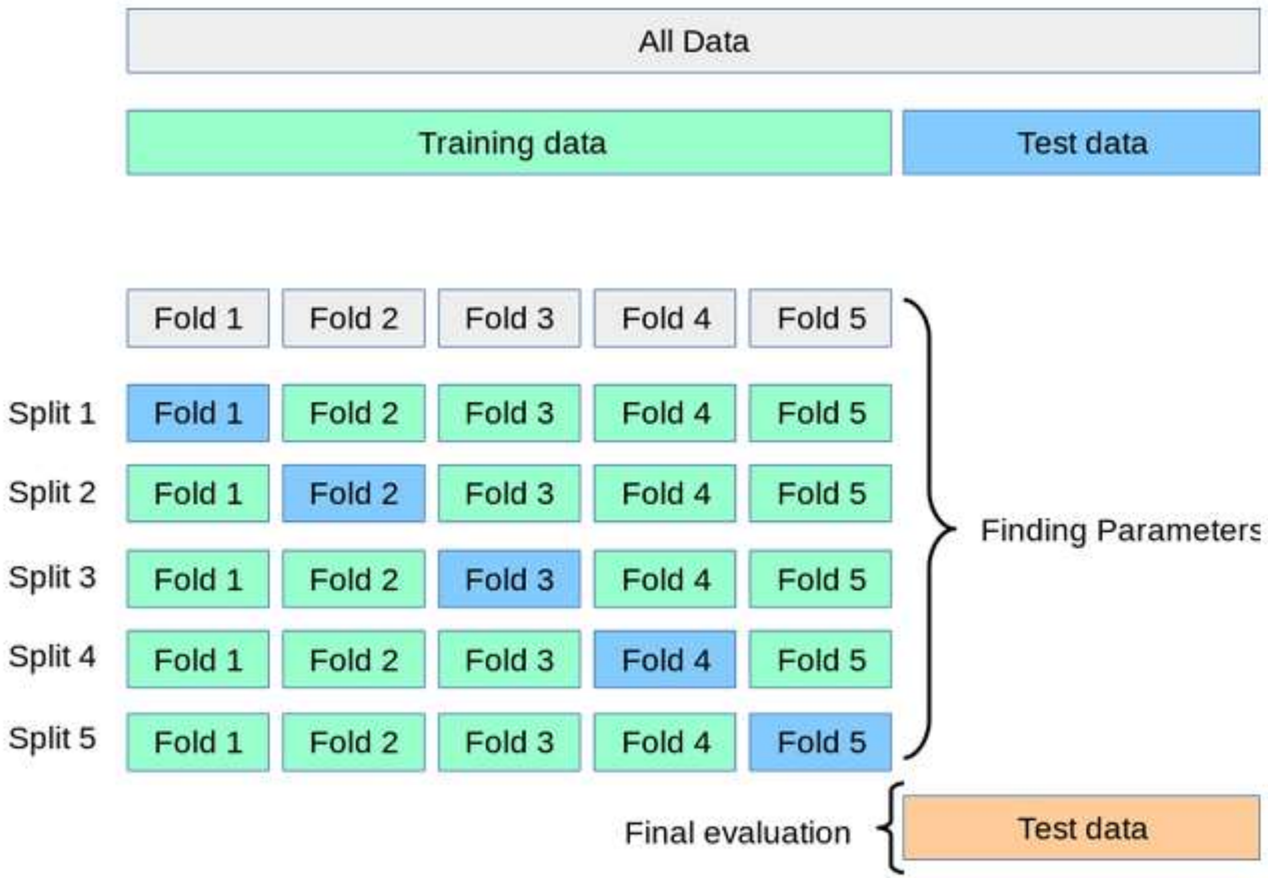
Choosing model parameters and cross-validation



2. Cross-validation (**крос-валідація або перехресна** (**ковзна**) **перевірка**).

A solution to this problem is a procedure called cross-validation (CV for short). A test set should still be held out for final evaluation, but the validation set is no longer needed when doing CV. In the basic approach, called k-fold CV, the training set is split into k smaller sets.

Choosing model parameters and cross-validation



The following procedure is followed for each of the k “folds”:

- A model is trained using k-1 of the folds as training data;
- the resulting model is validated on the remaining part of the data (i.e., it is used as a validation set to compute a performance measure such as accuracy).

The performance measure reported by k-fold cross-validation is then the average of the values computed in the loop.

This approach can be computationally expensive, but does not waste too much data (as is the case when fixing an arbitrary validation set)

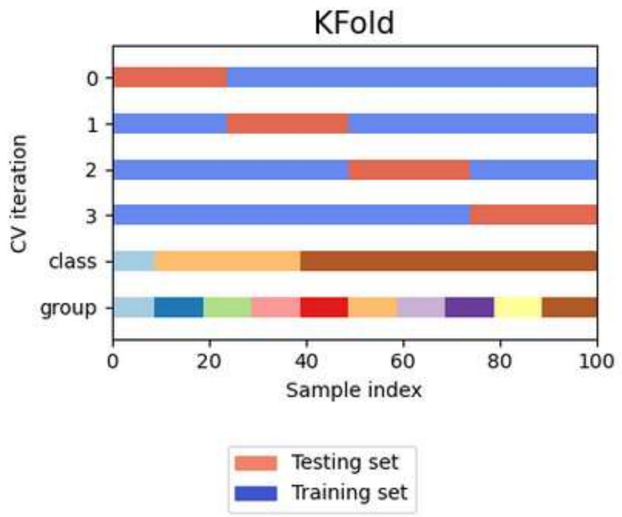
Main cross-validation advantages

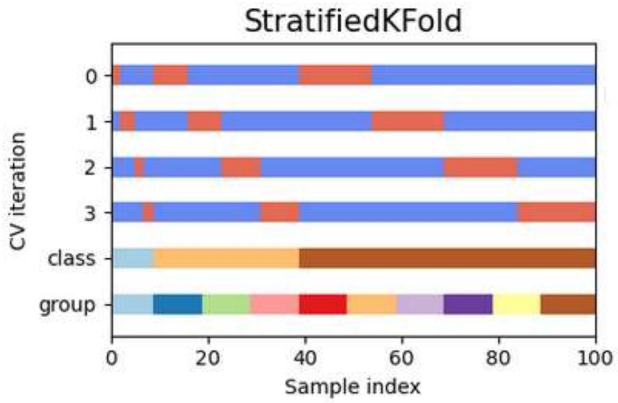
- Cross-validation has two main advantages over using one set for training and one for testing a model:

1. The distribution of classes is more even (**рівномірним**), which improves the quality of learning.

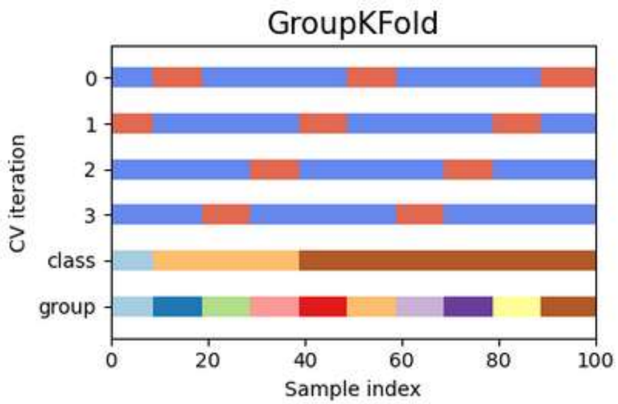
2. If, for each pass, the output error of the model is estimated and averaged over all passes, then the resulting estimate will be more reliable (**достовірною**).

Some variations on cross-validation

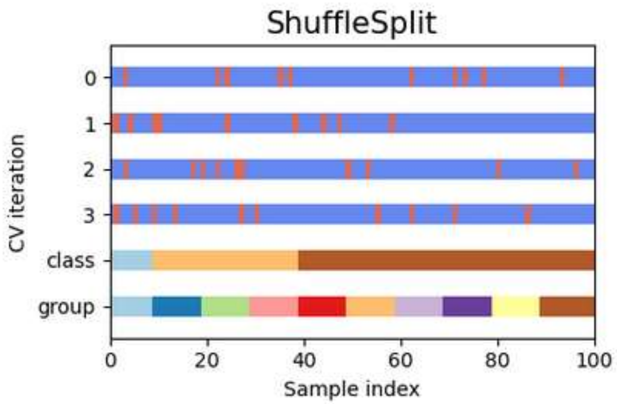


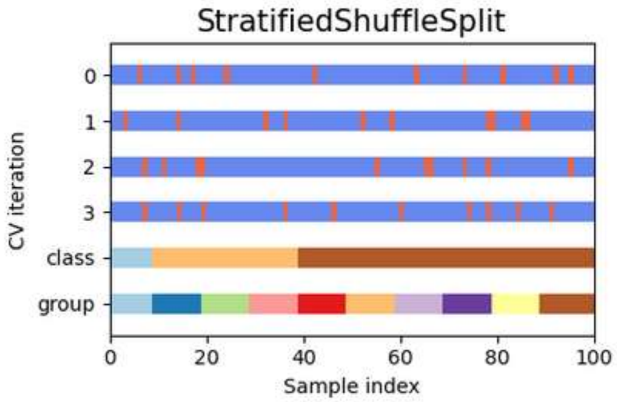


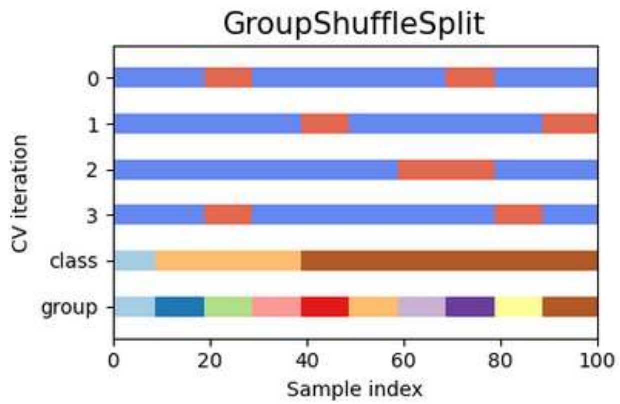
Each set contains approximately the same percentage of samples of each target class as the complete set



The same group is not represented in both testing and training sets (e.g. people from the same family)







A random sample (with replacement) of the train/test splits, when the number of groups is large enough that generating all possible partitions with groups withheld would be prohibitively expensive

Practical tips for using k-fold cross-validation

❑ Question: How many folds do we need?

Answer: With larger K, ...

- Error estimation tends to be more accurate
- But computation time will be greater
- In practice: usually use K = 3, 5 or 10, but larger dataset => choose smaller K

The most popular value used in applied machine learning to evaluate models is k=10. The reason for this is studies were performed and k=10 was found to provide good trade-off of low computational cost and low bias in an estimate of model performance.