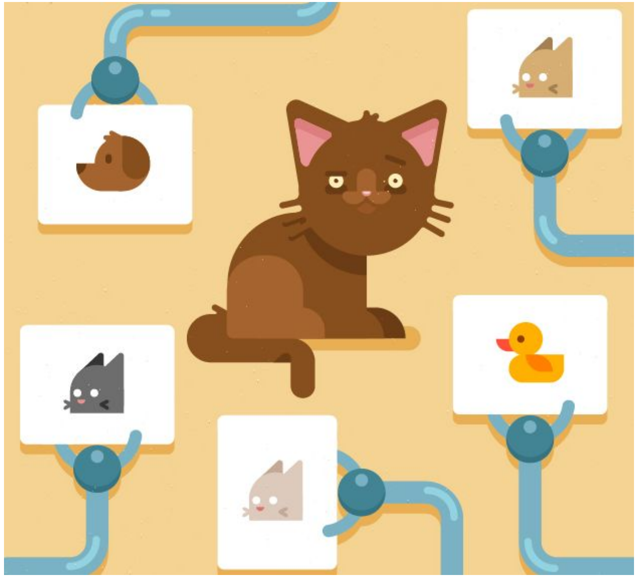
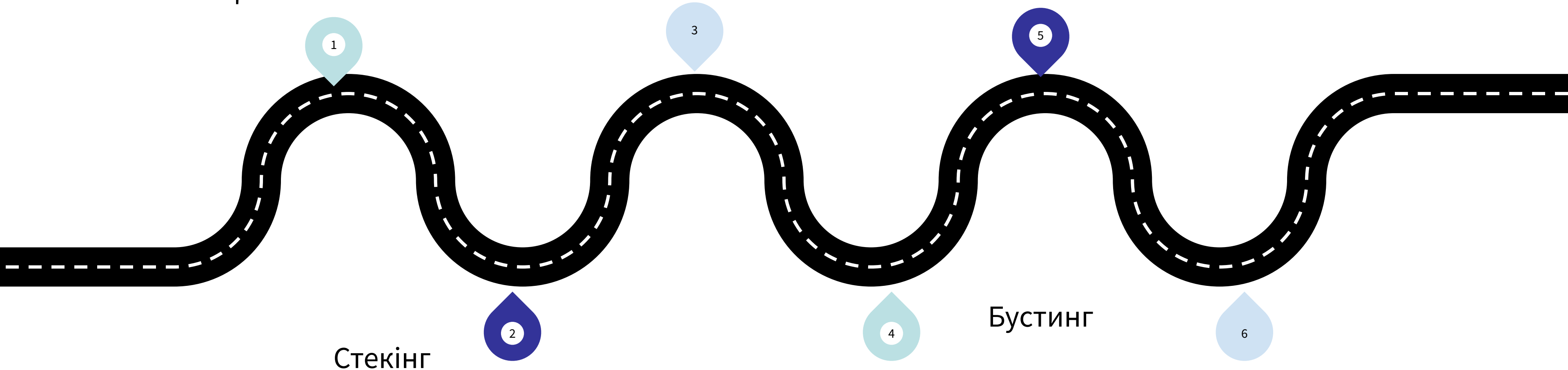
Розпізнавання образів

Ансамблеві методи



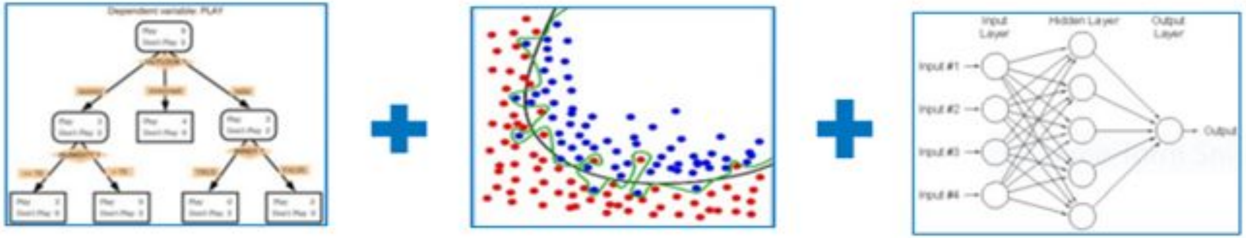
Сьогодні на лекції



Ансамблеві методи

- Послідовні ансамблі
- Паралельні ансамблі

- Однорідні ансамблі
- Різнорідні ансамблі

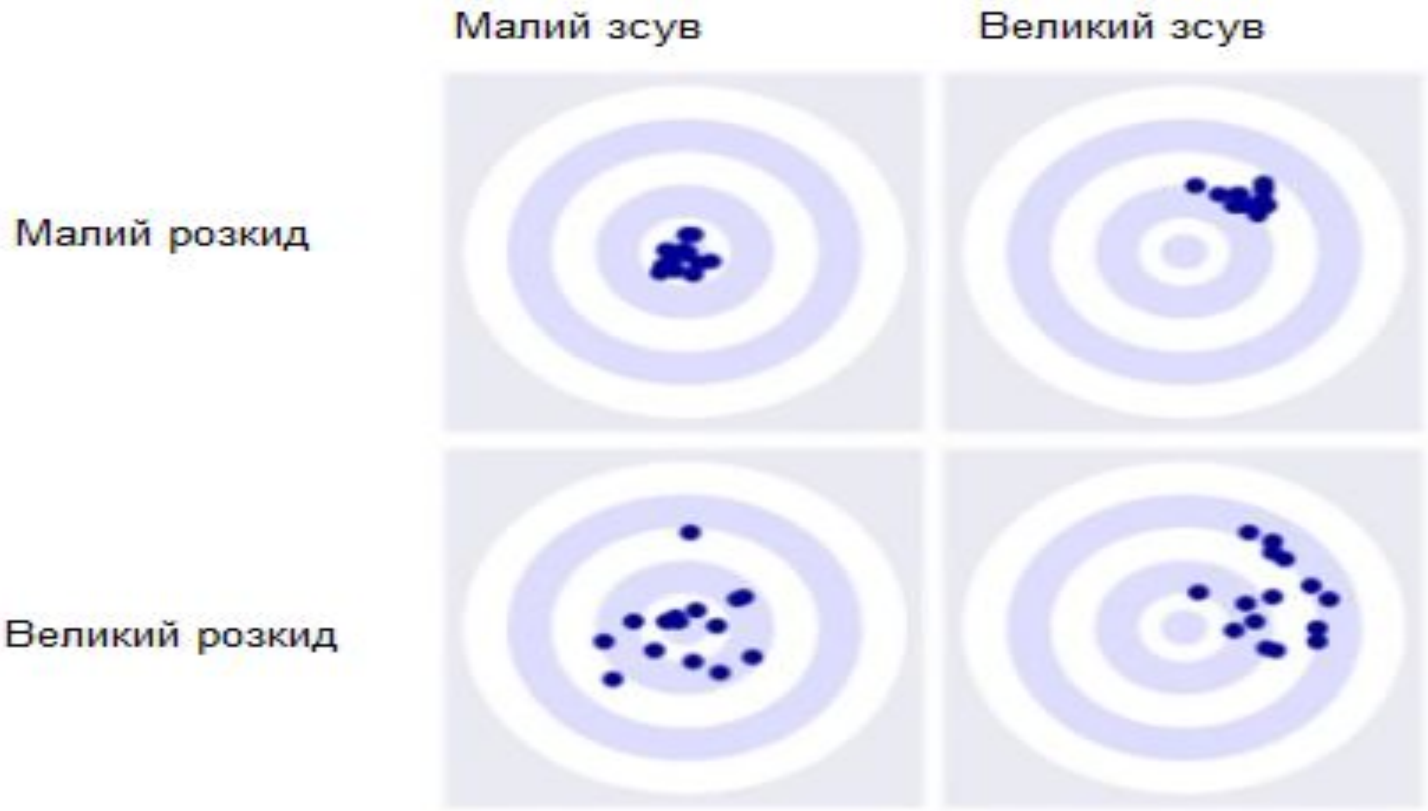


Unweighted average:

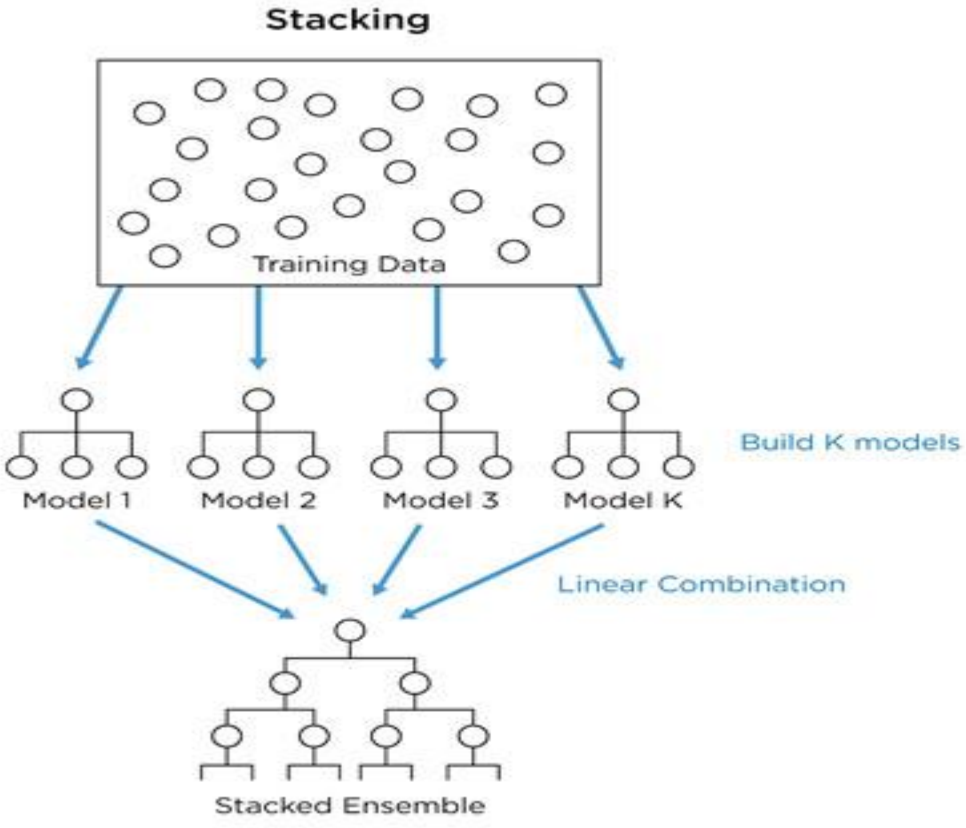
Weighted average:

VotingRegressor

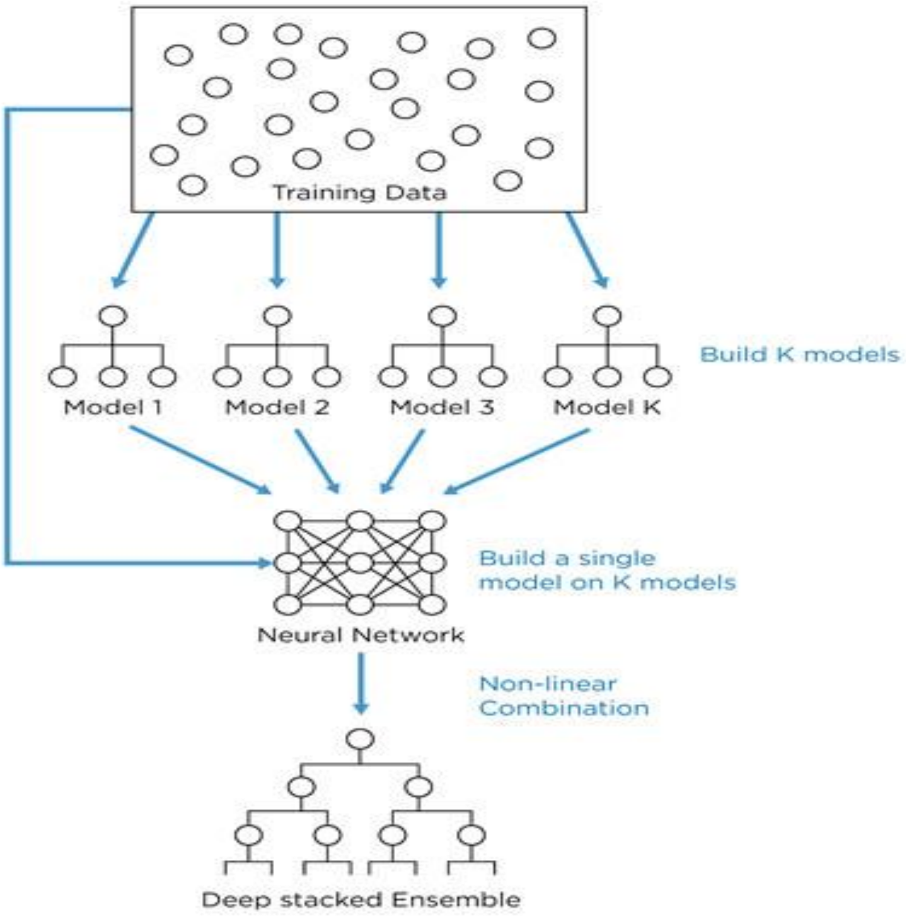
VotingClassifier


Малий зсув
Великий зсув
Малий розкид
Великий розкид

Як комбінувати базові моделі? Стекінг



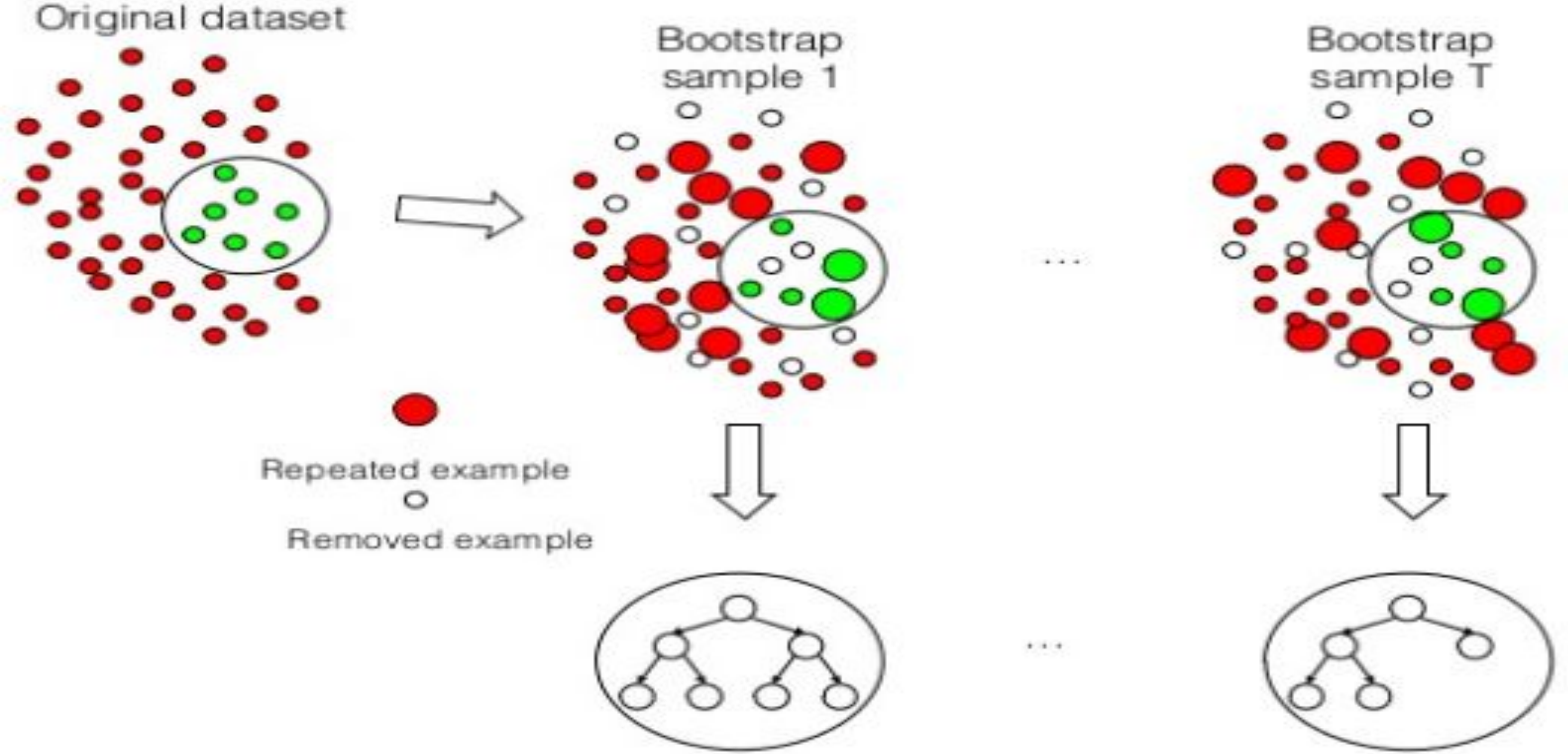
Deep Stacking

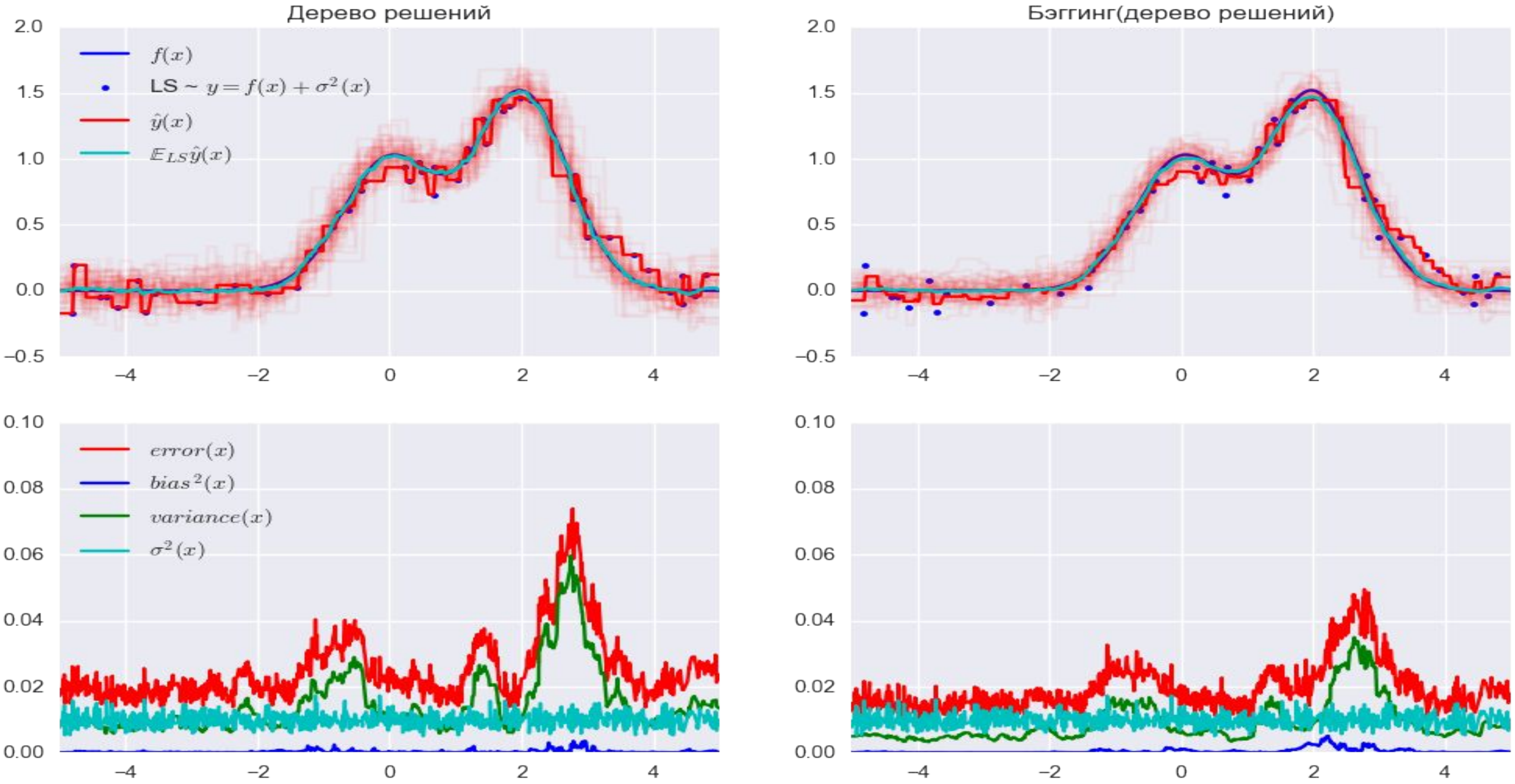


•Jeong-Yoon Lee, Winning Data Science Competitions

Як комбінувати базові моделі? Бегінг

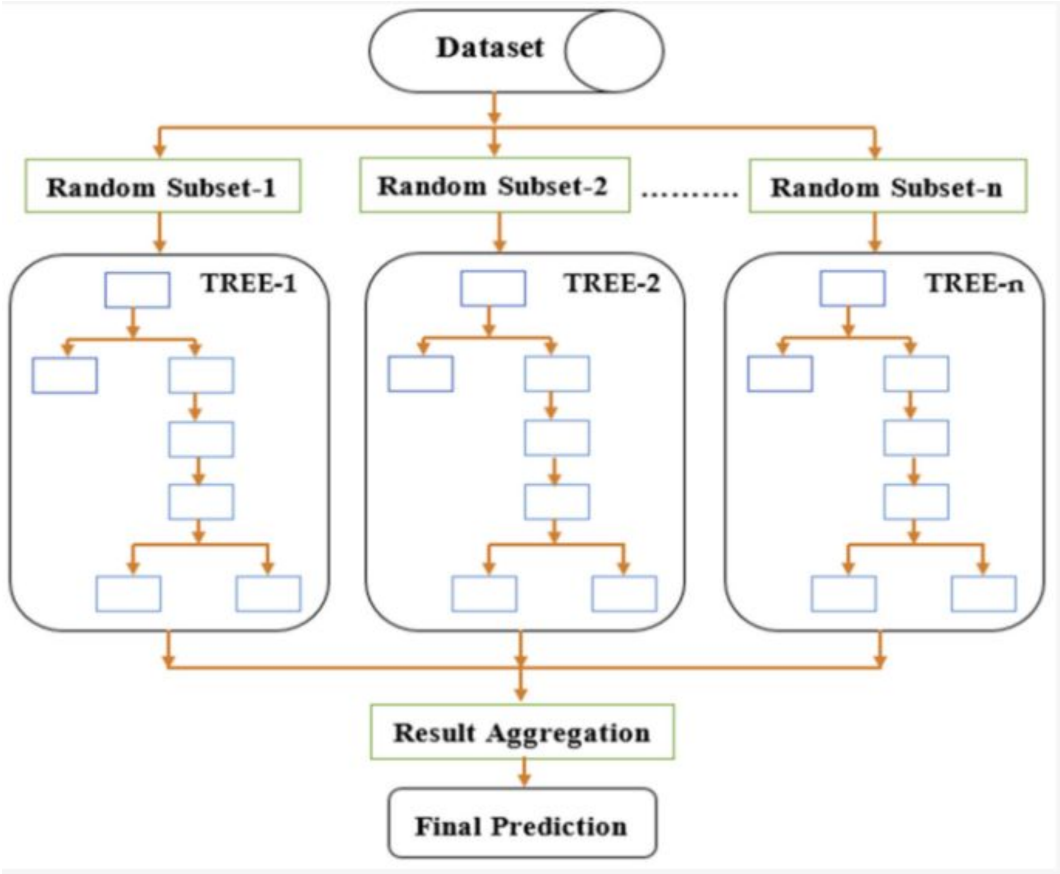
Bootstrap aggregation



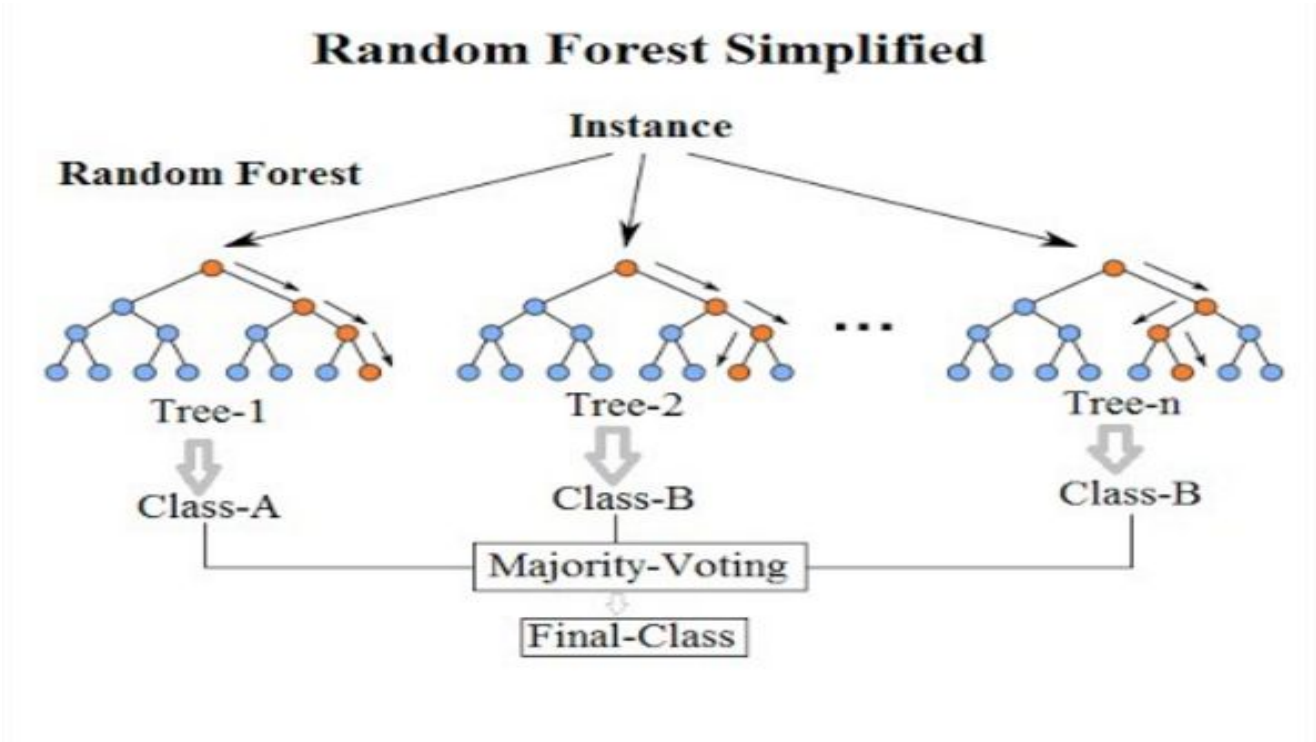

Дерево решений
Бэггинг(дерево решений)
$f(x)$
LS ~ $y = f(x) + \sigma^2(x)$
$\hat{y}(x)$
$\mathbb{E}_{LS}\hat{y}(x)$
$error(x)$
$bias^2(x)$
$variance(x)$
$\sigma^2(x)$
2.0
1.5
1.0
0.5
0.0
−0.5
0.10
0.08
0.06
0.04
0.02
0.00
−4
−2
0
2
4

Out-of-Bag

Бегінг над деревами



Випадковий ліс



When poll is active, respond at **pollev.com/nataliashovgun288**
Text **NATALIASHOVGUN288** to **37607** once to join

Чим відрізняється бегінг над деревами та випадковий ліс?

Способом генерування підвиборок

сортом дерев

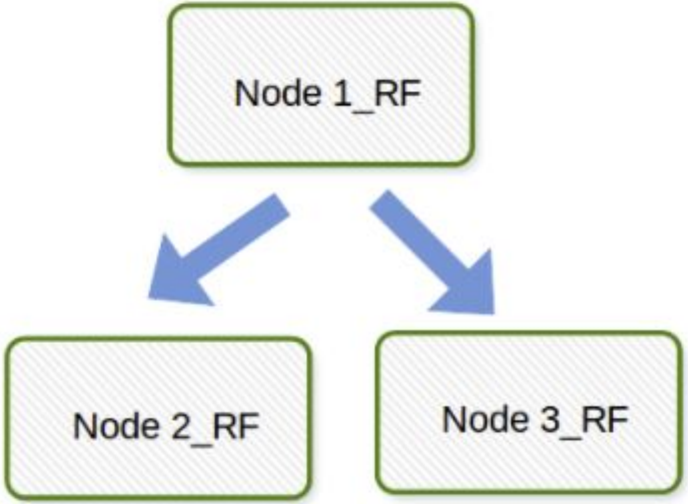
кількістю ознак що використовуються моделлю

способом навчання дерев

Powered by Poll Everywhere
Start the presentation to see live content. For screen share software, share the entire screen. Get help at **pollev.com/app**

Random forests--

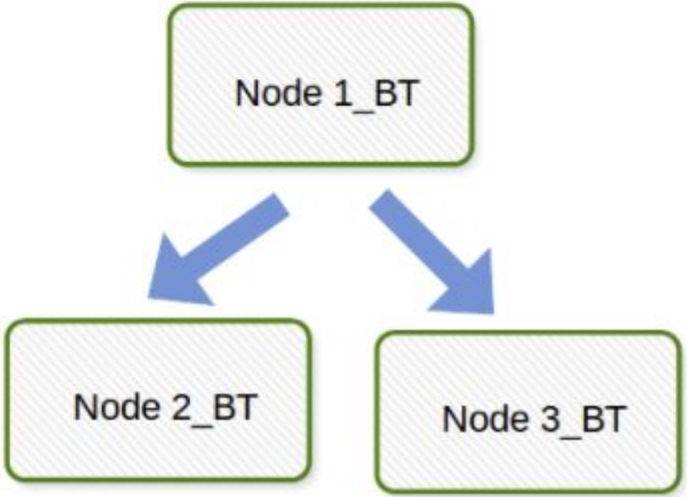
Only m<M features considered
for each node for split



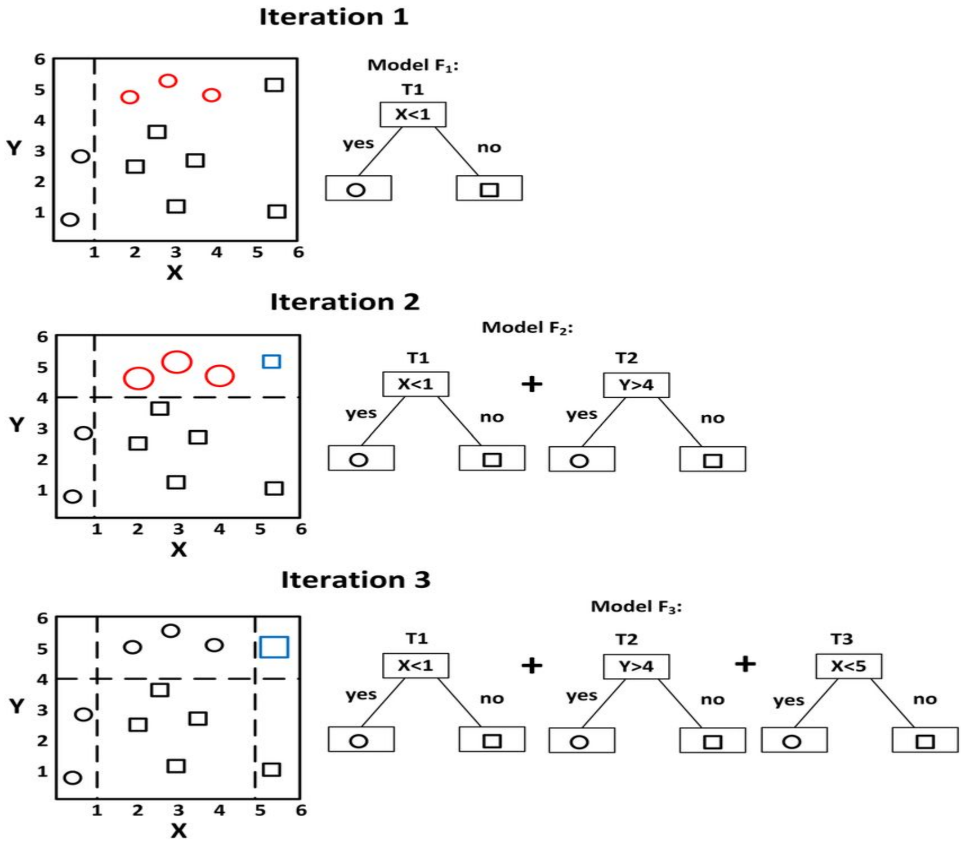
m can be selected via out-of-bag error,
but *m* = sqrt(*M*) is a good value to start with

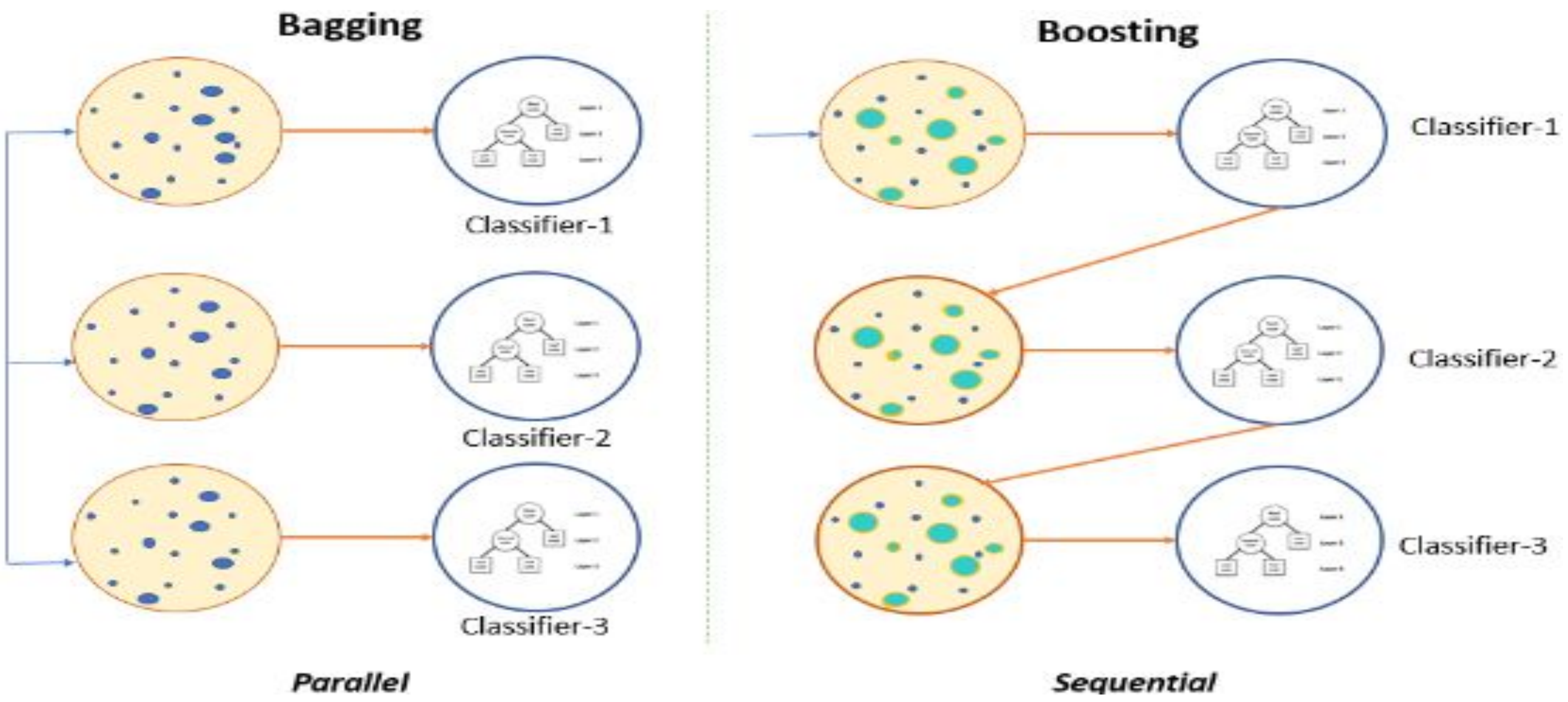
Bagging Trees--

All of M features considered
for each node for a split



Як комбінувати базові моделі? Бустинг




Bagging
Boosting
Classifier-1
Classifier-2
Classifier-3
Classifier-1
Classifier-2
Classifier-3
Parallel
Sequential

AdaBoost (*adaptive boosting*)

Навчальна вибірка: $(x_1, y_1), \ldots, (x_m, y_m)$ где $x_i \in X,\ y_i \in Y = \{-1, +1\}$

Призначаємо кожному екземпляру вибірки певну вагу. Отже на кожній ітерацію матимемо набір ваг Dt(i). На початку встановлюємо: $D_1(i) = \frac{1}{m}, i = 1, \ldots, m.$

Далі на кожній ітерації t:

1. Знаходимо классификатор $h_t : X \to \{-1, +1\}$ який мінімізує зважену помилку класифікації:

$$h_t = \arg\min_{h_j \in \mathcal{H}} \epsilon_j, \text{ где } \epsilon_j = \sum_{i=1}^{m} D_t(i)[y_i \neq h_j(x_i)]$$

де e_t - зважена помилка класифікації.

2. Якщо $e_t \geq 0.5$ то зупиняємось.

Алгоритм **AdaBoost**

3. Обираємо $\alpha_t = \frac{1}{2}\ln\frac{1-\epsilon_t}{\epsilon_t}$ важливість класифікатора

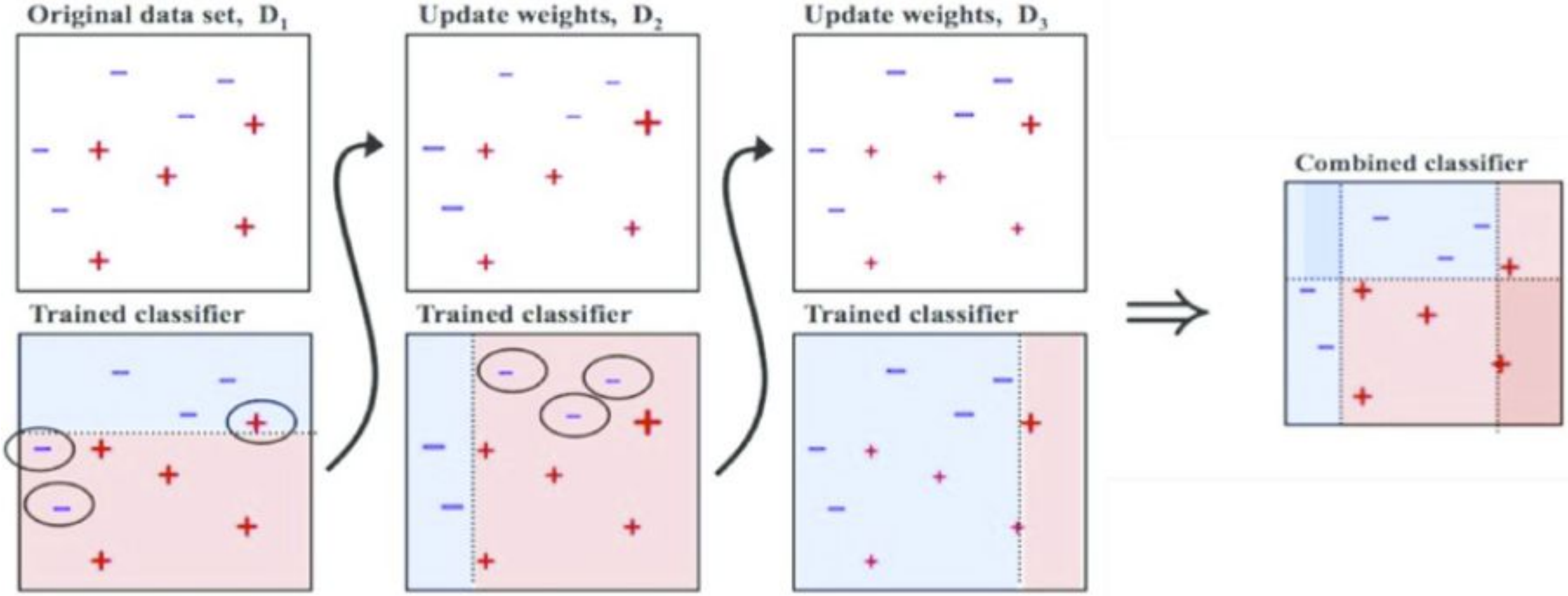
4. Оновлюємо ваги:

$$\begin{aligned} D_{t+1}(i) &= \frac{D_t(i)}{Z_t} \times \begin{cases} e^{-\alpha_t} & \text{if } h_t(x_i) = y_i \\ e^{\alpha_t} & \text{if } h_t(x_i) \neq y_i \end{cases} \\ &= \frac{D_t(i)\exp(-\alpha_t y_i h_t(x_i))}{Z_t} \end{aligned}$$

де Z_t - нормалізуючий параметр такий що $\sum_{i=1}^{m} D_{t+1}(i) = 1$).

Результуючий класифікатор

$$H(x) = \text{sign}\left(\sum_{t=1}^{T} \alpha_t h_t(x)\right)$$


Original data set, D_1
Update weights, D_2
Update weights, D_3
Trained classifier
Trained classifier
Trained classifier
Combined classifier

Градієнтний бустинг

1. Будуємо нашу модель на вибірці даних. $F_0(x) = \arg\min_{\gamma} \sum_{i=1}^{n} L(y_i, \gamma).$

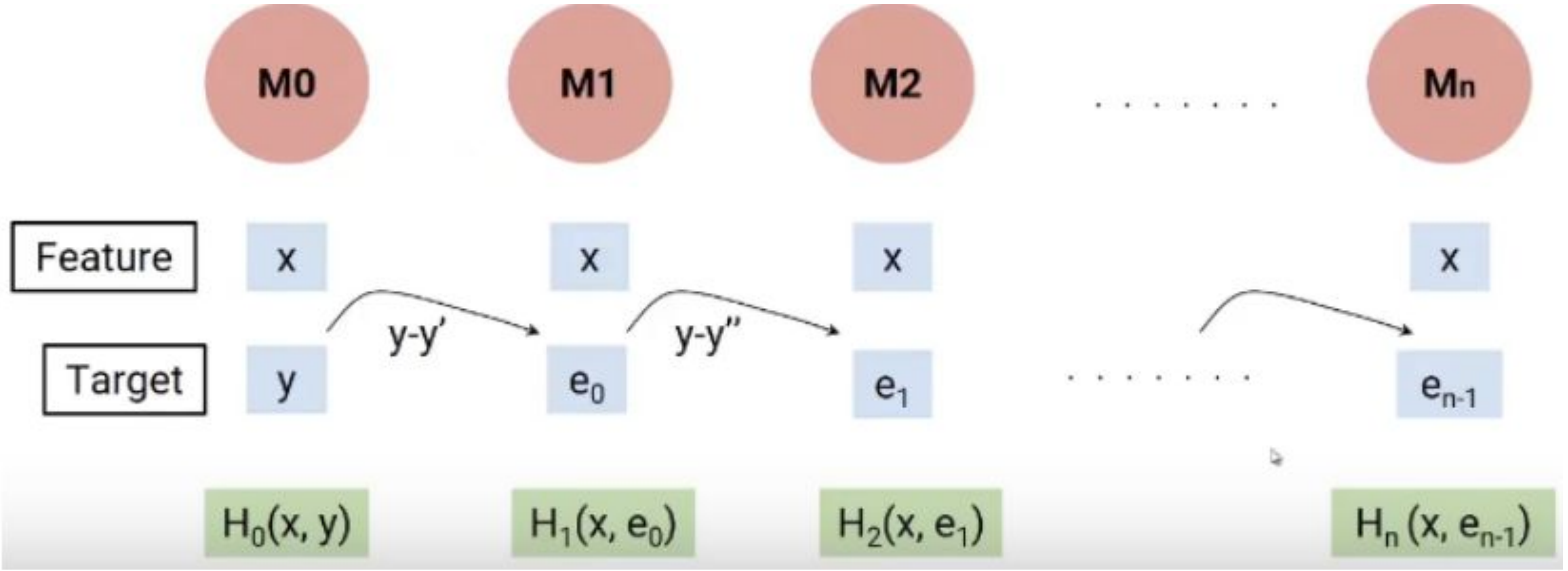
   Далі на кожній ітерації:

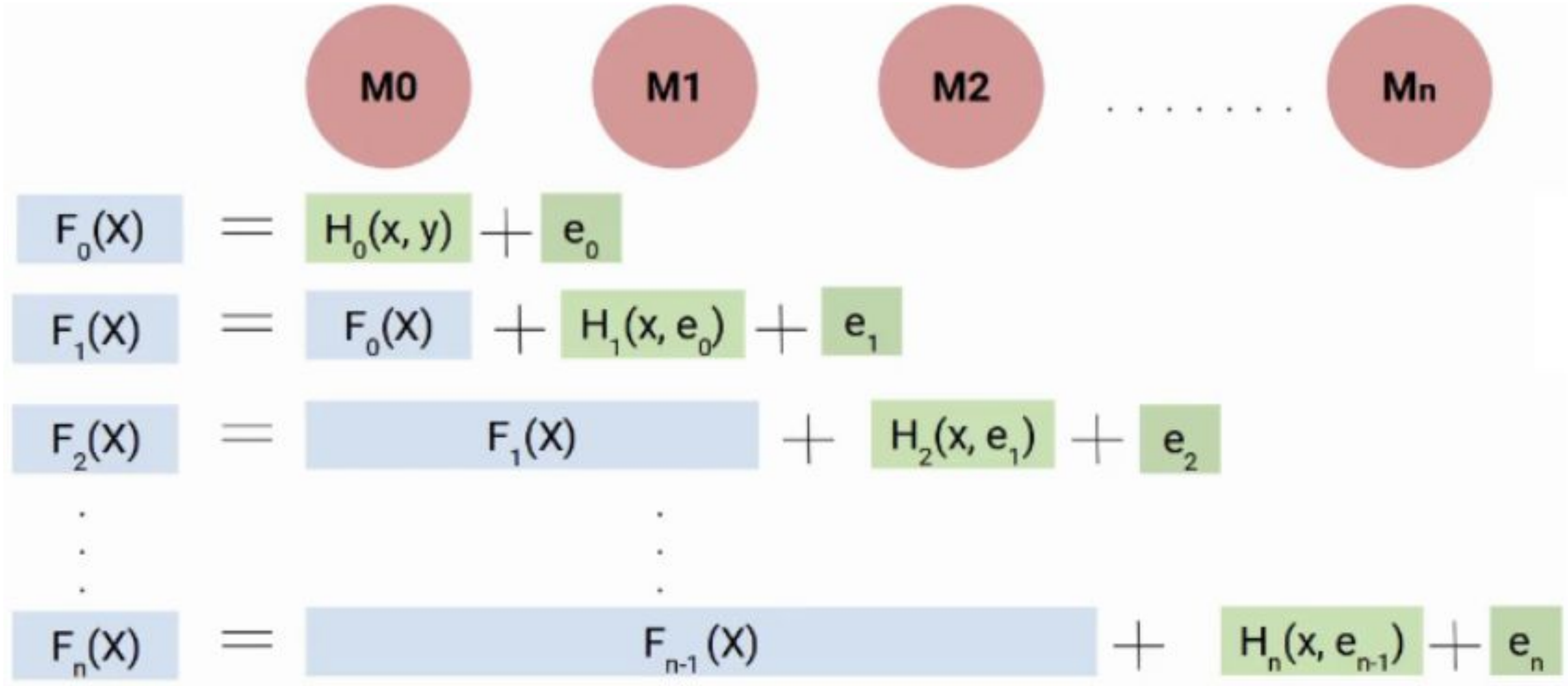
2. Вираховуємо псевдо залишки $r_{im} = -\left[\frac{\partial L(y_i, F(x_i))}{\partial F(x_i)}\right]_{F(x)=F_{m-1}(x)} \quad \text{for } i = 1, \ldots, n.$
3. Будуємо нову слабку модель $h_m(x)$, в якості цільової змінної беремо залишки $\{(x_i, r_{im})\}_{i=1}^{n}$
4. Вираховуємо параметри моделі на поточному кроці

$$\gamma_m = \arg\min_{\gamma} \sum_{i=1}^{n} L\left(y_i, F_{m-1}(x_i) + \gamma h_m(x_i)\right)$$

5. Оновлюємо модель на поточному кроці

$$F_m(x) = F_{m-1}(x) + \gamma_m h_m(x)$$


M0
M1
M2
Mn
Feature
x
x
x
x
Target
y
y-y'
e0
y-y''
e1
en-1
H0(x, y)
H1(x, e0)
H2(x, e1)
Hn (x, en-1)


M0
M1
M2
Mn
$F_0(X) = H_0(x, y) + e_0$
$F_1(X) = F_0(X) + H_1(x, e_0) + e_1$
$F_2(X) = F_1(X) + H_2(x, e_1) + e_2$
$F_n(X) = F_{n-1}(X) + H_n(x, e_{n-1}) + e_n$

12 10 2 1 11

$$F_{n+1}(X) = F_n(X) + \gamma_n \, H(x, e_n)$$

$$F_0(X) = \gamma_0 \, H_0(x, y) + e_0$$

$$F_1(X) = F_0(X) + \gamma_1 \, H_1(x, e_0) + e_1$$

$$F_2(X) = F_1(X) + \gamma_2 \, H_2(x, e_1) + e_2$$

$$\vdots \qquad\qquad \vdots$$

$$F_n(X) = F_{n-1}(X) + \gamma_n \, H_n(x, e_{n-1}) + e_n$$

Вигляд функції втрат

Для регресії вирішуємо, яку саме властивість умовного розподілу ми хочемо відновити. Найбільш часті варіанти:

- $L(y, f) = (y - f)^2$ L2 loss.
- $L(y, f) = |y - f|$ L1 loss.

Для задач класифікації:

- L (y, f) = log (1 + exp (-2yf)), Logistic loss.

  L (y, f) = exp (-yf), Adaboost loss.

When poll is active, respond at **pollev.com/nataliashovgun288**
Text **NATALIASHOVGUN288** to **37607** once to join

Яке твердження справедливе для градієнтного бустингу?

True

False

Powered by Poll Everywhere
Start the presentation to see live content. For screen share software, share the entire screen. Get help at **pollev.com/app**

Градієнтний бустинг над деревами

Припустимо, що ми хочемо навчити ансамбль з К дерев:

$$\hat{y}_i = \sum_{k=1}^{K} f_k(x_i), \text{ где } f_k \in \mathcal{F}.$$

Цільова функція – це втрати + регуляризатори:

$$\mathbf{Obj} = \sum_{i=1}^{N} l(y_i, \hat{y}_i) + \sum_{k=1}^{K} \Omega(f_k).$$

Ми не можемо просто мінімізувати загальну помилку – важко мінімізувати за всіма можливими деревами. Шукаємо ітеративне наближення:

$$\begin{aligned}
&\hat{y}_i^{(0)} = 0, \\
&\hat{y}_i^{(1)} = f_1(x_i) = \hat{y}_i^{(0)} + f_1(x_i), \\
&\hat{y}_i^{(2)} = f_1(x_i) + f_2(x_i) = \hat{y}_i^{(1)} + f_1(x_2), \\
&\quad \dots,
\end{aligned}$$

Попередні дерева завжди залишаються тими ж самими - фіксовані.

Функція оптимізації для градієнтного бустинга:

$$\mathcal{L}^{(t)} = \sum_{i=1}^{n} l(y_i, \hat{y_i}^{(t-1)} + f_t(x_i)) + \Omega(f_t)$$

y_i,$y_i^{t\wedge}$ — значення *i*-го елемента навчальної вибірки і сума прогнозів перших *t* дерев відповідно.

x_i — набір ознак *i*-го елемента навчальної вибірки.

f_t — функція (дерево), яку ми хочемо навчити на кроці *t*.

$f_t(x_i)$ — прогноз на *i*-ому елементі навчальної вибірки.

Ω(f)— регуляризація функції f.

$\Omega(f)=\gamma T+1/2\lambda\|w\|^2$, де T — кількість вершин в дереві, w — значення в листях, а γ і λ — параметри регуляризації.

Можемо наблизити шукану функцію L(t) наступним виразом:

$$\mathcal{L}^{(t)} = \sum_{i=1}^{n} l(y_i, \hat{y_i}^{(t-1)}) + g_i f_t(x_i) + 0.5 h_i f_t^2(x_i)) + \Omega(f_t)$$, где

$$g_i = \frac{\partial l(y_i, \hat{y}_i^{t-1})}{\partial \hat{y}_i^{t-1}}, \quad h_i = \frac{\partial^2 l(y_i, \hat{y}_i^{t-1})}{\partial^2 \hat{y}_i^{t-1}}$$

Оскільки ми хочемо мінімізувати помилку моделі на навчальній вибірці, нам треба знайти мінімум L(t) для кожного кроку *t*.

Кожне окреме дерево ансамбля $f_t(x_i)$ навчається стандартним алгоритмом.


Cache awareness and
out-of-core computing
Regularization for
avoiding overfitting
Tree pruning
using depth-first
approach
Efficient
heandling of
missing data
XGBoost
Parallelized
tree building
In-built cross-
validation
capability

XGBoost

1. Будуємо нашу модель на вибірці даних. $\hat{f}_{(0)}(x) = \arg\min_{\theta} \sum_{i=1}^{N} L(y_i, \theta)$

   Далі на кожній ітерації:

2. Вираховуємо $\hat{g}_m(x_i) = \left[\frac{\partial L(y_i, f(x_i))}{\partial f(x_i)}\right]_{f(x)=\hat{f}_{(m-1)}(x)}$ $\hat{h}_m(x_i) = \left[\frac{\partial^2 L(y_i, f(x_i))}{\partial f(x_i)^2}\right]_{f(x)=\hat{f}_{(m-1)}(x)}$
3. Будуємо нову слабку модель $h_m(x)$, в якості цільової змінної беремо

$$\{x_i, -\frac{\hat{g}_m(x_i)}{\hat{h}_m(x_i)}\}_{i=1}^{N}$$

4. Вираховуємо параметри моделі на поточному кроці

$$\hat{\phi}_m = \arg\min_{\phi \in \Phi} \sum_{i=1}^{N} \frac{1}{2} \hat{h}_m(x_i) \left[-\frac{\hat{g}_m(x_i)}{\hat{h}_m(x_i)} - \phi(x_i)\right]^2$$

$$\hat{f}_m(x) = \alpha \hat{\phi}_m(x).$$

5. Оновлюємо модель на поточному кроці

$$\hat{f}_{(m)}(x) = \hat{f}_{(m-1)}(x) + \hat{f}_m(x)$$

XGBoost

$$f_t(x) = w_{q(x)}, \qquad w \in \mathbb{R}^T, q: \mathbb{R}^d \to \{1,2,\ldots,Q\}$$

The leaf weight of the tree

The structure of the tree

$$\Omega(f_t) = \frac{1}{2}\lambda\left(\sum_{j=1}^{Q} w_j^2\right) + \alpha\left(\sum_{i=1}^{Q}|w_i|\right) + \gamma Q$$

Xgboost's lambda

L2 penalty

Xgboost's alpha

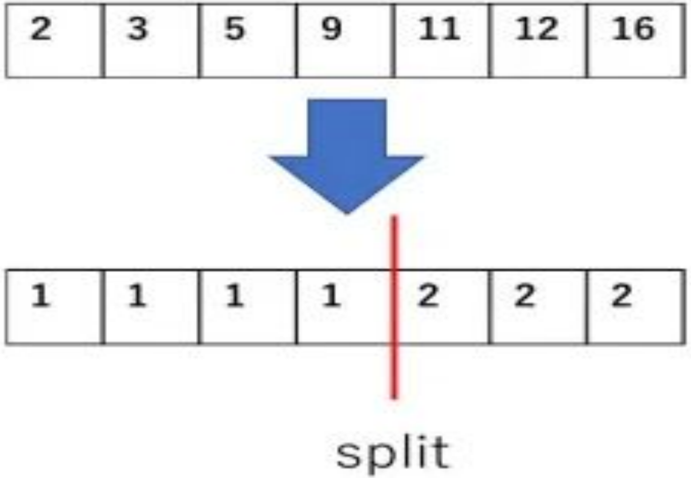
L1 penalty

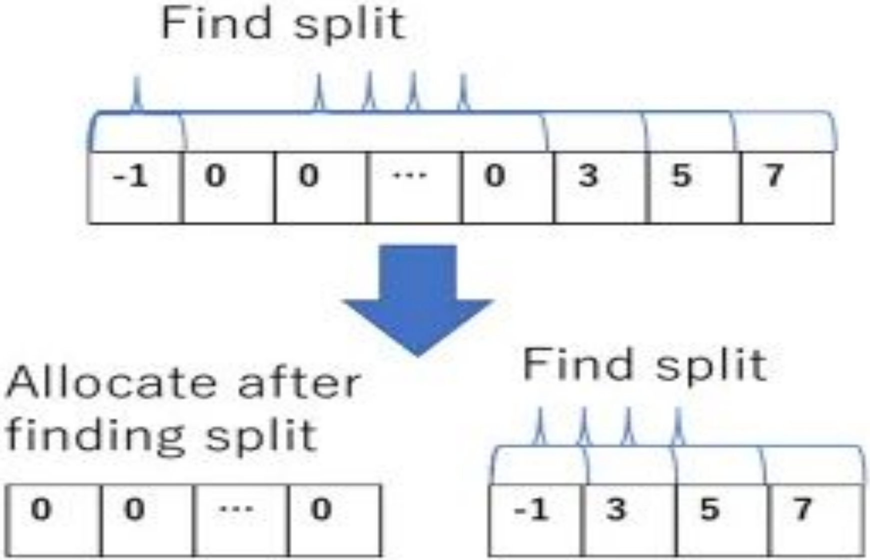
Xgboost's gamma

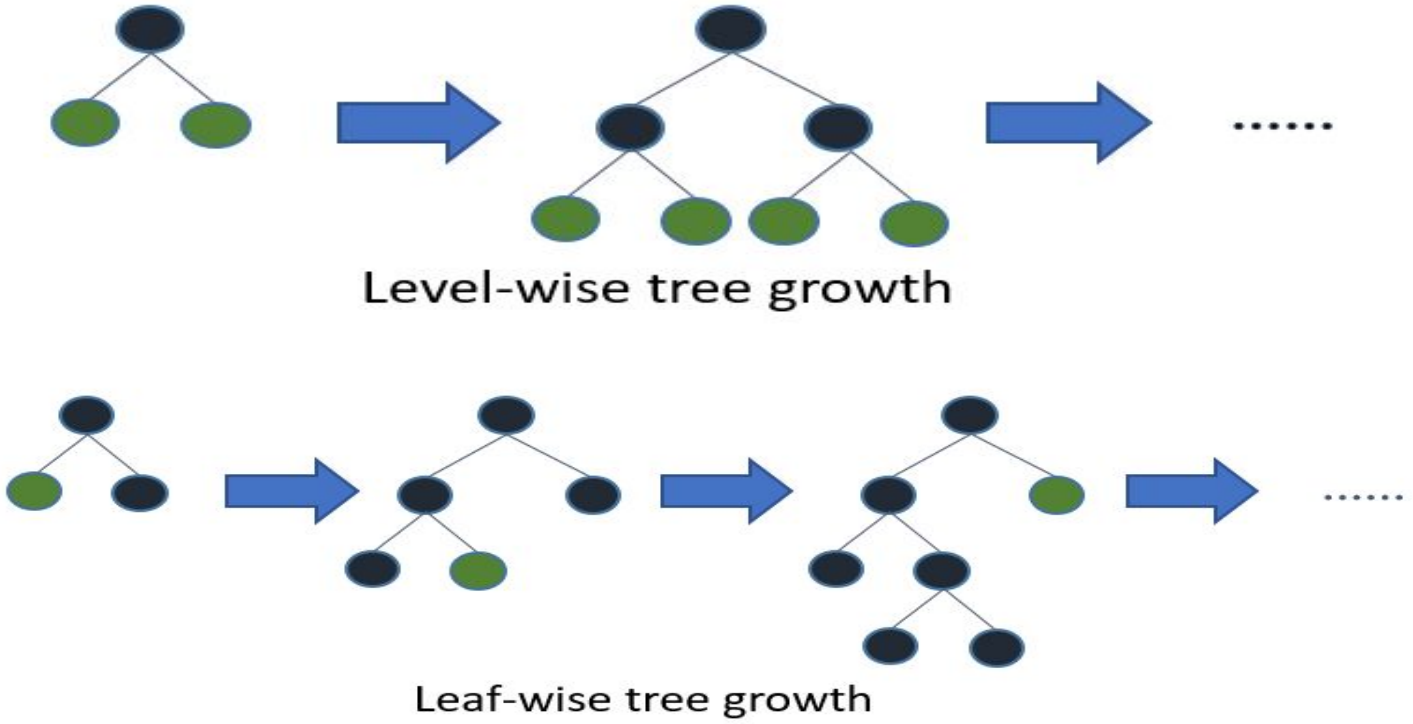
Number of leaves

Lightgbm

- **Швидша швидкість навчання та вища ефективність** : Lightgbm використовує алгоритм, заснований на гістограмі, тобто він об'єднує безперервні ознаки у дискретні блоки, що пришвидшує процедуру навчання.
- **Менше використання пам'яті:** замінює безперервні значення на дискретні блоки, що призводить до зменшення використання пам'яті.
- **Краща точність, ніж будь-який інший алгоритм бустингу:** Він створює набагато складніші дерева, дотримуючись підходу з розумним розбиттям листів, а не простому обмеженні глибини, що є головним фактором досягнення більш високої точності.
- **Підтримується паралельне навчання.**


2
3
5
9
11
12
16
1
1
1
1
2
2
2
split


Find split
-1
0
0
···
0
3
5
7
Allocate after
finding split
0
0
···
0
Find split
-1
3
5
7


......
Level-wise tree growth
......
Leaf-wise tree growth

The LightGBM algorithm

Input:

Training data: $D = \{(\chi 1, y1), (\chi 2, y2), \ldots, (\chi N, yN)\}$, $\chi i \in \chi$, $\chi \subseteq R$, $yi \in \{-1,+1\}$; loss function: $L(y, \theta(\chi))$;

Iterations:

M; Big gradient data sampling ratio: a; slight gradient data sampling ratio: b;

1: Combine features that are mutually exclusive (i.e., features never simultaneously accept nonzero values) of χi, $i = \{1, \ldots, N\}$ by the exclusive feature bundling (EFB) technique;

2: Set $\theta_0(\chi) = arg\, min_c \sum_i^N L(y_i, c)$;

3: For m = 1 to M do

4: Calculate gradient absolute values:

$$r_i = \left|\frac{\partial L(y_i, \theta(x_i))}{\partial \theta(x_i)}\right|_{\theta(x)=\theta_{m-1(x)}}, i = \{1, \ldots, N\}$$

5: Resample data set using gradient-based one-side sampling (GOSS) process:

topN = *a* × *len(D)*; *randN* = *b* × *len(D)*;

sorted = *GetSortedIndices(abs(r))*;

A = *sorted*[1 : *topN*]; *B* = *RandomPick(sorted*[*topN* : *len(D)*] , *randN)*;

$\acute{D} = A + B$;

6: Calculate information gains:

$$V_j(d) = \frac{1}{n}\left(\frac{\left(\sum_{x_i \in A_l} r_i + \frac{1-a}{b}\sum_{x_i \in B_l} r_i\right)^2}{n_l^j(d)} + \frac{\left(\sum_{x_i \in A_r} r_i + \frac{1-a}{b}\sum_{x_i \in B_r} r_i\right)^2}{n_r^j(d)}\right)$$

7: Develop a new decision tree $\theta_m(x)'$ on set D'

8: Update $\theta_m(\chi) = \theta_{m-1}(\chi) + \theta_m(\chi)$

9: End for

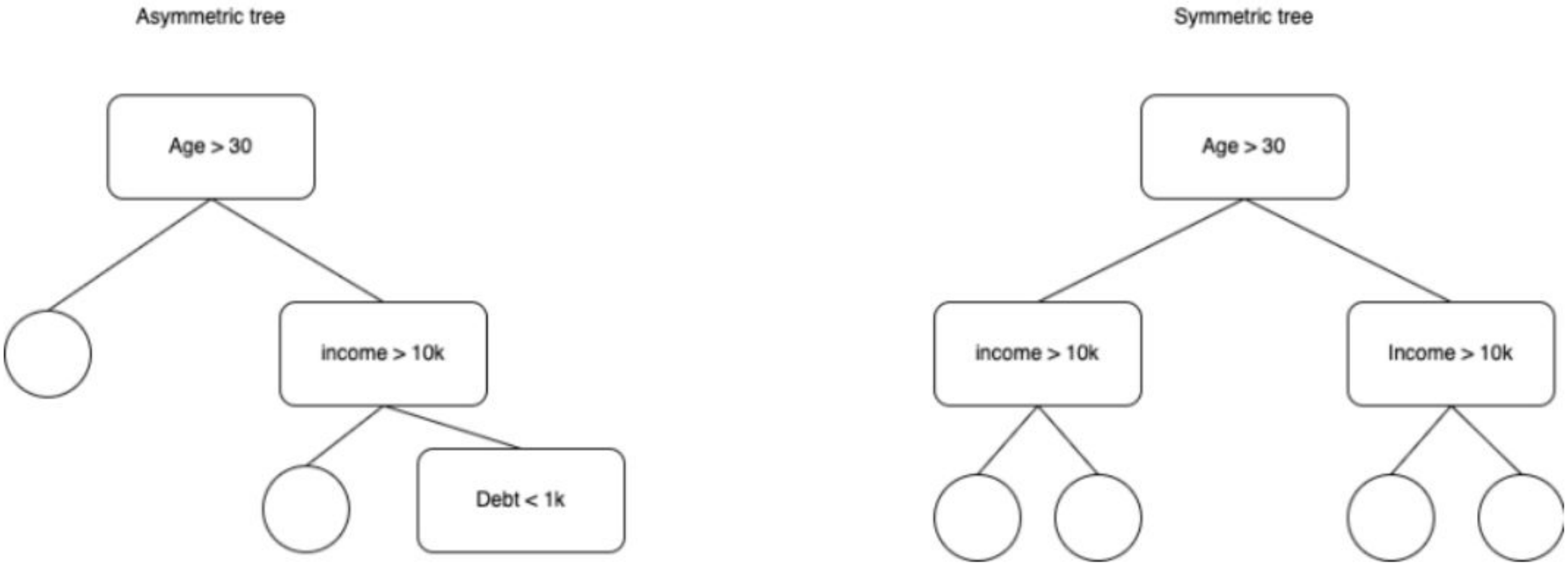
10: Return $\tilde{\theta}(x) = \theta_M(x)$

- Max depth
- Min_data_in_leaf
- Feature_fraction
- Bagging_fraction

Early_stopping_round

- Lambda
- Min_gain_to_split

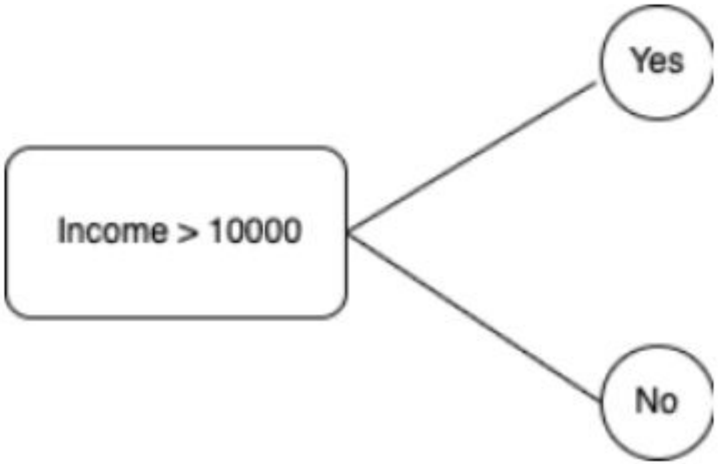
Catboost

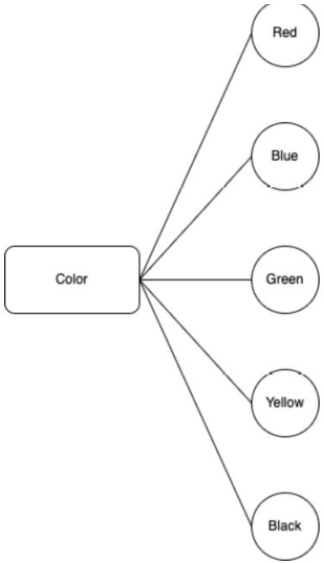

Asymmetric tree
Age > 30
income > 10k
Debt < 1k
Symmetric tree
Age > 30
income > 10k
Income > 10k

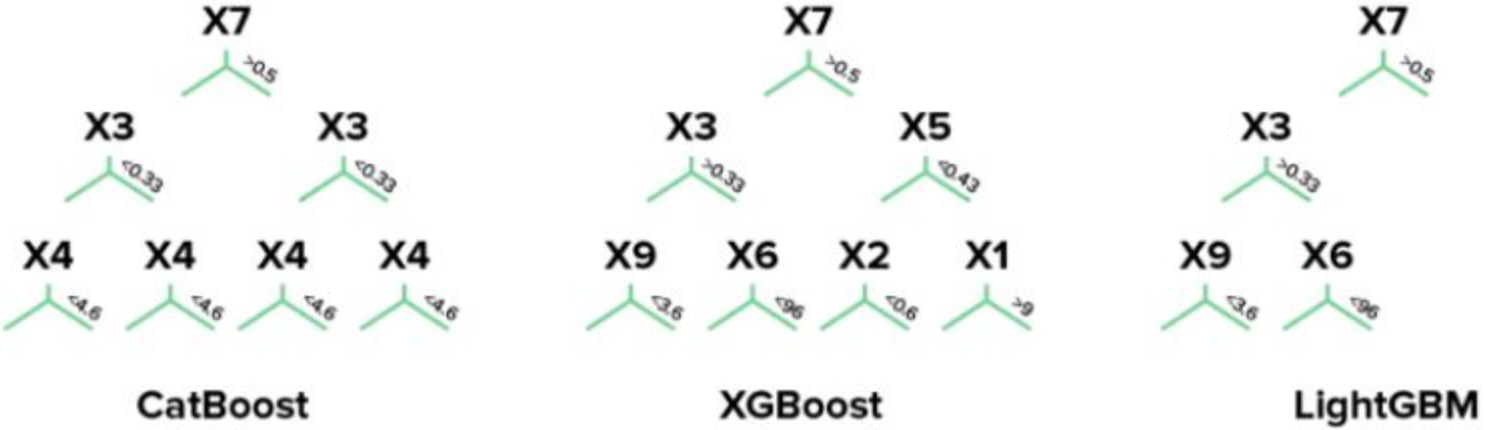
Catboost

Ordered boosting

Native feature support






X7
>0.5
X3
<0.33
X3
<0.33
X4
<4.6
X4
<4.6
X4
<4.6
X4
<4.6
CatBoost
X7
>0.5
X3
>0.33
X5
<0.43
X9
<3.6
X6
<96
X2
<0.6
X1
>9
XGBoost
X7
>0.5
X3
>0.33
X9
<3.6
X6
<96
LightGBM

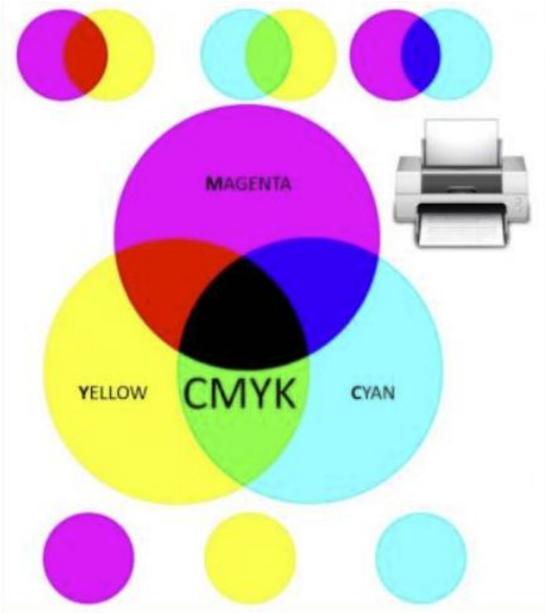
Розпізнавання образів

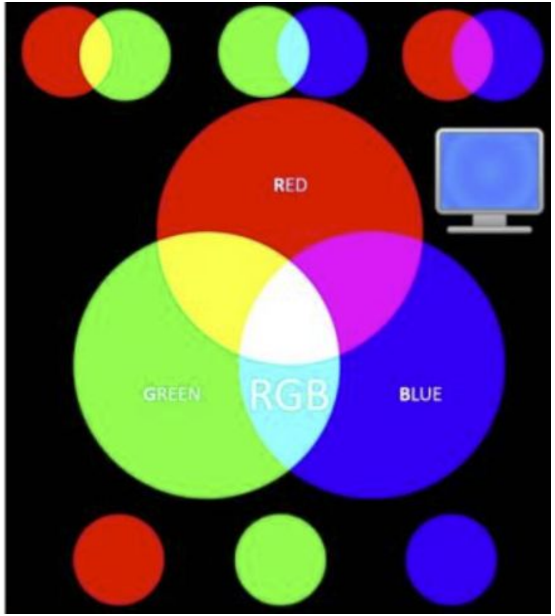
Колірні простори

Колірна модель

Колірна модель — абстрактна модель опису представлення кольорів у вигляді кортежів (наборів) чисел, зазвичай з трьох або чотирьох значень, званих колірними компонентами або колірними координатами.

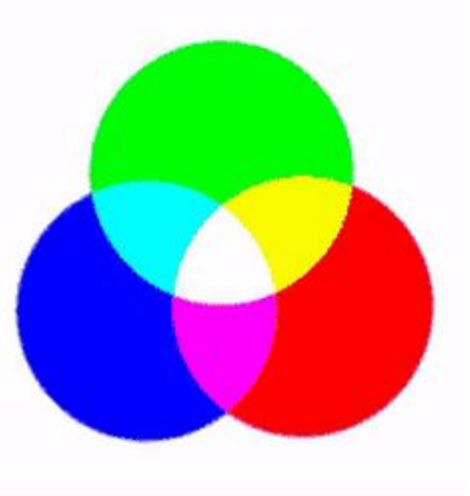
Разом з методом інтерпретації цих даних (наприклад, визначення умов відтворення та / або перегляду — тобто завдання способу реалізації), множина кольорів колірної моделі визначає **колірний простір**.

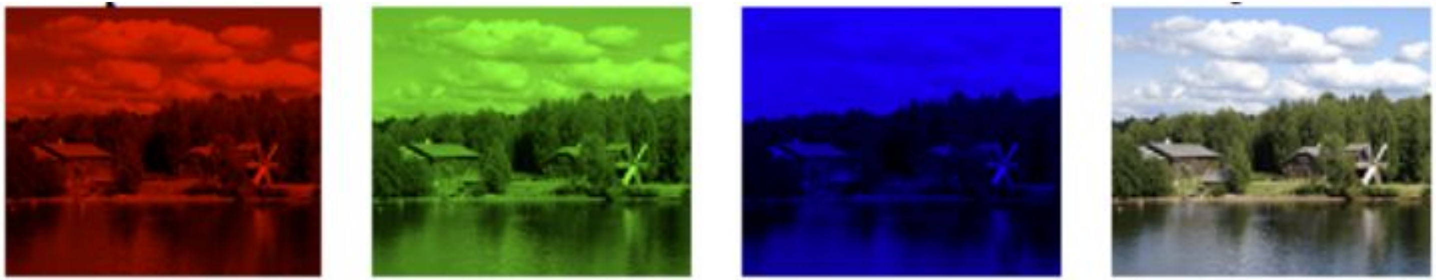




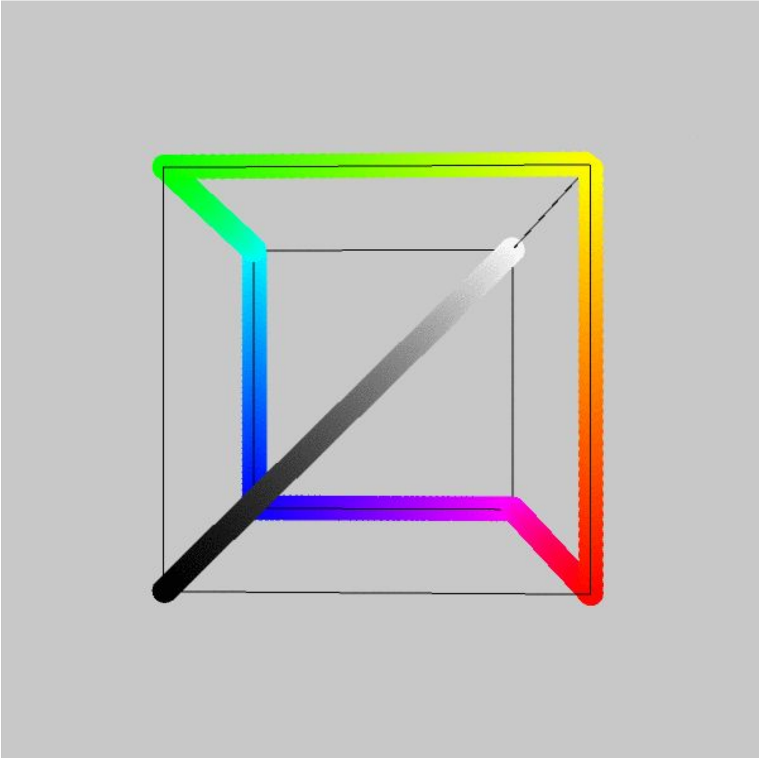
RGB

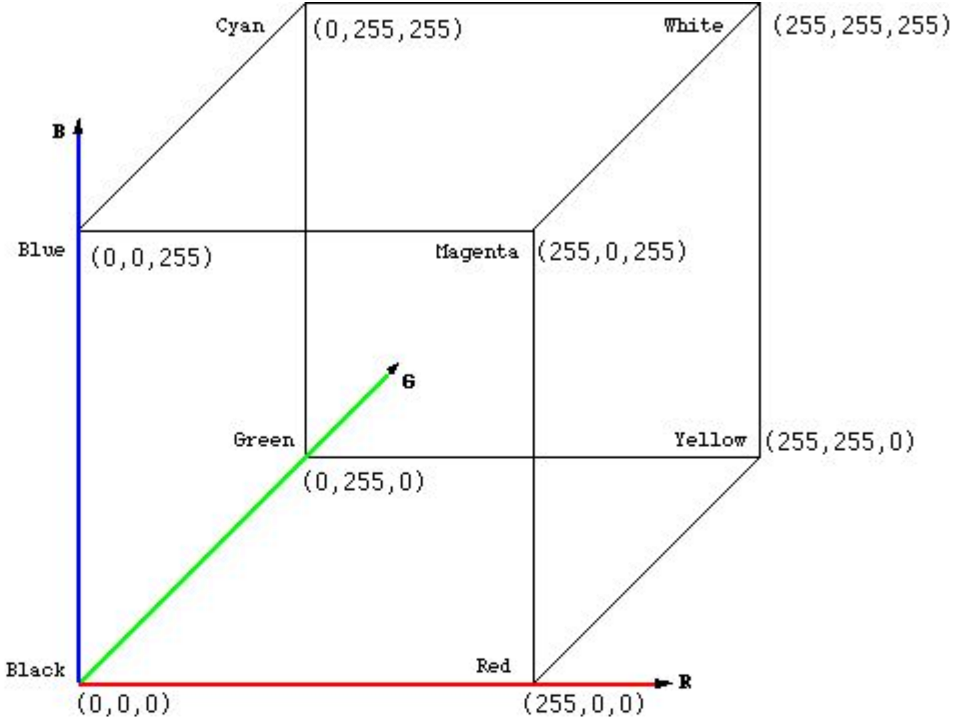
RGB-колір виходить в результаті змішування червоного, синього і зеленого в різних пропорціях: кожен відтінок можна



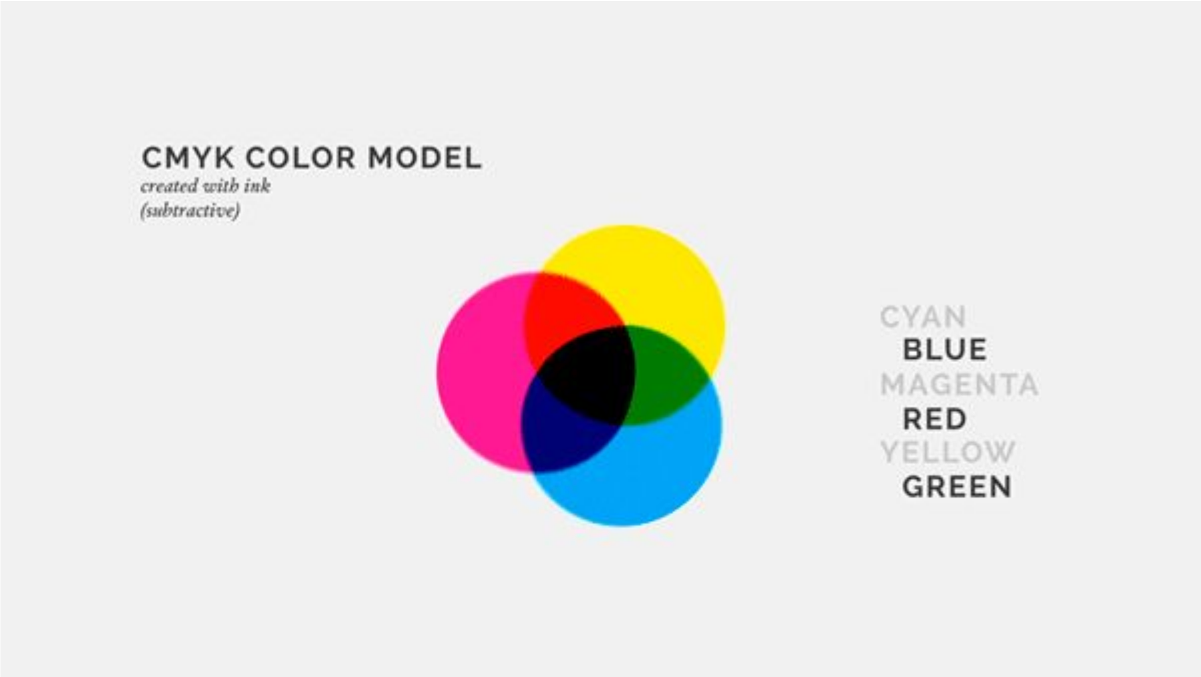


RGB

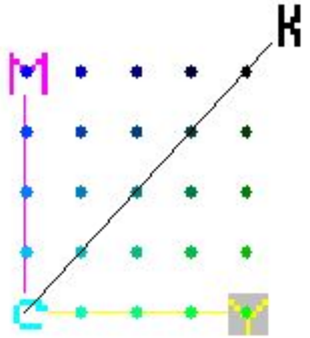



Cyan
(0,255,255)
White
(255,255,255)
B
Blue
(0,0,255)
Magenta
(255,0,255)
G
Green
(0,255,0)
Yellow
(255,255,0)
Black
(0,0,0)
Red
(255,0,0)
R

CMY

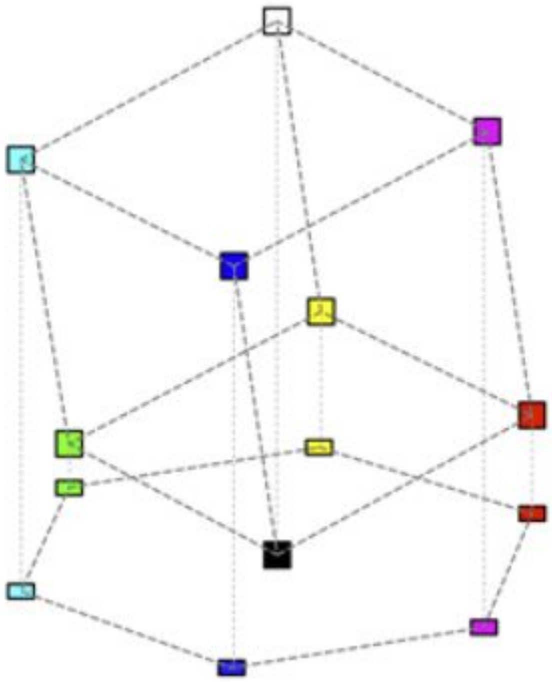


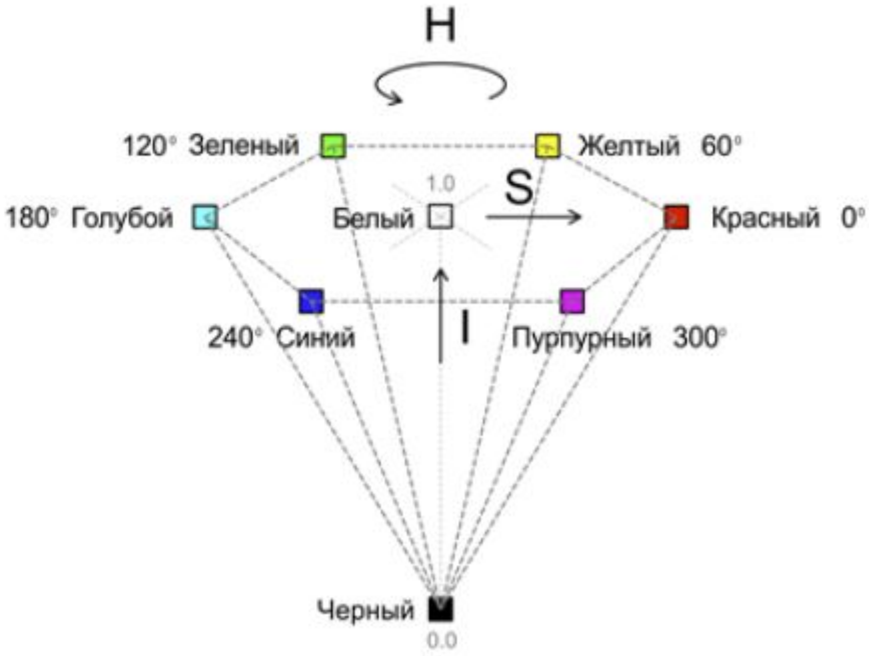
$$\begin{bmatrix} C \\ M \\ Y \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} - \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$

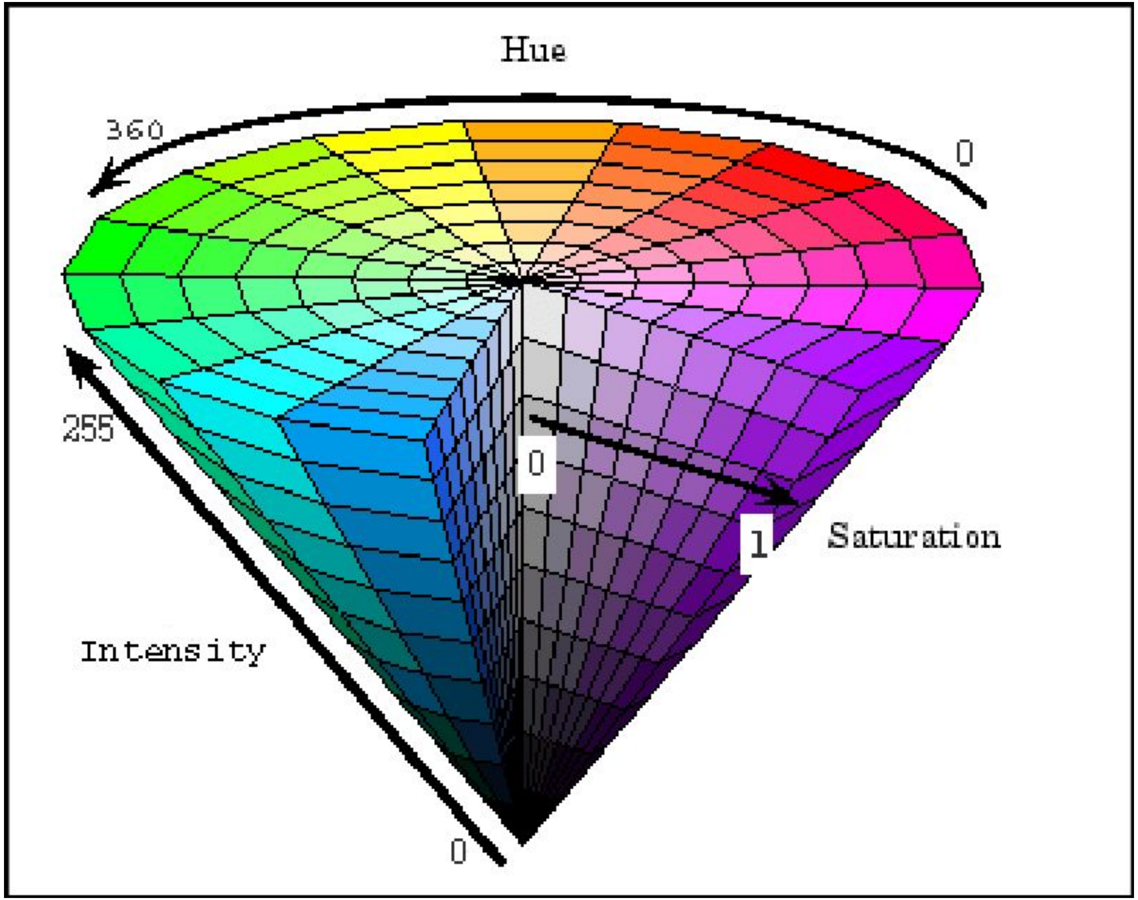


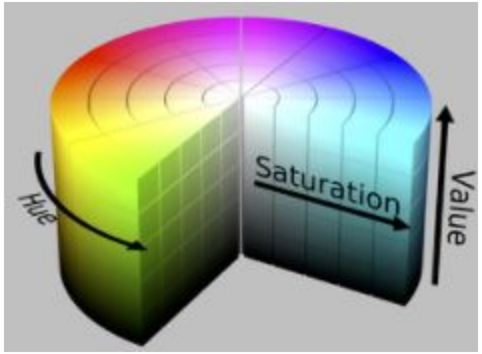
HSI

I - інтенсивність






Hue
360
0
255
Intensity
0
0
1
Saturation


Hue
Saturation
Value

Як ми задамо чорний колір в моделі HSI?

Powered by Poll Everywhere

Start the presentation to see live content. For screen share software, share the entire screen. Get help at **pollev.com/app**

$$\begin{cases} H = \begin{cases} \theta;\ B \le G \\ 360 - \theta; B > G \end{cases}, \text{где } \theta = \arccos\left(\dfrac{\frac{1}{2} * ((R-G) + (R-B))}{\sqrt{(R-G)^2 + (R-B)(G-B)}}\right) \\ S = 1 - \dfrac{3}{(R+G+B)} \min(R, G, B) \\ I = \dfrac{1}{3}(R+G+B) \end{cases}$$

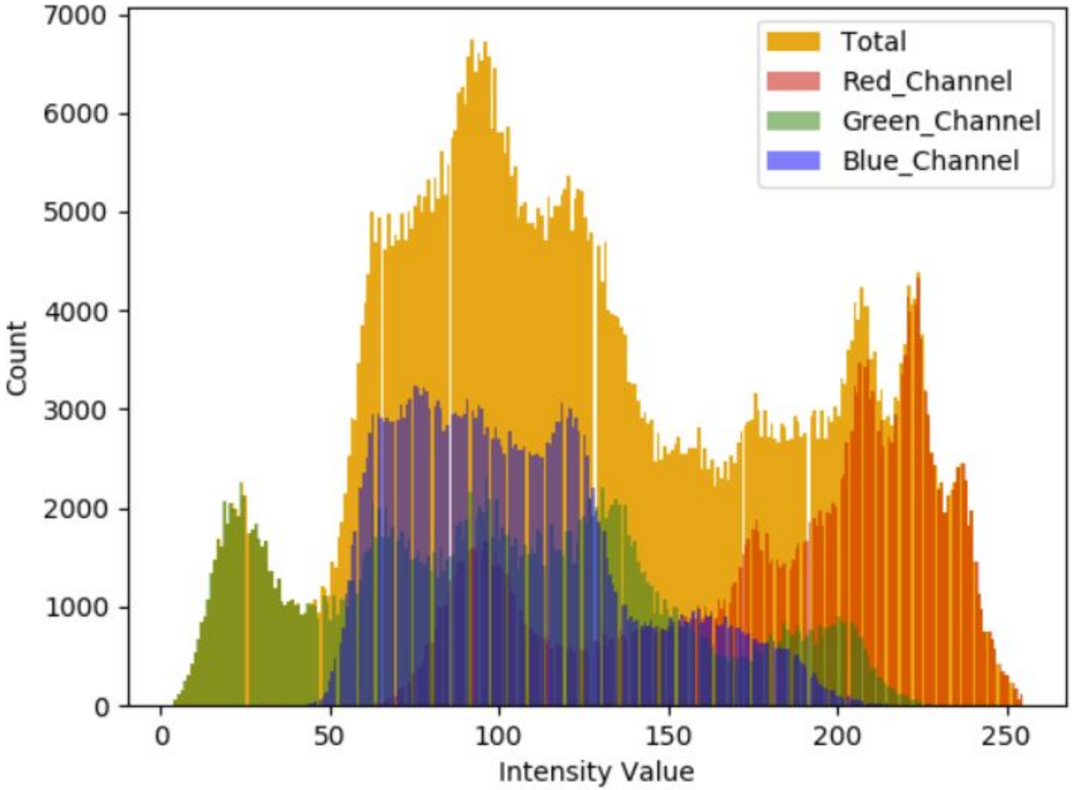
Гістограма цифрових зображень

Нехай є скалярне зображення I з пікселями (i,j,u) де 0≤u≤255. Ми визначаємо абсолютні частоти рахуючи скільки раз значення u зустрічається в носії Ω, який містить всі пікселі.

$$H_I(u) = \left|\{(x, y) \in \Omega : I(x, y) = u\}\right|,$$

Відносні частоти - значення між 0 і 1 - можна порівняти з функцією щільності ймовірності розподілу дискретної випадкової величини I (p):

$$h_I(u) = \frac{H_I(u)}{|\Omega|}$$



Вирівнювання гістограми

I ⟶ Inew, для всіх u.

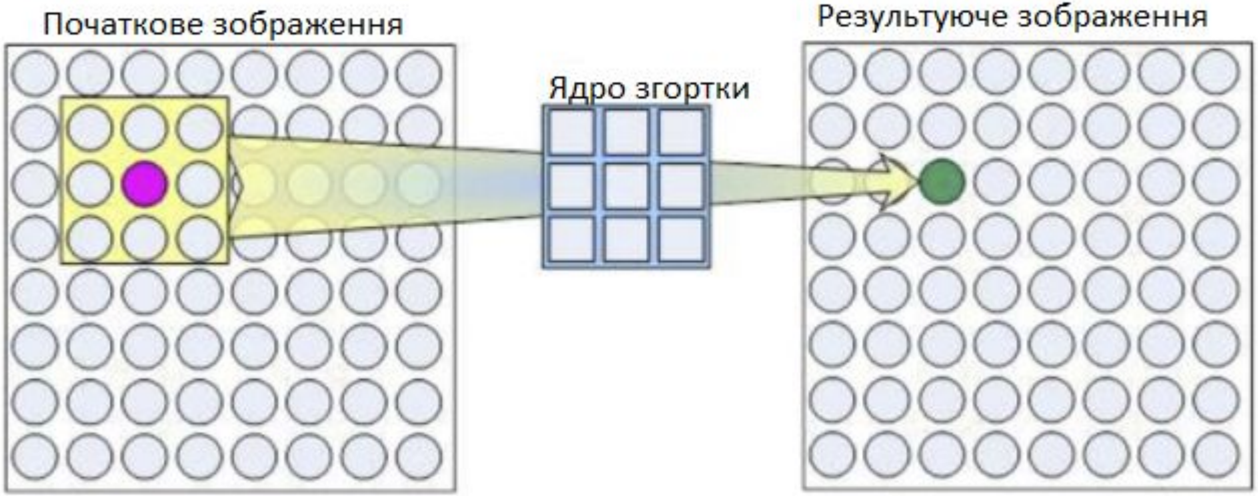
$$H_{I_{new}}(u) = \mathrm{const} = \frac{N_{cols} \cdot N_{rows}}{G_{\max} + 1}$$

I розміром Ncols x Nrows з абсолютним частотами $H_I(u)$

Перетворення здійснюється за допомогою

градаційної функції: $g(u) = c_I(u) \cdot G_{\max}$

Згладжування. Фільтр Гауса



$$G_{\sigma,\mu_x,\mu_y}(x,y) = \frac{1}{2\pi\sigma^2}\exp\left(-\frac{(x-\mu_x)^2+(y-\mu_y)^2}{2\sigma^2}\right) =$$

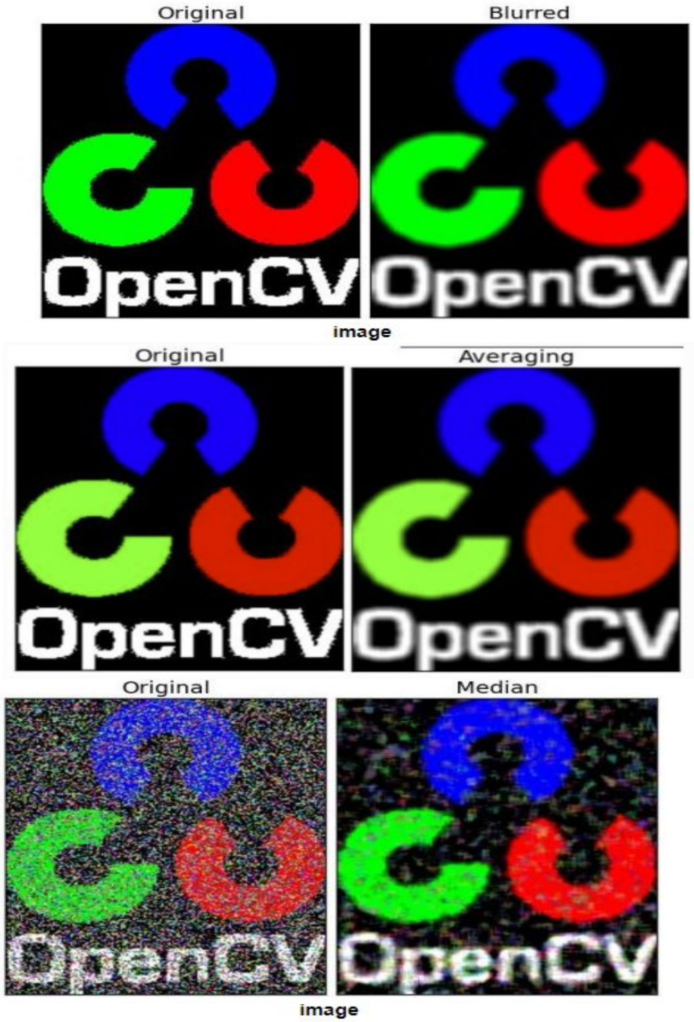
$$= \frac{1}{2\pi\sigma^2}\cdot e^{-\frac{(x-\mu_x)^2}{2\sigma^2}}\cdot e^{-\frac{(y-\mu_y)^2}{2\sigma^2}},$$

Гаусівський простір масштабів

$$L(x, y, a\sigma),\ a^n\cdot\sigma,\ a>1$$
$$n = 0, 1, \ldots, m$$

$$G_\sigma(x,y) = \frac{1}{2\pi\sigma^2}\exp\left(-\frac{x^2+y^2}{2\sigma^2}\right) = \frac{1}{\pi s}\cdot e^{-\frac{x^2}{2\sigma^2}}\cdot e^{-\frac{y^2}{2\sigma^2}}.$$

cv.GaussianBlur()



$\{4, 7, 3, 1, 8, 7, 4, 5, 2, 3, 8\}$ $\longrightarrow$ Медіана 4

$1, 2, 3, 3, 4, 4, 5, 7, 7, 8, 8.$

When poll is active, respond at **pollev.com/nataliashovgun288**

Що таке гаусівський масштаб?

- Середнє значення всіх пікселів
- Максимальне значення пікселів у вікні
- Стандартне відхилення
- Ступінь розмитості зображення

Powered by Poll Everywhere

Start the presentation to see live content. For screen share software, share the entire screen. Get help at **pollev.com/app**

LoG

Лапласіан гаусіана

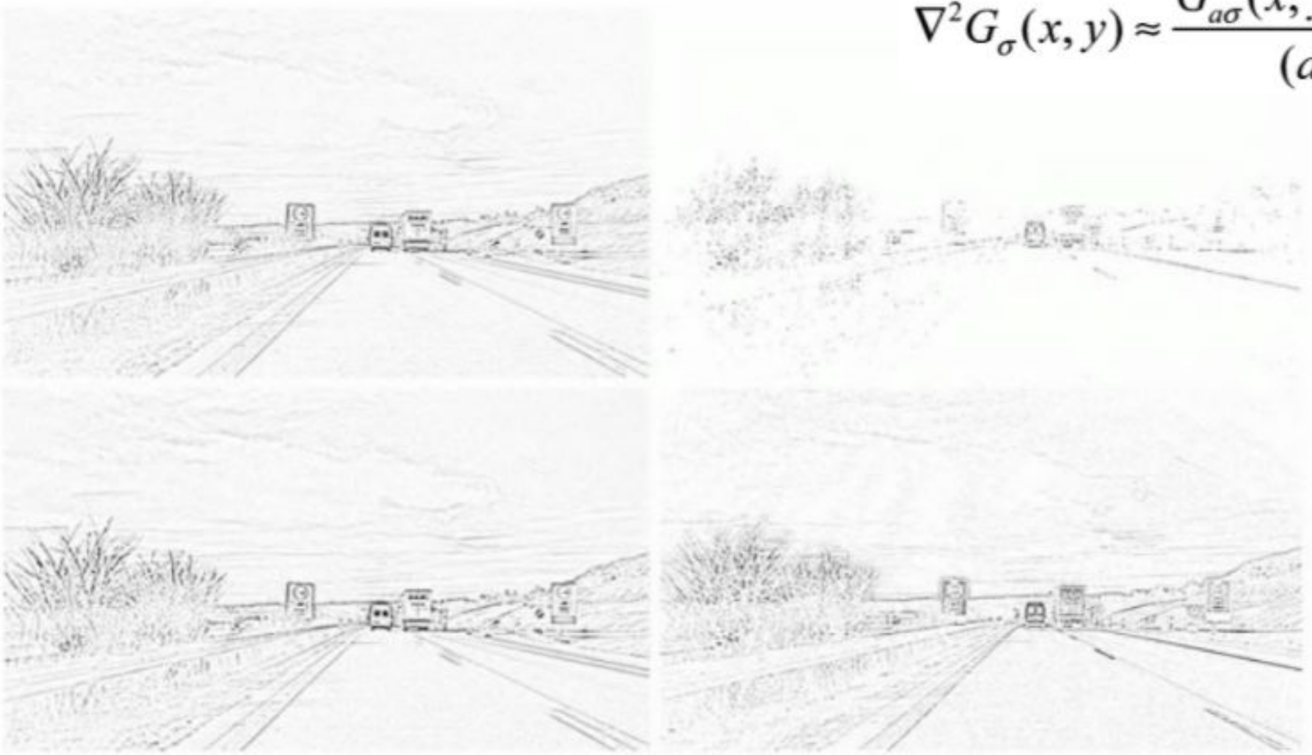
$$\nabla^2(G_\sigma * I) = I * \nabla^2 G_\sigma$$

DoG

Різниця гаусіан $D_{\sigma,a}(x, y) = L(x, y, \sigma) - L(x, y, a\sigma)$

a>1 - масштабний коефіцієнт

$$\nabla^2 G_\sigma(x, y) \approx \frac{G_{a\sigma}(x, y) - G_\sigma(x, y)}{(a-1)\sigma^2}, \text{ де } a = 1,6$$



```
Cv2.resize,
cv2.getRotationMatrix2D,
cv2.warpAffine
```

Аугментація

$$(x_t,\ y_t) = T[(x,\ y)]$$

$$\begin{bmatrix} x_t \\ y_t \end{bmatrix} = \begin{bmatrix} cos(\Theta) & sin(\Theta) \\ -sin(\Theta) & cos(\Theta) \end{bmatrix} \begin{bmatrix} yx \end{bmatrix}$$

1. Масштаб

$$(x_t,\ y_t) = T[(x,\ y)] = (c_x \cdot x,\ c_y \cdot y)$$

$$\begin{bmatrix} x_t \\ y_t \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} y \\ x \end{bmatrix} + \begin{bmatrix} t_x t_y \end{bmatrix}$$

$$\begin{bmatrix} x_t \\ y_t \end{bmatrix} = \begin{bmatrix} 1 & s_h \\ s_v & 1 \end{bmatrix} \begin{bmatrix} yx \end{bmatrix}$$