January 13, Kyiv

Data Science & Mathematical Modeling Master Program

Course “Hands-on Machine Learning”

Lecture 10: Ensemble methods

Oleg CHERTOV

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

Applied Mathematics Department
Igor Sikorsky Kyiv Polytechnic Institute
Ukraine

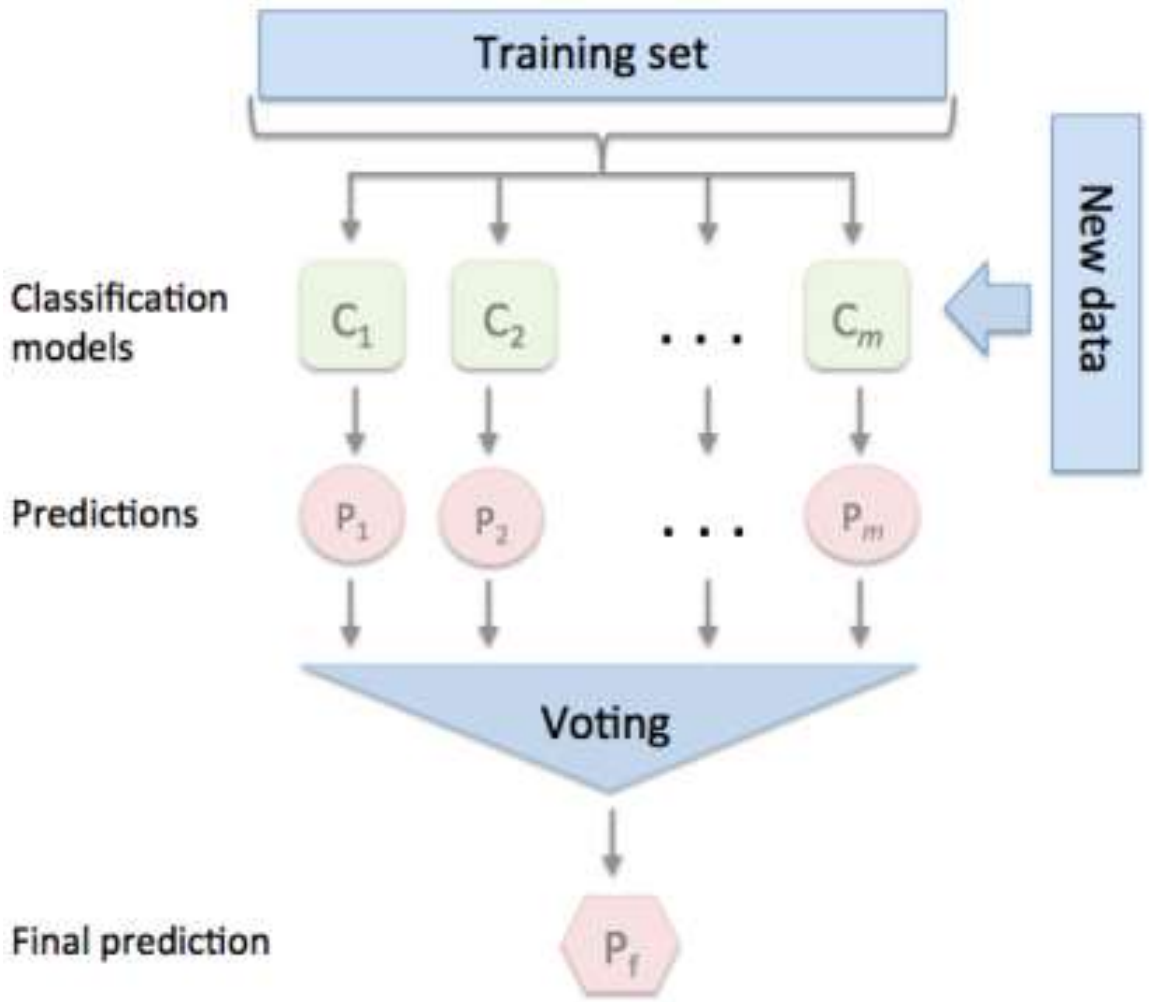
Одна голова – добре, а дві – краще!

- У житті справедливість цього прислів’я очевидна
- Якщо замінити «голову» на «характеристику», то прислів’я стане очевидним і для машинного навчання
- Результати можна ще покращити, якщо комбінувати не просто характеристики, а цілі моделі
- Комбінація моделей = **ансамбль моделей**
- Ансамблі моделей, зазвичай, переважають інші методи за своєю ефективністю, але за це потрібно платити складністю алгоритмів та моделей

А чи варто?

- Перші два місця в конкурсі рекомендувальних алгоритмів Netflix Prize $1M

Загальна схема ансамблю моделей



За рахунок чого перевага ансамблю над окремим класифікатором? (1)

- Дискретна випадкова величина ξ має біноміальний розподіл, якщо ймовірність набуття нею конкретних значень має вигляд:

$$P(\xi = k) = C_n^k p^k q^{n-k}, \; k=0, 1, \ldots n,$$

де p, n — параметри, що визначають розподіл,

p ϵ [0,1], q=1-p, n ϵ N

- В ТЙМС біноміальний розподіл є дискретним ймовірнісним розподілом, що характеризує кількість «успіхів» в послідовності експериментів, значення яких змінюється за принципом «так/ні», з ймовірністю p

За рахунок чого перевага ансамблю над окремим класифікатором? (2)

- Нехай розв'язується задача бінарної класифікації
- Нехай всі n базових класифікаторів мають однакову частоту появи помилки ε
- Нехай класифікатори є незалежними і частоти помилок не корельовані
- При таких допущеннях ймовірність появи помилки ансамблю з базових класифікаторів рахується як функція маси ймовірності біноміального розподілу:

$$\varepsilon_{\text{ансамблю}} = \sum_{k}^{n} C_n^k \varepsilon^k (1-\varepsilon)^{n-k}$$

За рахунок чого перевага ансамблю над окремим класифікатором? (3)

- Нехай розв’язується задача бінарної класифікації
- Нехай всі n=3 базових класифікаторів мають однакову частоту появи помилки ε
- Нехай класифікатори є незалежними і частоти помилок не корельовані
- При таких допущеннях ймовірність появи помилки ансамблю з базових класифікаторів дорівнює:

$\varepsilon_{\text{ансамблю}} = ??$

$$\sum_{k}^{n} C_n^k \varepsilon^k (1-\varepsilon)^{n-k}$$

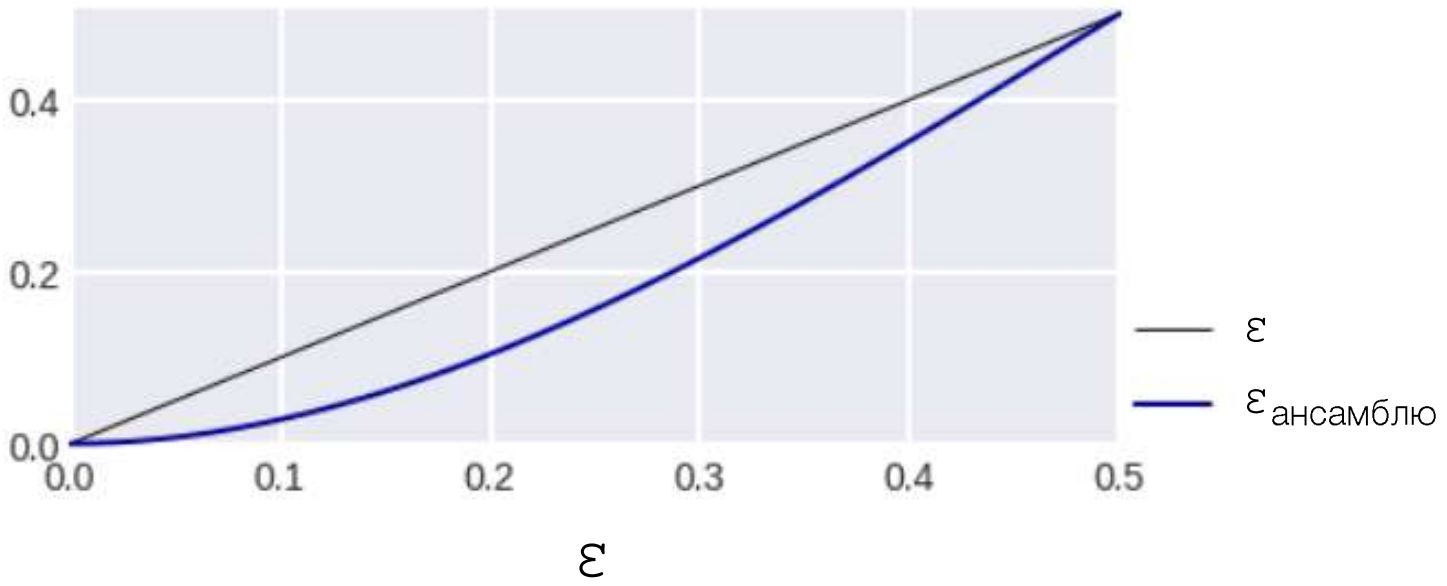
За рахунок чого перевага ансамблю над окремим класифікатором? (4)

- Нехай розв’язується задача бінарної класифікації
- Нехай всі n=3 базових класифікаторів мають однакову частоту появи помилки ε
- Нехай класифікатори є незалежними і частоти помилок не корельовані
- При таких допущеннях ймовірність появи помилки ансамблю з базових класифікаторів дорівнює:

$$\varepsilon_{\text{ансамблю}} = \varepsilon^2(3-2\varepsilon)$$

За рахунок чого перевага ансамблю над окремим класифікатором? (5)

- $\varepsilon_{\text{ансамблю}} = \varepsilon^2(3-2\varepsilon)$



За рахунок чого перевага ансамблю над окремим класифікатором? (6)

- Розглянемо приклад із 11 базових класифікаторів з частотою помилки 0,25: n = 11, ε = 0,25

Тоді $\varepsilon_{\text{ансамблю}} = \sum_{k=6}^{11} C_{11}^{k} 0{,}25^{k} (1-0{,}25)^{11-k} = 0{,}034$

На порядок кращий результат!

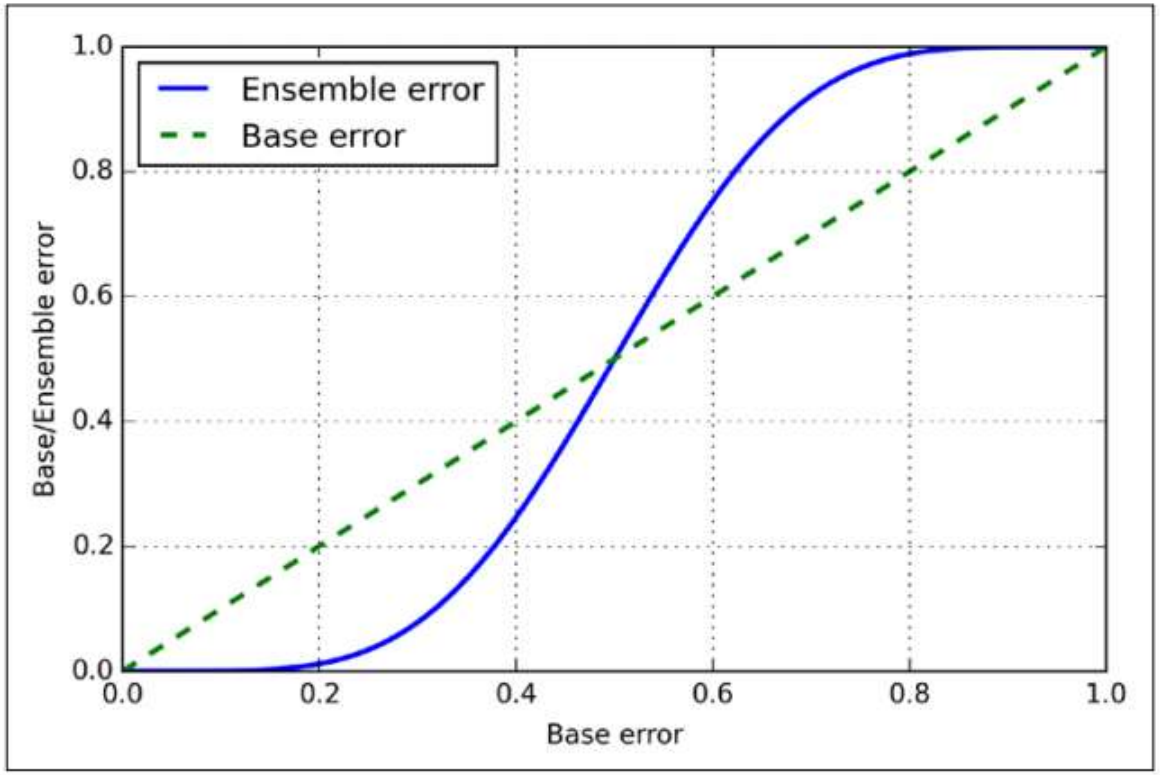
У загальному випадку, в силу нерівності Хьофдинга:

$\varepsilon_{\text{ансамблю}} \leq e^{-\frac{1}{2} n (2\varepsilon - 1)^2}$,

тобто помилка ансамблю експоненційно спадає зі зростанням кількості базових алгоритмів

За рахунок чого перевага ансамблю над окремим класифікатором? (7)

- Імовірність ансамблевої помилки завжди краще того ж показника окремого базового класифікатора до тих пір, поки базові класифікатори працюють краще випадкового вгадування (ε<0,5):

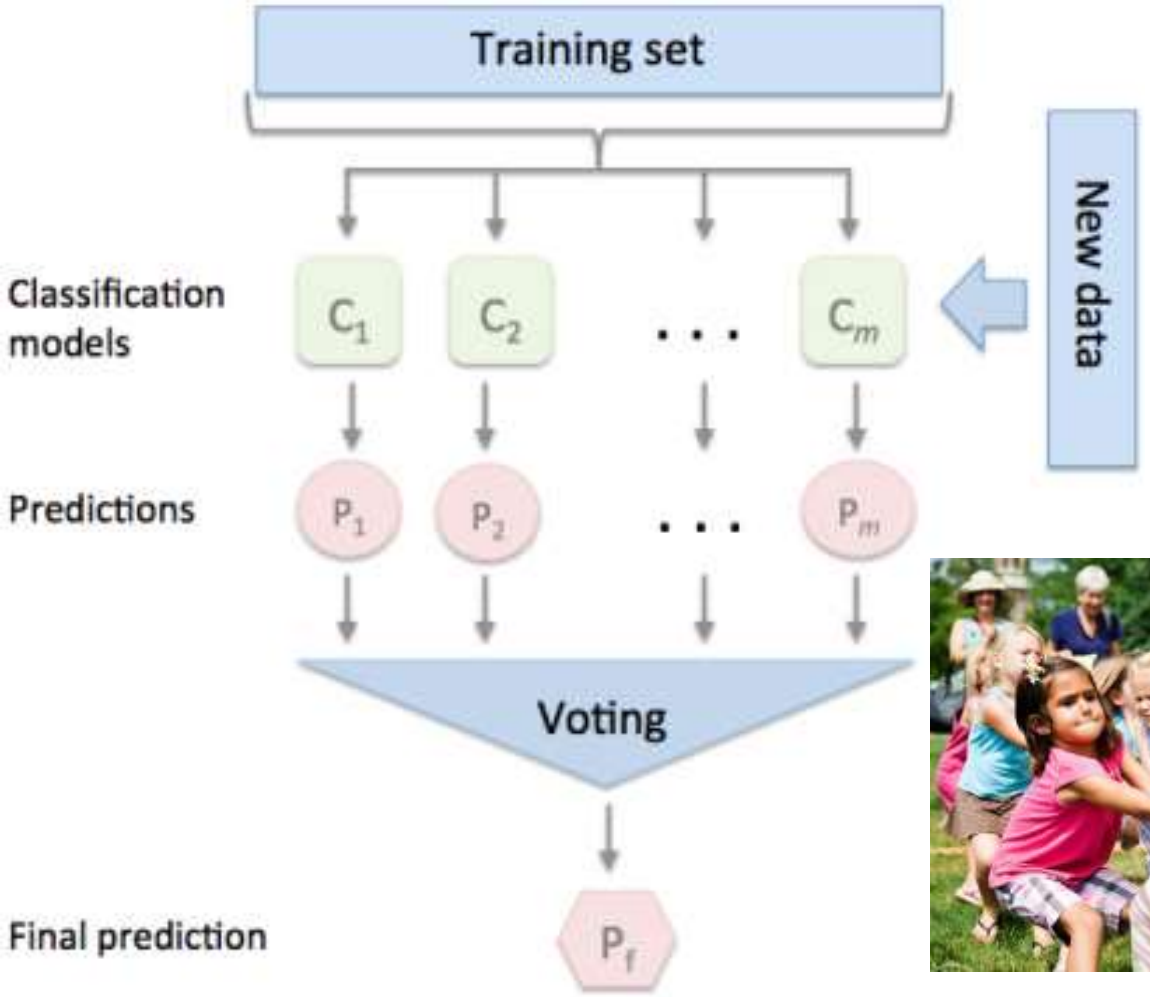


Загальна схема ансамблю моделей

Базові моделі можуть бути **різними**, маємо комбінацію класифікаторів (=combining classifiers):

- дерева прийняття рішень
- наївний байєсівський класифікатор
- логістична регресія
- SVM
- …

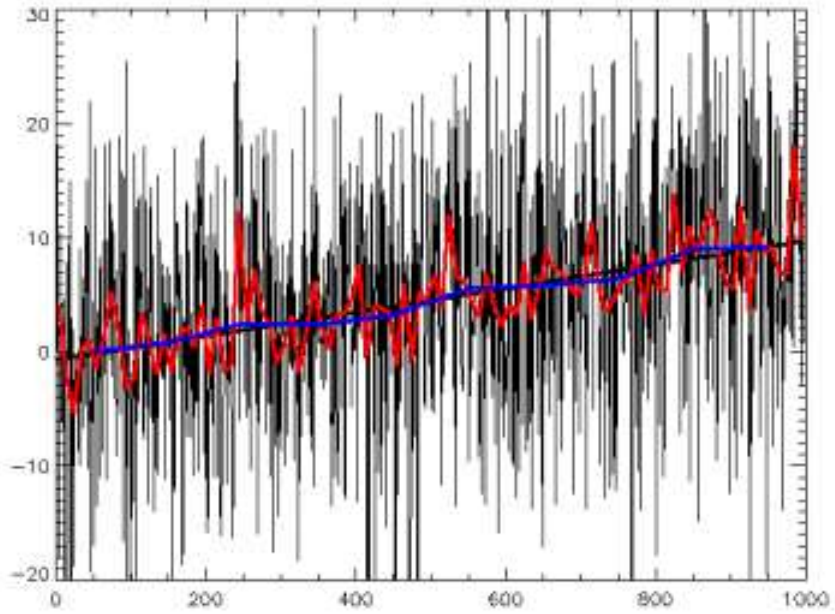
або **однаковими**, маємо ансамбль слабких моделей (=ensemble of weak learners)



Спостереження зі статистики (1)

- Усереднення результатів спостережень (=вимірювань) може дати більш стійку та надійну оцінку, оскільки зменшується вплив випадкових флуктуацій в окремому вимірюванні

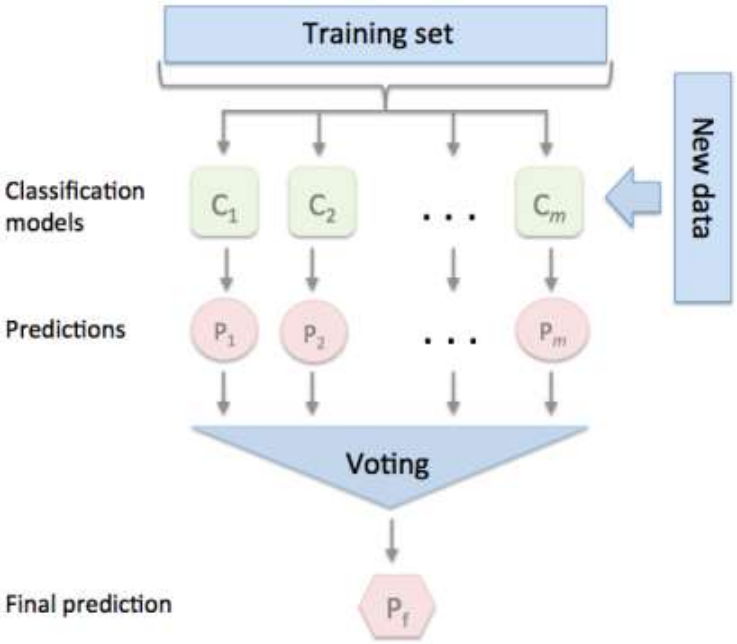
Серед шуму можна виділити сам сигнал, тренд тощо



Спостереження зі статистики (2)

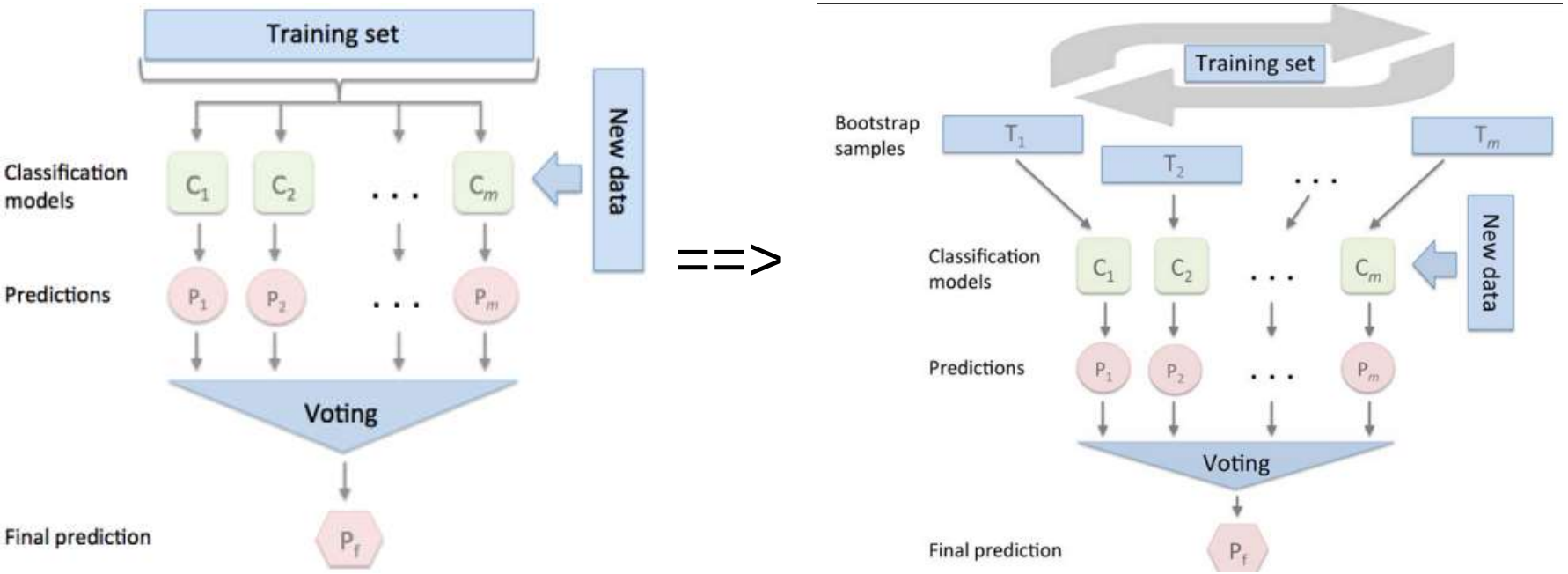
- Усереднення результатів спостережень (=вимірювань) може дати більш стійку та надійну оцінку, оскільки зменшується вплив випадкових флуктуацій в окремому вимірюванні
- Тому якби вдалося побудувати ансамбль трішки різних моделей по одним і тим же навчальним даним, то ми змогли б зменшити вплив випадкових флуктуацій в окремих моделях
- Ключове питання: як забезпечити необхідну різноманітність моделей?

Як забезпечити необхідну різноманітність моделей?



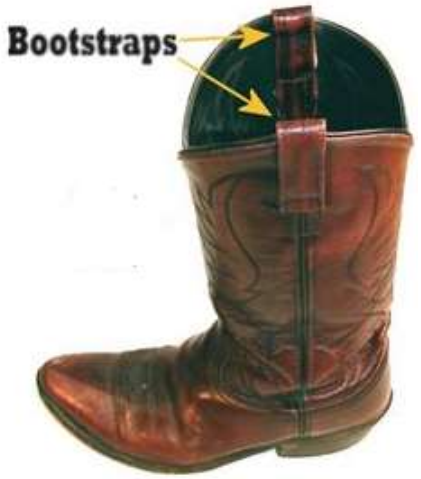
==> ???

Як забезпечити необхідну різноманітність моделей?

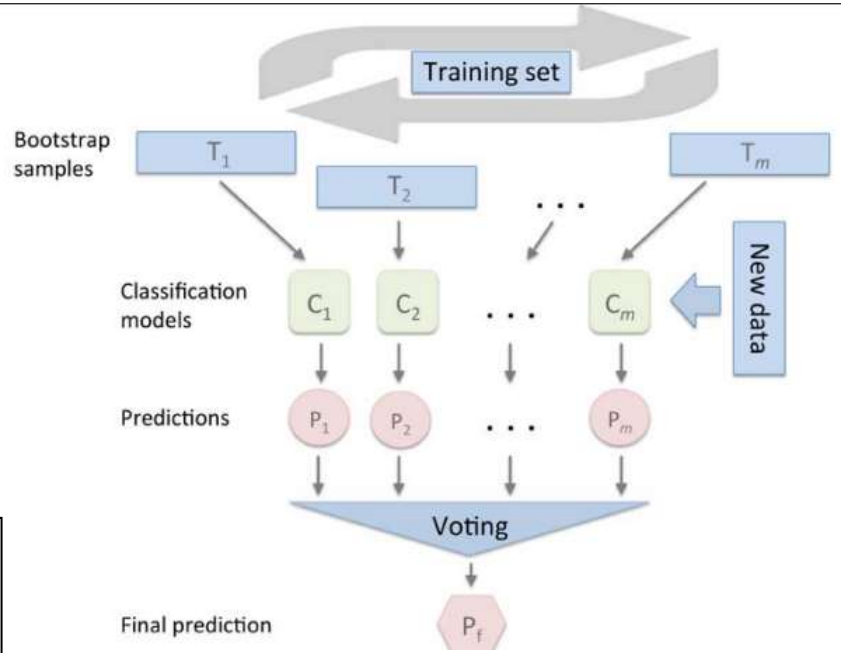


За рахунок урізноманітнення навчальної вибірки!

Як забезпечити необхідну різноманітність моделей?



Bootstrap – вушко, петля на заднику черевика чи калоші чобота



В IT цей термін має значення початкового завантаження:
a technique of loading a program into a computer by means of a few initial instructions which enable the introduction of the rest of the program from an input device

Bootstrap sample = random sample with replacement (випадкова вибірка з поверненням)

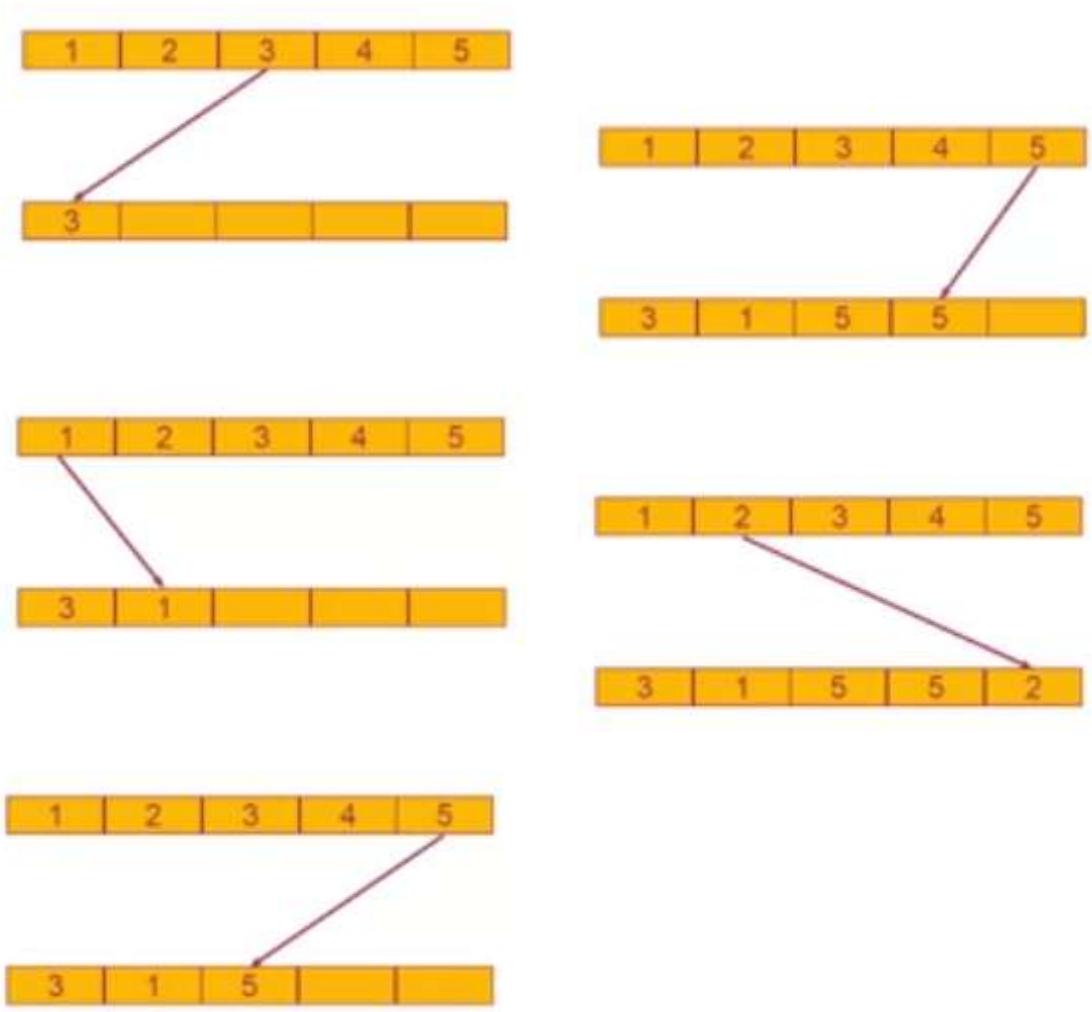
Bootstrap method

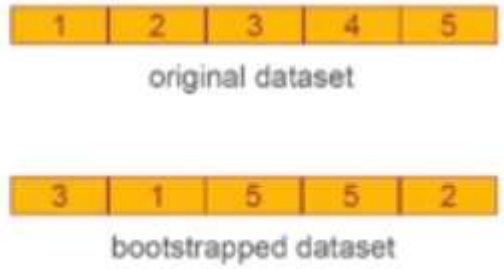
- The term “bootstrap” is a reference to the notion of “pulling oneself up by the bootstraps” when the usual methods for ascertaining statistical significance do not apply

{Efron, Bradley and Tibishirani. 1993. An Introduction to the Bootstrap. New York: Chapman & Hall/CRC]

- Цей метод часто застосовується в статистиці, коли потрібно мати багато вибірок, а фінансів чи часу на їх побудову не має

Example of bootstrap sample





- Those observations not chosen for the sample may be used as out of sample observations

- In the case of evaluating a machine learning model, the model is fit on the drawn sample and evaluated on the out-of-bag sample:

  oob = [4]

Python code of bootstrap sample

- Чи можна повторювати дані у вибірці, що будується?
- Розмір вибірки, що будується?
- А коли потрібна менша?
- Стала, що ініціалізує генератор псевдо-випадкових чисел

```
1  # scikit-learn bootstrap
2  from sklearn.utils import resample
3  # data sample
4  data = [0.1, 0.2, 0.3, 0.4, 0.5, 0.6]
5  # prepare bootstrap sample
6  boot = resample(data, replace=True, n_samples=4, random_state=1)
7  print('Bootstrap Sample: %s' % boot)
8  # out of bag observations
9  oob = [x for x in data if x not in boot]
10 print('OOB Sample: %s' % oob)
```

```
1 Bootstrap Sample: [0.6, 0.4, 0.5, 0.1]
2 OOB Sample: [0.2, 0.3]
```

Скільки початкових елементів буде відсутнім в bootstrap sample?

- Оскільки випадкові вибірки робляться із заміною, то в загальному випадку **bootstrap sample** містить дублікати
- Відповідно в ній будуть відсутні якісь елементи з початкової навчальної вибірки, навіть якщо розмір вибірки, що будується, залишається таким же, як і початкової
- Знайдемо приблизну кількість початкових елементів, відсутніх в **bootstrap sample** після певної кількості ітерацій

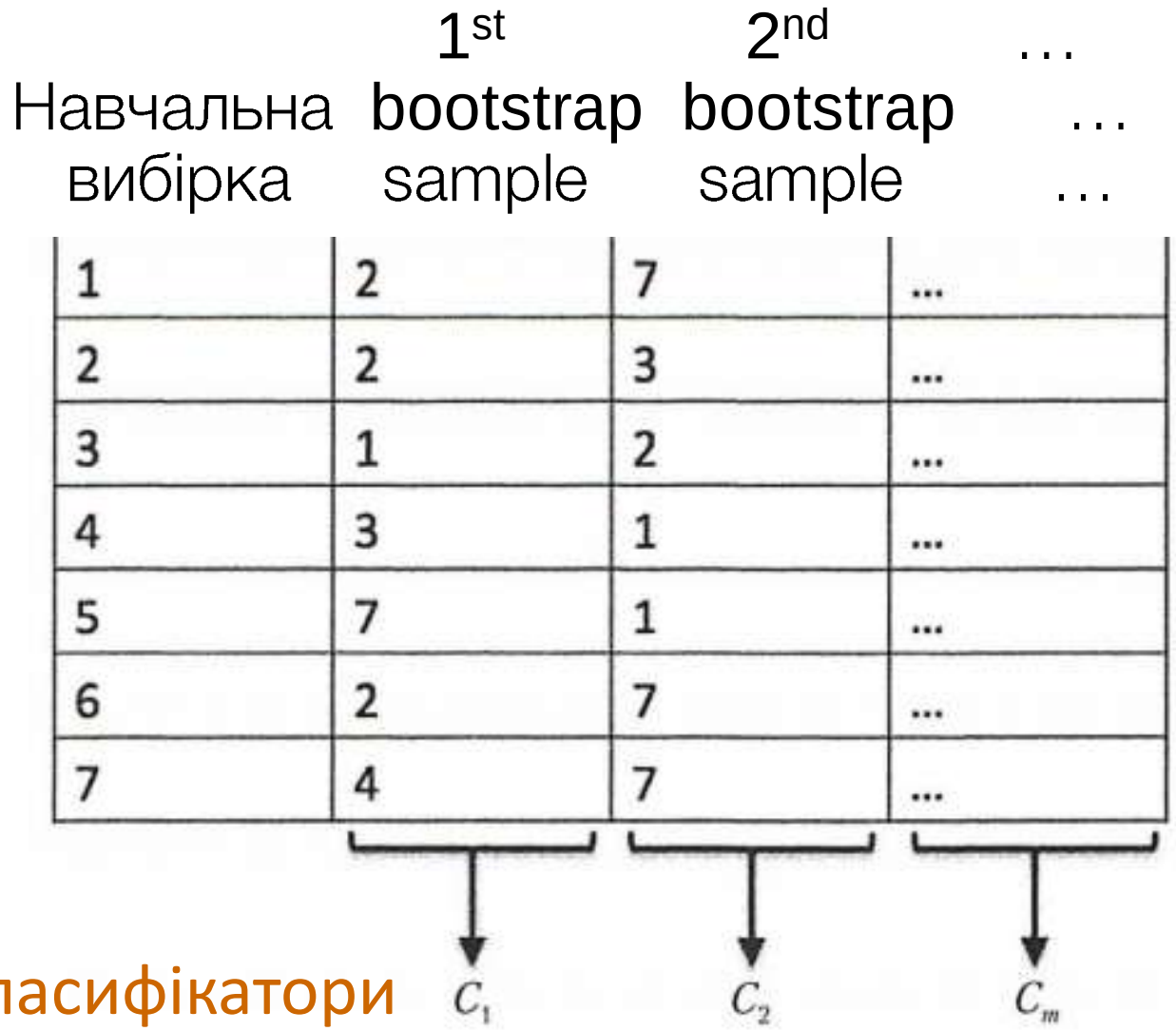
Скільки початкових елементів буде відсутнім в bootstrap sample?

- Знайдемо приблизну кількість початкових елементів, відсутніх в bootstrap sample після певної кількості ітерацій
- Спочатку знайдемо ймовірність того, що конкретний елемент не входить до bootstrap sample розміром n :

Скільки початкових елементів буде відсутнім в bootstrap sample?

- Знайдемо приблизну кількість початкових елементів, відсутніх в **bootstrap sample** після певної кількості ітерацій
- Спочатку знайдемо ймовірність того, що конкретний елемент не входить до **bootstrap sample** розміром **n** : $(1-1/n)^n$
- Вже при **n=5** ця ймовірність дорівнює ≈0,328
- При **n→∞** ймовірність прямує до 1/e ≈ 0,368
- Отже, в середньому в кожній **bootstrap sample** відсутня третина початкових елементів!

Bootstrap sample



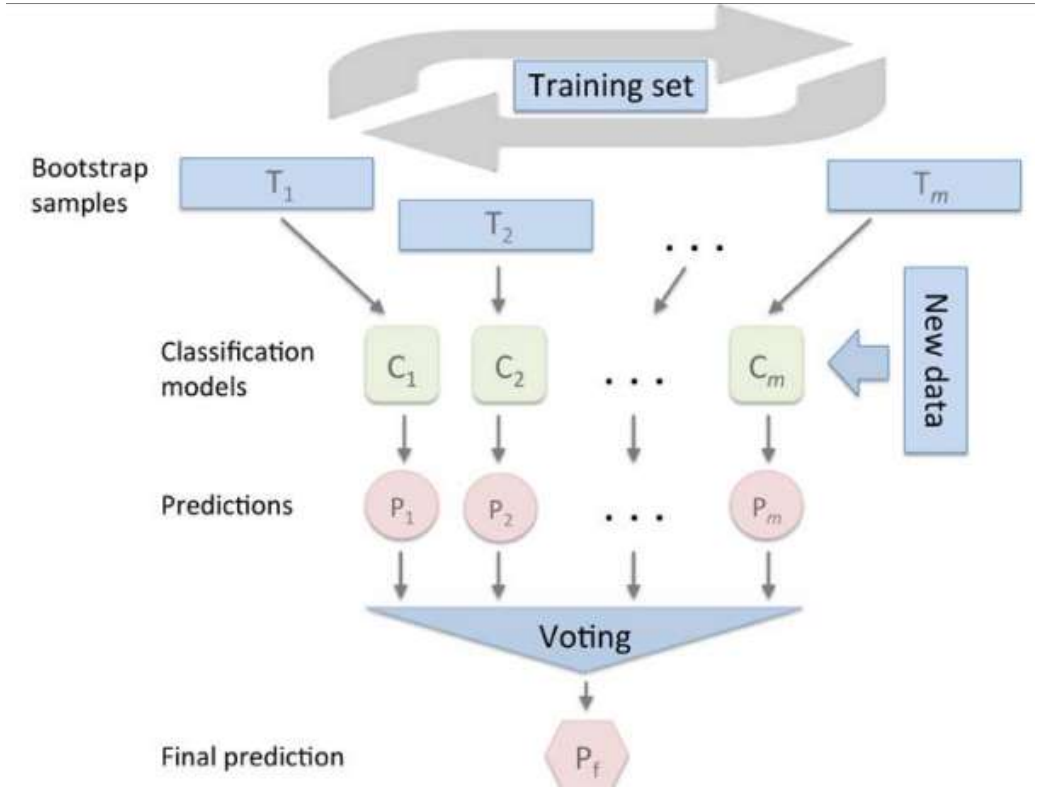
базові класифікатори

I. Bagging = Bootstrap + Aggregating

- **Ідея методу**

❖ Навчити кілька незалежних моделей на підставі випадково обраних (bootstrap) підмножин об'єктів з навчальної вибірки

❖ Класифікацію проводити на підставі результату голосування (aggregating) моделей



Algorithm Bagging

Algorithm Bagging($D, T, \mathscr{A}$)

Input : data set D; ensemble size T; learning algorithm $\mathscr{A}$.

Output : ensemble of models whose predictions are to be combined by voting or averaging.

1 **for** $t = 1$ to T **do**

2 | build a bootstrap sample D_t from D by sampling $|D|$ data points with replacement;

3 | run $\mathscr{A}$ on D_t to produce a model M_t;

4 **end**

5 **return** $\{M_t | 1 \leq t \leq T\}$

Приклади агрегуючих функцій

- **Пример 1:** Простое голосование (Simple Voting):

$$F\big(b_1(x),\ldots,b_T(x)\big) = \frac{1}{T}\sum_{t=1}^{T} b_t(x), \quad x \in X.$$

- **Пример 2:** Взвешенное голосование (Weighted Voting):

$$F\big(b_1(x),\ldots,b_T(x)\big) = \sum_{t=1}^{T} \alpha_t b_t(x), \quad x \in X, \quad \alpha_t \in \mathbb{R}.$$

- **Пример 3:** Смесь алгоритмов (Mixture of Experts)

$$F\big(b_1(x),\ldots,b_T(x)\big) = \sum_{t=1}^{T} g_t(x) b_t(x), \quad x \in X, \quad g_t \colon X \to \mathbb{R}.$$

Голосування: просте і зважене

Чтобы получше разобраться в принципе работы *взвешивания*, обратимся теперь к более конкретному примеру. Пусть имеется ансамбль из трех базовых классификаторов C_j ($j \in \{0,1\}$) и нужно идентифицировать метку класса конкретного экземпляра x. Два из трех базовых классификаторов идентифицируют метку класса 0, и один C_3 прогнозирует, что образец принадлежит классу 1. Если взвесить прогнозы каждого базового классификатора одинаково, то мажоритарное голосование спрогнозирует, что образец принадлежит классу 0:

$C_1(\boldsymbol{x}) \rightarrow 0$, $C_2(\boldsymbol{x}) \rightarrow 0$, $C_3(\boldsymbol{x}) \rightarrow 1$;
$\hat{y} = \text{mode}\{0, 0, 1\} = 0$.

Теперь назначим вес 0.6 классификатору C_3 и соответственно взвесим классификаторы C_1 и C_2 на коэффициент 0.2.

$$\hat{y} = \arg\max_i \sum_{j=1}^{m} w_j \chi_A(C_j(\boldsymbol{x}) = i)$$

$$= \arg\max_i \left[0.2 \times i_0 + 0.2 \times i_0 + 0.6 \times i_1\right] = 1.$$

В более интуитивном представлении, поскольку $3 \times 0.2 = 0.6$, можно сказать, что прогноз, сделанный классификатором C_3, имеет вес в три раза больше, чем прогнозы классификаторов, соответственно, C_1 или C_2.

Усереднення як агрегуюча функція

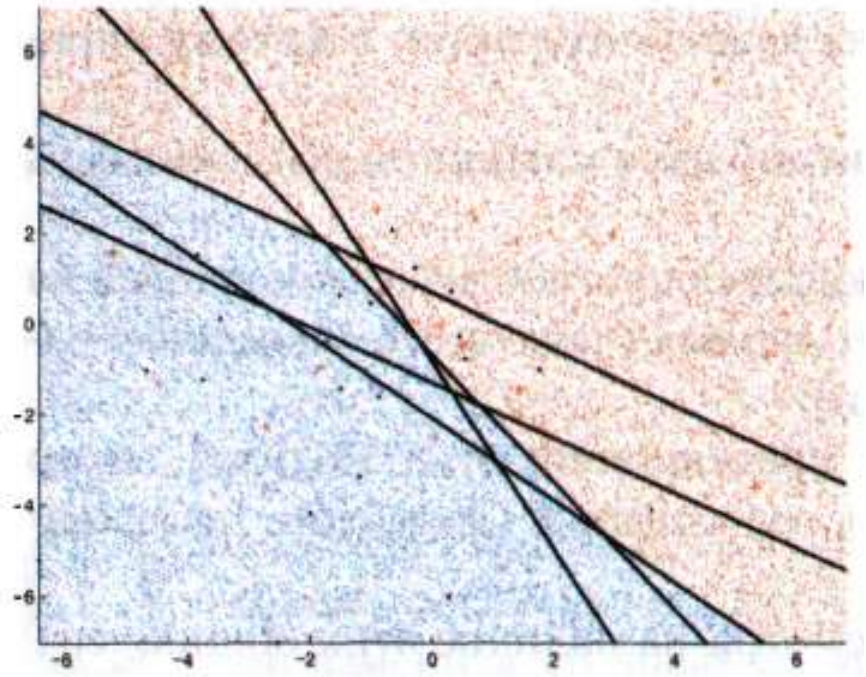
Продолжая с нашим предыдущим примером, пусть имеется задача двухклассовой классификации с метками классов $i \in \{0,1\}$ и ансамбль из трех классификаторов C_j ($j \in \{1,2,3\}$). И пусть классификатор C_j возвращает следующие вероятности принадлежности классу для отдельно взятого образца $\boldsymbol{x}$:

$C_1(\boldsymbol{x}) \rightarrow [0.9, 0.1]$, $C_2(\boldsymbol{x}) \rightarrow [0.8, 0.2]$, $C_3(\boldsymbol{x}) \rightarrow [0.4, 0.6]$.

Тогда отдельные вероятности класса можно вычислить следующим образом:

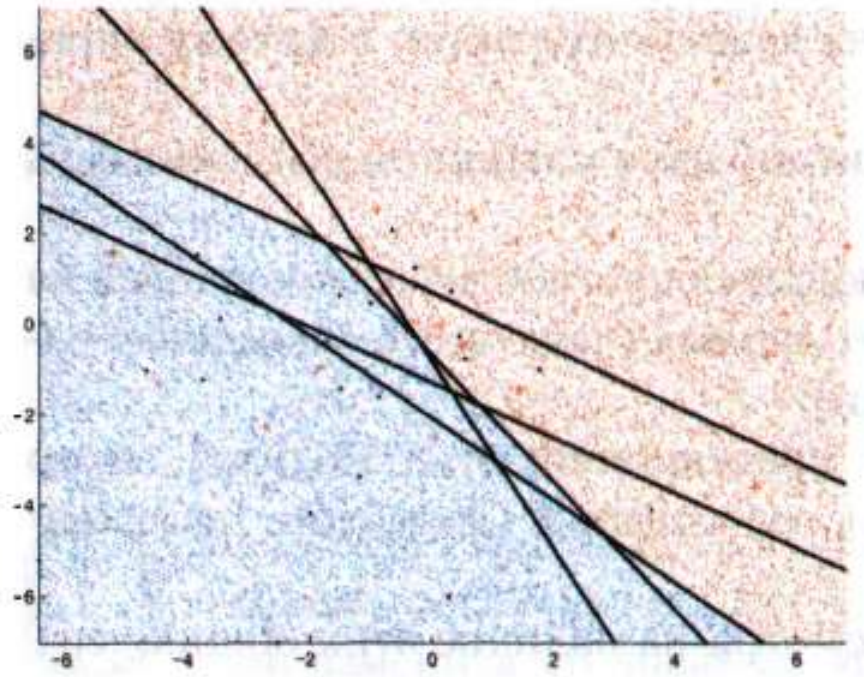
$$p(i_0|\boldsymbol{x}) = 0.2 \times 0.9 + 0.2 \times 0.8 + 0.6 \times 0.4 = 0.58;$$
$$p(i_1|\boldsymbol{x}) = 0.2 \times 0.1 + 0.2 \times 0.2 + 0.6 \times 0.6 = 0.42;$$
$$\hat{y} = \arg\max_i \left[p(i_0|\boldsymbol{x}), p(i_1|\boldsymbol{x}) \right] = 0.$$

Bagging. Example



- An ensemble of five basic linear classifiers built from bootstrap samples with bagging
- The decision rule is majority vote, leading to a piecewise linear decision boundary

Bagging. Example



- An ensemble of five basic linear classifiers built from bootstrap samples with bagging
- The decision rule is majority vote, leading to a piecewise linear decision boundary

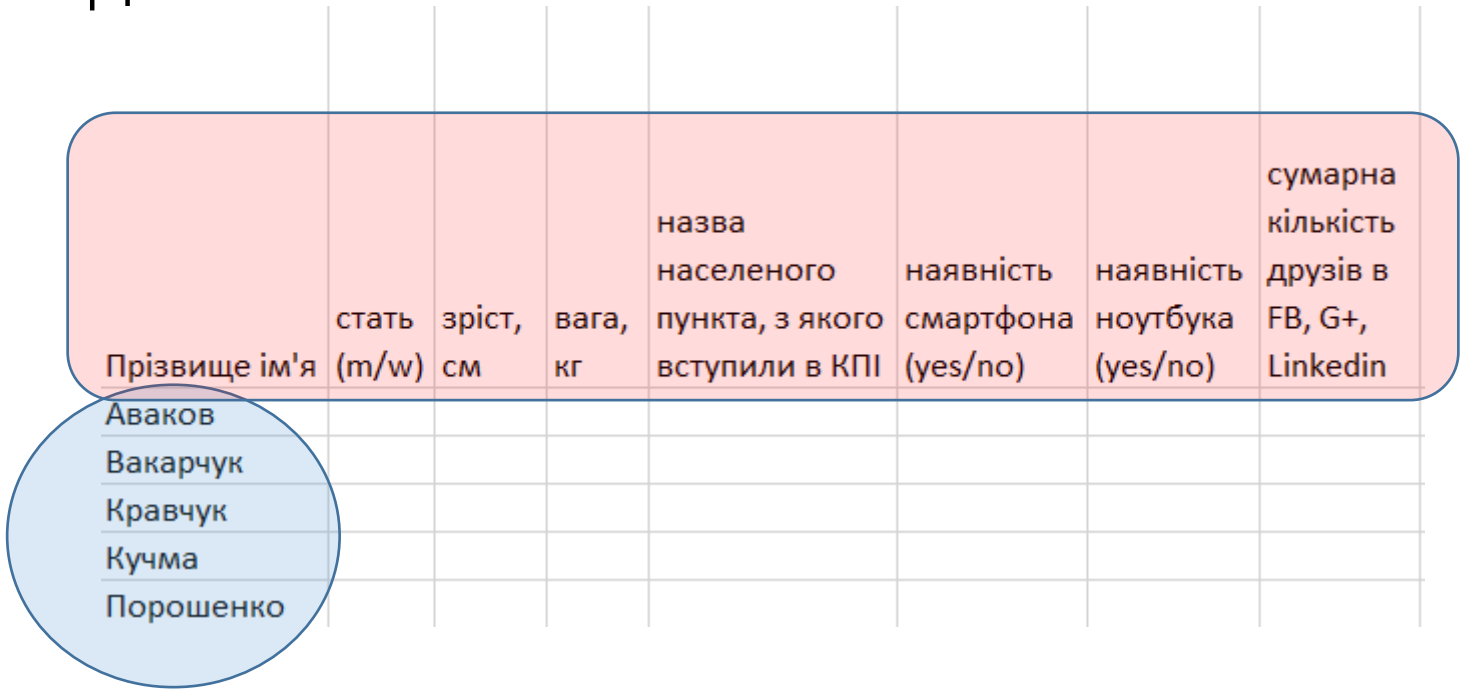
- Сила ансамблю – з лінійних класифікаторів робить нелінійний!

Доповнюючий підхід до bagging

- Так, можна будувати **bootstrap sample** із початкової навчальної вибірки
- Яким іншим чином також можна підвищити різноманітність ансамбля моделей?

Вибірку перепробували, а ...

- Які є інші ідеї для побудови ансамблів моделей?



Прізвище ім'я	стать (m/w)	зріст, см	вага, кг	назва населеного пункта, з якого вступили в КПІ	наявність смартфона (yes/no)	наявність ноутбука (yes/no)	сумарна кількість друзів в FB, G+, Linkedin
Аваков							
Вакарчук							
Кравчук							
Кучма							
Порошенко							

Випадковий ліс. Ідея

- Яким іншим чином також можна підвищити різноманітність ансамбля моделей?
- Subspace sampling - випадкова вибірка підпростору характеристик!
- Випадковий ліс (Random Forest) = bagging + subspace sampling для моделі дерев прийняття рішень
- Додатковий бонус - скорочення часу навчання кожної моделі (= дерева прийняття рішень)

Випадковий ліс. Алгоритм

Algorithm RandomForest(D, T, d)

Input : data set D; ensemble size T; subspace dimension d.

Output : ensemble of tree models whose predictions are to be combined by voting or averaging.

1 **for** $t = 1$ to T **do**

2 build a bootstrap sample D_t from D by sampling $|D|$ data points with replacement;

3 select d features at random and reduce dimensionality of D_t accordingly;

4 train a tree model M_t on D_t without pruning;

5 **end**

6 **return** $\{M_t | 1 \leq t \leq T\}$

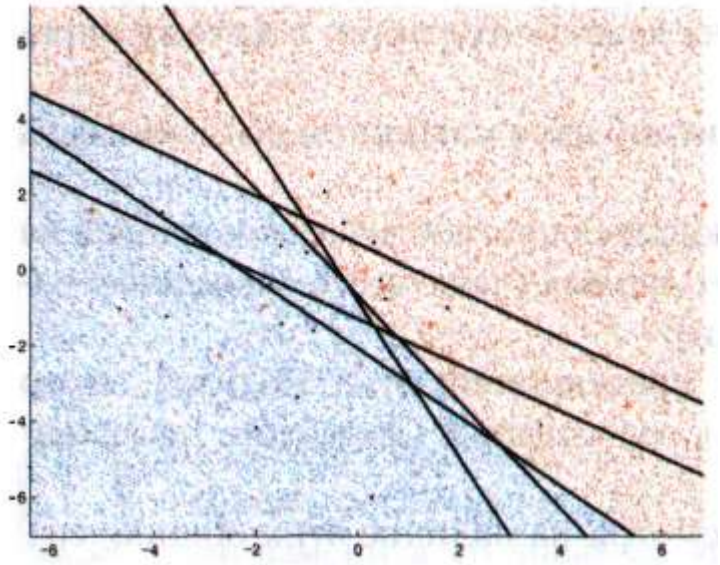
Випадковий ліс. Переваги (1)

- Листя дерева прийняття рішень утворюють розбиття простору об'єктів
- Розбиття простору об'єктів випадковим лісом є перетином розбиттів, утворених деревами, що входять до ансамблю моделей
- І хоча таке розбиття в загальному випадку достатньо мілке, по ньому можна відновити єдине дерево прийняття рішень, тобто алгоритм побудови випадкового лісу фактично є реалізацією альтернативного алгоритму навчання

Випадковий ліс. Переваги (2)

+ Висока точність
+ Мало перенавчання
+ Легко розпаралелюється

- Повільний без паралелізму



- На відміну наприклад, від баггінгу лінійних класифікаторів, при якому вирішувальний кордон ансамблю є кусково-лінійним, тобто не може бути отриманий як результат навчання одного базового класифікатора

Перепробували і вибірку, і ознаки

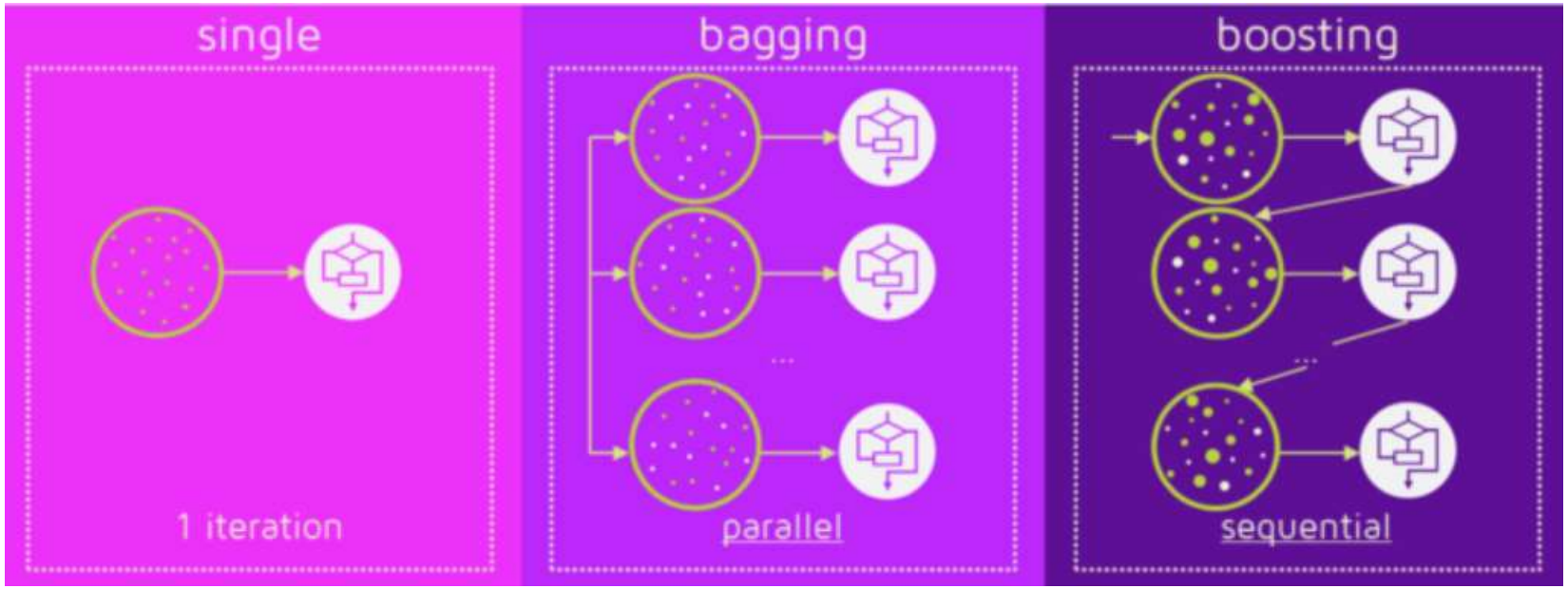
- Які є інші ідеї для побудови ансамблів моделей?

Прізвище ім'я	стать (m/w)	зріст, см	вага, кг	назва населеного пункта, з якого вступили в КПІ	наявність смартфона (yes/no)	наявність ноутбука (yes/no)	сумарна кількість друзів в FB, G+, Linkedin
Аваков							
Вакарчук							
Кравчук							
Кучма							
Порошенко							

Перепробували і вибірку, і ознаки

Метод	Опис
Bagging	Підвибірка навчальної вибірки береться за допомогою бутстрепа
Pasting	Випадкова навчальна підвибірка
Random Subspaces (випадкові підпростори)	Випадкова підмножина характеристик
Random Patches (випадкові шматки)	Одночасно беремо і випадкову підмножину об’єктів і випадкову підмножину характеристик

II. Boosting (=підсилення)

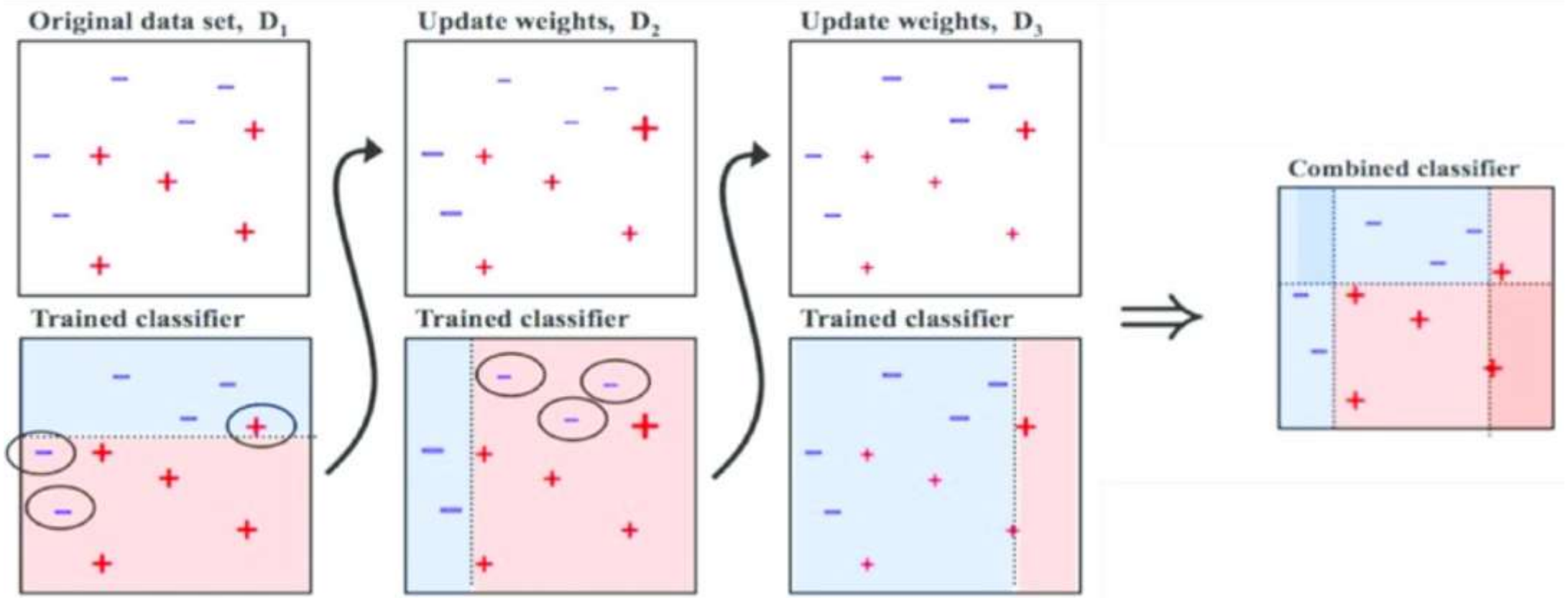


- **Ідея методу** Boosting

❖ Послідовно навчати слабкі моделі так, щоб кожна наступна модель "виправляла помилки" попередніх

❖ Для передбачення використовувати комбінацію з усіх моделей послідовності

Алгоритм методу Boosting (на прикладі AdaBoost)



- **Адаптивне підсилення** (Adaptive Boosting) працює шляхом зважування спостережень, надаючи більшої ваги екземплярам, що важко класифікуються, і меншої – тим, які вже добре оброблені
- Послідовно додаються нові слабкі алгоритми, які зосереджують навчання на складніших шаблонах

Алгоритм методу AdaBoost

Кроки реалізації **алгоритму** AdaBoost:

1. Спочатку всім точкам присвоєно рівні ваги
2. Модель будується на вибірці даних
3. За цією моделлю передбачення робляться для всього набору даних
4. За прогнозами та істинними значеннями обчислюються помилки
5. У побудові наступної моделі найбільші ваги надаються точкам даних, на передбаченні яких алгоритм помилився
6. Вага може бути визначена за величиною помилки, а саме: чим більша помилка, тим більша вага
7. Цей процес повторюється, поки функція помилки не перестане змінюватися або поки не буде досягнуто максимальної кількості предикторів

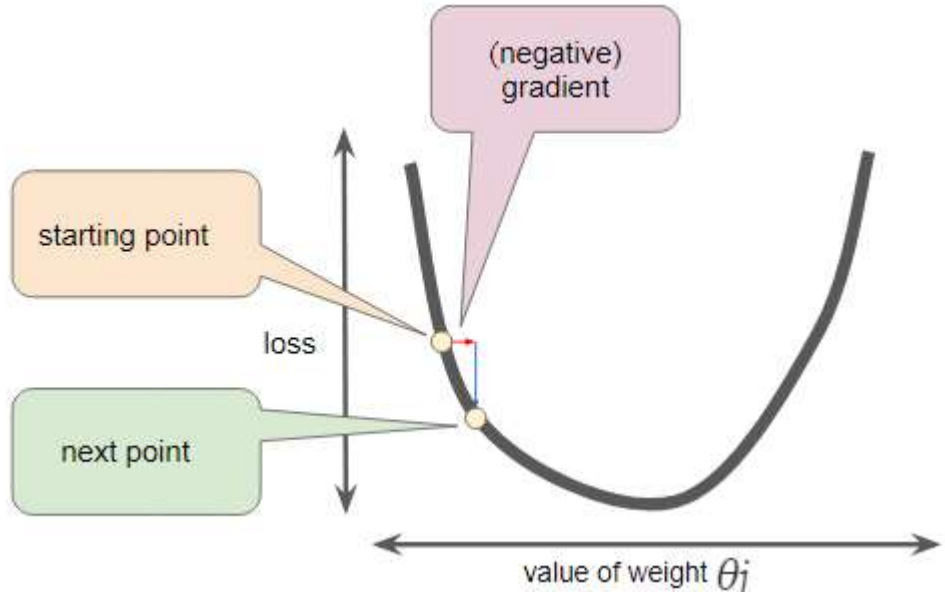
Parameters	Remarks
object base_estimator, default=None	The base estimator from which the boosted ensemble is built. If None, then the base estimator is DecisionTreeClassifier initialized with max_depth=1 or DecisionTreeRegressor initialized with max_depth=3
int n_estimators, default=50	The maximum number of estimators at which boosting is terminated. In case of perfect fit, the learning procedure is stopped early
float learning_rate, default=1.0	Weight applied to each classifier/regressor at each boosting iteration. A higher learning rate increases the contribution of each classifier/regressor. There is a trade-off between the learning_rate and n_estimators parameters
int random_state, RandomState instance or None, default=None	Controls the random seed given at each base_estimator at each boosting iteration. Thus, it is only used when base_estimator exposes a random_state. Pass an int for reproducible output across multiple function calls
algorithm {'SAMME', 'SAMME.R'}, default='SAMME.R'	(only for sklearn.ensemble.AdaBoostClassifier) If 'SAMME.R' then use the SAMME.R real boosting algorithm. base_estimator must support calculation of class probabilities. If 'SAMME' then use the SAMME discrete boosting algorithm. The SAMME.R algorithm typically converges faster than SAMME, achieving a lower test error with fewer boosting iterations
loss {'linear', 'square', 'exponential'}, default='linear'	(only for sklearn.ensemble.AdaBoostRegressor) The loss function to use when updating the weights after each boosting iteration

Градієнтний бустинг

- **Градієнтний бустинг** працює послідовно, додаючи до минулих моделей нові так, щоб виправлялися помилки, допущені попередніми предикторами

- Градієнтний бустинг відрізняється від адаптивного тим, що, на відміну від AdaBoost, що змінює ваги при кожній ітерації, градієнтний намагається навчати нові моделі за помилкою минулих (рухаючись до мінімуму функції втрат)

- Для кращого розуміння цього алгоритму важливо розібратися у роботі градієнтного спуску



Градієнтний спуск (Gradient Descent)

Наша мета - підібрати такий коефіцієнт, щоб функція втрат (різниця між істинною відповіддю та передбаченням моделі) стала мінімальною, тобто щоб прогноз максимально наблизився до істинного значення

Градієнтний бустинг

Кроки реалізації **алгоритму градієнтного бустингу**:

1. Модель будується на вибірці даних
2. За цією моделлю передбачення робляться для всього набору даних
3. За прогнозами та істинними значеннями обчислюються помилки
4. Нова модель будується з урахуванням помилок як цільових змінних. При цьому ми прагнемо знайти найкращий поділ для мінімізації помилки
5. Передбачення, зроблені за допомогою цієї нової моделі, поєднуються з попередніми прогнозами
6. Цей процес повторюється, поки функція помилки не перестане змінюватися або поки не буде досягнуто максимальної кількості предикторів

Градієнтний бустинг

КОМПОЗИЦИЯ

$$a_N(x) = \sum_{n=1}^{N} b_n(x)$$

Базовый алгоритм

Не усереднюємо, а додаємо.
Бо хочемо виправити помилку попередніх алгоритмів.

Функція втрат рахується для кожного об’єкту вибірки

ФУНКЦИЯ ПОТЕРЬ

- Ошибка на одном объекте: $L(y, z)$

Истинный ответ | Прогноз

- Ошибка на одном объекте: $L(y, z)$
- MSE : $L(y, z) = (z - y)^2$
- Логистическая функция потерь:

$$L(y, z) = \log\left(1 + \exp\left(-yz\right)\right)$$

- ...

Градієнтний бустинг. Ініціалізація

ИНИЦИАЛИЗАЦИЯ

- $b_0(x)$ — первый алгоритм в композиции
- Примеры:
  - $b_0(x) = 0$
  - $b_0(x) = \frac{1}{\ell} \sum_{i=1}^{\ell} y_i$
  - $b_0(x) = \underset{y \in Y}{\operatorname{argmax}} \sum_{i=1}^{\ell} [y_i = y]$

Перший алгоритм повинен бути простим, щоб ми не гаяли багато часу:

- для регресії – стала чи середнє значення по виборці,
- для класифікації – самий популярний клас

Градієнтний бустинг. Навчання базового алгоритму

ОБУЧЕНИЕ БАЗОВОГО АЛГОРИТМА

Будуємо ітеративно

- Уже построили:

$$a_{N-1}(x) = \sum_{n=0}^{N-1} b_n(x)$$

- Задача:

$$\sum_{i=1}^{\ell} L(y_i, a_{N-1}(x_i) + b(x_i)) \to \min_b$$

ОПТИМАЛЬНЫЙ СДВИГ

- Какие прогнозы оптимальны для обучающей выборки?

$$\sum_{i=1}^{\ell} L(y_i, a_{N-1}(x_i) + s_i) \to \min_{s_1,\dots,s_\ell}$$

- Вектор сдвигов: $s = (s_1, \dots, s_\ell)$

$$F(s) = \sum_{i=1}^{\ell} L(y_i, a_{N-1}(x_i) + s_i) \to \min_s$$

Мінімізує вектор «антиградієнт», бо він направлений в бік найшвидшого спадання функції!

Градієнтний бустинг. Оптимальний зсув

- Вектор сдвигов: $s = (s_1, \ldots, s_\ell)$

$$F(s) = \sum_{i=1}^{\ell} L(y_i, a_{N-1}(x_i) + s_i) \to \min_s$$

Using a first-order Taylor approximation, the value of l can be approximated as follows:

$$l(y_i, F_{m-1}(x_i) + h_m(x_i)) \approx l(y_i, F_{m-1}(x_i)) + h_m(x_i) \left[\frac{\partial l(y_i, F(x_i))}{\partial F(x_i)} \right]_{F=F_{m-1}}$$

Note: Briefly, a first-order Taylor approximation says that $l(z) \approx l(a) + (z-a)\frac{\partial l(a)}{\partial a}$. Here, z corresponds to $F_{m-1}(x_i) + h_m(x_i)$, and a corresponds to $F_{m-1}(x_i)$

ОПТИМАЛЬНЫЙ СДВИГ

- Сдвинемся в сторону наискорейшего убывания:

Сдвиг по первому объекту

$$s = -\nabla F = \left(-L'_z(y_1, a_{N-1}(x_1)), \ldots, -L'_z(y_\ell, a_{N-1}(x_\ell)) \right)$$

From https://scikit-learn.org/stable/modules/ensemble.html#gradient-boosting

Градієнтний бустинг. Навчання базового алгоритму

ОБУЧЕНИЕ БАЗОВОГО АЛГОРИТМА

$$b_N(x) = \operatorname*{argmin}_b \frac{1}{\ell} \sum_{i=1}^{\ell} (b(x_i) - s_i)^2$$

- Вся информация о функции потерь L содержится в сдвигах s_i
- Используем MSE независимо от исходной задачи

Градієнтний бустинг. Загальна схема

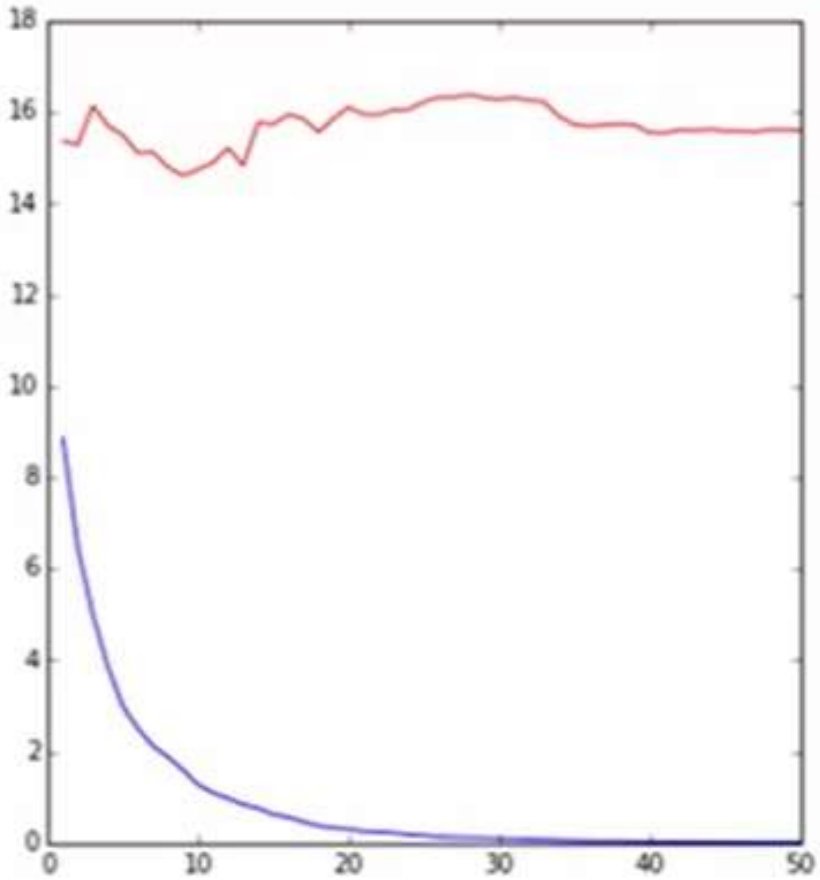
ГРАДИЕНТНЫЙ БУСТИНГ

- Построить начальный алгоритм $b_0(x)$
- Для $n = 1, \ldots, N$:
  - Вычислить сдвиги:
    $$s = \Big(-L'_z(y_1, a_{n-1}(x_1)), \ldots, -L'_z(y_\ell, a_{n-1}(x_\ell)) \Big)$$
  - Обучить новый базовый алгоритм:
    $$b_N(x) = \operatorname*{argmin}_b \frac{1}{\ell} \sum_{i=1}^{\ell} (b(x_i) - s_i)^2$$
  - Добавить алгоритм в композицию:
    $$a_n(x) = \sum_{m=1}^{n} b_m(x)$$

РЕЗЮМЕ

- Градиентный бустинг последовательно строит композицию
- Базовый алгоритм приближает антиградиент функции ошибки
- Результат — градиентный спуск в пространстве алгоритмов

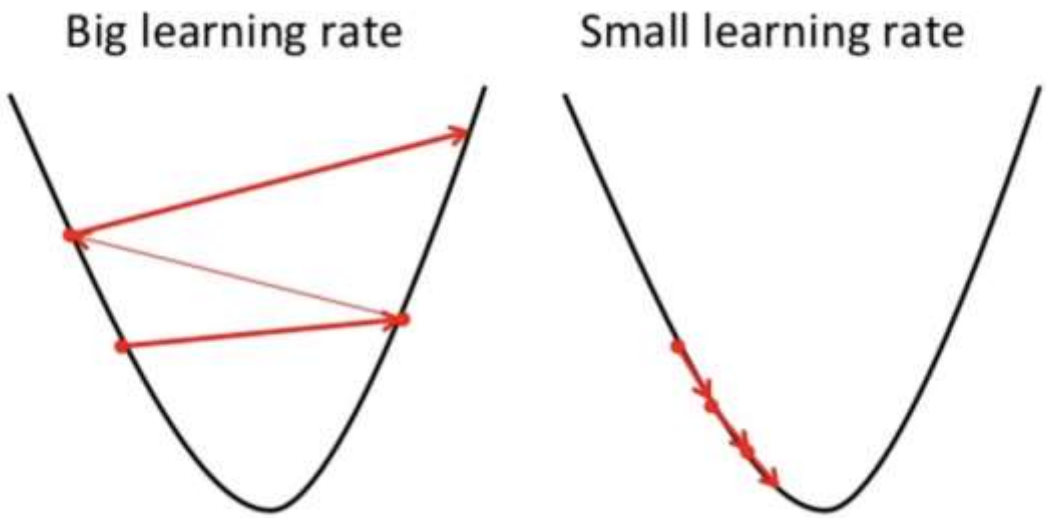
Перенавчання в градієнтному бустингу та його причини



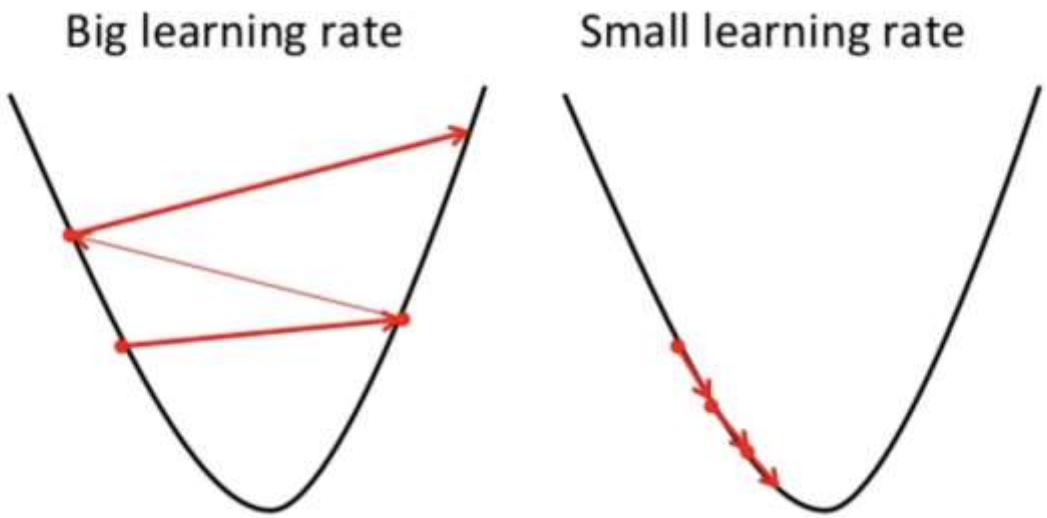
ПРИЧИНЫ ПЕРЕОБУЧЕНИЯ

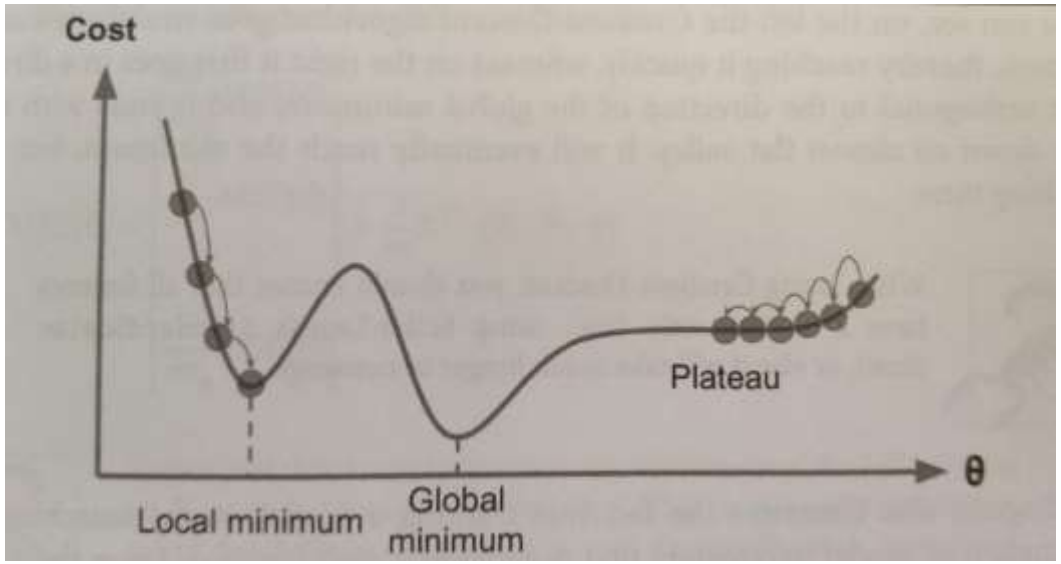
- Базовый алгоритм должен приближать вектор антиградиента
- Базовые алгоритмы очень слабые
- Приближение плохое
- Вместо градиентного спуска получаем случайное блуждание

Перенавчання в градієнтному бустингу та його причини



Перенавчання в градієнтному бустингу та його причини





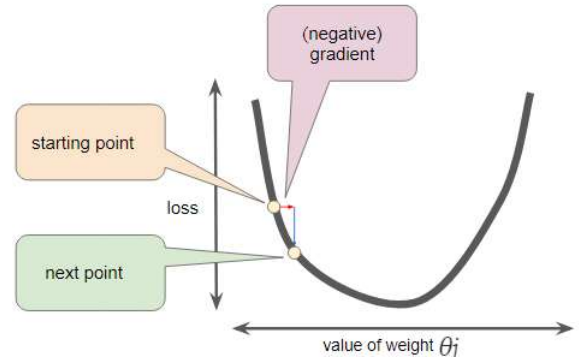
Градієнтний бустинг. Зменшення кроку

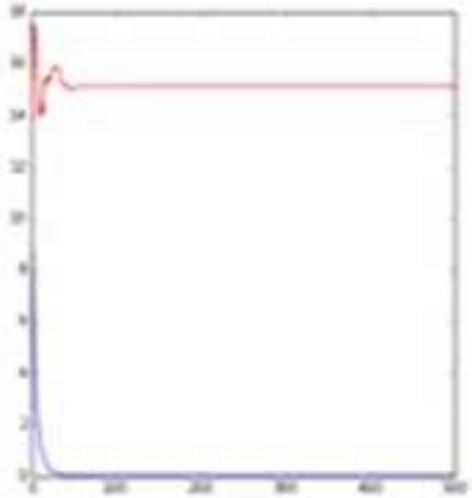
- Не будем доверять базовому алгоритму:

$$a_N(x) = a_{N-1}(x) + \eta b_N(x)$$

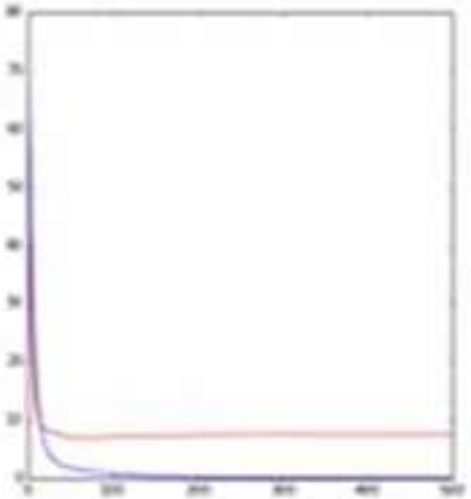
- $\eta \in (0, 1]$ — длина шага

shrinkage or learning rate

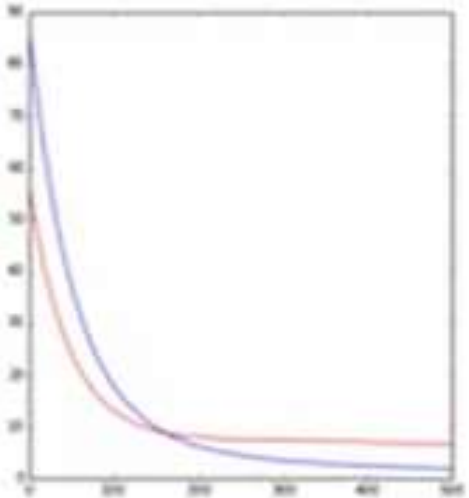




$\eta = 1$



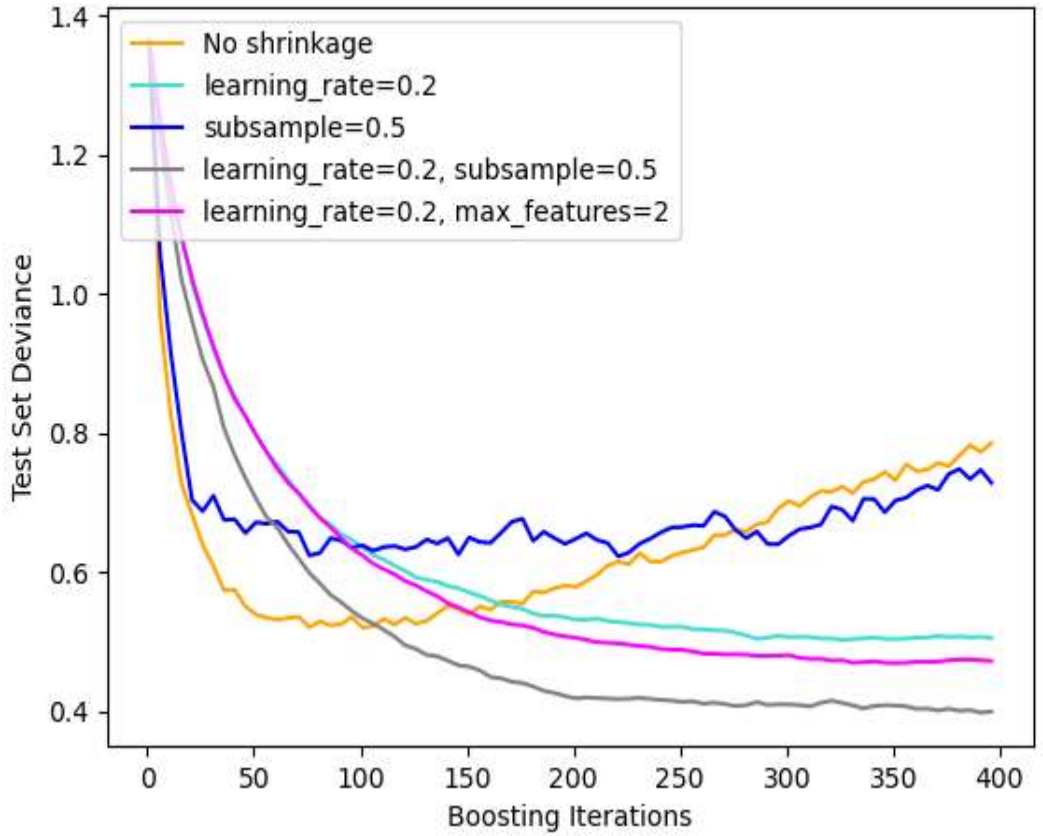
$\eta = 0.1$



$\eta = 0.01$

- Чем меньше шаг, тем больше нужно базовых алгоритмов
- Сокращение шага — гиперпараметр
- Две стратегии перебора:
  - Зафиксировать η, подбирать N
  - Зафиксировать N, подбирать η

Стохастичний градієнтний бустинг = градієнтний бустинг + bagging



- Stochastic gradient boosting combines gradient boosting with bootstrap averaging (bagging)
- At each iteration the base classifier is trained on a fraction subsample of the available training data. The subsample is drawn without replacement. A typical value of subsample is 0.5
- The figure illustrates the effect of shrinkage and subsampling on the goodness-of-fit of the model
- We can clearly see that shrinkage outperforms no-shrinkage. Subsampling with shrinkage can further increase the accuracy of the model. Subsampling without shrinkage, on the other hand, does poorly

Градієнтний бустинг. Практичні особливості

- Базові алгоритми – дерева прийняття рішень
- Обмеження на глибину дерева: від 2 до 8
- Для зменшення перенавчання:
  - зменшення кроку
  - навчання на підвибірках (bagging)

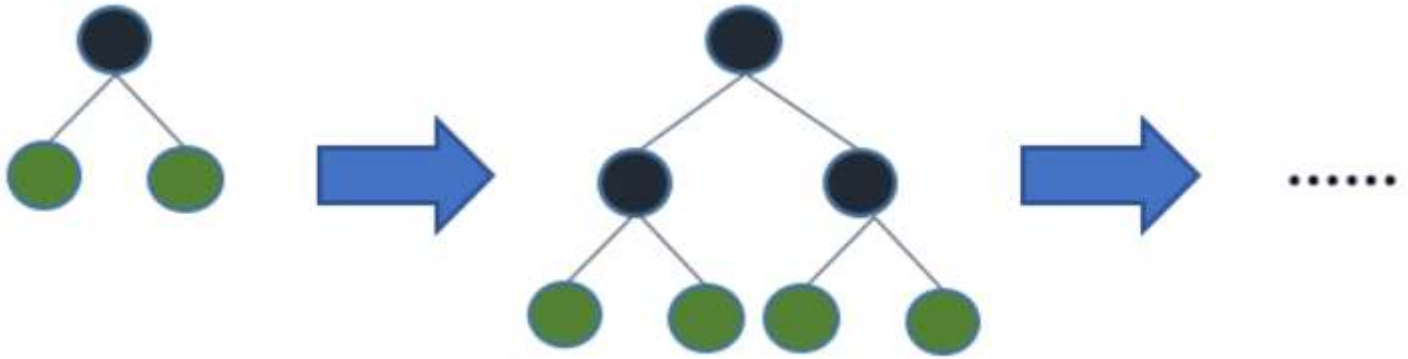
Градієнтний бустинг. Найкращі алгоритми

- AdaBoost (Adaptive Boosting) ==>

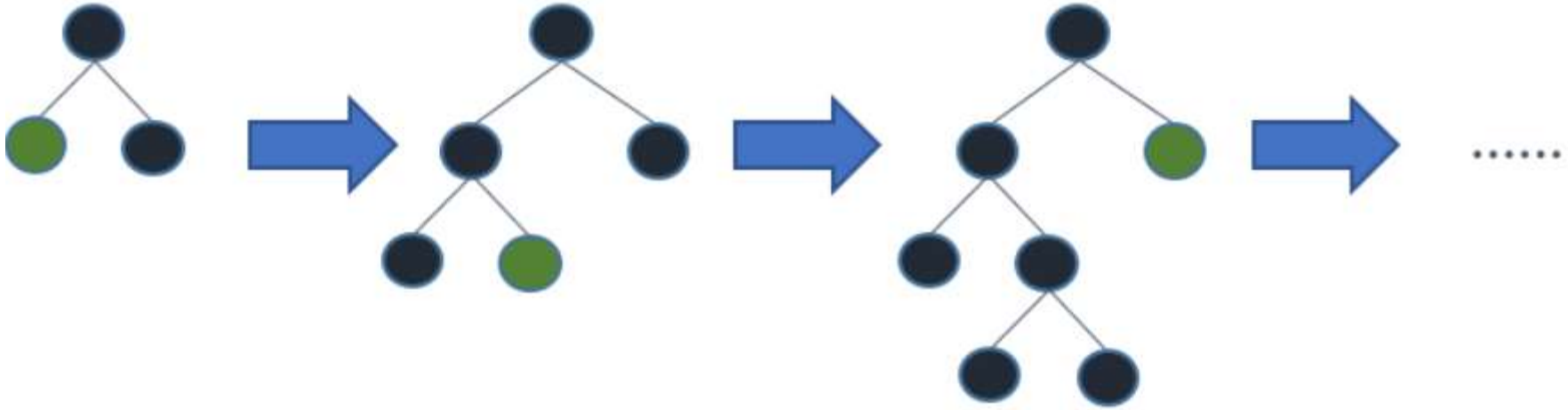
XGBoost (eXtreme Gradient Boosting, екстремальне градієнтне підсилювання) ==>

==> LightGBM (Light Gradient Boosted Machine)

Градієнтний бустинг. LightGBM vs XGBOOST



Level-wise tree growth



Leaf-wise tree growth

LightGBM (2017)

- Реалізація вводить дві ключові ідеї: GOSS і EFB
- Gradient Based One Side Sampling (GOSS, градієнтна одностороння вибірка) є модифікацією градієнтного бустинга, який фокусує увагу на тих навчальних прикладах, які призводять до більшого градієнту, в свою чергу, прискорюючи навчання і зменшуючи обчислювальну складність методу
- За допомогою GOSS виключається значна частка екземплярів даних з невеликими градієнтами, тому GOSS може дати досить точну оцінку інформаційного виграшу з набагато меншим розміром даних
- Exclusive Feature Bundling (EFB, об'єднання взаємовиключних ознак) — це підхід об'єднання розріджених (в основному нульових) взаємовиключних ознак, таких як категоріальні змінні вхідних даних, закодовані унітарною кодуванням. Таким чином, це тип автоматичного підбору ознак

А чи потрібно застосовувати ансамблі моделей?

- Відомий приз Netflix в 1 мільйон доларів, виграний за допомогою ансамблевих методів:

[A. Toescher, M. Jahrer and R. M. Bell, «The Bigchaos Solution to the Netflix Grand Prize», http://www.stat.osu.edu/~dmsl/GrandPrize2009_BPC_BigChaos.pdf]

- Netflix так і не вдалося впровадити модель-переможця через її складність, оскільки додаткова точність була не настільки великою як витрати на її реалізацію