

Задача класифікації. Логістична регресія

Наш план на цю тему

- Завдання бінарної класифікації та модель логістичної регресії: основні принципи
- Оцінка якості в завданні бінарної класифікації (precision, recall, F1-score, Area Under ROC)
- Повний розв'язок задачі із використанням логістичної регресії
- Задача мультикласової класифікації. Оцінка якості

Задачі класифікації

- це задачі, де кожен вхідний сигнал (або об'єкт) повинен бути віднесений до певної категорії (також називається міткою або класом).

При цьому якщо класів лише 2, то задача називається “задача бінарної класифікації”. А якщо класів більше 2 - задача багатокласової класифікації.

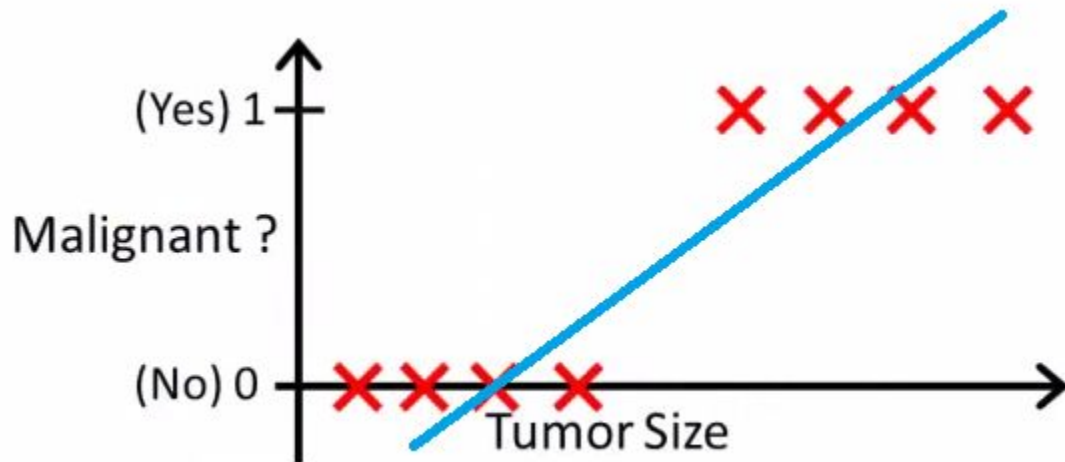
Приклади задач бінарної класифікації

Цільова змінна тут (y) тут приймає лише 2 значення: True/False, або 0 і 1.

	1 або Позитивний (Positive) клас	0 або Негативний (Negative) клас
Електронний лист	Спам	Не спам
Пухлина	Злоякісна	Доброякісний
Транзакція	Шахрайська	Не шахрайська

Чому нам тут не підходить лінійна регресія?

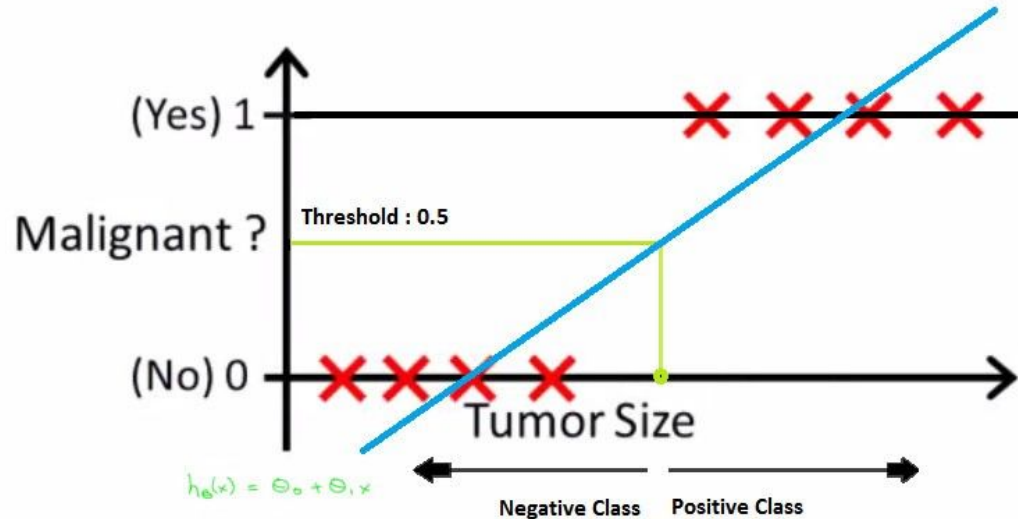
Припустимо, ми побудували таку лінійну регресію.



Чому нам тут не підходить лінійна регресія?

І встановили поріг між позитивними та негативними класами.

Виглядає непогано, але якщо з'являться викиди справа....



Чому нам тут не підходить лінійна регресія?

Наш поріг зсувається вправо і ми вже значно помиляємося в класифікації.



Отже, нам потрібна інша функція гіпотези*. Яка?

Функція гіпотези - це і є наша модель, яка апроксимує те, що ми хочемо побудувати.

Що ми прогнозуємо в бін. класифікації

Фактично ми робимо оцінку імовірності настання цільової події: клієнт клікне на посилання/емейл є спамом і тд. Тобто ми робимо оцінку такої функції:

$$p(y \mid x)$$

Що ми прогнозуємо в бін. класифікації

Фактично ми робимо оцінку імовірності настання цільової події: клієнт клікне на посилання/емейл є спамом і тд. Тобто ми робимо оцінку такої функції:

$$p(y \mid x)$$

І виявляється таку функцію класно апроксимує так звана “сигмоїда”.

Знайомтесь, сигмоїда! Вона ж логістична функція

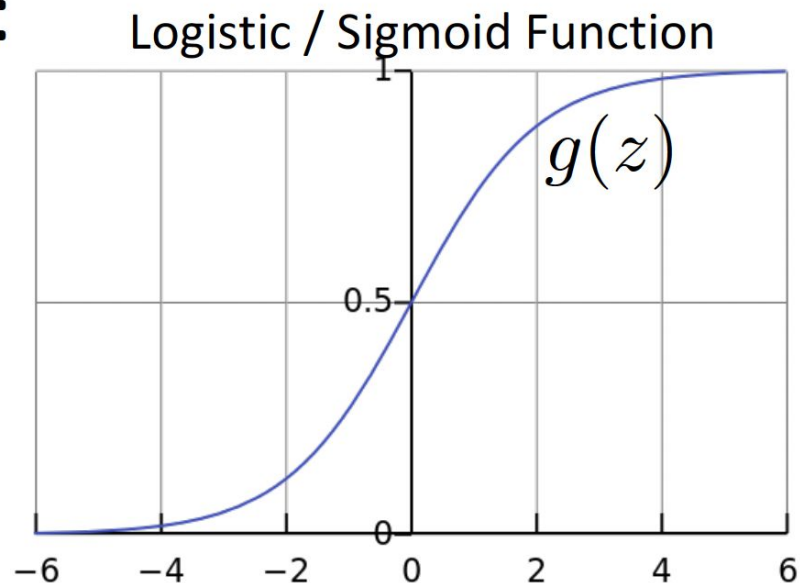
Logistic regression model:

$$h_{\theta}(x) = g(\theta^T x)$$

$$g(z) = \frac{1}{1 + e^{-z}}$$

$$h_{\theta}(x) = \frac{1}{1 + e^{-\theta^T x}}$$

оцінка ймовірності, що $y=1$ при вхідному x



Лінійна комбінація ($\theta^T x$):

$\theta^T x$ — це скалярний добуток двох векторів, що обчислюється наступним чином:

$$\theta^T x = \theta_0 x_0 + \theta_1 x_1 + \theta_2 x_2 + \dots + \theta_n x_n$$

Де:

- $\theta_0, \theta_1, \theta_2, \dots, \theta_n$ — компоненти вектора параметрів θ .
- $x_0, x_1, x_2, \dots, x_n$ — компоненти вектора ознак x .

Приклад:

Якщо у нас є дві ознаки x_1 і x_2 , тоді:

- $\theta = [\theta_0, \theta_1, \theta_2]$
- $x = [1, x_1, x_2]$

Тоді:

$$\theta^T x = \theta_0 \cdot 1 + \theta_1 \cdot x_1 + \theta_2 \cdot x_2$$

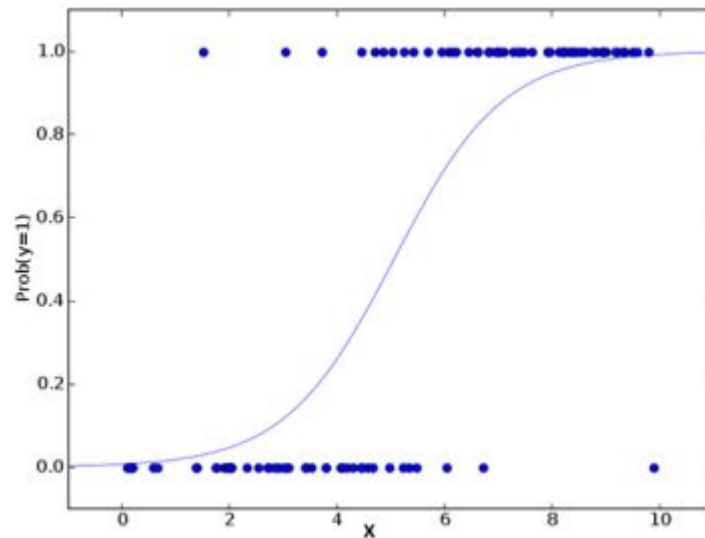
Ця лінійна комбінація параметрів і ознак передається в сигмоїдну функцію для обчислення ймовірності належності зразка до певного класу.

Знайомтесь, сигмоїда! Вона ж логістична функція

Знайм

$$h_{\theta}(x) = g(\theta^T x) = \frac{1}{1 + e^{-\theta^T x}},$$

$$g(z) = \frac{1}{1 + e^{-z}}$$



$h_{\theta}(x)$ - оцінка ймовірності, що $y=1$ при вхідному x

Приклад

Приклад з однією ознакою

Розглянемо випадок, коли у нас є одна ознака x .

Лінійна комбінація:

- Вектор параметрів: $\theta = [\theta_0, \theta_1]$
- Вектор ознак: $x = [1, x_1]$

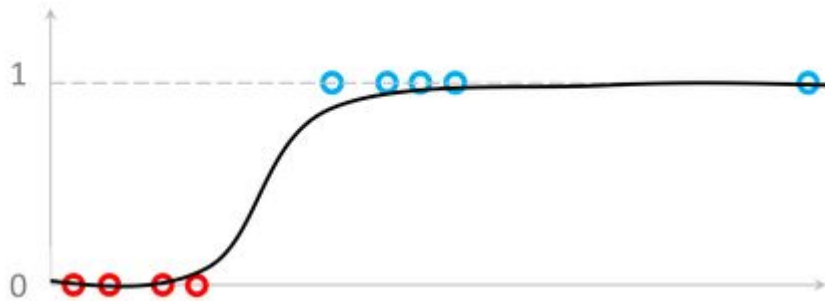
Тоді:

$$\theta^T x = \theta_0 \cdot 1 + \theta_1 \cdot x_1 = \theta_0 + \theta_1 x_1$$

Гіпотеза моделі:

Гіпотеза моделі логістичної регресії виглядає так:

$$h_{\theta}(x) = \frac{1}{1 + e^{-(\theta_0 + \theta_1 x_1)}}$$



Приклад

Припустимо, у нас є один зразок із ознакою $x_1 = 2$, справжня мітка класу $y = 1$, параметри моделі $\theta_0 = -3$ і $\theta_1 = 1$.

Гіпотеза моделі:

$$h_{\theta}(x) = \frac{1}{1+e^{-(-3+1 \cdot 2)}} = \frac{1}{1+e^{-(-3+2)}} = \frac{1}{1+e^{-(-1)}} = \frac{1}{1+e^1} \approx \frac{1}{1+2.718} \approx 0.2689$$



Значення функції гіпотези в задачі лог. регресії — ймовірність належності екземпляра до класу 1.

Функція витрат у задачі логістичної регресії

У лінійній регресії ми використовували наступну функцію витрат:

$$J(\theta) = \frac{1}{2m} \sum_{i=1}^m (h_{\theta}(x_i) - y_i)^2$$

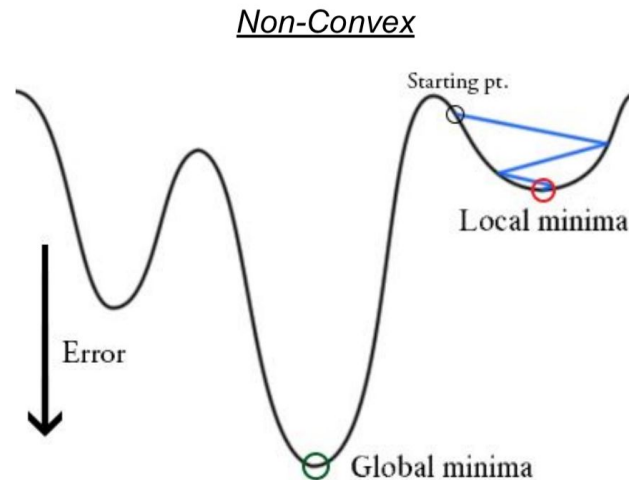
Функція витрат у задачі логістичної регресії

У лінійній регресії ми використовували наступну функцію витрат:

$$J(\theta) = \frac{1}{2m} \sum_{i=1}^m (h_{\theta}(x_i) - y_i)^2$$

Якщо ми спробуємо використати функцію витрат лінійної регресії в логістичній регресії, то це не буде ефективно, оскільки це буде **неопукла** функція з багатьма локальними мінімумами, в якій буде дуже важко мінімізувати витрати й знайти глобальний мінімум.

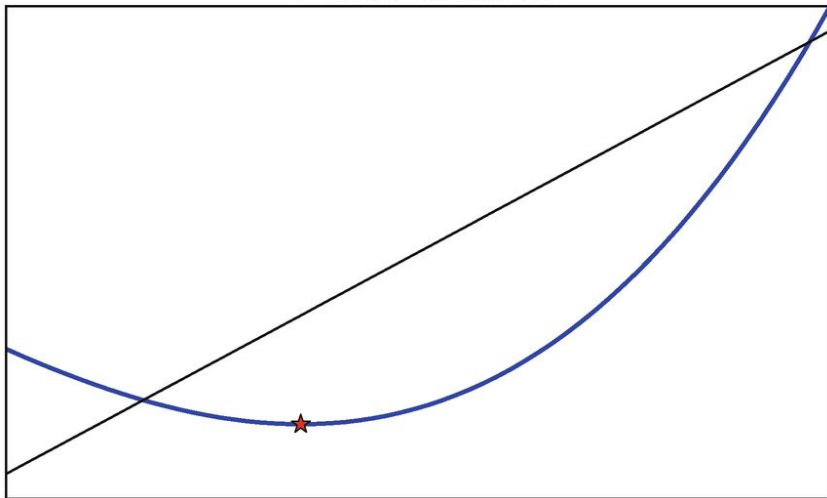
Тому функцію витрат ми теж будемо шукати іншу :)



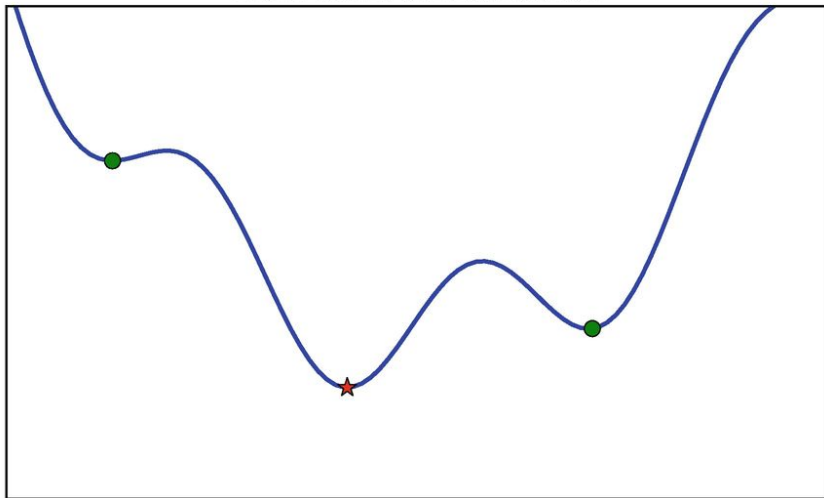
Різниця між опуклими та неопуклими функціями

Опукла функція — це така функція, яка завжди лежить нижче (або на одному рівні) від будь-якої прямої, що з'єднує будь-які дві точки на її графіку. Простіше кажучи, графік опуклої функції завжди "поглиблений" або "вигнутий" вгору, ніби чаша.

convex function



non-convex function



Просте пояснення

Уявімо, що у нас є функція $f(x)$. Функція $f(x)$ називається опуклою, якщо для будь-яких двох точок x_1 і x_2 на її графіку, пряма лінія, яка з'єднує точки $(x_1, f(x_1))$ і $(x_2, f(x_2))$, завжди знаходиться над графіком функції між цими двома точками. Це означає, що для будь-якого значення λ від 0 до 1 виконується наступна умова:

$$f(\lambda x_1 + (1 - \lambda)x_2) \leq \lambda f(x_1) + (1 - \lambda)f(x_2)$$

Візуальне уявлення

- Уявіть, що ви натягуєте гумку між двома точками на графіку функції. Якщо функція опукла, гумка завжди буде над або на одному рівні з графіком функції.
- Графік опуклої функції виглядає як "чаша" або "усмішка".

Приклад

Розглянемо приклад найпростішої опуклої функції: парабола $f(x) = x^2$.

- Якщо ви виберете будь-які дві точки на графіку цієї функції, наприклад $x_1 = -1$ і $x_2 = 1$, і з'єднаєте їх прямою лінією, ця лінія буде над графіком функції між цими точками.

Функція витрат в задачі логістичної регресії

Нам потрібно придумати функцію витрат для логістичної регресії.

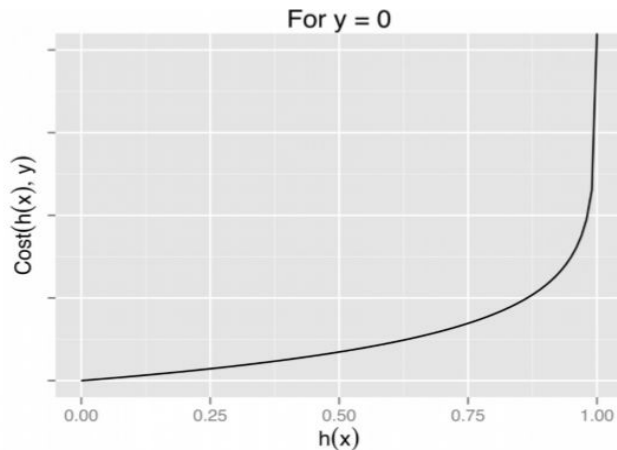
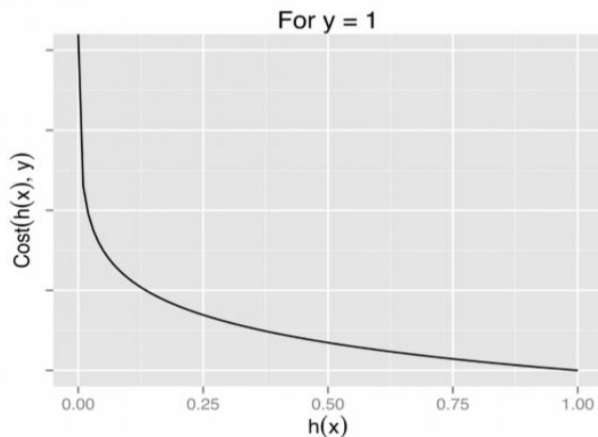
Розглянемо наступну форму функції витрат:

$$Cost(h_{\theta}(x), y) = \begin{cases} -\log(h_{\theta}(x)) & \text{if } y = 1 \\ -\log(1 - h_{\theta}(x)) & \text{if } y = 0 \end{cases}$$

Функція витрат у задачі логістичної регресії

Розглянемо наступну форму функції витрат:

$$Cost(h_{\theta}(x), y) = \begin{cases} -\log(h_{\theta}(x)) & \text{if } y = 1 \\ -\log(1 - h_{\theta}(x)) & \text{if } y = 0 \end{cases}$$



$Cost(h_{\theta}(x), y) = 0$ if $h_{\theta}(x) = y$

$Cost(h_{\theta}(x), y) \rightarrow \infty$ if $y = 0$ and $h_{\theta}(x) \rightarrow 1$

$Cost(h_{\theta}(x), y) \rightarrow \infty$ if $y = 1$ and $h_{\theta}(x) \rightarrow 0$

Функція витрат в задачі логістичної регресії

Розглянемо наступну форму функції витрат:

$$Cost(h_{\theta}(x), y) = \begin{cases} -\log(h_{\theta}(x)) & \text{if } y = 1 \\ -\log(1 - h_{\theta}(x)) & \text{if } y = 0 \end{cases}$$

Систему можна подати більш компактно наступним чином:

$$J(\theta) = -\frac{1}{m} \sum \left[y^{(i)} \log(h_{\theta}(x(i))) + (1 - y^{(i)}) \log(1 - h_{\theta}(x(i))) \right]$$

Градiєнтний спуск

Нагадаємо формулу градієнтного спуску. Ми оновлюємо до збіжності параметри за наступною формулою.

$$\theta_j := \theta_j - \alpha \frac{\partial L}{\partial \theta_j}$$

Нам потрібно знайти часткову похідну нашої нової функції витрат по θ_j .

Часткова похідна функції витрат по параметрам

Диференціювання наведене в “Математичне формулювання логістичної регресії.ipynb”, ви можете з ним ознайомитися самостійно. Фінальний результат наступний:

$$\frac{\partial L(\theta)}{\partial \theta_j} = \frac{1}{m} \sum_{i=1}^m [(h_{\theta}(x^{(i)}) - y^{(i)})x_j^{(i)}]$$

Така ж формула була отримана для лінійної регресії, але тут **інша функція гіпотези!**

Фінальна формула градієнтного спуску

Repeat until convergence {

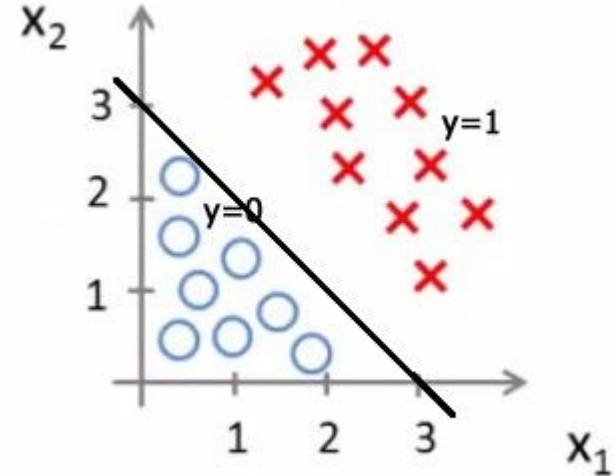
$$\theta_j := \theta_j + \alpha \sum_{i=1}^m \left(y^{(i)} - h_{\theta}(x^{(i)}) \right) x_j^{(i)} \quad (\text{for every } j).$$

}

Межа прийняття рішень / Decision boundary

Межа прийняття рішень допомагає розділити ймовірності на позитивний і негативний класи.

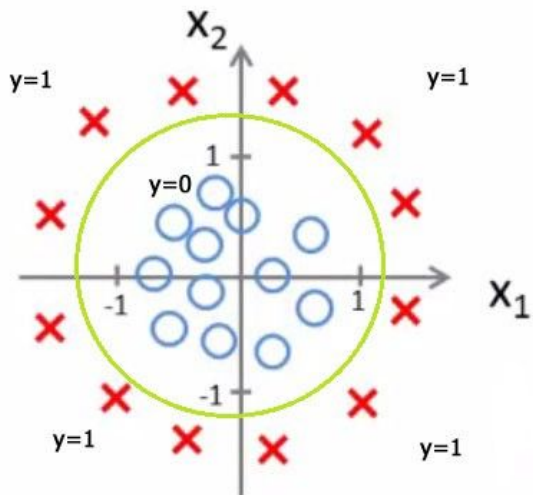
Межа лінійного рішення зображена на картинці.



For $y = 1$, Equation of line would be $x_1 + x_2 \geq 3$

For $y = 0$, Equation of line would be $x_1 + x_2 < 3$

Нелінійна межа прийняття рішення



For $y=1$, equation would be $x_1^2 + x_2^2 \geq 1$

For $y=0$, equation would be $x_1^2 + x_2^2 < 1$

Вимірювання якості моделі бінарної класифікації

Поріг класифікації / Classification Threshold

Логістична регресія повертає **ймовірність**. Ми можемо її використовувати «як є» (наприклад, «ймовірність того, що користувач натисне на цю рекламу, дорівнює 0.00023») або перетворити на бінарне значення (наприклад, «це лист є спамом»).

Поріг класифікації / Classification Threshold

Логістична регресія повертає **ймовірність**. Ми можемо її використовувати «як є» (наприклад, «ймовірність того, що користувач натисне на цю рекламу, дорівнює 0.00023») або перетворити на бінарне значення (наприклад, «це лист є спамом»).

Що, якщо електронне повідомлення має прогноз, що воно є спамом на 60%? Щоб порівняти значення логістичної регресії з бінарною категорією, потрібно визначити поріг класифікації (також називається **порогом прийняття рішення**). Значення вище цього порога вказує на «спам»; значення нижче означає «не спам». Пороги залежать від проблеми і, отже, є значеннями, які ви повинні налаштовувати. Вони не завжди повинні бути 0.5.

Налаштування порога прийняття рішень

«Налаштування» порога логістичної регресії відрізняється від налаштування гіперпараметрів, таких як швидкість навчання (learning rate).

Частина вибору порога — це оцінка того, наскільки ви постраждаєте за помилку. Наприклад, помилково позначити повідомлення, що не є спамом, як спам — це дуже погано. Однак помилково позначити спам-повідомлення як не спам — неприємно, але малоймовірно, що це кінець вашої роботи.

Матриця невідповідностей / Confusion matrix

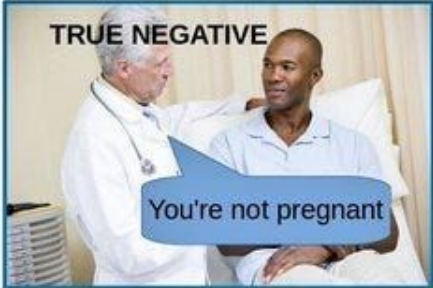



Матриця невідповідностей — це спосіб розбити об'єкти на чотири категорії залежно від комбінації справжньої відповіді та відповіді алгоритму (моделі).

Терміни:

- TP — справді позитивне рішення;
- TN — справді негативне рішення;
- FP — помилково позитивне рішення (Помилка першого роду);
- FN — помилково негативне рішення (Помилка другого роду).

n=165	Predicted: NO	Predicted: YES	
	Actual: NO	Actual: YES	
	TN = 50	FP = 10	60
	FN = 5	TP = 100	105
	55	110	

Матриця невідповідностей. Приклад

	$\hat{Y} = 0$ NEGATIVE	$\hat{Y} = 1$ POSITIVE
$Y = 0$ NOT PREGNANT	<p>TRUE NEGATIVE</p> 	<p>FALSE POSITIVE</p>  <p>TYPE 1 ERROR</p>
$Y = 1$ PREGNANT	<p>FALSE NEGATIVE</p>  <p>TYPE 2 ERROR</p>	<p>TRUE POSITIVE</p> 

Accuracy / Точність

$$\text{Accuracy} = (\text{TP} + \text{TN}) / (\text{TP} + \text{FP} + \text{FN} + \text{TN})$$

Точність – це частка правильних передбачень від усіх.

Коли використовувати?

Тільки коли дані збалансовані 50/50 класу 0 і класу 1.

Пастка використання ассигасу в класифікації

Наглядний приклад: Нерівномірно збалансовані класи

Уявімо, що у нас є задача класифікації, де потрібно визначити, чи є людина хворою чи здоровою. Припустимо, що в нашому наборі даних є 1000 прикладів, з яких 990 — здорові (клас 0), і лише 10 — хворі (клас 1).

Приклад 1: Модель завжди передбачає здоровий стан

Припустимо, що наша модель завжди передбачає, що людина здорова (всім прикладам присвоює клас 0). Тоді метрика точності (ассигасу) буде дуже високою.

- Правильні передбачення здорових людей: 990
- Неправильні передбачення хворих людей: 10
- Точність: $\frac{990}{1000} = 0.99$ або 99%

Чому точність (ассигасу) погана метрика в цьому випадку?

- Ігнорування важливих класів: Модель не виявляє жодного хворого, що є критичною помилкою в реальному світі.
- Помилкове відчуття надійності: Точність 99% створює враження, що модель працює дуже добре, тоді як насправді вона повністю ігнорує менш чисельний, але важливий клас.

Що ж робити?

Precision & Recall / Точність і повнота

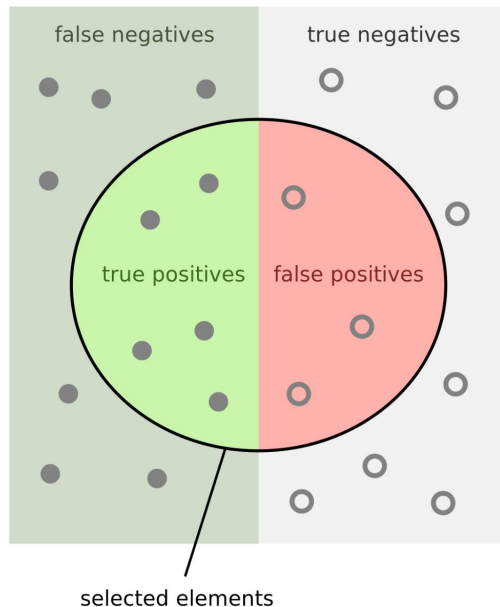
Точність (Precision) показує, яка частка об'єктів, виділених класифікатором як позитивні, дійсно є позитивними

$$Precision = \frac{TP}{TP + FP}$$

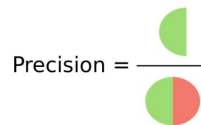
Повнота (Recall) показує, яка частина позитивних об'єктів була виділена класифікатором:

$$Recall = \frac{TP}{TP + FN}$$

https://en.wikipedia.org/wiki/Precision_and_recall

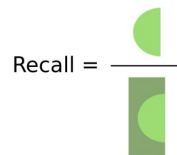


How many selected items are relevant?



Precision =

How many relevant items are selected?

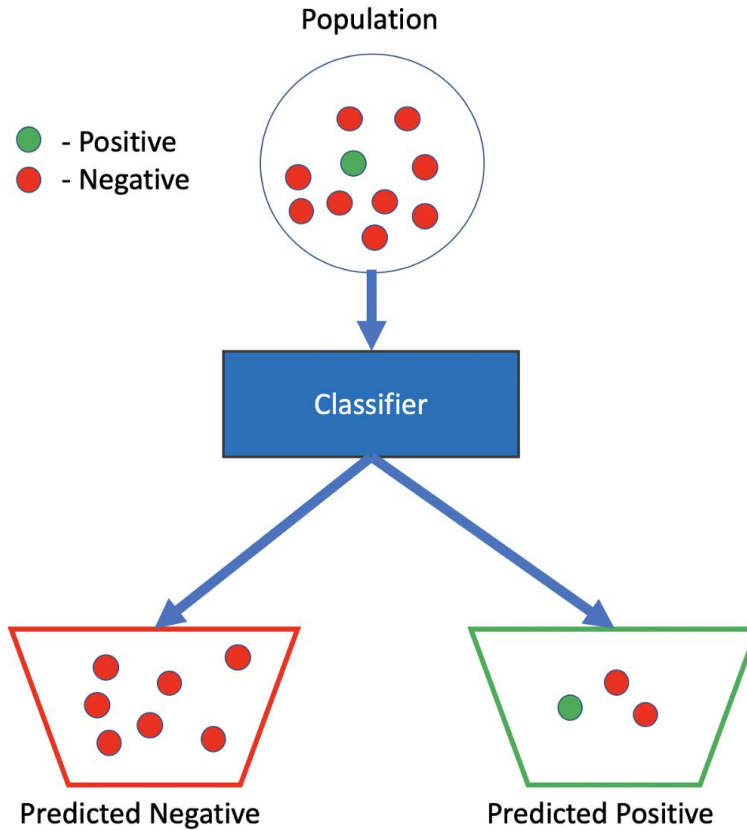


Recall =

Confusion matrix:

приклад розрахунку

і використання



		Real	
		Positive	Negative
Predicted	Positive	1	2
	Negative	0	7

$$\text{precision} = \frac{tp}{tp + fp} = \frac{1}{3} = 33\%$$

$$\text{recall} = \frac{tp}{tp + fn} = \frac{1}{1} = 100\%$$

$$\text{specificity} = \frac{tn}{tn + fp} = \frac{7}{9} = 78\%$$

$$\text{sensitivity} = \text{recall} = 100\%$$

Precision or recall?

Часто хочеться враховувати обидві метрики під час оцінювання.
Для цього існує F1-score - **гармонійне середнє** точності та повноти:

$$F_1 = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}$$

Середнє гармонійне має важливу властивість - **воно близьке до нуля, якщо хоча б один з аргументів близький до нуля**. Саме тому воно є кращим, ніж середнє арифметичне (якщо алгоритм буде відносити всі об'єкти до позитивного класу, то він матиме $\text{recall} = 1$ і precision більше за 0, а їхнє середнє арифметичне буде більше за $1/2$, що є неприпустимим).

F-beta score

Взагалі кажучи, F-score - метрика з параметром beta, який задає баланс precision і recall (що пріоритетніше).

$$F_{\beta} = (1 + \beta^2) \cdot \frac{\text{precision} \cdot \text{recall}}{(\beta^2 \cdot \text{precision}) + \text{recall}}$$

F_2 гостріше реагує на recall (тобто на частку хибнопозитивних відповідей), а $F_{0.5}$ чутливіший до точності (послаблює вплив хибнопозитивних відповідей).

TPR і FPR

True Positive Rate (TPR) є синонімом Recall (покриття) і тому визначається таким чином

$$TPR = \frac{TP}{TP + FN}$$

False Positive Rate (FPR) визначається таким чином:

$$FPR = \frac{FP}{FP + TN}$$

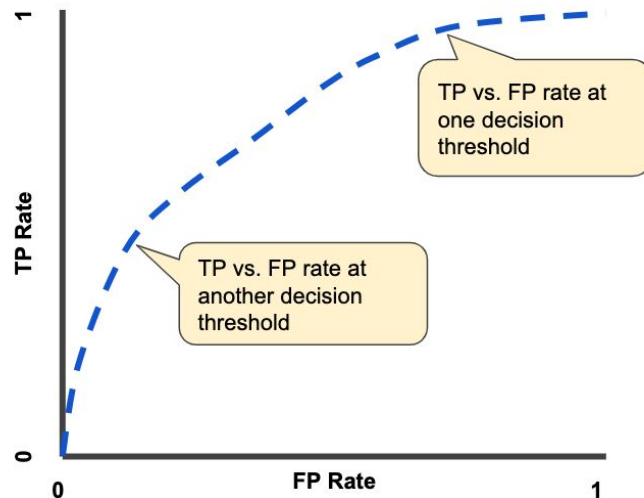
Точність, повнота та інші метрики з матриці невідповідностей

		Assigned class		
		Positive	Negative	
Real class	Positive	TP	FN	Recall $\frac{TP}{TP+FN}$
	Negative	FP	TN	False positive rate $\frac{FP}{TN+FP}$
		Precision $\frac{TP}{TP+FP}$	Specificity $\frac{TN}{TN+FN}$	Accuracy $\frac{TP+TN}{TP+TN+FP+FN}$

ROC-curve

Крива ROC (Receiver Operating Characteristic Curve, крива робочих характеристик приймача) - це графік, що показує ефективність моделі класифікації за всіх порогів класифікації.

Крива ROC відображає відношення **TPR** до **FPR** за різних **порогів класифікації**. При зниженні порога класифікації більша кількість елементів класифікується як позитивна, тим самим збільшуючи як помилкові, так і справжні позитивні результати.



Типова ROC-крива

Побудова ROC-кривої

1. **Обчислення ймовірностей:** Модель прогнозує ймовірності належності кожного зразка до позитивного класу.
2. **Встановлення порогів:** Вибираються різні порогові значення для перетворення ймовірностей в бінарні рішення (клас 1 або 0).
3. **Обчислення TPR та FPR:**

- **True Positive Rate (TPR)** або чутливість: доля правильно передбачених позитивних зразків серед усіх реальних позитивних зразків.

$$TPR = \frac{TP}{TP + FN}$$

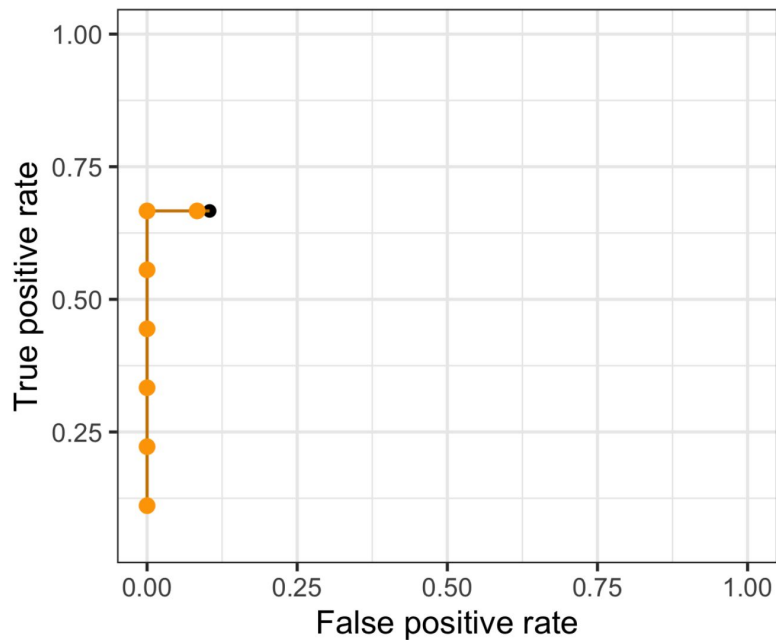
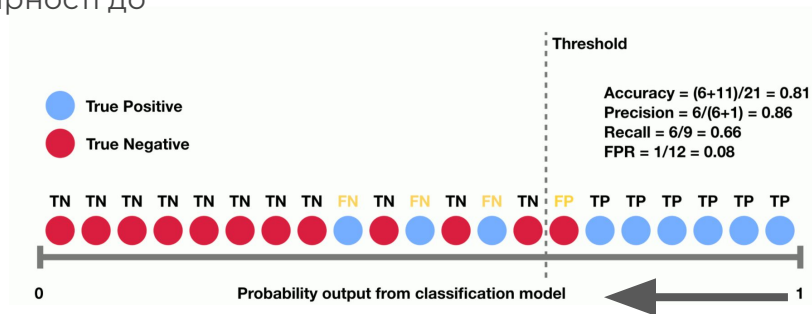
- **False Positive Rate (FPR):** доля хибно передбачених позитивних зразків серед усіх реальних негативних зразків.

$$FPR = \frac{FP}{FP + TN}$$

4. **Побудова кривої:** На осі X відкладається FPR, а на осі Y – TPR для кожного порогу. З'єднання точок утворює ROC-криву.

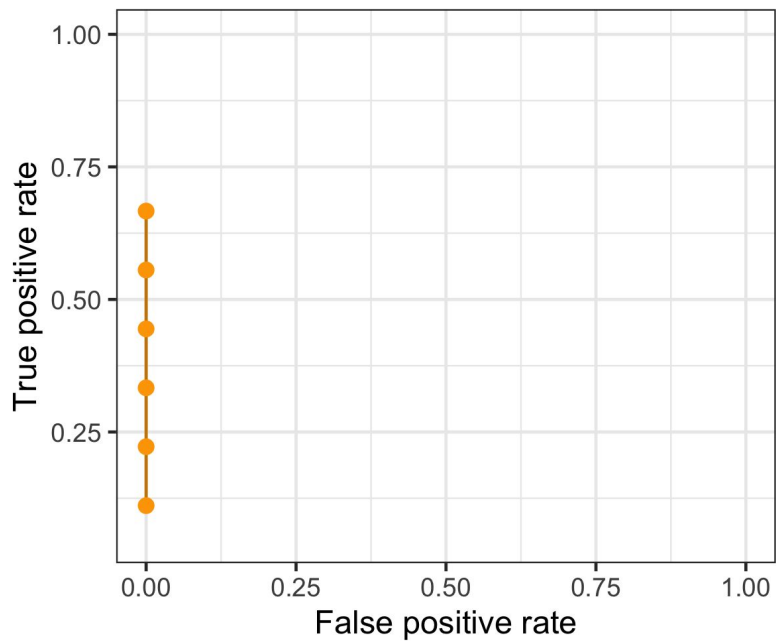
Ідемо від найвищої передбаченої імовірності до
найменшої змінюючи поріг

No	Class	Probability
1	P	0.99
2	P	0.85
3	P	0.77
4	P	0.72
5	P	0.60
6	P	0.56
7	N	0.53
8	N	0.49
9	P	0.45
10	N	0.44
11	P	0.38
12	N	0.31
13	P	0.27
14	N	0.18
15	N	0.17
16	N	0.14
17	N	0.11
18	N	0.11
19	N	0.10
20	N	0.05
21	N	0.04

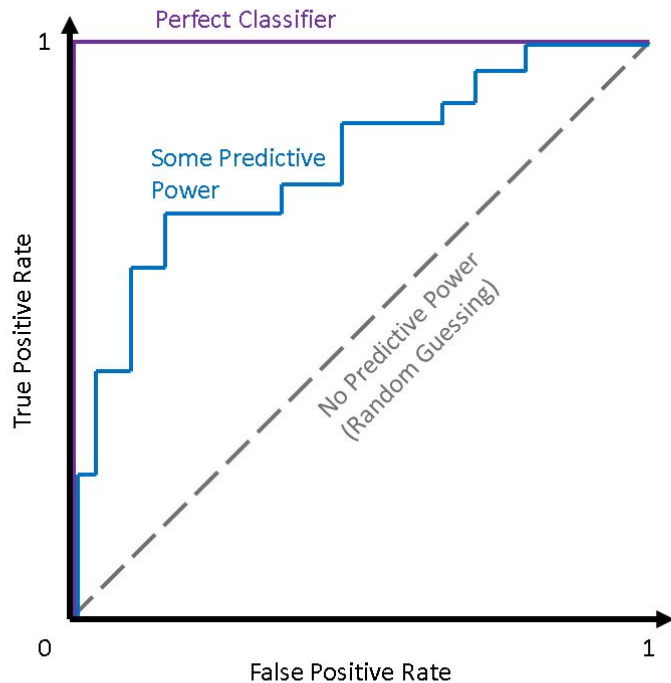


↓

No	Class	Probability
1	P	0.99
2	P	0.85
3	P	0.77
4	P	0.72
5	P	0.60
6	P	0.56
7	N	0.53
8	N	0.49
9	P	0.45
10	N	0.44
11	P	0.38
12	N	0.31
13	P	0.27
14	N	0.18
15	N	0.17
16	N	0.14
17	N	0.11
18	N	0.11
19	N	0.10
20	N	0.05
21	N	0.04



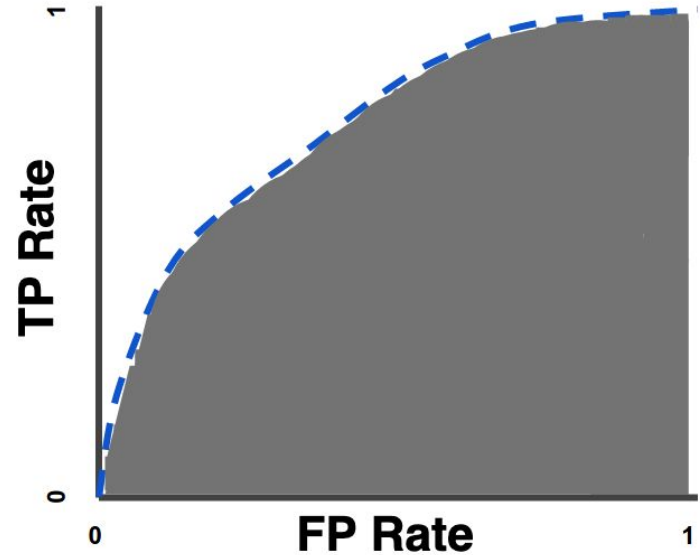
ROC крива для моделей з різною точністю



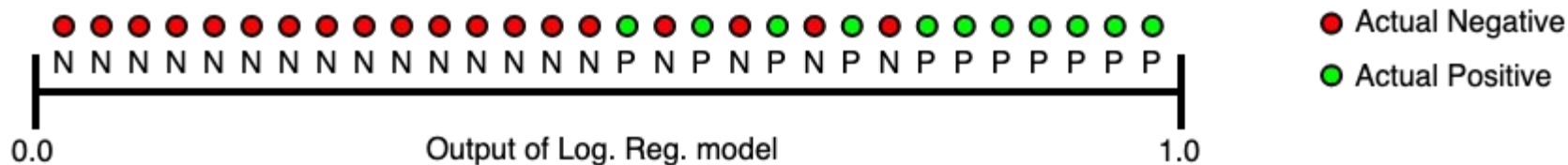
AUC: Area Under the ROC Curve

AUC ROC (Площа під кривою ROC) - метрика якості моделі класифікації.

AUC вимірює всю двовимірну область під всією кривою ROC (як в інтегральному численні) від (0,0) до (1,1).



AUC забезпечує сукупний показник якості моделі **за всіма можливими пороговими значеннями класифікації**. Один зі способів інтерпретації AUC - це ймовірність того, що модель оцінює випадковий позитивний приклад вище, ніж випадковий негативний приклад. Наприклад, у наступних прикладах, які розташовані зліва направо в порядку зростання прогнозів логістичної регресії:



Прогнози, відранговані в порядку зростання передбачених імовірностей належності до позитивного класу бінарним класифікатором.

AUC являє собою сукупну ймовірність того, що випадковий позитивний (зелений) приклад розташований праворуч від випадкового негативного (червоного) прикладу.

Значення AUC варіюється **від 0 до 1**. Модель, прогнози якої на 100% помилкові, має AUC 0.0; та, чиї прогнози вірні на 100%, має AUC 1.0.

Переваги AUC:

- AUC інваріантна до масштабу передбачення. Метрика вимірює, наскільки добре рангуються прогнози, а не їхні абсолютні значення.
- AUC не залежить від порога класифікації. Метрика вимірює якість прогнозів моделі незалежно від того, який поріг класифікації обрано.

Значення AUC варіюється **від 0 до 1**. Модель, прогнози якої на 100% помилкові, має AUC 0.0; та, чиї прогнози вірні на 100%, має AUC 1.0.

Переваги AUC:

- AUC інваріантна до масштабу передбачення. Метрика вимірює, наскільки добре ранжуються прогнози, а не їхні абсолютні значення.
- AUC не залежить від порога класифікації. Метрика вимірює якість прогнозів моделі незалежно від того, який поріг класифікації обрано.

Однак обидві ці причини мають **застереження**, які можуть обмежити корисність AUC у певних випадках використання:

- Інваріантність масштабу передбачення не завжди бажана. Наприклад, іноді нам дійсно потрібні добре **відкалібровані** ймовірнісні вихідні дані, AUC не повідомляє нам про це.
- Не завжди бажана й інваріантність до порога класифікації. У випадках, *коли існує велика різниця у вартості хибнонегативних і хибнопозитивних результатів*, може бути критично важливо мінімізувати один тип помилки класифікації. Наприклад, під час виявлення спаму електронною поштою ви, ймовірно, захочете встановити пріоритет мінімізації помилкових спрацьовувань (навіть якщо це призводить до значного збільшення кількості хибнонегативних результатів). AUC не є корисним показником для цього типу оптимізації.

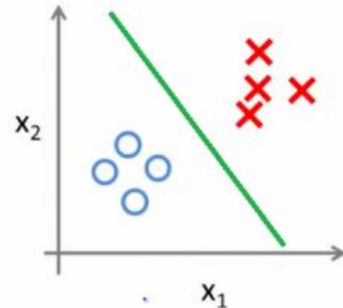
Мультикласова класифікація

Binary vs Multiclass classifications

Бінарна класифікація:

- + У наборі даних присутні тільки 2 різних класи.
- + Потрібна тільки 1 модель класифікатора.
- + Матрицю невідповідностей легко отримати і зрозуміти.
- + **Приклад:** класифікація електронної пошти на спам/не спам

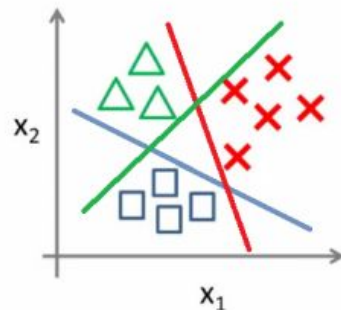
Binary classification:



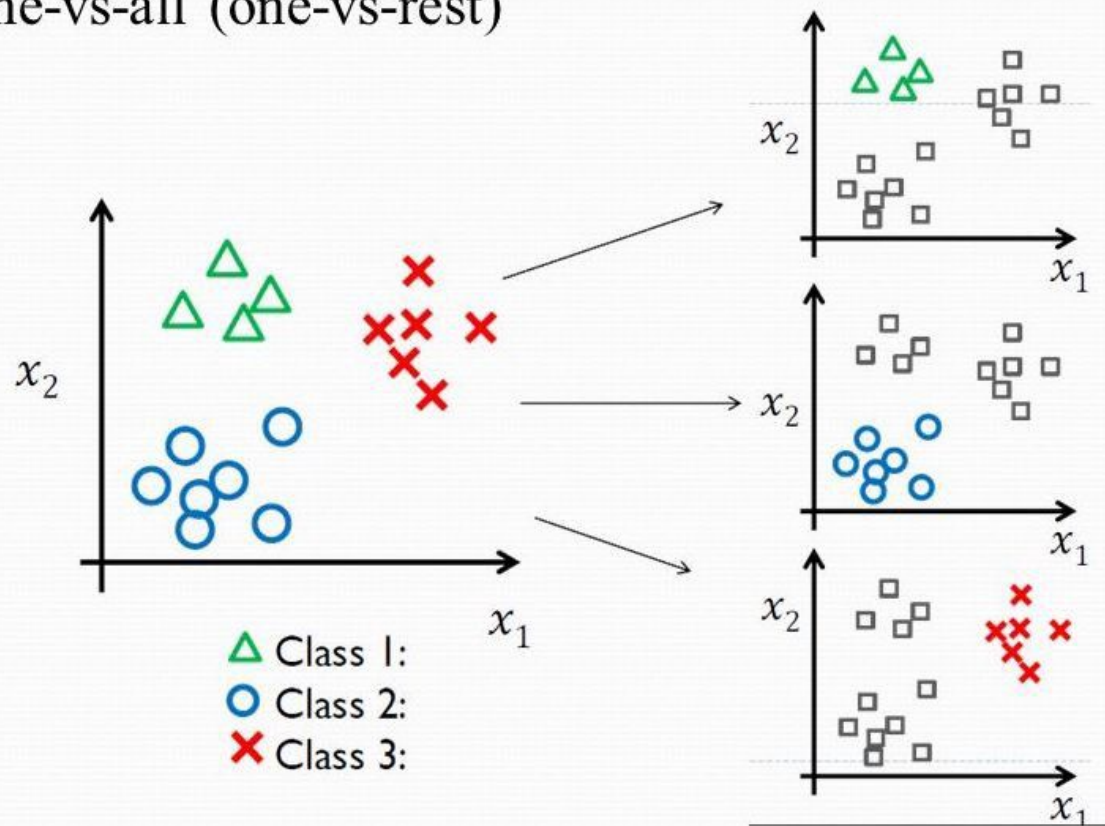
Мультикласова класифікація:

- + У наборі даних є **3 і більше** класів.
- + Кількість моделей класифікаторів залежить від застосовуваної нами методики класифікації.
- + Один проти всіх / One-vs-all: для N-класової класифікації використовуємо N бінарних класифікаторів.
- + Один проти одного / One-vs-one: для N-класової класифікації використовуємо $N * (N-1) / 2$ бінарних класифікаторів
- + Матрицю відповідностей легко отримати, але складно зрозуміти.
- + **Приклад:** класифікація інтересів клієнта на побутові товари, спортивні або вузько-спеціалізовані

Multi-class classification:



One-vs-all (one-vs-rest)



One-vs-Rest мультикласова класифікація. Приклад

Сформуємо три навчальні набори даних із новими значеннями цільової змінної.

Main Dataset

Features			Classes
x1	x2	x3	G
x4	x5	x6	B
x7	x8	x9	R
x10	x11	x12	G
x13	x14	x15	B
x16	x17	x18	R

Class 1 :- Green Class 2 :- Blue Class 3 :- Red

Training Dataset 1
Class :- Green

Features			Green
x1	x2	x3	+1
x4	x5	x6	-1
x7	x8	x9	-1
x10	x11	x12	+1
x13	x14	x15	-1
x16	x17	x18	-1

Training Dataset 2
Class :- Blue

Features			Blue
x1	x2	x3	-1
x4	x5	x6	+1
x7	x8	x9	-1
x10	x11	x12	-1
x13	x14	x15	+1
x16	x17	x18	-1

Training Dataset 3
Class :- Red

Features			Red
x1	x2	x3	-1
x4	x5	x6	-1
x7	x8	x9	+1
x10	x11	x12	-1
x13	x14	x15	-1
x16	x17	x18	+1

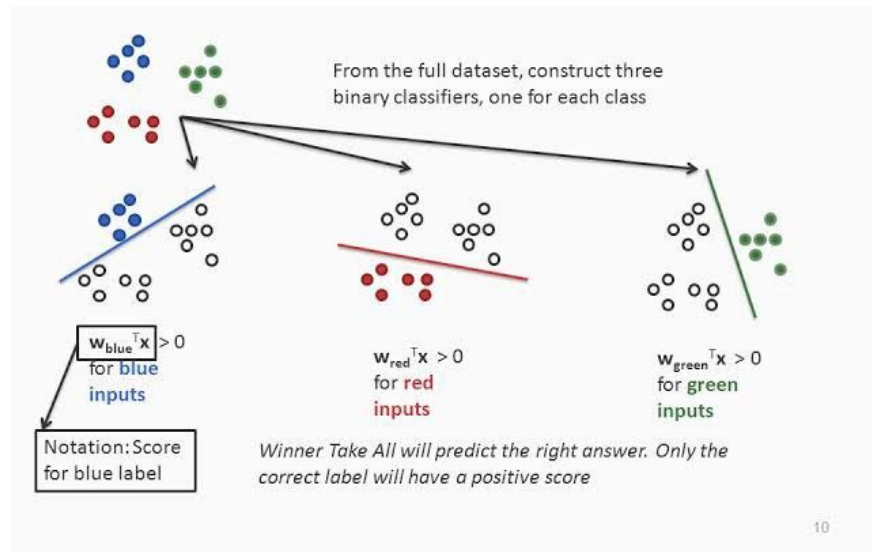
Після тренування трьох моделей - беремо той клас, для приналежності до якого найвища ймовірність

Є три значення тестових функцій y_1 , y_2 і y_3 відповідно.

Ми передали тестові дані в моделі класифікаторів. Ми отримали результат у

$$p(\text{Green})=0.9$$
$$p(\text{Blue})=0.4$$
$$p(\text{Red})=0.1$$

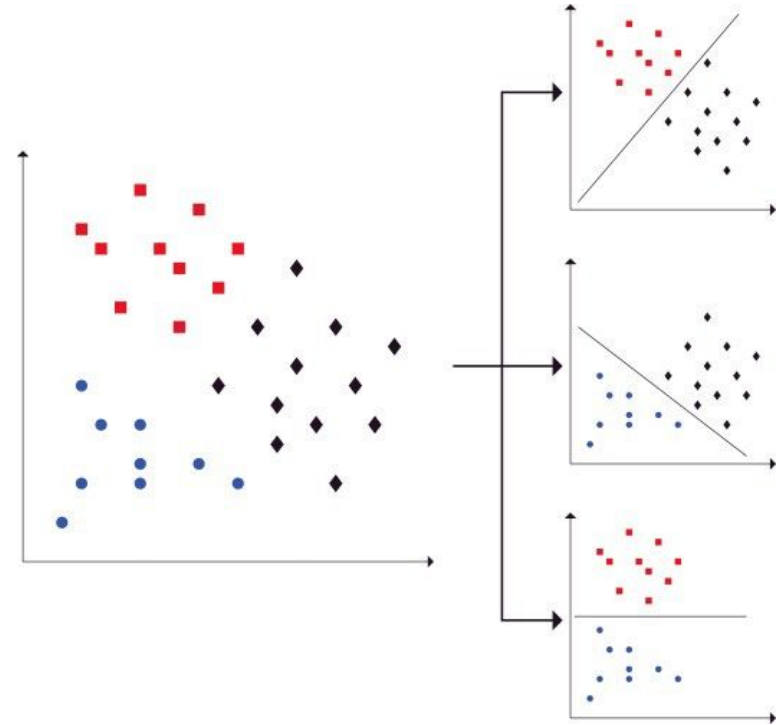
Отже, ми можемо сказати, що наші тестові дані належать до класу Green.



One vs one (OVO)

У мультикласовій класифікації «Один проти одного» ми розбиваємо первинний набір даних на один набір даних двійкової класифікації для кожної пари класів.

Кожен двійковий класифікатор передбачає одну мітку класу. Коли ми вводимо тестові дані в класифікатор, **результатом буде той клас, який «переміг» у максимальній кількості бінарних класифікацій.**



Метрики якості в мультикласовій класифікації

Micro-average показник Precision і Recall розраховується на основі кількості істинно позитивних (TP), істинно негативних (TN), хибнопозитивних (FP) і хибнонегативних (FN) передбачень моделі.

Macro-average оцінка Precision і Recall розраховується як середнє арифметичне оцінок Precision і Recall окремих класів.

Macro-average значення F1-score розраховується як середнє арифметичне F1-score окремих класів.

Метрики якості в мультикласовій класифікації. Приклад

Class 1: Urgent

	true urgent	true not
system urgent	8	11
system not	8	340

$$\text{precision} = \frac{8}{8+11} = .42$$

Class 2: Normal

	true normal	true not
system normal	60	55
system not	40	212

$$\text{precision} = \frac{60}{60+55} = .52$$

Class 3: Spam

	true spam	true not
system spam	200	33
system not	51	83

$$\text{precision} = \frac{200}{200+33} = .86$$

Pooled

	true yes	true no
system yes	268	99
system no	99	635

$$\text{microaverage precision} = \frac{268}{268+99} = .73$$

$$\text{macroaverage precision} = \frac{.42+.52+.86}{3} = .60$$

Метрики якості в мультикласовій класифікації.

Використовуйте оцінку **micro-average**, коли необхідно однаково зважити кожен екземпляр або прогноз.

Використовуйте оцінку **macro-average**, коли всі класи потрібно обробляти однаково, щоб оцінити загальну продуктивність класифікатора щодо міток класів, які найчастіше зустрічаються.

Використовуйте зважену (weighted) оцінку **macro-average у разі дисбалансу класів** (різна кількість екземплярів, пов'язаних із різними мітками класів). Середньозважене макро-середнє значення розраховується шляхом зважування оцінки кожної мітки класу за кількістю істинних екземплярів під час обчислення середнього.

Ще більше про метрики оцінки задачі багатокласової класифікації

<https://www.evidentlyai.com/classification-metrics/multi-class-metrics>

Sklearn API

https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html

https://scikit-learn.org/stable/modules/generated/sklearn.metrics.classification_report.html?highlight=classification%20report#sklearn.metrics.classification_report

```
#  
# Average is assigned micro  
#  
precisionScore_sklearn_microavg = precision_score(y_test,  
y_pred, average='micro')  
#  
# Average is assigned macro  
#  
precisionScore_sklearn_macroavg = precision_score(y_test,  
y_pred, average='macro')  
..
```