

## **Заняття 7.**

### **Інші типи регресій**

# План заняття

- + Розподіл набору даних на піднабори train, validation, test та для чого це потрібно
- + Масштабування ознак
- + Модель багатовимірної лінійної регресії. Побудова за допомогою sklearn
- + Інтерпретація коефіцієнтів лінійної регресії.
- + Проблеми high bias (underfit) та high variance (overfit)
- + Поняття регуляризації
- + Гребінна (ridge) регресія
- + Регресія методом лассо
- + Регресія “еластична мережа”
- + Поліноміальна регресія

# Розбиття набору даних на train/eval/test

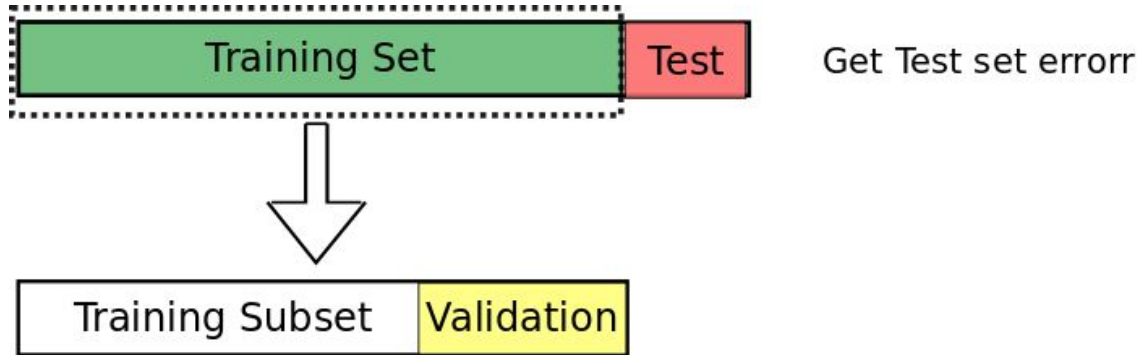
Ми навчили модель успішно передбачати на навчальних даних, чи буде вона працювати так само добре на нових даних, яких вона не бачила? Це гарне питання, на яке нам потрібно дати відповідь, проводячи моделювання.

# Розбиття набору даних на train/eval/test

**Train:** для навчання моделі

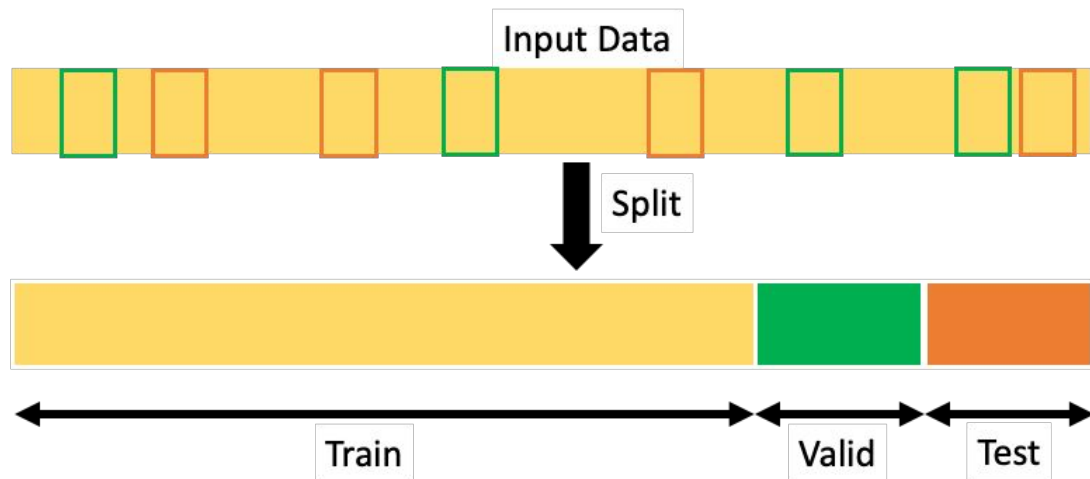
**Validation/Evaluation/Dev Set:** налаштування гіперпараметрів, вибір найкращої моделі

**Test:** для перевірки якості фінальної моделі

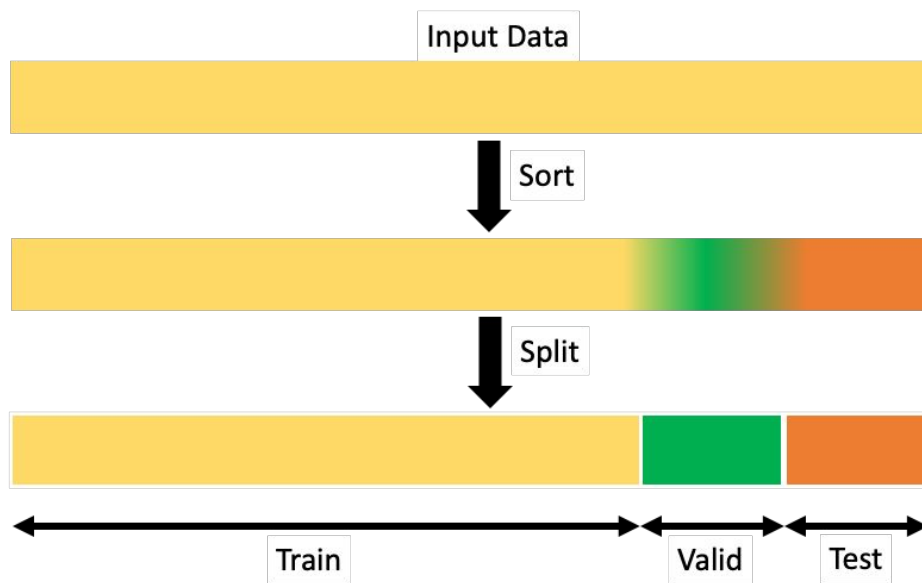


## Зазвичай ми вибираємо випадкові дані

Якщо забути перемішати дані, можемо внести упередженість у модель.



Але якщо є залежність  
від часу, то спочатку сортуємо  
за часом, потім розбиваємо



# Розбиття набору даних на train/eval/test

Кількість екземплярів у навчанні  $< 100\,000$ .

Працює принцип 80/20:

Розбиваємо набір даних на train/test у співвідношенні 80/20

Розбиваємо full train на train subset/validation також у співвідношенні 80/20

Також можна ділити train/test у співвідношенні 70/30 або ділити train subset/val/test у співвідношенні 60/20/20 або в пропорції 80/10/10.

Вибір варіанту розбиття можна здійснювати, виходячи з цільової метрики на train set.

# Розбиття набору даних на навчальну/валідацію/тест

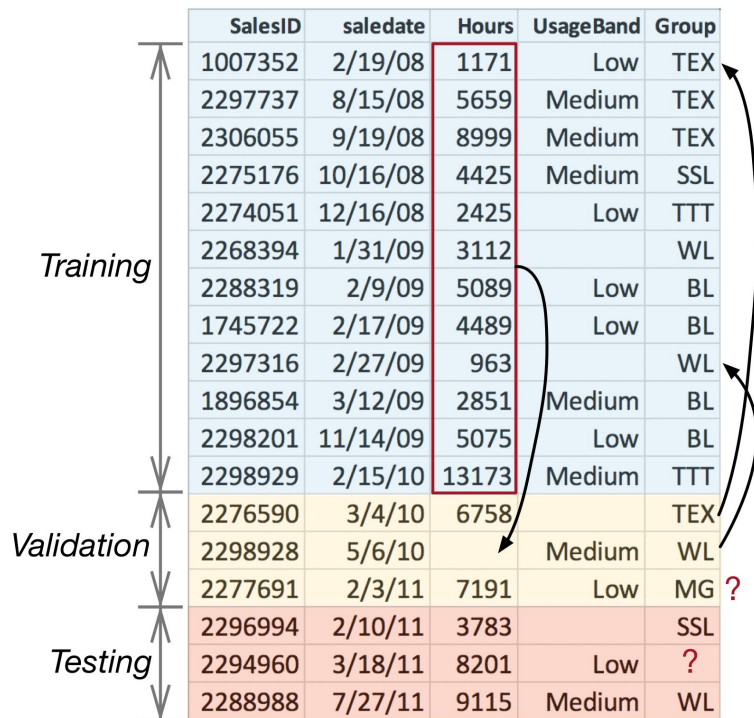
Кількість екземплярів у навчанні значно перевищує 100 000 (починаючи з 1 млн).

Розбиваємо набір даних на навчальну підмножину / валідацію / тест у співвідношенні 99/1/1, оскільки і так буде достатньо даних для тестування, а з іншого боку ми забезпечуємо максимальне різноманіття прикладів у навчальному наборі даних.



# Що враховувати при розбитті на підвибірки

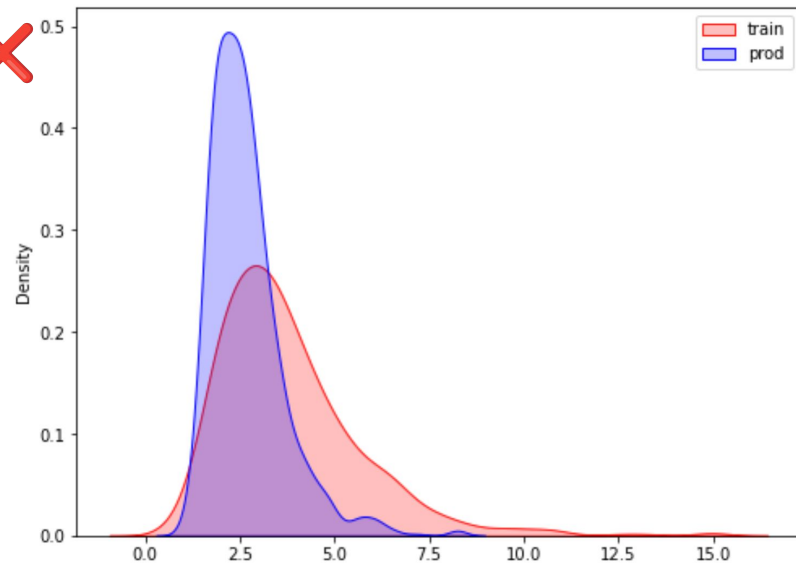
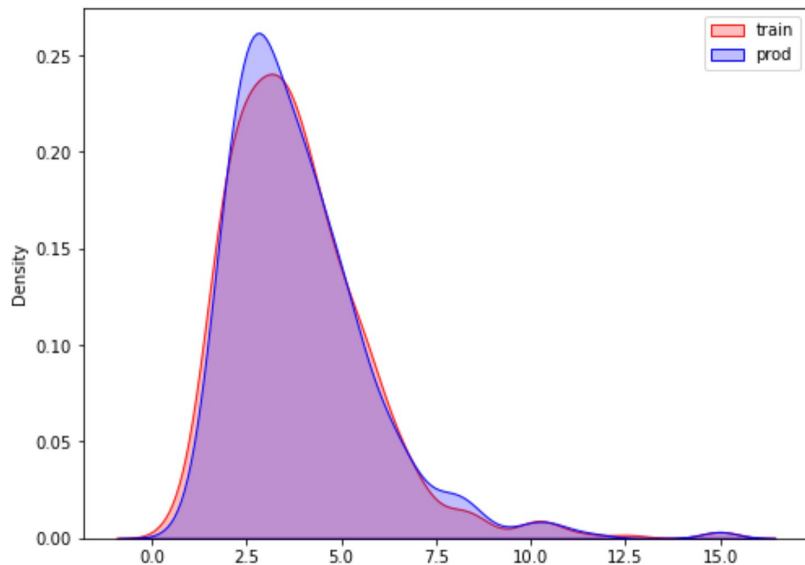
1. Переконатися, що піднабори train subset / val / test мають однаковий розподіл даних, передусім цільової змінної, але також можна перевірити й ознаки
2. Переконатися, що розбиття є відтворюваним (ми зафіксували спосіб розбиття, наприклад, зафіксувавши random seed)
3. Розбиваємо за індексами (а потім забираємо за ними дані)
4. Використовуючи метод попередньої обробки даних (Scaler, OneHotEncoder, ...) його необхідно фітити на train subset, а на validation/test просто застосовувати готовий до використання.



SalesID	saledate	Hours	UsageBand	Group
1007352	2/19/08	1171	Low	TEX
2297737	8/15/08	5659	Medium	TEX
2306055	9/19/08	8999	Medium	TEX
2275176	10/16/08	4425	Medium	SSL
2274051	12/16/08	2425	Low	TTT
2268394	1/31/09	3112		WL
2288319	2/9/09	5089	Low	BL
1745722	2/17/09	4489	Low	BL
2297316	2/27/09	963		WL
1896854	3/12/09	2851	Medium	BL
2298201	11/14/09	5075	Low	BL
2298929	2/15/10	13173	Medium	TTT
2276590	3/4/10	6758		TEX
2298928	5/6/10		Medium	WL
2277691	2/3/11	7191	Low	MG ?
2296994	2/10/11	3783		SSL
2294960	3/18/11	8201	Low	?
2288988	7/27/11	9115	Medium	WL

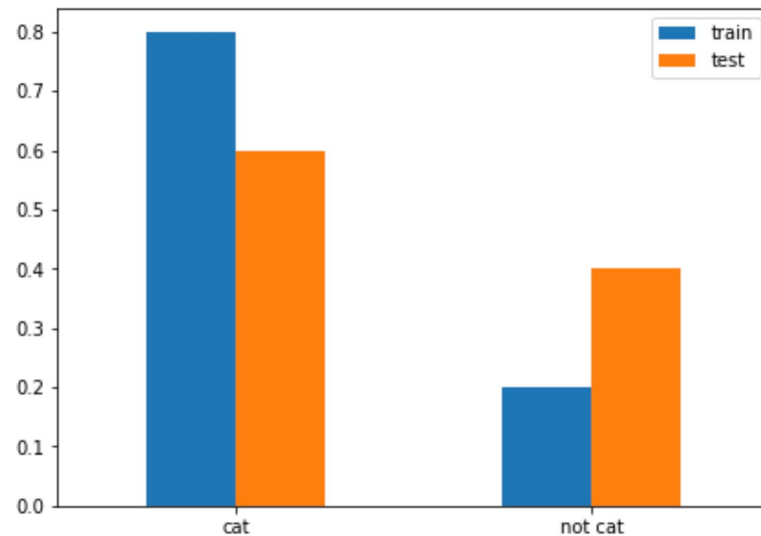
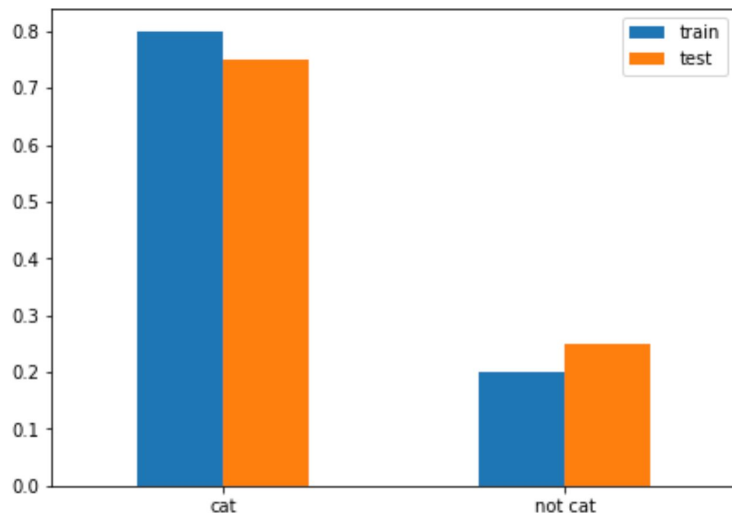
# Що означає однаковий розподіл даних

Для числової змінної



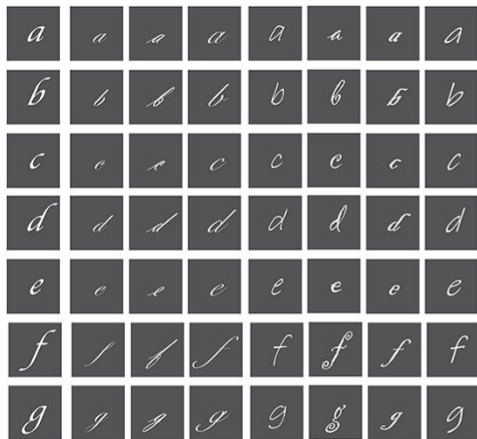
# Що означає однаковий розподіл даних

Для бінарної змінної



## Що означає однаковий розподіл даних

- В випадку картинок



## Training data

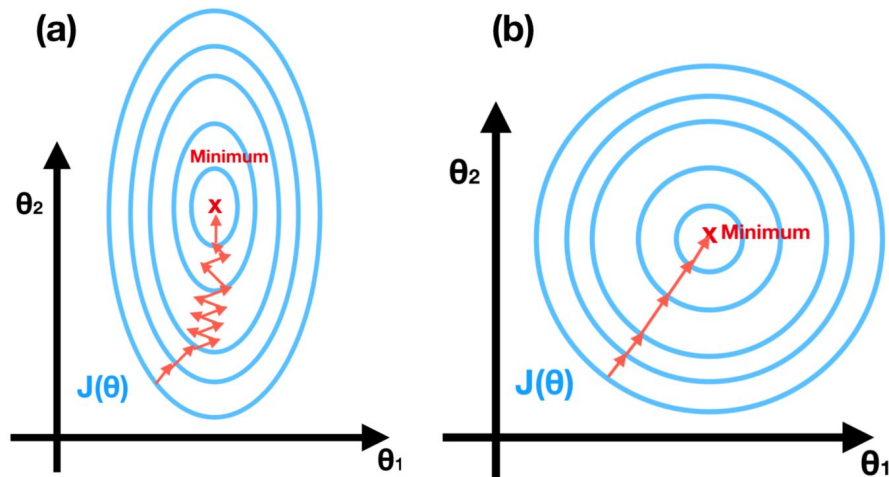


## Production data

# Масштабування ознак (Масштабування ознак)

При різному діапазоні кожної ознаки контури функції втрат можуть бути дуже тонкими, що зробить градієнтний спуск надзвичайно повільним, див. Рисунок (a).

Результат після масштабування функцій показаний на рисунку (b).



# 3 способи масштабувати ознаки

# Мінімаксне масштабування / від мінімуму до максимуму

$$x' = \frac{x - \min(x)}{\max(x) - \min(x)}$$

- + Перетворює кожен знак в діапазон [0,1].
- + Найпростіший і інтуїтивно зрозумілий спосіб.
- + Зберігає форму розподілу і чутливий до викидів.

# Стандартизація

$$x' = \frac{x - \bar{x}}{\sigma}$$

Цей метод масштабує функцію, видаляючи середнє значення і ділячи на стандартне відхилення. Він приводить розподіл до стандартного нормального з центром в 0 зі стандартним відхиленням 1.

Деякі алгоритми машинного навчання (SVM) передбачають, що ознаки знаходяться у цьому діапазоні.

З викидами дані будуть масштабовані до невеликого інтервалу.



# Робастне масштабування / Стійке масштабування

$$X = \frac{x - Q_1(x)}{Q_3(x) - Q_1(x)}$$

Схоже на Min-Max, але замість мінімального/максимального значення беруться перший і третій кuartилі. Міжквартильний розмах робить цей метод стійким до викидів (ось і назва).

# Подивимося, як це працює на практиці

[Лекція 8. Поза лінійною регресією.ipynb]



# Множинна (багатовимірна) лінійна регресія

Коли є більше одного ознаки.

Features				Label	
A data	Size ( <i>feet</i> <sup>2</sup> )	Number of bedrooms	Number of floors	age of home (years)	Price(\$1000)
	2104	5	1	45	460
	1416	3	2	40	232
	1534	2	2	30	315
	...	...	...	...	...

$$h_{\theta}(x) = \theta_0 x_0 + \theta_1 x_1 + \theta_2 x_2 + \dots + \theta_n x_n$$

Parameters:  $\theta = \{\theta_0, \theta_1, \theta_2, \dots, \theta_n\}$

Features:  $x = \{x_0, x_1, x_2, \dots, x_n\}$

$$h_{\theta}(x) = \theta_0 + 2104\theta_1 + 5\theta_2 + \theta_3 + 45\theta_4 = 460$$

$$h_{\theta}(x) = \theta_0 + 1416\theta_1 + 3\theta_2 + 2\theta_3 + 40\theta_4 = 232$$

$$h_{\theta}(x) = \theta_0 + 1534\theta_1 + 2\theta_2 + 2\theta_3 + 30\theta_4 = 315$$

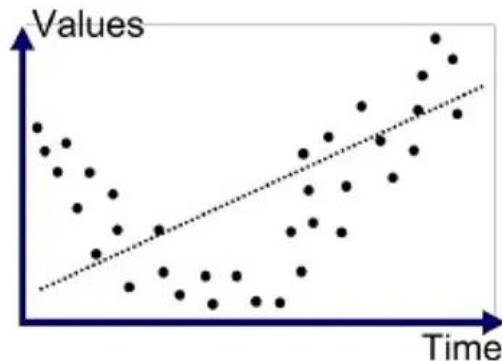
A data	Size ( <i>feet</i> <sup>2</sup> ) $x_1$	Number of bedrooms $x_2$	Number of floors $x_3$	age of home (years) $x_4$	Price(\$1000) $h_{\theta}(x) = y$
	2104	5	1	45	460
	1416	3	2	40	232
	1534	2	2	30	315
	...	...	...	...	...

Features (x) Label (  $h_{\theta}(x) = y$  )

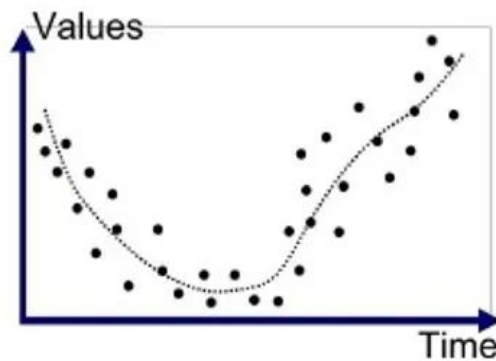
# Перенавчання /Недонавчання (Overfit / Underfit)

# НЕДОНАВЧАННЯ (Underfitting)

Ефект, коли модель є занадто простою для наших даних. Вона не може вловити складні залежності у даних, що призводить до низької точності як на тренувальних, так і на тестових даних. Іншими словами, модель має високу упередженість (bias).



Underfitted

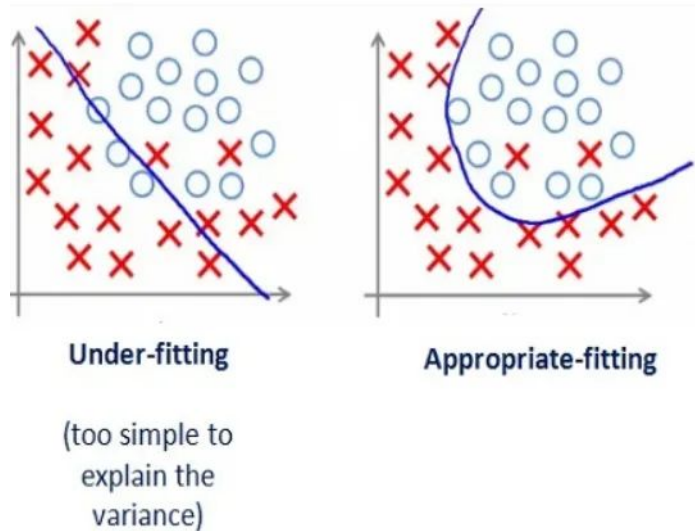


Good Fit/Robust

Порівняння недонавчання і нормального навчання для моделі прогнозування

# НЕДОНАВЧАННЯ (Underfitting)

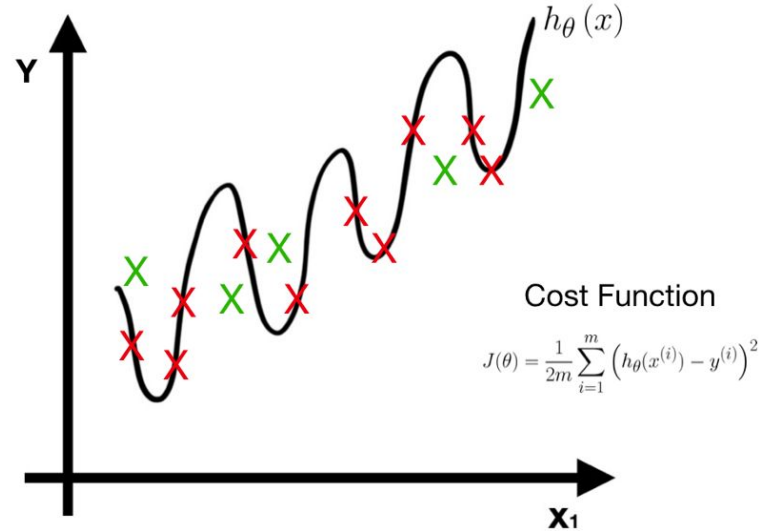
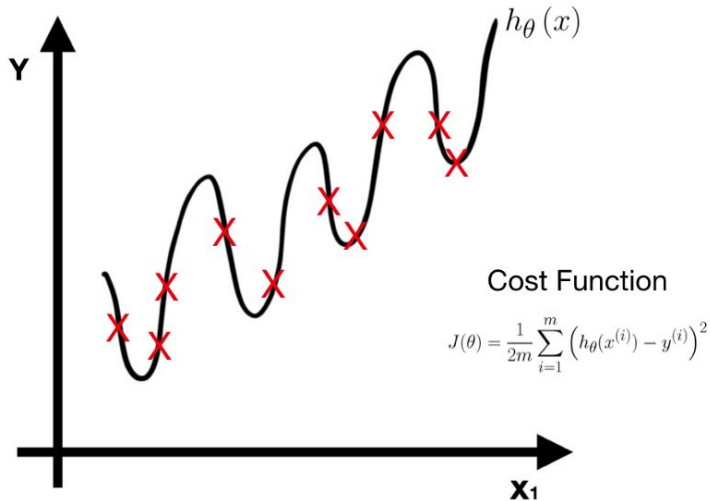
Ефект, коли модель є занадто простою для наших даних. Вона не може вловити складні залежності у даних, що призводить до низької точності як на тренувальних, так і на тестових даних. Іншими словами, модель має високу упередженість (bias).



Порівняння недонавчання і нормального навчання для моделі класифікації

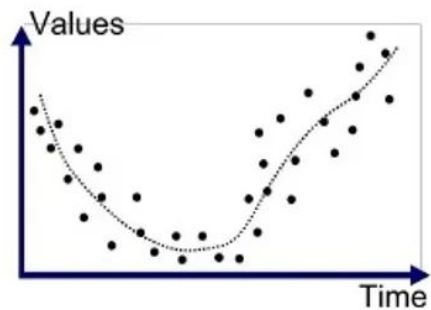
# ПЕРЕНавчання (Overfitting)

Коли помилка низька на тренувальних даних, але висока на валідаційних даних (яких модель не бачила). Тобто модель є занадто складною для наших даних. Вона вловлює навіть найменші шуми у тренувальних даних, що призводить до високої точності на тренувальних даних, але низької точності на тестових даних. Іншими словами, модель має високу варіативність або дисперсію (variance)

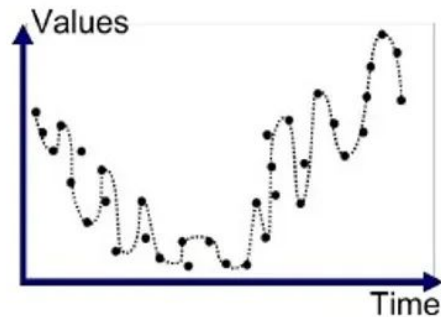




# PERЕнавчання: приклади



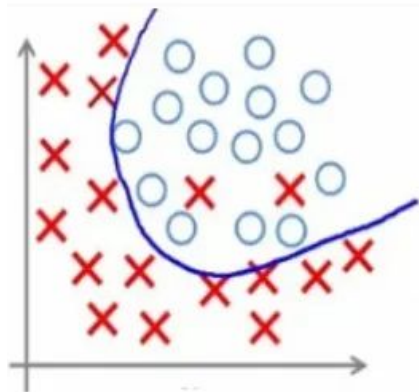
Good Fit/Robust



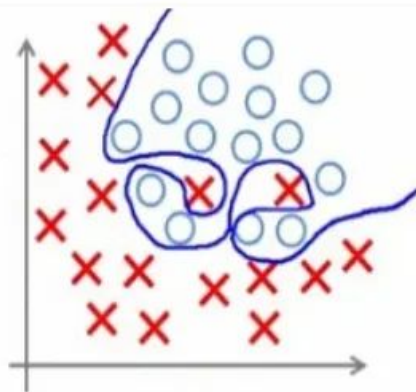
Overfitted

В задачі прогнозування

# ПЕРЕНавчання: приклади



Appropriate-fitting



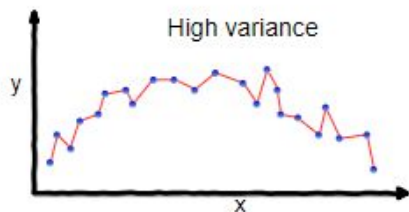
Over-fitting

(forcefitting -- too  
good to be true)

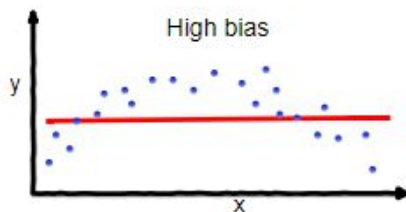
В задачі класифікації

# High variance vs High bias

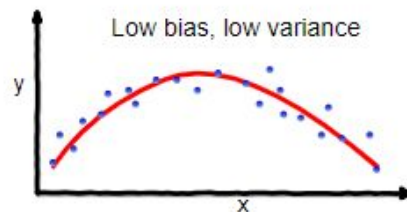
Проблема пошуку балансу між високою зміщеністю та високою варіативністю — одна з важливих задач у побудові ML-моделей і пошуку найкращої.



**overfitting**



**underfitting**

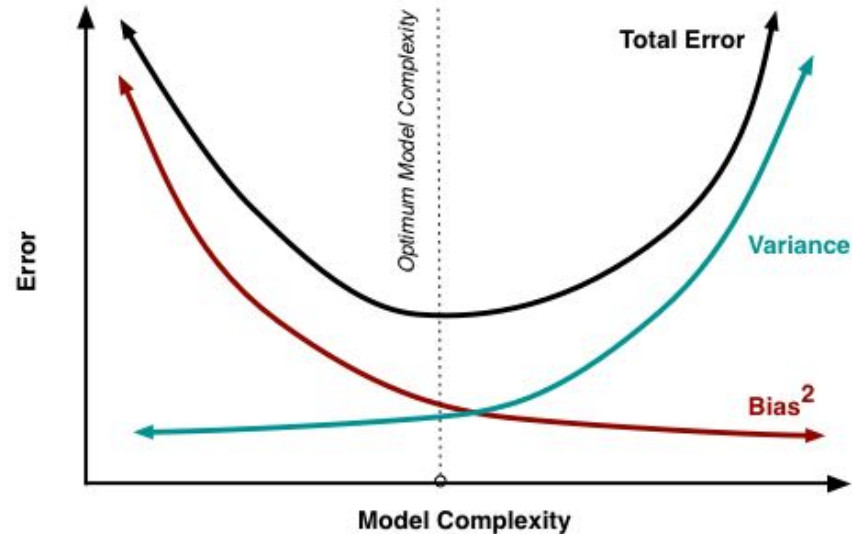


**Good balance**

# High variance vs High bias

Проблема пошуку балансу між високою зміщеністю та високою варіативністю — одна з важливих задач у побудові ML-моделей і пошуку найкращої.

І вирішується ця проблема за рахунок складності моделі: чим складніша модель, тим нижчий bias і вища variance.



**Розглянемо як це виглядає  
на прикладі використання  
ПОЛІНОМІАЛЬНОЇ РЕГРЕСІЇ**

# Поліноміальна регресія

Суть: як додаткові ознаки ми додаємо степені вихідних ознак.

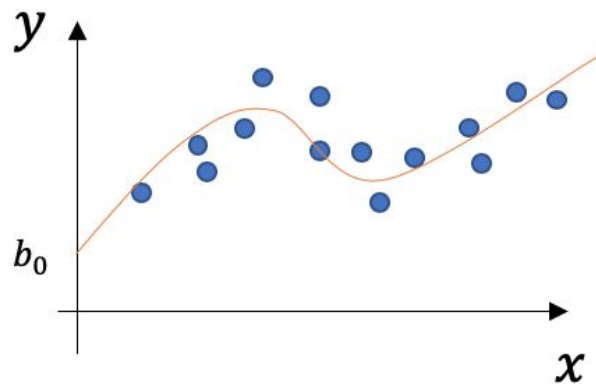
$$y = b_0 + b_1x + b_2x^2 + \dots$$

where,

$y$ : dependent variable

$b_*$ : coefficients

$x$ : Independent variable



## Типи поліномів

Linear —————  $ax + b = 0$

Quadratic —————  $ax^2 + bx + c = 0$

Cubic —  $ax^3 + bx^2 + cx + d = 0$

# Порівняння різних типів регресій

Simple  
Linear  
Regression

$$y = b_0 + b_1 x_1$$

Multiple  
Linear  
Regression

$$y = b_0 + b_1 x_1 + b_2 x_2 + \dots + b_n x_n$$

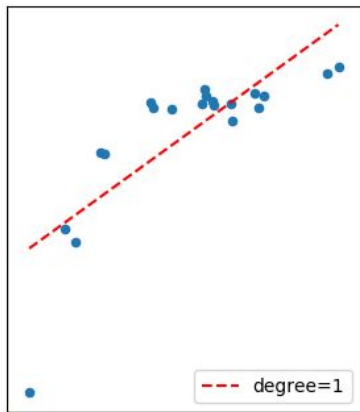
Polynomial  
Linear  
Regression

$$y = b_0 + b_1 x_1 + b_2 x_1^2 + \dots + b_n x_1^n$$

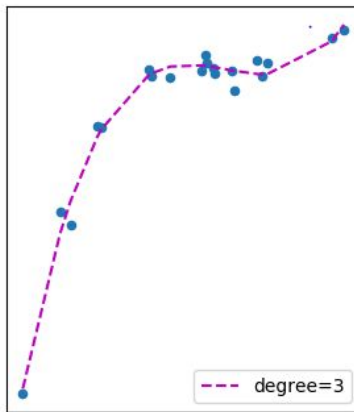


# Використання поліном регресії з високими степенями ознак призводить до перетренування

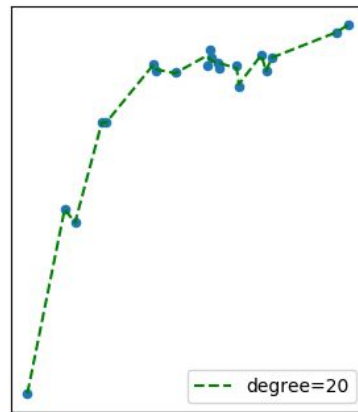
Але іноді використання степенів ознак може допомогти зменшити bias (що правда незрозуміло, як інтерпретувати такі ознаки).



Underfit  
High Bias  
Low Variance



Correct Fit  
Low Bias  
Low Variance



Overfit  
Low Bias  
High Variance

**Як визначити, що  
модель недо- або  
перевчилась?**

# Як визначити, що модель недо- або перевчилась?

2 способи, зазвичай використовуємо обидва.

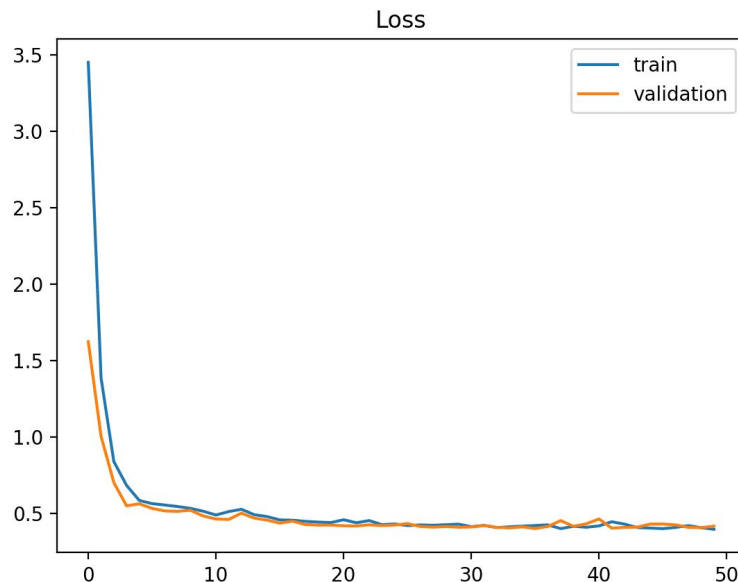
## 1. Оцінка на тренувальних і тестових даних:

- a. Якщо точність низька як на тренувальних, так і на тестових даних — це ознака недонавчання.
- b. Якщо точність висока на тренувальних, але низька на тестових даних — це ознака переонавчання.

## 2. Використання кривих навчання (Learning Curves):

- a. Криві навчання показують, як змінюється похибка моделі в залежності від кількості тренувальних епох. Якщо похибка на тренувальних і тестових даних залишається високою і не зменшується з додаванням більше епох, це ознака недонавчання.
- b. Якщо похибка на тренувальних даних низька, а на тестових висока і не зменшується з додаванням більше епох, це ознака перенавчання.

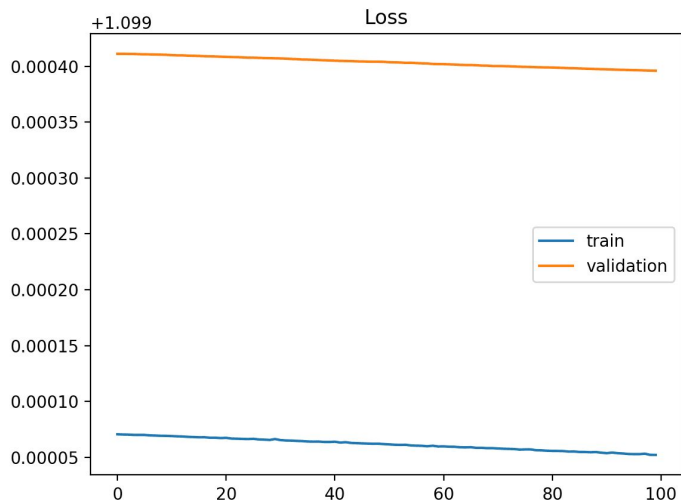
# Крива навчання для оцінки якості навчання моделі машинного навчання: ідеальний випадок



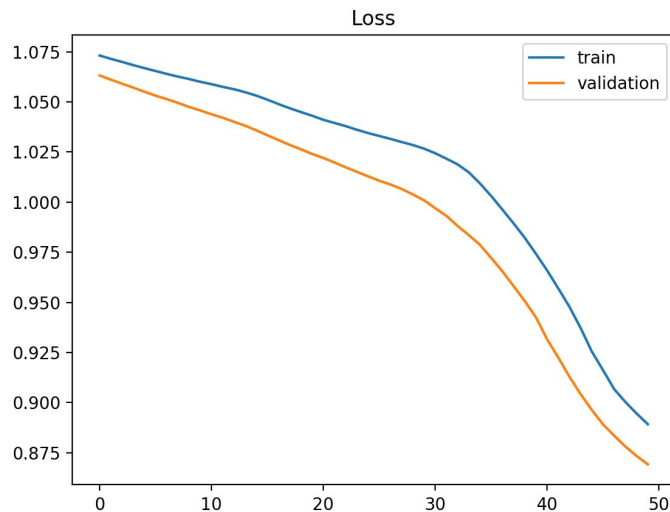
[Як використовувати криві навчання для діагностики результатів моделі машинного навчання](#)

# Крива навчання у разі недонавчання

Модель взагалі не навчилась



Модель не досягла плато, можливо, ще може покращитися

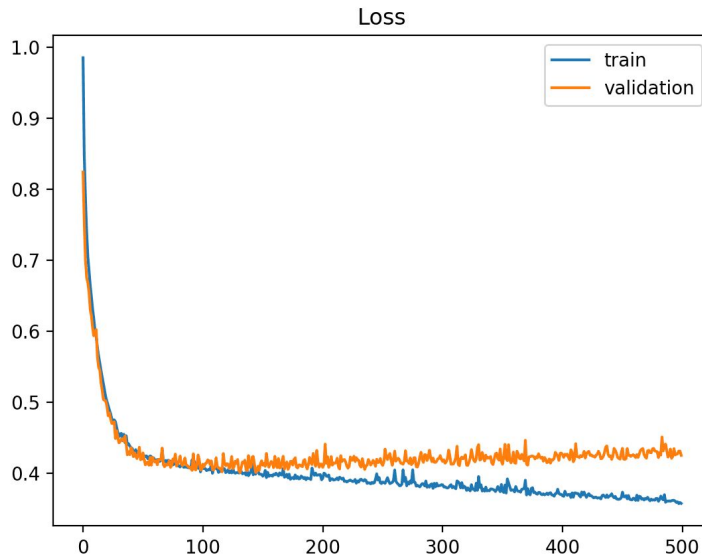


[Як використовувати криві навчання для діагностики результатів моделі машинного навчання](#)

# Крива навчання в разі перенавчання

Крива навчання показує перенавчання моделі, якщо:

- графік втрат на навчальних даних продовжує зменшуватись зі зростанням ітерацій
- графік втрат на валідаційних даних зменшується до певного моменту і знову починає зростати.



[Як використовувати криві навчання для діагностики результатів моделі машинного навчання](#)

# **Причини та методи боротьби з пере- та недонавчанням**

# Недонавчання: причини

1. **Надто проста модель:** Ми використали занадто простих алгоритмів або моделей, які не здатні вловити складні залежності у даних. Наприклад, лінійна регресія для задач, де залежність є нелінійною.
2. **Недостатня кількість характеристик (features):** Модель може не мати достатньо інформації для навчання, якщо у неї недостатньо характеристик або ці характеристики неінформативні.
3. **Недостатня кількість епох або ітерацій навчання (для алгоритмів на основі градієнтного спуску):** Якщо модель навчається недостатню кількість епох, вона може не встигнути підігнатися під дані.
4. **Неправильна обробка даних:** Недостатня або неправильна попередня обробка даних, як-то відсутність нормалізації або обробки категоріальних змінних, може призвести до недонавчання.



# Недонавчання: як вирішити проблему

- Навчити **більш потужну модель** (наприклад, ансамбль дерев прийняття рішень замість простої лінійної регресії, чи поліном. регресію замість звичайної)
- **Збільшити тривалість** навчання (кількість епох або ітерацій)
- Додати **більше ознак** шляхом виконання **feature engineering**, тоді моделі може стати легше виявляти закономірності в даних
- Розбити дані на сегменти і на кожному навчити просту модель

# Перенавчання: причини

1. **Надто складна модель:** Використання дуже складних моделей з великою кількістю параметрів, які здатні вловити навіть шуми у тренувальних даних. Наприклад, поліноміальна регресія високого ступеня.
2. **Занадто багато характеристик (features):** Якщо кількість характеристик значно перевищує кількість спостережень, модель може підігнатися під шуми.
3. **Надмірна кількість епох навчання:** Якщо модель навчається занадто довго, вона може підігнатися під всі деталі тренувальних даних, включаючи шуми.

# Перенавчання: причини

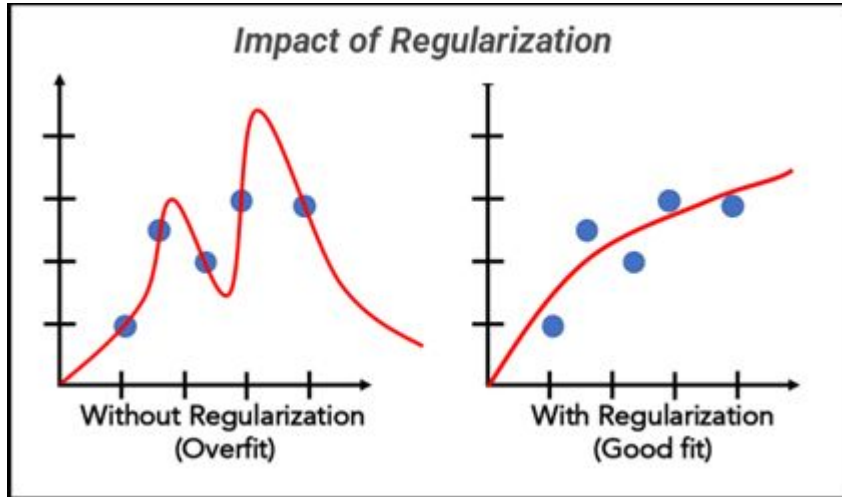
4. **Мала кількість тренувальних даних:** Модель може переонавчитися, якщо тренувальний набір даних занадто малий і не представляє всі можливі варіації даних.
5. **Відсутність регуляризації:** Регуляризація допомагає уникнути переонавчання, накладаючи штрафи на великі коефіцієнти моделі. Відсутність регуляризації може призвести до перенавчання.
6. **Погане розбиття даних на тренувальні та тестові набори:** Якщо дані для тестування не репрезентативні або змішані з тренувальними даними, це може призвести до неправильної оцінки продуктивності моделі і, як наслідок, переонавчання.

# Перенавчання: Як вирішити проблему

1. Використання **регуляризації** (L1, L2).
2. Використання **більшої кількості тренувальних даних**.
3. **Зменшення кількості ознак** на вході використовуючи методи feature selection.
4. **Зменшення складності моделі** (наприклад, зменшення ступеня поліному).
5. **Рання зупинка** (early stopping) з точки зору тривалості навчання в епохах.
6. Використання методів **зменшення вимірності** (наприклад, PCA).

# Регуляризація

Регуляризація – це метод у машинному навчанні, який дозволяє запобігти перенавчанню моделі, накладаючи певні обмеження або штрафи на параметри моделі. Це допомагає моделі узагальнювати інформацію і працювати краще на нових, невідомих даних.



# Регуляризація: чому потрібна

Перенавчання виникає, коли модель **занадто добре підігнана під тренувальні дані**, включаючи шуми і специфічні особливості, які не є загальними для всіх даних. Регуляризація допомагає уникнути цього, додаючи штрафи до функції втрат моделі, що змушує модель *знижувати складність* і узагальнювати краще.



# Регуляризація: як це працює

До функції витрат додаємо суму норми наших коефіцієнтів, помножені на певну константу — параметр регуляризації.

$$J(\theta) = \frac{1}{2m} \sum_{i=1}^m \left( h_{\theta}(x^{(i)}) - y^{(i)} \right)^2 + \lambda \sum_{j=1}^n \theta_j^2$$

**Regularization Term**

**Regularization Parameter**

start at  $\theta_1$

The diagram illustrates the regularization term in the cost function. A red box encloses the entire regularization term,  $\lambda \sum_{j=1}^n \theta_j^2$ , with the label "Regularization Term" in red text above it. Inside this box, a blue box highlights the regularization parameter  $\lambda$ , with a blue arrow pointing to it from the label "Regularization Parameter" in blue text below. A green arrow points from the label "start at  $\theta_1$ " to the summation  $\sum_{j=1}^n$ .

# Регуляризація: як це працює

До функції витрат додаємо суму норми наших коефіцієнтів, помножені на певну константу — параметр регуляризації.

$$J(\theta) = \frac{1}{2m} \sum_{i=1}^m \left( h_{\theta}(x^{(i)}) - y^{(i)} \right)^2 + \lambda \sum_{j=1}^n \theta_j^2$$

**Regularization Term**

**Regularization Parameter**

start at  $\theta_1$

The diagram highlights the regularization term  $\lambda \sum_{j=1}^n \theta_j^2$  with a red box. A blue box encloses the parameter  $\lambda$ , with a blue arrow pointing to it from the label 'Regularization Parameter' below. A green arrow points from the label 'start at  $\theta_1$ ' to the summation  $\sum_{j=1}^n$ .



# Регуляризація: як це працює

При мінімізації функції витрат, якщо параметр регуляризації високий, ми робимо дуже малими параметри, які потрапили до частини регуляризації. Тому що наше завдання — знайти мінімум функції витрат, яка залежить від параметрів моделі.

$$J(\theta) = \frac{1}{2m} \sum_{i=1}^m \left( h_{\theta}(x^{(i)}) - y^{(i)} \right)^2 + \text{Regularization Term}$$
$$10000\theta_3^2 + 10000\theta_4^2$$

*Min  $J(\theta)$ , getting  $\theta_3 \approx 0$ ,  $\theta_4 \approx 0$*

# Регуляризація: як це працює

Все працює через те, що значення  $J(\theta)$  представляє помилку навчання, і це значення має бути позитивним ( $\geq 0$ ). Параметр  $\lambda = 1000$  суттєво впливає на  $J(\theta)$ , тому  $\theta_3$  і  $\theta_4$  повинні бути близькі до нуля (наприклад, 0,000001), щоб уникнути великих помилок.

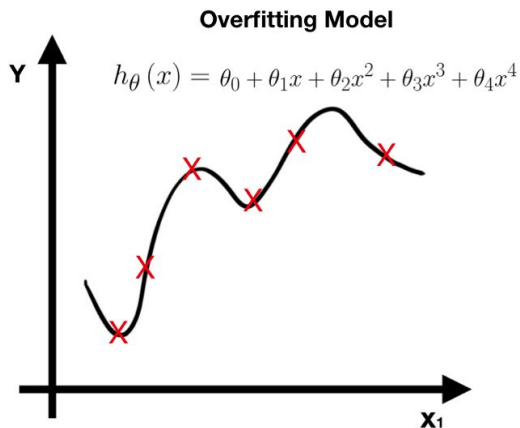
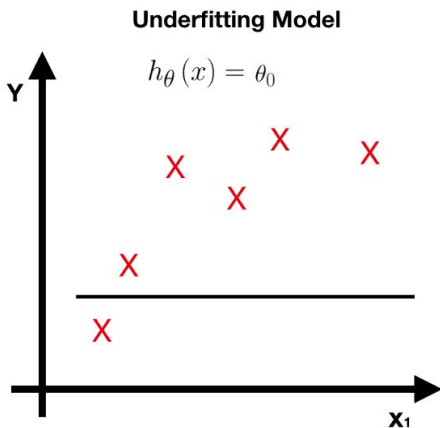
$$J(\theta) = \frac{1}{2m} \sum_{i=1}^m \left( h_{\theta}(x^{(i)}) - y^{(i)} \right)^2 + 1000 \theta_3^2 + 1000 \theta_4^2$$

Positive

```
graph TD; A["(h_theta(x^(i)) - y^(i))^2"] --> B[Positive]; C["1000 * theta_3^2"] --> B; D["1000 * theta_4^2"] --> B;
```

# Параметр регуляризації $\lambda$

- Якщо  $\lambda$  дуже високий — модель стане надто простою і ми зіткнемося з недоосвіченістю.
- Якщо  $\lambda$  дорівнює нулю або занадто мало, його вплив на параметри  $\theta$  незначний. Це може спричинити перенавчання.



# Параметр регуляризації $\lambda$ — гіперпараметр моделі

Ми не навчаємо  $\lambda$  під час навчання, а встановлюємо його **перед початком навчання**. Щоб вибрати оптимальне значення, ми можемо встановити сітку значень  $\lambda$  (наприклад, [0.001, 0.005, 0.01, 0.02, 0.05, 0.1, 0.5]), навчити модель з кожним значенням і вибрати ту, у якої найкраще значення помилки на валідаційних даних.

Ця процедура називається Grid Search — пошук гіперпараметрів по сітці.

# Як виглядає градієнтний спуск у випадку регуляризації

$$J(\theta) = \frac{1}{2m} \sum_{i=1}^m \left( h_{\theta}(x^{(i)}) - y^{(i)} \right)^2 + \boxed{\lambda \sum_{j=1}^n \theta_j^2}$$

Regularization Term  
Regularization Parameter

start at  $\theta_1$

Repeat until converge {

$$\theta_0 := \theta_0 - \alpha \frac{\partial J(\theta)}{\partial \theta_0} = \theta_0 - \alpha \frac{1}{m} \sum_{i=1}^m \left( h_{\theta}(x^{(i)}) - y^{(i)} \right) x_0^{(i)}$$

$$\theta_j := \theta_j - \alpha \frac{\partial J(\theta)}{\partial \theta_j} = \theta_j - \alpha \left[ \frac{1}{m} \sum_{i=1}^m \left( h_{\theta}(x^{(i)}) - y^{(i)} \right) x_j^{(i)} + \frac{\lambda}{m} \theta_j \right] \quad j = 1, 2, \dots, n$$

regularization term

}

# Гребінна регресія / Ridge Regression

Розглянута на останніх слайдах регресія називається гребінною, а така регуляризація - L2 регуляризацією, оскільки тут ми розглядаємо довжину вектора як суму квадратів координат. L2 - прийняте в математиці позначення для цієї метрики (довжина-2).

$$\text{Loss} = \text{Loss}_{\text{original}} + \lambda \sum_{i=1}^n w_i^2$$

де:

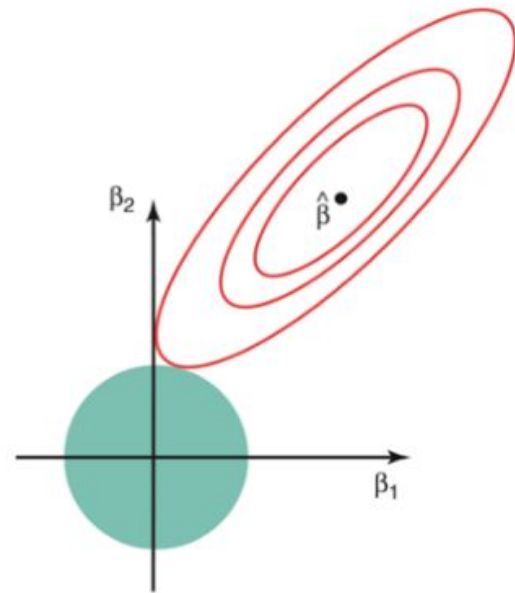
- $\text{Loss}_{\text{original}}$  – оригінальна функція втрат (наприклад, середньоквадратична похибка),
- $\lambda$  – параметр регуляризації, який визначає величину штрафу,
- $w_i$  – параметри моделі.

# Гребенева регресія

## / Ridge regression

Порівняння оцінок МНК та оцінок гребеневої регресії у двовимірному випадку. Оцінки гребеневої регресії обмежені колом (фактично, це задача оптимізації з обмеженням області значень змінної).

Оцінки ГР можна розглядати як точку, в якій контури коефіцієнтів лінійної регресії перетинають коло, визначене як  $\beta_1^2 + \beta_2^2 \leq \lambda$ .



# Ласо регресія

## / Регресія Лассо

L1-регуляризація додає штраф, пропорційний сумі абсолютних значень коефіцієнтів моделі, до функції втрат. Формально це виглядає так:

$$\text{Loss} = \text{Loss}_{\text{original}} + \lambda \sum_{i=1}^n |w_i|$$

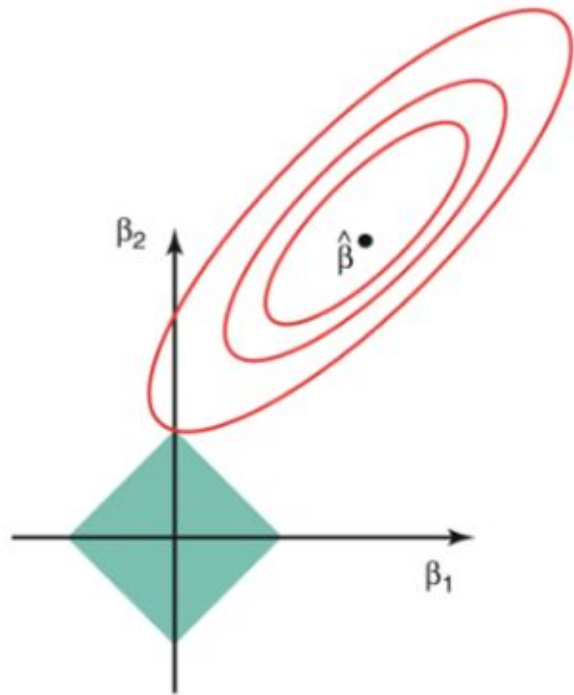


# Регресія Лассо

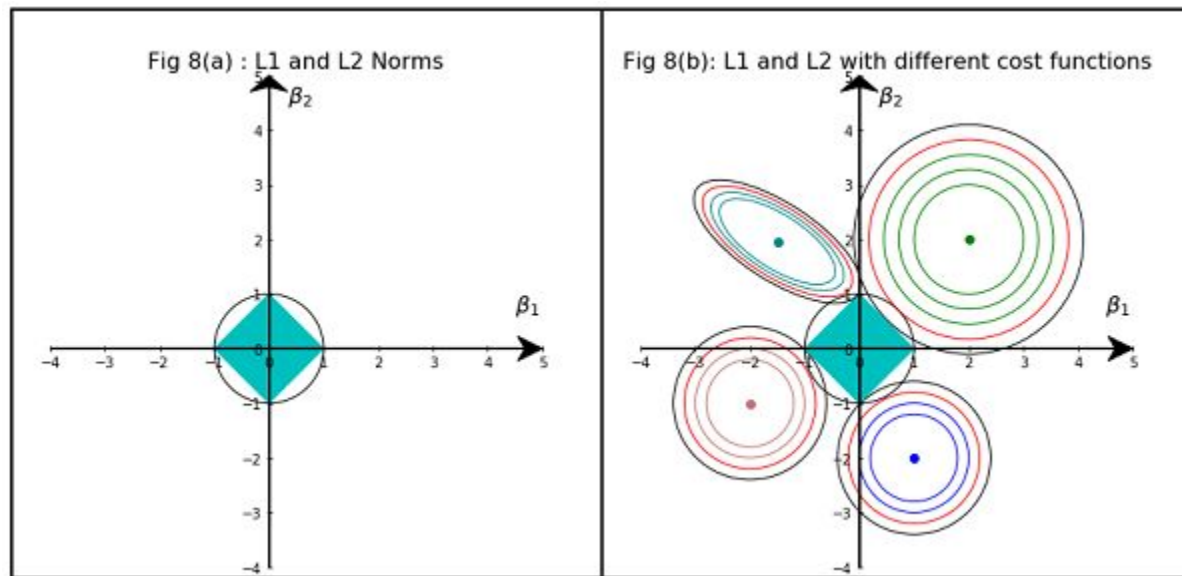
## / Lasso Regression

Тепер ми перетинаємо контури вагів не з кругом, а з квадратом і якщо перетин у кутку квадрата — то один з коефіцієнтів = 0 після оптимізації.

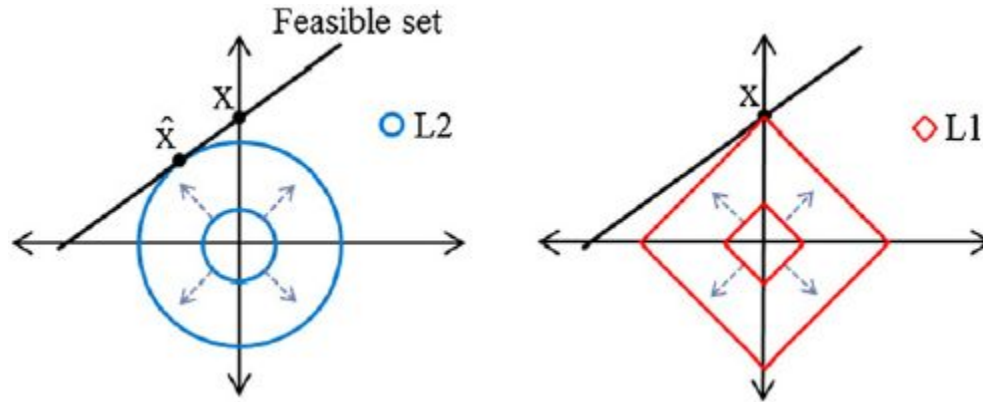
Тому регресія Lasso може використовуватись для відбору ознак (feature selection).



## Порівняння того, як контури дотикаються обмежень в L1 і L2



При L1 регуляризації коли наближаємося до області допустимих значень параметрів то часто можемо потрапити на кут, у той час, як при L2 ми майже завжди будемо потрапляти в область круга, де обидва параметри не дорівнюють 0.



# Еластична мережа / Elastic Net

Комбінація Ridge і Lasso регресій в одній (червона лінія на малюнку).  
Поєднує переваги обох методів, дозволяючи контролювати як розрідженість, так і величини параметрів.

$$\text{Loss} = \text{Loss}_{\text{original}} + \lambda_1 \sum_{i=1}^n |w_i| + \lambda_2 \sum_{i=1}^n w_i^2$$

