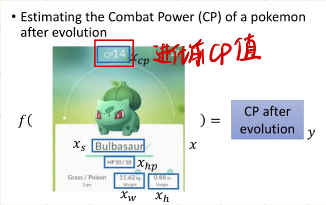
# regression

① Regression: output a scalar

eg: ① stock Market Forecast ② self-driving Car ③ Recommendation

② example: estimating the CP of a pokemon after evolution




## Step1 Model

simple model $y = b + w \underline{x_{cp}}$ 进化前CP（$y$：进化后CP）

$\Downarrow$

(linear model) $y = b + \sum w_i \underline{x_i}$

$x_i$: $x_{cp}$ $x_{hp}$ $x_w$ $x_h$ $\boxed{\text{feature}}$

$w$: weight　$b$: bias

## Step2 Goodness of function

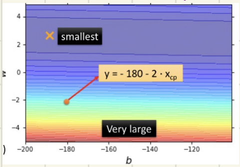
需评估模型好坏 ⟶ loss function

a. a set of function

model: $f_1, f_2$ ---

b. Goodness of function $f$

definite loss function $L$: $L(f) = L(w,b) = \sum_{n=1}^{10} (\hat{y}^n - f(x_{cp}^n))^2 = \sum_{n=1}^{10} (\hat{y}^n - (b + w x_{cp}^n))^2$




c. training data

## Step3: Best function

$\boxed{\text{pick the "Best" function}}$

$\Downarrow$

$f^* = \arg\min_f L(f)$

$w^*, b^* = \arg\min_{w,b} L(w,b)$

$\Downarrow$

$\boxed{\text{Gradient Descent}}$

## Gradient Descent

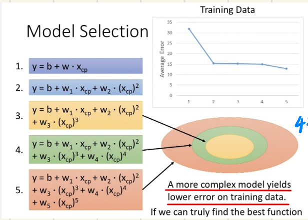
主要解决 local minima 与 global minima 的问题

linear regression 上没有 local minima problem

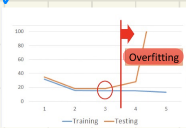
In linear regression, the loss function $L$ is convex. No local optimal

因为 linear regression 的 loss function 是凸面的，即像碗状的，把个球放在碗的任何一个位置都会滑到碗的底部，但若考虑更复杂的 model，则需考虑 local minima.

## Model Selection




4次的式子包含3次的式子，即 $w_4=0$

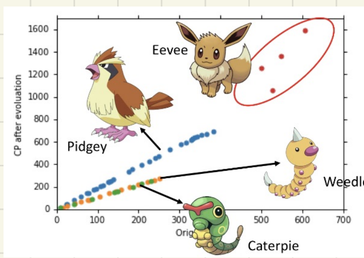



| | Training | Testing |
|---|---|---|
| 1 | 31.9 | 35.0 |
| 2 | 15.4 | 18.4 |
| 3 | 15.3 | 18.1 |
| 4 | 14.9 | 28.2 |
| 5 | 12.8 | 232.1 |

A more complex model does not always lead to better performance on testing data.

This is overfitting ⇒ select comfortable data

let's collect more data!




hidden factors：物种

## Then we redesign the Model:

$$
\begin{aligned}
y = b_1 &\cdot \delta(x_s = \text{Pidgey}) \\
+w_1 &\cdot \delta(x_s = \text{Pidgey})x_{cp} \\
+b_2 &\cdot \delta(x_s = \text{Weedle}) \\
+w_2 &\cdot \delta(x_s = \text{Weedle})x_{cp} \\
+b_3 &\cdot \delta(x_s = \text{Caterpie}) \\
+w_3 &\cdot \delta(x_s = \text{Caterpie})x_{cp} \\
+b_4 &\cdot \delta(x_s = \text{Eevee}) \\
+w_4 &\cdot \delta(x_s = \text{Eevee})x_{cp}
\end{aligned}
$$

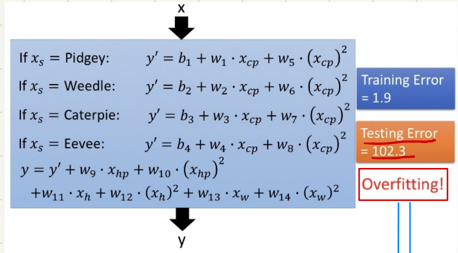
feature

$$\delta(x_s = \text{Pidgey}) = \begin{cases} 1 & \text{if } x_s = \text{Pidgey} \\ 0 & \text{otherwise} \end{cases}$$

if $x_s = \text{Pidgey}$

$y = b_1 + w_1 \cdot x_{cp}$

# Again redesign the model:




solve this problem

**Regularization**

Why: 因为我们本身不知道哪些factor是真正影响的，也没有办法过滤掉那些因素，但是有另外一种方式对很多不同的test都有用的方法，即Regularization 即改变 loss function

$$y = b + \sum w_i x_i$$

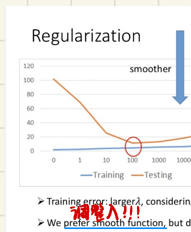
$$L = \sum_n \left(\hat{y}^n - (b + \sum w_i x_i)\right)^2 + \boxed{\lambda \sum (w_i)^2}$$

原来的 loss function 只考虑了 testing error 的大小，新的function加了一项：

$$y = b + \sum w_i x_i$$

$$y + \sum w_i \Delta x_i = b + \sum w_i (x_i + \Delta x_i)$$

If some noises corrupt input $x_i$ when testing, A smooth function has less influence.




| $\lambda$ | Training | Testing |
|---|---|---|
| 0 | 1.9 | 102.3 |
| 1 | 2.3 | 68.7 |
| 10 | 3.5 | 25.7 |
| 100 | 4.1 | 11.1 |
| 1000 | 5.6 | 12.8 |
| 10000 | 6.3 | 18.7 |
| 100000 | 8.5 | 26.8 |

- Training error: larger $\lambda$, considering the training error less

调整λ!!!

- We prefer smooth function, but don't be too smooth. 水平线，就没用了

Training error 随着 λ 的增大 ↑，合理，function 更倾向于考虑 W 的本身大小

## conclusion: