Regresión

Taller de Procesamiento de Señales / Introducción a la Inteligencia Artificial

Agenda

Teoría de Regresión

Bases

Objetivo: Predecir el valor de Y a partir de $X \rightarrow \quad \hat{Y} = \varphi(X)$
Función costo: Error cuadrático $\rightarrow \quad \ell(x, y) = (y - \varphi(x))^2$
Riesgo Esperado: MSE $\rightarrow \quad \mathbb{E}[\ell(X, Y)] = \mathbb{E}\left[(Y - \varphi(X))^2\right]$

Teoría de Regresión

Bases

Objetivo: Predecir el valor de Y a partir de $X \rightarrow \quad \hat{Y} = \varphi(X)$
Función costo: Error cuadrático $\rightarrow \quad \ell(x, y) = (y - \varphi(x))^2$
Riesgo Esperado: MSE $\rightarrow \quad \mathbb{E}[\ell(X, Y)] = \mathbb{E}\left[(Y - \varphi(X))^2\right]$

Optimalidad

$$\mathbb{E}\left[(Y - \varphi(X))^2\right] \geq \mathbb{E}\left[\text{var}(Y|X)\right]$$

con igualdad si y solo si $\varphi(x) = \mathbb{E}[Y|X = x]$.
Regresor óptimo: $\varphi(x) = \mathbb{E}[Y|X = x]$
Error Bayesiano: $\mathbb{E}\left[\text{var}(Y|X)\right]$

Reconocimiento de patrones

Objetivo

Quiero buscar $\varphi(\cdot)$ que minimice $\mathbb{E}[\ell(X, Y)]$. Es decir aprender la "esperanza condicional".

Reconocimiento de patrones

Objetivo

Quiero buscar $\varphi(\cdot)$ que minimice $\mathbb{E}[\ell(X, Y)]$. Es decir aprender la "esperanza condicional".

Empirical Risk Minimization (ERM)

Propongo buscar $\varphi(\cdot)$ que minimice el riesgo empírico: $\frac{1}{n}\sum_{i=1}^{n}\ell(X_i, Y_i)$

Reconocimiento de patrones

Objetivo

Quiero buscar $\varphi(\cdot)$ que minimice $\mathbb{E}[\ell(X, Y)]$. Es decir aprender la "esperanza condicional".

Empirical Risk Minimization (ERM)

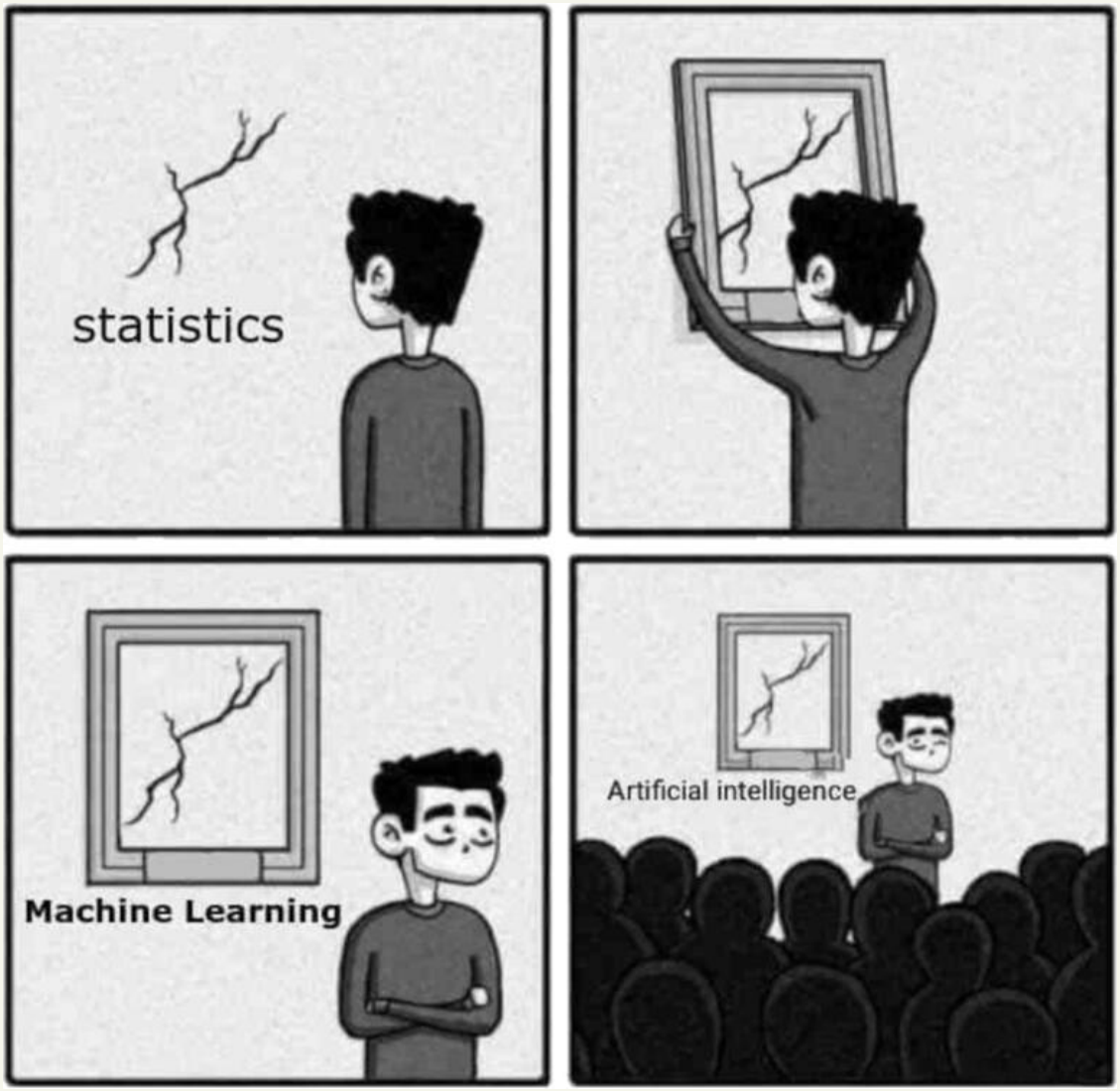
Propongo buscar $\varphi(\cdot)$ que minimice el riesgo empírico: $\frac{1}{n}\sum_{i=1}^{n}\ell(X_i, Y_i)$

Tradeoff: Sesgo/Varianza

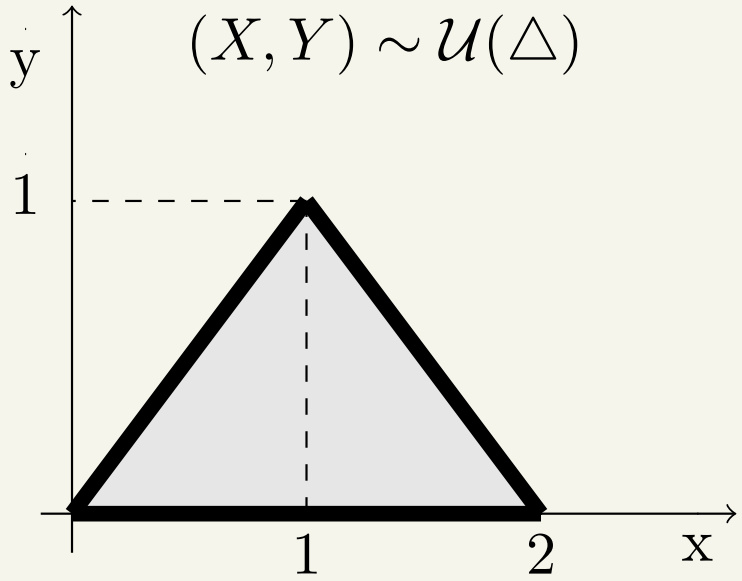
$$\underbrace{\mathbb{E}[\ell(X,Y)]}_{\text{Riesgo esperado}} = \underbrace{\frac{1}{n}\sum_{i=1}^{n}\ell(X_i,Y_i)}_{\text{Riesgo empírico}} + \left(\underbrace{\mathbb{E}[\ell(X,Y)] - \frac{1}{n}\sum_{i=1}^{n}\ell(X_i,Y_i)}_{\text{Gap de generalización}}\right)$$

Nota: El riesgo empírico se considera grande o pequeño comparándolo con el error bayesiano.

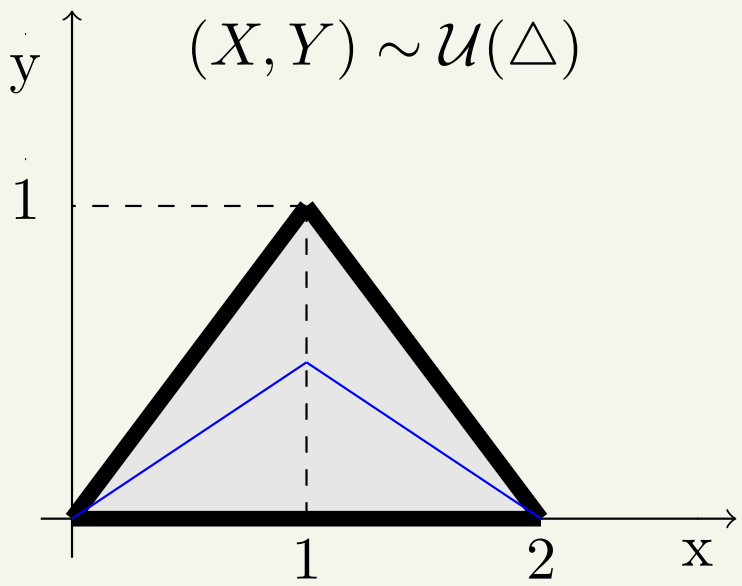
¿Que es la Inteligencia Artificial?



Overfitting y Underfitting



Overfitting y Underfitting

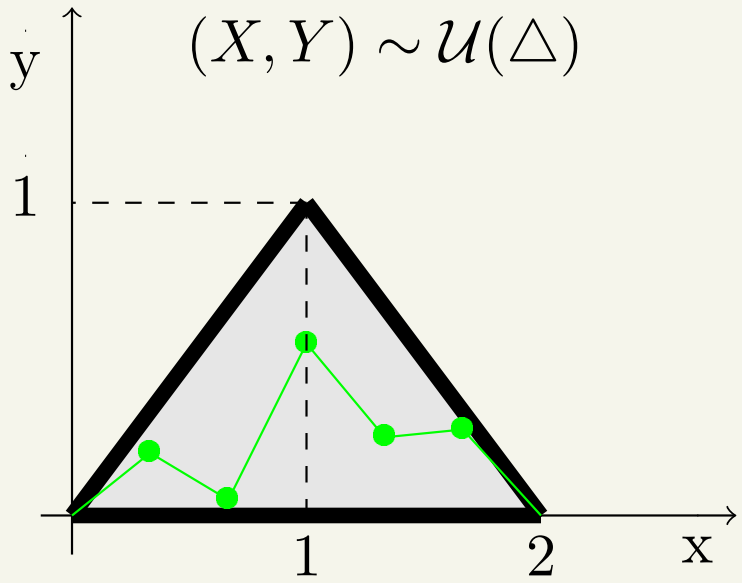


Solución Óptima

- El regresor elegido es efectivamente la esperanza condicional.
- El riesgo esperado alcanza el límite bayesiano

$$\mathbb{E}\left[\text{var}(Y|X)\right] = \frac{1}{24}$$

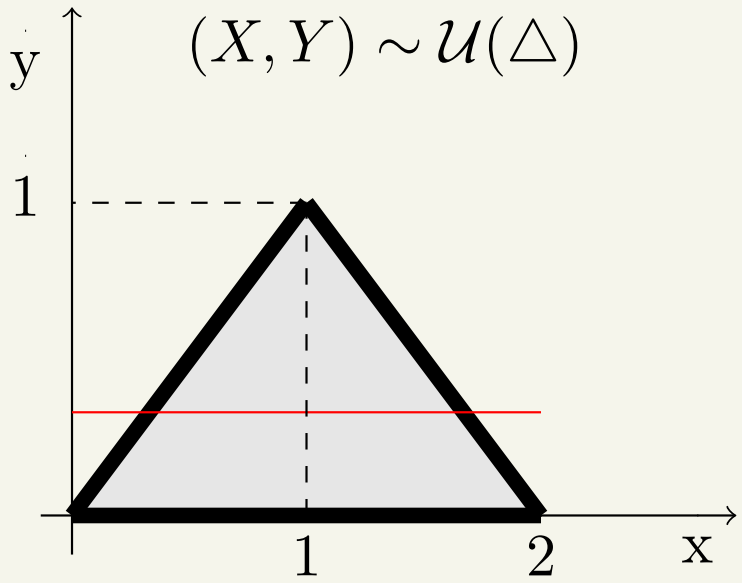
Overfitting y Underfitting



Problema de overfitting

- Riesgo empírico muy bajo (puede ser menor incluso que el bayesiano)
- Se detecta por el alto gap de generalización.
- Exceso de complejidad en el modelado.
- Se dice que el algoritmo tiene un problema de varianza.

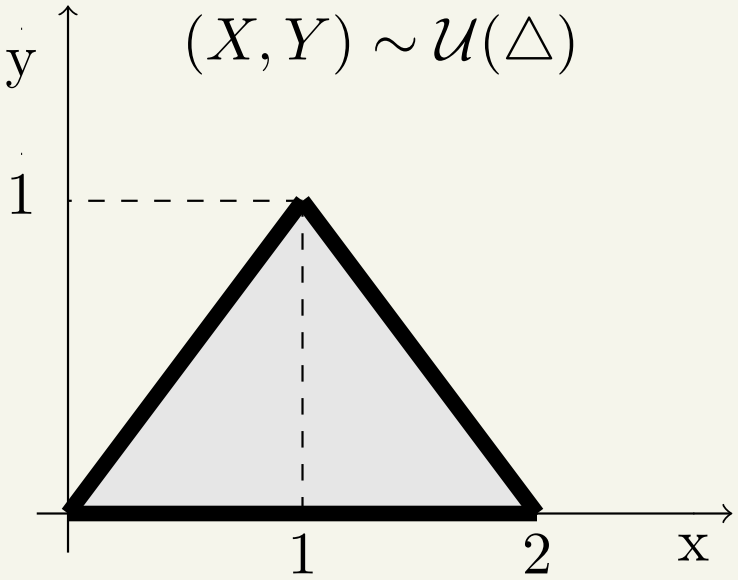
Overfitting y Underfitting



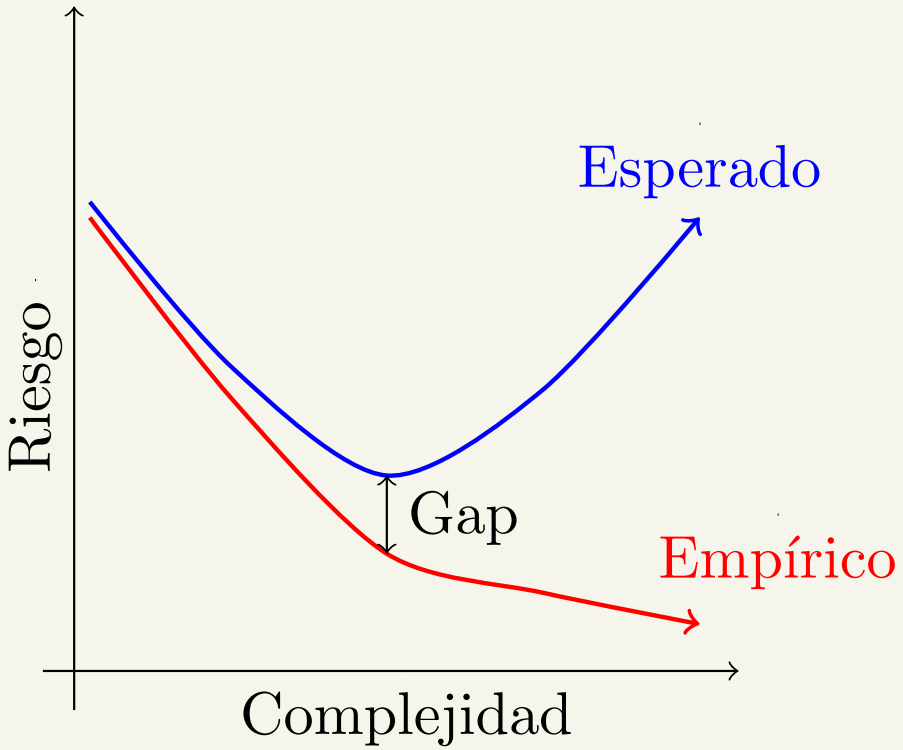
Problema de underfitting

- Suele tener bajo gap de generalización.
- Riesgo empírico muy superior al error bayesiano.
- Escasez de complejidad en el modelado.
- Se dice que el algoritmo tiene un problema de sesgo.

Overfitting y Underfitting



Teoría Clásica de Generalización



Problemas básicos del aprendizaje estadístico

Sobreajuste

Subajuste

Regresión Lineal: $\hat{Y} = w^T \cdot X + b$

Idea

Me aseguro mantener acotado el problema de overfitting proponiendo una solución de extremandamente baja complejidad. Si se alcanza bajo error empírico, entonces tengo ciertas garantías de que el algoritmo alcanza un buen desempeño.

Regresión Lineal: $\hat{Y} = w^T \cdot X + b$

Idea

Me aseguro mantener acotado el problema de overfitting proponiendo una solución de extremandamente baja complejidad. Si se alcanza bajo error empírico, entonces tengo ciertas garantías de que el algoritmo alcanza un buen desempeño.

Empirical Risk Minimization

$$(w, b) \in \arg\min_{(w,b)} \sum_{i=1}^{n} (w^T \cdot X_i + b - Y_i)^2$$

Regresión Lineal: $\hat{Y} = w^T \cdot X + b$

Empirical Risk Minimization

$$(w, b) \in \arg\min_{\mathbf{w}} \|\mathbf{X} \cdot \mathbf{w} - \mathbf{y}\|^2,$$

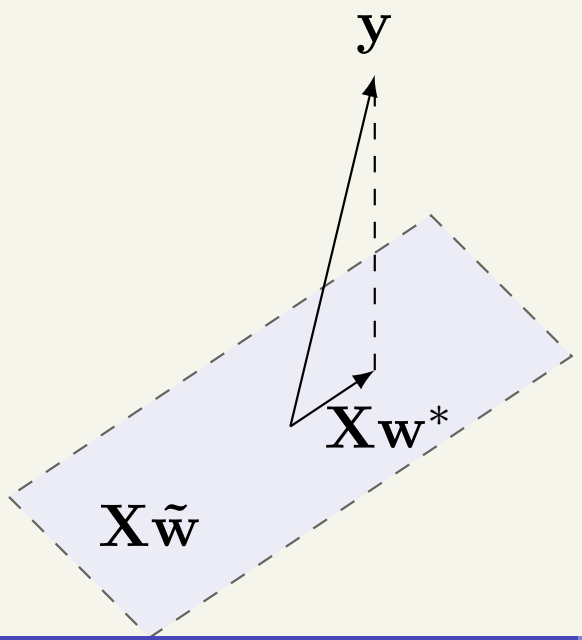
$$\mathbf{X} = \begin{pmatrix} 1 & X_1^T \\ 1 & X_2^T \\ \vdots & \vdots \\ 1 & X_n^T \end{pmatrix}, \quad \mathbf{y} = \begin{pmatrix} Y_1 \\ Y_2 \\ \vdots \\ Y_n \end{pmatrix}, \quad \mathbf{w} = \begin{pmatrix} b \\ w \end{pmatrix}$$

Regresión Lineal: $\hat{Y} = w^T \cdot X + b$

Empirical Risk Minimization

$$(w, b) \in \arg\min_{\mathbf{w}} \|\mathbf{X} \cdot \mathbf{w} - \mathbf{y}\|^2,$$

$$\mathbf{X} = \begin{pmatrix} 1 & X_1^T \\ 1 & X_2^T \\ \vdots & \vdots \\ 1 & X_n^T \end{pmatrix}, \quad \mathbf{y} = \begin{pmatrix} Y_1 \\ Y_2 \\ \vdots \\ Y_n \end{pmatrix}, \quad \mathbf{w} = \begin{pmatrix} b \\ w \end{pmatrix}$$



$$\mathbf{w}^* = \mathbf{X}^\dagger \mathbf{y}$$

Regresión Lineal

Solución matricial óptima: Recta de Regresión

$$\mathbf{w} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}$$

Petersen and Pedersen - "Matrix Cookbook".

Regresión Lineal

Solución matricial óptima: Recta de Regresión

$$\mathbf{w} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}$$

Derivadas Matriciales

$$\nabla(\mathbf{x}^T\mathbf{a}) = \nabla(\mathbf{a}^T\mathbf{x}) = \mathbf{a}$$

$$\nabla(\mathbf{x}^T\mathbf{B}\mathbf{x}) = (\mathbf{B} + \mathbf{B}^T)\mathbf{x}$$

$$\mathcal{H}(\mathbf{x}^T\mathbf{B}\mathbf{x}) = \mathbf{B} + \mathbf{B}^T$$

¿Cual es el gradiente de $J(\mathbf{w}) = \frac{1}{n}\|\mathbf{X}\cdot\mathbf{w} - \mathbf{y}\|^2$?

Petersen and Pedersen - "Matrix Cookbook".

Regresión Lineal

Solución matricial óptima: Recta de Regresión

$$\mathbf{w} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}$$

Derivadas Matriciales

$$\nabla(\mathbf{x}^T\mathbf{a}) = \nabla(\mathbf{a}^T\mathbf{x}) = \mathbf{a}$$

$$\nabla(\mathbf{x}^T\mathbf{B}\mathbf{x}) = (\mathbf{B} + \mathbf{B}^T)\mathbf{x}$$

$$\mathcal{H}(\mathbf{x}^T\mathbf{B}\mathbf{x}) = \mathbf{B} + \mathbf{B}^T$$

¿Cual es el gradiente de $J(\mathbf{w}) = \frac{1}{n}\|\mathbf{X}\cdot\mathbf{w} - \mathbf{y}\|^2$?

Optimización convexa

El problema de regresión lineal es un problema convexo.

Petersen and Pedersen - "Matrix Cookbook".

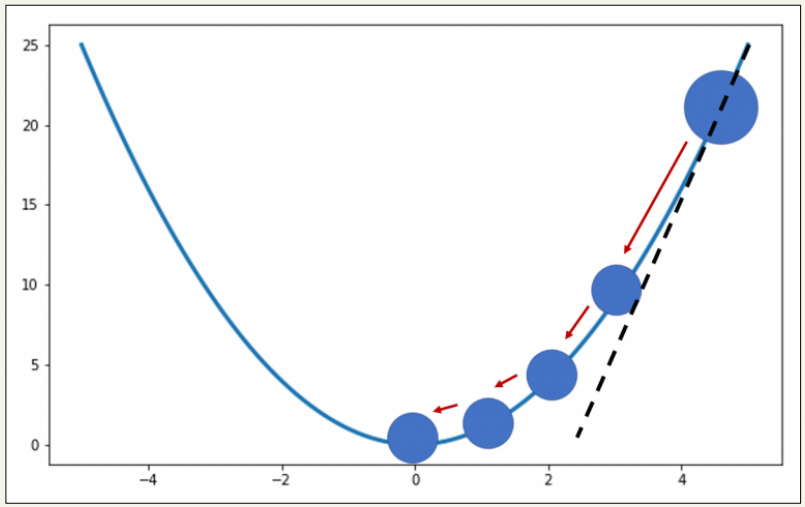
Outline

Gradiente Descendente

Problema a resolver: $\min_{\theta \in \Theta} J(\theta)$.

Solución:

$$\theta_{t+1} = \theta_t - \alpha \nabla J(\theta_t)$$



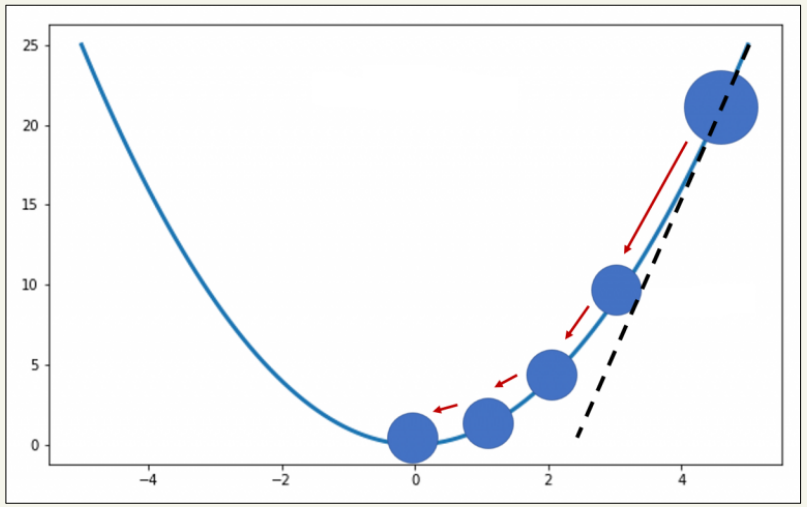
Cauchy 1847: "Méthode générale pour la résolution de systèmes d'équations simultanées".

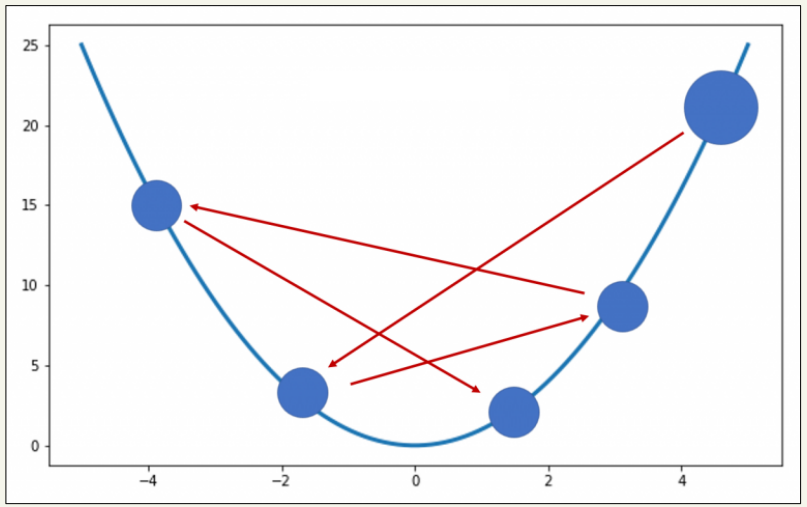
Gradiente Descendente

Problema a resolver: $\min_{\theta \in \Theta} J(\theta)$.

Solución:

$$\theta_{t+1} = \theta_t - \alpha \nabla J(\theta_t)$$





- Si α es chico la convergencia es lenta.
- Si α es grande puede no converger.

Cauchy 1847: "Méthode générale pour la résolution de systèmes d'équations simultanées".

Convergencia y optimización para problemas convexos

Modelo convexo con derivadas segunda continuas

- $\mathbf{w}_{t+1} = \mathbf{w}_t - \alpha \nabla J(\mathbf{w}_t)$.
- Existe un único $\mathbf{w}^*$ tal que $\nabla J(\mathbf{w}^*) = 0$.
- $\mathcal{H}(\mathbf{w})$ es definido positivo para todo $\mathbf{w}$.

Convergencia y optimización para problemas convexos

Modelo convexo con derivadas segunda continuas

- $\mathbf{w}_{t+1} = \mathbf{w}_t - \alpha \nabla J(\mathbf{w}_t)$.
- Existe un único $\mathbf{w}^*$ tal que $\nabla J(\mathbf{w}^*) = 0$.
- $\mathcal{H}(\mathbf{w})$ es definido positivo para todo $\mathbf{w}$.

Teorema de Taylor

$$\nabla J(\mathbf{w}_t) = \nabla J(\mathbf{w}^*) + \mathcal{H}(\tilde{\mathbf{w}}) \cdot (\mathbf{w}_t - \mathbf{w}^*)$$

para algún $\tilde{\mathbf{w}}$ en el segmento que une $\mathbf{w}_t$ y $\mathbf{w}^*$.

Convergencia y optimización para problemas convexos

Modelo convexo con derivadas segunda continuas

- $\mathbf{w}_{t+1} = \mathbf{w}_t - \alpha \nabla J(\mathbf{w}_t)$.
- Existe un único $\mathbf{w}^*$ tal que $\nabla J(\mathbf{w}^*) = 0$.
- $\mathcal{H}(\mathbf{w})$ es definido positivo para todo $\mathbf{w}$.

Teorema de Taylor

$$\nabla J(\mathbf{w}_t) = \nabla J(\mathbf{w}^*) + \mathcal{H}(\tilde{\mathbf{w}}) \cdot (\mathbf{w}_t - \mathbf{w}^*)$$

para algún $\tilde{\mathbf{w}}$ en el segmento que une $\mathbf{w}_t$ y $\mathbf{w}^*$.

Diagonalización ortogonal

Toda matriz real, cuadrada y simétrica puede escribirse como $H = Q^T \Lambda Q$ con una matriz de autovalores Λ diagonal y una de autovectores Q ortogonal $Q^T Q = Q Q^T = I$.

Convergencia y optimización para problemas convexos

$$\nabla J(\mathbf{w}_t) = \nabla J(\mathbf{w}^*) + \mathcal{H}(\tilde{\mathbf{w}}) \cdot (\mathbf{w}_t - \mathbf{w}^*) = Q^T \Lambda Q \cdot (\mathbf{w}_t - \mathbf{w}^*)$$

con Q y Λ las matrices correspondientes a la diagonalización de $\mathcal{H}(\tilde{\mathbf{w}})$.

Convergencia y optimización para problemas convexos

$$\nabla J(\mathbf{w}_t) = \nabla J(\mathbf{w}^*) + \mathcal{H}(\tilde{\mathbf{w}}) \cdot (\mathbf{w}_t - \mathbf{w}^*) = Q^T \Lambda Q \cdot (\mathbf{w}_t - \mathbf{w}^*)$$

con Q y Λ las matrices correspondientes a la diagonalización de $\mathcal{H}(\tilde{\mathbf{w}})$.

Regresión lineal

Si $J(\mathbf{w}) = \frac{1}{n}\|\mathbf{X} \cdot \mathbf{w} - \mathbf{y}\|^2$, $\mathcal{H}(\tilde{\mathbf{w}}) = \frac{2}{n}\mathbf{X}^T\mathbf{X}$ no depende del valor de los parámetros (solo de los datos).

Convergencia y optimización para problemas convexos

$$\nabla J(\mathbf{w}_t) = \nabla J(\mathbf{w}^*) + \mathcal{H}(\tilde{\mathbf{w}}) \cdot (\mathbf{w}_t - \mathbf{w}^*) = Q^T \Lambda Q \cdot (\mathbf{w}_t - \mathbf{w}^*)$$

con Q y Λ las matrices correspondientes a la diagonalización de $\mathcal{H}(\tilde{\mathbf{w}})$.

Regresión lineal

Si $J(\mathbf{w}) = \frac{1}{n}\|\mathbf{X} \cdot \mathbf{w} - \mathbf{y}\|^2$, $\mathcal{H}(\tilde{\mathbf{w}}) = \frac{2}{n}\mathbf{X}^T\mathbf{X}$ no depende del valor de los parámetros (solo de los datos).

$$\begin{aligned}
\mathbf{w}_{t+1} - \mathbf{w}^* &= \mathbf{w}_t - \mathbf{w}^* - \alpha \nabla J(\mathbf{w}_t) \\
&= \left(I - \alpha Q^T \Lambda Q\right)(\mathbf{w}_t - \mathbf{w}^*) \\
&= Q^T \left(I - \alpha \Lambda\right) Q \,(\mathbf{w}_t - \mathbf{w}^*)
\end{aligned}$$

Convergencia y optimización para problemas convexos

$$\nabla J(\mathbf{w}_t) = \nabla J(\mathbf{w}^*) + \mathcal{H}(\tilde{\mathbf{w}}) \cdot (\mathbf{w}_t - \mathbf{w}^*) = Q^T \Lambda Q \cdot (\mathbf{w}_t - \mathbf{w}^*)$$

con Q y Λ las matrices correspondientes a la diagonalización de $\mathcal{H}(\tilde{\mathbf{w}})$.

Regresión lineal

Si $J(\mathbf{w}) = \frac{1}{n}\|\mathbf{X} \cdot \mathbf{w} - \mathbf{y}\|^2$, $\mathcal{H}(\tilde{\mathbf{w}}) = \frac{2}{n}\mathbf{X}^T\mathbf{X}$ no depende del valor de los parámetros (solo de los datos).

$$\begin{aligned}
\mathbf{w}_{t+1} - \mathbf{w}^* &= \mathbf{w}_t - \mathbf{w}^* - \alpha \nabla J(\mathbf{w}_t) \\
&= \left(I - \alpha Q^T \Lambda Q\right)(\mathbf{w}_t - \mathbf{w}^*) \\
&= Q^T \left(I - \alpha \Lambda\right) Q \left(\mathbf{w}_t - \mathbf{w}^*\right)
\end{aligned}$$

Defino $v_t = Q\left(\mathbf{w}_t - \mathbf{w}^*\right)$, luego:

$$v_{t+1} = \left(I - \alpha \Lambda\right) v_t,$$

Convergencia y optimización para problemas convexos

$$\nabla J(\mathbf{w}_t) = \nabla J(\mathbf{w}^*) + \mathcal{H}(\tilde{\mathbf{w}}) \cdot (\mathbf{w}_t - \mathbf{w}^*) = Q^T \Lambda Q \cdot (\mathbf{w}_t - \mathbf{w}^*)$$

con Q y Λ las matrices correspondientes a la diagonalización de $\mathcal{H}(\tilde{\mathbf{w}})$.

Regresión lineal

Si $J(\mathbf{w}) = \frac{1}{n}\|\mathbf{X} \cdot \mathbf{w} - \mathbf{y}\|^2$, $\mathcal{H}(\tilde{\mathbf{w}}) = \frac{2}{n}\mathbf{X}^T\mathbf{X}$ no depende del valor de los parámetros (solo de los datos).

$$\begin{aligned}
\mathbf{w}_{t+1} - \mathbf{w}^* &= \mathbf{w}_t - \mathbf{w}^* - \alpha \nabla J(\mathbf{w}_t) \\
&= \left(I - \alpha Q^T \Lambda Q\right)(\mathbf{w}_t - \mathbf{w}^*) \\
&= Q^T \left(I - \alpha \Lambda\right) Q \left(\mathbf{w}_t - \mathbf{w}^*\right)
\end{aligned}$$

Defino $v_t = Q\left(\mathbf{w}_t - \mathbf{w}^*\right)$, luego:

$$v_{t+1} = \left(I - \alpha\Lambda\right) v_t, \qquad v_t = \left(I - \alpha\Lambda\right)^t v_0$$

Convergencia y optimización para problemas convexos

Condición y velocidad de convergencia

El GD convergerá si $|1 - \alpha\lambda_j| < 1$ para todo j y el learning rate óptimo estará asociado al criterio de peor caso:

$$\min_{\alpha} \max_{j} |1 - \alpha\lambda_j| \quad \text{s.t.} \quad |1 - \alpha\lambda_j| < 1 \quad \forall\, j$$

Convergencia y optimización para problemas convexos

Condición y velocidad de convergencia

El GD convergerá si $|1 - \alpha\lambda_j| < 1$ para todo j y el learning rate óptimo estará asociado al criterio de peor caso:

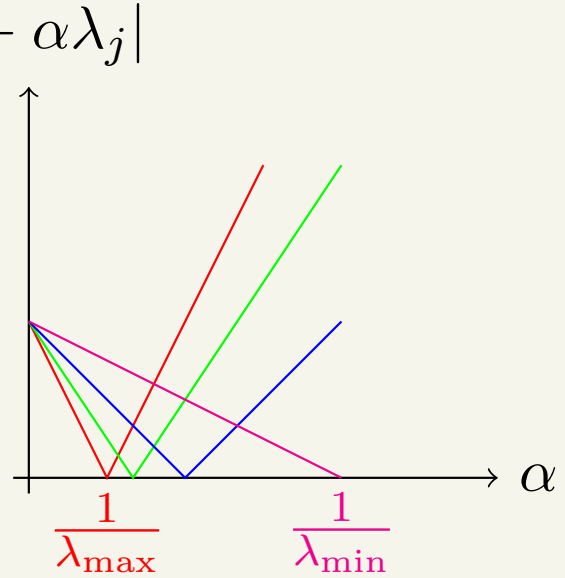
$$\min_{\alpha} \max_{j} |1 - \alpha\lambda_j| \quad \text{s.t.} \quad |1 - \alpha\lambda_j| < 1 \quad \forall\, j$$

Recordar que $\lambda_j > 0$ para todo j.

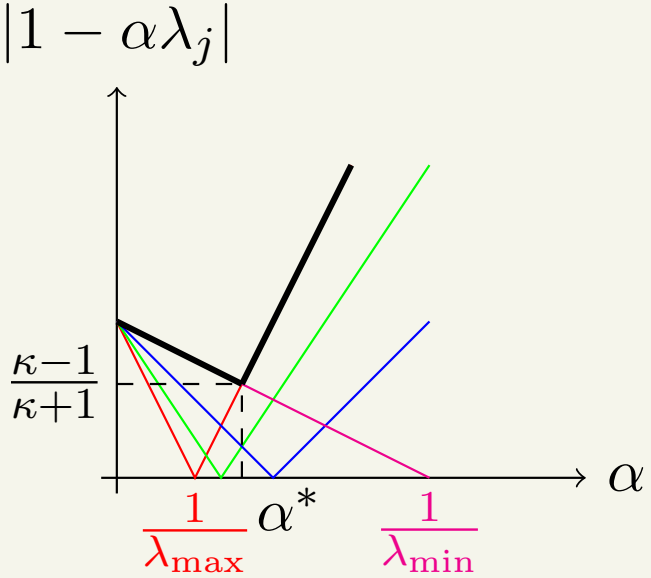
Condición de convergencia

$|1 - \alpha\lambda_j| < 1$ para todo j equivale a pedir $\alpha < \frac{2}{\lambda_{\max}}$.

Convergencia y optimización para problemas convexos



Convergencia y optimización para problemas convexos



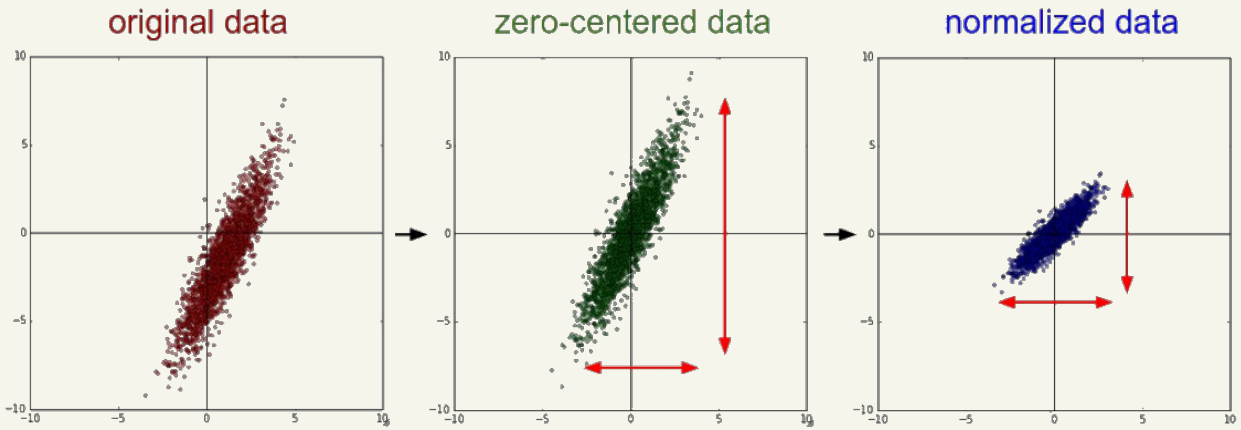
Velocidad de convergencia

El óptimo learning rate en este caso es $\alpha^* = \frac{2}{\lambda_{\max}+\lambda_{\min}}$ y su velocidad asociada $\left(\frac{\kappa-1}{\kappa+1}\right)^t$ depende del número de condición $\kappa = \frac{\lambda_{\max}}{\lambda_{\min}}$.

Optimalidad

El óptimo no es el más grande convergente: $\alpha^* = \frac{2}{\lambda_{\max}+\lambda_{\min}} < \frac{2}{\lambda_{\max}}$

Normalización de la entrada



Normalizar *cada componente* de la entrada tiene sus beneficios:

$$(\mathbf{x})_k \leftarrow \frac{(\mathbf{x})_k - \mu_k}{\sigma_k}$$

donde las μ_k y σ_k son calculadas previo al entrenamiento como:

$$\mu_k = \frac{1}{n_{\text{tr}}} \sum_{i=1}^{n_{\text{tr}}} (\mathbf{x}_i)_k, \qquad \sigma_k = \sqrt{\frac{1}{n_{\text{tr}}} \sum_{i=1}^{n_{\text{tr}}} \left[(\mathbf{x}_i)_k - \mu_k\right]^2}$$

¿Cuando y por que normalizar?

Normalizar si!

- Cuando quiero comparar magnitudes que por si solas no lo son.
- Cuando quiero corregir problemas de convergencia de los algoritmos.
- Cuando el algoritmo a utilizar, necesita la hipótesis de entradas normalizadas en su génesis.

Normalizar no!

Normalizar por las dudas o por costumbre es una mala práctica.

¿Cuando y por que normalizar?

Normalizar si!

- Cuando quiero comparar magnitudes que por si solas no lo son.
- Cuando quiero corregir problemas de convergencia de los algoritmos.
- Cuando el algoritmo a utilizar, necesita la hipótesis de entradas normalizadas en su génesis.

Normalizar no!

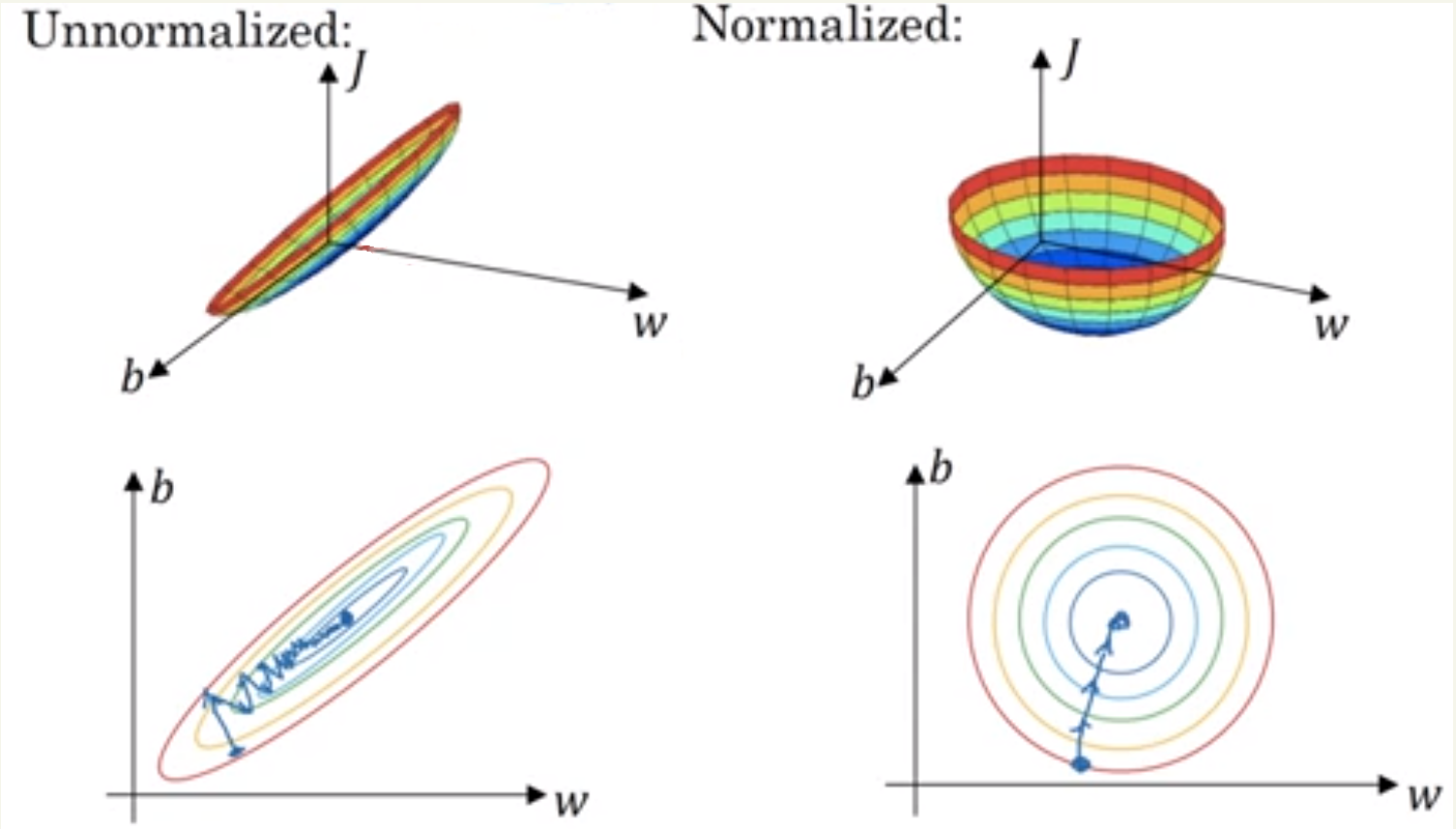
Normalizar por las dudas o por costumbre es una mala práctica.

¿Y si alguna varianza da cero?

Si algún $\sigma_k = 0$, significa que esa componente de la entrada es constante a lo largo de todo el conjunto de datos, y por lo tanto puede excluirse del análisis.

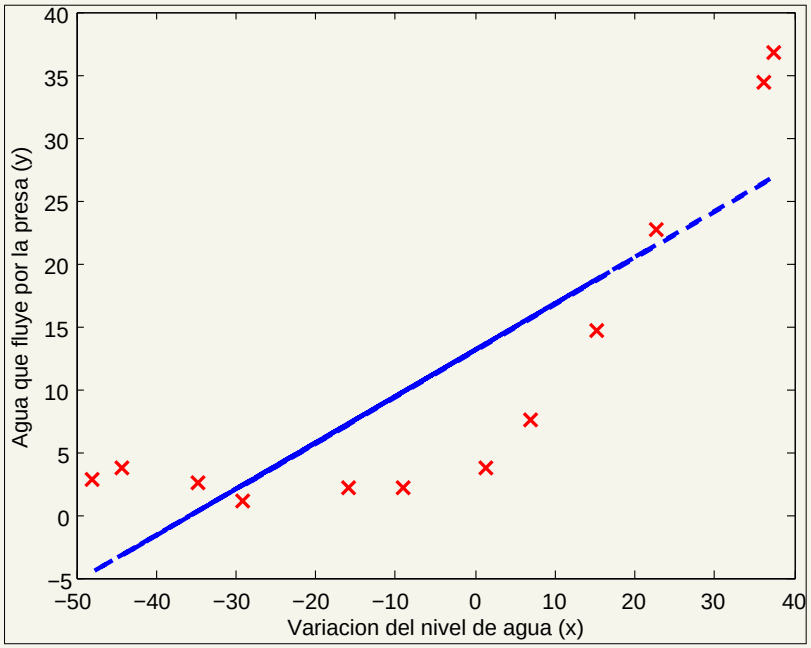
Normalización de la entrada

Normalizar me permite usar learning rates más grandes!

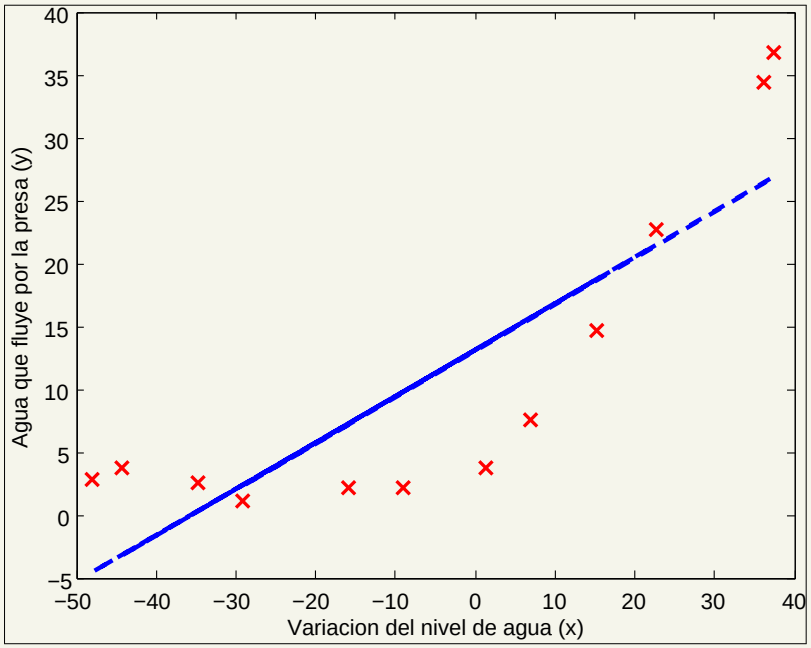


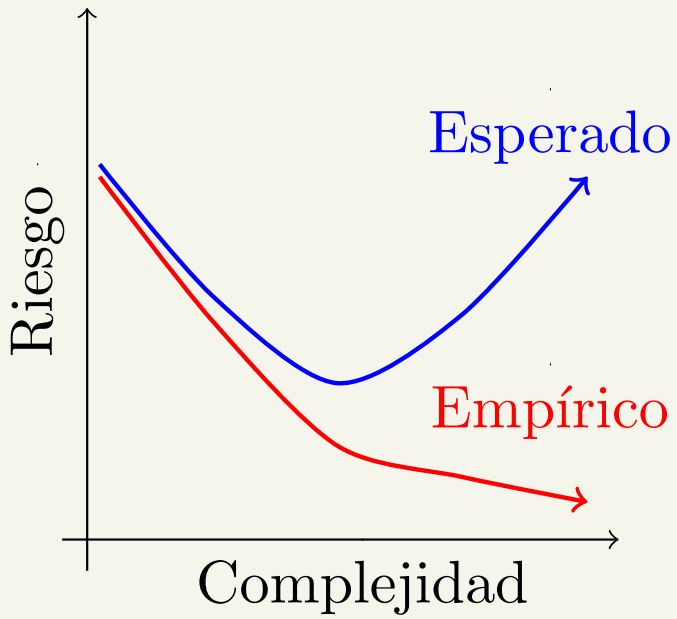
Outline

¿Y si la complejidad lineal no alcanza?



¿Y si la complejidad lineal no alcanza?

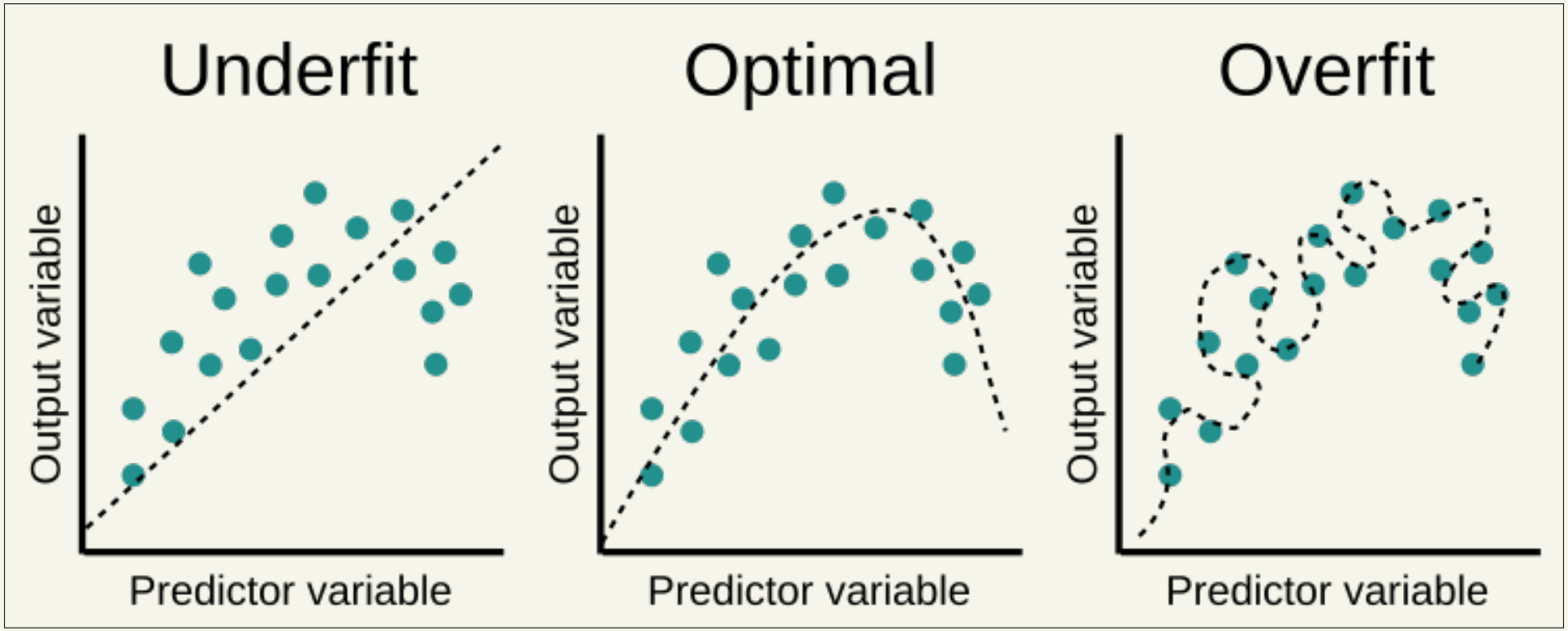




Regresión Polinómica

$$\mathbf{X} = \begin{pmatrix} 1 & X_{1,1} & X_{1,2} & X_{1,1}^2 & X_{1,2}^2 & X_{1,1}X_{1,2} \\ 1 & X_{2,1} & X_{2,2} & X_{2,1}^2 & X_{2,2}^2 & X_{2,1}X_{2,2} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & X_{n,1} & X_{n,2} & X_{n,1}^2 & X_{n,2}^2 & X_{n,1}X_{n,2} \end{pmatrix}$$

Compromiso Sesgo/Varianza



Si no puedo confiar en los datos de entrenamiento ¿Como procedo?

Conjuntos de datos

- Conjunto de entrenamiento (*train set*): Datos utilizados para minimizar el riesgo empírico. Sobre estos se produce el "aprendizaje". Las variables definidas a partir de este conjunto se llaman parámetros.
- Conjunto de validación (*validation or development set*): Datos utilizados para comparar modelos. Las variables definidas a partir de este conjunto (o definidas previas al entrenamiento) se llaman hiperparámetros.
- Conjunto de testeo (*test set*): Datos utilizados para evaluar la performance final del algoritmo. Su única función es presentar estimadores insesgados de las métricas de error y no es imprescindible.

Si la base de datos esta dividida, respetar la división!

Enfoque clásico: 60%/20%/20% - Típico para $1K, 10K$ muestras.

Big Data: Para $1M$ muestras, quizás alcanza con 98%/1%/1%.

Atacar el punto débil

¿Que conviene corregir? ¿Sesgo o varianza?

- **Avoidable bias**: Error de train - Error bayesiano
- **Generalization Gap**: Error de validación - Error de train

Atacar el punto débil

¿Que conviene corregir? ¿Sesgo o varianza?

- **Avoidable bias**: Error de train - Error bayesiano
- **Generalization Gap**: Error de validación - Error de train

Técnica Clásica de Regularización

Se agrega un término de penalización que perturba la optimización del riesgo empírico:

$$J(\theta) = \frac{1}{n} \sum_{i=1}^{n} L_i(\theta) + \lambda R(\theta)$$

Atacar el punto débil

¿Que conviene corregir? ¿Sesgo o varianza?

- **Avoidable bias**: Error de train - Error bayesiano
- **Generalization Gap**: Error de validación - Error de train

Técnica Clásica de Regularización

Se agrega un término de penalización que perturba la optimización del riesgo empírico:

$$J(\theta) = \frac{1}{n}\sum_{i=1}^{n} L_i(\theta) + \lambda R(\theta)$$

Motivación: Error de generalización

El regularizador trata ser representativo del error de generalización:

$$\mathbb{E}[L(\theta)] = \frac{1}{n}\sum_{i=1}^{n} L_i(\theta) + \left(\mathbb{E}[L(\theta)] - \frac{1}{n}\sum_{i=1}^{n} L_i(\theta)\right)$$

Regresión Lineal Regularizada

Weight decay or L2 regularization

$$J(\theta) = \frac{1}{n}\sum_{i=1}^{n} L_i(\theta) + \frac{\lambda}{n}\|\mathbf{w}\|^2 \rightarrow \frac{\partial \|\mathbf{w}\|^2}{\partial \mathbf{w}} = 2\mathbf{w}$$

Regresión Lineal Regularizada

Weight decay or L2 regularization

$$J(\theta) = \frac{1}{n}\sum_{i=1}^{n} L_i(\theta) + \frac{\lambda}{n}\|\mathbf{w}\|^2 \rightarrow \frac{\partial \|\mathbf{w}\|^2}{\partial \mathbf{w}} = 2\mathbf{w}$$

Interpretación 1: Apagar parámetros

$w_j \approx 0$ simplifica la complejidad del modelo.

Regresión Lineal Regularizada

Weight decay or L2 regularization

$$J(\theta) = \frac{1}{n}\sum_{i=1}^{n} L_i(\theta) + \frac{\lambda}{n}\|\mathbf{w}\|^2 \rightarrow \frac{\partial \|\mathbf{w}\|^2}{\partial \mathbf{w}} = 2\mathbf{w}$$

Interpretación 1: Apagar parámetros

$w_j \approx 0$ simplifica la complejidad del modelo.

Interpretación 2: Disminuir el máximo valor de la función costo

$$\mathbb{E}[L(\theta)] - \frac{1}{n}\sum_{i=1}^{n} L_i(\theta) \leq \max_{\phi \in \Theta} L(\phi)$$

Validación: ¿Como elijo el λ?

Set de Validación

Si tengo una buena cantidad de datos de validación, elijo el λ con menor error de validación.

Validación: ¿Como elijo el λ?

Set de Validación

Si tengo una buena cantidad de datos de validación, elijo el λ con menor error de validación.

Leave-one-out cross-validation (LOOCV)

Si tengo pocos datos no puedo tener un conjunto de datos de validación suficientemente rico. Entonces entreno con todas las muestras menos una y valido con la última. Luego repito esto con cada muestra y promedio.

Validación: ¿Como elijo el λ?

Set de Validación

Si tengo una buena cantidad de datos de validación, elijo el λ con menor error de validación.

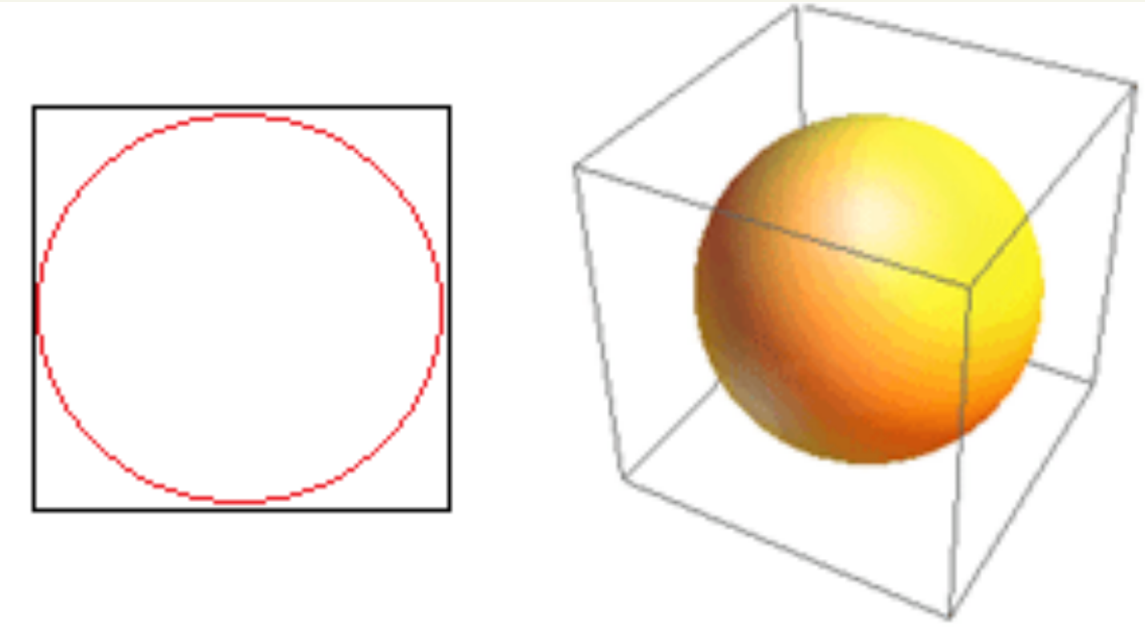
Leave-one-out cross-validation (LOOCV)

Si tengo pocos datos no puedo tener un conjunto de datos de validación suficientemente rico. Entonces entreno con todas las muestras menos una y valido con la última. Luego repito esto con cada muestra y promedio.

K-Fold

Separo en K subgrupos de $\frac{n}{K}$ muestras cada uno. Entreno con $K-1$ grupos y testeo con el último. Luego repito esto con cada grupo y promedio.

La maldición de la dimensionalidad



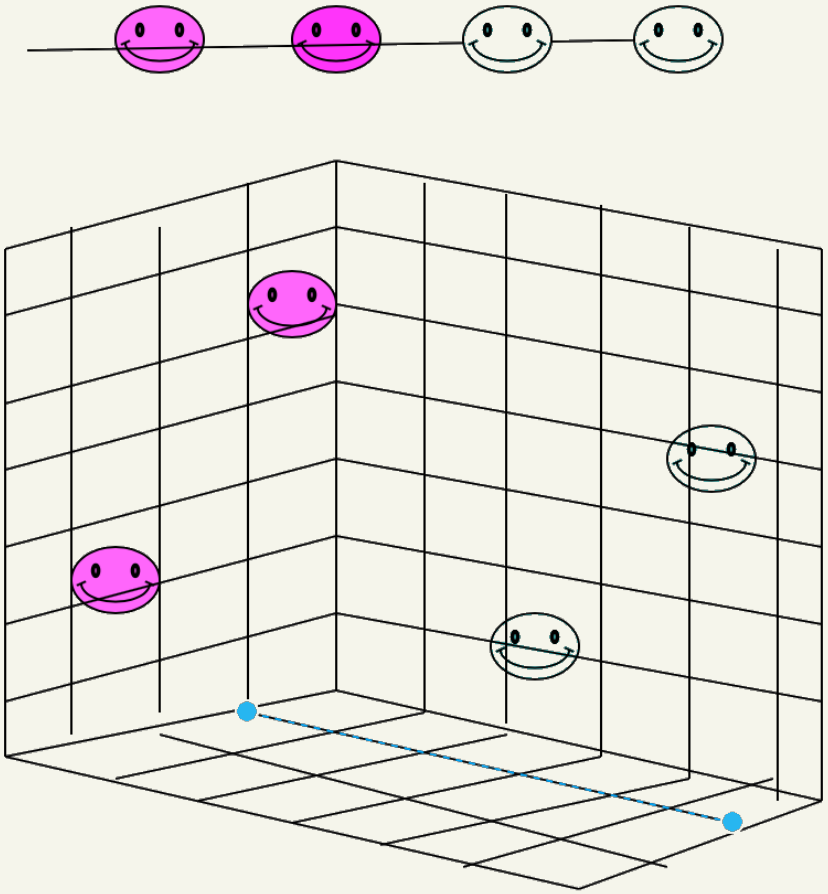
- 2d: $\frac{\pi r^2}{(2r)^2} \approx 78.5\%$
- 3d: $\frac{\frac{4}{3}\pi r^3}{(2r)^3} \approx 52.3\%$
- 10d: $\frac{\frac{r^{10}}{5!}\pi^5}{(2r)^{10}} \approx 0.25\%$

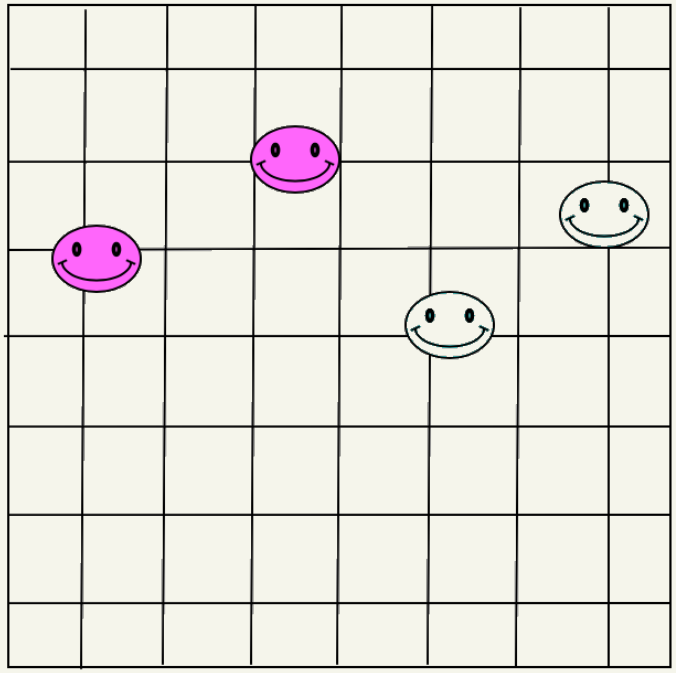
En grandes dimensiones:

- Los puntos están muy lejos.
- Las estructuras son muy sparce.
- La distancia euclidea no es buena métrica.
- La "necesidad" de muestras crece exponencialmente con la dimensión.

La maldición de la dimensionalidad

La maldición aplica a los hiperparámetros

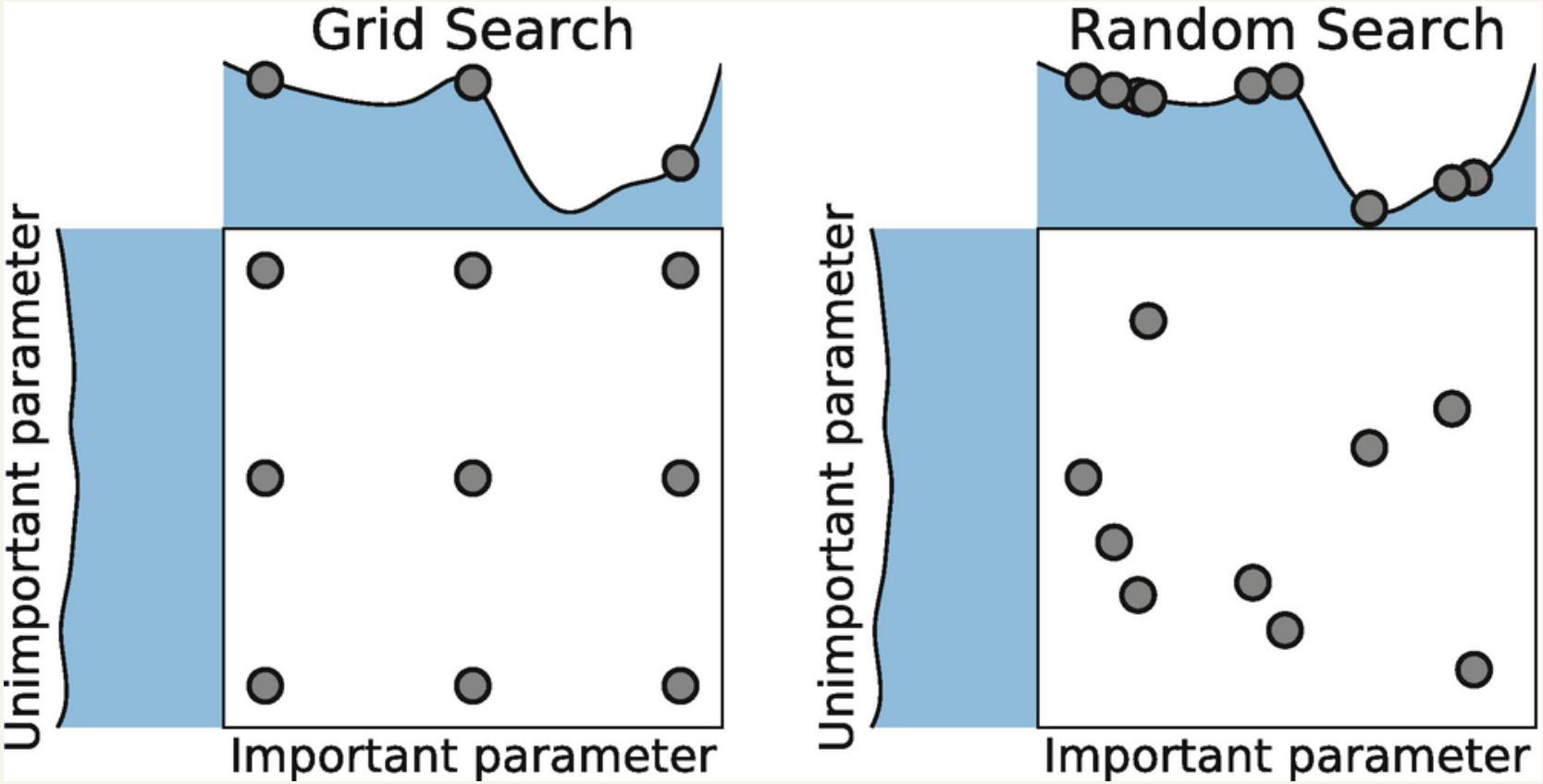




La necesidad de pruebas crece exponencialmente con la cantidad de hiperparámetros!

Búsqueda aleatoria

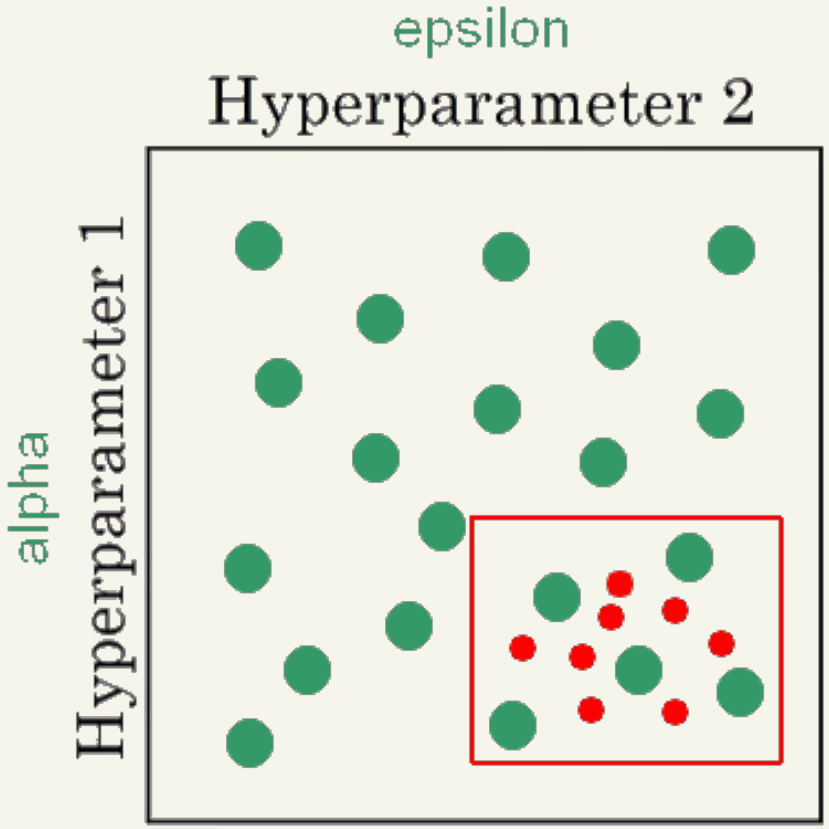
No todos los hiperparámetros son igual de importantes



Random search nos permite variar muchas veces todos los parámetros.

Búsqueda aleatoria

Hacerlo por etapas permite aprovechar más las simulaciones.



Simulo unos pocos puntos, veo donde está andando mejor y vuelvo a simular dentro de ese entorno.