



UNIVERSIDAD DE CHILE

Deep Learning

Deeper, Better, _____, Stronger than Machine Learning

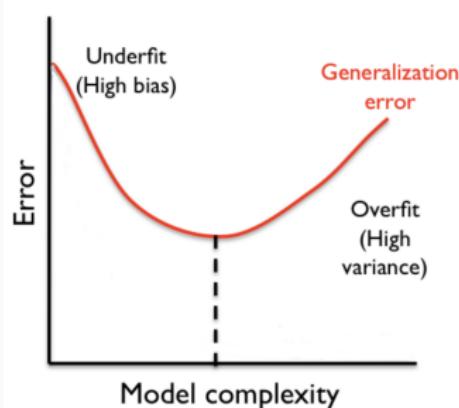
Valentin Barriere

Universidad de Chile – DCC

CC6204, Primavera 2024

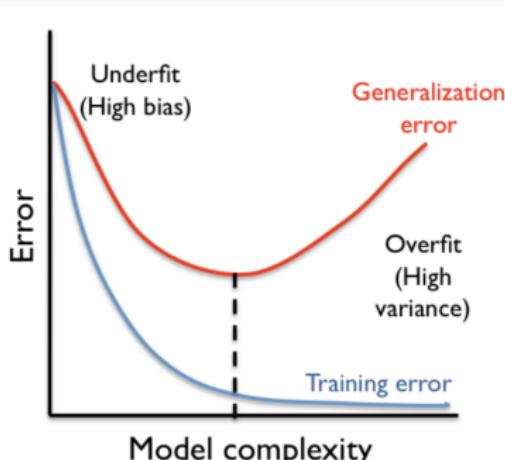
Regularization

Sobre-aprendizaje y sub-aprendizaje



- Según la complejidad del modelo (por ejemplo, tiempo de entrenamiento) se observa un comportamiento diferente
- Los modelos poco complejos son aprendidos fácilmente pero el error de aproximación puede ser grande (sub-aprendizaje)
- Los modelos muy complejos pueden tener el objetivo correcto pero un gran error de estimación (sobre-aprendizaje)

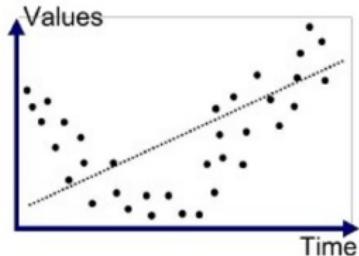
Sobre-aprendizaje: Problema



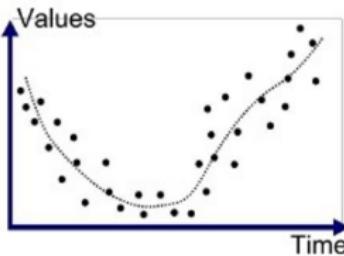
Error y riesgos

- El **riesgo empírico** (error en el conjunto de entrenamiento) disminuye con el aumento de la complejidad del modelo
- El **riesgo real** (error en observaciones de un nuevo conjunto) es muy diferente.
¡Tenemos un **problema de generalización!**
- Sobre-aprendizaje : los parámetros aprendidos son demasiado específicos para el conjunto de entrenamiento
- Se debe usar un criterio diferente al error en el conjunto de entrenamiento

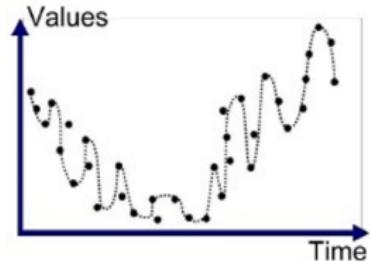
Sobre-aprendizaje: Complejidad



Underfitted



Good Fit/Robust



Overfitted

Complejidad

- Si el modelo es demasiado simple, entonces ya no sigue los datos
- Si el modelo es demasiado complejo, el modelo aprende todas las irregularidades del conjunto de datos \mathcal{D}_n : el objetivo no es memorizar los datos de entrenamiento, sino generalizar los nuevos datos
- Ejemplo : si el modelo es el de la curva del medio más una componente de ruido no considerada en las variables, el modelo de la derecha aprende ese ruido

Sobre-aprendizaje: Regularización

Una solución para combatir el problema de no generalización:
regularización

Principio

- El riesgo empírico de un estimador seleccionado de una familia de funciones dadas los datos está sesgado
- Al agregar una penalización en relación con la complejidad del modelo f_θ , podemos disminuirla y reducir el sobre-aprendizaje:

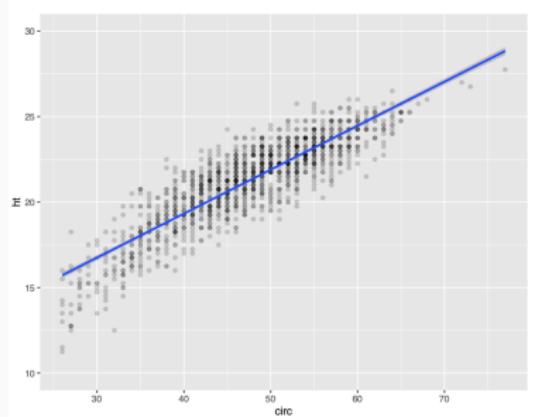
$$\mathcal{R}_n(f_\theta) \rightarrow \mathcal{R}_n(f_\theta) + pen(\theta)$$

- Entonces, en el riesgo:

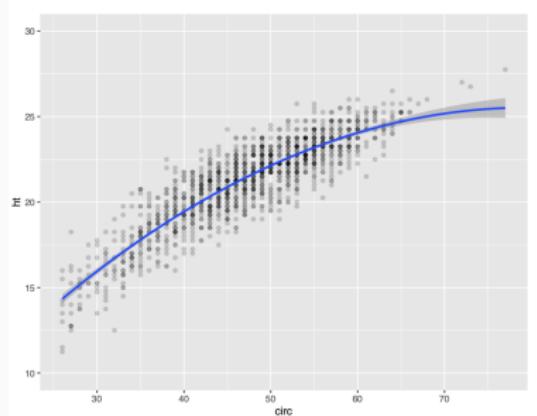
$$\arg \min_{f_\theta, \theta \in \Theta} \frac{1}{n} \sum_{i=1}^n \ell(Y_i, f_\theta(\mathbf{X}_i)) + pen(\theta)$$

Permite de usar un modelo grande, con menos sobre-aprendizaje

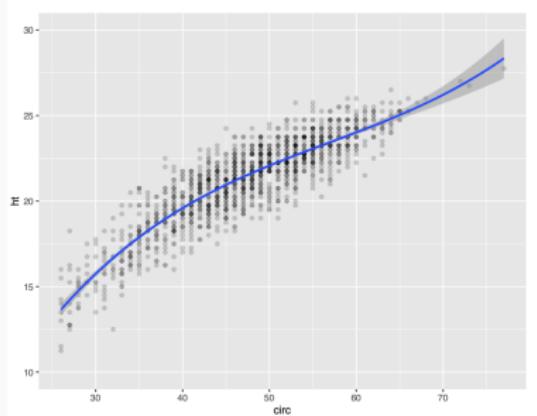
Sobre-aprendizaje: Regularización



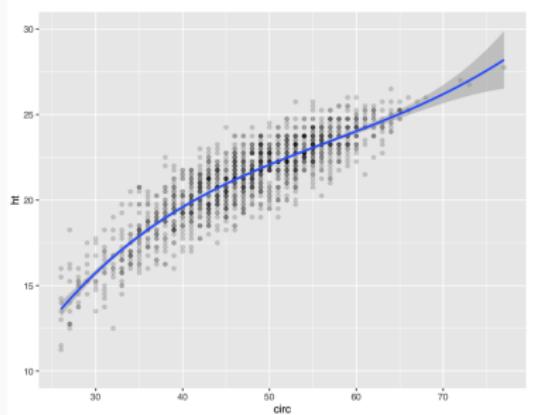
Sobre-aprendizaje: Regularización



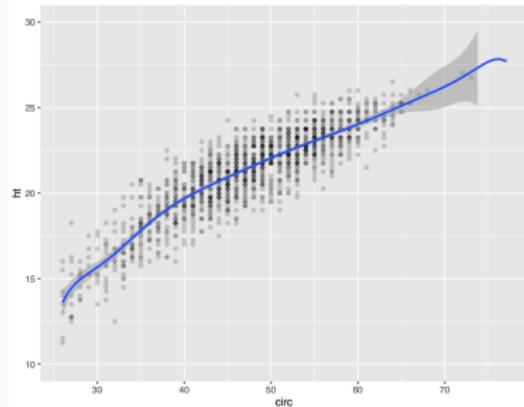
Sobre-aprendizaje: Regularización



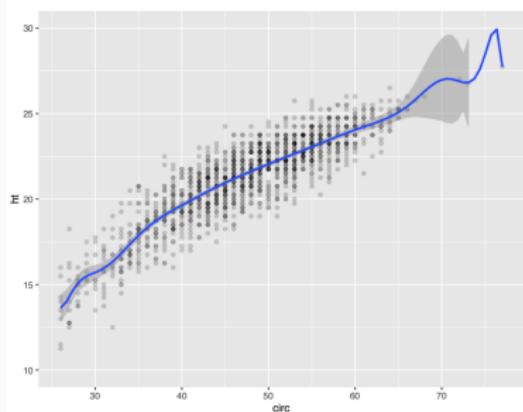
Sobre-aprendizaje: Regularización



Sobre-aprendizaje: Regularización



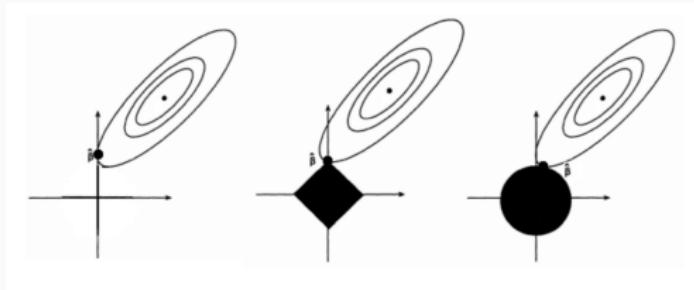
Sobre-aprendizaje: Regularización



Regularización

- Principio de parsimonia (navaja de Occam): cuanto más simple y funcione un modelo, mejor.
- Permite no tener en cuenta tantas particularidades de los datos.
- Intuición: disminuir la norma del modelo o su número de coeficientes, número de ramas del grafo (poda)

Sobre-aprendizaje: Regularización

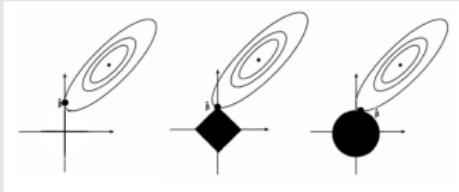


Regularizaciones clásicas

- AIC: $pen(\theta) = \lambda \|\theta\|_0$ (no convexa, parsimoniosa, poco utilizada)
- Ridge: $pen(\theta) = \lambda \|\theta\|_2$ (convexa, no parsimoniosa)
- Lasso: $pen(\theta) = \lambda \|\theta\|_1$ (convexa, parsimoniosa)
- Elastic Net: $pen(\theta) = \lambda_1 \|\theta\|_1 + \lambda_2 \|\theta\|_2$ (convexa, parsimoniosa)
- Optimización simple si el costo (la regularización) es convexo
- **Necesidad de especificar los λ** , que se convierten en nuevos hiperparámetros

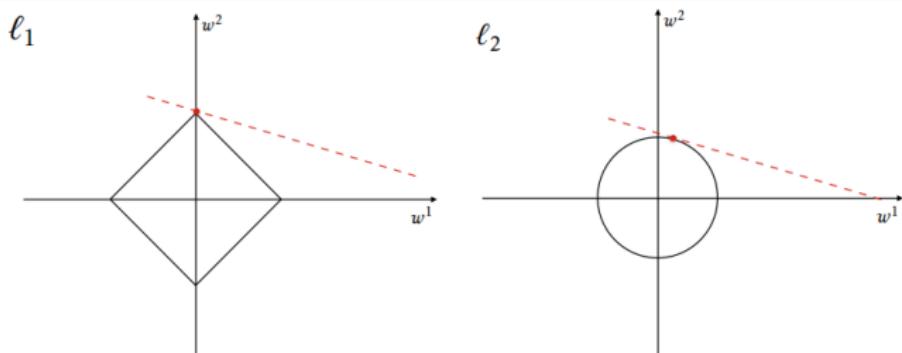
Intuición de la parsimonia

Lasso induce parsimonia

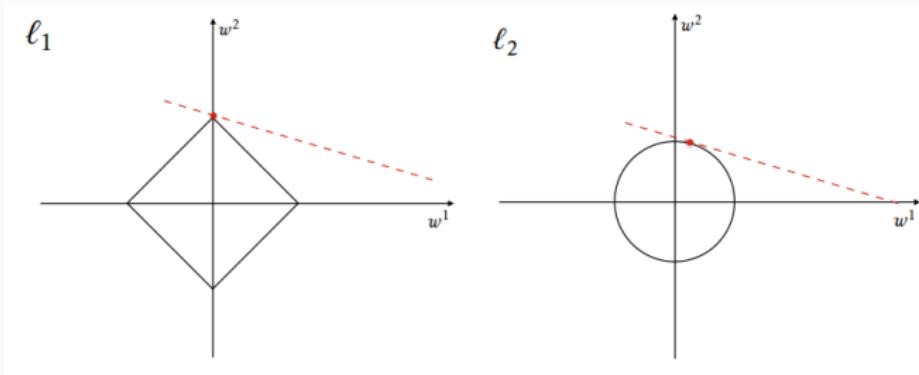


En negro $\mathcal{B}^n = \{\mathbf{x}/\mathbf{x} \in \mathbb{R}^d, \text{ y } \|\mathbf{x}\|_n < 1\}$ para $n = 0, 1, 2$ en \mathbb{R}^2

- En dimensiones grandes, la mayoría de \mathcal{B}^1 se concentra en los ejes. Esto equivale a tener valores nulos para otros ejes.



Intuición de la parsimonia



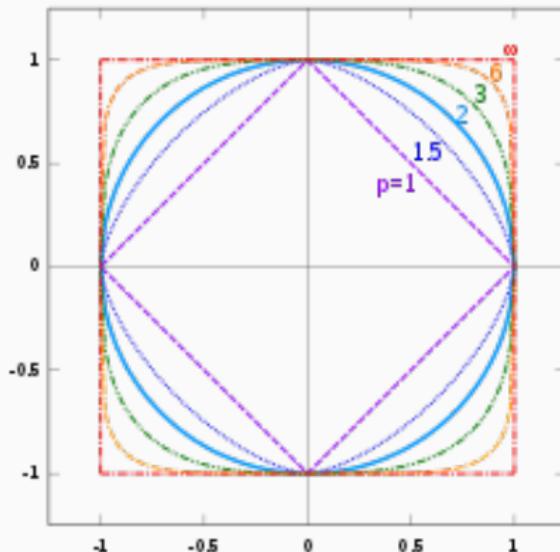
- La solución del problema se encuentra en la recta roja: cualquier punto de este satisfaciente
- Si tenemos una regularización, queremos la solución que nos da una valor mínima de la norma
- intuición: la norma ℓ_1 va a dar soluciones en los ejes, lo que significa en este caso el parámetro w^1 iguale a cero!

Intuición de la parsimonia : normas

Ejercicio: Hemos visto la bola 1, y la bola 2. ¿Que forma tiene la bola infinita?

Intuición de la parsimonia : normas

Ejercicio: Hemos visto la bola 1, y la bola 2. ¿Que forma tiene la bola infinita?

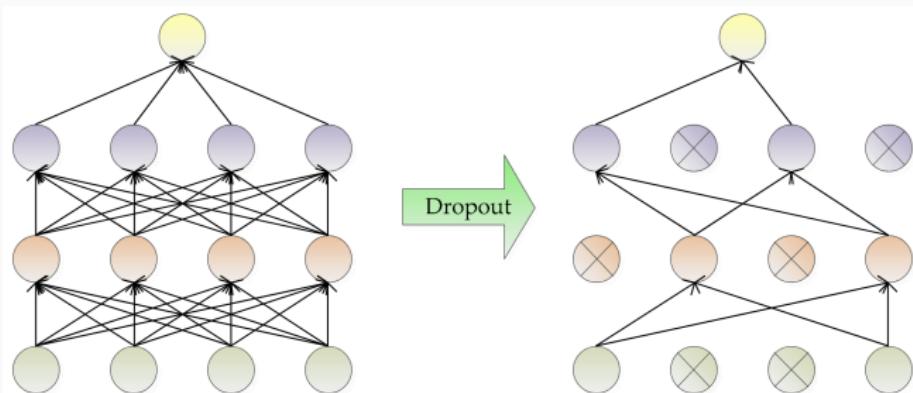


$$\|\mathbf{x}\|_n = \left(\sum_{i=1}^d |x_i|^n \right)^{\frac{1}{n}}, \text{ por eso } \|\mathbf{x}\|_\infty = \max_i |x_i|$$

Dropout: principle

What is Dropout?

- Dropout is a regularization technique used in neural networks to prevent overfitting, which was introduced by Geoffrey Hinton and his team in 2012.
- The idea is to randomly "drop out" (set to zero) a fraction of neurons **during the training process**.



At each training iteration, a random subset of neurons is ignored.

Dropout: principle

Why Use Dropout?

- It prevents the model from becoming too dependent on specific neurons, forcing the network to learn more robust features, and improves generalization to unseen data.
- At each iteration, the network effectively trains a different subset of the model, creating an ensemble effect

For patients with depression, these paths might represent negative thoughts or emotions that keep getting triggered.

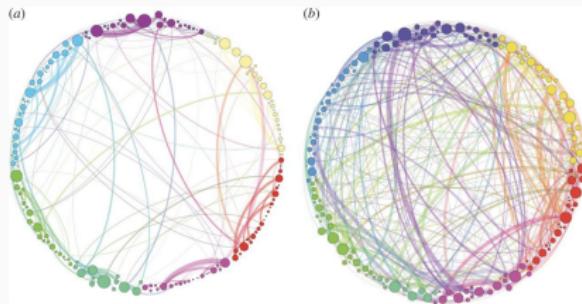
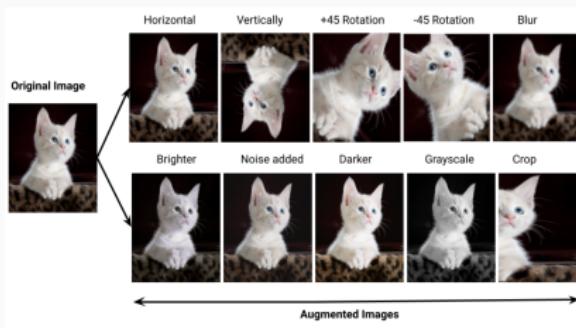


Figure 1: Visualization from Petri et al. (2014)

Data augmentation

Another solution is to simply use more data! But it can get costly to obtain, so you can artificially create some using data augmentation.

- More data always reduce overfitting, because we **introduce variability**
- A priori knowledge needs to be used in order to introduce meaningful, otherwise you perturbate the data distribution (north hemisphere satellite data faces always the sun the same way)
- In the case of images, typical transformation are: Small rotations, Flipping, Random crops, Color Jittering, Noise addition, ...



Data augmentation

Another solution is to simply use more data! But it can get costly to obtain, so you can artificially create some using data augmentation.

- More data always reduce overfitting, because we **introduce variability**
- A priori knowledge needs to be used in order to introduce meaningful, otherwise you perturbate the data distribution (north hemisphere satellite data faces always the sun the same way)
- In the case of images, typical transformation are: Small rotations, Flipping, Random crops, Color Jittering, Noise addition, ...
- In the case of texts: Gender swap, Random punctuation insertion, Lexicon-based perturbation (synonym/entities replacement), Back Translation, ...
- In the case of audio: Time Shifting, Pitch shifting, Reverberation, Amplitude scaling, Noise addition, ...

Audio examples: [original file](#), [with reverb](#), or [with reverb and noise](#).

Questions?

References i