



Deep Learning con Pytorch

Juan Pablo Morales
@juanpamf

Evaluar un modelo de Deep Learning



Recap

1. **Diseñamos nuestro algoritmo de aprendizaje**
 - a. Arquitectura (espacio de modelos)
 - b. Función de pérdida
 - c. Método de optimización
2. **Minimizamos la pérdida sobre el dataset**
 - a. Esto es equivalente a fijar el buen valor de los parámetros de la red

¿Nos asegura esto
una buena performance
sobre nuevos datos?

Fase de evaluación: Primera idea

- Separar dataset aleatoriamente en:
 - Entrenamiento (70%)
 - Aquí se realiza la minimización
 - Test (30%)
 - Aquí se evalúa el performance del modelo (tasa de error)
- ¿Qué hacemos con los hiperparámetros?

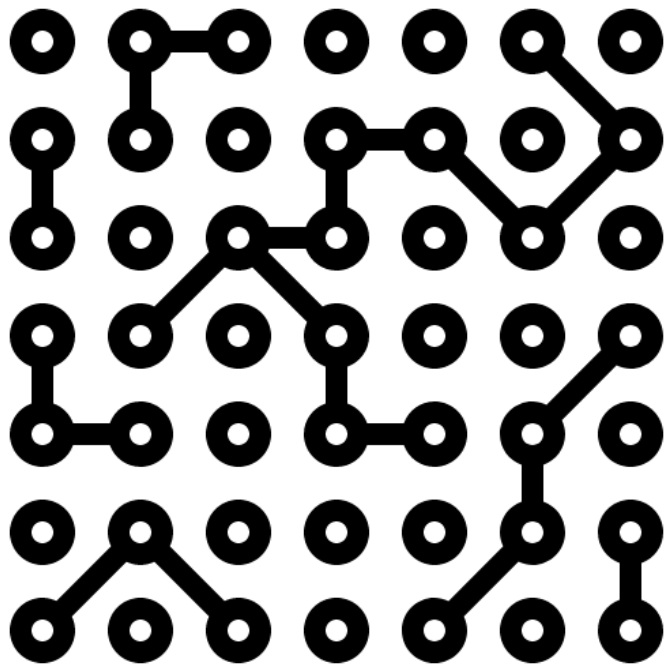
Si probamos varios modelos elegiremos un buen valor, pero nuestra estimación de performance estará sesgada.

Fase de evaluación: Segunda idea

Separar dataset aleatoriamente en:

- Entrenamiento (60%)
 - Aquí se realiza la minimización
- Validación (20%)
 - Con base a la tasa de error, elegimos el mejor valor de los hiperparámetros.
- Test (20%)
 - Aquí se evalúa la performance del modelo (Taza de error)

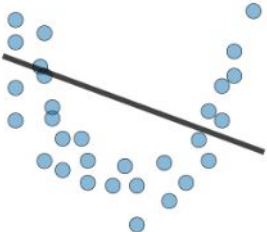
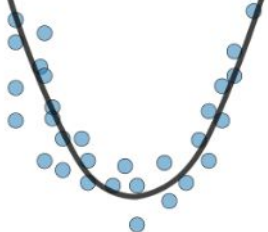

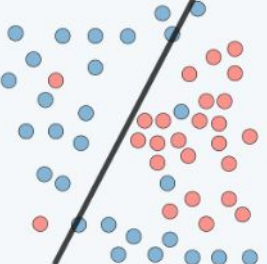
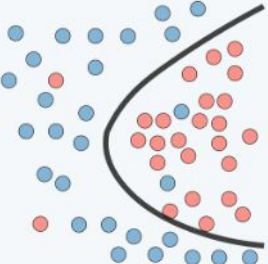
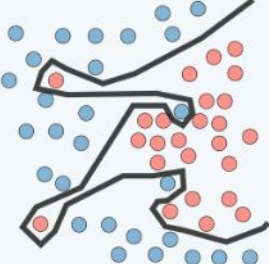
Fase de evaluación: Segunda idea



**En deep learning
(grandes datasets):**

- Entrenamiento: 98%
- Validación: 1%
- Test: 1%

Underfitting-Overfitting

	Underfitting	Just right	Overfitting
Symptoms	<ul style="list-style-type: none">• High training error• Training error close to test error• High bias	<ul style="list-style-type: none">• Training error slightly lower than test error	<ul style="list-style-type: none">• Very low training error• Training error much lower than test error• High variance
Regression illustration			
Classification illustration			
Deep learning illustration	