Aprendizaje Profundo

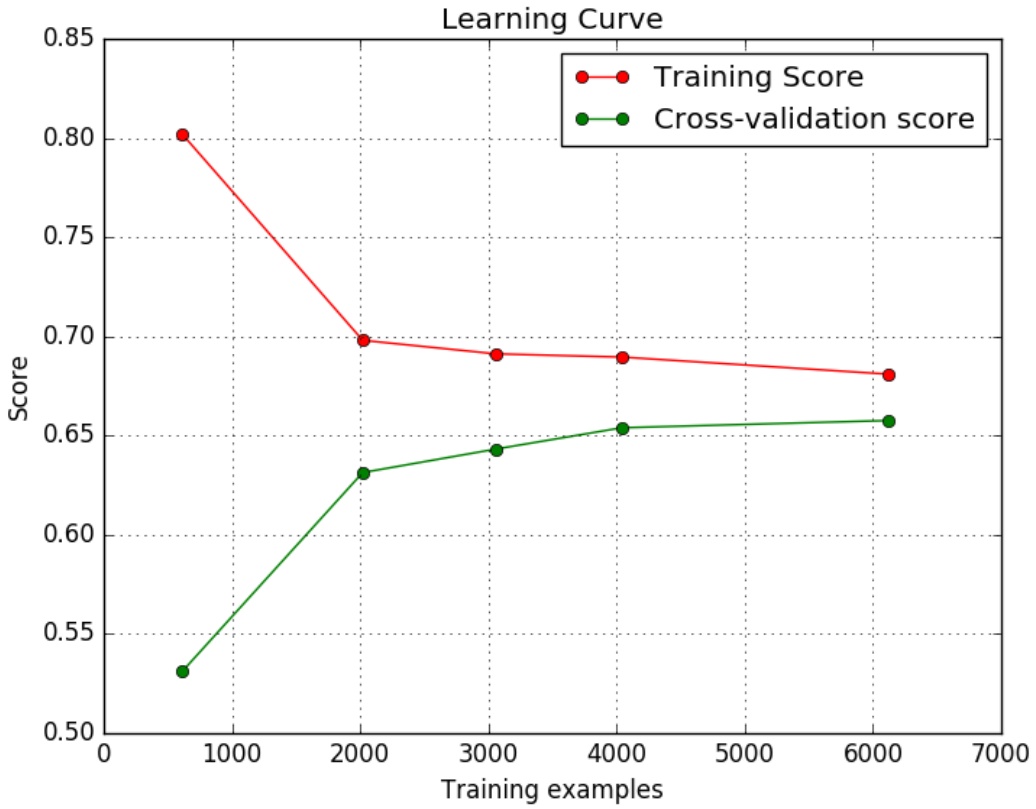
Facultad de Ingeniería
Universidad de Buenos Aires

Profesores:

Marcos Maillot
Antonio Zarauz
Gerardo Vilcamiza

Regularización

Learning curve



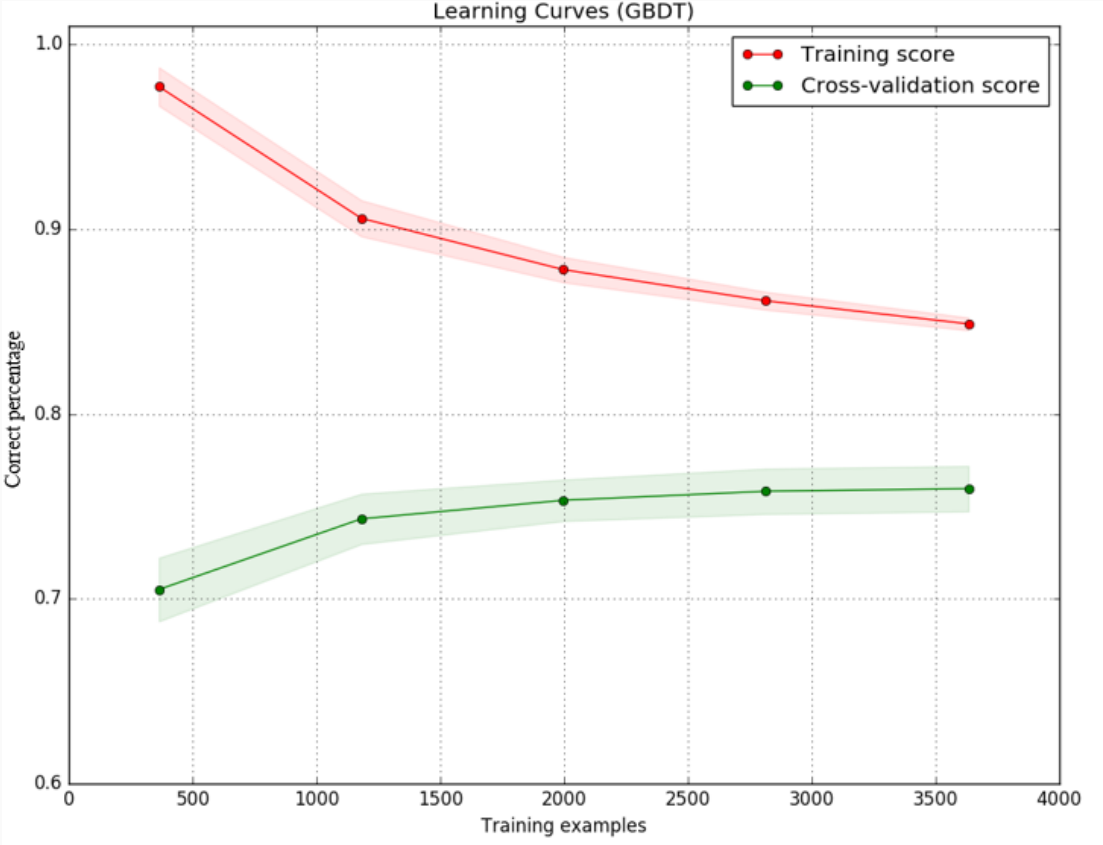
Underfitting

Gran desvío entre mi métrica ideal y el valor al cual tiende la learning curve

¿Qué puedo hacer?

- **Modelo**: aumentar la complejidad del modelo
- **Datos**: agregar más features al dataset

Learning curve



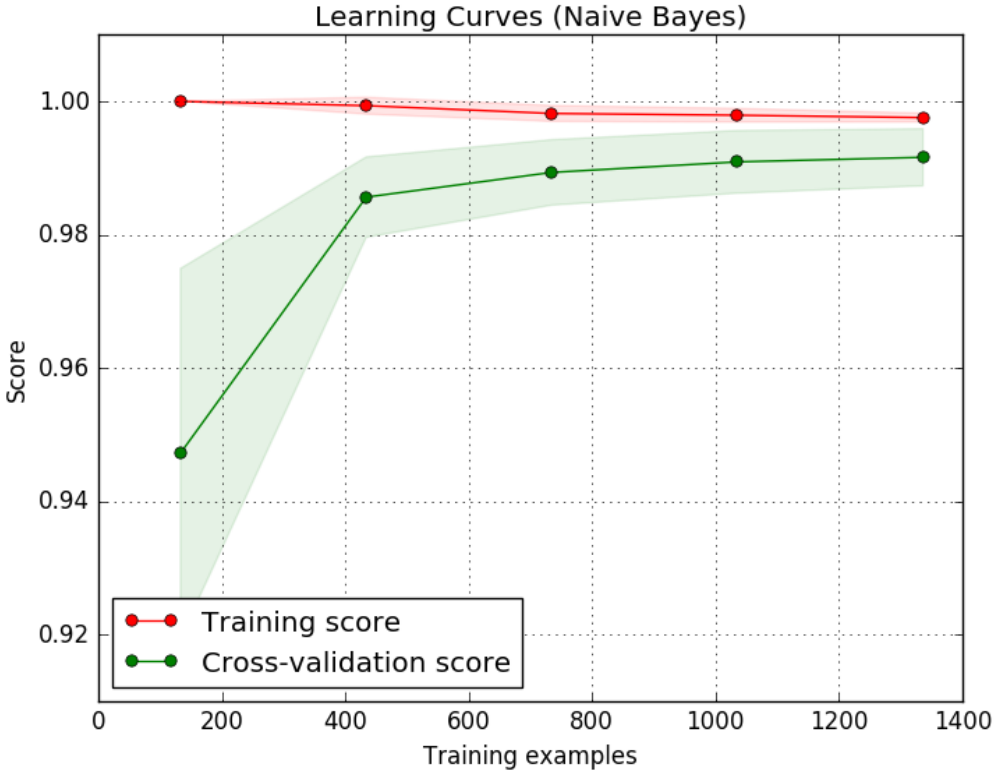
Overfitting

Gran desvío entre las métricas de entrenamiento y validación

¿Qué puedo hacer?

- **Modelo**: agregar regularización
- **Datos:** agregar más muestras

Learning curve



Fit perfecto

- Bajo GAP de la métrica de entrenamiento respecto a la métrica ideal
- Bajo GAP entre las métricas de entrenamiento y validación

Regularización

- **Técnicas que penalizan la función de costo con los valores de los pesos**
  - Regularización L2
  - Regularización L1
    - Regularizar
    - Bajar la complejidad del modelo
    - Selección de features

- **Bagging** (aplicable a modelos de ML)

- **Dropout** (aplicables a modelos de DL)

Regularización L2

$$\tilde{L}\left(\vec{W},\vec{x},\vec{y}\right) = L\left(\vec{W},\vec{x},\vec{y}\right) + \frac{\lambda}{2}\cdot W^T\cdot W \longrightarrow W^T\cdot W = w_1^2 + w_2^2 + \ldots + w_k^2 = \|W\|_2^2$$

1. ¿Cómo afecta a los pesos?
2. ¿Cómo afecta al modelo?

$$(1)\ \vec{\nabla}_W\left(\tilde{L}\right) = \vec{\nabla}_W(L) + \vec{\nabla}_W\left(\frac{\lambda}{2}\cdot W^T\cdot W\right)$$

$$= \vec{\nabla}_W(L) + \lambda\cdot\vec{W},\ \lambda \text{ hiperparámetro}$$

$$\vec{W} \leftarrow \vec{W} - \alpha\left(\lambda\cdot\vec{W} + \vec{\nabla}_W(L)\right)$$

$$\vec{W} \leftarrow \vec{W} - \alpha\cdot\lambda\cdot\vec{W} - \alpha\cdot\vec{\nabla}_W$$

$$\vec{W} \leftarrow (1-\lambda\cdot\alpha)\cdot\vec{W} - \alpha\cdot\vec{\nabla}_W$$

En cada update W decrece en un factor (1-αλ)

Regularización L2

2. ¿Cómo afecta al modelo?

Por simplicidad consideremos una regresión lineal

$$\text{Solución cerrada}: \vec{W} = \left(X^T \cdot X\right)^{-1} \cdot X^T \cdot y \rightarrow L = MSE$$

$$\text{Regularización L2}: \vec{W} = \left(X^T \cdot X + \lambda \cdot I\right)^{-1} \cdot X^T \cdot y \rightarrow \widetilde{L} = MSE + \lambda \cdot \|W\|_2^2$$

Matriz de covarianza

Suma valores positivos a la matriz de covarianza en la dirección de mayor varianza (diagonal principal). Estoy agregando mayor variabilidad al dataset de entrada.

Regularización L1

$$\tilde{L}\left(\vec{W},\vec{x},\vec{y}\right) = L\left(\vec{W},\vec{x},\vec{y}\right) + \lambda \cdot \|W\|_1 \longrightarrow \|W\|_1 = |w_1| + |w_2| + \ldots + |w_k|$$

$$\vec{\nabla}_W\left(\tilde{L}\right) = \vec{\nabla}_W(L) + \lambda \cdot \vec{\nabla}_W(|w_1| + |w_2| + \ldots + |w_k|)$$

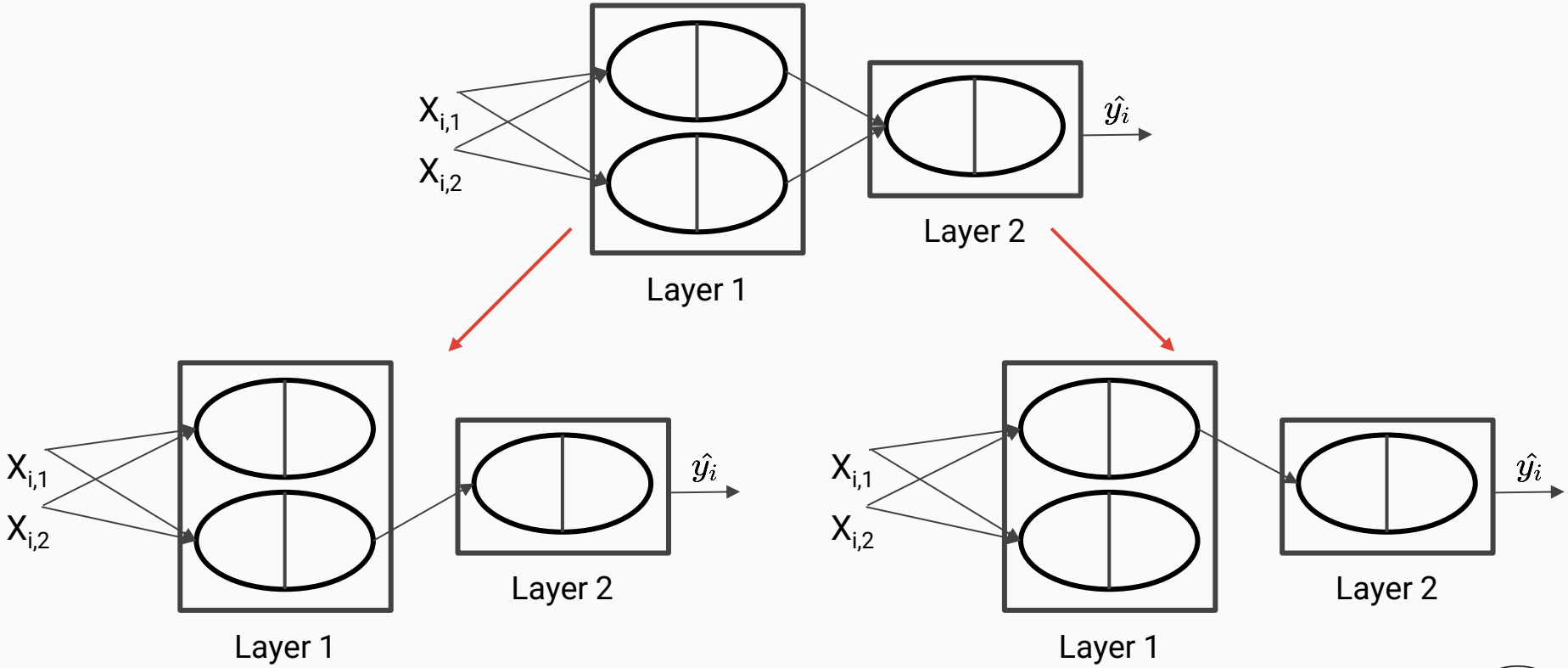
$$\vec{\nabla}_W(f) = \begin{pmatrix} \partial f/\partial w_1 \\ \partial f/\partial w_2 \\ \ldots \\ \partial f/\partial w_k \end{pmatrix} = \begin{pmatrix} \text{signo}(w_1) \\ \text{signo}(w_2) \\ \ldots \\ \text{signo}(w_k) \end{pmatrix}; \text{signo}(w_i) = \begin{cases} 1 & w_i \geq 0 \\ -1 & w_i < 0 \end{cases}$$

$$\vec{W} \leftarrow \vec{W} - \alpha\left(\lambda \cdot \text{signo}\left(\vec{W}\right) + \vec{\nabla}_W(L)\right)$$

$$\vec{W} \leftarrow \vec{W} - \alpha \cdot \lambda \cdot \text{signo}\left(\vec{W}\right) - \alpha \cdot \vec{\nabla}_W$$

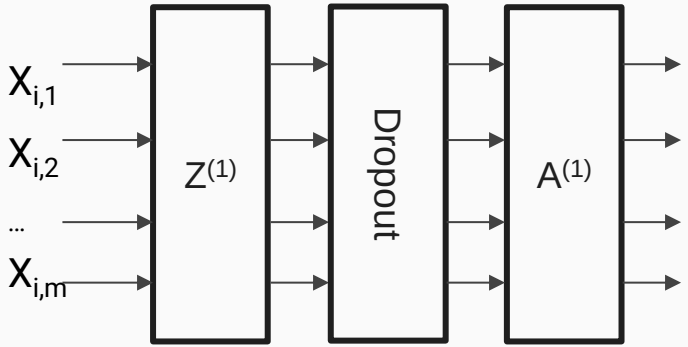
Reduzco W en forma constante

Dropout



Dropout

```
for epoch in epochs:
  for batch in batches:
    ŷ = f(NNet) → NNet tiene capas de dropout
    *Loss
    *Backward
    *Update
```



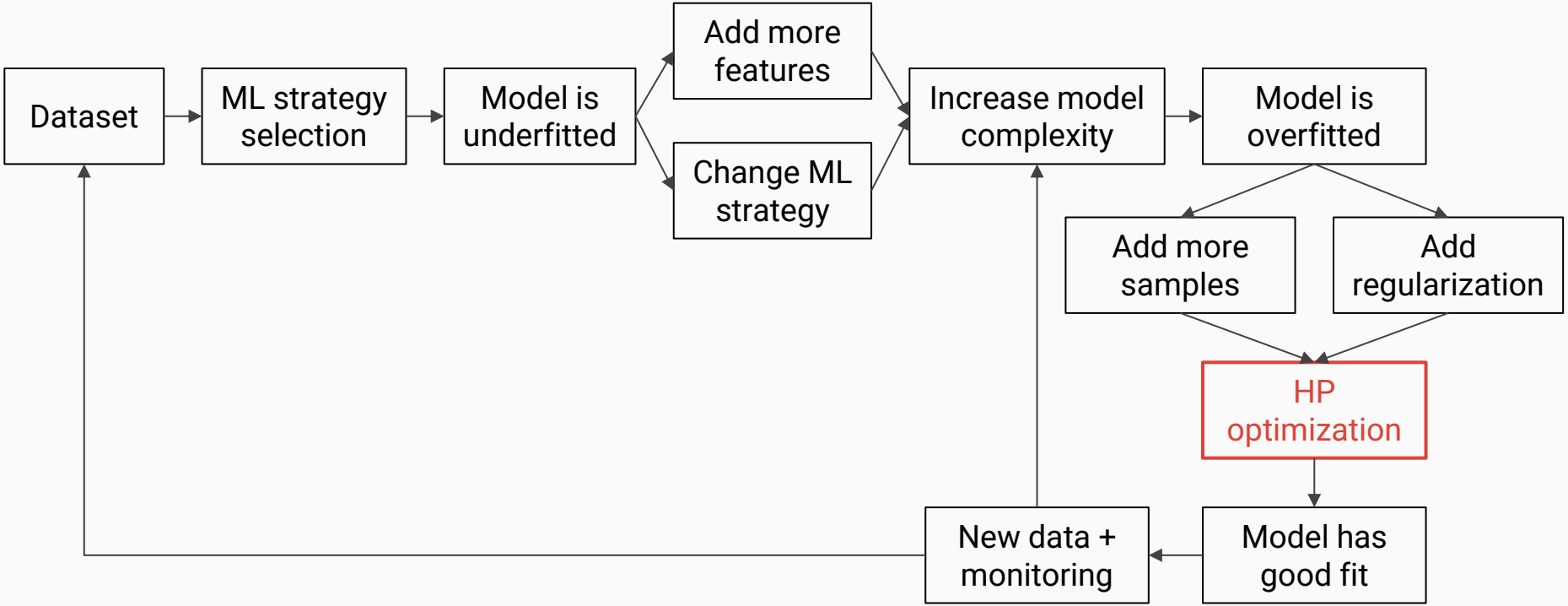
Dropout me “apaga” aleatoriamente neuronas con una probabilidad p (normalmente 20%)

- Las capas de dropout se utilizan únicamente en entrenamiento.
- En inferencia dropout se desactiva y se utilizan el 100% de las neuronas

Hyperparameter tunning*

*Diapositivas tomadas de presentación de profesor Ezequiel Esposito

ML Lifecycle



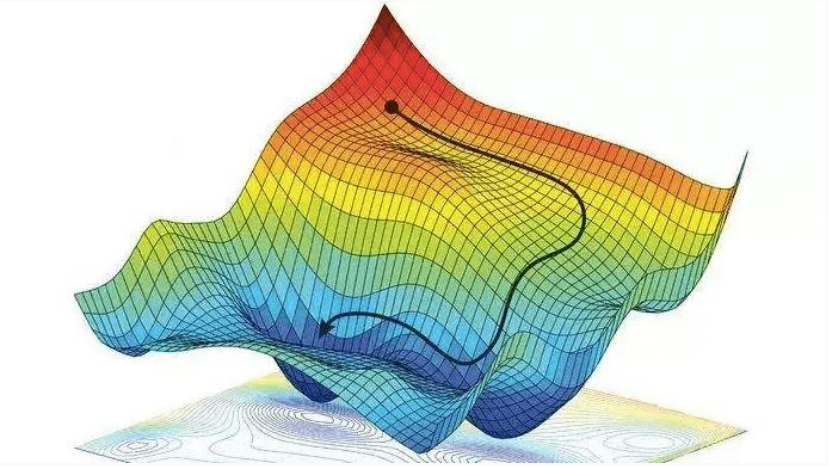
HP Tunning

- Los HP son parámetros que no pueden ser aprendidos como parte del algoritmo de optimización.

- El objetivo de HP tunning es encontrar los valores de los hiperparámetros del modelo y del algoritmo de optimización que obtengan la mejor métrica/pérdida sobre el set de datos de validación y, con optimismo, sobre el set de test.

- HP tunning es un problema de búsqueda, en una dimensión k.

HP Tunning



Objetivo: encontrar el loss mínimo para un conjunto de HP

¿Puedo aplicar GD? **No**

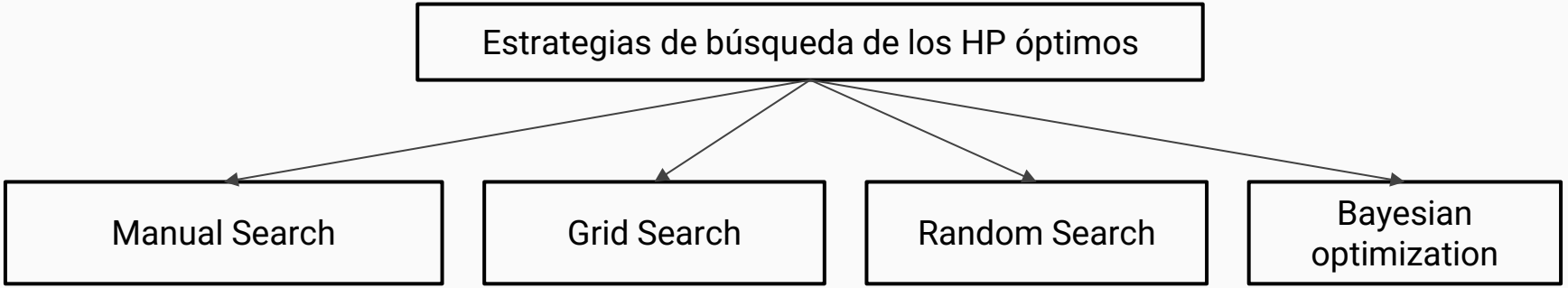
No conozco la función que me relaciona los HP con las métricas/error que voy a obtener al entrenar la red neuronal utilizando esos HP.

Selecciono HP, entreno el modelo, calculo el error / métrica y obtengo un punto de la curva. Debo repetir el proceso muchas veces con diferentes puntos para obtener el gráfico de la curva.

¡Computacionalmente intenso!

HP tunning se resuelve como un problema de búsqueda

HP Tunning



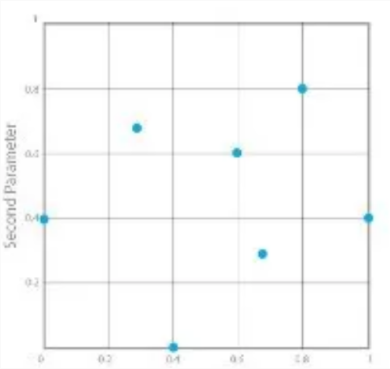
Elegimos un set de valores y manualmente los corregimos utilizando aproximaciones sucesivas. Es recomendable comenzar con los HP que más impactan (learning rate) y comenzar con los valores extremos.

Construyo una grilla equidistante de valores de HP y exploro todo el espacio. Ventaja: Paralelizable. Desventaja: CPU intensivo.

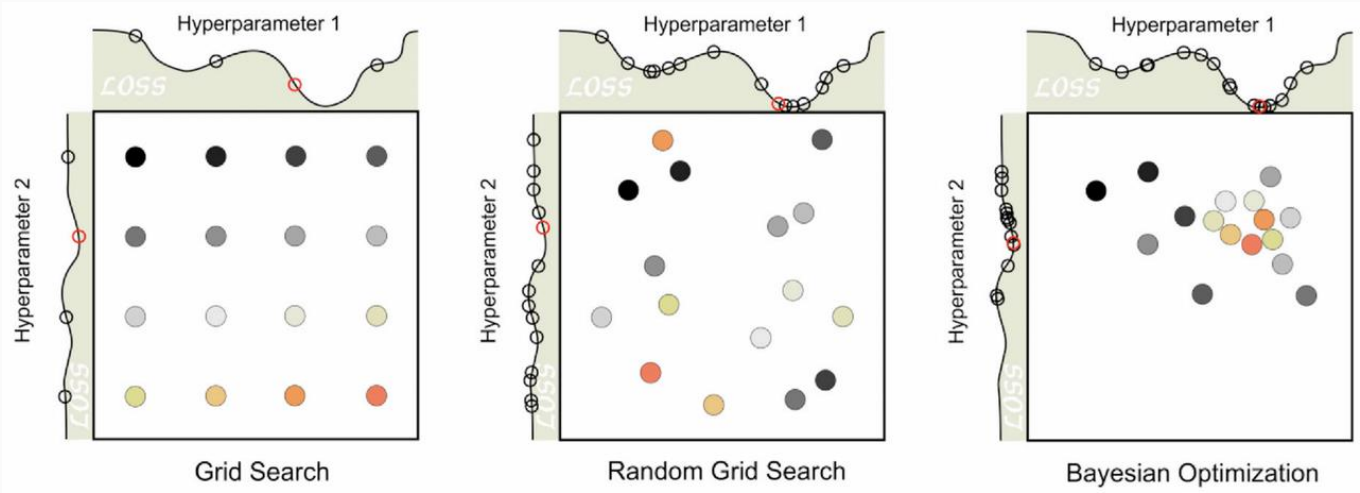
Mejora a grid search. En cada iteración varío todos los HP para realizar una mejor exploración del espacio.

Búsqueda inteligente. Utiliza los resultados de búsquedas anteriores para determinar el siguiente set de valores a probar.

HP Tunning



Manual search

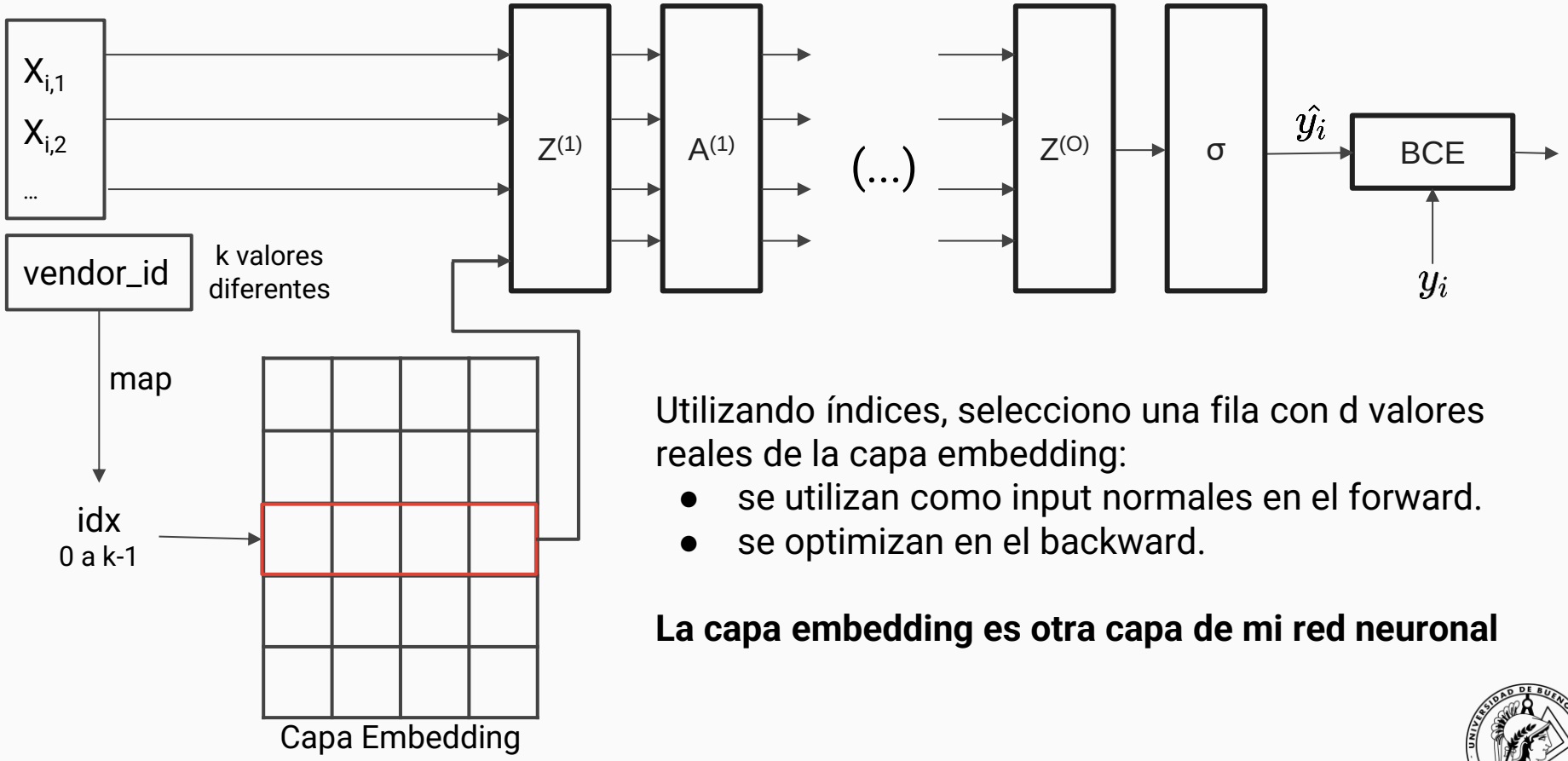


HP Tunning

- Comenzar con una búsqueda manual para reducir el espacio de búsqueda.
- Reservar una parte del dataset que conserve una distribución de datos similar al dataset original para validación.
- En DL no utilizar k-folds para ajustar los HP, usar validación cruzada.
- En DL usar optimización bayesiana.
- No desarrollar los algoritmos de búsqueda, utilizar librerías como Ray.
- Utilizar estrategias de early-stopping permite ahorrar tiempo.
- HP tunning no es la mejor técnica inicial para mejorar las métricas. En un escenario real inicialmente es conveniente invertir tiempo en mejorar el dataset.

Embedding Layer

Embedding Layer



Utilizando índices, selecciono una fila con d valores reales de la capa embedding:

- se utilizan como input normales en el forward.
- se optimizan en el backward.

La capa embedding es otra capa de mi red neuronal

Notebook: Embeddings en práctica

EJERCICIO

1. Terminar el entrenamiento del modelo con embeddings.
2. Comparar las métricas obtenidas con el modelo sin embeddings.