FACULTAD DE INGENIERIA
Universidad de Buenos Aires

Aprendizaje de máquina II

Carrera de Especialización en Inteligencia Artificial

Alex Barria

Especialista en Inteligencia artificial

Ingeniero electrónico

Contacto: alexbarria_14@hotmail.com

alexbarria14@gmail.com

SLACK

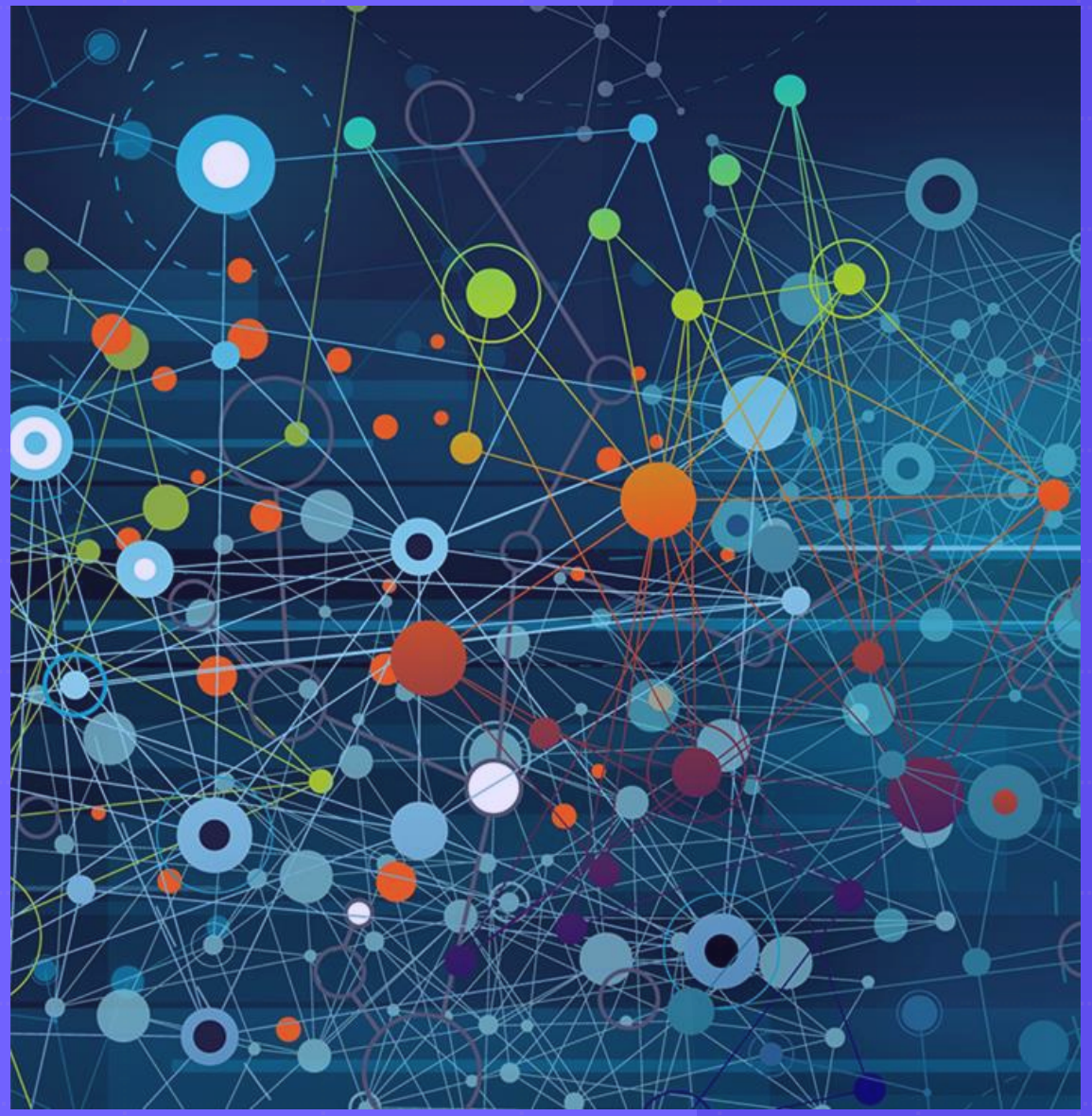
Docentes

Introducción a AMq2

Orientada a la parte operativa del aprendizaje automático (**MLOps**).

Empezar a interactuar con algunos componentes de **cloud** que son utilizados en el día a día de distintas empresas que trabajan con datos.

Brindar herramientas/habilidades de desarrollo de software que sean directamente **aplicables a un entorno de trabajo**.



Agenda

- Métodos de ensamble
- Optimización de hiperparámetros
- Roles dentro de la industria de datos
- Buenas prácticas de programación
- Pruebas unitarias
- Versionado de modelos de ML con MLflow
- Versionado de código con Git y GitHub
- Databricks como entorno de desarrollo
- Spark: procesamiento de grandes cantidades de datos
- Monitoreo de modelos y datasets
- Explicabilidad de modelos de ML

Forma de evaluación

Cuestionarios en **Google Forms** con algunas preguntas relacionadas al contenido de la clase previa – El objetivo de estas preguntas es repasar el contenido y darle continuidad a los temas de la materia.

Trabajo integrador final – Poner en práctica la mayor cantidad de conceptos posibles aplicados a un caso de la vida real.

Métodos de ensamble

Métodos de ensamble

¿Qué son?

Supongamos que queremos responder alguna pregunta. Si le hacemos esa pregunta a miles de personas, recolectamos las respuestas y las promediamos, es altamente probable que el resultado sea más acertado que la respuesta de un único experto.

→ *Wisdom of the crowd* (sabiduría de la multitud)

Si llevamos esta idea al campo del aprendizaje automático, al promediar la salida de un conjunto de predictores es altamente probable que obtengamos mejores métricas que considerando la salida de uno solo de ellos.
Dado que un conjunto de predictores se conoce como **ensamble**, estos métodos comúnmente se llaman **métodos de ensamble**.

Métodos de ensamble

Por ejemplo, podríamos entrenar un grupo de árboles de decisión, cada uno en una parte diferente del conjunto de entrenamiento, generar predicciones con cada uno de los árboles individuales y luego predecir la clase que tenga mayor cantidad de votos.

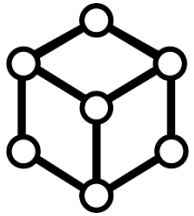
¿Cuándo usarlos?

Es más frecuente utilizar métodos de ensamble cuando ya probamos algunas arquitecturas más simples y para poder mejorar las métricas obtenidas combinamos los predictores ya desarrollados en un ensamble.

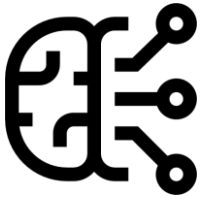
Voting classifiers

Regresión logística

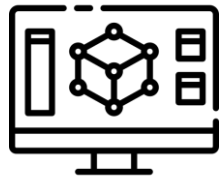
Clasificador SVM



Clasificador red neuronal



Otra arquitectura



Dataset

Supongamos que entrenamos estos modelos y que cada uno logra un accuracy de 80%

Hard voting classifiers

Por ejemplo, podríamos entrenar un grupo de árboles de decisión, cada uno en una parte diferente del conjunto de entrenamiento, generar predicciones con cada árbol individual y luego predecir la clase que tenga mayor cantidad de votos.

Soft voting classifiers

En este caso, si los modelos entrenados tienen la posibilidad de estimar la probabilidad por clase (en algunas librerías los modelos tienen un método llamado predict_proba()) lo que se hace es calcular la posibilidad promedio de cada clase sobre la cantidad total de modelos.

Hard voting classifiers

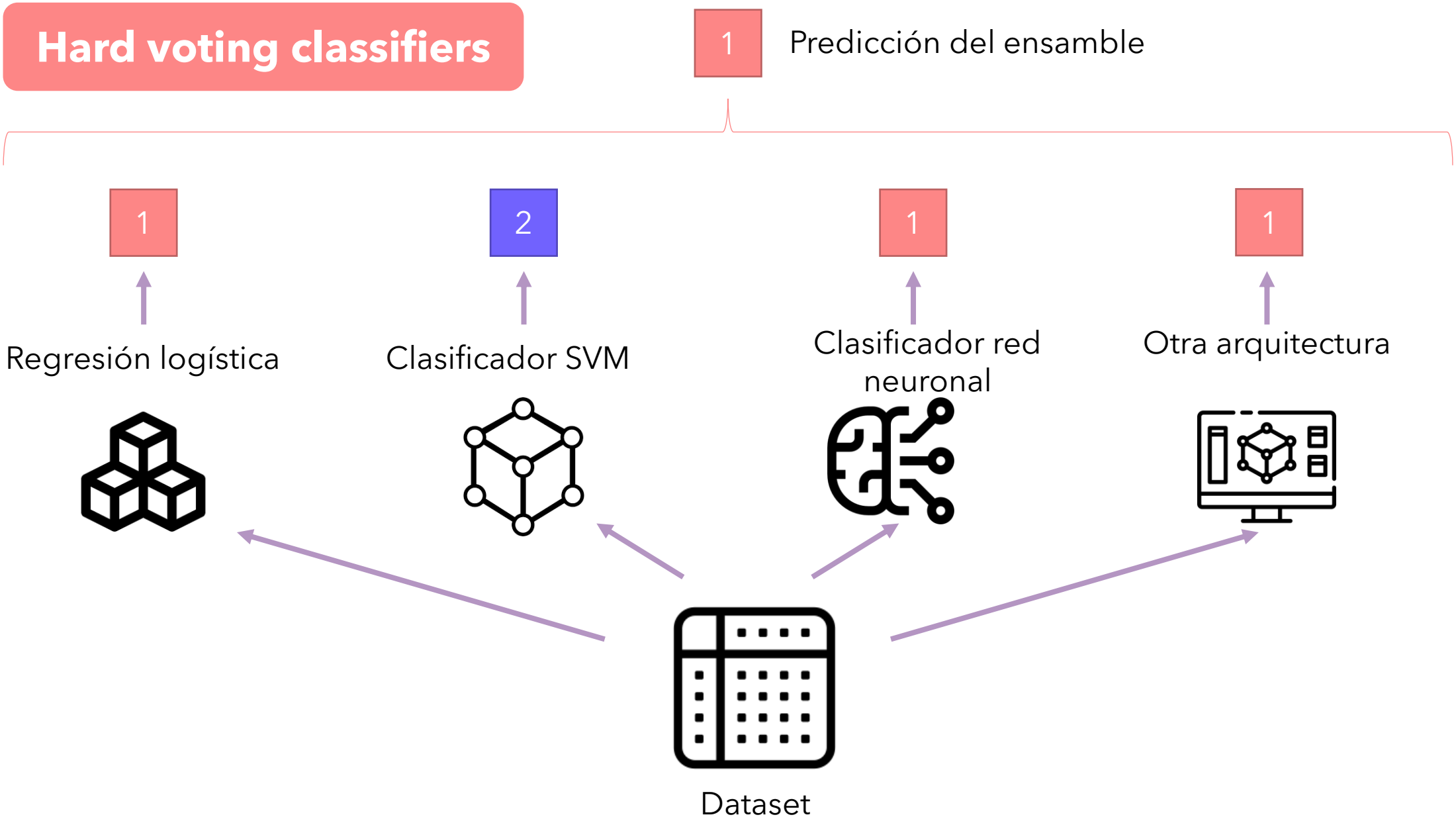
Por ejemplo, podríamos entrenar un grupo de árboles de decisión, cada uno en una parte diferente del conjunto de entrenamiento, generar predicciones con cada árbol individual y luego predecir la clase que tenga mayor cantidad de votos.

Soft voting classifiers

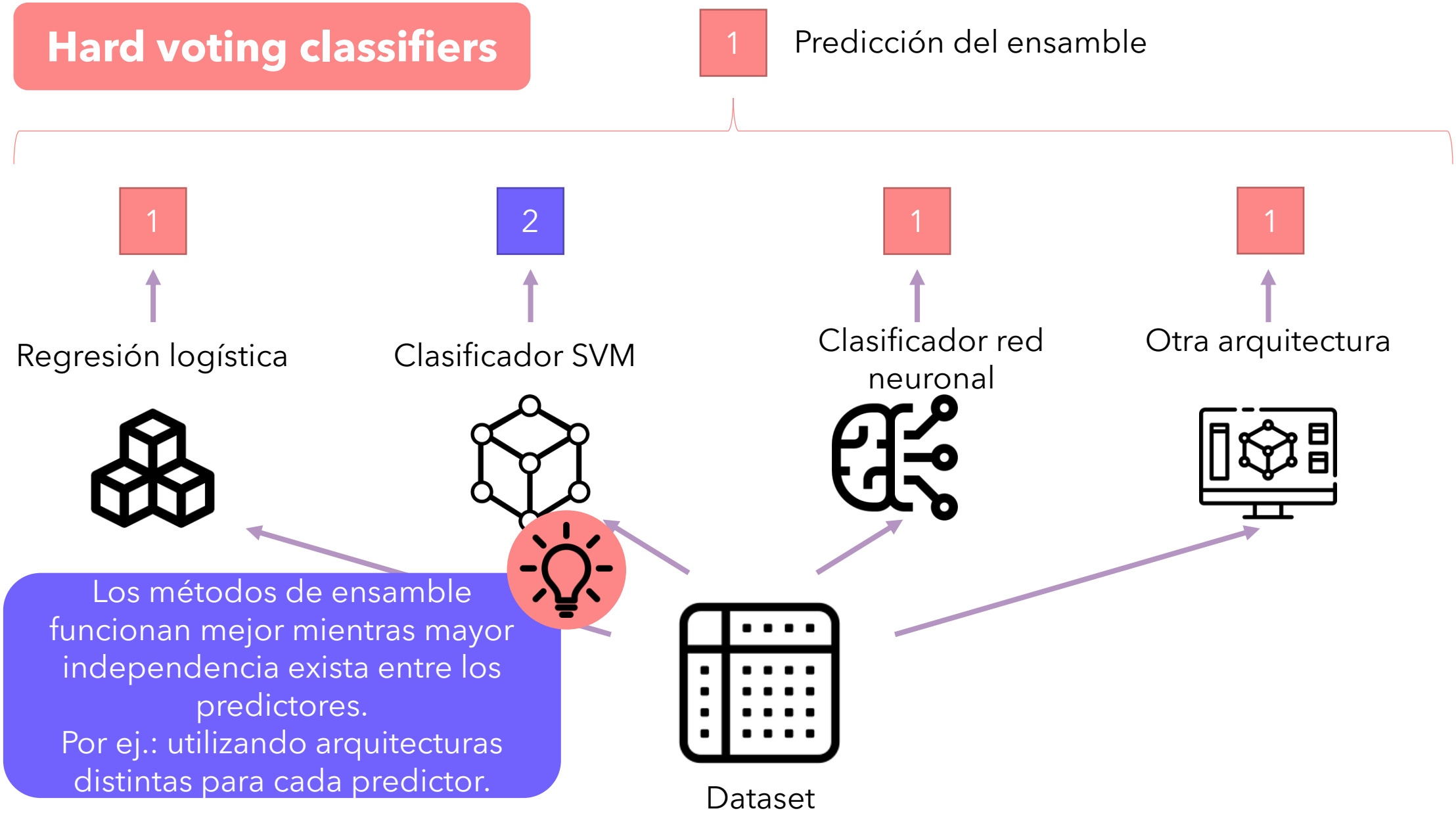
En este caso, si los modelos entrenados tienen la posibilidad de estimar la probabilidad por clase (en algunas librerías los modelos tienen un método llamado predict_proba()) lo que se hace es calcular la posibilidad promedio de cada clase sobre la cantidad total de modelos.

Una ventaja de esta manera de estimar las clases es que se le da más peso a los votos que tienen más confianza en su decisión.

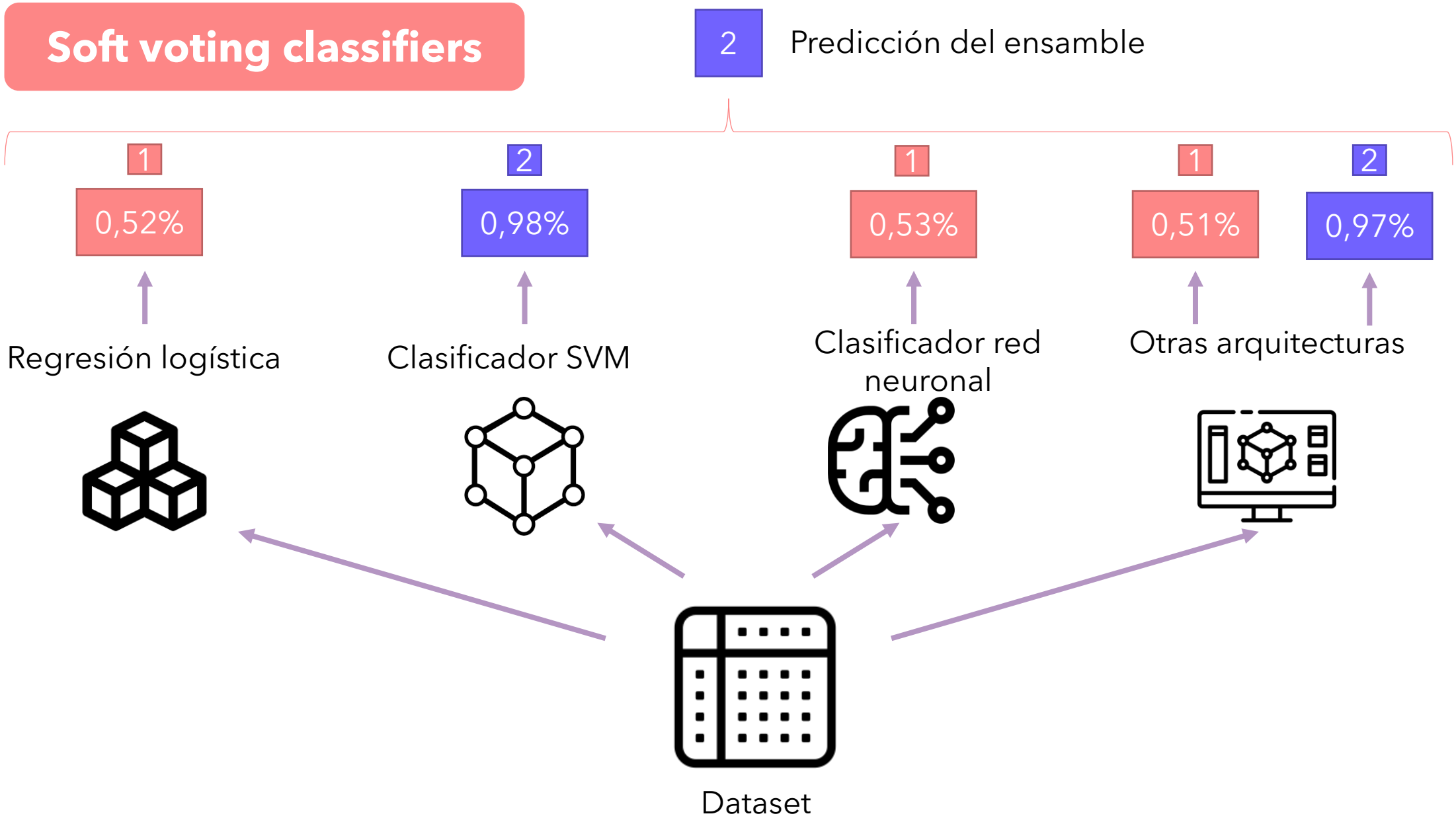
Hard voting classifiers



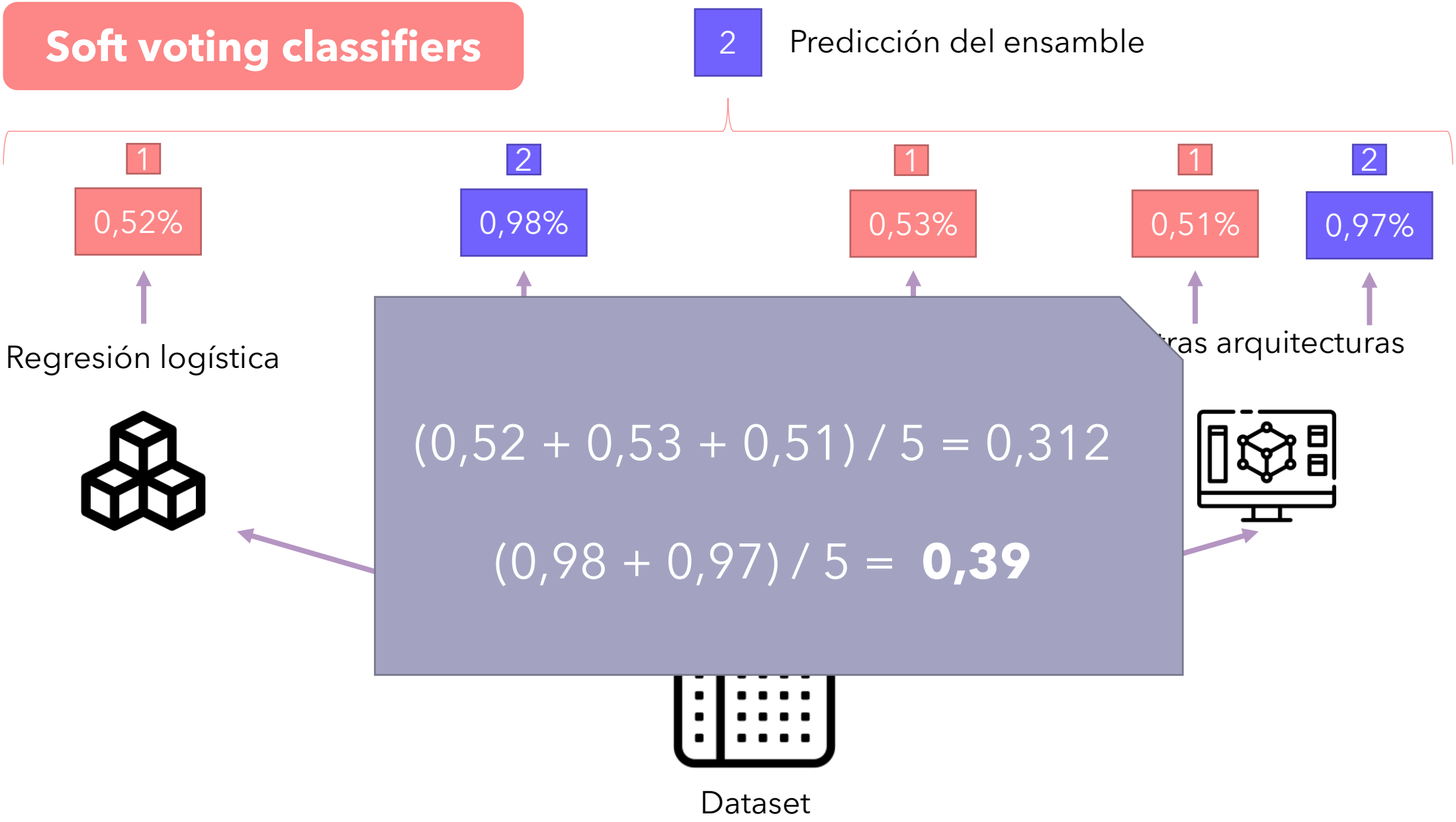
Hard voting classifiers



Soft voting classifiers



Soft voting classifiers



Bagging and Pasting

¿Cómo logramos diversidad en nuestros predictores?

Dijimos que una manera de lograr diversidad en nuestro ensamble era utilizando arquitecturas diferentes para cada predictor. Otra manera de lograr esto es utilizando la misma arquitectura pero entrenándola en distintos subconjuntos de nuestro set de entrenamiento.

→ *Cuando el muestreo es **CON** reposición, el método se llama **Bagging**. Cuando es **SIN** reposición, se llama **Pasting**.*

Cada una de las filas del dataset puede ser muestreada por múltiples predictores. Pero solo Bagging permite que una fila pueda ser muestreada más de una vez por el mismo predictor.

Bagging and Pasting

¿Cómo logramos diversidad en nuestros predictores?

Dijimos que una manera de lograr diversidad en nuestro ensamble era utilizando arquitecturas diferentes para cada predictor. Otra manera de lograr esto es utilizando la misma arquitectura pero entrenándola en distintos subconjuntos de nuestro set de entrenamiento.

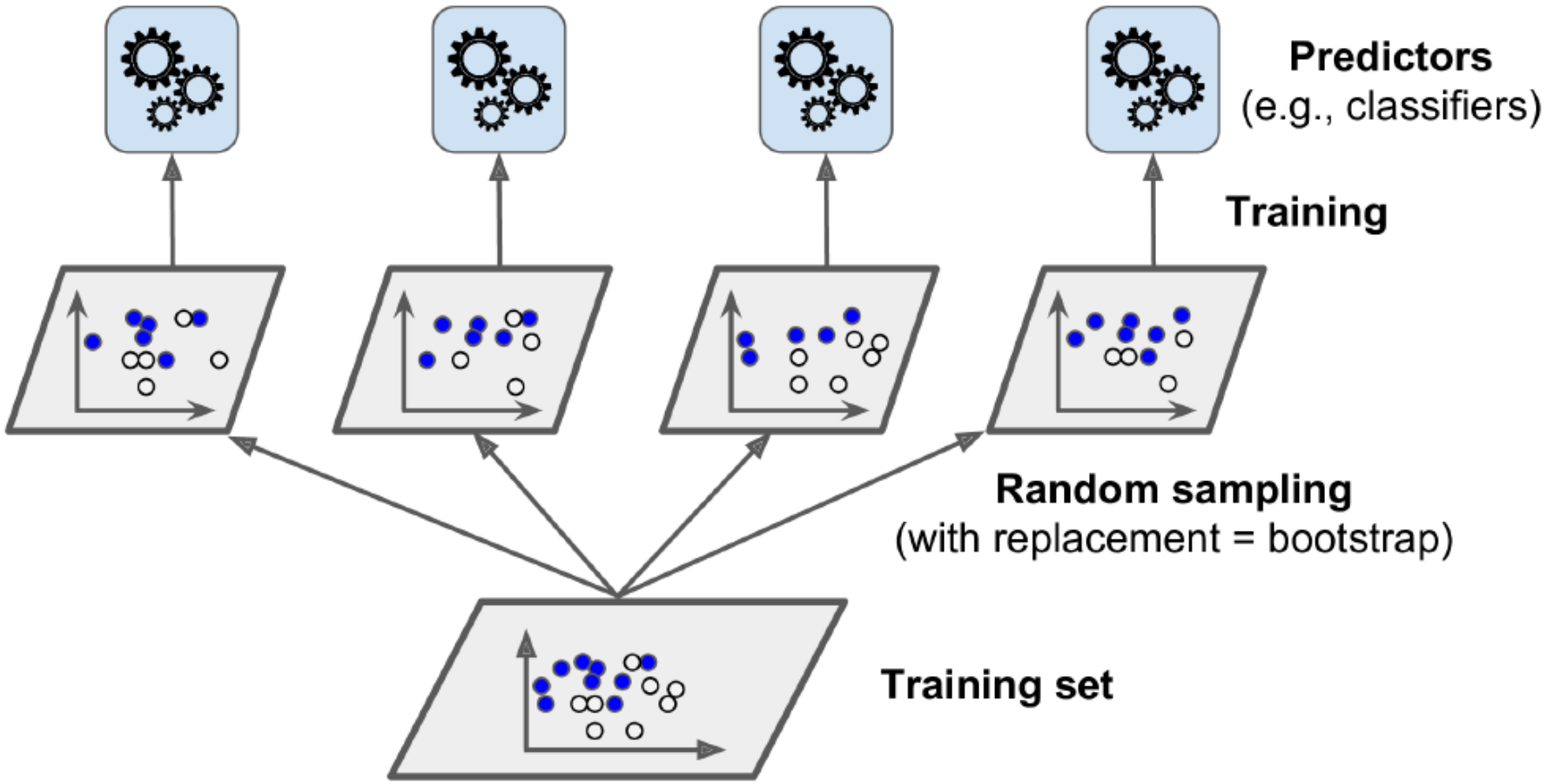
→ *Cuando el muestreo es **CON** reposición, el método se llama **Bagging**. Cuando es **SIN** reposición, se llama **Pasting**.*

Cada una de las filas del data
permite que una fila

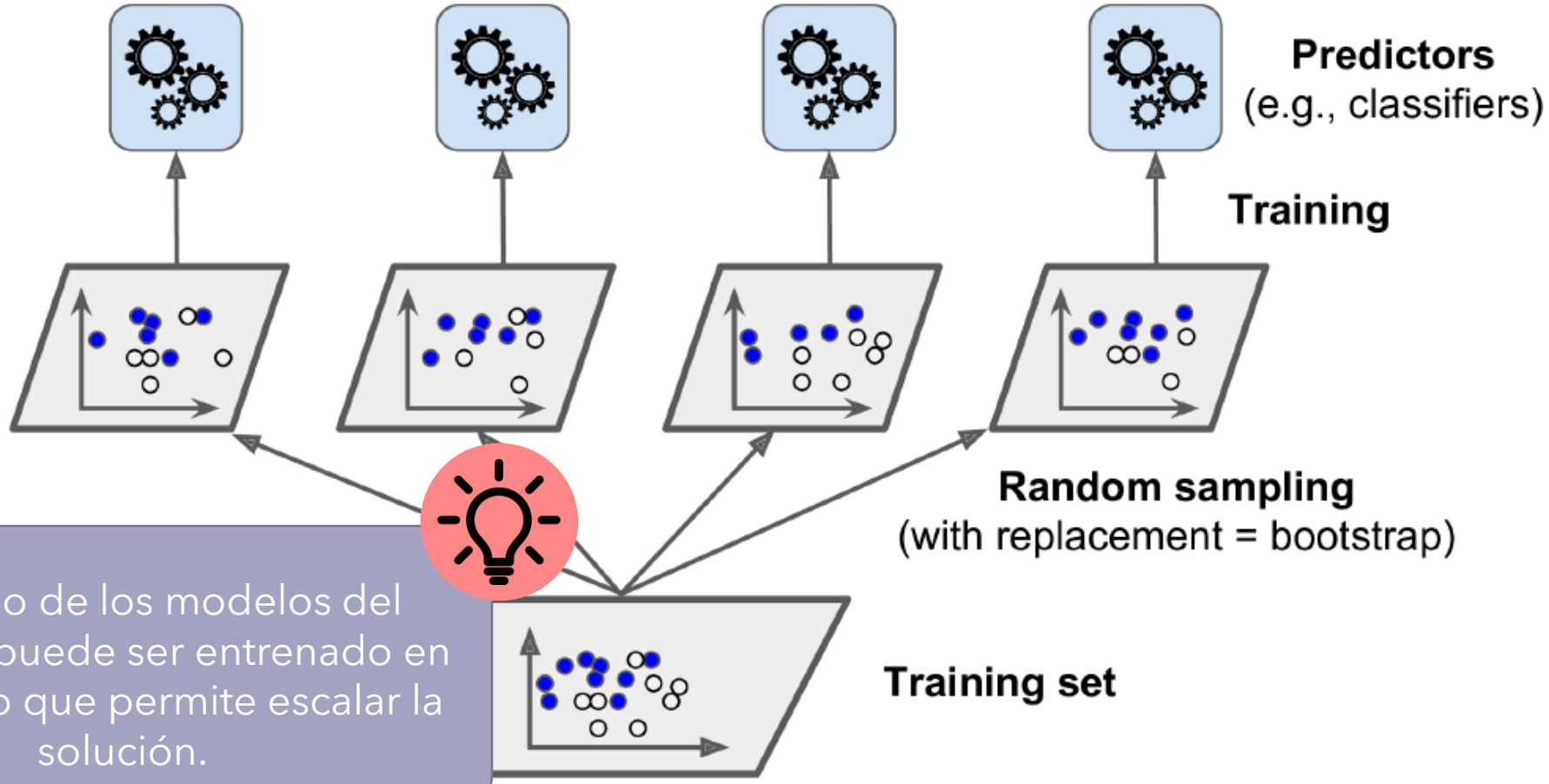
En estadística el muestreo con reemplazo se llama Bootstrapping. Lo que le da el nombre a Bagging (Bootstrap aggregating)

predictores. Pero solo Bagging
por el mismo predictor.

Bagging and Pasting



Bagging and Pasting

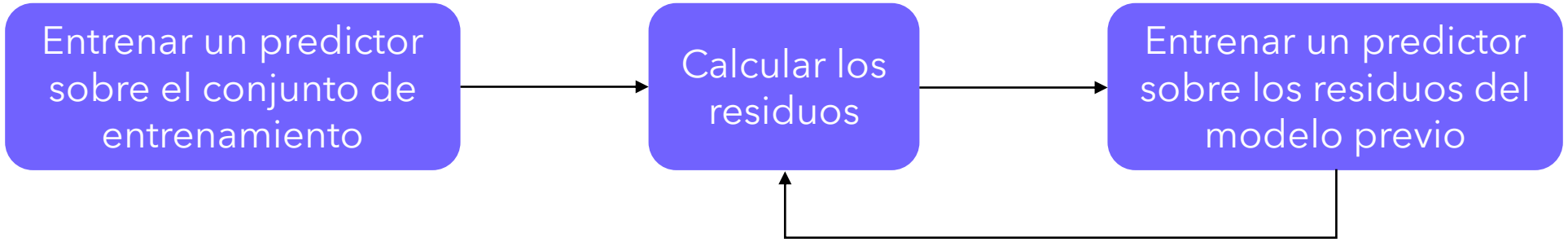


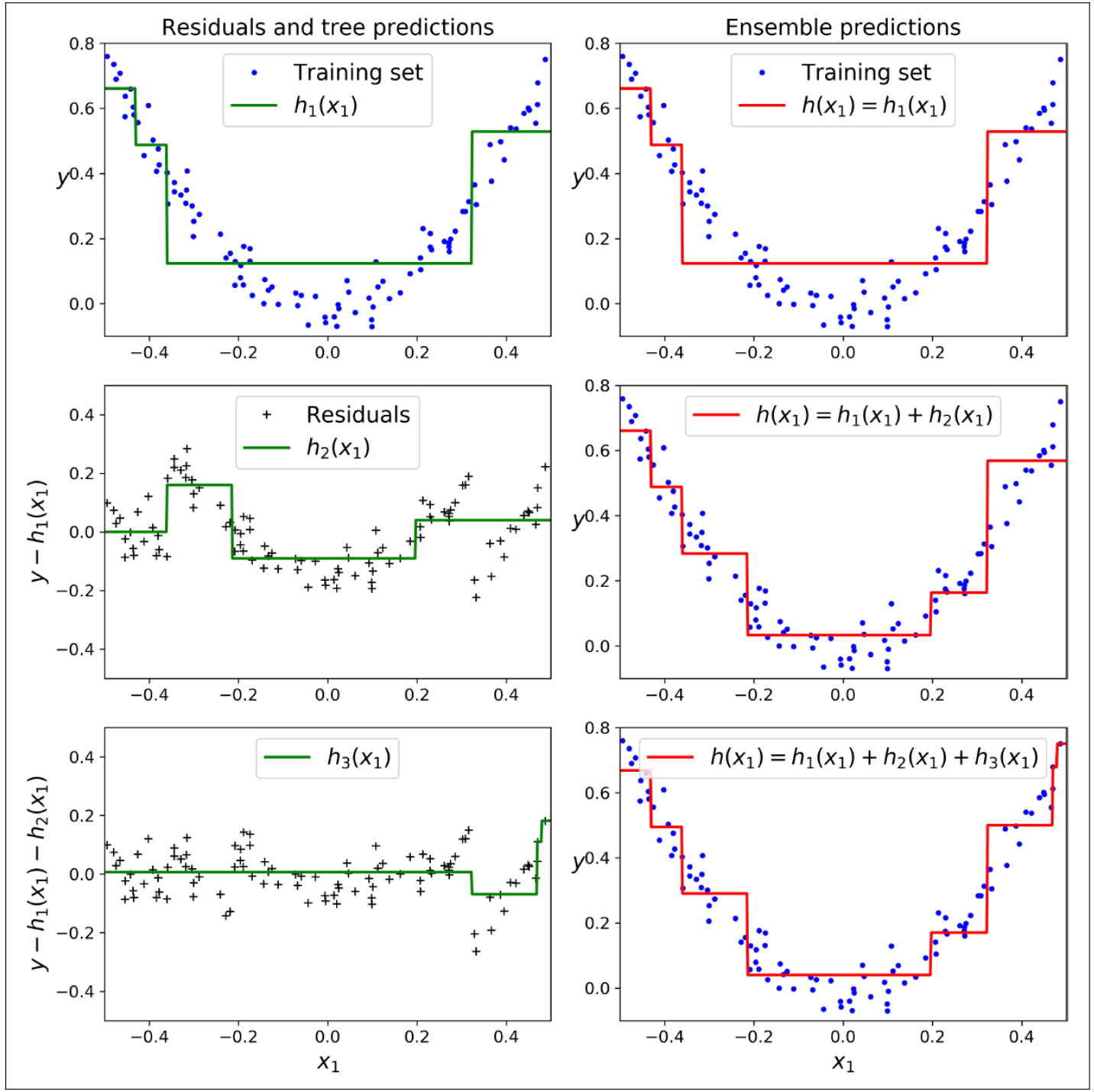
Cada uno de los modelos del ensamble puede ser entrenado en paralelo, lo que permite escalar la solución.

Boosting

Este tipo de arquitecturas son ensambles que al igual que los vistos anteriormente, combinan varios predictores para hacer una mejor predicción. La principal diferencia es que en las arquitecturas que aplican Boosting, el entrenamiento de los predictores se da de forma **secuencial**.

→ Cada nuevo predictor que se agrega al ensamble intenta **corregir a su predecesor**.




Residuals and tree predictions
Ensemble predictions
Training set
$h_1(x_1)$
$h(x_1) = h_1(x_1)$
Residuals
$h_2(x_1)$
$h(x_1) = h_1(x_1) + h_2(x_1)$
$h_3(x_1)$
$h(x_1) = h_1(x_1) + h_2(x_1) + h_3(x_1)$
y
$y - h_1(x_1)$
$y - h_1(x_1) - h_2(x_1)$
x_1

Boosting

Aspecto	Bagging	Boosting
Método de construcción del conjunto de modelos	Múltiples modelos de aprendizaje automático independientes entrenados en subconjuntos aleatorios de datos de entrenamiento	Construye modelos secuencialmente, centrándose en corregir los errores del modelo anterior
Tipo de modelos	Adecuado para cualquier modelo de aprendizaje automático	Se enfoca en modelos débiles y mejora su precisión al combinar modelos débiles
Proceso de predicción	Promedio de las predicciones de todos los modelos del conjunto	Asigna diferentes pesos a cada modelo y los combina para hacer una predicción final
Sensibilidad al ruido	Menos sensible al ruido y a los valores atípicos	Puede ser más sensible al ruido y a los valores atípicos
Tiempo de entrenamiento	Puede ser más rápido ya que los modelos se construyen de forma independiente	Puede requerir más tiempo de entrenamiento ya que cada modelo se entrena secuencialmente
Ventajas	Reduce la varianza, mejora la precisión y puede manejar grandes conjuntos de datos	Mejora la precisión y el rendimiento del modelo mediante la corrección de errores y la creación de modelos más precisos
Desventajas	Puede aumentar el sesgo y no mejorar la precisión si los modelos del conjunto son similares	Puede ser más propenso al sobreajuste y al ruido si se usa un modelo débil o ruidoso

Ajuste de hiper-parámetros

Hiperparámetros

¿Qué son?

Los hiperparámetros (HPs) son aquellos parámetros que caracterizan un modelo de aprendizaje automático / aprendizaje profundo, cuyos valores no se ajustan de forma automática durante el entrenamiento.

Árbol de decisión	Random forest	Máquina de soporte vectorial	Red neuronal (MLP)
• Métrica por la cual se van a realizar las divisiones en los nodos. • Profundidad del árbol. • Cantidad de características a evaluar en cada nodo.	• Cantidad de estimadores. • Taza de aprendizaje para GBM.	• Kernel. • Grado del kernel para polinómicos. • Parámetro de regularización C.	• Cantidad de capas. • Cantidad de neuronas por capa. • Taza de aprendizaje.

Hiperparámetros

Características

- El ajuste de los HPs tiene gran **impacto en el desempeño** del modelo.

- Los HPs óptimos **difieren para distintos conjuntos de datos**

  → tienen que ser optimizados para cada conjunto de datos.

Hiperparámetros

¿Cómo seleccionamos los hiperparámetros óptimos?

El proceso de selección de hiperparámetros se conoce como ajuste de hiperparámetros u optimización de hiperparámetros → método para seleccionar los HPs que minimizan el error de generalización.

Hiperparámetros → min(métricas de performance del modelo)

Los HPs son un conjunto de valores que minimizan las métricas de performance del modelo.

Si estamos intentando minimizar/maximizar una función…

¿Por qué no utilizamos algún método conocido para encontrar el mínimo/máximo?

Hiperparámetros → min(métricas de performance del modelo)

La métrica que se intenta minimizar depende de:

- La arquitectura del modelo
- Los datos
- Los propios HPs
- Métrica seleccionada

Superficie de respuesta de los HPs

¿Por qué no utilizamos algún método conocido para encontrar el mínimo/máximo?

Hiperparámetros → min(métricas de performance del modelo)

La métr se intenta minimizar depende de:

- La arquite
- Los datos
- Los propio
- Métrica seleccionada

NO ES POSIBLE ASOCIAR UNA FÓRMULA MATEMÁTICA QUE REPRESENTE LA SUPERFICIE DE RESPUESTA DE LOS HPS → **NO ES DIFERENCIABLE**

e de
de los

¿Qué se necesita para poder optimizar los HPs?

- Un espacio de hiperparámetros.
- Un método para seleccionar los candidatos dentro del grupo de HPs a evaluar.
- Un esquema de validación cruzada.
- Una métrica para evaluar la performance.

Métodos para seleccionar HPs óptimos

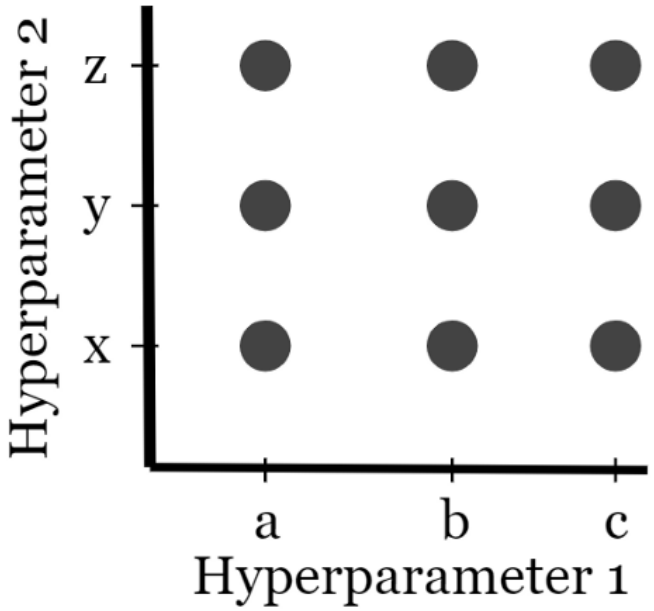
- Manual Search
- Grid Search
- Random Search
- Sequential Search
- Otros

Métodos clásicos

Grid Search

Pseudocode

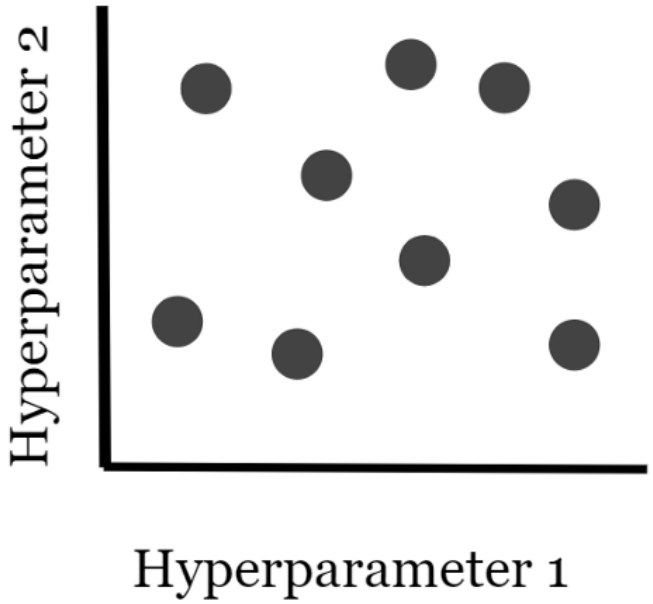
```
Hyperparameter_One = [a, b, c]
Hyperparameter_Two = [x, y, z]
```



Random Search

Pseudocode

```
Hyperparameter_One = random.num(range)
Hyperparameter_Two = random.num(range)
```



Esquema de búsqueda de hiperparámetros por los métodos clásicos Grid Search y Random Search

Búsqueda secuencial

Son técnicas que seleccionan un conjunto de HPs, evalúan la calidad/performance y en base a ello deciden hacia donde orientar el siguiente muestreo de HPs.

- Es un proceso **iterativo** y secuencial.
- **No es paralelizable** en su totalidad.
- Tiene como objetivo **probar una menor cantidad de HPs** pero hacerlo de manera inteligente sobre los más prometedores.

¿Por qué y cuándo utilizar búsqueda secuencial para ajustar los HPs?

Cuando la complejidad del modelo que se está entrenando es muy elevada, los métodos clásicos se vuelven muy costosos.

Existe una relación de compromiso entre:

- **Tiempo de entrenamiento**
- **Tiempo de estimación del próximo conjunto de HPs**

Tiempo de entrenamiento > Tiempo de estimación del próximo conjunto de HPs

¿Cómo podemos implementar una búsqueda secuencial de HPs?

Optimización bayesiana

Es una estrategia secuencial para **optimización** de funciones de **caja negra** las cuales no asumen ninguna fórmula/función.

Es utilizada comúnmente para optimizar funciones que son **complejas/costosas de evaluar**.

La función objetivo debe poder ser evaluada en puntos arbitrarios.

$$x^* = arg\,max\, f(x) \quad ; \quad donde\ f = superficie\ de\ respuesta\ de\ los\ HPs$$
$$x = HPs$$

Optimización bayesiana

Se trata a *f* con una función aleatoria y se realiza una estimación a **priori** de la distribución sobre ella.

Se evalúan algunos puntos de *f*.

Utilizando nuevos datos, la distribución a priori se actualiza a una distribución a **posteriori**.

La distribución a posteriori es utilizada para construir una función que determina donde continuar evaluando.

Optimización bayesiana

Se trata a *f* con una función aleatoria y se realiza una estimación a **priori** de la distribución sobre ella.

Gaussian processes
Tree-Parzen estimator
Random Forests

Se evalúan algunos puntos de *f*.

Utilizando nuevos datos, la distribución a priori se actualiza a una distribución a **posteriori**.

La distribución a posteriori es utilizada para construir una función que determina donde continuar evaluando.

Optimización bayesiana

Se trata a *f* con una función aleatoria y se realiza una estimación a **priori** de la distribución sobre ella.

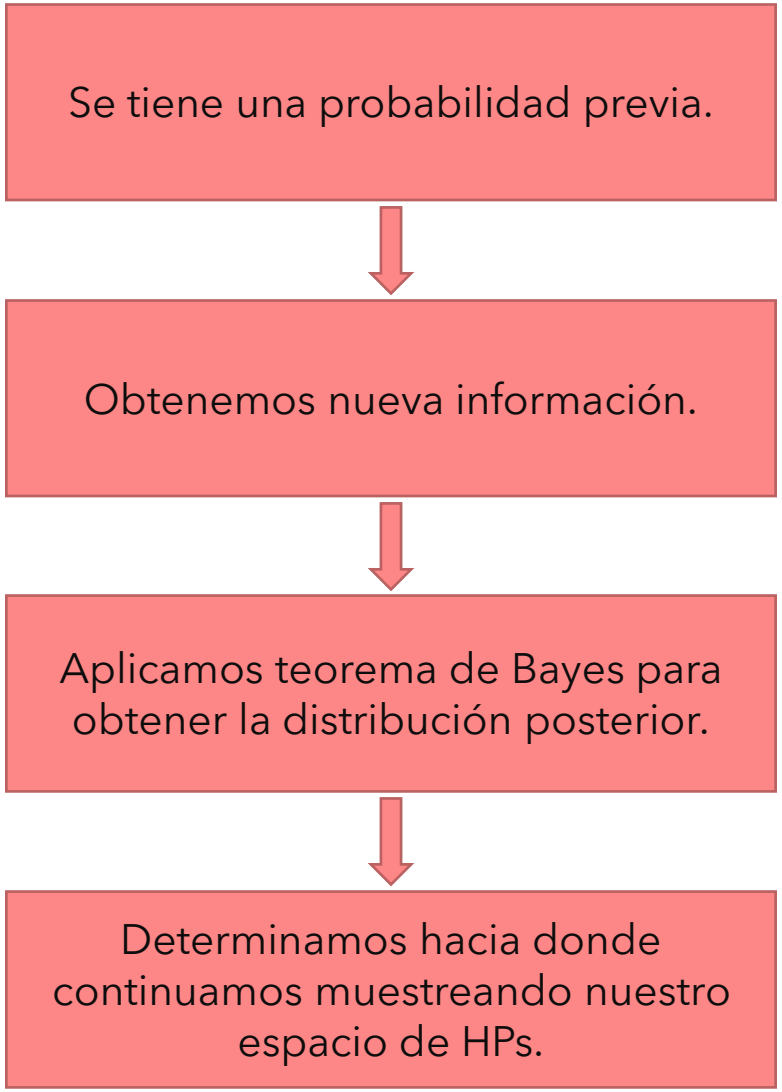
Se evalúan algunos puntos de *f*.

Utilizando nuevos datos, la distribución a priori se actualiza a una distribución a **posteriori**.

La distribución a posteriori es utilizada para construir una función que determina donde continuar evaluando.

Expected Improvement
Probability of Improvemenet
Gaussian process upper confidence bound

Optimización bayesiana



Inferencia Bayesiana

Tiene dos ideas principales:

- La inferencia Bayesiana nos permite reubicar las probabilidades de un evento a partir de la ocurrencia de otros sucesos posibles.
- Estos sucesos posibles que participan en la reubicación de la probabilidad de un evento, son los parámetros de un modelo matemático.

Teorema de Bayes

$$P(A|B) = \frac{P(B|A) \times P(A)}{P(B)}$$

- A y B son eventos.
- P(A|B) es la probabilidad a posterior, o condicional, de A dada la evidencia B.
- P(A) y P(B) son las probabilidades marginales de los eventos dados en forma independiente.

Teorema de Bayes

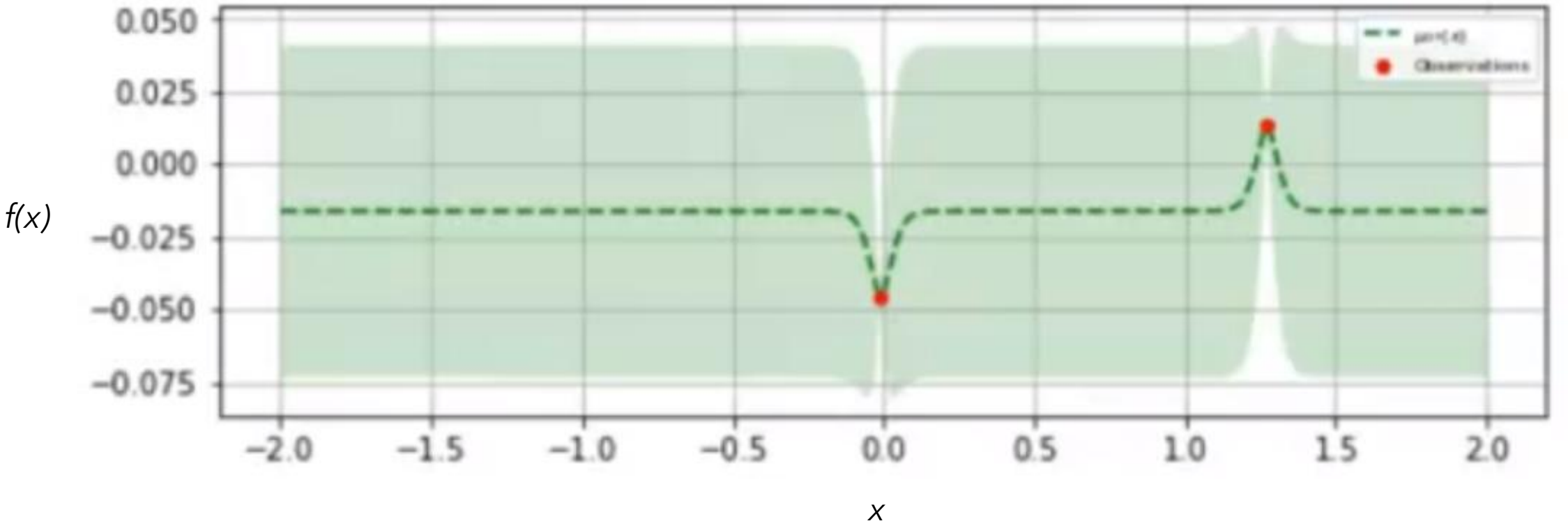
$$P(A|B) = \frac{P(B|A) \times P(A)}{P(B)}$$

$$P(w|D) = \frac{P(D|w) \times P(w)}{P(D)}$$

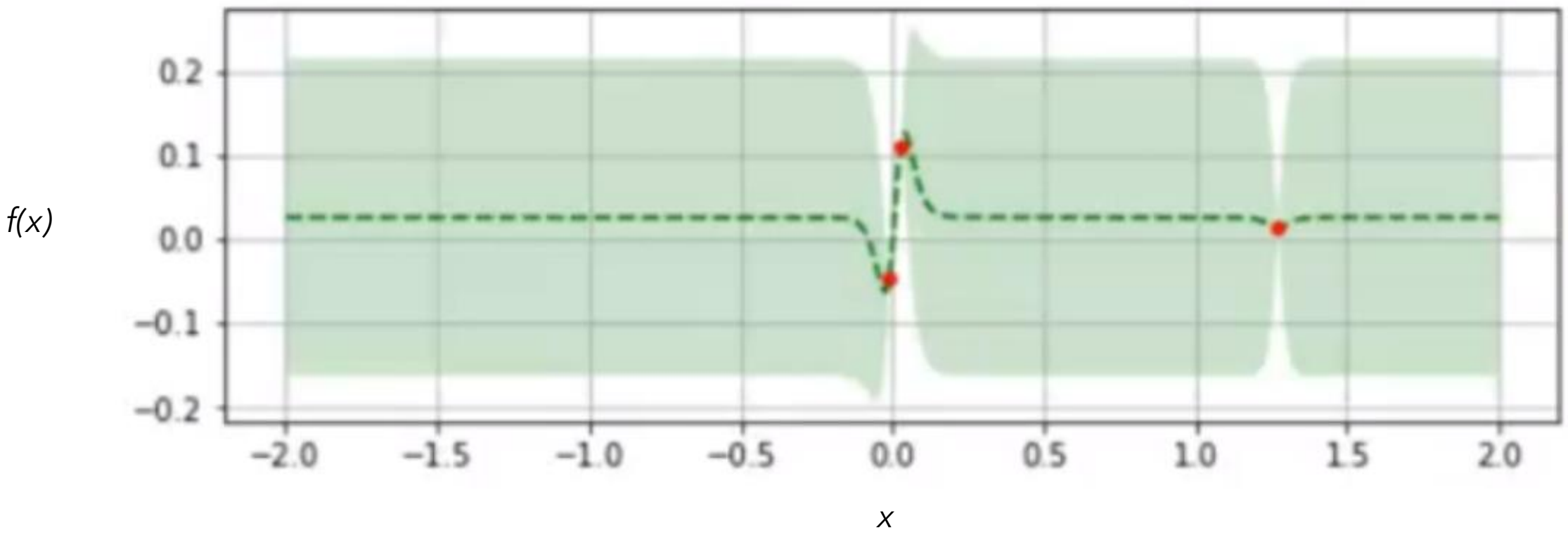
$w: f(x)$

$D: datos$

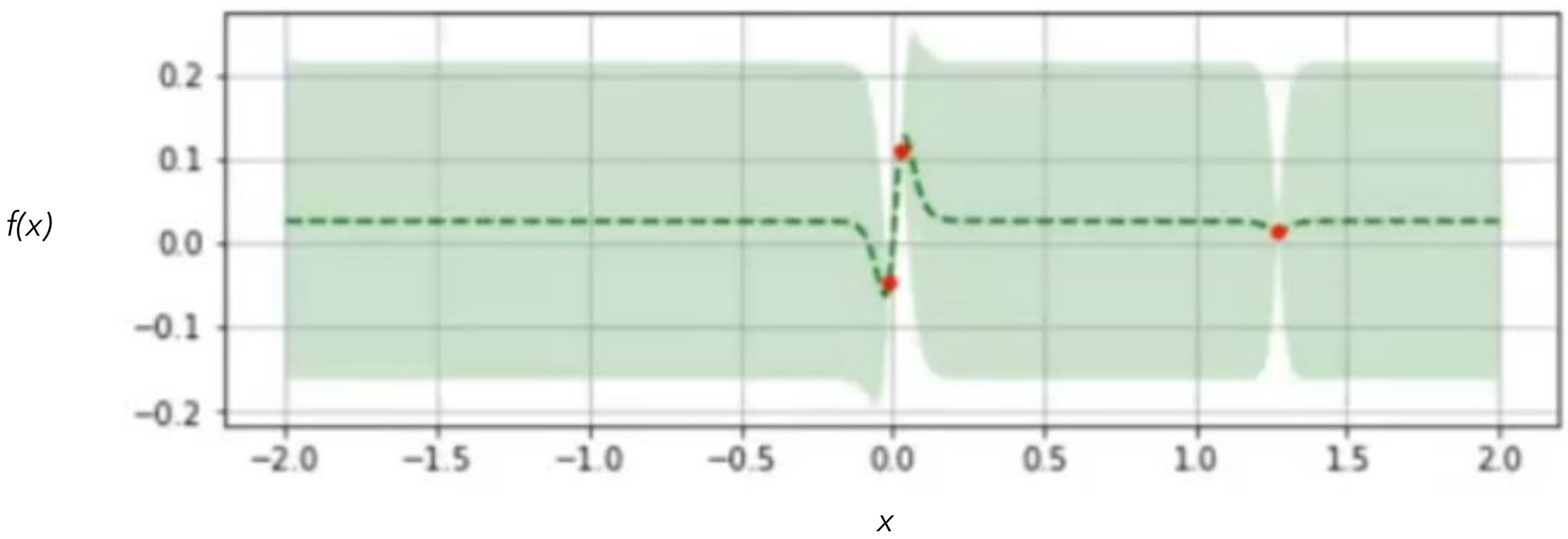
Búsqueda secuencial



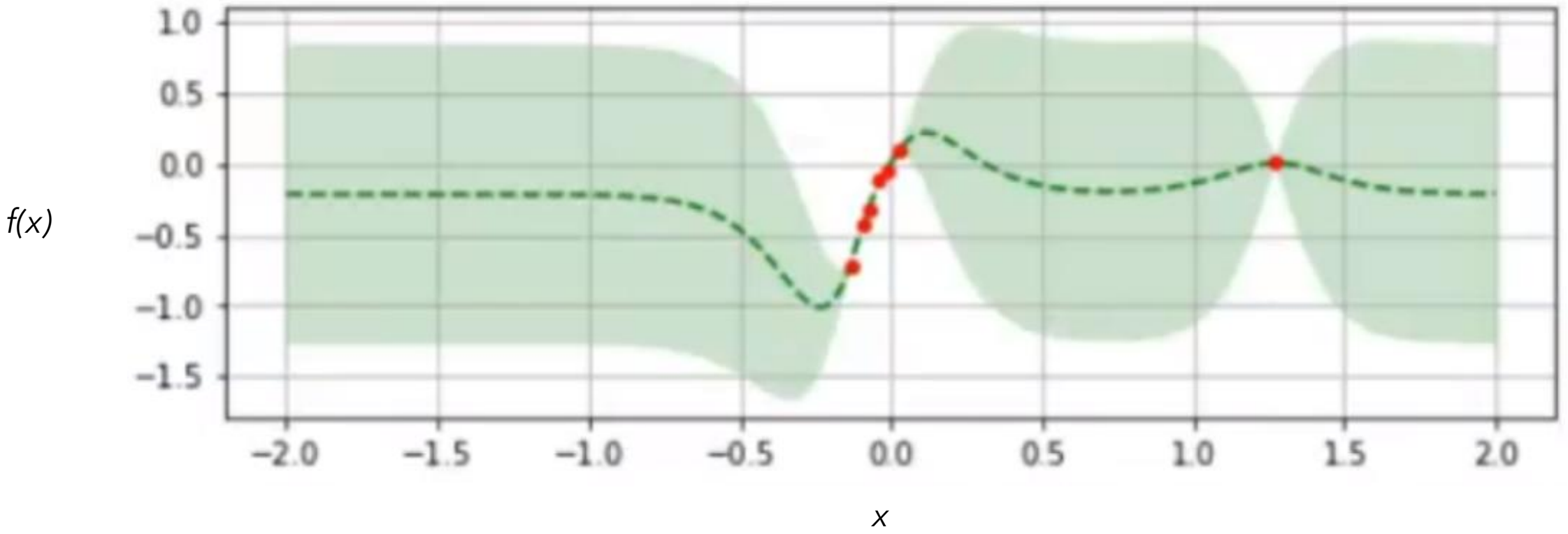
Búsqueda secuencial



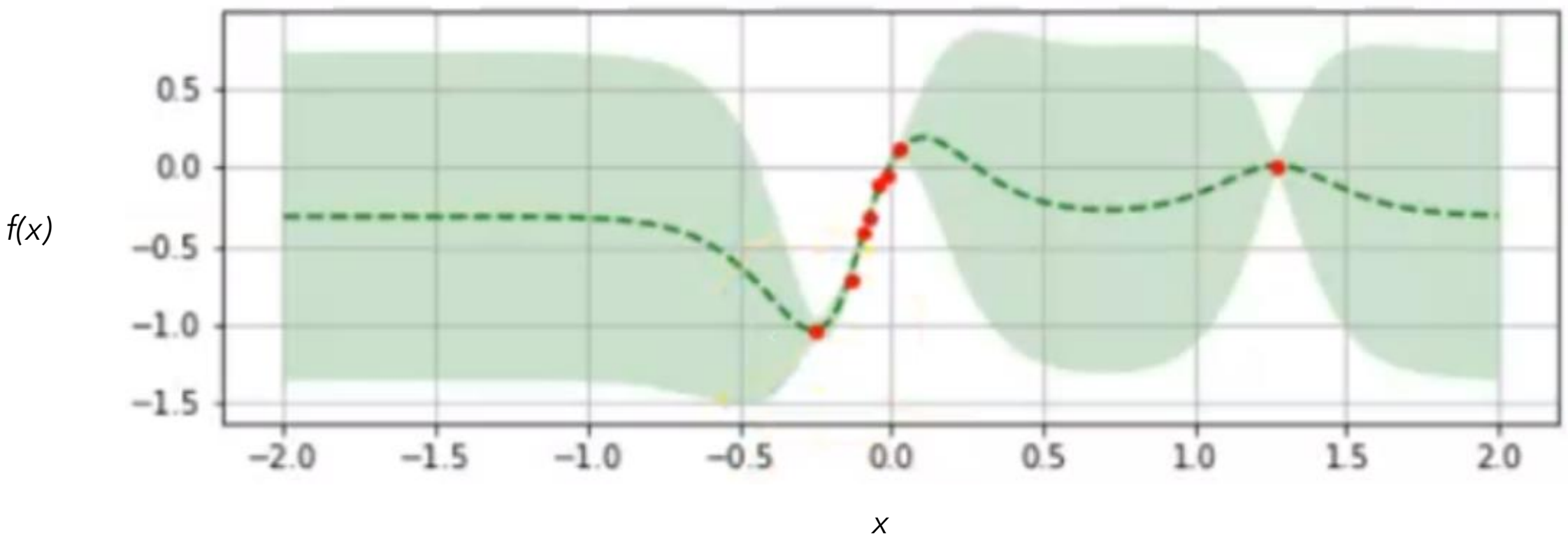
Búsqueda secuencial



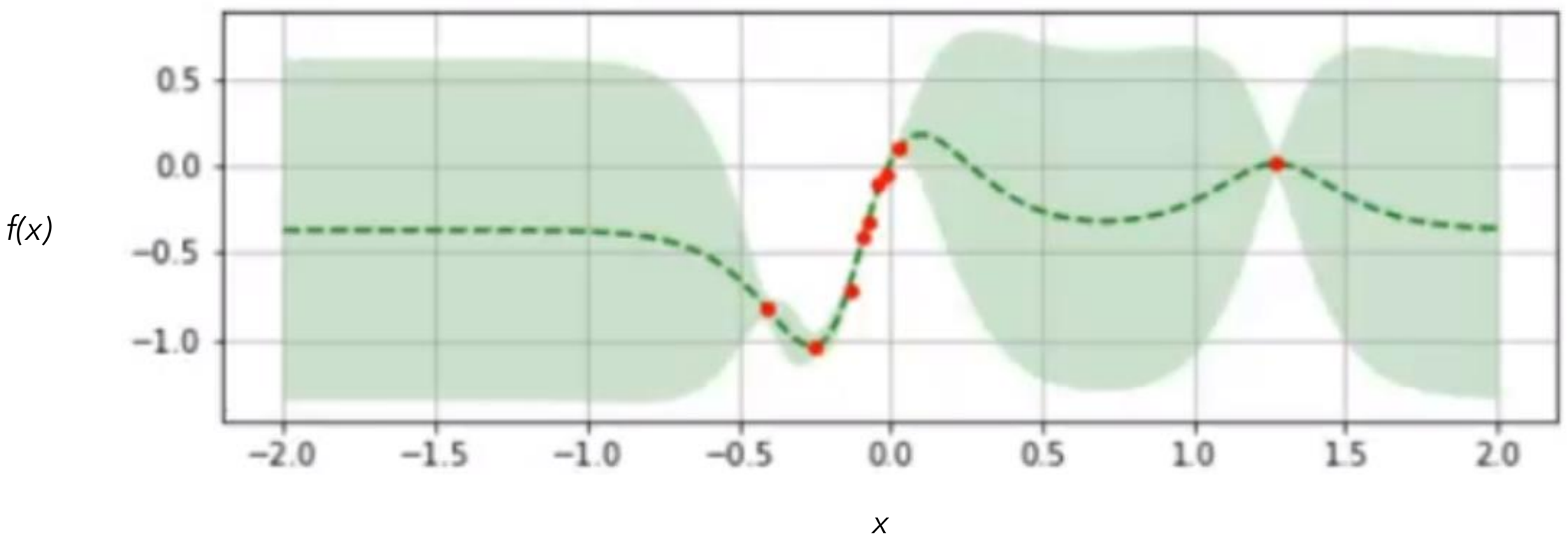
Búsqueda secuencial



Búsqueda secuencial



Búsqueda secuencial



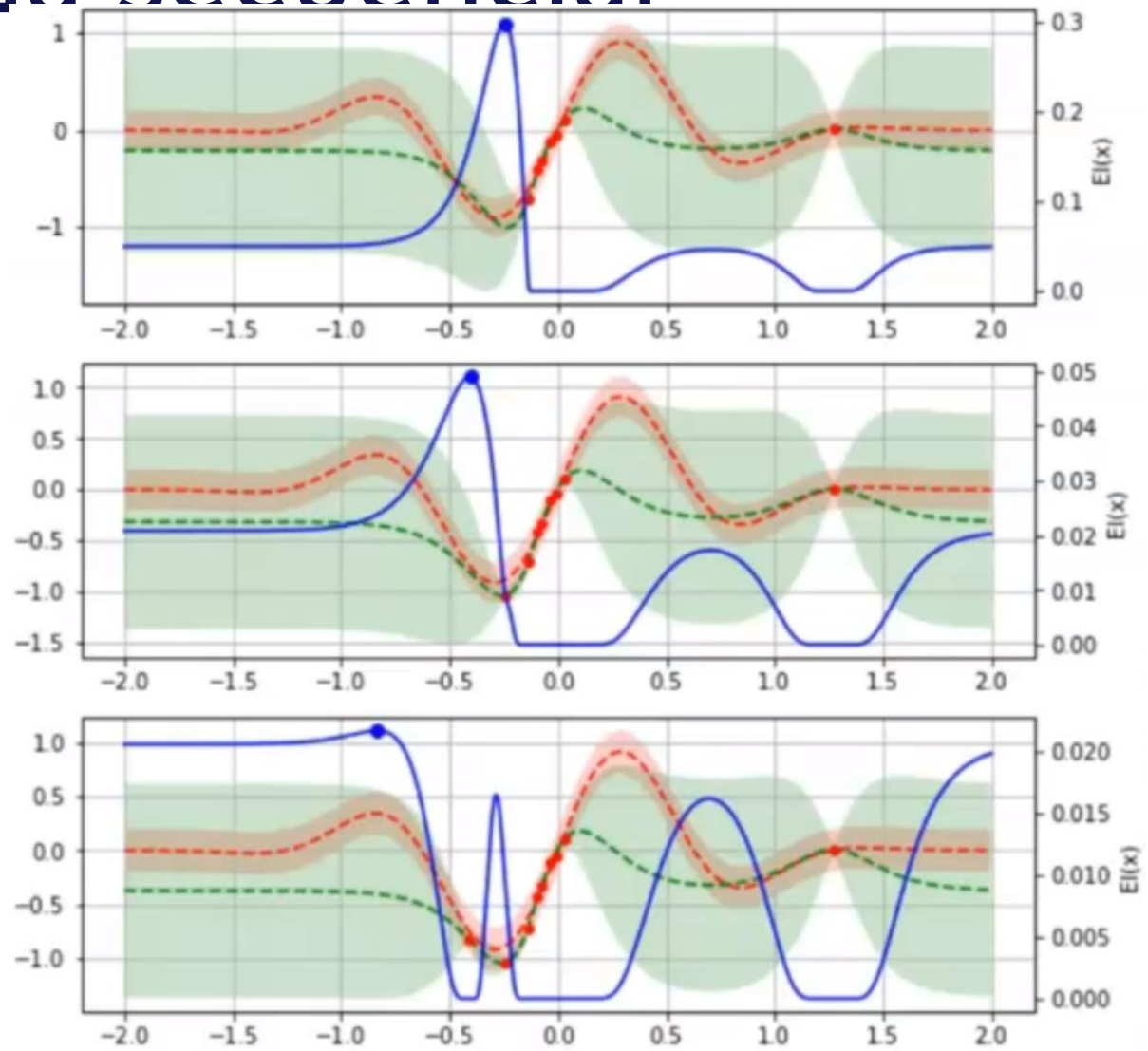
Búsqueda secuencial

- **Modelo sustituto (surrogate model)**

  Necesitamos inferir f(x) de alguna manera y estimar donde se pueden encontrar sus posibles valores

- **Función de adquisición**

  Necesitamos una manera de estimar donde continuar muestreando nuestro espacio de HPs



Búsqueda secuencial

La clave para este método de optimización es que tanto la función de adquisición como el modelo sustituto, sean mas sencillos de evaluar que la propia $f(x)$.

Tree Parzen Estimators

Muestrear f(x) en varios valores de x.

↓

Ordenar los valores muestreados en dos grupos, valores pequeños y valores grandes de f(x).

↓

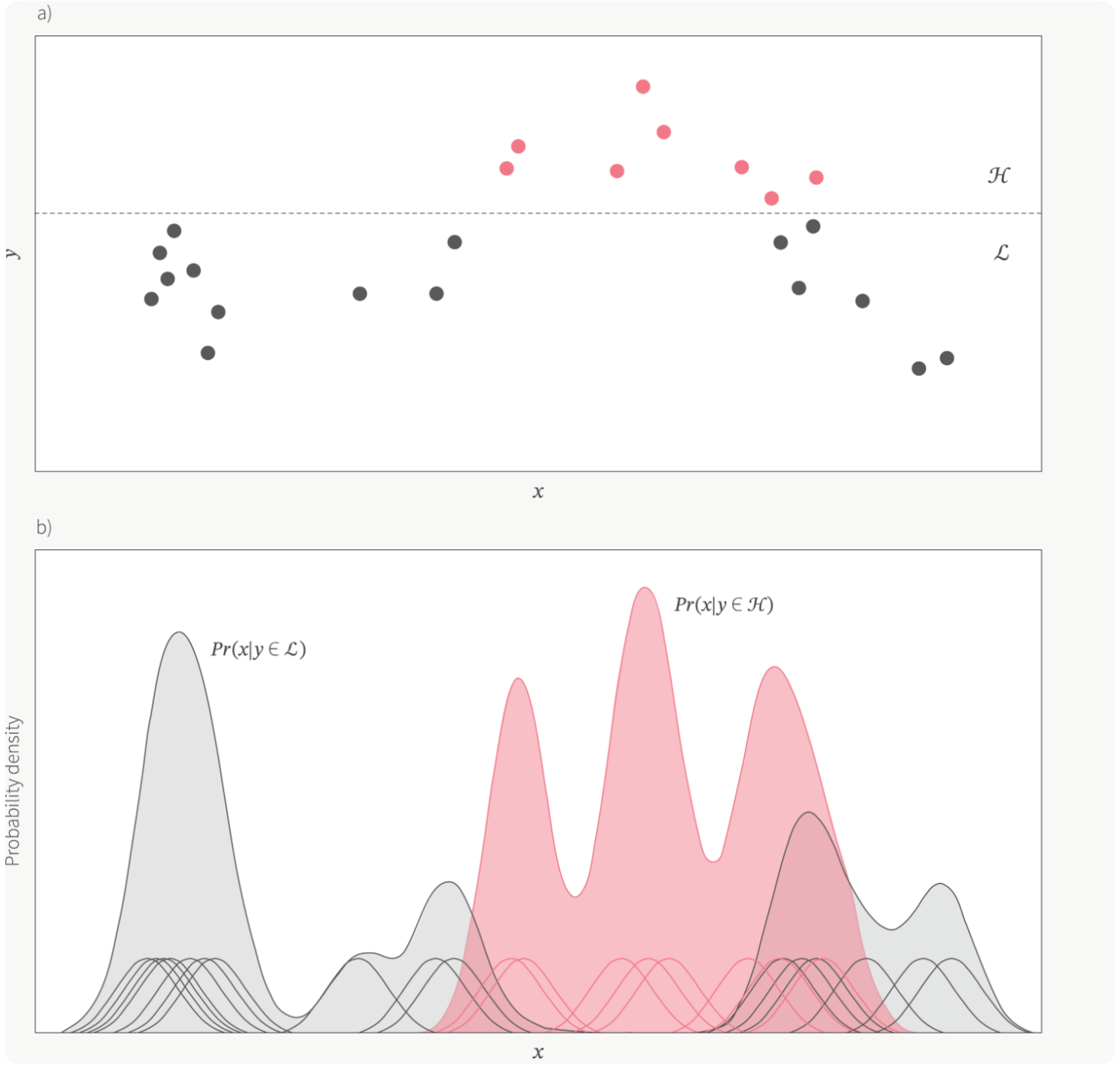
Estimar la distribución de los HPs dada por f(x). Utilizar Parzen Windows para realizar una estimación de densidad de la distribución.

↓

Seleccionar algunas muestras de L(x).

↓

(…)

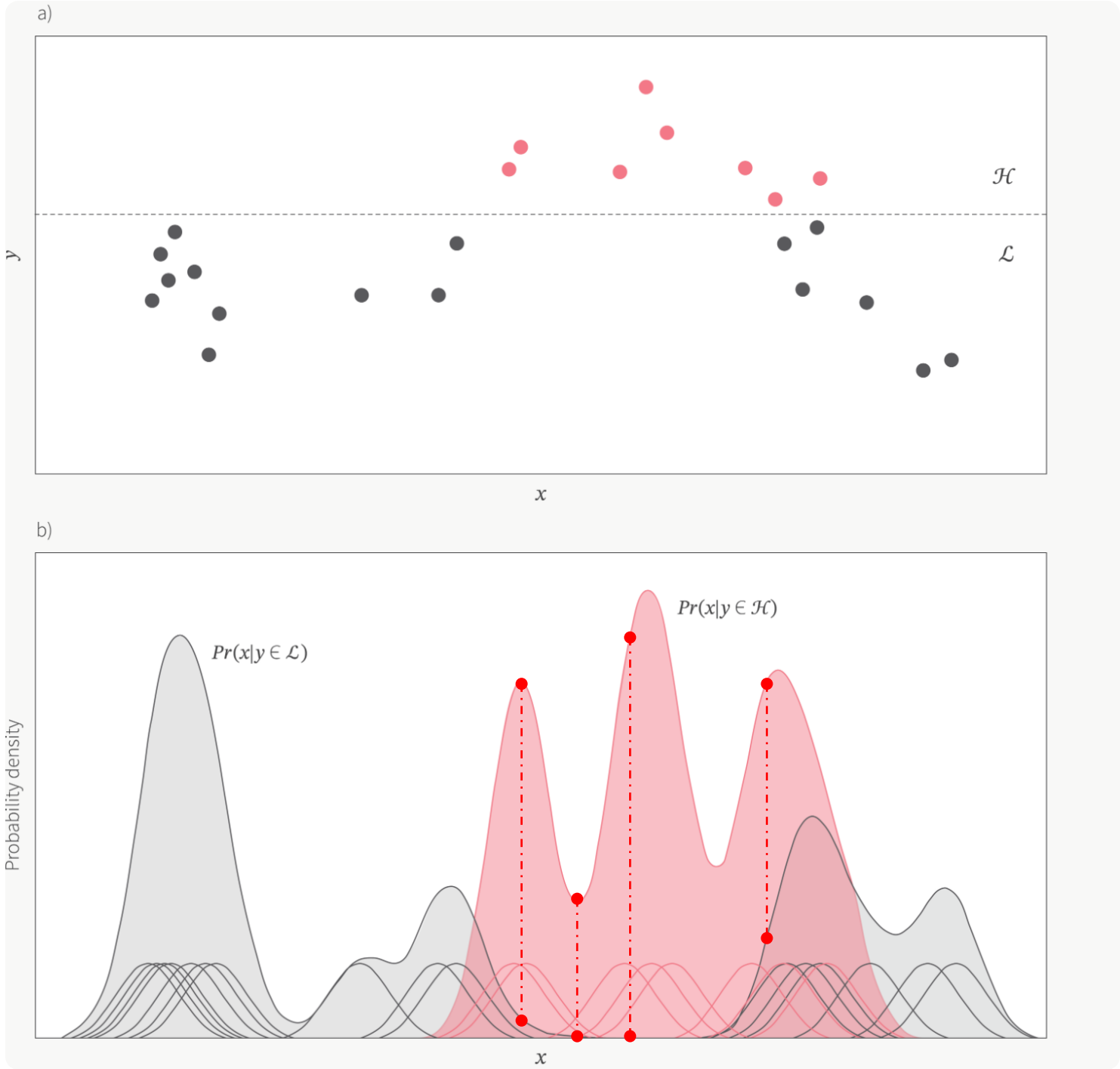


(…)

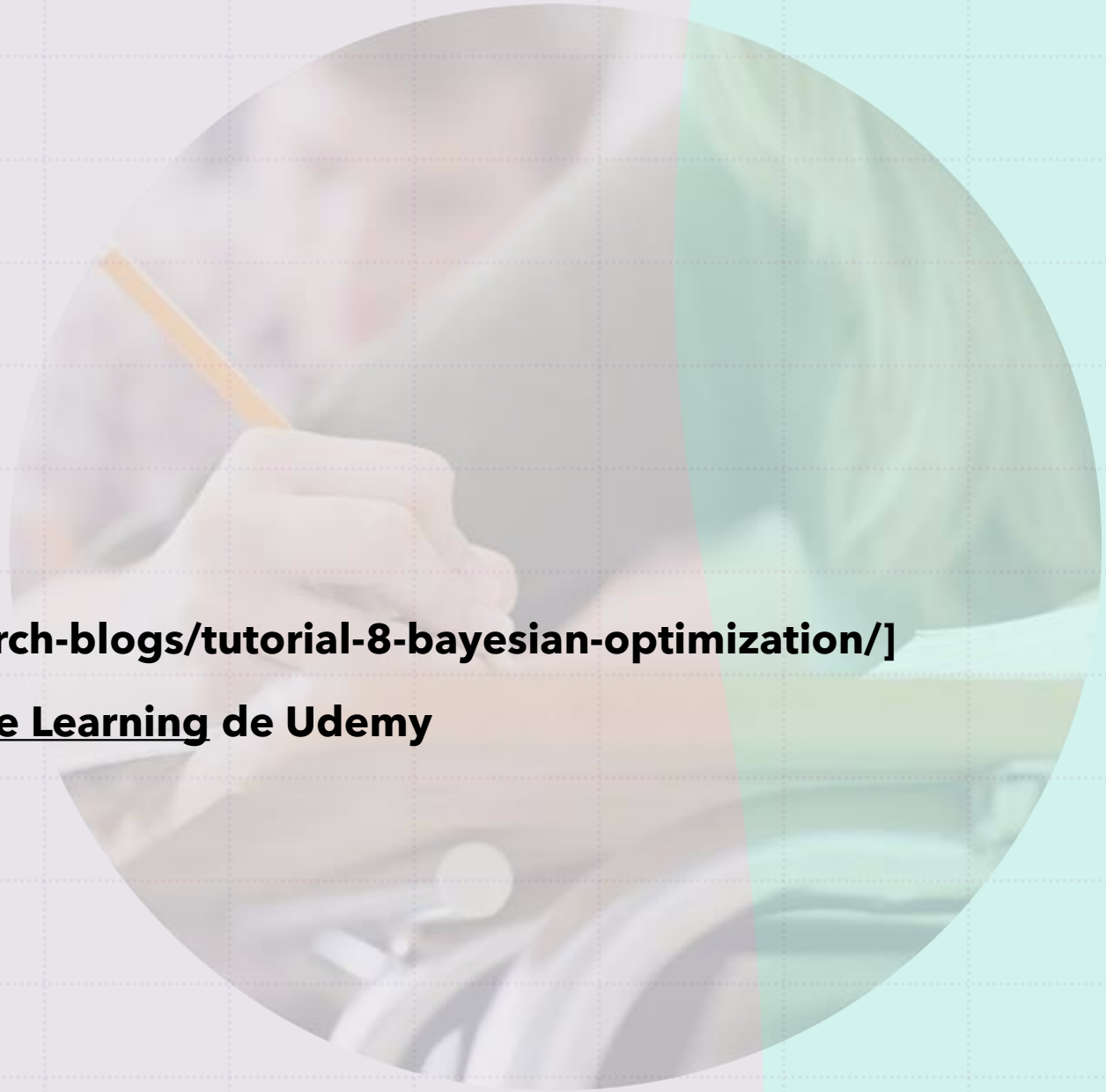
Calcular la mejora esperada (EI).

Encontrar la máxima EI.

Evaluar f(x) con nuevos hiperparámetros.



Bibliografía de referencia.



BOREALIS AI [https://www.borealisai.com/research-blogs/tutorial-8-bayesian-optimization/]

Curso: Hyperparameter Optimization for Machine Learning de Udemy