# Métodos de Ensamble

# Aprendizaje de Máquina Aplicado

Juan David Martínez Vargas, Ph.D.
jdmartinev@eafit.edu.co

Leandro Higuita, MSc.
clhiguitap@eafit.edu.co

2022



UNIVERSIDAD EAFIT®

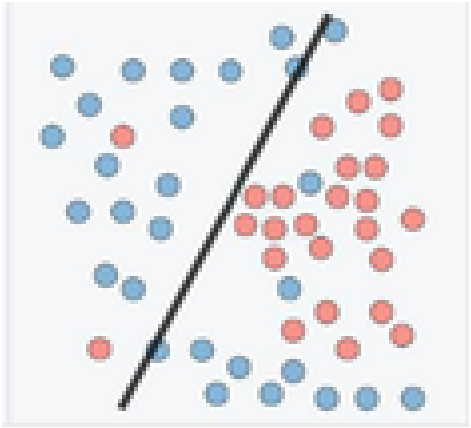
# Agenda

- Introducción
- Árboles de decisión
- Bootstrap aggregating
- Boosting
- Ejemplo práctico

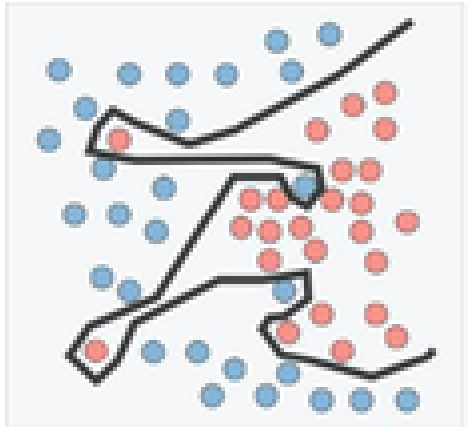
# Introducción

# Motivación

Los modelos simples son eficientes y de rápido entrenamiento, pero tienen poco poder explicativo sobre datos observados (subajuste).

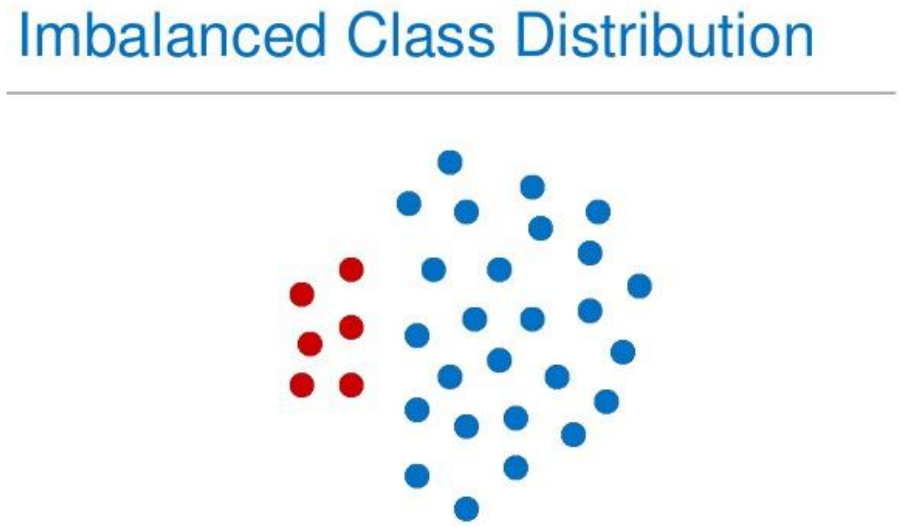


Por otro lado, los modelos muy complejos tienen mucho poder explicativo sobre un conjunto de entrenamiento pero poca habilidad predictiva para datos no observados (sobreajuste).



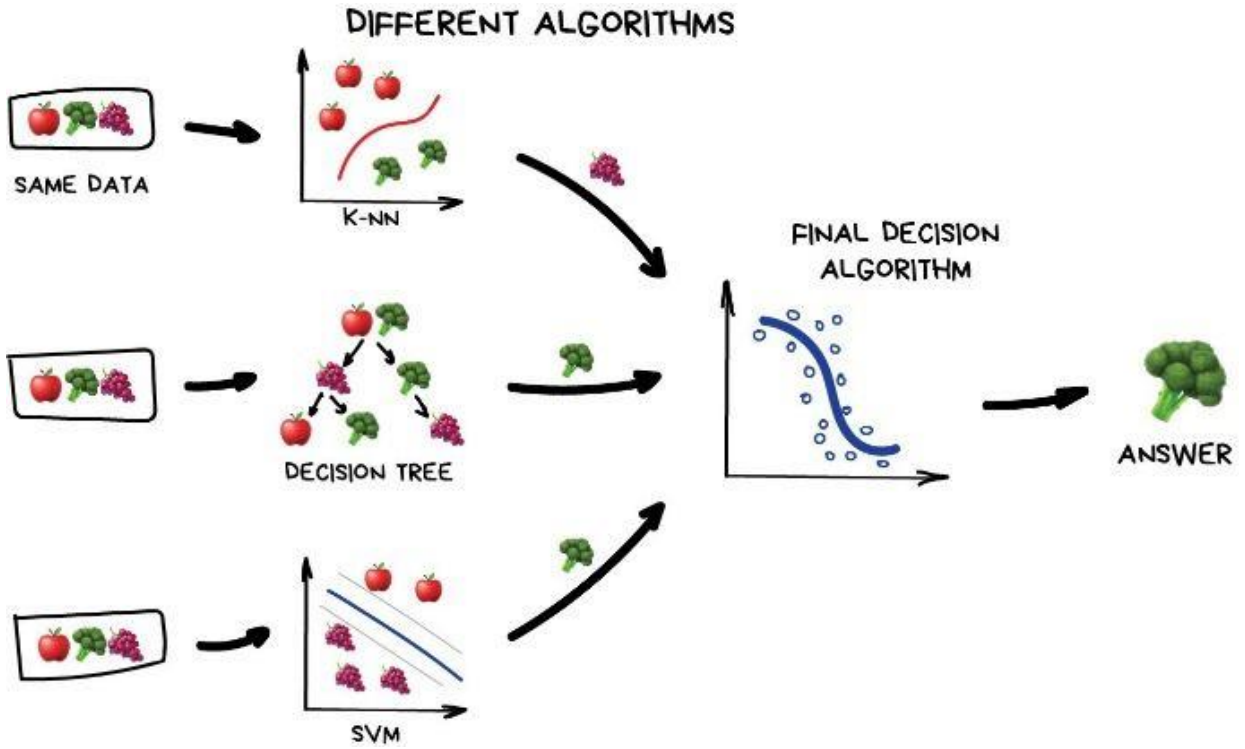
# Motivación

Además, los datos en ocasiones tienen una estructura compleja y, específicamente en problemas de clasificación, pueden presentarse clases no balanceadas.



# Ensambles de Modelos de ML

Los ensambles combinan varios modelos de Machine Learning para producir una única predicción y pueden presentar mejor desempeño que los modelos individuales.



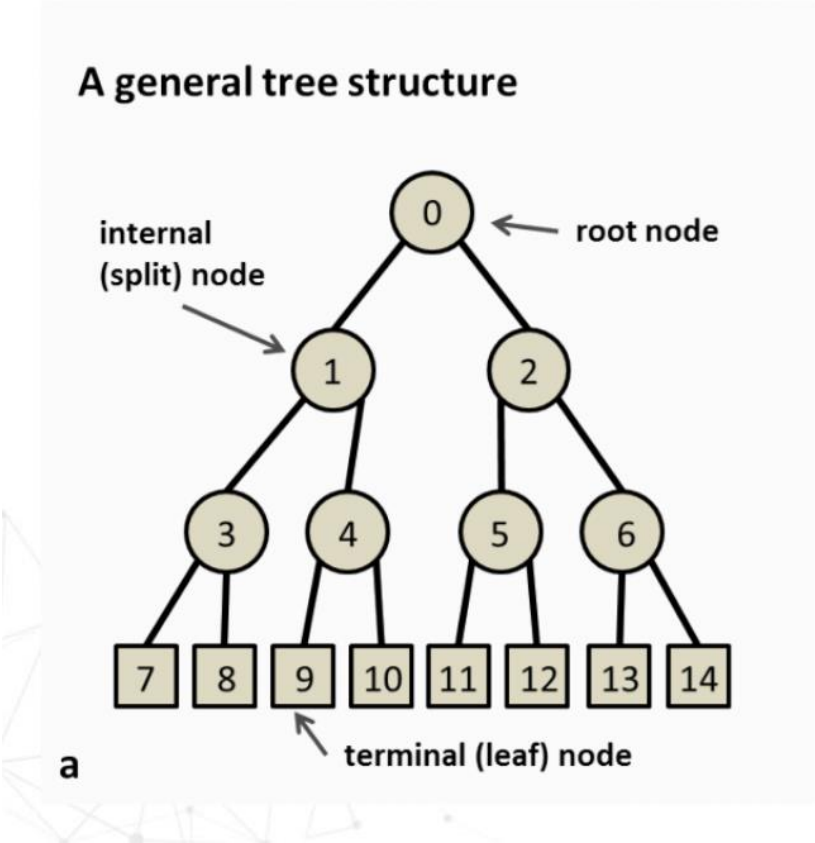
# Ensambles: Clasificación

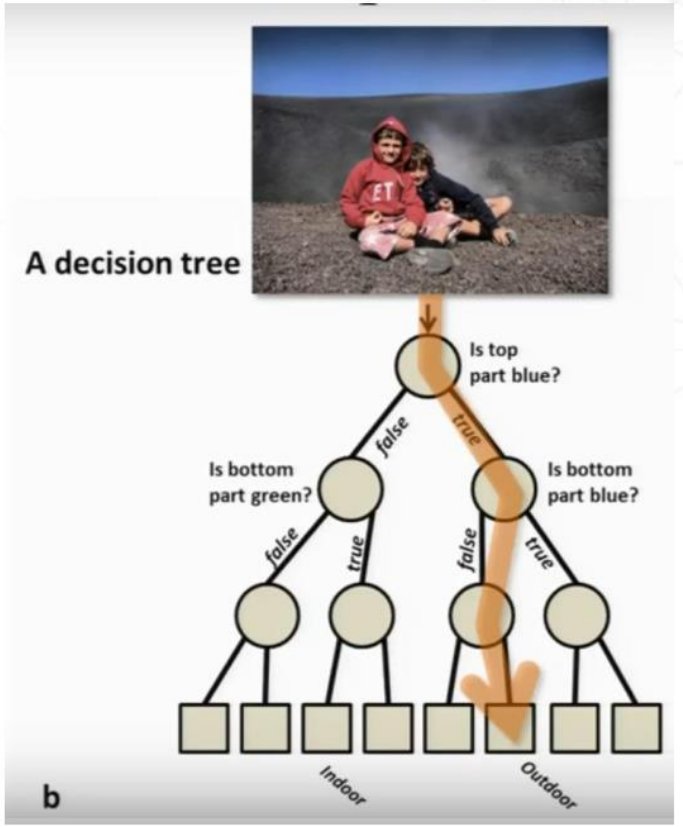
La clasificación de los modelos de ensamble se puede abordar desde tres perspectivas:

- Por el tipo de problema que procuran resolver: subajuste o sobreajuste.
- Por la manera en la que se entrena el ensamble: paralelos o secuenciales.
- Por la variedad de modelos en el ensamble: homogéneos o heterogéneos.

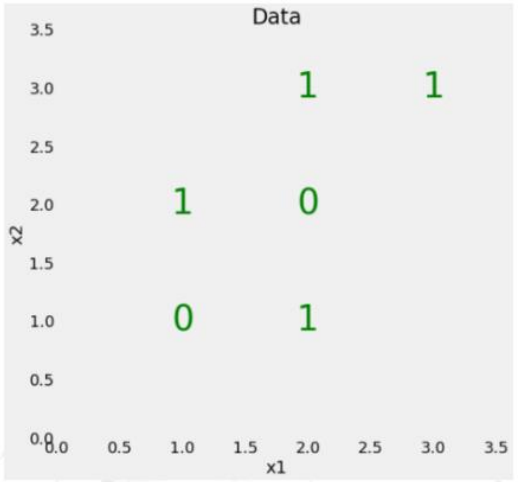
# Árboles de decisión

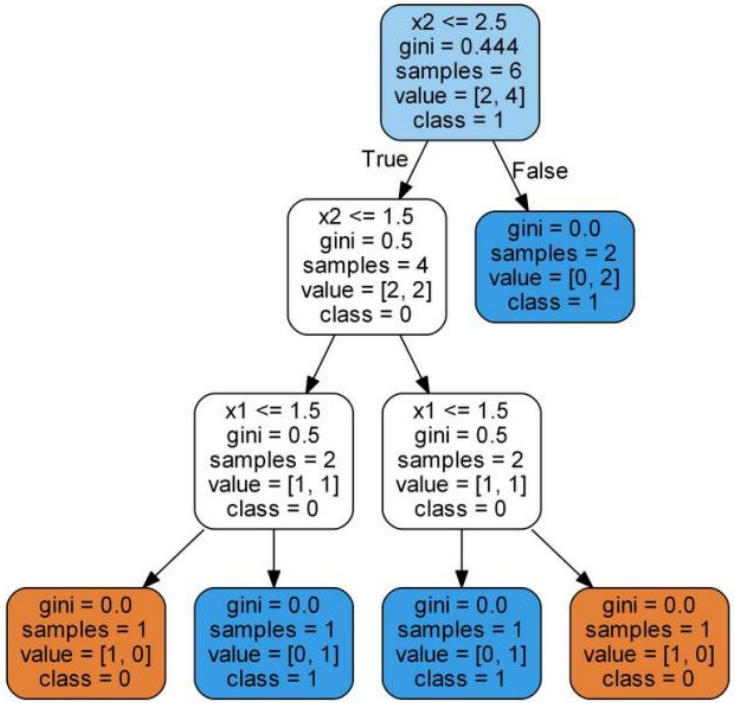
# Árboles de decisión

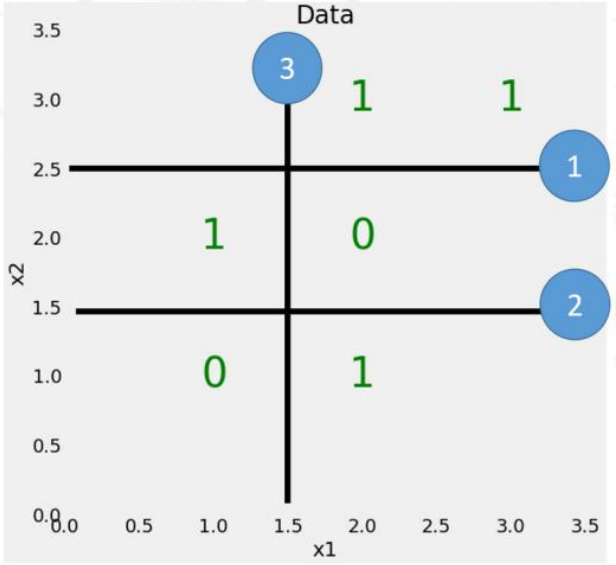


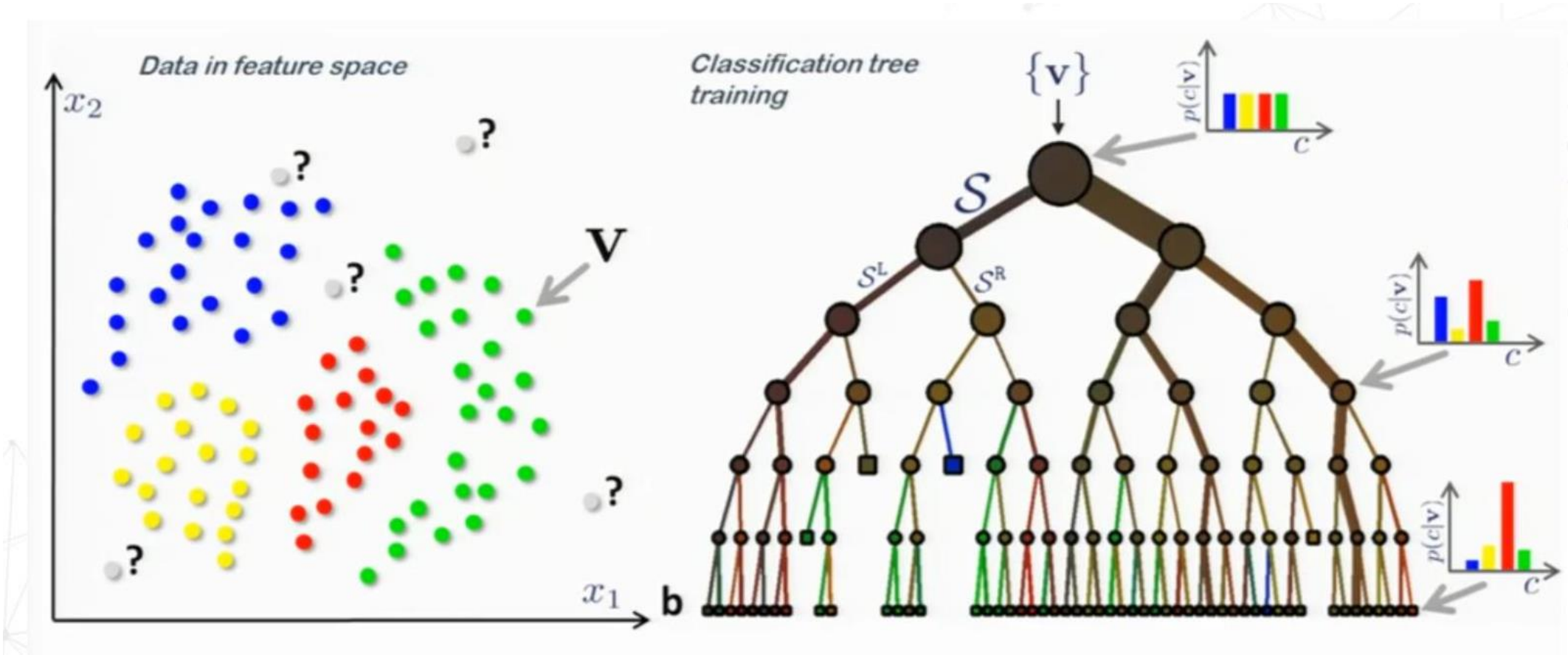
# Ejemplo bi-clase



# Ejemplo multi-clase



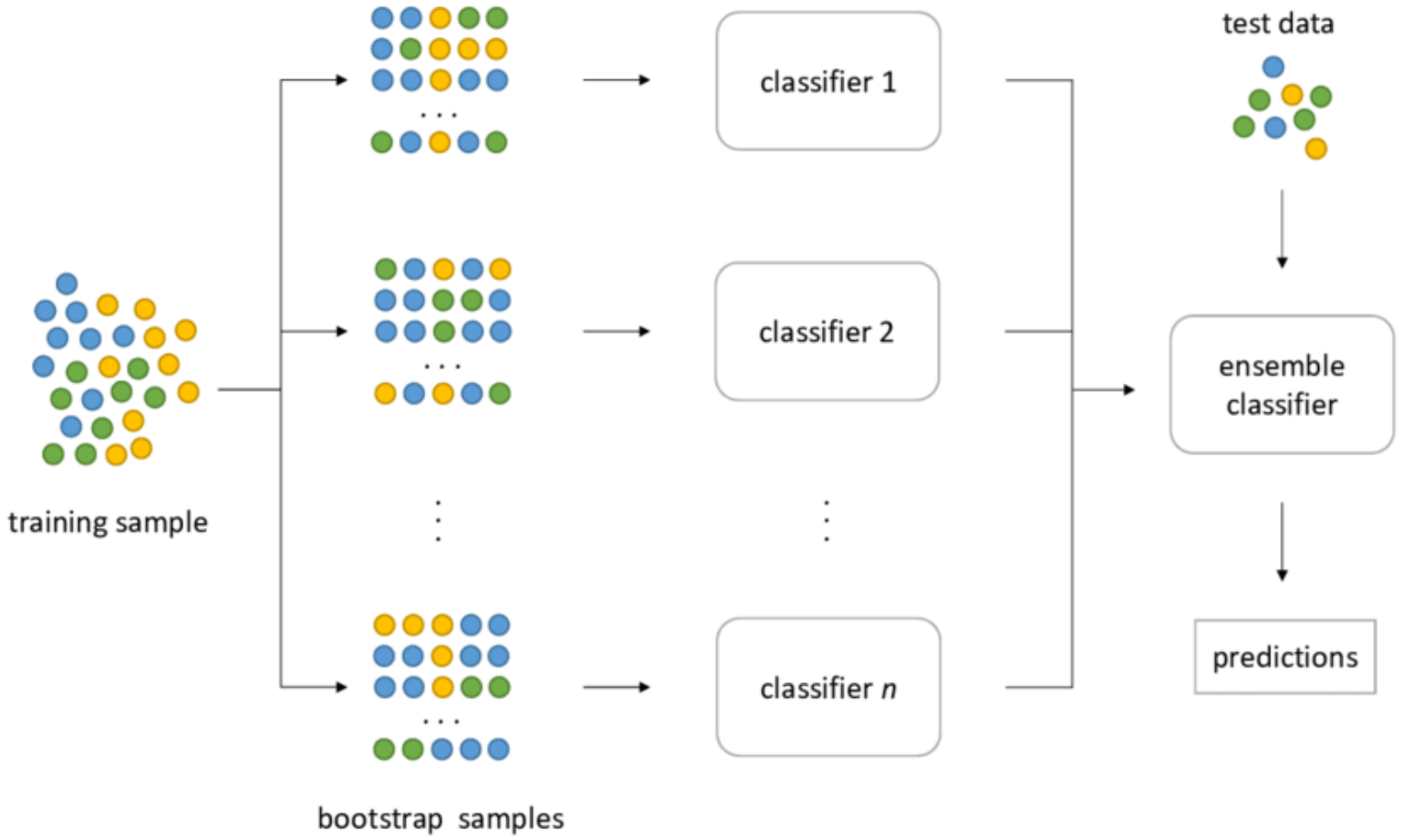
# Bootstrap Aggregating (Bagging)

# Bagging

Bagging (bootstrap aggregating) es uno de los métodos más simples de ensamble. La idea es tomar varios clasificadores simples y entrenar cada uno con un subconjunto de los datos. Finalmente la predicción para un ejemplo va a ser:

- El promedio de las predicciones de todos los clasificadores simples en el caso de un problema de regresión.
- La clase con el mayor número de votos entre todos los clasificadores en el caso de una clasificación.

# Bagging



# Bagging

Un ensamble basado en bagging crea los subconjuntos de datos para cada clasificador usando un método conocido como **bootstrapping.** De manera que el algoritmo se puede resumir en los siguientes pasos:

1. Para cada uno de los modelos simples:

    a. Cree un subconjunto de entrenamiento usando una muestra del conjunto de entrenamiento (tomada aleatoriamente con reemplazo). Puede ser un porcentaje definido.

    b. Entrene el modelo con el subconjunto de datos muestreado.

2. Para realizar inferencia:
    Promedie el resultado de todos los modelos si es regression.
    Haga votación de todos los modelos si es clasificación (escoja la moda).

# Bagging

El bagging tiene las siguientes características:

1. Trata de resolver problemas de sobreajuste.
2. En un ensamble paralelo, es decir que cada modelo es entrenado independiente del otro.
3. Suele ser homogéneo, es decir que se entrena el mismo tipo de clasificadores simples, aunque no hay una razón estricta para no entrenar diferentes.
4. Un ensamble de Árboles de Decisión usualmente se llama **Random Forest**.
5. En bagging también se puede hacer submuestreo de las variables de entrada; así lo hace Random Forest.
6. No funcionan bien con modelos lineales.

# Boosting

# Boosting

Boosting engloba a una familia de algoritmos cuya idea general es tomar modelos sencillos (por lo general árboles de decisión) y mejorar sus predicciones de manera secuencial.

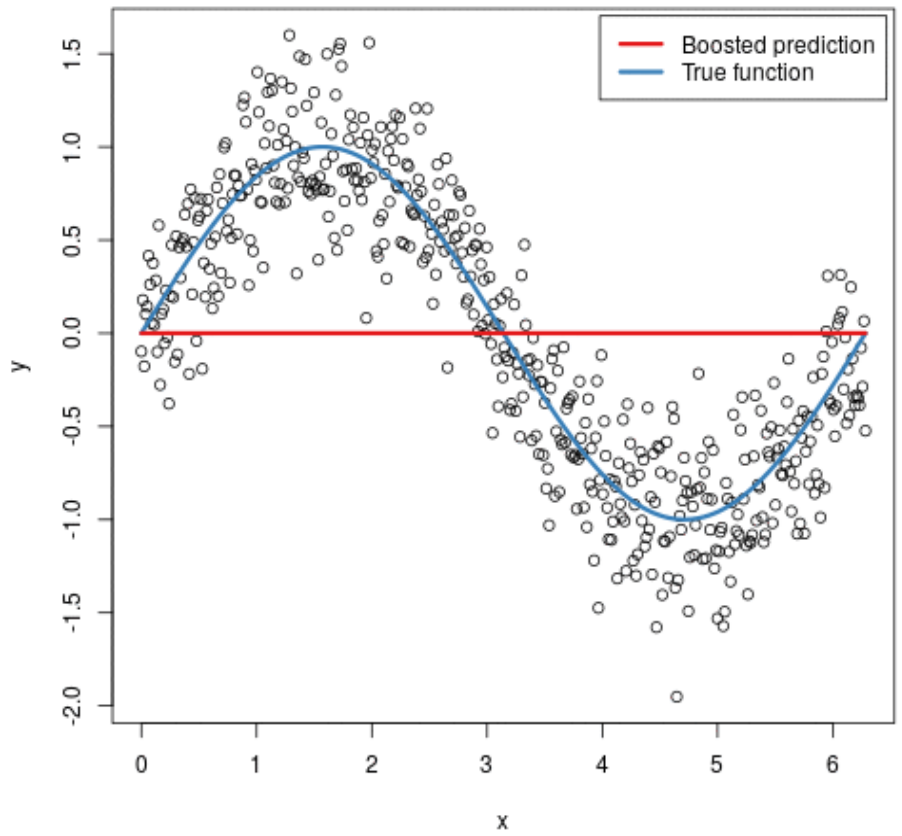
Para mejorar  esas predicciones el algoritmo entrena cada modelo secuencialmente con todos los datos y, para cada nuevo modelo, se le da más peso a los datos que no fueron bien clasificados o cuyo error en regresión sea más alto.

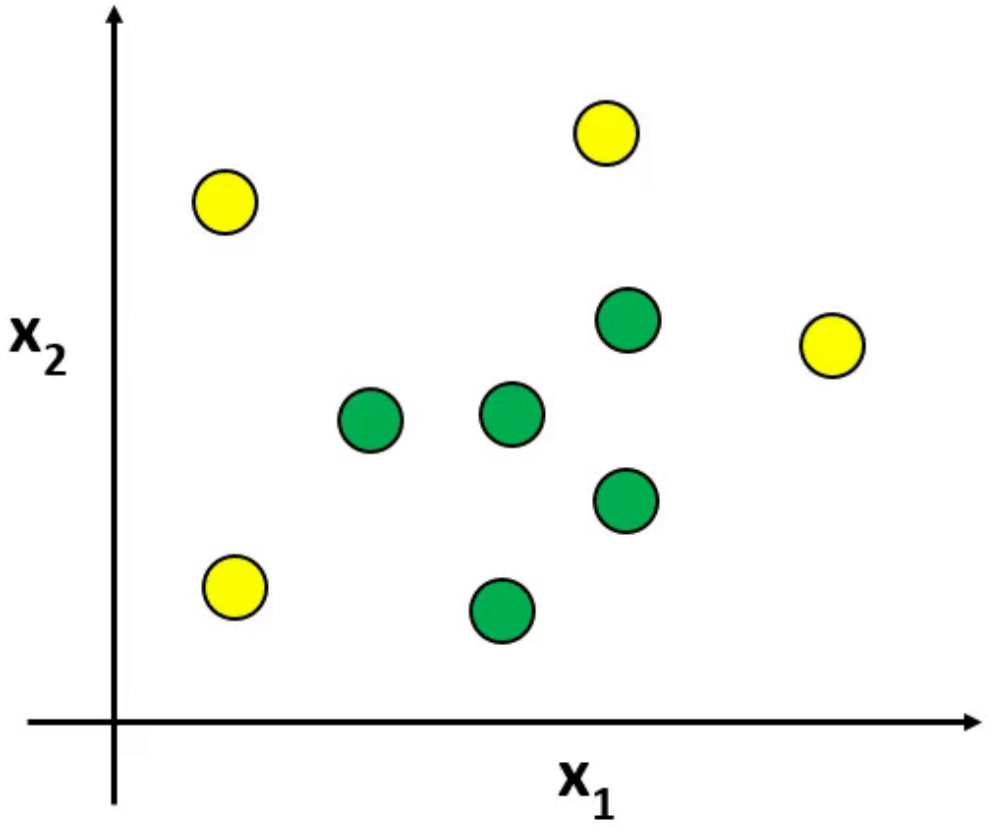
# Boosting

Finalmente la predicción será un **promedio ponderado** de todos los clasificadores base en el caso de regresión o una **votación ponderada** en el caso de clasificación.

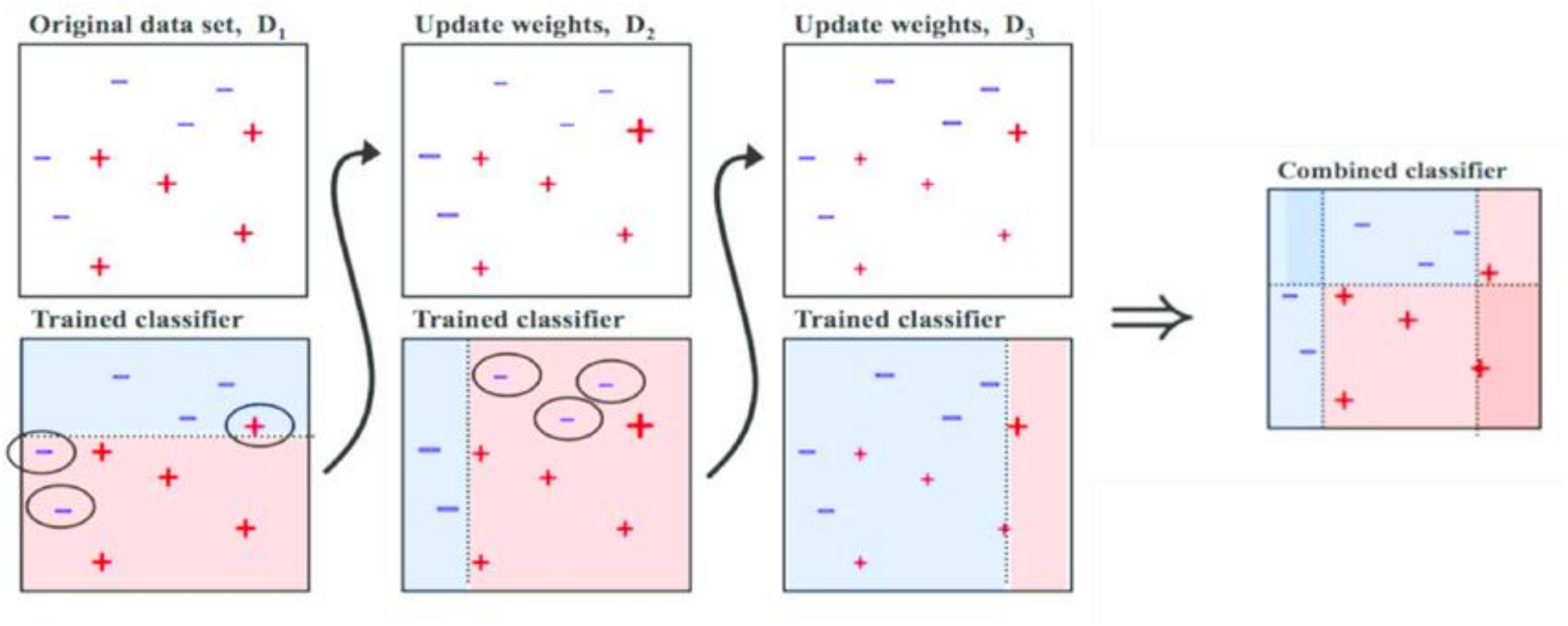
A diferencia del bagging, el boosting es secuencial y dependiente. Es decir, el modelo en la iteración actual depende de las predicciones en la iteración anterior.

# Boosting



# Boosting

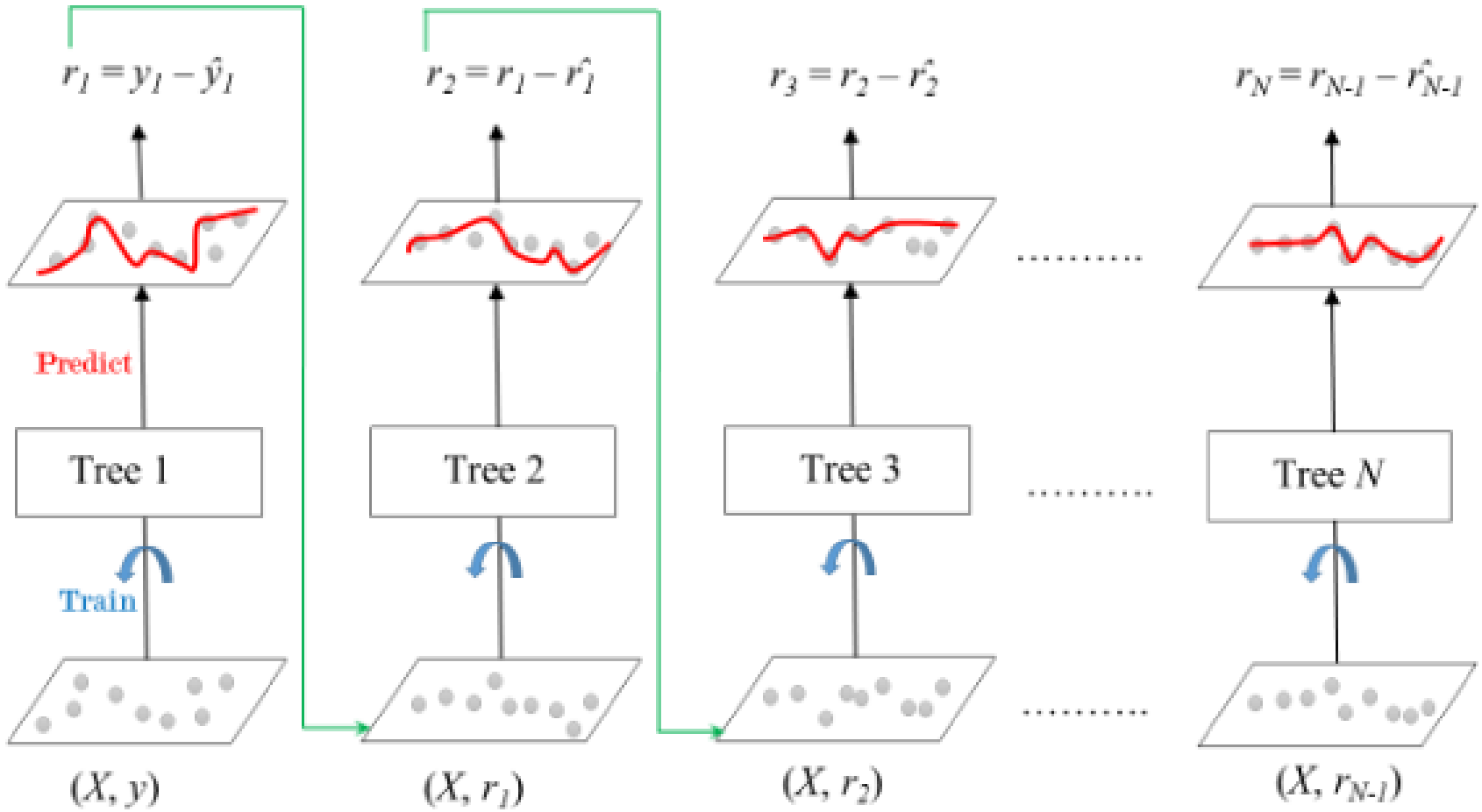


# Gradient Boosting

Gradient boosting es un algoritmo que generaliza la idea del boosting para tratarlo como un problema de optimización que se puede solucionar para diferentes funciones de pérdida y con un método similar al descenso por el gradiente.

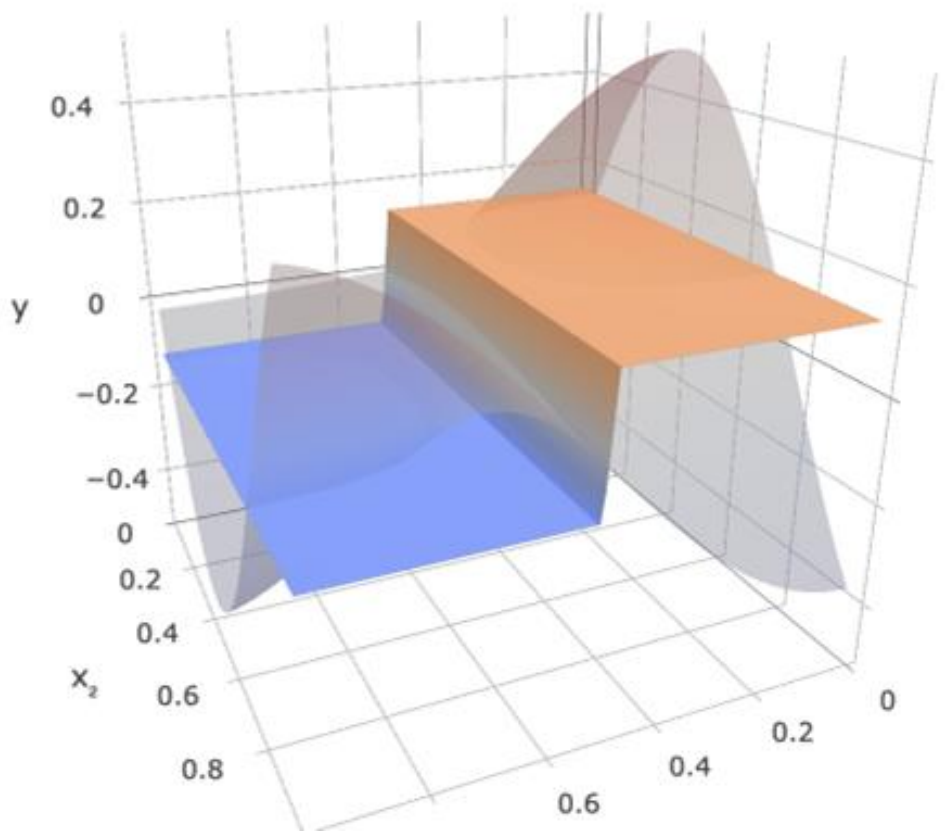
En lugar de darle más peso directamente a los datos con el mayor error, el gradient boosting entrena el siguiente modelo para que minimice el **residual** (la diferencia entre las etiquetas y las predicciones del ensamble actual) o para que **se ajuste al gradiente de la pérdida**.

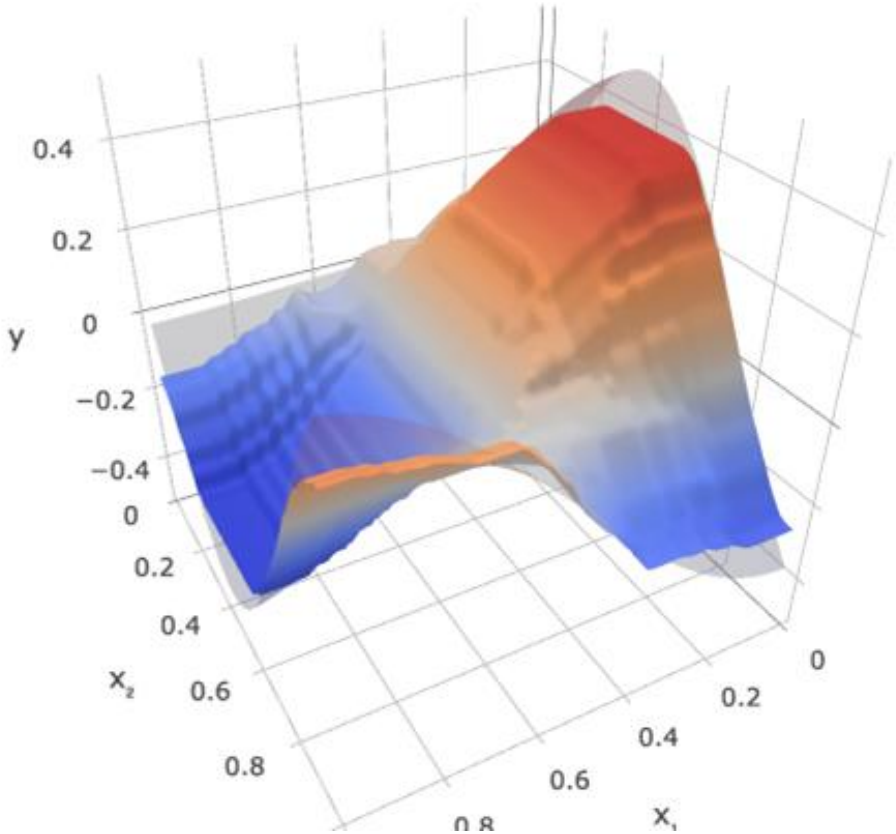
# Gradient Boosting



# Gradient Boosting

http://arogozhnikov.github.io/2016/06/24/gradient_boosting_explained.html



# Ejemplo de random forest y gradient boosting


GRACIAS
UNIVERSIDAD
EAFIT