HENRY

Data Science

M4L5 | Modelos de ensamble

→ soyhenry.com

Objetivos

- Implementar técnicas de balanceo de datos para mejorar la equidad y la precisión en modelos de aprendizaje automático.
- Explorar modelos de ensamble como Bagging, Boosting, Stacking y Voting, analizando su impacto en la optimización de predicciones.
- Comparar estrategias de combinación de modelos, evaluando su aplicación en distintos contextos y su influencia en el rendimiento final.

Agenda

COMENCEMOS →

#TEMAS

.01 Introducción tradeoff-entre sesgo y varianza

.02 Bagging a través de random forest

.03 Boosting a través de XGBoost/GBDT

.04 Otros métodos de ensamble

.05 Homework

<-->

¿Qué vimos en la lecture?

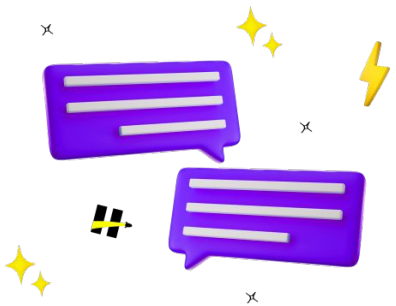
<01>

Introducción tradeoff-entre sesgo y varianza

El dilema del aprendizaje automático

- Los modelos individuales pueden ser simples o complejos, pero ninguno es perfecto.
- Algunos fallan por exceso de simplicidad (alto sesgo).
- Otros fallan por exceso de complejidad (alta varianza).
- Este equilibrio define el **bias-variance trade-off**.

¿Qué es el sesgo?



- Error por suposiciones simplificadas.
- Ignora información relevante del dataset.
- Asociado al **subajuste (underfitting)**.
- Ejemplo: modelar una parábola con una línea recta.
- Produce errores sistemáticos, incluso en entrenamiento.

¿Qué es la varianza?

- Sensibilidad del modelo a pequeñas variaciones en los datos.
- Asociado al **sobreajuste (overfitting)**.
- Aprende también el ruido.
- Ejemplo: árbol muy profundo que memoriza el entrenamiento.
- Excelente desempeño en entrenamiento, pobre en prueba.

La metáfora del blanco de dardos

- **Alto sesgo / Baja varianza →** agrupado pero lejos del centro.
- **Bajo sesgo / Alta varianza →** disperso pero promedio correcto.
- **Bajo sesgo / Baja varianza →** ideal: agrupado y preciso.
- **Alto sesgo / Alta varianza →** disperso y equivocado.

El trade-off

- Disminuir uno suele aumentar el otro.
- Modelos simples → bajo varianza, alto sesgo.
- Modelos complejos → bajo sesgo, alta varianza.
- El objetivo: **minimizar el error total de generalización.**

Error total

Error total= Sesgo^2 + Varianza + Error irreducible

- El error irreducible es el ruido natural del problema.

- Sesgo y varianza pueden controlarse con:

    - Selección de modelo
    - Ajuste de hiperparámetros
    - Métodos de ensamble

<02>

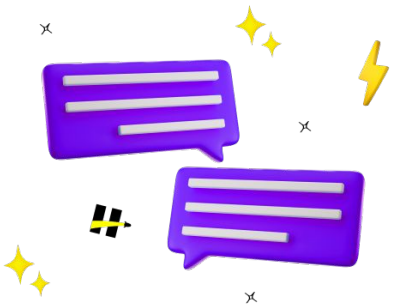
Bagging a través de random forest

Qué es Bagging

- Significa **Bootstrap Aggregating**.
- Reduce varianza en modelos inestables (como árboles).
- Pasos:
  - Crear subconjuntos del dataset (bootstrap).
  - Entrenar un modelo por subconjunto.
  - Promediar o votar las predicciones.

Qué es Bootstrap

- Muestreo aleatorio con reemplazo.
- Cada modelo ve un conjunto distinto de datos.
- Algunos registros se repiten; otros no aparecen.
- Promueve diversidad entre modelos.



Out-of-Bag (OOB) Error

- Cada muestra deja fuera ≈ 37 % de los datos.
- Esos datos sirven como validación interna.
- Permite estimar el error sin conjunto de test.

Random Forest

- Implementación más popular del Bagging.
- Entrena múltiples árboles sobre distintas muestras.
- Además varía las **features** usadas en cada división.
- Aumenta diversidad y reduce varianza.

Hiperparámetros principales

- ★ **Clasificación →** votación mayoritaria.
- ★ **Regresión →** promedio de predicciones.
- ★ **Los errores individuales se compensan.**
- ★ **Resultado:** modelo más estable y generalizable.

Ajustar estos valores mejora la generalización del modelo.

Hiperparámetros esenciales

- n_estimators → cantidad de árboles.
- min_samples_leaf → tamaño mínimo de hoja.
- max_depth → profundidad máxima.
- max_features → número de variables por split.
- bootstrap → uso de muestreo con reemplazo.

<03>

Boosting a través de XGBoost/GBDT

Qué es Boosting

- Entrena modelos **secuencialmente**.
- Cada modelo corrige los errores del anterior.
- Crea predictores fuertes a partir de modelos débiles.

Cómo funciona el Gradient Boosting

- Entrena un modelo inicial.
- Calcula errores (residuos).
- Entrena el siguiente modelo sobre esos errores.
- Suma las predicciones.
- Repite hasta que el error deje de mejorar.

XGBoost: versión optimizada

- Usa regularización L1 y L2.
- Optimización de segundo orden (usa gradiente y Hessiano).
- Maneja valores faltantes automáticamente.
- Entrenamiento paralelo y poda inteligente.

<04>

Otros métodos de ensamble

Más allá de Bagging y Boosting

- Otras estrategias combinan modelos distintos.
- Objetivo: aprovechar fortalezas diversas.
- Ejemplos: Voting, Stacking, Blending.

Voting

- Cada modelo emite un voto.
- **Hard Voting:** votación por mayoría.
- **Soft Voting:** promedio de probabilidades.
- Simple, reduce varianza, pero no aprende de errores.

Stacking

- Entrena varios modelos base.
- Sus predicciones se usan como nuevas features.
- Un meta-modelo aprende cómo combinarlas.
- Aprovecha lo mejor de cada algoritmo.

Blending

- Variante práctica del stacking.
- Usa un conjunto de validación (hold-out).
- Más simple y rápida, pero menos precisa.

<DATA SCIENCE/>

Vayamos a la práctica

Homework

Consigna

Entrenar y comparar dos métodos de ensamble —Random Forest Regressor y XGBoost Regressor— en un problema de regresión multivariable: predecir el consumo energético de edificios (Heating Load y Cooling Load) a partir de sus características estructurales.

Tareas a realizar

1. **Carga y exploración de datos**
   - Importar el dataset, visualizar sus dimensiones y explorar las variables.
   - Verificar si existen valores nulos o atípicos.
2. **Preparación del dataset**
   - Separar las variables predictoras (X) de las variables objetivo (Y1 y Y2).
   - Dividir en entrenamiento y prueba (ej: 80% / 20%).
3. **Modelo base: Árbol de decisión**
   - Entrenar un **DecisionTreeRegressor** como punto de comparación.
   - Evaluar con métricas de regresión (**RMSE, MAE, R^2**) sobre Heating y Cooling Load.
4. **Random Forest Regressor**
   - Entrenar un Random Forest con diferentes valores de `n_estimators` y `min_samples_leaf`.
   - Comparar resultados con el árbol individual.
   - Analizar la importancia de variables (`feature_importances_`).

Tareas a realizar

5. **XGBoost Regressor**

   - Entrenar un XGBoost ajustando `n_estimators`, `max_depth` y `learning_rate`.
   - Comparar con Random Forest en términos de métricas.

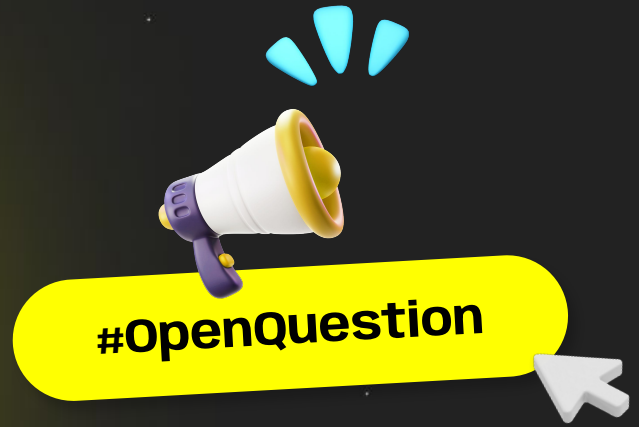
6. **Comparación de resultados**

   - Crear una tabla comparativa con todas las métricas para Árbol, Random Forest y XGBoost.
   - Representar visualmente los resultados con un barplot de RMSE y R^2.

7. **Reflexión final**

   - ¿Qué modelo recomendarías para un caso real de predicción energética?
   - ¿Cómo influye el trade-off sesgo/varianza en Random Forest y en XGBoost?
   - ¿Qué ventajas y desventajas encontraste en cada técnica?

HENRY



¿Preguntas?

HENRY

¡Muchas gracias!

→ soyhenry.com