Ensambles

Técnicas de Aprendizaje de Máquina

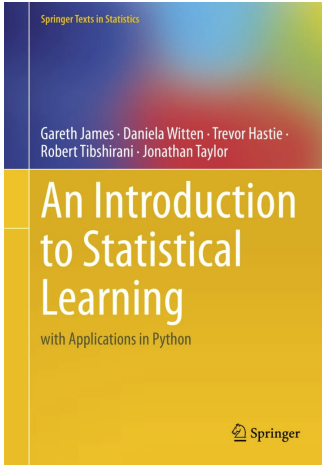
Oscar Bustos

August 24, 2025

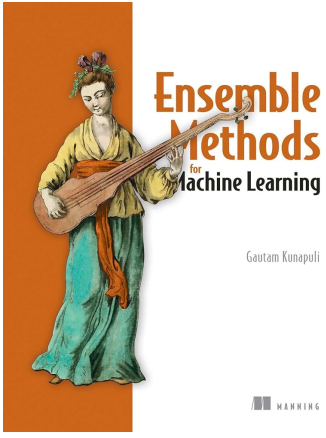
Table of Contents

Capítulos sobre Ensambles

1 Intuición Matemática



Capítulo 9

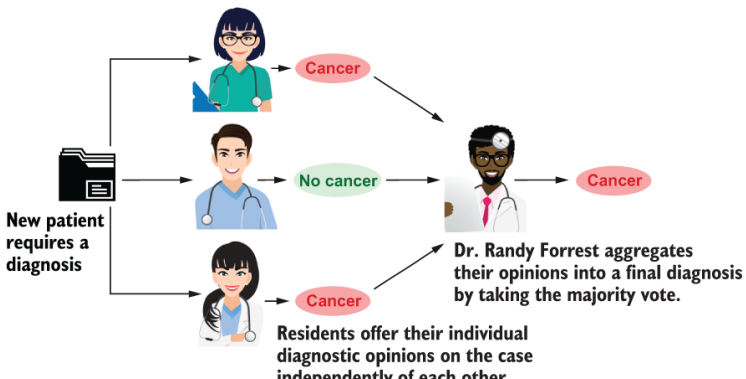


Capítulos 1,2,3,5

Wisdom of Crowds

1 Intuición Matemática

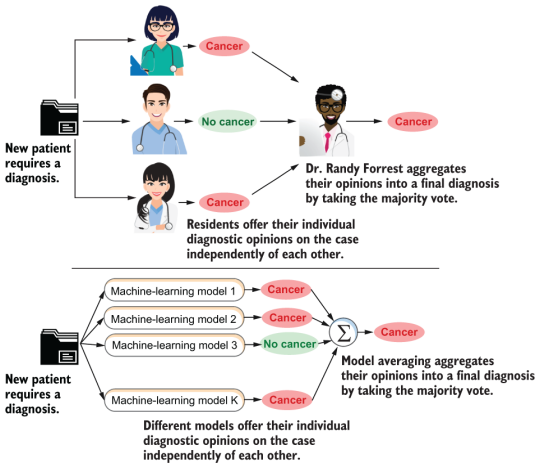
¿Cómo diagnostico si un paciente tiene cáncer o no?



Wisdom of Crowds

1 Intuición Matemática

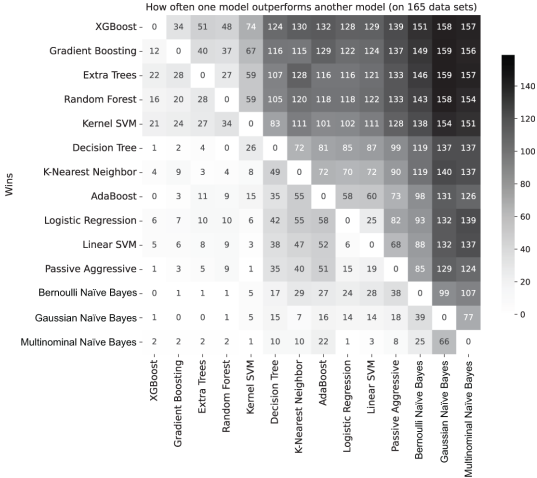
¿Cómo diagnostico si un paciente tiene cáncer o no usando IA?



Evidencia empírica en los ensambles

1 Intuición Matemática

¿Qué algoritmo de aprendizaje de máquina debo utilizar? Olson et al. 2018



¿Qué son los Ensambles?

1 Intuición Matemática

- Combinación de múltiples modelos más débiles para crear un modelo más fuerte.
- Idea principal: "La sabiduría de la multitud".
- Reducen el sesgo y la varianza.
- Mejoran la precisión y la robustez del modelo.

Resumen de Categorías de Ensambles

Existen tres principales categorías de ensambles en machine learning:

- **Parallel Homogeneous Ensembles:** Múltiples modelos del mismo tipo entrenados con diferentes muestras de datos. Mejora la estabilidad y reduce la varianza.
- **Parallel Heterogeneous Ensembles:** Combinación de diferentes tipos de modelos para aprovechar sus fortalezas individuales.
- **Sequential Gradient Boosting Ensembles:** Modelos entrenados secuencialmente, donde cada uno corrige los errores del anterior, logrando una mejor optimización de la función de pérdida.

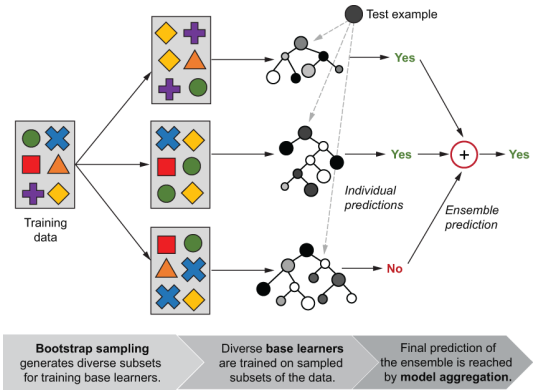
Table of Contents

2 Parallel Homogeneous Ensembles

Arquitectura de Ensamble Homogéneo

2 Parallel Homogeneous Ensembles

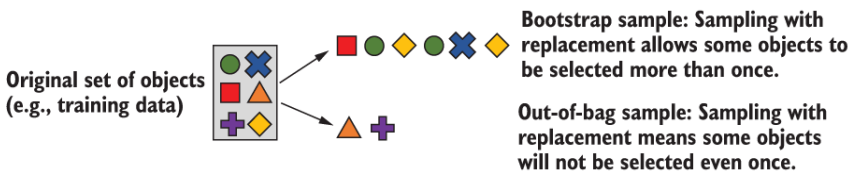
Cuando todos los clasificadores son del mismo tipo



Bootstrapping

2 Parallel Homogeneous Ensembles

El bootstrapping es un método estadístico que permite generar **múltiples subconjuntos de datos** a partir del conjunto de entrenamiento original. Se basa en el muestreo con reemplazo, lo que significa que algunos ejemplos pueden aparecer más de una vez en un subconjunto, mientras que otros pueden no aparecer en absoluto.



Bagging - Entrenamiento

2 Parallel Homogeneous Ensembles

```python
import numpy as np
from sklearn.tree import DecisionTreeClassifier

rng = np.random.RandomState(seed=4190)
def bagging_fit(X, y, n_estimators, max_depth=5, max_samples=200):
    n_examples = len(y)
    estimators = [DecisionTreeClassifier(max_depth=max_depth)
                  for _ in range(n_estimators)]

    for tree in estimators:
        bag = np.random.choice(n_examples, max_samples,
                               replace=True)
        tree.fit(X[bag, :], y[bag])

    return estimators
```

Bagging - Inferencia

2 Parallel Homogeneous Ensembles

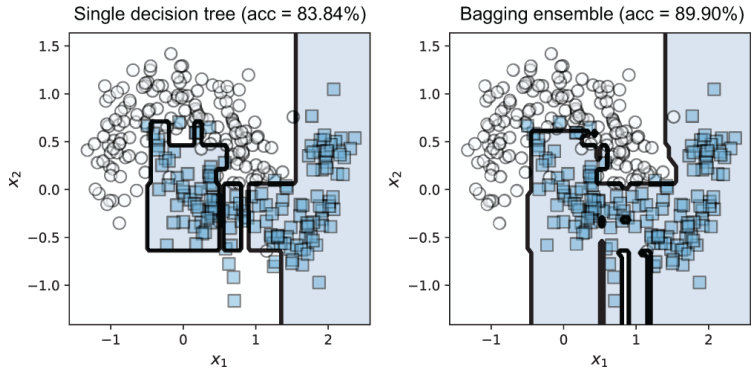
```
from scipy.stats import mode

def bagging_predict(X, estimators):
    all_predictions = np.array([tree.predict(X)
                                for tree in estimators])
ypred, _ = mode(all_predictions, axis=0,
                keepdims=False)
    return np.squeeze(ypred)
```

Comparación Bagging en Árboles de Decisión

2 Parallel Homogeneous Ensembles

Un único árbol de decisión (izquierda) **sobreajusta** el conjunto de entrenamiento y puede ser sensible a los valores atípicos. Un conjunto de bagging (derecha) **suaviza** los efectos de sobreajuste y las clasificaciones erróneas de varios de esos estimadores base.



Bagging en Sklearn

2 Parallel Homogeneous Ensembles

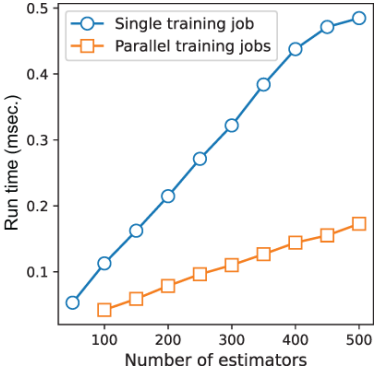
```
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import BaggingClassifier

base_estimator = DecisionTreeClassifier(max_depth=10)
bag_ens = BaggingClassifier(base_estimator=base_estimator,
                            n_estimators=500,
                            max_samples=100,
                            oob_score=True,
                            random_state=rng)
bag_ens.fit(Xtrn, ytrn)
ypred = bag_ens.predict(Xtst)
```

Optimización en Bagging

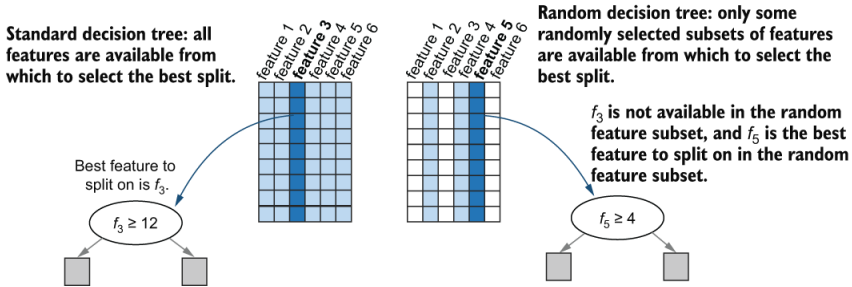
2 Parallel Homogeneous Ensembles

```
BaggingClassifier(base_estimator=DecisionTreeClassifier(),
                                  n_estimators=100, max_samples=100,
                                  oob_score=True, n_jobs=-1)
```



Características Aleatorias

2 Parallel Homogeneous Ensembles



Random Forest: Ventajas

2 Parallel Homogeneous Ensembles

- Alta precisión y robustez.
- Maneja bien datos de alta dimensión.
- Reduce el sobreajuste.
- Proporciona importancia de las características.

Random Forest: Hiperparámetros Clave

- `n_estimators`: Número de árboles en el bosque.
- `max_features`: Número de características a considerar en cada división.
- `max_depth`: Profundidad máxima de los árboles.
- `min_samples_split`: Número mínimo de muestras requeridas para dividir un nodo interno.

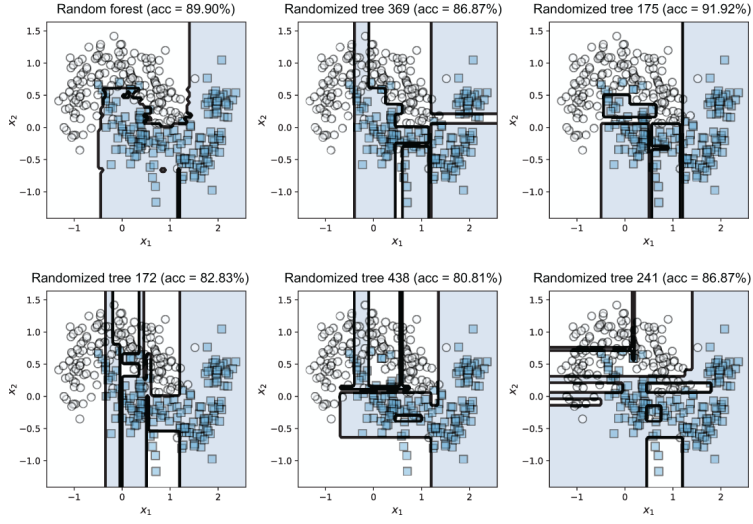
Random Forest en Sklearn


```
from sklearn.ensemble import RandomForestClassifier

rf_ens = RandomForestClassifier(n_estimators=500,
                                max_depth=10,
                                oob_score=True,
                                n_jobs=-1,
                                random_state=rng)
rf_ens.fit(Xtrn, ytrn)
ypred = rf_ens.predict(Xtst)
```

Random Forest en Sklearn

2 Parallel Homogeneous Ensembles



Random Forest en Sklearn

2 Parallel Homogeneous Ensembles

```
X_trn, X_tst, y_trn, y_tst = train_test_split(X, y, test_size=0.15)
n_features = X_trn.shape[1]

rf = RandomForestClassifier(max_leaf_nodes=24,
                            n_estimators=50, n_jobs=-1)
rf.fit(X_trn, y_trn)
err = 1 - accuracy_score(y_tst, rf.predict(X_tst))

importance_threshold = 0.02
for i, (feature, importance) in enumerate(zip(dataset['feature_names'],
                rf.feature_importances_)):
    if importance > importance_threshold:
        print('[{0}] {1} (score={2:4.3f})'.
              format(i, feature, importance))
```

Random Forest en Sklearn

2 Parallel Homogeneous Ensembles

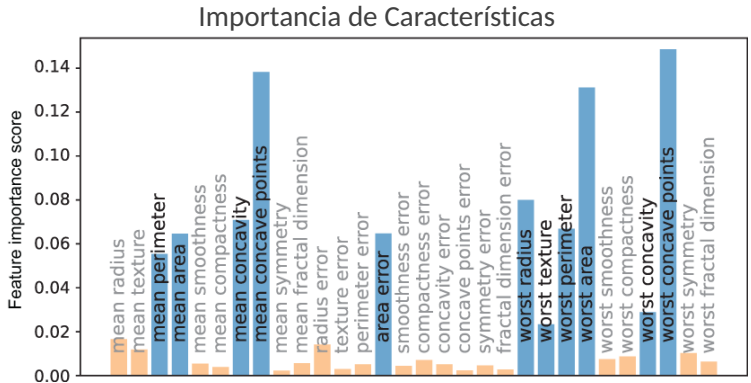


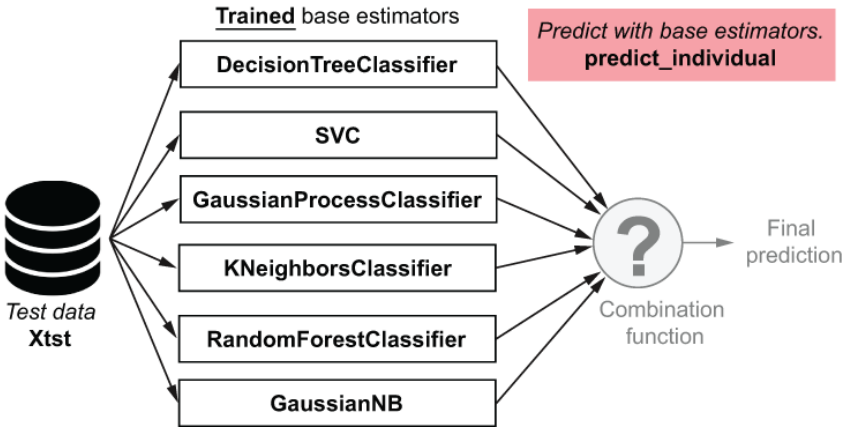
Table of Contents

3 Parallel Heterogeneous Ensembles

Arquitectura de Ensamble Heterogéneo

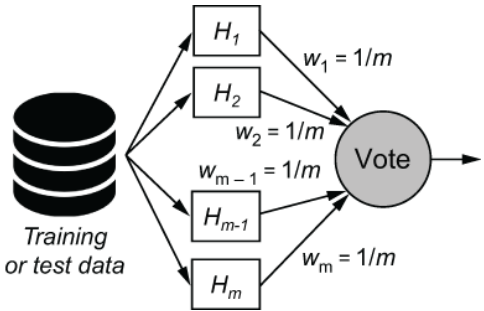
3 Parallel Heterogeneous Ensembles

Cuando todos los clasificadores son de tipos distintos

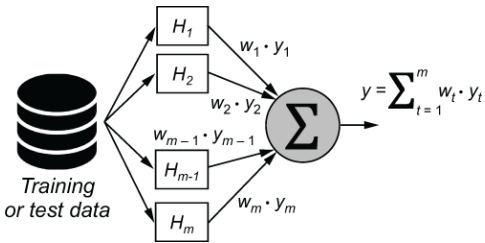


Voting

3 Parallel Heterogeneous Ensembles



Votación

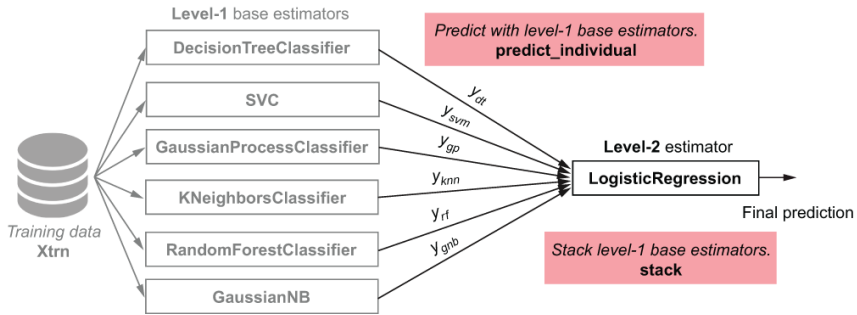


Votación Ponderada

Stacking

3 Parallel Heterogeneous Ensembles

Entrenando un modelo que usa como entrada las salidas de otro modelo



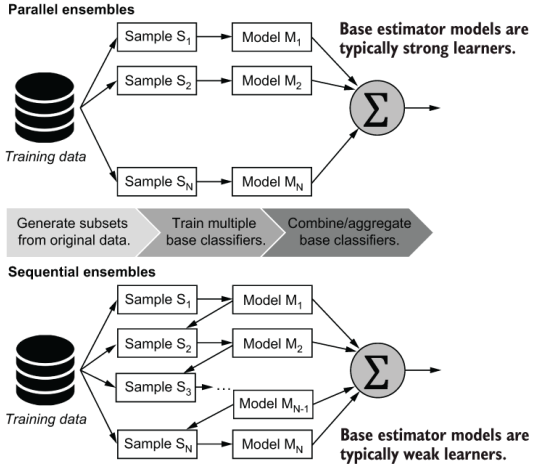
Pontificia Universidad JAVERIANA Bogotá

Table of Contents

4 Sequential Gradient Boosting Ensembles

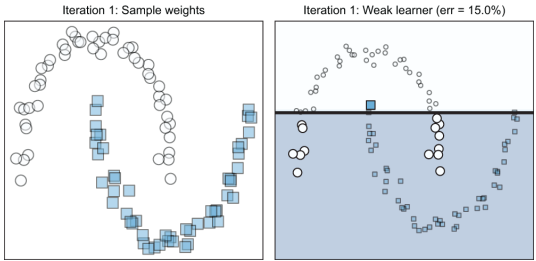
Arquitectura de Gradient Boosting

4 Sequential Gradient Boosting Ensembles



Gradient Boosted Trees (GBT): Intuición

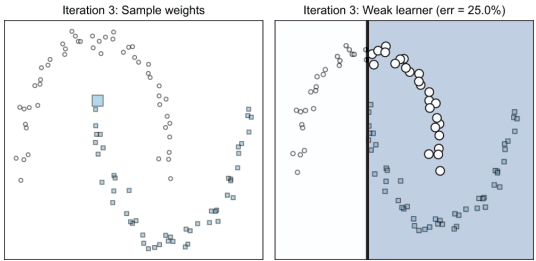
4 Sequential Gradient Boosting Ensembles



- Construye modelos de forma secuencial, donde cada nuevo modelo se enfoca en corregir los errores del modelo anterior.
- Utiliza el gradiente descendente para minimizar la función de pérdida.

Gradient Boosted Trees (GBT): Intuición

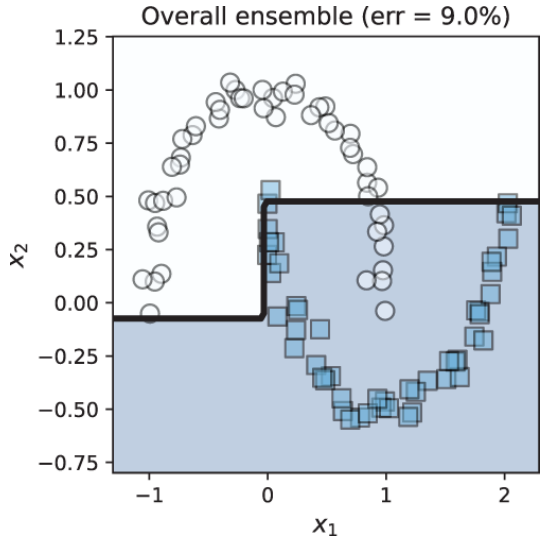
4 Sequential Gradient Boosting Ensembles



- Construye modelos de forma secuencial, donde cada nuevo modelo se enfoca en corregir los errores del modelo anterior.
- Utiliza el gradiente descendente para minimizar la función de pérdida.

Gradient Boosted Trees (GBT): Intuición

4 Sequential Gradient Boosting Ensembles



Gradient Boosted Trees (GBT): Hiperparámetros Clave

- `n_estimators`: Número de árboles en el modelo.
- `learning_rate`: Tasa de aprendizaje, controla el impacto de cada árbol.
- `max_depth`: Profundidad máxima de los árboles.
- `subsample`: Fracción de muestras utilizadas para entrenar cada árbol.
- `colsample_bytree`: Fracción de características utilizadas para entrenar cada árbol.

Table of Contents

5 Comparación

Random Forest vs Gradient Boosted Trees

5 Comparación

Característica	Random Forest	Gradient Boosted Trees
Enfoque	Bagging	Boosting
Paralelización	Sí	No (secuencial)
Precisión	Alta	Muy alta
Sobreajuste	Menor riesgo	Mayor riesgo (requiere ajuste cuidadoso)
Velocidad	Rápido	Más lento (depende de la implementación)
Importancia de características	Sí	Sí

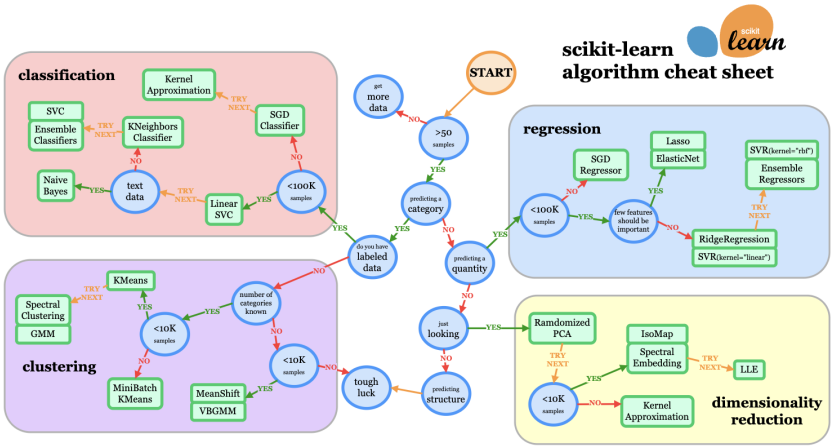
Table of Contents

6 Librerías

Seleccionando Algoritmos

6 Librerías

¿Cómo elijo el algoritmo más adecuado para mis datos?



Ensambles en Python

6 Librerías

- **Random Forest:** `sklearn.ensemble.RandomForestClassifier`, `sklearn.ensemble.RandomForestRegressor`
- **Gradient Boosting:** `sklearn.ensemble.GradientBoostingClassifier`, `sklearn.ensemble.GradientBoostingRegressor`
- **XGBoost:** `xgboost`
- **LightGBM:** `lightgbm` (Microsoft)
- **CatBoost:** `catboost`

Ejemplo de Código: Random Forest (scikit-learn)

6 Librerías

```python
from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size
    =0.2)

rf = RandomForestClassifier(n_estimators=100, max_depth=5)
rf.fit(X_train, y_train)

y_pred = rf.predict(X_test)
accuracy = accuracy_score(y_test, y_pred)
print("Accuracy:", accuracy)
```

Ejemplo de Código: Gradient Boosting (XGBoost)

6 Librerías

```python
import xgboost as xgb
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2)

xgb_model = xgb.XGBClassifier(n_estimators=100, max_depth=5, learning_rate=0.1)
xgb_model.fit(X_train, y_train)

y_pred = xgb_model.predict(X_test)
accuracy = accuracy_score(y_test, y_pred)
print("Accuracy:", accuracy)
```

References

7 Librerías

James, Gareth et al. (2023). *An introduction to statistical learning: With applications in python*. Springer Nature.

Kunapuli, Gautam (2023). *Ensemble methods for machine learning*. Simon and Schuster.

Olson, Randal S et al. (2018). "Data-driven advice for applying machine learning to bioinformatics problems". In: *Pacific symposium on biocomputing 2018: Proceedings of the pacific symposium*. World Scientific, pp. 192–203.