

# XGBoost

# Ensemble Methods Overview

- 1. Bagging:** Combines predictions from multiple models trained on different random subsets of the data to reduce variance.
- 2. Boosting:** Trains models sequentially, where each model focuses on correcting the errors of the previous one, reducing bias.
- 3. Stacking:** Combines the outputs of multiple models (even different types) into a "meta-model" for better predictions.

**XGBoost** קיצור של *Extreme Gradient Boosting* הוא אחד האלגוריתמים החזקים ללמידת מכונה.

הוא מבוסס על עצי החלטה ועובד בטכניקה של **Boosting** כלומר, בניית סדרת מודלים פשוטים (לרוב עצים קטנים) שמתקנים אחד את הטעויות של השני.

- הוא **מדויק מאוד** ומנצח בתחרויות Kaggle
- מתאים גם לבעיות **סיווג** וגם לבעיות **רגרסיה**
- מתמודד היטב עם **נתונים חסרים**, **פיצ'רים קטגוריים** ו-**Overfitting**

**eval\_metric**

**מטרה:** לקבוע איך XGBoost ימדוד ביצועים במהלך

האימון.

**אפשרויות נפוצות:**

• 'logloss' מתאים ל- binary / multi-class

הסתברויות

• 'error' אחוז טעויות (0-1)

• log loss – 'mlogloss' למספר קלאסים

• 'mae', 'rmse' לרגרסיה

## objective

**מטרה:** להגדיר את סוג הבעיה שהמודל פותר.

## אפשרויות נפוצות:

'binary:logistic' סיווג בינארי עם הסתברויות

'multi:softprob' סיווג מרובה מחלקות עם הסתברויות

'multi:softmax' סיווג מרובה מחלקות עם תחזית סופית

'reg:squarederror' רגרסיה רגילה

**tree\_method**

**tree\_method**

**מטרה:** לקבוע איך לבנות את עצי ההחלטה (איזה אלגוריתם).

**אפשרויות נפוצות:**

'auto' ברירת מחדל (יבחר לפי הדאטה)

'exact' חישוב מדויק, איטי (לדאטה קטן)

'approx' חישוב מקורב, מהיר יותר

'hist' מהיר במיוחד לדאטה גדול, תומך

ב-categorical

'gpu\_hist' מהיר מאוד על GPU

פרמטר	תפקיד	ערכים נפוצים	הערות שימושיות
objective	מגדיר את סוג הבעיה	'binary:logistic', 'multi:softprob', 'reg:squarederror'	חובה להתאים לבעיה (סיווג/רגרסיה)
eval_metric	מדד להערכת ביצועי המודל	'logloss', 'mlogloss', 'rmse', 'mae'	משפיע רק על מדידות בזמן אימון
tree_method	שיטה לבניית עצים	'auto', 'hist', 'approx', 'gpu_hist'	'hist' תומך בקטגוריים ומאיץ ביצועים
n_estimators	מספר העצים	100–1000	יותר עצים = מודל חזק יותר (ולעיתים איטי יותר)
max_depth	עומק מקסימלי לכל עץ	3–6	עומק גדול מדי = סכנת overfitting
learning_rate	קצב הלמידה של כל עץ	0.01–0.3	קטן = למידה איטית ובטוחה
subsample	אחוז דגימת השורות לכל עץ	0.5–1.0	עוזר להכללה ומניעת overfitting
colsample_bytree	אחוז דגימת העמודות לכל עץ	0.5–1.0	טוב אם יש הרבה פיצ'רים
gamma	מינימום רווח נדרש לפיצול	0, 1, 5	מעלה = פחות פיצולים = מודל פשוט יותר
reg_alpha	רגולריזציית L1 (עונש על גודל משקלים)	0, 0.1, 1	מעודד משקלים = 0 (sparsity)
reg_lambda	רגולריזציית L2	1, 5, 10	עוזר ליציבות המודל
early_stopping_rounds	עצירה אוטומטית אם אין שיפור	10–50	עובד עם eval_set, מונע overfitting