**הסבר על אלגוריתם XGBoost**

**מבוא**

XGBoost (Extreme Gradient Boosting) הוא אלגוריתם למידת מכונה מבוסס עצים שמשתמש בשיטת **Gradient Boosting** לשיפור דיוק התחזיות. האלגוריתם בנוי כך שהוא ממזער שגיאות באופן הדרגתי על ידי הוספת עצים חדשים שמתמקדים בדוגמאות עם שגיאות גבוהות יותר.

**איך האלגוריתם עובד?**

1. **אתחול המודל** – האלגוריתם מתחיל עם תחזיות בסיסיות, כמו ממוצע התוויות.
2. **יצירת עצים ברצף** – כל עץ חדש לומד משגיאות העצים הקודמים, תוך התאמתו כדי לתקן את השגיאות.
3. **חישוב גרדיאנט** – האלגוריתם מחשב את השגיאות של התחזיות הקודמות כדי לקבוע כיצד לשפר את המודל.
4. **שקלול העצים** – כל עץ מקבל משקל בהתאם ליכולתו לתקן שגיאות של קודמיו.
5. **סכימת התחזיות** – התחזיות של כל העצים משוקללות יחד לקבלת התחזית הסופית.

**יתרונות של XGBoost**

✅ **מהירות ויעילות** – האלגוריתם מהיר מאוד בזכות פרלליות פנימית ואופטימיזציות של שימוש בזיכרון. ✅ **מניעת אוברפיטינג** – כולל מנגנונים כמו regularization (L1, L2) למניעת התאמה יתרה. ✅ **תמיכה בנתונים חסרים** – האלגוריתם מסוגל להתמודד עם ערכים חסרים ללא צורך בהשלמתם מראש. ✅ **גמישות** – מתאים לבעיות רגרסיה, סיווג ודירוג (Ranking).

**חסרונות של XGBoost**

❌ **רגישות לאוברפיטינג** – אם לא מכוונים היטב את הפרמטרים, המודל עלול ללמוד את רעשי האימון ולא להכליל היטב. ❌ **דרישות חישוביות גבוהות** – השימוש בהרבה עצים ופרמטרים מתוחכמים מצריך זמן חישוב וזיכרון גבוהים. ❌ **קשה להבנה** – לעומת מודלים פשוטים כמו רגרסיה לינארית, קשה להסביר למשתמשים כיצד המודל מקבל החלטות.

**הסבר על הפרמטרים שנבחרו**

במהלך האימון, השתמשנו ב-**GridSearchCV** כדי למצוא את הפרמטרים האופטימליים עבור האלגוריתם. להלן הסבר על כל פרמטר:

* **n\_estimators (מספר עצים)** – קובע את מספר העצים במודל. ערך גבוה מדי עלול לגרום לאוברפיטינג.
* **learning\_rate (קצב למידה)** – מגדיר בכמה כל עץ חדש תורם לתיקון השגיאה. ערך נמוך מדי יאריך את זמן האימון אך ימנע אוברפיטינג.
* **max\_depth (עומק מקסימלי של עצים)** – שולט על מורכבות העצים. עצים עמוקים מדי עלולים לגרום לאוברפיטינג.
* **subsample (תת-מדגם)** – מגדיר את אחוז הדגימות מהנתונים שמשמש בכל סיבוב. ערך נמוך מסייע במניעת אוברפיטינג.
* **colsample\_bytree (יחס התכונות לכל עץ)** – אחוז הפיצ'רים שנבחרים לכל עץ. ערך נמוך מסייע בהפחתת אוברפיטינג.
* **gamma (מינימום ירידת שגיאה לפיצול)** – קובע מתי כדאי לבצע פיצול נוסף בעץ. ערך גבוה ימנע פיצולים מיותרים.
* **min\_child\_weight (משקל מינימלי של צומת בן)** – מונע יצירת צמתים שמכילים מעט מדי נתונים, ובכך מונע אוברפיטינג.

**מסקנה**

XGBoost הוא אחד האלגוריתמים החזקים ביותר ללמידת מכונה, עם יכולות גבוהות לניבוי מדויק ומהיר. עם זאת, יש להגדיר היטב את הפרמטרים כדי למנוע התאמת יתר לנתוני האימון. שימוש נכון ב-**GridSearchCV** ובקריטריונים להפחתת מורכבות המודל מאפשר להגיע לאיזון בין ביצועים טובים לבין יכולת הכללה טובה על נתונים חדשים.