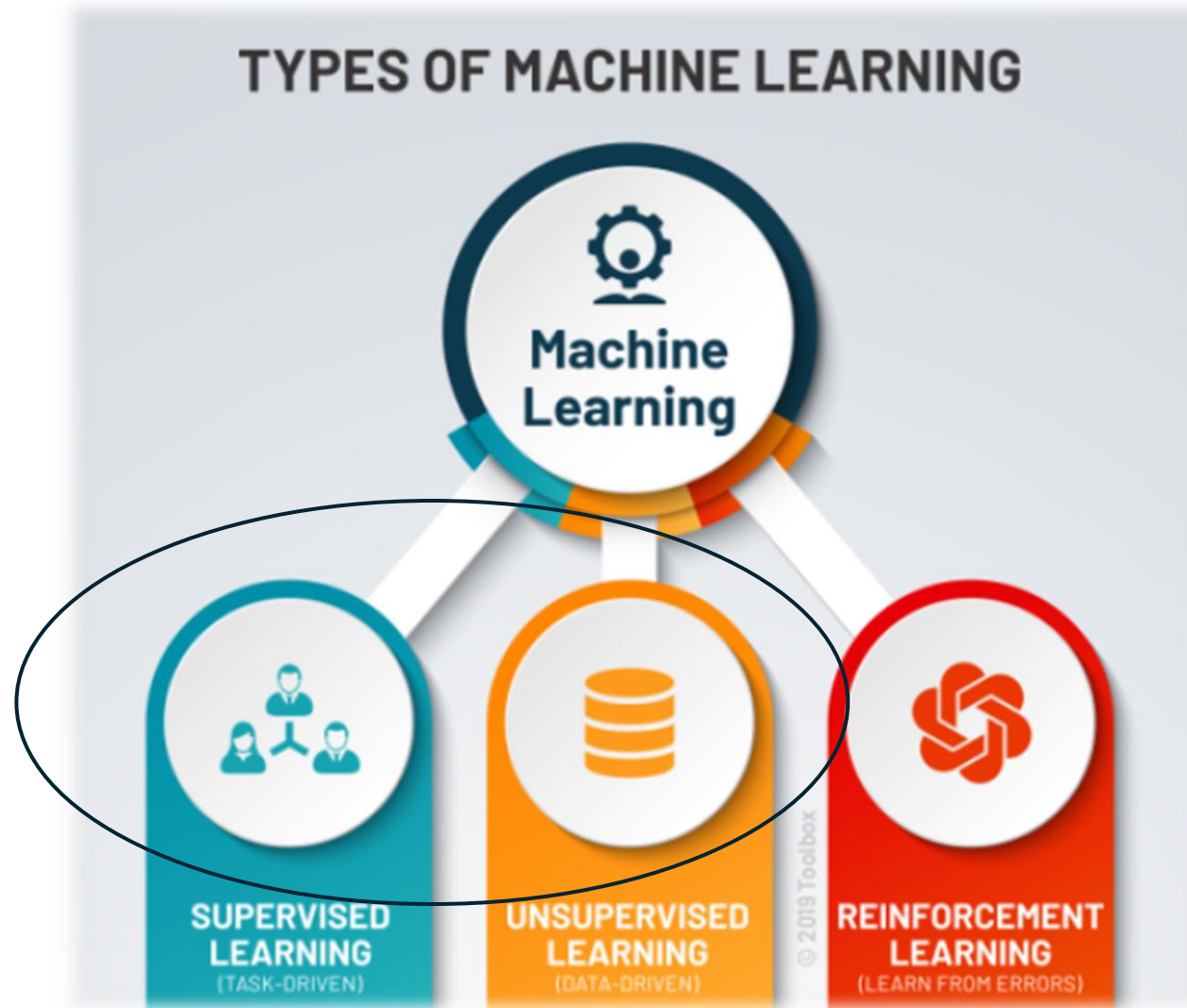
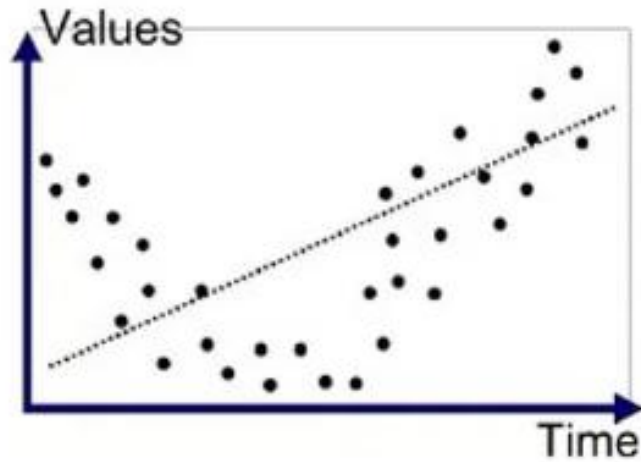


Machine Learning

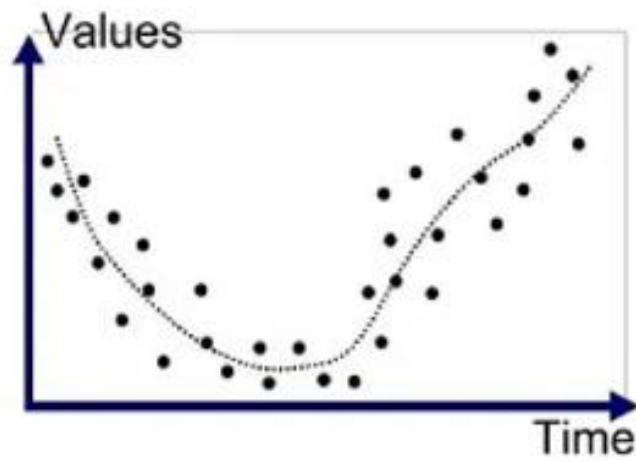
Types of ML Models



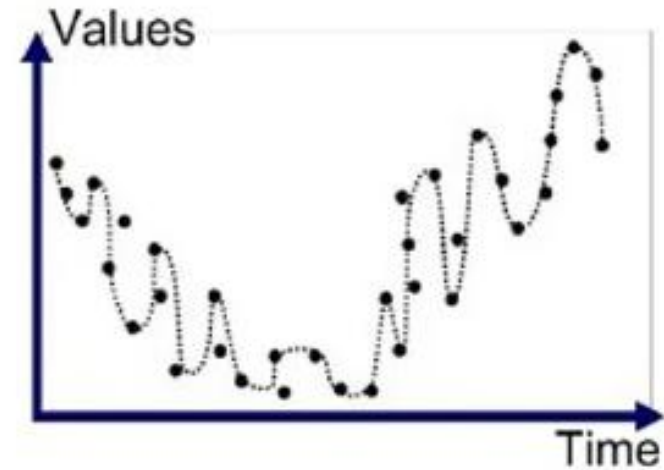
What is underfitting and overfitting in machine learning?



Underfitted

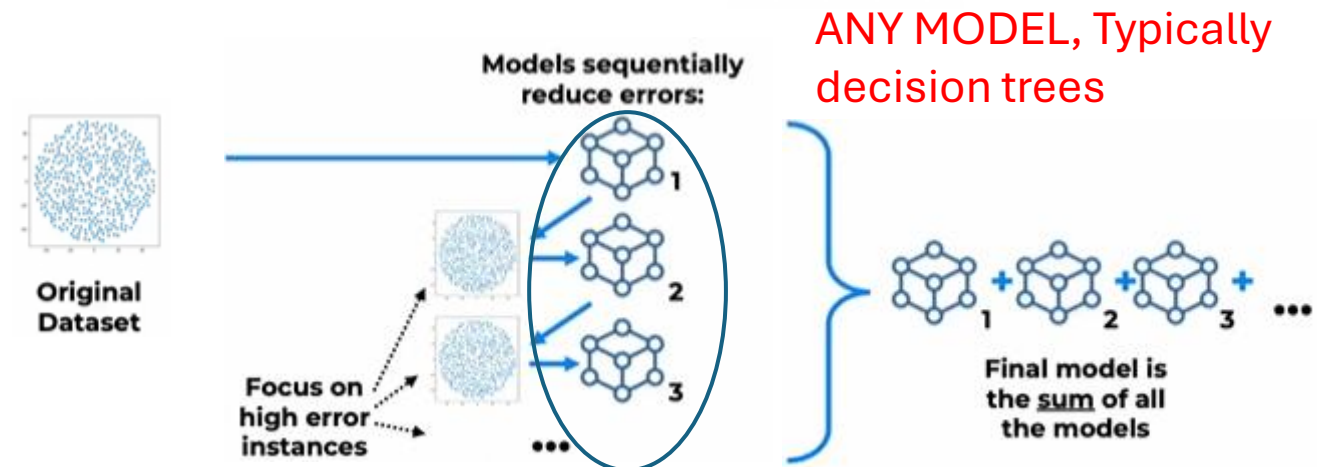
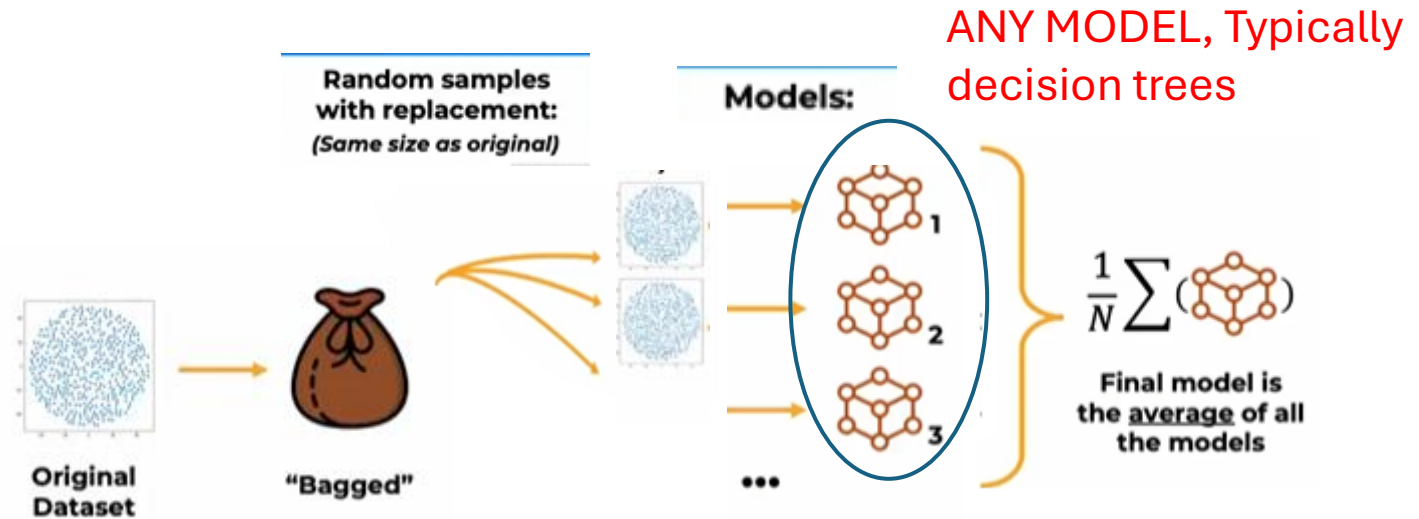
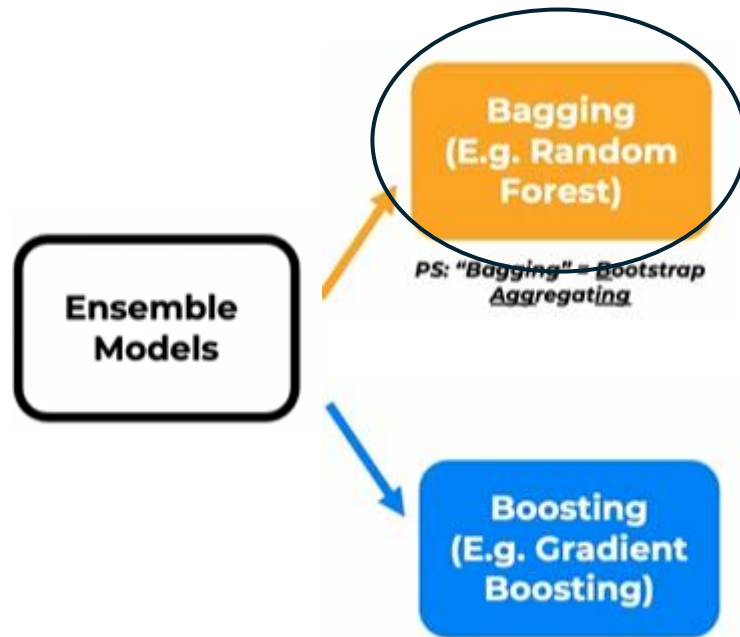


Good Fit/Robust



Overfitted

Ensemble Models



Ensemble Models

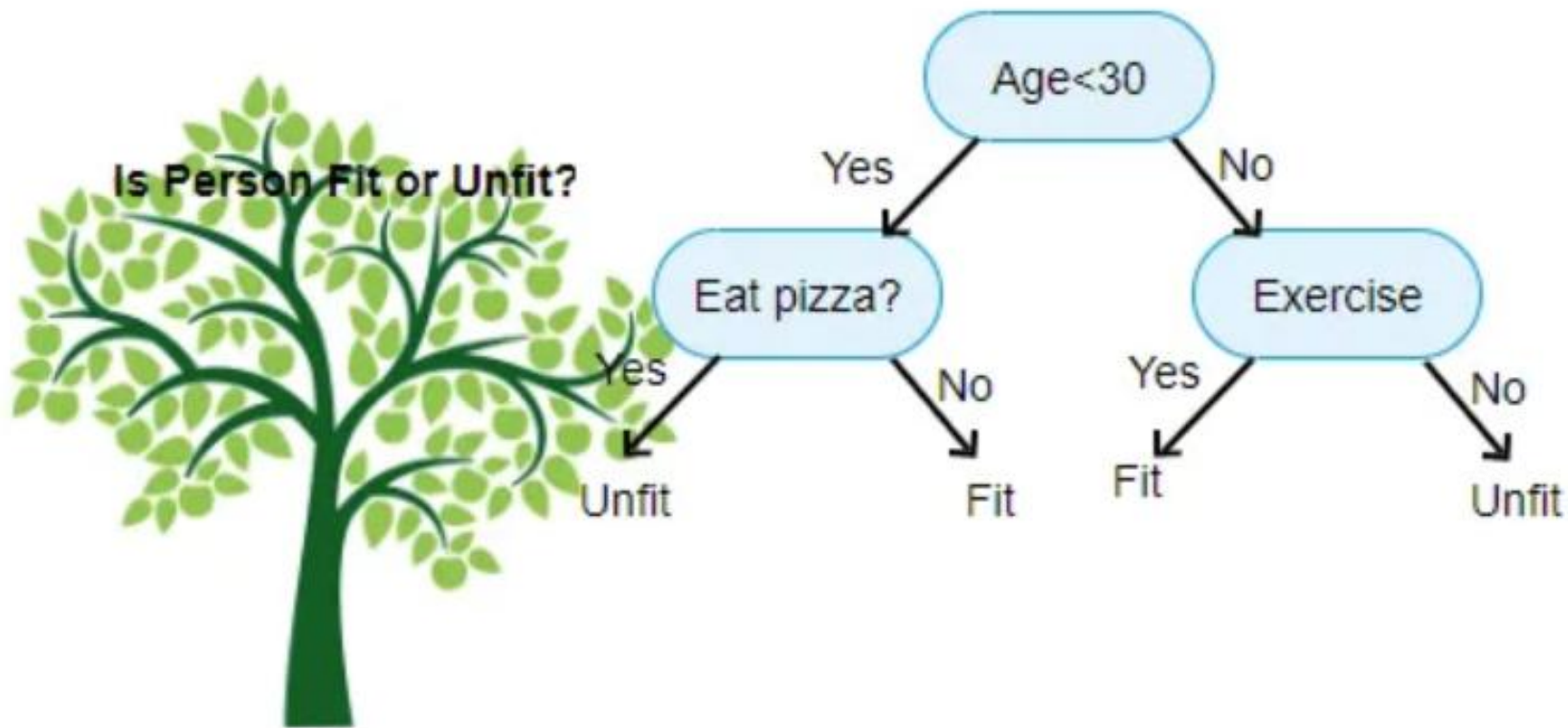
Bagging (Bootstrap Aggregating) ♦

דוגמה: Random Forest

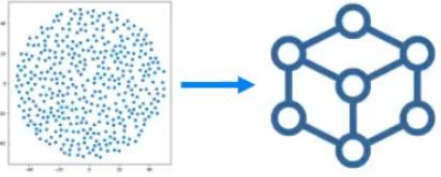
מה קורה בפועל?

- יוצרים כמה קבוצות נתונים על ידי דגימה אקראית עם החזרה (Random samples with replacement) מתוך הדאטה המקורי.
 - מאמנים כמה מודלים שונים במקביל על כל אחת מהדגימות.
 - המודל הסופי הוא ממוצע (לרגרסיה) או הצבעת רוב (לסיווג) של כל המודלים.
- עוזר למנוע overfitting.
 - מודלים פועלים במקביל (ניתן להריץ אותם בצורה מקבילית).

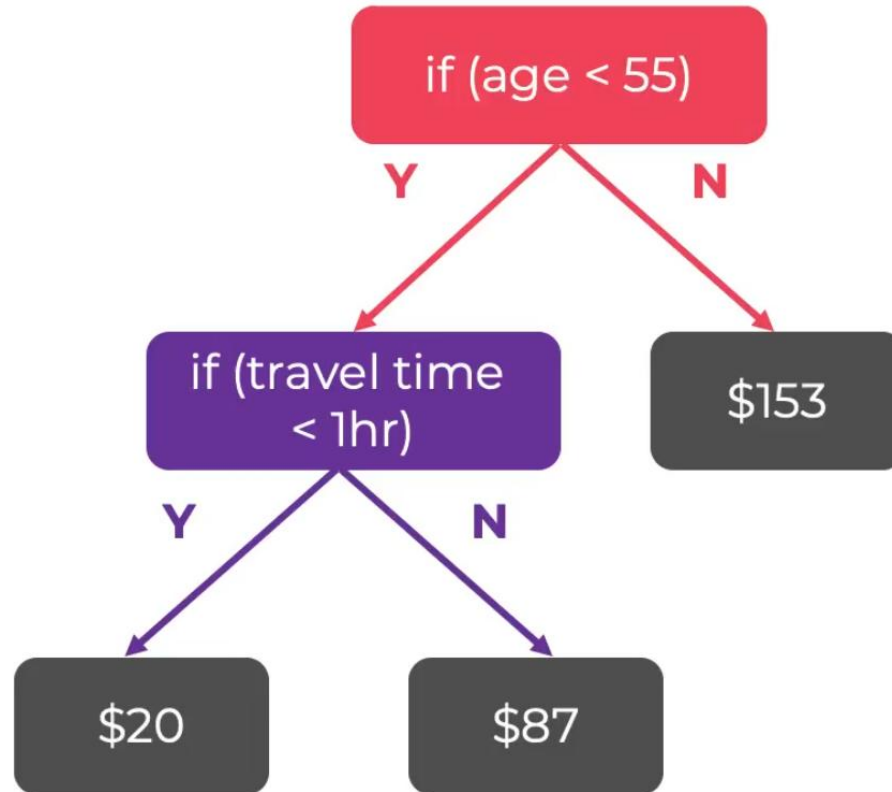
Ensemble Models (CART)



Decision Trees



Question: How much do customers spend at your store?



🌳 מבנה העץ:

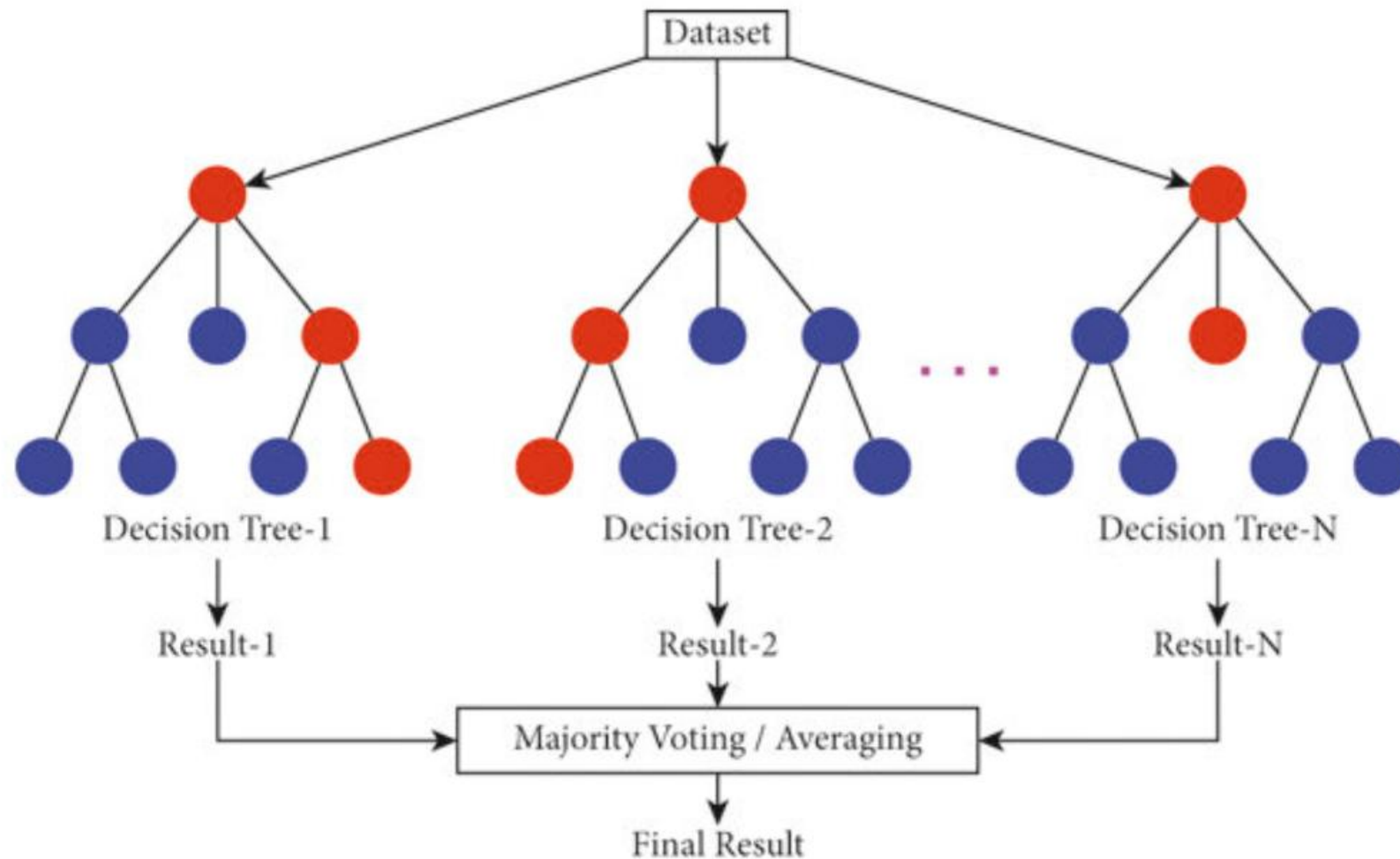
1. צומת ראשון (שורש):

- שואל: האם הגיל > 55 ?
- אם כן (Y) ממשיכים שמאלה.
- אם לא (N) → התחזית היא \$153.

2. צומת שני (אם הגיל > 55):

- שואל: האם זמן הנסיעה $>$ שעה?
- אם כן (Y) → התחזית היא \$20.
- אם לא (N) → התחזית היא \$87.

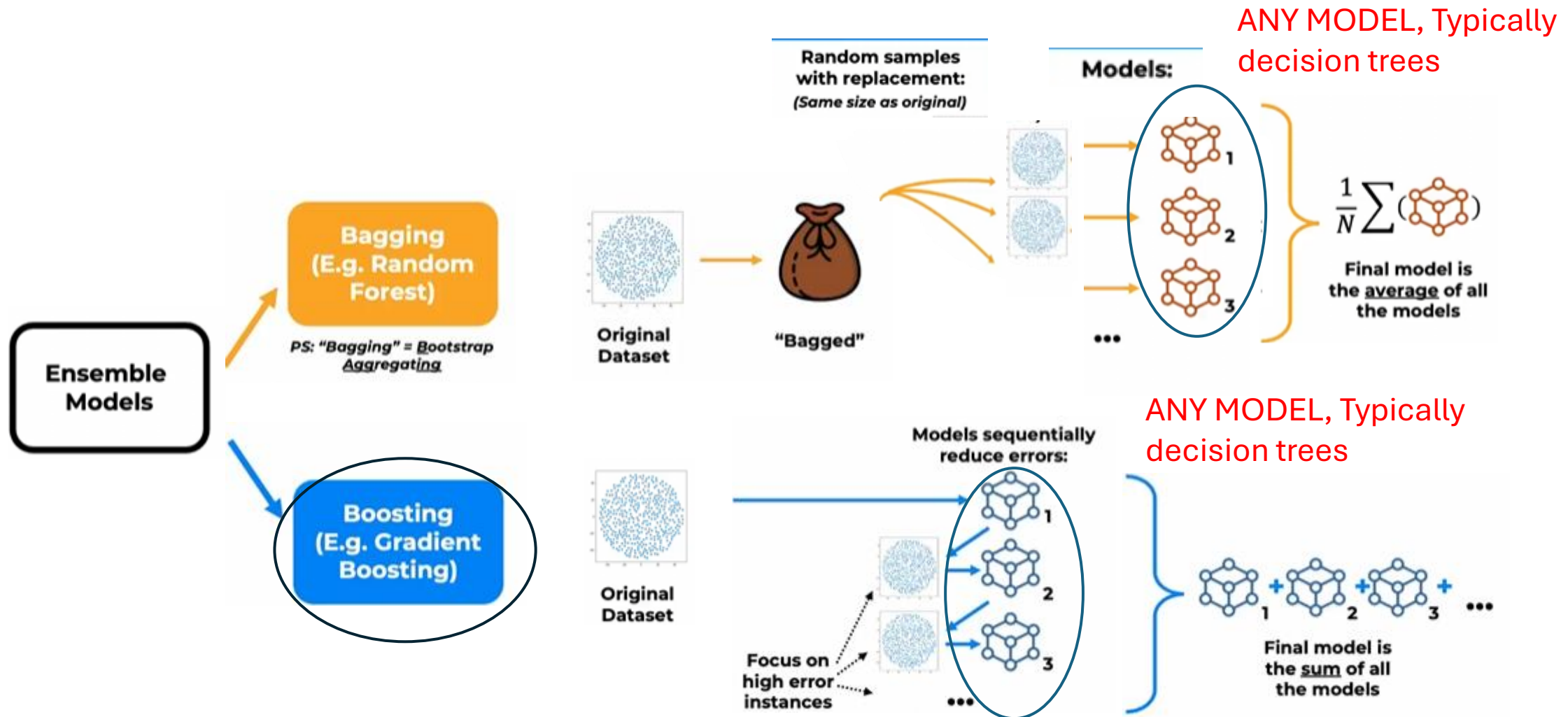
Ensemble Models (Random Forest)



Random Forest



Ensemble Models



Ensemble Models

Boosting

דוגמה: Gradient Boosting

מה קורה בפועל?

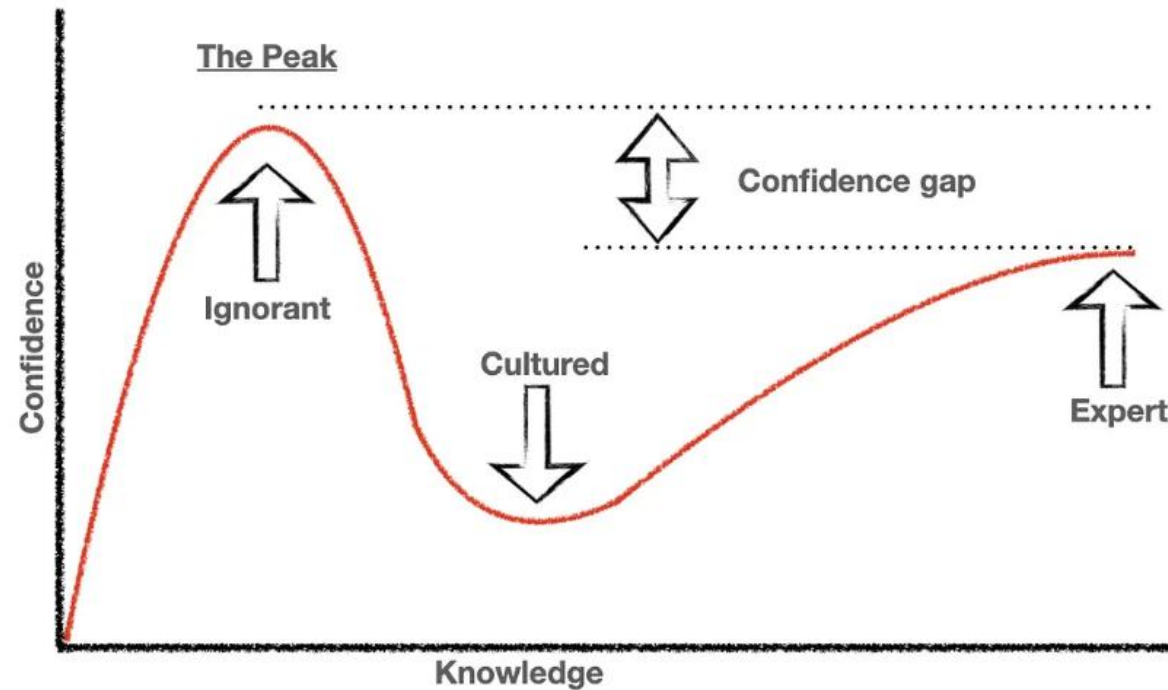
- מתחילים עם מודל פשוט ראשון.
- כל מודל חדש מתקן את השגיאות של המודל הקודם.
- הדאטה שדורש יותר תשומת לב (טעויות רבות) מקבל משקל גבוה יותר בשלבים הבאים.
- המודל הסופי הוא סכום משוקלל של כל המודלים.

יתרונות:

- מפחית bias.
- משיג ביצועים חזקים מאוד במקרים רבים.
- רגיש יותר ל־**overfitting**, ולכן נדרש טיפול נכון (למשל, עצים רדודים או רגולריזציה).

Gradient Boosting

Dunning-Kruger effect

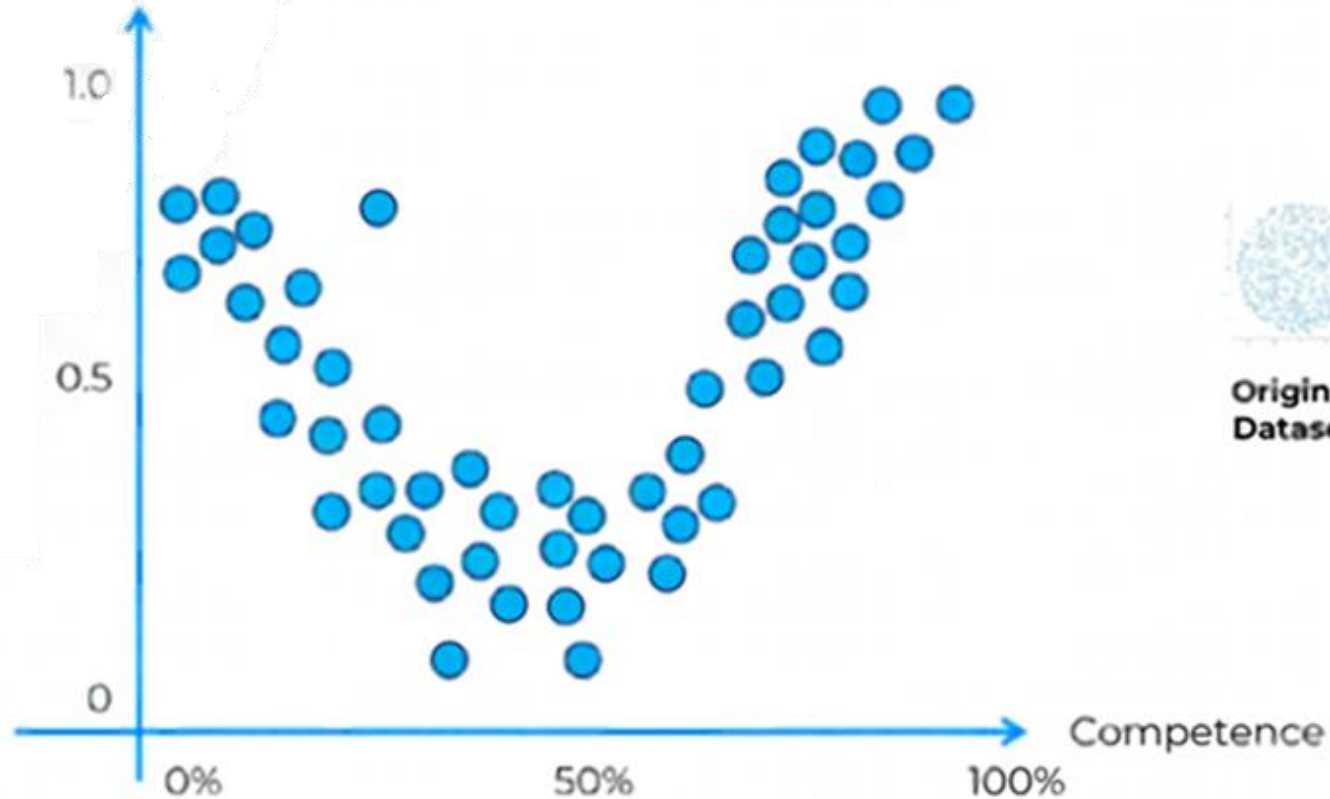


אפקט דאנינג-קרוגר: אנשים עם יכולת מוגבלת בתחום מסוים נוטים להעריך את עצמם מעבר למציאות.

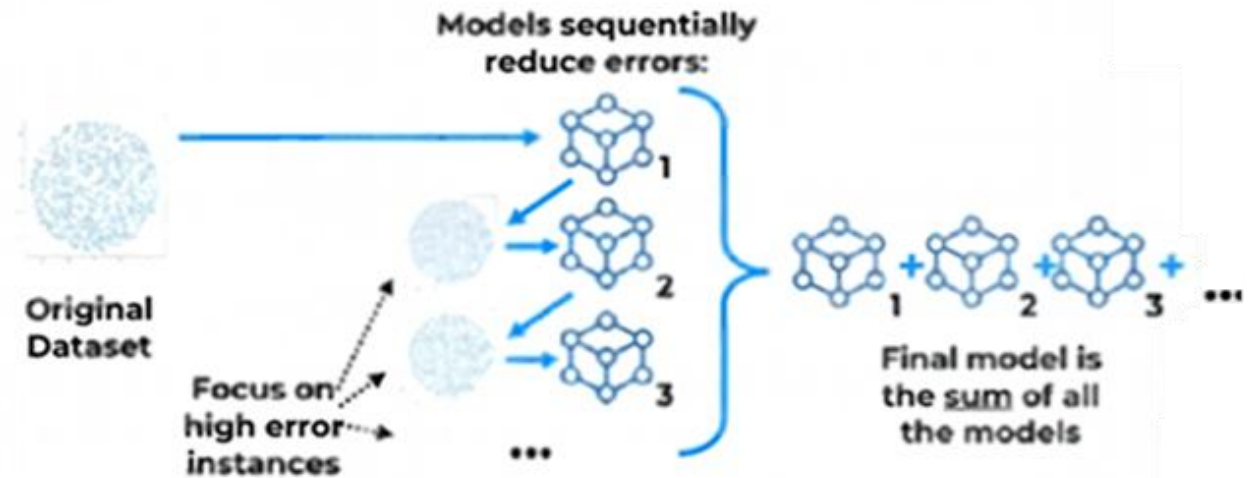
Gradient Boosting

Sample mock dataset*:

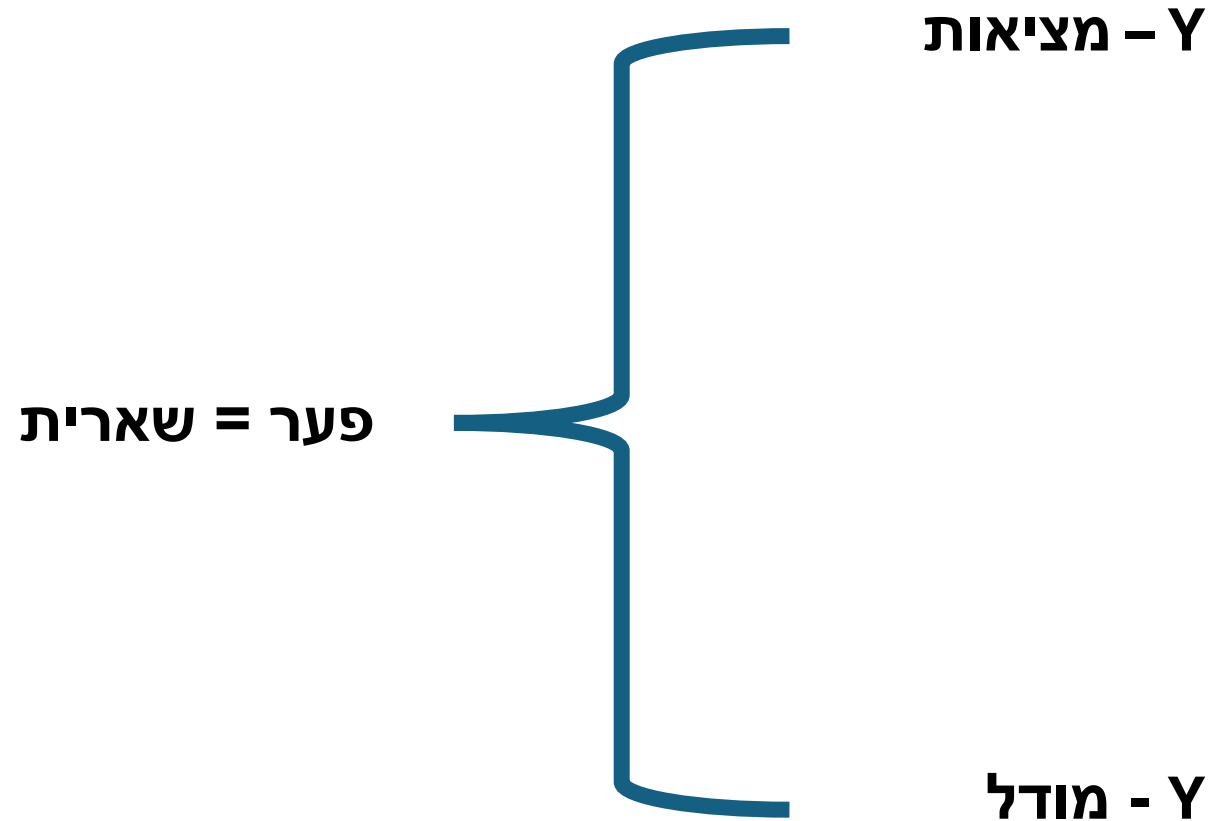
Confidence



**Based on the Dunning-Kruger effect: people with limited competence in a particular domain overestimate their abilities.*



Gradient Boosting

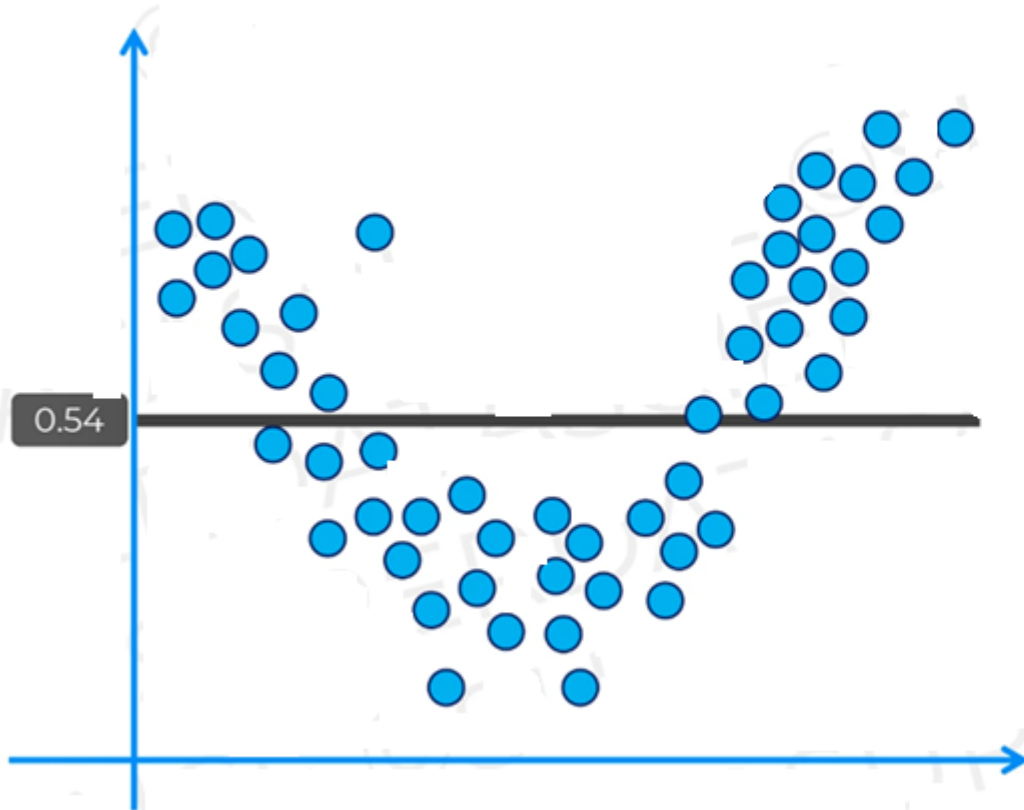


הרעיון – נבנה מודל על השאריות וכל פעם נשפר את המודל על ידי הוספת השאריות החזויות
(נעשה זאת בצורה איטרטיבית)

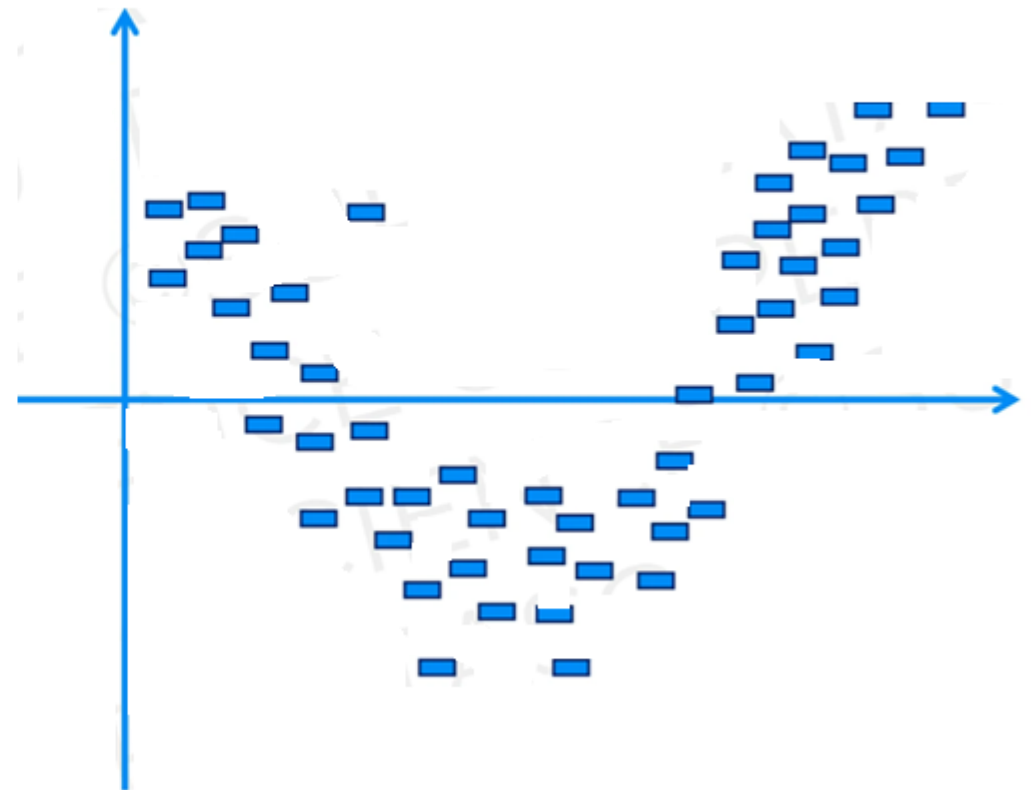


Gradient Boosting - xgboost

Original Data:



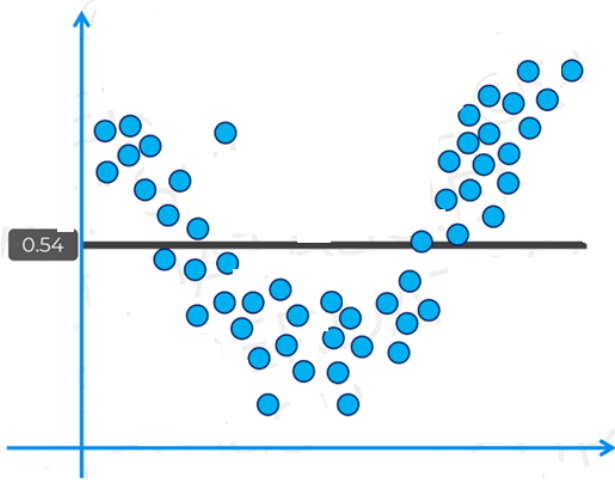
Residuals (Level 1):



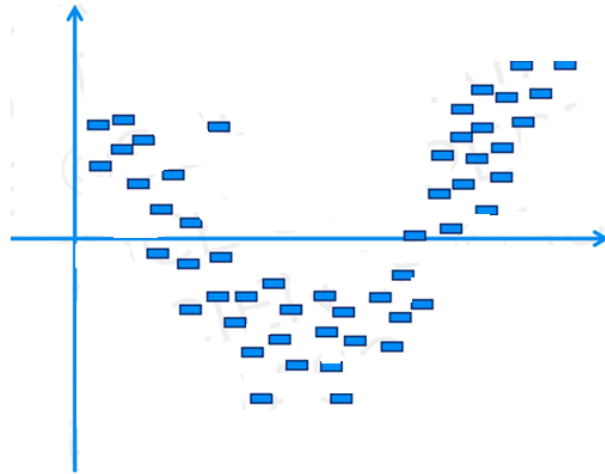


Gradient Boosting - xgboost

Original Data:



Residuals (Level 1):



■ Step 1: Start with the Original Data

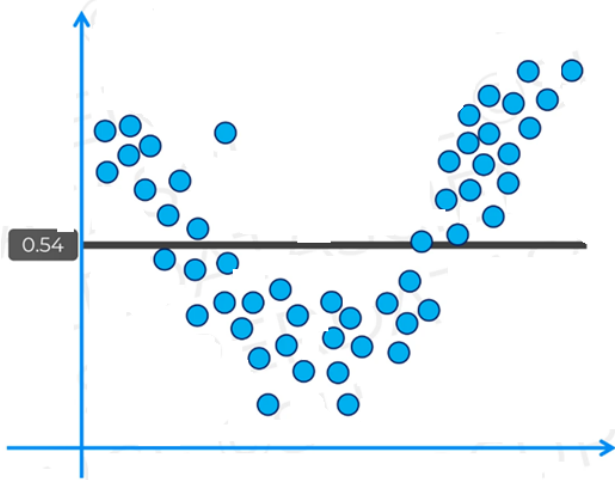
You have a dataset with features (x-axis) and target values (y-axis).

In the image, the data forms a **non-linear U-shape** pattern.

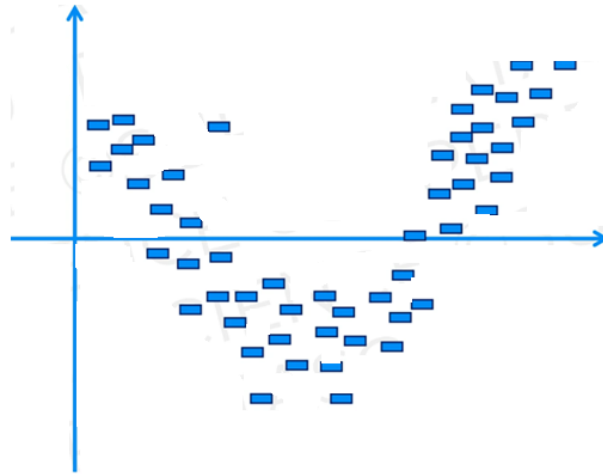


Gradient Boosting - xgboost

Original Data:



Residuals (Level 1):



■ Step 2: Fit the First Model (Model 1)

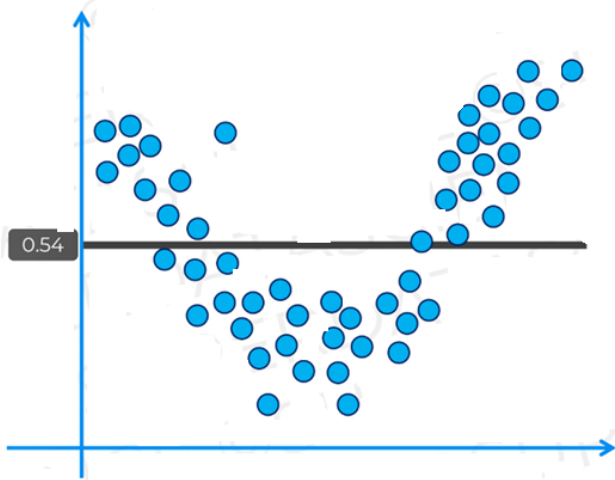
You train a **very simple model**, like a small decision tree or even a constant predictor.

- In the image, Model 1 predicts a **constant value** for all inputs:
→ Prediction = **0.54** (the average of the target values).
- This model clearly doesn't capture the U-shape at all.

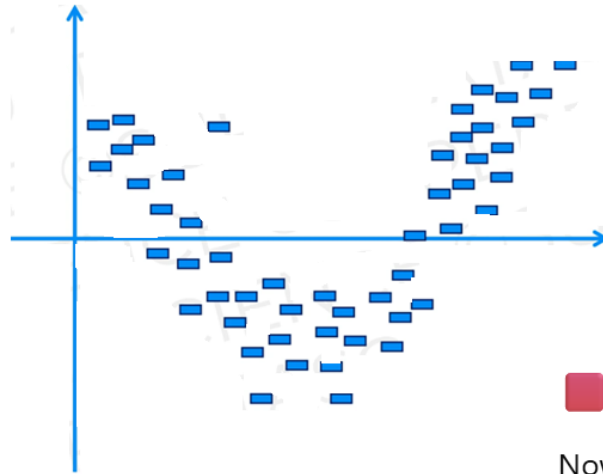


Gradient Boosting - xgboost

Original Data:



Residuals (Level 1):



Step 3: Compute the Residuals (Level 1)

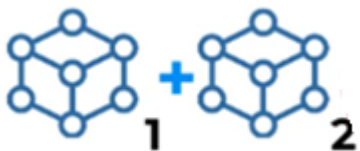
Now calculate the residuals (i.e., the errors):

$$\text{Residual} = y_{\text{true}} - \hat{y}_{\text{Model 1}} = y - 0.54$$

In the plot:

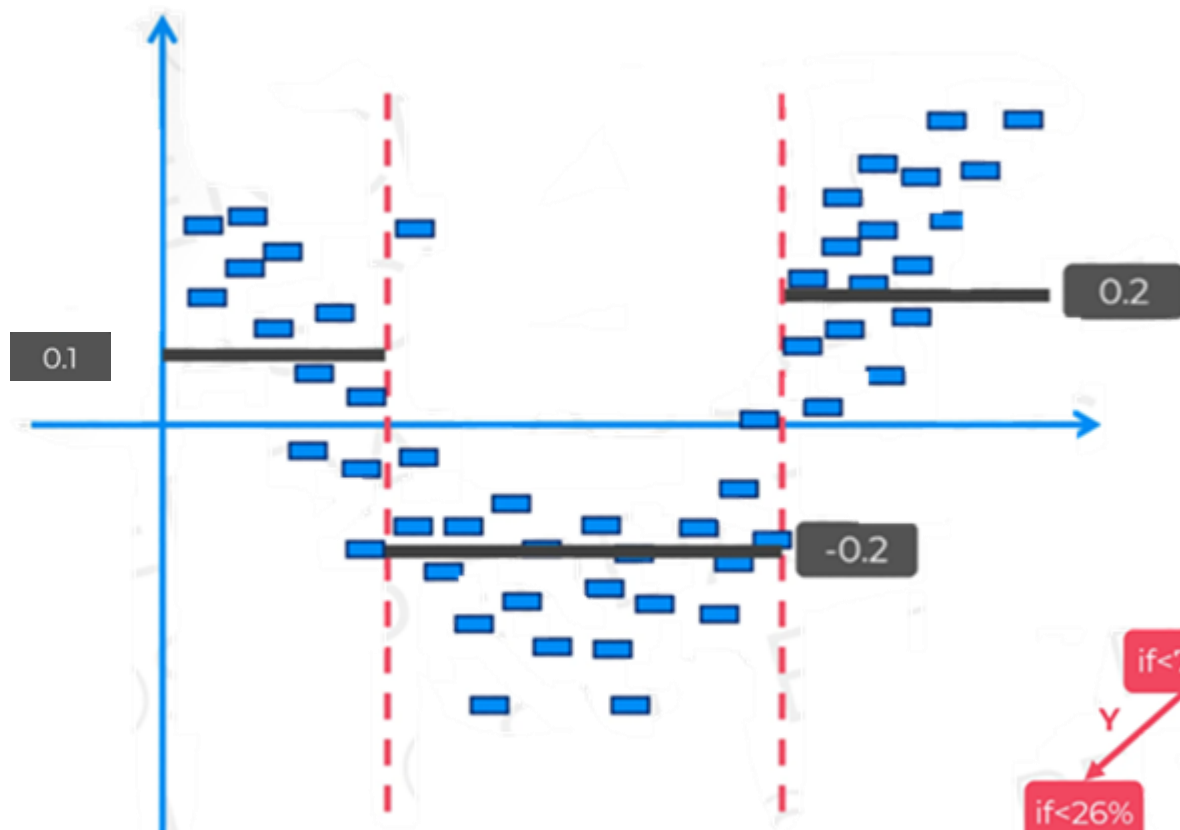
- On the **left side**, residuals are **positive** → model under-predicted.
- In the **middle**, residuals are **negative** → model over-predicted.
- On the **right**, residuals are **positive** again → under-predicted again.

These residuals are shown as **small blue rectangles** in the right graph.

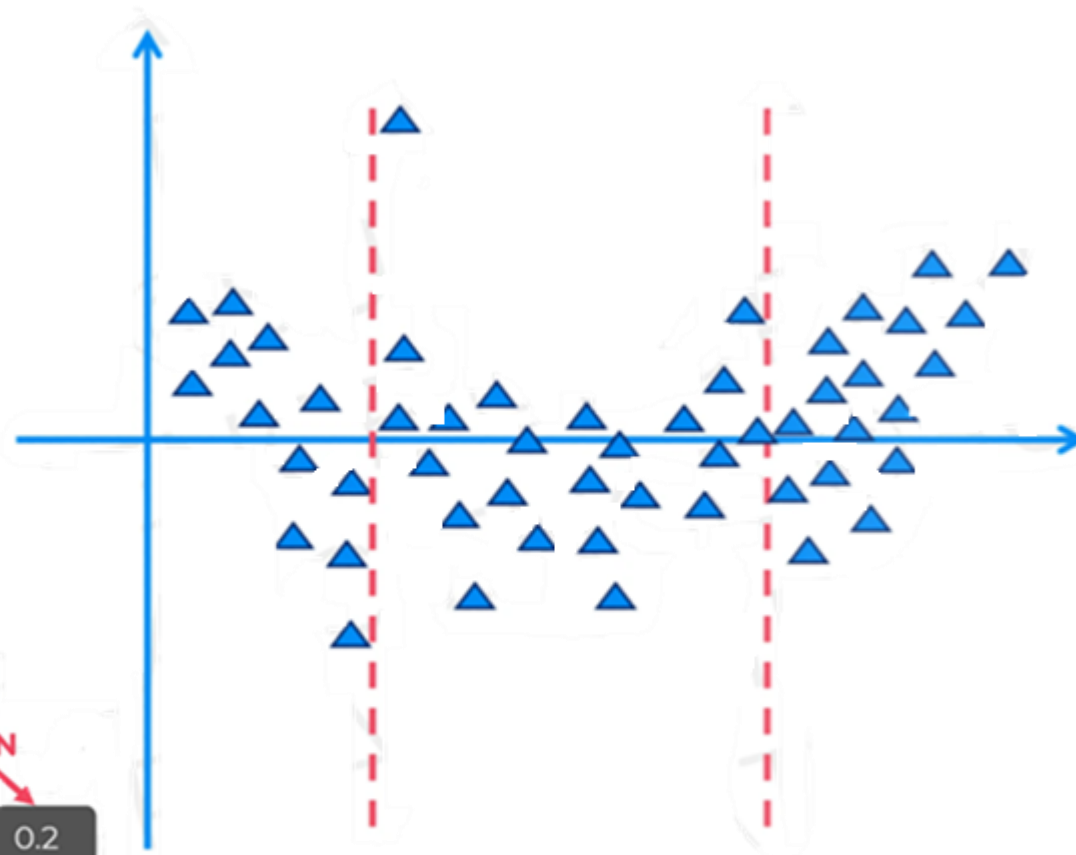


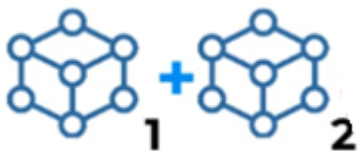
Gradient Boosting - xgboost

Residuals (Level 1):

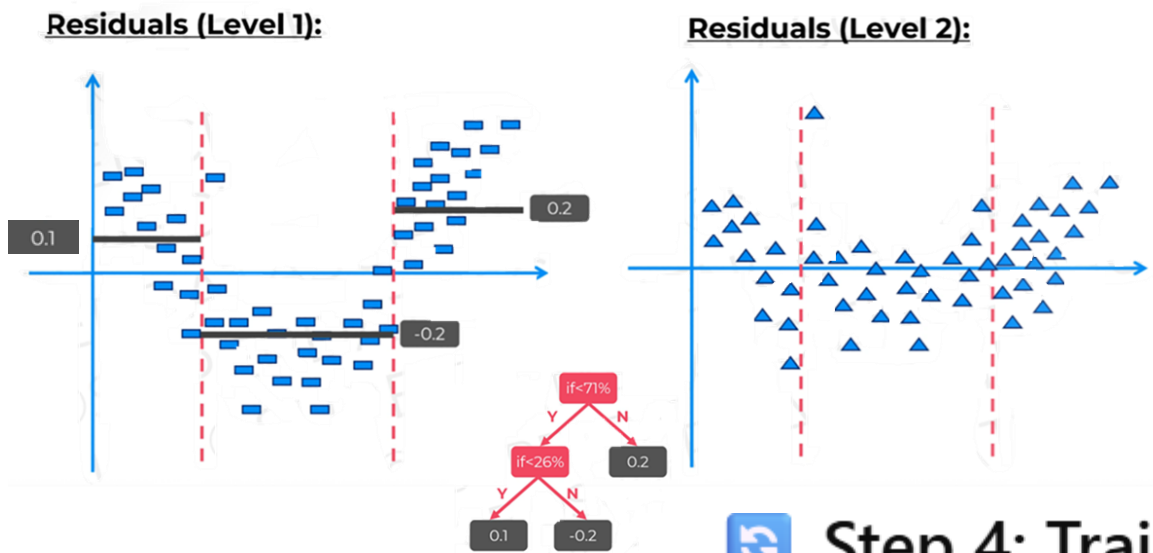


Residuals (Level 2):





Gradient Boosting - xgboost



Step 4: Train the Second Model on the Residuals

You now train a **new model** (Model 2) to predict the residuals.

This model's job is:

👉 *"How much did the previous model miss, and where?"*

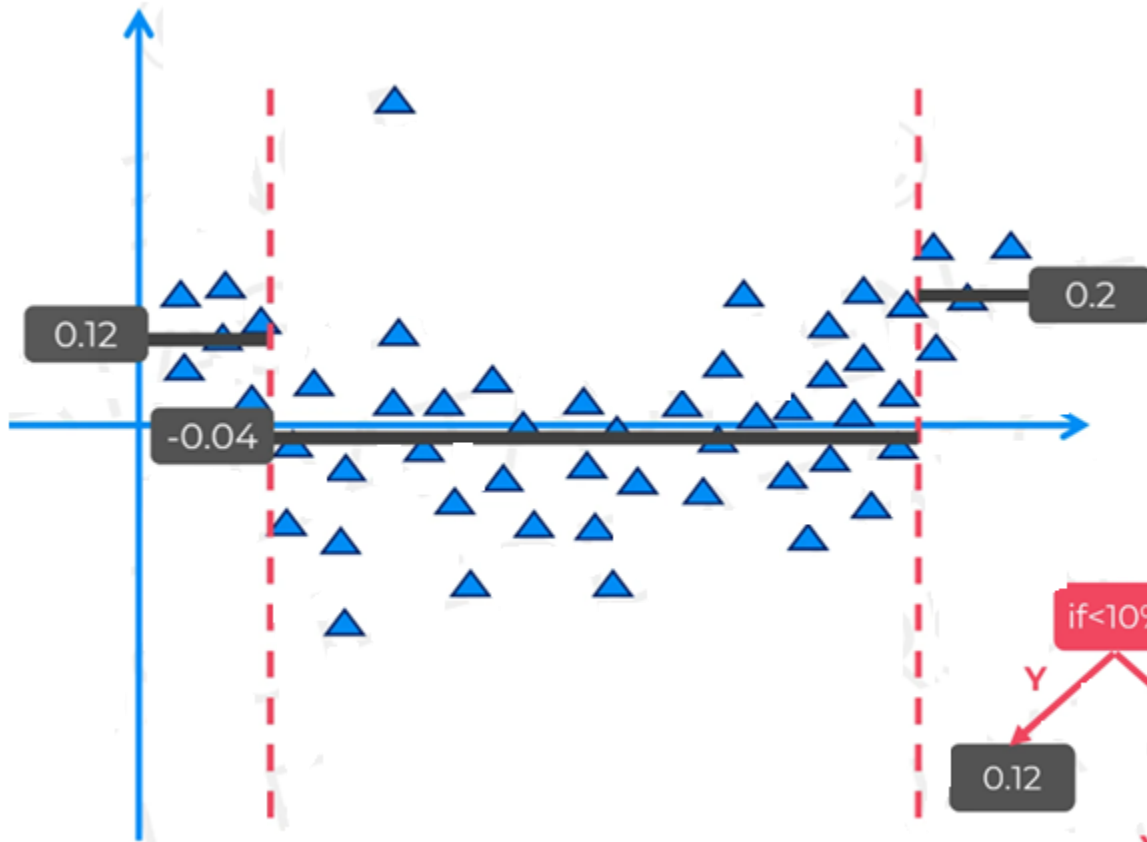
Because the residuals follow a pattern (like an upside-down U),

→ Model 2 can learn to fit that shape and make corrections.

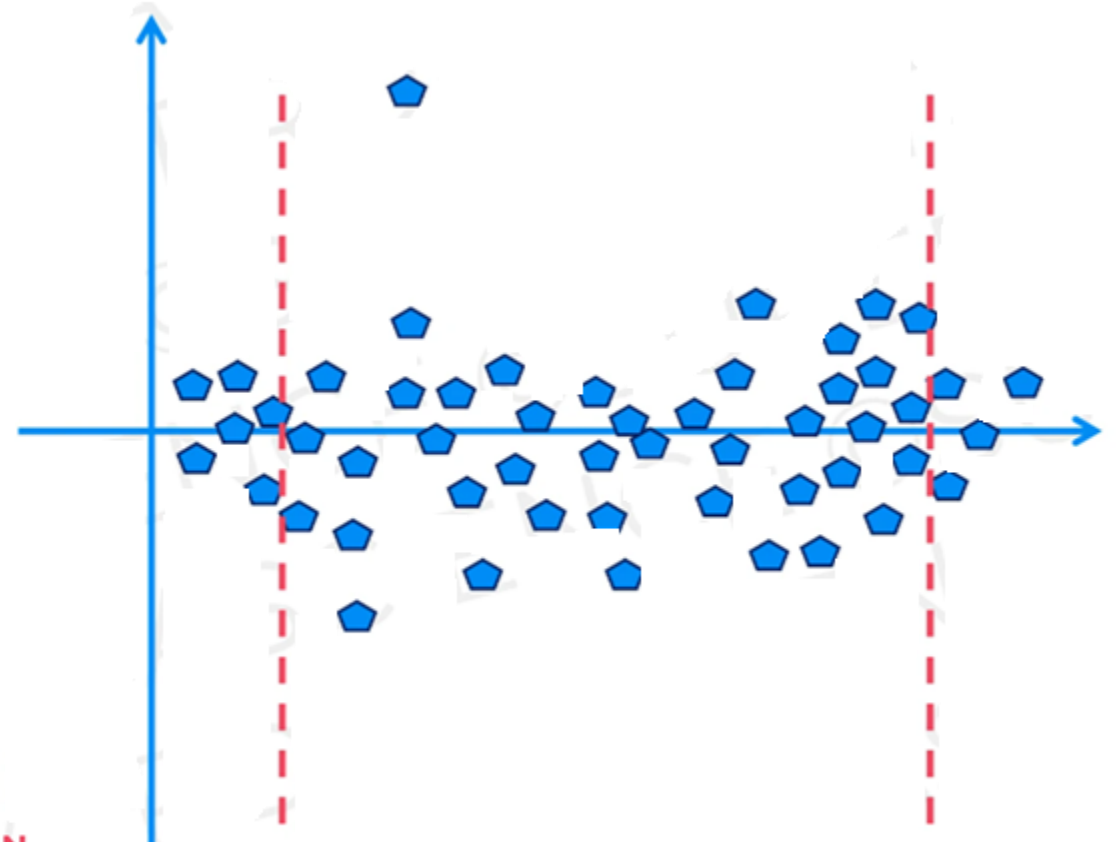


Gradient Boosting - xgboost

Residuals (Level 2):



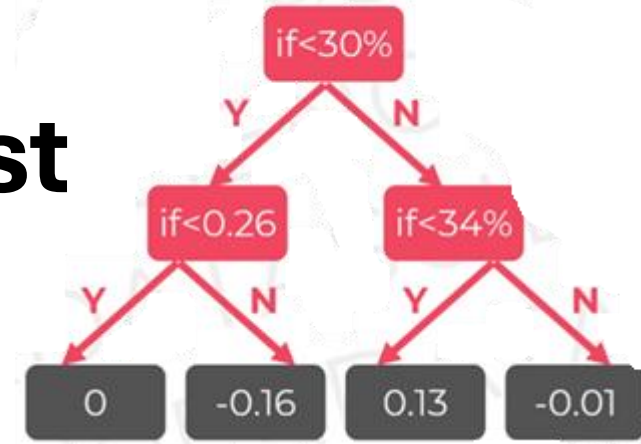
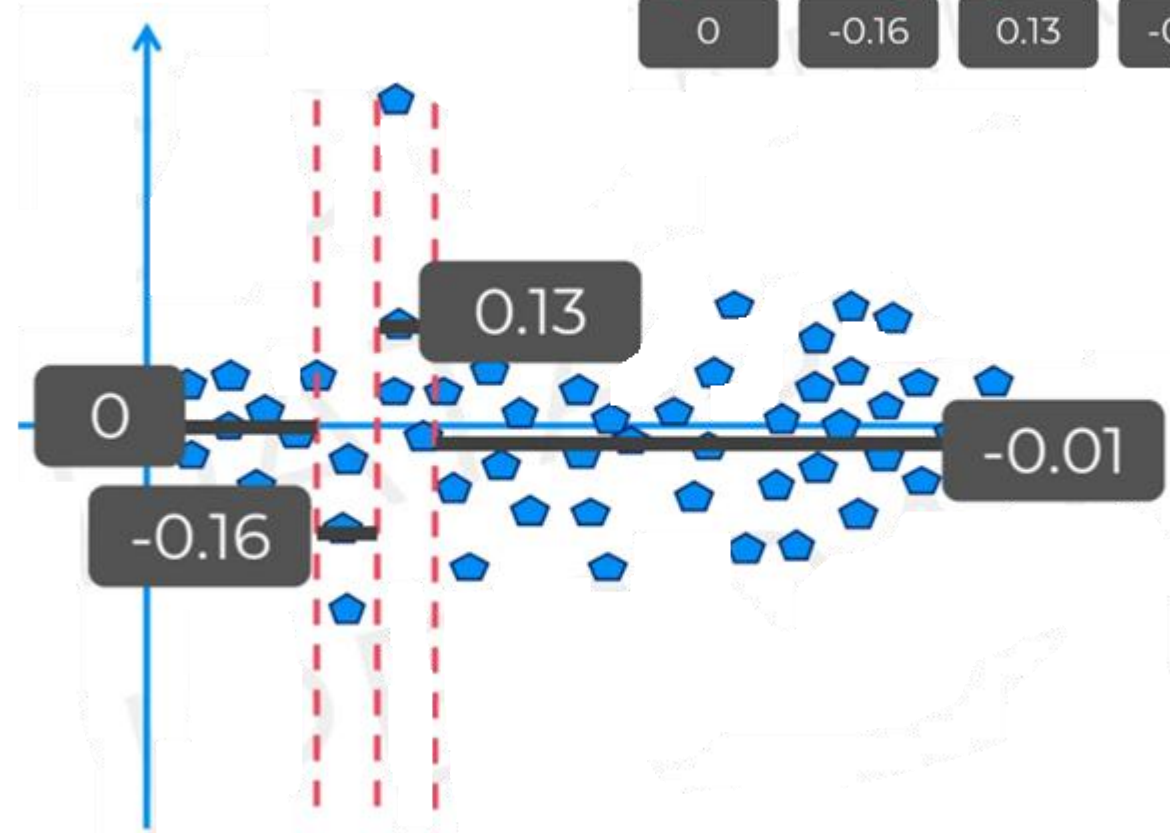
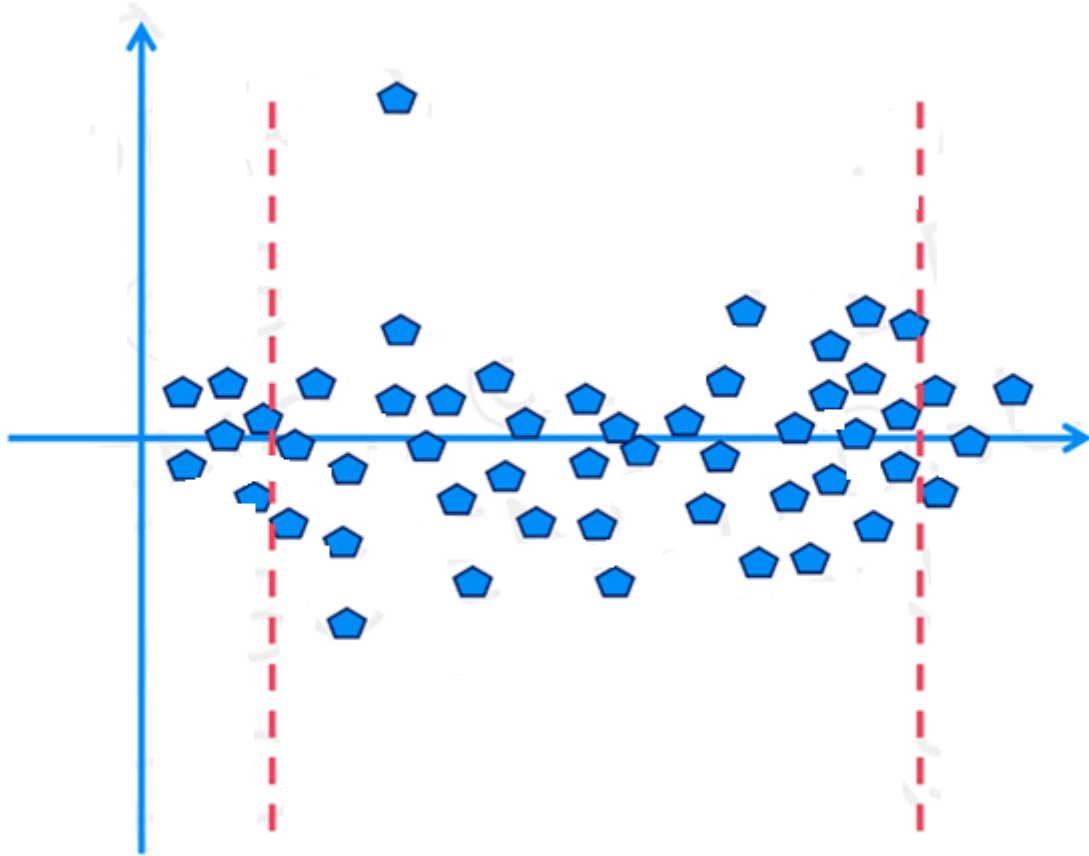
Residuals (Level 3):





Gradient Boosting - xgboost

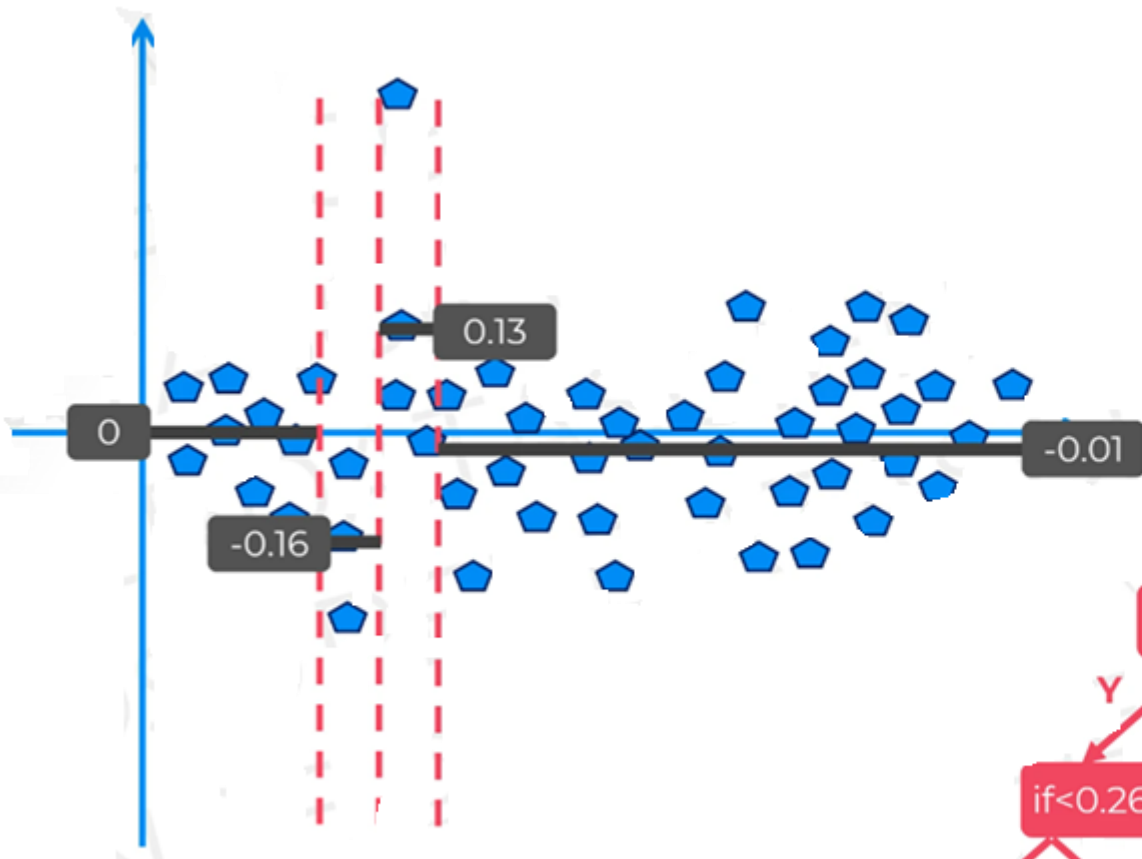
Residuals (Level 3):



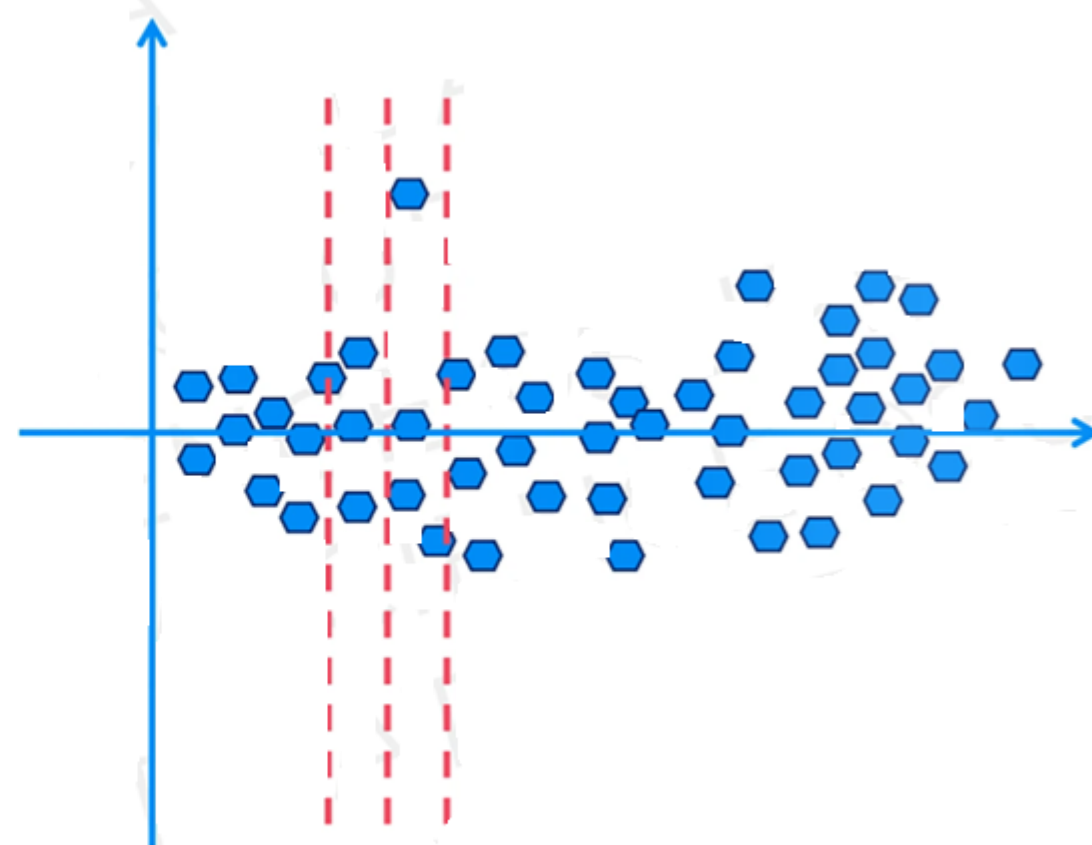


Gradient Boosting - xgboost

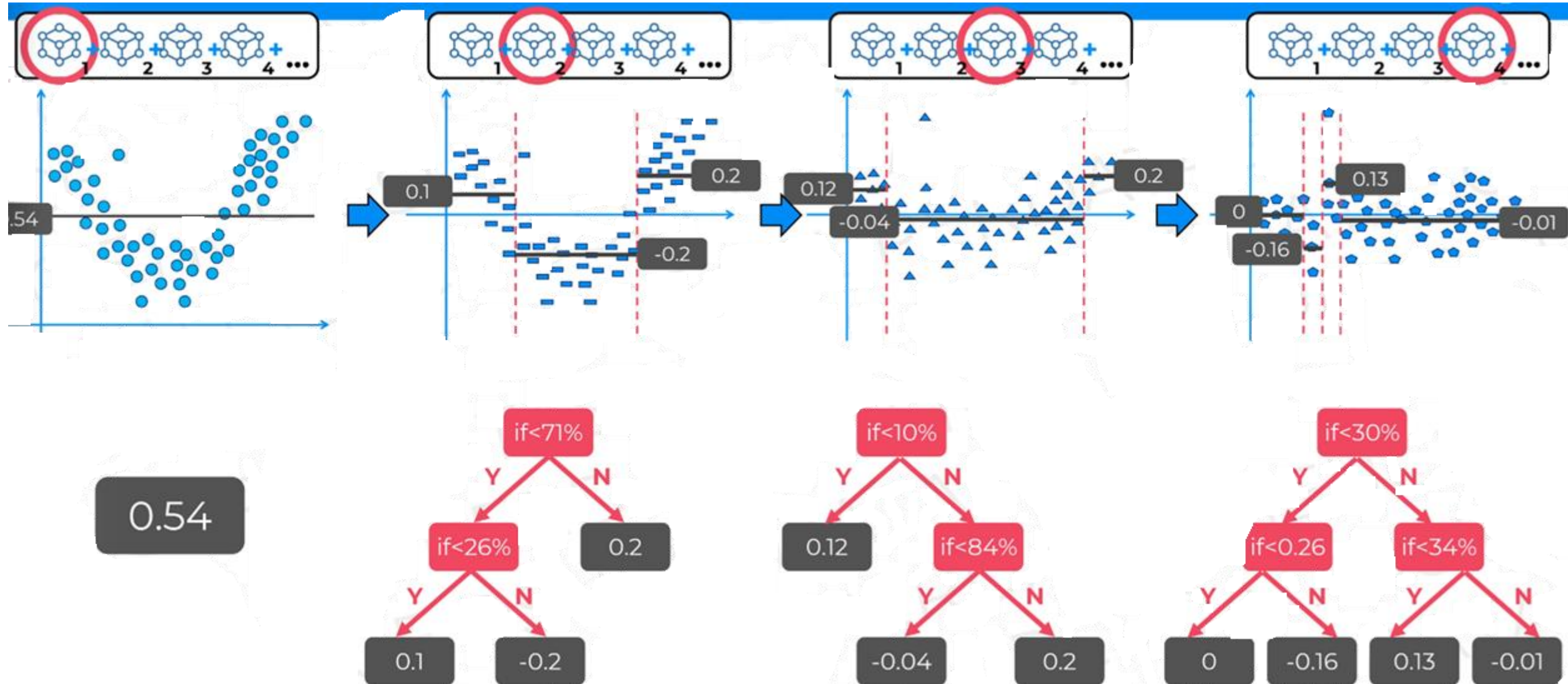
Residuals (Level 3):



Residuals (Level 4):

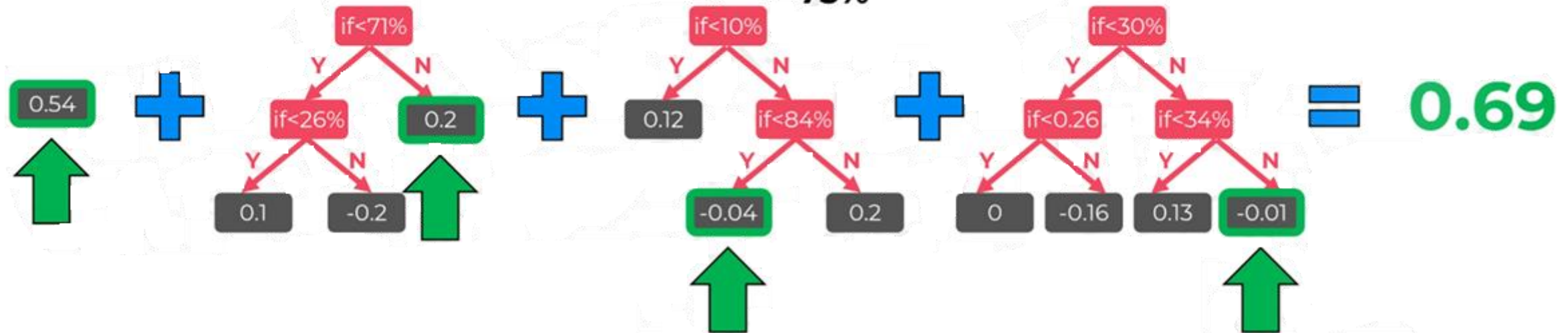
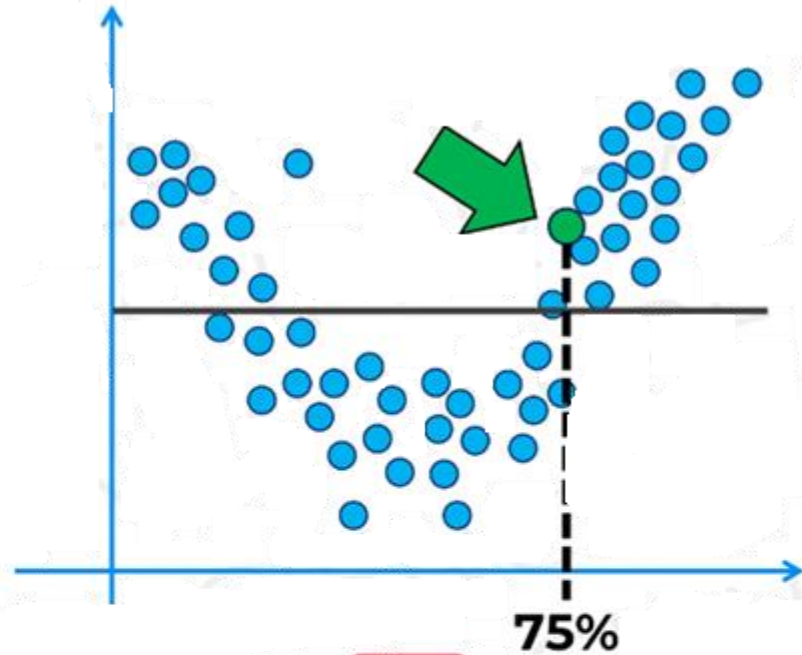


Gradient Boosting - xgboost



Gradient Boosting - xgboost

Sample calculation:



Gradient Boosting

During Training:

Row Sampling
(Subsampling)

ID	x_1	x_2	x_3	x_4	...	x_n	y
1
2
3
4
5
6
7
8
...
998
999
1,000

Gradient Boosting

During Training:

ID	x_1	x_2	x_3	x_4	...	x_n	y
1
2
3
4
5
6
7
8
...
998
999
1,000

Row Sampling
(Subsampling)

- בכל שלב של Gradient Boosting (למשל, כל פעם שאנחנו בונים עץ חדש), האלגוריתם לא משתמש בכל הדאטה, אלא רק בחלק ממנו — לדוגמה, רק 70% מהשורות נבחרות באקראי.

- זה דומה לבאגינג (bagging), אבל שונה:
- ב־Bagging עושים דגימה עם החזרה.
- ב־Gradient Boosting זו דגימה בלי החזרה.

למה עושים את זה? 📌

הסבר

מטרה

עוזר למנוע מהמודל "לזכור" את כל הדאטה בצורה מדויקת מדי.

להפחית Overfitting 🎯

עץ קטן שמתאמן רק על חלק מהנתונים רץ מהר יותר.

לשפר ביצועים ⚡

כל עץ לומד קצת זווית אחרת של השאריות.

לשפר גיוון בין המודלים 🧪



Gradient Boosting - xgboost

Why Use Row & Column Sampling in XGBoost?

✓ 1. Prevents Overfitting

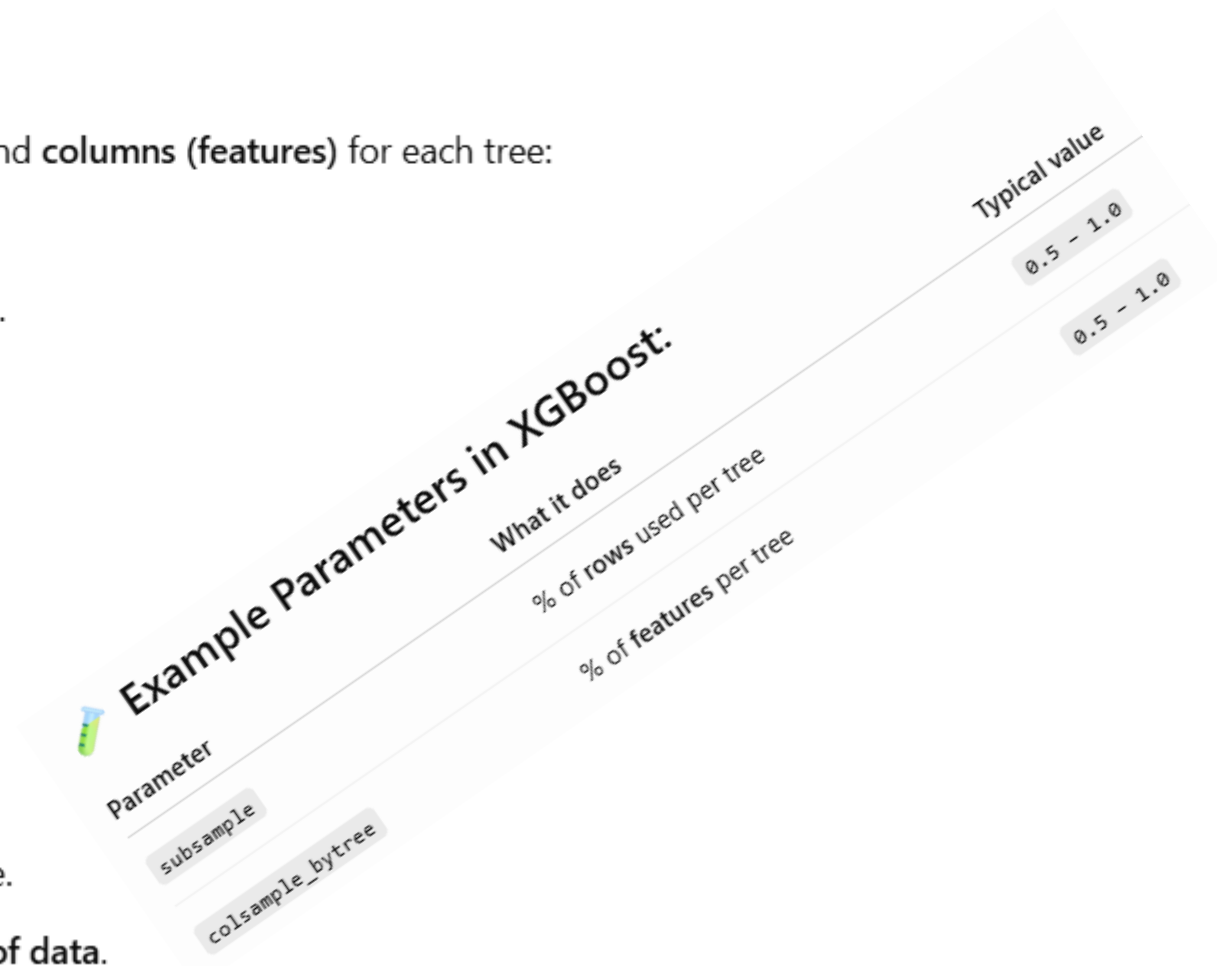
- By randomly selecting **only some rows (observations)** and **columns (features)** for each tree:
 - The model doesn't memorize the data.
 - Forces diversity across trees → better generalization.

✓ 2. Speeds Up Training

- Smaller data subset → each tree trains faster.
- Especially useful with **large datasets** or many features.

✓ 3. Creates Tree Diversity

- Similar to Random Forest's idea:
 - Less correlation between trees → stronger ensemble.
- Trees make decisions based on **different combinations of data**.



Parameter	What it does	Typical value
<code>subsample</code>	% of rows used per tree	0.5 - 1.0
<code>colsample_bytree</code>	% of features per tree	0.5 - 1.0

Gradient Boosting - xgboost

האם צריך לנרמל או לקודד?

1. נרמול / Scaling — לא צריך 🛠️

- XGBoost לא רגיש לסקאלה של המשתנים, כי הוא מבוסס על עצים (Trees), ולא על חישובי מרחק.
- כלומר: לא משנה אם ערך של משתנה הוא בין 0-1 או בין 0-10,000 — העץ רק בודק תנאים כמו `feature < threshold`.

✅ לכן: אין צורך לעשות MinMax או StandardScaler לפני שימוש ב-XGBoost.



Gradient Boosting - xgboost

האם צריך לנרמל או לקודד?

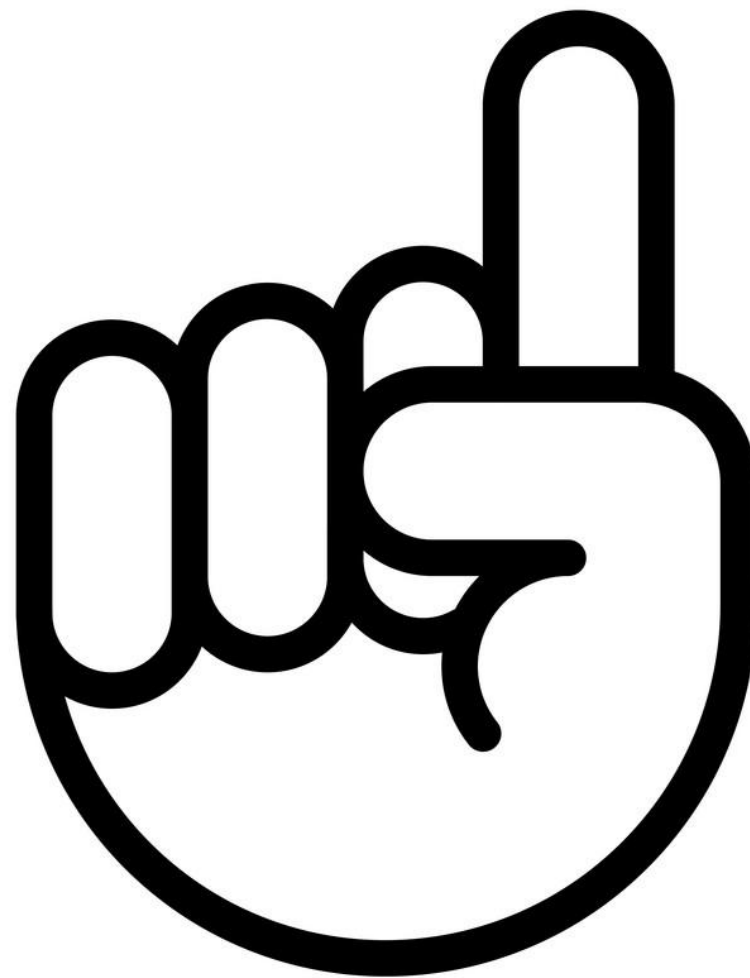
2. קידוד משתנים קטגוריאליים — כן צריך

- XGBoost לא תומך בתכונות קטגוריאליות באופן ישיר.
- לכן, חייבים לקודד את העמודות הקטגוריאליות למספרים לפני ההזנה למודל.

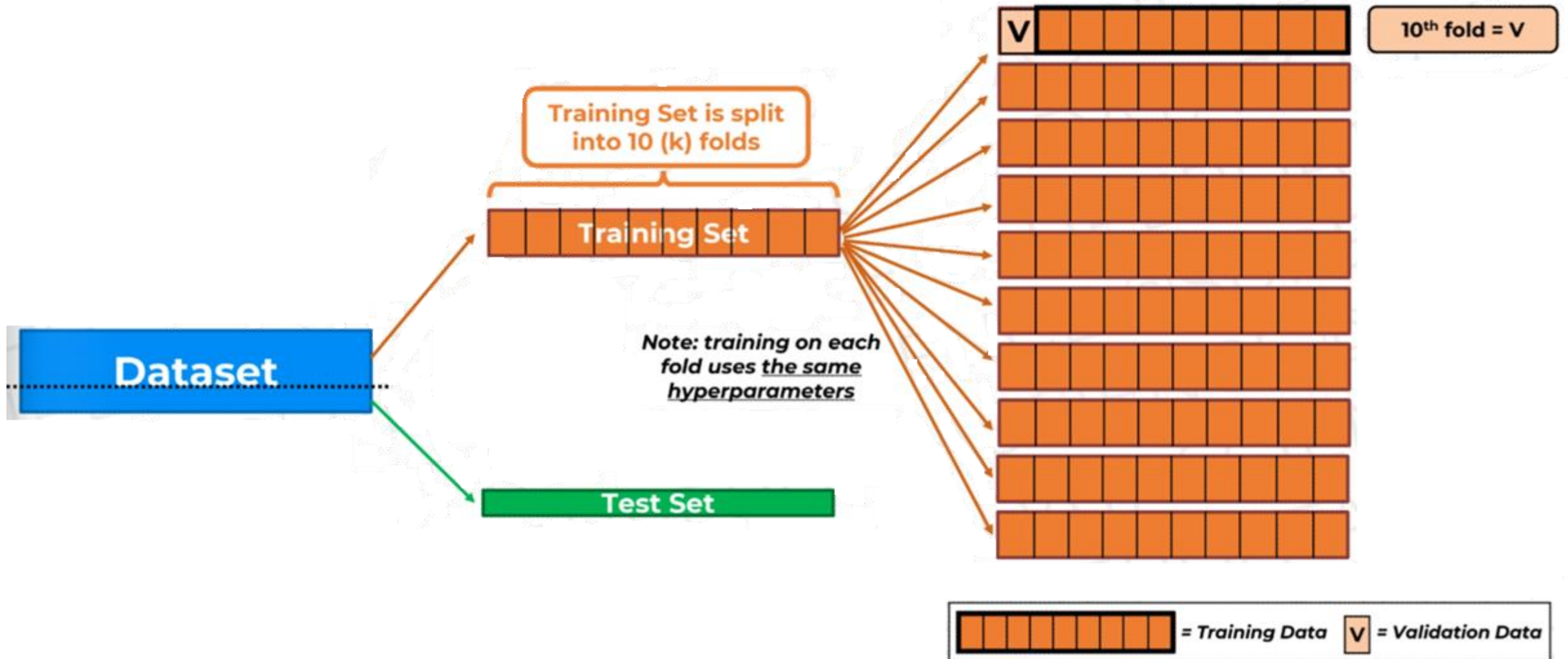
איך לקודד?

-  One-Hot Encoding – מתאים אם יש מעט קטגוריות:
- לדוגמה: `color = ['red', 'blue']` יהפוך ל-2 עמודות: `color_red`, `color_blue`.
-  Label Encoding – מתאים לקטגוריות רבות, אבל בזהירות:
- XGBoost עלול לפרש את המספרים כמשמעותיים (כלומר, $2 < 1$), למרות שאין בכך היגיון.

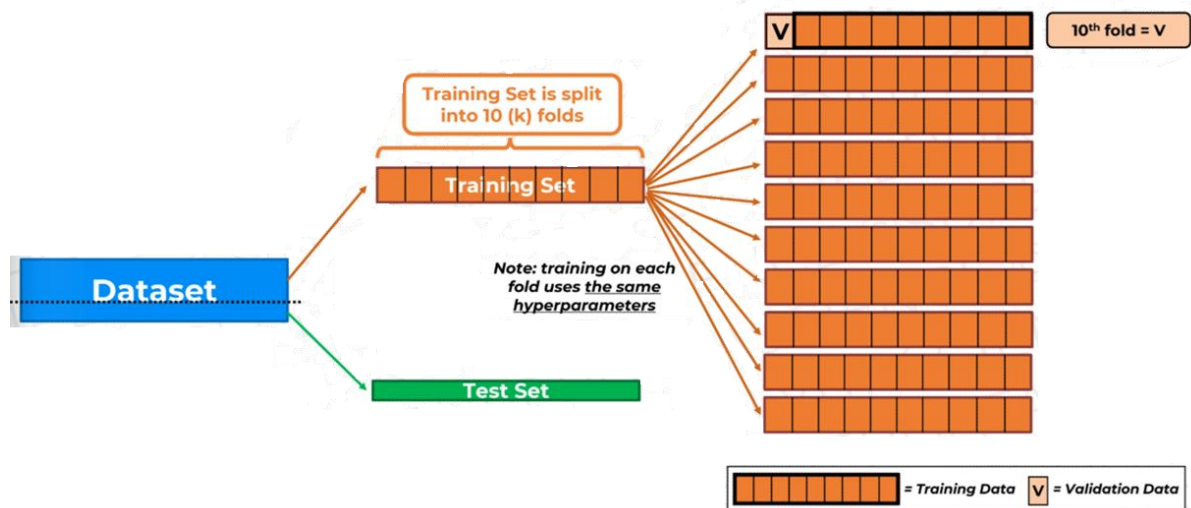
Xgboost Regressor



K-Fold Cross Validation



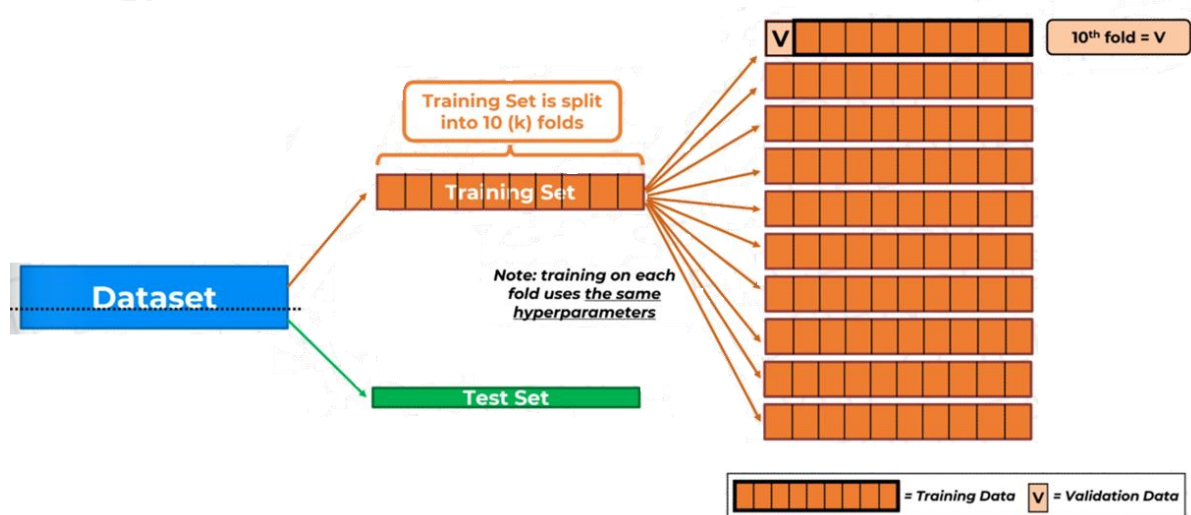
K-Fold Cross Validation



Dataset 1

- הנתונים מחולקים לשני חלקים:
- Training Set – עבור אימון והצלבת תוקף (Cross Validation)
- Test Set – נשמר בצד, ומשמש רק לבדיקה הסופית לאחר בחירת המודל.

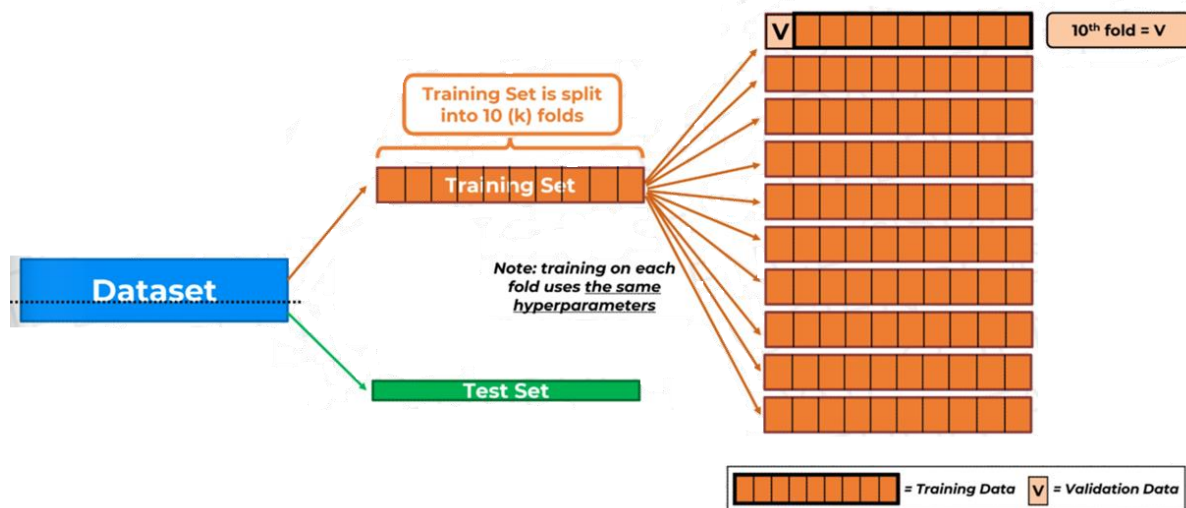
K-Fold Cross Validation



Training Set Split into K Folds .2

- קבוצת האימון (Training Set) מחולקת ל-K תתי-קבוצות שוות בגודלן, בתמונה: (10-Fold) $K = 10$.
- כל אחת מהתתי-קבוצות תיקרא fold.

K-Fold Cross Validation

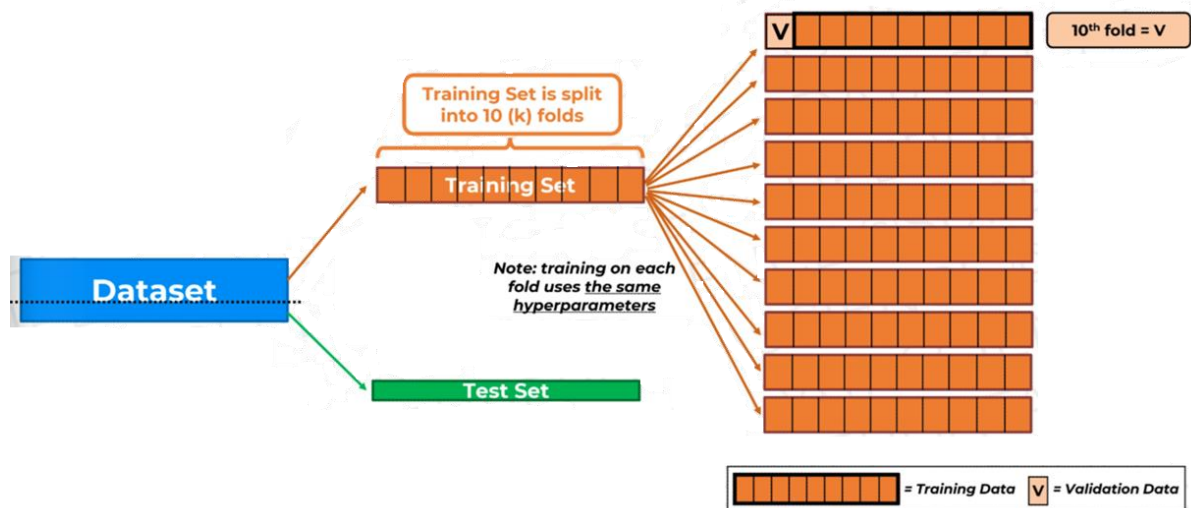


3. תהליך ההצלבה (Cross Validation Loops)

- נבצע K איטרציות של אימון:
- בכל איטרציה:
- נבחר Fold אחר להיות קבוצת האימות (Validation Set).
- כל שאר ה-folds $K-1$ משמשים כאימון (Training Set).
- התוצאה: נבנים K מודלים, וכל אחד מהם נבחן על חלק שונה מהנתונים.

לדוגמה, באיטרציה ה-3:
Fold 3 משמש כ-Validation, ו-10, 4, 2, 1 Folds משמשים לאימון.

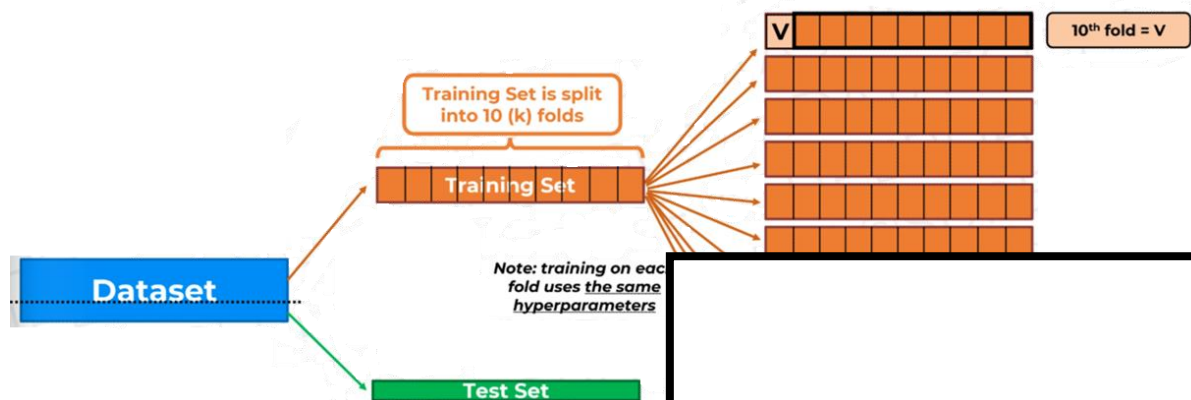
K-Fold Cross Validation



4. ✓ שקלול התוצאות

- בסוף, מחשבים ממוצע של ביצועי המודלים (לדוגמה: דיוק, AUC, RMSE וכו') על כל ה-K folds.
- זה נותן הערכה יציבה ואובייקטיבית לביצועי המודל על הדאטה.

K-Fold Cross Validation



Final Test Set 0.5

- לאחר בחירת מודל סופי והיפר-פרמטרים – מאמנים מודל על כל ה-Training Set, ובודקים אותו על ה-Test Set ששמרנו בצד בהתחלה.

למה זה חשוב? 🧠

הסבר

יתרון

כל נקודה משמשת גם לאימון וגם לאימות (אבל אף פעם לא בו זמנית).

שימוש מלא בדאטה 📊

לא מתבססים על חיתוך אקראי יחיד, אלא על ממוצע ביצועים.

הפחתת הטיית אימון 🧠

במיוחד בדאטה קטן או כאשר תוצאות משתנות בין ריצות.

הערכה אמינה יותר 🎯

Gradient Boosting - xgboost

Feature Importance

XGBoost Feature Importance 🧠

- XGBoost מאפשר לחשב חשיבות תכונות באופן טבעי כחלק מתהליך אימון המודל.
- החשיבות משקפת כמה תרומה הייתה לפיצ'ר בהחלטות הפיצול בעצים.
- שימושי ל:

- הסבר המודל (Explainability)

- צמצום משתנים (Feature Selection)

- שיפור ביצועים והבנת השפעות עסקיות

השוואת ביצועים 📊

לאחר בניית שני המודלים – עם כל הפיצ'רים ועם רק הפיצ'רים החשובים – נבדוק את ביצועי המודל לפי מדדים סטנדרטיים:

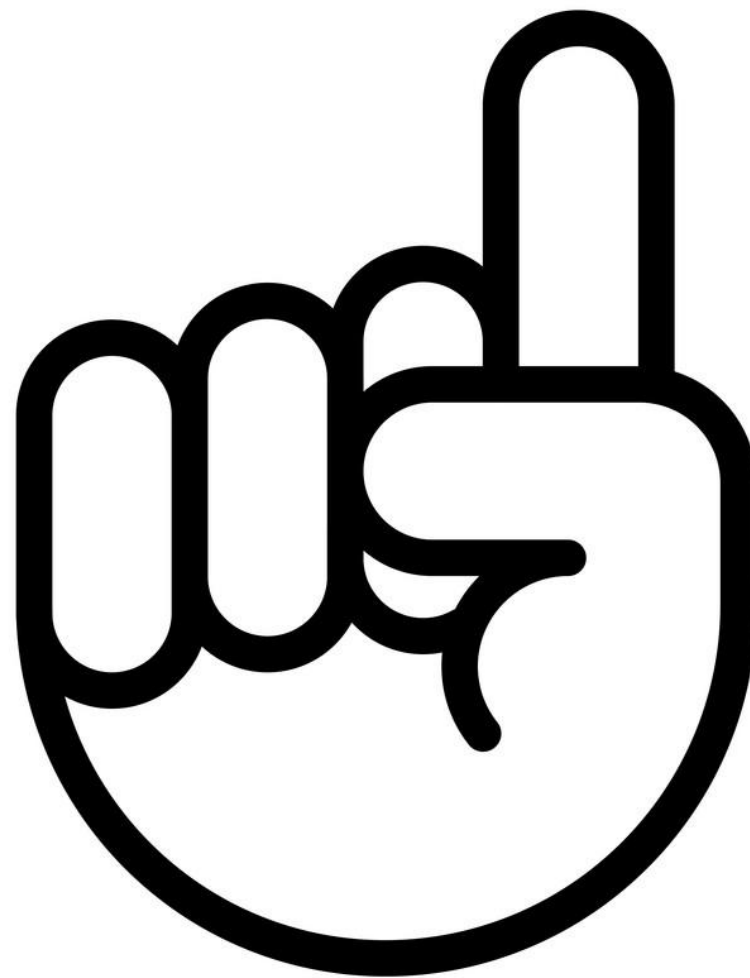
- RMSE – שורש ממוצע השגיאה הריבועית

- MAE – ממוצע השגיאה האבסולוטית

- R^2 – אחוז השונות שהמודל מסביר

אם הביצועים כמעט זהים — נעדיף את המודל המצומצם: פשוט יותר, מהיר יותר, וקל לפרשנות.

Xgboost Regressor



xgboost_classifier

