unacademy

LIVE

Predictors				Target
Outlook	Temp	Humidity	Windy	Hours Played
Rainy	Hot	High	False	25
Rainy	Hot	High	True	30
Overcast	Hot	High	False	46
Sunny	Mild	High	False	45
Sunny	Cool	Normal	False	52
Sunny	Cool	Normal	True	23
Overcast	Cool	Normal	True	43
Rainy	Mild	High	False	35
Rainy	Cool	Normal	False	38
Sunny	Mild	Normal	False	46
Rainy	Mild	Normal	True	48
Overcast	Mild	High	True	52
Overcast	Hot	Normal	False	44
Sunny	Mild	High	True	30

$n = 14$

$$\frac{46 + 43 + 52 + \cdots + 44}{4}$$

$$\sqrt{\frac{(x_i - \bar{x})^2}{4}}$$

LIVE

Decision Tree Algorithm

The core algorithm for building decision trees called **ID3** by J. R. Quinlan which employs a top-down, greedy search through the space of possible branches with no backtracking. The ID3 algorithm can be used to construct a decision tree for regression by replacing Information Gain with *Standard Deviation Reduction*.

unacademy

LIVE

Standard Deviation

$$Standard\ Deviation = S = \sqrt{\frac{\sum(x - \bar{x})^2}{n}}$$

$$Coeffeicient\ of\ Variation = CV = \frac{S}{\bar{x}} * 100\%$$

GO LIVE

unacademy

LIVE

a) Standard deviation for **one** attribute:

o/p variable

Hours Played
25
30
46
45
52
23
43
35
38
46
48
52
44
30

$(25-39.8)^2 + (30-39.8)^2 \cdots$

$$\text{Count} = n = 14$$

$$\text{Average} = \bar{x} = \frac{\sum x}{n} = 39.8$$

$$\text{Standard Deviation} = S = \sqrt{\frac{\sum (x-\bar{x})^2}{n}} = 9.32$$

$$\text{Coeffeicient of Variation} = CV = \frac{S}{\bar{x}} * 100\% = 23\%$$

- Standard Deviation (**S**) is for tree building (branching).
- Coefficient of Deviation (**CV**) is used to decide when to stop branching. We can use Count (**n**) as well.
- Average (**Avg**) is the value in the leaf nodes.

unacademy

$$S(T,X) = \sum_{c \in X} P(c)S(c)$$

		Hours Played (StDev)	Count
Outlook	Overcast	3.49	4
	Rainy	7.78	5
	Sunny	10.87	5
			14

S(Hours, Outlook) = P(Sunny)*S(Sunny) + P(Overcast)*S(Overcast) + P(Rainy)*S(Rainy)

= (4/14)*3.49 + (5/14)*7.78 + (5/14)*10.87

= 7.66

$S = 9.32$

$S(\text{Hours}, \text{Outlook}) = 7.66$

LIVE

Standard Deviation Reduction

The standard deviation reduction is based on the decrease in standard deviation after a dataset is split on an attribute. Constructing a decision tree is all about finding attribute that returns the highest standard deviation reduction (i.e., the most homogeneous branches).

Step 1: The standard deviation of the target is calculated.

Standard deviation (Hours Played) = 9.32

Step 2:
The dataset is then split on the different attributes. The standard deviation for each branch is calculated. The resulting standard deviation is subtracted from the standard deviation before the split. The result is the standard deviation reduction.

$$S(T,X) = \sum_{c \in X} P(c)S(c)$$

		Hours Played (StDev)	Count
Outlook	Overcast	3.49	4
	Rainy	7.78	5
	Sunny	10.87	5
			14

S(Hours, Outlook) = **P**(Sunny)***S**(Sunny) + **P**(Overcast)***S**(Overcast) + **P**(Rainy)***S**(Rainy)

= (4/14)*3.49 + (5/14)*7.78 + (5/14)*10.87

= 7.66

unacademy

LIVE

$$SDR(T,X) = S(T) - S(T,X)$$

SDR(Hours , Outlook) = **S**(Hours) – **S**(Hours, Outlook)

= 9.32 – 7.66 = 1.66

LIVE

		Hours Played (StDev)
Outlook	Overcast	3.49
	Rainy	7.78
	Sunny	10.87
SDR=1.66		

		Hours Played (StDev)
Temp.	Cool	10.51
	Hot	8.95
	Mild	7.65
SDR= 0.48		

		Hours Played (StDev)
Humidity	High	9.36
	Normal	8.37
SDR=0.28		

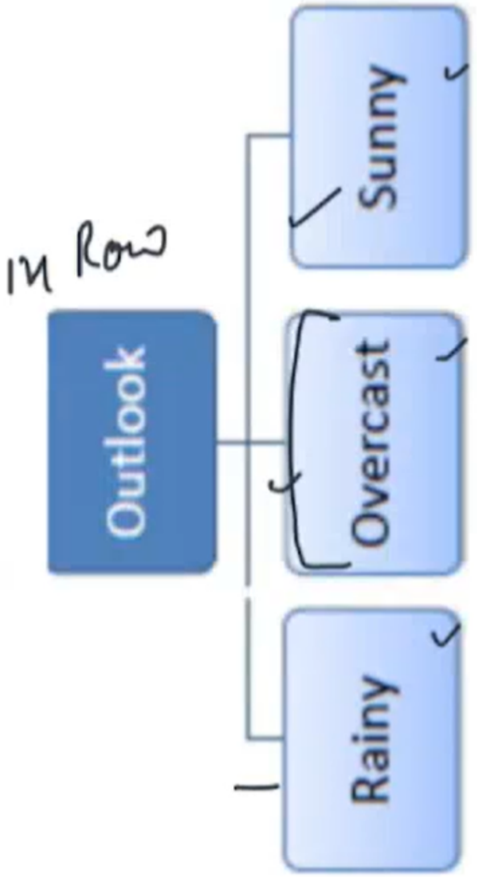
		Hours Played (StDev)
Windy	False	7.87
	True	10.59
SDR=0.29		

Step 3: The attribute with the largest standard deviation reduction is chosen for the decision node.

★		Hours Played (StDev)
Outlook	Overcast	3.49
	Rainy	7.78
	Sunny	10.87
SDR=1.66		

Step 4a:

The dataset is divided based on the values of the selected attribute. This process is run recursively on the non-leaf branches, until all data is processed.



Outlook	Temp	Humidity	Windy	Hours Played
Sunny	Mild	High	FALSE	45
Sunny	Cool	Normal	FALSE	52
Sunny	Cool	Normal	TRUE	23
Sunny	Mild	Normal	FALSE	46
Sunny	Mild	High	TRUE	30
Overcast	Hot	High	FALSE	46
Overcast	Cool	Normal	TRUE	43
Overcast	Mild	High	TRUE	52
Overcast	Hot	Normal	FALSE	44
Rainy	Hot	High	FALSE	25
Rainy	Hot	High	TRUE	30
Rainy	Mild	High	FALSE	35
Rainy	Cool	Normal	FALSE	38
Rainy	Mild	Normal	TRUE	48

Pritesh Saklecha

In practice, we need some termination criteria. For example, when coefficient of deviation (CV) for a branch becomes smaller than a certain threshold (e.g., 10%) and/or when too few instances (n) remain in the branch (e.g., 3).

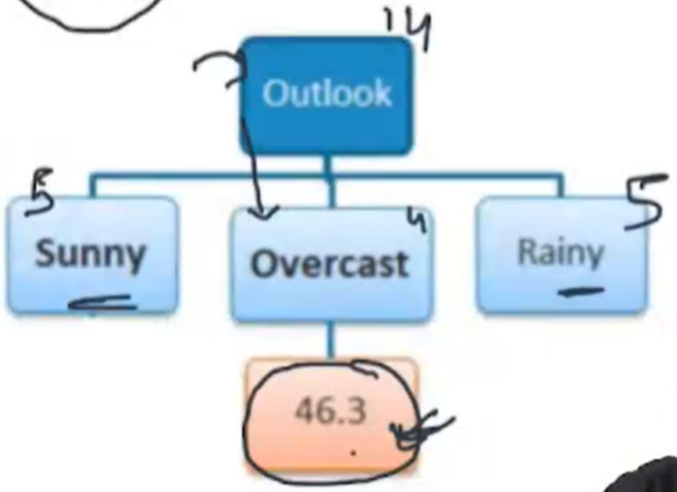
8 < 10

Step 4b: "Overcast" subset does not need any further splitting because its CV (8%) is less than the threshold (10%). The related leaf node gets the average of the "Overcast" subset.

3.49 / 46.3 ← $\frac{S}{\bar{x}} \times 100$

Outlook - Overcast

		Hours Played (StDev)	Hours Played (AVG)	Hours Played (CV)	Count
Outlook	Overcast	3.49	46.3	8%	4
	Rainy	7.78	35.2	22%	5
	Sunny	10.87	39.2	28%	5



LIVE

Step 4c: However, the "Sunny" branch has an CV (28%) more than the threshold (10%) which needs further splitting. We select "Temp" as the best best node after "Outlook" because it has the largest SDR.

$$S = \sqrt{\frac{(x - \bar{x})^2}{n}}$$

Outlook - Sunny

Temp	Humidity	Windy	Hours Played
Mild	High	FALSE	45
Cool	Normal	FALSE	52
Cool	Normal	TRUE	23
Mild	Normal	FALSE	46
Mild	High	TRUE	30
			S = 10.87
			AVG = 39.2
			CV = 28%

		Hours Played (StDev)	Count
Temp	Cool	14.50	2
	Mild	7.32	3

SDR = 10.87-((2/5)*14.5 + (3/5)*7.32) = 0.678

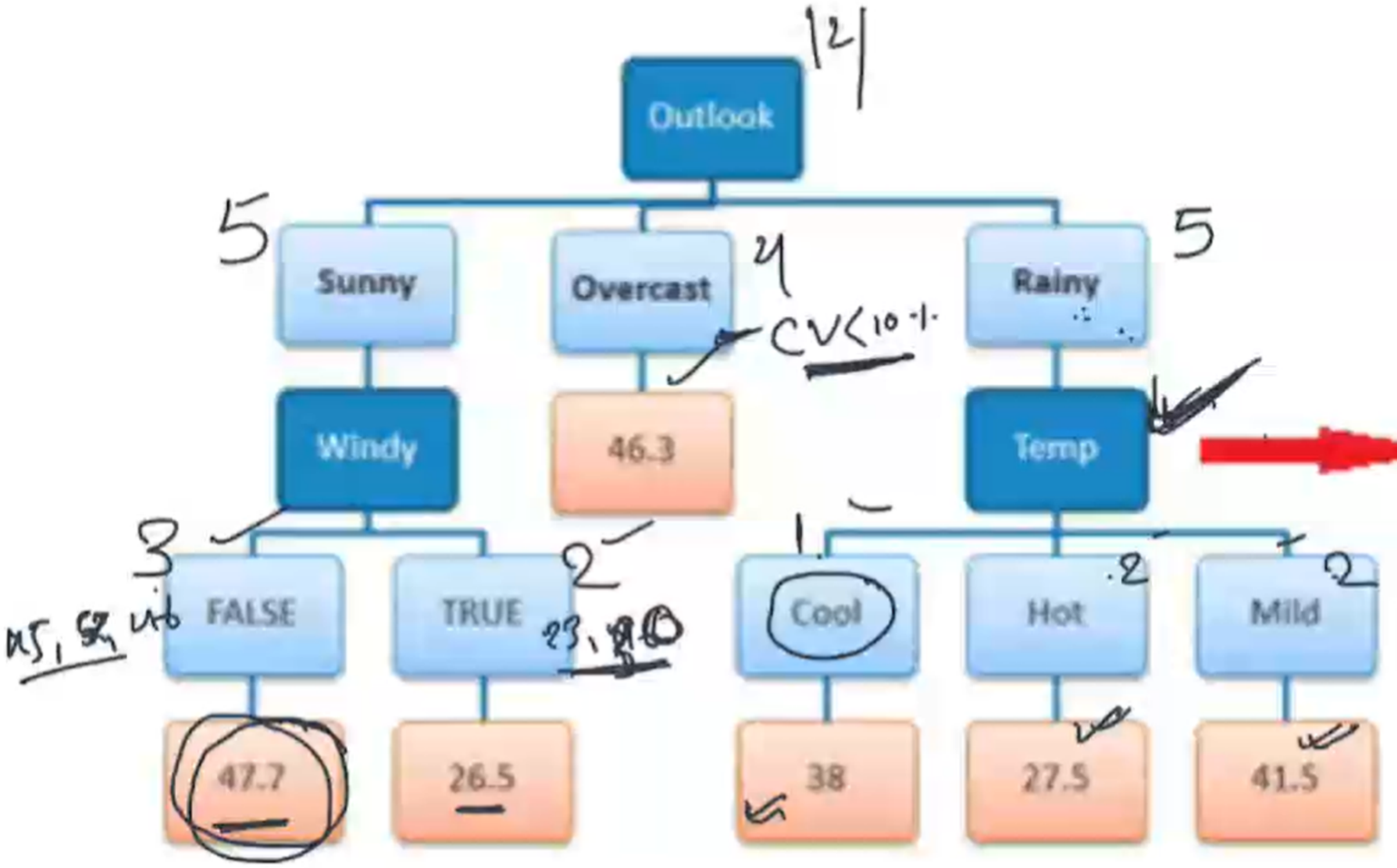
		Hours Played (StDev)	Count
Humidity	High	7.50	2
	Normal	12.50	3

SDR = 10.87-((2/5)*7.5 + (3/5)*12.5) = 0.370

		Hours Played (StDev)	Count
Windy	False	3.09	3
	True	3.50	2

SDR = 10.87-((3/5)*3.09 + (2/5)*3.5) = 7.62

Because the number of data points for all three branches (Cool, Hot and Mild) is equal or less than 3 we stop further branching and assign the average of each branch to the related leaf node.



Temp	Hours Played
Cool	38
Hot	25
Hot	30
Mild	35
Mild	48

ID3 classification → Entropy

$\Rightarrow -\sum p_i \log p_i$

Information Gain

$G(\text{child}) = E(\text{Parent}) - \sum_{\text{node}} P(c) \times E(c)$

Regression

Stand. deviation

$$\sqrt{\frac{\sum (x - \bar{x})}{n}}$$

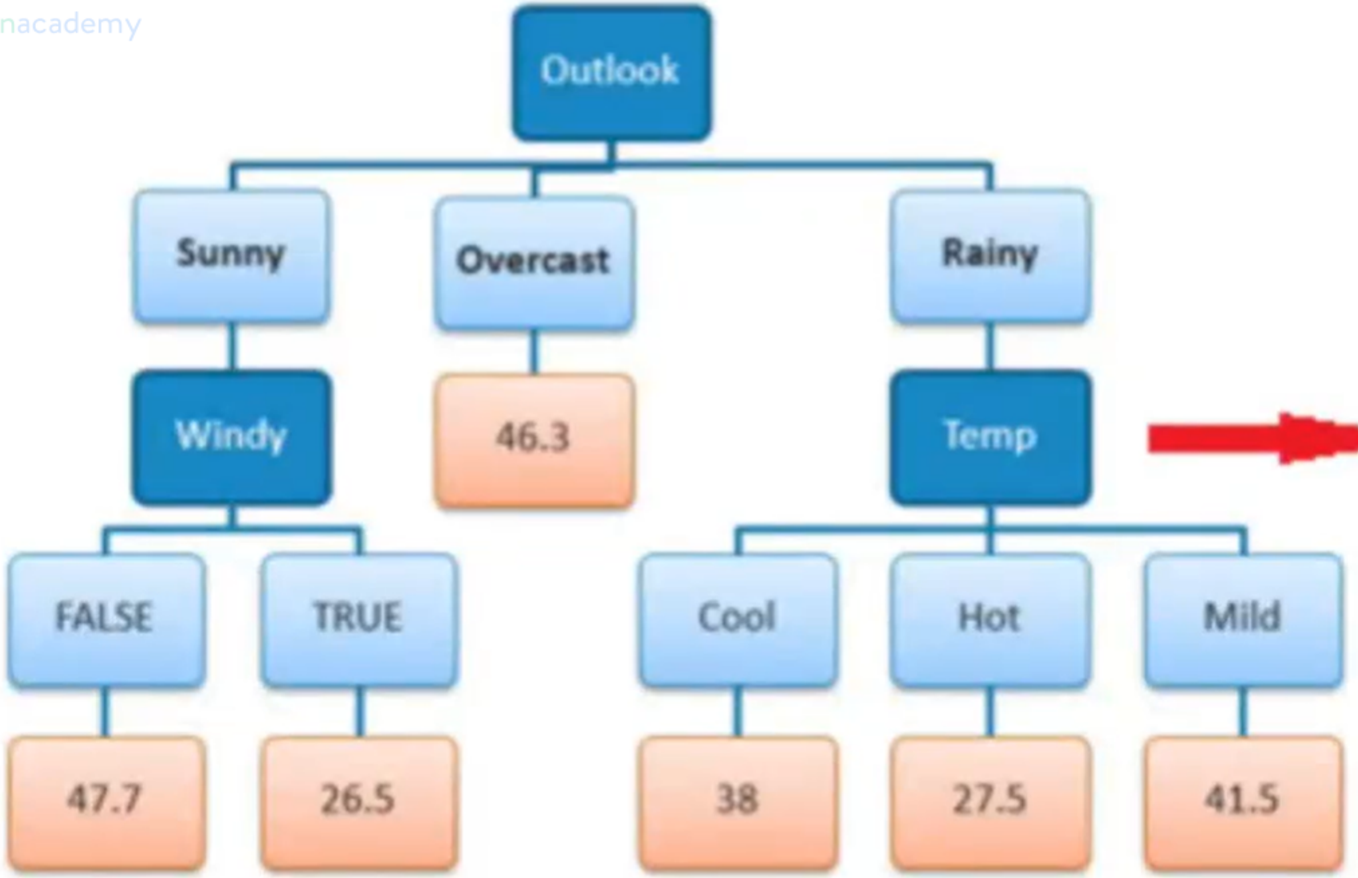
SDR

//

$S(P) - \sum P(c) * S(c)$

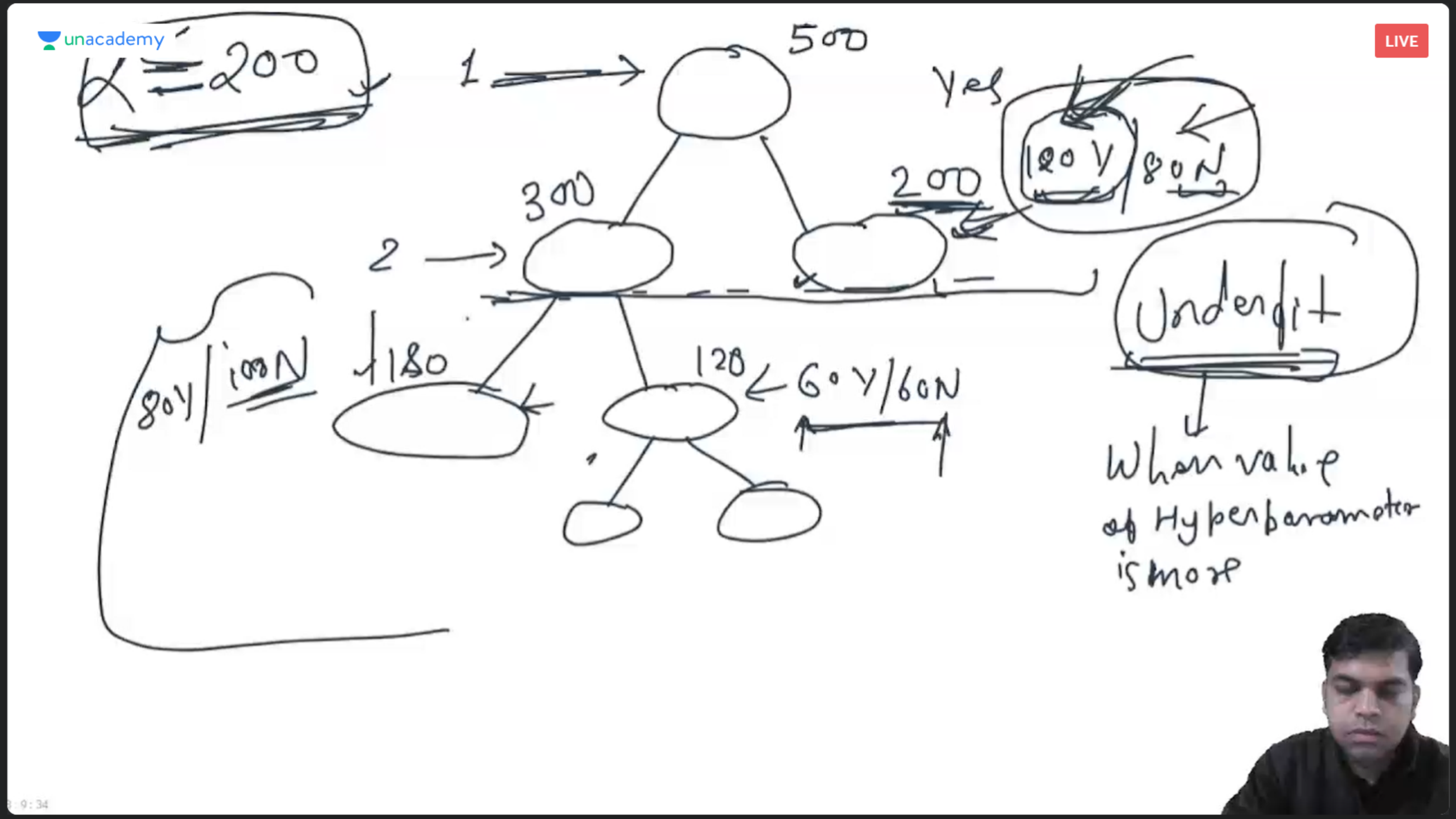
CART classification → $1 - \sum p_i^2$

Pritesh Saklecha



Temp	Hours Played
Cool	38
Hot	25
Hot	30
Mild	35
Mild	48

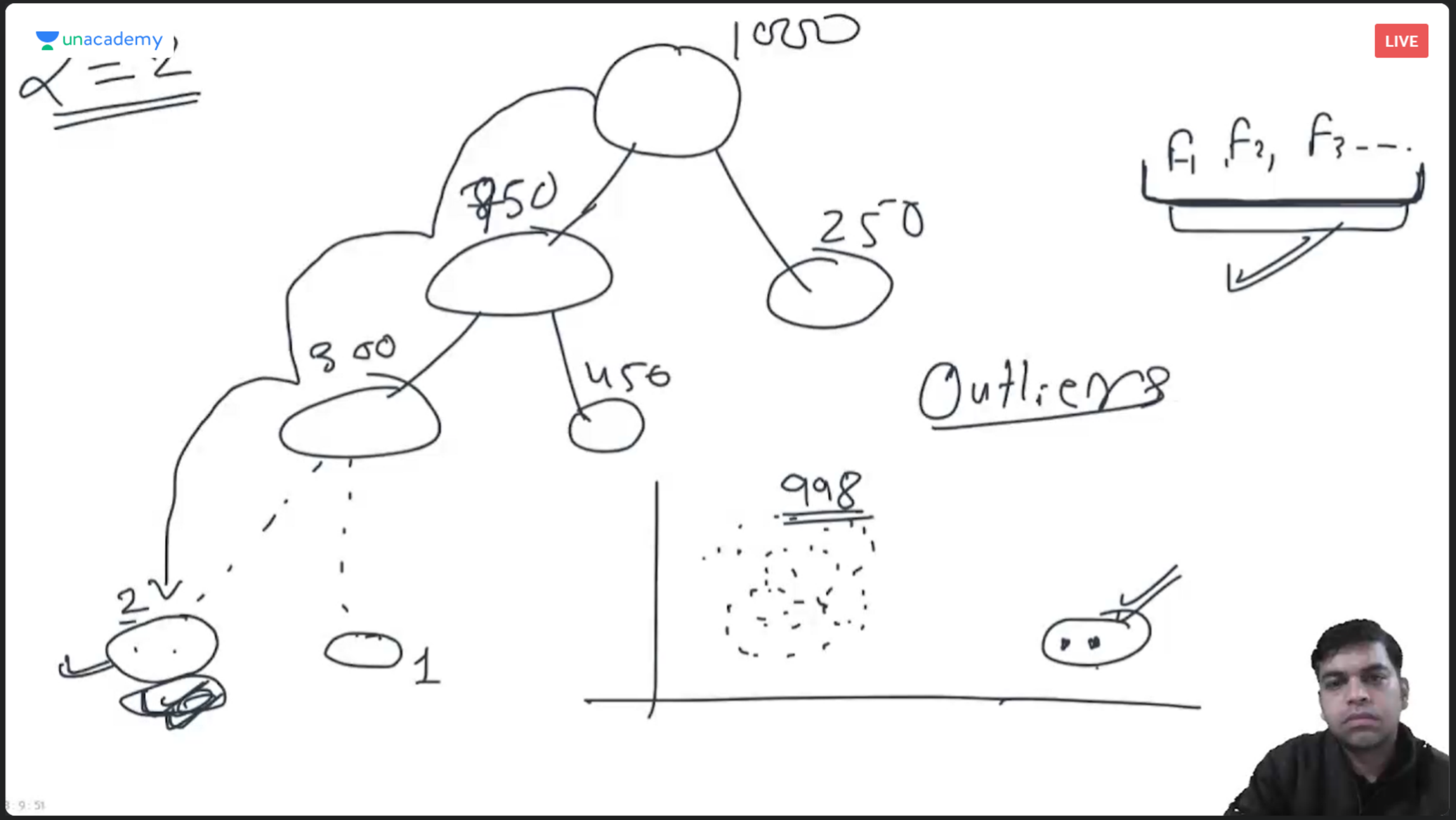
When the number of instances is more than one at a *leaf node* we calculate the *average* as the final value for the target.


K=200
1
500
Yes
100 Y/80N
300
200
2
80Y/100N
180
120
60Y/60N
Underfit
When value of Hyperparameter is more

Hyperparameter → apply Limit on No. of instance (data point) i.e $\alpha \leq 200$

→ Apply depth Limit

(1) If depth is very Low or value of α is very High then there is problem of Underfiting → High variance
→ High Bias


unacademy
LIVE
α = 2
1000
850
250
300
450
f1 f2, f3 ---
Outliers
998
2
1
GO LIVE

If value of α is very very less

or if depth Limit is very high then

there is a problem of overfiting

$\alpha = 2$ or depth $= 20$

Pritesh Saklecha

unacademy

LIVE

by Applying depth Limit of value of α, we just prune the node of tree

means we are not explore nodes below depth Limit

Pruning.

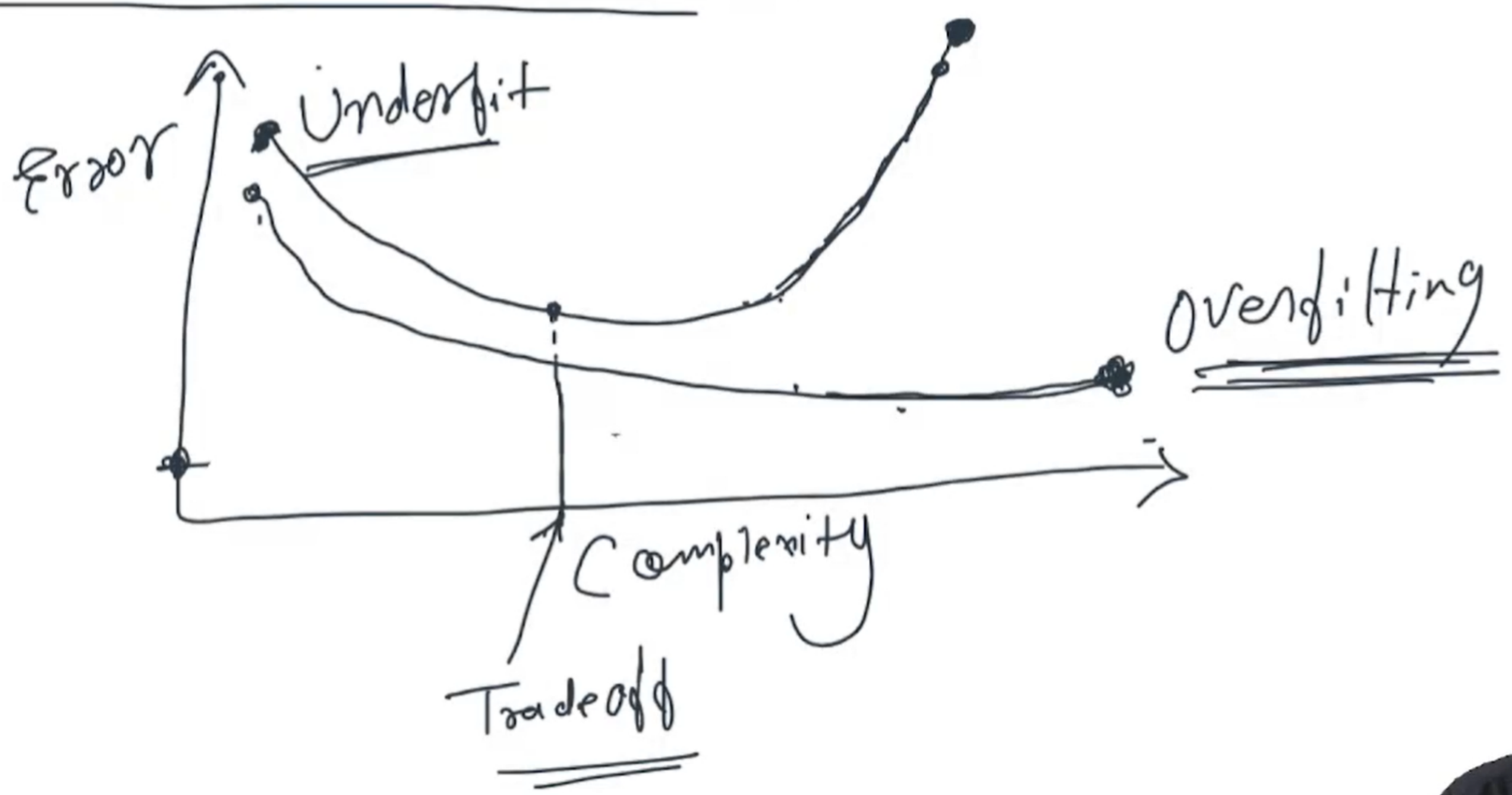
- Pre pruning → Apply depth Limit before tree construction
- Post prune
  - → Apply depth Limit after tree Construction

LIVE

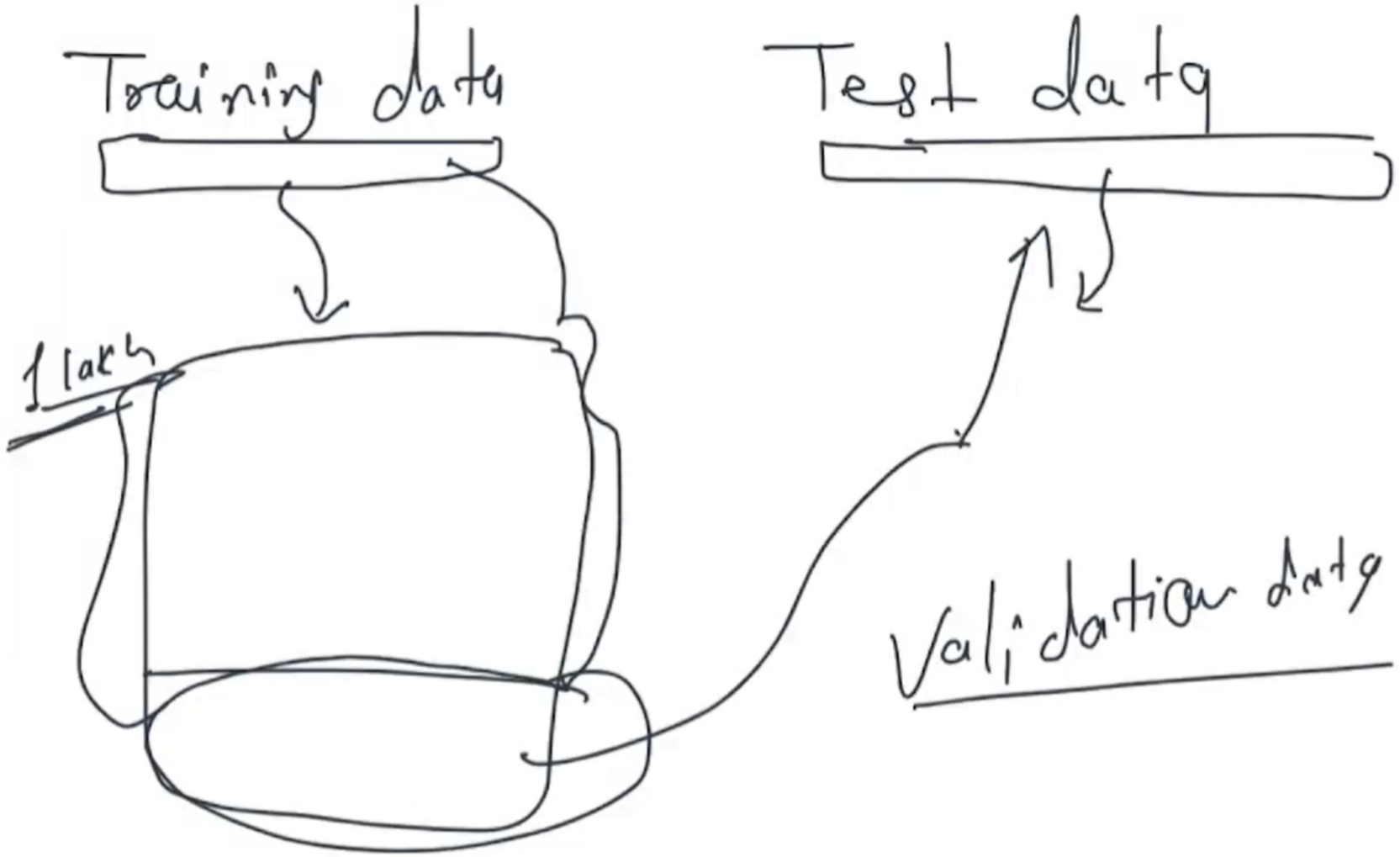
unacademy

LIVE

Bias - Variance Tradeoff



LIVE



In machine learning, the terms training data, testing data, and validation data refer to different subsets of a dataset used for various stages in the development and evaluation of a model. These subsets play crucial roles in training and assessing the performance of machine learning models. Here's a brief explanation of each:

Training Data:

- **Purpose:** This is the subset of data used to train the machine learning model.
- **Usage:** During the training phase, the model learns patterns and relationships within the training data.
- **Size:** The training dataset is typically the largest portion of the overall dataset.

Testing Data (or Test Data):

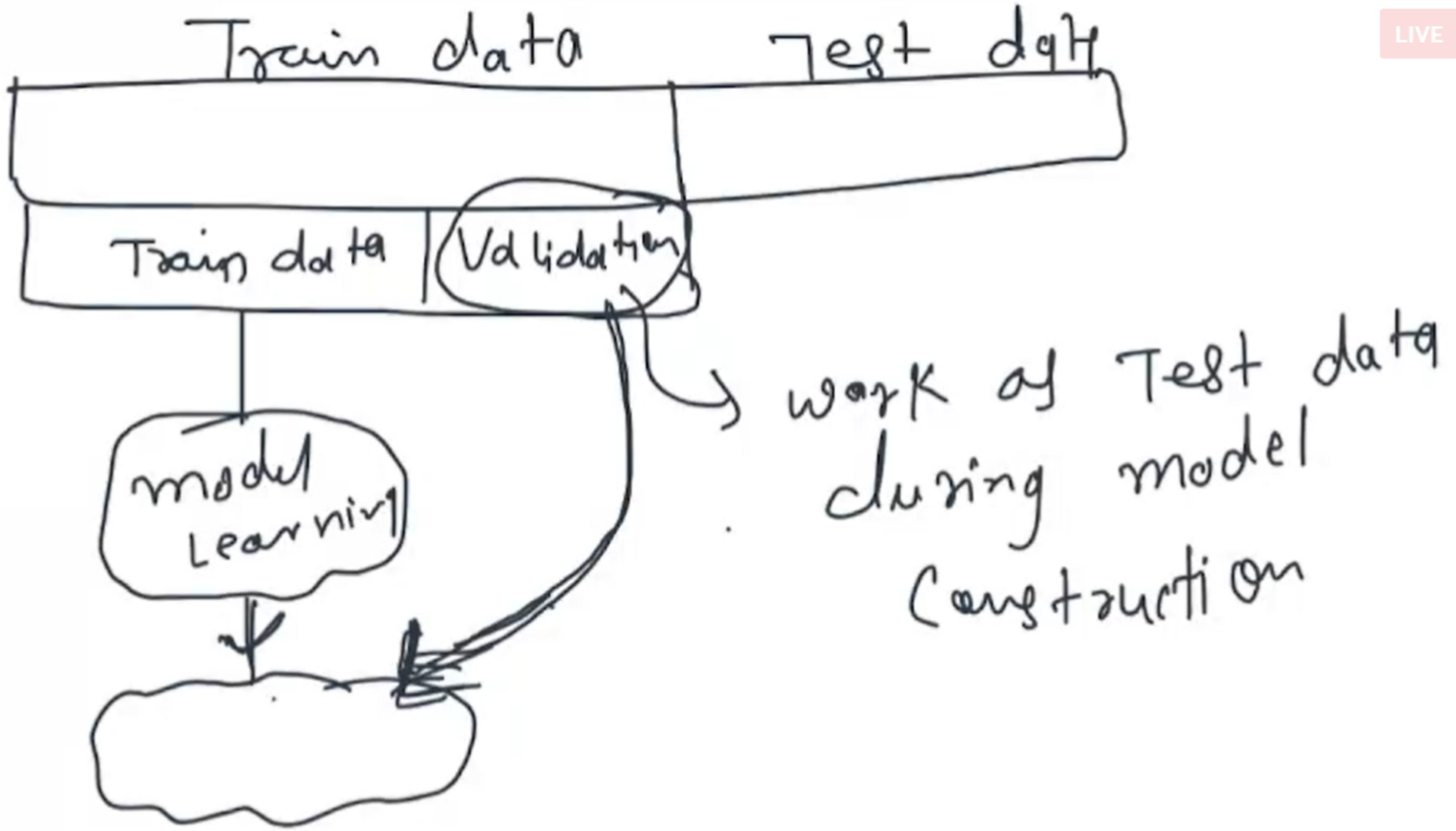
- **Purpose:** This is a separate subset of data used to evaluate the performance of the model after it has been trained.
- **Usage:** The model has never seen the testing data during training, so its ability to generalize to new, unseen examples is assessed using the testing data.
- **Size:** The testing dataset is held out until the model has been trained to avoid biasing the evaluation.

Validation Data:

•**Purpose:** This is another subset of data used to fine-tune the model during the training phase.

•**Usage:** It helps to assess the model's performance on data it has not seen during training and provides a basis for adjusting hyperparameters to improve generalization.

•**Size:** The validation dataset is used to make decisions about the model's architecture or hyperparameters, and it is not used in the final evaluation of the model's performance.



The typical split among these subsets can vary, but a common practice is to use a large portion (e.g., 70-80%) for training, a smaller portion (e.g., 10-15%) for testing, and the rest for validation. The exact split depends on factors such as the size of the dataset and the specific requirements of the machine learning task. The key idea is to ensure that the model is trained on diverse data, tested on unseen data, and validated to improve its generalization capabilities.

Validation data plays a crucial role in the training process of a machine learning model, particularly in the context of fine-tuning and improving its performance. Here are more details about validation data:

Purpose of Validation Data:

•**Fine-Tuning Hyperparameters:** During the training phase, a machine learning model has various hyperparameters (e.g., learning rate, regularization strength) that need to be set. The goal is to find the combination of hyperparameter values that results in the best model performance. The validation dataset is used to evaluate the model's performance for different hyperparameter settings, helping to choose the best configuration.

•**Preventing Overfitting**: Overfitting occurs when a model performs well on the training data but fails to generalize to new, unseen data. The validation data helps monitor the model's performance on examples it hasn't seen during training. If the model performs well on the training data but poorly on the validation data, it may be overfitting. Adjustments can then be made to prevent overfitting, such as reducing model complexity or applying regularization techniques.

Size of Validation Data:

The size of the validation dataset is crucial. It should be large enough to provide a representative sample of the data but not so large that it significantly reduces the amount of data available for training. Common splits include 80% for training, 10% for validation, and 10% for testing.

validation data is used to fine-tune a model during training, making adjustments to hyperparameters and preventing overfitting. It helps ensure that the model generalizes well to new, unseen data by providing an unbiased evaluation throughout the training process.

Cross Validation

- in a real-life scenario, the model will be tested for its efficiency and accuracy with an altogether different and unique data set.
- Under those circumstances, you'd want your model to be efficient enough or at least to be at par with the same efficiency that it shows for the training set.
- Basically this testing is known as cross-validation in Machine Learning so that it is fit to work with any model in the future.

Cross Validation

- Cross validation is a technique for assessing how the statistical analysis generalizes to an independent data set.
- It is a technique for evaluating machine learning models by training several models on subsets of the available input data and evaluating them on the complementary subset of the data.
- Using cross-validation, there are high chances that **we can detect over-fitting with ease.**

unacademy

Types Of Cross-Validation

LIVE

There are two types of cross-validation techniques in Machine Learning.

- **Exhaustive Cross-Validation** – This method basically involves testing the model in all possible ways, it is done by dividing the original data set into training and validation sets. Example: Leave-p-out Cross-Validation, Leave-one-out Cross-validation.
- **Non-Exhaustive Cross-Validation** – In this method, the original data set is not separated into all the possible permutations and combinations. Example: K-fold Cross-Validation, Holdout Method.

13 : 25

LIVE

Various Types of cross validation

- There are several **cross validation techniques** such as :-
- 1. Leave One-out Cross Validation
  2. Leave P-out Cross Validation
  3. K-Fold Cross Validation
  4. Stratified K-Fold Cross Validation
  5. Holdout Method

Leave P out

- In this approach, **p** data points are left out of the training data. Let's say there are **m** data points in the data set, then **m-p** data points are used for the training phase. And the **p** data points are kept as the validation set.
- This technique is rather exhaustive because the above process is repeated for all the possible combinations in the original data set. To check the overall effectiveness of the model, the error is averaged for all the trials.
- It becomes computationally infeasible since the model needs to train and validate for all possible combinations and for a considerably large **p**.

Leave 1 out