

Indian Institute of Technology Madras
ZANZIBAR CAMPUS

Z3003 · DATA MINING: SCALING TO LARGE DATA

Overfitting, Pruning and Scaling Decision Trees

Tuesday 3 November 2026 · 50 minutes

Dr. Innocent Nyalala | BS Semester 5 (DSAI) | IIT Madras Zanzibar

By the End of Today You Can

IIT Madras Zanzibar

1. Explain overfitting, and spot it from training vs test accuracy
2. Prune a tree: stop early (pre-pruning) or cut back (post-pruning)
3. Choose the tree size with a validation set, then test once
4. Explain how trees are built fast on millions of rows

Today's Plan (40 minutes + questions)

IIT Madras Zanzibar

minutes	part	what you get
0 – 10	What overfitting is	the past-paper student; seen vs new farms
10 – 21	Pruning	stop early; grow then cut; ID3, C4.5, CART
21 – 27	Choosing the size honestly	train / validation / test; code
27 – 35	Trees on big data + Weka	sampling, histograms, SPRINT, streaming; J48 demo
35 – 40	Exercise and summary	then questions and further reading

Recap: What Happened When the Tree Grew Deeper?

IIT Madras Zanzibar

2 questions

new farms: 66% right

too simple

6 questions

new farms: 71% right

best

no limit

training farms: 100% right
new farms: 62.9% right

worse!

More questions made the tree better on farms it had SEEN, and worse on NEW farms.
Why? And how do we stop it?

The Past-Paper Student

IIT Madras Zanzibar

Underfitting

Barely studied.
Last year's paper: poor
New exam: poor

too simple

Good fit

Learned the method.
Last year's paper: good
New exam: good

what we want

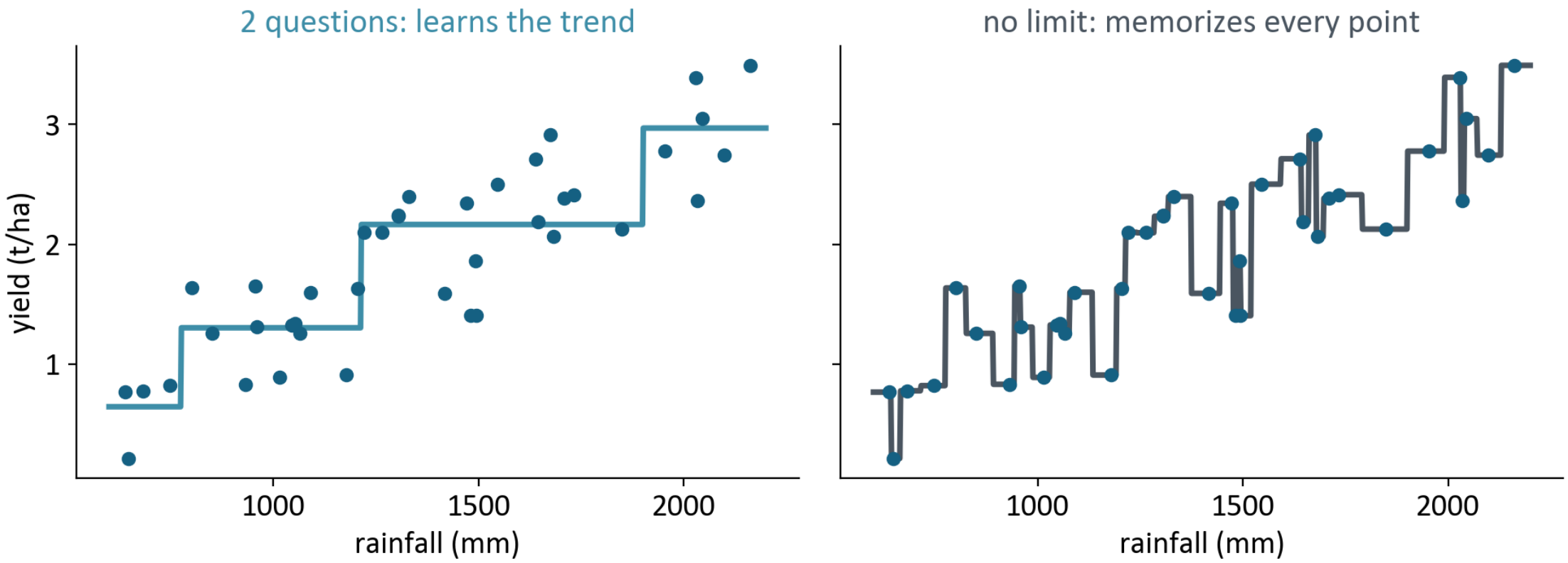
Overfitting

Memorized the answers.
Last year's paper: 100%
New exam: poor

our full tree

Definition: overfitting = learning the noise and accidents of the training data, so accuracy on new data falls

Overfitting in One Picture



Left: follows the trend. Right: passes through every dot, so it has learned the noise.

How to Spot It: Two Numbers

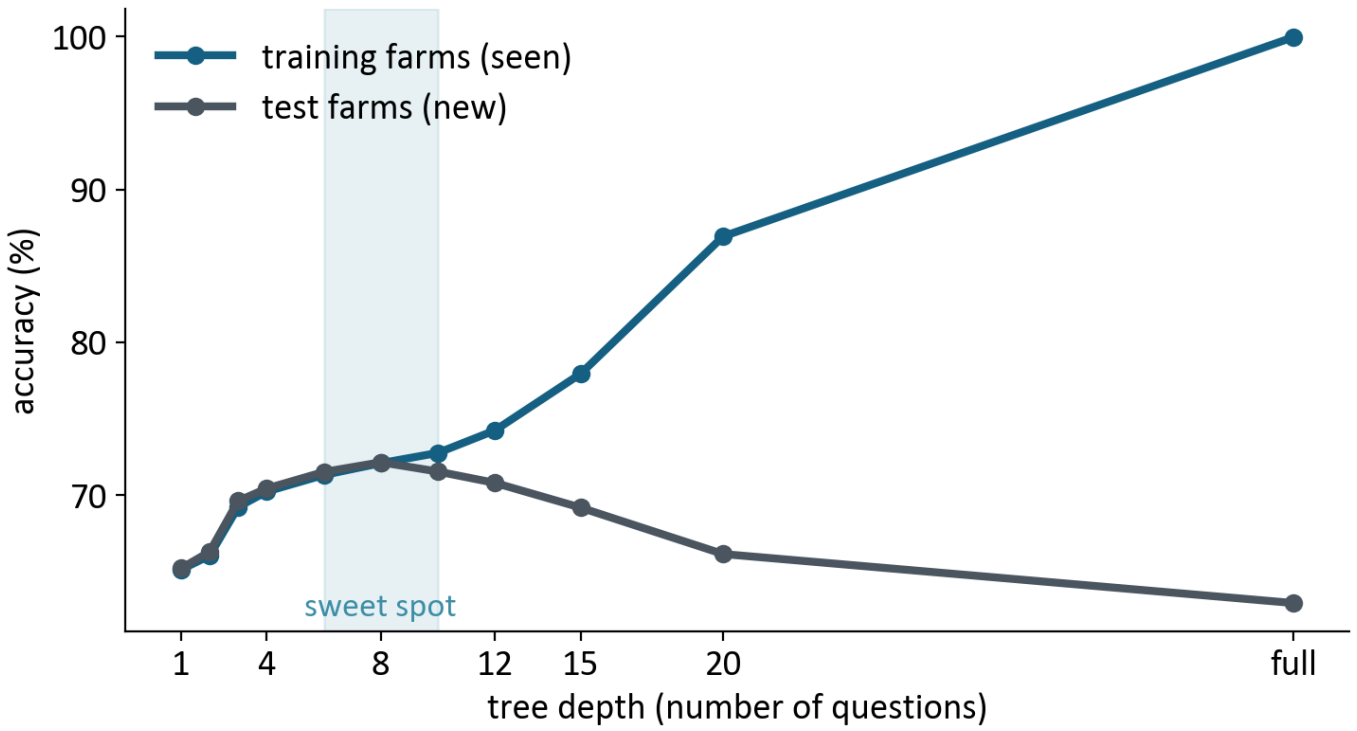
IIT Madras Zanzibar

1	**Underfitting**	training LOW, test LOW
2	**Good fit**	training good, test good, small gap
3	**Overfitting**	training HIGH, test much lower: big gap

Definition: training / test accuracy = share correct on farms the model learned from / on farms it has never seen

Rule: never judge a model by training accuracy. The train–test GAP is the warning light.

Measured on Our 190,000 Farm-Seasons



depth	train	test	leaves
3	69.2%	69.6%	8
8	72.1%	72.1%	249
15	77.9%	69.2%	7,051
full	100%	62.9%	38,513

train = % correct on the 132,972 training seasons (70% of 189,960); test = % correct on the 56,988 hidden seasons (30%). leaves = end boxes of the tree. Full tree: 100% on seen farms, 62.9% on new ones, worse than a 3-question tree.

Why Deep Trees Overfit: Tiny Leaves

IIT Madras Zanzibar

132,972
training farms
(70% of 189,960)

÷

38,513
leaves in the full tree

=

≈ 3.5
farms per leaf

A 'rule' learned from 3 farms is luck, not a pattern:
one odd season, one pest outbreak, one typing error.

The cure: stop before leaves become too small to trust. That is called pruning.

Cure 1: Pre-Pruning, Stop Growing Early

IIT Madras Zanzibar

1	**Max depth**	at most 8 questions in a row (max_depth=8)
2	**Min farms to split**	do not split a group under 200 (min_samples_split=200)
3	**Min farms per leaf**	every leaf keeps at least 100 (min_samples_leaf=100)
4	**Min improvement**	split only if purity improves enough (min_impurity_decrease)

+ fast: the huge tree is never built. – risk: may stop just before a useful question.

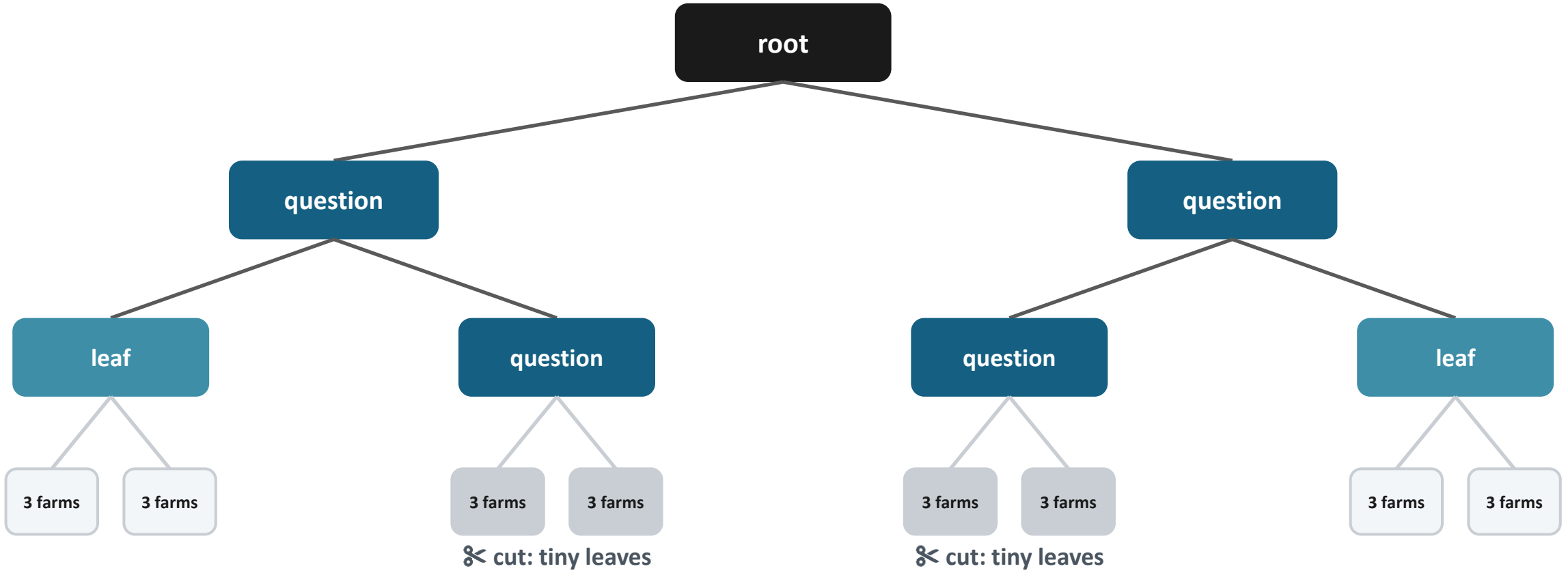
Pre-Pruning on Our Data

IIT Madras Zanzibar

setting	leaves	train	test (new farms)
full tree (no limits)	38,513	100%	62.9%
min_samples_leaf = 100	1,012	72.6%	71.5%
min_samples_leaf = 1000	101	71.8%	72.0%
max_depth = 8, min_samples_leaf = 100	222	72.0%	72.1%

38,513 ÷ 101 ≈ 380 times fewer leaves, and 72.0 − 62.9 ≈ 9 points MORE accurate on new farms.
min_samples_leaf = 100 means: no leaf may hold fewer than 100 training farms.

Cure 2: Post-Pruning, Grow Then Cut Back



> **Grow fully, then cut branches that do not earn their place. Their parent becomes a leaf.**

Post-Pruning: A Price per Leaf (α)

IIT Madras Zanzibar

cost(tree) = error + α × (number of leaves)
error = share of training farms the tree gets wrong; α = price per leaf

α (ccp_alpha)	leaves	test accuracy
0 (no price)	30,681	62.7%
0.00002	2,117	69.8%
0.0001	62	72.0%
0.002	10	70.5%

Example, α = 0.0001: a split that adds 1 leaf must lower the error by more than 0.0001 (1 farm in 10,000).
If it fixes only 1 farm in 100,000 (0.00001), the price is bigger than the gain → cut it. Best here: 62 leaves, 72.0%.

Choosing the Size Honestly: Three Sets

IIT Madras Zanzibar

TRAINING 56%
grow the trees (homework)

VALIDATION 14%
choose the size (mock exam)

TEST 30%
score once (final exam)

Golden rule: never tune on the test set. Choose with validation, then test ONCE.
How: hide 30% as test; split the other 70% into 80% / 20%: 70% × 0.8 = 56% train, 70% × 0.2 = 14% validation.

Choose the Depth on Validation, Then Test Once

IIT Madras Zanzibar

depth	validation accuracy
2	66.8%
4	70.4%
6	71.5%
8	71.7% ← best
10	71.1%
12	70.1%
15	68.1%

Choose depth 8
then open the test set ONCE:
72.1% on new farms

Pruning in Python, the Honest Way

IIT Madras Zanzibar

```
 1  from sklearn.tree import DecisionTreeClassifier
 2  from sklearn.model_selection import train_test_split
 3  Xtv, Xte, ytv, yte = train_test_split(X, y, test_size=0.3)   # lock test
 4  Xtr, Xva, ytr, yva = train_test_split(Xtv, ytv, test_size=0.2)  # validation
 5  scores = {}
 6  for depth in [2, 4, 6, 8, 10, 12, 15]:
 7      t = DecisionTreeClassifier(max_depth=depth).fit(Xtr, ytr)
 8      scores[depth] = t.score(Xva, yva)          # mock exam
 9  best = max(scores, key=scores.get)                              # 8
10  final = DecisionTreeClassifier(max_depth=best).fit(Xtv, ytv)
11  print(best, final.score(Xte, yte))            # real exam, once: 0.721
```

line	what it does
3	lock away the test set
4	make a validation set
6 – 8	try depths, score on validation
9	pick the best depth
10	refit on train + validation
11	test once

Same loop for ccp_alpha or min_samples_leaf: only line 7 changes.

Trees on Big Data: Where Does the Time Go?

IIT Madras Zanzibar

14,892
different rainfall values; a cut goes halfway between two neighbors → 14,891 cuts to test

× every attribute × every node
and with big data, every level = another pass

Fix 1: sample

train on a random sample of rows

Fix 2: bin

test 31 bin edges, not 14,891 cuts

Fix 3: share

count on many machines, add up

Same lesson as Week 5: with big data, the cost is passes over the data, not arithmetic.

Fix 1: Train on a Sample

IIT Madras Zanzibar

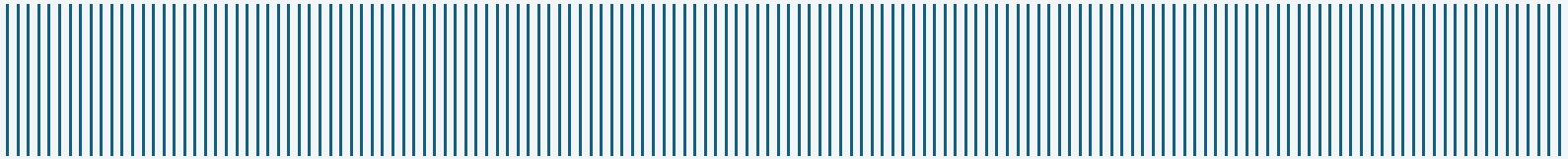
training farms	test accuracy	time
5,000	68.6%	0.03 s
20,000	71.2%	0.09 s
50,000	71.5%	0.22 s
132,972 (all)	72.1%	0.61 s

50,000 → all farms: +0.6 points, about 3 × the time. Accuracy grows slowly with more rows.

Fix 2: Bin the Values (Histogram Splits)

IIT Madras Zanzibar

exact: 14,891 cuts

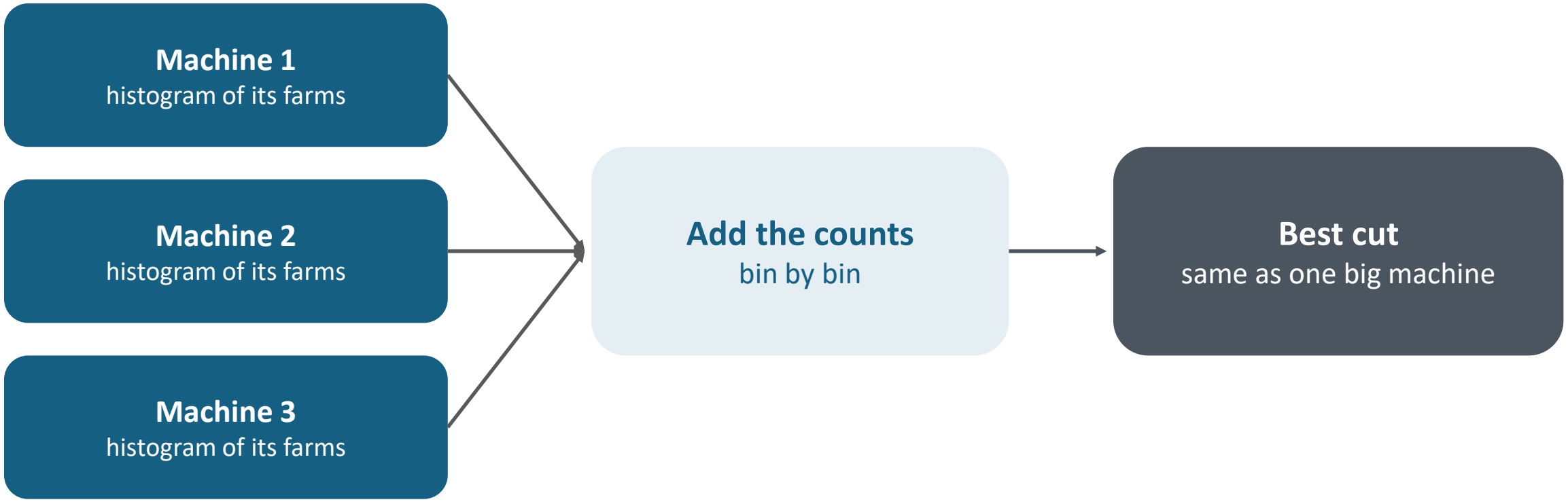


32 bins: 31 cuts
(one between each pair of bins)



method	cuts tested (rainfall)	test accuracy
exact splits	14,891	72.15%
32 bins	31	71.96%

Fix 3: Share the Counting Across Machines



Same idea as SON (Week 5): work on pieces, then combine the counts.

Speed-Ups in Python

IIT Madras Zanzibar

```
1  from sklearn.ensemble import HistGradientBoostingClassifier
2  idx = Xtr.sample(50_000, random_state=0).index              # Fix 1: sample
3  small = DecisionTreeClassifier(max_depth=8).fit(Xtr.loc[idx], ytr.loc[idx])
4  Xb = Xtr.copy()                                             # Fix 2: bins
5  Xb["rainfall_mm"] = pd.qcut(Xtr["rainfall_mm"], 32, labels=False)
6  binned = DecisionTreeClassifier(max_depth=8).fit(Xb, ytr)
7  hist = HistGradientBoostingClassifier(max_bins=32)           # bins built in
8  hist.fit(Xtr, ytr)
```

line	what it does
2 – 3	train on a sample
4 – 6	bin rainfall, train
7 – 8	automatic histogram trees

HistGradientBoosting (ensembles, Mon 9 Nov) uses histogram splits, like LightGBM and XGBoost.

Pruned and Scaled Trees in the Real World

IIT Madras Zanzibar

Credit scoring

millions of loans, histogram splits

Competitions

LightGBM, XGBoost on tables

Telecom

Spark trees on billions of calls

Agriculture

yield risk from satellite + weather

Fraud

small pruned trees, fast checks

Medicine

shallow trees doctors can audit

Common thread: prune for honesty, sample and bin for speed.

Three Classic Tree Algorithms and How They Prune

IIT Madras Zanzibar

algorithm	split measure	data	pruning	where you find it
ID3 (Quinlan, 1986)	information gain	categories only	none	teaching; Weka package
C4.5 (Quinlan, 1993)	gain ratio	numbers, missing values	error-based (pessimistic) post-pruning	Weka J48
CART (Breiman et al., 1984)	Gini index	numbers; binary splits	cost-complexity (α per leaf)	scikit-learn, R rpart
REPTree	information gain	numbers	reduced-error pruning on validation data	Weka REPTree

Same idea everywhere: grow, then remove branches that do not earn their place on new data.

Reduced-Error Pruning, Step by Step



keep the branch: 18 + 33 = 51 of 75 right

step	what	here
1	start at the bottom of the tree	pick a branch
2	compare on validation farms	branch 51 vs leaf 55
3	cut if the leaf does as well or better	55 ≥ 51 → cut

Six Ways to Fight Overfitting

IIT Madras Zanzibar

1. Prune

stop early or cut back

2. More data

2,000 farms → depth 4; 133,000 → depth 8

3. Fewer attributes

each useless one is a chance to learn noise

4. Ensembles

average many trees (next Monday)

5. Cross-validation

choose settings on many folds (Thursday)

6. Occam's razor

equal fit → prefer the simpler model (MDL)

Overfitting is the central danger of data mining: every method has its own cure.

Tree Algorithms Built for Data That Does Not Fit in Memory

IIT Madras Zanzibar

algorithm	key idea	big-data trick
SLIQ (Mehta et al., 1996)	sort each attribute once; class list in memory	pre-sorting
SPRINT (Shafer et al., 1996)	attribute lists on disk, split across machines	parallel, no memory limit
RainForest (Gehrke et al., 1998)	keep only counts per value and class (AVC sets)	counting, one scan per level
BOAT (Gehrke et al., 1999)	tree from bootstrap samples, then one correcting pass	sampling
Hoeffding tree / VFDT (2000)	split once enough stream examples arrive	streaming
LightGBM, XGBoost	histogram splits for boosted trees	binning

All of them count instead of storing: the same lesson as frequent itemsets in Week 5.

Pruning in Weka: No Code (5-Minute Demo)

IIT Madras Zanzibar

1	**Open**	Explorer → Preprocess → Open file → farm_seasons_5000.csv
2	**Pruned tree**	Classify → Choose → trees → J48 → 10-fold cross-validation → Start
3	**Unpruned tree**	J48 settings: unpruned = True → Start → more leaves, lower accuracy
4	**Tune**	confidenceFactor (smaller = more pruning); minNumObj = min farms per leaf

Right-click the result → Visualize tree. REPTree in Weka uses reduced-error pruning.

Your Turn (4 minutes, individually)

tree	training	test
A	66%	65%
B	74%	72%
C	99%	63%

1. Which overfits? Which underfits? Which to deploy?
2. Two pre-pruning settings that turn C into B
3. Why choose depth on validation, not test?
4. 50 million rows: two ways to speed up

Solution

IIT Madras Zanzibar

1	Q1	C overfits (99 vs 63 gap); A underfits (both low); deploy B
2	Q2	max_depth = 8 and min_samples_leaf = 100 (or ccp_alpha)
3	Q3	tuning on test makes its score too optimistic; use validation
4	Q4	sample the rows; bin values (histograms); share across machines

Summary

IIT Madras Zanzibar

1	**Overfitting**	learns noise; big train–test gap (100% vs 62.9%)
2	**Pruning**	stop early (max_depth, min_samples_leaf) or cut back (α)
3	**Honest choice**	choose on validation, test once
4	**Big data**	sample rows, bin values, share the counting

Next: evaluating classifiers. Why accuracy alone can fool you.

Questions

IIT Madras Zanzibar

Q: Is a deeper tree always worse?

A: No: better until the sweet spot (about depth 8 here), then worse.

Q: Pre- or post-pruning?

A: Post usually finds slightly better trees; pre is faster. Both work.

Q: Does Naive Bayes overfit?

A: Much less: it estimates only a few counts. It can underfit.

Q: Why not train on all the data?

A: Then nothing is left to check the model honestly.

Further Reading: Overfitting, Pruning and Scalable Trees

IIT Madras Zanzibar

resource	why read it	access
Han, Pei & Tong, Data Mining: Concepts and Techniques, 4th ed. (2022)	tree induction, pruning, RainForest, BOAT	library
Tan, Steinbach, Karpatne & Kumar, Introduction to Data Mining, 2nd ed. (2019)	overfitting and model complexity	library
Hastie, Tibshirani & Friedman, The Elements of Statistical Learning, 2nd ed. (2009)	cost-complexity pruning	free: hastie.su.domains/ElemStatLearn
Quinlan, C4.5: Programs for Machine Learning (1993); Breiman et al., CART (1984)	the original algorithms	library
Esposito, Malerba & Semeraro (1997), IEEE TPAMI 19(5): 476–491	comparison of pruning methods	journal
Shafer, Agrawal & Mehta (1996), VLDB: SPRINT; Gehrke et al. (1998), VLDB: RainForest	trees on disk-sized data	proceedings
Domingos & Hulten (2000), KDD: Mining high-speed data streams	Hoeffding tree (VFDT)	proceedings
scikit-learn User Guide: Decision Trees; Weka J48 documentation	settings and code	free: scikit-learn.org, Weka