TREE-BASED MODELS

Lecture 5

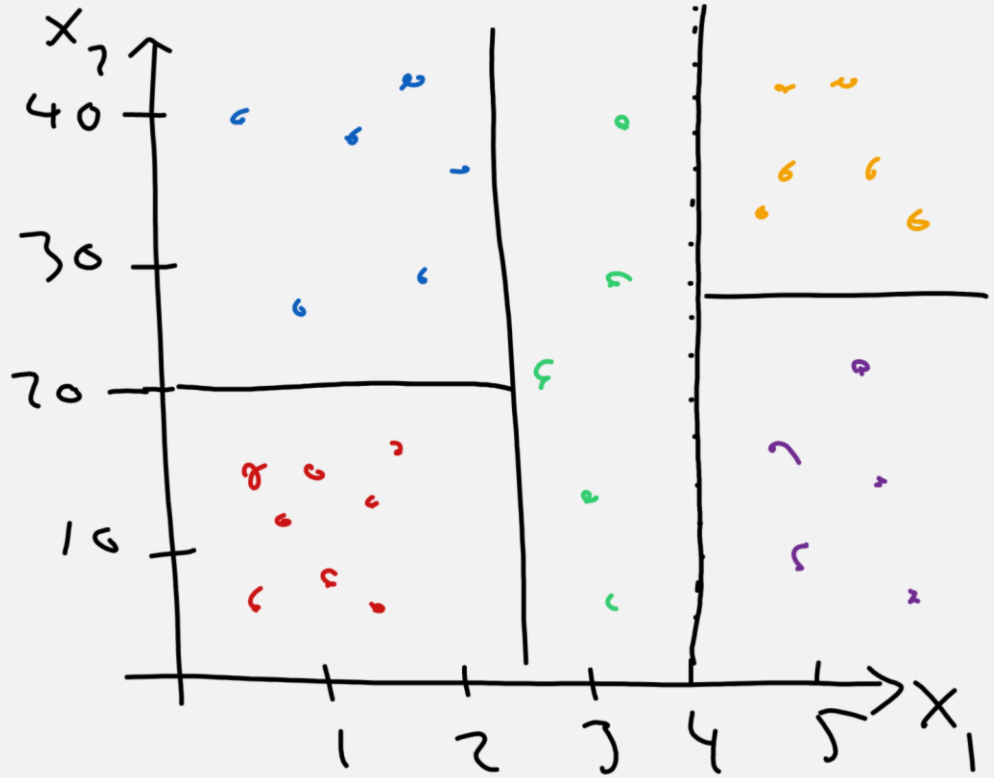
MAL1, 2025

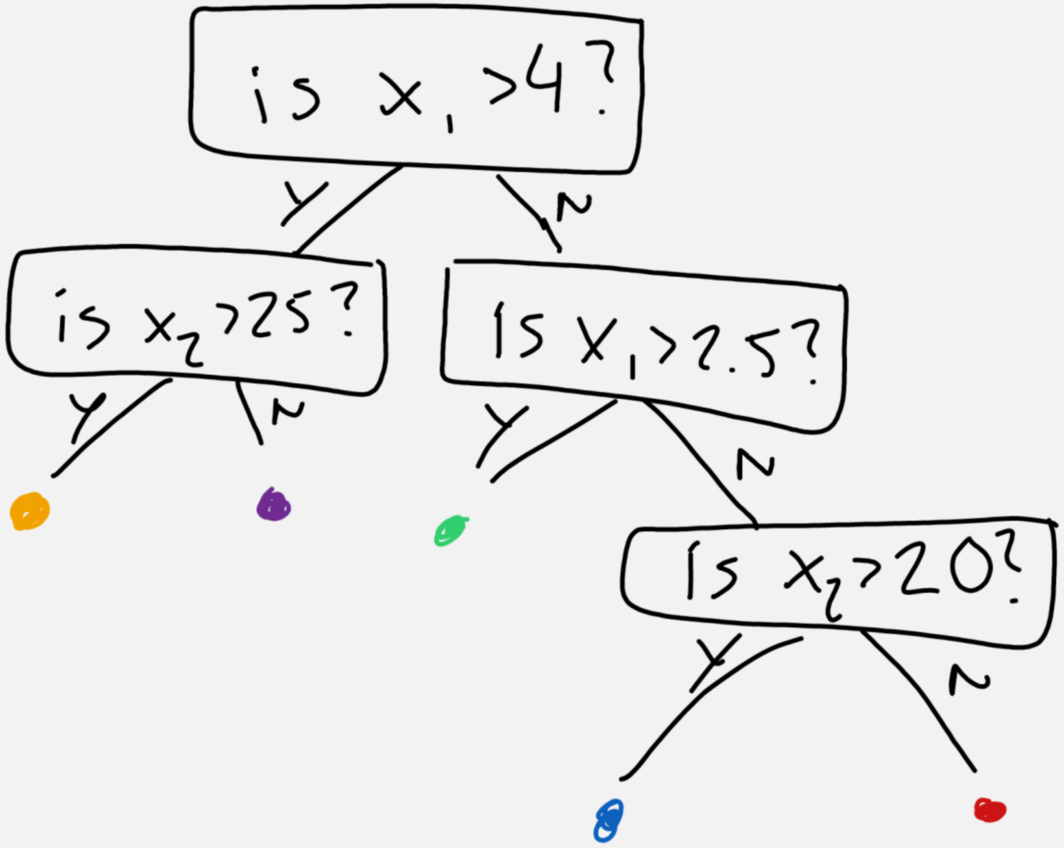
TREE-BASED MODELS

- Decision trees
- Random forests
- Gradient boosted trees

TREE-BASED MODELS

- Decision tree





Step 1

Find the feature that is the <u>best predictor</u> of your data

Step 2

Partition instances of your train data according to that feature

Step 3

Repeat 1-2 recursively

Stop when

- When all instances of a leaf belong to the same class
- When there are no more ways to split. Let dominant class decide

A LOAN IN THE BANK

salary	savings	debt	class
−	+	+	✓
−	−	−	%
+	−	−	✓
−	−	+	%
+	−	+	✓
+	+	−	✓
+	−	−	✓
−	−	+	%
+	+	+	✓
−	−	−	%
−	+	−	✓
−	−	+	%

Find best predictor using Gini impurity index

$$\underbrace{G(D)}_{\text{dataset}} = 1 - \sum_j p_j^2$$

← prob. of belonging to a certain class

$$G_k = \sum_i \frac{n_i}{n} G(D_i)$$

if we split on feature k

sum over partition

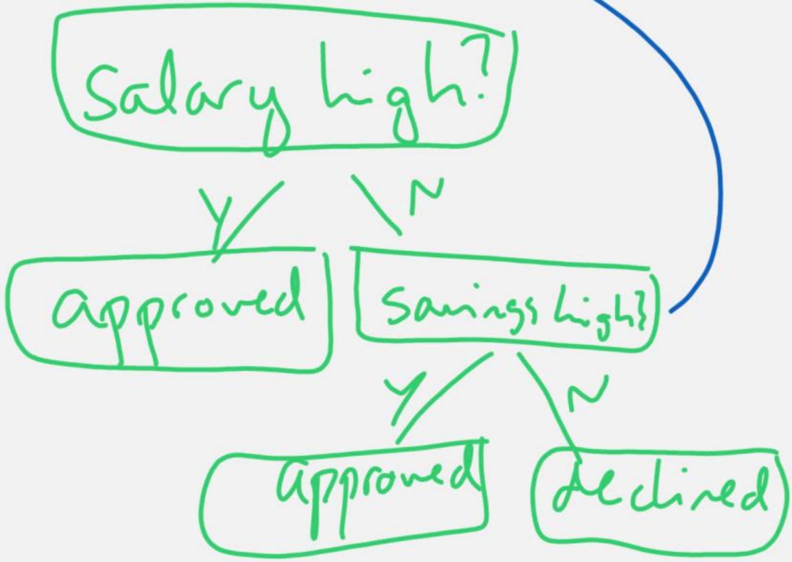
$$G_{salary} = \frac{7}{12}\underbrace{\left(1-\left(\frac{2}{7}\right)^2-\left(\frac{5}{7}\right)^2\right)}_{\text{low salary}} + \frac{5}{12}\underbrace{\left(1-\left(\frac{5}{5}\right)^2-\left(\frac{0}{5}\right)^2\right)}_{\text{high salary}} = 0.24$$

$$G_{salary} = 0.24$$

$$G_{savings} = 0.31$$

$$G_{debt} = 0.47$$

best predictor = lowest impurity index

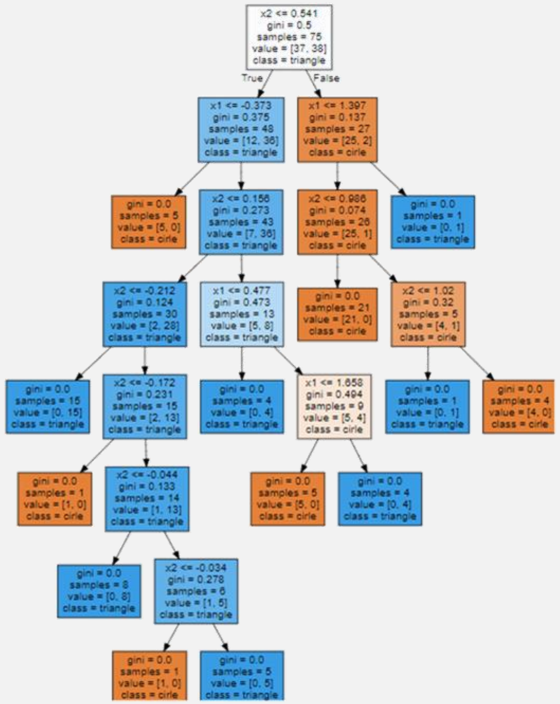


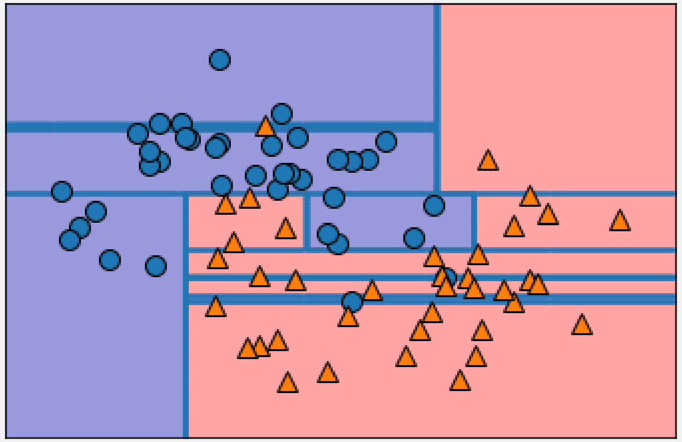
recalculate G's on this subset

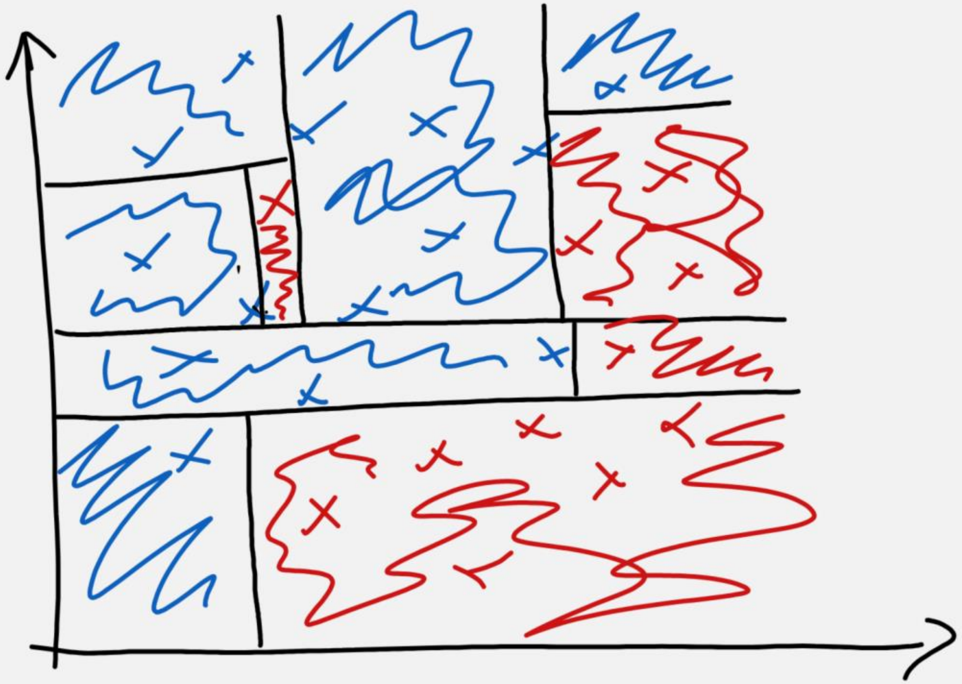
→ best predictor is savings

To learn a decision tree, the algorithm tries every possible yes/no question and goes for the best one - recursively

VISUALIZATION







Decision trees are very prone to OVERFITTING

PRE-PRUNING

Hyperparameters

max_depth: max no. of questions in a branch

max_leaf_nodes: max no. of leaves

min_samples_split: min no. of samples a node must contain for the model to split it

(criterion: default is Gini, others possible)

PROS	CONS
Fast!!	Not really accurate
Intepretable	Overfit, even with pre-pruning
Easy to visualize	
Invariant to data scaling	

ENSEMBLES OF DECISION TREES

method that combines multiple ML models to create more powerful models

- **Random forests** *(bagging)*

  a collection of slightly different decision trees that overfit differently

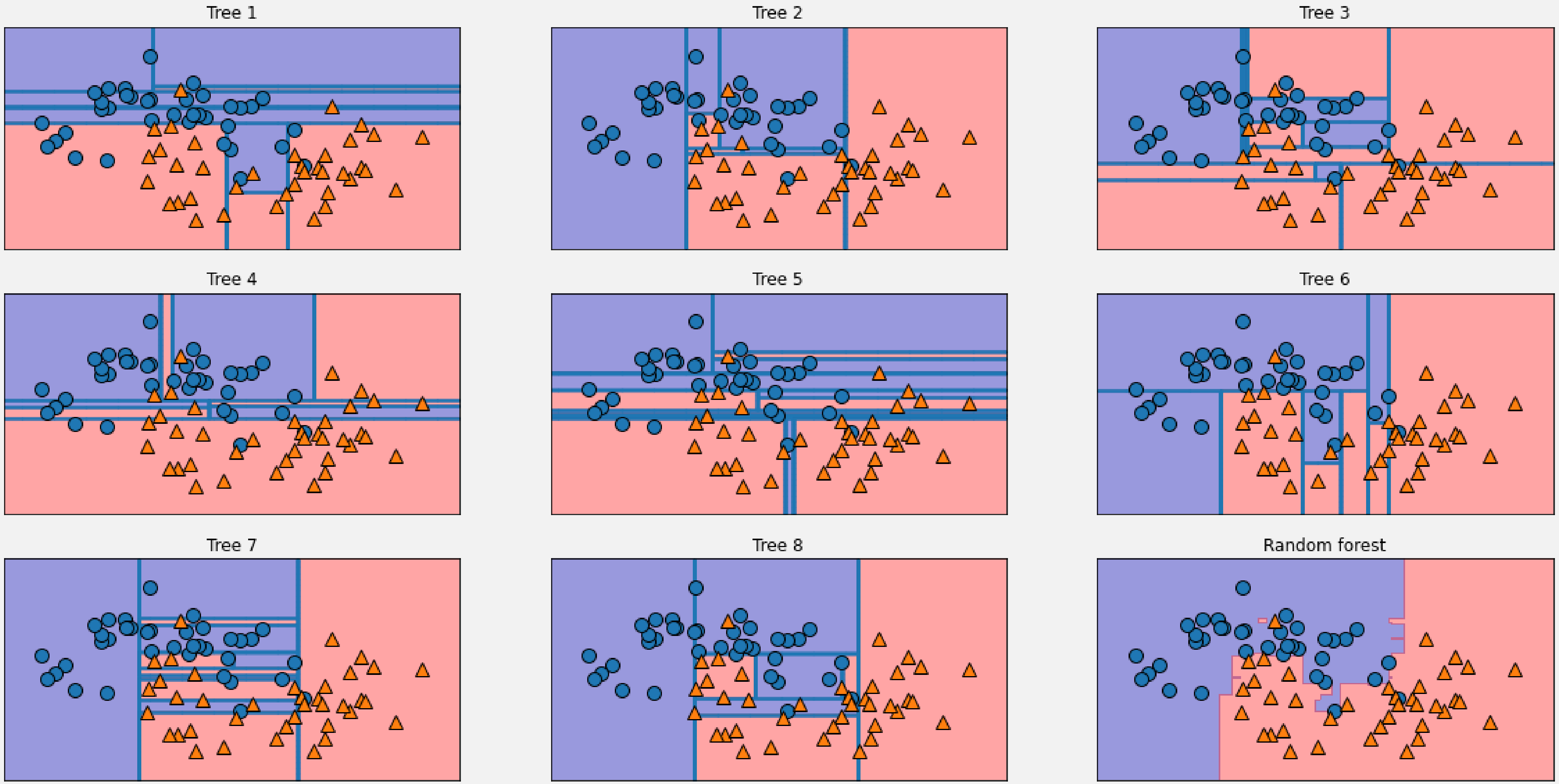
- **Gradient boosted decision trees** *(boosting)*

  a sequence of trees where each tree tries to correct the mistches of the previous one

TREE-BASED MODELS

- Decision trees
- Random forests
- Gradient boosted trees

RANDOM FORESTS



RANDOMIZATION I: BOOTSTRAPPING

features

samples

	f_1	f_2	f_3	f_4	f_5	f_6
x_1	45	5	21	45	15	1
x_2	87	2	12	44	64	2
x_3	24	8	15	43	36	3
x_4	67	7	17	44	87	2
x_5	13	5	12	44	65	3
x_6	87	4	16	42	34	1
x_7	89	7	13	42	2	2
x_8	68	3	14	43	54	3
x_9	35	6	11	41	63	2

RNG

Numbers	9
Min	1
Max	9

Go

7
9
4
8
7
2
3
3
8

A bootstrap dataset

	f_1	f_2	f_3	f_4	f_5	f_6
x_7	89	7	13	42	2	2
x_9	35	6	11	41	63	2
x_4	67	7	17	44	87	2
x_8	68	3	14	43	54	3
x_7	89	7	13	42	2	2
x_2	87	2	12	44	64	2
x_3	24	8	15	43	36	3
x_3	24	8	15	43	36	3
x_8	68	3	14	43	54	3

RANDOMIZATION I: BOOTSTRAPPING

Dataset for tree 1

	f_1	f_2	f_3	f_4	f_5	f_6
x_7	89	7	13	42	2	2
x_9	35	6	11	41	63	2
x_4	67	7	17	44	87	2
x_8	68	3	14	43	54	3
x_7	89	7	13	42	2	2
x_2	87	2	12	44	64	2
x_3	24	8	15	43	36	3
x_3	24	8	15	43	36	3
x_8	68	3	14	43	54	3

Dataset for tree 2

	f_1	f_2	f_3	f_4	f_5	f_6
x_6	87	4	16	42	34	1
x_8	68	3	14	43	54	3
x_2	87	2	12	44	64	2
x_2	87	2	12	44	64	2
x_3	24	8	15	43	36	3
x_7	89	7	13	42	2	2
x_4	67	7	17	44	87	2
x_2	87	2	12	44	64	2
x_8	68	3	14	43	54	3

Dataset for tree 3

	f_1	f_2	f_3	f_4	f_5	f_6
x_3	24	8	15	43	36	3
x_3	24	8	15	43	36	3
x_8	68	3	14	43	54	3
x_7	89	7	13	42	2	2
x_1	45	5	21	45	15	1
x_1	45	5	21	45	15	1
x_6	87	4	16	42	34	1
x_5	13	5	12	44	65	3
x_7	89	7	13	42	2	2

Each tree is based on different bootstrap datasets

RANDOMIZATION II: FEATURE SELECTION

Dataset for tree 1

	f_1	f_2	f_3	f_4	f_5	f_6
x_7	89	7	13	42	2	2
x_9	35	6	11	41	63	2
x_4	67	7	17	44	87	2
x_8	68	3	14	43	54	3
x_7	89	7	13	42	2	2
x_2	87	2	12	44	64	2
x_3	24	8	15	43	36	3
x_3	24	8	15	43	36	3
x_8	68	3	14	43	54	3

For each node, randomly select a subset of features and ask the best question involving one of these features

e.g. $f_2 > 6$?

RANDOMIZATION II: FEATURE SELECTION

Dataset for tree 1

	f_1	f_2	f_3	f_4	f_5	f_6
x_7	89	7	13	42	2	2
x_9	35	6	11	41	63	2
x_4	67	7	17	44	87	2
x_8	68	3	14	43	54	3
x_7	89	7	13	42	2	2
x_2	87	2	12	44	64	2
x_3	24	8	15	43	36	3
x_3	24	8	15	43	36	3
x_8	68	3	14	43	54	3

`max_features` controls how large this subset is

`max_features = n_features` ⇒ no randomness injected

`max_features = 1` forces the model to use a certain (random) feature

RANDOMIZATION II: FEATURE SELECTION

Dataset for tree 1

	f_1	f_2	f_3	f_4	f_5	f_6
x_7	89	7	13	42	2	2
x_9	35	6	11	41	63	2
x_4	67	7	17	44	87	2
x_8	68	3	14	43	54	3
x_7	89	7	13	42	2	2
x_2	87	2	12	44	64	2
x_3	24	8	15	43	36	3
x_3	24	8	15	43	36	3
x_8	68	3	14	43	54	3

A low value of `max_features`

$\Rightarrow$ very different, very deep trees

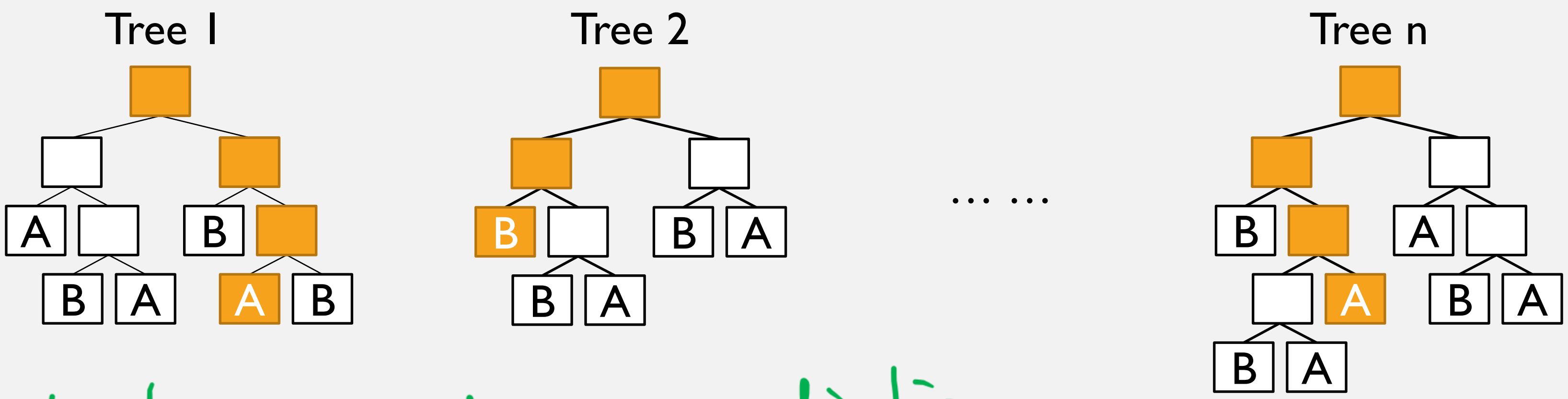
A high value of `max_features`

$\Rightarrow$ similar, shallow trees

A rule of thumb

$\Rightarrow \sqrt{\text{n\_features}}$

PREDICTIONS USING RANDOM FORESTS



Each tree makes a prediction

Regression: Average predicted values

Classification: "Soft voting": Average the probabilities of belonging to a certain class, and predict the highest probability one

PROS AND CONS OF RANDOM FORESTS

Pros

- Very powerful
- Work well with little parameter tuning

Cons

(- Different random states ⇒ diff. forest)

- Slower
- Difficult to interpret.

TREES VS. FORESTS

Jupyter Notebook **Decision Trees 2: Feature importance and ensembles of trees**

TREE-BASED MODELS

- Decision trees
- Random forests
- Gradient boosted trees

GRADIENT BOOSTED DECISION TREES

OR GRADIENT BOOSTED REGRESSION TREES *OR* GRADIENT BOOSTING MACHINES

→ Build a sequence of trees where each tree tries to correct the mistakes of the previous

→ Use very shallow trees ("weak learner")

HYPERPARAMETERS

n_estimators

how many trees in sequence?

max_depth

how deep each tree should be

learning_rate

how strongly does each tree depend on previous (~0.1-0.3 works well)

HOW DOES IT WORK?

- First tree

Fit a tree $T_1(x)$ to data x

- Second tree

target prediction

Calculate residuals: $r_2 = y - T_1(x)$

Fit a tree $t_2(r_2)$ on residuals:

Then $T_2(x) = T_1(x) + \eta\, t_2(r_2)$

learning rate

- *n*th tree

$T_n(x) = T_{n-1}(x) + \eta\, t_n(r_n)$

CODING BOOSTED TREES

Jupyter Notebook **Decision Trees 2: Feature importance and ensembles of trees**
Jupyter Notebook **Decision Trees 3: Gradient boosted trees for regression**

PROS AND CONS OF GRADIENT BOOSTED DECISION TREES

Pros

One of the most powerful models out there

Cons

Requires careful parameter tuning

WHEN TO USE WHAT

Tree	Forest	Boosted tree
when VISUALIZATION	ROBUSTNESS	ACCURACY is important
fast	slowest	slower

- Explain how tree-based models work
- Make informed decisions about when each model is appropriate
- Implement tree-based models for classification and regression problems


WHERE DOES A DATA SCIENTIST CAMP?
IN A RANDOM FOREST