

# Machine Learning for Networks: Trees and ensembles

**Andrea Araldo**

*May 21, 2025*

---

	ML task	Linear Regression	Logistic Regression	Tree-based learning	Neural Networks	$k$ -Nearest Neighbors
Supervised	Regression Classification	x	x	x	x x	
Unsupervised	Clustering Dimensionality reduction Anomaly detection			x	x x x	x  x
	Recommender Systems				x	

### Decision tree

- Training: CART algorithm
- Entropy, Gini impurity, Information Gain
- High variance

### Ensemble learning

- Bagging
- Random Forest
- Extra trees

### Interpretability

## Microsoft Kinect (Xbox)[[KS13](#)]



# Case Study: Activity Classification

4 / 46

## Problem:

- Authorities want to predict impact of a new infrastructure or service is introduced- ex. autonomous vehicles
- ... before making the investment
- We need *models* to predict user's behavior
- Traditional surveys are costly and have low penetration

## Idea:

- Automatically sense user behavior via a smartphone app ([Future Mobility Sensing - FMS](#))

## In [[KPZ<sup>+</sup>14](#)]:

- Data collected via FMS
- Goal: classify each activity in *Home* or *Work*.
- Sample = Activity
- Features:

Age of the individual, time of day, duration, phone use, girometer data, accelerometer, GPS, etc.



Massachusetts Institute of Technology



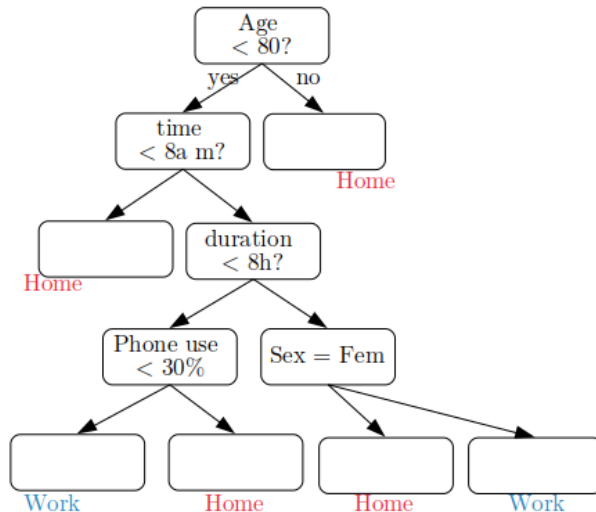
## Section 1

### **Decision tree**

# Decision Tree: Introduction

6 / 46

- Set of *splitting rules* organized as a tree.
- A class associated to each terminal node.
- Each sample traverses the tree up to a terminal node.
- Where does the following sample fall?  
age 50, time 10am, duration 7h, phone use 2%, sex Male



# Nodes and regions

7 / 46

N1



Time



N1

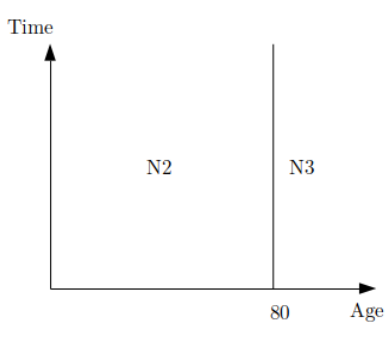
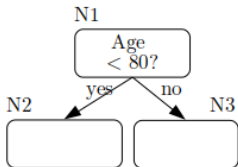


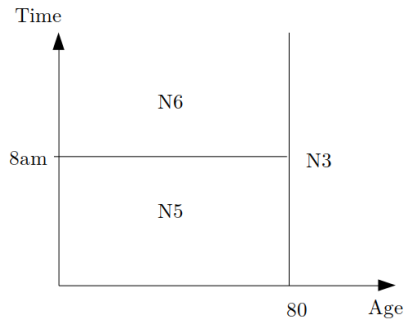
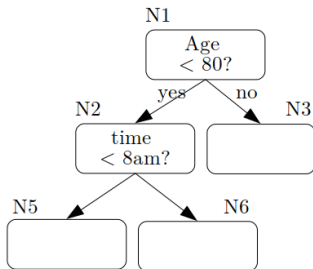
Age



# Nodes and regions

8 / 46





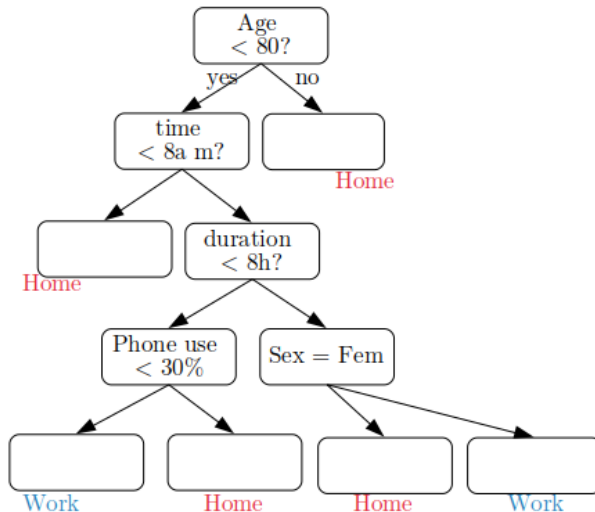
Question: Are random trees linear classifiers?

# Training a decision tree

10 / 46

Decide:

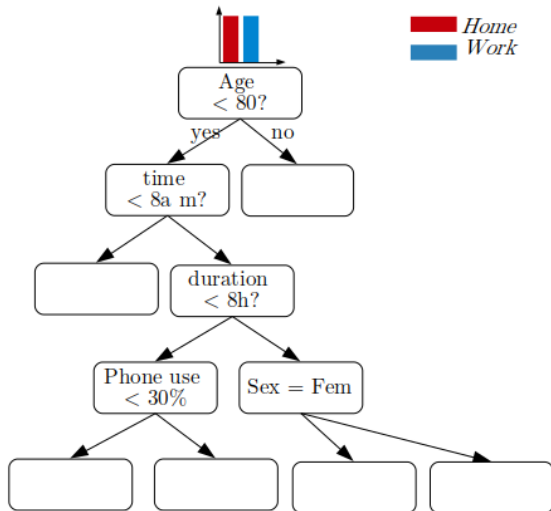
- The label of each node
- the splitting rules at each node



# Assign labels to nodes

11 / 46

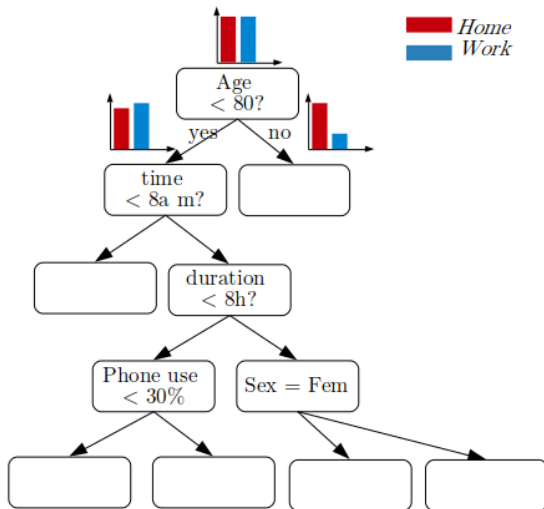
- How do we assign classes to the nodes?
  - Feed the tree with all the training samples



# Assign labels to nodes

11 / 46

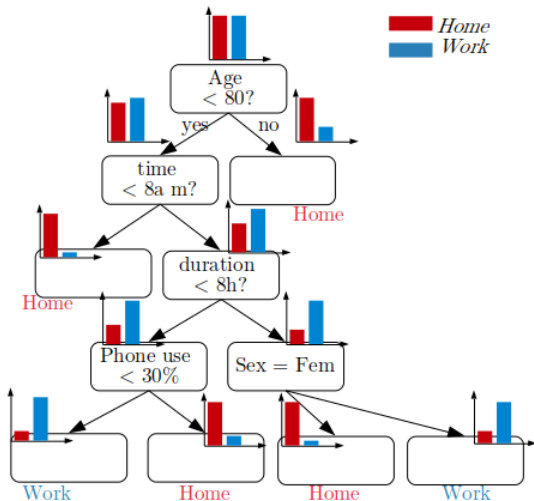
- How do we assign classes to the nodes?
  - Feed the tree with all the training samples
  - Observe the samples falling in each node and compute the histogram



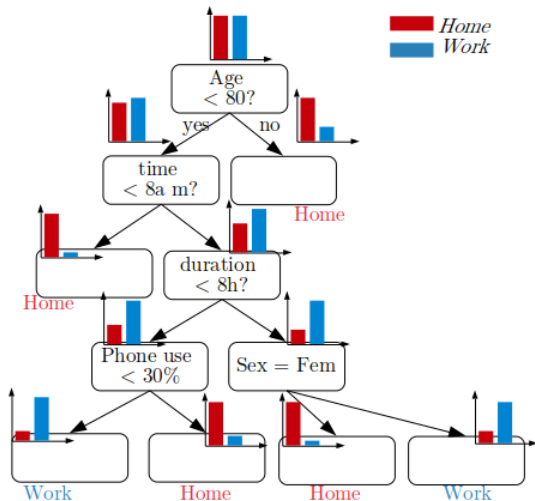
# Assign labels to nodes

11 / 46

- How do we assign classes to the nodes?
  - Feed the tree with all the training samples
  - Observe the samples falling in each node and compute the histogram



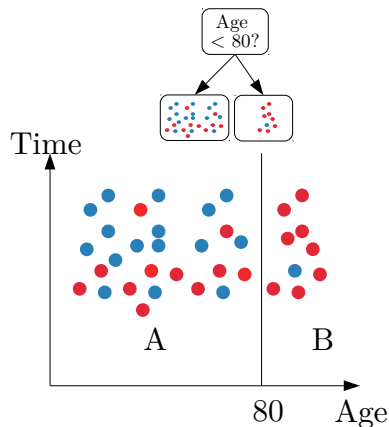
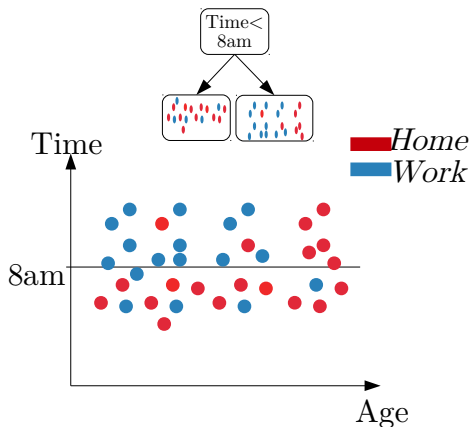
- How do we assign classes to the nodes?
  - Feed the tree with all the training samples
  - Observe the samples falling in each node and compute the histogram
  - Associate a node to its *prevalent class*.



## Decide splitting rules

12 / 46

- We want a tree with just two terminal nodes. How do we choose the splitting rule?
- Which splitting rule is better?



Two metrics for impurity: *entropy* or *Gini impurity* index.



# Entropy of a set of samples

13 / 46

Given a set of samples  $S$

- $p_b, p_r$ : ratio of samples in  $S$  that are blue / red.
- The entropy of  $S$  is defined as:

$$H(S) \triangleq p_b \cdot \log_2 \frac{1}{p_b} + p_r \cdot \log_2 \frac{1}{p_r}$$

Compute  $H(A), H(B)$

Other examples:

$$H(D) = 0.33 \cdot \log_2(1/0.33) + 0.67 \log_2(1/0.67) = 0.91$$

$$H(E) = 0.17 \cdot \log_2(1/0.17) + 0.83 \log_2(1/0.83) = 0.65$$

The entropy measures the uncertainty about the class of each sample.  
We can extend it to  $> 2$  classes.



- Average entropy of the subsets

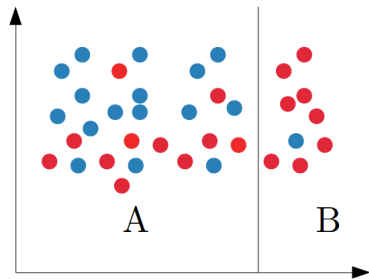
$$H(A,B) \triangleq p_A \cdot H(A) + p_B \cdot H(B)$$

where  $p_A, p_B$  are the ratios of samples in A and B.

- **Theorem:** partitioning a set always decreases the entropy (i.e. uncertainty):

$$H(A \cup B) \geq H(A,B)$$

- Knowing whether a sample is in A or B gives us additional info about its class.



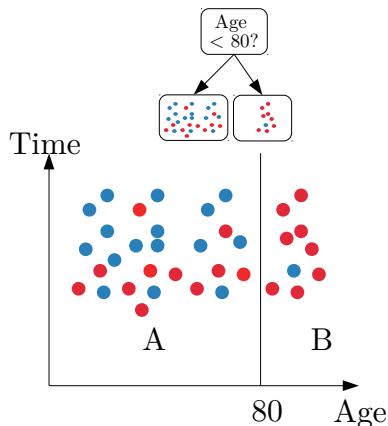
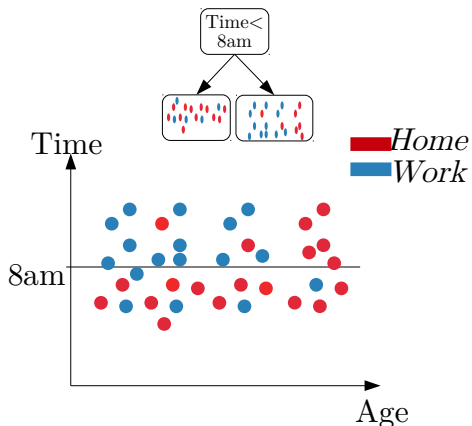
Information gain of a split:  
 $G \triangleq H(A \cup B) - H(A,B)$

$$\implies G \geq 0$$

# Information Gain of a Splitting Rule

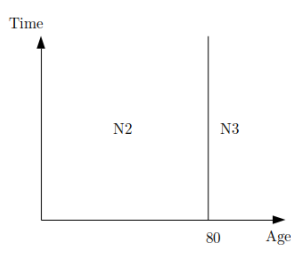
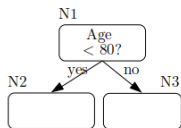
15 / 46

- A splitting rule partitions the samples
- We can associate a  $G$  to each splitting rule:  $G(\text{Age} < 80)$ ,  $G(\text{Time} < 8\text{am})$
- We prefer the rule that ?.....?



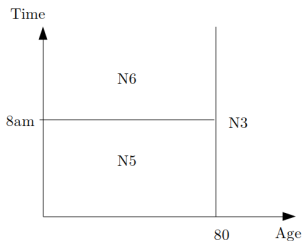
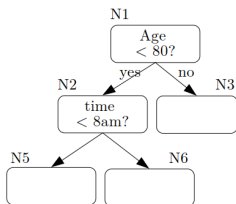
# Information Gain of further Splitting Rules

16 / 46



- We select  $\text{Age} < 80$  as our first splitting rule.
- After that, the information gain of  $\text{Time} < 8\text{am}$  is

$$G = H(N2, N3) - H(\underbrace{N5, N6}_{\text{partition of } N2}, N3)$$

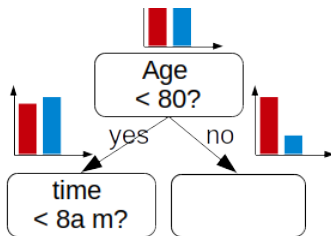


# CART algorithm

17 / 46

We will *grow* (=train) our tree *on* our training set.

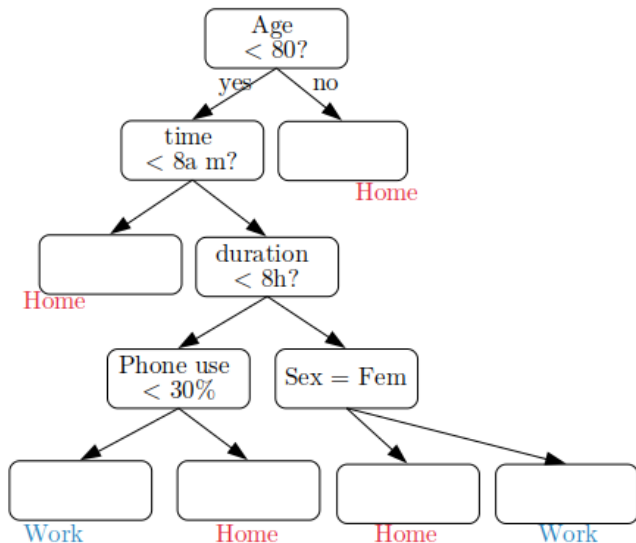
1. Decide the split points per each feature.
  2. Compute a  $G$  per each
  3. Select the (feature < split) with the highest  $G$
  4. Split the set of samples in two subsets (nodes)
  5. Repeat the same process on each node
1. Age: 30, 40, 50, 60, 80  
Duration: 2,4,6,8 ...
  2.  $G(\text{Age} < 30) = 0.3$ ,  
 $G(\text{Age} < 40) = 0.32$ ,  
...,  
 $G(\text{Duration} < 2) = 0.4$ ,  
 $G(\text{Duration} < 4) = 0.41$ , ...



CART = Classification and Regression Tree.

# CART Algorithm: Result

18 / 46



- CART is a *greedy algorithm*:  
At every node it selects the “best” split, i.e., the one that maximizes the information gain
- Nothing ensures this is optimal: we may grow better trees by taking worse splits

Ex. If you want to make a lot of money.

- When you are 18, should you greedily work and get money?
- Or is it better to study first?

- Entropy of a set  $S$  with  $K$  classes:

$$H(S) = \sum_{k=1}^K p_k \cdot \log_2 \frac{1}{p_k}$$

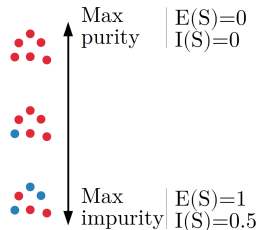
- Gini impurity of a set  $S$ :

$$I(S) = \sum_{k=1}^K p_k \cdot (1 - p_k)$$

Probability that, taking any two samples from  $S$ , they are of the same class

- Conceptually similar to entropy.
- Impurity of a partition  $A, B$ :  
 $I(A, B) = p_A \cdot I(A) + p_B \cdot I(B)$ .
- Impurity of a split.

With two classes:



In general

$$0 \leq E(S) \leq \log_2 K \text{ and}$$

$$0 \leq I(S) \leq \frac{1}{K}.$$



- Classification Error Rate of a set  $S$ : rate of error when classifying an element of  $S$  with its prevalent class

$$CER(S) = 1 - \max_k p_{k,S}$$

- CER of a partition  $A, B$ :  $CER(A, B) = p_A \cdot CER(A) + p_B \cdot CER(B)$ .
- CER of a split.
- Less used than entropy and Gini impurity.

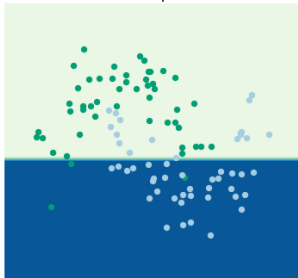
# Depth of a tree

22 / 46

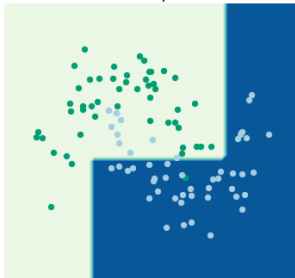
Image from *Pivotal, Pivotal Engineering Journal*:

<http://engineering.pivotal.io/post/interpreting-decision-trees-and-random-forests/>

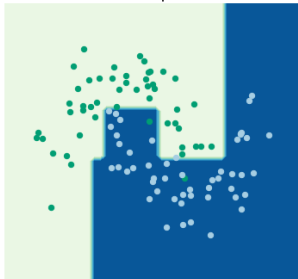
Max Depth: 1



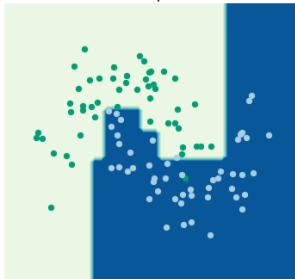
Max Depth: 2



Max Depth: 5



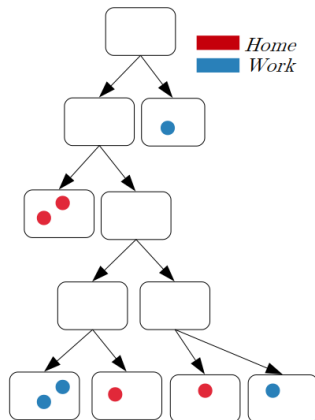
Max Depth: 10



## Depth of a tree

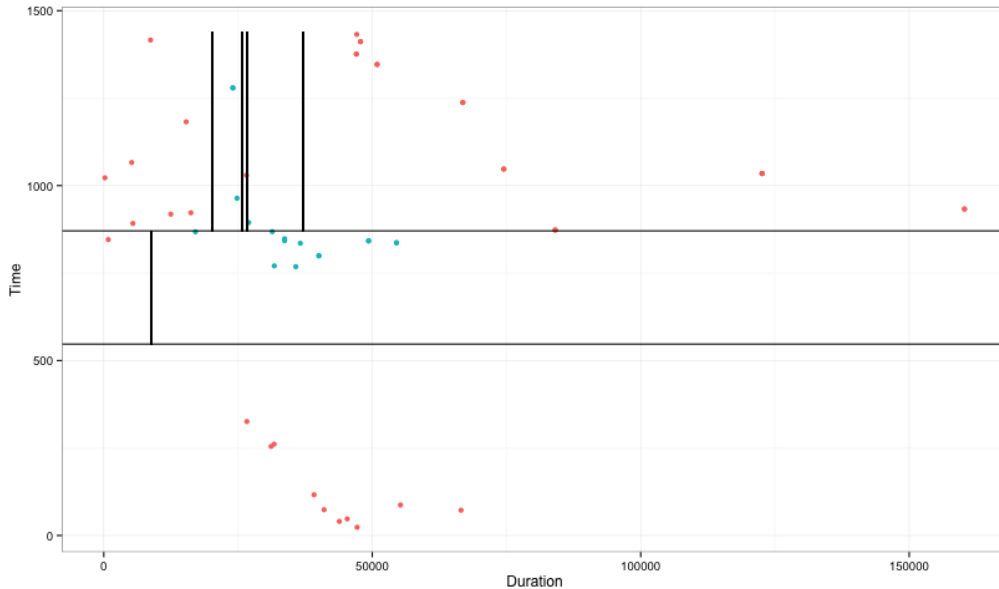
23 / 46

- CART stops when all terminal nodes are “pure”
- We say the tree is *fully grown*.



# Overfitting

24 / 46



- *Pre-pruning*: We stop creating children at a node if
  - Too few samples at that node
  - Splitting does not give a high information gain.
  - The maximum number of nodes is reached
- *Post-pruning*: Develop a fully grown tree, then trim the nodes in a bottom-up manner
  - Use the validation data set to compute the Classification Error Rate.
  - Stop pruning if it reduces significantly.

These are all hyperparameters.

## Section 2

# **Ensemble learning**

From lesson 02.regression:

## Variance

A model suffers **high variance** if, by perturbing a bit the training dataset, the model changes completely.

Suppose  $\tilde{\mathbf{X}}, \tilde{\mathbf{y}}$  is a slightly perturbed version of  $\mathbf{X}, \mathbf{y}$ . If a model has high variance:

$$\tilde{\theta}^* = \arg \min_{\theta} J(\theta, \tilde{\mathbf{X}}, \tilde{\mathbf{y}})$$

completely  $\neq$

$$\theta^* = \arg \min_{\theta} J(\theta, \mathbf{X}, \mathbf{y})$$

## Example:

In our activity-classification use case, suppose you have training set  $\mathbf{X}, \mathbf{y}$ .

- CART decides the 1st splitting rule.
- Suppose  $G(\text{Age} < 80) = 0.1002$  and  $G(\text{time} < 8am) = 0.1000$ .
- Which one does CART select?

If you had instead another training set  $\tilde{\mathbf{X}}, \tilde{\mathbf{y}}$  in which there is only another additional sample and

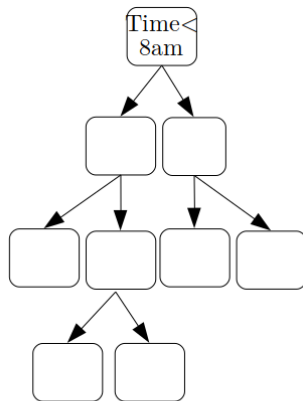
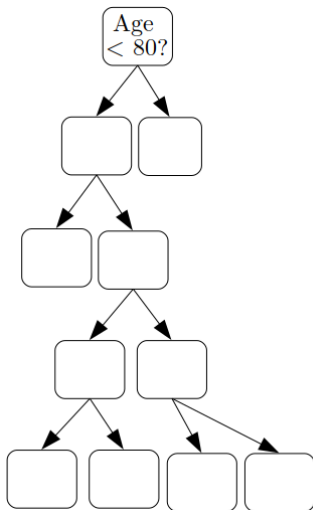
$G(\text{Age} < 80) = 0.1000$  and  
 $G(\text{time} < 8am) = 0.1001$ .

- Which would be the 1st splitting rule selected by CART?

# The origin of variance in trees

28 / 46

One sample only has  
changed completely our  
tree!



Would pruning solve this variance-problem?

aa: No, variance issue arises from the 1st splitting rule!



Why it works:

- If the President needs to face a pandemic, should he/she
  - Call the best expert in the world or
  - Call 10 good experts?
- A super-good expert can still make errors or be biased
- Plurality smooth errors and biases.



The following techniques have been proven empirically [BK11] to reduce the variance:

- Bagging
- Random Forests
- Extra Trees
- Boosting

Training on a set  $(\mathbf{X}, \mathbf{y})$

- *Bootstrap sampling*:  
Obtain  $K$  subsets  
 $(\mathbf{X}, \mathbf{y})_k \subseteq (\mathbf{X}, \mathbf{y})$ 
  - Set size can change
- Grow a tree  $h_k(\cdot)$  on each  
 $(\mathbf{X}, \mathbf{y})_k$

Prediction of a new  $\mathbf{x}$

- Obtain  $h_k(\mathbf{x})$  from each  
tree  $h_k(\cdot)$
- Use *majority voting*

How many trees?

- Typically several hundreds.
- **Interpretability: feature importance.**

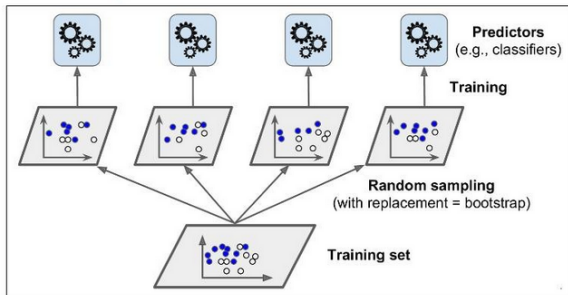


Figure 7-4. Bagging and pasting involves training several predictors on different random samples of the training set

From [Ger19]

Detail not important: If sampling is with replacement (the same sample can be taken multiple times when growing one tree), we talk about *bagging*. If instead sampling is without replacement, we talk about *Pasting*

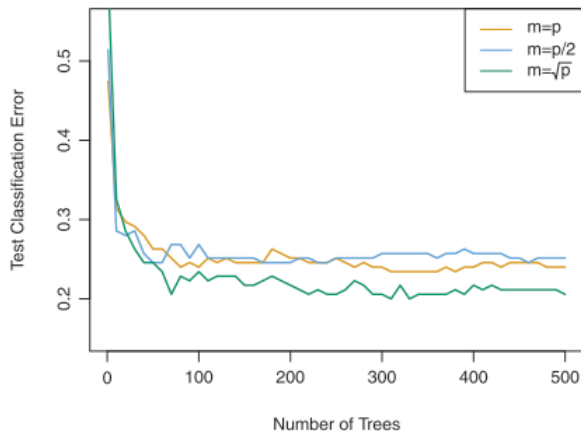
Modified version of bagging:

- Every time we decide the split rule, the candidate variables under considerations are a random subset.
- The inventors suggest  $\sqrt{N}$  candidates ( $N$  = num of features). However, it is a parameter to tune.

Advantages over Bagging

- **Less variance**, i.e., less overfitting
  - With bagging all trees tend to be the same at their top nodes  
⇒ Trees are correlated.
  - Random forests create more de-correlated trees.  
⇒ Predictions are more diverse.
  - (To face a pandemic, it is not worth having experts all thinking the same)
- Computation **efficiency**.

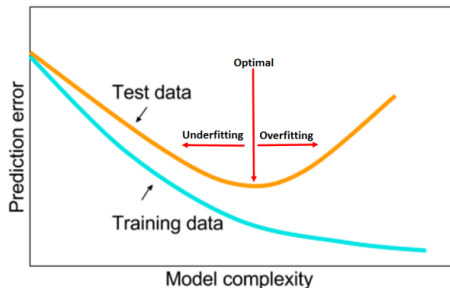
General lesson in ML: **stochasticity is your friend** to get (i) stable predictors, (ii) efficiency and (iii) interpretability.



**FIGURE 8.10.** Results from random forests for the 15-class gene expression data set with  $p = 500$  predictors. The test error is displayed as a function of the number of trees. Each colored line corresponds to a different value of  $m$ , the number of predictors available for splitting at each interior tree node. Random forests ( $m < p$ ) lead to a slight improvement over bagging ( $m = p$ ). A single classification tree has an error rate of 45.7%.

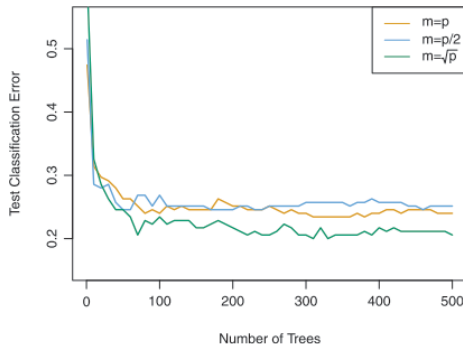
# No increase in complexity

34 / 46



From [Smi18]

Adding neurons in a NN increases complexity



From [JWHT13]

Adding a predictor in an ensemble does not increase complexity  
 $\Rightarrow$  We don't increase the risk of overfitting

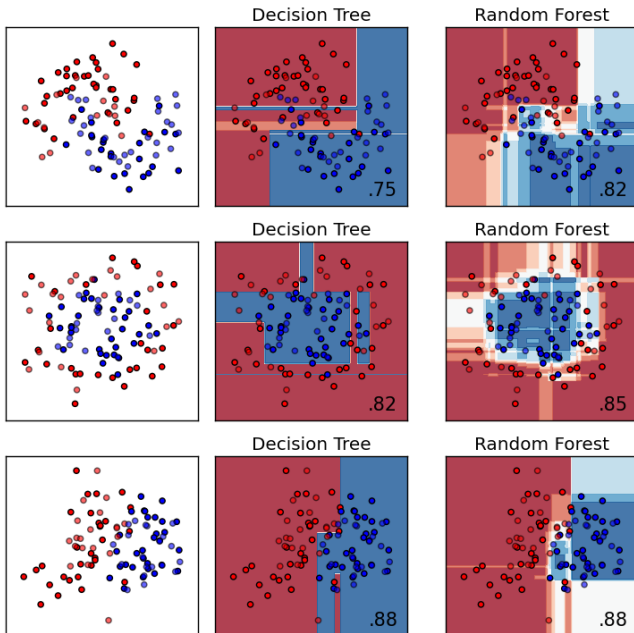
# Classification boundaries

35 / 46

What does the shading represent?

**Interpretability:  
Confidence in the  
prediction**

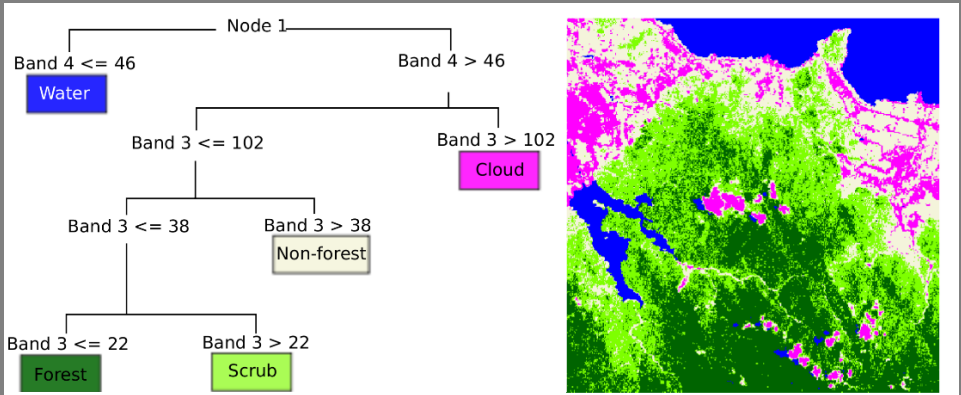
Picture from [Martin Thoma](#)



# Tree Classifier Application: Image Segmentation

36 / 46

- Predictors: red (band 3) and near-infrared (band 4) bands from the Landsat Enhanced Thematic Mapper Plus satellite sensor.



Ref: Horning N., *Random Forests : An algorithm for image classification and generation of continuous fields data sets*



- Single Tree
- Bagging
  - Sampling data points
- Random forests
  - ++ Random features
- Extra-trees (extremely randomized trees)
  - ++ Random splitting rule (instead of the best, as in CART)
  - An option allows deciding whether to use all the dataset to build each tree or a random subset

- Bagging, Random Forests and extra trees:
  - Each tree is independent
- Boosting:
  - Each tree tries to “correct” the errors of the others
  - Ex.: XGBoost

Applicable with Bagging Classification Trees, Random Forests and Extra-Forests.

Importance of a feature:<sup>1</sup>

- In each tree
  - Check all the rules which use that feature
  - Check the gain (either information gain or decrease in Gini impurity index)
  - Sum all these gains
- Score: Average the per-tree gain over all the trees (weighting with the samples-per-tree)

Scale all the feature scores so that they sum up to 1.

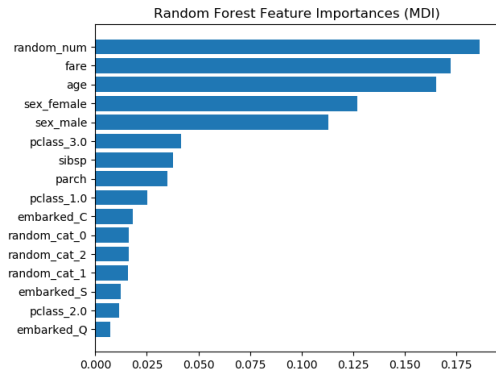
---

<sup>1</sup>See pagg.198-199 of [Ger19] and pagg.333-334 of [JWHT13]



# Interpretability: Feature Importance

40 / 46



From [Scikit Learn docs](#)  
and from [\[Ger19\]](#)

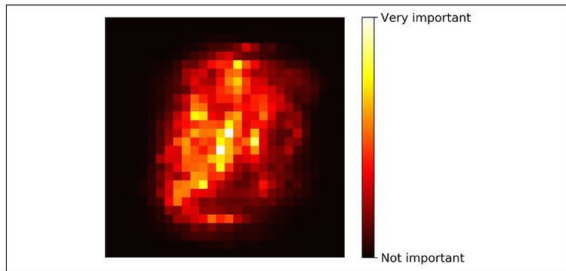


Figure 7-6. MNIST pixel importance (according to a Random Forest classifier)

## Scaling needed?

41 / 46

aa: No, as every time we decide a splitting rule, the threshold is automatically chosen in the range of the feature considered.

- Predicted value = avg of samples of a tree node
- Training = Find splits that minimize the Mean Squared Error (MSE)

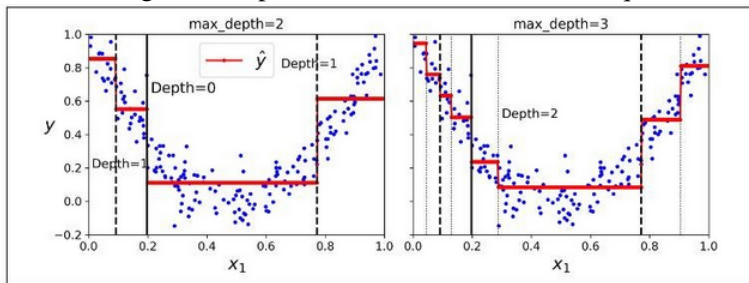


Figure 6-5. Predictions of two Decision Tree regression models

From [Ger19].

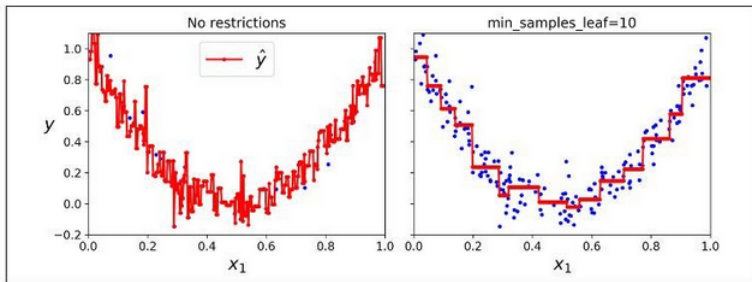


Figure 6-6. Regularizing a Decision Tree regressor

### Decision tree

- Training: CART algorithm
- Entropy, Gini impurity, Information Gain
- High variance

### Ensemble learning

- Bagging
- Random Forest
- Extra trees

### Interpretability



Ian H. Witten, Eibe Frank, Mark A. Hall. Data Mining: Practical Machine Learning Tools and Techniques 2nd Edition Elsevier - Section 6.1

- [BK11] Eric Bauer and Ron Kohavi, *An Empirical Comparison of Voting Classification Algorithms : Bagging , Boosting , and Variants*, Machine Learning **38** (2011), no. 1998, 1–38.
- [Ger19] Aurelien Geron, *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow*, O'Reilly, 2019.
- [JWHT13] Gareth James, Daniela Witten, Trevor Hastie, and Robert Tibshirani, *An introduction to Statistical Learning*, vol. 7, 2013.
- [KPZ<sup>+</sup>14] Youngsung Kim, Francisco C. Pereira, Fang Zhao, Ajinkya Ghorpade, P. Christopher Zegras, and Moshe Ben-Akiva, *Activity recognition for a smartphone based travel survey based on cross-user history data*, International Conference on Pattern Recognition (2014).
- [KS13] Pushmeet Kohli and Jamie Shotton, *Key developments in human pose estimation for kinect*, pp. 63–70, Springer London, London, 2013.

- [Smi18] Leslie N. Smith, *A disciplined approach to neural network hyper-parameters: Part 1 – learning rate, batch size, momentum, and weight decay*, Tech. report, US Naval Research Laboratory, 2018.