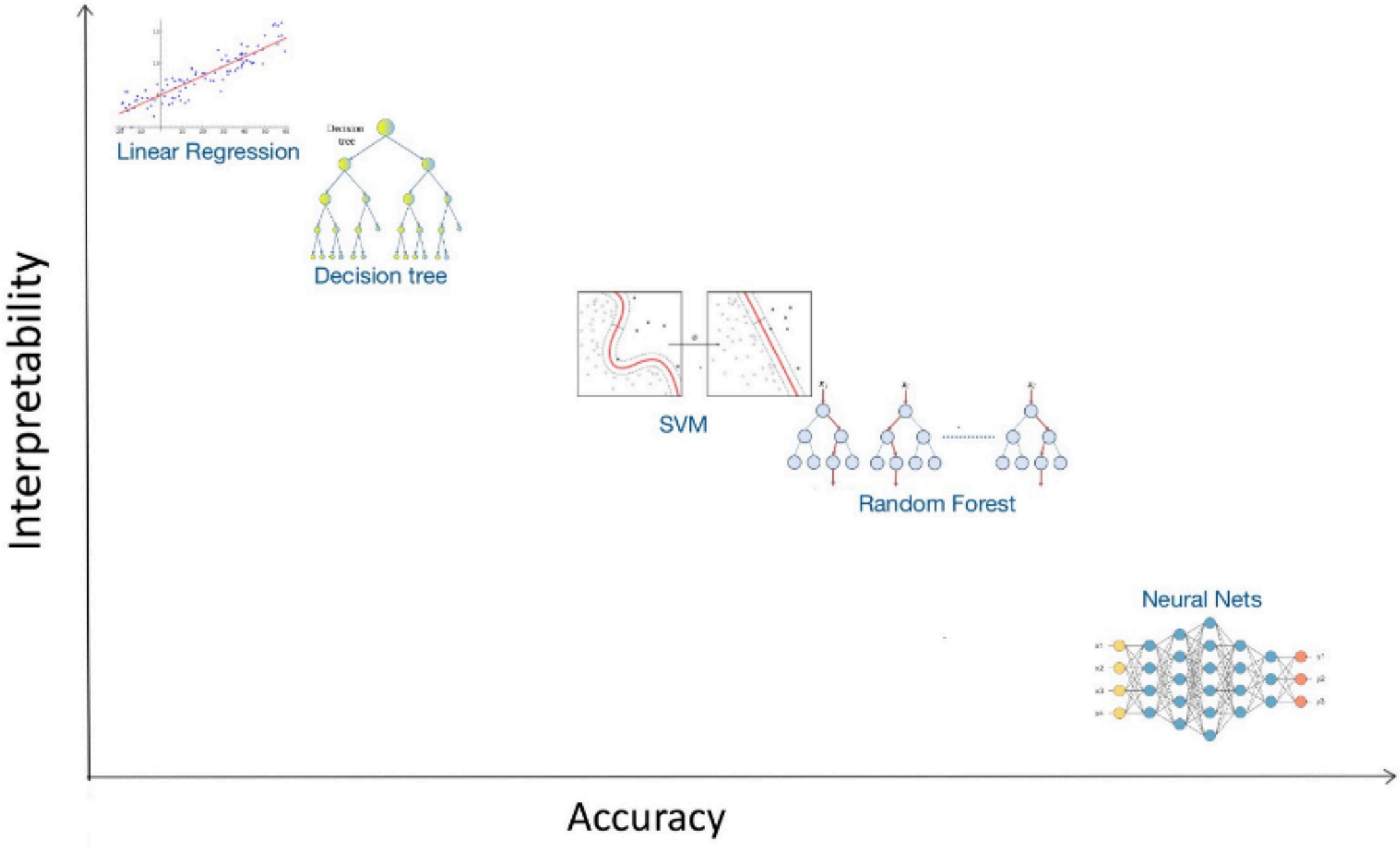
Decision Trees

Sewoong Oh

CSE/STAT 416
University of Washington


Interpretability
Linear Regression
Decision tree
Decision tree
SVM
Random Forest
Neural Nets
Accuracy

Decision trees:
An interpretable predictor

Example: predicting potential loan defaults

- Data: discrete for now
- Goal: Given a new loan application, predict the whether he will default on the loan

$$\text{predictor } f(x) \approx y$$

Credit	Term	Income	y
excellent	3 yrs	high	safe
fair	5 yrs	low	risky
fair	3 yrs	high	safe
poor	5 yrs	high	risky
excellent	3 yrs	low	risky
fair	5 yrs	low	safe
poor	3 yrs	high	risky
poor	5 yrs	low	safe
fair	3 yrs	high	safe

- **Learning**: fitting a model to data
- **Inference**: making prediction using a learned model

Did I pay previous loans on time?

Example: excellent, good, or fair

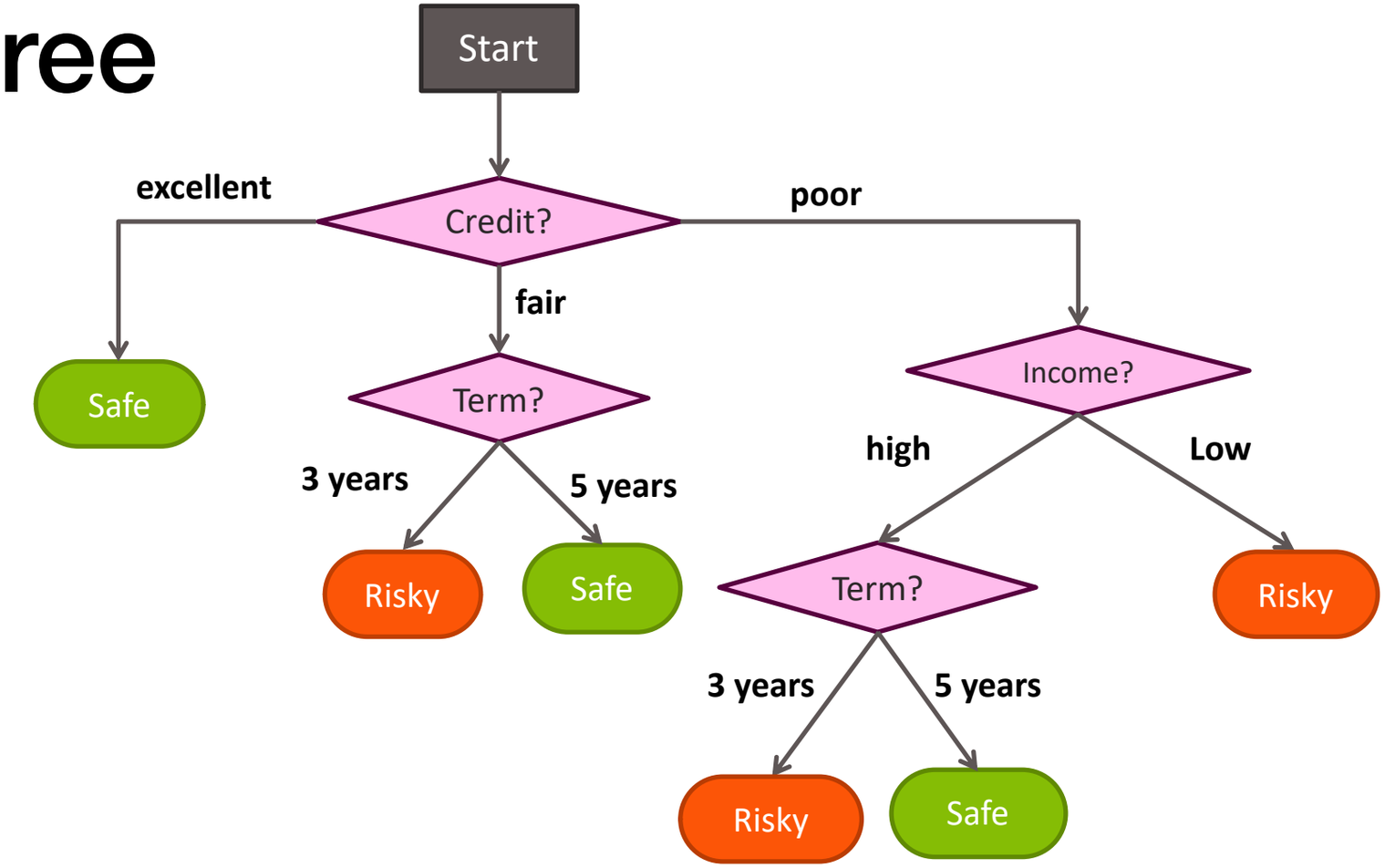
How soon do I need to pay the loan?

Example: 3 years, 5 years,…

What's my income?

Example:
>$80K per year

Decision tree



- Each **internal** node tests a feature ***x[i]***
- Each **branch** assigns a feature value ***x[i]=fair***
  (or a subset of feature values ***{fair,poor}***)
- Each **leaf** node assigns a class ***y***
- To **predict**, traverse the tree from root to a leaf

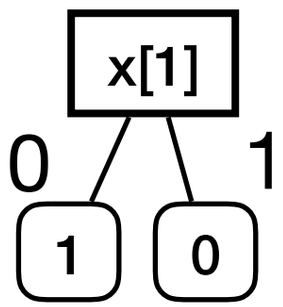
  f(poor credit, high income, 3 year) = ?
- Decision trees are naturally **human interpretable**!

What functions can be represented?

- For discrete input and output data, any function of the input can be represented as a decision tree
- However, in general, it could require exponentially many nodes to represent a arbitrary function (exponential in the dimension of the input)
- For example, a function that is sensitive to a small change in the input

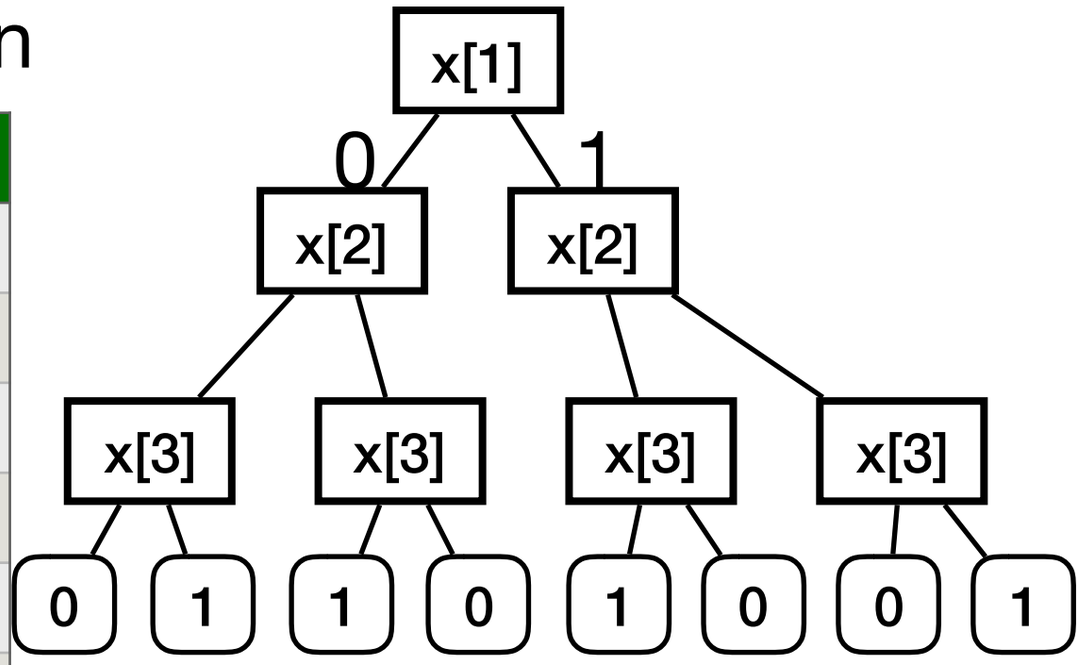
Simple function

x[1]	x[2]	x[3]	Y
0	0	0	**1**
0	0	1	**1**
0	1	0	**1**
1	0	0	**0**
0	1	1	**1**
1	0	1	**0**
1	1	0	**0**
1	1	1	**0**



Parity function

x[1]	x[2]	x[3]	Y
0	0	0	**0**
0	0	1	**1**
0	1	0	**1**
1	0	0	**1**
0	1	1	**0**
1	0	1	**0**
1	1	0	**0**
1	1	1	**1**

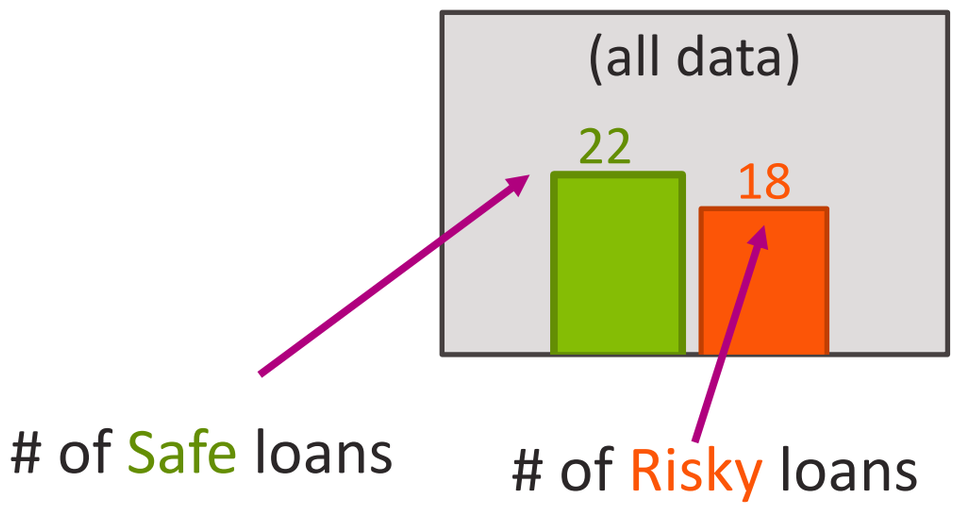


Which tree is better?

- Trade off between **Accuracy** vs**. Simplicity**
  - Accuracy is measured by $\frac{\#\text{ of correct predictions}}{\#\text{ of examples}}$
  - Simplicity can be measured by depth, number of leaves, etc

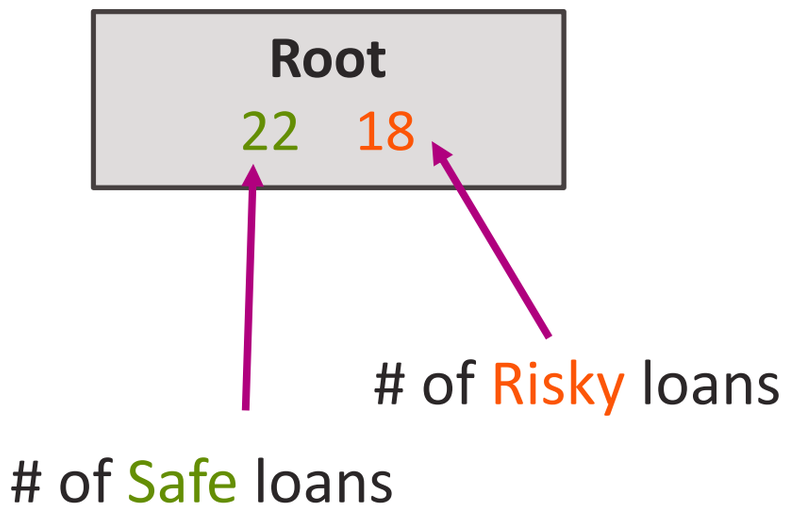
Training Data

Loan status: Safe Risky



Visualizing a decision tree and data

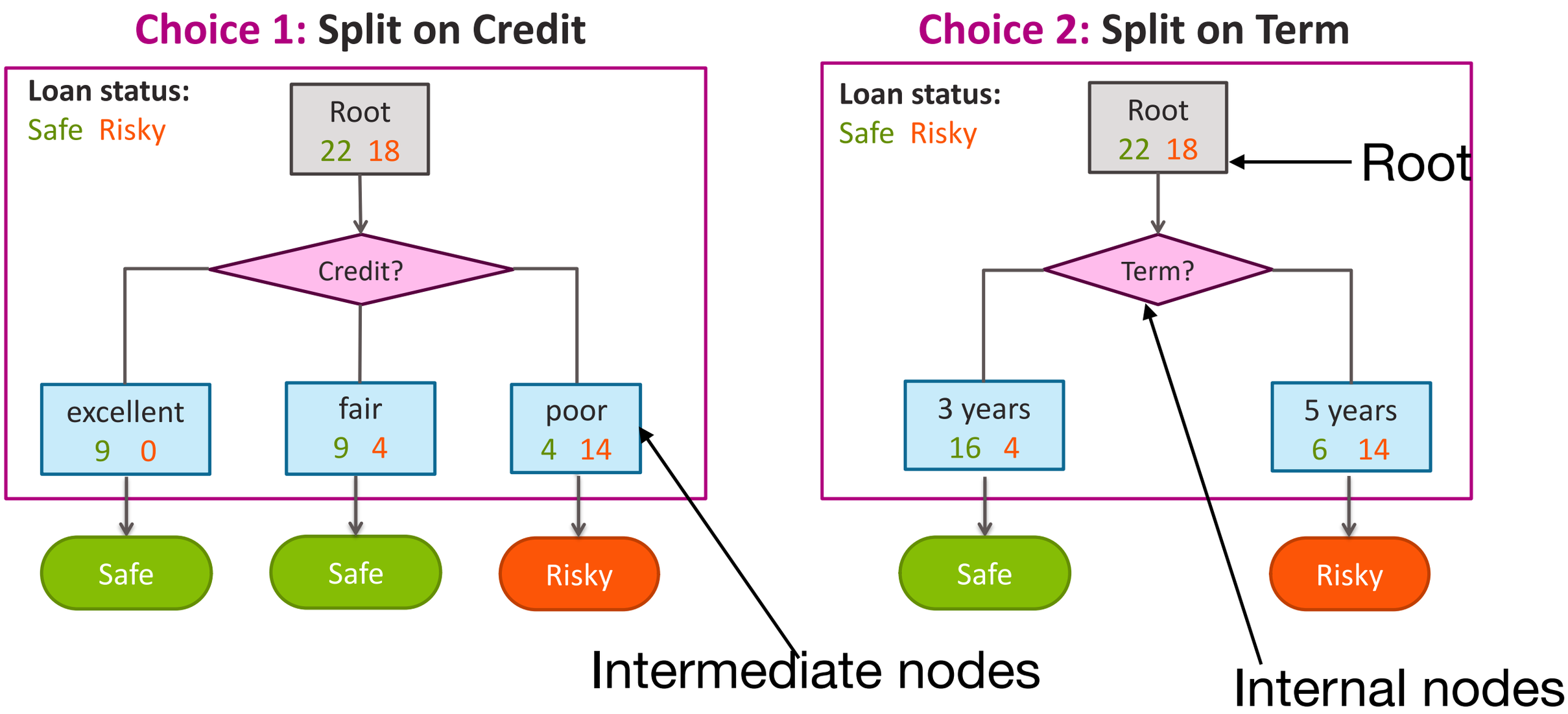
Loan status: Safe Risky



N = 40 examples

- If you have just the root node, for your decision tree, what should the decision be, and what is the accuracy and what is the error?

Decision stump: single level tree



- We grow the tree, by adding one more level of branching, and deciding which hypothesis (or feature) to test at the branch
- At the intermediate node, the **prediction** is determined by the majority rule
- In a greedy approach, we choose the hypothesis that gives better accuracy at the intermediate nodes; credit: 32/40=0.8, term: 30/40=0.75

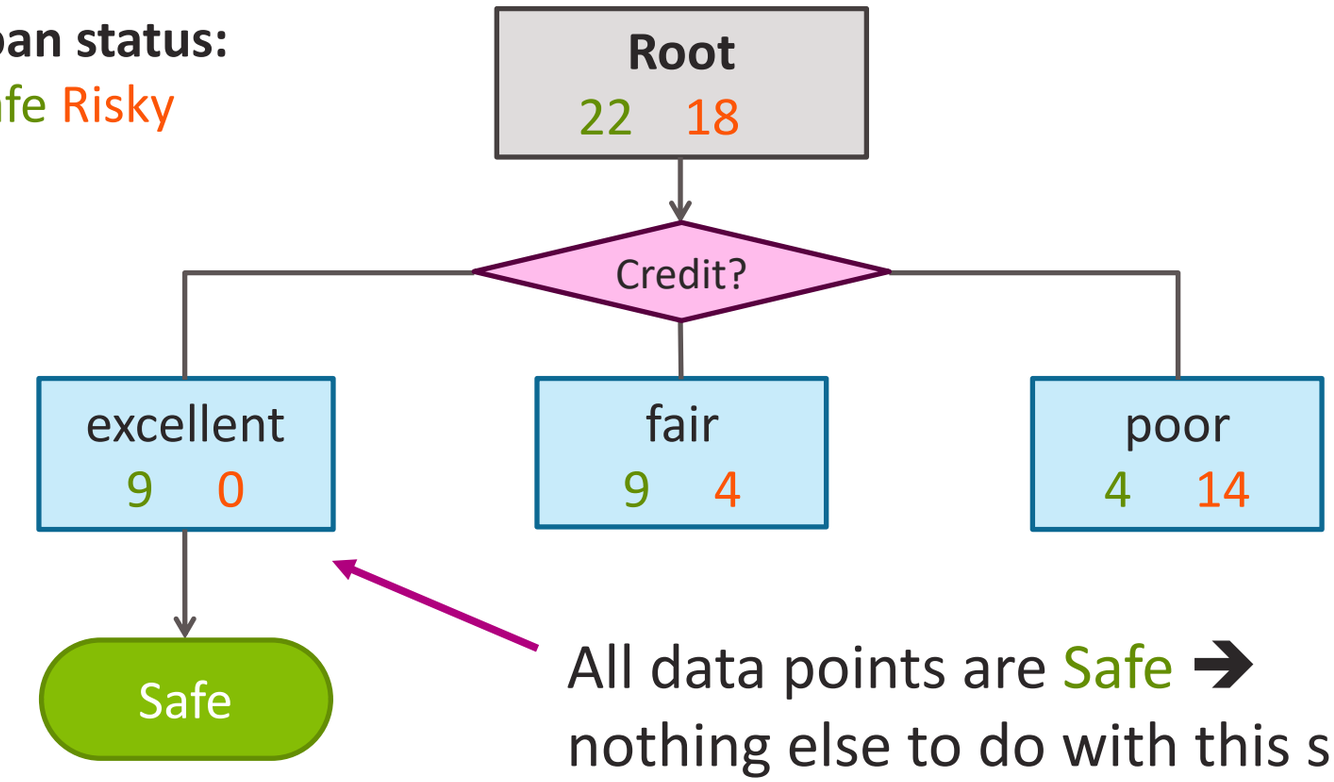
Greedy algorithm for growing a decision tree

- Start with the root node as an intermediate node
- Repeat if there exists an intermediate node
  - Choose a feature x[i] to split at the intermediate node that maximizes the accuracy
  - Change the intermediate node into a internal node branching on x[i]
  - Add intermediate nodes to each branch
  - If a intermediate node meets the stopping rule, change it to a leaf node and make a prediction

- Stopping rule:
  - 1. Do not branch if at that intermediate node, all data have the same label (perfect prediction)
  - 2. Do not branch if no feature left to branch

Greedy approach

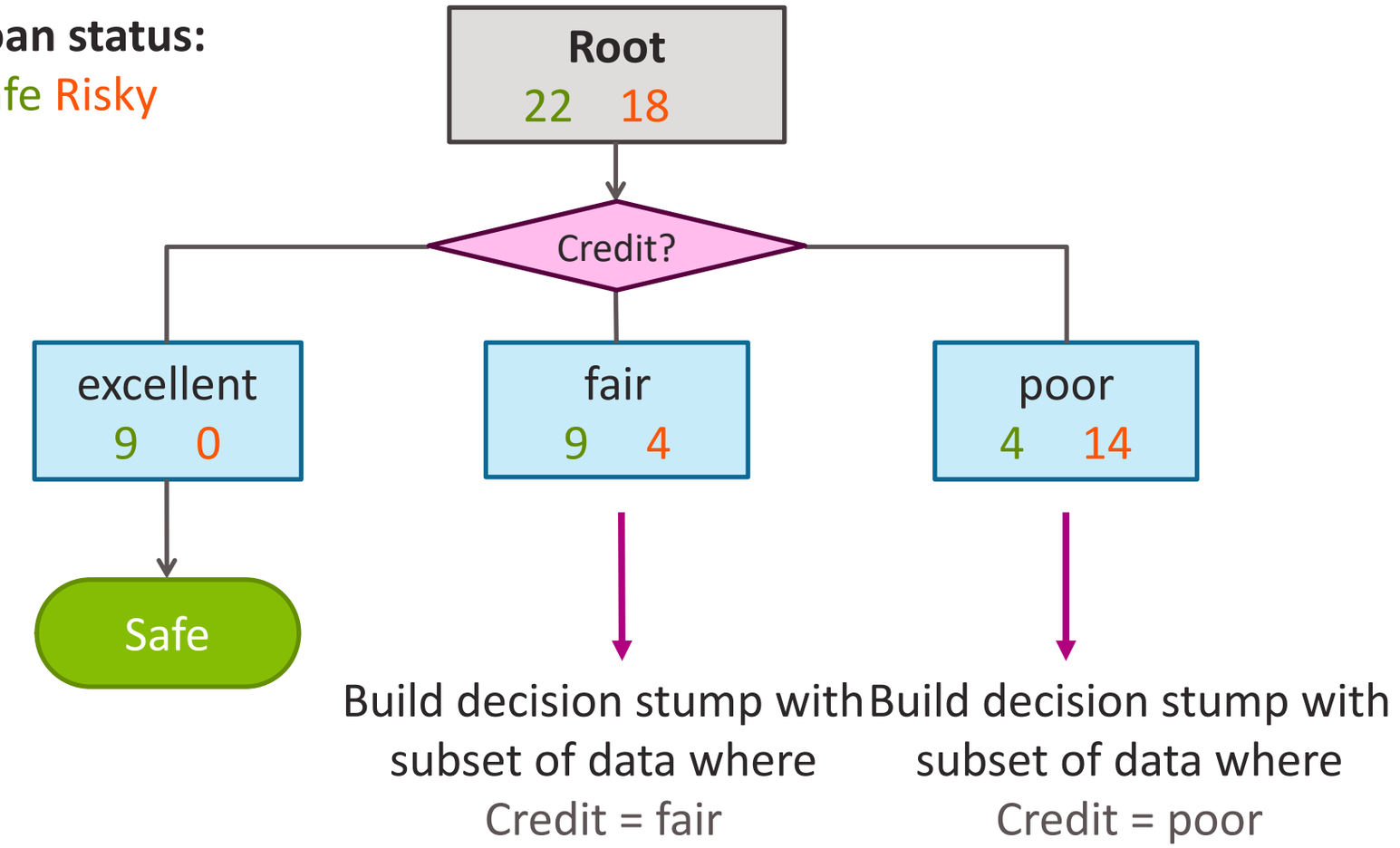
Loan status:
Safe Risky



All data points are Safe ➔
nothing else to do with this subset of data

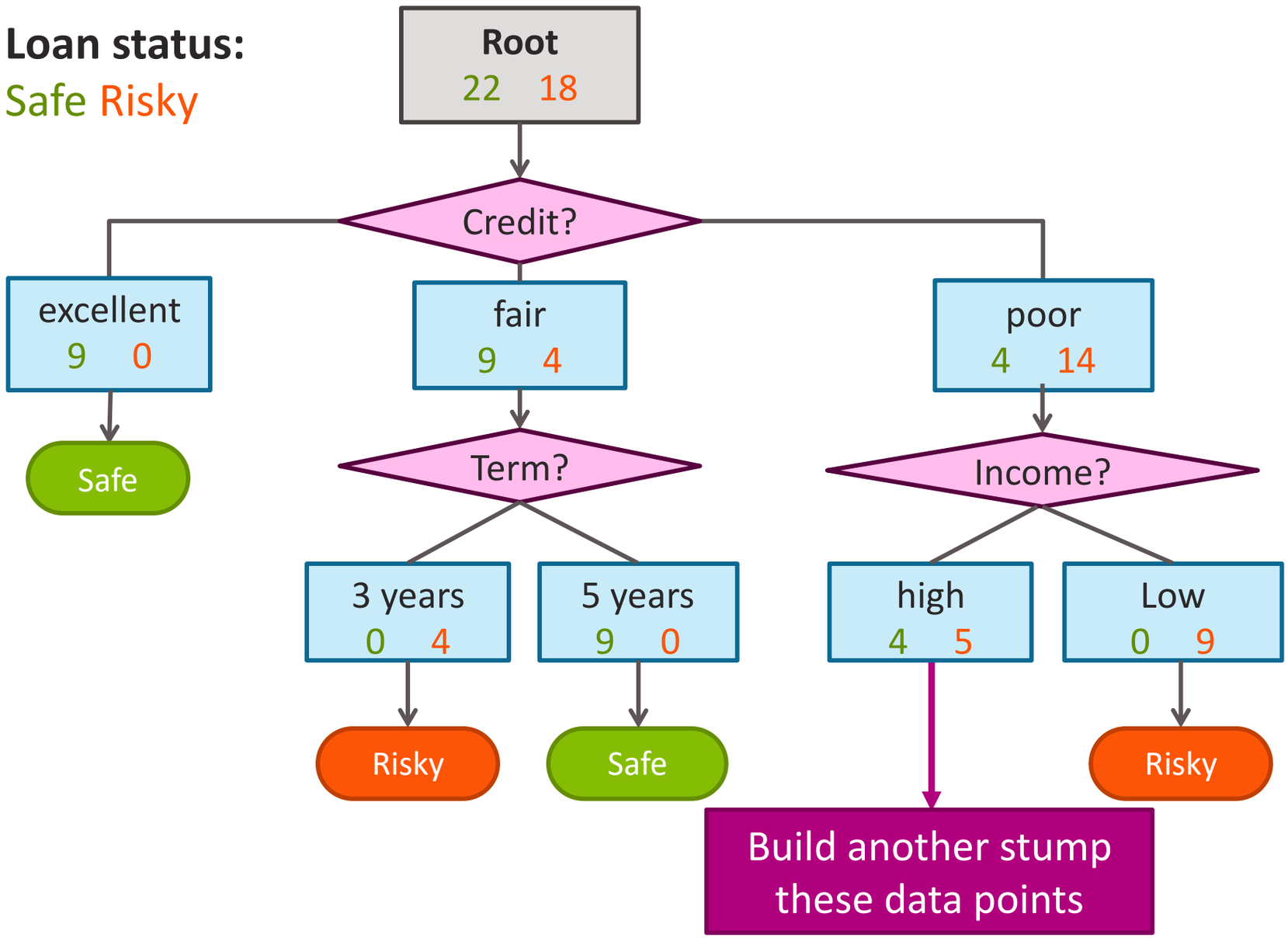
Greedy approach

Loan status:
Safe Risky

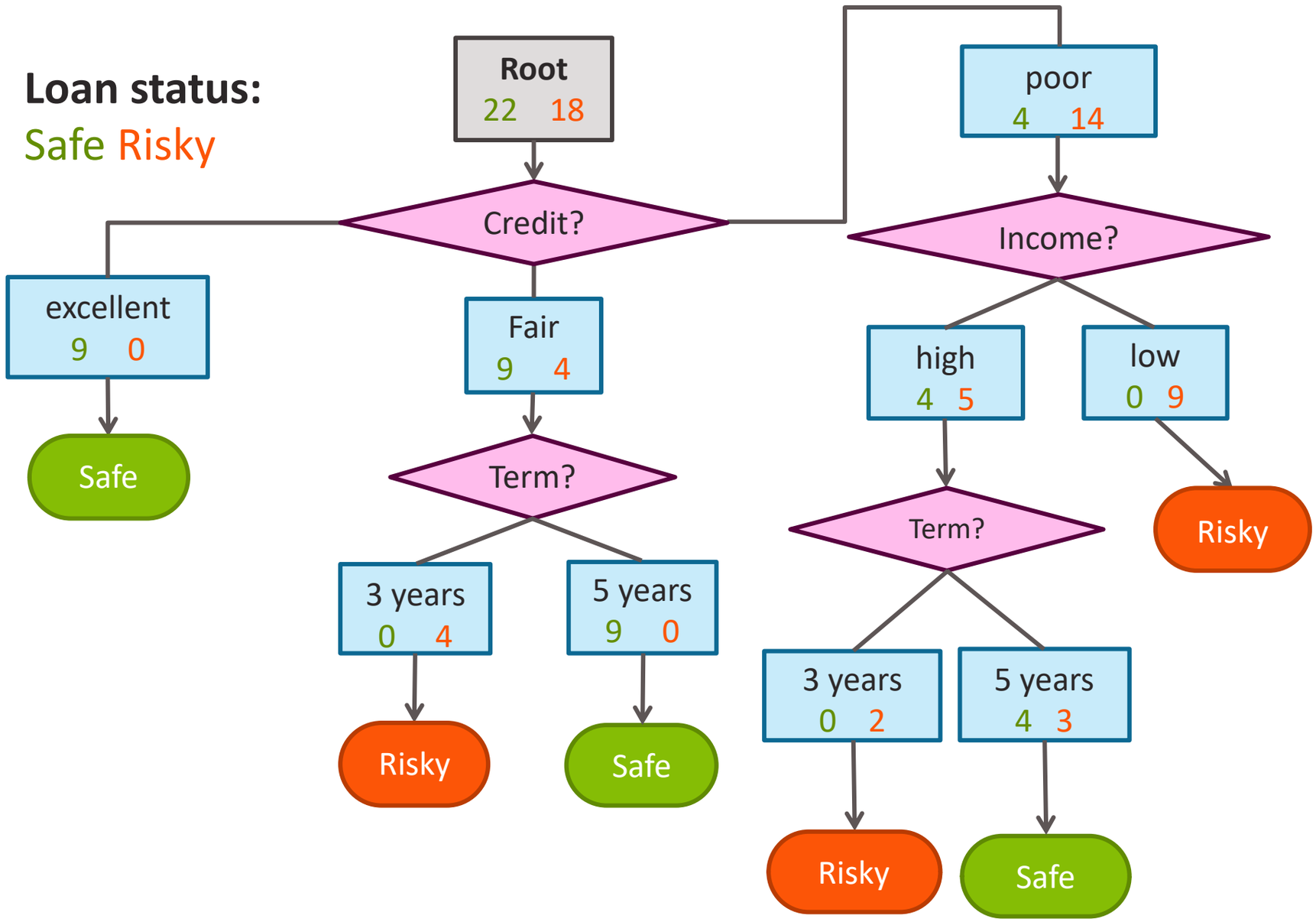


End of second level

Loan status:
Safe Risky



Final decision tree



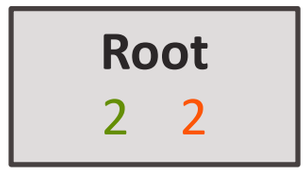
- Branching only increases the accuracy (and decreases the error)
- The final accuracy is 37/40
- Why not branch further?

Another potential early stopping rule

- If a branching does not increase accuracy, should we still branch?

XOR

x[1]	x[2]	y
FALSE	FALSE	FALSE
FALSE	TRUE	TRUE
TRUE	FALSE	TRUE
TRUE	TRUE	FALSE

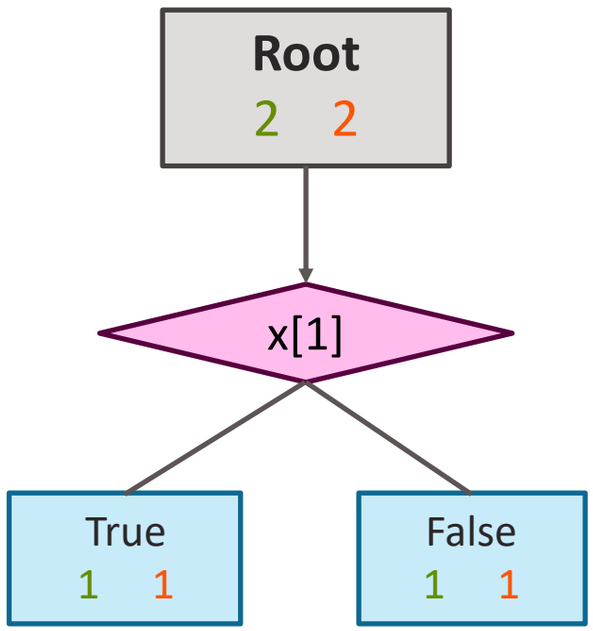


Tree	Classification error
(root)	0.5

Another potential early stopping rule

- If a branching does not increase accuracy, should we still branch?

x[1]	x[2]	y
FALSE	FALSE	FALSE
FALSE	TRUE	TRUE
TRUE	FALSE	TRUE
TRUE	TRUE	FALSE



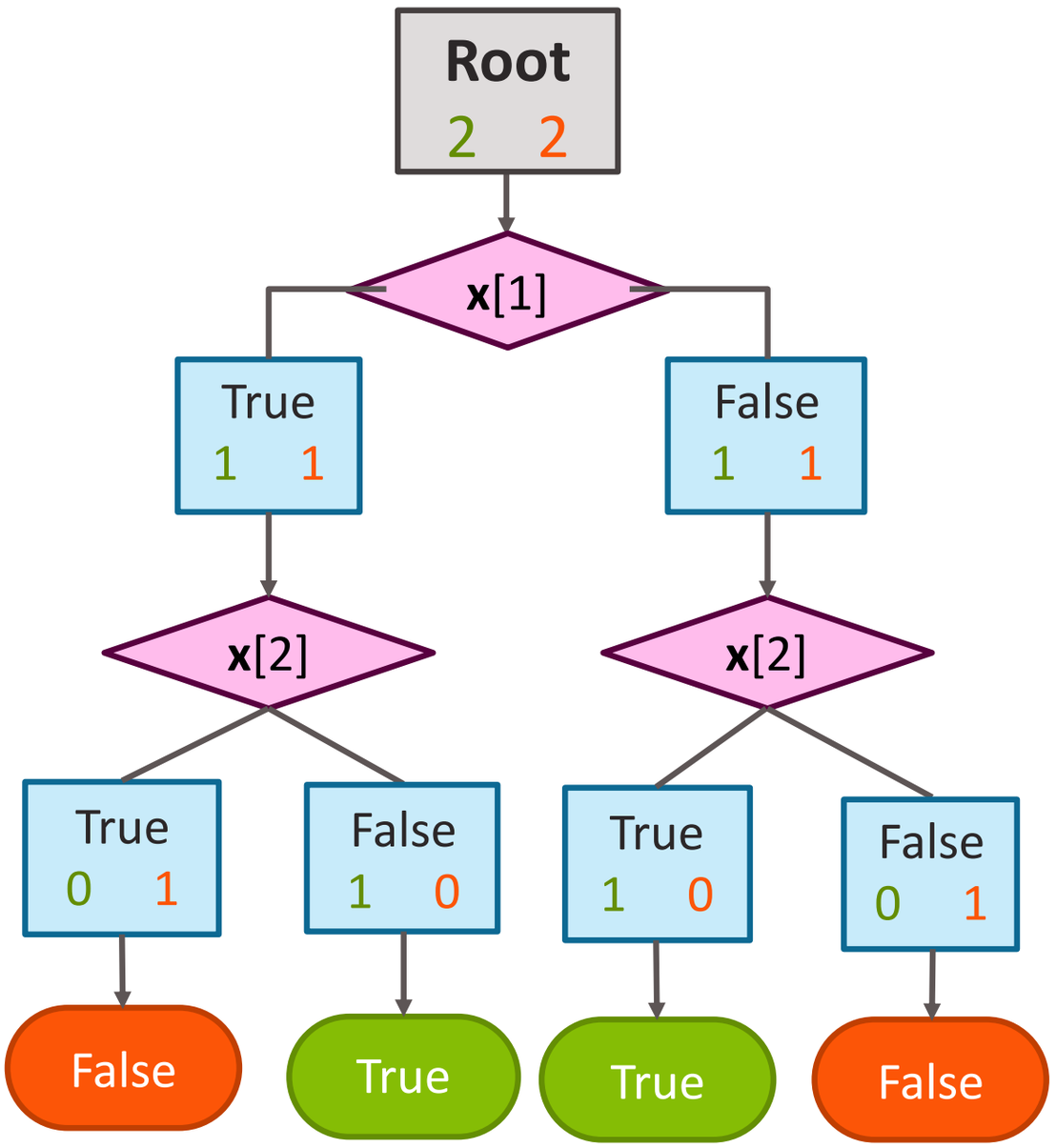
Tree	Classification error
(root)	0.5
Split on x[1]	0.5

Another potential early stopping rule

- If a branching does not increase accuracy, should we still branch? Yes.

x[1]	x[2]	y
FALSE	FALSE	FALSE
FALSE	TRUE	TRUE
TRUE	FALSE	TRUE
TRUE	TRUE	FALSE

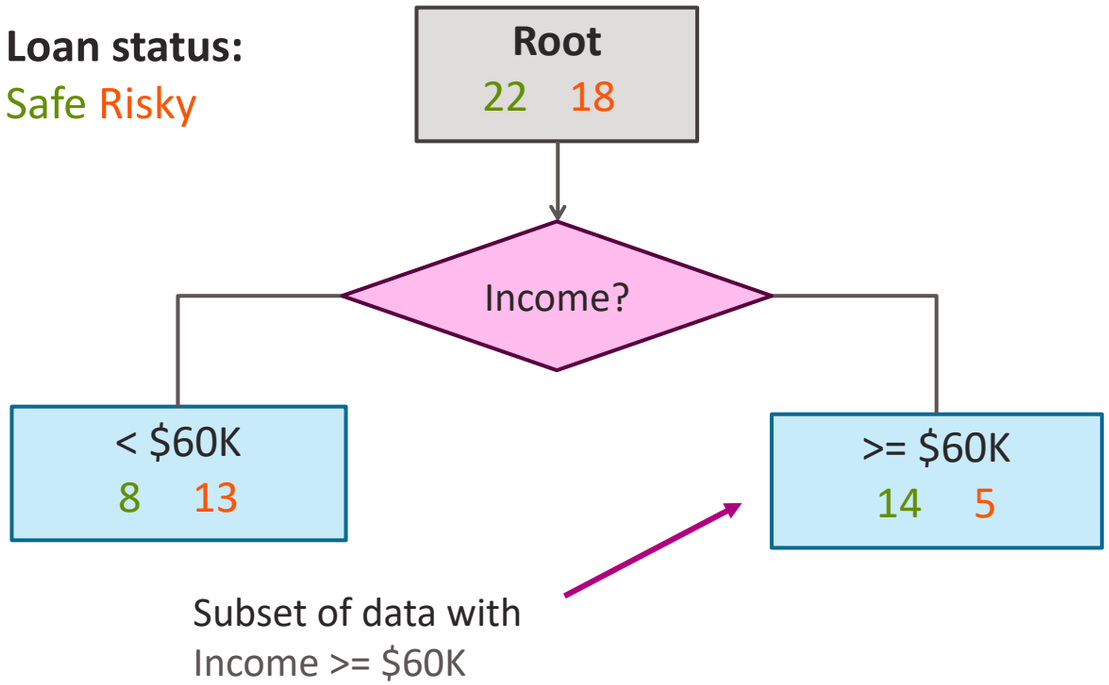
Tree	Classification error
(root)	0.5
Split on x[1]	0.5
Split on x[1],x[2]	0



Decision trees on real valued data

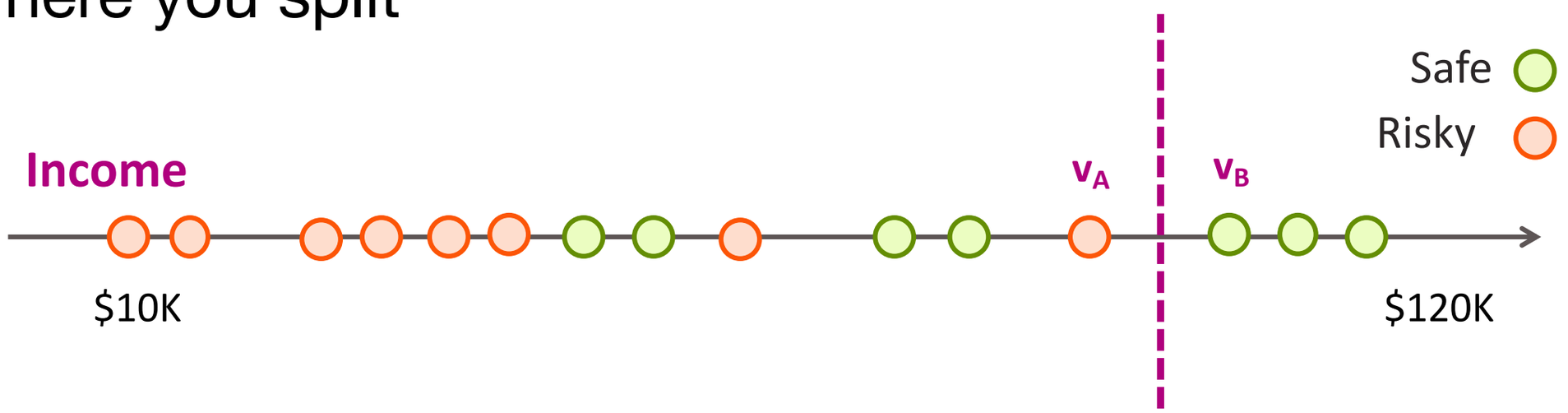
Binary branching on real valued data

Income	Credit	Term	y
$105 K	excellent	3 yrs	Safe
$112 K	good	5 yrs	Risky
$73 K	fair	3 yrs	Safe
$69 K	excellent	5 yrs	Safe
$217 K	excellent	3 yrs	Risky
$120 K	good	5 yrs	Safe
$64 K	fair	3 yrs	Risky
$340 K	excellent	5 yrs	Safe
$60 K	good	3 yrs	Risky

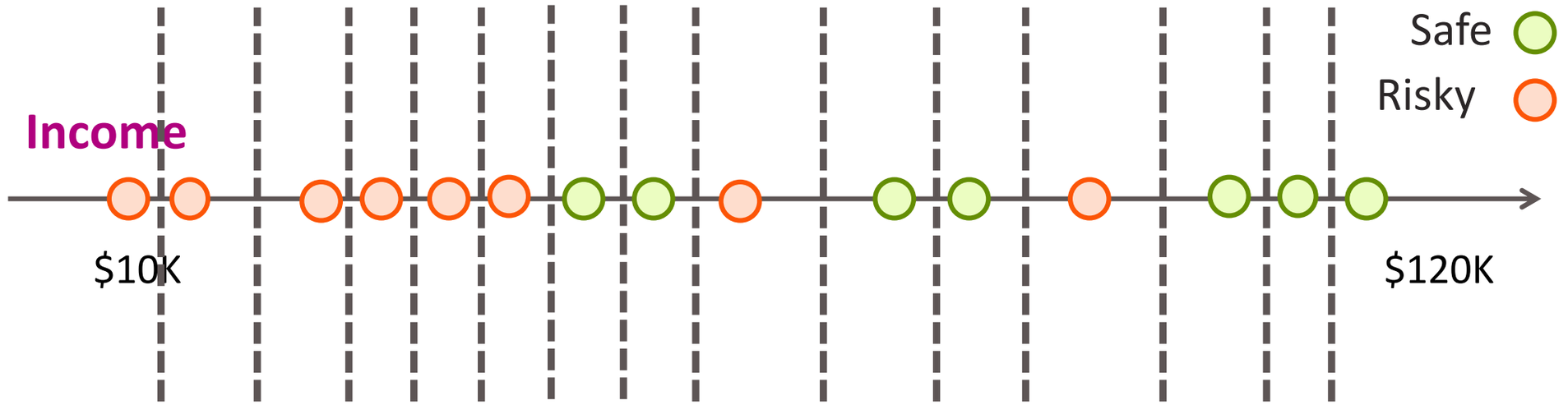


- Is there any gain in ternary or k-ary branching?

- Suppose we want to branch on a real valued feature x[i], then how do we choose the threshold?
  - Between two data points, it does not make any difference where you split

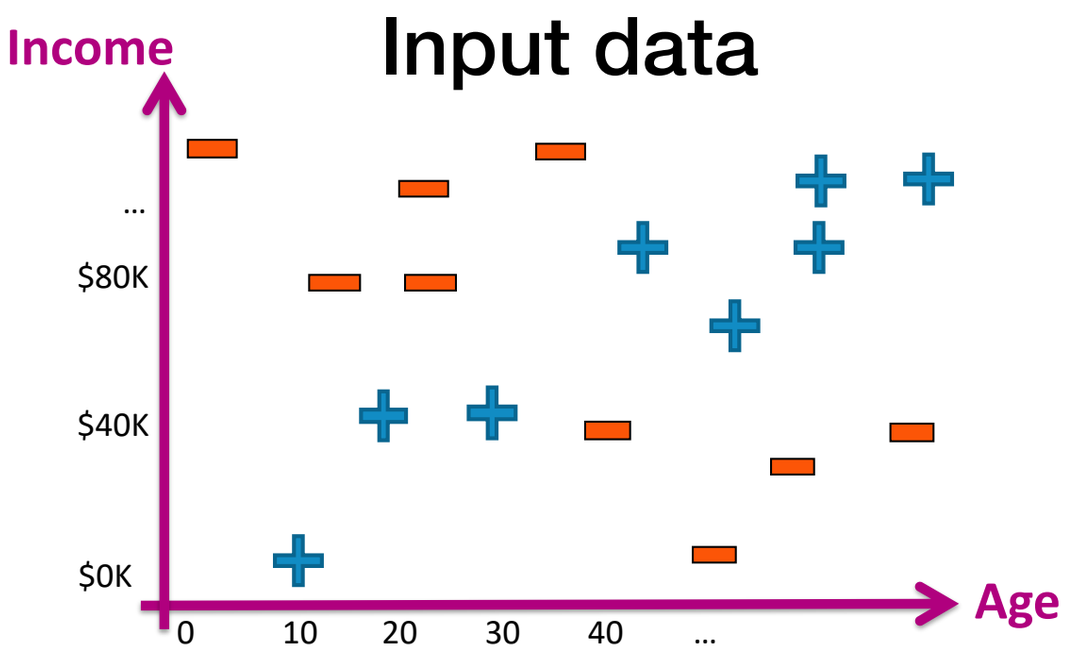


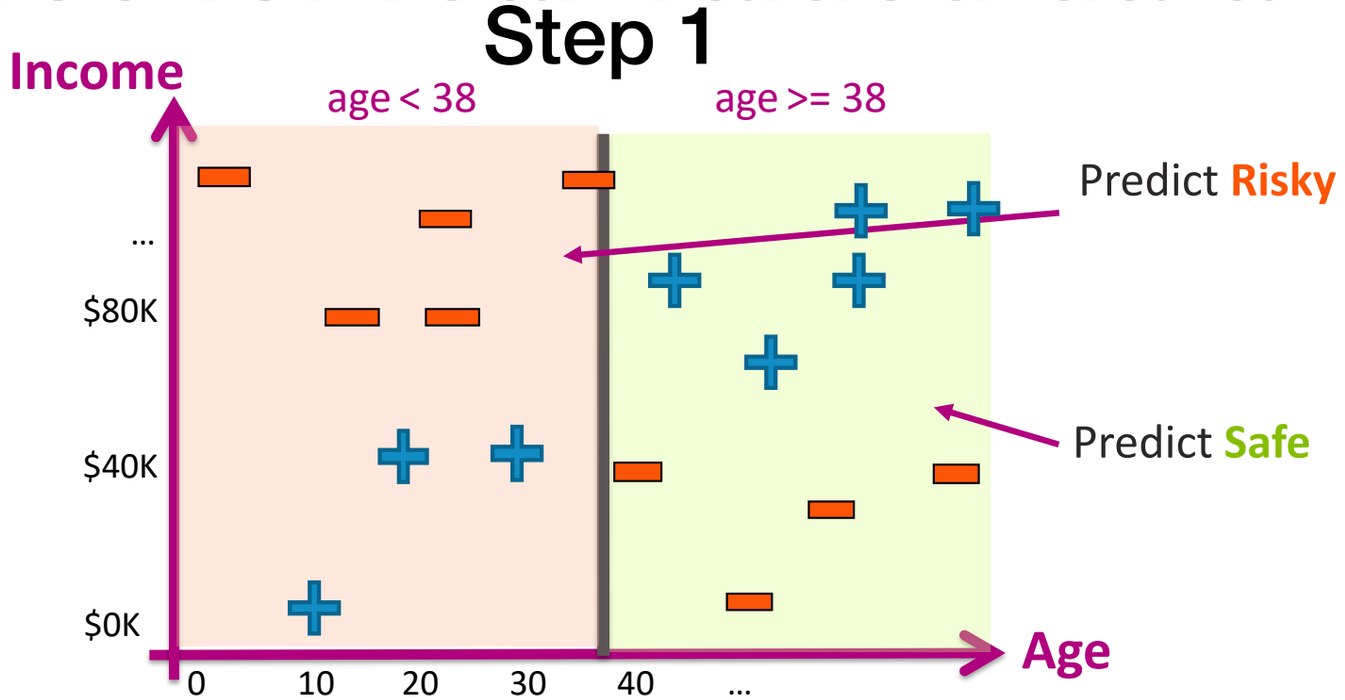
- there are really only a finite number of choices

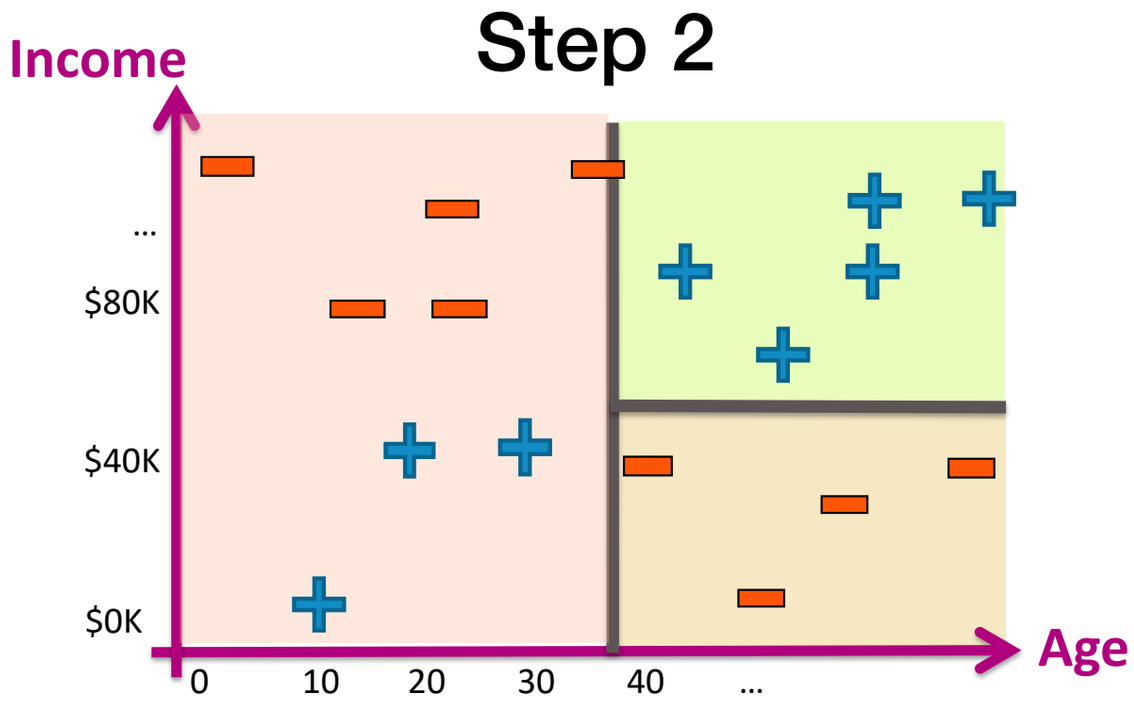


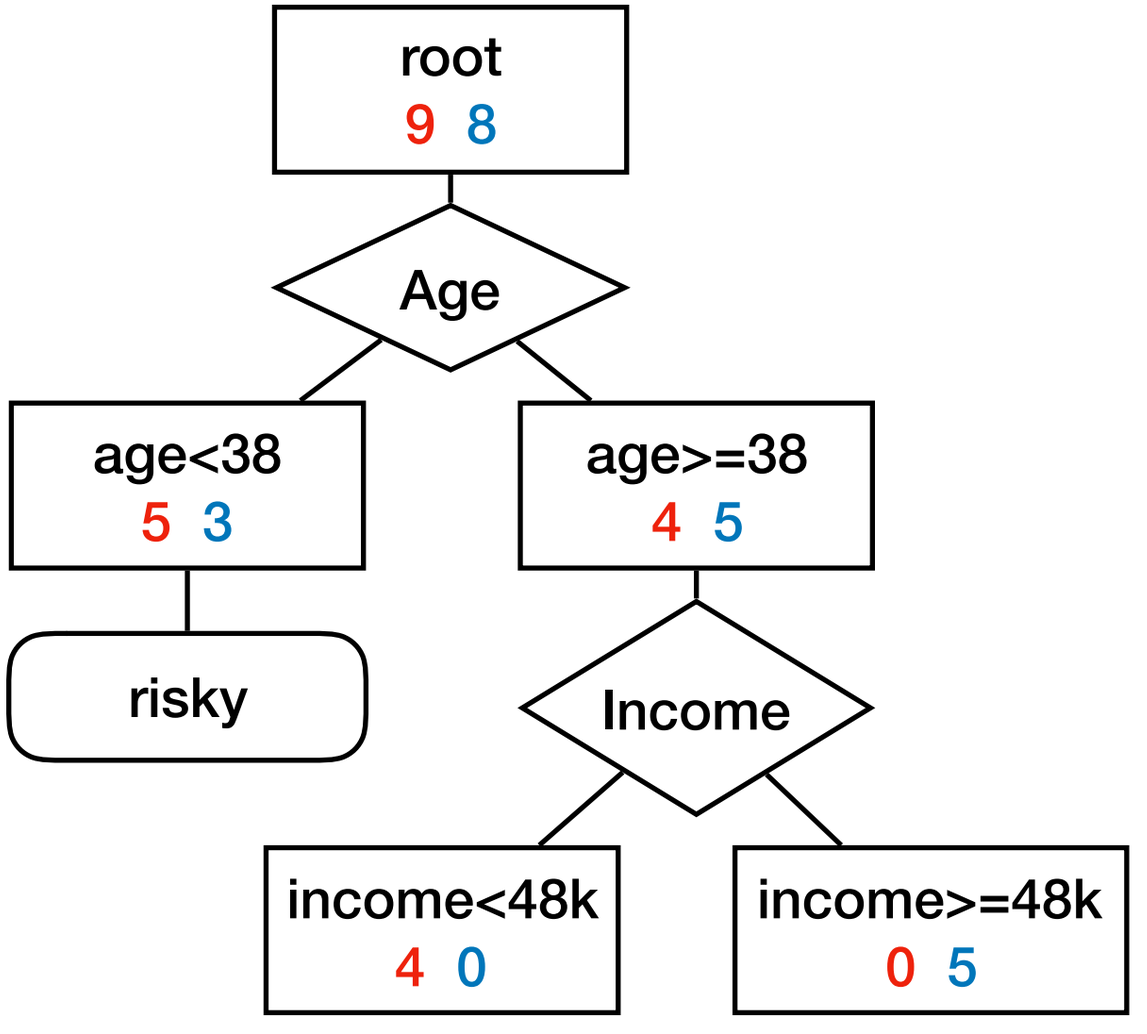
- We choose one with lowest error (or maximum accuracy)

Growing decision tree for real-valued data



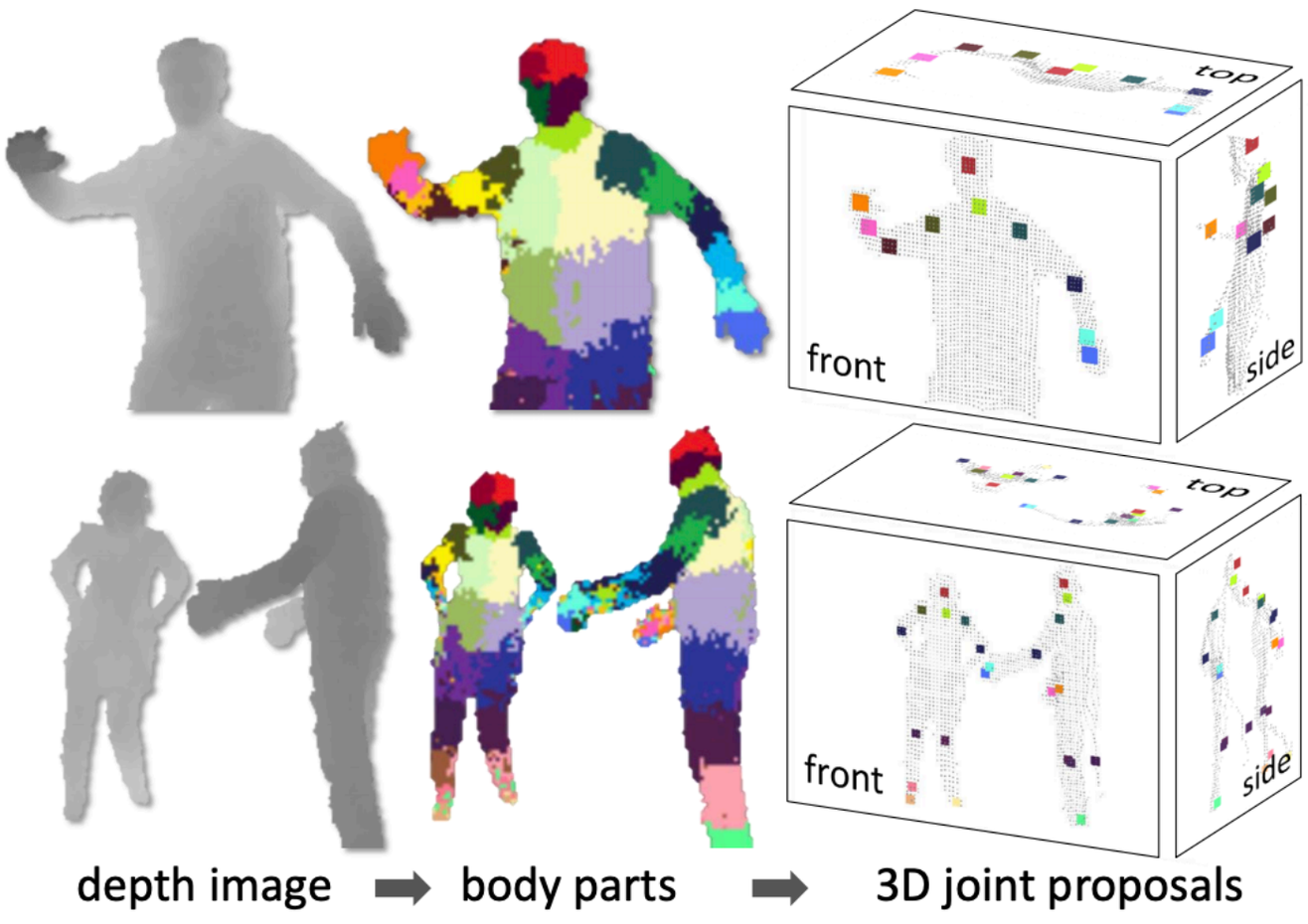






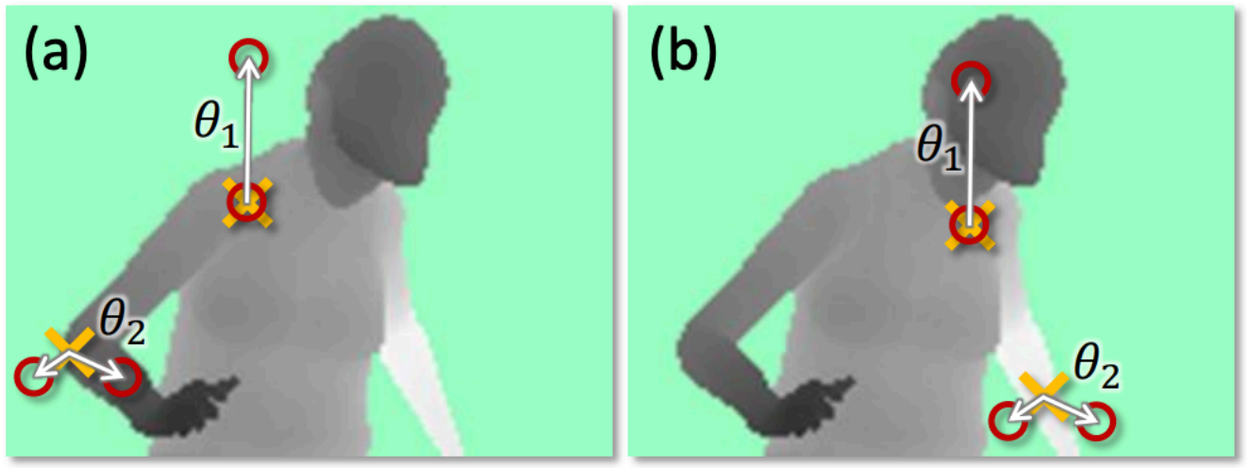
Real-Time Human Pose Recognition in Parts from Single Depth Images

Jamie Shotton Andrew Fitzgibbon Mat Cook Toby Sharp Mark Finocchio
Richard Moore Alex Kipman Andrew Blake
Microsoft Research Cambridge & Xbox Incubation

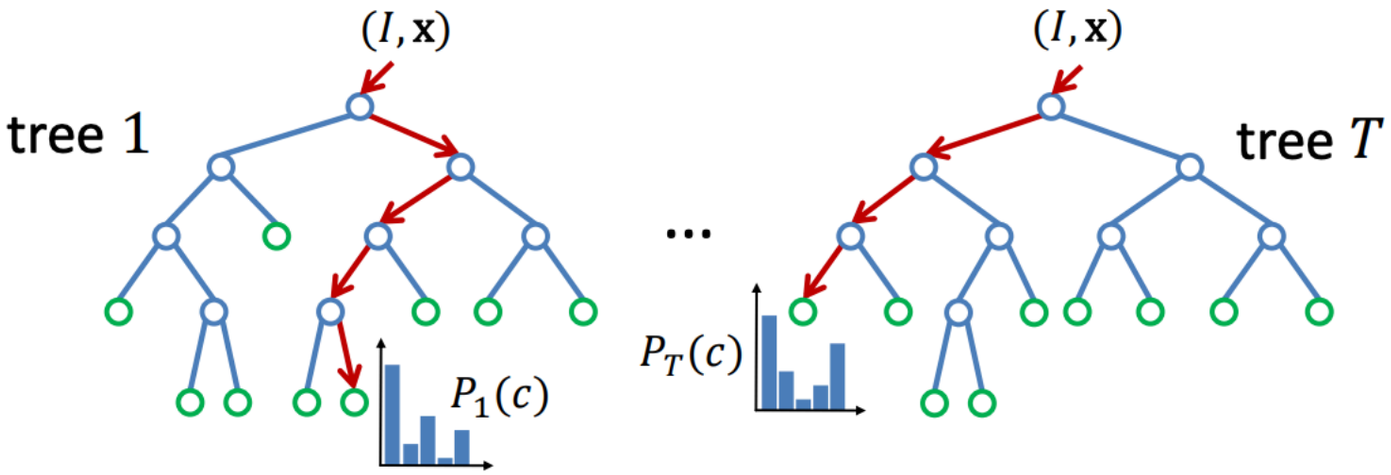


Class is 31 body parts: LU/RU/LW/RW head, neck, L/R shoulder, LU/RU/LW/RW arm, L/R elbow, L/R wrist, L/R hand, LU/RU/LW/RW torso, LU/RU/LW/RW leg, L/R knee, L/R ankle, L/R foot (Left, Right, Upper, loWer).

features:

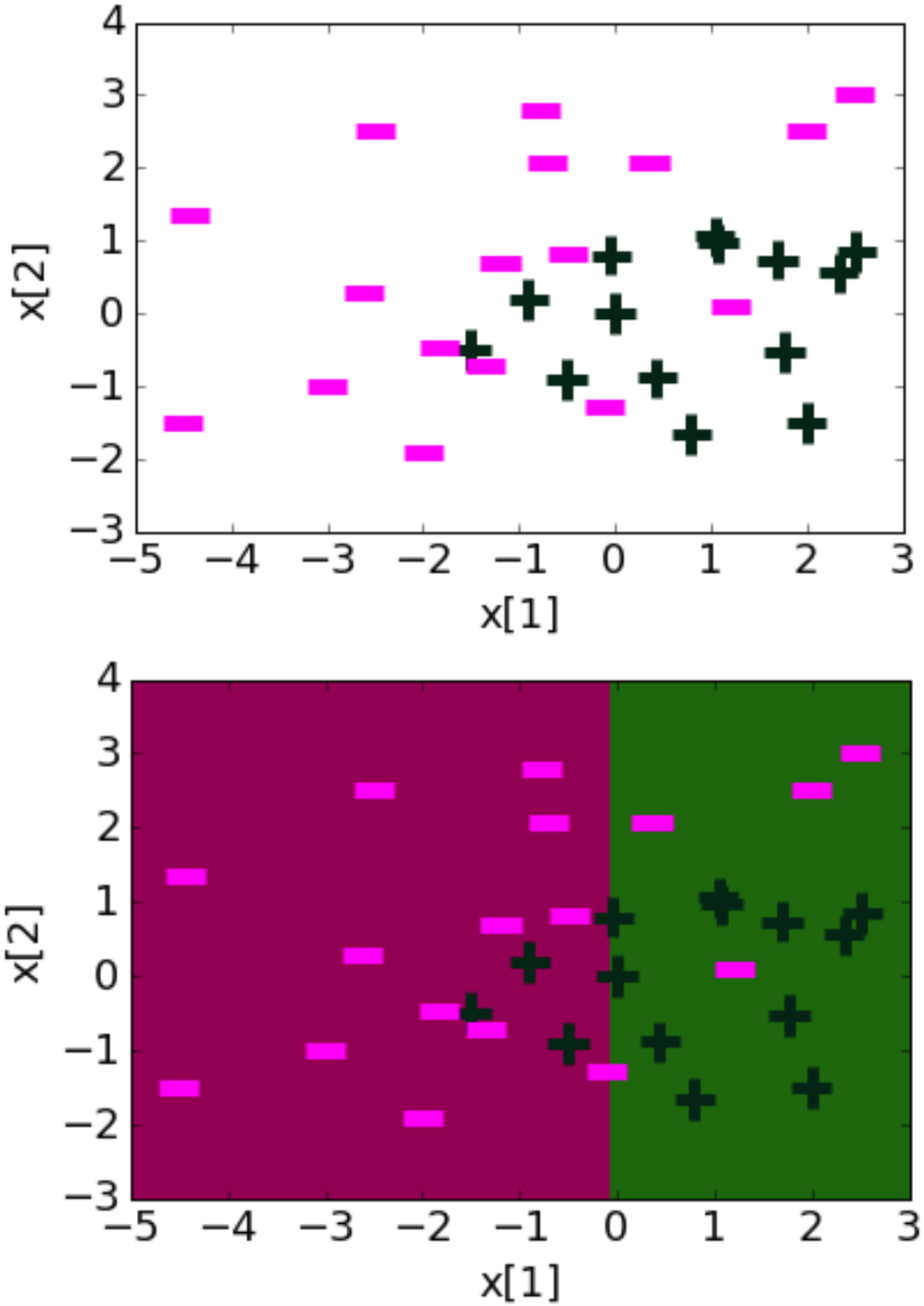


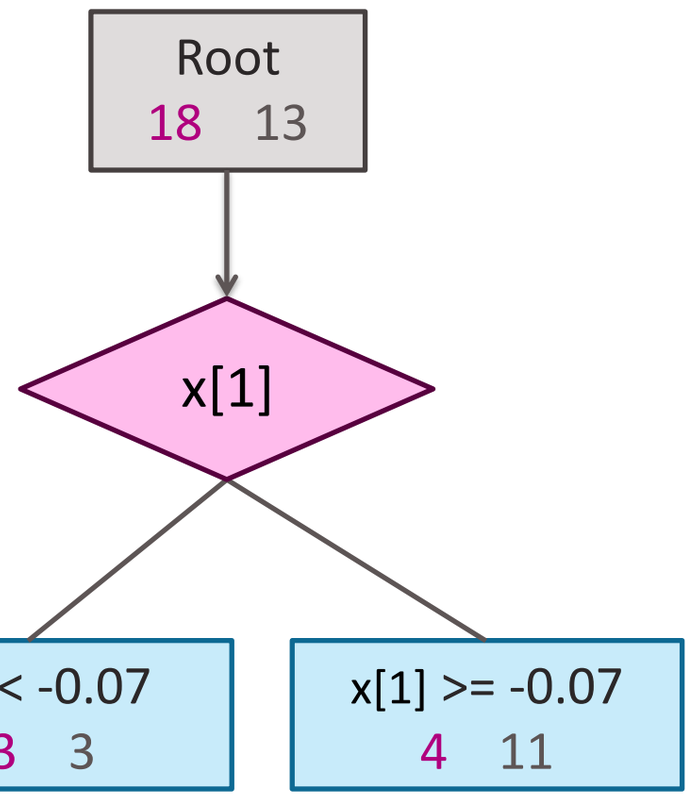
Decision tree:

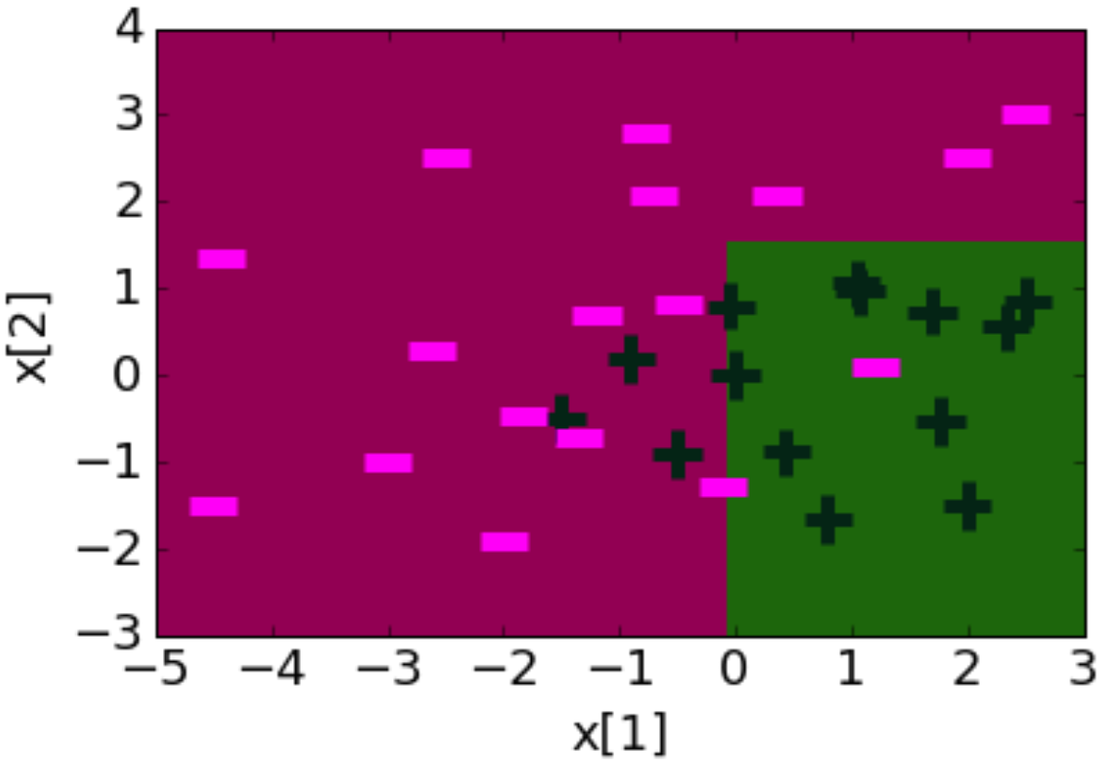


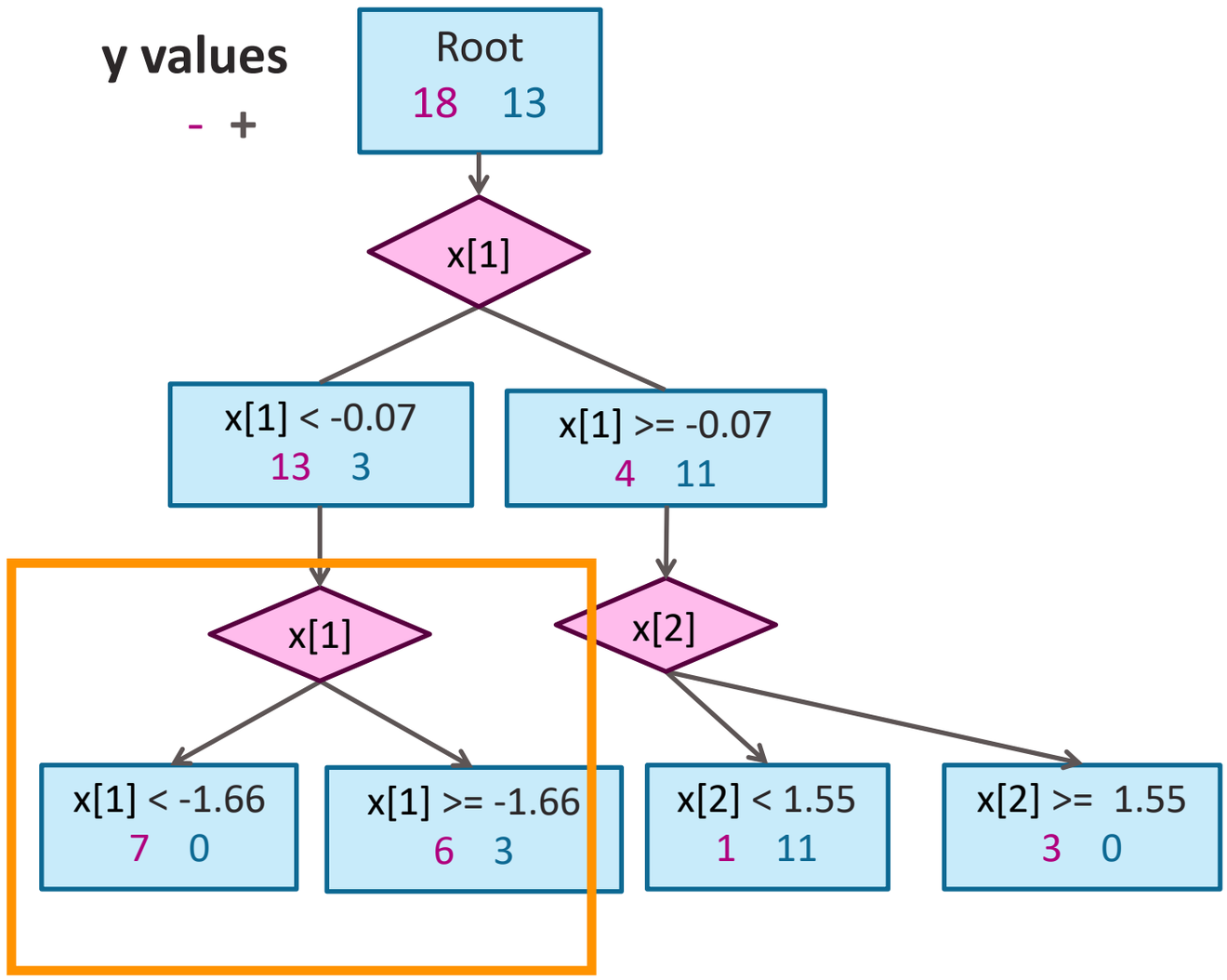
Classification with decision trees

- Data







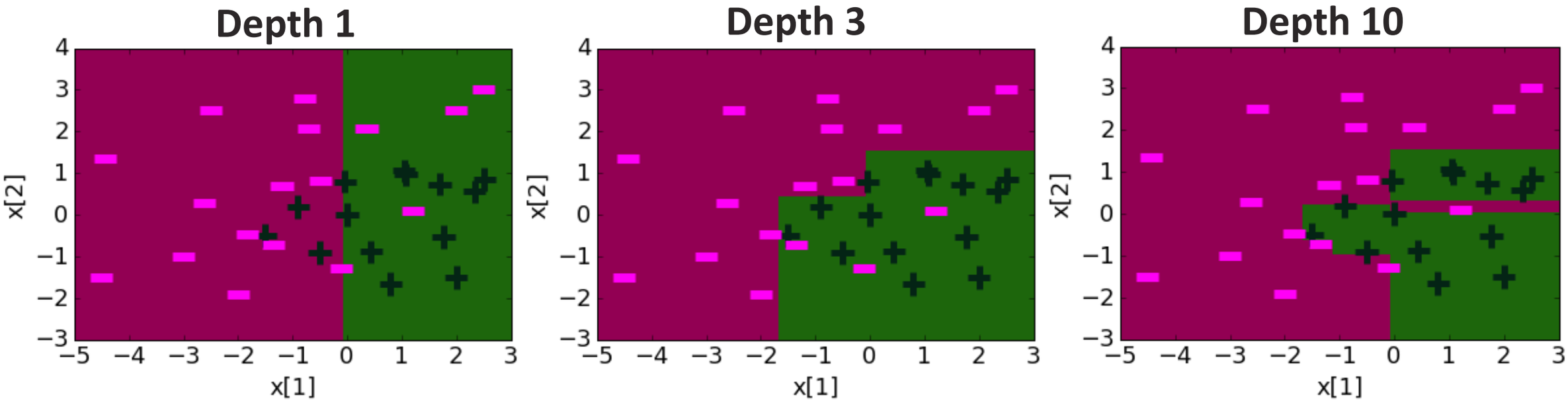


For threshold splits, same feature can be used multiple times

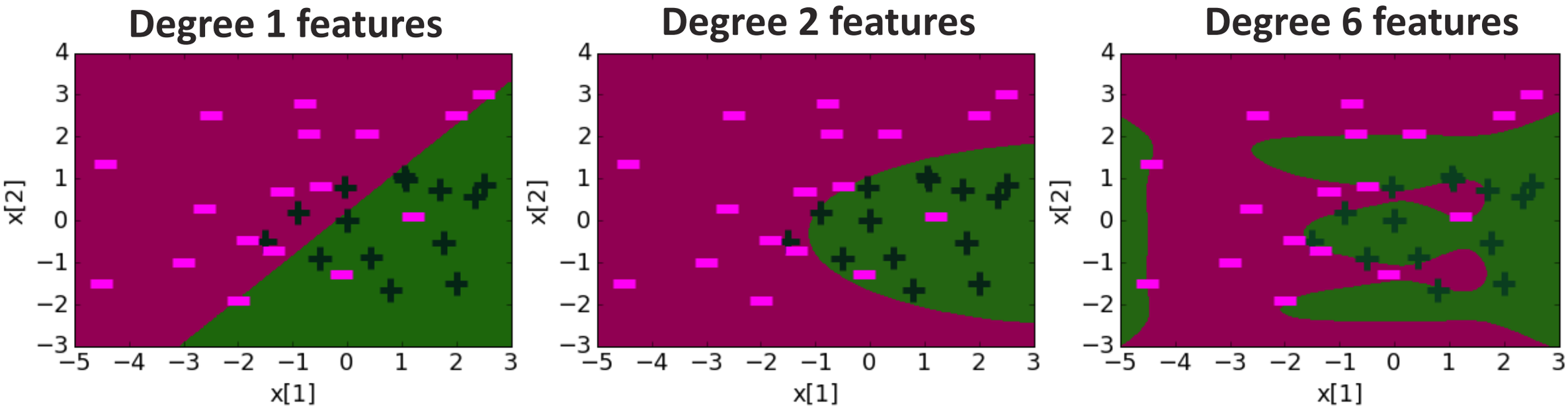
Evolution of decision boundaries

- Decision boundaries get more complicated as we increase the model complexity (here measured by the depth fo decision tree)

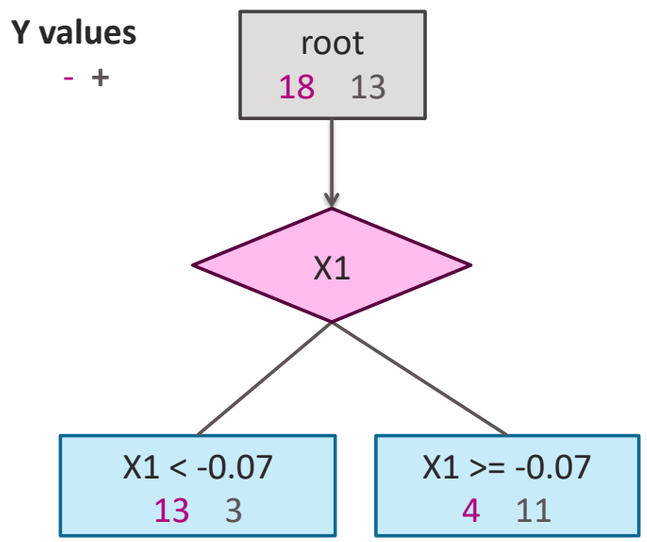
Decision Tree

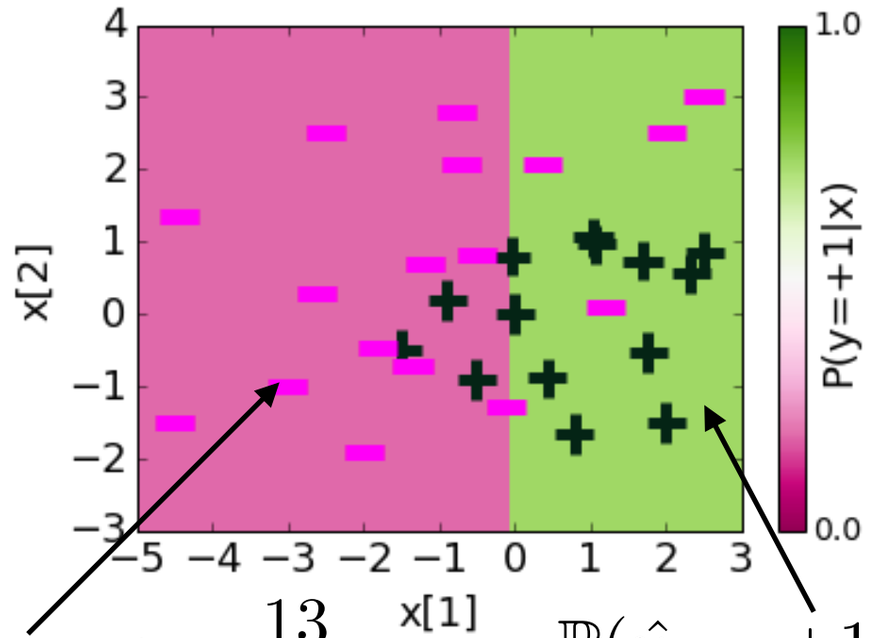


Logistic Regression



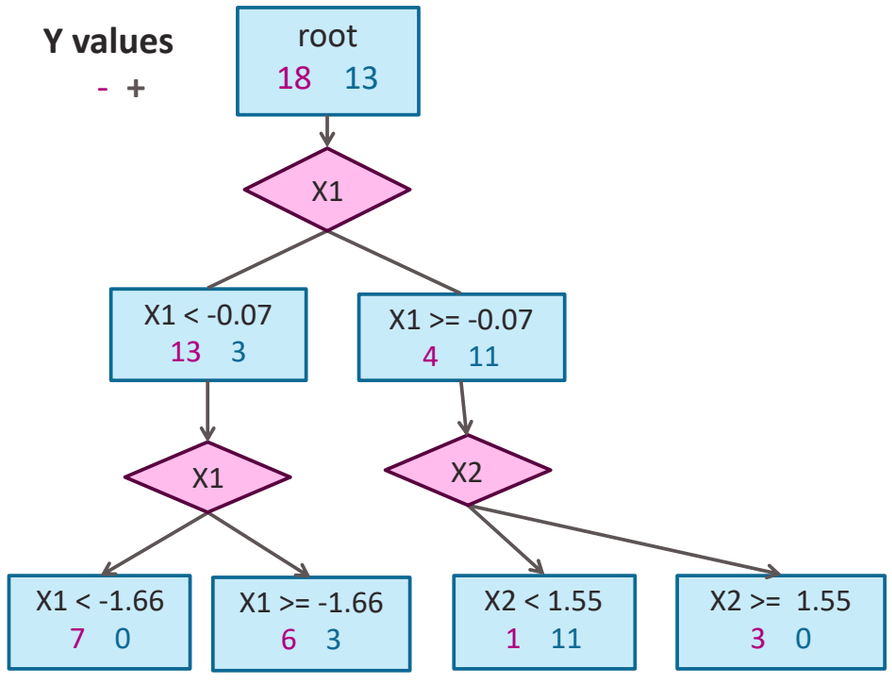
Probabilistic prediction with decision trees

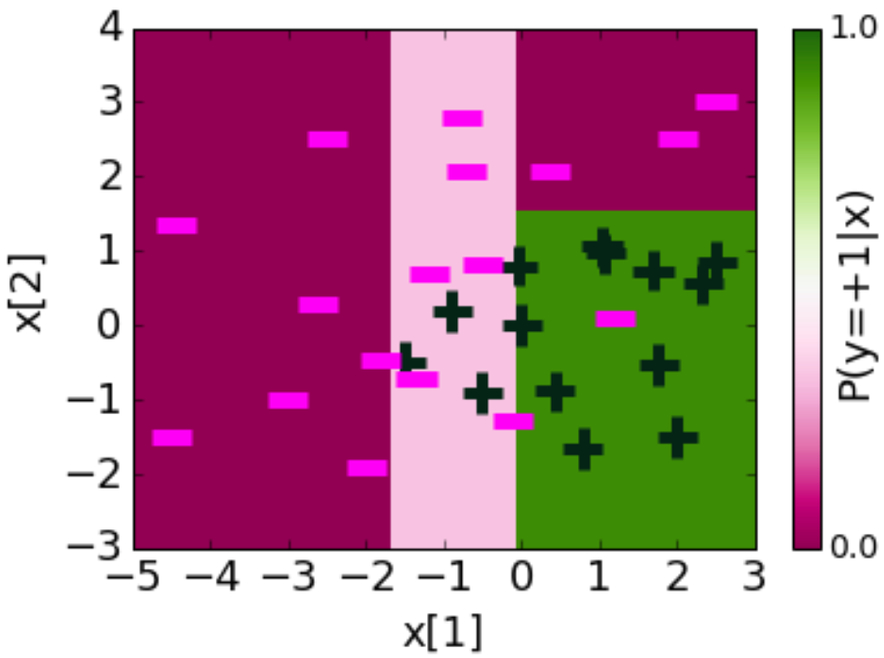




$$\mathbb{P}(\hat{y}_i = -1) = \frac{13}{16}$$

$$\mathbb{P}(\hat{y}_i = +1) = \frac{11}{15}$$





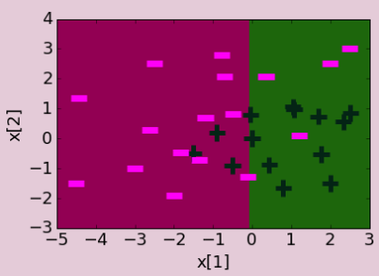
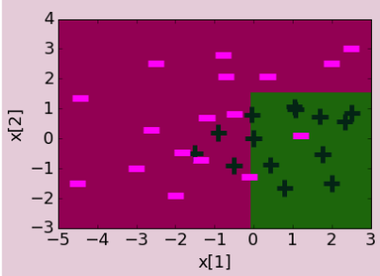
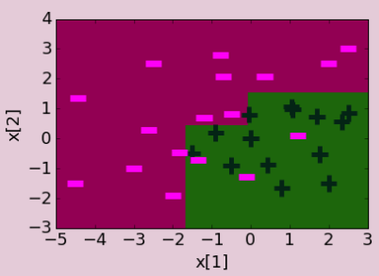
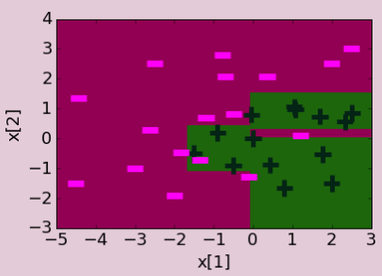
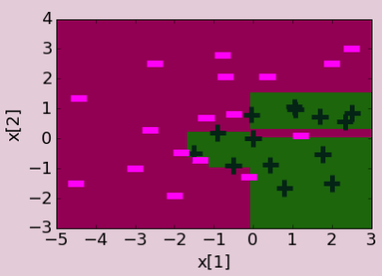
- By taking the probability of the training data in that subset, we can make probabilistic predictions

Overfitting

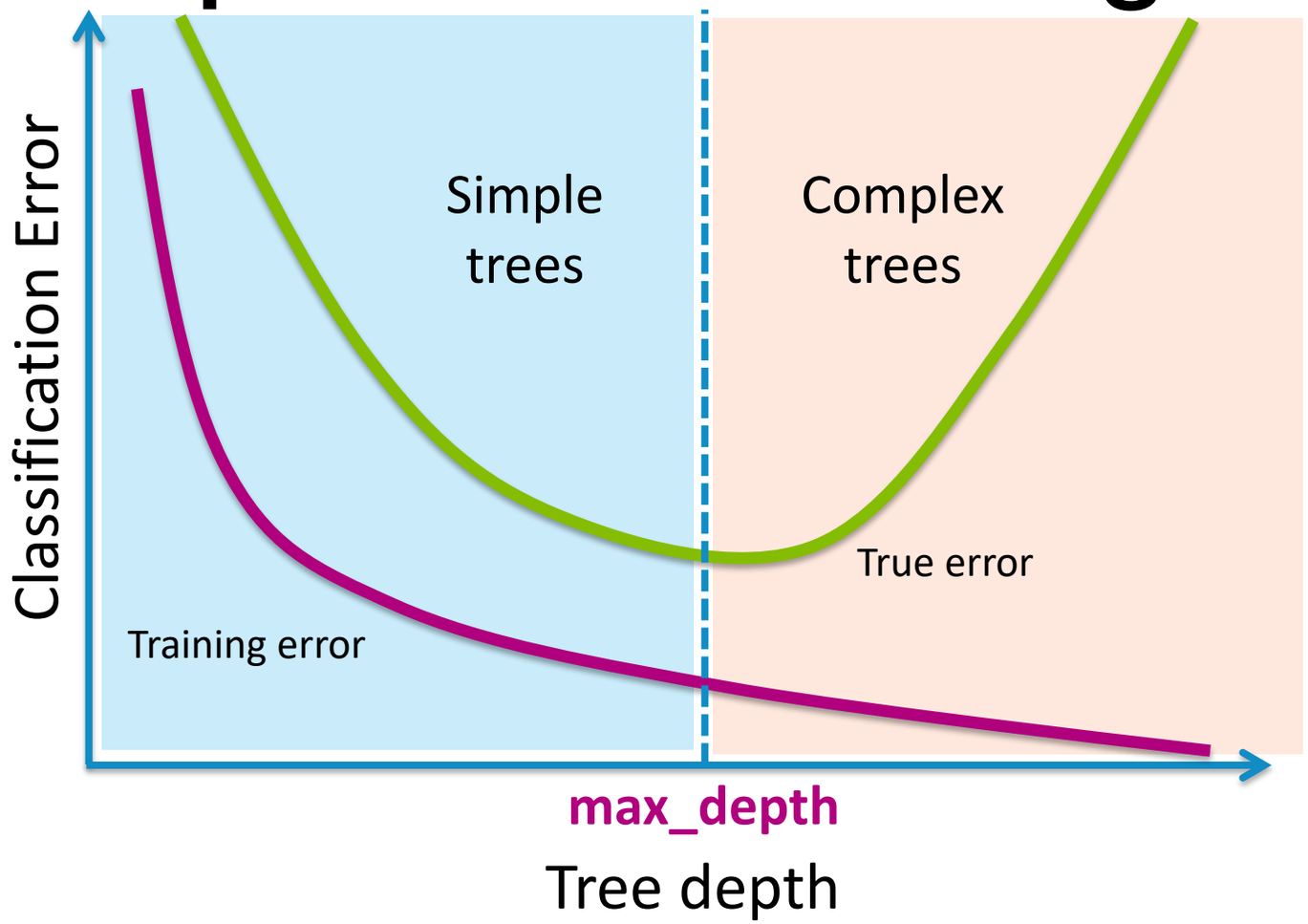
Trade-off between training error and depth

- As in other regression and classification approaches, training error monotonically decreases (or more precisely non-increases) with model complexity (usually measured in depth)
- Training error that is too small is a sign of overfitting

Training error reduces with depth

Tree depth	depth = 1	depth = 2	depth = 3	depth = 5	depth = 10
Training error	0.22	0.13	0.10	0.03	0.00
Decision boundary					

Two ways to prevent overfitting



- **1. Early stopping**
  - Stop before tree gets too complicated

- **2. Pruning**
  - Simplify after training a complex model

1. Early stopping

- **Stopping rule**: (Recap)
  - 1. All examples in the subset have the same label
  - 2. No more features left to split

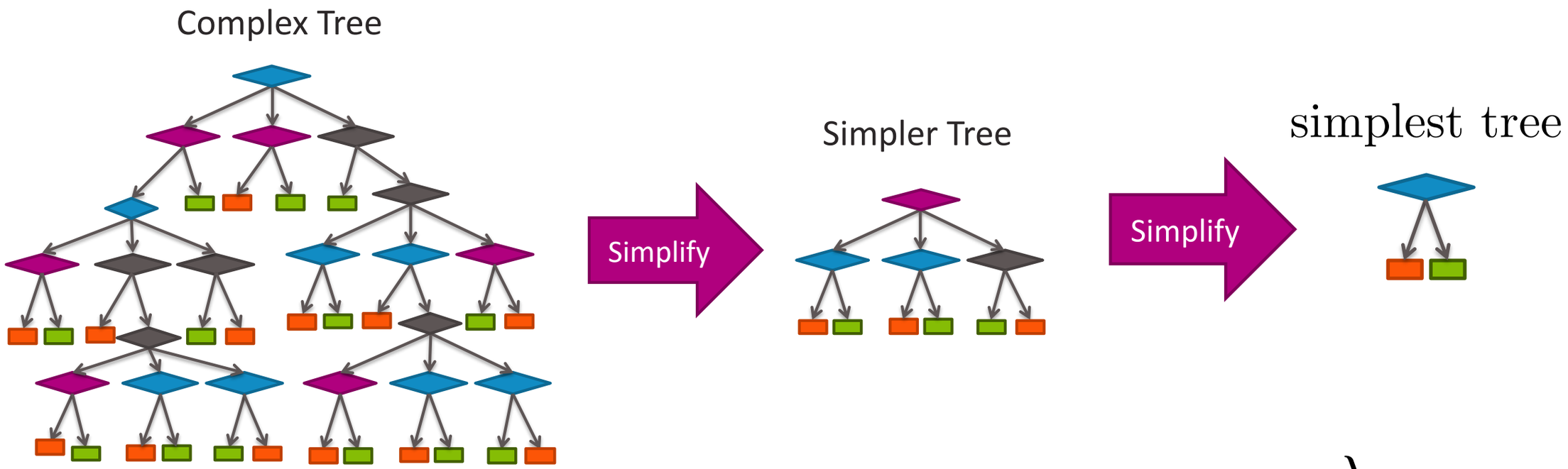
- **Early stopping rule**
  - Only grow up to max_depth (which is chosen via validation)
    - Hard to figure out the right depth
    - An imbalanced tree can be a better one
  - Do not split if it does not give sufficient decrease in error
    - There are cases where it takes some depth to get the gain
    - e.g. XOR

x[1]	x[2]	y
FALSE	FALSE	FALSE
FALSE	TRUE	TRUE
TRUE	FALSE	TRUE
TRUE	TRUE	FALSE

  - Do not split intermediate nodes with too few data points

2. Pruning

- Train till overfit, and then simplify



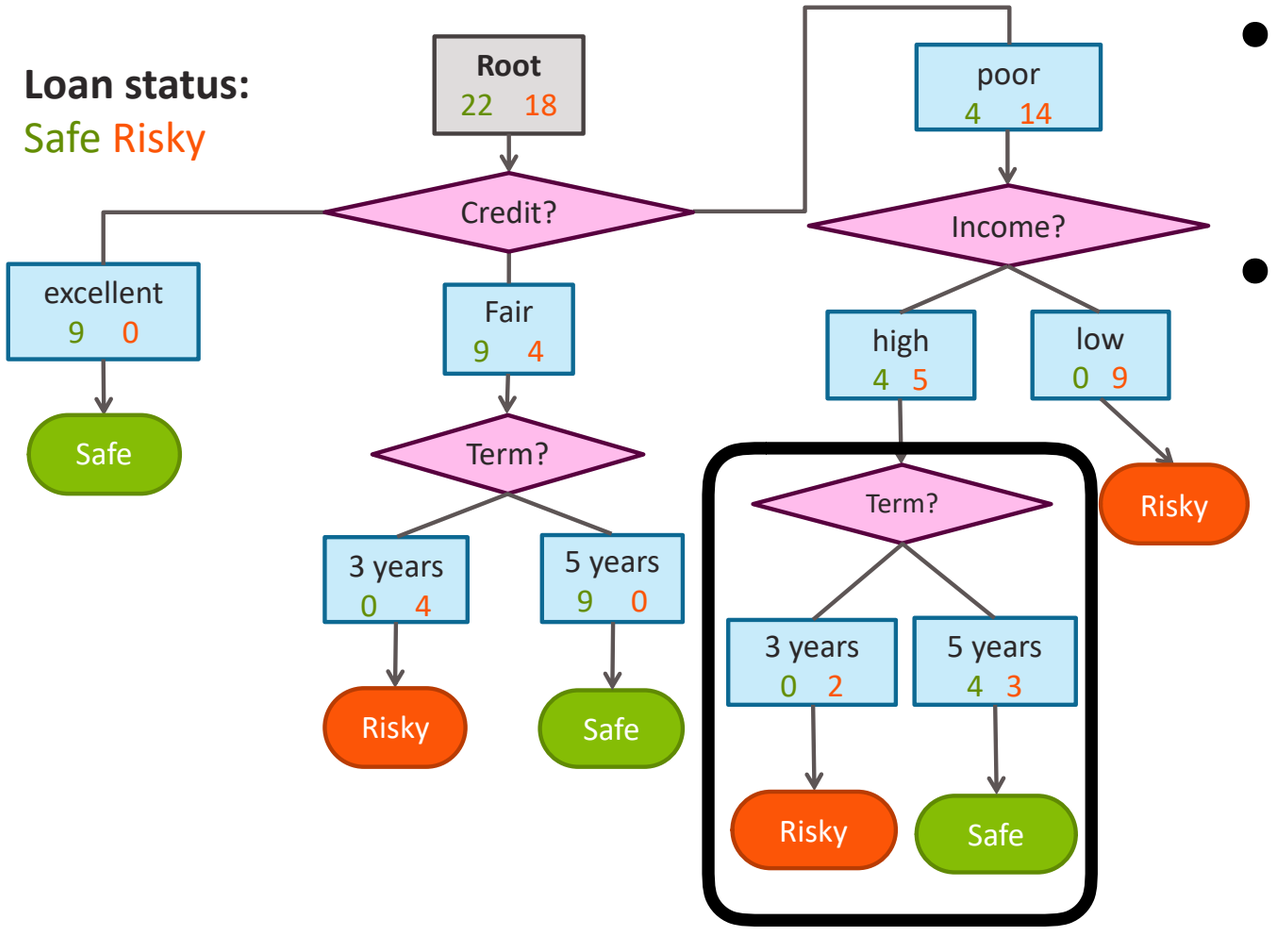
$\lambda = 0$ $\lambda = \infty$

- Pruning is guided by a choice of quality metric that balances fitting data and simplicity

$$\text{Loss}(T) \quad = \quad \text{Error}(T) + \lambda \underbrace{r(T)}_{\#\text{ of leaf nodes}}$$

  - Data fit is measured by the error
  - Simplicity is measured by the **number of leaves** in the tree

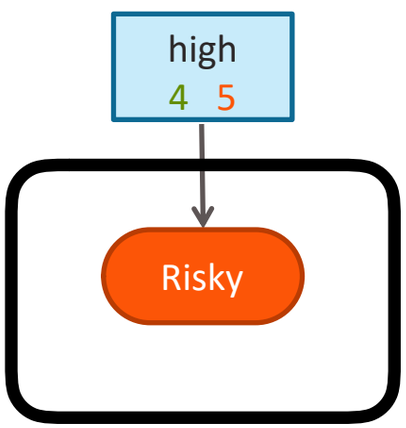
Pruning algorithm



- 1. Compute current loss, and find a **candidate split**
- 2. Replace the split by a leaf node and recompute

$$\lambda = 0.03$$

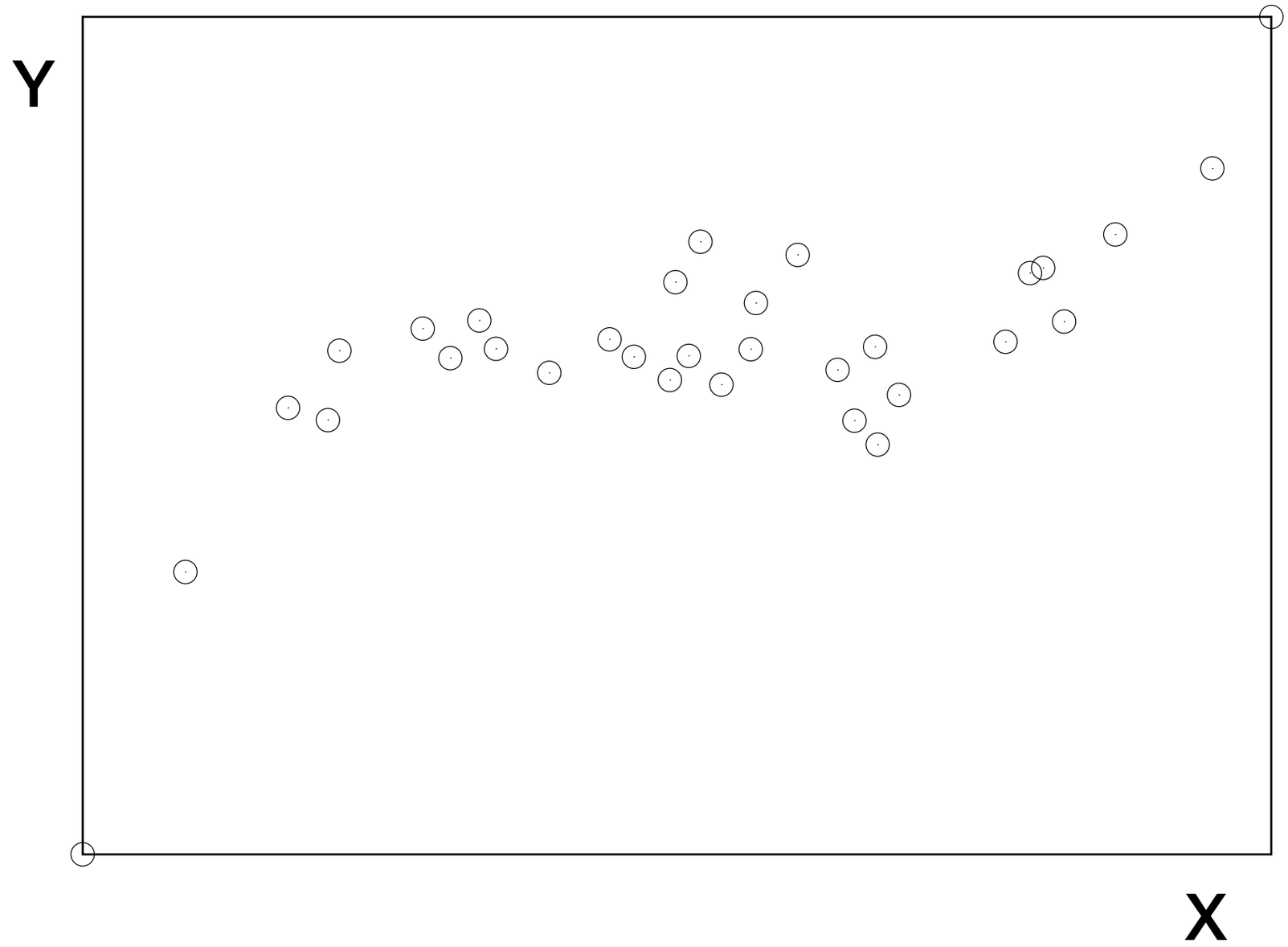
Tree	Error	#Leaves	Total
T	3/40	6	0.255
T-term	4/40	5	0.25
T-credit			
T-income			
T-term'			



$$\text{Loss}(T) = \text{Error}(T) + \lambda \underbrace{r(T)}_{\#\text{ of leaf nodes}}$$

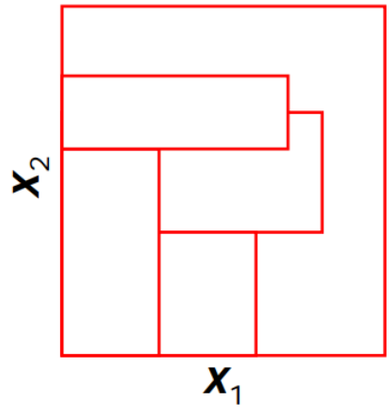
- 3. Repeat for all splits (there are 4 splits in this example)

Decision trees for regression

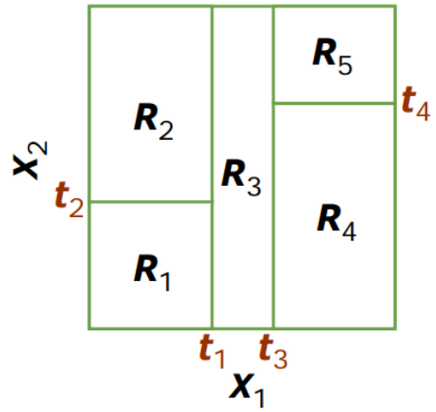


Decision trees for regression

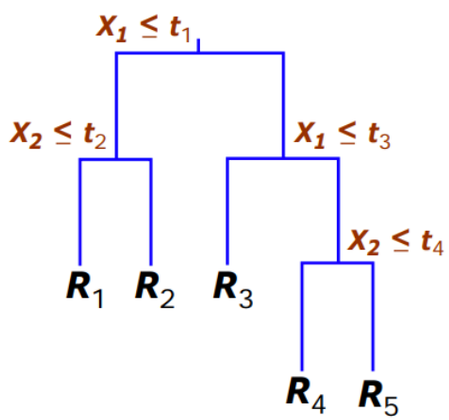
- Same process as classification, but
  - error measured in squared error
  - Prediction is the mean value of the samples in that partition



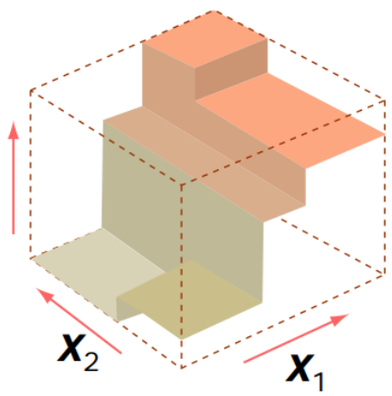
(a) General partition that cannot be obtained from recursive binary splitting.



(b) Partition of a two-dimensional feature space by recursive binary splitting, as used in CART, applied to some fake data.



(c) Tree corresponding to the partition in the top right panel.



(d) A perspective plot of the prediction surface.