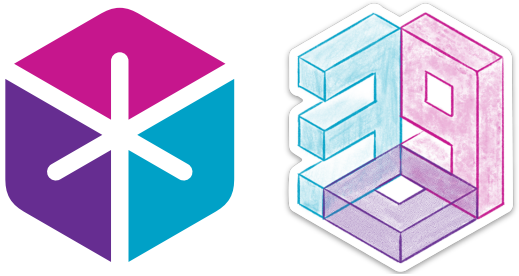


https://introml.mit.edu/

6.390 Intro to Machine Learning

Lecture 9: Non-parametric Models

Shen Shen

April 12, 2024

(many slides adapted from Tamara Broderick)

Outline

- Recap (transforermers)
- Non-parametric models
  - interpretability
  - ease of use/simplicity
- Decision tree
  - Terminologies
  - Learn via the BuildTree algorithm
    - Regression
    - Classification
- Nearest neighbor

Outline

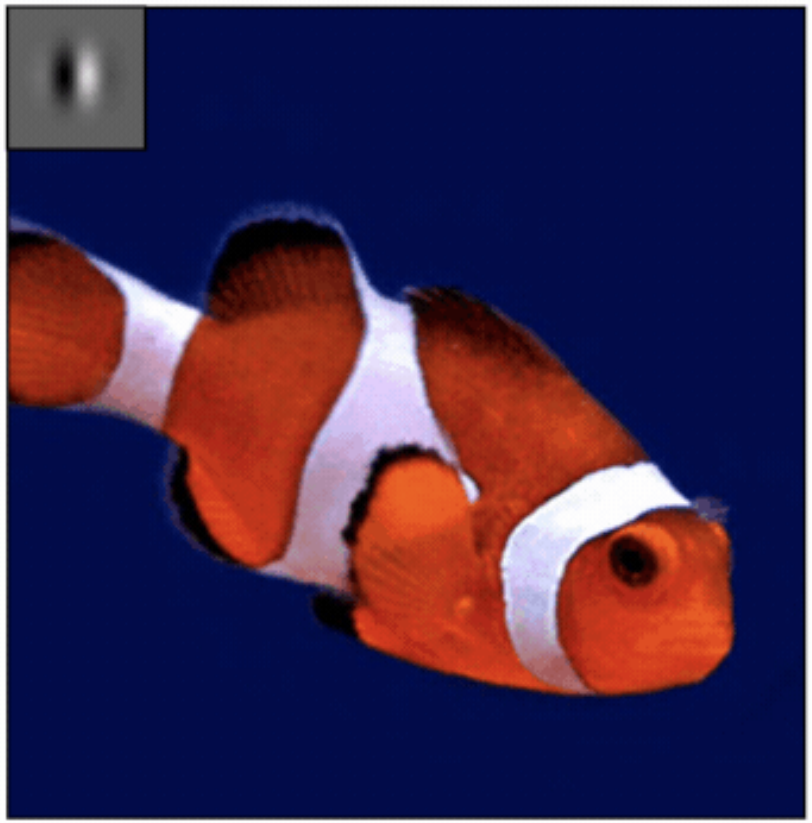
- Recap (transforermers)
- Non-parametric models
  - interpretability
  - ease of use/simplicity
- Decision tree
  - Terminologies
  - Learn via the BuildTree algorithm
    - Regression
    - Classification
- Nearest neighbor

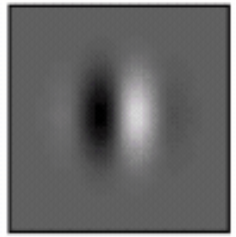
Lessons from CNNs

Enduring principles:

1. Chop up signal into patches (divide and conquer)
2. Process each patch **identically** (and in **parallel**)

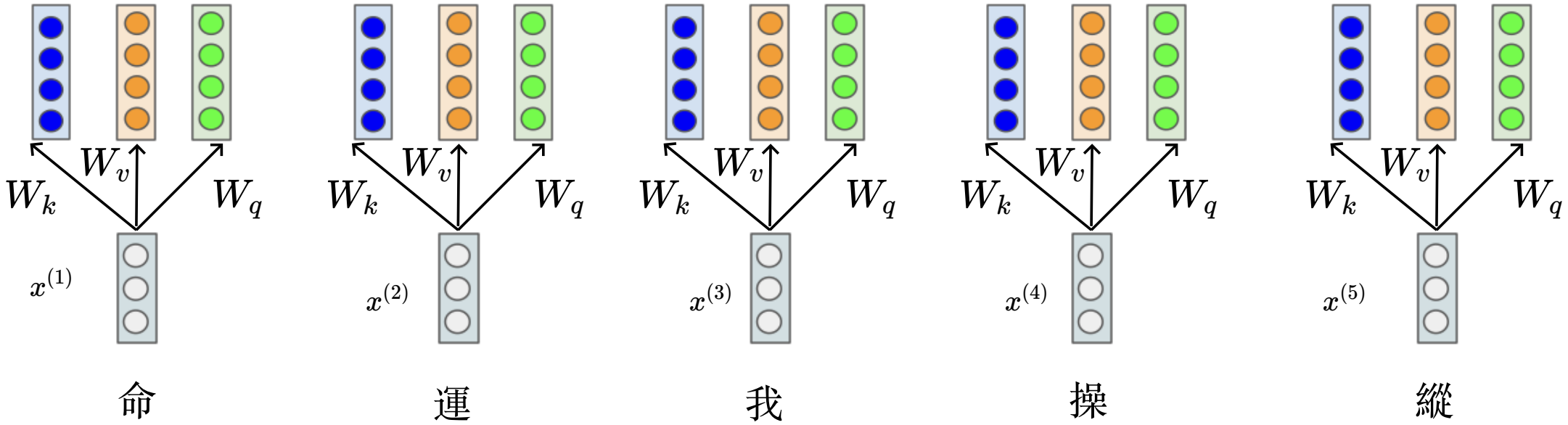
CNN





filter

Transformers



- Importantly, all these learned projection weights W are shared along the token sequence.
- Same "operation" repeated.

Interpretability

Giannis Daras ✈️ NeurIPS 2023
@giannis_daras

DALLE-2 has a secret language.
"Apoploe vesrreaitais" means birds.
"Contarra ccetnxniams luryca tanniounons" means bugs or pests.

The prompt: "Apoploe vesrreaitais eating Contarra ccetnxniams luryca tanniounons" gives images of birds eating bugs.

A thread (1/n)🧵

Another example: "Two whales talking about food, with subtitles". We get an image with the text "Wa ch zod rea" written on it. Apparently, the whales are actually talking about their food in the DALLE-2 language. (4/n)

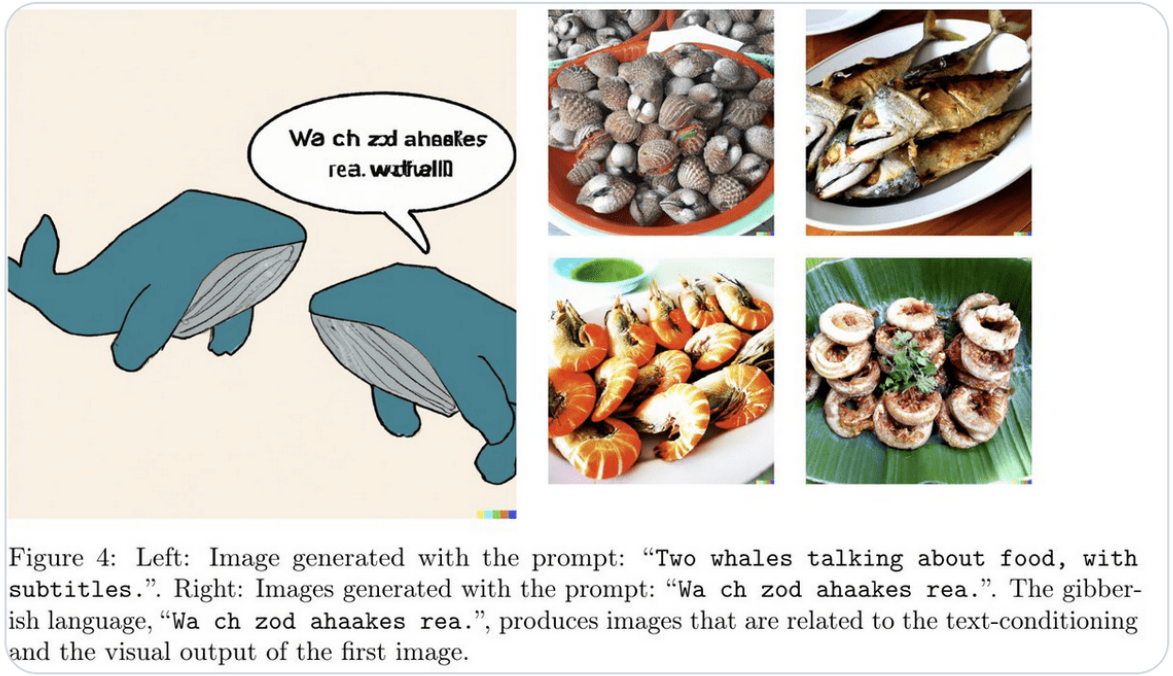


Figure 4: Left: Image generated with the prompt: "`Two whales talking about food, with subtitles.`". Right: Images generated with the prompt: "`Wa ch zod ahaakes rea.`". The gibberish language, "`Wa ch zod ahaakes rea.`", produces images that are related to the text-conditioning and the visual output of the first image.

Outline

- Recap (transforermers)
- Non-parametric models
  - interpretability
  - ease of use/simplicity
- Decision tree
  - Terminologies
  - Learn via the BuildTree algorithm
    - Regression
    - Classification
- Nearest neighbor

Non-parametric models

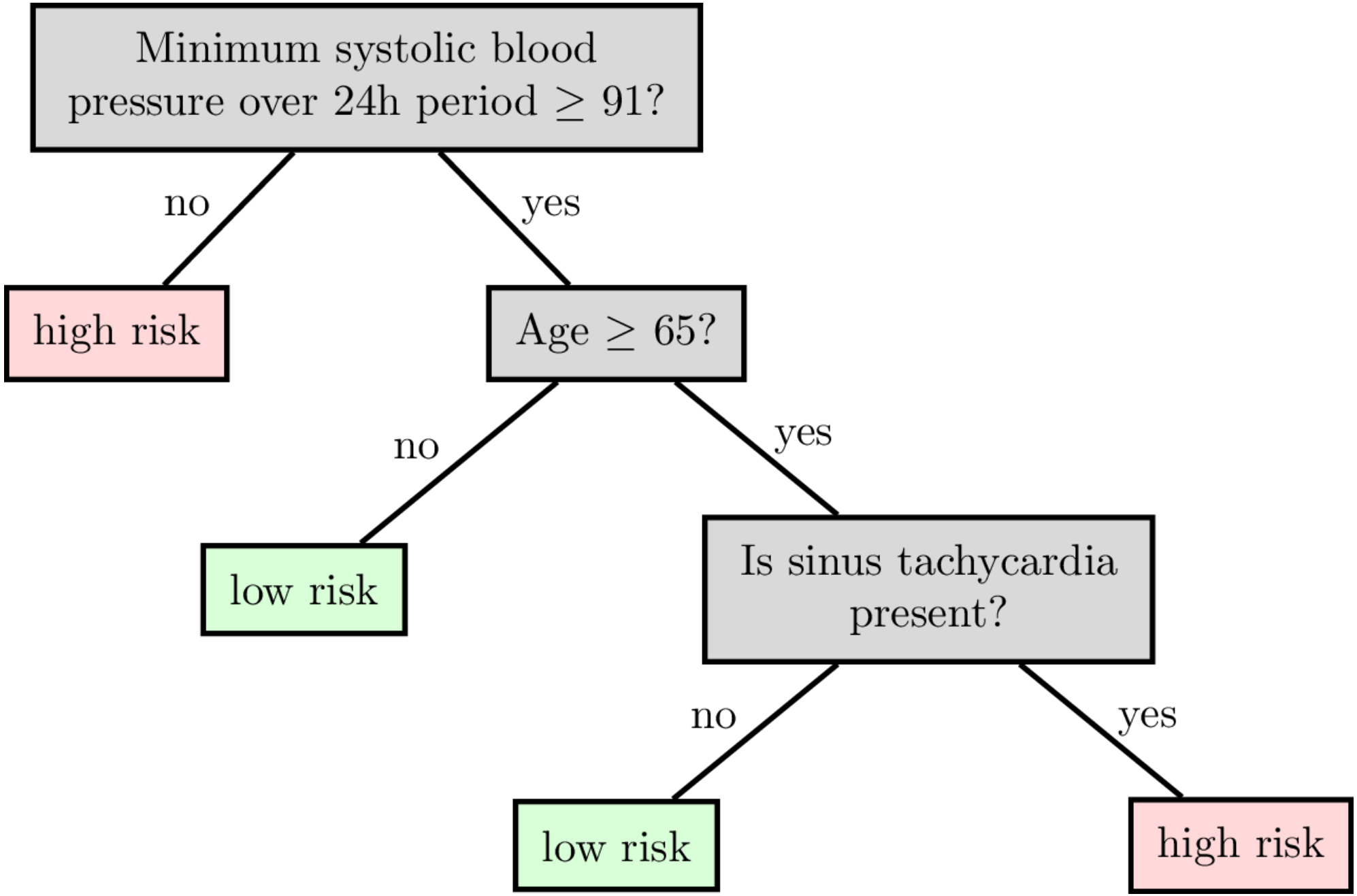
- does not mean "no parameters"
- there are still parameters to be learned to build a hypothesis/model.
- just that, the model/hypothesis does not have a fixed parameterization.
- (e.g. even the number of parameters can change.)

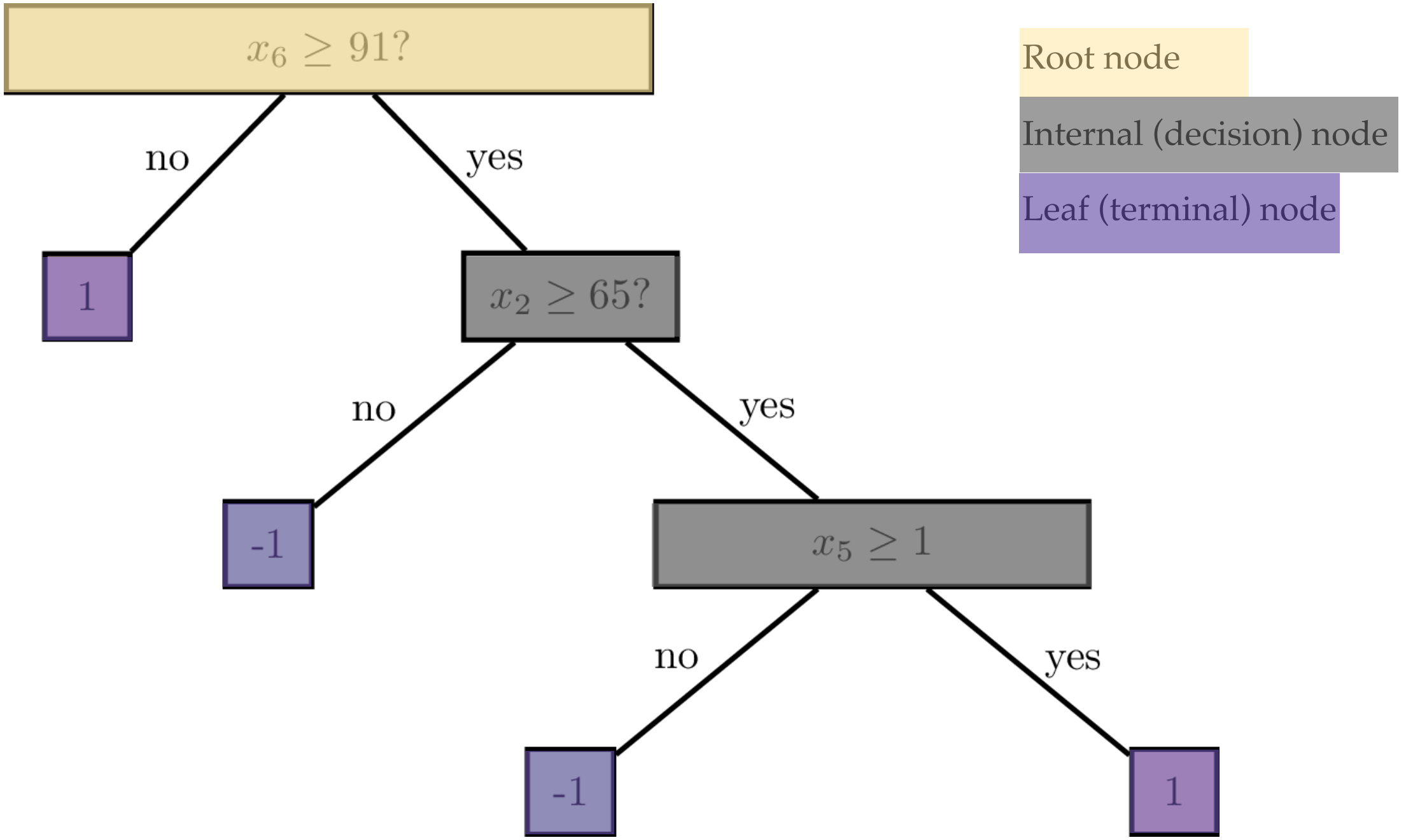
- Decision trees and
- Nearest neighbor

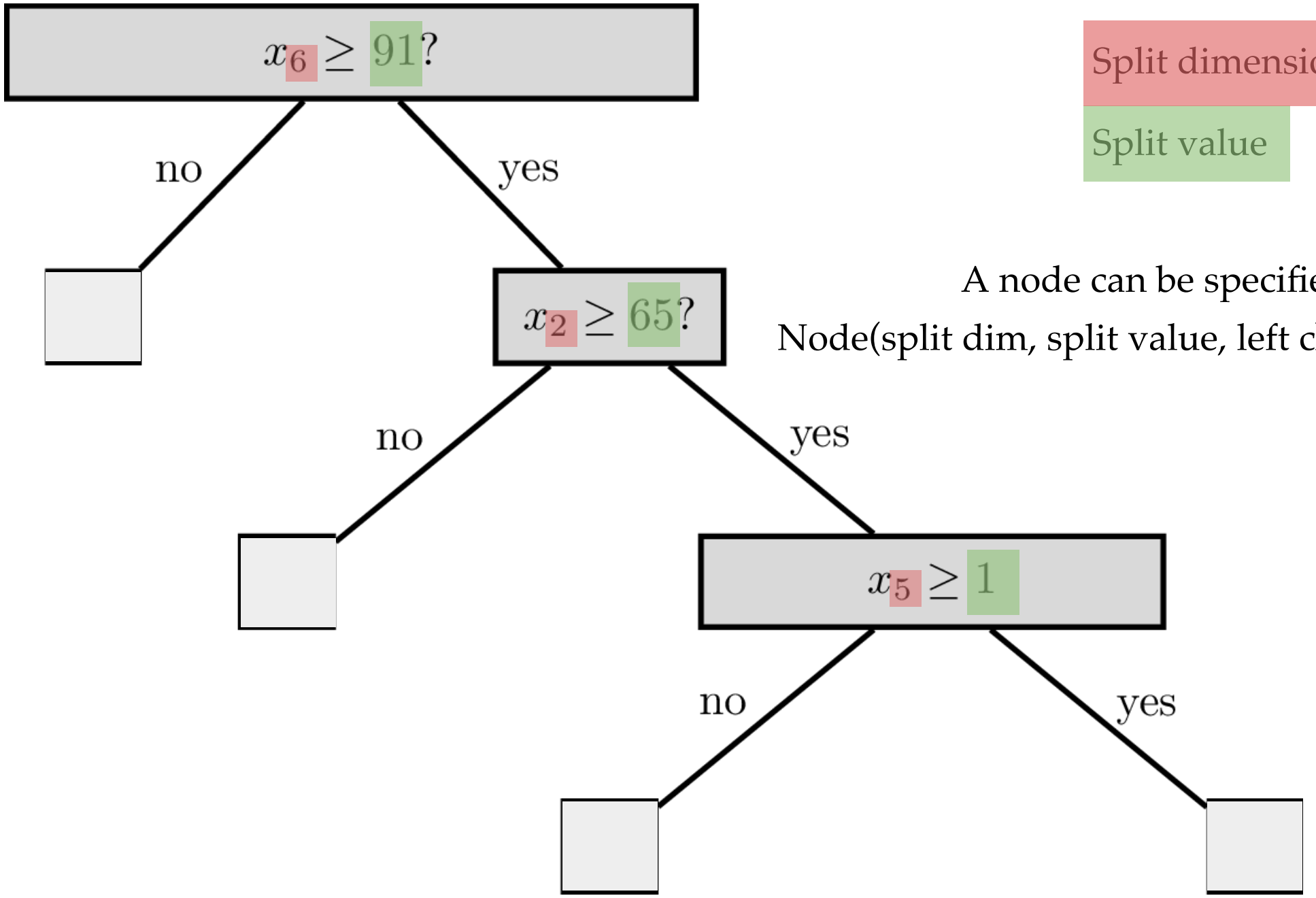
are the classical examples of non-parametric models

Outline

- Recap (transforermers)
- Non-parametric models
  - interpretability
  - ease of use/simplicity
- Decision tree
  - Terminologies
  - Learn via the BuildTree algorithm
    - Regression
    - Classification
- Nearest neighbor


Minimum systolic blood pressure over 24h period ≥ 91?
no
yes
high risk
Age ≥ 65?
no
yes
low risk
Is sinus tachycardia present?
no
yes
low risk
high risk


$x_6 \geq 91$?
no
yes
1
$x_2 \geq 65$?
no
yes
-1
$x_5 \geq 1$
no
yes
-1
1
Root node
Internal (decision) node
Leaf (terminal) node

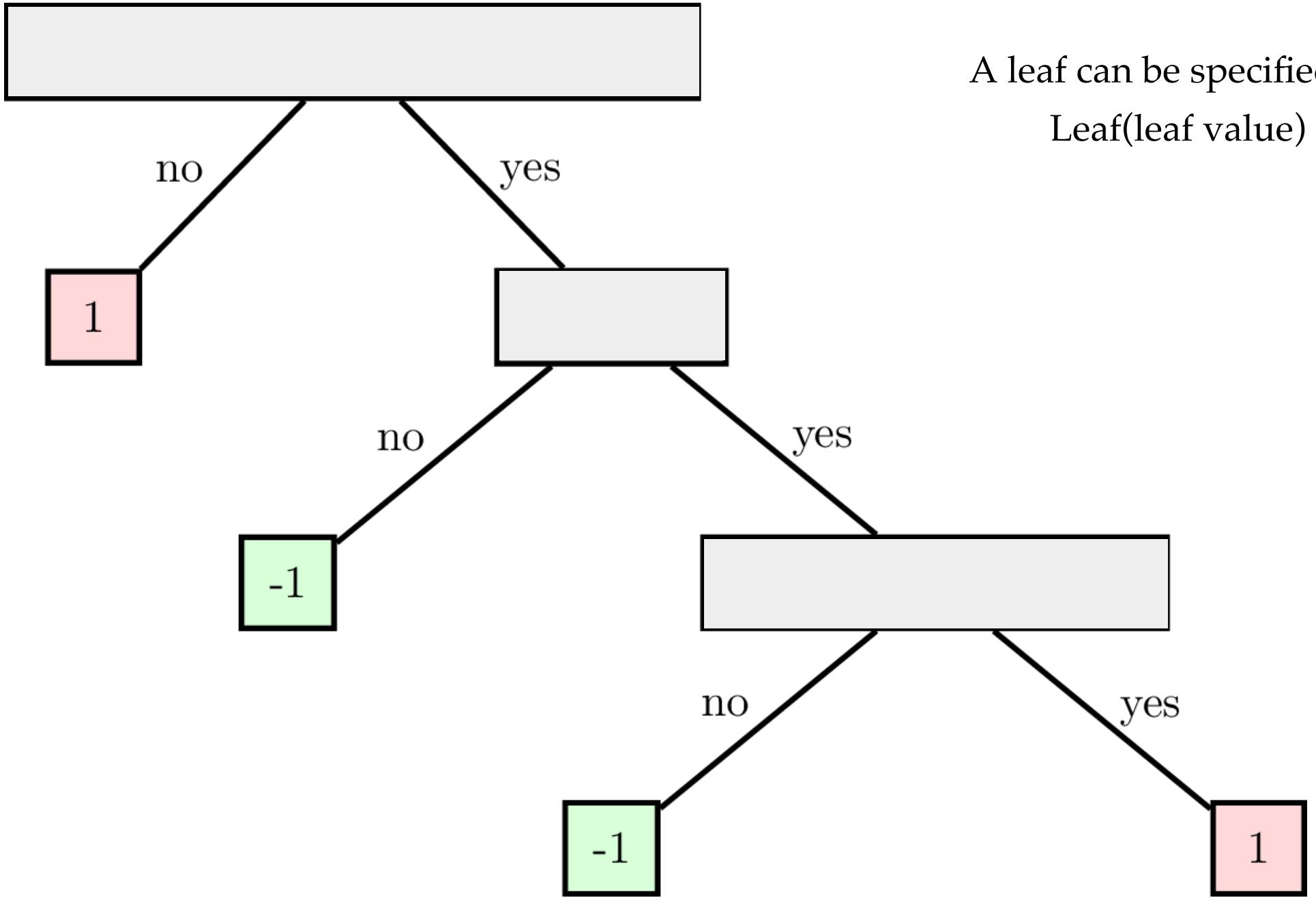


Split dimension

Split value

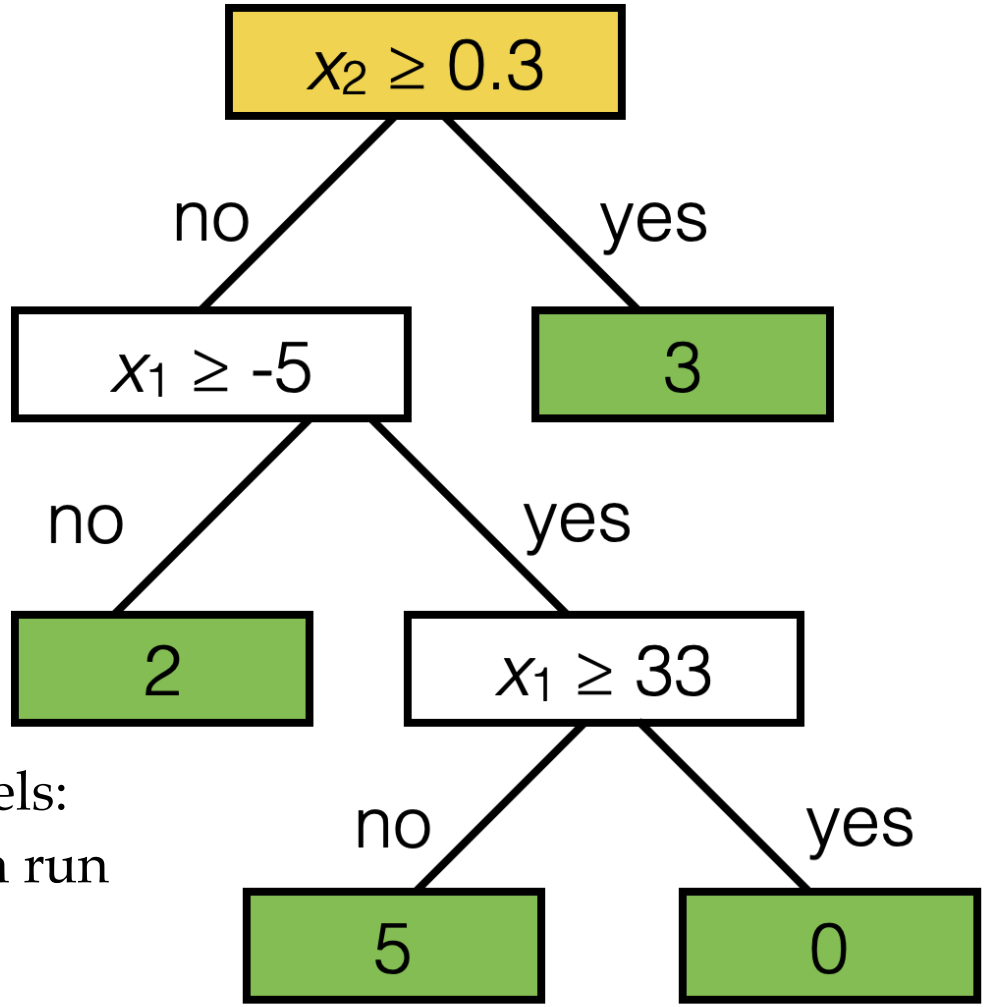
A node can be specified by

Node(split dim, split value, left child, right child)



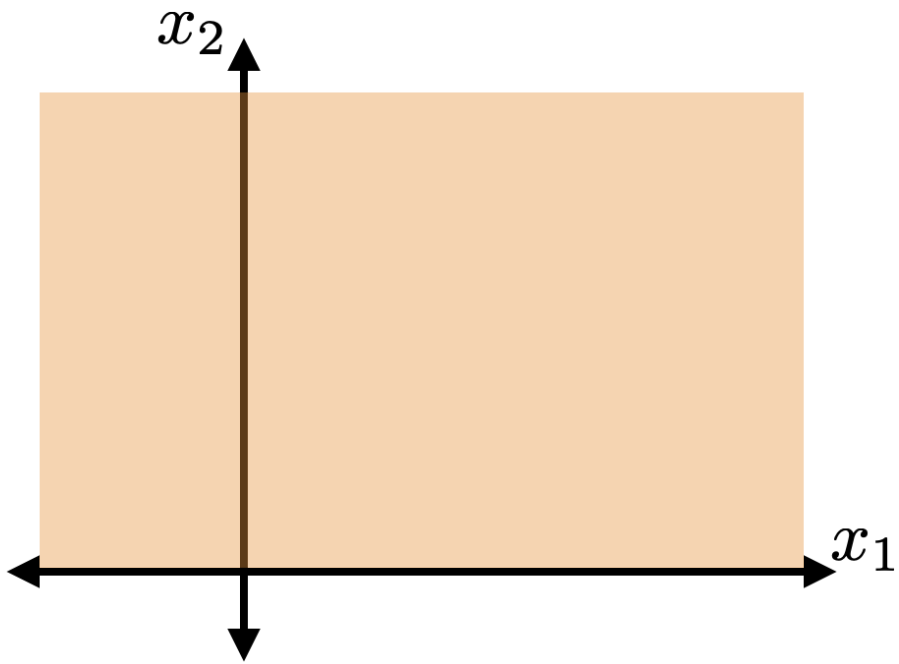
A leaf can be specified by

Leaf(leaf value)



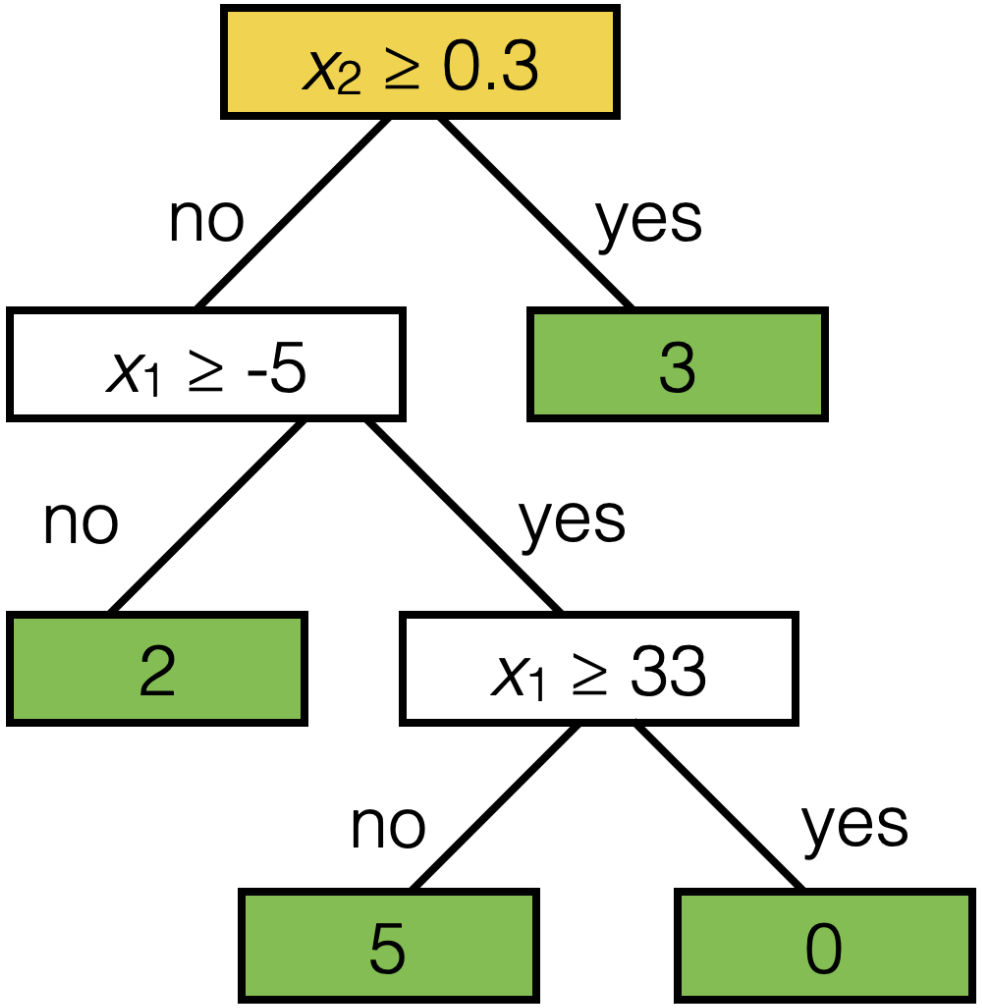
labels:
y: km run

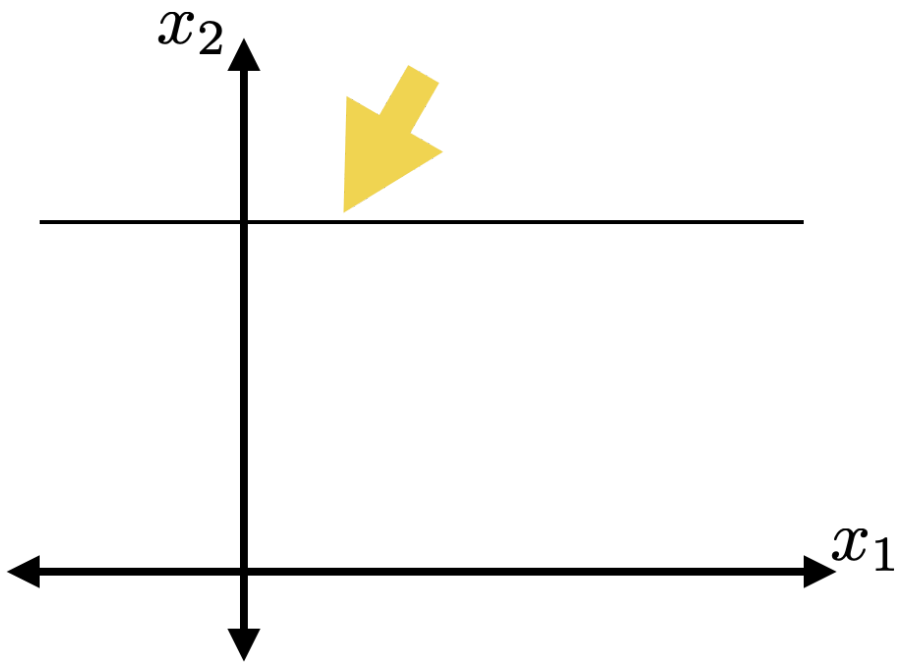
Tree defines an axis-aligned "partition" of the feature space:

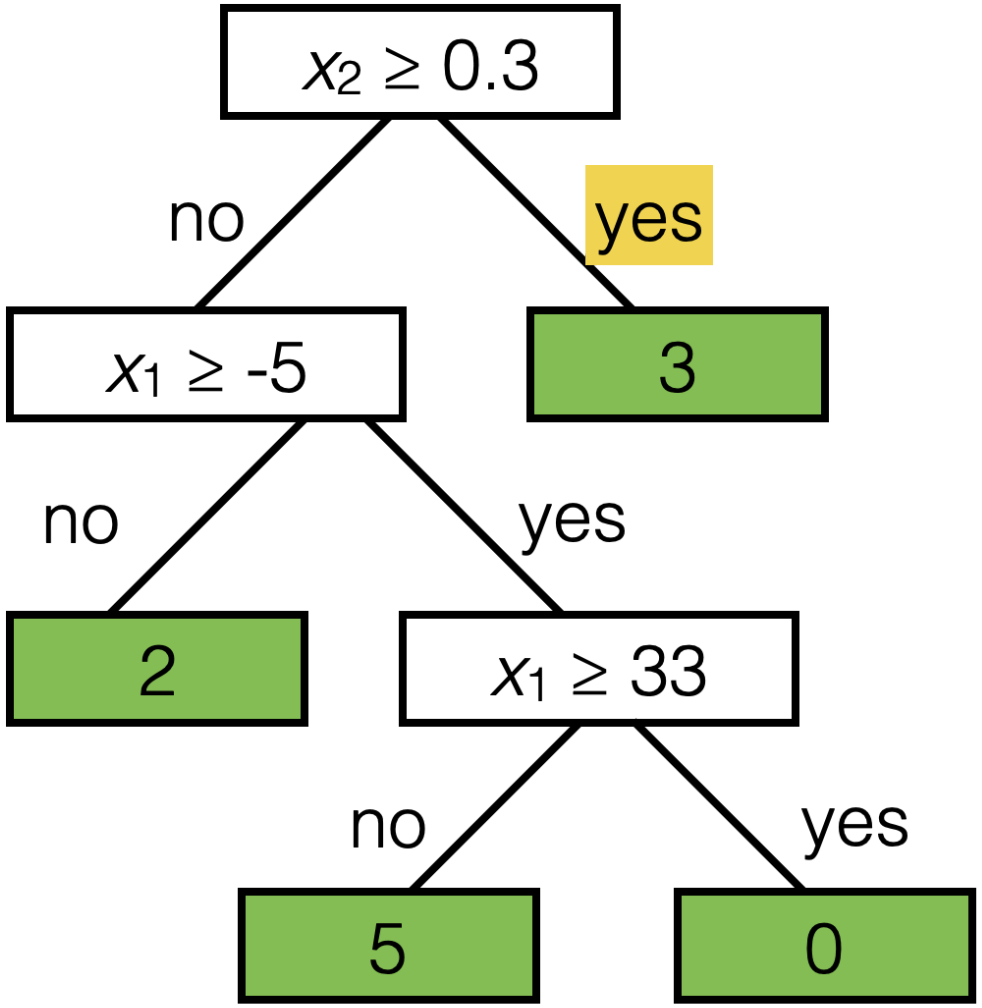


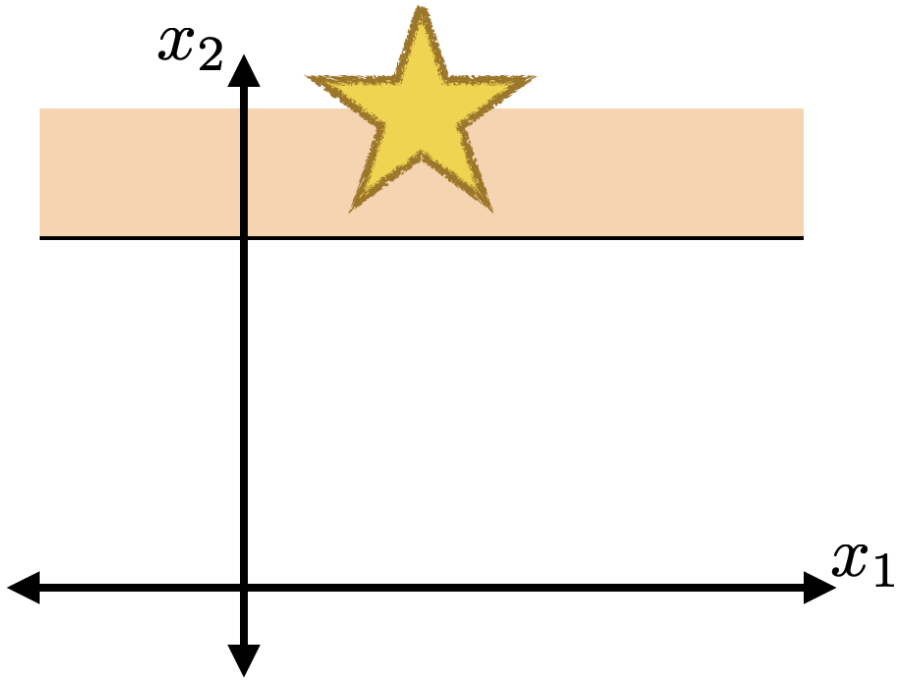
features:

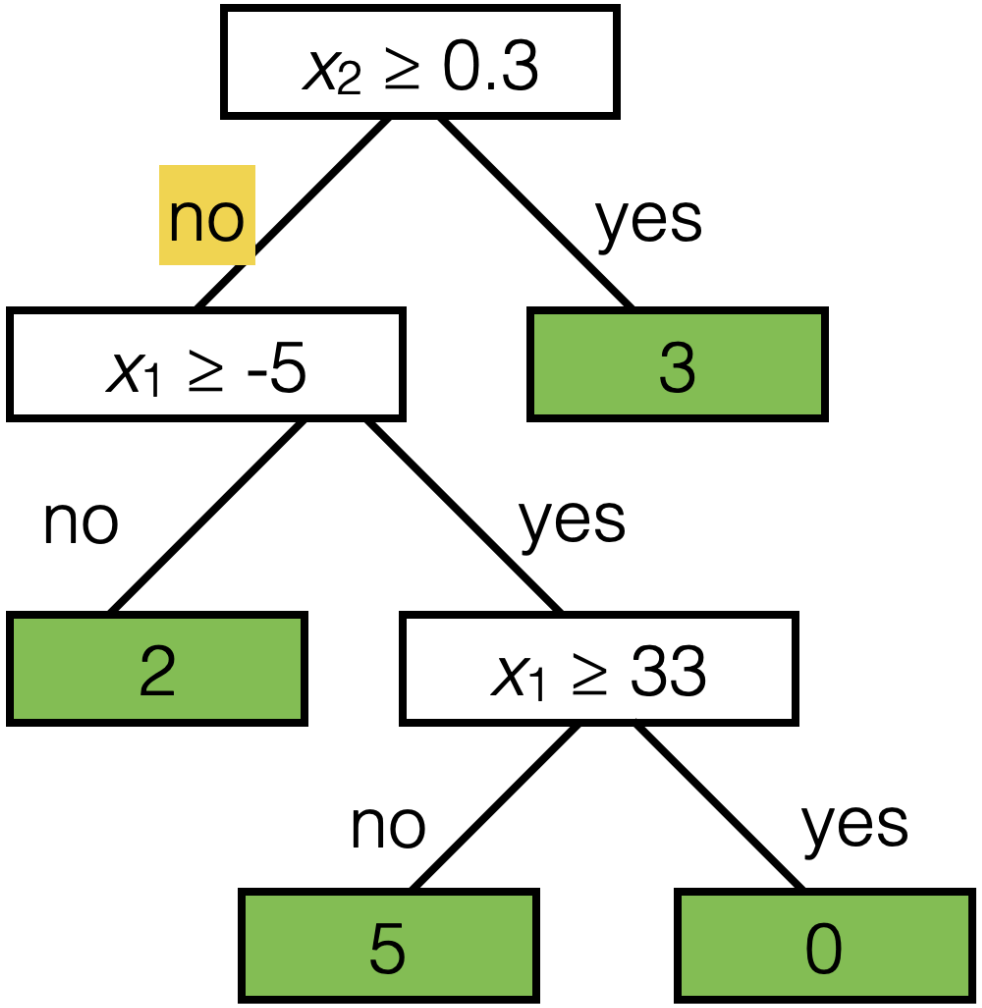
- x_1: temperature (deg C)
- x_2: precipitation (cm/hr)

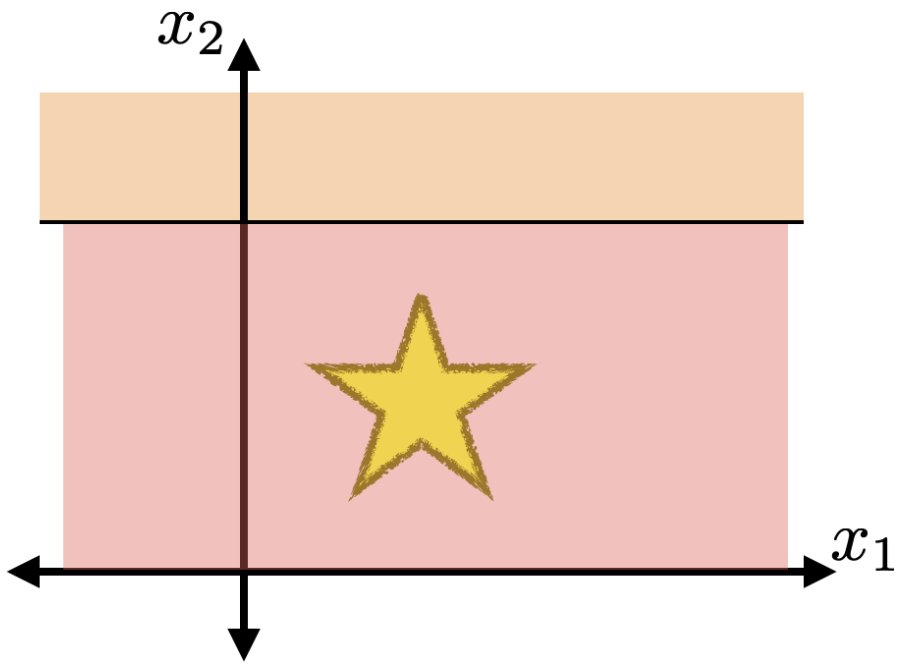

$x_2 \geq 0.3$
no
yes
$x_1 \geq -5$
3
no
yes
2
$x_1 \geq 33$
no
yes
5
0

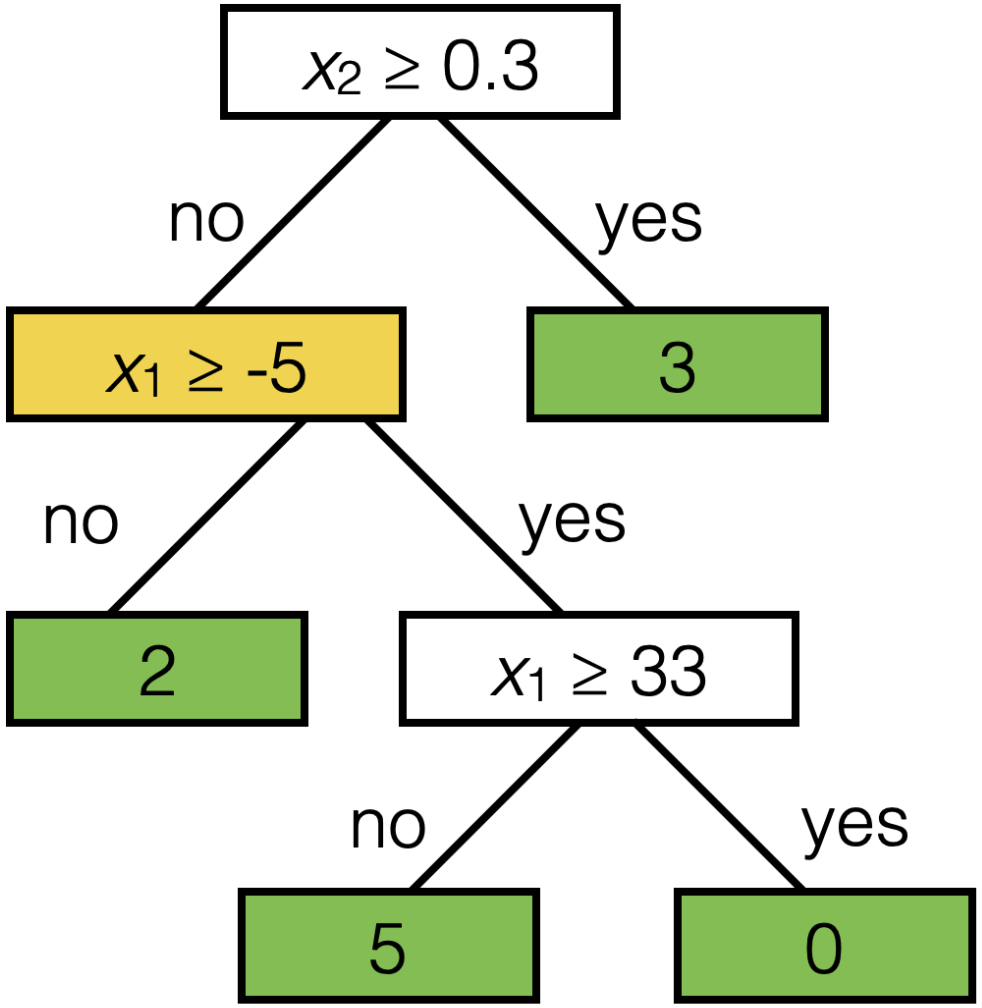

x_2
x_1

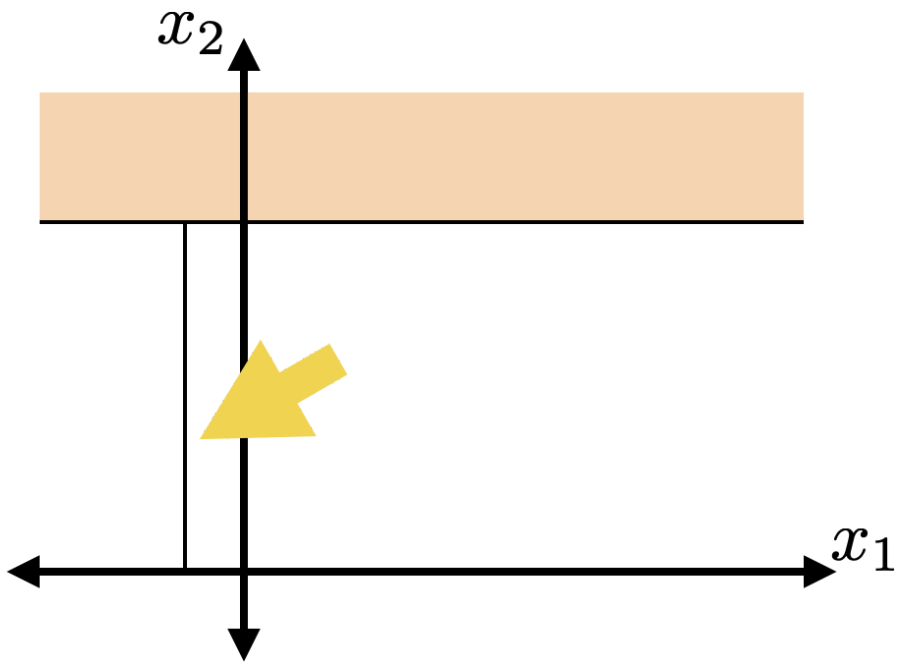

$x_2 \geq 0.3$
no
yes
$x_1 \geq -5$
3
no
yes
2
$x_1 \geq 33$
no
yes
5
0

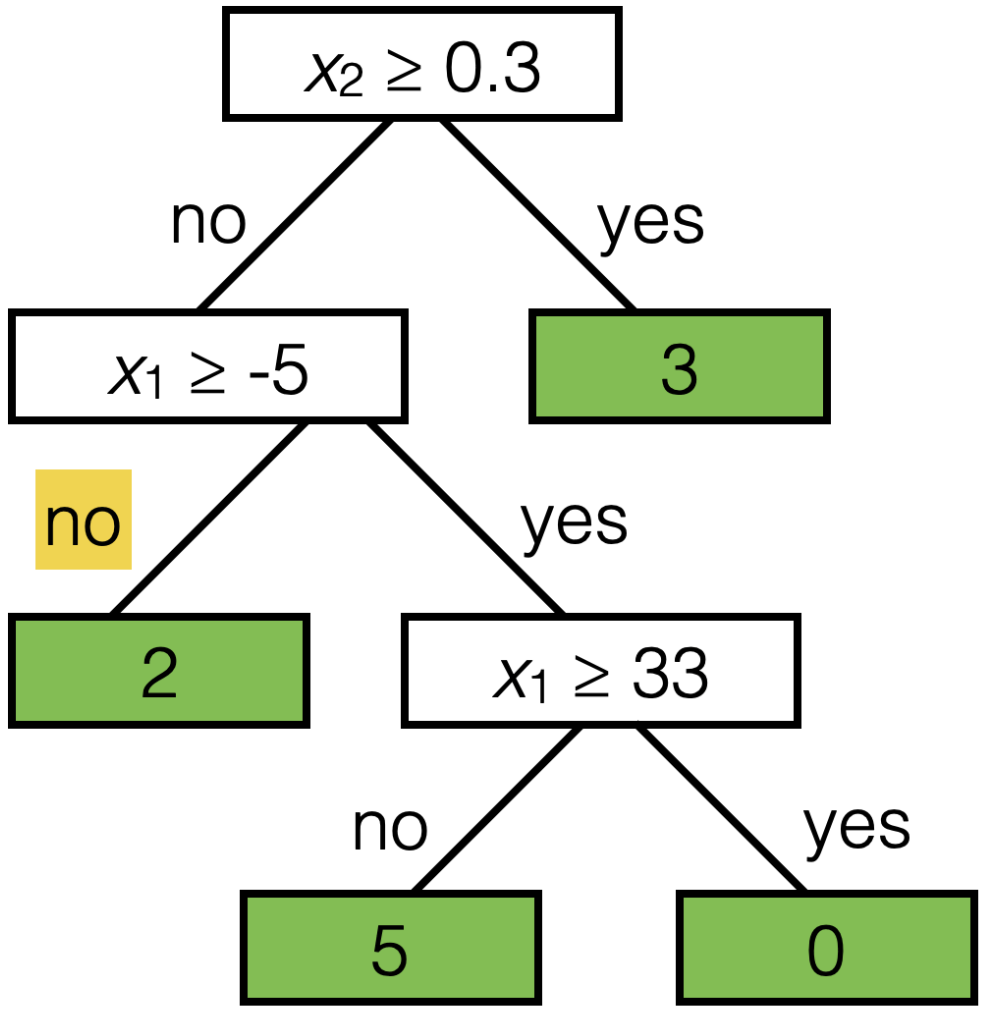

x_2
x_1

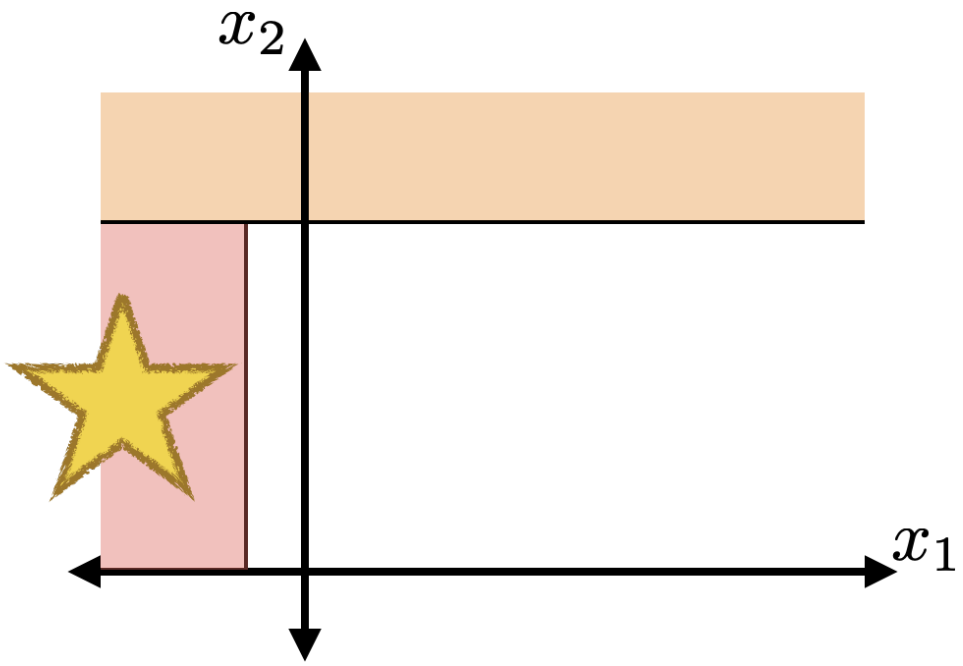

$x_2 \geq 0.3$
no
yes
$x_1 \geq -5$
3
no
yes
2
$x_1 \geq 33$
no
yes
5
0

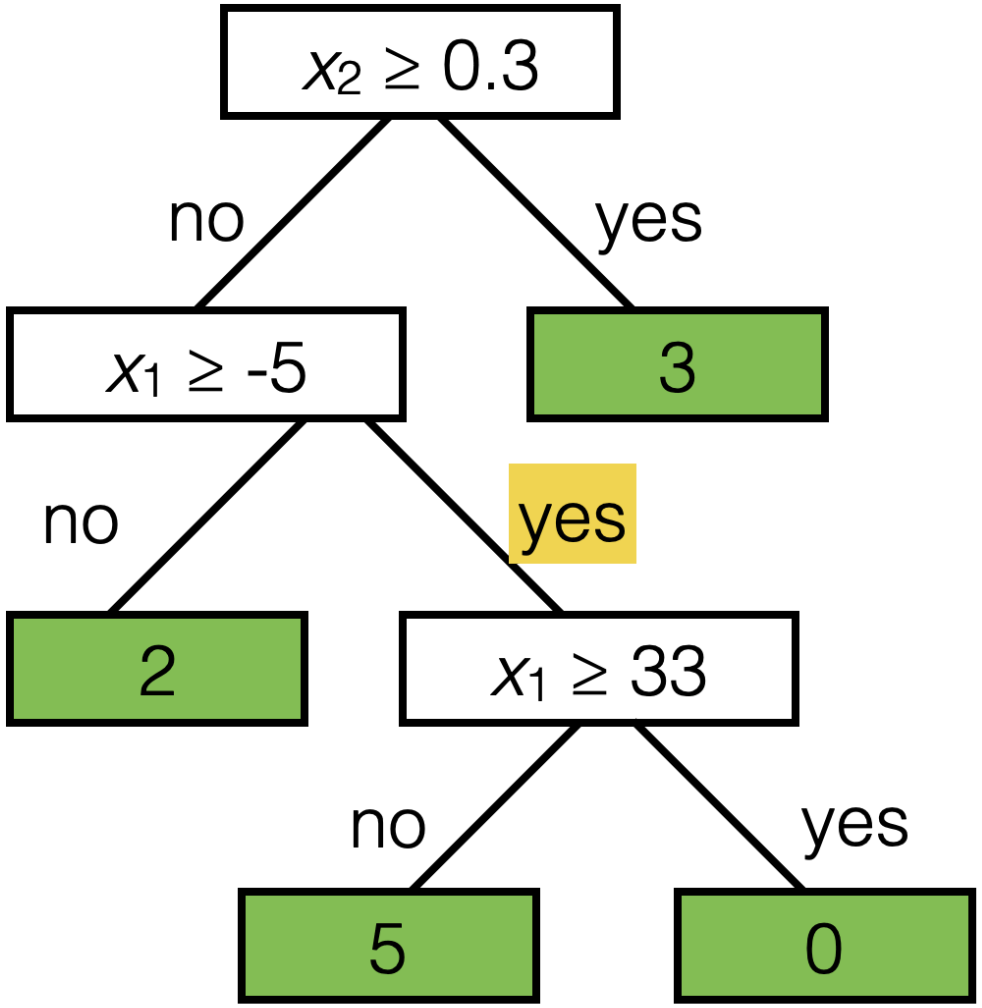

x_2
x_1

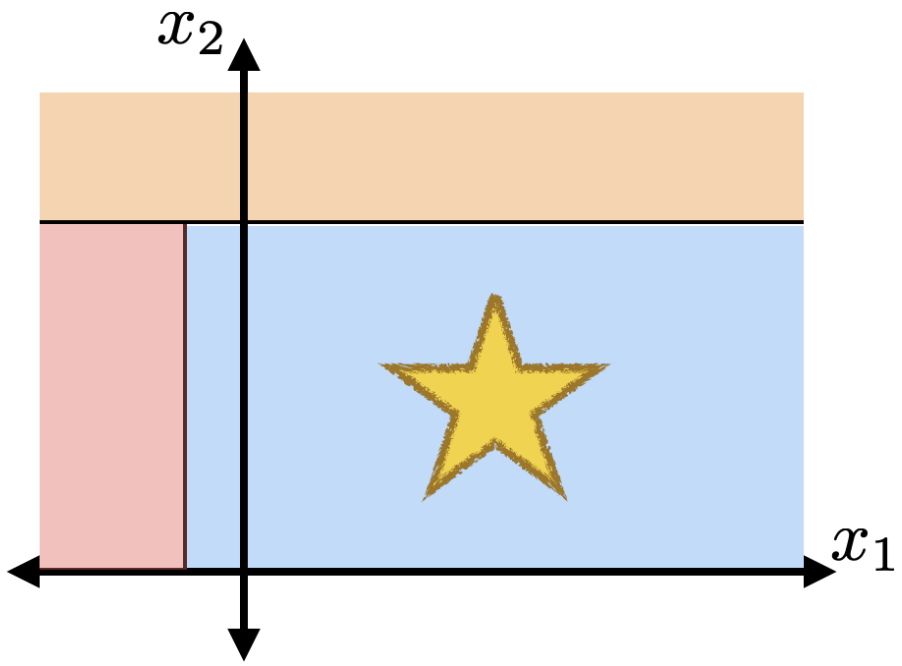

$x_2 \geq 0.3$
no
yes
$x_1 \geq -5$
3
no
yes
2
$x_1 \geq 33$
no
yes
5
0

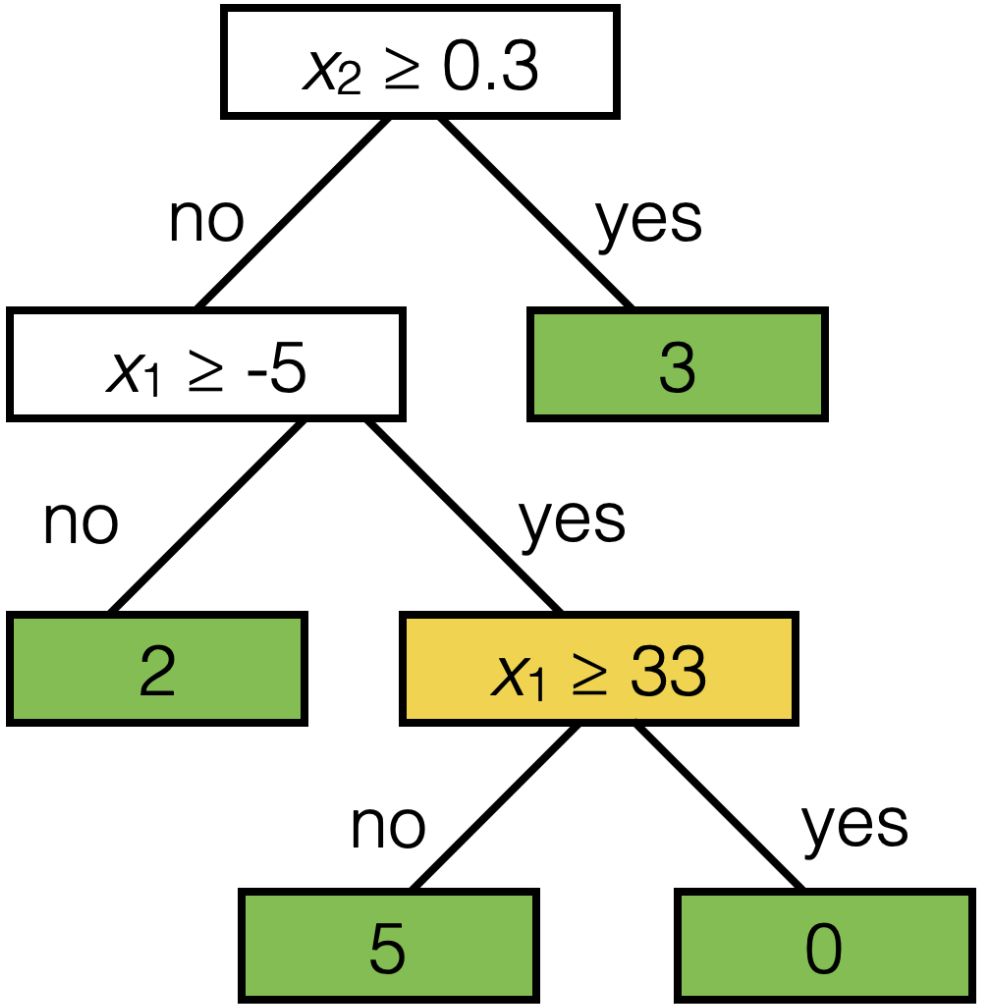

x_2
x_1

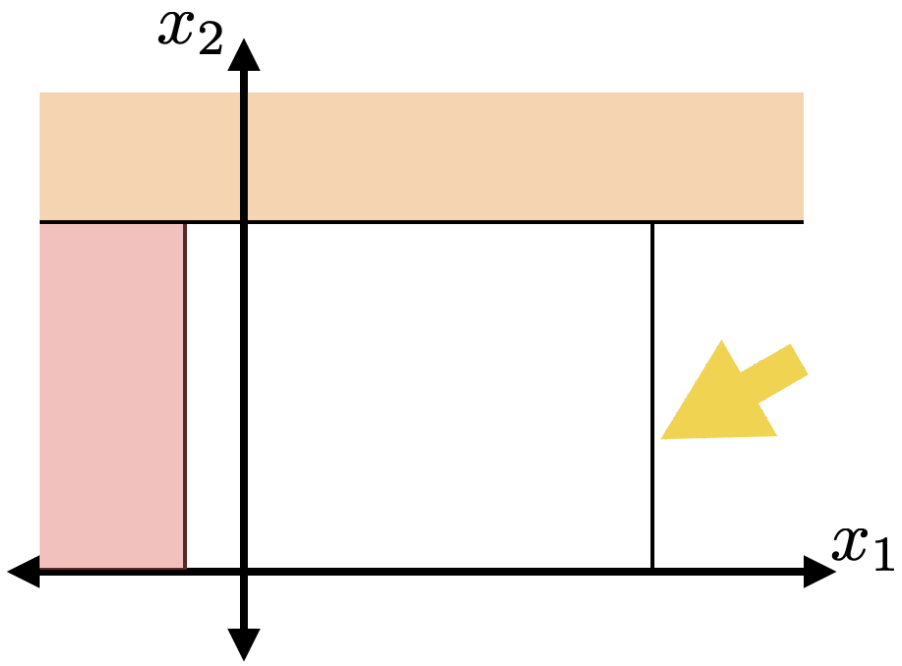

$x_2 \geq 0.3$
no
yes
$x_1 \geq -5$
3
no
yes
2
$x_1 \geq 33$
no
yes
5
0

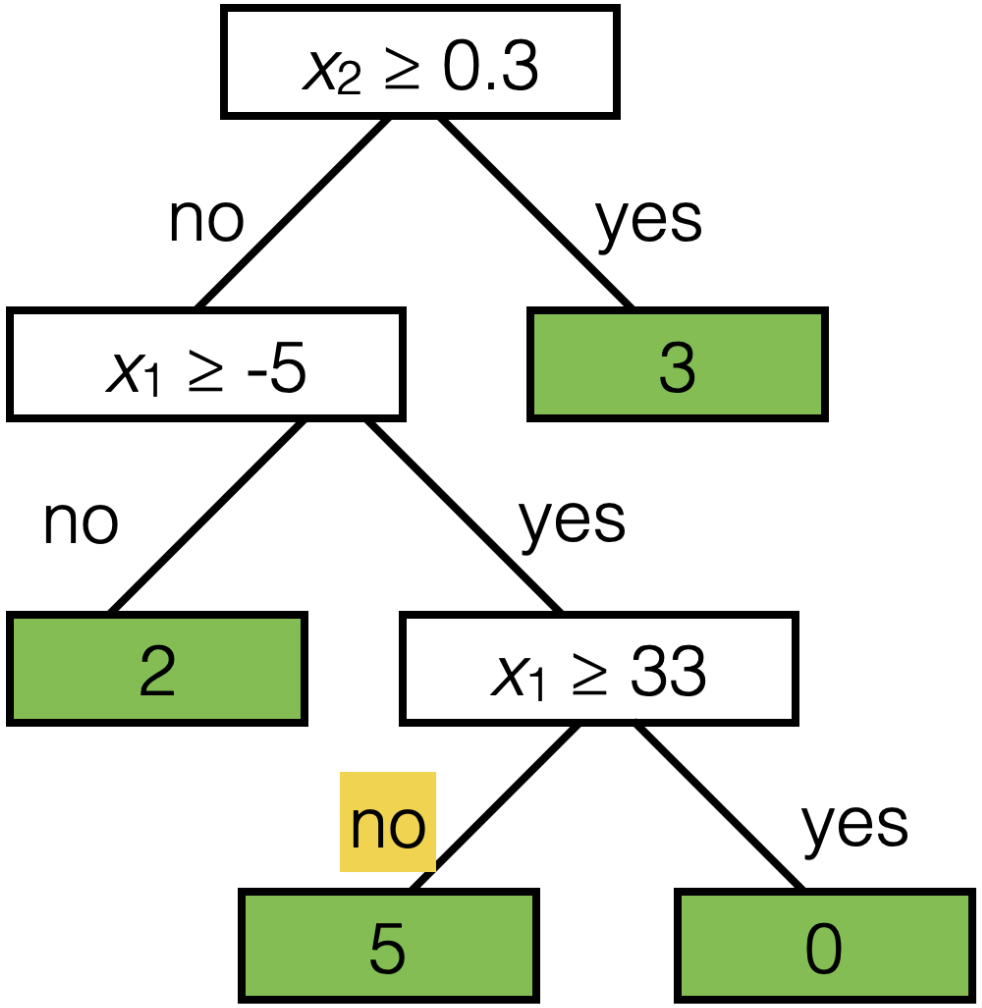

x_2
x_1

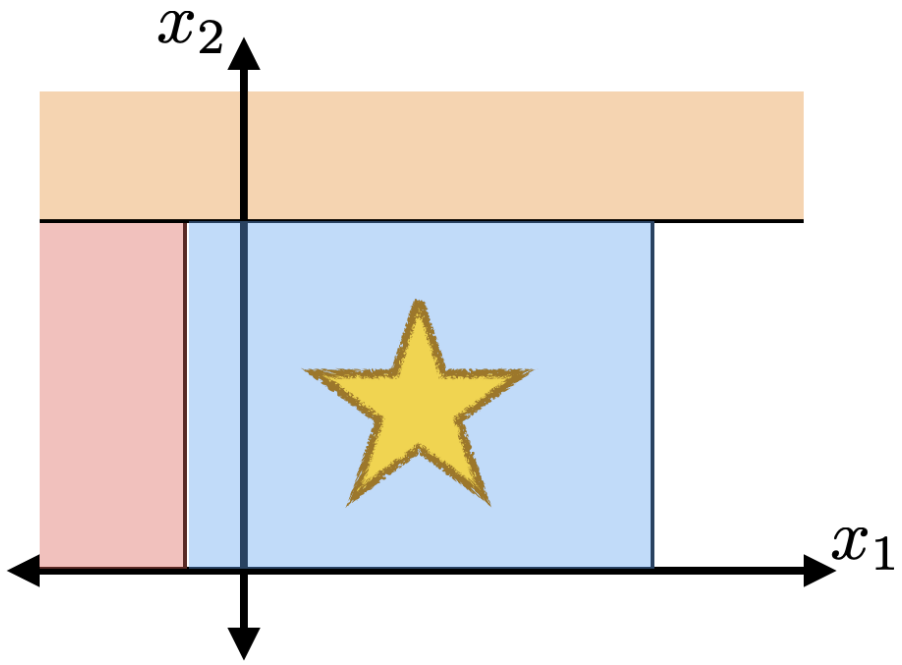

$x_2 \geq 0.3$
no
yes
$x_1 \geq -5$
3
no
yes
2
$x_1 \geq 33$
no
yes
5
0

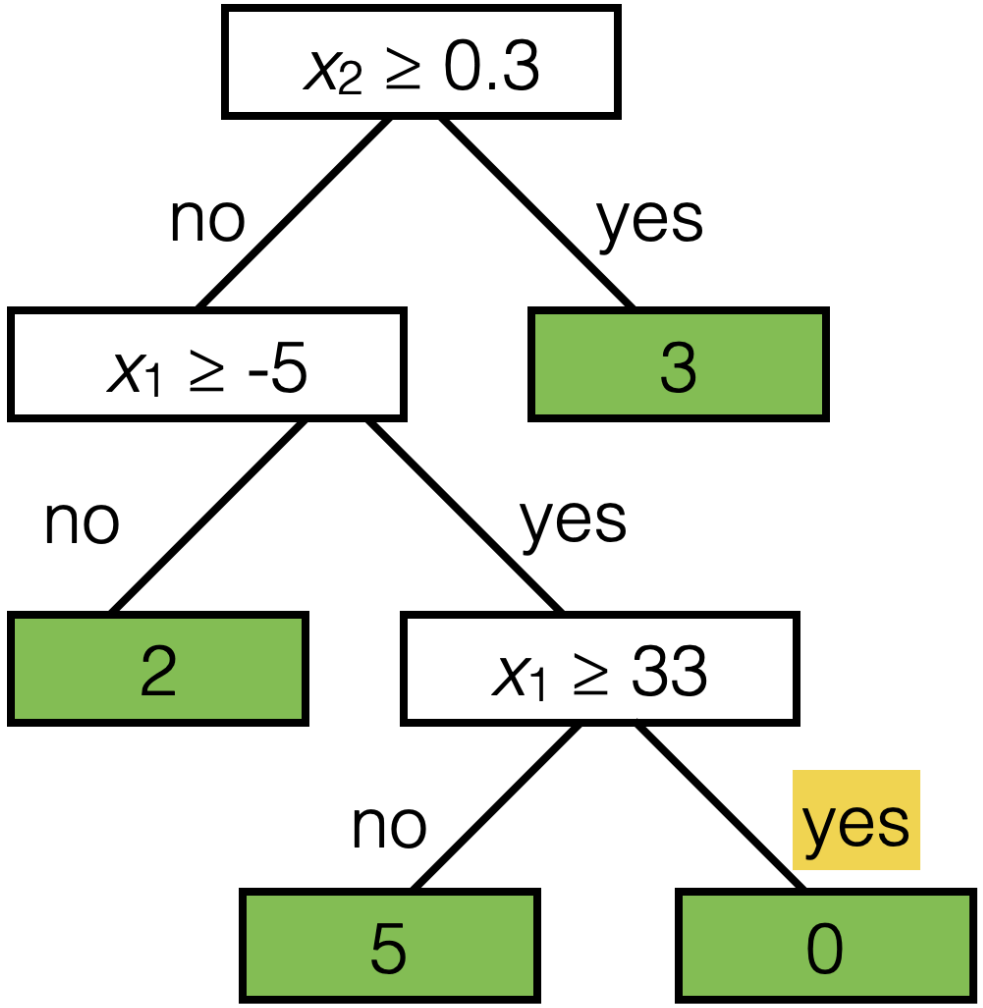

x_2
x_1

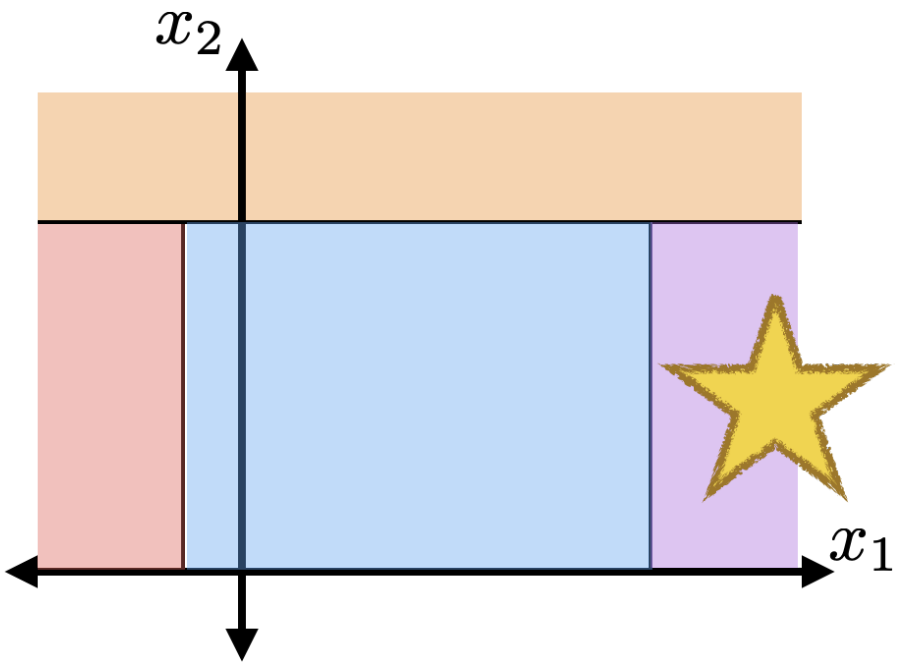

$x_2 \geq 0.3$
no
yes
$x_1 \geq -5$
3
no
yes
2
$x_1 \geq 33$
no
yes
5
0

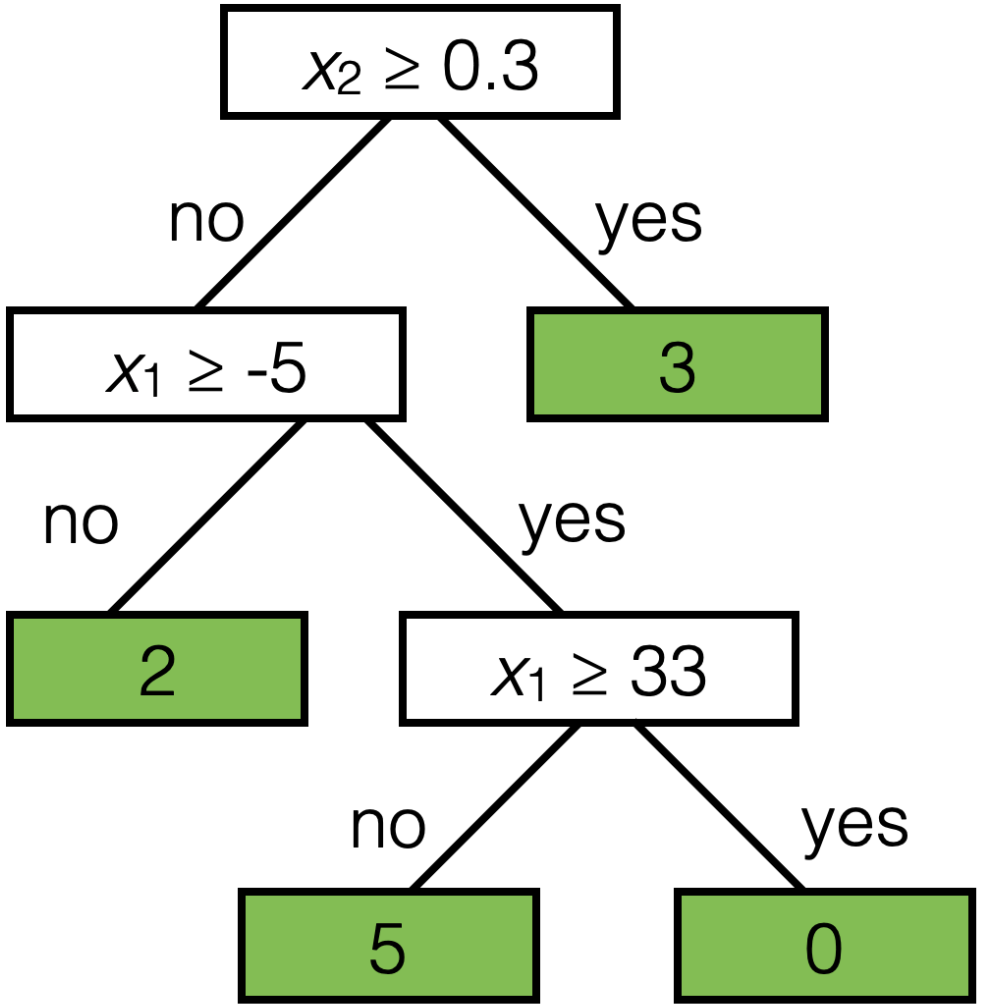

x_2
x_1

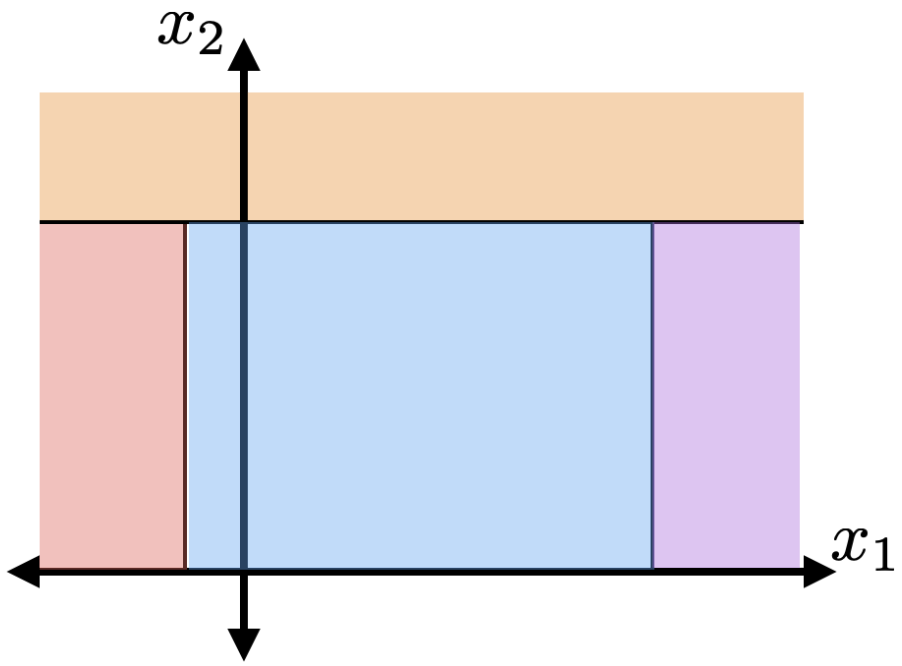

$x_2 \geq 0.3$
no
yes
$x_1 \geq -5$
3
no
yes
2
$x_1 \geq 33$
no
yes
5
0

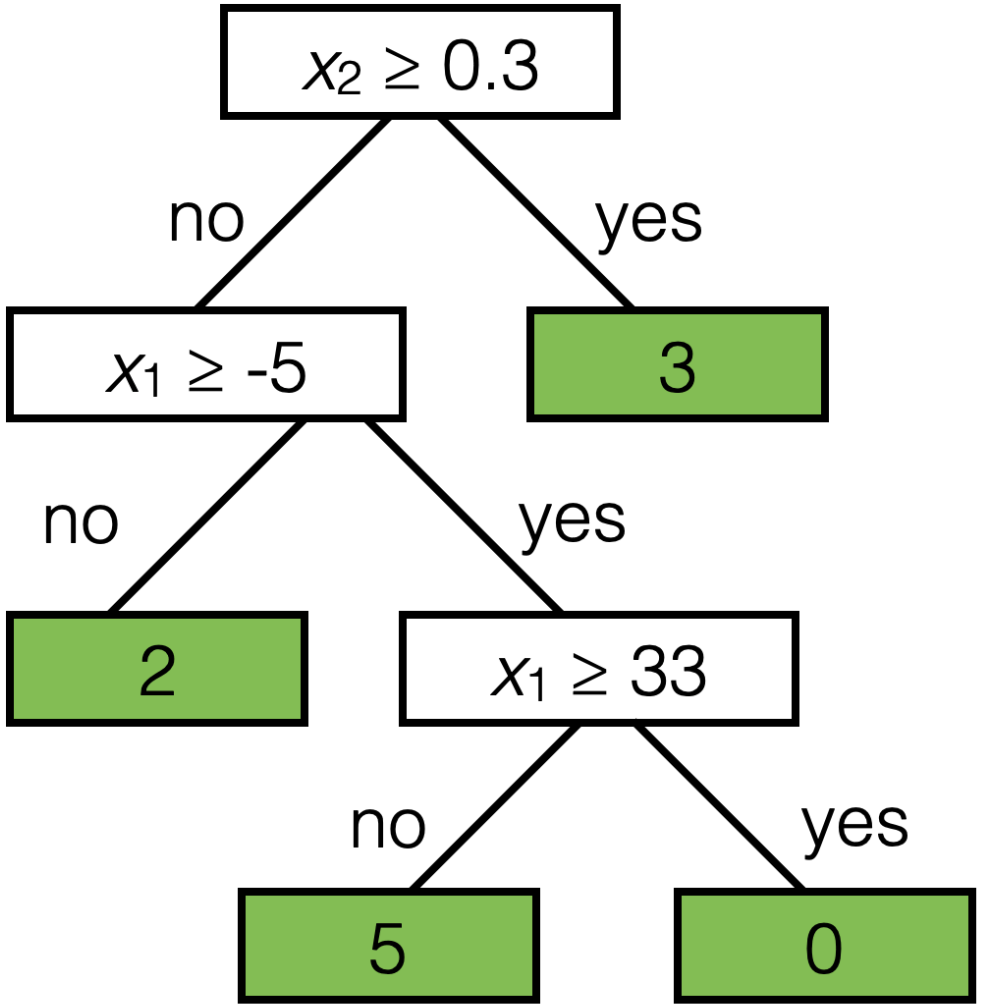

x_2
x_1

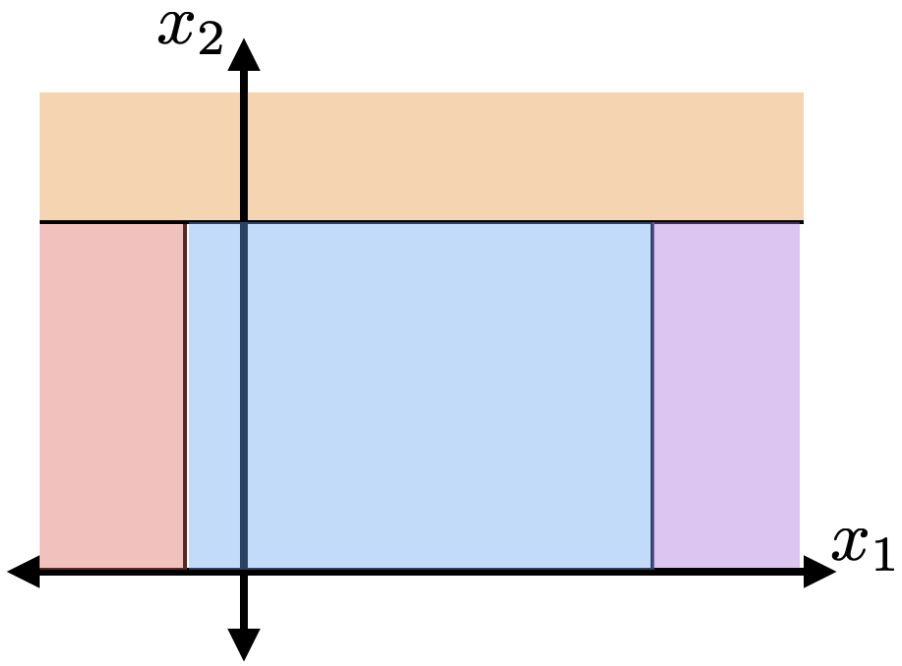

$x_2 \geq 0.3$
no
yes
$x_1 \geq -5$
3
no
yes
2
$x_1 \geq 33$
no
yes
5
0

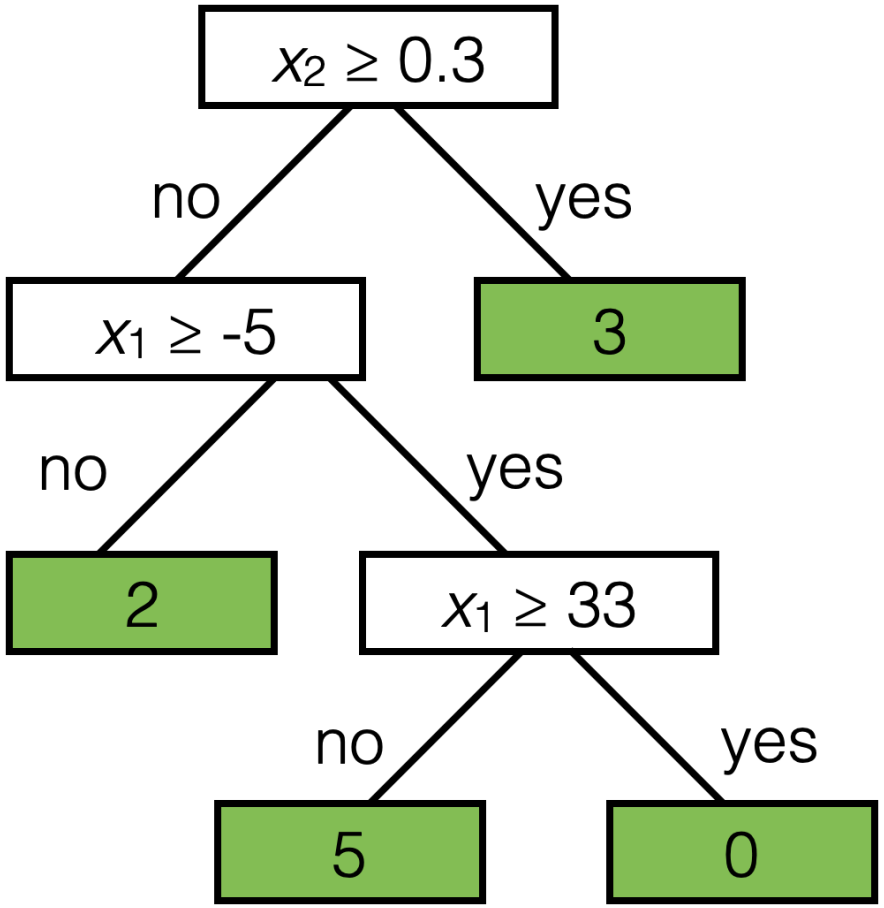

x_2
x_1


$x_2 \geq 0.3$
no
yes
$x_1 \geq -5$
3
no
yes
2
$x_1 \geq 33$
no
yes
5
0


x_2
x_1


$x_2 \geq 0.3$
no
yes
$x_1 \geq -5$
3
no
yes
2
$x_1 \geq 33$
no
yes
5
0


x_2
x_1



How to learn a tree?

Recall: familiar "recipe"

1. Choose how to predict label (given features & parameters)
2. Choose a loss (between guess & actual label)
3. Choose parameters by trying to minimize the training loss

Here, we need:

- For each internal node:
  - split dimension
  - split value
  - child nodes
- For each leaf node:
  - label

BuildTree$(I; k)$

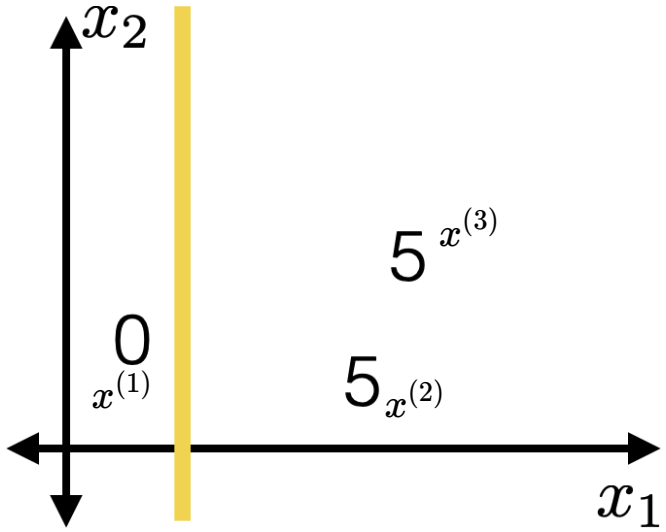
1. **if** $|I| > k$
2. **for** each split dim j and split value s
3. Set $I^+_{j,s} = \left\{ i \in I \mid x^{(i)}_j \geq s \right\}$
4. Set $I^-_{j,s} = \left\{ i \in I \mid x^{(i)}_j < s \right\}$
5. Set $\hat{y}^+_{j,s} = \text{average}_{\ i \in I^+_{j,s}}\ y^{(i)}$
6. Set $\hat{y}^-_{j,s} = \text{average}_{\ i \in I^-_{j,s}}\ y^{(i)}$
7. Set $E_{j,s} = \sum_{i \in I^+_{j,s}} \left(y^{(i)} - \hat{y}^+_{j,s} \right)^2 + \sum_{i \in I^-_{j,s}} \left(y^{(i)} - \hat{y}^-_{j,s} \right)^2$
8. Set $(j^*, s^*) = \arg\min_{j,s} E_{j,s}$
9. **else**
10. Set $\hat{y} = \text{average}_{\ i \in I}\ y^{(i)}$
11. **return** LEAF(leave_value=$\hat{y}$)
12. **return** Node $\left(j^*, s^*, \text{BuildTree}\left(I^-_{j^*,s^*}, k \right), \text{BuildTree}\left(I^+_{j^*,s^*}, k \right) \right)$

- input I: set of indices
- k: hyper-parameter, maximum leaf "size", i.e. how many training data ended in that leaf node.
- $\hat{y}$: (intermediate) prediction

- j: split dimension
- s: split value

BuildTree$(I; k)$

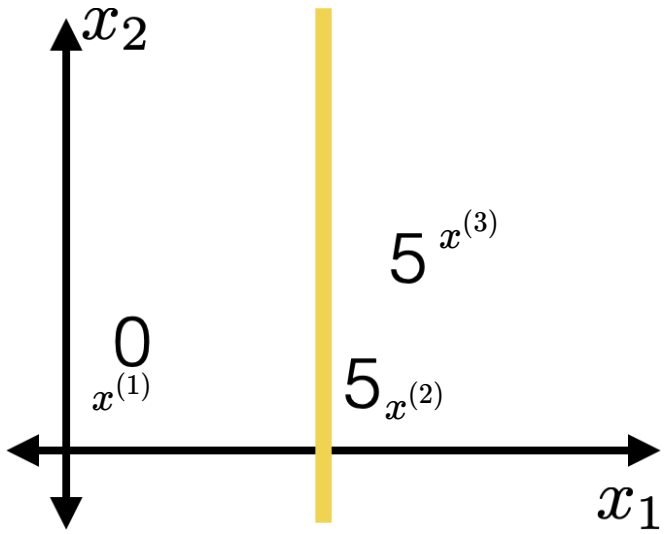
1. **if** $|I| > k$
2. **for** each split dim j and split value s
3. Set $I^+_{j,s} = \left\{ i \in I \mid x^{(i)}_j \geq s \right\}$
4. Set $I^-_{j,s} = \left\{ i \in I \mid x^{(i)}_j < s \right\}$
5. Set $\hat{y}^+_{j,s} = \text{average}_{\ i \in I^+_{j,s}}\ y^{(i)}$
6. Set $\hat{y}^-_{j,s} = \text{average}_{\ i \in I^-_{j,s}}\ y^{(i)}$
7. Set $E_{j,s} = \sum_{i \in I^+_{j,s}} \left(y^{(i)} - \hat{y}^+_{j,s} \right)^2 + \sum_{i \in I^-_{j,s}} \left(y^{(i)} - \hat{y}^-_{j,s} \right)^2$
8. Set $(j^*, s^*) = \arg\min_{j,s} E_{j,s}$
9. **else**
10. Set $\hat{y} = \text{average}_{\ i \in I}\ y^{(i)}$
11. **return** LEAF(leave_value=$\hat{y}$)
12. **return** Node $\left(j^*, s^*, \text{BuildTree}\left(I^-_{j^*,s^*}, k\right), \text{BuildTree}\left(I^+_{j^*,s^*}, k\right)\right)$



- Choose $k = 2$
- BuildTree$(\{1, 2, 3\}; 2)$
- Line 1 true
- Consider a fixed (j, s)
  - $I^+_{j,s} = \{2, 3\}$
  - $I^-_{j,s} = \{1\}$
  - $\hat{y}^+_{j,s} = 5$
  - $\hat{y}^-_{j,s} = 0$
  - $E_{j,s} = 0$

BuildTree$(I; k)$

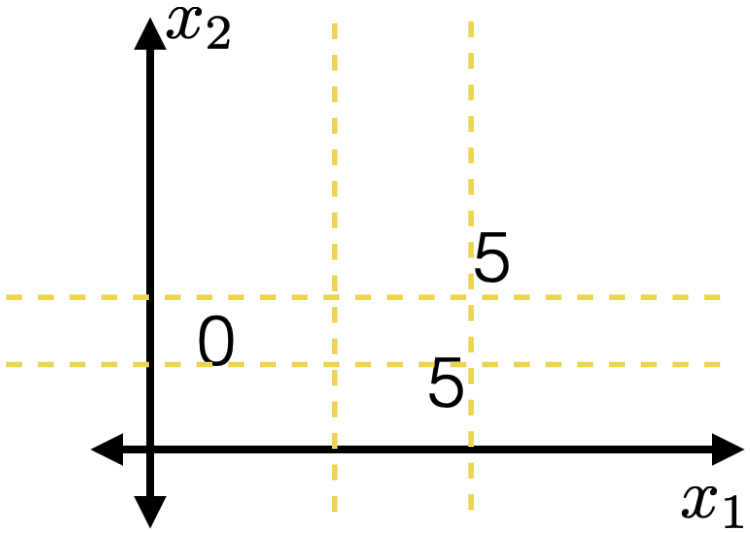
1. **if** $|I| > k$
2. **for** each split dim j and split value s
3. Set $I^+_{j,s} = \left\{ i \in I \mid x^{(i)}_j \geq s \right\}$
4. Set $I^-_{j,s} = \left\{ i \in I \mid x^{(i)}_j < s \right\}$
5. Set $\hat{y}^+_{j,s} = \text{average}_{\ i \in I^+_{j,s}}\ y^{(i)}$
6. Set $\hat{y}^-_{j,s} = \text{average}_{\ i \in I^-_{j,s}}\ y^{(i)}$
7. Set $E_{j,s} = \sum_{i \in I^+_{j,s}} \left(y^{(i)} - \hat{y}^+_{j,s} \right)^2 + \sum_{i \in I^-_{j,s}} \left(y^{(i)} - \hat{y}^-_{j,s} \right)^2$
8. Set $(j^*, s^*) = \arg\min_{j,s} E_{j,s}$
9. **else**
10. Set $\hat{y} = \text{average}_{\ i \in I}\ y^{(i)}$
11. **return** LEAF(leave_value=$\hat{y}$)
12. **return** Node $\left(j^*, s^*, \text{BuildTree}\left(I^-_{j^*,s^*}, k\right), \text{BuildTree}\left(I^+_{j^*,s^*}, k\right)\right)$



- Choose $k = 2$
- BuildTree$(\{1, 2, 3\}; 2)$
- Line 1 true
- Consider a fixed (j, s)
  - $I^+_{j,s} = \{2, 3\}$
  - $I^-_{j,s} = \{1\}$
  - $\hat{y}^+_{j,s} = 5$
  - $\hat{y}^-_{j,s} = 0$
  - $E_{j,s} = 0$

BuildTree$(I; k)$

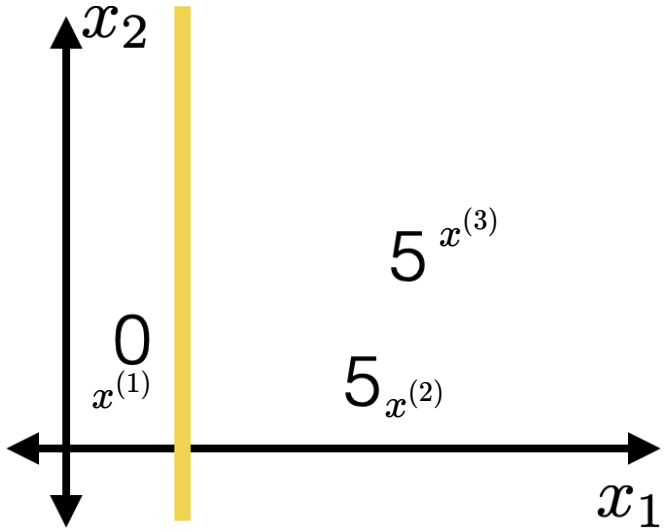
1. **if** $|I| > k$
2. **for** each split dim j and split value s
3. Set $I^+_{j,s} = \left\{ i \in I \mid x^{(i)}_j \geq s \right\}$
4. Set $I^-_{j,s} = \left\{ i \in I \mid x^{(i)}_j < s \right\}$
5. Set $\hat{y}^+_{j,s} =$average $_{i \in I^+_{j,s}}\ y^{(i)}$
6. Set $\hat{y}^-_{j,s} =$average $_{i \in I^-_{j,s}}\ y^{(i)}$
7. Set $E_{j,s} = \sum_{i \in I^+_{j,s}} \left(y^{(i)} - \hat{y}^+_{j,s} \right)^2 + \sum_{i \in I^-_{j,s}} \left(y^{(i)} - \hat{y}^-_{j,s} \right)^2$
8. Set $(j^*, s^*) = \arg\min_{j,s} E_{j,s}$
9. **else**
10. Set $\hat{y} =$ average $_{i \in I}\ y^{(i)}$
11. **return** LEAF(leave_value=$\hat{y}$)
12. **return** Node $\left(j^*, s^*, \text{BuildTree}\left(I^-_{j^*,s^*}, k\right), \text{BuildTree}\left(I^+_{j^*,s^*}, k\right)\right)$



- So for line 2: a finite number of (j, s) combo suffices (those splits in-between data points)
- Line 8 picks the "best" among these finite combos. (random tie-breaking)

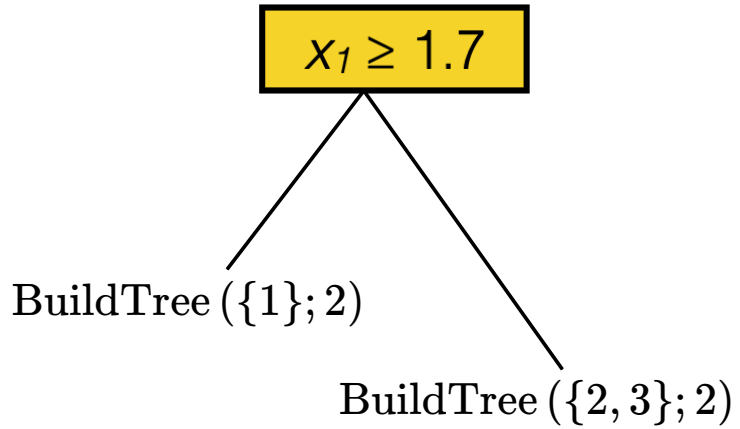
BuildTree$(I; k)$

1. **if** $|I| > k$
2. **for** each split dim j and split value s
3. Set $I^+_{j,s} = \left\{ i \in I \mid x^{(i)}_j \geq s \right\}$
4. Set $I^-_{j,s} = \left\{ i \in I \mid x^{(i)}_j < s \right\}$
5. Set $\hat{y}^+_{j,s} = \text{average}_{\ i \in I^+_{j,s}}\ y^{(i)}$
6. Set $\hat{y}^-_{j,s} = \text{average}_{\ i \in I^-_{j,s}}\ y^{(i)}$
7. Set $E_{j,s} = \sum_{i \in I^+_{j,s}} \left(y^{(i)} - \hat{y}^+_{j,s} \right)^2 + \sum_{i \in I^-_{j,s}} \left(y^{(i)} - \hat{y}^-_{j,s} \right)^2$
8. Set $(j^*, s^*) = \arg\min_{j,s} E_{j,s}$
9. **else**
10. Set $\hat{y} = \text{average}_{\ i \in I}\ y^{(i)}$
11. **return** LEAF(leave_value=$\hat{y}$)
12. **return** Node $\left(j^*, s^*, \text{BuildTree}\left(I^-_{j^*,s^*}, k\right), \text{BuildTree}\left(I^+_{j^*,s^*}, k\right)\right)$



Suppose line 8 sets this

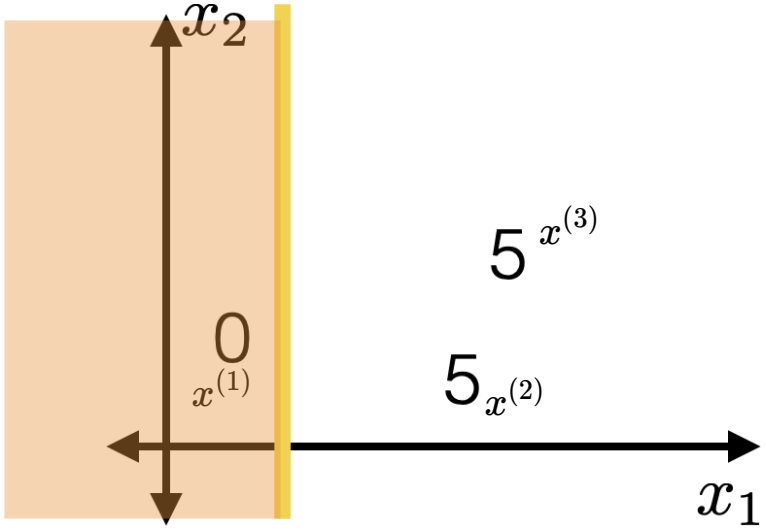
$(j^*, s^*) = (1, 1.7)$



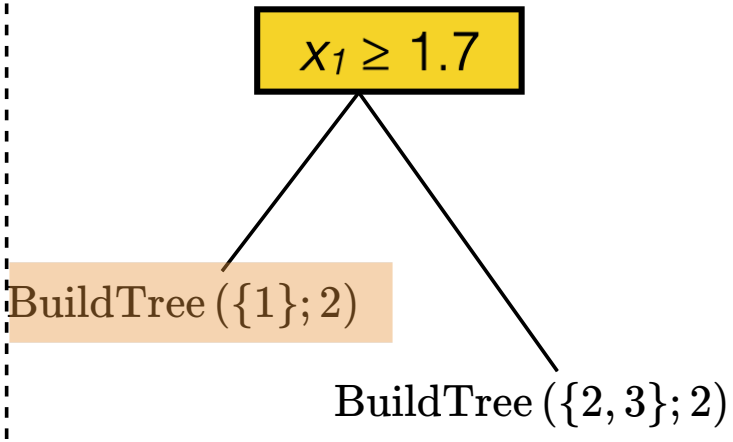
then 12 recursion

BuildTree$(I; k)$

1. **if** $|I| > k$
2. **for** each split dim j and split value s
3. Set $I^+_{j,s} = \left\{ i \in I \mid x^{(i)}_j \geq s \right\}$
4. Set $I^-_{j,s} = \left\{ i \in I \mid x^{(i)}_j < s \right\}$
5. Set $\hat{y}^+_{j,s} = \text{average}_{\ i \in I^+_{j,s}}\ y^{(i)}$
6. Set $\hat{y}^-_{j,s} = \text{average}_{\ i \in I^-_{j,s}}\ y^{(i)}$
7. Set $E_{j,s} = \sum_{i \in I^+_{j,s}} \left(y^{(i)} - \hat{y}^+_{j,s} \right)^2 + \sum_{i \in I^-_{j,s}} \left(y^{(i)} - \hat{y}^-_{j,s} \right)^2$
8. Set $(j^*, s^*) = \arg\min_{j,s} E_{j,s}$
9. **else**
10. Set $\hat{y} = \text{average}_{\ i \in I}\ y^{(i)}$
11. **return** LEAF(leave_value=$\hat{y}$)
12. **return** Node $\left(j^*, s^*, \text{BuildTree}\left(I^-_{j^*,s^*}, k\right), \text{BuildTree}\left(I^+_{j^*,s^*}, k\right)\right)$



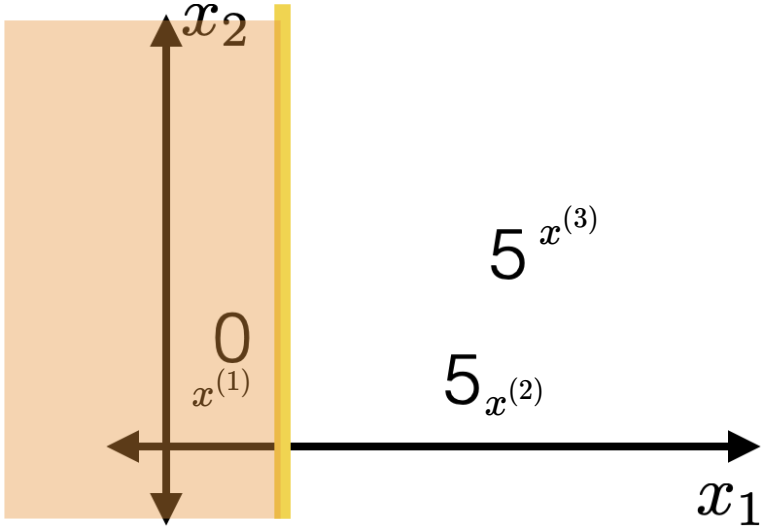
Line 8 sets this (j^*, s^*)



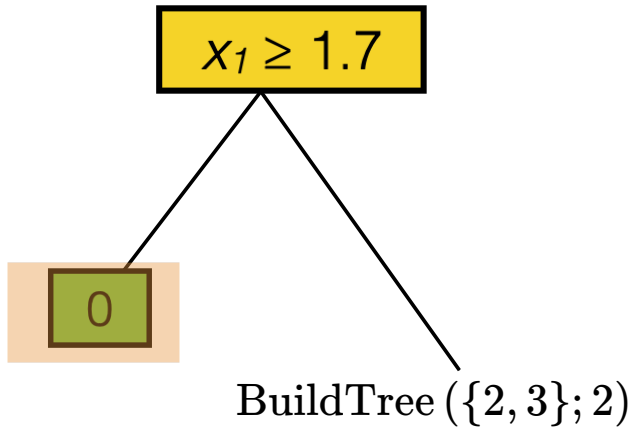
Line 12 recursion

BuildTree$(I; k)$

1. **if** $|I| > k$
2. **for** each split dim j and split value s
3. Set $I^+_{j,s} = \left\{ i \in I \mid x^{(i)}_j \geq s \right\}$
4. Set $I^-_{j,s} = \left\{ i \in I \mid x^{(i)}_j < s \right\}$
5. Set $\hat{y}^+_{j,s} = \text{average}_{\ i \in I^+_{j,s}}\ y^{(i)}$
6. Set $\hat{y}^-_{j,s} = \text{average}_{\ i \in I^-_{j,s}}\ y^{(i)}$
7. Set $E_{j,s} = \sum_{i \in I^+_{j,s}} \left(y^{(i)} - \hat{y}^+_{j,s} \right)^2 + \sum_{i \in I^-_{j,s}} \left(y^{(i)} - \hat{y}^-_{j,s} \right)^2$
8. Set $(j^*, s^*) = \arg\min_{j,s} E_{j,s}$
9. **else**
10. Set $\hat{y} = \text{average}_{\ i \in I}\ y^{(i)}$
11. **return** LEAF(leave_value=$\hat{y}$)
12. **return** Node $\left(j^*, s^*, \text{BuildTree}\left(I^-_{j^*,s^*}, k\right), \text{BuildTree}\left(I^+_{j^*,s^*}, k\right)\right)$



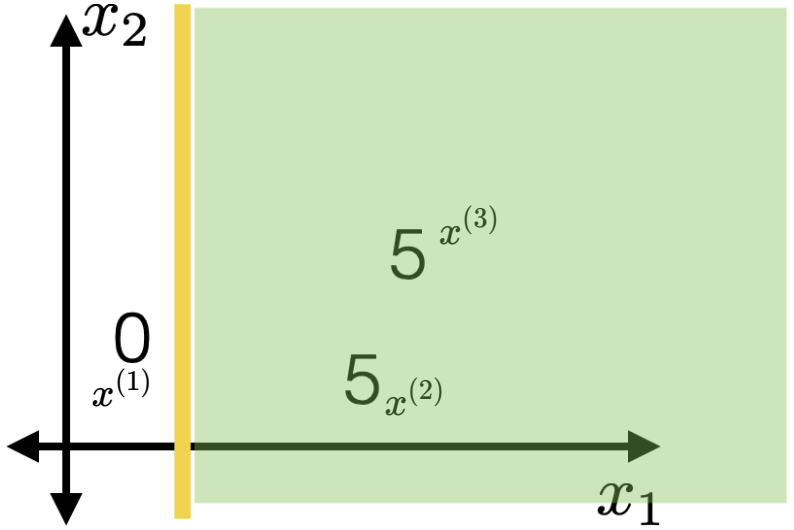
Line 8 sets this (j^*, s^*)



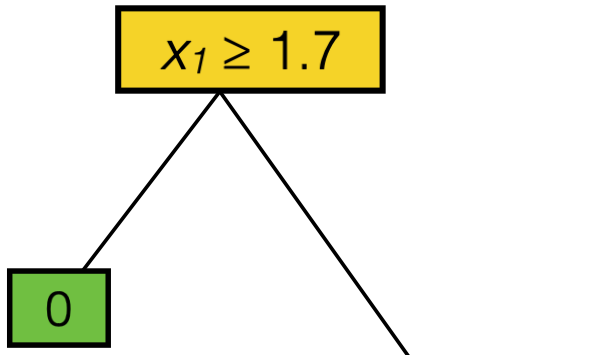
Line 12 recursion

BuildTree$(I; k)$

1. **if** $|I| > k$
2. **for** each split dim j and split value s
3. Set $I^+_{j,s} = \left\{ i \in I \mid x^{(i)}_j \geq s \right\}$
4. Set $I^-_{j,s} = \left\{ i \in I \mid x^{(i)}_j < s \right\}$
5. Set $\hat{y}^+_{j,s} = \text{average}_{\ i \in I^+_{j,s}}\ y^{(i)}$
6. Set $\hat{y}^-_{j,s} = \text{average}_{\ i \in I^-_{j,s}}\ y^{(i)}$
7. Set $E_{j,s} = \sum_{i \in I^+_{j,s}} \left(y^{(i)} - \hat{y}^+_{j,s} \right)^2 + \sum_{i \in I^-_{j,s}} \left(y^{(i)} - \hat{y}^-_{j,s} \right)^2$
8. Set $(j^*, s^*) = \arg\min_{j,s} E_{j,s}$
9. **else**
10. Set $\hat{y} = \text{average}_{\ i \in I}\ y^{(i)}$
11. **return** LEAF(leave_value=$\hat{y}$)
12. **return** Node $\left(j^*, s^*, \text{BuildTree}\left(I^-_{j^*,s^*}, k \right), \text{BuildTree}\left(I^+_{j^*,s^*}, k \right) \right)$



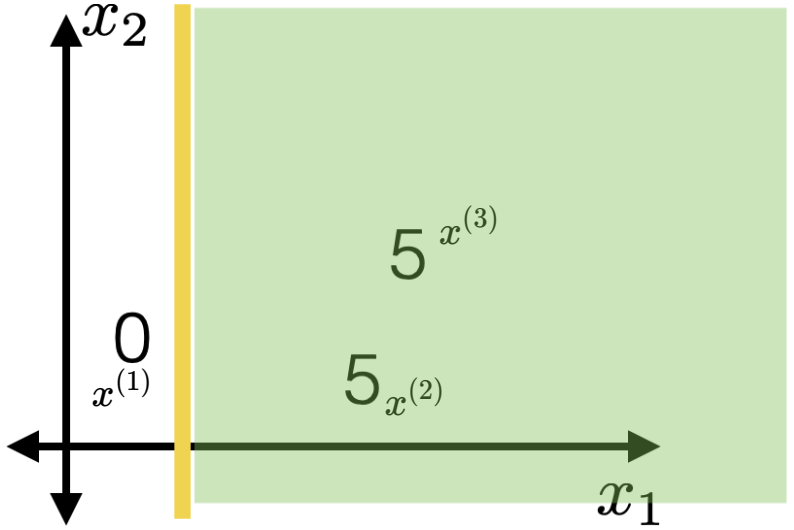
Line 8 sets this (j^*, s^*)



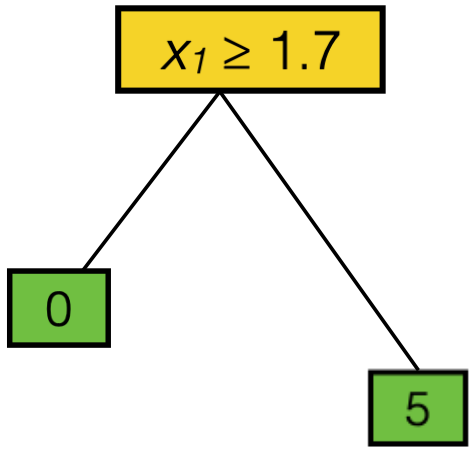
Line 12 recursion

BuildTree$(I; k)$

1. **if** $|I| > k$
2. **for** each split dim j and split value s
3. Set $I^+_{j,s} = \left\{ i \in I \mid x^{(i)}_j \geq s \right\}$
4. Set $I^-_{j,s} = \left\{ i \in I \mid x^{(i)}_j < s \right\}$
5. Set $\hat{y}^+_{j,s} = \text{average}_{\ i \in I^+_{j,s}}\ y^{(i)}$
6. Set $\hat{y}^-_{j,s} = \text{average}_{\ i \in I^-_{j,s}}\ y^{(i)}$
7. Set $E_{j,s} = \sum_{i \in I^+_{j,s}} \left(y^{(i)} - \hat{y}^+_{j,s} \right)^2 + \sum_{i \in I^-_{j,s}} \left(y^{(i)} - \hat{y}^-_{j,s} \right)^2$
8. Set $(j^*, s^*) = \arg\min_{j,s} E_{j,s}$
9. **else**
10. Set $\hat{y} = \text{average}_{\ i \in I}\ y^{(i)}$
11. **return** LEAF(leave_value=$\hat{y}$)
12. **return** Node $\left(j^*, s^*, \text{BuildTree}\left(I^-_{j^*,s^*}, k\right), \text{BuildTree}\left(I^+_{j^*,s^*}, k\right)\right)$



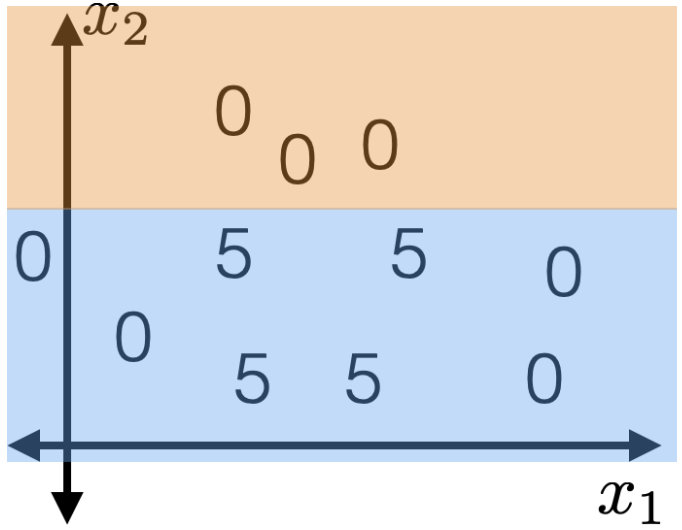
Line 8 sets this (j^*, s^*)



Line 12 recursion

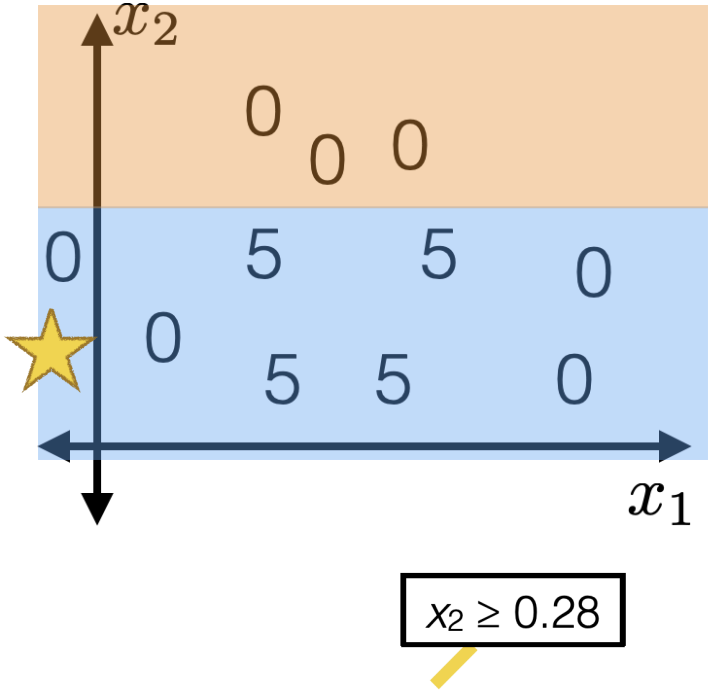
BuildTree$(I; k)$

1. **if** $|I| > k$
2. **for** each split dim j and split value s
3. Set $I^+_{j,s} = \left\{ i \in I \mid x^{(i)}_j \geq s \right\}$
4. Set $I^-_{j,s} = \left\{ i \in I \mid x^{(i)}_j < s \right\}$
5. Set $\hat{y}^+_{j,s} = \text{average}_{\ i \in I^+_{j,s}}\ y^{(i)}$
6. Set $\hat{y}^-_{j,s} = \text{average}_{\ i \in I^-_{j,s}}\ y^{(i)}$
7. Set $E_{j,s} = \sum_{i \in I^+_{j,s}} \left(y^{(i)} - \hat{y}^+_{j,s} \right)^2 + \sum_{i \in I^-_{j,s}} \left(y^{(i)} - \hat{y}^-_{j,s} \right)^2$
8. Set $(j^*, s^*) = \arg\min_{j,s} E_{j,s}$
9. **else**
10. Set $\hat{y} = \text{average}_{\ i \in I}\ y^{(i)}$
11. **return** LEAF(leave_value=$\hat{y}$)
12. **return** Node $\left(j^*, s^*, \text{BuildTree}\left(I^-_{j^*,s^*}, k\right), \text{BuildTree}\left(I^+_{j^*,s^*}, k\right)\right)$



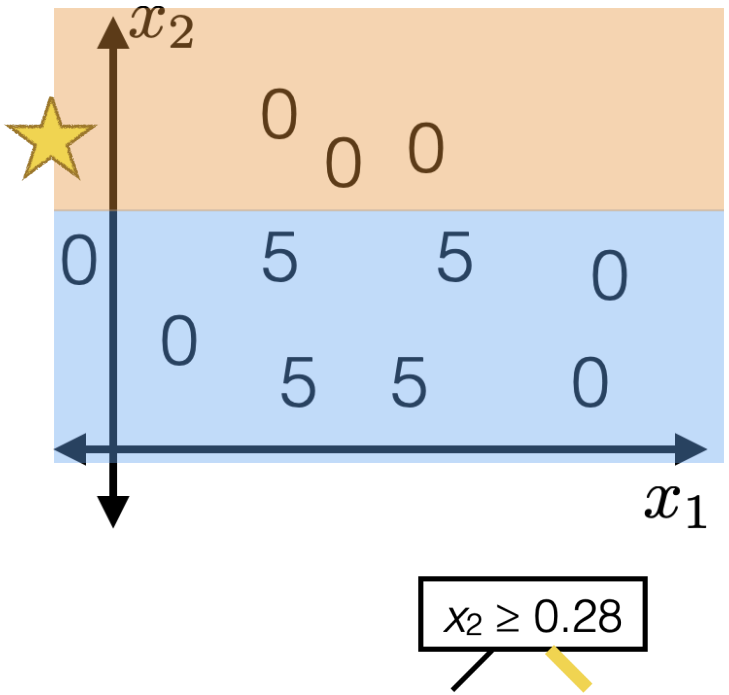
BuildTree$(I; k)$

1. **if** $|I| > k$
2. **for** each split dim j and split value s
3. Set $I^+_{j,s} = \left\{ i \in I \mid x^{(i)}_j \geq s \right\}$
4. Set $I^-_{j,s} = \left\{ i \in I \mid x^{(i)}_j < s \right\}$
5. Set $\hat{y}^+_{j,s} = \text{average}_{\ i \in I^+_{j,s}}\ y^{(i)}$
6. Set $\hat{y}^-_{j,s} = \text{average}_{\ i \in I^-_{j,s}}\ y^{(i)}$
7. Set $E_{j,s} = \sum_{i \in I^+_{j,s}} \left(y^{(i)} - \hat{y}^+_{j,s} \right)^2 + \sum_{i \in I^-_{j,s}} \left(y^{(i)} - \hat{y}^-_{j,s} \right)^2$
8. Set $(j^*, s^*) = \arg\min_{j,s} E_{j,s}$
9. **else**
10. Set $\hat{y} = \text{average}_{\ i \in I}\ y^{(i)}$
11. **return** LEAF(leave_value=$\hat{y}$)
12. **return** Node $\left(j^*, s^*, \text{BuildTree}\left(I^-_{j^*,s^*}, k\right), \text{BuildTree}\left(I^+_{j^*,s^*}, k\right)\right)$



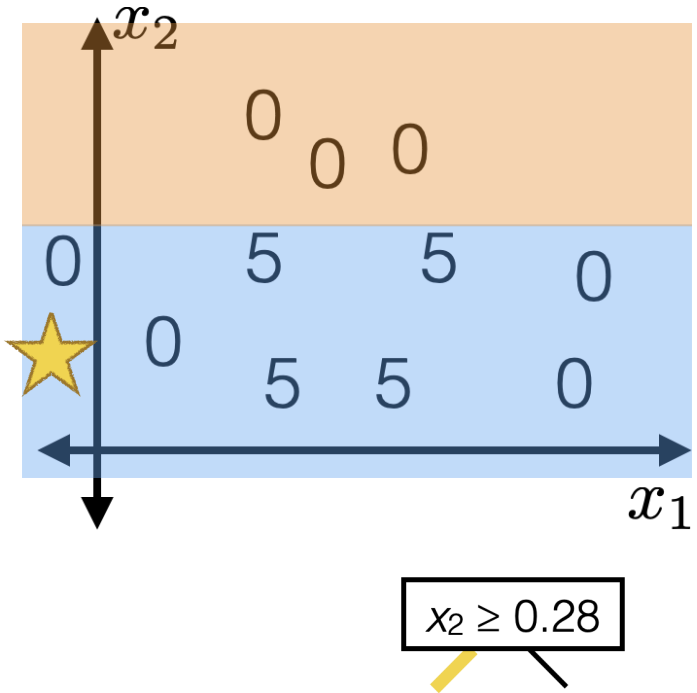
BuildTree$(I; k)$

1. **if** $|I| > k$
2. **for** each split dim j and split value s
3. Set $I^+_{j,s} = \left\{ i \in I \mid x^{(i)}_j \geq s \right\}$
4. Set $I^-_{j,s} = \left\{ i \in I \mid x^{(i)}_j < s \right\}$
5. Set. $\hat{y}^+_{j,s}$ =average $_{i \in I^+_{j,s}}\ y^{(i)}$
6. Set $\hat{y}^-_{j,s}$ =average $_{i \in I^-_{j,s}}\ y^{(i)}$
7. Set $E_{j,s} = \sum_{i \in I^+_{j,s}} \left(y^{(i)} - \hat{y}^+_{j,s} \right)^2 + \sum_{i \in I^-_{j,s}} \left(y^{(i)} - \hat{y}^-_{j,s} \right)^2$
8. Set $(j^*, s^*) = \arg\min_{j,s} E_{j,s}$
9. **else**
10. Set $\hat{y} =$ average $_{i \in I}\ y^{(i)}$
11. **return** LEAF(leave_value=$\hat{y}$)
12. **return** Node $\left(j^*, s^*, \text{BuildTree}\left(I^-_{j^*,s^*}, k \right), \text{BuildTree}\left(I^+_{j^*,s^*}, k \right) \right)$



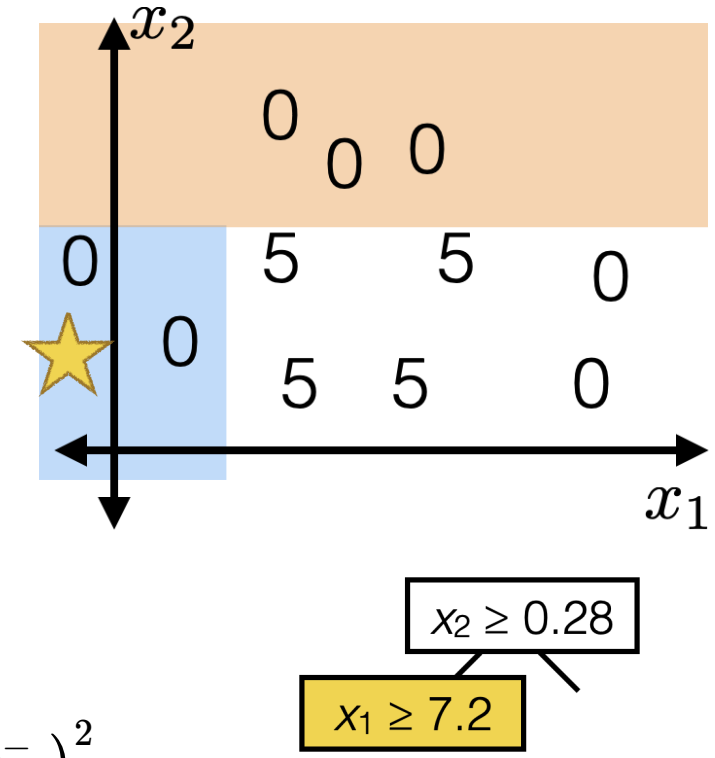
BuildTree$(I; k)$

1. **if** $|I| > k$
2. **for** each split dim j and split value s
3. Set $I^+_{j,s} = \left\{ i \in I \mid x^{(i)}_j \geq s \right\}$
4. Set $I^-_{j,s} = \left\{ i \in I \mid x^{(i)}_j < s \right\}$
5. Set. $\hat{y}^+_{j,s} =$average $_{i \in I^+_{j,s}}\ y^{(i)}$
6. Set $\hat{y}^-_{j,s} =$average $_{i \in I^-_{j,s}}\ y^{(i)}$
7. Set $E_{j,s} = \sum_{i \in I^+_{j,s}} \left(y^{(i)} - \hat{y}^+_{j,s} \right)^2 + \sum_{i \in I^-_{j,s}} \left(y^{(i)} - \hat{y}^-_{j,s} \right)^2$
8. Set $(j^*, s^*) = \arg\min_{j,s} E_{j,s}$
9. **else**
10. Set $\hat{y} =$ average $_{i \in I}\ y^{(i)}$
11. **return** LEAF(leave_value=$\hat{y}$)
12. **return** Node $\left(j^*, s^*, \text{BuildTree}\left(I^-_{j^*,s^*}, k\right), \text{BuildTree}\left(I^+_{j^*,s^*}, k\right)\right)$



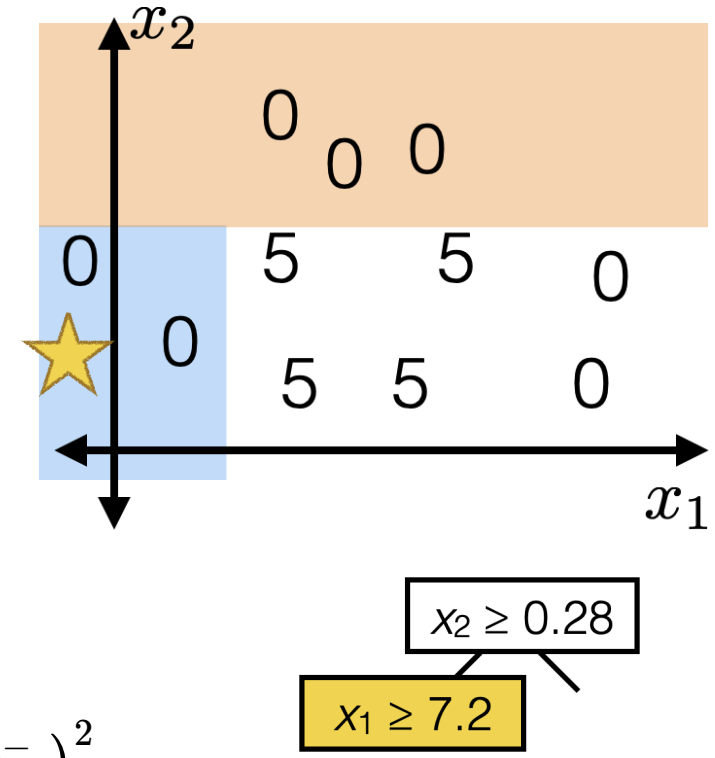
BuildTree$(I; k)$

1. **if** $|I| > k$
2. **for** each split dim j and split value s
3. Set $I^+_{j,s} = \left\{ i \in I \mid x^{(i)}_j \geq s \right\}$
4. Set $I^-_{j,s} = \left\{ i \in I \mid x^{(i)}_j < s \right\}$
5. Set. $\hat{y}^+_{j,s}$ =average $_{i \in I^+_{j,s}}\ y^{(i)}$
6. Set $\hat{y}^-_{j,s}$ =average $_{i \in I^-_{j,s}}\ y^{(i)}$
7. Set $E_{j,s} = \sum_{i \in I^+_{j,s}} \left(y^{(i)} - \hat{y}^+_{j,s} \right)^2 + \sum_{i \in I^-_{j,s}} \left(y^{(i)} - \hat{y}^-_{j,s} \right)^2$
8. Set $(j^*, s^*) = \arg\min_{j,s} E_{j,s}$
9. **else**
10. Set $\hat{y} =$ average $_{i \in I}\ y^{(i)}$
11. **return** LEAF(leave_value=$\hat{y}$)
12. **return** Node $\left(j^*, s^*, \text{BuildTree}\left(I^-_{j^*,s^*}, k\right), \text{BuildTree}\left(I^+_{j^*,s^*}, k\right)\right)$



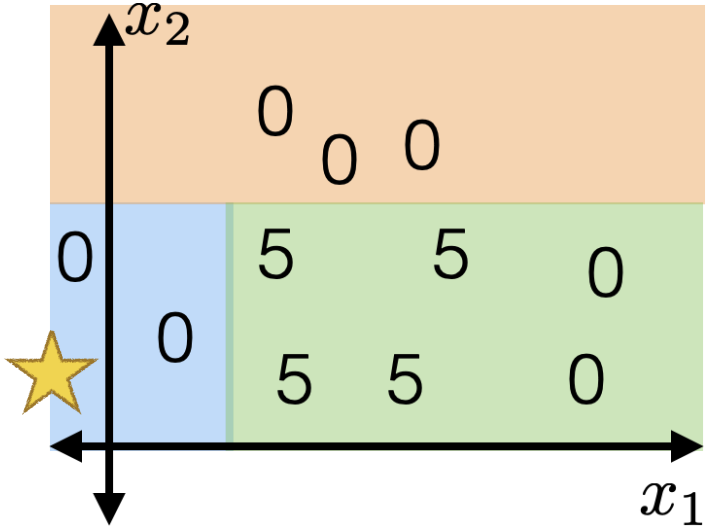
BuildTree$(I; k)$

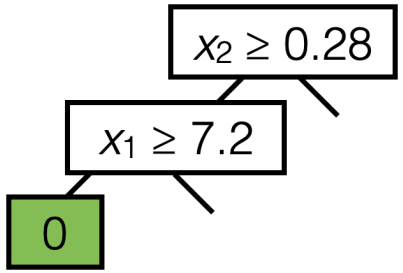
1. **if** $|I| > k$
2. **for** each split dim j and split value s
3. Set $I^+_{j,s} = \left\{ i \in I \mid x^{(i)}_j \geq s \right\}$
4. Set $I^-_{j,s} = \left\{ i \in I \mid x^{(i)}_j < s \right\}$
5. Set. $\hat{y}^+_{j,s}$ =average $_{i \in I^+_{j,s}}\ y^{(i)}$
6. Set $\hat{y}^-_{j,s}$ =average $_{i \in I^-_{j,s}}\ y^{(i)}$
7. Set $E_{j,s} = \sum_{i \in I^+_{j,s}} \left(y^{(i)} - \hat{y}^+_{j,s} \right)^2 + \sum_{i \in I^-_{j,s}} \left(y^{(i)} - \hat{y}^-_{j,s} \right)^2$
8. Set $(j^*, s^*) = \arg\min_{j,s} E_{j,s}$
9. **else**
10. Set $\hat{y} =$ average $_{i \in I}\ y^{(i)}$
11. **return** LEAF(leave_value=$\hat{y}$)
12. **return** Node $\left(j^*, s^*, \text{BuildTree}\left(I^-_{j^*,s^*}, k \right), \text{BuildTree}\left(I^+_{j^*,s^*}, k \right) \right)$



BuildTree$(I; k)$

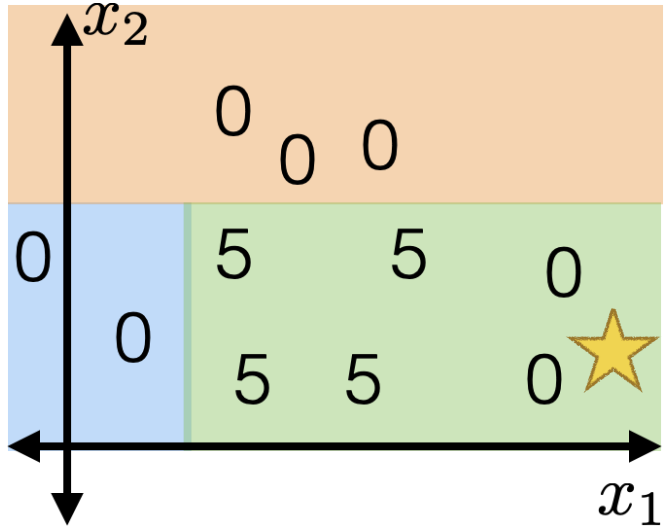
1. **if** $|I| > k$
2. **for** each split dim j and split value s
3. Set $I^+_{j,s} = \left\{ i \in I \mid x^{(i)}_j \geq s \right\}$
4. Set $I^-_{j,s} = \left\{ i \in I \mid x^{(i)}_j < s \right\}$
5. Set. $\hat{y}^+_{j,s}$ =average $_{i \in I^+_{j,s}}\ y^{(i)}$
6. Set $\hat{y}^-_{j,s}$ =average $_{i \in I^-_{j,s}}\ y^{(i)}$
7. Set $E_{j,s} = \sum_{i \in I^+_{j,s}} \left(y^{(i)} - \hat{y}^+_{j,s} \right)^2 + \sum_{i \in I^-_{j,s}} \left(y^{(i)} - \hat{y}^-_{j,s} \right)^2$
8. Set $(j^*, s^*) = \arg\min_{j,s} E_{j,s}$
9. **else**
10. Set $\hat{y}$ = average $_{i \in I}\ y^{(i)}$
11. **return** LEAF(leave_value=$\hat{y}$)
12. **return** Node $\left(j^*, s^*, \text{BuildTree}\left(I^-_{j^*,s^*}, k\right), \text{BuildTree}\left(I^+_{j^*,s^*}, k\right)\right)$

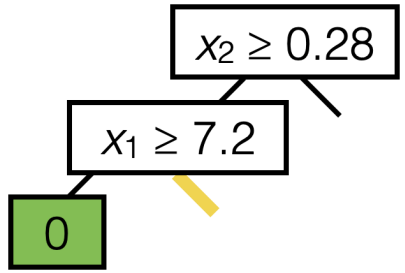




BuildTree$(I; k)$

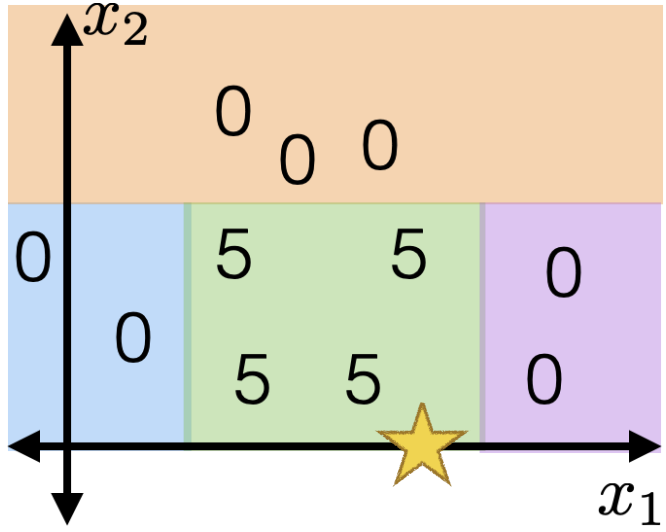
1. **if** $|I| > k$
2. **for** each split dim j and split value s
3. Set $I^+_{j,s} = \left\{ i \in I \mid x^{(i)}_j \geq s \right\}$
4. Set $I^-_{j,s} = \left\{ i \in I \mid x^{(i)}_j < s \right\}$
5. Set. $\hat{y}^+_{j,s}$ =average $_{i \in I^+_{j,s}}\ y^{(i)}$
6. Set $\hat{y}^-_{j,s}$ =average $_{i \in I^-_{j,s}}\ y^{(i)}$
7. Set $E_{j,s} = \sum_{i \in I^+_{j,s}} \left(y^{(i)} - \hat{y}^+_{j,s} \right)^2 + \sum_{i \in I^-_{j,s}} \left(y^{(i)} - \hat{y}^-_{j,s} \right)^2$
8. Set $(j^*, s^*) = \arg\min_{j,s} E_{j,s}$
9. **else**
10. Set $\hat{y} =$ average $_{i \in I}\ y^{(i)}$
11. **return** LEAF(leave_value=$\hat{y}$)
12. **return** Node $\left(j^*, s^*, \text{BuildTree}\left(I^-_{j^*,s^*}, k\right), \text{BuildTree}\left(I^+_{j^*,s^*}, k\right)\right)$

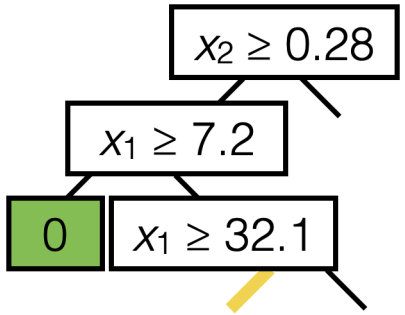




BuildTree$(I; k)$

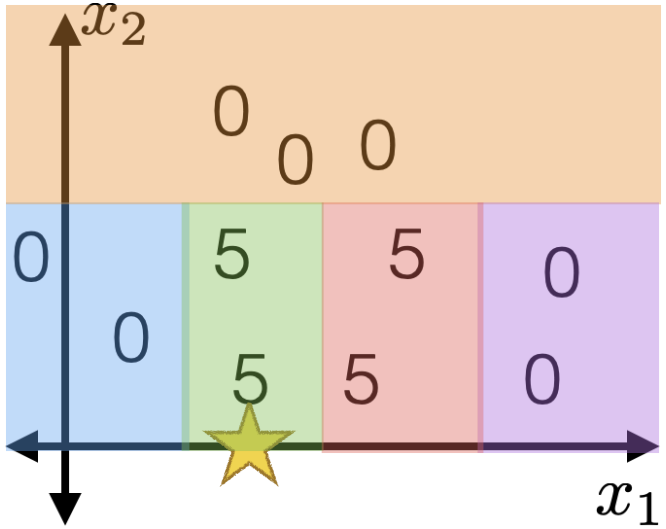
1. **if** $|I| > k$
2. **for** each split dim j and split value s
3. Set $I^+_{j,s} = \left\{ i \in I \mid x^{(i)}_j \geq s \right\}$
4. Set $I^-_{j,s} = \left\{ i \in I \mid x^{(i)}_j < s \right\}$
5. Set. $\hat{y}^+_{j,s}$ =average $_{i \in I^+_{j,s}}\ y^{(i)}$
6. Set $\hat{y}^-_{j,s}$ =average $_{i \in I^-_{j,s}}\ y^{(i)}$
7. Set $E_{j,s} = \sum_{i \in I^+_{j,s}} \left(y^{(i)} - \hat{y}^+_{j,s} \right)^2 + \sum_{i \in I^-_{j,s}} \left(y^{(i)} - \hat{y}^-_{j,s} \right)^2$
8. Set $(j^*, s^*) = \arg\min_{j,s} E_{j,s}$
9. **else**
10. Set $\hat{y} =$ average $_{i \in I}\ y^{(i)}$
11. **return** LEAF(leave_value=$\hat{y}$)
12. **return** Node $\left(j^*, s^*, \text{BuildTree}\left(I^-_{j^*,s^*}, k\right), \text{BuildTree}\left(I^+_{j^*,s^*}, k\right)\right)$

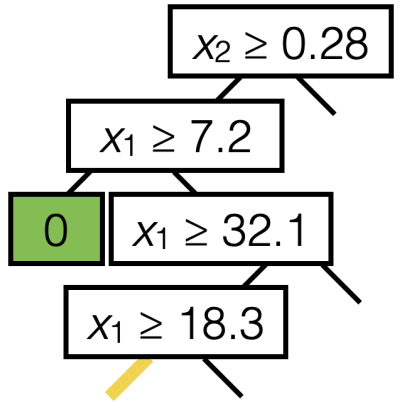




BuildTree$(I; k)$

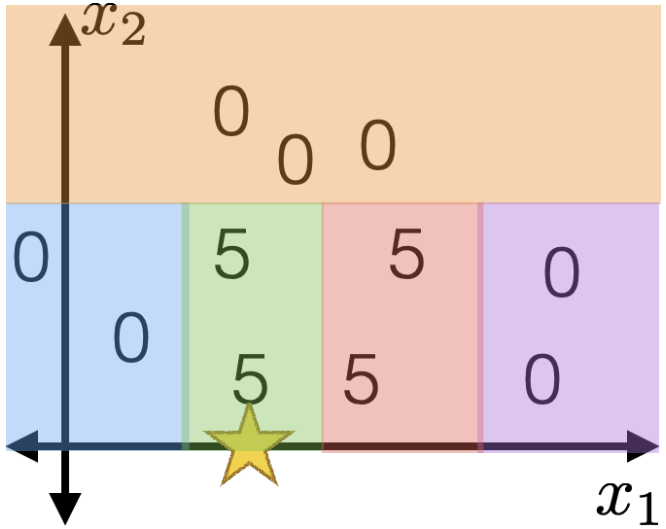
1. **if** $|I| > k$
2. **for** each split dim j and split value s
3. Set $I^+_{j,s} = \left\{ i \in I \mid x^{(i)}_j \geq s \right\}$
4. Set $I^-_{j,s} = \left\{ i \in I \mid x^{(i)}_j < s \right\}$
5. Set. $\hat{y}^+_{j,s}$ =average $_{i \in I^+_{j,s}}\ y^{(i)}$
6. Set $\hat{y}^-_{j,s}$ =average $_{i \in I^-_{j,s}}\ y^{(i)}$
7. Set $E_{j,s} = \sum_{i \in I^+_{j,s}} \left(y^{(i)} - \hat{y}^+_{j,s} \right)^2 + \sum_{i \in I^-_{j,s}} \left(y^{(i)} - \hat{y}^-_{j,s} \right)^2$
8. Set $(j^*, s^*) = \arg\min_{j,s} E_{j,s}$
9. **else**
10. Set $\hat{y} =$ average $_{i \in I}\ y^{(i)}$
11. **return** LEAF(leave_value=$\hat{y}$)
12. **return** Node $\left(j^*, s^*, \text{BuildTree}\left(I^-_{j^*,s^*}, k\right), \text{BuildTree}\left(I^+_{j^*,s^*}, k\right)\right)$

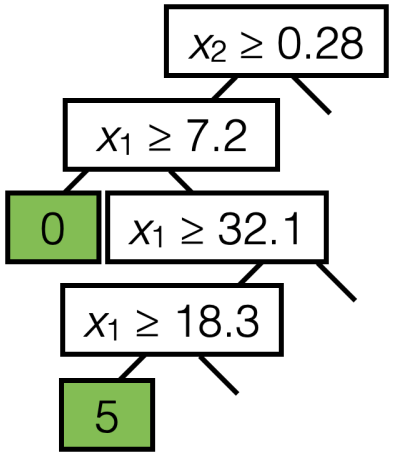




BuildTree$(I; k)$

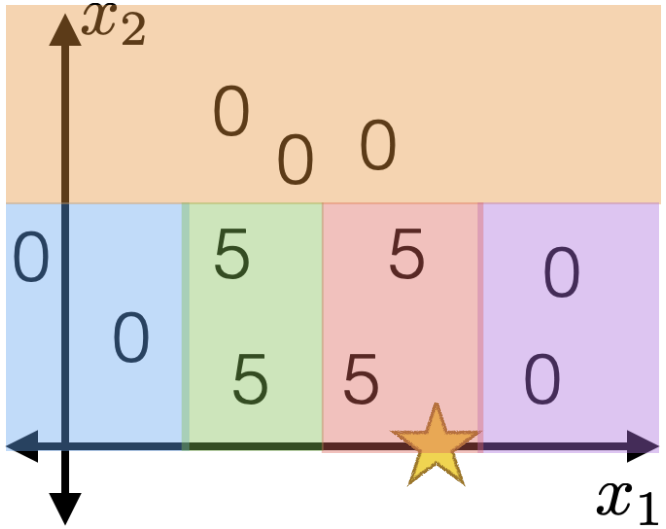
1. **if** $|I| > k$
2. **for** each split dim j and split value s
3. Set $I^+_{j,s} = \left\{ i \in I \mid x^{(i)}_j \geq s \right\}$
4. Set $I^-_{j,s} = \left\{ i \in I \mid x^{(i)}_j < s \right\}$
5. Set. $\hat{y}^+_{j,s}$ =average $_{i \in I^+_{j,s}}\ y^{(i)}$
6. Set $\hat{y}^-_{j,s}$ =average $_{i \in I^-_{j,s}}\ y^{(i)}$
7. Set $E_{j,s} = \sum_{i \in I^+_{j,s}} \left(y^{(i)} - \hat{y}^+_{j,s} \right)^2 + \sum_{i \in I^-_{j,s}} \left(y^{(i)} - \hat{y}^-_{j,s} \right)^2$
8. Set $(j^*, s^*) = \arg\min_{j,s} E_{j,s}$
9. **else**
10. Set $\hat{y} =$ average $_{i \in I}\ y^{(i)}$
11. **return** LEAF(leave_value=$\hat{y}$)
12. **return** Node $\left(j^*, s^*, \text{BuildTree}\left(I^-_{j^*,s^*}, k\right), \text{BuildTree}\left(I^+_{j^*,s^*}, k\right)\right)$

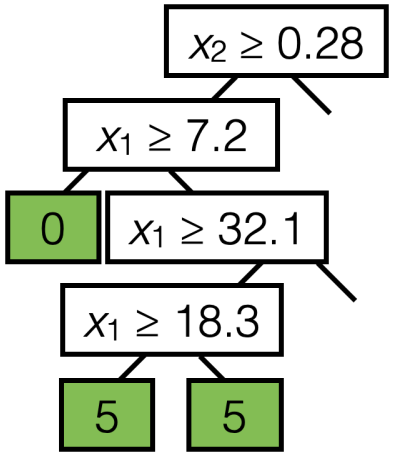




BuildTree$(I; k)$

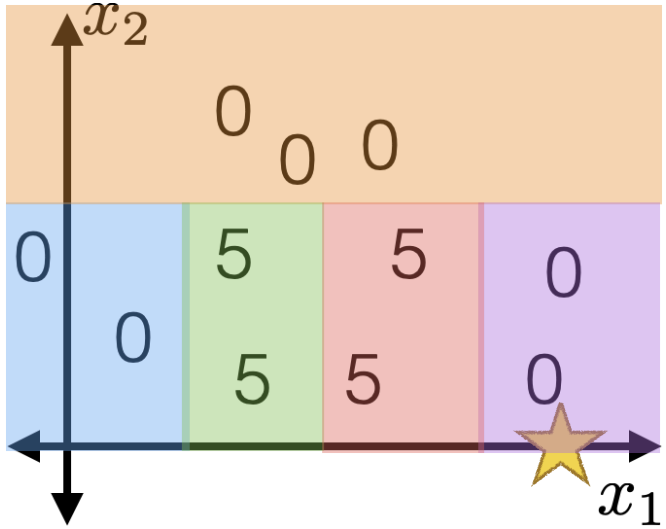
1. **if** $|I| > k$
2. **for** each split dim j and split value s
3. Set $I^+_{j,s} = \left\{ i \in I \mid x^{(i)}_j \geq s \right\}$
4. Set $I^-_{j,s} = \left\{ i \in I \mid x^{(i)}_j < s \right\}$
5. Set. $\hat{y}^+_{j,s}$ =average $_{i \in I^+_{j,s}}\ y^{(i)}$
6. Set $\hat{y}^-_{j,s}$ =average $_{i \in I^-_{j,s}}\ y^{(i)}$
7. Set $E_{j,s} = \sum_{i \in I^+_{j,s}} \left(y^{(i)} - \hat{y}^+_{j,s} \right)^2 + \sum_{i \in I^-_{j,s}} \left(y^{(i)} - \hat{y}^-_{j,s} \right)^2$
8. Set $(j^*, s^*) = \arg\min_{j,s} E_{j,s}$
9. **else**
10. Set $\hat{y} =$ average $_{i \in I}\ y^{(i)}$
11. **return** LEAF(leave_value=$\hat{y}$)
12. **return** Node $\left(j^*, s^*, \text{BuildTree}\left(I^-_{j^*,s^*}, k\right), \text{BuildTree}\left(I^+_{j^*,s^*}, k\right)\right)$

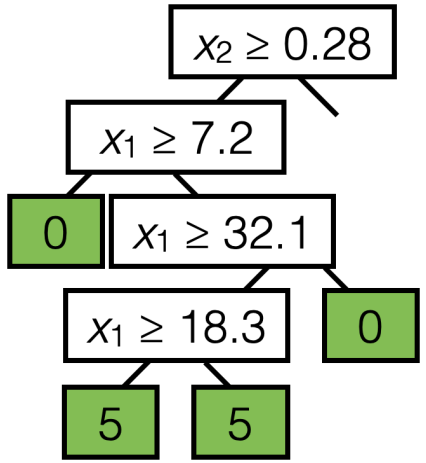




BuildTree$(I; k)$

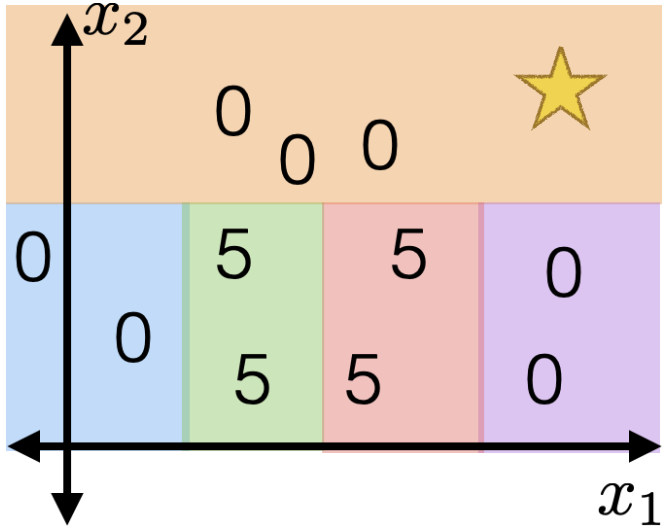
1. **if** $|I| > k$
2. **for** each split dim j and split value s
3. Set $I^+_{j,s} = \left\{ i \in I \mid x^{(i)}_j \geq s \right\}$
4. Set $I^-_{j,s} = \left\{ i \in I \mid x^{(i)}_j < s \right\}$
5. Set. $\hat{y}^+_{j,s}$ =average $_{i \in I^+_{j,s}}\ y^{(i)}$
6. Set $\hat{y}^-_{j,s}$ =average $_{i \in I^-_{j,s}}\ y^{(i)}$
7. Set $E_{j,s} = \sum_{i \in I^+_{j,s}} \left(y^{(i)} - \hat{y}^+_{j,s} \right)^2 + \sum_{i \in I^-_{j,s}} \left(y^{(i)} - \hat{y}^-_{j,s} \right)^2$
8. Set $(j^*, s^*) = \arg\min_{j,s} E_{j,s}$
9. **else**
10. Set $\hat{y} =$ average $_{i \in I}\ y^{(i)}$
11. **return** LEAF(leave_value=$\hat{y}$)
12. **return** Node $\left(j^*, s^*, \text{BuildTree}\left(I^-_{j^*,s^*}, k\right), \text{BuildTree}\left(I^+_{j^*,s^*}, k\right)\right)$

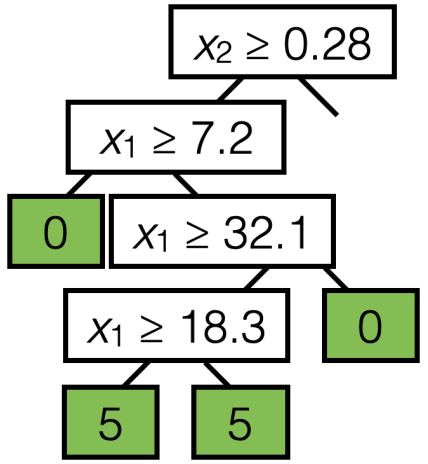




BuildTree$(I; k)$

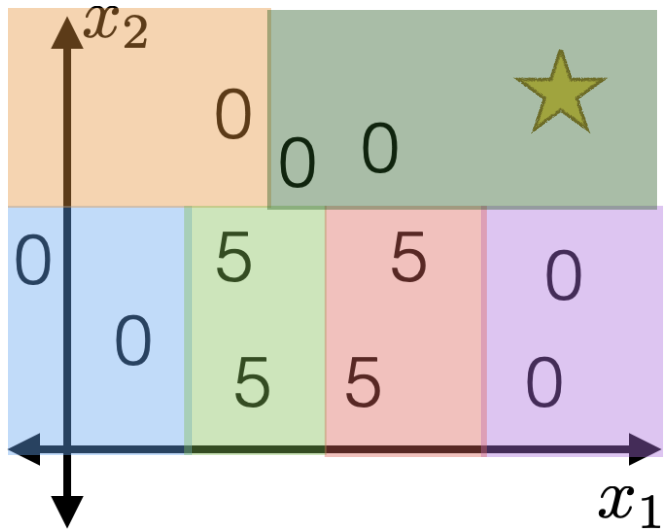
1. **if** $|I| > k$
2. **for** each split dim j and split value s
3. Set $I^+_{j,s} = \left\{ i \in I \mid x^{(i)}_j \geq s \right\}$
4. Set $I^-_{j,s} = \left\{ i \in I \mid x^{(i)}_j < s \right\}$
5. Set $\hat{y}^+_{j,s} = \text{average}_{\ i \in I^+_{j,s}}\ y^{(i)}$
6. Set $\hat{y}^-_{j,s} = \text{average}_{\ i \in I^-_{j,s}}\ y^{(i)}$
7. Set $E_{j,s} = \sum_{i \in I^+_{j,s}} \left(y^{(i)} - \hat{y}^+_{j,s} \right)^2 + \sum_{i \in I^-_{j,s}} \left(y^{(i)} - \hat{y}^-_{j,s} \right)^2$
8. Set $(j^*, s^*) = \arg\min_{j,s} E_{j,s}$
9. **else**
10. Set $\hat{y} = \text{average}_{\ i \in I}\ y^{(i)}$
11. **return** LEAF(leave_value=$\hat{y}$)
12. **return** Node $\left(j^*, s^*, \text{BuildTree}\left(I^-_{j^*,s^*}, k\right), \text{BuildTree}\left(I^+_{j^*,s^*}, k\right)\right)$

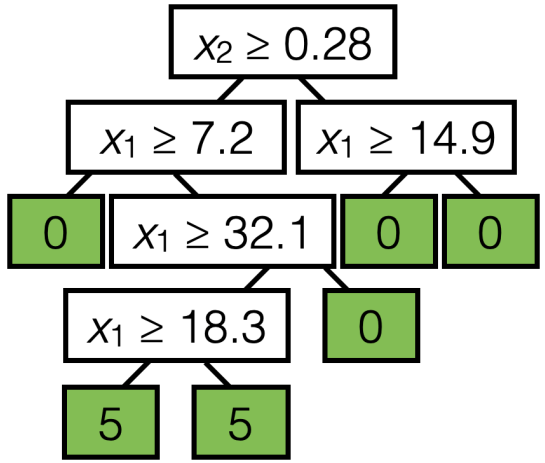




BuildTree$(I; k)$

1. **if** $|I| > k$
2. **for** each split dim j and split value s
3. Set $I^+_{j,s} = \left\{ i \in I \mid x^{(i)}_j \geq s \right\}$
4. Set $I^-_{j,s} = \left\{ i \in I \mid x^{(i)}_j < s \right\}$
5. Set $\hat{y}^+_{j,s} = \text{average}_{\ i \in I^+_{j,s}}\ y^{(i)}$
6. Set $\hat{y}^-_{j,s} = \text{average}_{\ i \in I^-_{j,s}}\ y^{(i)}$
7. Set $E_{j,s} = \sum_{i \in I^+_{j,s}} \left(y^{(i)} - \hat{y}^+_{j,s} \right)^2 + \sum_{i \in I^-_{j,s}} \left(y^{(i)} - \hat{y}^-_{j,s} \right)^2$
8. Set $(j^*, s^*) = \arg\min_{j,s} E_{j,s}$
9. **else**
10. Set $\hat{y} = \text{average}_{\ i \in I}\ y^{(i)}$
11. **return** LEAF(leave_value=$\hat{y}$)
12. **return** Node $\left(j^*, s^*, \text{BuildTree}\left(I^-_{j^*,s^*}, k\right), \text{BuildTree}\left(I^+_{j^*,s^*}, k\right)\right)$





BuildTree$(I; k)$

1. **if** $|I| > k$
2. **for** each split dim j and split value s
3. Set $I^+_{j,s} = \left\{ i \in I \mid x^{(i)}_j \geq s \right\}$
4. Set $I^-_{j,s} = \left\{ i \in I \mid x^{(i)}_j < s \right\}$
5. Set $\hat{y}^+_{j,s} = \text{majority}_{\ i \in I^+_{j,s}}\ y^{(i)}$
6. Set $\hat{y}^-_{j,s} = \text{majority}_{\ i \in I^-_{j,s}}\ y^{(i)}$
7. Set $E_{j,s} = \frac{\left|I^-_{j,s}\right|}{|I|} \cdot H\left(I^-_{j,s}\right) + \frac{\left|I^+_{j,s}\right|}{|I|} \cdot H\left(I^+_{j,s}\right)$
8. Set $(j^*, s^*) = \arg\min_{j,s} E_{j,s}$
9. **else**
10. Set $\hat{y} = \text{majority}_{\ i \in I}\ y^{(i)}$
11. **return** LEAF(leave_value=$\hat{y}$)
12. **return** Node $\left(j^*, s^*, \text{BuildTree}\left(I^-_{j^*,s^*}; k\right), \text{BuildTree}\left(I^+_{j^*,s^*}; k\right)\right)$

The only change from regression to classification:

- Line 5, 6, 10, average becomes majority vote
- Line 7 error more involved

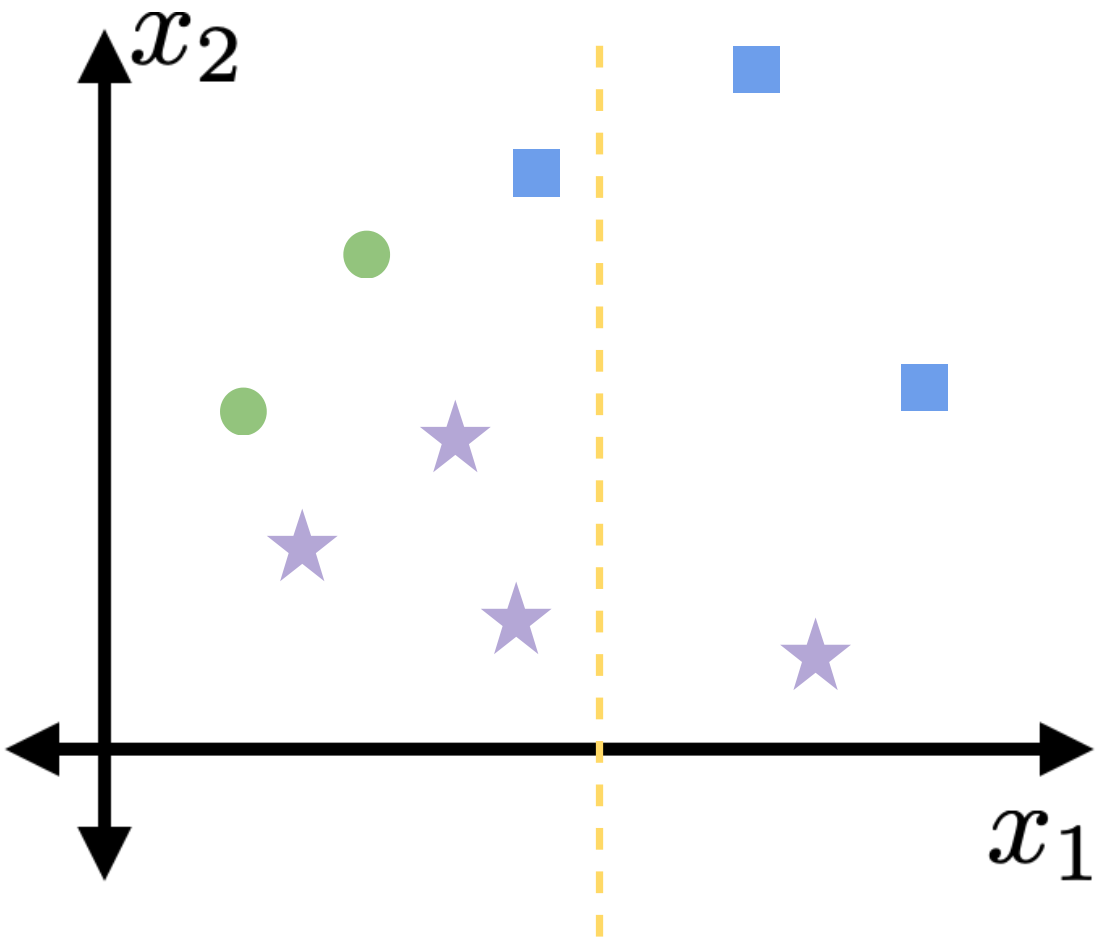
$$E_{j,s} = \frac{\left|I_{j,s}^{-}\right|}{|I|} \cdot H\left(I_{j,s}^{-}\right) + \frac{\left|I_{j,s}^{+}\right|}{|I|} \cdot H\left(I_{j,s}^{+}\right)$$

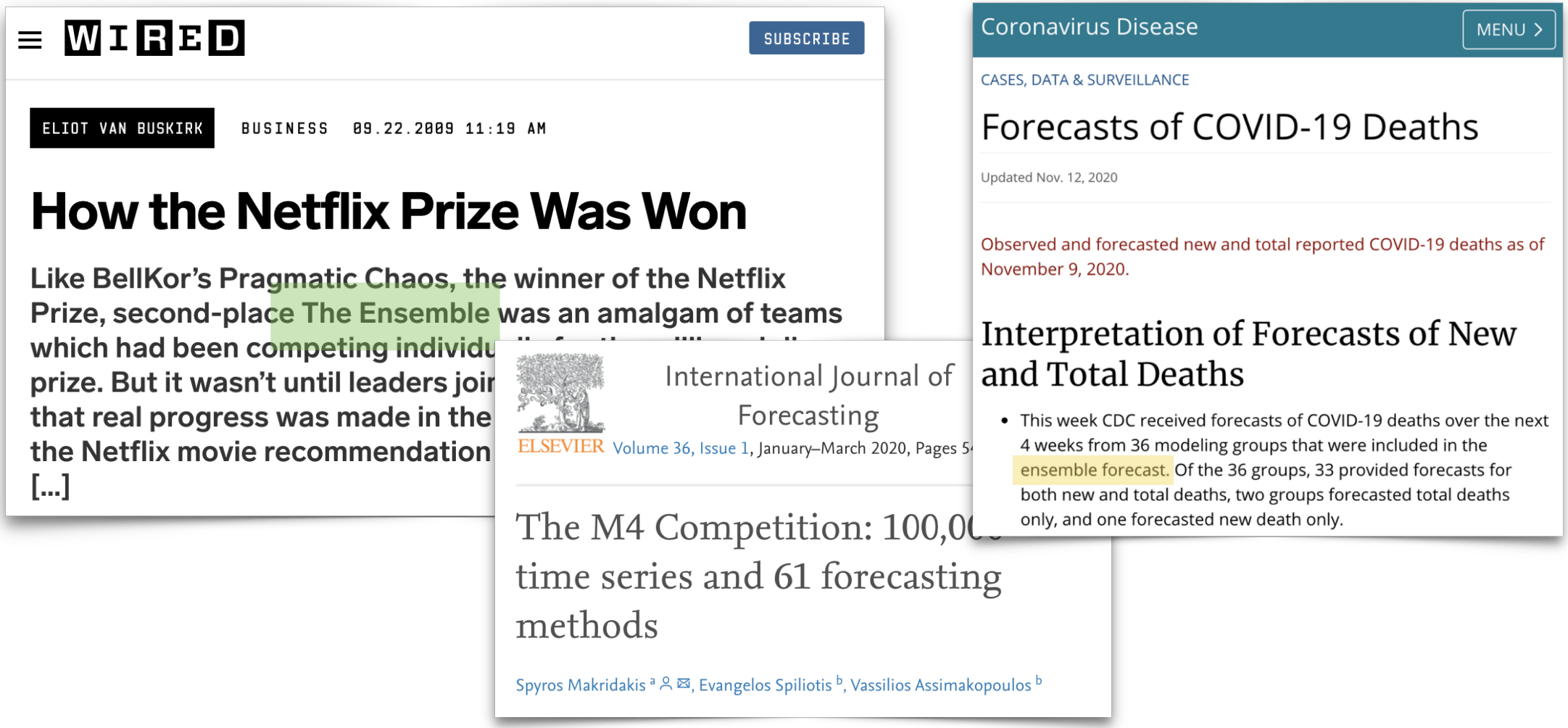
- $I = 9$, $\left|I_{j,s}^{-}\right| = 6$, $\left|I_{j,s}^{+}\right| = 3$
- So, $E_{j,s} = \frac{6}{9} H\left(I_{j,s}^{-}\right) + \frac{3}{9} H\left(I_{j,s}^{-}\right)$

$$H = -\sum_{\text{class } c} \hat{P}_c (\log_2 \hat{P}_c)$$

$$H\left(I_{j,s}^{-}\right) = -\left[\frac{3}{6}\log_2\left(\frac{3}{6}\right) + \frac{2}{6}\log_2\left(\frac{2}{6}\right) + \frac{1}{6}\log_2\left(\frac{1}{6}\right)\right]$$

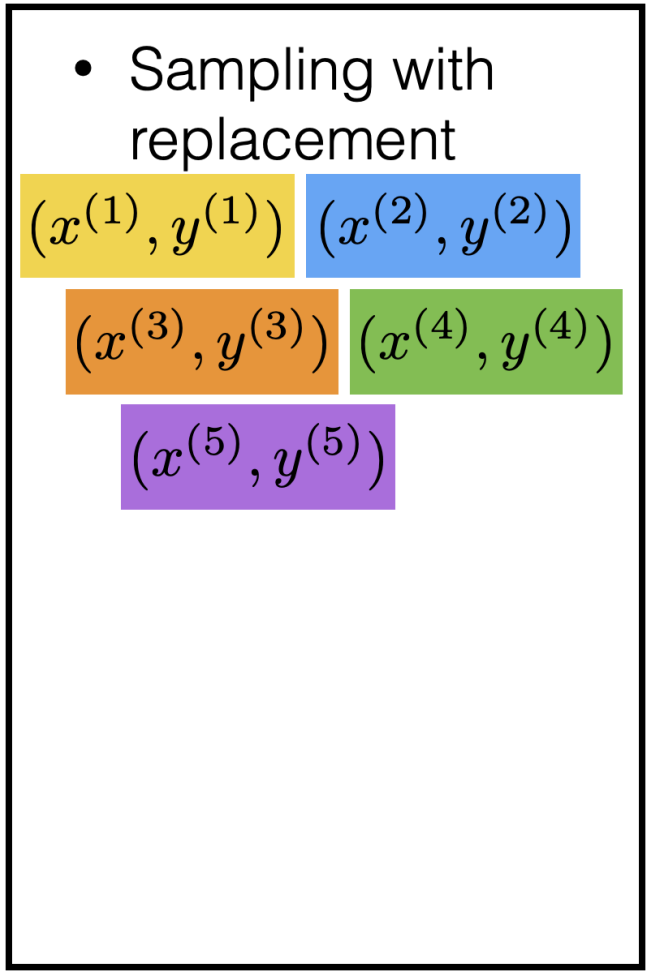
$$H\left(I_{j,s}^{+}\right) = -\left[\frac{1}{3}\ \log\left(\frac{1}{3}\right) + \frac{0}{3}\log_2\left(\frac{0}{3}\right) + \frac{2}{3}\log_2\left(\frac{2}{3}\right)\right]$$




WIRED
SUBSCRIBE
ELIOT VAN BUSKIRK
BUSINESS 09.22.2009 11:19 AM
How the Netflix Prize Was Won
Like BellKor's Pragmatic Chaos, the winner of the Netflix Prize, second-place The Ensemble was an amalgam of teams which had been competing individu
prize. But it wasn't until leaders joi
that real progress was made in the
the Netflix movie recommendation
[...]
ELSEVIER
International Journal of Forecasting
Volume 36, Issue 1, January–March 2020, Pages 5
The M4 Competition: 100,0
time series and 61 forecasting methods
Spyros Makridakis [a], Evangelos Spiliotis [b], Vassilios Assimakopoulos [b]
Coronavirus Disease
MENU
CASES, DATA & SURVEILLANCE
Forecasts of COVID-19 Deaths
Updated Nov. 12, 2020
Observed and forecasted new and total reported COVID-19 deaths as of November 9, 2020.
Interpretation of Forecasts of New and Total Deaths
This week CDC received forecasts of COVID-19 deaths over the next 4 weeks from 36 modeling groups that were included in the ensemble forecast. Of the 36 groups, 33 provided forecasts for both new and total deaths, two groups forecasted total deaths only, and one forecasted new death only.

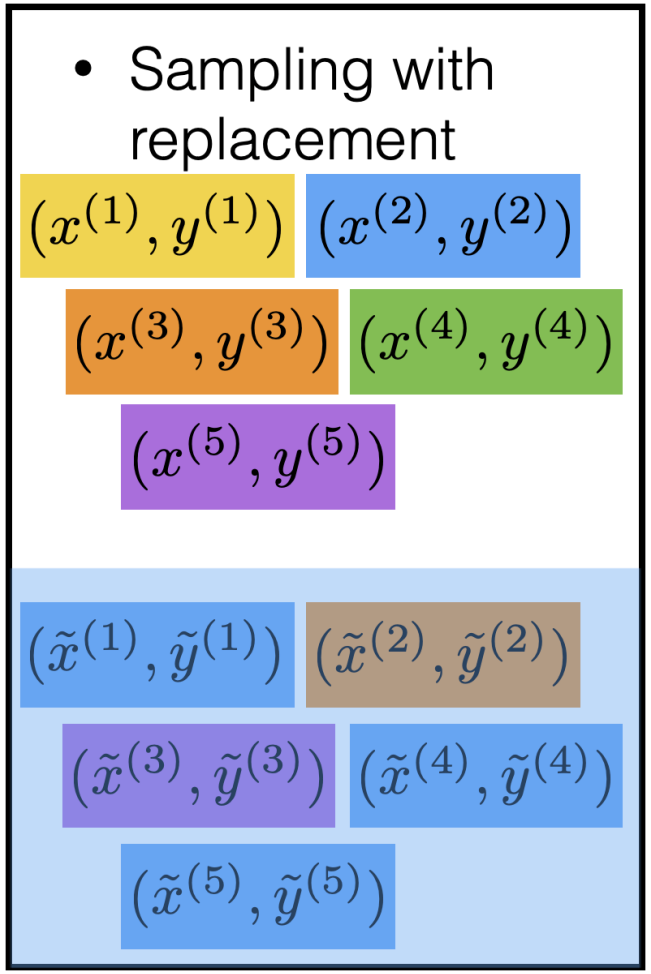
Bagging

- One of multiple ways to make and use an ensemble
- Bagging = **B**ootstrap **agg**regat**ing**
  - Training data $\mathcal{D}_n$



Bagging

- One of multiple ways to make and use an ensemble
- Bagging = **B**ootstrap **agg**regat**ing**
  - Training data $\mathcal{D}_n$
  - For $b = 1, \ldots, B$
    - Draw a new "data set" $\tilde{\mathcal{D}}_n^{(b)}$ of size n by sampling with replacement from $\mathcal{D}_n$
    - Train a predictor $\hat{f}^{(b)}$ on $\tilde{\mathcal{D}}_n^{(b)}$
  - Return
    - For regression: $\hat{f}_{\text{bag}}(x) = \frac{1}{B}\sum_{b=1}^{B} \hat{f}^{(b)}(x)$
    - For classification: predictor at a point is class with highest vote count at that point



Outline

- Recap (transforermers)
- Non-parametric models
  - interpretability
  - ease of use/simplicity
- Decision tree
  - Terminologies
  - Learn via the BuildTree algorithm
    - Regression
    - Classification
- Nearest neighbor

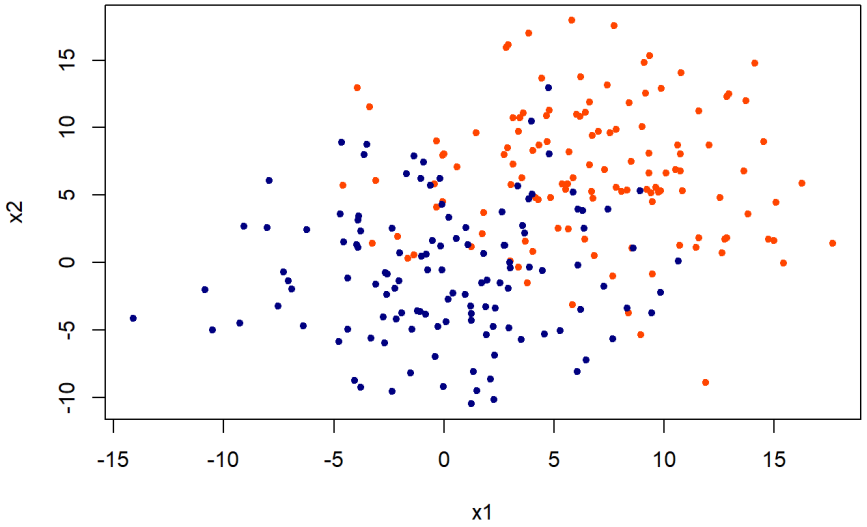
Nearest neighbor classifier

- Hyperparameter: k
- Also need
  - Distance metric (typically Euclidean or Manhattan distance)
  - A tie-breaking scheme (typically at random)

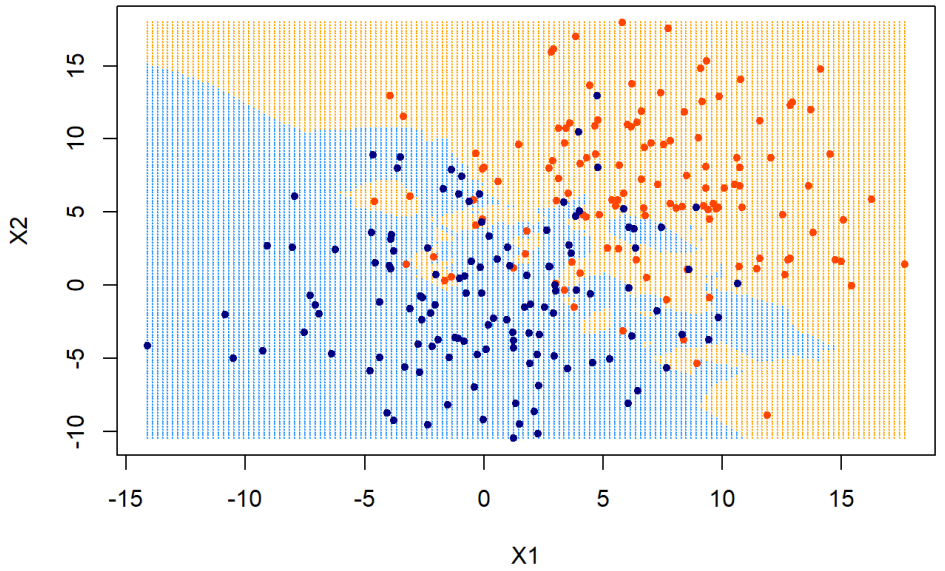
Training: None (or rather: memorize the entire training data)

Predicting/testing:

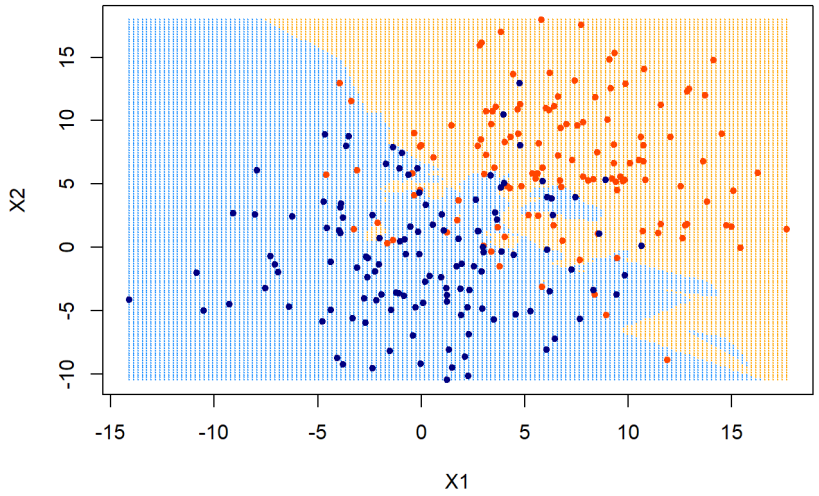
- for a new data point x_{new} do:
  - find the k points in training data nearest to x_{new}
    - For classification: predict label $\hat{y_{new}}$ for x_{new} by taking a majority vote of the k neighbors's labels y
    - For regression: predict label $\hat{y_{new}}$ for x_{new} by taking an average over the k neighbors' labels y



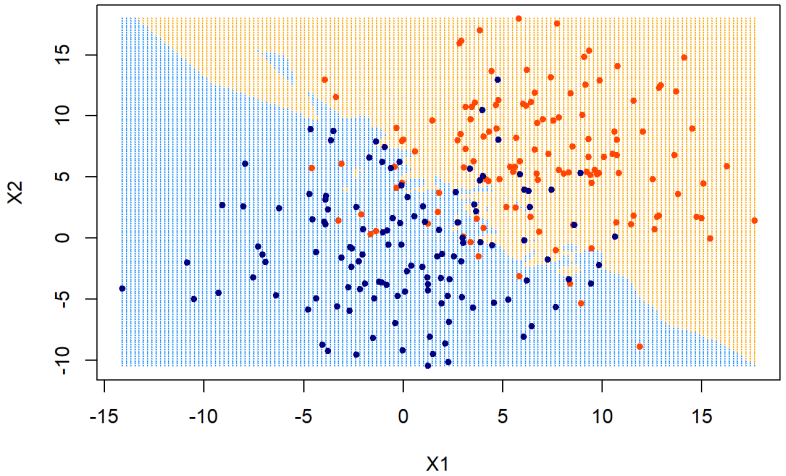
KNN = 3



KNN = 5



KNN = 10



We'd love it for you to share some lecture feedback.

Thanks!