Decision Trees

Week Goals

- Decision Trees: training and application
- Overfitting in Decision Trees
- Working with categorical features

Example: Predicting the Apartment Price

- **Target**: price of an apartment

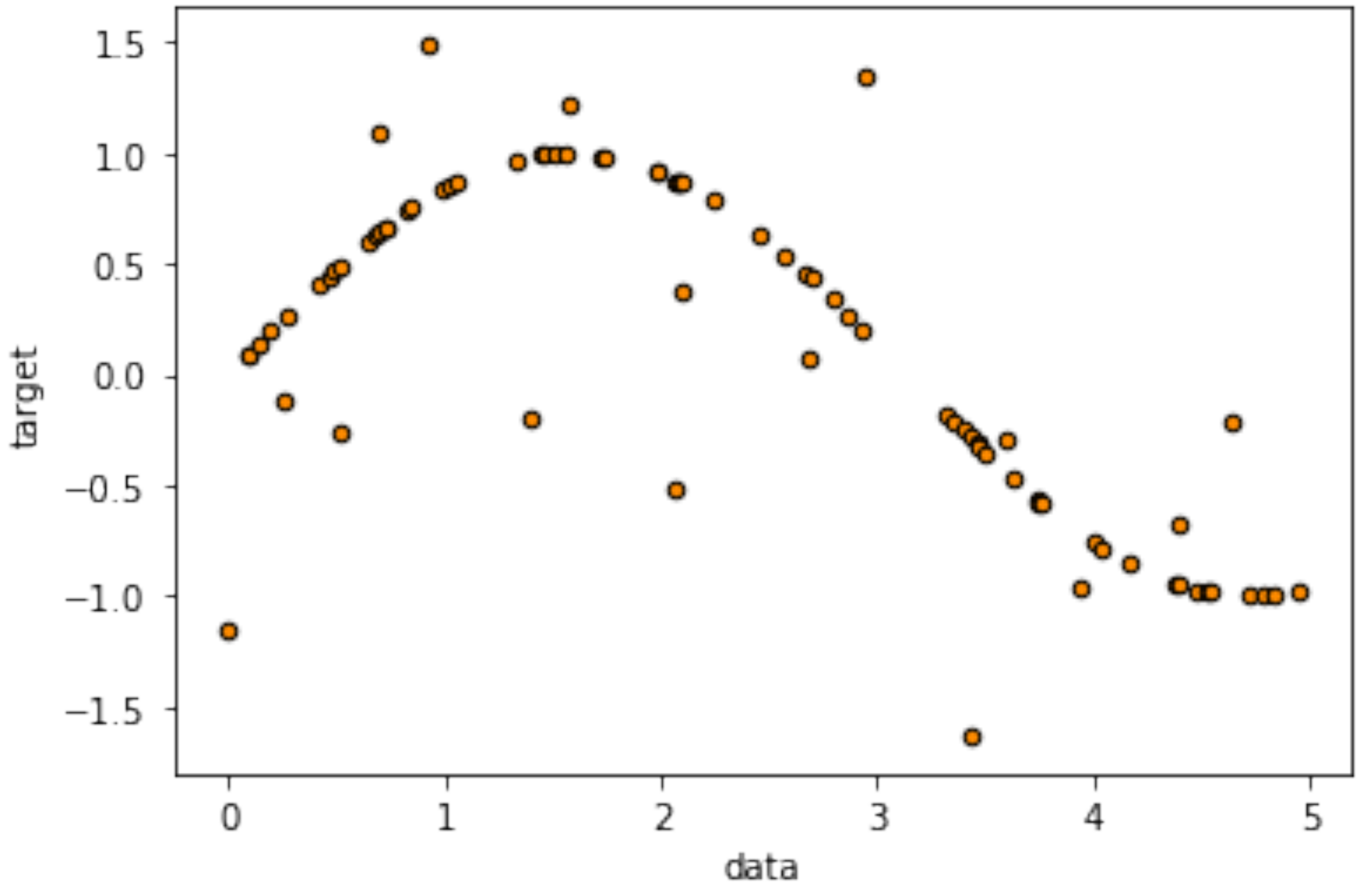
- **Features**: Area, Floor, Distance to nearest metro station, etc.

Example: Predicting the Apartment Price

- Linear Model:

$$a(x) = w_0 + w_1 * (area) + w_2 * \left(floor\right) + w_3 * (dist.\ ) + \ldots$$

- It is likely that the dependency is not linear



Example: Predicting the Apartment Price

- Linear Model:

$$a(x) = w_0 + w_1 * (area) + w_2 * \left(floor\right) + w_3 * (dist.\ ) + \ldots$$

- It is likely that the features are interconnected

Example: Predicting the Apartment Price

- Linear model with polynomial features:

$$a(x) = w_0 + w_1 * (area) + w_2 * \left(floow\right) \; + w_3 * (\,dist\,.)$$

$$+ w_4 * (area)^2 + w_5 * \left(floor\right)^2 + w_6 * (dist\,.)^2$$

$$+ w_7 * (area) * \left(floor\right) + \ldots$$

Example: Predicting the Apartment Price

- Linear model with polynomial features:

$$a(x) = w_0 + w_1 * (area) + w_2 * \left(floow\right) + w_3 * (\ dist\ .)$$

$$+ w_4 * (area)^2 + w_5 * \left(floor\right)^2 + w_6 * (dist\ .)^2$$

$$+ w_7 * (area) * \left(floor\right) + \ldots$$

- It maybe hard to interpret the result

- What does $(dist\ .) * \left(floor\right)^2$ mean?

Example: Predicting the Apartment Price

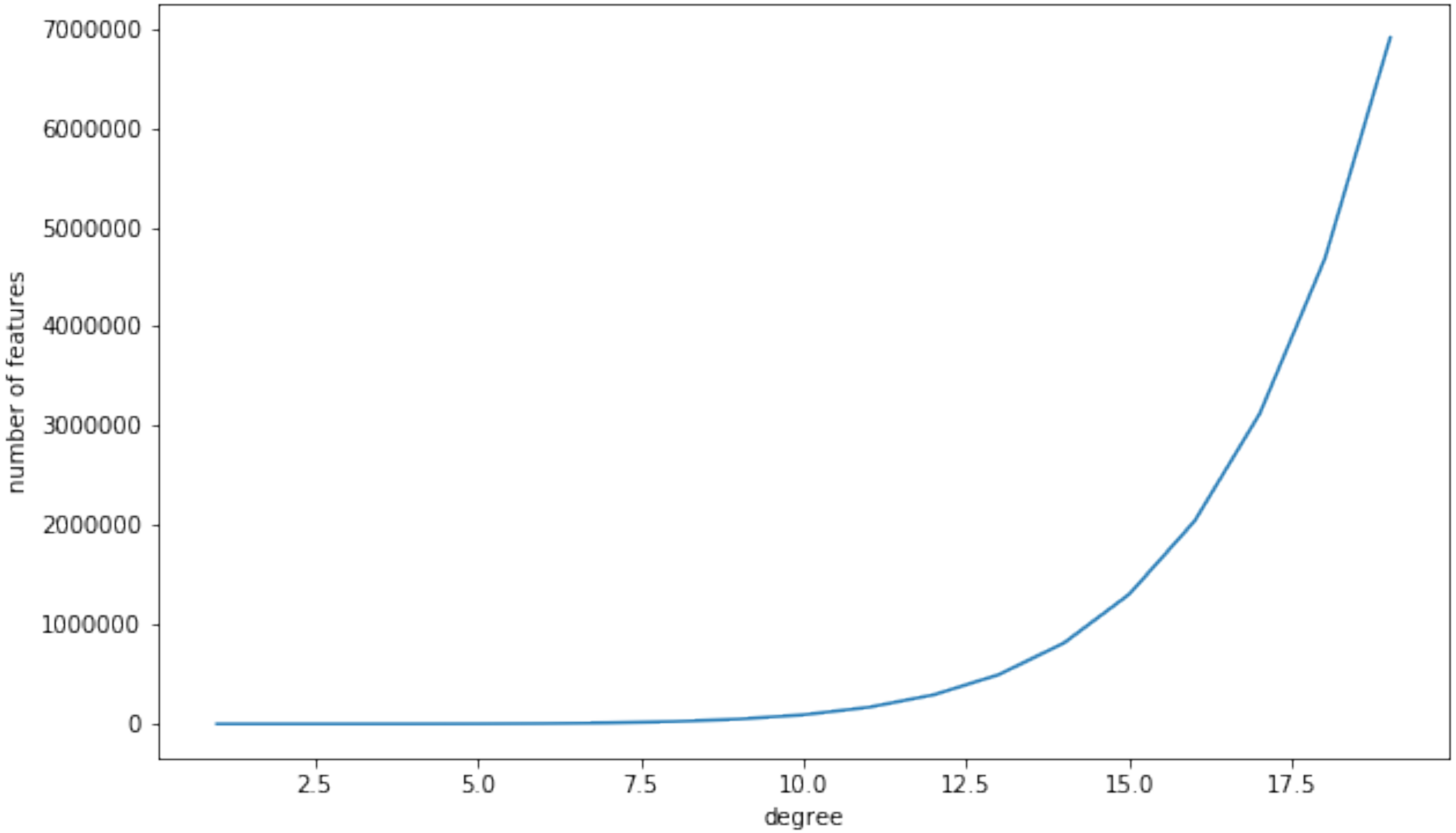
- Linear model with polynomial features:

$$a(x) = w_0 + w_1 * (area) + w_2 * \left(floow\right) + w_3 * (\,dist\,.\,)$$
$$+w_4 * (area)^2 + w_5 * \left(floor\right)^2 + w_6 * (dist\,.\,)^2$$
$$+w_7 * (area) * \left(floor\right) + \ldots$$

- Assume we had 10 features
- Polynomial of power 2 → 55 extra features
- Polynomial of power 3 → 220 extra features

Example: Predicting the Apartment Price

Linear model with polynomial features:



Example: Predicting the Apartment Price

- Add intervals of the features:

$$a(x) = w_0 + w_1 * [30 < area < 50] + w_2 * [50 < area < 80]$$

$$+ w_{20} * [2 < floor < 5] + \dots$$

$$+ w_{100} * [30 < area < 50][2 < floor < 5] + \dots$$

- It is easier to interpret features:

$$[30 < area < 50][2 < floor < 5]$$

- But we will have even more features!

Summary

- We need an efficient algorithm to find non-linear dependency between features and target variables

- Decision trees will allow us to do that

Decision Trees

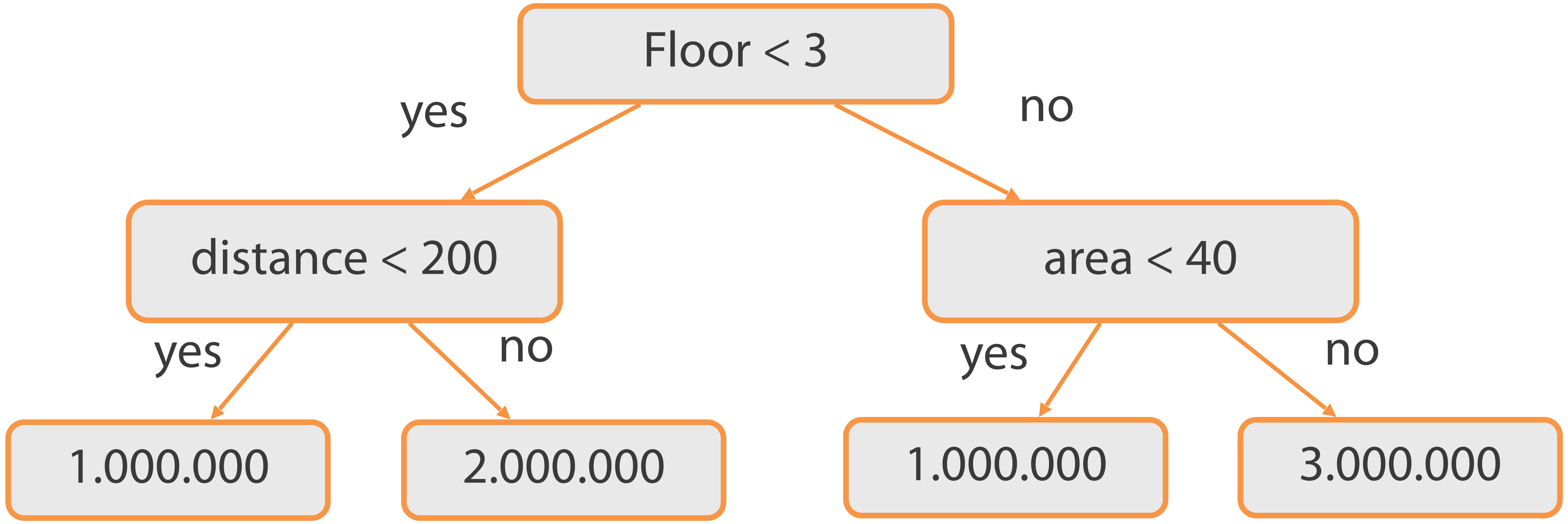
Logical Rule

$$[30 < area < 50]\,[2 < floor < 5]\,[500 < dist. < 1000]$$

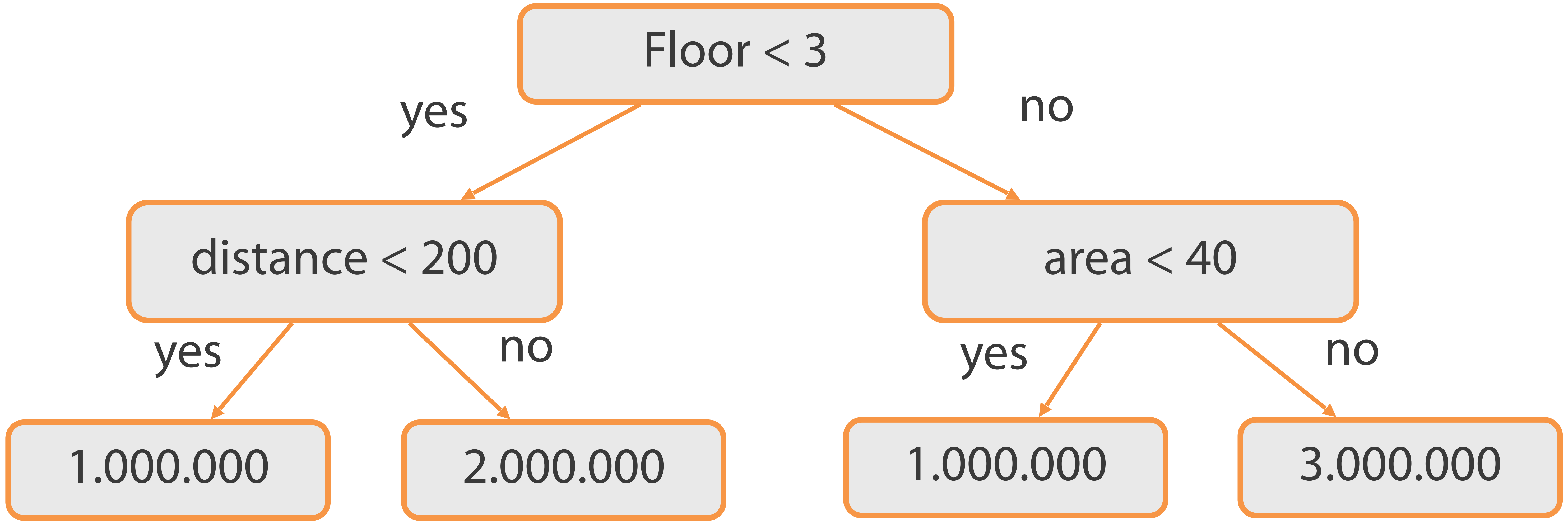
- Easy to explain
- Find non-linear dependencies

- We need to find good logical rules
- We need to build models out of them

Decision Trees

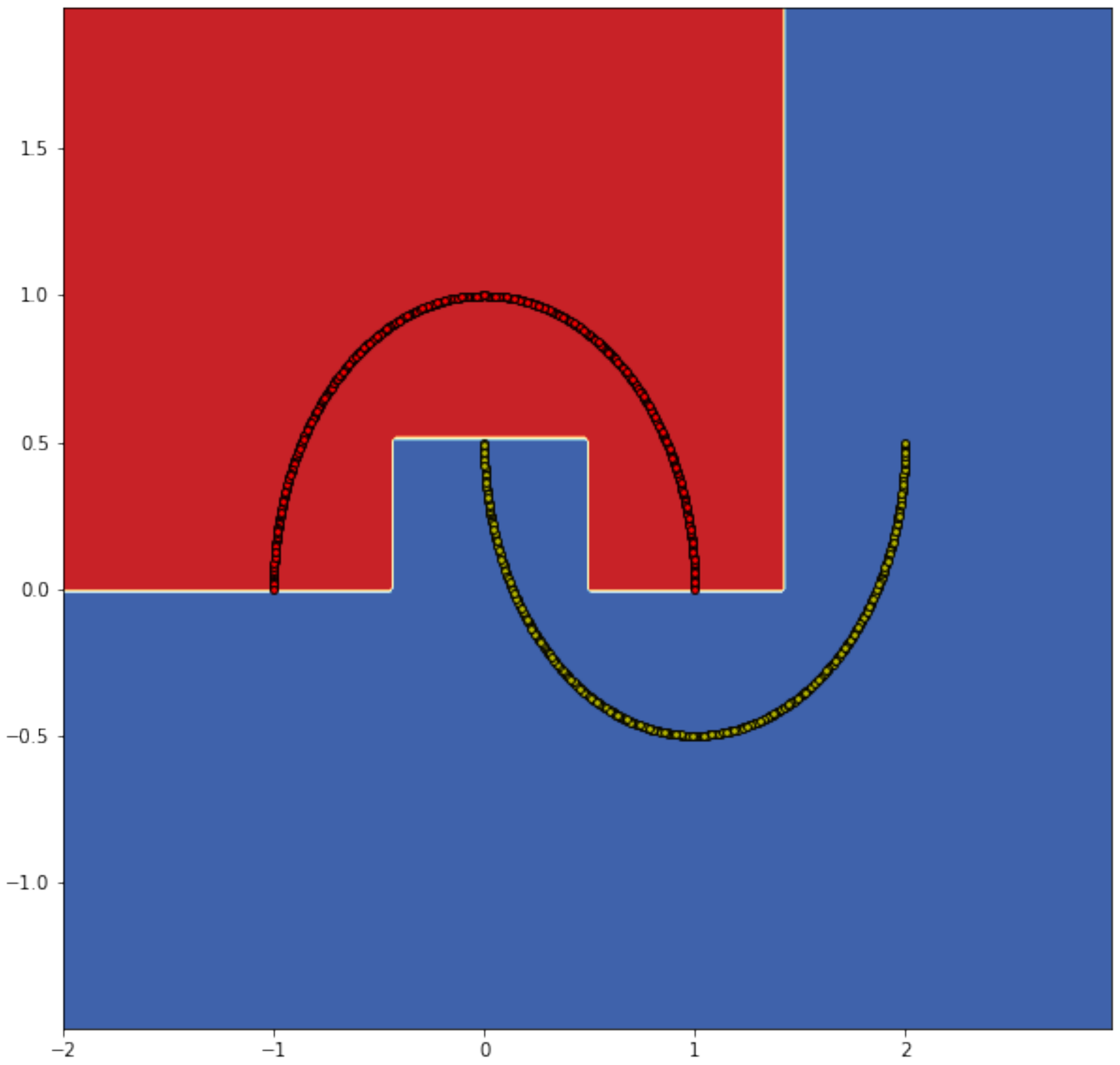


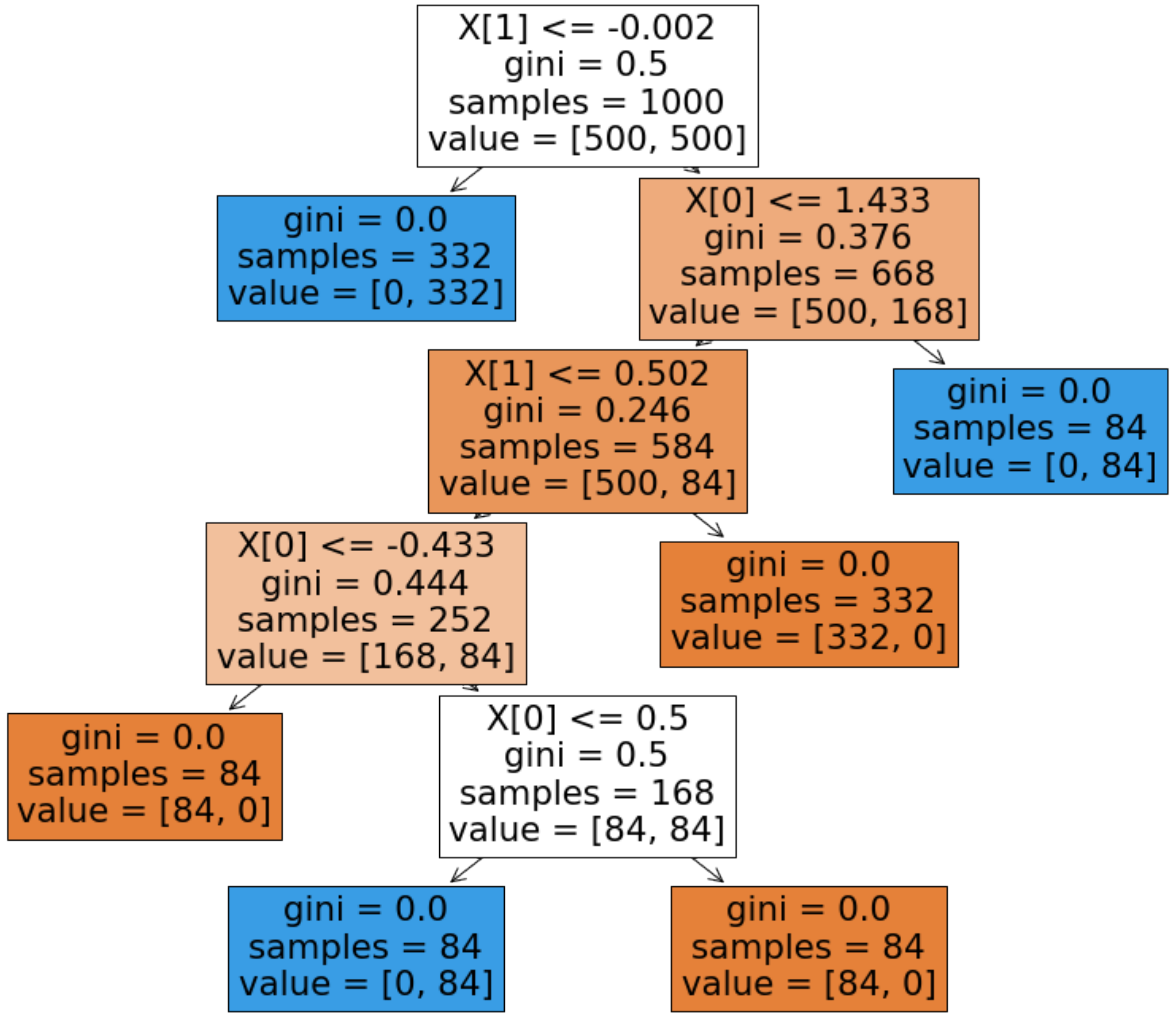
Decision Trees



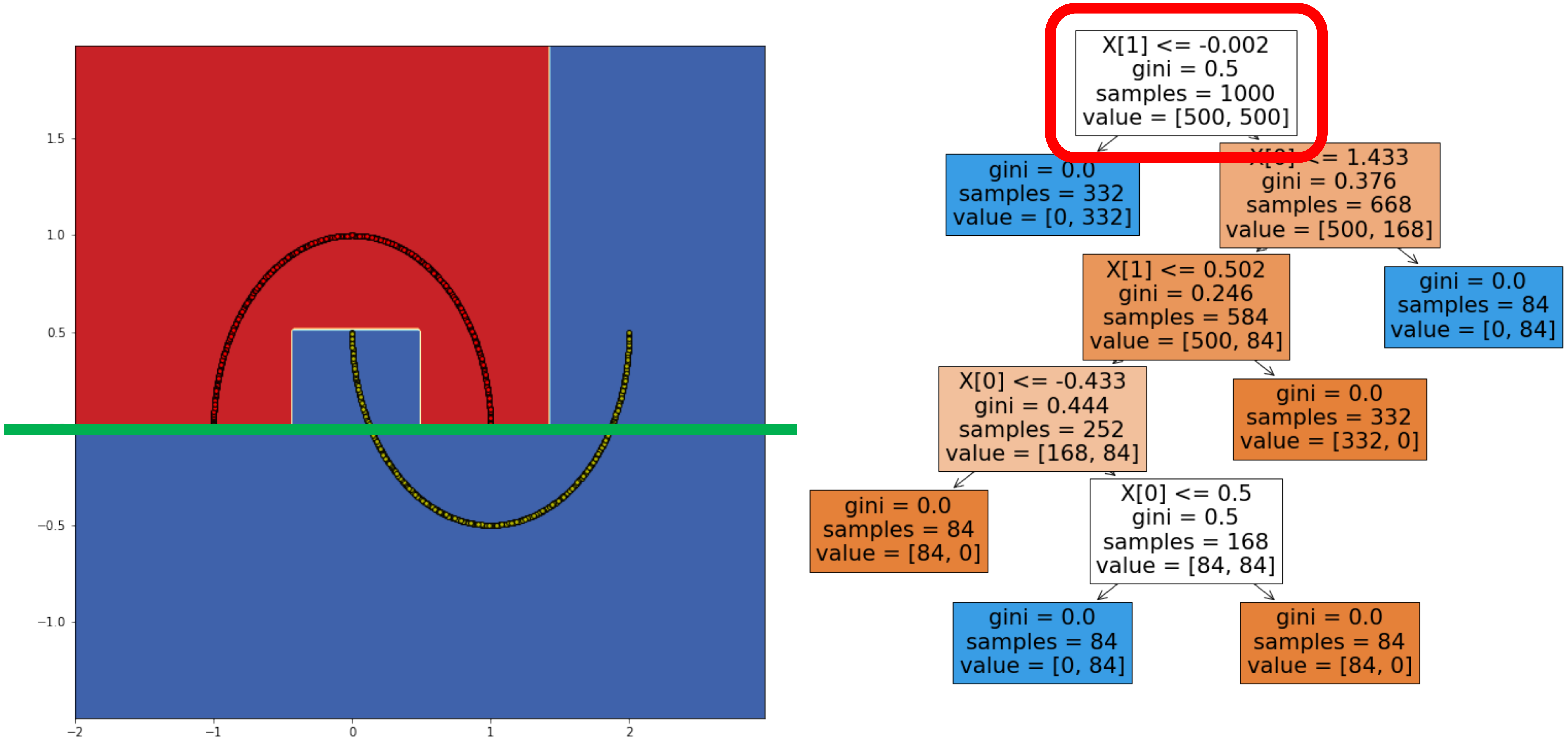
- **Internal Nodes**: splitting criterion $\left[x_j < t\right]$
- **Leaves**: predictions $c \in \mathbb{Y}$

Decision Trees

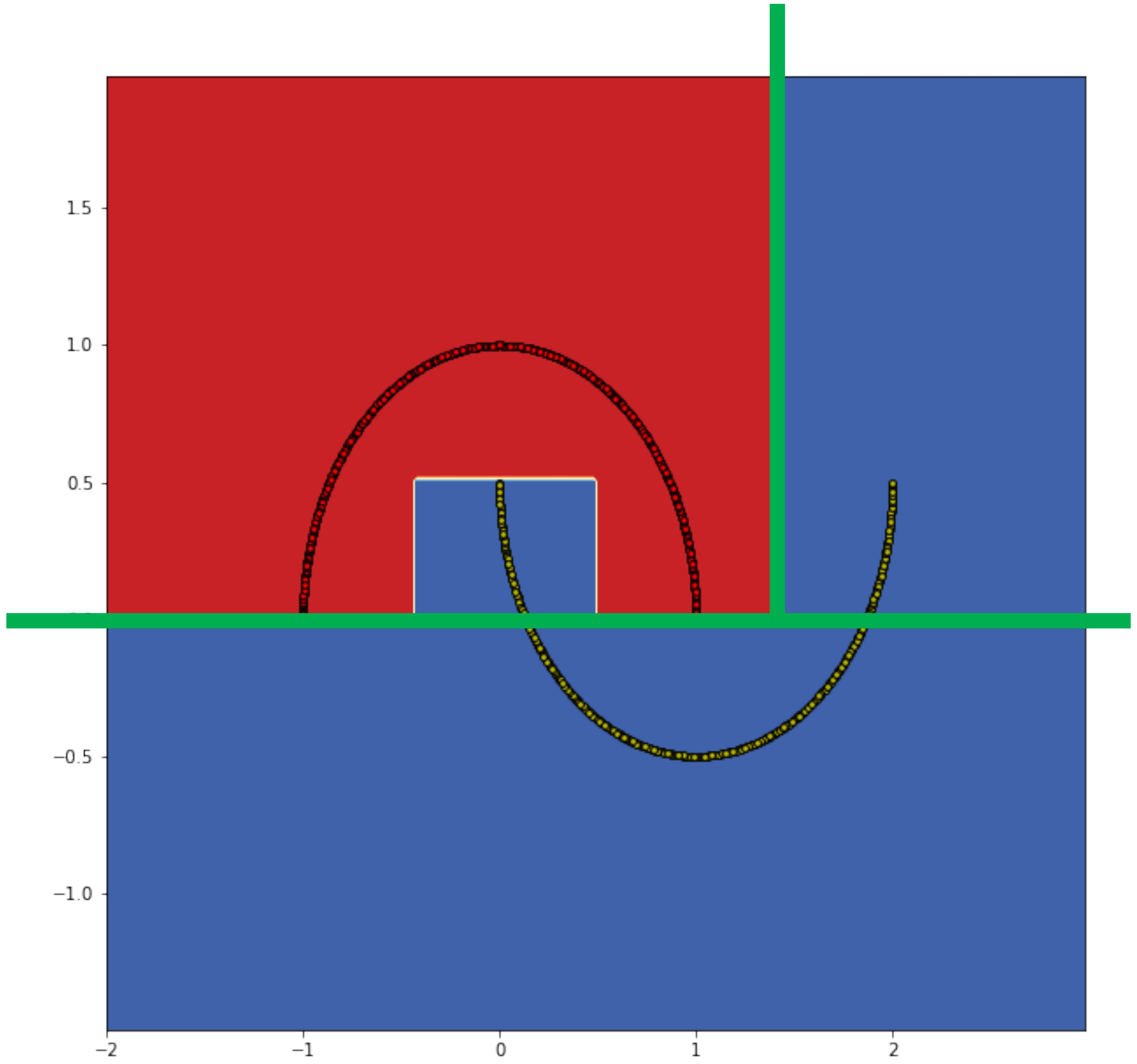


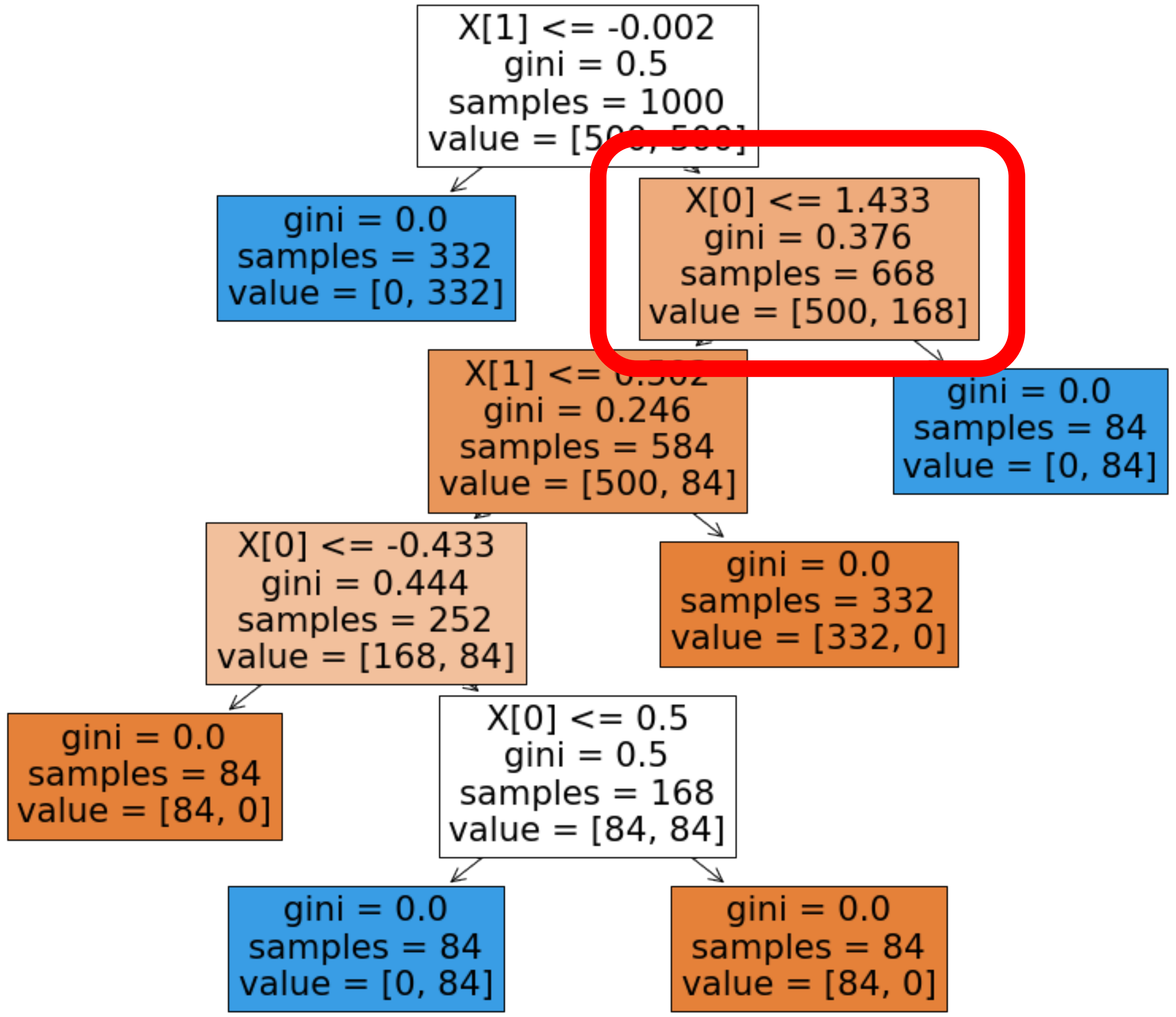


Decision Trees

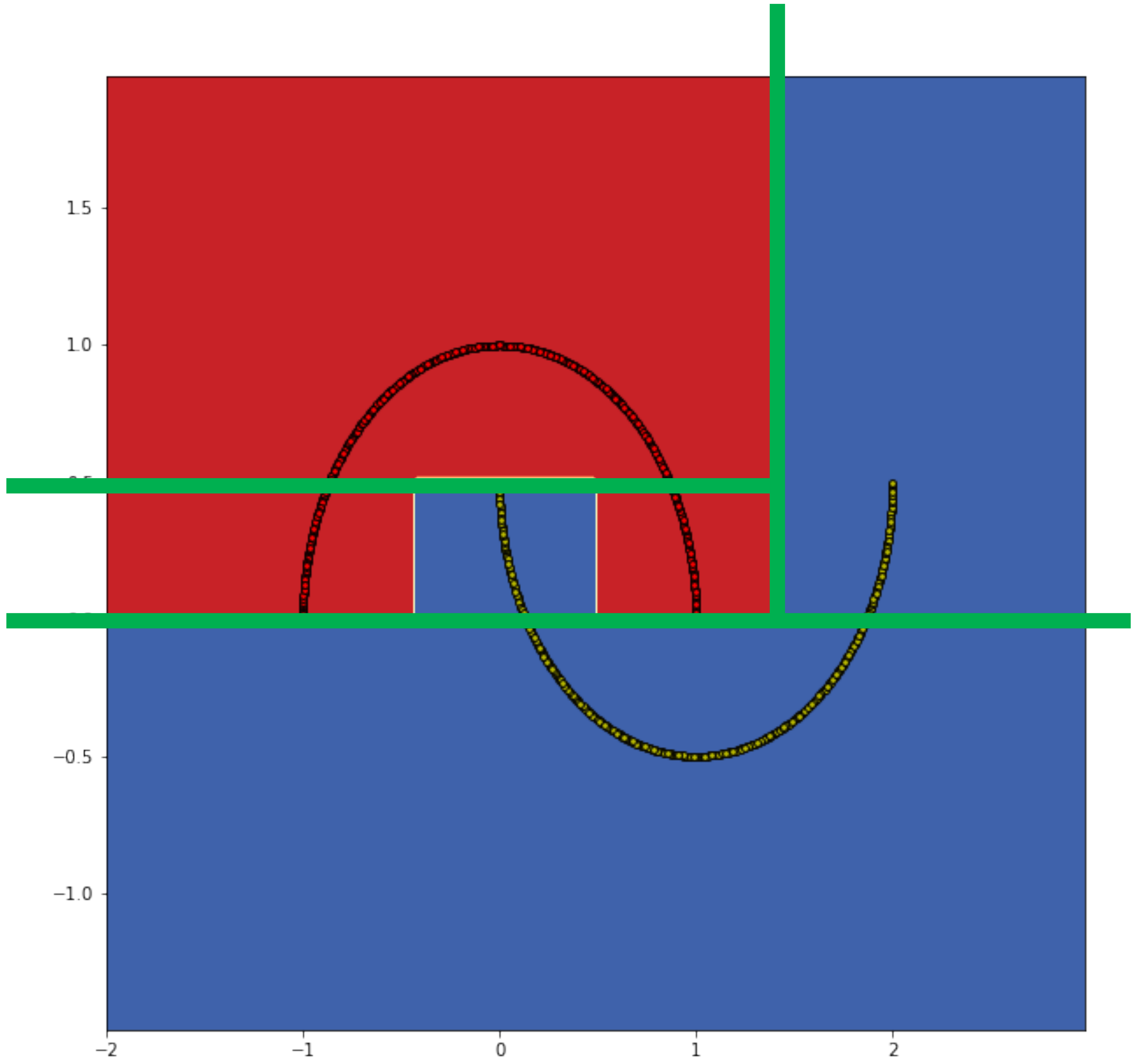


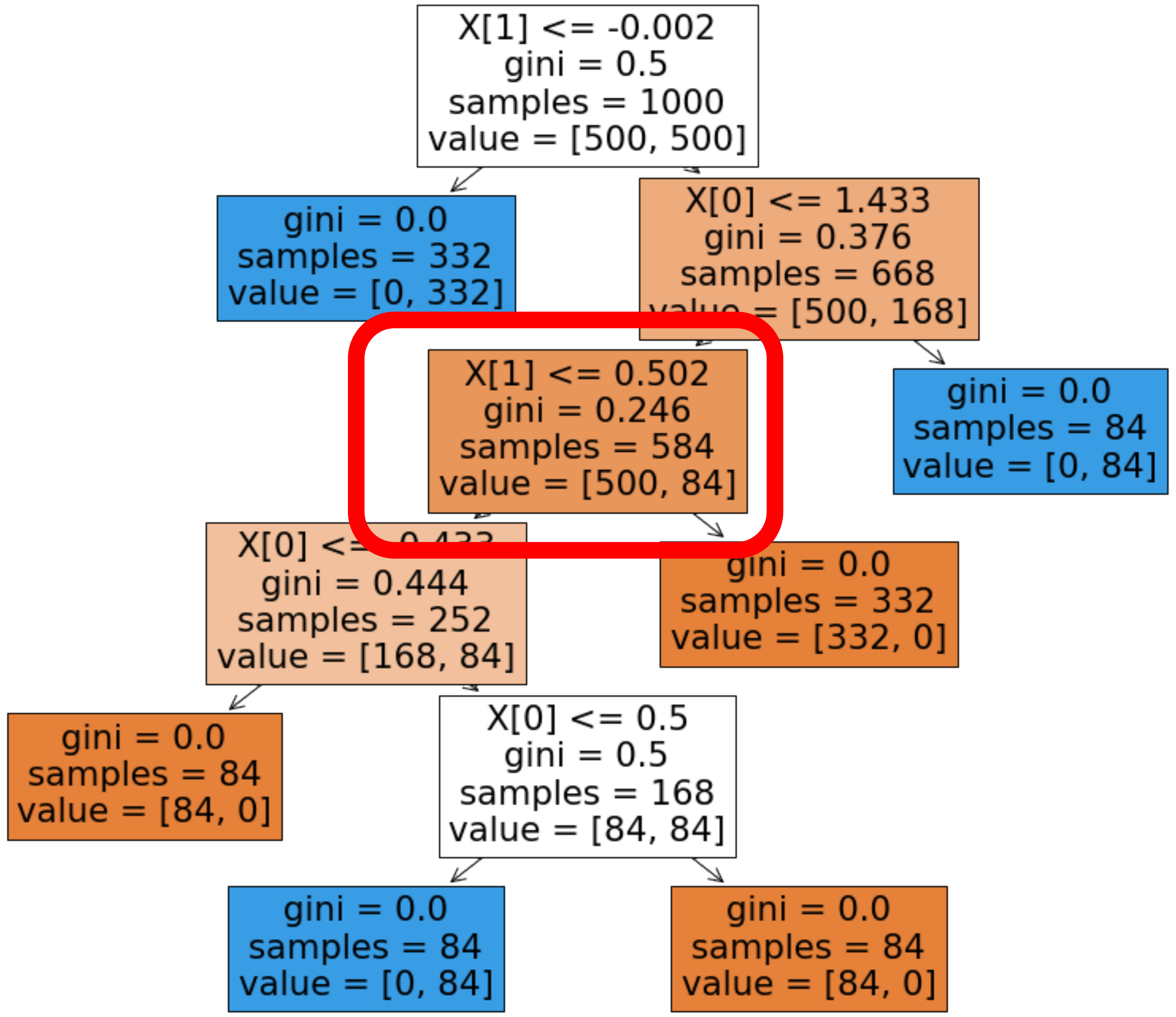
Decision Trees



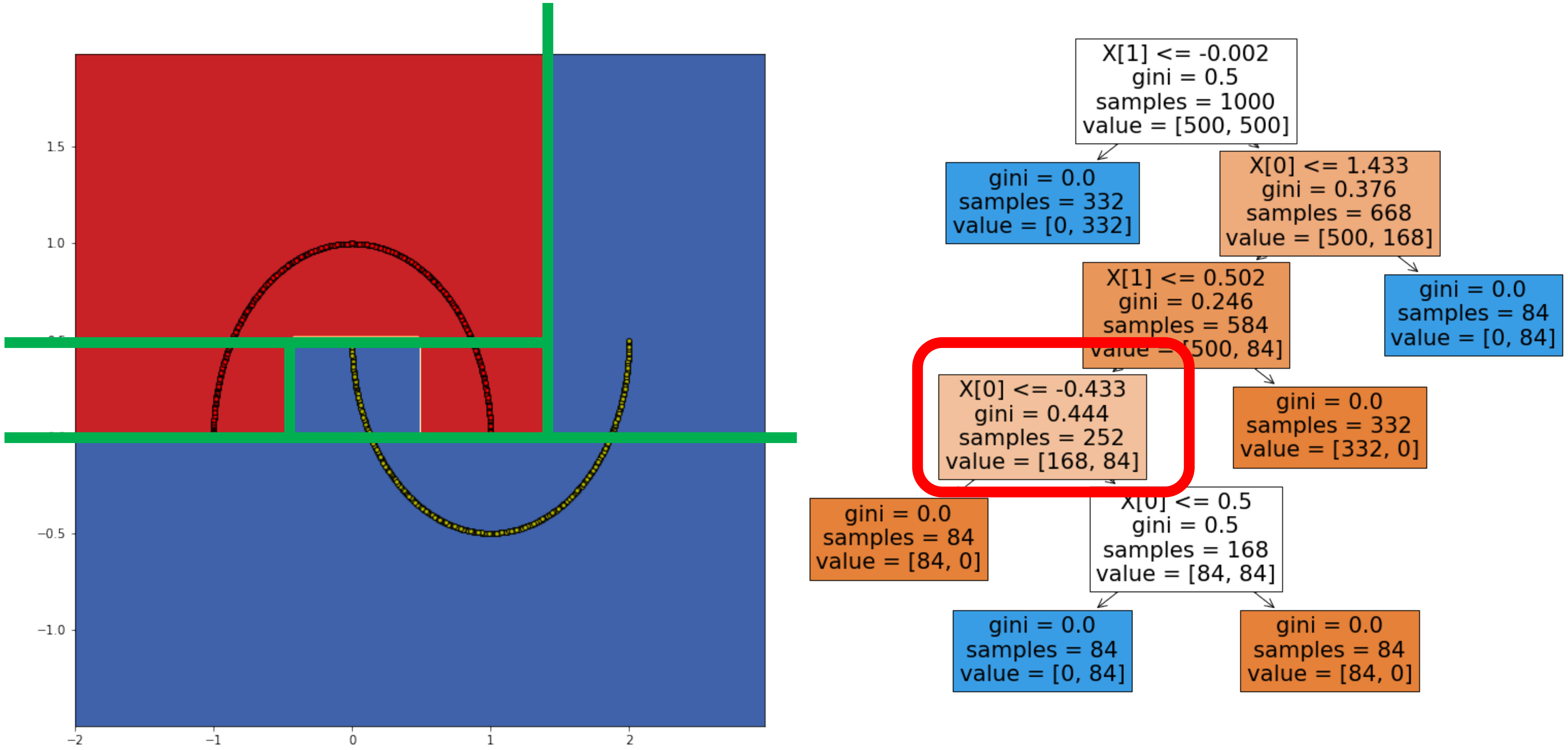


Decision Trees

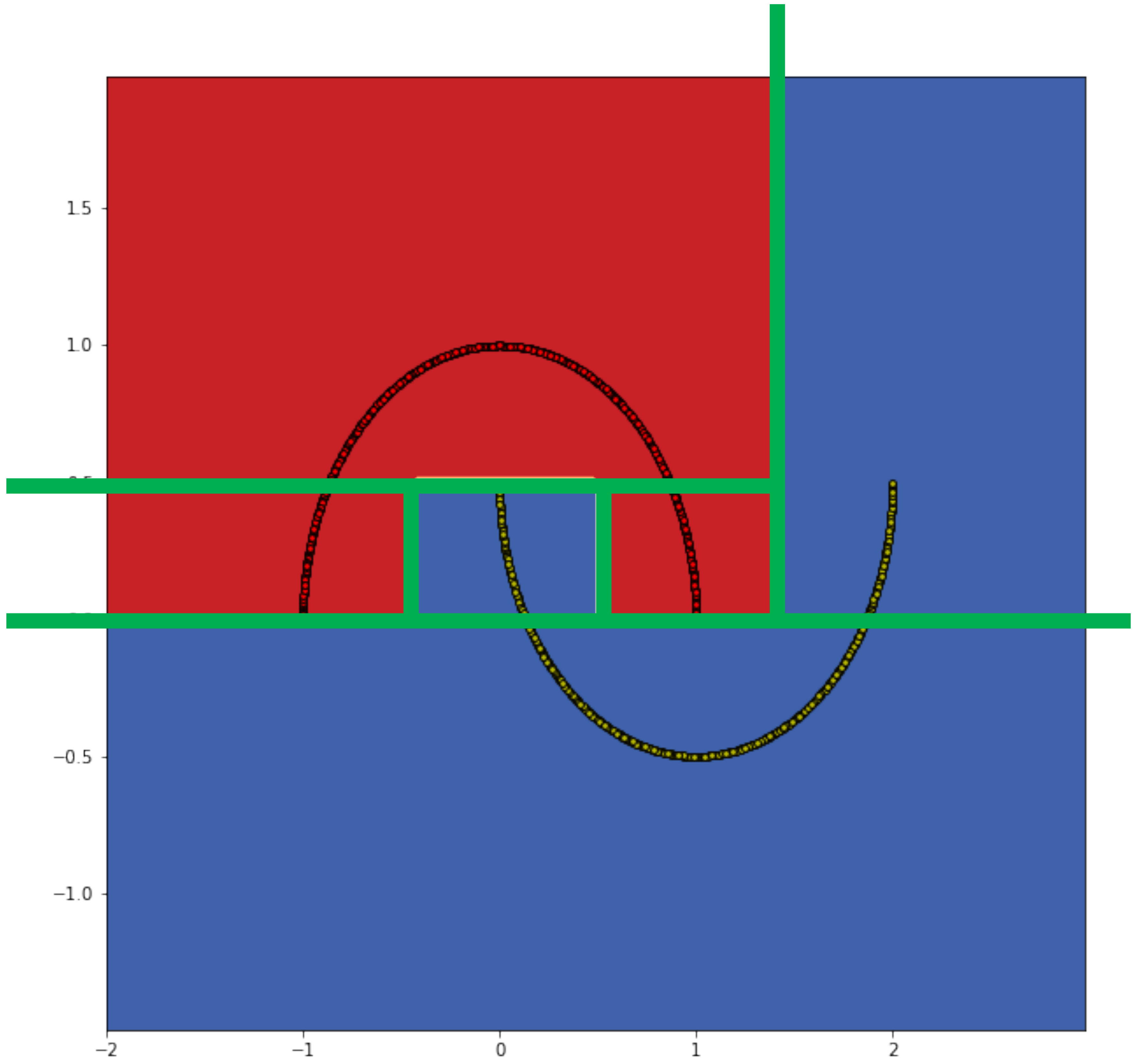


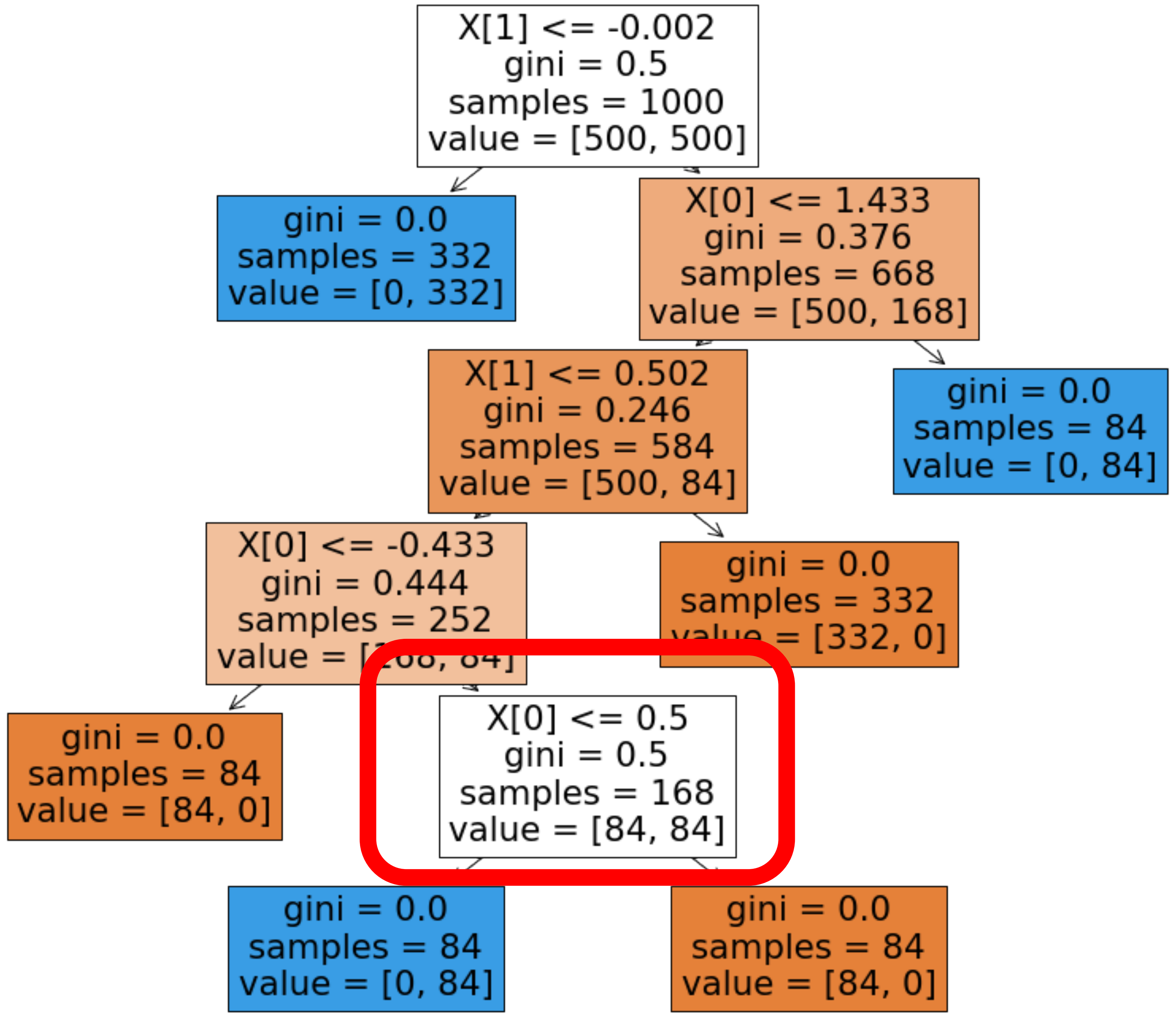


Decision Trees

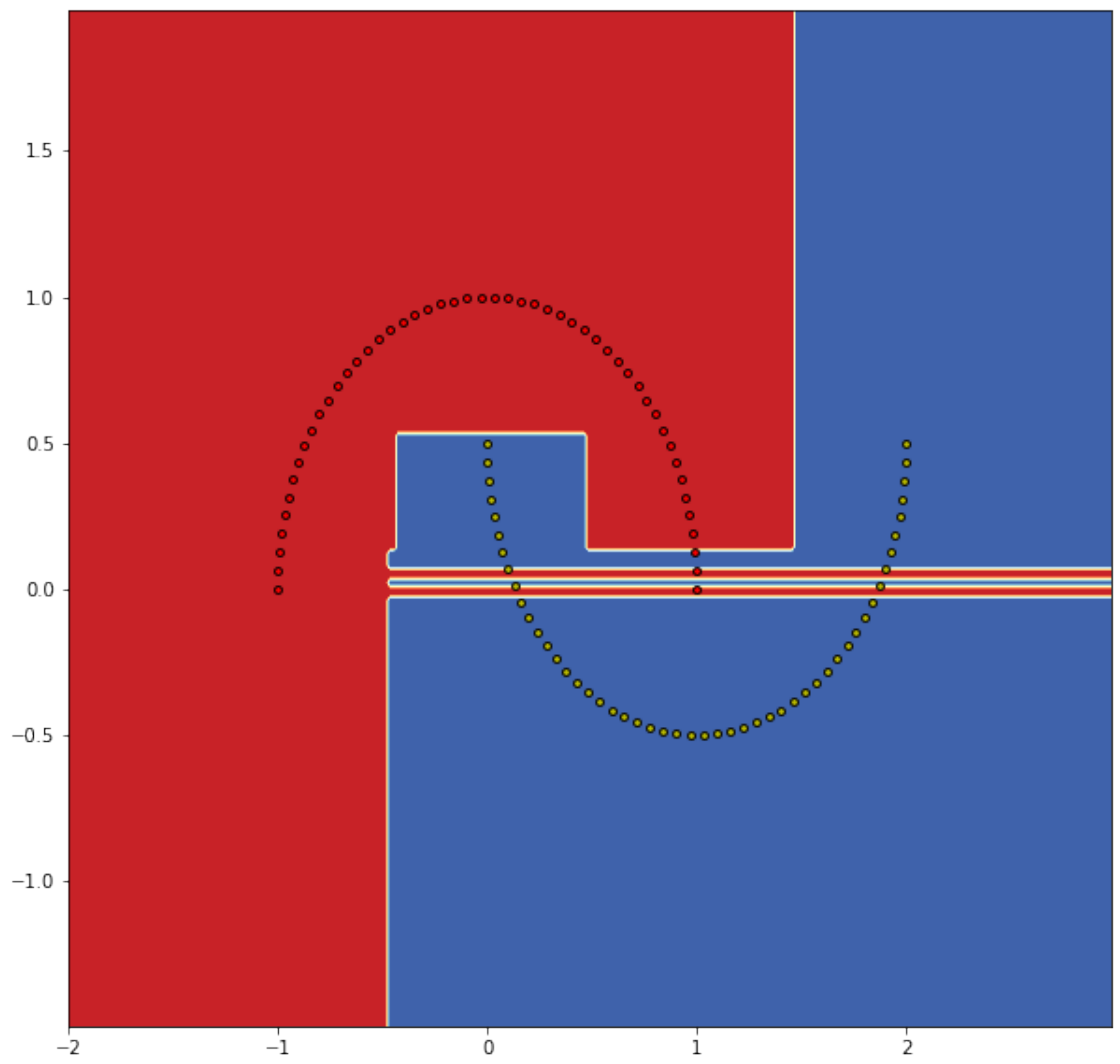


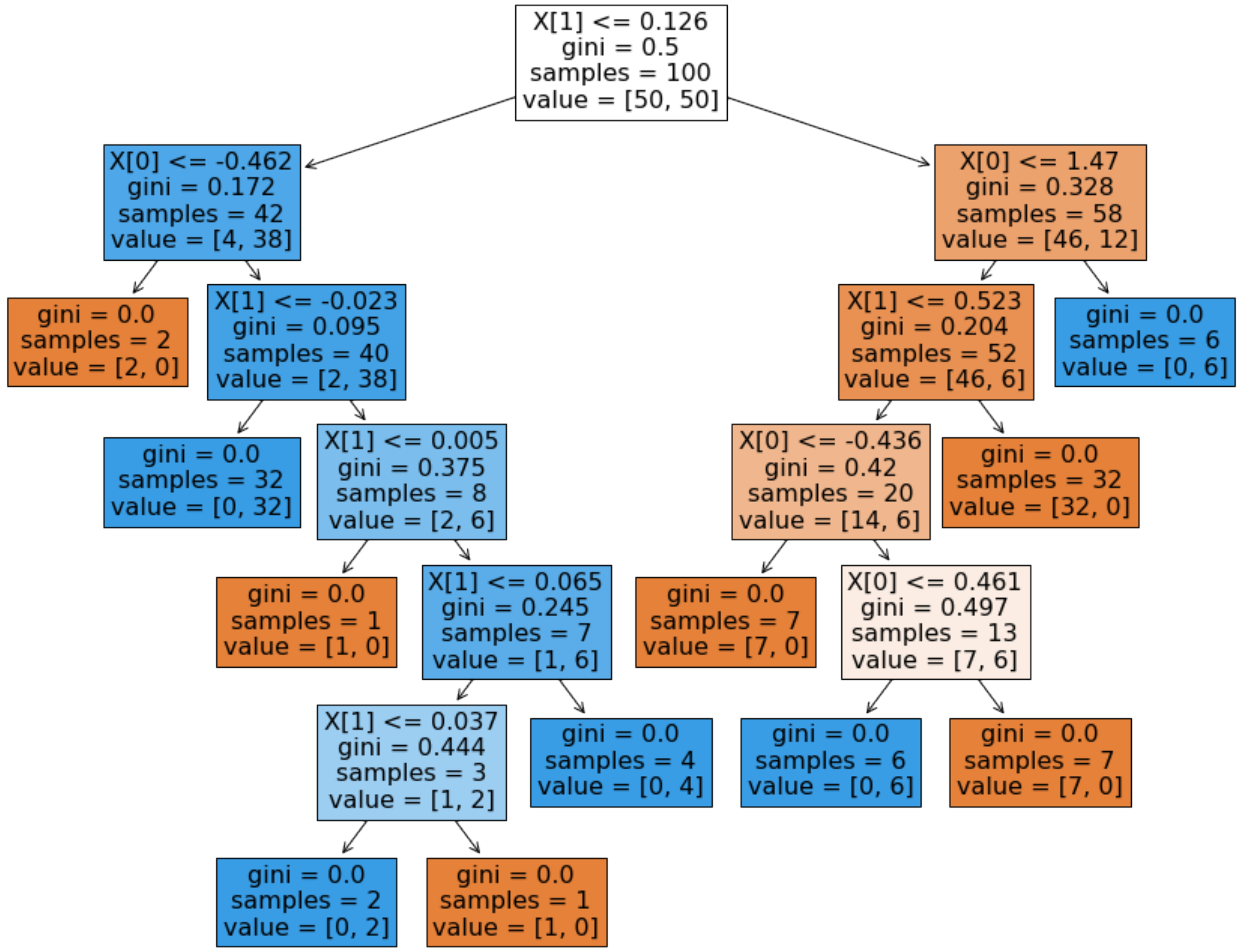
Decision Trees





Decision Trees and Overfitting





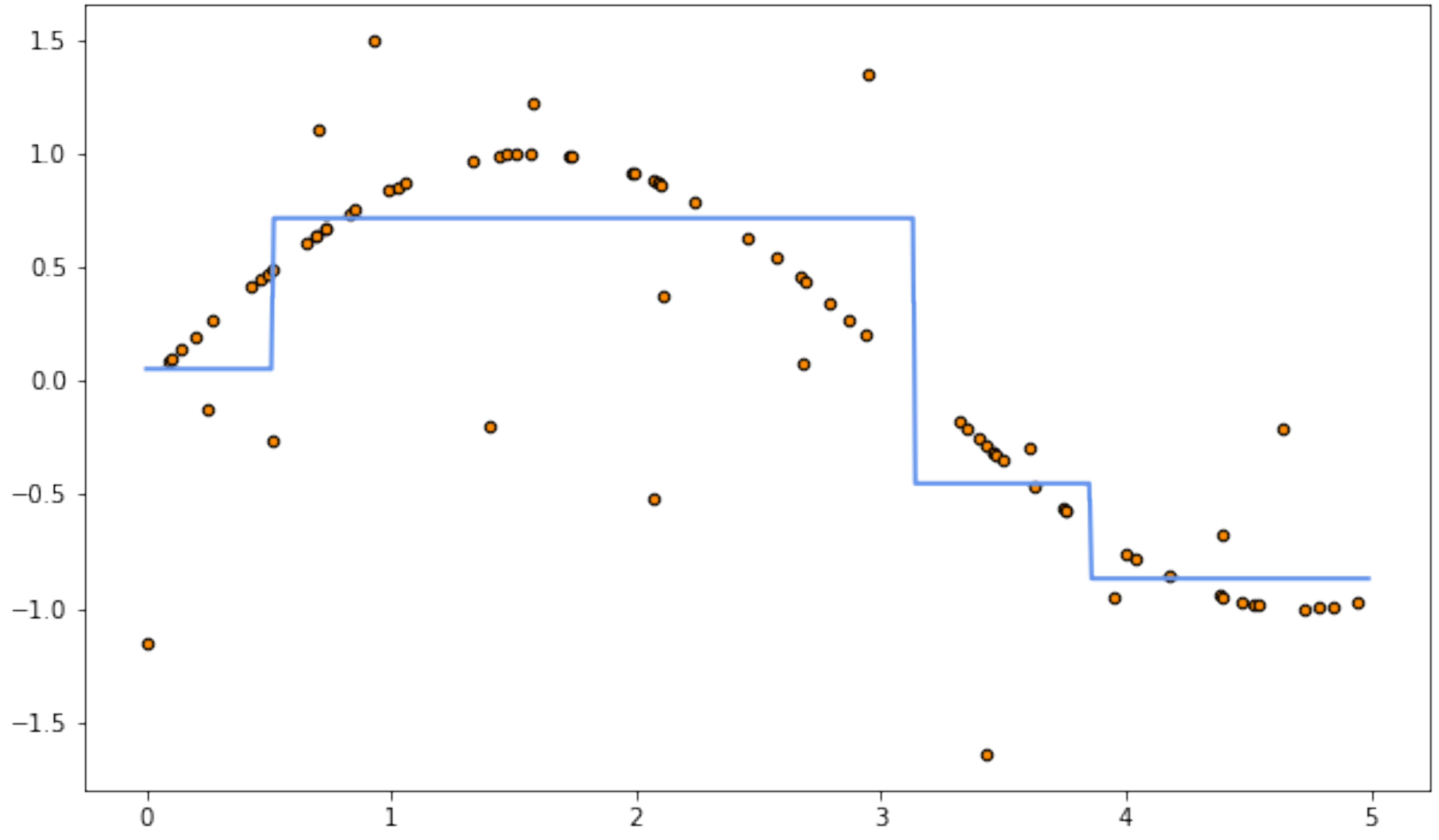
Complexity of Decision Trees

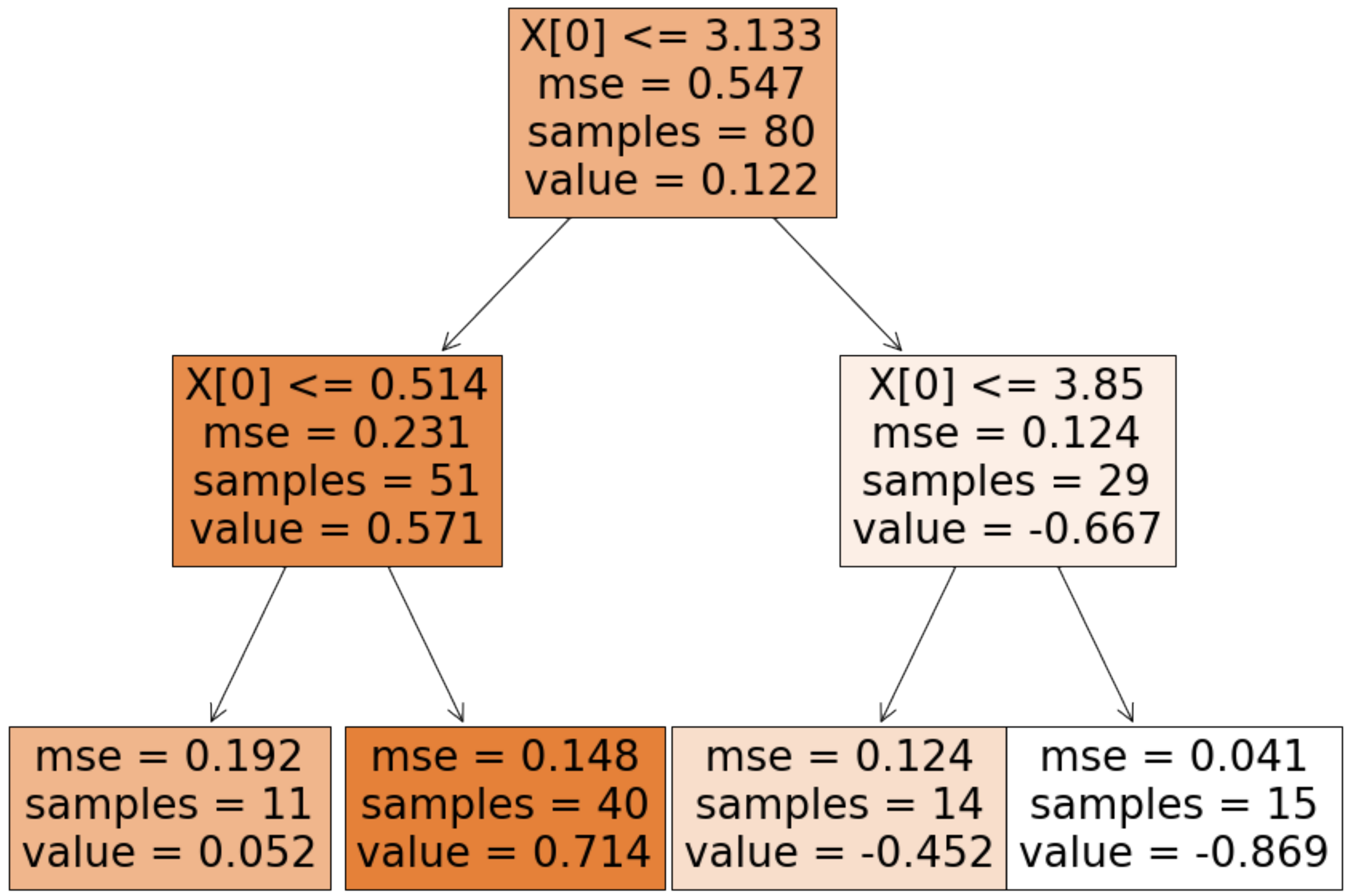
- We can keep splitting until we have only one object in each leaf

- We can ideally fit **any** training data

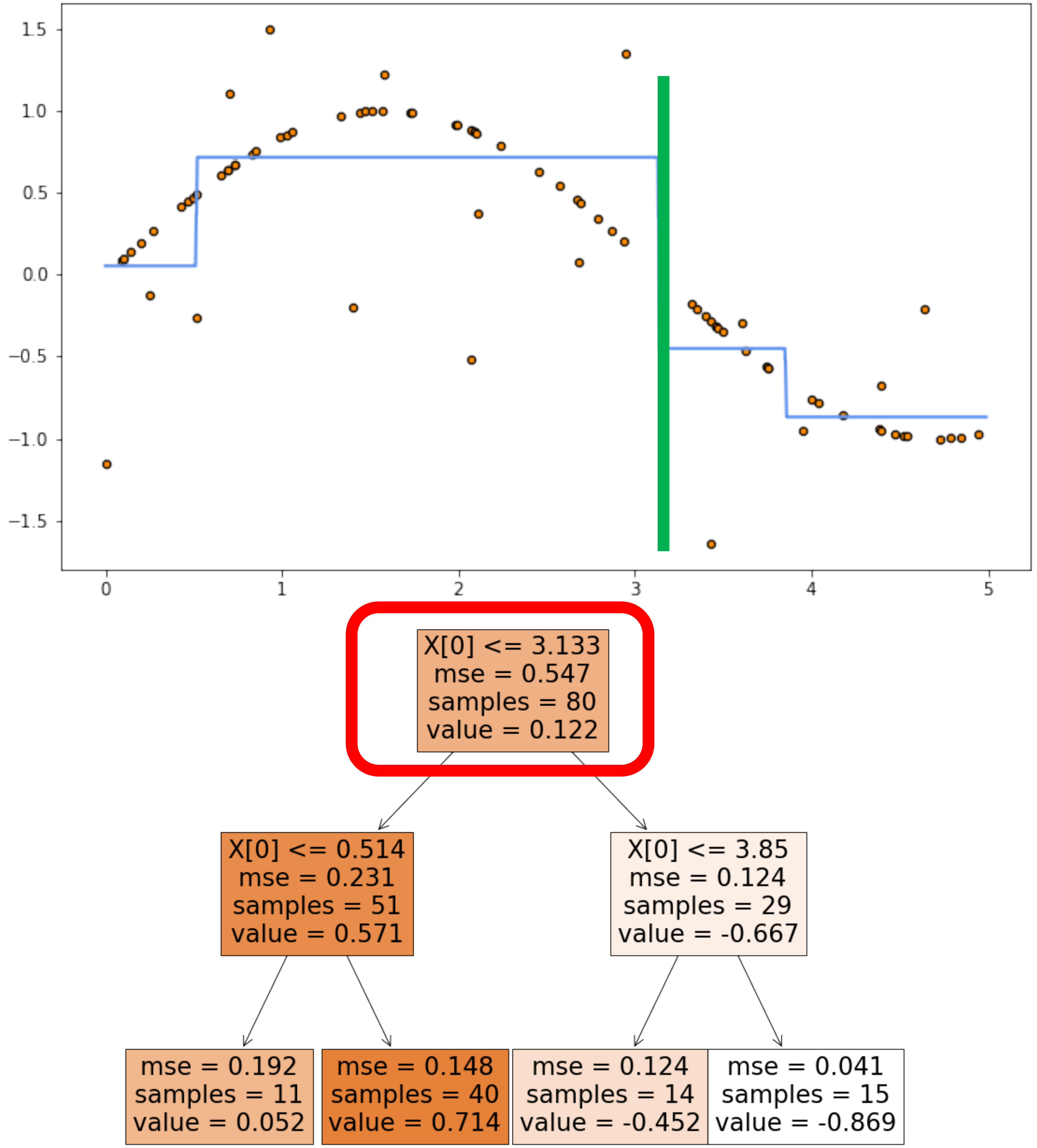
- Unless there are several objects with the same features and different target values

Decision Trees for Regression Task

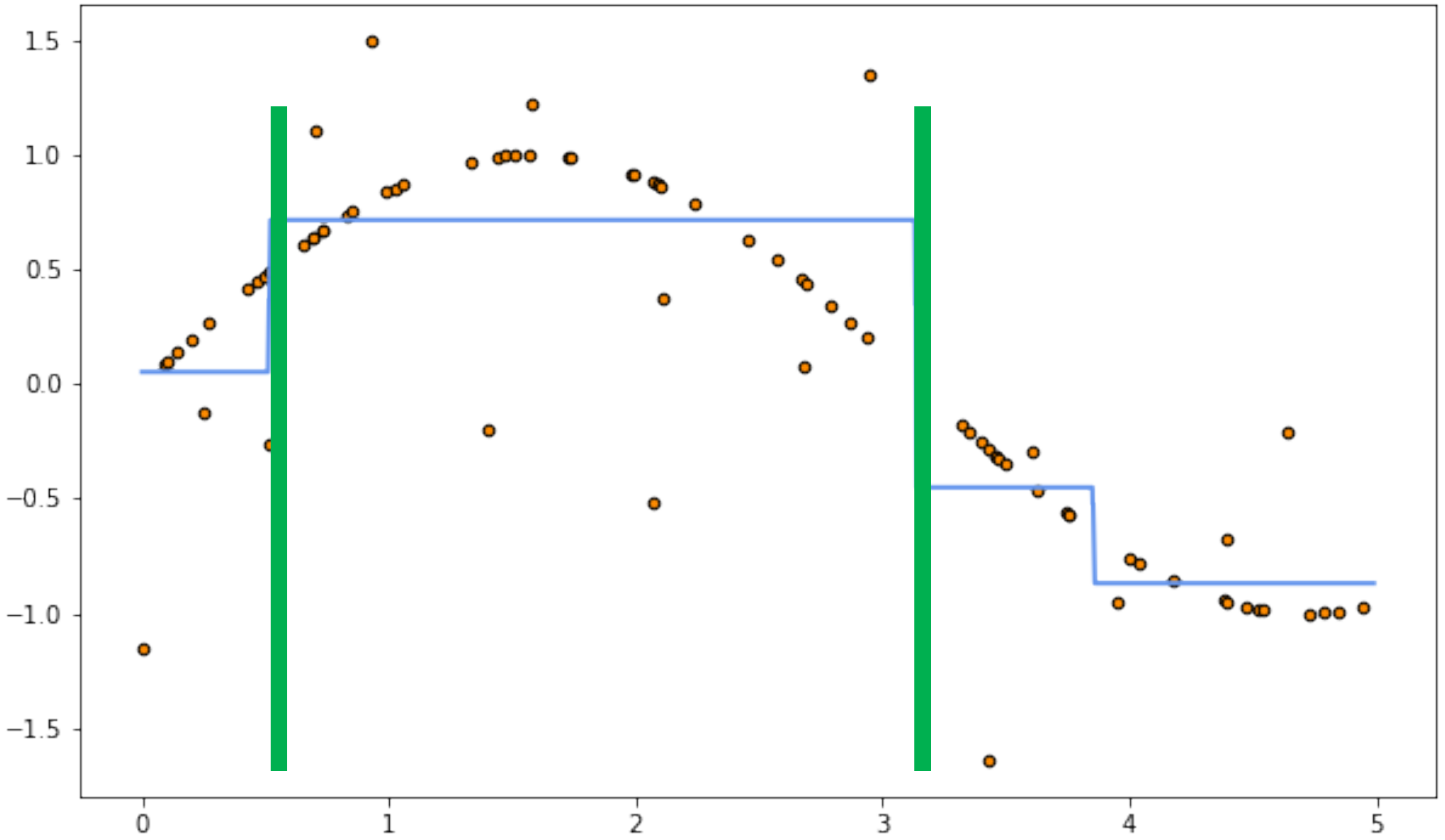


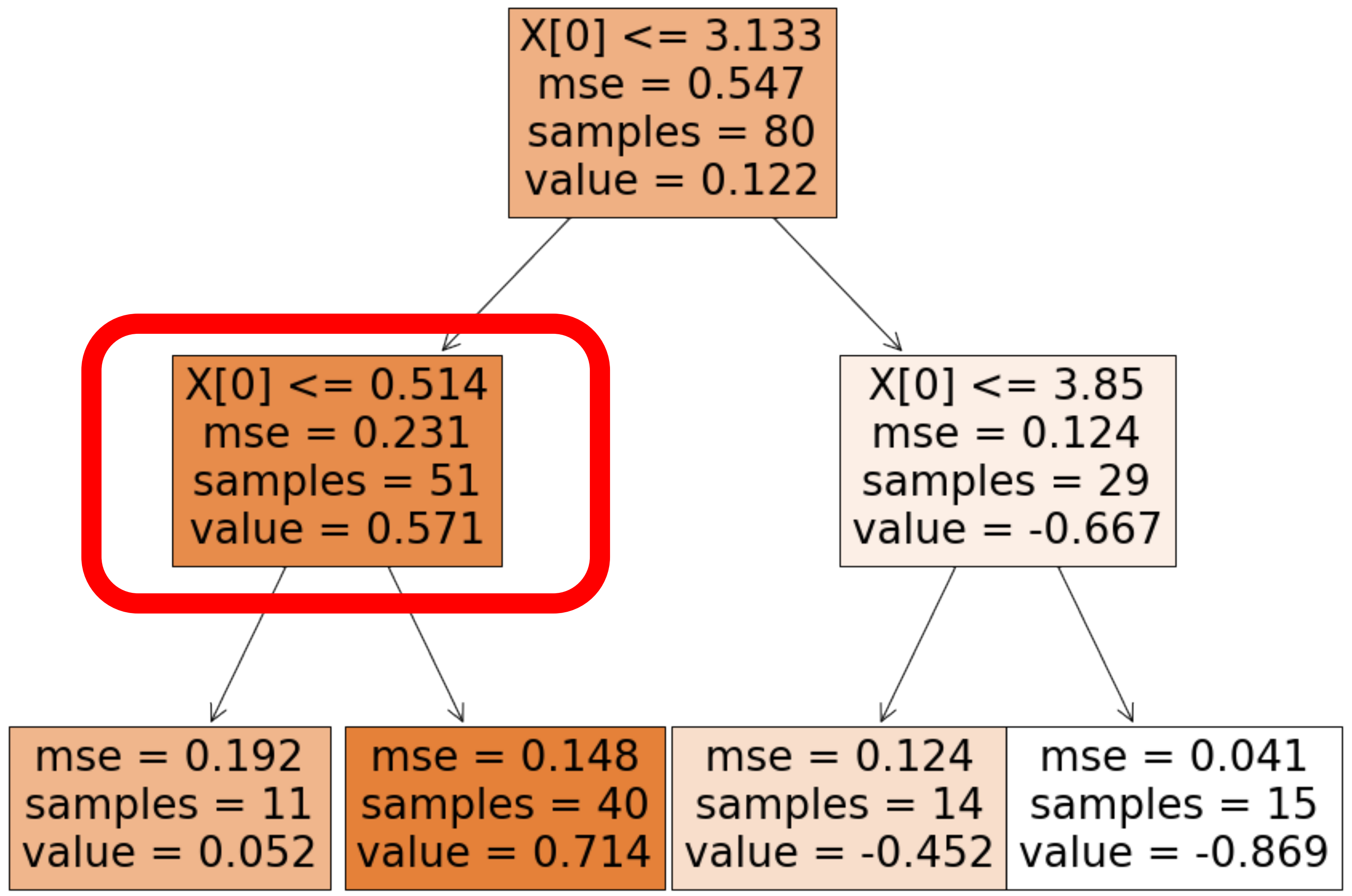


Decision Trees for Regression Task

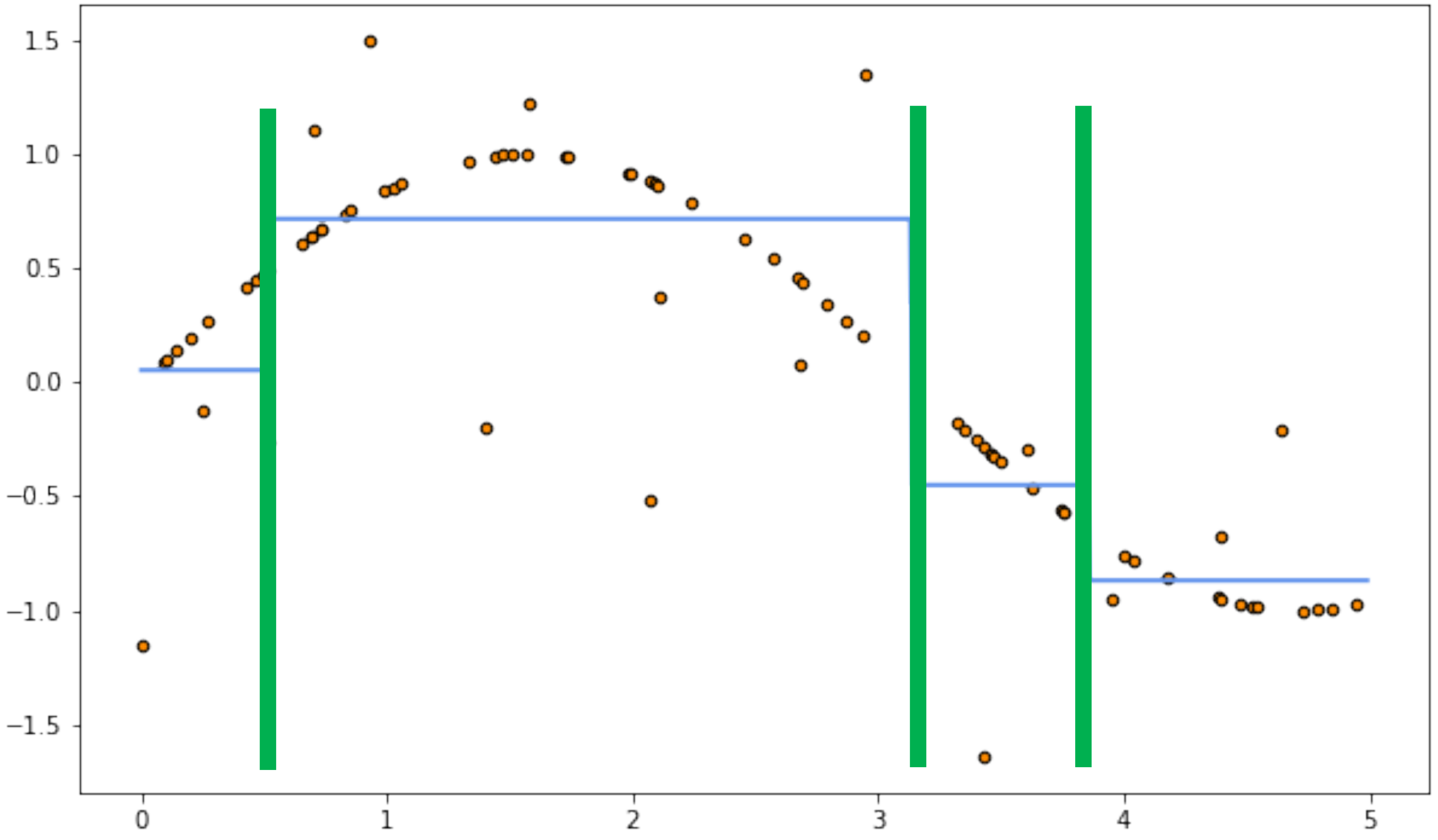


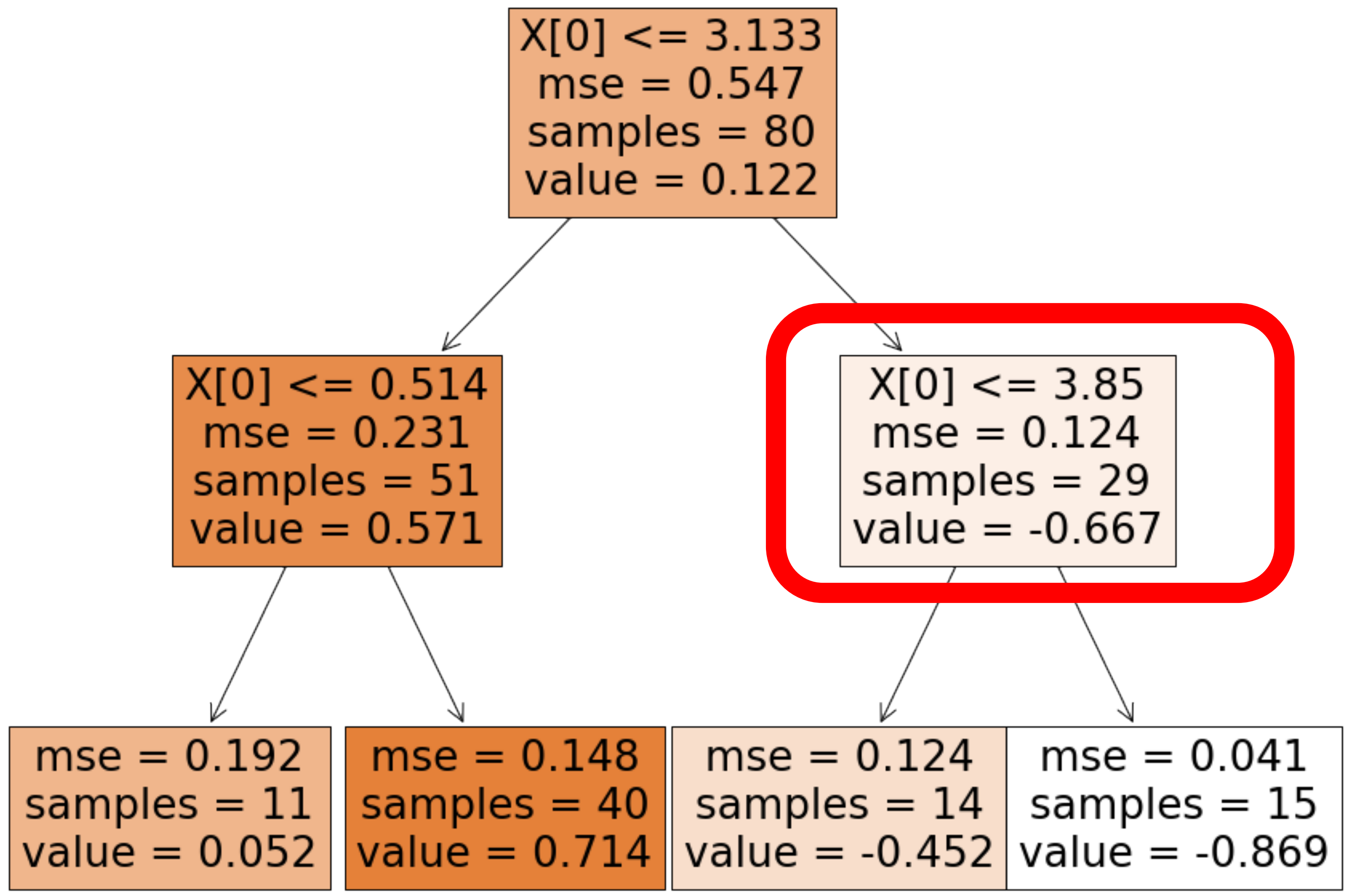
Decision Trees for Regression Task



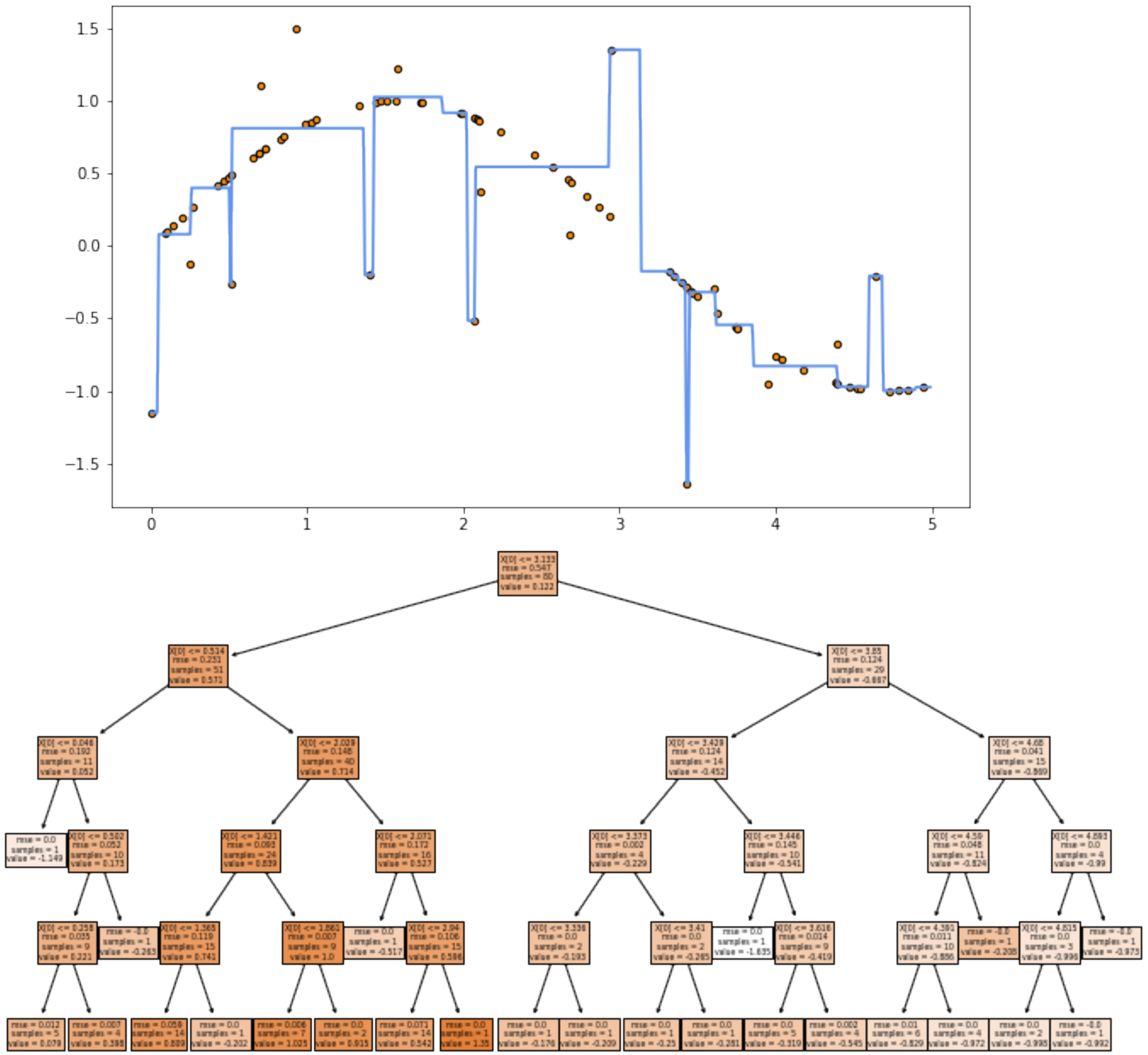


Decision Trees for Regression Task





Decision Trees for Regression and Overfitting

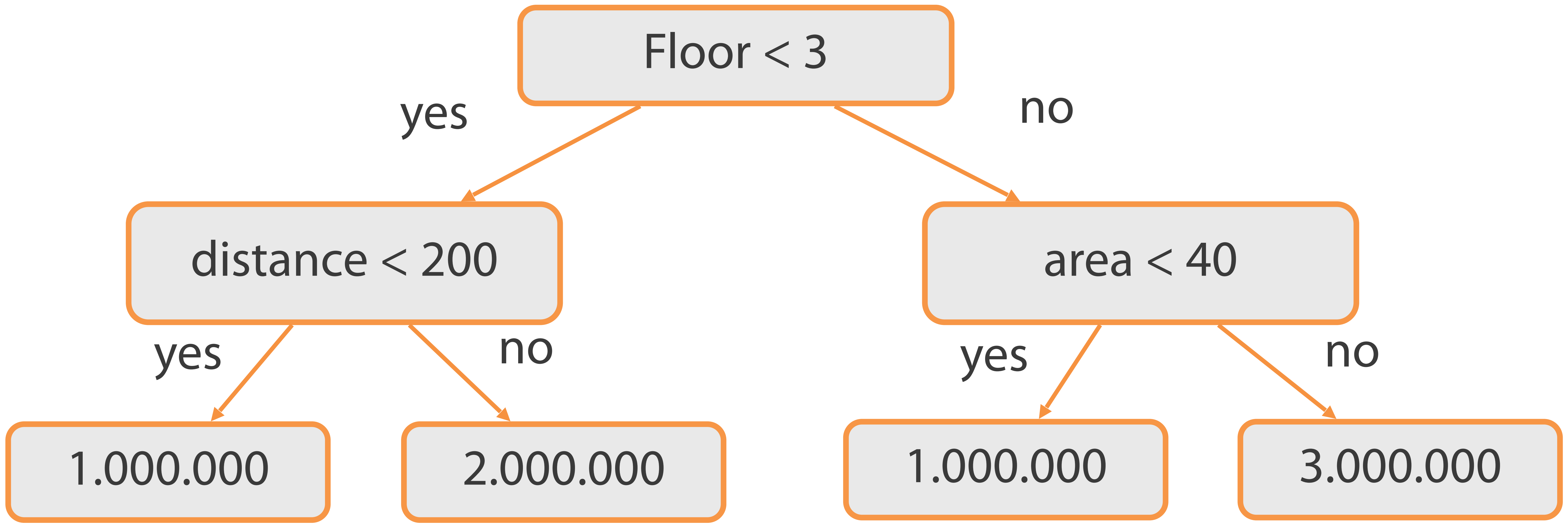


Summary

- Decision trees — combination of simple logic rules
- Decision Trees split feature space into several areas with constant prediction in each of them
- It is quite easy to overfit, when you use decision trees

Structure of Decision Trees

Decision Trees



- **Internal Nodes**: splitting criterion
- **Leaves**: predictions $c \in \mathbb{Y}$

Splitting Criterion

- Step function: $\left[x_j < t\right]$ — not the only option
- Use a linear model: $[\langle w, \ x \rangle < t]$
- A specific metric: $\left[\rho\left(x, \ x_0\right) < t\right]$
- …

- But we can build arbitrary complex models even with the most simple predicates

Predictions in Leaves: Regression

- We will use constant predictions $c_v \in \mathbb{Y}$
- Average value:

$$c_v = \frac{1}{|R_v|} \sum_{(x_i, y_i) \in R_v} y_i$$

Predictions in Leaves: Classification

- We will use constant predictions $c_v \in \mathbb{Y}$
- The most common class:

$$c_v = \underset{k \in \mathbb{Y}}{\operatorname{argmax}} \sum_{(x_i, y_i) \in R_v} [y_i = k]$$

Predictions in Leaves: Classification

- We will use constant predictions $c_v \in \mathbb{Y}$
- The most common class:

$$c_v = \underset{k \in \mathbb{Y}}{\operatorname{argmax}} \sum_{(x_i, y_i) \in R_v} [y_i = k]$$

- Class probabilities

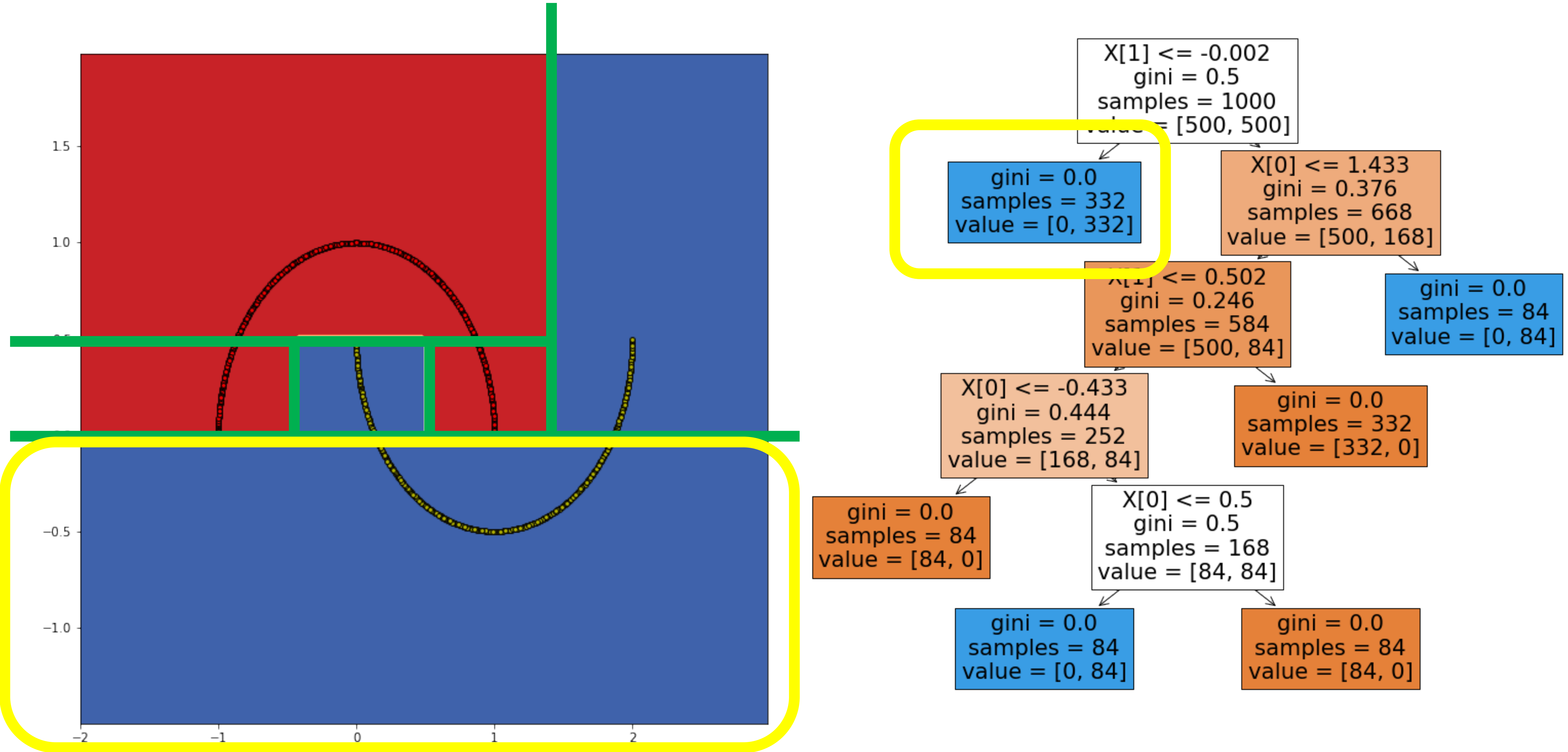
$$c_{vk} = \frac{1}{|R_v|} \sum_{(x_i, y_i) \in R_v} [y_i = k]$$

Predictions in Leaves

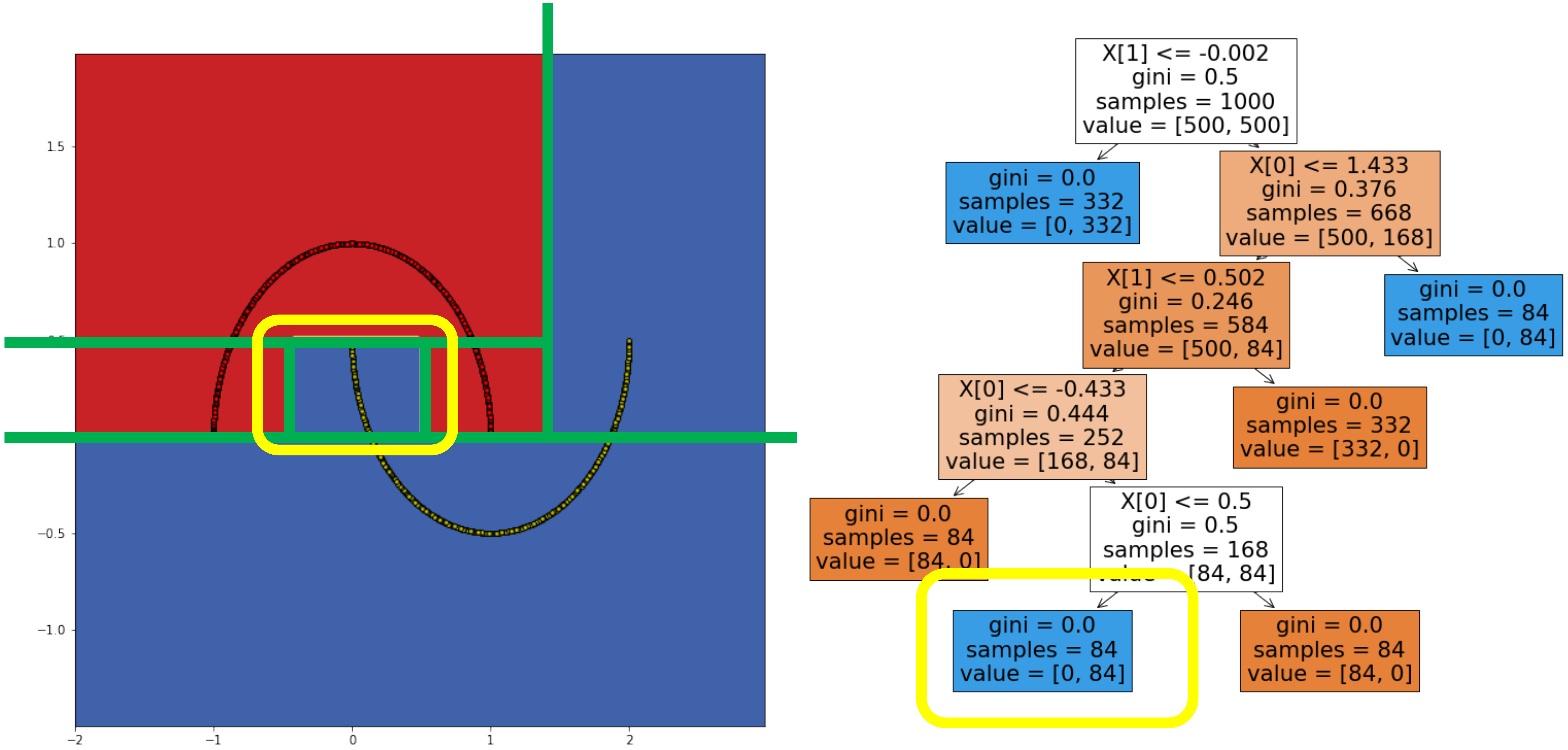
- We could use more complex prediction functions in leaves
- E.g. linear regression:

$$c_v(x) = \langle w_v, \; x \rangle$$

Decision Tree



Decision Tree



Decision Tree: Interpretation

- Tree splits feature space on disjoint sub-spaces $R_1, \ldots, R_J$
- Each sub-space R_j corresponds to the leaf
- At each sub-space R_j prediction c_j is constant

$$a(x) = \sum_{j=1}^{J} c_j \left[x \in R_j\right]$$

Decision Tree: Interpretation

$$a(x) = \sum_{j=1}^{J} c_j \left[x \in R_j \right]$$

- Decision tree constructs new powerful features
- The predictions then are linear combination of new features

Summary

- One could use different approaches in splitting and making predictions in leaves. Usually the simplest one is good enough.

- One could think about decision tree as linear model over new features

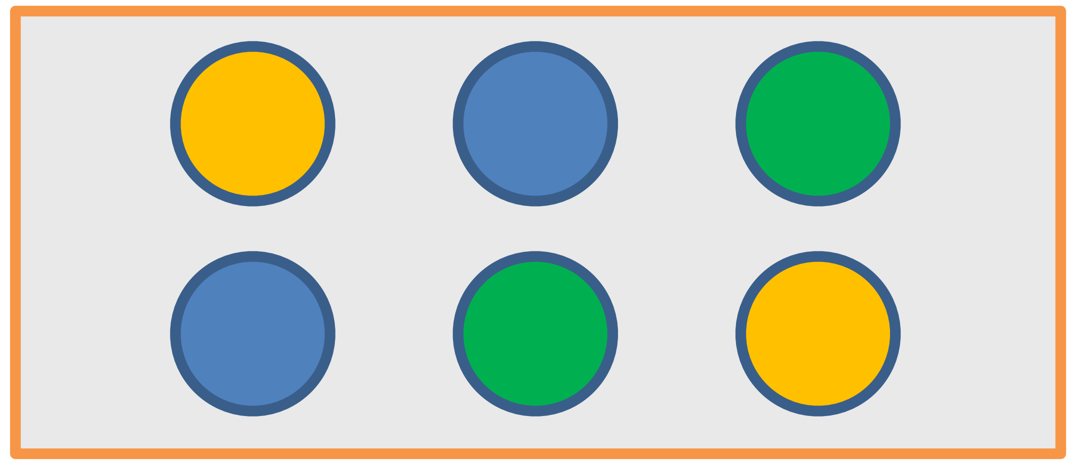
How to Train Decision Trees?

Greedy Tree Construction

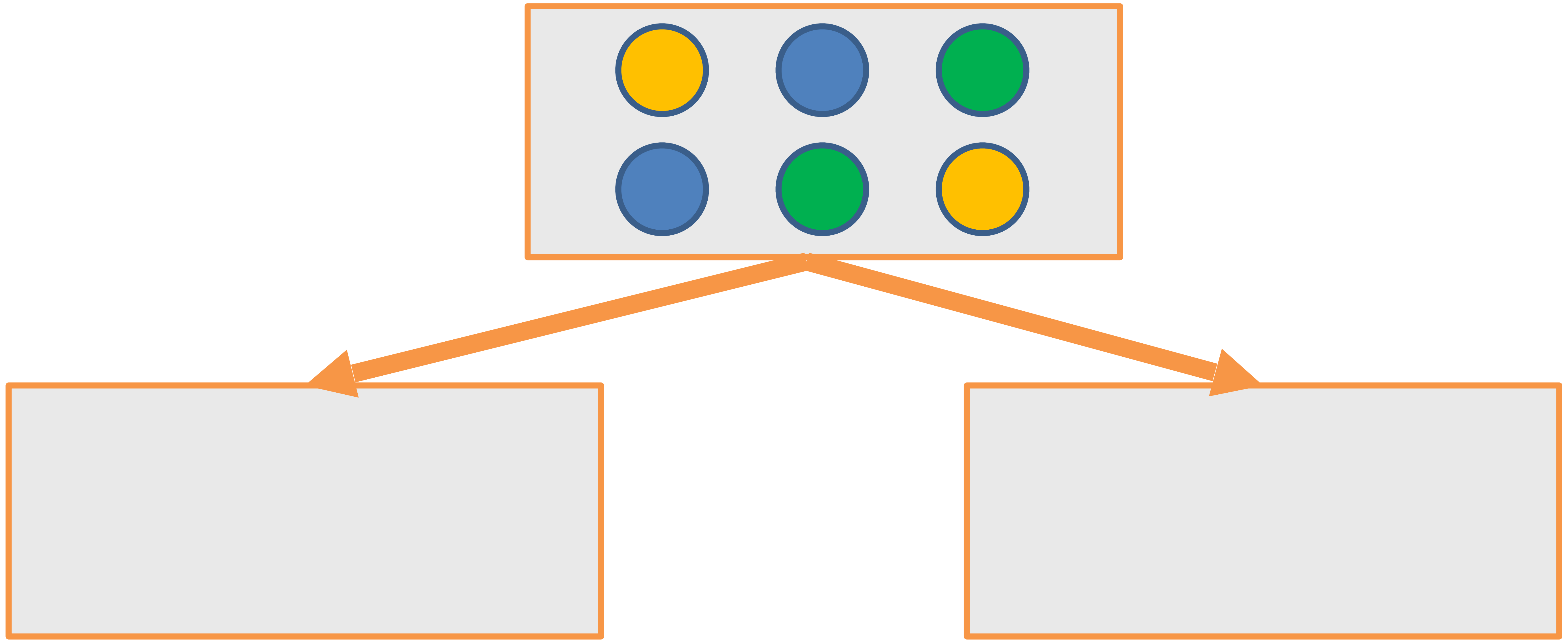
- Let’s develop the algorithm on the example

- We start from the classification task

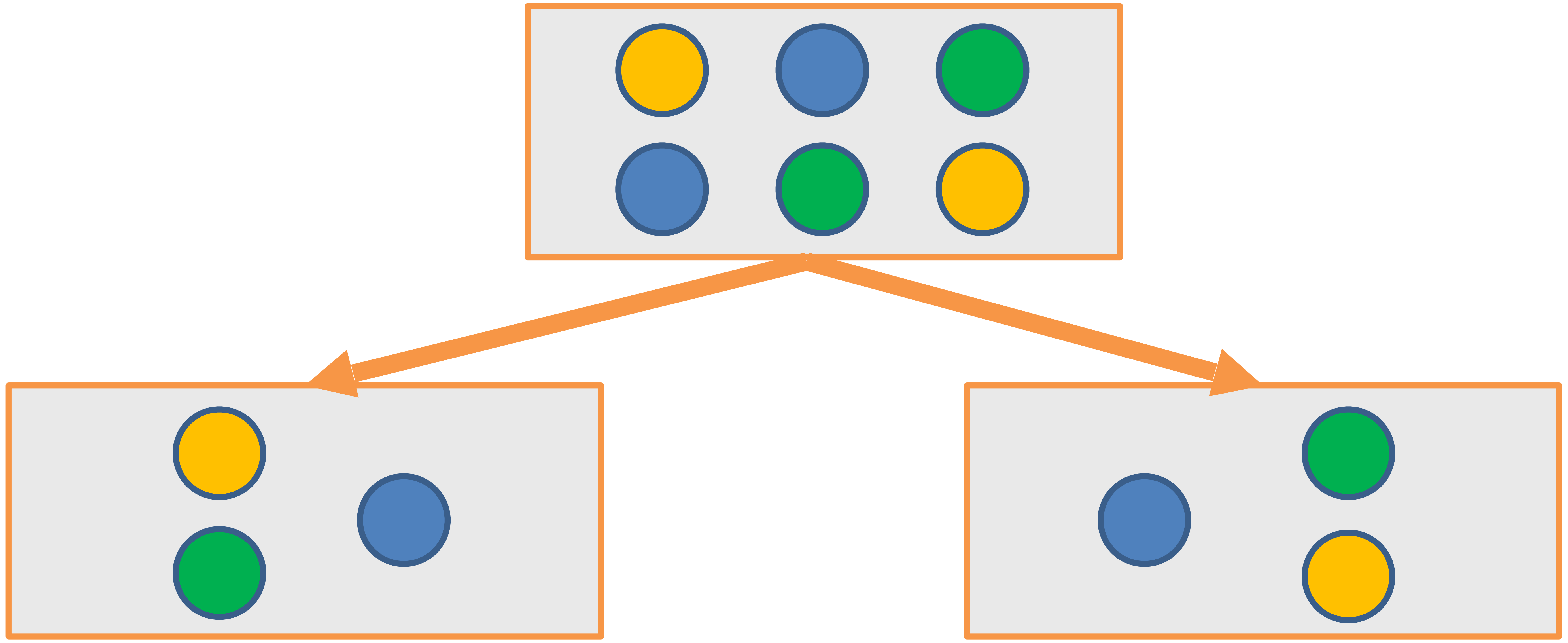
Greedy Tree Construction



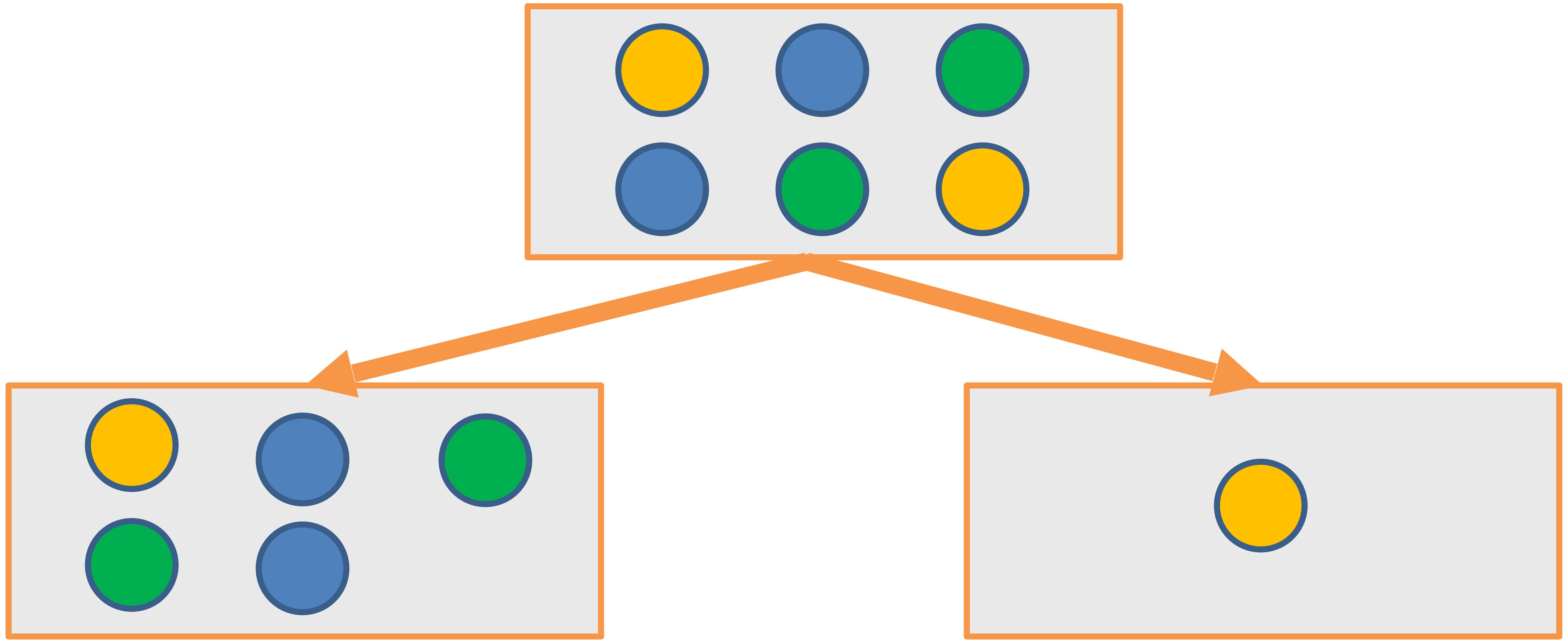
Greedy Tree Construction



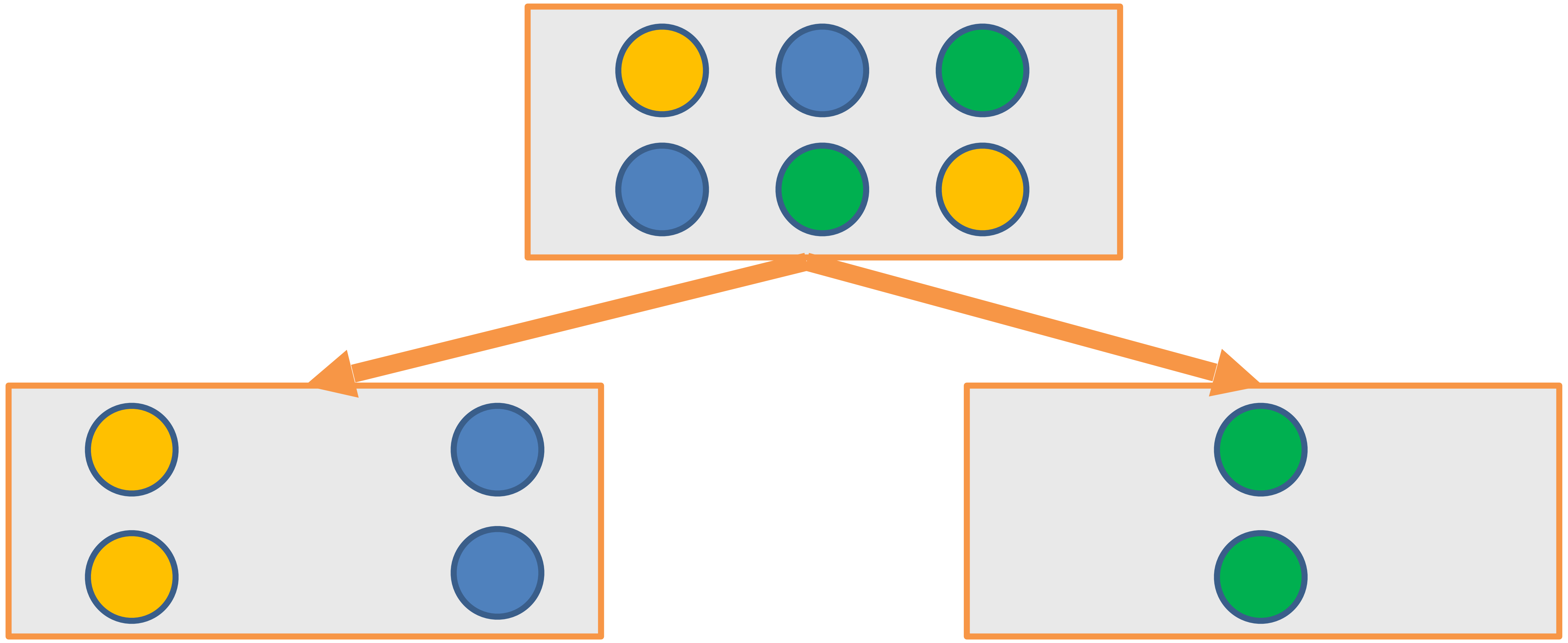
Greedy Tree Construction



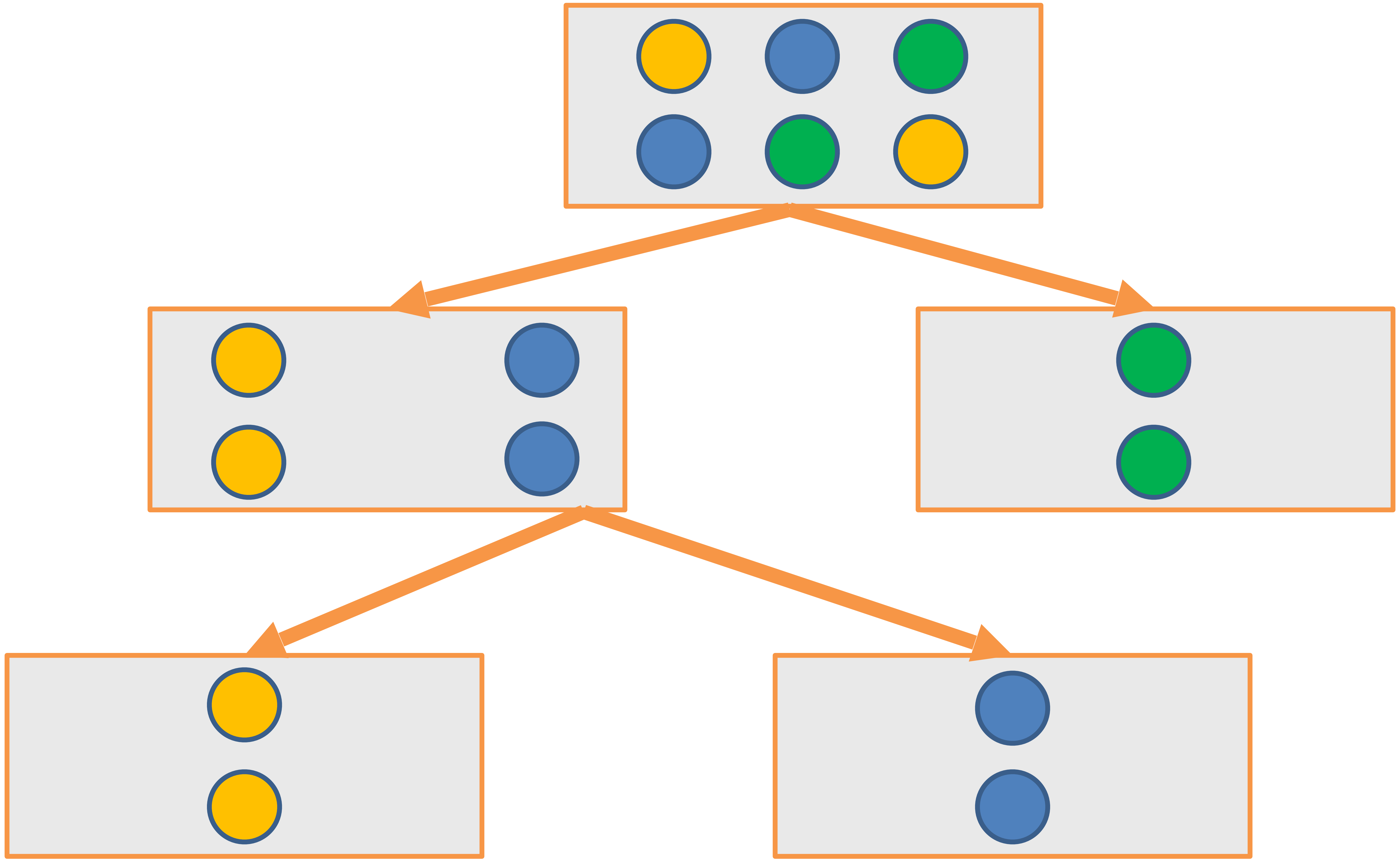
Greedy Tree Construction



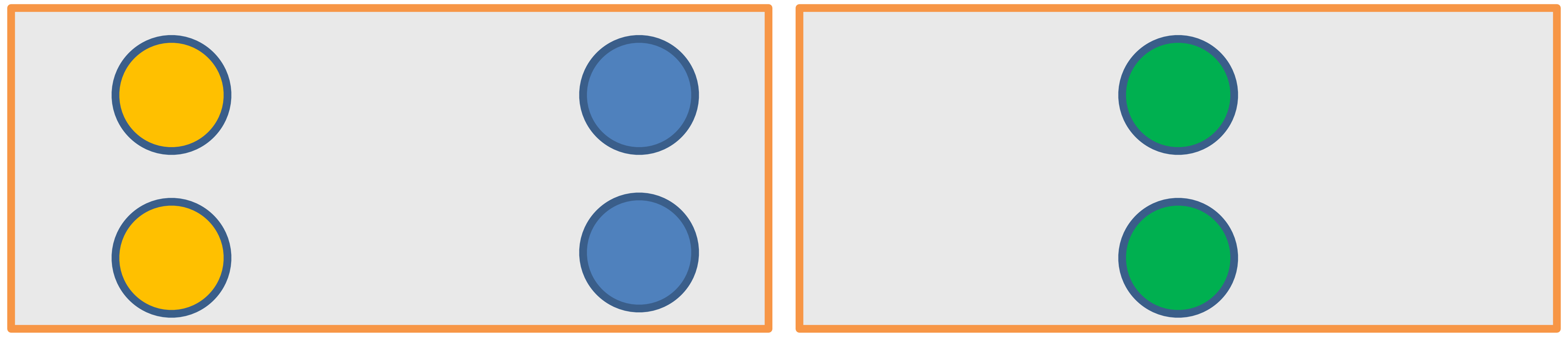
Greedy Tree Construction



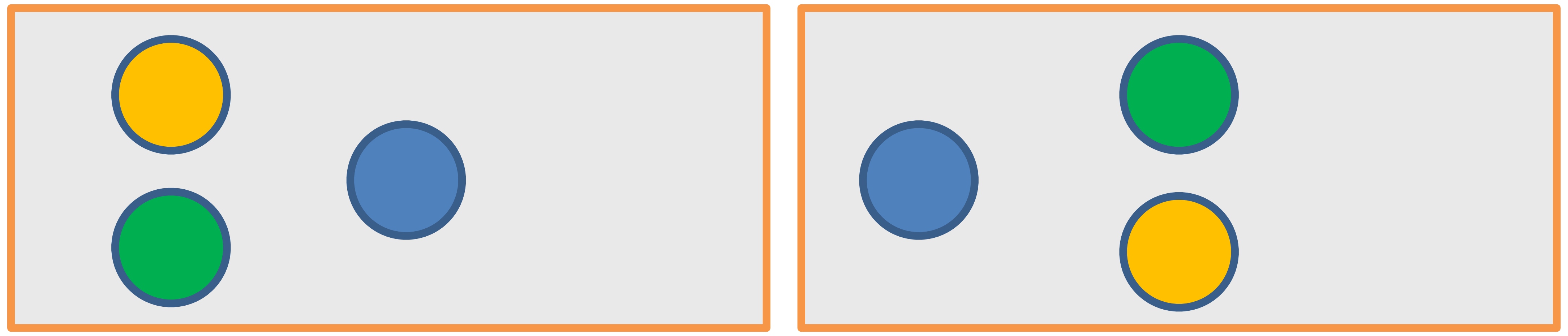
Greedy Tree Construction



How to Choose Between Two Splits ?

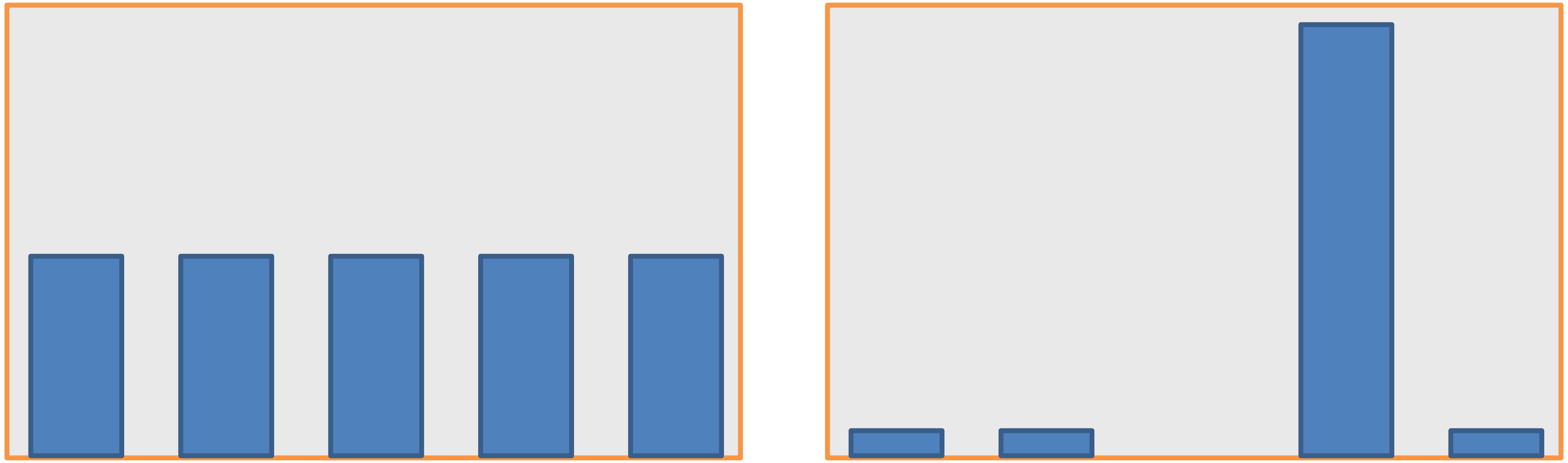


or



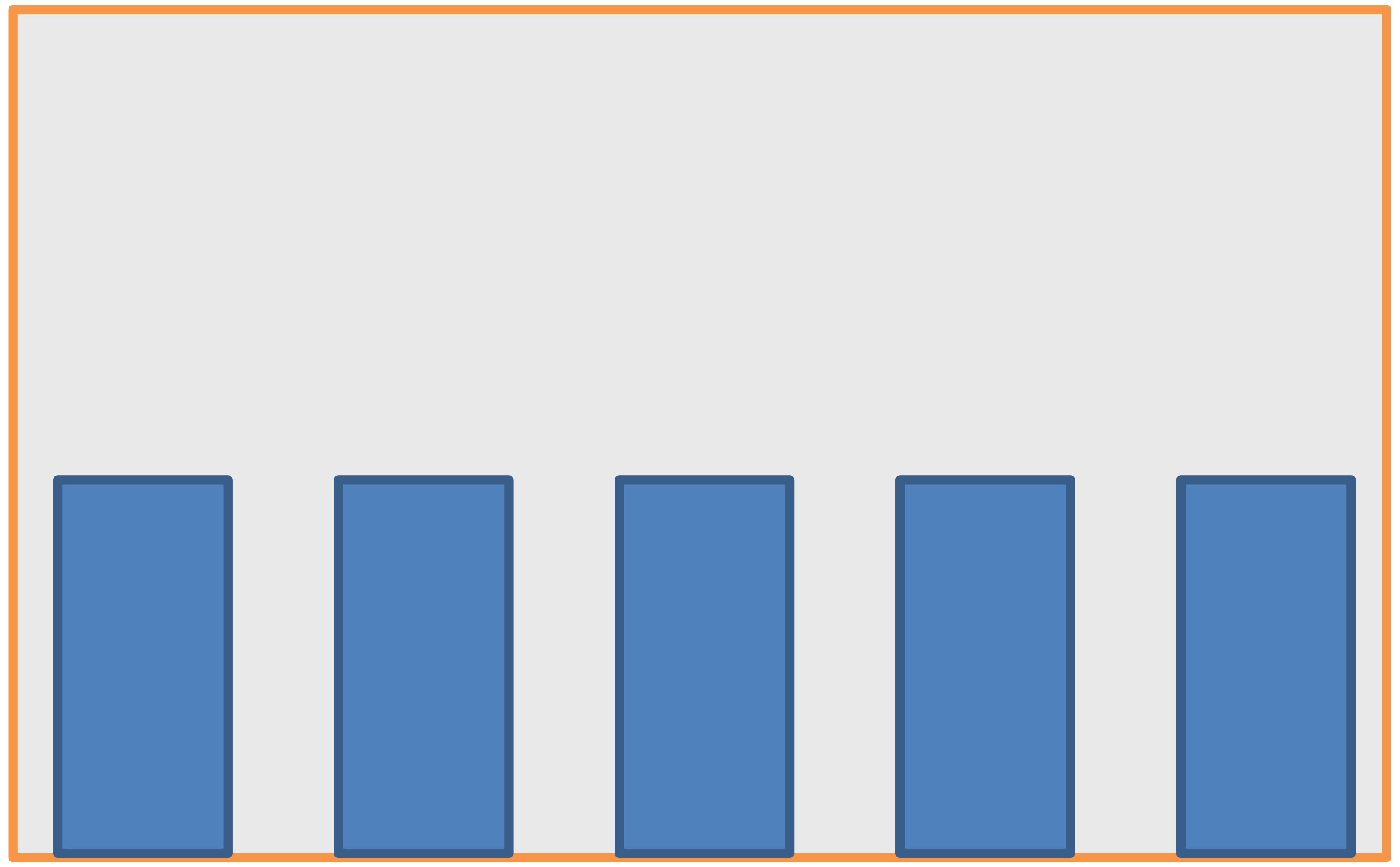
Entropy

We consider entropy as a way to measure the uncertainty of an experiment's outcome

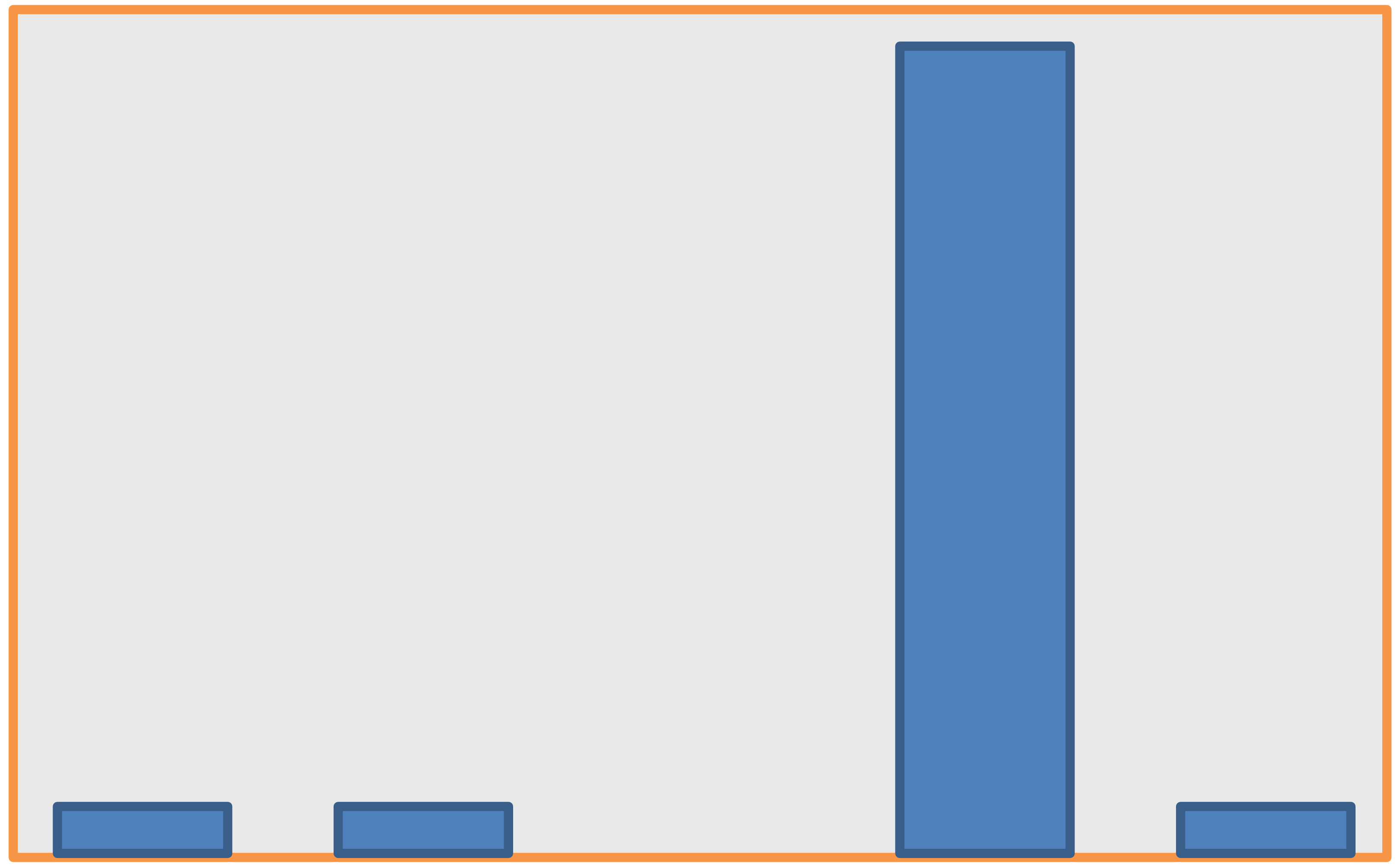


Entropy

We consider entropy as a way to measure the uncertainty of experiment's outcome



High entropy



Low entropy

Entropy

- Assume we are given discrete distribution with n possible outcomes
- Probability of outcomes: $p_1, p_2, \ldots, p_n$
- Entropy of distribution:

$$H(p_1, \ldots, p_n) = -\sum_{i=1}^{n} p_i \log_2 p_i$$

Entropy

$$H\left(p_1, \ldots, p_n\right) = -\sum_{i=1}^{n} p_i \log_2 p_i$$

- $p = (0.2,\ 0.2,\ 0.2,\ 0.2,\ 0.2)$
  - $H = 2.3219$

Entropy

$$H(p_1, \ldots, p_n) = -\sum_{i=1}^{n} p_i \log_2 p_i$$

- $p = (0.2,\ 0.2,\ 0.2,\ 0.2,\ 0.2)$
  - $H = 2.3219$

- $p = (0.9,\ 0.05,\ 0.05,\ 0,\ 0)$
  - $H = 0.5689$

Entropy

$$H\left(p_1, \ldots, p_n\right) = -\sum_{i=1}^{n} p_i \log_2 p_i$$

- $p = (0.2,\ 0.2,\ 0.2,\ 0.2,\ 0.2)$
  - $H = 2.3219$
- $p = (0.9,\ 0.05,\ 0.05,\ 0,\ 0)$
  - $H = 0.5689$
- $p = (0,\ 0,\ 0,\ 1,\ 0)$
  - $H = 0$

Entropy

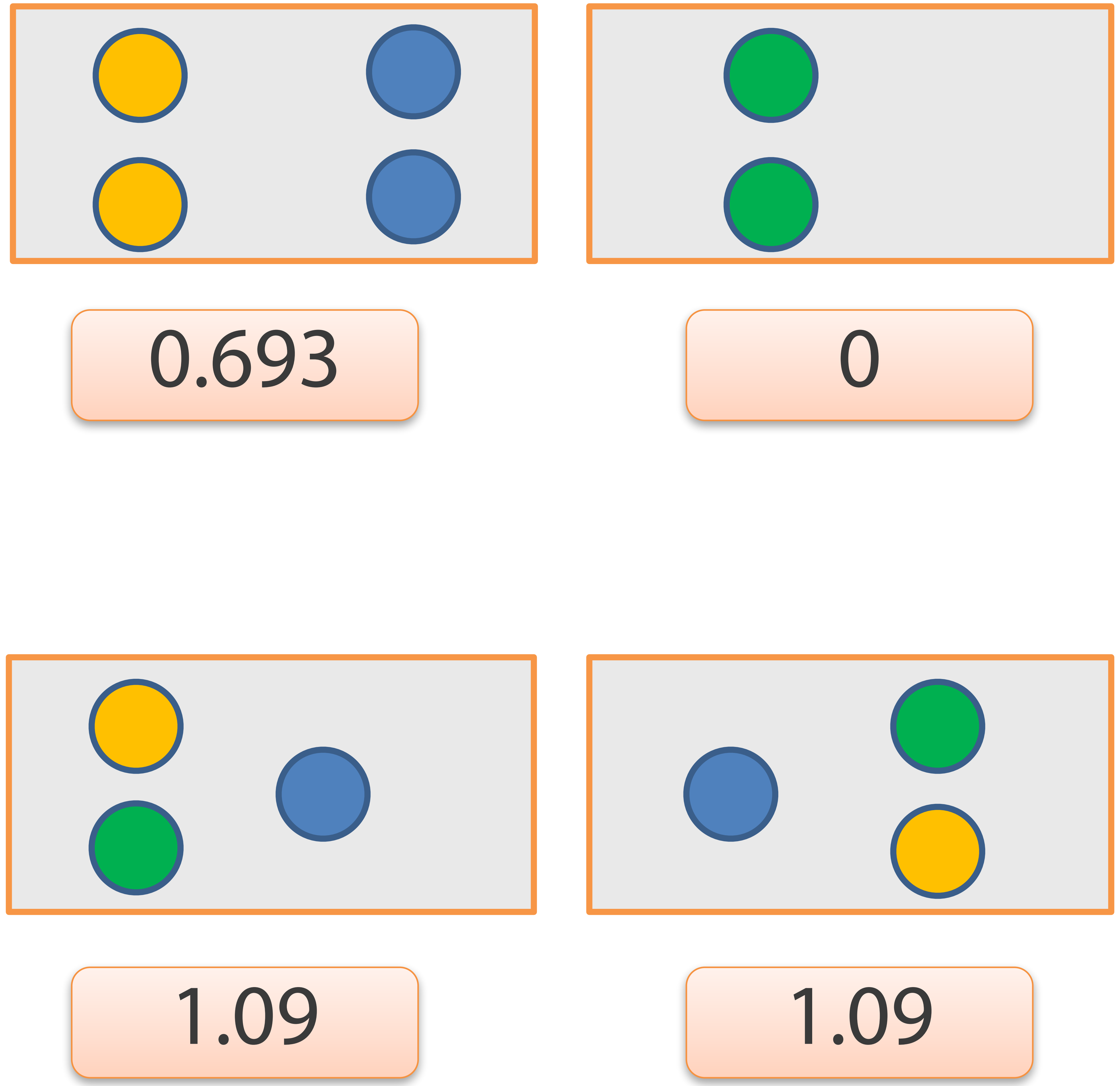
- In classification tasks the number of possible outcomes is the number of classes K

- Probability to be at the class k — fraction of objects of class k

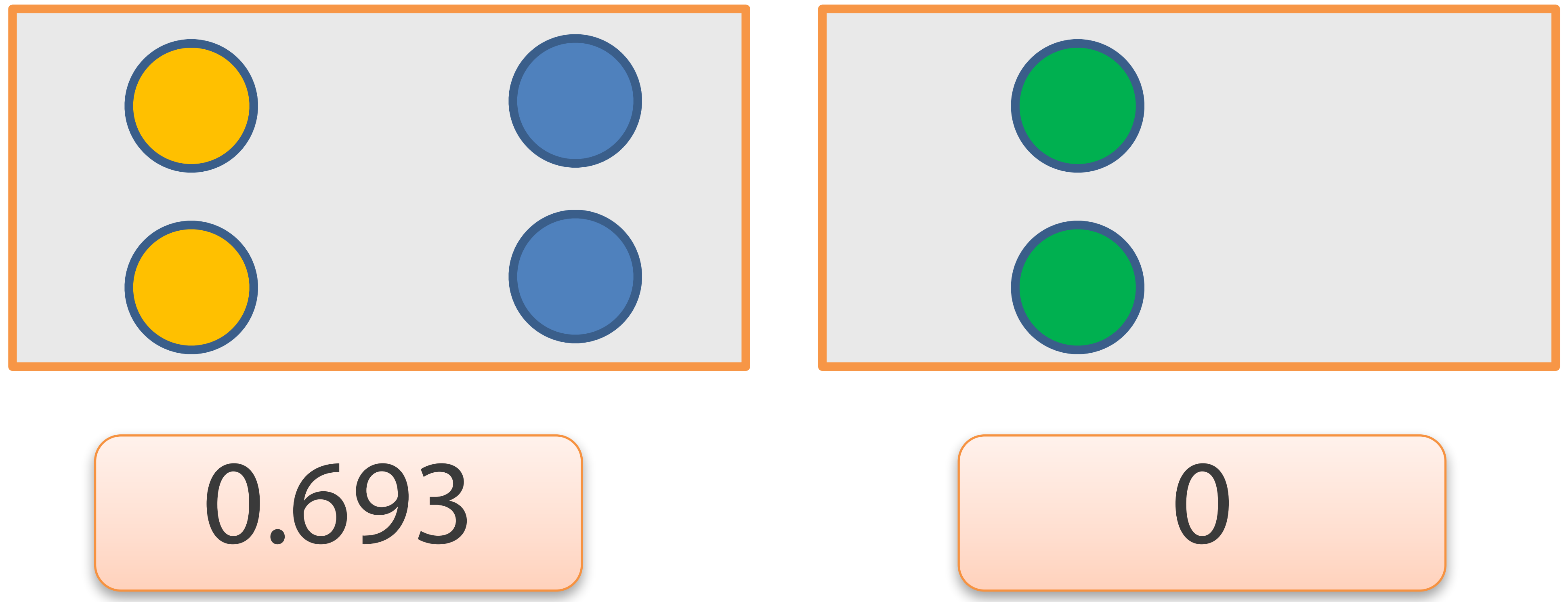
$$p_k = \frac{1}{|R|} \sum_{(x_i, y_i) \in R} [y_i = k]$$

- Zero Entropy — there are **only** objects form **one class** at the leaf
- Max. Entropy — there are **equal proportion** of objects from **each class**

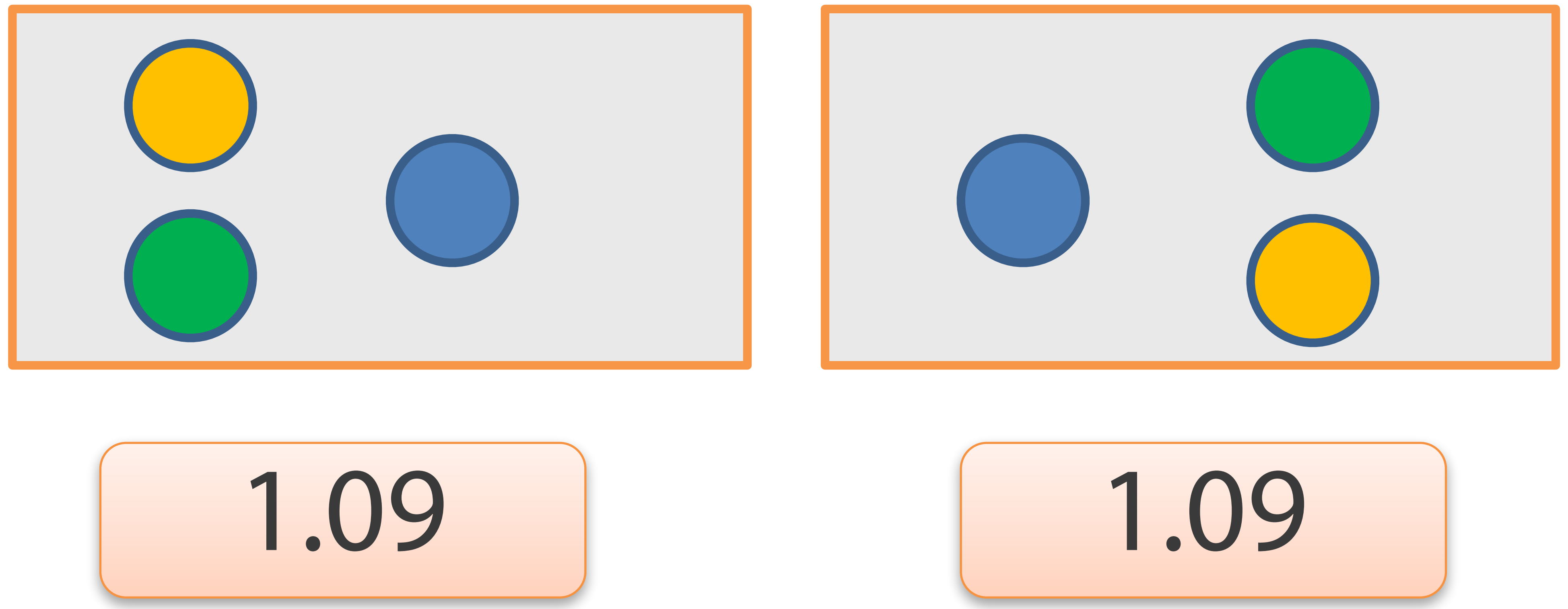
How to Select Between Two Splits?



How to Select Between Two Splits?



- $(0.5,\ 0.5,\ 0)$ and $(0,\ 0,\ 1)$
- $0.693 + 0 = 0.693$



- $(0.33,\ 0.33,\ 0.33)$ and $(0.33,\ 0.33,\ 0.33)$
- $1.09 + 1.09 = 2.18$

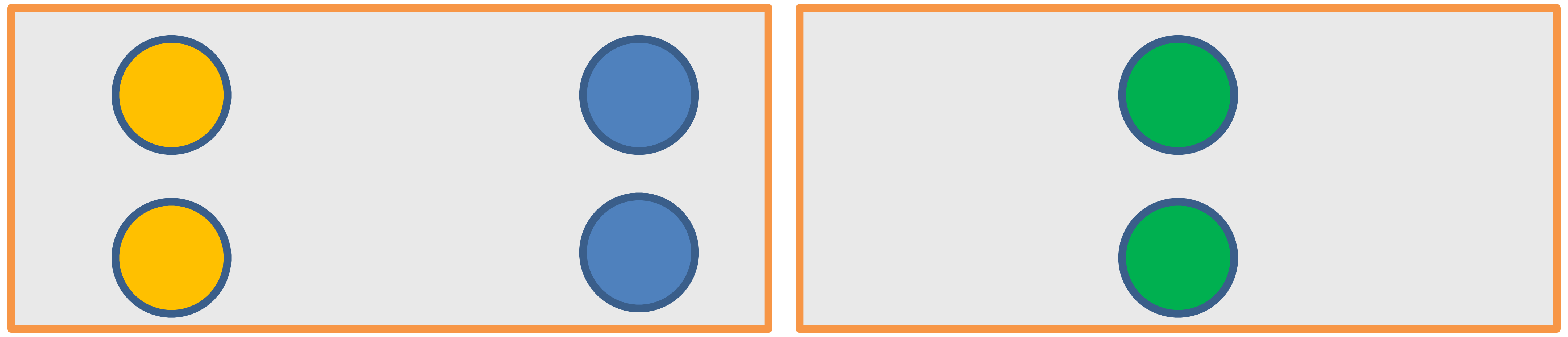
Summary

- Decision tree could be constructed in a greedy manner from the root node to the leaves

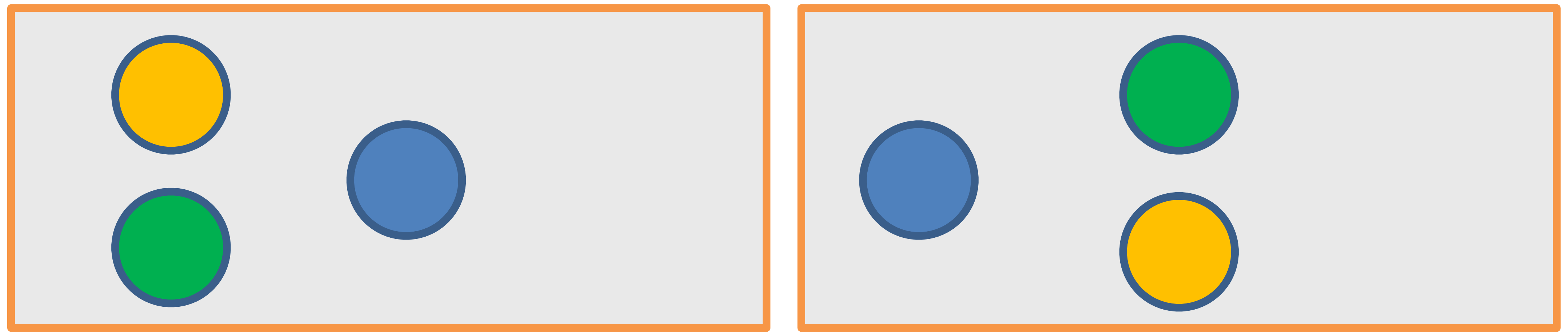
- For classification task we could choose a split, so that it minimizes class diversity at resulting groups

Measures of Impurity

How to Choose Between Two Splits ?



or



Entropy

$$H(p_1, \ldots, p_K) = -\sum_{i=1}^{K} p_i \log_2 p_i$$

- Measure of diversity at the node

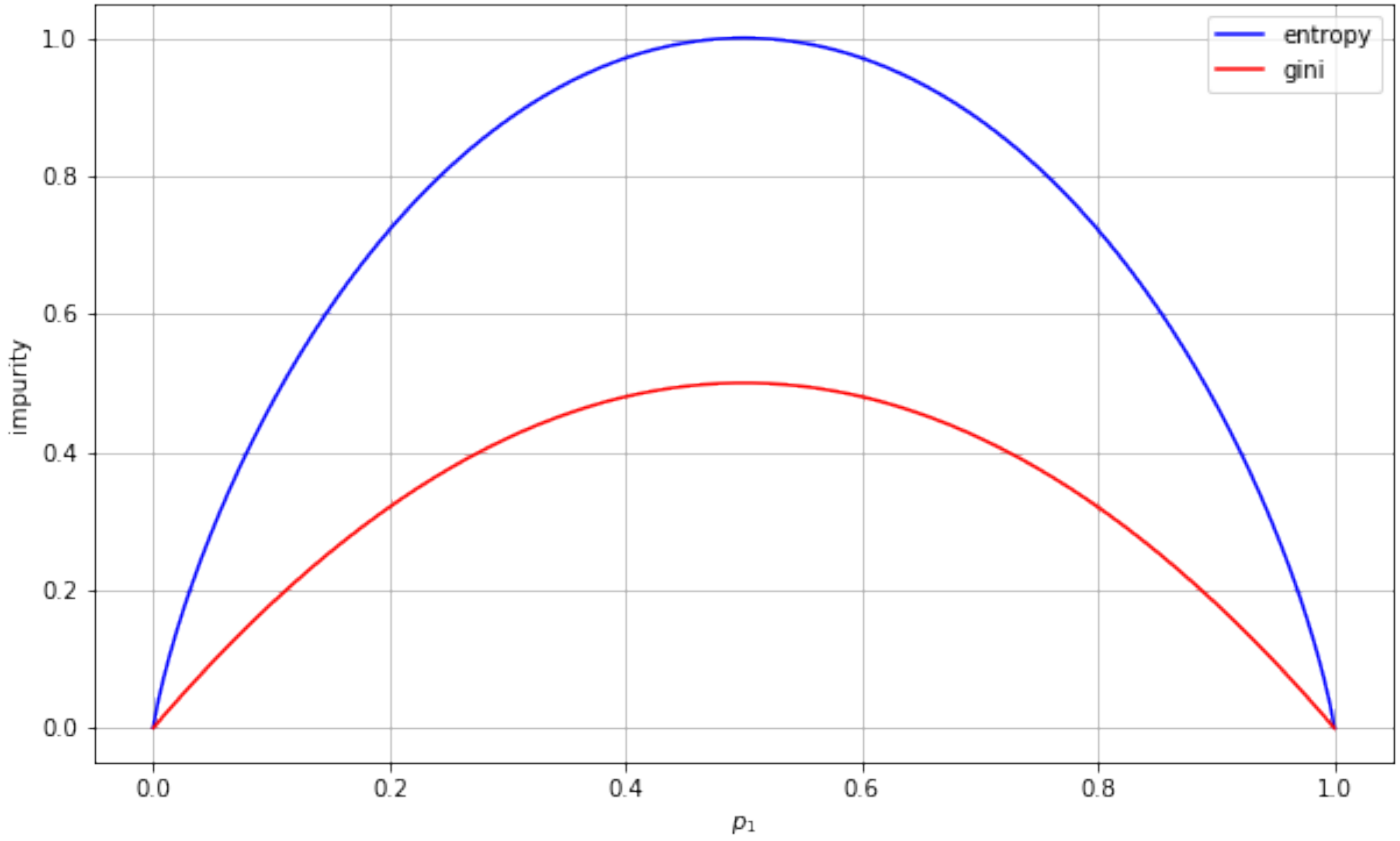
Gini Index

$$H(p_1, \ldots, p_K) = \sum_{i=1}^{K} p_i \left(1 - p_i\right)$$

- Consider a classifier, which outputs class k with probability p_k

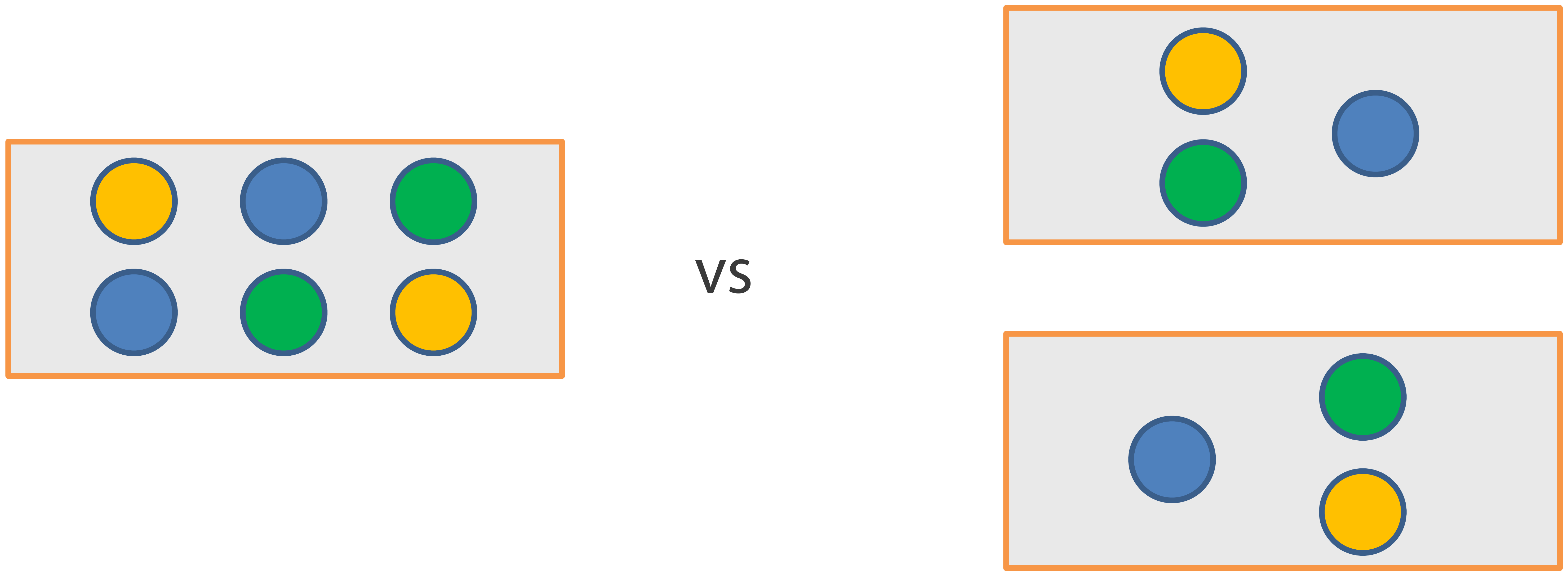
- Gini index is a probability that the object will be classified incorrectly if the class is assigned with probabilities $p_1, \ldots, p_k$

Gini Index vs Entropy



Impurity Criterions

- How to decide which split is better?
- Compare the impurity before the split (in the initial node R) and in the two nodes after the split (R_ℓ and R_r)



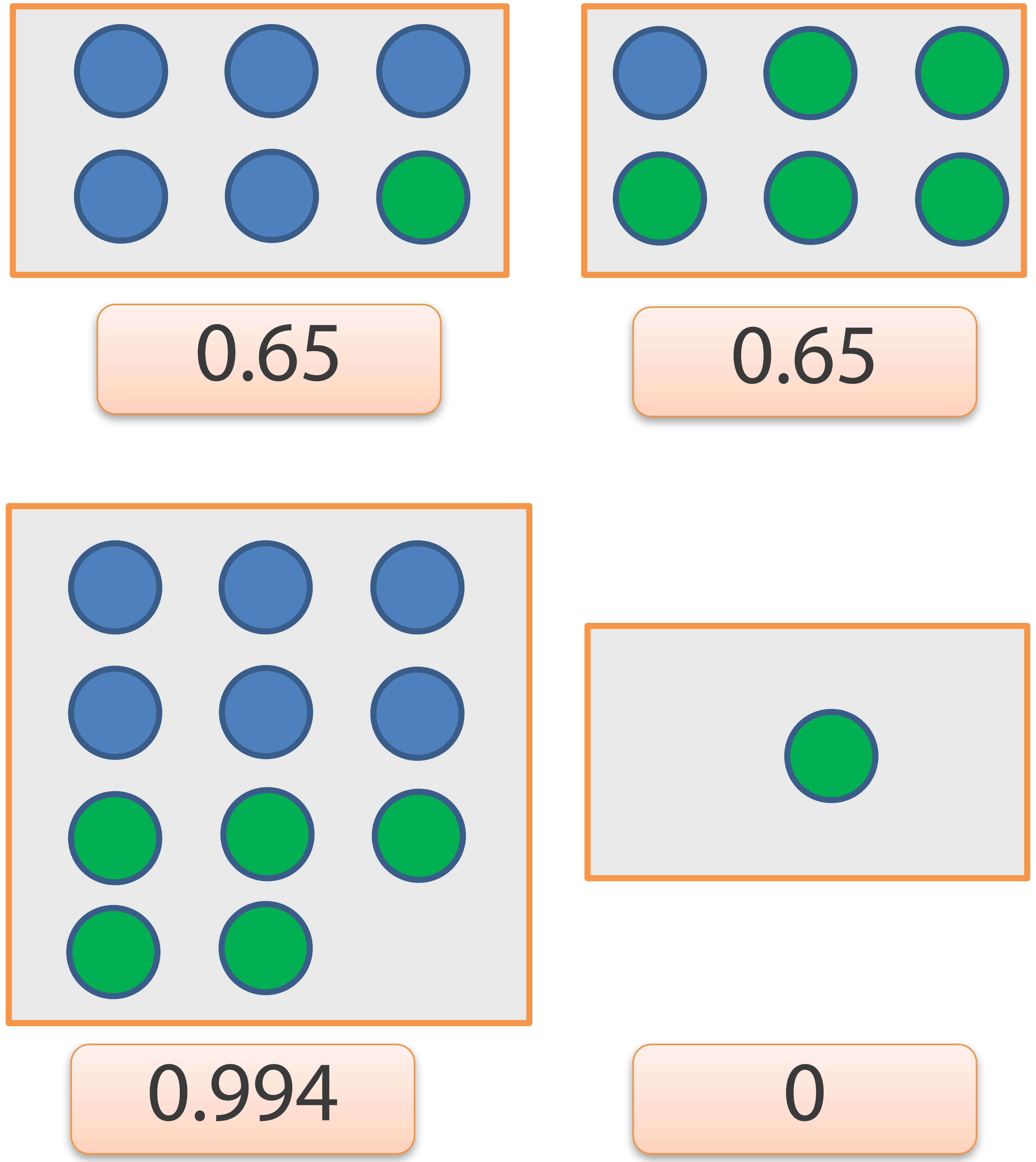
Impurity Criterions

- How to decide which split is better?
- Compare the impurity before the split (in the initial node R) and in the two nodes after the split (R_ℓ and R_r)

$$Q(R, j, t) = H(R) - \frac{|R_\ell|}{|R|} H(R_\ell) - \frac{|R_r|}{|R|} H(R_r) \to \max_{j,\ t}$$

How to Compare Two Splits?

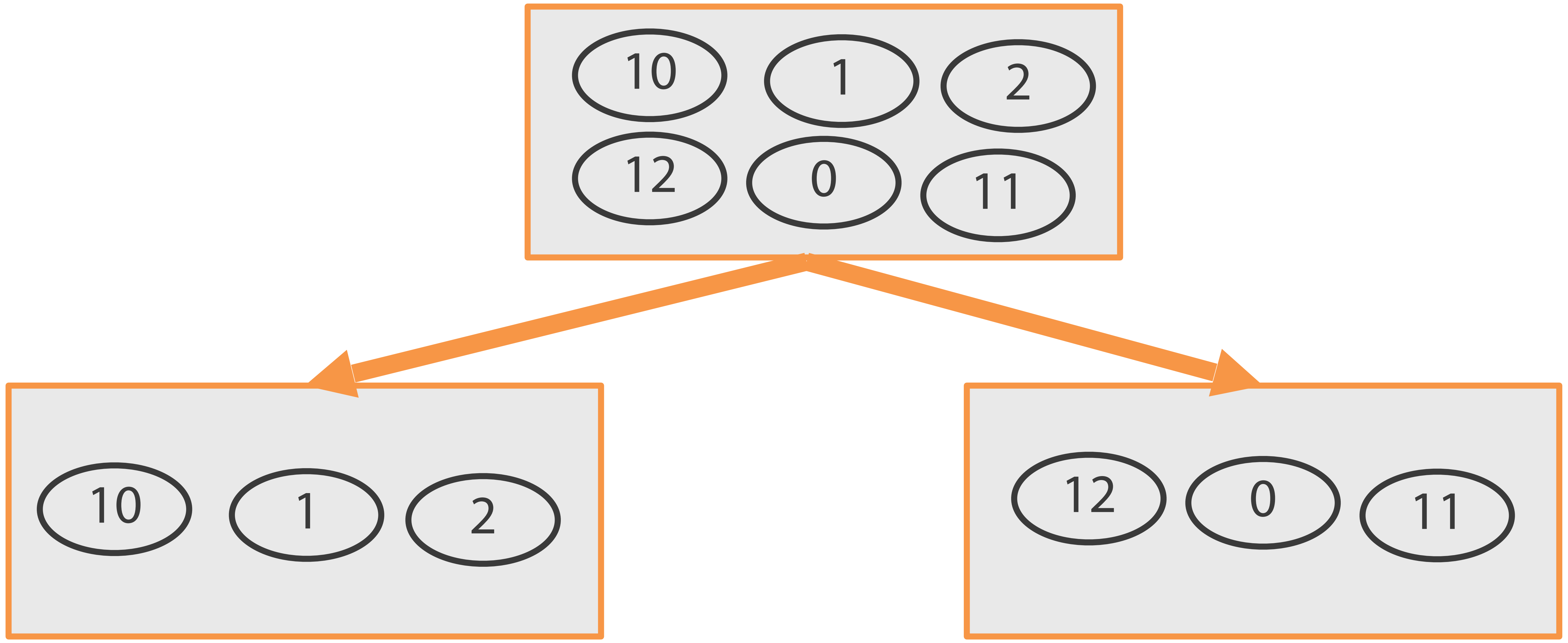
$$H(R) = 1$$



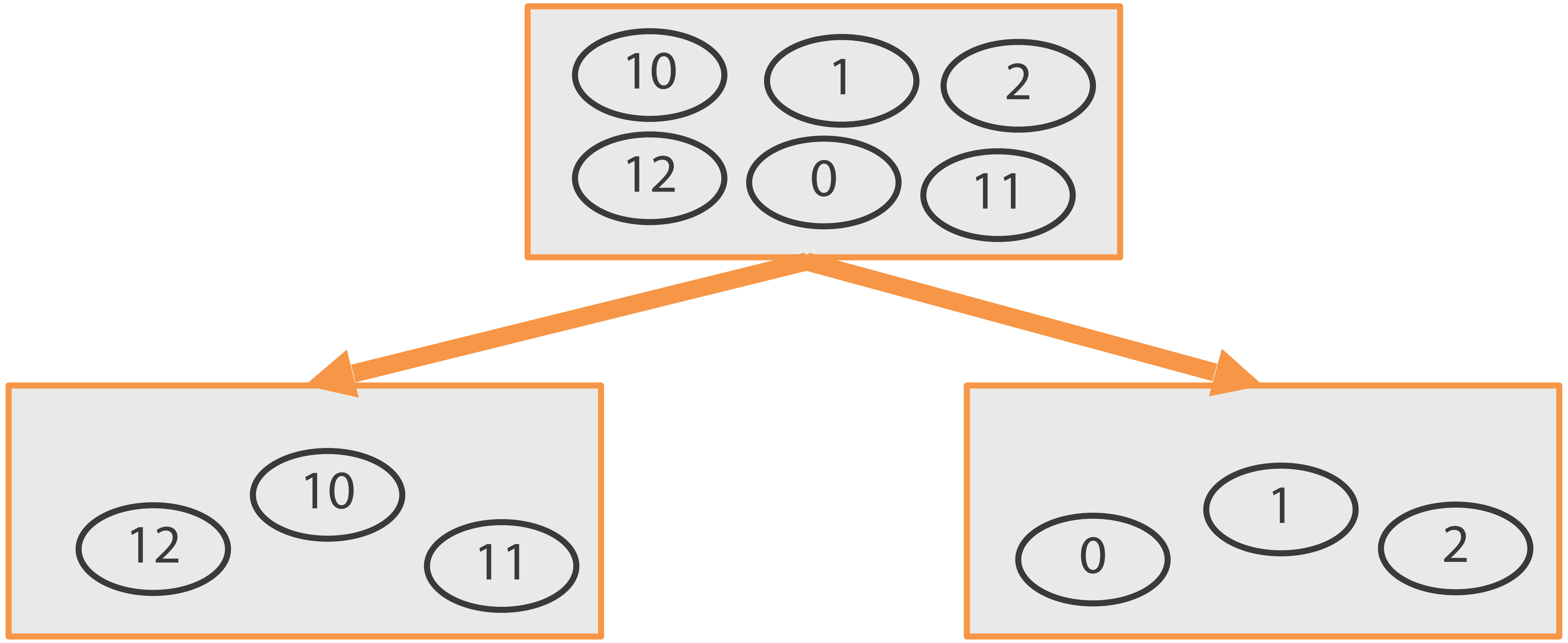
- $Q(R) = 1 - \frac{1}{2}0.65 - \frac{1}{2}0.65$
- $Q(R) = 0.35$

- $Q(R) = 1 - \frac{11}{12}0.994 - \frac{1}{12}0$
- $Q(R) = 0.088$

Greedy Construction: Regression



Greedy Construction: Regression



Regression Task

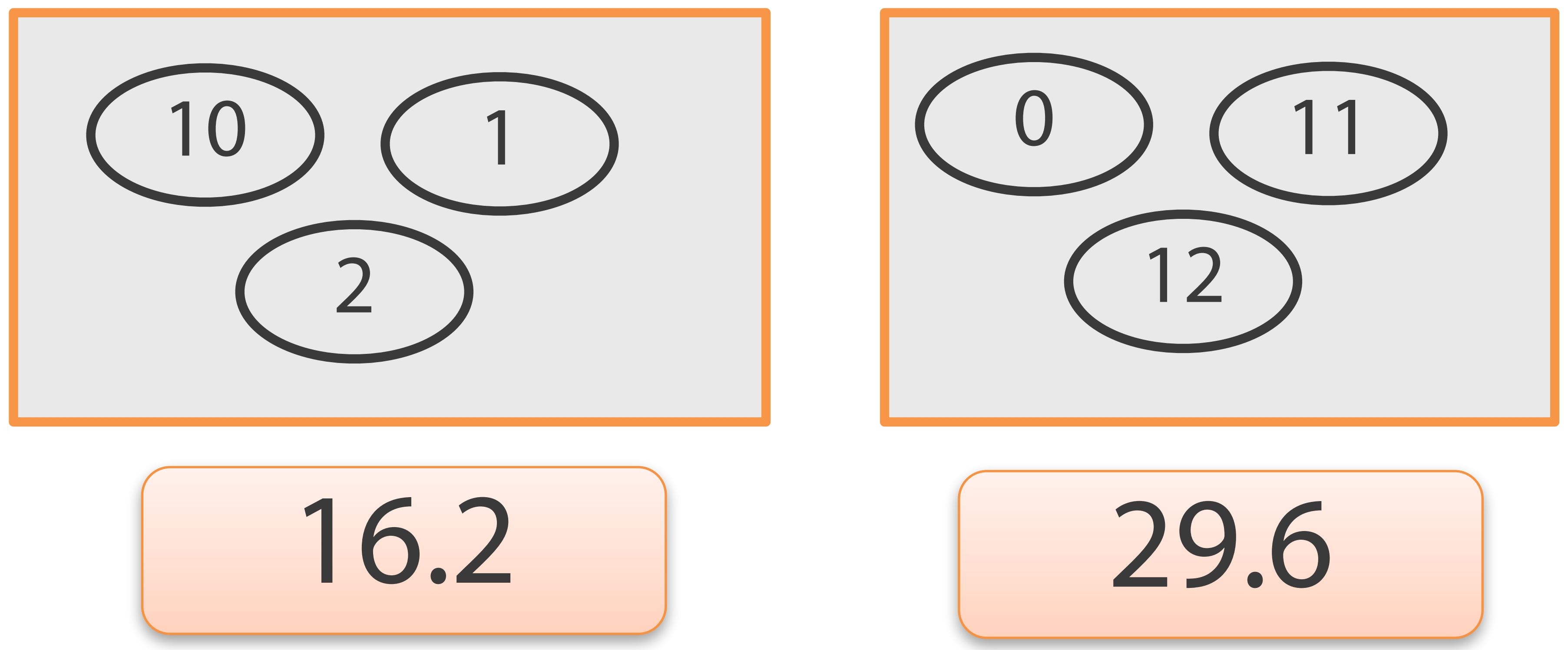
$$H(R) = \frac{1}{|R|} \sum_{(x_i,\, y_i) \in R} \left(y_i - y_R\right)^2$$

$$y_R = \frac{1}{|R|} \sum_{(x_i,\, y_i) \in R} y_i$$

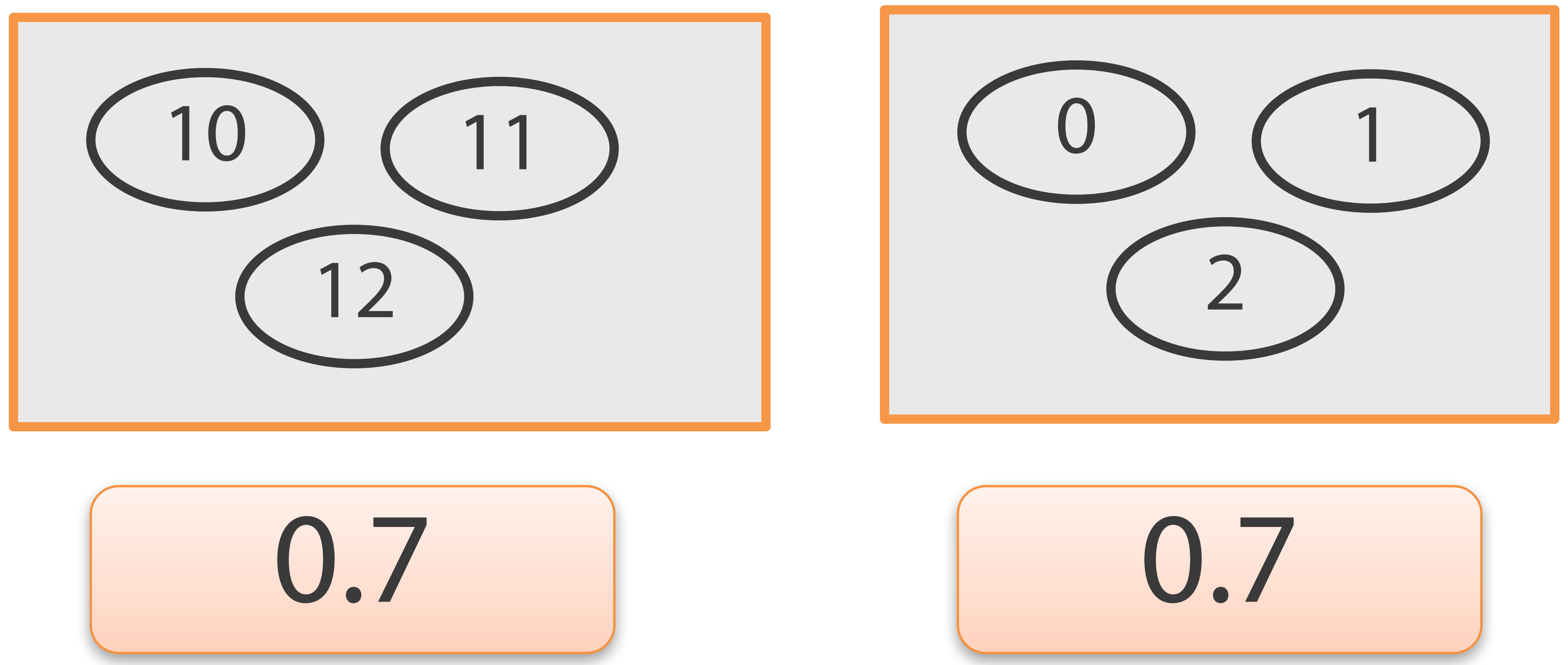
- So we can measure the variance of answers in the node

How to Compare Two Splits?

$$H(R) = 25.6$$



- $Q(R) = 25.6 - \frac{1}{2}16.2 - \frac{1}{2}29.6$
- $Q(R) = 2.7$



- $Q(R) = 25.6 - \frac{1}{2}0.7 - \frac{1}{2}0.7$
- $Q(R) = 24.9$

Summary

- We can choose the split, so that it reduces the diversity of answers in the resulting nodes

- We use impurity criterion to measure the quality of the split

- There are different criterions that might be used.
  The most popular are:
  - Entropy and Gini for classification
  - Variance for regression

Greedy Tree Construction

How to Construct a Tree?

- Optimal option:
  - Try all possible trees and select the smallest one

- That is too computationally expensive

How to Construct a Tree?

- We know how to choose the best split for a given node

- Let’s use greedy algorithm

- We start from the root node and will be splitting until some stopping criterion is satisfied

Stopping Criterions

- Restrict the maximal depth
- Restrict the number of leaves
- Fix the minimal number of objects in the node
- Set the minimal decrease in the diversity when splitting
- etc.

Greedy Algorithm

1. Put the whole dataset into the root: $R_1 = X$

2. Start the tree construction: $\text{SplitNode}(1, R_1)$

Greedy Algorithm

$\mathrm{SplitNode}\left(m,\ R_m\right)$

1. If stopping criterion is satisfied, then quit

2. Find the best split (feature and threshold):

$$j,\ t = \underset{\mathrm{j,t}}{\mathrm{argmax}} Q(R_m,\ j,\ t)$$

3. Split the objects:

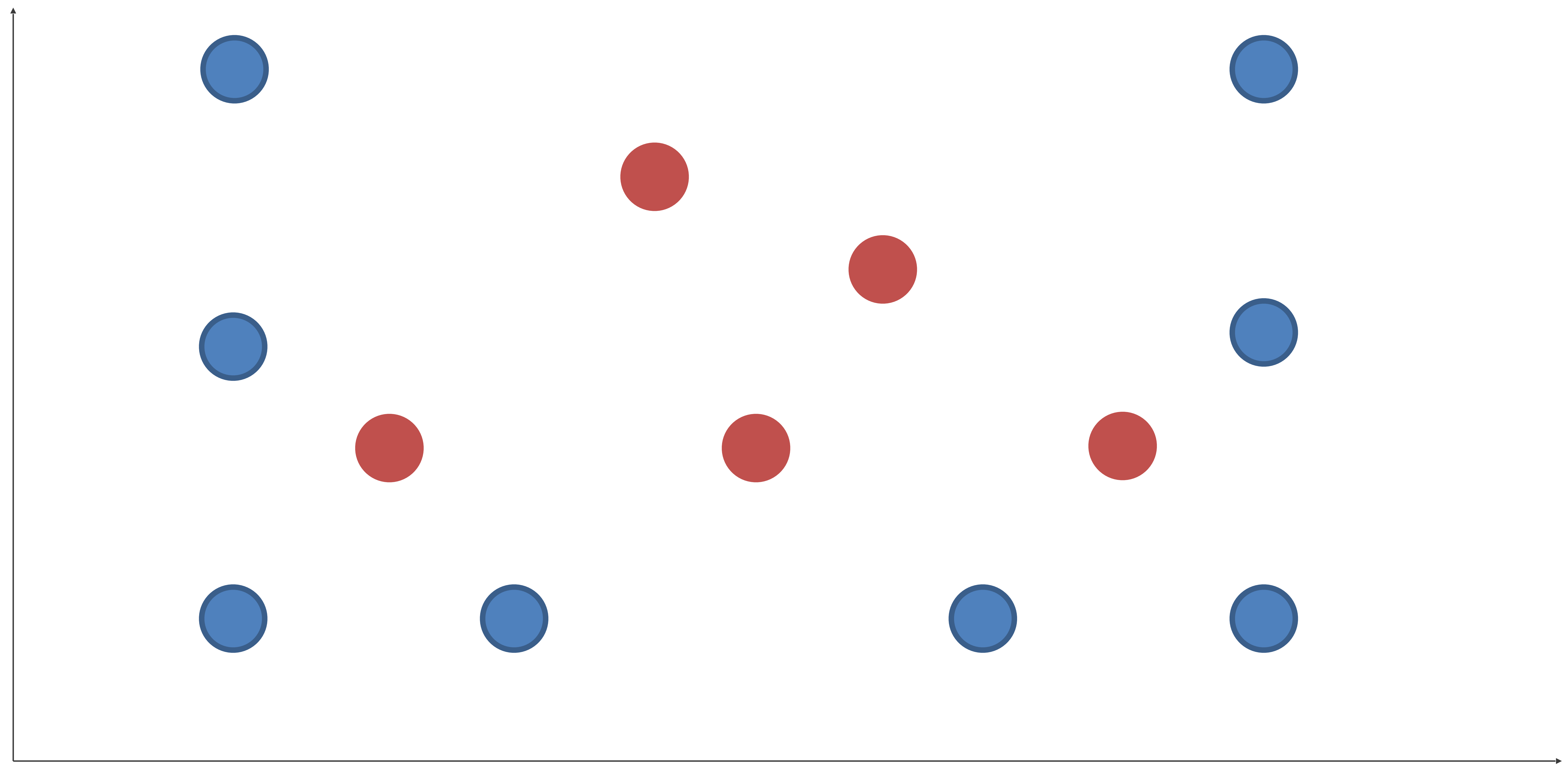
$$R_\ell = \left\{\left\{(x,\ y) \in R_m \middle| \left[x_j < t\right]\right\}\right.,$$

$$R_r = \left\{\left\{(x,\ y) \in R_m \middle| \left[x_j \geq t\right]\right\}\right.$$

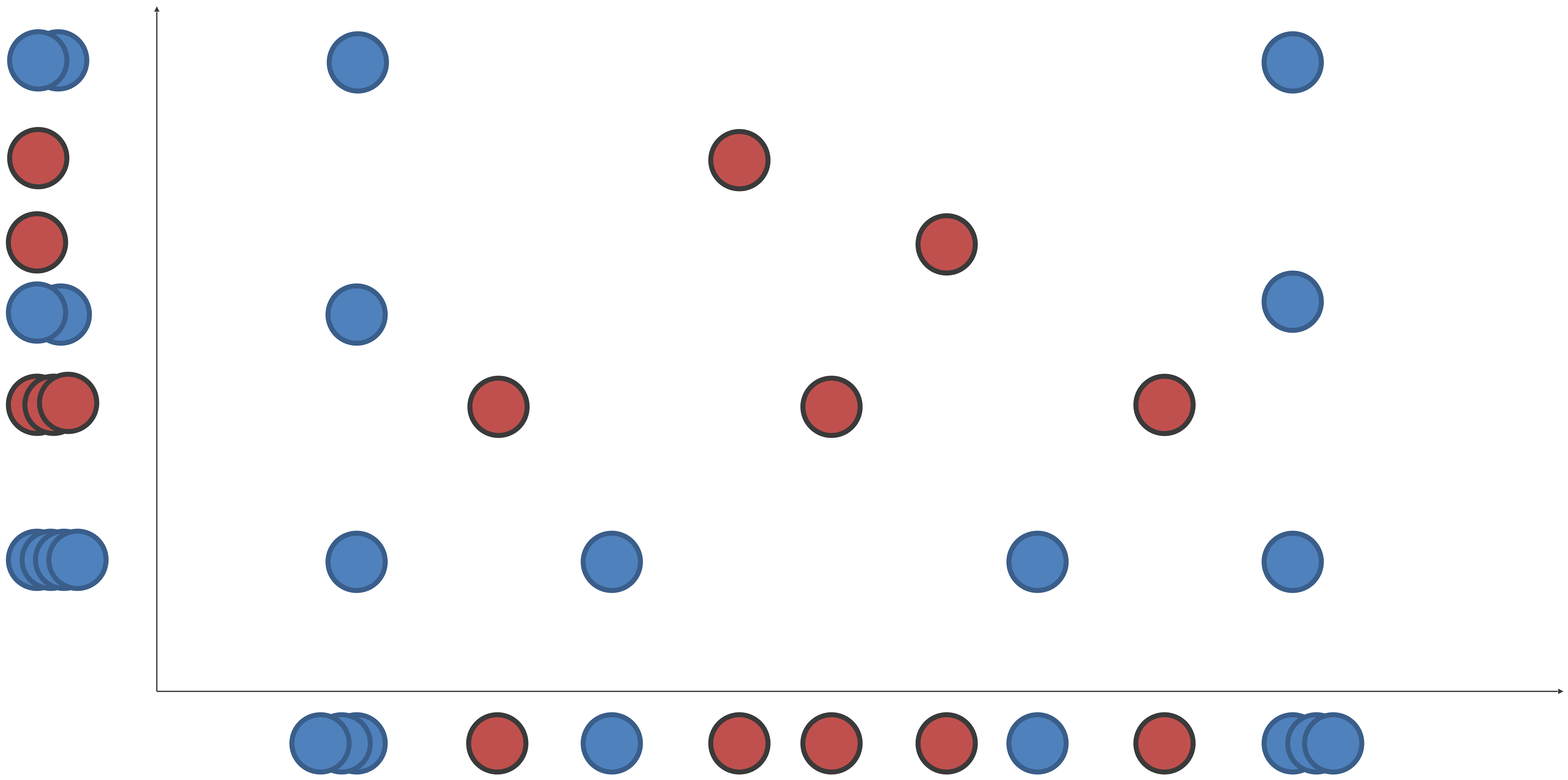
4. Repeat for the child nodes:

$\mathrm{SplitNode}\left(\ell,\ R_\ell\right)$ and $\mathrm{SplitNode}\left(r,\ R_r\right)$

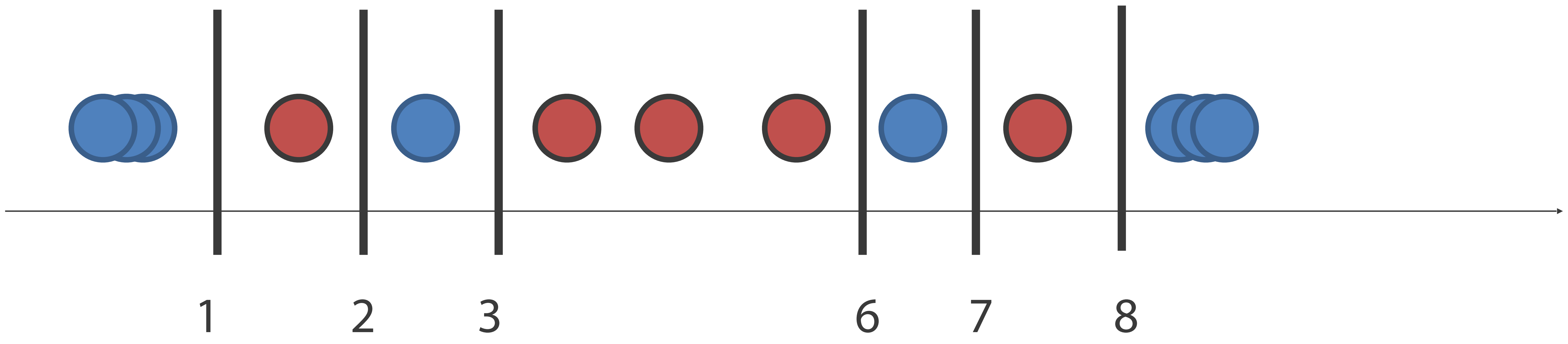
Greedy Algorithm



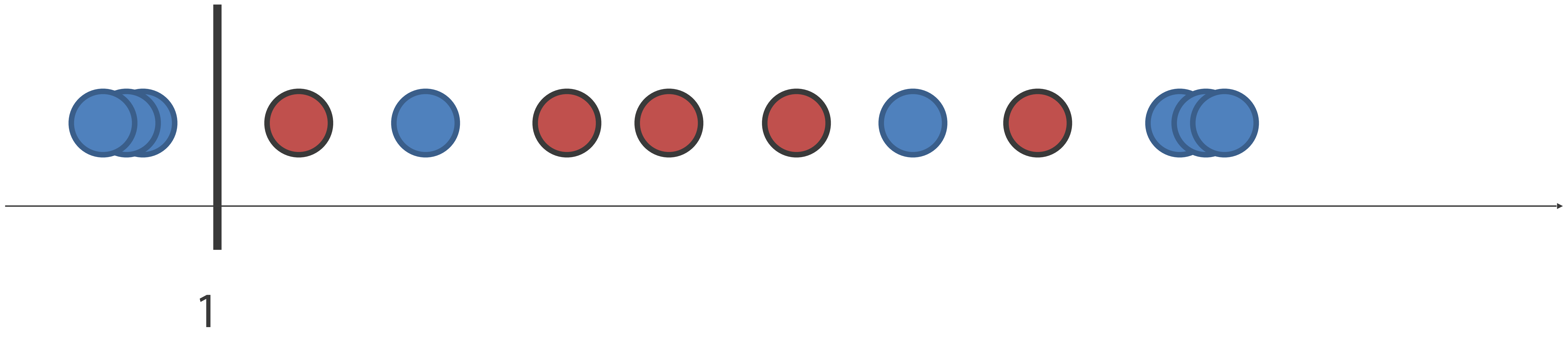
Greedy Algorithm



Splitting on the first feature



Splitting on the first feature

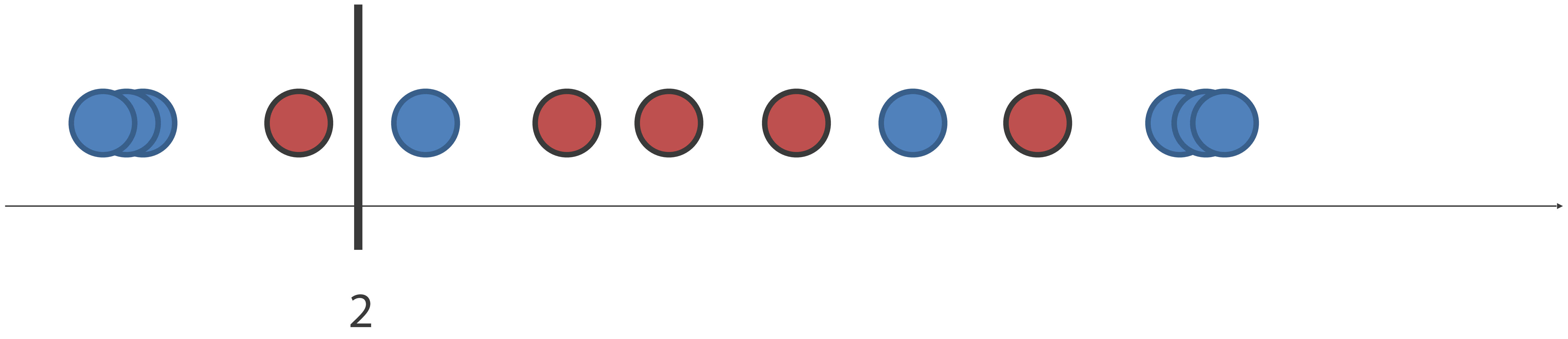


(1, 0)
H(p) = 0

(1/2, 1/2)
H(p) = 1

$$\frac{3}{13}H\left(p_l\right)+\frac{10}{13}H\left(p_r\right)=0.76$$

Splitting on the first feature

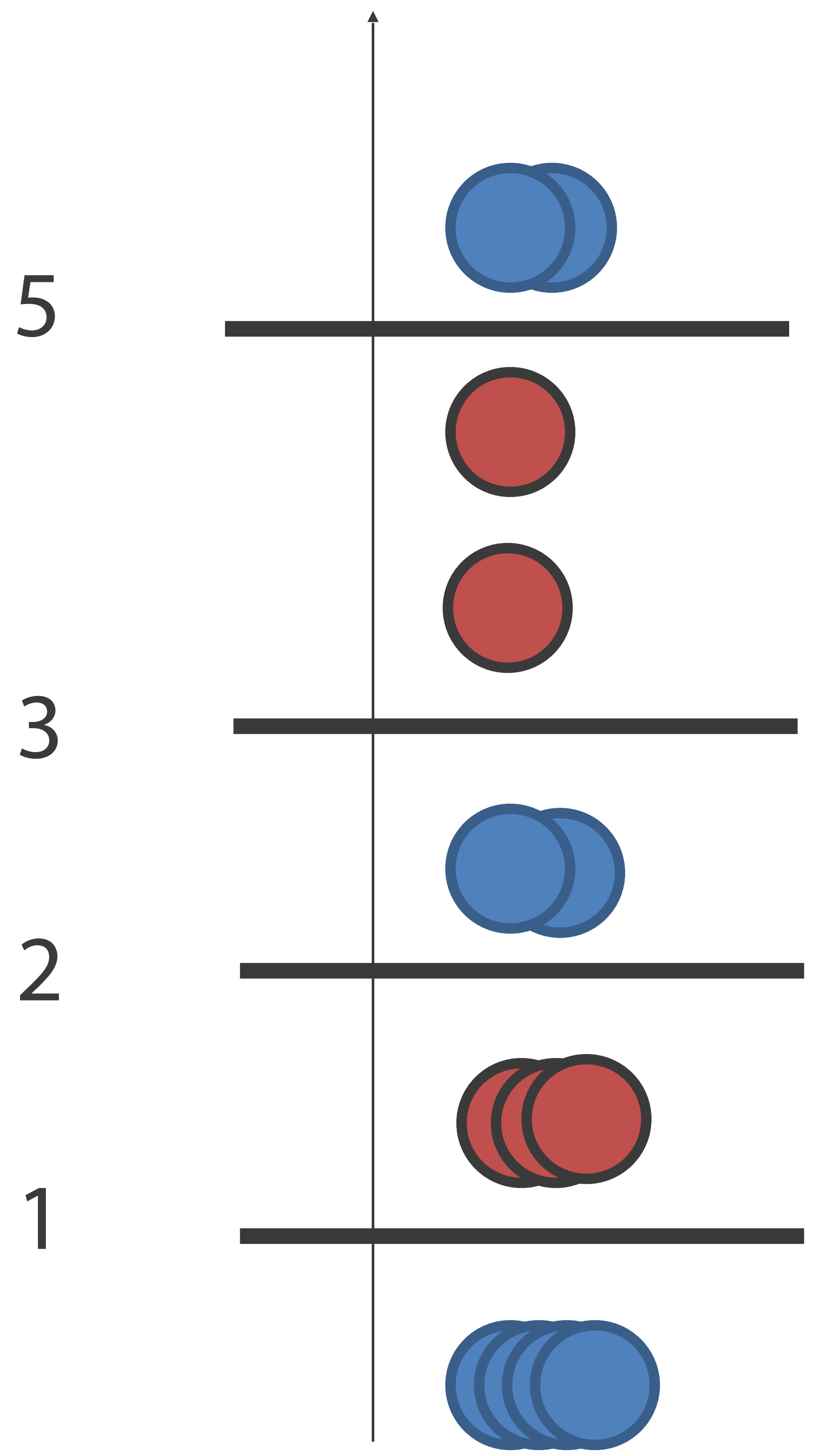


(3/4, 1/4)
H(p) = 0.81

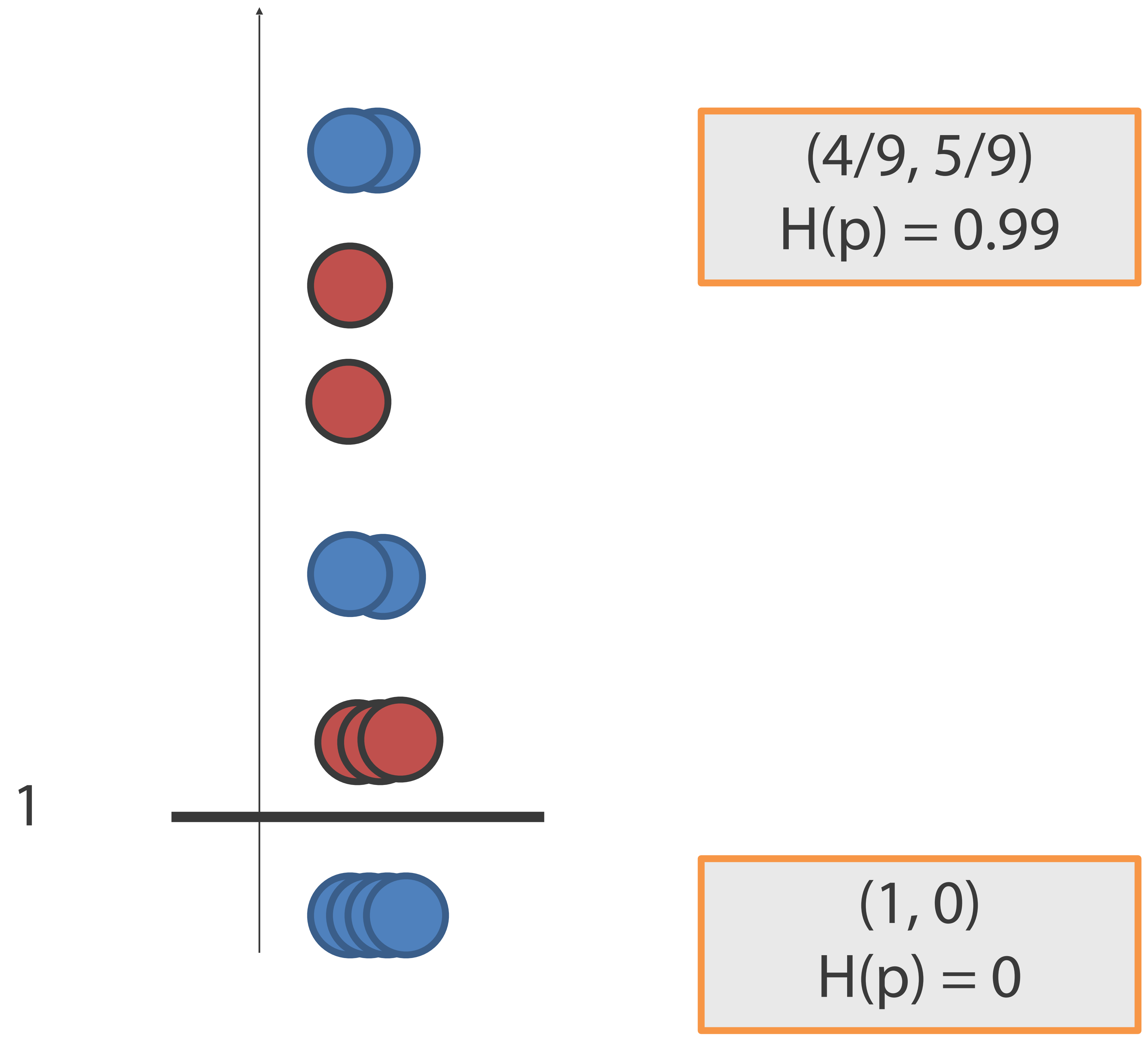
(5/9, 4/9)
H(p) = 0.99

$$\frac{4}{13}H\left(p_l\right) + \frac{9}{13}H\left(p_r\right) = 0.93$$

Splitting on the Second Feature

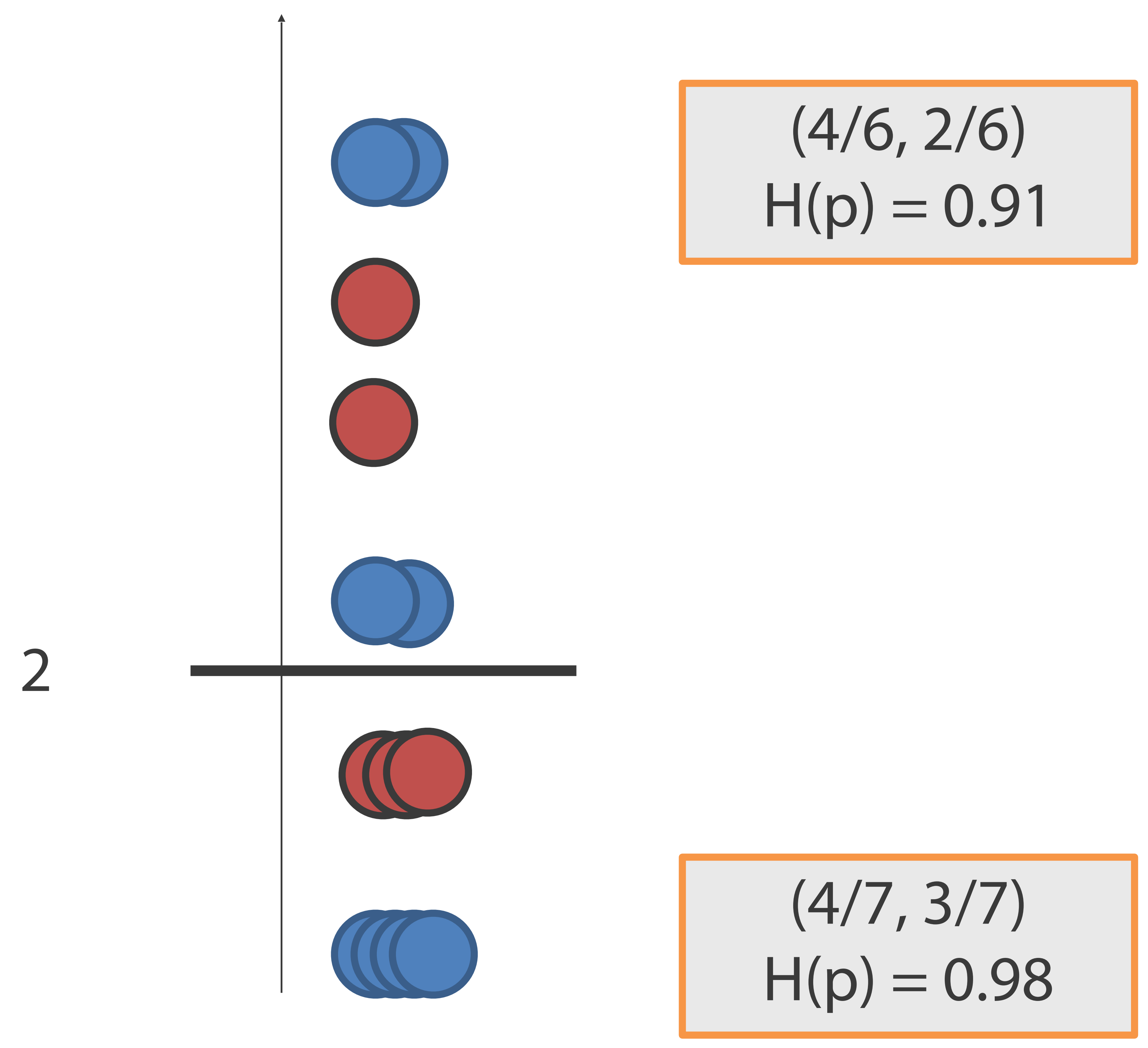


Splitting on the Second Feature



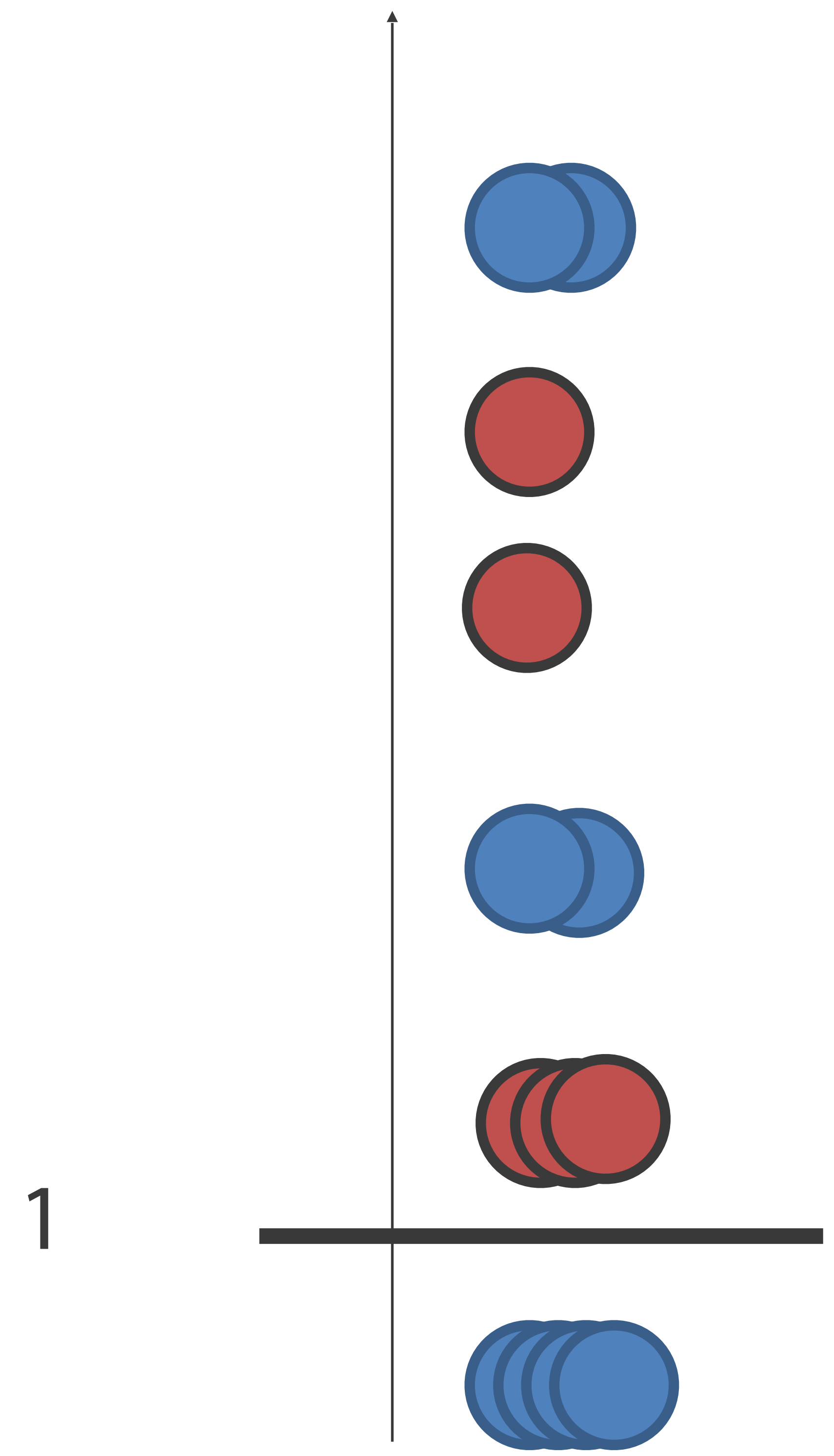
$$\frac{4}{13}H\left(p_l\right) + \frac{9}{13}H\left(p_r\right) = 0.68$$

Splitting on the Second Feature



$$\frac{7}{13}H\left(p_l\right) + \frac{6}{13}H\left(p_r\right) = 0.94$$

Splitting on the Second Feature



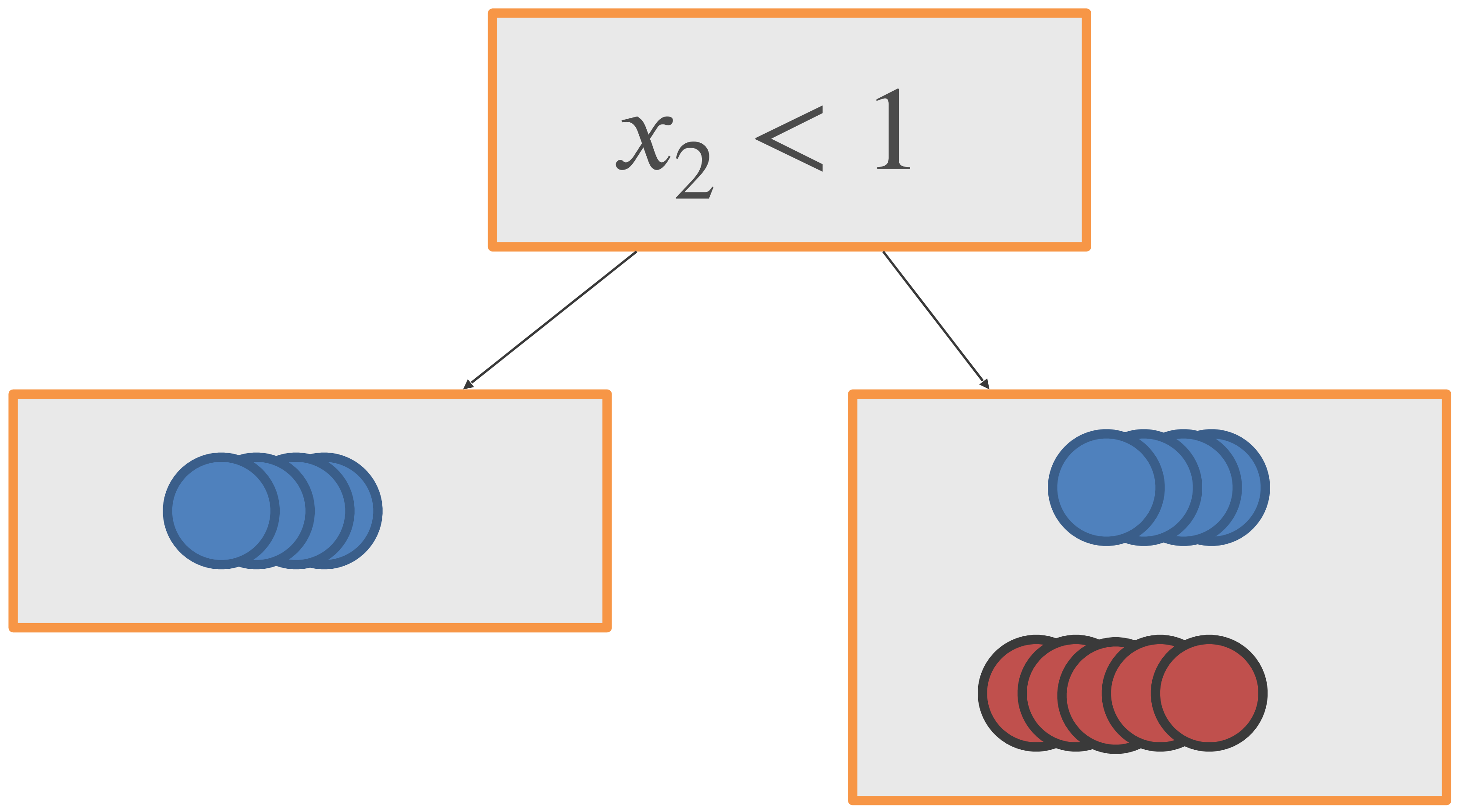
(4/9, 5/9)
H(p) = 0.99

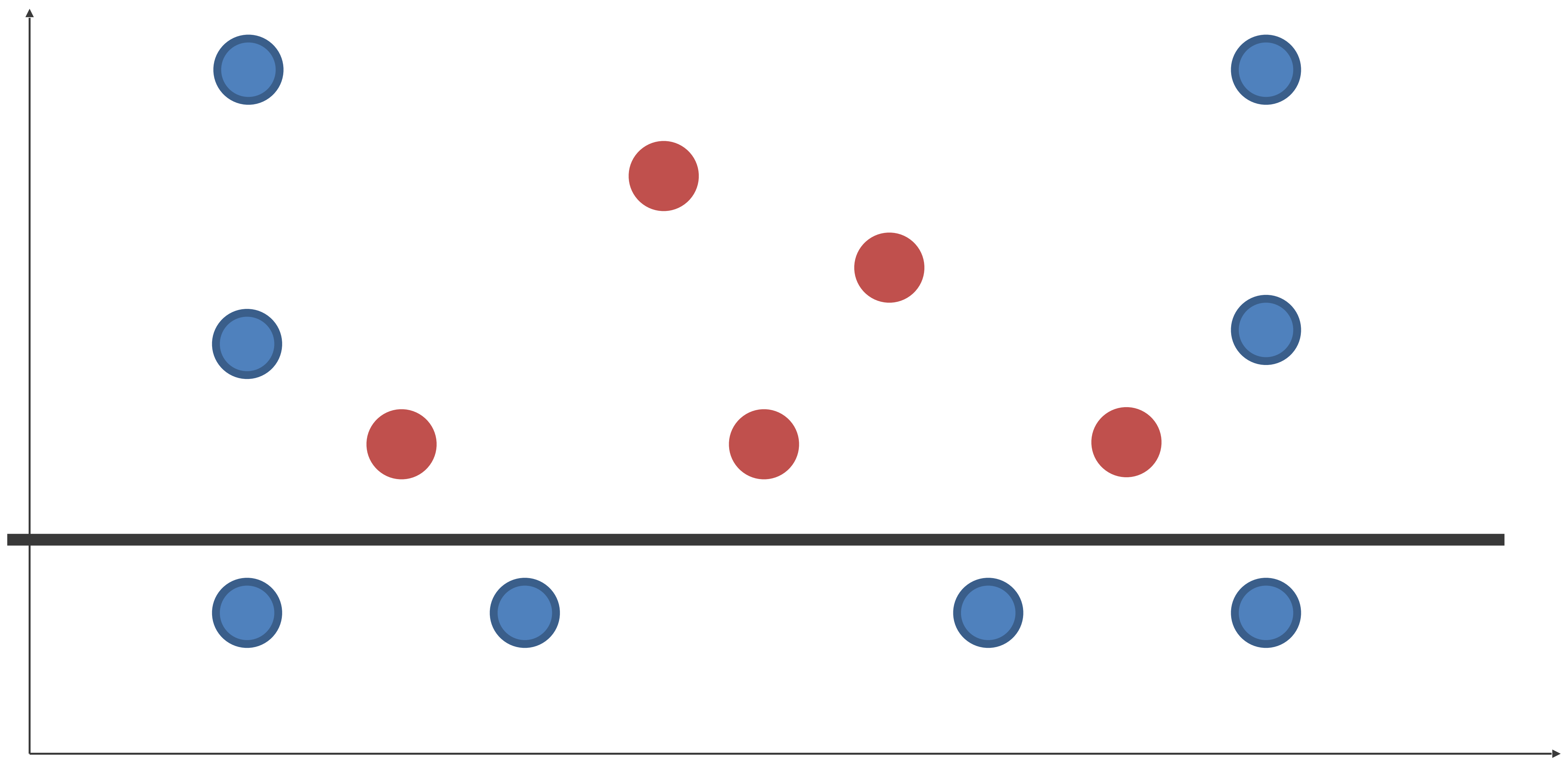
(1, 0)
H(p) = 0

$$\frac{4}{13}H\left(p_l\right) + \frac{9}{13}H\left(p_r\right) = 0.68$$

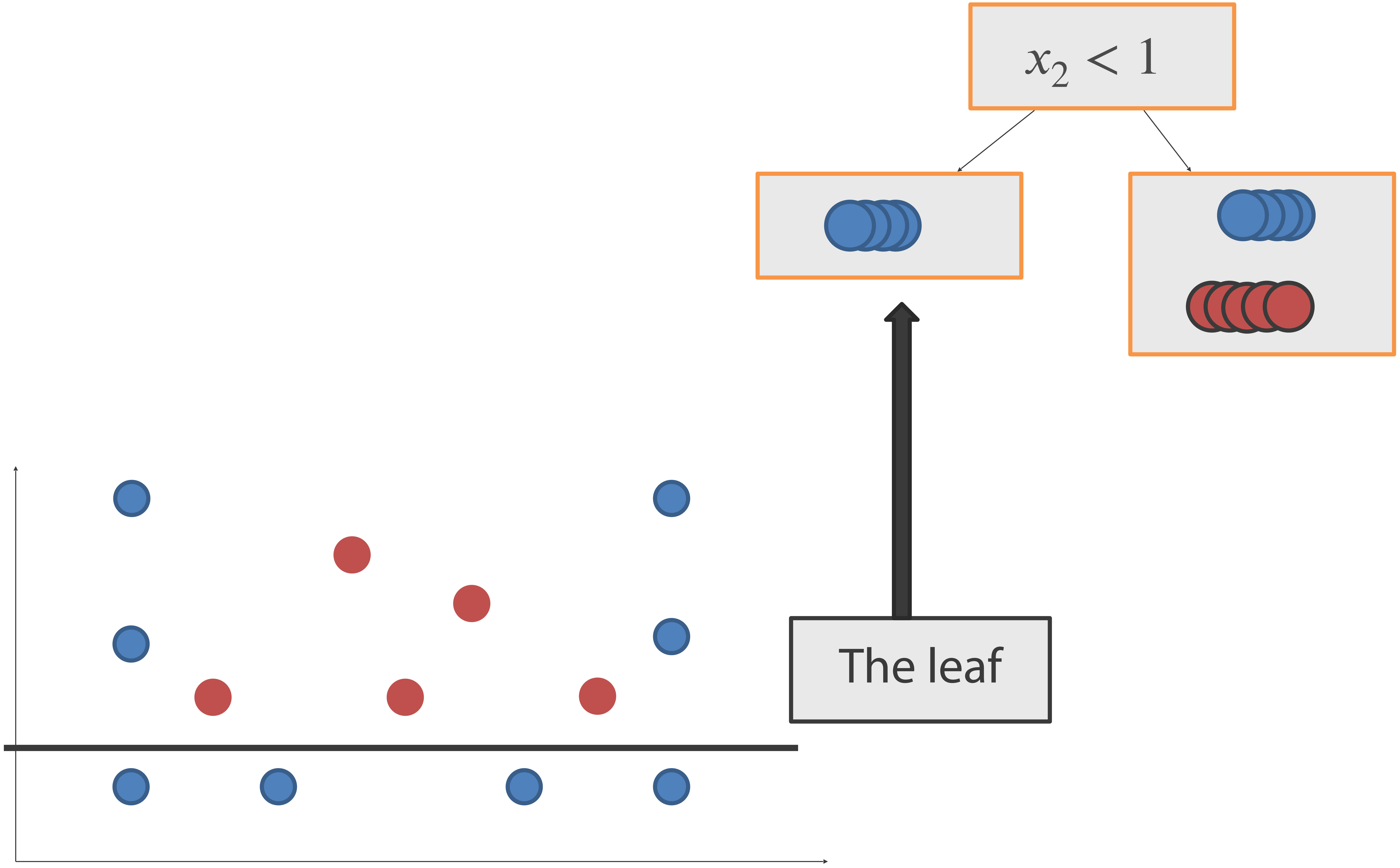
The Best Split

Greedy Algorithm

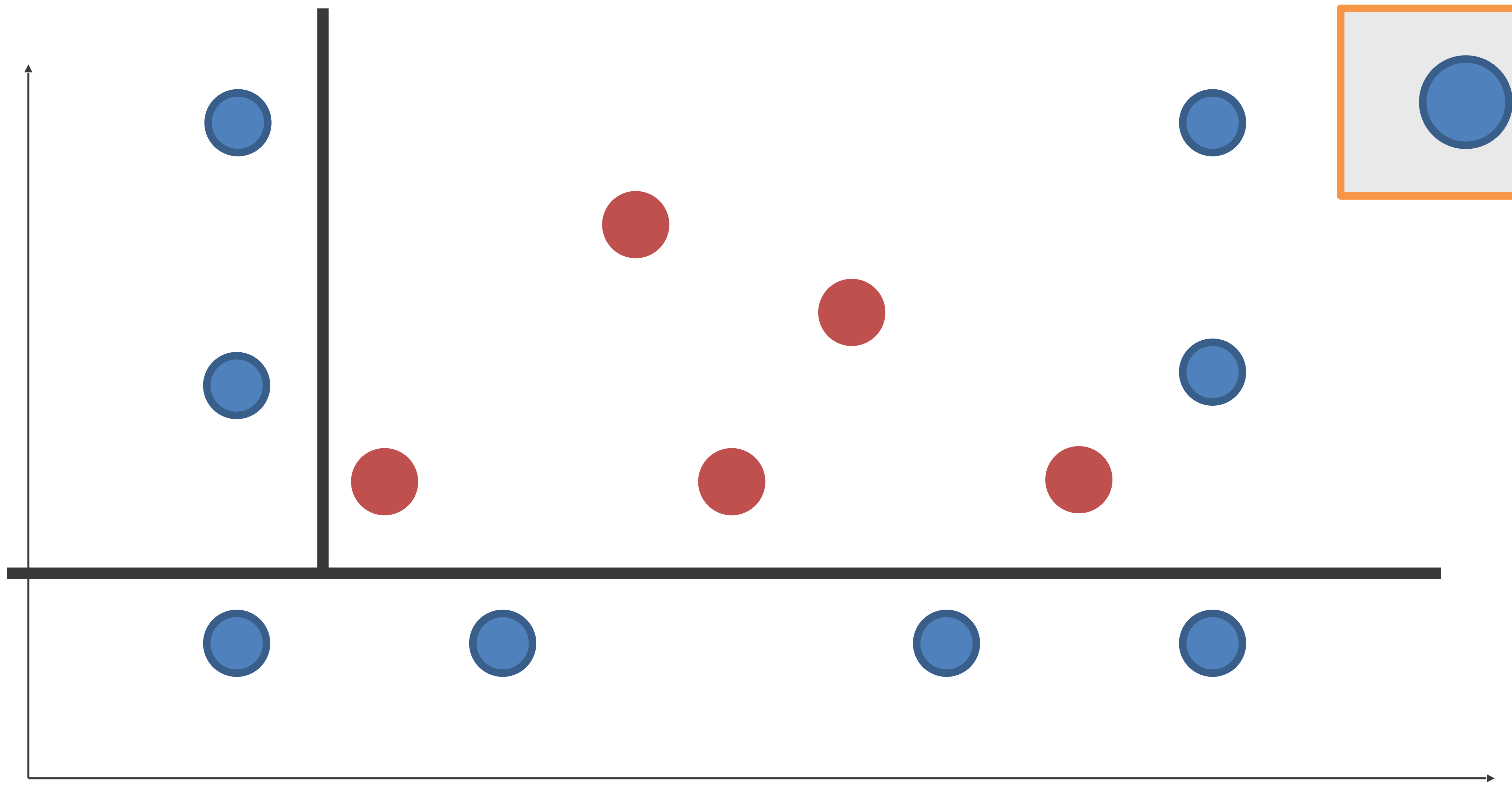


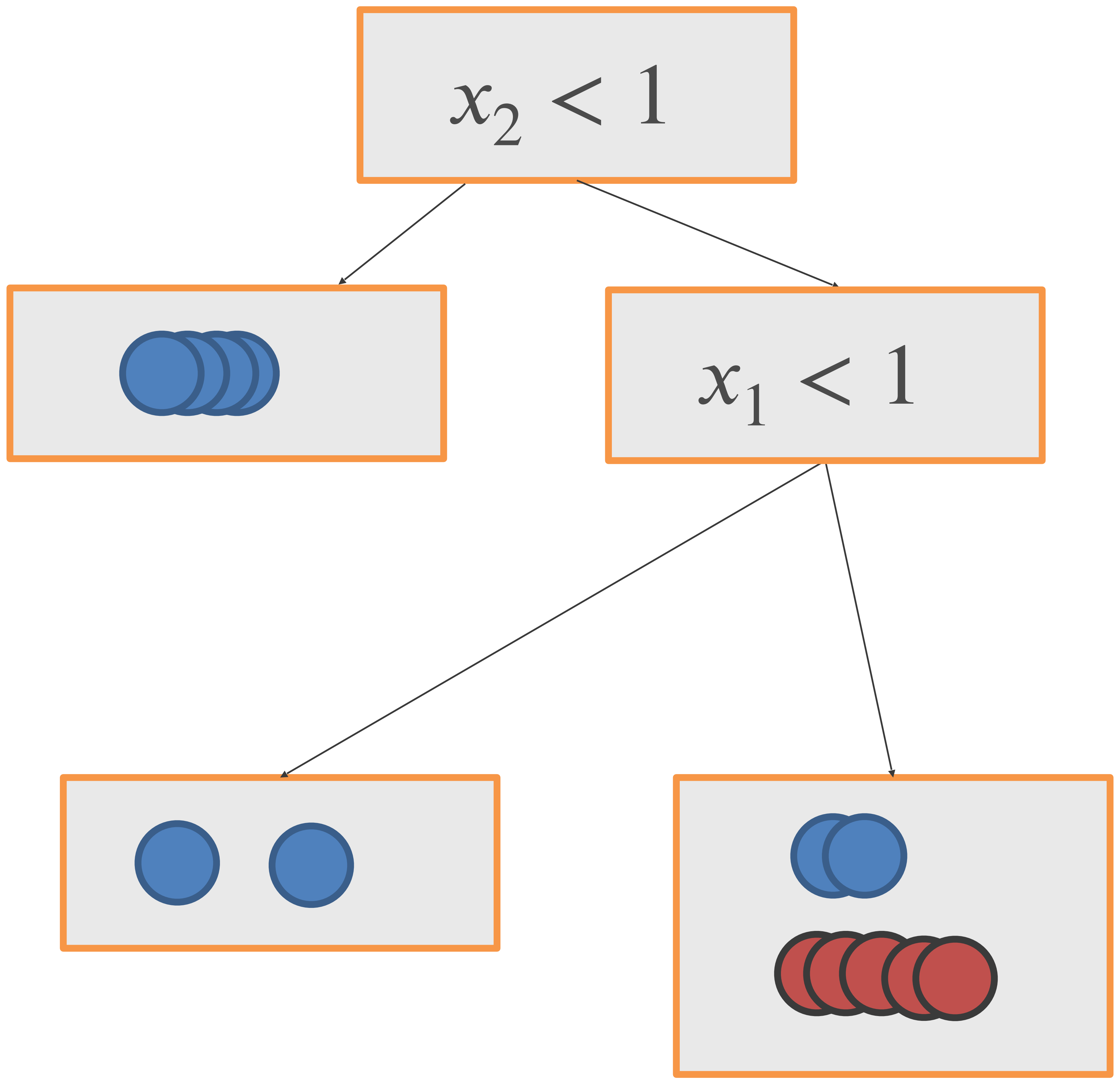


Greedy Algorithm

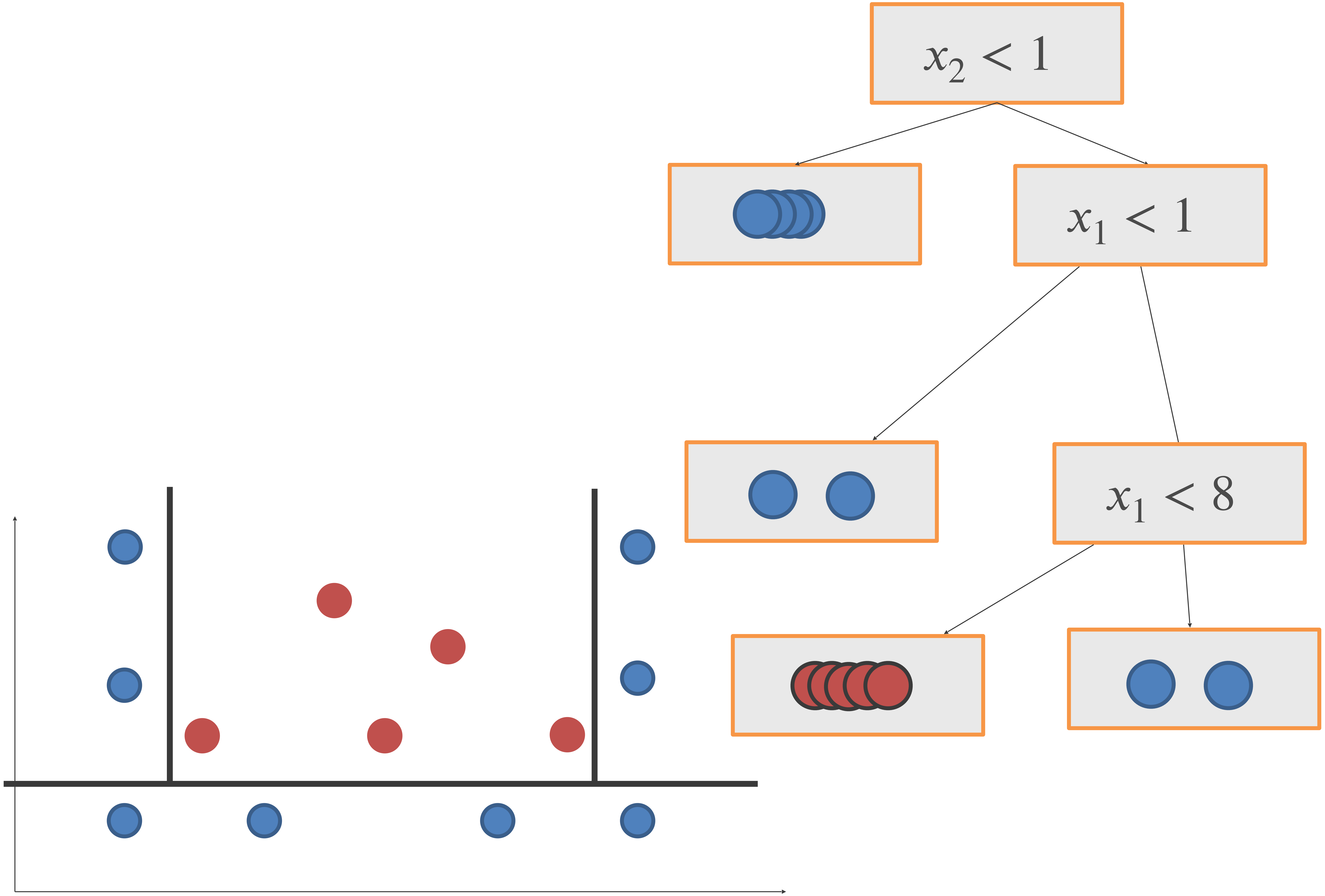


Greedy Algorithm





Greedy Algorithm

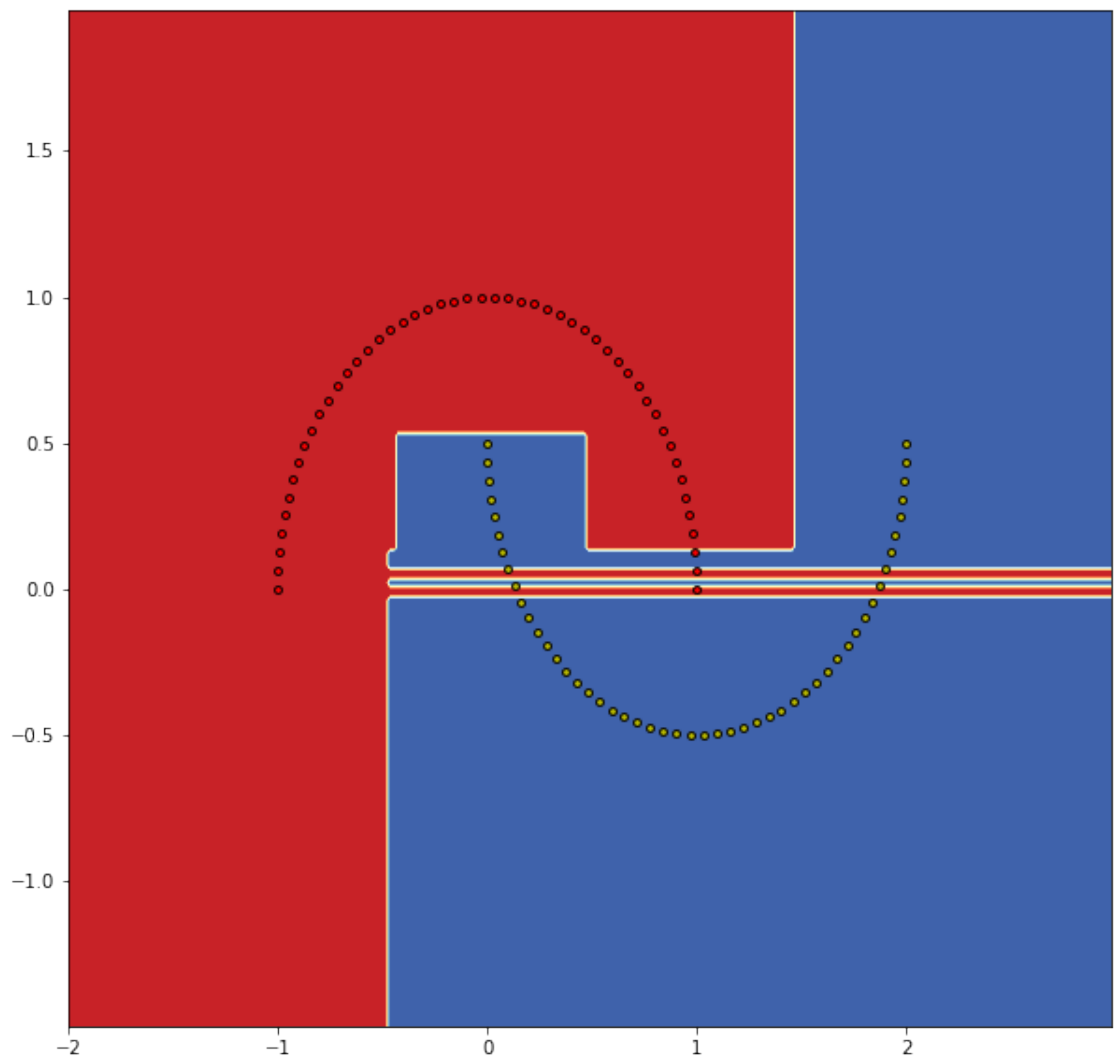


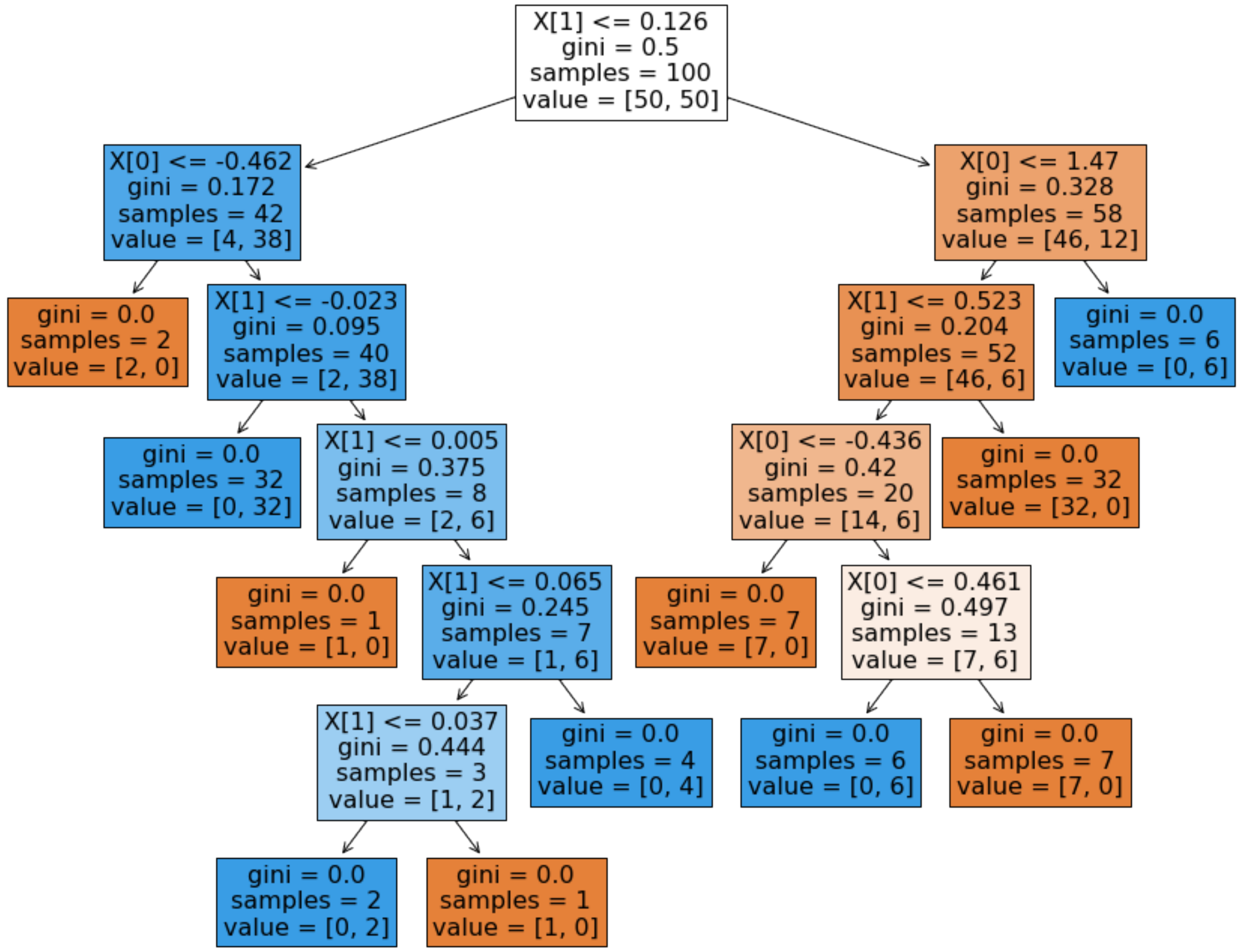
Summary

- We use greedy algorithm to construct a tree
- At each step we select the best split
- The algorithm is quite complex and requires brute force of all the splits at each step

High Variance of Decision Trees

Decision Trees and Overfitting

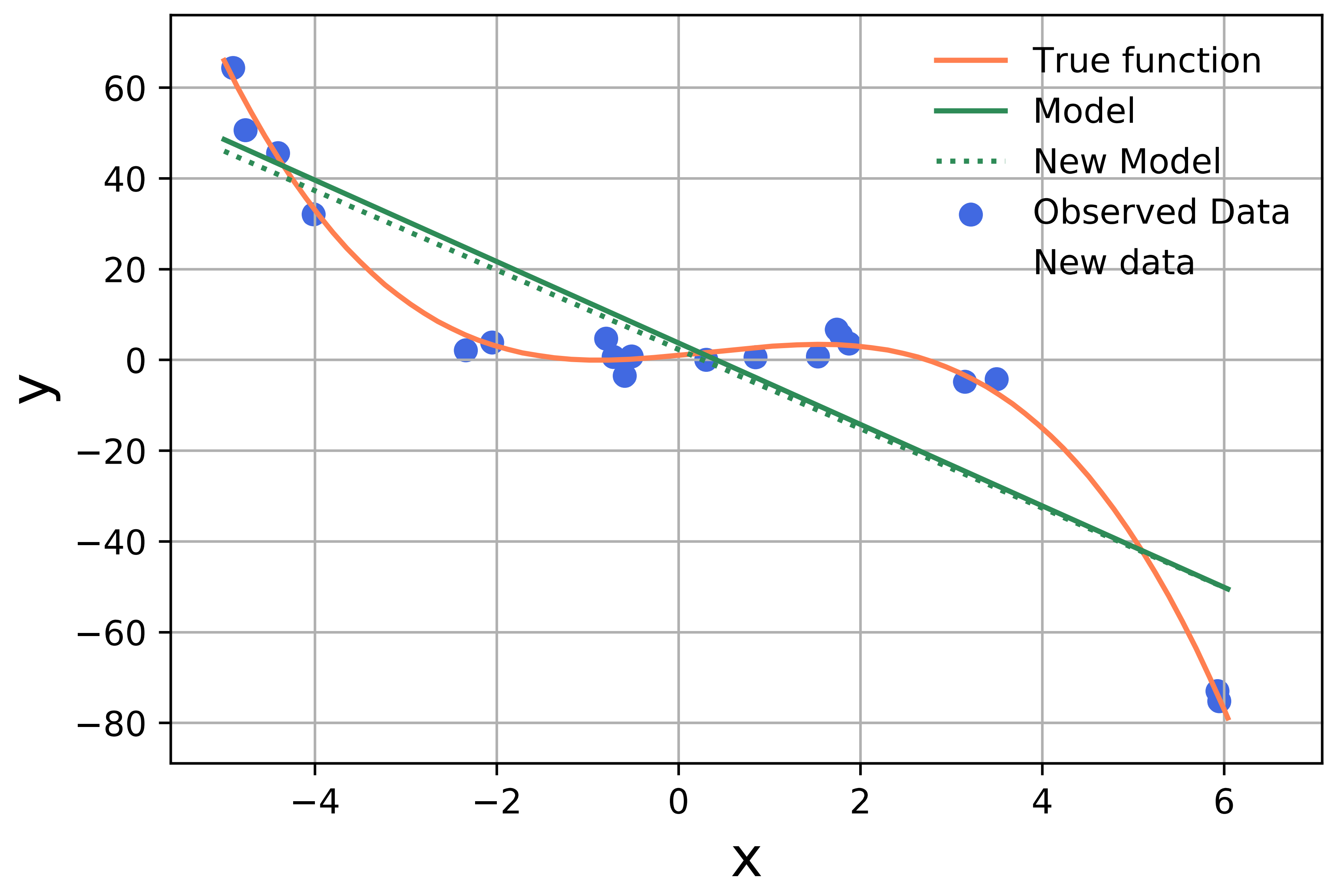




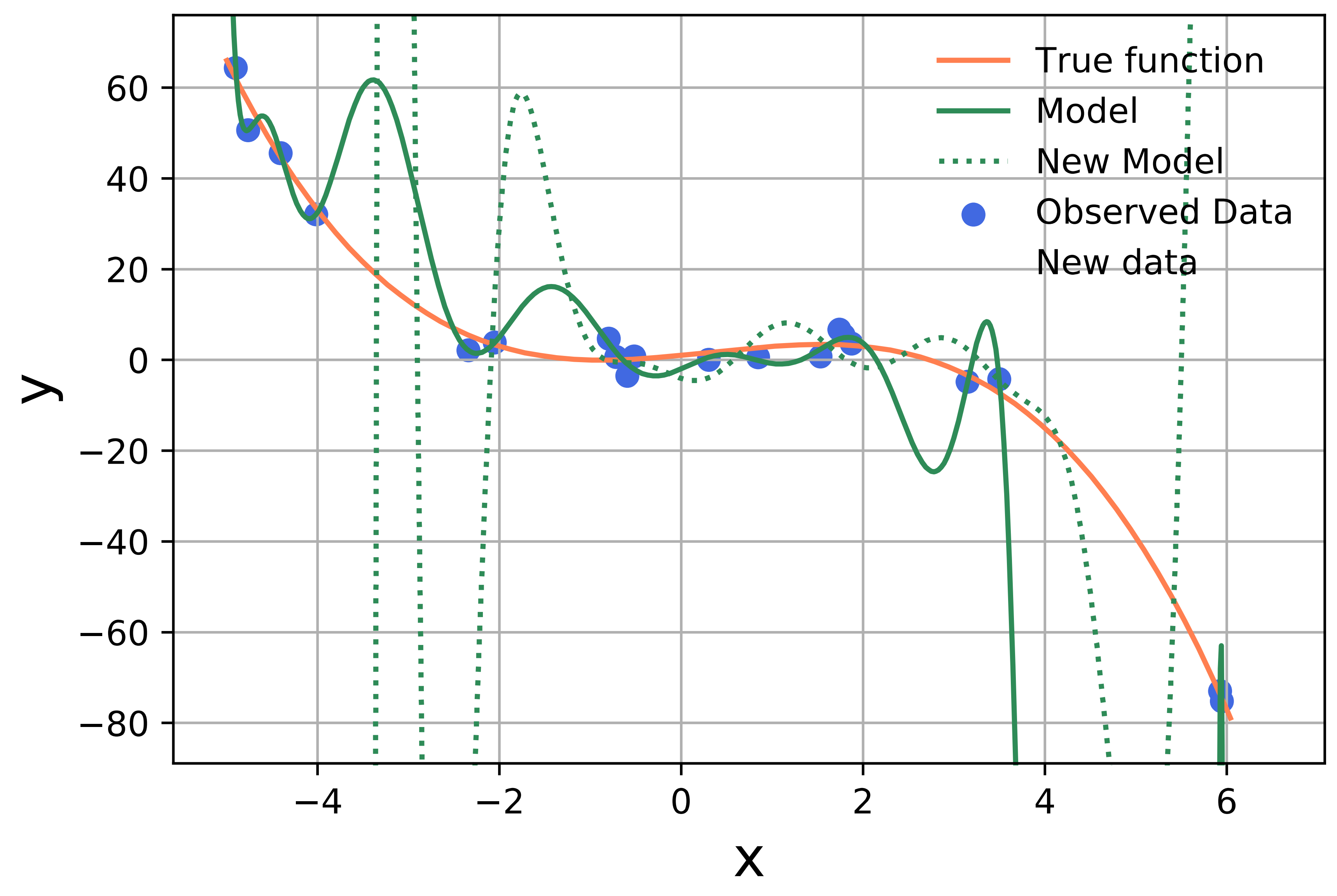
Variance

Variance – sensitivity of the model to the fluctuations in the data

Low Variance



High Variance

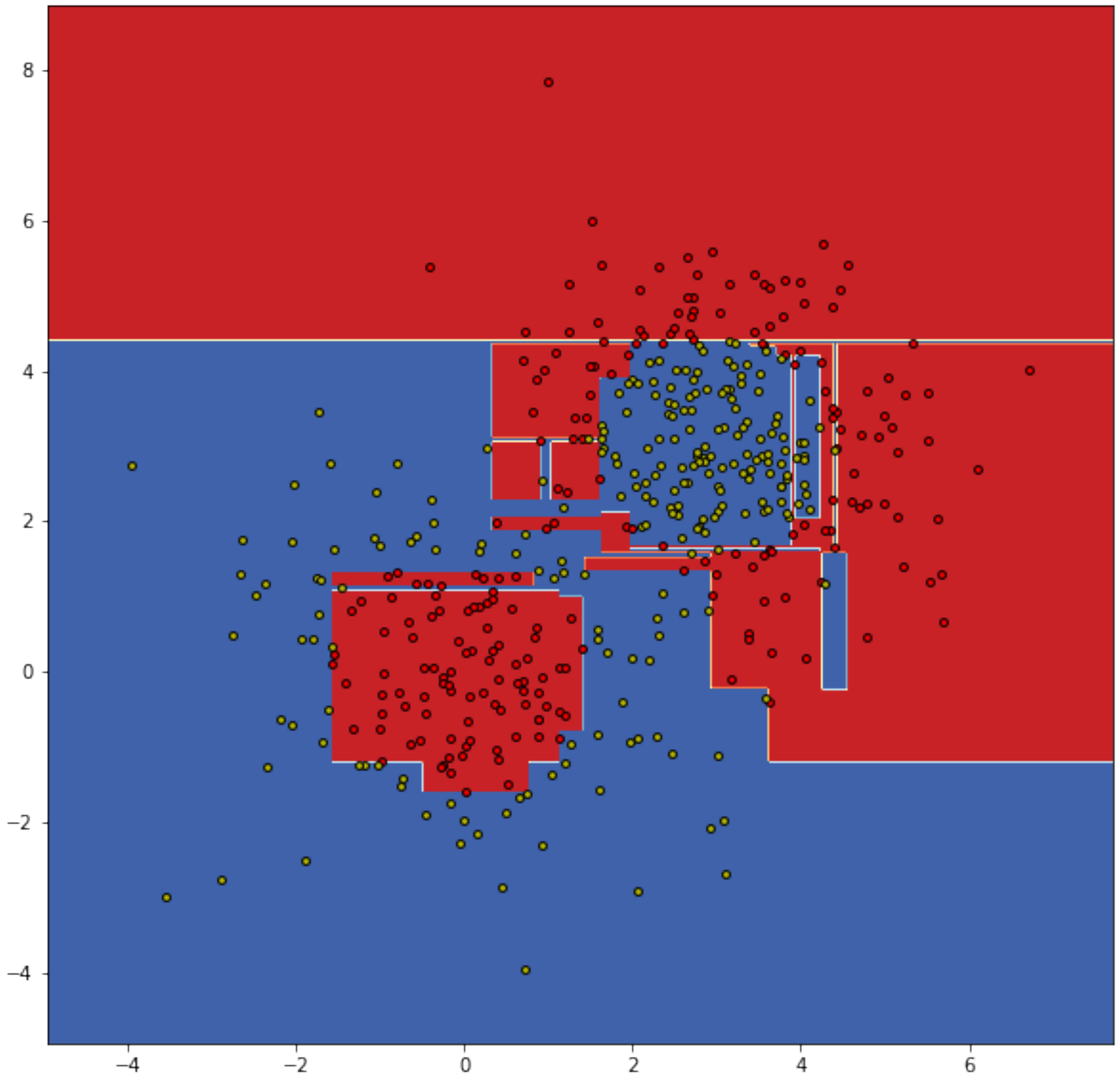


Stability of a Model

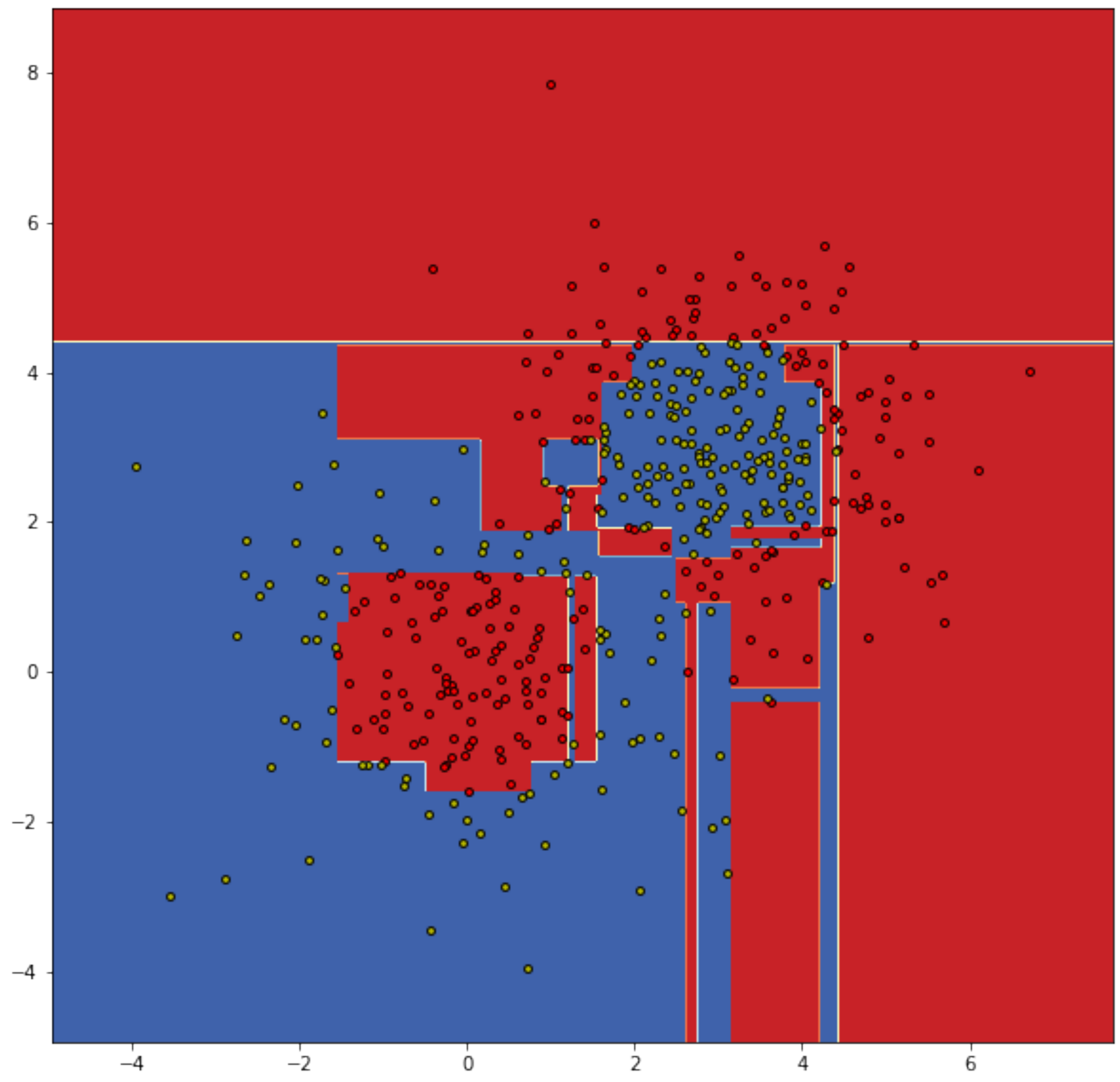
- $X = (x_i, y_i)_{i=1}^{\ell}$ — training dataset
- If the model has low variance, we expect is to look similar if we use different subset of the initial training dataset

- $\widetilde{X}$ — random subsample, approx. 90% from the initial
- Let us train decision trees on different $\widetilde{X}$

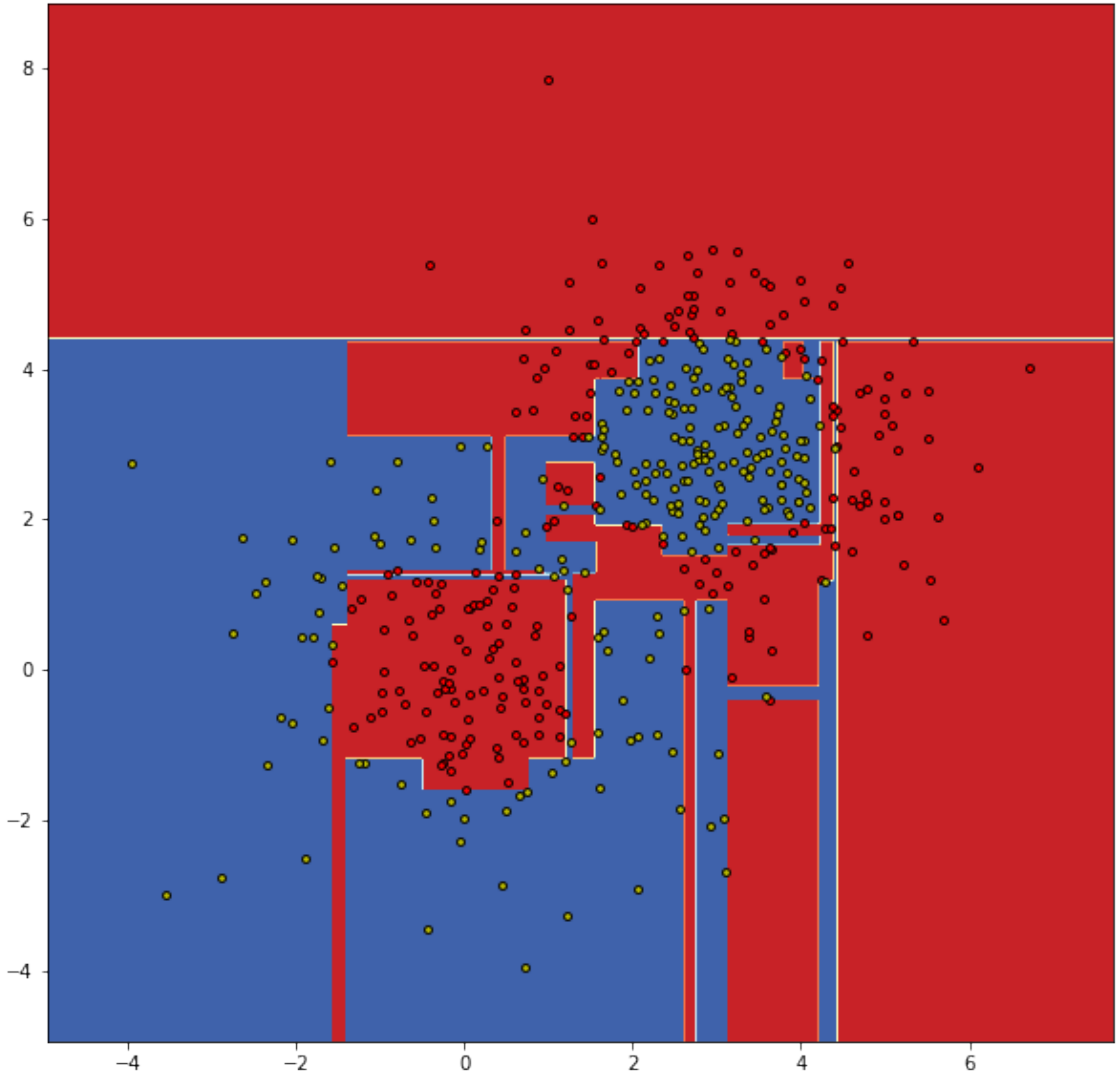
Training Decision Tree on Subsamples



Training Decision Tree on Subsamples

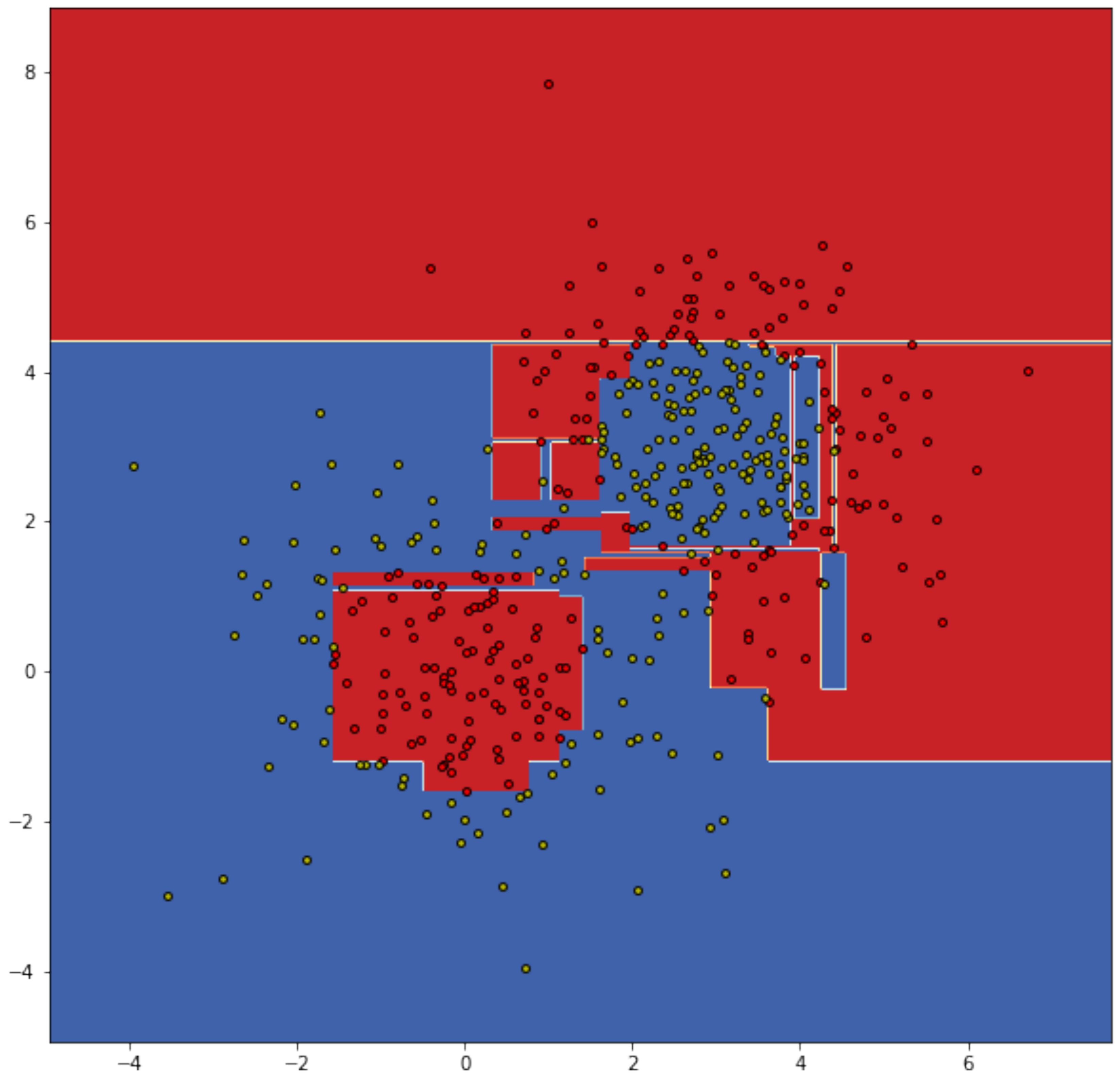


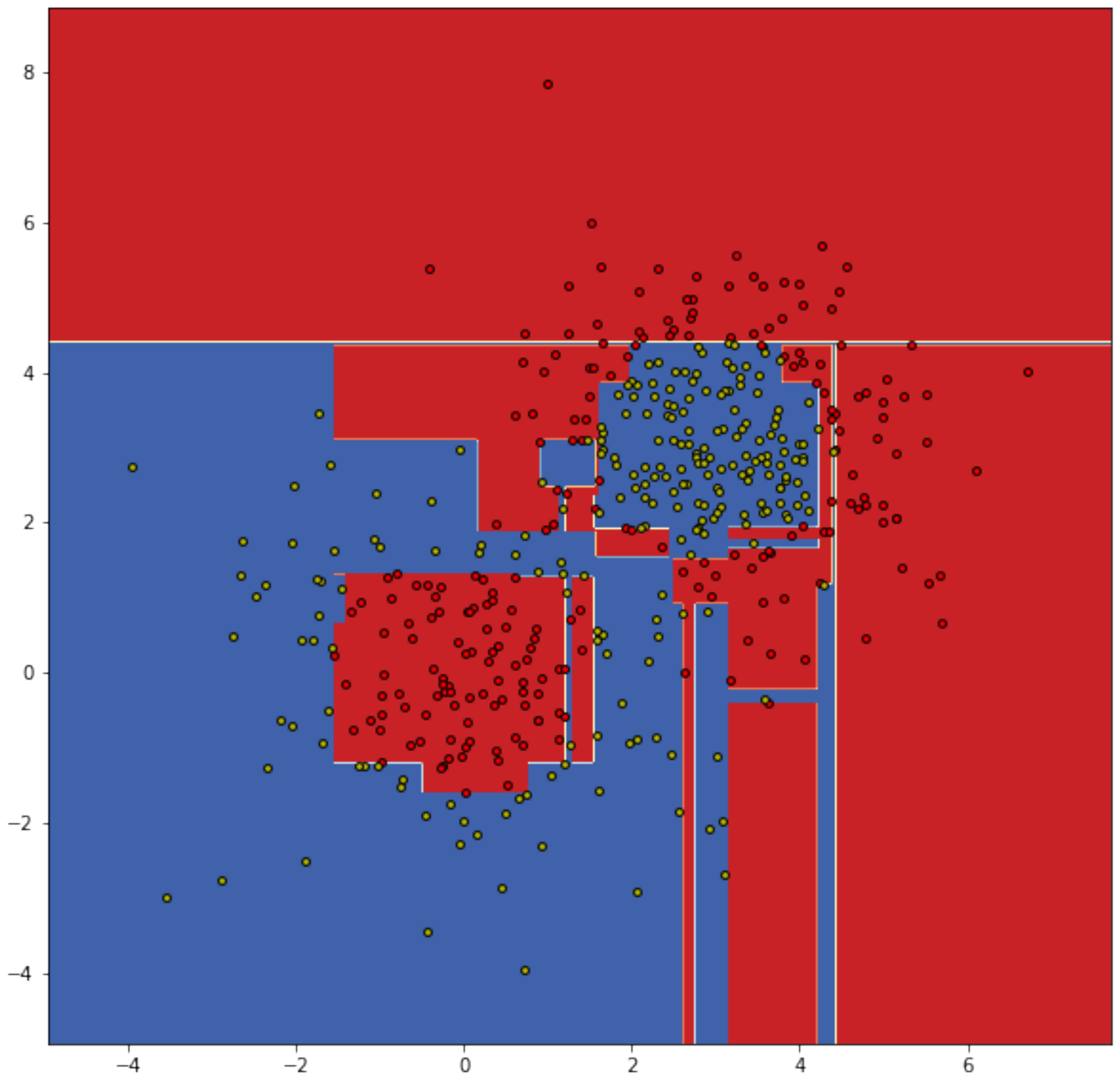
Training Decision Tree on Subsamples



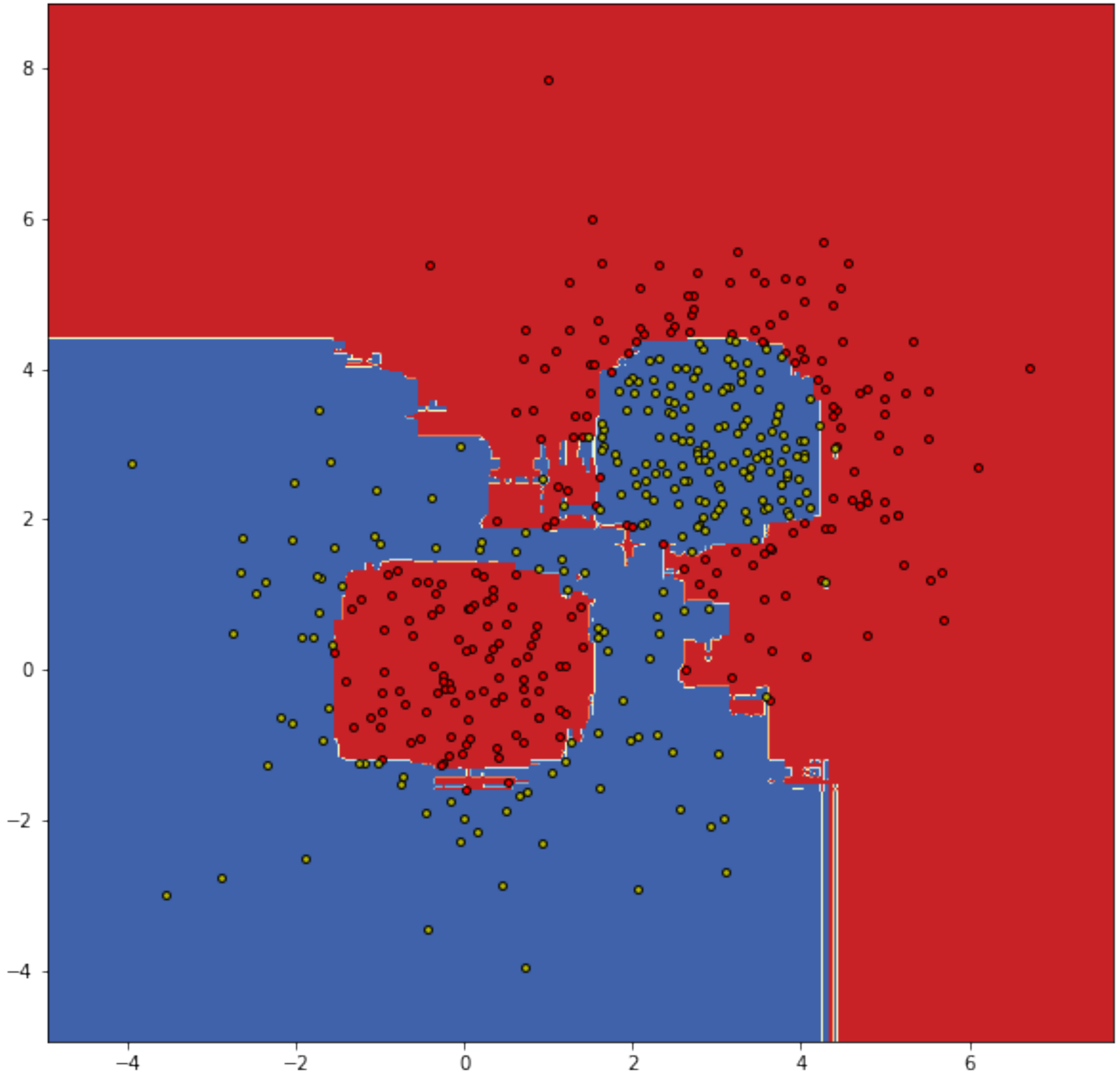
Decision Trees

- High variance
- Lack of smoothness





Composition of Models



Summary

- Decision trees are very sensitive to the changes in the dataset
- Prediction surface of decision trees in very non-smooth
- In practice composition of decision trees is used