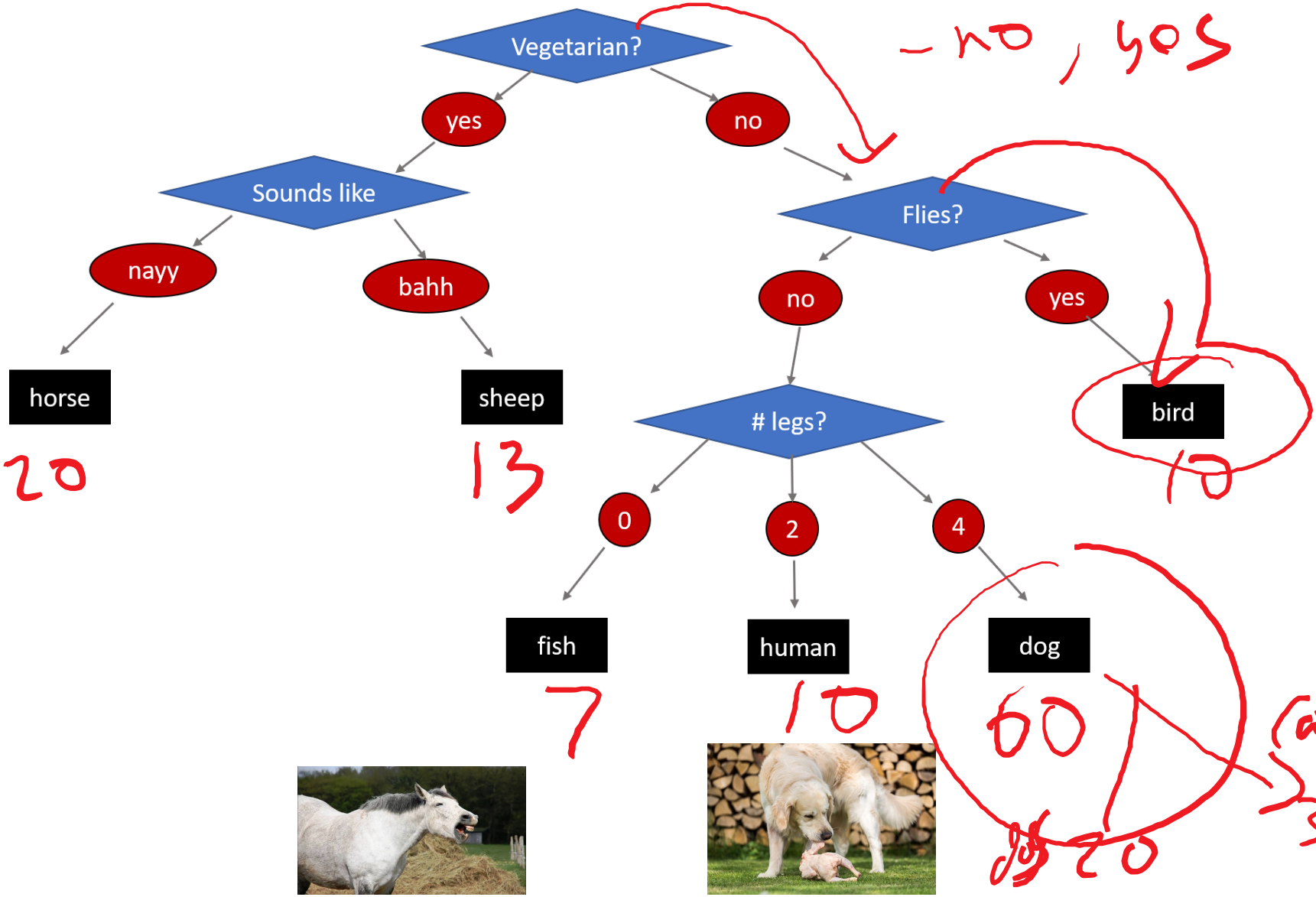
10. Decision trees

- Using a decision tree
- Constructing a decision tree
- Information gain
- Overfitting

Decision tree

Classify the animals on a farm

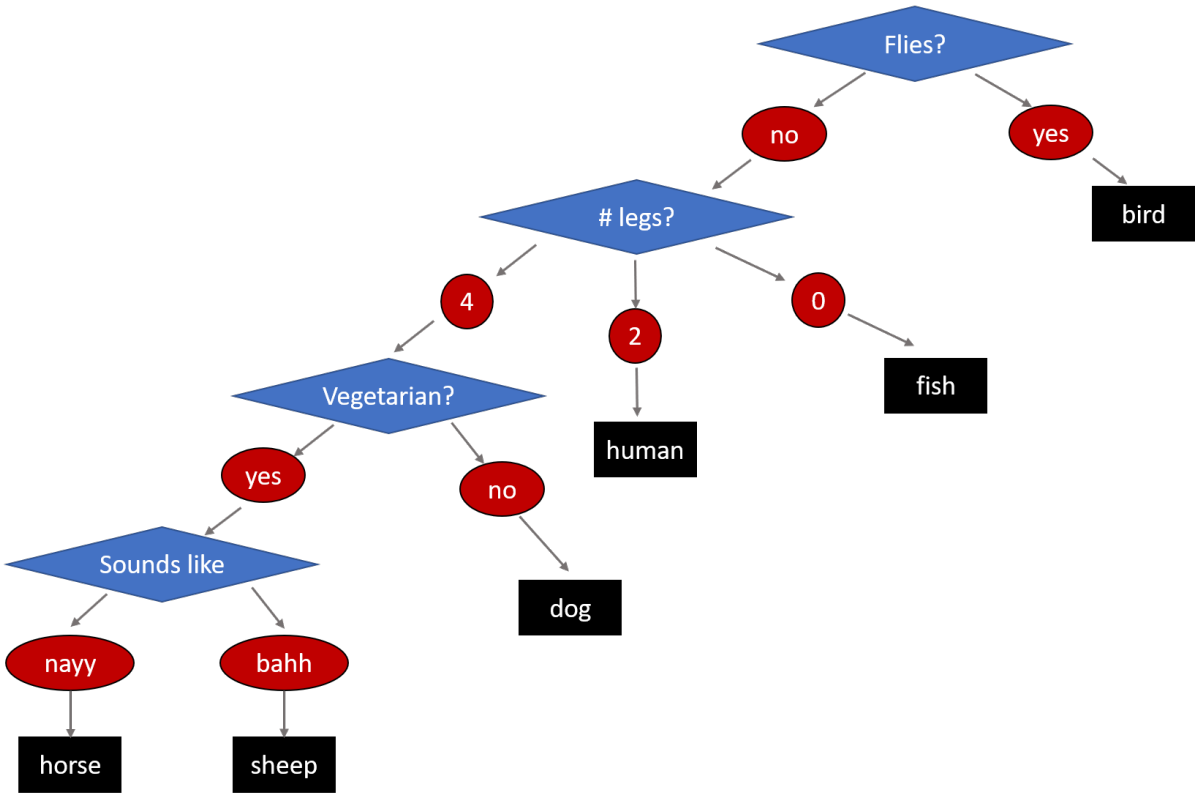


Growing a decision tree

sample	# legs	flies	eats	sound	hair density	animal
1	4	no	grass, grain		650	cow
2	0	no	sardines		0	fish
3	2	no	hot dogs		2,200	human
4	2	yes	worms, bugs		500	bird
⋮						

- Each feature may have different format, different # and range of values
- Desire shallow? deep? wide? skinny tree?
  - Considerations: memory, computation, generalization
- Can choose different loss functions (regression, classification, ...)
  - But can you find a globally optimal tree? Very computationally intense...

Deep vs shallow, skinny vs wide tree



Greedy growing method: regression

- Consider regression with loss $\mathcal{L}(y, \hat{y}) = (y - \hat{y})^2$.
- Denote $x_i[k]$ the kth feature of data sample i
- Given samples $\mathcal{S} \subset \{1, ..., m\}$, loss over set

$$\text{Loss:} \quad \mathcal{E}(\mathcal{S}) = \min_{\hat{y}} \sum_{i \in \mathcal{S}} (\hat{y} - y_i)^2$$

- Given samples $\mathcal{S}$, pick feature to split k, split value v.

$$\text{Quality of split:} \quad \mathcal{Q}(\mathcal{S}, k, v) = \min_{y_1, y_2} \sum_{\substack{i \in \mathcal{S} \\ x_i[k] < v}} (y_1 - y_i)^2 + \sum_{\substack{i \in \mathcal{S} \\ x_i[k] \geq v}} (y_2 - y_i)^2$$

Greedy growing method: regression

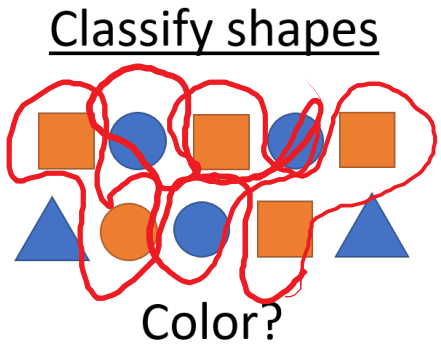
Loss: $$\mathcal{E}(\mathcal{S}) = \min_{\hat{y}} \sum_{i \in \mathcal{S}} (\hat{y} - y_i)^2$$

Quality of split: $$\mathcal{Q}(\mathcal{S}, k, v) = \min_{y_1, y_2} \sum_{\substack{i \in \mathcal{S} \\ x_i[k] < v}} (y_1 - y_i)^2 + \sum_{\substack{i \in \mathcal{S} \\ x_i[k] \geq v}} (y_2 - y_i)^2$$

First split:

- Start with $\mathcal{S} = \{1, ..., m\}$ entire training set.
- Find k and v the best split over $\mathcal{S}$.

$$k, v = \operatorname*{argmin}_{k,v} \mathcal{Q}(\mathcal{S}, k, v)$$



- Split forms two regions (**leaves** $= \{\mathcal{S}_1, \mathcal{S}_2\}$)

$$\mathcal{S}_1 = \{i \in \mathcal{S} : x_i[k] < v\}, \quad \mathcal{S}_2 = \{i \in \mathcal{S} : x_i[k] \geq v\}$$

Greedy growing method: ~~regression~~ classification

Loss: $$\mathcal{E}(\mathcal{S}) = \min_{\hat{y}} \sum_{i \in \mathcal{S}} (\hat{y} - y_i)^2$$

Quality of split: $$\mathcal{Q}(\mathcal{S}, k, v) = \min_{y_1, y_2} \sum_{\substack{i \in \mathcal{S} \\ x_i[k] < v}} (y_1 - y_i)^2 + \sum_{\substack{i \in \mathcal{S} \\ x_i[k] \geq v}} (y_2 - y_i)^2$$

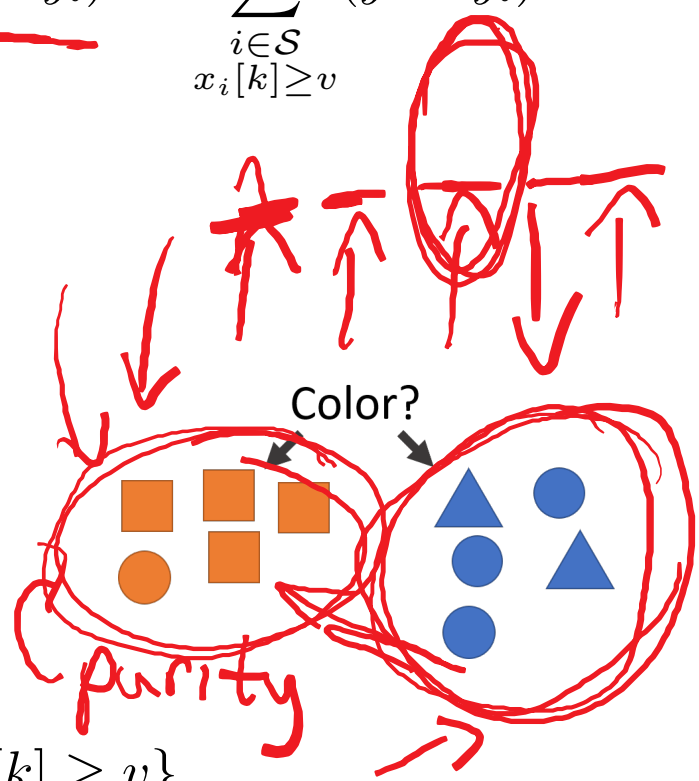
First split:

- Start with $\mathcal{S} = \{1, ..., m\}$ entire training set.
- Find k and v the best split over $\mathcal{S}$.

$$k, v = \operatorname*{argmin}_{k,v} \mathcal{Q}(\mathcal{S}, k, v)$$

- Split forms two regions (**leaves** $= \{\mathcal{S}_1, \mathcal{S}_2\}$)

$$\mathcal{S}_1 = \{i \in \mathcal{S} : x_i[k] < v\}, \quad \mathcal{S}_2 = \{i \in \mathcal{S} : x_i[k] \geq v\}$$



Greedy growing method: regression

Loss: $$\mathcal{E}(\mathcal{S}) = \min_{\hat{y}} \sum_{i \in \mathcal{S}} (\hat{y} - y_i)^2$$

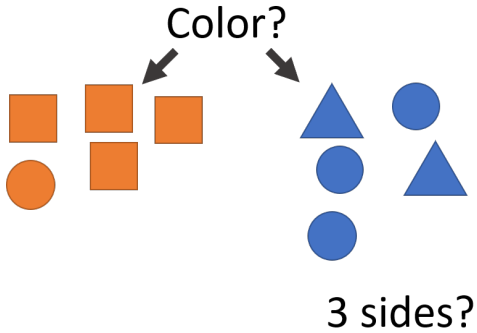
Quality of split: $$\mathcal{Q}(\mathcal{S}, k, v) = \min_{y_1, y_2} \sum_{\substack{i \in \mathcal{S} \\ x_i[k] < v}} (y_1 - y_i)^2 + \sum_{\substack{i \in \mathcal{S} \\ x_i[k] \geq v}} (y_2 - y_i)^2$$

Recursive procedure:

- Pick leaf with worst loss $\bar{\mathcal{S}} = \operatorname{argmax}_{\bar{\mathcal{S}} \in \textbf{leaves}} h(\bar{\mathcal{S}})$
- Find k, s the best split over picked region $\bar{\mathcal{S}}$.

$$k, v = \operatorname*{argmin}_{k,v} f(\bar{\mathcal{S}}, k, v),$$

$$\mathcal{S}'_1 = \{i \in \mathcal{S} : x_i[k] < v\}, \quad \mathcal{S}'_2 = \{i \in \mathcal{S} : x_i[k] \geq v\}$$



- Update leaves

$$\textbf{leaves} := (\textbf{leaves} \setminus \bar{\mathcal{S}}) \cup \{\mathcal{S}'_1, \mathcal{S}'_2\}$$

Greedy growing method: regression

Loss: $$\mathcal{E}(\mathcal{S}) = \min_{\hat{y}} \sum_{i \in \mathcal{S}} (\hat{y} - y_i)^2$$

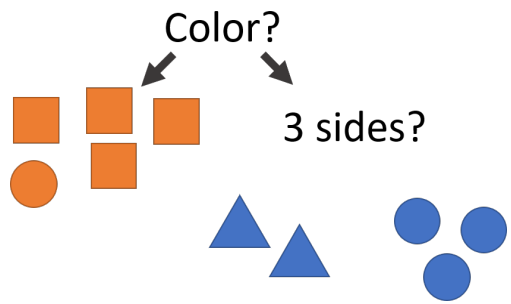
Quality of split: $$\mathcal{Q}(\mathcal{S}, k, v) = \min_{y_1, y_2} \sum_{\substack{i \in \mathcal{S} \\ x_i[k] < v}} (y_1 - y_i)^2 + \sum_{\substack{i \in \mathcal{S} \\ x_i[k] \geq v}} (y_2 - y_i)^2$$

Recursive procedure:

- Pick leaf with worst loss $\bar{\mathcal{S}} = \operatorname{argmax}_{\bar{\mathcal{S}} \in \textbf{leaves}} h(\bar{\mathcal{S}})$
- Find k, s the best split over picked region $\bar{\mathcal{S}}$.

$$k, v = \operatorname*{argmin}_{k,v} f(\bar{\mathcal{S}}, k, v),$$

$$\mathcal{S}_1' = \{i \in \mathcal{S} : x_i[k] < v\}, \quad \mathcal{S}_2' = \{i \in \mathcal{S} : x_i[k] \geq v\}$$



- Update leaves

$$\textbf{leaves} := (\textbf{leaves} \setminus \bar{\mathcal{S}}) \cup \{\mathcal{S}_1', \mathcal{S}_2'\}$$

Better method: maximize information gain

Given a random variable U on support $\mathcal{U} = \{u_1, ..., u_s\}$, the entropy

$$H(U) := -\sum_{u \in \mathcal{U}} \Pr(U = u) \log_2 \Pr(U = u)$$

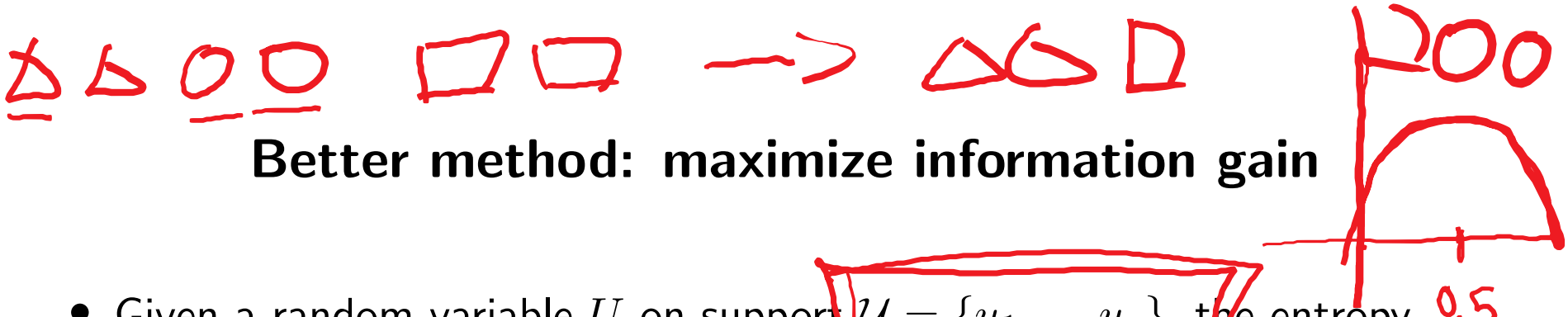
captures the amount of uncertainty in the distribution

Example: As of Nov 30, 2020, election projections *

	New York	Illinois	Florida	California
Biden	60.3%	53%	48%	61.4%
Trump	33.2%	40%	45.9%	31.0%

- What is entropy of election outcome in each state?
- Which state has highest entropy? Which has lowest?
- Assuming all states are equal** , which state's outcome provides me the most information to the national outcome?

*from fivethirtyeight.com, **untrue in practice



Better method: maximize information gain

- Given a random variable U on support $\mathcal{U} = \{u_1, ..., u_s\}$, the entropy

$$H(U) := -\sum_{u \in \mathcal{U}} \Pr(U = u) \log_2 \Pr(U = u)$$

captures the amount of uncertainty in the distribution

- Now given some revealing information V over $\mathcal{V}$, the conditional entropy

$$H(U|V) := -\sum_{u \in \mathcal{U}, v \in \mathcal{V}} \Pr(U = u, V = v) \log_2 \Pr(U = u|V = v)$$

- The information gain from revealing V is

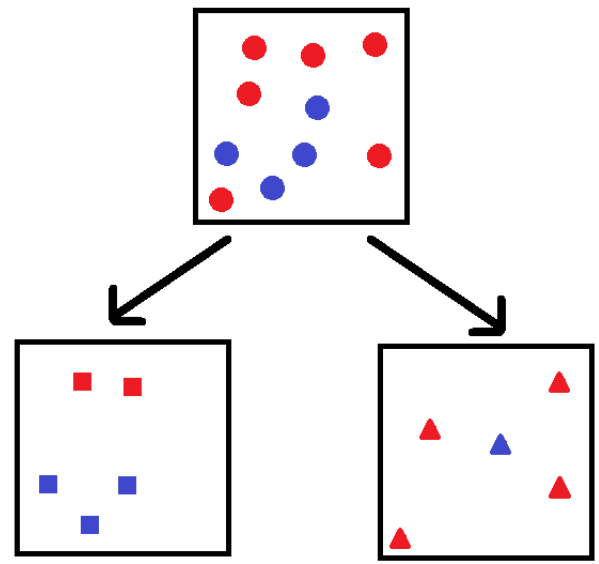
$$IG(V) := H(U) - H(U|V)$$

Mutual info (U; V)

A better measure of split quality: Information gain

- Split $\mathcal{S} \rightarrow \mathcal{S}_L, \mathcal{S}_R$

$$
\begin{aligned}
H(\mathcal{S}) &= \text{entropy of labels } y_i \text{ over } i \in \mathcal{S} \\
H(\mathcal{S}|\text{split}) &= \frac{|\mathcal{S}_L|}{|\mathcal{S}|} H(\mathcal{S}_L) + \frac{|\mathcal{S}_R|}{|\mathcal{S}|} H(\mathcal{S}_R)
\end{aligned}
$$



- Pick split that maximizes

$$IG(\text{split}) = H(\mathcal{S}) - H(\mathcal{S}|\text{split})$$

Stopping condition

Stop at...

- depth$> D$
- max size of leaf $|\bar{S}| < \epsilon$
- max loss per region $h(\bar{S}) < \epsilon$

What if...

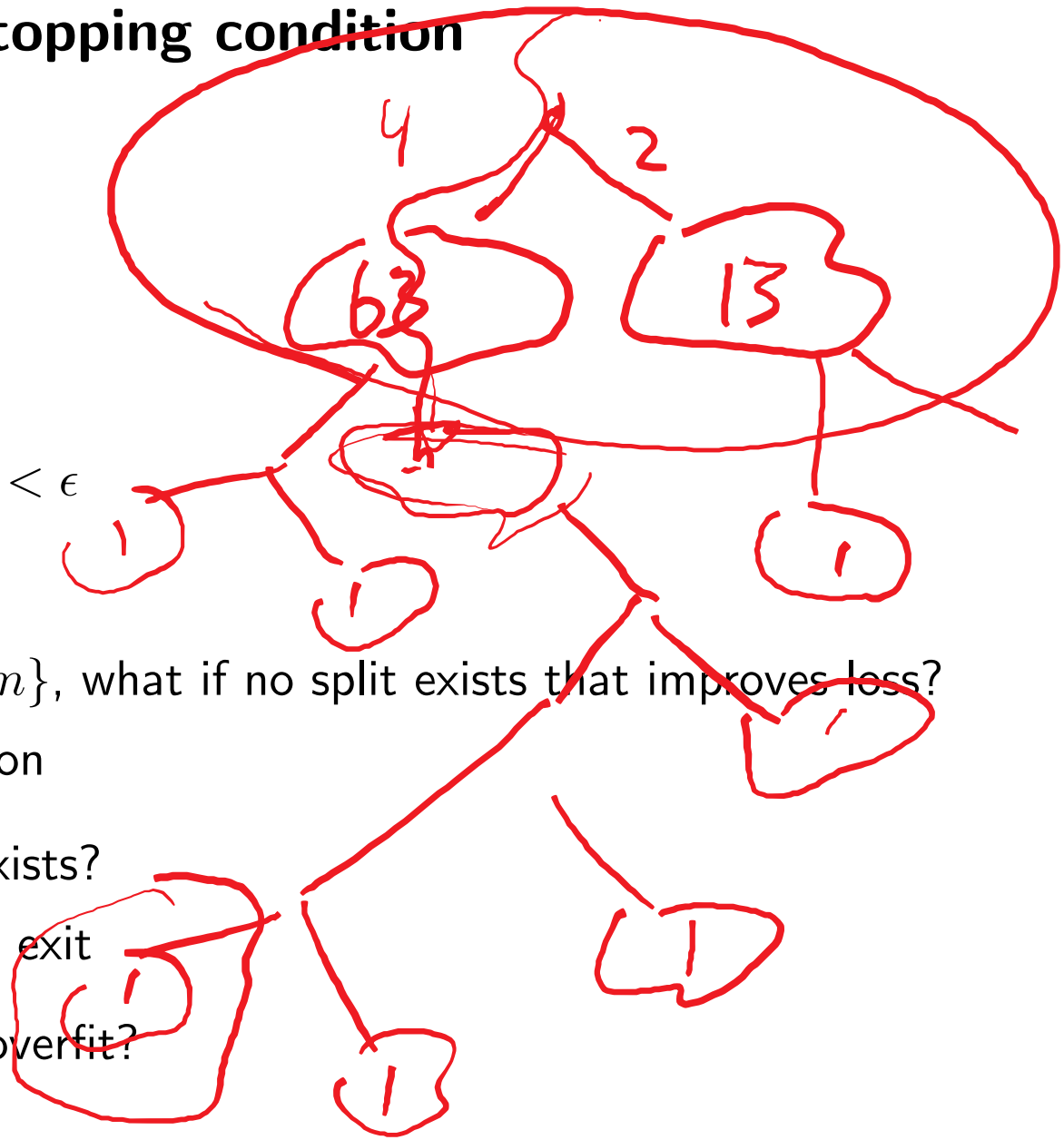
- Given region $\mathcal{S} \subset \{1, ..., m\}$, what if no split exists that improves loss?

  Ans: go to next best region

- What if no such region exists?

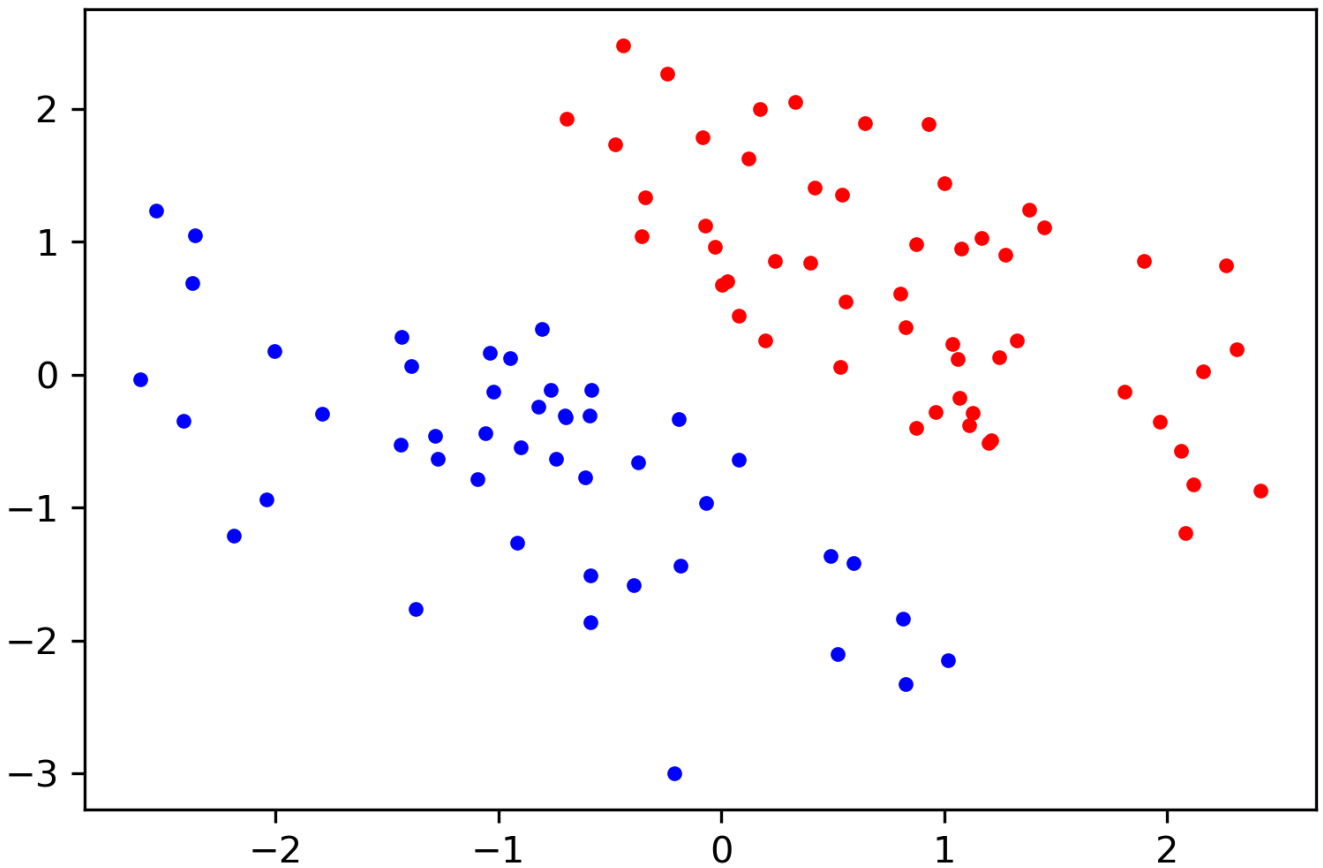
  Ans: nothing more to do, exit

Question: Can decision trees overfit?

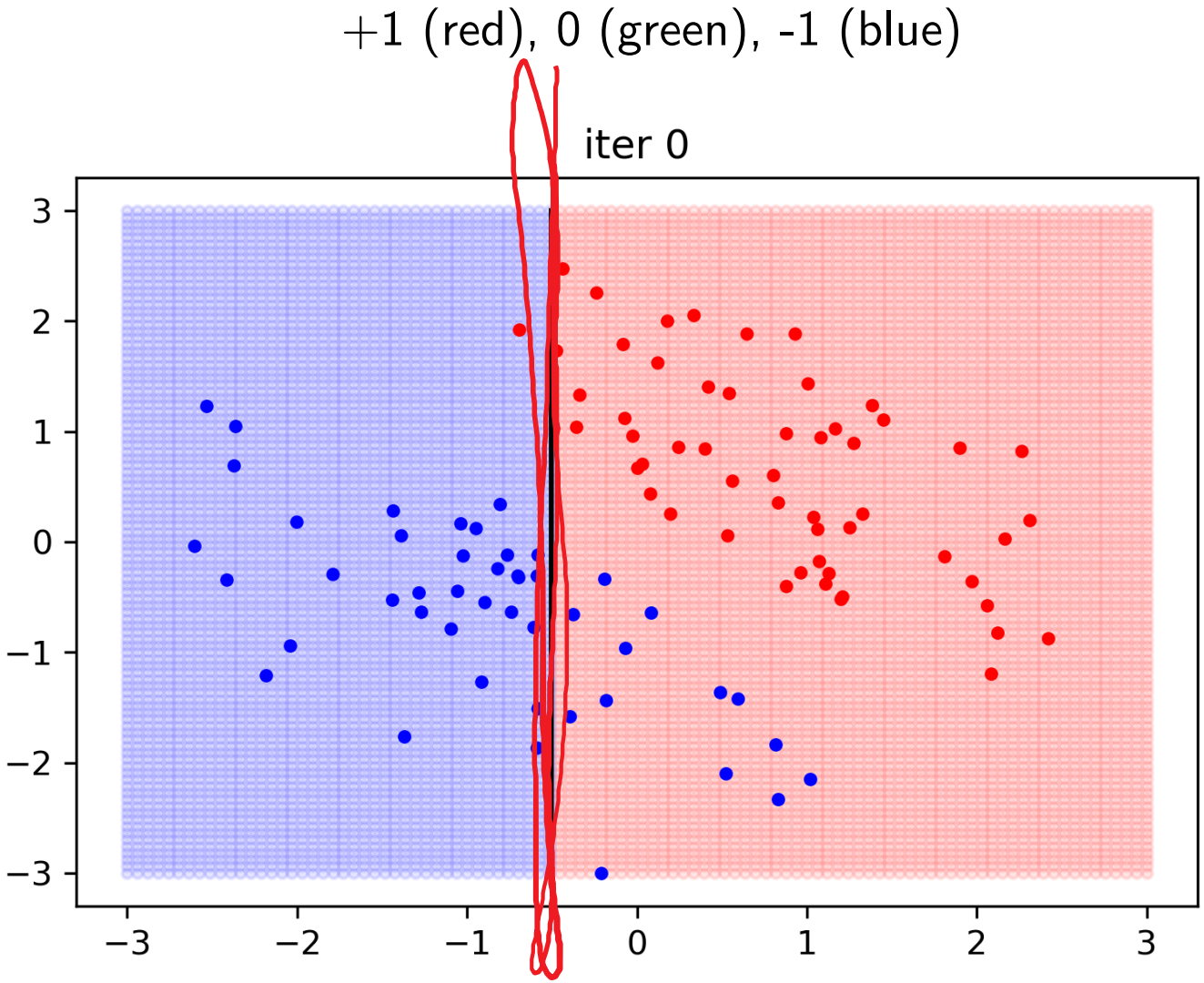


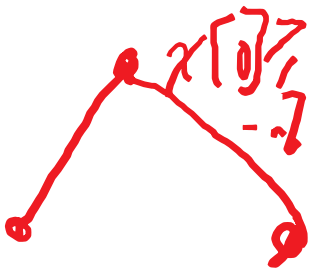
Example: good separation

+1 (red), 0 (green), -1 (blue)



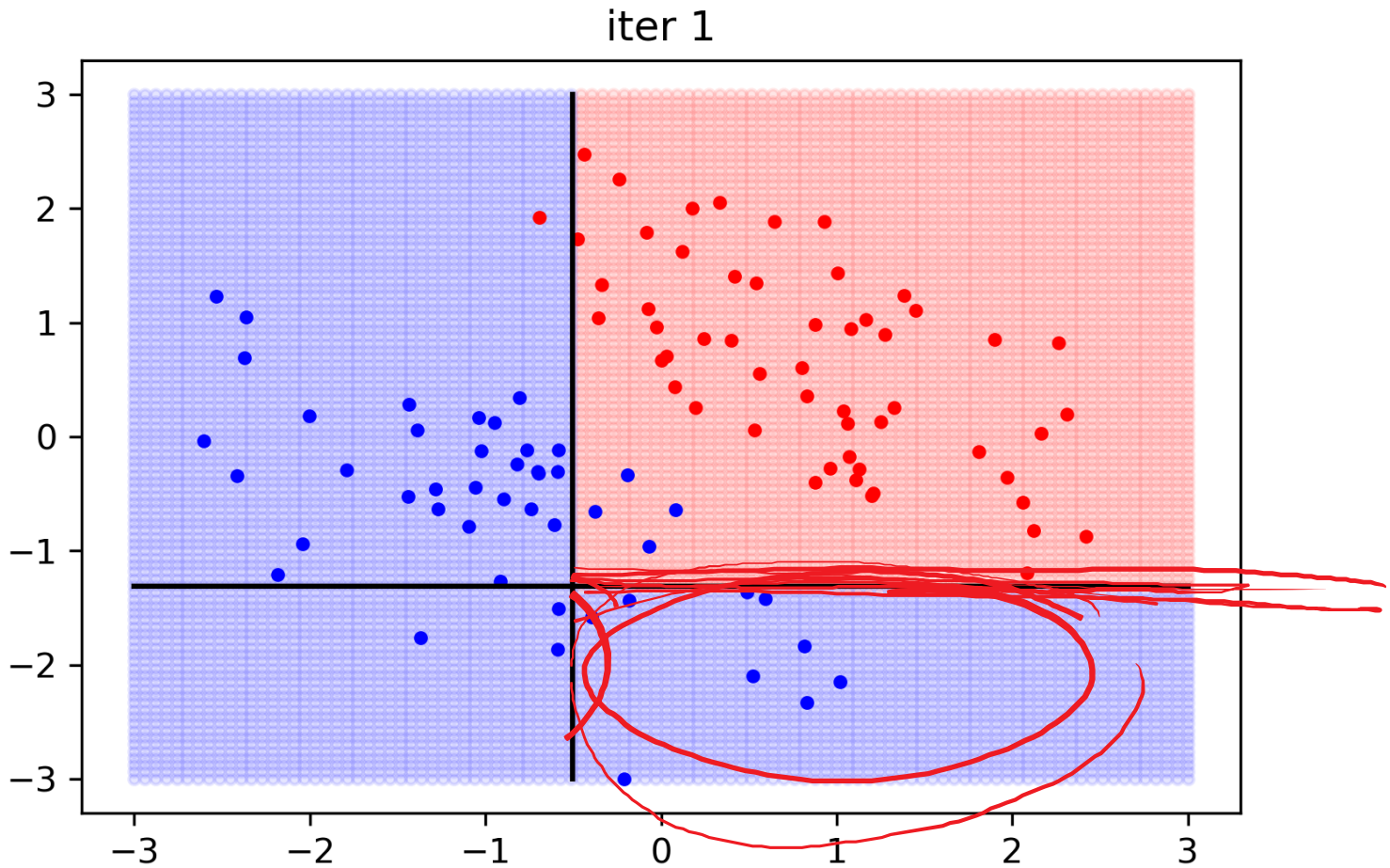
Example: good separation





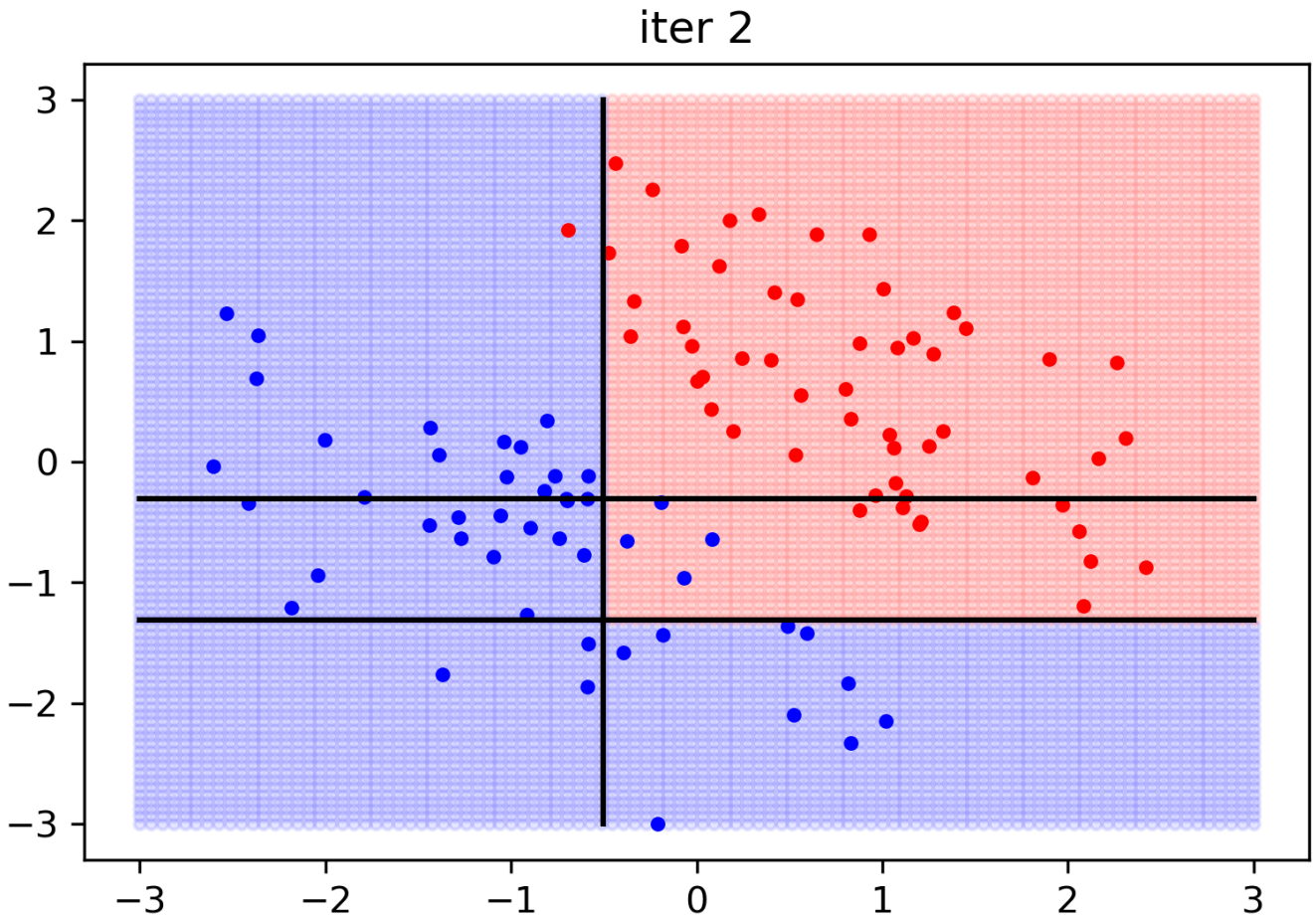
Example: good separation

+1 (red), 0 (green), -1 (blue)



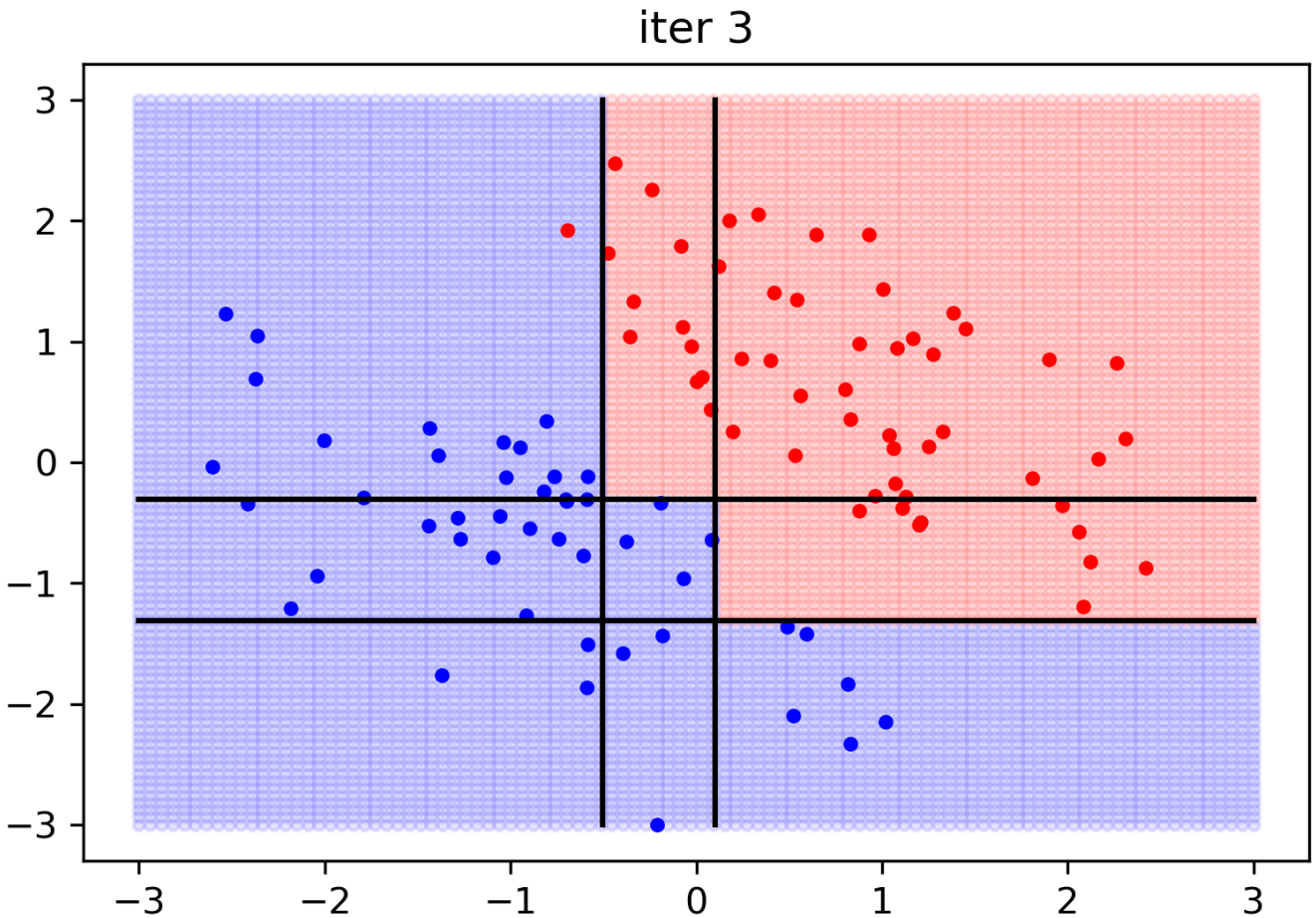
Example: good separation

+1 (red), 0 (green), -1 (blue)



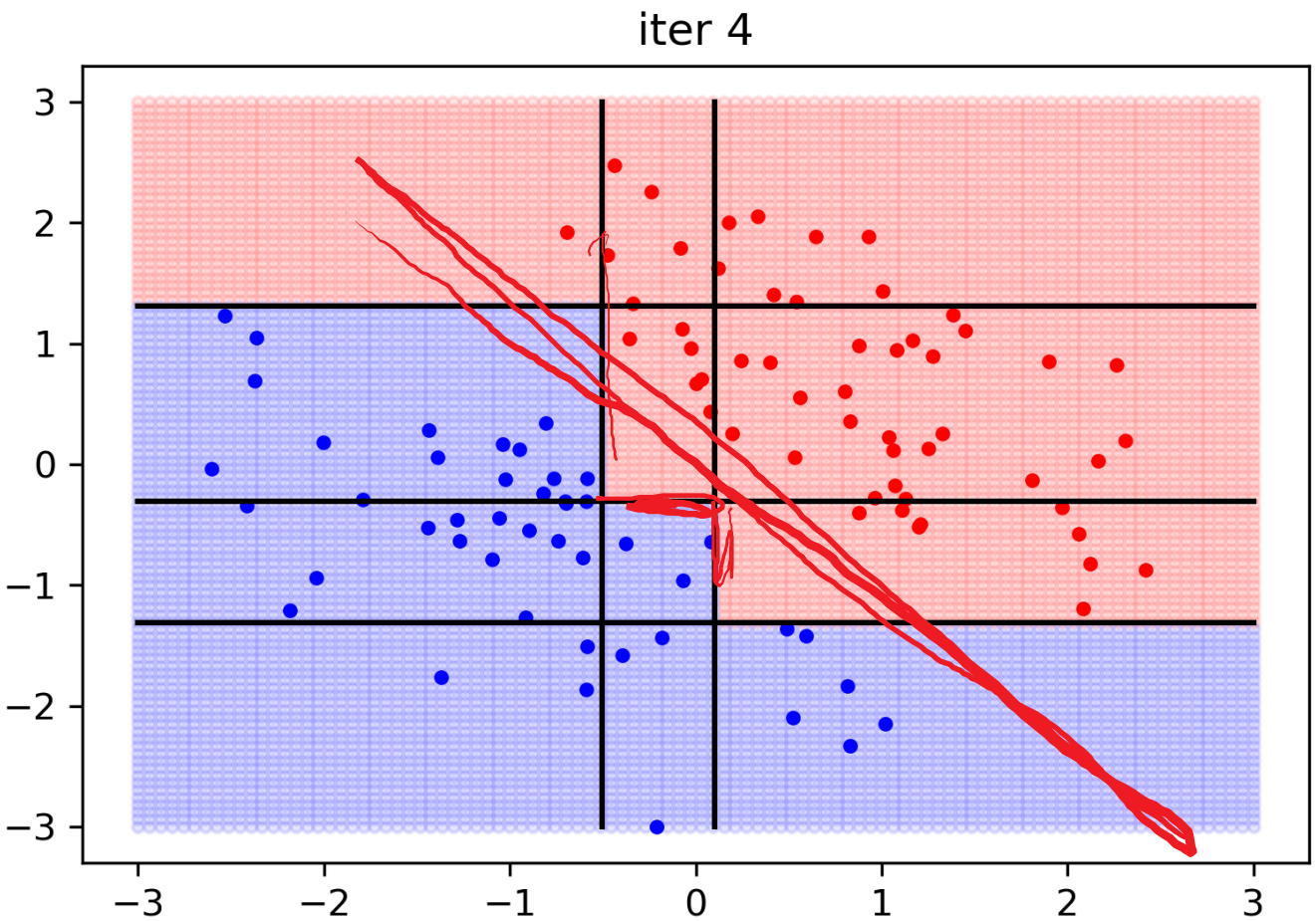
Example: good separation

+1 (red), 0 (green), -1 (blue)



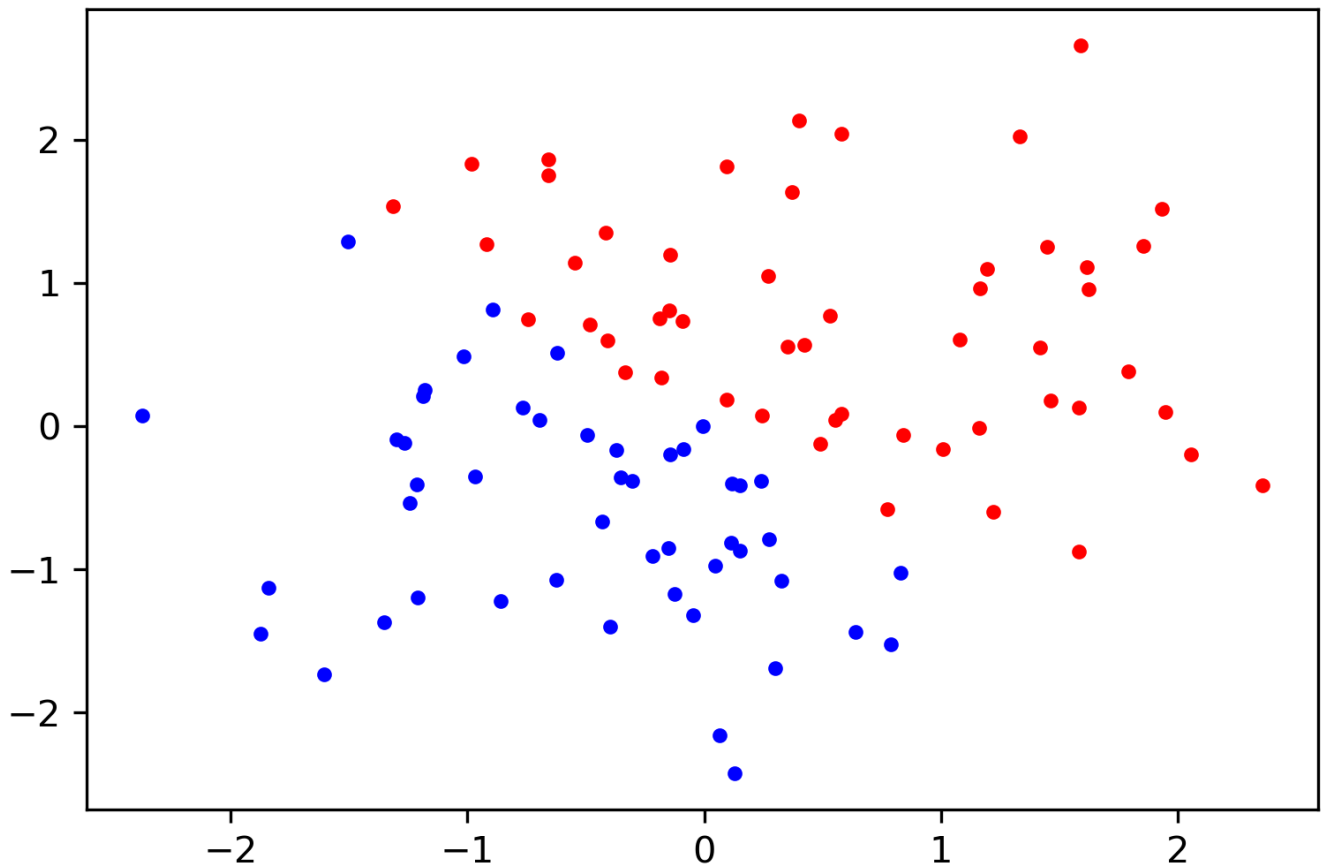
Example: good separation

+1 (red), 0 (green), -1 (blue)



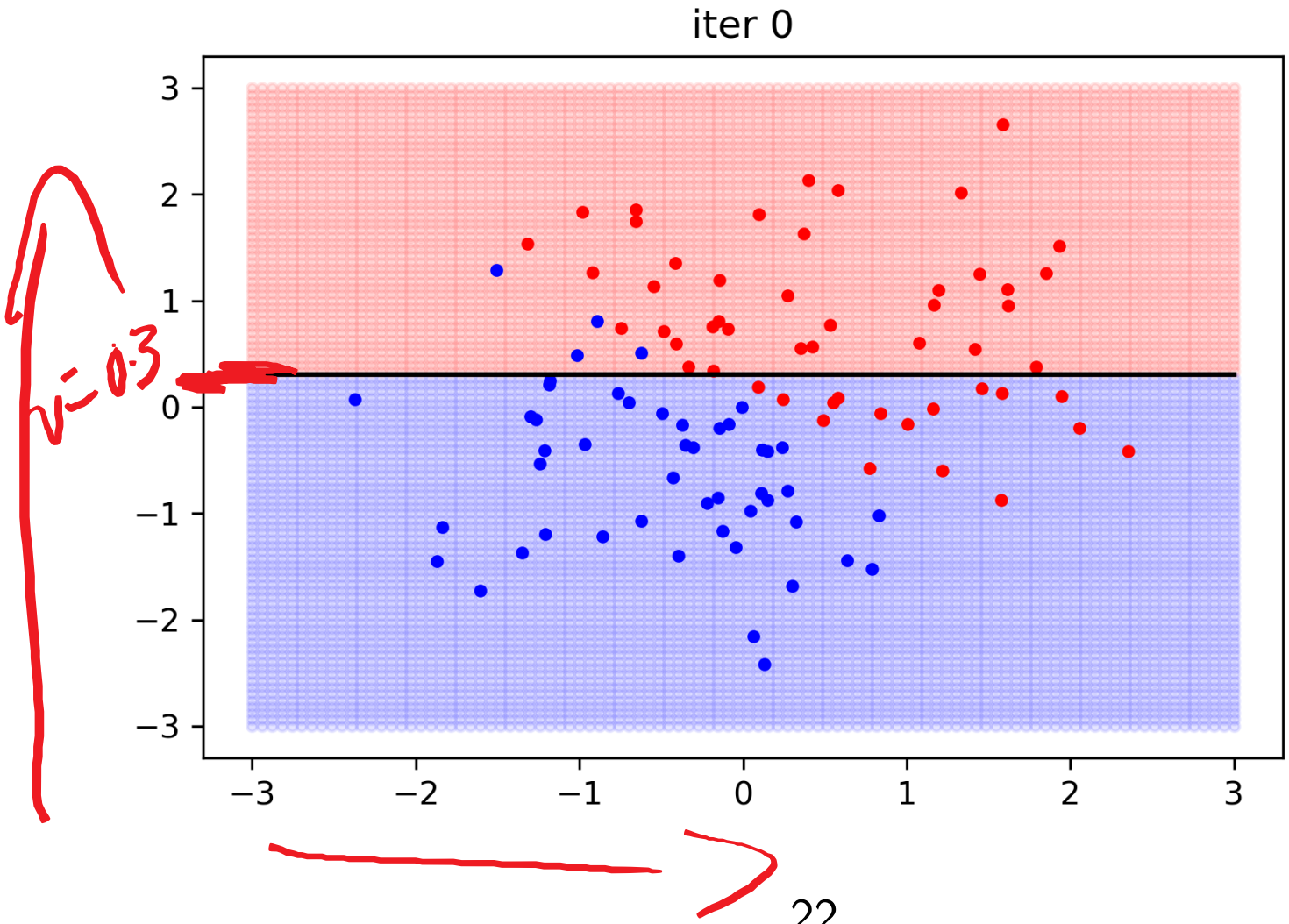
Example: tight separation

+1 (red), 0 (green), -1 (blue)



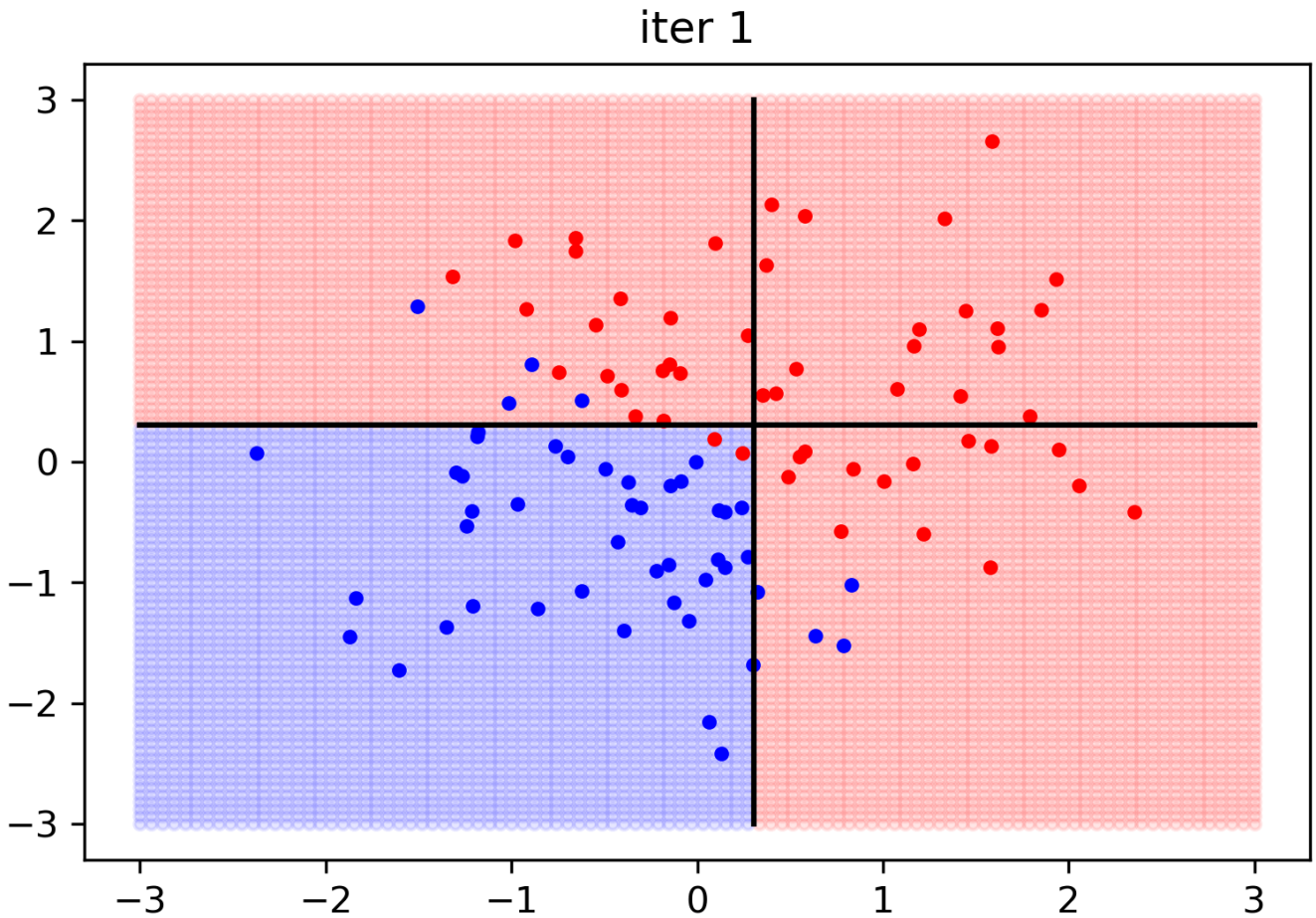
Example: tight separation

+1 (red), 0 (green), -1 (blue)



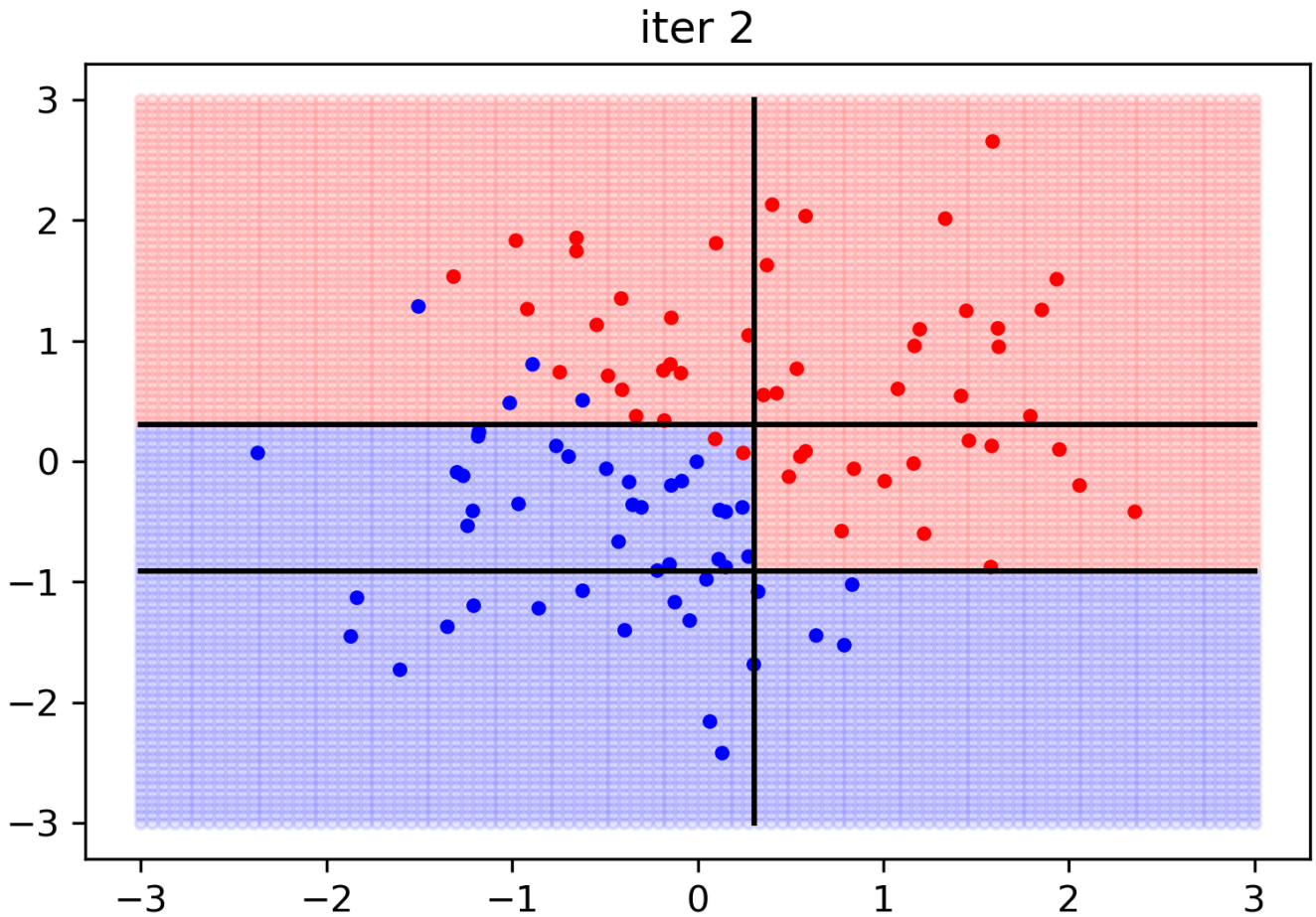
Example: tight separation

+1 (red), 0 (green), -1 (blue)



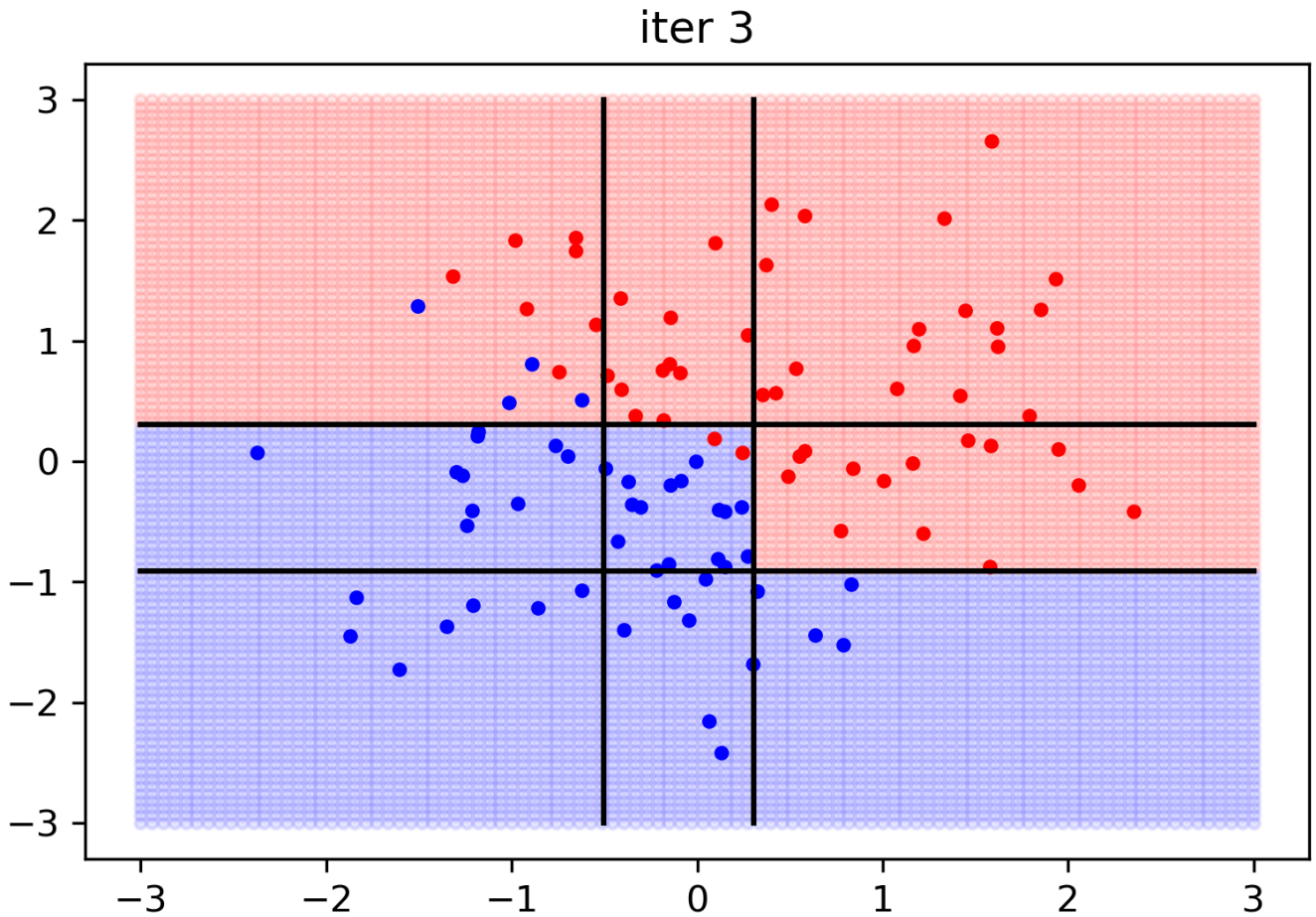
Example: tight separation

+1 (red), 0 (green), -1 (blue)



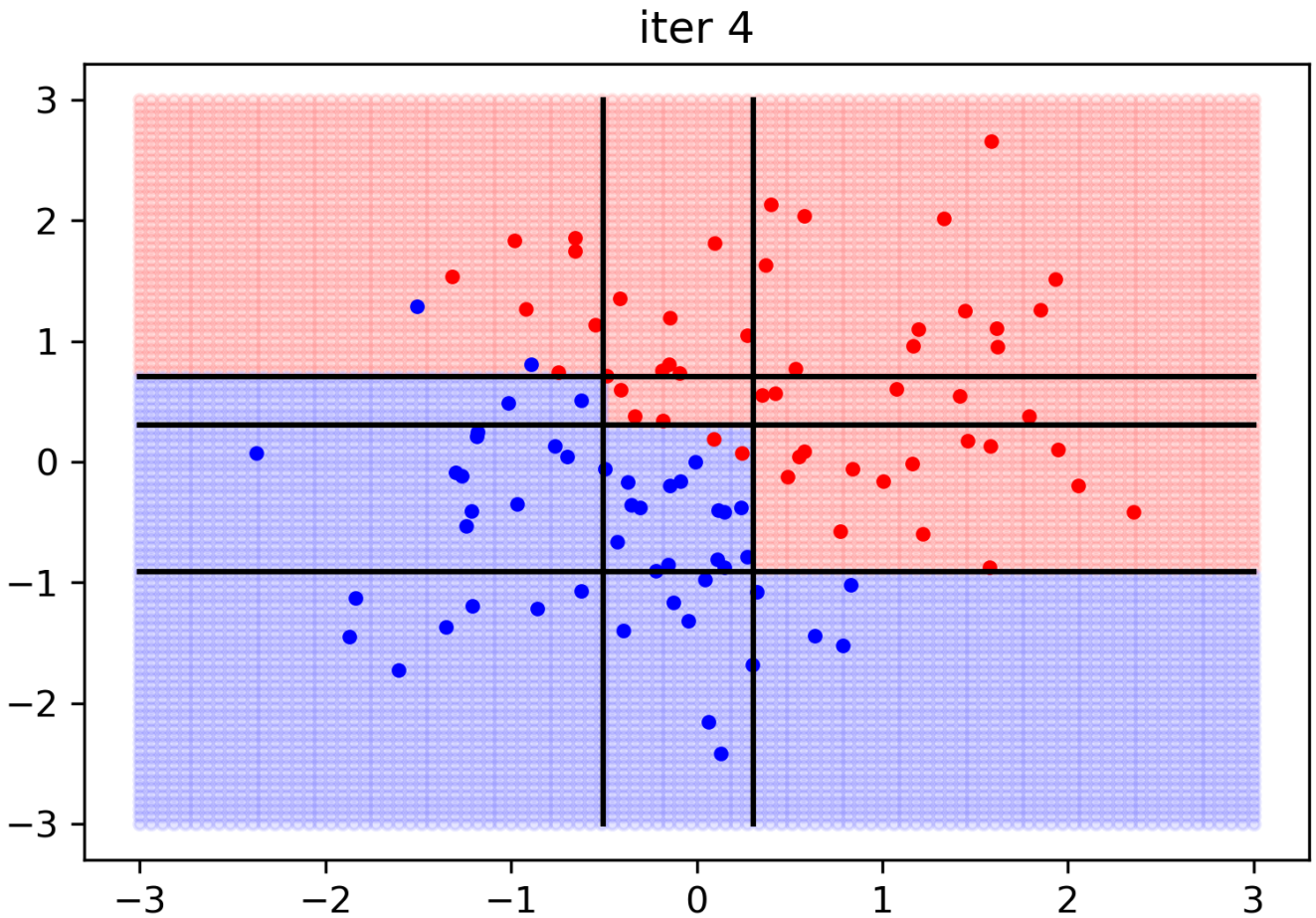
Example: tight separation

+1 (red), 0 (green), -1 (blue)



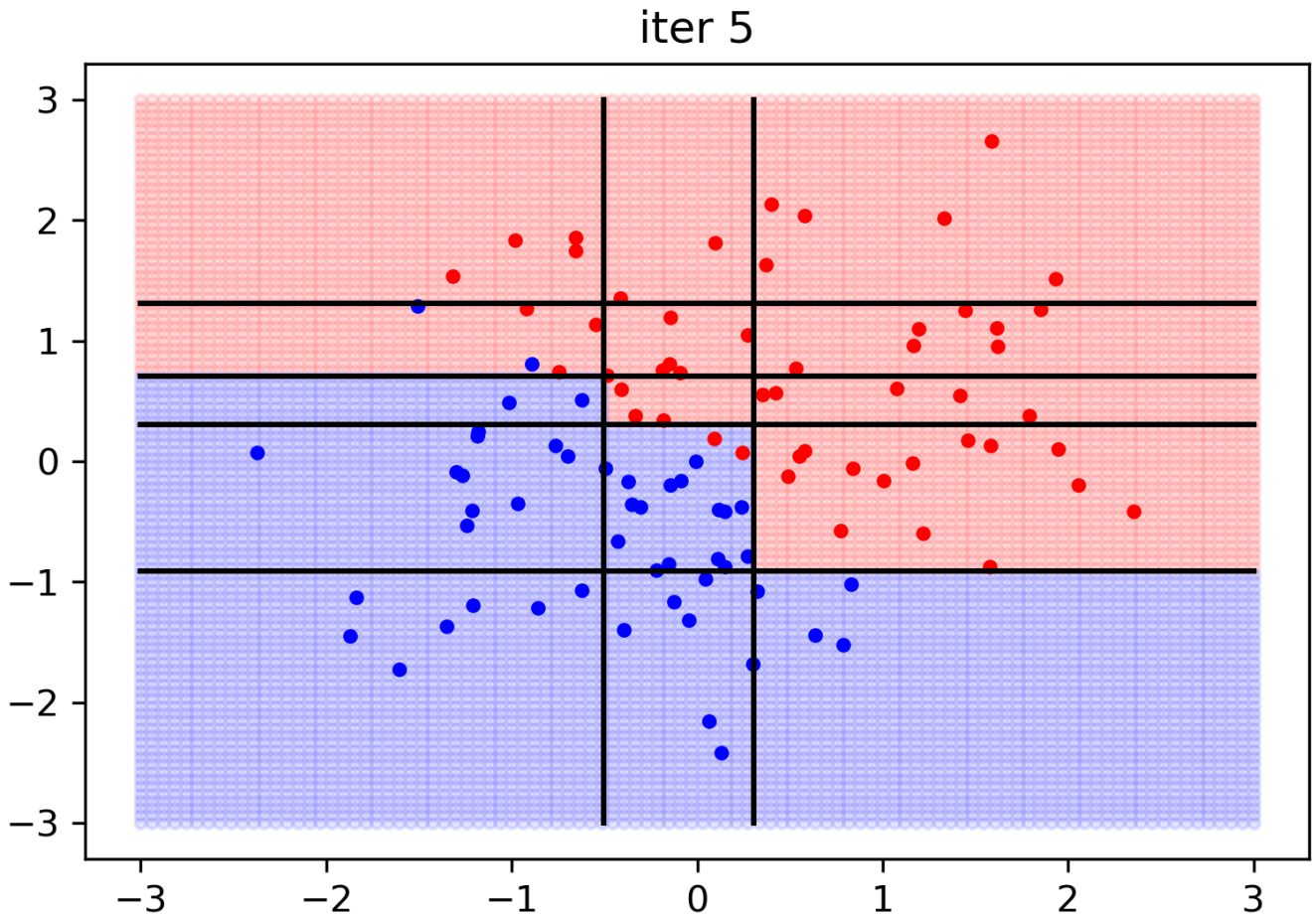
Example: tight separation

+1 (red), 0 (green), -1 (blue)



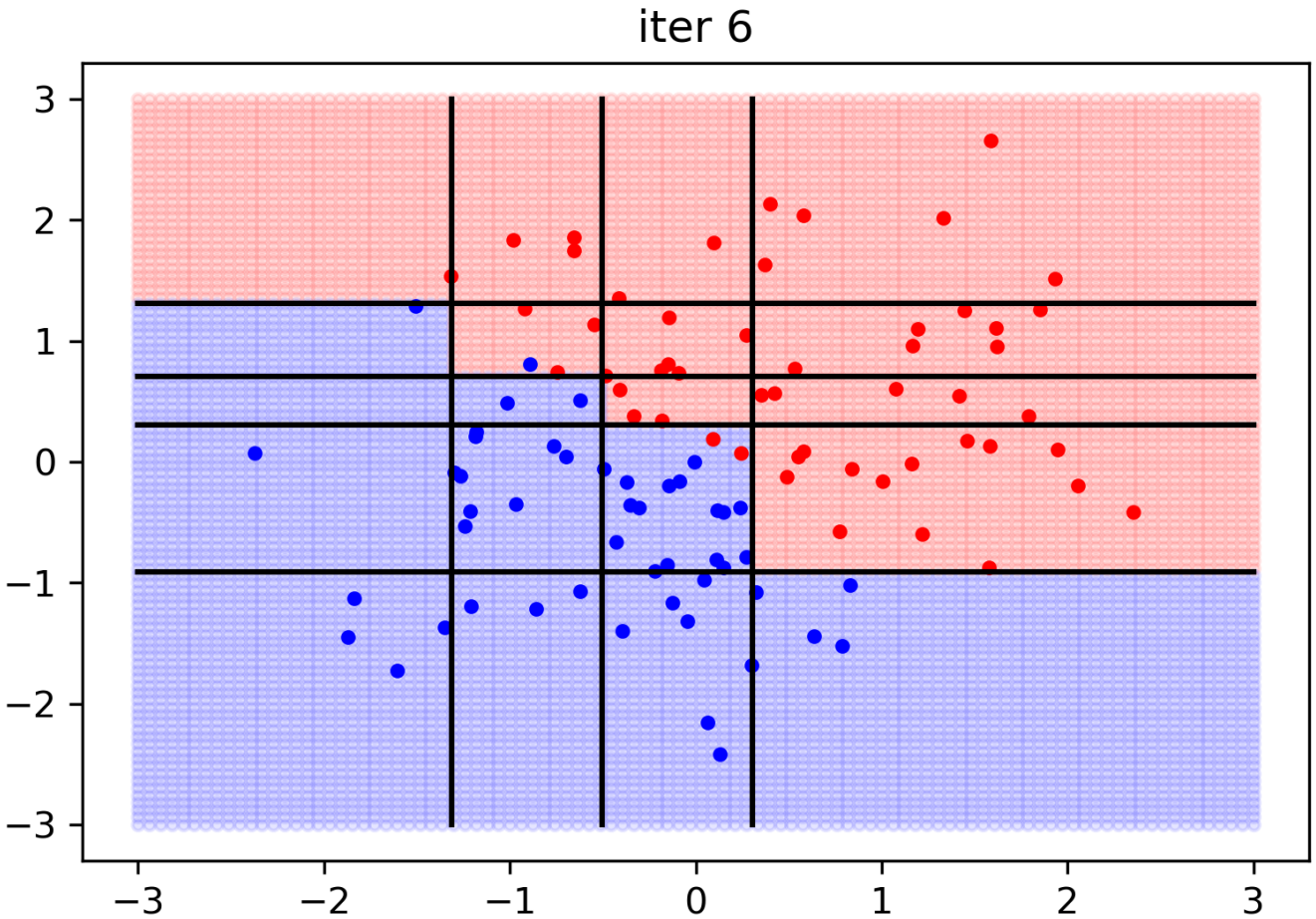
Example: tight separation

+1 (red), 0 (green), -1 (blue)



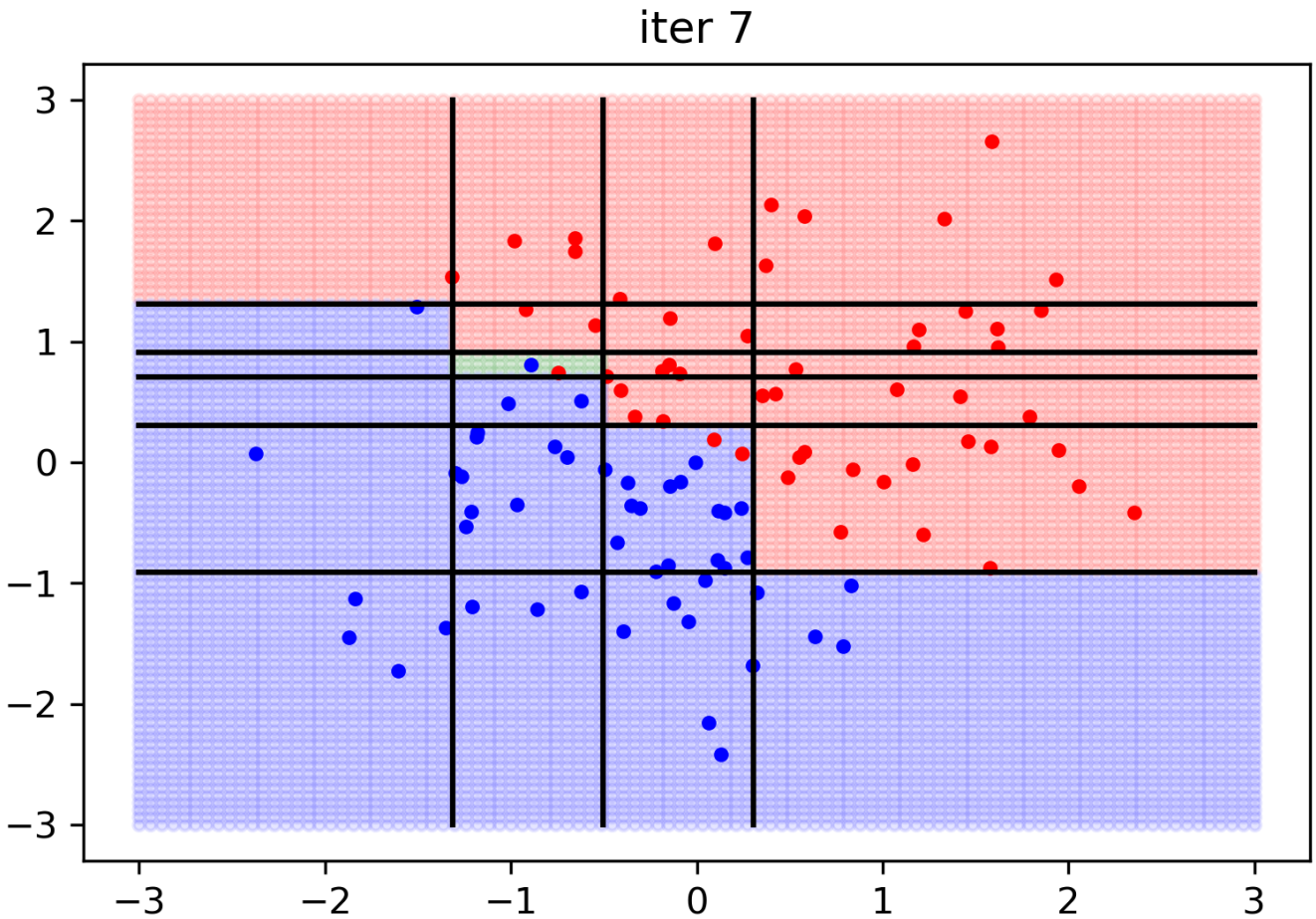
Example: tight separation

+1 (red), 0 (green), -1 (blue)



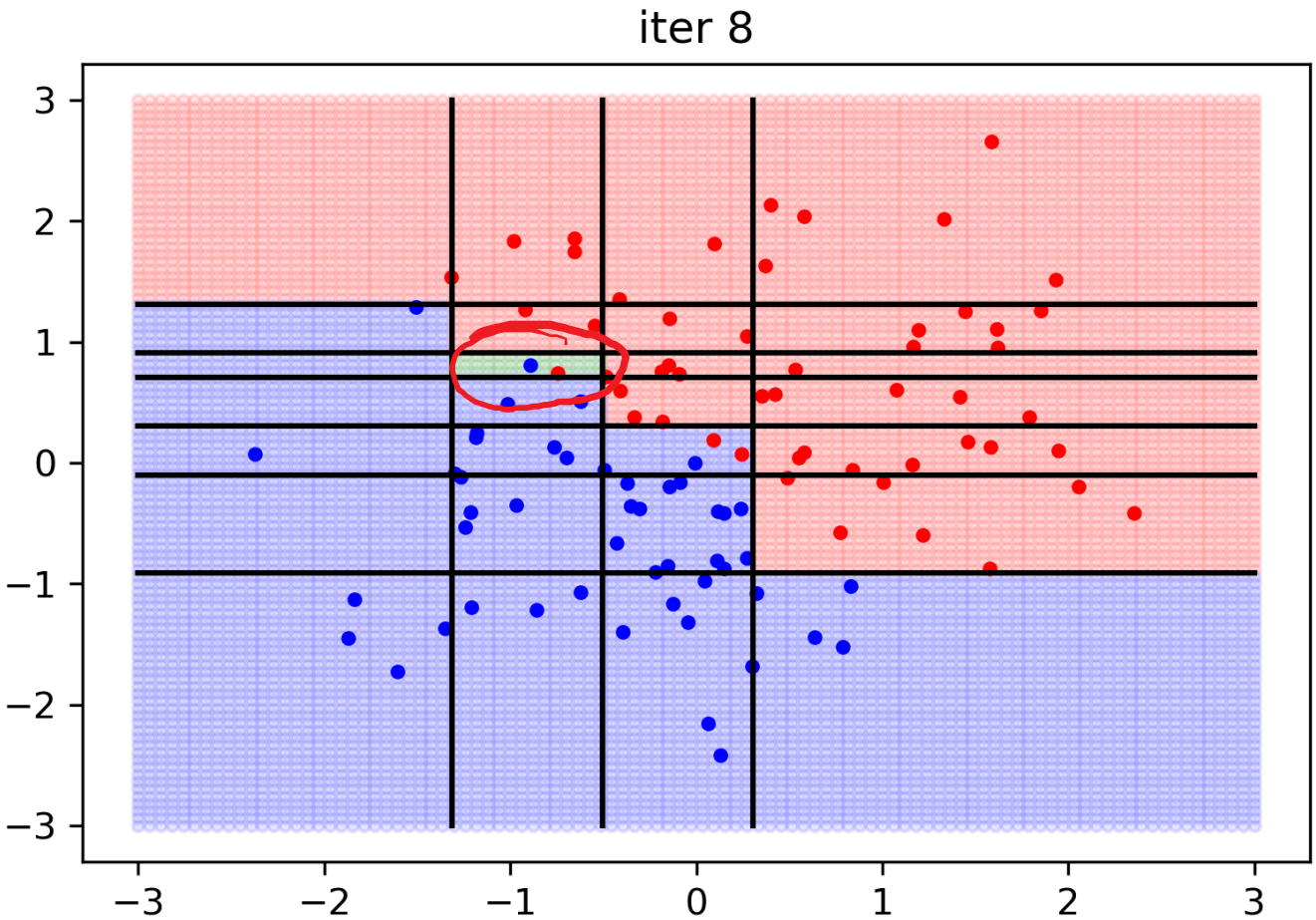
Example: tight separation

+1 (red), 0 (green), -1 (blue)



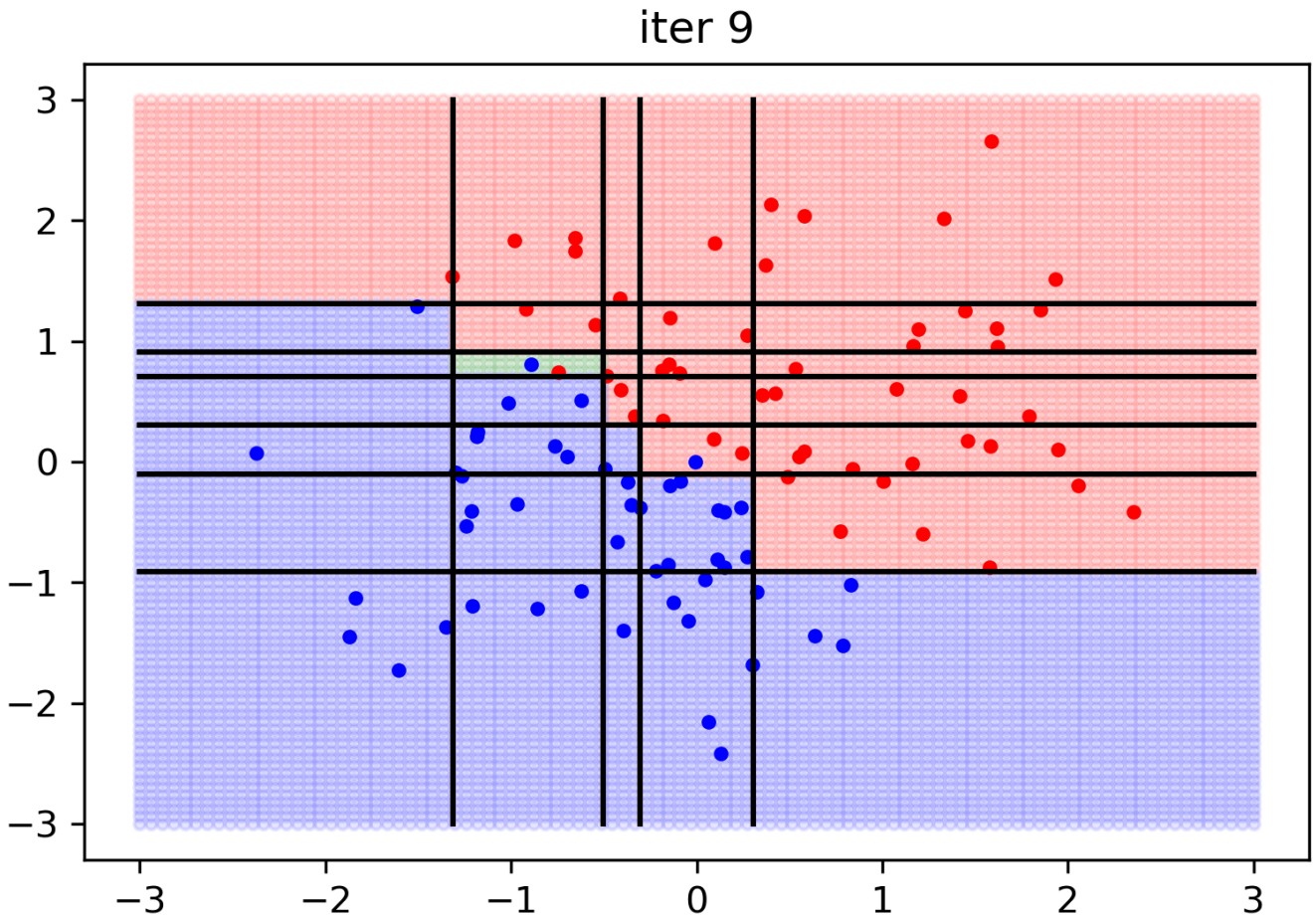
Example: tight separation

+1 (red), 0 (green), -1 (blue)



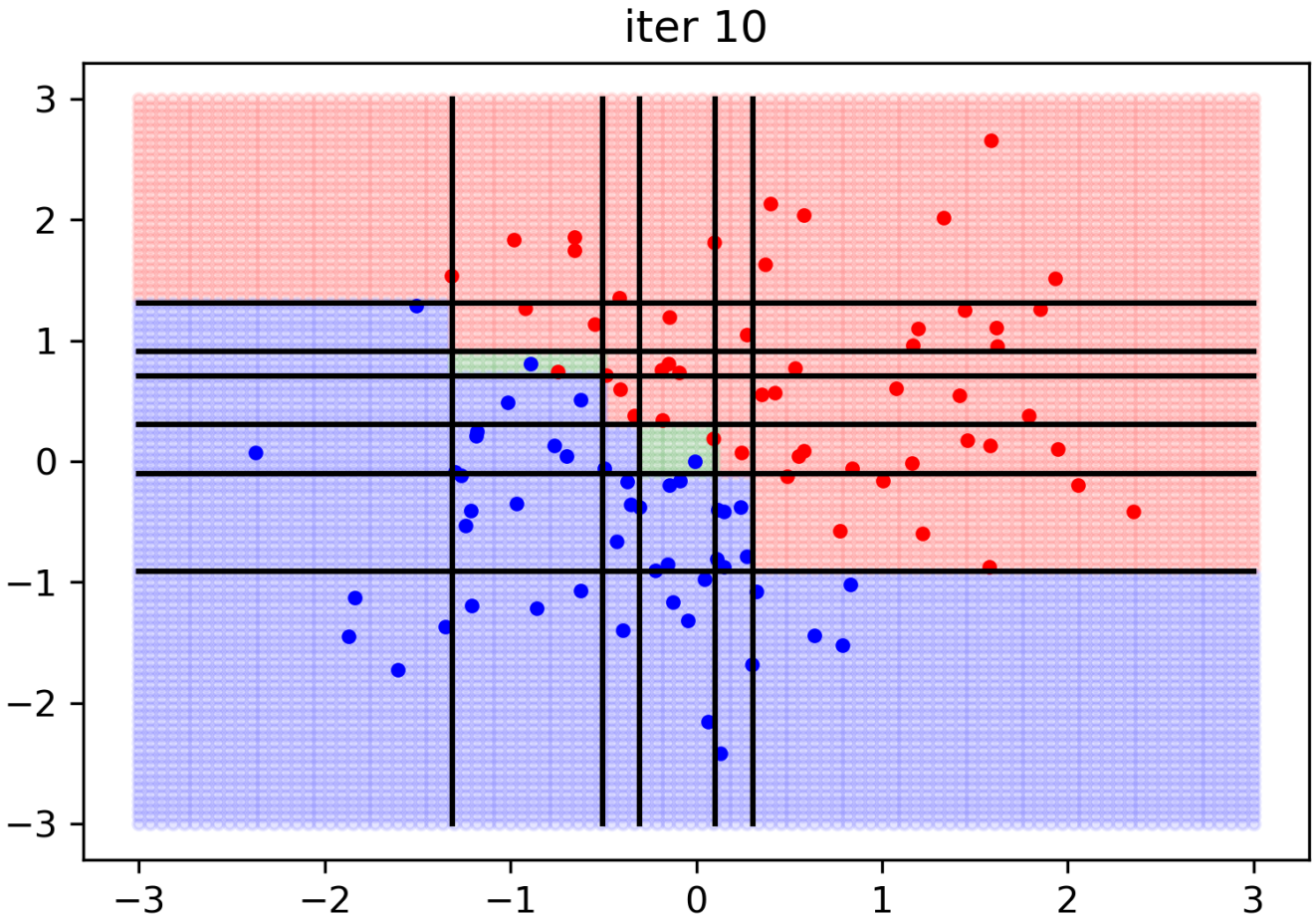
Example: tight separation

+1 (red), 0 (green), -1 (blue)



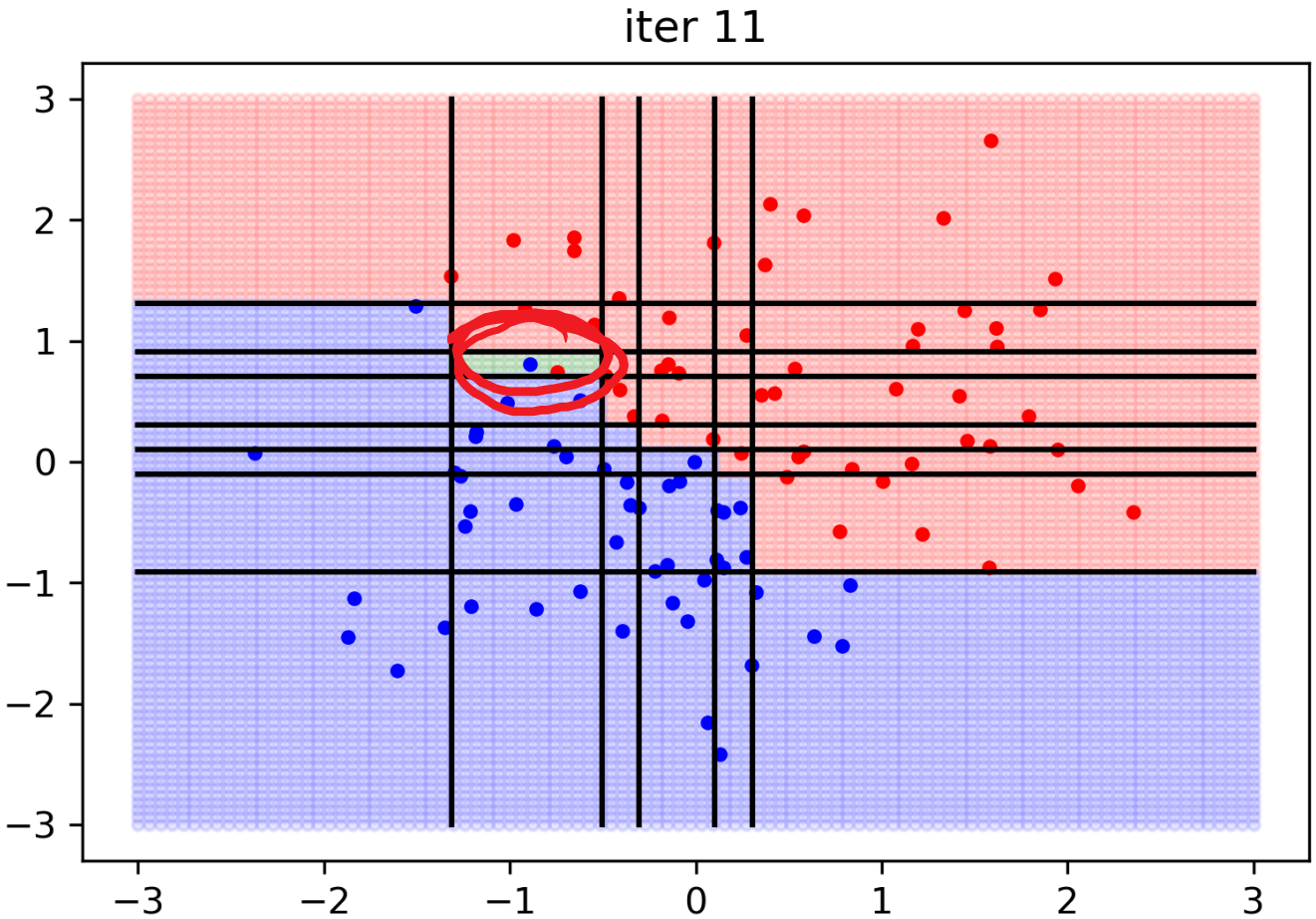
Example: tight separation

+1 (red), 0 (green), -1 (blue)



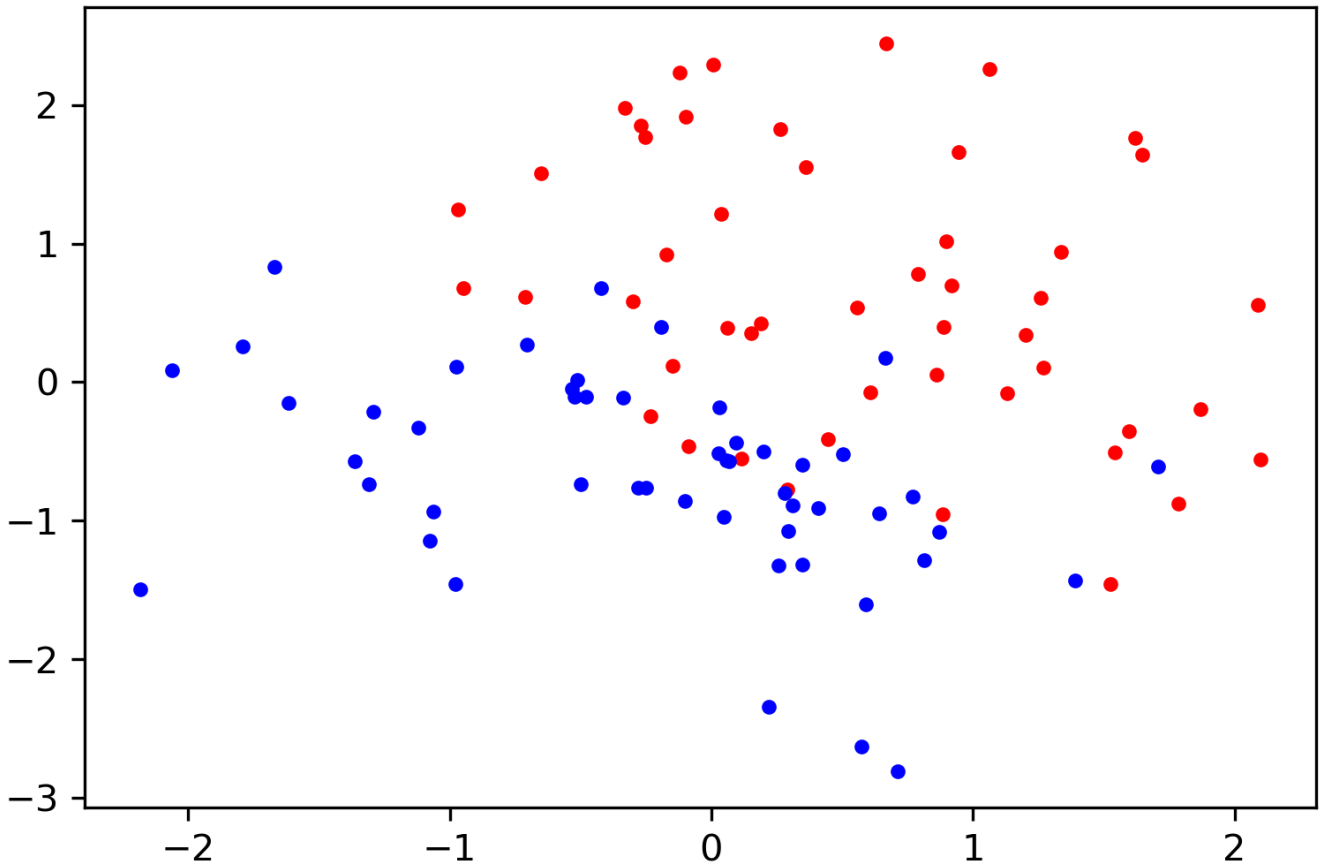
Example: tight separation

+1 (red), 0 (green), -1 (blue)



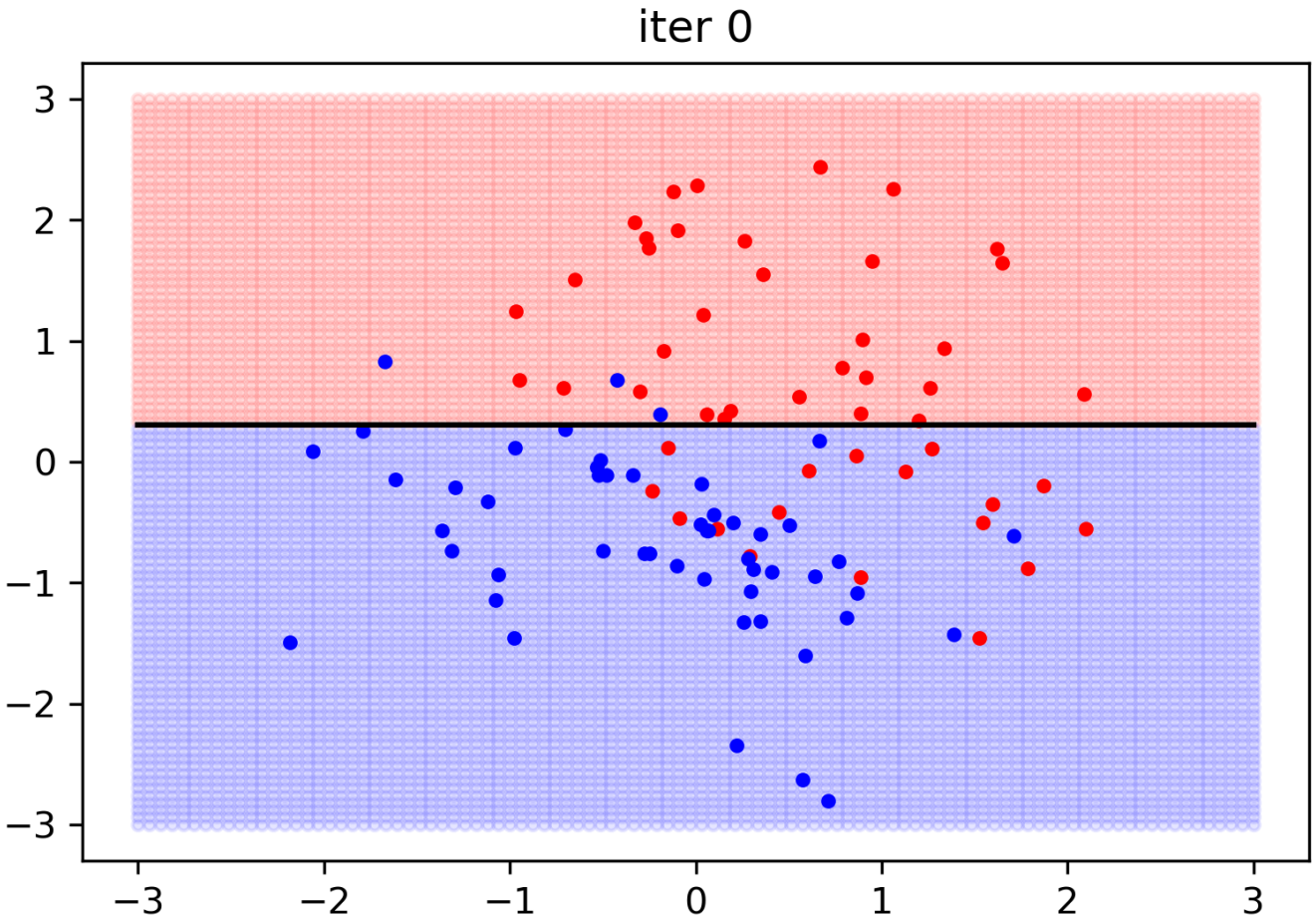
Example: bad separation

+1 (red), 0 (green), -1 (blue)



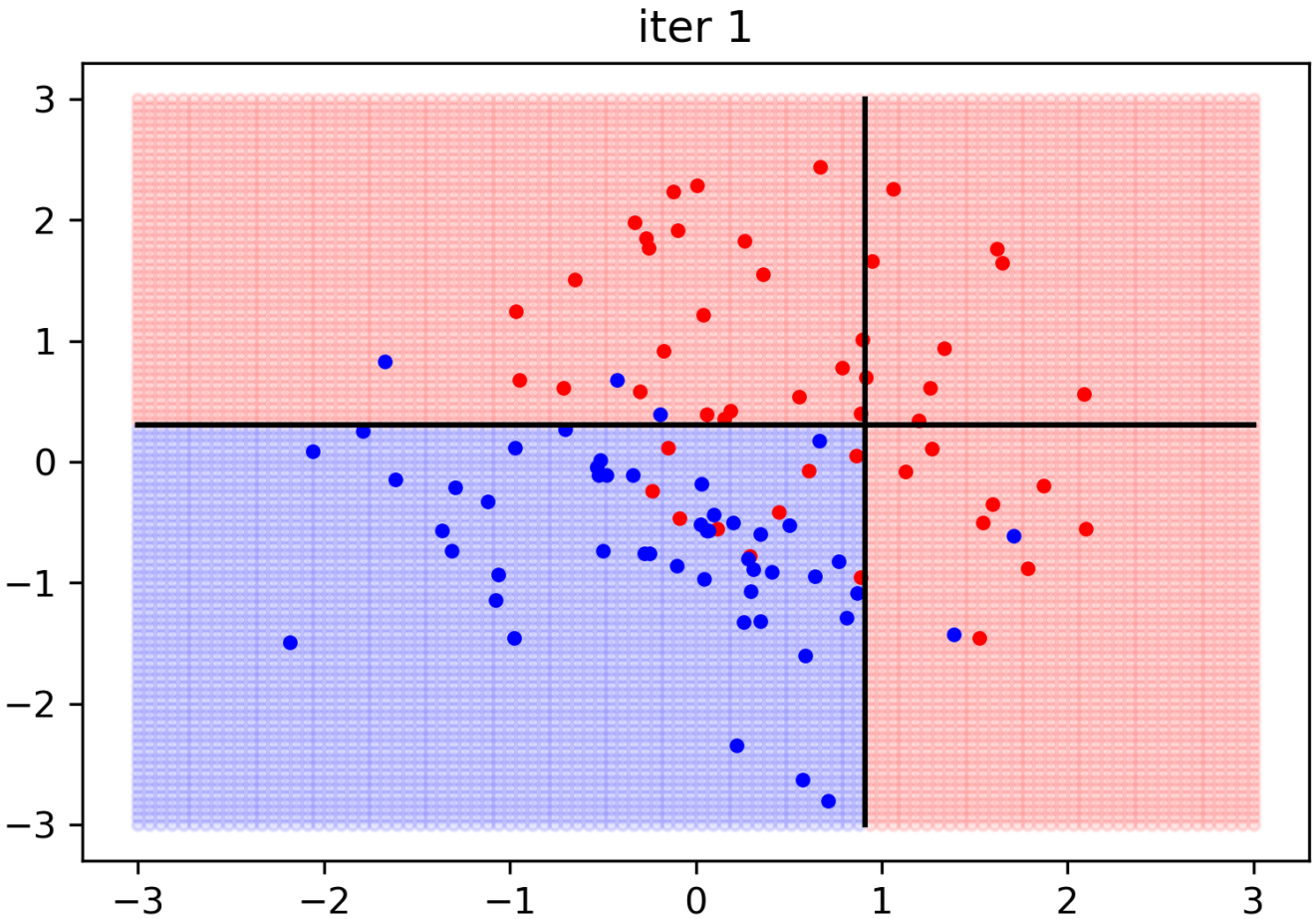
Example: bad separation

+1 (red), 0 (green), -1 (blue)



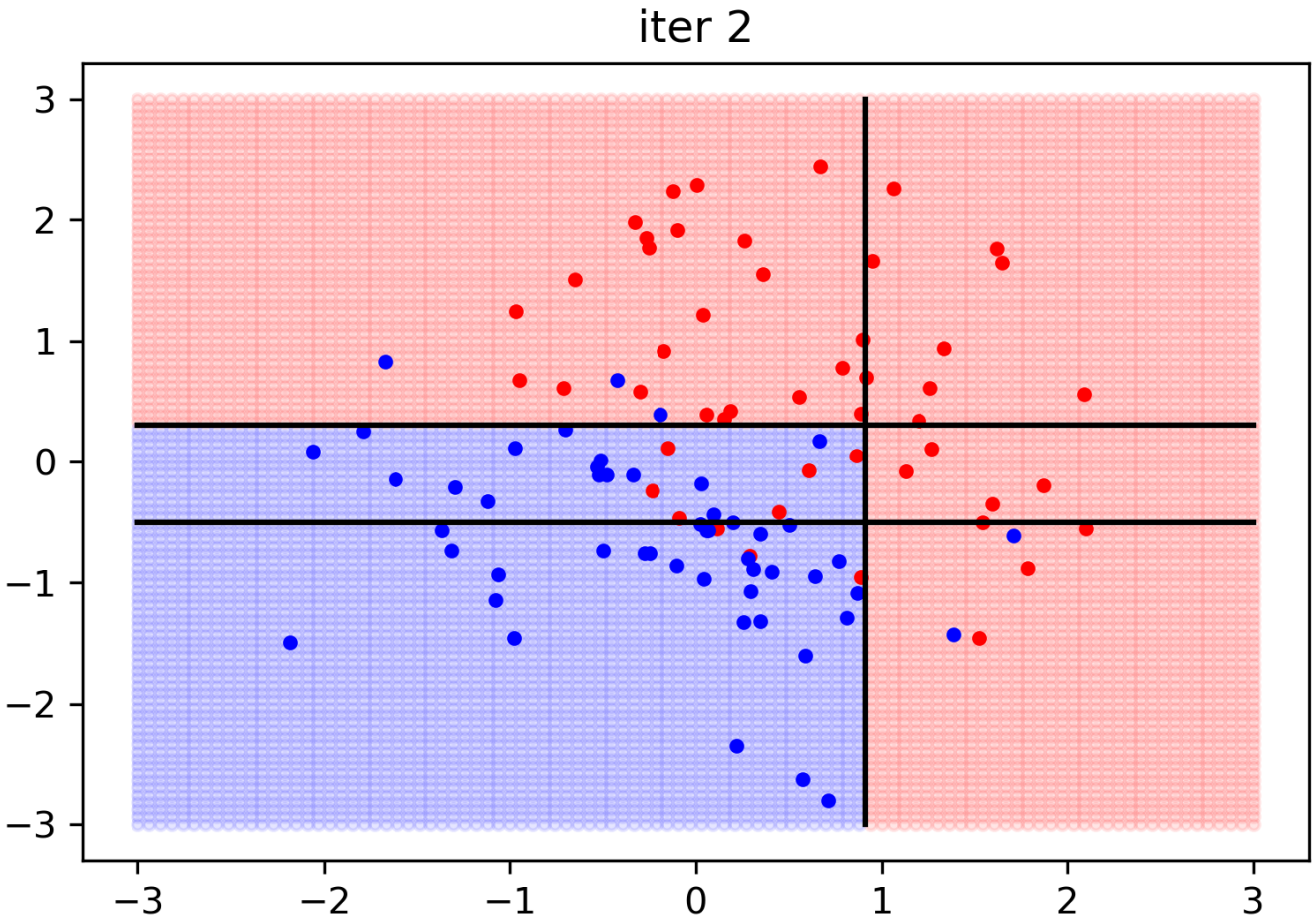
Example: bad separation

+1 (red), 0 (green), -1 (blue)



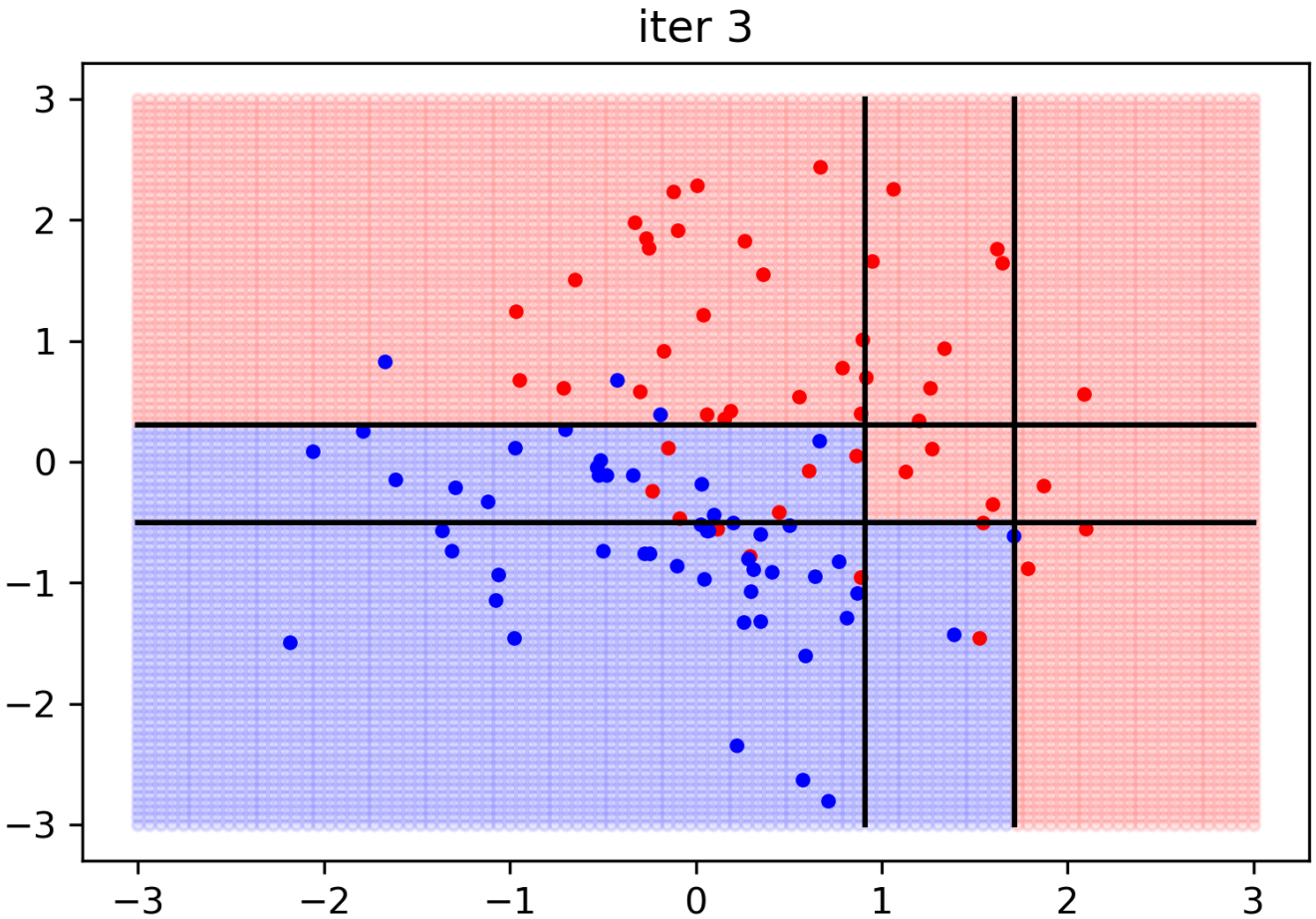
Example: bad separation

+1 (red), 0 (green), -1 (blue)



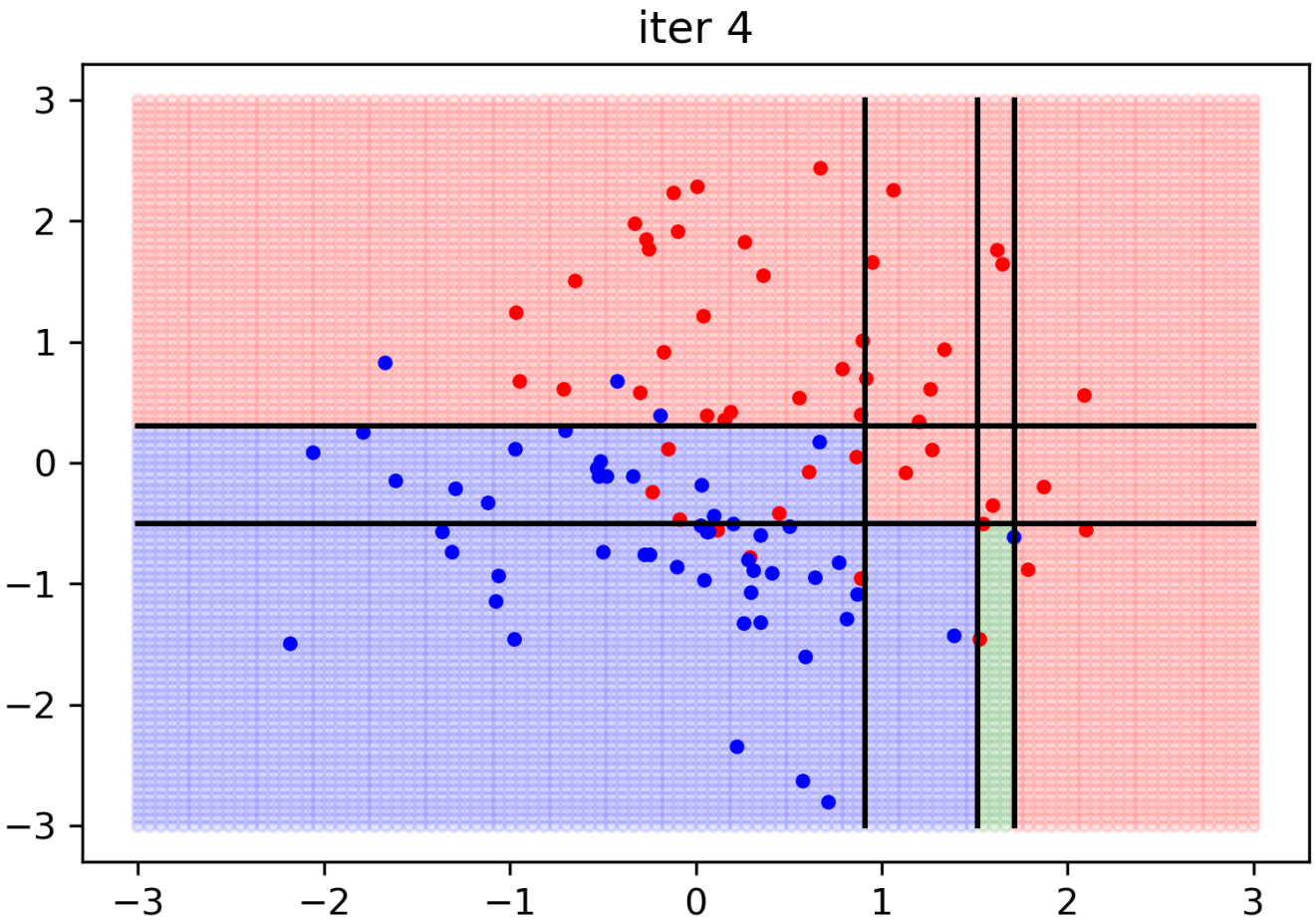
Example: bad separation

+1 (red), 0 (green), -1 (blue)



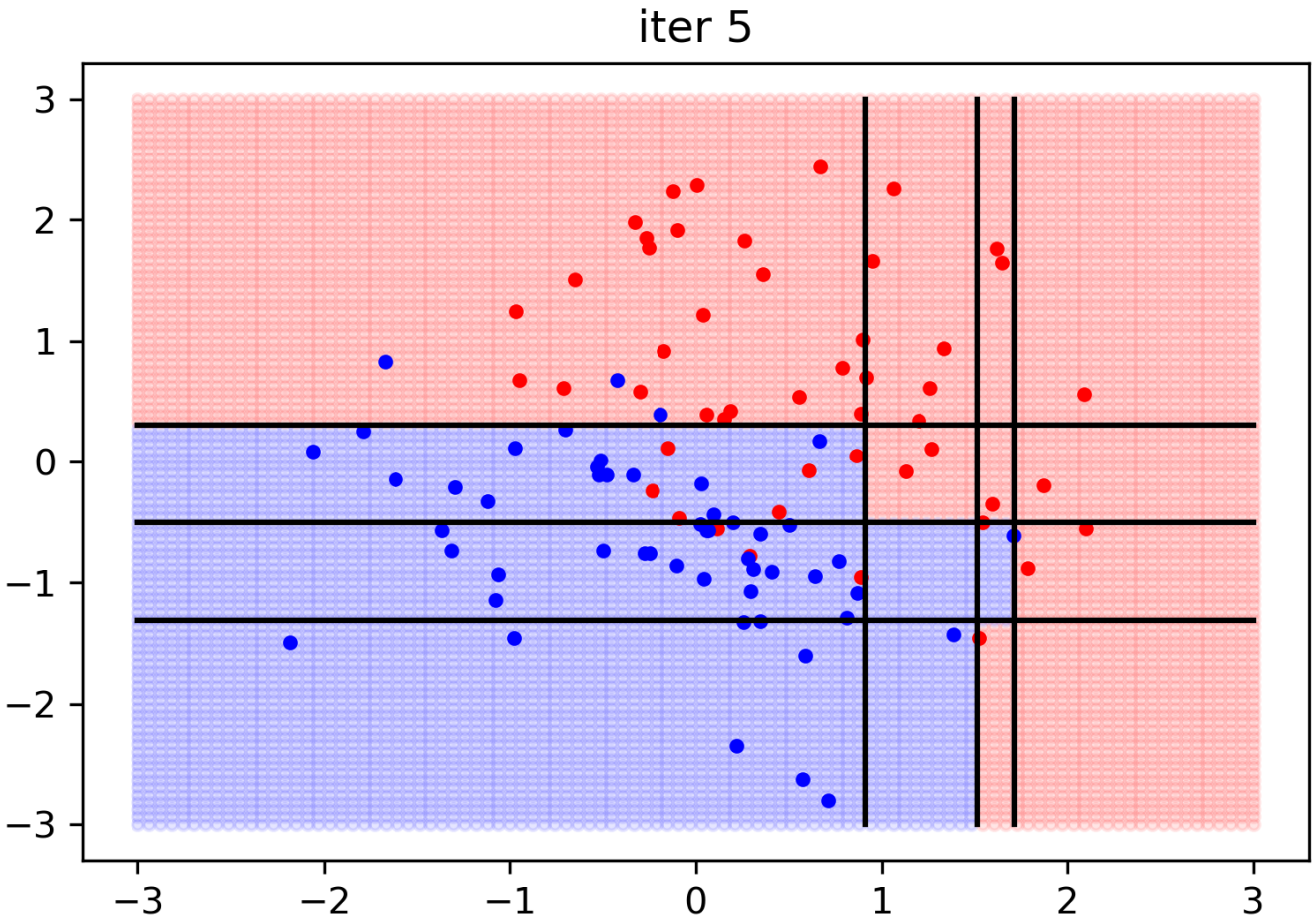
Example: bad separation

+1 (red), 0 (green), -1 (blue)



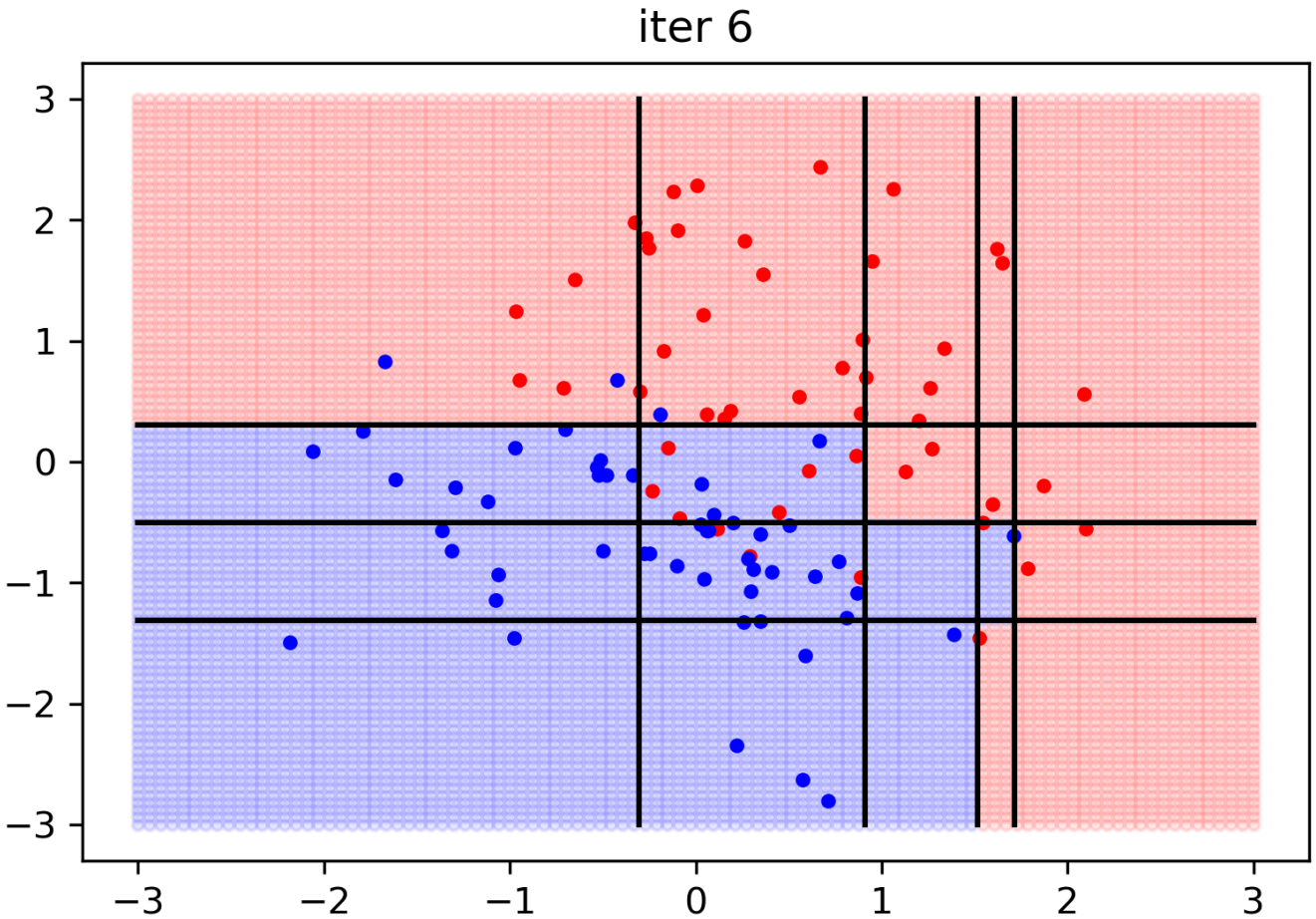
Example: bad separation

+1 (red), 0 (green), -1 (blue)



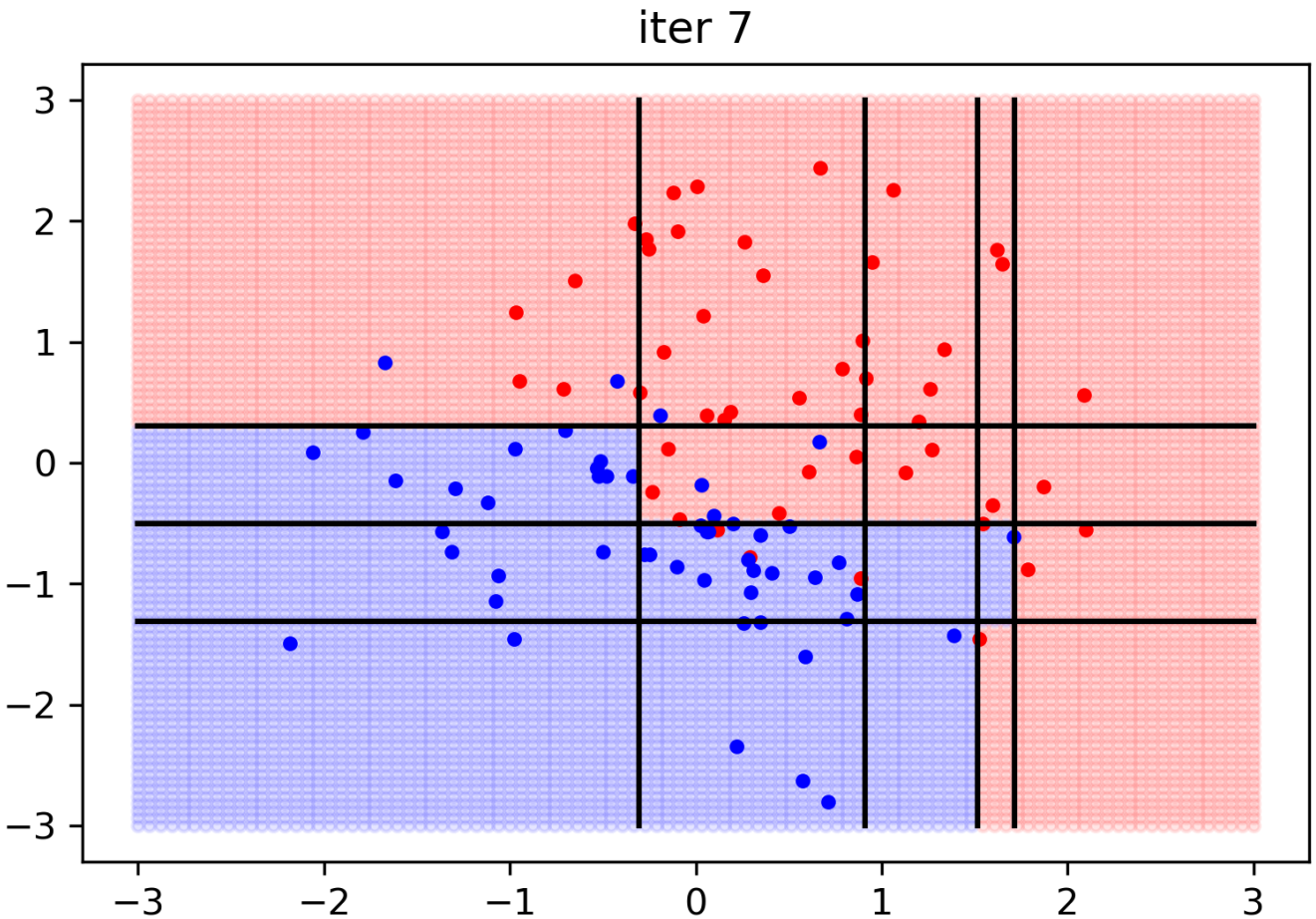
Example: bad separation

+1 (red), 0 (green), -1 (blue)



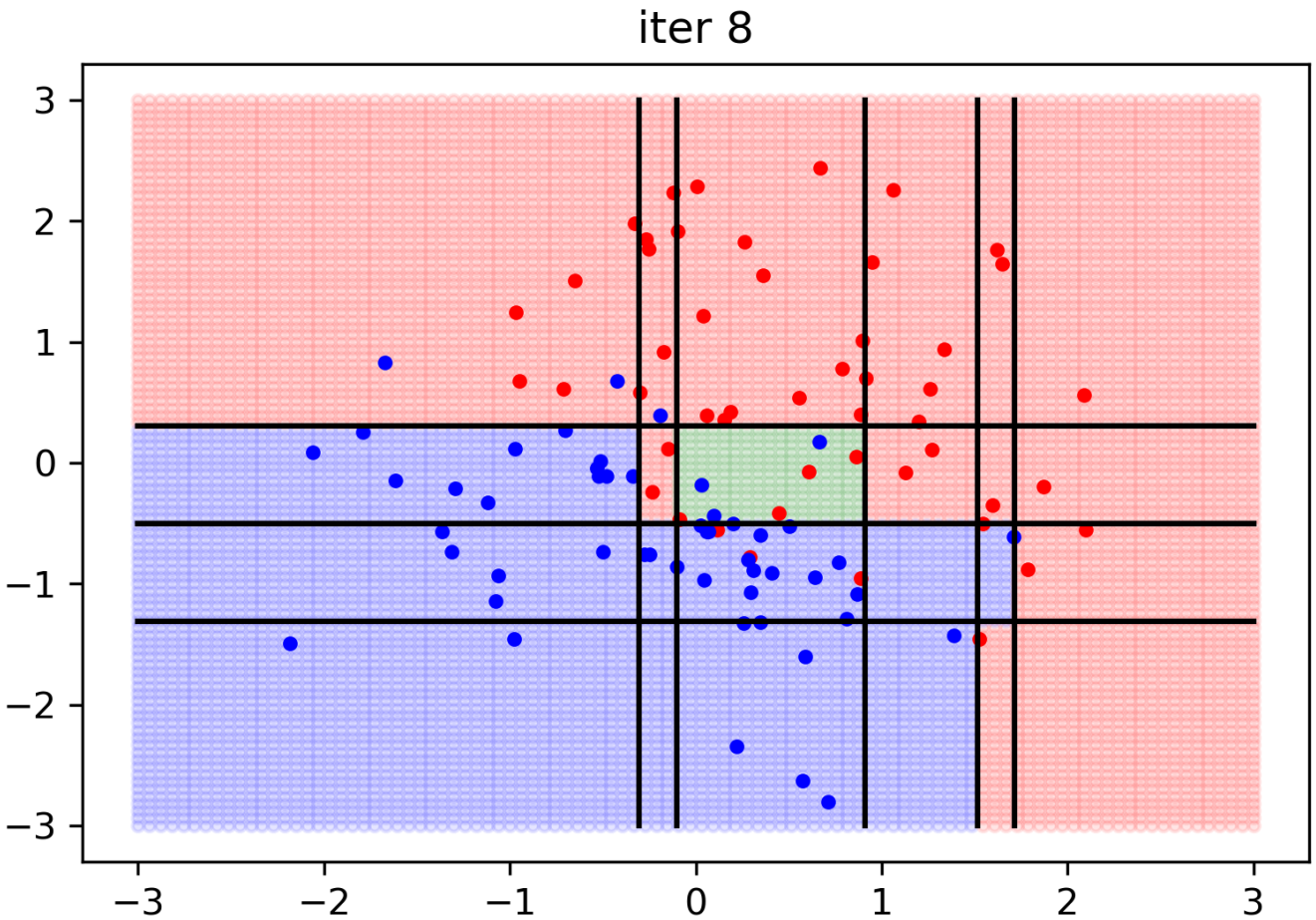
Example: bad separation

+1 (red), 0 (green), -1 (blue)



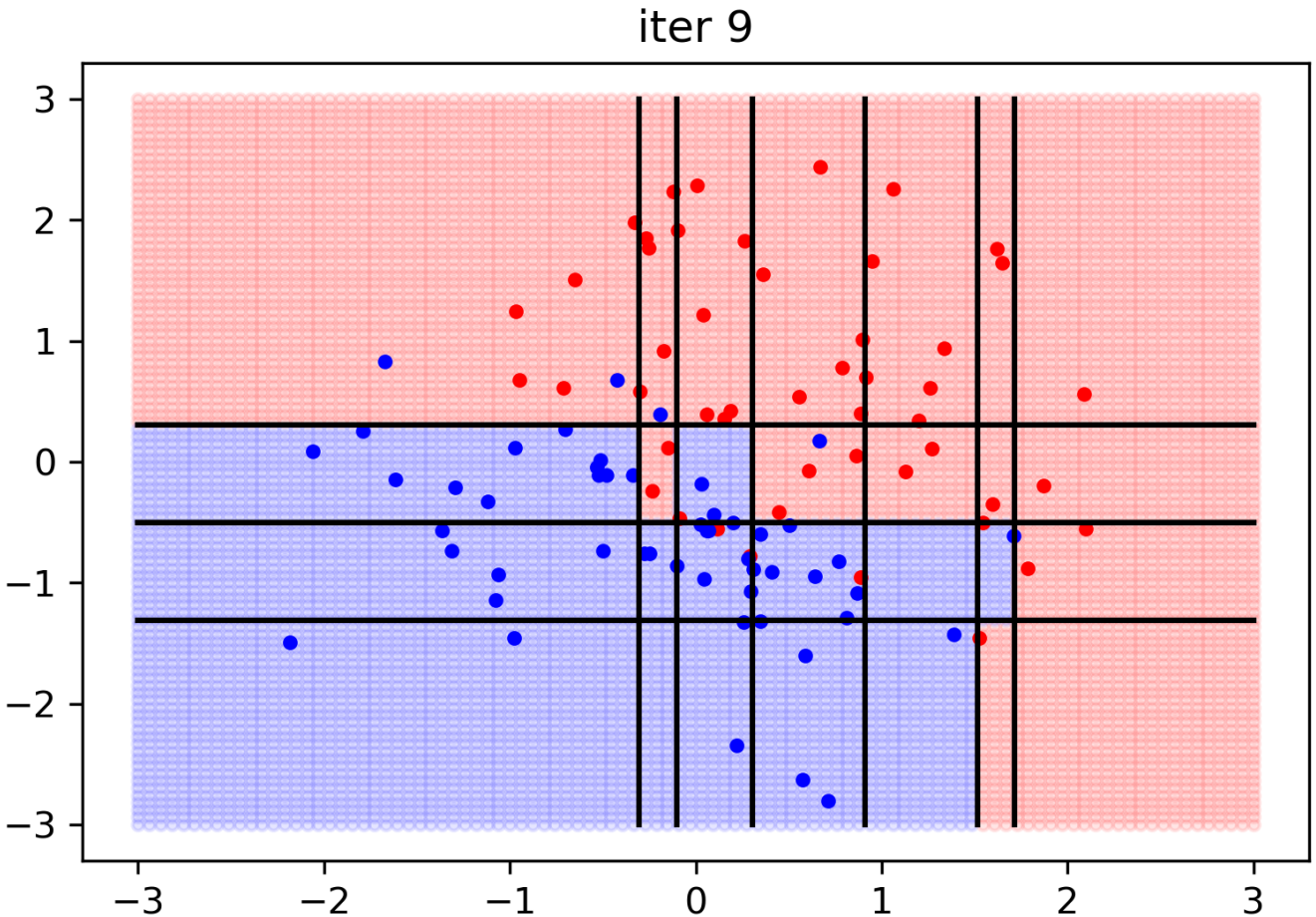
Example: bad separation

+1 (red), 0 (green), -1 (blue)



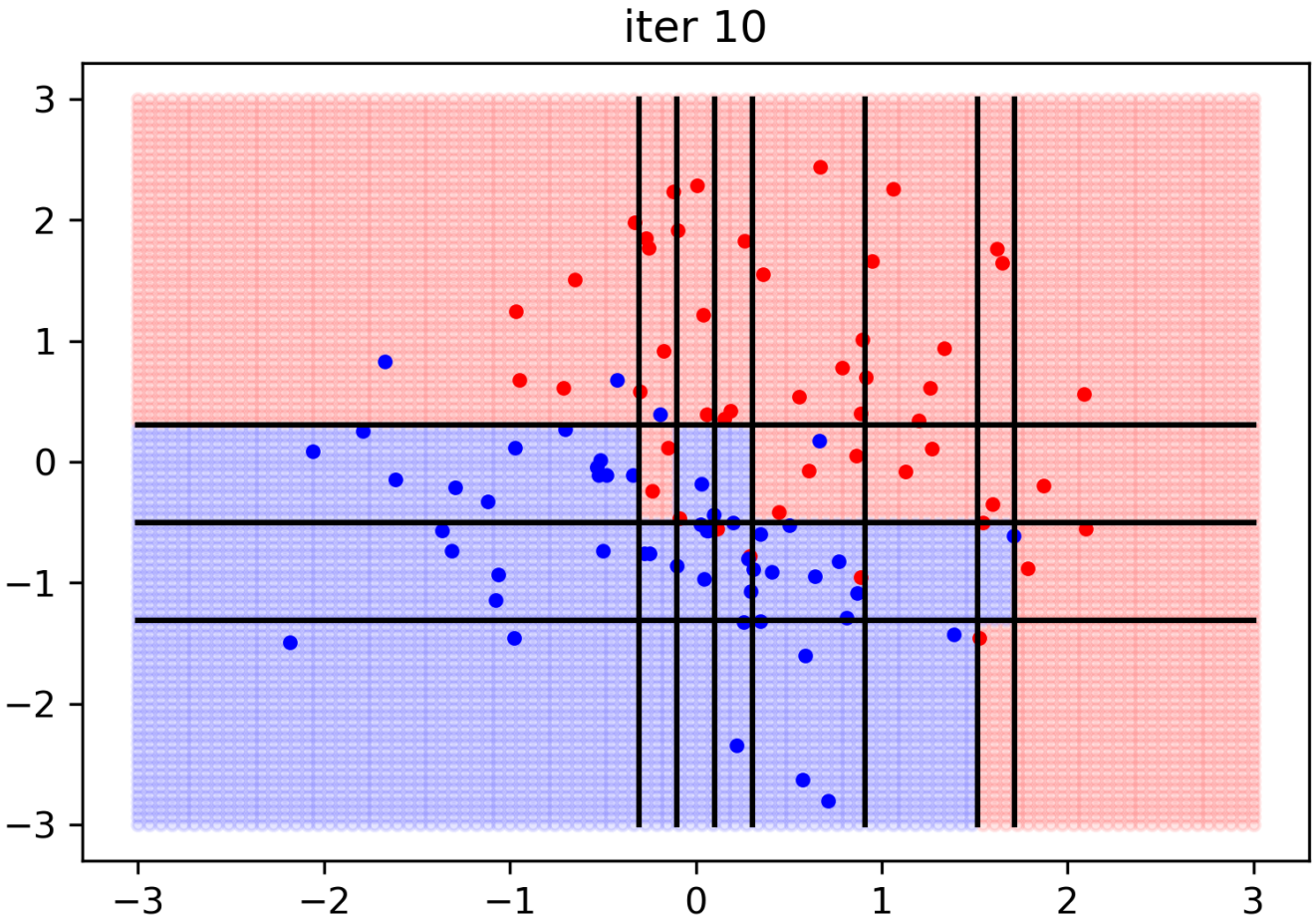
Example: bad separation

+1 (red), 0 (green), -1 (blue)



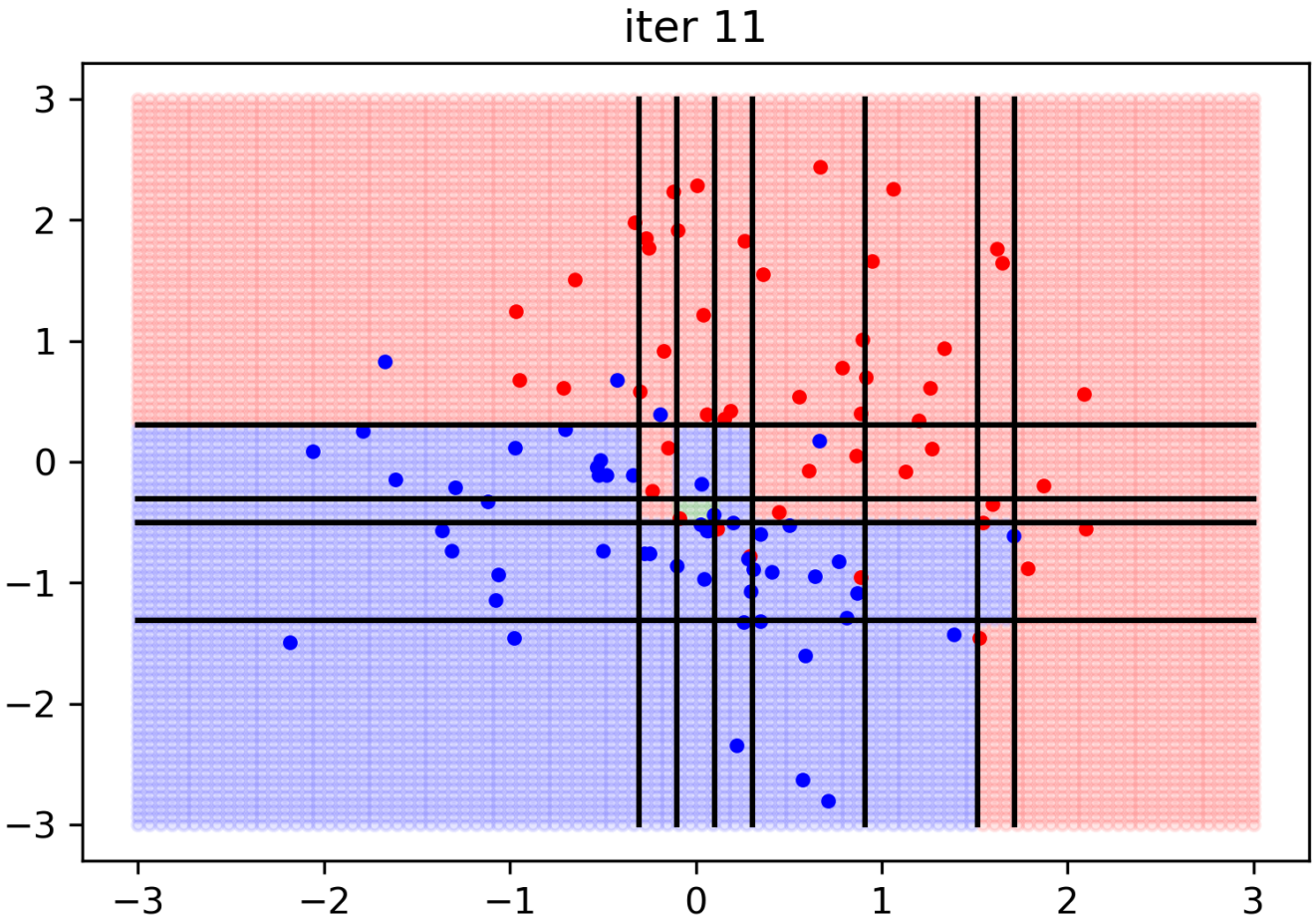
Example: bad separation

+1 (red), 0 (green), -1 (blue)



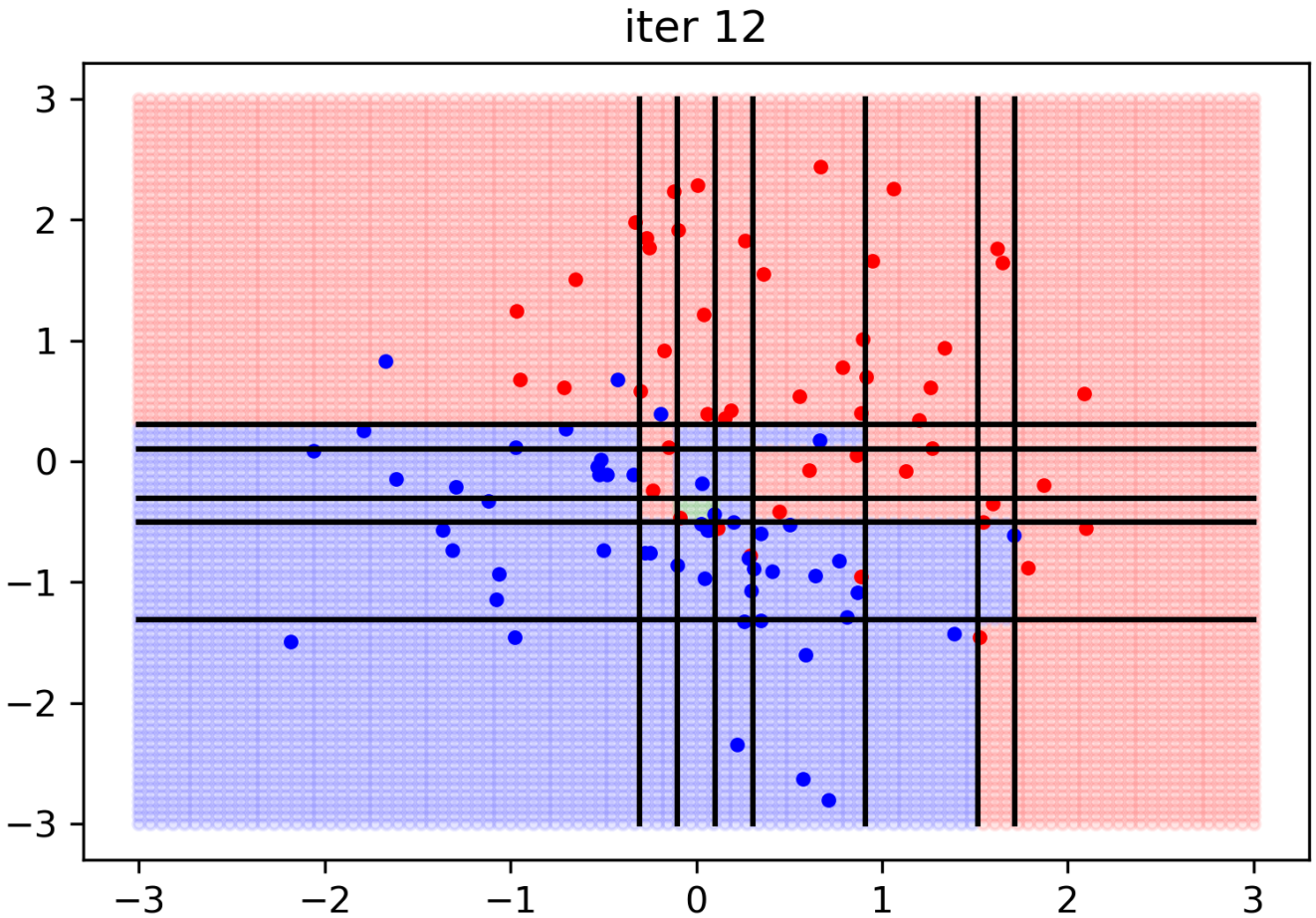
Example: bad separation

+1 (red), 0 (green), -1 (blue)



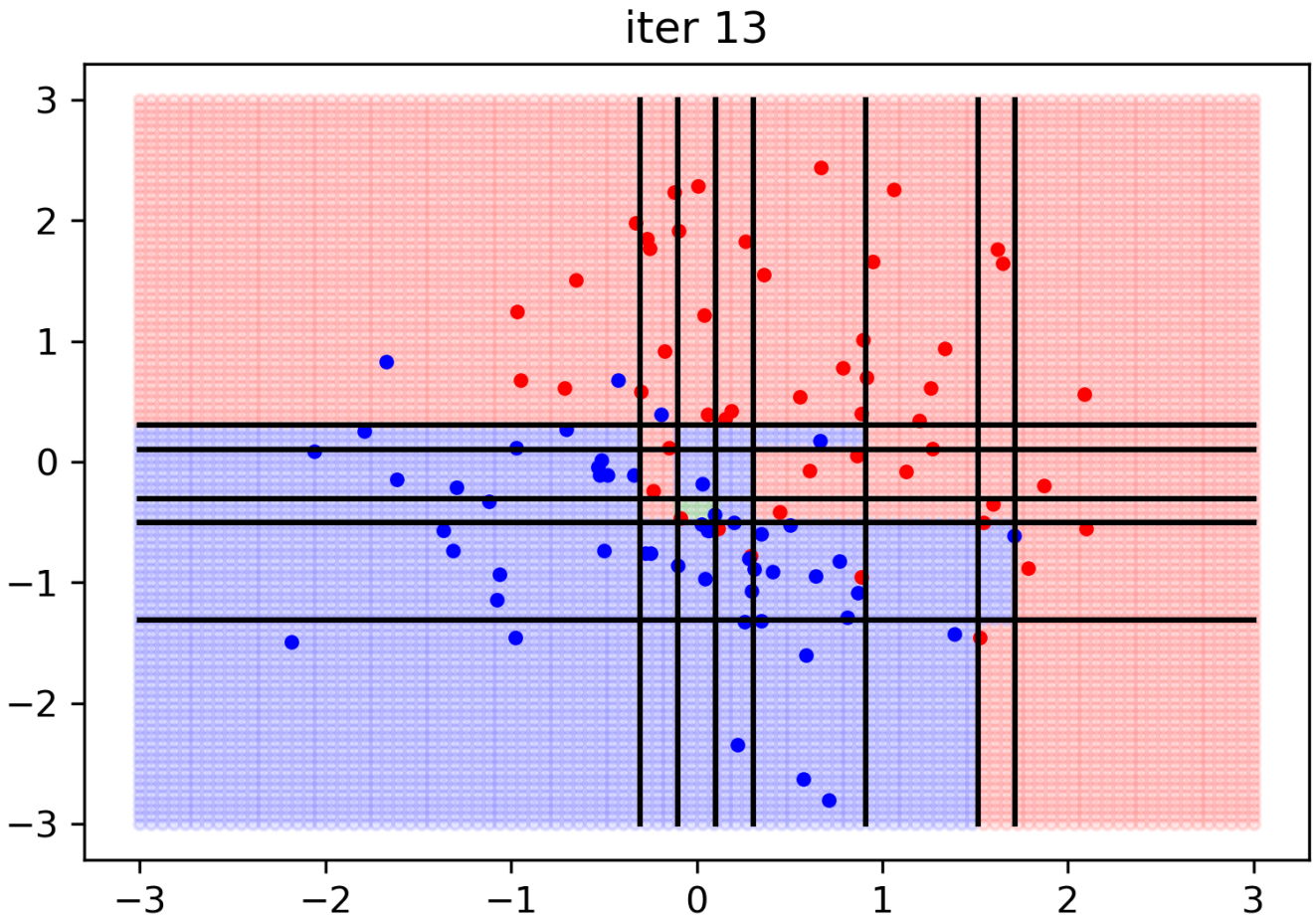
Example: bad separation

+1 (red), 0 (green), -1 (blue)



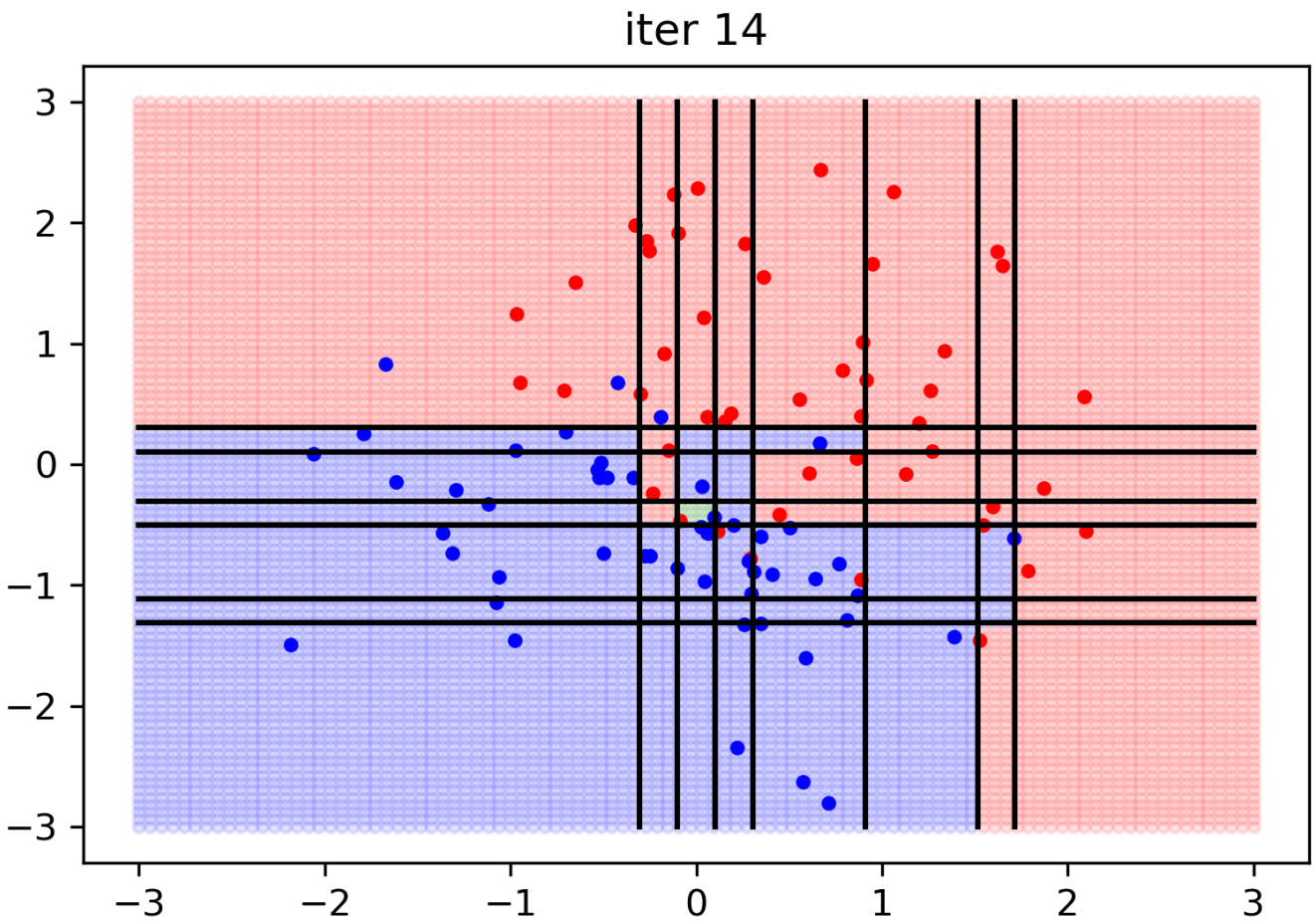
Example: bad separation

+1 (red), 0 (green), -1 (blue)



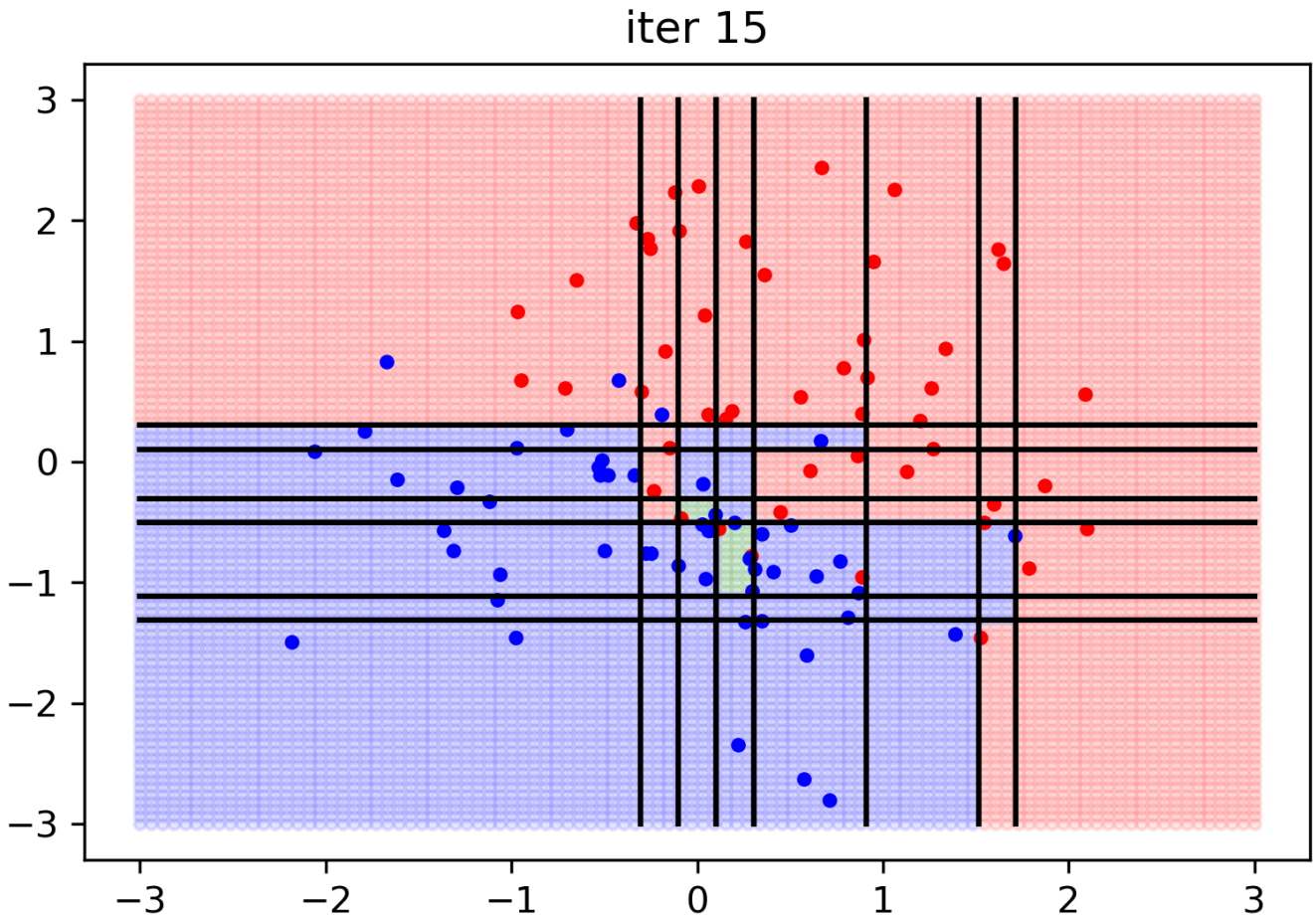
Example: bad separation

+1 (red), 0 (green), -1 (blue)



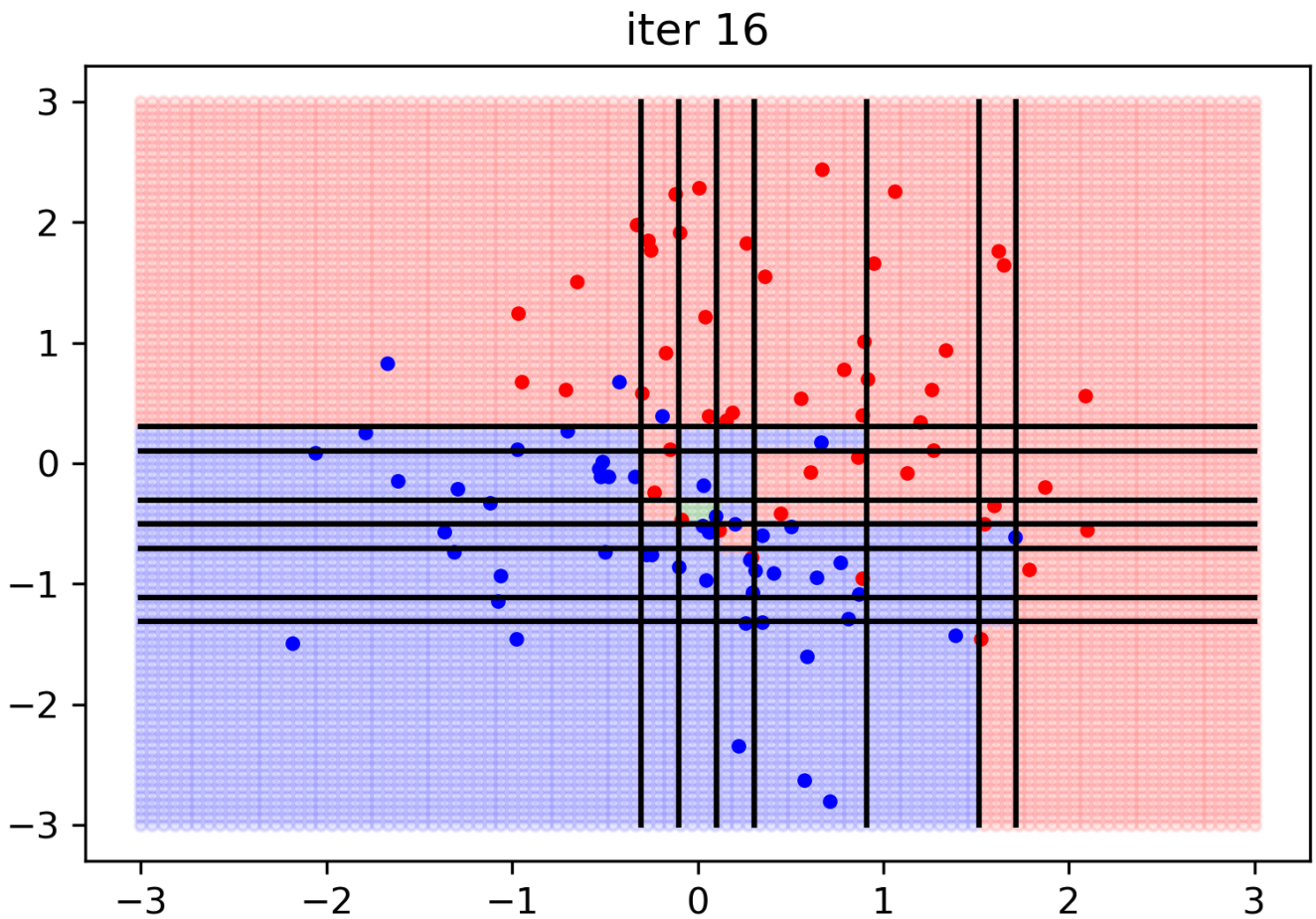
Example: bad separation

+1 (red), 0 (green), -1 (blue)



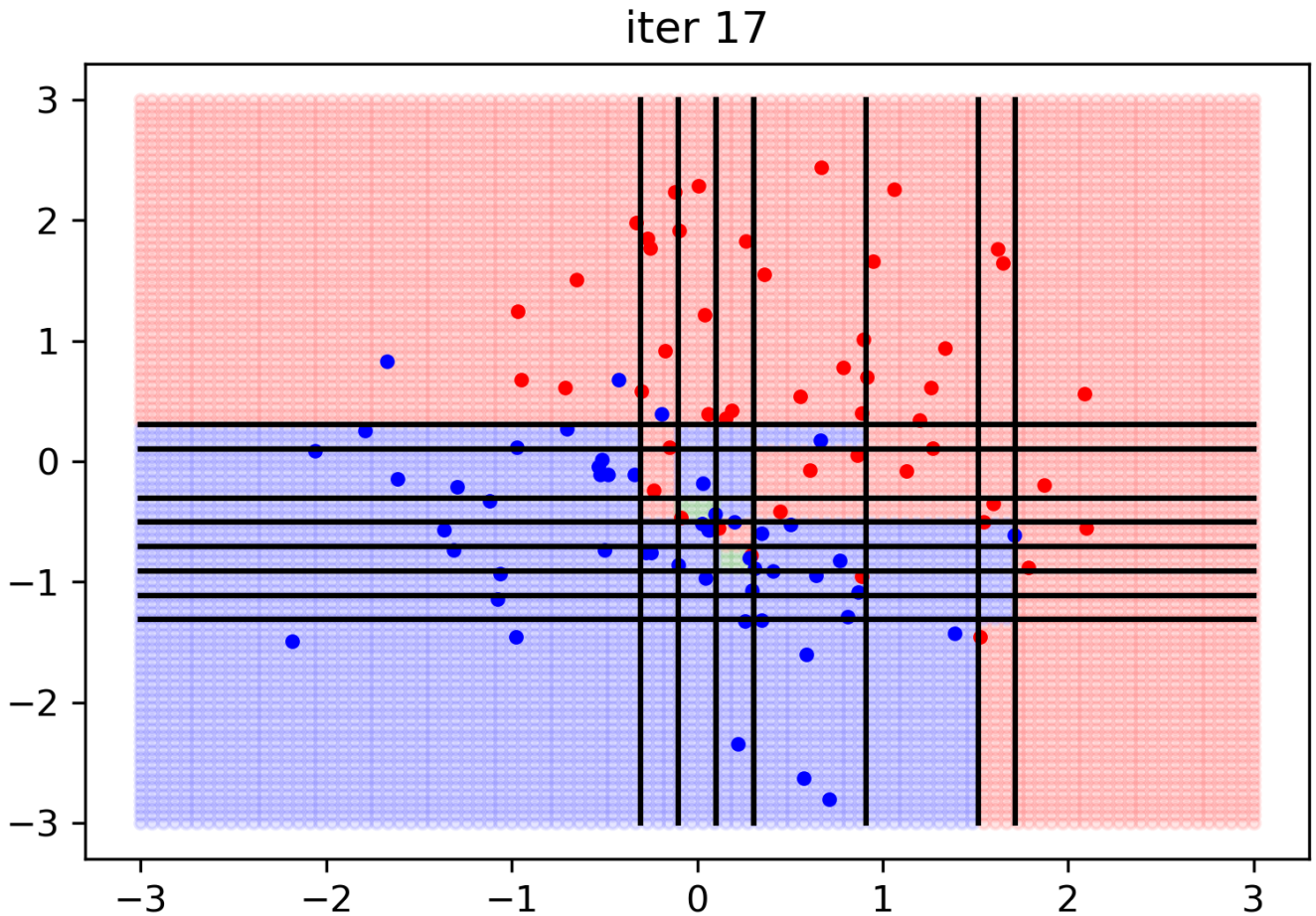
Example: bad separation

+1 (red), 0 (green), -1 (blue)



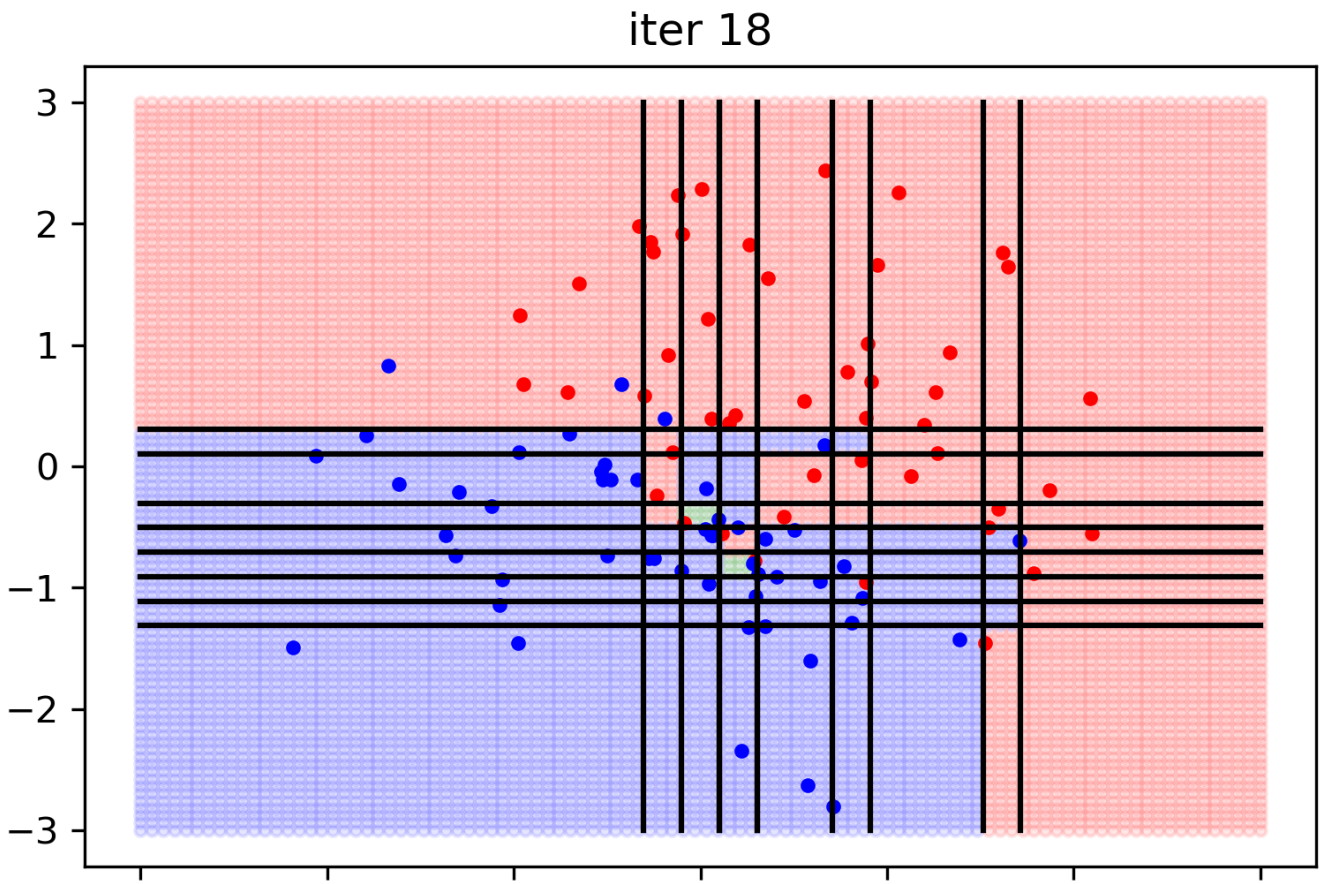
Example: bad separation

+1 (red), 0 (green), -1 (blue)



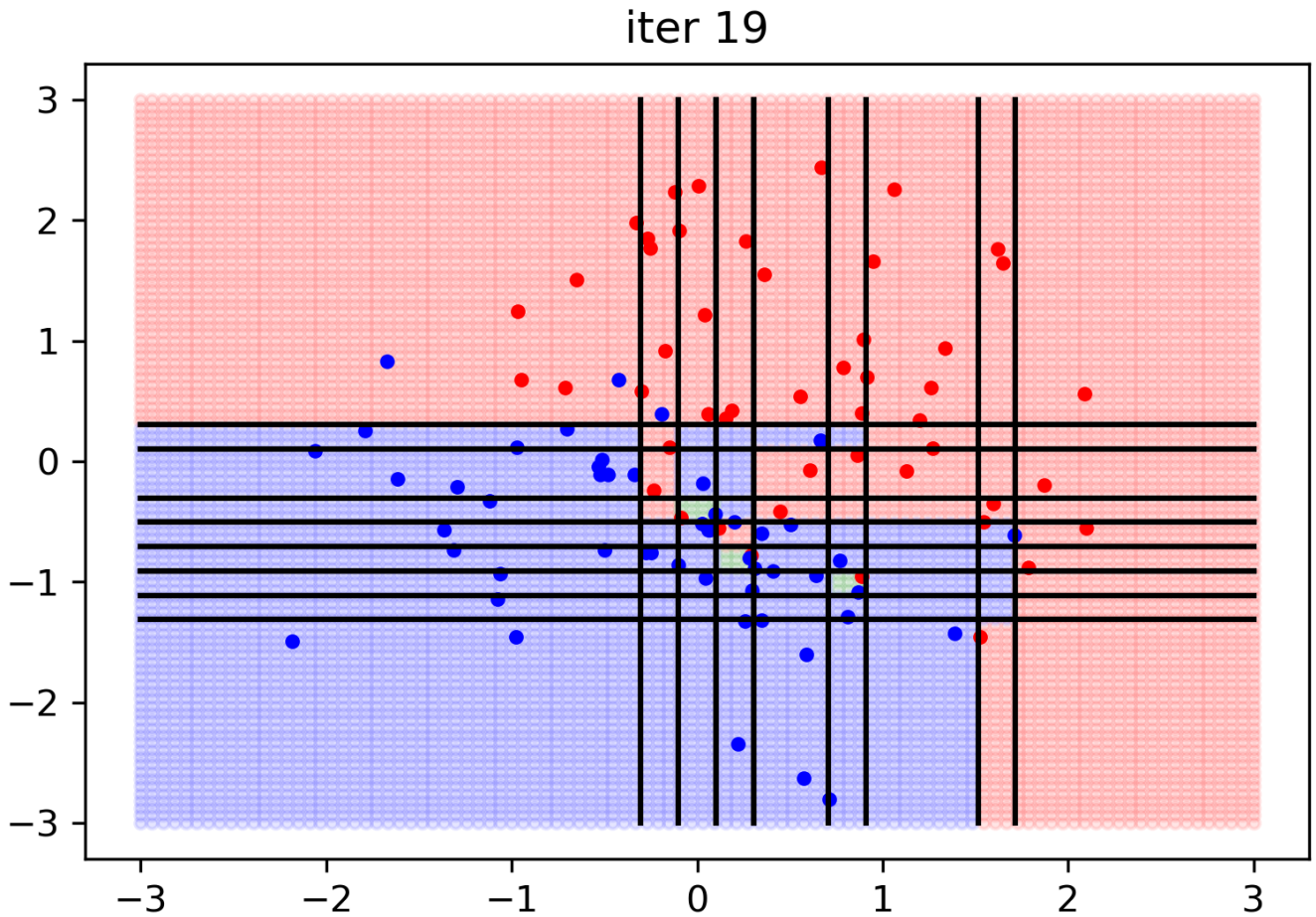
Example: bad separation

+1 (red), 0 (green), -1 (blue)



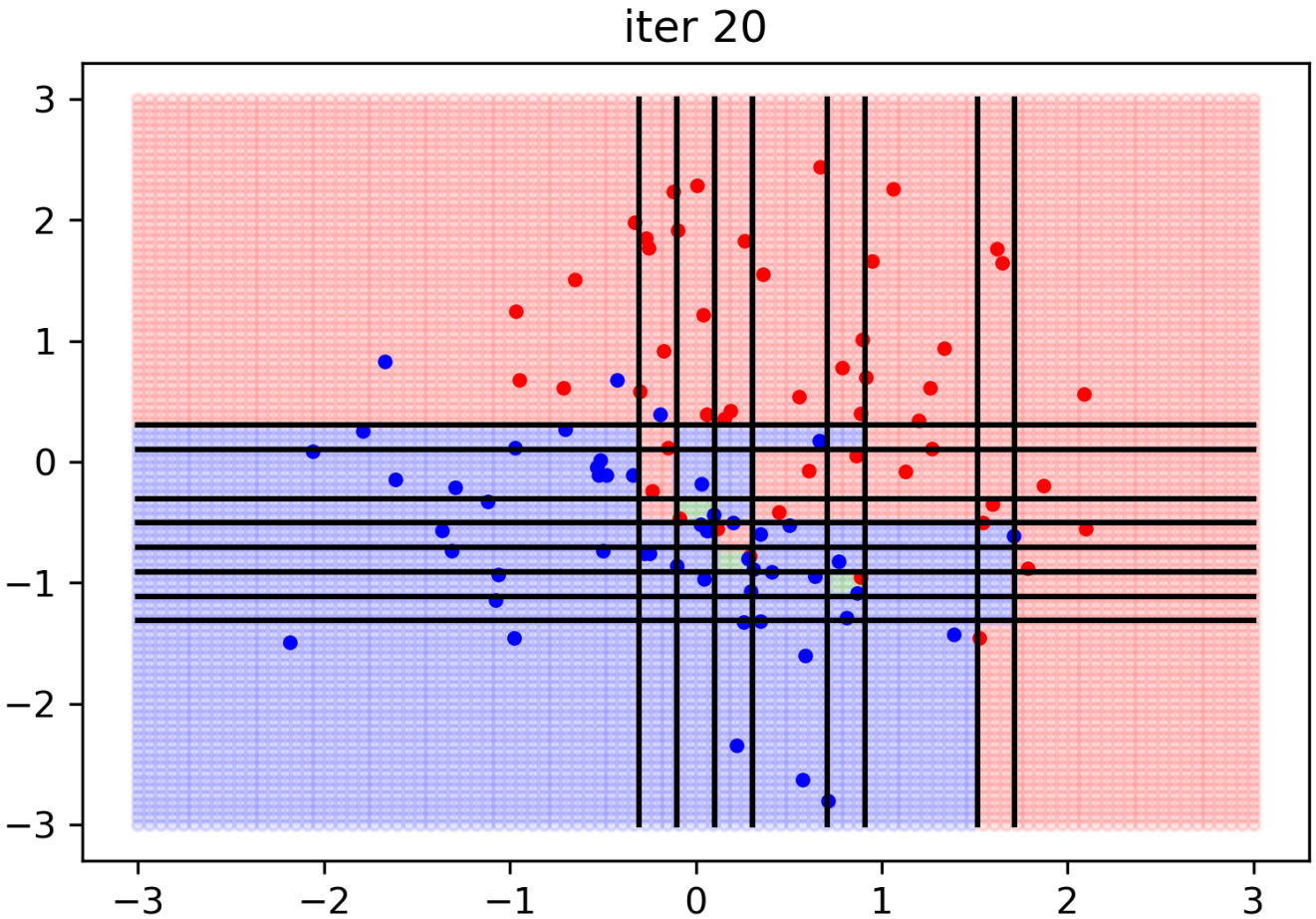
Example: bad separation

+1 (red), 0 (green), -1 (blue)



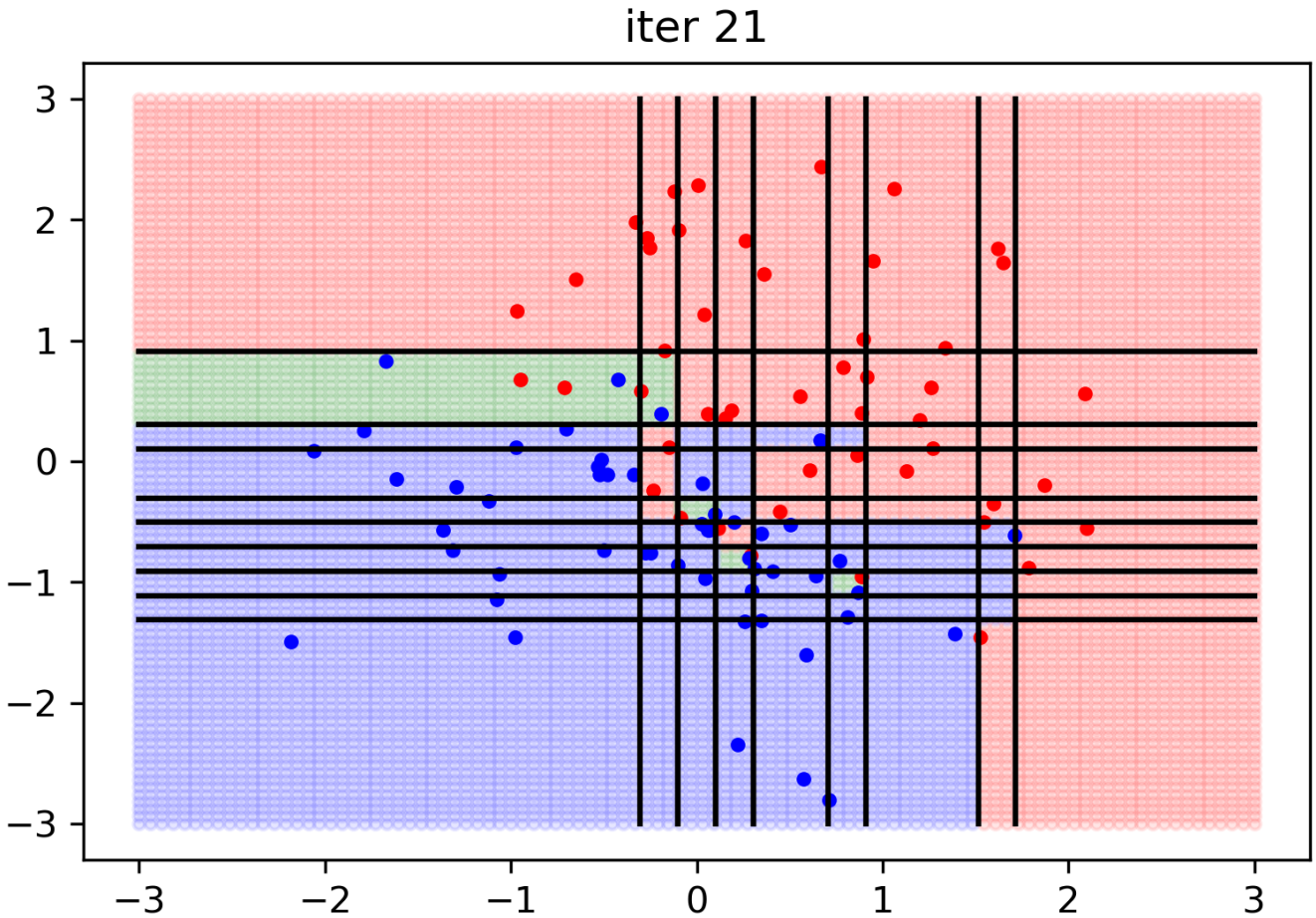
Example: bad separation

+1 (red), 0 (green), -1 (blue)



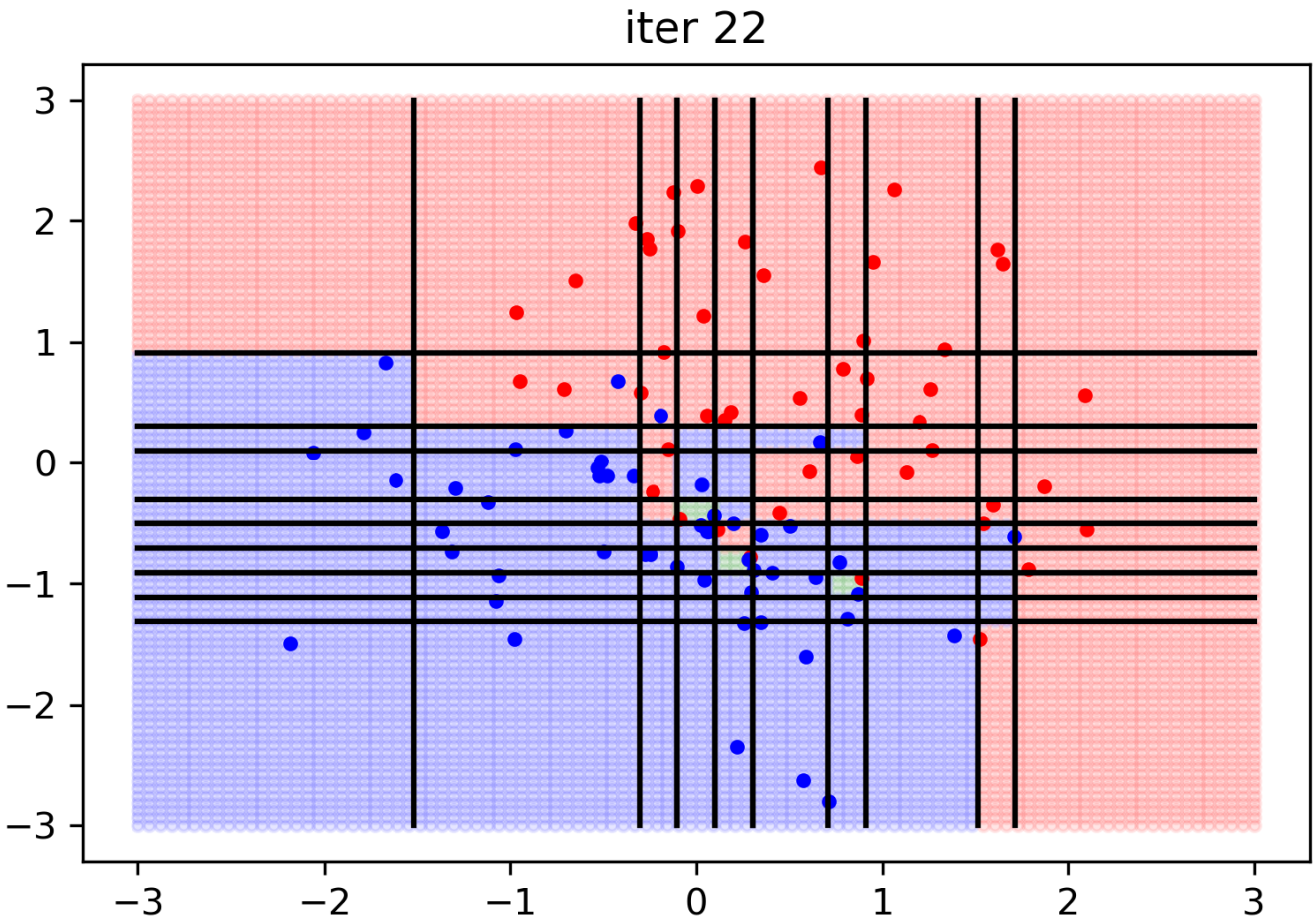
Example: bad separation

+1 (red), 0 (green), -1 (blue)



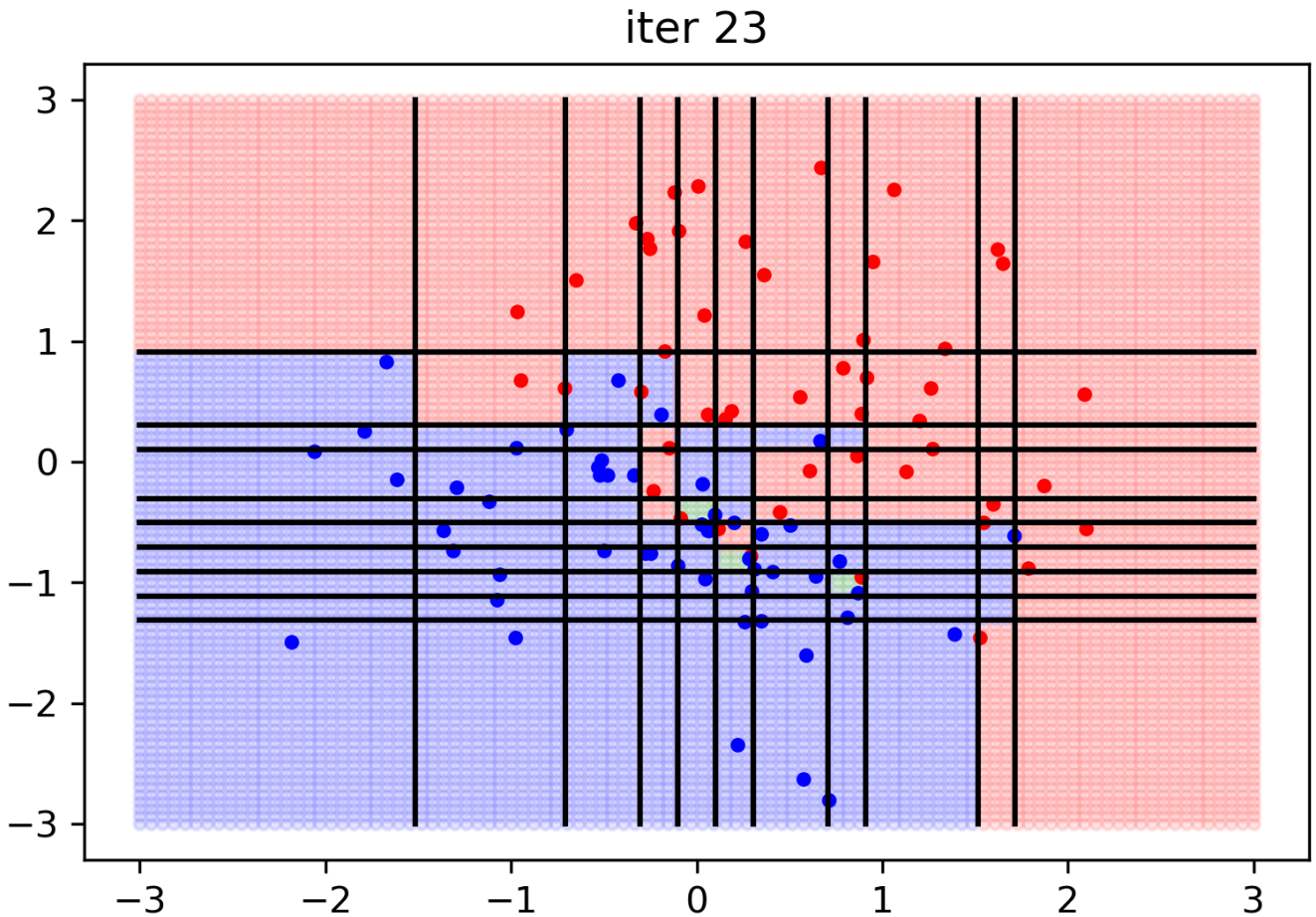
Example: bad separation

+1 (red), 0 (green), -1 (blue)



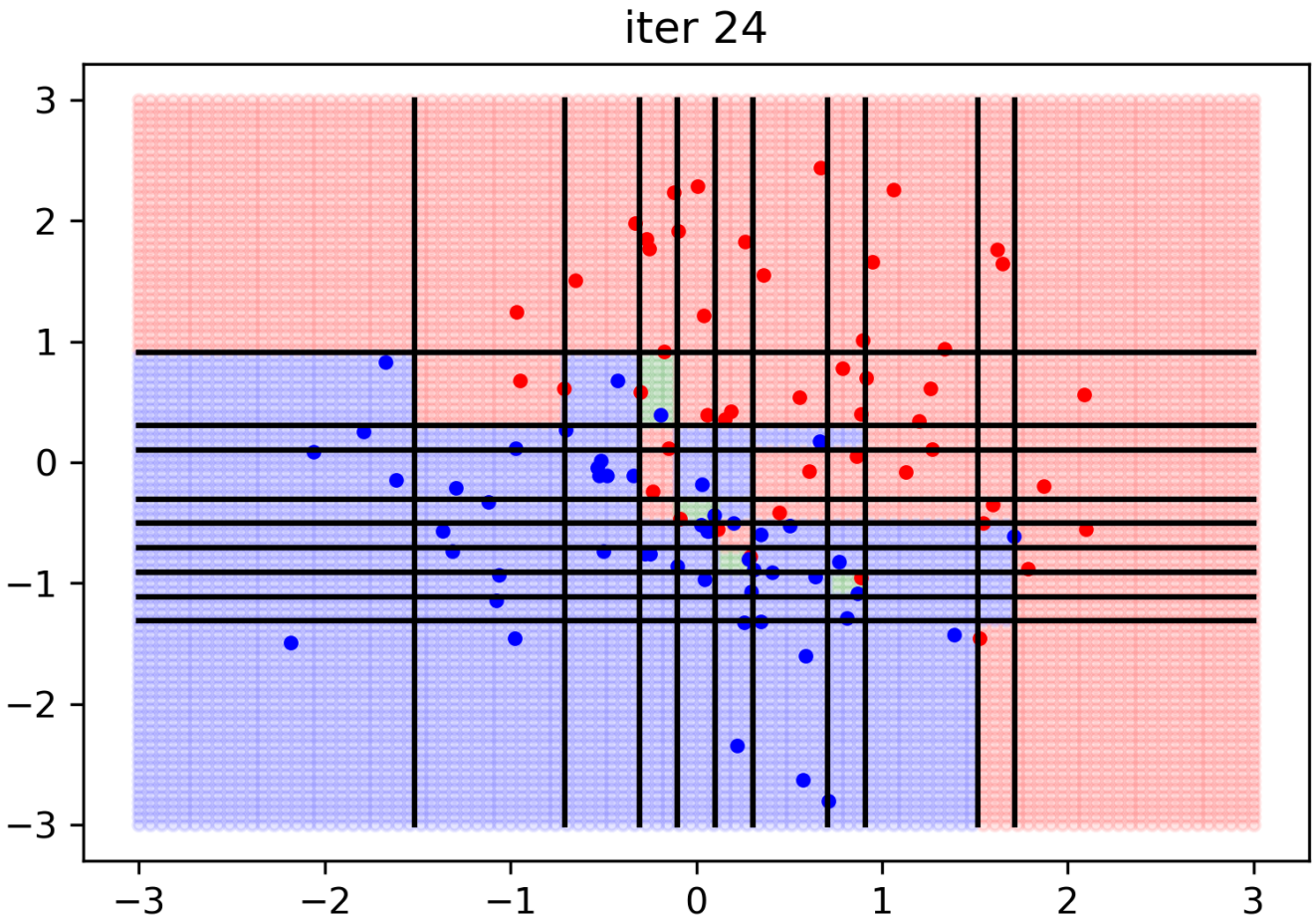
Example: bad separation

+1 (red), 0 (green), -1 (blue)



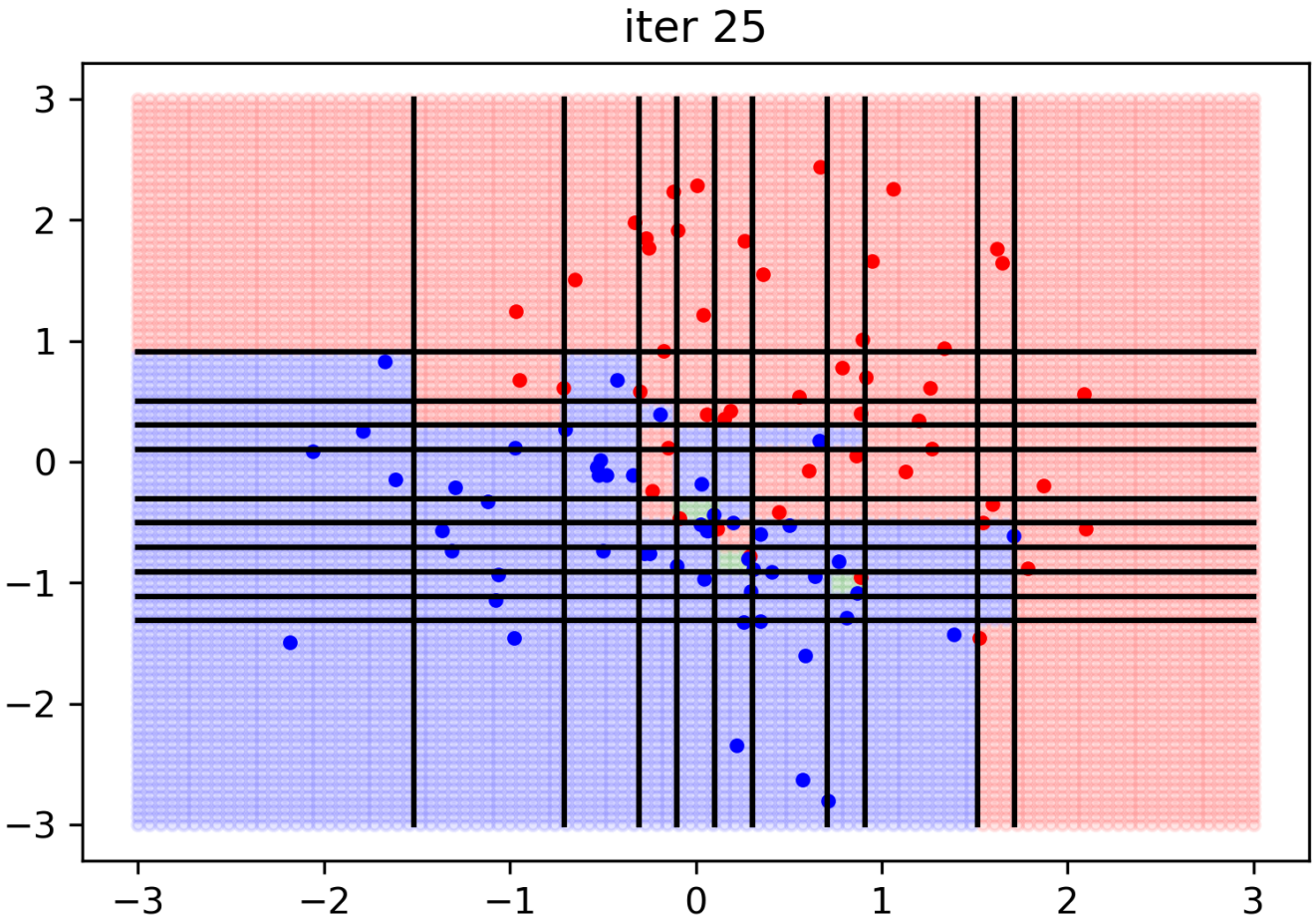
Example: bad separation

+1 (red), 0 (green), -1 (blue)



Example: bad separation

+1 (red), 0 (green), -1 (blue)



Overfitting

- If 0 error tree exists, greedy method can always find it
- Stopping conditions reduce overfitting, but greatly reduce power
- Better solution?

  Ans: One giant tree $\ll$ a lot of little trees!
- Next up: Ensemble methods!

Extra reading on information theory concepts

Sections 2.1 to 2.5 of *Elements of Information Theory*, Cover and Thomas

Summary

- Using a decision tree as a classifier / regressor
- Greedy method for constructing a decision tree
- Better metrics for "worst leaf"? Information gain
- Decision trees can overfit