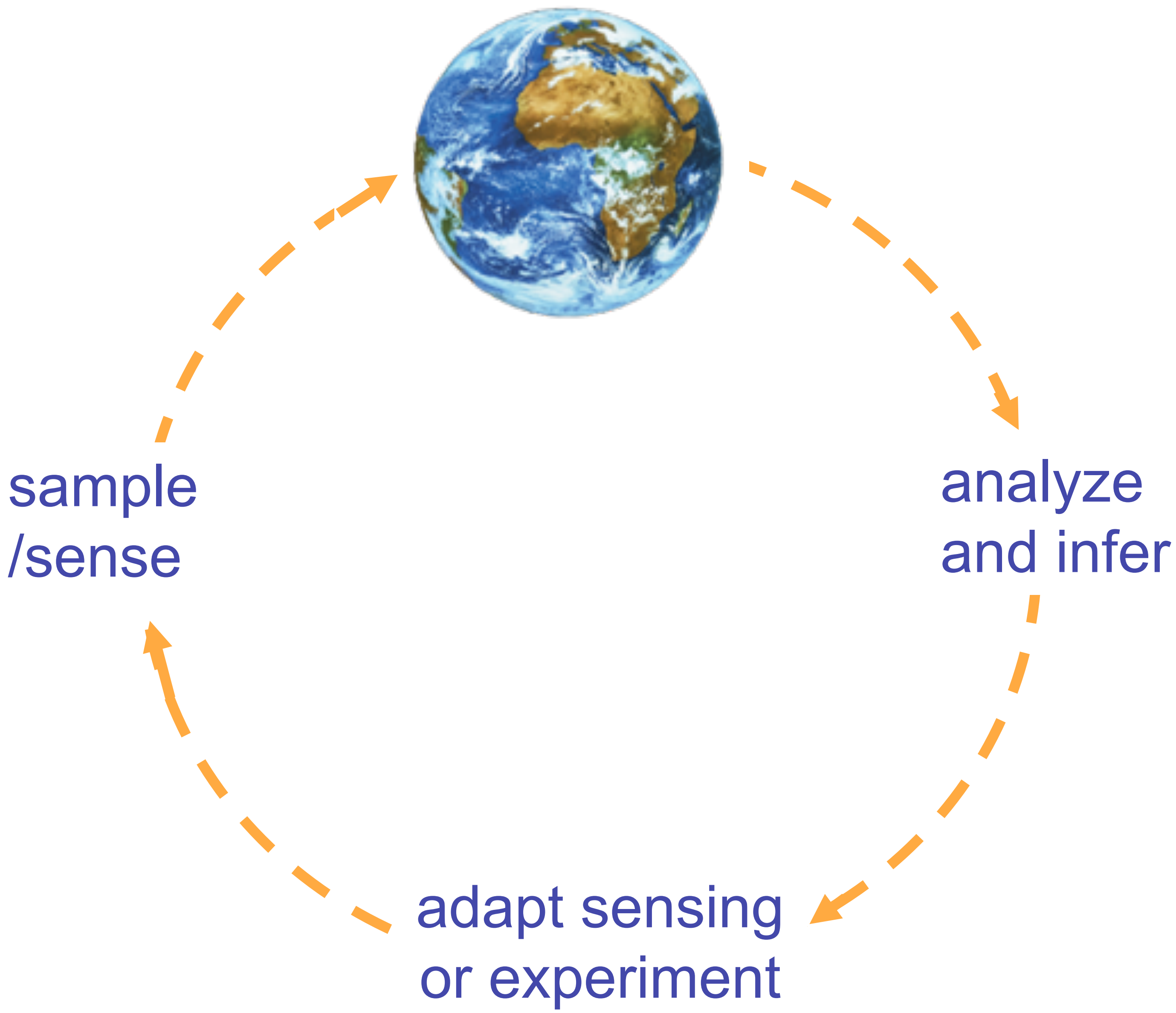
Active Learning



Machine Learning

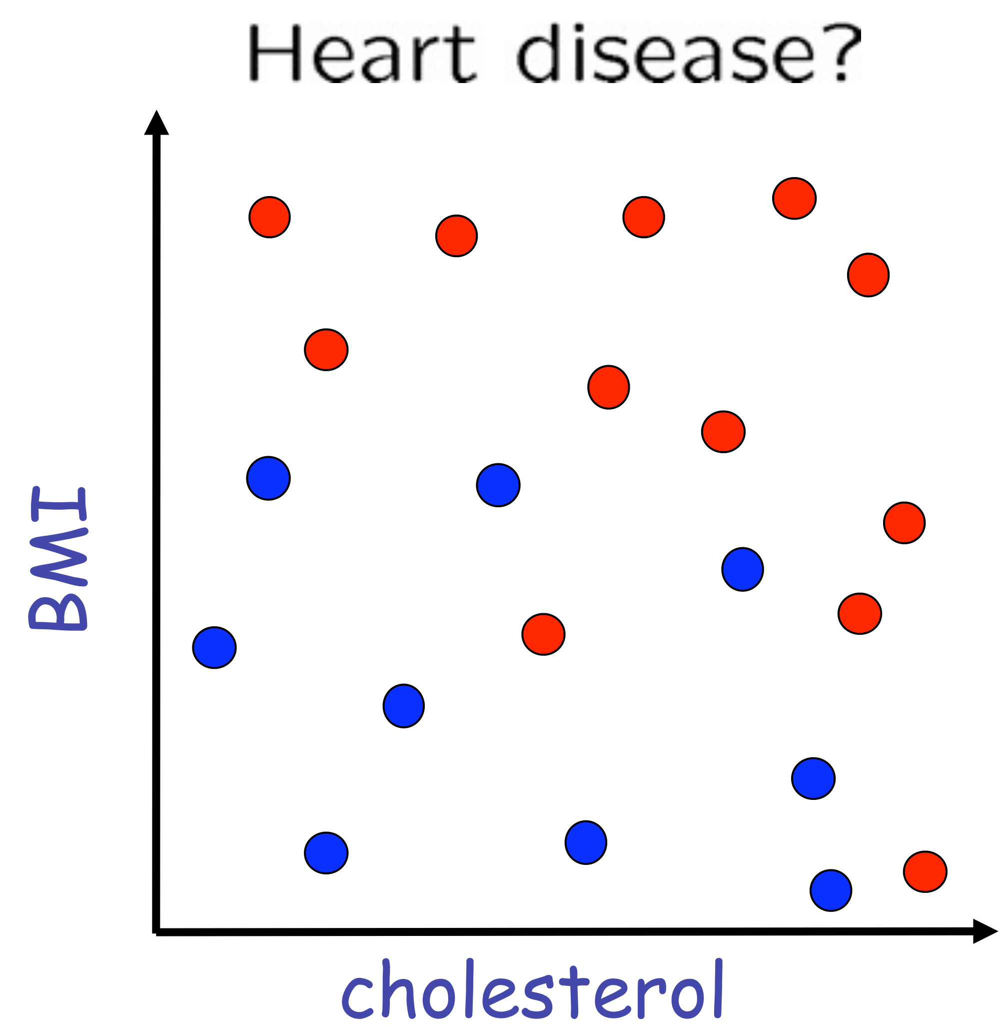
Training examples come in pairs, feature X and label Y.

Goal: Design a rule for predicting Y given X

Machine Learning

Training examples come in pairs, feature X and label Y.

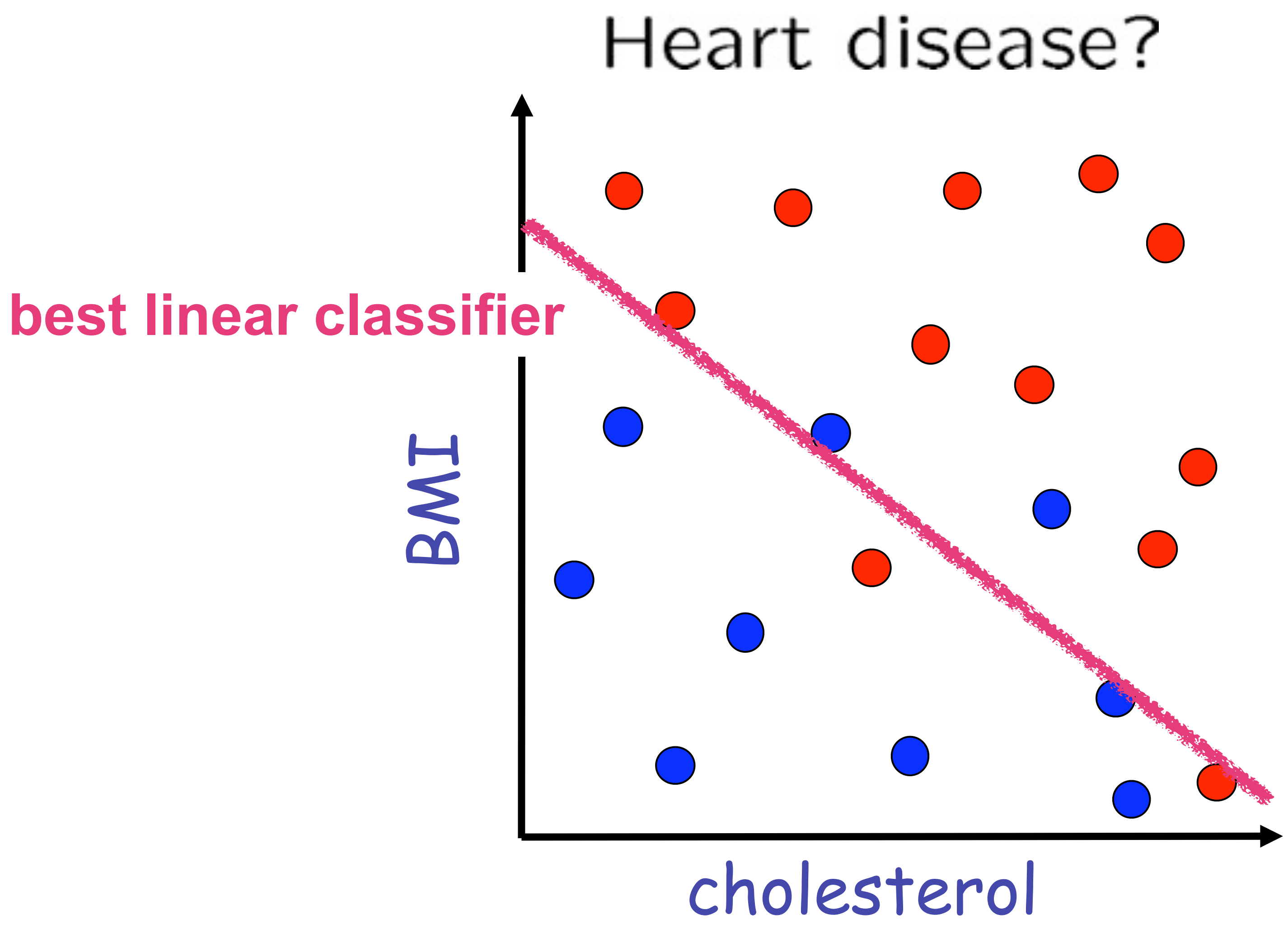
Goal: Design a rule for predicting Y given X



Machine Learning

Training examples come in pairs, feature X and label Y.

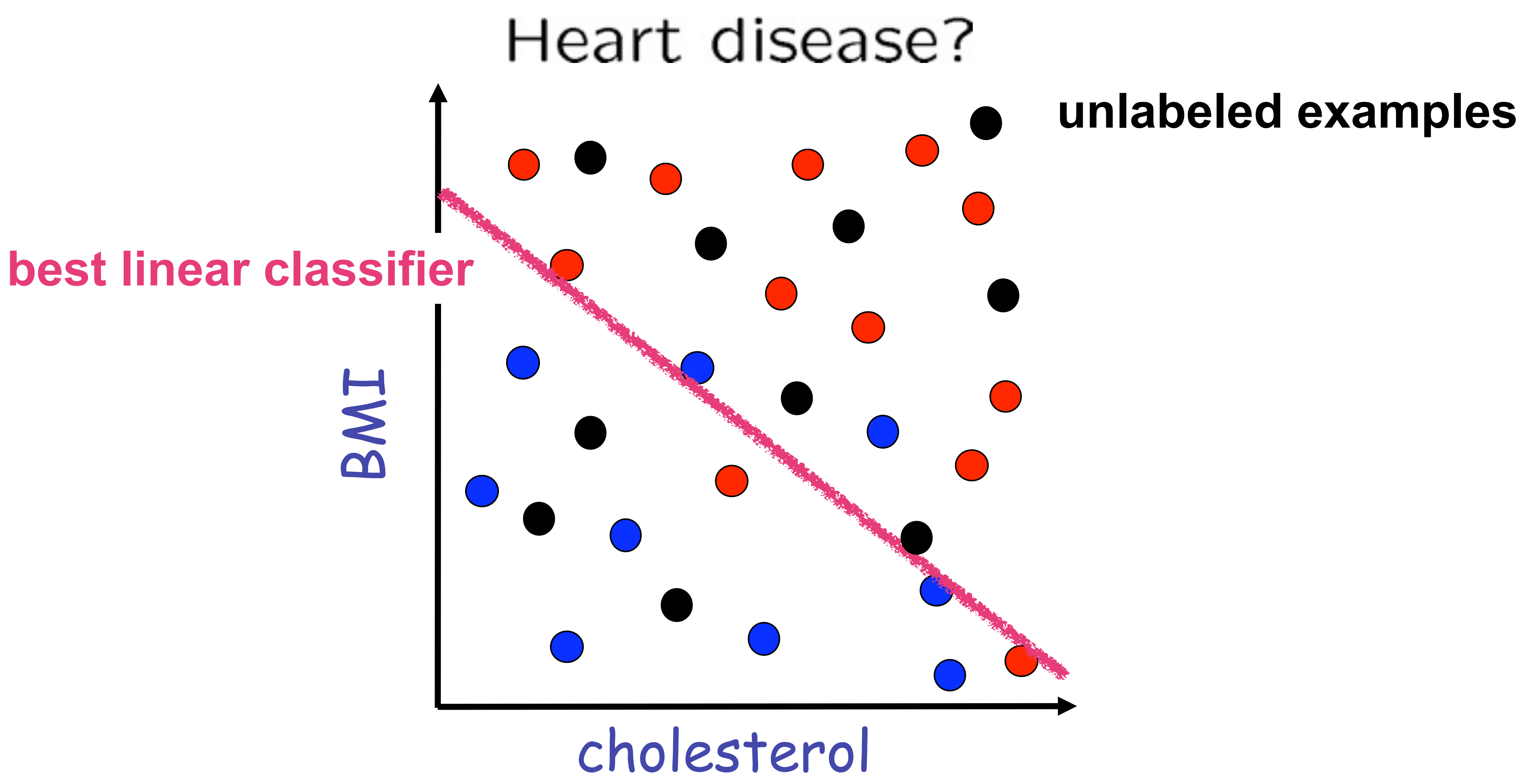
Goal: Design a rule for predicting Y given X



Machine Learning

Training examples come in pairs, feature X and label Y.

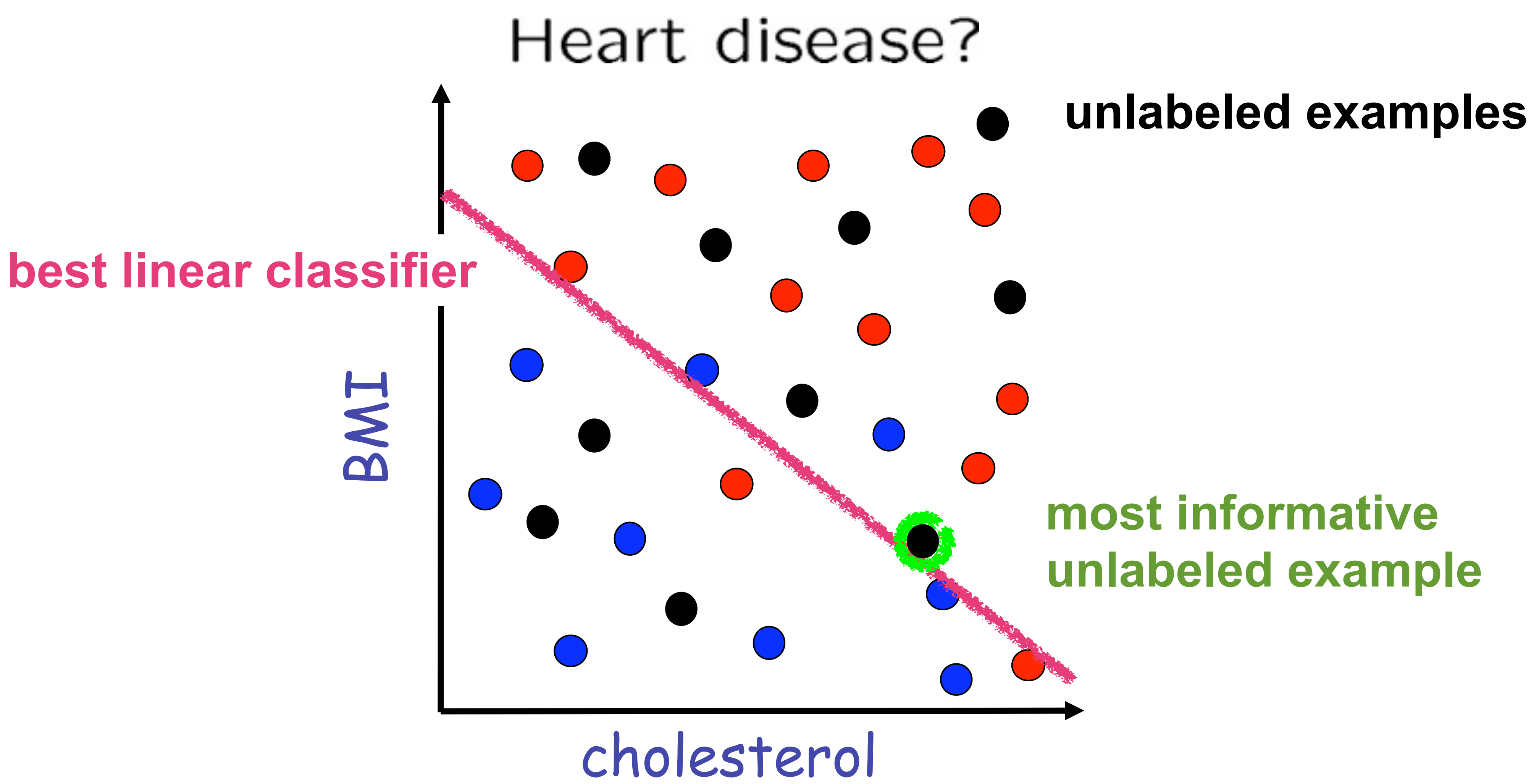
Goal: Design a rule for predicting Y given X



Machine Learning

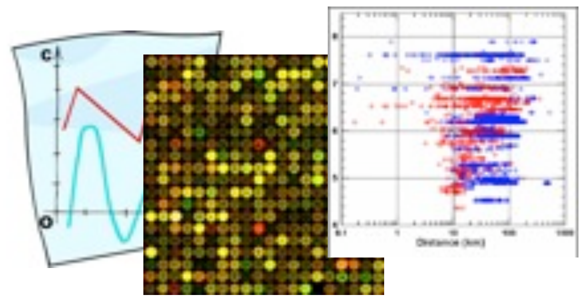
Training examples come in pairs, feature X and label Y.

Goal: Design a rule for predicting Y given X



Machine Learning (Passive)

Raw unlabeled data



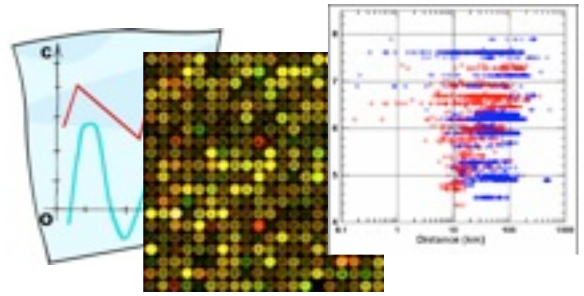
$$X_1, X_2, X_3, \dots$$

passive learner

expert/oracle
analyzes/experiments
to determine labels

Machine Learning (Passive)

Raw unlabeled data



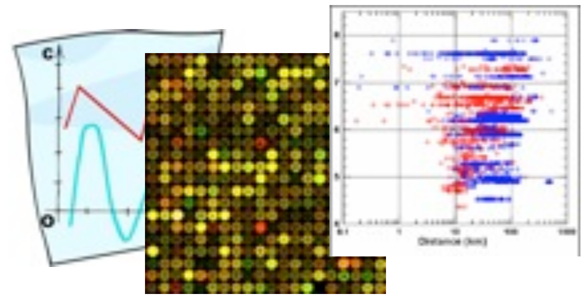
$$X_1, X_2, X_3, \ldots$$

passive learner

expert/oracle
analyzes/experiments
to determine labels

Machine Learning (Passive)

Raw unlabeled data



$X_1, X_2, X_3, \ldots$

$(X_1, Y_1), (X_2, Y_2), (X_3, Y_3), \ldots$

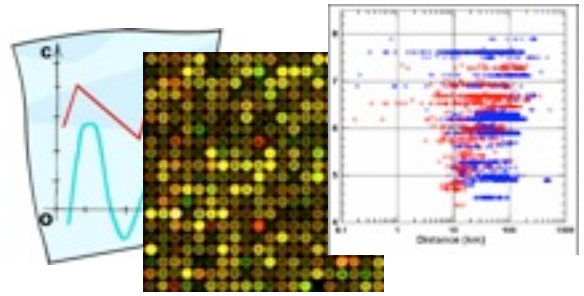
Labeled data

passive learner

expert/oracle
analyzes/experiments
to determine labels

Machine Learning (Passive)

Raw unlabeled data



$$X_1, X_2, X_3, \ldots$$

$$(X_1, Y_1), (X_2, Y_2), (X_3, Y_3), \ldots$$

Labeled data

passive learner

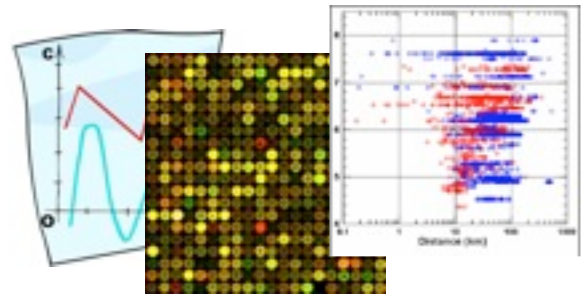
automatic classifier

expert/oracle

analyzes/experiments to determine labels

Active Learning

Raw unlabeled data



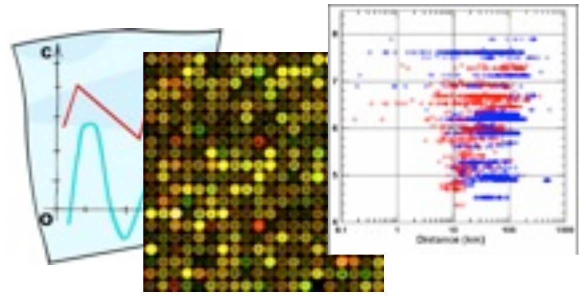
$$X_1, X_2, X_3, \ldots$$

active learner

expert/oracle
analyzes/experiments
to determine labels

Active Learning

Raw unlabeled data



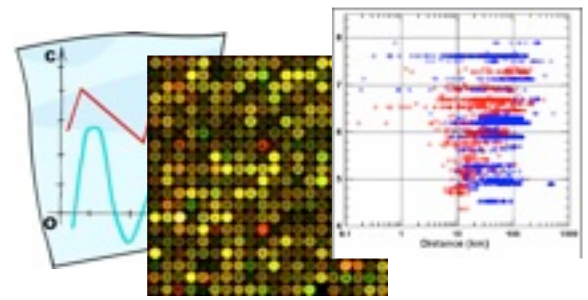
$$X_1, X_2, X_3, \ldots$$

active learner

expert/oracle
analyzes/experiments
to determine labels

Active Learning

Raw unlabeled data



$X_1, X_2, X_3, \ldots$

Learner requests labels
for **selected** data

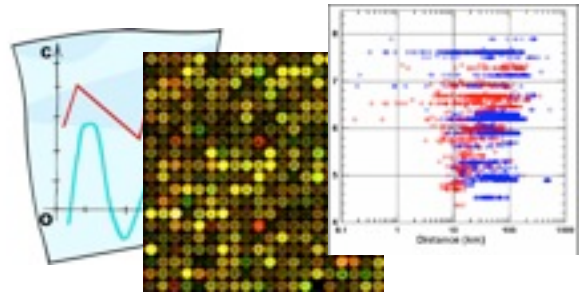
$(X_1, ?)$

active learner

expert/oracle
analyzes/experiments
to determine labels

Active Learning

Raw unlabeled data



$X_1, X_2, X_3, \ldots$

Learner requests labels
for **selected** data

$(X_1, ?)$

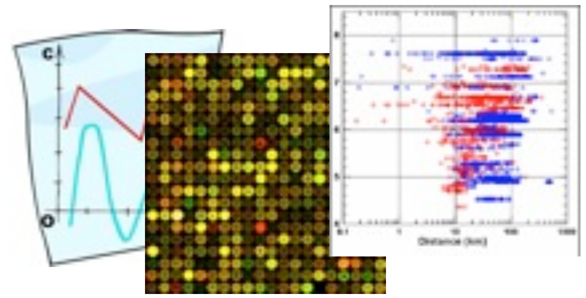
(X_1, Y_1)

active learner

expert/oracle
analyzes/experiments
to determine labels

Active Learning

Raw unlabeled data



$X_1, X_2, X_3, \ldots$

Learner requests labels
for **selected** data

$(X_1, ?)$

(X_1, Y_1)

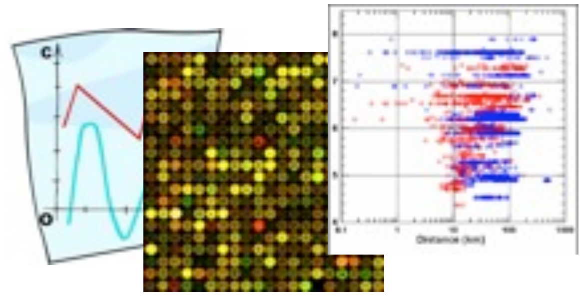
$(X_3, ?)$

active learner

expert/oracle
analyzes/experiments
to determine labels

Active Learning

Raw unlabeled data



$X_1, X_2, X_3, \ldots$

Learner requests labels
for **selected** data

$(X_1, ?)$

(X_1, Y_1)

$(X_3, ?)$

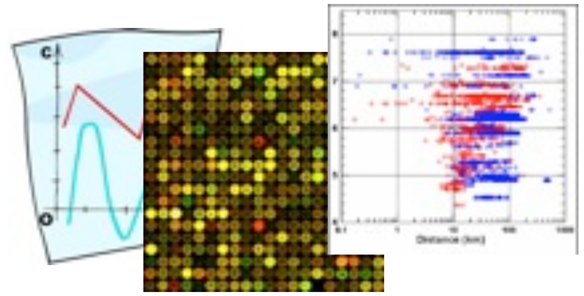
(X_3, Y_3)

active learner

expert/oracle
analyzes/experiments
to determine labels

Active Learning

Raw unlabeled data



$$X_1, X_2, X_3, \ldots$$

Learner requests labels
for **selected** data

$(X_1, ?)$

(X_1, Y_1)

$(X_3, ?)$

(X_3, Y_3)

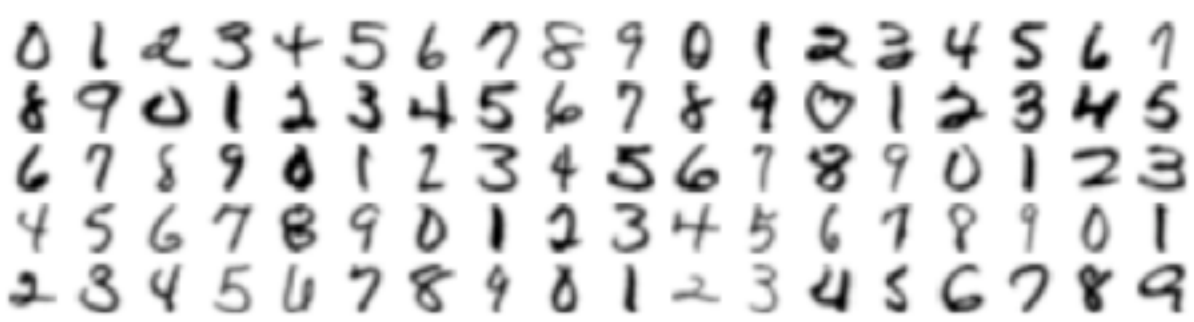
active learner

expert/oracle
analyzes/experiments
to determine labels

automatic classifier

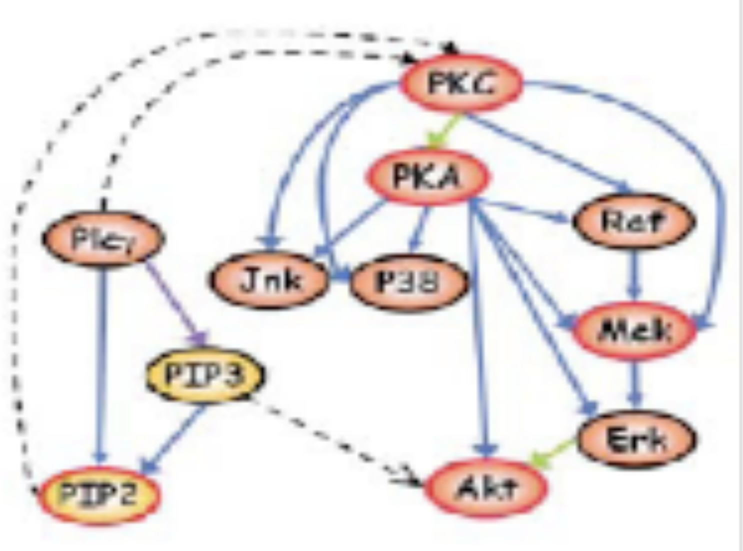
Applications of Active Learning

Hand-written character recognition



Document classification

Systems biology



Sensor networks

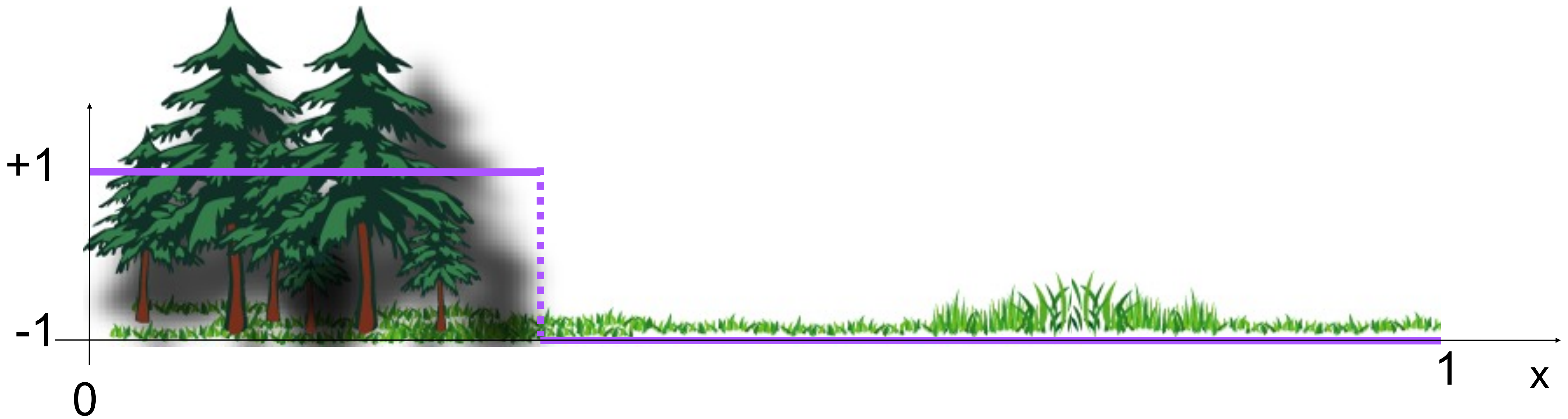
In many applications, obtaining labels or running experiments is costly !

A Stylized Environmental Sensing Task

x

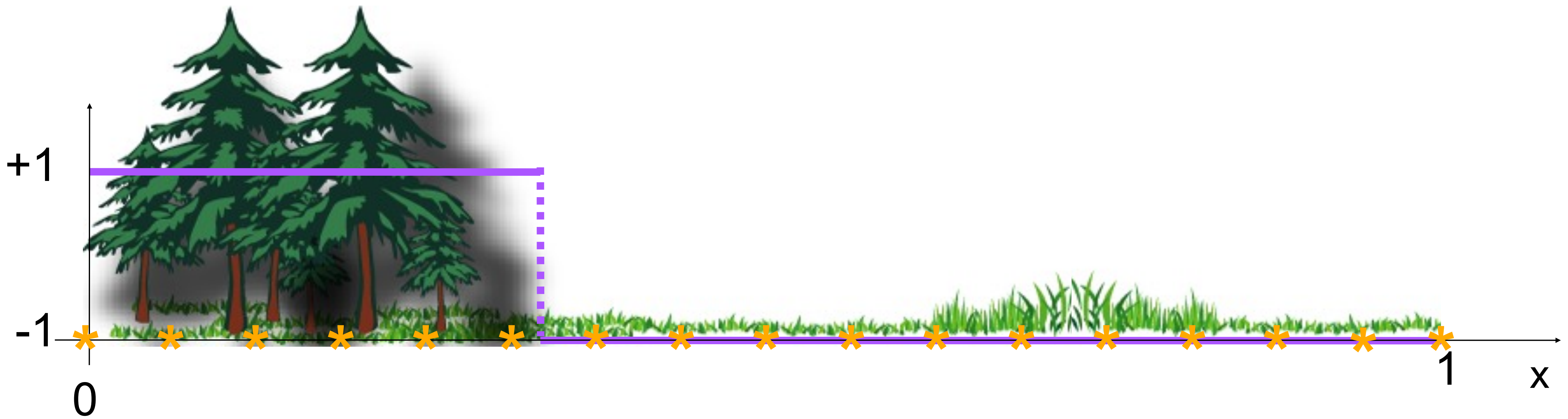
Where is it shady vs. sunny ?

A Stylized Environmental Sensing Task



Where is it shady vs. sunny ?

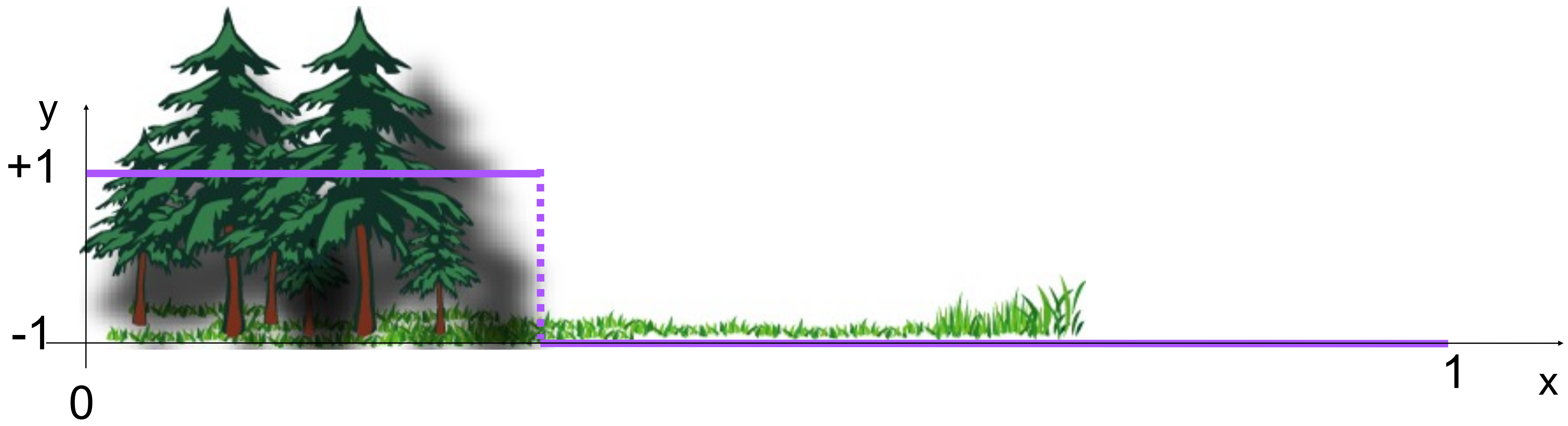
A Stylized Environmental Sensing Task



Where is it shady vs. sunny ?

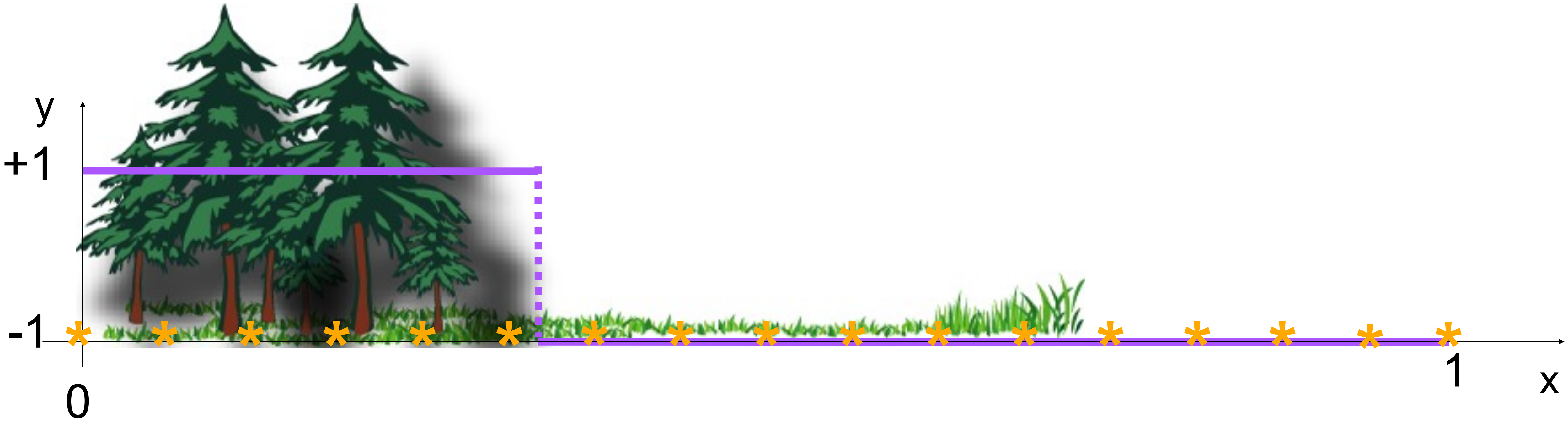
Suppose we have N wireless sensors. Do we need to query them all?

Classic Binary Search



Where is it shady vs. sunny ?

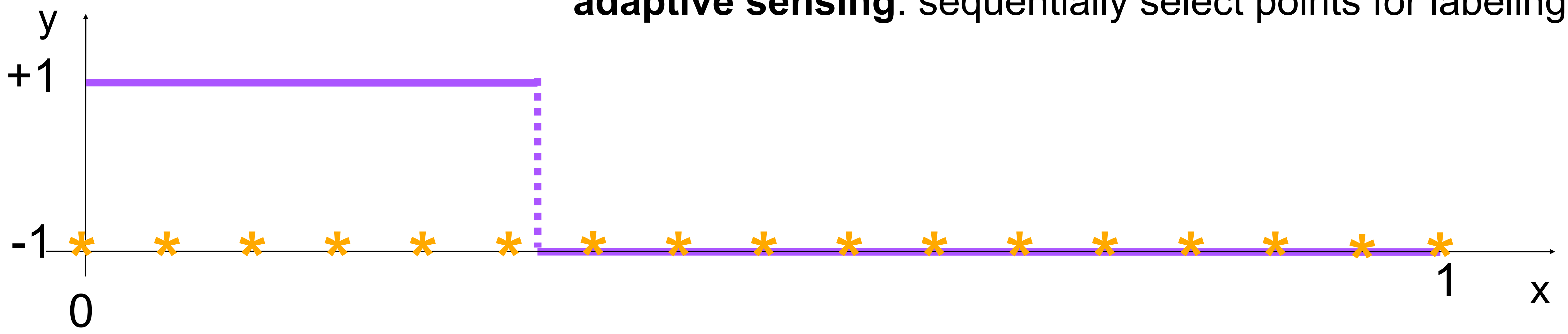
Classic Binary Search



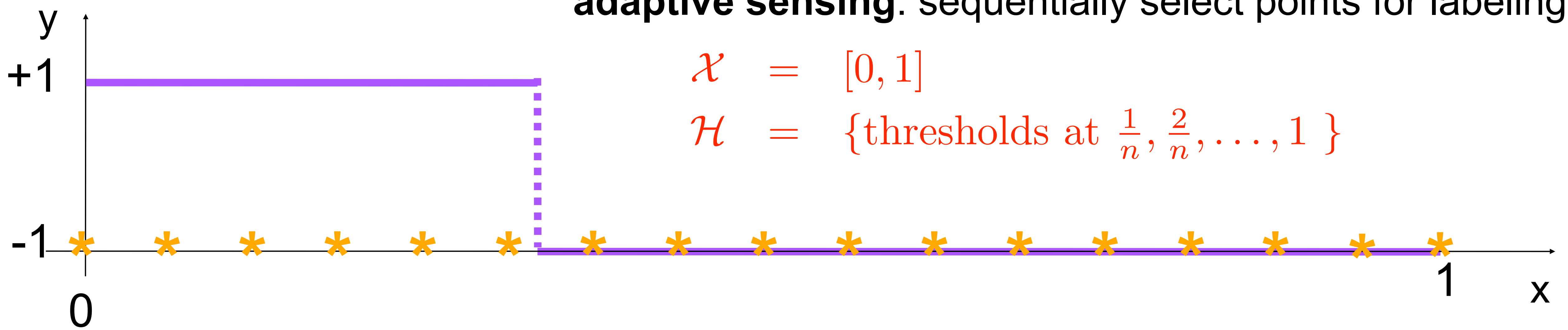
Where is it shady vs. sunny ?

Classic Binary Search

adaptive sensing: sequentially select points for labeling



Classic Binary Search



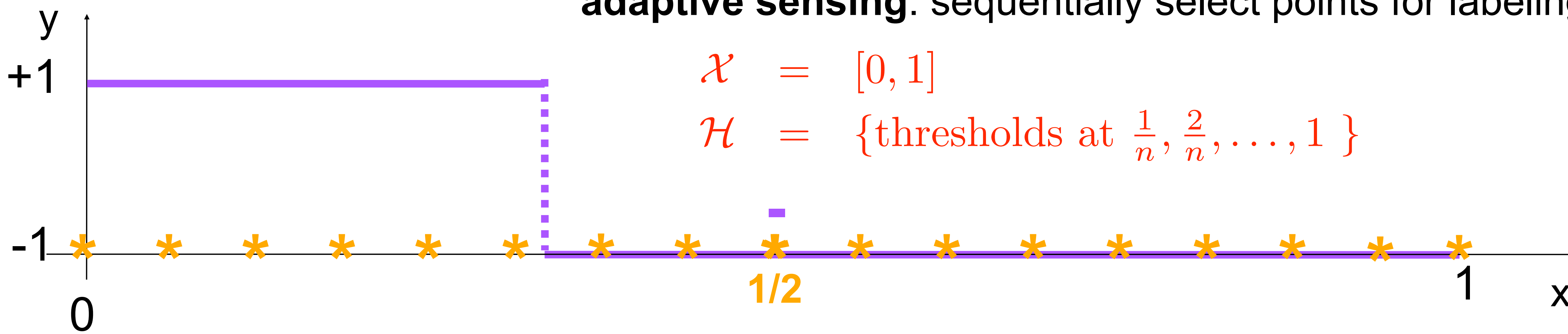
adaptive sensing: sequentially select points for labeling

$$\mathcal{X} = [0,1]$$
$$\mathcal{H} = \{\text{thresholds at } \tfrac{1}{n}, \tfrac{2}{n}, \dots, 1 \}$$

Classic Binary Search

adaptive sensing: sequentially select points for labeling

$$\mathcal{X} = [0,1]$$
$$\mathcal{H} = \{\text{thresholds at } \tfrac{1}{n}, \tfrac{2}{n}, \ldots, 1\ \}$$

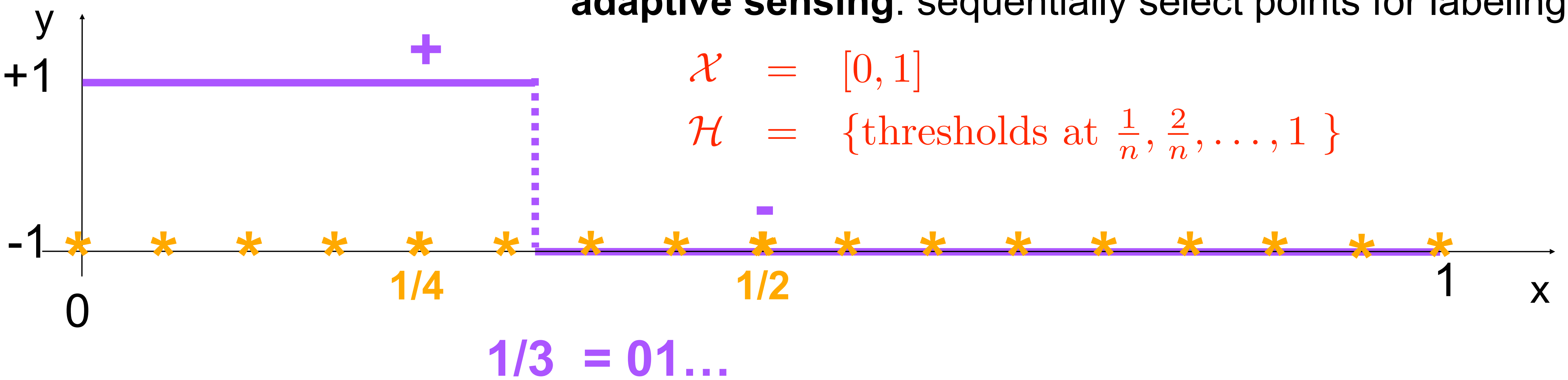


1/3 = 0…

Classic Binary Search

adaptive sensing: sequentially select points for labeling

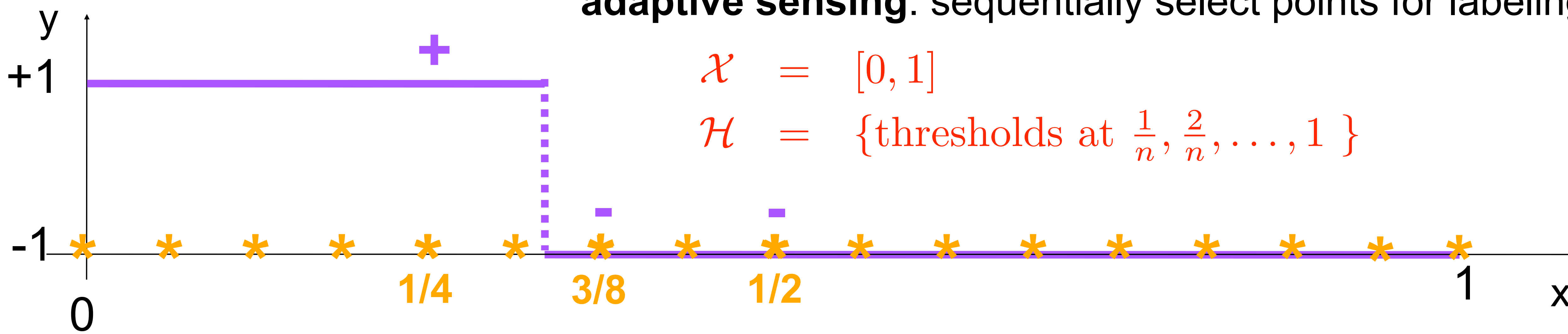
$$\mathcal{X} = [0,1]$$
$$\mathcal{H} = \{\text{thresholds at } \tfrac{1}{n}, \tfrac{2}{n}, \ldots, 1\ \}$$



Classic Binary Search

adaptive sensing: sequentially select points for labeling

$$\mathcal{X} = [0,1]$$
$$\mathcal{H} = \{\text{thresholds at } \tfrac{1}{n}, \tfrac{2}{n}, \ldots, 1 \ \}$$

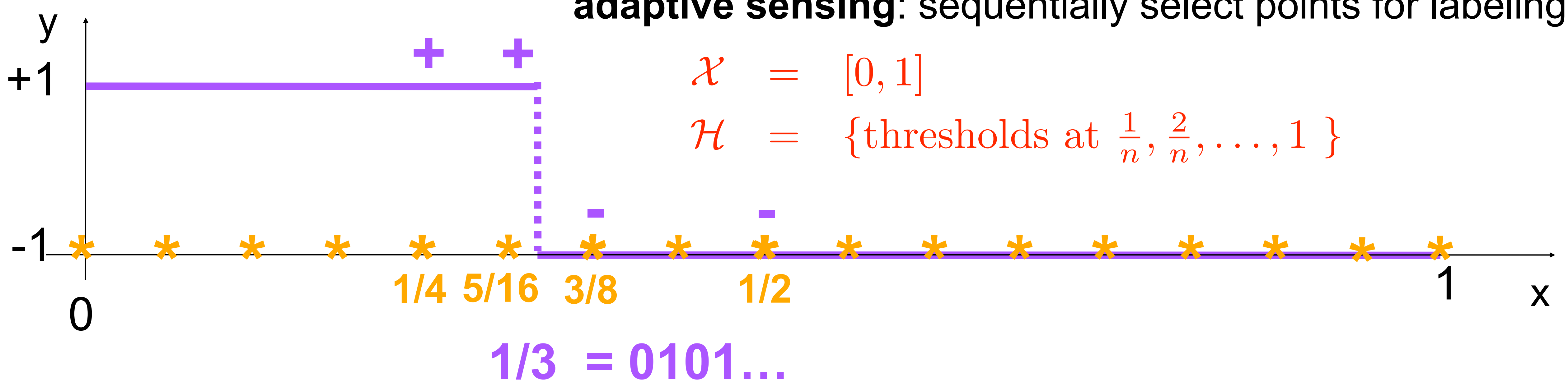


1/3 = 010…

Classic Binary Search

adaptive sensing: sequentially select points for labeling

$$\mathcal{X} = [0, 1]$$
$$\mathcal{H} = \{\text{thresholds at } \tfrac{1}{n}, \tfrac{2}{n}, \ldots, 1 \}$$

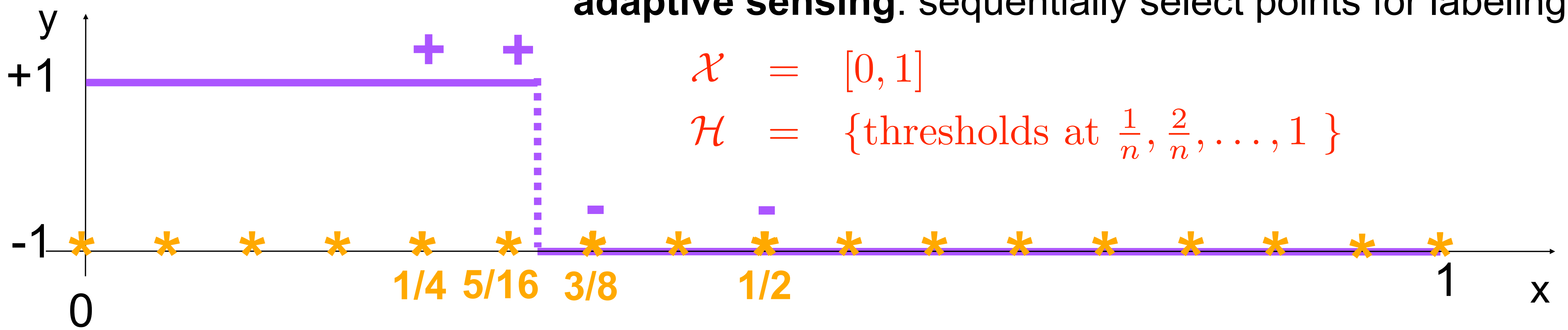


Classic Binary Search

adaptive sensing: sequentially select points for labeling

$$\mathcal{X} = [0,1]$$

$$\mathcal{H} = \{\text{thresholds at } \tfrac{1}{n}, \tfrac{2}{n}, \ldots, 1\ \}$$



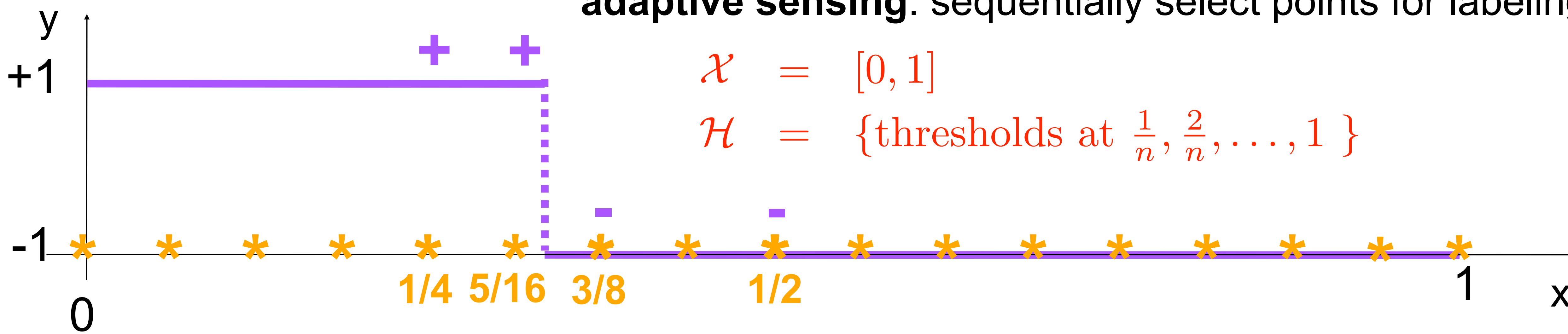
1/3 = 0101…

requires $\log_2 N$ queries

Classic Binary Search

adaptive sensing: sequentially select points for labeling

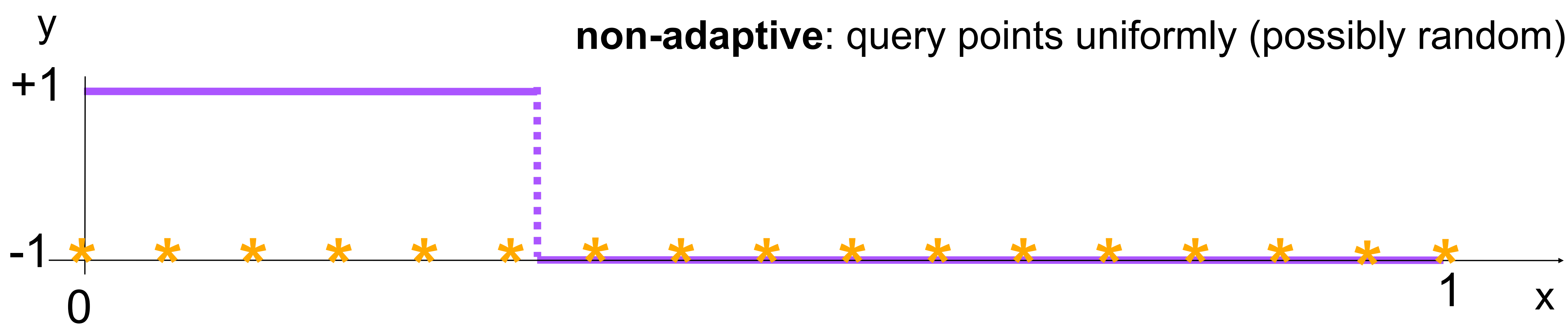
$$\mathcal{X} = [0,1]$$
$$\mathcal{H} = \{\text{thresholds at } \tfrac{1}{n}, \tfrac{2}{n}, \ldots, 1\ \}$$



1/3 = 0101…

requires $\log_2 N$ queries

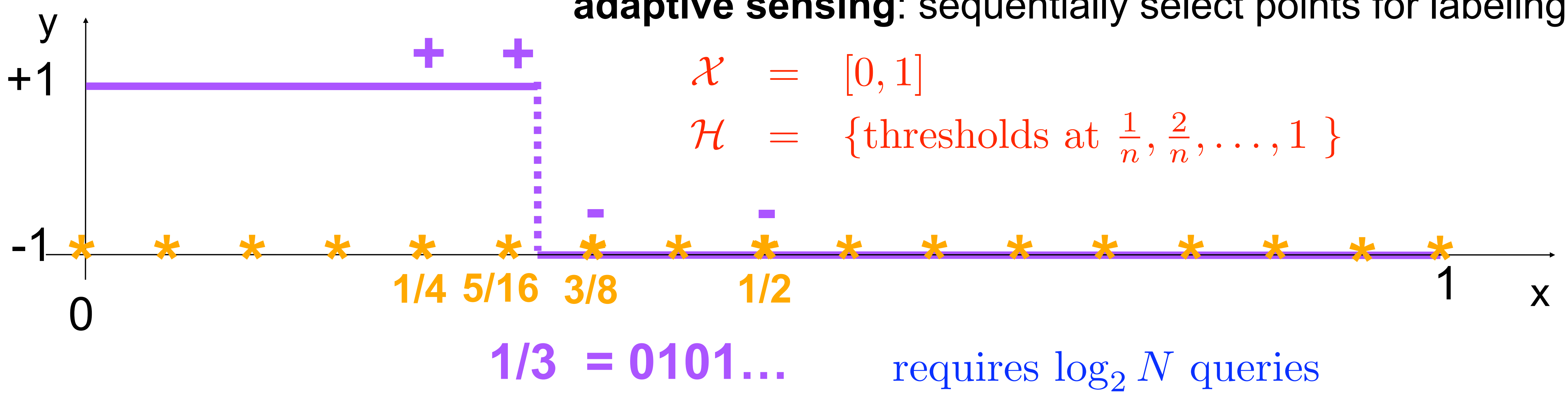
non-adaptive: query points uniformly (possibly random)



Classic Binary Search

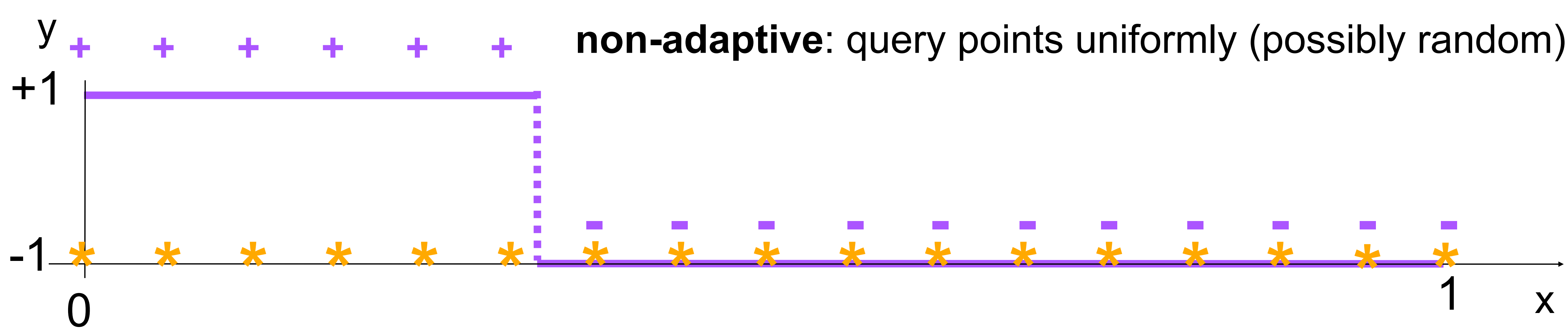
adaptive sensing: sequentially select points for labeling

$$\mathcal{X} = [0,1]$$
$$\mathcal{H} = \{\text{thresholds at } \tfrac{1}{n}, \tfrac{2}{n}, \ldots, 1\ \}$$



requires $\log_2 N$ queries

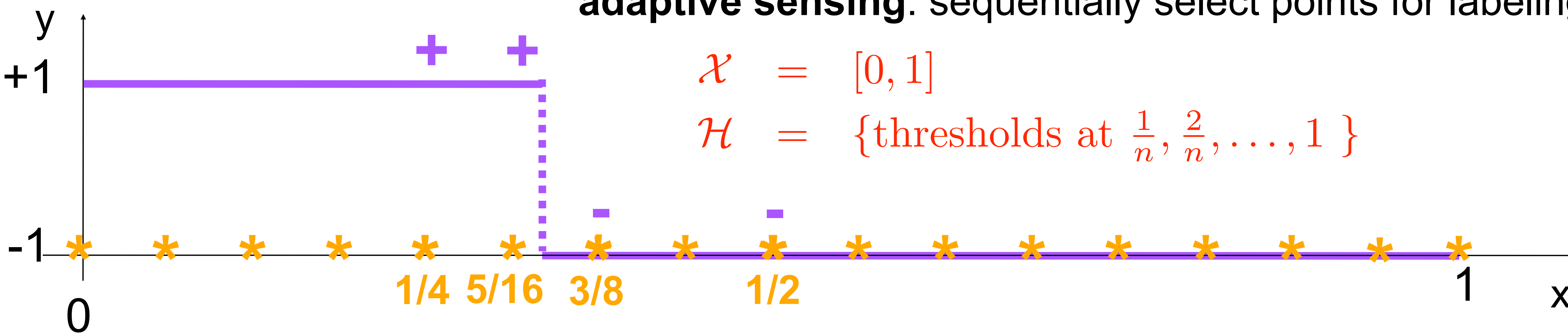
non-adaptive: query points uniformly (possibly random)



Classic Binary Search

adaptive sensing: sequentially select points for labeling

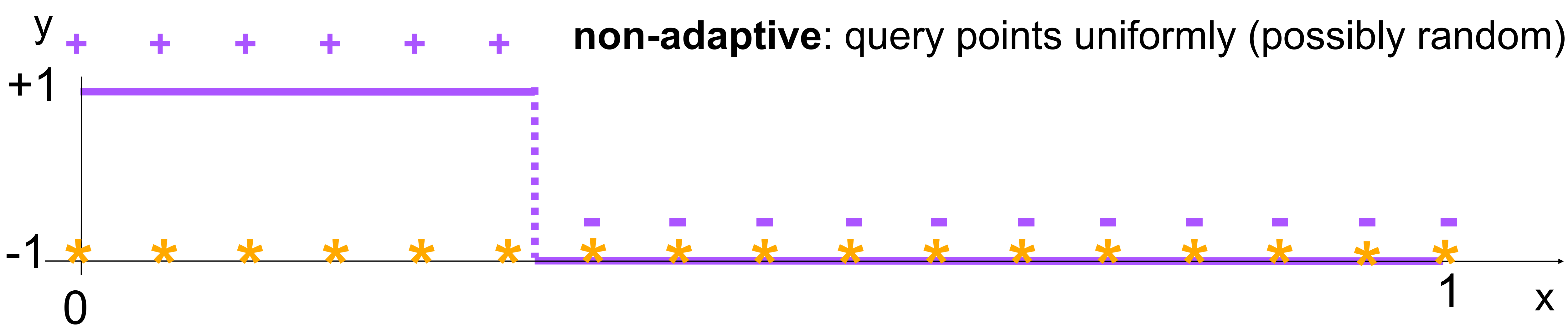
$$\mathcal{X} = [0,1]$$
$$\mathcal{H} = \{\text{thresholds at } \tfrac{1}{n}, \tfrac{2}{n}, \ldots, 1\ \}$$



1/3 = 0101…

requires $\log_2 N$ queries

non-adaptive: query points uniformly (possibly random)

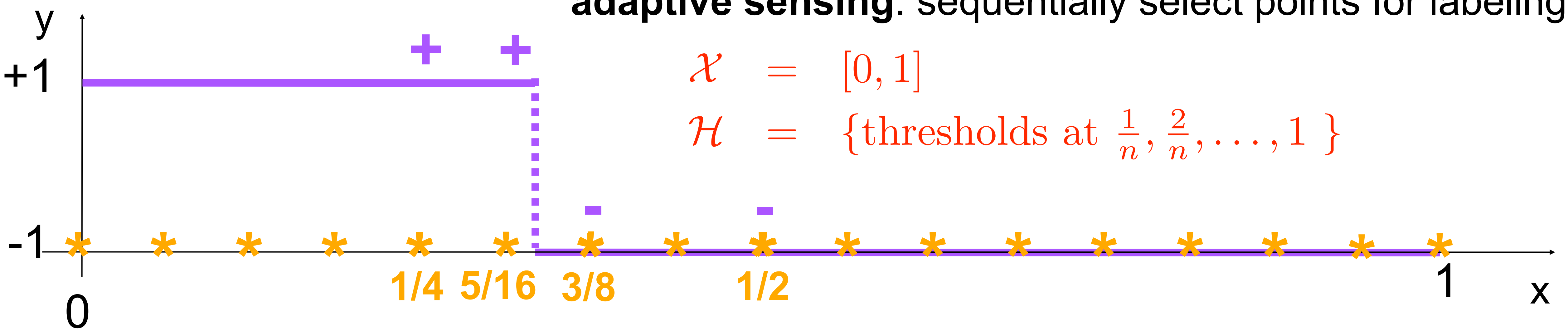


requires $O(N)$ queries

Classic Binary Search

adaptive sensing: sequentially select points for labeling

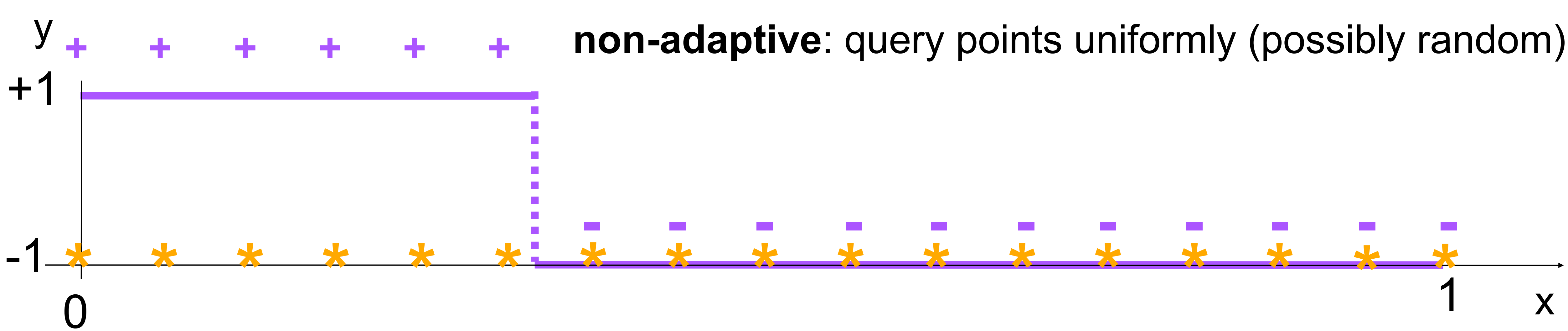
$$\mathcal{X} = [0,1]$$
$$\mathcal{H} = \{\text{thresholds at } \tfrac{1}{n}, \tfrac{2}{n}, \ldots, 1\ \}$$



1/3 = 0101…

requires $\log_2 N$ queries

non-adaptive: query points uniformly (possibly random)



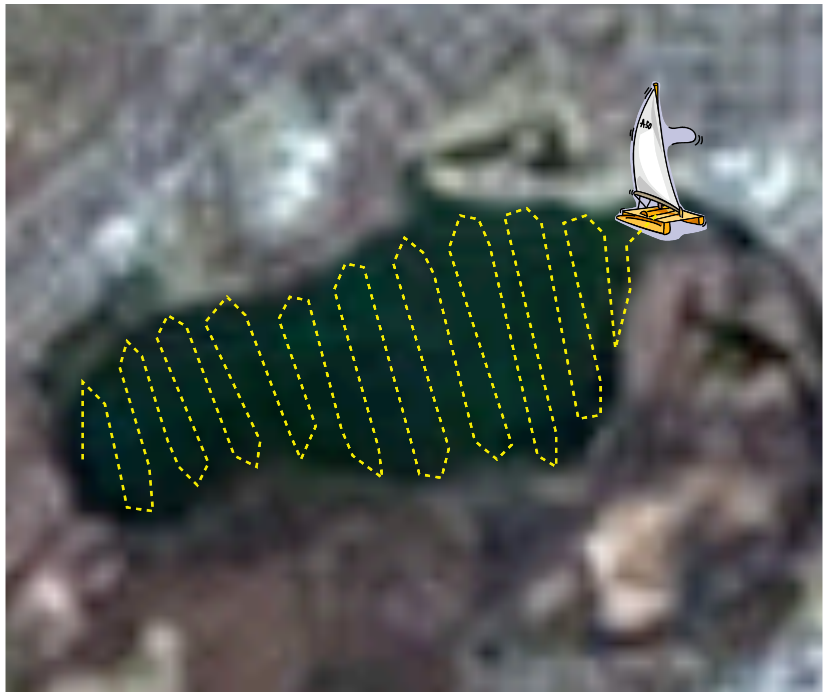
requires $O(N)$ queries

adaptive sensing is dramatically more efficient

Environmental Sensing

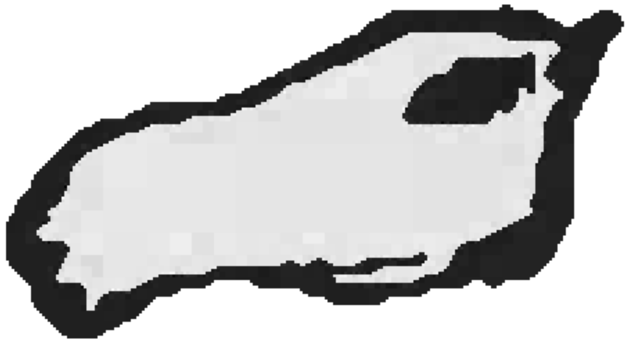
Chin Wu, Civil & Environmental Engr.
http://limnology.wisc.edu/

Lake Wingra, Madison WI



acoustic doppler sensing of
water current in Lake Wingra

water current velocity map
(darker = high velocity)



classification into high-
and low-velocity regions

Non-adaptive Survey: 48 hrs
Adaptive Survey: 14 hrs

A. Singh, R. Nowak and P. Ramanathan. **Active Learning for Adaptive Mobile Sensing Networks**. *ACM/IEEE Interntional Conference on Information Processing in Sensor Networks, IPSN 2006*.

Outline of Part 3

Outline of Part 3

Noisy Binary Search: What if the expert/oracle responses are not completely reliable ?

Outline of Part 3

Noisy Binary Search: What if the expert/oracle responses are not completely reliable ?

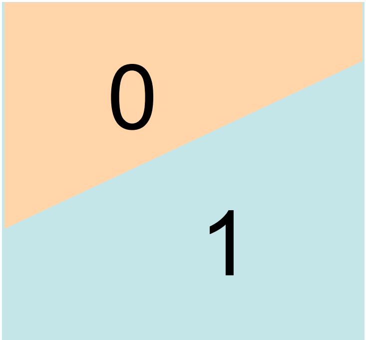
Minimax Analysis of Active Learning: What are the fundamental capabilities and limits of active learning ?

Outline of Part 3

Noisy Binary Search: What if the expert/oracle responses are not completely reliable ?

Minimax Analysis of Active Learning: What are the fundamental capabilities and limits of active learning ?

Generalized Binary Search: Can binary search be generalized in order to learn more complex decision rules ?

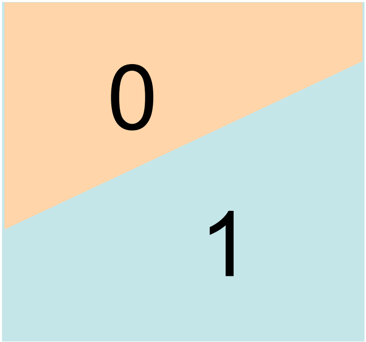


Outline of Part 3

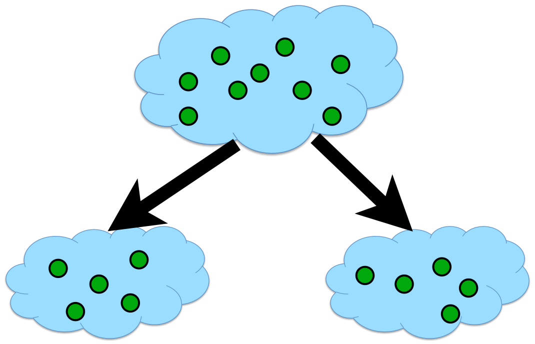
Noisy Binary Search: What if the expert/oracle responses are not completely reliable ?

Minimax Analysis of Active Learning: What are the fundamental capabilities and limits of active learning ?

Generalized Binary Search: Can binary search be generalized in order to learn more complex decision rules ?

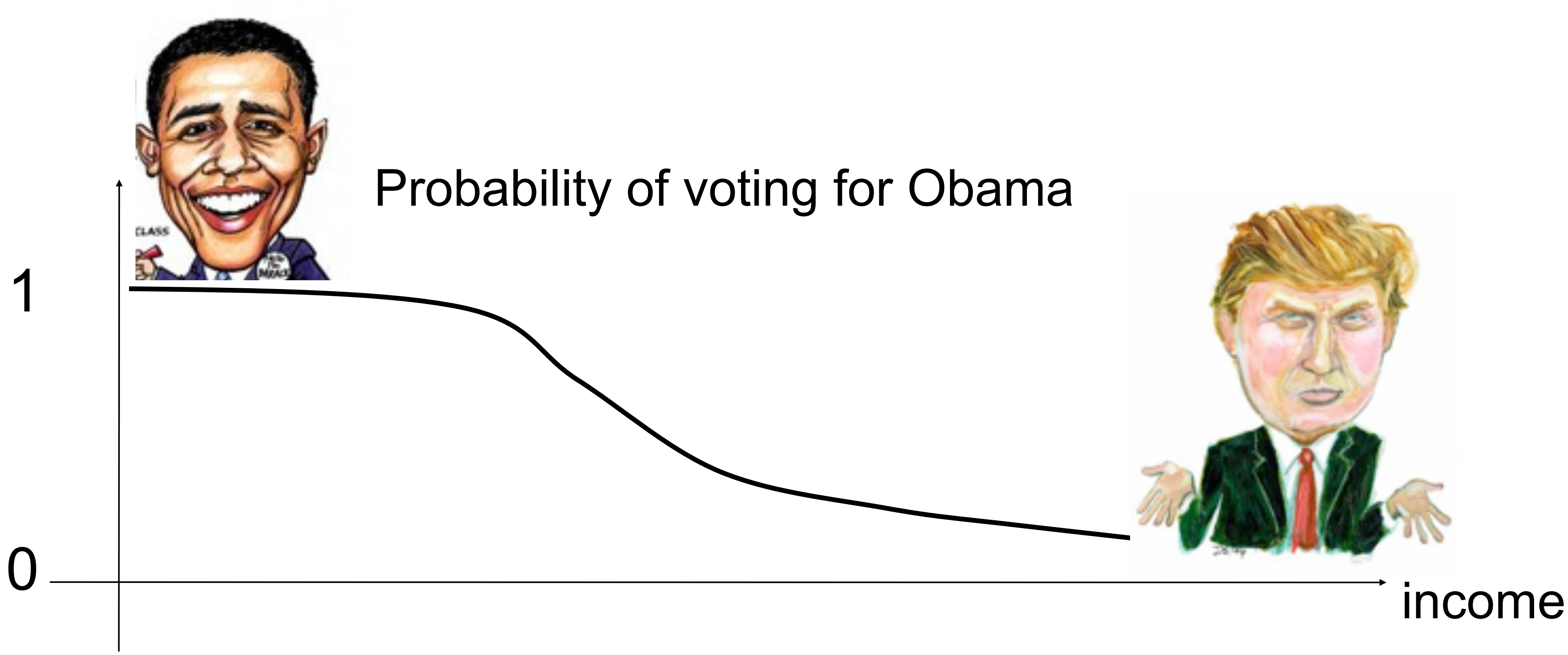


Unsupervised Active Learning: Can active learning help in unsupervised learning problems such as clustering ?



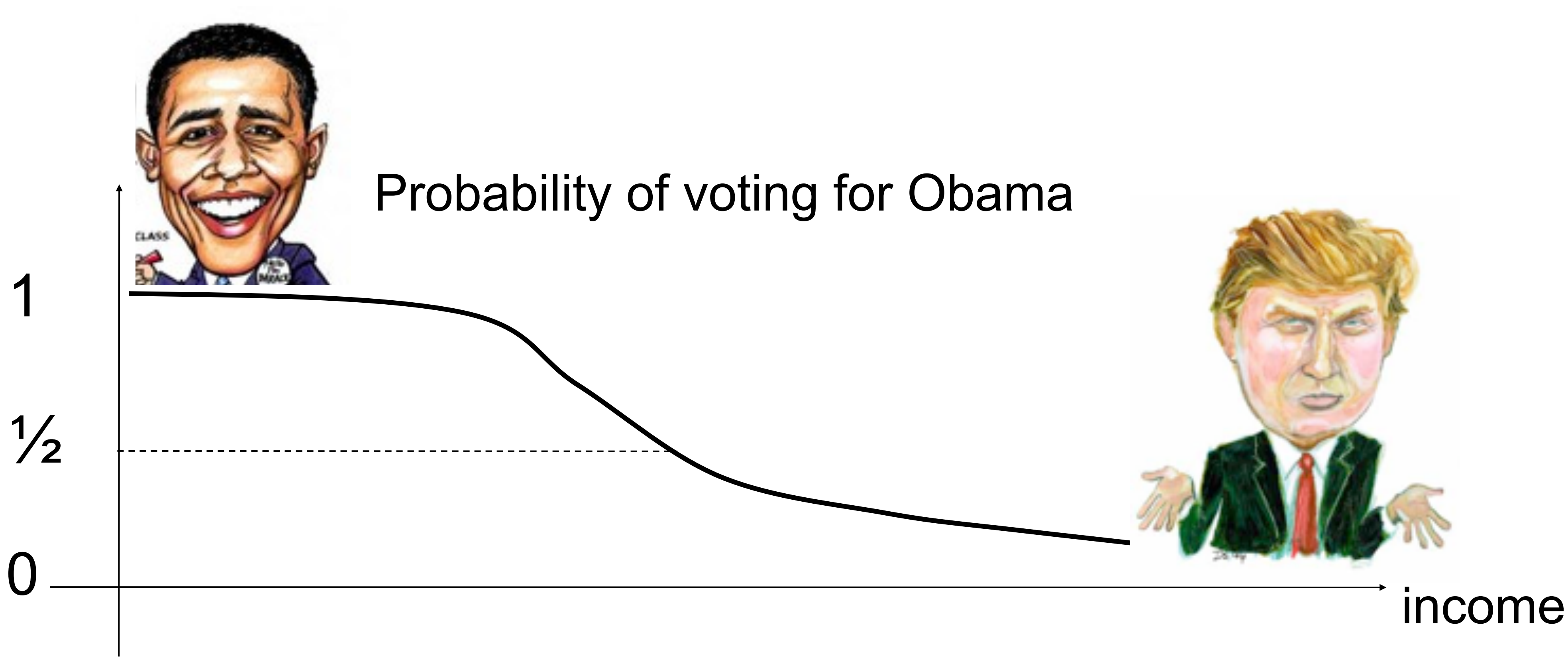
Binary Search and Noise

At what income level is a person more likely to be Republican vs. Democrat ?



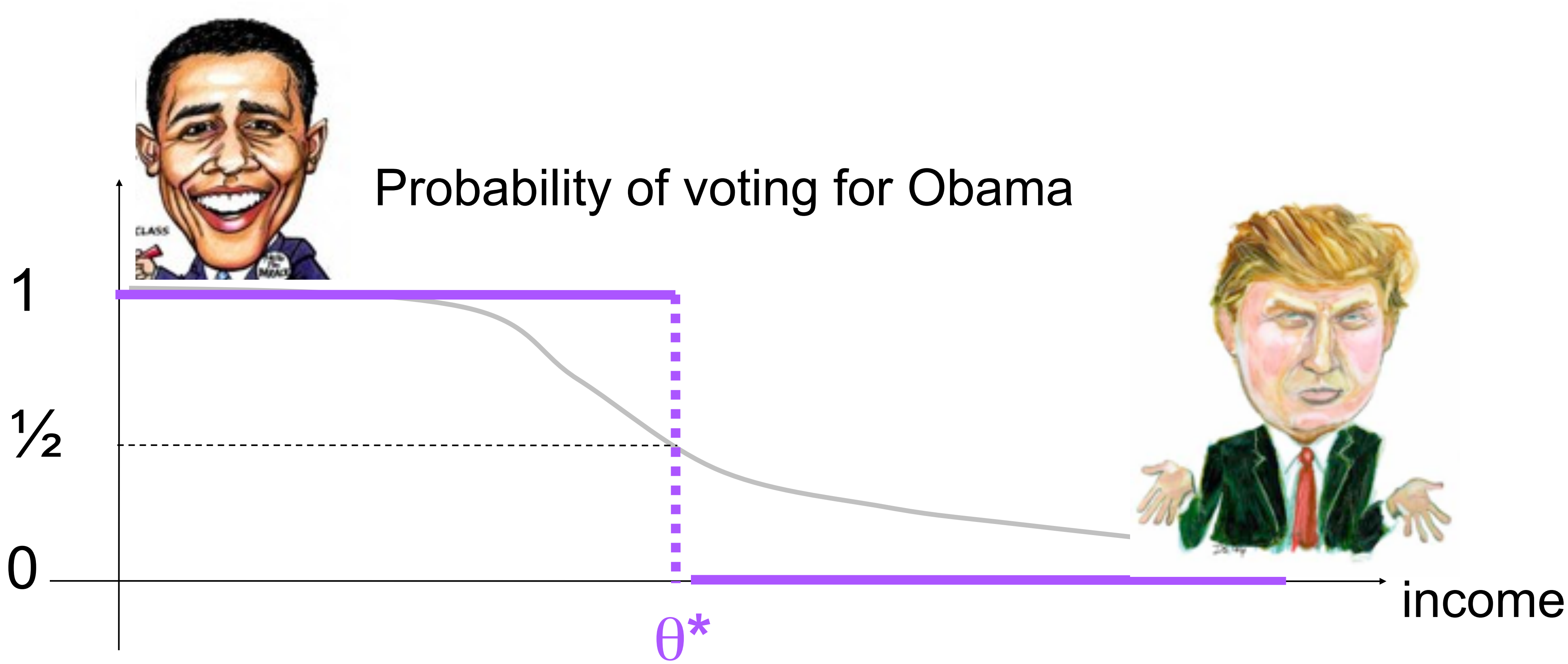
Binary Search and Noise

At what income level is a person more likely to be Republican vs. Democrat ?



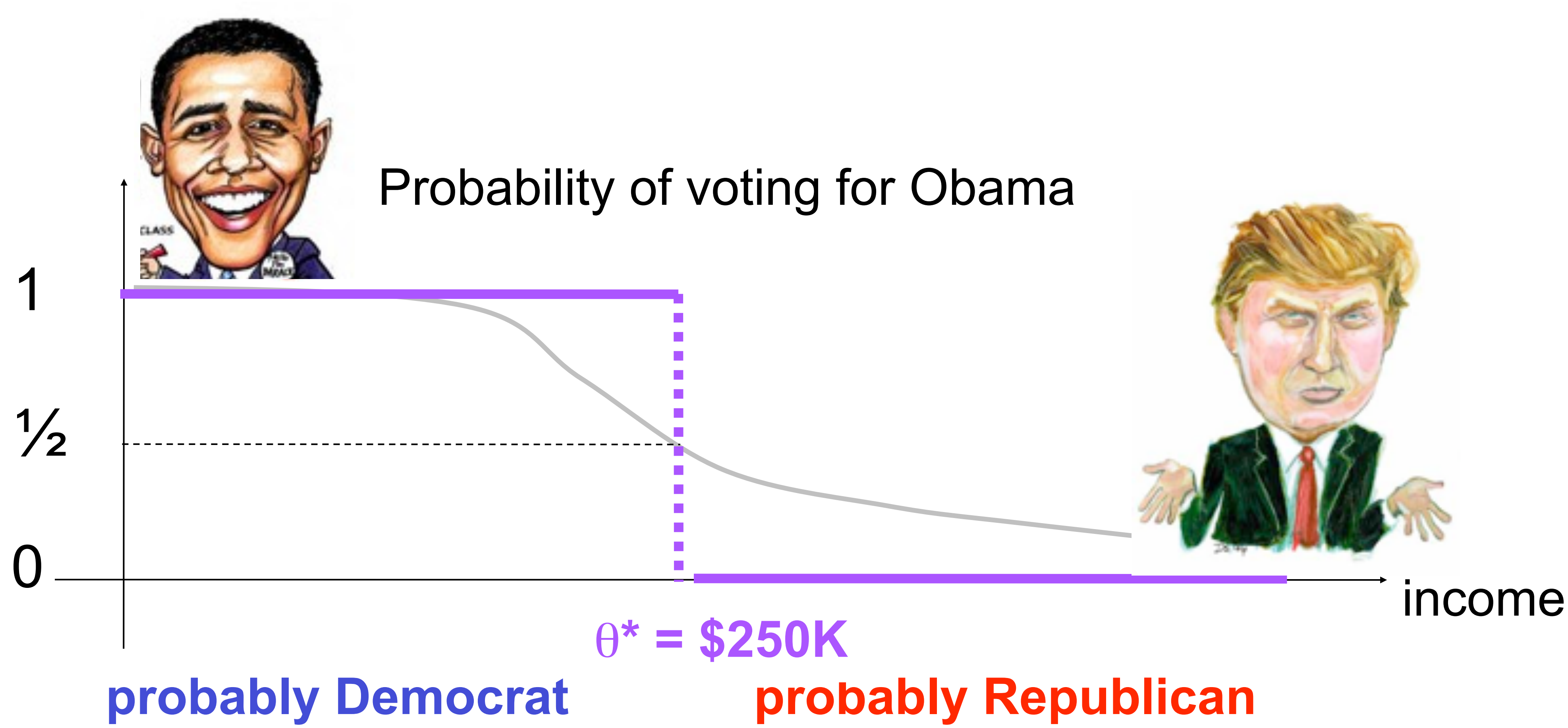
Binary Search and Noise

At what income level is a person more likely to be Republican vs. Democrat ?

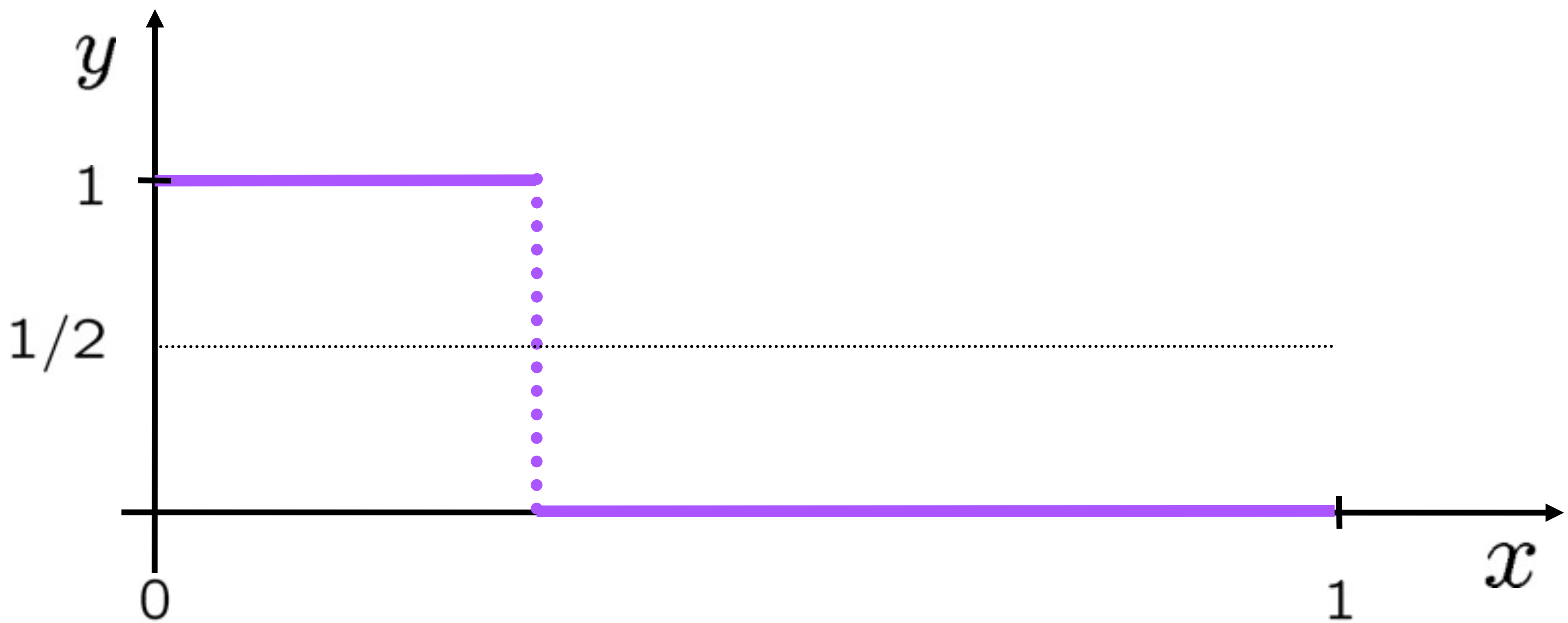


Binary Search and Noise

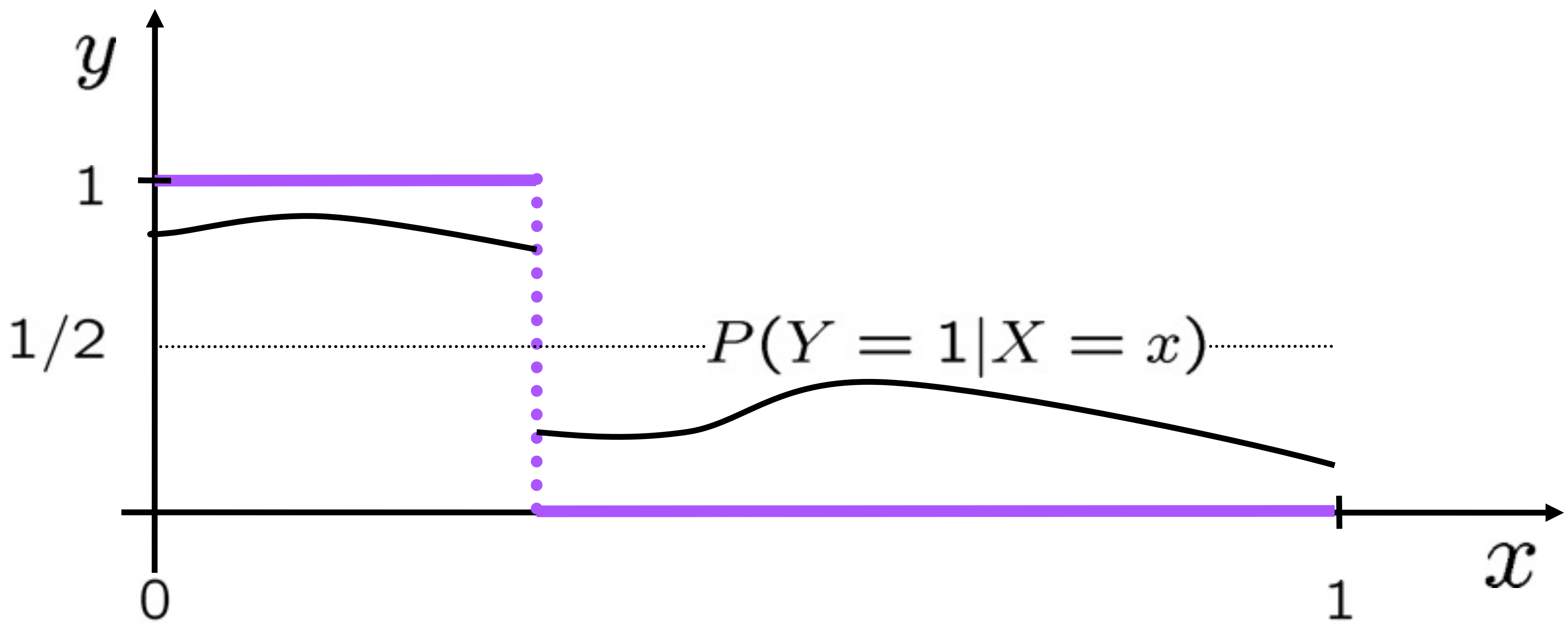
At what income level is a person more likely to be Republican vs. Democrat ?



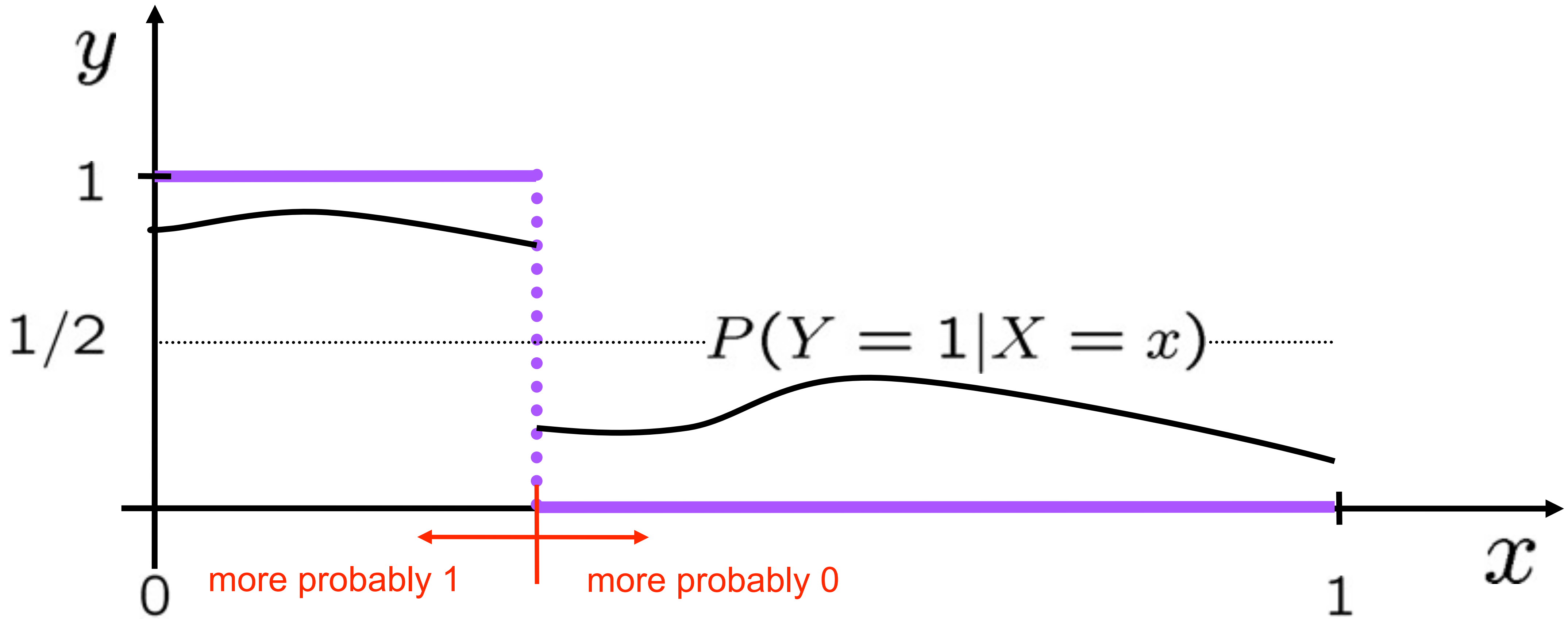
Bounded and Unbounded Noise



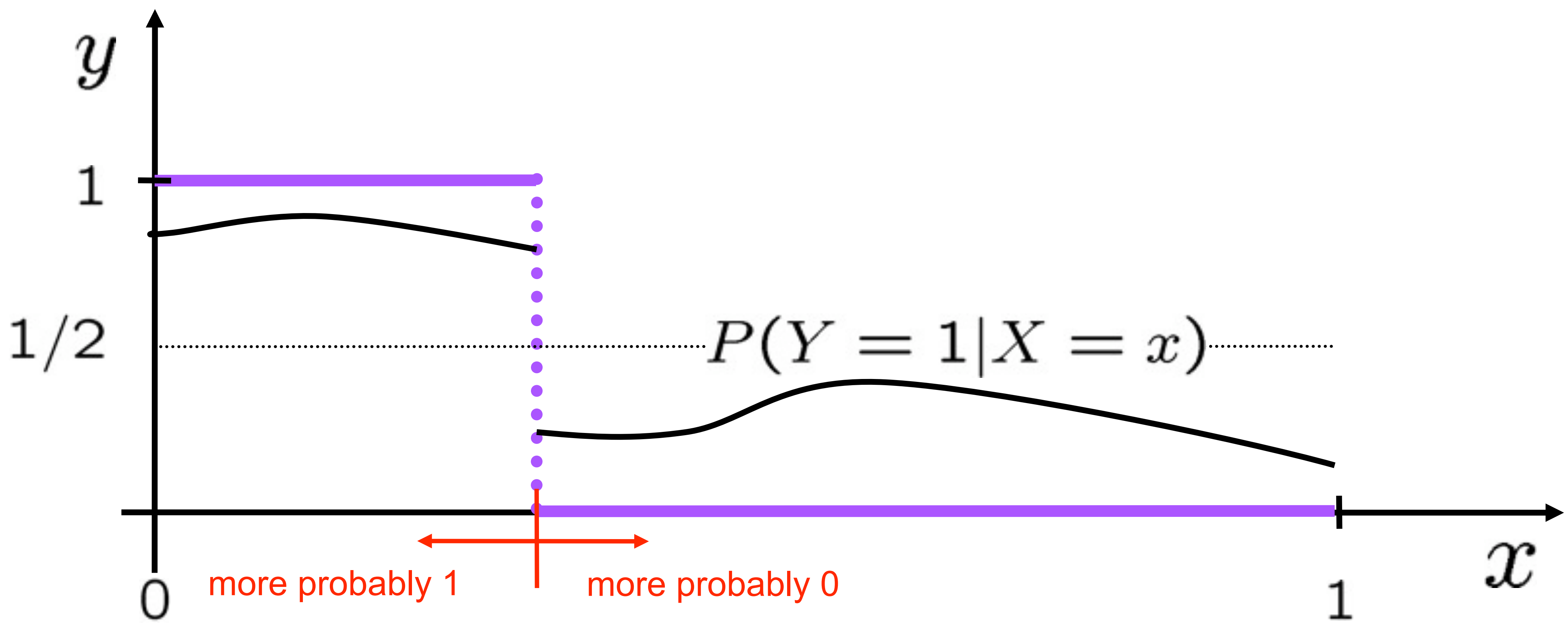
Bounded and Unbounded Noise



Bounded and Unbounded Noise

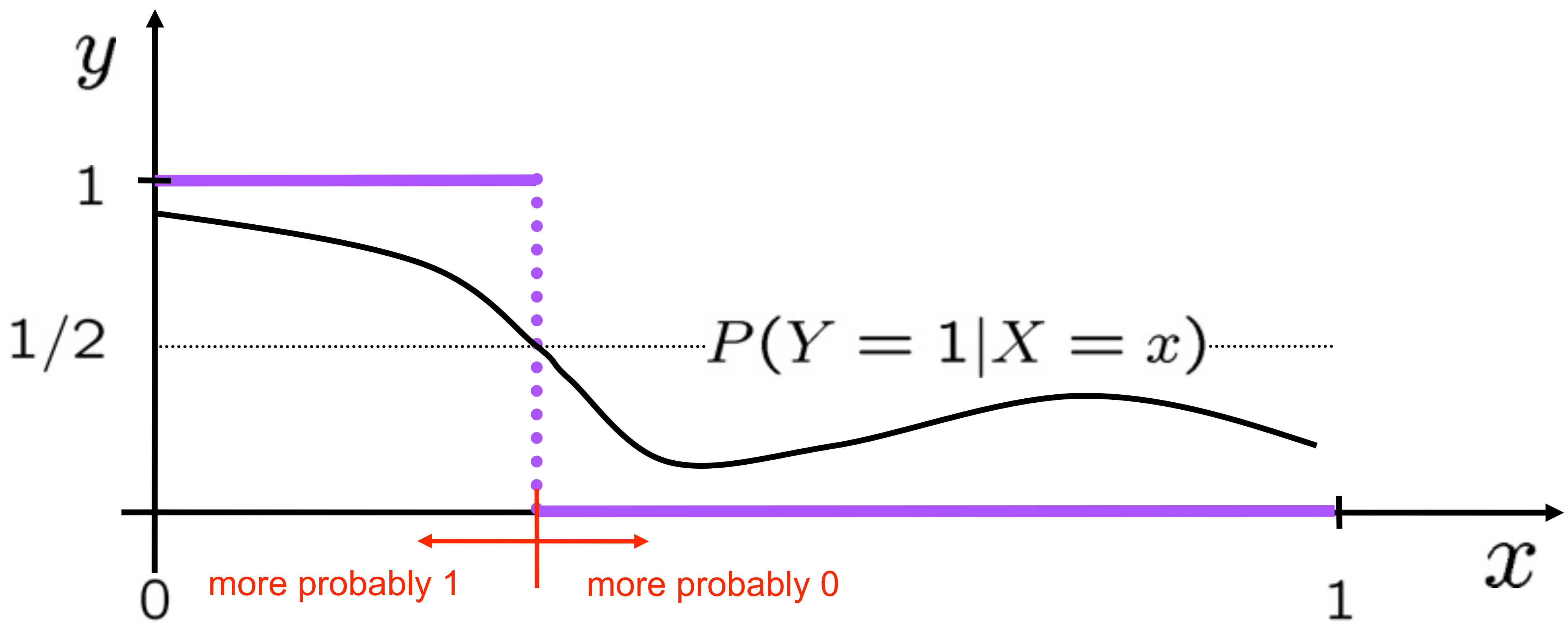


Bounded and Unbounded Noise



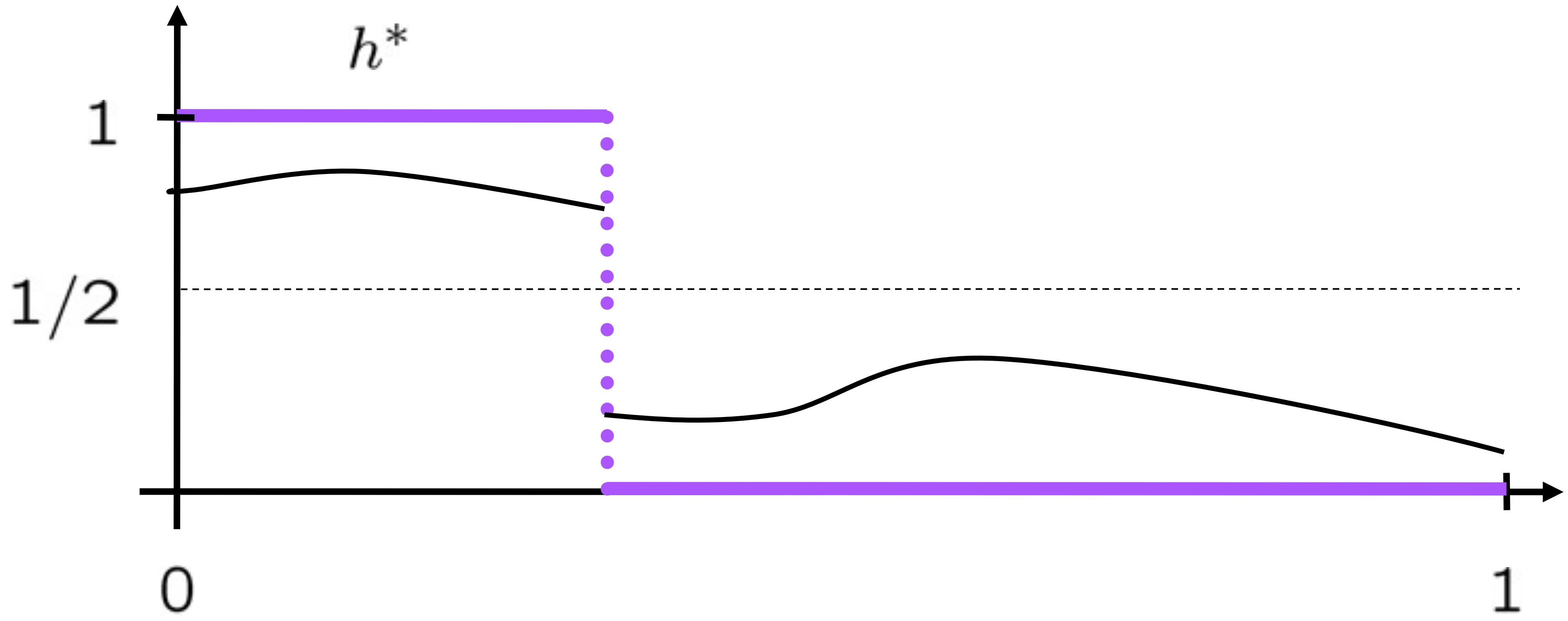
“bounded noise” : strictly more/less probably 1 at all locations

Bounded and Unbounded Noise

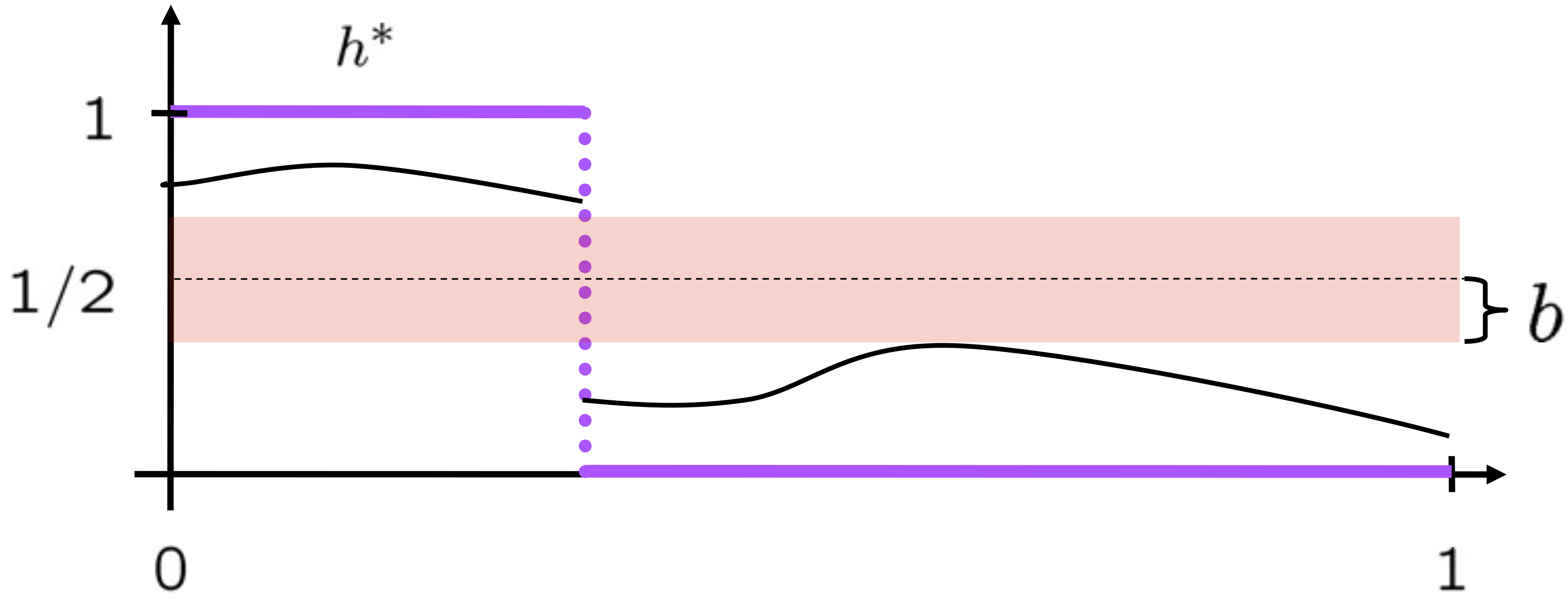


"unbounded noise" : like the toss of a fair coin at threshold

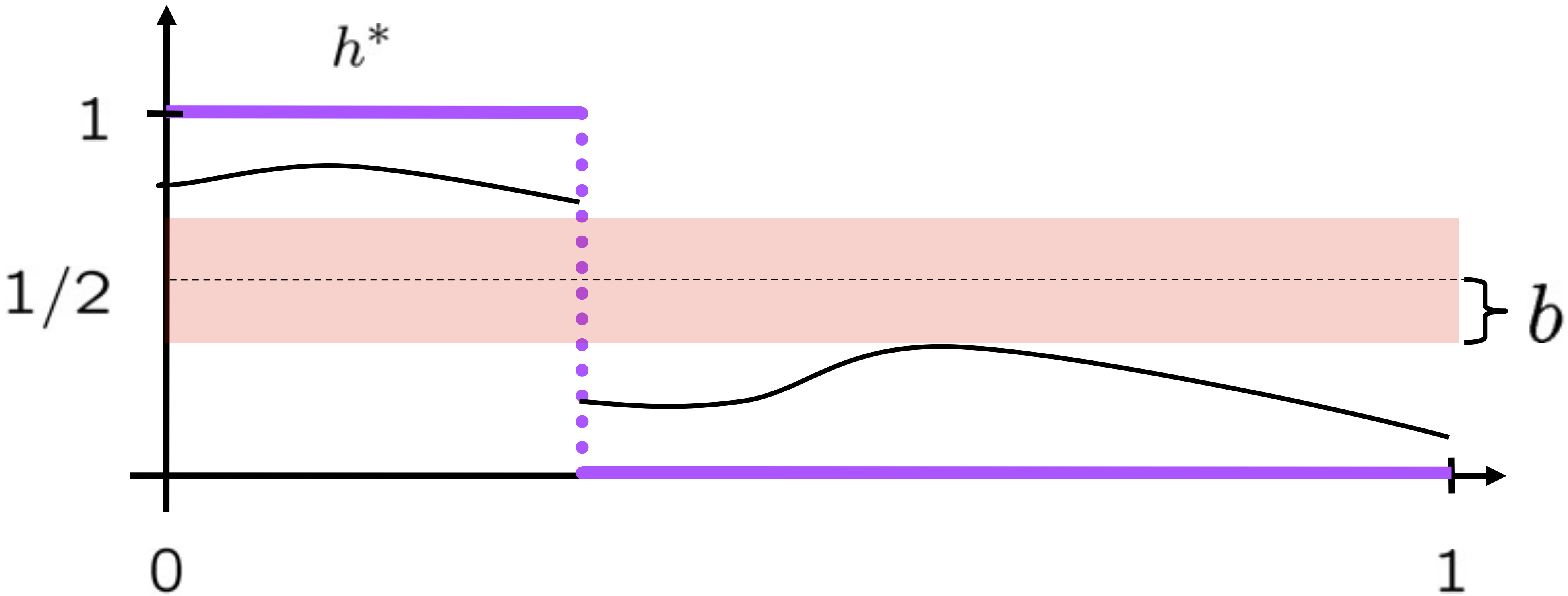
Horstein’s Multiplicative Weighting Method

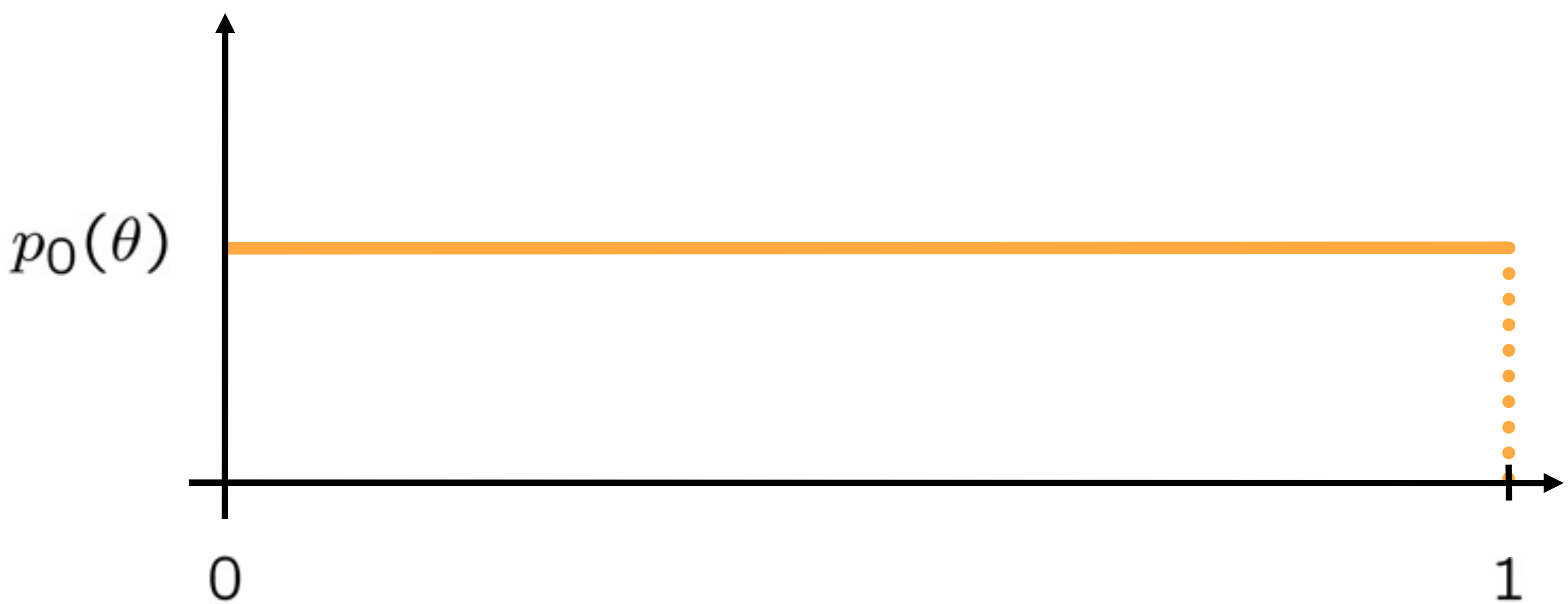


Horstein's Multiplicative Weighting Method

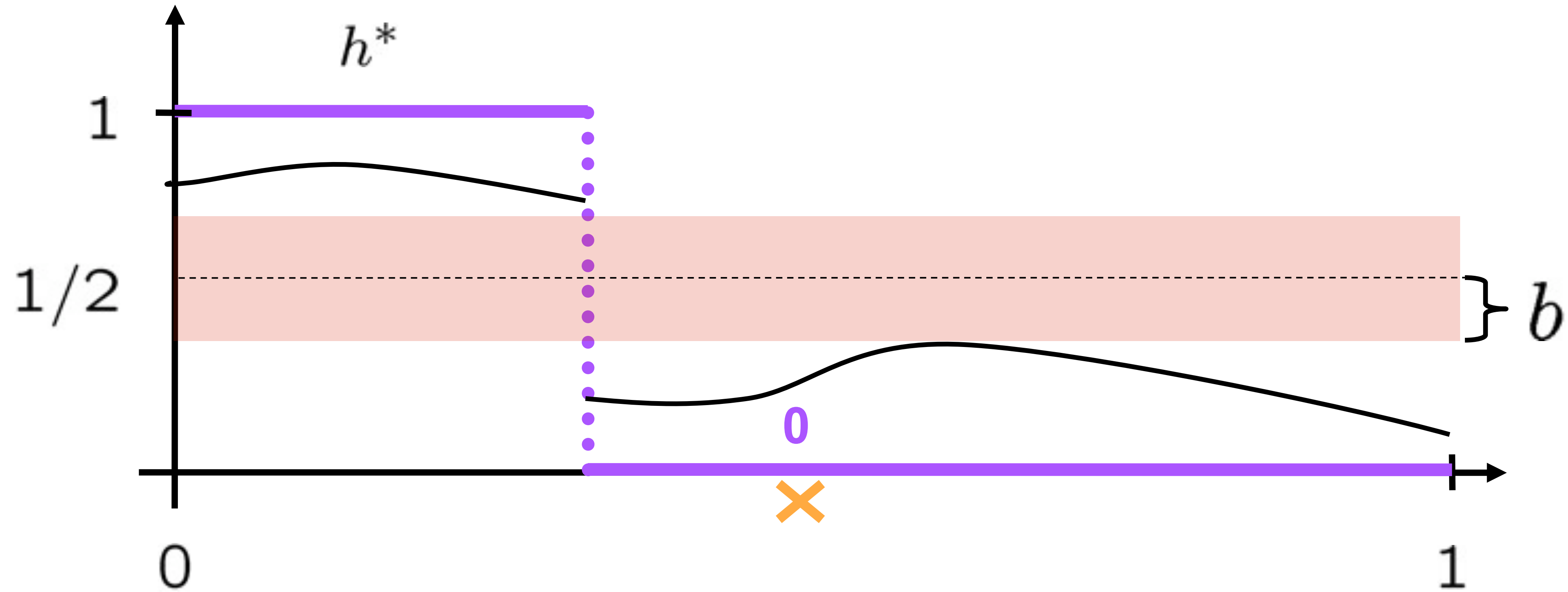


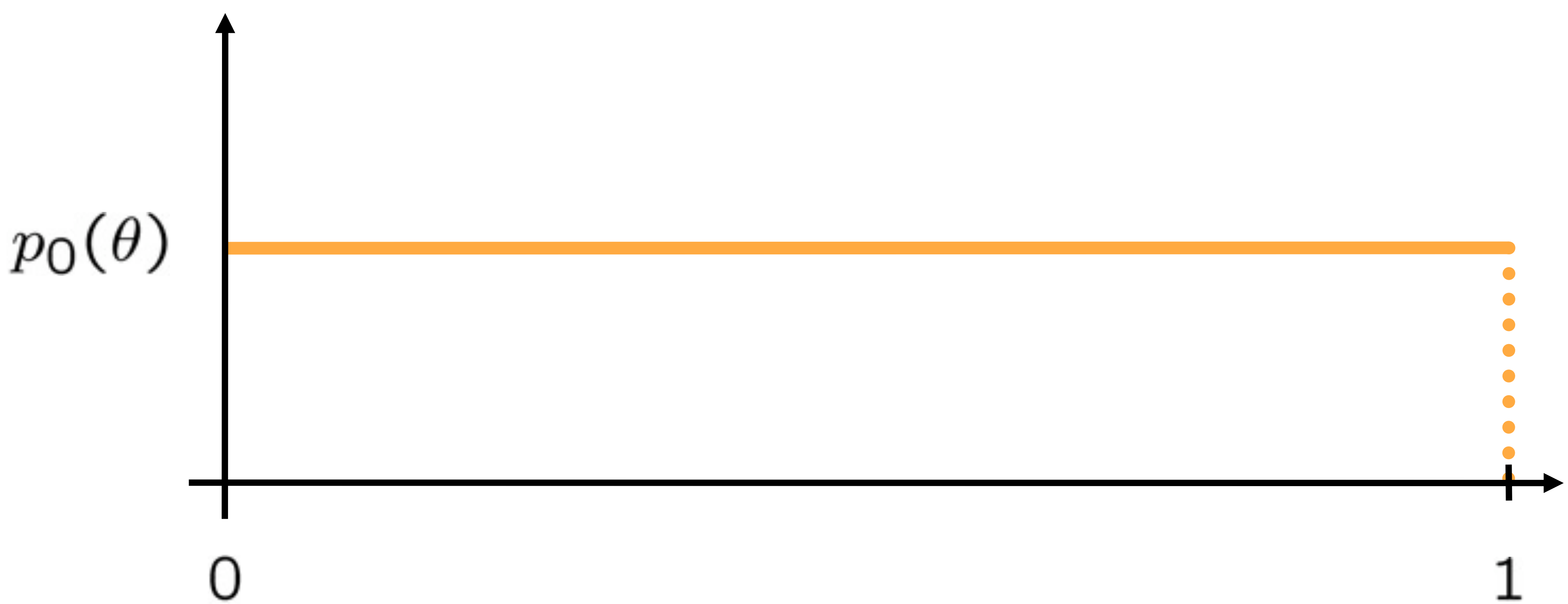
Horstein's Multiplicative Weighting Method



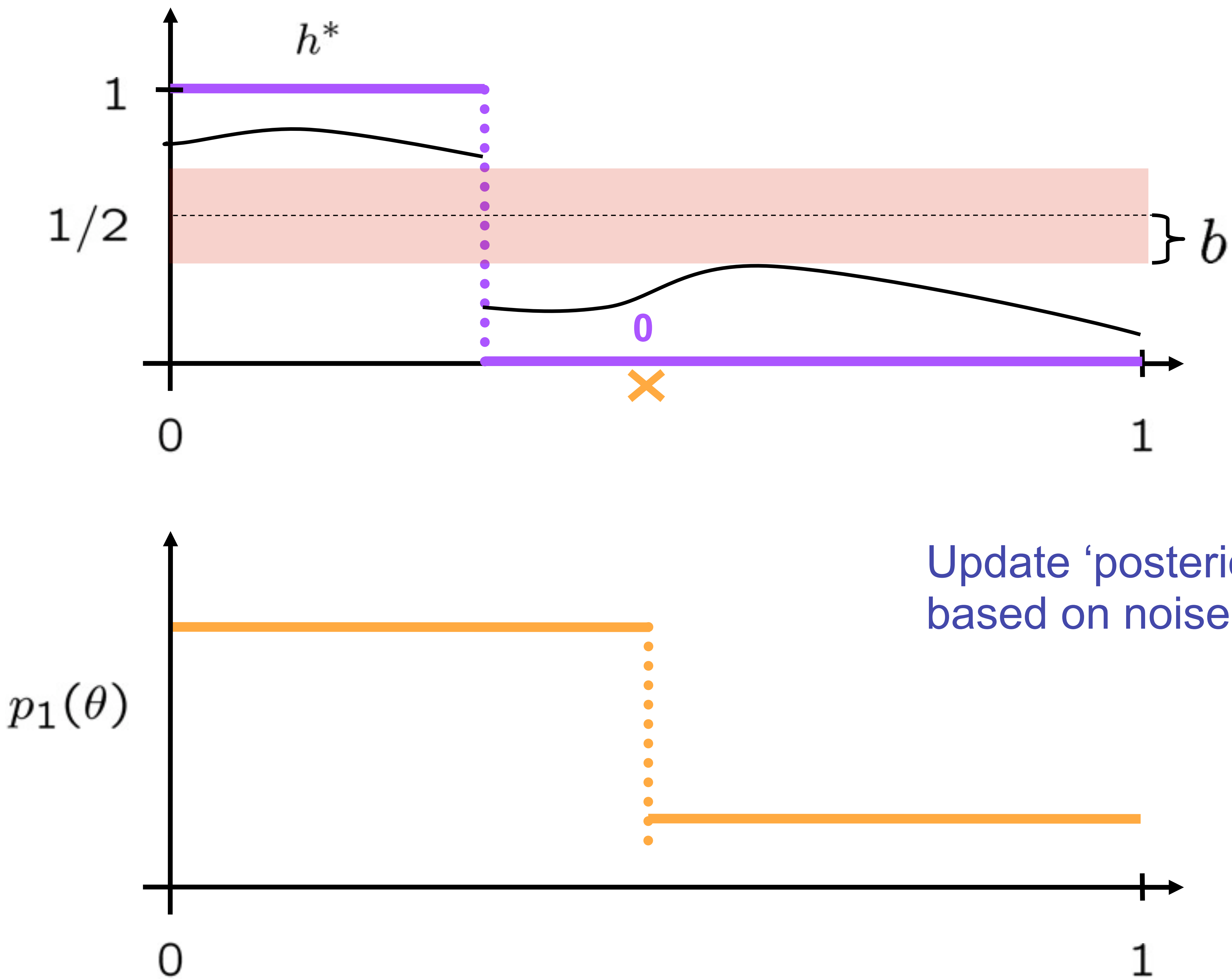


Horstein's Multiplicative Weighting Method



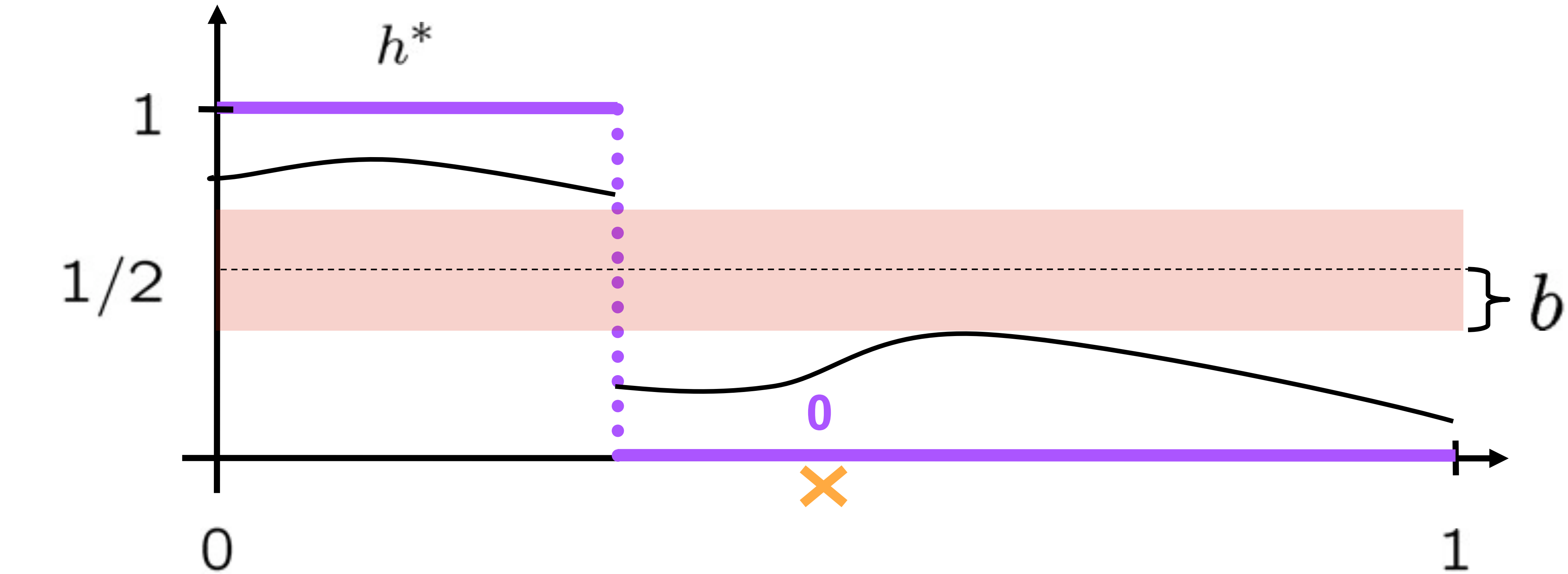


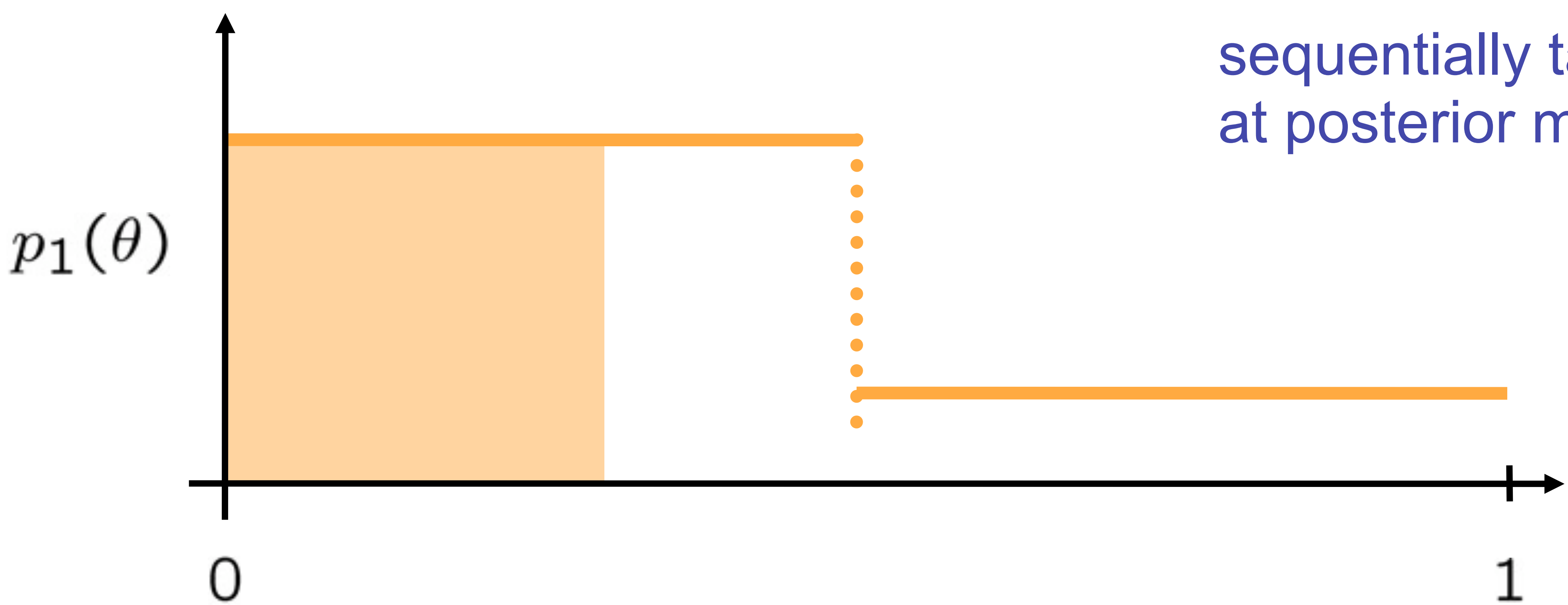
Horstein's Multiplicative Weighting Method



Update 'posterior' density based on noise bound b

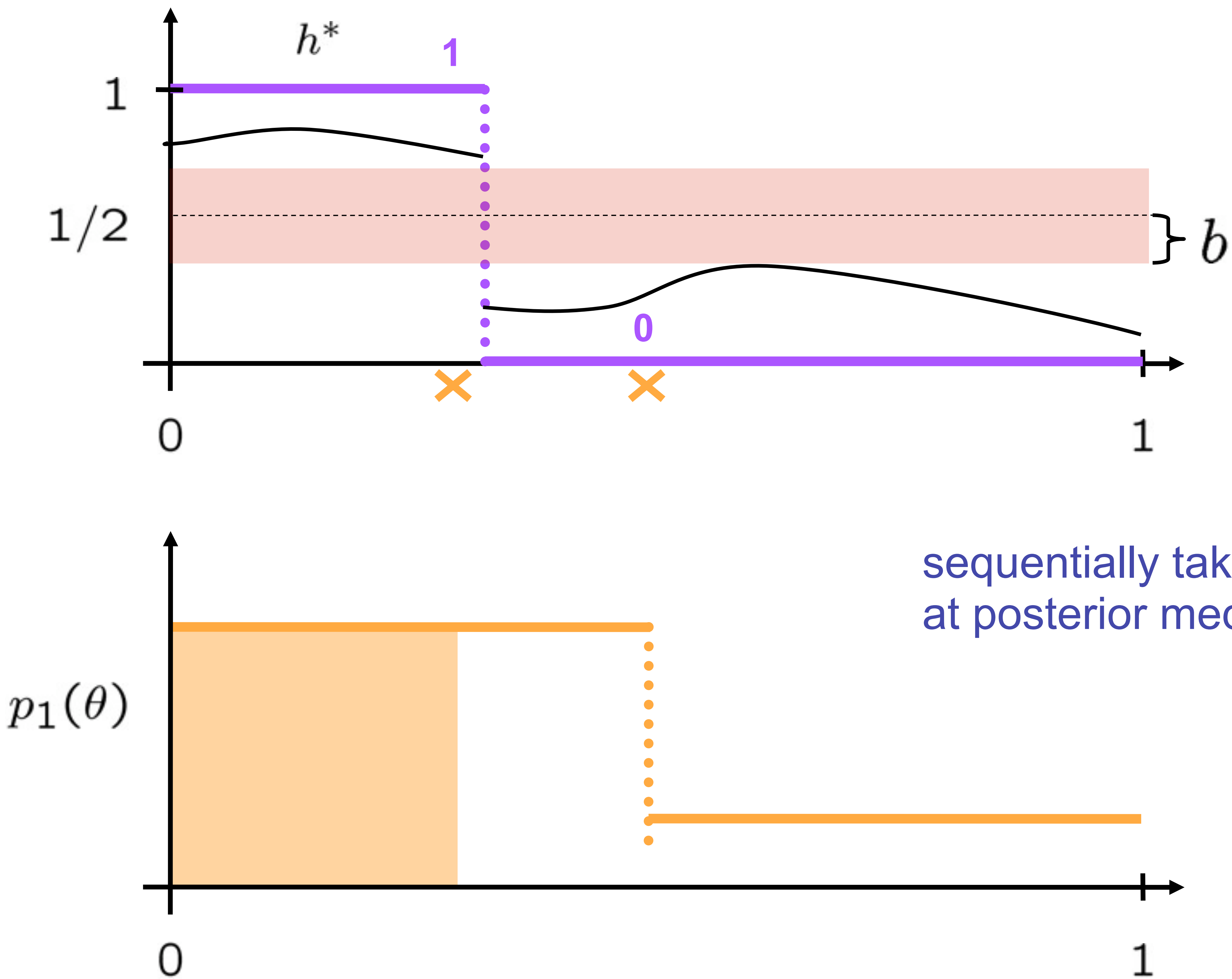
Horstein's Multiplicative Weighting Method





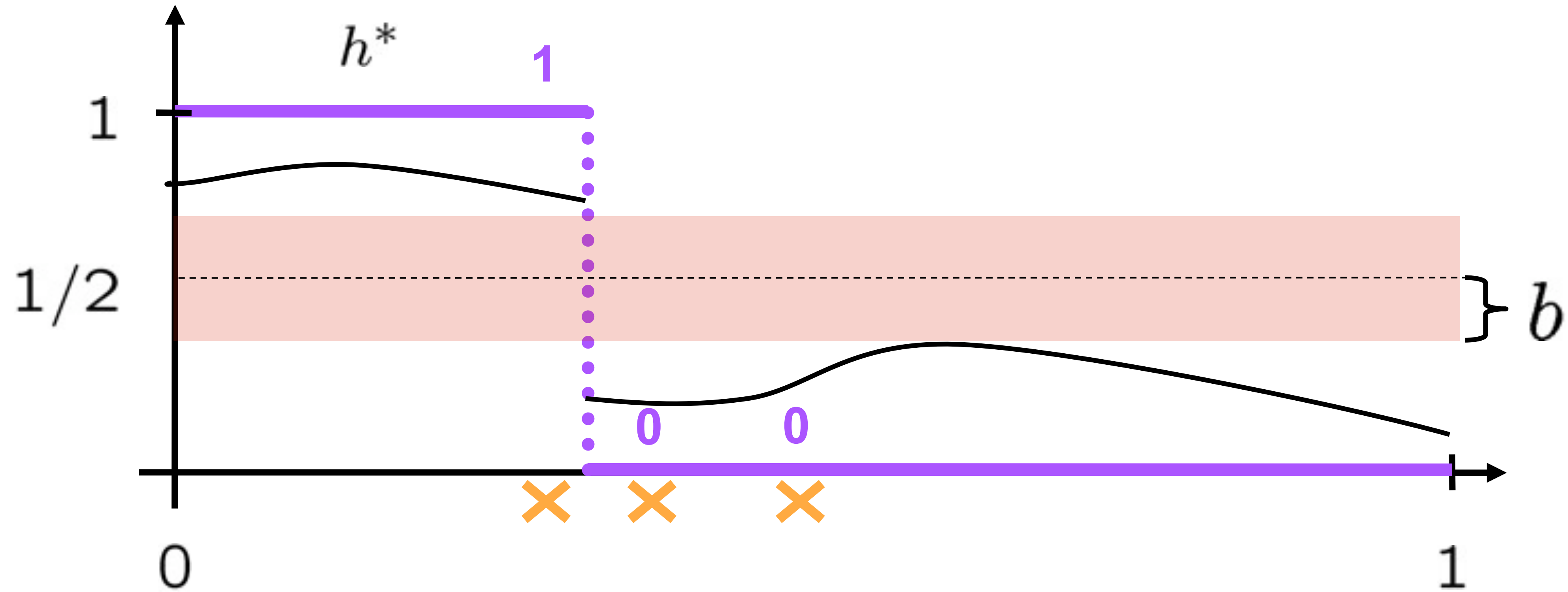
sequentially take samples at posterior median

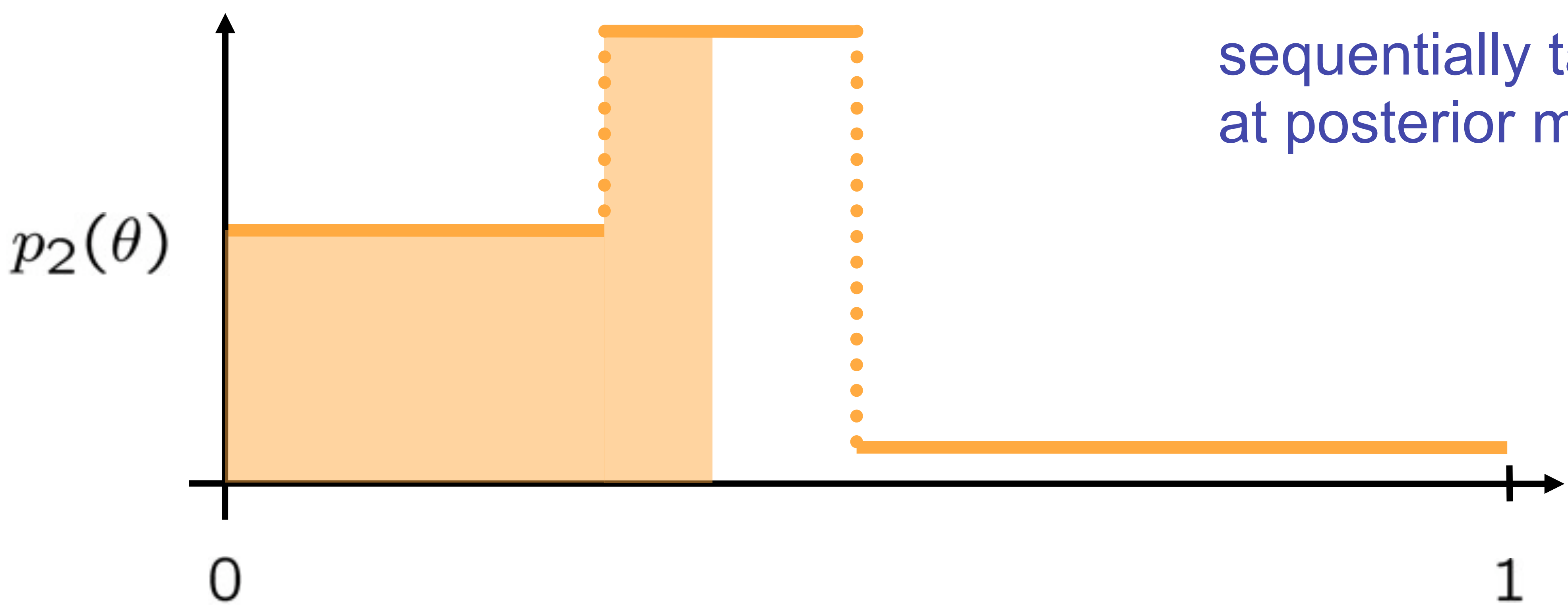
Horstein's Multiplicative Weighting Method



sequentially take samples at posterior median

Horstein's Multiplicative Weighting Method

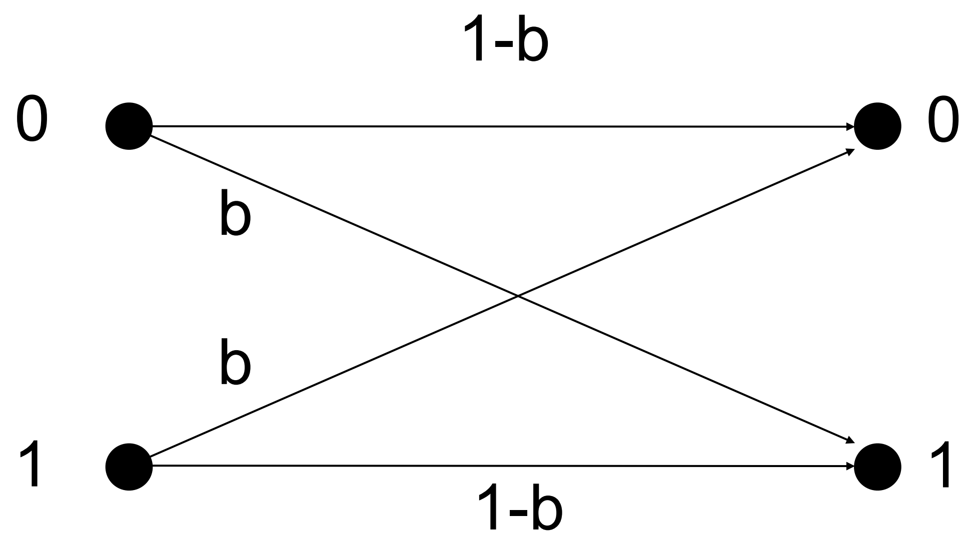




sequentially take samples at posterior median

Channel Coding with Noiseless Feedback

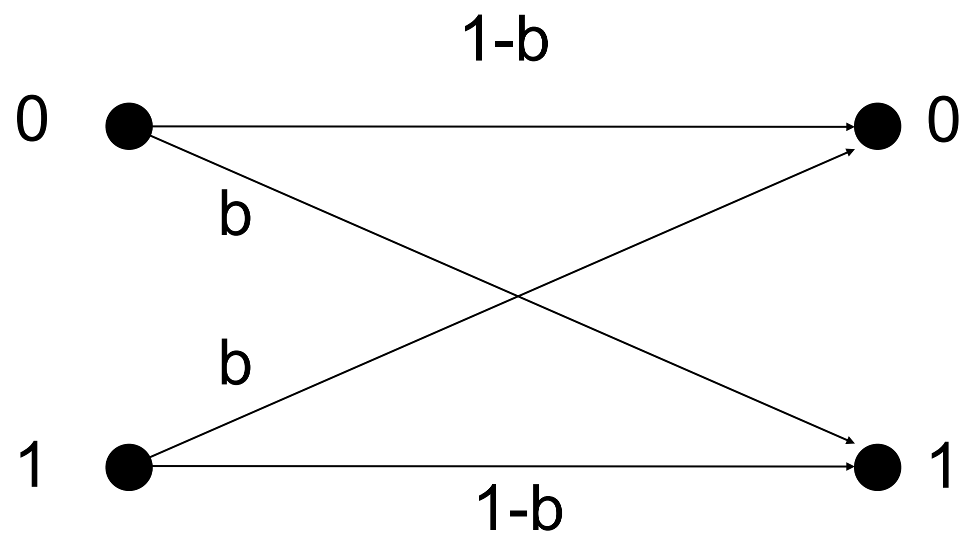
sender



receiver

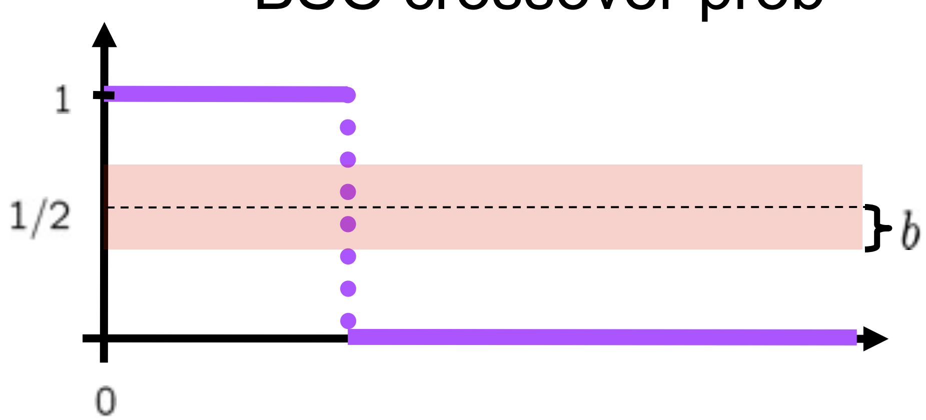
Channel Coding with Noiseless Feedback

sender



receiver

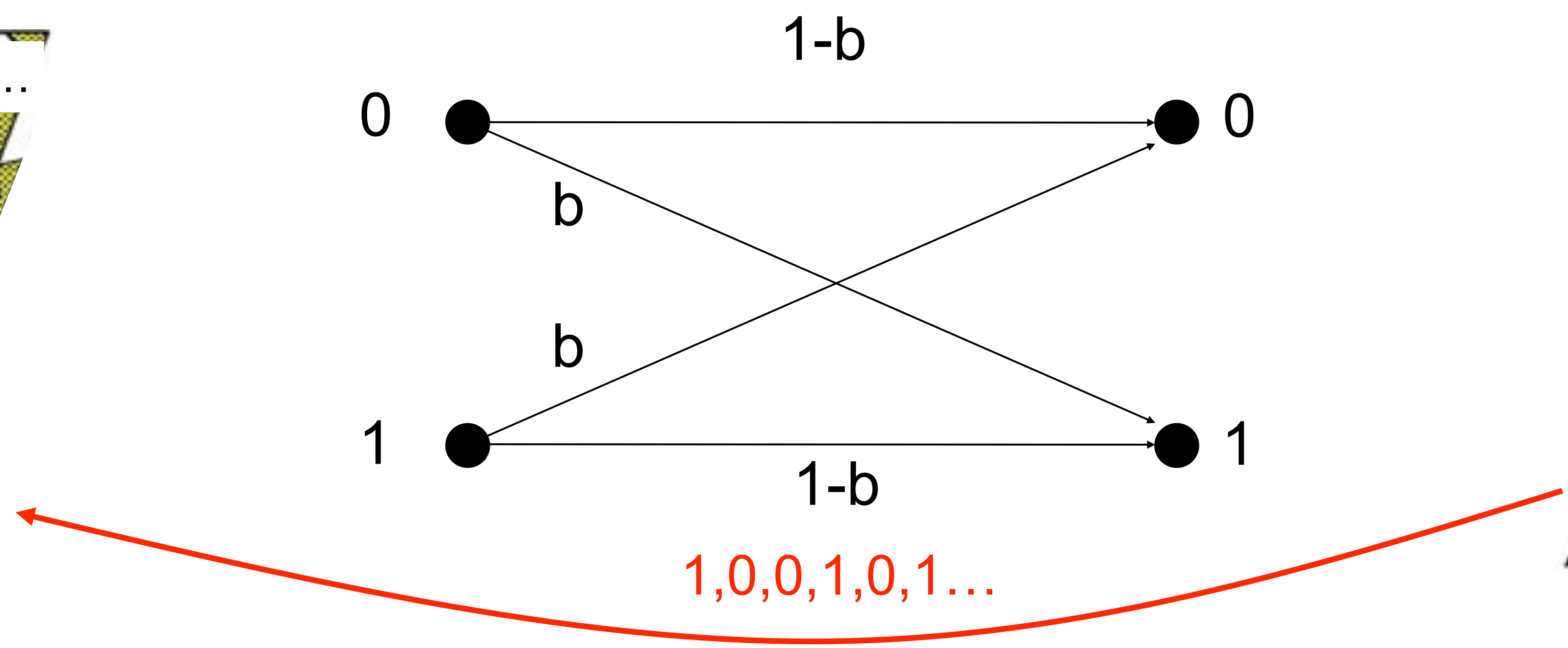
noise bound
= BSC crossover prob



threshold location
= n bit message

Channel Coding with Noiseless Feedback

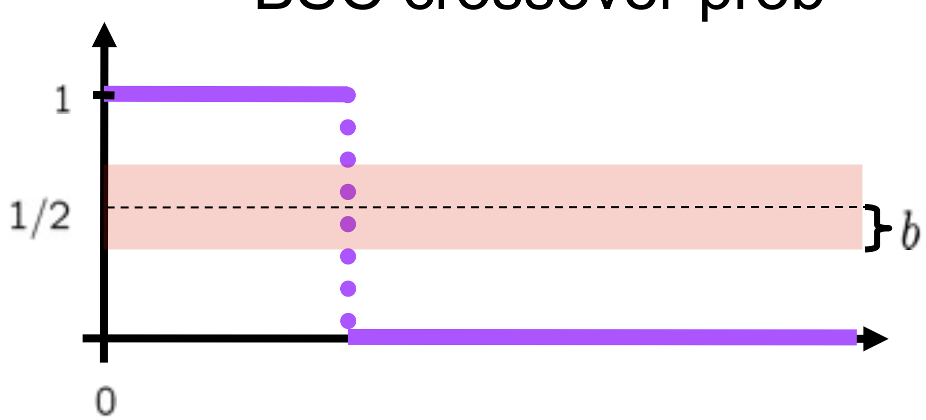
sender



receiver

noiseless feedback

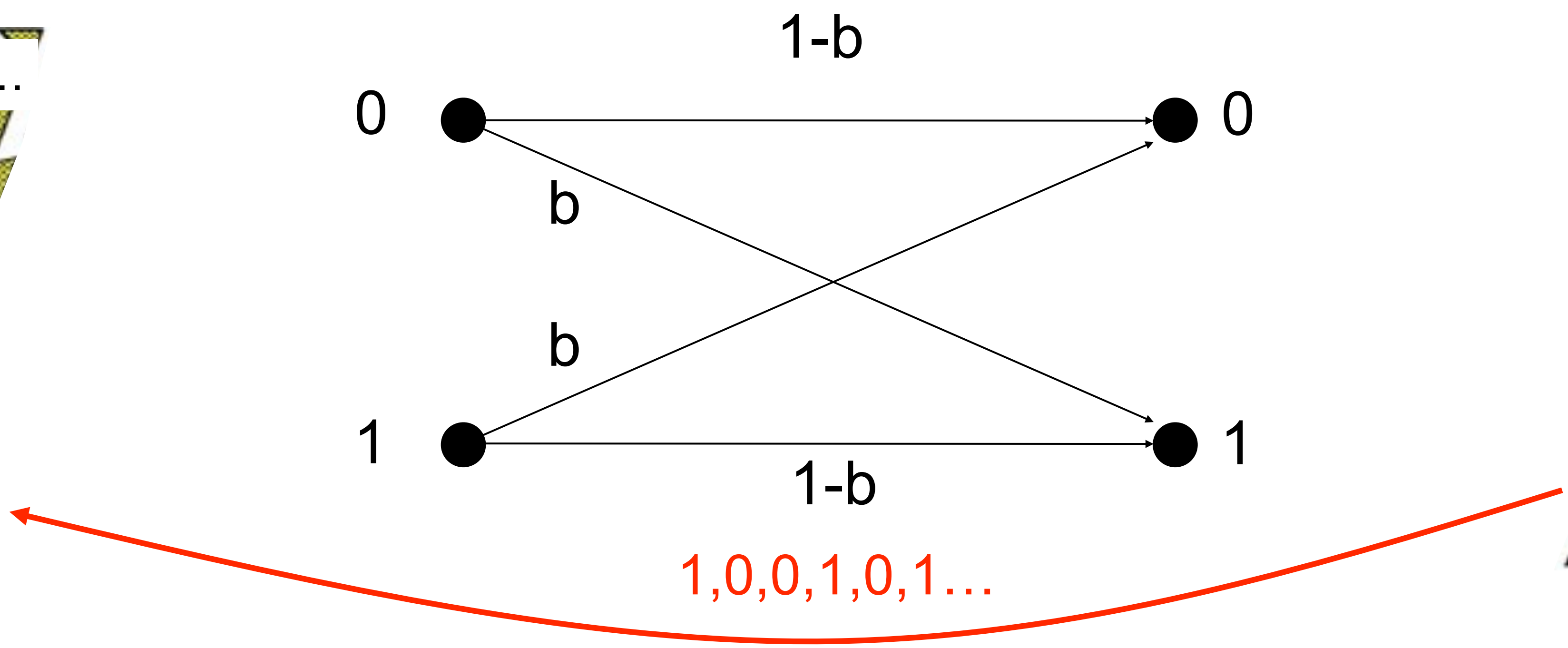
noise bound
= BSC crossover prob



threshold location
= n bit message

Channel Coding with Noiseless Feedback

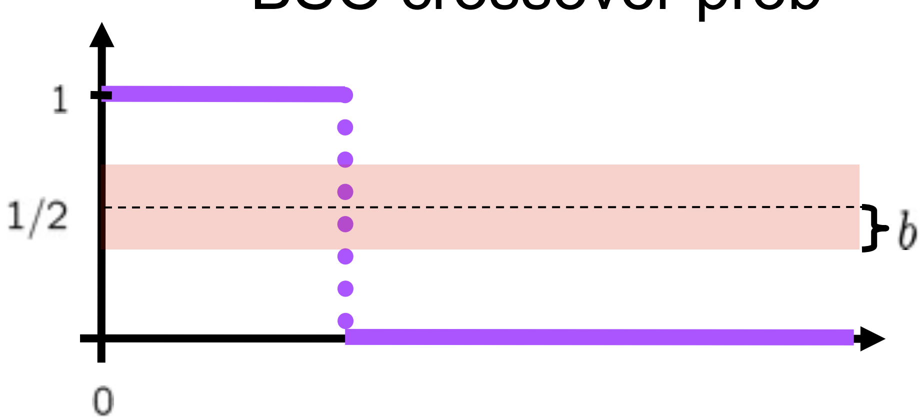
sender



noiseless feedback

receiver

noise bound
= BSC crossover prob



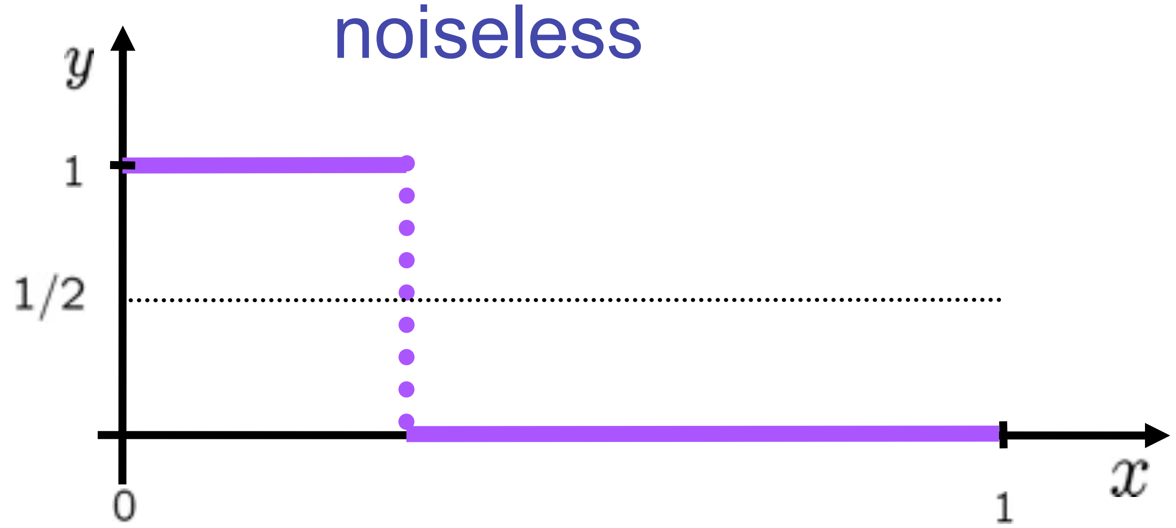
threshold location
= n bit message

Both sender and receiver implement Horstein’s algorithm

Sender deduces which binary symbol to send next in order to yield the greatest possible reduction in the receiver's uncertainty about n-bit message

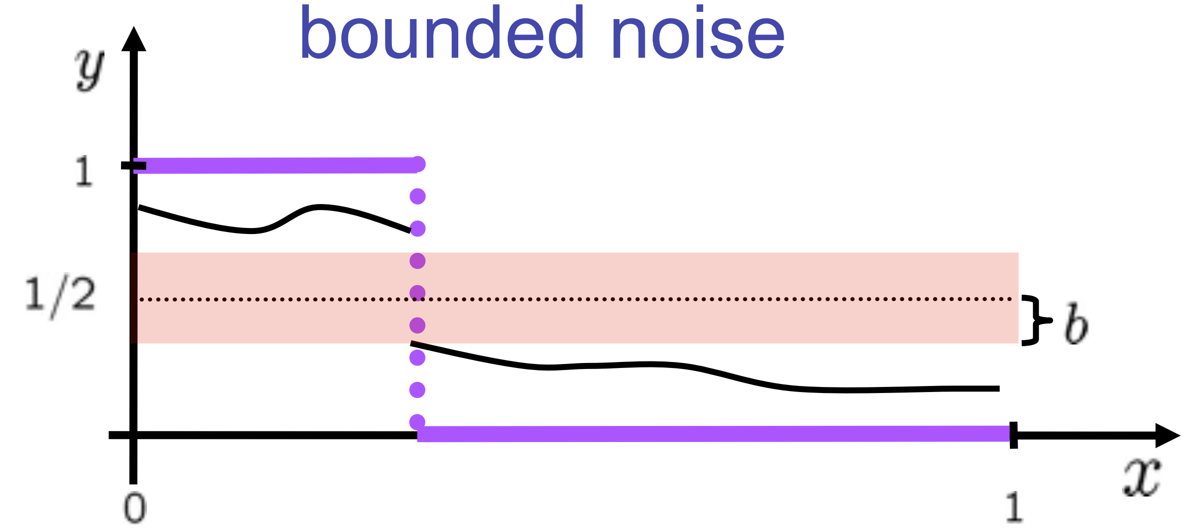
Active Learning in Unbounded Noise

noiseless



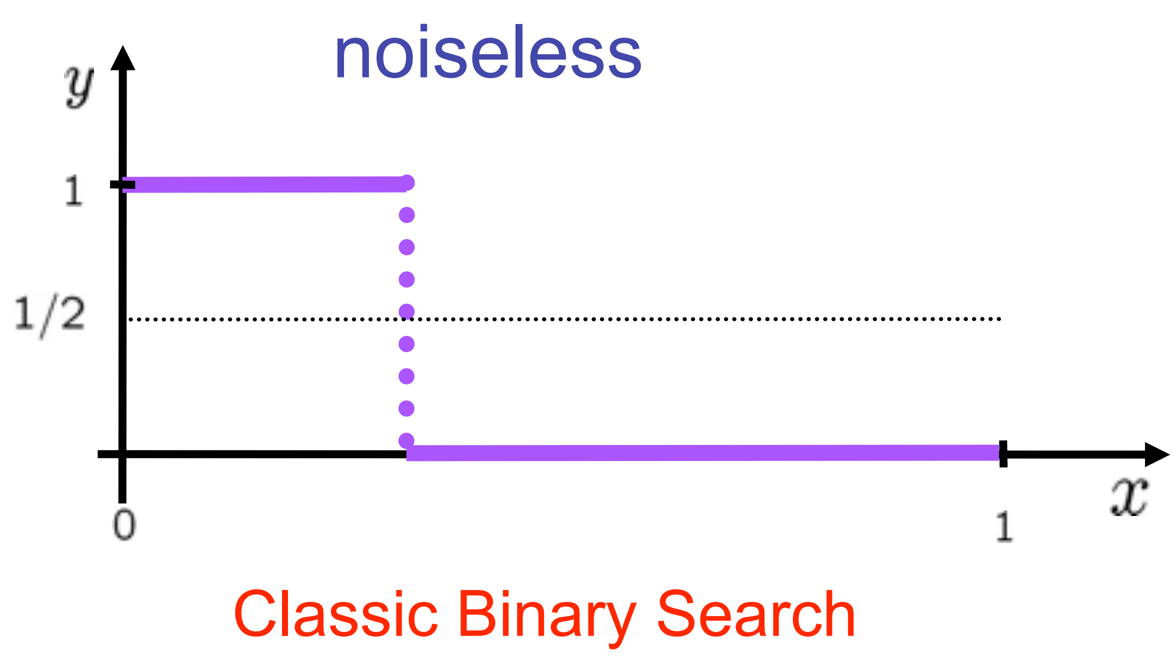
Classic Binary Search

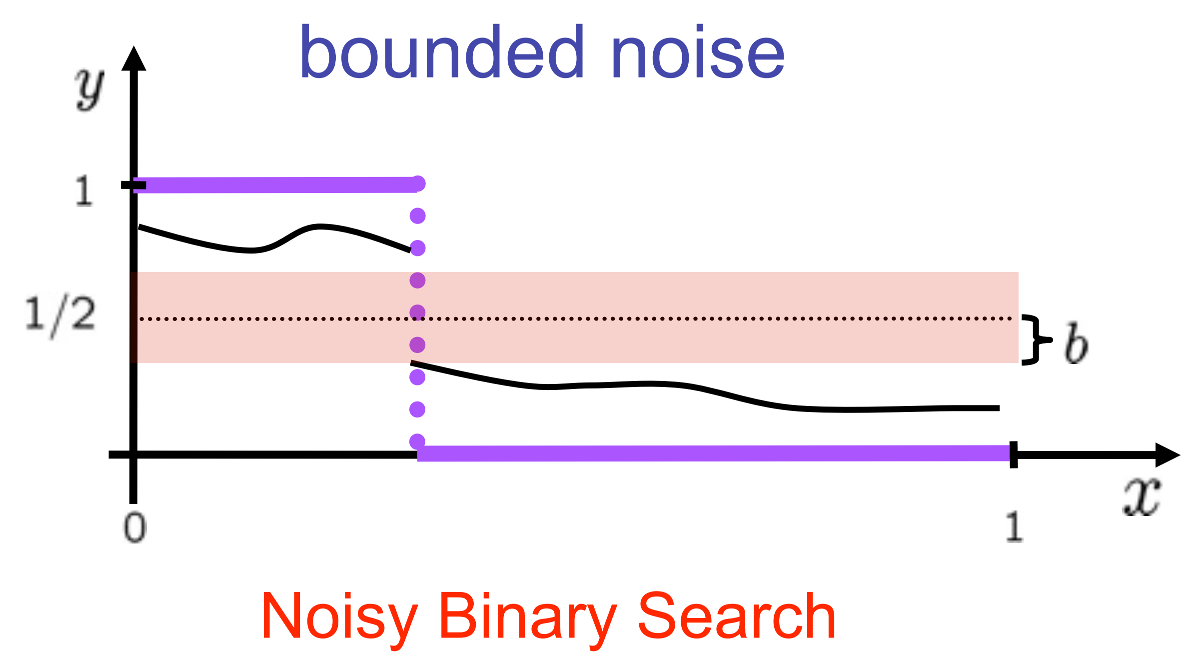
bounded noise

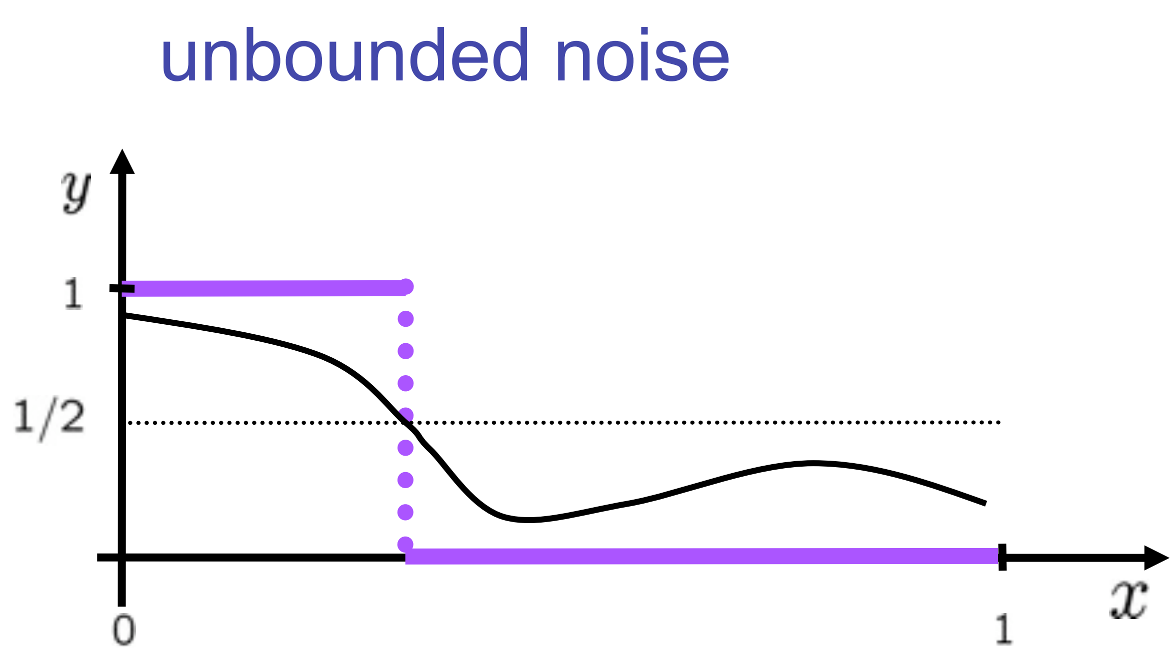


Noisy Binary Search

Active Learning in Unbounded Noise

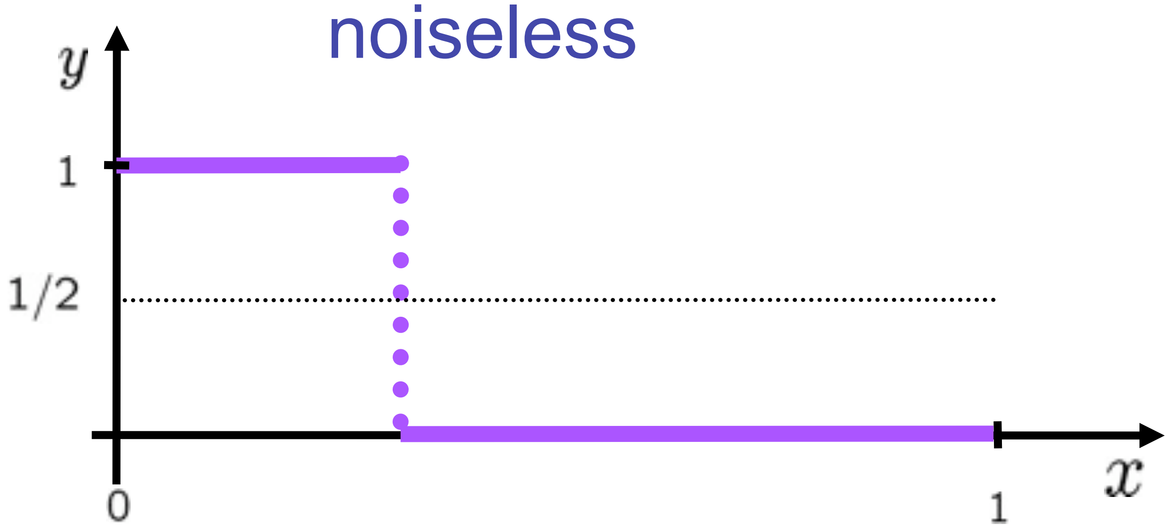






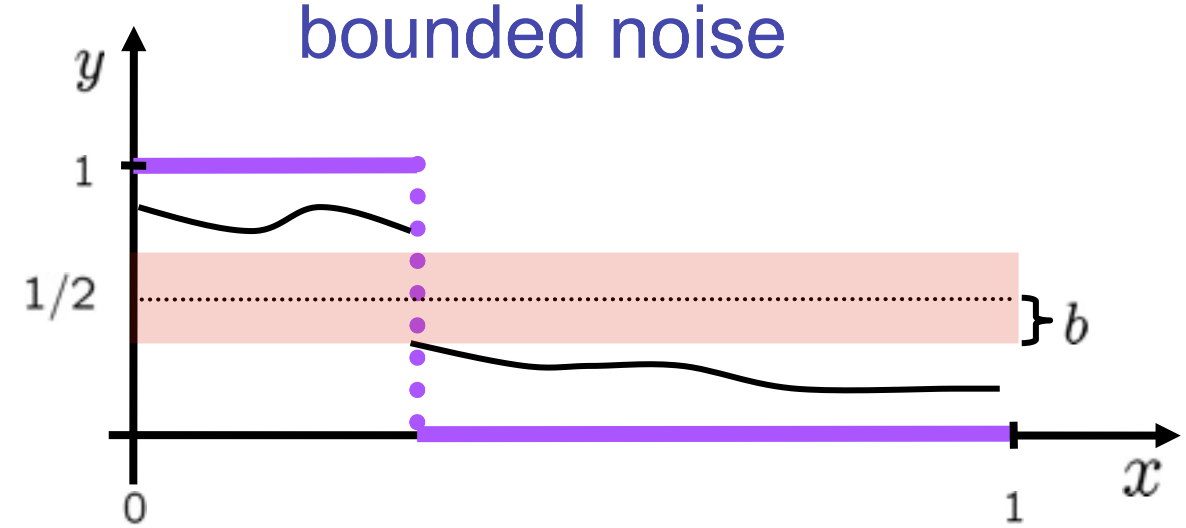
Active Learning in Unbounded Noise

noiseless



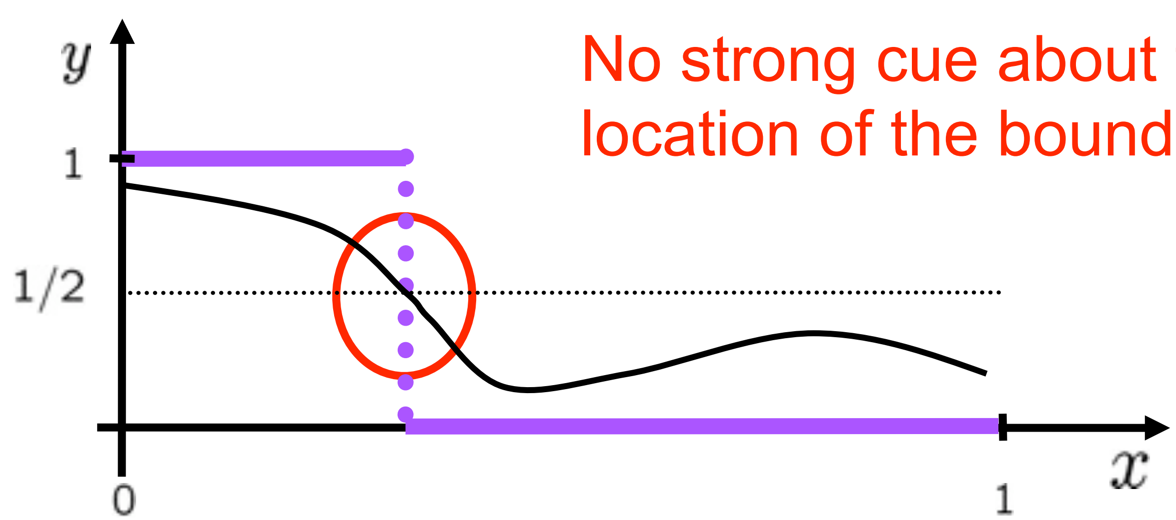
Classic Binary Search

bounded noise



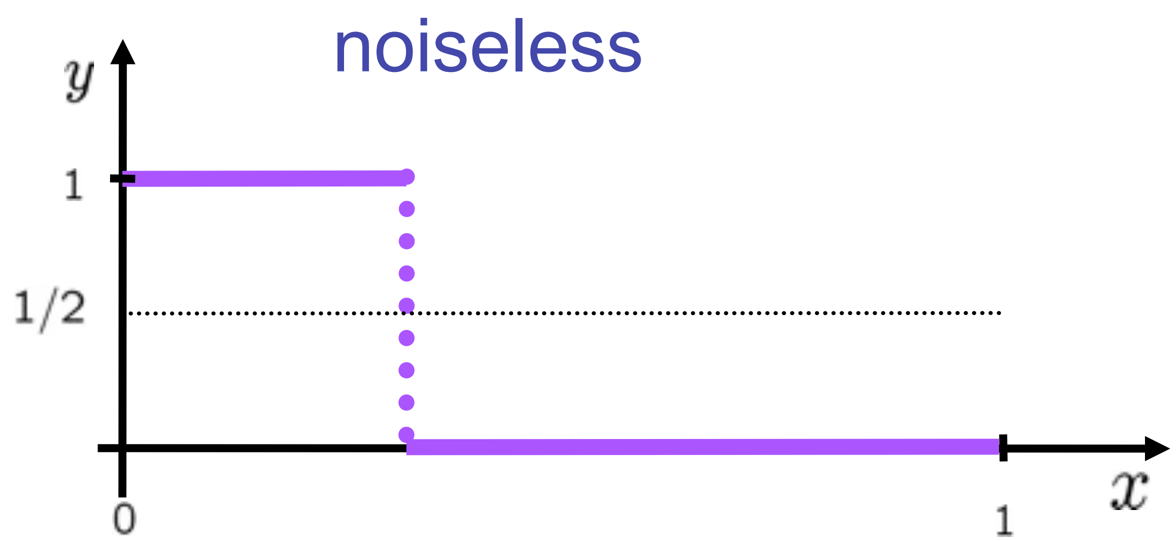
Noisy Binary Search

unbounded noise

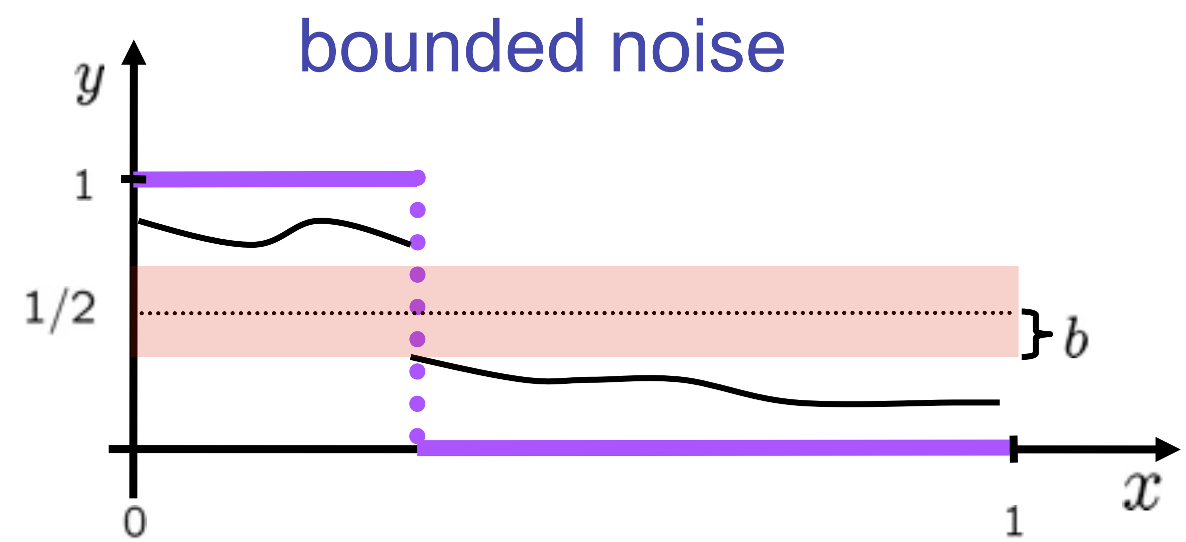


No strong cue about the location of the boundary

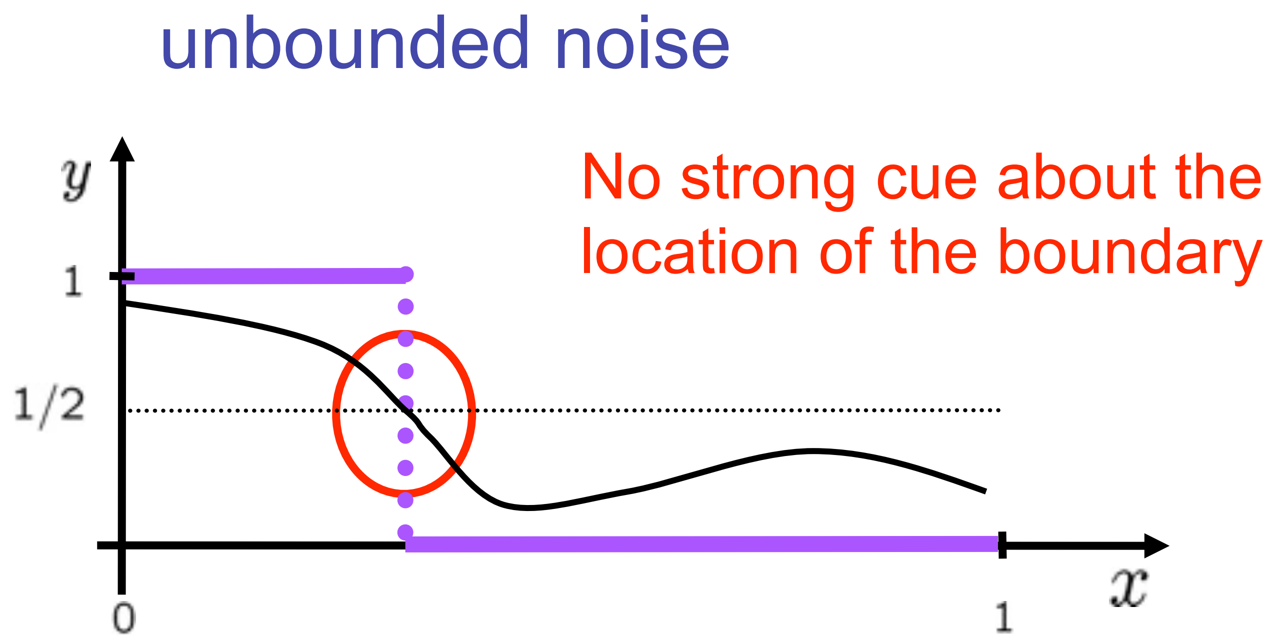
Active Learning in Unbounded Noise



Classic Binary Search



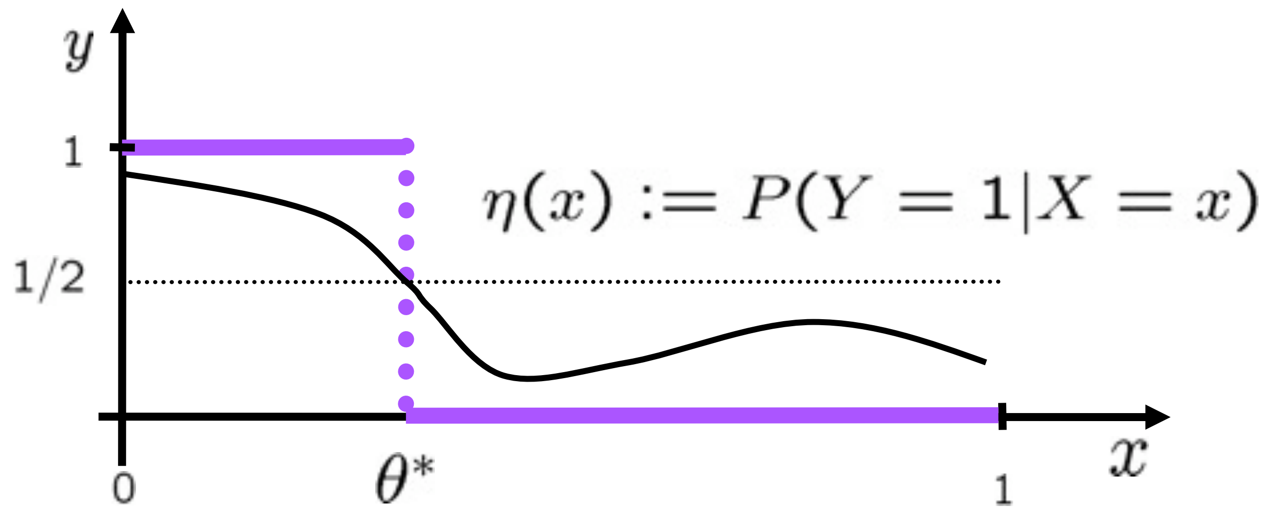
Noisy Binary Search



Rui Castro (Columbia): “How much does active learning help in this case ?”

Unbounded Noise Effects

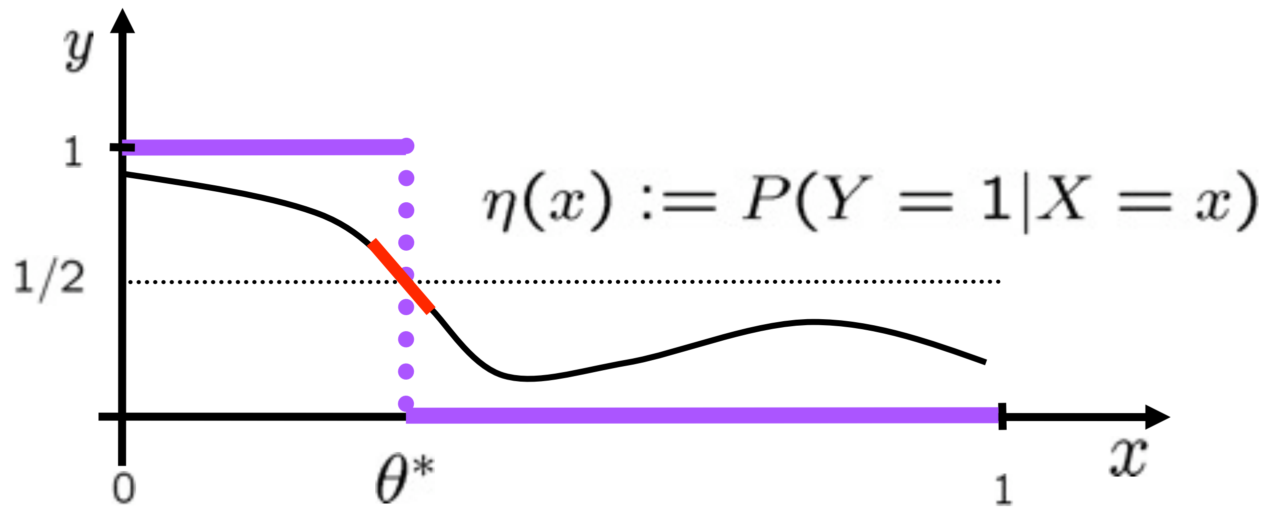
Near $\frac{1}{2}$-level, $c|x - \theta^*|^{\kappa-1} \leq |\eta(x) - 1/2| \leq C|x - \theta^*|^{\kappa-1}, \quad \kappa \geq 1$



similar conditions are commonly employed in nonparametric statistics, Tsybakov (2004)

Unbounded Noise Effects

Near $\frac{1}{2}$-level, $c|x-\theta^*|^{\kappa-1} \le |\eta(x) - 1/2| \le C|x-\theta^*|^{\kappa-1}$, $\kappa \ge 1$

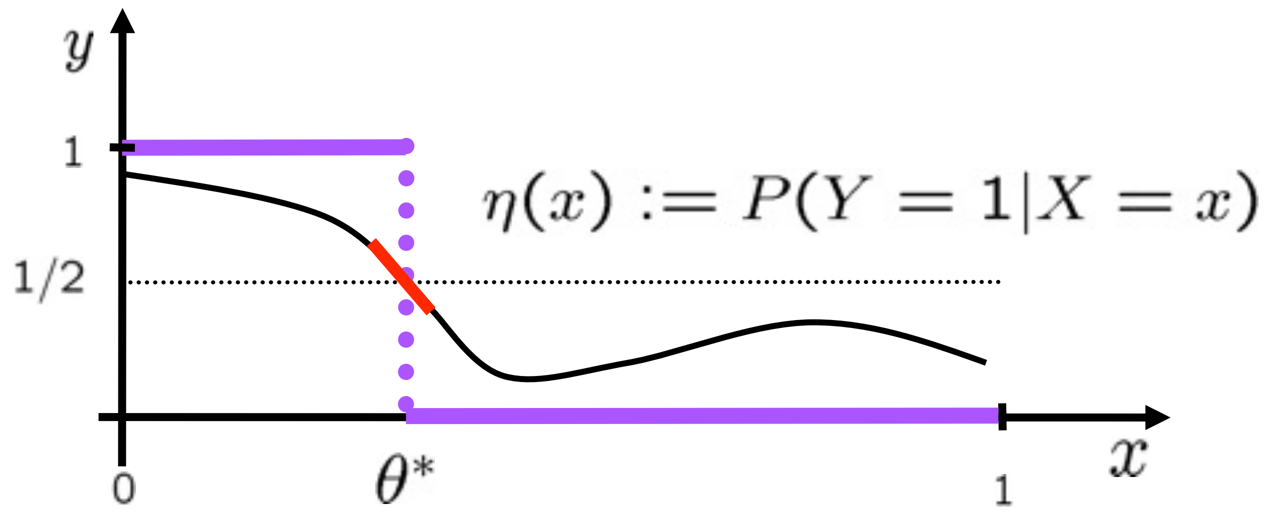


$\kappa = 2$

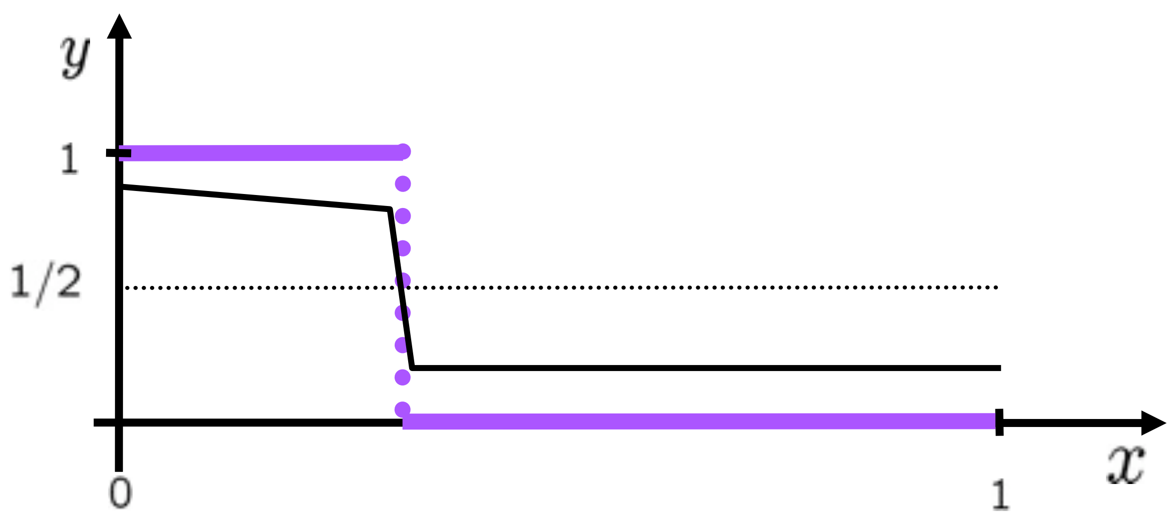
similar conditions are commonly employed in nonparametric statistics, Tsybakov (2004)

Unbounded Noise Effects

Near $\frac{1}{2}$-level, $\boxed{c|x-\theta^*|^{\kappa-1} \;\leq\; |\eta(x)-1/2| \;\leq\; C|x-\theta^*|^{\kappa-1}}$, $\kappa \geq 1$



$\kappa = 2$

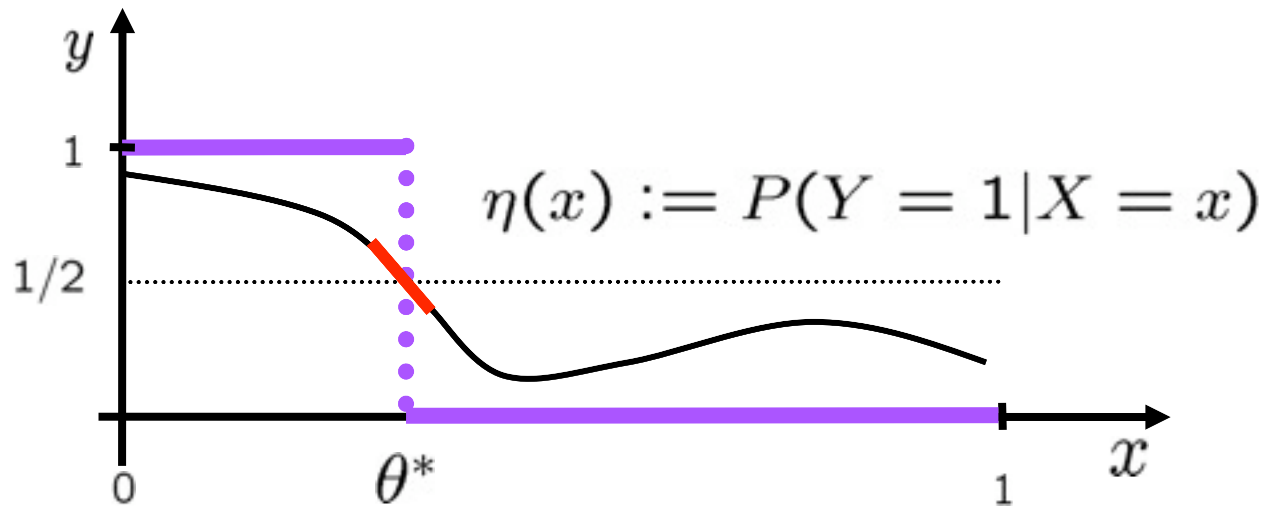


$\kappa \to 1$

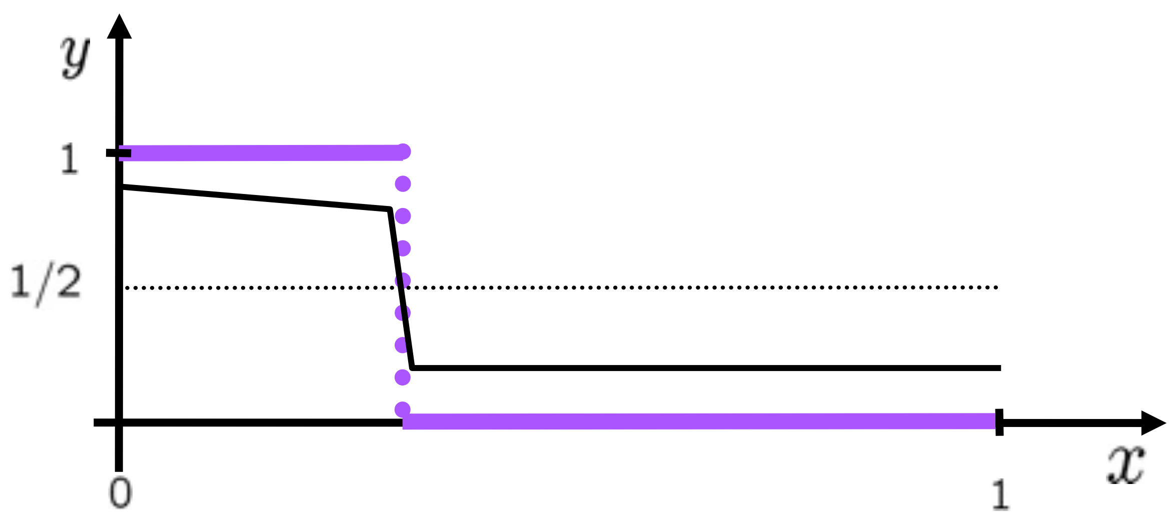
similar conditions are commonly employed in nonparametric statistics, Tsybakov (2004)

Unbounded Noise Effects

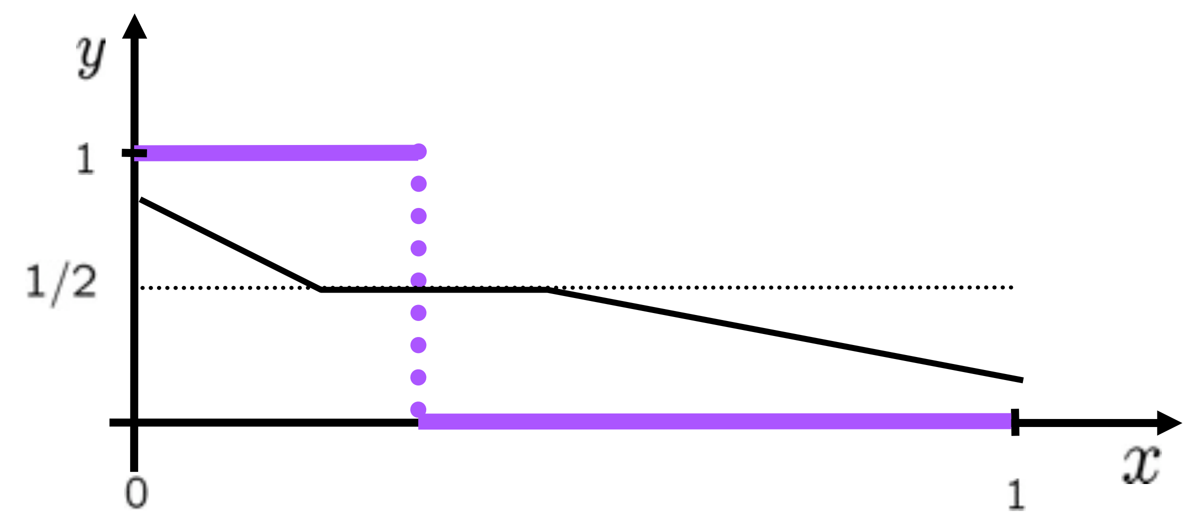
Near $\frac{1}{2}$-level, $c|x-\theta^*|^{\kappa-1} \leq |\eta(x) - 1/2| \leq C|x-\theta^*|^{\kappa-1}$, $\kappa \geq 1$



$\kappa = 2$



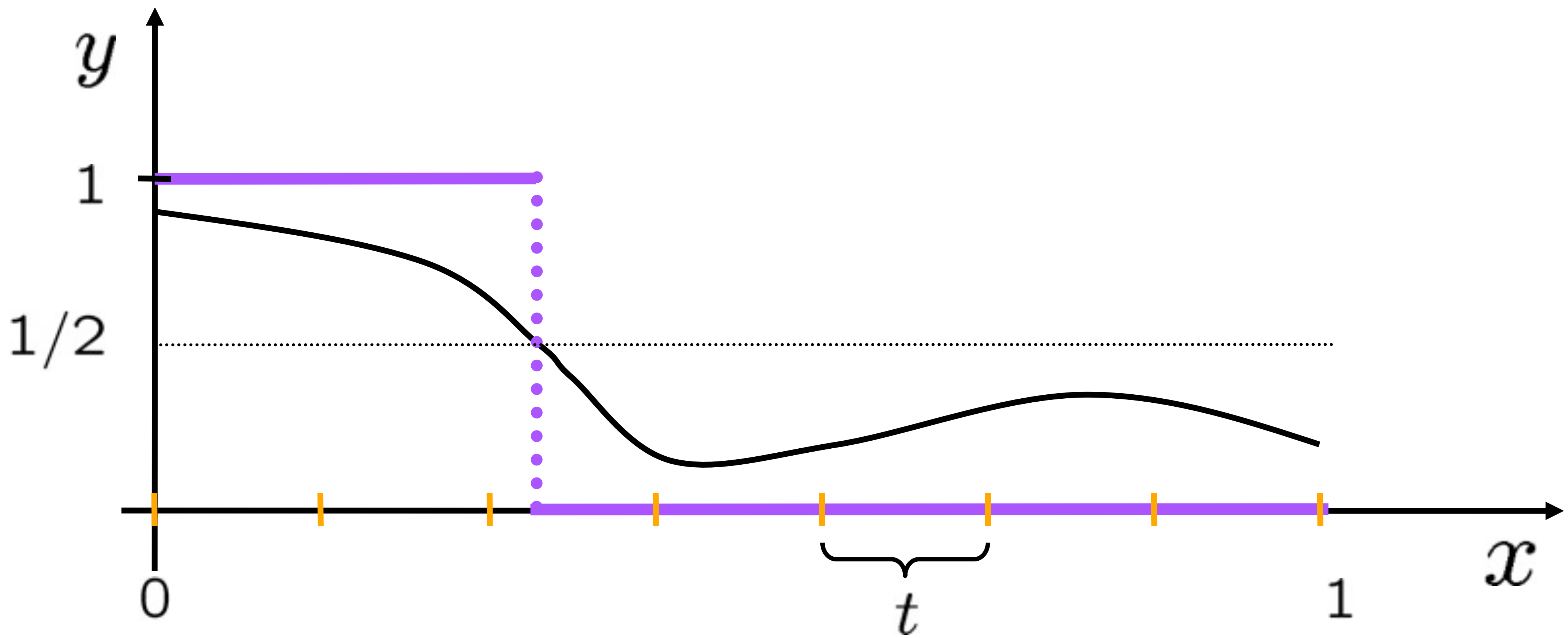
$\kappa \rightarrow 1$



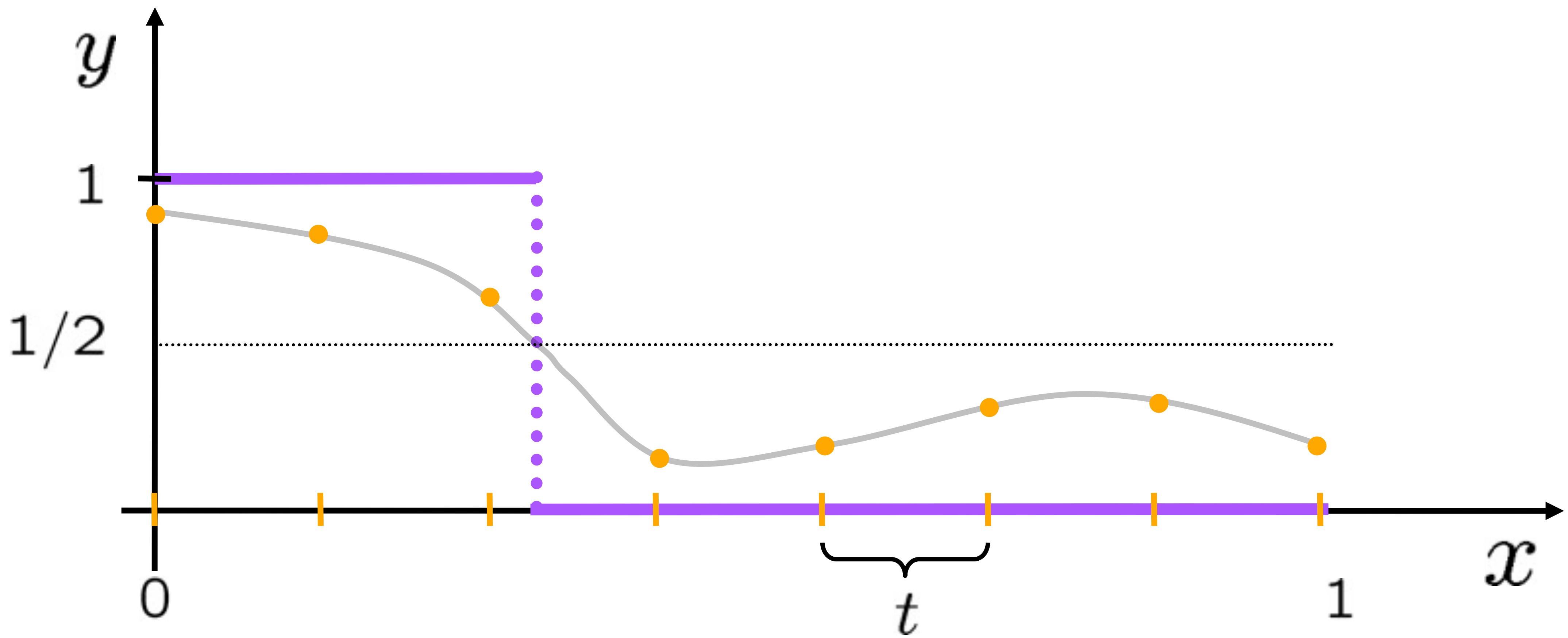
$\kappa \rightarrow \infty$

similar conditions are commonly employed in nonparametric statistics, Tsybakov (2004)

Horstein's Algorithm in Unbounded Noise

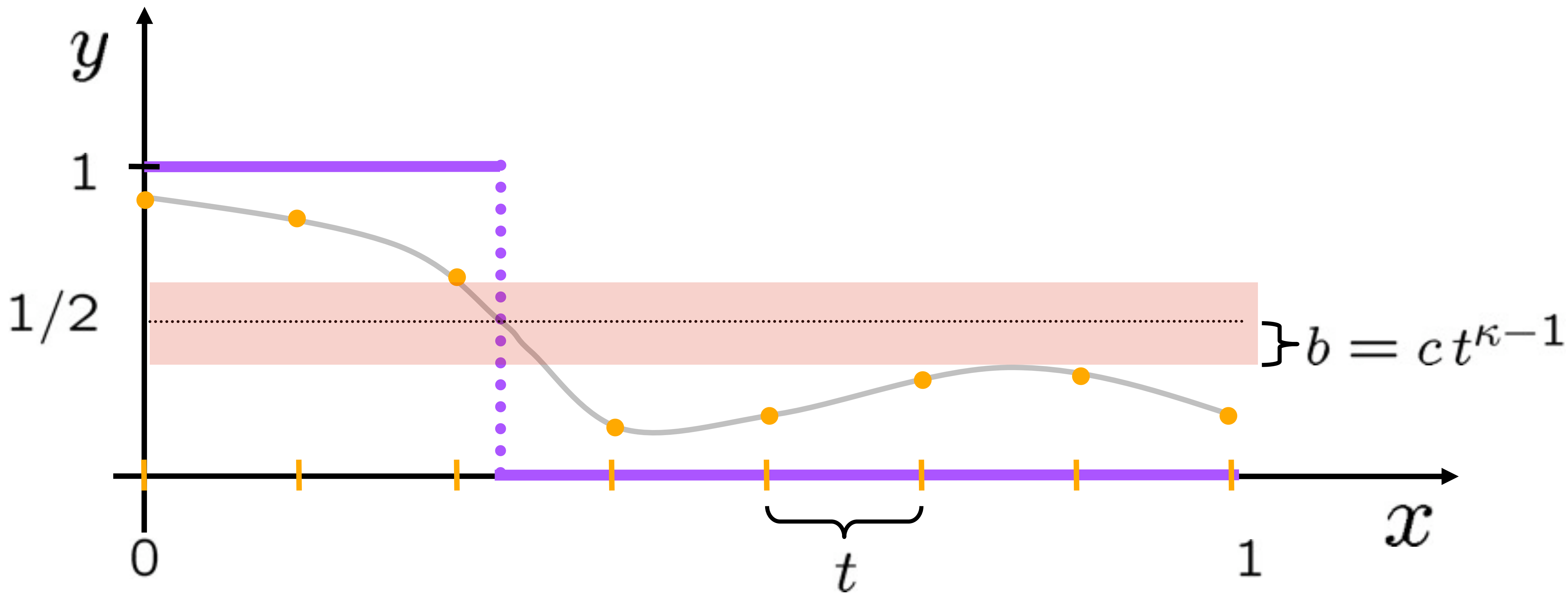


Horstein's Algorithm in Unbounded Noise



Consider discrete set of thresholds and discretized version of P(Y=1|X=x)

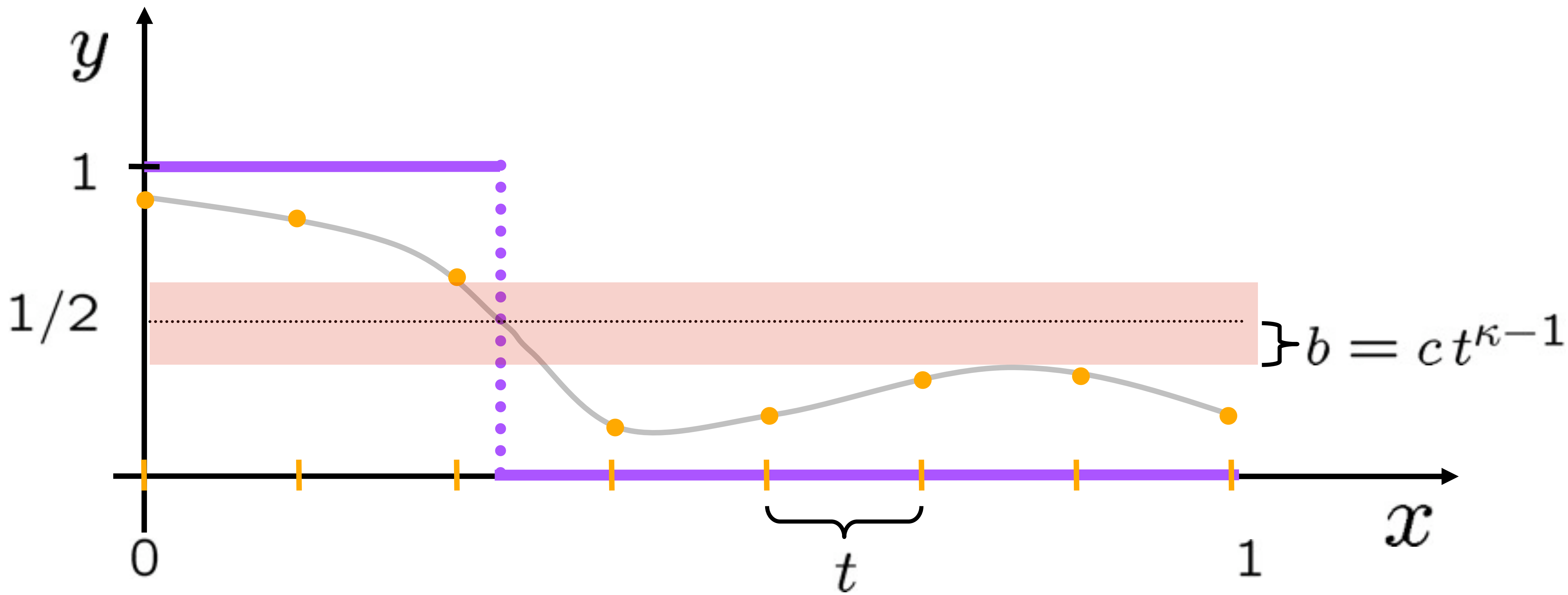
Horstein's Algorithm in Unbounded Noise



Consider discrete set of thresholds and discretized version of P(Y=1|X=x)

If ½ level is not aligned with discrete thresholds, then noise of discretized problem is bounded, but depends on resolution of discretization t and the behavior of P(Y=1|X=x) at the ½ level

Horstein's Algorithm in Unbounded Noise

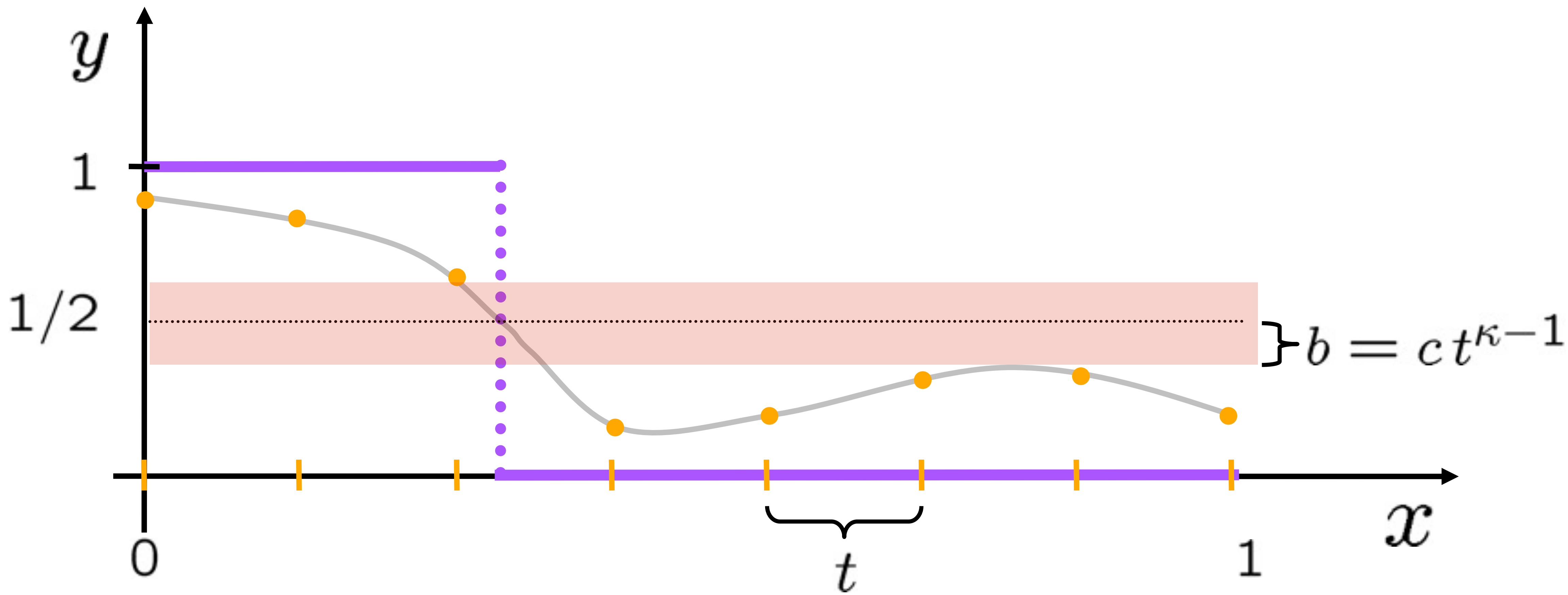


Consider discrete set of thresholds and discretized version of P(Y=1|X=x)

If ½ level is not aligned with discrete thresholds, then noise of discretized problem is bounded, but depends on resolution of discretization t and the behavior of P(Y=1|X=x) at the ½ level

$$\mathbb{P}[h_n(X) \neq Y] - \mathbb{P}[h^*(X) \neq Y] \quad \leq \quad t^{\kappa} \; + \; t^{-1}\exp(-nc^2 t^{2\kappa-2})$$

Horstein's Algorithm in Unbounded Noise

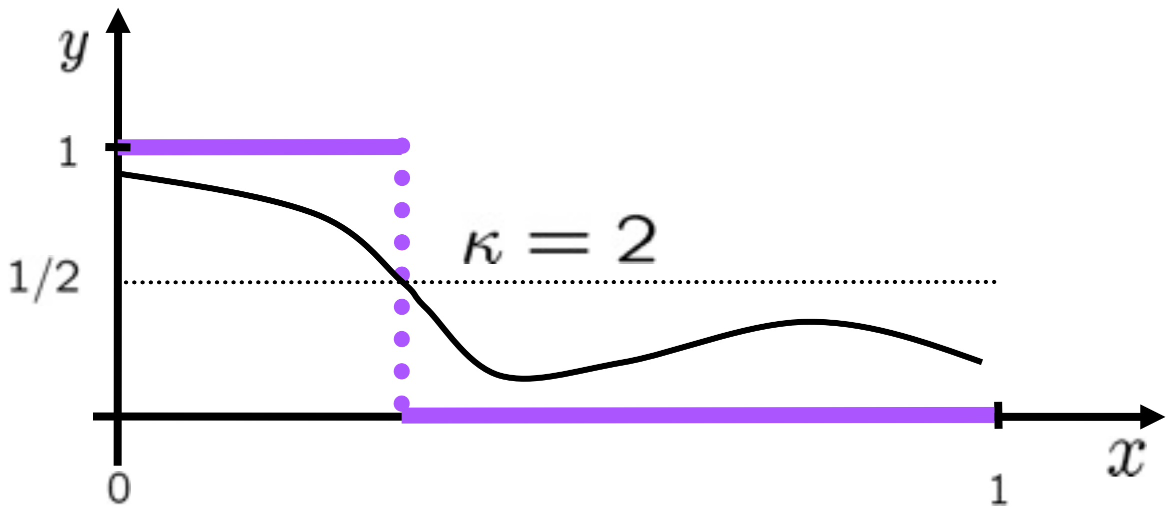


Consider discrete set of thresholds and discretized version of P(Y=1|X=x)

If ½ level is not aligned with discrete thresholds, then noise of discretized problem is bounded, but depends on resolution of discretization t and the behavior of P(Y=1|X=x) at the ½ level

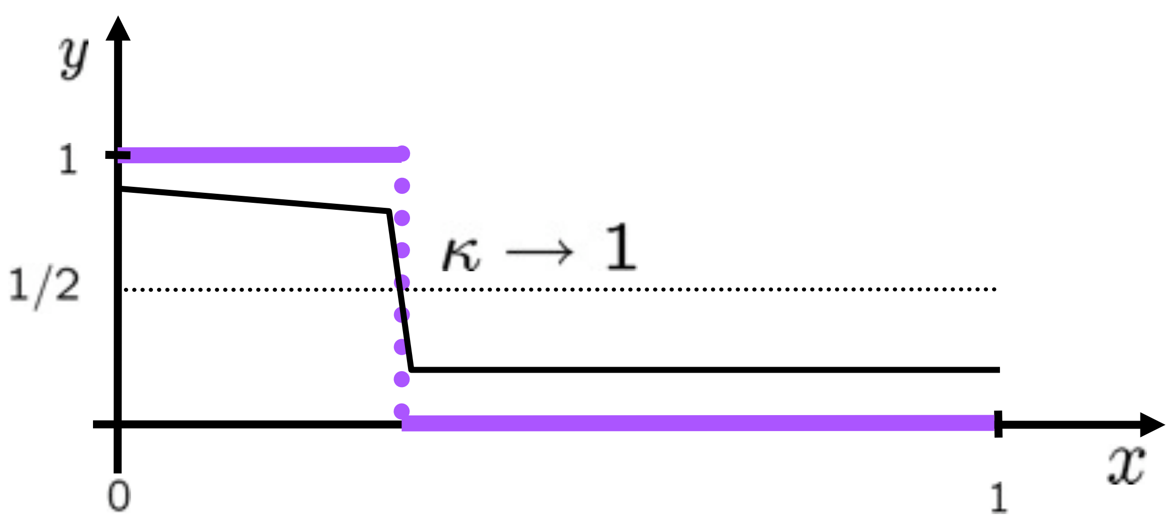
$$\begin{aligned}\mathbb{P}[h_n(X) \neq Y] - \mathbb{P}[h^*(X) \neq Y] &\leq t^{\kappa} + t^{-1}\exp(-nc^2 t^{2\kappa-2}) \\ &= O\left(\left[\frac{\log n}{n}\right]^{\frac{\kappa}{2\kappa-2}}\right)\end{aligned}$$

Rates of Convergence



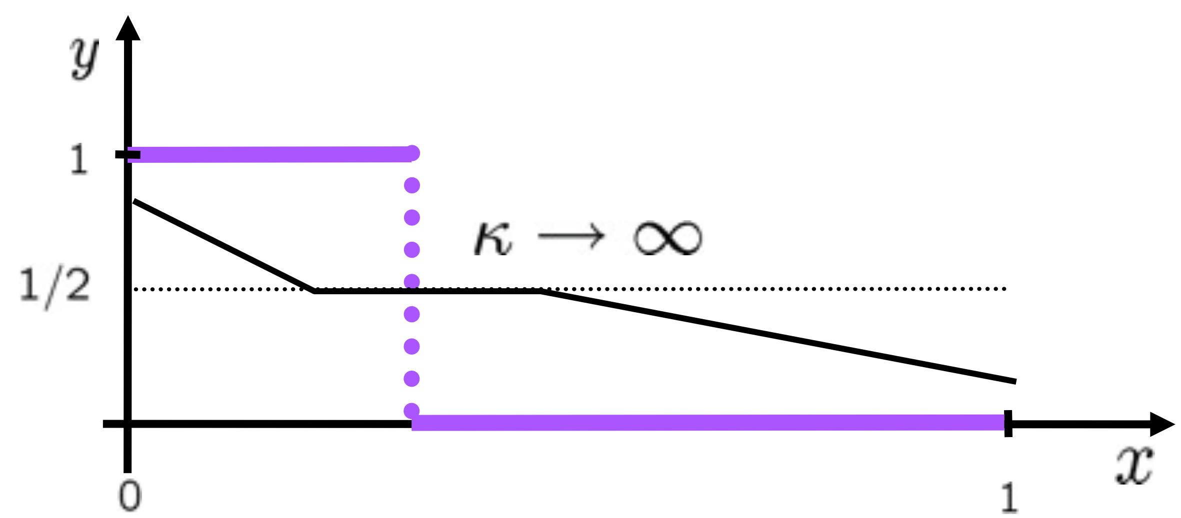
passive: $n^{-2/3}$

active: n^{-1}



passive: $\to n^{-1}$

active: $\to e^{-cn}$



passive: $\to n^{-1/2}$

active: $\to n^{-1/2}$

Are you a good active learner ?

Castro, Kalish, Nowak, Qian, Rogers & Zhu (NIPS 2008)

Investigate human active learning in task analogous to 1-d threshold problem

Are you a good active learner ?

Castro, Kalish, Nowak, Qian, Rogers & Zhu (NIPS 2008)

Investigate human active learning in task
analogous to 1-d threshold problem

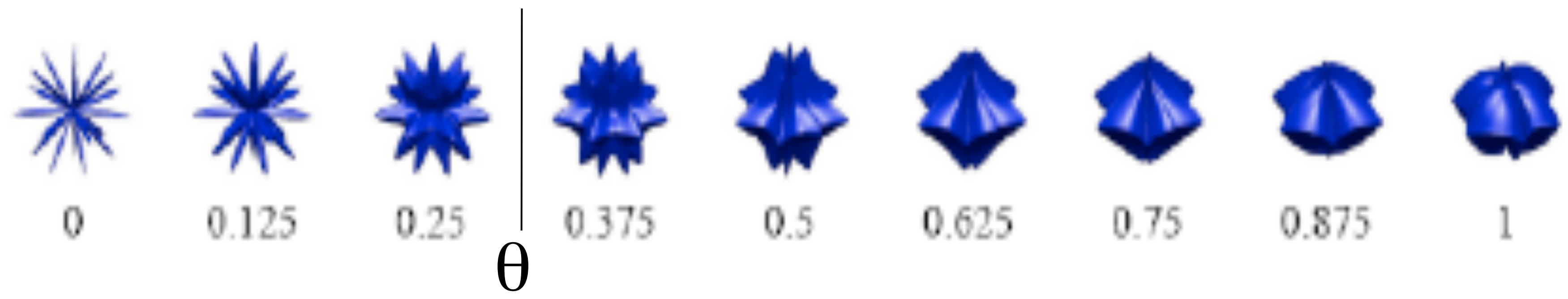
alien eggs

Are you a good active learner ?

Castro, Kalish, Nowak, Qian, Rogers & Zhu (NIPS 2008)

Investigate human active learning in task analogous to 1-d threshold problem

alien eggs

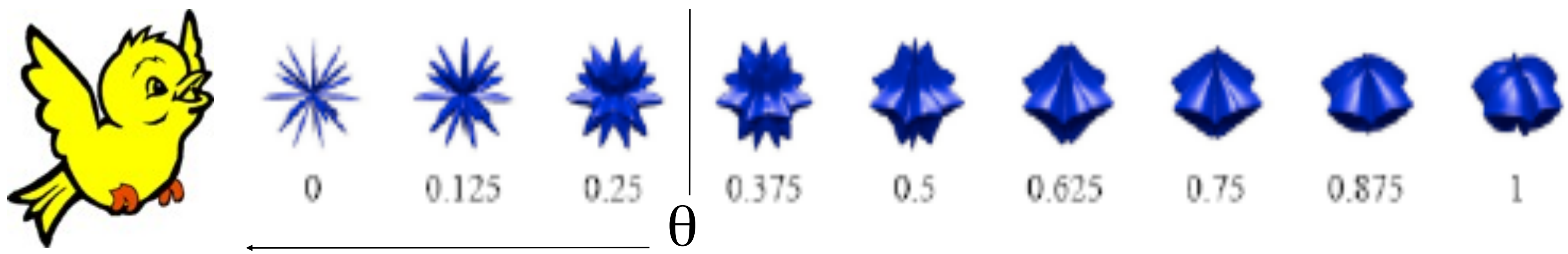


Are you a good active learner ?

Castro, Kalish, Nowak, Qian, Rogers & Zhu (NIPS 2008)

Investigate human active learning in task analogous to 1-d threshold problem

alien eggs



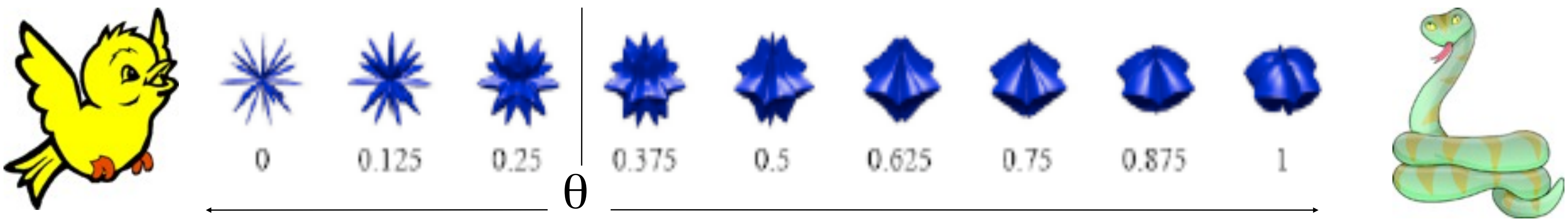
more probably birds

Are you a good active learner ?

Castro, Kalish, Nowak, Qian, Rogers & Zhu (NIPS 2008)

Investigate human active learning in task analogous to 1-d threshold problem

alien eggs



more probably birds

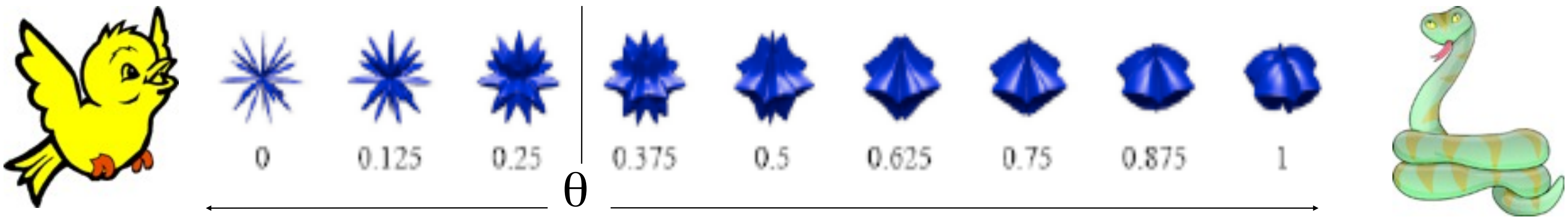
more probably snakes

Are you a good active learner ?

Castro, Kalish, Nowak, Qian, Rogers & Zhu (NIPS 2008)

Investigate human active learning in task analogous to 1-d threshold problem

alien eggs



more probably birds

more probably snakes

Subjects observe random egg hatchings (passive learning) or they can select eggs to hatch (active learning).

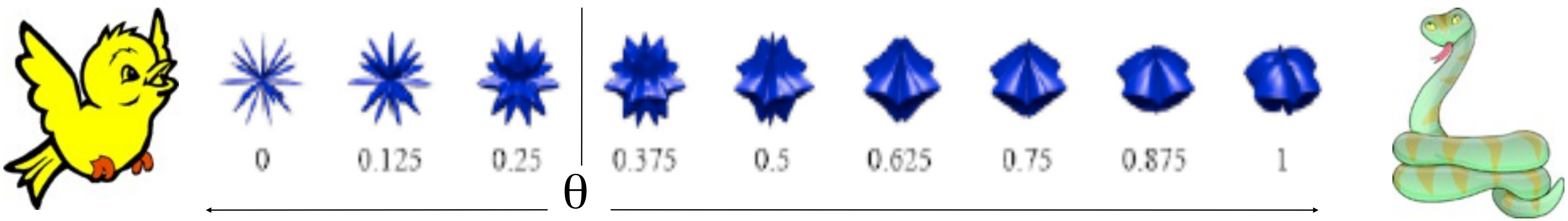
They are asked to determine the egg shape where snakes become more probable than birds.

Are you a good active learner ?

Castro, Kalish, Nowak, Qian, Rogers & Zhu (NIPS 2008)

Investigate human active learning in task analogous to 1-d threshold problem

alien eggs



more probably birds

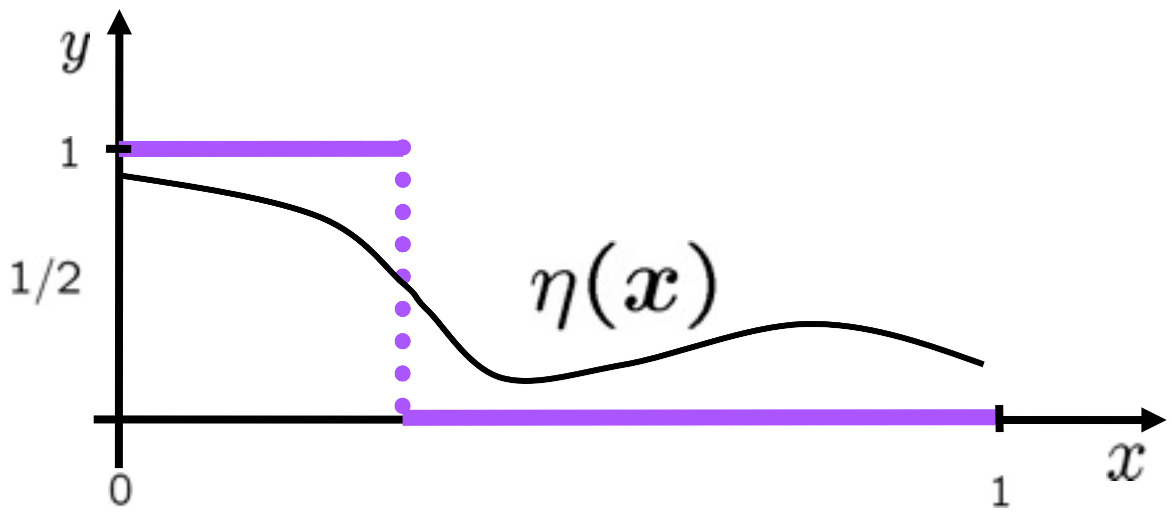
more probably snakes

Subjects observe random egg hatchings (passive learning) or they can select eggs to hatch (active learning).

They are asked to determine the egg shape where snakes become more probable than birds.

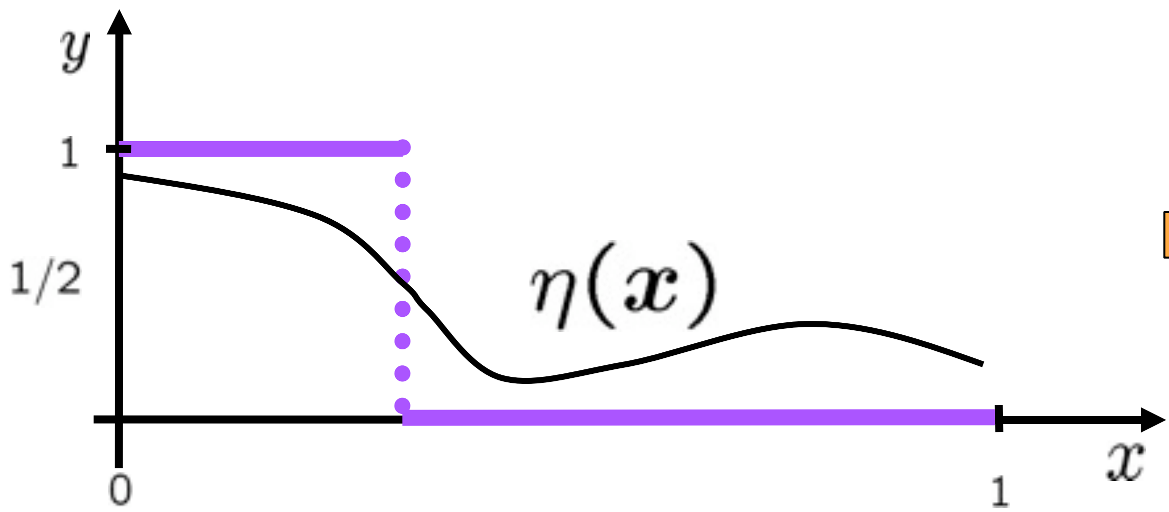
Results: Human learning rates agree with theory, $1/n$ in passive mode and $\exp(-cn)$ in active mode.

Learning Multidimensional Threshold Functions



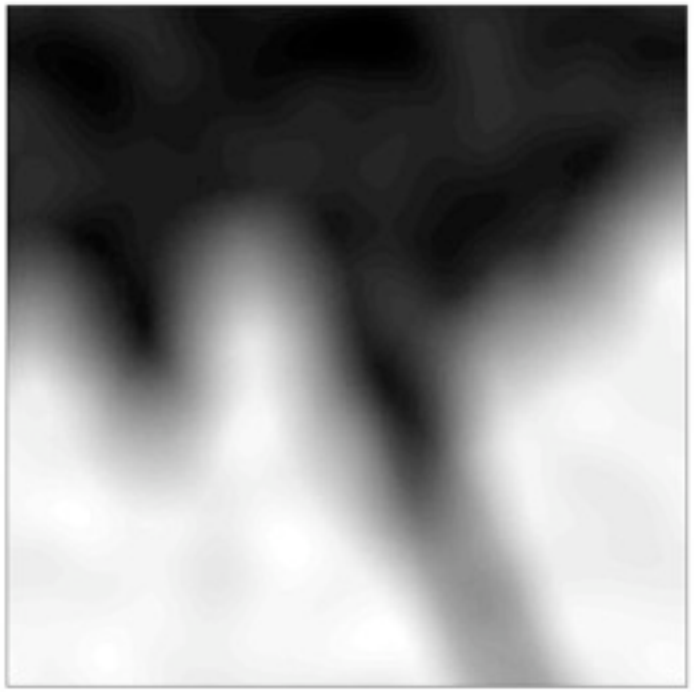
$$d = 1$$

Learning Multidimensional Threshold Functions



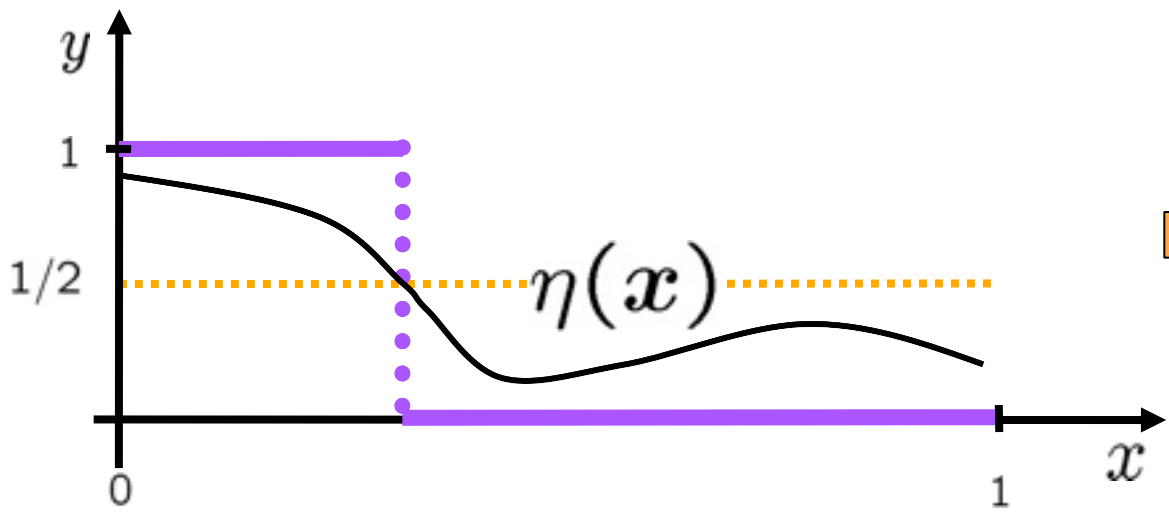
$d = 1$

$\eta(\boldsymbol{x})$

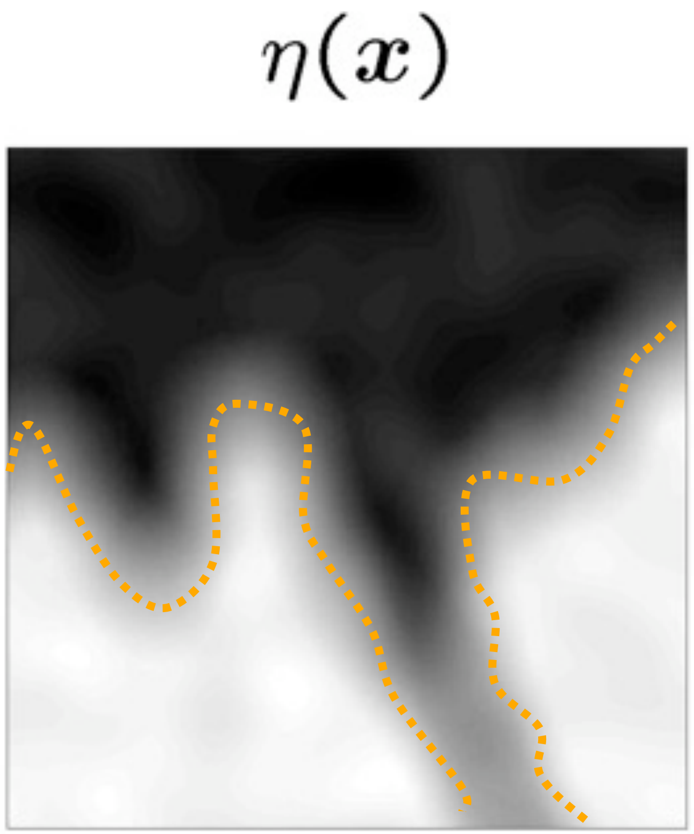


$d > 1$

Learning Multidimensional Threshold Functions



$d = 1$



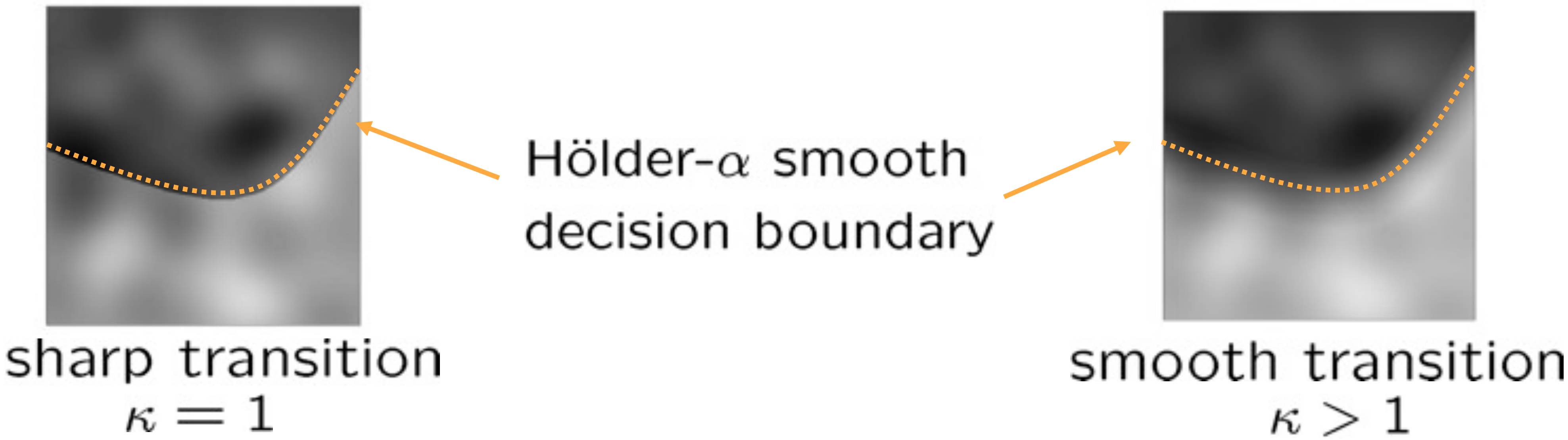
$d > 1$

Learning Rates for Multidimensional Thresholds

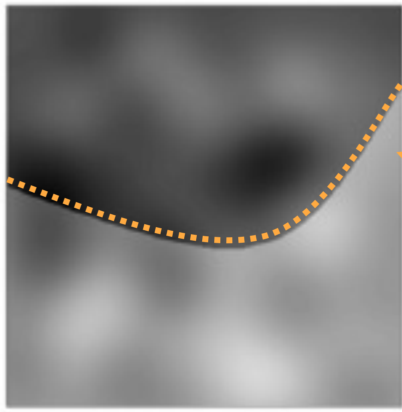
sharp transition
$\kappa = 1$

smooth transition
$\kappa > 1$

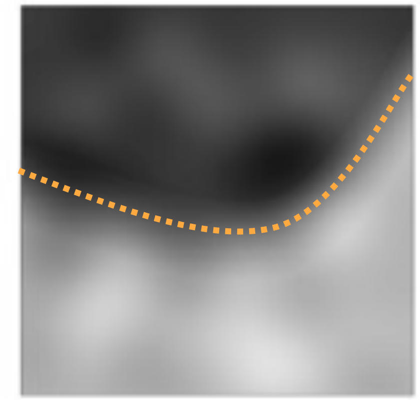
Learning Rates for Multidimensional Thresholds



Learning Rates for Multidimensional Thresholds



sharp transition
$\kappa = 1$



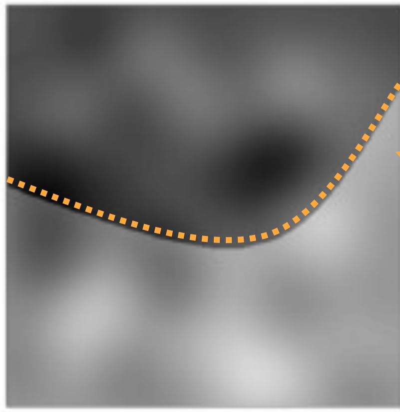
smooth transition
$\kappa > 1$

Active Learning: Theorem (R. Castro and RN '07)

$$\left(\frac{1}{n}\right)^{\frac{\kappa}{2\kappa+\rho-2}} \preceq \inf_{h_n, S_n} \sup_{P_{XY} \in \mathrm{BF}(\alpha,\kappa)} \mathcal{E}(h_n) \preceq \left(\frac{\log n}{n}\right)^{\frac{\kappa}{2\kappa+\rho-2}}$$

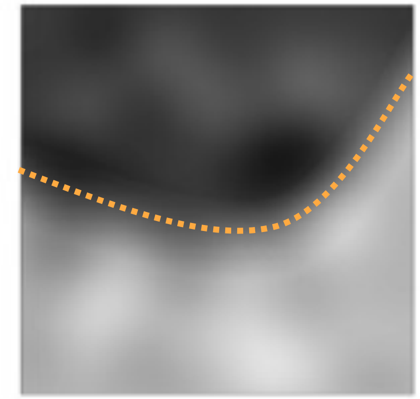
$(\rho = (d-1)/\alpha)$

Learning Rates for Multidimensional Thresholds



sharp transition
$\kappa = 1$

Hölder-α smooth
decision boundary



smooth transition
$\kappa > 1$

Active Learning: Theorem (R. Castro and RN '07)

$$\left(\frac{1}{n}\right)^{\frac{\kappa}{2\kappa+\rho-2}} \preceq \inf_{h_n, S_n} \sup_{P_{XY} \in \mathrm{BF}(\alpha,\kappa)} \mathcal{E}(h_n) \preceq \left(\frac{\log n}{n}\right)^{\frac{\kappa}{2\kappa+\rho-2}}$$

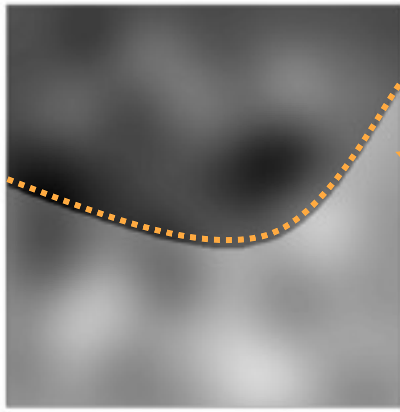
$(\rho = (d-1)/\alpha)$

Compare with passive learning

$$\inf_{h_n} \sup_{P_{XY} \in \mathrm{BF}(\alpha,\kappa)} \mathcal{E}(h_n) \asymp \left(\frac{1}{n}\right)^{\frac{\kappa}{2\kappa+\rho-1}}$$

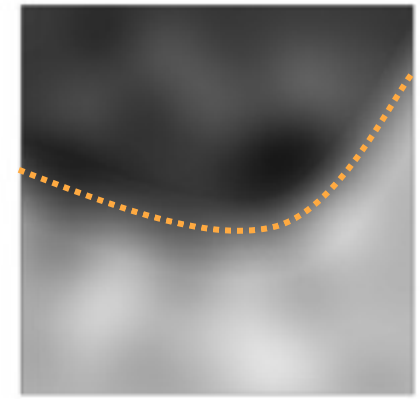
as $\rho \to 0$
and $\kappa \to 1$

Learning Rates for Multidimensional Thresholds



sharp transition
$\kappa = 1$

Hölder-α smooth
decision boundary



smooth transition
$\kappa > 1$

Active Learning: Theorem (R. Castro and RN '07)

$$\left(\frac{1}{n}\right)^{\frac{\kappa}{2\kappa+\rho-2}} \preceq \inf_{h_n, S_n} \sup_{P_{XY} \in \mathrm{BF}(\alpha,\kappa)} \mathcal{E}(h_n) \preceq \left(\frac{\log n}{n}\right)^{\frac{\kappa}{2\kappa+\rho-2}}$$

$(\rho = (d-1)/\alpha)$

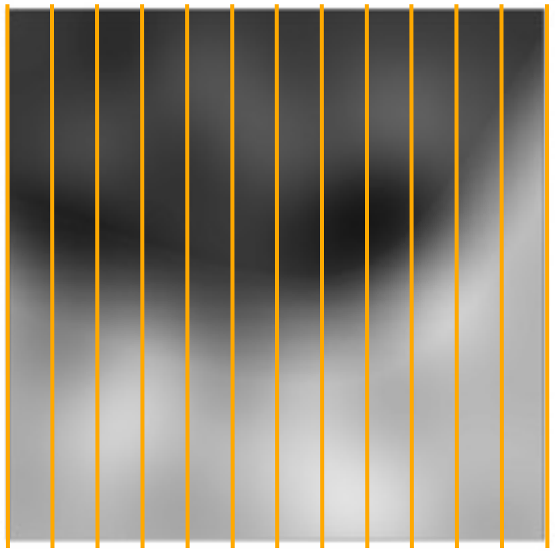
Compare with passive learning

$$\inf_{h_n} \sup_{P_{XY} \in \mathrm{BF}(\alpha,\kappa)} \mathcal{E}(h_n) \asymp \left(\frac{1}{n}\right)^{\frac{\kappa}{2\kappa+\rho-1}}$$

as $\rho \to 0$
and $\kappa \to 1$

Learning Rates for Multidimensional Thresholds

Main idea: reduce multidimensional problem to a sequence of 1-dim problems



Active Learning: Theorem (R. Castro and RN '07)

$$\left(\frac{1}{n}\right)^{\frac{\kappa}{2\kappa+\rho-2}} \preceq \inf_{h_n, S_n} \sup_{P_{XY} \in \mathsf{BF}(\alpha,\kappa)} \mathcal{E}(h_n) \preceq \left(\frac{\log n}{n}\right)^{\frac{\kappa}{2\kappa+\rho-2}}$$

$(\rho = (d-1)/\alpha)$

Compare with passive learning

$$\inf_{h_n} \sup_{P_{XY} \in \mathsf{BF}(\alpha,\kappa)} \mathcal{E}(h_n) \asymp \left(\frac{1}{n}\right)^{\frac{\kappa}{2\kappa+\rho-1}}$$

as $\rho \to 0$
and $\kappa \to 1$

Algorithms for Active Learning

$\mathcal{X}$:= domain or *query space*

$\mathcal{Y} := \{-1, +1\}$

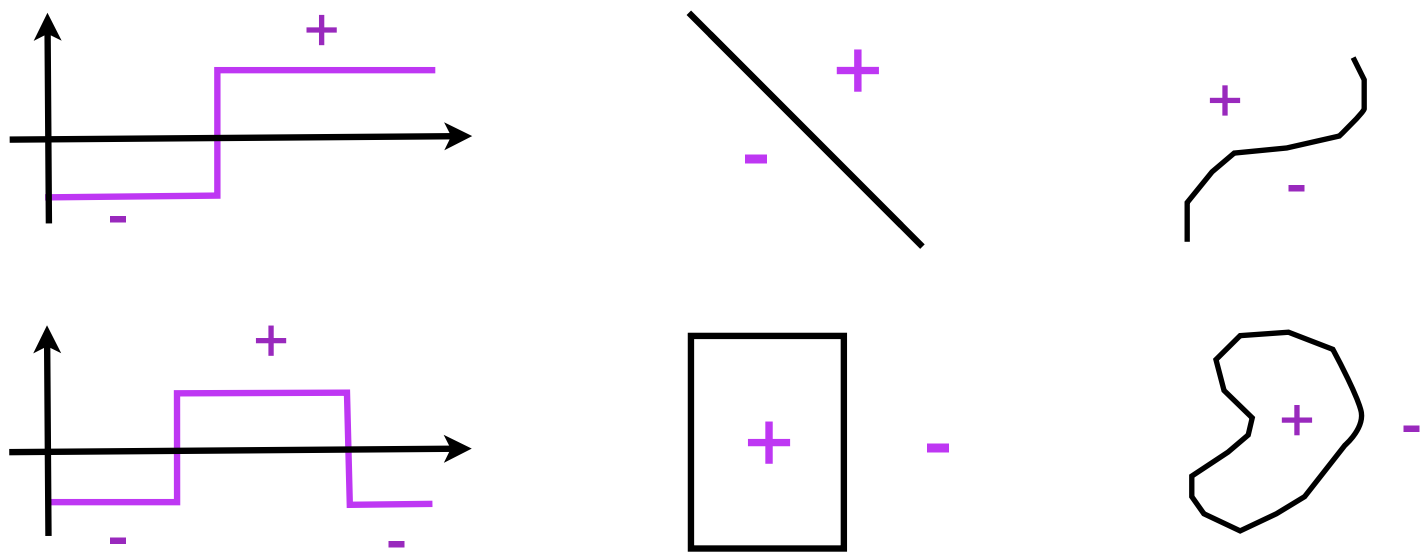
$\mathcal{H}$:= *hypothesis space* $\quad \forall h \in H, \ \ h : \mathcal{X} \to \mathcal{Y}$

Algorithms for Active Learning

$\mathcal{X}$:= domain or *query space*

$\mathcal{Y} := \{-1, +1\}$

$\mathcal{H}$:= *hypothesis space* $\quad \forall h \in H,\ \ h : \mathcal{X} \to \mathcal{Y}$

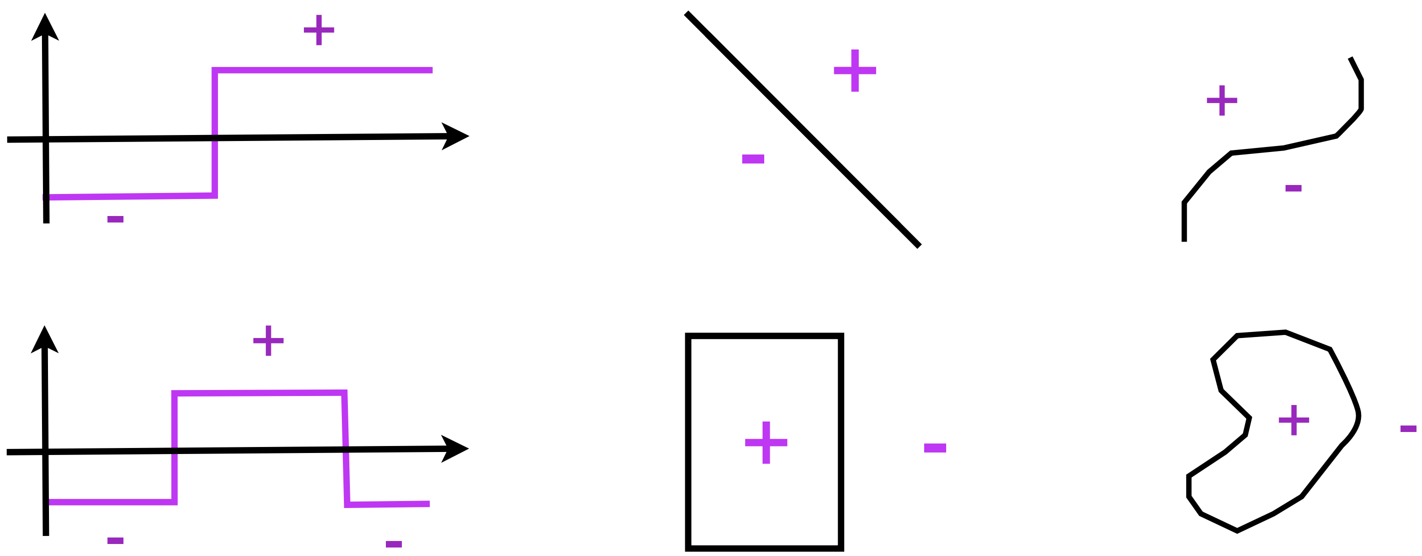


Algorithms for Active Learning

$\mathcal{X}$:= domain or *query space*

$\mathcal{Y} := \{-1, +1\}$

$\mathcal{H}$:= *hypothesis space* $\quad \forall h \in H,\ \ h : \mathcal{X} \to \mathcal{Y}$



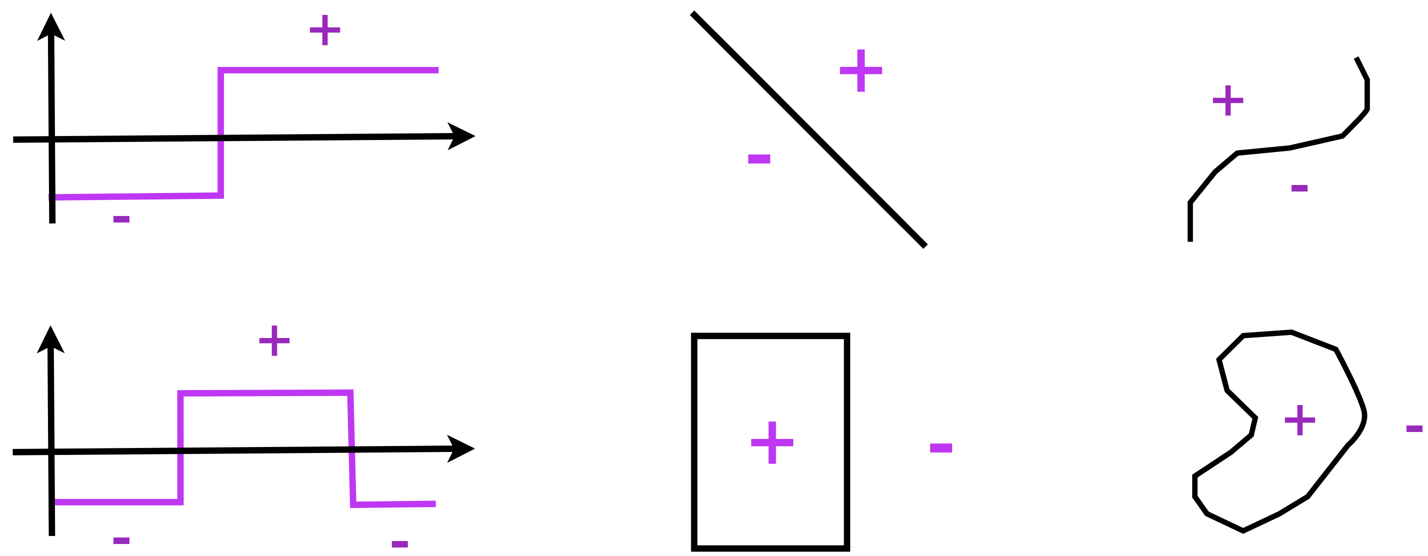
Question: How many queries are required to determine h^*?

Algorithms for Active Learning

$\mathcal{X}$:= domain or *query space*

$\mathcal{Y} := \{-1, +1\}$

$\mathcal{H}$:= *hypothesis space* $\quad \forall h \in H,\ h : \mathcal{X} \to \mathcal{Y}$



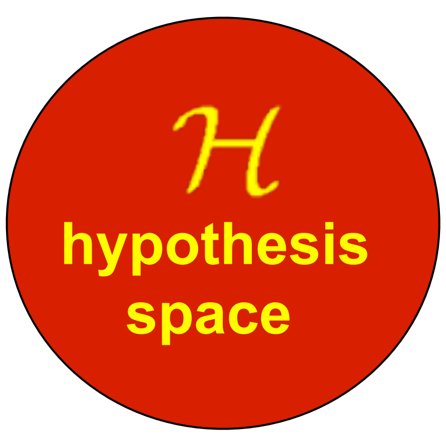
Question: How many queries are required to determine h^*?

If $\mathcal{H}$ is finite with $N := |\mathcal{H}|$, then identification of h^* requires at least $\log_2 N$ bits/queries.

Generalized Binary Search (aka Splitting Algorithm)

initialize: $n = 0$, $\mathcal{H}_0 = \mathcal{H}$

while $|\mathcal{H}_n| > 1$

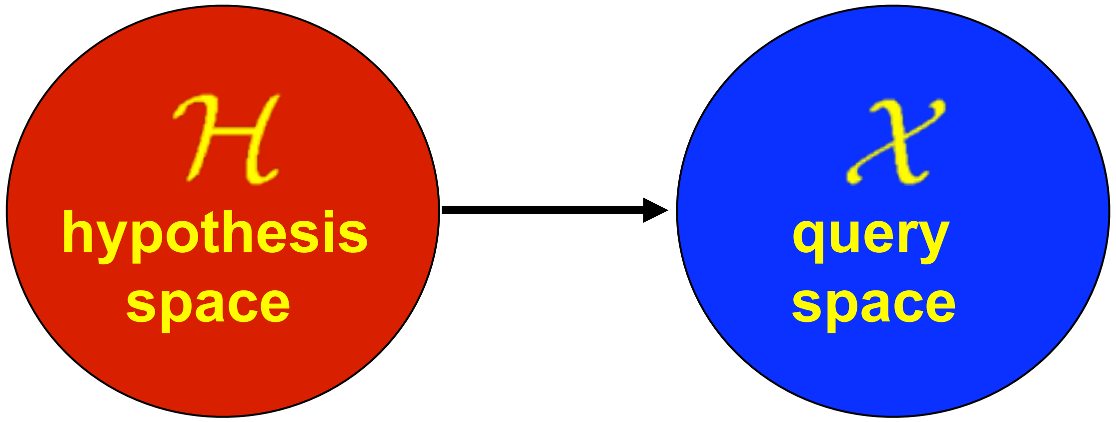


Generalized Binary Search (aka Splitting Algorithm)

initialize: $n = 0$, $\mathcal{H}_0 = \mathcal{H}$

while $|\mathcal{H}_n| > 1$

1) Select $x_n = \arg\min_{x \in \mathcal{X}} |\sum_{h \in \mathcal{H}_n} h(x)|$.



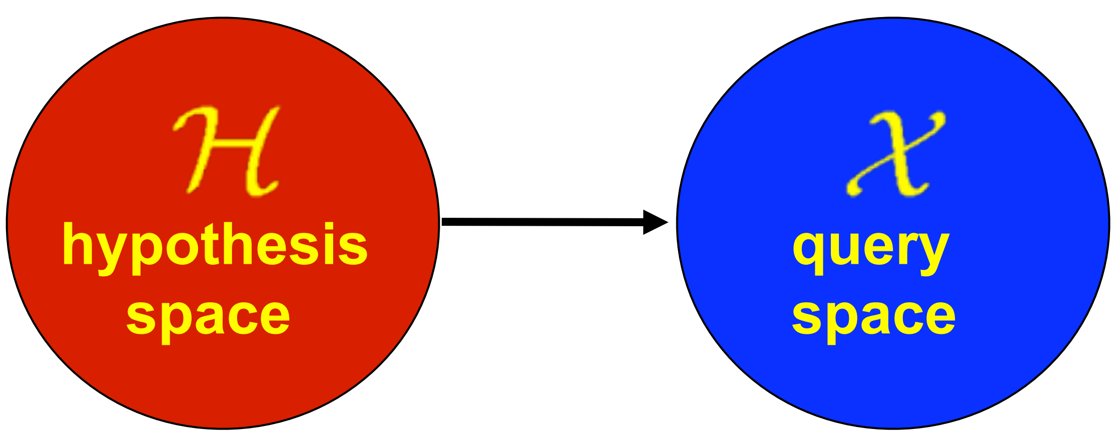
Generalized Binary Search (aka Splitting Algorithm)

initialize: $n = 0$, $\mathcal{H}_0 = \mathcal{H}$

while $|\mathcal{H}_n| > 1$

1) Select $x_n = \arg\min_{x \in \mathcal{X}} |\sum_{h \in \mathcal{H}_n} h(x)|$.

Selects a query for which disagreement among hypotheses is maximal



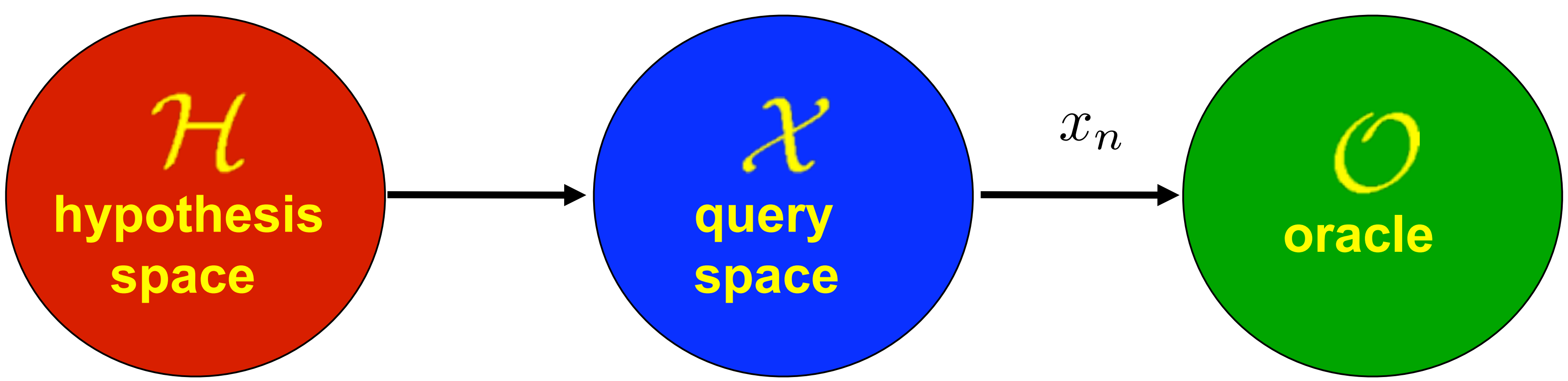
Generalized Binary Search (aka Splitting Algorithm)

initialize: $n = 0$, $\mathcal{H}_0 = \mathcal{H}$

while $|\mathcal{H}_n| > 1$

1) Select $x_n = \arg\min_{x \in \mathcal{X}} |\sum_{h \in \mathcal{H}_n} h(x)|$.

2) Query with x_n to obtain response $y_n = h^*(x_n)$.



Generalized Binary Search (aka Splitting Algorithm)

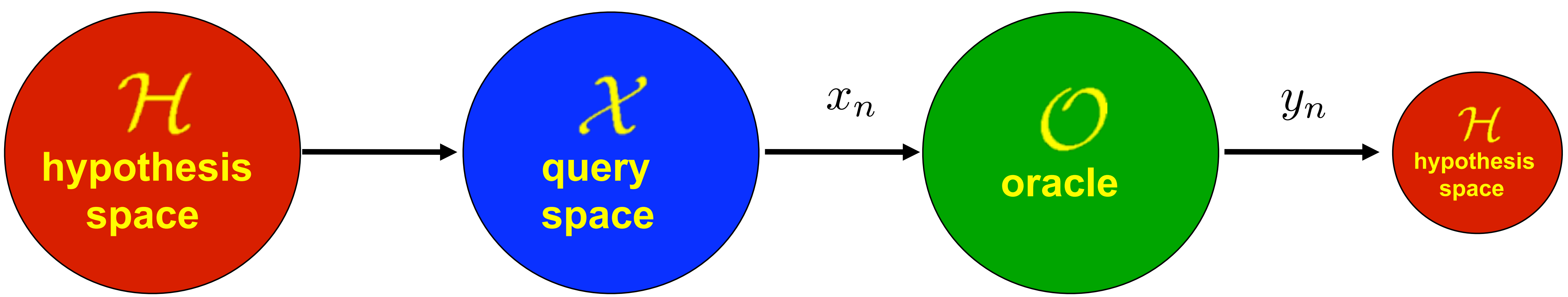
initialize: $n = 0$, $\mathcal{H}_0 = \mathcal{H}$

while $|\mathcal{H}_n| > 1$

1) Select $x_n = \arg\min_{x \in \mathcal{X}} |\sum_{h \in \mathcal{H}_n} h(x)|$.

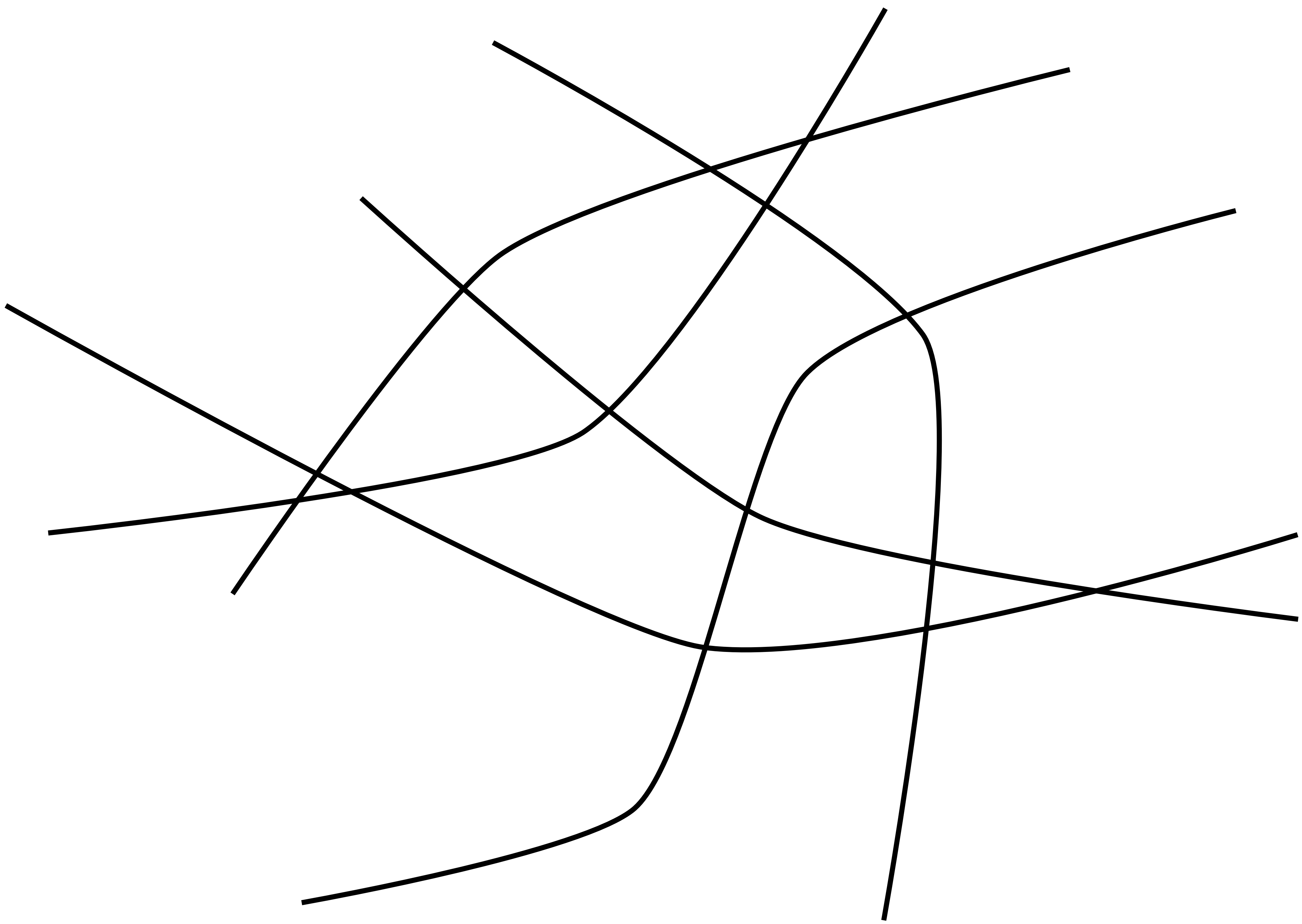
2) Query with x_n to obtain response $y_n = h^*(x_n)$.

3) Set $\mathcal{H}_{n+1} = \{h \in \mathcal{H}_n : h(x_n) = y_n\}$, $n = n + 1$.



Bisection in Higher Dimensions

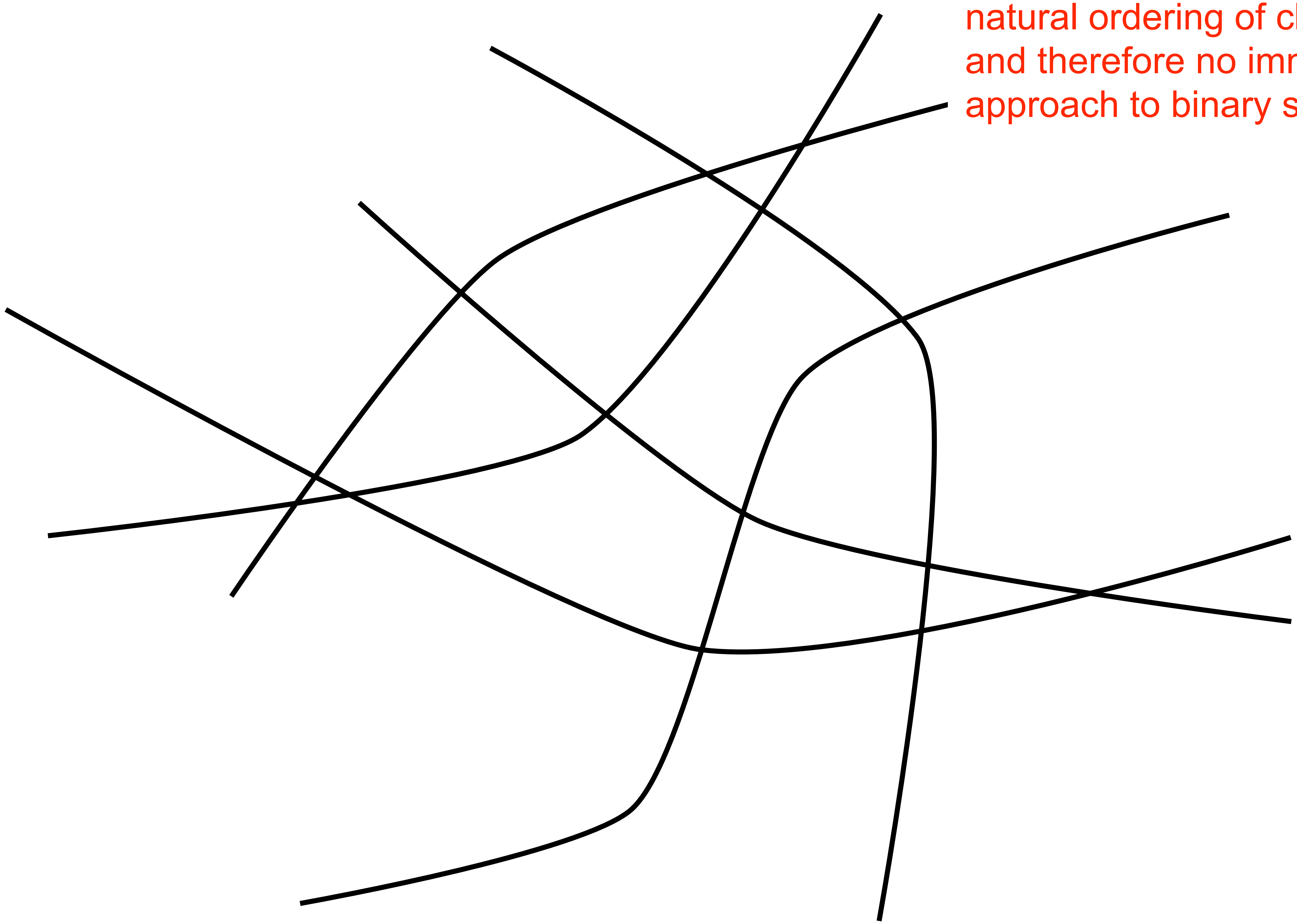
Consider the decision boundaries of a collection of classifiers in a multidimensional feature space



Bisection in Higher Dimensions

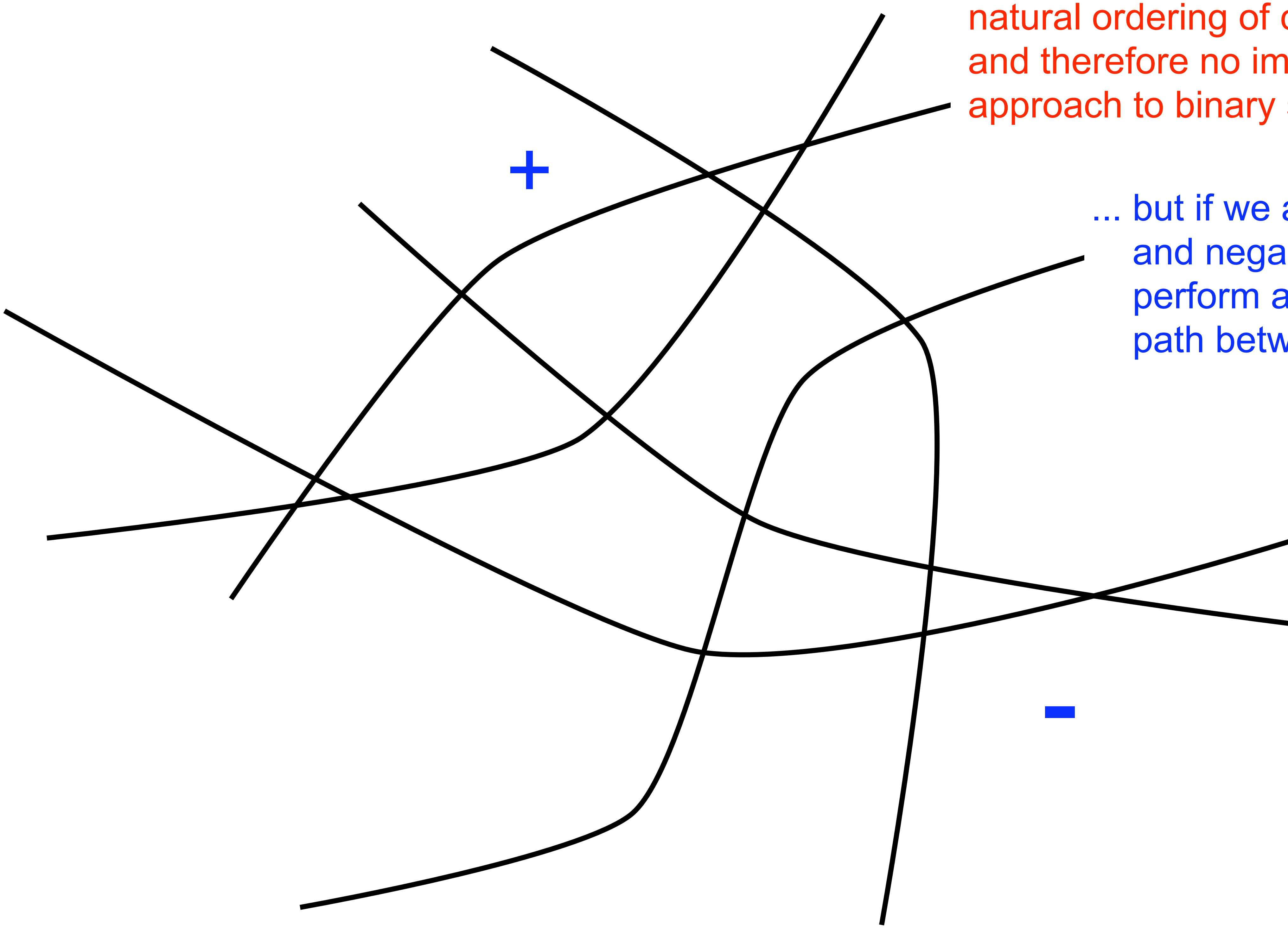
Consider the decision boundaries of a collection of classifiers in a multidimensional feature space

Unlike the situation in 1-d, there is no natural ordering of classifiers/boundaries and therefore no immediately obvious approach to binary search



Bisection in Higher Dimensions

Consider the decision boundaries of a collection of classifiers in a multidimensional feature space



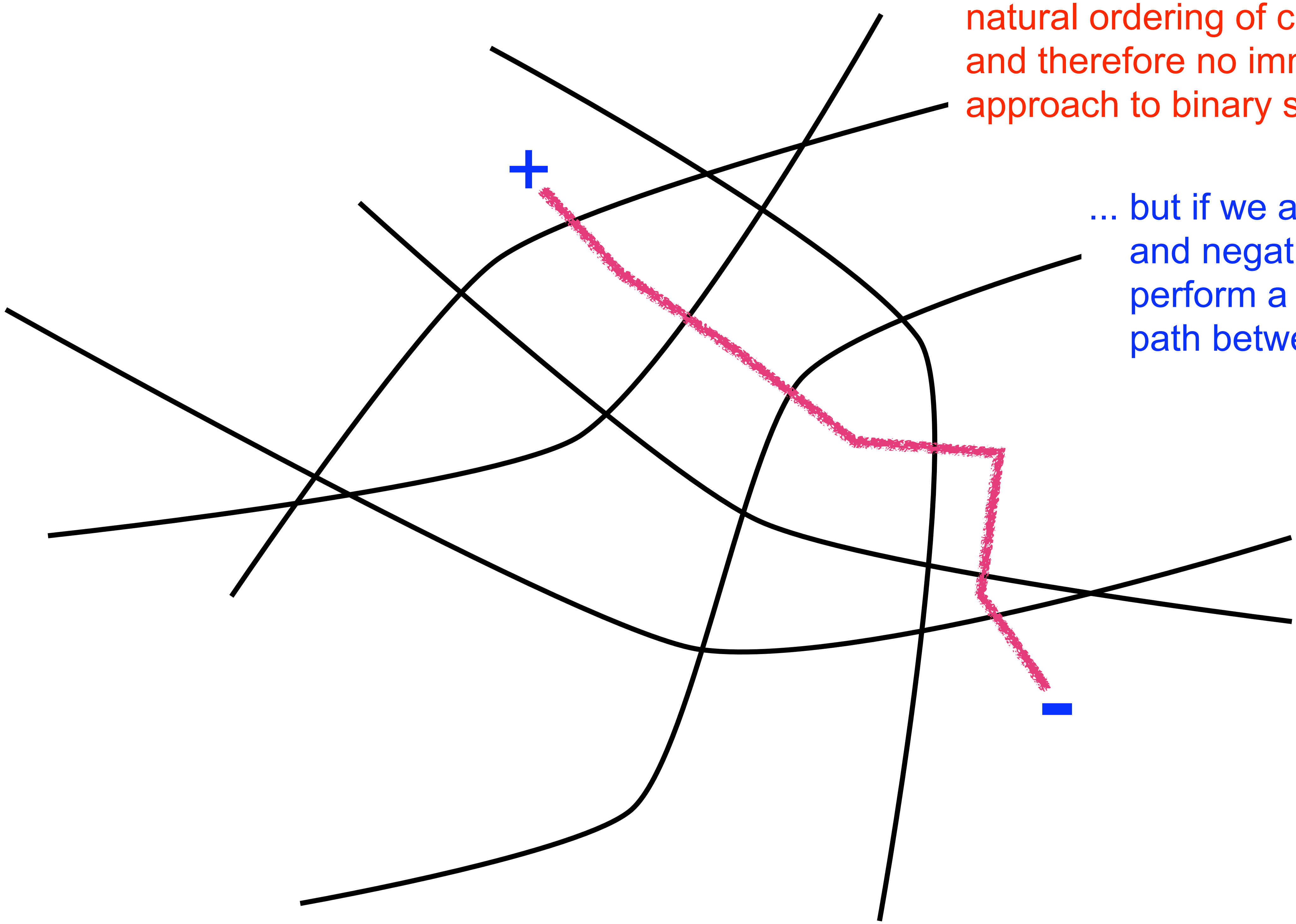
Unlike the situation in 1-d, there is no natural ordering of classifiers/boundaries and therefore no immediately obvious approach to binary search

... but if we are given one positive and negative example, then we perform a bisection along the path between these points

Bisection in Higher Dimensions

Consider the decision boundaries of a collection of classifiers in a multidimensional feature space

Unlike the situation in 1-d, there is no natural ordering of classifiers/boundaries and therefore no immediately obvious approach to binary search



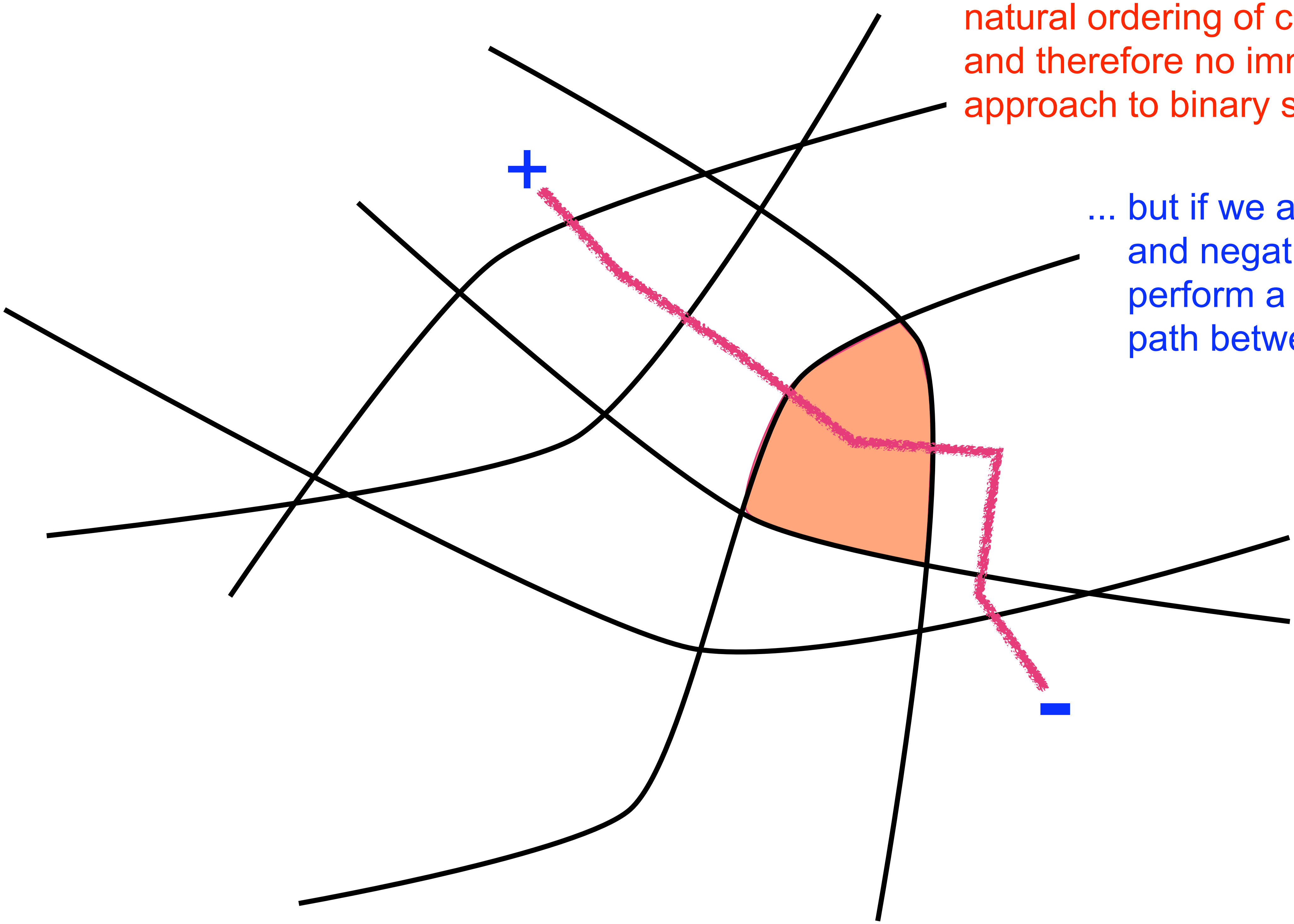
... but if we are given one positive and negative example, then we perform a bisection along the path between these points

Bisection in Higher Dimensions

Consider the decision boundaries of a collection of classifiers in a multidimensional feature space

Unlike the situation in 1-d, there is no natural ordering of classifiers/boundaries and therefore no immediately obvious approach to binary search

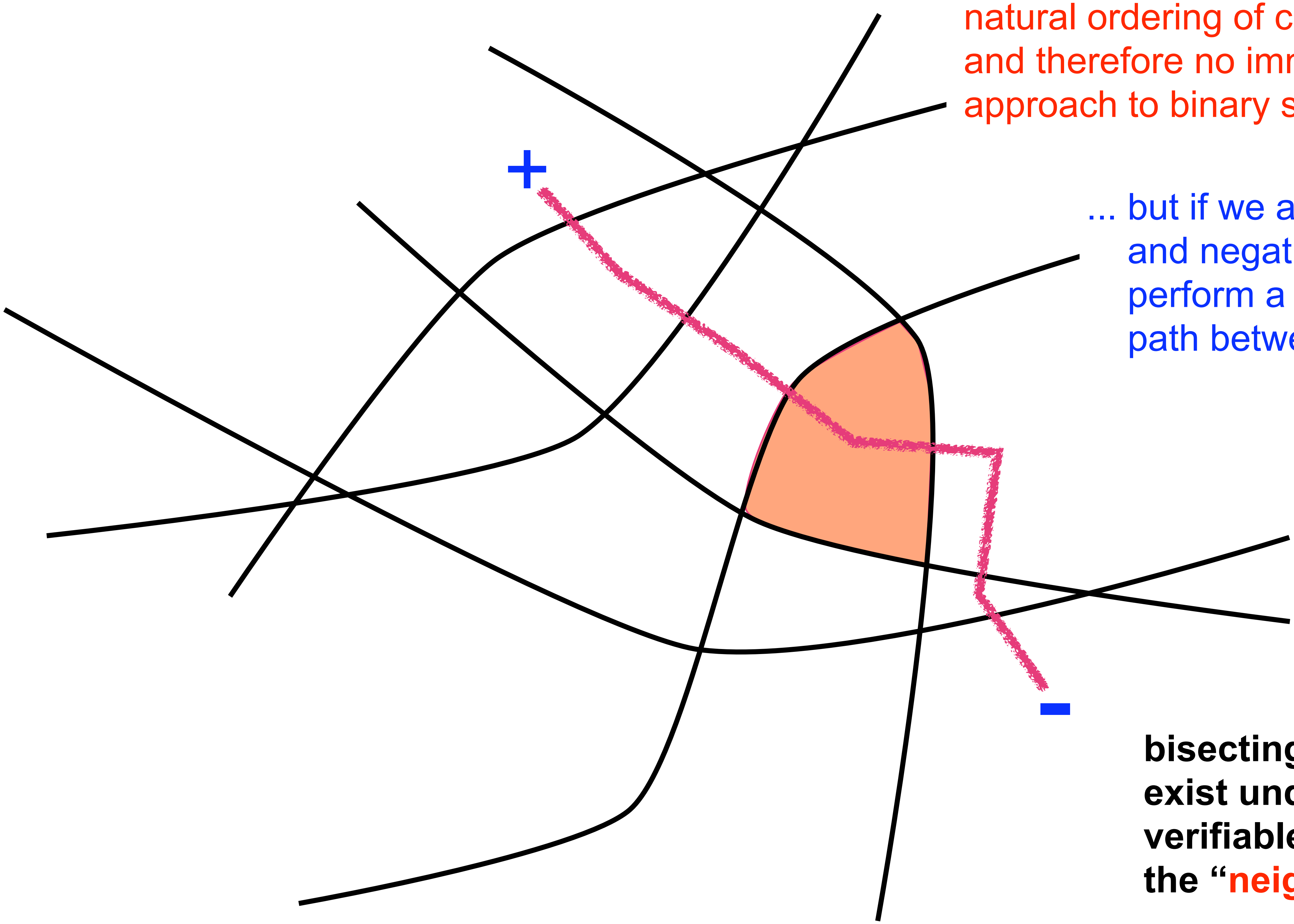
... but if we are given one positive and negative example, then we perform a bisection along the path between these points



Bisection in Higher Dimensions

Consider the decision boundaries of a collection of classifiers in a multidimensional feature space

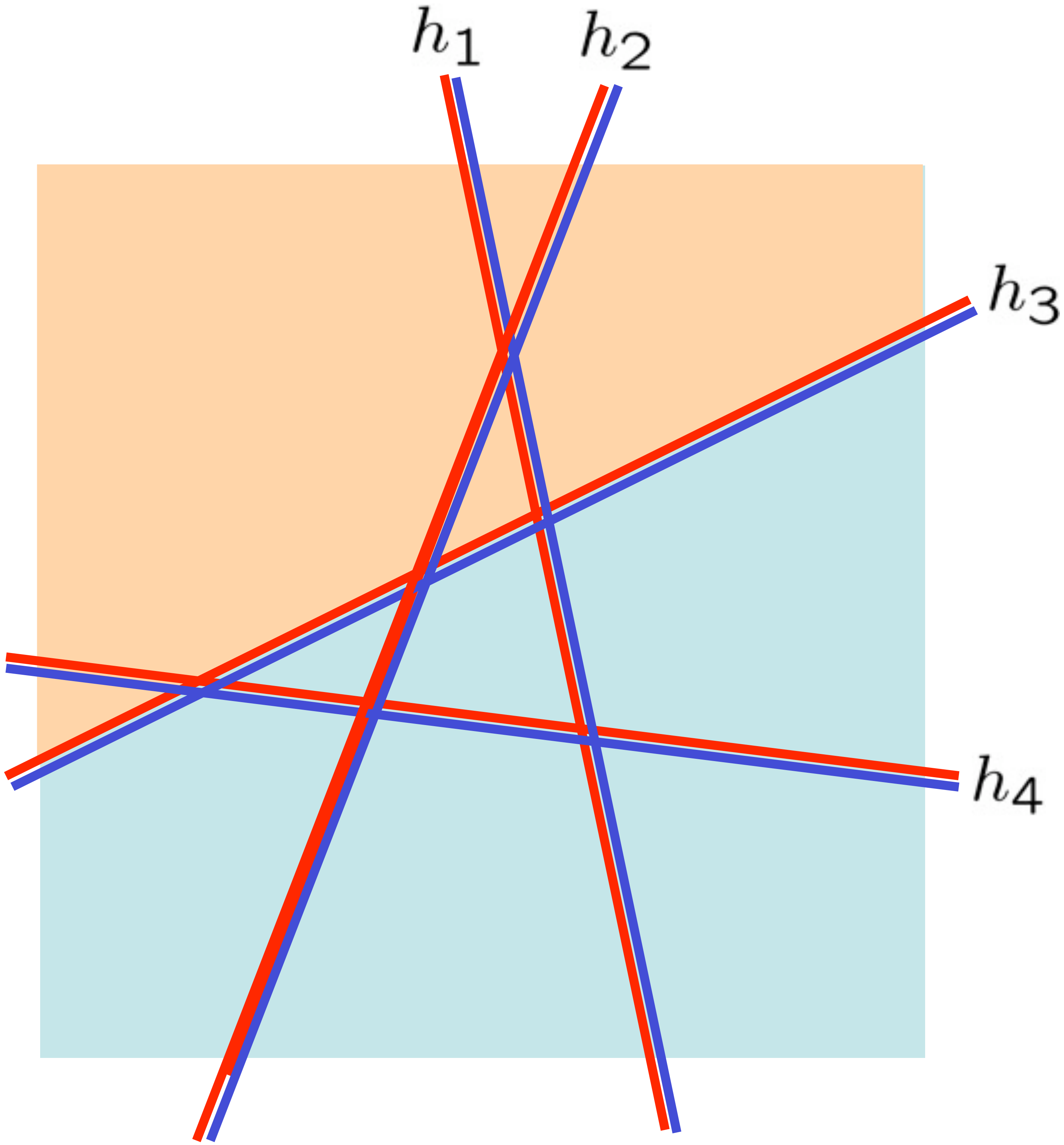
Unlike the situation in 1-d, there is no natural ordering of classifiers/boundaries and therefore no immediately obvious approach to binary search



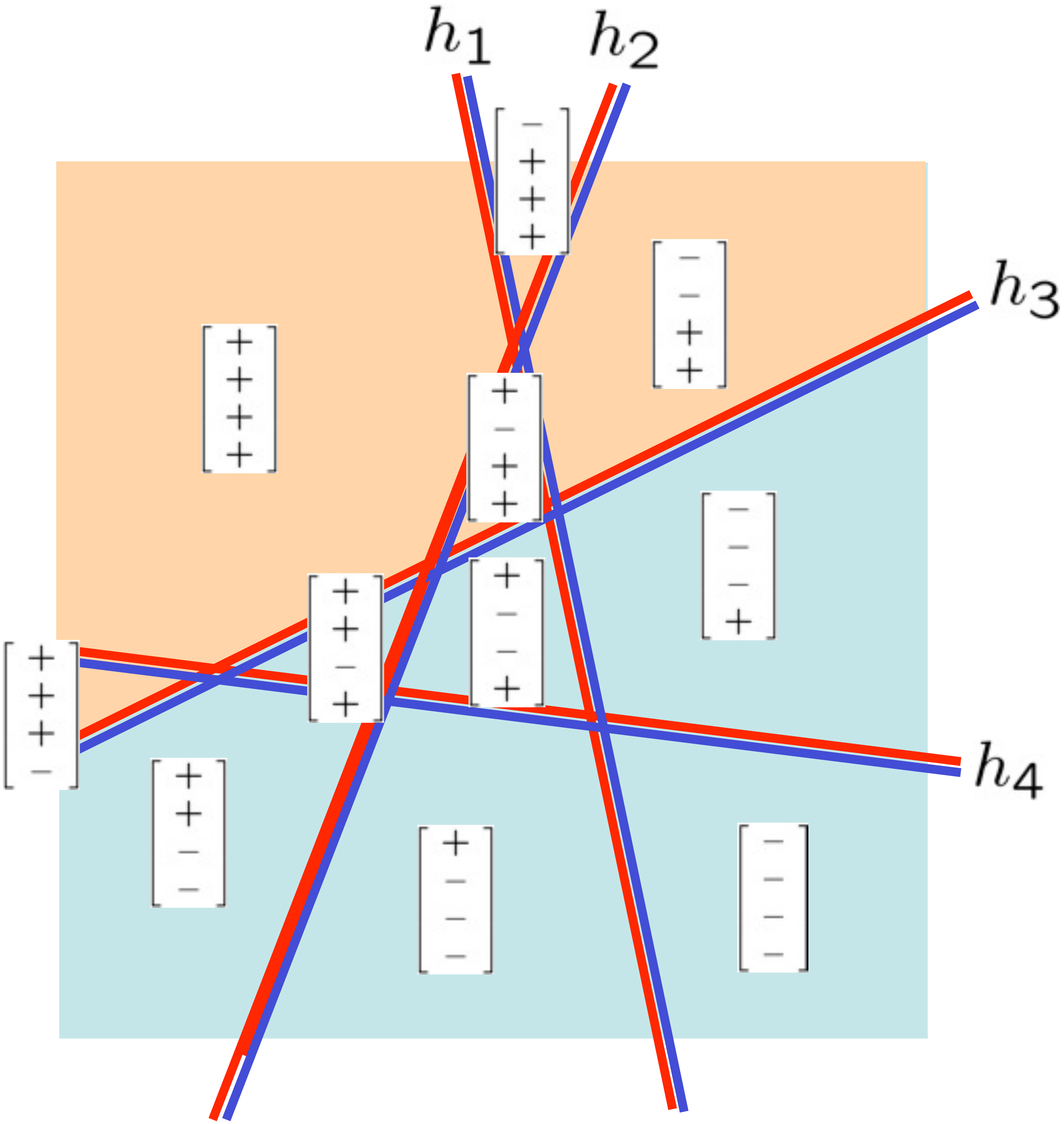
... but if we are given one positive and negative example, then we perform a bisection along the path between these points

bisecting paths of this sort exist under a mild and verifiable property we call the "neighborly condition"

Learning Halfspaces in $\mathbb{R}^d$

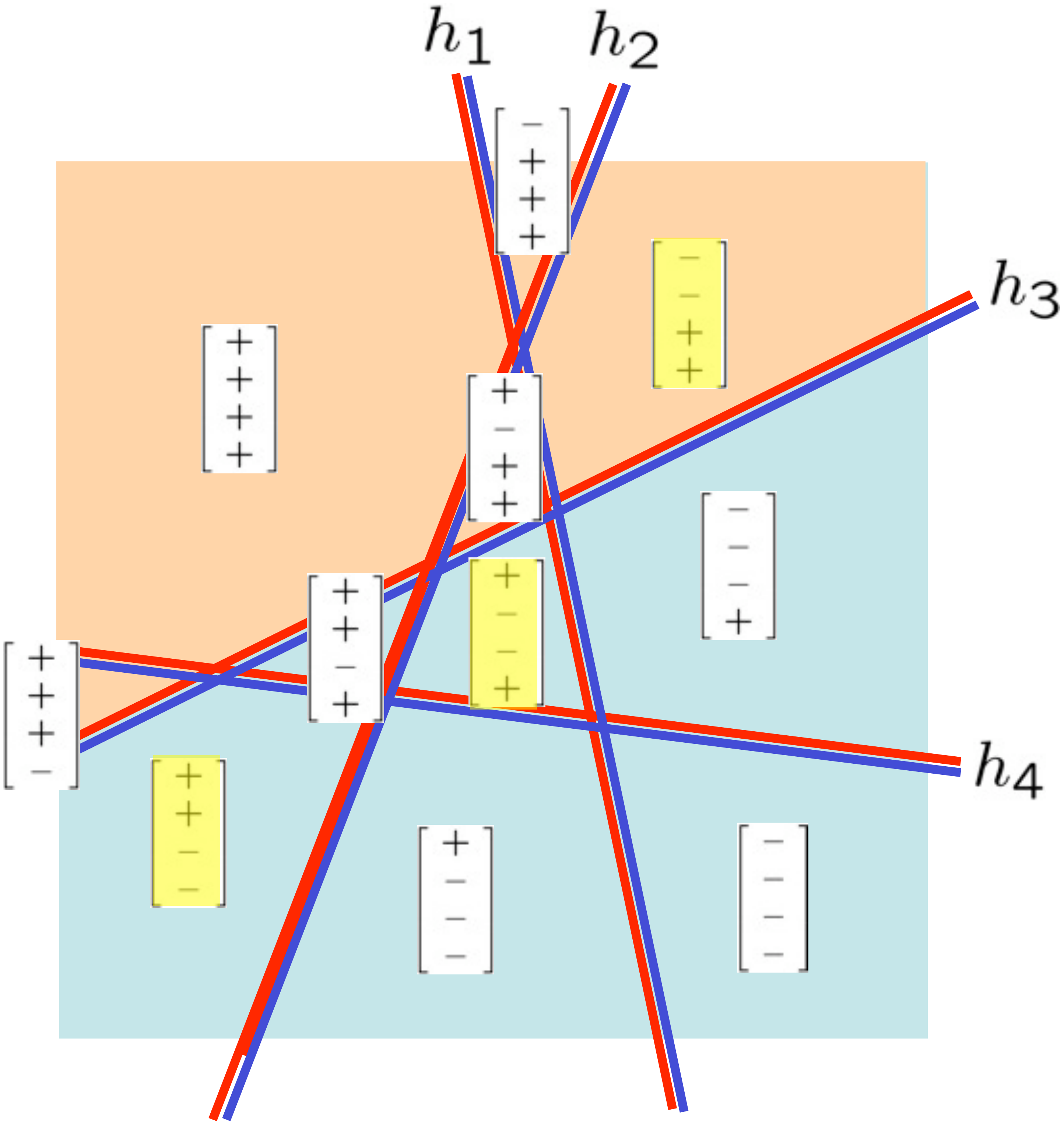


Learning Halfspaces in $\mathbb{R}^d$



$$
\begin{array}{c}
\\ h_1 \\ h_2 \\ h_3 \\ h_4
\end{array}
\begin{array}{c}
\begin{array}{ccccccccccc} A_1 & A_2 & A_3 & A_4 & A_5 & A_6 & A_7 & A_8 & A_9 & A_{10} & A_{11} \end{array} \\
\left[\begin{array}{ccccccccccc}
+ & - & - & - & - & + & + & + & + & + & + \\
+ & + & - & - & - & - & + & + & + & - & - \\
+ & + & + & - & - & - & - & + & - & - & + \\
+ & + & + & + & - & - & - & - & + & + & +
\end{array}\right]
\end{array}
$$

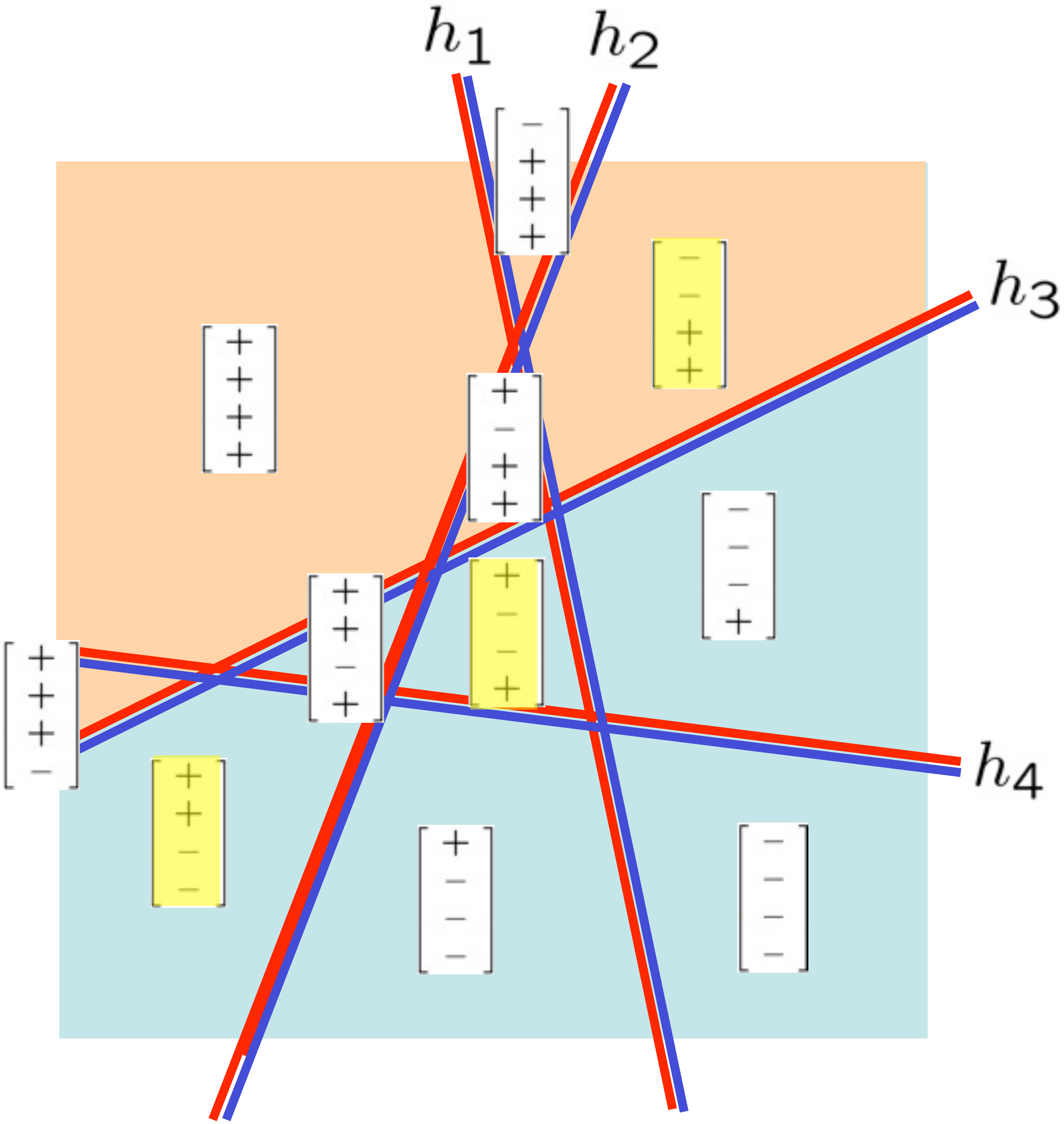
Learning Halfspaces in $\mathbb{R}^d$



$$
\begin{array}{c}
 \\ h_1 \\ h_2 \\ h_3 \\ h_4
\end{array}
\begin{array}{c}
\begin{array}{ccccccccccc} A_1 & A_2 & A_3 & A_4 & A_5 & A_6 & A_7 & A_8 & A_9 & A_{10} & A_{11} \end{array} \\
\left[\begin{array}{ccccccccccc}
+ & - & - & - & - & + & + & + & + & + & + \\
+ & + & - & - & - & - & + & + & + & - & - \\
+ & + & + & - & - & - & - & + & - & - & + \\
+ & + & + & + & - & - & - & - & + & + & +
\end{array}\right]
\end{array}
$$

bisecting queries

Learning Halfspaces in $\mathbb{R}^d$

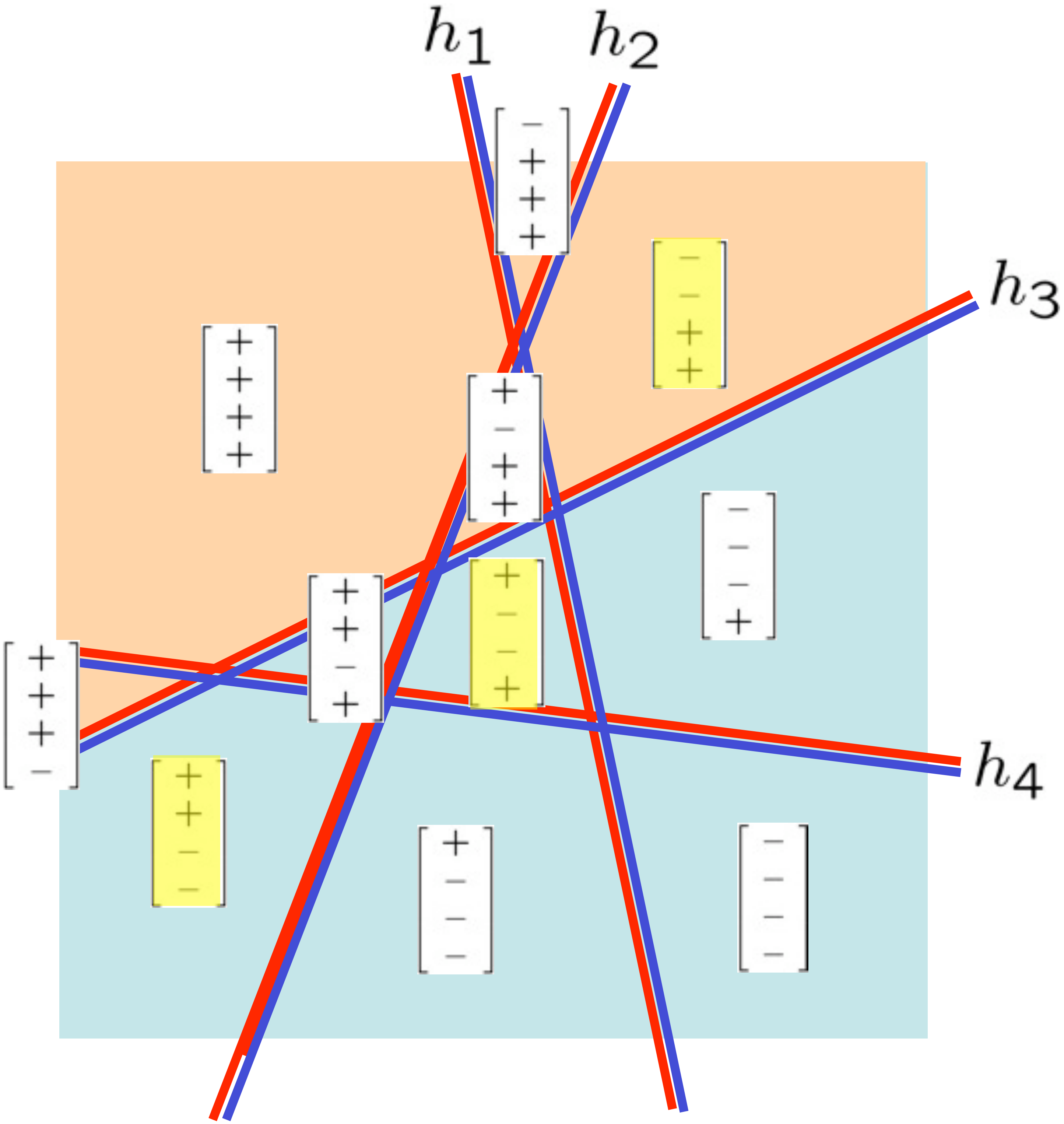


$$
\begin{array}{c}
 \\ h_1 \\ h_2 \\ h_3 \\ h_4
\end{array}
\begin{array}{c}
\begin{array}{ccccccccccc} A_1 & A_2 & A_3 & A_4 & A_5 & A_6 & A_7 & A_8 & A_9 & A_{10} & A_{11} \end{array} \\
\begin{bmatrix}
+ & - & - & - & - & + & + & + & + & + & + \\
+ & + & - & - & - & - & + & + & + & - & - \\
+ & + & + & - & - & - & - & + & - & - & + \\
+ & + & + & + & - & - & - & - & + & + & +
\end{bmatrix}
\end{array}
$$

bisecting queries

queries generate only $O(N^d)$ of the possible 2^N binary patterns!

Learning Halfspaces in $\mathbb{R}^d$



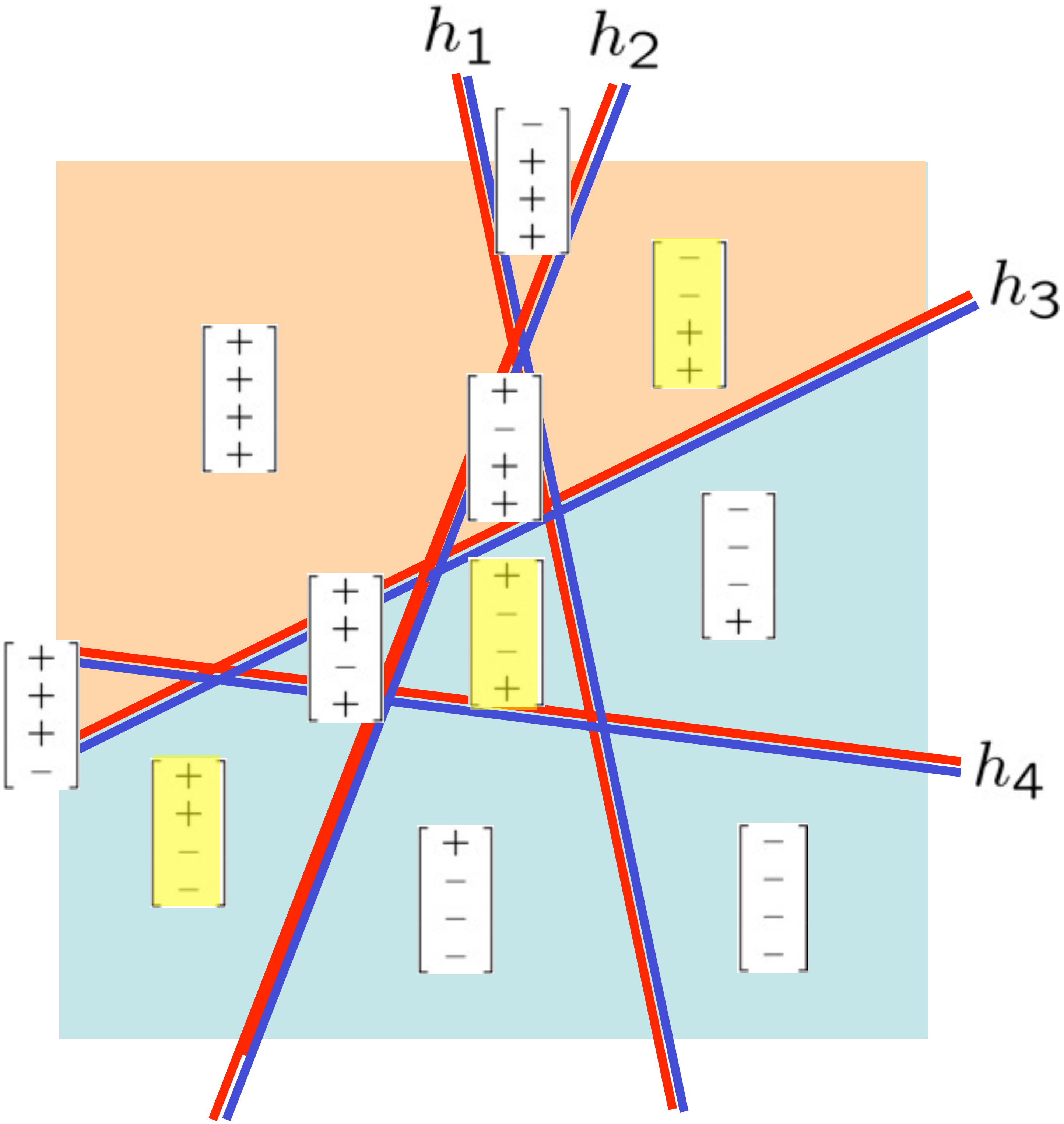
$$
\begin{array}{c|ccccccccccc}
 & A_1 & A_2 & A_3 & A_4 & A_5 & A_6 & A_7 & A_8 & A_9 & A_{10} & A_{11} \\
\hline
h_1 & + & - & - & - & - & + & + & + & + & + & + \\
h_2 & + & + & - & - & - & - & + & + & + & - & - \\
h_3 & + & + & + & - & - & - & - & + & - & - & + \\
h_4 & + & + & + & + & - & - & - & - & + & + & +
\end{array}
$$

bisecting queries

queries generate only $O(N^d)$ of the possible 2^N binary patterns!

Can GBS find near-bisecting queries in general?

Learning Halfspaces in $\mathbb{R}^d$



$$
\begin{array}{c|ccccccccccc}
 & A_1 & A_2 & A_3 & A_4 & A_5 & A_6 & A_7 & A_8 & A_9 & A_{10} & A_{11} \\
\hline
h_1 & + & - & - & - & - & + & + & + & + & + & + \\
h_2 & + & + & - & - & - & - & + & + & + & - & - \\
h_3 & + & + & + & - & - & - & - & + & - & - & + \\
h_4 & + & + & + & + & - & - & - & - & + & + & +
\end{array}
$$

bisecting queries

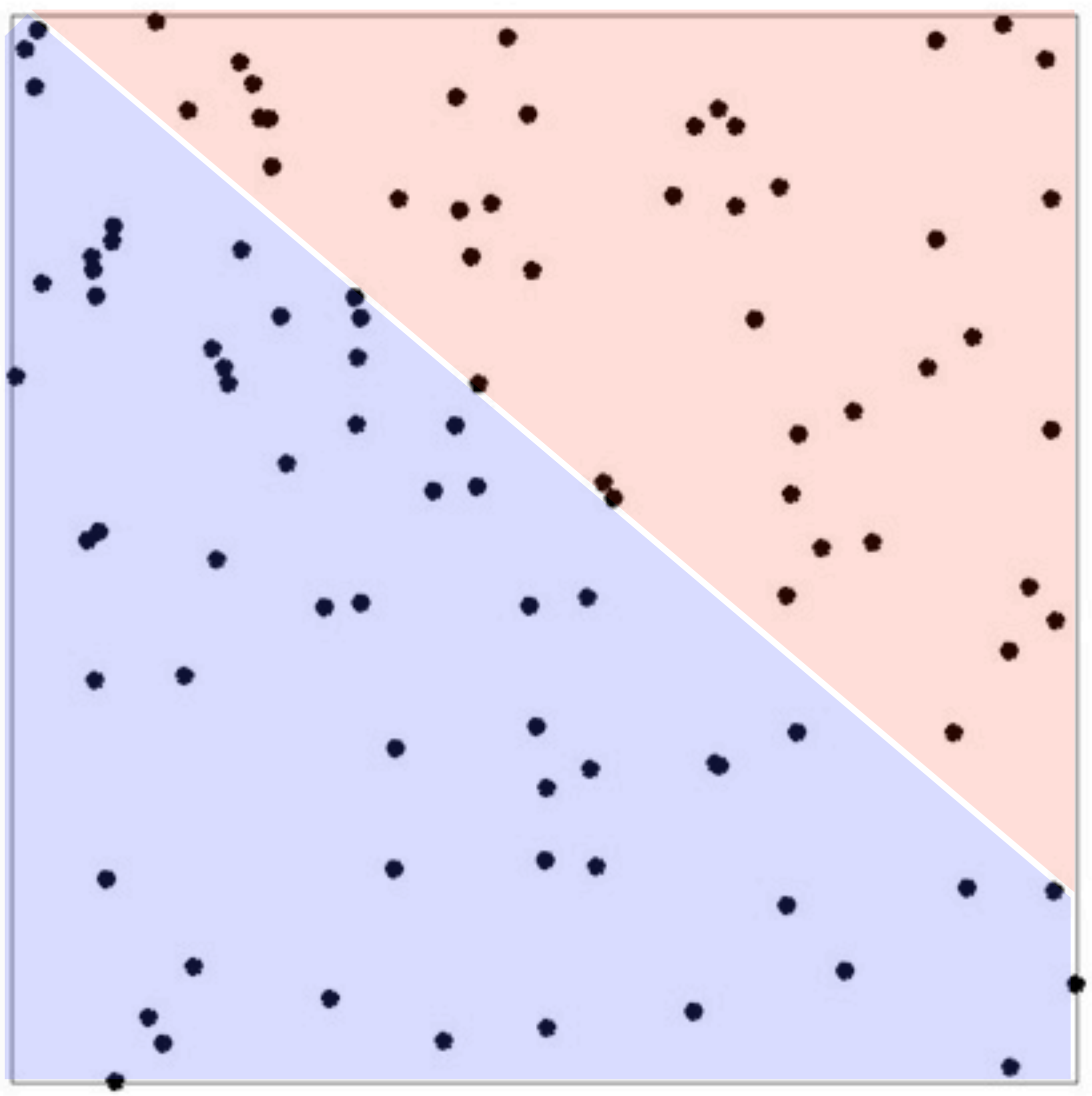
queries generate only $O(N^d)$ of the possible 2^N binary patterns!

Can GBS find near-bisecting queries in general?

If $\mathcal{H}$ is a collection of N halfspaces on $\mathcal{X} = \mathbb{R}^d$, then GBS terminates with the correct halfspace after $O(\log N)$ queries.

Example

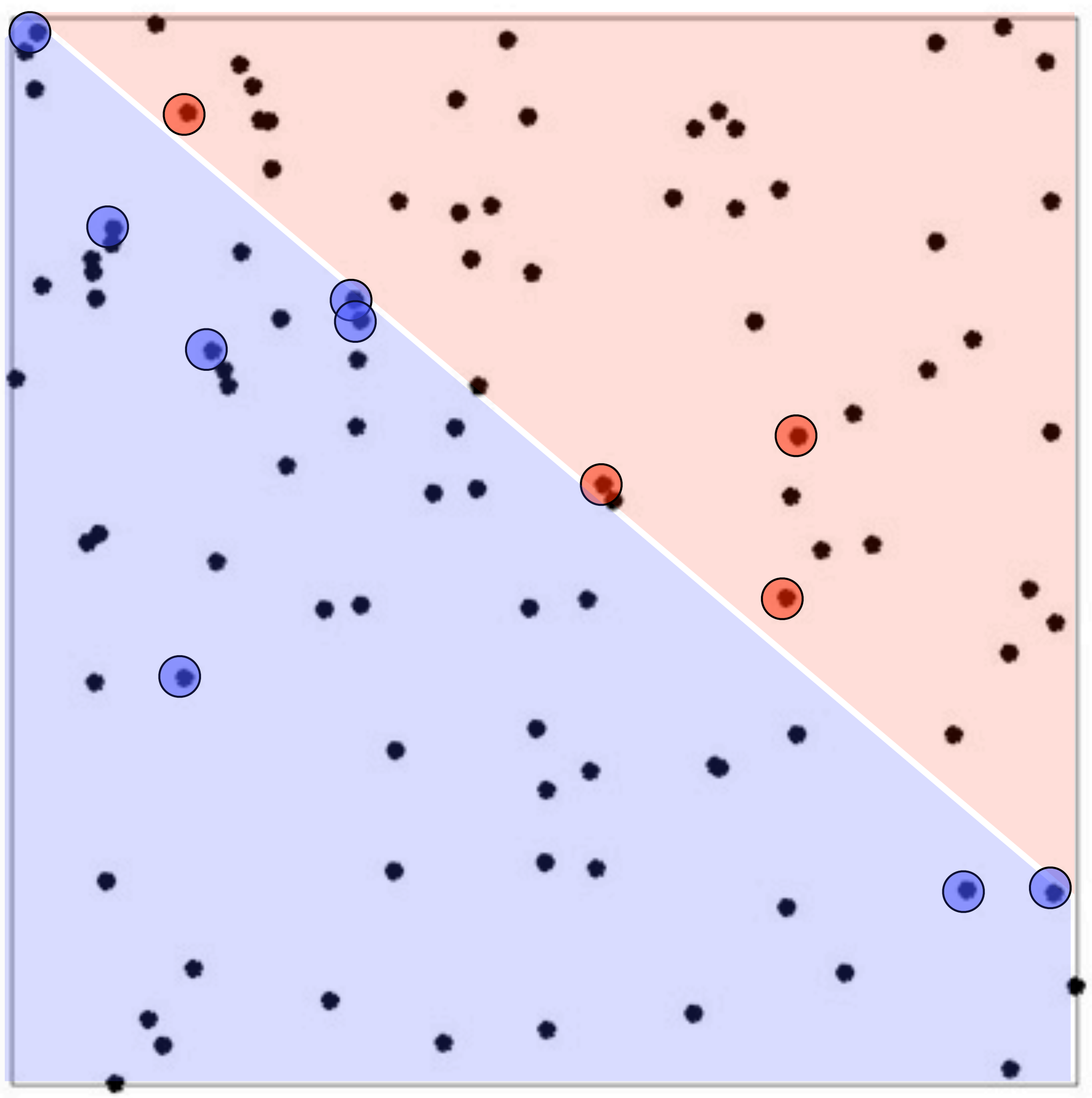
Suppose we have a sensor network observing a binary activation pattern with a linear boundary. How many sensors must be queried to determine the pattern?



100 sensors, 9900 possible linear boundaries

Example

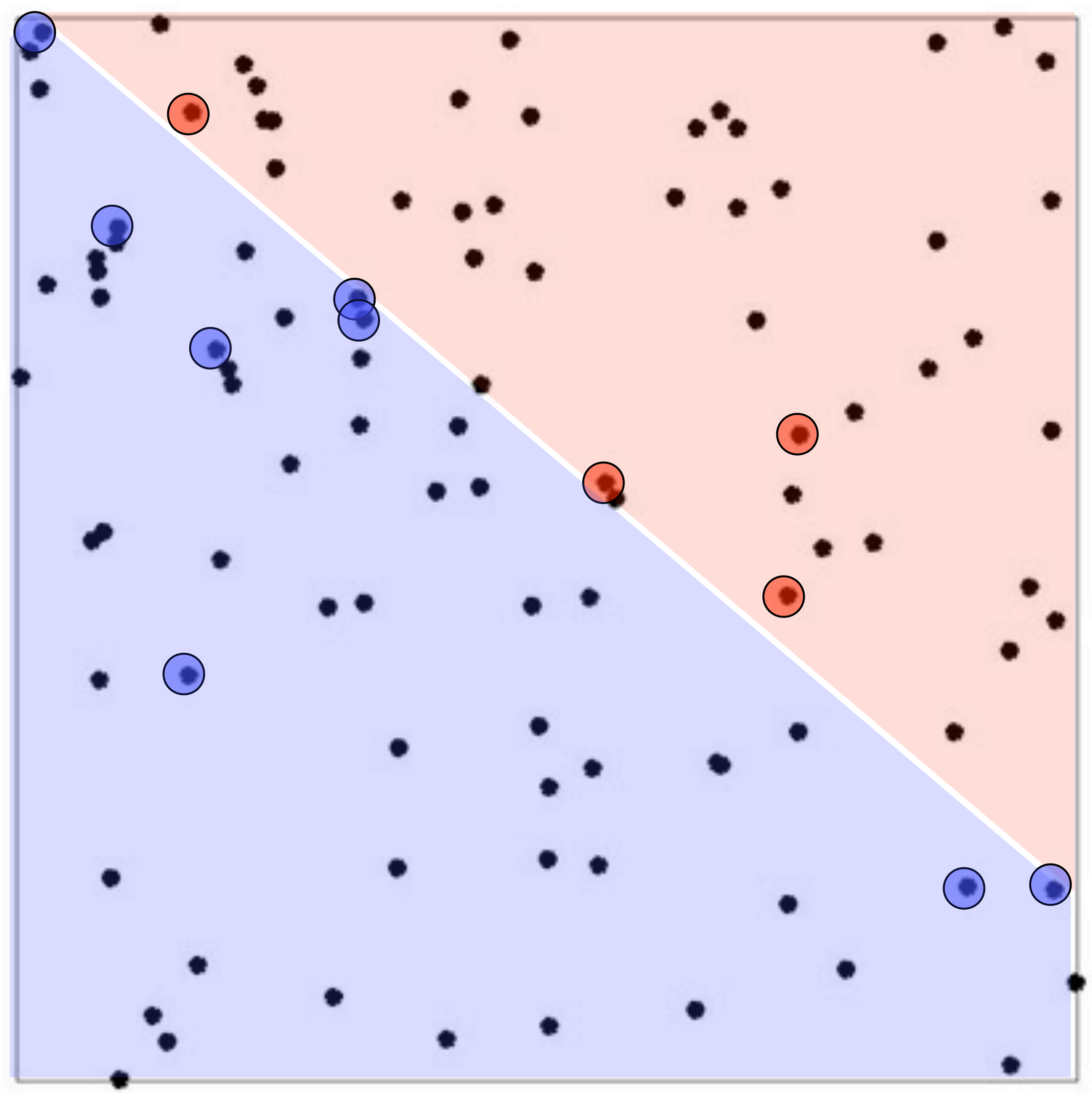
Suppose we have a sensor network observing a binary activation pattern with a linear boundary. How many sensors must be queried to determine the pattern?

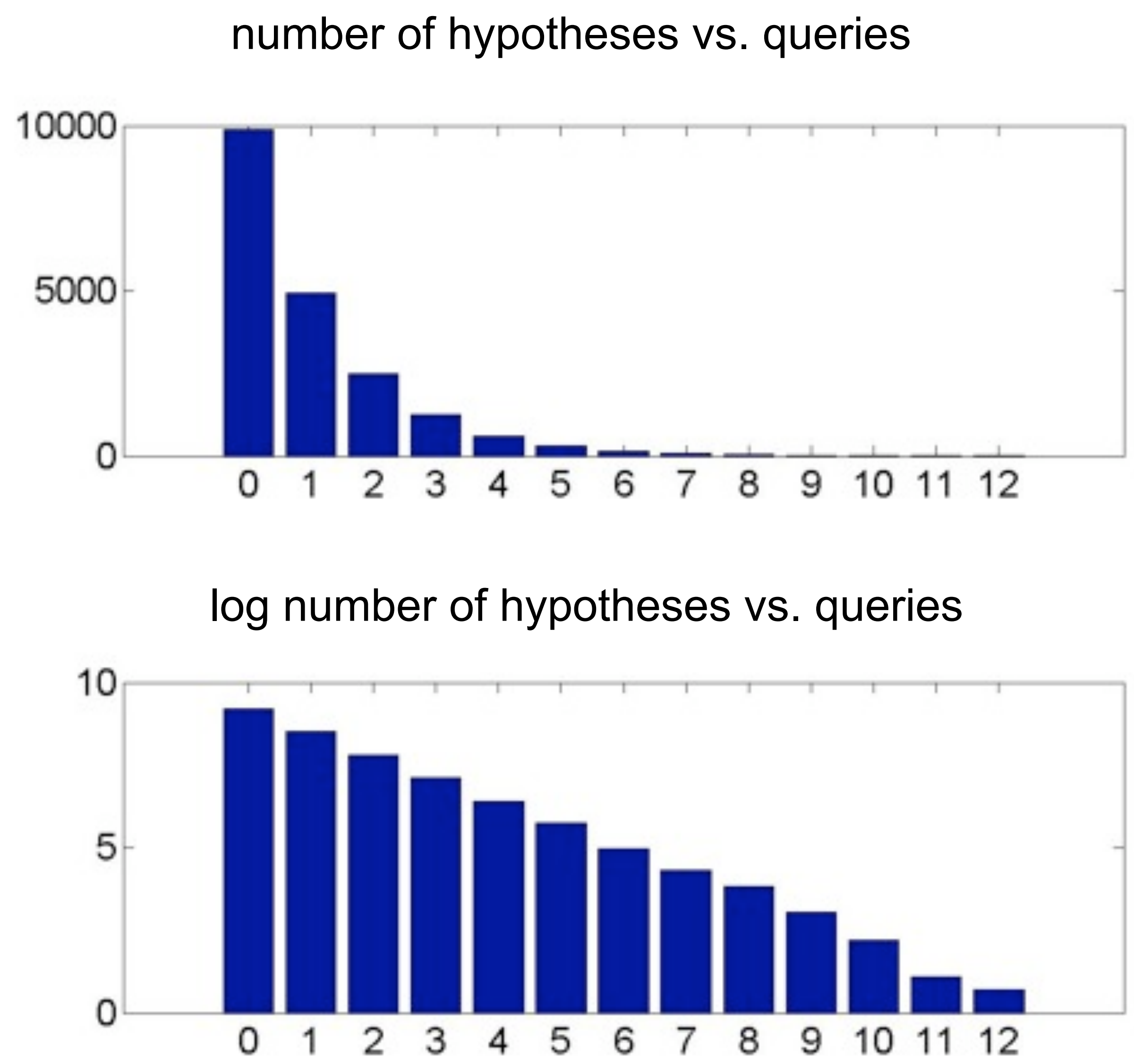


100 sensors, 9900 possible linear boundaries

Example

Suppose we have a sensor network observing a binary activation pattern with a linear boundary. How many sensors must be queried to determine the pattern?



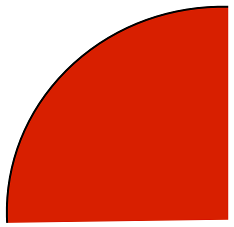


Correct boundary determined after querying 12 sensors

$\mathcal{H}$

hypothesis space

“Is the person wearing a hat ?”



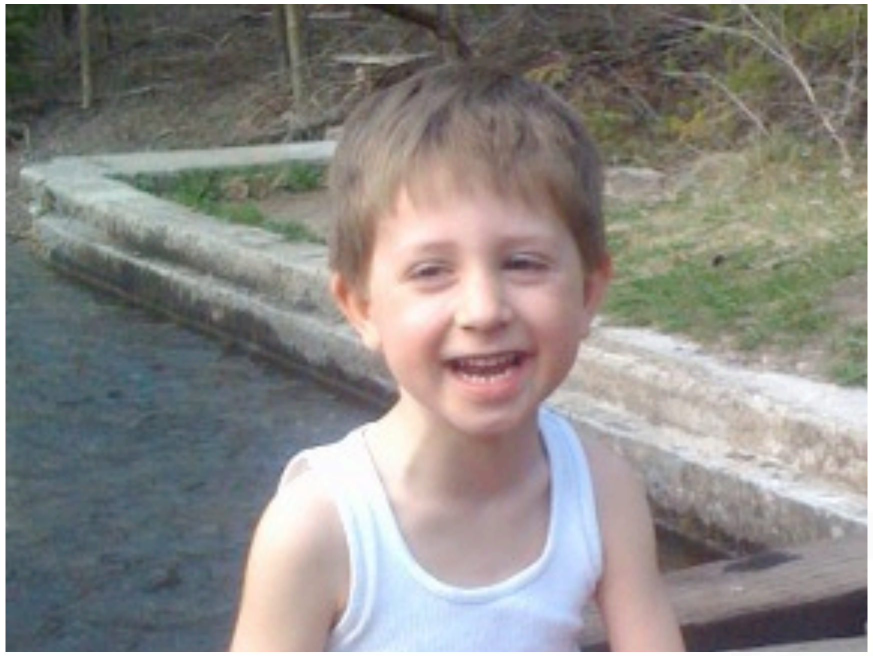
“Is the person wearing a hat ?”

“Does the person have blue eyes ?”

“Is the person wearing a hat ?”

“Does the person have blue eyes ?”

GBS is quite effective if responses are reliable



“Is the person
wearing a hat ?”

“Does the person
have blue eyes ?”

GBS is quite effective if responses are reliable

Generalized Binary Search with Noise

Generalized Binary Search (GBS)

initialize: $n = 0$, $\mathcal{H}_0 = \mathcal{H}$

while $|\mathcal{H}_n| > 1$

1) Select $x_n = \arg\min_{x \in \mathcal{X}} |\sum_{h \in \mathcal{H}_n} h(x)|$

2) Query with x_n to obtain response $y_n = h^*(x_n)$

3) Set $\mathcal{H}_{n+1} = \{h \in \mathcal{H}_n : h(x_n) = y_n\}$, $n = n + 1$

Generalized Binary Search with Noise

Generalized Binary Search (GBS)

initialize: $n = 0$, $\mathcal{H}_0 = \mathcal{H}$

while $|\mathcal{H}_n| > 1$

1) Select $x_n = \arg\min_{x \in \mathcal{X}} |\sum_{h \in \mathcal{H}_n} h(x)|$

2) Query with x_n to obtain response $y_n = h^*(x_n)$

3) Set $\mathcal{H}_{n+1} = \{h \in \mathcal{H}_n : h(x_n) = y_n\}$, $n = n + 1$

Suppose that the binary response $y \in \{-1, 1\}$ to query $x \in \mathcal{X}$ is an independent realization of the random variable Y satisfying $\mathbb{P}(Y = h^*(x)) > \mathbb{P}(Y = -h^*(x))$, where $h^* \in \mathcal{H}$ is fixed but unknown (i.e., the response is only probably correct)

Generalized Binary Search with Noise

Generalized Binary Search (GBS)

initialize: $n = 0$, $\mathcal{H}_0 = \mathcal{H}$

while $|\mathcal{H}_n| > 1$

1) Select $x_n = \arg\min_{x \in \mathcal{X}} |\sum_{h \in \mathcal{H}_n} h(x)|$

2) Query with x_n to obtain response $y_n = h^*(x_n)$

3) Set $\mathcal{H}_{n+1} = \{h \in \mathcal{H}_n : h(x_n) = y_n\}$, $n = n + 1$

Suppose that the binary response $y \in \{-1, 1\}$ to query $x \in \mathcal{X}$ is an independent realization of the random variable Y satisfying $\mathbb{P}(Y = h^*(x)) > \mathbb{P}(Y = -h^*(x))$, where $h^* \in \mathcal{H}$ is fixed but unknown (i.e., the response is only probably correct)

The *noise bound* is defined as $\alpha := \sup_{x \in \mathcal{X}} \mathbb{P}(Y \neq h^*(x))$

Generalized Binary Search with Noise

Noise-tolerant GBS

initialize: p_0 uniform over $\mathcal{H}$ and $\alpha < \beta < 1/2$.

for $n = 0, 1, 2, \ldots$

1) $x_n = \arg\min_{x \in \mathcal{X}} |\sum_{h \in \mathcal{H}} p_n(h) h(x)|$

2) Obtain noisy response y_n

3) Bayes update: $\forall\, h$

$$p_{n+1}(h) \;\propto\; p_n(h) \times \begin{cases} 1-\beta & , h(x_n) = y_n \\ \beta & , h(x_n) \neq y_n \end{cases}$$

hypothesis selected at each step:

$\widehat{h}_n := \arg\max_{h \in H} p_n(h)$

Suppose that the binary response $y \in \{-1, 1\}$ to query $x \in \mathcal{X}$ is an independent realization of the random variable Y satisfying $\mathbb{P}(Y = h^*(x)) > \mathbb{P}(Y = -h^*(x))$, where $h^* \in \mathcal{H}$ is fixed but unknown (i.e., the response is only probably correct)

The *noise bound* is defined as $\alpha := \sup_{x \in \mathcal{X}} \mathbb{P}(Y \neq h^*(x))$

Theory of Generalized Binary Search

GBS with N hypotheses/classifiers

Theory of Generalized Binary Search

GBS with N hypotheses/classifiers

Noiseless Search

Theorem 1 If the neighborly condition holds, then GBS terminates with the correct hypothesis after at most $c \log N$ queries, where $c > 0$ is a small constant.

Theory of Generalized Binary Search

GBS with N hypotheses/classifiers

Noiseless Search

Theorem 1 If the neighborly condition holds, then GBS terminates with the correct hypothesis after at most $c \log N$ queries, where $c > 0$ is a small constant.

Noisy Search

Theorem 2 Let $\mathbb{P}$ denotes the underlying probability measure (governing noises and algorithm randomization). If $\beta > \alpha$ and the neighborly condition holds, then the noisy GBS algorithm generates a sequence of hypotheses satisfying

$$\mathbb{P}(\widehat{h}_n \neq h^*) \;\leq\; N\,(1-\lambda)^n \;\leq\; N\,e^{-c\,n} \quad , \; n = 0, 1, \ldots$$

with exponential constant $c > 0$.

Theory of Generalized Binary Search

GBS with N hypotheses/classifiers

Noiseless Search

Theorem 1 If the neighborly condition holds, then GBS terminates with the correct hypothesis after at most $c \log N$ queries, where $c > 0$ is a small constant.

Noisy Search

Theorem 2 Let $\mathbb{P}$ denotes the underlying probability measure (governing noises and algorithm randomization). If $\beta > \alpha$ and the neighborly condition holds, then the noisy GBS algorithm generates a sequence of hypotheses satisfying

$$\mathbb{P}(\widehat{h}_n \neq h^*) \;\leq\; N\,(1-\lambda)^n \;\leq\; N\,e^{-c\,n} \quad,\; n = 0, 1, \ldots$$

with exponential constant $c > 0$.

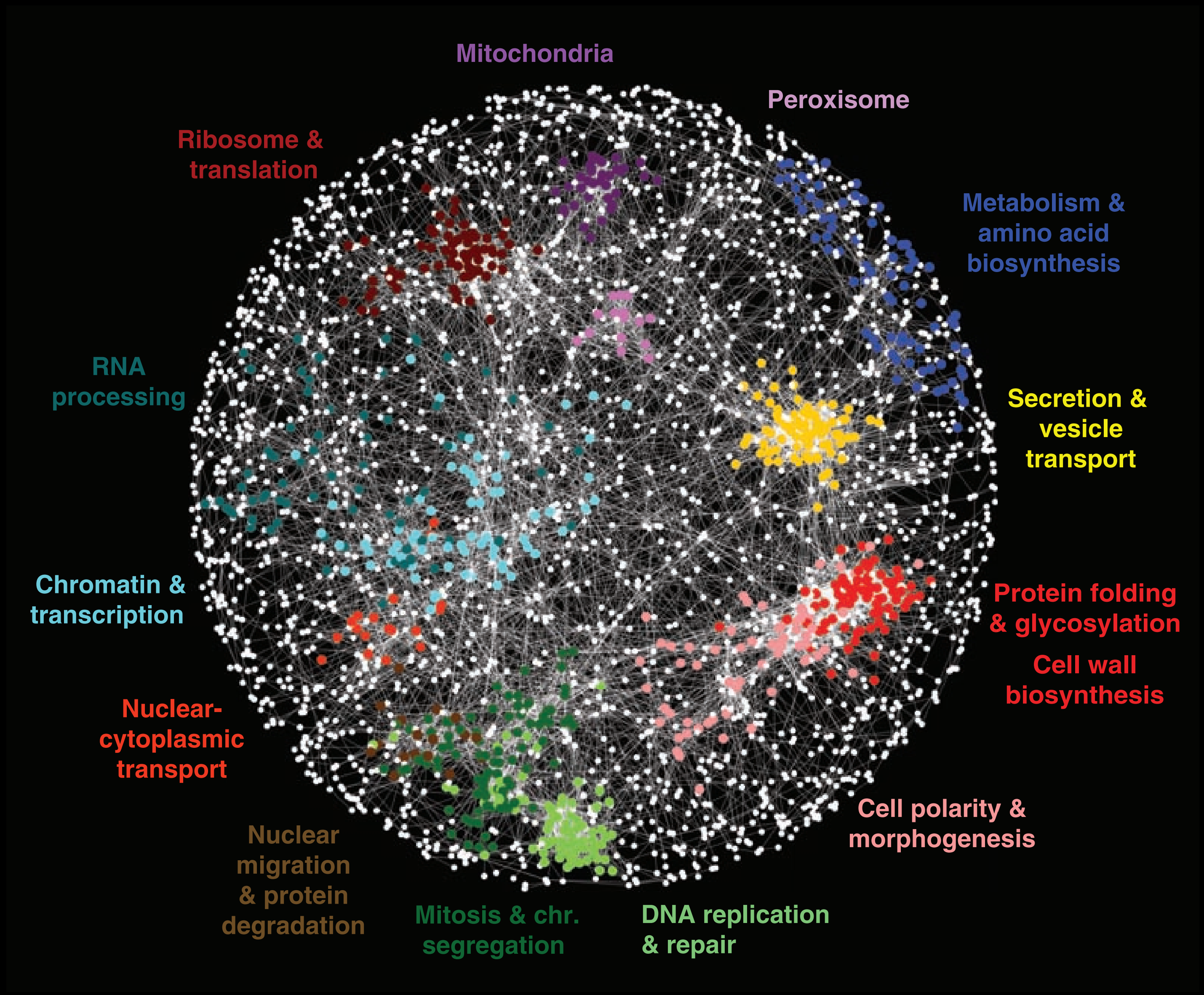
If we desire $\mathbb{P}(\widehat{h}_n \neq h^*) < \delta$, then we require only $n = \frac{1}{\lambda} \log \frac{N}{\delta}$ queries.

Active Clustering

Clustering in Large-Scale Networked Systems

Difficult or impossible to measure/observe everything in large systems

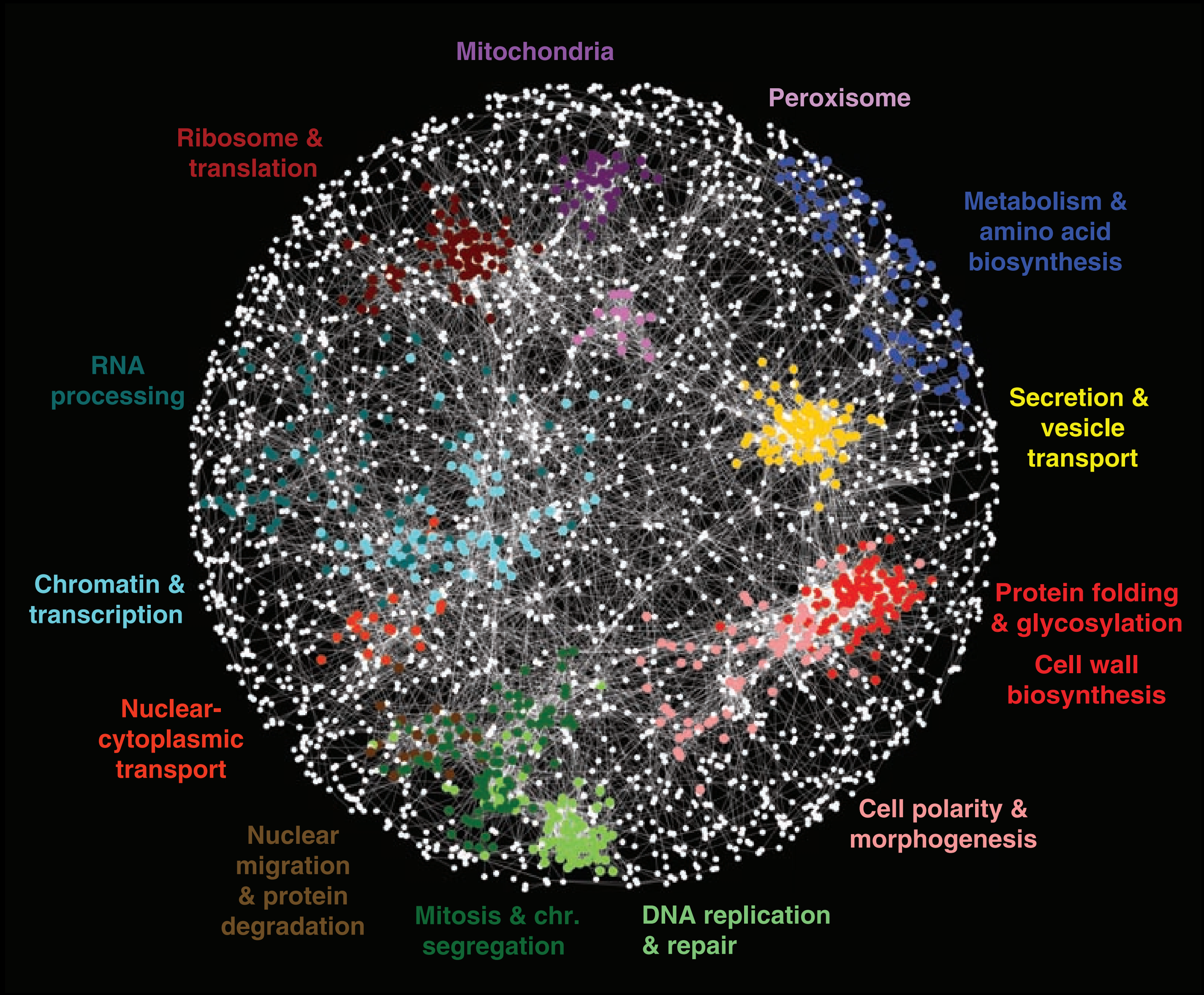
Clustering in Large-Scale Networked Systems



Genetic Landscape of a Cell
Boone Lab - Toronto

Difficult or impossible to measure/observe everything in large systems

Clustering in Large-Scale Networked Systems



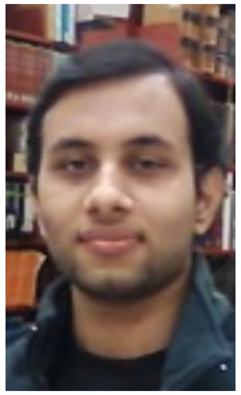
Genetic Landscape of a Cell
Boone Lab - Toronto

State of the Internet
Akamai

Difficult or impossible to measure/observe everything in large systems

Network Structure and Clustering

Complex systems are not defined by the independent functions of individual components, rather they depend on the orchestrated interactions of these elements.



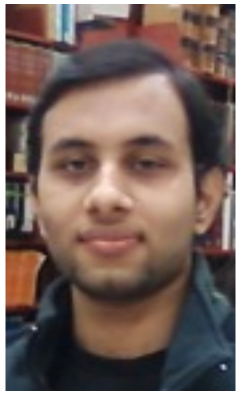
Gautam Dasarathy

Brian Eriksson

Network Structure and Clustering

Complex systems are not defined by the independent functions of individual components, rather they depend on the orchestrated interactions of these elements.

Network(s) of interactions can be revealed via **clustering** based on measured features



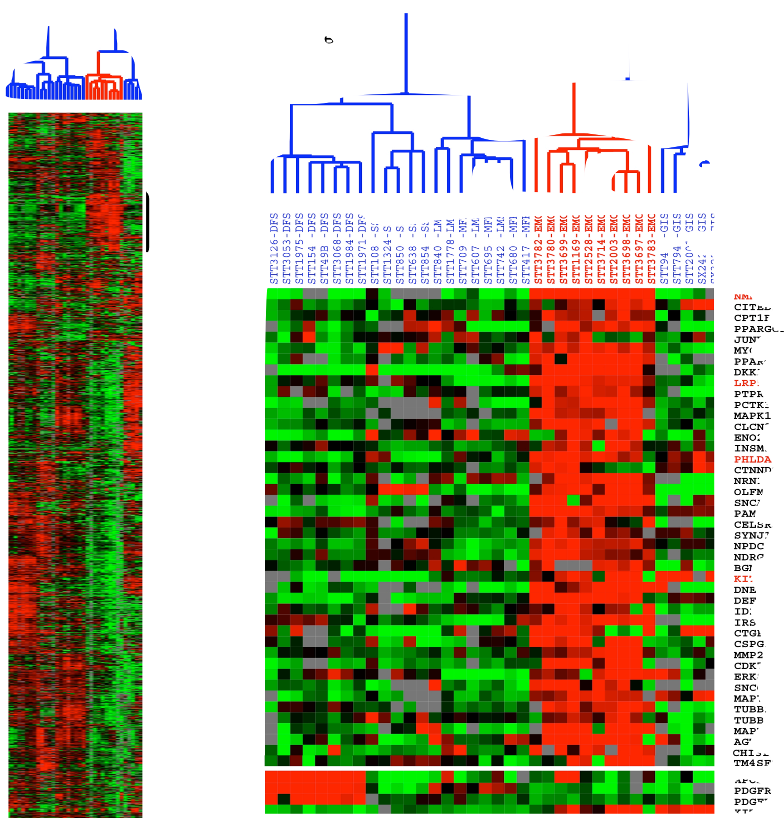
Gautam Dasarathy

Brian Eriksson

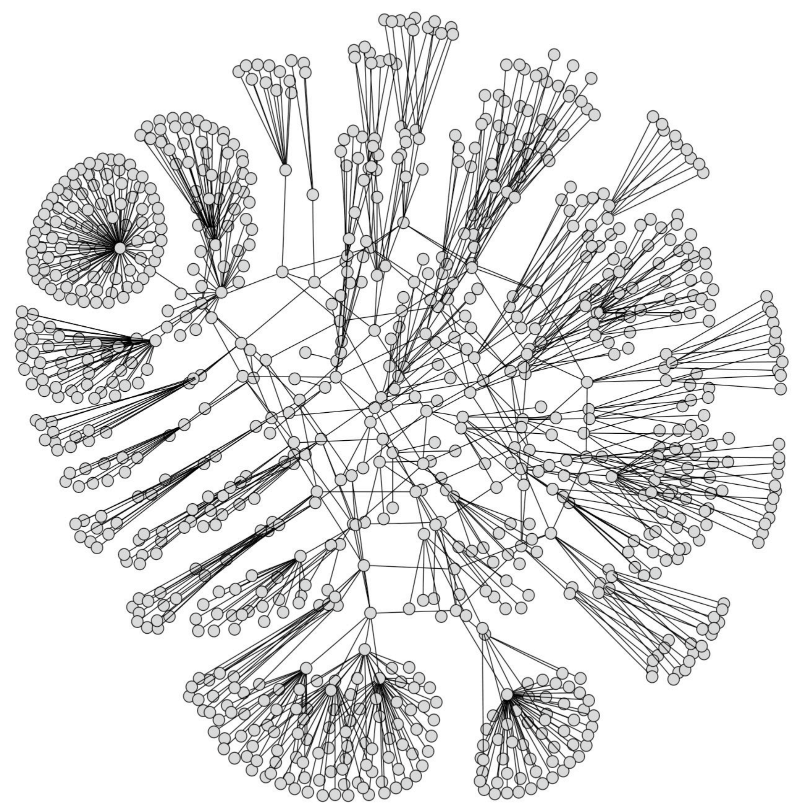
Network Structure and Clustering

Complex systems are not defined by the independent functions of individual components, rather they depend on the orchestrated interactions of these elements.

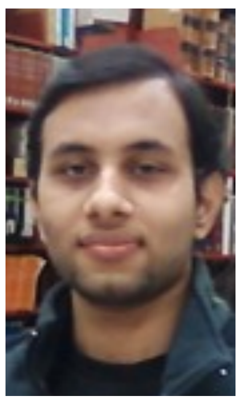
Network(s) of interactions can be revealed via **clustering** based on measured features



genes and expression/ interaction profiles



network routers and traffic/distance profiles



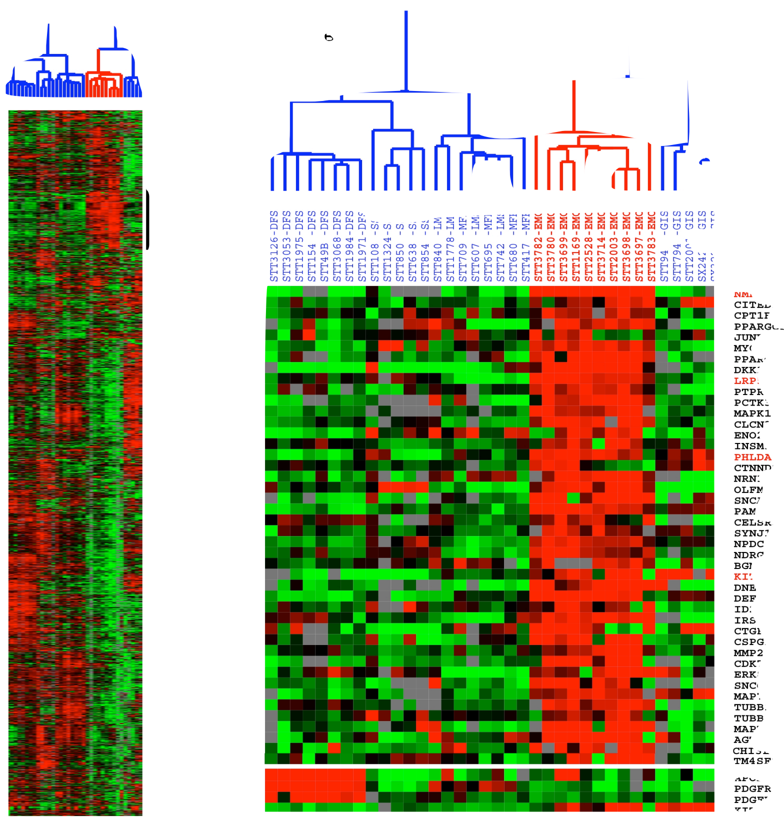
Gautam Dasarathy

Brian Eriksson

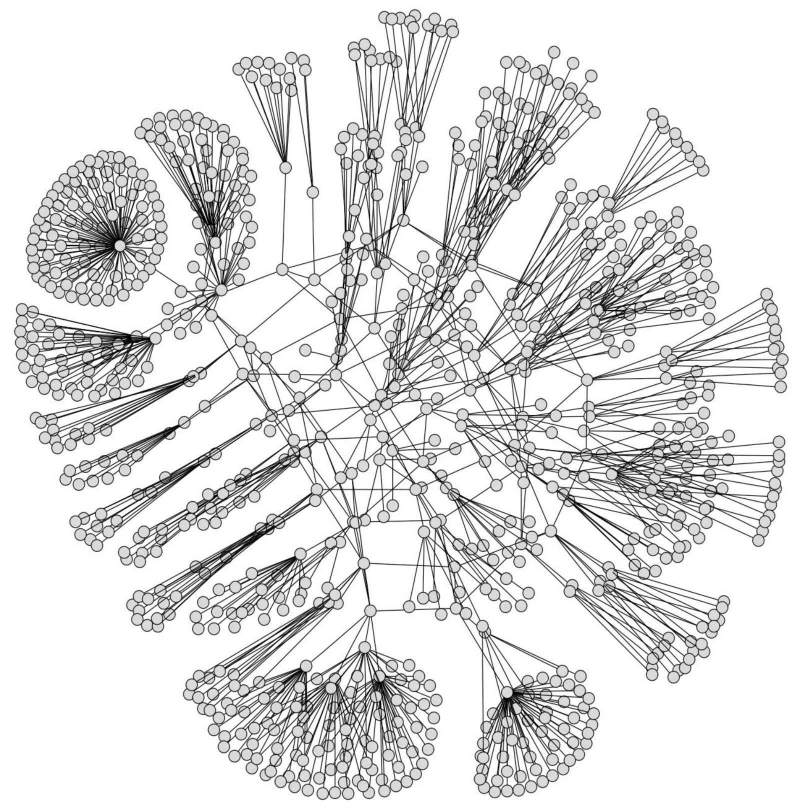
Network Structure and Clustering

Complex systems are not defined by the independent functions of individual components, rather they depend on the orchestrated interactions of these elements.

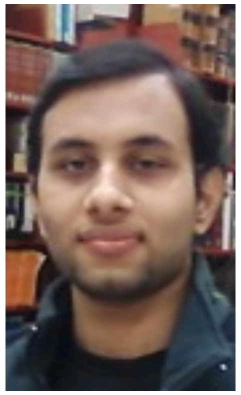
Network(s) of interactions can be revealed via **clustering** based on measured features



genes and expression/
interaction profiles



network routers and
traffic/distance profiles



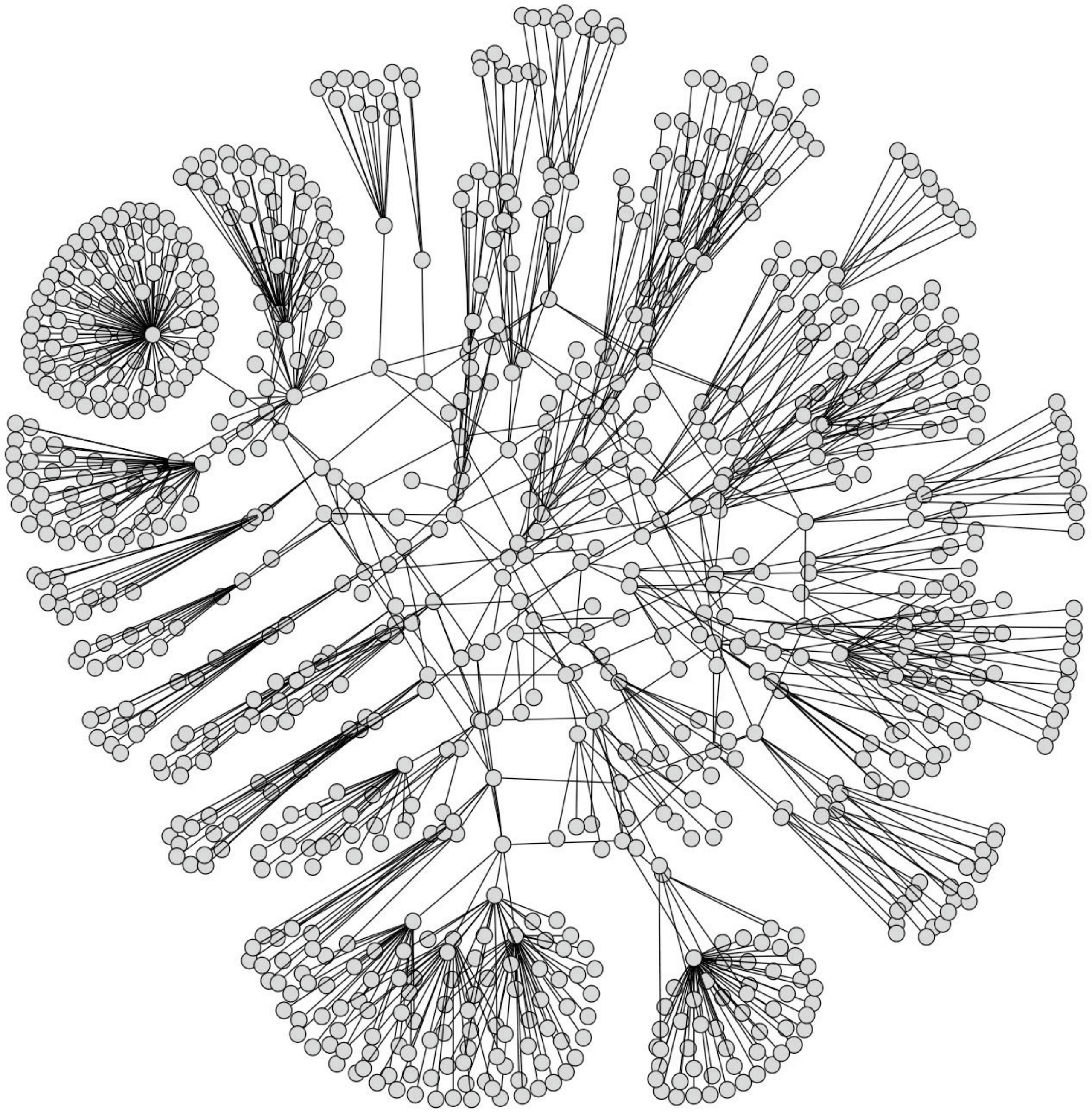
Gautam
Dasarathy

Brian
Eriksson

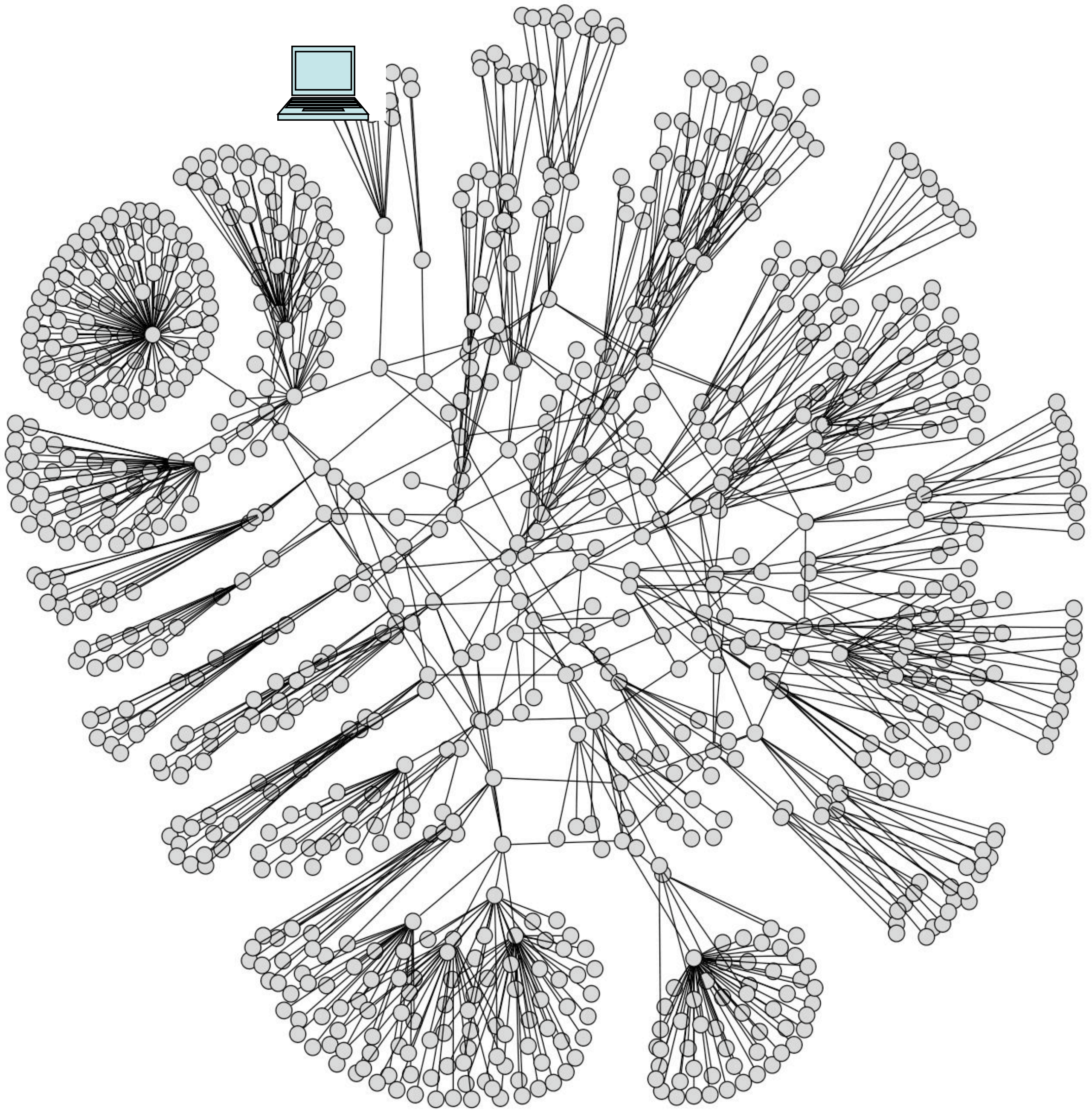
Similarity-Based Clustering: Each component (gene/router) has an associated feature (measurement profile). Components can be clustered based on feature similarities.

Internet Topology Inference

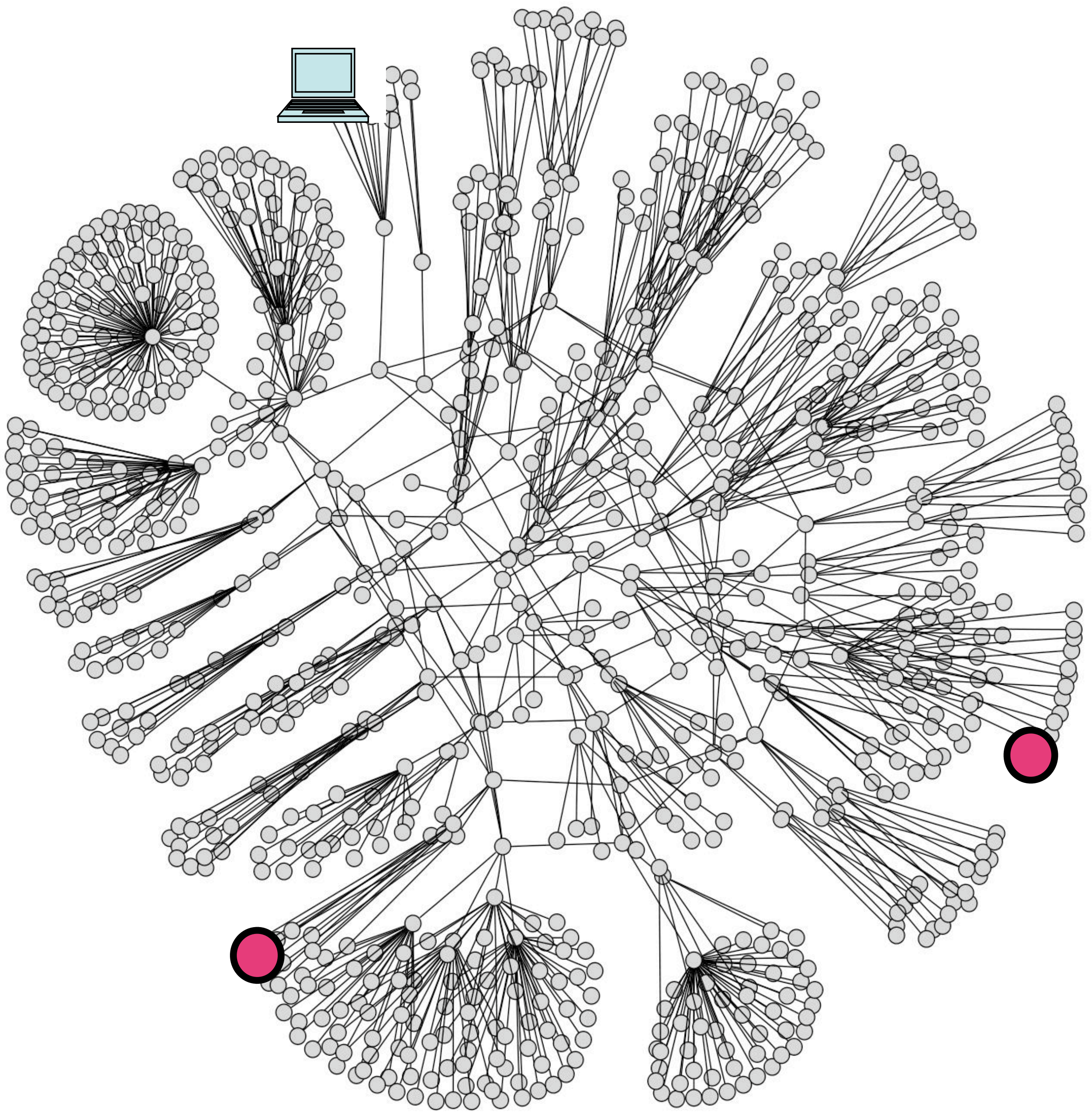
Internet Topology Inference



Internet Topology Inference

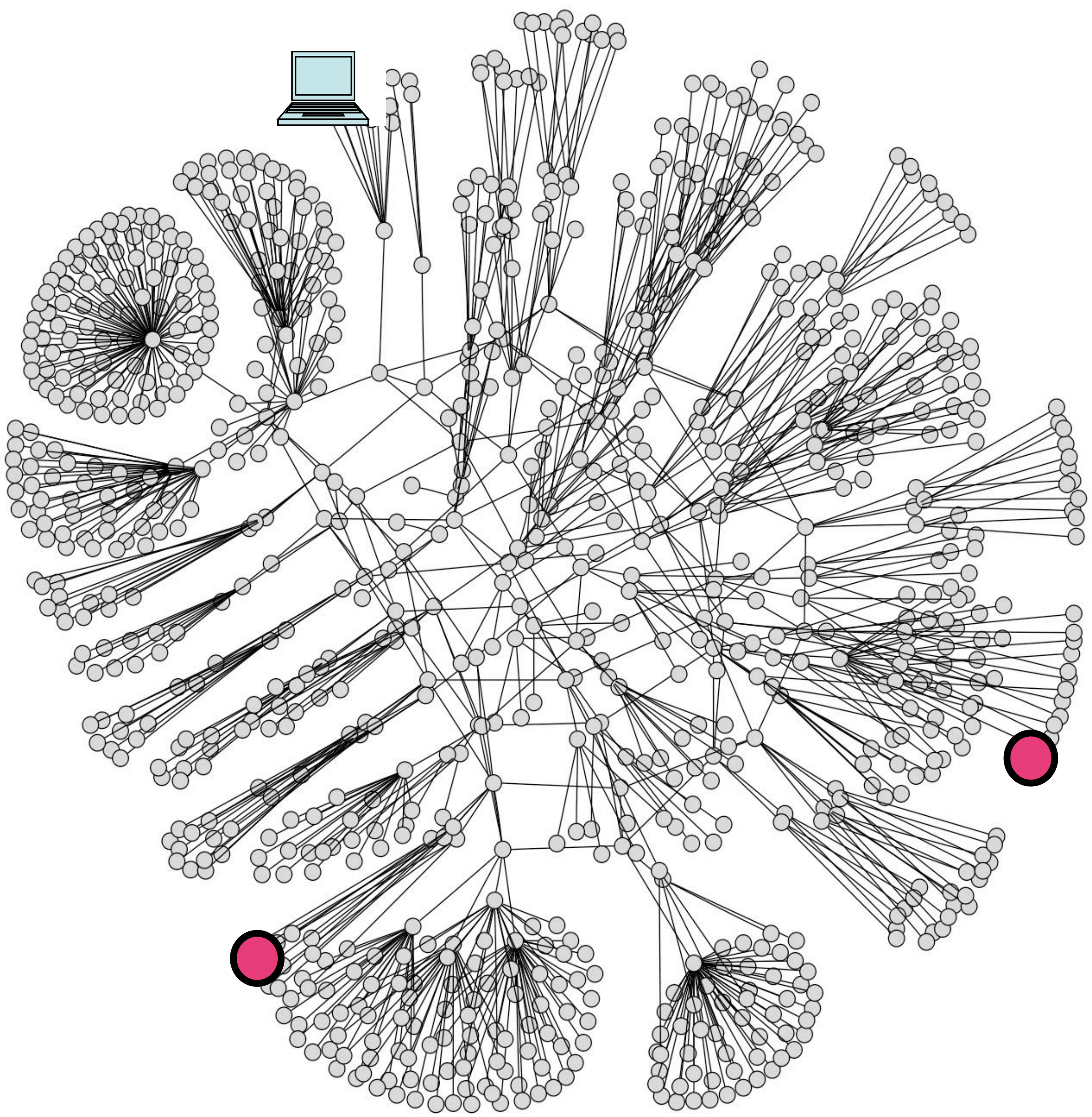


Internet Topology Inference

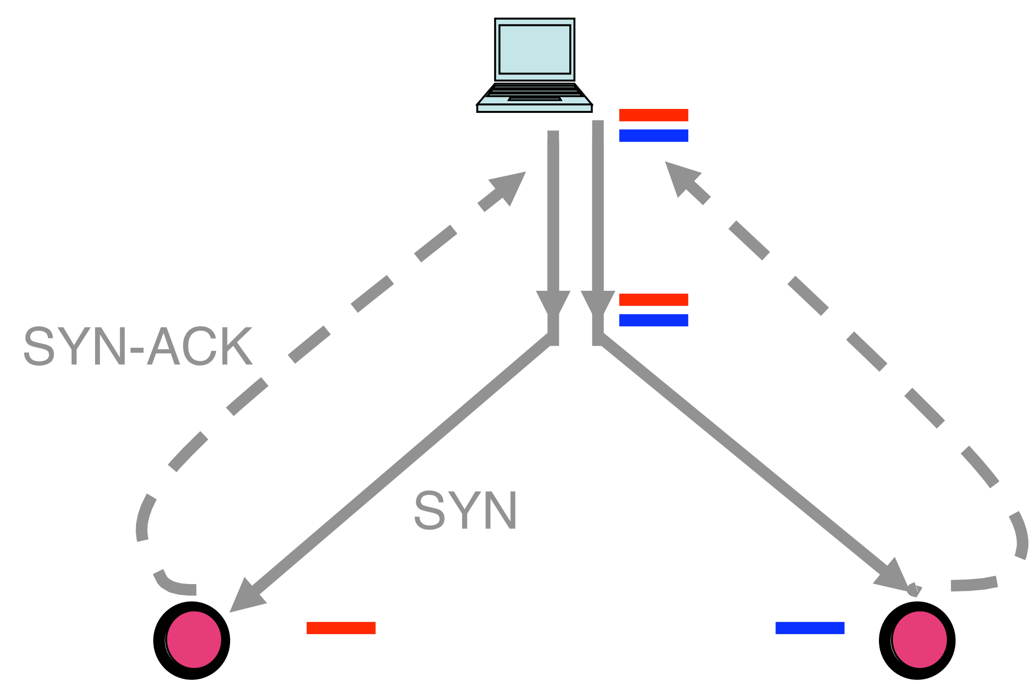


Correlation between traffic patterns at two points can indicate the **similarity** between nodes (e.g., number of shared links in paths)

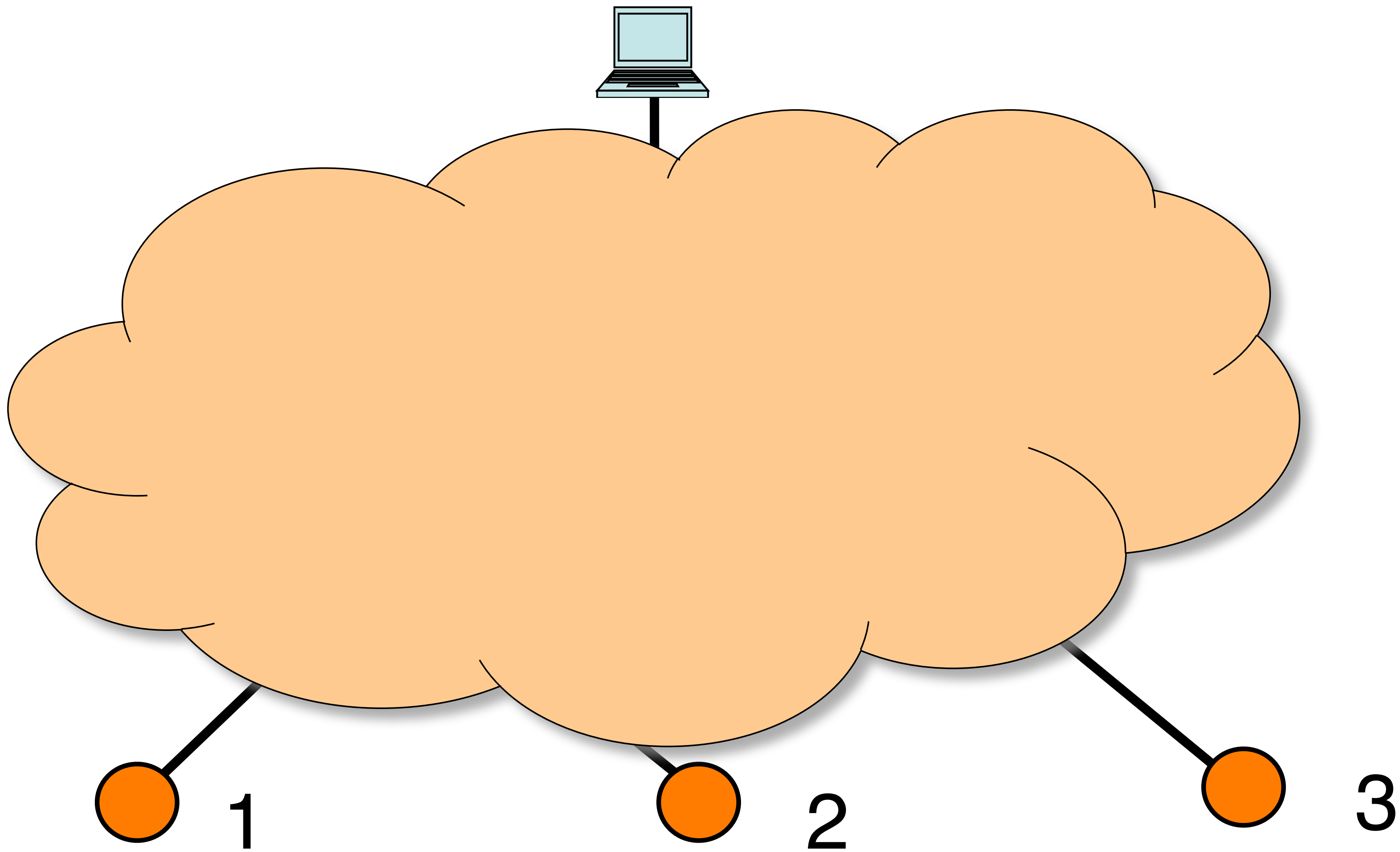
Internet Topology Inference



Correlation between traffic patterns at two points can indicate the **similarity** between nodes (e.g., number of shared links in paths)



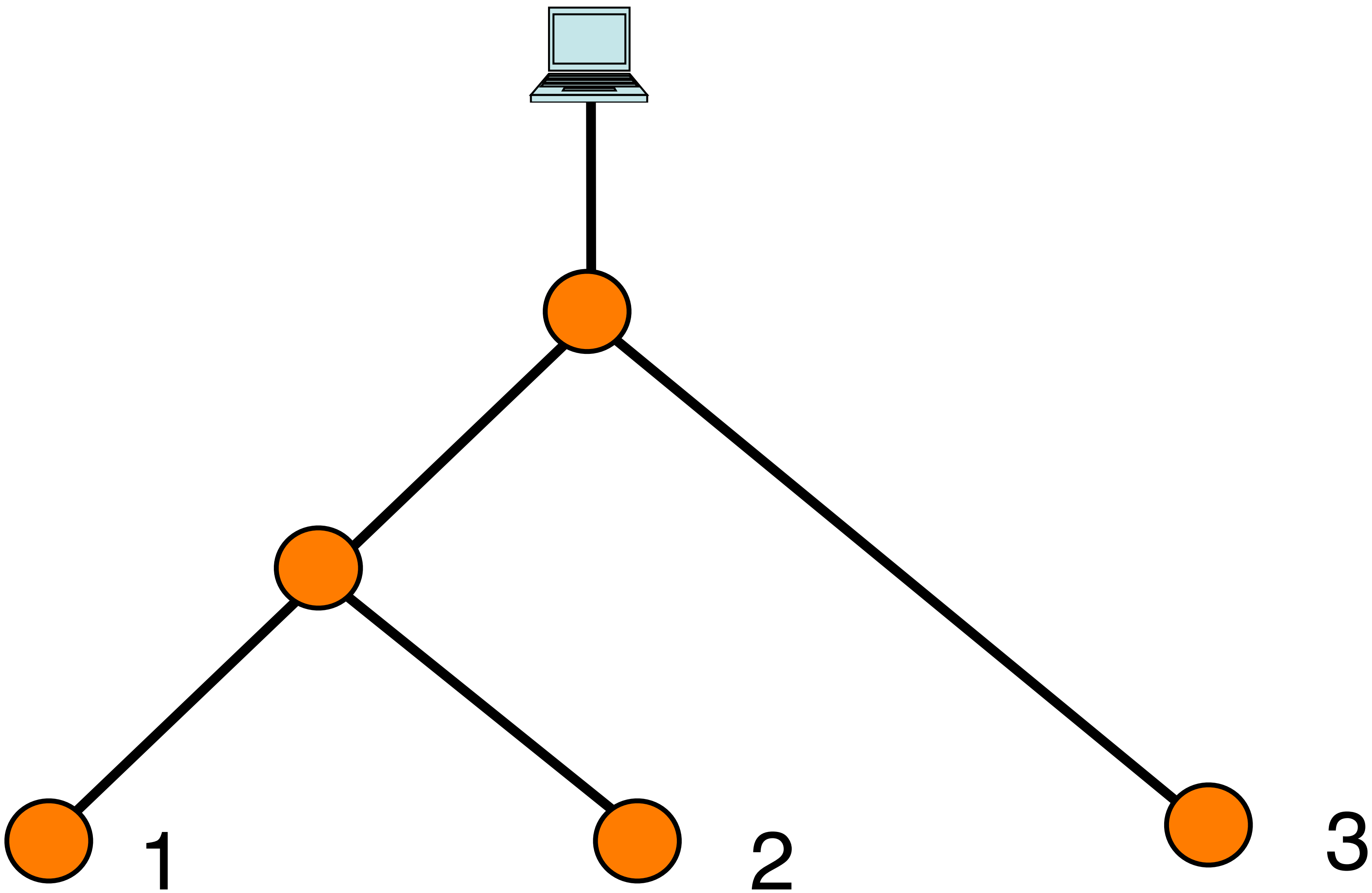
Network Mapping



$$s_{1,2} > s_{1,3},\ s_{2,3}$$

RTT_1 & RTT_2 more correlated than RTT_1 & RTT_3 or RTT_2 & RTT_3

Network Mapping



$$s_{1,2} > s_{1,3},\ s_{2,3}$$

RTT_1 & RTT_2 more correlated than RTT_1 & RTT_3 or RTT_2 & RTT_3

Active Clustering

Active Clustering

Questions :

1. Can we cluster from a subsample similarities?

2. Does random subsampling suffice?

Active Clustering

Questions :
1. Can we cluster from a subsample similarities?

2. Does random subsampling suffice?

Redundancy

Active Clustering

Questions :

1. Can we cluster from a subsample similarities?

2. Does random subsampling suffice?

Redundancy

Active Clustering

Questions :

1. Can we cluster from a subsample similarities?

2. Does random subsampling suffice?

Redundancy

Active Clustering

Questions :

1. Can we cluster from a subsample similarities?

A : Maybe unnecessary to obtain all pairwise similarities

2. Does random subsampling suffice?

Redundancy

Active Clustering

Questions :

1. Can we cluster from a subsample similarities?

A : Maybe unnecessary to obtain all pairwise similarities

2. Does random subsampling suffice?

Redundancy

Passive (Random) Subsampling

Random subsampling will miss small clusters

Actually, we can show that at least $O(n^2/m)$ pairwise similarities are required to recover clusters of size m.

Active Clustering

Questions :

1. Can we cluster from a subsample similarities?

A : Maybe unnecessary to obtain all pairwise similarities

2. Does random subsampling suffice?

A: No ! We will require $O(n^2)$ random similarities

Redundancy

Passive (Random) Subsampling

Random subsampling will miss small clusters

Actually, we can show that at least $O(n^2/m)$ pairwise similarities are required to recover clusters of size m.

Active Clustering: Efficient Hierarchical Clustering

Active Clustering: Efficient Hierarchical Clustering

The proposed method **adaptively** selects the most informative pairwise similarities to recover the hierarchical clustering.

Under mild assumptions, we can discern the “outlier” of three items using only 3 pairwise similarities. i.e.,

Active Clustering: Efficient Hierarchical Clustering

The proposed method **adaptively** selects the most informative pairwise similarities to recover the hierarchical clustering.

Under mild assumptions, we can discern the “outlier” of three items using only 3 pairwise similarities. i.e.,

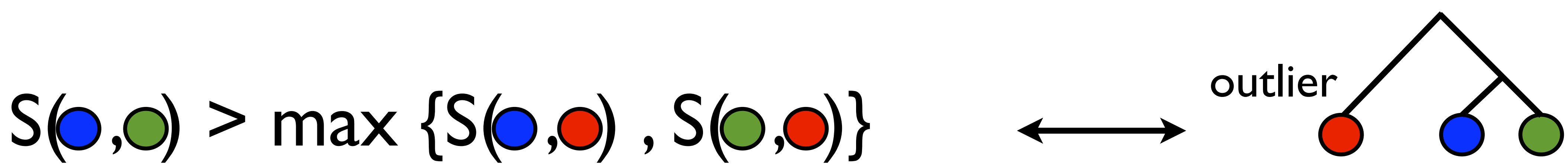
intra-cluster similarities > inter-cluster similarities

Active Clustering: Efficient Hierarchical Clustering

The proposed method **adaptively** selects the most informative pairwise similarities to recover the hierarchical clustering.

Under mild assumptions, we can discern the “outlier” of three items using only 3 pairwise similarities. i.e.,

intra-cluster similarities > inter-cluster similarities



Active Clustering: Efficient Hierarchical Clustering

Active Clustering: Efficient Hierarchical Clustering

This is a sequential procedure ...

Active Clustering: Efficient Hierarchical Clustering

This is a sequential procedure ...

Inserting a new object into a tree with *i* leaves

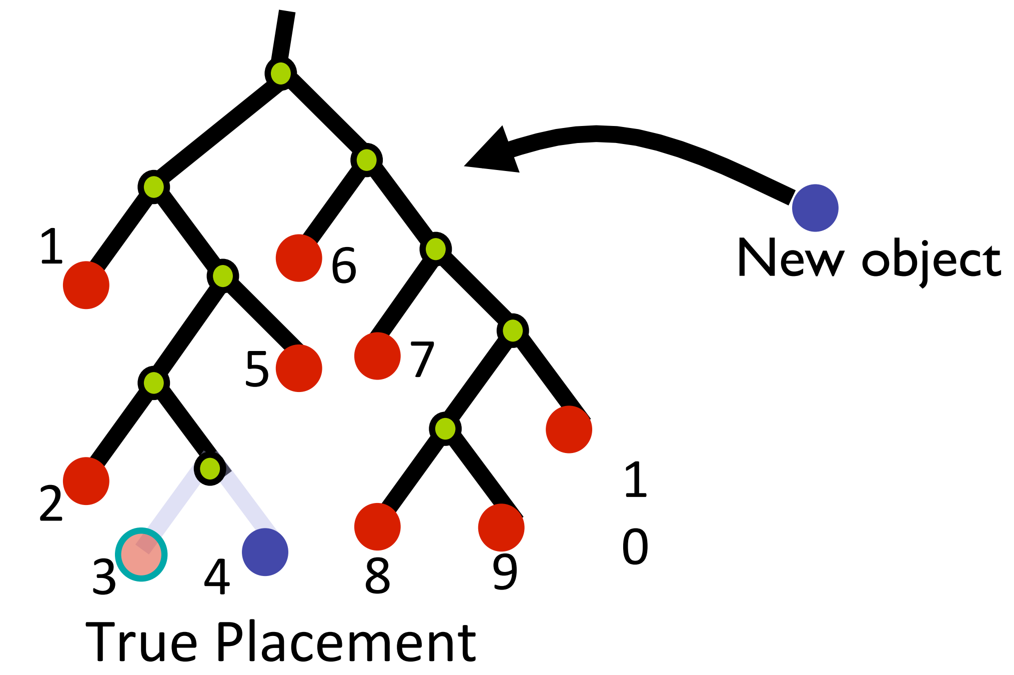
- Pick an internal node v with ≈ $i/2$ objects as descendants
- Find two leaves x_k and x_j whose common ancestor is v
- Find outlier(x_k, x_j, v) and discard a portion of the tree
- Proceed till there are only two leaves left and insert using a final outlier test.

Active Clustering: Efficient Hierarchical Clustering

This is a sequential procedure ...

Inserting a new object into a tree with *i* leaves

- Pick an internal node v with ≈ $i/2$ objects as descendants
- Find two leaves x_k and x_j whose common ancestor is v
- Find outlier(x_k, x_j, v) and discard a portion of the tree
- Proceed till there are only two leaves left and insert using a final outlier test.

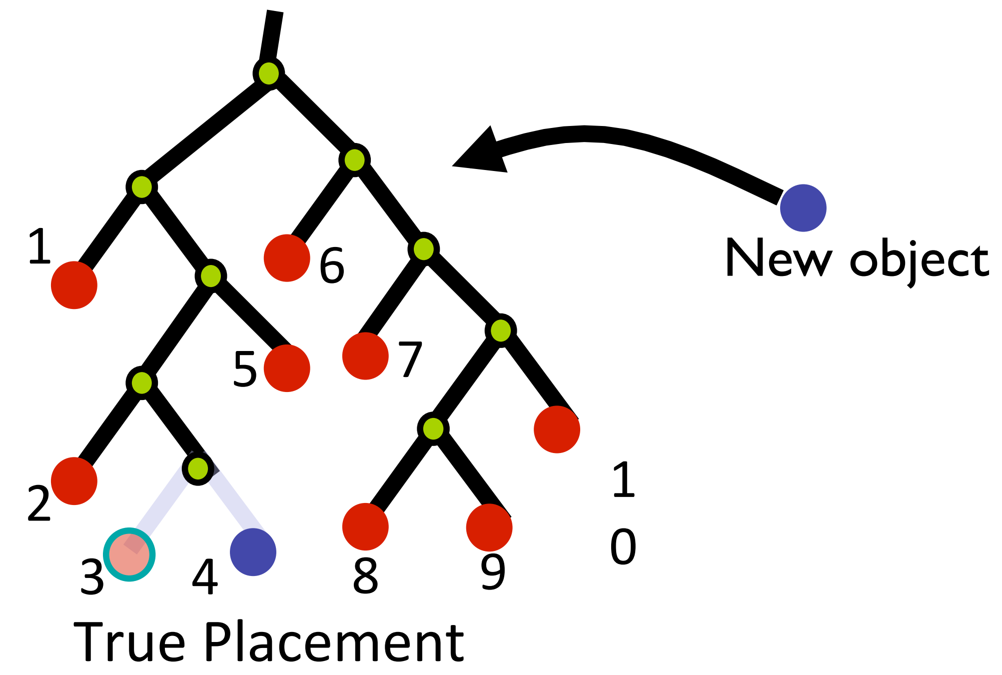


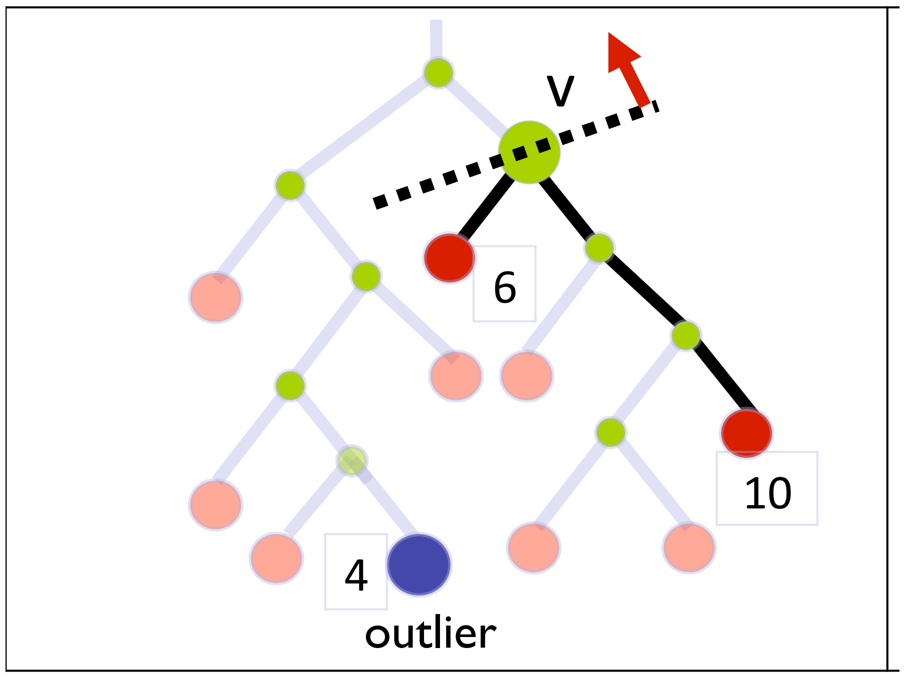
Active Clustering: Efficient Hierarchical Clustering

This is a sequential procedure ...

Inserting a new object into a tree with *i* leaves

- Pick an internal node v with ≈ $i/2$ objects as descendants
- Find two leaves x_k and x_j whose common ancestor is v
- Find outlier(x_k, x_j, v) and discard a portion of the tree
- Proceed till there are only two leaves left and insert using a final outlier test.





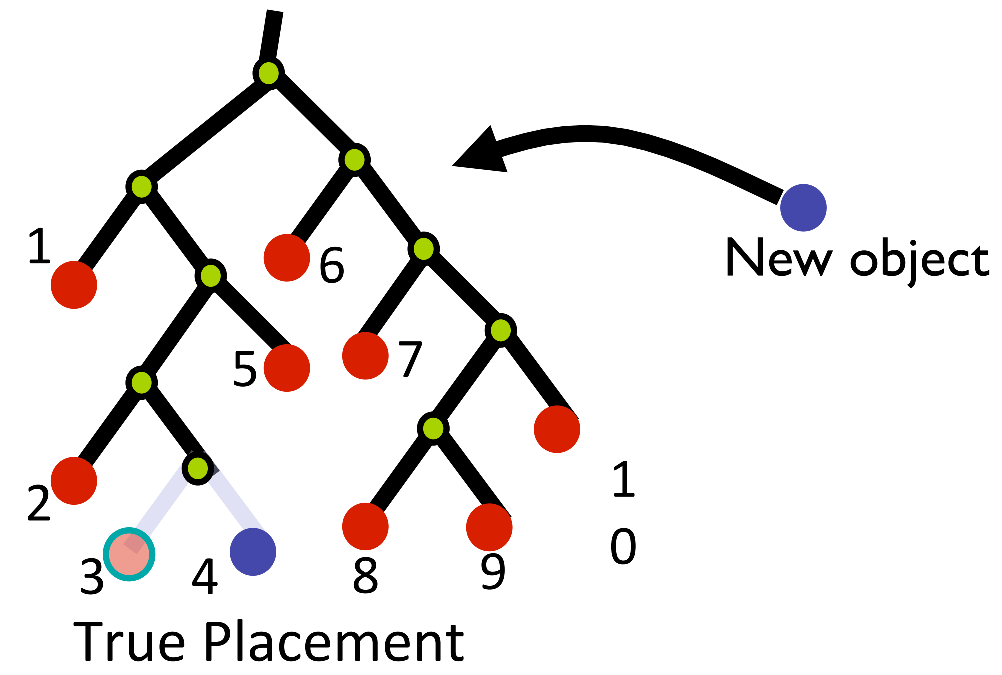
Step 1

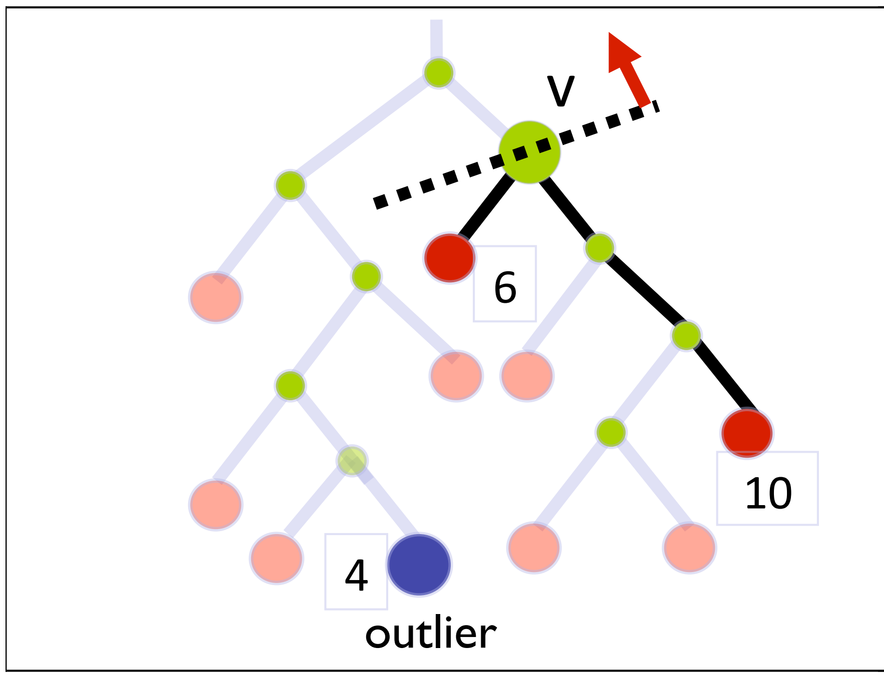
Active Clustering: Efficient Hierarchical Clustering

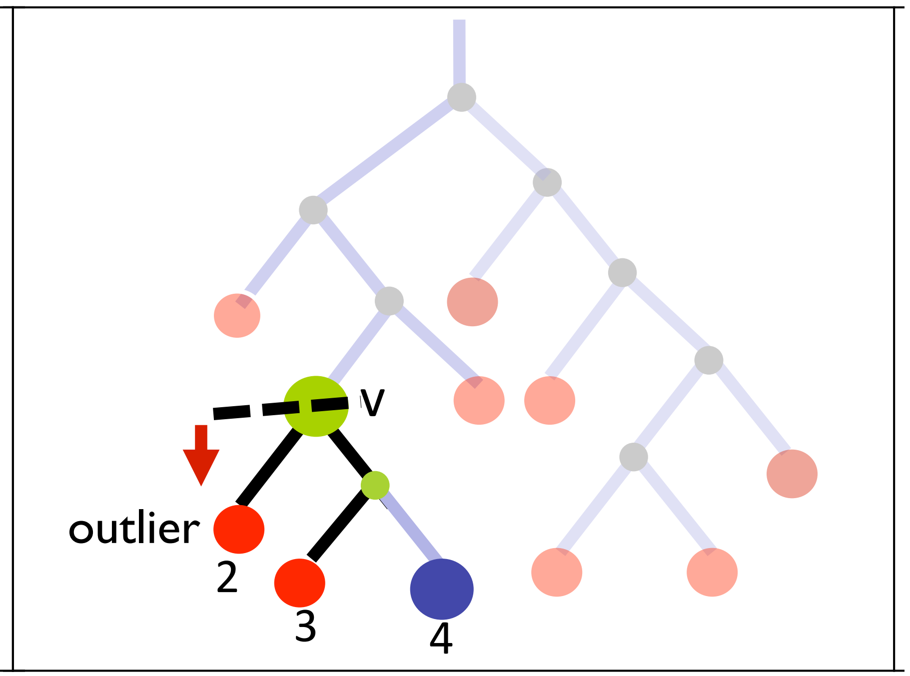
This is a sequential procedure ...

Inserting a new object into a tree with *i* leaves

- Pick an internal node v with ≈ $i/2$ objects as descendants
- Find two leaves x_k and x_j whose common ancestor is v
- Find outlier(x_k, x_j, v) and discard a portion of the tree
- Proceed till there are only two leaves left and insert using a final outlier test.







Step 1

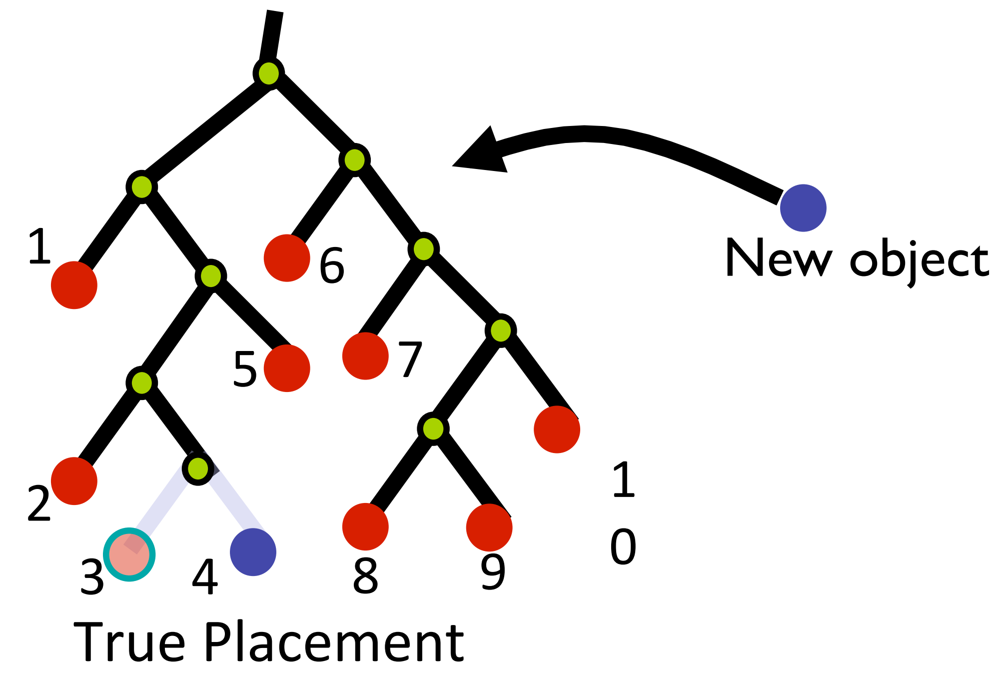
Step 2

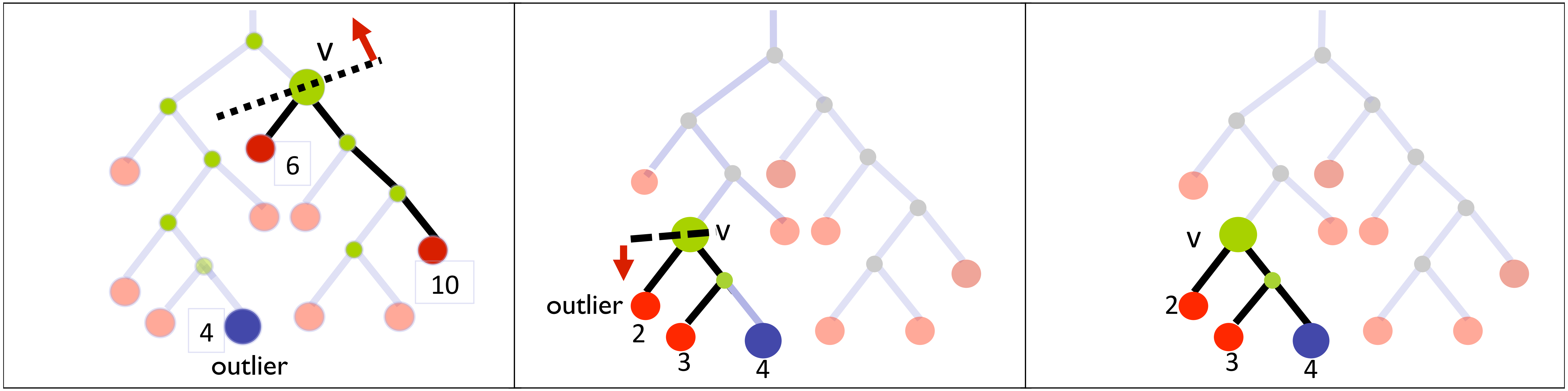
Active Clustering: Efficient Hierarchical Clustering

This is a sequential procedure ...

Inserting a new object into a tree with *i* leaves

- Pick an internal node v with $\approx i/2$ objects as descendants
- Find two leaves x_k and x_j whose common ancestor is v
- Find outlier(x_k, x_j, v) and discard a portion of the tree
- Proceed till there are only two leaves left and insert using a final outlier test.





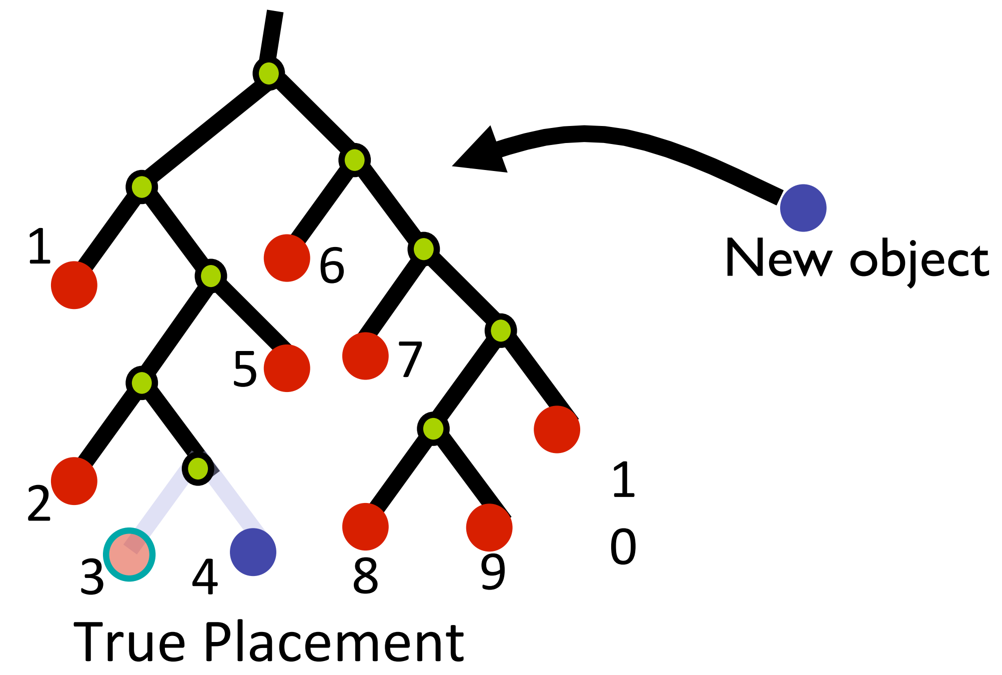
Step 1 Step 2 Step 3

Active Clustering: Efficient Hierarchical Clustering

This is a sequential procedure ...

Inserting a new object into a tree with *i* leaves

- Pick an internal node v with ≈ $i/2$ objects as descendants
- Find two leaves x_k and x_j whose common ancestor is v
- Find outlier(x_k, x_j, v) and discard a portion of the tree
- Proceed till there are only two leaves left and insert using a final outlier test.

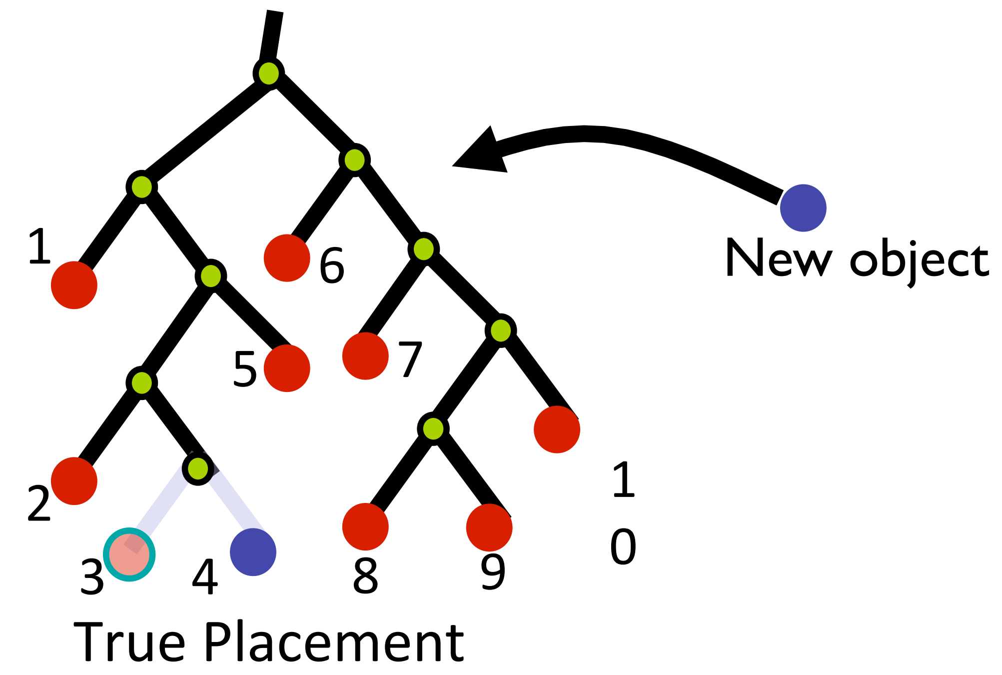


Active Clustering: Efficient Hierarchical Clustering

This is a sequential procedure ...

Inserting a new object into a tree with *i* leaves

- Pick an internal node v with $\approx i/2$ objects as descendants
- Find two leaves x_k and x_j whose common ancestor is v
- Find outlier(x_k, x_j, v) and discard a portion of the tree
- Proceed till there are only two leaves left and insert using a final outlier test.



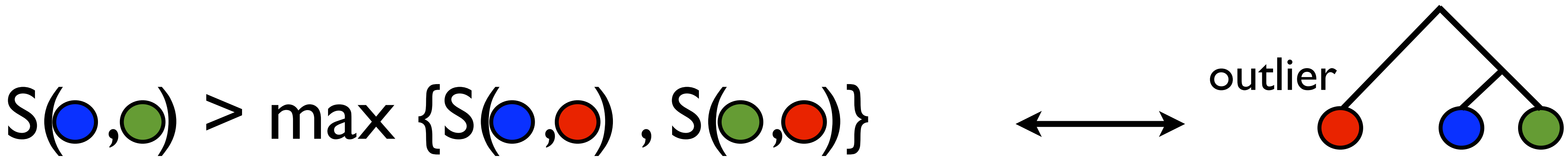
Theorem:
Under certain assumptions, the hierarchical clustering of n objects can be recovered using no more than $\underbrace{3n \log n}$ sequentially and adaptively selected pairwise similarities.

within a constant factor of the *information theoretic lower bound*

Robust Active Clustering

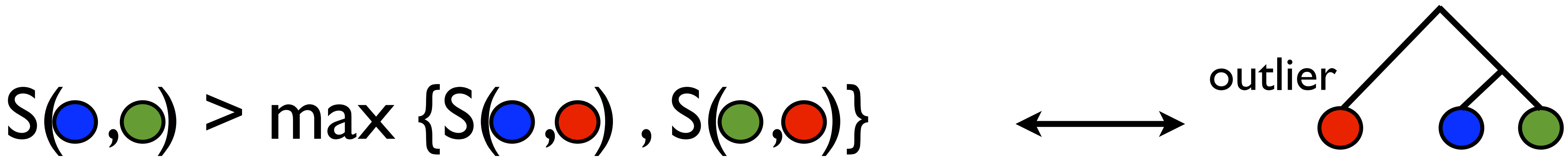
Robust Active Clustering

The previous technique is very sensitive to noise/errors and violations of the assumptions.



Robust Active Clustering

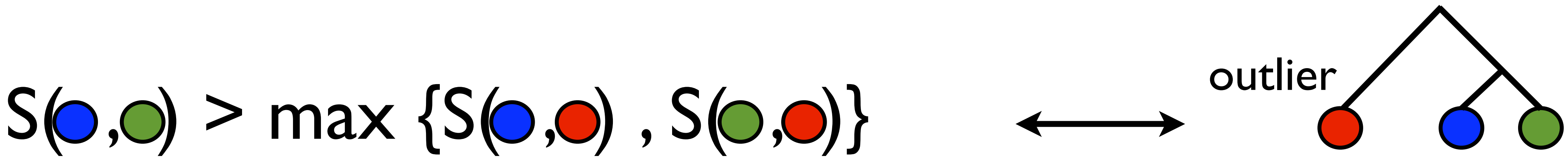
The previous technique is very sensitive to noise/errors and violations of the assumptions.



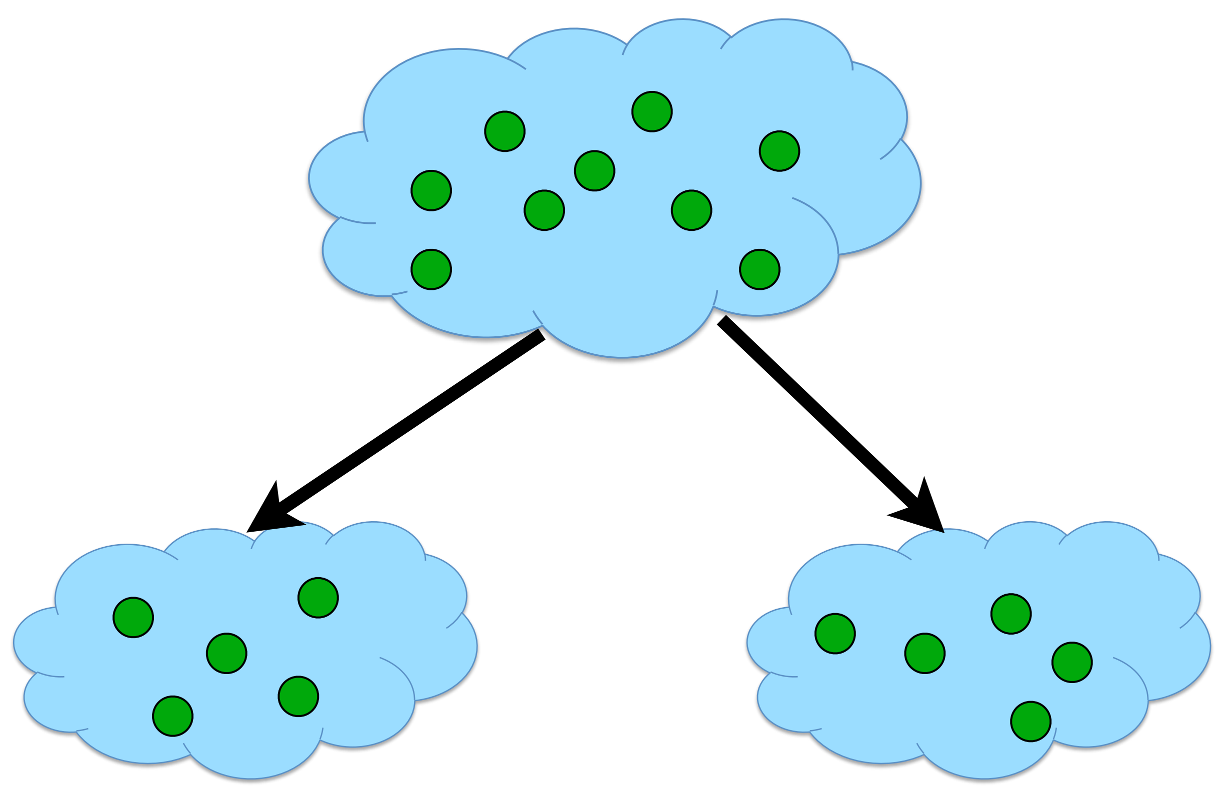
To overcome this, we design a **top-down recursive splitting approach** and use **voting** to boost our confidence about each decision we make.

Robust Active Clustering

The previous technique is very sensitive to noise/errors and violations of the assumptions.

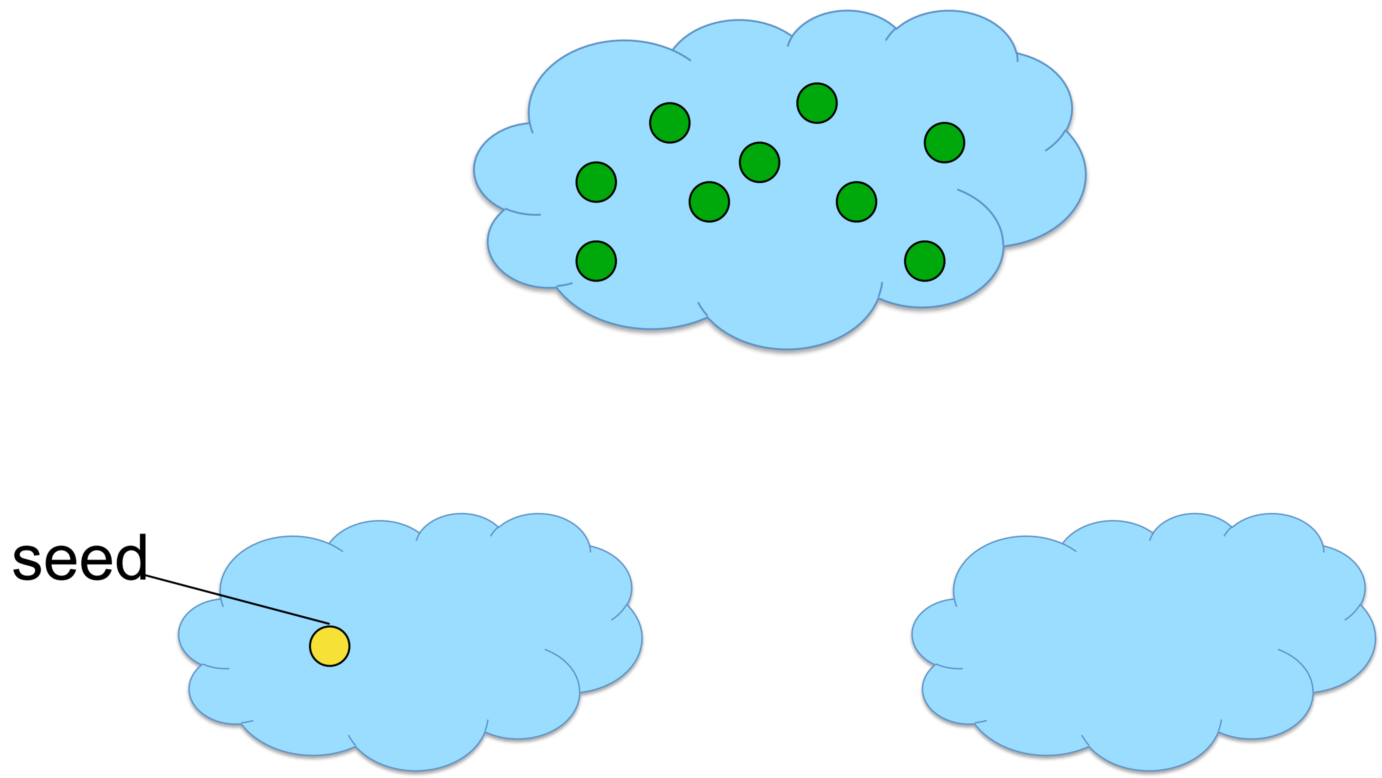


To overcome this, we design a **top-down recursive splitting approach** and use **voting** to boost our confidence about each decision we make.



Goal : In each step, split a single cluster into 2 sub-clusters efficiently

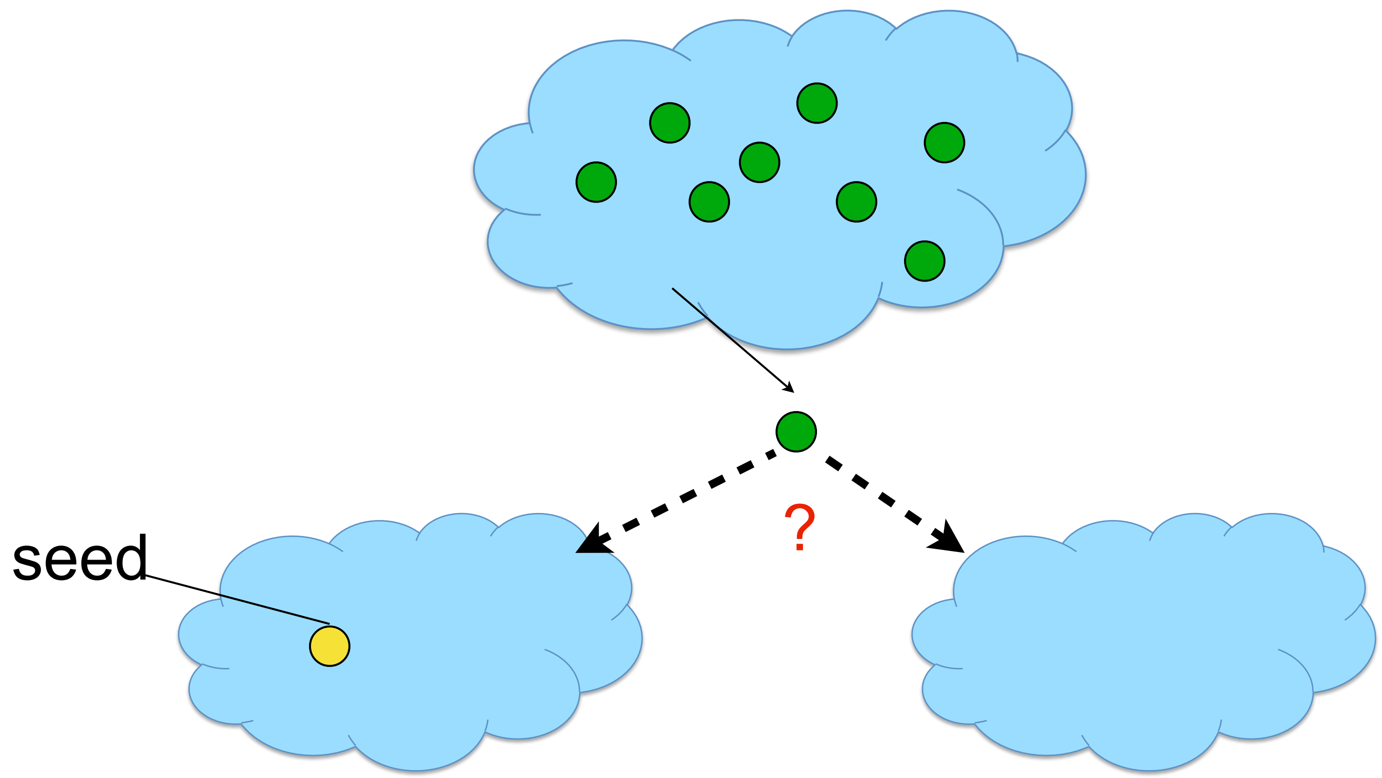
Robust Active Clustering Procedure



<u>Strategy:</u> Sequentially decide which of the two sub-clusters each ● goes into.

1. Pick a random object and call it the “seed”

Robust Active Clustering Procedure

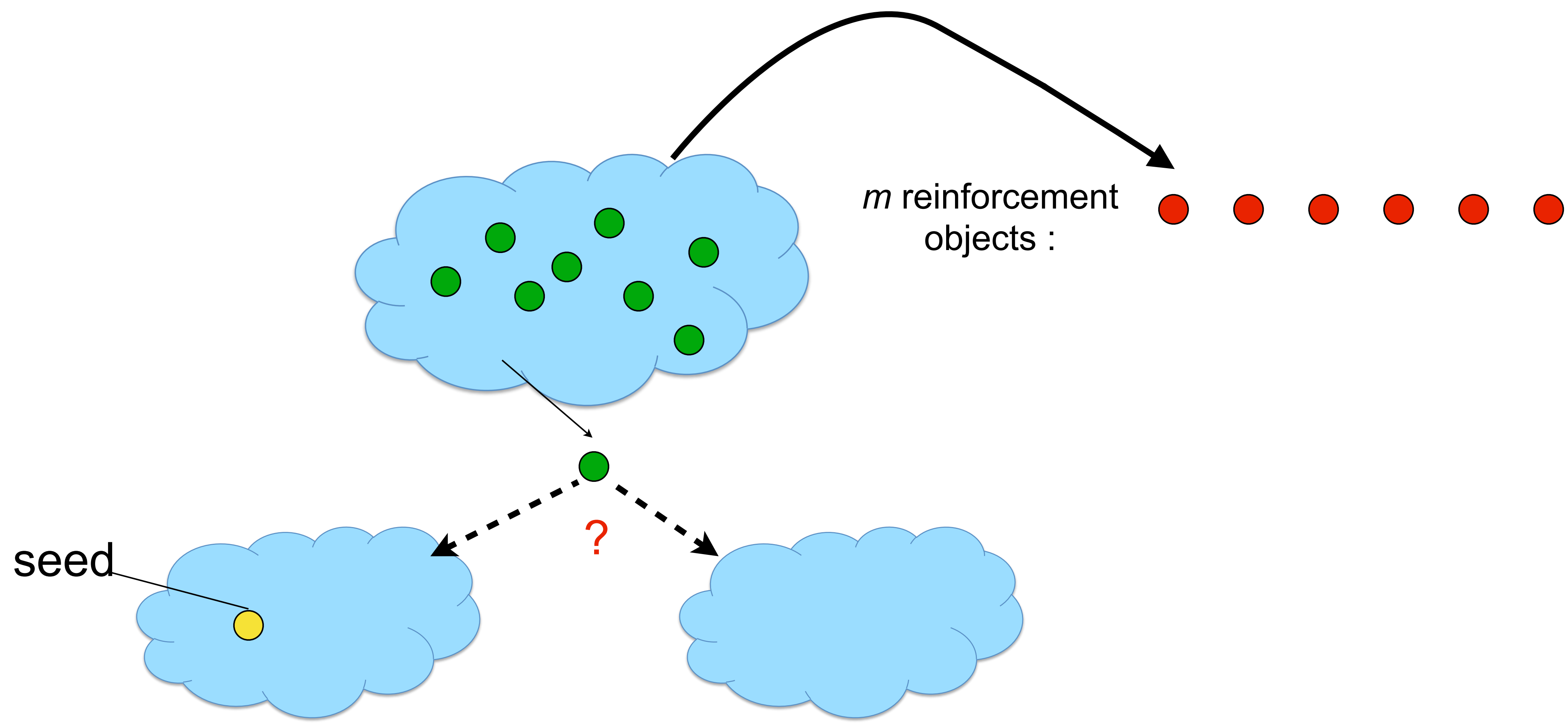


<u>Strategy:</u> Sequentially decide which of the two sub-clusters each ● goes into.

1. Pick a random object and call it the “seed”

2. For the other objects, decide if they are similar to ● or not.

Robust Active Clustering Procedure



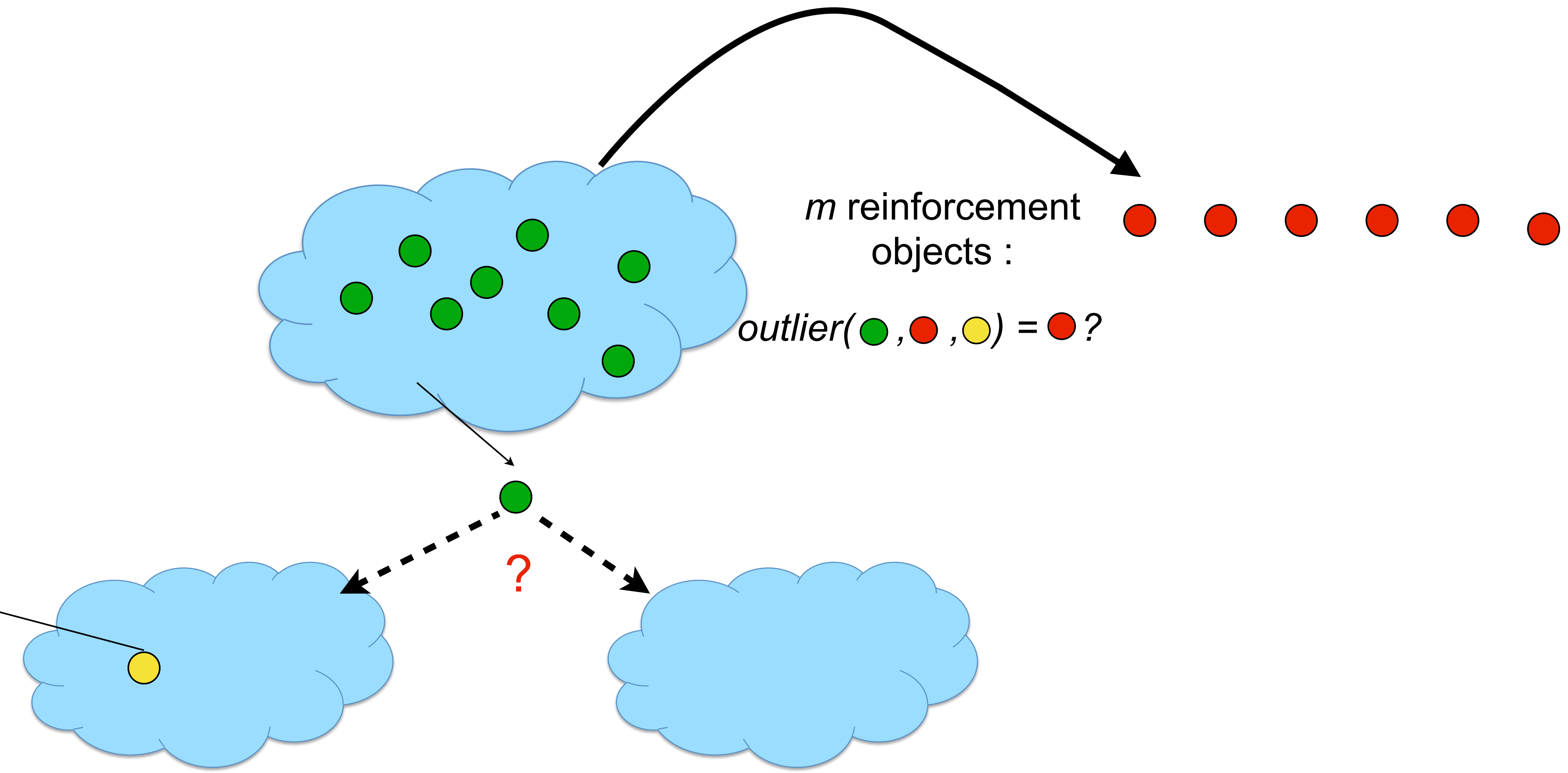
Strategy: Sequentially decide which of the two sub-clusters each ● goes into.

1. Pick a random object and call it the “seed”.

2. For the other objects, decide if they are similar to ● or not.

3. Towards this, randomly pick m “reinforcement” objects from C.

Robust Active Clustering Procedure



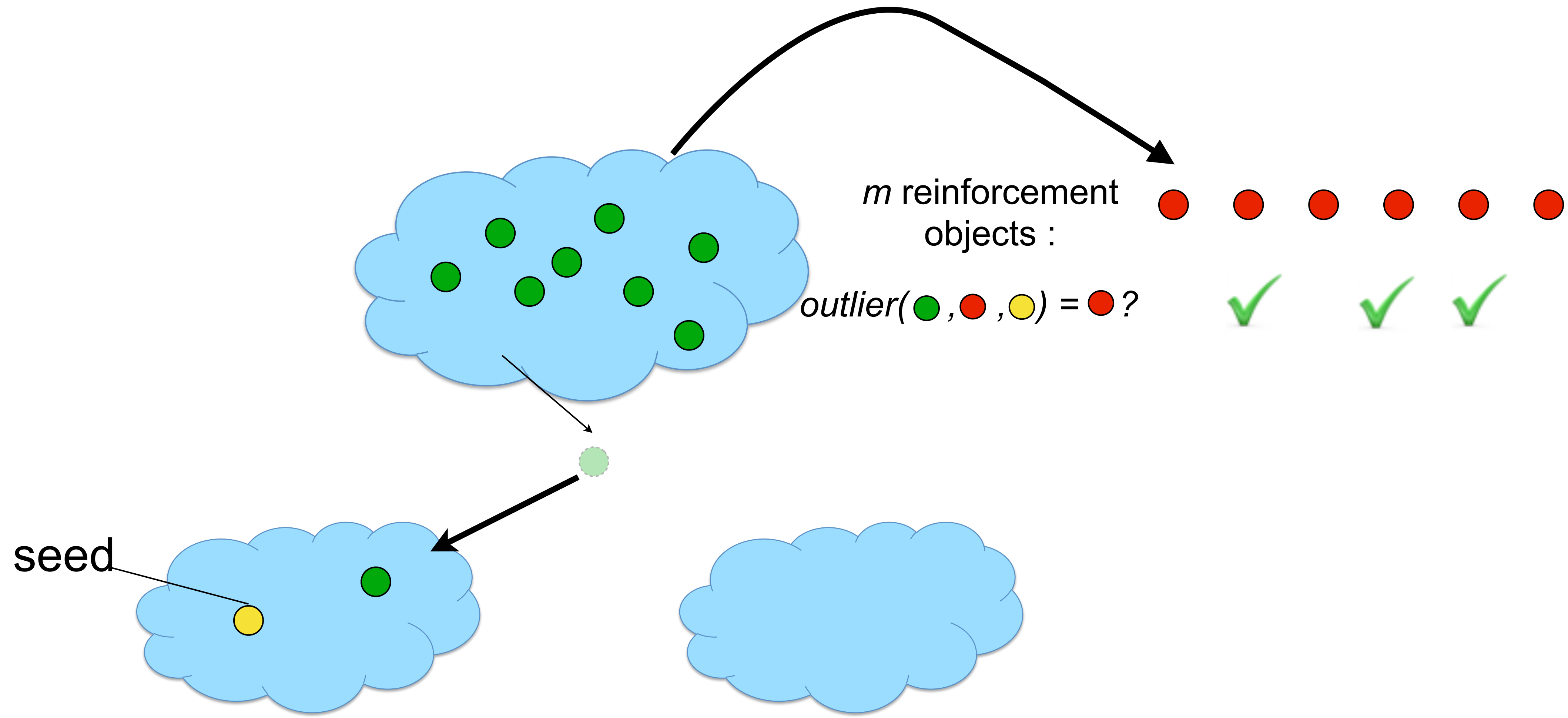
Strategy: Sequentially decide which of the two sub-clusters each goes into.

1. Pick a random object and call it the "seed".

2. For the other objects, decide if they are similar to or not.

3. Towards this, randomly pick *m* "reinforcement" objects from *C.* Count the number of times outlier(, ,) is .

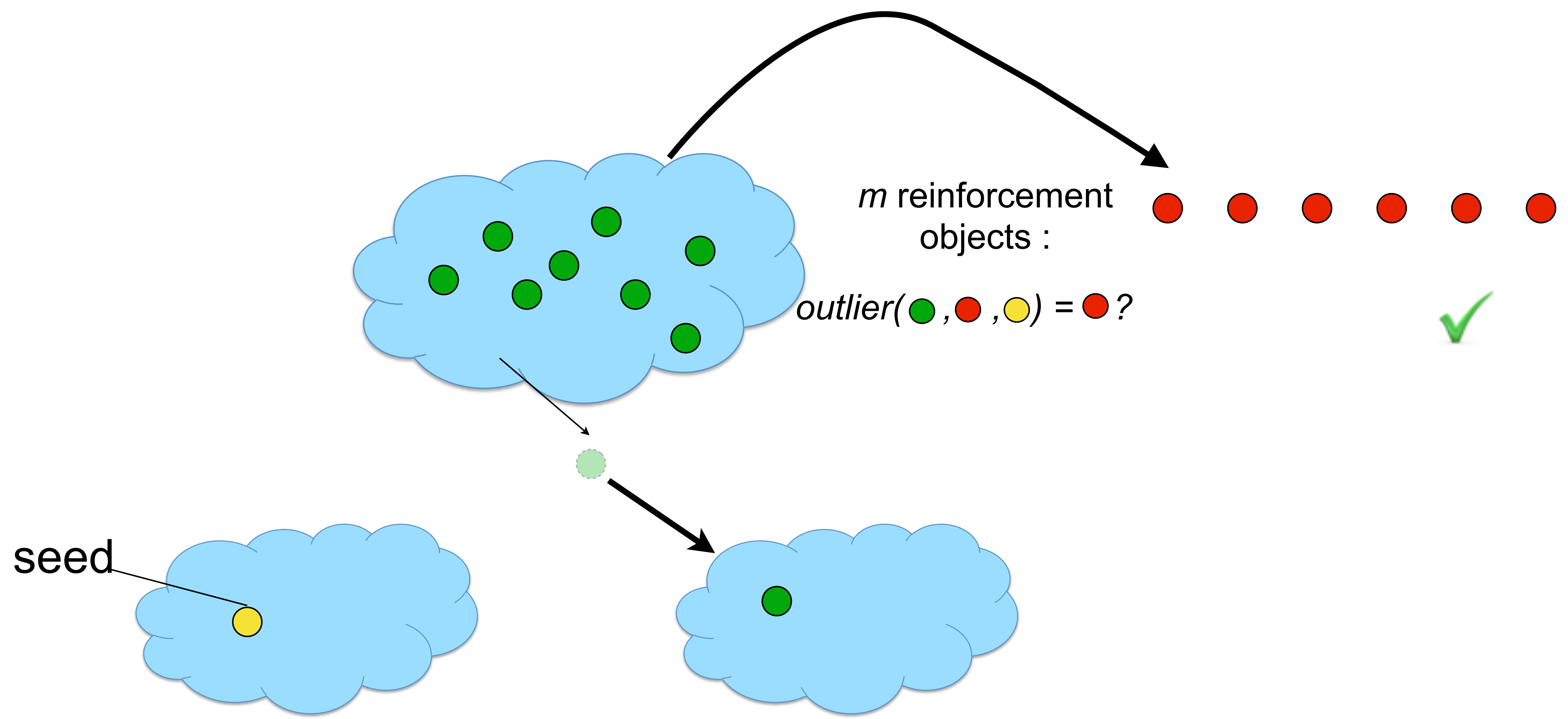
Robust Active Clustering Procedure



Strategy: Sequentially decide which of the two sub-clusters each goes into.

1. Pick a random object and call it the "seed".
2. For the other objects, decide if they are similar to or not.
3. Towards this, randomly pick *m* "reinforcement" objects from *C.* Count the number of times outlier(, ,) is .
4. If roughly *m/2* times, is **similar** to .

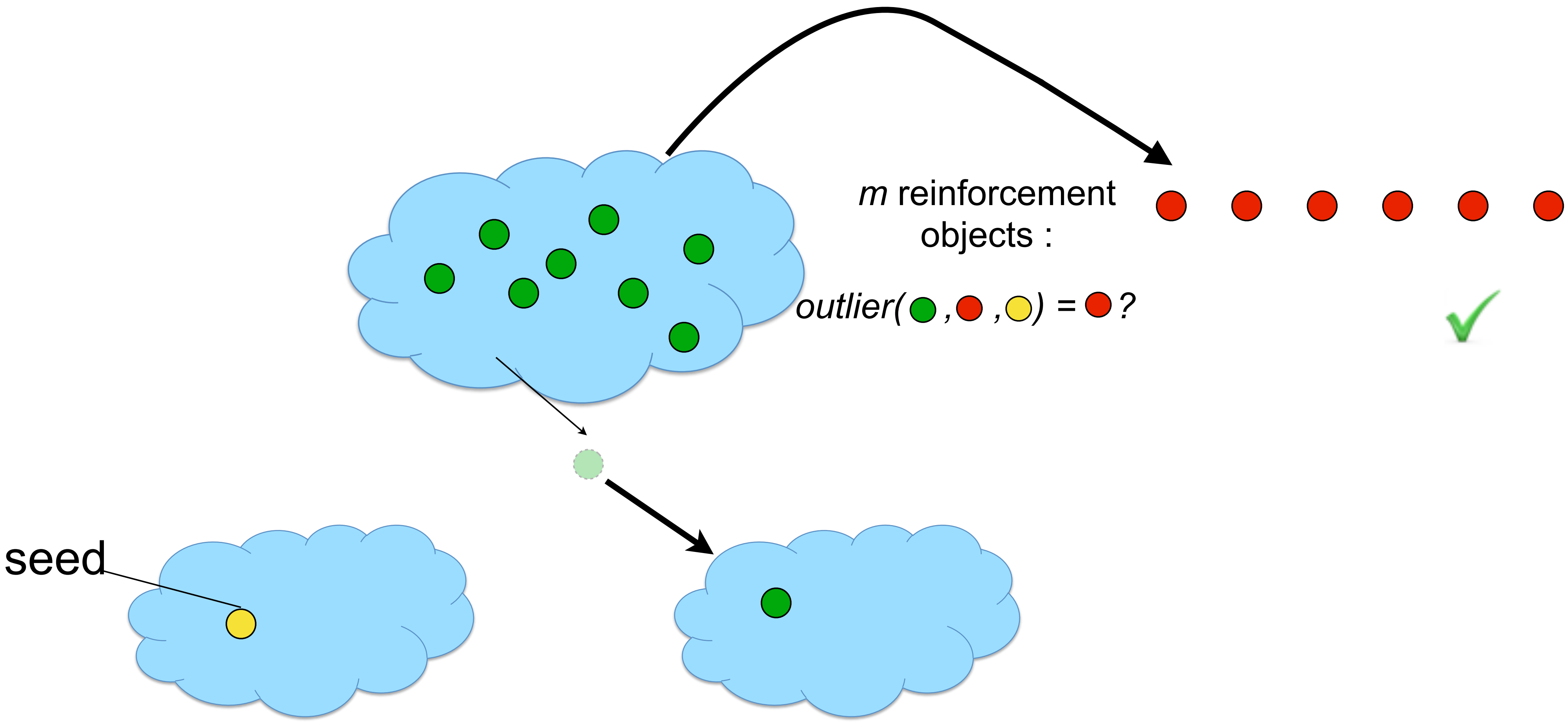
Robust Active Clustering Procedure



Strategy: Sequentially decide which of the two sub-clusters each ● goes into.

1. Pick a random object and call it the "seed".
2. For the other objects, decide if they are similar to ● or not.
3. Towards this, randomly pick *m* "reinforcement" objects from *C*. Count the number of times outlier(●,●,●) is ●.
4. If roughly *m/2* times, ● is **similar** to ●. If almost never, ● goes in the other cluster.

Robust Active Clustering Procedure

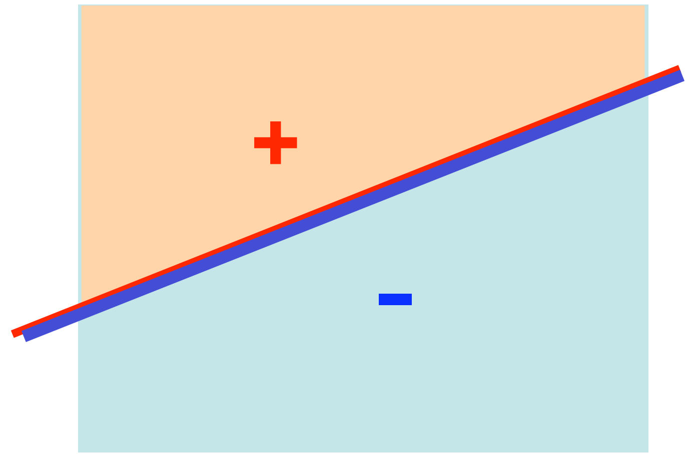


Strategy: Sequentially decide which of the two sub-clusters each goes into.

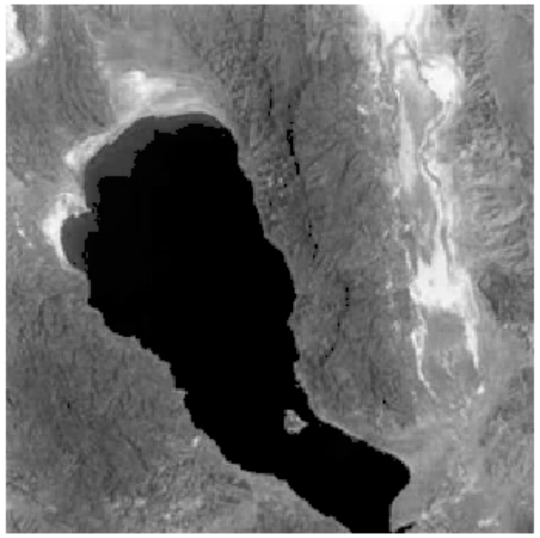
1. Pick a random object and call it the “seed”.
2. For the other objects, decide if they are similar to or not.
3. Towards this, randomly pick *m* “reinforcement” objects from *C.* Count the number of times outlier(, ,) is .
4. If roughly *m/2* times, is **similar** to . If almost never, goes in the other cluster.

Theorem: This procedure correctly clusters n objects using **O(n $\log^2$ n)** similarities and is robust to a significant fraction of errors.

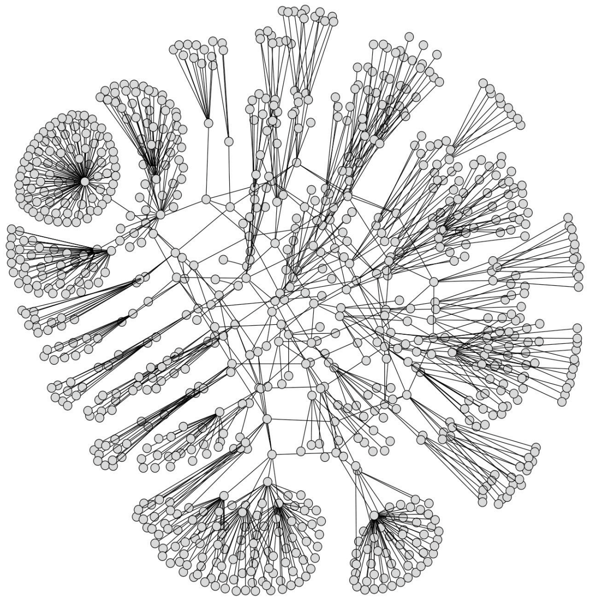
Active Learning Summary



Classification:
NA $\Rightarrow$ sample complexity $n \sim d/\epsilon$
A $\Rightarrow$ sample complexity $n \sim d \log \epsilon^{-1}$



Remote Sensing:
NA $\Rightarrow$ error $\sim O(n^{-1/2})$
A $\Rightarrow$ error $\sim O(n^{-2})$



Network Mapping:
NA $\Rightarrow O(n^2)$ probes
A $\Rightarrow O(n \log n)$ probes

Related Work (an incomplete list)

Active learning
Kulkarni, Mitter, &Tsitsiklis (1993), Cohn, Atlas & Ladner (1994), P. Hall & I. Molchanov (2003), Willett, Castro & Nowak (2005), Dasgupta (2004,2005), Balcan, Beygelzimer & Langford (2006), Kääriäinen (2006), Hanneke (2007), Dasgupta, Hsu, Monteleoni (2007), Castro & Nowak (2008), Beygelzimer, Dasgupta & Langford (2009), Hanneke (2011)

Minimax Analysis of Statistical Learning
Marron (1983), Yatrocos (1985), Barron (1991), Korostelev & Tsybakov (1993), Mammen & Tsybakov (1999), Tsybakov (2004), Scott & Nowak (2006)

Binary Search and Learning by Queries
Rivest, Meyer, & Kleitman (1980), Hegedüs (1995), Hellerstein et al (2006), Karp & Kleinberg (2007)

Channel Coding with Feedback (just the classics)
Horstein (1963), Schalkwijk & Kailath (1966), Burnashev & Zigangirov (1974), Burnashev (1976)