CS57300
PURDUE UNIVERSITY

MARCH 19, 2019

DATA MINING

ANNOUNCEMENT

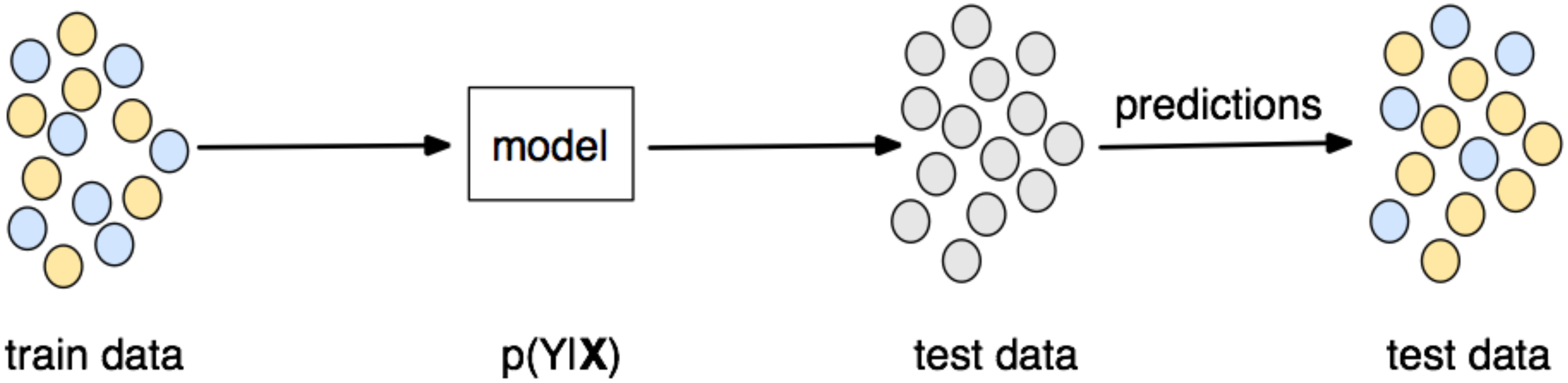
- Midterm grade is out!
  - Mean: 40.1, median 38, standard deviation: 8.1
- Assignment 4 is out!
  - Implement decision trees, bagging, and random forests
  - Due on March 31 (Sunday), 11:59pm
  - If you use any extension days, specify it clearly on your pdf report!

ANNOUNCEMENT

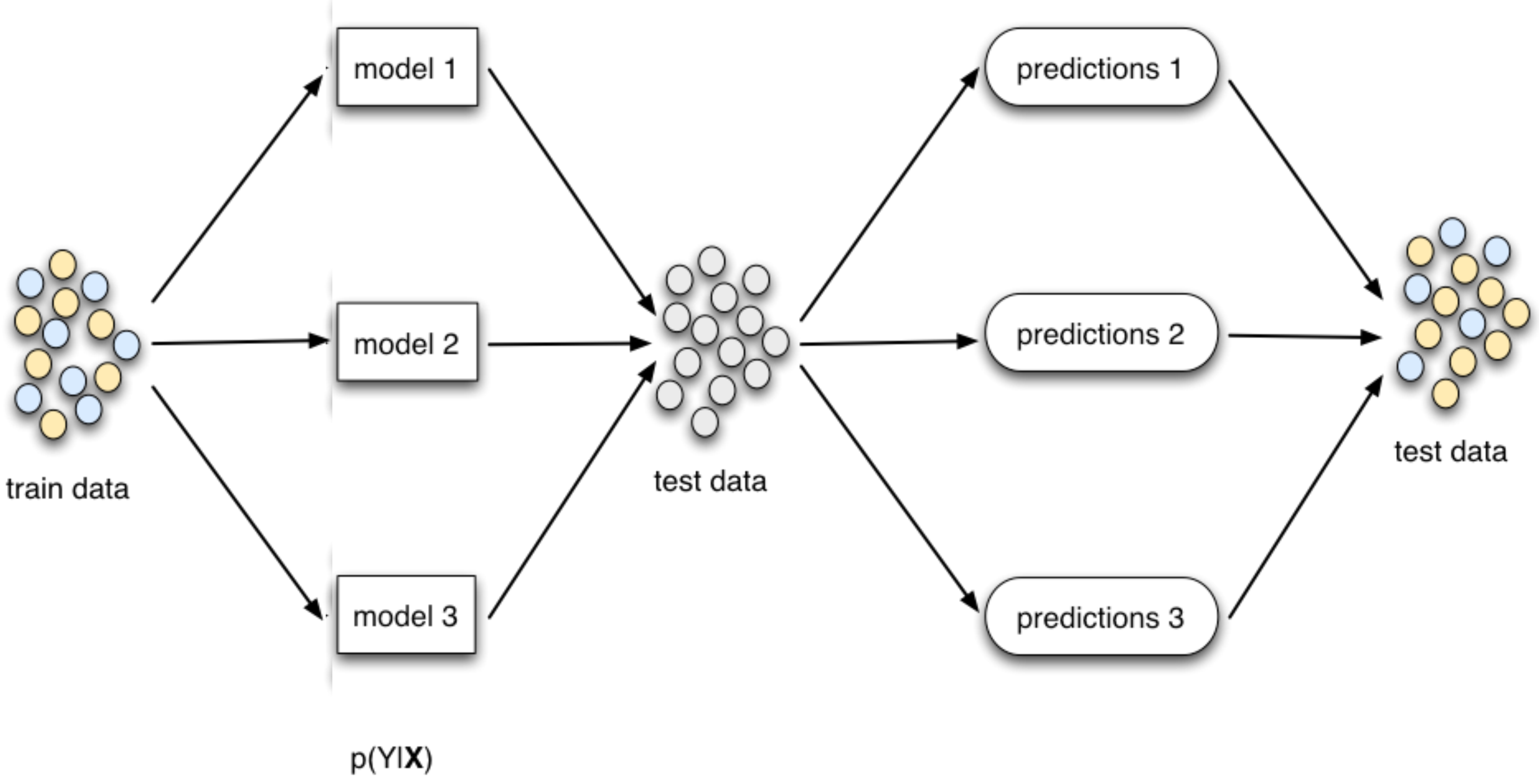
- Final project pitch
  - In-class pitch (March 26, Tuesday): 2 minutes of presentations; 1 minute of Q&A
  - Content to include: (1) the topic you proposed to work on; (2) why you are excited about it; (3) what's the expected outcome of your project
  - Pitch slides due on March 24 (Sunday), 11:59pm
  - Pitch presentation order will be decided soon
  - Distance student: please submit your pitch video via Blackboard before March 26, 11:59pm
- Next class: Guest lecture by Professor Yexiang Xue (**Deep learning**)

ENSEMBLE METHODS

CONVENTIONAL CLASSIFICATION



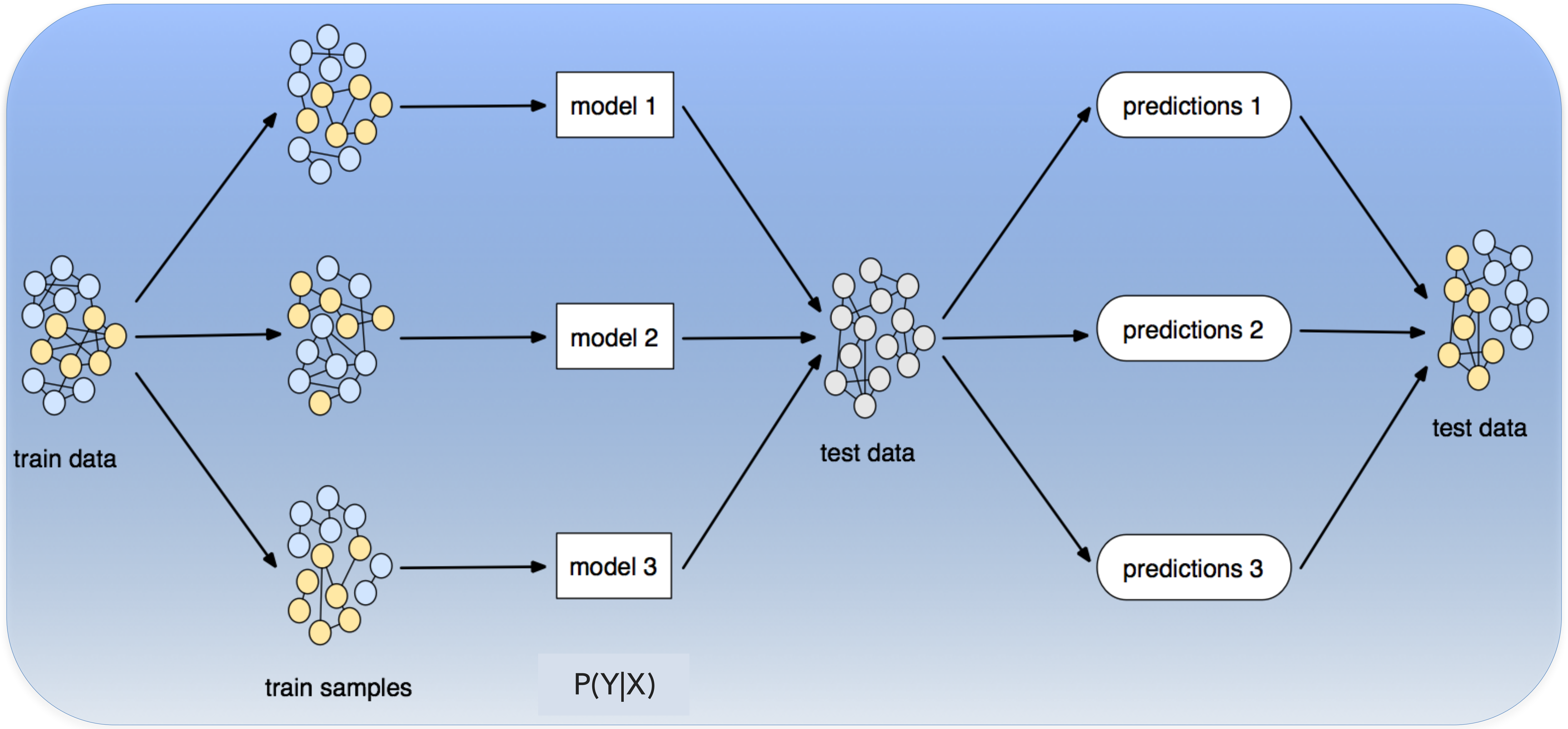
ENSEMBLE CLASSIFICATION



BAGGING

- **B**ootstrap **agg**regat**ing**
- Main assumption
  - Combining many *unstable* predictors in an ensemble produces a *stable* predictor (i.e., reduces variance)
  - Unstable predictor: small changes in training data produces large changes in the model (e.g., trees)
- Model space: non-parametric, can model any function if an appropriate base model is used

BAGGING



TREATMENT OF INPUT DATA

- sample with replacement

CHOICE OF BASE CLASSIFIER

- unstable predictor (e.g., fully-grown decision tree)

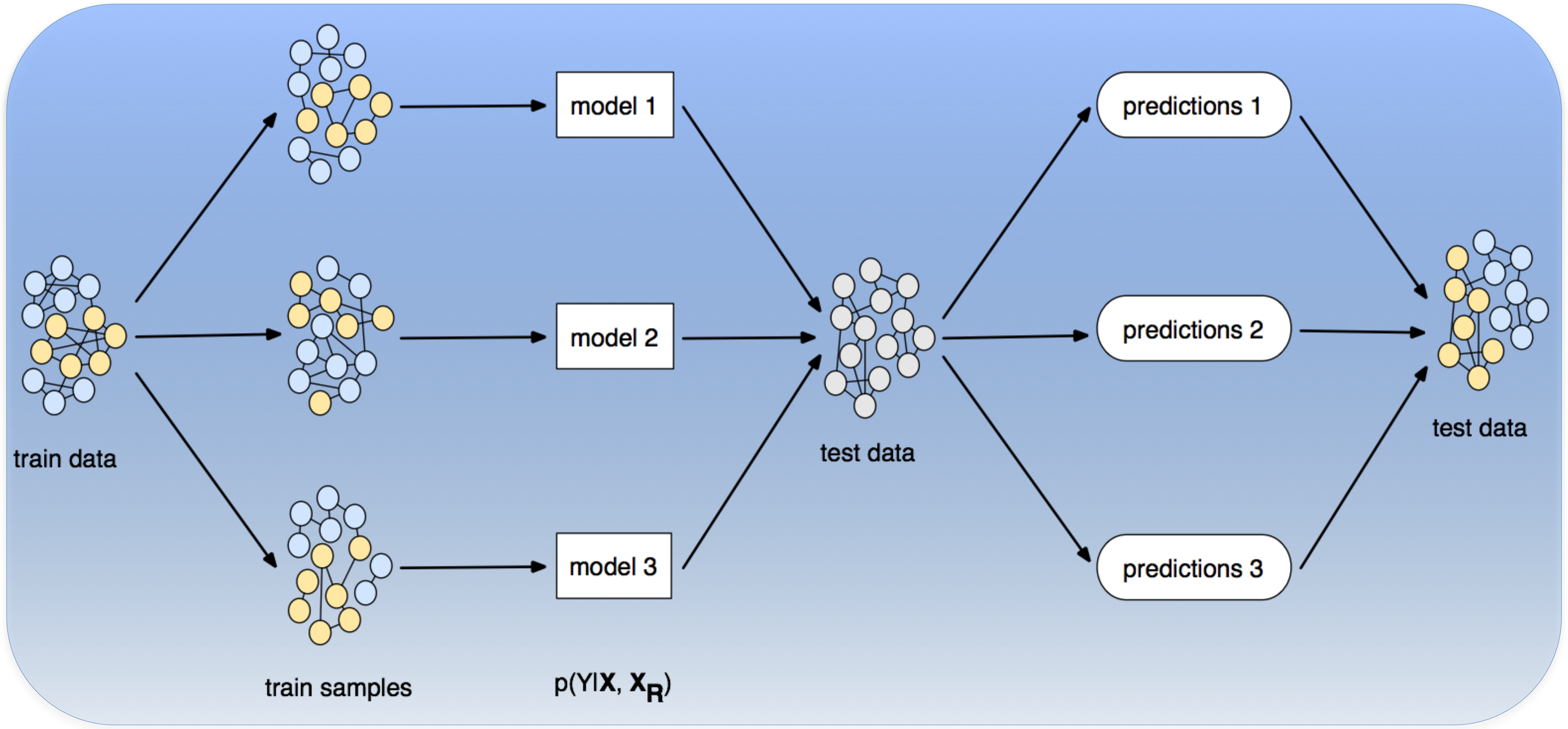
PREDICTION AGGREGATION

- averaging/majority voting

RANDOM FORESTS

- Random forests is a variant that aims to improve on bagged decision trees by reducing the correlation between the models
  - Each tree is learned from a bootstrap sample (same as before)
  - For each tree split, a random sample of *k* features is drawn first, and **only** those features are considered when selecting the best feature to split on (typically $k=\sqrt{p}$ or $k=\log p$, p is the total number of features)

RANDOM FORESTS

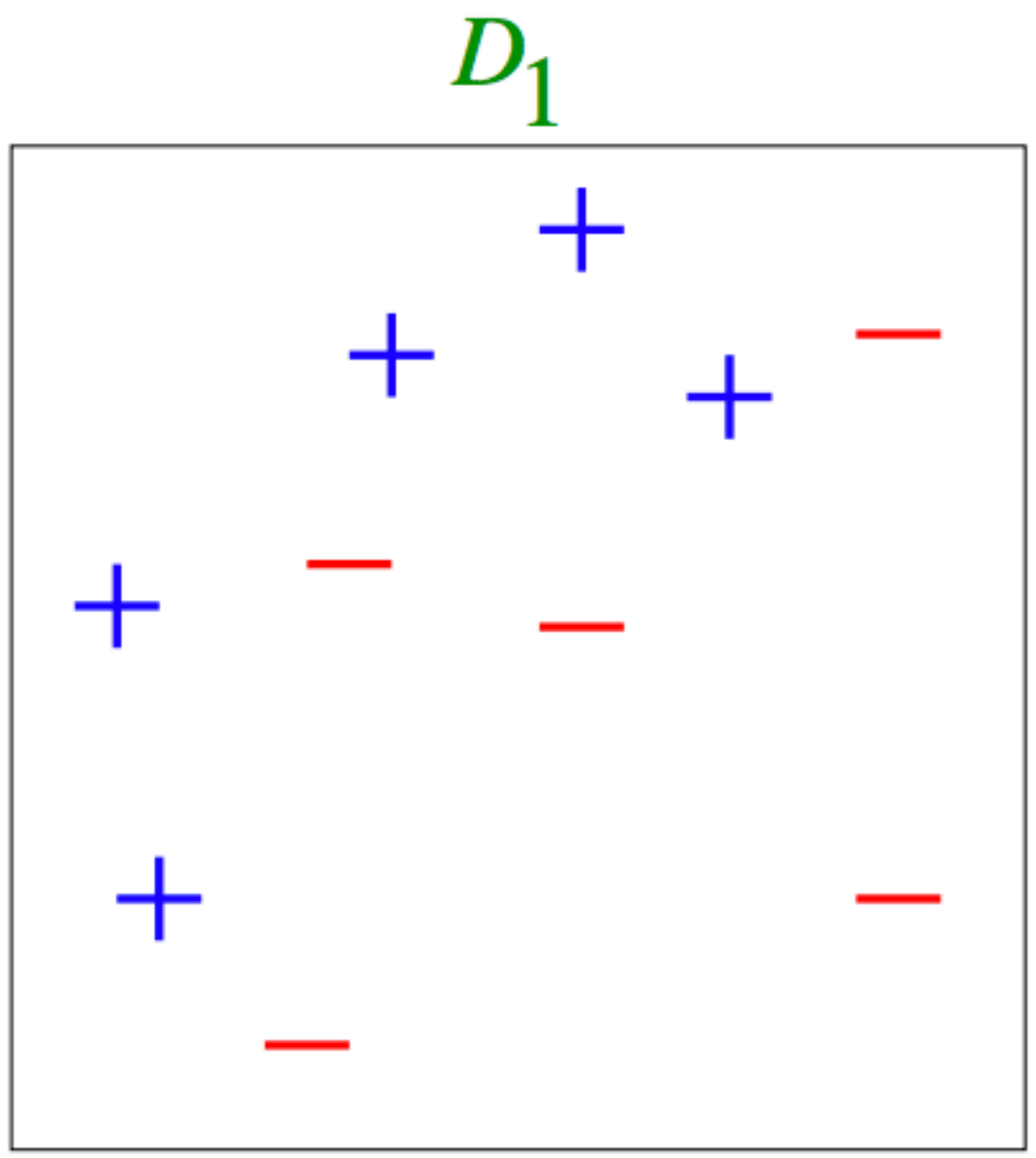


TREATMENT OF INPUT DATA	CHOICE OF BASE CLASSIFIER	PREDICTION AGGREGATION
• sampling with replacement	• decision tree (limited attributes are considered at each node)	• averaging/majority voting

BOOSTING

- Bagging and random forests share the same idea of combining multiple models that are trained on bootstrapped samples of the training data
  - Mimic learning the model from different training data
  - Each model has an equal amount of say (i.e., equal weights) in influencing the aggregated prediction
- Boosting
  - Combine multiple “complementary” models
  - Aggregate model predictions by considering how accurately each model can predict

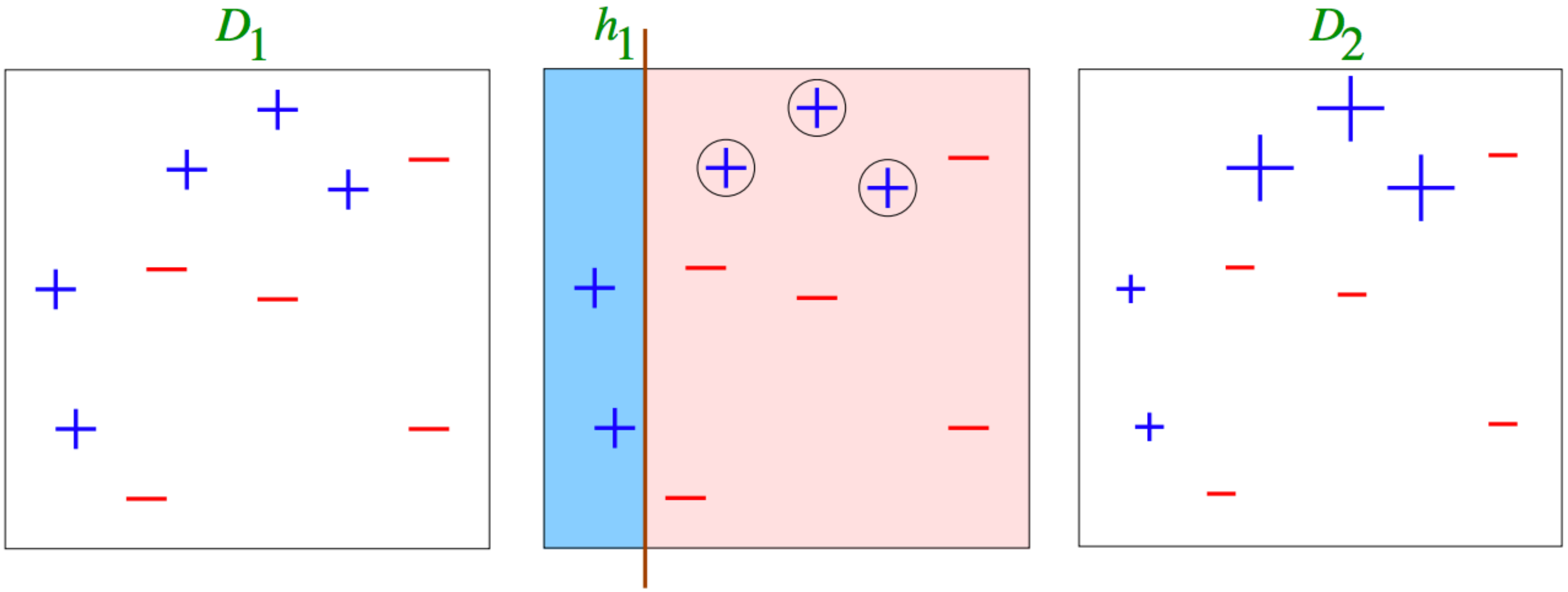
BOOSTING EXAMPLE



Model: Decision stump

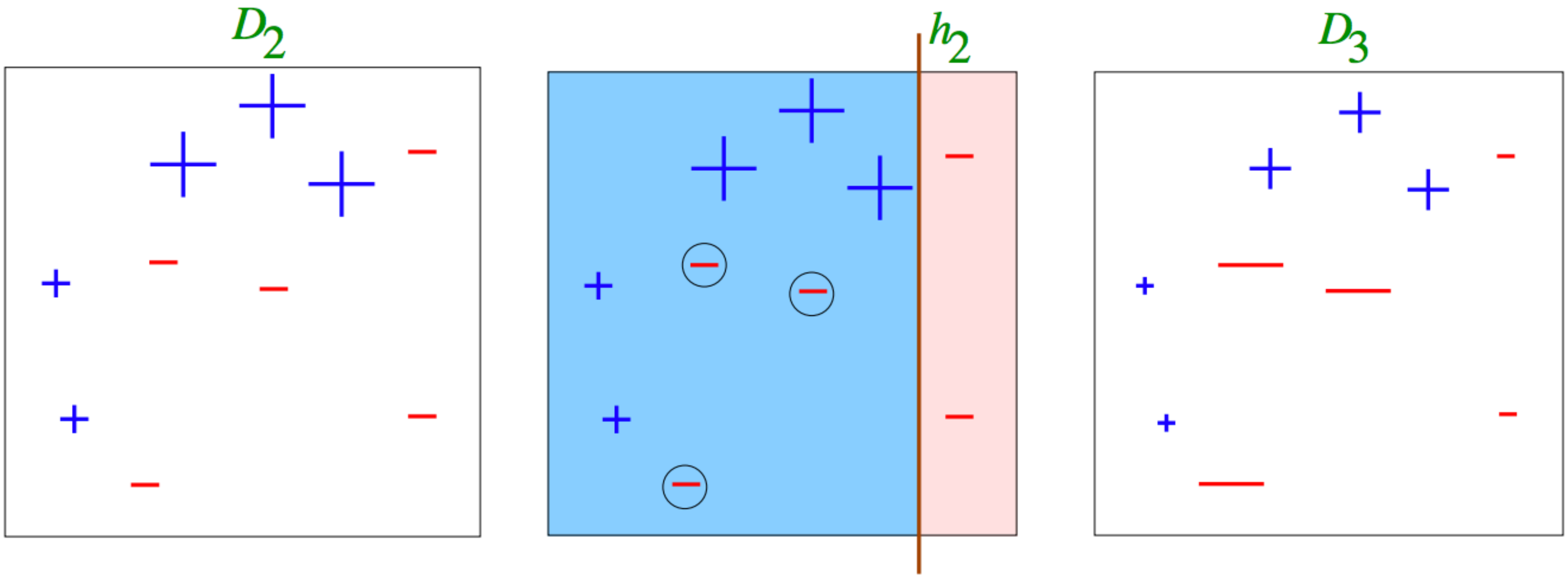
If $x_i > c$, then "+"; otherwise "-"

BOOSTING EXAMPLE: ROUND 1

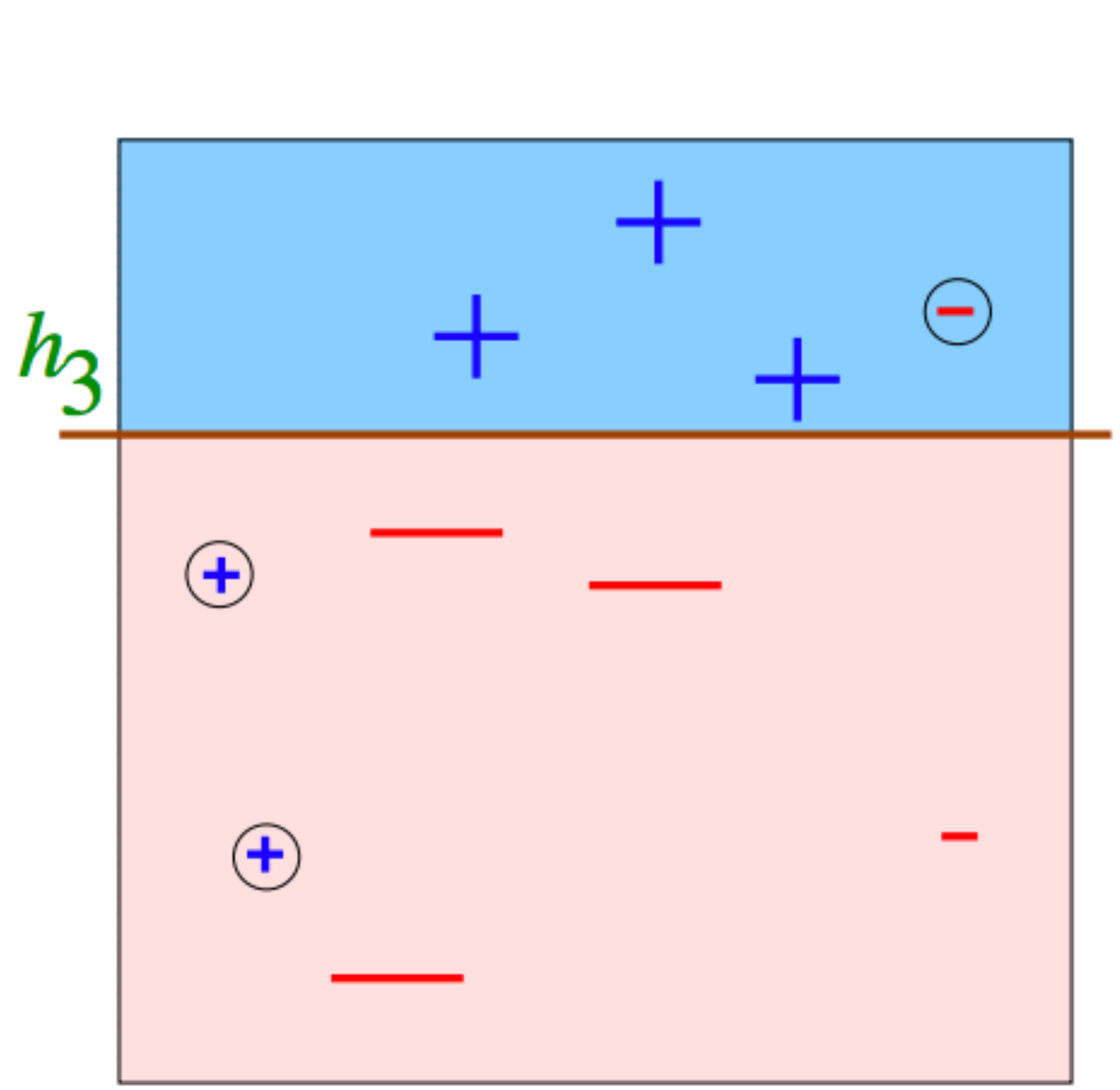


Construct "complementary" models? Re-weighting!

BOOSTING EXAMPLE: ROUND 2

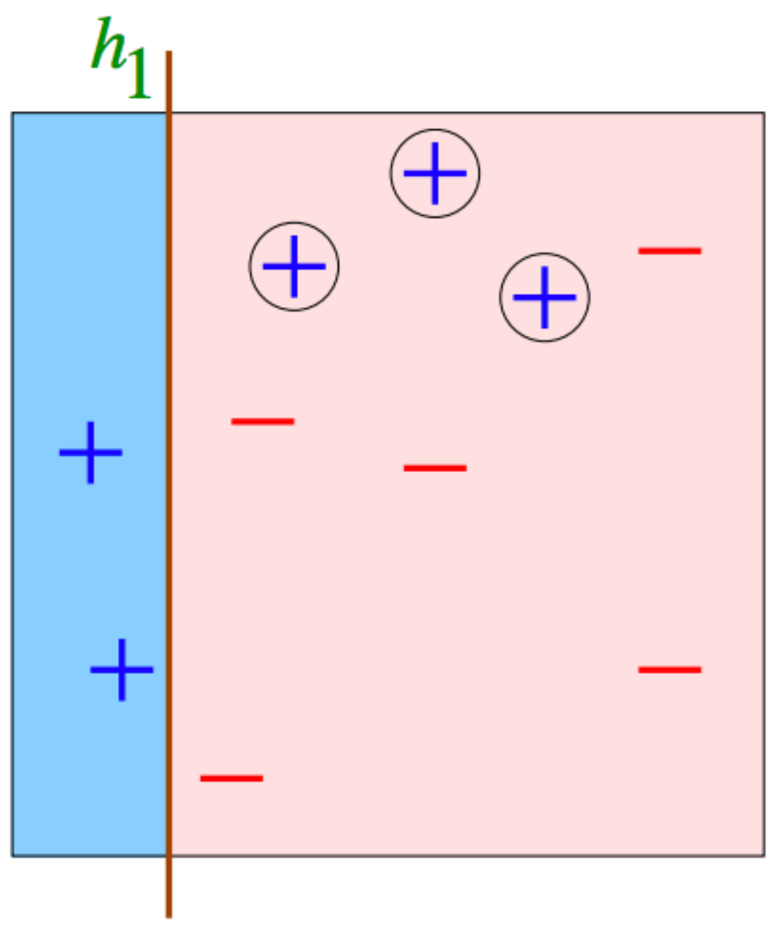


BOOSTING EXAMPLE: ROUND 3

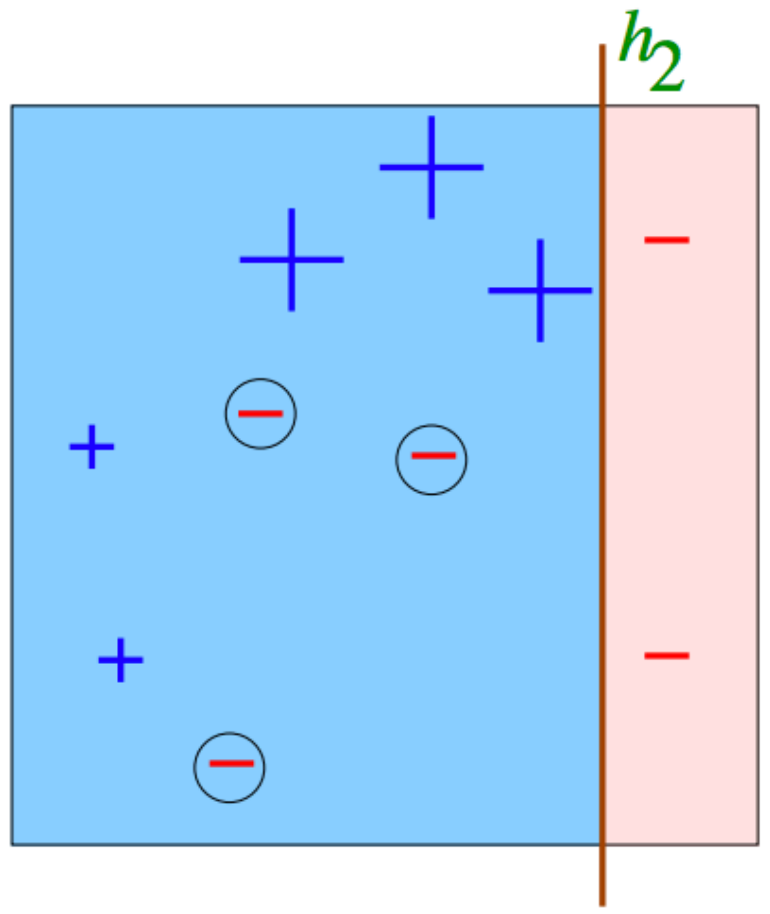


BOOSTING EXAMPLE: AGGREGATING

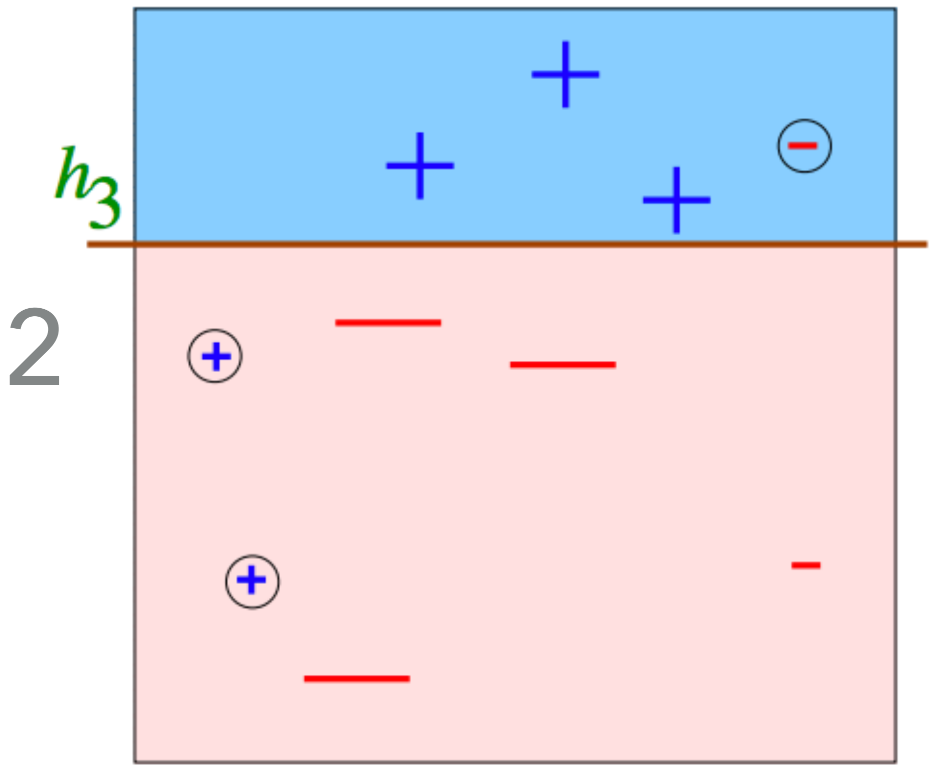
$H_{final} = \Big(0.42$



$+0.65$

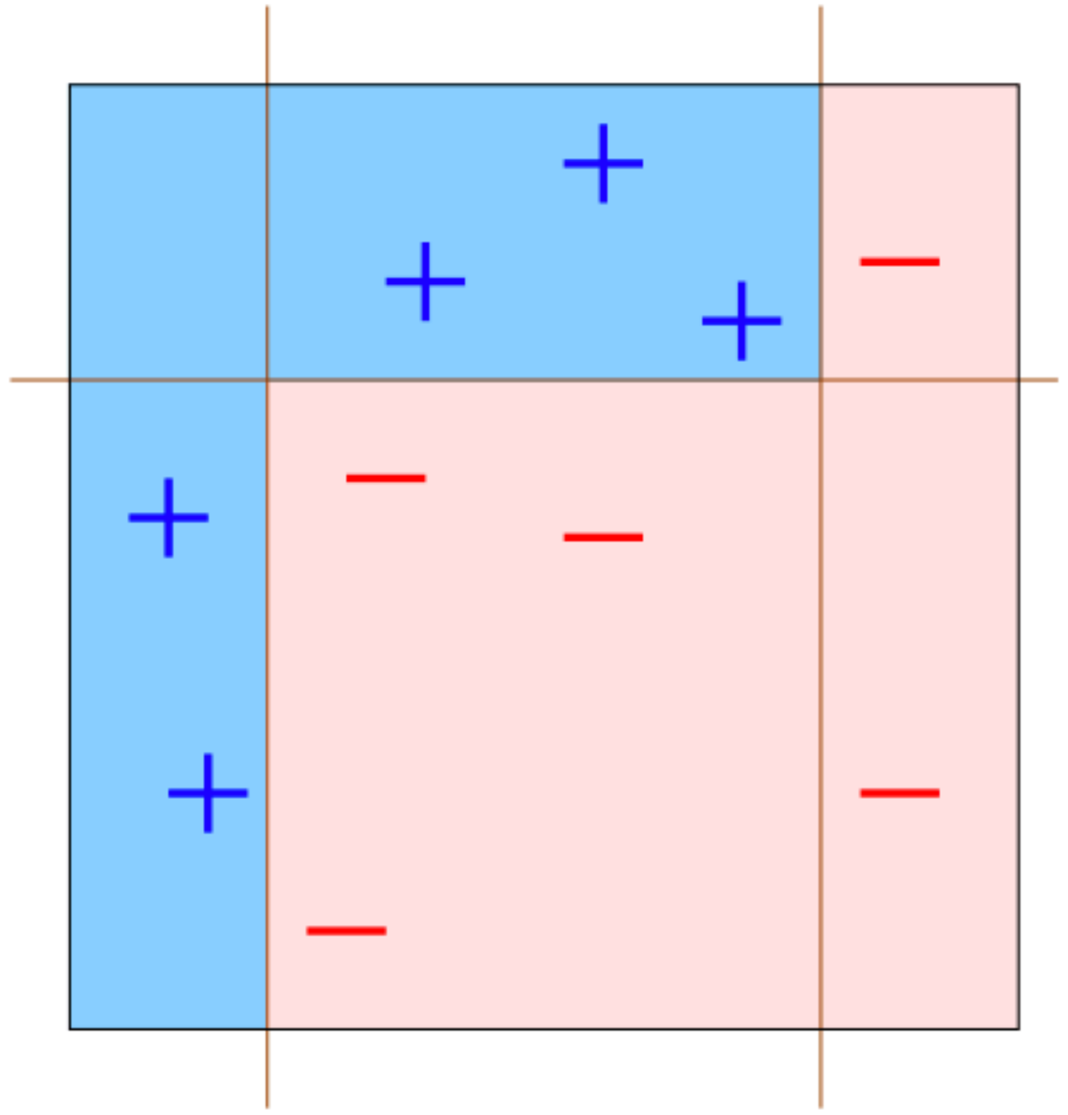


$+0.92$



$\Big)$

$=$



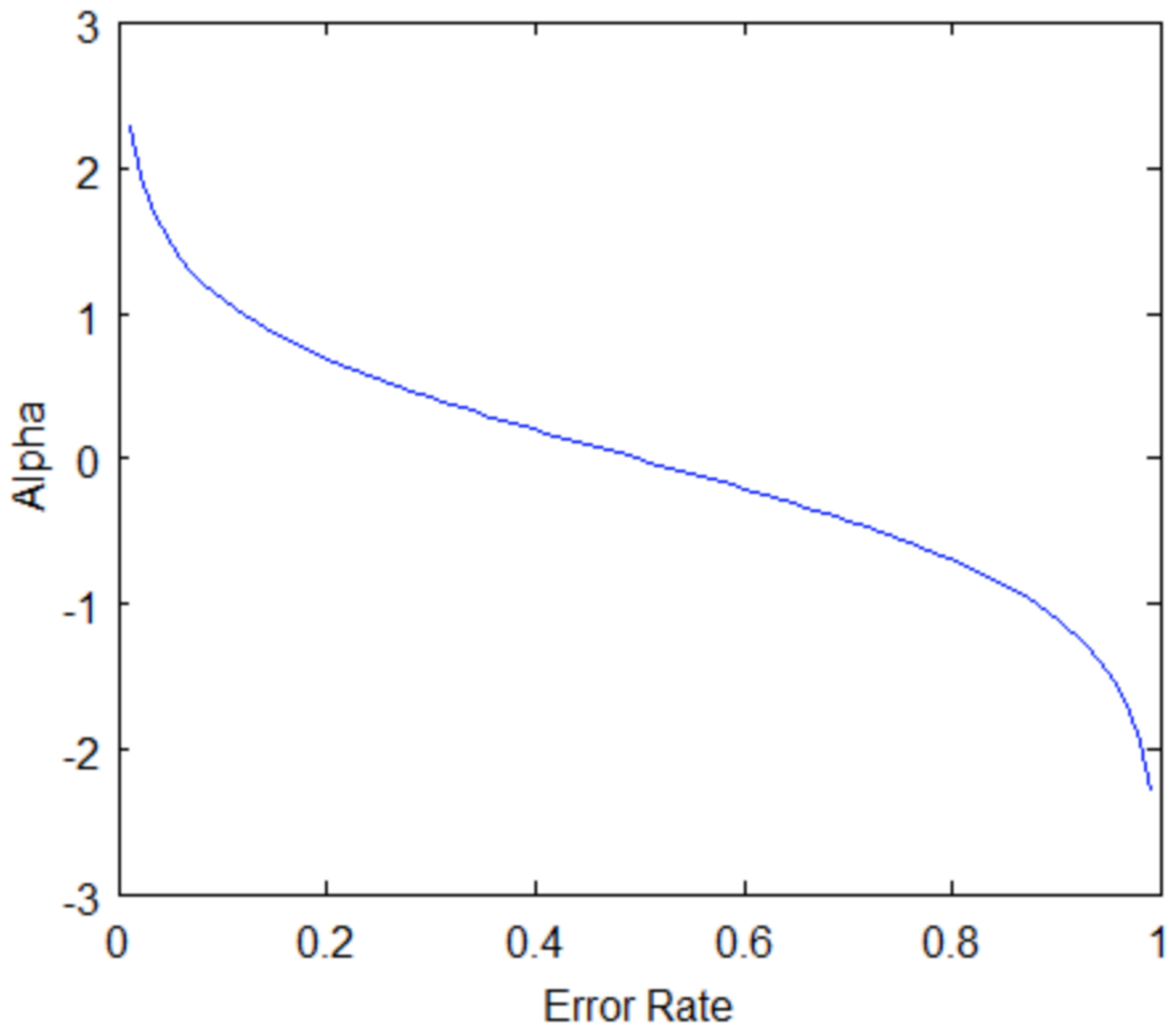
ADABOOST

- Given N training examples $(x_1, y_1), \ldots, (x_N, y_N)$, assign every example in with an equal weight $D_1(i)=1/N$
- For t=1:T
  - Learn model $h_t(x)$ to minimize the weighted error: $\epsilon_t = \mathrm{Pr}_{i \sim D_t}[h_t(x_i) \neq y_i] = \sum_{i=1}^{N} D_t(i)\mathrm{I}(h_t(x_i) \neq y_i)$
  - Set the weight of this model: $\alpha_t = \frac{1}{2} ln(\frac{1-\epsilon_t}{\epsilon_t})$
  - Update training example weights: up-weight the examples that are incorrectly classified and downright examples that are correctly classified: $D_{t+1}(i) = \frac{D_t(i)exp(-\alpha_t y_i h_t(x_i))}{Z_t}$ where $Z_t = \sum_{i=1}^{N} D_t(i)exp(-\alpha_t y_i h_t(x_i))$ is a normalization factor
- To classify new test instance x', apply each model $h_t(x)$ to x' and take weighted vote of predictions

$$H(x') = \mathrm{sign}(\sum_{t=1}^{T} \alpha_t h_t(x'))$$

BOOSTING INTUITION: UNDERSTANDING ALPHA

$$\alpha_t = \frac{1}{2} ln(\frac{1-\epsilon_t}{\epsilon_t})$$



Low error rate: Large (positive) voting power

Error rate close to 0.5: small voting power

High error rate: Large (negative) voting power

BOOSTING INTUITION: UNDERSTANDING RE-WEIGHTING

$$D_{t+1}(i) = \frac{D_t(i)exp(-\alpha_t y_i h_t(x_i))}{Z_t}$$

- When $h_t(x_i) = y_i$, the prediction is correct; $D_{t+1}(i) \propto D_t(i)exp(-\alpha_t)$
- When $h_t(x_i)$!= y_i, the prediction is incorrect; $D_{t+1}(i) \propto D_t(i)exp(\alpha_t)$

WHY ADABOOST WORKS?

- Minimize exponential loss $\sum_{i=1}^{N} exp(-y_i f_T(x_i))$ greedily, where $f_T(x) = \sum_{t=1}^{T} \alpha_t h_t(x)$
- How to get $f_T(x)$ from $f_{T-1}(x)$?

$$\sum_{i=1}^{N} exp(-y_i f_T(x_i)) = \sum_{i=1}^{N} exp(-y_i f_{T-1}(x_i)) exp(-y_i \alpha_T h_T(x_i))$$

$$\propto \sum_{i=1}^{N} D_T(i) exp(-y_i \alpha_T h_T(x_i))$$

$$= \sum_{y_i \neq h_T(x_i)} D_T(i) e^{\alpha_T} + \sum_{y_i = h_T(x_i)} D_T(i) e^{-\alpha_T}$$

$$= \epsilon_T e^{\alpha_T} + (1 - \epsilon_T) e^{-\alpha_T} = \epsilon_T (e^{\alpha_T} - e^{-\alpha_T}) + e^{-\alpha_T}$$

Learn $h_T(x)$ to minimize ϵ_T

Set $\alpha_T = \frac{1}{2} ln(\frac{1 - \epsilon_T}{\epsilon_T})$

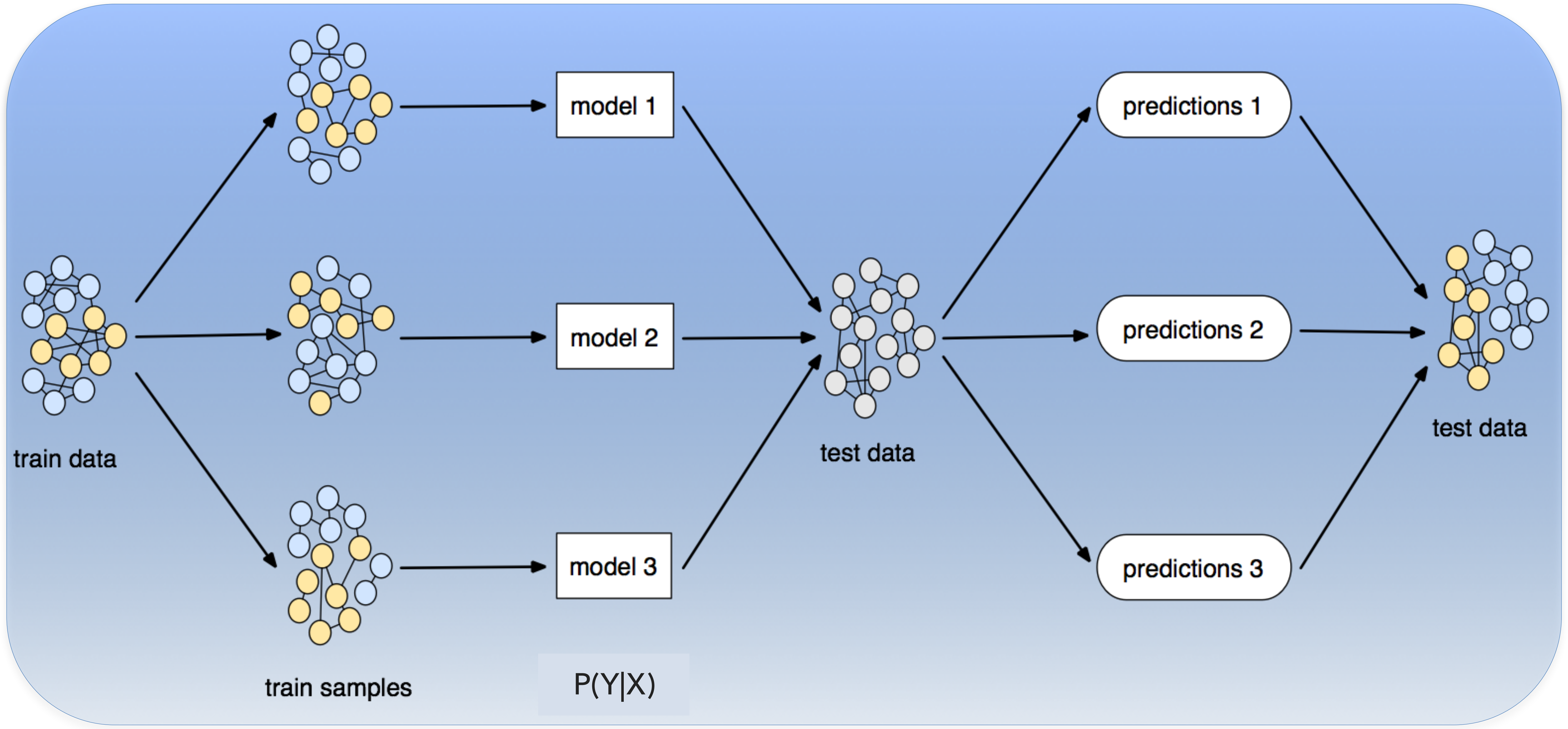
BOOSTING: HOW TO LEARN A MODEL ON WEIGHTED SAMPLES?

- Directly modify the scoring function
  - Weighted log likelihood $\sum_{i=1}^{N} D_t(i) log(\mathsf{P}(y_i | x_i))$ (e.g., logistic regression)
  - Weighted squared loss $\sum_{i=1}^{N} D_t(i)(y_i - o_i)^2$ (e.g, neural network)
- What about models that are learned through heuristic search (e.g., decision trees)?
  - Weighted version of selection criteria: $H(A) = -\sum_{v} wp(x_A = v) log(wp(x_A = v))$, where $wp(x_A = v) = \sum_{i=1}^{N} D_t(i)\mathsf{I}(x_i(A) = v)$
  - Re-sample the training examples according to D_t

BOOSTING

- Main assumption
  - Combining many *weak* (but stable) predictors in an ensemble produces a *strong* predictor (i.e., reduces bias)
  - Weak predictor: only weakly predicts correct class of instances (e.g., decision stumps)
- Model space: non-parametric, can model any function if an appropriate base model is used

BOOSTING



TREATMENT OF INPUT DATA

- re-weight examples

CHOICE OF BASE CLASSIFIER

- weak predictor (e.g., decision stump)

PREDICTION AGGREGATION

- weighted vote