INTRODUCTION TO MACHINE LEARNING COMPSCI 4ML3

Lecture 29

Hassan Ashtiani

ENSEMBLE LEARNING

SIMPLE RECIPE TO IMPROVE THE PERFORMANCE:

1. TRAIN MULTIPLE CLASSIFIERS
2. AGGREGATE THEIR DECISIONS
   - E.G., VOTING

THE RESULT CAN BE BETTER THAN THE INDIVIDUAL CLASSIFIERS!

- REDUCING VARIANCE?
- REDUCING BIAS?

THE NETFLIX CHALLENGE

- Given some user ratings for various films,
  - Predict user ratings for other films
- Collaborative filtering

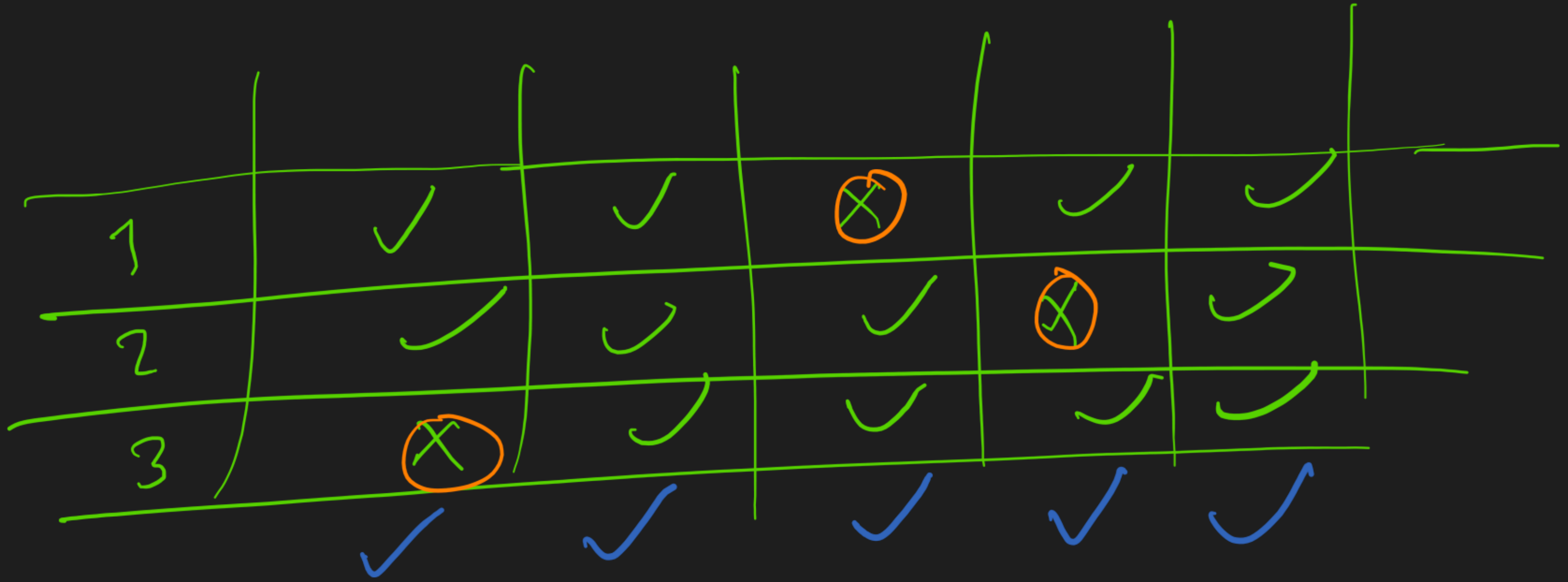
THE NETFLIX CHALLENGE

- 2006: Competition began, 1M USD for improving 10% over Netflix's own method
- 2007: 8.43% improvement (BellKor won 50k)
- 2008: No individual team better than 9.43%
  - BellKor+BigChaos merged… >9.43% improvement!
- 2009: Top three merge! BellKor+BigChaos+Pragmatic >10%
  - New team in the last month: GrandPrizeTeam (9.46%)
    - Anyone could join, and share the prize based on the improvement
  - Ensemble: GrandPrizeTeam+Vanderlay Industries (>10%)

  BellKor+BigChaos+Pragmatic (10.09%) and Ensemble (10.10%)
  - They both get the exact same accuracy on the private test set!
  - BellKor+BigChaos+Pragmatic submitted 20mins earlier….
- 2007: linkage attacks with IMDB
- 2010: Privacy concerns and class-action lawsuits…competition was canceled

AGGREGATION

- When does combining models help?
  - The base learners should be accurate
  - The base learners should be diverse (less correlated)
- Example for classification



AGGREGATION

- Example for regression

1	1.1			
2	1.5			
3	0.9			

median/mean/...

- Aggregation can reduce the variance
  - Helps tackling overfitting
- How to diversify the base learners?

THE ENSEMBLE

How to create a diverse ensemble of learners?

- Different classifiers (neural nets, linear, nearest neighbor,...)
- Different hyper-parameters
  - Weight initialization in neural networks (random seed)
  - Network architectures
- Different training subsets
- Different feature subsets

BAGGING

- Using non-overlapping training subsets creates truly independent/diverse classifiers
  - i.i.d. assumption!
- But can be wasteful
  - Each classifier is trained using only a small train set…
- Bagging (bootstrap aggregating)
  - Random sampling with replacement!

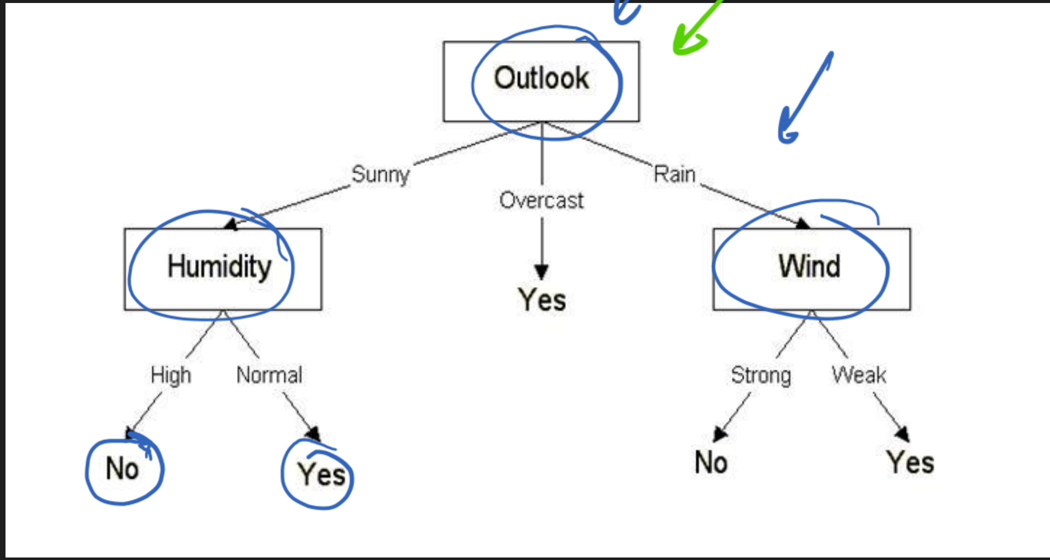
RANDOM SUBSPACE METHOD

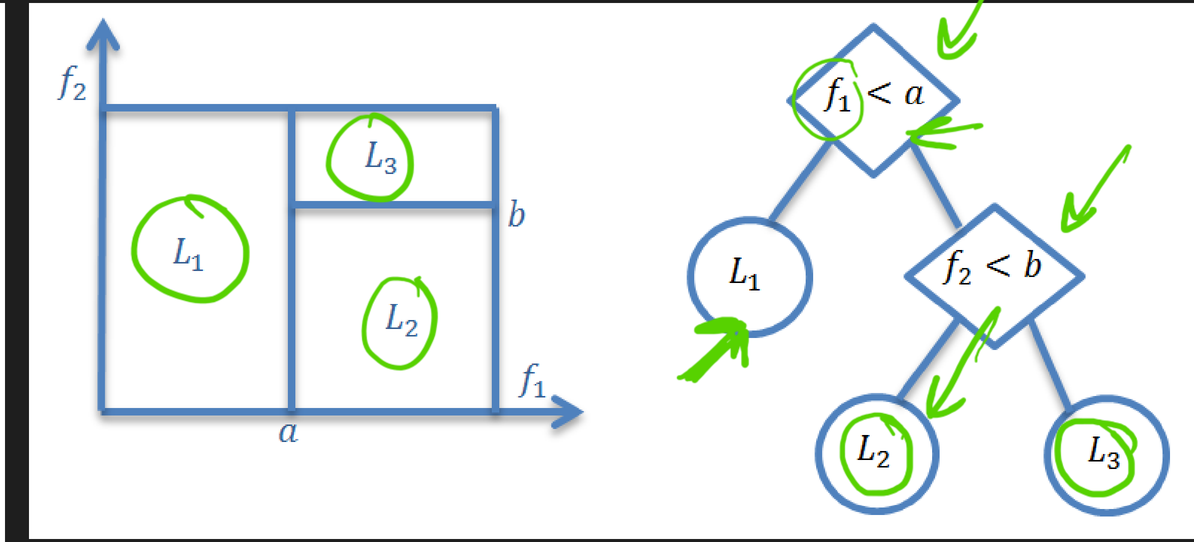
- Train each classifier using a random subset of features
- So each classifier operates in a random subspace
- Also called feature bagging, or attribute bagging
- Are the classifiers diverse?
  - There is correlation between the features
  - There is only so much you can learn from a data point

RANDOM FORESTS

- COMBINES THE IDEAS OF BAGGING AND RANDOM SUBSPACE METHODS
- USES DECISION TREES AS BASE CLASSIFIERS

DECISION TREE

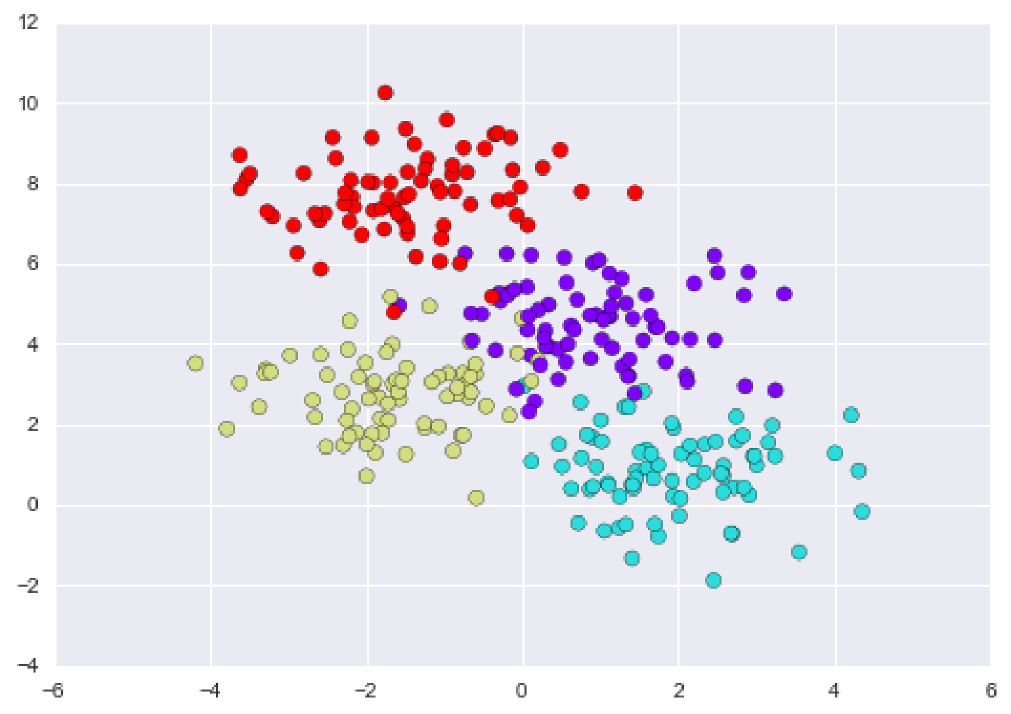


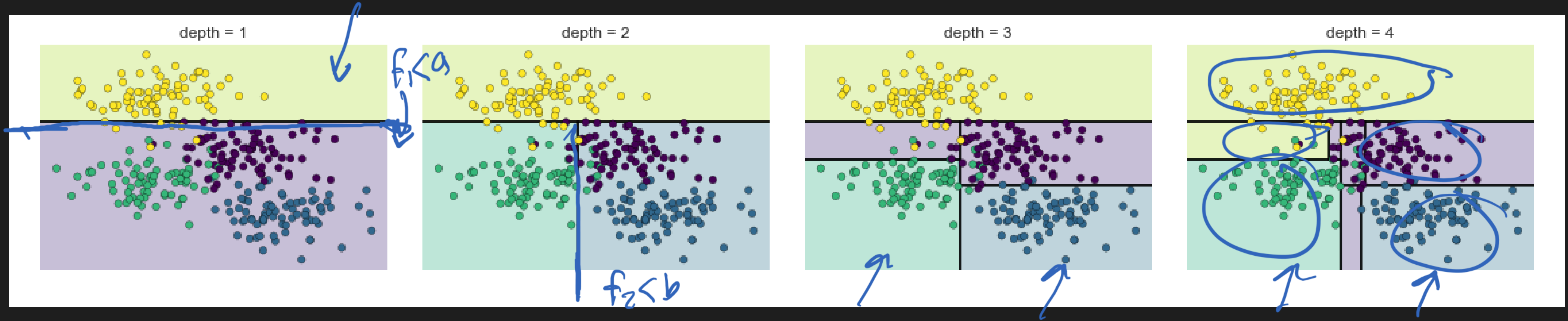


- Can handle categorical features
- Deeper trees can overfit easily
- How do you "train" decision trees?

DECISION TREES

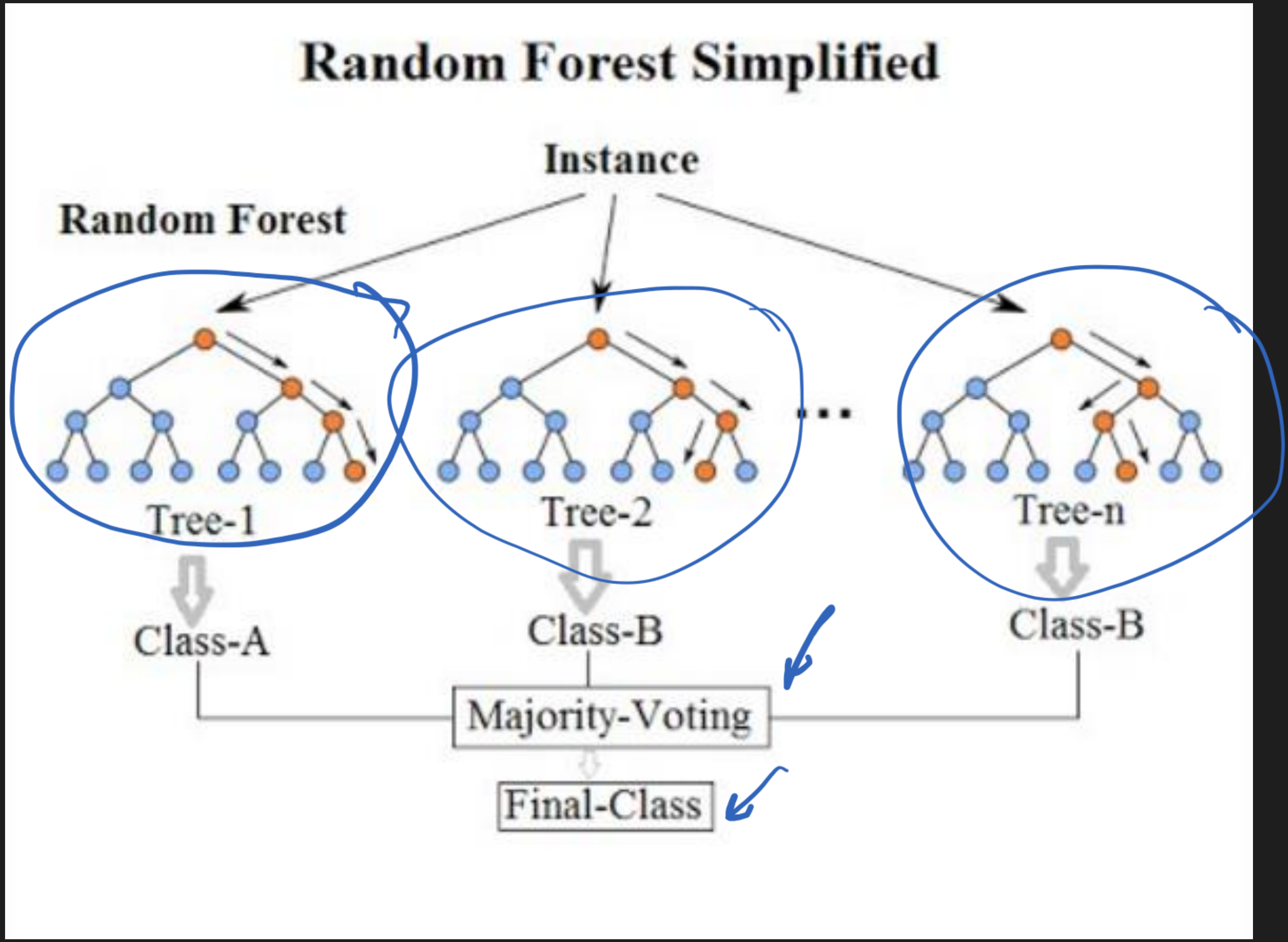
- A possible approach: Select the tree nodes randomly!
- Label leaves by doing majority vote in the training data





RANDOM FORESTS

- CREATE MANY DEEP RANDOM TREES
- USE RANDOM SUBSETS OF DATA FOR EACH TREE TO DETERMINE THE LABEL OF LEAVES
- FOR A TEST POINT, TAKE MAJORITY VOTE BETWEEN THE TREES



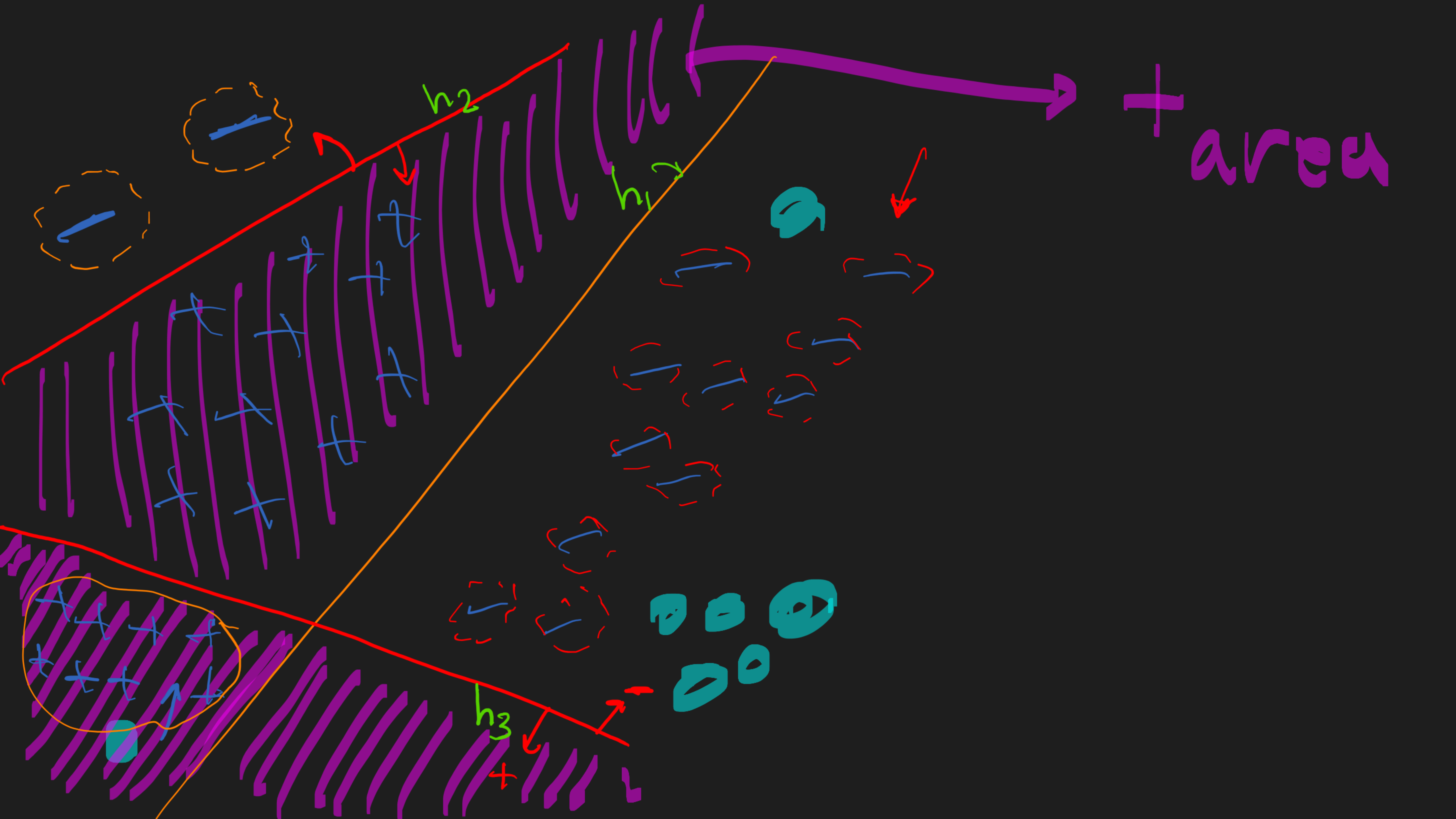
BOOSTING

- Up to now we picked the base classifiers independently
- The goal was to reduce variance
- But can we combine classifiers to reduce bias?
  - A "stronger classifier" out of "weak classifiers"?

BOOSTING

A GREEDIER APPROACH

- PICK THE BASE CLASSIFIERS ONE-BY-ONE (INCREMENTALLY)
- EACH NEW CLASSIFIER (CALLED A WEAK LEARNER) TRIES TO ADDRESS THE SHORTCOMINGS OF THE PREVIOUS ONES
- THE COMBINATION OF "WEAK LEARNERS" CAN BE A "STRONG LEARNER"



+ area
h2
h1
h3

TRAINING ON A WEIGHTED DATA SET

- Regular training
  - $\min_{\theta} \frac{1}{n} \sum_{i=1}^{n} l(y^i, h_\theta(x^i))$
- Weighted training
  - $\min_{\theta} \sum_{i=1}^{n} D_i . l(y^i, h_\theta(x^i))$
- We can put more emphasis on some of the training points

BOOSTING

1. Initialize the weights of all training points to be equal
2. Do for a number of iterations:
   - Train a weak learner for the weights (from the base class)
   - Store the accuracy of this weak learner (α_i) $\boxed{1 - \varepsilon_i \approx \alpha_i}$
   - See where the learner makes mistakes
   - Increase the weights of those misclassified points (D_j)
     - So that they are classified correctly in the next rounds

The final classifier is a weighted majority of all weak classifiers where the weights are proportional α_i

AdaBoost

$y_i \in \{-1, +1\}$, $y_i h(x_i)$

input:

training set $S = (\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_m, y_m)$

weak learner WL

number of rounds T

initialize $\mathbf{D}^{(1)} = \left(\frac{1}{m}, \ldots, \frac{1}{m}\right)$.

for $t = 1, \ldots, T$:

invoke weak learner $h_t = \mathrm{WL}(\mathbf{D}^{(t)}, S)$

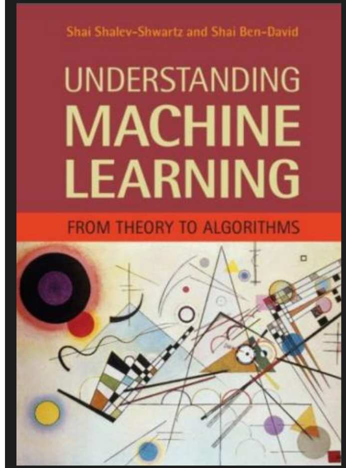
compute $\epsilon_t = \sum_{i=1}^{m} D_i^{(t)} \mathbb{1}_{[y_i \neq h_t(\mathbf{x}_i)]}$

let $w_t = \frac{1}{2} \log\left(\frac{1}{\epsilon_t} - 1\right)$

update $D_i^{(t+1)} = \dfrac{D_i^{(t)} \exp(-w_t y_i h_t(\mathbf{x}_i))}{\sum_{j=1}^{m} D_j^{(t)} \exp(-w_t y_j h_t(\mathbf{x}_j))}$ for all $i = 1, \ldots, m$

output the hypothesis $h_s(\mathbf{x}) = \mathrm{sign}\left(\sum_{t=1}^{T} w_t h_t(\mathbf{x})\right)$.

$$\mathbb{1}_{[y_i \neq h_t(x_i)]} = \begin{cases} 1 & y_i \neq \cdots \\ 0 & y = \cdots \end{cases}$$



BOOSTING THEORY

- IF ALL THE INTERMEDIATE WEAK LEARNERS ARE BETTER THAN RANDOM (E.G., ERROR <49% FOR BINARY CLASSIFICATION)
  - THEN THE TRAINING ERROR OF THE COMBINED MODEL CONVERGES QUICKLY TO 0!

0.01 = γ

THEOREM 10.2 *Let S be a training set and assume that at each iteration of AdaBoost, the weak learner returns a hypothesis for which $\epsilon_t \leq 1/2 - \gamma$. Then, the training error of the output hypothesis of AdaBoost is at most*

$$L_S(h_s) = \frac{1}{m}\sum_{i=1}^{m} \mathbb{1}_{[h_s(\mathbf{x}_i) \neq y_i]} \leq \exp(-2\,\gamma^2 T) .$$

BOOSTING THEORY

- So the goal is not reducing the variance anymore
- The goal is reducing the bias!
- What about the test error?