Boosting

April 12, 2018

Boosting

(train performance, morale, grades, social skills, ...)

Are more heads better than 1?

A) Yes, if they’re my friends
B) Yes, if they’re not my friends
C) A and B
D) No, not ever
E) Define ‘better’

Idea: Let’s get a bunch of hypotheses and have them vote.

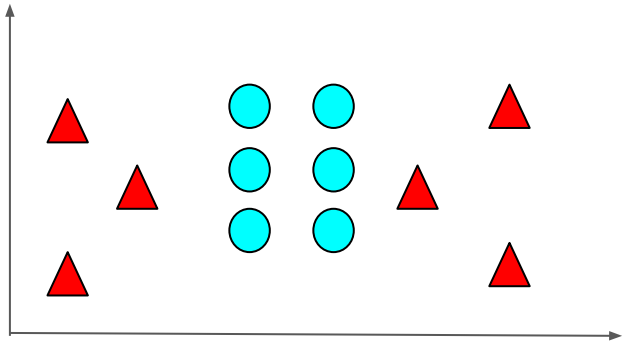
[Note that a bunch of hypotheses, even from different classes, is called an ‘ensemble’]

Why might we do this?

- Diversity - Spread out the wrong answers
- Task splitting - Focus on different aspects of the problem
- Non-Realizability - No one hypothesis can solve the whole problem

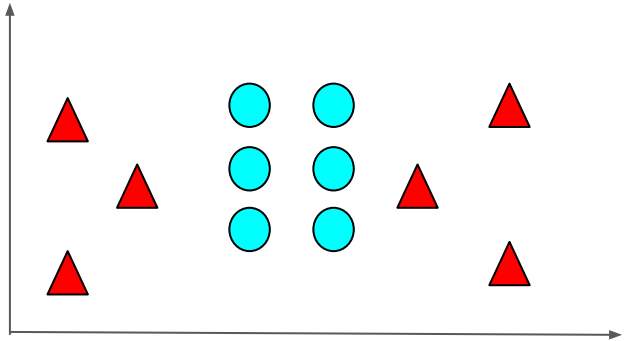
Decision Stumps: Is this realizable?

A) Yes.
B) No.
C) What's 'realizable'?
D) This is dumb.
E) Why isn't Michael here?



Boosted Decision Stumps: Is this realizable?

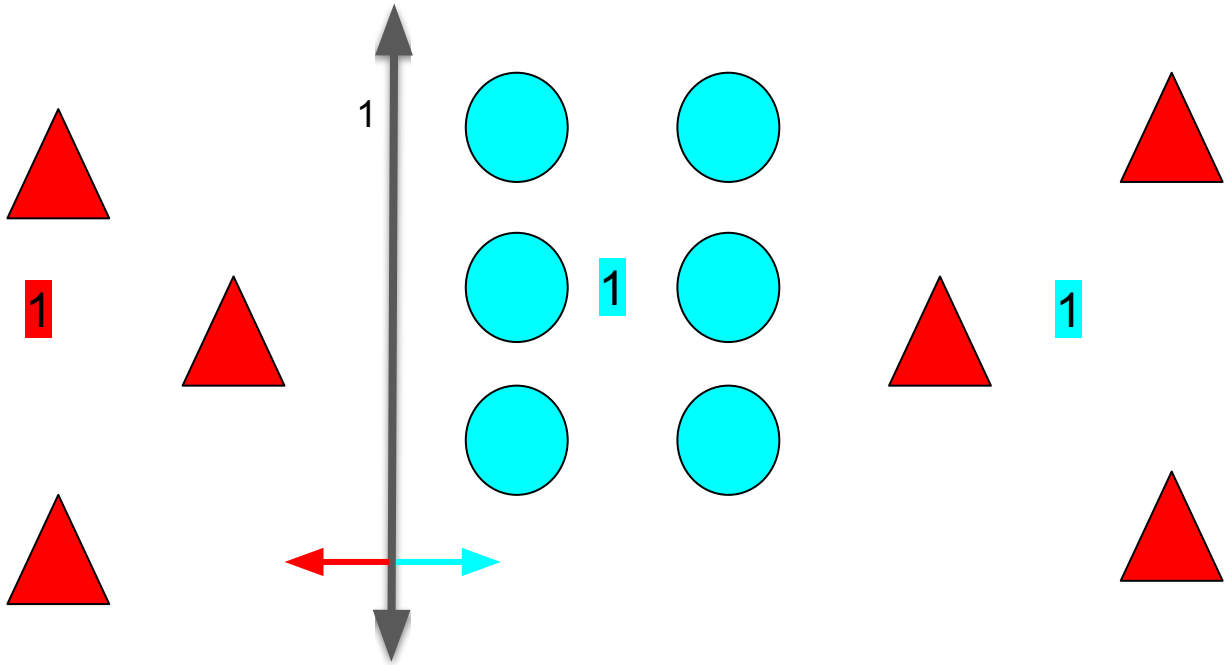
A) Obviously.
B) Yes, but it's not obvious.
C) I don't know.
D) Definitely not.
E) But why isn't Michael here?



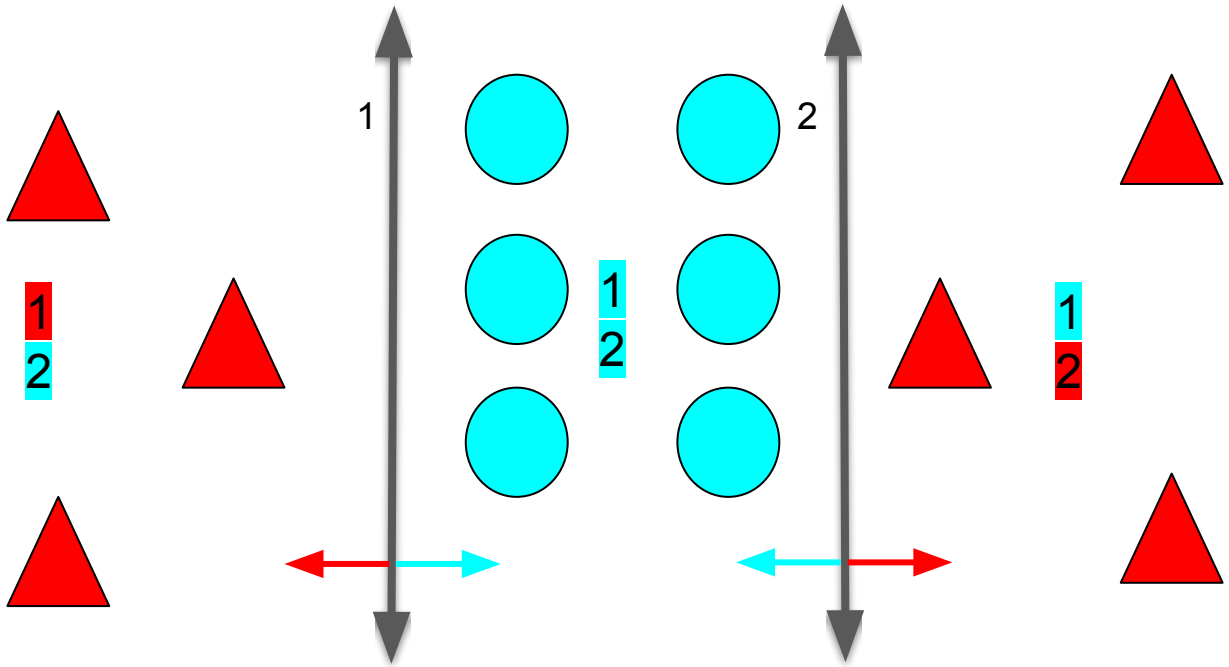
The Answer: Obviously

Why would (S)am introduce an idea and then immediately show an example where it doesn’t work?

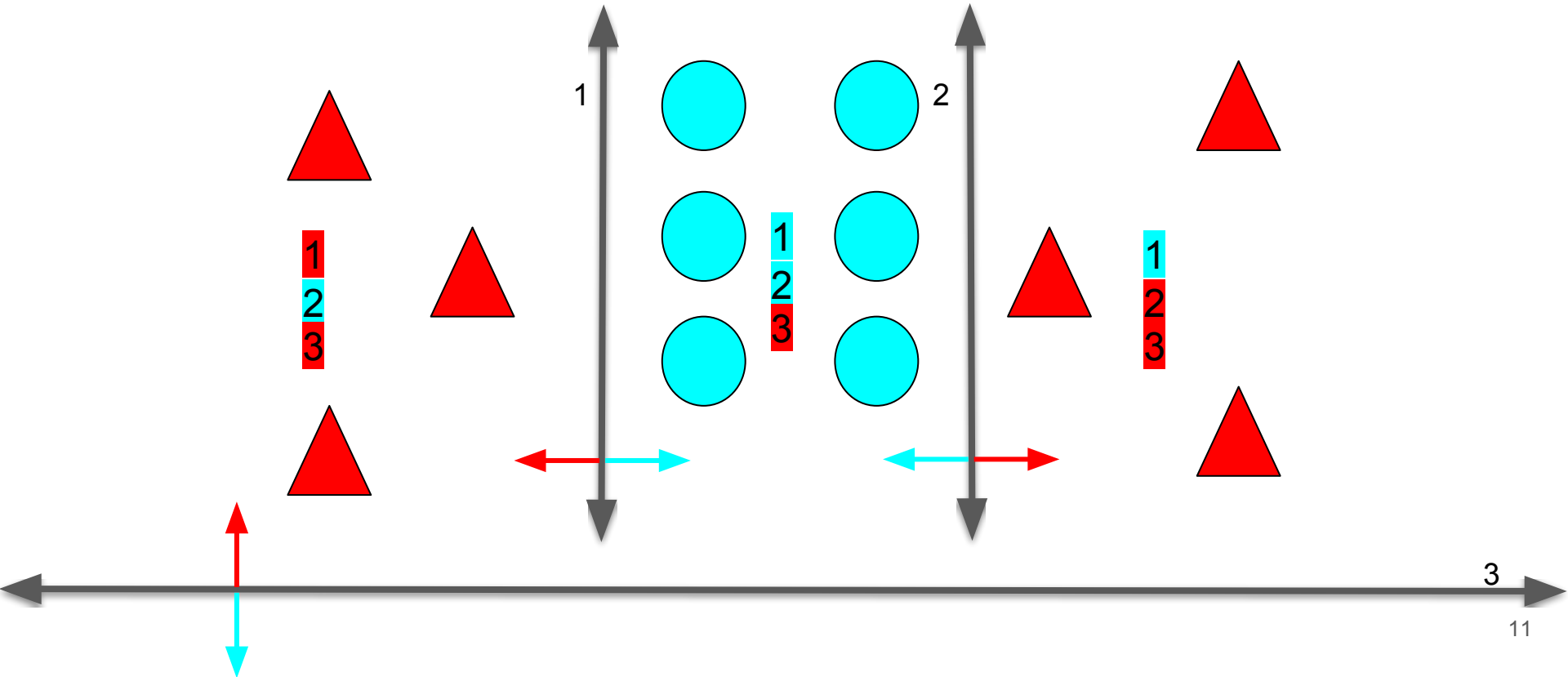
The Answer: Yes, but it’s not obvious.



The Answer: Yes, but it’s not obvious.

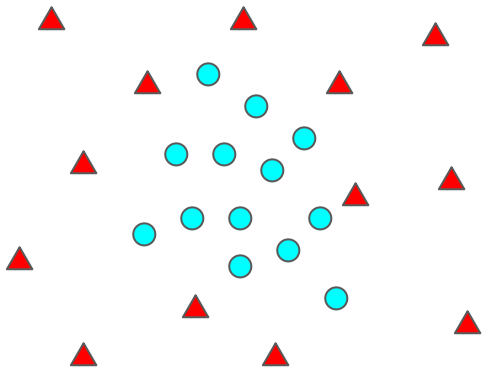


The Answer: Yes, but it’s not obvious.



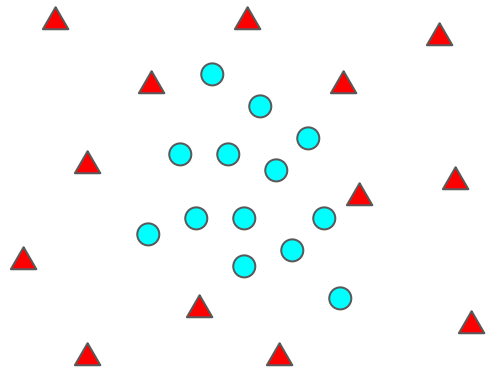
Linear Separators: Is this realizable?

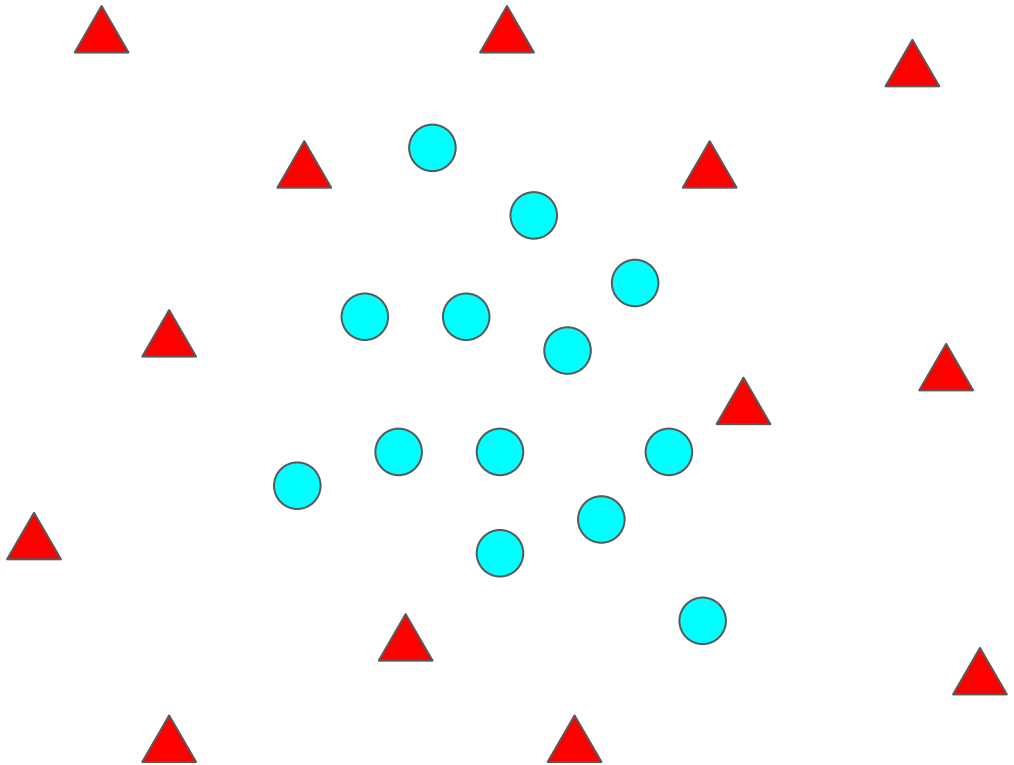
A) Yes.
B) No.
C) What's 'is'?
D) This is still dumb.
E) Why is (S)am here?

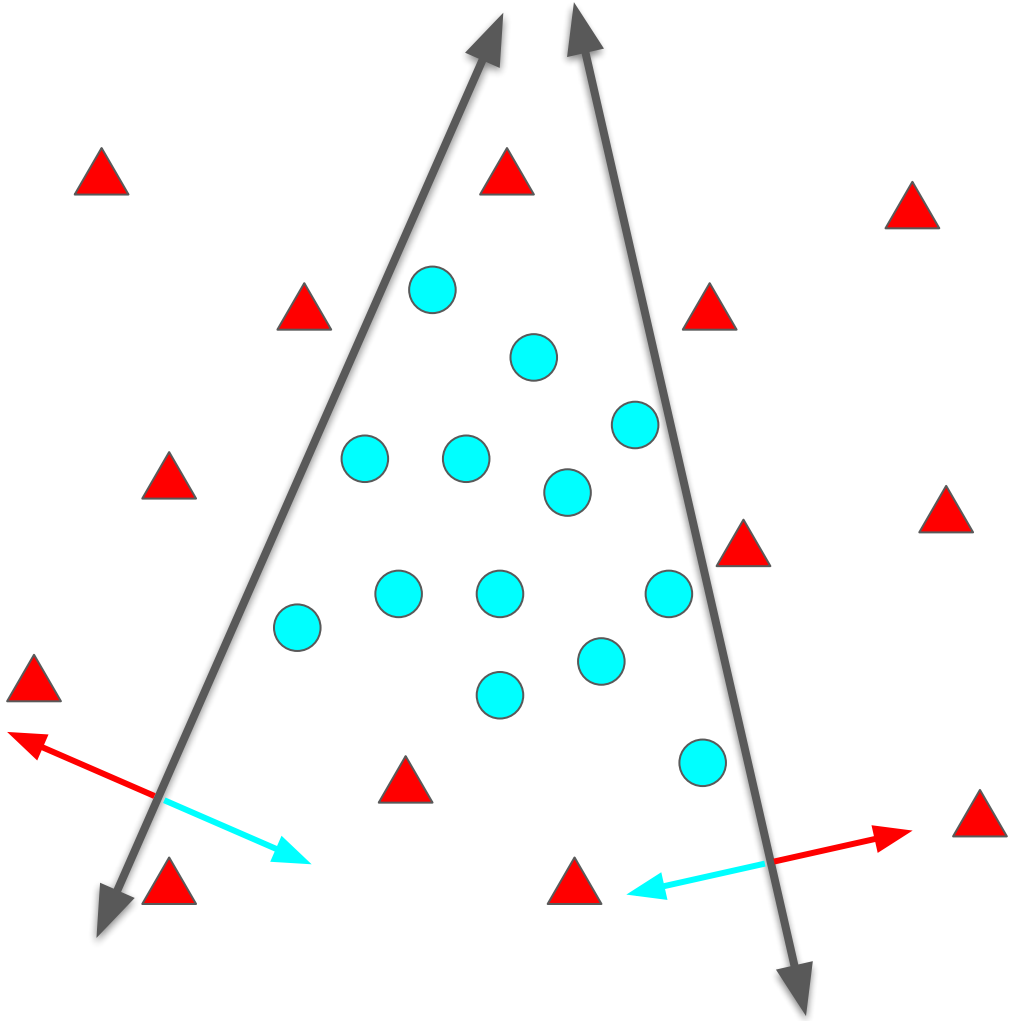


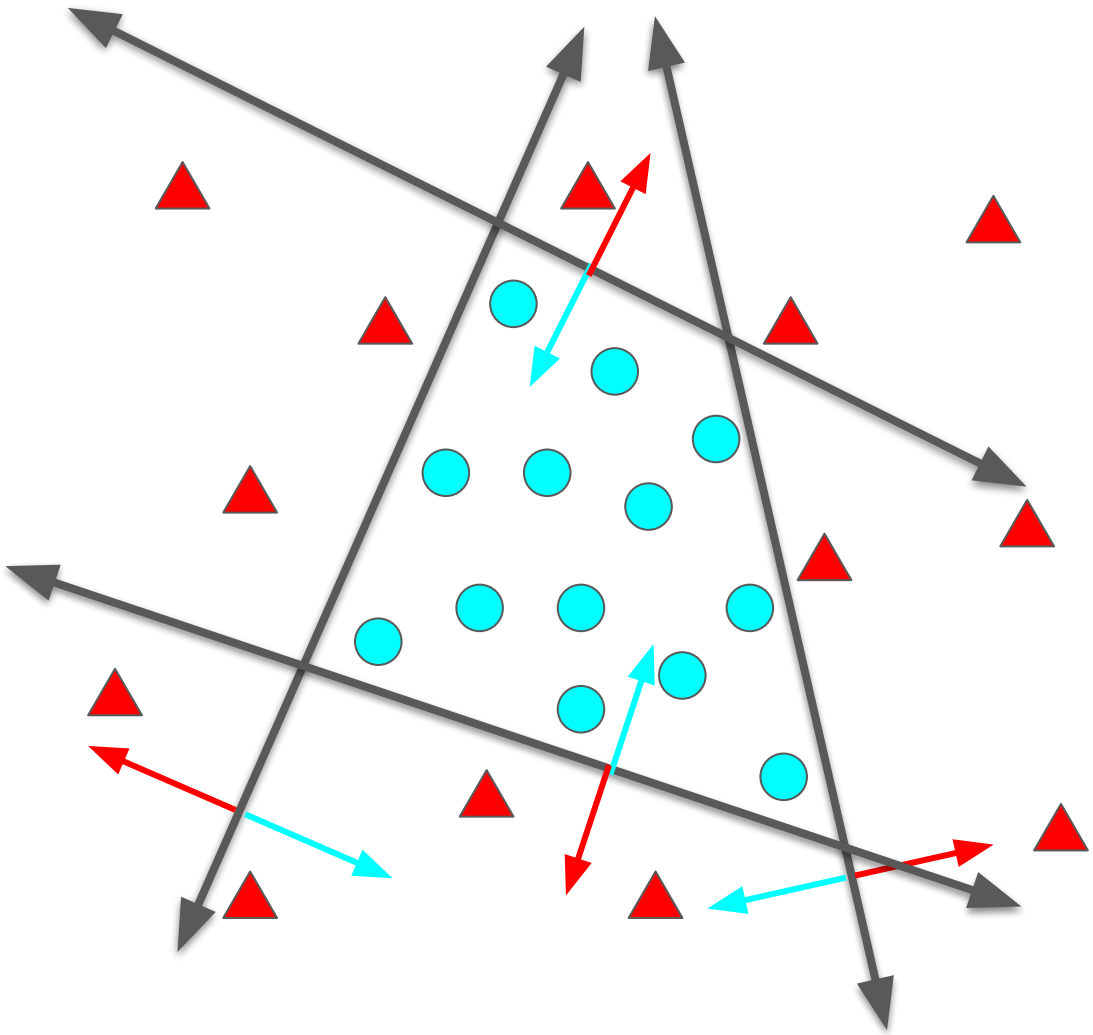
How many lines for perfect Boosted Linear Separation?

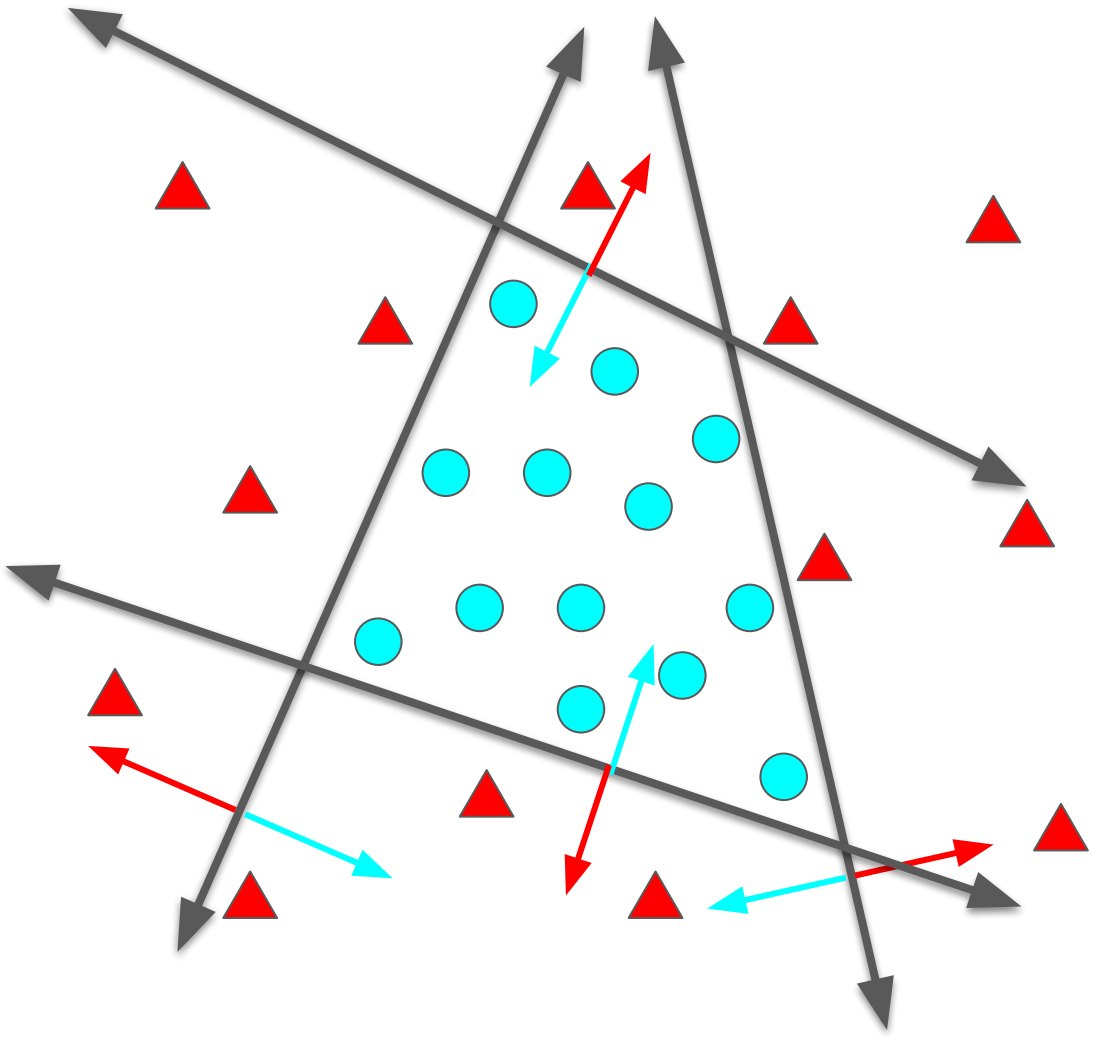
A) 3
B) 4
C) 5
D) 6
E) 7

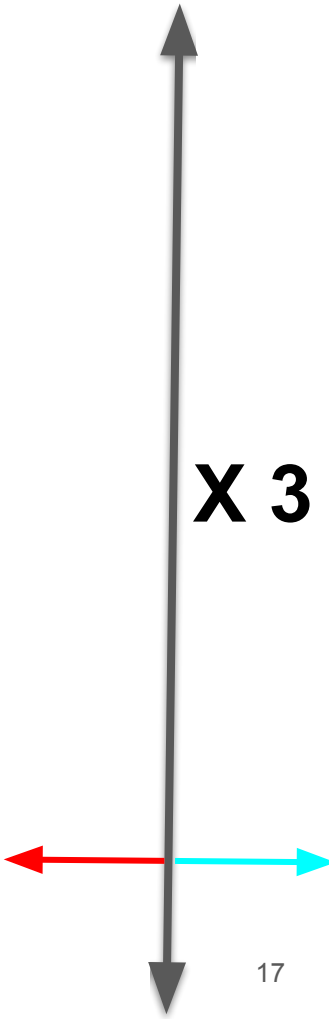











X 3


BOOSTING
A MACHINE LEARNING STORY

The Model

Let H be the class of not-boosted hypotheses.

Let H_E be the class of ensemble hypotheses built using elements of H.

$H_E(x) = \{\sum_i w_i h_i(x) \mid h_i \in H, w_i \in \mathbb{R}\}$

The Loss

$L_E(H_E) = L(H_E)$

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

But we can redefine the loss or training examples for H in the “inner loop” of boosting.

The “Optimization”

AdaBoost

input:

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

weak learner WL

number of rounds T

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

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

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

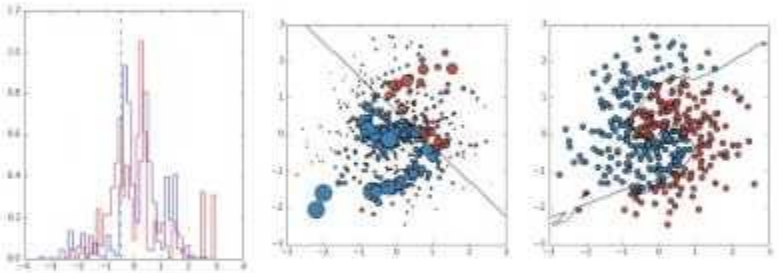
compute $\epsilon_t = \sum_{i=1}^{m} D_i^{(t)} \mathbb{1}_{[y_i \neq h_t(\mathbf{x}_i)]}$

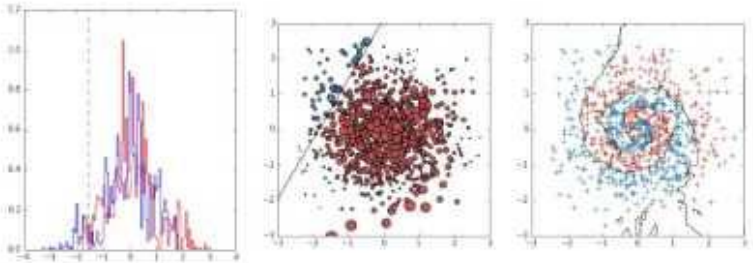
let $w_t = \frac{1}{2} \log\left(\frac{1}{\epsilon_t} - 1\right)$

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

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

Some Demonstrations:





AdaBoost (Adaptive Boosting)

- Freund, Yoav, and Robert E. Schapire. "A decision-theoretic generalization of on-line learning and an application to boosting." Journal of computer and system sciences 55.1 (1997): 119-139.
- 15 000 + citations
- won the 2003 Gödel Prize
  - (For Theoretical Computer Science)
- turns decision stumps into one of the best “out of the box” classifiers
  - Arguably more general than SVM with RBF kernel - harder to over fit.
- Generalized by “Multiplicative Weights Update” algorithm - cool theory

Why does it work? Intuition

1. Better hypotheses get more weight.
2. Hard training examples get more attention.
3. If at first you don't succeed…
   … use a weak PAC learner until you do!
   (Assume distribution-independent bounded error is a powerful assumption.)

$$\epsilon_{t+1} \leq \frac{1}{2} - \gamma$$

Why does it work? Proof

1. We will place an upper bound (called Z) on the loss L_E.
   (This loss will be related to the sum of weights across all training examples.)
2. We will show that at each time step, Z shrinks by a multiplicative factor.
3. Therefore, the empirical training loss shrinks exponentially with T.

Why does it work? Proof

$$L_S(h_s) = \frac{1}{m} \sum_{i=1}^{m} \mathbb{1}_{[h_s(\mathbf{x}_i) \neq y_i]}$$

For each t, denote $f_t = \sum_{p \leq t} w_p h_p$

$$Z_t = \frac{1}{m} \sum_{i=1}^{m} e^{-y_i f_t(x_i)}$$

Why does it work? Proof

True or False: $\mathbb{1}_{[h(x) \neq y]} \leq e^{-yh(x)}$

a) True
b) False
c) It depends on h and y
d) Um...
e) Why am *I* here?

Why does it work? Proof

Z is an upper bound on our loss. Specifically:

$$Z_T \geq L_S(f_T)$$

$$Z_t = \frac{1}{m} \sum_{i=1}^{m} e^{-y_i f_t(x_i)}$$

Why does it work? Proof

Now we turn Z into a product across time steps:

$$Z_T = \frac{Z_T}{Z_0} = \frac{Z_T}{Z_{T-1}} \cdot \frac{Z_{T-1}}{Z_{T-2}} \cdots \frac{Z_2}{Z_1} \cdot \frac{Z_1}{Z_0}$$

Note that we define Z_0 to be 1 because we define f_0 to always return 0.

This is guaranteed to be wrong, but still satisfies the condition $Z_T \geq L_S(f_T)$

Why does it work? Proof

$$D_i^{(t+1)} = \frac{e^{-y_i f_t(x_i)}}{\sum_{j=1}^m e^{-y_j f_t(x_j)}}.$$

Hence,

$$\begin{aligned}
\frac{Z_{t+1}}{Z_t} &= \frac{\sum_{i=1}^m e^{-y_i f_{t+1}(x_i)}}{\sum_{j=1}^m e^{-y_j f_t(x_j)}} \\
&= \frac{\sum_{i=1}^m e^{-y_i f_t(x_i)} e^{-y_i w_{t+1} h_{t+1}(x_i)}}{\sum_{j=1}^m e^{-y_j f_t(x_j)}} \\
&= \sum_{i=1}^m D_i^{(t+1)} e^{-y_i w_{t+1} h_{t+1}(x_i)} \\
&= e^{-w_{t+1}} \sum_{i:y_i h_{t+1}(x_i)=1} D_i^{(t+1)} + e^{w_{t+1}} \sum_{i:y_i h_{t+1}(x_i)=-1} D_i^{(t+1)} \\
&= e^{-w_{t+1}}(1-\epsilon_{t+1}) + e^{w_{t+1}} \epsilon_{t+1} \\
&= \frac{1}{\sqrt{1/\epsilon_{t+1}-1}}(1-\epsilon_{t+1}) + \sqrt{1/\epsilon_{t+1}-1}\,\epsilon_{t+1} \\
&= \sqrt{\frac{\epsilon_{t+1}}{1-\epsilon_{t+1}}}(1-\epsilon_{t+1}) + \sqrt{\frac{1-\epsilon_{t+1}}{\epsilon_{t+1}}}\epsilon_{t+1} \\
&= 2\sqrt{\epsilon_{t+1}(1-\epsilon_{t+1})}.
\end{aligned}$$

Remember:

$$f_t = \sum_{p \le t} w_p h_p$$

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

Why does it work? Proof

Every time step, the training loss goes down by a multiplicative factor of:

$$2\sqrt{\epsilon_{t+1}(1-\epsilon_{t+1})} \leq 2\sqrt{\left(\frac{1}{2}-\gamma\right)\left(\frac{1}{2}+\gamma\right)} = \sqrt{1-4\gamma^2}$$

Recall that $0 < \gamma < 1/2$

Why does boosting work?

a) Magic
b) Weak PAC-Learners are actually Strong
c) Subproblems are usually easier than the whole thing
d) To earn a paycheck
e) It doesn’t

Tradeoffs

1. |H| - Hypothesis complexity
2. Number of Iterations


BOOSTING
THINGS

Gradient Boosting

1. Suppose we use a decision stump for regression.
2. Decision stumps can be added together to make a nonlinear function.
3. This looks a little bit like boosting…

$$F_{m+1}(x) = F_m(x) + h(x)$$

We want:

$$h(x) = y - F_m(x)$$

4. Training a hypothesis on the “residual” is called gradient boosting.

ResNets

He, Kaiming, et al. "Deep residual learning for image recognition." Proceedings of the IEEE conference on computer vision and pattern recognition. 2016.

- 7500+ citations since 2016
- Use a *very* deep network (~1000 layers) where intermediate layers are trying to predict the residual on the previous layers.
- Similar in spirit to gradient boosting.

Heterogeneous Ensembles

- Suppose $H^* = \min_L H_1, H_2$
- Or suppose we want to combine several experts...

Boosting is a Function

AdaBoost: Weak ML alg's → Sweet ML alg's


BOOST
ING
(S)am

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

$$D_i^{(t+1)} = \frac{e^{-y_i f_t(x_i)}}{\sum_{j=1}^{m} e^{-y_j f_t(x_j)}}$$


BOOST
ING
(S)am

Verse 1

Let me tell you about boosting, an algorithm so sweet
It deserved to be included in this lecture to a beat.
Suppose your models are weak, and you hypothesize
Zero training loss just can’t be realized.
Boosting can improve it by using an ensemble
Trained by reweighting the training examples.
The model gets slower, but is slow to **o**ver**-**fit
So if your loss is too high, use boosting to lower it.

Chorus

Boosting

Keeps moving loss lower

But the

Inference gets slower

Verse 2

If you want to boost adaptively, then here's what you do:
inverse error, minus one, take the log, over 2.
This weight's for the hypothesis, when they all go to vote,
It's also used to choose which training data to promote.
To update, exponentiate by the weight times the sign
Of the hypothesis' prediction times the true label, y.
This weight redistributes the training samples
Renormalize for next time's training examples.

Chorus

Boosting

Keeps moving loss lower

But the

Inference gets slower

Verse 3

You’ve got a regression, but it keeps working wrong
Perhaps because your model class just isn’t strong
Use nonlinear hypotheses to learn the residual
Watch the loss descend, let me give you this visual
Gradient Boosting makes the loss roll down hill
It only takes gravity, it doesn’t take skill.
When your best isn’t enough, boosting’s your friend.
I hope you liked this rap, cause this is the end.