Summary

→ SGD

→ SGD analysis

→ Extensions of GD

Today

→ Online Learning

→ Mistake bound / Perceptron

→ Regret Minimization

Assignment 1 due today @ 10 PM.

Summary of GD

→ "Universal" algorithm

→ Monotonic decrease

→ Solves Convex Optimization

Summary of NAGD

→ Use momentum to speed-up convergence

Summary of SGD

→ Need only an estimator for gradient

→ The algorithm in practice

→ Improves speed per iteration

→ "Slower" convergence but converges for Convex.

Extensions:

→ Can use Subgradients

→ Constrained optimization: Need projection oracle.

Some important topics we did not cover

→ Adaptive step-sizes: Critical in practice.

→ "Different" norms for steepest distance.

→ Changing the landscape / Non-Convex optimization.

→ Mirror Descent (Change variable space & then do GD?).

Online Learning

→ You do not see all of the data at once.

Example:
- → Online advertising
- → Weather prediction
- → Boarding information / Resource allocation.
- → Poker / Games ...
- → Stock market predictions

How to formalize Online Learning?

Idea 1: Mistake bounded model

Idea 2: Regret Minimization.

MISTAKE BOUNDED MODEL

Day 1: $(X_1, \hat{y}_1, y_1) \rightarrow$ Is $y_1 = \hat{y}_1$?

($\hat{y}_1$ → your prediction; y_1 → ✓ / ✗)

$\in \{0,1\}$ true label for the day $\in \{0,1\}$

Day 2: $(x_2, \hat{y}_2, y_2)$ — … Is $y_2 = \hat{y}_2$

⋮

Day

⋮

You are only allowed to make a bounded number of mistakes.

Assume: The true labels are generated using some function f_* from a hypothesis class H.

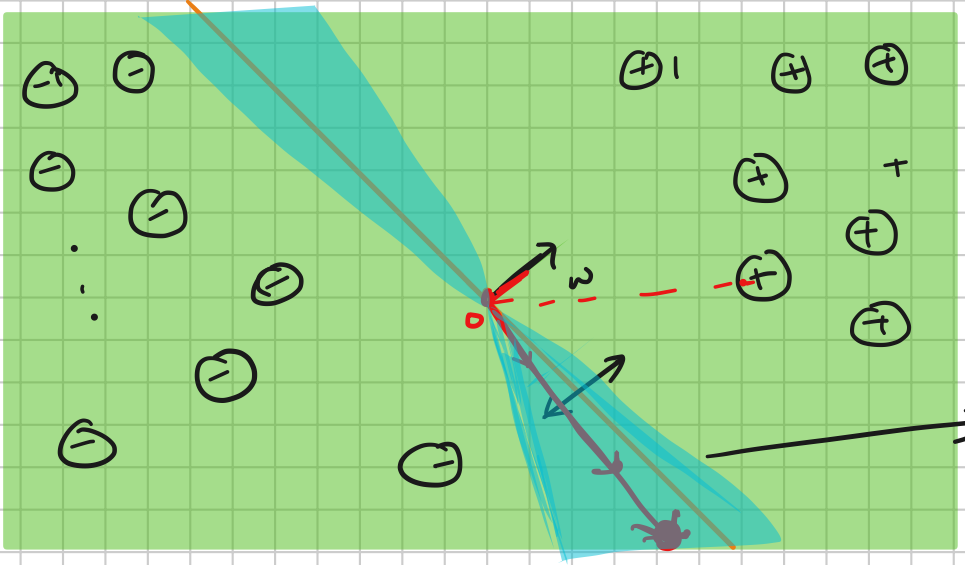
$$y_i = f_*(x_i)$$

Online Learning of HalfSpaces

$$f_*(x) = \text{Sign}(\langle w_*, x \rangle) \quad \text{for some } w_*.$$

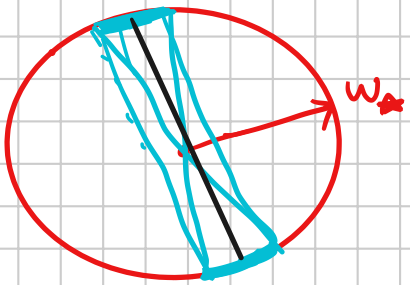
Domain $X \in \mathbb{R}^d$. $\mathcal{Y} \equiv \{1, -1\}$

$$h(x) = \begin{cases} 1 & \text{if } \langle w, x \rangle > 0 \\ -1 & \text{if } \langle w, x \rangle < 0 \end{cases}$$



Margin of halfspace $\gamma = \min_{x \in D} \frac{|\langle w_*, x \rangle|}{\|w_*\| \cdot \|x\|_2}$

Assume that $\|w_*\| = 1$ and let us normalize all examples $\|x\| = 1$.

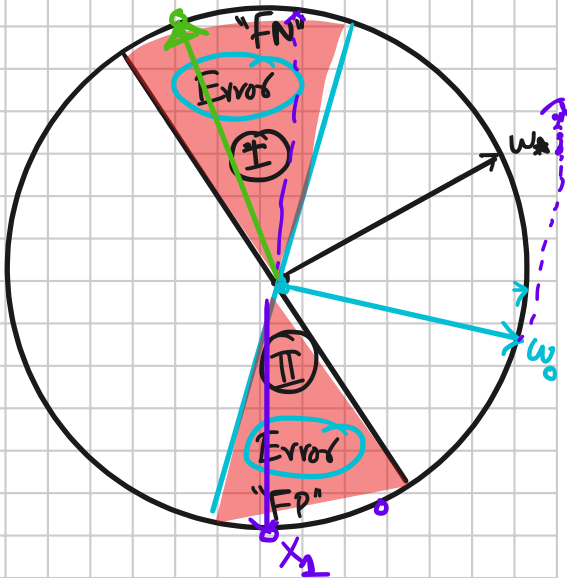


PERCEPTRON (1958, Rosenblatt)

→ $w_0 \equiv$ random vector

→ On Day i:

$$w_i = \begin{cases} w_{i-1} & \text{if no mistake} \\ w_{i-1} + y_i \cdot x_i & \text{if mistake} \end{cases}$$



Day 1: 1. Start with random w_0. Predict $\hat{y}_1 = \text{sign}(\langle w_0, x \rangle)$.

$y_i \equiv$ true label.

If correct: Do nothing.

If wrong: If true label is "−", but we say "+".

$\rightarrow w_1 = w_0 - x_1$.

If true label is "+", but we say "−".

$\rightarrow w_1 = w_0 + x_1$.

Thm: Perceptron makes at most $\frac{1}{\gamma^2} + 1$ mistakes.
($\gamma \equiv$ margin of w_*).

Proof: When mistake $w_i = w_{i-1} + y_i \cdot x_i$.

$$\langle w_i, w_* \rangle = \langle w_{i-1}, w_* \rangle + y_i \langle x_i, w_* \rangle$$

$$= \langle w_{i-1}, w_* \rangle + |\langle w_*, x_i \rangle|$$

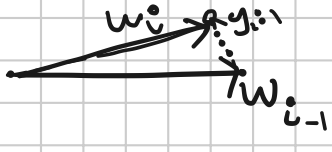
$$\geq \langle w_{i-1}, w_* \rangle + \gamma$$

(because we have margin γ!).

Claim: $\|w_i\|^2 \leq \|w_{i-1}\|^2 + 1$. (for each iteration)

Proof: $w_i = w_{i-1} + y_i \cdot x_i$

(i is mistake)



$$\|w_i\|^2 = \|w_{i-1}\|^2 + y_i^2 \cdot \|x_i\|^2 + 2 \langle w_{i-1}, y_i \cdot x_i \rangle$$

$$= \|w_{i-1}\|^2 + 1 + \underbrace{2y_i \cdot \langle w_{i-1}, x_i \rangle}_{\text{is } \leq 0 \text{ because we made a mistake}}$$

$$\leq \|w_{i-1}\|^2 + 1.$$

So we have whenever we make a mistake

$$\langle w_{i-1}, w_* \rangle + \gamma \leq \langle w_i, w_* \rangle \quad \text{and} \quad \|w_i\|^2 \leq \|w_{i-1}\|^2 + 1.$$

At any point in time

$$\|w_n\|^2 \leq 1 + M_n$$

M_n = # mistakes made until then.

$$\langle w_0, w_* \rangle + \gamma \cdot M_n \leq \langle w_n, w_* \rangle \leq \|w_n\| \cdot \|w_*\|$$

$$= \|w_n\| \leq \sqrt{1 + M_n}.$$

$$(\langle u, v \rangle \leq \|u\| \cdot \|v\|)$$

$$\gamma \cdot M_n \leq \sqrt{1 + M_n} - \langle w_0, w_* \rangle$$

$$\boxed{\gamma \cdot M_n \leq \sqrt{1 + M_n} + 1.}$$

$$\Downarrow$$

$$M_n \leq \frac{1}{\gamma^2} + 1.$$

Perceptron as SGD with "Hinge Loss"

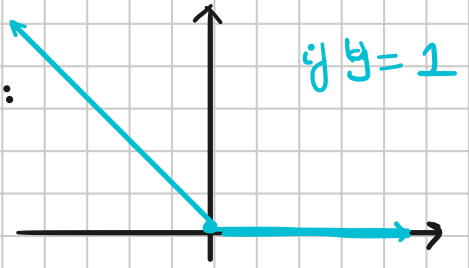
SGD : $w_i = w_{i-1} - \eta \cdot \underbrace{G_t(w_{i-1})}_{\text{estimator for gradient.}}$

SGD for ERM: $w_i = w_{i-1} - \eta \nabla_w \left(\ell(h_w(x_i), y_i) \right)$.

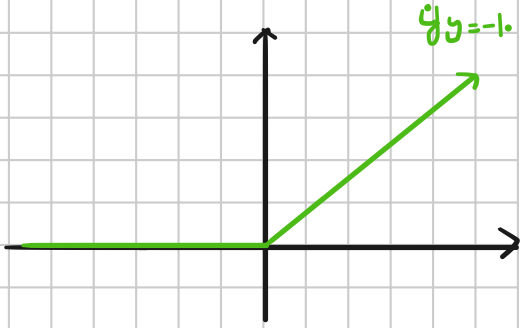
$(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)$

$$L(w) = \frac{1}{n} \sum_{i=1}^{n} \ell(h_w(x_i), y_i) .$$

$h_w(x) = \langle w, x \rangle$ · Loss function:



$$\nabla_w \ell(h_w(x_i), y_i) = \begin{cases} 0 & \text{if } \operatorname{sign}(\langle w, x_i \rangle) = y_i \\ -y_i \cdot x_i & \text{if not} \end{cases} .$$



"SGD" can be used for online learning problems

Learning with Experts

- "Predict" based on prediction of experts.

	E_1	E_2			E_d	Our prediction	Truth
Day 1:	Up	Down	Up	Down	Down	Down	Up
Day 2:	Down	Up	Down	Up	Up	Up	Down

⋮

	"Loss" of E_1	"Loss of E_2"	. .	"Loss of E_d"	Our loss

$L(i,t)$ = Loss of expert i on day t. ($\begin{cases} 1 & \text{if wrong} \\ 0 \end{cases}$)

$L(t)$ = our loss on day t.

Goal: Do as well as the best expert in hindsight

$$\text{Regret}(n) = \sum_{t=1}^{n} L(t) - \min_{i=1,\dots,d} \sum_{t=1}^{n} L(i,t)$$

LECTURE 7

Last class
- Online Learning
- Perceptron
- Regret Minimization

This class
- How to minimize regret?
- Multiplicative weights.

Assignment 1 will be graded and solutions posted soon (Waiting for reader confirmation ...).

Learning with Experts

Day	E_1	E_2	...	E_d	Our
1	0	1	1 ...		1
2			...		
3					
4					
⋮					

$L(t,i)$ = Loss of expert i on day t.

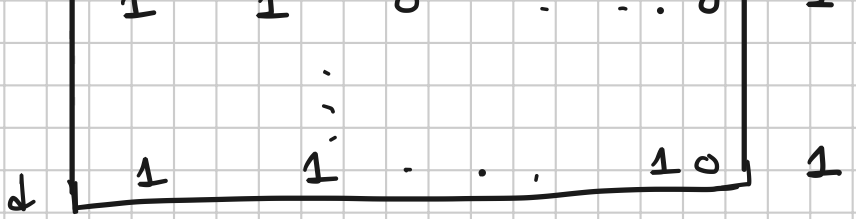
$L(t)$ = Loss of our algorithm.

$$\text{Regret}(T) = \sum_{t=1}^{T} L(t) - \min_i \underbrace{\sum_{t=1}^{T} L(t,i)}_{\text{Loss of best expert.}}$$

Strategy 1: Follow the Leader (FTL)

Day	E_1	E_2	E_3	...	E_d	Our algorithm
1	Up	Down	Down		Down	Up ← Prediction
	1	0	0		0	1 ← Losses
2	Up	Up	Down	...	Down	UP
	1	1	0		0	1

After d days



Regret(d) = d !

Can generalize this so that

$L_*(T) \equiv$ Loss of best expert $= \frac{T}{d}$

$L(T) \equiv T.$

"Toy Case": Suppose there is infact an infallible expert.

(ie; $\exists$ some $i^* \in \{1, .., d\}$ such that $L(t, i^*) = 0$ $\forall t$).

Q: What regret can we achieve now?

Suggestion 1:
- Keep a set of "eligible experts"
- Whenever someone makes a mistake, throw them out.

Follow the Majority (FTM)

Algorithm
→ $\mathcal{E}[0] = \{1, 2, .., d\}$
→ On each day $t = 1, .., d$:
→ Predict according to the majority prediction of experts in $\mathcal{E}[t-1]$.
→ $\mathcal{E}[t] = \mathcal{E}[t-1] \setminus$ experts who made mistake on day t.

Thm: Regret $(T) \leq \log_2 d$.

Proof: **Idea:** Whenever we make a mistake, the set of eligible experts shrinks by atleast a factor of 2.

$W(t)$ = # Eligible experts
$= |\mathcal{E}[t]|$.

(a) $W(0) = d$

(b) Every mistake, $W(t) \leq \frac{W(t-1)}{2}$.

$L(t)$ = # mistakes our algorithm makes.

$\Rightarrow 1 \leq W(t) \leq W(0) \cdot \left(\frac{1}{2}\right)^{L(t)}$.

$\Rightarrow 1 \leq d \cdot \left(\frac{1}{2}\right)^{L(t)}$

$\Rightarrow 2^{L(t)} \leq d$

$\Rightarrow L(t) \leq \log_2 d$.

Assumption of infallible expert is very strong!

In general:
→ $w(t,i)$ for expert i after round t

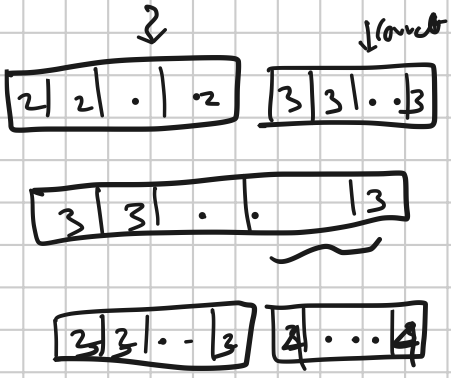
→ We will make prediction based on "weighted majority"

→ How do we update weights?
→ Every mistake halves your weight!

Weighted Majority Algorithm (WMA)

→ $w(0,i) = 1 \quad \forall \; i = 1, \dots, d$

→ On each day $t = 1, \dots, T$:
→ Predict according to weighted majority with weights $w(t-1,i)$
→ For each expert i:
→ If mistake, then $w(t,i) = \frac{w(t-1,i)}{2}$.



If Sum of weights of experts predicting UP > Sum of weights of experts predicting Down
→ Then predict UP.

Thm: $L(T) \leq (2.4)\left(L_*(T) + \log_2 d\right)$.

$L(T)$: Our loss
$L_*(T)$: Loss of best expert.

Proof: $W(t) = \sum_{i=1}^{d} w(t,i) =$ Total weight (of all experts).

Claim: $W(t) \leq W(t-1)$.

Claim: If we make a mistake on day t,

$$W(t) \leq \left(\frac{3}{4}\right) W(t-1).$$

Proof:

$$W(t) = \sum_{i=1}^{d} w(t,i)$$

$$= \sum_{i:\ \text{expert } i \text{ is correct}} w(t,i) + \sum_{i:\ \text{expert } i \text{ was wrong}} w(t,i)$$

$$= \sum_{i:\ \text{expert } i \text{ is correct}} w(t-1,i) + \frac{1}{2} \sum_{i:\ \text{expert } i \text{ was wrong.}} w(t-1,i)$$

$$\geq \frac{W(t-1)}{2}$$

$$= \sum_{i:\ \text{expert } i \text{ is correct}} w(t-1,i) + \sum_{i:\ \text{expert } i \text{ was wrong}} w(t-1,i) - \frac{1}{2} \sum_{i:\ \text{expert } i \text{ was wrong}} w(t-1,i)$$

$$\leq W(t-1) - \frac{1}{2} \cdot \frac{W(t-1)}{2} = \frac{3}{4} W(t-1).$$

If $L(t)$ = our loss after t rounds, then

$$W(t) \leq W(0) \cdot \left(\frac{3}{4}\right)^{L(t)}.$$

$$= d \cdot \left(\frac{3}{4}\right)^{L(t)}.$$

Claim: $\left(\frac{1}{2}\right)^{L_*(t)} \leq W(t)$.

$L_*(t)$ = # mistakes of best expert.

(Prog: $w(t,i) = \left(\frac{1}{2}\right)^{\sum_{j \leq t} L(j,i)}$.)

To continue ...

$$\left(\frac{1}{2}\right)^{L_*(t)} \leq W(t) \leq d \cdot \left(\frac{3}{4}\right)^{L(t)}$$

$$\Rightarrow \left(\frac{4}{3}\right)^{L(t)} \leq 2^{L_*(t)} \cdot d$$

$$\Rightarrow L(t) \cdot \left(\log_2 \left(\frac{4}{3}\right)\right) \leq L_*(t) + \log_2 d$$

$$\Rightarrow L(t) \leq \left(\frac{1}{\log_2\left(\frac{4}{3}\right)}\right)\left(L_*(t) + \log_2 d\right)$$

$$\approx 2.4 \left(L_*(t) + \log_2 d\right).$$

$$\text{Regret}(T) = L(t) - L_*(T)$$

$$\leq 1.4\, L_*(T) + (2.4) \log_2 d.$$

Algorithms with $\frac{\text{Regret}(T)}{T} \to 0$ (as $T \to \infty$) are called "No-Regret Algorithms".

Remark: WMA is not a "No-Regret" Algorithm!

Claim: There is no deterministic "No-Regret" algorithm.
There is no deterministic algorithm that can do better than a factor of 2!

Theorem: Multiplicative Weights Method (MWM) satisfies $\mathbb{E}[L(T)] \leq L_*(T) + 2 \cdot \sqrt{T \ln d}$.

$$\Rightarrow \text{Regret}(T) \leq 2 \cdot \sqrt{T \ln d}.$$

$$\Rightarrow \frac{\text{Regret}(T)}{T} \leq 2 \cdot \sqrt{\frac{\ln d}{T}}.$$

Multiplicative Weights Update Method

① $w(0,i) = 1$, $i = 1, \dots, d$

② On each day $t = 1, \dots, T$:

→ • Pick an expert i with probability $\propto w(t-1,i)$

$$\Pr[\text{expert } i \text{ is picked}] = \frac{w(t-1,i)}{\sum_{i=1}^{d} w(t-1,i)}.$$

• Then follow expert i.

→ Update weights of every expert:

→ If correct, $w(t,i) = w(t-1,i)$

→ If wrong, $w(t,j) = (1-\varepsilon)\, w(t-1,j)$

↓ Some parameter.

Theorem: $\mathbb{E}[L(T)] \leq (1+\varepsilon) L_*(T) + \frac{\ln d}{\varepsilon}$.

$(\varepsilon < \frac{1}{2})$.

Corollary: Set $\varepsilon = \sqrt{\frac{\ln d}{T}}$. Then,

$$\mathbb{E}[L(T)] \leq L_*(T) + 2 \cdot \sqrt{T \ln d}.$$

Proof of Corollary: $\mathbb{E}[L(T)] \leq L_*(T) + \varepsilon \cdot L_*(T) + \frac{\ln d}{\varepsilon}$

$$= L_*(T) + \sqrt{\frac{\ln d}{T}} \cdot L_*(T) + \sqrt{T \ln d}$$

$$\leq L_*(T) + 2 \cdot \sqrt{T \ln d}.$$

(as $L_*(T) \leq T$ always).

LECTURE 8

Last class
- → Weighted Majority Algorithm
- → Multiplicative Weights Method

Today
- → MWM achieves no-regret!
- → Application to boosting.

- Assignment 2 will be posted today at 6PM.
- Due 04/27/22 @ 9:59 PM.
- Assignment 1 solutions are now up on Edstem.

Multiplicative Weights Update Method

① $w(0,i) = 1$, $i = 1, \dots, d$

② On each day $t = 1, \dots, T$:

→ . Pick an expert i with probability $\propto w(t-1,i)$

$$\Pr[\text{expert } i \text{ is picked}] = \frac{w(t-1,i)}{\sum_{i=1}^{d} w(t-1,i)}.$$

. Then follow expert i.

→ Update weights of every expert:

$$w(t,j) = w(t-1,j)\cdot(1-\varepsilon L(t,j)) \equiv \begin{cases} \text{→ If correct, } w(t,j) = w(t-1,j) \\ \text{→ If wrong, } w(t,j) = (1-\varepsilon)\, w(t-1,j) \end{cases}$$

Thm: $\text{Total-Loss}_{\text{alg}}^{T \text{ steps}} \leq (1+\varepsilon) L_*(T) + \frac{\ln d}{\varepsilon}$

$\equiv$ $\mathbb{E}[\text{Regret}(T)] \leq \varepsilon \cdot L_*(T) + \frac{\ln d}{\varepsilon}$.

Proof: Main idea: Track $W(t) = \sum_{i=1}^{d} w(t,i)$

$i=1$

Remember: Expert 1: $w(0,i) = 1$

$\downarrow$

Day1 $w(1,1) = (1-\varepsilon)^{L(1,1)}$

Day2: $w(2,1) = (1-\varepsilon)^{L(1,1)+L(2,1)}$

$\vdots$

$$\boxed{w(T,i) = (1-\varepsilon)^{\sum_{t=1}^{T} L(t,i)}}$$

$\rightarrow$ $L(t) = \mathbb{E}[\text{loss incurred on day } t]$.

$$= \sum_{i=1}^{d} \Pr[\text{We pick expert } i] \cdot L(t,i)$$

$$= \sum_{i=1}^{d} \frac{w(t-1,i)}{\sum_{j=1}^{d} w(t-1,j)} \cdot L(t,i)$$

$\sum_{j=1}^{d} w(t-1,j) \rightarrow W(t-1)$.

$$\boxed{L(t) = \frac{1}{W(t-1)} \cdot \sum_{i=1}^{d} w(t-1,i) \cdot L(t,i)} \cdot \rightarrow (\star)$$

$$W(t) = \sum_{i=1}^{d} w(t,i)$$

$$= \sum_{i=1}^{d} w(t,i) \cdot (1 - \varepsilon \cdot L(t,i))$$

$$= \sum_{i=1}^{d} w(t-1,i) - \varepsilon \cdot \sum_{i=1}^{d} w(t-1,i) \cdot L(t,i)$$

$$= W(t-1) - \varepsilon \cdot W(t-1) \cdot L(t) \quad \text{(by } \star\text{)}$$

$$\boxed{W(t) = W(t-1) \cdot (1 - \varepsilon L(t)) \cdot}$$

→ Idea: Compare total weight upper and lower bounds as before.

Claim: $1 - x \leq e^{-x} \quad \forall\, x.$

⇓

Claim: $W(t) = W(t-1)(1-\varepsilon L(t)) \leq W(t-1) \cdot e^{-\varepsilon L(t)}.$

Therefore,

$$W(T) \leq W(0) \cdot e^{-\varepsilon L(1)} \cdot e^{-\varepsilon L(2)} \cdot \cdots \cdot e^{-\varepsilon L(T)}$$

$$= W(0) \cdot e^{-\varepsilon \cdot (L(1) + L(2) + \cdots + L(T))}$$

↓ our total loss.

$$= W(0) \cdot e^{-\varepsilon \cdot A(T)}$$

↓ $A(T) \equiv$ Our total expected loss.

Claim: $W(T) \geq (1-\varepsilon)^{L_\star(T)} \cdot 1.$

Thus,

$$(1-\varepsilon)^{L_\star(T)} \leq d \cdot e^{-\varepsilon A(T)}$$

$$\Rightarrow \quad L_*(T) \cdot \ln(1-\varepsilon) \leq \ln d - \varepsilon \cdot A(T) .$$

$$\Rightarrow \varepsilon A(T) \leq \left(-\ln(1-\varepsilon)\right) \cdot L_*(T) + \ln d .$$

$$\Rightarrow \quad A(T) \leq \left(\frac{-\ln(1-\varepsilon)}{\varepsilon}\right) L_*(T) + \frac{\ln d}{\varepsilon} .$$

Claim: $\quad \frac{-\ln(1-x)}{x} \leq 1+x \quad$ if $\quad x < \frac{1}{2}$.

Therefore, as long as $\varepsilon < \frac{1}{2}$, we get

$$A(T) \leq (1+\varepsilon)\, L_*(T) + \frac{\ln d}{\varepsilon}$$

$\Rightarrow$ Theorem.

Corollary: Setting $\varepsilon = \sqrt{\frac{\ln d}{T}}$, we get

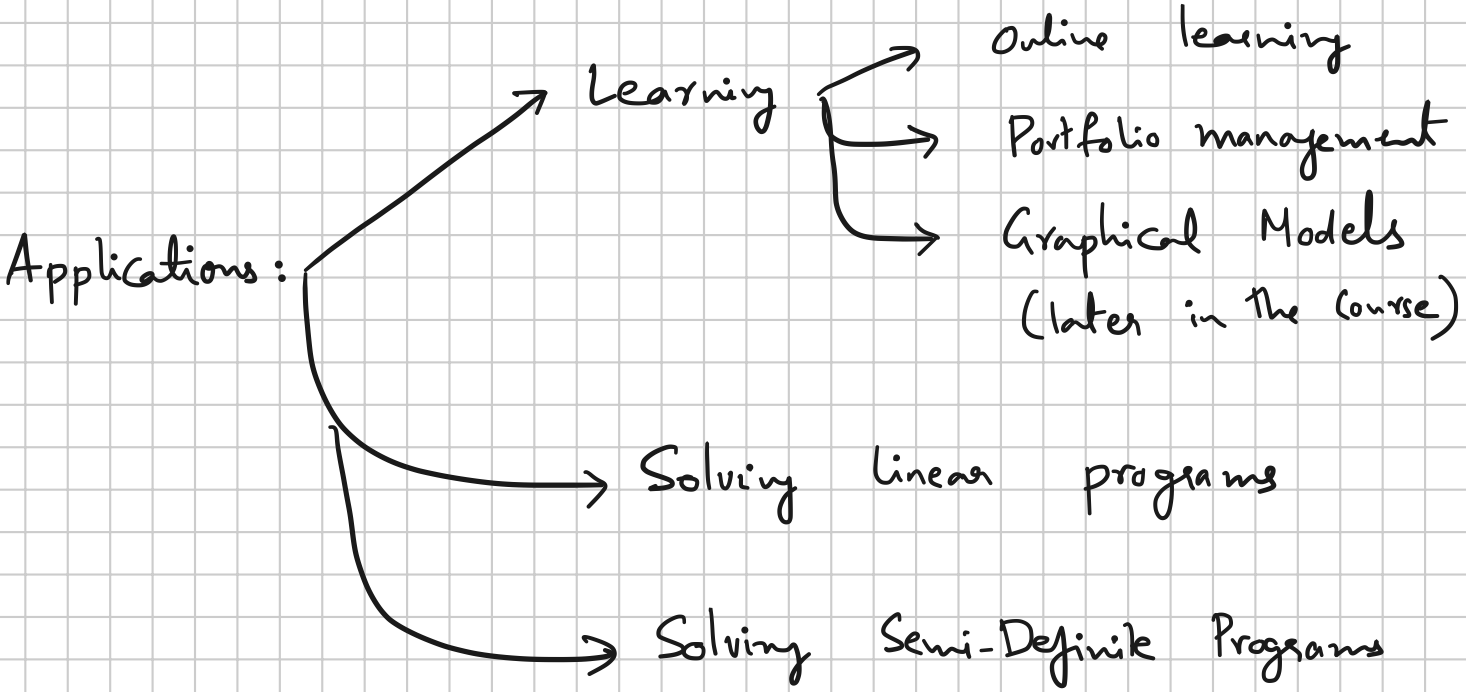
$$A(T) \leq L_*(T) + 2\sqrt{T \ln d} .$$

$\Downarrow$

We have a "No-Regret" algorithm.

$\rightarrow$ Is the best possible regret! (Cannot beat $\Omega(\sqrt{T})$)

$\rightarrow$ $L(t,i) \in [0,1]$.



BOOSTING

→ Goal is to get 90% accuracy (Want)

→ We can get 60% accuracy (Have).

Meta-Question: Can we boost "weak-learners" to strong learners.

BOOSTING ON SAMPLES

Dataset: $(x^1, y^1), (x^2, y^2), \ldots, (x^d, y^d) \in \mathcal{X} \times \mathcal{L}$

$\mathcal{X}$ → Domain, $\mathcal{L}$ → Labels

Hypothesis class $\mathcal{H}$.

Weak Learner: For every distribution D on the dataset, we can find a $h \in \mathcal{H}$, $\Pr_{(x^i, y^i) \sim D}[h(x^i) \neq y^i] \leq \gamma$

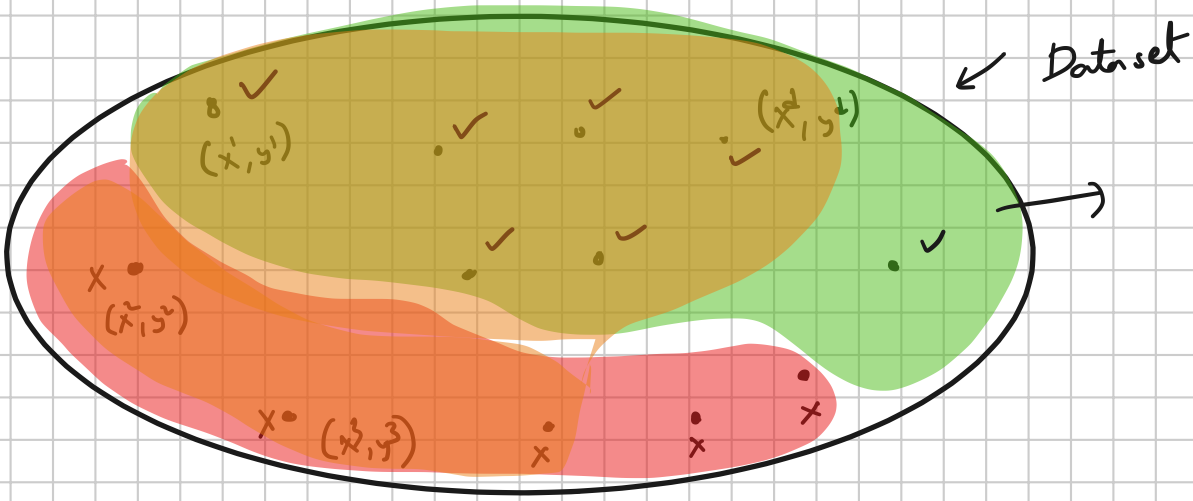
(say $\gamma = 0.4$).

Goal: Combine a few of these weak-learners to get error $\leq \delta$ (say 0.01).

→ Was asked in 1970s by Valiant.

→ Schapire (1989) solved it.

→ Freund & Schapire (1995) gave a practical algorithm (AdaBoost).



What do we have:

dataset

→ Skp 1: $D^{(0)} \equiv$ uniform on the whole dataset.

$h^{(0)} \equiv WL(D^{(0)})$. If only wrong data, the h_i might give very bad results for the current correct ones.

→ Idea: Put more "weight" on the points that you were wrong on before.

- Start with $D^{(0)} \equiv \left(\frac{1}{d}, \frac{1}{d}, \frac{1}{d}, \dots, \frac{1}{d}\right)$ distribution on the d points in the dataset.
- $h^{(0)} \equiv WL(D^{(0)})$.
- For $t = 1, \dots, T$:
  - → Update $D^{(t-1)}$ to $D^{(t)}$
  - → $h^{(t)} \equiv WL(D^{t})$.
- Output $h_{strong} \equiv$ Combine $(h^0, h^1, \dots, h^T)$.

Labels $\equiv \{0, 1\}$.

Idea: Combiner $\equiv$ MAJORITY

Update distributions using MWM.

Imaginary "Learning with Experts game".

	(x^1, y^1)	(x^2, y^2)	...						(x^d, y^d)
h^0	✓	✓	×	×	✓	✓		×	×
"Loss"	1	1	◎	0	1	1		0	0
h^1	✓	✓	×	×	✓	✓	✓	×	×
"Loss"	1	1	◎	0	1	1	1	0	0
			◎	·					

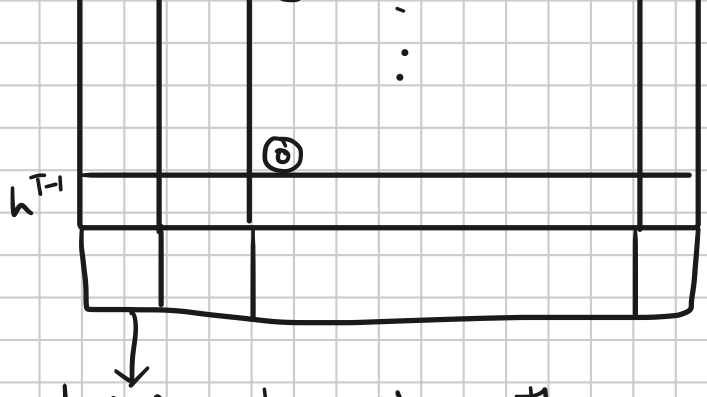
ADABOOST

→ $D^{(0)} \equiv \left(\frac{1}{d}, \frac{1}{d}, \dots, \frac{1}{d}\right)$. $w(0,i) = 1$ for $i = 1, 2, \dots, d$.

→ $h^{(0)} = WL(D^{(0)})$.

For $t = 1, \dots, T$:

- Define $w(t,i) = \begin{cases} (1-\varepsilon)\, w(t-1,i) & \text{if "correct"} \\ \end{cases}$



$w(t-1,i)$ if wrong.

- $D^{(t)}$ = distribution proportional to weights
- $h^t = WL(D^t)$

- Output $h \equiv MAJ(h^0, h^1, h^2, \ldots, h^T)$.

$h(x^i) = y^i$ if there are more than $\frac{T}{2}$ 1's in the column.

"correct" means $h^{t-1}(x^i) = y^i$.

Thm: Adaboost achieves accuracy $1-\delta$ on the dataset if $T \geq \frac{2\ln(1/\delta)}{(\frac{1}{2}-\delta)^2}$.

(For example, $\delta = 0.4$, $\delta = 0.1$

$$\frac{2 \cdot \ln(10)}{(0.1)^2} \simeq 600$$

Why does Adaboost work?

Imagine Adaboost fails to get $1-\delta$ accuracy

$\Rightarrow$ We have at least $d \cdot \delta$ examples where MAJORITY were wrong!

$\Rightarrow$ "Sum of losses" on that column is $< \frac{T}{2}$.

$\Rightarrow$ if MAJORITY is wrong on example i,

Then $w(T, i) \geq (1-\varepsilon)^{\frac{T}{2}}$.

$$d \cdot \delta \cdot (1-\varepsilon)^{\frac{T}{2}} \leq \sum_{i=1}^{d} w(T, i) \leq d \cdot e^{-\varepsilon\delta \cdot T}.$$

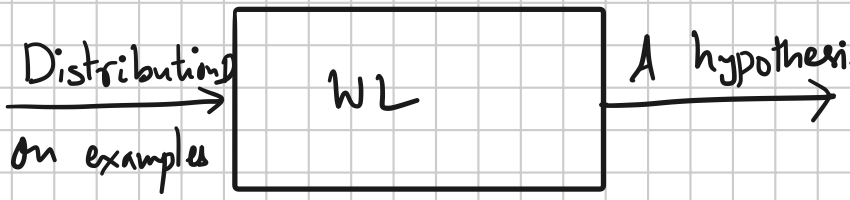
Assuming Adaboost failed.

We will see this in next class!.

Analysis of boosting (Content from lecture 9)

Boosting: $(x^1, y^1), (x^2, y^2), \ldots, (x^d, y^d)$ (d examples)

WL:



$$P_{(x^i, y^i) \sim D}\left[h(x^i) = y^i\right] \geq 1 - \delta.$$

Theorem: ADABOOST after $T \geq \frac{2\ln(1/\delta)}{(\frac{1}{2} - \gamma)^2}$ rounds achieves accuracy $1-\delta$.

Proof idea: Track $W(t) = \sum_{i=1}^{d} w(i, t)$ as a function of t.

ADABOOST

→ $D^{(0)} \equiv \left(\frac{1}{d}, \frac{1}{d}, \dots, \frac{1}{d}\right)$. $w(0,i) = 1$ for $i = 1, 2, \dots, d$.

→ $h^{(1)} = WL(D^{(0)})$.

For $t = 1, \dots, T$:

- Define $w(t,i) = \begin{cases} (1-\varepsilon)\, w(t-1,i) & \text{if "correct"} \\ w(t-1,i) & \text{if wrong.} \end{cases}$
- $D^{(t)}$ = distribution proportional to weights
- $h^t = WL(D^{t-1})$

- Output $h \equiv MAJ(h^0, h^1, h^2, \dots, h^T)$.

Remark: ADABOOST is like running a fictional Learning with experts game where

$$L(i,t) = \begin{cases} 1 & \text{if } h^t(x^i) = y^i \\ 0 & \text{else} \end{cases}$$

$L(t)$ = Expected loss of our algorithm.

Claim: $W(t) \leq W(t-1) \cdot (1-\varepsilon)^{L(t)}$.

$L(t)$: expected "loss" of our algorithm

Claim: $L(t) \geq 1 - \gamma$.

Proof:

$$L(t) = \mathbb{E}[\text{loss we incur}]$$

$$= \sum_{i=1}^{d} \Pr[i \text{ is picked}] \cdot L(i,t)$$

$$= \sum_{i=1}^{d} \Pr[i \text{ is picked when using distribution } D^{t-1}] \cdot \mathbb{1}\left(h^t(x^i) = y^i\right)$$

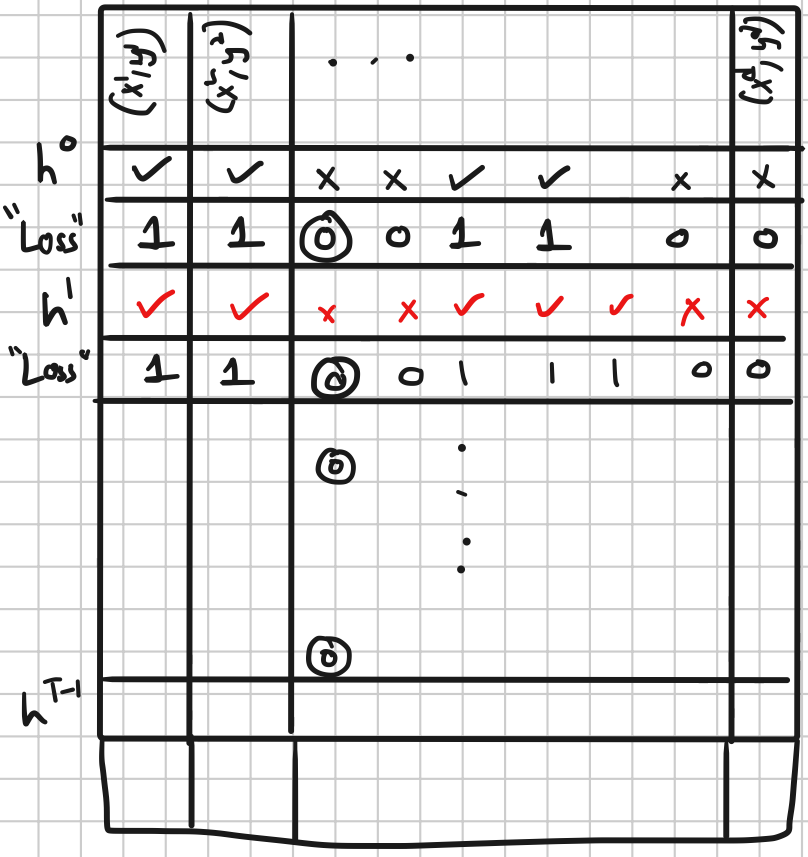
$\geq 1-\delta$

(because $h^t = WL(D^{t-1})$!)

Claim: $W(t) \leq W(t-1) \cdot (1-\varepsilon)^{L(t)} \leq W(t-1) \cdot (1-\varepsilon)^{(1-\delta)}$

Claim: $W(t) \leq W(t-1) \cdot e^{-\varepsilon(1-\delta)}$

$\Rightarrow W(T) \leq W(0) \cdot e^{-\varepsilon T(1-\delta)}$.



$h \equiv MAJ(h^0, h^1, \ldots, h^{T-1})$. Let Bad = All examples where the majority is wrong!

For every i in BAD, we must have

The total loss $\left(= \sum_{t=1}^{T} L(i,t)\right)$ is at most $\frac{T}{2}$.

Claim: $w(i,T) = (1-\varepsilon)^{\sum_{t=1}^{T} L(i,t)}$.

$\Rightarrow$ For every bad index i, $w(i,T) \geq (1-\varepsilon)^{\frac{T}{2}}$.

What do we have:

$$\sum_{i \in BAD} w(i,T) \leq \sum_{i=1}^{d} w(i,T) = W(T) \leq W(0) \cdot e^{-\varepsilon T(1-\delta)}.$$

$$\Rightarrow |BAD| \cdot (1-\varepsilon)^{\frac{T}{2}} \leq d \cdot e^{-\varepsilon T(1-\delta)}.$$

$$\Rightarrow \left(\frac{|BAD|}{d}\right) \leq e^{-\varepsilon T(1-\delta)} \cdot (1-\varepsilon)^{-\frac{T}{2}}.$$

(Recall the inequality $-\frac{\ln(1-\varepsilon)}{\varepsilon} \leq 1+\varepsilon$).

$$\ln\left(\frac{1}{1-\varepsilon}\right) \leq \varepsilon(1+\varepsilon) \Leftarrow \frac{\ln\left(\frac{1}{1-\varepsilon}\right)}{\varepsilon} \leq 1+\varepsilon$$

$$\left(\frac{|BAD|}{d}\right) \leq e^{-\varepsilon T(1-\delta)} \cdot \left(\frac{1}{1-\varepsilon}\right)^{\frac{T}{2}}.$$

$$-\varepsilon T(1-\delta) \qquad \frac{\varepsilon(1+\varepsilon)\,T}{2}$$

$$\leq e \cdot e^{\frac{\varepsilon}{2}} \cdot$$

$$= e^{-\varepsilon T\left((1-\delta) - \frac{1}{2} - \frac{\varepsilon}{2}\right)}$$

$$= e^{-\varepsilon T\left(\left(\frac{1}{2}-\delta\right) - \frac{\varepsilon}{2}\right)}$$

$x(a-x)$

Recall: We get to choose ε.

So set $\varepsilon = \left(\frac{1}{2} - \delta\right)$.

$$\Rightarrow \frac{|BAD|}{d} \leq e^{-\left(\frac{1}{2}-\delta\right)T \cdot \frac{1}{2}\left(\frac{1}{2}-\delta\right)}$$

$$= e^{-\frac{T\left(\frac{1}{2}-\delta\right)^2}{2}}.$$

So, if $T \geq \frac{2\ln(1/\delta)}{\left(\frac{1}{2}-\delta\right)^2}$, then

$$\boxed{\frac{|BAD|}{d} \leq \delta}$$

Summary: Boosting is possible.

Is possible with a very practical algorithm: ADABOOST.

→ No regret algorithms are very powerful.

→ We can use learning with experts / MWM for problems that have nothing to do with online learning! → "Private" algorithms
→ Graphical Models.

MWM is quite useful when you have to come up with clever distributions.