Introduction to Online Learning Algorithms

Yoav Freund

January 6, 2020

Outline

Example trace for Halving Algorithm

Example trace for Halving Algorithm

expert1
expert2
expert3
expert4
expert5
expert6
expert7
expert8

alg.

Example trace for Halving Algorithm

expert1
expert2
expert3
expert4
expert5
expert6
expert7
expert8

alg.
outcome

Example trace for Halving Algorithm

	$t = 1$
expert1	1
expert2	1
expert3	0
expert4	1
expert5	1
expert6	0
expert7	1
expert8	1
alg.	
outcome	

Example trace for Halving Algorithm

	$t = 1$
expert1	1
expert2	1
expert3	0
expert4	1
expert5	1
expert6	0
expert7	1
expert8	1
alg.	1
outcome	

Example trace for Halving Algorithm

	$t = 1$
expert1	1
expert2	1
expert3	0
expert4	1
expert5	1
expert6	0
expert7	1
expert8	1
alg.	1
outcome	1

Example trace for Halving Algorithm

	$t = 1$	$t = 2$
expert1	1	1
expert2	1	0
expert3	0	-
expert4	1	0
expert5	1	0
expert6	0	-
expert7	1	1
expert8	1	1
alg.	1	
outcome	1	

Example trace for Halving Algorithm

	$t = 1$	$t = 2$
expert1	1	1
expert2	1	0
expert3	0	-
expert4	1	0
expert5	1	0
expert6	0	-
expert7	1	1
expert8	1	1
alg.	1	0
outcome	1	

Example trace for Halving Algorithm

	$t = 1$	$t = 2$
expert1	1	1
expert2	1	0
expert3	0	-
expert4	1	0
expert5	1	0
expert6	0	-
expert7	1	1
expert8	1	1
alg.	1	0
outcome	1	1

Example trace for Halving Algorithm

	$t = 1$	$t = 2$	$t = 3$
expert1	1	1	1
expert2	1	0	-
expert3	0	-	-
expert4	1	0	-
expert5	1	0	-
expert6	0	-	-
expert7	1	1	1
expert8	1	1	1
alg.	1	0	
outcome	1	1	

Example trace for Halving Algorithm

	$t = 1$	$t = 2$	$t = 3$
expert1	1	1	1
expert2	1	0	-
expert3	0	-	-
expert4	1	0	-
expert5	1	0	-
expert6	0	-	-
expert7	1	1	1
expert8	1	1	1
alg.	1	0	1
outcome	1	1	

Example trace for Halving Algorithm

	$t = 1$	$t = 2$	$t = 3$
expert1	1	1	1
expert2	1	0	-
expert3	0	-	-
expert4	1	0	-
expert5	1	0	-
expert6	0	-	-
expert7	1	1	1
expert8	1	1	1
alg.	1	0	1
outcome	1	1	1

Example trace for Halving Algorithm

	$t = 1$	$t = 2$	$t = 3$	$t = 4$
expert1	1	1	1	1
expert2	1	0	-	-
expert3	0	-	-	-
expert4	1	0	-	-
expert5	1	0	-	-
expert6	0	-	-	-
expert7	1	1	1	1
expert8	1	1	1	0
alg.	1	0	1	
outcome	1	1	1	

Example trace for Halving Algorithm

	$t = 1$	$t = 2$	$t = 3$	$t = 4$
expert1	1	1	1	1
expert2	1	0	-	-
expert3	0	-	-	-
expert4	1	0	-	-
expert5	1	0	-	-
expert6	0	-	-	-
expert7	1	1	1	1
expert8	1	1	1	0
alg.	1	0	1	1
outcome	1	1	1	

Example trace for Halving Algorithm

	$t = 1$	$t = 2$	$t = 3$	$t = 4$
expert1	1	1	1	1
expert2	1	0	-	-
expert3	0	-	-	-
expert4	1	0	-	-
expert5	1	0	-	-
expert6	0	-	-	-
expert7	1	1	1	1
expert8	1	1	1	0
alg.	1	0	1	1
outcome	1	1	1	0

Example trace for Halving Algorithm

	$t = 1$	$t = 2$	$t = 3$	$t = 4$	$t = 5$
expert1	1	1	1	1	-
expert2	1	0	-	-	-
expert3	0	-	-	-	-
expert4	1	0	-	-	-
expert5	1	0	-	-	-
expert6	0	-	-	-	-
expert7	1	1	1	1	0
expert8	1	1	1	0	-
alg.	1	0	1	1	
outcome	1	1	1	0	

Example trace for Halving Algorithm

	$t = 1$	$t = 2$	$t = 3$	$t = 4$	$t = 5$
expert1	1	1	1	1	-
expert2	1	0	-	-	-
expert3	0	-	-	-	-
expert4	1	0	-	-	-
expert5	1	0	-	-	-
expert6	0	-	-	-	-
expert7	1	1	1	1	0
expert8	1	1	1	0	-
alg.	1	0	1	1	0
outcome	1	1	1	0	

Example trace for Halving Algorithm

	$t = 1$	$t = 2$	$t = 3$	$t = 4$	$t = 5$
expert1	1	1	1	1	-
expert2	1	0	-	-	-
expert3	0	-	-	-	-
expert4	1	0	-	-	-
expert5	1	0	-	-	-
expert6	0	-	-	-	-
expert7	1	1	1	1	0
expert8	1	1	1	0	-
alg.	1	0	1	1	0
outcome	1	1	1	0	0

Mistake bound for Halving algorithm

- Each time algorithm makes a mistakes, the pool of perfect experts is halved (at least).

Mistake bound for Halving algorithm

- Each time algorithm makes a mistakes, the pool of perfect experts is halved (at least).
- We assume that at least one expert is perfect.

Mistake bound for Halving algorithm

- Each time algorithm makes a mistakes, the pool of perfect experts is halved (at least).
- We assume that at least one expert is perfect.
- Number of mistakes is at most $\log_2 N$.

Characteristics of individual sequence analysis

- No stochastic assumptions.

Characteristics of individual sequence analysis

- No stochastic assumptions.
- Proof is based on combining a lower and upper bounds on a Potential

Characteristics of individual sequence analysis

- No stochastic assumptions.
- Proof is based on combining a lower and upper bounds on a Potential
- Here, Potential = number of experts in pool.

Optimality of Halving algorithm

- Adversary chooses expert predictions and outcomes so as to maximize the number of mistakes.

Optimality of Halving algorithm

- Adversary chooses expert predictions and outcomes so as to maximize the number of mistakes.
- The best strategy for the adversary: split a pool with m experts into
  $\lfloor m/2 \rfloor$ predicting 0
  $\lceil m/2 \rceil$ predicting 1
  Make the actual output 0.

Optimality of Halving algorithm

- Adversary chooses expert predictions and outcomes so as to maximize the number of mistakes.
- The best strategy for the adversary: split a pool with m experts into
  $\lfloor m/2 \rfloor$ predicting 0
  $\lceil m/2 \rceil$ predicting 1
  Make the actual output 0.
- The halving algorithm can be forced to k^* mistakes.
  $k^* \leq \log N$.

Optimality of Halving algorithm

- Adversary chooses expert predictions and outcomes so as to maximize the number of mistakes.
- The best strategy for the adversary: split a pool with m experts into
  $\lfloor m/2 \rfloor$ predicting 0
  $\lceil m/2 \rceil$ predicting 1
  Make the actual output 0.
- The halving algorithm can be forced to k^* mistakes.
  $k^* \leq \log N$.
- $k^{\cdot}$ $m_0 = N$, $m_{i+1} = \lfloor m_i \rfloor$, $m_{k^*} = 1$.

Optimality of Halving algorithm

- Adversary chooses expert predictions and outcomes so as to maximize the number of mistakes.
- The best strategy for the adversary: split a pool with m experts into
  $\lfloor m/2 \rfloor$ predicting 0
  $\lceil m/2 \rceil$ predicting 1
  Make the actual output 0.
- The halving algorithm can be forced to k^* mistakes.
  $k^* \leq \log N$.
- $k^{:}$ $m_0 = N$, $m_{i+1} = \lfloor m_i \rfloor$, $m_{k^*} = 1$.
- If $N = 2^k$ then $k^* = k$, otherwise $k^* < \log N$

Optimality of Halving algorithm

- Adversary chooses expert predictions and outcomes so as to maximize the number of mistakes.
- The best strategy for the adversary: split a pool with m experts into
  $\lfloor m/2 \rfloor$ predicting 0
  $\lceil m/2 \rceil$ predicting 1
  Make the actual output 0.
- The halving algorithm can be forced to k^* mistakes.
  $k^* \leq \log N$.
- $k^:$ $m_0 = N$, $m_{i+1} = \lfloor m_i \rfloor$, $m_{k^*} = 1$.
- If $N = 2^k$ then $k^* = k$, otherwise $k^* < \log N$
- k^* is the Min-Max number of mistakes: algorithm never makes more than k^* mistakes, adversary can force k^* mistakes.

The hedging problem

What should we do if none of the experts is perfect, and we have no a-priori knowledge how good the best expert is?

- N possible actions

The hedging problem

What should we do if none of the experts is perfect, and we have no a-priori knowledge how good the best expert is?

- N possible actions
- At each time step t:

The hedging problem

What should we do if none of the experts is perfect, and we have no a-priori knowledge how good the best expert is?

- N possible actions
- At each time step t:
    - Algorithm chooses a distribution $\vec{P}^t$ over actions.

The hedging problem

What should we do if none of the experts is perfect, and we have no a-priori knowledge how good the best expert is?

- N possible actions
- At each time step t:
  - Algorithm chooses a distribution $\vec{P}^t$ over actions.
  - Losses $0 \leq \ell_i^t \leq 1$ of all actions $i = 1, \ldots, N$ are revealed.

The hedging problem

What should we do if none of the experts is perfect, and we have no a-priori knowledge how good the best expert is?

- N possible actions
- At each time step t:
  - Algorithm chooses a distribution $\vec{P}^t$ over actions.
  - Losses $0 \leq \ell_i^t \leq 1$ of all actions $i = 1, \ldots, N$ are revealed.
  - Algorithm suffers **expected** loss.

The hedging problem

What should we do if none of the experts is perfect, and we have no a-priori knowledge how good the best expert is?

- N possible actions
- At each time step t:
  - Algorithm chooses a distribution $\vec{P}^t$ over actions.
  - Losses $0 \leq \ell_i^t \leq 1$ of all actions $i = 1, \ldots, N$ are revealed.
  - Algorithm suffers **expected** loss.
- **Goal:** minimize total expected loss

The hedging problem

What should we do if none of the experts is perfect, and we have no a-priori knowledge how good the best expert is?

- N possible actions
- At each time step t:
  - Algorithm chooses a distribution $\vec{P}^t$ over actions.
  - Losses $0 \leq \ell_i^t \leq 1$ of all actions $i = 1, \ldots, N$ are revealed.
  - Algorithm suffers **expected** loss.
- **Goal:** minimize total expected loss
- Here we have stochasticity - but only in **algorithm**, not in **outcome**

The hedging problem

What should we do if none of the experts is perfect, and we have no a-priori knowledge how good the best expert is?

- N possible actions
- At each time step t:
    - Algorithm chooses a distribution $\vec{P}^t$ over actions.
    - Losses $0 \leq \ell_i^t \leq 1$ of all actions $i = 1, \ldots, N$ are revealed.
    - Algorithm suffers **expected** loss.
- **Goal:** minimize total expected loss
- Here we have stochasticity - but only in **algorithm**, not in **outcome**
- Fits nicely in game theory

Hedging vs. Halving

- Like halving - we want to zoom into best action (expert).

Hedging vs. Halving

- Like halving - we want to zoom into best action (expert).
- Unlike halving - no action is perfect.

Hedging vs. Halving

- Like halving - we want to zoom into best action (expert).
- Unlike halving - no action is perfect.
- Basic idea - reduce probability of lossy actions,
  but not all the way to zero.

Hedging vs. Halving

- Like halving - we want to zoom into best action (expert).
- Unlike halving - no action is perfect.
- Basic idea - reduce probability of lossy actions,
  but not all the way to zero.
- **Modified Goal:**
  minimize REGRET =
  expected total loss
  minus
  minimal total loss of repeating one action.

The Hedge Algorithm

Consider action i at time t

- Total loss:

$$L_i^t = \sum_{s=1}^{t-1} \ell_i^s$$

The Hedge Algorithm

Consider action i at time t

- Total loss:

$$L_i^t = \sum_{s=1}^{t-1} \ell_i^s$$

- Weight:

$$W_i^t = e^{-\eta L_i^t}$$

The Hedge Algorithm

Consider action i at time t

- Total loss:
$$L_i^t = \sum_{s=1}^{t-1} \ell_i^s$$
- Weight:
$$W_i^t = e^{-\eta L_i^t}$$
- $\eta > 0$ is the learning rate parameter. Halving: $\eta = \infty$

The Hedge Algorithm

Consider action i at time t

- Total loss:

$$L_i^t = \sum_{s=1}^{t-1} \ell_i^s$$

- Weight:

$$W_i^t = e^{-\eta L_i^t}$$

- $\eta > 0$ is the learning rate parameter. Halving: $\eta = \infty$
- Probability:

$$P_i^t = \frac{W_i^t}{\sum_{j=1}^{N} W_j^t}$$

Example trace for Hedge Algorithm

$\eta = 1$

Example trace for Hedge Algorithm

$$\eta = 1$$

expert1
expert2
expert3
expert4
expert5
expert6
expert7
expert8

alg.

Example trace for Hedge Algorithm

$\eta = 1$

	$\vec{W}^1$
expert1	1
expert2	1
expert3	1
expert4	1
expert5	1
expert6	1
expert7	1
expert8	1
alg.	

Example trace for Hedge Algorithm

$\eta = 1$

	$\vec{W}^1$	L^1
expert1	1	.1
expert2	1	.8
expert3	1	.3
expert4	1	.1
expert5	1	.9
expert6	1	0
expert7	1	1
expert8	1	.8

alg.

Example trace for Hedge Algorithm

$\eta = 1$

	$\vec{W}^1$	L^1
expert1	1	.1
expert2	1	.8
expert3	1	.3
expert4	1	.1
expert5	1	.9
expert6	1	0
expert7	1	1
expert8	1	.8
alg.		.5

Example trace for Hedge Algorithm

$\eta = 1$

	$\vec{W}^1$	L^1	$\vec{W}^2$
expert1	1	.1	.90
expert2	1	.8	.45
expert3	1	.3	.74
expert4	1	.1	.90
expert5	1	.9	.41
expert6	1	0	1
expert7	1	1	.37
expert8	1	.8	.45
alg.		.5	

Example trace for Hedge Algorithm

$\eta = 1$

	$\vec{W}^1$	L^1	$\vec{W}^2$	L^2
expert1	1	.1	.90	.1
expert2	1	.8	.45	.5
expert3	1	.3	.74	.2
expert4	1	.1	.90	.7
expert5	1	.9	.41	1
expert6	1	0	1	.1
expert7	1	1	.37	.5
expert8	1	.8	.45	.2
alg.		.5		

Example trace for Hedge Algorithm

$\eta = 1$

	$\vec{W}^1$	L^1	$\vec{W}^2$	L^2
expert1	1	.1	.90	.1
expert2	1	.8	.45	.5
expert3	1	.3	.74	.2
expert4	1	.1	.90	.7
expert5	1	.9	.41	1
expert6	1	0	1	.1
expert7	1	1	.37	.5
expert8	1	.8	.45	.2
alg.		.5		.36

Example trace for Hedge Algorithm

$\eta = 1$

	$\vec{W}^1$	L^1	$\vec{W}^2$	L^2	$\vec{W}^3$
expert1	1	.1	.90	.1	0.82
expert2	1	.8	.45	.5	0.27
expert3	1	.3	.74	.2	0.61
expert4	1	.1	.90	.7	0.45
expert5	1	.9	.41	1	0.15
expert6	1	0	1	.1	0.91
expert7	1	1	.37	.5	0.22
expert8	1	.8	.45	.2	0.37
alg.		.5		.36	

Example trace for Hedge Algorithm

$\eta = 1$

	$\vec{W}^1$	L^1	$\vec{W}^2$	L^2	$\vec{W}^3$	L^3
expert1	1	.1	.90	.1	0.82	0
expert2	1	.8	.45	.5	0.27	.2
expert3	1	.3	.74	.2	0.61	.2
expert4	1	.1	.90	.7	0.45	.8
expert5	1	.9	.41	1	0.15	.8
expert6	1	0	1	.1	0.91	.2
expert7	1	1	.37	.5	0.22	.4
expert8	1	.8	.45	.2	0.37	.6
alg.		.5		.36		

Example trace for Hedge Algorithm

$\eta = 1$

	$\vec{W}^1$	L^1	$\vec{W}^2$	L^2	$\vec{W}^3$	L^3
expert1	1	.1	.90	.1	0.82	0
expert2	1	.8	.45	.5	0.27	.2
expert3	1	.3	.74	.2	0.61	.2
expert4	1	.1	.90	.7	0.45	.8
expert5	1	.9	.41	1	0.15	.8
expert6	1	0	1	.1	0.91	.2
expert7	1	1	.37	.5	0.22	.4
expert8	1	.8	.45	.2	0.37	.6
alg.		.5		.36		.30

Example trace for Hedge Algorithm

$\eta = 1$

	$\vec{W}^1$	L^1	$\vec{W}^2$	L^2	$\vec{W}^3$	L^3	total
expert1	1	.1	.90	.1	0.82	0	.2
expert2	1	.8	.45	.5	0.27	.2	1.5
expert3	1	.3	.74	.2	0.61	.2	.7
expert4	1	.1	.90	.7	0.45	.8	1.6
expert5	1	.9	.41	1	0.15	.8	2.7
expert6	1	0	1	.1	0.91	.2	.3
expert7	1	1	.37	.5	0.22	.4	1.9
expert8	1	.8	.45	.2	0.37	.6	1.6
alg.		.5		.36		.30	

Example trace for Hedge Algorithm

$\eta = 1$

	$\vec{W}^1$	L^1	$\vec{W}^2$	L^2	$\vec{W}^3$	L^3	total
expert1	1	.1	.90	.1	0.82	0	.2
expert2	1	.8	.45	.5	0.27	.2	1.5
expert3	1	.3	.74	.2	0.61	.2	.7
expert4	1	.1	.90	.7	0.45	.8	1.6
expert5	1	.9	.41	1	0.15	.8	2.7
expert6	1	0	1	.1	0.91	.2	.3
expert7	1	1	.37	.5	0.22	.4	1.9
expert8	1	.8	.45	.2	0.37	.6	1.6
alg.		.5		.36		.30	1.16

Bound for Hedge Algorithm

- L^t_{Hedge}: Expected total loss of Hedge algorithm for time $1, 2, \ldots, t$

Bound for Hedge Algorithm

- L^t_{Hedge}: Expected total loss of Hedge algorithm for time $1, 2, \ldots, t$
- $$\forall t, i, \quad L_{\text{Hedge}} \leq \frac{\ln N + \eta L^t_i}{1 - e^{-\eta}}$$

Bound for Hedge Algorithm

- L^t_{Hedge}: Expected total loss of Hedge algorithm for time $1, 2, \ldots, t$
- $$\forall t, i, \quad L_{\text{Hedge}} \leq \frac{\ln N + \eta L^t_i}{1 - e^{-\eta}}$$
- Which implies

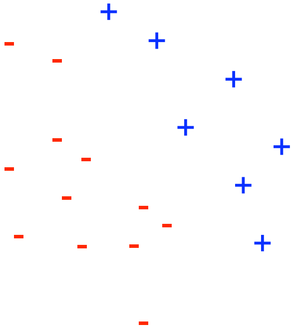
$$\forall t, \quad L_{\text{Hedge}} \leq \min_i \left(\frac{\ln N + \eta L^t_i}{1 - e^{-\eta}} \right)$$

Bound for Hedge Algorithm

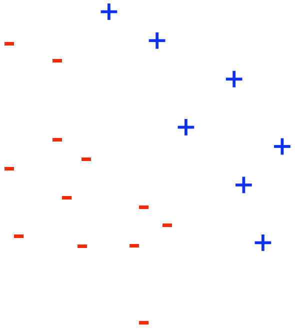
- L^t_{Hedge}: Expected total loss of Hedge algorithm for time $1, 2, \ldots, t$
- $$\forall t, i, \quad L_{\text{Hedge}} \leq \frac{\ln N + \eta L^t_i}{1 - e^{-\eta}}$$
- Which implies
$$\forall t, \quad L_{\text{Hedge}} \leq \min_i \left(\frac{\ln N + \eta L^t_i}{1 - e^{-\eta}} \right)$$
- Proof and choice of η: next class.

The Perceptron Problem

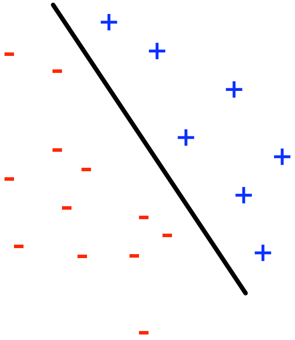
The Perceptron Problem



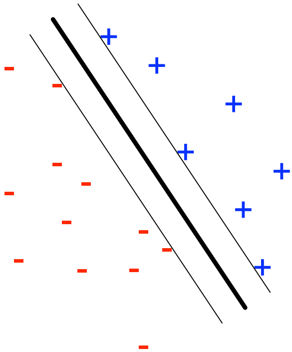
The Perceptron Problem



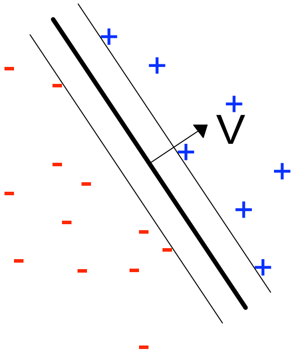
The Perceptron Problem



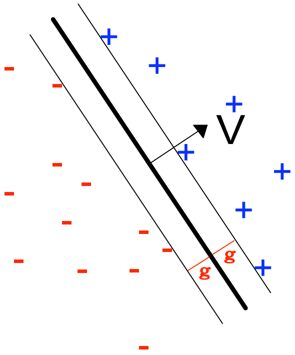
The Perceptron Problem



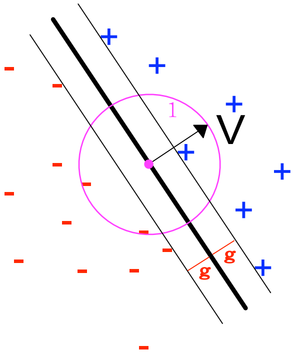
The Perceptron Problem



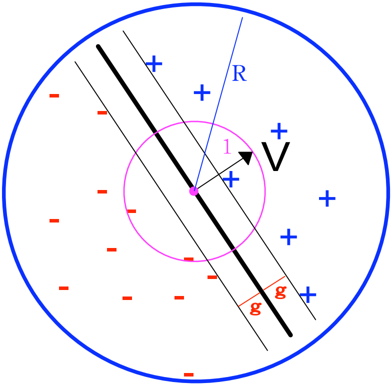
The Perceptron Problem



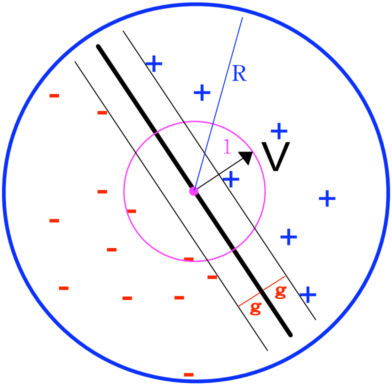
The Perceptron Problem



The Perceptron Problem

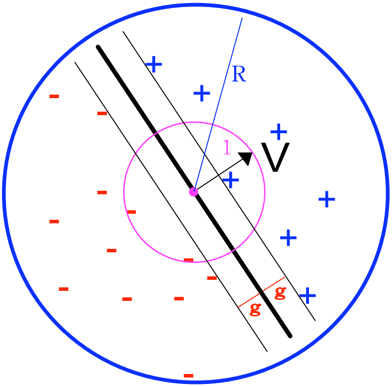


The Perceptron Problem



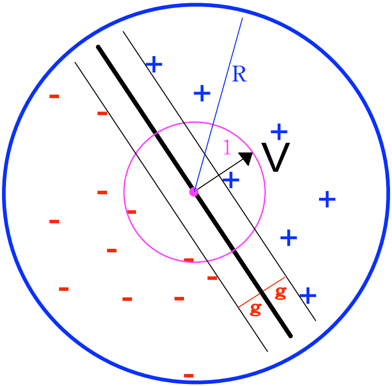
- $\|\vec{V}\| = 1$

The Perceptron Problem



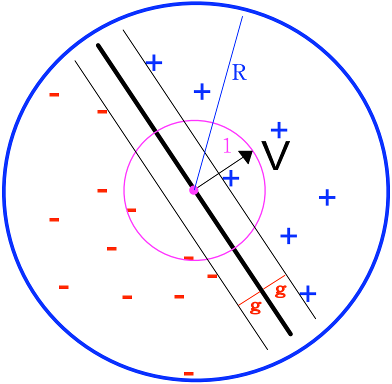
- $\|\vec{V}\| = 1$
- Example = $(\vec{X}, y)$, $y \in \{-1, +1\}$.

The Perceptron Problem



- $\|\vec{V}\| = 1$
- Example = $(\vec{X}, y)$, $y \in \{-1, +1\}$.
- $\forall \vec{X},\ \ \|\vec{X}\| \leq R$.

The Perceptron Problem



- $\|\vec{V}\| = 1$
- Example = $(\vec{X}, y)$, $y \in \{-1, +1\}$.
- $\forall \vec{X}, \ \ \|\vec{X}\| \leq R$.
- $\forall (\vec{X}, y)$, $y(\vec{X} \cdot \vec{V}) \geq g$

The Perceptron learning algorithm

- An online algorithm. Examples presented one by one.

The Perceptron learning algorithm

- An online algorithm. Examples presented one by one.
- start with $\vec{W}_0 = \vec{0}$.

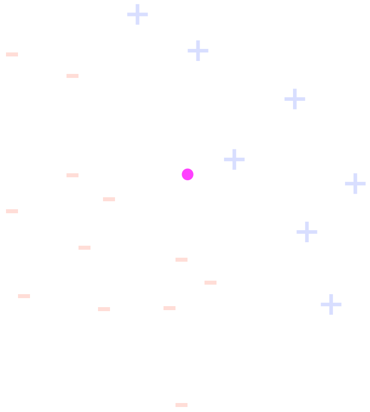
The Perceptron learning algorithm

- An online algorithm. Examples presented one by one.
- start with $\vec{W}_0 = \vec{0}$.
- If mistake: $(\vec{W}_i \cdot \vec{X}_i) y_i \leq 0$

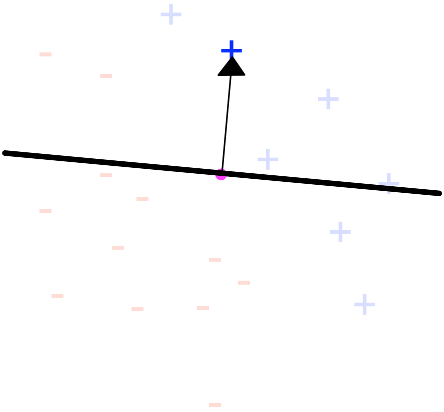
The Perceptron learning algorithm

- An online algorithm. Examples presented one by one.
- start with $\vec{W}_0 = \vec{0}$.
- If mistake: $(\vec{W}_i \cdot \vec{X}_i)y_i \leq 0$
  - Update $\vec{W}_{i+1} = \vec{W}_i + y_i X_i$.

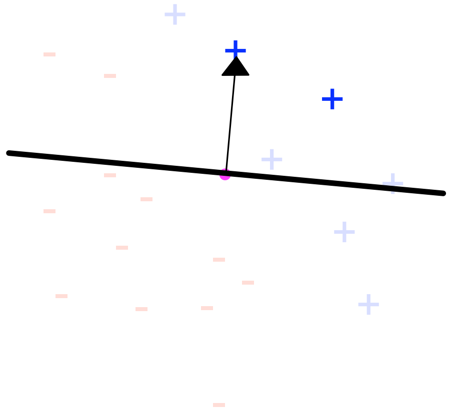
Example trace for the perceptron algorithm



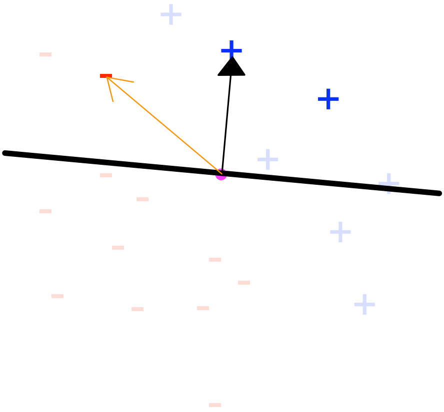
Example trace for the perceptron algorithm



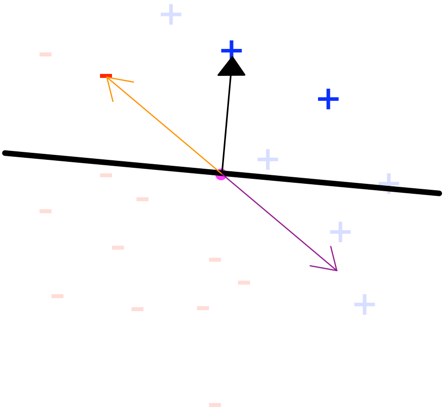
Example trace for the perceptron algorithm



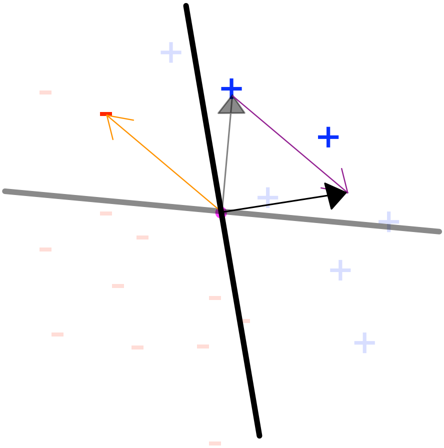
Example trace for the perceptron algorithm



Example trace for the perceptron algorithm



Example trace for the perceptron algorithm



Bound on number of mistakes

- The number of mistakes that the perceptron algorithm can make is at most $\left(\frac{R}{g}\right)^2$.

Bound on number of mistakes

- The number of mistakes that the perceptron algorithm can make is at most $\left(\frac{R}{g}\right)^2$.
- Proof by combining upper and lower bounds on $\|\vec{W}\|$.

Pythagorian Lemma

If $(\vec{W}_i \cdot X_i)y < 0$ then

Pythagorian Lemma

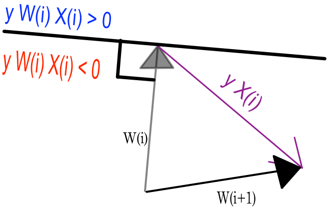
If $(\vec{W}_i \cdot X_i)y < 0$ then

$$\|\vec{W}_{i+1}\|^2 = \|\vec{W}_i + y_i\vec{X}_i\|^2 \leq \|\vec{W}_i\|^2 + \|\vec{X}_i\|^2$$

Pythagorian Lemma

If $(\vec{W}_i \cdot X_i)y < 0$ then

$$\|\vec{W}_{i+1}\|^2 = \|\vec{W}_i + y_i\vec{X}_i\|^2 \leq \|\vec{W}_i\|^2 + \|\vec{X}_i\|^2$$



Upper bound on $\|\vec{W}_i\|$

Upper bound on $\|\vec{W}_i\|$

Proof by induction

- Claim: $\|\vec{W}_i\|^2 \leq iR^2$

Upper bound on $\|\vec{W}_i\|$

Proof by induction

- Claim: $\|\vec{W}_i\|^2 \leq iR^2$
- Base: $i = 0$, $\|\vec{W}_0\|^2 = 0$

Upper bound on $\|\vec{W}_i\|$

Proof by induction

- Claim: $\|\vec{W}_i\|^2 \leq iR^2$
- Base: $i = 0$, $\|\vec{W}_0\|^2 = 0$
- Induction step (assume for i and prove for $i + 1$):
  $\|\vec{W}_{i+1}\|^2 \leq \|\vec{W}_i\|^2 + \|\vec{X}_i\|^2$
  $\leq \|\vec{W}_i\|^2 + R^2 \leq (i + 1)R^2$

Lower bound on $\|\vec{W}_i\|$

Lower bound on $\|\vec{W}_i\|$

$$\|\vec{W}_i\| \geq \vec{W}_i \cdot \vec{V} \text{ because } \|\vec{V}\| = 1.$$

Lower bound on $\|\vec{W}_i\|$

$\|\vec{W}_i\| \geq \vec{W}_i \cdot \vec{V}$ because $\|\vec{V}\| = 1$.
We prove a lower bound on $\vec{W}_i \cdot \vec{V}$ using induction over i

- Claim: $\vec{W}_i \cdot \vec{V} \geq ig$

Lower bound on $\|\vec{W}_i\|$

$\|\vec{W}_i\| \geq \vec{W}_i \cdot \vec{V}$ because $\|\vec{V}\| = 1$.
We prove a lower bound on $\vec{W}_i \cdot \vec{V}$ using induction over i

- Claim: $\vec{W}_i \cdot \vec{V} \geq ig$
- Base: $i = 0$, $\vec{W}_0 \cdot \vec{V} = 0$

Lower bound on $\|\vec{W}_i\|$

$\|\vec{W}_i\| \geq \vec{W}_i \cdot \vec{V}$ because $\|\vec{V}\| = 1$.
We prove a lower bound on $\vec{W}_i \cdot \vec{V}$ using induction over i

- Claim: $\vec{W}_i \cdot \vec{V} \geq ig$
- Base: $i = 0$, $\vec{W}_0 \cdot \vec{V} = 0$
- Induction step (assume for i and prove for $i + 1$):
  $\vec{W}_{i+1} \cdot \vec{V} = \left(\vec{W}_i + \vec{X}_i y_i\right) \vec{V}$

Lower bound on $\|\vec{W}_i\|$

$\|\vec{W}_i\| \geq \vec{W}_i \cdot \vec{V}$ because $\|\vec{V}\| = 1$.
We prove a lower bound on $\vec{W}_i \cdot \vec{V}$ using induction over i

- Claim: $\vec{W}_i \cdot \vec{V} \geq ig$
- Base: $i = 0$, $\vec{W}_0 \cdot \vec{V} = 0$
- Induction step (assume for i and prove for $i + 1$):
  $\vec{W}_{i+1} \cdot \vec{V} = \left(\vec{W}_i + \vec{X}_i y_i\right) \vec{V}$

Lower bound on $\|\vec{W}_i\|$

$\|\vec{W}_i\| \geq \vec{W}_i \cdot \vec{V}$ because $\|\vec{V}\| = 1$.
We prove a lower bound on $\vec{W}_i \cdot \vec{V}$ using induction over i

- Claim: $\vec{W}_i \cdot \vec{V} \geq ig$
- Base: $i = 0$, $\vec{W}_0 \cdot \vec{V} = 0$
- Induction step (assume for i and prove for $i+1$):
  $\vec{W}_{i+1} \cdot \vec{V} = \left(\vec{W}_i + \vec{X}_i y_i\right) \vec{V} = \vec{W}_i \cdot \vec{V} + y_i \vec{X}_i \cdot \vec{V}$
  $\geq ig + g = (i+1)g$

Combining the upper and lower bounds

Combining the upper and lower bounds

$$(ig)^2 \leq \|\vec{W}_i\|^2 \leq iR^2$$

Combining the upper and lower bounds

$$(ig)^2 \leq \|\vec{W}_i\|^2 \leq iR^2$$

Thus:

$$i \leq \left(\frac{R}{g}\right)^2$$

Estimating the bias of a coin

- We observe n coin flips:
  H,T,T,H,H,T,H,T,T

Estimating the bias of a coin

- We observe n coin flips:
  H,T,T,H,H,T,H,T,T
- We want to estimate the **probability** that the next flip will be **H**ead.

Estimating the bias of a coin

- We observe n coin flips:
  H,T,T,H,H,T,H,T,T
- We want to estimate the **probability** that the next flip will be **H**ead.
- Natural Answer:

$$\frac{\#\mathbf{H}}{n} = \frac{4}{9}$$

What if the estimation has to be done online?

- We observe a bf sequence of coin flips, and have to predict the probability of **H**ead at each step:

  p_0,

What if the estimation has to be done online?

- We observe a bf sequence of coin flips, and have to predict the probability of **H**ead at each step:
  p_0,

What if the estimation has to be done online?

- We observe a bf sequence of coin flips, and have to predict the probability of **H**ead at each step:
  p_0,**H**,

What if the estimation has to be done online?

- We observe a bf sequence of coin flips, and have to predict the probability of **H**ead at each step:
  p_0,**H**,p_1,

What if the estimation has to be done online?

- We observe a bf sequence of coin flips, and have to predict the probability of **H**ead at each step:
  p_0,**H**,p_1,**T**,

What if the estimation has to be done online?

- We observe a bf sequence of coin flips, and have to predict the probability of **H**ead at each step:
  p_0,**H**,p_1,**T**,p_2,

What if the estimation has to be done online?

- We observe a bf sequence of coin flips, and have to predict the probability of **H**ead at each step:
  p_0,**H**,p_1,**T**,p_2,...
- What would be a good value for p_0?

What if the estimation has to be done online?

- We observe a bf sequence of coin flips, and have to predict the probability of **H**ead at each step:
  p_0,**H**,p_1,**T**,p_2,...
- What would be a good value for p_0?
- For p_1?

What if the estimation has to be done online?

- We observe a bf sequence of coin flips, and have to predict the probability of **H**ead at each step:
  p_0,**H**,p_1,**T**,p_2,...
- What would be a good value for p_0?
- For p_1?
- Laplace Law of succession

$$\frac{\#\mathbf{H}+1}{n+2}$$

What if the estimation has to be done online?

- We observe a bf sequence of coin flips, and have to predict the probability of **H**ead at each step:
  p_0,**H**,p_1,**T**,p_2,. . .
- What would be a good value for p_0?
- For p_1?
- Laplace Law of succession

$$\frac{\#\mathbf{H}+1}{n+2}$$

- Turns out that a better rule is

$$\frac{\#\mathbf{H}+1/2}{n+1}$$

  Krichevsky and Trofimov, 1981

What if the estimation has to be done online?

- We observe a bf sequence of coin flips, and have to predict the probability of **H**ead at each step:
  p_0,**H**,p_1,**T**,p_2,...
- What would be a good value for p_0?
- For p_1?
- Laplace Law of succession
  $$\frac{\#\mathbf{H}+1}{n+2}$$
- Turns out that a better rule is
  $$\frac{\#\mathbf{H}+1/2}{n+1}$$
  Krichevsky and Trofimov, 1981
- Why?

What if the estimation has to be done online?

- We observe a bf sequence of coin flips, and have to predict the probability of **H**ead at each step: p_0,**H**,p_1,**T**,p_2,...
- What would be a good value for p_0?
- For p_1?
- Laplace Law of succession
$$\frac{\#\mathbf{H} + 1}{n + 2}$$
- Turns out that a better rule is
$$\frac{\#\mathbf{H} + 1/2}{n + 1}$$
Krichevsky and Trofimov, 1981
- Why?
- What does “better” mean?

Next time

Analysis of the Hedge Algorithm.

- slides in talk1.handout.pdf on:
  `https://github.com/yoavfreund/2020-Online-Learni`

Next time

Analysis of the Hedge Algorithm.

- slides in talk1.handout.pdf on:
  https://github.com/yoavfreund/2020-Online-Learni
- **Grading:** 4HW (40%), Midterm (20%) Final (40%)

Next time

Analysis of the Hedge Algorithm.

- slides in talk1.handout.pdf on:
  https://github.com/yoavfreund/2020-Online-Learni
- **Grading:** 4HW (40%), Midterm (20%) Final (40%)
- See you on Thursday!