Statistical Learning Techniques Based on Worst-case On-line Algorithms

Claudio Gentile
DICOM
Universita' dell'Insubria, Italy
claudio.gentile@uninsubria.it

July 21st, 2004

Content of this tutorial

- Worst-case on-line setting:
  - Learning setting, examples
  - Learning with expert advice (Bayes voting)
  - Learning linear-threshold functions
  - Learning ~~regression~~ functions

focus on
BINARY
classification

- Statistical batch setting:
  - ~~Expectation~~ analysis
  - Data-dependent analysis

(Worst-case) on-Line Learning

[L,A]

experts

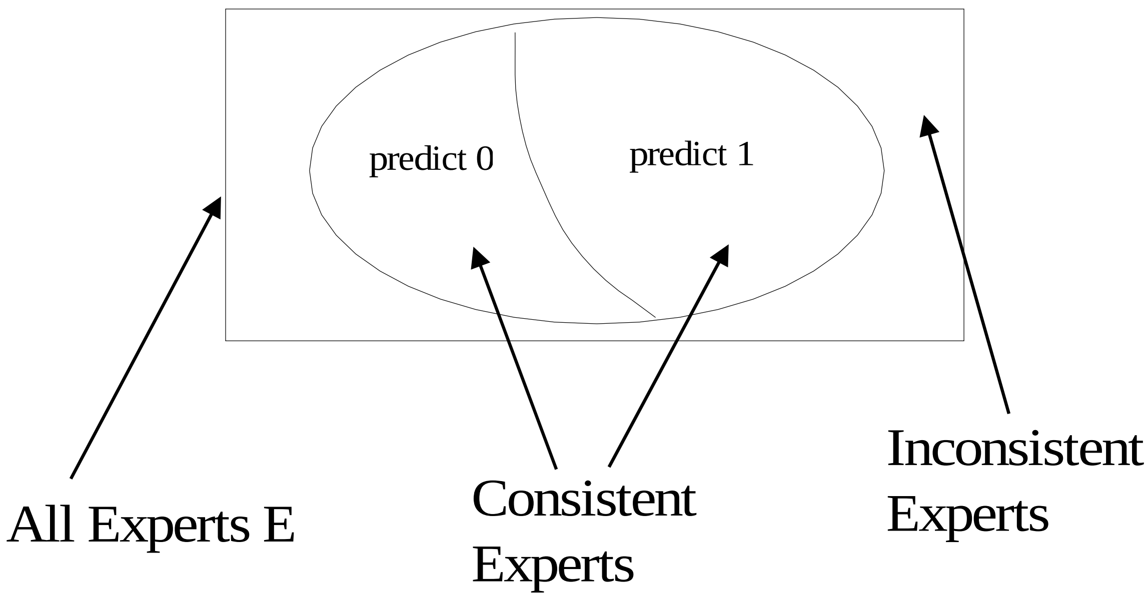
	E_1	E_2	E_3	...	E_n	pred.	true lab.	loss
day 1	1	1	-1	...	-1	-1	1	1
day 2	1	-1	1	...	-1	1	-1	1
day 3	-1	1	1	...	1	1	1	0
day t	$z_{t,1}$	$z_{t,2}$	$z_{t,3}$	...	$z_{t,n}$	$\widehat{y}_t$	y_t	$\frac{1}{2}\lvert y_t - \widehat{y}_t\rvert$

On-line protocol

For $t = 1, ..., T$ do:

	Get vector	$\boldsymbol{z}_t \in \{-1, 1\}^n$
	Predict	$\widehat{y}_t \in \{-1, 1\}$
	Get label	$y_t \in \{-1, 1\}$
	Incur loss	$\frac{1}{2}\lvert y_t - \widehat{y}_t\rvert \in \{0, 1\}$

Halving Algorithm [BF]



- Predicts with majority
- If mistake is made then number of consistent Experts is (at least) halved

A run of the Halving Algorithm (HA)

E_1	E_2	E_3	E_4	E_5	E_6	E_7	E_8	*majo rity*	*true label*	loss
1	1	-1	-1	1	1	-1	-1	1	-1	1
		-1	1			1	1	1	1	0
			1			-1	-1	-1	1	1
			$\uparrow$							
			consistent							

$\forall$ sequence with k consistent experts (out of n)

HA makes $m \leq \log_2(n/k)$ mistakes: $n/2^m = k$

Learning with expert advice/1

What if no expert E_i is consistent?

Sequence of examples $S = (\boldsymbol{z}_1, y_1), \ldots, (\boldsymbol{z}_T, y_T)$

- $L_A(S)$ be the total loss of alg. A on sequence S
- $L_i(S)$ be the total loss of i-th expert E_i on S

Want bounds of the form:

$$\forall S : \quad L_A(S) \;\leq\; a \; \min_i L_i(S) + b \; \log(n)$$

where a, b are constants

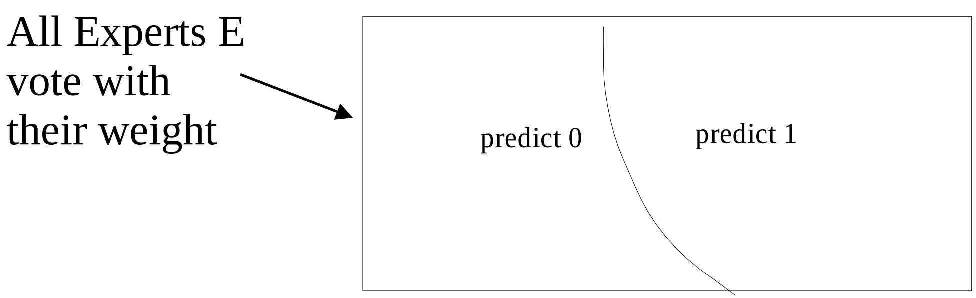
Bounds loss of algorithm relative to loss of best expert

Learning with expert advice/2

Can't wipe out experts!

Keep one weight per expert

The Weighted Majority Algorithm [LW]



- Predicts with larger side
- Weights of wrong experts are slashed by $\beta \in [0,1)$ factor

Learning with expert advice/3
More general/1

Several loss functions:

absolute	$L(y, \widehat{y})$	$=$	$\frac{1}{2}\lvert y - \widehat{y}\rvert$
square	$L(y, \widehat{y})$	$=$	$\frac{1}{2}(y - \widehat{y})^2$
entropic	$L(y, \widehat{y})$	$=$	$\frac{1+y}{2} \ln \frac{1+y}{1+\widehat{y}} + \frac{1-y}{2} \ln \frac{1-y}{1-\widehat{y}}, \quad y, \widehat{y} \in [-1, 1]$

One weight per expert: [V]

$$w_{t,i} = \beta^{\ L_{t,i}} = e^{-\eta\ L_{t,i}},$$

$L_{t,i}$ is total loss of E_i before trial t,

η is positive learning rate

Learning with expert advice/3
More general/2

Alg. A predicts with the weighted average [KW]

$$\begin{aligned} v_{t,i} &= w_{t,i}/\sum_{i=1}^{n} w_{t,i} \quad \text{normalized weights} \\ \widehat{y}_t &= \boldsymbol{v}_t \cdot \boldsymbol{z}_t, \end{aligned}$$

where $z_{t,i} \in [-1,+1]$ is prediction of E_i in trial t

$\forall$ sequences $S = (\boldsymbol{z}_1, y_1), ..., (\boldsymbol{z}_T, y_T)$, $\boldsymbol{z}_t \in [-1,1]^n$, $y_t \in [-1,1]$

$$L_A(S) \leq \min_i \underbrace{1}_{a} L_i(S) + \underbrace{1/\eta}_{b} \ln(n)$$

Learning with expert advice/3
More general/3

$1/\eta$	dot pred	fancy
entropic	1	1
square	2	.5
hellinger	1	.71

- Improved constants of $1/\eta$ when alg. A uses fancier prediction [V]
- For 0-1 loss and absolute loss $a > 1$ (with constant η)
  Regret bounds ($a = 1$) need time-changing η [ACBG]

Learning with expert advice/4

- Weighted Majority is just a Bayes voting scheme
- Easy to combine good experts (algorithms) so that prediction alg. is almost as good as best expert
- Bounds are logarithmic in # of experts

So far:

Learning relative to best expert/component

From now on:

Learning relative to best (thresholded) linear combination of experts/components

A more general setting

Instance	Prediction of alg A	Label	Loss of alg A
$\boldsymbol{x}_1$	$\widehat{y}_1$	y_1	$L(y_1, \widehat{y}_1)$
$\vdots$	$\vdots$	$\vdots$	$\vdots$
$\boldsymbol{x}_t$	$\widehat{y}_t$	y_t	$L(y_t, \widehat{y}_t)$
$\vdots$	$\vdots$	$\vdots$	$\vdots$
$\boldsymbol{x}_T$	$\hat{y}_T$	y_T	$L(y_T, \widehat{y}_T)$
		Total Loss	$L_A(S)$

Sequence of examples $S = (\boldsymbol{x}_1, y_1), ..., (\boldsymbol{x}_T, y_T) \in \mathbb{R}^n \times \{-1, 1\}$

Comparison class $\{\boldsymbol{u}\}$

Relative loss $\quad L_A(S) - \inf_{\{\boldsymbol{u}\}} Loss_{\boldsymbol{u}}(S)$

Goal: Bound relative loss for arbitrary sequence S

Learning linear-threshold functions/1
Another run of the Halving Algorithm/1

Sequence of examples $S = (\boldsymbol{x}_1, y_1), ..., (\boldsymbol{x}_T, y_T) \in \mathbb{R}^2 \times \{-1, 1\}$

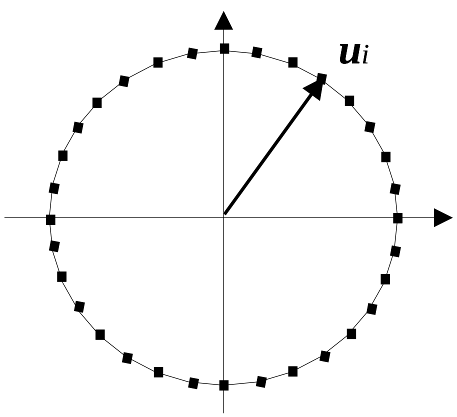
S is lin. separated by $\boldsymbol{u} \in \mathbb{R}^2 : ||\boldsymbol{u}||_2 = 1$ with margin

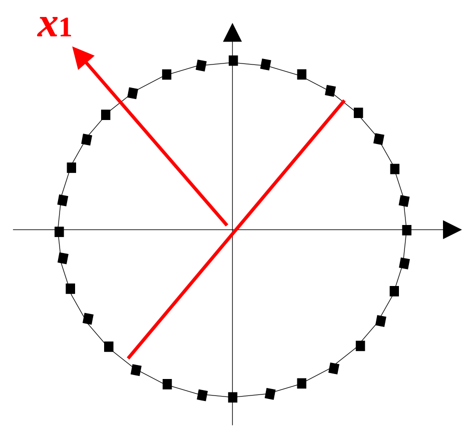
$0 < \gamma \le y_t \boldsymbol{u}^\top \boldsymbol{x}_t \; \forall t$ $\qquad R = \max_t ||\boldsymbol{x}_t||_2$

$\inf_{\{\boldsymbol{u}\}} \mathrm{Loss}_{\boldsymbol{u}}(S) = 0$

Experts:
n (large) linear−threshold functions
evenly spread over unit circle
Expert i preditcts $z_{it} = \mathrm{sgn}(\boldsymbol{u}_i^\top \boldsymbol{x}_t)$

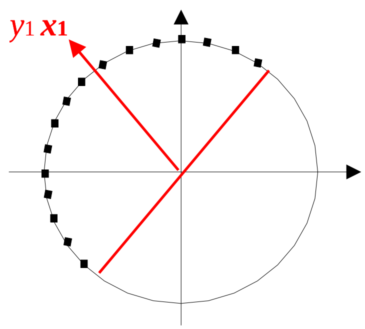
Feed experts with $\boldsymbol{x}_1$ and get expert prediction vector $\boldsymbol{z}_1$



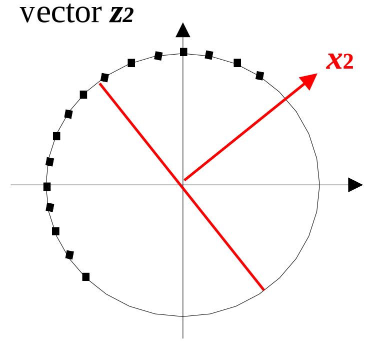


Learning linear-threshold functions/1
Another run of the Halving Algorithm/2

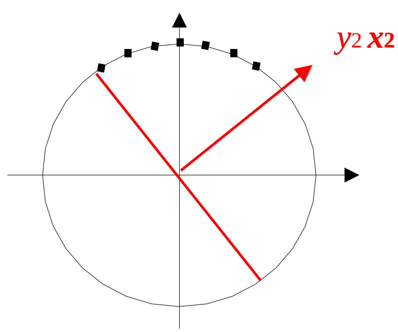
Get true label $y_1 = 1$ (mistake)
version space gets halved



Feed experts with $\boldsymbol{x}_2$
and get expert prediction
vector $\boldsymbol{z}_2$

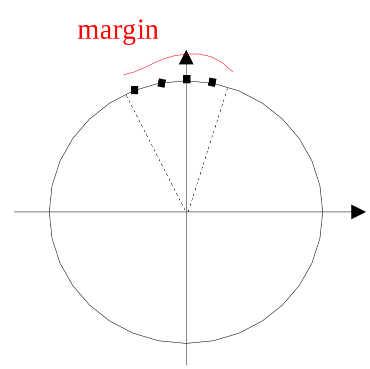


Get true label $y_2 = 1$ (mistake)
version space gets (at least) halved



...

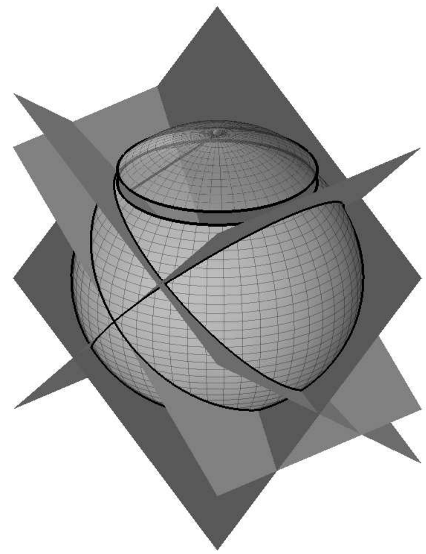
...at the end



$$m_{HA} \leq \log_2(n/k) \approx \log_2(R/\gamma) \text{ for large } n$$

Learning linear-threshold functions/1
Another run of the Halving Algorithm/3

[GH,GBNT,...]



Courtesy: R. Herbrich

For n-dim vectors:

$$m_{HA} \leq \log_2 1/\mathrm{Vol}(\mathrm{consistent}(S))$$
$$= O(n \log(R/\gamma)),$$

$R = \max_t ||\boldsymbol{x}_t||_2$

Proof: $y_t \boldsymbol{u}^\top \boldsymbol{x}_t \geq \gamma$ and $||\boldsymbol{u} - \boldsymbol{u}'||_2 < \gamma/R \implies y_t \boldsymbol{u}^\top \boldsymbol{x}_t > 0$
$\exists$ ball B of radius $\gamma/2R$: $B \subseteq \mathrm{consistent}(S)$,
$\mathrm{Vol}(B) = (\gamma/2R)^n \mathrm{Vol}(\text{unit } n\text{-sphere})$

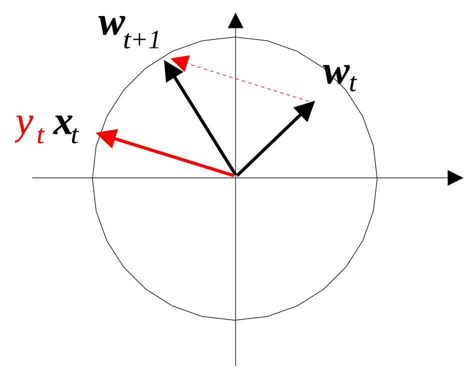
Linear dependence on n

Learning linear-threshold functions/2

The (first-order) Perceptron algorithm [Ros, ...]

Keep weight vector $\boldsymbol{w}_t \in \mathbb{R}^n$

In trial t:



- Get instance $\boldsymbol{x}_t \in \mathbb{R}^n$
- Predict with $\widehat{y}_t = \text{SGN}(\boldsymbol{w}_t^\top \boldsymbol{x}_t) \in \{-1, 1\}$
- Get label $y_t \in \{-1, 1\}$
- **If** mistake $(y_t \boldsymbol{w}_t^\top \boldsymbol{x}_t \le 0)$ **then** update $\boldsymbol{w}_{t+1} := \boldsymbol{w}_t + y_t\, \boldsymbol{x}_t$

Learning linear-threshold functions/3

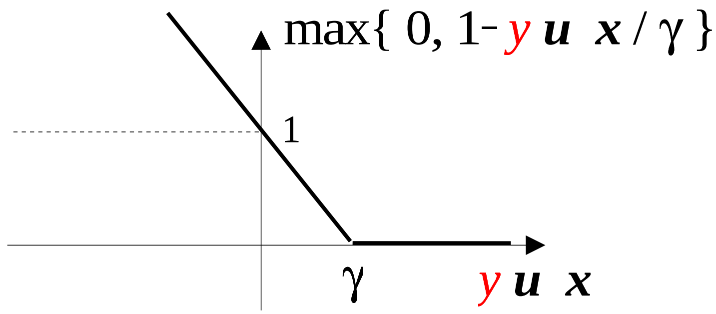
Perceptron convergence theorem/1 [Bl,No,...]

Arbitrary sequence $S = (\boldsymbol{x}_1, y_1), ..., (\boldsymbol{x}_T, y_T) \in \mathbb{R}^n \times \{-1, 1\}$

$$\#\text{ of mistakes} \leq \inf_{\gamma>0,\ ||\boldsymbol{u}||_2=1} \left(\underbrace{D_\gamma(\boldsymbol{u}; S)}_{\text{"loss" of } \boldsymbol{u}} + \frac{\sqrt{\sum_{t\in\mathcal{M}} ||\boldsymbol{x}_t||_2^2}}{\gamma} \right),$$

$\mathcal{M}$ is set of mistaken trials t,

$D_\gamma(\boldsymbol{u}; S) = \sum_{t\in\mathcal{M}} \max\{0, 1 - y_t \boldsymbol{u}^\top \boldsymbol{x}_t/\gamma\}$

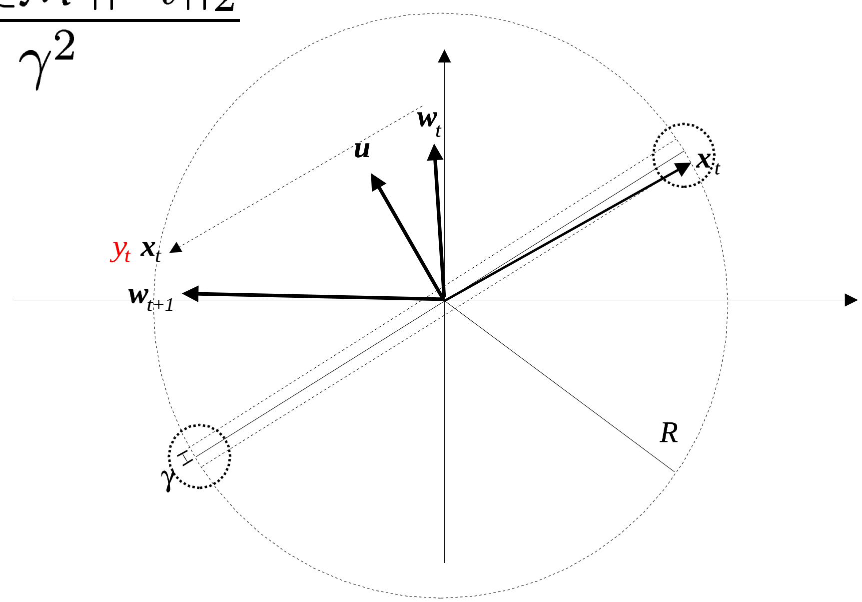


Learning linear-threshold functions/3
Perceptron convergence theorem/2

When S is separated by $\boldsymbol{u} : ||\boldsymbol{u}||_2 = 1$ with margin $\gamma \leq y_t \boldsymbol{u}^\top \boldsymbol{x}_t \; \forall t$

gets

$$\#\text{ of mistakes} \leq \frac{\max_{t \in \mathcal{M}} ||\boldsymbol{x}_t||_2^2}{\gamma^2}$$



Pointwise bound:

Depends on radius R and margin γ

Learning linear-threshold functions/4
The second-order Perceptron algorithm [CBCG]

Keep weight vector $\boldsymbol{w}_t \in \mathbb{R}^n$ and matrix S_t

In trial t:

positive parameter

- Get instance $\boldsymbol{x}_t \in \mathbb{R}^n$
- Predict with $\widehat{y}_t = \text{SGN}(\boldsymbol{w}_t^\top (aI + S_t)^{-1} \boldsymbol{x}_t) \in \{-1, 1\}$
- Get label $y_t \in \{-1, 1\}$
- **If** mistake **then** update
  - $\boldsymbol{w}_{t+1} := \boldsymbol{w}_t + y_t \, \hat{\boldsymbol{x}}_t$
  - $S_{t+1} = S_t + \hat{\boldsymbol{x}}_t \hat{\boldsymbol{x}}_t^\top$, $\qquad \hat{\boldsymbol{x}}_t = \boldsymbol{x}_t / ||\boldsymbol{x}_t||$

Turns to first-order when $a \to \infty$

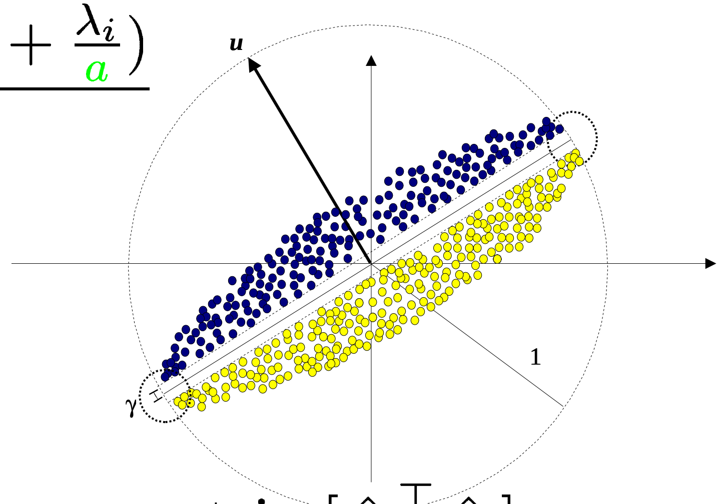
Learning linear-threshold functions/5
Second-order convergence theorem [G]

When $S = (\boldsymbol{x}_1, y_1), ..., (\boldsymbol{x}_T, y_T) \in \mathbb{R}^n \times \{-1, 1\}$

is separated by $\boldsymbol{u}$ with margin $\gamma \leq y_t \boldsymbol{u}^\top \hat{\boldsymbol{x}}_t \; \forall t$

gets

$$\# \text{ of mistakes} \leq \frac{a + \sum_{i=1}^n \ln(1 + \frac{\lambda_i}{a})}{\gamma}$$



More complicated bound
in the nonseparable case

Pointwise bound:

Depends on eigenstructure $\{\lambda_i\}$ of Gram matrix $[\hat{\boldsymbol{x}}_i{}^\top \hat{\boldsymbol{x}}_j]_{i,j \in \mathcal{M}}$

and linearly on inverse margin γ

Learning linear-threshold functions/6

Kernel Perceptron [FS,...]

Keep pool of "support vectors" $\mathcal{M}_t$

In trial t:

- Get instance $\boldsymbol{x}_t \in \mathbb{R}^n$
- Predict with $\widehat{y}_t = \textsc{sgn}(\sum_{i \in \mathcal{M}_t} y_i \, K(\boldsymbol{x}_i, \boldsymbol{x}_t)) \in \{-1, 1\}$
- Get label $y_t \in \{-1, 1\}$
- **If** mistake **then** update $\mathcal{M}_{t+1} := M_t \cup \{t\}$

Learning linear-threshold functions/7
Kernel Perceptron convergence theorem/1

Arbitrary sequence $S = (\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_T, y_T) \in \mathbb{R}^n \times \{-1, 1\}$

$$\# \text{ of mist.} \leq \inf_{\gamma > 0,\ f \in H_K,\, ||f||=1} \left(\underbrace{D_\gamma(f; S)}_{\text{"loss" of } f} + \frac{\sqrt{\sum_{t \in \mathcal{M}} K(\boldsymbol{x}_t, \boldsymbol{x}_t)}}{\gamma} \right)$$

$H_K = \{f(\cdot) = \sum_{t=1}^{T} \alpha_t K(\boldsymbol{x}_t, \cdot) \,:\, \alpha_t \in \mathbb{R}\}$,

$\mathcal{M}$ is set of mistaken trials t,

$D_\gamma(f; S) = \sum_{t \in \mathcal{M}} \max\{0, 1 - y_t f(\boldsymbol{x}_t)/\gamma\}$

Separable case:

$$\# \text{ of mistakes} \leq \frac{\max_{t \in \mathcal{M}} K(\boldsymbol{x}_t, \boldsymbol{x}_t)}{\gamma^2}$$

Learning linear-threshold functions/8

Kernel Second-order Perceptron [CBCG]

Keep pool of "support vectors" $\mathcal{M}_t$

In trial t:

$$\begin{bmatrix} \vdots \\ y_i \\ \vdots \end{bmatrix} \; i \in M_t \qquad\qquad \begin{bmatrix} \vdots \\ \hat{K}(x_i, x_t) \\ \vdots \end{bmatrix} \; i \in M$$

- Get instance $\boldsymbol{x}_t \in \mathbb{R}^n$
- Predict with $\widehat{y}_t =$

$$\text{SGN}\left(\boldsymbol{y}_t^\top \left(a\, I + \underbrace{[\hat{K}(\boldsymbol{x}_i, \boldsymbol{x}_j)]_{i,j \in \mathcal{M}_t}}_{\textit{current Gram matrix}} \right)^{-1} \boldsymbol{v}_t \right) \in \{-1, 1\},$$

- Get label $y_t \in \{-1, 1\}$
- **If** mistake **then** update $\mathcal{M}_{t+1} := M_t \cup \{t\}$

Learning linear-threshold functions/9
Kernel Second-order convergence theorem

When $S = (\boldsymbol{x}_1, y_1), ..., (\boldsymbol{x}_T, y_T) \in \mathbb{R}^n \times \{-1, 1\}$
is separated by $f(\cdot) = \sum_{t=1}^{T} \alpha_t \hat{K}(\boldsymbol{x}_t, \cdot)$, $\alpha_t \in \mathbb{R}$,
with margin $\gamma \leq y_t f(\boldsymbol{x}_t)\ \forall t$
gets

$$\#\text{ of mist.} \leq \frac{a + \sum_i \ln(1 + \frac{\lambda_i}{a})}{\gamma},$$

λ_i is i-th eigenvalue of (normalized) kernel Gram matrix
$[\hat{K}(\boldsymbol{x}_i, \boldsymbol{x}_j)]_{i,j \in \mathcal{M}}$,

$\mathcal{M}$ is set of mistaken trials t

Learning linear-threshold functions/10
Additive algorithms

An additive algorithm (e.g. first/second-order Perceptron):

- Relies on linear algebra
- Is rotation invariant (depends on data via angles)
- Can be easily kernelized ($\boldsymbol{x}_i^\top \boldsymbol{x}_j \rightarrow K(\boldsymbol{x}_i, \boldsymbol{x}_j)$)
- Has no bias for axes-parallel directions (no feature selection)

Learning linear-threshold functions/11
Nonadditive algorithms

- No linear algebra
- No rotation invariance
- Harder to kernelize
- Bias for sparse solutions (built-in feature selection)

Example: p-norm algorithms

Learning linear-threshold functions/12

p-norm algs [GLS,GL,G]

Keep weight vector $\boldsymbol{w}_t \in \mathbb{R}^n$

In trial t: $\boldsymbol{f}(\cdot) = \nabla \frac{1}{2}||\cdot||_p^2$, $p \geq 2$

- Get instance $\boldsymbol{x}_t \in \mathbb{R}^n$
- Predict $\widehat{y}_t = \text{SGN}(\boldsymbol{f}(\boldsymbol{w}_t)^\top \boldsymbol{x}_t) \in \{-1, 1\}$
- Get label $y_t \in \{-1, 1\}$
- **If** mistake **then** update $\boldsymbol{w}_{t+1} := \boldsymbol{w}_t + y_t\, \boldsymbol{x}_t$

Notice:

- $p = 2$ gets (first-order) Perceptron
- $p = O(\ln n)$ gets Weighted Majority/Winnow [L,LW]
- $2 < p < O(\ln n)$ interpolates between the two extremes

Learning linear-threshold functions/13

p-norm Perceptron convergence theorem/1

[GLS,GL,G]

Arbitrary sequence $S = (\boldsymbol{x}_1, y_1), ..., (\boldsymbol{x}_T, y_T) \in \mathbb{R}^n \times \{-1, 1\}$

$$\#\text{ mistakes} \le \inf_{\gamma>0,\ ||\boldsymbol{u}||_q=1} \left(\underbrace{D_\gamma(\boldsymbol{u}; S)}_{\text{"loss" of } \boldsymbol{u}} + \frac{\sqrt{(p-1)\sum_{t\in\mathcal{M}} ||\boldsymbol{x}_t||_p^2}}{\gamma} \right)$$

$\mathcal{M}$ is set of mistaken trials t,

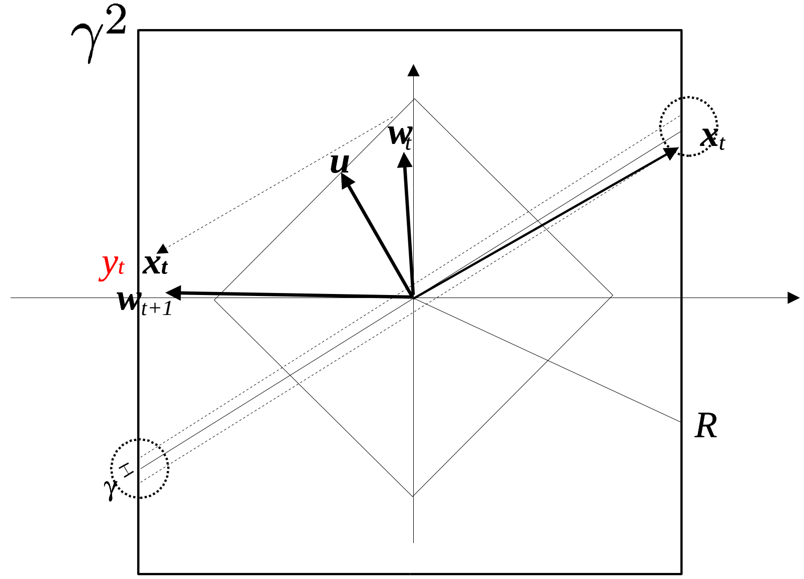
$D_\gamma(\boldsymbol{u}; S) = \sum_{t\in\mathcal{M}} \max\{0, 1 - y_t \boldsymbol{u}^\top \boldsymbol{x}_t/\gamma\}$

Learning linear-threshold functions/13
p-norm Perceptron convergence theorem/2

When S is separated by $\boldsymbol{u} : ||\boldsymbol{u}||_q = 1$ with margin
$\gamma \le y_t \boldsymbol{u}^\top \boldsymbol{x}_t \ \forall t \quad \underbrace{(1/p + 1/q = 1)}_{\text{dual norms}}$

gets

$$\# \text{ of mistakes} \le (p-1) \frac{\max_{t \in \mathcal{M}} ||\boldsymbol{x}_t||_p^2}{\gamma^2}$$



Pointwise bound:

Depends on p-norm radius R

and (q-norm) margin γ

Learning linear-threshold functions/14
p-norm algorithms with kernels/1 (wild slide) [G]

$$\boldsymbol{x} = \begin{matrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{matrix} \Rightarrow \Phi(\boldsymbol{x}) = \begin{matrix} 1 \\ x_1 \\ x_2 \\ \vdots \\ x_n \\ x_1\, x_2 \\ \vdots \\ x_1\, x_2 \ldots x_n \end{matrix}$$

$$K(\boldsymbol{x}, \boldsymbol{y}) = \Phi(\boldsymbol{x})^\top \Phi(\boldsymbol{y})$$

$$= \prod_{i=1}^{n} (1 + x_i\, y_i)$$

(Simple poly kernel)

Learning linear-threshold functions/14
p-norm algorithms with kernels/2 (wild slide)

p-norm hypothesis: $\boldsymbol{w} = \sum_{i \in \mathcal{M}} y_i \Phi(\boldsymbol{x}_i)$

p-norm margin: $= \boldsymbol{f}(\boldsymbol{w})^\top \Phi(\boldsymbol{x})$ $\qquad$ $\boldsymbol{f}(\boldsymbol{w}) = \boldsymbol{w}^{p-1}$

$$= \underbrace{\left(\sum_{i \in \mathcal{M}} y_i \Phi(\boldsymbol{x}_i)^\top \right)^{p-1}}_{expand!} \Phi(\boldsymbol{x})$$

Then expand polynomial and use $\Phi(\boldsymbol{x})\Phi(\boldsymbol{y}) = \Phi(\boldsymbol{x}\boldsymbol{y})$

Learning linear-threshold functions/14

p-norm algorithms with kernels/3 (wild slide)

Example: $p = 4$, $\boldsymbol{f}(\boldsymbol{w}) = \boldsymbol{w}^3$

$$\boldsymbol{w} = y_1\Phi(\boldsymbol{x}_1) + y_2\Phi(\boldsymbol{x}_2)$$

follow pattern $(a+b)^3 = a^3 + 3a^2b + 3ab^2 + b^3$

$\boldsymbol{f}(\boldsymbol{w}) =$

$$y_1{}^3\Phi^3(\boldsymbol{x}_1) + 3y_1{}^2y_2\Phi^2(\boldsymbol{x}_1)\Phi(\boldsymbol{x}_2) + 3y_1y_2{}^2\Phi(\boldsymbol{x}_1)\Phi^2(\boldsymbol{x}_2) + y_2{}^3\Phi^3(\boldsymbol{x}_2) =$$

$$y_1\Phi(\boldsymbol{x}_1^3) + 3y_2\Phi(\boldsymbol{x}_1^2)\Phi(\boldsymbol{x}_2) + 3y_1\Phi(\boldsymbol{x}_1)\Phi(\boldsymbol{x}_2^2) + y_2\Phi(\boldsymbol{x}_2^3) =$$

$$y_1\Phi(\boldsymbol{x}_1^3) + 3y_2\Phi(\boldsymbol{x}_1^2\boldsymbol{x}_2) + 3y_1\Phi(\boldsymbol{x}_1\boldsymbol{x}_2^2) + y_2\Phi(\boldsymbol{x}_2^3)$$

Then p-norm margin $\boldsymbol{f}(\boldsymbol{w})^\top\Phi(\boldsymbol{x})=$

$$y_1K(\underbrace{\boldsymbol{x}_1^3}_{SV}, \boldsymbol{x}) + 3y_2K(\underbrace{\boldsymbol{x}_1^2\boldsymbol{x}_2}_{SV}, \boldsymbol{x}) + 3y_1K(\underbrace{\boldsymbol{x}_1\boldsymbol{x}_2^2}_{SV}, \boldsymbol{x}) + y_2K(\underbrace{\boldsymbol{x}_2^3}_{SV}, \boldsymbol{x})$$

Generalization bounds/1

Given

- class $\mathcal{H}$ of ± 1 functions
- i.i.d. sequence $S = (X_1, Y_1), \ldots, (X_T, Y_T)$ over $\mathbb{R}^n \times \{-1, 1\}$,

0−1 loss
in our case

want to compute hypothesis $\widehat{H} = \widehat{H}_S$ with small risk
$\text{risk}(\widehat{H}) = \mathbb{E}_{X,Y}[loss(Y, \widehat{H}(X))]$:

$$\mathbb{P}\left(\text{risk}(\widehat{H}) \leq \inf_{h \in \mathcal{H}} \text{risk}(h) + \epsilon\right) \geq 1 - \delta$$

Generalization bounds/2: VC Uniform conv. [VC]

Key quantity is empirical risk

$$\text{risk}_{\text{emp}}(h) = \frac{1}{T}\sum_{t=1}^{T} loss(Y_t, h(X_t))$$

VC-bound: [VC,L, ...]

constant VC−dim(H)

$$\mathbb{P}\left(\sup_{h\in\mathcal{H}} |\text{risk}_{\text{emp}}(h) - \text{risk}(h)| \geq c\sqrt{\frac{d + \ln 1/\delta}{T}}\right) \leq \delta$$

$$\Longrightarrow \quad \widehat{H} = \text{arginf}_{h\in\mathcal{H}}\text{risk}_{\text{emp}}(h) \;\text{ is s.t.}$$

$$\mathbb{P}\left(\text{risk}(\widehat{H}) \leq \inf_{h\in\mathcal{H}} \text{risk}(h) + 2c\sqrt{\frac{d + \ln 2/\delta}{T}}\right) \geq 1 - \delta$$

Generalization bounds/3:
Data-dep. uniform conv./1 [B,BLM,WSTSS,BM, ...]

$$\sqrt{\frac{d + \ln 2/\delta}{T}} \quad \rightarrow \quad C_T(S) + \sqrt{\frac{\ln 1/\delta}{T}}$$

$C_T(S) = C_T(S, \mathcal{H})$

is sample statistic: $\begin{cases} \text{empirical VC-entropy [BLM,WSTSS]} \\ \text{Rademacher complexity[BM]} \\ \text{Maximum discrepancy[BBL]} \\ \ldots \end{cases}$

Stronger than VC since $C_T(S) \approx \mathbb{E}[C_T(S)] << \sqrt{d/T}$

Generalization bounds/3:
Data-dep. uniform conv./2

Others (e.g., margin-based bounds for linear-threshold functions) [AKLL,KP,LSM,SFBL, ...]

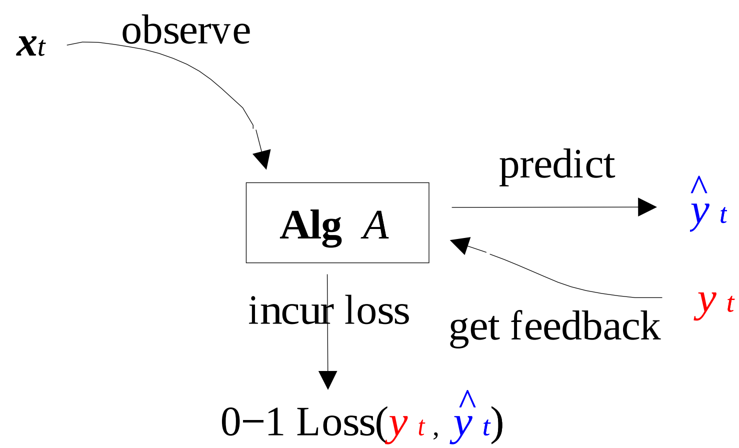
$$\mathbb{P}\left(\exists h \in \mathcal{H} : \mathrm{risk}(h) \leq \mathrm{risk}_{\mathrm{emp}}(h) + C_T(h,S) + c\sqrt{\frac{\ln 1/\delta}{T}} \right) \geq 1-\delta$$

Leave algorithmic problem of computing $h \in \mathcal{H}$ optimizing trade-off

$$\mathrm{risk}_{\mathrm{emp}}(h) \quad \text{vs} \quad C_T(h,S)$$

On-line pointwise → i.i.d. data-dependent/1

$S = (\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_T, y_T)$



Pointwise bounds so far:

$$\text{Total } \# \text{ mistakes}_A(S) \leq \text{some\_function}(S)$$

n, R, γ (Halving)

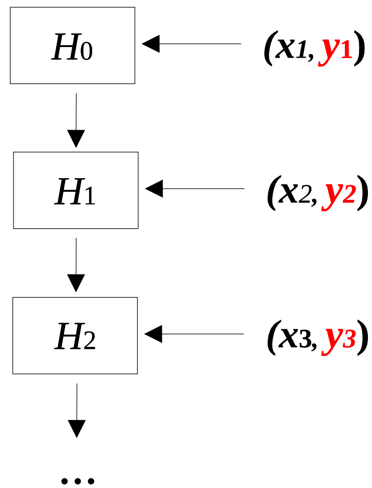
R, γ (1st Perc)

λ_i, γ (2nd Perc)

R, γ(dual) (p- norm)

...

On-line pointwise → i.i.d. data-dependent/2



Sweep through sequence of examples S just once!

Get sequence of hypotheses

$H_0, H_1, H_2, ..., H_T : \;\; H_t = H_t((\boldsymbol{x}_1, y_1), ..., (\boldsymbol{x}_t, y_t))$

Goal: Extract one with small risk

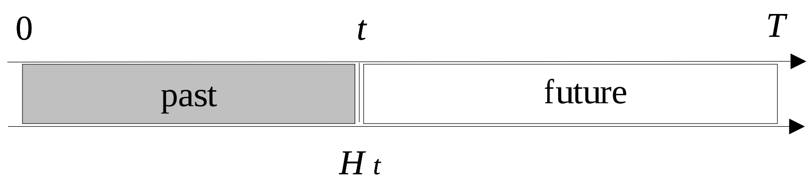
Early ref: [L] (separate test set)

On-line pointwise → i.i.d. data-dependent/3

Which one?

1. Last one: H_T (back to uniform convergence ...)
2. Average one: $\overline{H} = \frac{1}{T}\sum_{t=0}^{T} H_t \in [0,1]$
   (convex upper bound on 0-1 loss)
3. Best penalized one:

$$\text{risk}_{\text{emp}}(H_t, t+1) = \frac{1}{T-t}\sum_{i=t+1}^{T} loss(Y_i, H_t(X_i))$$



$$\widehat{H} = \operatorname*{argmin}_{t=0\ldots T-1}\left(\text{risk}_{\text{emp}}(H_t, t+1) + \underbrace{\sqrt{\frac{1}{T-t}\ln\frac{T}{\delta}}}_{\text{penalty}}\right)$$

On-line pointwise → i.i.d. data-dependent/4
Proof technique

$$\overbrace{\frac{1}{T}\underbrace{\sum_{t=1}^{T} loss(Y_t, H_{t-1}(X_t))}_{\#\text{ of mistakes}}}^{M_T} \underbrace{\approx}_{(*)} \frac{1}{T}\sum_{t=1}^{T} \mathrm{risk}(H_{t-1}) \begin{cases} \underbrace{\approx}_{(**)} \mathrm{risk}(\overline{H}) \\ \underbrace{\approx}_{(***)} \mathrm{risk}(\widehat{H}) \end{cases}$$

(*) (Hoeffding-Azuma) [DGL]

(**) bounded and convex (Jensen)

(***) general bounded (Chernoff-Hoeffding) [DGL]

On-line pointwise → i.i.d. data-dependent/5
Simplest bounds

$$\text{Convex:}\quad \mathbb{P}\left(\text{risk}(\overline{H}) \geq M_T + L\sqrt{\frac{2}{T}\ln\frac{1}{\delta}}\right) \leq \delta$$

bound on range of convex loss

$$\text{More general:}\quad \mathbb{P}\left(\text{risk}(\widehat{H}) \geq M_T + 6\sqrt{\frac{1}{T}\ln\frac{T}{\delta}}\right) \leq \delta$$

On-line pointwise → i.i.d. data-dependent/6
Some applications: plug and play/1

Recall bound on Halving Algorithm for separable case:

$$M_T \leq \frac{1}{T} O\left(n \log(R/\gamma)\right)$$

Just plug back into

$$\mathbb{P}\left(\mathrm{risk}(\widehat{H}) \geq M_T + 6\sqrt{\frac{1}{T}\ln\frac{T}{\delta}}\right) \leq \delta$$

Gets

$$\mathbb{P}\left(\mathrm{risk}(\widehat{H}) \geq \frac{1}{T} O\left(n \log(R/\gamma)\right) + 6\sqrt{\frac{1}{T}\ln\frac{T}{\delta}}\right) \leq \delta$$

Similar to [GH]

On-line pointwise → i.i.d. data-dependent/6
Some applications: plug and play/2

Recall bound on Kernel Perceptron:

$$M_T \le \inf_{\gamma>0,\; f\in H_K,\, ||f||=1} \frac{1}{T}\left(D_\gamma(f;S) + \frac{\sqrt{\sum_{t\in\mathcal{M}} K(\boldsymbol{x}_t,\boldsymbol{x}_t)}}{\gamma}\right)$$

Separable case:

$$M_T \le \frac{1}{T}\frac{\max_{t\in\mathcal{M}} K(\boldsymbol{x}_t,\boldsymbol{x}_t)}{\gamma^2}$$

Plug back into

$$\mathbb{P}\left(\mathrm{risk}(\widehat{H}) \ge M_T + 6\sqrt{\frac{1}{T}\ln\frac{T}{\delta}}\right) \le \delta$$

Similar to [BM] for SVM

On-line pointwise → i.i.d. data-dependent/6
Some applications: plug and play/3

Recall bound on Kernel Second-order Perceptron (separable case)

$$M_T \le \frac{1}{T} \frac{a + \sum_i \ln(1 + \frac{\lambda_i}{a})}{\gamma},$$

Plug into

$$\mathbb{P}\left(\mathrm{risk}(\widehat{H}) \ge M_T + 6\sqrt{\frac{1}{T}\ln\frac{T}{\delta}} \right) \le \delta$$

Try it yourself with other algs.

On-line pointwise $\rightarrow$ i.i.d. data-dependent/7 Remarks

These bounds:

- are algorithm-specific (NO uniform convergence arguments, closer in spirit to algorithmic stability/luckiness) [BE,HW,...]
- proven by simple large deviation on martingales
- refer to efficient algs (on-line, one sweep)
- are tight (I believe ...)

On-line pointwise → i.i.d. data-dependent/8

Refinements

Tigher bound 1:

$$\mathbb{P}\left(\operatorname{risk}(\widehat{H}) \geq \min_{t=0\ldots T-1}\left(M_{t,T} + 6\sqrt{\frac{1}{T-t}\ln\frac{T}{\delta}}\right)\right) \leq \delta,$$

where $M_{t,T} = \frac{1}{T-t}\sum_{i=t+1}^{T} loss(Y_i, H_{i-1}(X_i))$ (loss on suffix)

Tigher bound 2:

$$\mathbb{P}\left(\operatorname{risk}(\widehat{H}) \geq M_T + O\left(\frac{1}{T}\ln\frac{T}{\delta} + \sqrt{\frac{M_T}{T}\ln\frac{T}{\delta}}\right)\right) \leq \delta,$$

(Uses Bernstein-type inequalities for martingales) [F,DvZ]

Conclusions

- Pointwise bounds for on-line algorithms directly turn to (tight) data-dependent i.i.d. bounds
- Easy plug and play
- Resulting algs. are still as efficient as on-line (one cycle over training sequence)
- Simple proofs, algorithm-specific, no uniform convergence
- Can be generalized to regression frameworks

Disclaimer: This is by no means a complete bibliography on the subject of this tutorial

References

[L, p. 3, 27] N. Littlestone, Learning quickly when irrelevant attributes abound: a new linear threshold algorithm. *Machine Learning*, 2:285–318, 1988.

[A, p. 3] Angluin, D. (1988). Queries and concept learning. *Machine Learning*, 2:4, 319–342.

[BF, p. 4] J. M. Barzdin and R. V. Frievald. On the prediction of general recursive functions. *Soviet Math. Doklady*, 13:1224–1228, 1972.

[LW, p. 7, 27] N. Littlestone and M. K. Warmuth. The weighted majority algorithm. *Information and Computation*, 108(2):212–261, 1994. An extended abstract appeared in FOCS 89.

[V, p. 8, 10] V. Vovk. Aggregating strategies. In *Proc. 3rd Annu. Workshop on Comput. Learning Theory*, pages 371–383. Morgan Kaufmann, 1990.

[KW, p.9] J. Kivinen and M. K. Warmuth. Averaging expert predictions. In Paul Fischer and Hans Ulrich Simon, editors, *Computational Learning Theory: 4th European Conference (EuroCOLT '99)*, pages 153–167, Berlin, March 1999. Springer.

[ACBG, p.10] P. Auer, N. Cesa-Bianchi, C. Gentile, Adaptive and self-confident on-line learning algorithms. *Journal of Computer and System Science*, 64:1, 2002.

[GH, p.15, 42] R. Herbrich, T. Graepel, A PAC-Bayesian margin bound for linear classifiers. *IEEE Trans. on Information Theory*, 2002.

[GH, p.15] R. Gilad-Bachrach, T. Navot, N. Tishby. Bayes and Tukey Meet at the Center Point. In *Proc. 17th COLT*, 2004.

[Ros, p.16] Rosenblatt, F. *Principles of neurodynamics: Perceptrons and the theory of brain mechanisms.* Spartan Books, Washington, D.C., 1962.

[Bl, p. 17] Block, H. D. (1962). The perceptron: A model for brain functioning. *Reviews of Modern Physics*, 34, 123–135. Reprinted in Neurocomputing by Anderson and Rosenfeld.

[No, p. 17] Novikov, A. B. J. (1962). On convergence proofs on perceptrons. *Proc. of the Symposium on the Mathematical Theory of Automata, vol. XII* (pp. 615–622).

[CBCG, p. 19, 23] N. Cesa-Bianchi, A. Conconi, and C. Gentile. A second-order Perceptron algorithm. In *Proc. 15th COLT*, pages 121–137. LNAI 2375, Springer, 2002.

[G, p. 20, 30] C. Gentile, Unpublished. 2004

[FS, p. 21] Freund, Y., & Schapire, R. E. (1999). Large margin classification using the perceptron algorithm. *Machine Learning*, 37:3, 277–296.

[GLS, p. 27, 28] Grove, A. J., Littlestone, N., & Schuurmans, D. (2001). General convergence results for linear discriminant updates. *Machine Learning*, 43:3, 173–210.

[GL, p.27, 28] C. Gentile, N. Littlestone. The robustness of the p-norm algorithms. In *Proc. 12th Annu. Conf. on Comput. Learning Theory*, pages 1–11. ACM, 1999.

[G, p.27, 28] C. Gentile. The robustness of the p-norm algorithms. *Machine Learning*, 53:3, 2003.

[VC, p. 34] V. Vapnik and A. Chervonenkis, "On the uniform convergence of relative frequencies of events to their probabilities," *Theory of Probability and its Applications*, vol. 16, no. 2, pp. 264–280, 1971.

[B, p. 35] P. Bartlett, "The sample complexity of pattern classification with neural networks," *IEEE Transactions on Information Theory*, vol. 44, no. 2, pp. 525–536, 1998.

[BLM, p. 35] S. Boucheron, G. Lugosi, and P. Massart, "A sharp concentration inequality with applications," *Random Structures and Algorithms*, vol. 16, pp. 277–292, 2000.

[WSTSS, p. 35] R. Williamson, J. Shawe-Taylor, B. Schölkopf, and A. Smola, "Sample based generalization bounds," NeuroCOLT, Tech. Rep. NC-TR-99-055, 1999.

[BM, p. 35] P. Bartlett and S. Mendelson, "Rademacher and Gaussian complexities: Risk bounds and structural results," *Journal of Machine Learning Research*, vol. 3, pp. 463–482, 2002.

[AKLL, p. 36] A. Antos, B. Kégl, T. Linder, and G. Lugosi, "Data-dependent margin-based generalization bounds for classification," *Journal of Machine Learning Research*, vol. 3, pp. 73–98, 2002.

[KP, p. 36] V. Koltchinskii and D. Panchenko, "Empirical margin distributions and bounding the generalization error of combined classifiers," *Annals of Statistics*, vol. 30, no. 1, pp. 1–50, 2002.

[LSM, p. 36] J. Langford, M. Seeger, and N. Megiddo, "An improved predictive accuracy bound for averaging classifiers," in *Proceedings of the 18th International Conference on Machine Learning*, 2001, pp. 290–297.

[SFBL, p. 36] R. Schapire, Y. Freund, P. Bartlett, and W. Lee, "Boosting the margin: A new explanation for the effectiveness of voting methods," *The Annals of Statistics*, vol. 26, no. 5, pp. 1651–1686, 1998.

[BE, p. 45] O. Bousquet and A. Elisseff, "Stability and generalization," *Journal of Machine Learning Research*, vol. 2, pp. 499–526, 2002.

[HW, p. 45] R. Herbrich and R. Williamson, "Algorithmic luckiness," *Journal of Machine Learning Research*, vol. 3, pp. 175–212, 2002.

[L, p. 38] N. Littlestone. From on-line to batch learning. In *Proc. 2nd Annu. Workshop on Comput. Learning Theory*, pages 269–284, San Mateo, CA, 1989. Morgan Kaufmann.

[DGL, p. 40] L. Devroye, L. Giorfy, G. Lugosi. *A probabilistic theory of pattern recognition.* Springer, 1996.

[F, p. 46] D. A. Freedman. On tail probabilities for martingales. *The annals of probability*, 3:1, 1975.

[DvZ, p. 46] K. Dzhaparidze, J.H. van Zanten. On Bernstein-type inequalities for martingales. *Stochastic processes and their applications*, 93, 2001.