On-Line Learning

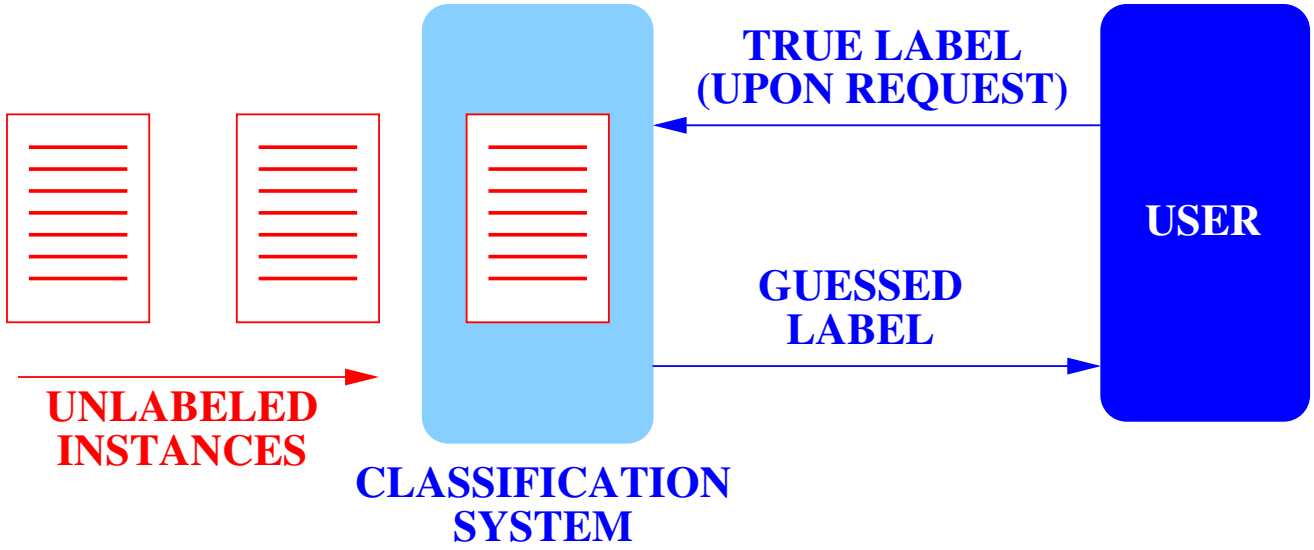
Nicolò Cesa-Bianchi

Università degli Studi di Milano

Summary

Summary

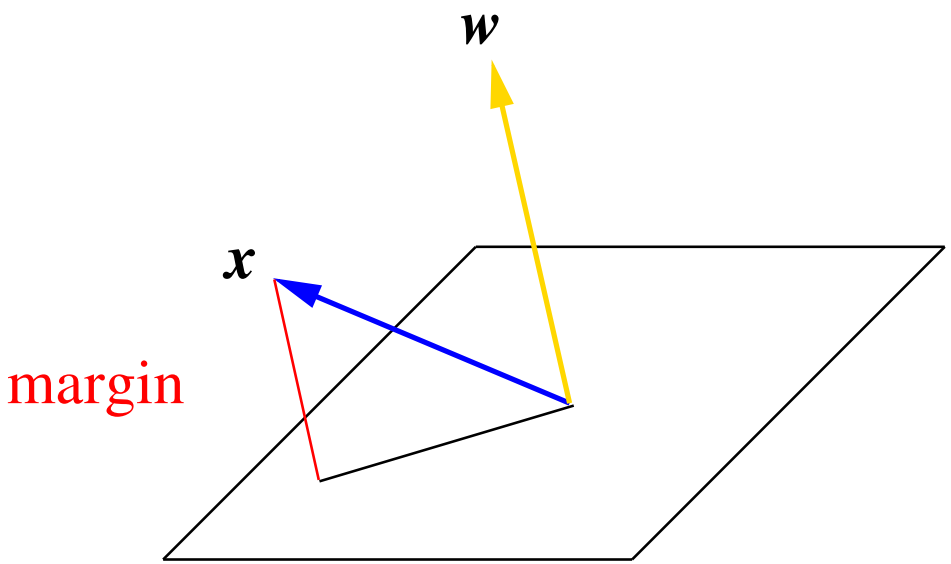
On-line classification



Linear classifiers

- Stream of data instances encoded as vectors $x_1, x_2, \cdots \in \mathbb{R}^d$
- A binary label $y_t \in \{-1, 1\}$ associated to each x_t
- A linear classifier $w_{t-1} \in \mathbb{R}^d$ predicts label y_t of x_t with

$$\widehat{y}_t = \text{SGN}(w_{t-1}^\top x_t) \qquad w_{t-1} \in \mathbb{R}^d$$



On-line learning protocol

- Learning proceeds in rounds
- Initial classifier $\boldsymbol{w}_0 \in \mathbb{R}^d$.

On-line learning protocol

- Learning proceeds in rounds
- Initial classifier $\boldsymbol{w}_0 \in \mathbb{R}^d$.

On each round $t = 1, 2, \ldots$

1. Get $\boldsymbol{x}_t \in \mathbb{R}^d$
2. Predict $\widehat{y}_t = \text{SGN}\left(\boldsymbol{w}_{t-1}^\top \boldsymbol{x}_t\right)$
3. Observe $y_t \in \{-1, +1\}$
4. Update $\boldsymbol{w}_{t-1} \to \boldsymbol{w}_t$

On-line learning protocol

- Learning proceeds in rounds
- Initial classifier $\boldsymbol{w}_0 \in \mathbb{R}^d$.

On each round $t = 1, 2, \ldots$

1. Get $\boldsymbol{x}_t \in \mathbb{R}^d$
2. Predict $\widehat{y}_t = \text{SGN}\left(\boldsymbol{w}_{t-1}^\top \boldsymbol{x}_t\right)$
3. Observe $y_t \in \{-1, +1\}$
4. Update $\boldsymbol{w}_{t-1} \to \boldsymbol{w}_t$

- Learner observes a stream of examples $(\boldsymbol{x}_t, y_t) \quad t = 1, 2, \ldots$ and produces a sequence of classifiers $\boldsymbol{w}_0, \boldsymbol{w}_1, \ldots$
- Learner's performance measure: number of classification mistakes on the stream

Remarks

- natural on certain tasks (e.g., market forecasting)
- efficient to run (scalability) and easy to code
- kernel-based learning
- strong performance guarantees (nonstochastic)
- stochastic risk bounds easy to prove (online to batch conversions)
- active learning variants when y_t comes at a cost
- tracking abilities

Summary

Perceptron algorithm

[Rosenblatt, 1952]

- Initial weight $\boldsymbol{w}_0 = (0, \ldots, 0)$.
- Update $\boldsymbol{w}_t = \begin{cases} \boldsymbol{w}_{t-1} & \text{if } \widehat{y}_t = y_t \text{ (no mistake)} \\ \boldsymbol{w}_{t-1} + y_t \boldsymbol{x}_t & \text{otherwise} \end{cases}$

Perceptron algorithm

[Rosenblatt, 1952]

- Initial weight $w_0 = (0, \ldots, 0)$.
- Update $w_t = \begin{cases} w_{t-1} & \text{if } \widehat{y}_t = y_t \text{ (no mistake)} \\ w_{t-1} + y_t x_t & \text{otherwise} \end{cases}$

Intuition

- Want positive margin: $\widehat{y}_t \neq y_t \quad \text{iff} \quad \underbrace{y_t \, w_{t-1}^\top x_t}_{\text{margin}} \leqslant 0$
- Effect of Perceptron update on margin:

$$y_t w_t^\top x_t = y_t \left(w_{t-1} + y_t x_t\right)^\top x_t = y_t w_{t-1}^\top x_t + \|x_t\|^2$$

- Margin has increased by $\|x_t\|^2$ (non-corrective update)

Linear separability

Linearly separable dataset

$(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_T, y_T)$

$$\exists \mathbf{u} \in \mathbb{R}^d \quad y_t \mathbf{u}^\top \mathbf{x}_t > 0 \quad t = 1, \ldots, T$$

Note: $\|\mathbf{u}\|$ does not matter

Linear separability

Linearly separable dataset

$(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_T, y_T)$

$$\exists \mathbf{u} \in \mathbb{R}^d \quad y_t \mathbf{u}^\top \mathbf{x}_t > 0 \quad t = 1, \ldots, T$$

Note: $\|\mathbf{u}\|$ does not matter

- Does the Perceptron find any such $\mathbf{u}$ if cycled on the stream?
- What if stream is not linearly separable?

Relative loss bound

$$\sum_t m_t(\boldsymbol{w}_{t-1}) \leqslant \inf_{\boldsymbol{u}\in\mathbb{R}^d} \sum_t m_t(\boldsymbol{u}) + \text{smaller terms}$$

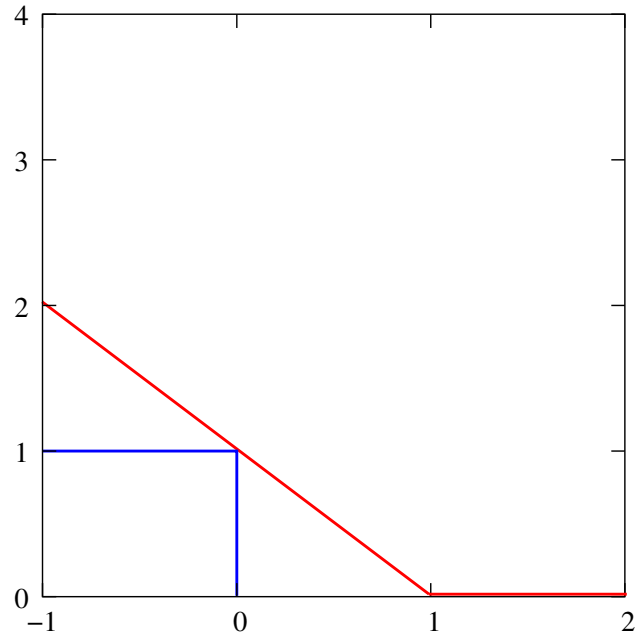
Relative loss bound

$$\sum_t m_t(\boldsymbol{w}_{t-1}) \leqslant \inf_{\boldsymbol{u}\in\mathbb{R}^d} \sum_t m_t(\boldsymbol{u}) + \text{smaller terms}$$

- Computing an hyperplane minimizing the number of misclassified examples is NP-hard

Relative loss bound

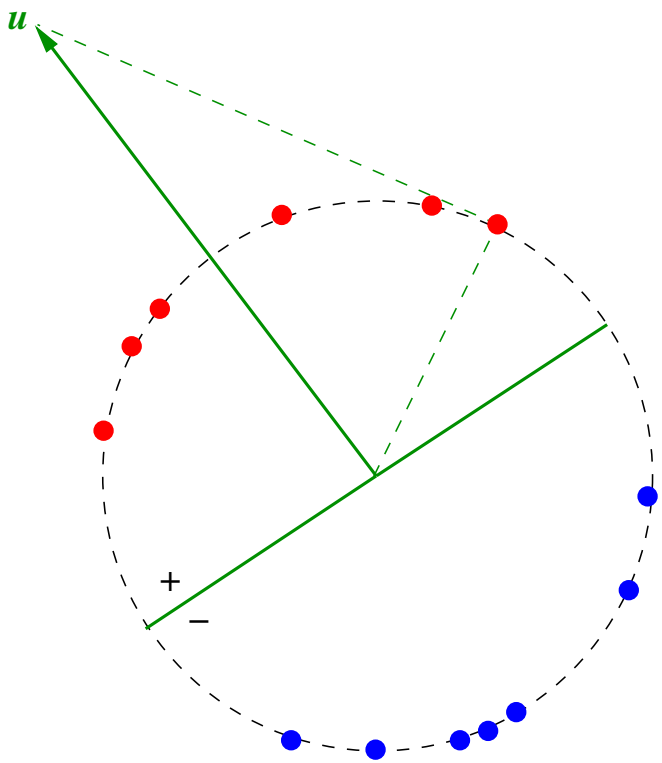
$$\sum_t m_t(w_{t-1}) \leqslant \inf_{u \in \mathbb{R}^d} \sum_t \ell_t(u) + \text{smaller terms}$$



$$\underbrace{\mathbb{I}_{\{\mathrm{SGN}(z) \neq y\}}}_{\text{mistake ind.}} \leqslant \underbrace{(1 - y\,z)_+}_{\text{hinge loss}}$$

- Computing an hyperplane minimizing the number of misclassified examples is NP-hard
- Relax goal: the hinge loss is a convex upper bound on the mistake indicator function
- Hinge loss grows linearly if margin is smaller than 1

Norm of the separator



- $\ell_t(\mathbf{u}) = [1 - y_t \mathbf{u}^\top \mathbf{x}_t]_+$
- Now $\|\mathbf{u}\|$ matters
- If the stream is hardly separable, then $\|\mathbf{u}\|$ has to be big to keep the hinge loss at zero

Analysis of Perceptron

- Any (nonseparable) stream, any $\mathbf{u} \in \mathbb{R}^d$, $\|\mathbf{x}_t\| = 1$
- Two proof techniques:

1. evolution of $\|\mathbf{u} - \mathbf{w}_t\|^2$
2. evolution of $\mathbf{u}^\top \mathbf{w}_t$

Analysis: when a mistake occurs

$$
\begin{aligned}
\mathbf{u}^\top \mathbf{w}_t &\geqslant \mathbf{u}^\top (\mathbf{w}_{t-1} + y_t \mathbf{x}_t) \\
&= \mathbf{u}^\top \mathbf{w}_{t-1} + \underbrace{y_t \mathbf{u}^\top \mathbf{x}_t}_{1 - \left(1 - y_t \mathbf{u}^\top \mathbf{x}_t\right)} \\
&\geqslant \mathbf{u}^\top \mathbf{w}_{t-1} + 1 - \ell_t(\mathbf{u})
\end{aligned}
$$

Analysis: when a mistake occurs

$$
\begin{aligned}
\mathbf{u}^\top \mathbf{w}_t &\geqslant \mathbf{u}^\top (\mathbf{w}_{t-1} + y_t \mathbf{x}_t) \\
&= \mathbf{u}^\top \mathbf{w}_{t-1} + \underbrace{y_t \mathbf{u}^\top \mathbf{x}_t}_{1-\left(1-y_t \mathbf{u}^\top \mathbf{x}_t\right)} \\
&\geqslant \mathbf{u}^\top \mathbf{w}_{t-1} + 1 - \ell_t(\mathbf{u})
\end{aligned}
$$

Unwrapping the recurrence

$$
\mathbf{u}^\top \mathbf{w}_T \geqslant \mathbf{u}^\top \mathbf{w}_0 + \sum_t m_t(\mathbf{w}_{t-1}) - \sum_t \ell_t(\mathbf{u})
$$

When a mistake occurs (cont.)

$$
\begin{aligned}
\|\boldsymbol{w}_t\|^2 &\leqslant \|\boldsymbol{w}_{t-1} + y_t \boldsymbol{x}_t\|^2 \\
&= \|\boldsymbol{w}_{t-1}\|^2 + \|\boldsymbol{x}_t\|^2 + 2 \underbrace{y_t \boldsymbol{w}_{t-1} \boldsymbol{x}_t}_{\leqslant 0} \\
&\leqslant \|\boldsymbol{w}_{t-1}\|^2 + 1
\end{aligned}
$$

When a mistake occurs (cont.)

$$\begin{aligned} \|w_t\|^2 &\leqslant \|w_{t-1} + y_t x_t\|^2 \\ &= \|w_{t-1}\|^2 + \|x_t\|^2 + 2\underbrace{y_t w_{t-1} x_t}_{\leqslant 0} \\ &\leqslant \|w_{t-1}\|^2 + 1 \end{aligned}$$

Unwrapping the recurrence

$$\|w_T\|^2 \leqslant \|w_0\|^2 + \sum_t m_t(w_{t-1})$$

When a mistake occurs (cont.)

$$\begin{aligned} \|w_t\|^2 &\leqslant \|w_{t-1} + y_t x_t\|^2 \\ &= \|w_{t-1}\|^2 + \|x_t\|^2 + 2\underbrace{y_t w_{t-1} x_t}_{\leqslant 0} \\ &\leqslant \|w_{t-1}\|^2 + 1 \end{aligned}$$

Unwrapping the recurrence

$$\|w_T\|^2 \leqslant \|w_0\|^2 + \sum_t m_t(w_{t-1})$$

Use $u^\top w_T \leqslant \|u\| \, \|w\|_T$ and solve for $\sum_t m_t(w_{t-1})$

The relative mistake bound

Generalized Perceptron convergence theorem

$$\sum_t m_t(\boldsymbol{w}_{t-1}) \leqslant \inf_{\boldsymbol{u}\in\mathbb{R}^d} \sum_t \ell_t(\boldsymbol{u}) + \|\boldsymbol{u}\|^2 + \|\boldsymbol{u}\| \sqrt{\sum_t \ell_t(\boldsymbol{u})}$$

The relative mistake bound

Generalized Perceptron convergence theorem

$$\sum_t m_t(w_{t-1}) \leqslant \inf_{u \in \mathbb{R}^d} \sum_t \ell_t(u) + \|u\|^2 + \|u\| \sqrt{\sum_t \ell_t(u)}$$

Remarks

- Oracle inequality for individual sequences
- If cycled over a linear separable $(x_1, y_1), \ldots, (x_T, y_T)$ converges to some linear separator after at most $\|u^*\|^2$ updates, where u^* is the SVM hyperplane $\min \|u\|^2 \quad \text{s.t.} \quad y_t u^\top x_t \geqslant 1 \quad \forall t$
- This implies that there always exists a linear separator expressible as a linear combination of at most $\|u^*\|^2$ supports

Summary

Aggressive updates: Hildreth algorithm (1957)

- What if we want to enforce a large margin?

Aggressive updates: Hildreth algorithm (1957)

- What if we want to enforce a large margin?
- If $y_t \boldsymbol{w}_{t-1}^\top \boldsymbol{x}_t < 1$ then update $\boldsymbol{w}_{t-1} \to \boldsymbol{w}_t$ so that $y_t \boldsymbol{w}_t^\top \boldsymbol{x}_t \geqslant 1$

Aggressive updates: Hildreth algorithm (1957)

- What if we want to enforce a large margin?
- If $y_t \boldsymbol{w}_{t-1}^\top \boldsymbol{x}_t < 1$ then update $\boldsymbol{w}_{t-1} \to \boldsymbol{w}_t$ so that $y_t \boldsymbol{w}_t^\top \boldsymbol{x}_t \geqslant 1$
- Guess form of solution: $\boldsymbol{w}_t = \boldsymbol{w}_{t-1} + \eta_t y_t \boldsymbol{x}_t$

Aggressive updates: Hildreth algorithm (1957)

- What if we want to enforce a large margin?
- If $y_t \boldsymbol{w}_{t-1}^\top \boldsymbol{x}_t < 1$ then update $\boldsymbol{w}_{t-1} \to \boldsymbol{w}_t$ so that $y_t \boldsymbol{w}_t^\top \boldsymbol{x}_t \geqslant 1$
- Guess form of solution: $\boldsymbol{w}_t = \boldsymbol{w}_{t-1} + \eta_t y_t \boldsymbol{x}_t$
- $y_t \left(\boldsymbol{w}_{t-1} + \eta_t y_t \boldsymbol{x}_t\right)^\top \boldsymbol{x}_t \geqslant 1$ iff $\eta_t = \dfrac{1}{\|\boldsymbol{x}_t\|^2}\left(1 - y_t \boldsymbol{w}_{t-1}^\top \boldsymbol{x}_t\right)$

Aggressive updates: Hildreth algorithm (1957)

- What if we want to enforce a large margin?
- If $y_t \boldsymbol{w}_{t-1}^\top \boldsymbol{x}_t < 1$ then update $\boldsymbol{w}_{t-1} \to \boldsymbol{w}_t$ so that $y_t \boldsymbol{w}_t^\top \boldsymbol{x}_t \geqslant 1$
- Guess form of solution: $\boldsymbol{w}_t = \boldsymbol{w}_{t-1} + \eta_t y_t \boldsymbol{x}_t$
- $y_t \left(\boldsymbol{w}_{t-1} + \eta_t y_t \boldsymbol{x}_t\right)^\top \boldsymbol{x}_t \geqslant 1$ iff $\eta_t = \dfrac{1}{\|\boldsymbol{x}_t\|^2}\left(1 - y_t \boldsymbol{w}_{t-1}^\top \boldsymbol{x}_t\right)$
- Equivalent to $\eta_t = \dfrac{\ell_t(\boldsymbol{w}_{t-1})}{\|\boldsymbol{x}_t\|^2}$

Aggressive updates: Hildreth algorithm (1957)

- What if we want to enforce a large margin?
- If $y_t \boldsymbol{w}_{t-1}^\top \boldsymbol{x}_t < 1$ then update $\boldsymbol{w}_{t-1} \to \boldsymbol{w}_t$ so that $y_t \boldsymbol{w}_t^\top \boldsymbol{x}_t \geqslant 1$
- Guess form of solution: $\boldsymbol{w}_t = \boldsymbol{w}_{t-1} + \eta_t y_t \boldsymbol{x}_t$
- $y_t \left(\boldsymbol{w}_{t-1} + \eta_t y_t \boldsymbol{x}_t\right)^\top \boldsymbol{x}_t \geqslant 1$ iff $\eta_t = \dfrac{1}{\|\boldsymbol{x}_t\|^2}\left(1 - y_t \boldsymbol{w}_{t-1}^\top \boldsymbol{x}_t\right)$
- Equivalent to $\eta_t = \dfrac{\ell_t(\boldsymbol{w}_{t-1})}{\|\boldsymbol{x}_t\|^2}$
- This is also the solution of the first-order SVM

$$\operatorname*{argmin}_{\boldsymbol{w}} \frac{1}{2}\|\boldsymbol{w} - \boldsymbol{w}_{t-1}\|^2 \quad \text{s.t.} \quad y_t \boldsymbol{w}^\top \boldsymbol{x}_t \geqslant 1$$

Aggressive updates: Hildreth algorithm (1957)

- What if we want to enforce a large margin?
- If $y_t \boldsymbol{w}_{t-1}^\top \boldsymbol{x}_t < 1$ then update $\boldsymbol{w}_{t-1} \to \boldsymbol{w}_t$ so that $y_t \boldsymbol{w}_t^\top \boldsymbol{x}_t \geqslant 1$
- Guess form of solution: $\boldsymbol{w}_t = \boldsymbol{w}_{t-1} + \eta_t y_t \boldsymbol{x}_t$
- $y_t \left(\boldsymbol{w}_{t-1} + \eta_t y_t \boldsymbol{x}_t\right)^\top \boldsymbol{x}_t \geqslant 1$ iff $\eta_t = \dfrac{1}{\|\boldsymbol{x}_t\|^2}\left(1 - y_t \boldsymbol{w}_{t-1}^\top \boldsymbol{x}_t\right)$
- Equivalent to $\eta_t = \dfrac{\ell_t(\boldsymbol{w}_{t-1})}{\|\boldsymbol{x}_t\|^2}$
- This is also the solution of the first-order SVM

$$\operatorname*{argmin}_{\boldsymbol{w}} \frac{1}{2}\|\boldsymbol{w} - \boldsymbol{w}_{t-1}\|^2 \quad \text{s.t.} \quad y_t \boldsymbol{w}^\top \boldsymbol{x}_t \geqslant 1$$

- Geometrically, we project $\boldsymbol{w}_{t-1}$ onto the hyperplane

$$\left\{\boldsymbol{w} \in \mathbb{R}^d : y_t \boldsymbol{w}^\top \boldsymbol{x}_t = 1\right\}$$

Analysis for linearly separable streams

Recall

$\ell_t(\boldsymbol{w}) = [1 - y_t \boldsymbol{w}^\top \boldsymbol{x}_t]_+$, which implies $\ell_t(\boldsymbol{w}_t) = 0$

Analysis for linearly separable streams

Recall

$\ell_t(\boldsymbol{w}) = [1 - y_t \boldsymbol{w}^\top \boldsymbol{x}_t]_+$, which implies $\ell_t(\boldsymbol{w}_t) = 0$

$$
\begin{aligned}
\ell_t(\boldsymbol{w}_{t-1}) &= \ell_t(\boldsymbol{w}_{t-1}) - \ell_t(\boldsymbol{w}_t) \\
&\leqslant 1 - y_t \boldsymbol{w}_{t-1}^\top \boldsymbol{x}_t - 1 + y_t \boldsymbol{w}_t^\top \boldsymbol{x}_t \\
&= y_t \left(\boldsymbol{w}_t - \boldsymbol{w}_{t-1}\right)^\top \boldsymbol{x}_t \\
&\leqslant \|\boldsymbol{w}_t - \boldsymbol{w}_{t-1}\| \, \|\boldsymbol{x}_t\|
\end{aligned}
$$

Analysis for linearly separable streams

Recall

$$\ell_t(w) = [1 - y_t w^\top x_t]_+, \text{ which implies } \ell_t(w_t) = 0$$

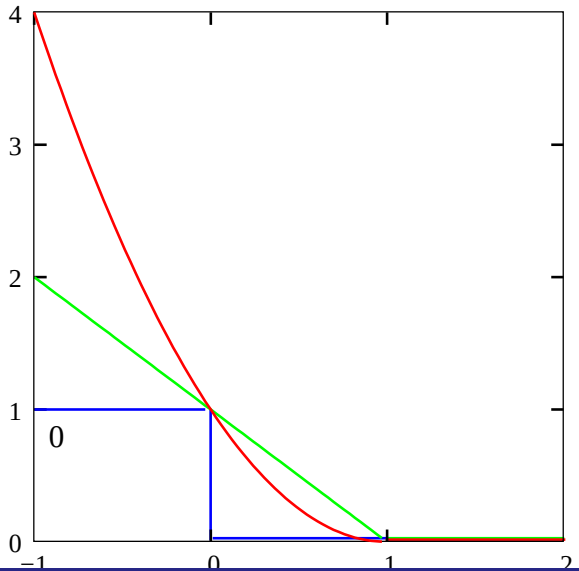
$$\begin{aligned}
\ell_t(w_{t-1}) &= \ell_t(w_{t-1}) - \ell_t(w_t) \\
&\leqslant 1 - y_t w_{t-1}^\top x_t - 1 + y_t w_t^\top x_t \\
&= y_t (w_t - w_{t-1})^\top x_t \\
&\leqslant \|w_t - w_{t-1}\| \, \|x_t\|
\end{aligned}$$

Thus we get

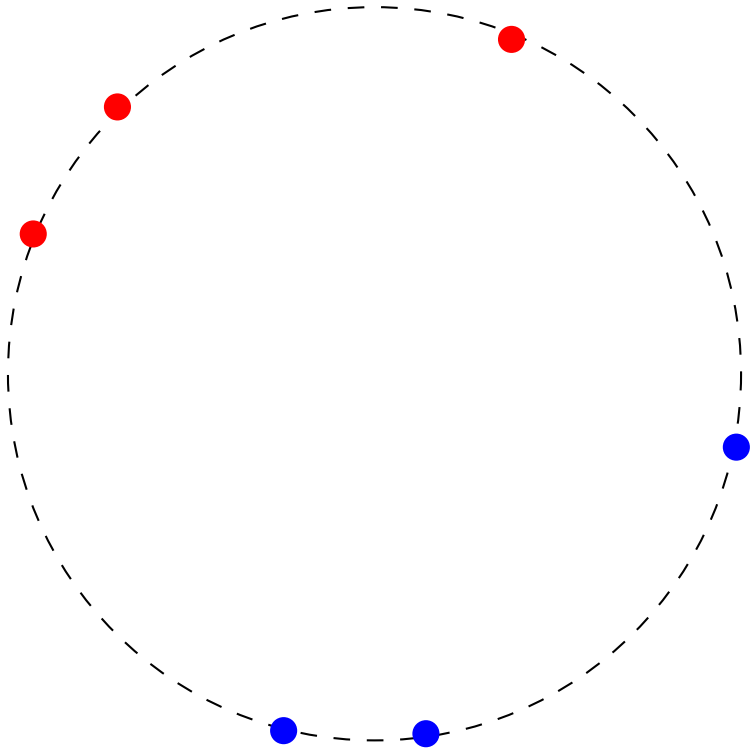
$$\ell_t(w_{t-1})^2 \leqslant \|w_t - w_{t-1}\|^2 \, \|x_t\|^2$$

Analysis (cont.)

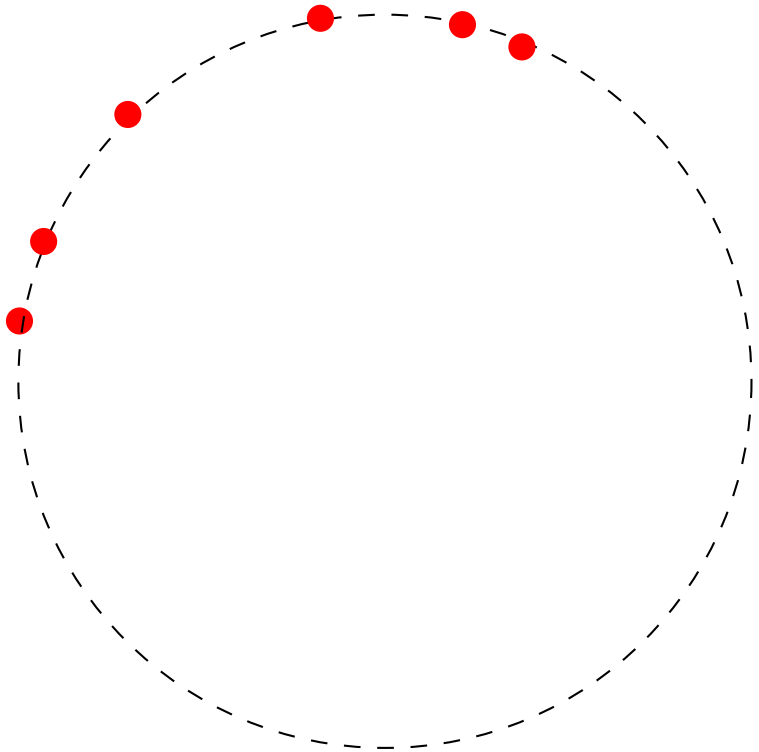
- Fix shorter separator $\mathbf{u}$ with zero hinge loss (SVM solution)
- By the law of cosines $\|\boldsymbol{w}_t - \boldsymbol{w}_{t-1}\|^2 \leqslant \|\boldsymbol{w}_{t-1} - \mathbf{u}\|^2 - \|\boldsymbol{w}_t - \mathbf{u}\|^2$
- Thus $\sum_t m_t(\boldsymbol{w}_{t-1})^2 \leqslant \sum_t \ell_t(\boldsymbol{w}_{t-1})^2 \leqslant \|\mathbf{u}\|^2 \left(\max_t \|\mathbf{x}_t\|\right)^2$
- This is a stronger result than Perceptron convergence theorem, but it does not imply convergence to SVM solution



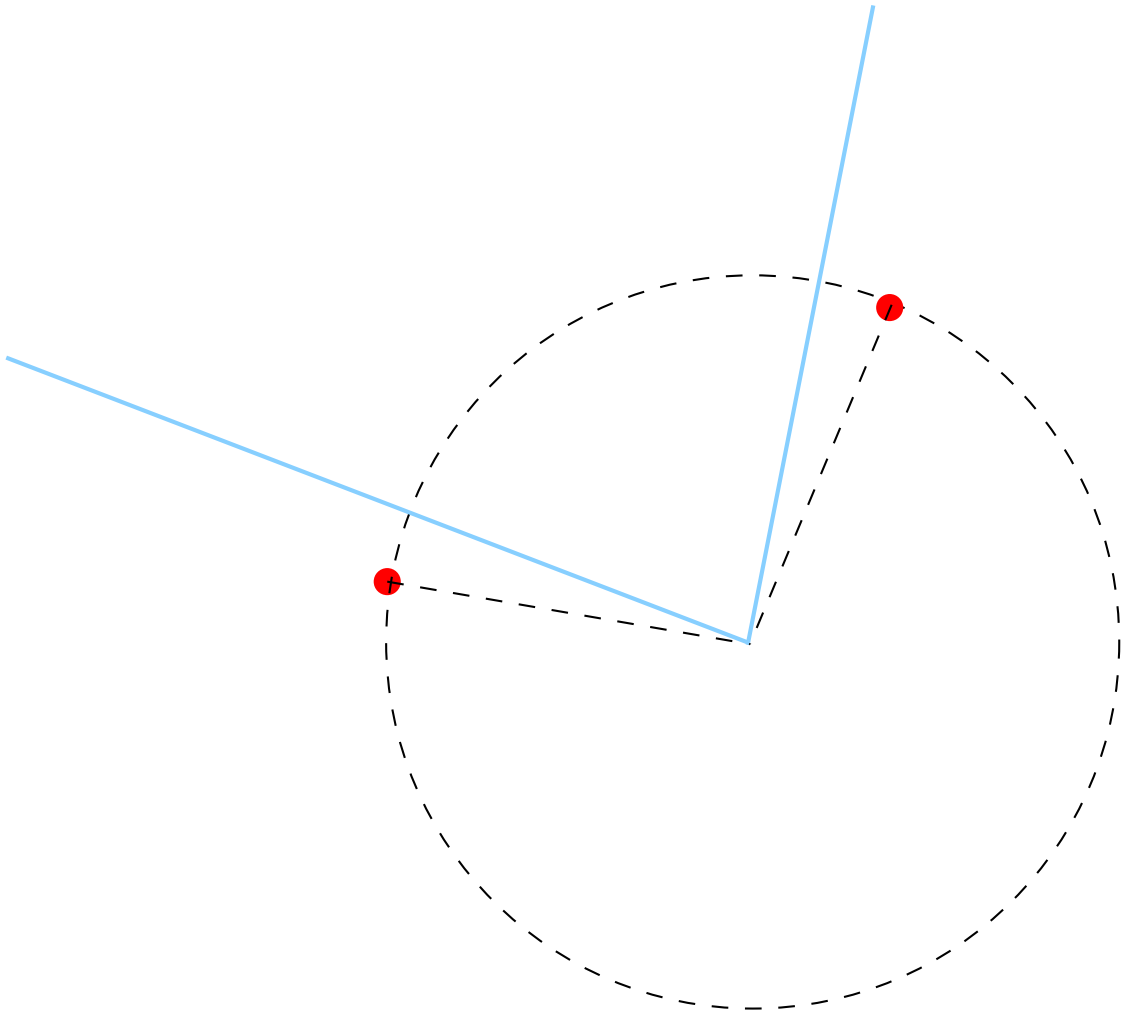
The cone of consistent hyperplanes



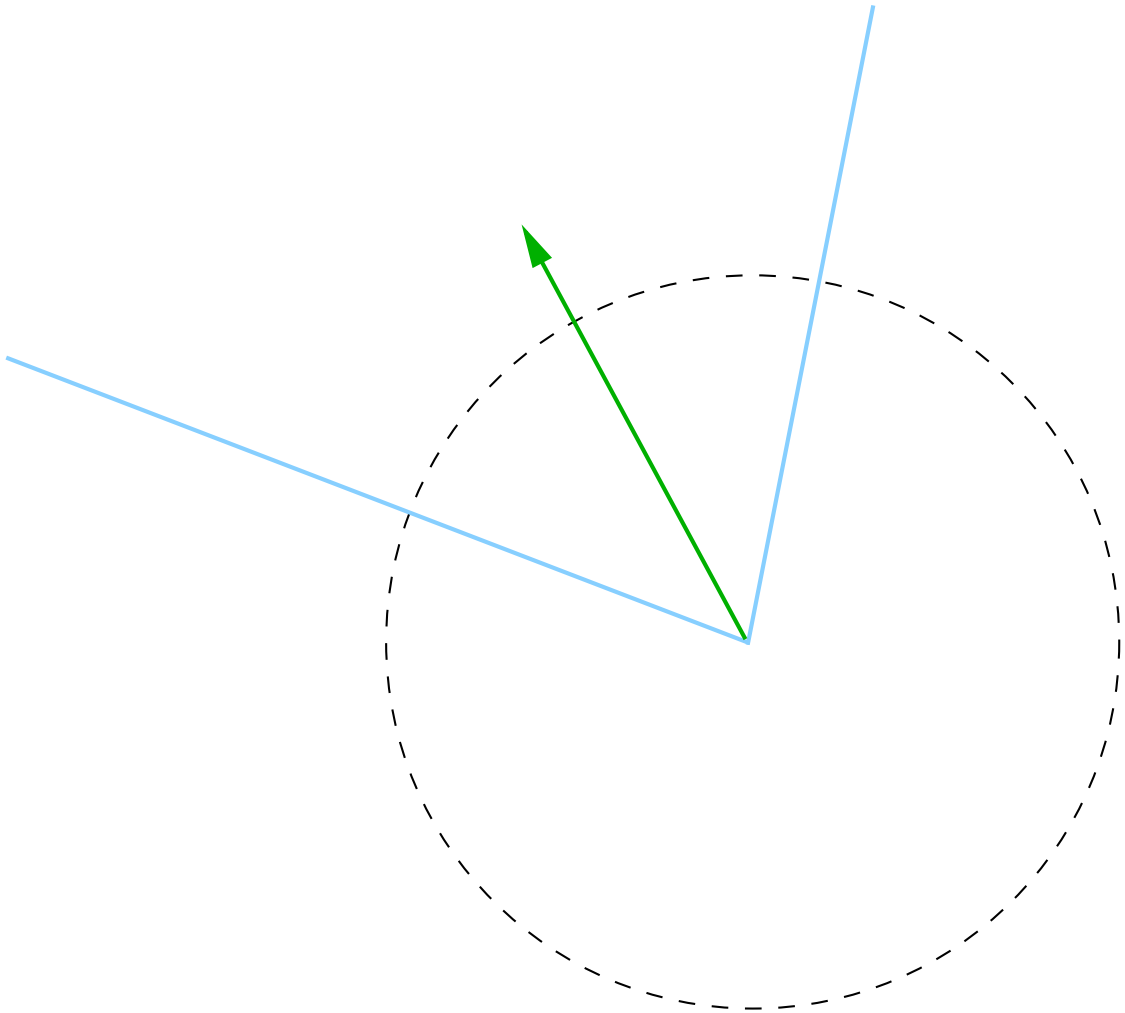
The cone of consistent hyperplanes



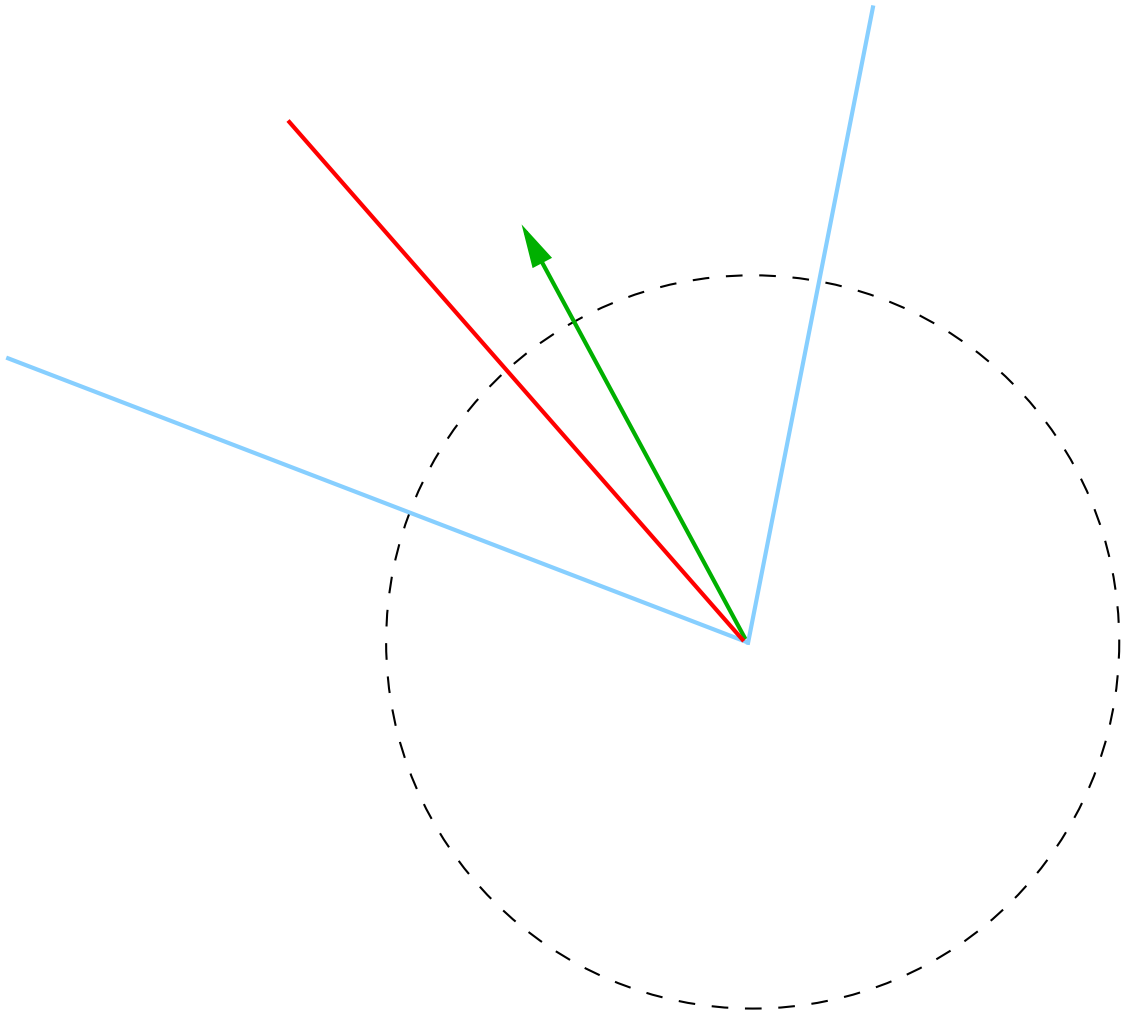
The cone of consistent hyperplanes



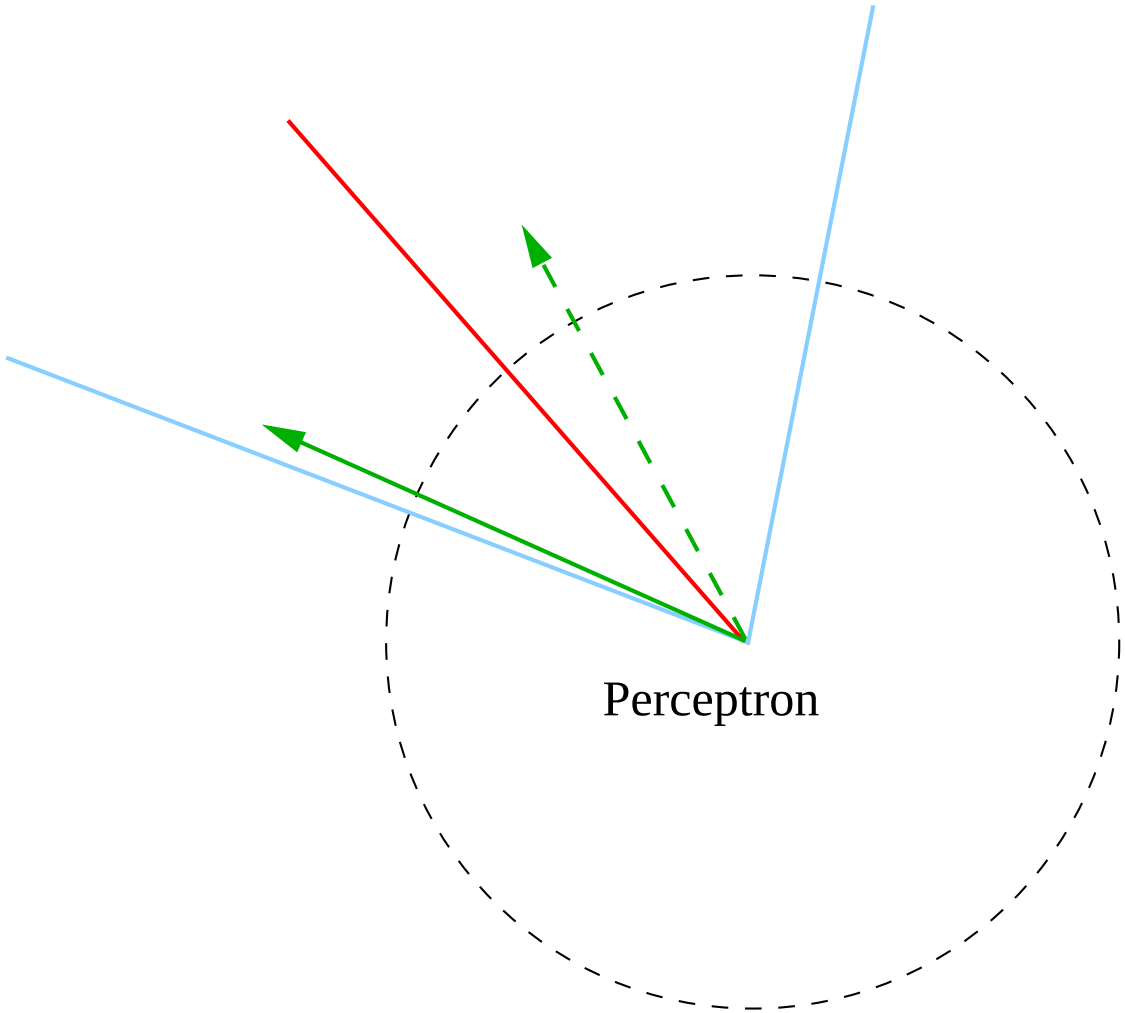
The cone of consistent hyperplanes



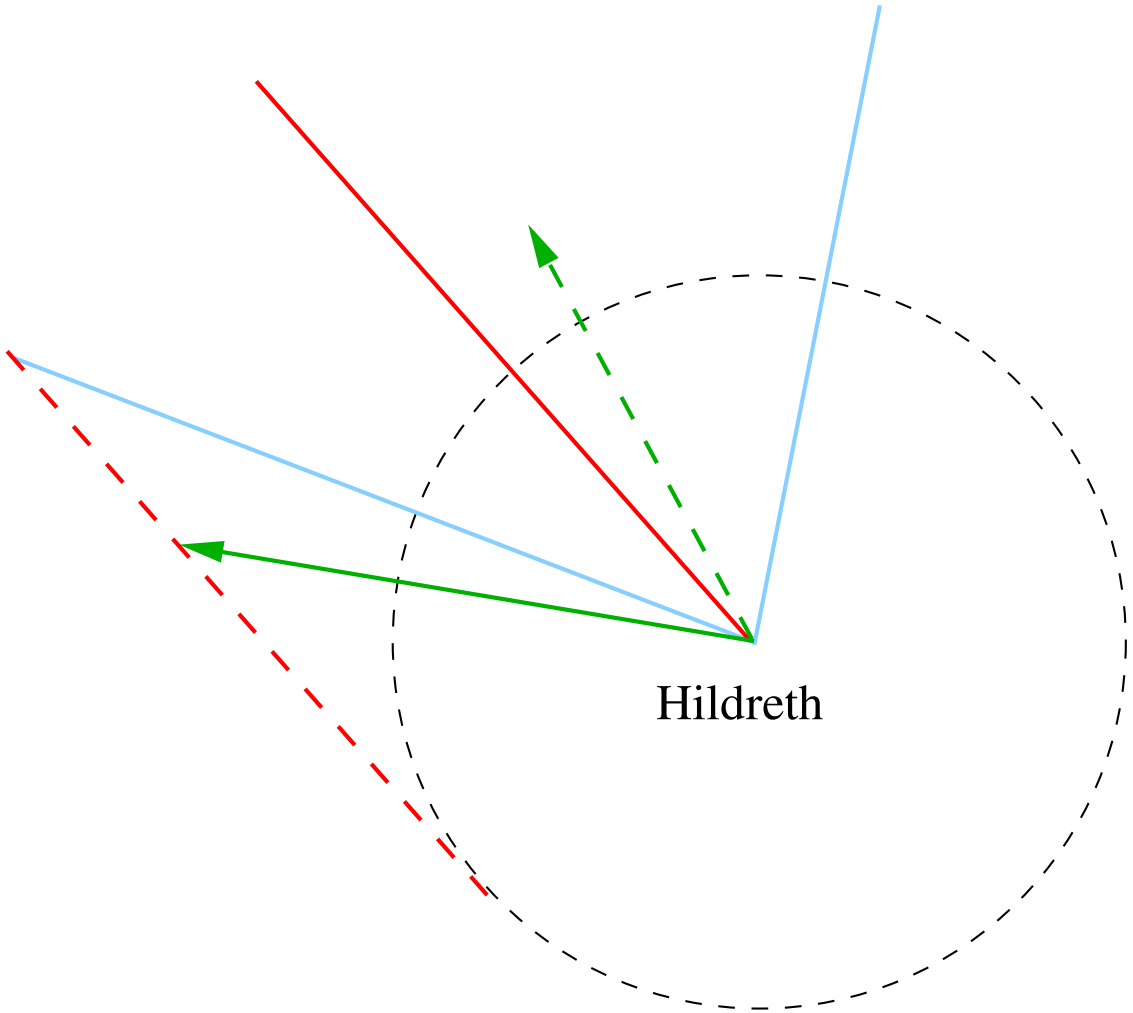
The cone of consistent hyperplanes



The cone of consistent hyperplanes



The cone of consistent hyperplanes



Mistake bounds for various updates

On any sequence of examples such that $y_t \mathbf{u}^\top \mathbf{x}_t \geqslant 1$ with $\|\mathbf{u}\| = U$

Mistake bounds for various updates

On any sequence of examples such that $y_t \mathbf{u}^\top \mathbf{x}_t \geqslant 1$ with $\|\mathbf{u}\| = U$

BOUND	ALGORITHM	UPDATE TIME
U^2	Perceptron	$O(d)$
U^2	Hildreth	$O(d)$
$dU \ln U$	2nd order Perceptron	$O(d^2)$
$d^2 \ln U$	Ellipsoid	$O(d^3)$
$d \ln U$	Volumetric center (U known)	$O(d^{3.5})$
$d \ln U$	Geometric center (U known)	$O(d^4)$

Mistake bounds for various updates

On any sequence of examples such that $y_t \mathbf{u}^\top \mathbf{x}_t \geqslant 1$ with $\|\mathbf{u}\| = U$

BOUND	ALGORITHM	UPDATE TIME
U^2	Perceptron	$O(d)$
U^2	Hildreth	$O(d)$
$dU \ln U$	2nd order Perceptron	$O(d^2)$
$d^2 \ln U$	Ellipsoid	$O(d^3)$
$d \ln U$	Volumetric center (U known)	$O(d^{3.5})$
$d \ln U$	Geometric center (U known)	$O(d^4)$

Note

- U^2 optimal for fixed U
- $d \ln U$ optimal for fixed U and d

Summary

Aggressive updates for nonseparable streams

Passive-aggressive algorithm [Crammer et al. 2006]

- General update form is $\boldsymbol{w}_t = \boldsymbol{w}_{t-1} + \eta_t y_t \boldsymbol{x}_t$

Aggressive updates for nonseparable streams

Passive-aggressive algorithm [Crammer et al. 2006]

- General update form is $\boldsymbol{w}_t = \boldsymbol{w}_{t-1} + \eta_t y_t \boldsymbol{x}_t$
- PA-I for hinge loss $\eta_t = \min \left\{ C, \frac{\ell_t(\boldsymbol{w}_{t-1})}{\|\boldsymbol{x}_t\|^2} \right\}$

Aggressive updates for nonseparable streams

Passive-aggressive algorithm [Crammer et al. 2006]

- General update form is $w_t = w_{t-1} + \eta_t y_t x_t$
- PA-I for hinge loss $\eta_t = \min \left\{ C, \frac{\ell_t(w_{t-1})}{\|x_t\|^2} \right\}$
- This is the solution of $\operatorname{argmin}_w \frac{1}{2} \|w - w_{t-1}\|^2 + C\, \ell_t(w)$

Aggressive updates for nonseparable streams

Passive-aggressive algorithm [Crammer et al. 2006]

- General update form is $w_t = w_{t-1} + \eta_t y_t x_t$
- PA-I for hinge loss $\eta_t = \min \left\{ C, \frac{\ell_t(w_{t-1})}{\|x_t\|^2} \right\}$
- This is the solution of $\operatorname{argmin}_w \frac{1}{2} \|w - w_{t-1}\|^2 + C\, \ell_t(w)$
- PA-II for squared hinge loss $\eta_t = \frac{\ell_t(w_{t-1})}{\|x_t\|^2 + \frac{1}{C}}$

Aggressive updates for nonseparable streams

Passive-aggressive algorithm [Crammer et al. 2006]

- General update form is $w_t = w_{t-1} + \eta_t y_t x_t$
- PA-I for hinge loss $\eta_t = \min \left\{ C, \frac{\ell_t(w_{t-1})}{\|x_t\|^2} \right\}$
- This is the solution of $\operatorname{argmin}_w \frac{1}{2} \|w - w_{t-1}\|^2 + C\, \ell_t(w)$
- PA-II for squared hinge loss $\eta_t = \frac{\ell_t(w_{t-1})}{\|x_t\|^2 + \frac{1}{C}}$
- This is the solution of $\operatorname{argmin}_w \frac{1}{2} \|w - w_{t-1}\|^2 + \frac{C}{2} \ell_t(w)^2$

SVM and passive-aggressive

SVM primal objective

$$\text{SVMOPT} = \min_{\mathbf{u}} \left(\frac{1}{2} \|\mathbf{u}\|^2 + C \sum_{t=1}^{T} \xi_t \right) \quad \text{s.t.} \quad y_t \mathbf{u}^\top \mathbf{x}_t \geqslant 1 - \xi_t \ \forall t$$

SVM and passive-aggressive

SVM primal objective

$$\text{SVMOPT} = \min_{\mathbf{u}} \left(\frac{1}{2} \|\mathbf{u}\|^2 + C \sum_{t=1}^{T} \xi_t \right) \quad \text{s.t.} \quad y_t \mathbf{u}^\top \mathbf{x}_t \geqslant 1 - \xi_t \ \forall t$$

SVM dual Lagrange function

$$D_{\text{svm}}(\alpha_1, \ldots, \alpha_T) = -\frac{1}{2} \left\| \sum_t \alpha_t \mathbf{x}_t \right\|^2 + \sum_t \alpha_t \qquad 0 \leqslant \alpha_t \leqslant C \quad \forall t$$

SVM and passive-aggressive

SVM primal objective

$$\text{SVMOPT} = \min_{\mathbf{u}} \left(\frac{1}{2} \|\mathbf{u}\|^2 + C \sum_{t=1}^{T} \xi_t \right) \quad \text{s.t.} \quad y_t \mathbf{u}^\top \mathbf{x}_t \geqslant 1 - \xi_t \ \forall t$$

SVM dual Lagrange function

$$D_{\text{svm}}(\alpha_1, \ldots, \alpha_T) = -\frac{1}{2} \left\| \sum_t \alpha_t \mathbf{x}_t \right\|^2 + \sum_t \alpha_t \qquad 0 \leqslant \alpha_t \leqslant C \quad \forall t$$

Duality

For all $\alpha'_1, \ldots, \alpha'_T \geqslant 0$

$$D_{\text{svm}}(\alpha'_1, \ldots, \alpha'_T) \leqslant \sup_{\boldsymbol{\alpha}} D_{\text{svm}}(\boldsymbol{\alpha}) = \text{SVMOPT}$$

SVM and passive-aggressive (cont.)

Following [Shalev-Shwartz and Singer, 2006]

$$
\begin{aligned}
D_{\mathrm{svm}}(0,\ldots,0) &= 0 \\
D_{\mathrm{svm}}(\alpha_1,\ldots,\alpha_T) &= \sum_{t=1}^{T} \Big(D_{\mathrm{svm}}(\alpha_1,\ldots,\alpha_t,0,\ldots,0) \\
&\qquad - D_{\mathrm{svm}}(\alpha_1,\ldots,\alpha_{t-1},0,\ldots,0) \Big)
\end{aligned}
$$

SVM and passive-aggressive (cont.)

Following [Shalev-Shwartz and Singer, 2006]

$$
\begin{aligned}
D_{\text{svm}}(0,\ldots,0) &= 0 \\
D_{\text{svm}}(\alpha_1,\ldots,\alpha_T) &= \sum_{t=1}^{T} \Big(D_{\text{svm}}(\alpha_1,\ldots,\alpha_t,0,\ldots,0) \\
&\qquad - D_{\text{svm}}(\alpha_1,\ldots,\alpha_{t-1},0,\ldots,0) \Big)
\end{aligned}
$$

$$
\begin{aligned}
D_{\text{svm}}(\alpha_1,\ldots,\alpha_t,0,\ldots,0) &- D_{\text{svm}}(\alpha_1,\ldots,\alpha_{t-1},0,\ldots,0) \\
&= -\frac{1}{2}\alpha_t^2 \|x_t\|^2 + \alpha_t \left(1 - y_t w_{t-1}^\top x_t\right)
\end{aligned}
$$

where $w_{t-1} = \sum_{s=1}^{t-1} \alpha_s y_s x_s$

SVM and passive-aggressive (cont.)

PA-I primal objective

$$\min_{w} \left(\frac{1}{2} \|w - w_{t-1}\|^2 + C\xi_t \right) \quad \text{s.t.} \quad y_t w^\top x_t \geqslant 1 - \xi_t$$

SVM and passive-aggressive (cont.)

PA-I primal objective

$$\min_{w} \left(\frac{1}{2} \|w - w_{t-1}\|^2 + C\xi_t \right) \quad \text{s.t.} \quad y_t w^\top x_t \geqslant 1 - \xi_t$$

PA-I dual Lagrange function

$$D_t(\alpha) = -\frac{1}{2}\alpha^2 \|x_t\|^2 + \alpha(1 - y_t w_{t-1}^\top x_t)$$

SVM and passive-aggressive (cont.)

PA-I primal objective

$$\min_{w} \left(\frac{1}{2} \|w - w_{t-1}\|^2 + C\xi_t \right) \quad \text{s.t.} \quad y_t w^\top x_t \geqslant 1 - \xi_t$$

PA-I dual Lagrange function

$$D_t(\alpha) = -\frac{1}{2}\alpha^2 \|x_t\|^2 + \alpha(1 - y_t w_{t-1}^\top x_t)$$

PA-I learning rate is maximizer of dual objective

$$\eta_t = \operatorname{argmin} \left\{ C, \frac{\ell_t(w_{t-1})}{\|x_t\|^2} \right\} = \operatorname*{argmax}_{0 \leqslant \alpha \leqslant C} D_t(\alpha) \stackrel{\text{def}}{=} \alpha_t$$

Mistake bounds for PA-I

We note that

$$D_t(\alpha_t) = D_{\text{svm}}(\alpha_1, \ldots, \alpha_t, 0, \ldots, 0) - D_{\text{svm}}(\alpha_1, \ldots, \alpha_{t-1}, 0, \ldots, 0)$$

Mistake bounds for PA-I

We note that

$$D_t(\alpha_t) = D_{\text{svm}}(\alpha_1, \ldots, \alpha_t, 0, \ldots, 0) - D_{\text{svm}}(\alpha_1, \ldots, \alpha_{t-1}, 0, \ldots, 0)$$

This shows

$$\sum_{t=1}^{T} D_t(\alpha_t) \leqslant \text{SVMOPT} = \min_{\mathbf{u}} \left(\frac{1}{2} \|\mathbf{u}\|^2 + C \sum_{t=1}^{T} \ell_t(\mathbf{u}) \right)$$

Mistake bounds for PA-I

We note that

$$D_t(\alpha_t) = D_{\text{svm}}(\alpha_1, \ldots, \alpha_t, 0, \ldots, 0) - D_{\text{svm}}(\alpha_1, \ldots, \alpha_{t-1}, 0, \ldots, 0)$$

This shows

$$\sum_{t=1}^{T} D_t(\alpha_t) \leqslant \text{SVMOPT} = \min_{\mathbf{u}} \left(\frac{1}{2} \|\mathbf{u}\|^2 + C \sum_{t=1}^{T} \ell_t(\mathbf{u}) \right)$$

- We can use this inequality to prove mistake bounds for PA-I
- Proof idea: lower bound $D_t(\alpha_t)$ at mistakes

Proof of mistake bound for PA-I

Assume for simplicity $\|\mathbf{x}_t\| = 1$

Proof of mistake bound for PA-I

Assume for simplicity $\|\mathbf{x}_t\| = 1$

Lemma

$$y_t \mathbf{w}_{t-1}^\top \mathbf{x}_t \leqslant 0 \quad \text{implies} \quad D_t(\alpha_t) \geqslant \min\{C, \ell_t(\mathbf{w}_{t-1})\} \frac{\ell_t(\mathbf{w}_{t-1})}{2}$$

Proof of mistake bound for PA-I

Assume for simplicity $\|x_t\| = 1$

Lemma

$$y_t w_{t-1}^\top x_t \leqslant 0 \quad \text{implies} \quad D_t(\alpha_t) \geqslant \min\{C, \ell_t(w_{t-1})\} \frac{\ell_t(w_{t-1})}{2}$$

This implies

$$\min\{C, 1\} m_t(w_{t-1}) \leqslant \min\{C, \underbrace{\ell_t(w_{t-1})}_{\geqslant 1}\} \underbrace{\ell_t(w_{t-1})}_{\geqslant m_t(w_{t-1})} \leqslant 2 D_t(\alpha_t)$$

Proof of mistake bound for PA-I

Assume for simplicity $\|x_t\| = 1$

Lemma

$$y_t w_{t-1}^\top x_t \leqslant 0 \quad \text{implies} \quad D_t(\alpha_t) \geqslant \min\{C, \ell_t(w_{t-1})\} \frac{\ell_t(w_{t-1})}{2}$$

This implies

$$\min\{C,1\} m_t(w_{t-1}) \leqslant \min\{C, \underbrace{\ell_t(w_{t-1})}_{\geqslant 1}\} \underbrace{\ell_t(w_{t-1})}_{\geqslant m_t(w_{t-1})} \leqslant 2 D_t(\alpha_t)$$

Theorem [Crammer et al., 2006]

$$\sum_{t=1}^{T} m_t(w_{t-1}) \leqslant \max\left\{\frac{1}{C}, 1\right\} \inf_{u \in \mathbb{R}^d} \left(\|u\|^2 + 2C \sum_{t=1}^{T} \ell_t(u) \right)$$

Summary

On-line learning with kernels

- Feature map $\phi : \mathbb{R}^d \to \text{RKHS}$

On-line learning with kernels

- Feature map $\phi : \mathbb{R}^d \to \text{RKHS}$
- Kernel $K(\mathbf{x}, \mathbf{x}') = \langle \phi(\mathbf{x}), \phi(\mathbf{x}') \rangle$

On-line learning with kernels

- Feature map $\phi : \mathbb{R}^d \to \text{RKHS}$
- Kernel $K(x, x') = \langle \phi(x), \phi(x') \rangle$
- Assume a linear algorithm learns w such that

$$w = \sum_i \alpha_i x_{t_i}$$

On-line learning with kernels

- Feature map $\phi : \mathbb{R}^d \to \text{RKHS}$
- Kernel $K(x, x') = \langle \phi(x), \phi(x') \rangle$
- Assume a linear algorithm learns w such that

$$w = \sum_i \alpha_i x_{t_i}$$

- Then we can learn $w = \sum_i \alpha_i \phi(x_{t_i})$ in the RKHS because

$$\begin{aligned} \text{SGN}\big(\langle w, \phi(x) \rangle\big) &= \text{SGN}\left(\sum_i y_{t_i} \langle \phi(x_{t_i}), \phi(x) \rangle \right) \\ &= \text{SGN}\left(\sum_i y_{t_i} K(x_{t_i}, x) \right) \end{aligned}$$

Kernel Perceptron

Start with empty cache $\mathcal{L}$ of examples

Loop:

Kernel Perceptron

Start with empty cache $\mathcal{L}$ of examples

Loop:

1. Read next instance $\mathbf{x}_t$

Kernel Perceptron

Start with empty cache $\mathcal{L}$ of examples

Loop:

1. Read next instance $\mathbf{x}_t$
2. Predict y_t with $\widehat{y}_t = \text{SGN}\left(\sum_{\mathbf{v}\in\mathcal{L}} K(\mathbf{v}, \mathbf{x}_t)\right)$

Kernel Perceptron

Start with empty cache $\mathcal{L}$ of examples

Loop:

1. Read next instance $\mathbf{x}_t$
2. Predict y_t with $\widehat{y}_t = \text{SGN}\left(\sum_{\mathbf{v}\in\mathcal{L}} K(\mathbf{v}, \mathbf{x}_t)\right)$
3. Obtain true label y_t

Kernel Perceptron

Start with empty cache $\mathcal{L}$ of examples

Loop:

1. Read next instance $\mathbf{x}_t$
2. Predict y_t with $\widehat{y}_t = \text{SGN}\left(\sum_{v \in \mathcal{L}} K(v, \mathbf{x}_t)\right)$
3. Obtain true label y_t
4. If $\widehat{y}_t \neq y_t$ (mistake) then store new support $(y_t \mathbf{x}_t)$ in $\mathcal{L}$

Kernel Perceptron

Start with empty cache $\mathcal{L}$ of examples

Loop:

1. Read next instance $\mathbf{x}_t$
2. Predict y_t with $\widehat{y}_t = \text{SGN}\Big(\sum_{v \in \mathcal{L}} K(v, \mathbf{x}_t)\Big)$
3. Obtain true label y_t
4. If $\widehat{y}_t \neq y_t$ (mistake) then store new support $(y_t \mathbf{x}_t)$ in $\mathcal{L}$

Mistake bounds hold in the whole RKHS

Memory bounded learning

Can we control the rate of mistakes when at most $B < \infty$ supports are used?

Memory bounded learning

Can we control the rate of mistakes when at most $B < \infty$ supports are used?

Fact

Using at most B supports, any learner makes an unbounded number of mistakes on a sequence that is perfectly classified by some $\mathbf{u} \in \mathbb{R}^d$ with zero hinge loss and $\|\mathbf{u}\| = \sqrt{B+1}$

Memory bounded learning

Can we control the rate of mistakes when at most $B < \infty$ supports are used?

Fact

Using at most B supports, any learner makes an unbounded number of mistakes on a sequence that is perfectly classified by some $\mathbf{u} \in \mathbb{R}^d$ with zero hinge loss and $\|\mathbf{u}\| = \sqrt{B+1}$

- Thus $B \geqslant U^2$ is necessary to compete against $\mathbf{u}$ of length U

Memory bounded learning

Can we control the rate of mistakes when at most $B < \infty$ supports are used?

Fact

Using at most B supports, any learner makes an unbounded number of mistakes on a sequence that is perfectly classified by some $\mathbf{u} \in \mathbb{R}^d$ with zero hinge loss and $\|\mathbf{u}\| = \sqrt{B+1}$

- Thus $B \geqslant U^2$ is necessary to compete against $\mathbf{u}$ of length U
- Can we compete against any $\mathbf{u}$ with $\|\mathbf{u}\| \leqslant U$ using $B = (1+\varepsilon)U^2$ supports?

A randomized perceptron

Randomized Budget Perceptron

Parameter: size B of cache for supports

Start with empty cache $\mathcal{L}$

A randomized perceptron

Randomized Budget Perceptron

Parameter: size B of cache for supports

Start with empty cache $\mathcal{L}$

Loop:

1. Read next instance $\mathbf{x}_t$

A randomized perceptron

Randomized Budget Perceptron

Parameter: size B of cache for supports

Start with empty cache $\mathcal{L}$

Loop:

1. Read next instance $\mathbf{x}_t$
2. Predict y_t with $\widehat{y}_t = \text{SGN}\left(\sum_{\mathbf{v}\in\mathcal{L}} \mathbf{v}^\top \mathbf{x}_t\right)$

A randomized perceptron

Randomized Budget Perceptron

Parameter: size B of cache for supports

Start with empty cache $\mathcal{L}$

Loop:

1. Read next instance $\mathbf{x}_t$
2. Predict y_t with $\widehat{y}_t = \text{SGN}\left(\sum_{\mathbf{v}\in\mathcal{L}} \mathbf{v}^\top \mathbf{x}_t\right)$
3. Obtain true label y_t

A randomized perceptron

Randomized Budget Perceptron

Parameter: size B of cache for supports

Start with empty cache $\mathcal{L}$

Loop:

1. Read next instance $\mathbf{x}_t$
2. Predict y_t with $\widehat{y}_t = \text{SGN}\left(\sum_{\mathbf{v}\in\mathcal{L}} \mathbf{v}^\top \mathbf{x}_t\right)$
3. Obtain true label y_t
4. If $\widehat{y}_t \neq y_t$ then:

A randomized perceptron

Randomized Budget Perceptron

Parameter: size B of cache for supports

Start with empty cache $\mathcal{L}$

Loop:

1. Read next instance $\mathbf{x}_t$
2. Predict y_t with $\widehat{y}_t = \text{SGN}\left(\sum_{\mathbf{v}\in\mathcal{L}} \mathbf{v}^\top \mathbf{x}_t\right)$
3. Obtain true label y_t
4. If $\widehat{y}_t \neq y_t$ then:
   1. If $|\mathcal{L}| = B$, then throw away a random support from $\mathcal{L}$

A randomized perceptron

Randomized Budget Perceptron

Parameter: size B of cache for supports

Start with empty cache $\mathcal{L}$

Loop:

1. Read next instance $\mathbf{x}_t$
2. Predict y_t with $\widehat{y}_t = \text{SGN}\left(\sum_{\mathbf{v}\in\mathcal{L}} \mathbf{v}^\top \mathbf{x}_t\right)$
3. Obtain true label y_t
4. If $\widehat{y}_t \neq y_t$ then:
   1. If $|\mathcal{L}| = B$, then throw away a random support from $\mathcal{L}$
   2. Add $y_t \mathbf{x}_t$ to $\mathcal{L}$

A randomized perceptron

Randomized Budget Perceptron

Parameter: size B of cache for supports

Start with empty cache $\mathcal{L}$

Loop:

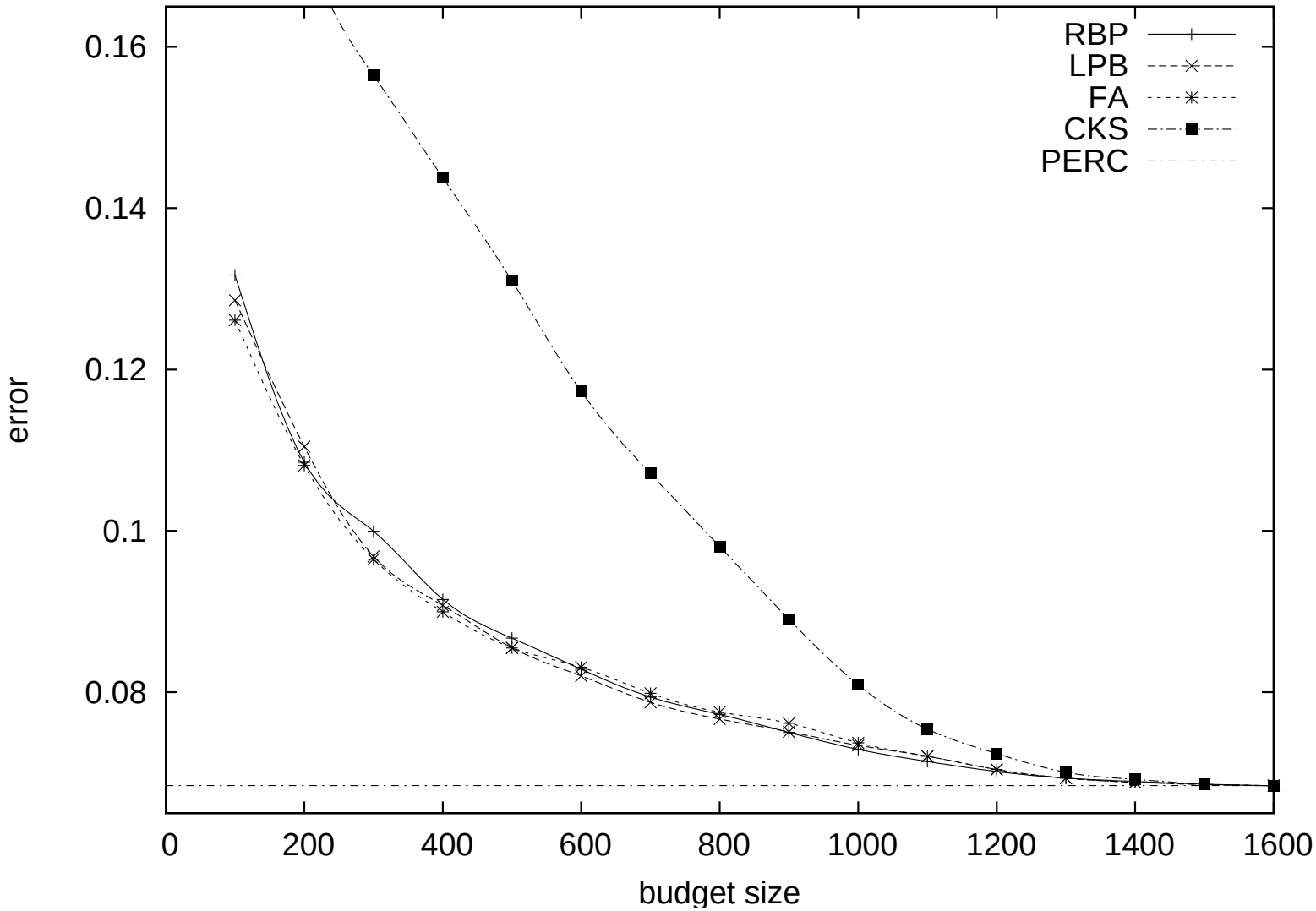
1. Read next instance $\mathbf{x}_t$
2. Predict y_t with $\widehat{y}_t = \text{SGN}\Big(\sum_{\mathbf{v}\in\mathcal{L}} \mathbf{v}^\top \mathbf{x}_t\Big)$
3. Obtain true label y_t
4. If $\widehat{y}_t \neq y_t$ then:
   1. If $|\mathcal{L}| = B$, then throw away a random support from $\mathcal{L}$
   2. Add $y_t \mathbf{x}_t$ to $\mathcal{L}$

Result:

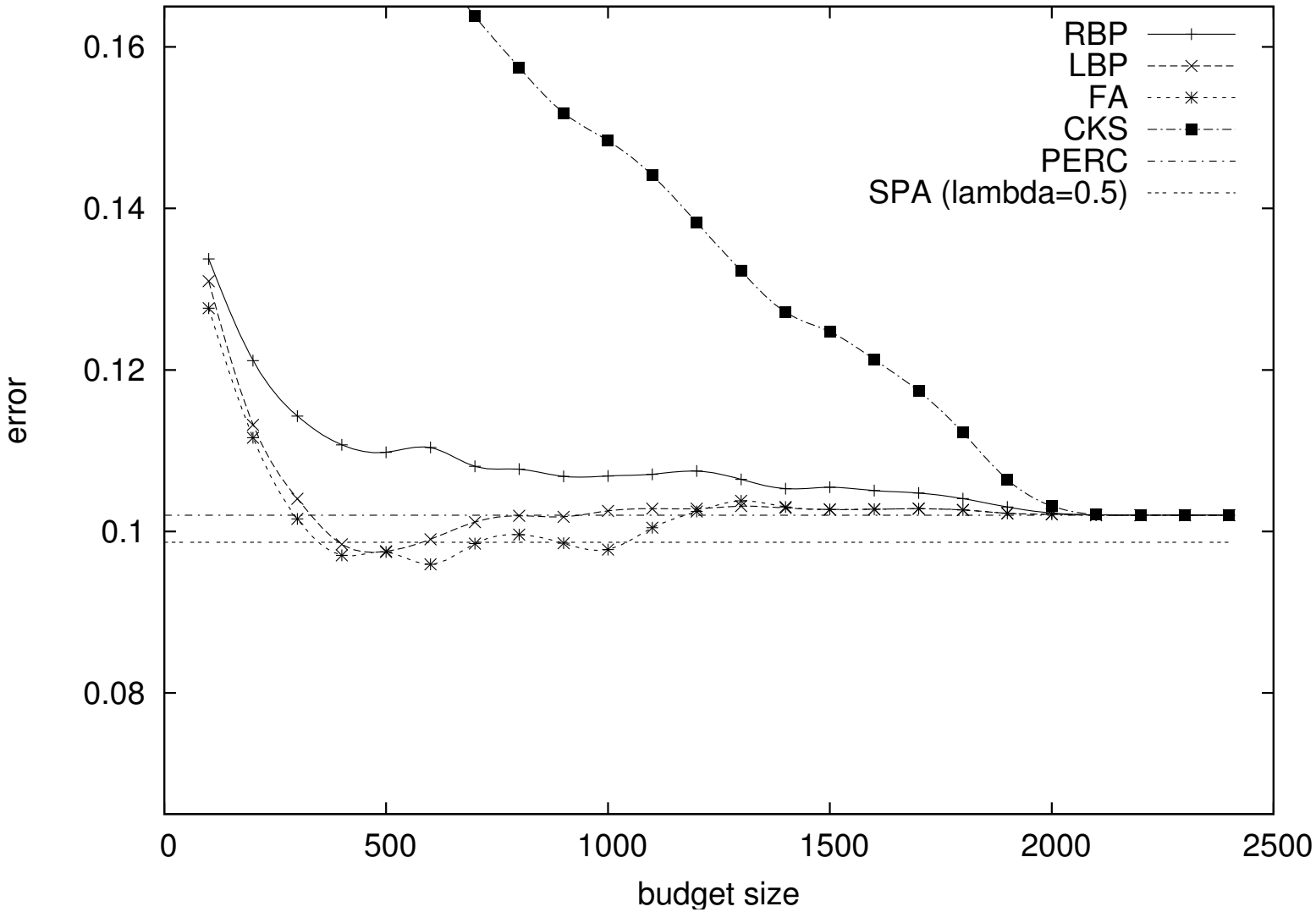
Bound on mistakes scales roughly with $1 + 1/\varepsilon$

(when using $B = (1+\varepsilon)U^2$ supports)

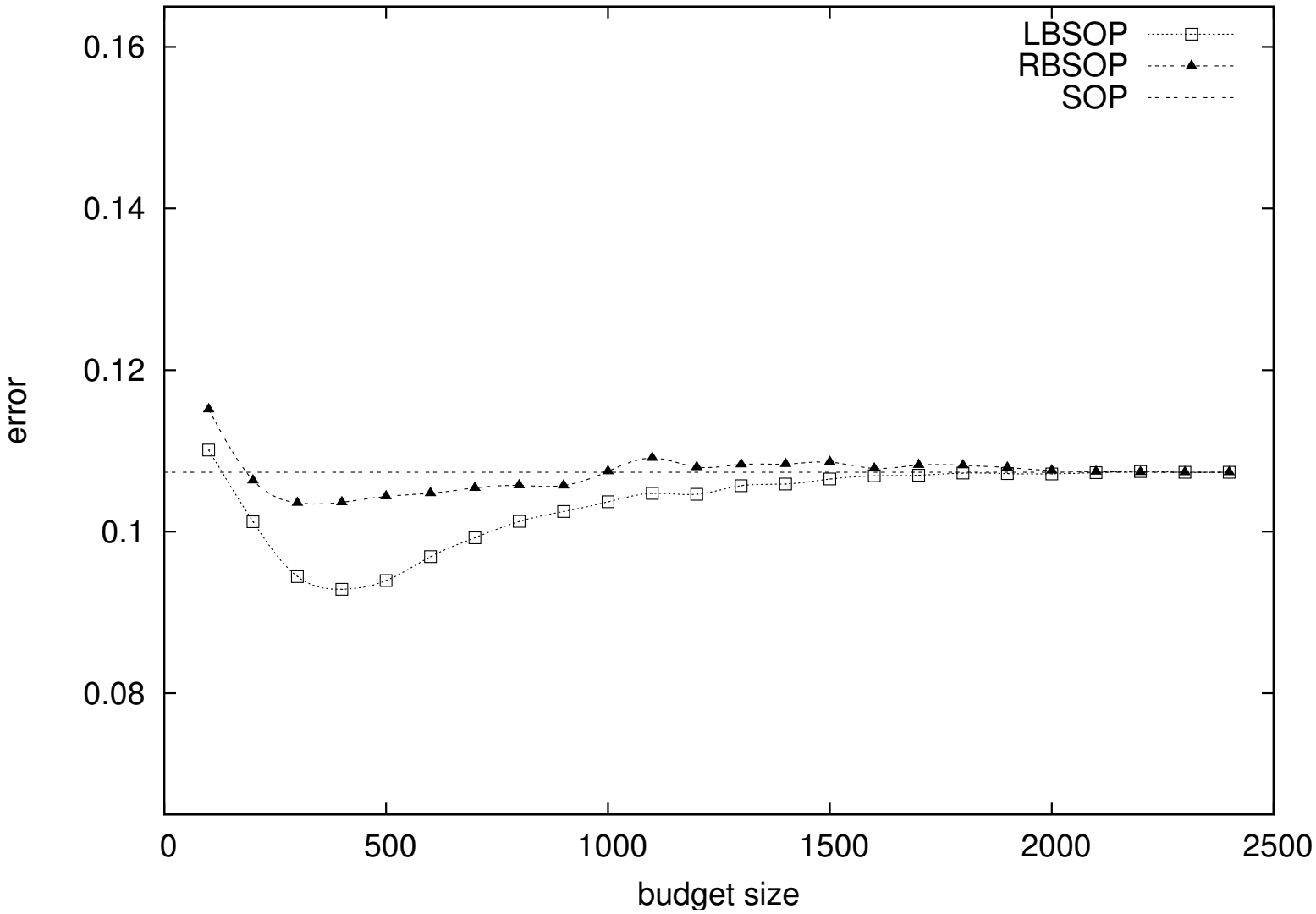
Empirical performance — stationary



Empirical performance — nonstationary



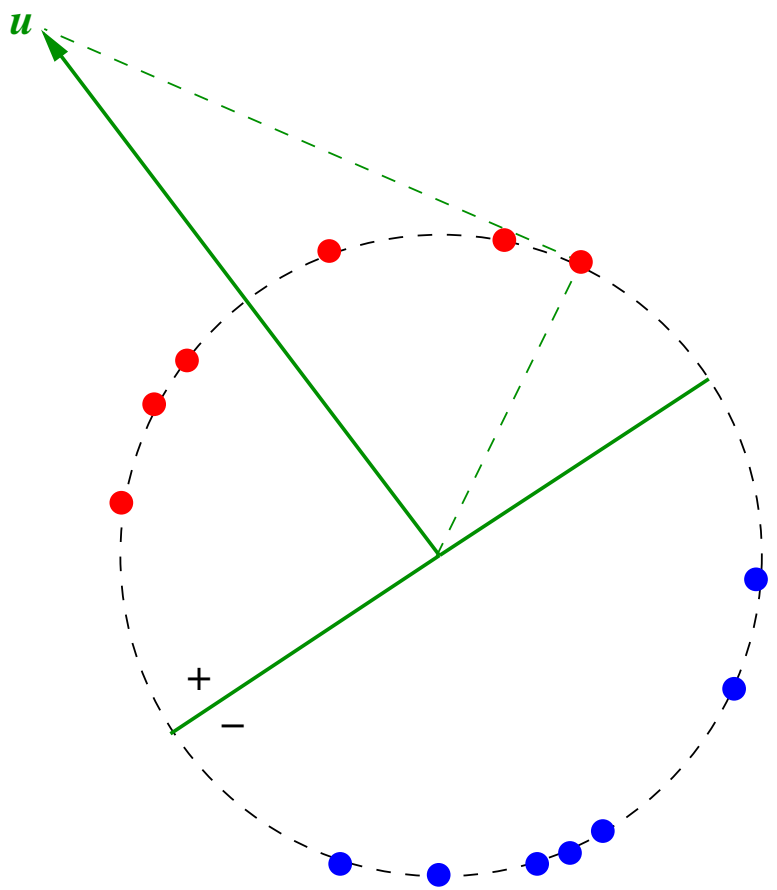
Empirical performance 2nd order — nonstationary



Summary

Online approximation of SVM hyperplane

The SVM hyperplane is the shortest $\mathbf{u}$ such that $y_t \mathbf{u}^\top \mathbf{x}_t \geqslant 1$ for all t



Online approximation of SVM hyperplane (cont.)

The ALMA algorithm

Parameter: $0 < \alpha \leqslant 1$

Set mistake counter $k = 1$

Online approximation of SVM hyperplane (cont.)

The ALMA algorithm

Parameter: $0 < \alpha \leqslant 1$

Set mistake counter $k = 1$

Loop:

1. Read next instance $\mathbf{x}_t$

Online approximation of SVM hyperplane (cont.)

The ALMA algorithm

Parameter: $0 < \alpha \leqslant 1$

Set mistake counter $k = 1$

Loop:

1. Read next instance $\boldsymbol{x}_t$
2. Predict y_t with $\widehat{y}_t = \text{SGN}\left(\boldsymbol{w}^\top \boldsymbol{x}_t\right)$

Online approximation of SVM hyperplane (cont.)

The ALMA algorithm

Parameter: $0 < \alpha \leqslant 1$

Set mistake counter $k = 1$

Loop:

1. Read next instance $\boldsymbol{x}_t$
2. Predict y_t with $\widehat{y}_t = \text{SGN}\left(\boldsymbol{w}^\top \boldsymbol{x}_t\right)$
3. Obtain true label y_t

Online approximation of SVM hyperplane (cont.)

The ALMA algorithm

Parameter: $0 < \alpha \leqslant 1$

Set mistake counter $k = 1$

Loop:

1. Read next instance $\boldsymbol{x}_t$
2. Predict y_t with $\widehat{y}_t = \text{SGN}\left(\boldsymbol{w}^\top \boldsymbol{x}_t\right)$
3. Obtain true label y_t
4. If margin smaller than $c(1-\alpha)/\sqrt{k}$ then:

Online approximation of SVM hyperplane (cont.)

The ALMA algorithm

Parameter: $0 < \alpha \leqslant 1$

Set mistake counter $k = 1$

Loop:

1. Read next instance $\boldsymbol{x}_t$
2. Predict y_t with $\widehat{y}_t = \text{SGN}\left(\boldsymbol{w}^\top \boldsymbol{x}_t\right)$
3. Obtain true label y_t
4. If margin smaller than $c(1-\alpha)/\sqrt{k}$ then:
   1. $\boldsymbol{w}' = \boldsymbol{w} + y_t \boldsymbol{x}_t / \sqrt{k}$

Online approximation of SVM hyperplane (cont.)

The ALMA algorithm

Parameter: $0 < \alpha \leqslant 1$

Set mistake counter $k = 1$

Loop:

1. Read next instance $\boldsymbol{x}_t$
2. Predict y_t with $\widehat{y}_t = \text{SGN}\left(\boldsymbol{w}^\top \boldsymbol{x}_t\right)$
3. Obtain true label y_t
4. If margin smaller than $c(1-\alpha)/\sqrt{k}$ then:
   1. $\boldsymbol{w}' = \boldsymbol{w} + y_t \boldsymbol{x}_t / \sqrt{k}$
   2. $\boldsymbol{w} = \boldsymbol{w}' / \|\boldsymbol{w}'\| \qquad k \leftarrow k+1$

Online approximation of SVM hyperplane (cont.)

The ALMA algorithm

Parameter: $0 < \alpha \leqslant 1$

Set mistake counter $k = 1$

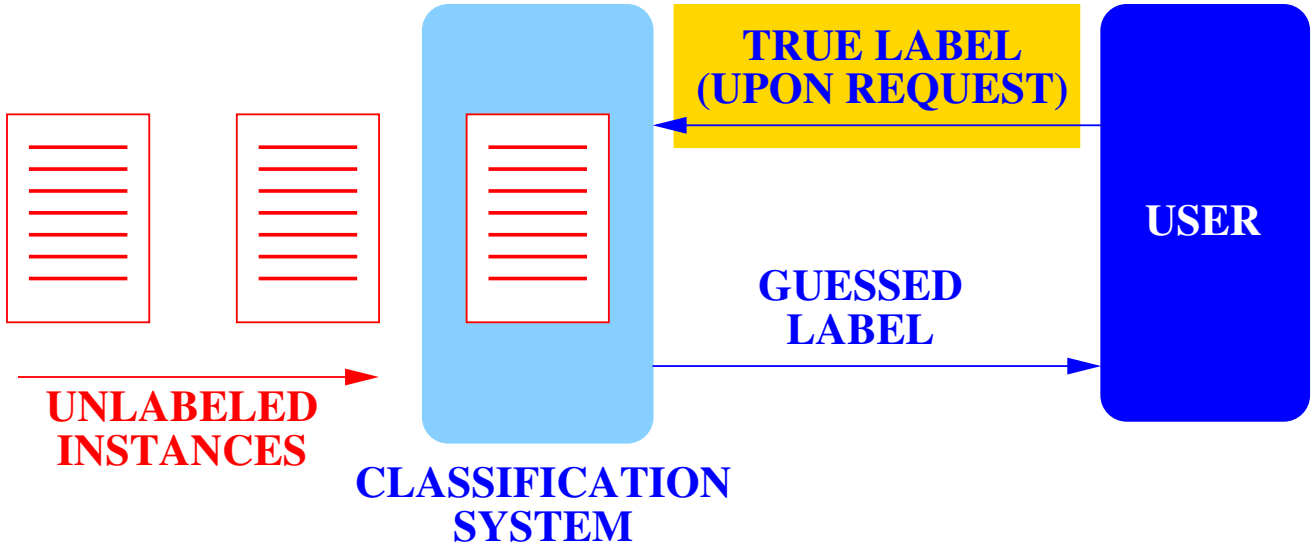
Loop:

1. Read next instance $\boldsymbol{x}_t$
2. Predict y_t with $\widehat{y}_t = \text{SGN}\left(\boldsymbol{w}^\top \boldsymbol{x}_t\right)$
3. Obtain true label y_t
4. If margin smaller than $c(1-\alpha)/\sqrt{k}$ then:
   1. $\boldsymbol{w}' = \boldsymbol{w} + y_t \boldsymbol{x}_t / \sqrt{k}$
   2. $\boldsymbol{w} = \boldsymbol{w}' / \|\boldsymbol{w}'\| \qquad k \leftarrow k+1$

Result

Finds separating $\boldsymbol{u}$ with $\|\boldsymbol{u}\| \leqslant \|\boldsymbol{u}_{\text{SVM}}\| / (1-\alpha)$ after at most $\left(\|\boldsymbol{u}_{\text{SVM}}\| / \alpha\right)^2$ updates

Selective sampling



A selective sampling classifier

1. Classify next instance $\mathbf{x}_t$ with $\text{SGN}(\boldsymbol{w}^\top \mathbf{x}_t)$

A selective sampling classifier

1. Classify next instance x_t with $\text{SGN}(w^\top x_t)$
2. If $|w^\top x_t| \leqslant \|x_t\| \sqrt{\frac{c \ln t}{N_t}}$ then query label y_t of x_t

$N_t =$ number of labels sampled so far

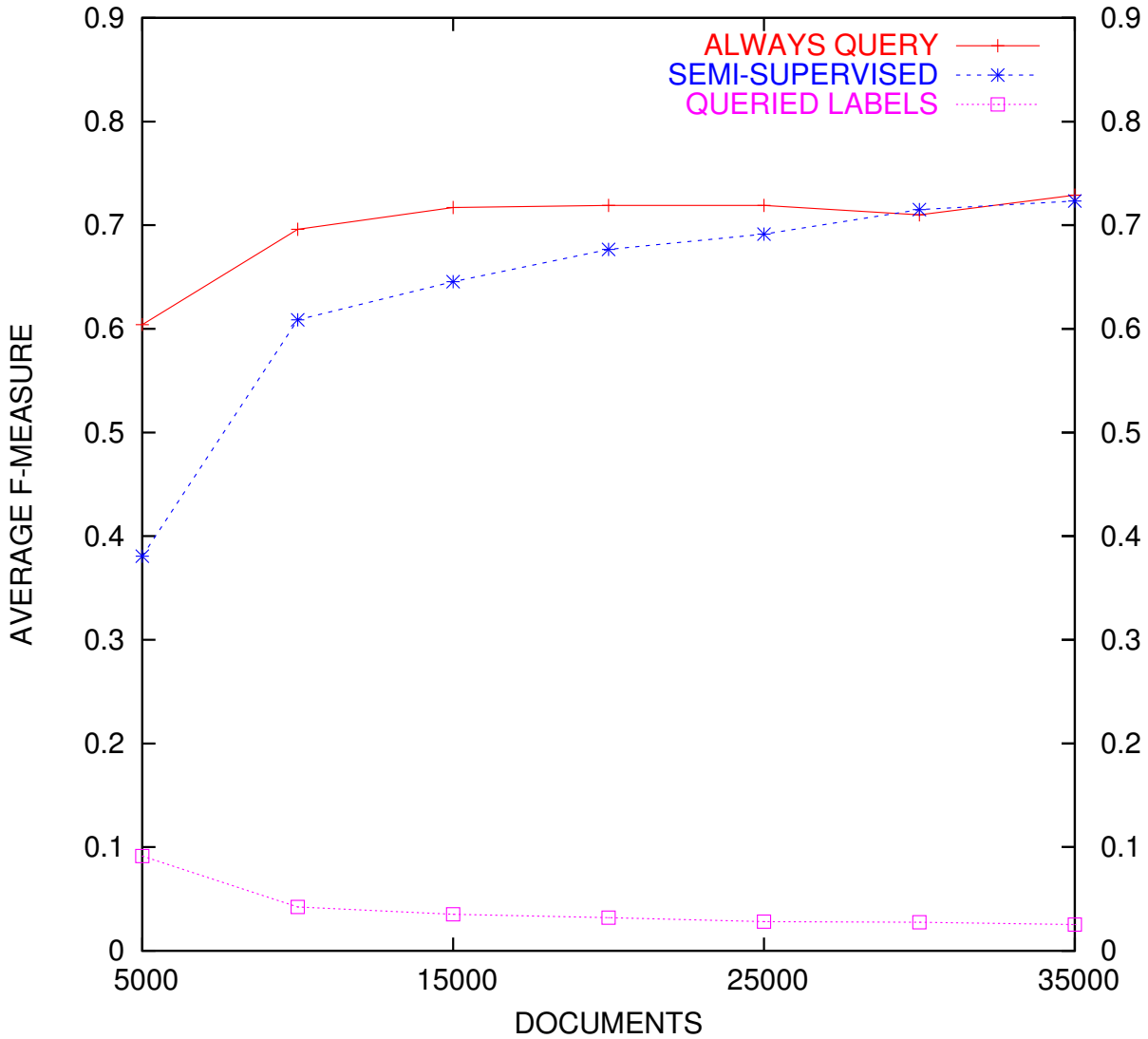
A selective sampling classifier

1. Classify next instance x_t with $\text{SGN}(w^\top x_t)$
2. If $|w^\top x_t| \leqslant \|x_t\| \sqrt{\frac{c \ln t}{N_t}}$ then query label y_t of x_t
3. If label queried then use (x_t, y_t) to update w

$N_t =$ number of labels sampled so far

w updated with the 2nd order Perceptron update rule

Empirical performance on RCV1



Summary

Statistical learning theory

- Linear classifiers $\mathrm{H}(\boldsymbol{x}) = \mathrm{SGN}\left(\boldsymbol{w}^\top \boldsymbol{x}\right)$

Statistical learning theory

- Linear classifiers $H(\boldsymbol{x}) = \text{SGN}\left(\boldsymbol{w}^\top \boldsymbol{x}\right)$
- Examples $(\boldsymbol{x}_t, y_t)$ are i.i.d. according to a fixed and unknown probability distribution on $\mathbb{R}^d \times \{-1, +1\}$

Statistical learning theory

- Linear classifiers $H(\boldsymbol{x}) = \text{SGN}\left(\boldsymbol{w}^\top \boldsymbol{x}\right)$
- Examples $(\boldsymbol{x}_t, y_t)$ are i.i.d. according to a fixed and unknown probability distribution on $\mathbb{R}^d \times \{-1, +1\}$
- $\texttt{risk}(H) = \mathbb{P}\left(H(\boldsymbol{x}) \neq y\right)$

Statistical learning theory

- Linear classifiers $H(\boldsymbol{x}) = \text{SGN}\left(\boldsymbol{w}^\top \boldsymbol{x}\right)$
- Examples $(\boldsymbol{x}_t, y_t)$ are i.i.d. according to a fixed and unknown probability distribution on $\mathbb{R}^d \times \{-1, +1\}$
- $\texttt{risk}(H) = \mathbb{P}\left(H(\boldsymbol{x}) \neq y\right)$
- Learning algorithm

$$(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_n, y_n) \longrightarrow \boxed{A} \longrightarrow \widehat{H} : \mathbb{R}^d \to \{-1, +1\}$$

$\widehat{H}$ is (random) hypothesis output by learner

The ensemble of hypotheses

- Run an incremental learner on the training set

The ensemble of hypotheses

- Run an incremental learner on the training set
- Everytime $H(\mathbf{x}_t) \neq y_t$, H is changed by the update rule

The ensemble of hypotheses

- Run an incremental learner on the training set
- Everytime $H(\mathbf{x}_t) \neq y_t$, H is changed by the update rule
- This process generates an ensemble of classifiers

$$H_0, H_1, \ldots, H_n$$

The ensemble of hypotheses

- Run an incremental learner on the training set
- Everytime $H(x_t) \neq y_t$, H is changed by the update rule
- This process generates an ensemble of classifiers

$$H_0, H_1, \ldots, H_n$$

Goals

1. Bound the average risk of the ensemble in terms of the size of the ensemble

The ensemble of hypotheses

- Run an incremental learner on the training set
- Everytime $H(x_t) \neq y_t$, H is changed by the update rule
- This process generates an ensemble of classifiers

$$H_0, H_1, \ldots, H_n$$

Goals

1. Bound the average risk of the ensemble in terms of the size of the ensemble
2. Find an element of the ensemble whose risk is close to the ensemble average

Step 1: bound the average risk

The difference

$$\mathtt{risk}(H_{t-1}) - \mathbb{I}_{\{H_{t-1}(x_t) \neq y_t\}}$$

is a martingale difference sequence because

$$\mathbb{E}\left[\mathtt{risk}(H_{t-1}) - \mathbb{I}_{\{H_{t-1}(x_t) \neq y_t\}} \,\middle|\, (x_1, y_1), \ldots, (x_{t-1}, y_{t-1})\right] = 0$$

Step 1: bound the average risk

The difference

$$\texttt{risk}(H_{t-1}) - \mathbb{I}_{\{H_{t-1}(x_t) \neq y_t\}}$$

is a martingale difference sequence because

$$\mathbb{E}\left[\texttt{risk}(H_{t-1}) - \mathbb{I}_{\{H_{t-1}(x_t) \neq y_t\}} \,\middle|\, (x_1, y_1), \ldots, (x_{t-1}, y_{t-1})\right] = 0$$

The associated martingale is

$$\sum_{t=1}^{n} \left(\texttt{risk}(H_{t-1}) - \mathbb{I}_{\{H_{t-1}(x_t) \neq y_t\}}\right)$$

$$\iff \underbrace{\frac{1}{n}\sum_{t=1}^{n} \texttt{risk}(H_{t-1})}_{\text{average risk}} - \underbrace{\frac{1}{n}\sum_{t=1}^{n} \mathbb{I}_{\{H_{t-1}(x_t) \neq y_t\}}}_{\text{fraction of mistakes}}$$

Bernstein's bound

If $Z_1, Z_2, \ldots$ is a martingale difference sequence with increments bounded by 1 and

$$V_n = \sum_{t=1}^{n} \mathbb{E}\left[Z_t^2 \mid Z_1, \ldots, Z_{t-1}\right]$$

then for all $S, K > 0$

$$\mathbb{P}\left(\sum_{t=1}^{n} Z_n \geqslant S, \quad V_n \leqslant K\right) \leqslant \exp\left(-\frac{S^2}{2(S/3 + K)}\right)$$

Application of Bernstein's bound

Since $0 \leqslant \mathbb{I}_{\{H(x) \neq y\}} \leqslant 1$,

$$\begin{aligned} &\textsc{var}\left[\mathbb{I}_{\{H_{t-1}(x_t), y_t\}} \,\middle|\, (x_1, y_1), \ldots, (x_{t-1}, y_{t-1})\right] \\ &\leqslant \quad \mathbb{E}\left[\mathtt{risk}(H_{t-1}) \,\middle|\, (x_1, y_1), \ldots, (x_{t-1}, y_{t-1})\right] \end{aligned}$$

Application of Bernstein's bound

Since $0 \leqslant \mathbb{I}_{\{H(x) \neq y\}} \leqslant 1$,

$$\begin{aligned} &\textsc{var}\left[\mathbb{I}_{\{H_{t-1}(x_t), y_t\}} \,\middle|\, (x_1, y_1), \ldots, (x_{t-1}, y_{t-1})\right] \\ &\leqslant \quad \mathbb{E}\left[\mathtt{risk}(H_{t-1}) \,\middle|\, (x_1, y_1), \ldots, (x_{t-1}, y_{t-1})\right] \end{aligned}$$

Applying Bernstein's gives

$$\frac{1}{n}\sum_{t=1}^{n} \mathtt{risk}(H_{t-1}) \quad \leqslant \quad \frac{M_n}{n} + \frac{c}{n}\left(\ln M_n + \sqrt{M_n \ln M_n}\right) \quad \text{w.h.p.}$$

Where $\dfrac{M_n}{n} = \dfrac{1}{n}\displaystyle\sum_{t=1}^{n} \mathbb{I}_{\{H_{t-1}(x_t) \neq Y_t\}}$ is the fraction of mistakes

Step 2: pick a good classifier in the ensemble

- Start from the ensemble $H_0, H_1, \ldots, H_n$
- Do the following:
  1. test each H_t on $(x_{t+1}, y_{t+1}), \ldots, (x_n, y_n)$
  2. pick $\widehat{H} = H_{t^*}$ minimizing a penalized risk estimate

Step 2: pick a good classifier in the ensemble

- Start from the ensemble $H_0, H_1, \ldots, H_n$
- Do the following:
  1. test each H_t on $(x_{t+1}, y_{t+1}), \ldots, (x_n, y_n)$
  2. pick $\widehat{H} = H_{t^*}$ minimizing a penalized risk estimate

Guaranteed bound

$$\texttt{risk}(\widehat{H}) \quad \leqslant \quad \frac{M_n}{n} + \frac{c}{n}\left((\ln n)^2 + \sqrt{M_n \ln n}\right) \qquad \text{w.h.p.}$$