Advanced Machine Learning

Bogdan Alexe,

bogdan.alexe@fmi.unibuc.ro

University of Bucharest, 2nd semester, 2020-2021

Recap - AdaBoost

- construct distribution $\mathbf{D}^{(t)}$ on {1,..., m}:
  - $\mathbf{D}^{(1)}(i) = 1/m$
  - given $\mathbf{D}^{(t)}$ and h_t: $D^{(t+1)}(i) = \dfrac{D^{(t)}(i) \times e^{-w_t h_t(x_i) y_i}}{Z_{t+1}}$

where Z_{t+1} normalization factor ($\mathbf{D}^{(t+1)}$ is a distribution): $Z_{t+1} = \sum_{i=1}^{n} D^{(t)}(i) \times e^{-w_t h_t(x_i) y_i}$

w_t is a weight: $w_t = \frac{1}{2} \ln(\frac{1}{\varepsilon_t} - 1) > 0$ as the error $\varepsilon_t < 0.5$

ε_t is the error of h_t on $\mathbf{D}^{(t)}$: $\varepsilon_t = \Pr_{i \sim D^{(t)}}[h_t(x_i) \neq y_i] = \sum_{i=1}^{m} D^{(t)}(i) \times 1_{[h_t(x_i) \neq y_i]}$

If example $\mathbf{x}_i$ is correctly classified then $h_t(\mathbf{x}_i) = y_i$ so at the next iteration t+1 its importance (probability distribution) will be decreased to $D^{(t+1)}(i) = \dfrac{D^{(t)}(i) \times e^{-w_t}}{Z_{t+1}}$

If example x_i is misclassified then $h_t(x_i) \neq y_i$ so at the next iteration t+1 its importance (probability distribution) will be increased to $D^{(t+1)}(i) = \dfrac{D^{(t)}(i) \times e^{w_t}}{Z_{t+1}}$

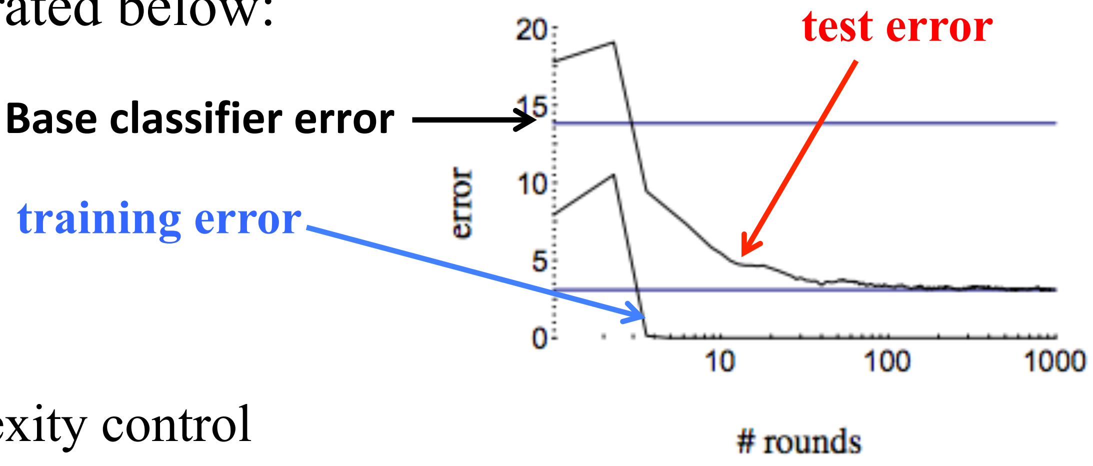
- output final/combined classifier h_{final}: $h_{final}(x) = sign(\sum_{t=1}^{T} w_t h_t(x))$

Recap - Generalization error for AdaBoost

$\text{VCdim}(L(\mathcal{B},T)) \leq T \times (\text{VCdim}(\mathcal{B})+1) \times (3\times\log(T \times (\text{VCdim}(\mathcal{B})+1))+2)$.

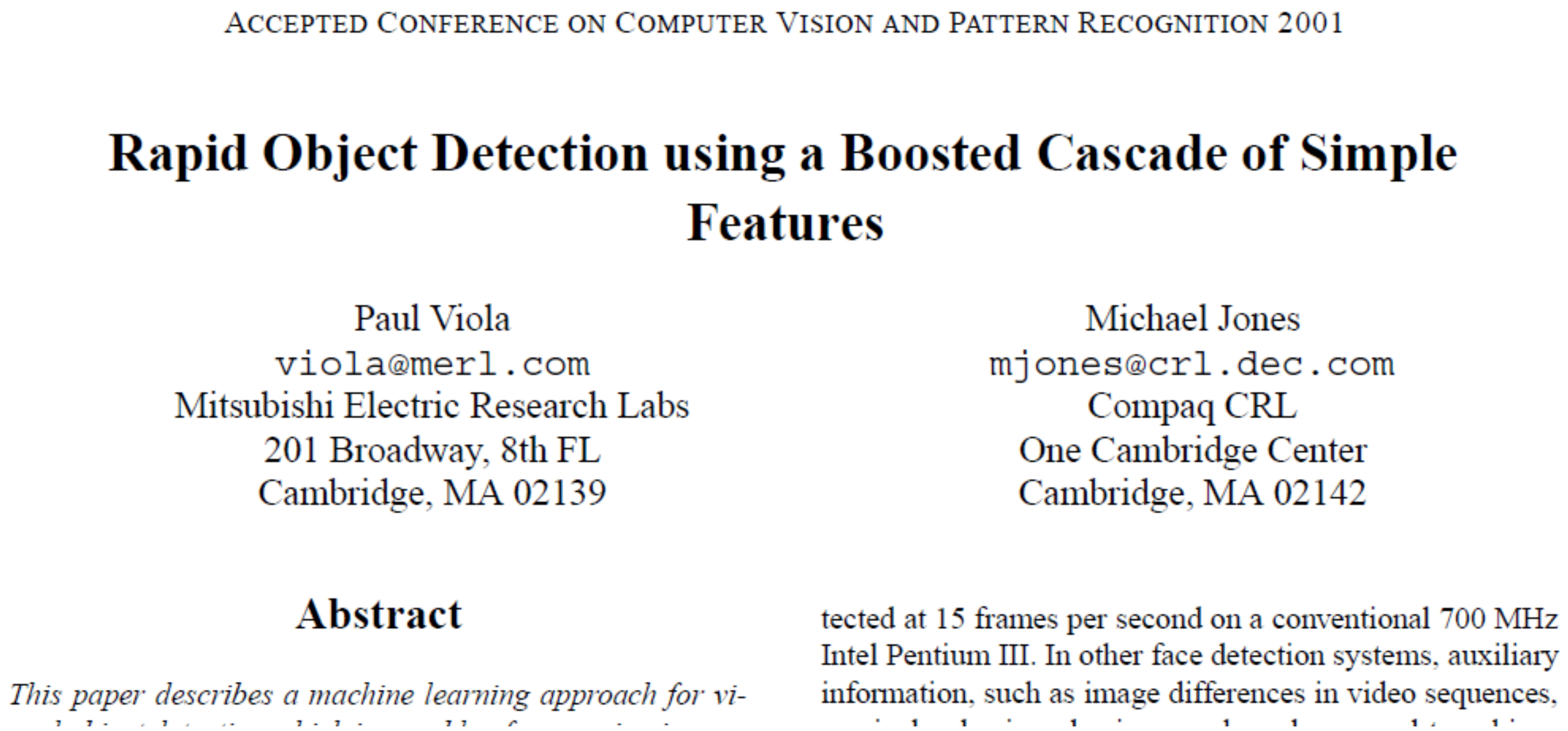
The upper bound grows as $O(T\times\text{VCdim}(\mathcal{B}) \times \log(T\times\text{VCdim}(\mathcal{B})))$, thus, the bound suggests that AdaBoost could overfit for large values of T, and indeed this can occur.

However, in many cases, it has been observed empirically that the generalization error of AdaBoost decreases as a function of the number of rounds of boosting T, as illustrated below:



- number of rounds *T* is complexity control
- use validation set + "early stopping" to select *T*

Recap: Viola-Jones face detector



ACCEPTED CONFERENCE ON COMPUTER VISION AND PATTERN RECOGNITION 2001

Rapid Object Detection using a Boosted Cascade of Simple Features

Paul Viola
viola@merl.com
Mitsubishi Electric Research Labs
201 Broadway, 8th FL
Cambridge, MA 02139

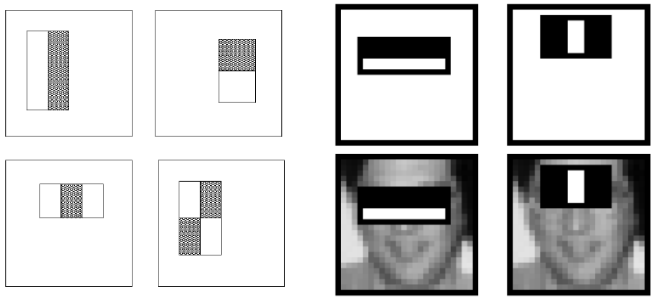
Michael Jones
mjones@crl.dec.com
Compaq CRL
One Cambridge Center
Cambridge, MA 02142

Abstract

This paper describes a machine learning approach for vi-

tected at 15 frames per second on a conventional 700 MHz Intel Pentium III. In other face detection systems, auxiliary information, such as image differences in video sequences,

Recap: Viola-Jones face detector: features

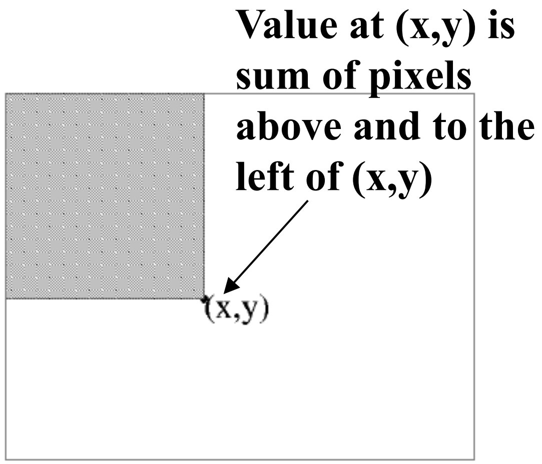


“Rectangular” filters

Feature output is difference between adjacent regions

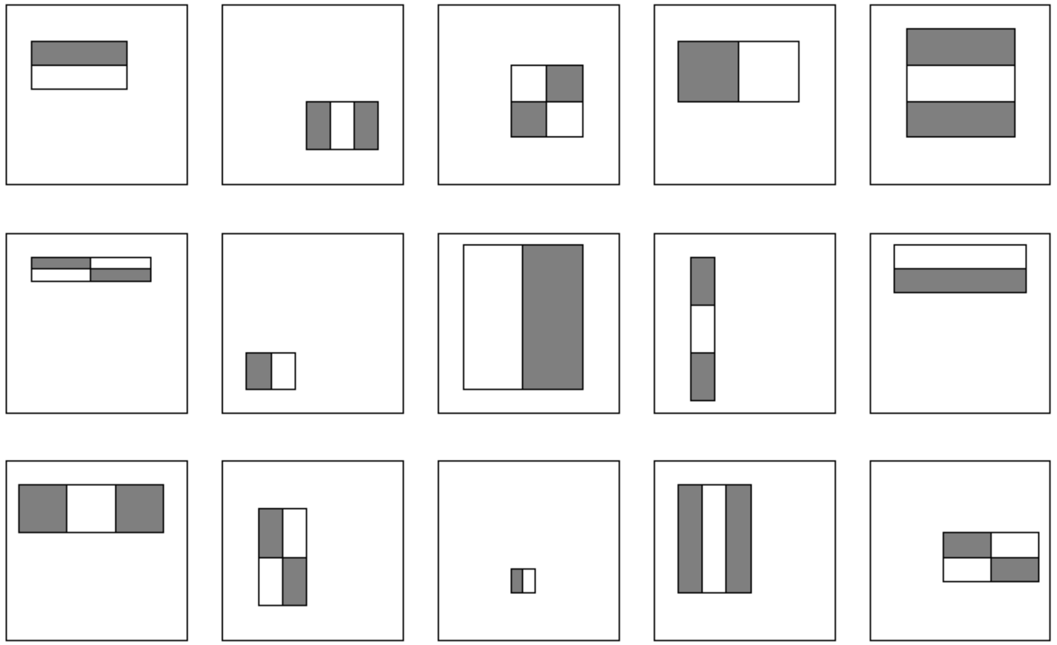
Efficiently computable with integral image: any sum can be computed in constant time

Avoid scaling images → scale features directly for same cost


Value at (x,y) is sum of pixels above and to the left of (x,y)
(x,y)

Integral image

Recap: Viola-Jones face detector: features



Considering all possible filter parameters: position, scale, and type:

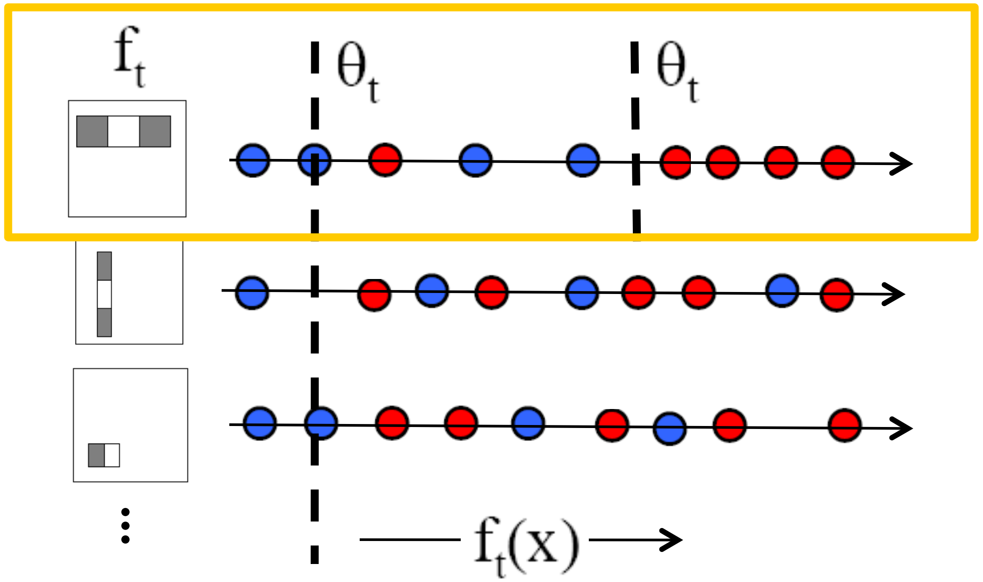
180,000+ possible features associated with each 24 x 24 window

Which subset of these features should we use to determine if a window has a face?

Use AdaBoost both to select the informative features and to form the classifier

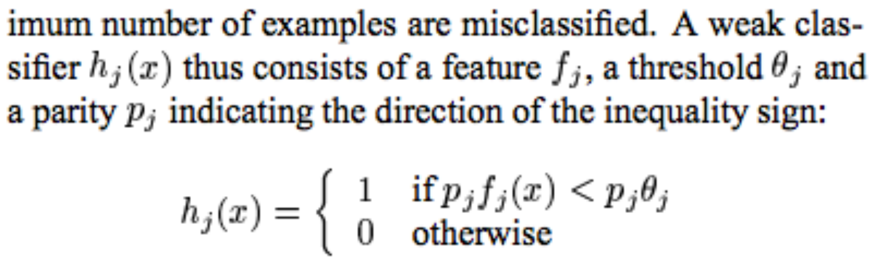
Recap: Viola-Jones detector: AdaBoost

- Want to select the single rectangle feature and threshold that best separates positive (faces) and negative (non-faces) training examples, in terms of *weighted* error.



Outputs of a possible rectangle feature on faces and non-faces.

Resulting weak classifier:



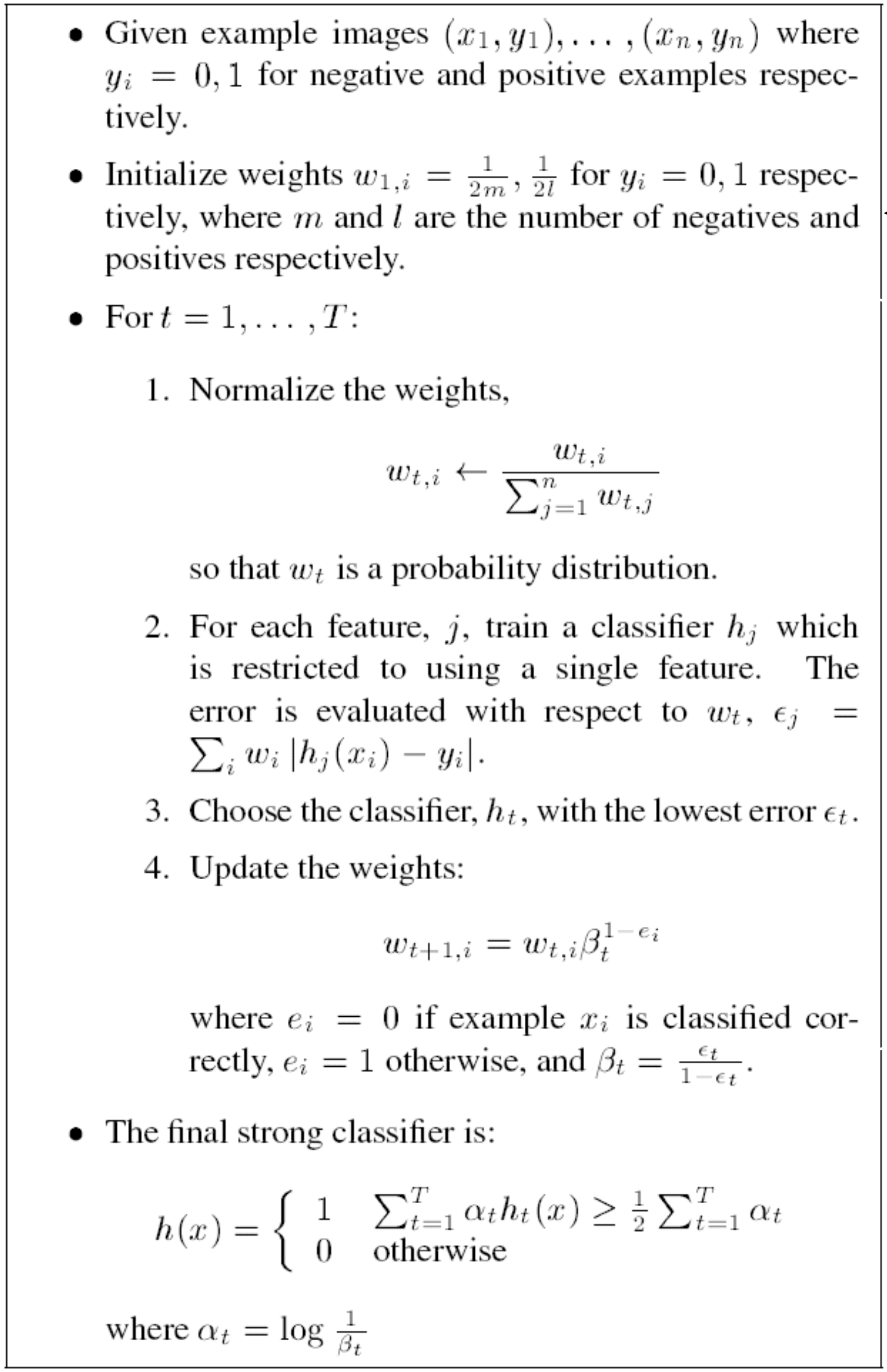
imum number of examples are misclassified. A weak classifier $h_j(x)$ thus consists of a feature f_j, a threshold θ_j and a parity p_j indicating the direction of the inequality sign:

$$h_j(x) = \begin{cases} 1 & \text{if } p_j f_j(x) < p_j \theta_j \\ 0 & \text{otherwise} \end{cases}$$

For next round, reweight the examples according to errors, choose another filter/threshold combo.

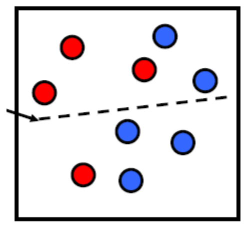
The resulting weak classifier is in fact from $\mathcal{H}_{DS}{}^{d} = \{h_{i,\theta,b}: \mathbf{R}^d \rightarrow \{-1,1\},$
$h_{i,\theta,b}(\mathbf{x}) = \text{sign}(\theta - x_i) \times b,\ 1 \leq i \leq d,\ \theta \in \mathbf{R},\ b \in \{-1,+1\}\}$

AdaBoost Algorithm



- Given example images $(x_1, y_1), \ldots, (x_n, y_n)$ where $y_i = 0, 1$ for negative and positive examples respectively.
- Initialize weights $w_{1,i} = \frac{1}{2m}, \frac{1}{2l}$ for $y_i = 0, 1$ respectively, where m and l are the number of negatives and positives respectively.
- For $t = 1, \ldots, T$:
  1. Normalize the weights,
     $$w_{t,i} \leftarrow \frac{w_{t,i}}{\sum_{j=1}^{n} w_{t,j}}$$
     so that w_t is a probability distribution.
  2. For each feature, j, train a classifier h_j which is restricted to using a single feature. The error is evaluated with respect to w_t, $\epsilon_j = \sum_i w_i \, |h_j(x_i) - y_i|$.
  3. Choose the classifier, h_t, with the lowest error ϵ_t.
  4. Update the weights:
     $$w_{t+1,i} = w_{t,i}\beta_t^{1-e_i}$$
     where $e_i = 0$ if example x_i is classified correctly, $e_i = 1$ otherwise, and $\beta_t = \frac{\epsilon_t}{1-\epsilon_t}$.
- The final strong classifier is:
  $$h(x) = \begin{cases} 1 & \sum_{t=1}^{T} \alpha_t h_t(x) \geq \frac{1}{2} \sum_{t=1}^{T} \alpha_t \\ 0 & \text{otherwise} \end{cases}$$
  where $\alpha_t = \log \frac{1}{\beta_t}$

Start with uniform weights on training examples



$\{\mathbf{x}_1, \ldots \mathbf{x}_n\}$

For T rounds

Evaluate *weighted* error for each feature, pick best.

Re-weight the examples:
Incorrectly classified -> more weight
Correctly classified -> less weight

Final classifier is combination of the weak ones, weighted according to error they had.

Freund & Schapire 1995

Today’s lecture: Overview

- SVMs

Support Vector Machines (SVMs)

SVMs

SVM Definition

- A **Support Vector Machine (SVM)** is a *non-probabilistic binary linear classifier*.
  - **Classifier** – A supervised learning method which predicts a categorical class.
  - **Linear** – The decision boundary is an n-dimensional hyperplane.
  - **Binary** – It can learn to predict one of two classes (a "+" class and a "-" class).
  - **Non-probabilistic** – The output of its *decision function* is not bounded, so it cannot be interpreted as probability.

There are methods of adapting an SVM to be used for **regression** problems and for making it **non-linear**, **multiclass** and/or **probabilistic**.

- An SVM tries to find a *separating* hyperplane, which is as far away from all training points at the same time (a **maximum-margin hyperplane**)
  - A point is classified as "+" or "-", depending on which part of the hyperplane it lies.

Slide credit Sparktech

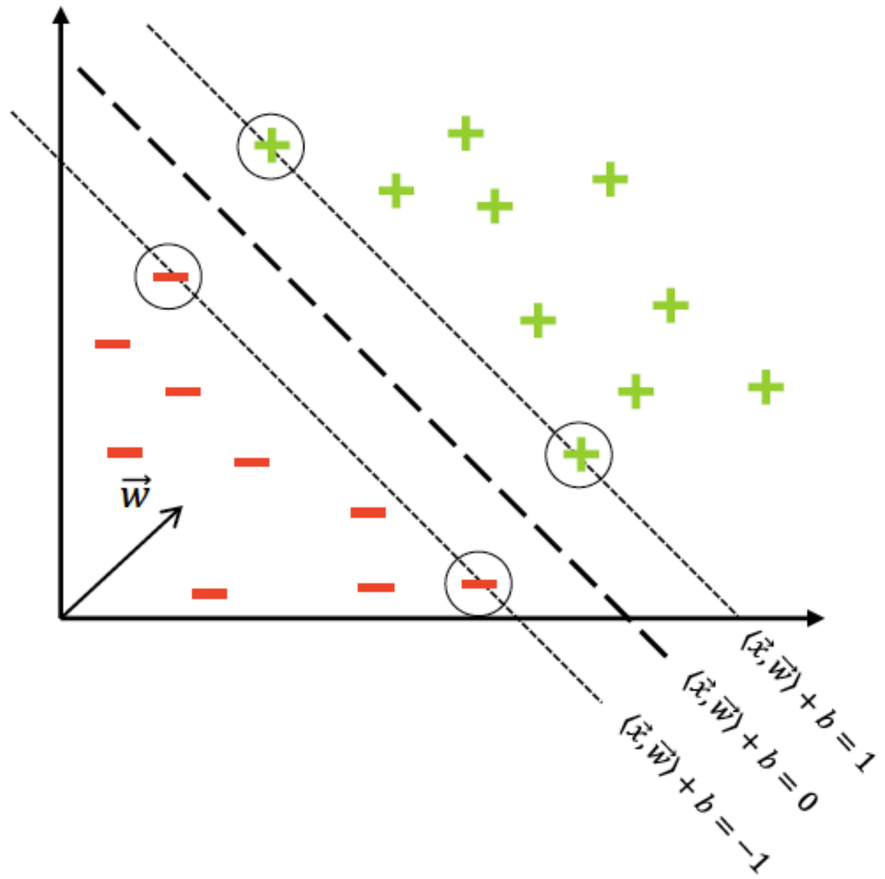
SVMs

Learning a "good" separating hyperplane

- For training examples $(\vec{x}^{(i)}, y^{(i)})$, which lie exactly on the edges of the gap:

$$\boxed{y^{(i)}\left(\langle \vec{x}^{(i)}, \vec{w} \rangle + b\right) - 1 = 0}$$

- We call these examples "**Support Vectors**"



$\vec{w}$
$\langle \vec{x}, \vec{w} \rangle + b = 1$
$\langle \vec{x}, \vec{w} \rangle + b = 0$
$\langle \vec{x}, \vec{w} \rangle + b = -1$

Slide credit Sparktech

SVMs

Making the margin "wide"

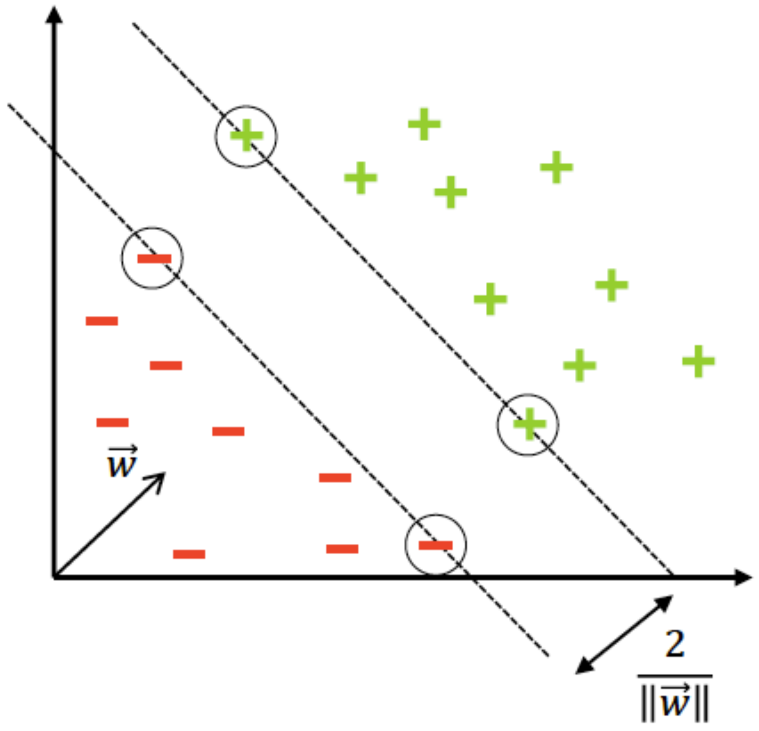
- How do we express the width of the gap?

$$g = \frac{2}{\|\vec{w}\|}$$

- We want to *maximize the gap:*

$$\text{maximize} \frac{2}{\|\vec{w}\|} \Rightarrow \text{minimize } \|\vec{w}\| \Rightarrow$$

$$\boxed{\text{minimize} \frac{\|\vec{w}\|^2}{2}}$$



Slide credit Sparktech

SVMs

What we have so far

SVM Primal Form

- The decision rule is:

$$\vec{x} \text{ is a "+" sample if } \langle \vec{x}, \vec{w} \rangle + b \geq 0$$

- In order to obtain $\vec{w}$ and b we need to:

$$\text{minimize } \frac{\|\vec{w}\|^2}{2}$$

$$\text{subject to } y^{(i)}\left(\langle \vec{x}^{(i)}, \vec{w} \rangle + b\right) - 1 \geq 0$$

Slide credit Sparktech

Recap: The fundamental theorem of statistical learning – quantitative version

Theorem

Let $\mathcal{H}$ be a hypothesis class of functions from a domain $\mathcal{X}$ to $\{0,1\}$ and let the loss function be the 0−1 loss. Assume that VCdim($\mathcal{H}$) = d <∞. Then, there are absolute constants C_1, C_2 such that:

1. $\mathcal{H}$ has the uniform convergence property with sample complexity:

$$C_1 \frac{d+\log(1/\delta)}{\epsilon^2} \leq m_{\mathcal{H}}^{UC}(\epsilon,\delta) \leq C_2 \frac{d+\log(1/\delta)}{\epsilon^2}$$

2. $\mathcal{H}$ is agnostic PAC learnable with sample complexity:

$$C_1 \frac{d+\log(1/\delta)}{\epsilon^2} \leq m_{\mathcal{H}}(\epsilon,\delta) \leq C_2 \frac{d+\log(1/\delta)}{\epsilon^2}$$

3. $\mathcal{H}$ is PAC learnable with sample complexity:

$$C_1 \frac{d+\log(1/\delta)}{\epsilon} \leq m_{\mathcal{H}}(\epsilon,\delta) \leq C_2 \frac{d\log(1/\epsilon)+\log(1/\delta)}{\epsilon}$$

The VC dimension determines (along with ε, δ) the samples complexities of learning a class. It gives us a lower and an upper bound.

15.1.2 The Sample Complexity of Hard-SVM

Recall that the VC-dimension of halfspaces in $\mathbb{R}^d$ is $d+1$. It follows that the sample complexity of learning halfspaces grows with the dimensionality of the problem. Furthermore, the fundamental theorem of learning tells us that if the number of examples is significantly smaller than d/ϵ then no algorithm can learn an ϵ-accurate halfspace. This is problematic when d is very large.

To overcome this problem, we will make an additional assumption on the underlying data distribution. In particular, we will define a "separability with margin γ" assumption and will show that if the data is separable with margin γ then the sample complexity is bounded from above by a function of $1/\gamma^2$. It follows that even if the dimensionality is very large (or even infinite), as long as the data adheres to the separability with margin assumption we can still have a small sample complexity. There is no contradiction to the lower bound given in the fundamental theorem of learning because we are now making an additional assumption on the underlying data distribution.

Hard SVM

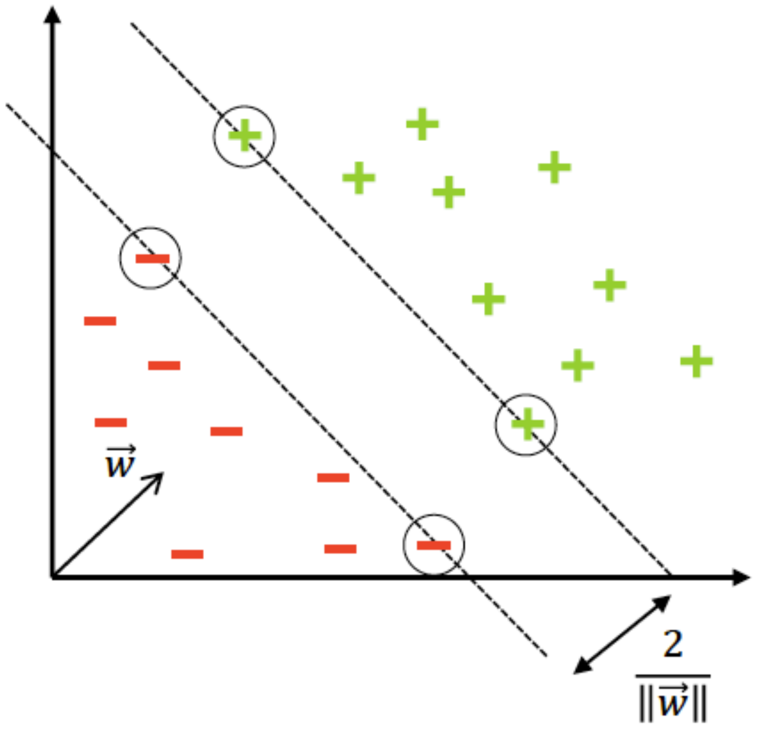
Hard-SVM is the learning rule in which we return an ERM hyperplane that separates the training set $S = \{(\mathbf{x}_1, y_1), (\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_m, y_m)\}$ with the largest possible margin.

$$\text{minimize } \frac{\|\vec{w}\|^2}{2}$$

$$\text{subject to } y^{(i)}\left(\langle \vec{x}^{(i)}, \vec{w}\rangle + b\right) - 1 \geq 0$$

$$\Big\updownarrow \quad \gamma = \frac{1}{\|w\|}$$

$$\boxed{\begin{array}{l} \textit{minimize } \dfrac{1}{2\gamma^2} \\ \textit{subject to } y^{(i)}\left(\left\langle x^{(i)}, \dfrac{w}{\|w\|}\right\rangle + \dfrac{b}{\|w\|}\right) - \dfrac{1}{\|w\|} \geq 0 \end{array}}$$



Hard SVM

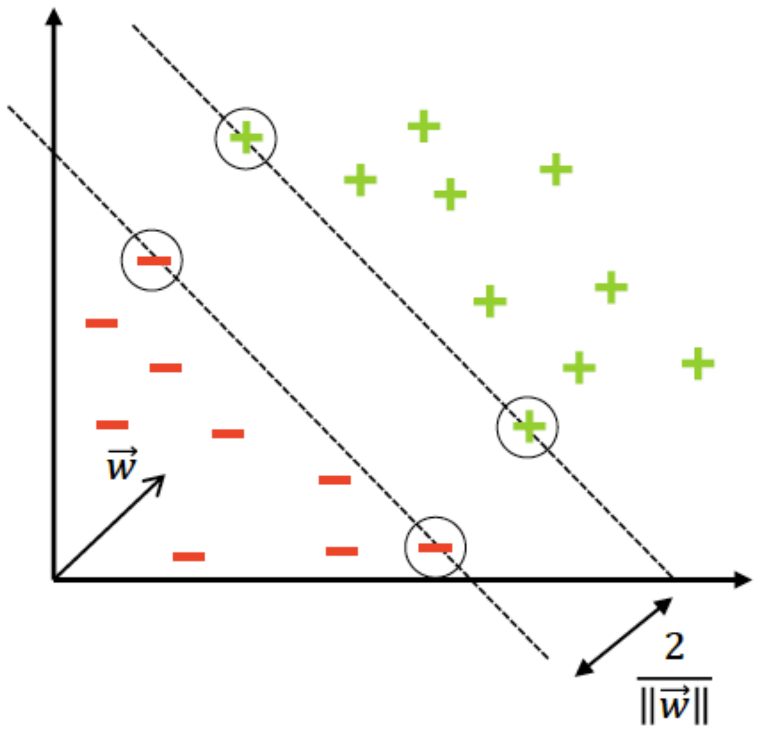
Hard-SVM is the learning rule in which we return an ERM hyperplane that separates the training set $S = \{(x_1, y_1), (x_1, y_1), \ldots, (x_m, y_m)\}$ with the largest possible margin.

$$\text{minimize } \frac{\|\vec{w}\|^2}{2}$$

$$\text{subject to } y^{(i)}\left(\langle \vec{x}^{(i)}, \vec{w}\rangle + b\right) - 1 \geq 0$$

$$\Updownarrow \quad \gamma = \frac{1}{\|w\|}$$

$$\boxed{\begin{array}{l} \textbf{maximize } \gamma^2 \\ \textbf{subject to } y^{(i)}\left(\left\langle x^{(i)}, \frac{w}{\|w\|}\right\rangle + \frac{b}{\|w\|}\right) \geq \gamma \end{array}}$$



Hard SVM

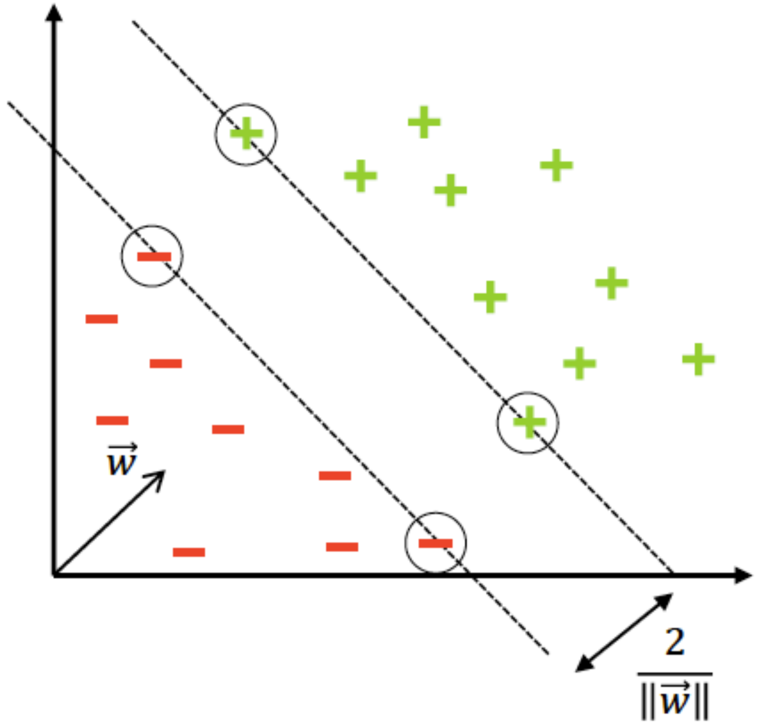
Hard-SVM is the learning rule in which we return an ERM hyperplane that separates the training set S = $\{(x_1, y_1), (x_1, y_1), \ldots, (x_m, y_m)\}$ with the largest possible margin.

$$\text{minimize } \frac{\|\vec{w}\|^2}{2}$$
$$\text{subject to } y^{(i)}\left(\langle \vec{x}^{(i)}, \vec{w}\rangle + b\right) - 1 \geq 0$$

$$\Updownarrow \quad \gamma = \frac{1}{\|w\|}$$

maximize γ

subject to $y^{(i)}\left(\left\langle x^{(i)}, \frac{w}{\|w\|}\right\rangle + \frac{b}{\|w\|}\right) \geq \gamma$



Hard SVM

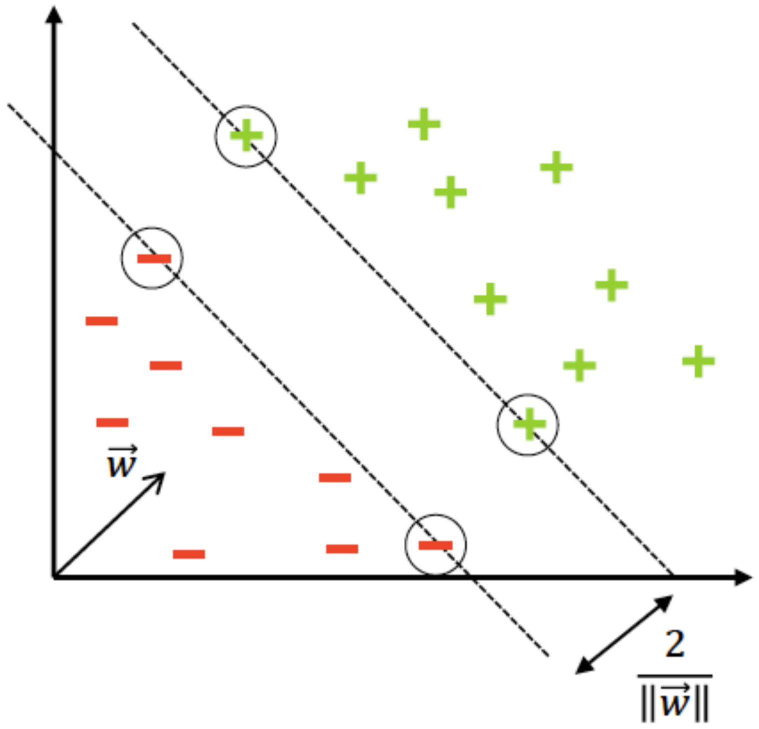
Hard-SVM is the learning rule in which we return an ERM hyperplane that separates the training set S = {(x_1, y_1), (x_1, y_1), …, (x_m, y_m)} with the largest possible margin.

$$\text{minimize} \quad \frac{\|\vec{w}\|^2}{2}$$

$$\text{subject to } y^{(i)}\left(\langle \vec{x}^{(i)}, \vec{w}\rangle + b\right) - 1 \geq 0$$

$$\Updownarrow \quad \gamma = \frac{1}{\|w\|}$$

$$\boxed{\operatorname*{argmax}_{w,b} \min_{i=1,m} y^{(i)}\left(\left\langle x^{(i)}, \frac{w}{\|w\|}\right\rangle + \frac{b}{\|w\|}\right)}$$



Hard SVM

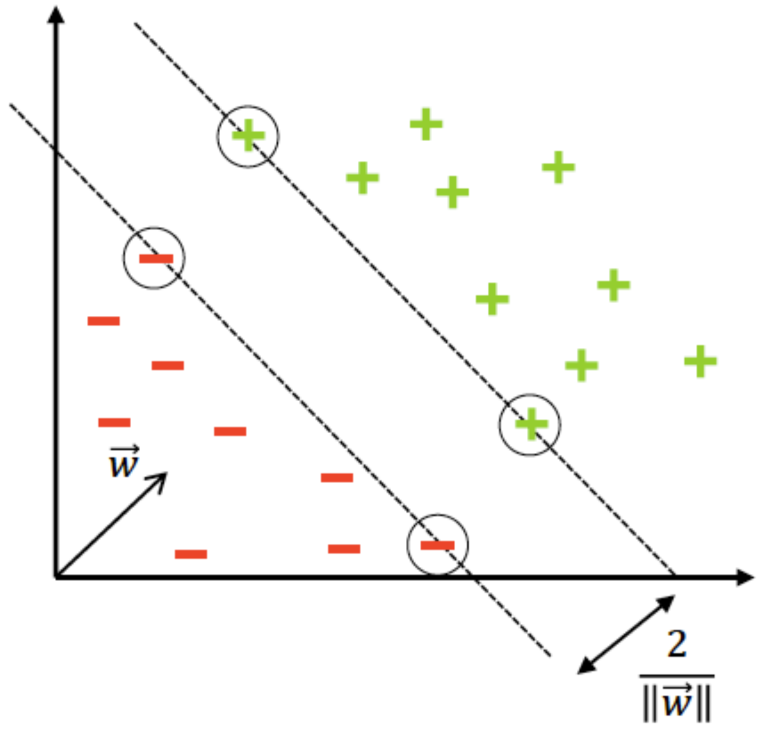
Hard-SVM is the learning rule in which we return an ERM hyperplane that separates the training set S = $\{(x_1, y_1), (x_1, y_1), \ldots, (x_m, y_m)\}$ with the largest possible margin.

$$\text{minimize} \quad \frac{\|\vec{w}\|^2}{2}$$

$$\text{subject to } y^{(i)}\left(\langle \vec{x}^{(i)}, \vec{w}\rangle + b\right) - 1 \geq 0$$

$$\Updownarrow \quad \gamma = \frac{1}{\|w\|}$$

$$\boxed{\operatorname*{argmax}_{w_0, b_0} \min_{i=1,m} y^{(i)}\left(\left\langle x^{(i)}, \frac{w_0}{\|w_0\|}\right\rangle + \frac{b_0}{\|w_0\|}\right)}$$



Hard SVM

Hard-SVM is the learning rule in which we return an ERM hyperplane that separates the training set S = {(x_1, y_1), (x_1, y_1), …, (x_m, y_m)} with the largest possible margin.

$$\text{minimize } \frac{\|\vec{w}\|^2}{2}$$

$$\text{subject to } y^{(i)}(\langle \vec{x}^{(i)}, \vec{w}\rangle + b) - 1 \geq 0$$

Hard-SVM

input: $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_m, y_m)$

solve:

$$(\mathbf{w}_0, b_0) = \underset{(\mathbf{w}, b)}{\operatorname{argmin}} \|\mathbf{w}\|^2 \text{ s.t. } \forall i,\ y_i(\langle \mathbf{w}, \mathbf{x}_i\rangle + b) \geq 1$$

output: $\hat{\mathbf{w}} = \frac{\mathbf{w}_0}{\|\mathbf{w}_0\|},\ \hat{b} = \frac{b_0}{\|\mathbf{w}_0\|}$

$$\gamma = \frac{1}{\|w\|}$$

$$\underset{w_0, b_0}{\operatorname{argmax}} \min_{i=1,m} y^{(i)}\left(\left\langle x^{(i)}, \frac{w_0}{\|w_0\|}\right\rangle + \frac{b_0}{\|w_0\|}\right)$$

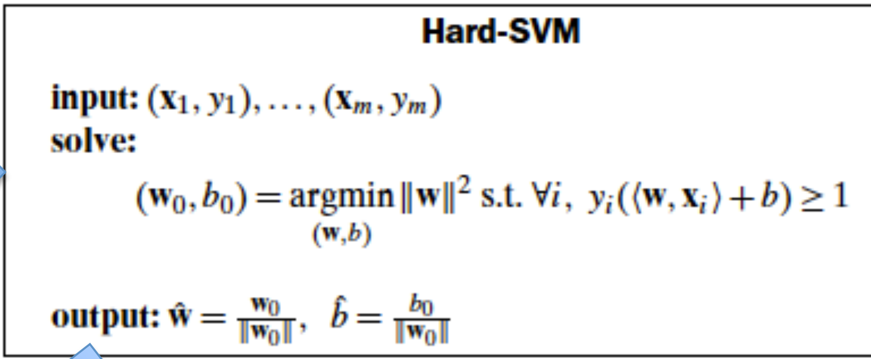
Hard SVM

Hard-SVM is the learning rule in which we return an ERM hyperplane that separates the training set $S = \{(x_1, y_1), (x_1, y_1), \ldots, (x_m, y_m)\}$ with the largest possible margin.

$$\text{minimize } \frac{\|\vec{w}\|^2}{2}$$

$$\text{subject to } y^{(i)}(\langle \vec{x}^{(i)}, \vec{w}\rangle + b) - 1 \geq 0$$

$$\gamma = \frac{1}{\|w\|}$$



Hard-SVM

input: $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_m, y_m)$
solve:

$$(\mathbf{w}_0, b_0) = \operatorname*{argmin}_{(\mathbf{w},b)} \|\mathbf{w}\|^2 \text{ s.t. } \forall i,\ y_i(\langle \mathbf{w}, \mathbf{x}_i\rangle + b) \geq 1$$

output: $\hat{\mathbf{w}} = \frac{\mathbf{w}_0}{\|\mathbf{w}_0\|},\ \hat{b} = \frac{b_0}{\|\mathbf{w}_0\|}$

$$\operatorname*{argmax}_{\|w\|=1,b} \min_{i=1,m} y^{(i)}\left(\left\langle x^{(i)}, w\right\rangle + b\right)$$

The sample complexity of Hard SVM

The sample complexity of Hard SVM

Consider $\mathcal{H} = \mathcal{HS}^d$ be the set of halfspaces (linear classifiers) in $\mathbf{R}^d$

$$\mathcal{H} = \mathcal{HS}^d = \{h_{w,b}: \mathbf{R}^d \to \{-1, 1\},\ h_{w,b}(x) = sign\left(\sum_{i=1}^{d} w_i x_i + b\right) \mid w \in \mathbf{R}^d, b \in \mathbf{R}\}$$

"Homogenous" linear classifiers: b = 0.

$$\mathcal{HS}_0{}^d = \{h_{w,0}: \mathbf{R}^d \to \{-1, 1\},\ h_{w,0}(x) = sign\left(\sum_{i=1}^{d} w_i x_i\right) \mid w \in \mathbf{R}^d\}$$

VCdim($\mathcal{HS}_0{}^d$) = d (proof in Lecture 7)

VCdim($\mathcal{HS}^d$) = d + 1 (proof in the book, easy to extend from the one given in the lecture)

The sample complexity of Hard SVM

$\mathcal{H} = \mathcal{H}S^d$, VCdim($\mathcal{H}$) = d <∞. From the fundamental theorem of statistical learning we have that there are absolute constants C_1, C_2 such that:
$\mathcal{H}$ is PAC learnable with sample complexity:

$$C_1 \frac{d + \log(1/\delta)}{\epsilon} \le m_{\mathcal{H}}(\epsilon, \delta) \le C_2 \frac{d \log(1/\epsilon) + \log(1/\delta)}{\epsilon}$$

The sample complexity of learning halfspaces $m_{\mathcal{H}}(\varepsilon,\delta)$ grows with the dimensionality *d* of the problem. Furthermore, the fundamental theorem of learning tells us that if the number of examples is significantly smaller than d/ε then no algorithm can learn an accurate halfspace. This is problematic when *d* is very large.

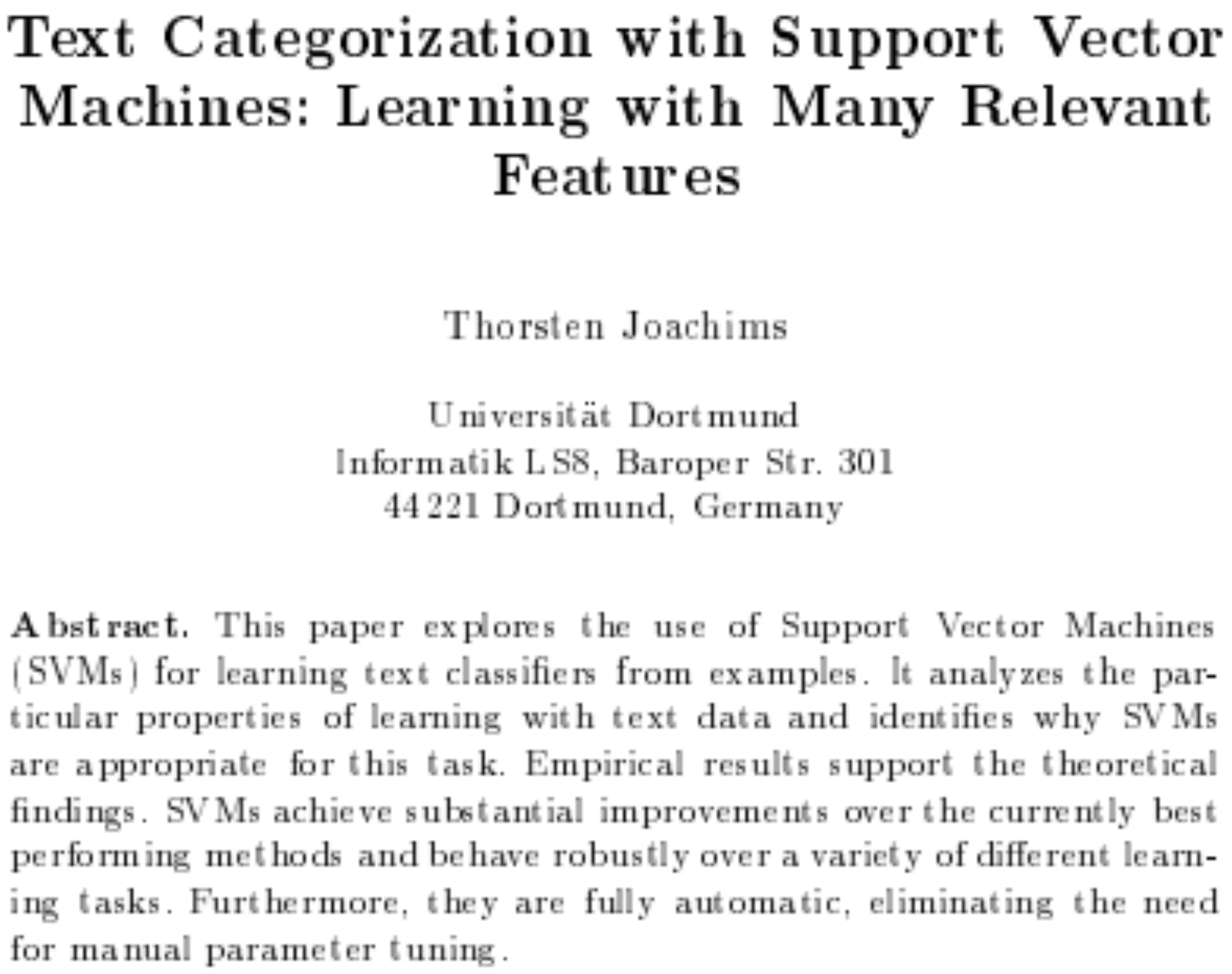
Text classification with bag-of-words

- classify a short text document according to its topic, say, whether the document is about sports or not.

- represent documents as vectors.
  - effective way is to use a bag-of-words representation.
  - define a dictionary of words and set the dimension d to be the number of words in the dictionary.
  - we represent a document as a vector $\mathbf{x} \in \{0,1\}^d$, where $x_i = 1$ if the i-th word in the dictionary appears in the document and $x_i = 0$ otherwise.

- a halfspace for the problem of text classification assigns weights to words

- common to have $d >$ number of training examples. In practice the problem is solvable, we can categorize text based on BOW.



Text Categorization with Support Vector Machines: Learning with Many Relevant Features

Thorsten Joachims

Universität Dortmund
Informatik LS8, Baroper Str. 301
44221 Dortmund, Germany

Abstract. This paper explores the use of Support Vector Machines (SVMs) for learning text classifiers from examples. It analyzes the particular properties of learning with text data and identifies why SVMs are appropriate for this task. Empirical results support the theoretical findings. SVMs achieve substantial improvements over the currently best performing methods and behave robustly over a variety of different learning tasks. Furthermore, they are fully automatic, eliminating the need for manual parameter tuning.

https://www.cs.cornell.edu/people/tj/publications/joachims_98a.pdf

Sample complexity for separability case

- if the number of examples is significantly smaller than d/ε then no algorithm can learn an accurate halfspace.
  - this is problematic when d is very large.

- make an additional assumption on the underlying data distribution
  - define a "separability with margin γ" assumption
  - if the data follows this assumption (is separable with margin γ) then the sample complexity is bounded from above by a function of $1/\gamma^2$.
  - even if the dimensionality d is very large (or even infinite), as long as the data adheres to the separability with margin assumption we can still have a small sample complexity.
  - ***the sample complexity will not depend on d***.
  - there is no contradiction to the lower bound given in the fundamental theorem of learning because we are now making an additional assumption on the underlying data distribution.

Sample complexity for separability case

If S = $\{(\mathbf{x_1},y_1), (\mathbf{x_2},y_2),\ldots, (\mathbf{x_m},y_m)\}$ separable with margin γ then we have that

S' = $\{(\alpha\mathbf{x_1},y_1), (\alpha\mathbf{x_2},y_2),\ldots, (\alpha\mathbf{x_m},y_m)\}$ is separable with margin $\alpha\gamma$ for any $\alpha > 0$.

Definition 15.3. Let $\mathcal{D}$ be a distribution over $\mathbb{R}^d \times \{\pm 1\}$. We say that $\mathcal{D}$ is separable with a (γ,ρ)-margin if there exists $(\mathbf{w}^\star, b^\star)$ such that $\|\mathbf{w}^\star\| = 1$ and such that with probability 1 over the choice of $(\mathbf{x}, y) \sim \mathcal{D}$ we have that $y(\langle \mathbf{w}^\star, \mathbf{x}\rangle + b^\star) \geq \gamma$ and $\|\mathbf{x}\| \leq \rho$. Similarly, we say that $\mathcal{D}$ is separable with a (γ,ρ)-margin using a homogenous halfspace if the preceding holds with a halfspace of the form $(\mathbf{w}^\star, 0)$.

$$\text{minimize } \frac{\|\vec{w}\|^2}{2}$$
$$\text{subject to } y^{(i)}\left(\langle \vec{x}^{(i)}, \vec{w}\rangle + b\right) - 1 \geq 0$$

$$\gamma = \frac{1}{\|w\|}$$

Hard-SVM

input: $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_m, y_m)$

solve:

$$(\mathbf{w}_0, b_0) = \underset{(\mathbf{w},b)}{\operatorname{argmin}} \|\mathbf{w}\|^2 \text{ s.t. } \forall i,\ y_i(\langle \mathbf{w}, \mathbf{x}_i\rangle + b) \geq 1$$

output: $\hat{\mathbf{w}} = \frac{\mathbf{w}_0}{\|\mathbf{w}_0\|}$, $\hat{b} = \frac{b_0}{\|\mathbf{w}_0\|}$

$$\underset{\|w\|=1,b}{\operatorname{argmax}} \min_{i=1,m} y^{(i)}\left(\left\langle x^{(i)}, w\right\rangle + b\right)$$

Sample complexity for separability case

If S = $\{(\mathbf{x_1},y_1), (\mathbf{x_2},y_2),\ldots, (\mathbf{x_m},y_m)\}$ separable with margin γ then we have that

S' = $\{(\alpha\mathbf{x_1},y_1), (\alpha\mathbf{x_2},y_2),\ldots, (\alpha\mathbf{x_m},y_m)\}$ is separable with margin $\alpha\gamma$ for any $\alpha > 0$.

Definition 15.3. Let $\mathcal{D}$ be a distribution over $\mathbb{R}^d \times \{\pm 1\}$. We say that $\mathcal{D}$ is separable with a (γ,ρ)-margin if there exists $(\mathbf{w}^\star, b^\star)$ such that $\|\mathbf{w}^\star\| = 1$ and such that with probability 1 over the choice of $(\mathbf{x},y) \sim \mathcal{D}$ we have that $y(\langle \mathbf{w}^\star, \mathbf{x}\rangle + b^\star) \geq \gamma$ and $\|\mathbf{x}\| \leq \rho$. Similarly, we say that $\mathcal{D}$ is separable with a (γ,ρ)-margin using a homogenous halfspace if the preceding holds with a halfspace of the form $(\mathbf{w}^\star, 0)$.

It can be shown that the sample complexity of Hard-SVM depends on $(\rho/\gamma)^2$ and is independent of the dimension d.

Theorem 15.4. *Let $\mathcal{D}$ be a distribution over $\mathbb{R}^d \times \{\pm 1\}$ that satisfies the (γ,ρ)-separability with margin assumption using a homogenous halfspace. Then, with probability of at least $1-\delta$ over the choice of a training set of size m, the 0-1 error of the output of Hard-SVM is at most*

$$\sqrt{\frac{4(\rho/\gamma)^2}{m}} + \sqrt{\frac{2\log(2/\delta)}{m}}.$$

Soft SVM

```
Hard-SVM

input: $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_m, y_m)$
solve:
    $(\mathbf{w}_0, b_0) = \underset{(\mathbf{w},b)}{\operatorname{argmin}} \|\mathbf{w}\|^2 \text{ s.t. } \forall i,\; y_i(\langle \mathbf{w}, \mathbf{x}_i \rangle + b) \geq 1$    (15.2)
output: $\hat{\mathbf{w}} = \frac{\mathbf{w}_0}{\|\mathbf{w}_0\|},\quad \hat{b} = \frac{b_0}{\|\mathbf{w}_0\|}$
```

```
Soft-SVM

input: $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_m, y_m)$
parameter: $\lambda > 0$
solve:
    $\min_{\mathbf{w},b,\boldsymbol{\xi}} \left( \lambda \|\mathbf{w}\|^2 + \frac{1}{m} \sum_{i=1}^{m} \xi_i \right)$
    s.t. $\forall i,\; y_i(\langle \mathbf{w}, \mathbf{x}_i \rangle + b) \geq 1 - \xi_i$ and $\xi_i \geq 0$    (15.4)
output: $\mathbf{w}, b$
```

The optimization problem in Equation (15.2) enforces the hard constraints $y_i(\langle \mathbf{w}, \mathbf{x}_i \rangle + b) \geq 1$ for all i. A natural relaxation is to allow the constraint to be violated for some of the examples in the training set. This can be modeled by introducing nonnegative slack variables, $\xi_1, \ldots, \xi_m$, and replacing each constraint $y_i(\langle \mathbf{w}, \mathbf{x}_i \rangle + b) \geq 1$ by the constraint $y_i(\langle \mathbf{w}, \mathbf{x}_i \rangle + b) \geq 1 - \xi_i$. That is, ξ_i measures by how much the constraint $y_i(\langle \mathbf{w}, \mathbf{x}_i \rangle + b) \geq 1$ is being violated. Soft-SVM jointly minimizes the norm of $\mathbf{w}$ (corresponding to the margin) and the average of ξ_i (corresponding to the violations of the constraints). The tradeoff between the two terms is controlled by a parameter λ. '

Soft SVM

Hard-SVM

input: $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_m, y_m)$
solve:

$$(\mathbf{w}_0, b_0) = \underset{(\mathbf{w},b)}{\operatorname{argmin}} \|\mathbf{w}\|^2 \text{ s.t. } \forall i,\ y_i(\langle \mathbf{w}, \mathbf{x}_i \rangle + b) \geq 1 \qquad (15.2)$$

output: $\hat{\mathbf{w}} = \frac{\mathbf{w}_0}{\|\mathbf{w}_0\|}, \quad \hat{b} = \frac{b_0}{\|\mathbf{w}_0\|}$

Soft-SVM

input: $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_m, y_m)$
parameter: $\lambda > 0$
solve:

$$\min_{\mathbf{w},b,\boldsymbol{\xi}} \left(\lambda \|\mathbf{w}\|^2 + \frac{1}{m} \sum_{i=1}^m \xi_i \right) \qquad (15.4)$$
$$\text{s.t. } \forall i, \quad y_i(\langle \mathbf{w}, \mathbf{x}_i \rangle + b) \geq 1 - \xi_i \quad \text{and} \quad \xi_i \geq 0$$

output: $\mathbf{w}, b$

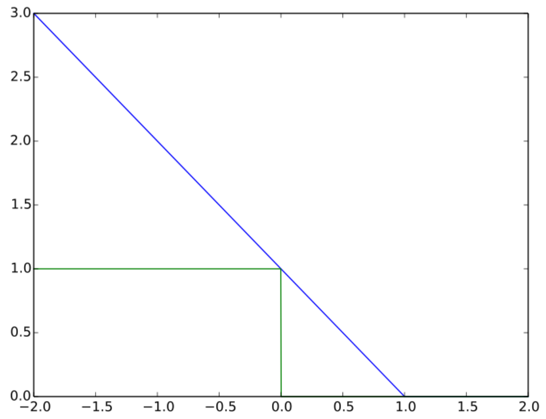
We can rewrite Equation (15.4) as a regularized loss minimization problem. Recall the definition of the hinge loss:

$$\ell^{\text{hinge}}((\mathbf{w}, b), (\mathbf{x}, y)) = \max\{0, 1 - y(\langle \mathbf{w}, \mathbf{x} \rangle + b)\}.$$

Given $(\mathbf{w}, b)$ and a training set S, the averaged hinge loss on S is denoted by $L_S^{\text{hinge}}((\mathbf{w}, b))$. Now, consider the regularized loss minimization problem:

$$\min_{\mathbf{w},b} \left(\lambda \|\mathbf{w}\|^2 + L_S^{\text{hinge}}((\mathbf{w}, b)) \right). \qquad (15.5)$$

CLAIM 15.5 *Equation (15.4) and Equation (15.5) are equivalent.*



Hinge loss vs **0-1 loss**

Soft SVM

We can rewrite Equation (15.4) as a regularized loss minimization problem. Recall the definition of the hinge loss:

$$\ell^{\text{hinge}}((\mathbf{w},b),(\mathbf{x},y)) = \max\{0, 1 - y(\langle \mathbf{w},\mathbf{x}\rangle + b)\}.$$

Given $(\mathbf{w}, b)$ and a training set S, the averaged hinge loss on S is denoted by $L_S^{\text{hinge}}((\mathbf{w}, b))$. Now, consider the regularized loss minimization problem:

$$\min_{\mathbf{w},b} \left(\lambda\|\mathbf{w}\|^2 + L_S^{\text{hinge}}((\mathbf{w},b))\right). \tag{15.5}$$

CLAIM 15.5 *Equation (15.4) and Equation (15.5) are equivalent.*

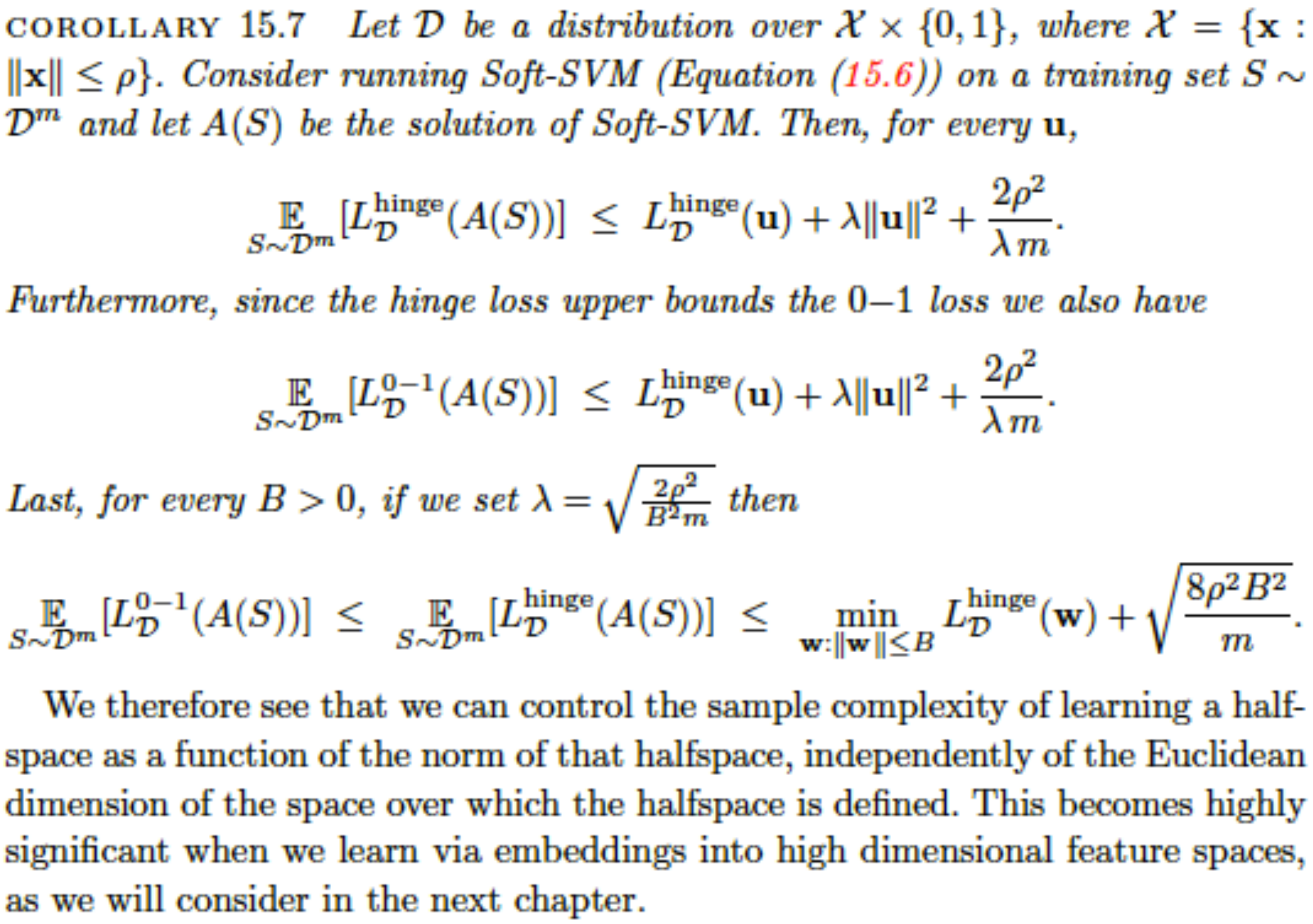
It is often more convenient to consider Soft-SVM for learning a homogenous halfspace, where the bias term b is set to be zero, which yields the following optimization problem:

$$\min_{\mathbf{w}} \left(\lambda\|\mathbf{w}\|^2 + L_S^{\text{hinge}}(\mathbf{w})\right), \tag{15.6}$$

where

$$L_S^{\text{hinge}}(\mathbf{w}) = \frac{1}{m}\sum_{i=1}^{m} \max\{0, 1 - y\langle \mathbf{w}, \mathbf{x}_i\rangle\}.$$

The sample complexity for Soft-SVM



COROLLARY 15.7 *Let $\mathcal{D}$ be a distribution over $\mathcal{X} \times \{0,1\}$, where $\mathcal{X} = \{\mathbf{x} : \|\mathbf{x}\| \le \rho\}$. Consider running Soft-SVM (Equation (15.6)) on a training set $S \sim \mathcal{D}^m$ and let $A(S)$ be the solution of Soft-SVM. Then, for every $\mathbf{u}$,*

$$\mathop{\mathbb{E}}_{S\sim\mathcal{D}^m}[L_{\mathcal{D}}^{\text{hinge}}(A(S))] \le L_{\mathcal{D}}^{\text{hinge}}(\mathbf{u}) + \lambda\|\mathbf{u}\|^2 + \frac{2\rho^2}{\lambda m}.$$

Furthermore, since the hinge loss upper bounds the 0–1 loss we also have

$$\mathop{\mathbb{E}}_{S\sim\mathcal{D}^m}[L_{\mathcal{D}}^{0-1}(A(S))] \le L_{\mathcal{D}}^{\text{hinge}}(\mathbf{u}) + \lambda\|\mathbf{u}\|^2 + \frac{2\rho^2}{\lambda m}.$$

Last, for every $B > 0$, if we set $\lambda = \sqrt{\frac{2\rho^2}{B^2 m}}$ then

$$\mathop{\mathbb{E}}_{S\sim\mathcal{D}^m}[L_{\mathcal{D}}^{0-1}(A(S))] \le \mathop{\mathbb{E}}_{S\sim\mathcal{D}^m}[L_{\mathcal{D}}^{\text{hinge}}(A(S))] \le \min_{\mathbf{w}:\|\mathbf{w}\|\le B} L_{\mathcal{D}}^{\text{hinge}}(\mathbf{w}) + \sqrt{\frac{8\rho^2 B^2}{m}}.$$

We therefore see that we can control the sample complexity of learning a halfspace as a function of the norm of that halfspace, independently of the Euclidean dimension of the space over which the halfspace is defined. This becomes highly significant when we learn via embeddings into high dimensional feature spaces, as we will consider in the next chapter.

Margin and norm based bounds vs dimension

Remember from lecture 7 (use of the Sauer lemma): for every $\mathcal{D}$ and every $\delta \in (0,1)$, with probability of at least $1 - \delta$ over the choice of $S \sim \mathcal{D}^m$ we have:

$$|L_D(h) - L_S(h)| \leq \frac{4 + \sqrt{\log(\tau_H(2m))}}{\delta\sqrt{2m}} \leq \frac{4 + \sqrt{d\log(2em/d)}}{\delta\sqrt{2m}} \leq \frac{2\sqrt{d\log(2em/d)}}{\delta\sqrt{2m}}$$

$d = \text{VCdim}(\mathcal{H})$, $\mathcal{H}$ is the class of halfspaces

probability of at least $1-\delta$ over the choice of a training set of size m, the 0-1 error of the output of Hard-SVM is at most

$$\sqrt{\frac{4(\rho/\gamma)^2}{m}} + \sqrt{\frac{2\log(2/\delta)}{m}}.$$

Loss for Hard-SVM

$$\mathop{\mathbb{E}}_{S\sim\mathcal{D}^m}[L_{\mathcal{D}}^{0-1}(A(S))] \leq \mathop{\mathbb{E}}_{S\sim\mathcal{D}^m}[L_{\mathcal{D}}^{\text{hinge}}(A(S))] \leq \min_{\mathbf{w}:\|\mathbf{w}\|\leq B} L_{\mathcal{D}}^{\text{hinge}}(\mathbf{w}) + \sqrt{\frac{8\rho^2B^2}{m}}.$$

Loss for Soft-SVM

The bounds we have derived for Hard-SVM and Soft-SVM do not depend on the dimension of the instance space. Instead, the bounds depend on the norm of the examples, ρ, the norm of the halfspace B (or equivalently the margin parameter γ) and, in the nonseparable case, the bounds also depend on the minimum hinge loss of all halfspaces of norm $\leq B$. In contrast, the VC-dimension of the class of homogenous halfspaces is d, which implies that the error of an ERM hypothesis decreases as $\sqrt{d/m}$ does. We now give an example in which $\rho^2B^2 \ll d$; hence the bound given in Corollary 15.7 is much better than the VC bound.

Margin and norm based bounds vs dimension

Consider the problem of learning to classify a short text document according to its topic, say, whether the document is about sports or not. We first need to represent documents as vectors. One simple yet effective way is to use a *bag-of-words* representation. That is, we define a dictionary of words and set the dimension d to be the number of words in the dictionary. Given a document, we represent it as a vector $\mathbf{x} \in \{0,1\}^d$, where $x_i = 1$ if the i'th word in the dictionary appears in the document and $x_i = 0$ otherwise. Therefore, for this problem, the value of ρ^2 will be the maximal number of distinct words in a given document.

A halfspace for this problem assigns weights to words. It is natural to assume that by assigning positive and negative weights to a few dozen words we will be able to determine whether a given document is about sports or not with reasonable accuracy. Therefore, for this problem, the value of B^2 can be set to be less than 100. Overall, it is reasonable to say that the value of $B^2\rho^2$ is smaller than 10,000.

On the other hand, a typical size of a dictionary is much larger than 10,000. For example, there are more than 100,000 distinct words in English. We have therefore shown a problem in which there can be an order of magnitude difference between learning a halfspace with the SVM rule and learning a halfspace using the vanilla ERM rule.