CSC 580 Principles of Machine Learning

07 Linear models for classification

Chicheng Zhang

Department of Computer Science

THE UNIVERSITY OF ARIZONA

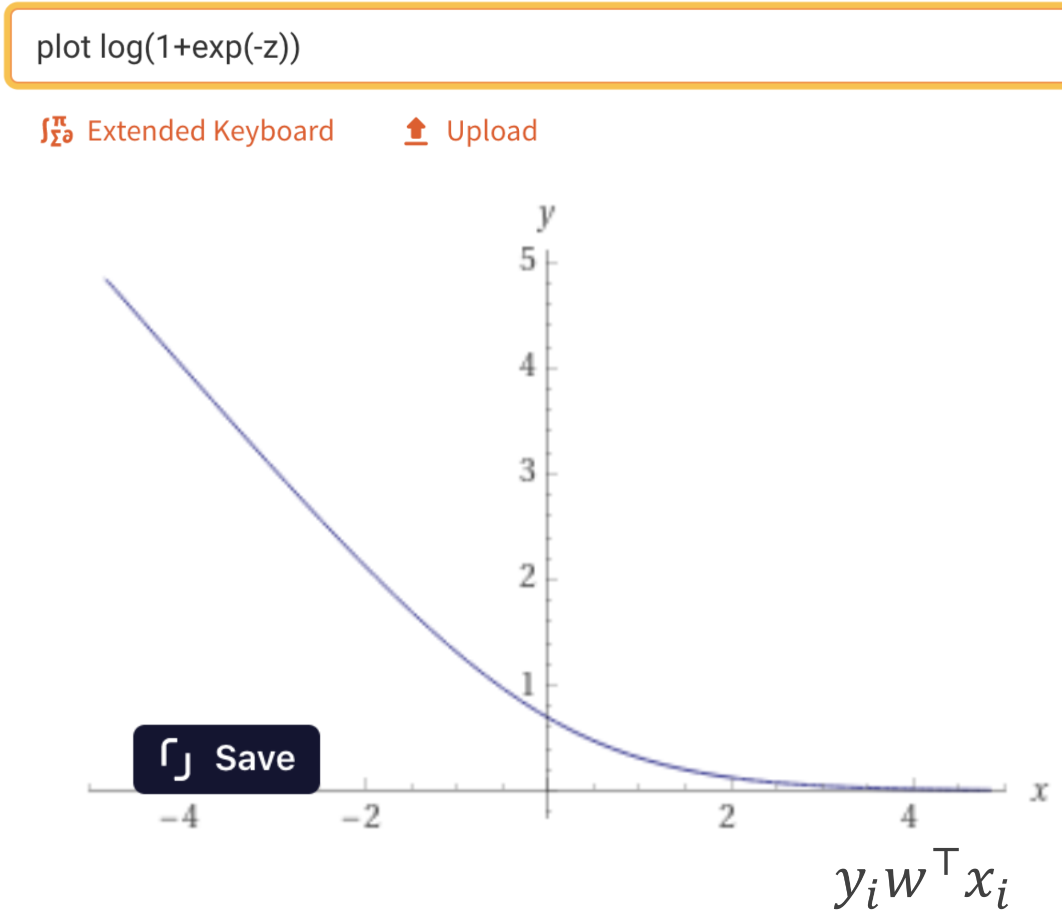
*slides credit: built upon CSC 580 Fall 2021 lecture slides by Kwang-Sung Jun

Classification with linear models

- Logistic loss
  - $x_i \in \mathbb{R}^d, \;\; y_i \in \{1, -1\}$
  - $S = \{(x_i, y_i)\}_{i=1}^n$
  - $\ell(w; x_i, y_i) = \log(1 + \exp(-y_i \cdot w^\top x_i))$

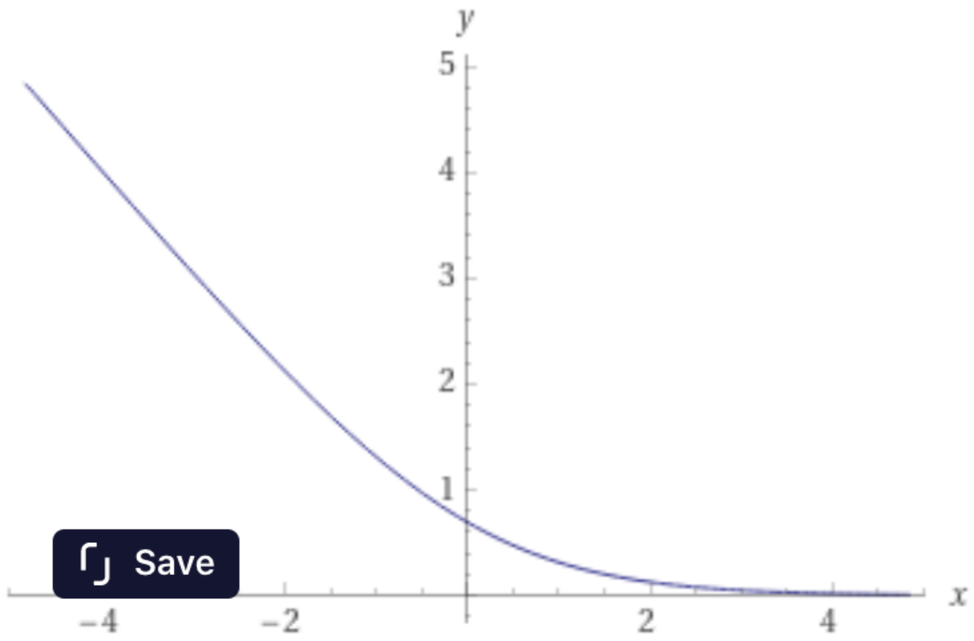
- The ERM principle, again!
  $\hat{w} = \mathrm{argmin}_{w \in \mathbb{R}^d} F(w),\;\; F(w) \coloneqq \sum_{i=1}^n \ell(w; x_i, y_i)$

- How to optimize?



First, is it convex?

- How do we check the convexity of F?
  - Is $\ell(w; x_i, y_i) = \log(1 + \exp(-y_i \cdot w^\top x_i))$ convex in w?
  - Observation: $\ell(w; x_i, y_i) = h(y_i \cdot w^\top x_i)$ where $h(z) = \log(1 + \exp(-z))$
  - It suffices to check that $h(z)$ is convex
  - Indeed, $h''(z) = \frac{e^{-z}}{(1+e^{-z})^2} \geq 0$



- Alternative route: check the PSD-ness of $\nabla^2 \ell(w; x_i, y_i)$

- Great! Let's solve $\nabla F(w) = 0$

Finding the minimizer of F: gradient descent

- Algorithm

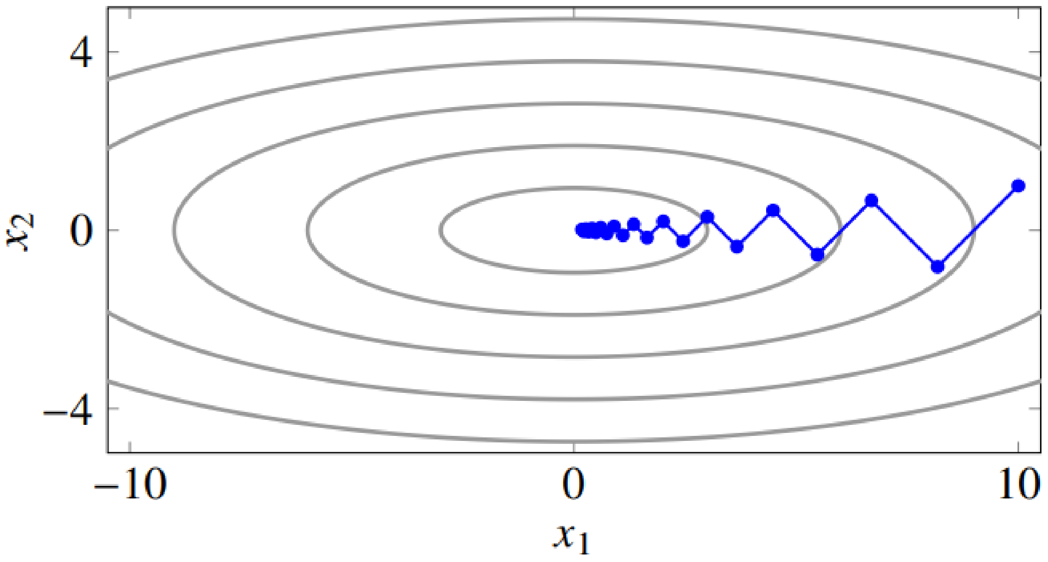
  Input: initial point $w_0 \in \mathbb{R}^d$

  step sizes $\{\eta_t\}_{t=1}^{\infty}$

  stopping tolerence $\epsilon > 0$

  For $t = 1, \dots,$ max_iter

  - $w_t \leftarrow w_{t-1} - \eta_t \cdot \nabla F(w_{t-1})$
  - stop if $\left|\frac{F(w_t) - F(w_{t-1})}{F(w_{t-1})}\right| \leq \epsilon$



Hyperparameters

- w_0: set it to 0
  - warmstart possible if you have a good guess
- stepsize
  - constant scheme: $\eta_t = \eta, \forall t$
  - $\eta_t = \frac{1}{\sqrt{t}}$
  - $\eta_t = \frac{1}{t}$
  - Line search possible
- ϵ: 10^{-4} to 10^{-7}… more of engineering.

http://www.seas.ucla.edu/~vandenbe/236C/lectures/gradient.pdf

More iterative methods

Algorithms	Number if iterations until convergence	Time complexity per iteration
Newton's method	Very small	nd^3
LBFGS	small	nmd
Gradient descent (GD)	large	nd
Stochastic gradient descent (SGD)	Very large	d

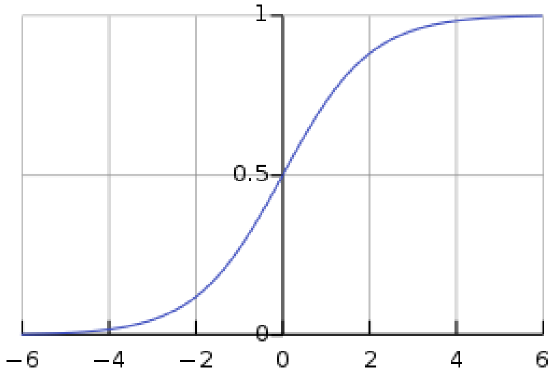
- n: #training examples
- d: dimensionality
- m: LBFGS's memory hyperparameter
- Will come back to SGD in later part of this lecture

Probabilistic interpretation of logistic regression

- How did they come up with the logistic loss?
- Let us begin using 1/0 encoding for the label (then later turn into 1/-1 encoding)

- $y_i \mid x_i \sim \mathrm{Bernoulli}(p_i)$, where $p_i = g(x_i)$

- Modeling attempt 1: $g(x_i) = w^\top x_i$
- Modeling attempt 2: $g(x_i) = \sigma(w^\top x_i)$, where $\sigma(z) = \frac{1}{1+e^{-z}}$ is the sigmoid function
  - i.e. logit $\log\left(\frac{p_i}{1-p_i}\right) = w^\top x_i$



Probabilistic interpretation of logistic regression

Logistic regression as maximum likelihood estimation $\quad y_i \mid x_i \sim \mathrm{Bernoulli}(\sigma(w^\top x_i))$

→ now, just like in regression, maximize the "likelihood"

$$\hat{w} = \arg\max_w \prod_{i=1}^n P(Y_i = y_i, X_i = x_i)$$

$$= \;'' \; \prod P(Y_i = y_i \mid X_i = x_i)\, P(X_i = x_i)$$

$$= \;'' \; \prod_{i=1}^n \mathbb{P}(Y_i = y_i \,;\, p_i = \sigma(w^\top x_i))$$

$$= \arg\max_w \sum_{i=1}^n \log P(\quad '' \quad)$$

$$= \arg\min_w \sum_{i=1}^n -\log P(\quad '' \quad)$$

$$\text{if } \begin{cases} y_i = 1, & -\log p_i \\ y_i = 0, & -\log(1-p_i) \end{cases}$$

$$= \begin{cases} (y_i = 1) \Rightarrow (-1)\cdot 1\cdot(-\log(1+e^{-w^\top x_i})) \\ (y_i = 0) \Rightarrow (-1)\cdot 1\cdot(-\log(1+e^{w^\top x_i})) \end{cases}$$

Let $\begin{cases} \tilde{y}_i = 1 & \text{if } y_i = 1 \\ \tilde{y}_i = -1 & \text{if } y_i = 0 \end{cases}$

$$\Rightarrow \; = \log(1 + e^{-\tilde{y}_i w^\top x_i})$$

$\in \{0,1\}$

Prb of observing $Y_i = y_i$

$$p_i^{y_i}\cdot(1-p_i)^{1-y_i}$$

$$\begin{cases} \text{if } y_i = 1: & p_i \\ \text{if } y_i = 0: & 1 - p_i \end{cases}$$

⊙ $\log p_i$

$$= \log \sigma(w^\top x_i)$$

$$= \log\left(\frac{1}{1+e^{-w^\top x_i}}\right)$$

$$= -\log(1+e^{-w^\top x_i})$$

⊙ $\log(1-p_i)$

$$= \log\left(1 - \frac{1}{1+e^{-w^\top x_i}}\right)$$

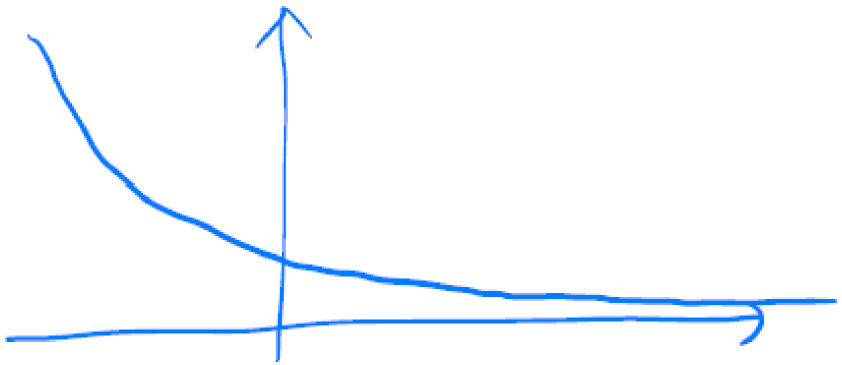
$$= \log\left(\frac{e^{-w^\top x_i}}{1+e^{-w^\top x_i}}\right)$$

$$= \log\left(\frac{1}{1+e^{w^\top x_i}}\right)$$

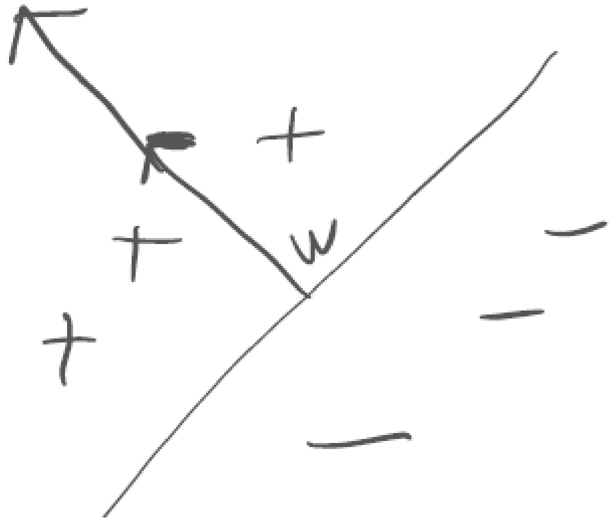
$$= -\log(1+e^{w^\top x_i})$$

Caveat: Logistic regression may not have a minimizer without a regularizer

- E.g.,
  - training set has only one data point



  - more generally, linearly separable data.



  - Structure of minimizers, optimization properties discussed in

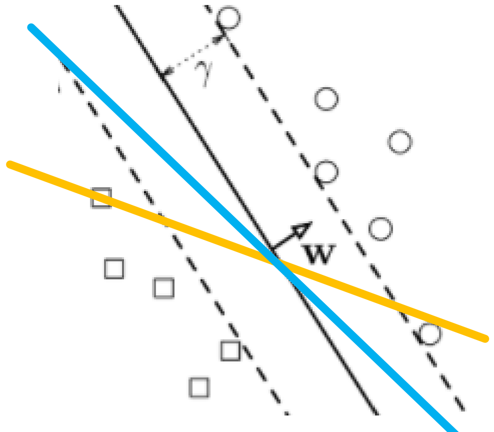
Convex Analysis at Infinity: An Introduction to Astral Space

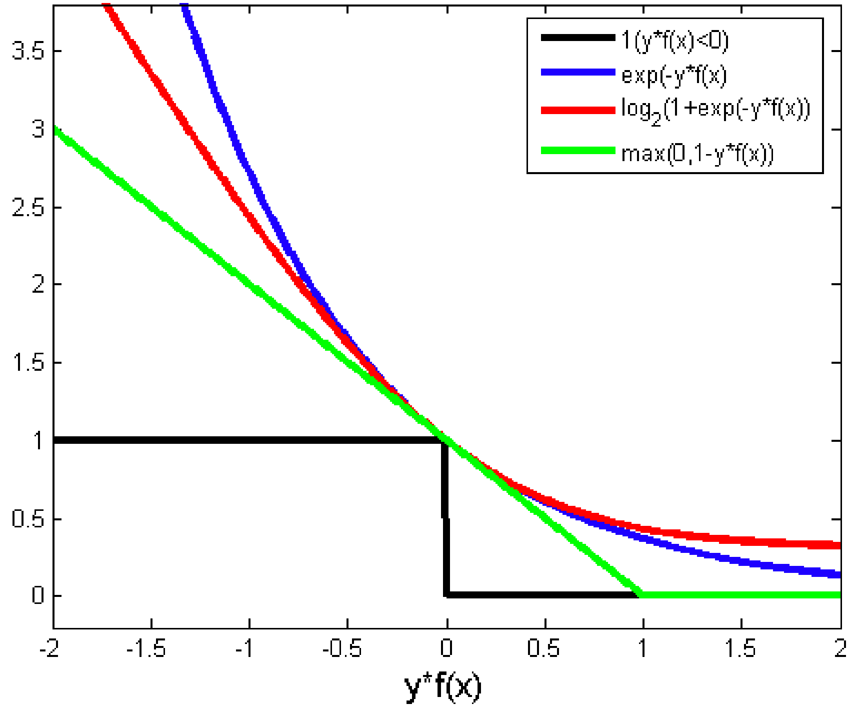
Miroslav Dudík, Ziwei Ji, Robert E. Schapire, Matus Telgarsky

  - Adding regularization addresses this issue:
    $\hat{w} = \operatorname{argmin}_{w\in\mathbb{R}^d} \sum_{i=1}^{n} \ell(w; x_i, y_i) + \lambda \|w\|_2^2$

Support Vector Machines

- In a nutshell
  - Perform regularized ERM $\hat{w} = \mathrm{argmin}_{w \in \mathbb{R}^d} \sum_{i=1}^{n} \ell(w; x_i, y_i) + \lambda \|w\|_2^2$
    with the loss
    $$\ell(w; x, y) = (1 - y \cdot w^\top x)_+$$ hinge loss
    - notation: $(z)_+ \coloneqq \max\{0, z\}$
- Interesting aspects
  - Works well in general
  - No corresponding probabilistic motivation
  - Geometric Interpretation: **maximize the margin**.





https://rohanvarma.me/Loss-Functions/

Remaining parts of the lecture

- Q1: How is the loss function motivated and how is it maximizing the margin?

- Q2: How to solve the SVM optimization problem efficiently?

Next class (9/28)

- Answering
  - Q1 (geometric perspective; margin maximization)
  - Q2 (induced practical optimization algorithms; Dual of SVM)

- Kernel methods

- Plan to release HW2

- Assigned reading: CIML 11.1-11.2

SVM: motivation

- The goal of linear classifier: Find w so that the rule $h_w(x) = \mathrm{sign}(w^\top x)$ will have small generalization error $\mathrm{err}(h_w)$.

- ERM: it seems natural to use the loss $1\{h_w(x) \neq y\}$, but…
  - NP-hard (e.g. Guruswami and Raghavendra, 2009)
  - There might be multiple minima. How to break ties?

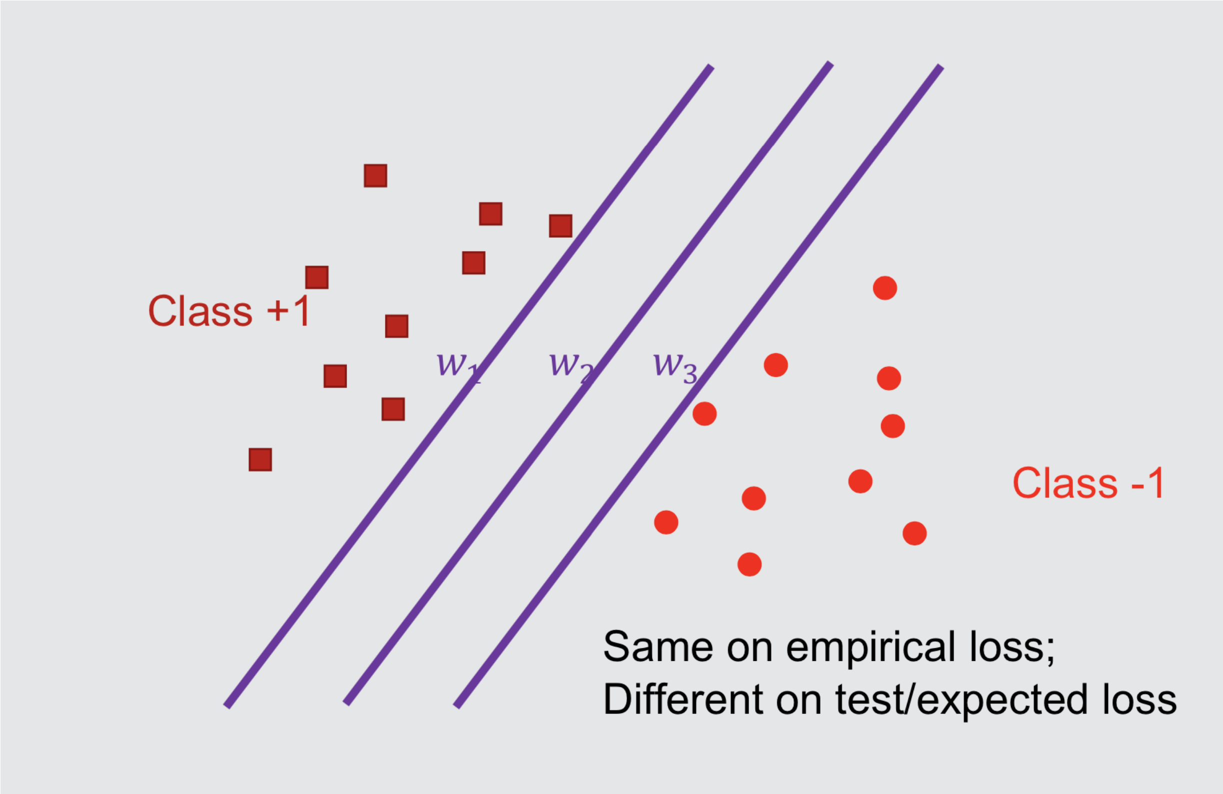
- Okay, we're stuck. Let us consider a **simple problem** and then try to extend it to the generic problem.
- The simple case: **linearly separable data** (recall perceptron)

https://www.cs.cornell.edu/courses/cs6781/2020sp/lectures/10_hardness2.pdf

Linearly separable data

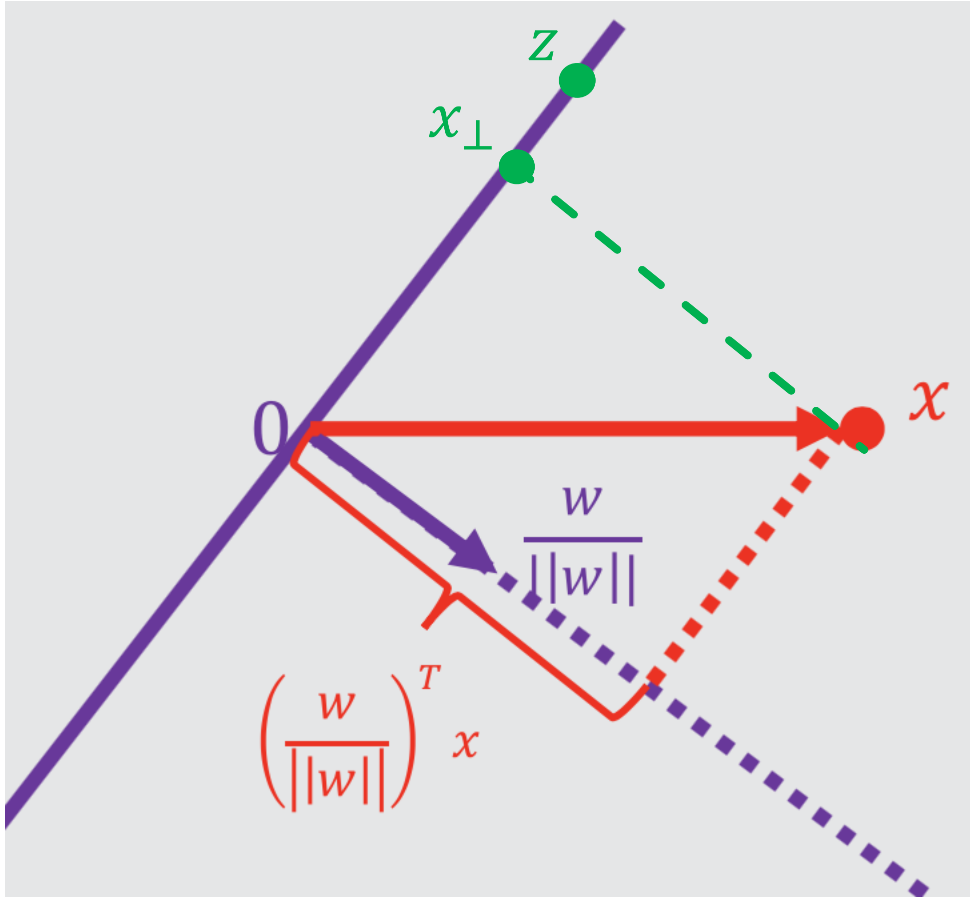
- Recall: we can minimize 0-1 loss here with a reasonable time complexity!
  - e.g., run perceptron until it classifies train set perfectly
- But, among these minimizers, which one should we pick?

- Idea: pick the hyperplane such that its distances to all training examples are far



Facts on vectors

- (Lem 1) a vector x has distance $\frac{w^\top x}{\|w\|}$ to the hyperplane $w^\top x = 0$



- How about with bias? $w^\top x + b = 0$
- Let us be explicit on the bias: $f(x; w, b) = w^\top x + b$

- recall: w is orthogonal to the hyperplane $w^\top x + b = 0$
  - why? (left as exercise)

Facts on vectors

- (Lem 2) x has distance $\frac{|w^\top x+b|}{\|w\|}$ to the hyperplane $w^\top x + b = 0$

claim1 : x can be written as $x = x_\perp + r\frac{w}{\|w\|}$ where $x_\perp$ is the projection of x onto the hyperplane.

claim2 : then, $|r|$ is the distance between x and the hyperplane

Solving for r: $w^\top x + b = w^\top x_\perp + r\frac{w^\top w}{\|w\|} + b = r\|w\|$.

this implies $|r| = \frac{|w^\top x+b|}{\|w\|}$

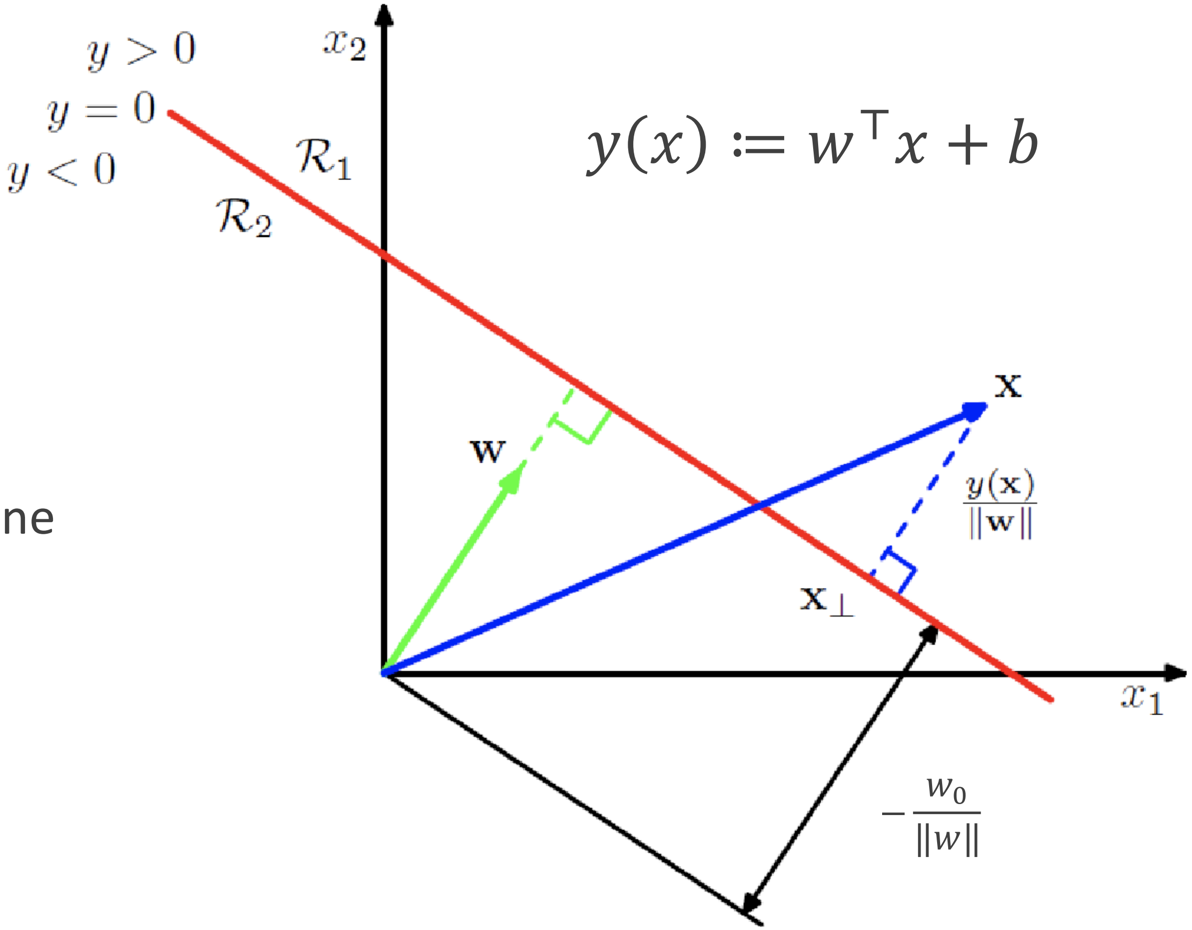
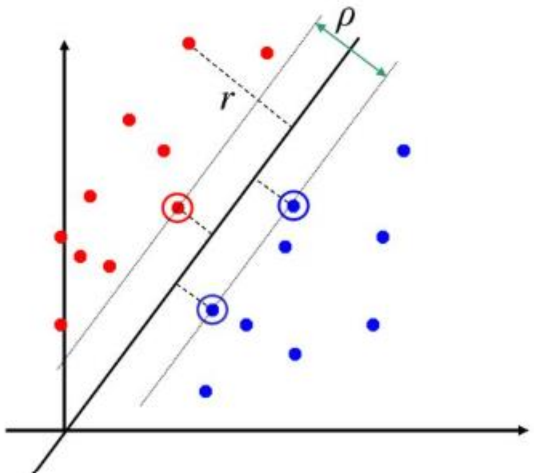


Figure from *Pattern Recognition and Machine Learning*, Bishop

SVM derivation (1)



- Margin of (w, b) over all training points: $\gamma'(w, b) = \min_i \frac{|w^\top x_i + b|}{\|w\|}$

- Choose (w, b) with the maximum margin? .. wait, we also want it to be a perfect classifier
  - redefine it

$$\gamma(w, b) = \min_i \frac{y_i(w^\top x_i + b)}{\|w\|}$$

- Choose w with the maximum margin (and perfect classification)

$$\left(\hat{w}, \hat{b}\right) = \max_{w,b} \min_{i=1}^{n} \frac{y_i(w^\top x_i + b)}{\|w\|}$$

  - One more issue: still, infinitely many solutions..!

SVM derivation (2)

$$(\hat{w}, \hat{b}) = \max_{w,b} \min_{i=1}^{n} \frac{y_i(w^\top x_i + b)}{\|w\|}$$

- Infinitely many solutions..
- It's actually a matter of removing 'duplicates'; ∃ many (w,b)'s that actually represent the same hyperplane.

- Quick solution = achieves the smallest margin
  - For any solution $(\hat{w}, \hat{b})$, let x_{i^*} be the **closest to the hyperplane** $\hat{w} x_i + \hat{b} = 0$
  - Imagine rescaling $(\hat{w}, \hat{b})$ so that $\left|\hat{w}^\top x_{i^*} + \hat{b}\right| = 1$
- We can always do that, but can we find a formulation that automatically finds that modified solution?
  - add the constraint $\min_i y_i(w^\top x_i + b) = 1$

SVM derivation (3)

$$\max_{w,b} \min_{i=1}^{n} \frac{y_i(w^\top x_i + b)}{\|w\|}$$
$$s.t.\ \min_{\mathrm{i}} y_i(w^\top x_i + b) = 1$$

- Summary: the constraint encodes (1) correct classification (2) there are no two solutions that represent the same hyperplane!
  - Note: If $(\hat{w}, \hat{b})$ is a solution, then the margin is $\frac{1}{\|\hat{w}\|}$

$$\max_{w,b} \frac{1}{\|w\|} \quad s.t.\ \min_{\mathrm{i}} y_i(w^\top x_i + b) = 1$$

$$\max_{w,b} \frac{1}{\|w\|} \quad s.t.\ \min_{i} y_i(w^\top x_i + b) \geq 1$$

(turns out to be equivalent..)

$$\max_{w,b} \frac{1}{\|w\|} \quad s.t.\ y_i(w^\top x_i + b) \geq 1, \forall i$$

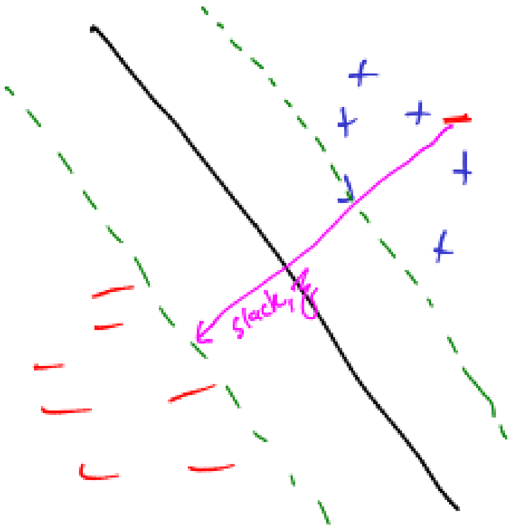
Final formulation in the linearly separable setting:
(quadratic programming)

$$\min_{w,b} \|w\|^2$$
$$\mathrm{s.t.}\ y_i(w^\top x_i + b) \geq 1, \forall i$$

SVM formulation: the nonseparable setting

$$\min_{w,b} \|w\|^2$$
$$s.t.\ \ y_i(w^\top x_i + b) \geq 1, \forall i$$

- What if data is linearly nonseparable?
- Introduce 'slack' variables



$$\min_{w,b,\{\xi_i \geq 0\}} \|w\|^2 + C\sum_{i=1}^{n} \xi_i \qquad // \ C \text{ is a hyper-parameter}$$
$$s.t.\ \ y_i(w^\top x_i + b) \geq 1 - \xi_i, \forall i$$

- Again, a quadratic programming problem
- Fix any w, b, the optimal ξ?

$\xi_i = 0$ if $y_i(w^\top x_i + b) \geq 1$, and $\xi_i = 1 - y_i(w^\top x_i + b)$

$$\min_{w,b}\ \|w\|^2 + C\sum_{i=1}^{n} \left(1 - y_i(w^\top x_i + b)\right)_+ \Leftrightarrow \text{Regularized hinge loss minimization } \lambda = \frac{1}{C}$$

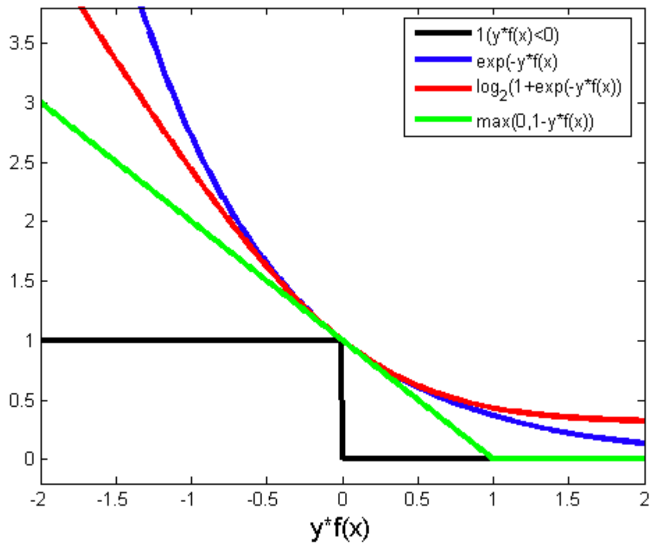
Solving SVM optimization problem

- Two popular methods

- Method 1: stochastic gradient descent

- Method 2: solve the *dual problem* and transform the dual solution back

Stochastic gradient descent (SGD)

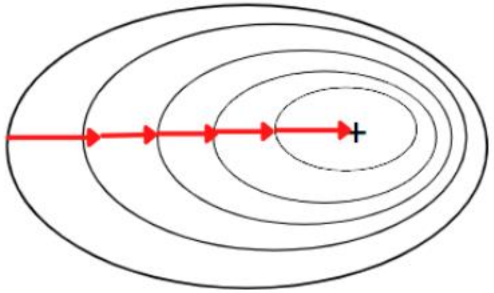


- Finding $\hat{w} = \mathrm{argmin}_{w \in \mathbb{R}^d} F(w),\ F(w) = \frac{1}{n}\sum_{i=1}^{n} f_i(w)$,
  where $f_i(w)$ is convex + quadratic, e.g.
  $(1 - y_i\langle w, x_i\rangle)_+ + \lambda\|w\|_2^2$,
  $\log(1 + \exp(-y_i \cdot w^\top x_i)) + \lambda\|w\|_2^2$

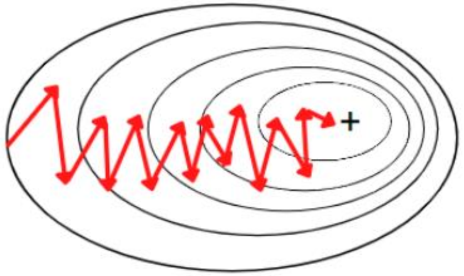
- Observation: gradient descent is computationally expensive
  - calculating exact gradient $\nabla F(w)$ takes at least $\Omega(n)$ time

- Key idea (Robbins-Monro'51): descend in directions that are in-expectation $\nabla F(w)$
- For $t = 1, 2, \ldots, T$:
  - Choose $i_t \sim \mathrm{Uniform}(\{1, \ldots, n\})$
  - $w_{t+1} \leftarrow w_t - \eta_t \nabla f_{i_t}(w)$
- Output: (1) $\bar{w}_T := \frac{1}{T}\sum_{t=1}^{T} w_t$ (average iterate); (2) w_T (last iterate)

Batch Gradient Descent



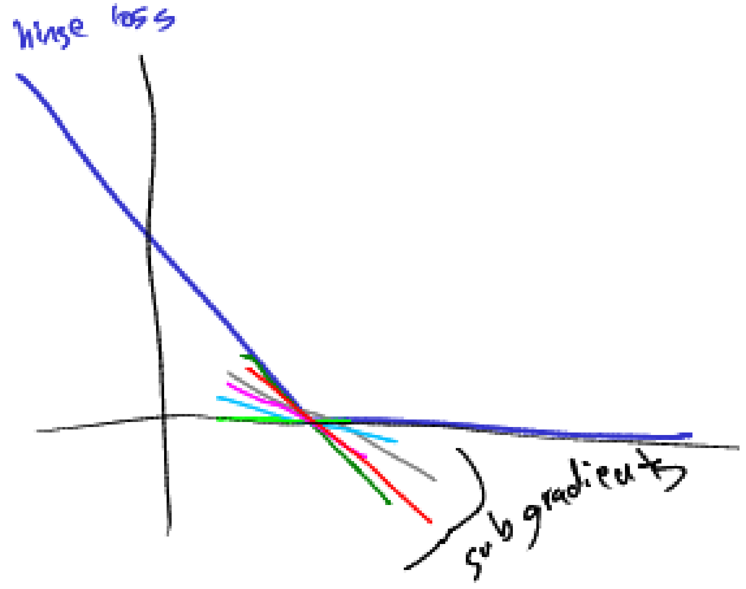
Stochastic Gradient Descent



SGD: handling nondifferentiable objectives

- Hinge loss:

  $f(w) = h(w) + \frac{\lambda}{2}\|w\|_2^2$, where $h(w) = (1 - y\langle w, x\rangle)_+$



- For some w, $\nabla h(w)$ does not exist (say, d=1)

- Workaround: descent in the *subgradient* direction

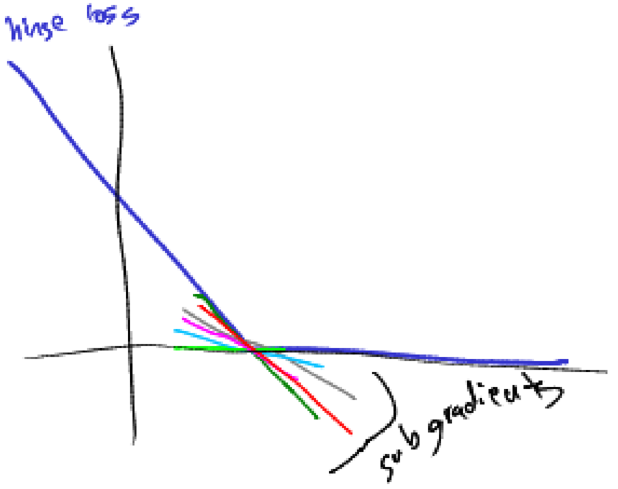
- [Def] For convex function h, $g \in \mathbb{R}^d$ is said to be a subgradient of h at w, if for any u,

$$h(u) \geq h(w) + \langle g, u - w\rangle$$

  The set of subgradients of h at w is denoted as $\partial h(w)$

- For differentiable h, $\partial h(w) = \{\nabla h(w)\}$

Subgradient: intuition and properties



- Example: $h(w) \;=\; (1-w)_+$,

$$\partial h(w) = \begin{cases} \{-1\}, & w < 1 \\ [-1,0], & w = 1 \\ \{0\}, & w > 1 \end{cases}$$

- (Lem) If $h(w) = l(\langle a, w\rangle + b)$ for some convex l on $\mathbb{R}$, and suppose $z \in \partial l(\langle a, w\rangle + b)$. Then, $az \in \partial h(w)$
    - Generalizes chain rule of differentiation

- Practical implication: For $f(w) = (1 - y\langle w, x\rangle)_+$, the following vector(s) are in $\partial f(w)$ (and are thus valid descent directions):

$$\begin{cases} -yx, & y\langle w, x\rangle < 1 \\ -uyx \text{ for } u \in [0,1], & y\langle w, x\rangle = 1 \\ 0, & y\langle w, x\rangle > 1 \end{cases}$$

SGD: convergence guarantee

- (Thm) Suppose $F(w) = \frac{1}{n}\sum_{i=1}^{n} f_i(w)$, where $f_i(w) = h_i(w) + \lambda\|w\|_2^2$, and $h_i(w)$ is L-Lipschitz, then SGD with step size $\eta_t = \frac{1}{\lambda t}$ satisfies that

$$\mathbb{E}[F(\bar{w}_T)] - \min_{w} F(w) \leq O\left(\frac{L^2 \log T}{\lambda T}\right),$$

where $\bar{w}_T = \frac{1}{T}\sum_{t=1}^{T} w_t$

- [Def] h is said to be L-Lipschitz, if for any u, v, $|h(u) - h(v)| \leq L\|u - v\|_2$

- $\tilde{O}\left(\frac{1}{T}\right)$ rate; if target optimization precision ϵ, then $O\left(\frac{1}{T}\right) \leq \epsilon \Leftarrow T \geq O\left(\frac{1}{\epsilon}\right)$
- Larger λ, “Smoother” $h_i \Longrightarrow$ easier to optimize