Statistical Machine Learning

https://cvml.ist.ac.at/courses/SML_W18

Christoph Lampert

IST AUSTRIA

Institute of Science and Technology

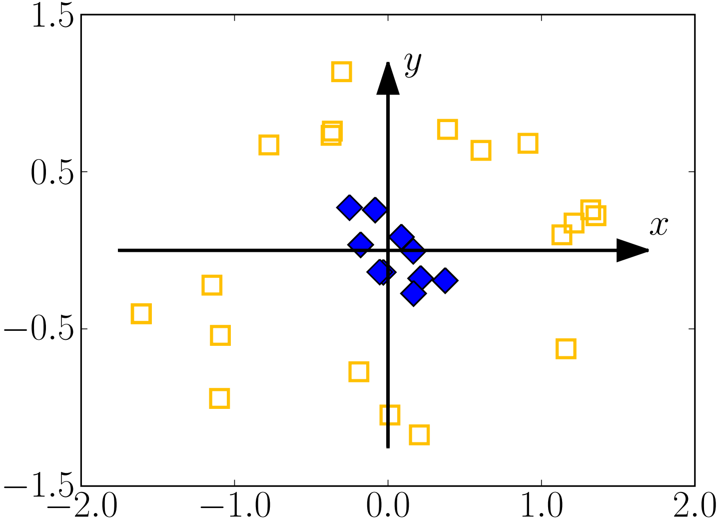
Spring Semester 2018/2019

Lecture 4

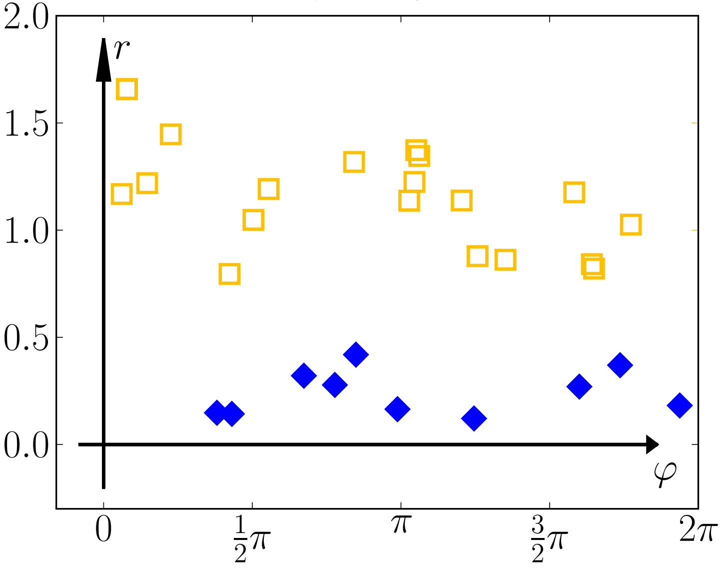
Overview (tentative)

Date		no.	Topic
Oct 08	Mon	1	A Hands-On Introduction
Oct 10	Wed	–	self-study (Christoph traveling)
Oct 15	Mon	2	Bayesian Decision Theory
			Generative Probabilistic Models
Oct 17	Wed	3	Discriminative Probabilistic Models
			Maximum Margin Classifiers
Oct 22	Mon	4	Generalized Linear Classifiers, Optimization
Oct 24	Wed	5	Evaluating Predictors; Model Selection
Oct 29	Mon	–	self-study (Christoph traveling)
Oct 31	Wed	6	Overfitting/Underfitting, Regularization
Nov 05	Mon	7	Learning Theory I: classical/Rademacher bounds
Nov 07	Wed	8	Learning Theory II: miscellaneous
Nov 12	Mon	9	Probabilistic Graphical Models I
Nov 14	Wed	10	Probabilistic Graphical Models II
Nov 19	Mon	11	Probabilistic Graphical Models III
Nov 21	Wed	12	Probabilistic Graphical Models IV
until Nov 25			final project

What, if a linear classifier is really not a good choice?



What, if a linear classifier is really not a good choice?



Change the data representation, e.g. Cartesian $\rightarrow$ polar coordinates

Definition (Max-margin Generalized Linear Classifier)

Let $C > 0$. Assume a necessarily linearly separable training set

$$\mathcal{D} = \{(x^1, y^1), \dots (x^n, y^n)\} \subset \mathcal{X} \times \mathcal{Y}.$$

Let $\phi : \mathcal{X} \to \mathbb{R}^D$ be a feature map from $\mathcal{X}$ into a feature space $\mathbb{R}^D$.

Then we can form a new training set

$$\mathcal{D}^\phi = \{\ (\phi(x^1), y^1),\ \dots,\ (\phi(x^n), y^n)\ \} \subset \mathbb{R}^D \times \mathcal{Y}.$$

The maximum-(soft)-margin linear classifier in $\mathbb{R}^D$,

$$g(x) = \operatorname{sign}[\langle w, \phi(x) \rangle_{\mathbb{R}^D} + b]$$

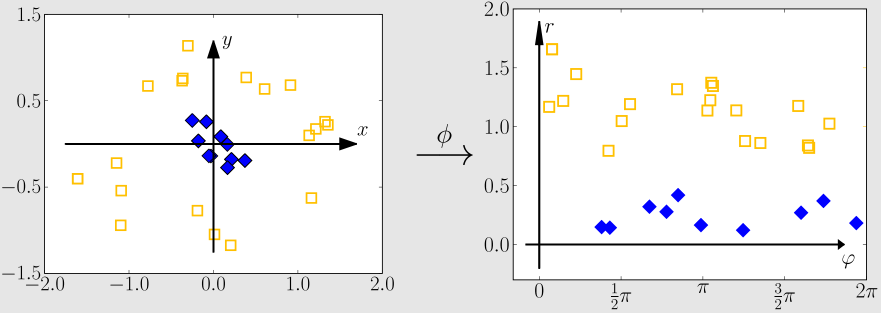
for $w \in \mathbb{R}^D$ and $b \in \mathbb{R}$ is called **max-margin generalized linear classifier**.

It is still *linear* w.r.t w, but (in general) nonlinear with respect to x.

Example (Polar coordinates)

Left: dataset $\mathcal{D}$ for which no good linear classifier exists.
Right: dataset $\mathcal{D}^\phi$ for $\phi : \mathcal{X} \to \mathbb{R}^D$ with $\mathcal{X} = \mathbb{R}^2$ and $\mathbb{R}^D = \mathbb{R}^2$

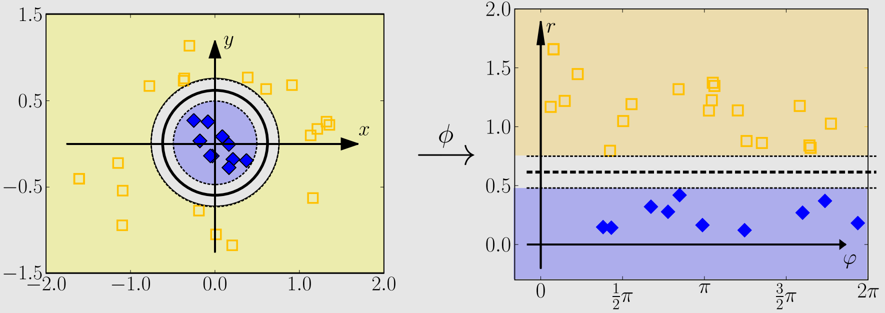
$$\phi(x, y) = (\sqrt{x^2 + y^2}, \arctan \frac{y}{x}) \qquad \text{(and } \phi(0,0) = (0,0)\text{)}$$



Example (Polar coordinates)

Left: dataset $\mathcal{D}$ for which no good linear classifier exists.
Right: dataset $\mathcal{D}^\phi$ for $\phi : \mathcal{X} \to \mathbb{R}^D$ with $\mathcal{X} = \mathbb{R}^2$ and $\mathbb{R}^D = \mathbb{R}^2$

$$\phi(x, y) = \left(\sqrt{x^2 + y^2}, \arctan \frac{y}{x}\right) \qquad \text{(and } \phi(0,0) = (0,0)\text{)}$$



Any classifier in $\mathbb{R}^D$ induces a classifier in $\mathcal{X}$.

Other popular feature mappings, ϕ

Example (d-th degree polynomials)

$$\phi : \Big(x_1, \dots, x_n\Big) \mapsto \Big(1, x_1, \dots, x_n, x_1^2, \dots, x_n^2, x_1^2, x_1 x_2, \dots, x_n^2, \dots, x_n^d\Big)$$

Resulting classifier: d-th degree polynomial in x. $g(x) = \operatorname{sign} f(x)$ with

$$f(x) = \langle w, \phi(x) \rangle = \sum\nolimits_j w_j \phi(x)_j = \sum\nolimits_i a_i x_i + \sum\nolimits_{ij} b_{ij} x_i x_j + \dots$$

Example (Distance map)

For a set of prototype $p_1, \dots, p_N \in \mathcal{X}$:

$$\phi : \vec{x} \mapsto \Big(e^{-\|\vec{x} - \vec{p}_1\|^2}, \dots, e^{-\|\vec{x} - \vec{p}_N\|^2}\Big)$$

Classifier: combine weights from close enough prototypes

$$g(x) = \operatorname{sign}\langle w, \phi(x) \rangle = \operatorname{sign} \sum\nolimits_{i=1}^n a_i e^{-\|\vec{x} - \vec{p}_i\|^2}.$$

Other popular feature mappings, ϕ

Example (Pre-trained deep network)

Imagine somebody trained a (deep) neural network on a large dataset, e.g. ImageNet for image classification.

Idea: use initial segment of network as feature extractor for other data:

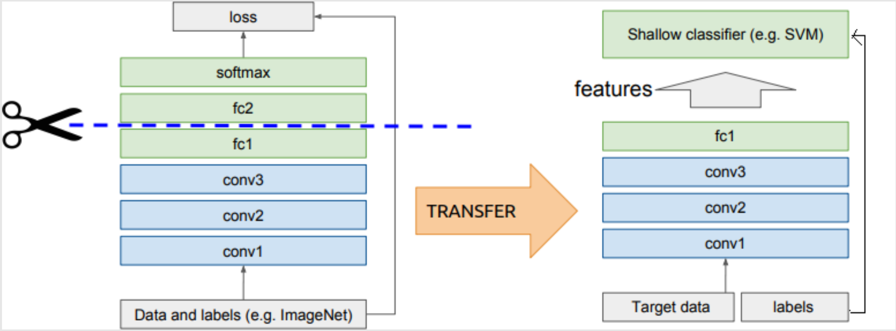


Image: Steven Schmatz, https://www.quora.com/What-is-the-difference-between-transfer-learning-domain-adaptation-and-multitask-learning-in-machine-learning

(Generalized) Maximum Margin Classifiers – Optimization II

$$\min_{w\in\mathbb{R}^D, b\in\mathbb{R}, \xi\in\mathbb{R}^n} \quad \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{n}\xi^i$$

subject to

$$y^i(\langle w, \phi(x^i)\rangle + b) \geq 1 - \xi^i, \quad \text{for } i = 1, \dots, n,$$
$$\xi^i \geq 0. \quad \text{for } i = 1, \dots, n.$$

How to solve numerically?

- off-the-shelf Quadratic Program (QP) solver
  only for small dimensions and training sets (a few hundred),
- variants of gradient descent,
  high dimensional data, large training sets (millions)
- by convex duality,
  for very high dimensional data and not so many examples ($d \gg n$)

(Generalized) Maximum Margin Classifiers – Optimization II

For simplifity of notation, switch back to linear classifier:

$$\min_{w\in\mathbb{R}^d, b\in\mathbb{R}, \xi\in\mathbb{R}^n} \quad \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{n} \xi^i$$

subject to

$$y^i(\langle w, x^i\rangle + b) \geq 1 - \xi^i, \quad \text{for } i = 1, \ldots, n,$$
$$\xi^i \geq 0. \quad \text{for } i = 1, \ldots, n.$$

How to solve numerically?

- off-the-shelf Quadratic Program (QP) solver
  only for small dimensions and training sets (a few hundred),
- variants of gradient descent,
  high dimensional data, large training sets (millions)
- by convex duality,
  for very high dimensional data and not so many examples ($d \gg n$)

Subgradient-Based Optimization

$$\min_{w \in \mathbb{R}^d, b \in \mathbb{R}, \xi \in \mathbb{R}^n} \quad \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{n} \xi^i$$

subject to

$$y^i(\langle w, x^i \rangle + b) \geq 1 - \xi^i, \quad \text{and} \quad \xi^i \geq 0, \qquad \text{for } i = 1, \ldots, n.$$

$$\min_{w\in\mathbb{R}^d, b\in\mathbb{R}, \xi\in\mathbb{R}^n} \quad \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{n}\xi^i$$

subject to

$$y^i(\langle w, x^i\rangle + b) \geq 1 - \xi^i, \quad \text{and} \quad \xi^i \geq 0, \qquad \text{for } i = 1, \ldots, n.$$

For any fixed (w, b) we can find the optimal $\xi^1, \ldots, \xi^n$:

$$\xi^i = \mathbf{max}\{\ 0, 1 - y_i(\langle w, x_i\rangle + b)\ \}.$$

$$\min_{w\in\mathbb{R}^d, b\in\mathbb{R}, \xi\in\mathbb{R}^n} \quad \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{n}\xi^i$$

subject to

$$y^i(\langle w, x^i\rangle + b) \geq 1 - \xi^i, \quad \text{and} \quad \xi^i \geq 0, \qquad \text{for } i = 1, \dots, n.$$

For any fixed (w, b) we can find the optimal $\xi^1, \dots, \xi^n$:

$$\xi^i = \mathbf{max}\{\ 0, 1 - y_i(\langle w, x_i\rangle + b)\ \}.$$

Plug into original problem:

$$\min_{w\in\mathbb{R}^d, b\in\mathbb{R}} \quad \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{n}\underbrace{\mathbf{max}\{\ 0, 1 - y_i(\langle w, x_i\rangle + b)\}}_{\text{"Hinge loss"}}.$$

SVM Training in the Primal

$$\min_{w\in\mathbb{R}^d, b\in\mathbb{R}} \quad \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{n} \mathbf{max}\{\, 0, 1 - y_i(\langle w, x_i\rangle + b)\}.$$

- unconstrained optimization problem
- convex
  - $\frac{1}{2}\|w\|^2$ is convex (differentiable with Hessian = Id $\succcurlyeq 0$)
  - linear/affine functions are convex
  - pointwise $\mathbf{max}$ over convex functions is convex.
  - sum of convex functions is convex.
- *not differentiable!*

SVM Training in the Primal

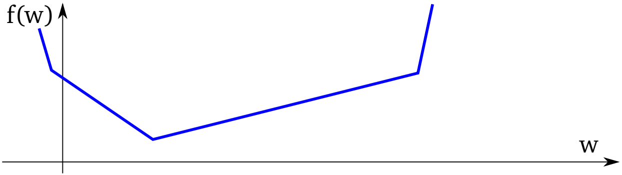
$$\min_{w \in \mathbb{R}^d, b \in \mathbb{R}} \quad \frac{1}{2}\|w\|^2 + C \sum_{i=1}^{n} \mathbf{max}\{\, 0, 1 - y_i(\langle w, x_i \rangle + b)\}.$$

- unconstrained optimization problem
- convex
  - $\frac{1}{2}\|w\|^2$ is convex (differentiable with Hessian $=$ Id $\succcurlyeq 0$)
  - linear/affine functions are convex
  - pointwise $\mathbf{max}$ over convex functions is convex.
  - sum of convex functions is convex.
- *not differentiable!*

$\boxed{\text{We can't use gradient descent, since some points have no gradients!}}$

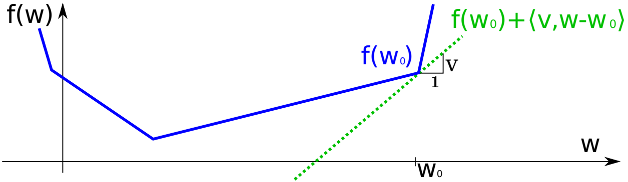
Definition: Let $f : \mathbb{R}^d \to \mathbb{R}$ be a **convex** function. A vector $v \in \mathbb{R}^d$ is called a **subgradient** of f at w_0, if

$$f(w) \geq f(w_0) + \langle v, w - w_0 \rangle \quad \text{for all } w.$$



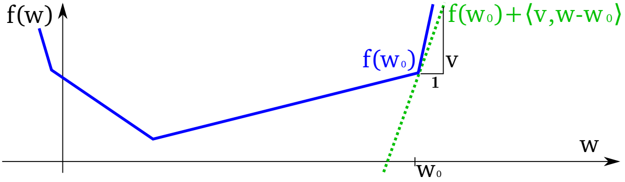
Definition: Let $f : \mathbb{R}^d \to \mathbb{R}$ be a **convex** function. A vector $v \in \mathbb{R}^d$ is called a **subgradient** of f at w_0, if

$$f(w) \geq f(w_0) + \langle v, w - w_0 \rangle \quad \text{for all } w.$$



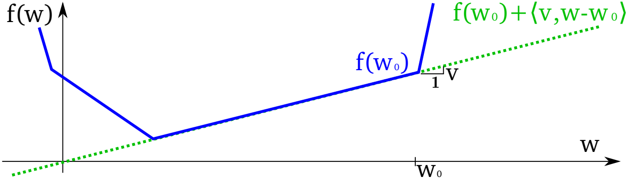
Definition: Let $f : \mathbb{R}^d \to \mathbb{R}$ be a **convex** function. A vector $v \in \mathbb{R}^d$ is called a **subgradient** of f at w_0, if

$$f(w) \geq f(w_0) + \langle v, w - w_0 \rangle \quad \text{for all } w.$$



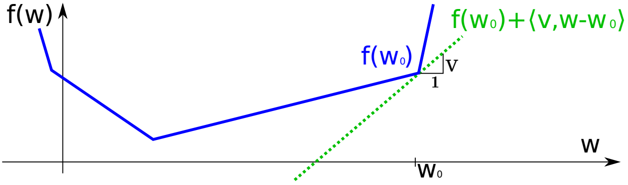
Definition: Let $f : \mathbb{R}^d \to \mathbb{R}$ be a **convex** function. A vector $v \in \mathbb{R}^d$ is called a **subgradient** of f at w_0, if

$$f(w) \geq f(w_0) + \langle v, w - w_0 \rangle \quad \text{for all } w.$$



Definition: Let $f : \mathbb{R}^d \to \mathbb{R}$ be a **convex** function. A vector $v \in \mathbb{R}^d$ is called a **subgradient** of f at w_0, if

$$f(w) \geq f(w_0) + \langle v, w - w_0 \rangle \quad \text{for all } w.$$

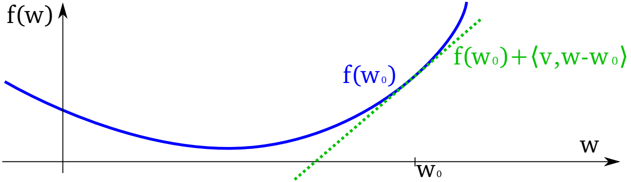


A general convex f can have more than one subgradient at a position.

- We write $\nabla f(w_0)$ for the set of subgradients of f at w_0,
- $v \in \nabla f(w_0)$ indicates that v is a subgradient of f at w_0.

Subgradients

- For differentiable f, the gradient $v = \nabla f(w_0)$ is the only subgradient.



- If $f_1, \ldots, f_K$ are differentiable at w_0 and

$$f(w) = \mathbf{max}\{f_1(w), \ldots, f_K(w)\},$$

then $v = \nabla f_k(w_0)$ is a subgradient of f at w_0, where k any index for which $f_k(w_0) = f(w_0)$.

- Subgradients are only well defined for convex functions!

Illustration: Optimization using Gradients

$f(w_1, w_2) = (w_1)^2 + 2(w_2)^2$ strictly convex, differentiable

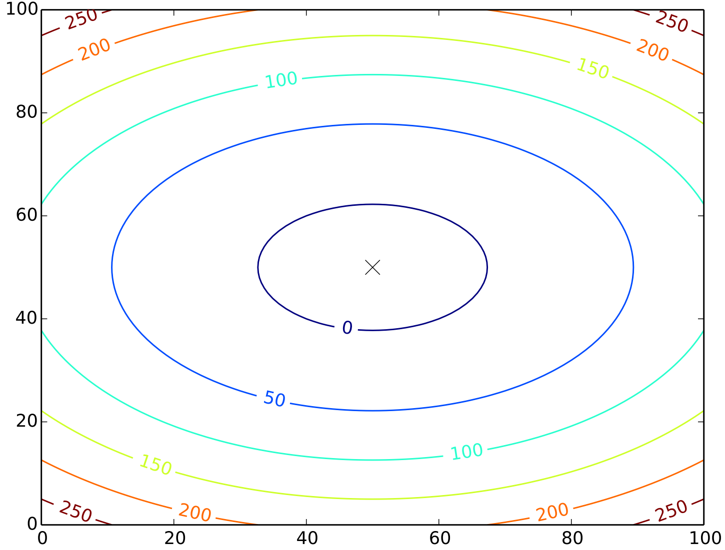


Illustration: Optimization using Gradients

$f(w_1, w_2) = (w_1)^2 + 2(w_2)^2$ strictly convex, differentiable

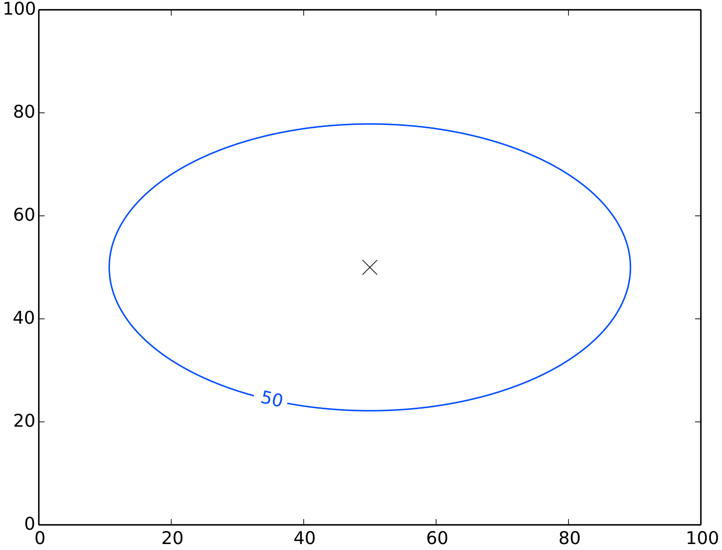
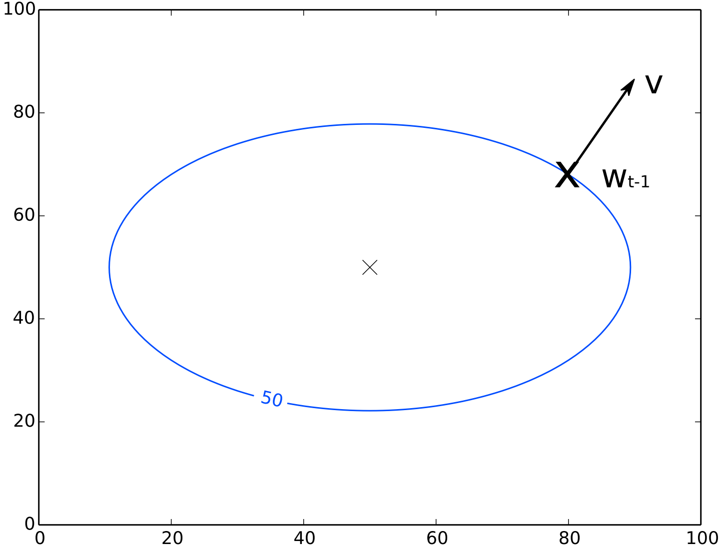


Illustration: Optimization using Gradients

$f(w_1, w_2) = (w_1)^2 + 2(w_2)^2$ strictly convex, differentiable



$f(w_1, w_2) = (w_1)^2 + 2(w_2)^2$ strictly convex, differentiable

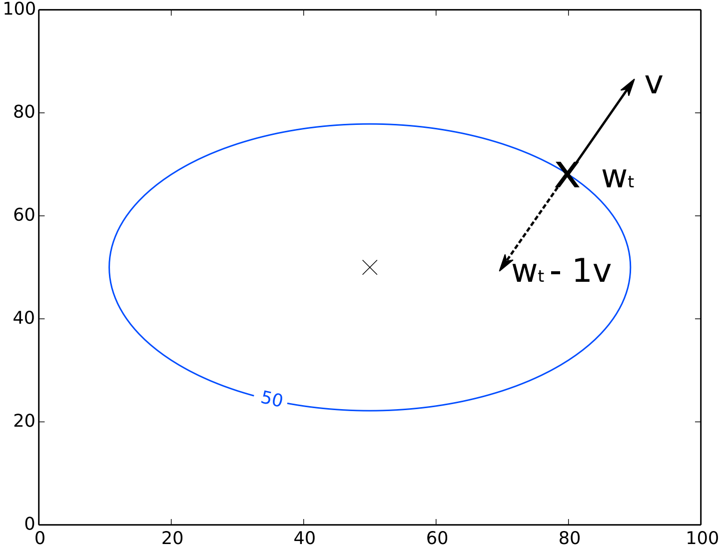


Illustration: Optimization using Gradients

$f(w_1, w_2) = (w_1)^2 + 2(w_2)^2$ strictly convex, differentiable

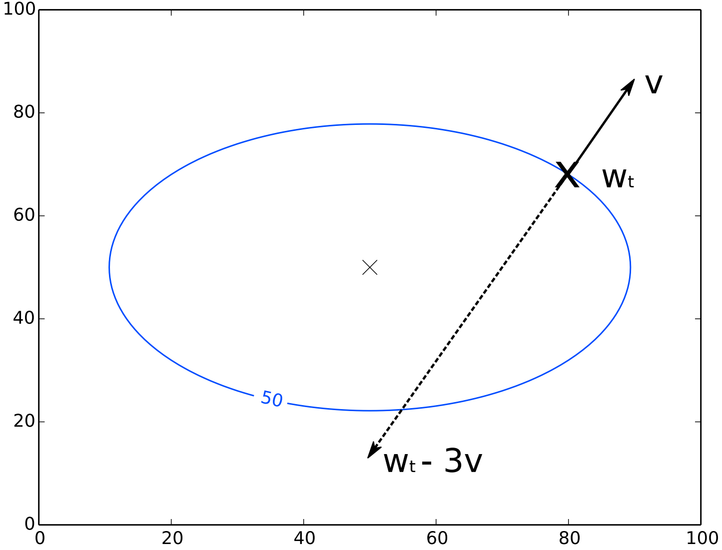
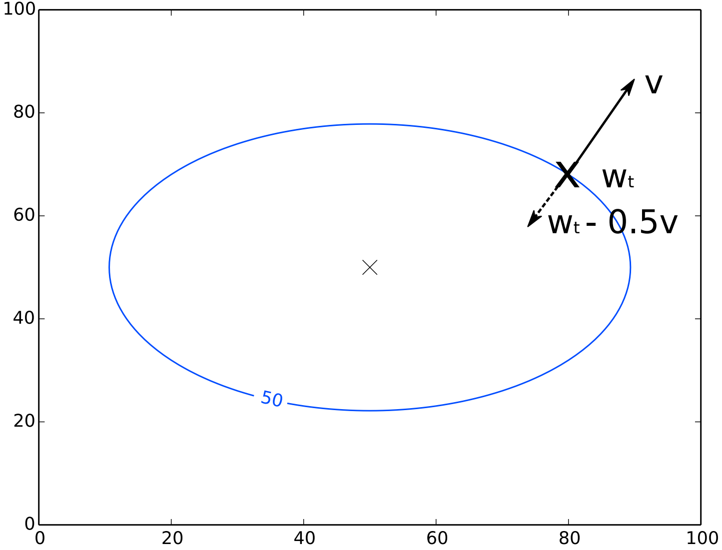
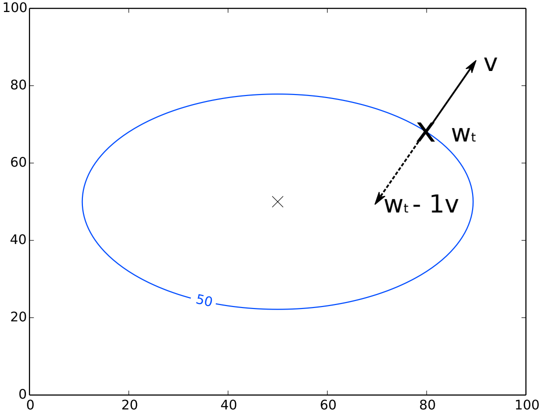


Illustration: Optimization using Gradients

$f(w_1, w_2) = (w_1)^2 + 2(w_2)^2$ strictly convex, differentiable



$$f(w_1, w_2) = (w_1)^2 + 2(w_2)^2$$ strictly convex, differentiable



Gradient of a differentiable function is a **descent direction**:

- for any w_t there exists an η such that $f(w_t + \eta v) < f(w_t)$

Illustration: Optimization using Subgradients?

$f(w_1, w_2) = |w_1| + 2|w_2|$ convex, not differentiable

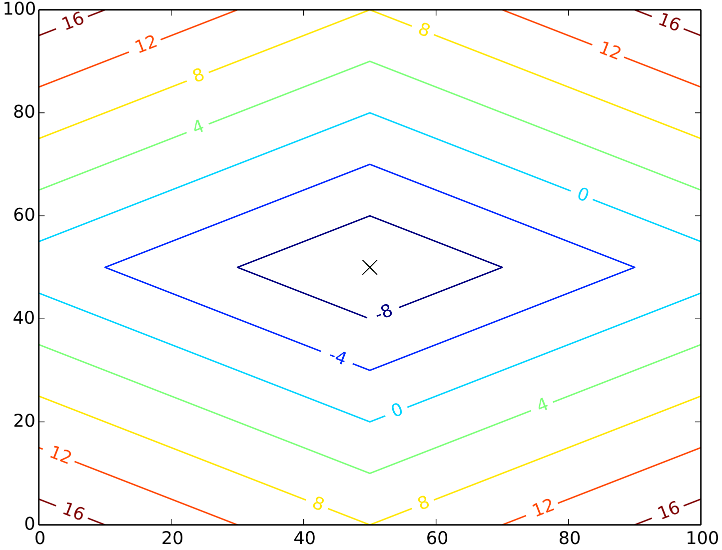


Illustration: Optimization using Subgradients?

$f(w_1, w_2) = |w_1| + 2|w_2|$ convex, not differentiable

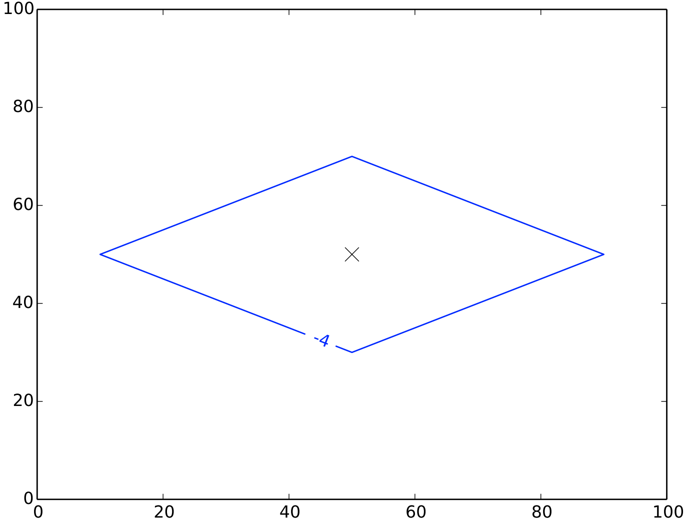


Illustration: Optimization using Subgradients?

$f(w_1, w_2) = |w_1| + 2|w_2|$ convex, not differentiable

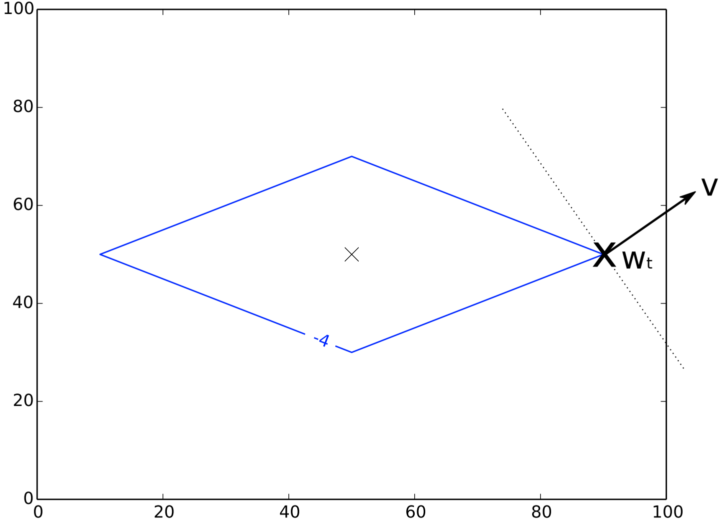


Illustration: Optimization using Subgradients?

$f(w_1, w_2) = |w_1| + 2|w_2|$ convex, not differentiable

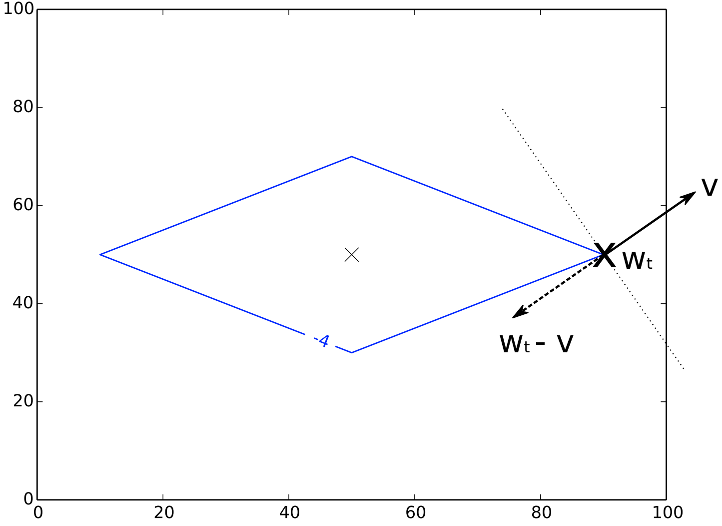


Illustration: Optimization using Subgradients?

$f(w_1, w_2) = |w_1| + 2|w_2|$ convex, not differentiable

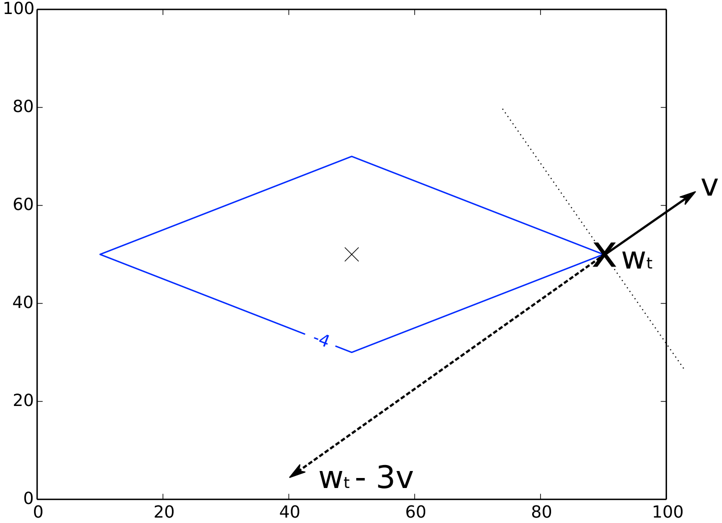
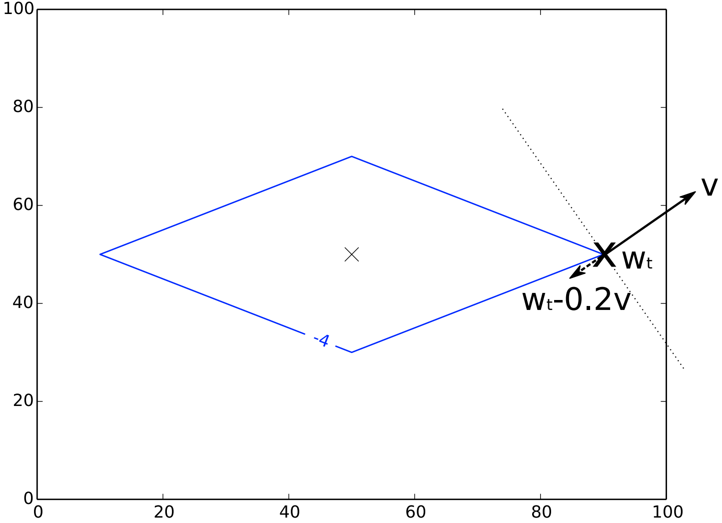
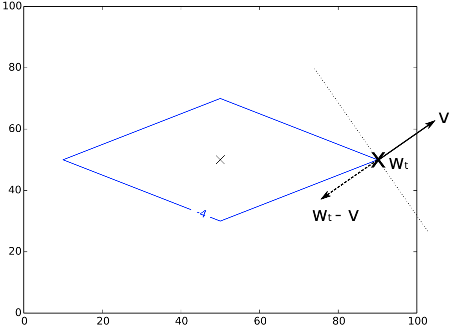


Illustration: Optimization using Subgradients?

$f(w_1, w_2) = |w_1| + 2|w_2|$ convex, not differentiable



$$f(w_1, w_2) = |w_1| + 2|w_2| \qquad \text{convex, not differentiable}$$

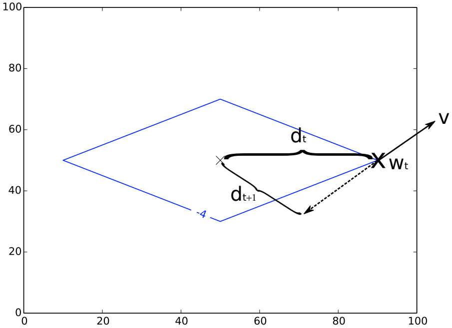


Subgradient might not be a **not a descent direction**:

- for w_t we might have $f(w_t + \eta v) \geq f(w_t)$ for all $\eta \in \mathbb{R}$

Illustration: Optimization using Subgradients?

$f(w_1, w_2) = |w_1| + 2|w_2|$ convex, not differentiable



Subgradient might not be a **not a descent direction**:

- for w_t we might have $f(w_t + \eta v) \geq f(w_t)$ for all $\eta \in \mathbb{R}$
- but: there is an η that brings us closer to the optimum, $\|w_{t+1} - w^*\| < \|w_t - w^*\|$ (Proof: exercise...)

Subgradient Method (not Descent!)

input step sizes $\eta_1, \eta_2, \ldots$

1: $w_1 \leftarrow 0$

2: **for** $t = 1, \ldots, T$ **do**

3: $\quad v \leftarrow$ a subgradient of $\mathcal{L}$ at w_t

4: $\quad w_{t+1} \leftarrow w_t - \eta_t v$

5: **end for**

output w_t with smallest values $\mathcal{L}(w_t)$ for $t = 1, \ldots, T$

Subgradient Method (not Descent!)

input step sizes $\eta_1, \eta_2, \ldots$

1: $w_1 \leftarrow 0$

2: **for** $t = 1, \ldots, T$ **do**

3: $\quad v \leftarrow$ a subgradient of $\mathcal{L}$ at w_t

4: $\quad w_{t+1} \leftarrow w_t - \eta_t v$

5: **end for**

output w_t with smallest values $\mathcal{L}(w_t)$ for $t = 1, \ldots, T$

Stepsize rules: how to choose $\eta_1, \eta_2, \ldots,$?

- $\eta_t = \eta$ constant: will get us (only) close to the optimum
- decrease slowly, but not too slowly: converges to optimum

$$\sum_{t=1}^{\infty} \eta_t = \infty \qquad \sum_{t=1}^{\infty} (\eta_t)^2 < \infty \qquad \text{e.g. } \eta_t = \frac{\eta}{t + t_0}$$

How to choose overall η? trial-and-error

- Try different values, see which one decreases the objective (fastest)

Stochastic Optimization

Many objective functions in ML contain a sum over all training exampes:

$$\mathcal{L}_{LogReg}(w) = \sum_{i=1}^{n} \log(1 + \exp(-y_i(\langle w, x_i \rangle + b))),$$

$$\mathcal{L}_{SVM}(w) = \frac{1}{2}\|w\|^2 + C \sum_{i=1}^{n} \mathbf{max}\{\, 0, 1 - y_i(\langle w, x_i \rangle + b)\}.$$

Computing the gradient or subgradient scales like $O(nd)$,

- d is the dimensionality of the data
- n is the number of training examples.

Both d and n can be big (millions). What can we do?

- we'll not get rid of $O(d)$, since $w \in \mathbb{R}^d$,
- but we can get rid of the scaling with $O(n)$ for each update!

Let $f(w) = \sum_i f_i(w)$, with convex, differentiable $f_1, \dots, f_n$.

Stochastic Gradient Descent

input step sizes $\eta_1, \eta_2, \dots$

$w_1 \leftarrow 0$

for $t = 1, \dots, T$ **do**

  $i \leftarrow$ random index in $1, 2, \dots, n$

  $v \leftarrow n \nabla f_i(w_t)$

  $w_{t+1} \leftarrow w_t - \eta_t v$

end for

output w_T, or average $\frac{1}{T-T_0} \sum_{t=T_0}^{T} w_t$

- Each iteration takes only $O(d)$,
- Gradient is "wrong" is each step, but correct in expectation.
- No line search, since evaluating $f(w - \eta v)$ would be $O(nd)$,
- Objective does not decrease in every step,
- Converges to optimum if η_t is square summable, but not summable.

Let $f(w) = \sum_i f_i(w)$, with convex $f_1, \dots, f_n$.

Stochastic Subgradient Method

input step sizes $\eta_1, \eta_2, \dots$

1: $w_1 \leftarrow 0$
2: **for** $t = 1, \dots, T$ **do**
3: $\quad i \leftarrow$ random index in $1, 2, \dots, n$
4: $\quad v \leftarrow n\bar{v}$ for $\bar{v} \in \nabla f_i(w_t)$
5: $\quad w_{t+1} \leftarrow w_t - \eta_t v$
6: **end for**

output w_T, or average $\frac{1}{T-T_0}\sum_{t=T_0}^{T} w_t$

- Each iteration takes only $O(d)$,
- Converges to optimum if η_t is square summable, but not summable.
- Even better: pick not completely at random but go in epochs: randomly shuffle dataset, go through all examples, reshuffle, etc.

Stochastic Primal SVMs Training

$$\mathcal{L}_{SVM}(w,b) = \sum_{i=1}^{n} \Big(\frac{1}{2n} \|w\|^2 + C \,\mathbf{max}\{\, 0, 1 - y_i(\langle w, x_i\rangle + b)\} \,\Big).$$

input step sizes $\eta_1, \eta_2, \dots$ or step size rule, such as $\eta_t = \frac{\eta}{t+t_0}$

```
(w_1, b_1) ← (0, 0)
for t = 1, ..., T do
    pick (x, y) from D (randomly, or in epochs)
    if y⟨x, w⟩ + b ≥ 1 then
        w_{t+1} ← (1 − η_t) w_t
    else
        w_{t+1} ← (1 − η_t) w_t + nCη_t y x
        b_{t+1} ← η_t nCy
    end if
end for
```

output w_T, or average $\frac{1}{T-T_0}\sum_{t=T_0}^{T} w_t$

Widely used for SVM training, but setting stepsizes can be painful.

Back to the original formulation

$$\min_{w\in\mathbb{R}^d, b\in\mathbb{R}, \xi\in\mathbb{R}^n} \quad \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{n}\xi^i$$

subject to, for $i = 1, \dots, n$,

$$y^i(\langle w, x^i\rangle + b) \geq 1 - \xi^i, \qquad \text{and} \qquad \xi^i \geq 0.$$

Convex optimization problem: we can study its **dual problem**.

General Principle of Dualization

Assume a constrained optimization problem:

$$\min_{\theta \in \Theta \subset \mathbb{R}^K} \quad f(\theta)$$

subject to

$$g_1(\theta) \leq 0, \quad g_2(\theta) \leq 0, \quad \dots, \quad g_k(\theta) \leq 0.$$

General Principle of Dualization

Assume a constrained optimization problem:

$$\min_{\theta \in \Theta \subset \mathbb{R}^K} \quad f(\theta)$$

subject to

$$g_1(\theta) \leq 0, \quad g_2(\theta) \leq 0, \quad \dots, \quad g_k(\theta) \leq 0.$$

We define the **Lagrangian**, that combines objective and constraints

$$\mathcal{L}(\theta, \alpha) = f(\theta) + \alpha_1 g_1(\theta) + \dots + \alpha_k g_k(\theta)$$

with **Lagrange multipliers,** $\alpha_1, \dots, \alpha_k \geq 0$. Note:

$$\max_{\alpha_1 \geq 0, \dots, \alpha_k \geq 0} \mathcal{L}(\theta, \alpha) = \begin{cases} f(\theta) & \text{if } g_1(\theta) \leq 0,\ g_2(\theta) \leq 0,\ \dots,\ g_k(\theta) \leq 0 \\ \infty & \text{otherwise.} \end{cases}$$

Any optimal solution, θ, for $\min_{\theta \in \Theta} \max_{\alpha \geq 0} \mathcal{L}(\theta, \alpha)$ is also optimal for the original constrained problem.

Theorem (Special Case of Slater's Condition)

*If f is convex, $g_1, \dots, g_k$ are affine functions, and there exists at least one point $\theta \in$ **relint**(Θ) that is feasible (i.e. $g_i(\theta) \leq 0$ for $i = 1, \dots, k$). Then*

$$\min_{\theta \in \Theta} \max_{\alpha \geq 0} \ \mathcal{L}(\theta, \alpha) \quad = \quad \max_{\alpha \geq 0} \ \min_{\theta \in \Theta} \ \mathcal{L}(\theta, \alpha)$$

General Principle of Dualization

Theorem (Special Case of Slater's Condition)

*If f is convex, $g_1, \ldots, g_k$ are affine functions, and there exists at least one point $\theta \in$ **relint**(Θ) that is feasible (i.e. $g_i(\theta) \leq 0$ for $i = 1, \ldots, k$). Then*

$$\min_{\theta \in \Theta} \max_{\alpha \geq 0} \mathcal{L}(\theta, \alpha) \quad = \quad \max_{\alpha \geq 0} \min_{\theta \in \Theta} \mathcal{L}(\theta, \alpha)$$

Call $f(\theta)$ the **primal** and $h(\alpha) = \mathbf{min}_{\theta \in \Theta} \mathcal{L}(\theta, \alpha)$ be the **dual function**.

The theorem states that minimizing the primal $f(\theta)$ (with constraints given by the g_k) is equivalent to maximizing its dual $h(\alpha)$ (with $\alpha \geq 0$).

$$\min_{\theta \in \mathbb{R}^K} f(\theta) = \max_{\alpha \in \mathbb{R}_+^k} h(\alpha)$$

Dualizing of the SVM optimization problem

The SVM optimization problem fulfills the conditions of the theorem.

$$\min_{w\in\mathbb{R}^d, b\in\mathbb{R}, \xi\in\mathbb{R}^n} \quad \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{n}\xi^i$$

subject to, for $i = 1, \dots, n$,

$$y^i(\langle w, x^i\rangle + b) \geq 1 - \xi^i, \qquad \text{and} \qquad \xi^i \geq 0.$$

We can compute its minimal value as $\max_{\alpha\geq 0,\beta\geq 0} h(\alpha,\beta)$ with

$$h(\alpha,\beta) = \min_{(w,b)} \quad \frac{1}{2}\|w\|^2 + C\sum_i \xi_i + \sum_i \alpha_i\big(1 - \xi_i - y^i(\langle w, x^i\rangle + b)\big) - \sum_i \beta_i\xi_i$$

(Blackboard...)

Dualizing of the SVM optimization problem

In a minimum w.r.t. (w, b):

$$0 = \frac{\partial}{\partial w}\mathcal{L}(w, b, \xi, \alpha, \beta) = w - \sum_i \alpha_i y^i x^i \quad \Rightarrow \quad w = \sum_i \alpha_i y^i x^i$$

$$0 = \frac{\partial}{\partial b}\mathcal{L}(w, b, \xi, \alpha, \beta) = \sum_i \alpha_i y^i$$

$$0 = \frac{\partial}{\partial \xi_i}\mathcal{L}(w, b, \xi, \alpha, \beta) = C - \alpha_i - \beta_i$$

Insert new constraints into objective:

$$\max_{\alpha \geq 0} \; \frac{1}{2}\|\sum_i \alpha_i y^i x^i\|^2 + \sum_i \alpha_i - \sum_i \alpha_i y_i \langle \sum_j \alpha_j y^j x^j, x^i \rangle$$

$$\max_{\alpha \geq 0} \quad -\frac{1}{2}\sum_{i,j} \alpha_i \alpha_j y^i y^j \langle x^i, x^j \rangle + \sum_i \alpha_i$$

subject to $\sum_i \alpha_i y_i = 0$ and $0 \leq \alpha_i \leq C$, for $i = 1, \dots, n$.

- Examples x^i with $\alpha_i \neq 0$ are called **support vectors**.
- From the coefficients $\alpha_1, \dots, \alpha_n$ we can recover the optimal w:

$$w = \sum_i \alpha_i y^i x^i$$

$$b = 1 - y^i \langle x^i, w \rangle \qquad \text{for any } i \text{ with } 0 < \alpha_i < C$$

(more complex rule for b if no such i exists).

- The prediction rule becomes

$$g(x) = \operatorname{sign}\Big(\langle w, x \rangle + b\Big) = \operatorname{sign}\Big(\sum_{i=1}^{n} \alpha_i y_i \langle x_i, x \rangle + b\Big)$$

SVM Dual Optimization Problem

$$\max_{\alpha \geq 0} \quad -\frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y^i y^j \langle x^i, x^j \rangle + \sum_i \alpha_i$$

subject to

$$\sum_i \alpha_i y_i = 0 \qquad \text{and} \qquad 0 \leq \alpha_i \leq C, \quad \text{for } i = 1, \ldots, n.$$

Why solve the dual optimization problem?

- fewer unknowns: $\alpha \in \mathbb{R}^n$ instead of $(w, b, \xi) \in \mathbb{R}^{d+1+n}$
- less storage when $d \gg n$:
  $(\langle x^i, x^j \rangle)_{i,j} \in \mathbb{R}^{n \times n}$ instead of $(x^1, \ldots, x^n) \in \mathbb{R}^{n \times d}$
- Kernelization (not in this course)

SVMs Without Bias Term

For optimization, the *bias term* is an annoyance

- In primal optimization, it often requires a different stepsize.
- In dual optimization, sometimes not straight-forward to recover.
- It couples the dual variables by an equality constraint: $\sum_i \alpha_i y_i = 0$.

We can get rid of the bias by the **augmentation trick**.

Original:

- $f(x) = \langle w, x \rangle_{\mathbb{R}^d} + b$, with $w \in \mathbb{R}^d, b \in \mathbb{R}$.

New augmented:

- linear: $f(x) = \langle \tilde{w}, \tilde{x} \rangle_{\mathbb{R}^{d+1}}$, with $\tilde{w} = (w, b)$, $\tilde{x} = (x, 1)$.
- generalized: $f(x) = \langle \tilde{w}, \tilde{\phi}(x) \rangle_{\tilde{\mathcal{H}}}$ with $\tilde{w} = (w, b)$, $\tilde{\phi}(x) = (\phi(x), 1)$.

SVMs Without Bias Term – Optimization

SVM without bias term – primal optimization problem

$$\min_{w\in\mathbb{R}^d,\xi\in\mathbb{R}^n} \quad \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{n}\xi^i$$

subject to, for $i = 1, \dots, n$,

$$y^i\langle w, x^i\rangle \geq 1 - \xi^i, \qquad \text{and} \qquad \xi^i \geq 0.$$

Difference: no b variable to optimize over

SVMs Without Bias Term – Optimization

SVM without bias term – primal optimization problem

$$\min_{w\in\mathbb{R}^d,\xi\in\mathbb{R}^n} \quad \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{n}\xi^i$$

subject to, for $i = 1, \dots, n$,

$$y^i\langle w, x^i\rangle \geq 1 - \xi^i, \qquad \text{and} \qquad \xi^i \geq 0.$$

Difference: no b variable to optimize over

SVM without bias term – dual optimization problem

$$\max_{\alpha} \quad -\frac{1}{2}\sum_{i,j}\alpha_i\alpha_j y^i y^j \left\langle x^i, x^j\right\rangle + \sum_i \alpha_i$$

subject to, $\quad 0 \leq \alpha_i \leq C, \quad$ for $i = 1, \dots, n$.

Difference to variant with bias term: no constraint $\sum_i y_i\alpha_i = 0$.

Linear SVM Optimization in the Dual

Stochastic Coordinate Dual Ascent

$\alpha \leftarrow \mathbf{0}$.
for $t = 1, \dots, T$ **do**
  $i \leftarrow$ random index (uniformly random or in epochs)
  solve QP w.r.t. α_i with all α_j for $j \neq i$ fixed.
end for
return α

Properties:

- converges monotonically to global optimum
- each subproblem has smallest possible size: 1-dimensional

Open Problem:

- how to make each step efficient?

SVM Optimization in the Dual

What's the complexity of the update step? Derive an explicit expression:

Original problem: $\mathbf{max}_{\alpha \in [0,C]^n} \quad -\frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y^i y^j \langle x^i, x^j \rangle + \sum_i \alpha_i$

SVM Optimization in the Dual

What's the complexity of the update step? Derive an explicit expression:

Original problem: $\mathbf{max}_{\alpha\in[0,C]^n} \quad -\frac{1}{2}\sum_{i,j}\alpha_i\alpha_j y^i y^j \langle x^i, x^j\rangle + \sum_i \alpha_i$

When all α_j except α_i are fixed: $\quad \mathbf{max}_{\alpha_i\in[0,C]} F(\alpha_i)$, $\quad$ with

$$F(\alpha_i) = -\frac{1}{2}\alpha_i^2 \underbrace{\langle x^i, x^i\rangle}_{=\|x^i\|^2} + \alpha_i\Big(1 - y^i \sum_{j\neq i}\alpha_j y^j \langle x^i, x^j\rangle\Big) + \text{const.}$$

$$\frac{\partial}{\partial\alpha_i}F(\alpha_i) = -\alpha_i\|x^i\|^2 + \Big(1 - y^i\sum_{j\neq i}\alpha_j y^j\langle x^i, x^j\rangle\Big) + \text{const.}$$

$$\alpha_i^{\text{new}} = \alpha_i + \frac{1 - y^i\sum_{j=1}^n \alpha_j y^j \langle x^i, x^j\rangle}{\|x^i\|^2}, \quad \alpha_i = \begin{cases} 0 & \text{if } \alpha_i^{\text{new}} < 0, \\ C & \text{if } \alpha_i^{\text{new}} > C, \\ \alpha_i^{\text{new}} & \text{otherwise.} \end{cases}$$

(α_i show up, because sum range is $j = 1, \ldots, n$, not $j \neq i$)

- complexity of each update: n inner products $= O(nd)$
- if we pre-compute and store all $\langle x_i, x_j\rangle$: $O(n)$ with $O(n^2)$ storage

(Generalized) Linear SVM Optimization in the Dual

For $n \gg d$, we can improve using the linearity of $\langle\cdot,\cdot\rangle$:

$$\alpha_i^{\text{new}} = \alpha_i + \frac{1 - y^i \sum_j \alpha_j y^j \langle x^i, x^j \rangle}{\|x^i\|^2}$$

$$= \alpha_i + \frac{1 - y^i \langle x^i, \sum_j \alpha_j y^j x^j \rangle}{\|x^i\|^2}$$

remember $w = \sum_j \alpha_j y^j x^j$. If we keep w stored explicitly:

$$= \alpha_i + \frac{1 - y^i \langle w, x^i \rangle}{\|x^i\|^2},$$

- each update: $O(d)$, independent of n
  - $\langle w, x^i \rangle$ takes $O(d)$ for explicit $w \in \mathbb{R}^d$
  - taking care that w stays up-to-date: also $O(d)$

$$w^{\text{new}} = w^{\text{old}} + (\alpha_i^{\text{new}} - \alpha_i^{\text{old}}) y^i x^i$$

SCDA for (Generalized) Linear SVMs [Hsieh, 2008]

initialize $\alpha \leftarrow \mathbf{0}$, $w \leftarrow \mathbf{0}$
for $t = 1, \dots, T$ **do**
  $i \leftarrow$ random index (uniformly random or in epochs)
  $\delta \leftarrow \frac{1 - y^i \langle w, x^i \rangle}{\|x^i\|^2}$
  $$\bar{\alpha} \leftarrow \begin{cases} 0, & \text{if } \alpha_i + \delta < 0, \\ C, & \text{if } \alpha_i + \delta > C, \\ \alpha_i + \delta, & \text{otherwise.} \end{cases}$$
  $w \leftarrow w + (\bar{\alpha} - \alpha_i) y^i x^i$
  $\alpha_i \leftarrow \bar{\alpha}$
end for
return α, w

Properties:

- converges monotonically to global optimum
- complexity of each step is independent of n
- resembles stochastic gradient method, but **step size is automatic**