Statistical Machine Learning

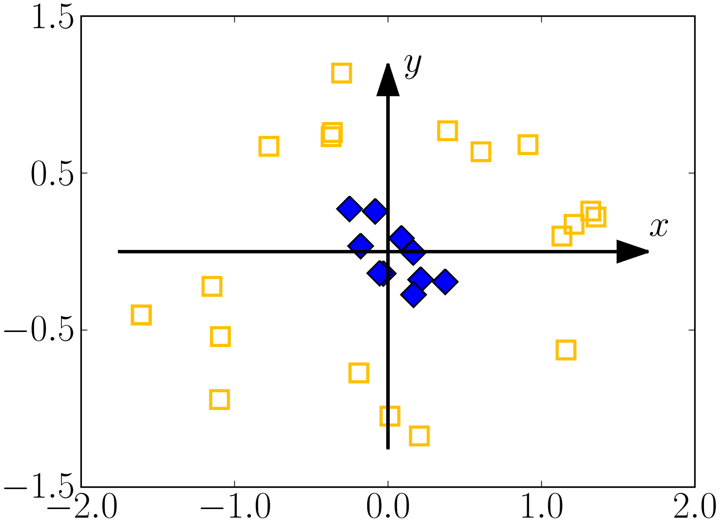
Christoph Lampert

IST AUSTRIA

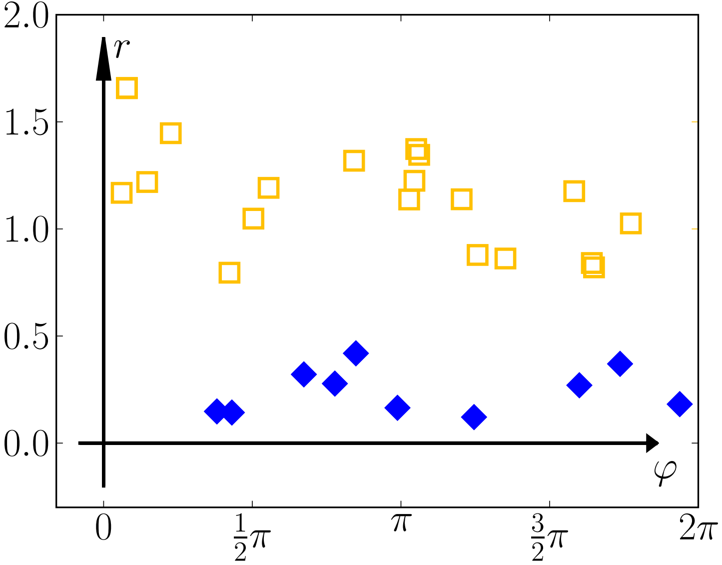
Institute of Science and Technology

Spring Semester 2015/2016 // Lecture 4

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 → polar coordinates

Definition (Max-margin Generalized Linear Classifier)

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

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

Let $\phi : \mathcal{X} \to \mathcal{H}$ be a feature map from $\mathcal{X}$ into a Hilbert space $\mathcal{H}$.

Then we can form a new training set

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

The maximum-(soft)-margin linear classifier in $\mathcal{H}$,

$$g(x) = \operatorname{sign}[\langle w, \phi(x)\rangle_{\mathcal{H}} + b]$$

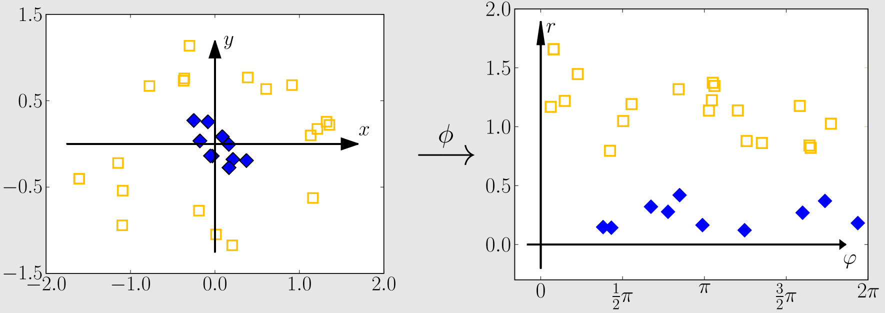
for $w \in \mathcal{H}$ and $b \in \mathbb{R}$ is called **max-margin generalized linear classifier**.

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 $\phi : \mathcal{X} \to \mathcal{H}$ with $\mathcal{X} = \mathbb{R}^2$ and $\mathcal{H} = \mathbb{R}^2$

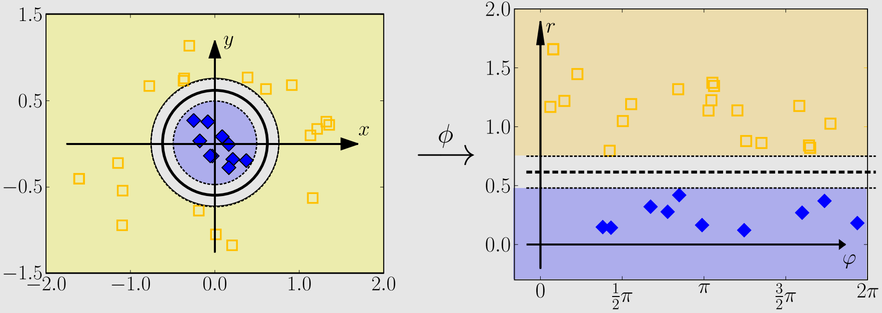
$$\phi(x, y) = (\sqrt{x^2 + y^2}, \arctan \frac{y}{x}) \qquad \text{(and } \phi(0,0) = (0,0))$$



Example (Polar coordinates)

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

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



Any classifier in $\mathcal{H}$ induces a classifier in $\mathcal{X}$.

Other popular feature mappings, ϕ

Example (d-th degree polynomials)

$$\phi : \big(x_1, \ldots, x_n\big) \mapsto \big(1, x_1, \ldots, x_n, x_1^2, \ldots, x_n^2, \ldots, x_1^d, \ldots, x_n^d\big)$$

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

$$f(x) = \langle w, \phi(x)\rangle = \sum_j w_j \phi(x)_j = \sum_i a_i x_i + \sum_{ij} b_{ij} x_i x_j + \ldots$$

Example (Distance map)

For a set of prototype $p_1, \ldots, p_N \in \mathcal{H}$:

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

Classifier: combine weights from close enough prototypes

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

(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

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

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

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

subject to

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

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 ($d \gg n$)

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

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

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

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

For any fixed (w, b) we can find the optimal $\xi_1, \dots, \xi_n$:

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

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

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

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:

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

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

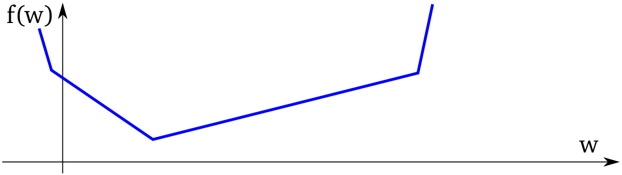
$$\min_{w\in\mathbb{R}^d, b\in\mathbb{R}} \quad \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{n} \mathbf{max}\{\, 0, 1 - y_i(\langle w, x_i\rangle + b)\}.$$

- 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!*

We can't use gradient descent, since some points have no gradients!

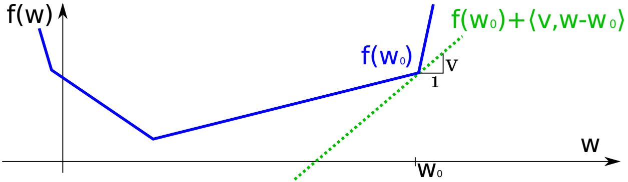
Definition: Let $f : \mathbb{R}^d \to \mathbb{R}$ be a **convex** function. A vector $v \in \mathbb{R}^d$ is called a **subgradient** of f at w_0, if

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



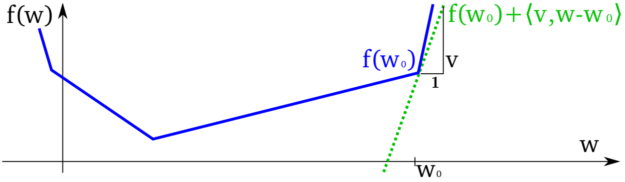
Definition: Let $f : \mathbb{R}^d \to \mathbb{R}$ be a **convex** function. A vector $v \in \mathbb{R}^d$ is called a **subgradient** of f at w_0, if

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



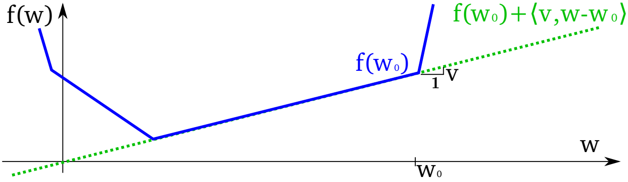
Definition: Let $f : \mathbb{R}^d \to \mathbb{R}$ be a **convex** function. A vector $v \in \mathbb{R}^d$ is called a **subgradient** of f at w_0, if

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



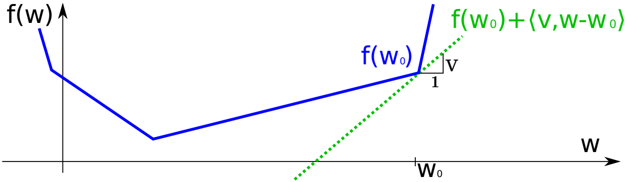
Definition: Let $f : \mathbb{R}^d \to \mathbb{R}$ be a **convex** function. A vector $v \in \mathbb{R}^d$ is called a **subgradient** of f at w_0, if

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



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

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

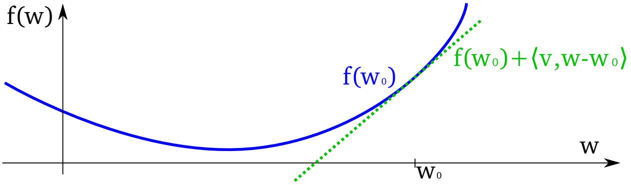


A general convex f can have more than one subgradient at a position.

- 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

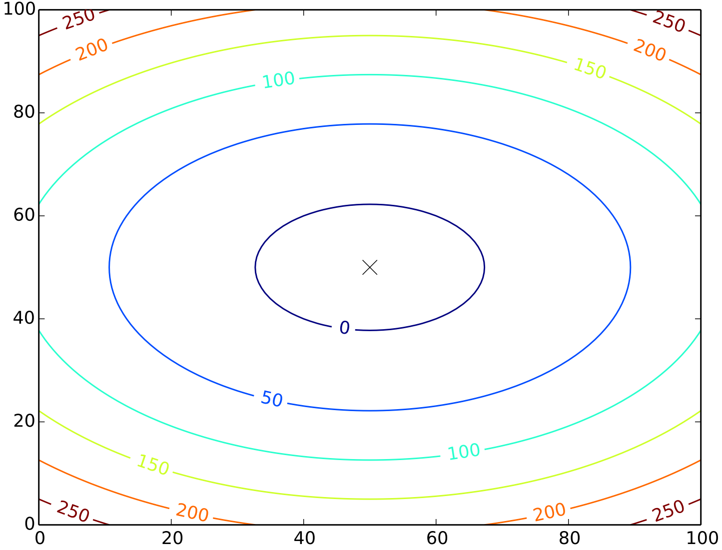
$$f(w) = \mathbf{max}\{f_1(w), \ldots, f_K(w)\},$$

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

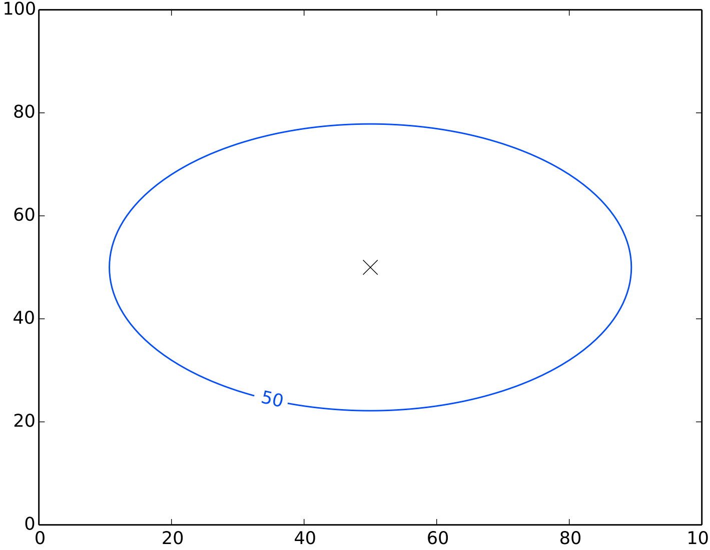
- Subgradients are only well defined for convex functions!

Illustration: Optimization using Gradients

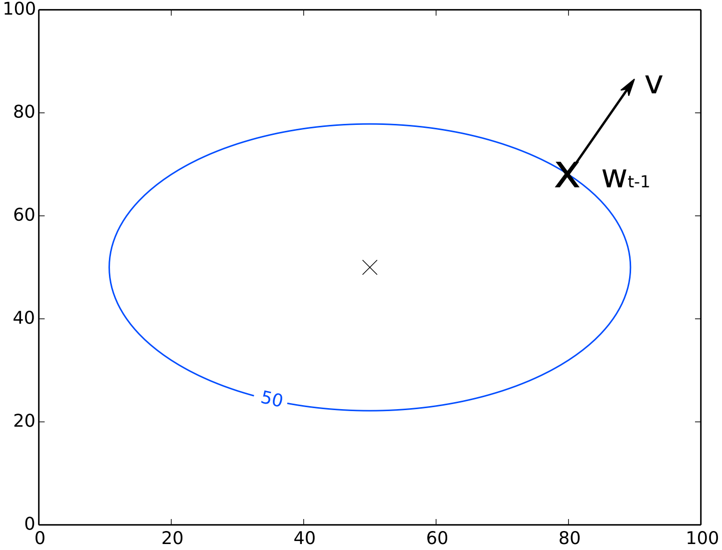
$f(w_1, w_2) = (w_1)^2 + 2(w_2)^2$ strictly convex, differentiable



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



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



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

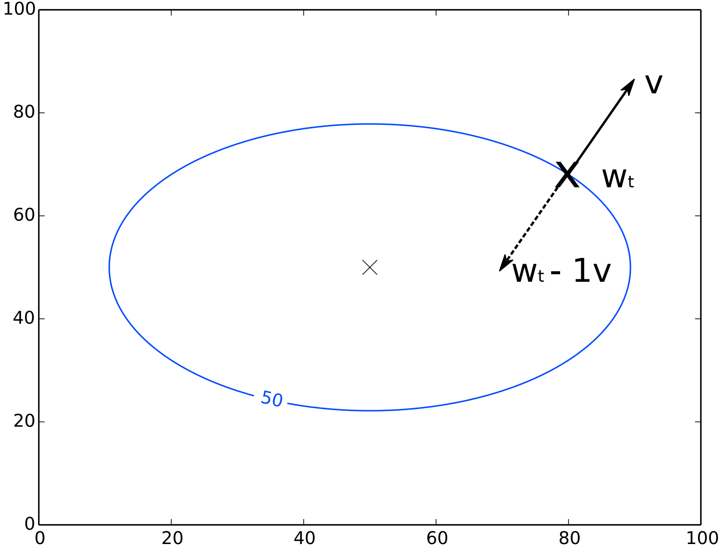
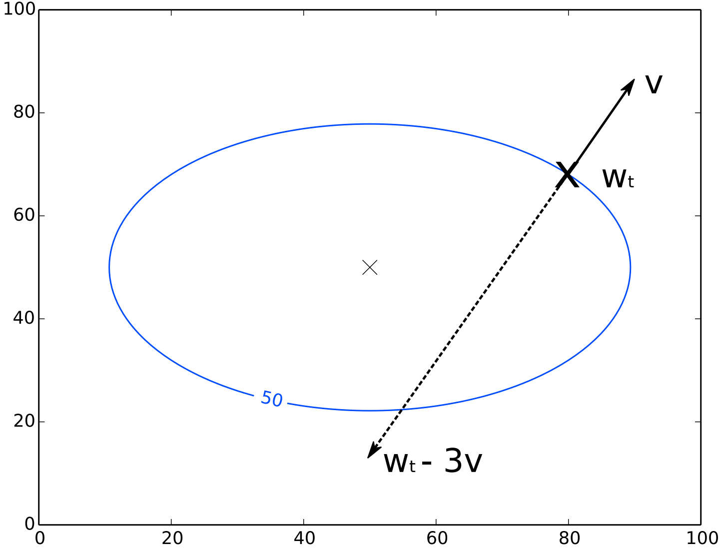


Illustration: Optimization using Gradients

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



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

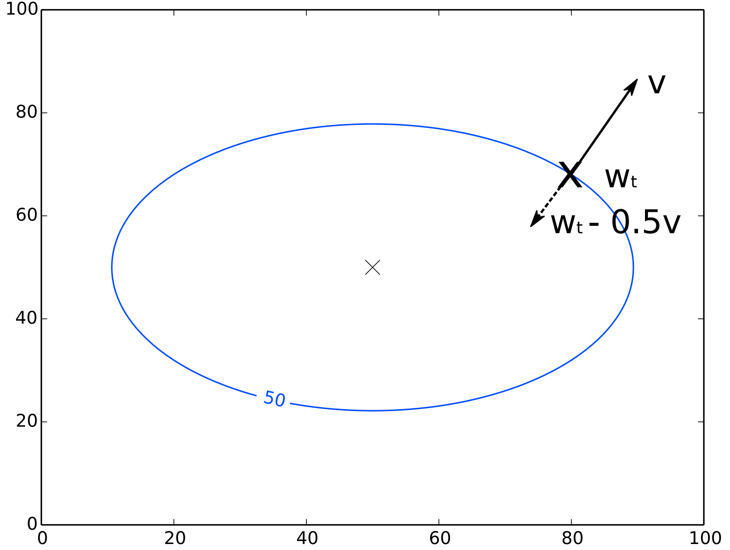
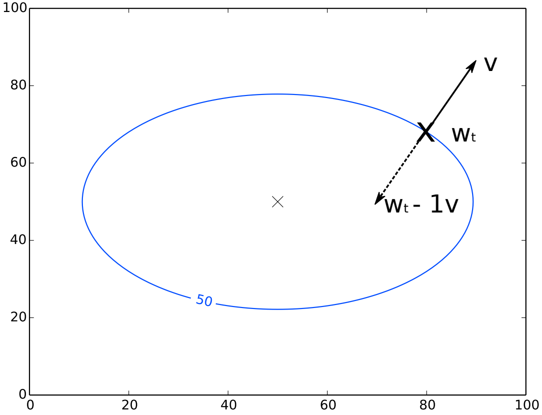


Illustration: Optimization using Gradients

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



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

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

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

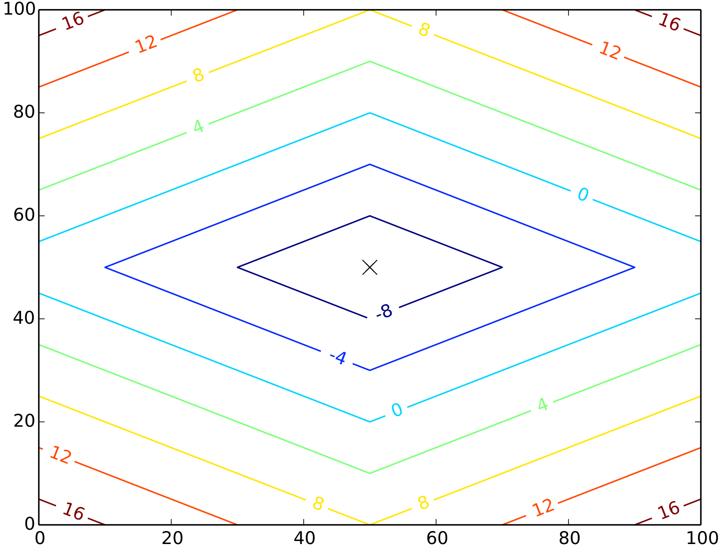


Illustration: Optimization using Subgradients?

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

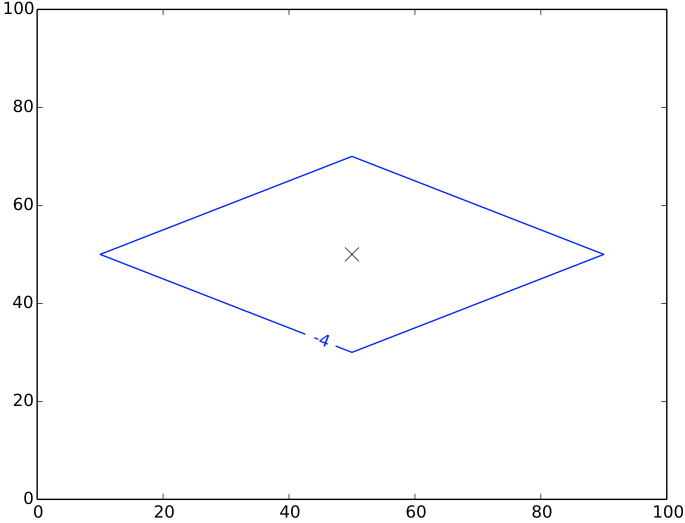


Illustration: Optimization using Subgradients?

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

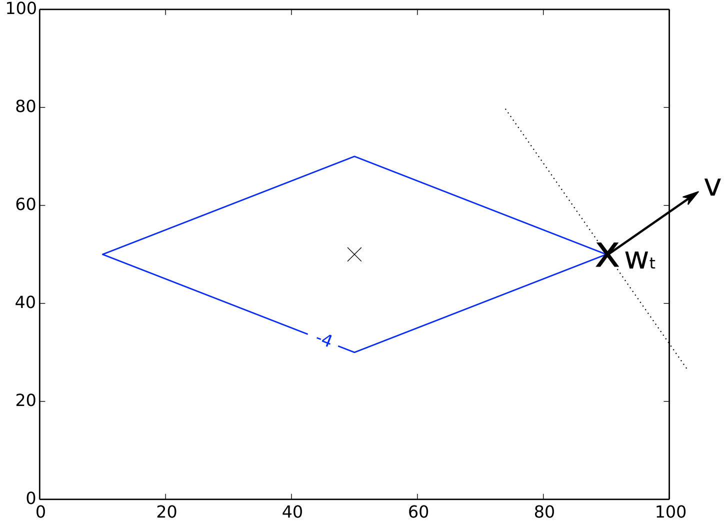


Illustration: Optimization using Subgradients?

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

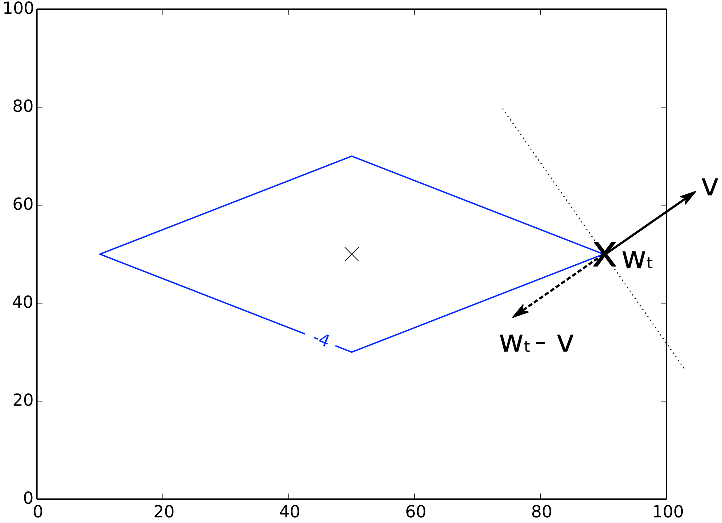
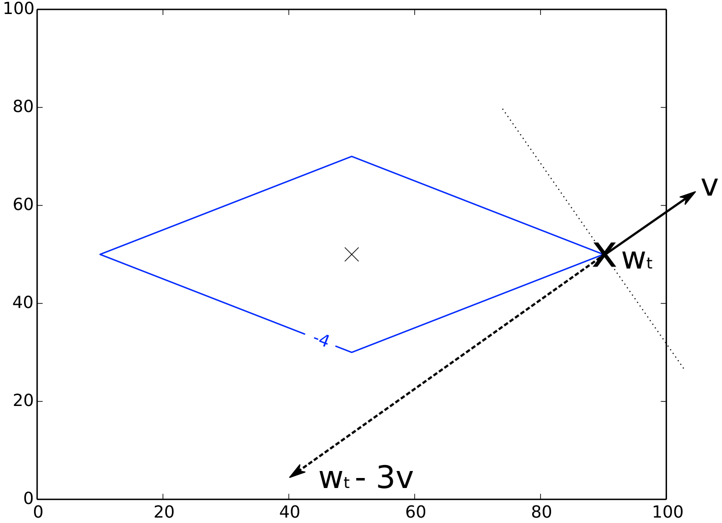


Illustration: Optimization using Subgradients?

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



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

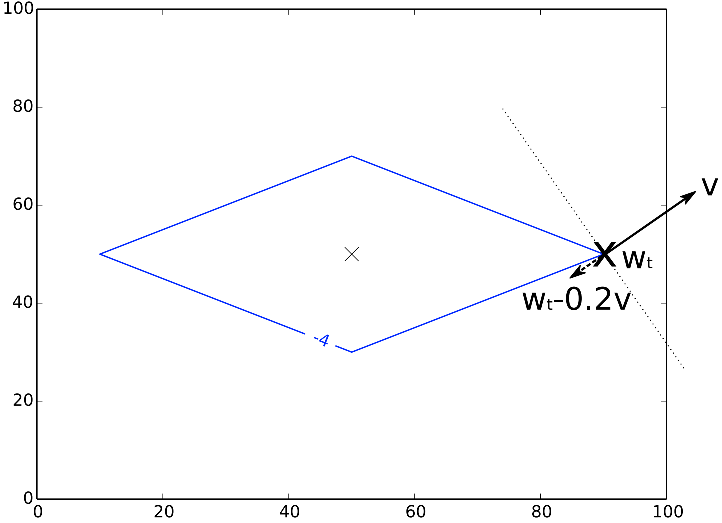
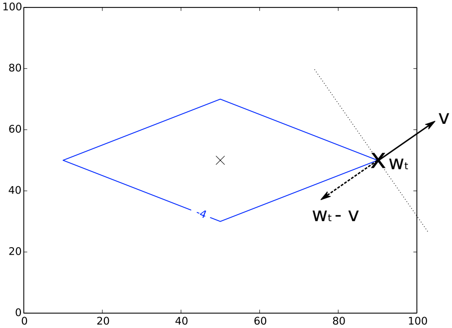


Illustration: Optimization using Subgradients?

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

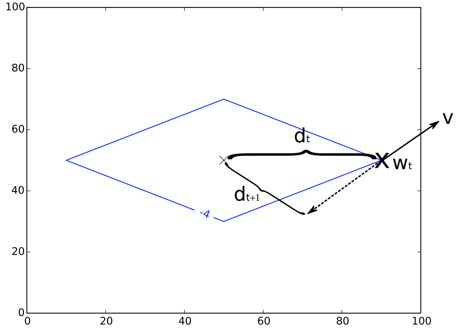


Subgradient might not be a **not a descent direction**:

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

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 $\eta_1, \eta_2, \ldots$

$w_1 \leftarrow 0$
for $t = 1, \ldots, T$ **do**
  $v \leftarrow$ a subgradient of $\mathcal{L}$ at w_t
  $w_{t+1} \leftarrow w_t - \eta_t v$
end for

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

Subgradient Method (not Descent!)

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

$w_1 \leftarrow 0$
for $t = 1, \ldots, T$ **do**
    $v \leftarrow$ a subgradient of $\mathcal{L}$ at w_t
    $w_{t+1} \leftarrow w_t - \eta_t v$
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 $$f(w) = \sum_{i=1}^{n} f_i(w),$$ with convex, differentiable $f_1, \dots, f_n$.

Stochastic Gradient Descent

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

```
w_1 ← 0
for t = 1, ..., T do
    i ← random index in 1, 2, ..., n
    v ← n ∇f_i(w_t)
    w_{t+1} ← w_t − η_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 $f(w) = \sum_{i=1}^{n} f_i(w)$, with convex $f_1, \dots, f_n$.

Stochastic Subgradient Method

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

$w_1 \leftarrow 0$

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

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

  $v \leftarrow n$ times a subgradient of f_i at 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 $O(d)$,
- 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 $\eta_1, \eta_2, \ldots$ 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 yx
        b_{t+1} ← η_t nCy
    end if
end for
```

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

State-of-the-art in 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 $i = 1, \ldots, n$,

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

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

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

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

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

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.

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. $g_i(\theta) \le 0$ for $i = 1, \ldots, k$). Then*

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

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 $\mathbf{max}_{\alpha\geq 0, \beta\geq 0}\, h(\alpha, \beta)$ with

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

(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:

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

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 $\sum_i \alpha_i y_i = 0$ 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 not 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

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

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 $(x^1, \dots, x^n) \in \mathbb{R}^{n \times d}$
- **Kernelization**

Definition (Positive Definite Kernel Function)

Let $\mathcal{X}$ be a non-empty set. A function $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ is called **positive definite kernel function**, if the following conditions hold:

- k is symmetric, i.e. $k(x, x') = k(x', x)$ for all $x, x' \in \mathcal{X}$.
- For any finite set of points $x_1, \ldots, x_n \in \mathcal{X}$, the *kernel matrix*

$$K_{ij} = \left(k(x_i, x_j)\right)_{i,j} \tag{1}$$

  is positive semidefinite, i.e. for all vectors $t \in \mathbb{R}^n$

$$\sum_{i,j=1}^{n} t_i K_{ij} t_j \geq 0. \tag{2}$$

Lemma (Kernel function)

Let $\phi : \mathcal{X} \to \mathcal{H}$ be a feature map into a Hilbert space $\mathcal{H}$. Then the function

$$k(x, \bar{x}) = \langle\ \phi(x), \phi(\bar{x})\ \rangle_{\mathcal{H}}$$

is a positive definite kernel function.

Proof.

- symmetry: $k(x, \bar{x}) = \langle \phi(x), \phi(\bar{x}) \rangle_{\mathcal{H}} = \langle \phi(\bar{x}), \phi(x) \rangle_{\mathcal{H}} = k(\bar{x}, x)$
- positive definiteness: $x_1, \dots, x_n \in \mathcal{X}$, and arbitrary $t \in \mathbb{R}^n$, then

$$\sum_{i,j=1}^{n} t_i k(x_i, x_j) t_j = \sum_{i,j=1}^{n} t_i t_j \langle \phi(x^i), \phi(x^j) \rangle_{\mathcal{H}}$$

$$= \Big\langle \sum_i t_i \phi(x^i), \sum_j t_j \phi(x^j) \Big\rangle_{\mathcal{H}} = \Big\| \sum_i t_i \phi(x^i) \Big\|_{\mathcal{H}}^2 \geq 0.$$

□

Theorem (Mercer's Condition)

Let $\mathcal{X}$ be non-empty set. For any positive definite kernel function $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$, there exists a Hilbert space $\mathcal{H}$ with inner product $\langle \cdot\,, \cdot \rangle_{\mathcal{H}}$, and a feature map $\phi : \mathcal{X} \to \mathcal{H}$ such that

$$k(x, \bar{x}) = \langle \ \phi(x), \phi(\bar{x}) \ \rangle_{\mathcal{H}}.$$

Proof. later, in more refined form

Note: $\mathcal{H}$ and ϕ are not unique, e.g.

$$k(x, \bar{x}) = 2x\bar{x}$$

- $\mathcal{H}_1 = \mathbb{R}$, $\phi_1(x) = \sqrt{2}x$, $\quad \langle \phi_1(x), \phi_1(\bar{x}) \rangle_{\mathcal{H}_1} = 2x\bar{x}$
- $\mathcal{H}_2 = \mathbb{R}^2$, $\phi_2(x) = \begin{pmatrix} x \\ -x \end{pmatrix}$, $\quad \langle \phi_1(x), \phi_2(\bar{x}) \rangle_{\mathcal{H}_2} = 2x\bar{x}$
- $\mathcal{H}_3 = \mathbb{R}^3$, $\phi_3(x) = \begin{pmatrix} x \\ 0 \\ x \end{pmatrix}$, $\quad \langle \phi_3(x), \phi_3(\bar{x}) \rangle_{\mathcal{H}_3} = 2x\bar{x}$, etc.

Definition (Reproducing Kernel Hilbert Space)

Let $\mathcal{H}$ be a Hilbert space of functions $f : \mathcal{X} \to \mathbb{R}$. A kernel $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ is called **reproducing kernel**, if

$$f(x) = \langle k(x, \cdot),\ f(\cdot)\rangle_{\mathcal{H}} \qquad \text{for all } f \in \mathcal{H}.$$

$\mathcal{H}$ is then called a **reproducing kernel Hilbert space (RKHS).**

Theorem (Moore-Aronszajn Theorem)

Let $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ be a positive definite kernel on $\mathcal{X}$. Then there is a unique Hilbert space of functions, $f : \mathcal{X} \to \mathbb{R}$, for which k is a reproducing kernel.

Proof sketch. One can construct the space explicitly: Set

$$\mathcal{H}^{\text{pre}} = \text{span}\{\ k(\cdot, x) \text{ for } x \in \mathcal{X}\ \},$$

i.e., for every $f \in \mathcal{H}^{\text{pre}}$ exist $x^1, \dots, x^m \in \mathcal{X}$ and $\alpha^1, \dots, \alpha^m \in \mathbb{R}$, with

$$f(\cdot) = \sum_{i=1}^{m} \alpha^i k(\cdot, x^i).$$

We define an inner product as

$$\langle f, g \rangle = \Big\langle \sum_i \alpha^i k(\cdot, x^i), \sum_j \bar{\alpha}^j k(\cdot, \bar{x}^j) \Big\rangle := \sum_{i,j} \alpha^i \bar{\alpha}^j k(x^i, \bar{x}^j).$$

Make $\mathcal{H}^{\text{pre}}$ into Hilbert space $\mathcal{H}$ by enforcing *completeness*.

Complete proof: [B. Schölkopf, A. Smola, *"Learning with Kernels"*, 2001].

Let

- $\mathcal{D} = \{(x^1, y^1), \ldots, (x^n, y^n)\ \} \subset \mathcal{X} \times \{\pm 1\}$ training set
- $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ be a pos.def. kernel with feature map $\phi : \mathcal{X} \to \mathcal{H}$.

Support Vector Machine in Kernelized Form

For any $C > 0$, the max-margin classifier for the feature map ϕ is

$$g(x) = \operatorname{sign} f(x) \qquad \text{with} \quad f(x) = \sum_i \alpha_i k(x^i, x) + b,$$

for coefficients $\alpha_1, \ldots, \alpha_n$ obtained by solving

$$\min_{\alpha^1,\ldots,\alpha^n \in \mathbb{R}} \quad -\frac{1}{2} \sum_{i,j=1}^{n} \alpha^i \alpha^j y^i y^j k(x^i, x^j) + \sum_{i=1}^{n} \alpha^i$$

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

Note: we don't need to know ϕ or $\mathcal{H}$, explicitly. Knowing k is enough.

Useful and Popular Kernel Functions

For $x, \bar{x} \in \mathbb{R}^d$:

- $k(x, \bar{x}) = (1 + \langle x, x' \rangle)^p$ for $p \in \mathbb{N}$ *(polynomial kernel)*

  $f(x) = \sum_i \alpha_i y^i k(x^i, x) =$ polynomial of degree d

- $k(x, \bar{x}) = \exp(-\lambda \|x - \bar{x}\|^2)$ for $\lambda > 0$ *(Gaussian or RBF kernel)*

  $f(x) = \sum_i \alpha_i y^i \exp(-\lambda \|x^i - x\|^2) =$ weighted/soft nearest neighbor

For $x, \bar{x}$ histograms with d bins:

- $k(x, \bar{x}) = \sum_{j=1}^{d} \mathbf{min}(x_j, \bar{x}_j)$ histogram intersection kernel
- $k(x, \bar{x}) = \sum_{j=1}^{d} \frac{x_j \bar{x}_j}{x_j + \bar{x}_j}$ χ^2 kernel
- $k(x, \bar{x}) = \exp\big(-\lambda \sum_{j=1}^{d} \frac{(x_j - \bar{x}_j)^2}{x_j + \bar{x}_j}\big)$ exponentiated χ^2 kernel

Generally: interpret kernel function as a **similarly measure**.