Support Vector Machine and Convex Optimization

Ian En-Hsu Yen

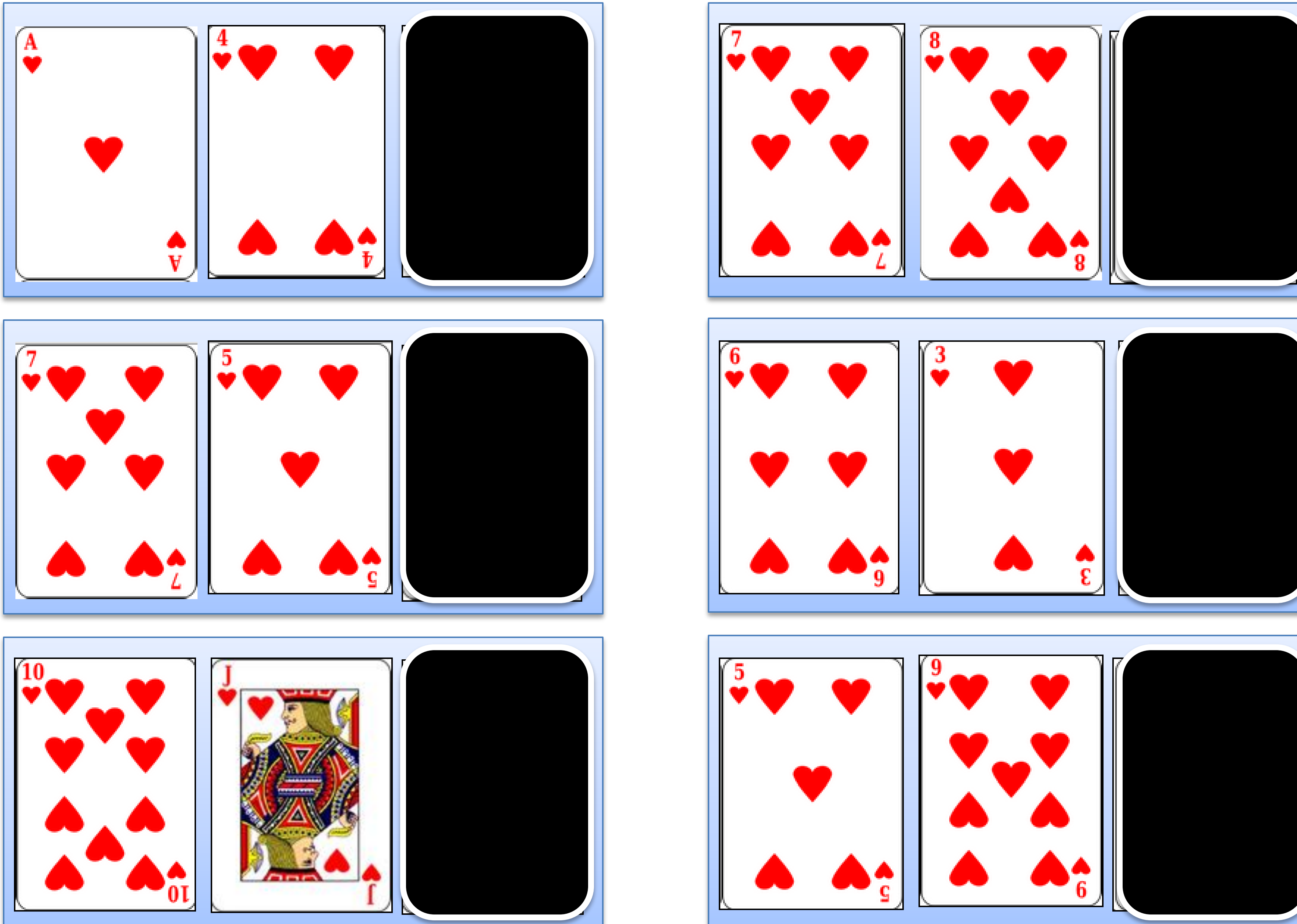
Overview

- **Support Vector Machine**
  - The Art of Modeling --- Large Margin and Kernel Trick
  - Convex Analysis
  - Optimality Conditions
  - Duality

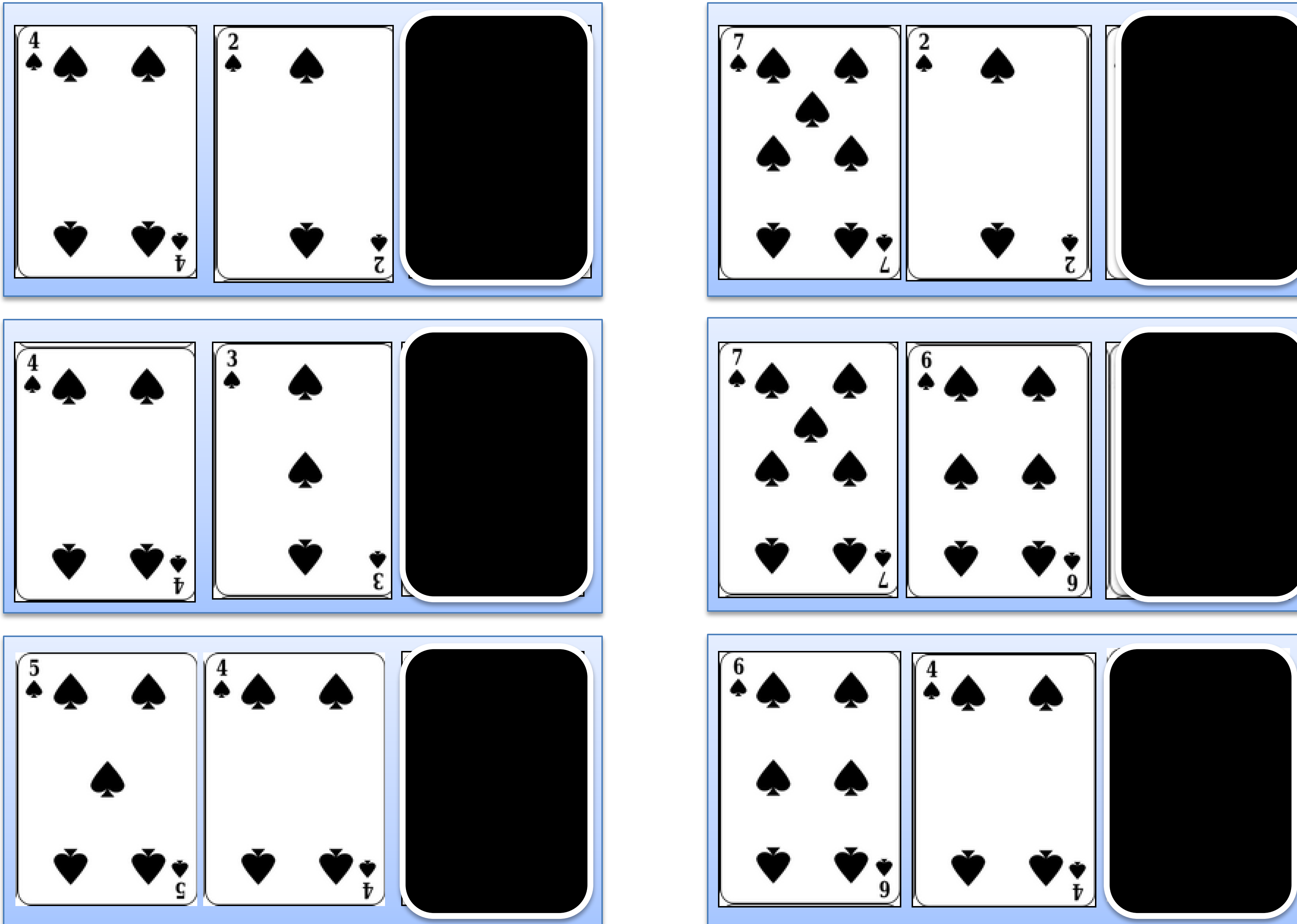
- **Optimization for Machine Learning**
  - Dual Coordinate Descent (fast convergence, moderate cost)
    - libLinear (Stochastic)
    - libSVM (Greedy)
  - Primal Methods
    - Non-smooth Loss → Stochastic Gradient Descent (slow convergence, cheap iter.)
    - Differentiable Loss → Quasi-Newton Method (very fast convergence, expensive iter.)
  - Demo

A Learning/Prediction Game

- Your team members suggest a Hypothesis Space : {h1, h2 … }
- You can only request one sample.
- Finding a hypothesis with accuracy > 50%, you earn $100,000.
  wrong hypothesis (acc <= 50%) get $100,000 punishment.



H={ h1 }, h1: (A+B) mod 13 = C



H={ h1, h2 } , h1: (A+B) mod 13 = C, h2: (A-B) mod 13 = C

Large |H| with Small |Data| Guarantees Nothing

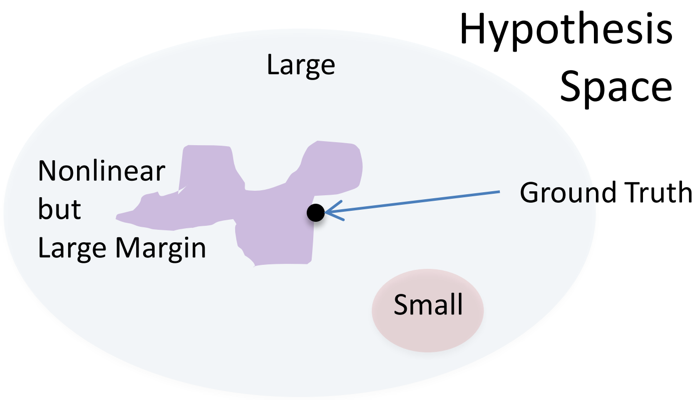
- First case: only one hypothesis h1
  - Pr { |Train_Error – Test_Error| >= 50% } <= 1/2 .

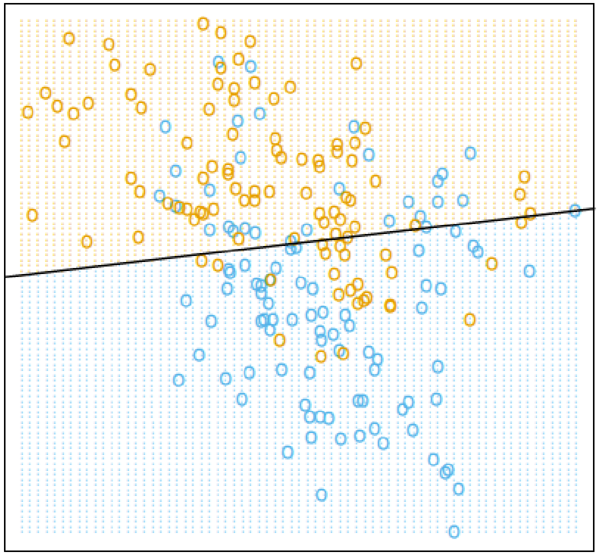
- Second case: two hypotheses h1, h2
  - Pr{ |Train_Error - Test_Error| >= 50% for h1 or h2 } <= 1/2 + 1/2 = 1.

➔ Guarantee Nothing.

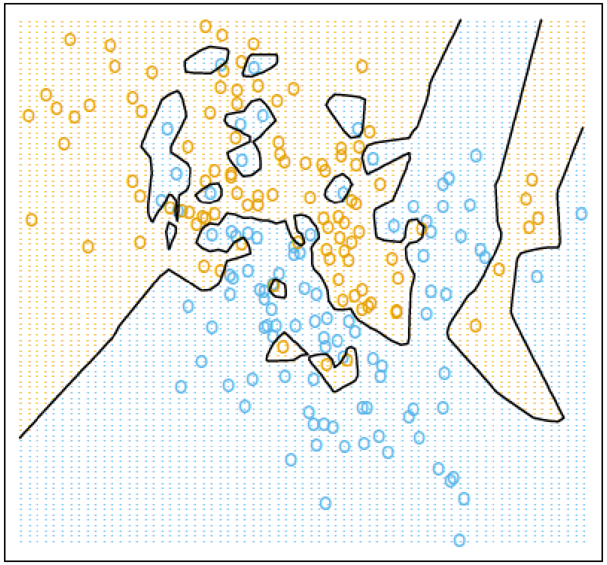
Why Support Vector Machine (SVM) ?

- Flexible Hypothesis Space. (Non-linear Kernel)
- Not to Overfit (Large-Margin)
- Sparsity (Support Vectors)
- Easy to find Global Optimum (Convex Problem)





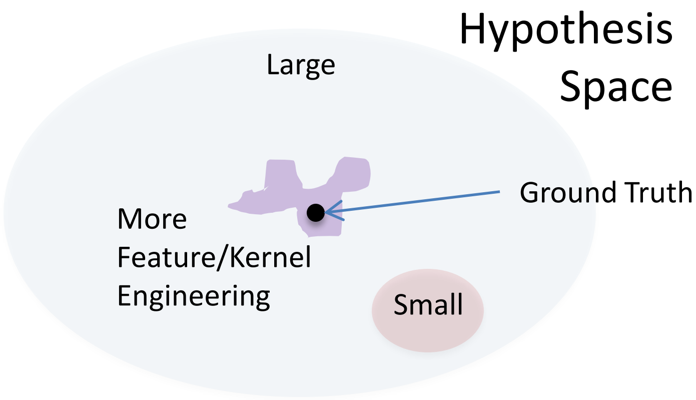
Linear Model

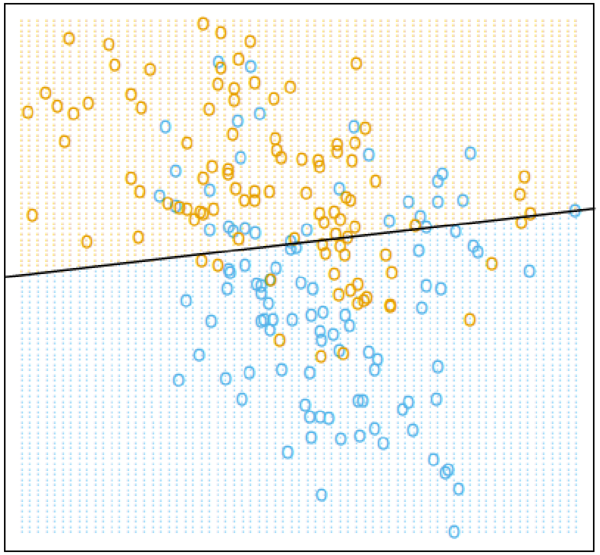


KNN

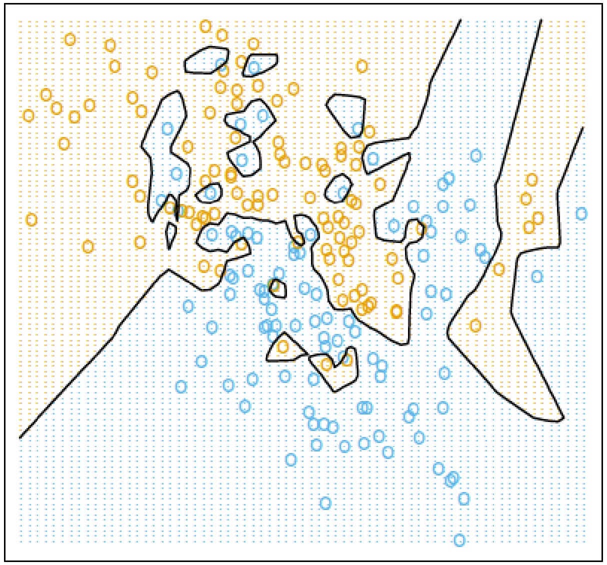
Why Support Vector Machine (SVM) ?

- Flexible Hypothesis Space. (Non-linear Kernel)
- Not to Overfit (Large-Margin)
- Sparsity (Support Vectors)
- Easy to find Global Optimum (Convex Problem)





Linear Model



KNN

SVM: Large-Margin Perceptron

$$w^* = \arg\max_{w} \left\{ \min_{n} y_n \left(\frac{w}{\|w\|}^T x_n \right) \right\}$$

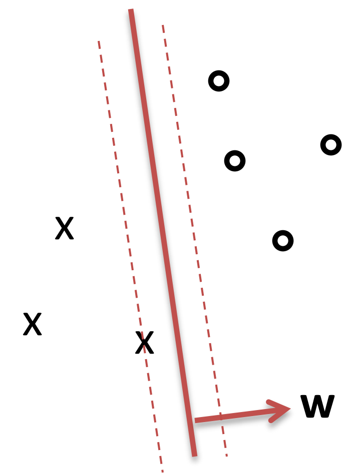
$$\max_{w,\rho} \ \rho$$

$$s.t. \ \ y_n \left(\frac{w}{\|w\|}^T x_n \right) \geq \ \rho, \ \ \forall n$$

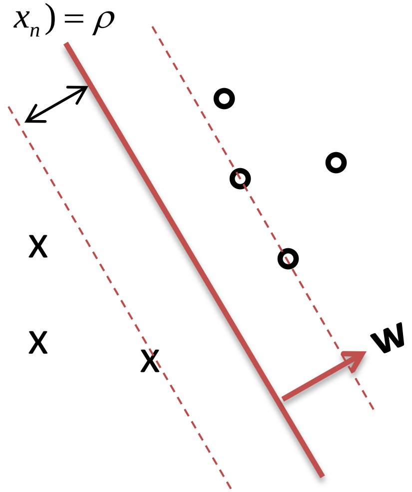
Choose $\|w\| = \frac{1}{\rho}$

$$\max_{w} \ \frac{1}{\|w\|}$$

$$s.t. \ \ y_n (w^T x_n) \geq \ 1, \ \forall n$$



$$\min_{n} \ y_n \left(\frac{w}{\|w\|}^T x_n \right) = \rho$$



$$\min_{w} \ \|w\|$$

$$s.t. \ \ y_n (w^T x_n) \geq \ 1, \ \forall n$$

$$\min_{w} \ \|w\|^2$$

$$s.t. \ \ y_n (w^T x_n) \geq \ 1, \ \forall n$$

SVM: Large-Margin Perceptron

Hard Margin

$$\min_{w} \frac{1}{2}\|w\|^2$$
$$s.t.\ y_n(w^T x_n) \geq 1,\ \forall n$$

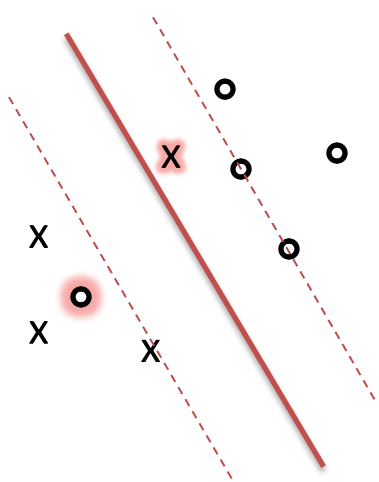
Non-Separable Problem ⟹

Soft Margin

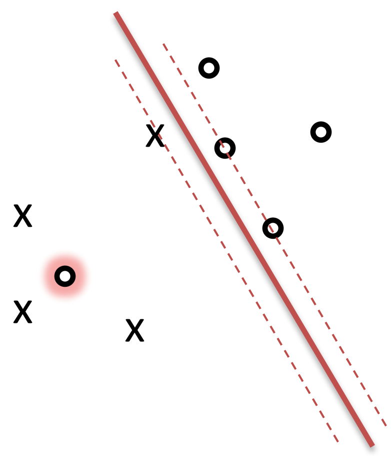
$$\min_{w,\xi\geq 0} \frac{1}{2}\|w\|^2 + C\sum_n \xi_n$$
$$s.t.\ y_n(w^T x_n) \geq 1-\xi_n,\ \forall n$$

A drawback of SVM: Solution sensitive to C

Small **C**

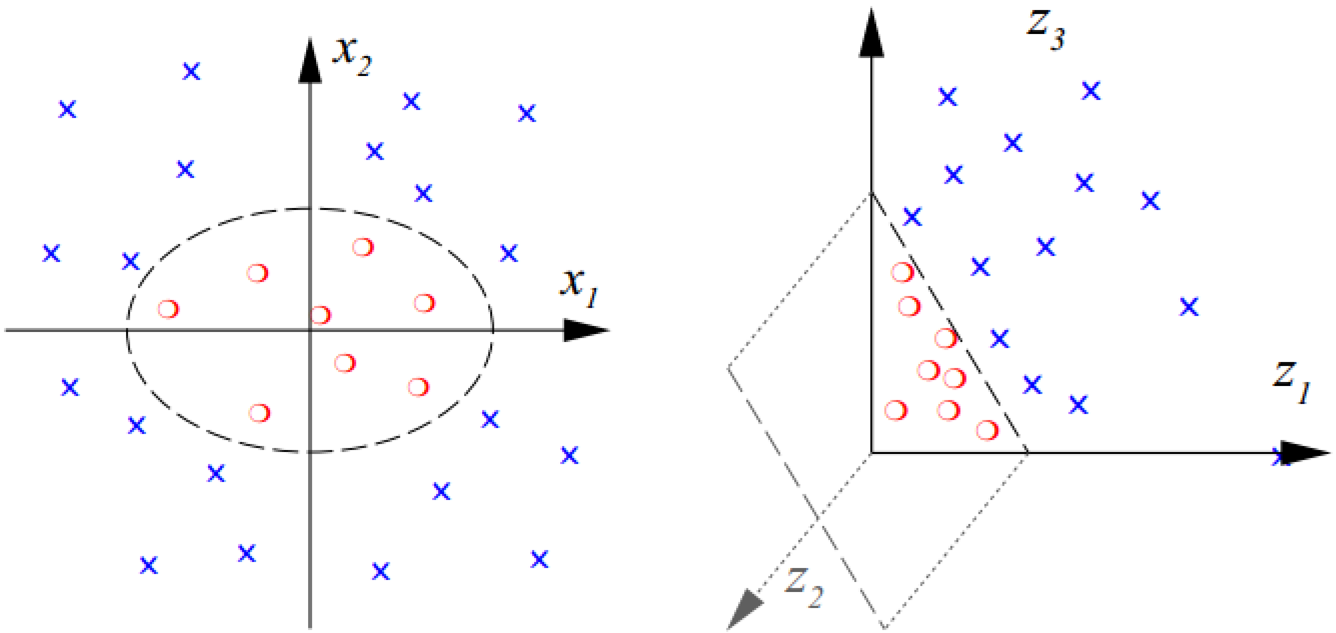


Large **C**



From Linear to Non-Linear

$$\Phi : R^2 \rightarrow R^3$$
$$(x_1, x_2) \mapsto (z_1, z_2, z_3) := (x_1^2, \sqrt{(}2)x_1 x_2, x_2^2)$$



Perceptron: a x_1 + b x_2 = 0

Ellipse: a x_1^2 + b x_2^2 + c x_1x_2 = 0 (center at origin)

From Linear to Non-Linear

Linear SVM:

$$\min_{w,\xi\geq 0} \frac{1}{2}\|w\|^2 + C\sum_n \xi_n$$
$$s.t.\ \ y_n(w^T x_n) \geq 1-\xi_n,\ \forall n$$

Non-linear SVM:

$$\min_{w,\xi\geq 0} \frac{1}{2}\|w\|^2 + C\sum_n \xi_n$$
$$s.t.\ \ y_n(w^T \phi(x_n)) \geq 1-\xi_n,\ \forall n$$

$$x = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} \rightarrow \phi(x_n) = \begin{bmatrix} x_1^2 \\ x_2^2 \\ x_3^2 \\ \sqrt{2}x_1x_2 \\ \sqrt{2}x_2x_3 \\ \sqrt{2}x_1x_3 \end{bmatrix}$$

SVM: Kernel Trick

Feature Expansion

$$x \rightarrow \phi(x)$$

$$x = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} \rightarrow \phi(x_n) = \begin{bmatrix} x_1^2 \\ x_2^2 \\ x_3^2 \\ \sqrt{2}x_1x_2 \\ \sqrt{2}x_2x_3 \\ \sqrt{2}x_1x_3 \end{bmatrix}$$

3 features ➔ 3 + $C^3{}_2$ = 6
100 features ➔ 100 + $C^{100}{}_2$ = 5050

Deg-2 Feature Expansion ➔ $O(D^2)$
Deg-K Feature Expansion ➔ $O(D^K)$

Dot Product can be computed efficiently:

Compute dot Product using $K(x,z)=(x^Tz)^2$ ⇔ deg-2 feature expansion

$$\phi(x)^T\phi(z) = \begin{bmatrix} x_1^2 \\ x_2^2 \\ x_3^2 \\ \sqrt{2}x_1x_2 \\ \sqrt{2}x_2x_3 \\ \sqrt{2}x_1x_3 \end{bmatrix}^T \begin{bmatrix} z_1^2 \\ z_2^2 \\ z_3^2 \\ \sqrt{2}z_1z_2 \\ \sqrt{2}z_2z_3 \\ \sqrt{2}z_1z_3 \end{bmatrix} = x_1^2z_1^2 + x_2^2z_2^2 + x_3^2z_3^2 + 2(x_1x_2z_1z_2 + x_2x_3z_2z_3 + x_1x_3z_1z_3) = \left(\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}^T \begin{bmatrix} z_1 \\ z_2 \\ z_3 \end{bmatrix} \right)^2 = (x^Tz)^2$$

$O(D^K)$ **$O(D)$**

SVM: Kernel Trick

Feature Expansion

$$x \rightarrow \phi(x)$$

$$\min_{w,\xi\geq 0} \frac{1}{2}\|w\|^2 + C\sum_n \xi_n$$
$$s.t.\ y_n w^T \phi(x_n) \geq 1-\xi_n,\ \forall n$$

⇕

$$\min_{w,\xi\geq 0} \frac{1}{2}\alpha^T\Phi^T\Phi\alpha + C\sum_n \xi_n$$
$$s.t.\ y_n \sum_{i=1} \alpha_i y_i \phi(x_i)^T \phi(x_n) \geq 1-\xi_n,\ \forall n$$

⇕

$$\min_{\alpha,\xi\geq 0} \frac{1}{2}\alpha^T Q\alpha + C\sum_n \xi_n$$
$$s.t.\ y_n \sum_{i=1} \alpha_i y_i K(x_i, x_n) \geq 1-\xi_n,\ \forall n$$

$$Q_{ij} = (y_i\phi(x_i))(y_j\phi(x_j)) = y_i y_j K(x_i, x_j)$$

Can we formulate the problem only using dot product $\phi(x_i)^T\phi(x_j)$?

By **Representer Theorem**, solution **w*** of the problem can be expressed as **linear combination of instances**:

$$w^* = \sum_n \alpha_n y_n \phi(x_n) = \left[y_1\phi(x_1)\ \ \dots\ \ y_N\phi(x_N)\right]_{D*N} \begin{bmatrix}\alpha_1 \\ \dots \\ \alpha_N\end{bmatrix} = \Phi\alpha$$

Prediction using only dot product $\phi(x_i)^T\phi(x_j)$:

$$w^T\phi(x_t) = \left(\sum_n \alpha_n y_n \phi(x_n)\right)^T(\phi(x_t))$$
$$= \sum_n \alpha_n y_n \phi(x_n)^T\phi(x_t) = \sum_n \alpha_n y_n K(x_n, x_t)$$

O(N*D)

or O(|Support Vector|*D)

SVM: Kernel Trick

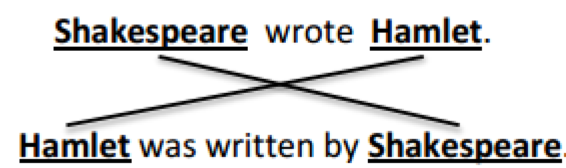
Some popular Kernels:

Polynomial Kernel: $K(x, x') = (x^T x' + 1)^d$

RBF Kernel: $K(x, x') = \exp(-\gamma \| x - x' \|^2)$

Linear Kernel: $K(x, x') = x^T x'$

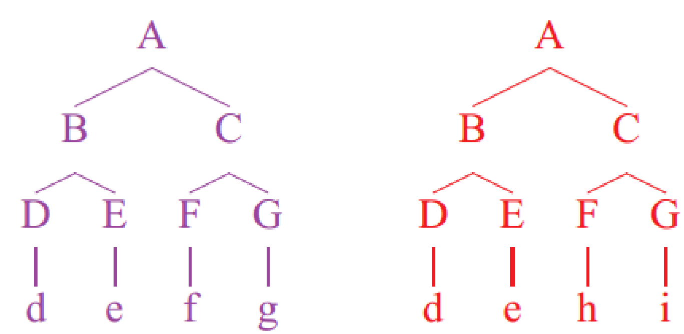
Kernels may be easier to define than Features :



String Kernel: Gene Classification / Rewriting or not

Tree Kernel: Syntactic parse tree classification

Graph Kernel: Graph Type Classification



Overview

- **Support Vector Machine**
  - The Art of Modeling --- Large Margin and Kernel Trick
  - Convex Analysis
  - Optimality Conditions
  - Duality

- **Optimization for Machine Learning**
  - Dual Coordinate Descent (fast convergence, moderate cost)
    - libLinear (Stochastic)
    - libSVM (Greedy)
  - Primal Methods
    - Non-smooth Loss → Stochastic Gradient Descent (slow convergence, cheap iter.)
    - Differentiable Loss → Quasi-Newton Method (very fast convergence, expensive iter.)

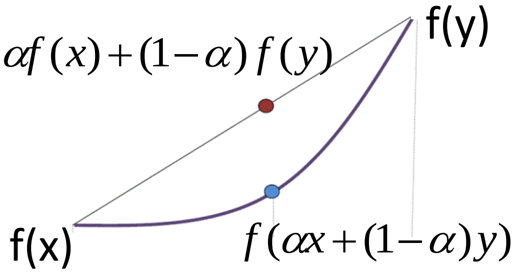
Convex Analysis

General Optimization Problem:

$$\begin{array}{ll} \text{minimize} & f_0(x) \\ \text{subject to} & f_i(x) \le b_i, \quad i = 1, \ldots, m \end{array}$$

is very difficult to solve. (very long time vs. approximate)

Optimization is much easier if the problem is **convex**, that is:

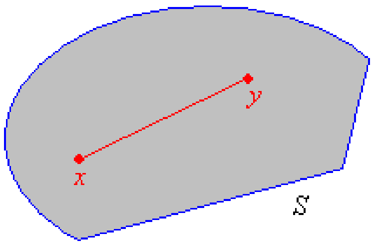


1. The **objective** function is **convex**:

$$f_0(\alpha x + (1-\alpha)y) \le \alpha f(x) + (1-\alpha) f(y) \quad \text{for} \quad 0 \le \alpha \le 1$$

2. The **feasible domain** (constrained space) is **convex**:

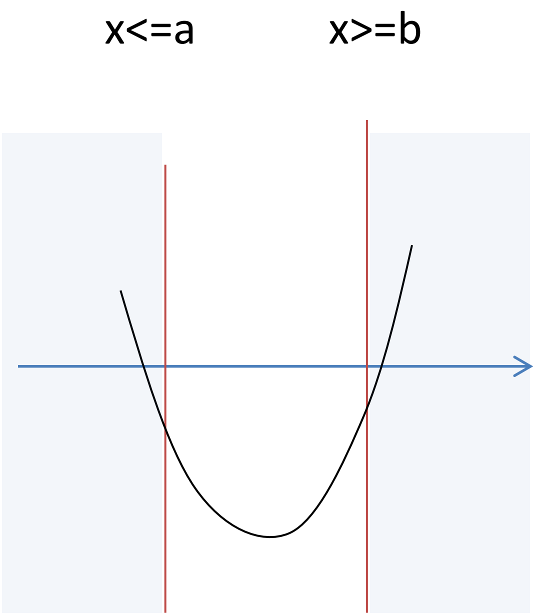
$$\text{if} \;\; x \in C, y \in C \;\; => \;\; \alpha x + (1-\alpha) y \in C, \quad 0 \le \alpha \le 1$$



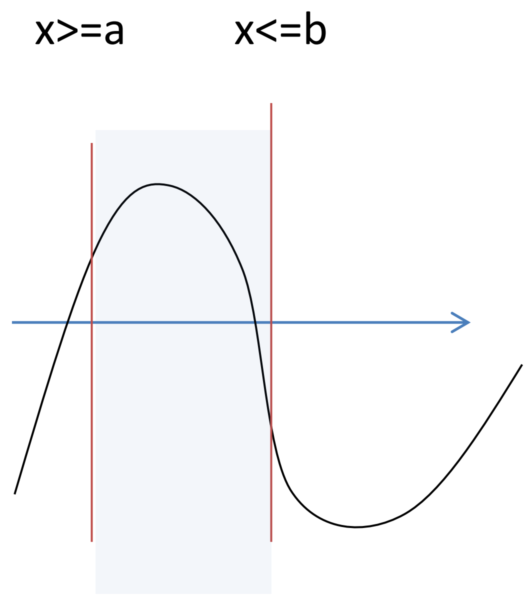
➔ **All local minimum is global minimum !!**

Convex Analysis

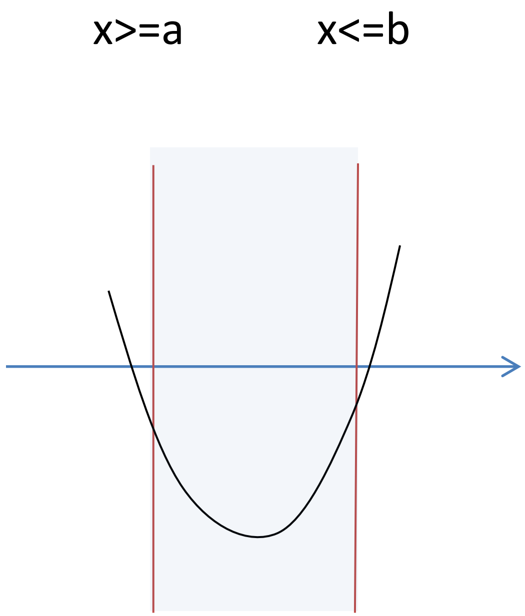
Simple Example:



Non-Convex Set ;
Convex function



Convex Set ;
Non-Convex function

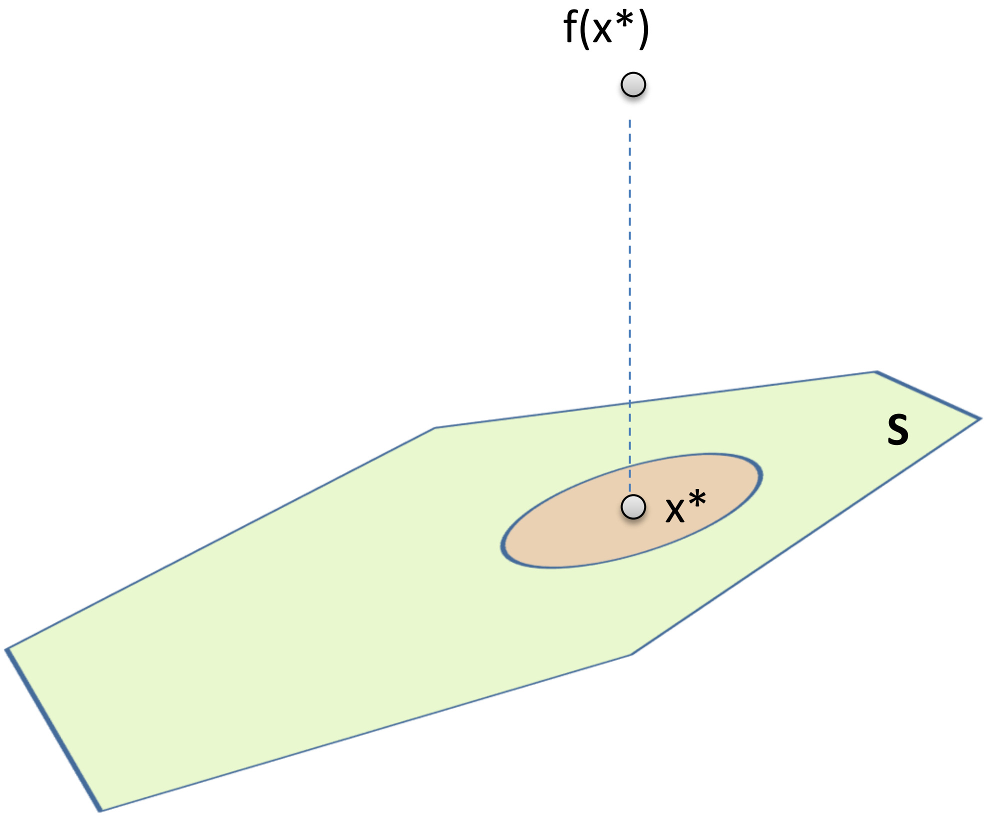


Convex Set ;
Convex function

Convex Analysis

x* is local minimum ➔ x* is global minimum (why?)

If x* is a **local minimum**, there is a "ball", in which any feasible x' has f(x') >= f(x*).

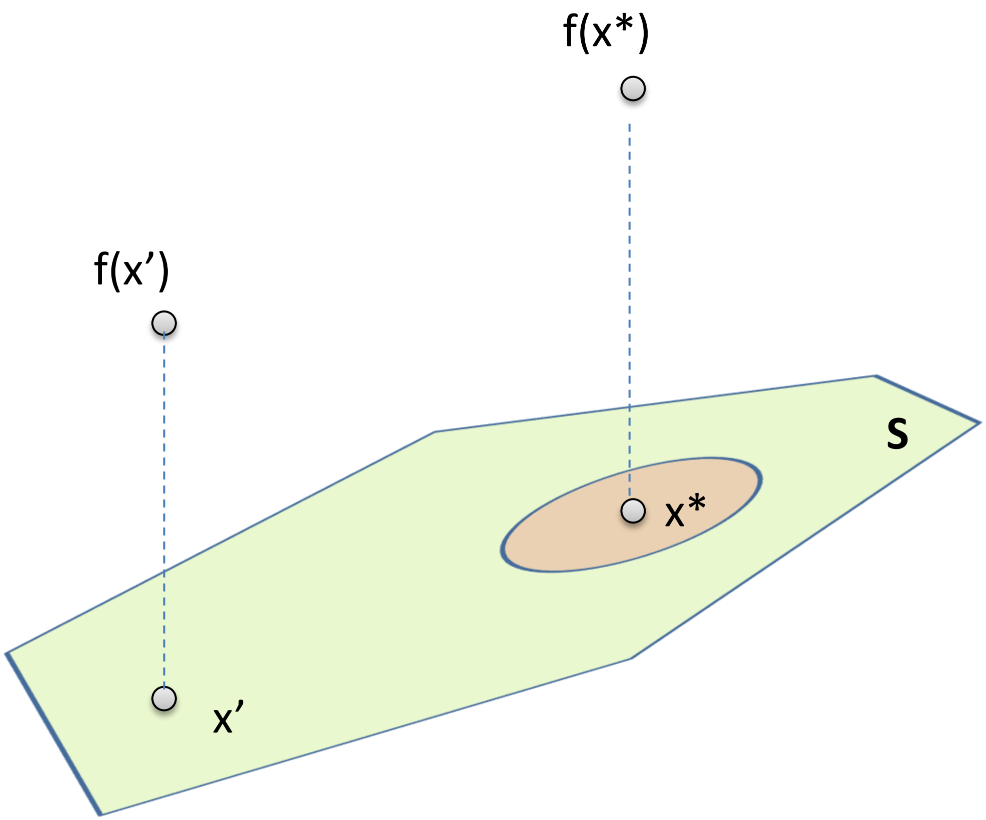


Convex Analysis

x* is local minimum ➔ x* is global minimum (why?)

If x* is a **local minimum**, there is a "ball", in which any feasible x' has f(x') >= f(x*).

Assume for contradiction that x* is **not a global minimum**. There should be a feasible x' with f(x') < f(x*).



Convex Analysis

x* is local minimum ➔ x* is global minimum (why?)

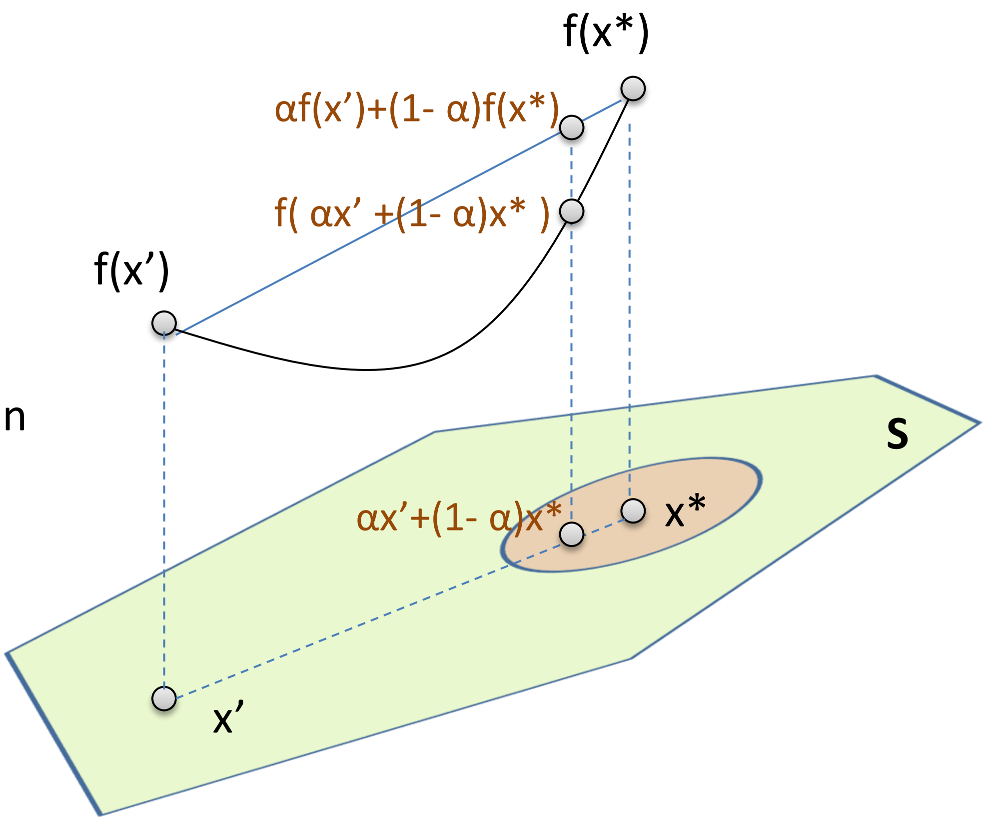
If x* is a **local minimum**, there is a "ball", in which any feasible x' has f(x') >= f(x*).

Assume for contradiction that x* is **not a global minimum**. There should be a feasible x' with f(x') < f(x*).

Then we can find a **feasible** $\alpha x' + (1-\alpha)x^*$ in the **ball** with:

$$f(\alpha x' + (1-\alpha)x^*) \;\mathbf{<=}\; \alpha f(x') + (1-\alpha) f(x^*)$$
$$\mathbf{<}\; f(x^*)$$

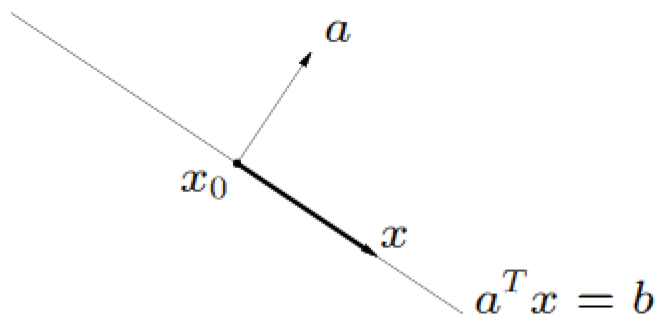
➔ contradiction.



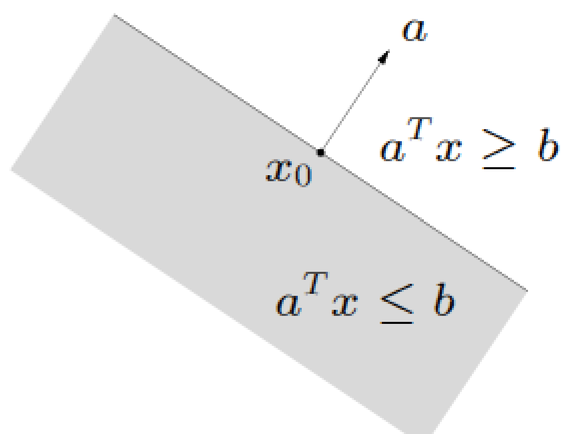
Convex Analysis

Example of Convex Set: $if\ x \in C, y \in C \ => \alpha x + (1-\alpha) y \in C, \quad 0 \le \alpha \le 1$

Linear equality constraint (Hyperplane) $\{x \mid a^T x = b\}\ (a \neq 0)$



Linear inequality constraint (Halfspace) $\{x \mid a^T x \le b\}\ (a \neq 0)$



Convex Analysis

Example of Convex Set: $if \ \ x \in C, y \in C \quad => \ \alpha x + (1-\alpha) y \in C, \quad 0 \le \alpha \le 1$

Intersection of Convex Set :

$$\begin{cases} a_1 x \le b_1 \\ a_2 x \le b_2 \\ a_3 x \le b_3 \quad (Ax \le b, \ \mathrm{Cx} = \mathrm{d}) \\ c_4 x = d_4 \\ c_5 x = d_5 \end{cases}$$

$$x, y \in A \cap B, \qquad A, B \ is \ convex$$

$$\alpha x + (1-\alpha) y \in A \cap B \ \ ?$$

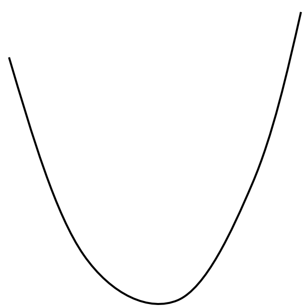
Convex Analysis

Example of Convex Function: $f_0(\alpha x + (1-\alpha)y) \le \alpha f(x) + (1-\alpha) f(y) \quad for \;\; 0 \le \alpha \le 1$

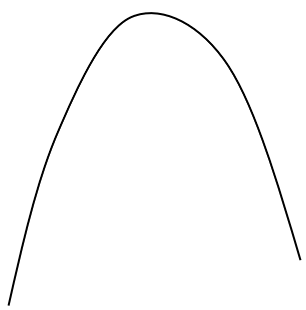
Linear Function $f(x) = c^T x$

Quadratic Function $f(x) = \frac{1}{2} x^T Q x + \mathrm{c}^{\mathrm{T}} x \quad ?$

Obviously, it depends ……



$ax^2 + bx + c, \;\; a > 0$



$ax^2 + bx + c, \;\; a < 0$

Convex Analysis

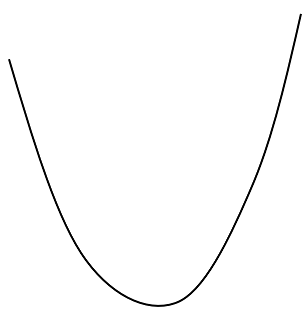
Example of Convex Function: $f_0(\alpha x + (1-\alpha)y) \le \alpha f(x) + (1-\alpha) f(y) \quad for \ \ 0 \le \alpha \le 1$

Linear Function $f(x) = c^T x$

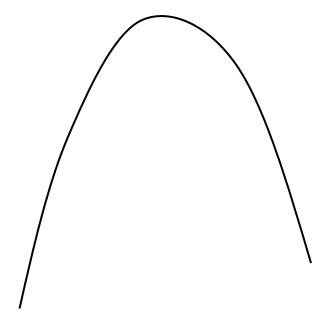
Quadratic Function $f(x) = \dfrac{1}{2} x^T Q x + \mathrm{c}^{\mathrm{T}} x \quad ?$

A practical way to check **convexity** :

Check the **second derivative** $\dfrac{\partial^2 f(x)}{\partial x^2} \ge 0 \ \ at \ \ \forall x$



$ax^2 + bx + c, \ \ a > 0$



$ax^2 + bx + c, \ \ a < 0$

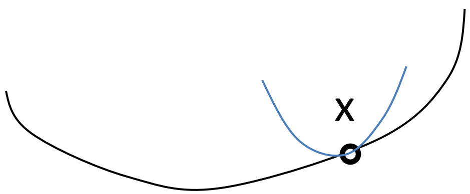
Convex Analysis

Example of Convex Function: $f_0(\alpha x+(1-\alpha)y) \le \alpha f(x)+(1-\alpha)f(y) \;\; for \;\; 0\le\alpha\le 1$

Linear Function $f(x)=c^T x$

Quadratic Function $f(x)=\frac{1}{2}x^T Qx+\mathrm{c}^{\mathrm{T}}x \quad ?$

In $\mathbf{R}^{\mathbf{D}}$, we have convexity if the **Hassian Matrix** :



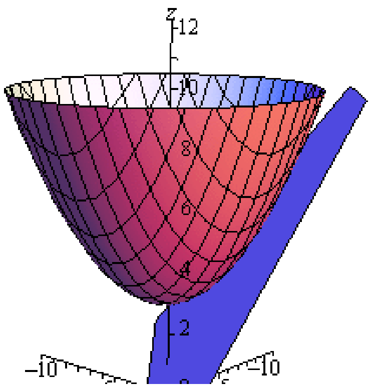
$$\frac{\partial^2 f(x)}{\partial x^2}=H(x)=\begin{bmatrix}\frac{\partial^2 f(x)}{\partial x_1^2} & \dots & \frac{\partial^2 f(x)}{\partial x_1 x_D}\\ \dots & \dots & \dots \\ \frac{\partial^2 f(x)}{\partial x_D x_1} & \dots & \frac{\partial^2 f(x)}{\partial x_D^2}\end{bmatrix}$$

is postivesemidefinte at $\forall$x

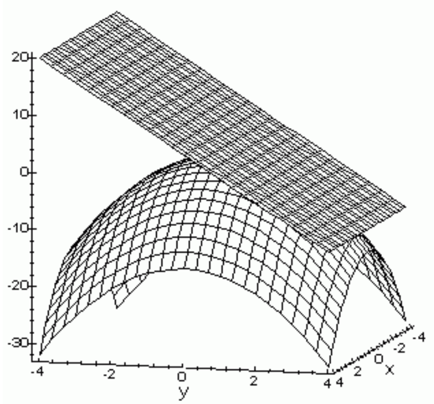
$(\;\; z^T H(x)z\ge 0 \;\; for \; \forall z \;\;)$

(all eigenvalue ≥ 0)

Convex Analysis

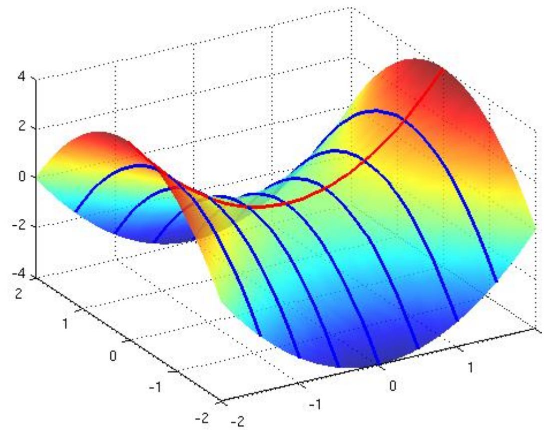


H is **positive definite**



H is **negative definite**

Other Cases ?



H is not **postive (semi-)definite**
not **negitive (semi-)definite**

In $\mathbf{R}^{\mathbf{D}}$, we have convexity if the **Hassian Matrix** :

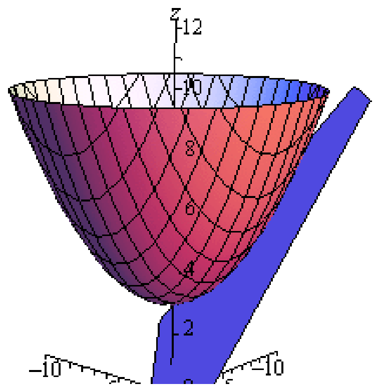
$$\frac{\partial^2 f(x)}{\partial x^2} = H(x) = \begin{bmatrix} \frac{\partial^2 f(x)}{\partial x_1^2} & \dots & \frac{\partial^2 f(x)}{\partial x_1 x_D} \\ \dots & \dots & \dots \\ \frac{\partial^2 f(x)}{\partial x_D x_1} & \dots & \frac{\partial^2 f(x)}{\partial x_D^2} \end{bmatrix}$$

is postive(semi-)definte at $\forall$x

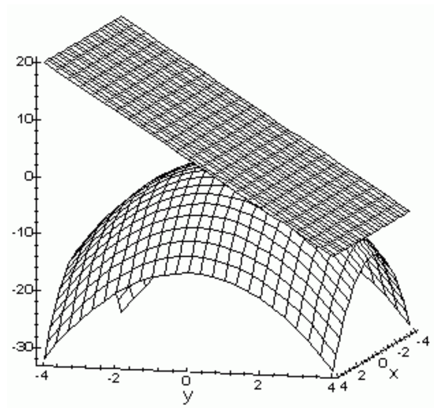
$(\; z^T H(x) z \geq 0 \; \mathit{for} \; \forall z \;)$

$(\; \text{all } e\text{igenvalue} \geq 0 \;)$

Convex Analysis

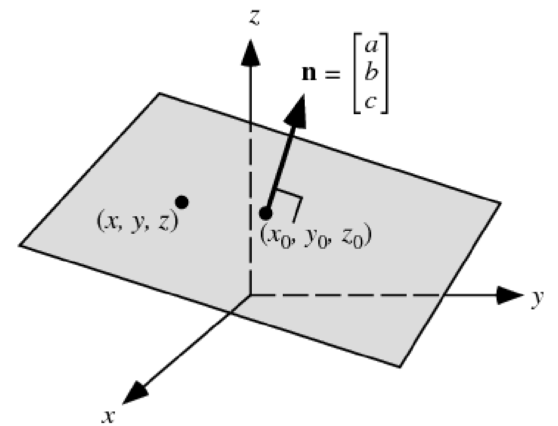


H is **positive definite**



H is **negative definite**

Other Cases ?



H=O is **postive (semi-)definite**
is **negitive (semi-)definite**

In $\mathbf{R^D}$, we have convexity if the **Hassian Matrix** :

$$\frac{\partial^2 f(x)}{\partial x^2} = H(x) = \begin{bmatrix} \frac{\partial^2 f(x)}{\partial x_1^2} & \cdots & \frac{\partial^2 f(x)}{\partial x_1 x_D} \\ \cdots & \cdots & \cdots \\ \frac{\partial^2 f(x)}{\partial x_D x_1} & \cdots & \frac{\partial^2 f(x)}{\partial x_D^2} \end{bmatrix}$$

$$is \text{ postive(semi-)definte at } \forall \text{x}$$

$$(\ z^T H(x) z \geq 0 \ \ for \ \forall z \ )$$

$$(\text{ all } e\text{igenvalue} \geq 0 \ )$$

Convex Analysis

Example of Convex Function: $f_0(\alpha x + (1-\alpha)y) \leq \alpha f(x) + (1-\alpha) f(y) \quad for \ \ 0 \leq \alpha \leq 1$

Linear Function $f(x) = c^T x \qquad (\frac{\partial^2 f(x)}{\partial x^2} = H(x) = 0)$

Quadratic Function $f(x) = \frac{1}{2} x^T Q x + \mathrm{c}^{\mathrm{T}} x \quad ? \quad (\frac{\partial^2 f(x)}{\partial x^2} = H(x) = Q)$

Quadratic Function is convex if **Q is positive semi-definite.**

$$\min_w \ \frac{1}{2}\|w\|^2$$
$$s.t. \ \ y_n w^T \phi(x_n) \ \geq \ 1, \ \forall n$$

$\frac{1}{2}\|w\|^2 = \frac{1}{2} w^T I \ w \qquad Is \ \ I \ \ positive-semidefinite\,?$

Convex Analysis

Example of Convex Function: $f_0(\alpha x+(1-\alpha)y) \le \alpha f(x)+(1-\alpha)f(y) \;\; for \;\; 0 \le \alpha \le 1$

Linear Function $f(x)=c^T x \quad (\frac{\partial^2 f(x)}{\partial x^2}=H(x)=0)$

Quadratic Function $f(x)=\frac{1}{2}x^T Q x + \mathrm{c}^{\mathrm{T}} x \quad ? \quad (\frac{\partial^2 f(x)}{\partial x^2}=H(x)=Q)$

Quadratic Function is convex if **Q is positive semi-definite.**

$$\min_{w,\,\xi\ge 0} \; \frac{1}{2}\|w\|^2 + C\sum_n \xi_n$$
$$s.t. \;\; y_n w^T \phi(x_n) \;\ge\; 1-\xi_n, \;\; \forall n$$

$$Is \;\; H(\begin{bmatrix} w \\ \xi \end{bmatrix}) = \begin{bmatrix} I_{D*D} & O_{D*N} \\ O_{D*N} & O_{N*N} \end{bmatrix} \; positive\text{-}semidefinite\,?$$

Half-space constraint

SVM problem is a convex problem.
(Quadratic Program)

Convex Analysis

Example of Convex Problem:

Linear Programming:

$$\begin{array}{ll} \text{minimize} & c^T x + d \\ \text{subject to} & Gx \preceq h \\ & Ax = b \end{array}$$

Linear objective function
s.t. **Linear** Constraint.

Quadratic Programming:

$$\begin{array}{ll} \text{minimize} & (1/2)x^T P x + q^T x + r \\ \text{subject to} & Gx \preceq h \\ & Ax = b \end{array}$$

Quadratic objective function
s.t. **Linear** Constraint.

$$where\ P\ must\ be\ positive-semidefinite$$

Overview

- **Support Vector Machine**
  - The Art of Modeling --- Large Margin and Kernel Trick
  - Convex Analysis
  - Optimality Conditions
  - Duality

- **Optimization for Machine Learning**
  - Dual Coordinate Descent (fast convergence, moderate cost)
    - libLinear (Stochastic)
    - libSVM (Greedy)
  - Primal Methods
    - Non-smooth Loss → Stochastic Gradient Descent (slow convergence, cheap iter.)
    - Differentiable Loss → Quasi-Newton Method (very fast convergence, expensive iter.)

Optimality Condition

There are many, many different **solvers** designed for different problem, but they share the same **optimality condition**.

First, we consider **Unconstrained Problem**:

$$\min_{x} f(x)$$

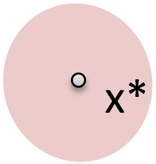
Example: Matrix Factorization (non-convex)

$$\min_{P,Q} \sum (r_{ui} - \boldsymbol{p}_u{}^T \boldsymbol{q}_i)^2 + \lambda_p \|\boldsymbol{P}\|^2 + \lambda_q \|\boldsymbol{Q}\|^2$$

Optimality Condition

There are many, many different **solvers** designed for different problem, but they share the same **optimality condition**.

First, we consider **Unconstrained Problem**:



$$\min_{x} \; f(x)$$

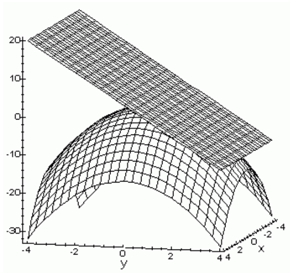
x* is Local minimizer $\leftrightarrow$ $f(x^*) \le f(x^* + p) \;\; \textit{for all } \; p \textit{ with } \; \|p\| < \varepsilon$

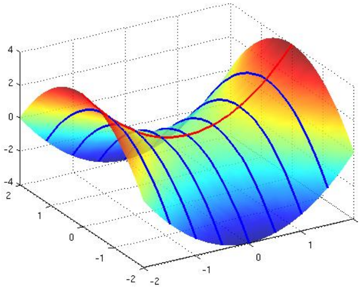
For twice-differenciable **f(x)**, consider the **Taylor Expansion:**

$$f(x^* + p) = f(x^*) + \nabla f(x^*)^T p + \frac{1}{2} p^T \nabla^2 f(x^*) p + ...$$

x* is Local minimizer $\rightarrow$ $\nabla f(x^*) = 0$

$\nabla f(x^*) = 0$ $\rightarrow$ **x*** is Local minimizer **?**

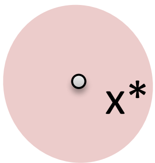




Optimality Condition

There are many, many different **solvers** designed for different problem, but they share the same **optimality condition**.

First, we consider **Unconstrained Problem**:



$$\min_{x} \ f(x)$$

x* is Local minimizer ←→ $f(x^*) \leq f(x^* + p) \ \ \textit{for all} \ \ p \ \textit{with} \ \|p\| < \varepsilon$

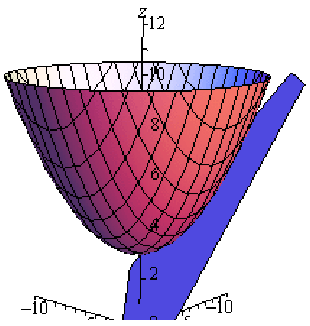
For twice-differenciable **f(x)**, consider the **Taylor Expansion:**

$$f(x^* + p) = f(x^*) + \nabla f(x^*)^T p + \frac{1}{2} p^T \nabla^2 f(x^*) p + ...$$

x* is Local minimizer ➔ $\nabla f(x^*) = 0$

$\nabla f(x^*) = 0$

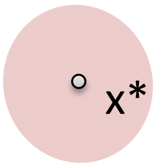
$\nabla^2 f(x^*) \ is \ positive-semidefinite$ ➔ **x*** is Local minimizer



No need to check for Convex function (why?)

Optimality Condition

There are many, many different **solvers** designed for different problem, but they share the same **optimality condition**.



First, we consider **Unconstrained Problem**:

$$\min_{x} f(x)$$

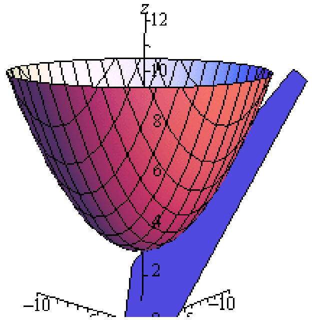
x* is Local minimizer ←→ $f(x^*) \le f(x^* + p) \;\; \textit{for all } p \textit{ with } \|p\| < \varepsilon$

For twice-differenciable **f(x)**, consider the **Taylor Expansion:**

$$f(x^* + p) = f(x^*) + \nabla f(x^*)^T p + \frac{1}{2} p^T \nabla^2 f(x^*) p + \ldots.$$

For **Convex** f(x):

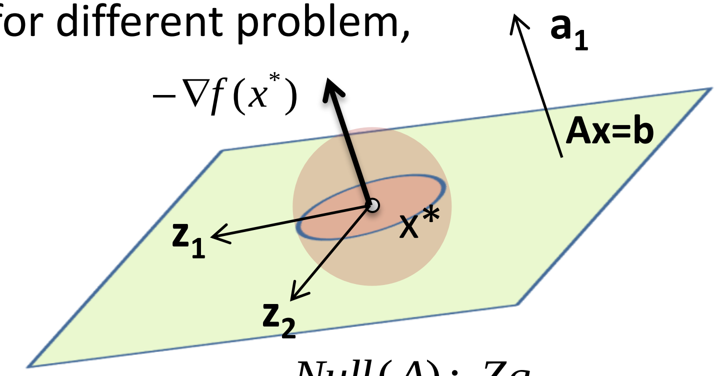
x* is Global minimizer ←→ $\nabla f(x^*) = 0$



Assume convexity for now on……

Optimality Condition

$Row(A):\ A^T\lambda$
$= \lambda_1\vec{a}_1 + \lambda_2\vec{a}_2 + \ldots + \lambda_n\vec{a}_n$



$Null(A):\ Zq,$
$where\ \ Z_{d*(d-n)} = [z_1, \ldots z_{(d-n)}]$

There are many, many different **solvers** designed for different problem, but they share the same **optimality condition**.

Now, consider **Equality Constrained Problem**:

$$\min_{x}\ f(x)$$
$$s.t.\ \ \mathrm{Ax} = \mathrm{b} \qquad (\mathrm{ex.}\ \ \mathrm{a}_1^T \mathrm{x} = \mathrm{b})$$

(nonlinear equality is, in general, not convex)

x* is Local minimizer $\leftrightarrow\ \ f(x^*) \le f(x^* + p)\ \ \textit{for all "feasible" } p \textit{ with } \|p\| < \varepsilon$

$\leftrightarrow\ \ f(x^*) \le f(x^* + Zq)\ \ \textit{for all } q \textit{ with } \|q\| < \varepsilon$

For twice-differenciable **f(x)**, consider the **Taylor Expansion:**

$$f(x^* + Z^T q) = f(x^*) + (Z^T \nabla f(x^*))^T q + \frac{1}{2} q^T Z^T \nabla^2 f(x^*) Z q + \ldots.$$

f(x) is convex, **x*** is Glocal minimizer $\leftrightarrow\ \ Z^T \nabla f(x^*) = 0$

Largrange Multipliers

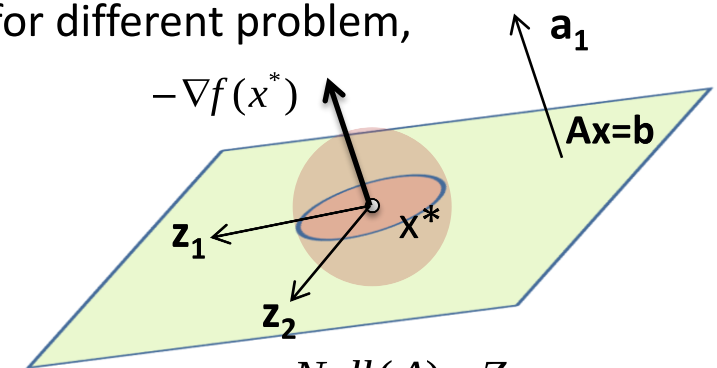
$\leftrightarrow\ \ -\nabla f(x^*) = A^T \lambda \quad (\nabla f(x^*)\ in\ Row(A))$

Optimality Condition

$Row(A):\ A^T \lambda$

$= \lambda_1 \vec{a}_1 + \lambda_2 \vec{a}_2 + \ldots + \lambda_n \vec{a}_n$

There are many, many different **solvers** designed for different problem, but they share the same **optimality condition**.



Now, consider **Equality Constrained Problem**:

$$\min_{x} \ f(x)$$

$$s.t.\ \ Ax = b \qquad (ex.\ \ a_1^T x = b)$$

(nonlinear equality is, in general, not convex)

$Null(A):\ Zq,$

$where\ \ Z_{d*(d-n)} = [z_1, \ldots z_{(d-n)}]$

x* is Local (Glocal) minimizer **↔** $Z^T \nabla f(x^*) = 0$

Largrange Multipliers

or $-\nabla f(x^*) = A^T \lambda \qquad (-\nabla f(x^*)\ in\ Row(A))$

$(ex.\ -\nabla f(x^*) = \lambda a_1)$

$\lambda_n > 0 \quad => \ ?$

$\lambda_n < 0 \quad => \ ?$

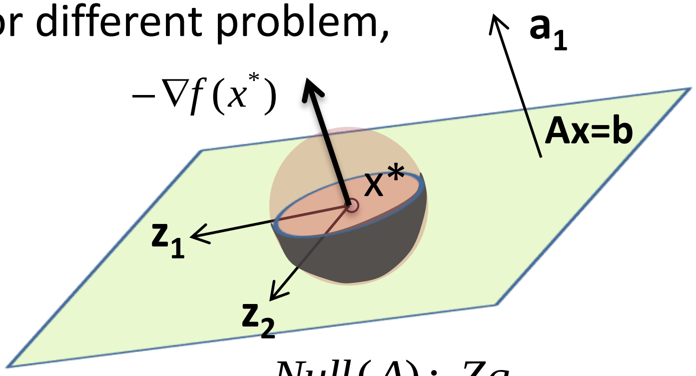
$\lambda_n = 0 \quad => \ ?$

Optimality Condition

$Row(A): \ A^T\lambda$

$= \lambda_1\vec{a}_1 + \lambda_2\vec{a}_2 + \ldots + \lambda_n\vec{a}_n$

There are many, many different **solvers** designed for different problem, but they share the same **optimality condition**.



Now, consider **Inequality Constrained Problem**:

$$\min_{x} \ f(x)$$

$$s.t. \ \ Ax \le b \qquad (\text{ex. } a_1^T x \le b)$$

(Assume linear inequality for simplicity.)

$Null(A): \ Zq,$

$where \ \ Z_{d*(d-n)} = [z_1, \ldots, z_{(d-n)}]$

Let $\mathbf{A}^*$ (some rows of A) be the coefficients of **binding constraints**:

x* is Local (Global) minimizer ←→ $-\nabla f(x^*) = A^{*T}\lambda \quad (ex. \ -\nabla f(x^*) = \lambda\vec{a}_1)$

and $\lambda \ge 0$ (feasible direction **not decrease** f(x))

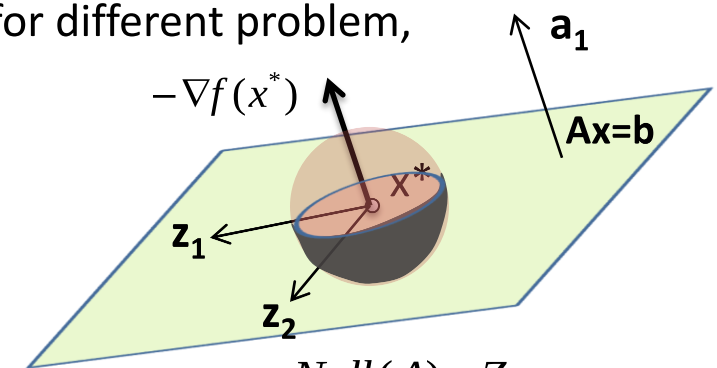
$\lambda_n < 0$ → Detach from $a_n x < b$ can decrease $f(x)$

Optimality Condition

$Row(A):\ A^T\lambda$

$= \lambda_1\vec{a}_1 + \lambda_2\vec{a}_2 + \ldots + \lambda_n\vec{a}_n$

There are many, many different **solvers** designed for different problem, but they share the same **optimality condition**.



$Null(A):\ Zq,$

$where\ \ Z_{d*(d-n)} = [z_1, \ldots z_{(d-n)}]$

Now, consider **Inequality Constrained Problem**:

$$\min_{x}\ f(x)$$

$$s.t.\ \boxed{Ax \le b} \qquad (\text{ex. } a_1^T x \le b)$$

(Assume linear inequality for simplicity.)

Require $\lambda_n = 0$ **for non-binding constraint:**

$$\boxed{\lambda_n(a_n x - b_n) = 0},\ \forall n$$

x* is Local (Global) minimizer **⟺** $\boxed{-\nabla f(x^*) = A^T\lambda}$ $(ex.\ -\nabla f(x^*) = \lambda \vec{a}_1)$

KKT conditions. and $\boxed{\lambda \ge 0}$ (feasible direction **not decrease** f(x))

Optimality Condition for SVM

What's the KKT condition for:

$$\begin{bmatrix} -\nabla_w f(w,\xi) \\ -\nabla_\xi f(w,\xi) \end{bmatrix} = \begin{bmatrix} -w \\ -C \end{bmatrix} = \begin{bmatrix} -\sum_n \alpha_n y_n \phi(x_n) \\ -\alpha_n - \beta_n \end{bmatrix}$$

$$\beta_n \rightarrow \min_{w,\xi \geq 0} \frac{1}{2}\|w\|^2 + C\sum_n \xi_n$$

$$\alpha_n \rightarrow \text{s.t. } y_n w^T \phi(x_n) \geq 1 - \xi_n, \ \forall n$$

$$-\nabla f(x^*) = A^T \lambda \rightarrow w = \sum_n \alpha_n y_n \phi(x_n), \quad C = \alpha_n + \beta_n$$

$$\lambda \geq 0 \rightarrow \alpha_n \geq 0, \quad \beta_n \geq 0$$

$$\lambda_n (a_n x - b_n) = 0 \rightarrow \alpha_n (y_n w^T \phi(x_n) - 1 + \xi_n) \geq 0, \quad \beta_n \xi_n \geq 0$$

Optimality Condition for SVM

What's the KKT condition for:

$$\beta_n \rightarrow \min_{w,\xi\geq 0} \frac{1}{2}\|w\|^2 + C\sum_n \xi_n$$

$$\alpha_n \rightarrow \text{s.t. } y_n w^T \phi(x_n) \geq 1-\xi_n, \ \forall n$$

$$-\nabla f(x^*) = A^T\lambda \ \Rightarrow \ w = \sum_n \alpha_n y_n \phi(x_n), \quad \beta_n = C - \alpha_n$$

$$\lambda \geq 0 \ \Rightarrow \ \alpha_n \geq 0, \quad (C-\alpha_n) \geq 0$$

$$\lambda_n(a_n x - b_n) = 0 \ \Rightarrow \ \alpha_n(y_n w^T \phi(x_n) - 1 + \xi_n) \geq 0, \quad (C-\alpha_n)\xi_n \geq 0$$

Optimality Condition for SVM

What's the KKT condition for:

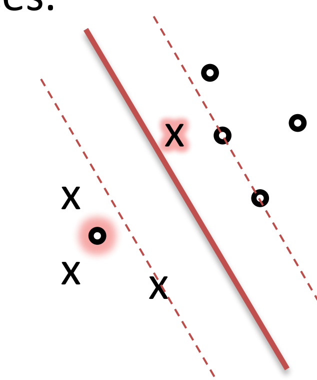
$$\begin{bmatrix} \nabla_w f(w,\xi) \\ \nabla_\xi f(w,\xi) \end{bmatrix} = \begin{bmatrix} w \\ C \end{bmatrix} = \begin{bmatrix} \sum_n \alpha_n y_n \phi(x_n) \\ \alpha_n + \beta_n \end{bmatrix}$$

$$\beta_n \rightarrow \min_{w, \xi \geq 0} \frac{1}{2}\|w\|^2 + C\sum_n \xi_n \qquad -\nabla f(x^*) = A^T \lambda \rightarrow \boxed{w = \sum_n \alpha_n y_n \phi(x_n)}$$

$$\alpha_n \rightarrow s.t. \ \boxed{y_n w^T \phi(x_n) \geq 1 - \xi_n}, \ \forall n$$

$$\lambda \geq 0 \rightarrow \boxed{0 \leq \alpha_n \leq C} \qquad \lambda_n (a_n x - b_n) = 0 \rightarrow \boxed{\begin{matrix} \alpha_n (y_n w^T \phi(x_n) - 1 + \xi_n) = 0 \\ (C - \alpha_n)\xi_n = 0 \end{matrix}}$$

1. $w = \sum_n \alpha_n y_n \phi(x_n)$ can be expressed as linear combination of instances.
2. If constraint $y_n w^T \phi(x_n) \geq 1 - \xi_n$ not binding ➔ $\alpha_n = 0$
3. If $\alpha_n > 0$ ➔ constraint is binding (**Support Vectors !**)
4. If loss of n-th instance $\xi_n > 0$ ➔ $\alpha_n = C$



Overview

- **Support Vector Machine**
  - The Art of Modeling --- Large Margin and Kernel Trick
  - Convex Analysis
  - Optimality Conditions
  - Duality

- **Optimization for Machine Learning**
  - Dual Coordinate Descent (fast convergence, moderate cost)
    - libLinear (Stochastic)
    - libSVM (Greedy)
  - **Primal Methods**
    - **Non-smooth Loss → Stochastic Gradient Descent (slow convergence, cheap iter.)**
    - **Differentiable Loss → Quasi-Newton Method (very fast convergence, expensive iter.)**

Primal SVM Problem

$$\min_{w,\,\xi\geq 0} \ \frac{1}{2}\|w\|^2 + C\sum_n \xi_n$$

$$s.t. \ \ y_n w^T \phi(x_n) \ \geq \ 1-\xi_n, \ \ \forall n$$

(Let D:#feature , N:#samples)

Quadratic Program (QP) with:

- D + N variables
- N Linear constraints
- N nonnegative constraints

➔ Intractable for median scale
(ex. N=1000, D=1000)

Primal SVM Problem
Constrained Problem ➔ Non-smooth Unconstrained

$$\min_{w,\,\xi\geq 0} \ \frac{1}{2}\|w\|^2 + C\sum_n \xi_n$$
$$s.t.\ \ y_n f(x_n) \ \geq\ 1-\xi_n,\ \ \forall n$$

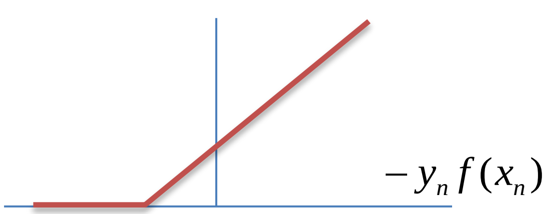
- Every constrained problem can be transformed to an Nomsmooth Unconstrained Problem

Given **w**, minimize w.r.t. ξ_n

$$\xi_n = \begin{cases} 0 & if\ \ 1-y_n f(x_n) \leq 0 \\ 1-y_n f(x_n), & otherwise \end{cases}$$

➔ $$\xi_n = \max\{\ 1-y_n f(x_n),\ 0\ \}$$

Hinge-Loss L(.)



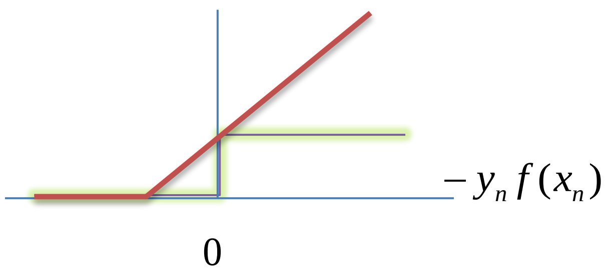
$$\min_{w} \ \frac{1}{2}\|w\|^2 + C\sum_n L(f(x_n), y_n)$$

(Nonsmooth, Unconstrained)

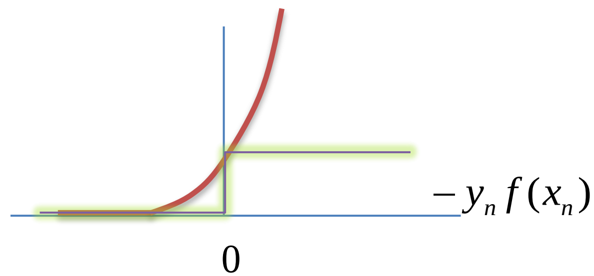
L2-Regularized Loss Minimization

$$\min_{w} \ \frac{\lambda}{2}\|w\|^2 + \sum_{n} L(f(x_n), y_n)$$

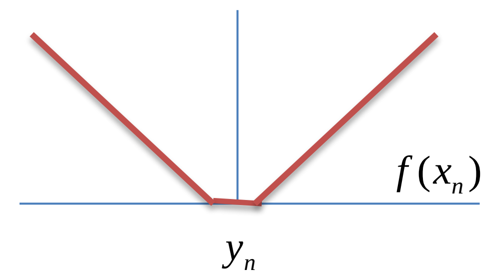
Hinge-Loss (L1-SVM, Structural SVM)



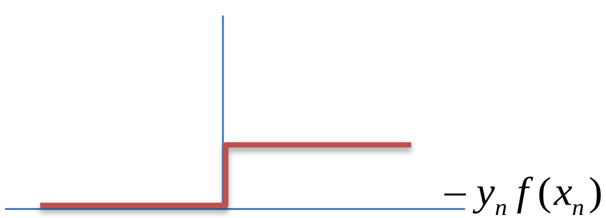
L2-SVM



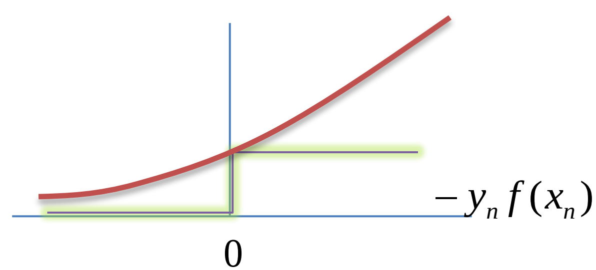
SVR



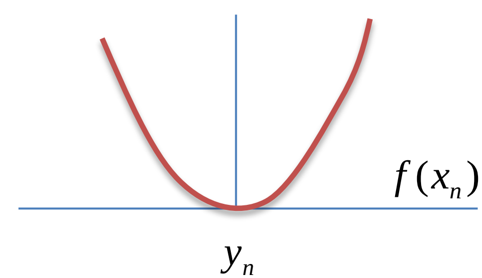
0/1-Loss (Accuracy)



Logistic Regression (CRF)



(Least-Square) Regression



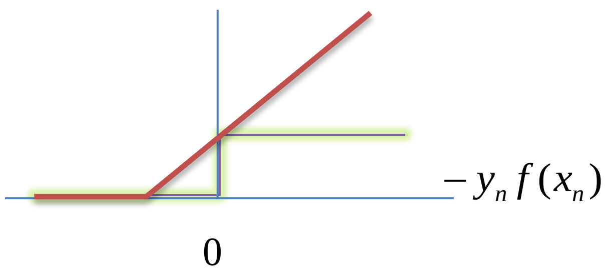
L2-Regularized Loss Minimization

$$\min_{w} \frac{\lambda}{2}\|w\|^2 + \sum_n L(f(x_n), y_n)$$

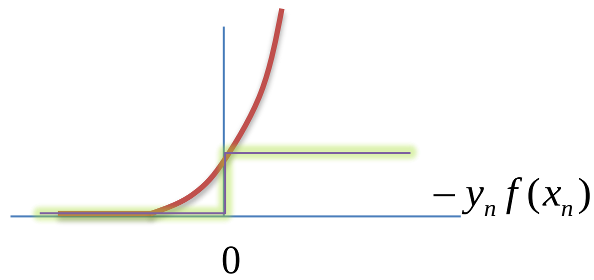
Convex Loss

- **Solve with Global Minimum**

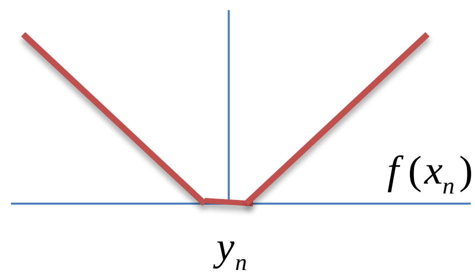
Hinge-Loss (L1-SVM, Structural SVM)



L2-SVM

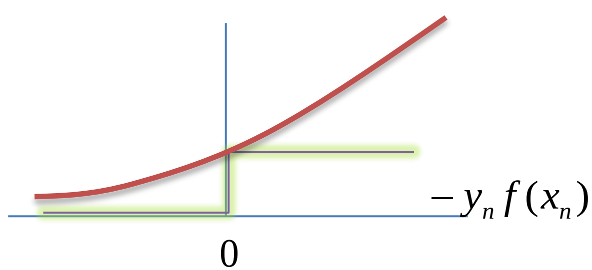


SVR

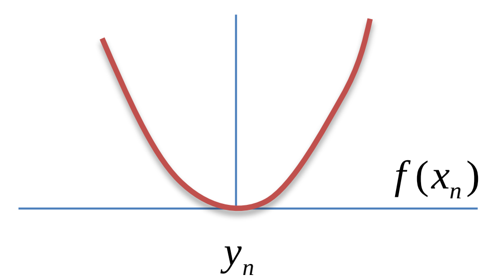


0/1-Loss (Accuracy)

Logistic Regression (CRF)



(Least-Square) Regression



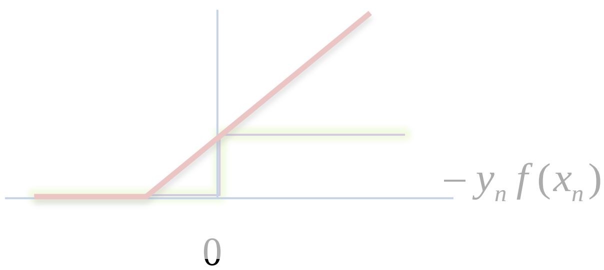
L2-Regularized Loss Minimization

$$\min_{w} \frac{\lambda}{2}\|w\|^2 + \sum_n L(f(x_n), y_n)$$

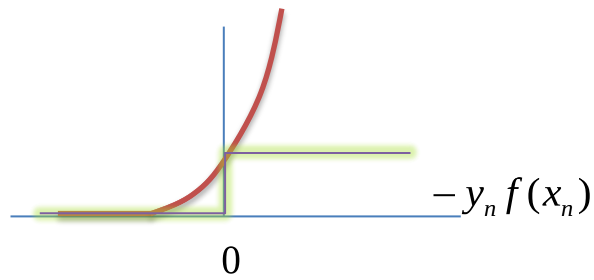
Convex Smooth Loss

- **Applicable for Second-Order Method**
- **Coordinate Descent (primal)**
- **Gradient Descent ➔ O(log(1/ε)) rate**
  Non-smooth ➔ O(1/ ε) rate

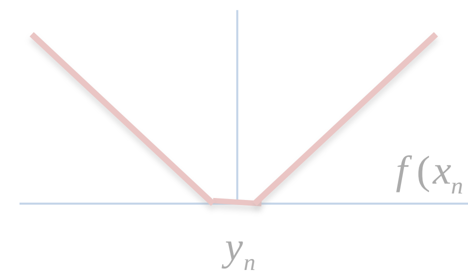
Hinge-Loss (L1-SVM, Structural SVM)



L2-SVM

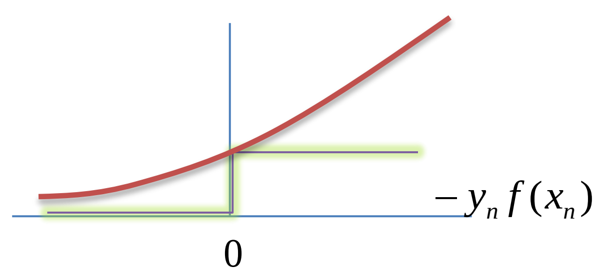


SVR

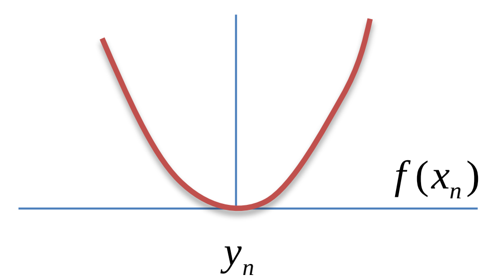


0/1-Loss (Accuracy)

Logistic Regression (CRF)



(Least-Square) Regression

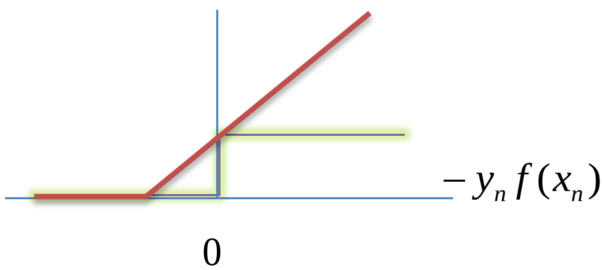


L2-Regularized Loss Minimization

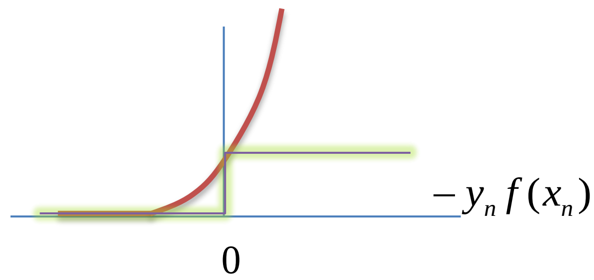
$$\min_{w} \frac{\lambda}{2}\|w\|^2 + \sum_n L(f(x_n), y_n)$$

Dual Sparsity

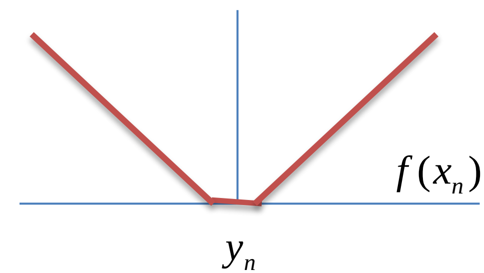
Hinge-Loss (L1-SVM, Structural SVM)



L2-SVM

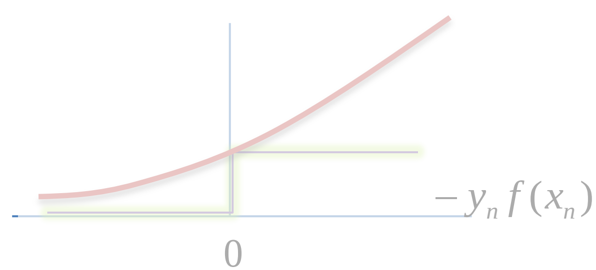


SVR

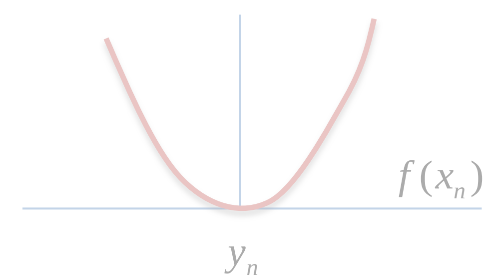


0/1-Loss (Accuracy)

Logistic Regression (CRF)



(Least-Square) Regression

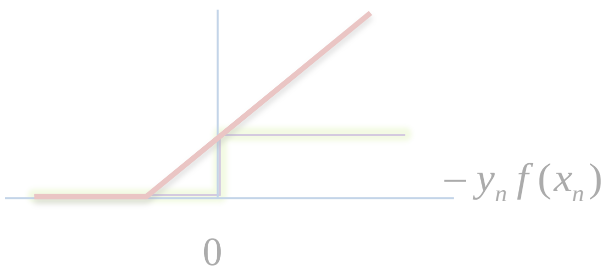


L2-Regularized Loss Minimization

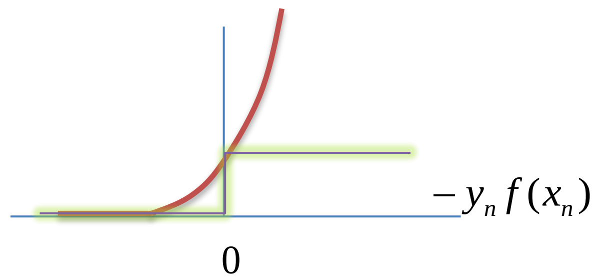
$$\min_{w} \; \frac{\lambda}{2}\|w\|^2 + \sum_{n} L(f(x_n), y_n)$$

Noise-Sensitive

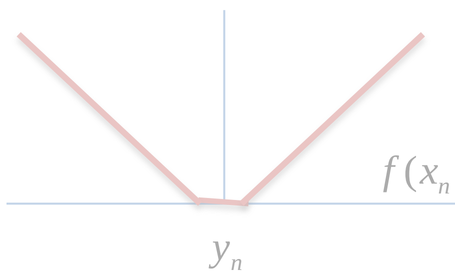
Hinge-Loss (L1-SVM, Structural SVM)



L2-SVM

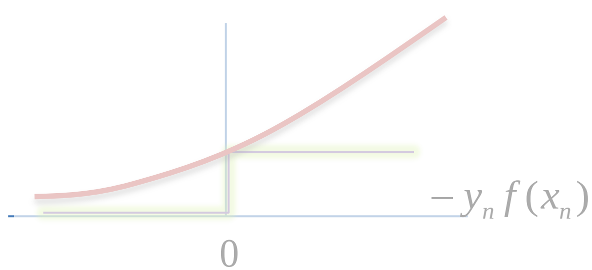


SVR

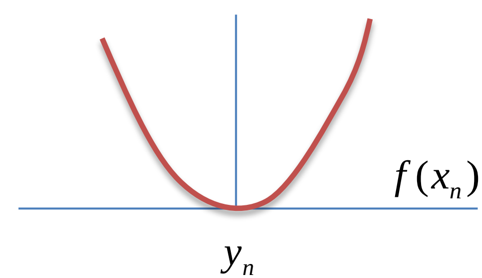


0/1-Loss (Accuracy)

Logistic Regression (CRF)



(Least-Square) Regression



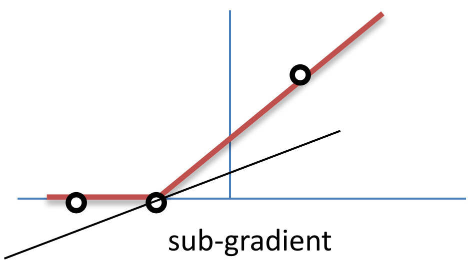
Most insensitive

Stochastic (sub-)Gradient Descent

(S. Shalev-Shwartz et al.,. ICML 2007)

$$\min_{w} \frac{\lambda}{2}\|w\|^2 + \frac{1}{N}\sum_n L(f(x_n), y_n)$$

Hinge-Loss (L1-SVM, Structural SVM)



iteration cost: O(N*D)

Algorithm: Subgradient Descent

For t = 1…T

$$w^{(t+1)} = w^{(t)} - \eta_t\left(\lambda w^{(t)} + \frac{1}{N}\sum_n L'(n)\phi_n\right)$$

End

A common choice: $\eta_t = \frac{1}{t}$

iteration cost: O(D)

Algorithm: Stochastic Subgradient Descent

For t = 1…T

Draw $\tilde{n}$ from uniformly from {1…N}

$$w^{(t+1)} = w^{(t)} - \eta_t\left(\lambda w^{(t)} + L'(\tilde{n})\phi_{\tilde{n}}\right)$$

End

$$\bar{w}^{(k)} := \frac{2}{k(k+1)} \sum_{t=1}^{k} t w^{(t)}$$

(avg. over iterations ➔ much faster)

(Shamir, O. and Zhang, ICML, 2013)

Stochastic (sub-)Gradient Descent

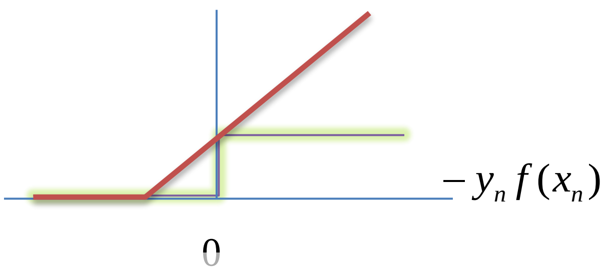
(S. Shalev-Shwartz etal.,. ICML 2007)

$$\min_{w} \ \frac{1}{2}\|w\|^2 + C\sum_n L(f(x_n), y_n)$$

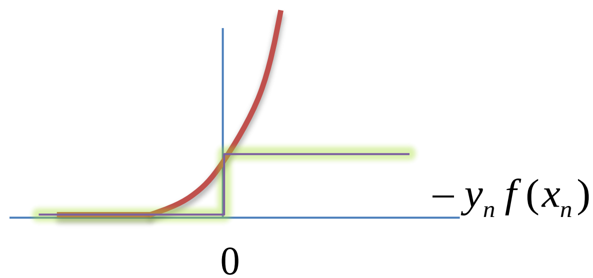
SGD

- **Applicable to all.**
- **Non-Smooth ➔ GD:O(1/ε), SGD:O(1/ε)**
- **Smooth ➔ GD:O(log1/ε), SGD:O(1/ε)**

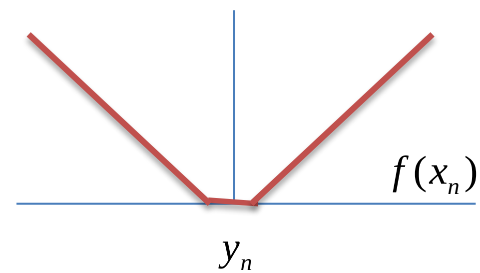
Hinge-Loss (L1-SVM, Structural SVM)



L2-SVM

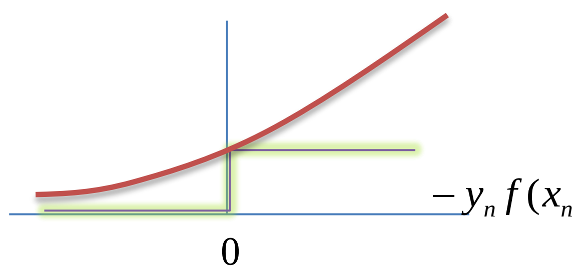


SVR

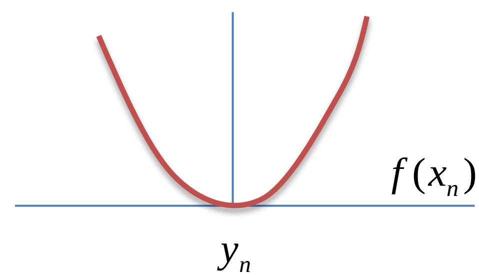


0/1-Loss (Accuracy)

Logistic Regression (CRF)

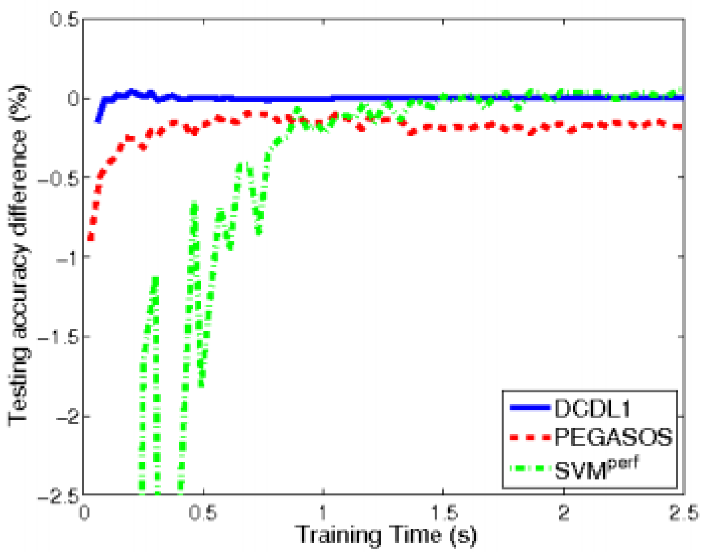


(Least-Square) Regression



SGD (Pegasos) vs. Batch Method (LibLinear)

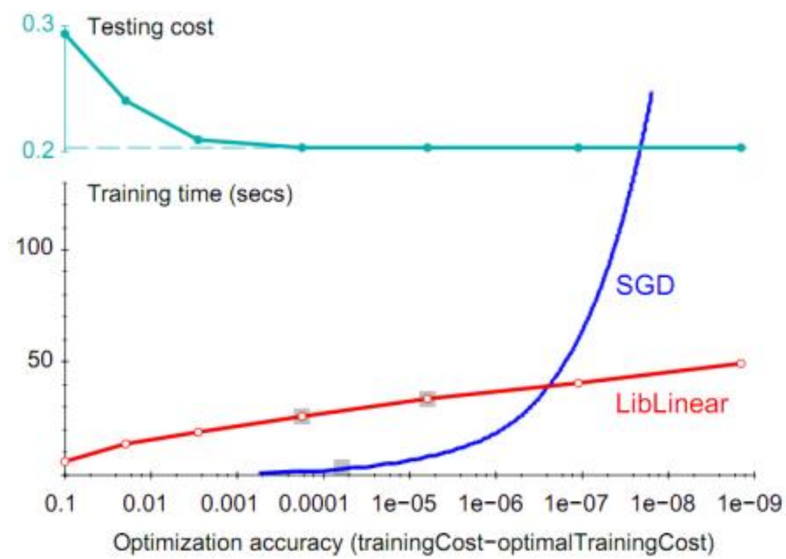
- Cons: SGD converges very slowly. (sometimes seems not convergent….)



(c) L1-SVM: real-sim

(Heish, ICML 2008) (LibLinear)

- Pros: SGD (online method) has same convergence rate for Testing and Training.



Bottou, Léon. 2007. Learning with Large Scale Datasets. *NIPS Tutorial*.

- Do you care a "**Ratio Improvement**" or "**Absolute Improvement**" in Testing ?
- What's your evaluation measure ? (AUC, Prec/Recall, Accuracy….)
- ill-conditioned problems (pos/neg ratio, Large C)

Overview

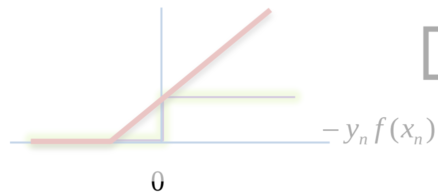
- **Support Vector Machine**
  - The Art of Modeling --- Large Margin and Kernel Trick
  - Convex Analysis
  - Optimality Conditions
  - Duality

- **Optimization for Machine Learning**
  - Dual Coordinate Descent (fast convergence, moderate cost)
    - libLinear (Stochastic)
    - libSVM (Greedy)
  - Primal Methods
    - Non-smooth Loss → Stochastic Gradient Descent (slow convergence, cheap iter.)
    - **Differentiable Loss → Quasi-Newton Method (very fast convergence, expensive iter.)**
  - L1-regularized
    - Primal Coordinate Descent

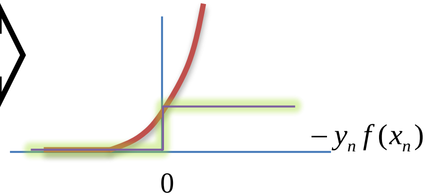
Smooth Loss vs. Non-smooth Loss

$$\min_{w} \frac{\lambda}{2}\|w\|^2 + \sum_n L(f(x_n), y_n)$$

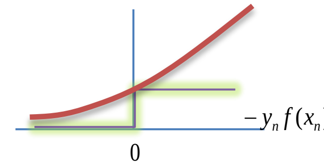
L1-SVM: $\max(1-y_n f(x_n),0)$



L2-SVM: $\max(1-y_n f(x_n), 0)^2$



Logistic Regression (CRF)



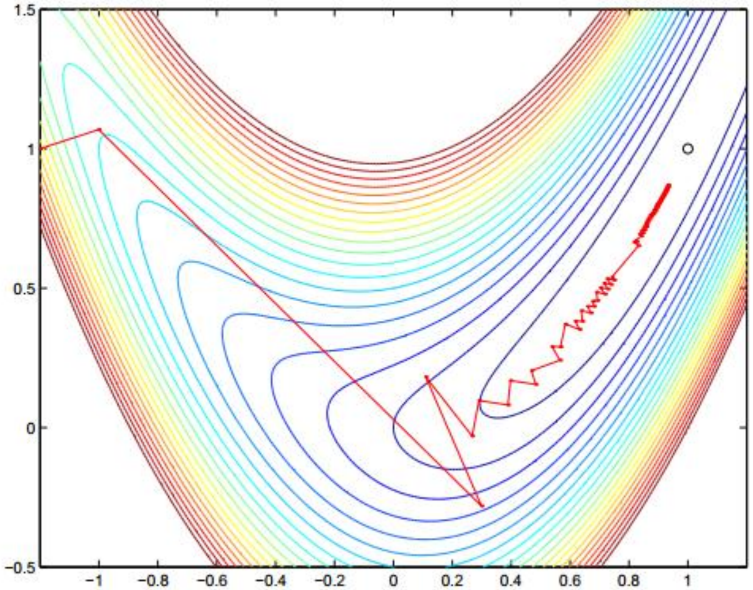
➔ Unconstrained Differentiable Problem.

```
Usage: train [options] training_set_file [model_file]
options:
-s type : set type of solver (default 1)
  for multi-class classification
         0 -- L2-regularized logistic regression (primal)
         1 -- L2-regularized L2-loss support vector classification (dual)
         2 -- L2-regularized L2-loss support vector classification (primal)
         3 -- L2-regularized L1-loss support vector classification (dual)
         4 -- support vector classification by Crammer and Singer
         5 -- L1-regularized L2-loss support vector classification
         6 -- L1-regularized logistic regression
         7 -- L2-regularized logistic regression (dual)
  for regression
        11 -- L2-regularized L2-loss support vector regression (primal)
        12 -- L2-regularized L2-loss support vector regression (dual)
        13 -- L2-regularized L1-loss support vector regression (dual)
```

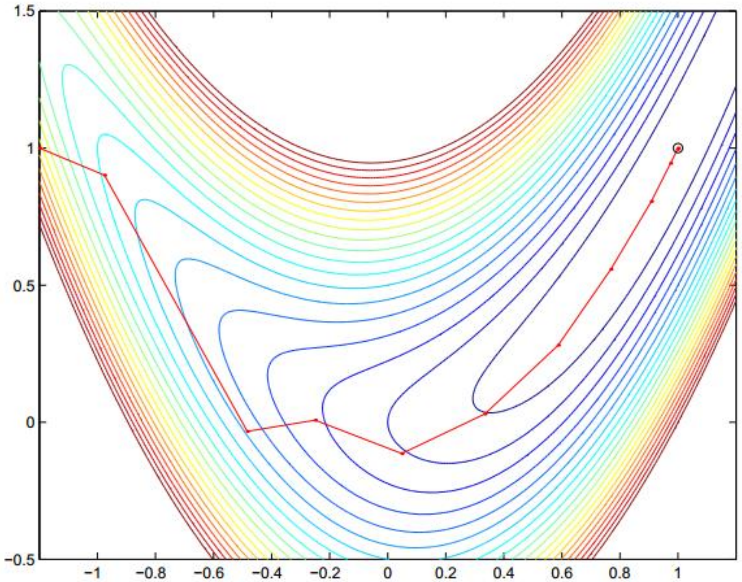
Primal Quasi-Newton Method

$$\min_{w} \ \frac{1}{2}\|w\|^2 + C\sum_n L(f(x_n), y_n)$$

- Gradient Descent (1st order) uses Linear Approximation by $\nabla \mathrm{f(w)} = \mathrm{g}$
- Newton Method (2nd order) uses Quadratic Approximation by $\nabla \mathrm{f(w)} = \mathrm{g}$ and $\nabla^2 \mathrm{f(w)} = \mathrm{H}$



Gradient Descent



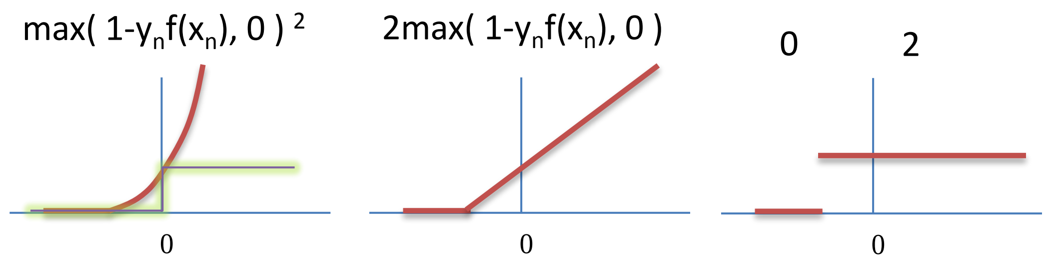
Newton Method

Primal Quasi-Newton Method

$$\min_w \; f(w) = \frac{1}{2}\|w\|^2 + C\sum_n L(w^T x_n, y_n)$$

$$g = \nabla f(w) = w + C\sum_n L'(n)\, x_n$$

$$H = \nabla^2 f(w) = I + C\sum_n L''(n)\; x_n x_n^{\mathrm{T}}$$



Quadratic Approximation at $\mathbf{w}^{(t)}$: $$\min_{s=w-w^{(t)}} \frac{1}{2}s^T H s + g^T s + f(w^{(t)})$$

Minimum at $\mathbf{s}^*$: $$Hs^* = -g$$

iteration cost: $O(N^*D^2 + D^3)$

Algorithm: Newton Method

For t = 1…T

$$Solve\;\; H^{(t)} s = -g^{(t)}$$

$$w^{(t+1)} = w^{(t)} + \eta_t s^*$$

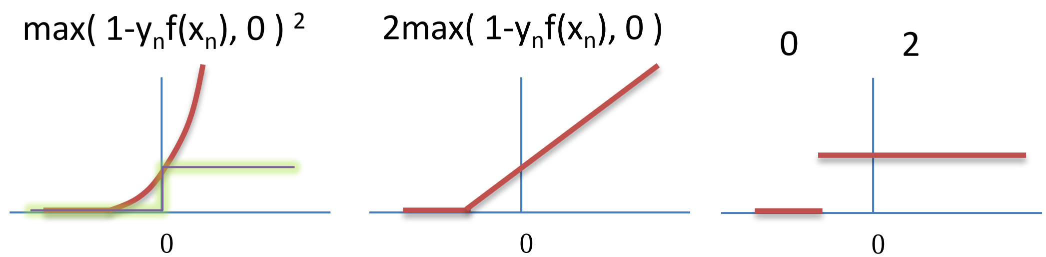
End

Primal Quasi-Newton Method

$$\min_{w} \; f(w) = \frac{1}{2}\|w\|^2 + C\sum_n L(w^T x_n, y_n)$$

$$g = \nabla f(w) = \; w + C\sum_n L'(n)\, x_n$$

$$H = \nabla^2 f(w) = \; I + C\sum_n L''(n) \;\; x_n x_n^{\mathrm{T}}$$



Quadratic Approximation at $\mathbf{w}^{(\mathbf{t})}$: $$\min_{s = w - w^{(t)}} \; \frac{1}{2} s^T H s + g^T s + f(w^{(t)})$$

Minimum at $\mathbf{s}^*$: $$Hs^* + g = 0$$

Iteration cost:

O(N*D + |SV|*D*|T_{inner}|)

Algorithm: Conjugate Gradient for $Ax = b$.

For t = 1…T_{inner}

$$r^{(t)} = b - Ax^{(t)}$$

$$d^{(t+1)} = \mathrm{d}^{(\mathrm{t})} + \eta_{\mathrm{t}} r^{(\mathrm{t})}$$

$$x^{(t+1)} = \mathrm{x}^{(\mathrm{t})} - \eta'_{\mathrm{t}} d^{(\mathrm{t})}$$

End

Algorithm: Quasi-Newton Method

For t = 1…T

$$\textit{Solve } \; H^{(t)} s = -g^{(t)} \; \textit{approximately}.$$

$$w^{(t+1)} = \mathrm{w}^{(\mathrm{t})} + \eta_{\mathrm{t}} s^*$$

End

Overview

- **Support Vector Machine**
  - The Art of Modeling --- Large Margin and Kernel Trick
  - Convex Analysis
  - Optimality Conditions
  - Duality

- **Optimization for Machine Learning**
  - Dual Coordinate Descent (fast convergence, moderate cost)
    - libLinear (Stochastic)
    - libSVM (Greedy)
  - Primal Methods
    - Non-smooth Loss → Stochastic Gradient Descent (slow convergence, cheap iter.)
    - Differentiable Loss → Quasi-Newton Method (very fast convergence, expensive iter.)

Lagrangian Duality

First, we consider the **Equality Constrained Problem:**

$$\min_{x} \; f(x)$$
$$s.t. \;\; Ax = b$$

The **optimal solution x*** is found iff:

$$Ax^* = b$$
$$-\nabla f(x^*) = A^T \lambda^*$$

If we define **Lagrangian Function** (Lagrangian) as:

$$L(x, \lambda) = f(x) + \lambda^T (Ax - b)$$

Then the **optimality condition** can be written as:

$$\frac{\partial L(x, \lambda)}{\partial \lambda} = 0 \;\; => Ax^* = b$$ (λ cannot increase L(.))

$$\frac{\partial L(x, \lambda)}{\partial x} = 0 \;\; => -\nabla f(x^*) = A^T \lambda^*$$

(x cannot decrease L(.))

$$\min_{x} \left\{ \max_{\lambda} L(x, \lambda) \right\}$$

$$\max_{\lambda} \left\{ \min_{x} L(x, \lambda) \right\}$$

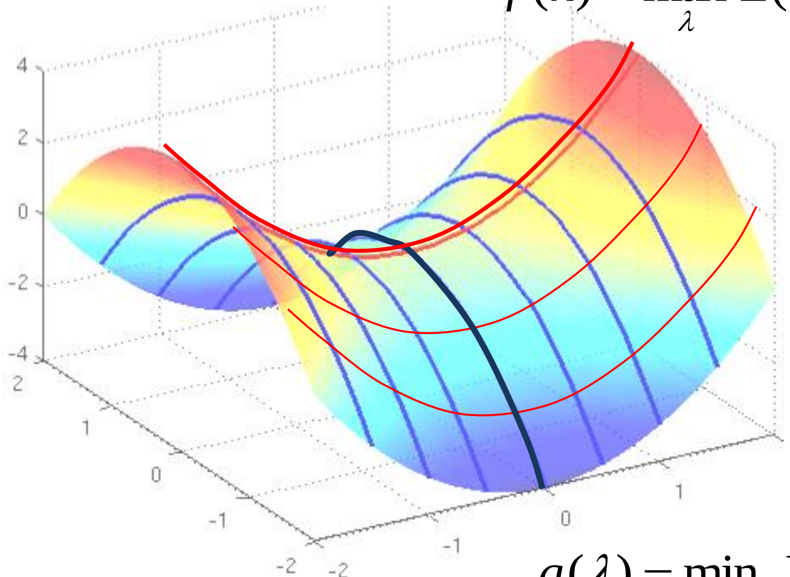
Lagrangian Duality

If we define **Lagrangian Function** (Lagrangian) as:

$$L(\mathrm{x}, \lambda) = f(x) + \lambda^T (Ax - b)$$

$L(\mathrm{x}, \lambda)$

$$f(x) = \max_{\lambda} L(x, \lambda)$$



X

λ

$$g(\lambda) = \min_{x} \ \mathrm{L}(\mathrm{x}, \lambda)$$

Every point satisfies

$$\frac{\partial \mathrm{L}(\mathrm{x}, \lambda)}{\partial \lambda} = 0 \quad => \quad \mathrm{Ax}^* = \mathrm{b}$$

Every point satisfies

$$\frac{\partial \mathrm{L}(x, \lambda)}{\partial x} = 0 \quad => \quad -\nabla f(x^*) = A^T \lambda^*$$

Lagrangian Duality

Original (Primal) problem is:

$$\min_{x} \; L(x, \lambda) = f(x) + \lambda^T (Ax - b)$$

$$s.t. \quad \frac{\partial L(x, \lambda)}{\partial \lambda} = Ax - b = 0$$

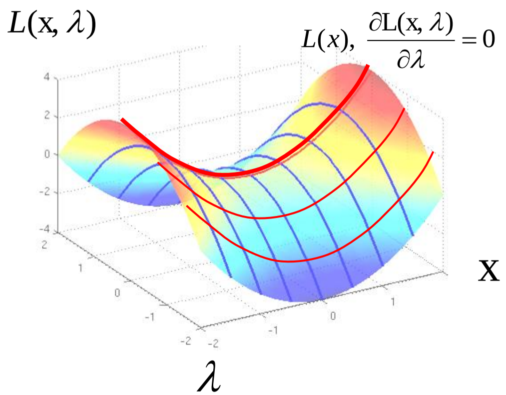
Dual Problem:

$$\max_{\lambda} \; L(x, \lambda) = f(x) + \lambda^T (Ax - b)$$

$$s.t. \quad \frac{\partial L(x, \lambda)}{\partial x} = \nabla f(x) + A^T \lambda = 0$$

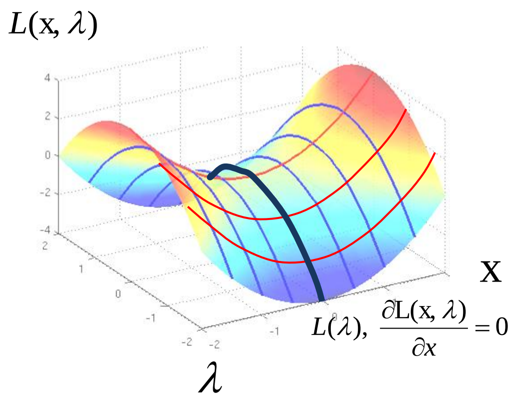
Primal problem is:

$$\min_{x} \max_{\lambda} L(x, \lambda) = f(x) + \lambda^T (Ax - b)$$



Dual problem is:

$$\max_{\lambda} \min_{x} L(x, \lambda) = f(x) + \lambda^T (Ax - b)$$



Lagrangian Duality

For **Inequality Constrained Problem:**

$$\min_{\mathrm{x}} \; f(x)$$
$$s.t. \; \mathrm{Ax} \leq \mathrm{b}$$

Primal problem is:

$$\min_{\mathrm{x}} \max_{\lambda \geq 0} L(\mathrm{x}, \lambda) = f(x) + \lambda^T (Ax - b)$$

$$\max_{\lambda \geq 0} L(\mathrm{x}, \lambda) = \begin{cases} f(x), & Ax - b \leq 0 \\ \infty, & Ax - b > 0 \end{cases}$$

Dual problem is:

$$\max_{\lambda \geq 0} \min_{x} L(\mathrm{x}, \lambda) = f(x) + \lambda^T (Ax - b)$$

$$\min_{x} L(\mathrm{x}, \lambda)$$

➔ $\nabla f(x) + A^T \lambda = 0$

SVM Dual Problem

Primal Problem:

$$\min_{w,\xi\geq 0} \frac{1}{2}\|w\|^2 + C\sum_n \xi_n \quad \Leftrightarrow \quad \min_{w,\xi} \max_{\alpha\geq 0,\beta\geq 0} \mathrm{L}(w,\xi,\alpha,\beta)$$

$$s.t.\ y_n w^T\phi(x_n) \geq 1-\xi_n,\ \forall n$$

Lagrangian:

$$\mathrm{L}(w,\xi,\alpha,\beta) = \left(\frac{1}{2}\|w\|^2 + C\sum_n \xi_n\right) - \sum_n \alpha_n \left(y_n w^T\phi(x_n) - 1 + \xi_n\right) - \sum_n \beta_n \xi_n$$

Dual Problem:

$$\max_{\alpha\geq 0,\beta\geq 0} \min_{w,\xi} \mathrm{L}(w,\xi,\alpha,\beta)$$

SVM Dual Problem

Primal Problem:

$$\min_{w,\xi\geq 0} \frac{1}{2}\|w\|^2 + C\sum_n \xi_n$$

$$s.t.\ \ y_n w^T \phi(x_n) \ \geq\ 1-\xi_n,\ \ \forall n$$

$\leftrightarrow$

$$\min_{w,\xi} \max_{\alpha\geq 0,\beta\geq 0} \mathrm{L}(w,\xi,\alpha,\beta)$$

Lagrangian:

$$\mathrm{L}(w,\xi,\alpha,\beta) = \left\{\frac{1}{2}\|w\|^2 - \sum_n \alpha_n y_n w^T \phi(x_n)\right\} + \left\{\sum_n (C-\alpha_n-\beta_n)\xi_n\right\} + \sum_n \alpha_n$$

Dual Problem:

$$\max_{\alpha\geq 0,\beta\geq 0} \min_{w,\xi} \mathrm{L}(w,\xi,\alpha,\beta)$$

$\leftrightarrow$

$$\max_{\alpha\geq 0,\beta\geq 0} \mathrm{L}(w,\xi,\alpha,\beta)$$

$$\begin{bmatrix} y_1\phi(x_1) & \ldots & y_N\phi(x_N) \end{bmatrix} \begin{bmatrix} \alpha_1 \\ \ldots \\ \alpha_N \end{bmatrix}$$

$$s.t.\ \ \frac{\partial L}{\partial w} = 0 \ \ => \ w = \sum_n \alpha_n y_n \phi(x_n) = \Phi\alpha$$

$$\frac{\partial L}{\partial \xi} = 0 \ \ => C = \alpha_n + \beta_n$$

SVM Dual Problem

Primal Problem:

$$\min_{w,\xi\geq 0} \frac{1}{2}\|w\|^2 + C\sum_n \xi_n \quad \Leftrightarrow \quad \min_{w,\xi} \max_{\alpha\geq 0,\beta\geq 0} \mathrm{L}(w,\xi,\alpha,\beta)$$

$$s.t.\ \ y_n w^T \phi(x_n) \ \geq\ 1-\xi_n,\ \forall n$$

Lagrangian:

$$\mathrm{L}(\alpha,\beta) = \left\{\frac{1}{2}\alpha^T\Phi^T\Phi\alpha - \alpha^T\Phi^T\Phi\alpha\right\} + \left\{\, 0 \,\right\} + \sum_n \alpha_n$$

Dual Problem:

$$\max_{\alpha\geq 0,\beta\geq 0} \min_{w,\xi} \mathrm{L}(w,\xi,\alpha,\beta) \quad \Leftrightarrow \quad \max_{\alpha\geq 0,\beta\geq 0} \mathrm{L}(w,\xi,\alpha,\beta)$$

$$\begin{bmatrix} y_1\phi(x_1) & \ldots & y_N\phi(x_N) \end{bmatrix} \begin{bmatrix} \alpha_1 \\ \ldots \\ \alpha_N \end{bmatrix}$$

$$s.t.\ \ \frac{\partial L}{\partial w} = 0 \ \ => \ w = \sum_n \alpha_n y_n \phi(x_n) = \Phi\alpha$$

$$\frac{\partial L}{\partial \xi} = 0 \ \ => C = \alpha_n + \beta_n$$

SVM Dual Problem

Primal Problem:

$$\min_{w,\xi\geq 0} \frac{1}{2}\|w\|^2 + C\sum_n \xi_n$$

$$s.t.\ y_n w^T \phi(x_n) \geq 1-\xi_n,\ \forall n$$

$$\leftrightarrow \quad \min_{w,\xi} \max_{\alpha\geq 0,\beta\geq 0} \mathrm{L}(w,\xi,\alpha,\beta)$$

Lagrangian:

$$\mathrm{L}(\alpha,\beta) = \left\{\frac{1}{2}\alpha^T\Phi^T\Phi\alpha - \alpha^T\Phi^T\Phi\alpha\right\} + \left\{0\right\} + \sum_n \alpha_n$$

Dual Problem:

$$\max_{\alpha\geq 0,\beta\geq 0} \min_{w,\xi} \mathrm{L}(w,\xi,\alpha,\beta) \quad \leftrightarrow \quad \max_{\alpha\geq 0,\beta\geq 0} \mathrm{L}(\alpha,\beta) = \sum_n \alpha_n - \frac{1}{2}\alpha^T\Phi^T\Phi\alpha$$

$$s.t.\ C = \alpha_n + \beta_n$$

SVM Dual Problem

Primal Problem:

$$\min_{w,\xi\geq 0} \frac{1}{2}\|w\|^2 + C\sum_n \xi_n$$

$$s.t.\ \ y_n w^T \phi(x_n) \ \geq \ 1-\xi_n,\ \ \forall n$$

$\Leftrightarrow$

$$\min_{w,\xi} \max_{\alpha\geq 0,\beta\geq 0} \mathrm{L}(w,\xi,\alpha,\beta)$$

Lagrangian:

$$\mathrm{L}(\alpha,\beta) = \left\{\frac{1}{2}\alpha^T\Phi^T\Phi\alpha - \alpha^T\Phi^T\Phi\alpha\right\} + \left\{0\right\} + \sum_n \alpha_n$$

Dual Problem:

$$\max_{\alpha\geq 0,\beta\geq 0} \min_{w,\xi} \mathrm{L}(w,\xi,\alpha,\beta)$$

$\Leftrightarrow$

$$\max_{\alpha} \mathrm{L}(\alpha,\beta) = \sum_n \alpha_n \ \text{-}\ \frac{1}{2}\alpha^T\Phi^T\Phi\alpha$$

$$s.t.\ \ 0 \leq \alpha \leq C$$

SVM Dual Problem

Dual Problem (only involve product $\phi(x_i)^T\phi(x_j)$) :

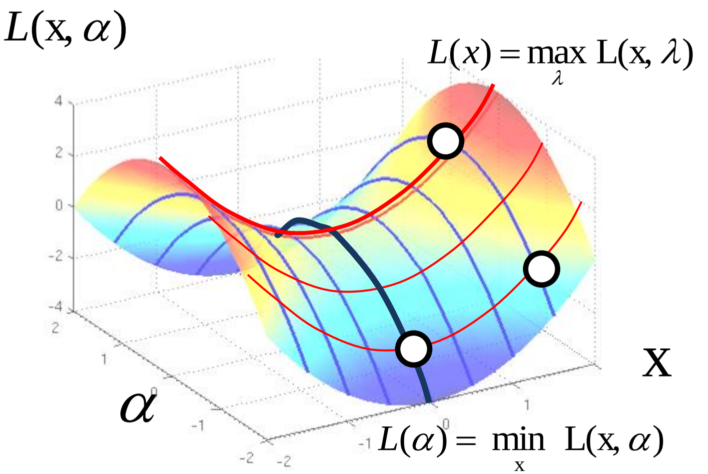
$$\max_{\alpha} \sum_n \alpha_n - \frac{1}{2}\alpha^T \Phi^T \Phi \alpha$$
$$s.t.\ 0 \le \alpha \le C$$

$\leftrightarrow$

$$\max_{\alpha} \sum_n \alpha_n - \frac{1}{2}\alpha^T Q \alpha$$
$$s.t.\ 0 \le \alpha \le C$$

$$Q_{ij} = (y_i\phi(x_i))(y_j\phi(x_j)) = y_i y_j K(x_i, x_j)$$

1. **Only "Box Constraint" ➔ Easy to solve.**
2. **dim(α) = N = |instance| , dim(w) = D = |features|**
3. **Weak Duality: Dual(α) <= Primal(w)**
4. **String Duality: Dual(α*) = Primal(w*)**
   (if primal is convex)



Overview

- **Support Vector Machine**
  - The Art of Modeling --- Large Margin and Kernel Trick
  - Convex Analysis
  - Optimality Conditions
  - Duality

- **Optimization for Machine Learning**
  - Dual Coordinate Descent (DCD) (fast convergence, moderate cost)
    - libLinear (Stochastic CD)
    - libSVM (Greedy CD)
  - Primal Methods
    - Non-smooth Loss → Stochastic Gradient Descent (slow convergence, cheap iter.)
    - Differentiable Loss → Quasi-Newton Method (very fast convergence, expensive iter.)

Dual Optimization of SVM

$$\max_{\alpha} \sum_{n} \alpha_n - \frac{1}{2}\alpha^T Q \alpha \quad \rightarrow \quad \min_{\alpha} \frac{1}{2}\alpha^T Q \alpha - \sum_{n} \alpha_n$$

$$s.t.\ 0 \le \alpha \le C \qquad\qquad s.t.\ 0 \le \alpha \le C$$

$$Q_{ij} = (y_i \phi(x_i))(y_j \phi(x_j)) = y_i y_j K(x_i, x_j)$$

```
Usage: train [options] training_set_file [model_file]
options:
-s type : set type of solver (default 1)
  for multi-class classification
         0 -- L2-regularized logistic regression (primal)
         1 -- L2-regularized L2-loss support vector classification (dual)
         2 -- L2-regularized L2-loss support vector classification (primal)
         3 -- L2-regularized L1-loss support vector classification (dual)
         4 -- support vector classification by Crammer and Singer
         5 -- L1-regularized L2-loss support vector classification
         6 -- L1-regularized logistic regression
         7 -- L2-regularized logistic regression (dual) ?!
  for regression
        11 -- L2-regularized L2-loss support vector regression (primal)
        12 -- L2-regularized L2-loss support vector regression (dual)
        13 -- L2-regularized L1-loss support vector regression (dual)
```

Constrained Minimization

$$\min_{w,\,\xi\geq 0}\ \frac{1}{2}\|w\|^2 + C\sum_n \xi_n$$

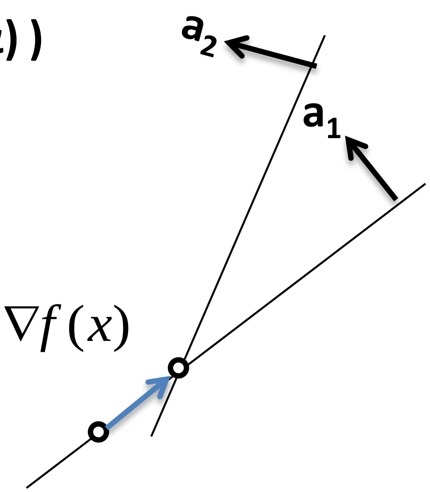
$$s.t.\ \ y_n w^T \phi(x_n)\ \geq\ 1-\xi_n,\ \forall n$$

General Constraint → Very Expensive:

1. Detecting binding constraint : **O(|constraint|*dim(α))**

2. Compute "Projected Gradient" : **O(|binding constraint|*dim(α))**

$$-\nabla^P f(x) = -Z_1^{\,T}\nabla f(x)$$



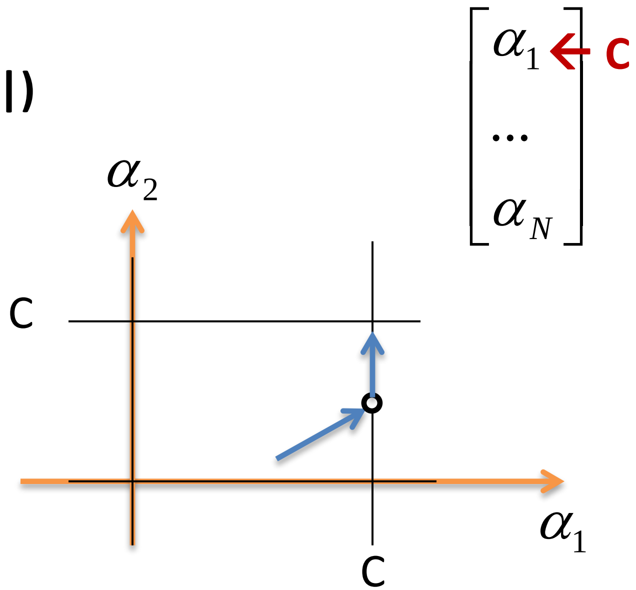
Constrained Minimization for "Box Constraint"

$$\min_{\alpha} \; \frac{1}{2}\alpha^T Q\alpha - \sum_n \alpha_n$$

$$s.t. \; \boxed{0 \le \alpha \le C}$$

Cheap:

1. Detecting binding constraint : **O(|constraint|)**
2. Compute "Projected Gradient" : **O(|binding constraint|)**



Dual Coordinate Descent

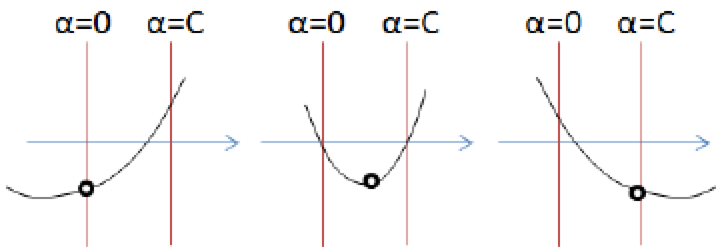
$$\min_{\alpha} \quad \frac{1}{2}\alpha^T Q\alpha - \sum_n \alpha_n$$
$$s.t. \; \boxed{0 \le \alpha \le C}$$

Minimize w.r.t. α_i

$\rightarrow$

$$\min_{\alpha_i} \quad \frac{1}{2}[\nabla^2_{ii} f(\alpha)]\alpha_i^2 + [\nabla_i f(\alpha)]\alpha_i + const.$$
$$s.t. \; 0 \le \alpha_i \le C$$

$$\nabla^2_{ii} f(\alpha) = Q_{ii}$$
$$\nabla_i f(\alpha) = [Q\alpha - 1]_i$$



$$\alpha_i \leftarrow \min\left(\max\left(\alpha_i - \frac{\nabla f(\alpha)_i}{\nabla^2 f(\alpha)_{ii}},\ 0\right), \mathrm{C}\right)$$

Dual Optimization of SVM

$$\min_{\alpha} \ \frac{1}{2}\alpha^T Q\alpha - \sum_n \alpha_n$$

$$s.t. \ 0 \le \alpha \le C$$

Even Computing Gradient is Expensive: $\nabla f(\alpha) = Q_{N*N}\ \alpha_{N*1} - 1$ **(O(N²))**

Coordinate Descent: (Optimize w.r.t. one variable at a time)

$$\nabla f(\alpha)_{(i)} = [Q]_{i,:}\ \alpha_{N*1} - 1 = \sum_k \alpha_k y_i y_k K(x_i, x_k) - 1$$ **(O(N))**

How many variables ? → As few as possible

→ Sequential Minimal Optimization (LibSVM)
(2 variable at a time)

→ Coordinate Descent (LibLinear)
(1 variable at a time)

NonLinear (LibSVM) vs. Linear (LibLinear)

Linear:

$$\nabla f(\alpha)_{(i)} = [Q]_{i,:}\, \alpha_{N*1} - 1 = \sum_k \alpha_k y_i y_k K(x_i, x_k) - 1$$

$$= \sum_k \alpha_k y_i y_k (x_i^T x_k) - 1 = y_i x_i^T (\sum_k \alpha_k y_k x_k) - 1 = y_i x_i^T w - 1$$

O(|non-zero Feature|)

Non-Linear:

$$\nabla f(\alpha)_{(i)} = [Q]_{i,:}\, \alpha_{N*1} - 1 = \sum_k \alpha_k y_i y_k K(x_i, x_k) - 1$$ **O(|Instances|*|non-zero Features|)**

NonLinear (LibSVM) vs. Linear (LibLinear)

Linear:

$$\nabla f(\alpha)_{(i)} = y_i x_i^T w - 1$$

O(|Feature|)

α_1 … α_N → $w = \sum_n \alpha_n y_n x_n$ → $\nabla f(\alpha)_{(i)} = y_i x_i^T w - 1$

(Cheap Update → Random Select Coordinate)

Non-Linear:

$$\nabla f(\alpha)_{(i)} = \sum_k \alpha_k y_i y_k K(x_i, x_k) - 1$$

O(|Instances|)

α_1 … α_N → $\nabla f(\alpha)_{(i)}$

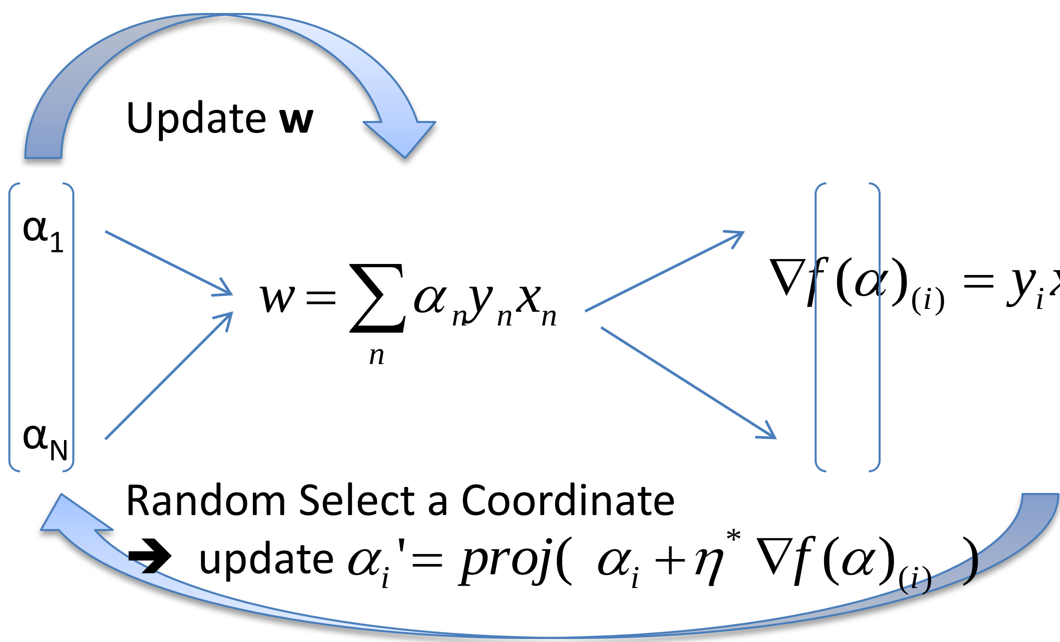
(Expensive Update → Select most Promising Coordinate)

LibLinear

Linear:

$$\nabla f(\alpha)_{(i)} = y_i x_i^T w - 1$$

O(|Feature|)



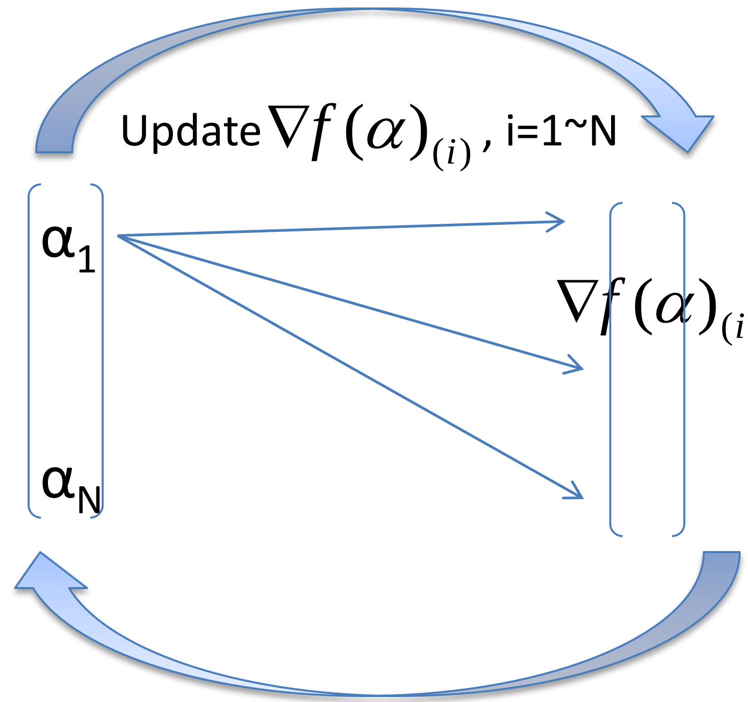
LibSVM

Non-Linear:

$$\nabla f(\alpha)_{(i)} = \sum_k \alpha_k y_i y_k K(x_i, x_k) - 1$$

O(|Instances|*|Features|)
(no cache)

O(|Instances|)
(cache)



Choose 2 **most Promising** Coordinates
➔ update $\alpha_i' = proj(\ \alpha_i + \eta^* \nabla f(\alpha)_{(i)}\ )$

Demo: libSVM, libLinear

- Normalize Features:
  - svm-scale -s [range_file] [train] > train.scale
  - svm-scale -r [range_file] [test] > test.scale

- Training:
  - LibSVM: svm-train [train.scale] (produce train.scale.model)
  - LibSVM: svm-predict [test.scale] [train.scale.model] [pred_output]

  - LibLinear: train [train.scale] (produce train.scale.model)
  - LibLinear: predict [test.scale] [train.scale.model] [pred_output]