CSE 417T
Introduction to Machine Learning

Lecture 18

Instructor: Chien-Ju (CJ) Ho

Logistics

- Homework 4 is due November 14 (Monday)

- Keep track of your own late days
  - Gradescope doesn't allow separate deadlines
  - Your submissions won't be graded if you exceed the late-day limit

- Homework 5 will be announced early next week
  - This will be our last assignment

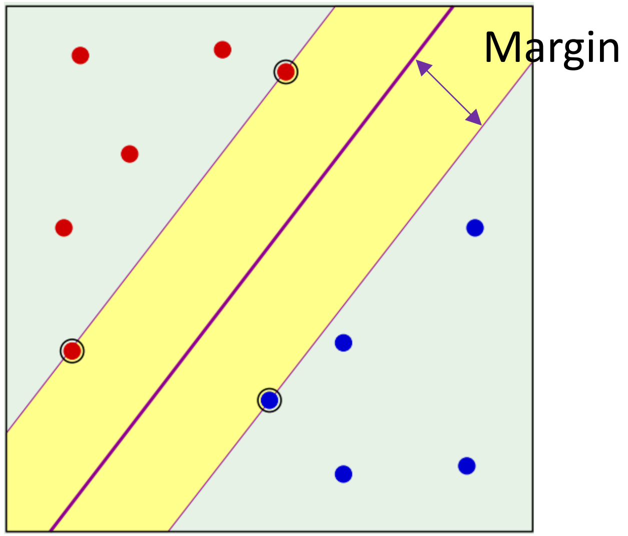
Recap

Support Vector Machine

- Goal: Find the max-margin linear separator that separates the data

- Hard-Margin SVM (Assume data is linearly separable)

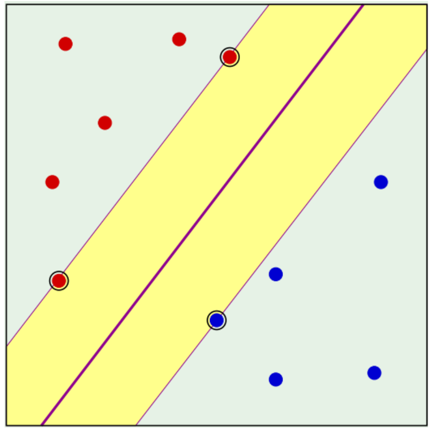
$$\text{minimize}_{\vec{w},b} \quad \frac{1}{2}\vec{w}^T\vec{w}$$
$$\text{subject to} \quad y_n(\vec{w}^T\vec{x}_n + b) \geq 1, \forall n$$



- Solvable using Quadratic Program (QP)
- Given solution $(\vec{w}^*, b^*)$, the learned hypothesis $g(\vec{x}) = sign({\vec{w}^*}^T\vec{x} + b^*)$

Support Vectors

- We call the points closest to the separator (candidate) support vectors
  - Since they support the separator

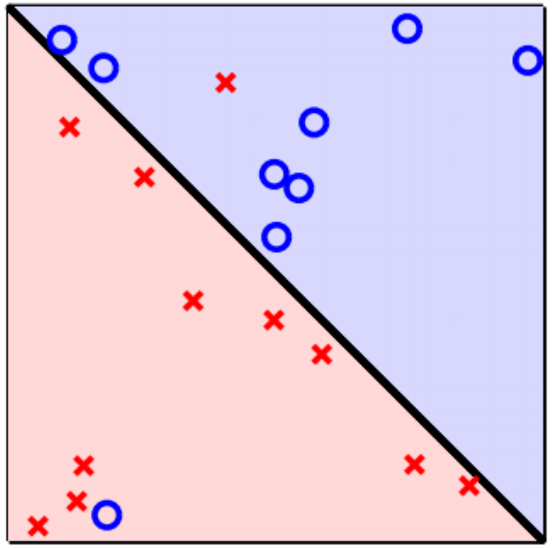


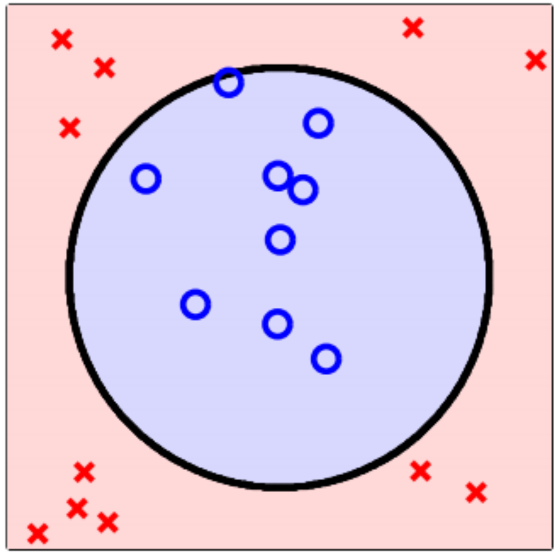
- What are the properties of (candidate) support vectors?
  - They are the points that the equality holds in the constraints
    - If $\vec{x}_n$ is a support vector, $y_n(\vec{w}^T \vec{x}_n + b) = 1$
  - Removing the non-support vectors will not impact the linear separator

- Leave-One-Out Cross-Validation (LOOCV) error for SVM?
  - $E_{LOOCV} \leq \frac{\#\,\text{support vectors}}{N}$ (an upper bound, could be smaller)

Non-Separable Data

- Two scenarios

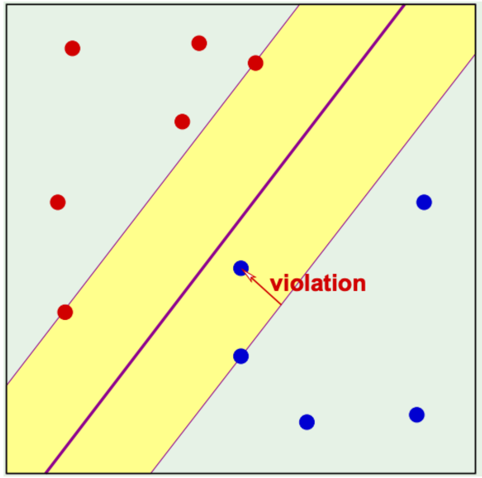




- Tolerate some noise
  - Soft-Margin SVM
- Nonlinear transform
  - Dual formulation and kernel tricks

Soft-Margin SVM

- For each point $(\vec{x}_n, y_n)$, we allow a violation $\xi_n \geq 0$
  - The constraint becomes: $y_n(\vec{w}^T\vec{x}_n + b) \geq 1 - \xi_n$
  - We add a penalty for each violation : Total penalty $C\sum_{n=1}^{N}\xi_n$



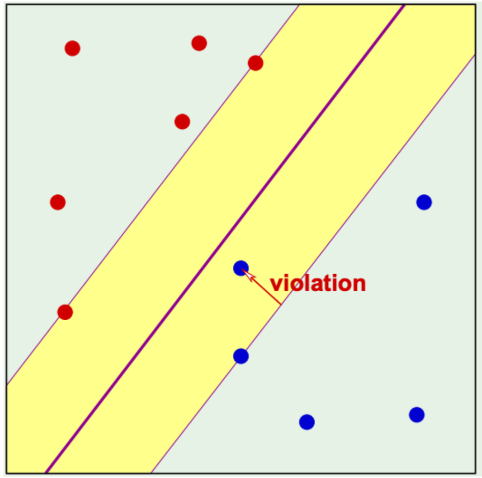
$$
\begin{aligned}
&\text{minimize}_{\vec{w},b,\vec{\xi}} && \frac{1}{2}\vec{w}^T\vec{w} + C\sum_{n=1}^{N}\xi_n \\
&\text{subject to} && y_n(\vec{w}^T\vec{x}_n + b) \geq 1 - \xi_n, \forall n \\
& && \xi_n \geq 0, \forall n
\end{aligned}
$$

Remarks:

- C is a hyper-parameter we can choose, e.g., using validation
  - Larger C => less tolerable to noise => smaller margin
- Soft-margin SVM is still a Quadratic Program, with efficient solvers

Soft-Margin SVM

- For each point $(\vec{x}_n, y_n)$, we allow a violation $\xi_n \geq 0$
  - The constraint becomes: $y_n(\vec{w}^T\vec{x}_n + b) \geq 1 - \xi_n$
  - We add a penalty for each violation : Total penalty $C\sum_{n=1}^{N}\xi_n$

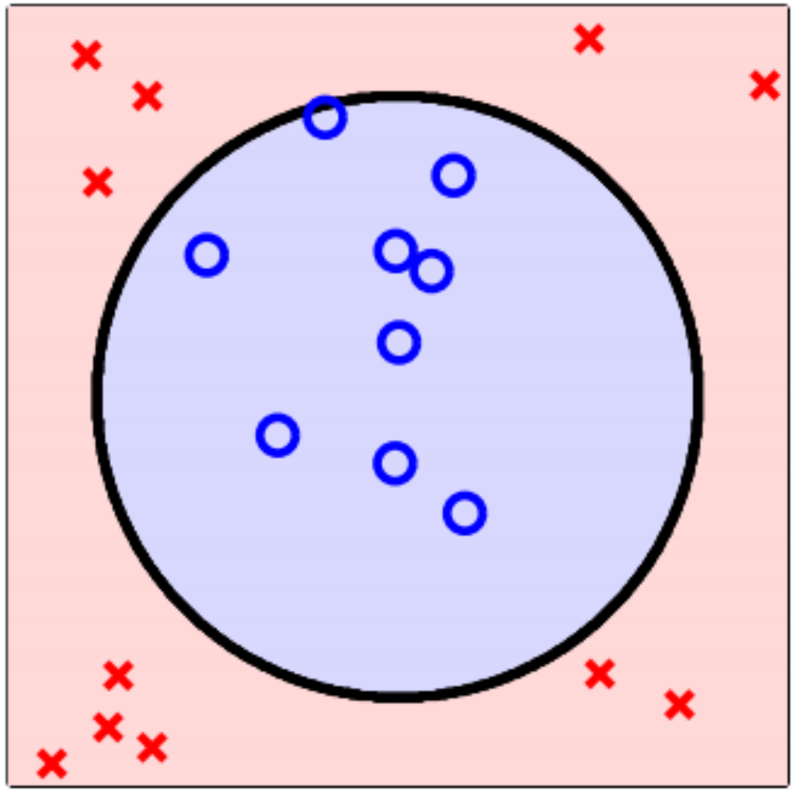


$$
\begin{array}{ll}
\text{minimize}_{\vec{w},b,\vec{\xi}} & \frac{1}{2}\vec{w}^T\vec{w} + C\sum_{n=1}^{N}\xi_n \\
\text{subject to} & y_n(\vec{w}^T\vec{x}_n + b) \geq 1 - \xi_n, \forall n \\
 & \xi_n \geq 0, \forall n
\end{array}
$$

Additional Remarks: Think about ξ_n

- $\xi_n = 0$: $\vec{x}_n$ is outside of the margin
- $\xi_n \in (0,1)$: $\vec{x}_n$ is correctly classified, but inside the margin
- $\xi_n \geq 1$: $\vec{x}_n$ is incorrectly classified

What if Tolerating Small Noises Is Not Enough



Nonlinear transform

We can apply standard nonlinear transformation procedure we talked about before

In SVM, we can combine the ideas of dual formulation and kernel tricks for the transformation

This is one of the key ingredients that makes SVM powerful

Today’s Lecture

(Get prepared for heavier math today…)

The notes are not intended to be comprehensive. They should be accompanied by lectures and/or textbook.

Let me know if you spot errors.

Lagrangian Duality and Convex Optimization

[The next few slides are safe to skip for the exam,
but they contain useful concepts for optimization/ML]

Convex Optimization

- Standard form of convex optimization

$$\begin{aligned} &\text{minimize}_{\vec{w}} && f(\vec{w}) && \\ &\text{subject to} && g_i(\vec{w}) \le 0, && i = 1, \dots, k \\ & && h_j(\vec{w}) = 0, && j = 1, \dots, \ell \end{aligned}$$

Objective

Inequality constraints

Equality constraints

- Convex program
  - f and g_i are convex and h_j are affine
  - Mostly implies the existence of efficient solvers
  - Special cases
    - Linear program: f, g_i, h_j are all affine
    - Quadratic program: f is quadratic; g_i and h_j are affine

An affine function is in the form of $A\vec{w} + \vec{b}$

[Safe to Skip for the Exam]

Lagrangian

$$\begin{array}{lll} \text{minimize}_{\vec{w}} & f(\vec{w}) & \\ \text{subject to} & g_i(\vec{w}) \leq 0, & i = 1, \dots, k \\ & h_j(\vec{w}) = 0, & j = 1, \dots, \ell \end{array}$$

- The Lagrangian of the convex program can be written as

$$L(\vec{w}, \vec{\alpha}, \vec{\beta}) = f(\vec{w}) + \sum_{i=1}^{k} \alpha_i g_i(\vec{w}) + \sum_{j=1}^{\ell} \beta_j h_j(\vec{w})$$

  - Couple each inequality constraint g_i with a dual variable α_i
  - Couple each equality constraint h_j with a dual variable β_j

- Think about the following expression

$$\max_{\vec{\alpha}, \vec{\beta}; \alpha_i \geq 0} L(\vec{w}, \vec{\alpha}, \vec{\beta}) = \begin{cases} f(\vec{w}), & \text{if all constraints are satisfied} \\ \infty, & \text{otherwise} \end{cases}$$

[Safe to Skip for the Exam]

Primal-Dual Formulation

- **Primal** problem (the standard form of convex optimization)

$$\min_{\vec{w}} \max_{\vec{\alpha},\vec{\beta};\alpha_i \geq 0} L(\vec{w},\vec{\alpha},\vec{\beta})$$

- **Dual** problem

$$\max_{\vec{\alpha},\vec{\beta};\alpha_i \geq 0} \min_{\vec{w}} L(\vec{w},\vec{\alpha},\vec{\beta})$$

<u>Reminders of definitions:</u>

$$\begin{aligned} &\text{minimize}_{\vec{w}} && f(\vec{w}) && \\ &\text{subject to} && g_i(\vec{w}) \leq 0, && i = 1,\dots,k \\ & && h_j(\vec{w}) = 0, && j = 1,\dots,\ell \end{aligned}$$

$$L(\vec{w},\vec{\alpha},\vec{\beta}) = f(\vec{w}) + \sum_{i=1}^{k} \alpha_i g_i(\vec{w}) + \sum_{j=1}^{\ell} \beta_j h_j(\vec{w})$$

- Minimax theorem [von Neumann, 1928]:

For convex programs, under mild conditions,

$$\min_{\vec{w}} \max_{\vec{\alpha},\vec{\beta};\alpha_i \geq 0} L(\vec{w},\vec{\alpha},\vec{\beta}) = \max_{\vec{\alpha},\vec{\beta};\alpha_i \geq 0} \min_{\vec{w}} L(\vec{w},\vec{\alpha},\vec{\beta})$$

[Safe to Skip for the Exam]

SVM and Regularization

Reminders of definitions in general convex program:

$$\begin{aligned} &\text{minimize}_{\vec{w}} \quad f(\vec{w}) \\ &\text{subject to} \quad g_i(\vec{w}) \leq 0, \qquad i = 1, \dots, k \\ &\qquad\qquad\quad\; h_j(\vec{w}) = 0, \qquad j = 1, \dots, \ell \end{aligned}$$

$$L(\vec{w}, \vec{\alpha}, \vec{\beta}) = f(\vec{w}) + \sum_{i=1}^{k} \alpha_i g_i(\vec{w}) + \sum_{j=1}^{\ell} \beta_j h_j(\vec{w})$$

Primal: $\min_{\vec{w}} \max_{\vec{\alpha},\vec{\beta};\alpha_i \geq 0} L(\vec{w}, \vec{\alpha}, \vec{\beta})$

Dual: $\max_{\vec{\alpha},\vec{\beta};\alpha_i \geq 0} \min_{\vec{w}} L(\vec{w}, \vec{\alpha}, \vec{\beta})$

Exercise:
Remember the weight-decay regularization:

$$\begin{aligned} &\text{minimize}_{\vec{w}} \quad E_{in}(\vec{w}) \\ &\text{subject to} \quad \vec{w}^T \vec{w} \leq C \end{aligned}$$

And the hard-margin SVM

$$\begin{aligned} &\text{minimize}_{\vec{w}} \, \vec{w}^T \vec{w} \\ &\text{subject to} \quad E_{in}(\vec{w}) = 0 \end{aligned}$$

Use what we talked about to write the unconstrained optimization problem.

[Safe to Skip for the Exam]

Minimax Theorem [von Neumann, 1928]

$$\min_{\vec{w}} \max_{\vec{\alpha},\vec{\beta};\alpha_i\geq 0} L(\vec{w},\vec{\alpha},\vec{\beta}) = \max_{\vec{\alpha},\vec{\beta};\alpha_i\geq 0} \min_{\vec{w}} L(\vec{w},\vec{\alpha},\vec{\beta})$$

- Remarks
  - The optimal primal is the same as the optimal dual for (most) convex programs!
    - We can work on a different problem space to address the original problem
    - We'll demonstrate the usage of this in SVM, but it's also useful in other applications
  - This is an important result in many areas -- e.g., it is considered as the starting point of game theory (two-player zero-sum game).
- Now we know the objectives of the optimal dual and the optimal primal are the same. How are the optimal solutions related?

[Safe to Skip for the Exam]

Karush-Kuhn-Tucker (KKT) Conditions

Lagrangian:

$$L(\vec{w}, \vec{\alpha}, \vec{\beta}) = f(\vec{w}) + \sum_{i=1}^{k} \alpha_i g_i(\vec{w}) + \sum_{j=1}^{\ell} \beta_j h_j(\vec{w})$$

Primal: $\min_{\vec{w}} \max_{\vec{\alpha},\vec{\beta};\alpha_i \geq 0} L(\vec{w}, \vec{\alpha}, \vec{\beta})$

Dual: $\max_{\vec{\alpha},\vec{\beta};\alpha_i \geq 0} \min_{\vec{w}} L(\vec{w}, \vec{\alpha}, \vec{\beta})$

- The optimal solutions $(\vec{w}^*, \vec{\alpha}^*, \vec{\beta}^*)$ satisfy the following conditions
  - Stationary condition: $\nabla_{\vec{w}} L(\vec{w}, \vec{\alpha}^*, \vec{\beta}^*)|_{\vec{w}=\vec{w}^*} = \vec{0}$
  - Primal feasibility: $g_i(\vec{w}^*) \leq 0$; $h_j(\vec{w}^*) = 0$ for all (i, j)
  - Dual feasibility: $\alpha_i^* \geq 0$ for all i
  - Complementary slackness: $\alpha_i^* g_i(\vec{w}^*) = 0$ for all i

[Safe to Skip for the Exam]

Dual SVM

1. Derive the corresponding dual from hard-margin SVM
2. Connect optimal primal solution with optimal dual solution using KKT conditions

Derive the Dual for Hard-Margin SVM

- Hard-margin SVM

$$\begin{aligned} &\text{minimize}_{\vec{w},b} && \frac{1}{2}\vec{w}^T\vec{w} \\ &\text{subject to} && y_n(\vec{w}^T\vec{x}_n + b) \geq 1, \forall n \end{aligned}$$

- First write down the Lagrangian

<u>Reminders of definitions in general convex program:</u>

$$\begin{aligned} &\text{minimize}_{\vec{w}} && f(\vec{w}) && \\ &\text{subject to} && g_i(\vec{w}) \leq 0, && i = 1, \dots, k \\ & && h_j(\vec{w}) = 0, && j = 1, \dots, \ell \end{aligned}$$

$$L(\vec{w}, \vec{\alpha}, \vec{\beta}) = f(\vec{w}) + \sum_{i=1}^{k} \alpha_i g_i(\vec{w}) + \sum_{j=1}^{\ell} \beta_j h_j(\vec{w})$$

Dual: $\max\limits_{\vec{\alpha},\vec{\beta};\alpha_i \geq 0} \min\limits_{\vec{w}} L(\vec{w}, \vec{\alpha}, \vec{\beta})$

Derive the Dual for Hard-Margin SVM

- Hard-margin SVM

$$\begin{aligned} &\text{minimize}_{\vec{w},b} && \frac{1}{2}\vec{w}^T\vec{w} \\ &\text{subject to} && y_n(\vec{w}^T\vec{x}_n + b) \geq 1, \forall n \end{aligned}$$

<u>Reminders of definitions in general convex program:</u>

$$\begin{aligned} &\text{minimize}_{\vec{w}} && f(\vec{w}) && \\ &\text{subject to} && g_i(\vec{w}) \leq 0, && i = 1, \dots, k \\ & && h_j(\vec{w}) = 0, && j = 1, \dots, \ell \end{aligned}$$

$$L(\vec{w}, \vec{\alpha}, \vec{\beta}) = f(\vec{w}) + \sum_{i=1}^{k} \alpha_i g_i(\vec{w}) + \sum_{j=1}^{\ell} \beta_j h_j(\vec{w})$$

Dual: $\max_{\vec{\alpha}, \vec{\beta}; \alpha_i \geq 0} \min_{\vec{w}} L(\vec{w}, \vec{\alpha}, \vec{\beta})$

- First write down the Lagrangian

$$\begin{aligned} L(\vec{w}, b, \vec{\alpha}) &= \frac{1}{2}\vec{w}^T\vec{w} + \sum_{n=1}^{N} \alpha_n \left(1 - y_n(\vec{w}^T\vec{x}_n + b)\right) \\ &= \frac{1}{2}\vec{w}^T\vec{w} + \sum_{n=1}^{N} \alpha_n - \sum_{n=1}^{N} \alpha_n y_n(\vec{w}^T\vec{x}_n + b) \end{aligned}$$

- Dual

$$\max_{\vec{\alpha}; \alpha_i \geq 0} \min_{\vec{w}, b} L(\vec{w}, b, \vec{\alpha})$$

- Lagrangian $L(\vec{w}, b, \vec{\alpha}) = \frac{1}{2}\vec{w}^T\vec{w} + \sum_{n=1}^{N} \alpha_n - \sum_{n=1}^{N} \alpha_n y_n(\vec{w}^T\vec{x}_n + b)$
- Dual $\max\limits_{\vec{\alpha};\alpha_i \geq 0} \min\limits_{\vec{w},b} L(\vec{w}, b, \vec{\alpha})$ (the variables in the dual are $\vec{\alpha}$)

- Derivations
  - Express $\vec{w}$ and b using $\vec{\alpha}$ in the dual objective $\min\limits_{\vec{w},b} L(\vec{w}, b, \vec{\alpha})$
    - Solve for $\nabla_{\vec{w},b} L(\vec{w}, b, \vec{\alpha}) = 0$

- Lagrangian $L(\vec{w}, b, \vec{\alpha}) = \frac{1}{2}\vec{w}^T\vec{w} + \sum_{n=1}^{N}\alpha_n - \sum_{n=1}^{N}\alpha_n y_n(\vec{w}^T\vec{x}_n + b)$
- Dual $\max_{\vec{\alpha};\alpha_i \geq 0} \min_{\vec{w},b} L(\vec{w}, b, \vec{\alpha})$ (the variables in the dual are $\vec{\alpha}$)

- Derivations
  - Express $\vec{w}$ and b using $\vec{\alpha}$ in the dual objective $\min_{\vec{w},b} L(\vec{w}, b, \vec{\alpha})$
    - Solve for $\nabla_{\vec{w},b} L(\vec{w}, b, \vec{\alpha}) = 0$
      - $\nabla_{\vec{w}} L(\vec{w}, b, \vec{\alpha}) = 0$ => $\vec{w} - \sum_{n=1}^{N}\alpha_n y_n \vec{x}_n = 0$ => $\vec{w} = \sum_{n=1}^{N}\alpha_n y_n \vec{x}_n$
      - $\nabla_{b} L(\vec{w}, b, \vec{\alpha}) = 0$ => $\sum_{n=1}^{N}\alpha_n y_n = 0$

    - Plug $\vec{w} = \sum_{n=1}^{N}\alpha_n y_n \vec{x}_n$ into $L(\vec{w}, b, \vec{\alpha})$
      - $\frac{1}{2}\vec{w}^T\vec{w} = \frac{1}{2}\sum_{n=1}^{N}\sum_{m=1}^{N}\alpha_n\alpha_m y_n y_m \vec{x}_n^T\vec{x}_m$
      - $\sum_{n=1}^{N}\alpha_n y_n(\vec{w}^T\vec{x} + b) = \sum_{n=1}^{N}\sum_{m=1}^{N}\alpha_n\alpha_m y_n y_m \vec{x}_n^T\vec{x}_m + b\sum_{n=1}^{N}\alpha_n y_n$

  - $\min_{\vec{w},b} L(\vec{w}, b, \vec{\alpha}) = \sum_{n=1}^{N}\alpha_n - \frac{1}{2}\sum_{n=1}^{N}\sum_{m=1}^{N}\alpha_n\alpha_m y_n y_m \vec{x}_n^T\vec{x}_m$

- Lagrangian $L(\vec{w}, b, \vec{\alpha}) = \frac{1}{2}\vec{w}^T\vec{w} + \sum_{n=1}^{N} \alpha_n - \sum_{n=1}^{N} \alpha_n y_n(\vec{w}^T\vec{x}_n + b)$
- Dual $\max_{\vec{\alpha};\alpha_i \geq 0} \min_{\vec{w},b} L(\vec{w}, b, \vec{\alpha})$ (the variables in the dual are $\vec{\alpha}$)

Dual Constraint

- Derivations
  - Express $\vec{w}$ and b using $\vec{\alpha}$ in the dual objective $\min_{\vec{w},b} L(\vec{w}, b, \vec{\alpha})$
    - Solve for $\nabla_{\vec{w},b} L(\vec{w}, b, \vec{\alpha}) = 0$
      - $\nabla_{\vec{w}} L(\vec{w}, b, \vec{\alpha}) = 0$ => $\vec{w} - \sum_{n=1}^{N} \alpha_n y_n \vec{x}_n = 0$ => $\vec{w} = \sum_{n=1}^{N} \alpha_n y_n \vec{x}_n$
      - $\nabla_b L(\vec{w}, b, \vec{\alpha}) = 0$ => $\sum_{n=1}^{N} \alpha_n y_n = 0$

Dual Constraint

    - Plug $\vec{w} = \sum_{n=1}^{N} \alpha_n y_n \vec{x}_n$ into $L(\vec{w}, b, \vec{\alpha})$
      - $\frac{1}{2}\vec{w}^T\vec{w} = \frac{1}{2}\sum_{n=1}^{N}\sum_{m=1}^{N} \alpha_n \alpha_m y_n y_m \vec{x}_n^T \vec{x}_m$
      - $\sum_{n=1}^{N} \alpha_n y_n(\vec{w}^T\vec{x} + b) = \sum_{n=1}^{N}\sum_{m=1}^{N} \alpha_n \alpha_m y_n y_m \vec{x}_n^T \vec{x}_m + b\sum_{n=1}^{N} \alpha_n y_n$
  - $\min_{\vec{w},b} L(\vec{w}, b, \vec{\alpha}) = \sum_{n=1}^{N} \alpha_n - \frac{1}{2}\sum_{n=1}^{N}\sum_{m=1}^{N} \alpha_n \alpha_m y_n y_m \vec{x}_n^T \vec{x}_m$

Dual Objective

Dual SVM

- Dual of the hard-margin SVM

$$\begin{aligned} &\text{maximize}_{\vec{\alpha}} \sum_{n=1}^{N} \alpha_n - \frac{1}{2}\sum_{n=1}^{N}\sum_{m=1}^{N} \alpha_n \alpha_m y_n y_m \vec{x}_n^T \vec{x}_m \\ &\text{subject to} \quad \sum_{n=1}^{N} \alpha_n y_n = 0 \\ &\qquad\qquad\quad\;\; \alpha_n \geq 0, \forall n \end{aligned}$$

  - The dual is still a Quadratic Program, with efficient solvers to find $\vec{\alpha}^*$

- We know that the objective of the optimal dual is the same as the optimal primal
- Say we obtain $\vec{\alpha}^*$, how do we recover the optimal primal $(\vec{w}^*, b^*)$?
  - Apply KKT conditions

Recover $(\vec{w}^*, b^*)$ from $\vec{\alpha}^*$

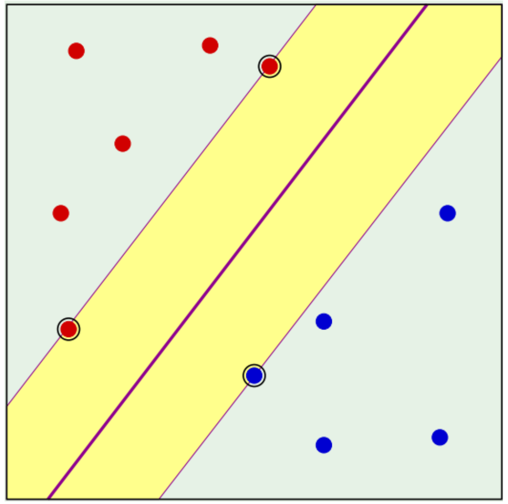
$L(\vec{w}, b, \vec{\alpha}) = \frac{1}{2}\vec{w}^T\vec{w} + \sum_{n=1}^{N} \alpha_n - \sum_{n=1}^{N} \alpha_n y_n(\vec{w}^T\vec{x}_n + b)$

- Using stationary conditions in KKT
  - $\nabla_{\vec{w}} L(\vec{w}, b^*, \vec{\alpha}^*)|_{\vec{w}=\vec{w}^*} = \vec{0}$
  - $\vec{w}^* = \sum_{n=1}^{N} \alpha_n^* y_n \vec{x}_n$
  - Since $\alpha_n^* \geq 0$, we can rewrite $\vec{w}^* = \sum_{\alpha_n^*>0} \alpha_n^* y_n \vec{x}_n$

- Using complementary slackness in KKT
  - $\alpha_n^*\left(1 - y_n(\vec{x}_n^T\vec{w}^* + b^*)\right) = 0$
  - Find a $\alpha_n^* > 0$, we have $y_n(\vec{x}_n^T\vec{w}^* + b^*) = 1$
  - Since $y_n \in \{+1, -1\}$, we have $\vec{x}_n^T\vec{w}^* + b^* = y_n$
  - Therefore,
    - $b^* = y_n - \vec{x}_n^T\vec{w}^*$ (with $\vec{w}^* = \sum_{\alpha_n^*>0} \alpha_n^* y_n \vec{x}_n$)

Note that $\vec{w}^T\vec{x} = \vec{x}^T\vec{w}$.
I swapped the order to avoid two superscripts in $\vec{w}$

Recover $(\vec{w}^*, b^*)$ from $\vec{\alpha}^*$



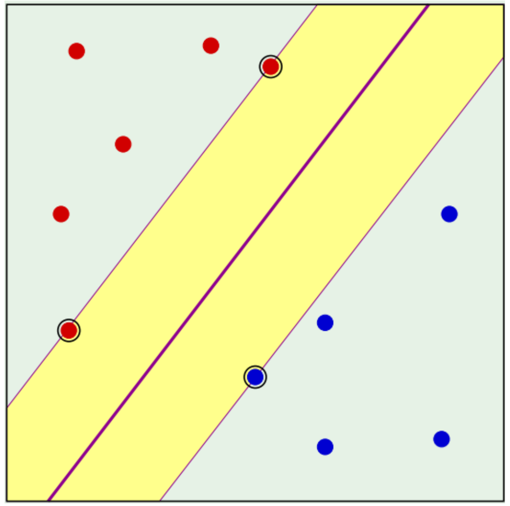
- Solve the dual and find $\vec{\alpha}^*$
  - $\vec{w}^* = \sum_{\alpha_n^*>0} \alpha_n^* y_n \vec{x}_n$
  - $b^* = y_n - \vec{x}_n^T \vec{w}^*$ for some $\alpha_n^* > 0$
  - $g(\vec{x}) = sign({\vec{w}^*}^T \vec{x} + b^*)$
- What does $\alpha_n^* > 0$ imply?
  - Complementary slackness $\alpha_n^* \left(1 - y_n(\vec{x}_n^T \vec{w}^* + b^*)\right) = 0$
  - $\alpha_n^* > 0$ => $y_n(\vec{x}_n^T \vec{w}^* + b^*) = 1$
- $\alpha_n^* > 0$ => $(\vec{x}_n, y_n)$ is the support vector
  - $\vec{w}^* = \sum_{\alpha_n^*>0} \alpha_n^* y_n \vec{x}_n$ is the linear combination of support vectors!
  - Support vector machine!

Support Vectors

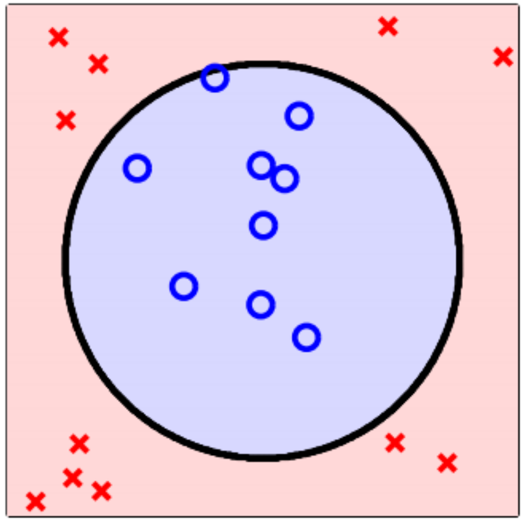
- Primal point of view
  - We call the points closest to the separator (candidate) support vectors
  - They are the points that the equality holds in the constraints
    - If $\vec{x}_n$ is a support vector, $y_n(\vec{w}^T\vec{x}_n + b) = 1$
  - Removing the non-support vectors will not impact the linear separator

- Dual point of view
  - If $\alpha_n^* > 0$ => $(\vec{x}_n, y_n)$ is the support vector
  - The optimal separator $(\vec{w}^*, b^*)$
    - $\vec{w}^* = \sum_{\alpha_n^*>0} \alpha_n^* y_n \vec{x}_n$
    - $b^* = y_n - \vec{x}_n^T\vec{w}^*$ for some $\alpha_n^* > 0$

  - $(\vec{w}^*, b^*)$ can be defined by "support vectors"
    - Support vector machine!



Nonlinear Transform and Kernel Tricks



Primal-Dual Formulations of Hard-Margin SVM

- Primal

$$\begin{aligned} &\text{minimize}_{\vec{w},b} \quad \frac{1}{2}\vec{w}^T\vec{w} \\ &\text{subject to} \quad y_n(\vec{w}^T\vec{x}_n + b) \geq 1, \forall n \end{aligned}$$

- Dual

$$\begin{aligned} &\text{maximize}_{\vec{\alpha}} \sum_{n=1}^{N} \alpha_n - \frac{1}{2}\sum_{n=1}^{N}\sum_{m=1}^{N} \alpha_n\alpha_m y_n y_m \vec{x}_n^T\vec{x}_m \\ &\text{subject to} \quad \sum_{n=1}^{N} \alpha_n y_n = 0 \\ &\qquad\qquad\quad \alpha_n \geq 0, \forall n \end{aligned}$$

- Both can be efficiently solved using QP solvers
- We can infer the solution from one to the other

Given optimal $\vec{\alpha}^*$:

- $\vec{w}^* = \sum_{\alpha_n^*>0} \alpha_n^* y_n \vec{x}_n$
- Find a $\alpha_n^* > 0,\ b^* = y_n - \vec{x}_n^T\vec{w}^*$

Nonlinear Transform: $\vec{z} = \Phi(\vec{x})$

- Primal

$$\begin{aligned} &\text{minimize}_{\vec{w},b} && \frac{1}{2}\vec{w}^T\vec{w} \\ &\text{subject to} && y_n(\vec{w}^T\vec{z}_n + b) \geq 1, \forall n \end{aligned}$$

Involves changing $\vec{w}$ and $\vec{z}$.
The computation grows as the dimension of the $\vec{z}$ space grows

- Dual

$$\begin{aligned} &\text{maximize}_{\vec{\alpha}} \sum_{n=1}^{N} \alpha_n - \frac{1}{2}\sum_{n=1}^{N}\sum_{m=1}^{N} \alpha_n\alpha_m y_n y_m \vec{z}_n^T\vec{z}_m \\ &\text{subject to} \ \sum_{n=1}^{N} \alpha_n y_n = 0 \\ &\qquad\qquad\quad \alpha_n \geq 0, \forall n \end{aligned}$$

The only difference is from calculating $\vec{x}_n^T\vec{x}_m$ to $\vec{z}_n^T\vec{z}_m$

- Intuition: If we can find an efficient way to calculate $\vec{z}_n^T\vec{z}_m$, we can derive the optimal dual to infer the optimal primal.
  - Doing nonlinear transform without sacrificing much about computation.

Example: 2nd Order Polynomial Transform

We define the transform slight differently

- The $\sqrt{2}$ and the initial 1 are not in the original transform, but we include them for convenience.

- $\vec{x} = (x_1, x_2)$
- 2nd order polynomial transform
  - $\vec{z} = \Phi_2(\vec{x}) = (1, \sqrt{2}x_1, \sqrt{2}x_2, \sqrt{2}\, x_1x_2, x_1^2, x_2^2)$

$$
\begin{aligned}
\vec{z}^T\vec{z}' &= 1 + 2x_1x_1' + 2x_2x_2' + 2x_1x_1'x_2x_2' + x_1^2{x_1'}^2 + x_2^2{x_2'}^2 \\
&= 1 + 2x_1x_1' + 2x_2x_2' + 2x_1x_1'x_2x_2' + (x_1x_1')^2 + (x_2x_2')^2 \\
&= (1 + x_1x_1' + x_2x_2')^2 \\
&= (1 + \vec{x}^T\vec{x}')^2
\end{aligned}
$$

- We can calculate $\vec{z}^T\vec{z}'$ from the operation in the $\vec{x}$ space!

Kernel Functions

- Define kernel function $K_\Phi(\vec{x}, \vec{x}') = \Phi(\vec{x})^T \Phi(\vec{x}')$
  - The similarity of two vectors in the projected space
- Goal: Compute $K_\Phi(\vec{x}, \vec{x}')$ without transforming $\vec{x}$ and $\vec{x}'$
- Why? This enables us to operate in the higher dimensional space without really worrying about the computational overhead.

Kernel Trick: Utilize Dual and Kernel Functions

- The dual with nonlinear transform

$$\begin{aligned}&\text{maximize}_{\vec{\alpha}} \sum_{n=1}^{N} \alpha_n - \frac{1}{2}\sum_{n=1}^{N}\sum_{m=1}^{N} \alpha_n \alpha_m y_n y_m \vec{z}_n^T \vec{z}_m \\ &\text{subject to} \quad \sum_{n=1}^{N} \alpha_n y_n = 0 \\ &\qquad\qquad\quad\ \alpha_n \geq 0, \forall n\end{aligned}$$

- Plug in the kernel function $K_\Phi(\vec{x}, \vec{x}') = \Phi(\vec{x})^T \Phi(\vec{x}')$

$$\begin{aligned}&\text{maximize}_{\vec{\alpha}} \sum_{n=1}^{N} \alpha_n - \frac{1}{2}\sum_{n=1}^{N}\sum_{m=1}^{N} \alpha_n \alpha_m y_n y_m K_\Phi(\vec{x}_n, \vec{x}_m) \\ &\text{subject to} \quad \sum_{n=1}^{N} \alpha_n y_n = 0 \\ &\qquad\qquad\quad\ \alpha_n \geq 0, \forall n\end{aligned}$$

  - If the kernel can be computed efficiently, we can solve $\vec{\alpha}^*$ efficiently.
  - With kernel tricks, we can avoid the dependency on the dimension of $\vec{z}$

Recover $(\vec{w}^*, b^*)$ from $\vec{\alpha}^*$ with Kernel Tricks

- Note that $\vec{\alpha}^*$ is solved in the $\vec{z}$ space
  - $\vec{w}^* = \sum_{\alpha_n^*>0} \alpha_n^* y_n \Phi(\vec{x}_n)$
  - Find a $\alpha_n^* > 0,\ b^* = y_n - \vec{w}^{*T} \Phi(\vec{x}_n)$
  - We want to avoid the transformation!

- Let's look at the hypothesis $g(\vec{x}) = sign\left(\vec{w}^{*T} \Phi(\vec{x}) + b^*\right)$

$$\begin{aligned}\vec{w}^{*T} \Phi(\vec{x}) &= \left(\sum_{\alpha_n^*>0} \alpha_n^* y_n \Phi(\vec{x}_n)\right)^T \Phi(\vec{x}) \\ &= \sum_{\alpha_n^*>0} \alpha_n^* y_n \Phi(\vec{x}_n)^{\mathrm{T}} \Phi(\vec{x}) \\ &= \sum_{\alpha_n^*>0} \alpha_n^* y_n K(\vec{x}_n, \vec{x})\end{aligned}$$

$$\begin{aligned}b^* &= y_n - \vec{w}^{*T} \Phi(\vec{x}_n) \text{ (for some } n \text{ that } \alpha_n^* > 0) \\ &= y_n - \left(\sum_{\alpha_m^*>0} \alpha_m^* y_m \Phi(\vec{x}_m)\right)^{\mathrm{T}} \Phi(\vec{x}_n) \\ &= y_n - \sum_{\alpha_m^*>0} \alpha_m^* y_m K(\vec{x}_m, \vec{x}_n)\end{aligned}$$

  - Utilize support vectors to make predictions on $\vec{x}$
    - Still can be computed in the $\vec{x}$ space!

Kernel Functions

$K_\Phi(\vec{x}, \vec{x}')$: Inner products of two points $\Phi(\vec{x})^T \Phi(\vec{x}')$ in the transformed space

Similarity of two points $\Phi(\vec{x})$ and $\Phi(\vec{x}')$ in the transformed space

Polynomial Kernel

Kernel $K(\vec{x}, \vec{x}') = \Phi(\vec{x})^T \Phi(\vec{x}')$

- Example we have discussed: 2nd order polynomial for 2-d $\vec{x}$
  - $\vec{x} = (x_1, x_2)$
  - $\vec{z} = \Phi_2(\vec{x}) = (1, \sqrt{2}x_1, \sqrt{2}x_2, \sqrt{2}\, x_1 x_2, x_1^2, x_2^2)$
  - $\vec{z}' = \Phi_2(\vec{x}') = (1, \sqrt{2}x_1', \sqrt{2}x_2', \sqrt{2}\, x_1' x_2', {x'}_1^2, {x'}_2^2)$
  - $$\begin{aligned} \vec{z}^T \vec{z}' &= 1 + 2x_1x_1' + 2x_2x_2' + 2x_1x_1'x_2x_2' + (x_1x_1')^2 + (x_2x_2')^2 \\ &= (1 + x_1x_1' + x_2x_2')^2 \\ &= (1 + \vec{x}^T\vec{x}')^2 \end{aligned}$$
- General 2nd order polynomial
  - $\vec{x} = (x_1, x_2, \dots, x_d)$
  - $$\begin{aligned} K_{\Phi_2}(\vec{x}, \vec{x}') &= (1 + \vec{x}^T\vec{x}')^2 \\ &= (1 + x_1x_1' + x_2x_2' + \cdots + x_dx_d')^2 \end{aligned}$$

Polynomial Kernel

General form of polynomial kernel:
$K(\vec{x},\vec{x}') = (a\vec{x}^T\vec{x}' + b)^Q$

- $\vec{x} = (x_1, x_2, \dots, x_d)$

- 2nd order polynomial kernel $K_{\Phi_2}(\vec{x},\vec{x}') = (1 + \vec{x}^T\vec{x}')^2$
- Q-th order Polynomial kernel $K_{\Phi_Q}(\vec{x},\vec{x}') = (1 + \vec{x}^T\vec{x}')^Q$

$$= (1 + x_1x_1' + \cdots + x_dx_d')^Q$$

- Computational complexity
  - Dimension of $\Phi_Q(\vec{x})$: $\binom{Q+d}{Q}$
  - Direct computation of $\Phi_Q(\vec{x})^T\Phi_Q(\vec{x}')$: $O\left(\binom{Q+d}{Q}\right)$
  - Computation through kernel $K_{\Phi_Q}(\vec{x},\vec{x}')$: $O(d)$

We Only Need $\vec{z}$ Space to Exist

- In the discussion of polynomial kernels
  - We have a target transformation in mind
  - We want to find a corresponding kernel function

- In fact, as long as $K(\vec{x}, \vec{x}')$ is an inner product in **some** $\vec{z}$ space, we are good
  - Just plug in the kernel in the dual formulation
  - We obtain a linear separator in the corresponding $\vec{z}$ space

$$\text{maximize}_{\vec{\alpha}} \sum_{n=1}^{N} \alpha_n - \frac{1}{2}\sum_{n=1}^{N}\sum_{m=1}^{N} \alpha_n \alpha_m y_n y_m K_{\Phi}(\vec{x}_n, \vec{x}_m)$$

$$\text{subject to } \sum_{n=1}^{N} \alpha_n y_n = 0$$

$$\alpha_n \geq 0, \forall n$$

Gaussian RBF Kernel

- $K(\vec{x}, \vec{x}') = e^{-\gamma\|\vec{x}-\vec{x}'\|^2}$
- What's the corresponding $\vec{z}$ space? (What is Φ such that $\Phi(\vec{x})^T\Phi(\vec{x}') = e^{-\gamma\|\vec{x}-\vec{x}'\|^2}$)
  - For illustrative purpose, make $\vec{x} = x$ be 1 dimensional and $\gamma = 1$
    - $K(\vec{x}, \vec{x}') = e^{-(x-x')^2}$

$$
\begin{aligned}
&= e^{-x^2+2xx'-x'^2} \\
&= e^{-x^2}e^{-x'^2}e^{2xx'} \\
&= e^{-x^2}e^{-x'^2}\sum_{k=0}^{\infty}\frac{(2xx')^k}{k!} \\
&= \sum_{k=0}^{\infty} e^{-x^2}\sqrt{\frac{2^k}{k!}}x^k e^{-x'^2}\sqrt{\frac{2^k}{k!}}x'^k
\end{aligned}
$$

Taylor expansion: $e^{2xx'} = \sum_{k=0}^{\infty}\frac{(2xx')^k}{k!}$

- The corresponding $\Phi(x) = e^{-x^2}\left(1, \sqrt{\frac{2}{1}}x, \sqrt{\frac{2^2}{2!}}x^2, \dots\right)$

Gaussian RBF Kernel

- $K(\vec{x}, \vec{x}') = e^{-\gamma \|\vec{x} - \vec{x}'\|^2}$
- The corresponding transform in 1-dim input $\vec{x} = x$
  - $\Phi(x) = e^{-x^2}\left(1, \sqrt{\frac{2}{1}}x, \sqrt{\frac{2^2}{2!}}x^2, \ldots\right)$
- $K(\vec{x}, \vec{x}')$ is the inner product of two vectors in an infinite dimensional space!
- When we plug in $K(\vec{x}, \vec{x}')$ in dual SVM
  - We are finding the max-margin separator in an infinite dimensional space
  - Seems to introduce infinite generalization error?
    - Maximizing margin help mitigate this issue
    - The number of support vectors provides indicators on the generalization

Design Your Own Kernel? [Safe to Skip]

- Say we design a kernel function, how do we know whether it is valid, i.e., whether there is a corresponding $\vec{z}$ space?

- Mercer's condition (See discussion in LFD 8.3.2)
  - Kernel matrix

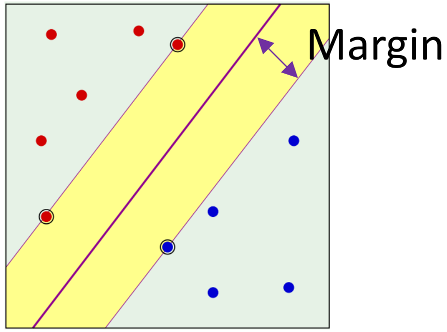
$$\begin{bmatrix} K(\mathbf{x}_1, \mathbf{x}_1) & K(\mathbf{x}_1, \mathbf{x}_2) & \dots & K(\mathbf{x}_1, \mathbf{x}_N) \\ K(\mathbf{x}_2, \mathbf{x}_1) & K(\mathbf{x}_2, \mathbf{x}_2) & \dots & K(\mathbf{x}_2, \mathbf{x}_N) \\ \dots & \dots & \dots & \dots \\ K(\mathbf{x}_N, \mathbf{x}_1) & K(\mathbf{x}_N, \mathbf{x}_2) & \dots & K(\mathbf{x}_N, \mathbf{x}_N) \end{bmatrix}$$

  - $K(\vec{x}, \vec{x}')$ is a valid kernel if and only if the kernel matrix is always symmetric positive semi-definite for any $\vec{x}_1, \dots, \vec{x}^N$

Summary of What We Talked About So Far

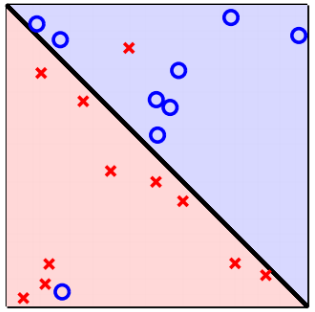
Hard-Margin SVM (Separable Data)

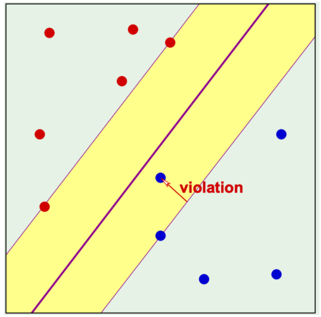
$$\text{minimize}_{\vec{w},b} \quad \frac{1}{2}\vec{w}^T\vec{w}$$
$$\text{subject to} \quad y_n(\vec{w}^T\vec{x}_n + b) \geq 1, \forall n$$



Soft-Margin SVM (Tolerate Noise)

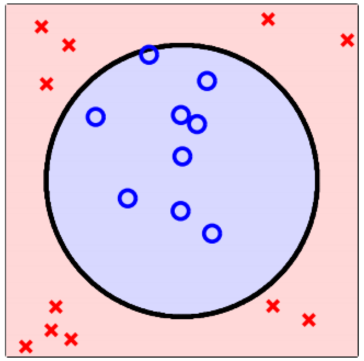
$$\text{minimize}_{\vec{w},b,\vec{\xi}} \quad \frac{1}{2}\vec{w}^T\vec{w} + C\sum_{n=1}^{N}\xi_n$$
$$\text{subject to} \quad y_n(\vec{w}^T\vec{x}_n + b) \geq 1 - \xi_n, \forall n$$
$$\xi_n \geq 0, \forall n$$





Kernel Formulation of Hard-Margin SVM

$$\text{maximize}_{\vec{\alpha}} \sum_{n=1}^{N}\alpha_n - \frac{1}{2}\sum_{n=1}^{N}\sum_{m=1}^{N}\alpha_n\alpha_m y_n y_m K_\Phi(\vec{x}_n, \vec{x}_m)$$
$$\text{subject to} \quad \sum_{n=1}^{N}\alpha_n y_n = 0$$
$$\alpha_n \geq 0, \forall n$$



Kernel Version of Soft-Margin SVM

- Soft-Margin SVM

$$\begin{aligned}
&\text{minimize}_{\vec{w},b,\vec{\xi}} && \frac{1}{2}\vec{w}^T\vec{w} + C\sum_{n=1}^{N}\xi_n \\
&\text{subject to} && y_n(\vec{w}^T\vec{x}_n + b) \geq 1 - \xi_n, \forall n \\
& && \xi_n \geq 0, \forall n
\end{aligned}$$

- Kernel Version of Soft-Margin SVM

$$\begin{aligned}
&\text{maximize}_{\vec{\alpha}} \sum_{n=1}^{N}\alpha_n - \frac{1}{2}\sum_{n=1}^{N}\sum_{m=1}^{N}\alpha_n\alpha_m y_n y_m K_\Phi(\vec{x}_n, \vec{x}_m) \\
&\text{subject to } \sum_{n=1}^{N}\alpha_n y_n = 0 \\
&\qquad\qquad\quad 0 \leq \alpha_n \leq C, \forall n
\end{aligned}$$

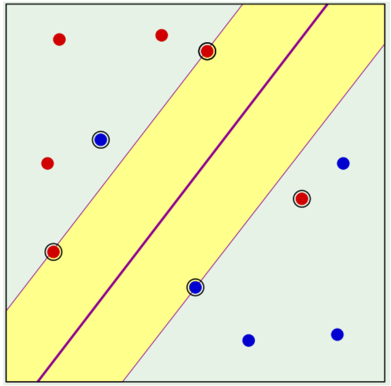
  - It can be obtained by similar procedure as hard-margin version
  - We can obtain the same relationship between $\vec{\alpha}^*$ and $(\vec{w}^*, b^*)$

Interpretation of Support Vectors

- $\alpha_n^* > 0$ => $(\vec{x}_n, y_n)$ is a support vector
  - $y_n\left(\vec{w}^{*T}\vec{x}_n + b^*\right) = 1 - \xi_n$
- Utilizing complementary slackness
  - When $0 < \alpha_n^* < C$
    - $\xi_n = 0$
    - $y_n\left(\vec{w}^{*T}\vec{x}_n + b^*\right) = 1$
    - $(\vec{x}_n, y_n)$ is a "margin" support vector
  - When $\alpha_n^* = C$
    - $\xi_n > 0$
    - $y_n\left(\vec{w}^{*T}\vec{x}_n + b^*\right) < 1$
    - $(\vec{x}_n, y_n)$ is a "non-margin" support vector

$$\text{minimize}_{\vec{w},b,\vec{\xi}} \quad \frac{1}{2}\vec{w}^T\vec{w} + C\sum_{n=1}^{N}\xi_n$$
$$\text{subject to} \quad y_n(\vec{w}^T\vec{x}_n + b) \geq 1 - \xi_n, \forall n$$
$$\xi_n \geq 0, \forall n$$

$$\text{maximize}_{\vec{\alpha}} \sum_{n=1}^{N}\alpha_n - \frac{1}{2}\sum_{n=1}^{N}\sum_{m=1}^{N}\alpha_n\alpha_m y_n y_m K_\Phi(\vec{x}_n, \vec{x}_m)$$
$$\text{subject to} \quad \sum_{n=1}^{N}\alpha_n y_n = 0$$
$$0 \leq \alpha_n \leq C, \forall n$$



Another Look at Primal vs. Dual SVM

- Primal

$$\begin{aligned}&\text{minimize}_{\vec{w},b} && \frac{1}{2}\vec{w}^T\vec{w}\\ &\text{subject to} && y_n(\vec{w}^T\vec{z}_n + b) \geq 1, \forall n\end{aligned}$$

  - Learned hypothesis
    - $g(\vec{x}) = sign\left(\vec{w}^{*^T}\Phi(\vec{x}) + b^*\right)$

- Dual

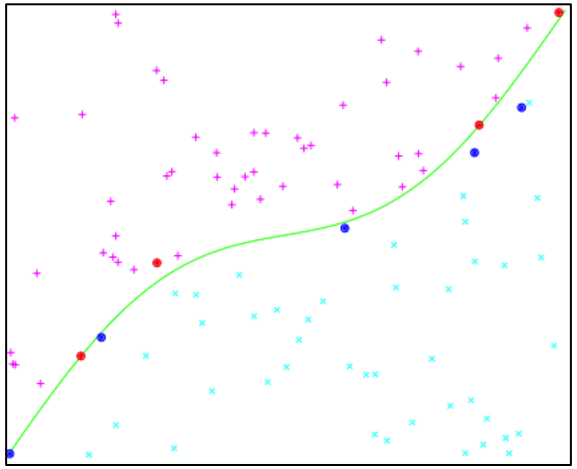
$$\begin{aligned}&\text{maximize}_{\vec{\alpha}} && \textstyle\sum_{n=1}^N \alpha_n - \frac{1}{2}\sum_{n=1}^N \sum_{m=1}^N \alpha_n\alpha_m y_n y_m \vec{z}_n^T\vec{z}_m\\ &\text{subject to} && \textstyle\sum_{n=1}^N \alpha_n y_n = 0\\ &&& \alpha_n \geq 0, \forall n\end{aligned}$$

  - Learned hypothesis
    - $g(\vec{x}) = sign(\sum_{\alpha_n^*>0} \alpha_n^* y_n K(\vec{x}_n, \vec{x}) + b^*))$
    - ($\alpha_n^* > 0 \Rightarrow \vec{x}_n$ is a support vector)

- Primal view of SVM (parametric)
  - We are learning the weights for SVM, i.e., $(\vec{w}^*, b^*)$
  - When using RBF Kernel, there are infinite number of parameters
- Dual kernel view of SVM (nonparametric)
  - We are learning the support vectors, and use those for prediction

Kernel SVM and Radial Basis Functions

- Kernel SVM
  - $g(\vec{x}) = sign\left(\sum_{\alpha_n^*>0} \alpha_n^* y_n K(\vec{x}_n, \vec{x}) + b^*)\right)$
  - Use support vectors to characterize a hypothesis



- Radial Basis Functions
  - $h(\vec{x}) = sign\left(\sum_{k=1}^{K} w_k \, \phi\left(\frac{\|\vec{x}-\vec{\mu}_k\|}{r}\right)\right)$
  - Use cluster centers to characterize a hypothesis