CSE 417T
Introduction to Machine Learning

Lecture 17

Instructor: Chien-Ju (CJ) Ho

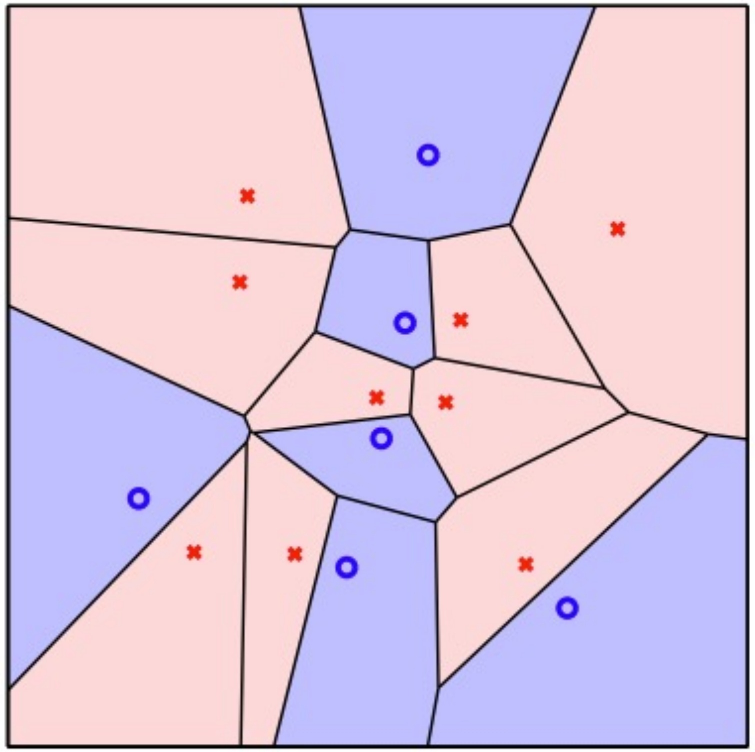
Logistics

- 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

Recap

Nearest Neighbor

$g(\vec{x})$ looks like a Voronoi diagram



- Properties of Nearest Neighbor (NN)
  - No training is needed
  - Good interpretability
  - In-sample error $E_{in} = 0$
  - VC dimension is ∞
- This seems to imply bad learning models from what we talk about so far? Why we care?
- Nearest Neighbor is 2-Optimal
  - When $N \to \infty$, with high probability,
    $E_{out} \leq 2E^*_{out}$

k-Nearest Neighbor (K-NN)

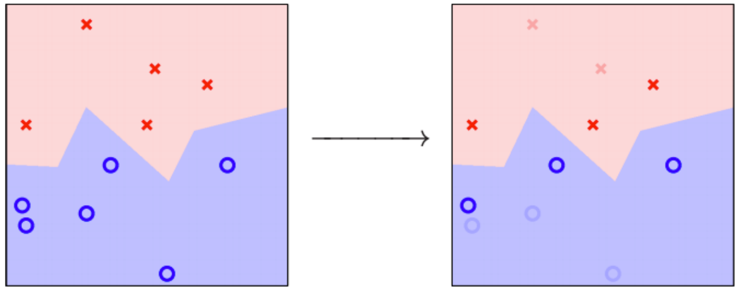
- k-nearest neighbor (K-NN)
  - $g(\vec{x}) = sign\left(\sum_{i=1}^{k} y_{[i]}(\vec{x})\right)$

- How to choose k?
  - Making the choice of k a function of N, denoted by $k(N)$
  - Theorem:
    - For $N \to \infty$, if $k(N) \to \infty$ and $\frac{k(N)}{N} \to 0$
    - Then $E_{in}(g) \to E_{out}(g)$ and $E_{out}(g) \to E_{out}(g^*)$
    - E.g., $k(N) = \sqrt{N}$
  - Other practical rules of thumb:
    - Setting a small k is often a good enough choice
    - Using validation to choose k

With suitable choice of k, when $N \to \infty$, we can recover the optimal hypothesis.

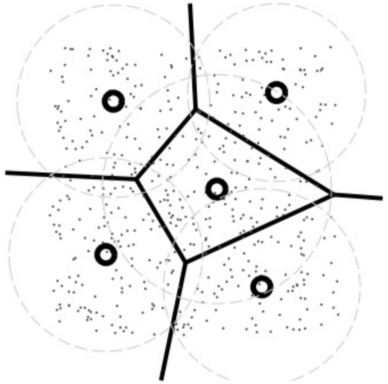
Dealing with Computational Issues

- Reduce the number of data points



- Intuition: remove points that will not impact the decision boundary.
- Generally a hard problem. But there are heuristic approaches (e.g., Condensed Nearest Neighbor).

- Store the data in some data structure to speed up searching



- Intuition: Clustering data points
- For a new data point, we might be able to "ignore" some clusters when searching for nearest neighbor.

Radial Basis Function (RBF)

- Using distance to the points as the basis function to form hypothesis

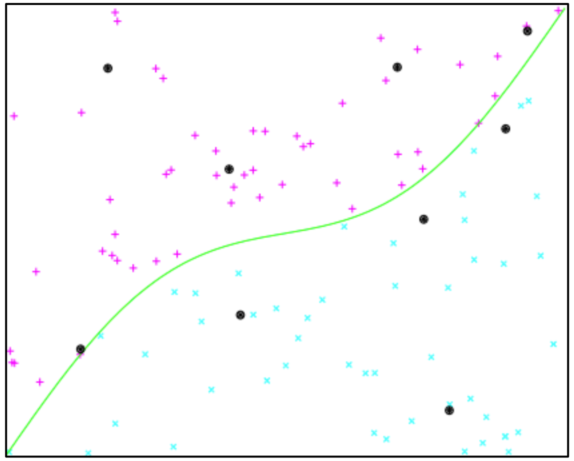
- Radial Basis Function:
  - $g(\vec{x}) = \frac{1}{Z(\vec{x})} \sum_{n=1}^{N} \phi\left(\frac{\|\vec{x}-\vec{x}_n\|}{r}\right) y_n$
  - $\phi(s)$: a monotonically decreasing function
    - Gaussian RBF (we have seen this in SVM): $\phi(s) = e^{-s}$

- This is for regression. We can take a sign and make it a classification.
- $Z(\vec{x}) = \sum_{m=1}^{N} \phi\left(\frac{\|\vec{x}-\vec{x}_m\|}{r}\right)$ is for normalization

Nonparametric and Parametric RBF

- Nonparametric RBF
  - $g(\vec{x}) = \sum_{n=1}^{N} \frac{y_n}{Z(\vec{x})} \phi\left(\frac{\|\vec{x}-\vec{x}_n\|}{r}\right)$
  - $g(\vec{x}) = \sum_{n=1}^{N} w_n(\vec{x}) \phi\left(\frac{\|\vec{x}-\vec{x}_n\|}{r}\right)$
  - The hypothesis is defined by dataset

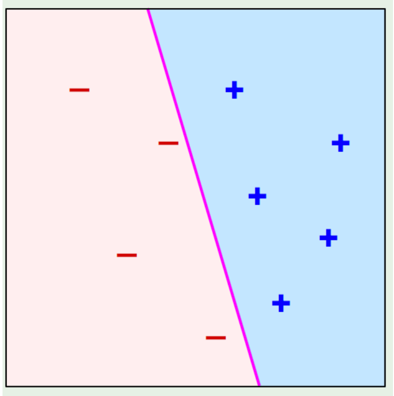
- Parametric RBF hypothesis set
  - $h(\vec{x}) = \sum_{k=1}^{K} w_k \phi\left(\frac{\|\vec{x}-\vec{\mu}_k\|}{r}\right)$
  - Find K representative points (e.g., clustering) $\vec{\mu}_1, \dots, \vec{\mu}_K$
  - Learn w_k from data



Support Vector Machines (SVM)

What Do We Know about Linear Classification?

- What we discussed so far:
  - PLA: Find a linear separator that separates the data within finite steps, if data is linear separable.
  - Pocket algorithm: empirically keep the best separator during PLA.
  - Surrogate loss: Using logistic regression for linear classification.

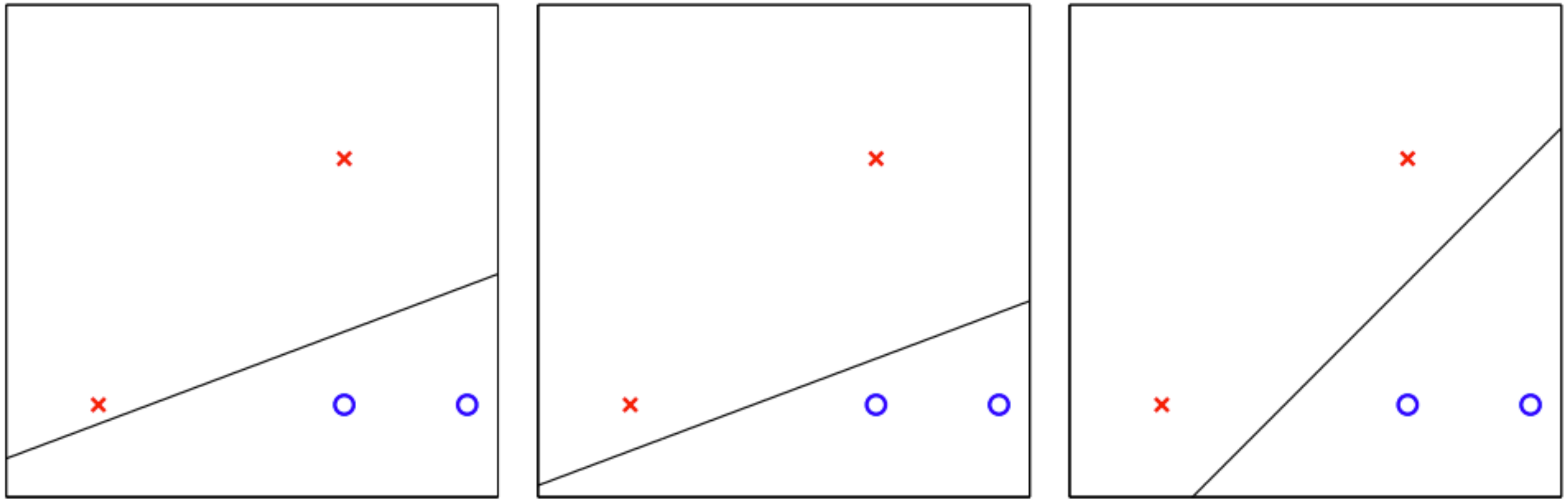


- Challenges
  - Binary classification error is hard to optimize
  - We cannot use "gradient descent" type of algorithm to minimize E_{in}

- Support vector machines (SVM) tries to look at things a bit differently.

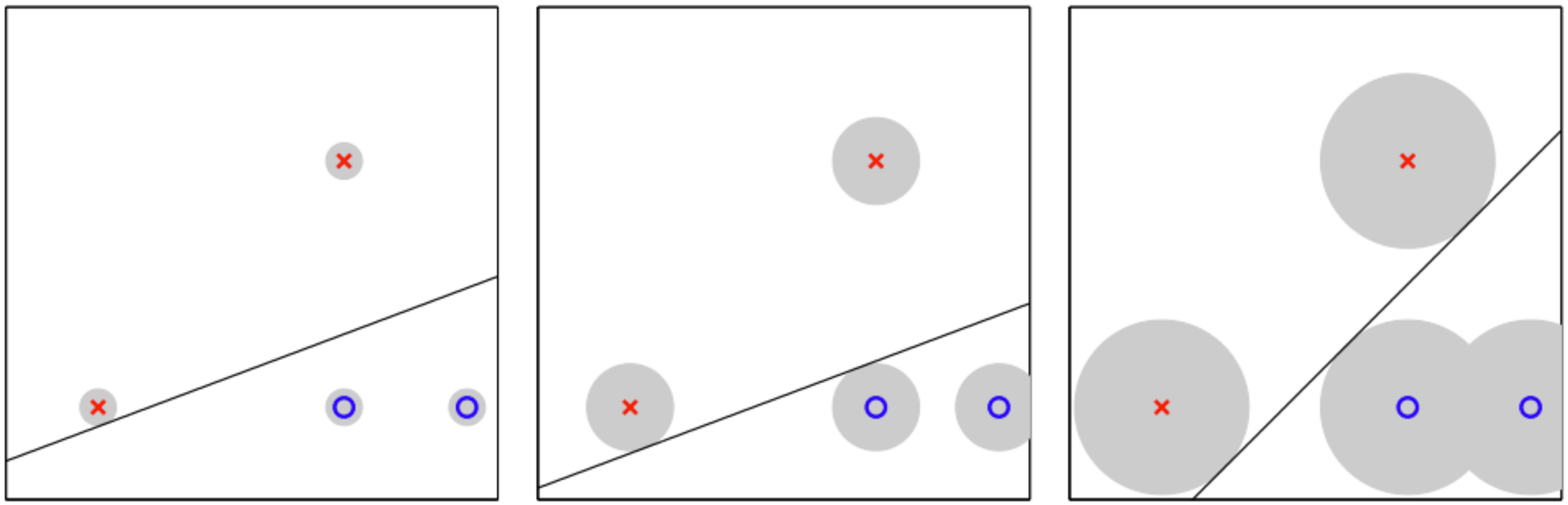
Linear Classification

- Which separator would you choose?



Linear Classification

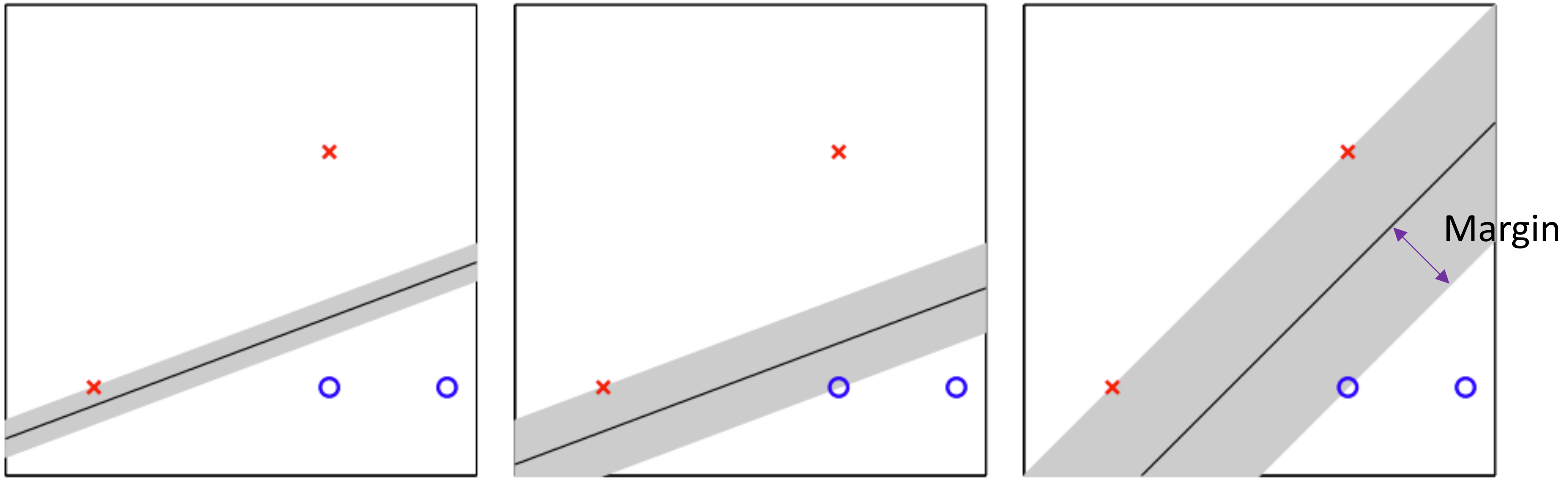
- Which separator would you choose? (Probably the right one.)



More robust to noise (e.g., measurement error of $\vec{x}$)

Linear Classification

- Which separator would you choose? (Probably the right one.)



Margin: shortest distance from the separator to the points in D

(Informal argument)

Higher margin => more “constrained” hypothesis => lower VC dimension

Support Vector Machine

- Goal:
  - Find the max-margin linear separator that separates the data
  - Recall the goal of PLA: Find the linear separator that separates the data

- Notations:

Notations we used so far:

- $\vec{x} = (x_0, x_1, \dots, x_d)$
- $\vec{w} = (w_0, w_1, \dots, w_d)$
- Linear separator

$$h(\vec{x}) = sign(\vec{w}^T \vec{x})$$

Notations we will use in SVM

- $\vec{x} = (x_1, \dots, x_d)$
- $\vec{w} = (w_1, \dots, w_d)$
- Linear separator

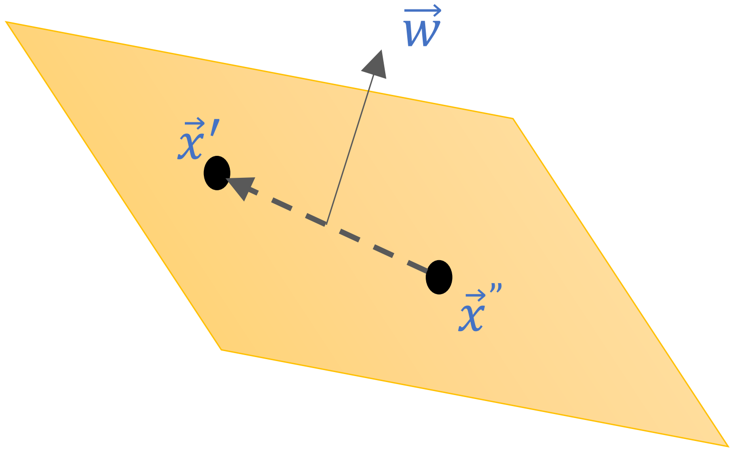
$$h(\vec{x}) = sign(\vec{w}^T \vec{x} + b)$$

Separating the bias/intercept b is important for us to characterize the margin.

We will use $(\vec{w}, b)$ to characterize the hypothesis

Relevant Review of Linear Algebra

- Claim: $\vec{w}$ is the norm vector of the hyperplane $\vec{w}^T\vec{x} + b = 0$



$\vec{w}^T\vec{x} + b = 0$

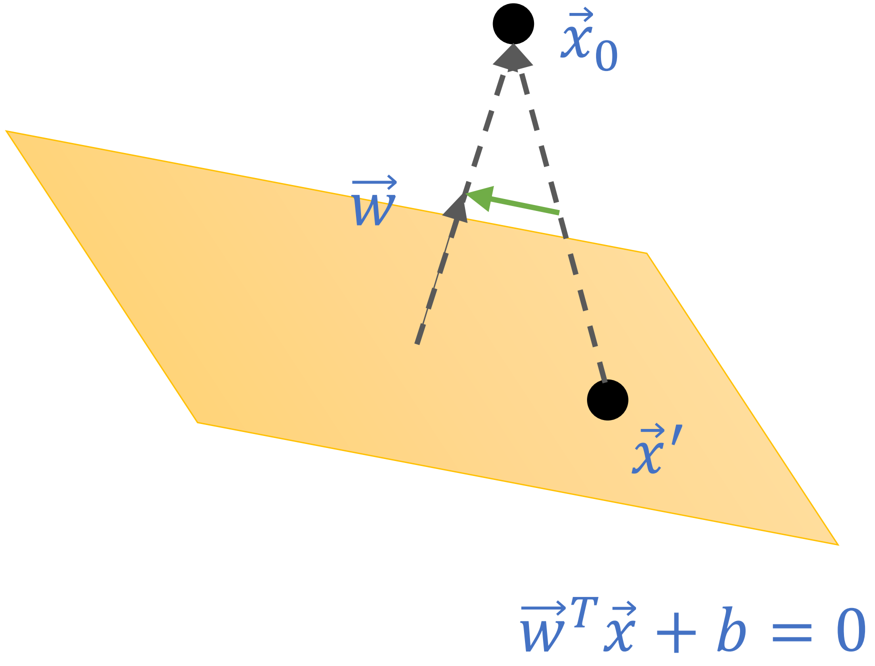
- Consider any two points $\vec{x}'$ and $\vec{x}''$ on the hyperplane
  - $\vec{w}^T\vec{x}' + b = 0$
  - $\vec{w}^T\vec{x}'' + b = 0$

- Combining the above
  - $\vec{w}^T(\vec{x}' - \vec{x}'') = 0$

- $\vec{w}$ is orthogonal to the hyperplane
- $\vec{w}$ is the norm vector of the hyperplane

Relevant Review of Linear Algebra

- What is the distance between a point $\vec{x}_0$ and a hyperplane $\vec{w}^T\vec{x} + b = 0$



- Consider an arbitrary point $\vec{x}'$ on the hyperplane

- Distance between the point $\vec{x}_0$ and the hyperplane

$$dist(\vec{x}_0, \vec{w}, b) = \left|\frac{\vec{w}^T}{\|\vec{w}\|}(\vec{x}_0 - \vec{x}')\right|$$

$$\left|\frac{1}{\|\vec{w}\|}(\vec{w}^T\vec{x}_0 - \vec{w}^T\vec{x}')\right|$$

$$\left|\frac{1}{\|\vec{w}\|}(\vec{w}^T\vec{x}_0 + b)\right|$$

Outline of Our Discussion for SVM

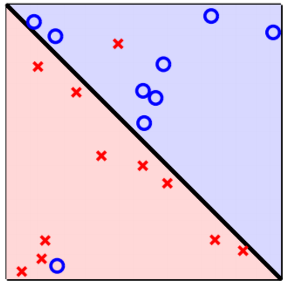
- Assume data is linearly separable
  - Formulate the hard-margin SVM

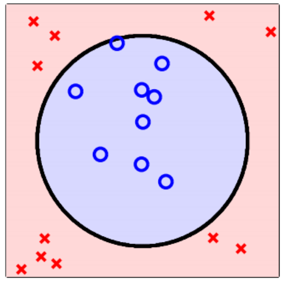
Given D, find separator $(\vec{w}, b)$ that

maximize margin$(\vec{w}, b)$
s.t. all points in D is correctly classified

Margin: shortest distance from the separator to the points in D

- When data is not linearly separable
  - Tolerate some noise
    - Soft-margin SVM
  - Nonlinear transform
    - Dual formulation and kernel tricks





Today’s Lecture

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

Let me know if you spot errors.

Hard-Margin SVM

Hard-Margin SVM

- Goal
  - Given $D = \{(\vec{x}_1, y_1), \ldots, (\vec{x}_N, y_N)\}$ that is linearly separable
  - Find separator $(\vec{w}, b)$ that (1) separates D and (2) maximizes the margin
- $(\vec{w}, b)$ separates D (making correct predictions for all points in D)
- $(\vec{w}, b)$ maximizes margin (shortest distance from the separator to points in D)

$$dist(\vec{x}_n, \vec{w}, b) = \left|\frac{1}{\|\vec{w}\|}(\vec{w}^T\vec{x}_n + b)\right|$$

$y_n \in \{-1, +1\}$ and
$y_n(\vec{w}^T\vec{x}_n + b) \geq 0$

Hard-Margin SVM

- Goal
  - Given $D = \{(\vec{x}_1, y_1), \ldots, (\vec{x}_N, y_N)\}$ that is linearly separable
  - Find separator $(\vec{w}, b)$ that (1) separates D and (2) maximizes the margin
- $(\vec{w}, b)$ separates D (making correct predictions for all points in D)
  - $y_n = sign(\vec{w}^T \vec{x}_n + b)$ for all n
  - $y_n(\vec{w}^T \vec{x}_n + b) > 0$ for all n
- $(\vec{w}, b)$ maximizes margin (shortest distance from the separator to points in D)

$$\begin{aligned} \text{margin}(\vec{w}, b) &= min_n\, dist(\vec{x}_n, \vec{w}, b) \\ &= min_n \left| \frac{1}{\|\vec{w}\|} (\vec{w}^T \vec{x}_n + b) \right| \\ &= min_n \frac{1}{\|\vec{w}\|} y_n(\vec{w}^T \vec{x}_n + b) \end{aligned}$$

$dist(\vec{x}_n, \vec{w}, b) = \left| \frac{1}{\|\vec{w}\|} (\vec{w}^T \vec{x}_n + b) \right|$

$y_n \in \{-1, +1\}$ and $y_n(\vec{w}^T \vec{x}_n + b) \geq 0$

Hard-Margin SVM

- Goal
  - Given $D = \{(\vec{x}_1, y_1), \dots, (\vec{x}_N, y_N)\}$ that is linearly separable
  - Find separator $(\vec{w}, b)$ that (1) separates D and (2) maximizes the margin

- Formulate it as a constrained optimization problem

$$
\begin{aligned}
&\text{maximize}_{\vec{w},b} \quad && \text{margin}(\vec{w}, b) \\
&\text{subject to} && y_n(\vec{w}^T \vec{x}_n + b) > 0, \forall n \\
& && \text{margin}(\vec{w}, b) = min_n \frac{1}{\|\vec{w}\|} y_n(\vec{w}^T \vec{x}_n + b)
\end{aligned}
$$

Hard-Margin SVM

- The constrained optimization problem

$$\begin{aligned} &\text{maximize}_{\vec{w},b} && \text{margin}(\vec{w}, b) \\ &\text{subject to} && y_n(\vec{w}^T\vec{x}_n + b) > 0, \forall n \\ & && \text{margin}(\vec{w}, b) = min_n \frac{1}{\|\vec{w}\|} y_n(\vec{w}^T\vec{x}_n + b) \end{aligned}$$

- Normalizing $(\vec{w}, b)$
  - Note that $\vec{w}^T\vec{x} + b = 0$ is equivalent to $c\vec{w}^T\vec{x} + cb = 0$ for any c
  - We will normalize $(\vec{w}, b)$ such that $min_n\, y_n(\vec{w}^T\vec{x}_n + b) = 1$
    - $\text{margin}(\vec{w}, b) = \frac{1}{\|\vec{w}\|}$
    - $y_n(\vec{w}^T\vec{x}_n + b) \geq 1, \forall n$

Hard-Margin SVM

- The constrained optimization problem

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

- Some final adjustments

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

Final Form of Hard-Margin SVM

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

- How to solve it?
  - Hard-margin SVM is a Quadratic Program
  - Standard form of Quadratic Program (QP)

$$\begin{aligned} \text{minimize}_{\vec{u}} \quad & \frac{1}{2}\vec{u}^T Q\vec{u} + \vec{p}^T\vec{u} \\ \text{subject to} \quad & A\vec{u} \geq \vec{c} \end{aligned}$$

  - There exist efficient QP solvers we can utilize

Linear Hard-Margin SVM with QP

1: Let $\mathbf{p} = \mathbf{0}_{d+1}$ ($(d+1)$-dimensional zero vector) and $\mathbf{c} = \mathbf{1}_N$ (N-dimensional vector of ones). Construct matrices Q and A, where

$$\mathrm{Q} = \begin{bmatrix} 0 & \mathbf{0}_d^{\mathrm{T}} \\ \mathbf{0}_d & \mathrm{I}_d \end{bmatrix}, \qquad \mathrm{A} = \underbrace{\begin{bmatrix} y_1 & -y_1\mathbf{x}_1^{\mathrm{T}}- \\ \vdots & \vdots \\ y_N & -y_N\mathbf{x}_N^{\mathrm{T}}- \end{bmatrix}}_{\text{signed data matrix}}.$$

2: Calculate $\begin{bmatrix} b^* \\ \mathbf{w}^* \end{bmatrix} = \mathbf{u}^* \leftarrow \mathsf{QP}(\mathrm{Q}, \mathbf{p}, \mathrm{A}, \mathbf{c})$.

3: Return the hypothesis $g(\mathbf{x}) = \operatorname{sign}(\mathbf{w}^{*\mathrm{T}}\mathbf{x} + b^*)$.

Some Discussion on SVM

Connection to Regularization

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

- Another way to look at SVM

minimize $\vec{w}^T\vec{w}$

subject to $E_{in}(\vec{w}) = 0$

- Weight decay regularization

minimize $E_{in}(\vec{w})$

subject to $\vec{w}^T\vec{w} \leq C$

- Maximizing margin is similar to applying regularization!

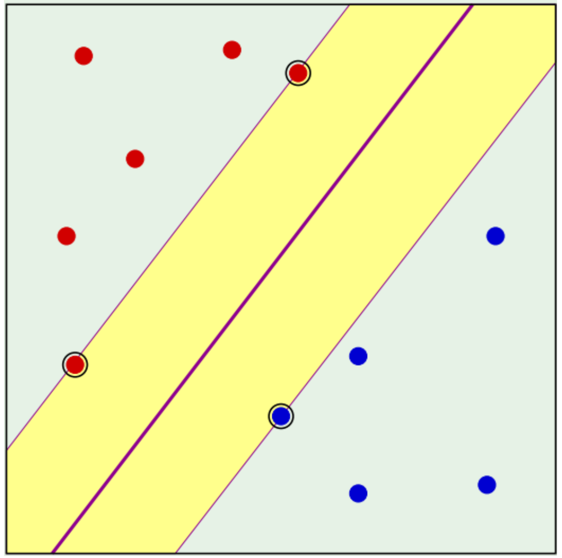
- You'll see that these two interpretations are somewhat "equivalent" when we introduce Lagrangian later this lecture or next lecture.

Support Vectors

We'll more formally define support vectors next lecture.

- We call the points closest to the separator (candidate) support vectors
  - Since they support the separator
- What are the math properties of support vectors?
  - They are the points that the equality holds in the constraints
    - If $\vec{x}_n$ is a support vector, $y_n(\vec{w}^T\vec{x}_n + b) = 1$ (the reverse might not be true)

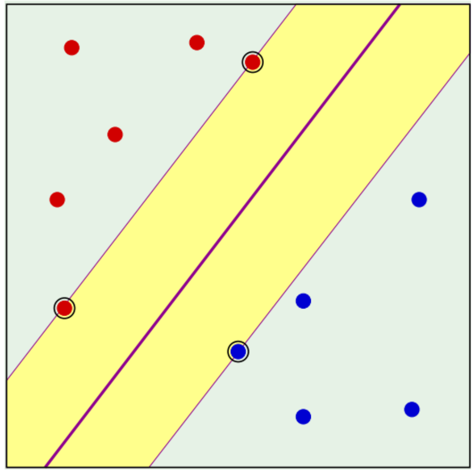
$$\begin{aligned} &\text{minimize}_{\vec{w},b} && \frac{1}{2}\vec{w}^T\vec{w} \\ &\text{subject to} && y_n(\vec{w}^T\vec{x}_n + b) \geq 1, \forall n \end{aligned}$$



  - Removing the non-support vectors will not impact the linear separator

Leave-One-Out Cross Validation (LOOCV)

- Two things we know so far
  - Removing non-support vectors will not impact the separator
  - LOOCV error (when not used for model selection) is an unbiased estimate of $E_{out}(N-1)$ (E_{out} when trained on $N-1$ points)

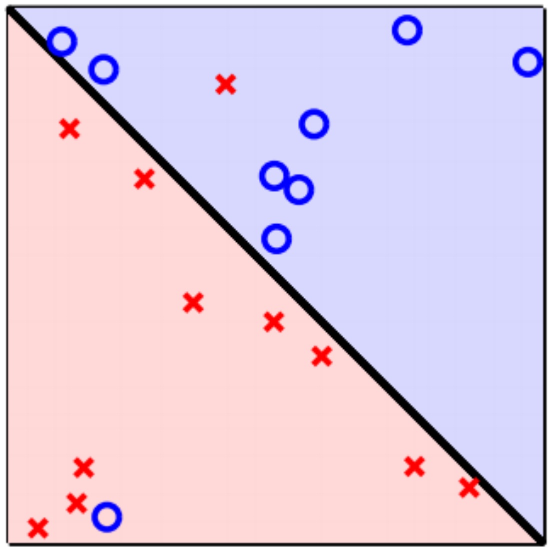


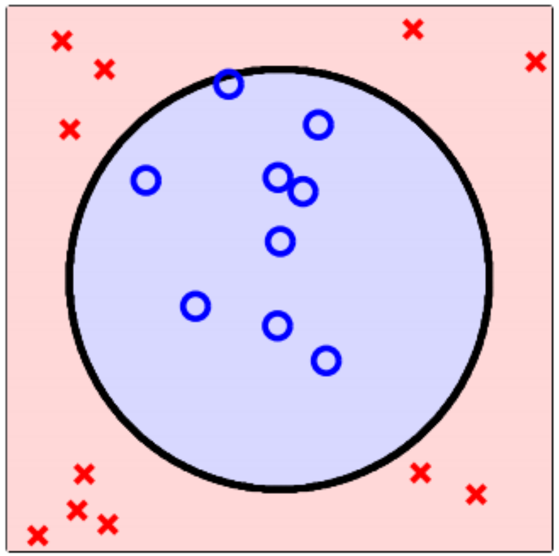
- What's the upper bound of LOOCV error for SVM?
  - $E_{LOOCV} \leq \frac{\#\,\text{support vectors}}{N}$
- Note that we know # support vectors after training
  - Count # points that satisfy $y_n(\vec{w}^T\vec{x}_n + b) = 1$
- Another method to estimate/bound E_{out} (counting # support vectors)

What if Data is Not Linearly Separable

Non-Separable Data

- Two scenarios





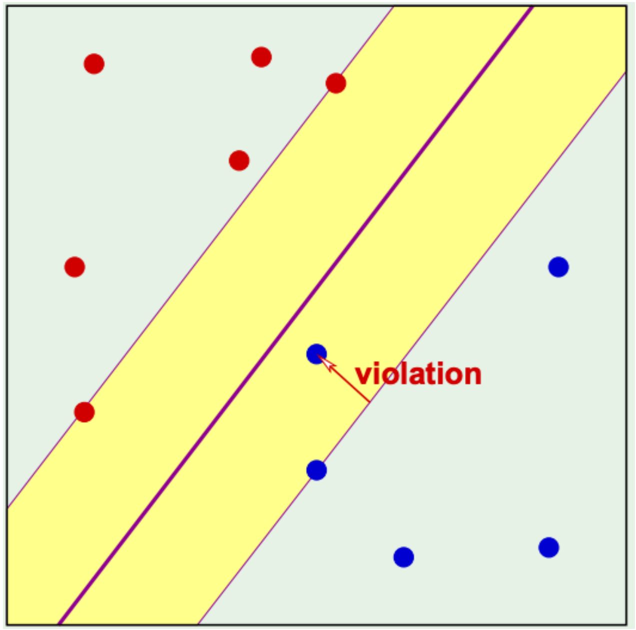
- Tolerate some noise
  - Soft-Margin SVM
- Nonlinear transform
  - Dual formulation and kernel tricks

Soft-Margin SVM

- Intuition: We want to tolerate small noises when maintaining large margin

- For each point $(\vec{x}_n, y_n)$, we allow a deviation $\xi_n \geq 0$
  - Instead of requiring $y_n(\vec{w}^T \vec{x}_n + b) \geq 1$
  - The constraint becomes

$$y_n(\vec{w}^T \vec{x}_n + b) \geq 1 - \xi_n$$

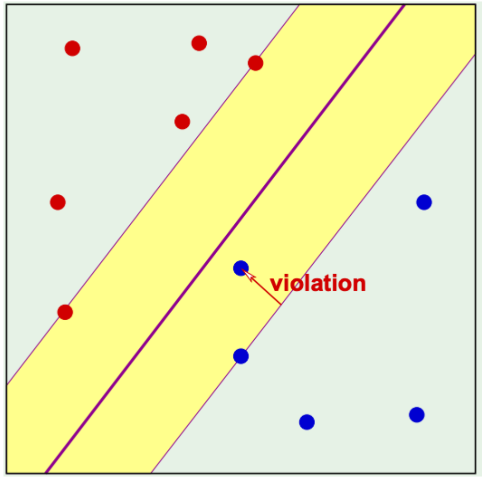


- We add a penalty for each deviation
  - Total penalty $C \sum_{n=1}^{N} \xi_n$

Soft-Margin SVM

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

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

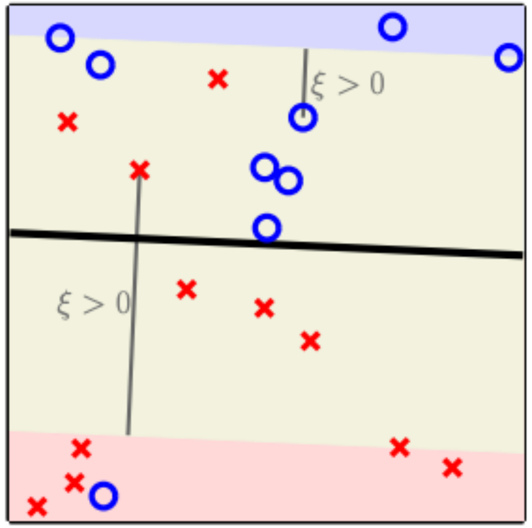


Remarks:

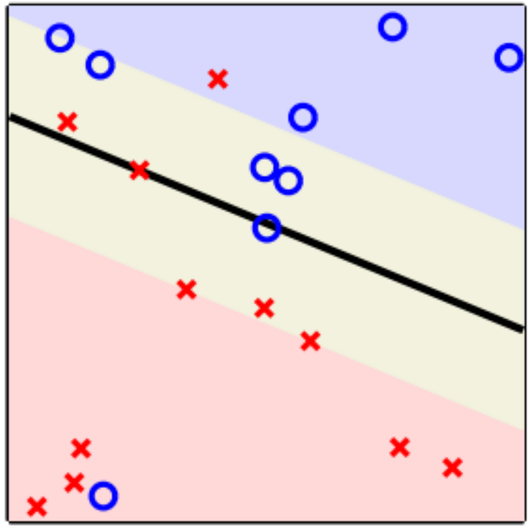
- C is a hyper-parameter we can choose, e.g., using validation
- Soft-margin SVM is still a Quadratic Program, with efficient solvers

Impacts of C in Soft-Margin SVM

- What happens when C is larger
  - less tolerate to noise, having smaller margin



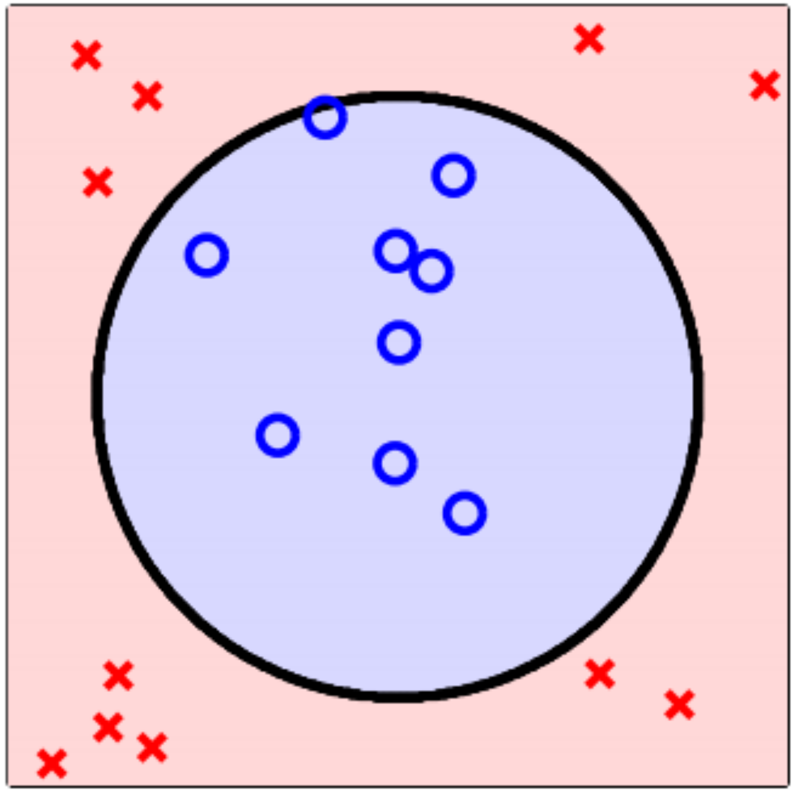
$C = 1$



$C = 500$

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

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

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

- Consider hard-margin SVM in the $\vec{z}$ space

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

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

- There exists a corresponding dual formulation (more next lecture)

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

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

- Why dual
  - The optimal primal is the same as the optimal dual
  - We can infer the optimal primal solutions from the optimal dual solutions

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

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

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

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

- 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

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

[Safe to Skip for the Exam]

Lagrangian

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

- 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

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

[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

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

Reminders of definitions:

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

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

- 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]

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]

Connection to Weight-Decay Regularization

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

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

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

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

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

Exercise:
Remember the weight-decay regularization:

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

And the hard-margin SVM

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

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

[Safe to Skip for the Exam]

Why Talk about Dual

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

- Consider hard-margin SVM in the $\vec{z}$ space

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

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

- There exists a corresponding dual formulation (more next lecture)

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

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

- Why dual
  - The optimal primal is the same as the optimal dual
  - We can infer the optimal primal solutions from the optimal dual solutions