CS 760: Machine Learning
SVMs and Kernels

Josiah Hanna

University of Wisconsin-Madison

November 21, 2023

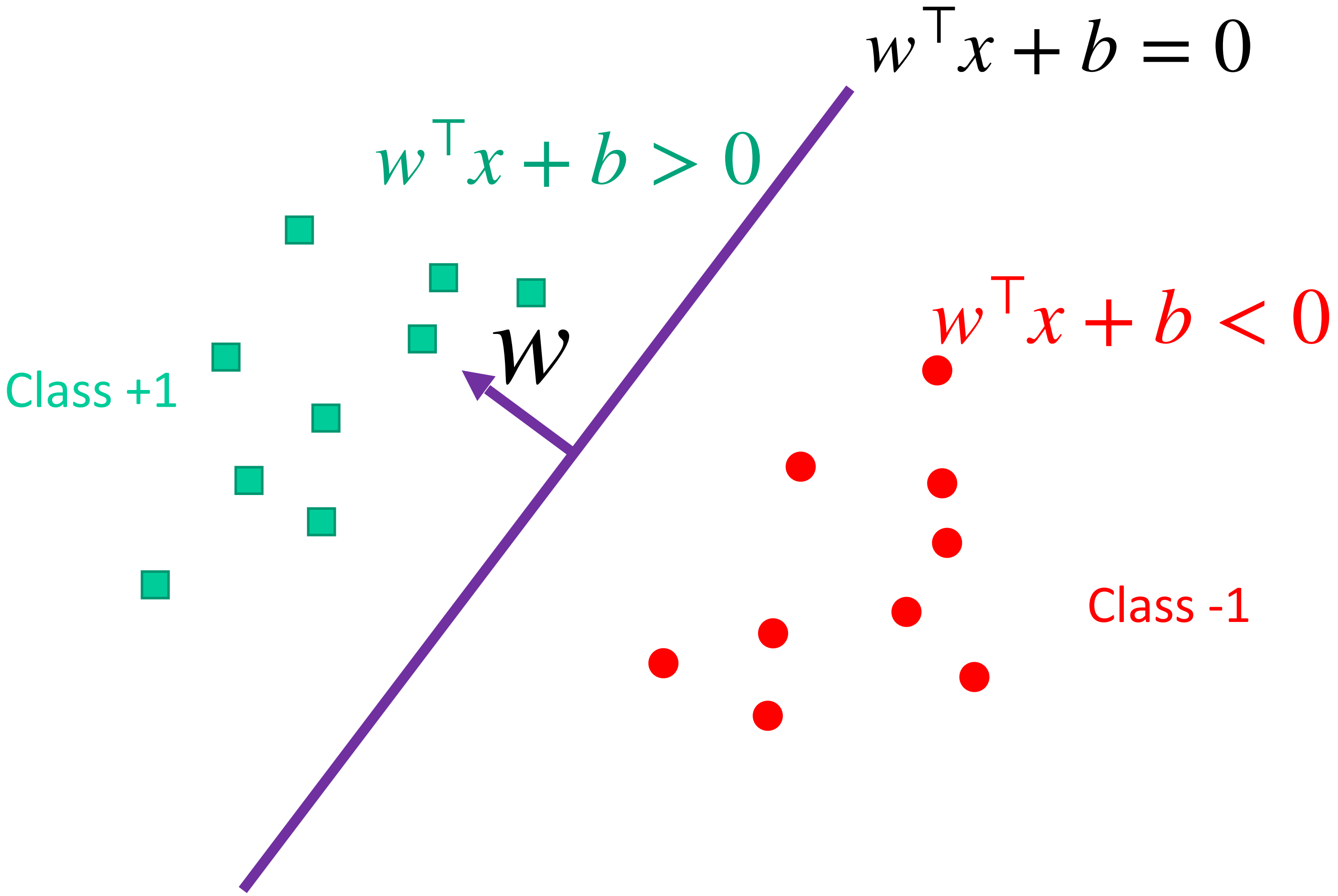
Outline

- **Support Vector Machines (SVMs)**
  - margins, training objectives
- **Dual Formulation**
  - Lagrangian, primal and dual problems
- **Kernels**
  - Feature maps, kernel trick, conditions

Outline

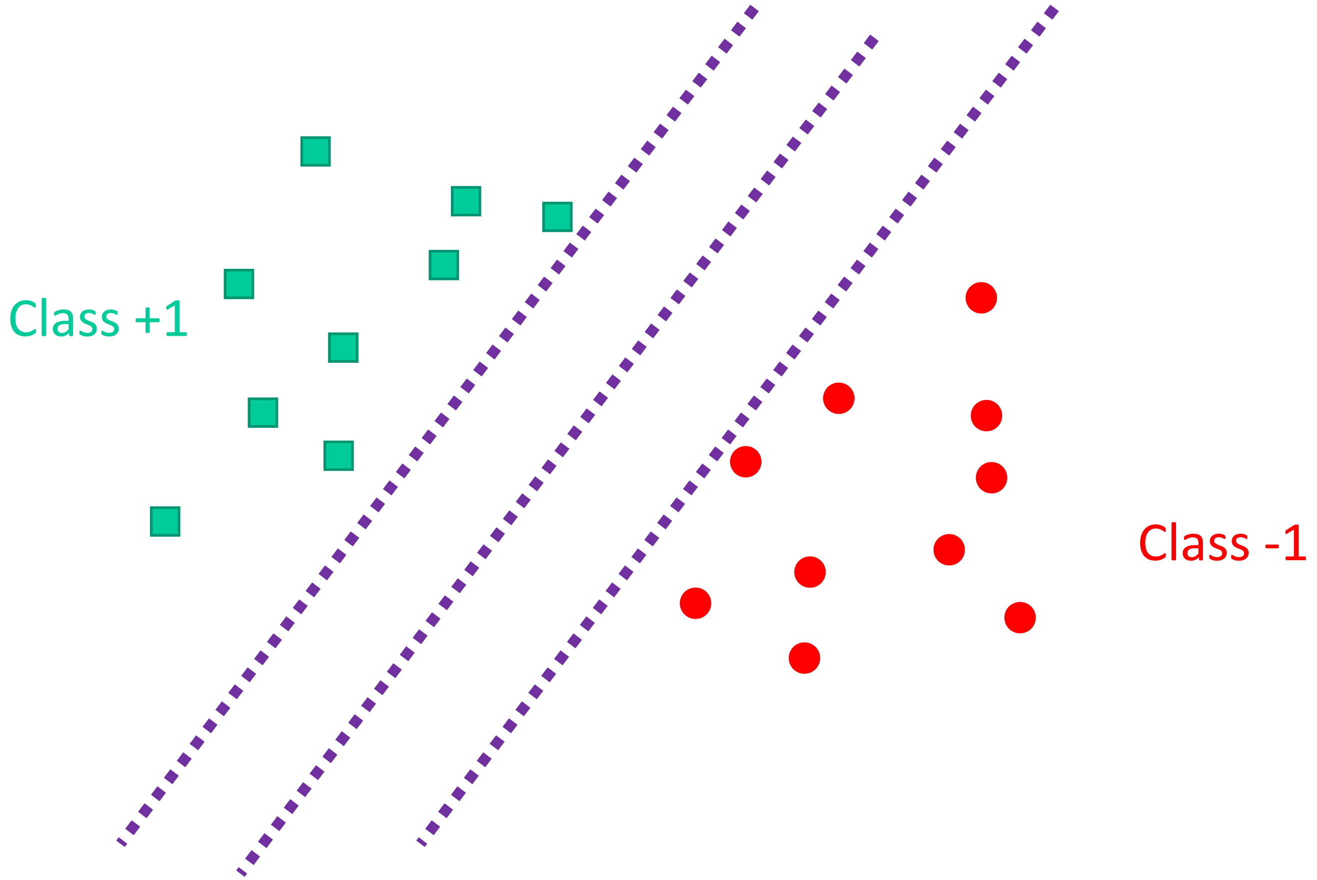
- **Support Vector Machines (SVMs)**
  - margins, training objectives
- **Dual Formulation**
  - Lagrangian, primal and dual problems
- **Kernels**
  - Feature maps, kernel trick, conditions

Linear classification revisited



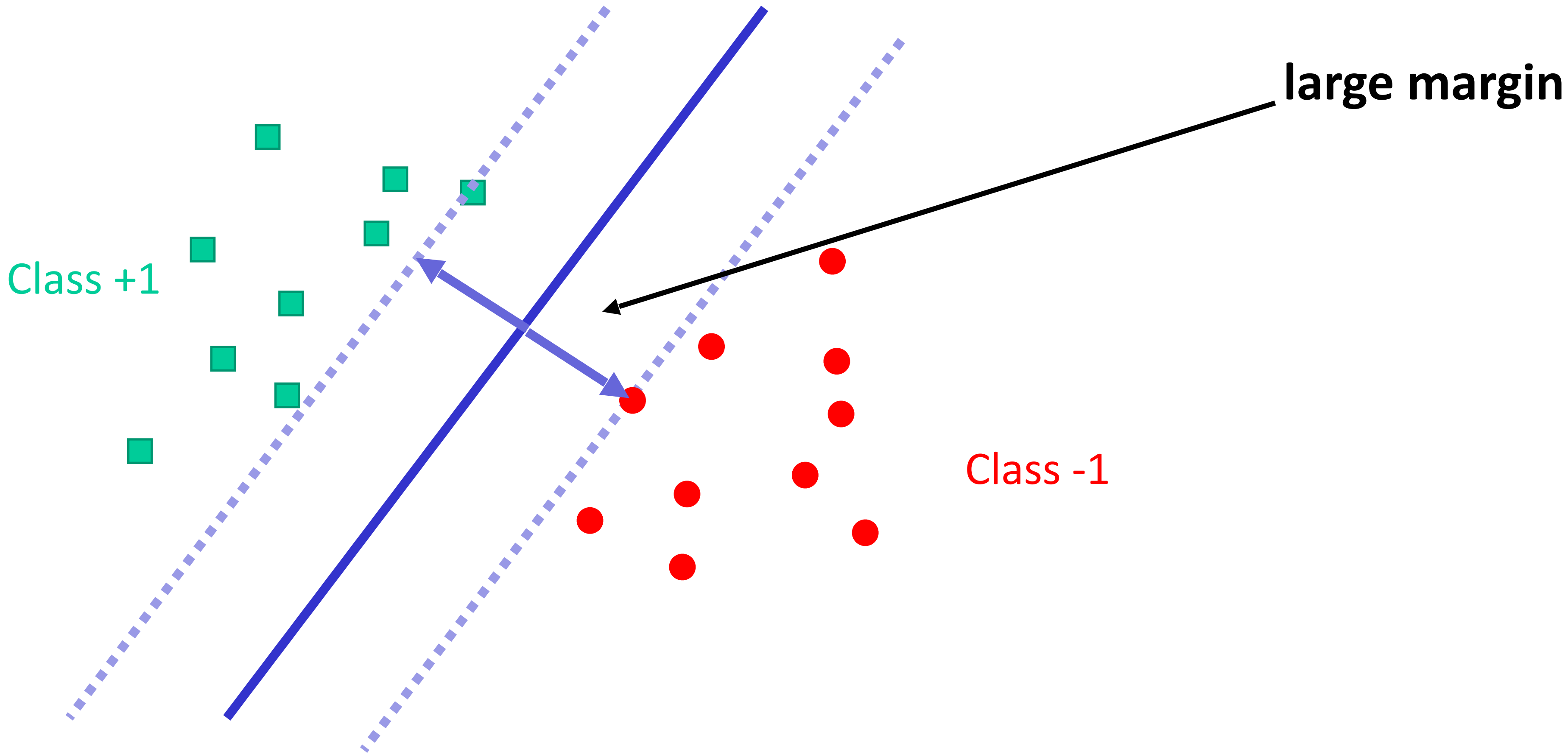
Linear classification revisited

- Which classifier is better for generalization?



Linear classification revisited

- Intuitively, expect a **large margin** to generalize better.



Both direction and location of hyperplane affects the margin.

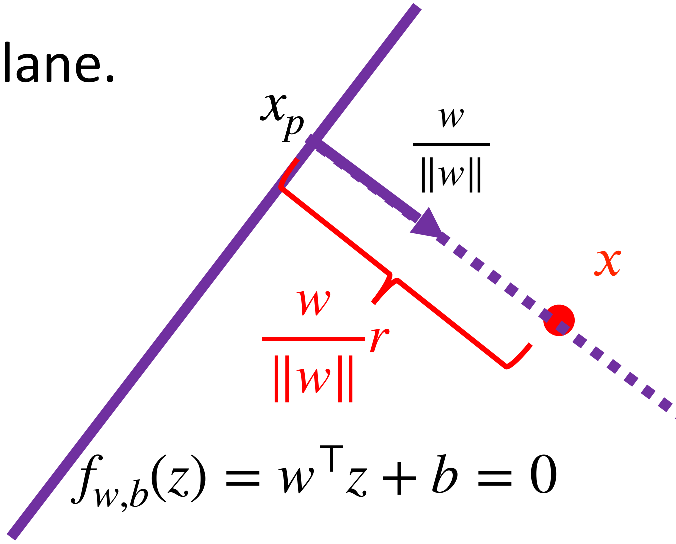
Distance to a hyperplane

x has distance $\dfrac{|f_{w,b}(x)|}{\|w\|}$ to the hyperplane $f_{w,b}(z) = w^\top z + b = 0$

Proof (on your own): Let x_p denote the projection of x onto the hyperplane.

Then, we can write $x = x_p + r\dfrac{w}{\|w\|}$ for some $r \in \mathbb{R}$ (Why?).

Hence, the distance to the hyperplane is $|r|$ (Why?).



We have $f_{w,b}(x) = w^\top x + b = \underbrace{w^\top x_p + b}_{=0} + r\dfrac{w^\top w}{\|w\|} = r\|w\|.$

Therefore, $|r| = \dfrac{|f_{w,b}(x)|}{\|w\|} = \dfrac{|w^\top x + b|}{\|w\|}$

Support Vector Machines

- We wish to maximize the “minimum margin” over all points.
- The minimum margin over all training data points and margin w:

Using our result ⟶ $$\gamma(w,b) = \min_i \frac{|f_{w,b}(x_i)|}{||w||}$$

- We can write it equivalently as:

$$\gamma(w,b) = \min_i \frac{y_i f_{w,b}(x_i)}{||w||} \qquad y_i \in \{-1,1\}$$

- If $f_{w,b}$ incorrect on some x_i, the margin is **negative**

Support Vector Machines: Candidate Goal

- Assume data is linearly separable for now.

- One way: maximize margin over all training data points:

$$\max_{w,b} \gamma(w,b) = \max_{w,b} \min_{i} \frac{y_i f_{w,b}(x_i)}{||w||} = \max_{w,b} \min_{i} \frac{y_i w^\top x_i + b}{||w||}$$

Minimax Optimization may be difficult to solve!

SVM: Simplified Goal

- Observation: when (w, b) scaled by a factor $c > 0$, the margin is unchanged

$$\frac{y_i(cw^T x_i + cb)}{||cw||} = \frac{y_i(w^T x_i + b)}{||w||}$$

- Let us consider a fixed scale such that

$$y_{i*}\left(w^T x_{i*} + b\right) = 1$$

where x_{i*} is the point closest to the hyperplane

SVM: Simplified Goal

- Let us consider a fixed scale such that

$$y_{i^*}\left(w^T x_{i^*} + b\right) = 1$$

where x_{i^*} is the point closest to the hyperplane

- Now we have for all data

$$y_i\left(w^T x_i + b\right) \geq 1$$

and at least for one i the equality holds

- Then the margin over all training points is $\dfrac{|w^\top x_i + b|}{\|w\|} = \dfrac{1}{\|w\|}$

Writing the SVM as an optimization problem

- Optimization problem can be written as

$$\max_{w,b} \frac{1}{\|w\|_2} \qquad \text{subject to } y_i(w^\top x_i + b) \geq 1 \; \forall i.$$

- Instead we will write this as,

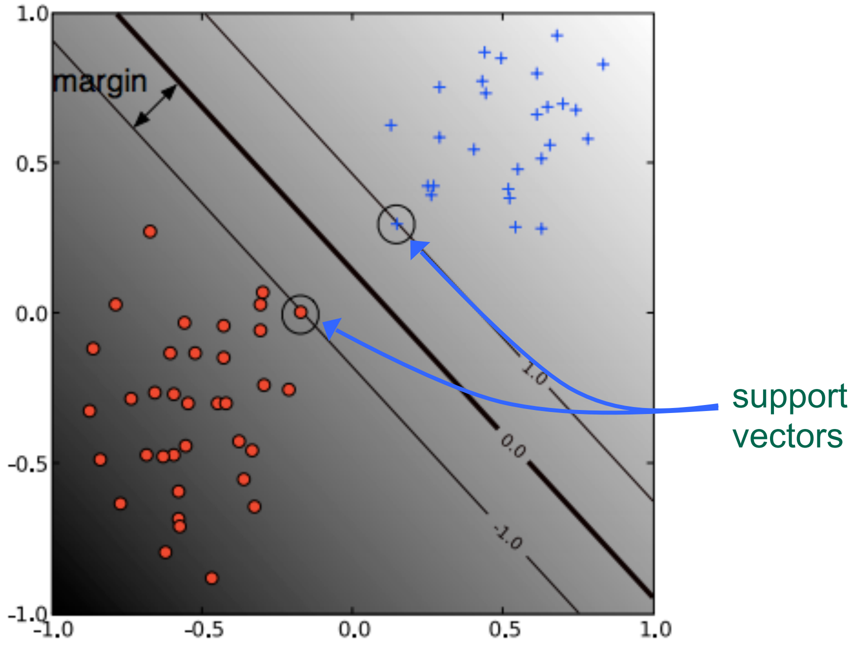
$$\min_{w,b} \frac{1}{2}\|w\|_2^2$$

$$\text{subject to } y_i(w^\top x_i + b) \geq 1 \; \forall i$$

- Why?
    - This is a Quadratic program (a type of convex program). Many efficient solvers**!**
    - Allows us to apply the kernel trick for nonlinear classification (coming up)

SVM: Support Vectors

- Instances where inequality is tight are the ***support vectors***
  - Lie on the margin boundary
- Solution does not change if we delete other instances!



SVM: Soft Margin

What if our data isn't linearly separable?

- Can adjust our approach by using *slack variables* (denoted by ζ_i) to tolerate errors

$$\min_{w,b,\zeta_i} \frac{1}{2}||w||^2 + C\sum_i \zeta_i$$

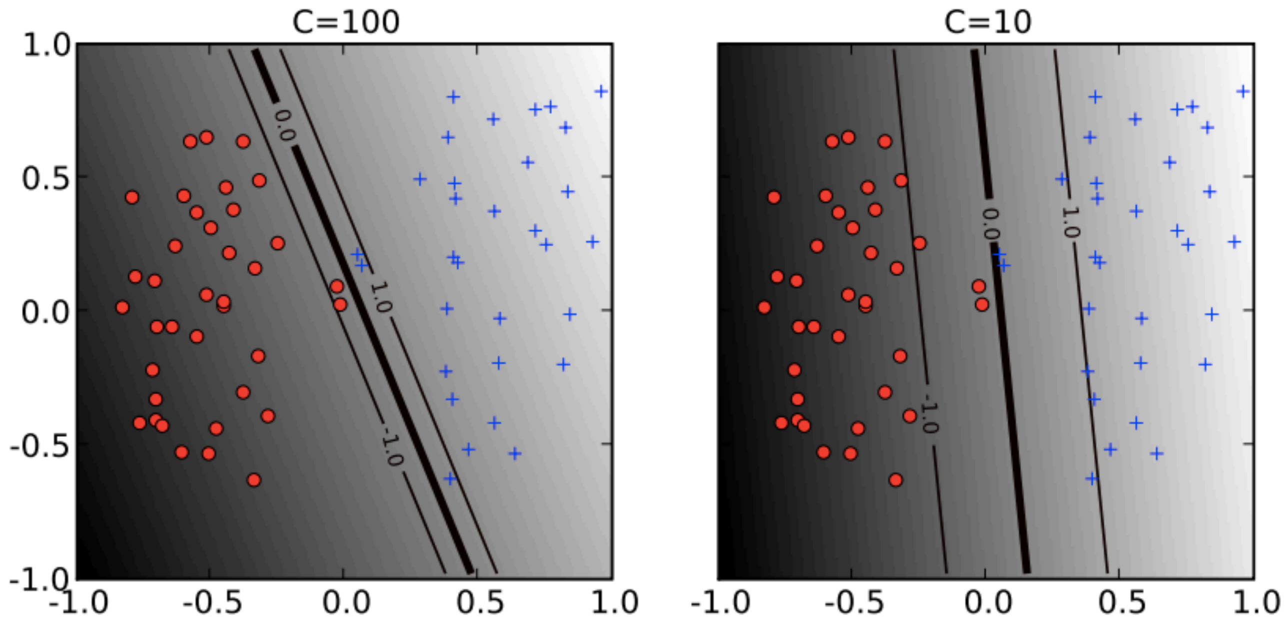
$$y_i\left(w^T x_i + b\right) \geq 1 - \zeta_i, \zeta_i \geq 0, \ \forall i$$

- C determines the relative importance of maximizing margin vs. minimizing slack

SVM: Soft Margin

$$\min_{w,b,\zeta_i} \frac{1}{2}\left|\left|w\right|\right|^2 + C\sum_i \zeta_i$$

$$y_i\left(w^T x_i + b\right) \geq 1 - \zeta_i, \zeta_i \geq 0, \ \ \forall i$$



Ben-Hur & Weston, *Methods in Molecular Biology* 2010

Outline

- **Support Vector Machines (SVMs)**
  - margins, training objectives
- **Dual Formulation**
  - Lagrangian, primal and dual problems
- **Kernels**
  - Feature maps, kernel trick, conditions

Constrained Optimization

- Consider the optimization problem:

$$\min_{w} f(w)$$ Objective

$$g_i(w) \leq 0,\ \forall 1 \leq i \leq k$$ Constraints

$$h_j(w) = 0,\ \forall 1 \leq j \leq l$$

- Generalized Lagrangian:

$$\mathscr{L}\left(w, \boldsymbol{\alpha},\ \boldsymbol{\beta}\right) = f(w) + \sum_i \alpha_i g_i(w) + \sum_j \beta_j h_j(w)$$

where α_i, β_j's are called **Lagrange multipliers**

Lagrangian

- Form the quantity:

$$\theta_P(w) := \max_{\boldsymbol{\alpha},\, \boldsymbol{\beta}: \alpha_i \geq 0} \mathcal{L}\left(w, \boldsymbol{\alpha},\ \boldsymbol{\beta}\right)$$

$$:= \max_{\boldsymbol{\alpha},\, \boldsymbol{\beta}: \alpha_i \geq 0} f(w) + \sum_i \alpha_i g_i(w) + \sum_j \beta_j h_j(w)$$

$g_i(w) \leq 0,\ \forall 1 \leq i \leq k$

$h_j(w) = 0,\ \forall 1 \leq j \leq l$

- Note:

$$\theta_P(w) = \begin{cases} f(w), & \text{if } w \text{ satisfies all the constraints} \\ +\infty, & \text{if } w \text{ does not satisfy the constraints} \end{cases}$$

Lagrangian

- Form the quantity:

$$\theta_P(w) := \max_{\boldsymbol{\alpha},\ \boldsymbol{\beta}:\alpha_i \geq 0} \mathscr{L}\left(w, \boldsymbol{\alpha},\ \boldsymbol{\beta}\right)$$

- Note:

$$\theta_P(w) = \begin{cases} f(w), & \text{if } w \text{ satisfies all the constraints} \\ +\infty, & \text{if } w \text{ does not satisfy the constraints} \end{cases}$$

- Minimizing $f(w)$ with constraints is the same as minimizing $\theta_P(w)$

$$\min_w f(w) = \min_w \theta_P(w) = \min_w \max_{\boldsymbol{\alpha},\ \boldsymbol{\beta}:\alpha_i \geq 0} \mathscr{L}\left(w, \boldsymbol{\alpha},\ \boldsymbol{\beta}\right)$$

Duality

The primal problem

$$p^* := \min_{w} f(w) = \min_{w} \max_{\boldsymbol{\alpha},\ \boldsymbol{\beta}: \alpha_i \geq 0} \mathscr{L}\left(w, \boldsymbol{\alpha},\ \boldsymbol{\beta}\right)$$

The dual problem

$$d^* := \max_{\boldsymbol{\alpha},\ \boldsymbol{\beta}: \alpha_i \geq 0} \min_{w} \mathscr{L}\left(w, \boldsymbol{\alpha},\ \boldsymbol{\beta}\right)$$

.

- Always true: $d^* \leq p^*$

Duality Gap

- Always true: $d^* \leq p^*$

If **actual equality**, could solve dual instead of primal… when?

- Under some assumptions (ex: Slater's conditions), there exists $\left(w^*, \boldsymbol{\alpha}^*,\ \boldsymbol{\beta}^*\right)$ such that

$$d^* = \mathscr{L}\left(w^*, \boldsymbol{\alpha}^*,\ \boldsymbol{\beta}^*\right) = p^*$$

- $\left(w^*, \boldsymbol{\alpha}^*,\ \boldsymbol{\beta}^*\right)$ satisfy Karush-Kuhn-Tucker (KKT) conditions:

$$\frac{\partial \mathscr{L}}{\partial w_i} = 0, \quad \alpha_i g_i(w) = 0\,,\ \ g_i(w) \leq 0,\ \ h_j(w) = 0,\ \ \alpha_i \geq 0$$

Alternative optimization procedure for SVMs

- Recall our "primal" SVM optimization problem:

$$\min_{w,b} \frac{1}{2}\left|\left|w\right|\right|^2$$

$$y_i\left(w^T x_i + b\right) \geq 1,\ \forall i$$

- **Dual:** Write out the Lagrangian, maximize w.r.t w, b, and then solve the maximization problem!

$$\mathscr{L}(w, b, \boldsymbol{\alpha}) = \frac{1}{2}\left|\left|w\right|\right|^2 - \sum_i \alpha_i[y_i\left(w^T x_i + b\right) - 1]$$

SVM: Optimization

- First, minimimize $\mathcal{L}(w, b, \alpha)$ w.r.t w, b:

$$\frac{\partial \mathcal{L}}{\partial w} = 0, \rightarrow \quad w = \sum_i \alpha_i y_i x_i \text{ (1)}$$

$$\frac{\partial \mathcal{L}}{\partial b} = 0, \rightarrow \quad 0 = \sum_i \alpha_i y_i \qquad \text{(2)}$$

$$\mathcal{L}(w, b, \boldsymbol{\alpha}) = \frac{1}{2}\left|\,|w|\,\right|^2 - \sum_i \alpha_i [y_i\left(w^T x_i + b\right) - 1]$$

- Plug into $\mathcal{L}$:

$$\mathcal{L}(w, b, \boldsymbol{\alpha}) = \sum_i \alpha_i - \frac{1}{2}\sum_{ij} \alpha_i \alpha_j y_i y_j x_i^T x_j \text{ (3)}$$

combined with $0 = \sum_i \alpha_i y_i, \ \alpha_i \geq 0$

(From solution for b (above) and KKT Conditions)

SVM: Training with dual version

- Can write as:

$$\max_{\boldsymbol{\alpha}} \sum_i \alpha_i - \frac{1}{2} \sum_{ij} \alpha_i \alpha_j y_i y_j x_i^T x_j$$

$$\sum_i \alpha_i y_i = 0,\ \alpha_i \geq 0$$

Note: training only deals with data via inner products $x_i^\top x_j$

SVM: Testing with Dual Version

- Suppose the solution is $\alpha^\star$. How do we recover our classifier?
- Optimal $w^\star$ is: $w^\star = \sum_i \alpha_i^\star y_i x_i$ (from a couple of slides before)
- Optimal $b^\star$ is: (do at home, hint: look at the primal problem)

$$b^\star = \frac{-1}{2}\left(\max_{j, y_j=-1} (w^\star)^\top x_j + \min_{j, y_j=+1} (w^\star)^\top x_j\right) = \frac{-1}{2}\left(\max_{j, y_j=-1} \sum_i \alpha^\star y_i x_i^\top x_j + \min_{j, y_j=+1} \sum_i \alpha^\star y_i x_i^\top x_j\right)$$

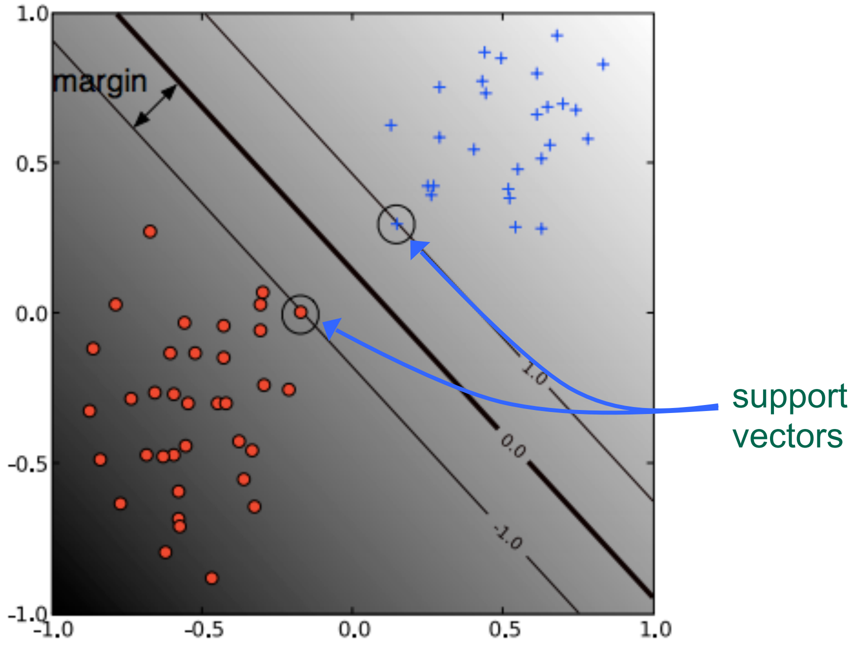
- To compute a prediction at x_{test}, we check if

$$(w^\star)^\top x_{\text{test}} + b^\star = \sum_i \alpha_i^\star y_i x_i^\top x_{\text{test}} + b^\star \geq 0$$

- Note: testing only deals with data via inner products $x_i^\top x_{\text{test}}$ **(and $x_i^\top x_j$).**

SVM: Support Vectors

- Those instances with $\alpha_i > 0$ are called ***support vectors***
  - Lie on the margin boundary
- Solution is a linear combination of support vectors!
- Solution does not change if we delete instances with $\alpha_i = 0$



Break & Quiz

Quiz

Which of the following statements are true?

A. The solution of an SVM will always change if we remove some instances from the training set.

B. If we know that our data is linearly separable, then it does not make sense to use slack variables.

C. If you only had access to the labels $\{y_i\}_i$ and the inner products $\{x_i^\top x_j\}_{i,j}$, we can still find the solution to the SVM.

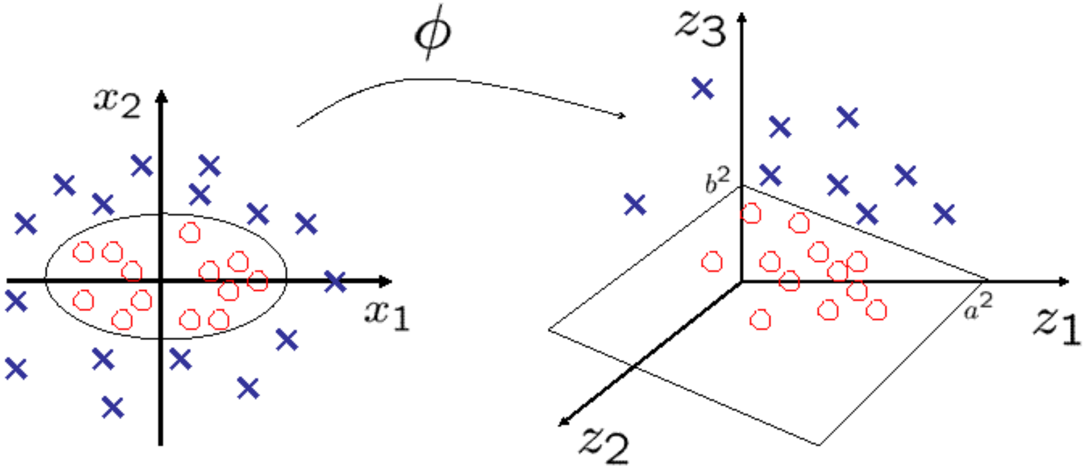
A: False, B: False, C: True

Outline

- **Support Vector Machines (SVMs)**
  - margins, training objectives
- **Dual Formulation**
  - Lagrangian, primal and dual problems
- **Kernels**
  - Feature maps, kernel trick

Feature Maps

- Can take a set of features and map them into another
  - Do this to construct non-linear features (recall basis functions from linear models).
  - Then use non-linear features in a linear classifier to learn non-linear decision boundaries.



$$\phi : (x_1, x_2) \longrightarrow (x_1^2, \sqrt{2}x_1x_2, x_2^2)$$

Feature Maps and SVMs

Goal: use feature space $\left\{ \phi\left(x_i\right) \right\}$ in a linear classifier…

- Downside: dimension might be high (possibly infinite)
- So we do not want to write down $\phi\left(x_i\right) = [0.2, 0.3, \dots]$

Recall our SVM dual form:

- Training and testing only rely on inner products $x_i^T x_j$

$$\mathscr{L}(w, b, \boldsymbol{\alpha}) = \sum_i \alpha_i - \frac{1}{2} \sum_{ij} \alpha_i \alpha_j y_i y_j x_i^T x_j \quad \text{s.t} \sum_i \alpha_i y_i = 0,\ \alpha_i \geq 0$$

Kernel Trick

- Using SVM on the feature space $\{\phi(x_i)\}$: only need $\phi(x_i)^T\phi(x_j)$

- Therefore, no need to design $\phi(\cdot)$, only need to design

$$\underbrace{k(x_i, x_j)}_{\text{Kernel}} = \underbrace{\phi(x_i)^T\phi(x_j)}_{\text{Feature Maps}}$$

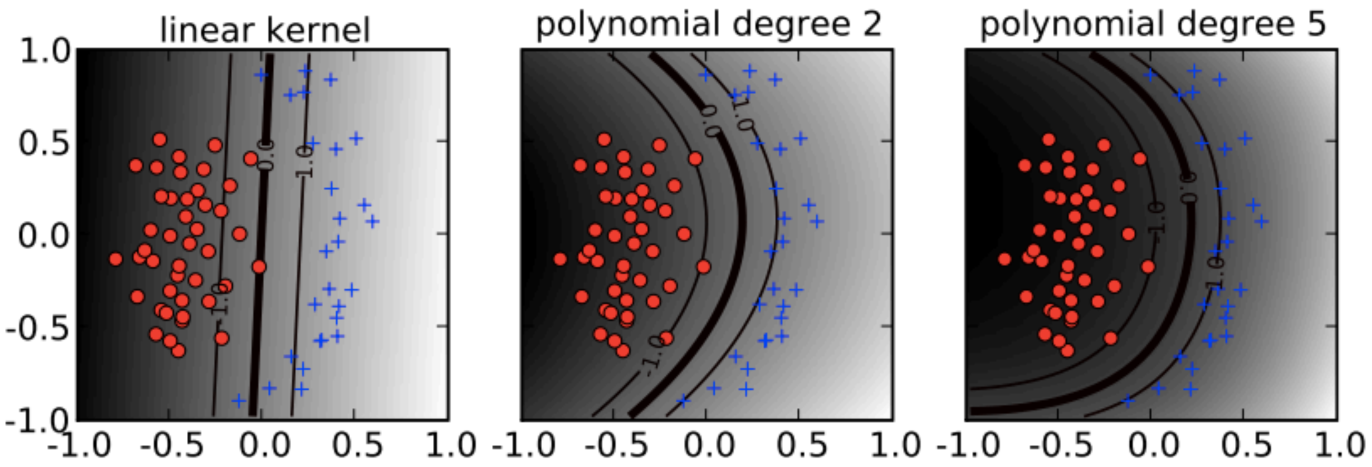
Kernel Types: Polynomial

- Fix degree d and constant c:

$$k(x, x') = \left(x^T x' + c\right)^d$$

- What are $\phi(x)$?
- Expand the expression to get $\phi(x)$

$$\forall \mathbf{x}, \mathbf{x}' \in \mathbb{R}^2, \quad K(\mathbf{x}, \mathbf{x}') = (x_1 x_1' + x_2 x_2' + c)^2 = \begin{bmatrix} x_1^2 \\ x_2^2 \\ \sqrt{2}\, x_1 x_2 \\ \sqrt{2c}\, x_1 \\ \sqrt{2c}\, x_2 \\ c \end{bmatrix} \cdot \begin{bmatrix} {x'}_1^2 \\ {x'}_2^2 \\ \sqrt{2}\, x_1' x_2' \\ \sqrt{2c}\, x_1' \\ \sqrt{2c}\, x_2' \\ c \end{bmatrix}$$



Ben-Hur & Weston, *Methods in Molecular Biology* 2010

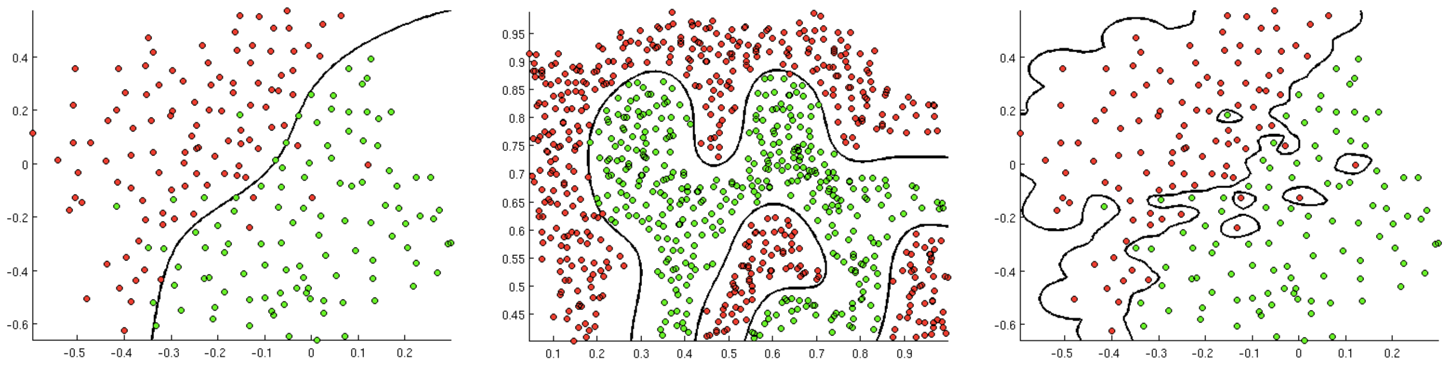
Kernel Types: Gaussian/RBF

- Fix γ:

$$k(x, x') = \exp(-\gamma \|x - x'\|^2)$$

- With RBF kernels, you are projecting to an infinite dimensional space

$\gamma = 10$ $\gamma = 100$ $\gamma = 1000$



SVM: Training dual problem with kernels

$$\max_{\alpha} \sum_i \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j k(x_i, x_j)$$

$$\sum_i \alpha_i y_i = 0,\ \alpha_i \geq 0$$

Simply replaced $x_i^\top x_j$ in the linear SVM with $k(x_i, x_j)$.

Can do so with slack variables as well.

SVM Summary

1. Understand maximum margin classification. Why do we write this as:

   $$\max_{w,b} \|w\|_2^{-1} \quad \left(\text{or } \min_{w,b} \frac{1}{2}\|w\|_2^2\right) \quad \text{subject to } y_i(w^\top x_i + b) \geq 1 \ \forall i.$$

2. Going from primal to dual formulation
3. Kernel trick enables SVM to represent complex non-linear decision boundaries.

Break & Quiz

Quiz

Which of the following statements are true?

A. SVMs with nonlinear kernels implicitly transform the low dimensional features to a high dimensional space and then performing linear classification in that space.

B. The “Kernel trick” refers to computing this transformation and then applying the dot product between the transformed points.

A: True, B: False

Quiz

Consider the kernel $k(x, x') = (xx' + 1)^3$ for $x \in \mathbb{R}$. Give an explicit expression for a feature map ϕ such that $\phi(x)^\top \phi(x') = k(x, x')$.

1. $\phi(x)^\top = [x^3, x^2, x, 1]$
2. $\phi(x)^\top = [x^3, \sqrt{3}x^2, \sqrt{3}x, 1]$
3. $\phi(x)^\top = [x^3, \sqrt{3}x^2, x, \sqrt{3}]$
4. $\phi(x)^\top = [x^3, \sqrt{3}x^2, \sqrt{3}x]$

Ans: 2

$$\begin{aligned} k(x, x') &= (xx' + 1)^3 \\ &= (xx')^3 + 3(xx')^2 + 3xx' + 1 \\ &= \begin{bmatrix} x^3 & \sqrt{3}x^2 & \sqrt{3}x & 1 \end{bmatrix} \begin{bmatrix} (x')^3 \\ \sqrt{3}(x')^2 \\ \sqrt{3}x' \\ 1 \end{bmatrix} \end{aligned}$$

Quiz

Why might we prefer an SVM over a neural network?

A. With an SVM we can map inputs to an infinite dimensional space. With neural networks, we cannot.

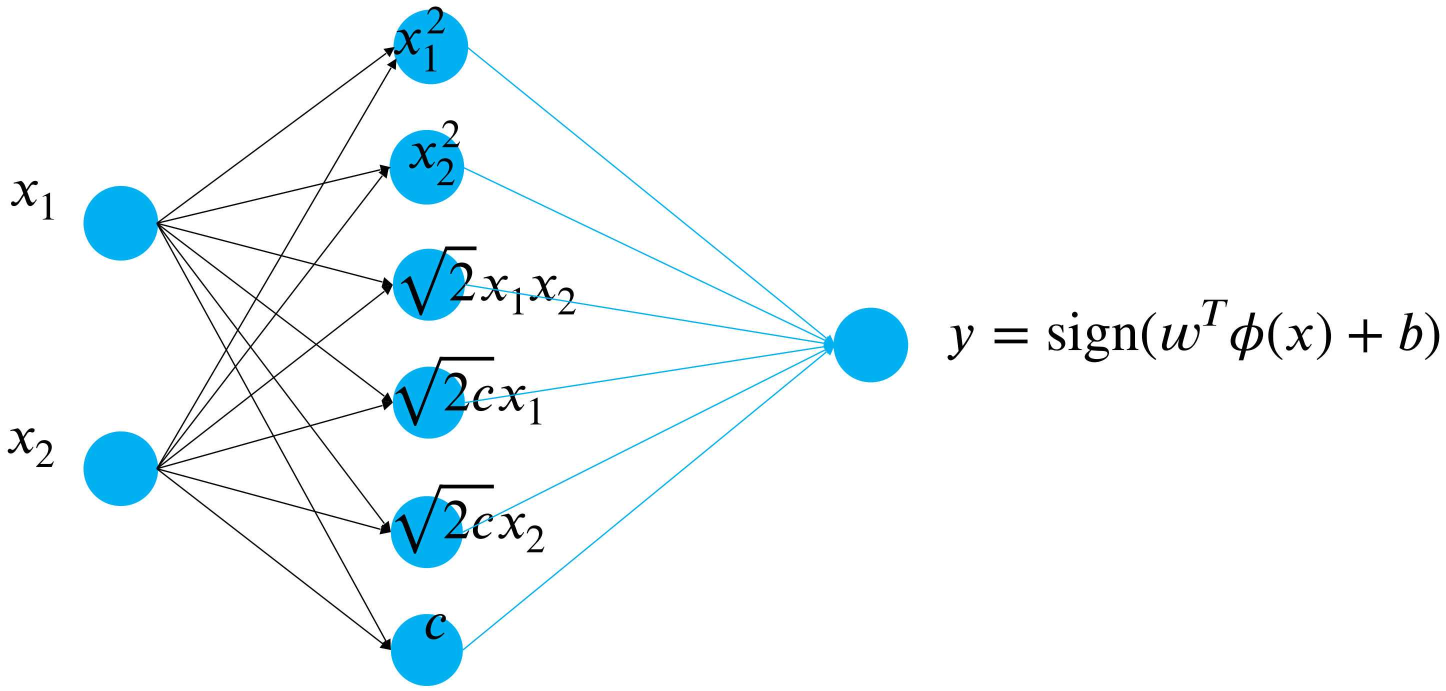
B. SVMs are easier to train: An SVM would not get stuck in a local optima, whereas a neural network might.

C. Tuning hyper-parameters in an SVM may be easier than in neural networks.

Ans: all of the above

Kernel Methods VS Neural Networks

- Can think of our kernel SVM approach as fixing a layer of a neural network.
  - Using kernel feature representations instead of usual activation functions (sigmoid, RELU etc)



Kernel Methods VS Neural Networks

- Kernel methods popular in 90's and 2000's.
- Kernels are still powerful (and probably better than NNs) in small/ moderate data regimes.
- Challenges with Kernel methods (when we have a lot of data):
  - Computational
    - Computing all pairs of kernel values requires $O(n^2)$ memory
    - Compute, typically $O(n^3)$. Solving LP with n constraints or inverting the $n \times n$ kernel matrix is needed.
  - Representation:
    - Using fixed representations is limiting.

Thanks Everyone!

Some of the slides in these lectures have been adapted/borrowed from materials developed by Mark Craven, David Page, Jude Shavlik, Tom Mitchell, Nina Balcan, Elad Hazan, Tom Dietterich, Pedro Domingos, Jerry Zhu, Yingyu Liang, Volodymyr Kuleshov, Fei-Fei Li, Justin Johnson, Serena Yeung, Pieter Abbeel, Peter Chen, Jonathan Ho, Aravind Srinivas