

Announcements

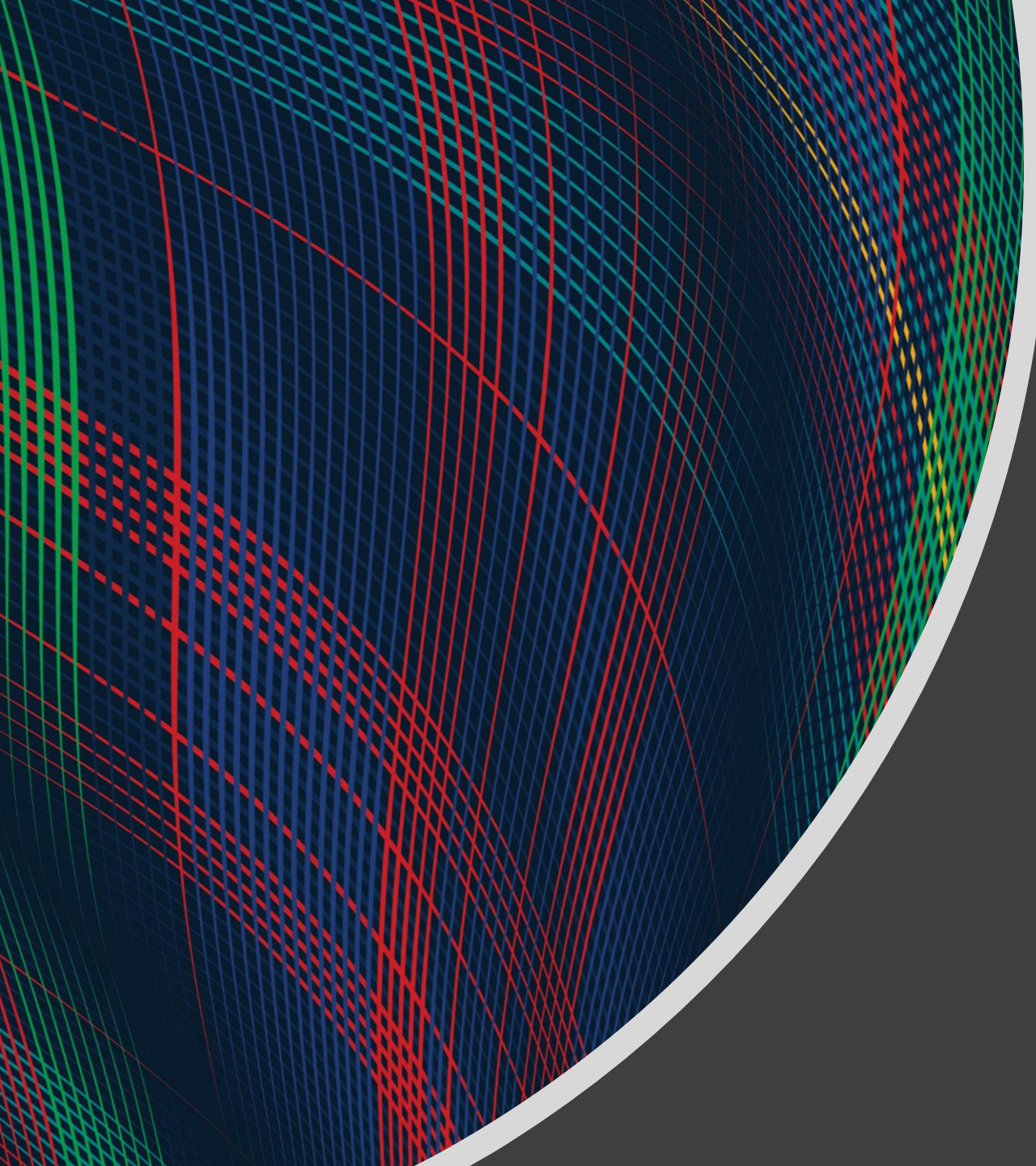


Assignments

- HW8: due Thu, 12/3, 11:59 pm
- HW9
 - Out Friday
 - Due Wed, 12/9, 11:59 pm
 - The two slip days are free (last possible submission Fri, 12/11, 11:59 pm)

Final Exam

- Mon, 12/14
- Stay tuned to Piazza for more details

An abstract graphic on the left side of the slide, featuring a sphere-like shape composed of a dense grid of intersecting red, green, and blue lines. The lines are curved and follow the contour of the sphere, creating a complex, woven pattern. The sphere is set against a dark gray background.

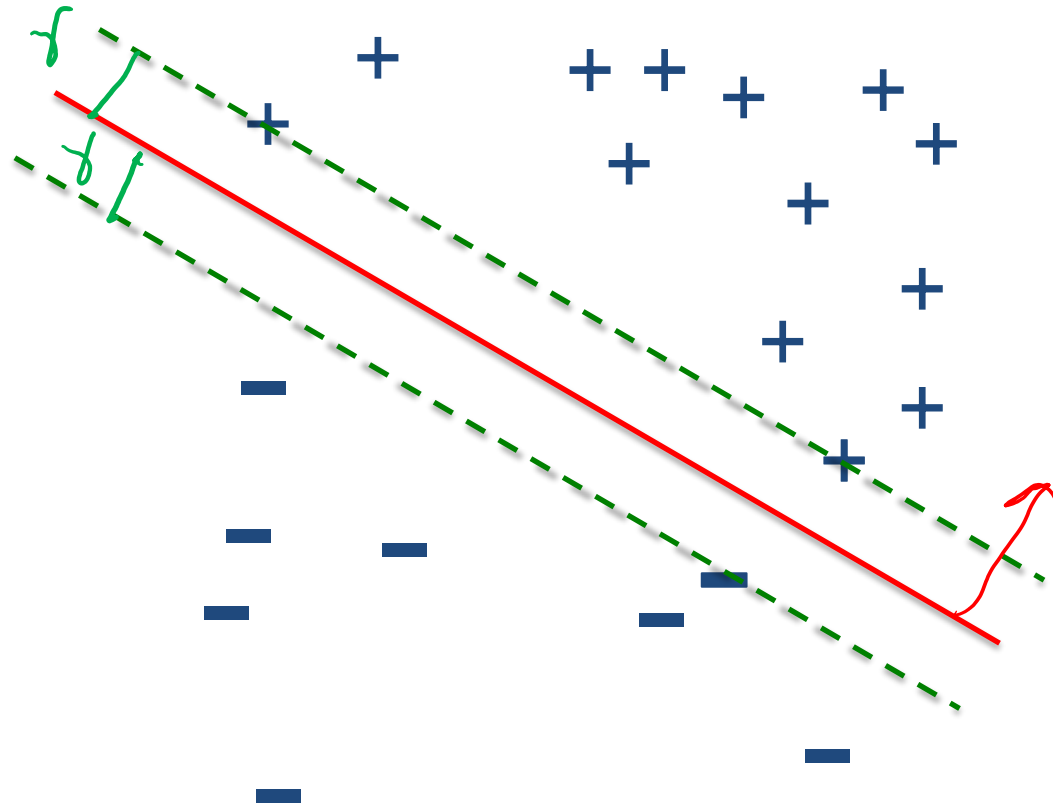
Introduction to Machine Learning

Support Vector Machines

Instructor: Pat Virtue

Support Vector Machines

Find linear separator with maximum margin



$$w^T x + b \geq 0$$

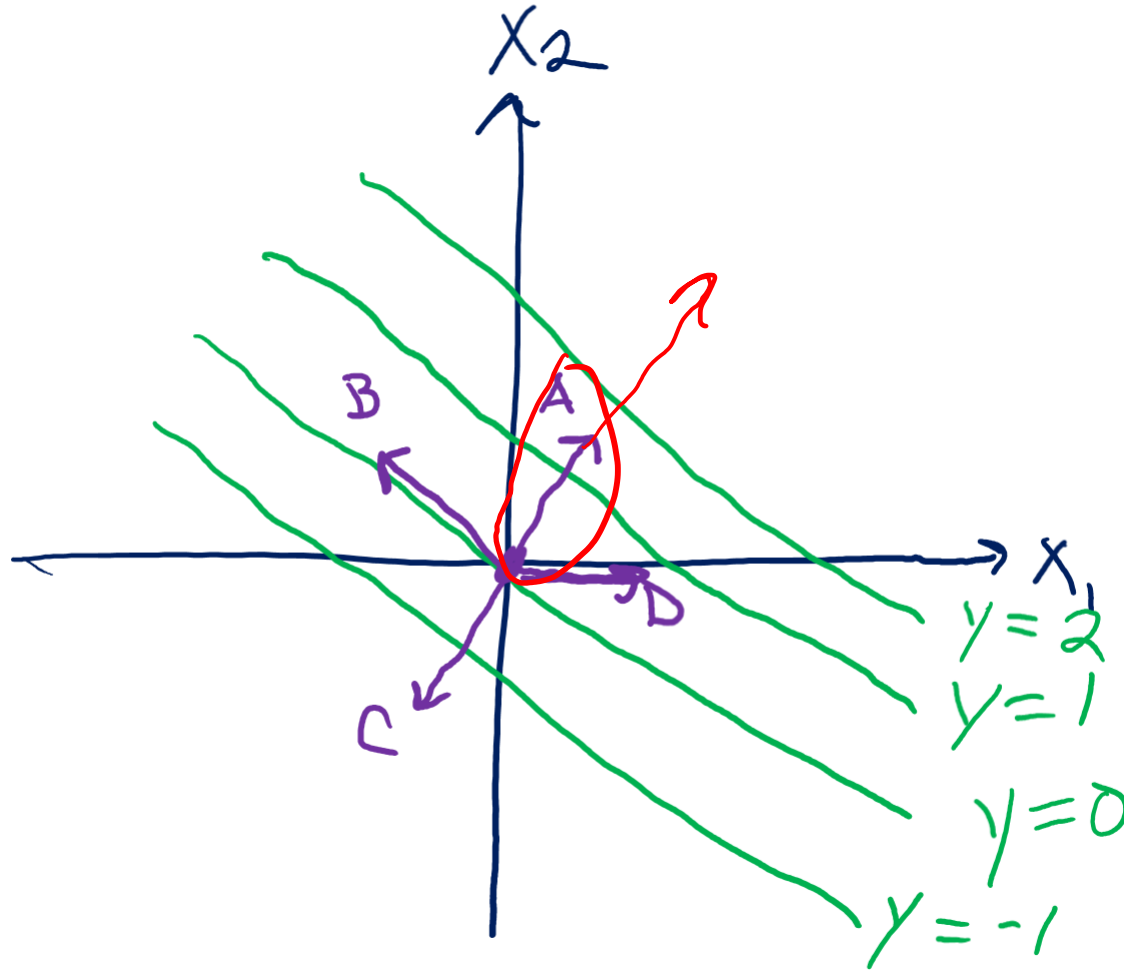
$$\hat{y} = 1$$

$$w^T x + b < 0$$

$$\hat{y} = -1$$

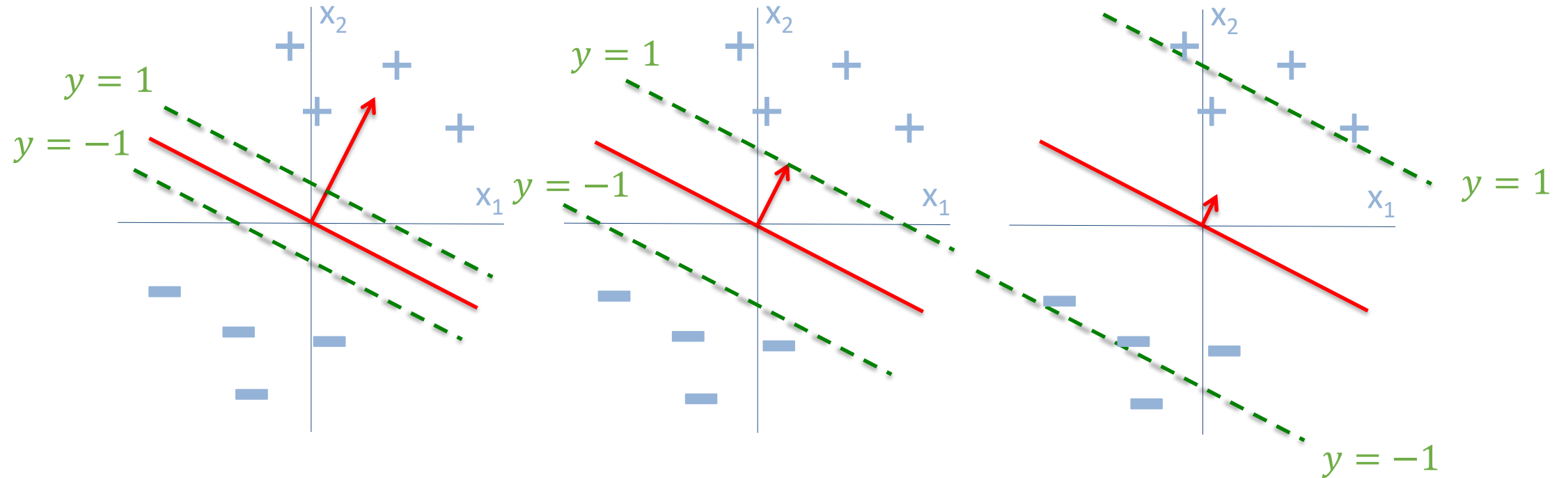
Previous Piazza Poll

As the magnitude of w increases, will the distance between the contour lines of $y = \mathbf{w}^T \mathbf{x} + b$ increase or decrease?



Support Vector Machines

Find linear separator with maximum margin



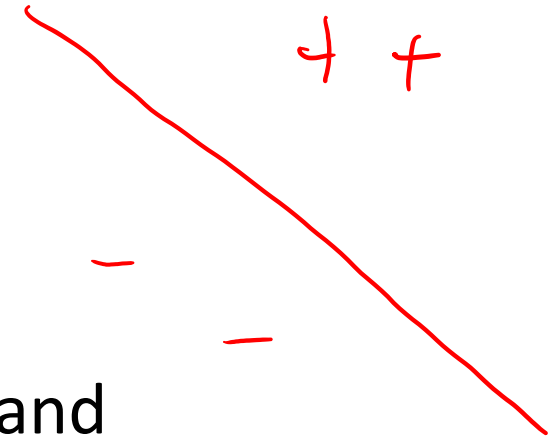
Linear Separability

Data

$$\mathcal{D} = \{\mathbf{x}^{(i)}, y^{(i)}\}_{i=1}^N \quad \mathbf{x} \in \mathbb{R}^M, \quad y \in \{-1, +1\}$$

Linearly separable iff:

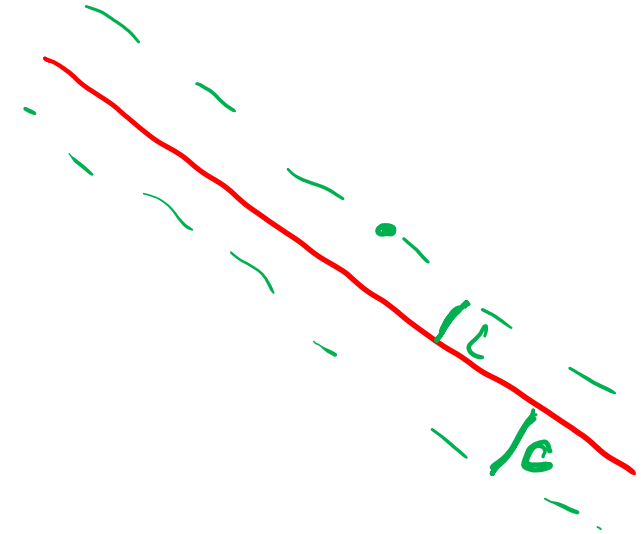
$$\begin{aligned} \exists \mathbf{w}, b \quad s.t. \quad & \mathbf{w}^T \mathbf{x}^{(i)} + b > 0 \quad \text{if } y^{(i)} = +1 \quad \text{and} \\ & \mathbf{w}^T \mathbf{x}^{(i)} + b < 0 \quad \text{if } y^{(i)} = -1 \end{aligned}$$



Linear Separability

Data

$$\mathcal{D} = \{\mathbf{x}^{(i)}, y^{(i)}\}_{i=1}^N \quad \mathbf{x} \in \mathbb{R}^M, \quad y \in \{-1, +1\}$$



Linearly separable iff:

$$\begin{aligned} \exists \mathbf{w}, b \quad s.t. \quad & \mathbf{w}^T \mathbf{x}^{(i)} + b > 0 \quad \text{if } y^{(i)} = +1 \quad \text{and} \\ & \mathbf{w}^T \mathbf{x}^{(i)} + b < 0 \quad \text{if } y^{(i)} = -1 \end{aligned}$$

$$\Leftrightarrow \exists \mathbf{w}, b \quad s.t. \quad \underline{y^{(i)}(\mathbf{w}^T \mathbf{x}^{(i)} + b)} > 0$$

$$\Leftrightarrow \exists \mathbf{w}, b, \mathbf{c} \quad s.t. \quad y^{(i)}(\mathbf{w}^T \mathbf{x}^{(i)} + b) \geq \mathbf{c} \quad \text{and} \quad \mathbf{c} > 0$$

Piazza Poll 1

Yes

max width
s.t. _____

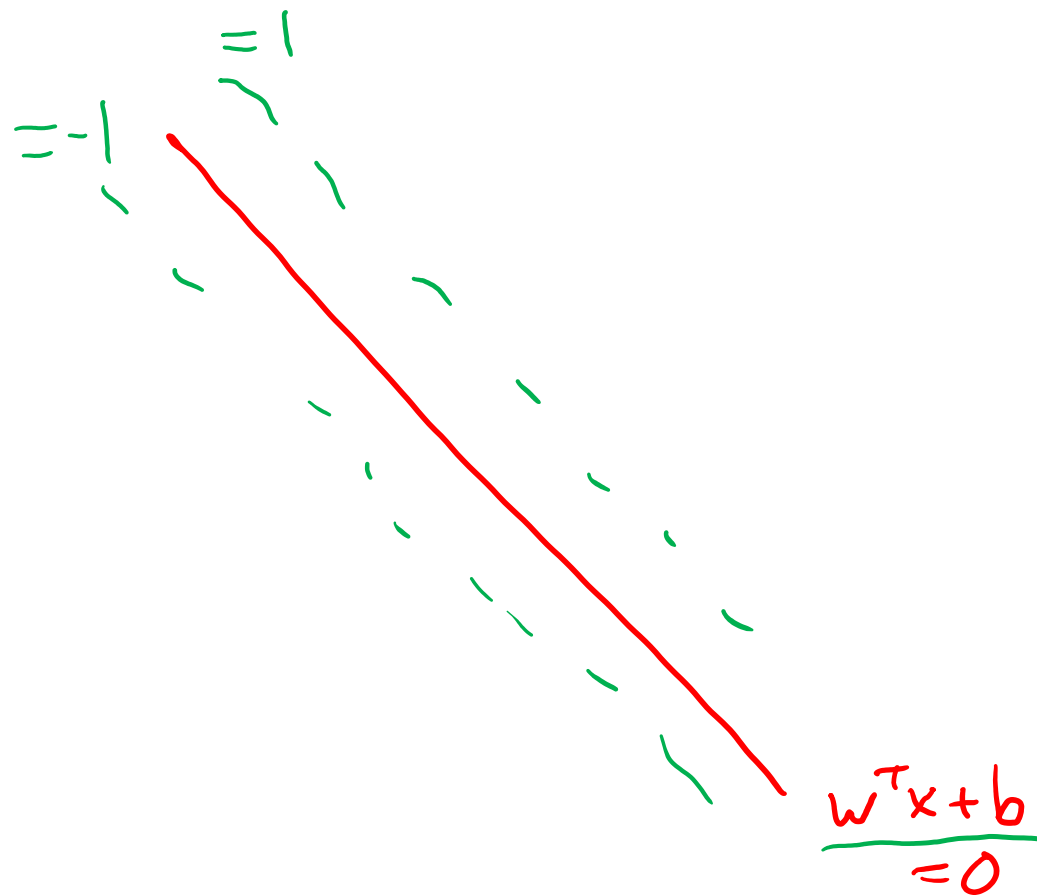
Are these two statements equivalent?

$$\rightarrow \exists \mathbf{w}, b, \underline{c} \text{ s.t. } y^{(i)}(\mathbf{w}^T \mathbf{x}^{(i)} + b) \geq \underline{c} \text{ and } \underline{c} > 0$$

$$\rightarrow \exists \mathbf{w}, b \text{ s.t. } y^{(i)}(\tilde{\mathbf{w}}^T \mathbf{x}^{(i)} + \tilde{b}) \geq \underline{1}$$

$$\tilde{\mathbf{w}} = \frac{\mathbf{w}}{c}$$

$$\tilde{b} = \frac{b}{c}$$



Linear Separability

Data

$$\mathcal{D} = \{\mathbf{x}^{(i)}, y^{(i)}\}_{i=1}^N \quad \mathbf{x} \in \mathbb{R}^M, \quad y \in \{-1, +1\}$$

Linearly separable iff:

$$\begin{aligned} \exists \mathbf{w}, b \quad s.t. \quad & \mathbf{w}^T \mathbf{x}^{(i)} + b > 0 \quad \text{if } y^{(i)} = +1 \quad \text{and} \\ & \mathbf{w}^T \mathbf{x}^{(i)} + b < 0 \quad \text{if } y^{(i)} = -1 \end{aligned}$$

$$\Leftrightarrow \exists \mathbf{w}, b \quad s.t. \quad y^{(i)}(\mathbf{w}^T \mathbf{x}^{(i)} + b) > 0$$

$$\Leftrightarrow \exists \mathbf{w}, b, c \quad s.t. \quad y^{(i)}(\mathbf{w}^T \mathbf{x}^{(i)} + b) \geq c \quad \text{and} \quad c > 0$$

$$\Leftrightarrow \exists \mathbf{w}, b \quad s.t. \quad y^{(i)}(\mathbf{w}^T \mathbf{x}^{(i)} + b) \geq 1 \leftarrow$$

Support Vector Machines

Find linear separator with maximum margin

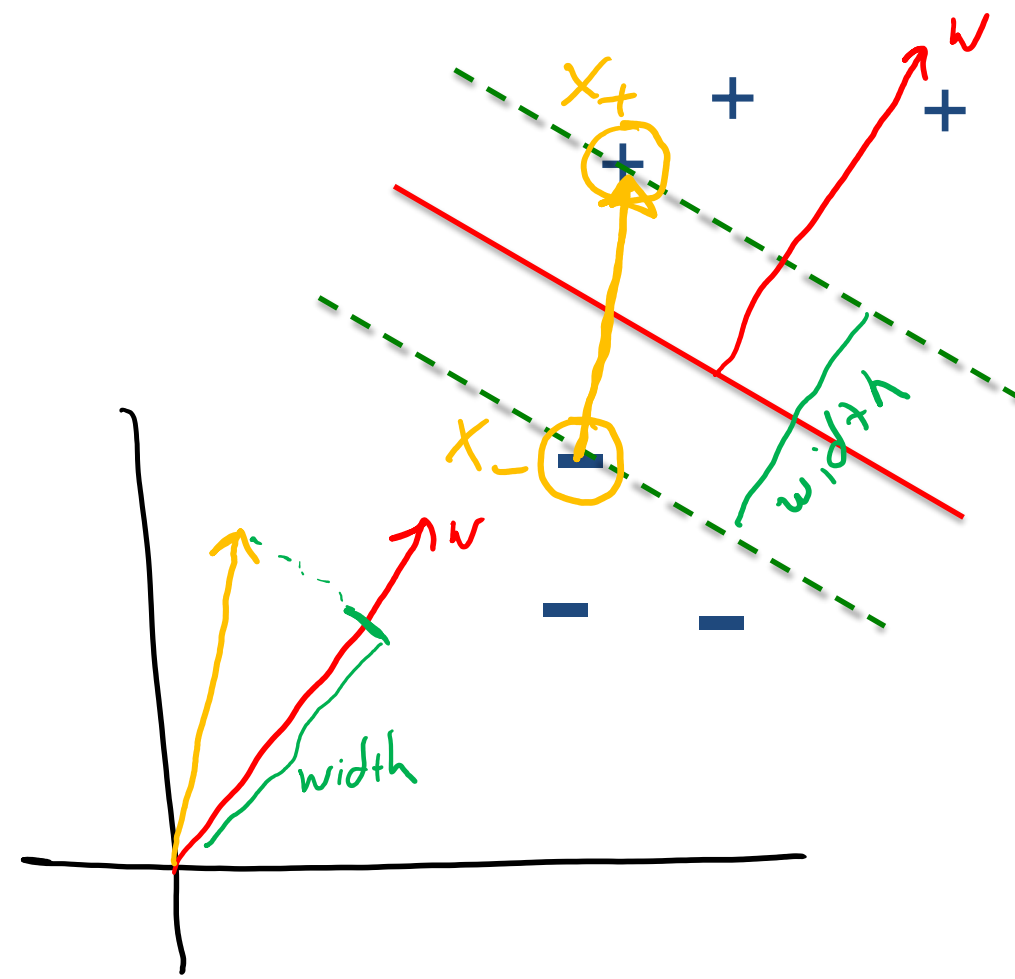
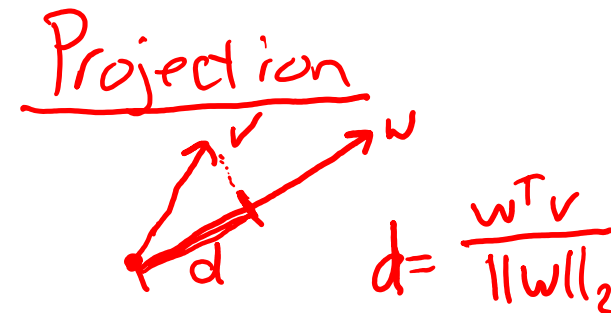
Let $\mathbf{x}_+$ and $\mathbf{x}_-$ be hypothetical points on the +/- margin from the decision boundary

$$\exists \mathbf{w}, b \quad s.t. \quad y^{(i)}(\mathbf{w}^T \mathbf{x}^{(i)} + b) \geq 1$$

$$\Leftrightarrow \exists \mathbf{w}, b \quad s.t. \quad \mathbf{w}^T \mathbf{x}_+ + b \geq +1 \quad \text{and} \\ \mathbf{w}^T \mathbf{x}_- + b \leq -1$$

Consider the vector from $\mathbf{x}_-$ to $\mathbf{x}_+$ and its projection onto the vector $\mathbf{w}$:

$$\text{width} = \frac{\mathbf{w}^T (\mathbf{x}_+ - \mathbf{x}_-)}{\|\mathbf{w}\|_2}$$



Support Vector Machines

Find linear separator with maximum margin

$$\max_{\mathbf{w}, b} \quad \text{"width"}$$

$$\text{s.t.} \quad y^{(i)}(\mathbf{w}^T \mathbf{x}^{(i)} + b) \geq 1 \quad \forall i$$

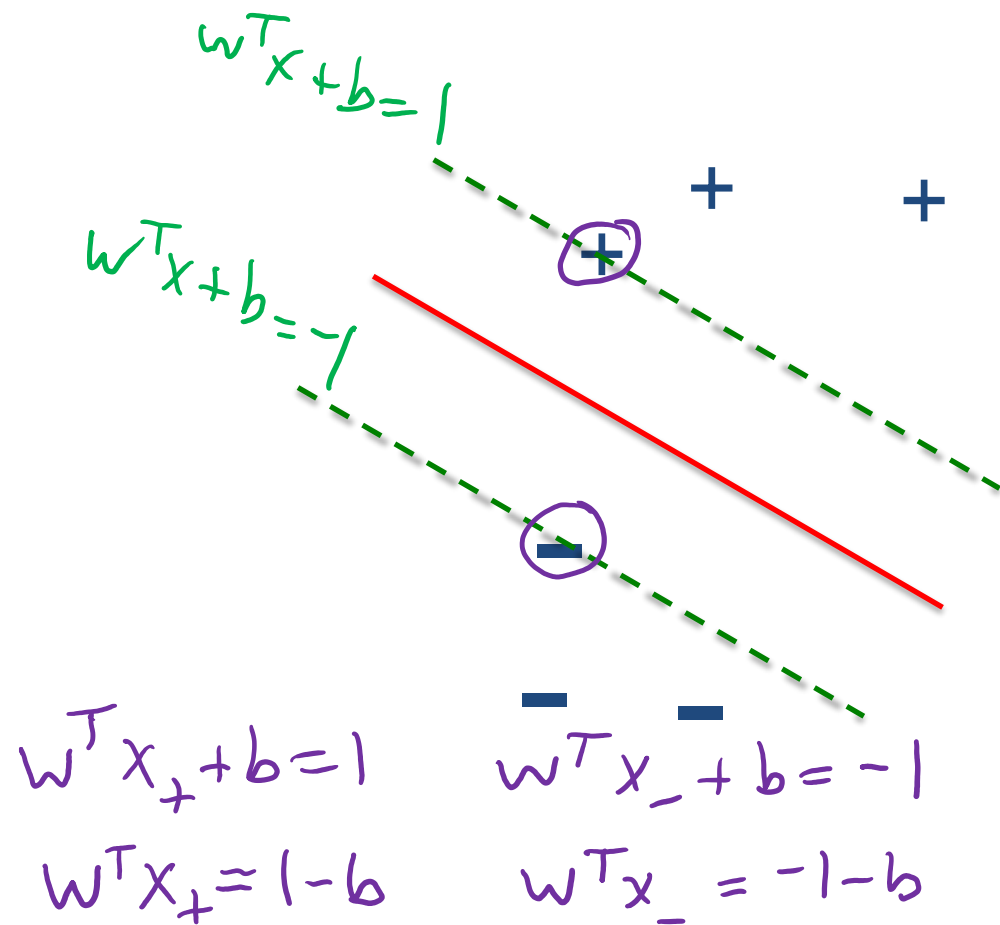
$$\text{width} = \frac{\mathbf{w}^T}{\|\mathbf{w}\|_2} (x_+ - x_-)$$

$$= \frac{1}{\|\mathbf{w}\|_2} (\underbrace{\mathbf{w}^T x_+}_{\downarrow} - \underbrace{\mathbf{w}^T x_-}_{\downarrow})$$

$$= \frac{1}{\|\mathbf{w}\|_2} ((1-b) - (-1-b))$$

$$= \frac{2}{\|\mathbf{w}\|_2}$$


$$\text{width} = \frac{\mathbf{w}^T}{\|\mathbf{w}\|_2} (x_+ - x_-)$$

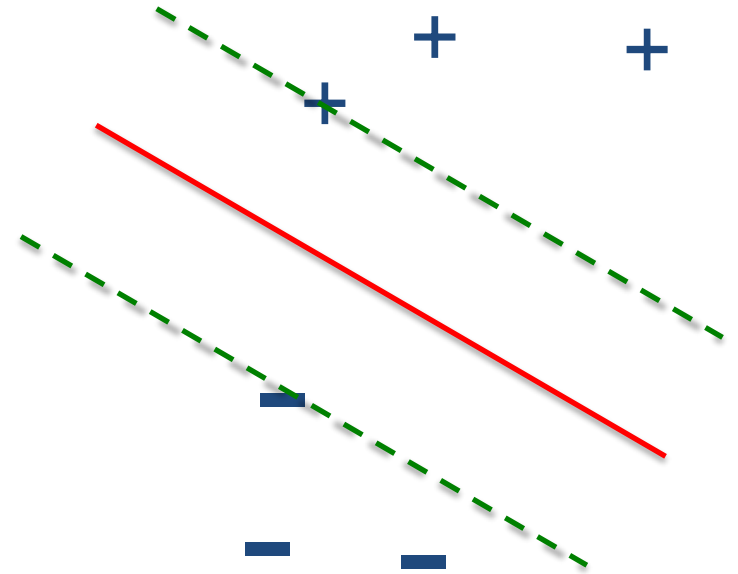


Support Vector Machines

Find linear separator with maximum margin

$$\operatorname{argmax}_{w,b} \quad \text{width}$$

$$\text{width} = \frac{2}{\|w\|_2}$$



Support Vector Machines

Find linear separator with maximum margin

$$\rightarrow \text{width} = \frac{2}{\|\mathbf{w}\|_2}$$

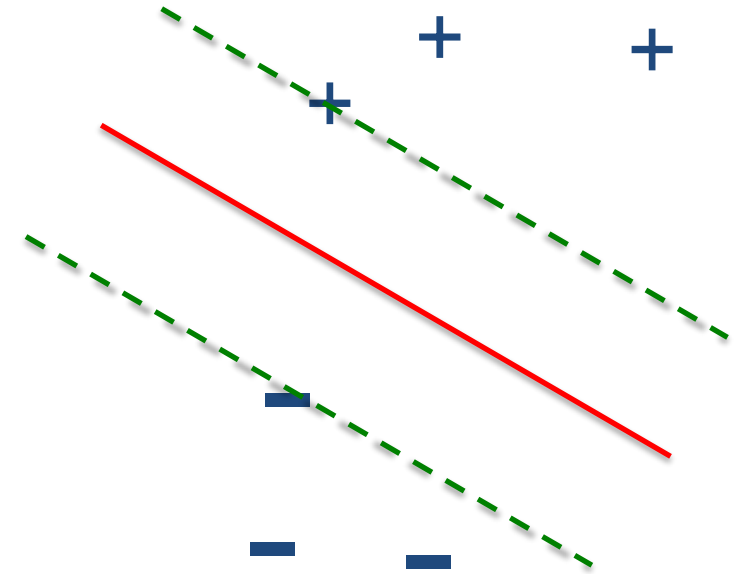
$$\operatorname{argmax}_{\mathbf{w}, b} \quad \text{width}$$

$$\Leftrightarrow \operatorname{argmax}_{\mathbf{w}, b} \quad \frac{2}{\|\mathbf{w}\|_2} \quad \curvearrowright$$

$$\Leftrightarrow \operatorname{argmin}_{\mathbf{w}, b} \quad \frac{1}{2} \|\mathbf{w}\|_2$$

$$\Leftrightarrow \operatorname{argmin}_{\mathbf{w}, b} \quad \frac{1}{2} \|\mathbf{w}\|_2^2$$

$$\Leftrightarrow \operatorname{argmin}_{\mathbf{w}, b} \quad \frac{1}{2} \mathbf{w}^T \mathbf{w}$$



SVM Optimization

Quadratic program!

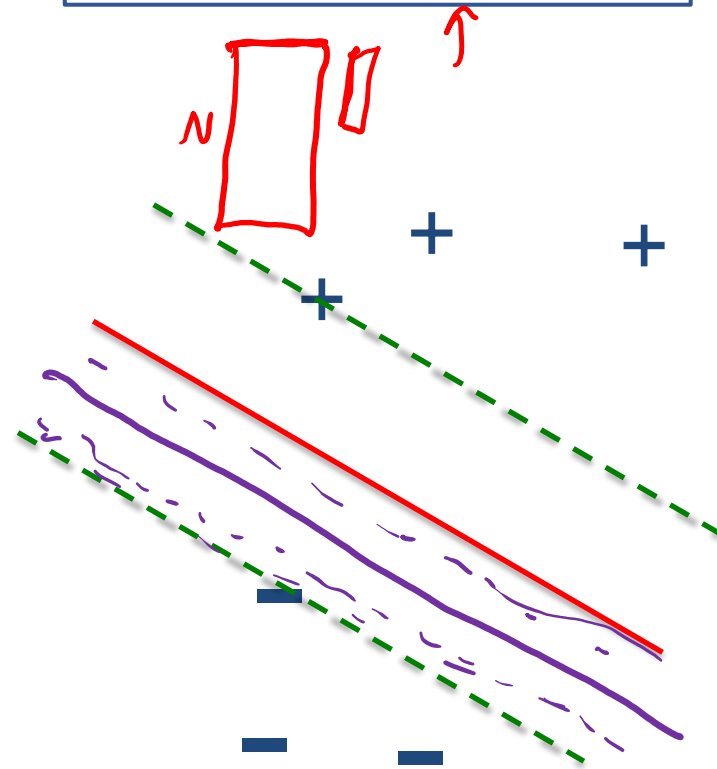
$$\begin{aligned} \min_{\mathbf{w}, b} \quad & \frac{1}{2} \mathbf{w}^T \mathbf{w} \\ \text{s.t.} \quad & y^{(i)} (\mathbf{w}^T \mathbf{x}^{(i)} + b) \geq 1 \quad \forall i \end{aligned}$$

$$y^{(i)} (\theta^T x^{(i)}) \geq 1$$

$$-y^{(i)} (\theta^T x^{(i)}) \leq -1$$

$$-y^{(i)} x^{(i)T} \theta \leq -1$$

$$\begin{aligned} \text{Quadratic Program} \\ \min_x \quad & \mathbf{x}^T \mathbf{Q} \mathbf{x} + \mathbf{c}^T \mathbf{x} \\ \text{s.t.} \quad & \mathbf{A} \mathbf{x} \leq \mathbf{b} \end{aligned}$$

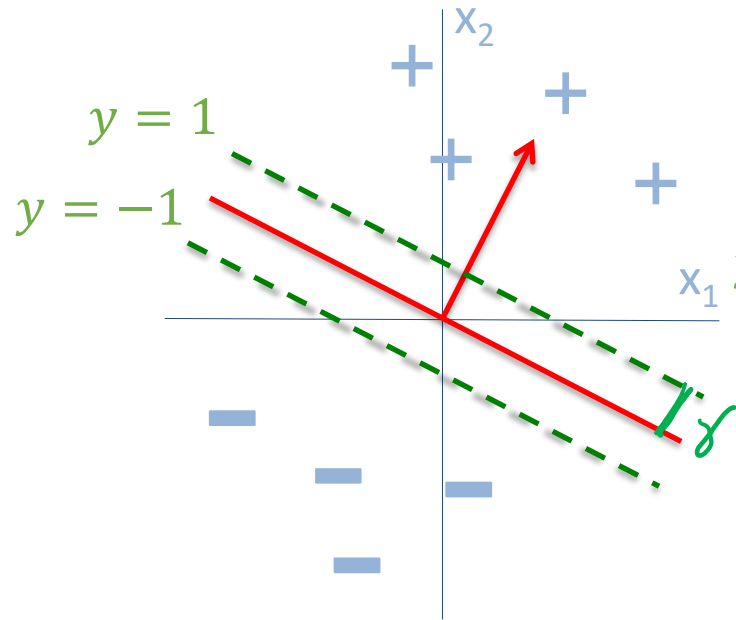


SVM Optimization

How did we go from maximizing margin to minimizing $\|\mathbf{w}\|_2$?

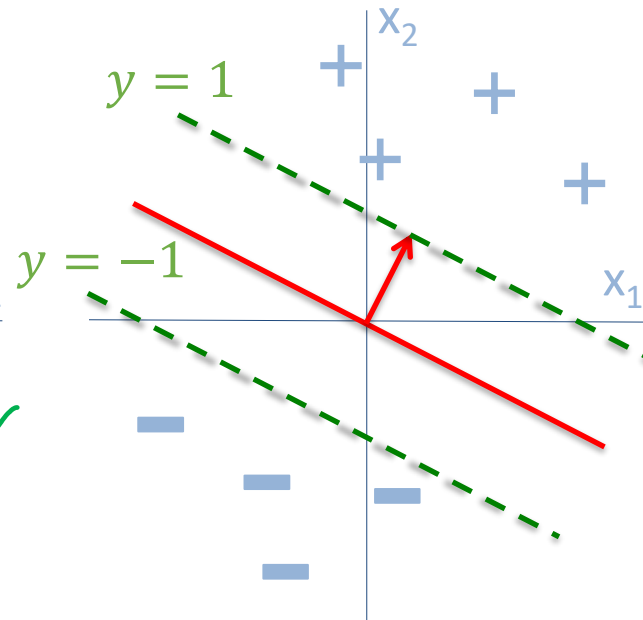
SVM Optimization

How did we go from maximizing margin to minimizing $\|\mathbf{w}\|_2$?



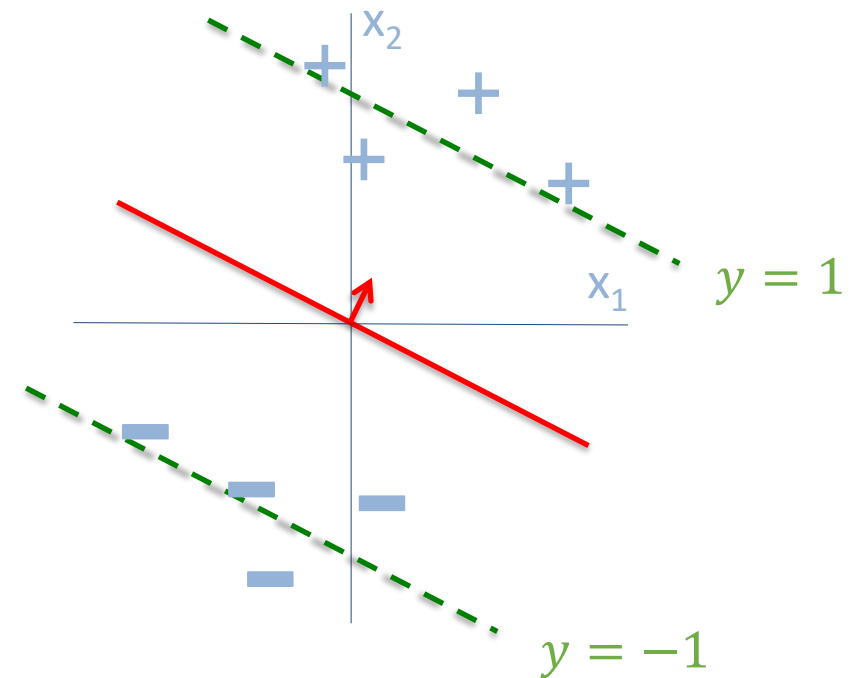
$$\|\mathbf{w}\|_2 = 2$$

$$\gamma = \frac{1}{\|\mathbf{w}\|_2} = \frac{1}{2}$$



$$\|\mathbf{w}\|_2 = 1$$

$$\frac{1}{\|\mathbf{w}\|_2} = 1$$



$$\|\mathbf{w}\|_2 = \frac{1}{2}$$

$$\frac{1}{\|\mathbf{w}\|_2} = 2$$

Linear Separability

Data

$$\mathcal{D} = \{\mathbf{x}^{(i)}, y^{(i)}\}_{i=1}^N \quad \mathbf{x} \in \mathbb{R}^M, \quad y \in \{-1, +1\}$$

Linearly separable iff:

$$\begin{aligned} \exists \mathbf{w}, b \quad s.t. \quad & \mathbf{w}^T \mathbf{x}^{(i)} + b > 0 \quad \text{if } y^{(i)} = +1 \quad \text{and} \\ & \mathbf{w}^T \mathbf{x}^{(i)} + b < 0 \quad \text{if } y^{(i)} = -1 \end{aligned}$$

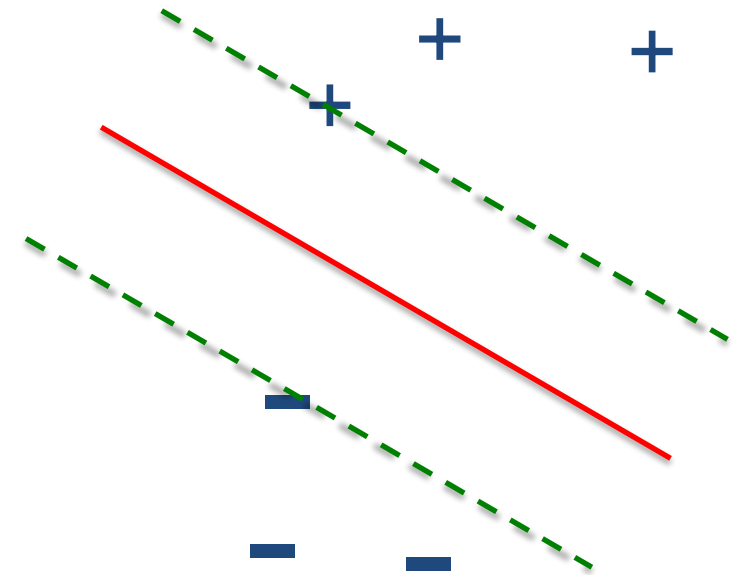
Support Vector Machines

Find linear separator with maximum margin

$\max_{\mathbf{w}, b}$ "width"

s.t. $y^{(i)}(\mathbf{w}^T \mathbf{x}^{(i)} + b) \geq 1 \quad \forall i$

$$width = \frac{\mathbf{w}^T}{\|\mathbf{w}\|_2} (\mathbf{x}_+ - \mathbf{x}_-)$$



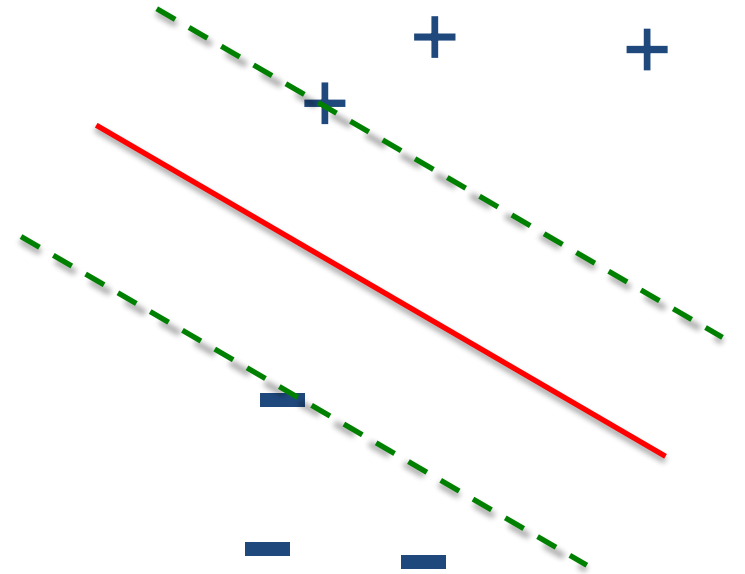
SVM Optimization

Quadratic program!

$$\begin{aligned} \min_{\mathbf{w}, b} \quad & \frac{1}{2} \|\mathbf{w}\|_2^2 \\ \text{s.t.} \quad & y^{(i)} (\mathbf{w}^T \mathbf{x}^{(i)} + b) \geq 1 \quad \forall i \end{aligned}$$

Quadratic Program

$$\begin{aligned} \min_x \quad & \mathbf{x}^T \mathbf{Q} \mathbf{x} + \mathbf{c}^T \mathbf{x} \\ \text{s.t.} \quad & \mathbf{A} \mathbf{x} \leq \mathbf{b} \end{aligned}$$



Constrained Optimization

Linear Program

$$\begin{array}{ll}\min_x & \mathbf{c}^T \mathbf{x} \\ \text{s.t.} & \mathbf{Ax} \leq \mathbf{b}\end{array}$$

Solvers

- Simplex
- Interior point methods

Quadratic Program

$$\begin{array}{ll}\min_x & \mathbf{x}^T \mathbf{Q} \mathbf{x} + \mathbf{c}^T \mathbf{x} \\ \text{s.t.} & \mathbf{Ax} \leq \mathbf{b}\end{array}$$

Solvers

- Conjugate gradient
- Ellipsoid method
- Interior point methods

Constrained Optimization

Linear Program

$$\begin{array}{ll}\min_x & \mathbf{c}^T \mathbf{x} \\ \text{s.t.} & \mathbf{Ax} \leq \mathbf{b}\end{array}$$

Solvers

- Simplex
- Interior point methods

Quadratic Program

$$\begin{array}{ll}\min_x & \mathbf{x}^T \mathbf{Q} \mathbf{x} + \mathbf{c}^T \mathbf{x} \\ \text{s.t.} & \mathbf{Ax} \leq \mathbf{b}\end{array}$$

Special Case

- If $\mathbf{Q}$ is **positive-definite**, the problem is **convex**
- $\mathbf{Q}$ is positive-definite if:
$$\mathbf{v}^T \mathbf{Q} \mathbf{v} > 0 \quad \forall \mathbf{v} \in \mathbb{R}^M \setminus \mathbf{0}$$
- A symmetric $\mathbf{Q}$ is positive-definite if all of its eigenvalues are positive

Support Vector Machines

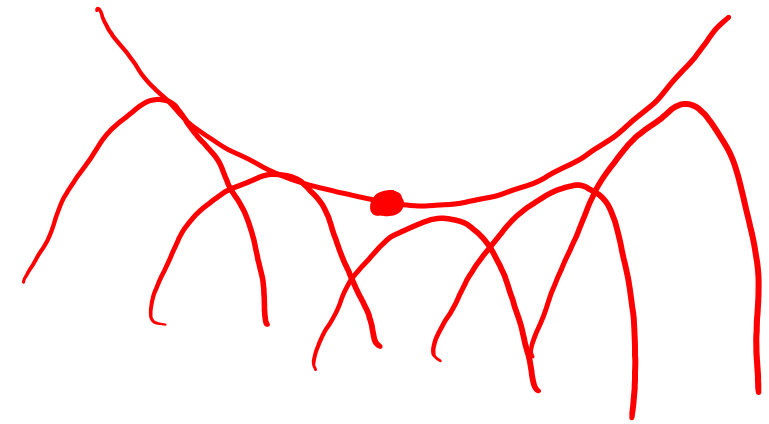
Next steps

- Different optimization formulation
 - Primal → dual ←
 - "Support vectors"
- Support non-linear classification
 - Feature maps
 - Kernel trick
- Support non-separable data
 - Hard-margin SVM → soft-margin SVM

Method of Lagrange Multipliers

Goal

$$\begin{array}{ll} \min_{\mathbf{x}} & f(\mathbf{x}) \\ \text{s.t.} & g(\mathbf{x}) \leq c \end{array}$$



$$g(\mathbf{x}) - c \leq 0$$

$$f(\mathbf{x}) + \lambda (g(\mathbf{x}) - c)$$

↑ dual variable

Step 1: Construct Lagrangian

$$\mathcal{L}(\mathbf{x}, \lambda) = f(\mathbf{x}) + \lambda(g(\mathbf{x}) - c)$$

Step 2: Solve

$$\min_{\mathbf{x}} \max_{\lambda \geq 0} \mathcal{L}(\mathbf{x}, \lambda)$$

Find saddle point:

$$\nabla \mathcal{L}(\mathbf{x}, \lambda) \quad \text{s.t.} \quad \lambda \geq 0$$

Equivalent to solving:

$$\nabla f(\mathbf{x}) = \lambda \nabla g(\mathbf{x}) \quad \text{s.t.} \quad \lambda \geq 0$$

SVM Primal vs Dual

Construct Lagrangian

Primal

$$\begin{aligned} \min_{\mathbf{w}, \underline{\mathbf{b}}} \quad & \frac{1}{2} \|\mathbf{w}\|_2^2 \\ \text{s.t.} \quad & y^{(i)}(\mathbf{w}^T \mathbf{x}^{(i)} + b) \geq 1 \quad \forall i \end{aligned}$$

$$y^{(i)}(\mathbf{w}^T \mathbf{x}^{(i)} + b) - 1 \geq 0$$

$$-(y^{(i)}(\mathbf{w}^T \mathbf{x}^{(i)} + b) - 1) \leq 0$$

$$\mathcal{L}(\vec{\mathbf{w}}, b, \vec{\alpha}) = \frac{1}{2} \|\mathbf{w}\|_2^2 - \sum_{i=1}^N \alpha_i (y^{(i)}(\mathbf{w}^T \mathbf{x}^{(i)} + b) - 1)$$

Lagrange Multipliers

$$\min_{\mathbf{x}} f(\mathbf{x}) \quad \text{s.t.} \quad \underline{g(\mathbf{x}) \leq c}$$

Construct Lagrangian

$$\underline{\mathcal{L}(\mathbf{x}, \lambda)} = f(\mathbf{x}) + \lambda (\underline{g(\mathbf{x}) - c})$$

$$\text{Solve: } \min_{\mathbf{x}} \max_{\lambda \geq 0} \mathcal{L}(\mathbf{x}, \lambda)$$

SVM Dual Optimization

$$\mathcal{L}(w, b, \alpha) = \frac{1}{2} \mathbf{w}^T \mathbf{w} - \sum_i^N \alpha_i [y^{(i)} (\mathbf{w}^T \mathbf{x}^{(i)} + b) - 1]$$

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

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

Slater's condition

$$\frac{\partial \mathcal{L}}{\partial w} = w - \sum_i \alpha_i y^{(i)} \vec{x}^{(i)} = 0$$

$$\frac{\partial \mathcal{L}}{\partial b} = -\sum_i \alpha_i y^{(i)} = 0$$

$$\max_{\alpha \geq 0} \mathcal{L}(\alpha), \quad \mathcal{L}(\alpha) = \sum_i \alpha_i - \frac{1}{2} \sum_{i=1}^N \sum_{j=1}^N \alpha_i \alpha_j y^{(i)} y^{(j)} \mathbf{x}^{(i)T} \mathbf{x}^{(j)}$$

$$\vec{w} = \sum_i \alpha_i y^{(i)} \vec{x}^{(i)}$$

$$\sum_i \alpha_i y^{(i)} = 0$$

SVM Dual Optimization

Dual

$$\begin{aligned} \max_{\alpha} \quad & \sum_i^N \alpha_i - \frac{1}{2} \sum_i^N \sum_j^N \alpha_i \alpha_j y^{(i)} y^{(j)} \mathbf{x}^{(i)T} \mathbf{x}^{(j)} \\ \text{s.t.} \quad & \alpha_i \geq 0 \quad \forall i \end{aligned}$$

$$\mathbf{w} = \sum_i^N \alpha_i y^{(i)} \mathbf{x}^{(i)}$$

$$b = y^{(k)} - \mathbf{w}^T \mathbf{x}^{(k)} \text{ for any } k \text{ where } \alpha_k > 0$$

Prediction

$$\hat{y} = \text{sign}(\mathbf{w}^T \mathbf{x} + b)$$

Dual SVM: Sparsity of dual solution

$$\min_{\mathbf{w}, b} \max_{\alpha \geq 0} \mathcal{L}(\mathbf{w}, b, \alpha)$$

$$\min_{\mathbf{w}, b} \max_{\alpha \geq 0} \frac{1}{2} \mathbf{w}^T \mathbf{w} - \sum_i^N \alpha_i [\underbrace{y^{(i)} (\mathbf{w}^T \mathbf{x}^{(i)} + b) - 1}_{\text{negative}}]$$

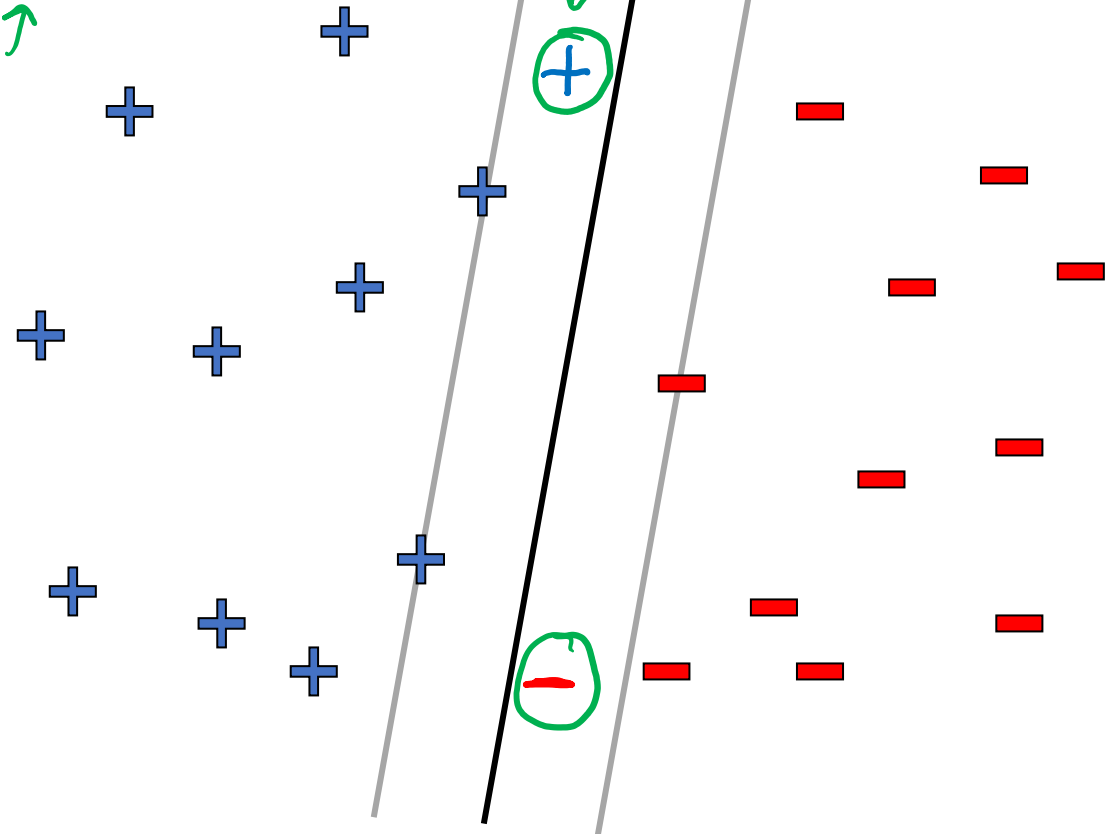
↑

↓
∞

negative
↑

$\mathbf{w}^T \mathbf{x} + b = 1$

Violate
constraint

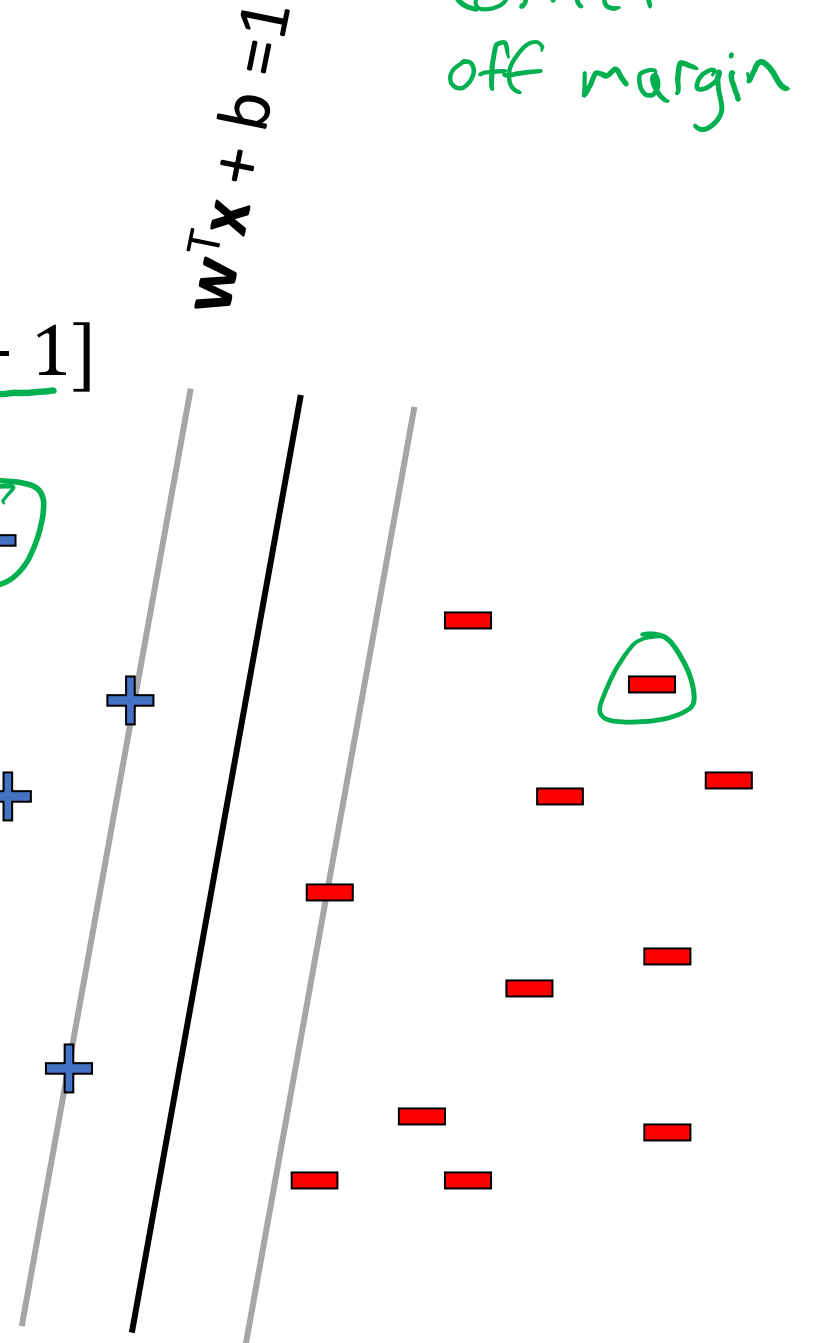


Dual SVM: Sparsity of dual solution

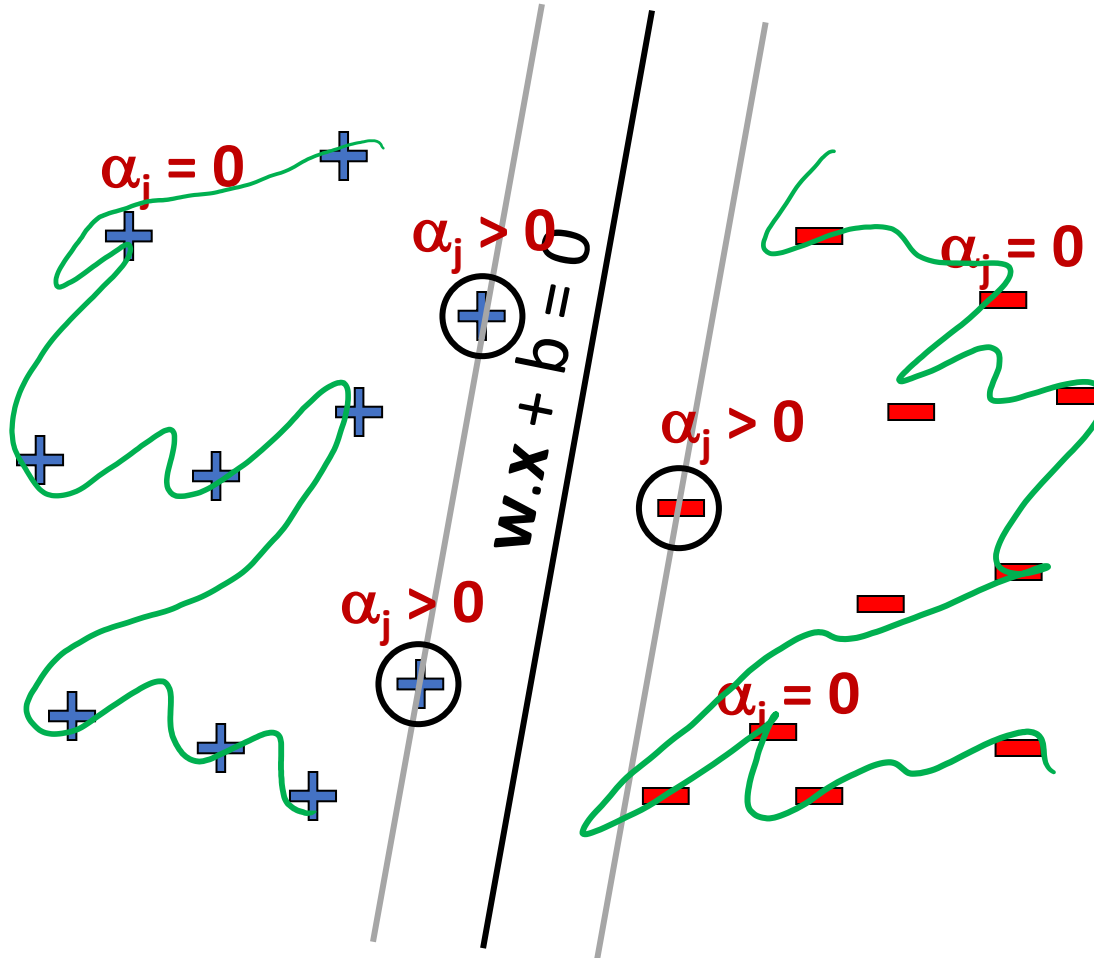
$$\min_{\mathbf{w}, b} \max_{\alpha \geq 0} \mathcal{L}(\mathbf{w}, b, \alpha)$$

$$\min_{\mathbf{w}, b} \max_{\alpha \geq 0} \left[\frac{1}{2} \mathbf{w}^T \mathbf{w} - \sum_i^N \alpha_i \underbrace{[y^{(i)} (\mathbf{w}^T \mathbf{x}^{(i)} + b) - 1]}_{\text{positive}} \right]$$

Annotations: A green arrow points from the α_i term to a green circle containing a 0. Another green arrow points from the underlined term to the word "positive".



Dual SVM: Sparsity of dual solution



$$w = \sum_j \alpha_j y_j x_j$$

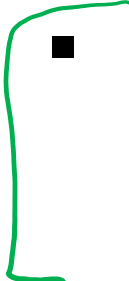
Only few α_j s can be non-zero : where constraint is active and tight

$$(w \cdot x_j + b) y_j = 1$$

Support vectors – training points j whose α_j s are non-zero

Support Vector Machines

Next steps

- Different optimization formulation
 - Primal $\rightarrow$ dual
 - “Support vectors” 
- 
 - Support non-linear classification
 - Feature maps
 - Kernel trick
 - Support non-separable data
 - Hard-margin SVM $\rightarrow$ soft-margin SVM

Kernels: Motivation

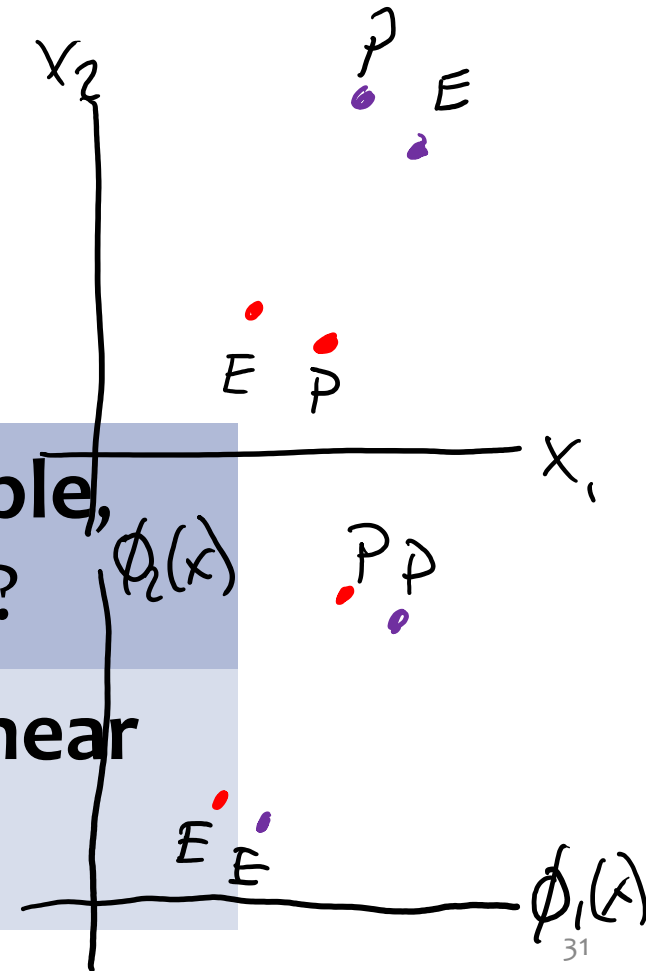
Most real-world problems exhibit data that is not linearly separable.

Example: pixel representation for Facial Recognition:



Q: When your data is **not linearly separable**, how can you still use a linear classifier?

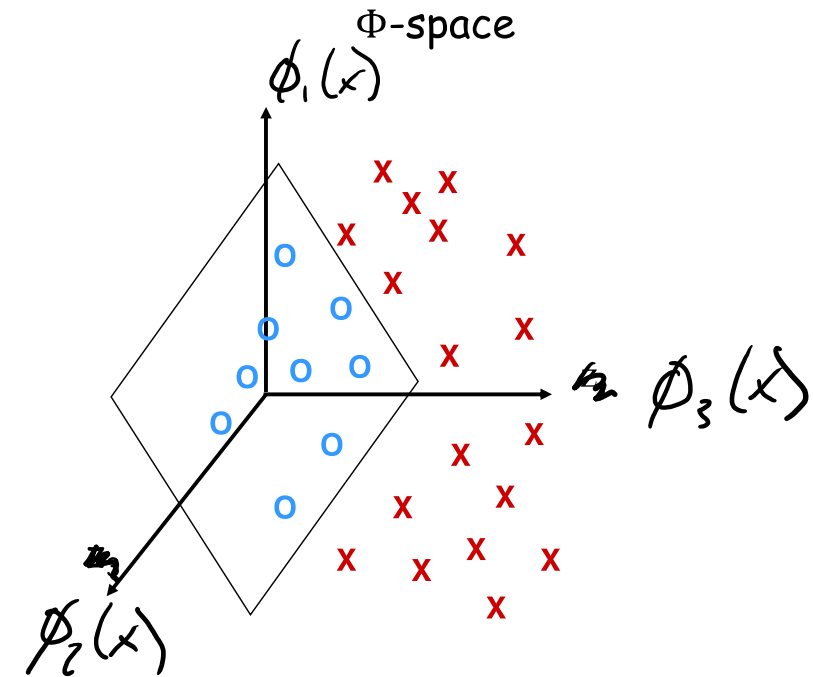
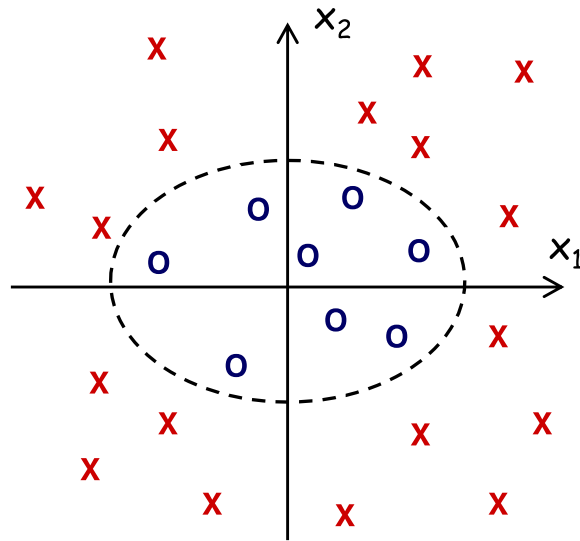
A: Preprocess the data to produce **nonlinear features**



Example: Polynomial Kernel

<https://www.youtube.com/watch?v=3liCbRZPrZA>

Original space



Kernels: Motivation

$$x \xrightarrow{\phi} 1, x, x^2, x^3, x^4$$

- Motivation #1: Inefficient Features
 - Non-linearly separable data requires **high dimensional** representation
 - Might be **prohibitively expensive** to compute or store ←
- Motivation #2: Memory-based Methods
 - k-Nearest Neighbors (KNN) for facial recognition allows a **distance metric** between images -- no need to worry about linearity restriction at all

RBF

Kernel Methods

feature map $\left\{ \begin{array}{l} x^{(i)\top} x^{(j)} \\ \phi(x^{(i)})^\top \phi(x^{(j)}) \\ k(x^{(i)}, x^{(j)}) \end{array} \right.$

- **Key idea:**

1. **Rewrite** the algorithm so that we only work with **dot products** $x^\top z$ of feature vectors
2. **Replace** the **dot products** $x^\top z$ with a **kernel function** $k(x, z)$

- The kernel $k(x, z)$ can be **any** legal definition of a dot product:

$$k(x, z) = \varphi(x)^\top \varphi(z) \text{ for any function } \varphi: X \rightarrow \mathbf{R}^D$$

So we only compute the φ dot product **implicitly**

- This “**kernel trick**” can be applied to many algorithms:
 - classification: perceptron, SVM, ...
 - regression: ridge regression, ...
 - clustering: k-means, ...

SVM: Kernel Trick

Hard-margin SVM (Primal)

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

$$\text{s.t. } y^{(i)}(\mathbf{w}^T \mathbf{x}^{(i)} + b) \geq 1, \quad \forall i$$

- Suppose we do some feature engineering
- Our feature function is ϕ
- We apply ϕ to each input vector $\mathbf{x}$

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

$$\text{s.t. } y^{(i)}(\mathbf{w}^T \phi(\mathbf{x}^{(i)}) + b) \geq 1, \quad \forall i$$

Hard-margin SVM (Lagrangian Dual)

$$\max_{\alpha} \sum_{i=1}^N \alpha_i - \frac{1}{2} \sum_{i=1}^N \sum_{j=1}^N \alpha_i \alpha_j y^{(i)} y^{(j)} \mathbf{x}^{(i)} \cdot \mathbf{x}^{(j)}$$

$$\text{s.t. } \alpha_i \geq 0, \quad \forall i = 1, \dots, N$$

$$\sum_{i=1}^N \alpha_i y^{(i)} = 0$$

feature mapping

$$\max_{\alpha} \sum_{i=1}^N \alpha_i - \frac{1}{2} \sum_{i=1}^N \sum_{j=1}^N \alpha_i \alpha_j y^{(i)} y^{(j)} \phi(\mathbf{x}^{(i)}) \cdot \phi(\mathbf{x}^{(j)})$$

$$\text{s.t. } \alpha_i \geq 0, \quad \forall i = 1, \dots, N$$

$$\sum_{i=1}^N \alpha_i y^{(i)} = 0$$

SVM: Kernel Trick

Hard-margin SVM (Lagrangian Dual)

$$\max_{\alpha} \sum_{i=1}^N \alpha_i - \frac{1}{2} \sum_{i=1}^N \sum_{j=1}^N \alpha_i \alpha_j y^{(i)} y^{(j)} \phi(\mathbf{x}^{(i)}) \cdot \phi(\mathbf{x}^{(j)})$$

$$\text{s.t. } \alpha_i \geq 0, \quad \forall i = 1, \dots, N$$

$$\sum_{i=1}^N \alpha_i y^{(i)} = 0$$

We could replace the dot product of the two feature vectors in the transformed space with a function $k(\mathbf{x}, \mathbf{z})$ where $k(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}) = \phi(\mathbf{x}^{(i)}) \cdot \phi(\mathbf{x}^{(j)})$

SVM: Kernel Trick

Hard-margin SVM (Lagrangian Dual)

$$\max_{\alpha} \sum_{i=1}^N \alpha_i - \frac{1}{2} \sum_{i=1}^N \sum_{j=1}^N \alpha_i \alpha_j y^{(i)} y^{(j)} k(\mathbf{x}^{(i)}, \mathbf{x}^{(j)})$$

$$\text{s.t. } \alpha_i \geq 0, \quad \forall i = 1, \dots, N$$

$$\sum_{i=1}^N \alpha_i y^{(i)} = 0$$

We could replace the dot product of the two feature vectors in the transformed space with a function $k(\mathbf{x}, \mathbf{z})$ where $k(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}) = \phi(\mathbf{x}^{(i)}) \cdot \phi(\mathbf{x}^{(j)})$

Kernel Methods

- **Key idea:**
 1. **Rewrite** the algorithm so that we only work with **dot products** $x^T z$ of feature vectors
 2. **Replace** the **dot products** $x^T z$ with a **kernel function** $k(x, z)$
- The kernel $k(x, z)$ can be **any** legal definition of a dot product:

$$k(x, z) = \varphi(x)^T \varphi(z) \text{ for any function } \varphi: X \rightarrow \mathbf{R}^D$$

So we only compute the φ dot product **implicitly**

- This “**kernel trick**” can be applied to many algorithms:
 - classification: perceptron, SVM, ...
 - regression: ridge regression, ...
 - clustering: k-means, ...

Kernel Methods

Q: These are just non-linear features, right?

A: Yes, but...

Q: Can't we just compute the feature transformation φ explicitly?

A: That depends...

Q: So, why all the hype about the kernel trick?

A: Because the **explicit features** might either be **prohibitively expensive** to compute or **infinite length** vectors

Example: Polynomial Kernel

For $n=2$, $d=2$, the kernel $K(x, z) = (x \cdot z)^d$ corresponds to

$$\phi: \mathbb{R}^2 \rightarrow \mathbb{R}^3, (x_1, x_2) \rightarrow \Phi(x) = (x_1^2, x_2^2, \sqrt{2}x_1x_2)$$

$$\begin{aligned} \phi(x) \cdot \phi(z) &= (x_1^2, x_2^2, \sqrt{2}x_1x_2) \cdot (z_1^2, z_2^2, \sqrt{2}z_1z_2) \\ &= (x_1z_1 + x_2z_2)^2 = (x \cdot z)^2 = K(x, z) \end{aligned}$$

Kernel Examples

Side Note: The feature space might not be unique!

Explicit representation #1:

$$\phi: \mathbb{R}^2 \rightarrow \mathbb{R}^3, (x_1, x_2) \rightarrow \Phi(x) = \underline{(x_1^2, x_2^2, \sqrt{2}x_1x_2)}$$

①

$$\begin{aligned}\phi(x) \cdot \phi(z) &= (x_1^2, x_2^2, \sqrt{2}x_1x_2) \cdot (z_1^2, z_2^2, \sqrt{2}z_1z_2) \\ &= (x_1z_1 + x_2z_2)^2 = \underline{(x \cdot z)^2} = K(x, z)\end{aligned}$$

Explicit representation #2:

$$\phi: \mathbb{R}^2 \rightarrow \mathbb{R}^4, (x_1, x_2) \rightarrow \Phi(x) = \underline{(x_1^2, x_2^2, x_1x_2, x_2x_1)}$$

②

$$\begin{aligned}\phi(x) \cdot \phi(z) &= (x_1^2, x_2^2, x_1x_2, x_2x_1) \cdot (z_1^2, z_2^2, z_1z_2, z_2z_1) \\ &= \underline{(x \cdot z)^2} = K(x, z)\end{aligned}$$

These two different feature representations correspond to the same kernel function!

Kernel Examples

Name	Kernel Function (implicit dot product)	Feature Space (explicit dot product)
Linear	$K(\mathbf{x}, \mathbf{z}) = \mathbf{x}^T \mathbf{z}$	Same as original input space
Polynomial (v1)	$K(\mathbf{x}, \mathbf{z}) = (\mathbf{x}^T \mathbf{z})^d$	All polynomials of degree d
Polynomial (v2)	$K(\mathbf{x}, \mathbf{z}) = (\mathbf{x}^T \mathbf{z} + 1)^d$	All polynomials up to degree d
Gaussian (RBF)	$K(\mathbf{x}, \mathbf{z}) = \exp\left(-\frac{\ \mathbf{x} - \mathbf{z}\ _2^2}{2\sigma^2}\right)$	Infinite dimensional space
Hyperbolic Tangent (Sigmoid) Kernel	$K(\mathbf{x}, \mathbf{z}) = \tanh(\alpha \mathbf{x}^T \mathbf{z} + c)$	(With SVM, this is equivalent to a 2-layer neural network)

Kernels: Mercer's Theorem

What functions are valid kernels that correspond to feature vectors $\phi(\mathbf{x})$?

Answer: Mercer kernels for $k(\mathbf{x}, \mathbf{z})$ and matrix K , where $K_{i,j} = k(\underline{x^{(i)}}, \underline{x^{(j)}})$

- $k(\mathbf{x}, \mathbf{z})$ is continuous
- K is symmetric
- K is positive semi-definite, i.e. $\mathbf{z}^T K \mathbf{z} \geq 0$ for all $\mathbf{z}$

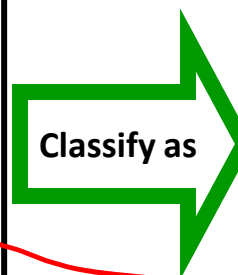
SVMs with Kernels

- Choose a set of features and kernel function
- Solve dual problem to obtain support vectors α_i
- At classification time, compute:

$$\mathbf{w} \cdot \Phi(\mathbf{x}) = \sum_i \alpha_i y_i K(\mathbf{x}, \mathbf{x}_i)$$
$$b = y_k - \sum_i \alpha_i y_i K(\mathbf{x}_k, \mathbf{x}_i)$$

for any k where $C > \alpha_k > 0$

$x^T x^i$

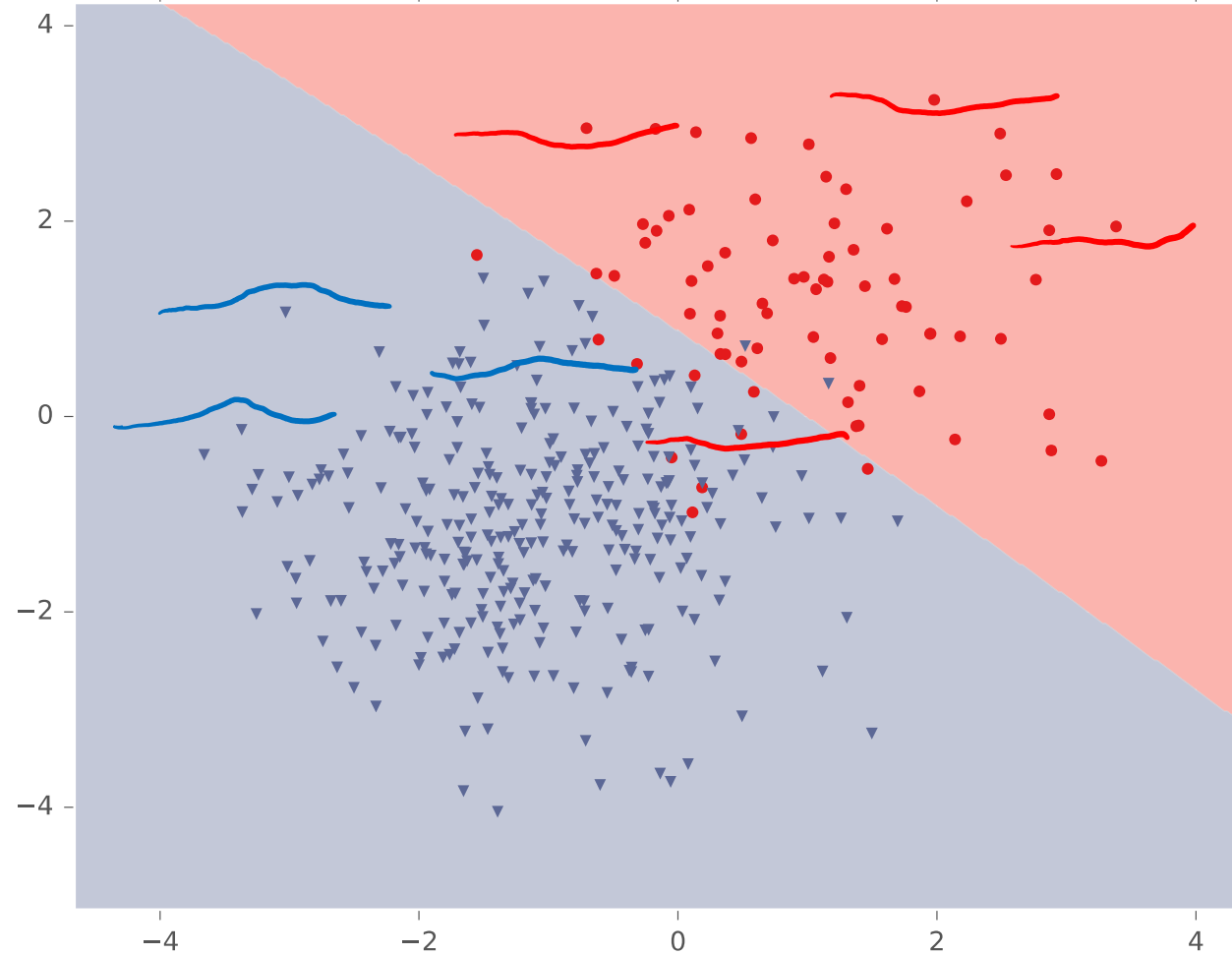


Classify as

$$\text{sign}(\mathbf{w} \cdot \Phi(\mathbf{x}) + b)$$

RBF Kernel Example

Classification with SVM (kernel=rbf, gamma=0.010000)

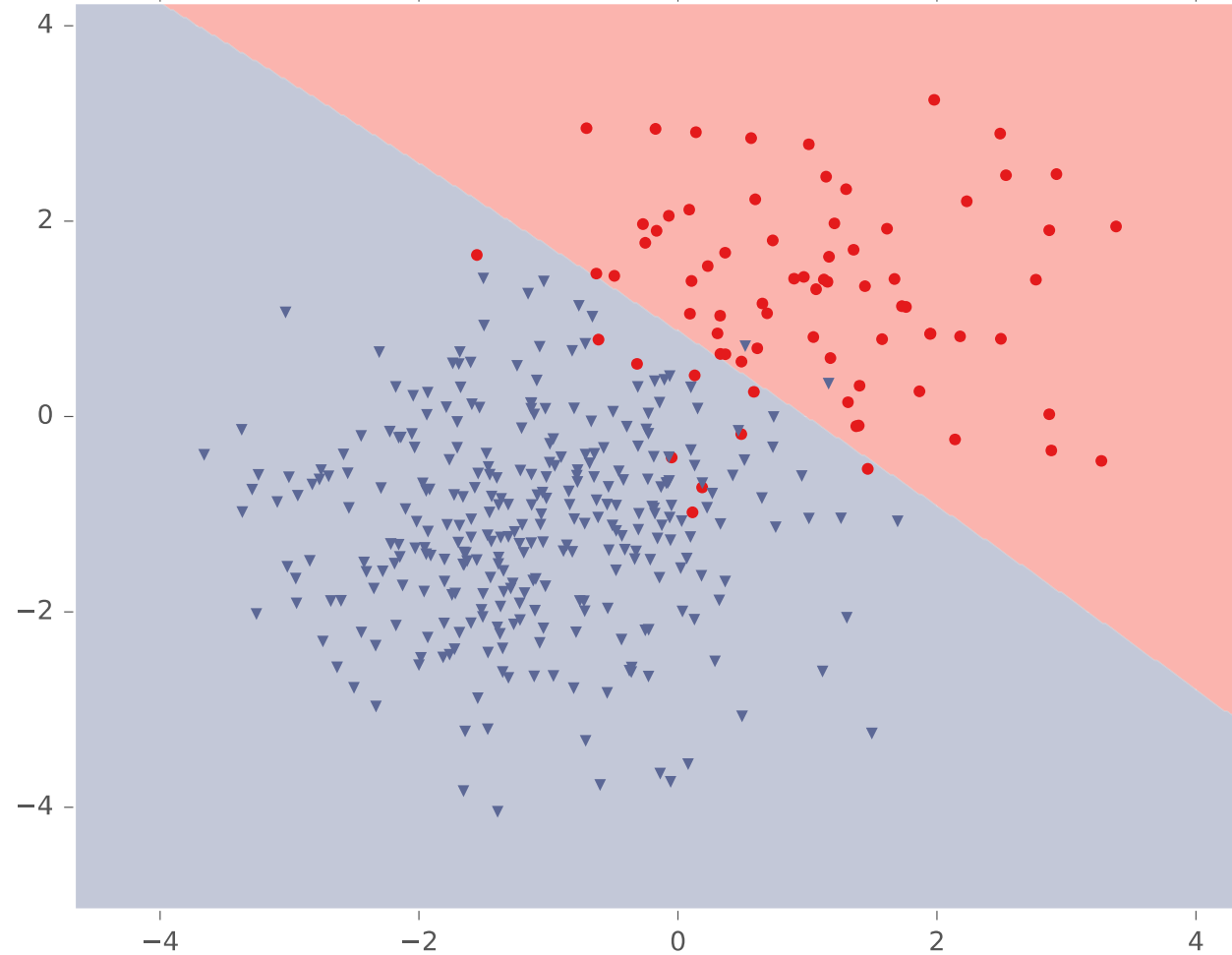


RBF Kernel: $K(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}) = \exp(-\gamma \|\mathbf{x}^{(i)} - \mathbf{x}^{(j)}\|_2^2)$

$$\gamma = \frac{1}{2\sigma^2}$$

RBF Kernel Example

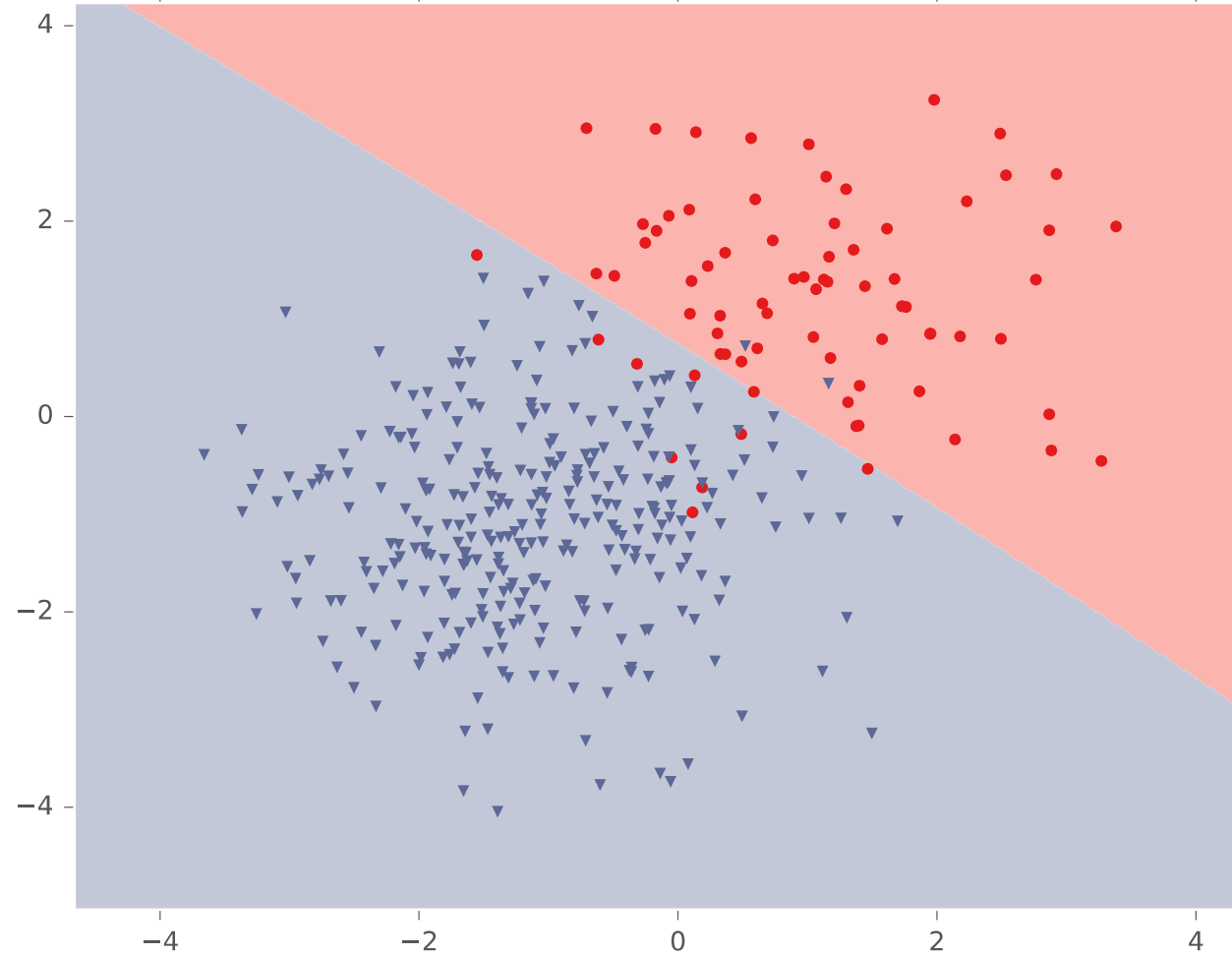
Classification with SVM (kernel=rbf, gamma=0.010000)



RBF Kernel: $K(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}) = \exp(-\gamma \|\mathbf{x}^{(i)} - \mathbf{x}^{(j)}\|_2^2)$

RBF Kernel Example

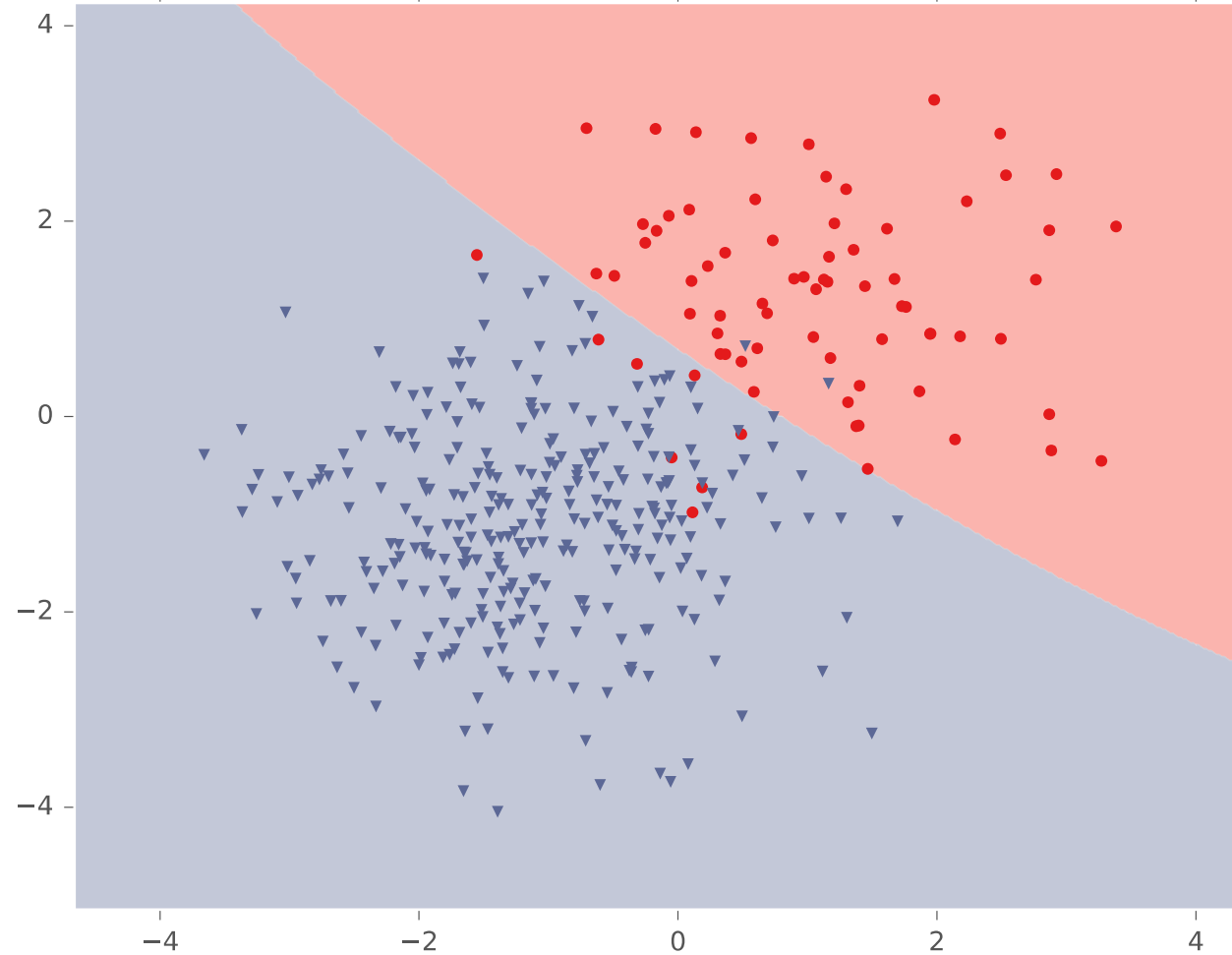
Classification with SVM (kernel=rbf, gamma=0.020000)



RBF Kernel: $K(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}) = \exp(-\gamma \|\mathbf{x}^{(i)} - \mathbf{x}^{(j)}\|_2^2)$

RBF Kernel Example

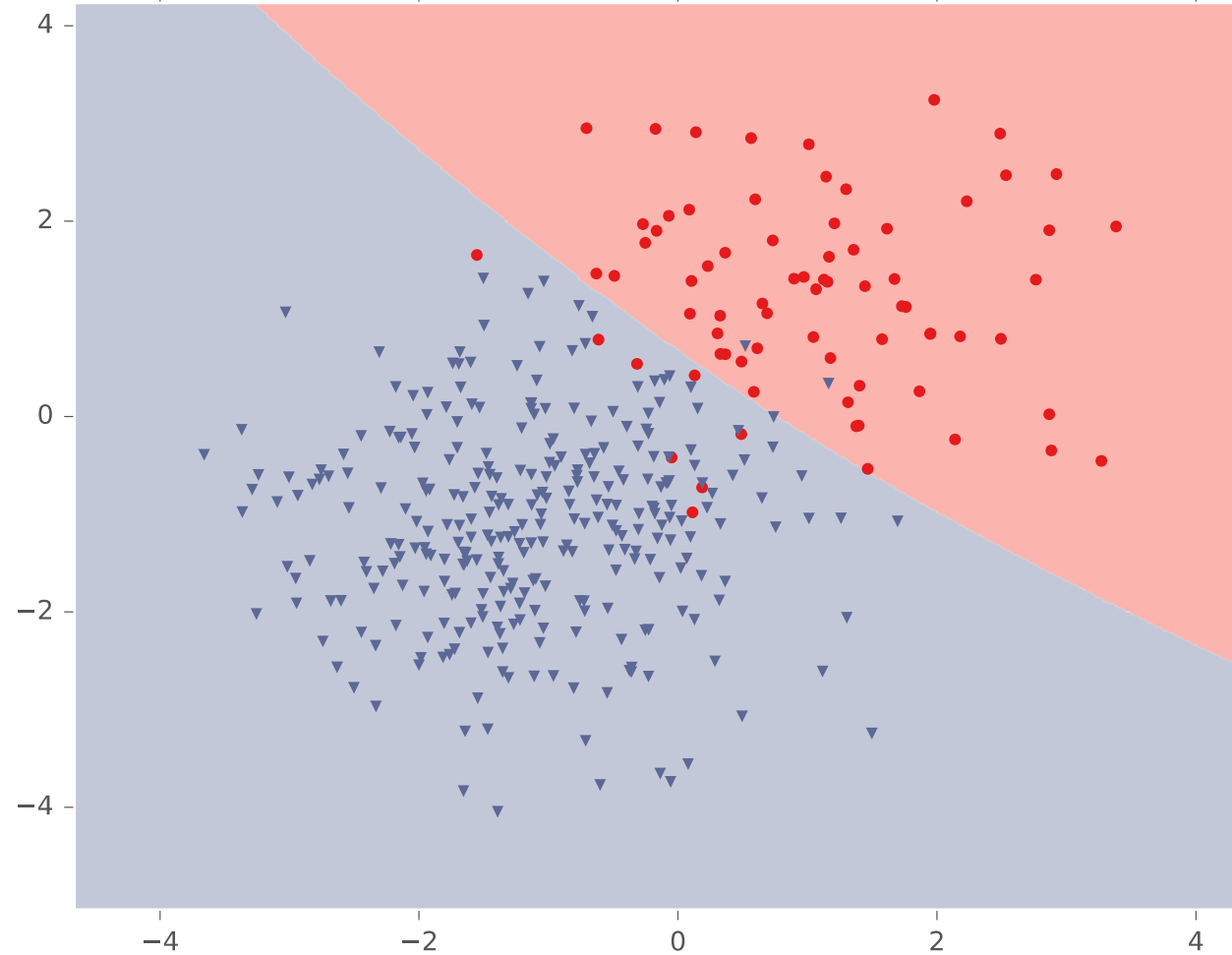
Classification with SVM (kernel=rbf, gamma=0.040000)



RBF Kernel: $K(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}) = \exp(-\gamma \|\mathbf{x}^{(i)} - \mathbf{x}^{(j)}\|_2^2)$

RBF Kernel Example

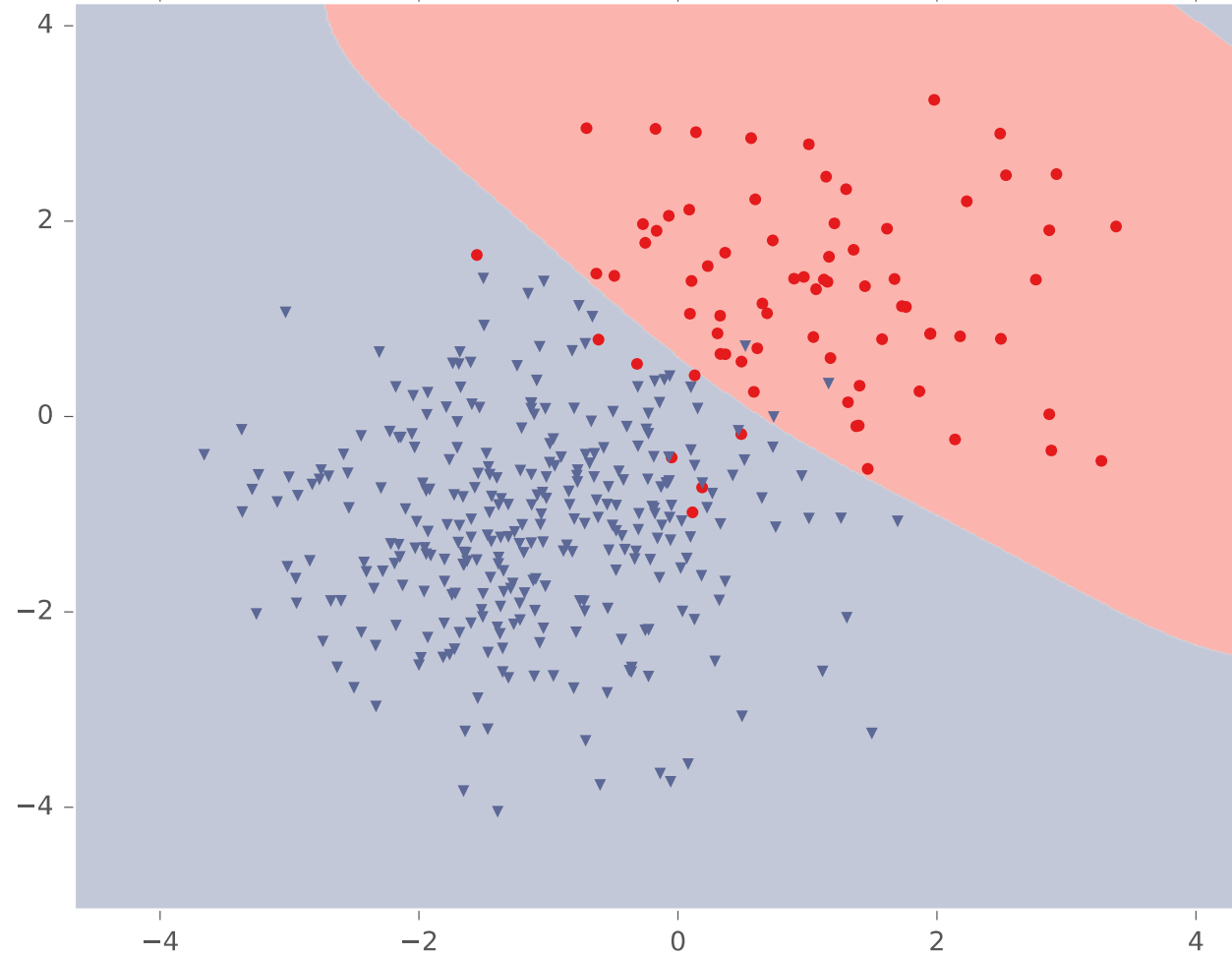
Classification with SVM (kernel=rbf, gamma=0.080000)



RBF Kernel: $K(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}) = \exp(-\gamma \|\mathbf{x}^{(i)} - \mathbf{x}^{(j)}\|_2^2)$

RBF Kernel Example

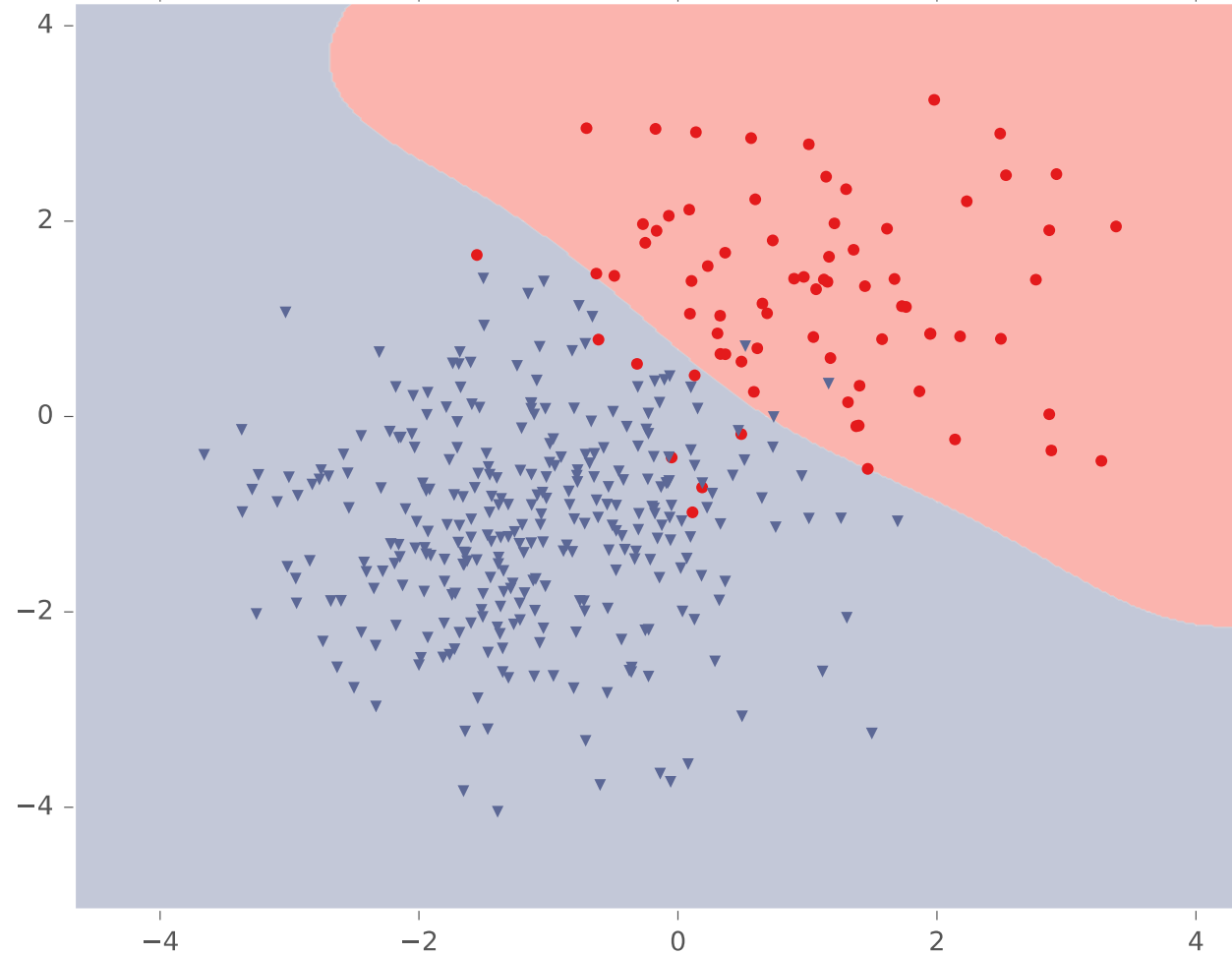
Classification with SVM (kernel=rbf, gamma=0.160000)



RBF Kernel: $K(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}) = \exp(-\gamma \|\mathbf{x}^{(i)} - \mathbf{x}^{(j)}\|_2^2)$

RBF Kernel Example

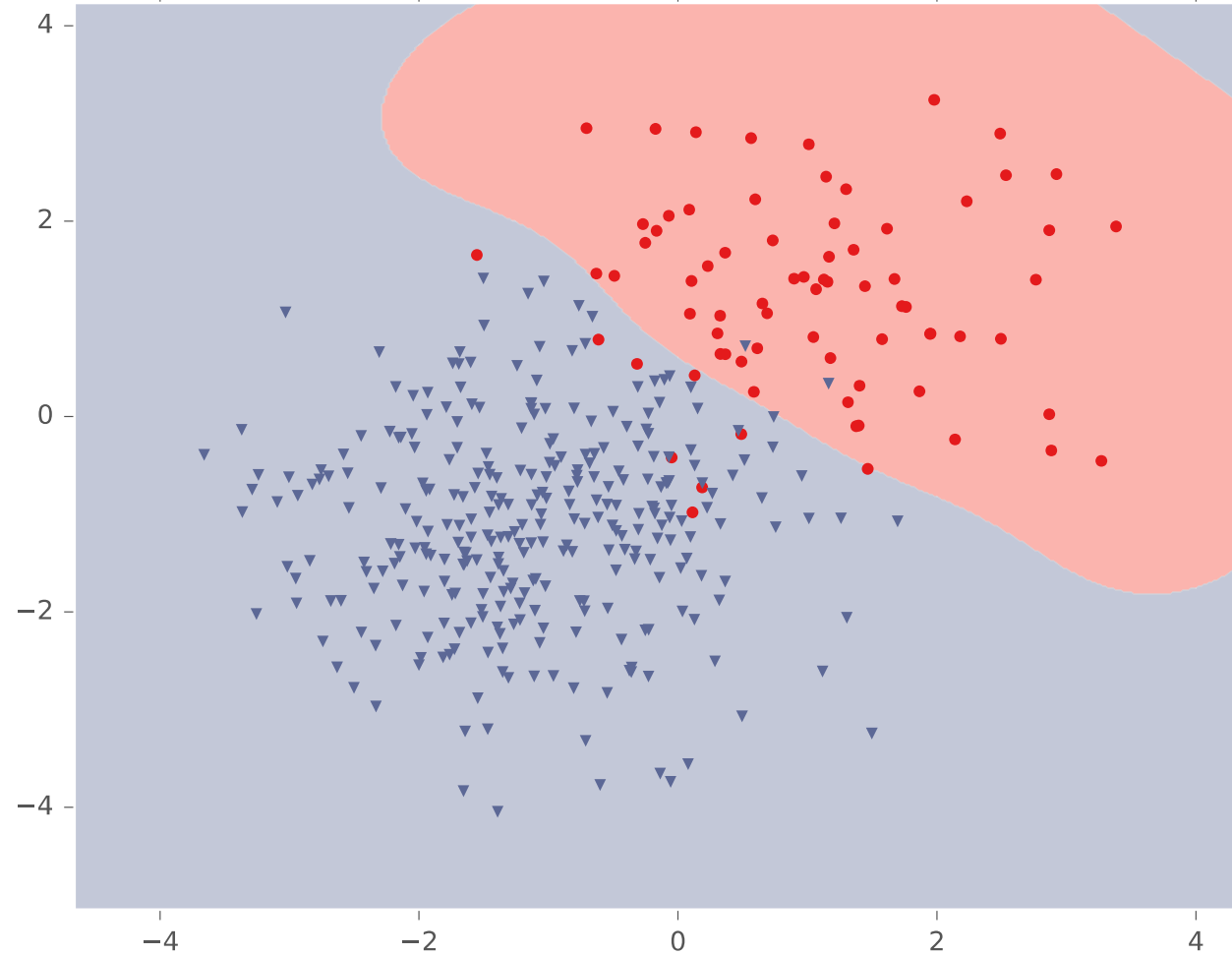
Classification with SVM (kernel=rbf, gamma=0.320000)



RBF Kernel: $K(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}) = \exp(-\gamma \|\mathbf{x}^{(i)} - \mathbf{x}^{(j)}\|_2^2)$

RBF Kernel Example

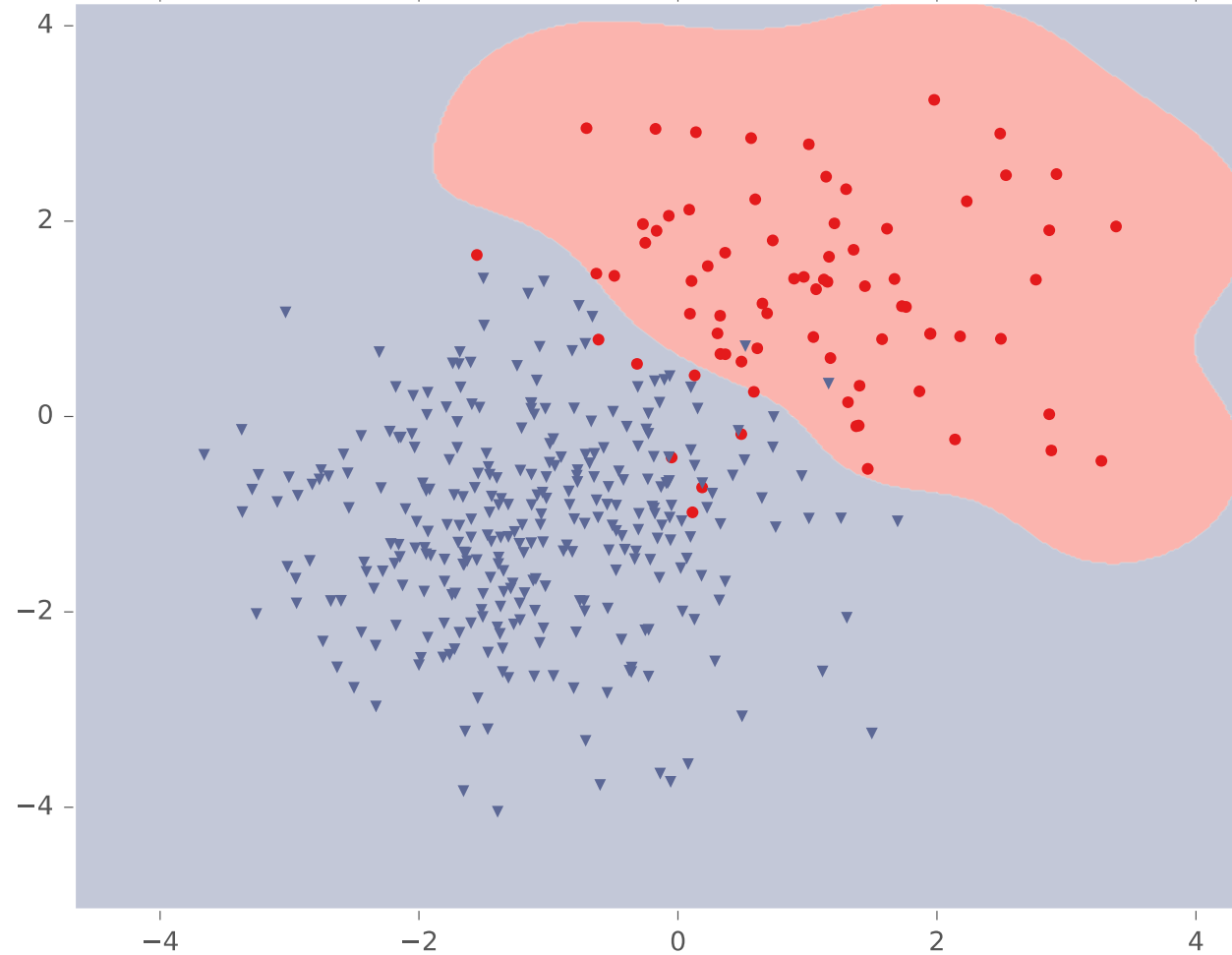
Classification with SVM (kernel=rbf, gamma=0.640000)



RBF Kernel: $K(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}) = \exp(-\gamma \|\mathbf{x}^{(i)} - \mathbf{x}^{(j)}\|_2^2)$

RBF Kernel Example

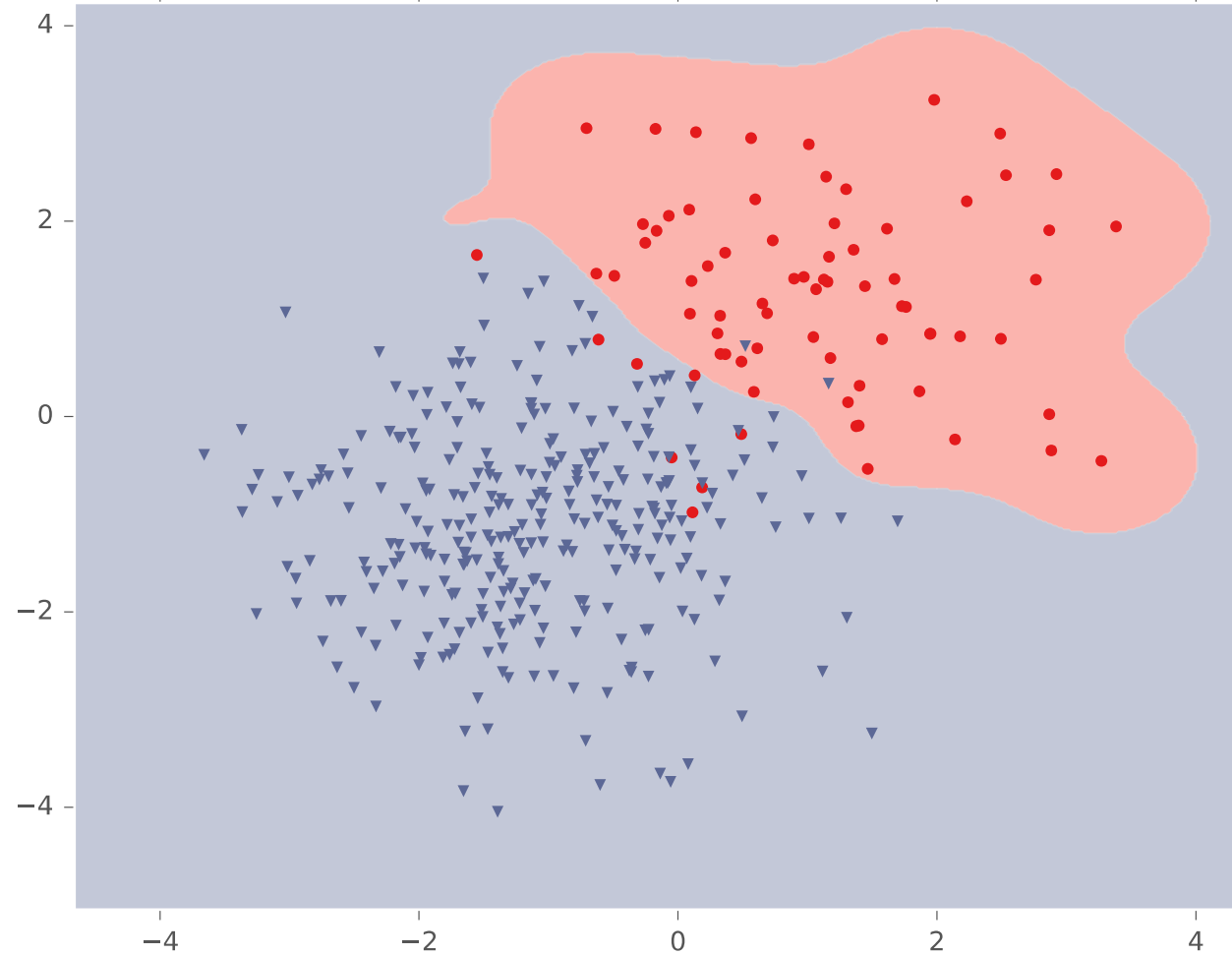
Classification with SVM (kernel=rbf, gamma=1.280000)



RBF Kernel: $K(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}) = \exp(-\gamma \|\mathbf{x}^{(i)} - \mathbf{x}^{(j)}\|_2^2)$

RBF Kernel Example

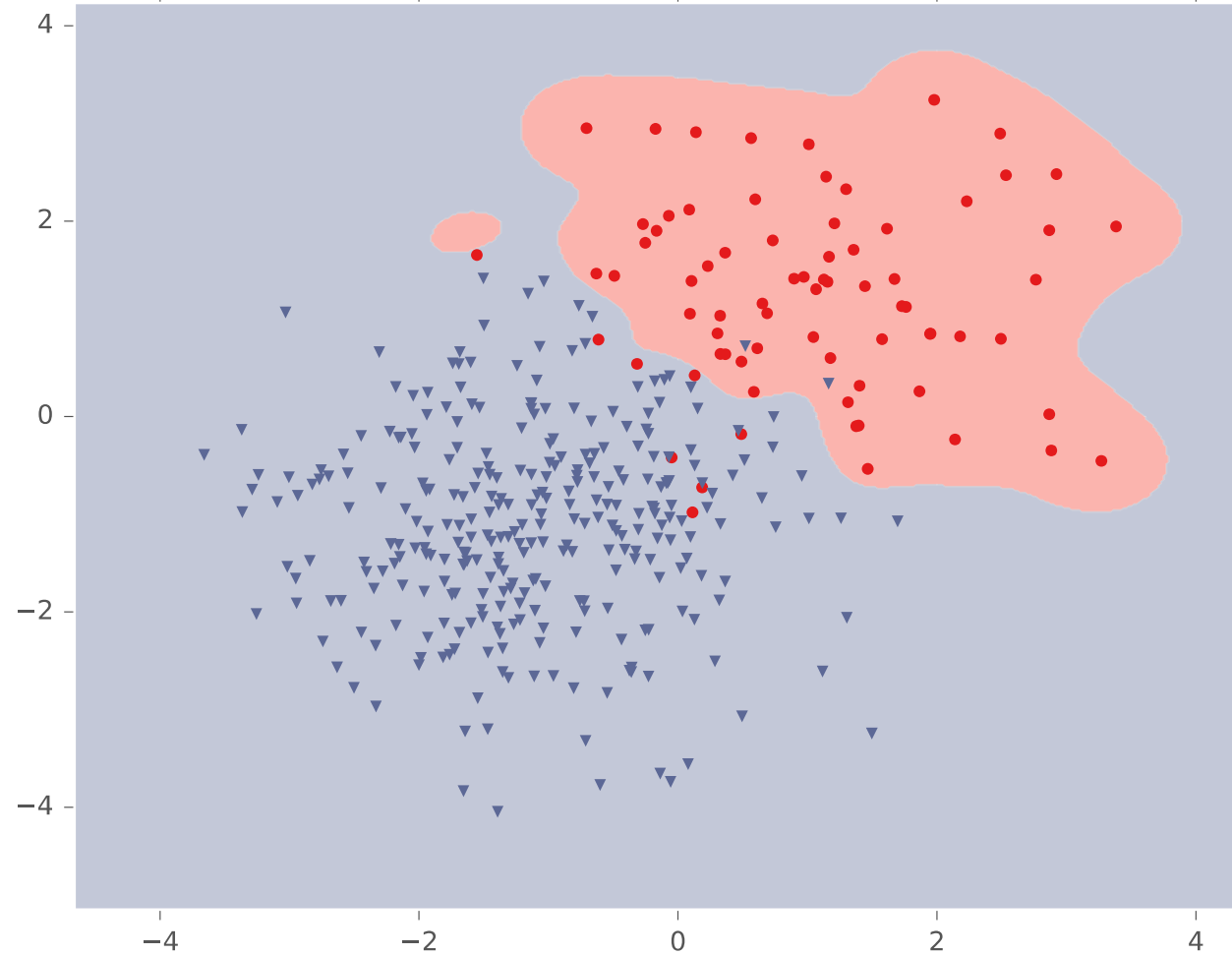
Classification with SVM (kernel=rbf, gamma=2.560000)



RBF Kernel: $K(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}) = \exp(-\gamma \|\mathbf{x}^{(i)} - \mathbf{x}^{(j)}\|_2^2)$

RBF Kernel Example

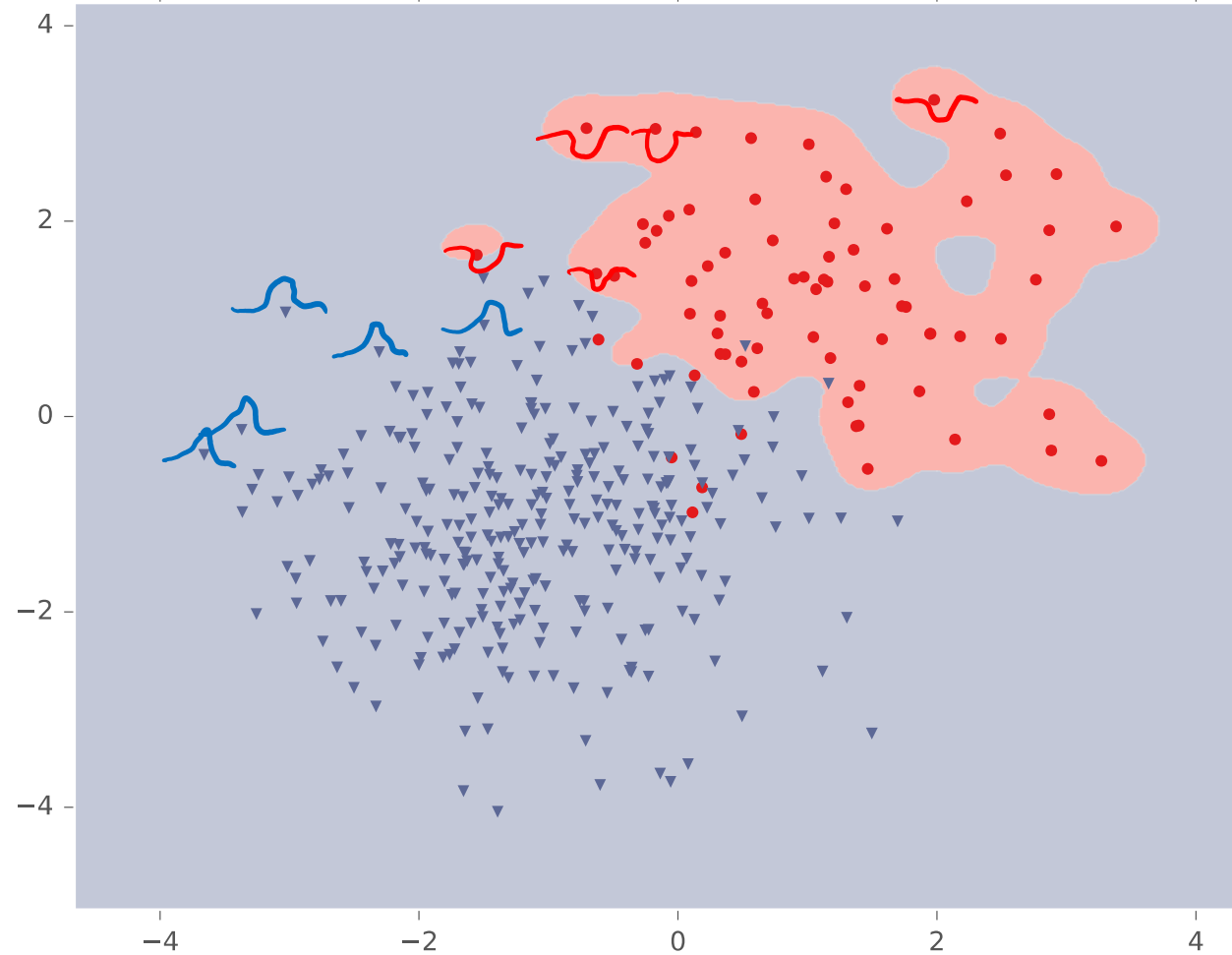
Classification with SVM (kernel=rbf, gamma=5.120000)



RBF Kernel: $K(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}) = \exp(-\gamma \|\mathbf{x}^{(i)} - \mathbf{x}^{(j)}\|_2^2)$

RBF Kernel Example

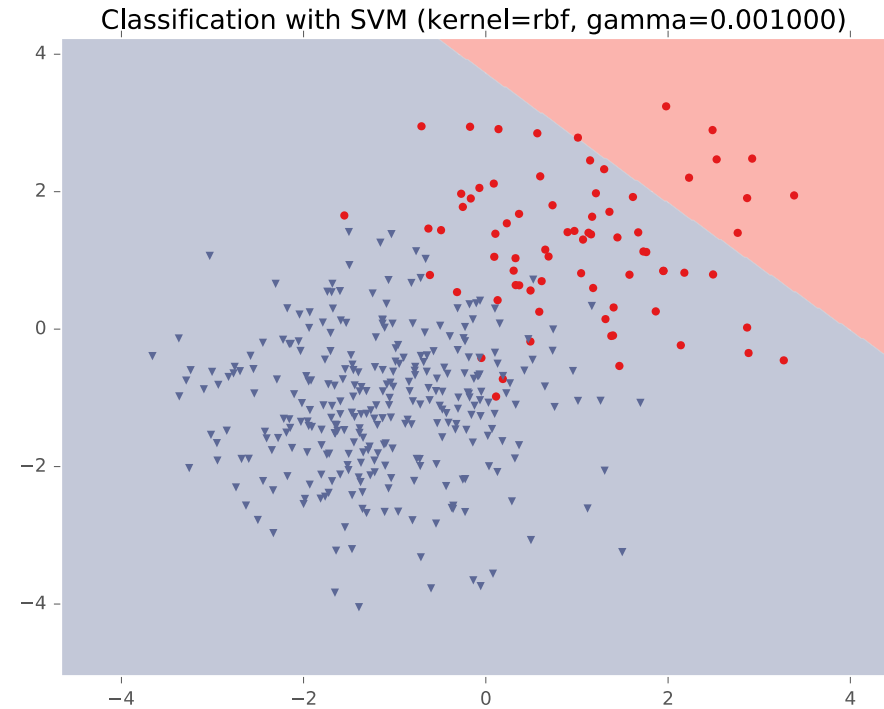
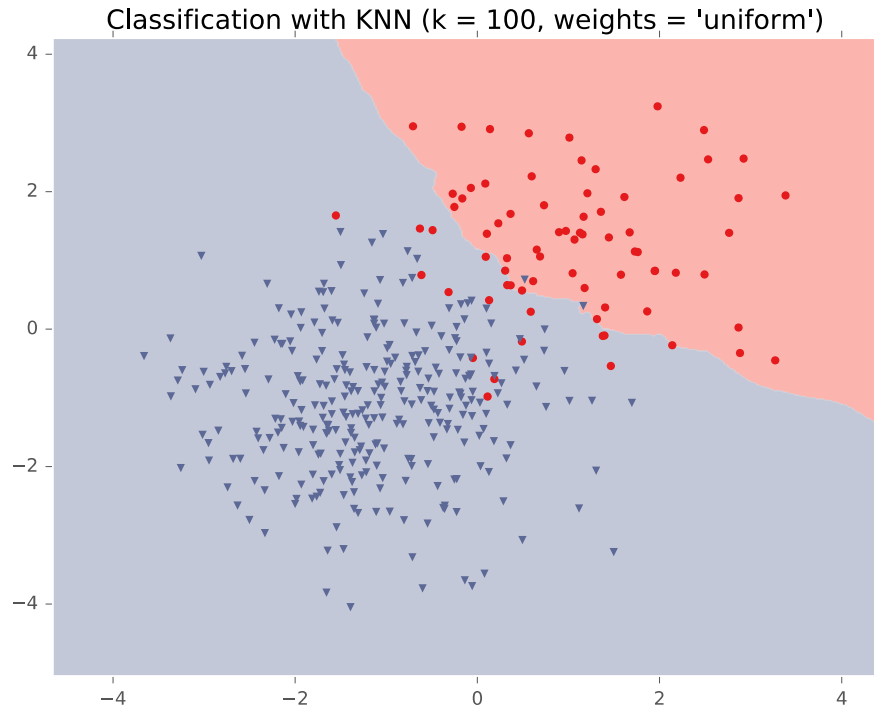
Classification with SVM (kernel=rbf, gamma=10.000000)



RBF Kernel: $K(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}) = \exp(-\gamma \|\mathbf{x}^{(i)} - \mathbf{x}^{(j)}\|_2^2)$

RBF Kernel Example

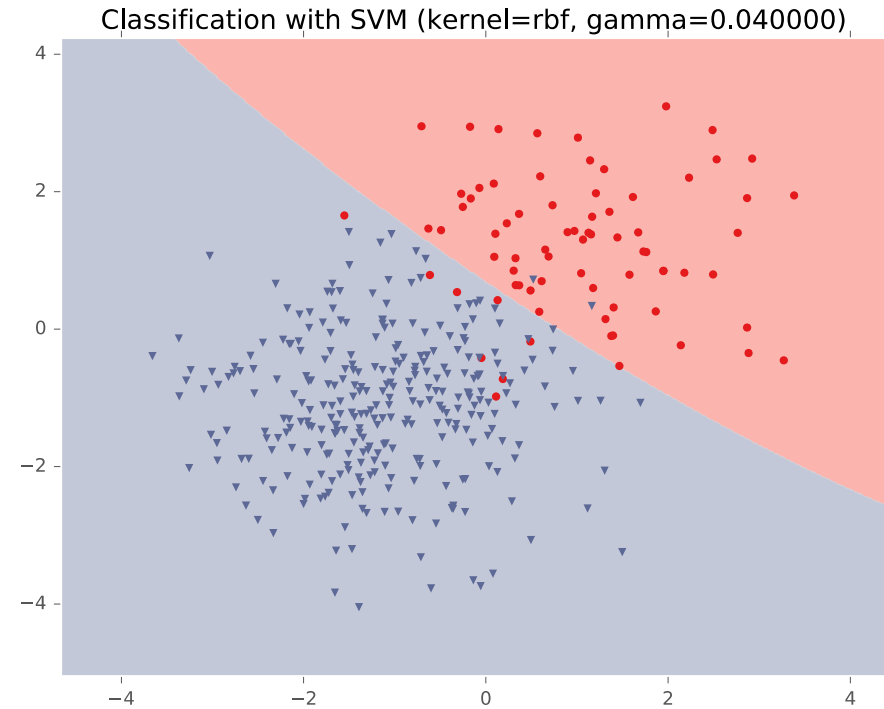
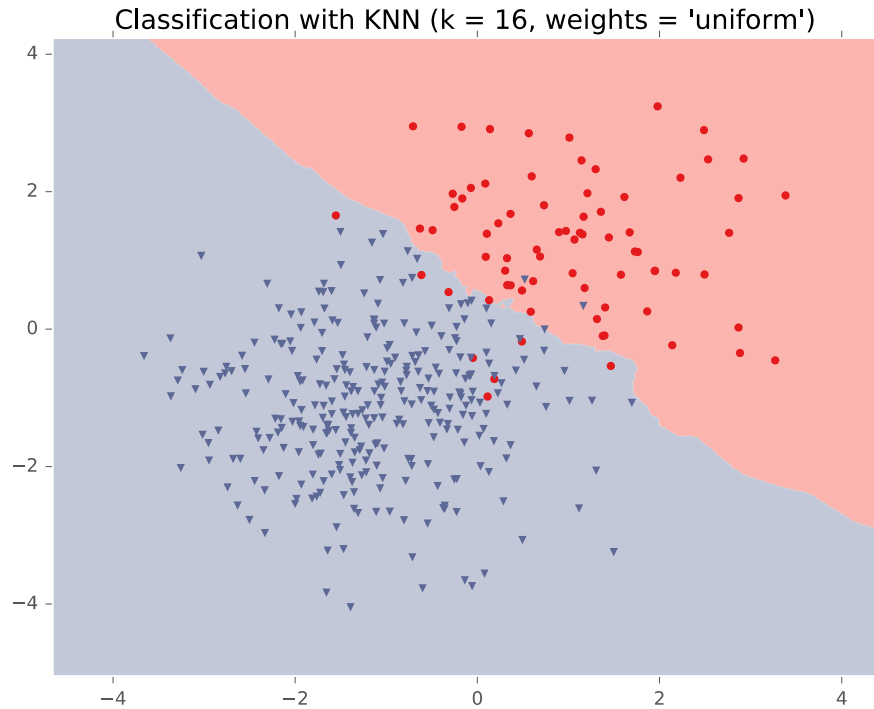
KNN vs. SVM



RBF Kernel: $K(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}) = \exp(-\gamma \|\mathbf{x}^{(i)} - \mathbf{x}^{(j)}\|_2^2)$

RBF Kernel Example

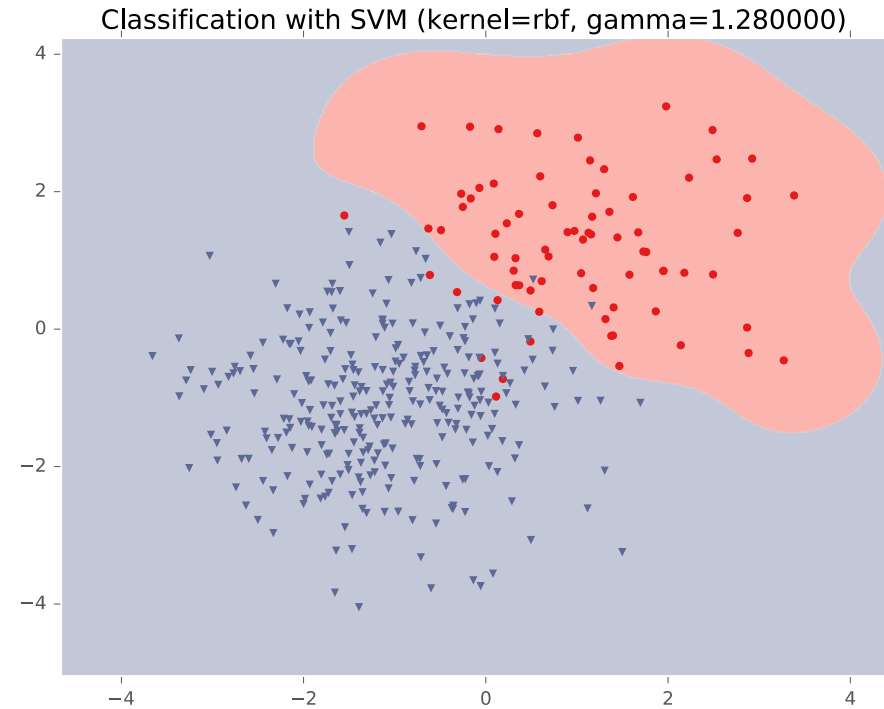
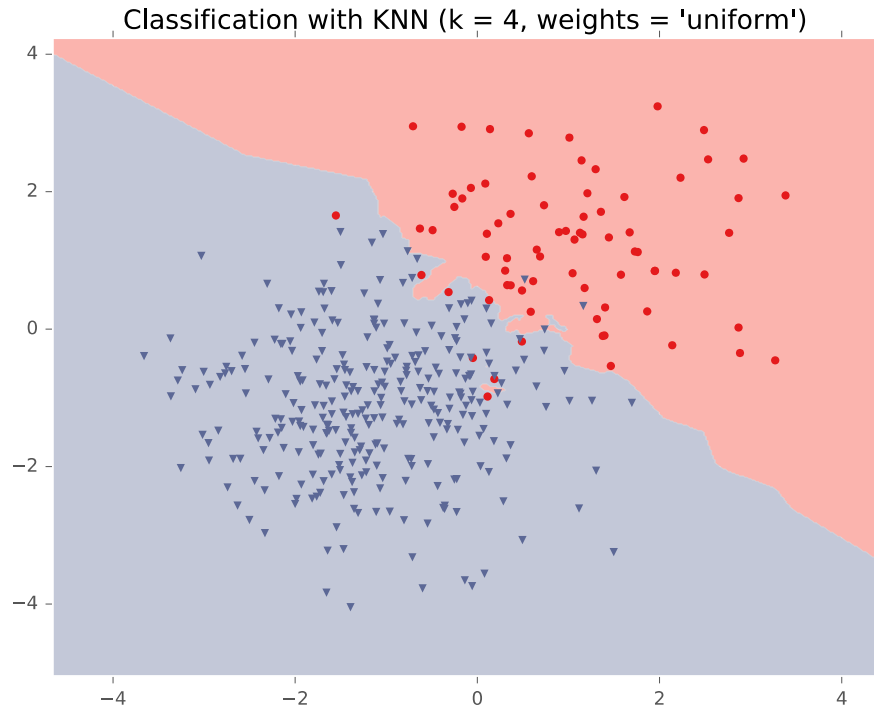
KNN vs. SVM



RBF Kernel: $K(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}) = \exp(-\gamma \|\mathbf{x}^{(i)} - \mathbf{x}^{(j)}\|_2^2)$

RBF Kernel Example

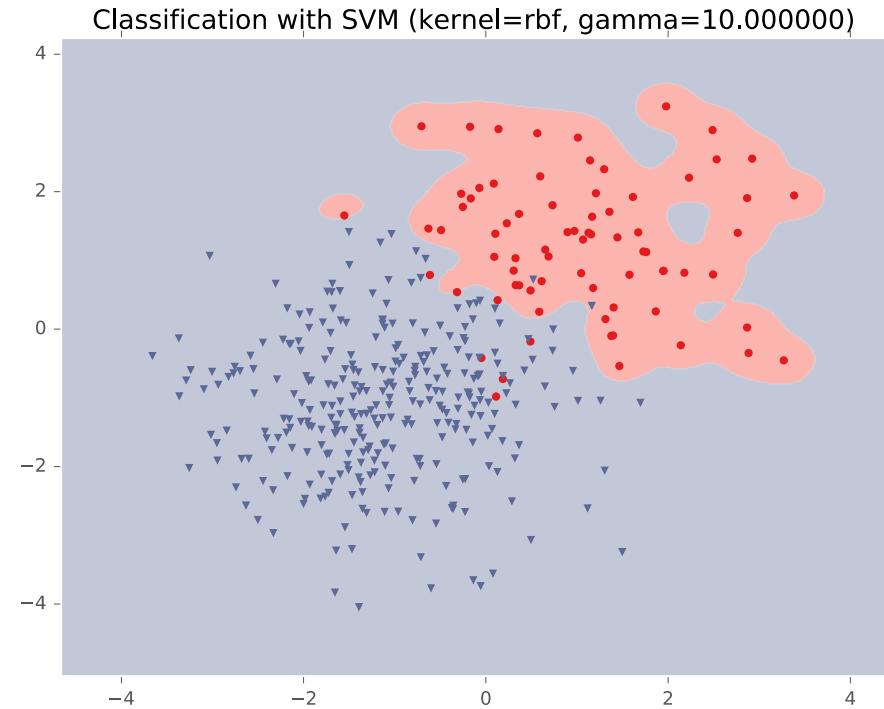
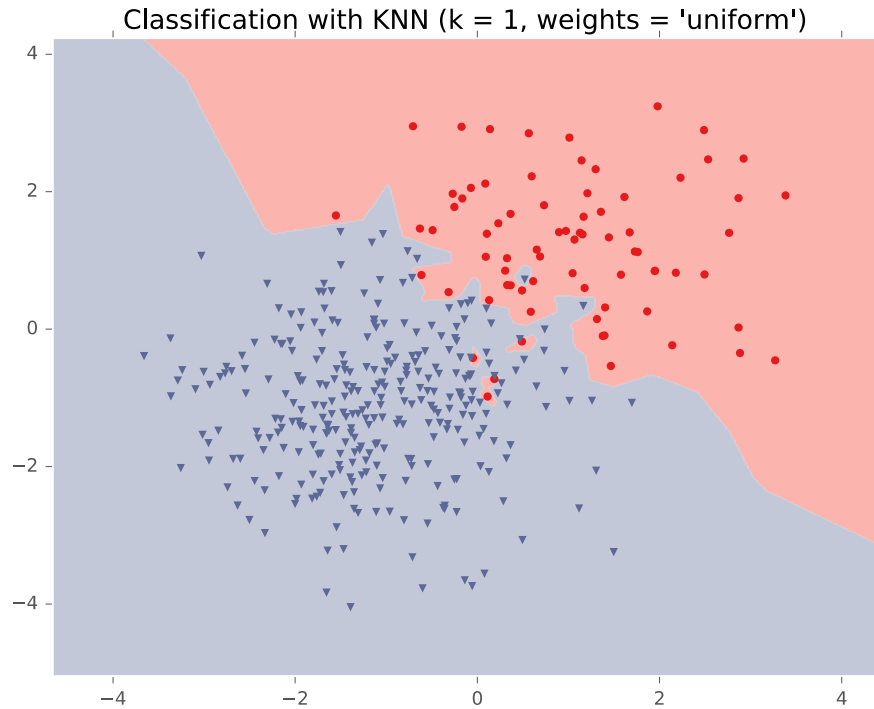
KNN vs. SVM



RBF Kernel: $K(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}) = \exp(-\gamma \|\mathbf{x}^{(i)} - \mathbf{x}^{(j)}\|_2^2)$

RBF Kernel Example

KNN vs. SVM



RBF Kernel: $K(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}) = \exp(-\gamma \|\mathbf{x}^{(i)} - \mathbf{x}^{(j)}\|_2^2)$

Kernel Methods

- **Key idea:**
 1. **Rewrite** the algorithm so that we only work with **dot products** $x^T z$ of feature vectors
 2. **Replace** the **dot products** $x^T z$ with a **kernel function** $k(x, z)$
- The kernel $k(x, z)$ can be **any** legal definition of a dot product:

$$k(x, z) = \varphi(x)^T \varphi(z) \text{ for any function } \varphi: X \rightarrow \mathbf{R}^D$$

So we only compute the φ dot product **implicitly**

- This “**kernel trick**” can be applied to many algorithms:
 - classification: perceptron, SVM, ...
 - regression: ridge regression, ...
 - clustering: k-means, ...

SVM + Kernels: Takeaways

- Maximizing the margin of a linear separator is a **good training criteria**
- Support Vector Machines (SVMs) learn a **max-margin linear classifier**
- The SVM optimization problem can be solved with **black-box Quadratic Programming (QP) solvers**
- Learned decision boundary is defined by its **support vectors**
- Kernel methods allow us to work in a transformed feature space **without explicitly representing that space**
- The **kernel-trick** can be applied to **SVMs**, as well as many other algorithms

Support Vector Machines

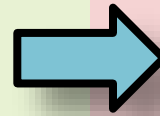
Next steps

- Different optimization formulation
 - Primal $\rightarrow$ dual
 - “Support vectors”
- Support non-linear classification
 - Feature maps
 - Kernel trick
- Support non-separable data
 - Hard-margin SVM $\rightarrow$ soft-margin SVM

Support Vector Machines (SVMs)

Hard-margin SVM (Primal)

$$\begin{aligned} \min_{\mathbf{w}, b} \quad & \frac{1}{2} \|\mathbf{w}\|_2^2 \\ \text{s.t.} \quad & y^{(i)}(\mathbf{w}^T \mathbf{x}^{(i)} + b) \geq 1, \quad \forall i = 1, \dots, N \end{aligned}$$



Hard-margin SVM (Lagrangian Dual)

$$\begin{aligned} \max_{\alpha} \quad & \sum_{i=1}^N \alpha_i - \frac{1}{2} \sum_{i=1}^N \sum_{j=1}^N \alpha_i \alpha_j y^{(i)} y^{(j)} \mathbf{x}^{(i)} \cdot \mathbf{x}^{(j)} \\ \text{s.t.} \quad & \alpha_i \geq 0, \quad \forall i = 1, \dots, N \\ & \sum_{i=1}^N \alpha_i y^{(i)} = 0 \end{aligned}$$

- Instead of minimizing the primal, we can maximize the dual problem
- For the SVM, these two problems give the same answer (i.e. the minimum of one is the maximum of the other)
- *Definition: support vectors* are those points $\mathbf{x}^{(i)}$ for which $\alpha^{(i)} \neq 0$

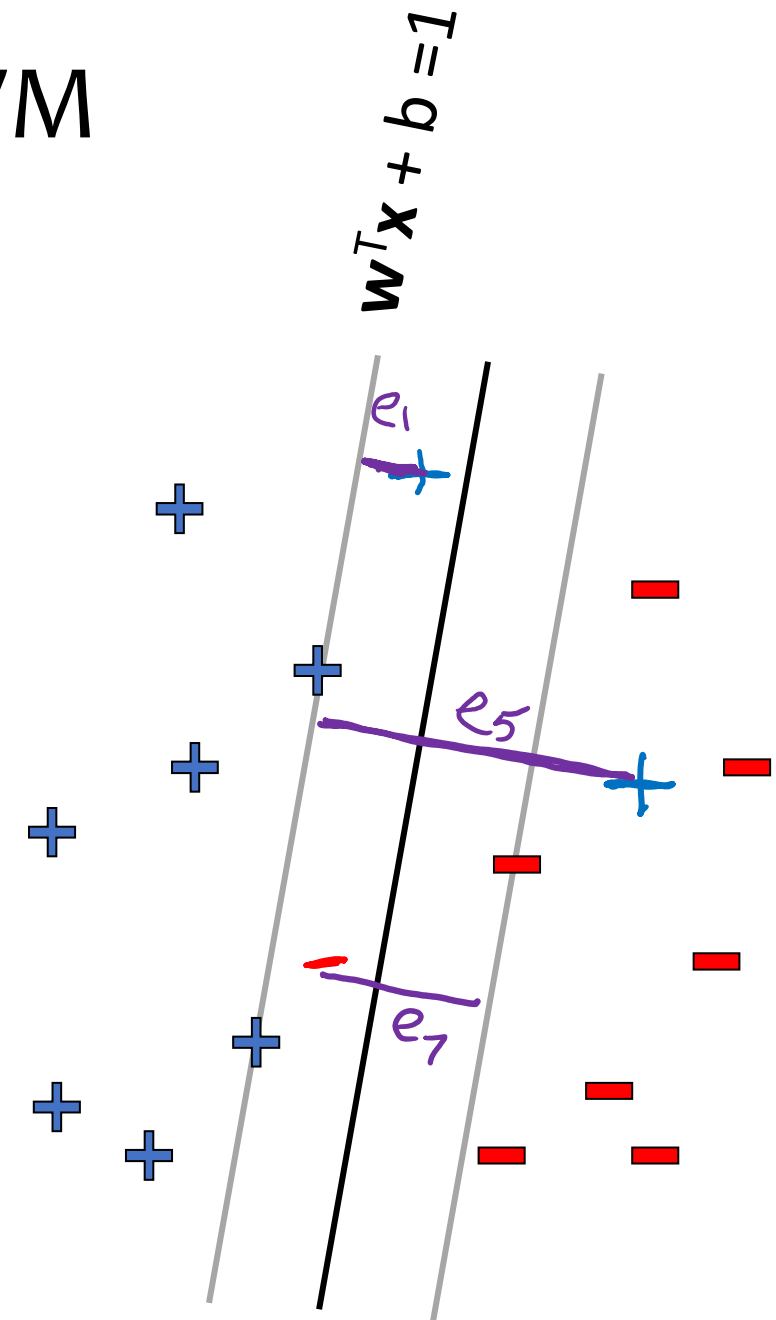
Soft-Margin SVM

Hard-margin SVM (Primal)

$$\begin{aligned} \min_{\mathbf{w}, b} \quad & \frac{1}{2} \|\mathbf{w}\|_2^2 \\ \text{s.t.} \quad & y^{(i)}(\mathbf{w}^T \mathbf{x}^{(i)} + b) \geq 1, \quad \forall i = 1, \dots, N \end{aligned}$$

Soft-margin SVM (Primal)

$$\begin{aligned} \min_{\mathbf{w}, b} \quad & \frac{1}{2} \|\mathbf{w}\|_2^2 + C \left(\sum_{i=1}^N e_i \right) \\ \text{s.t.} \quad & y^{(i)}(\mathbf{w}^T \mathbf{x}^{(i)} + b) \geq \underline{1 - e_i}, \quad \forall i = 1, \dots, N \\ & \underline{e_i} \geq 0, \quad \forall i = 1, \dots, N \end{aligned}$$



Soft-Margin SVM

Hard-margin SVM (Primal)

$$\begin{aligned} \min_{\mathbf{w}, b} \quad & \frac{1}{2} \|\mathbf{w}\|_2^2 \\ \text{s.t.} \quad & y^{(i)}(\mathbf{w}^T \mathbf{x}^{(i)} + b) \geq 1, \quad \forall i = 1, \dots, N \end{aligned}$$

Hard-margin SVM (Lagrangian Dual)

$$\begin{aligned} \max_{\alpha} \quad & \sum_{i=1}^N \alpha_i - \frac{1}{2} \sum_{i=1}^N \sum_{j=1}^N \alpha_i \alpha_j y^{(i)} y^{(j)} \mathbf{x}^{(i)} \cdot \mathbf{x}^{(j)} \\ \text{s.t.} \quad & \alpha_i \geq 0, \quad \forall i = 1, \dots, N \\ & \sum_{i=1}^N \alpha_i y^{(i)} = 0 \end{aligned}$$

Soft-margin SVM (Primal)

$$\begin{aligned} \min_{\mathbf{w}, b} \quad & \frac{1}{2} \|\mathbf{w}\|_2^2 + C \left(\sum_{i=1}^N e_i \right) \\ \text{s.t.} \quad & y^{(i)}(\mathbf{w}^T \mathbf{x}^{(i)} + b) \geq 1 - e_i, \quad \forall i = 1, \dots, N \\ & e_i \geq 0, \quad \forall i = 1, \dots, N \end{aligned}$$

Soft-margin SVM (Lagrangian Dual)

$$\begin{aligned} \max_{\alpha} \quad & \sum_{i=1}^N \alpha_i - \frac{1}{2} \sum_{i=1}^N \sum_{j=1}^N \alpha_i \alpha_j y^{(i)} y^{(j)} \mathbf{x}^{(i)} \cdot \mathbf{x}^{(j)} \\ \text{s.t.} \quad & 0 \leq \alpha_i \leq C, \quad \forall i = 1, \dots, N \\ & \sum_{i=1}^N \alpha_i y^{(i)} = 0 \end{aligned}$$

We can also work with the dual of the soft-margin SVM