

CS 480/680

Introduction to Machine Learning

Lecture 6

Support Vector Machines Part I

Maximum Margin Classifier and Constrained Optimization

Kathryn Simone

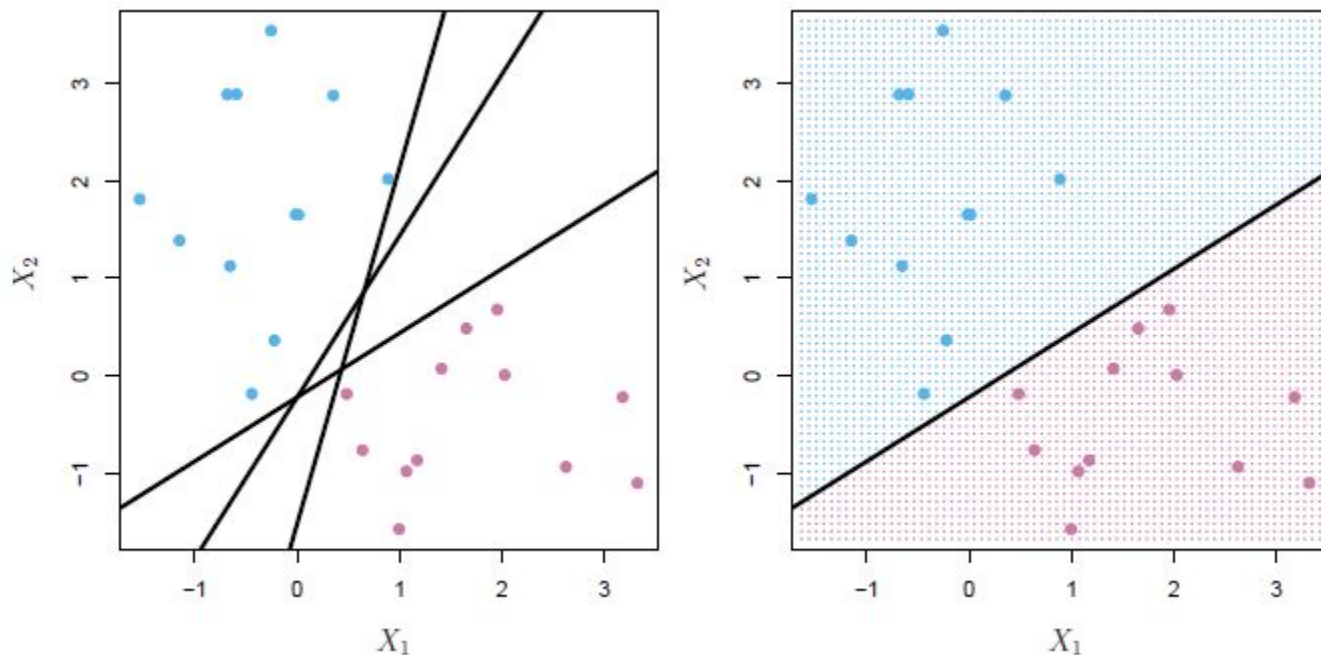
26 September 2024




UNIVERSITY OF
WATERLOO

FACULTY OF
MATHEMATICS

Large margin classifiers are more robust





The symbol  indicates a technically difficult section, one that can be skipped without interrupting the flow of the discussion.

4.5.2 *Optimal Separating Hyperplanes*



The *optimal separating hyperplane* separates the two classes and maximizes the distance to the closest point from either class (Vapnik, 1996). Not only does this provide a unique solution to the separating hyperplane problem, but by maximizing the margin between the two classes on the training data, this leads to better classification performance on test data.



Computing the distance to the decision boundary

For a hyperplane defined by:

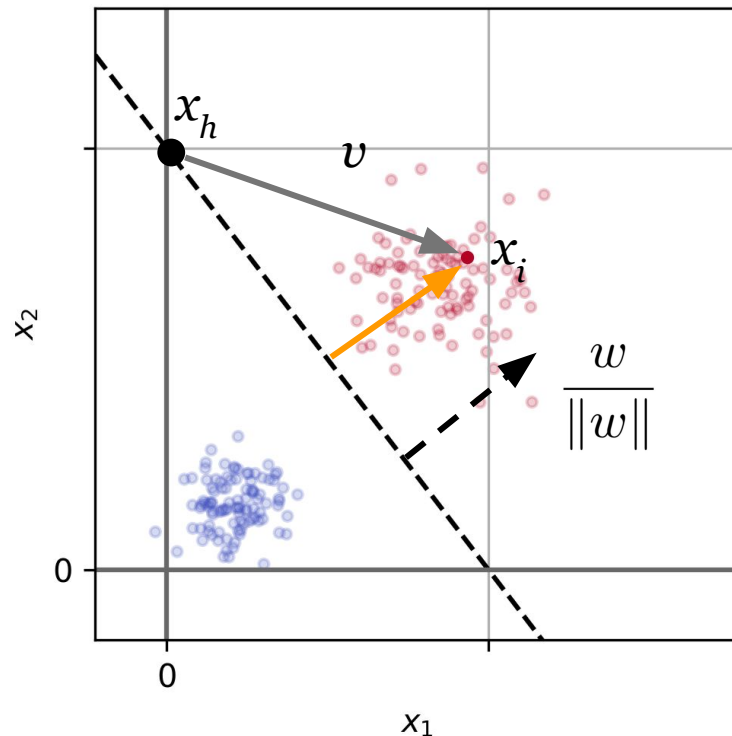
$$w^T x = -b$$

where $w, x \in \mathbb{R}^d, b \in \mathbb{R}$ the vector v from any point x_i and a point on the hyperplane x_h is

$$\begin{aligned} v &= x_i - x_h \\ \Rightarrow v &= x_i - \left(\frac{-b}{w} \right) \\ \Rightarrow v &= x_i + \frac{b}{w} \end{aligned}$$

The minimum distance d between the point x_i and the hyperplane is the projection of v onto the surface normal:

$$\begin{aligned} d &= \frac{w}{\|w\|} \cdot v \\ &= \frac{w}{\|w\|} \left(x_i + \frac{b}{w} \right) \\ &= \frac{1}{\|w\|} (w^T x_i + b) \end{aligned}$$



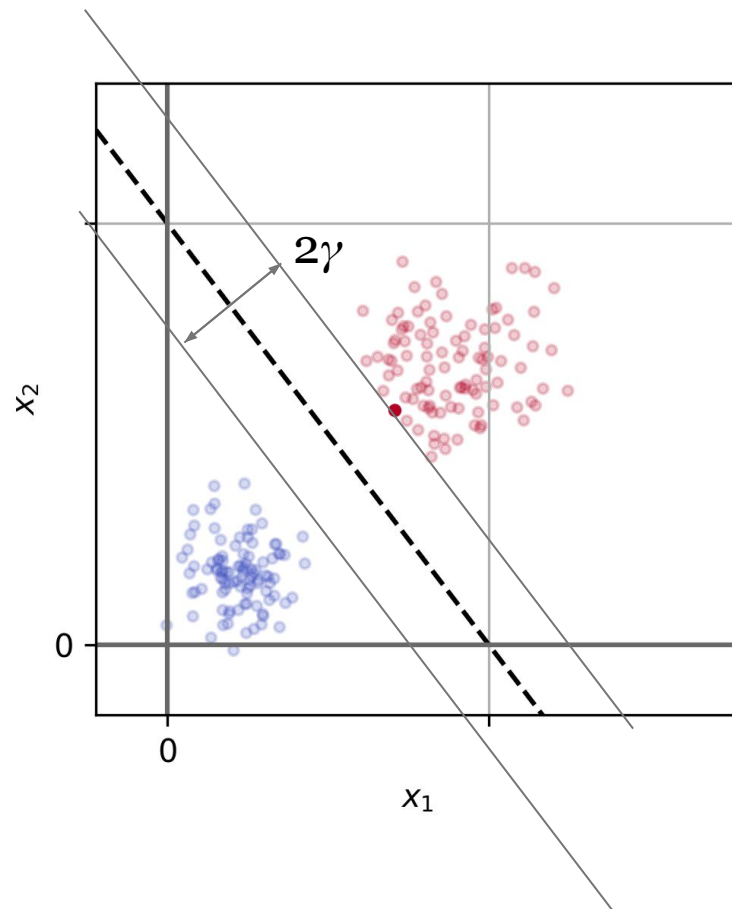
The margin is the distance from the boundary to the closest point

For a dataset of samples $x_i \in \mathbb{R}^d$, with labels $y_i \in \{\pm 1\}$, linearly separated by a hyperplane parametrized by w and b , the margin γ is the distance from the hyperplane to the closest point:

$$\gamma = \min_i \frac{1}{\|w\|} y_i (w^T x_i + b)$$

Where y_i converts the signed distance to an unsigned distance. The factor $1/\|w\|$ can then be taken outside of the optimization over i , because the w does not depend on i :

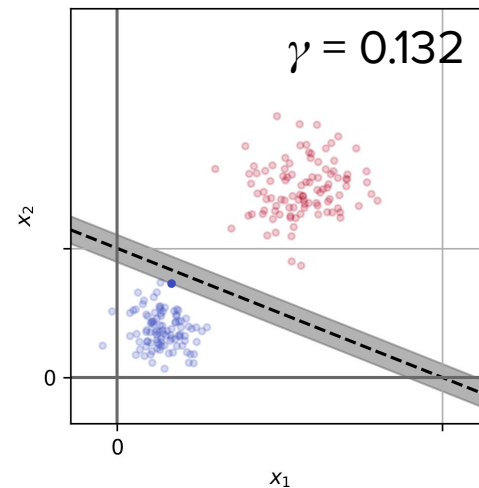
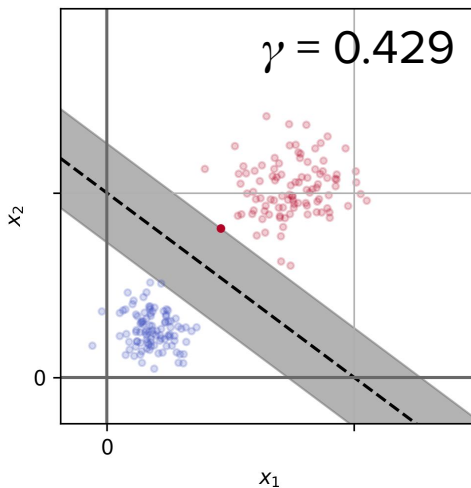
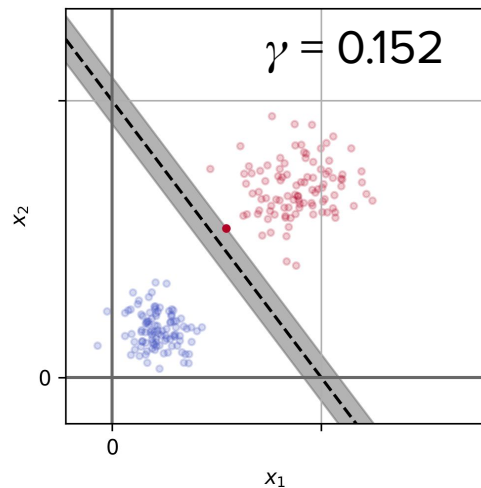
$$\gamma = \frac{1}{\|w\|} \min_i y_i (w^T x_i + b)$$



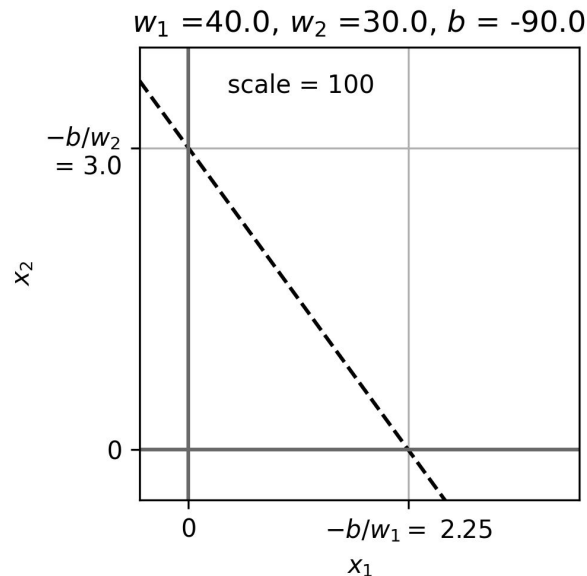
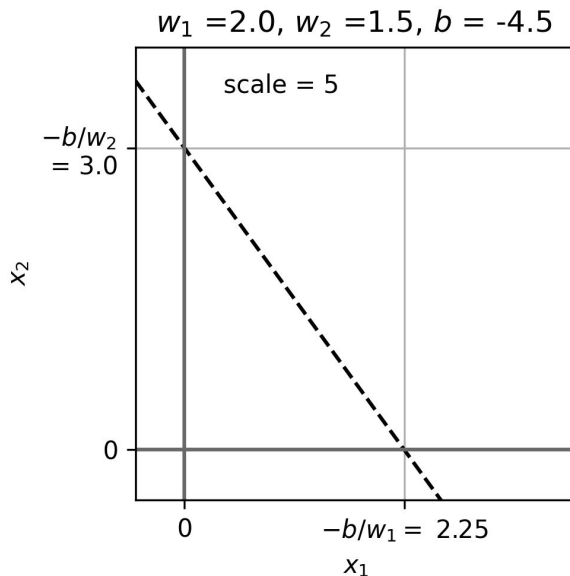
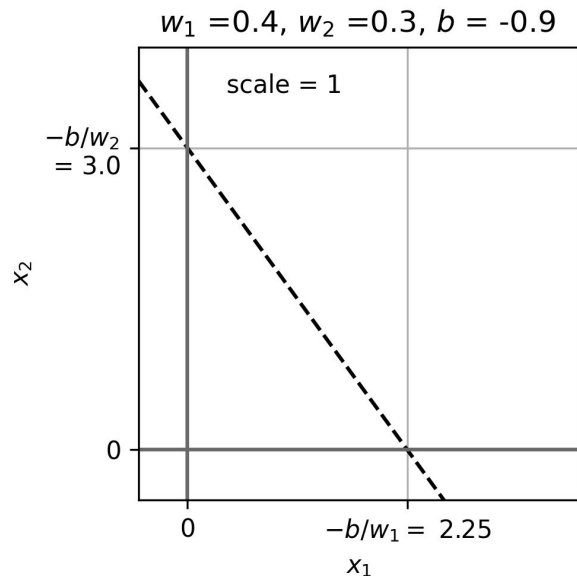
The maximum margin classifier problem

$$\hat{w}, \hat{b} = \operatorname{argmax}_{w, b} \gamma$$

$$\hat{w}, \hat{b} = \operatorname{argmax}_{w, b} \frac{1}{\|w\|} \min_i y_i (w^T x_i + b)$$



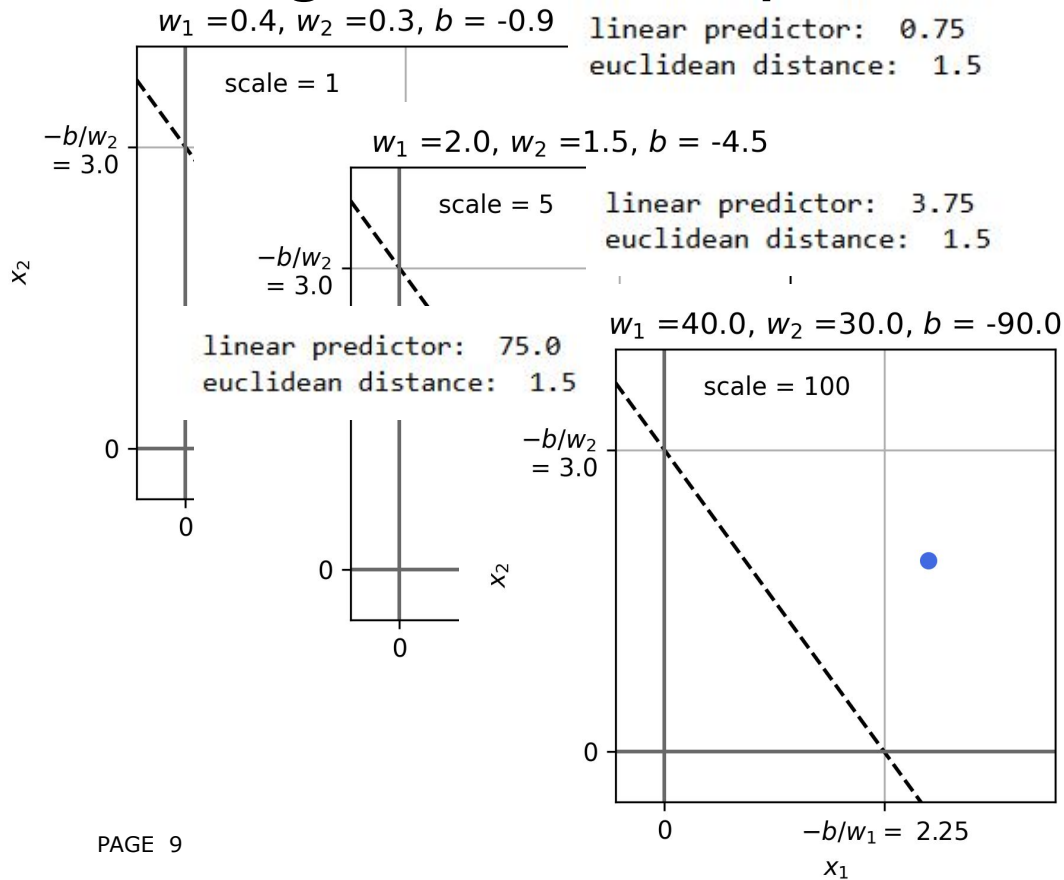
Scaling parameters doesn't change decision boundary



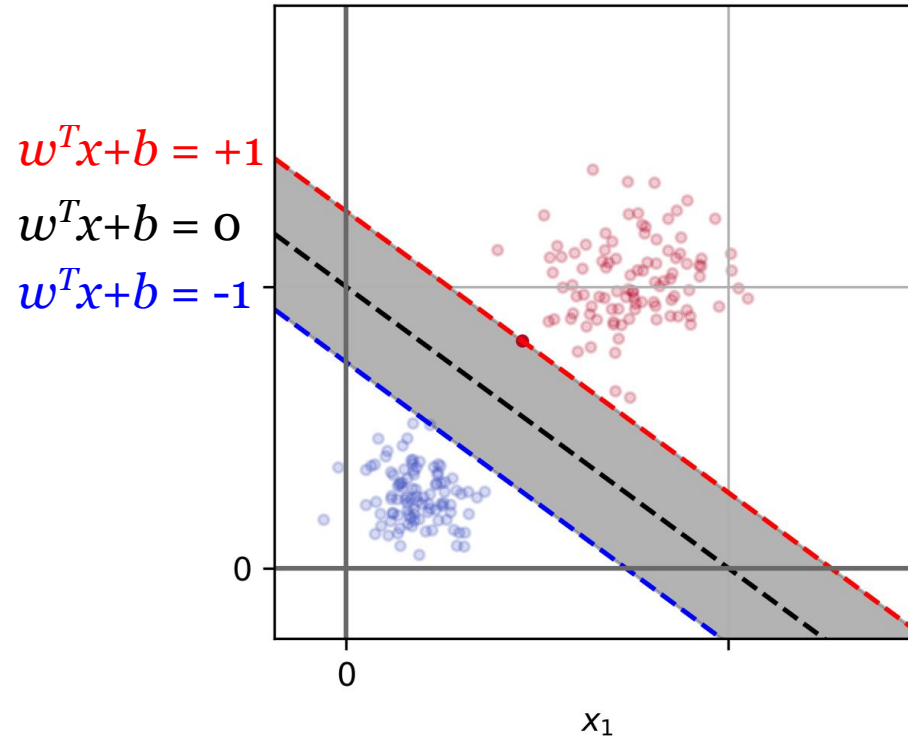
Scaling parameters doesn't change distance to points

Rescaling parameters by some scalar α , such that $w \rightarrow \alpha w$ and $b \rightarrow \alpha b$, does not change the decision boundary or the distance of any point x_i to the boundary:

$$\begin{aligned}d &= \frac{1}{\|\alpha w\|}(\alpha w^T x_i + \alpha b) \\&= \frac{\alpha}{\alpha} \frac{1}{\|w\|}(w^T x_i + b) \\&= \frac{1}{\|w\|}(w^T x_i + b)\end{aligned}$$



The canonical representation of the decision hyperplane



Maximum margin classification as constrained optimization

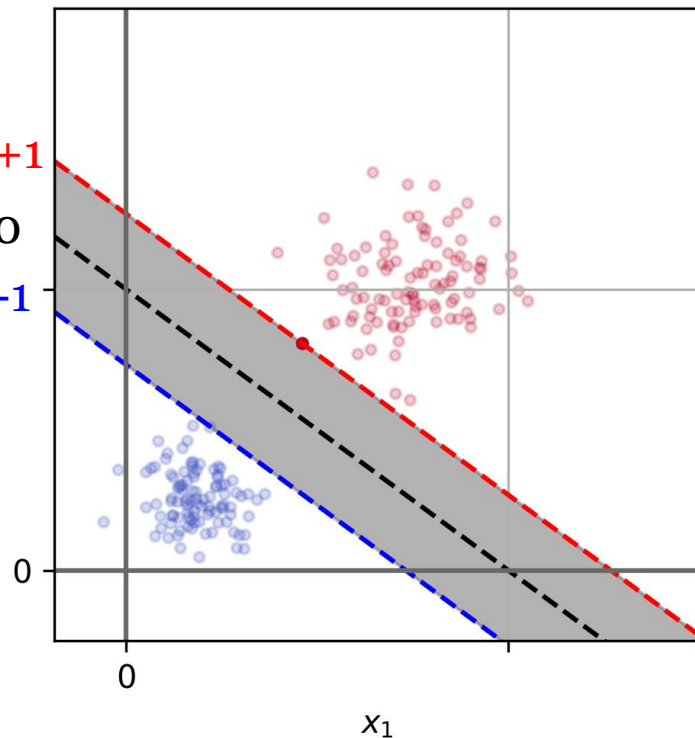
We are interested in finding the parameters \hat{w}, \hat{b} that maximize the margin γ

$$\begin{aligned}\hat{w}, \hat{b} &= \operatorname{argmax}_{w, b} \gamma \\ \implies \hat{w}, \hat{b} &= \operatorname{argmax}_{w, b} \frac{1}{\|w\|} \min_i y_i (w^T x_i + b) \\ &= \operatorname{argmax}_{w, b} \frac{1}{\|w\|} \quad \text{subject to: } y_i (w^T x_i + b) \geq 1 \quad \forall i \\ &= \operatorname{argmin}_{w, b} \frac{1}{2} \|w\|^2 \quad \text{subject to: } y_i (w^T x_i + b) \geq 1 \quad \forall i\end{aligned}$$

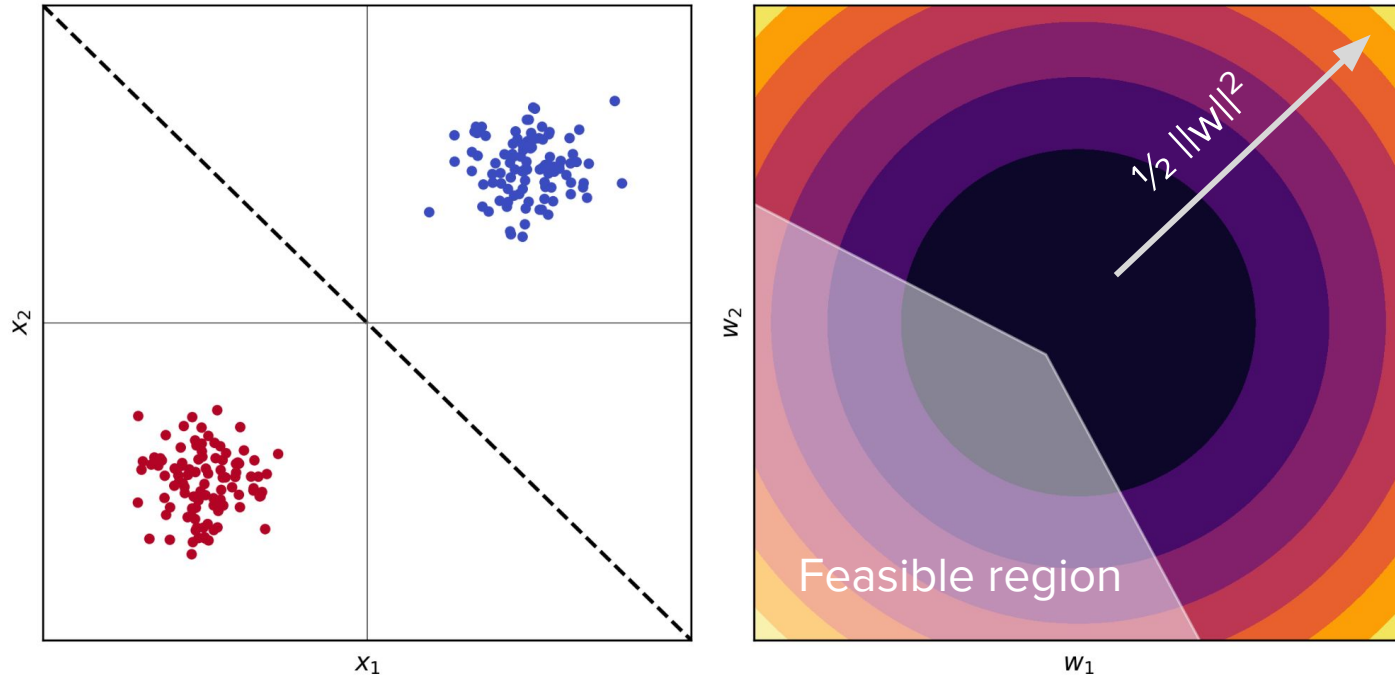
$$w^T x + b = +1$$

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

$$w^T x + b = -1$$



Constrained optimization problem



Constrained optimization with Lagrange multipliers

Consider the constrained objective

$$\begin{aligned}\min y(x) &= 0.2x^2 - x + 1 \\ \text{subject to: } &x \geq 5\end{aligned}$$

This can be transformed into the *dual* problem

$$\begin{aligned}\max \mathcal{L}(x, \lambda) &= 0.2x^2 - x + 1 - \lambda(x - 5) \\ \text{subject to: } &\lambda \geq 0\end{aligned}$$

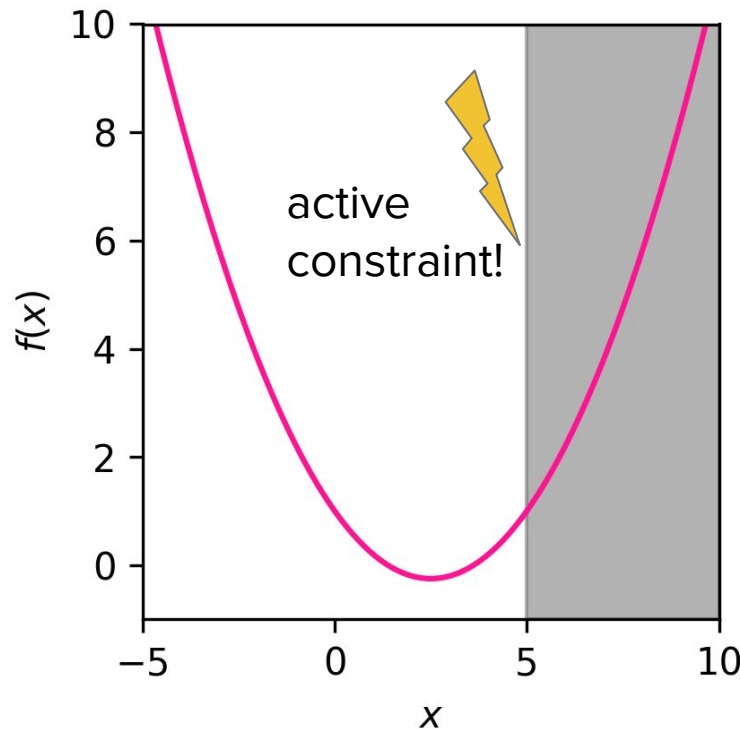
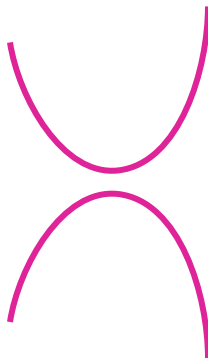
Where λ is a Lagrange multiplier. To maximize it, we compute the partial derivatives and set them to zero:

$$\begin{aligned}\frac{\partial \mathcal{L}}{\partial x} &= 2(0.2)x - 1 - \lambda \\ &= 0.4x - 1 - \lambda = 0\end{aligned}$$

$$\frac{\partial \mathcal{L}}{\partial \lambda} = -(x - 5) = 0$$

$$\implies x = 5$$

$$\implies \lambda = 0.4(5) - 1 = 2 - 1 = 1$$



Constrained optimization with Lagrange multipliers

$$\min y(x) = 0.2x^2 - x + 1$$

$$\text{subject to: } x \geq -3$$

$$\max \mathcal{L}(x, \lambda) = 0.2x^2 - x + 1 - \lambda(x + 3)$$

$$\text{subject to: } \lambda \geq 0$$

$$\frac{\partial \mathcal{L}}{\partial x} = 0.4x - 1 - \lambda = 0$$

$$\frac{\partial \mathcal{L}}{\partial \lambda} = -(x + 3) = 0$$

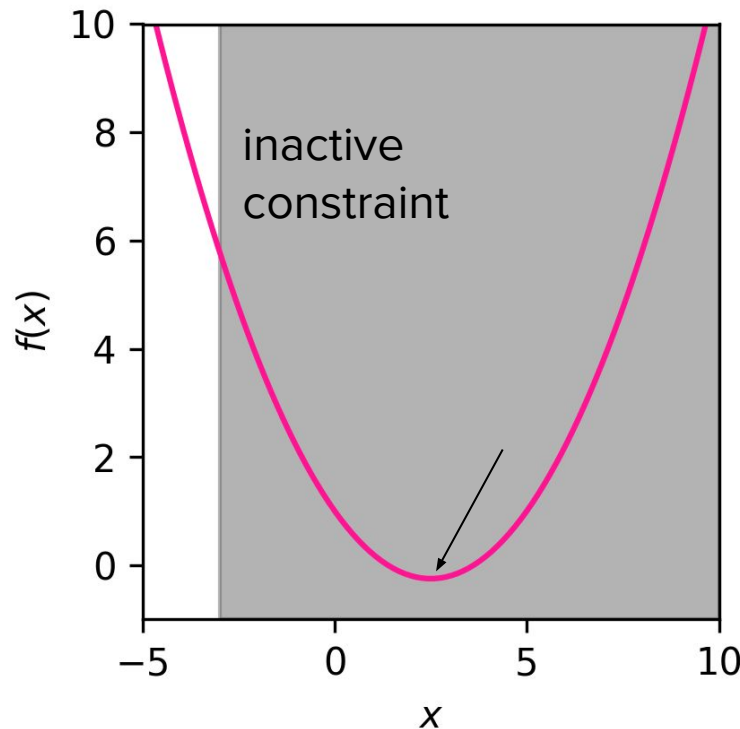
$$\implies x = -3$$

$$\implies \lambda = 0.4(-3) - 1 = -2.2$$

As the solution does not satisfy $\lambda \geq 0$, $x = -3$ is not a feasible solution to the dual problem.

In this case, $\lambda = 0$, and we solve for x

$$x = \frac{1}{0.4} = 2.5$$



Lagrangian dual of maximum margin classification problem

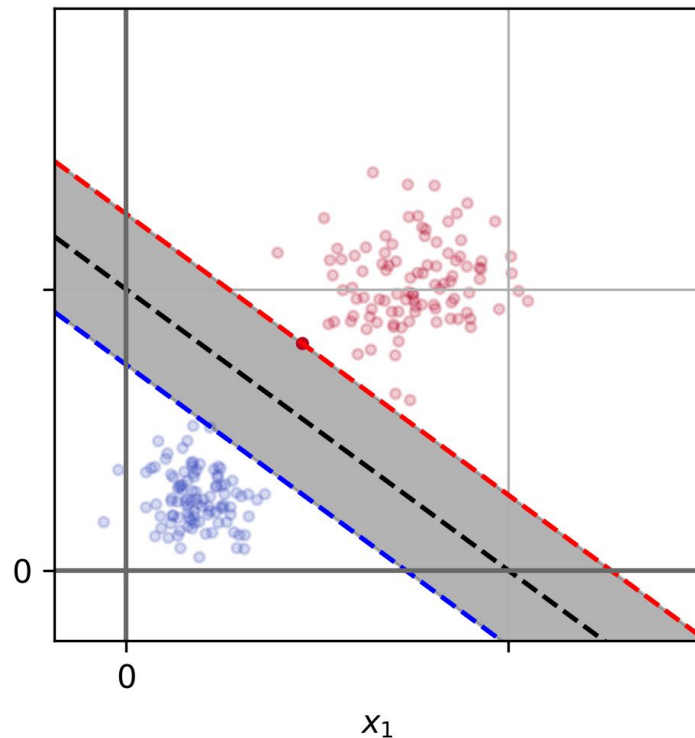
$$\hat{w}, \hat{b} = \operatorname{argmax}_{w, b} \frac{1}{\|w\|} \min_i y_i (w^T x_i + b)$$

$$\hat{w}, \hat{b} = \operatorname{argmin}_{w, b} \frac{1}{2} \|w\|^2$$

$$\text{subject to: } y_i (w^T x_i + b) \geq 1 \quad \forall i$$

$$\mathcal{L}(w, b, \lambda) = \frac{1}{2} \|w\|^2 - \sum_{i=1}^n \lambda_i (y_i (w^T x_i + b) - 1)$$

$$\text{subject to: } \lambda_i \geq 0 \quad \forall i$$



Karush-Kuhn-Tucker conditions and support vectors

Primal feasibility : $y_i(w^T x_i + b) - 1 \geq 0$

Dual feasibility : $\lambda_i \geq 0$

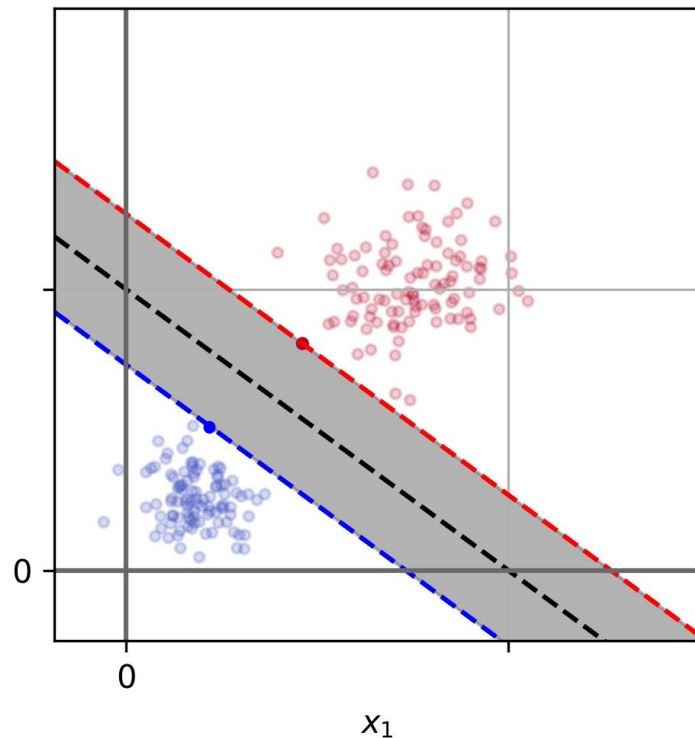
Complementary slackness : $\lambda_i(y_i(w^T x_i + b) - 1) \geq 0$

$\lambda_i > 0 \implies y_i(w^T x_i + b) = 1$

constraint is active, x_i defines the margin

$y_i(w^T x_i + b) > 1 \implies \lambda_i = 0$

constraint is inactive, x_i is far from the margin



Maximizing the Lagrangian dual

$$\mathcal{L}(w, b, \lambda) = \frac{1}{2} \|w\|^2 - \sum_{i=1}^n \lambda_i (y_i (w^T x_i + b) - 1)$$

$$\mathcal{L}(w, b, \lambda) = \frac{1}{2} \|w\|^2 - \sum_{i=1}^n \lambda_i y_i w^T x_i - \sum_{i=1}^n \lambda_i y_i b + \sum_{i=1}^n \lambda_i$$

$$\frac{\partial \mathcal{L}}{\partial w} = w - \sum_{i=1}^n \lambda_i y_i x_i = 0$$

$$\implies w = \sum_{i=1}^n \lambda_i y_i x_i$$

$$\frac{\partial \mathcal{L}}{\partial b} = \sum_{i=1}^n \lambda_i y_i = 0$$

$$\implies \sum_{i=1}^n \lambda_i y_i = 0$$

Apply derived constraints to solve for λ_i

$$\begin{aligned}
 \mathcal{L}(w, b, \lambda) &= \frac{1}{2} \|w\|^2 - \sum_{i=1}^n \lambda_i y_i w^T x_i - \sum_{i=1}^n \lambda_i y_i b + \sum_{i=1}^n \lambda_i & w &= \sum_{i=1}^n \lambda_i y_i x_i & \sum_{i=1}^n \lambda_i y_i &= 0 & \lambda_i &\geq 0 \\
 &= \frac{1}{2} \left(\sum_{i=1}^n \lambda_i y_i x_i \right)^T \left(\sum_{j=1}^n \lambda_j y_j x_j \right) - \sum_{i=1}^n \lambda_i y_i \left(\sum_{j=1}^n \lambda_j y_j x_j \right) x_i - \sum_{i=1}^n \lambda_i y_i b + \sum_{i=1}^n \lambda_i \\
 &= \frac{1}{2} \sum_{i=1}^n \sum_{j=1}^n \lambda_i \lambda_j y_i y_j x_i x_j - \sum_{i=1}^n \sum_{j=1}^n \lambda_i \lambda_j y_i y_j x_i x_j - \cancel{\sum_{i=1}^n \lambda_i y_i b} + \sum_{i=1}^n \lambda_i \\
 &= -\frac{1}{2} \sum_{i=1}^n \sum_{j=1}^n \lambda_i \lambda_j y_i y_j x_i x_j + \sum_{i=1}^n \lambda_i \\
 &= \sum_{i=1}^n \lambda_i - \frac{1}{2} \sum_{i=1}^n \sum_{j=1}^n \lambda_i \lambda_j y_i y_j x_i x_j & \lambda_i &\geq 0
 \end{aligned}$$

Solve for λ_i

$$\sum_{i=1}^n \lambda_i - \frac{1}{2} \sum_{i=1}^n \sum_{j=1}^n \lambda_i \lambda_j y_i y_j x_i x_j$$

$$\lambda_i \geq 0$$

$$\sum_{i=1}^n \lambda_i y_i = 0$$

λ_i

```
import cvxpy as cp
import numpy as np

# Problem data.
m = 30
n = 20
np.random.seed(1)
A = np.random.randn(m, n)
b = np.random.randn(m)

# Construct the problem.
x = cp.Variable(n)
objective = cp.Minimize(cp.sum_squares(A @ x - b))
constraints = [0 <= x, x <= 1]
prob = cp.Problem(objective, constraints)

# The optimal objective value is returned by `prob.solve()`.
result = prob.solve()
# The optimal value for x is stored in `x.value`.
print(x.value)
# The optimal Lagrange multiplier for a constraint is stored in
# `constraint.dual_value`.
print(constraints[0].dual_value)
```

Solving for b

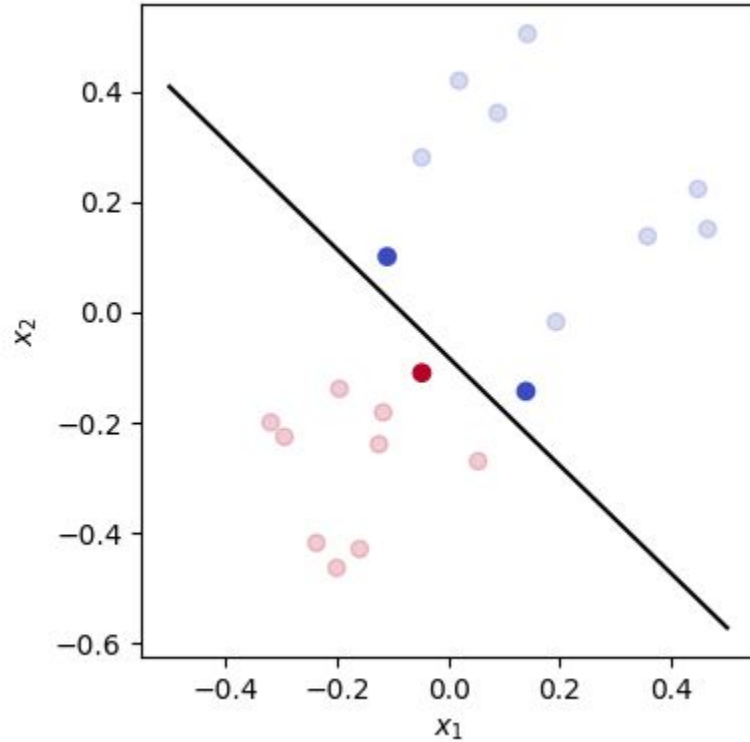
$$y_i(w^T x_i + b) = 1, x_i \quad x_i \in S_v$$

$$\implies b = \frac{1}{y_i} - w^T x_i$$

$$= y_i - w^T x_i$$

$$\approx \frac{1}{N_{sv}} \sum_{i \in S_v} y_i - w^T x_i$$

Does it work?



Now that we're at the end of the lecture, you should be able to...

- ★ Motivate the need for a large-margin classifier with reference to properties of **robustness, error sensitivity, and generalization**.
- ★ Define **margin** and recognize the maximum margin classification problem.
- ★ Apply **Lagrangian multipliers** to derive the **dual of an optimization problem** with **inequality constraints**.
- ★ Interpret the meaning of the solution to the dual objective of the maximum margin classifier, with reference to **support vectors, active/inactive constraints**, and the **KKT conditions**.
- ★ Apply off-the-shelf solution to solve the **quadratic programming problem**.
- ★ Use the solution to the dual objective to **predict class labels** and **identify the support vectors**.
- ★ Defend the utility of SVMs with reference to **sparsity**.