CS 446/ECE 449: Machine Learning

Lecture 5: Support Vector Machine

Han Zhao
01/30/2024

ILLINOIS
Computer Science
GRAINGER COLLEGE OF ENGINEERING

Recap: Logistic Regression

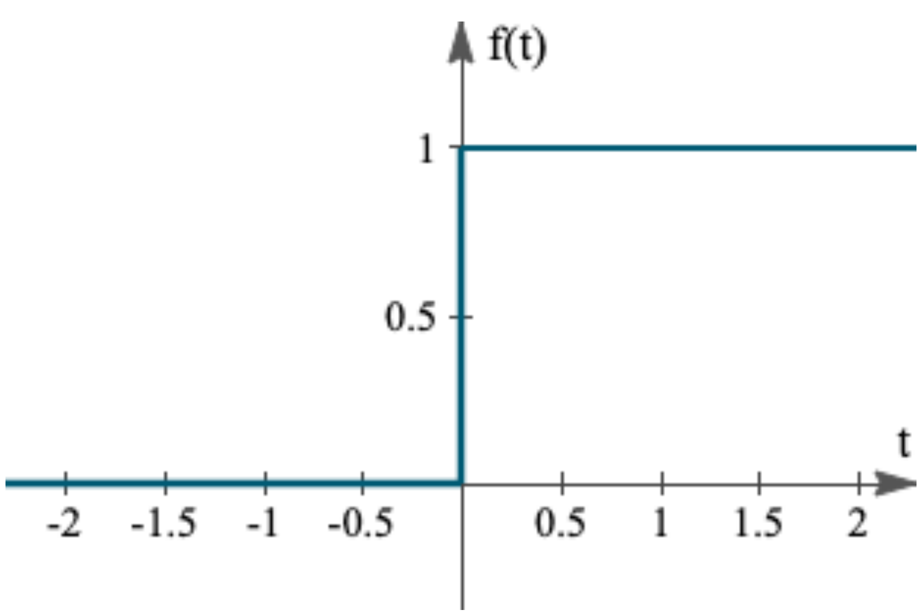
Linear classifier for binary classification:

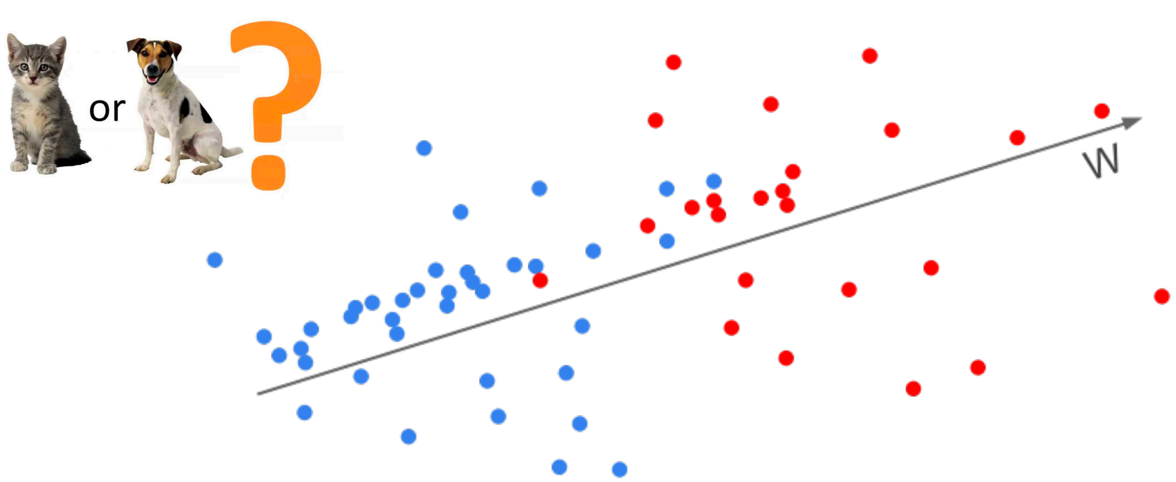
Given a weight vector $w \in \mathbb{R}^d$ and an intercept $b \in \mathbb{R}$, the decision rule of a linear classifier:

$$f(x) = \operatorname{sgn}(w^\top x + b) = \operatorname{sgn}\left(\sum_{i=1}^{d} w_i x_i + b\right)$$

where $\operatorname{sgn}(\cdot)$ is the sign function:

$$\operatorname{sgn}(t) = \begin{cases} 1 & \text{if } t \geq 0 \\ 0 & \text{o.w.} \end{cases}$$





Recap: Logistic Regression

Linear classifier for binary classification:

The 0-1 loss minimization problem:

$$\min_{w} \quad \frac{1}{n}\sum_{i=1}^{n} \mathbb{I}\left(\operatorname{sgn}\left(w^\top x^{(i)} + b\right) \neq y^{(i)}\right)$$

where $\mathbb{I}(E) = 1$ iff the event E is true otherwise 0.

Cons:

- Loss function is dis-continuous, non-convex
- Gradient is 0 a.e., not informative for local improvement
- Intractable to solve the opt. (NP-hard, even approximately)

Pros:

- Robust to outliers

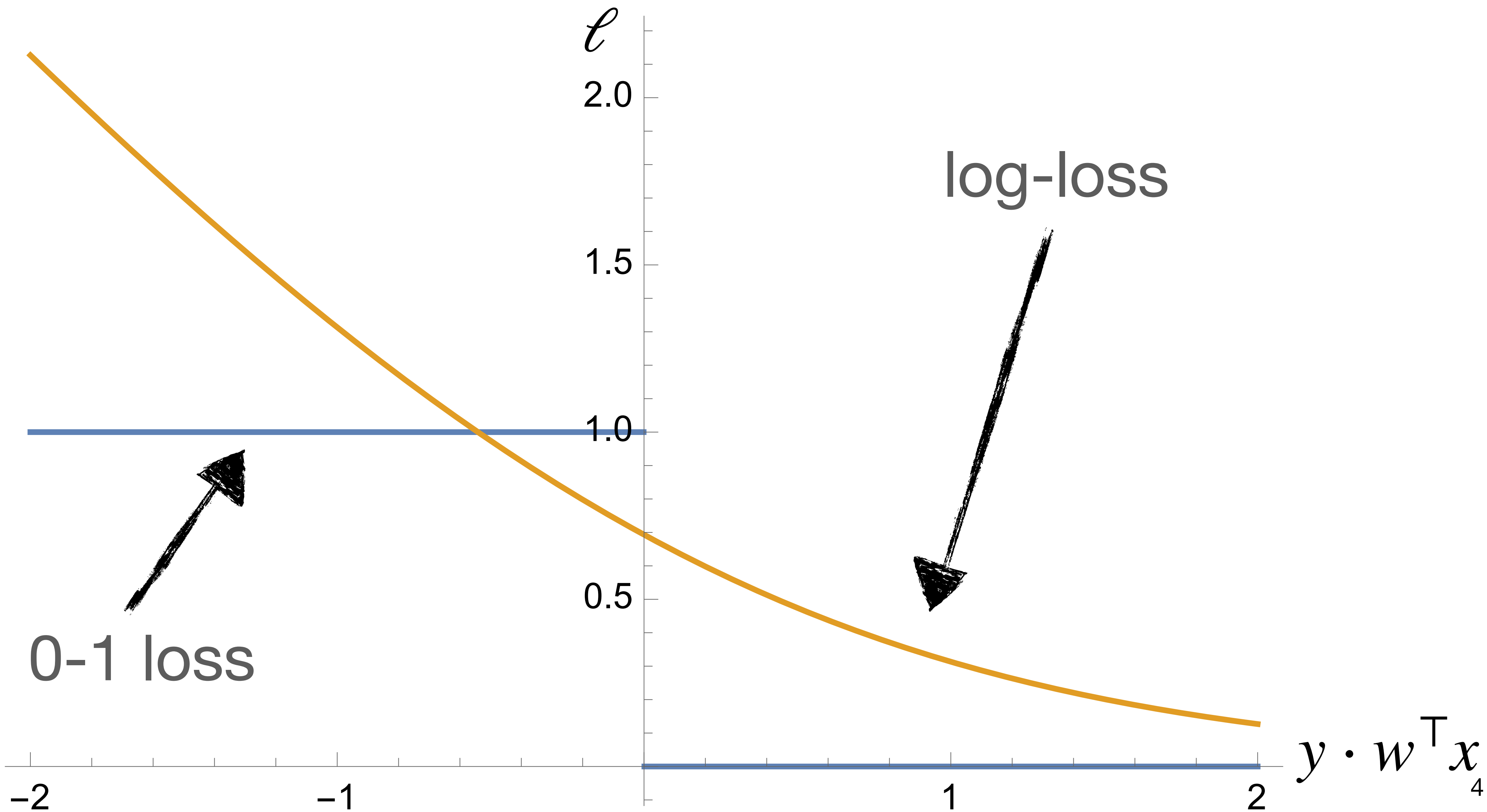
Moving forward, to simplify the notation, we will let $w' = (w, b)$ and $x' = (x,1)$ so that $w^\top x + b = w'^\top x'$. When the context is clear, we will omit the ‘ in w' and x'.

“On the difficulty of approximately maximizing agreements”, Ben-David et al., JCSS, 03

Recap: Logistic Regression

Key idea: convex relaxation

- A neat trick: transform $y \in \{0,1\} \Rightarrow y \in \{-1,1\}$
- For a linear classifier, making a wrong prediction $\iff y \cdot w^\top x \leq 0$
- Log-loss: $\ell_{\log}(y, x; w) = \log\left(1 + \exp(-y \cdot w^\top x)\right)$

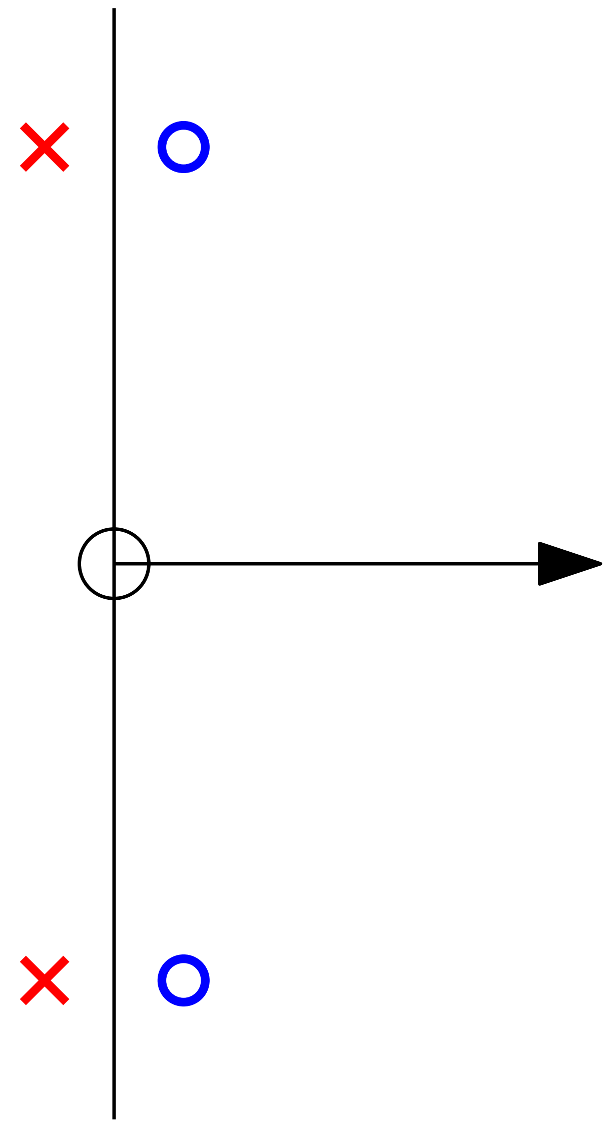


Lecture Today

- Max-margin Linear Classifier
- Support Vector Machine (primal)

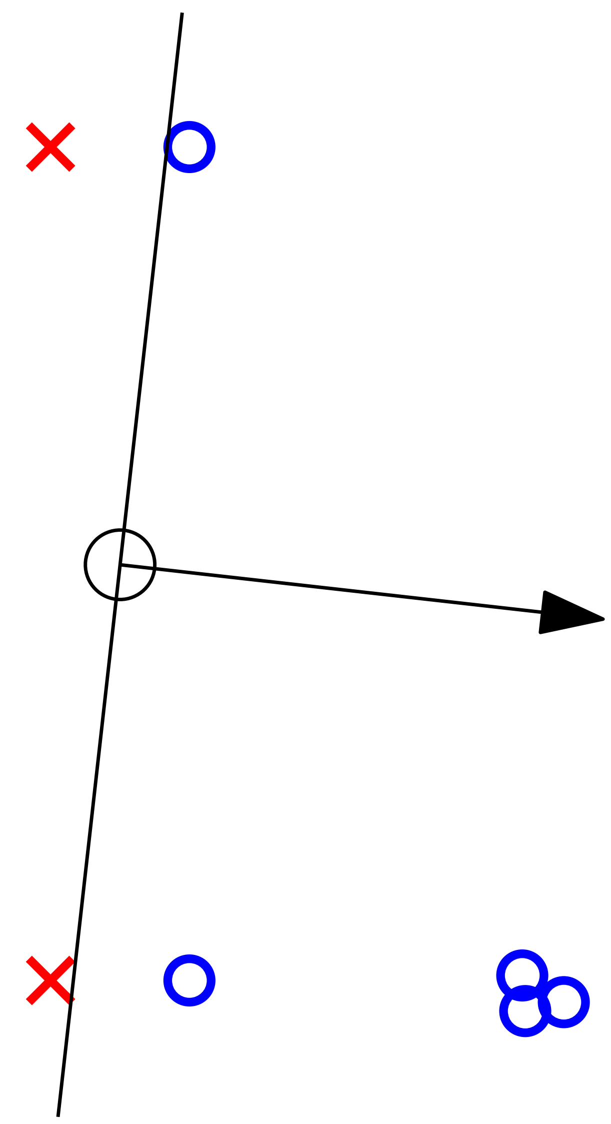
Max-Margin Classifier

Think: given a linearly separable data for binary classification, how to choose a linear classifier?



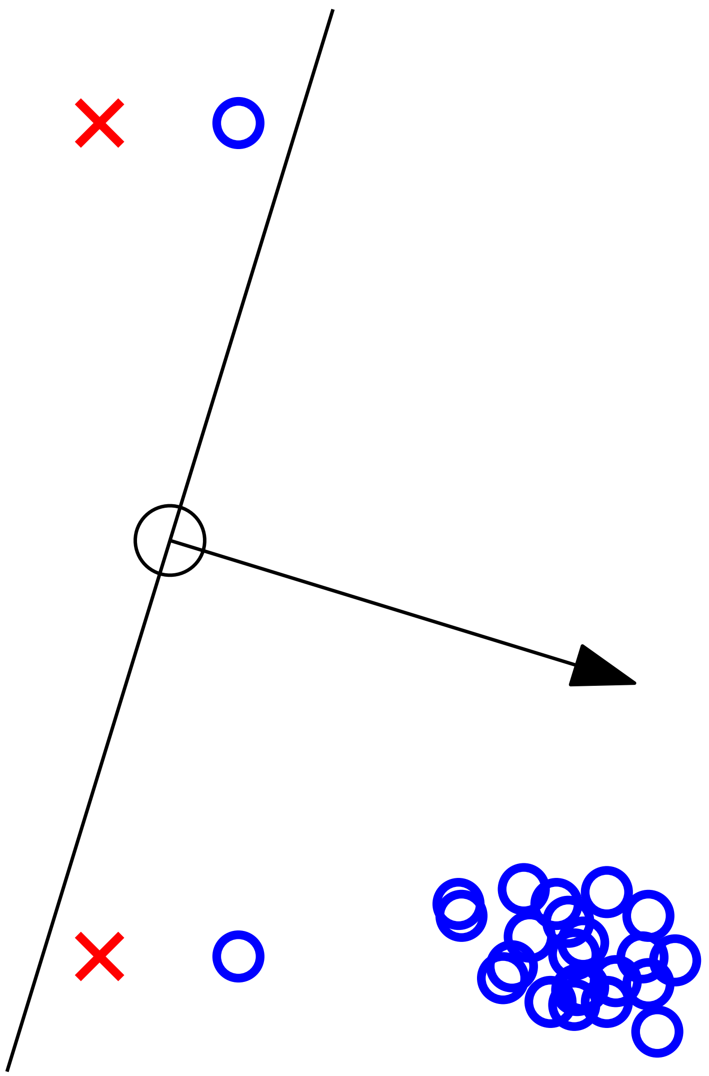
Max-Margin Classifier

Think: given a linearly separable data for binary classification, how to choose a linear classifier?



Max-Margin Classifier

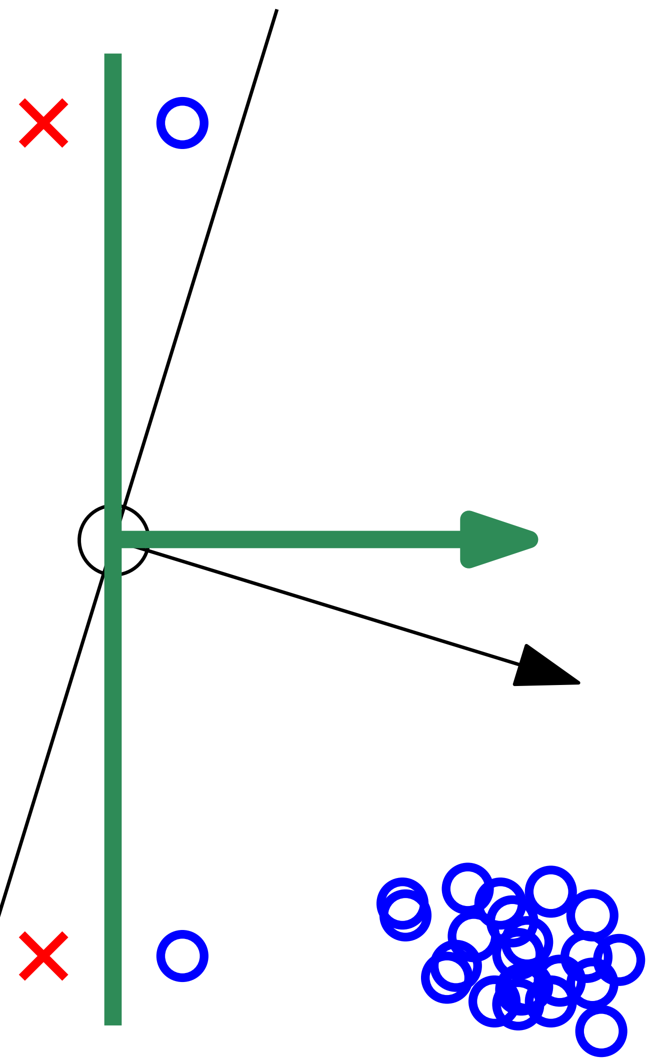
Think: given a linearly separable data for binary classification, how to choose a linear classifier?



How to pick $w \in \mathbb{R}^d$ so that all the predictions are correct?

Max-Margin Classifier

Think: given a linearly separable data for binary classification, how to choose a linear classifier?

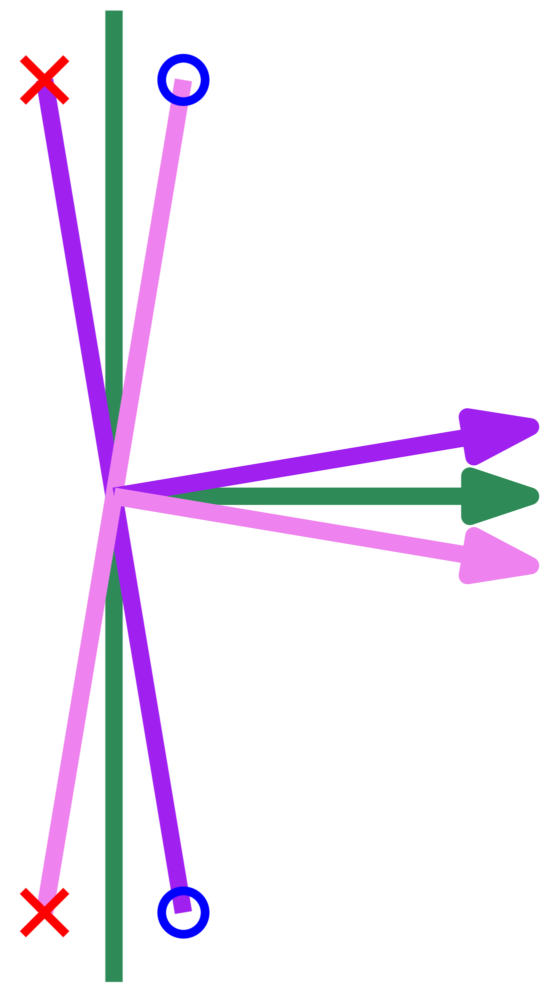


Find $w \in \mathbb{R}^d$, such that $y^{(i)} \cdot w^\top x^{(i)} > 0, \ \forall i \in [n]$

This is a linear feasibility problem, hence solvable (when data is linearly separable)

Max-Margin Classifier

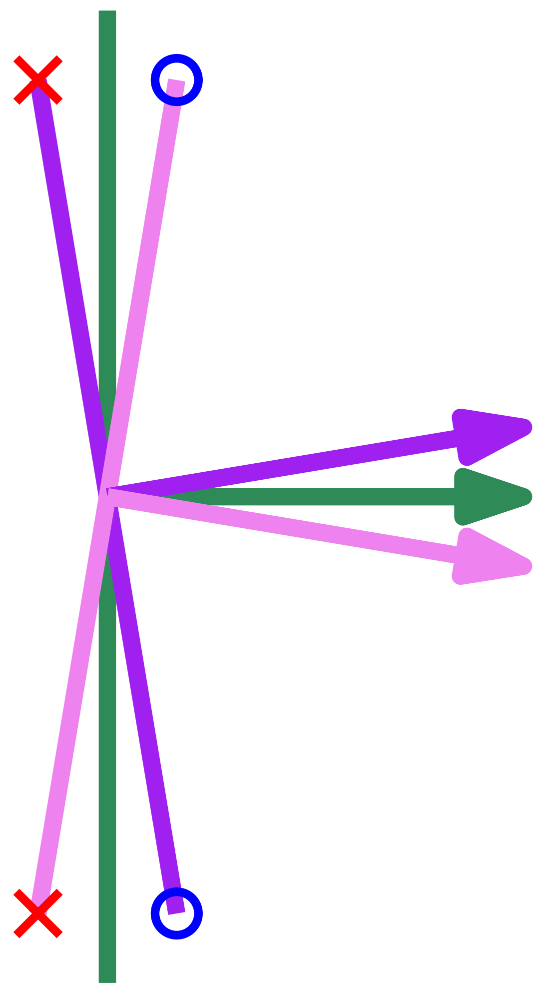
Think: if the data is linearly separable, which (correct) classifier should I choose?



Any classifier between the pink and the purple works

Max-Margin Classifier

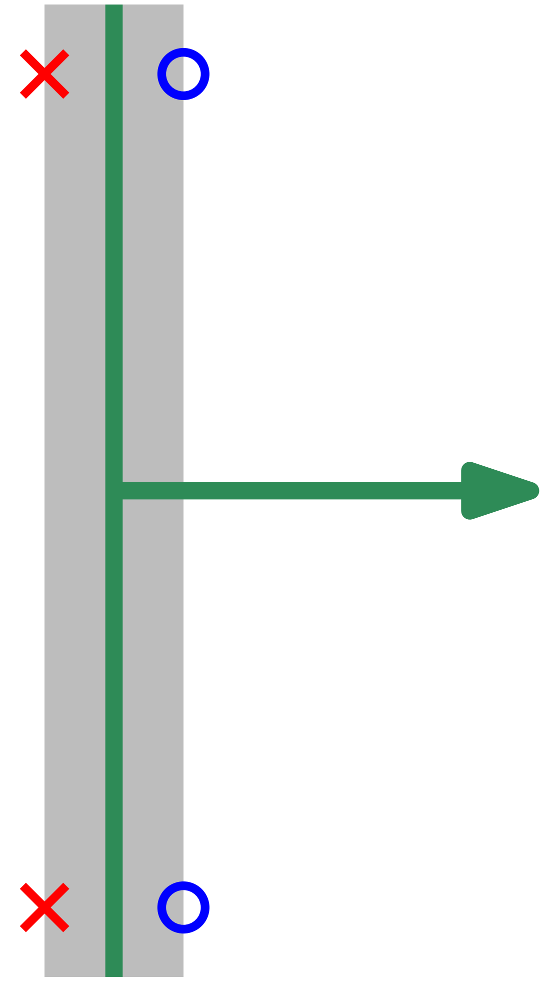
Think: if the data is linearly separable, which (correct) classifier should I choose?



Max-margin principle: (Vapink’ 82): choose w that maximizes the margin (distance to the closest data point)

Max-Margin Classifier

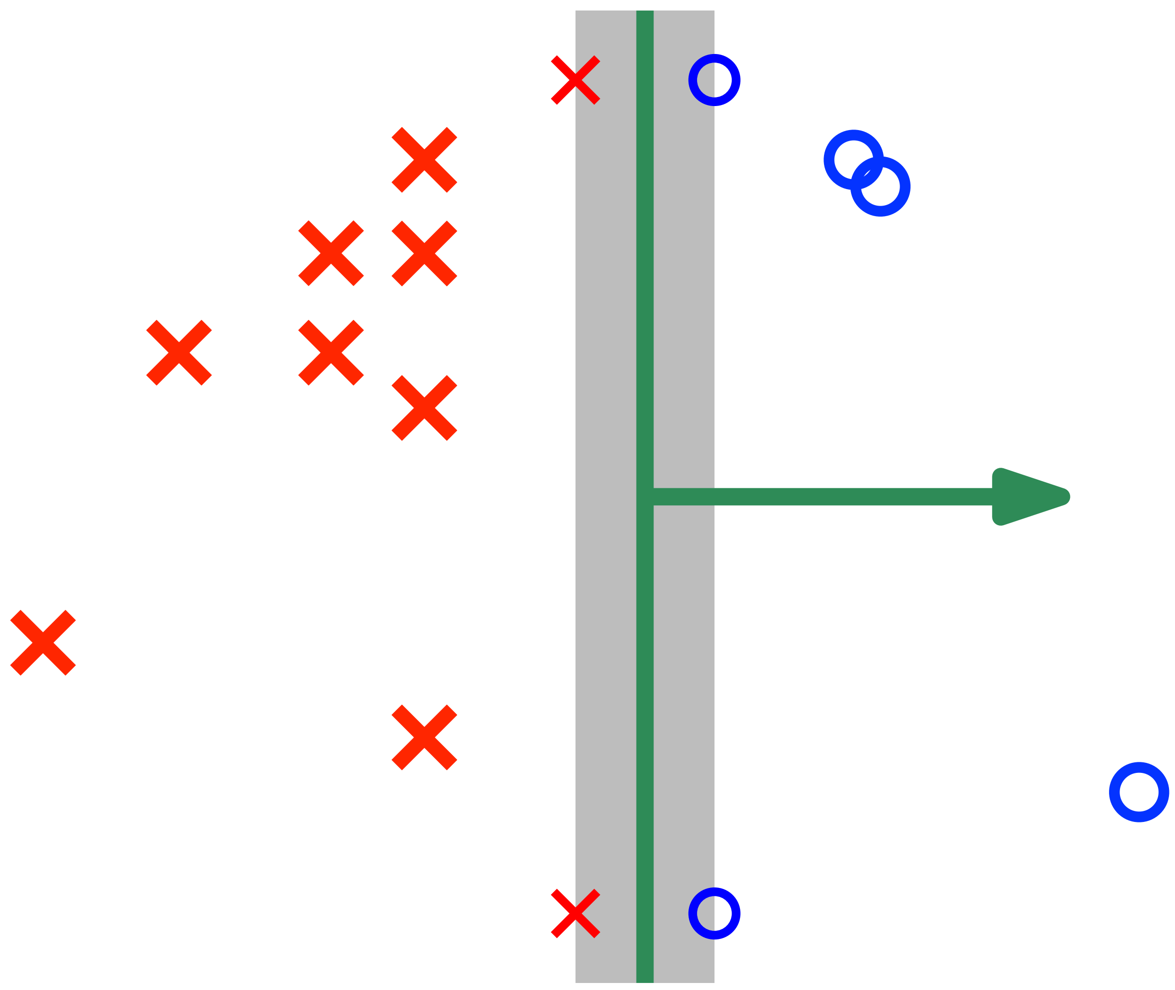
Think: if the data is linearly separable, which (correct) classifier should I choose?



Max-margin principle: (Vapink' 82): choose w that maximizes the margin (distance to the closest data point)

Max-Margin Classifier

Think: if the data is linearly separable, which (correct) classifier should I choose?



Max-margin principle: (Vapink' 82): choose w that maximizes the margin (distance to the closest data point)

Max-Margin Classifier

Maximize margin: distance to the closest data point

Given w, (signed) distance to the closest example is

$$\min_{i \in [n]} \quad \frac{y^{(i)} \cdot w^\top x^{(i)}}{\|w\|_2}$$

Hence the max-margin classifier is given by

$$\max_{w \in \mathbb{R}^d} \min_{i \in [n]} \quad \frac{y^{(i)} \cdot w^\top x^{(i)}}{\|w\|_2}$$

Note that the objective function is scale-invariant of w, hence wlog, it's equivalent to solve the following optimization problem:

$$\max_{w \in \mathbb{R}^d} \frac{1}{\|w\|_2}, \quad \text{s.t.} \quad y^{(i)} \cdot w^\top x^{(i)} \geq 1, \forall i \in [n]$$

Max-Margin Classifier

Maximize margin: distance to the closest data point

Further simplification:

$$\max_{w \in \mathbb{R}^d} \frac{1}{\|w\|_2}, \quad \text{s.t.} \quad y^{(i)} \cdot w^\top x^{(i)} \geq 1, \forall i \in [n]$$

$$\min_{w \in \mathbb{R}^d} \frac{1}{2}\|w\|_2^2, \quad \text{s.t.} \quad y^{(i)} \cdot w^\top x^{(i)} \geq 1, \forall i \in [n]$$

Note: This optimization problem

- is (strictly) convex
- if a solution exists, then it is unique
- since we assume the data to be linearly separable, then a solution exists

Max-Margin Classifier

Maximize margin: distance to the closest data point

The optimization problem of linearly separable SVM:

$$\min_{w \in \mathbb{R}^d} \frac{1}{2}\|w\|_2^2, \quad \text{s.t.} \quad y^{(i)} \cdot w^\top x^{(i)} \geq 1, \forall i \in [n]$$

How to solve it?

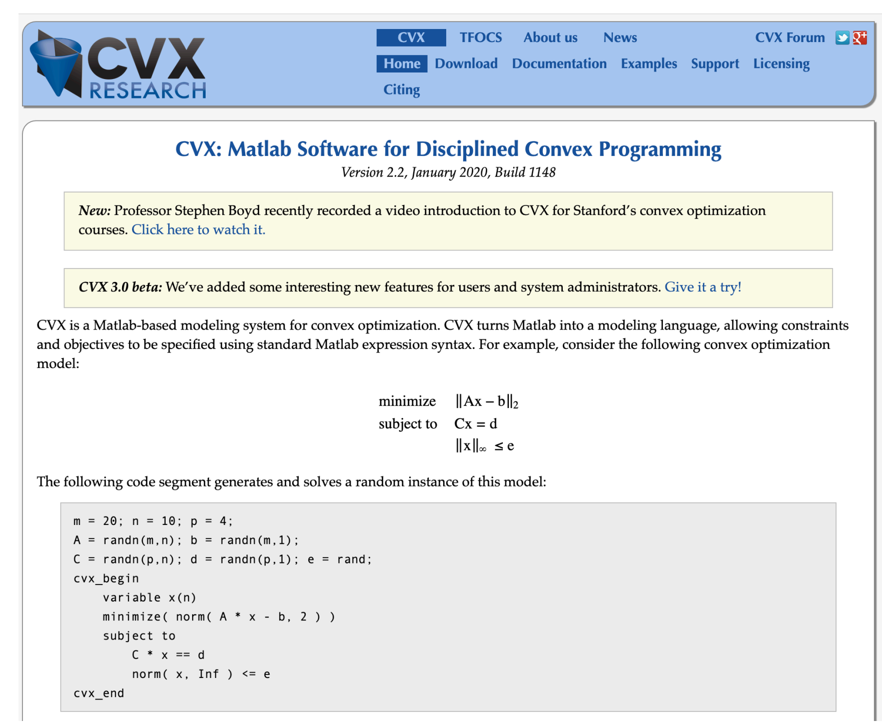
- This is an instance of the so-called "Quadratic Program"
- One can apply Gradient Descent (GD) to solve it
- Off-the-shelf general purpose convex solvers: CVX

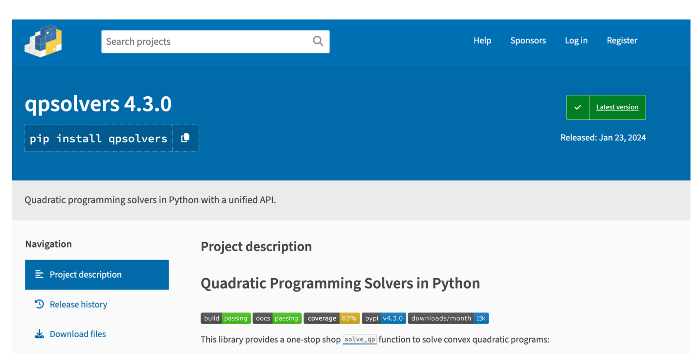
CVXpy

$$x^\star(\theta) = \operatorname*{argmin}_x f(x;\theta) \quad \text{subject to } g(x;\theta) \leq 0, \; h(x;\theta) = 0$$

→ PyTorch

→ TensorFlow





Support Vector Machine

What if my data is not linearly separable?

The following optimization problem will be infeasible:

$$\min_{w \in \mathbb{R}^d} \frac{1}{2}\|w\|_2^2, \quad \text{s.t.} \quad y^{(i)} \cdot w^\top x^{(i)} \geq 1, \forall i \in [n]$$

Key idea: introduce slack variables $\xi_i \geq 0$ for each data point $(x^{(i)}, y^{(i)})$

$$\min_{w \in \mathbb{R}^d, \xi \in \mathbb{R}^n_{\geq 0}} \frac{1}{2}\|w\|_2^2 + C \sum_{i \in [n]} \xi_i, \quad \text{s.t.} \quad y^{(i)} \cdot w^\top x^{(i)} \geq 1 - \xi_i, \forall i \in [n]$$

Geometric interpretation of $\sum_{i \in [n]} \xi_i$: the minimum amount of translation needed to make the optimization problem feasible

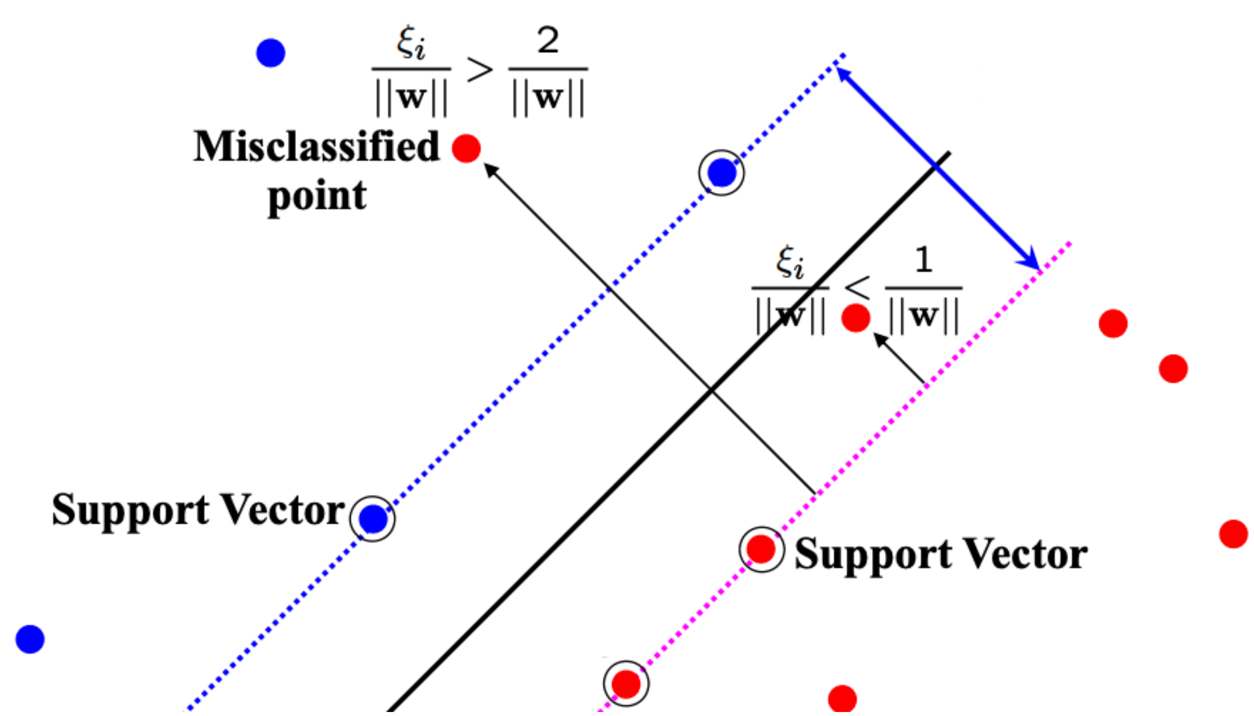
Support Vector Machine

Key idea: introduce slack variables $\xi_i \geq 0$ for each data point $(x^{(i)}, y^{(i)})$

$$\min_{w \in \mathbb{R}^d, \xi \in \mathbb{R}^n_{\geq 0}} \frac{1}{2}\|w\|_2^2 + C \sum_{i \in [n]} \xi_i, \quad \text{s.t.} \quad y^{(i)} \cdot w^\top x^{(i)} \geq 1 - \xi_i, \forall i \in [n]$$

At the optimal solution, we can tell the location of $(x^{(i)}, y^{(i)})$ from ξ_i:

- If $\xi_i = 0$, $(x^{(i)}, y^{(i)})$ is correctly classified (beyond margin)
- If $0 < \xi_i \leq 1$, $(x^{(i)}, y^{(i)})$ is correctly classified (within margin)
- If $\xi_i > 1$, $(x^{(i)}, y^{(i)})$ is wrongly classified



Support Vector Machine

Key idea: introduce slack variables $\xi_i \geq 0$ for each data point $(x^{(i)}, y^{(i)})$

$$\min_{w \in \mathbb{R}^d, \xi \in \mathbb{R}^n_{\geq 0}} \frac{1}{2}\|w\|_2^2 + C \sum_{i \in [n]} \xi_i, \quad \text{s.t.} \quad y^{(i)} \cdot w^\top x^{(i)} \geq 1 - \xi_i, \ \forall i \in [n]$$

This formulation is also known as the soft-margin SVM

Regularized form:

$$\min_{w \in \mathbb{R}^d, \xi \in \mathbb{R}^n_{\geq 0}} \sum_{i \in [n]} \xi_i + \frac{\lambda}{2}\|w\|_2^2, \quad \text{s.t.} \quad y^{(i)} \cdot w^\top x^{(i)} \geq 1 - \xi_i, \ \forall i \in [n]$$

Further transformation into an unconstrained optimization problem:

$$\min_{w \in \mathbb{R}^d} \sum_{i \in [n]} \ell_{\text{hinge}}(y^{(i)} \cdot w^\top x^{(i)}) + \frac{\lambda}{2}\|w\|_2^2,$$

where $\ell_{\text{hinge}}(t) := \max\{0, 1 - t\}$ is called the hinge-loss

The last formulation is what most people call Support Vector Machine (SVM)

Support Vector Machine

Hinge-loss formulation:

$$\min_{w \in \mathbb{R}^d} \sum_{i \in [n]} \ell_{\text{hinge}}(y^{(i)} \cdot w^\top x^{(i)}) + \frac{\lambda}{2}\|w\|_2^2,$$

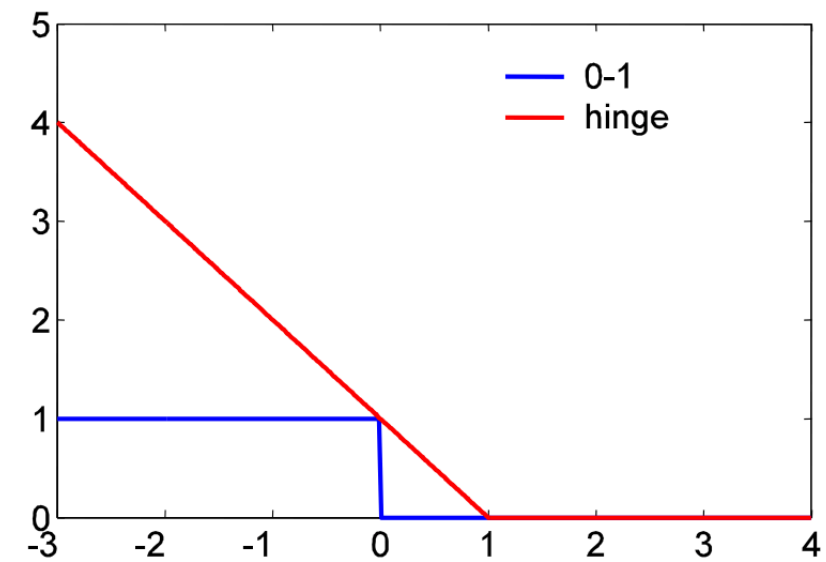
where $\ell_{\text{hinge}}(t) := \max\{0, 1 - t\}$ is called the hinge-loss

Hinge loss

l_2 regularization of w

The hyper-parameter $\lambda \geq 0$ controls the strength of regularization:

- If $\lambda \to 0$, then less focus on regularization, more focus on loss
- If $\lambda \to \infty$, then more focus on regularization, less focus on regularization

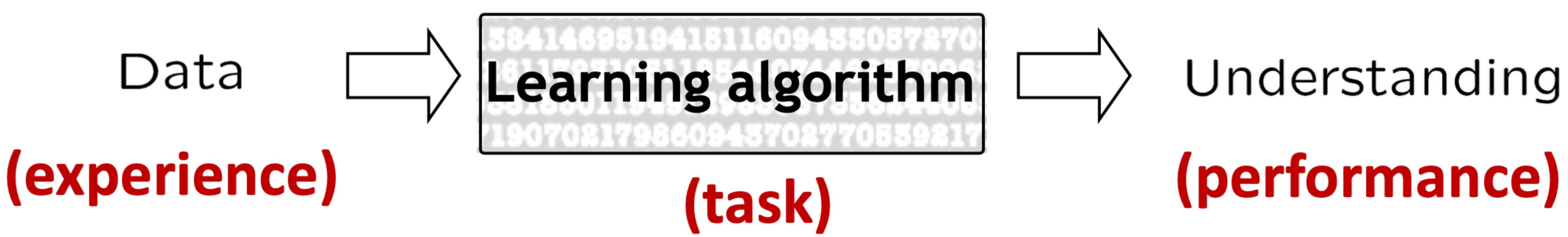


Support Vector Machine

Recall:

> "A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E."
>
> — Tom M. Mitchell

- E: data
- T: task of interest
- P: objective function



Support Vector Machine

Comparisons:

- E: supervised
- T: linear prediction
- P: zero-one, hinge, logistic, squared

Regularized linear regression (Ridge regression):

$$\min_{w \in \mathbb{R}^d} \sum_{i \in [n]} (y^{(i)} - w^\top x^{(i)})^2 + \frac{\lambda}{2} \|w\|_2^2,$$

Regularized logistic regression:

$$\min_{w \in \mathbb{R}^d} \sum_{i \in [n]} \ell_{\log}(y^{(i)} \cdot w^\top x^{(i)}) + \frac{\lambda}{2} \|w\|_2^2,$$

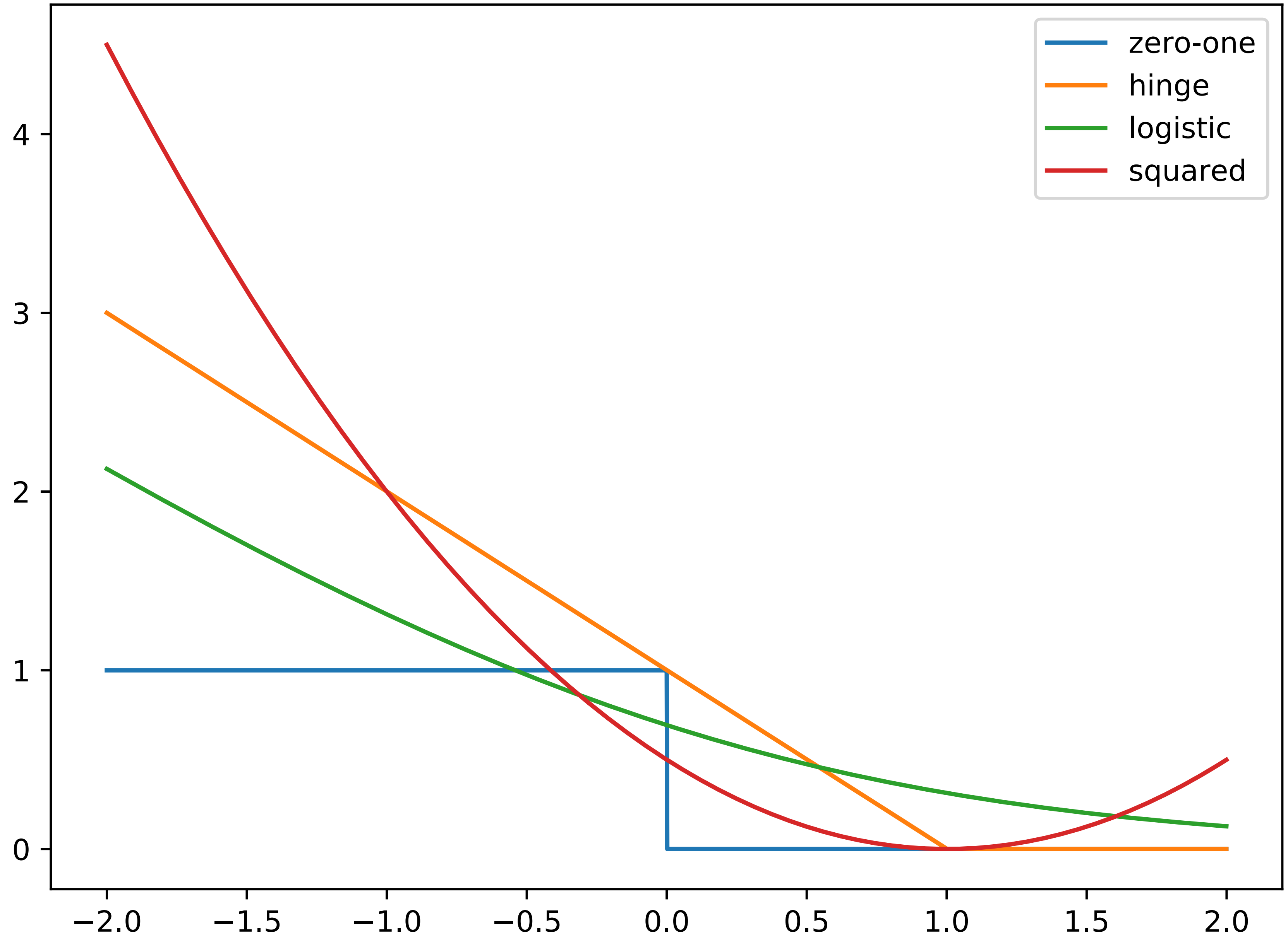
Support vector machines:

$$\min_{w \in \mathbb{R}^d} \sum_{i \in [n]} \ell_{\text{hinge}}(y^{(i)} \cdot w^\top x^{(i)}) + \frac{\lambda}{2} \|w\|_2^2,$$

Support Vector Machine

Comparisons:

- E: supervised
- T: linear prediction
- P: zero-one, hinge, logistic, squared



Next Time

- Support Vector Machine (Dual)
- Kernel Methods