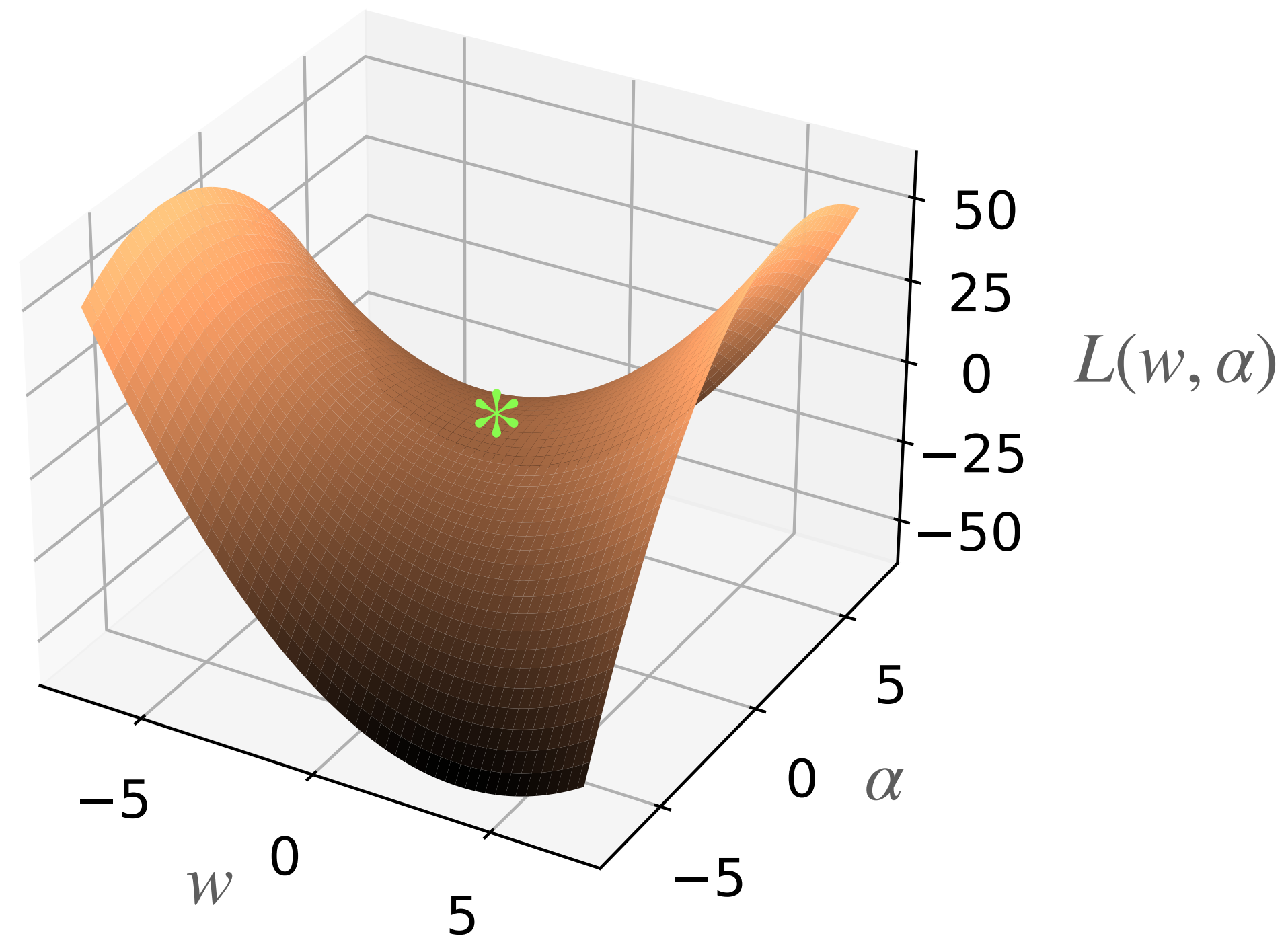


Duality

*10-701 Introduction to Machine Learning
Geoff Gordon and Pradeep Ravikumar*

Adversarial (min-max) problems



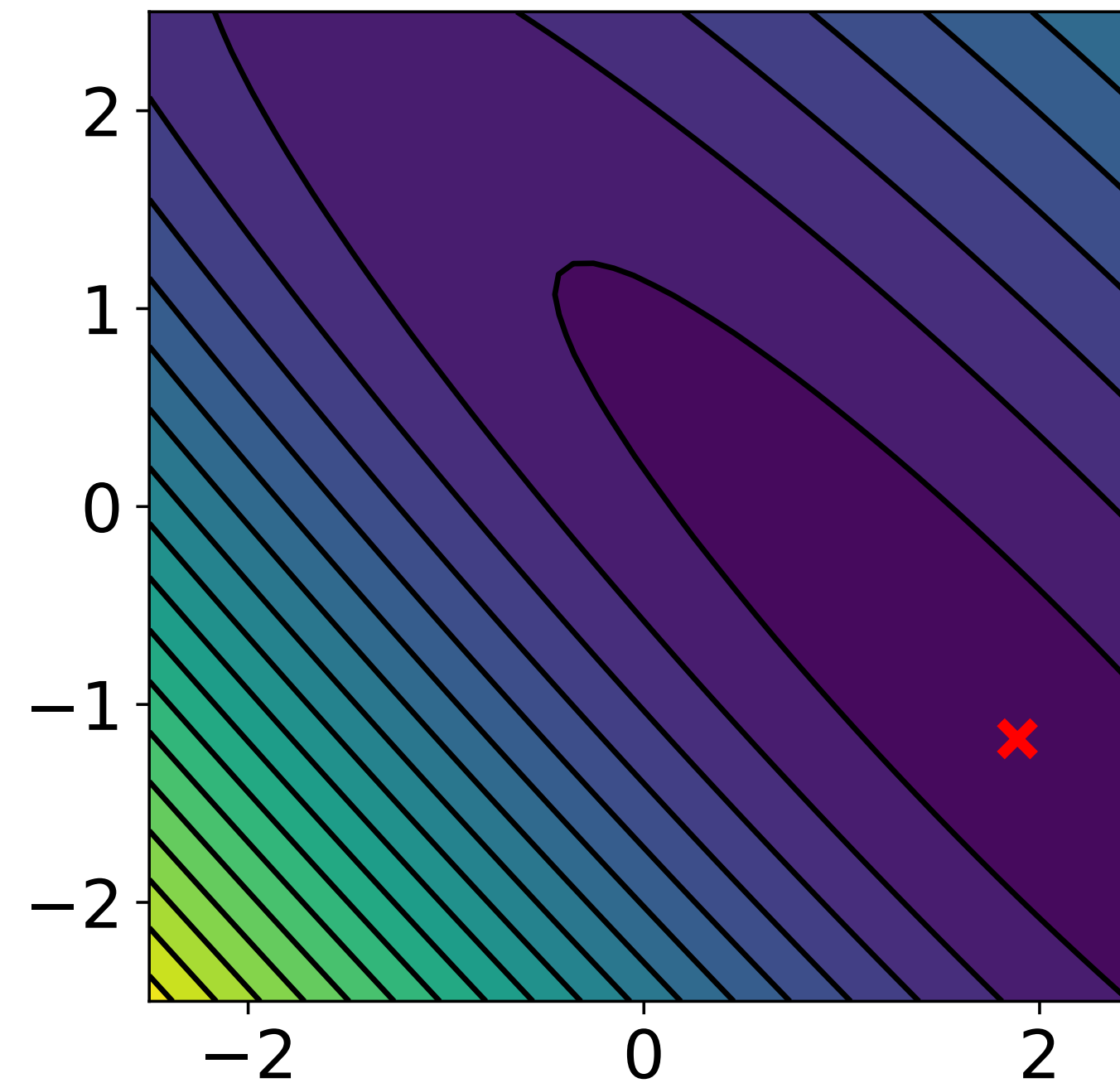
- So far we've looked at problems of form $\min_w L(w)$
- Some ML problems instead have form $\min_w \max_\alpha L(w, \alpha)$
 - ▶ often called *adversarial learning*: the w player and α player have opposing goals
 - ▶ a (local) optimum is a point where w can't (locally) decrease and α can't (locally) increase objective value
 - ▶ (locally) convex-concave, looks like a saddle: *saddle-point*

Algorithms

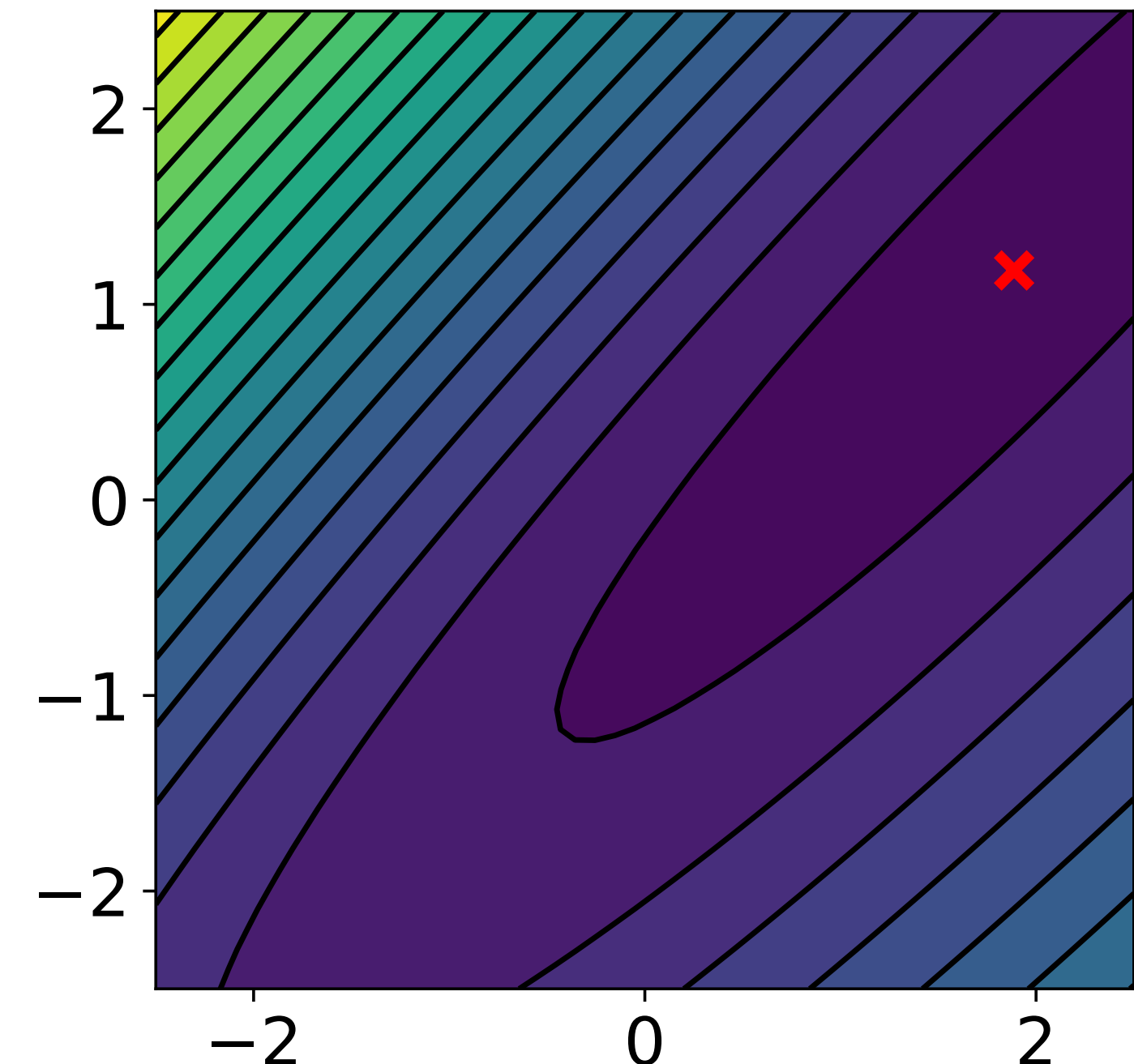
- Idea: **(stochastic) gradient descent-ascent (S)GDA**
 - ▶ min player runs (stochastic) gradient descent (maybe with momentum, Adam, etc.)
 - ▶ max player runs (stochastic) gradient ascent, interleaved
 - ▶ kind of works, no guarantees — people force it w/ hyperparameter tuning (learning rate schedules, update frequencies, ...)
- There exist methods with better guarantees: **no-regret** learning, **optimistic** (S)GDA
 - ▶ they seem not to be used that much in practice
 - ▶ hypothesis: they hurt the mystical properties of SGD while helping convergence
 - ▶ and some guarantees don't extend to practical situations
 - ▶ and some methods require inconvenient modifications like storing extra copies of our network

Example: problem splitting

Learner 1 loss function



Learner 2 loss function

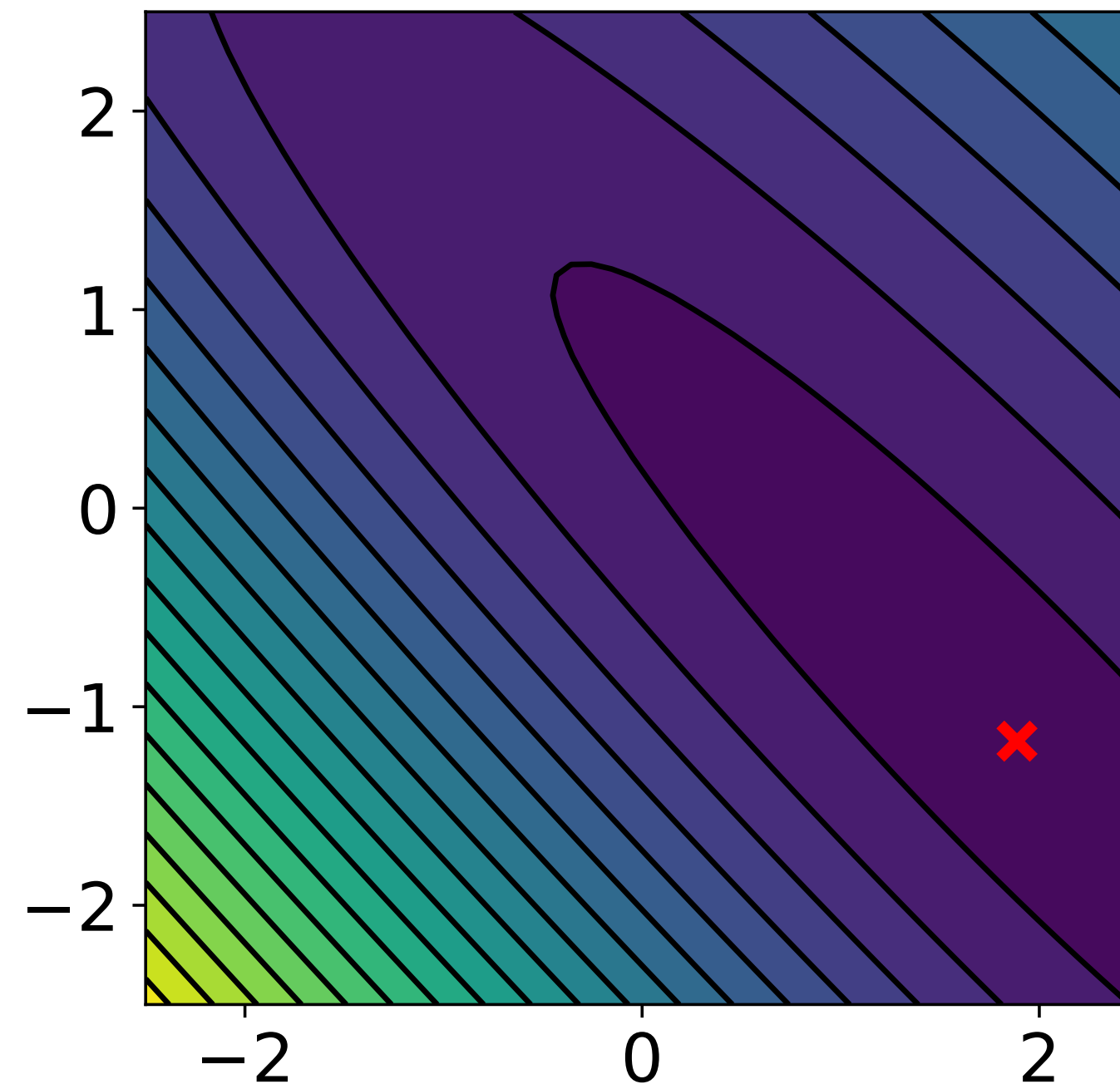


- Two learners, each with half of our dataset
 - ▶ for speed with big dataset
 - ▶ or for privacy: e.g., two hospitals unwilling to share full patient records — ***federated learning***

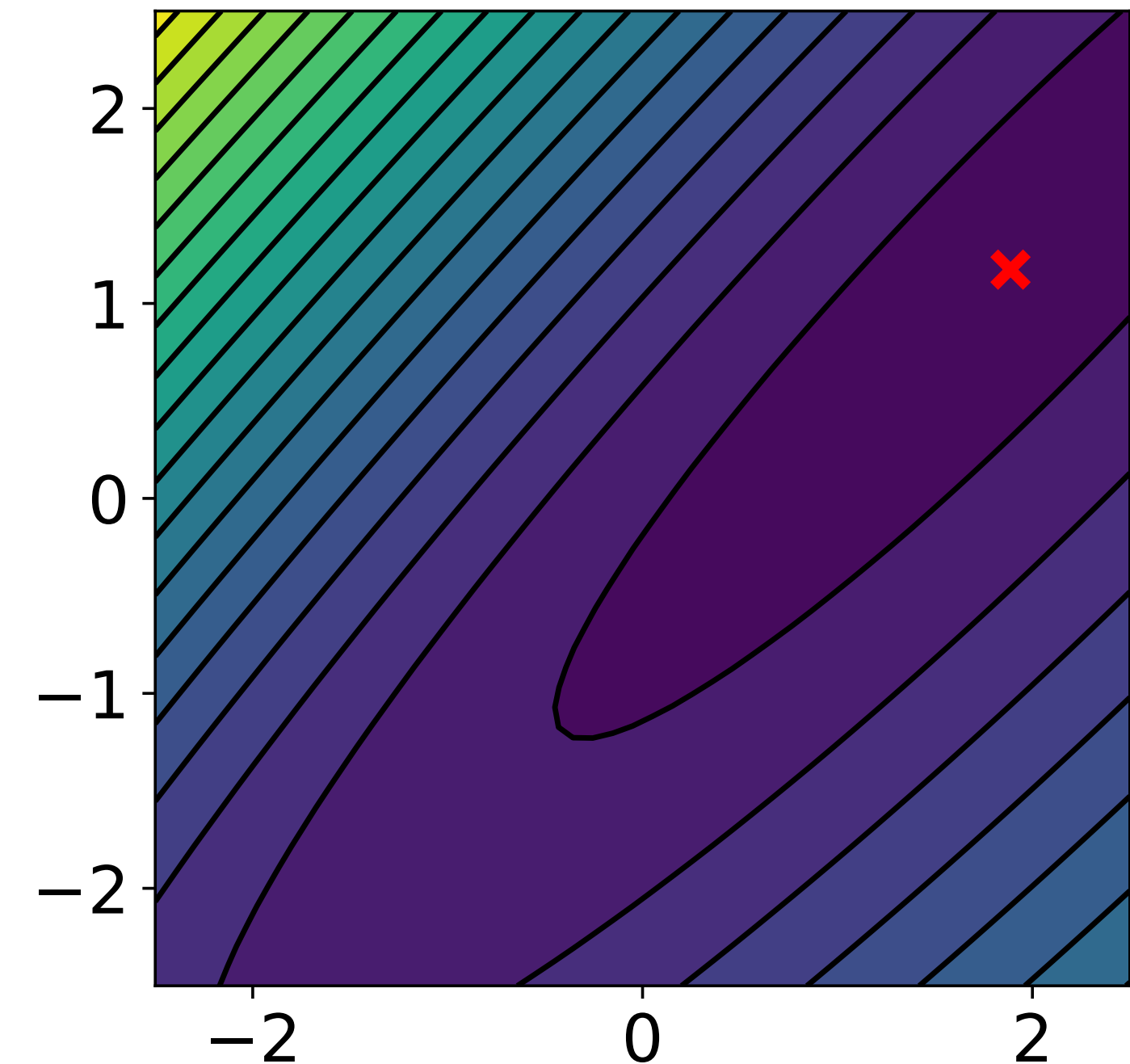
Example: splitting data

If datasets really are from different distributions, we need **personalization** — but players still hope to learn from each other's data

Learner 1 loss function



Learner 2 loss function

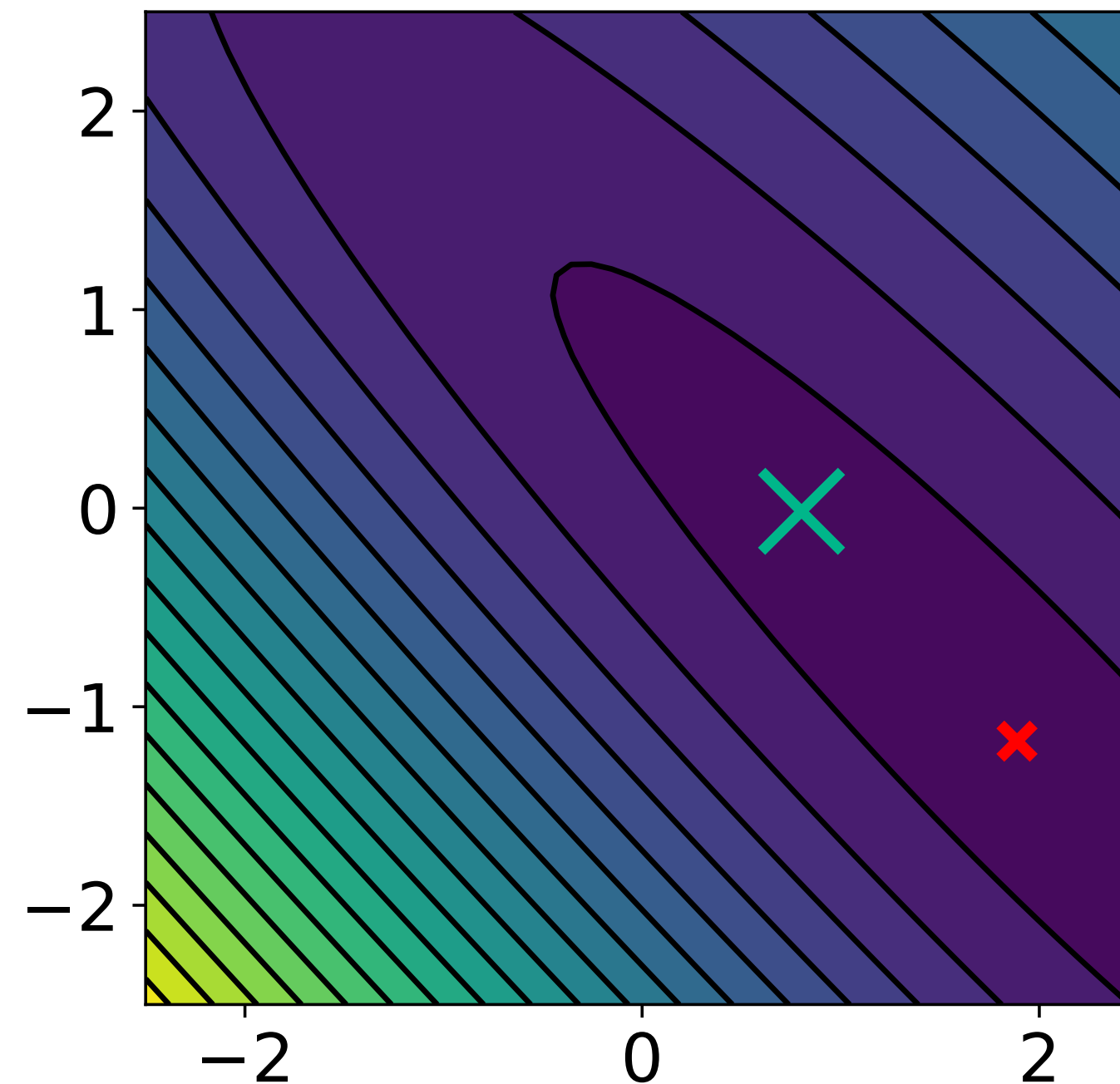


- If optimized independently: $\min_{\theta_1} L_1(\theta_1)$, $\min_{\theta_2} L_2(\theta_2)$
 - ▶ players get different and suboptimal solutions
- Best solution: $\min_{\theta_1, \theta_2} [L_1(\theta_1) + L_2(\theta_2)]$ s.t. $\theta_1 = \theta_2$
 - ▶ equivalent to centralized answer $\min_{\theta} [L_1(\theta) + L_2(\theta)]$

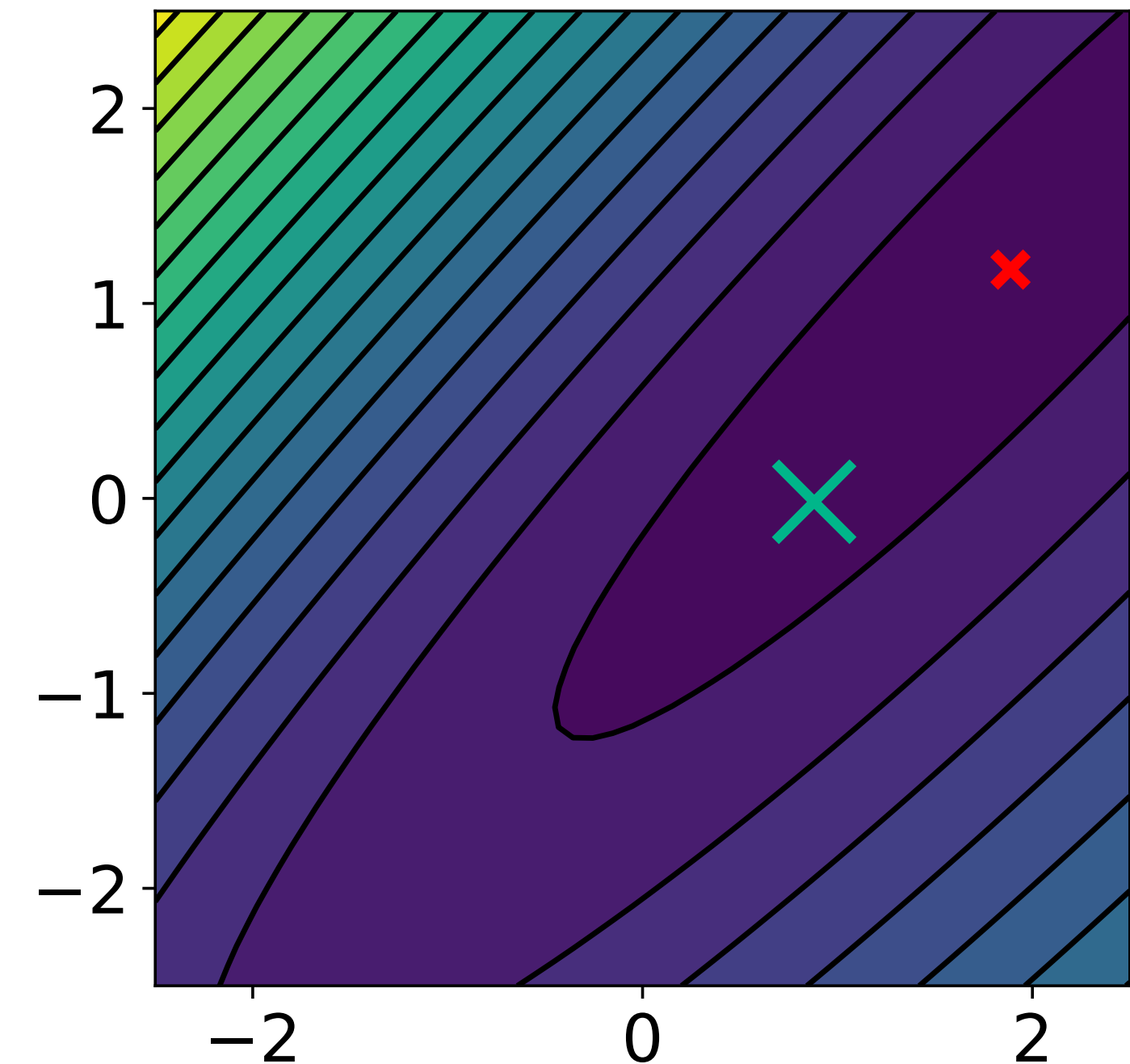
Example: splitting data

If datasets really are from different distributions, we need **personalization** — but players still hope to learn from each other's data

Learner 1 loss function

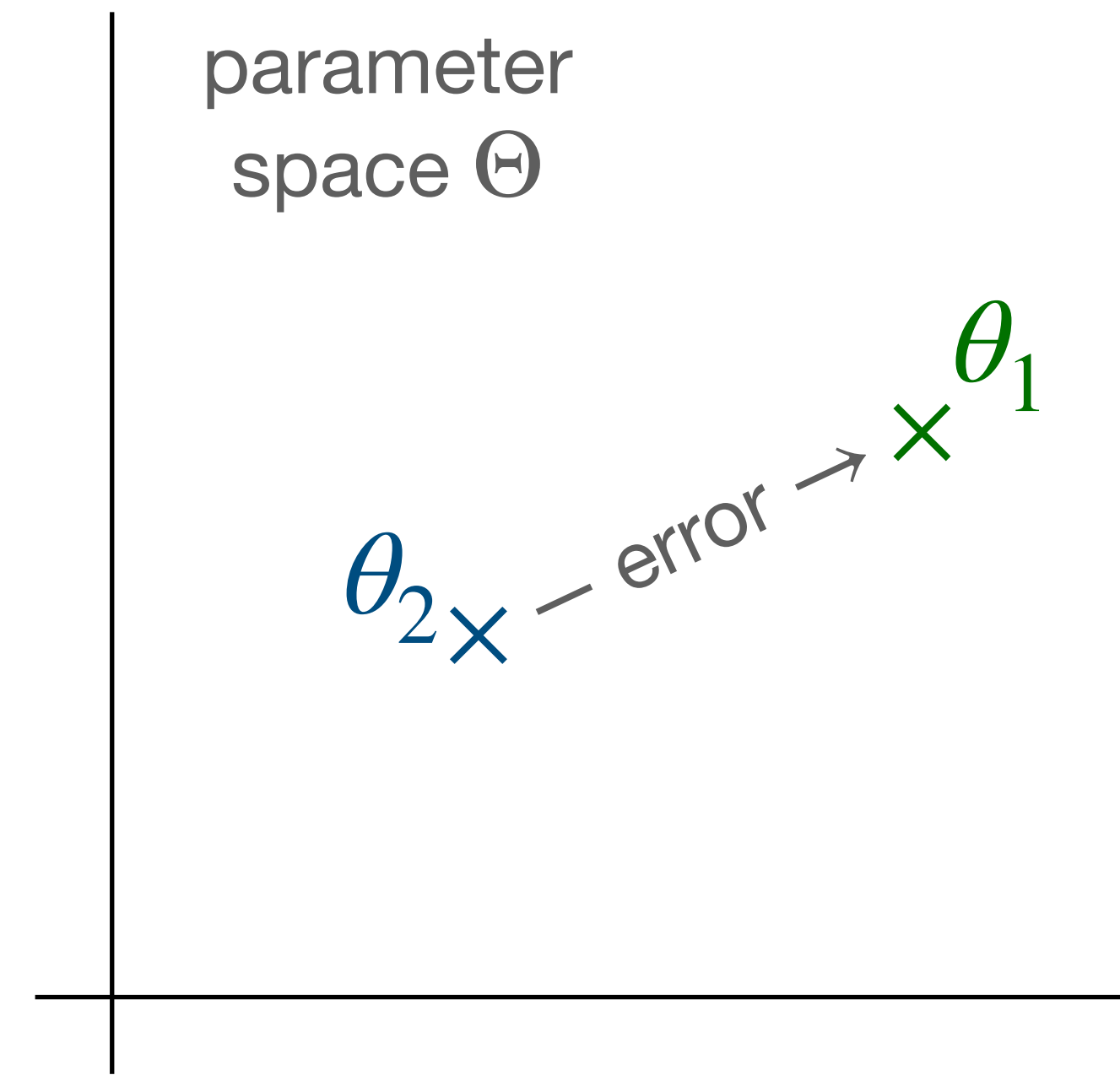


Learner 2 loss function



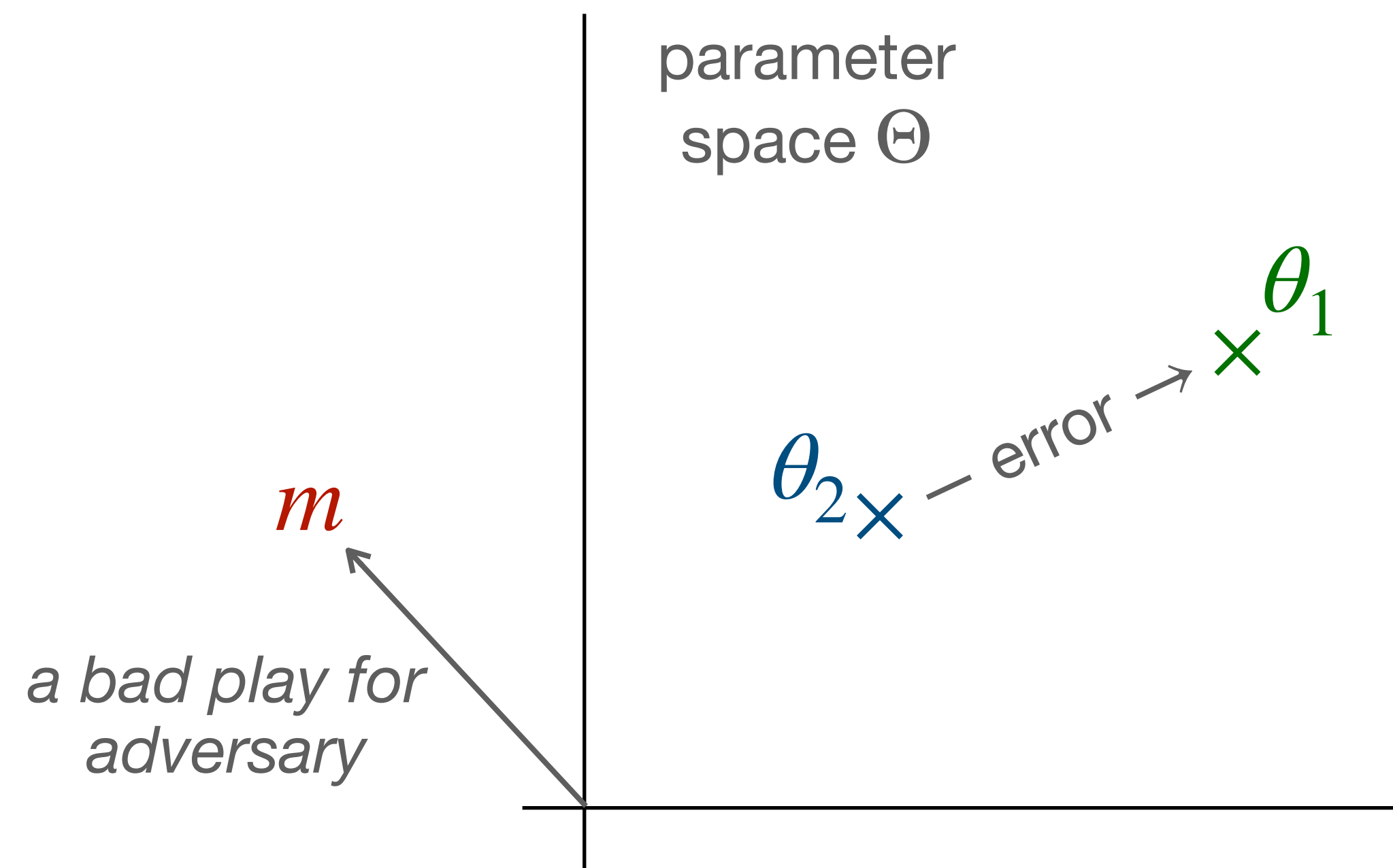
- If optimized independently: $\min_{\theta_1} L_1(\theta_1)$, $\min_{\theta_2} L_2(\theta_2)$
 - ▶ players get different and suboptimal solutions
- Best solution: $\min_{\theta_1, \theta_2} [L_1(\theta_1) + L_2(\theta_2)]$ s.t. $\theta_1 = \theta_2$
 - ▶ equivalent to centralized answer $\min_{\theta} [L_1(\theta) + L_2(\theta)]$

Lagrange multiplier



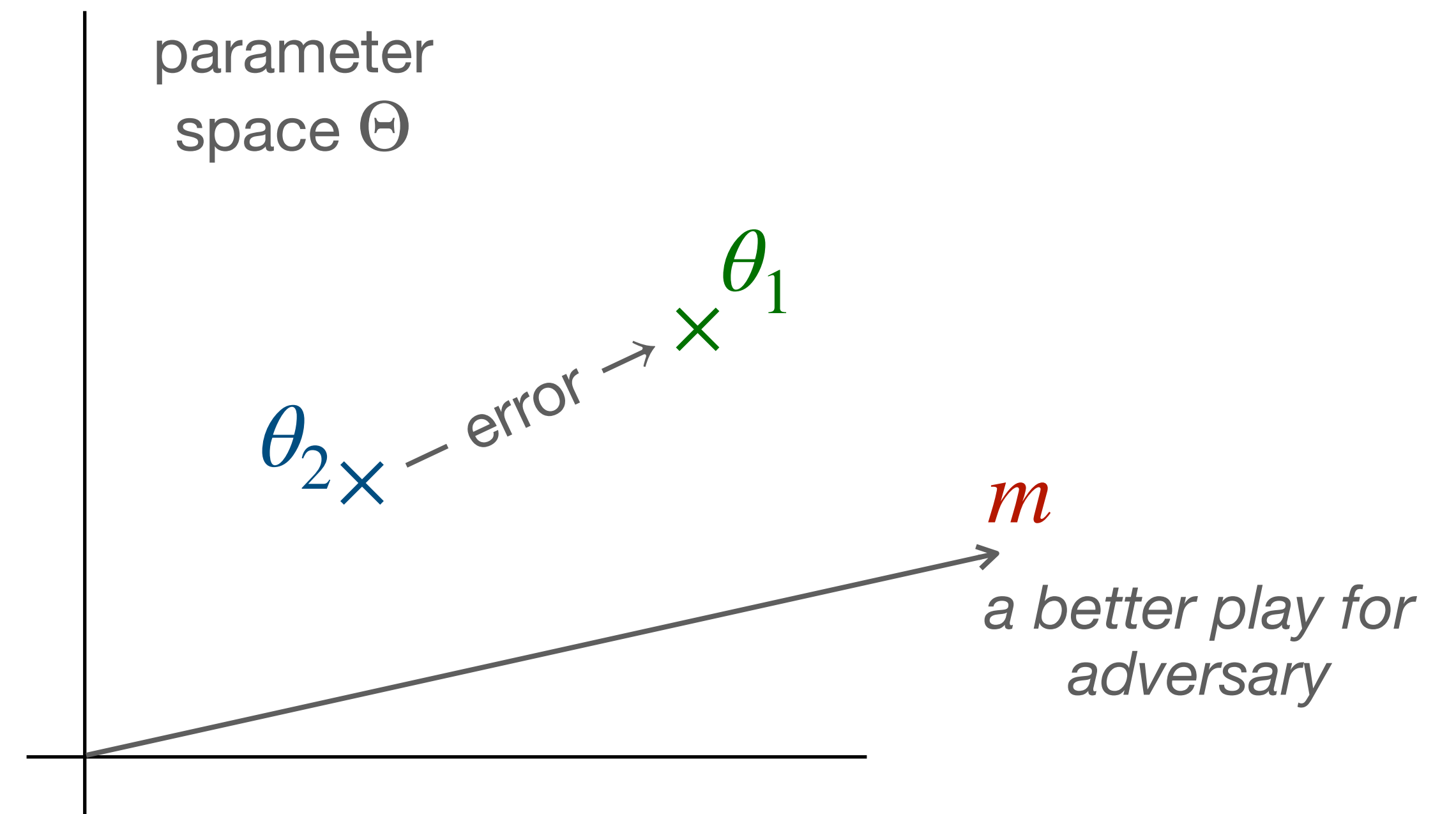
- To enforce the equality constraint, add an adversary that searches for and penalizes violations
 - ▶ $\min_{\theta_1, \theta_2} \max_m [L_1(\theta_1) + L_2(\theta_2) + m^\top (\theta_1 - \theta_2)]$
- The θ_1, θ_2 players lose badly if they choose $\theta_1 \neq \theta_2$: loss $+\infty$ if m isn't regularized

Lagrange multiplier



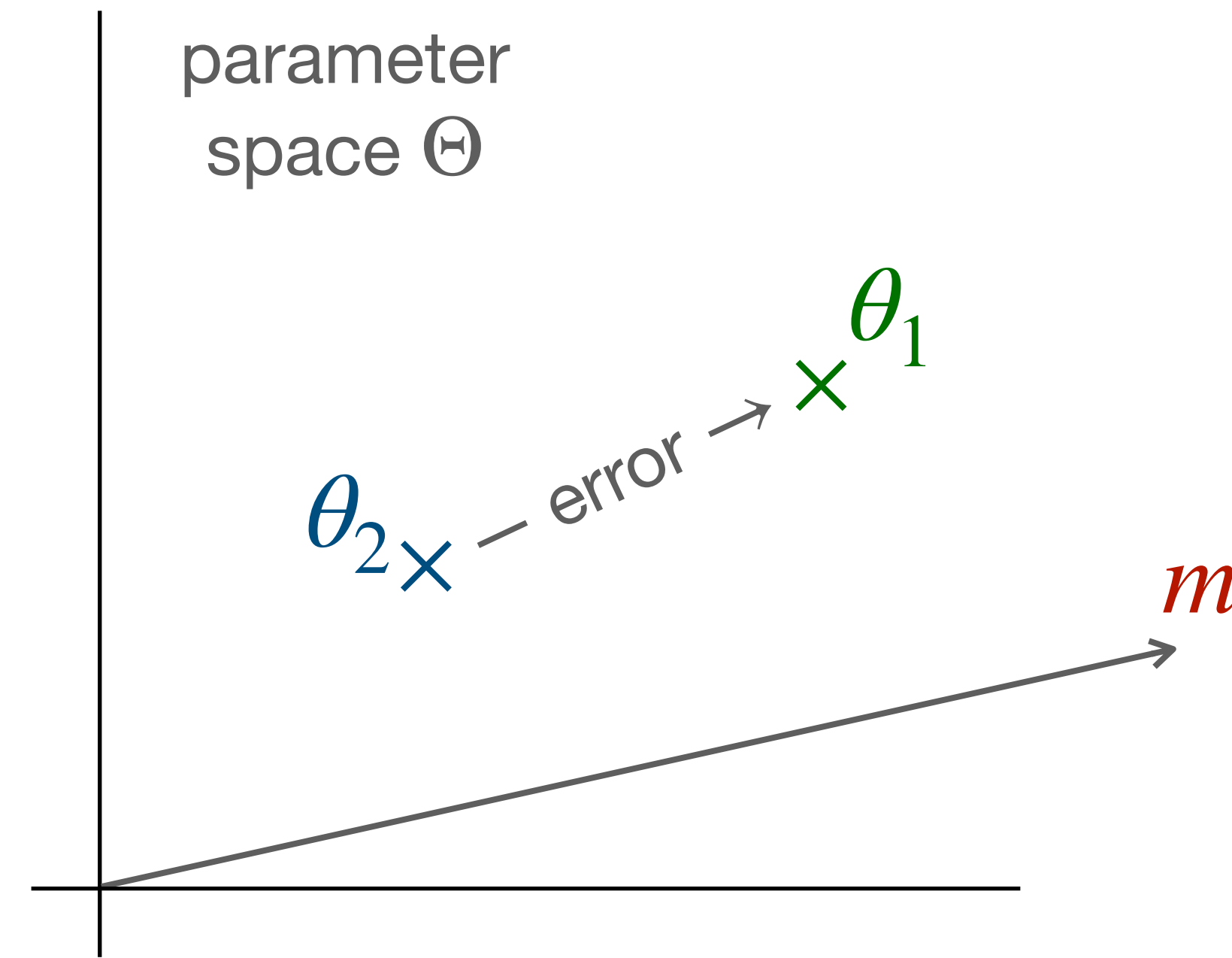
- To enforce the equality constraint, add an adversary that searches for and penalizes violations
 - ▶ $\min_{\theta_1, \theta_2} \max_m [L_1(\theta_1) + L_2(\theta_2) + m^\top (\theta_1 - \theta_2)]$
- The θ_1, θ_2 players lose badly if they choose $\theta_1 \neq \theta_2$: loss $+\infty$ if m isn't regularized

Lagrange multiplier



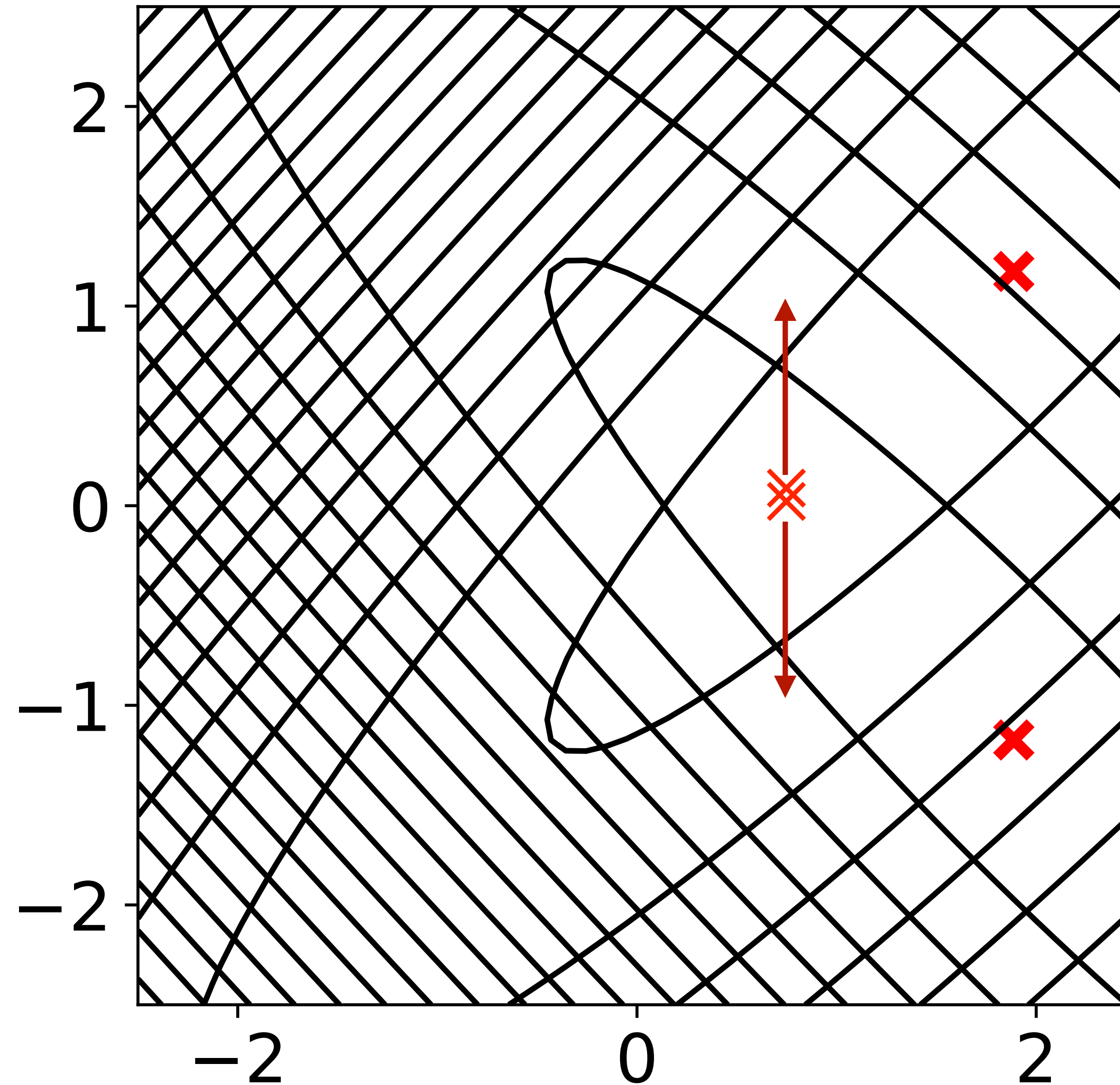
- To enforce the equality constraint, add an adversary that searches for and penalizes violations
 - ▶ $\min_{\theta_1, \theta_2} \max_m [L_1(\theta_1) + L_2(\theta_2) + m^\top (\theta_1 - \theta_2)]$
- The θ_1, θ_2 players lose badly if they choose $\theta_1 \neq \theta_2$: loss $+\infty$ if m isn't regularized

Adversary corrects constraint violations



- Now θ_1 player and θ_2 player can optimize mostly independently — for fixed m the problem splits:
 - ▶ $\min_{\theta_1} [L_1(\theta_1) + m^\top \theta_1]$ and $\min_{\theta_2} [L_2(\theta_2) - m^\top \theta_2]$
- And if we fix θ s, good plays for adversary are directions that encourage θ_1 and θ_2 to move closer together

Equilibrium



- If everyone learns at once, we hope to converge to a (local or global) optimum, called an ***equilibrium***: gradient for each player = zero, no incentive to move

Finding the equilibrium

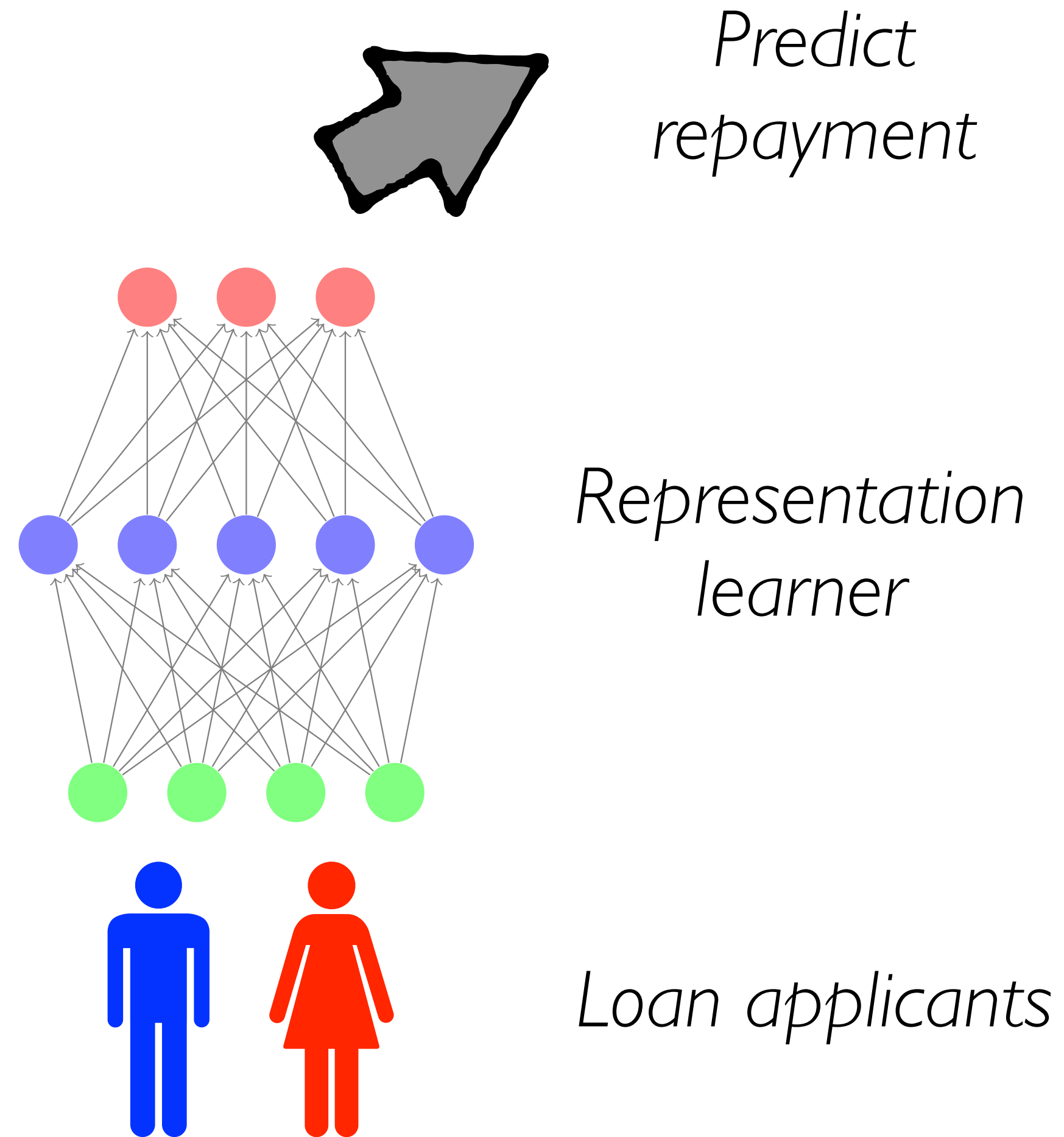
$$\min_{\theta_1, \theta_2} \max_m [L_1(\theta_1) + L_2(\theta_2) + m^\top (\theta_1 - \theta_2)]$$

- Find equilibrium by setting gradients wrt θ_1, θ_2, m to 0:

- ▶ $0 = \nabla_m = (\theta_1 - \theta_2) \Rightarrow \theta_1 = \theta_2$

- ▶ substitute back in: gets to local min over
 $\theta = \theta_1 = \theta_2$

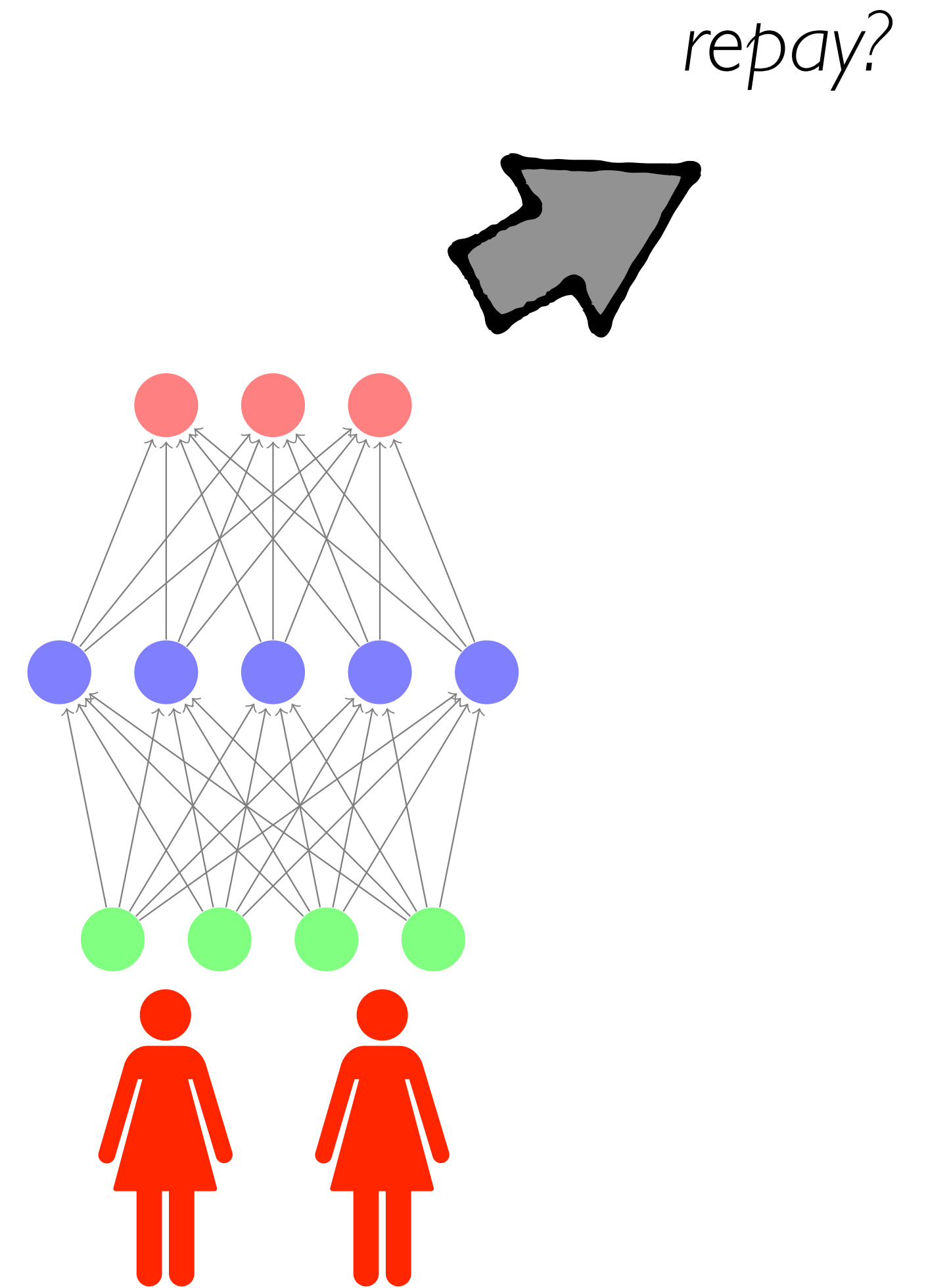
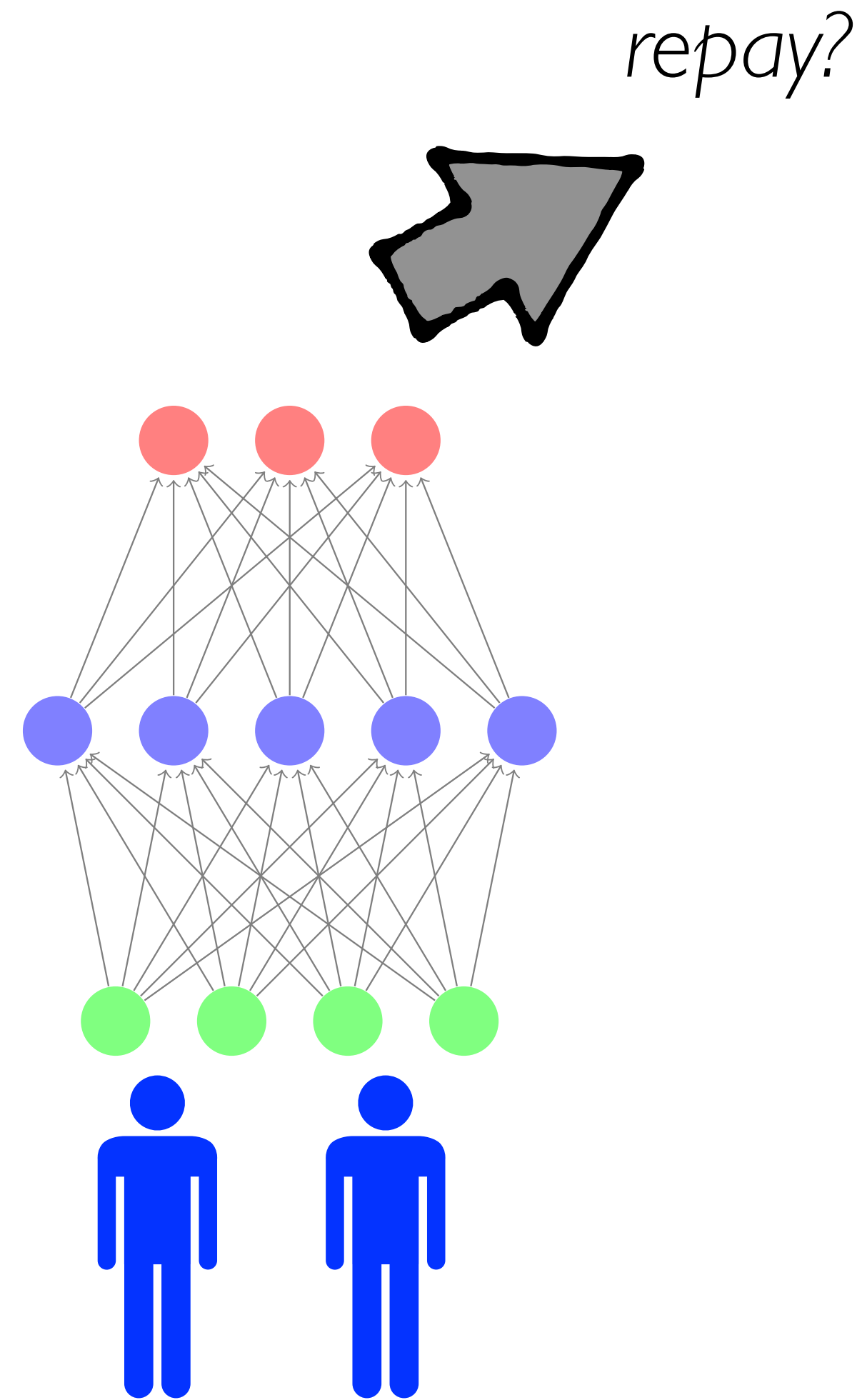
Example: fairness



- **Problem:** naive classifier can't be better than the dataset we start from!
 - ▶ Human loan officers used different criteria to approve women and men
 - ▶ Systemic biases made equally qualified candidates more or less likely to repay
- In fact, often worse: ML learns to apply bias more consistently and effectively
- We can do *better at prediction* by doing *worse at minimizing holdout error*

Enforce constraints to improve fairness

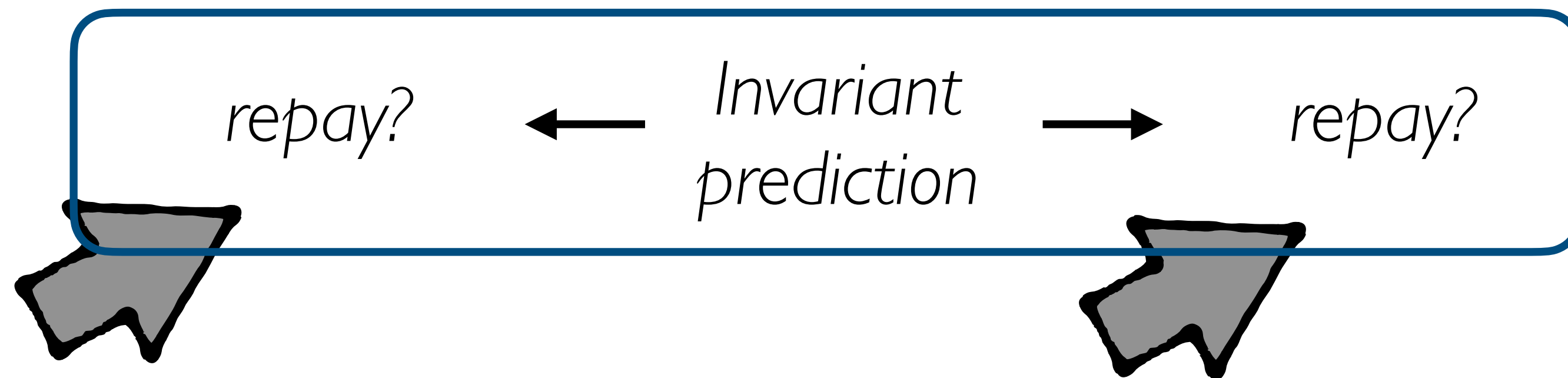
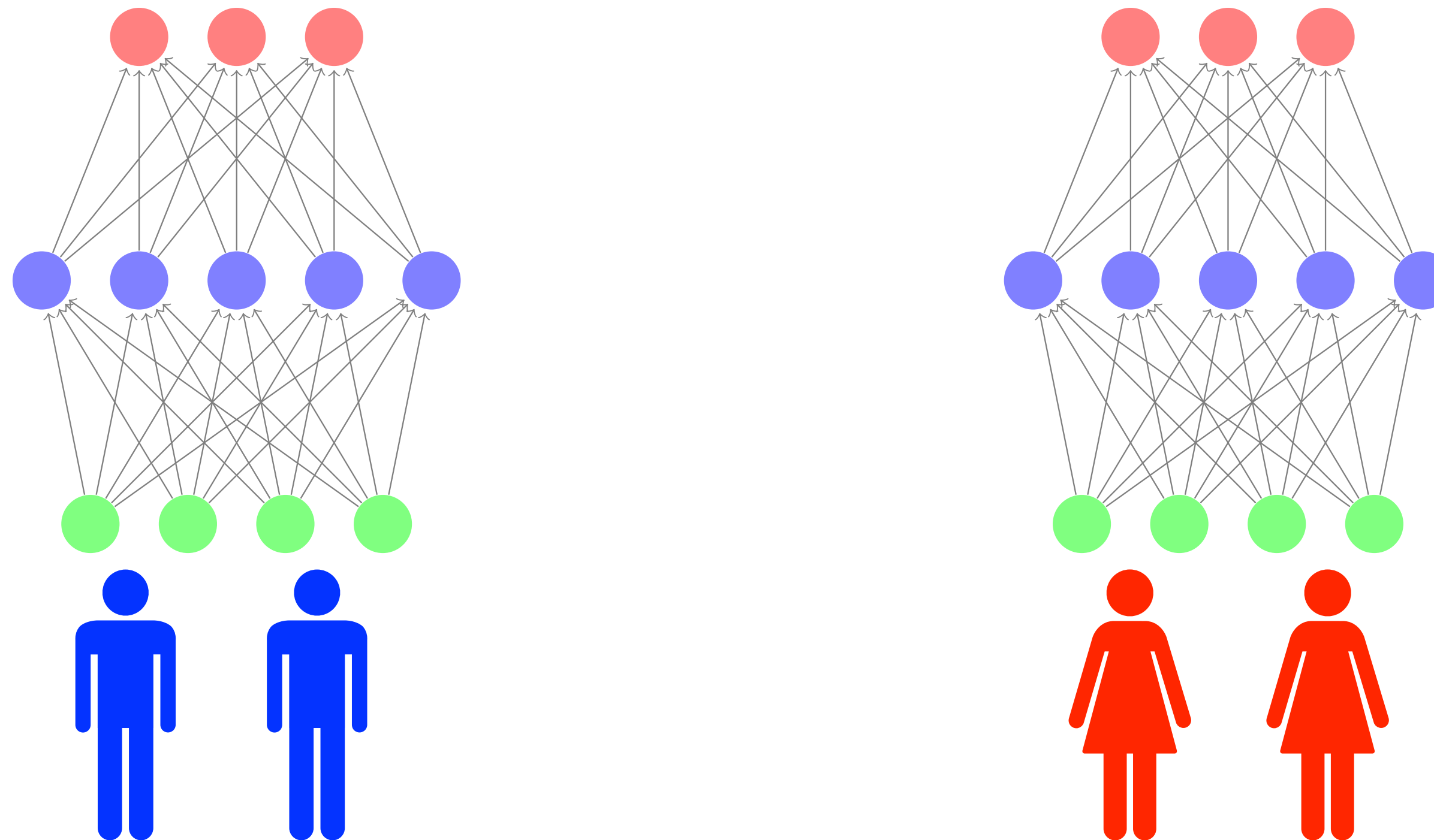
reduce training set accuracy to improve generalization



- Constrain predicted $P(\text{replay} \mid M) = P(\text{replay} \mid F)$
 - ▶ statistical parity (independence)

Enforce constraints to improve fairness

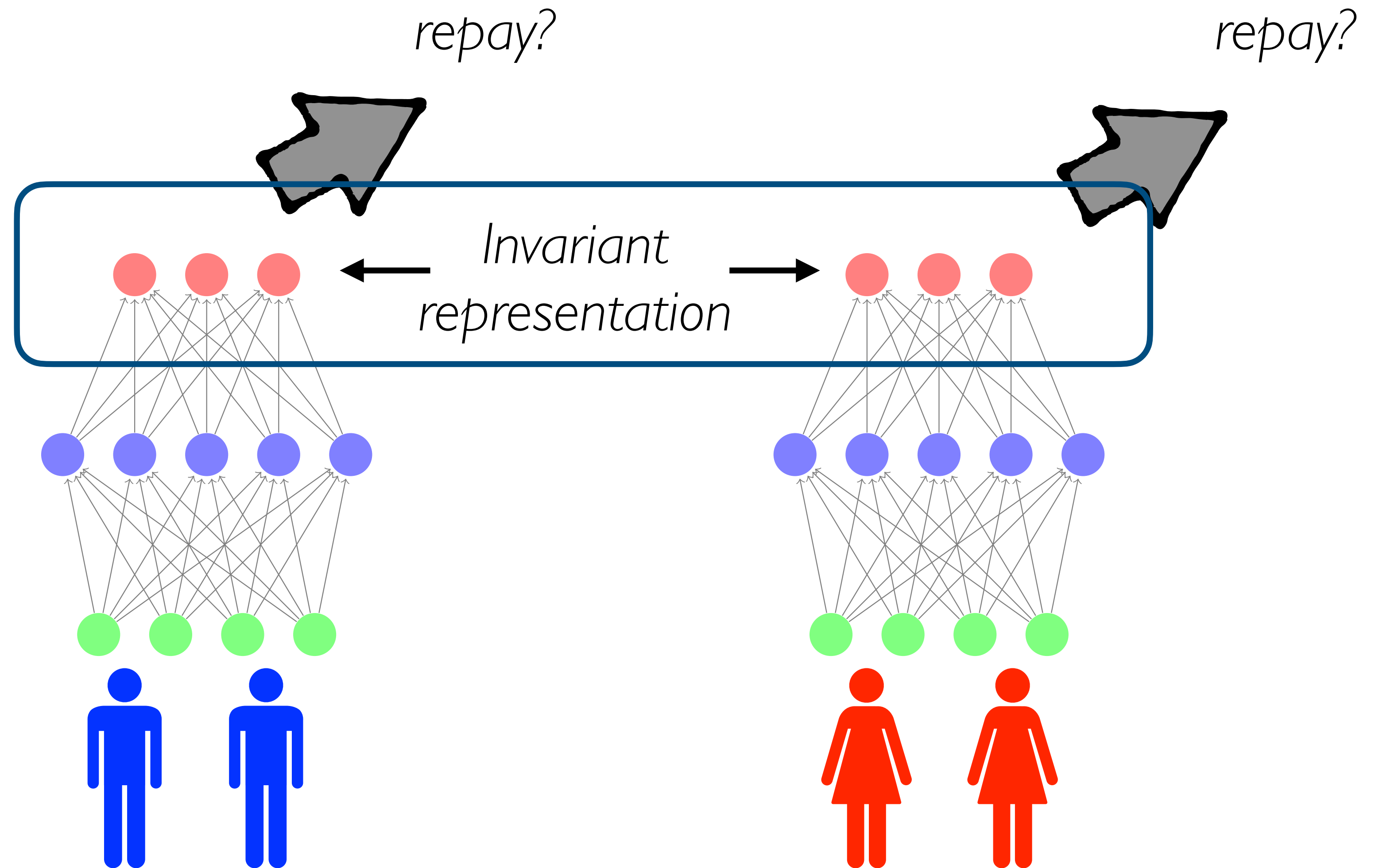
reduce training set accuracy to improve generalization



- Constrain predicted $P(\text{replay} \mid M) = P(\text{replay} \mid F)$
 - ▶ statistical parity (independence)

Enforce constraints to improve fairness

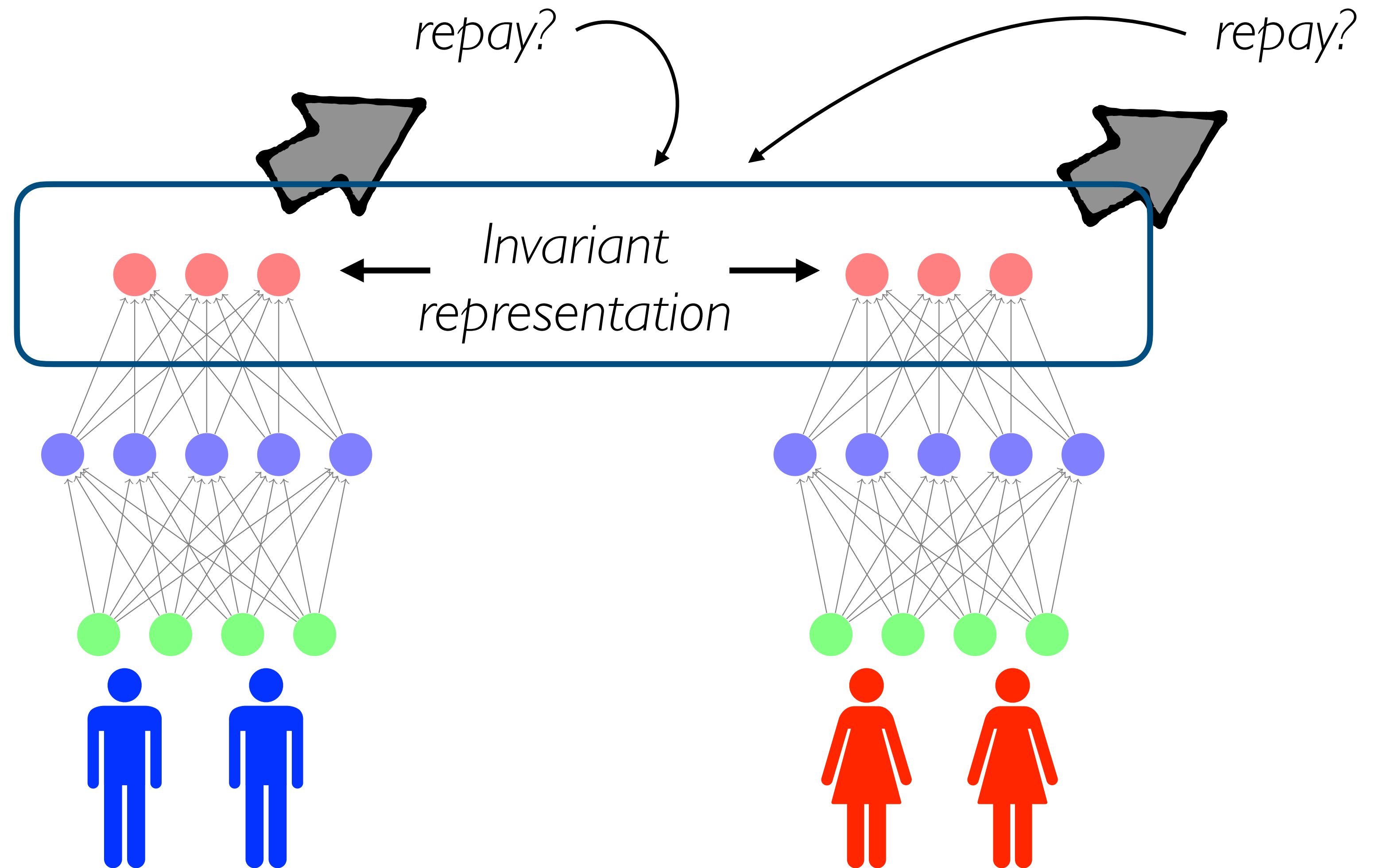
reduce training set accuracy to improve generalization



- Constrain $P(\text{representation} \mid M) = P(\text{representation} \mid F)$
 - ▶ statistical parity, but maybe more robust

Enforce constraints to improve fairness

reduce training set accuracy to improve generalization



- $P(\text{representation} \mid M, \text{replay}) = P(\text{representation} \mid F, \text{replay})$
 - ▶ separation: representation (and therefore prediction) \perp M/F given true label

Adversary

- Can express any of these constraints with an adversary
 - ▶ independence by invariant prediction: scalar constraint, adversary is a single real Lagrange multiplier
 - ▶ independence by invariant representation: much more complicated constraint, adversary needs to be an ML model
 - ▶ a **discriminator** that tries to classify representations to predict whether they came from M or F
 - ▶ if $P(M \mid \text{representation}) \neq P(M)$, adversary can win (equivalent to constraint above)
 - ▶ separation by invariant representation: discriminator now sees both representation and true label, predicts M/F
- Soft constraint: $\min_{\theta} \max_{\phi} \underbrace{L(\theta)}_{\text{original loss}} + \lambda \underbrace{D(\phi)}_{\text{discriminator accuracy}}$

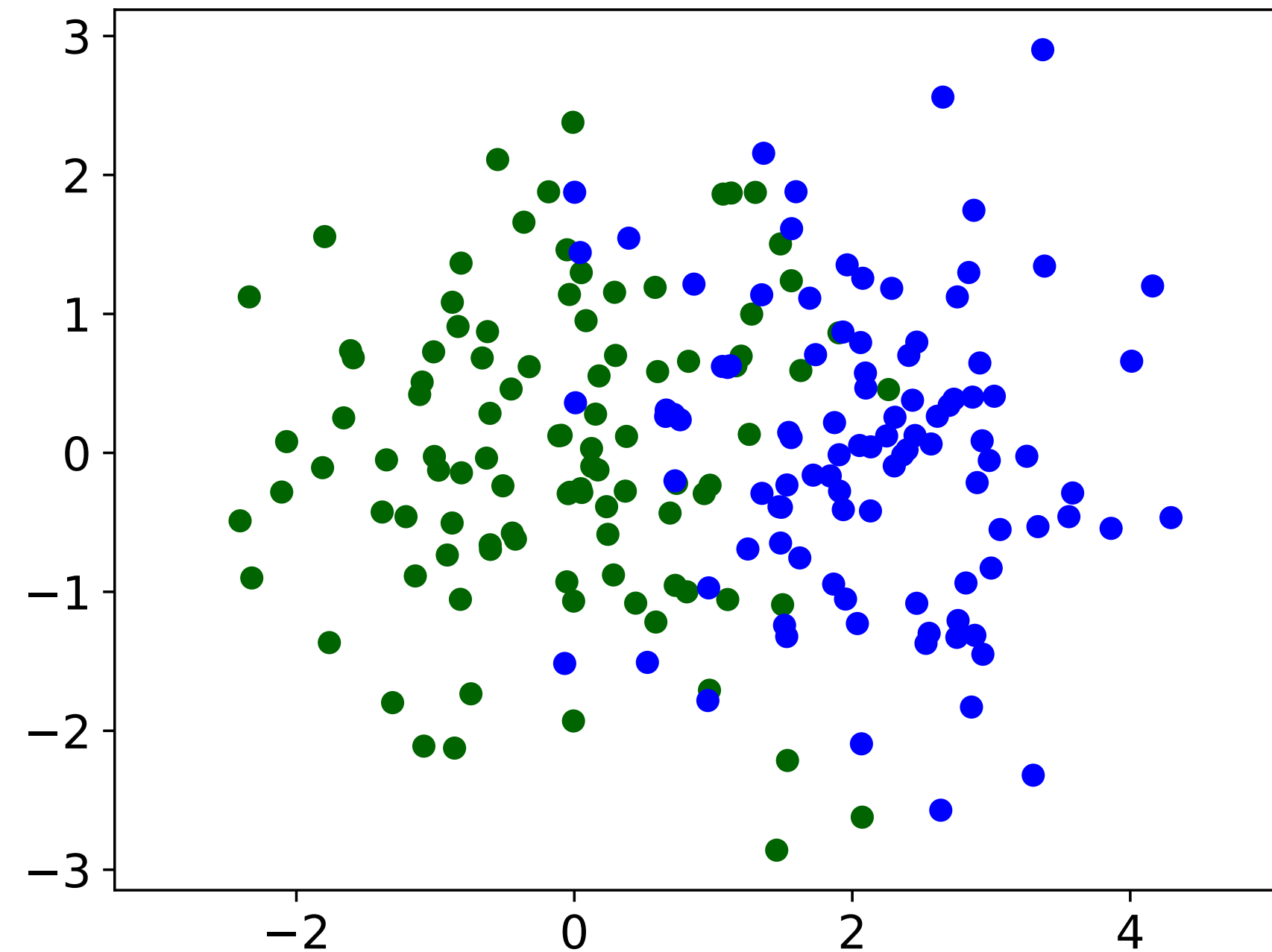
if conditional is marginal, we have dependence - discover by optimizing by prediction

$P(M)$
(typo)

ML model as adversary (discriminator)

simple adversaries like a linear function or a linear classifier are enough for simple differences

- Suppose our two groups have different distribution of representations:

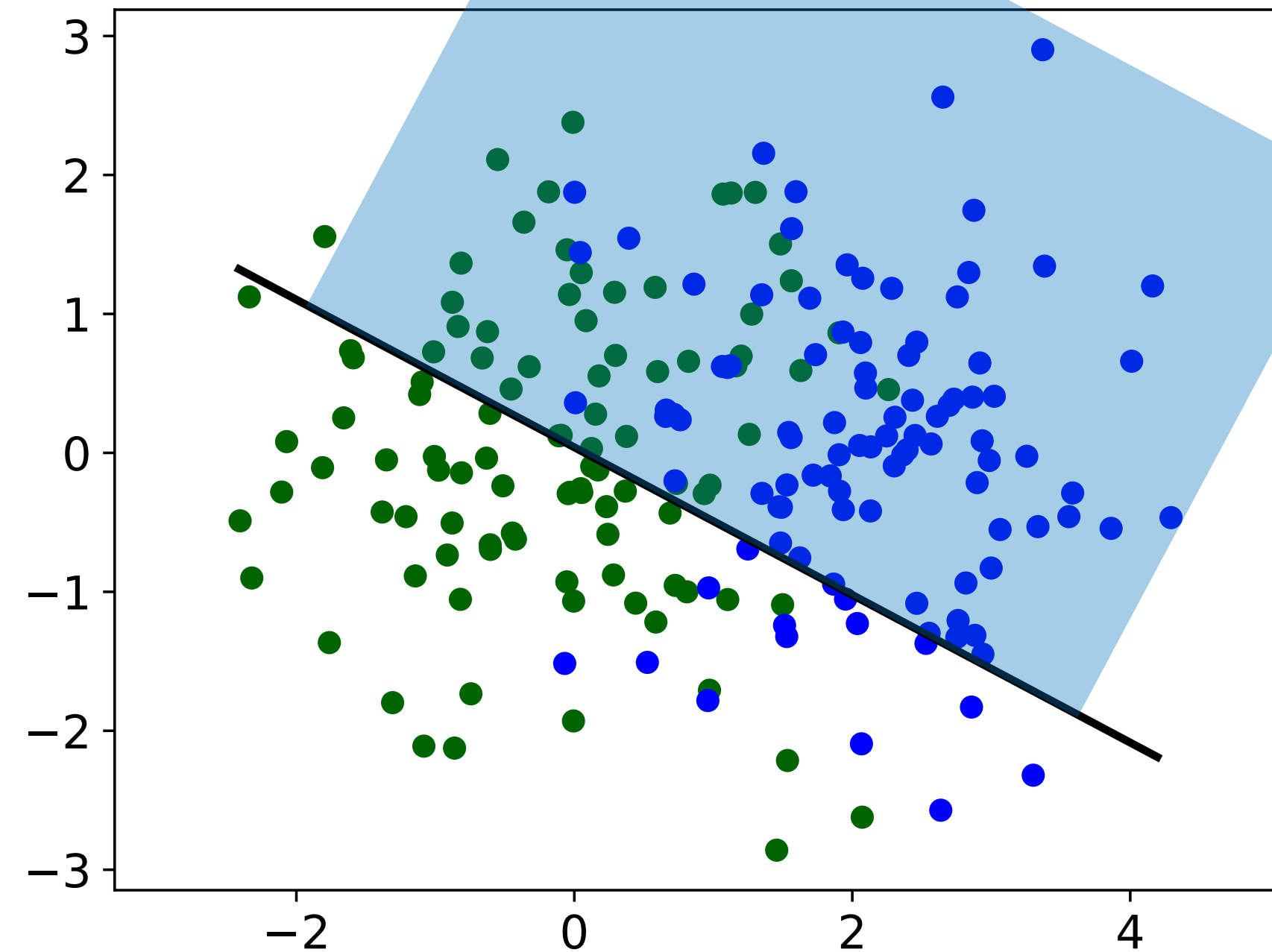


- Optimal discriminator tells us how they differ

ML model as adversary (discriminator)

simple adversaries like a linear function or a linear classifier are enough for simple differences

- Suppose our two groups have different distribution of representations:

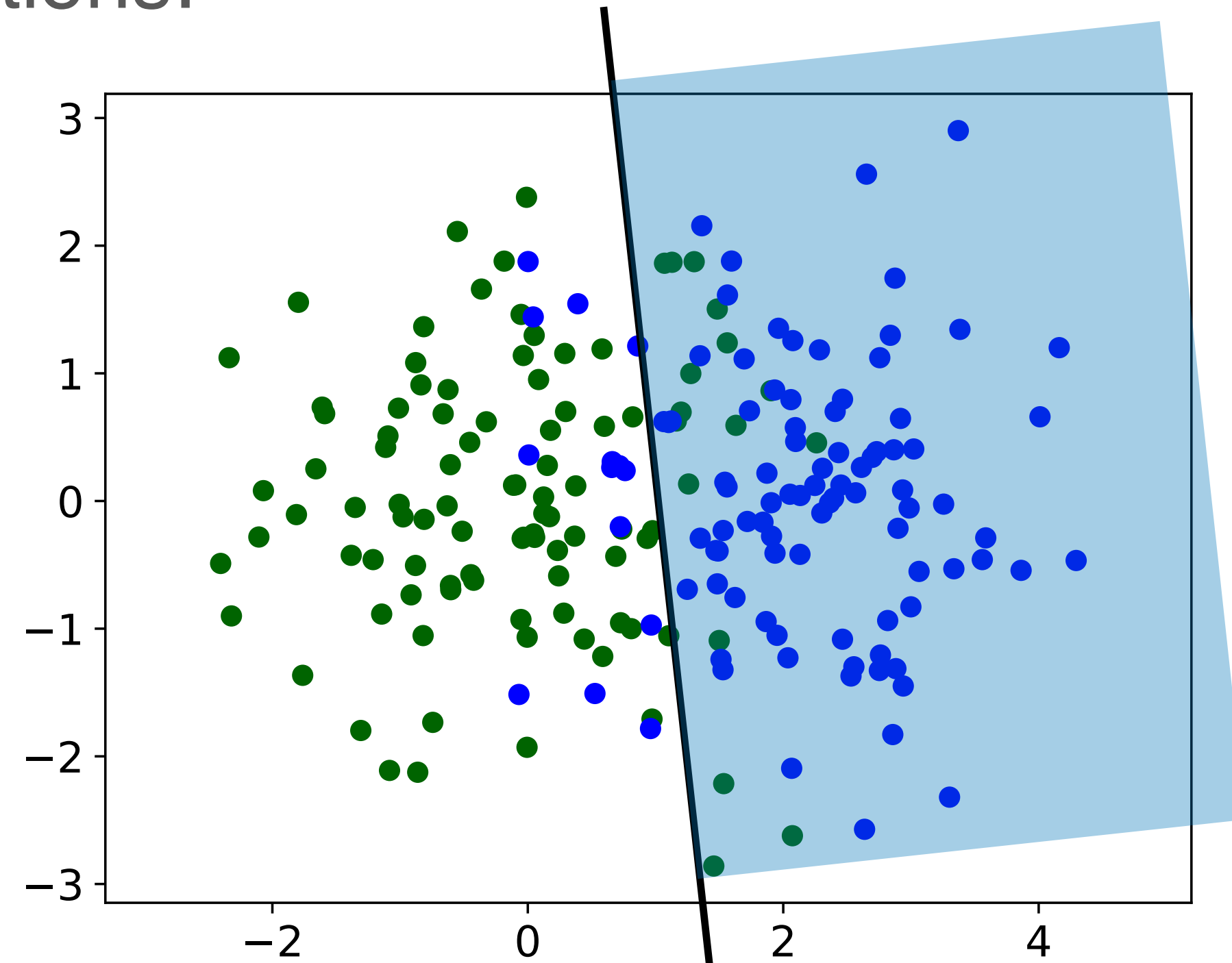


- Optimal discriminator tells us how they differ

ML model as adversary (discriminator)

simple adversaries like a linear function or a linear classifier are enough for simple differences

- Suppose our two groups have different distribution of representations:

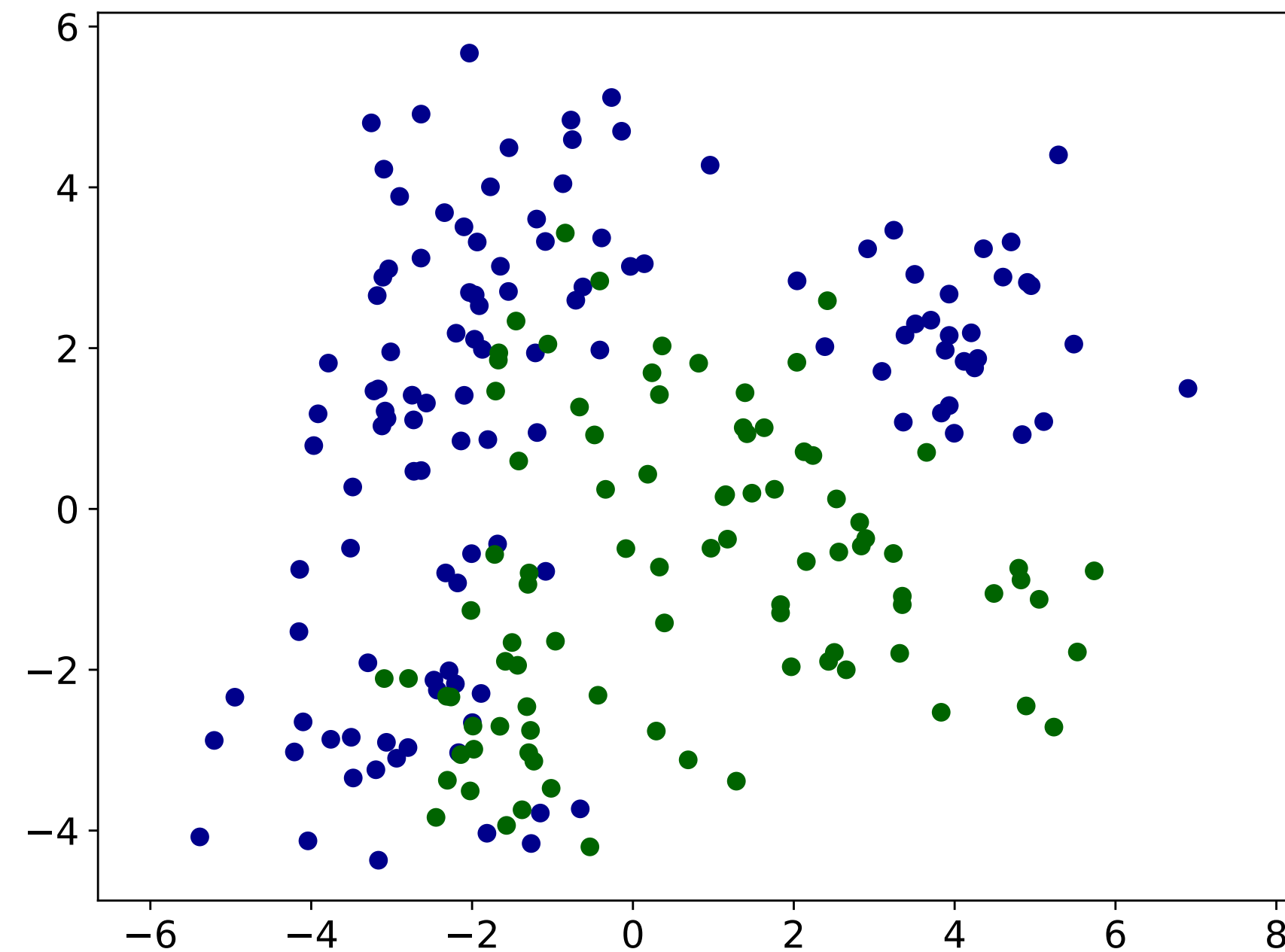


- Optimal discriminator tells us how they differ

ML model as adversary (discriminator)

*more-complicated
functions help to discover
more-complicated
differences*

- Suppose our two groups have different distribution of representations:

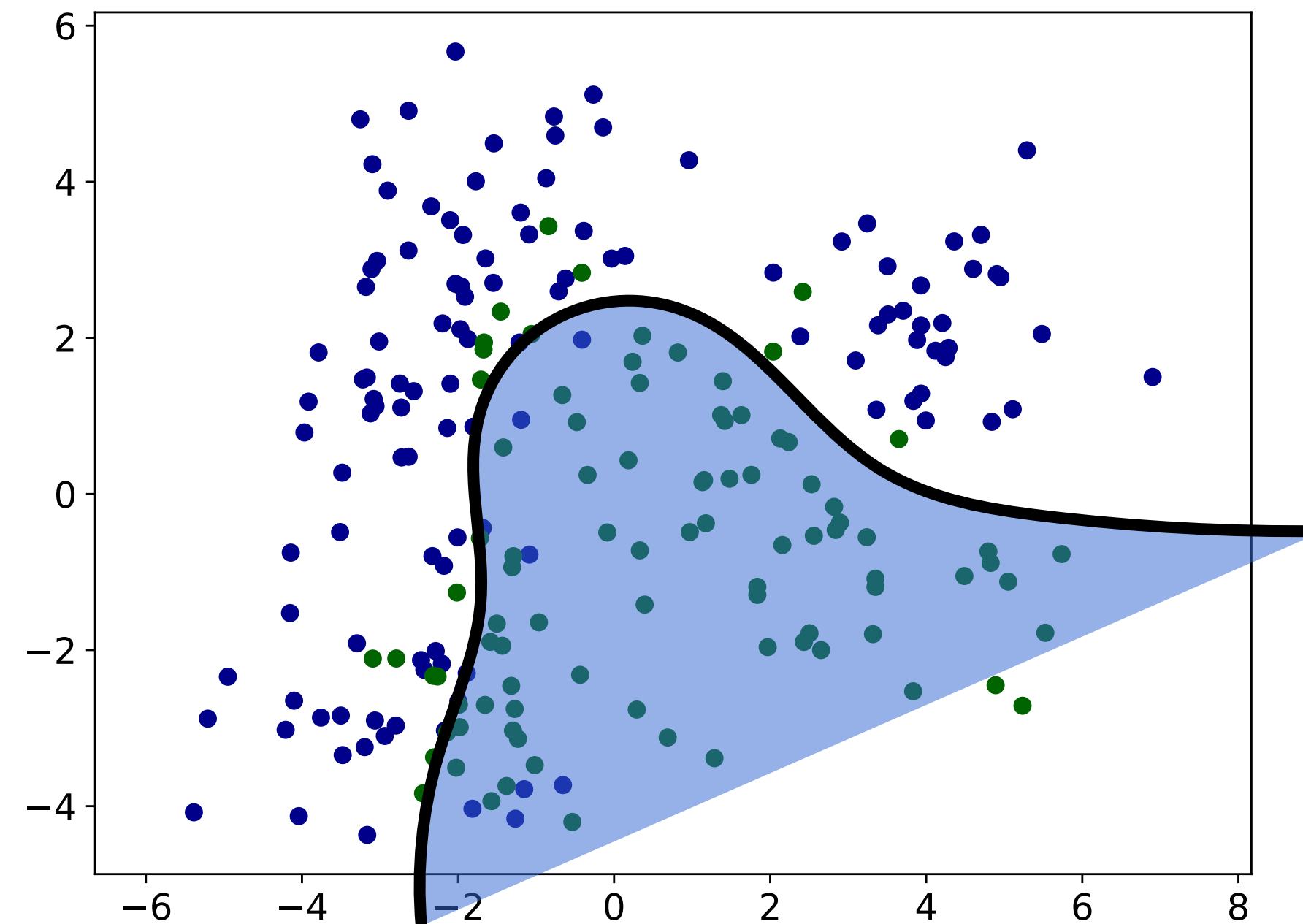


- Optimal discriminator tells us how they differ

ML model as adversary (discriminator)

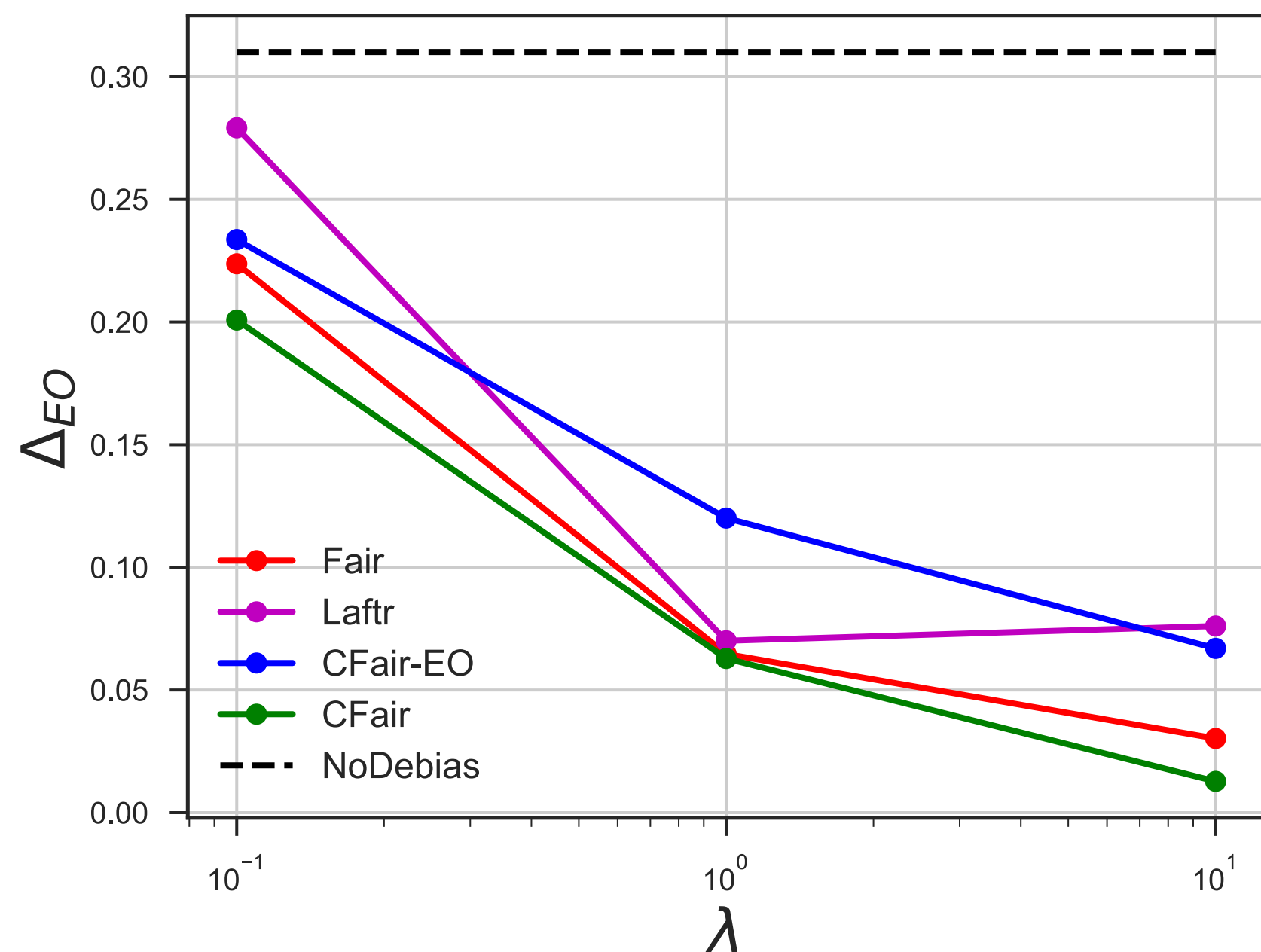
more-complicated functions help to discover more-complicated differences

- Suppose our two groups have different distribution of representations:

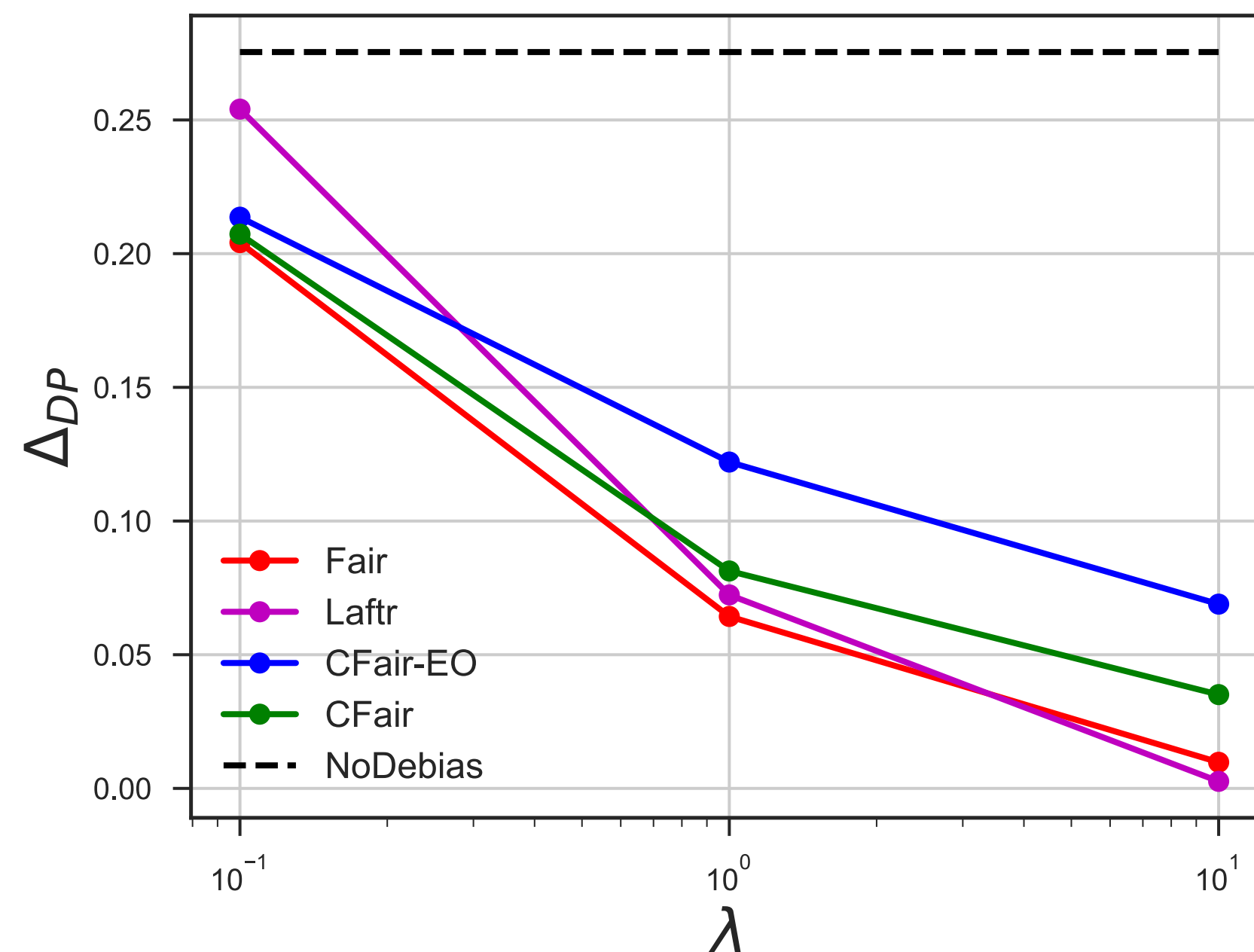


- Optimal discriminator tells us how they differ

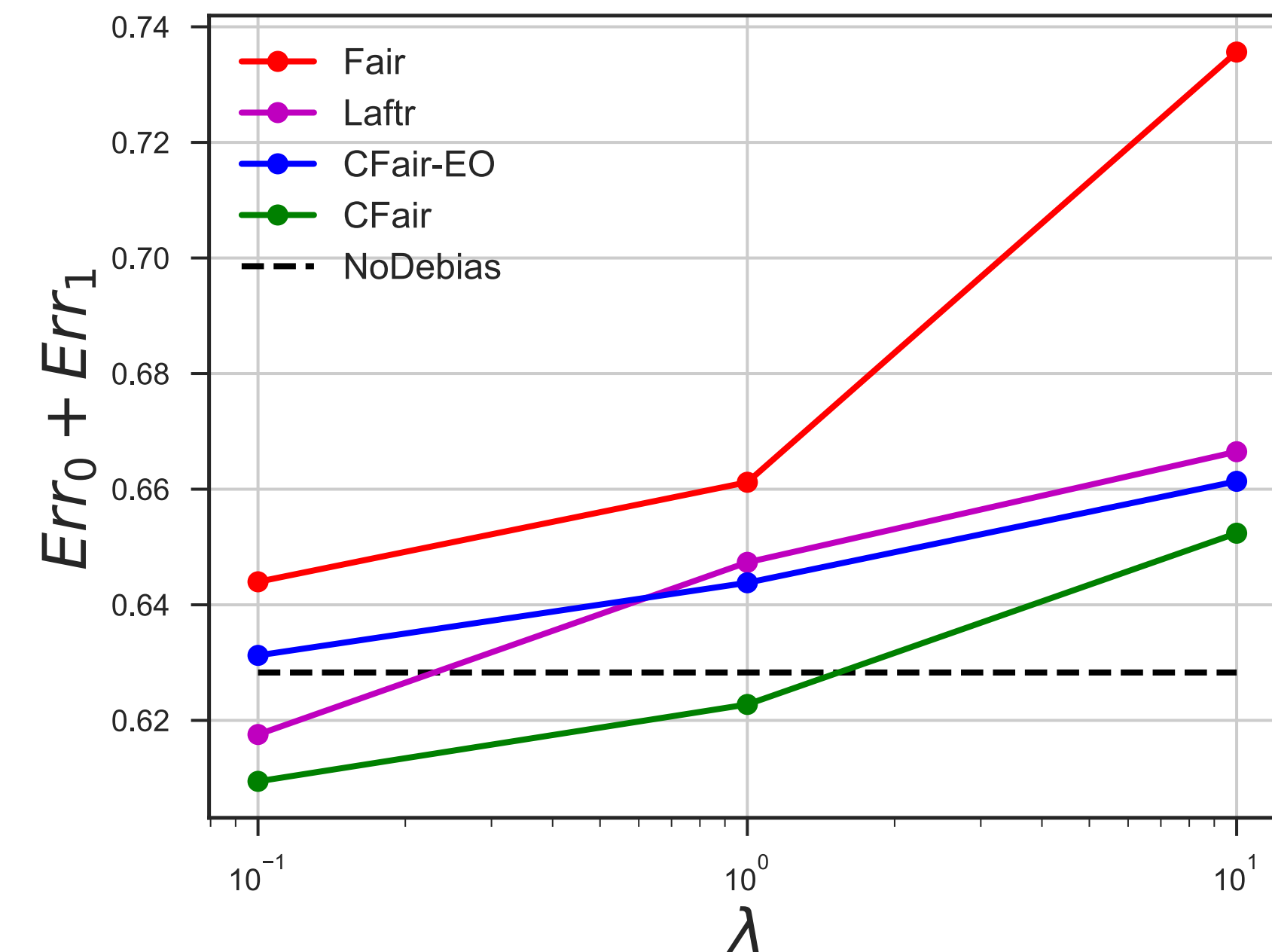
observed separation (equality of odds)



observed independence (demographic parity)



observed error rate



Experiment

- Compare observed separation, independence, and error rate for different soft constraints (strength λ)
 - ▶ — — no constraint; — independence; — separation
 - ▶ constraints enforced on *invariant representation*
- Separation seems to achieve a good tradeoff

[Zhao et al., ICLR 2020]

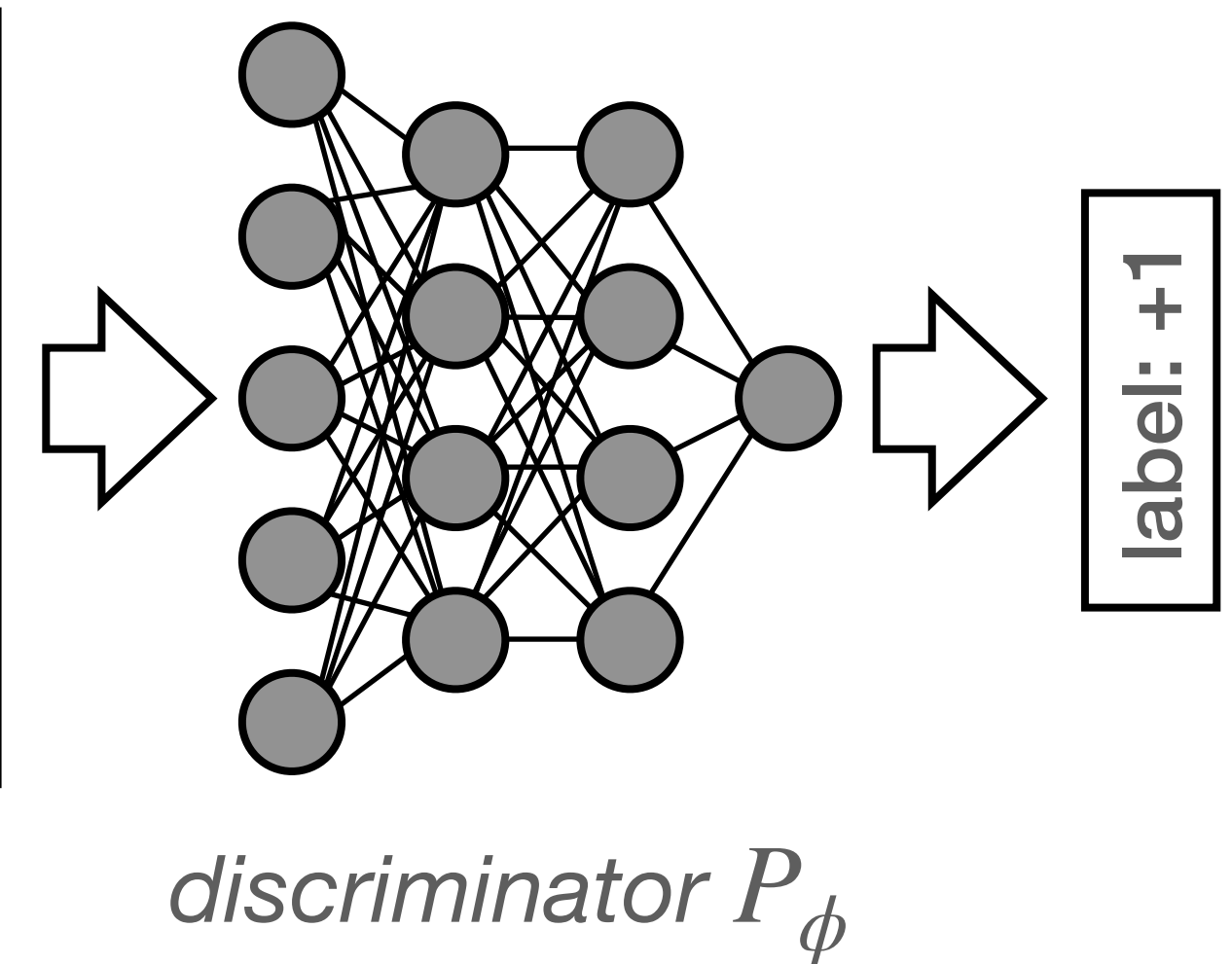
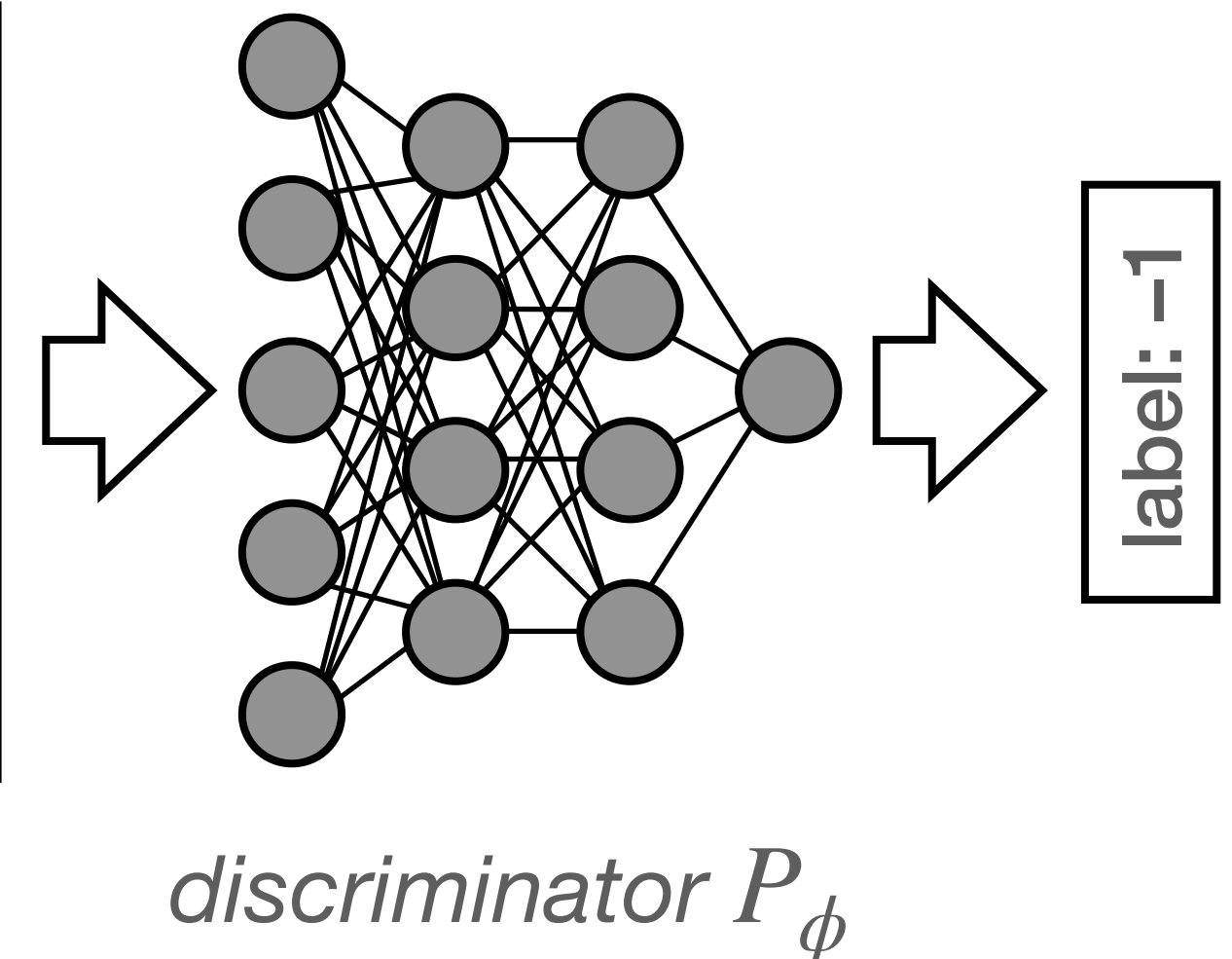
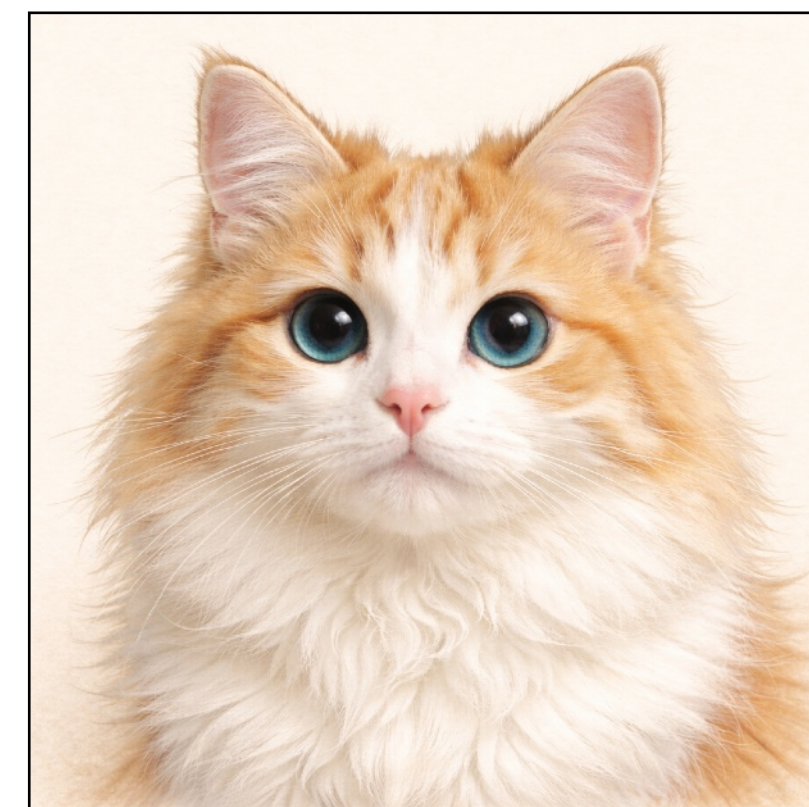
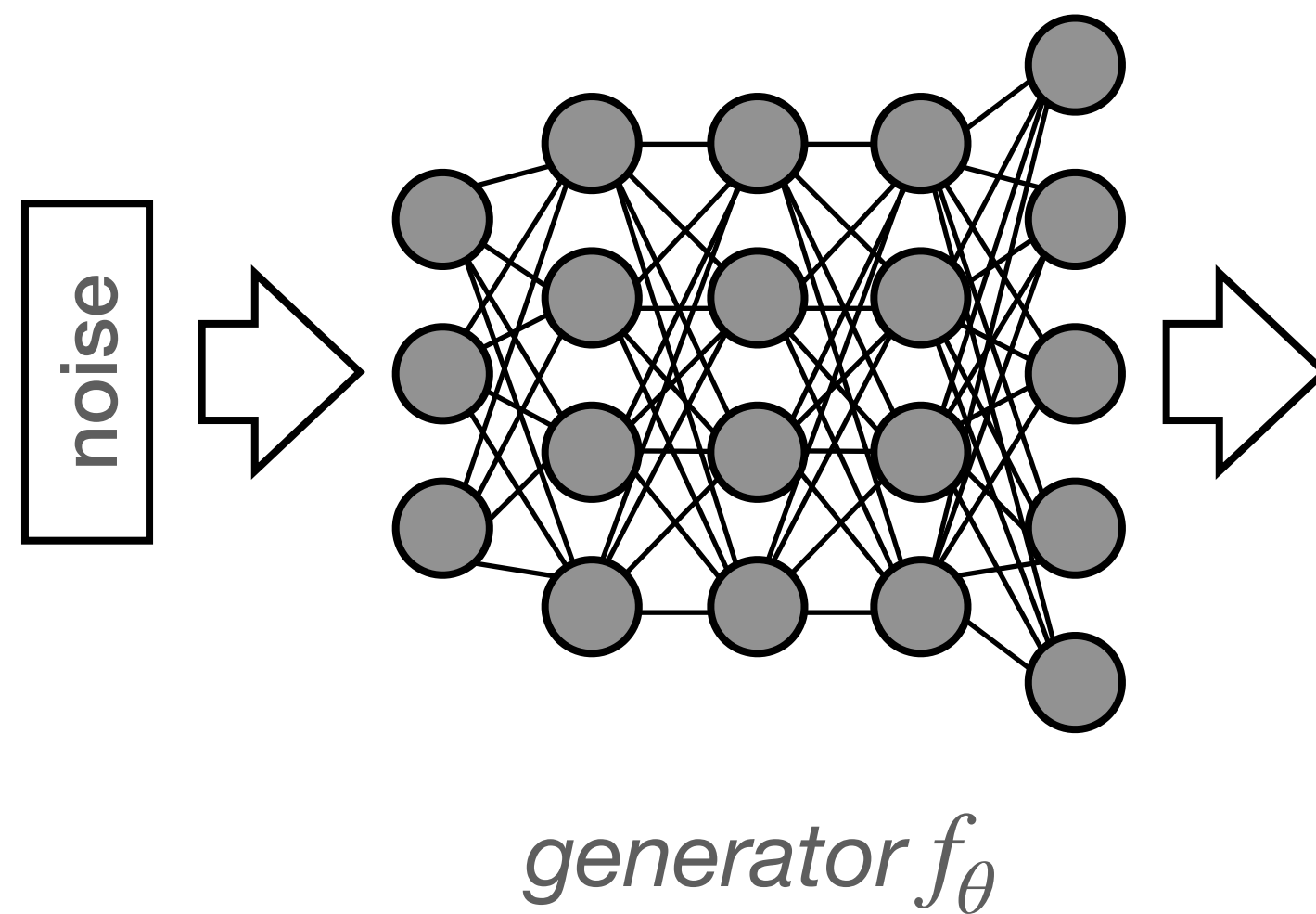
Example: generative adversarial networks

- Another constraint that's tough to represent:
 - ▶ train a neural net (generator) applied to a noise input: $f_{\theta}(z)$
 - ▶ goal: output distribution should match some given samples (e.g., cat pics downloaded from internet)
 - ▶ variant: *conditional* generation: generate cat pics on input "cat", ferret pics on input "ferret"
- Approach:
 - ▶ add an adversary (discriminator): classifier with parameters ϕ , maps images to real-valued score
 - ▶ discriminator tries to adjust ϕ to recognize generated images vs. real sampled cat pics: +ve score for real, -ve for generated
 - ▶ generator tries to adjust θ to fool discriminator

GAN min-max objective

overall log likelihood:
$$\frac{1}{2} \mathbb{E}(\ln P_{\phi}(-1 | f_{\theta}(z^{(i)}))) + \frac{1}{2} \mathbb{E}(\ln P_{\phi}(+1 | x^{(i)}))$$

noise samples $z^{(i)}$, real samples $x^{(i)}$
(many variations proposed)



GAN min-max objective

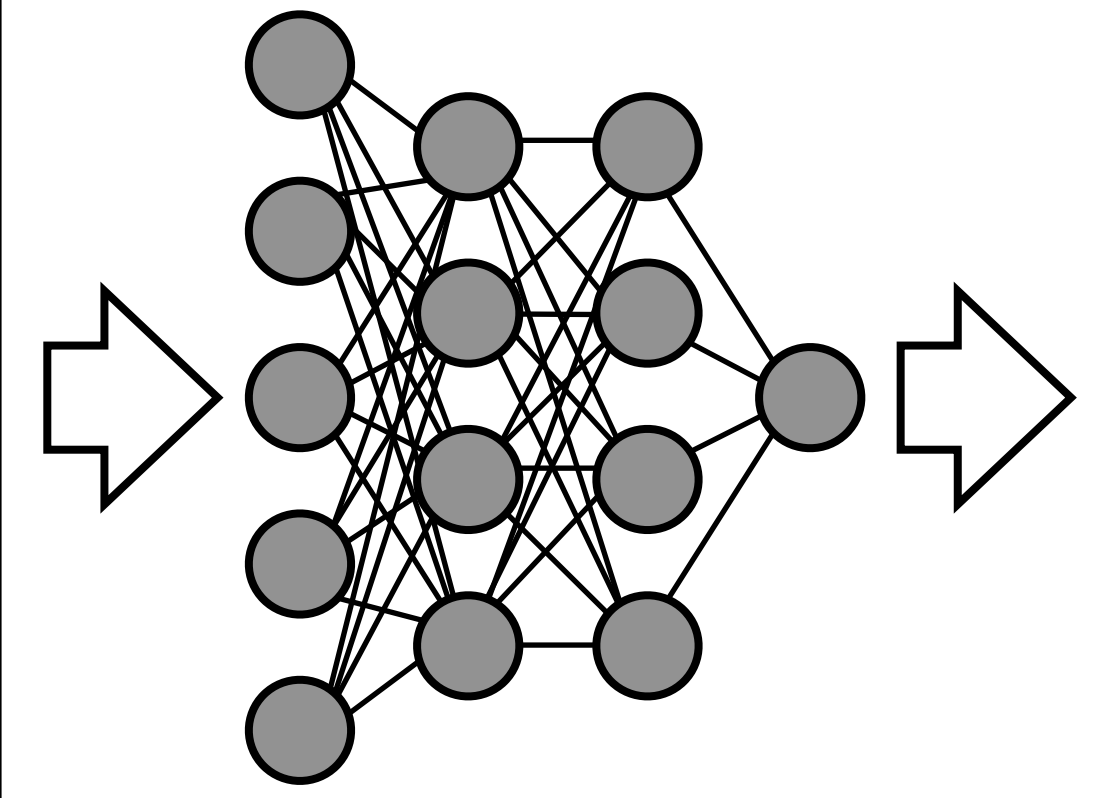
overall log likelihood:
$$\frac{1}{2} \mathbb{E}(\ln P_{\phi}(-1 | f_{\theta}(z^{(i)}))) + \frac{1}{2} \mathbb{E}(\ln P_{\phi}(+1 | x^{(i)}))$$

noise samples $z^{(i)}$, real samples $x^{(i)}$
(many variations proposed)

min log likelihood wrt θ

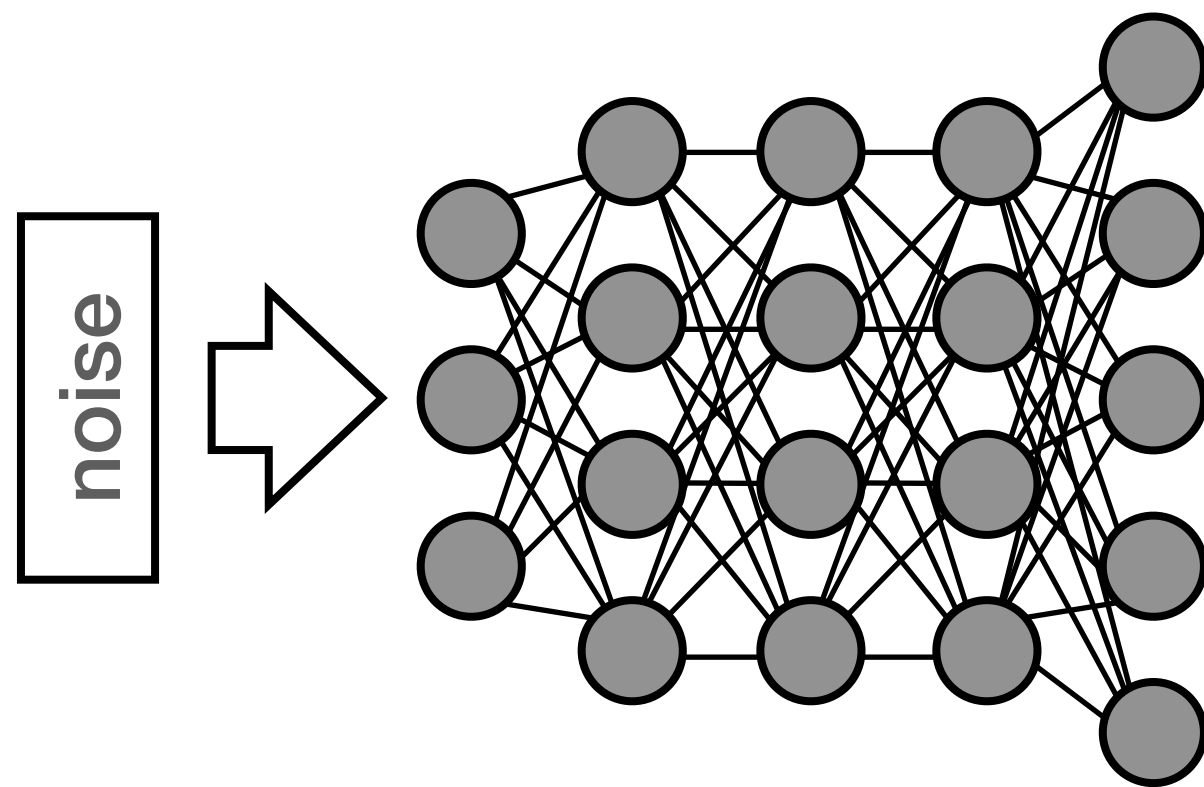


real image

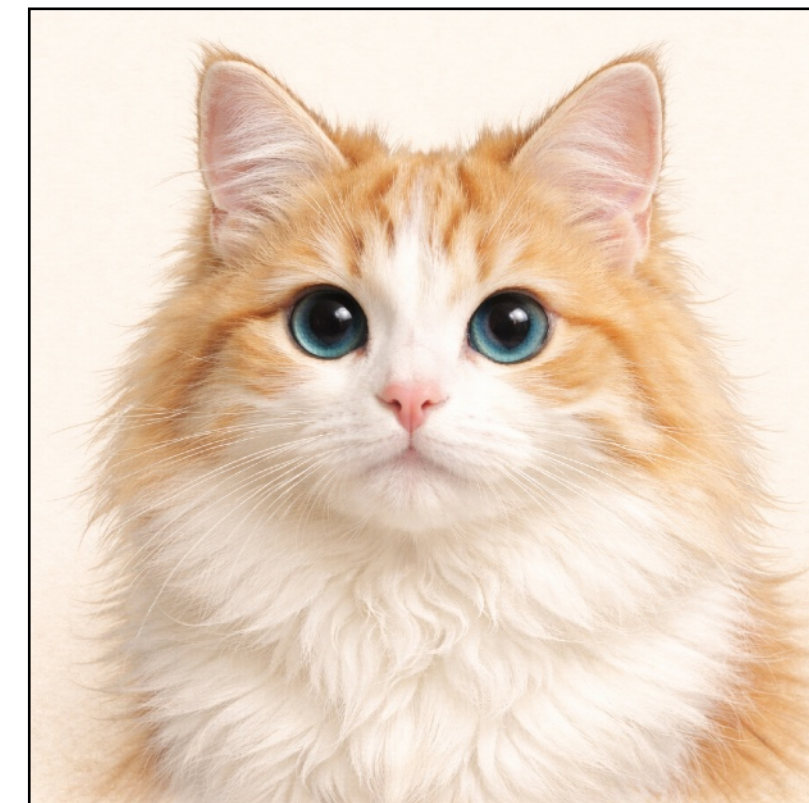


label: +1

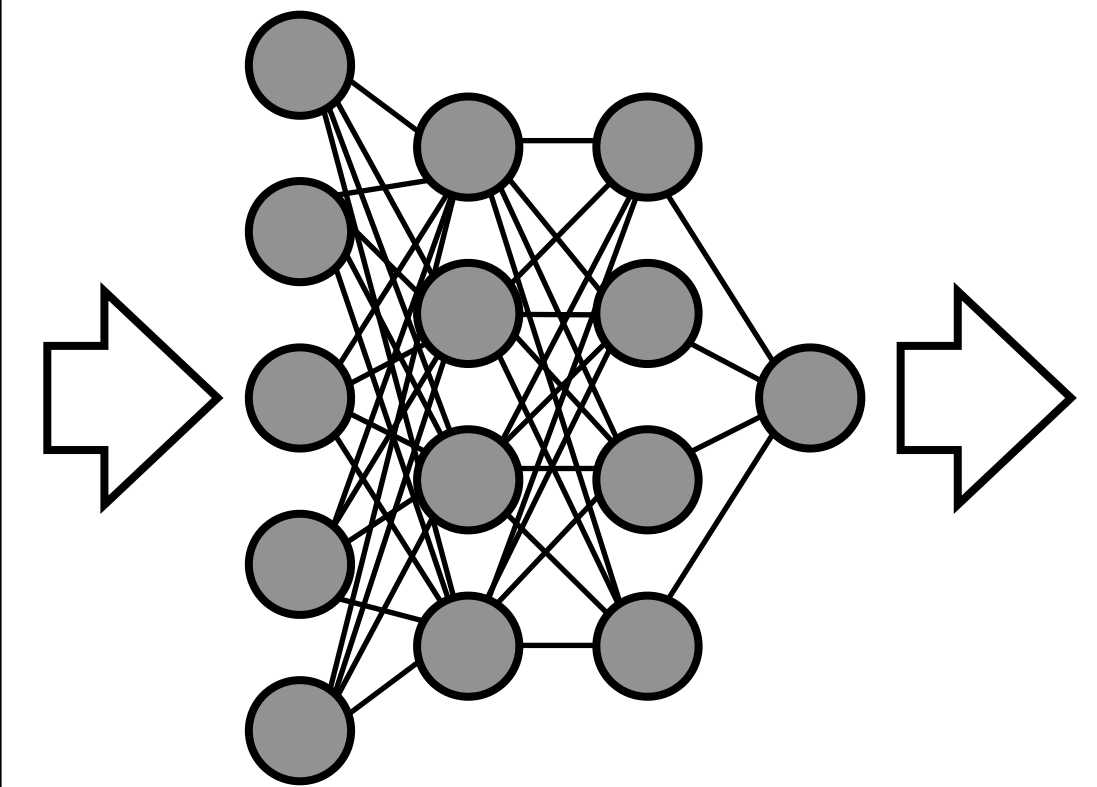
discriminator P_{ϕ}



generator f_{θ}



generated image



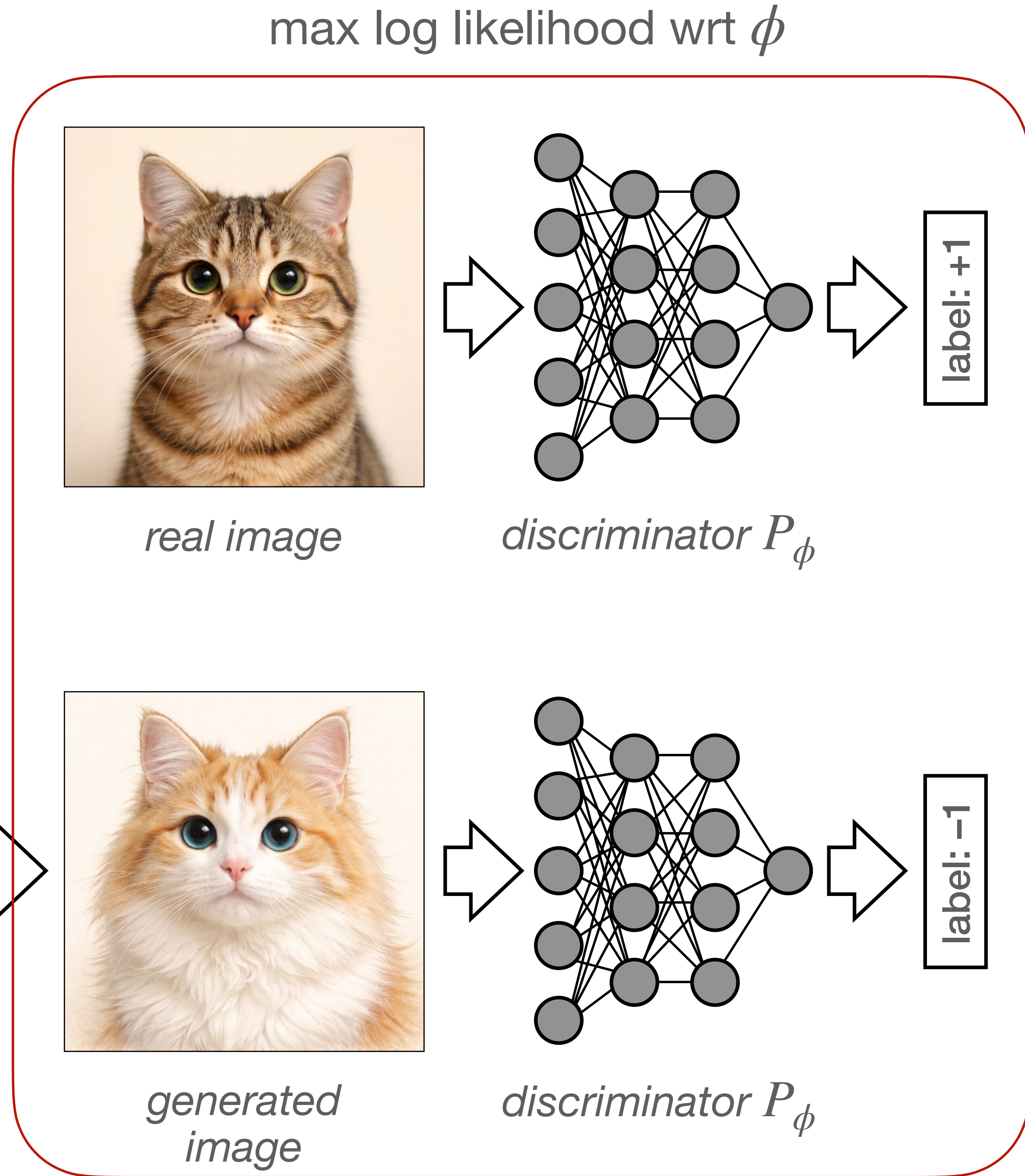
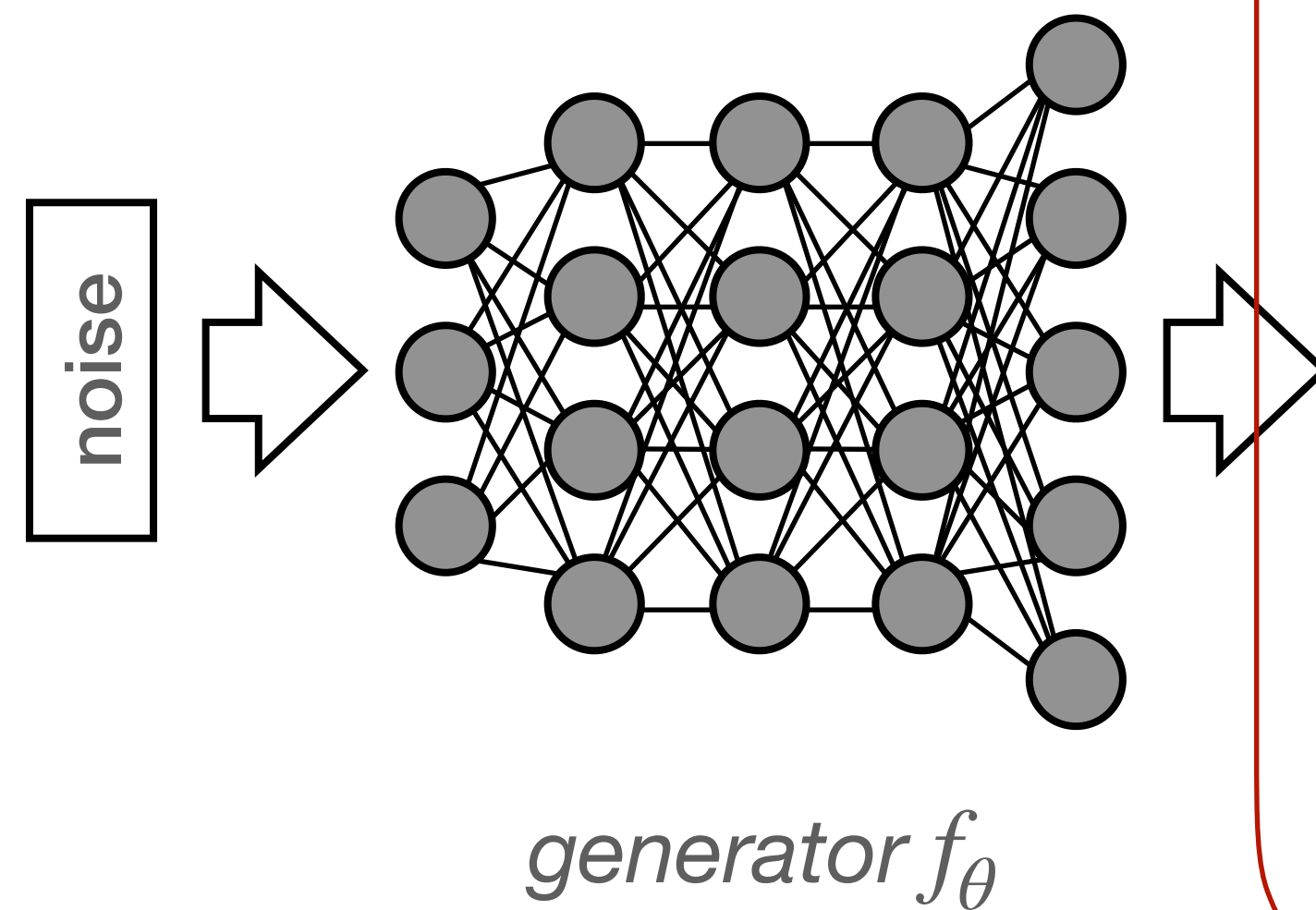
label: -1

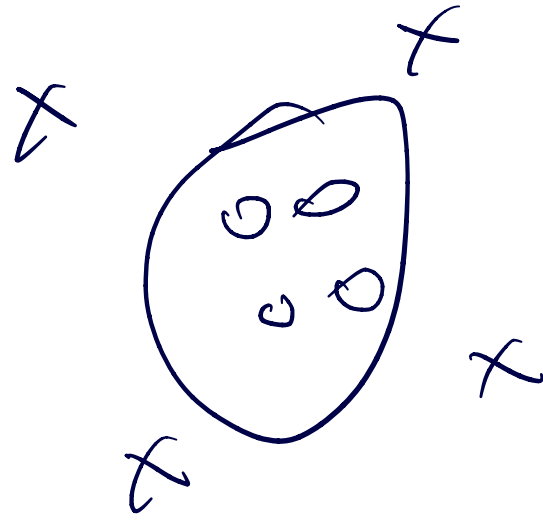
discriminator P_{ϕ}

GAN min-max objective

overall log likelihood:
$$\frac{1}{2} \mathbb{E}(\ln P_{\phi}(-1 | f_{\theta}(z^{(i)}))) + \frac{1}{2} \mathbb{E}(\ln P_{\phi}(+1 | x^{(i)}))$$


noise samples $z^{(i)}$, real samples $x^{(i)}$
(many variations proposed)



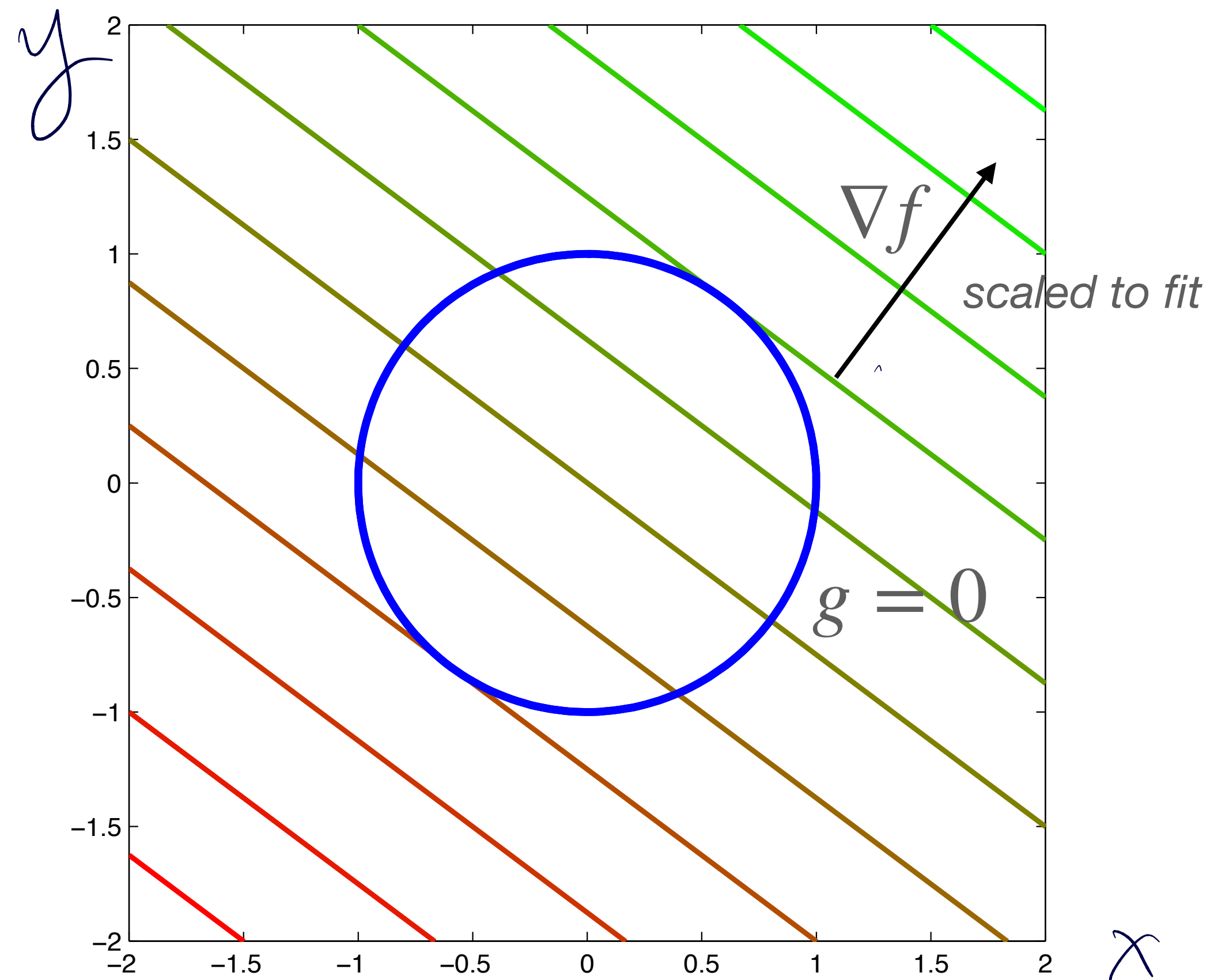


Poll



- Which of the following statements are true about adversarial learning?
 - A. A linear adversary $a_w(z) = w \cdot z$, $a_w \in \mathbb{R}^d \rightarrow \mathbb{R}$, can constrain two learned functions $f_\theta(x^{(i)}) \in \mathbb{R}^d$ and $g_\theta(x^{(i)}) \in \mathbb{R}^d$ to have the same mean on our dataset, even if f and g are nonlinear in both x and θ
 - B. A linear adversary can't directly enforce a nonlinear constraint like equal variance, but we can help it do so by defining new feature functions like $f_\theta(x^{(i)})^2$
 - C. If we don't know what feature functions to define, it can help to make the adversary more powerful — e.g., a deep net instead of a linear function
 - D. All of the above 
 - E. None of the above

Solving adversarial learning objectives



- For simple problems, we can find equilibria exactly

- E.g.: $\max_{x,y} f(x, y)$ s.t. $g(x, y) = 0$ where

$$f(x, y) = 6x + 8y$$

$$g(x, y) = x^2 + y^2 - 1$$

Construct and solve min-max problem

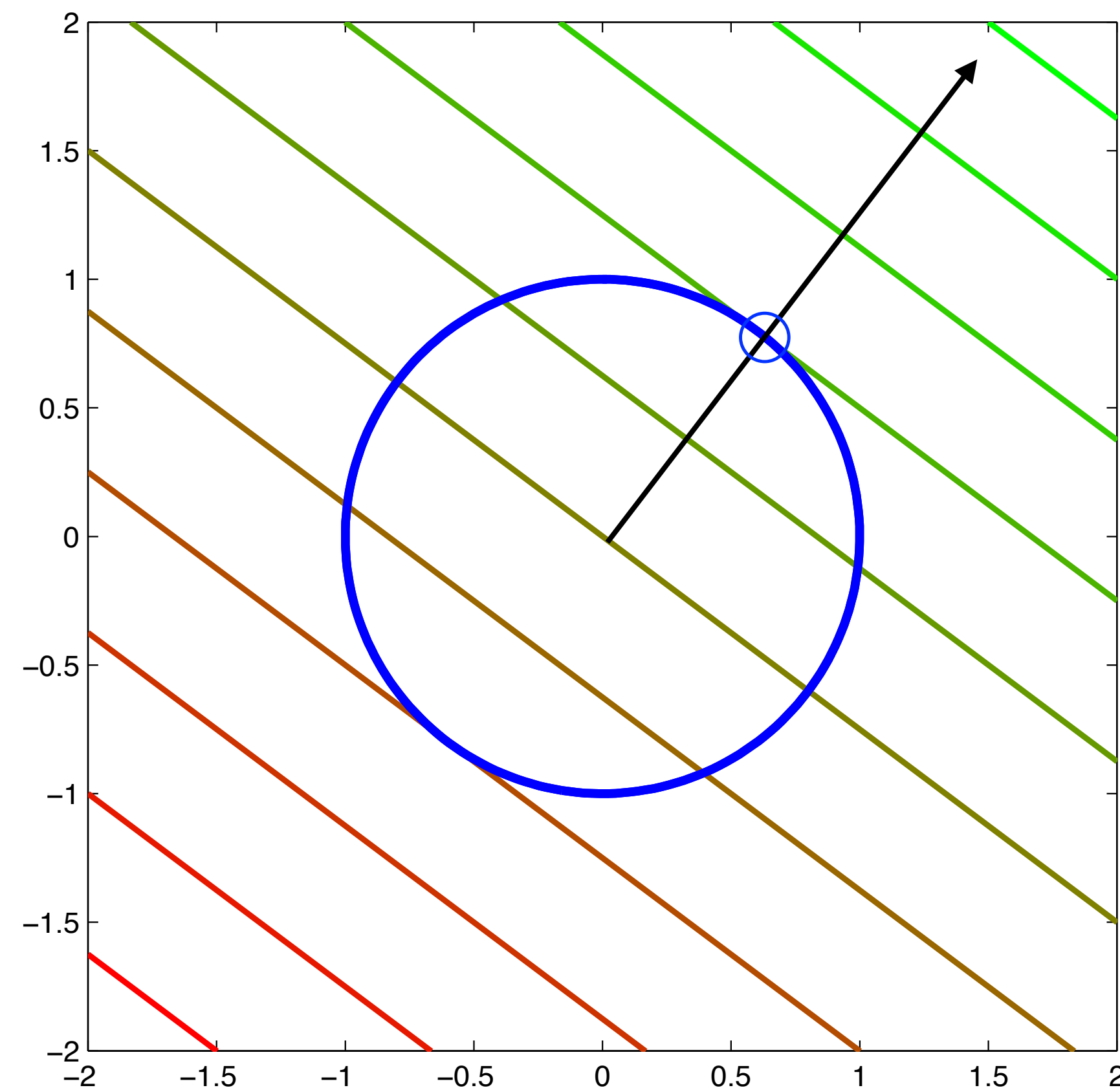
$$f(x, y) = 6x + 8y$$
$$g(x, y) = x^2 + y^2 - 1$$

- Enforce equality with an adversary (Lagrange multiplier)
 - ▶ $\min_{x,y} \max_a L(x, y, a) = f(x, y) + ag(x, y)$
- At equilibrium, gradients are 0:

$$0 = \nabla_{x,y} L = \begin{pmatrix} 6 \\ 8 \end{pmatrix} + a 2 \begin{pmatrix} x \\ y \end{pmatrix} \rightarrow \begin{pmatrix} x \\ y \end{pmatrix} \text{ is multiple of } \begin{pmatrix} 6 \\ 8 \end{pmatrix}$$

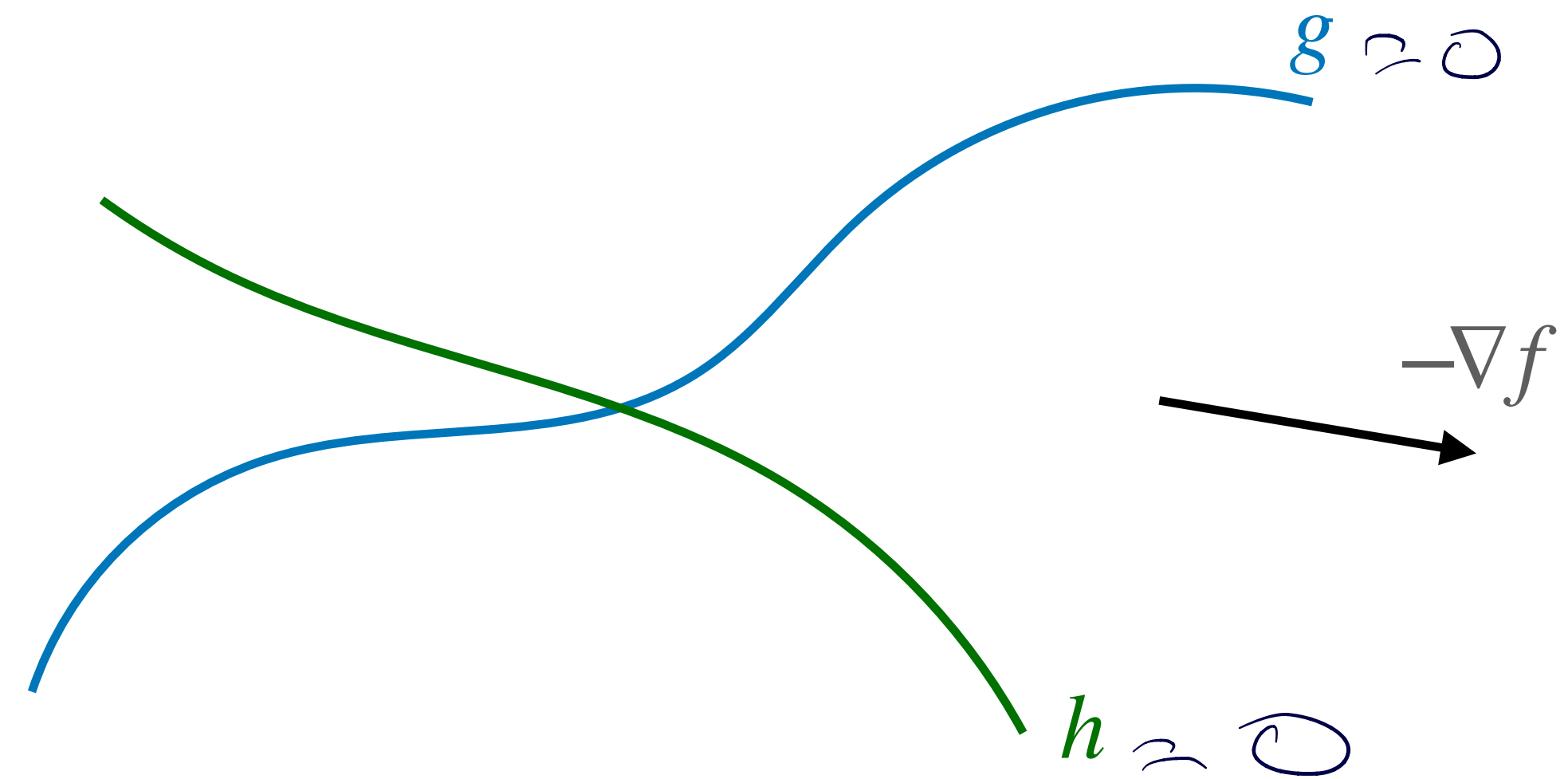
$$0 = \nabla_a L = g(x, y) \rightarrow \text{on unit } \odot$$

Solution



- So: (x, y) are a multiple of $(6, 8)$ and also lie on unit circle: must be $(\frac{3}{5}, \frac{4}{5})$

Multiple constraints



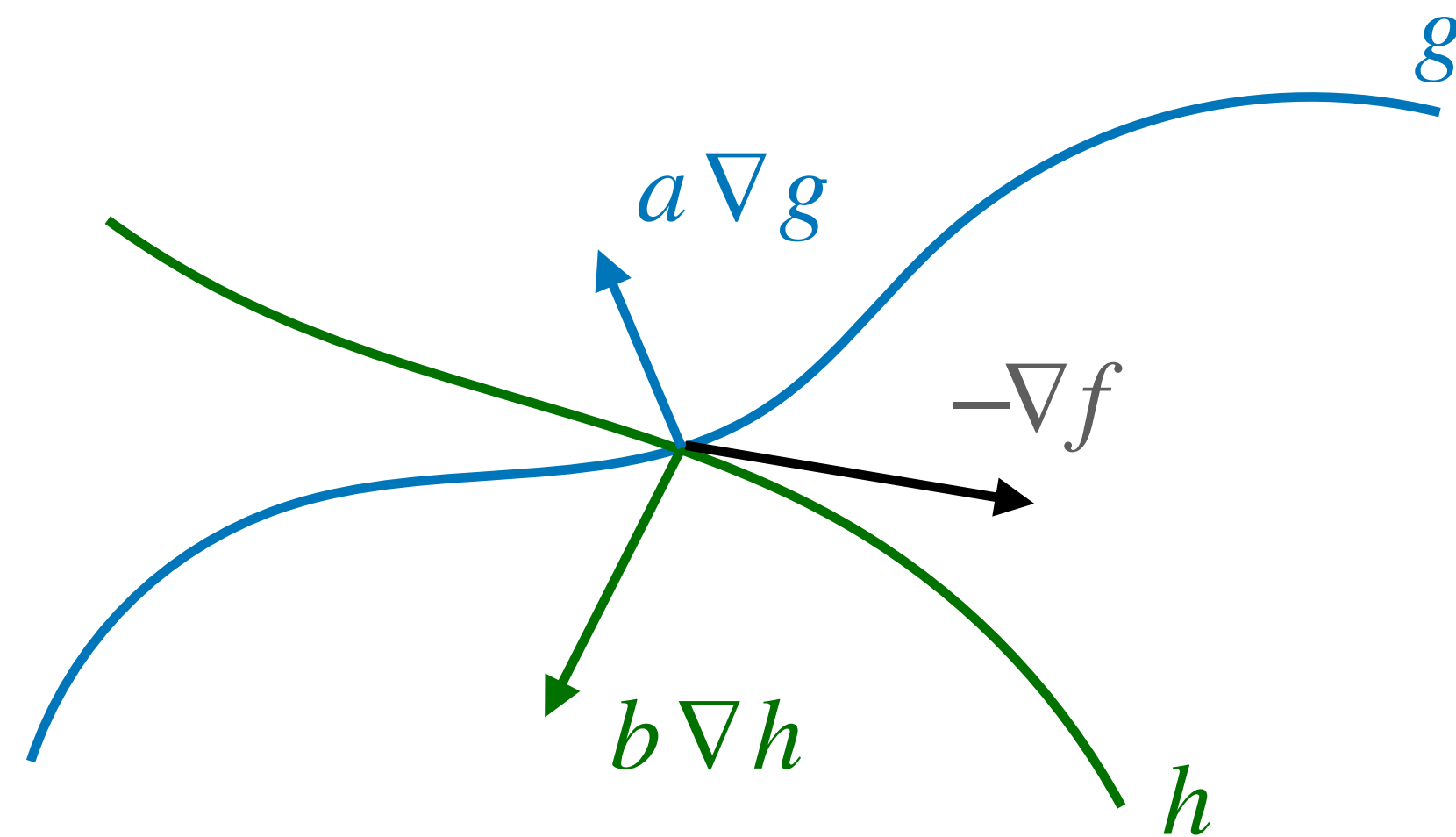
- We enforced an equality constraint by adding a single real-valued Lagrange multiplier
- It works equally well for multiple constraints: one multiplier per constraint

$$\min_{x,y} \max_{a,b} f(x,y) \text{ s.t. } g(x,y) = 0, h(x,y) = 0$$

$$\min_{x,y} \max_{a,b} L(x,y,a) = f(x,y) + ag(x,y) + bh(x,y)$$

- Interpretation: $a \nabla g(x,y)$, $b \nabla h(x,y)$, $-\nabla f$ are forces acting on (x,y) — at optimum, forces are in equilibrium

Multiple constraints



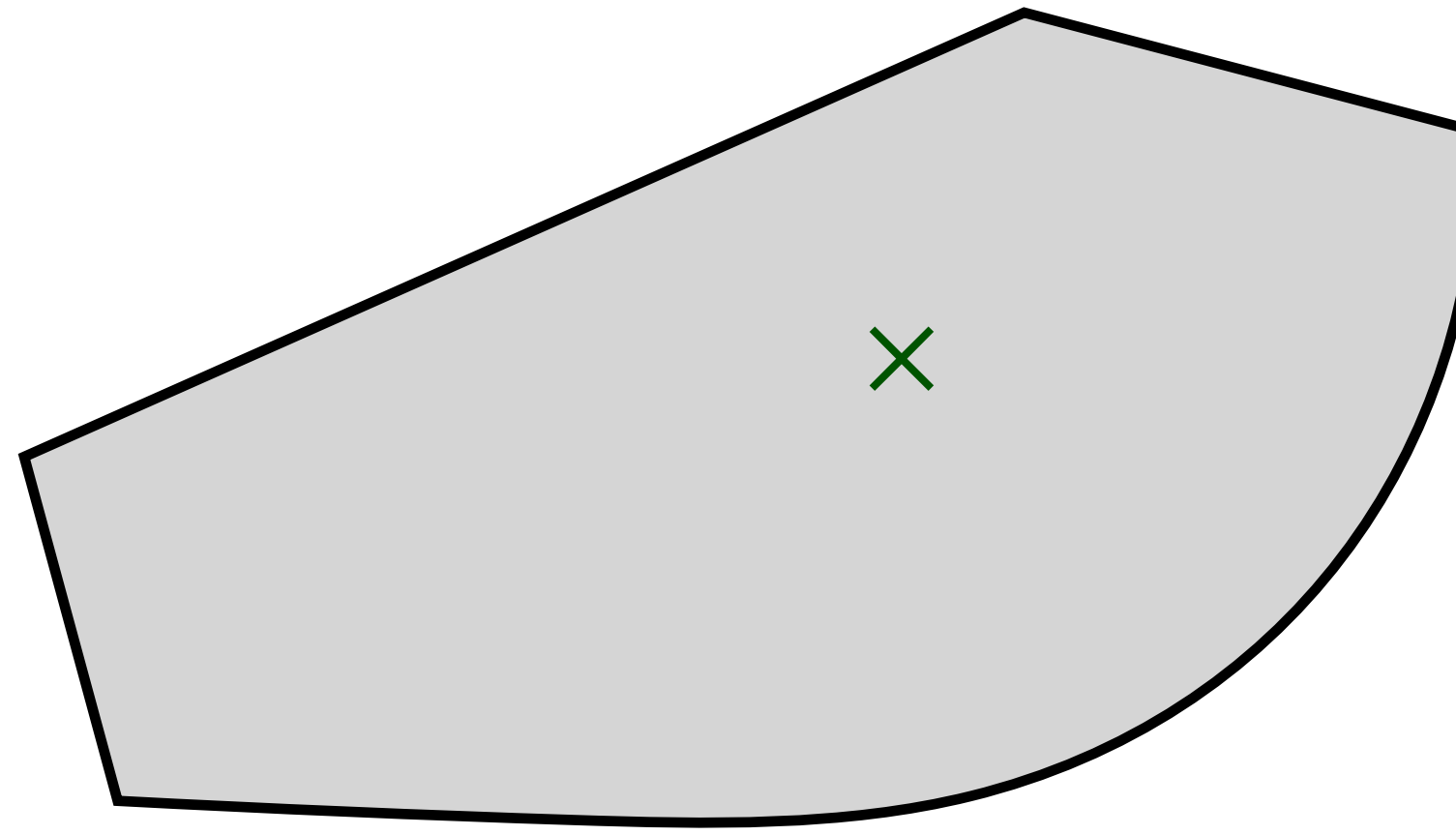
- We enforced an equality constraint by adding a single real-valued Lagrange multiplier
- It works equally well for multiple constraints: one multiplier per constraint

$$\max_{x,y} f(x, y) \text{ s.t. } g(x, y) = 0, h(x, y) = 0$$

$$\min_{x,y} \max_{a,b} L(x, y, a) = f(x, y) + ag(x, y) + bh(x, y)$$

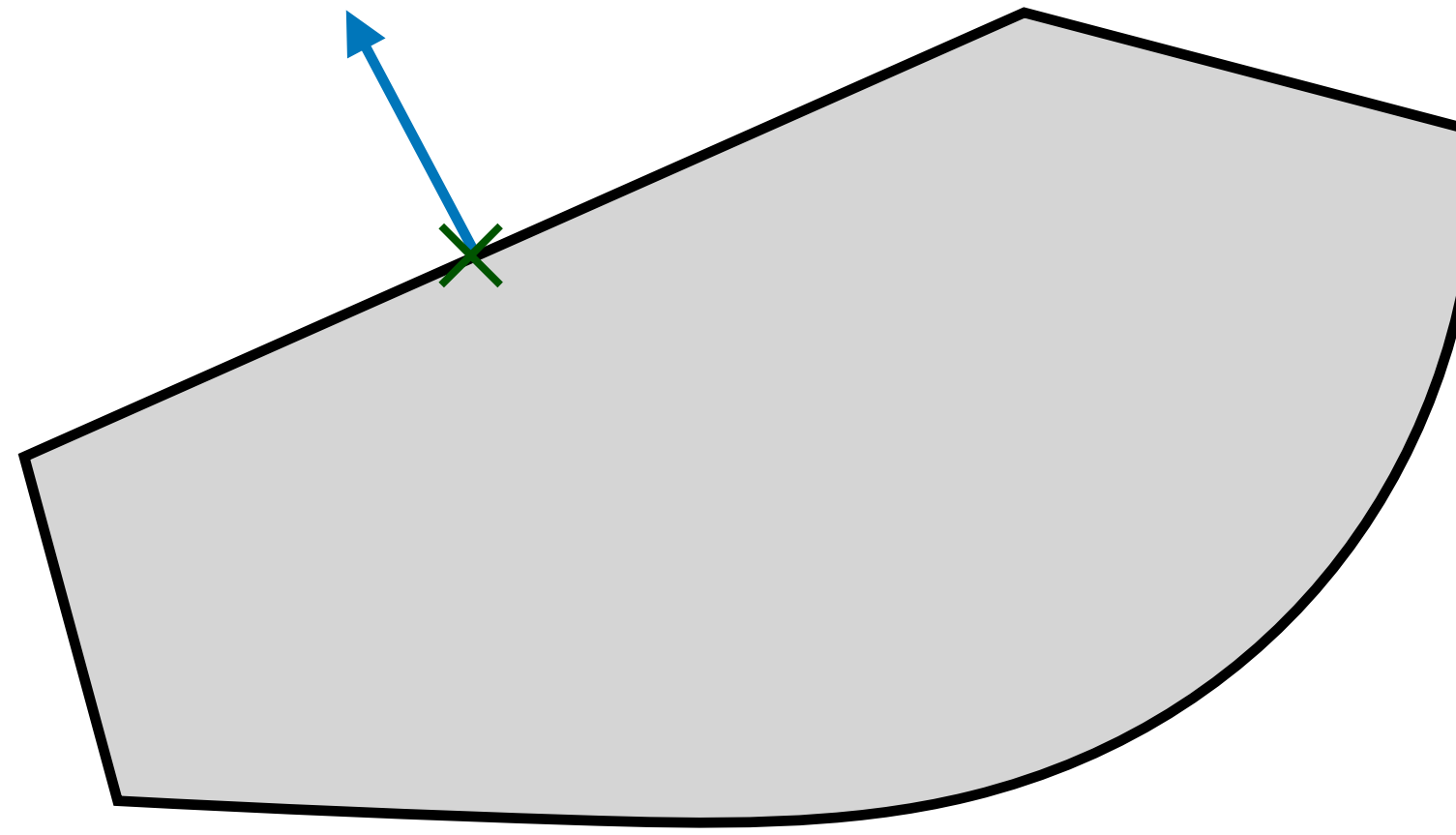
- Interpretation: $a \nabla g(x, y)$, $b \nabla h(x, y)$, $-\nabla f$ are forces acting on (x, y) — at optimum, forces are in equilibrium

One-sided (inequality) constraints



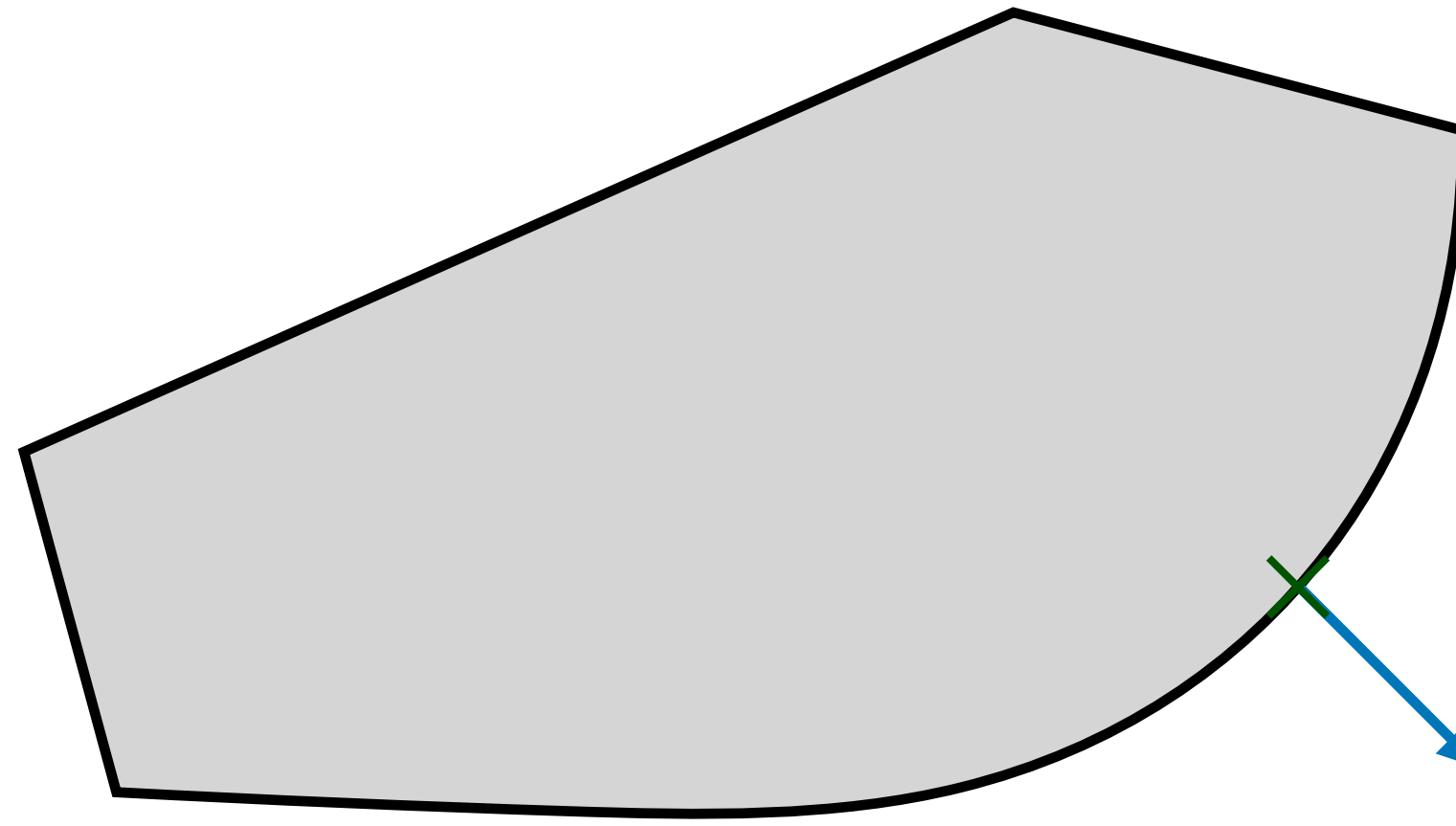
- Constrain point x to lie in a convex set: use a Lagrange multiplier that is a multiple of the **normal** to the set at x
 - ▶ interpret as a force pushing x back into the set
 - ▶ only negative or zero multiples allowed: only push inward
- At a corner, there is a **normal cone**, and any vector in that cone is a legal value for $-\text{multiplier}$
- Normal forces for all constraints must cancel at equilibrium

One-sided (inequality) constraints



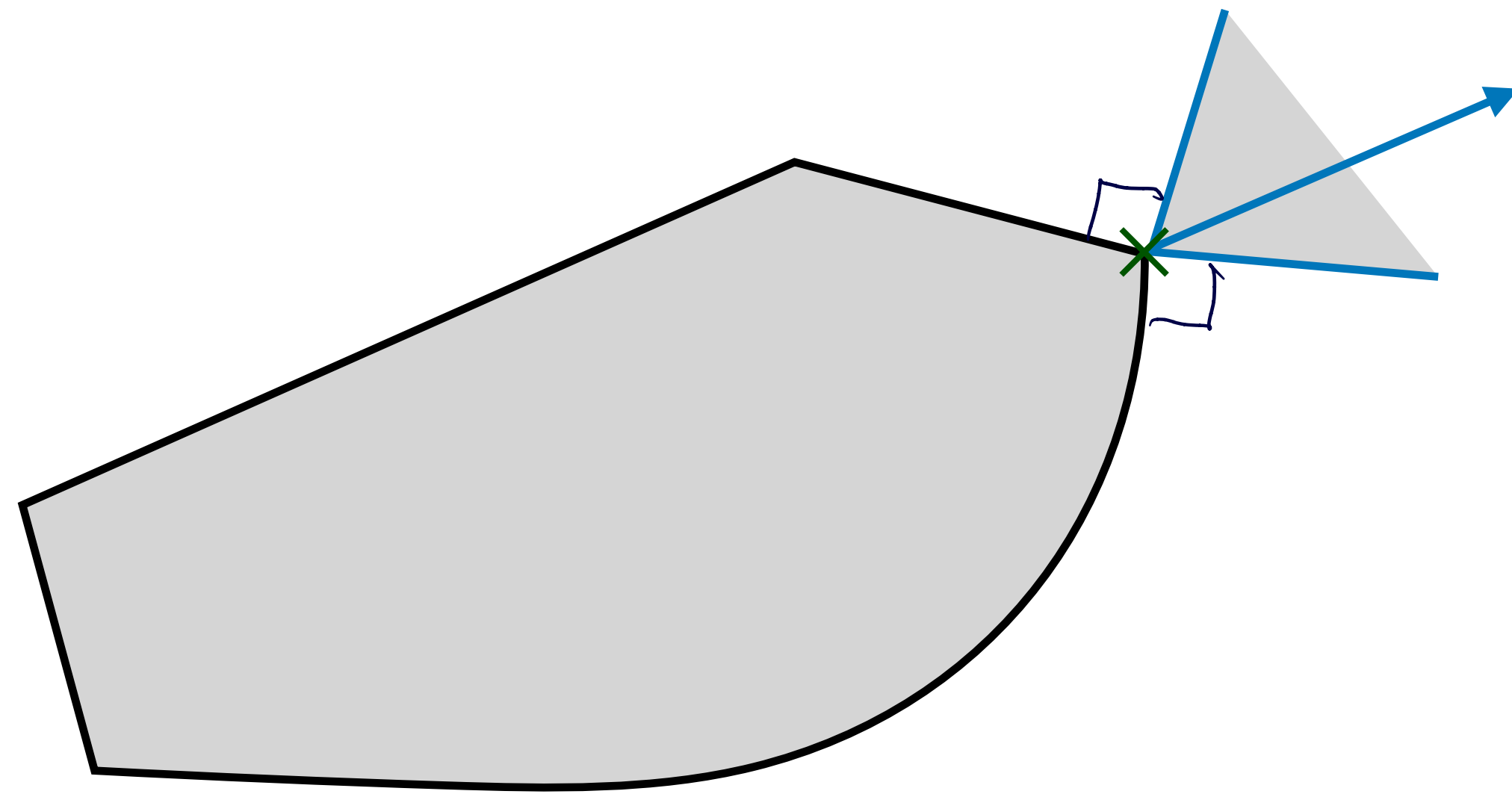
- Constrain point x to lie in a convex set: use a Lagrange multiplier that is a multiple of the ***normal*** to the set at x
 - ▶ interpret as a force pushing x back into the set
 - ▶ only negative or zero multiples allowed: only push inward
- At a corner, there is a ***normal cone***, and any vector in that cone is a legal value for $-\text{multiplier}$
- Normal forces for all constraints must cancel at equilibrium

One-sided (inequality) constraints



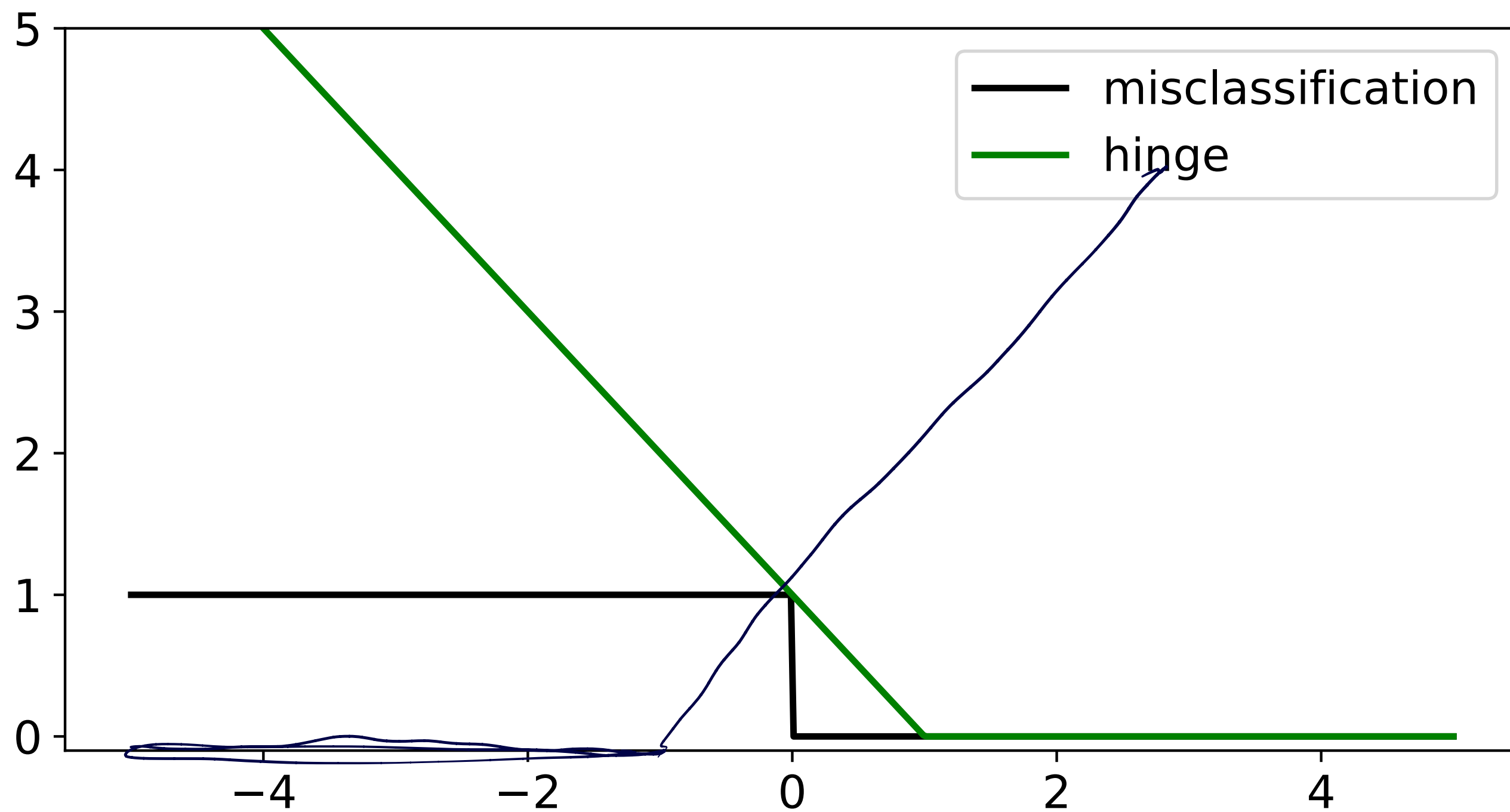
- Constrain point x to lie in a convex set: use a Lagrange multiplier that is a multiple of the **normal** to the set at x
 - ▶ interpret as a force pushing x back into the set
 - ▶ only negative or zero multiples allowed: only push inward
- At a corner, there is a **normal cone**, and any vector in that cone is a legal value for $-\text{multiplier}$
- Normal forces for all constraints must cancel at equilibrium

One-sided (inequality) constraints



- Constrain point x to lie in a convex set: use a Lagrange multiplier that is a multiple of the **normal** to the set at x
 - ▶ interpret as a force pushing x back into the set
 - ▶ only negative or zero multiples allowed: only push inward
- At a corner, there is a **normal cone**, and any vector in that cone is a legal value for $-\text{multiplier}$
- Normal forces for all constraints must cancel at equilibrium

Example: classifier

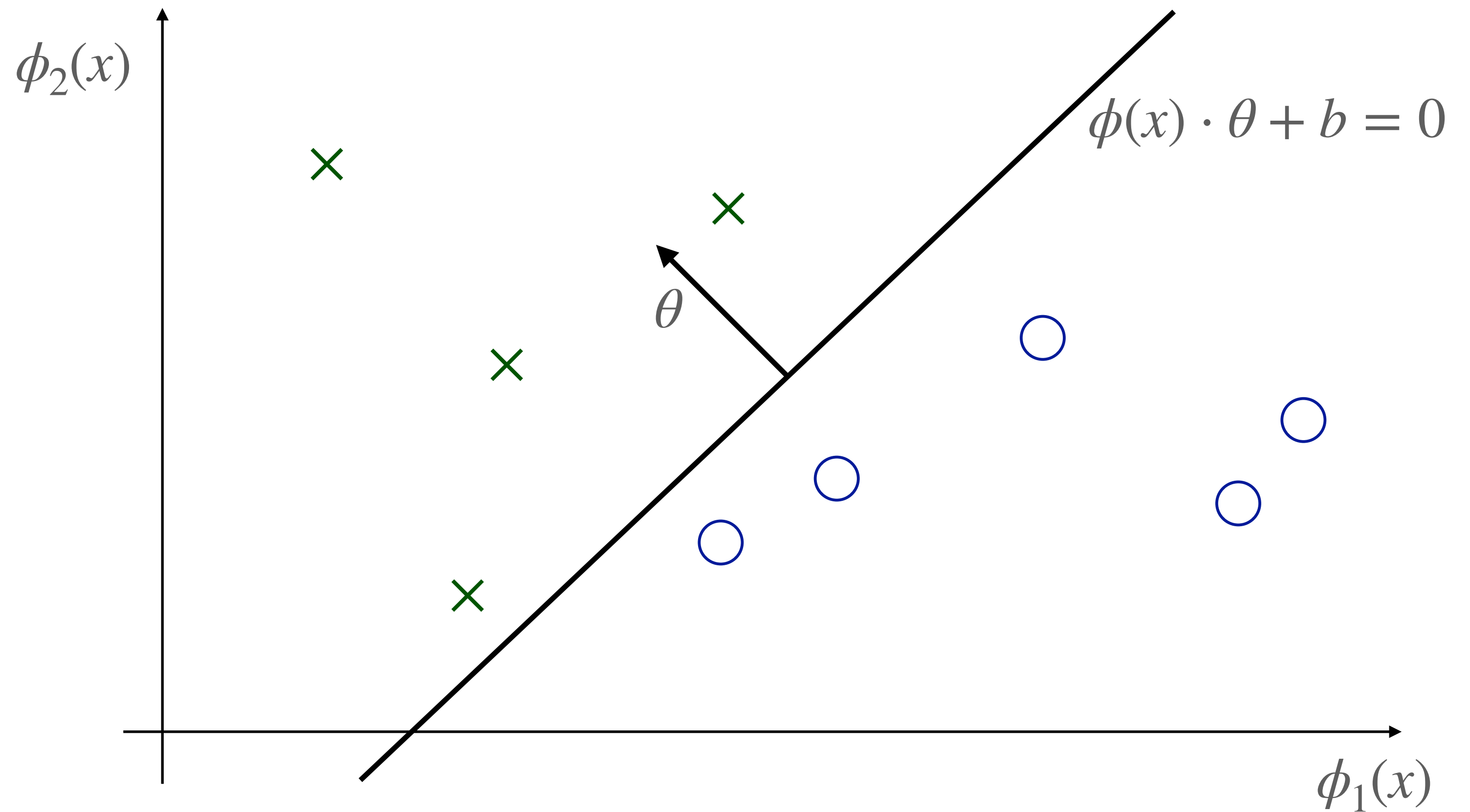


$$\min_{f \in \mathcal{H}} L(f) = \sum_{i=1}^N \ell(y^{(i)} f(x^{(i)}))$$

- ▶ choose hinge loss as bound: $\ell(z) = \max(0, 1 - z)$
- ▶ minimizer wants $f(x^{(i)}) \geq 1$ for +ve examples,
 $f(x^{(i)}) \leq -1$ for -ve examples

Linear classifier

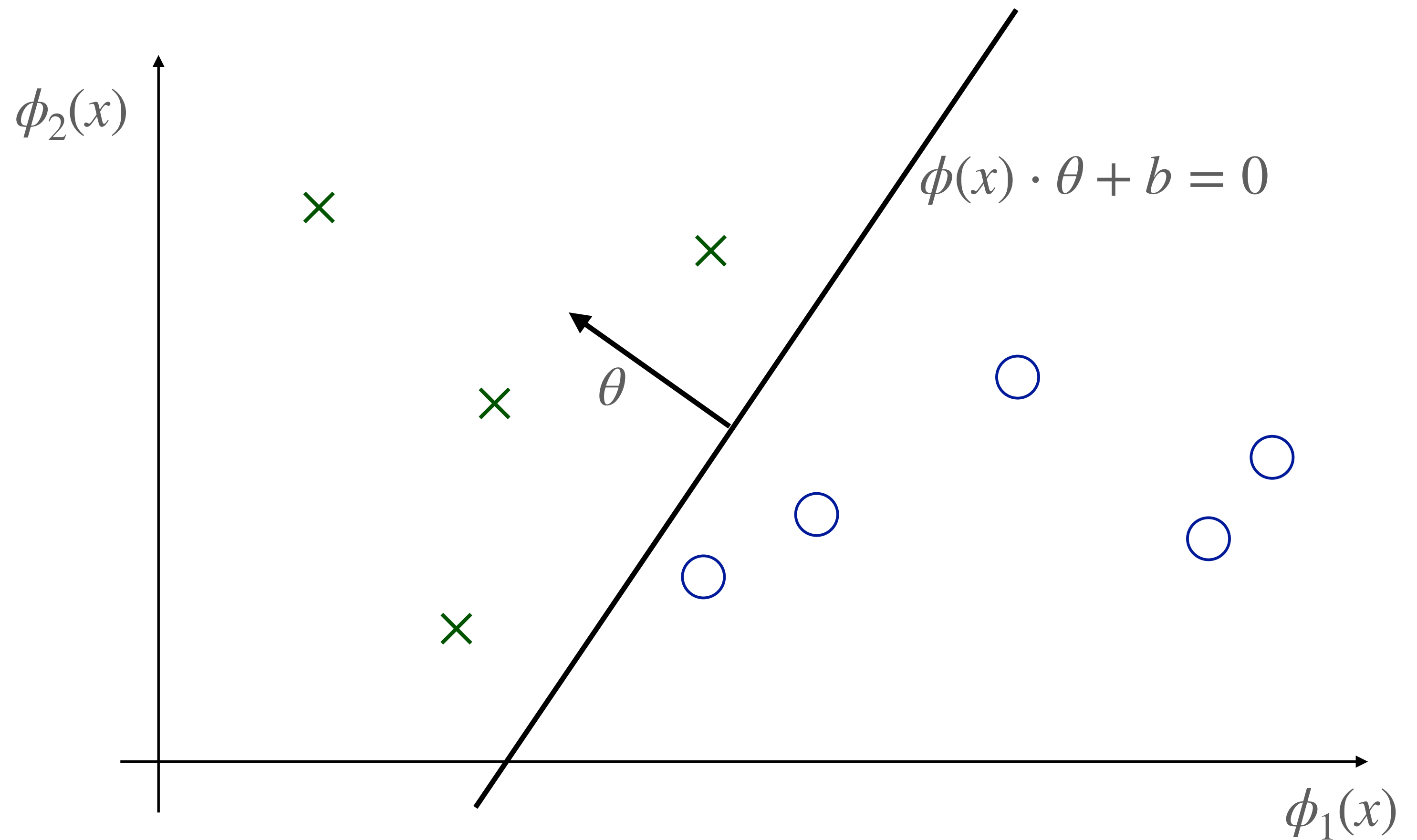
unlike last lecture, we'll focus on linear f for today



- Linear $f(x) = \phi(x) \cdot \theta + b$
 - ▶ note intercept b (not marked in picture)
- Decision boundary is where $\phi(x) \cdot \theta + b = 0$

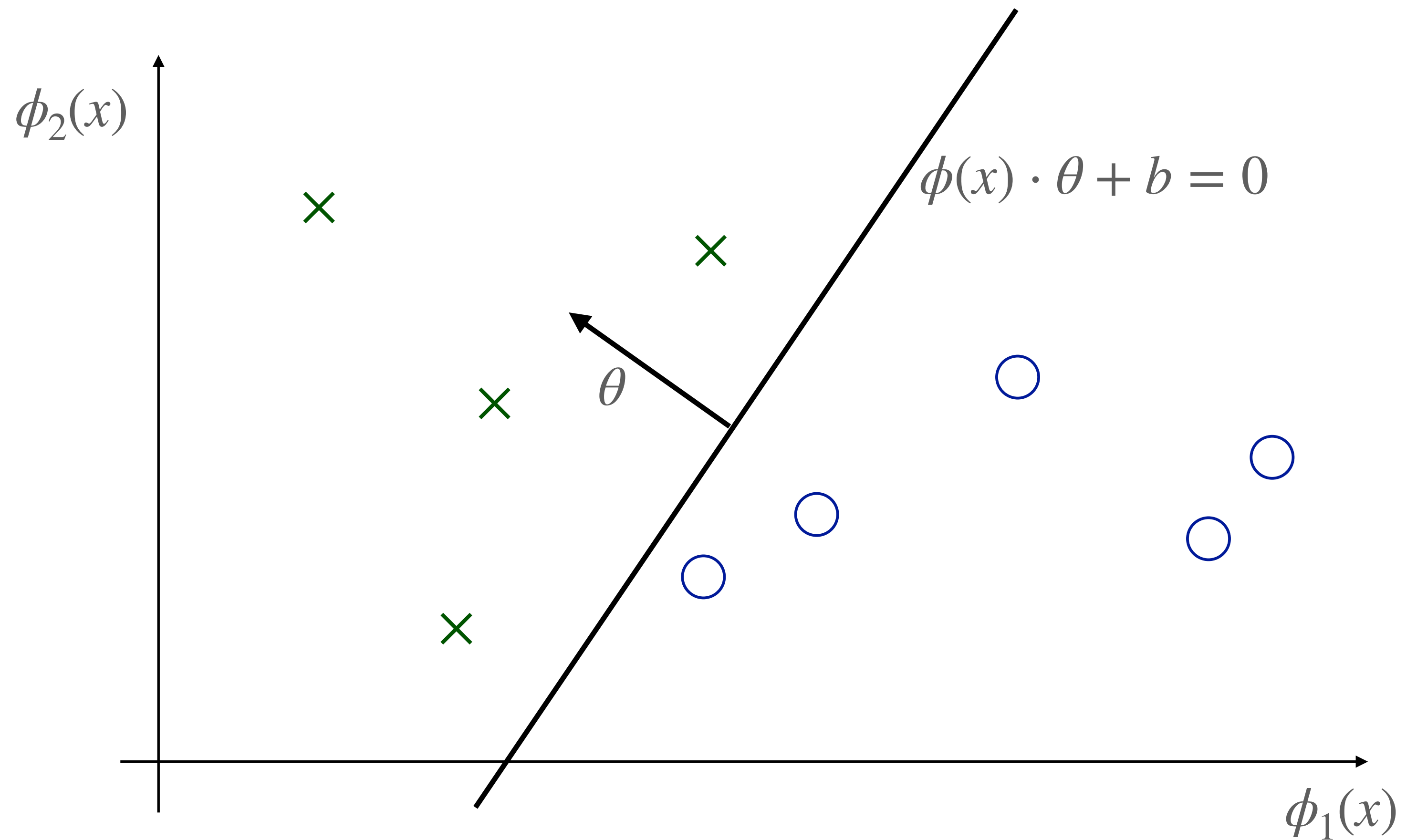
Linear classifier

unlike last lecture, we'll focus on linear f for today



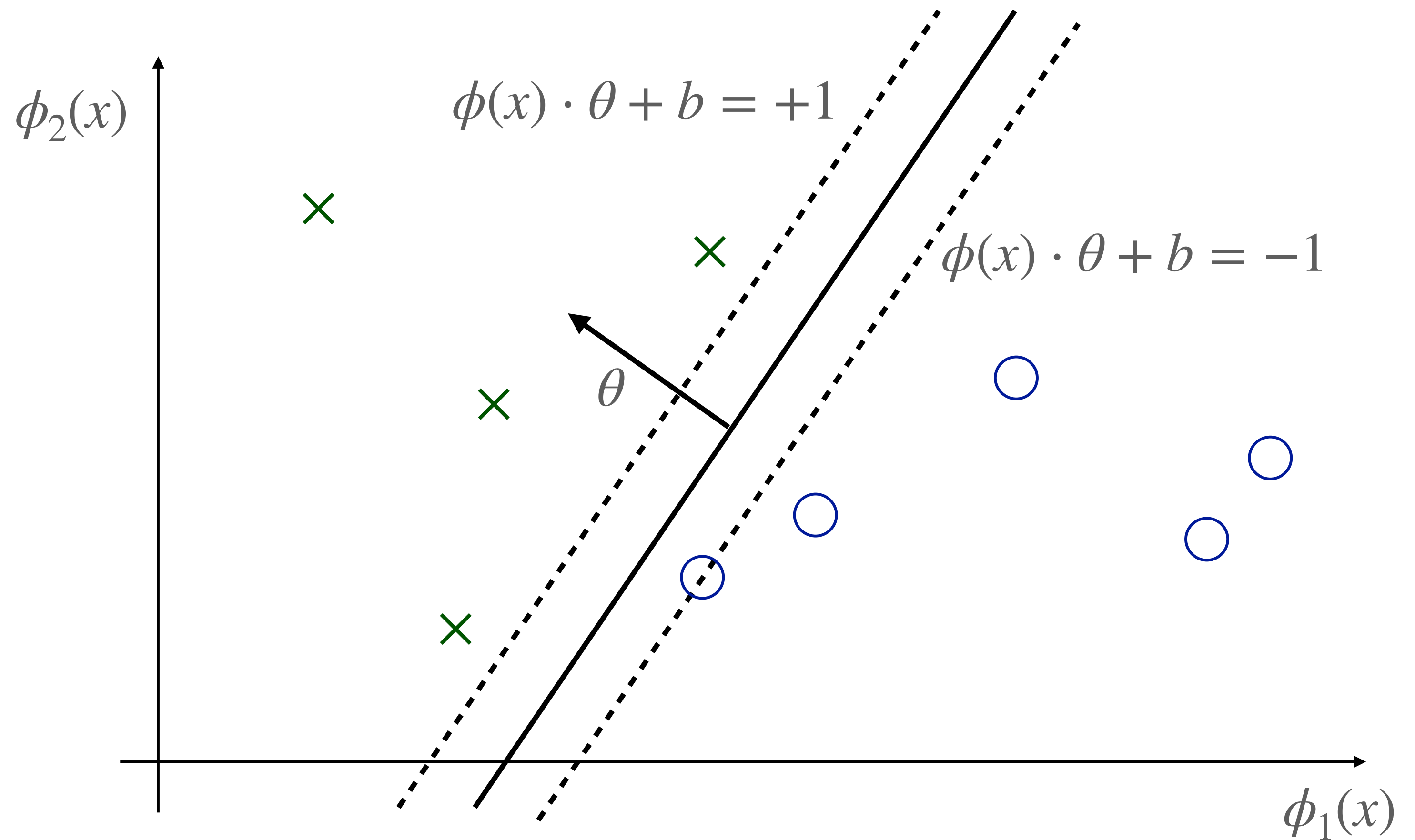
- Linear $f(x) = \phi(x) \cdot \theta + b$
 - ▶ note intercept b (not marked in picture)
- Decision boundary is where $\phi(x) \cdot \theta + b = 0$

Margin



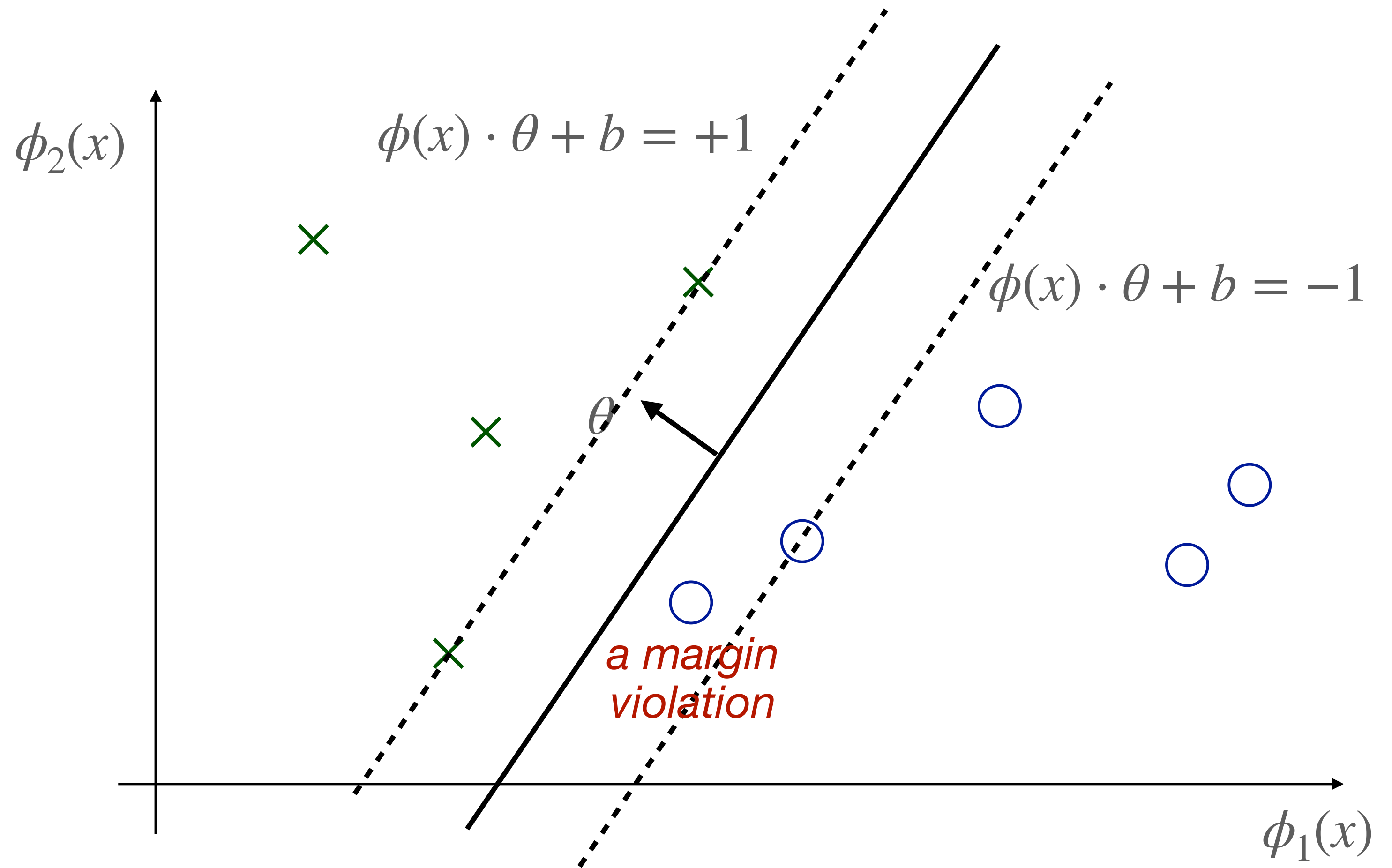
- Hinge loss starts increasing when $\phi(x) \cdot \theta + b = 1$ (positive examples) or -1 (negative examples)
 - ▶ boundaries are called *margin*s
- Scaling θ, b changes location of margins but not boundary

Margin



- Hinge loss starts increasing when $\phi(x) \cdot \theta + b = 1$ (positive examples) or -1 (negative examples)
 - ▶ boundaries are called *margin*s
- Scaling θ, b changes location of margins but not boundary

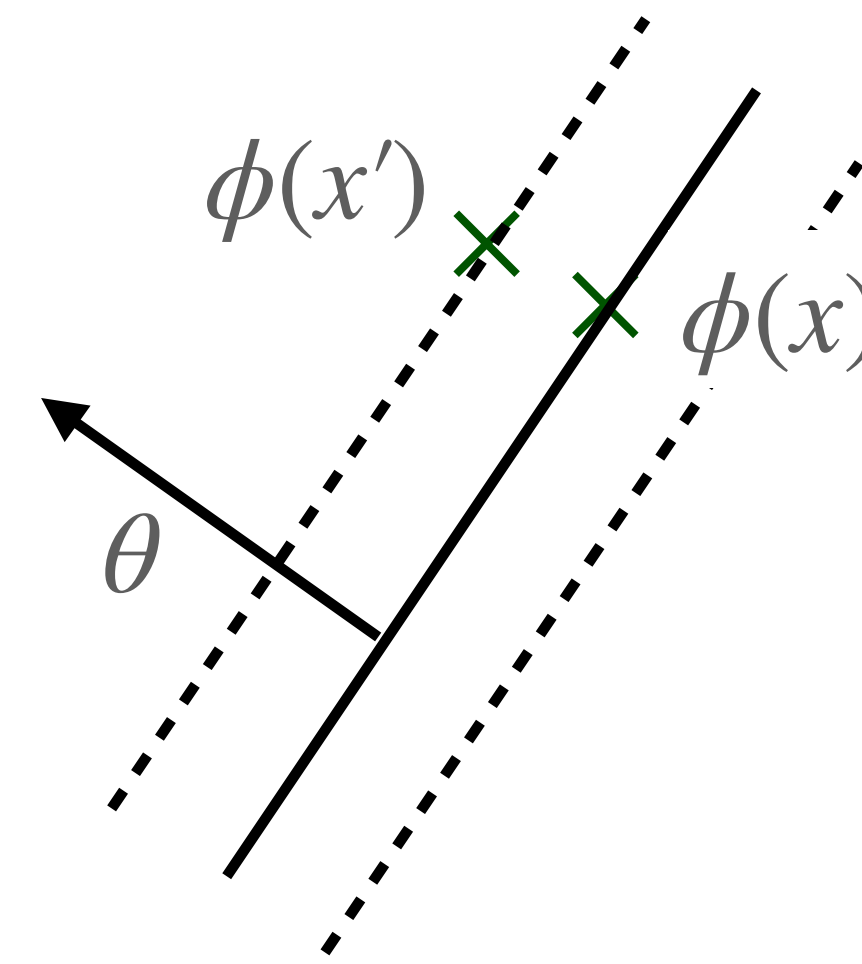
Margin



- Hinge loss starts increasing when $\phi(x) \cdot \theta + b = 1$ (positive examples) or -1 (negative examples)
 - ▶ boundaries are called *margins*
- Scaling θ, b changes location of margins but not boundary

Distance to margin

confusingly, *margin* can mean both the dashed line itself and the distance between the dashed line and decision boundary



- We can calculate distance from the decision boundary to the margin: an example with $\phi(x) \cdot \theta + b = 0$ is on the boundary, and one with $\phi(x') \cdot \theta + b = 1$ is on the margin. Subtracting, $(\phi(x') - \phi(x)) \cdot \theta = 1$. Shortest distance is if $\phi(x') - \phi(x)$ is parallel to θ : i.e., equal to $a\theta$ for some $a \in \mathbb{R}$. Substituting, $a\theta \cdot \theta = 1$. Solving,

$$a = \frac{1}{\|\theta\|^2} \text{ and the distance is } \|a\theta\| = \frac{1}{\|\theta\|}$$

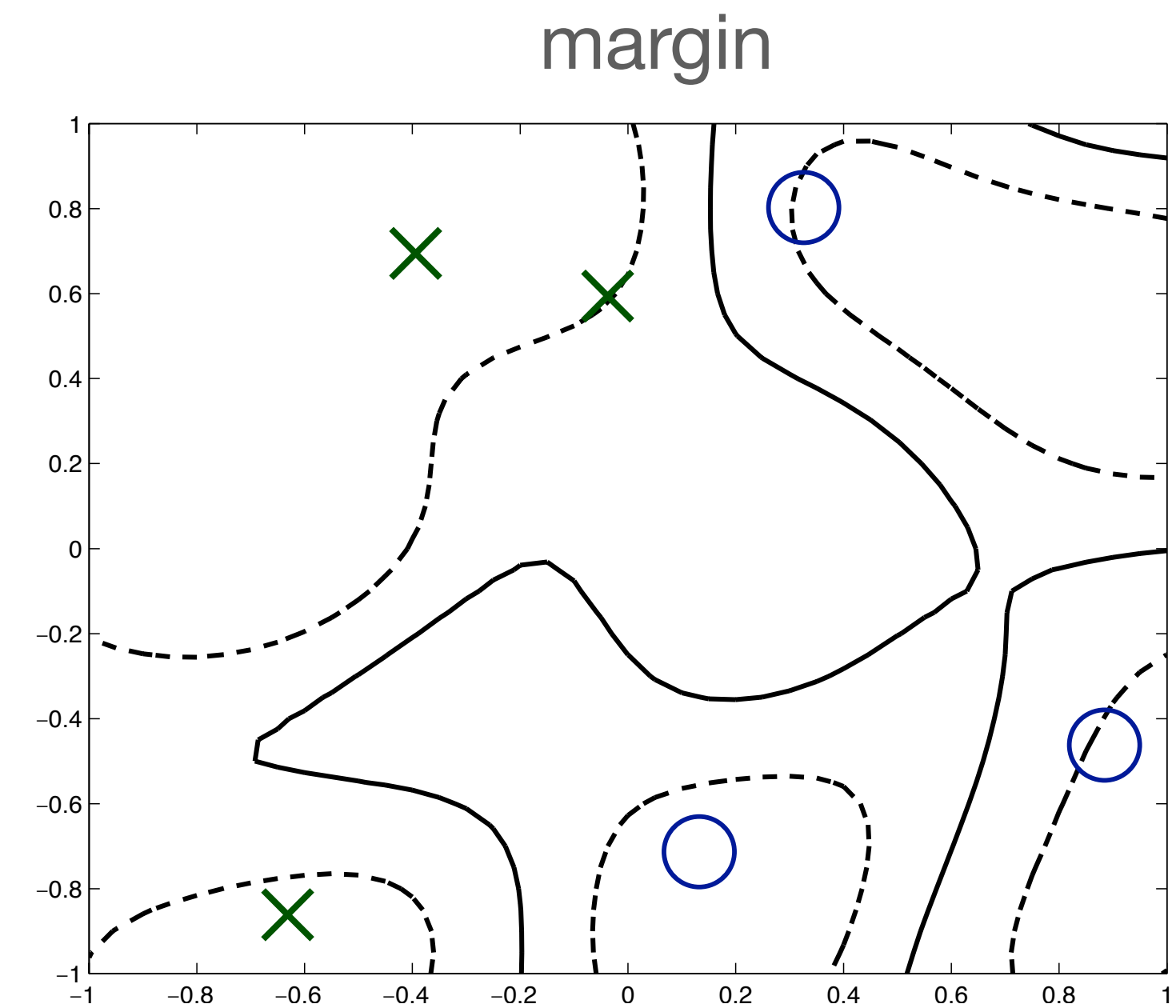
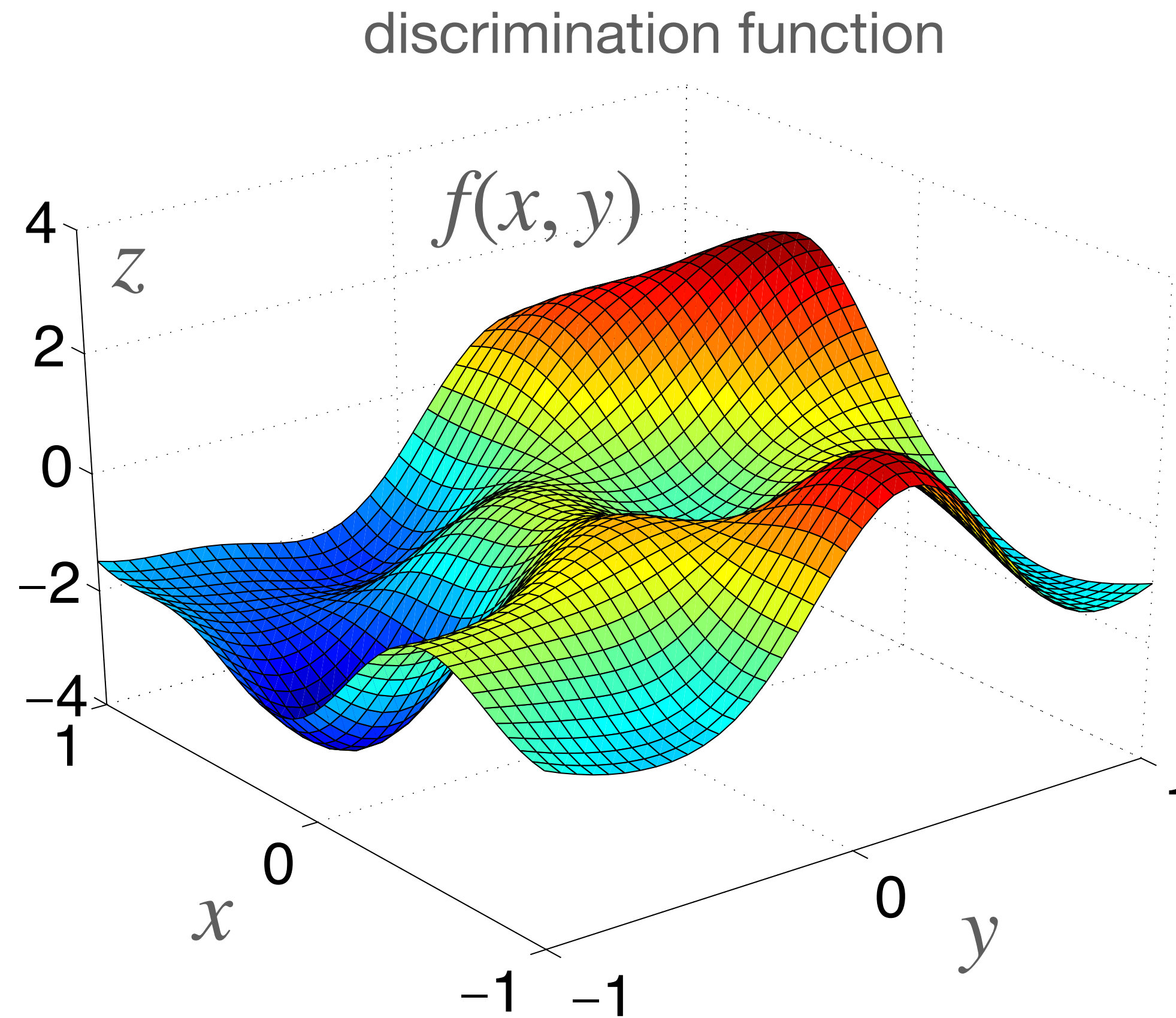
SVM

- For a linearly separable dataset, without regularization, we'll just scale up θ so that all points are outside margin
- To prevent this, regularize:

$$\min_{\theta, b} C \sum_{i=1}^N \ell(y^{(i)}(\phi(x^{(i)}) \cdot \theta + b)) + \frac{1}{2} \|\theta\|^2$$

- ▶ make θ small (distance to margin $\frac{1}{\|\theta\|}$ large) while still getting low total hinge loss ℓ
- Constant $C > 0$ weights loss vs. regularizer
 - ▶ large C prioritizes ℓ : larger θ , fewer margin violations
 - ▶ C near 0 prioritizes regularizer: smaller θ at cost of more margin violations
- This is a (linear) **support vector machine** or **SVM**

Nonlinear SVM

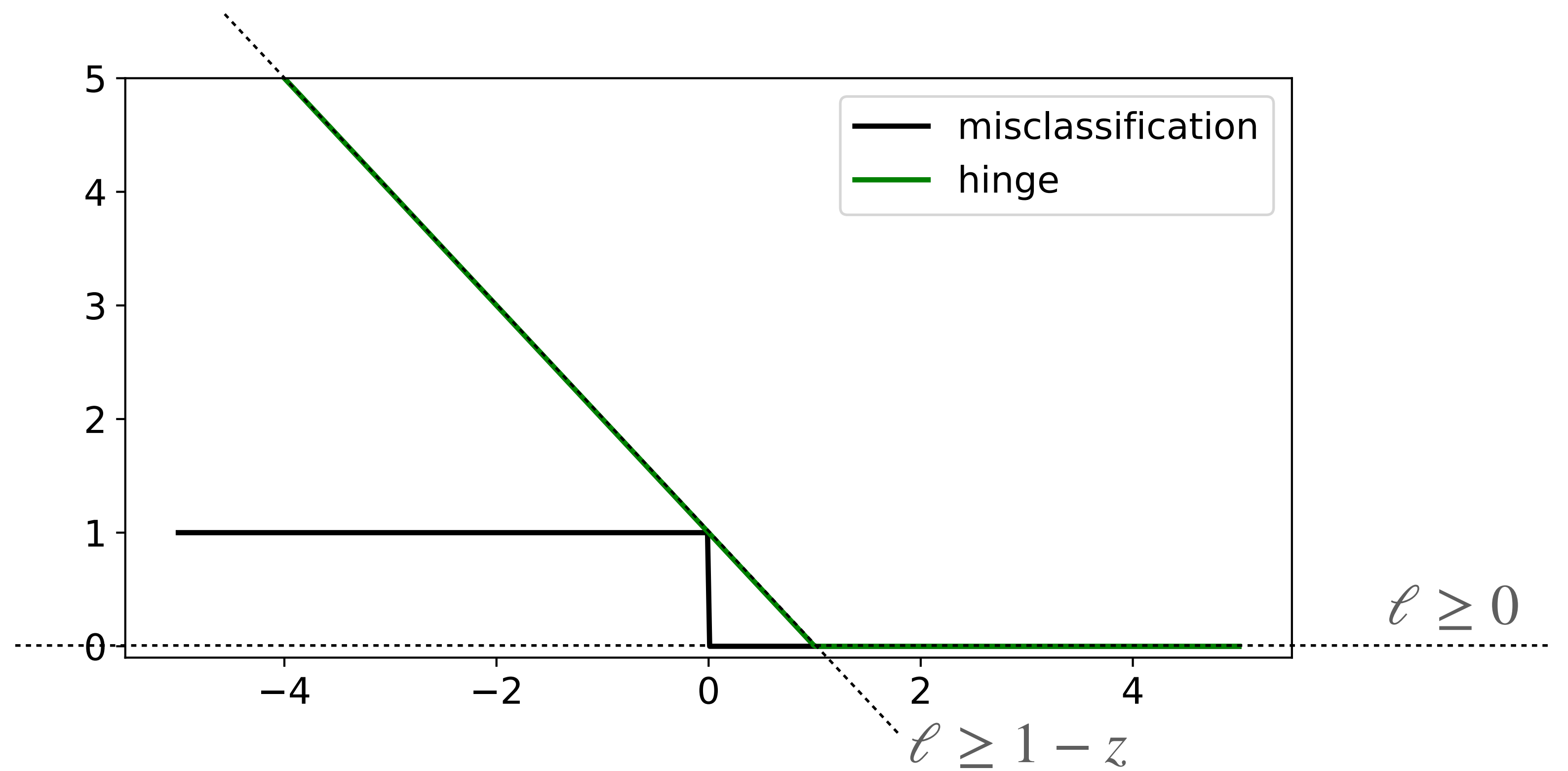


- We can also define a nonlinear SVM (more on that soon!)

Why maximize distance to margin?

- It's another way of limiting the complexity of our hypothesis class
- Just like
 - ▶ Haussler bound limits number of distinct hypotheses
 - ▶ VC bound limits number of hypotheses that are distinct *on our given set of inputs $x^{(i)}$*
- We can make a margin-based bound: there aren't that many ways we can fit a fat margin in between datapoints
 - ▶ wrinkle: margin bound needs to condition on our dataset
 - ▶ i.e., we can't say ahead of time that an SVM will generalize; but if we optimize and observe that we got a large margin, we can be confident of good generalization

Solving the linear SVM



- Objective: $\min_{\theta, b} C \sum_{i=1}^N \ell(y^{(i)}(\phi(x^{(i)}) \cdot \theta + b)) + \frac{1}{2} \|\theta\|^2$
- Rewrite ℓ : we know $\ell \geq 0$ and $\ell \geq 1 - z$
 - ▶ and since we're minimizing ℓ , we can just enforce these constraints, and the optimizer will make one of them tight

Quadratic program for SVM

$$\begin{aligned} \min_{\theta, b, \ell_i} \quad & C \sum_{i=1}^N \ell_i + \frac{1}{2} \|\theta\|^2 \quad \text{s.t.} \\ (\forall i) \quad & \ell_i \geq 0 \\ (\forall i) \quad & \ell_i \geq 1 - y^{(i)}(\phi(x^{(i)}) \cdot \theta + b) \end{aligned}$$

- We've eliminated the nonlinear hinge loss function $\ell(z)$, replacing it by extra variables ℓ_i ($i = 1 \dots N$) and linear inequality constraints
- Minimizing a quadratic objective under linear constraints is a *quadratic program*; this one is a *convex* QP because the objective is convex for minimization
 - ▶ in fact there's a unique globally optimal θ
 - ▶ and there are fast algorithms for finding it, if dimension of θ and $\phi(x^{(i)})$ is not too high (QP size \propto # dims $\times N$)

Quadratic program for SVM

pay a penalty for margin violations \swarrow \nwarrow maximize distance to margin

$$\min_{\theta, b, \ell_i} C \sum_{i=1}^N \ell_i + \frac{1}{2} \|\theta\|^2 \text{ s.t.}$$

$$(\forall i) \ell_i \geq 0 \quad \leftarrow \text{don't worry if we're on good side of margin}$$

$$(\forall i) \ell_i \geq 1 - y^{(i)}(\phi(x^{(i)}) \cdot \theta + b) \quad \leftarrow \text{if not, measure margin violation}$$

- We've eliminated the nonlinear hinge loss function $\ell(z)$, replacing it by extra variables ℓ_i ($i = 1 \dots N$) and linear inequality constraints
- Minimizing a quadratic objective under linear constraints is a *quadratic program*; this one is a *convex* QP because the objective is convex for minimization
 - ▶ in fact there's a unique globally optimal θ
 - ▶ and there are fast algorithms for finding it, if dimension of θ and $\phi(x^{(i)})$ is not too high (QP size \propto # dims $\times N$)

Reformulate the SVM

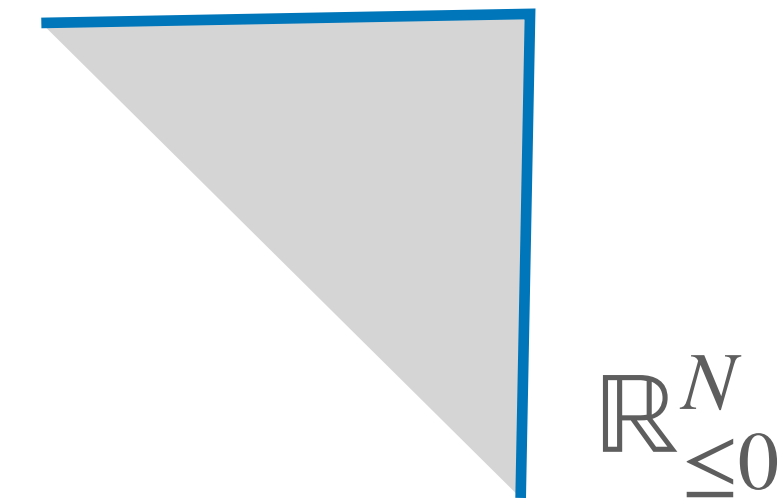
- What if dimension of $\phi(x)$ is too high?
- We can use adversarial learning to reformulate the SVM
 - ▶ the result is called the **dual** of the original (primal) SVM
- The size of the QP will become **independent** of the dimension of $\phi(x)$
 - ▶ but unfortunately will be proportional to N^2 not N
 - ▶ a win if $N \ll \#$ dims of ϕ , and further shrinking is possible via approximations
 - ▶ in a future lecture, we'll even use functional ML to handle infinite-dimensional features $\phi(x^{(i)})$

Dual SVM

$$\min_{\theta, b, \ell_i} C \sum_{i=1}^N \ell_i + \frac{1}{2} \|\theta\|^2 \text{ s.t.}$$

$$(\forall i) \ell_i \geq 0$$

$$(\forall i) \ell_i \geq 1 - y^{(i)}(\phi(x^{(i)}) \cdot \theta + b)$$



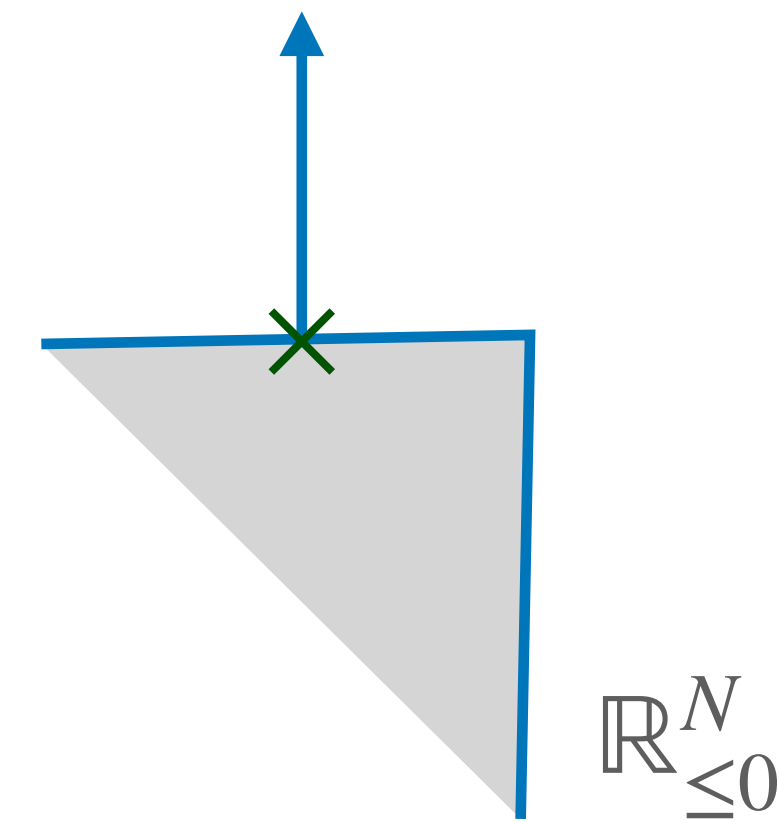
- Last constraint is the annoying one:
 - ▶ [complicated function]_i ≤ 0 for all i
 - ▶ i.e., [complicated function] has to be in $\mathbb{R}_{\le 0}^N$
 - ▶ introduce an adversary α that takes values in the normal cone of $\mathbb{R}_{\le 0}^N$
 - ▶ α is also called a Lagrange multiplier or a dual variable

Dual SVM

$$\min_{\theta, b, \ell_i} C \sum_{i=1}^N \ell_i + \frac{1}{2} \|\theta\|^2 \text{ s.t.}$$

$$(\forall i) \ell_i \geq 0$$

$$(\forall i) \ell_i \geq 1 - y^{(i)}(\phi(x^{(i)}) \cdot \theta + b)$$



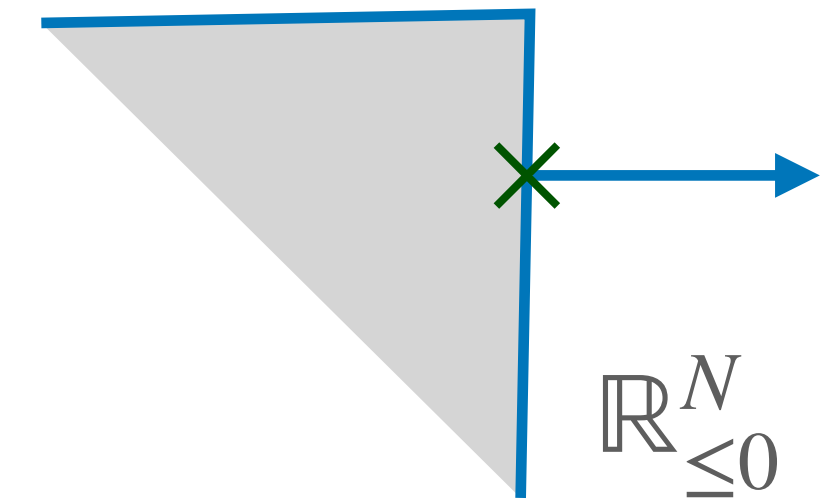
- Last constraint is the annoying one:
 - ▶ [complicated function] $_i \leq 0$ for all i
 - ▶ i.e., [complicated function] has to be in $\mathbb{R}_{\leq 0}^N$
 - ▶ introduce an adversary α that takes values in the normal cone of $\mathbb{R}_{\leq 0}^N$
 - ▶ α is also called a Lagrange multiplier or a dual variable

Dual SVM

$$\min_{\theta, b, \ell_i} C \sum_{i=1}^N \ell_i + \frac{1}{2} \|\theta\|^2 \text{ s.t.}$$

$$(\forall i) \ell_i \geq 0$$

$$(\forall i) \ell_i \geq 1 - y^{(i)}(\phi(x^{(i)}) \cdot \theta + b)$$



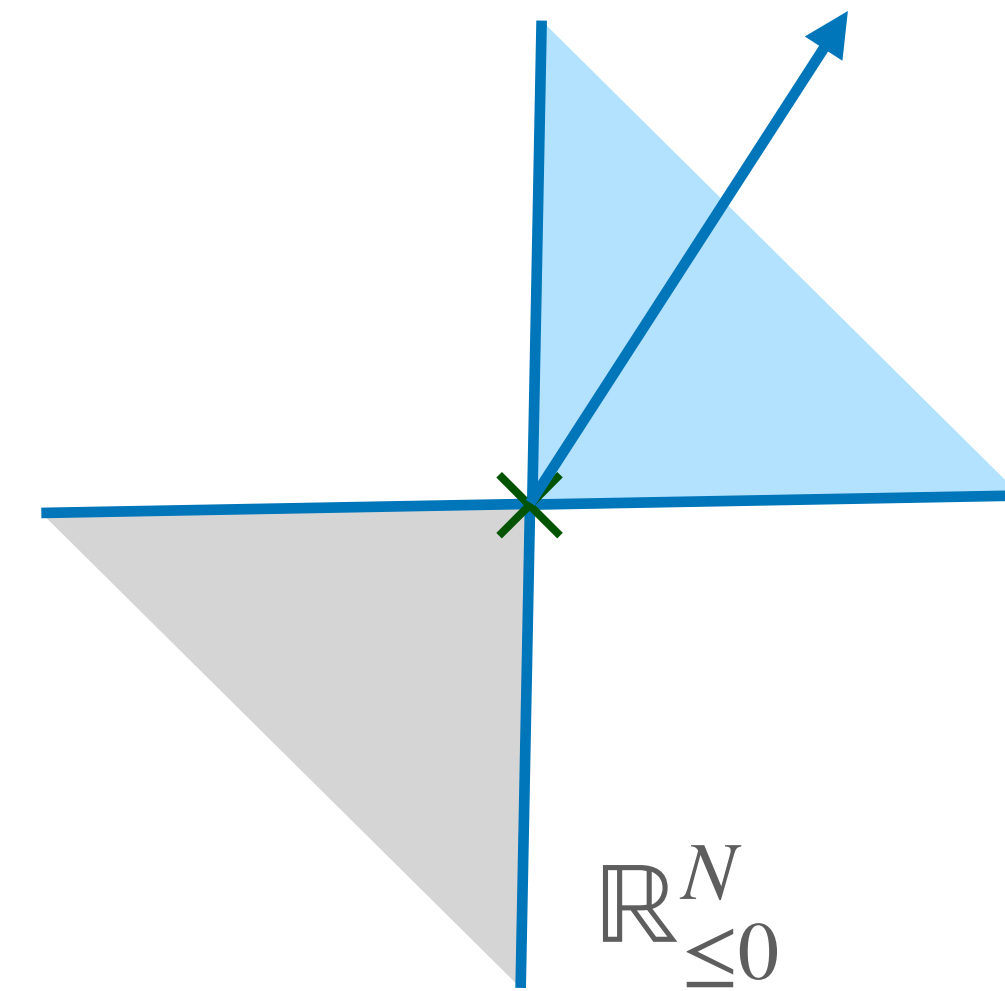
- Last constraint is the annoying one:
 - ▶ [complicated function]_i ≤ 0 for all i
 - ▶ i.e., [complicated function] has to be in $\mathbb{R}_{\leq 0}^N$
 - ▶ introduce an adversary α that takes values in the normal cone of $\mathbb{R}_{\leq 0}^N$
 - ▶ α is also called a Lagrange multiplier or a dual variable

Dual SVM

$$\min_{\theta, b, \ell_i} C \sum_{i=1}^N \ell_i + \frac{1}{2} \|\theta\|^2 \text{ s.t.}$$

$$(\forall i) \ell_i \geq 0$$

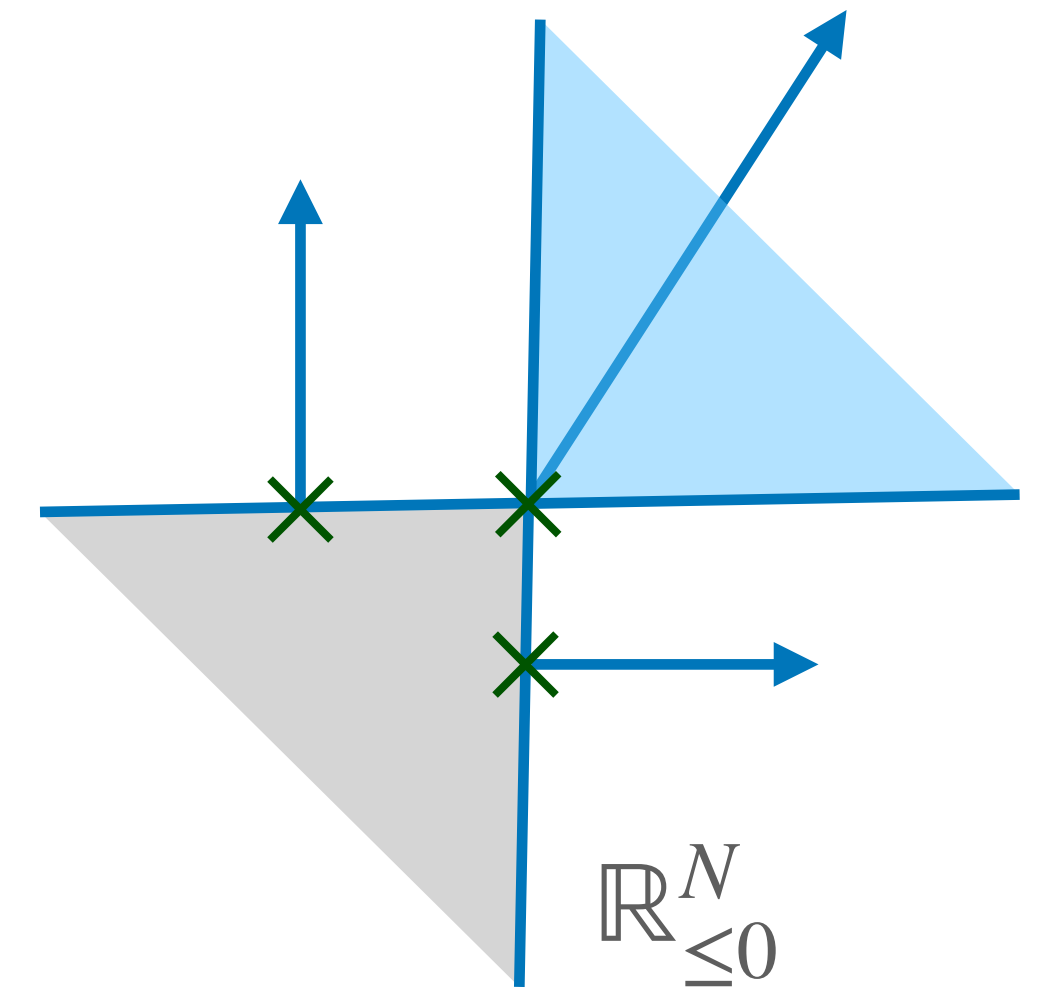
$$(\forall i) \ell_i \geq 1 - y^{(i)}(\phi(x^{(i)}) \cdot \theta + b)$$



- Last constraint is the annoying one:
 - ▶ [complicated function] $_i \leq 0$ for all i
 - ▶ i.e., [complicated function] has to be in $\mathbb{R}_{\leq 0}^N$
 - ▶ introduce an adversary α that takes values in the normal cone of $\mathbb{R}_{\leq 0}^N$
 - ▶ α is also called a Lagrange multiplier or a dual variable

Complementary variables

- Normal cone takes different shapes depending on
$$s_i = 1 - y^{(i)}(\phi(x^{(i)}) \cdot \theta + b) - \ell_i$$
 - ▶ it's always a subset of $\mathbb{R}_{\geq 0}^N$
 - ▶ if $s_i < 0$ then we're not on the boundary in dimension i , and α_i must be 0
 - ▶ and if $\alpha_i > 0$ it means we must be on the boundary in dimension i
- Write constraints compactly:
 - ▶ $\alpha \geq 0$ componentwise
 - ▶ $s \leq 0$ componentwise
 - ▶ each $\alpha_i s_i = 0$: at most one of α_i or s_i can be nonzero
 - ▶ equivalently $\alpha^\top s = 0$: since each $\alpha_i s_i$ term is ≤ 0 , sum can only be zero if all are zero



called **complementarity**

Deriving the dual

- Original (primal) SVM:

$$\min_{\theta, b, \ell_i} C \sum_{i=1}^N \ell_i + \frac{1}{2} \|\theta\|^2$$

$$\text{with } \ell_i \geq 0 \text{ and } s_i = 1 - y^{(i)}(\phi(x^{(i)}) \cdot \theta + b) - \ell_i \leq 0$$

- Dual variable $\alpha_i \geq 0$, want $\alpha^\top s = 0$

- ▶ we know $\alpha^\top s \leq 0$ since $\alpha \geq 0$ and $s \leq 0$

- ▶ in fact it's enough to maximize $\alpha^\top s$

- Substitute in:

- ▶ $\min_{\theta, b, \ell_i} \max_{\alpha_i} C \sum_{i=1}^N \ell_i + \frac{1}{2} \|\theta\|^2 + \sum_{i=1}^N \alpha_i s_i$

- ▶ with same constraints as above

Deriving the dual

- Copying the problem:

$$\min_{\theta, b, \ell_i} \max_{\alpha_i} C \sum_{i=1}^N \ell_i + \frac{1}{2} \|\theta\|^2 + \sum_{i=1}^N \alpha_i s_i$$

$$\text{with } \ell_i, \alpha_i \geq 0, s_i = 1 - y^{(i)}(\phi(x^{(i)}) \cdot \theta + b) - \ell_i \leq 0$$

- Take gradients wrt θ, b, ℓ_i

▶ ∇_b : $-\sum_i \alpha_i y^{(i)}$ ~~$\neq 0$~~ $= 0$ *typo — should be no b*

▶ ∇_{θ} : $\theta = \sum_i y^{(i)} \phi(x^{(i)}) \alpha_i$

▶ ∇_{ℓ_i} : $C - \alpha_i \geq 0$

- Substitute in:

Dual SVM

$$\max_{\alpha_i, \theta} \sum_{i=1}^N \alpha_i - \frac{1}{2} \left\| \sum_{i=1}^N \alpha_i y^{(i)} \phi(x^{(i)}) \right\|^2 \quad \text{s.t.} \quad 0 = \sum_{i=1}^N \alpha_i y^{(i)}$$

where $(\forall i) 0 \leq \alpha_i \leq C$

- Introducing the dual variables α_i leads to a min-max objective (previous slide)
- We can then analytically eliminate θ, b, ℓ_i to get the **dual SVM** quadratic program (above)
- In the process we derive $\theta = \sum_{i=1}^N \alpha_i y^{(i)} \phi(x^{(i)})$, which lets us recover the optimal classifier from the dual α_i s
 - ▶ intercept b is then a 1-d optimization of a piecewise linear objective: similar computation to finding a median

Interpreting the dual variable

- Recall $\alpha_i s_i = 0$ where s_i is constraint RHS for example i
 - ▶ If $\alpha_i > 0$ then constraint must be tight: example i is on the margin (if $\ell_i = 0$) or margin violation (if $\ell_i > 0$)
 - ▶ if example i is strictly on good side of margin (constraint is loose), then $\alpha_i = 0$
 - ▶ so $\alpha_i > 0$ picks out examples that are actively constraining θ : the **support vectors**
 - ▶ bigger α_i means the example is affecting θ more strongly
 - ▶ smaller α_i means the example has less influence
 - ▶ $\alpha_i = 0$ means *no* influence: we could delete all non-support-vector examples and θ wouldn't change
 - ▶ sparsity is common: often # support vectors $\ll N$

SVM ***summary***

- If we train a linear classifier using hinge loss + regularizer, we can optimize via a convex quadratic program
- Adversarial learning (duality) allows us to reformulate:
 - ▶ original (primal) program's size depends on dimensionality of feature vector \times number of examples
 - ▶ dual program's size depends on (number of examples)² and is completely independent of feature dimension
 - ▶ we can solve whichever is smaller and recover best θ, b
- We can interpret both versions:
 - ▶ primal: maximize the margin while avoiding too many margin violations
 - ▶ dual: discover which examples are support vectors (actively constrain decision boundary because they are on the margin or violate it) and recover θ as linear combination of SVs