COS234:

Introduction To Machine Learning

Prof. Yoram Singer

Topic: Stochastic Gradient Decent

Convexity: Q & A

Convexity: Q & A

Q: Can real ML problems be modeled as convex optimization?

Convexity: Q & A

Q: Can real ML problems be modeled as convex optimization?

A: Some but not the majority

Convexity: Q & A

Q: Can real ML problems be modeled as convex optimization?

A: Some but not the majority

Q: Is the notion of convexity useful?

Convexity: Q & A

Q: Can real ML problems be modeled as convex optimization?

A: Some but not the majority

Q: Is the notion of convexity useful?

A: Yes, it is an important building block in non-convex problems

Convexity: Q & A

Q: Can real ML problems be modeled as convex optimization?

A: Some but not the majority

Q: Is the notion of convexity useful?

A: Yes, it is an important building block in non-convex problems

Q: Are complex properties of convex functions needed?

Convexity: Q & A

Q: Can real ML problems be modeled as convex optimization?

A: Some but not the majority

Q: Is the notion of convexity useful?

A: Yes, it is an important building block in non-convex problems

Q: Are complex properties of convex functions needed?

A: Not really unless you are interested in ML algo design & theory

Convexity: Q & A

Q: Can real ML problems be modeled as convex optimization?

A: Some but not the majority

Q: Is the notion of convexity useful?

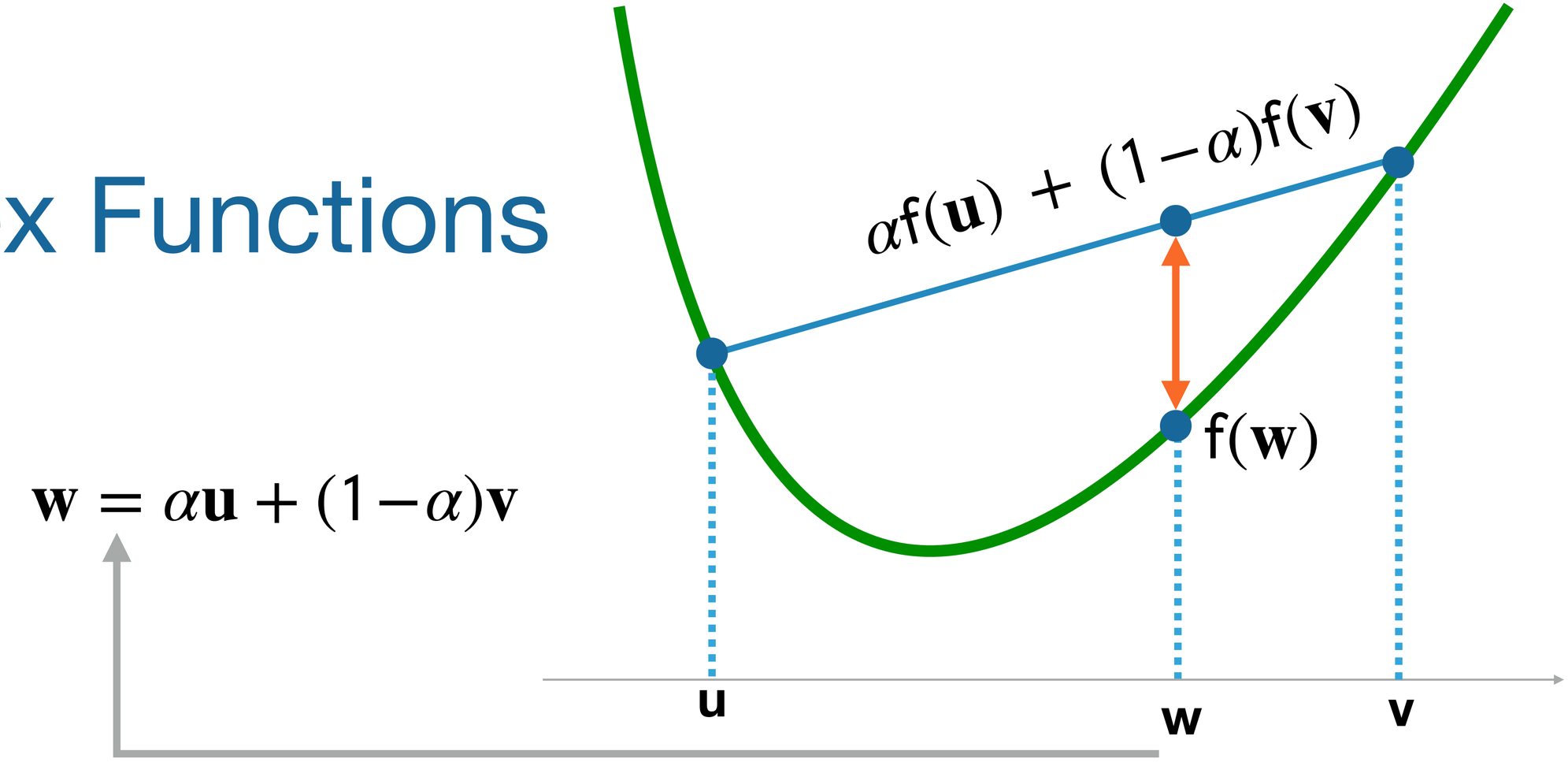
A: Yes, it is an important building block in non-convex problems

Q: Are complex properties of convex functions needed?

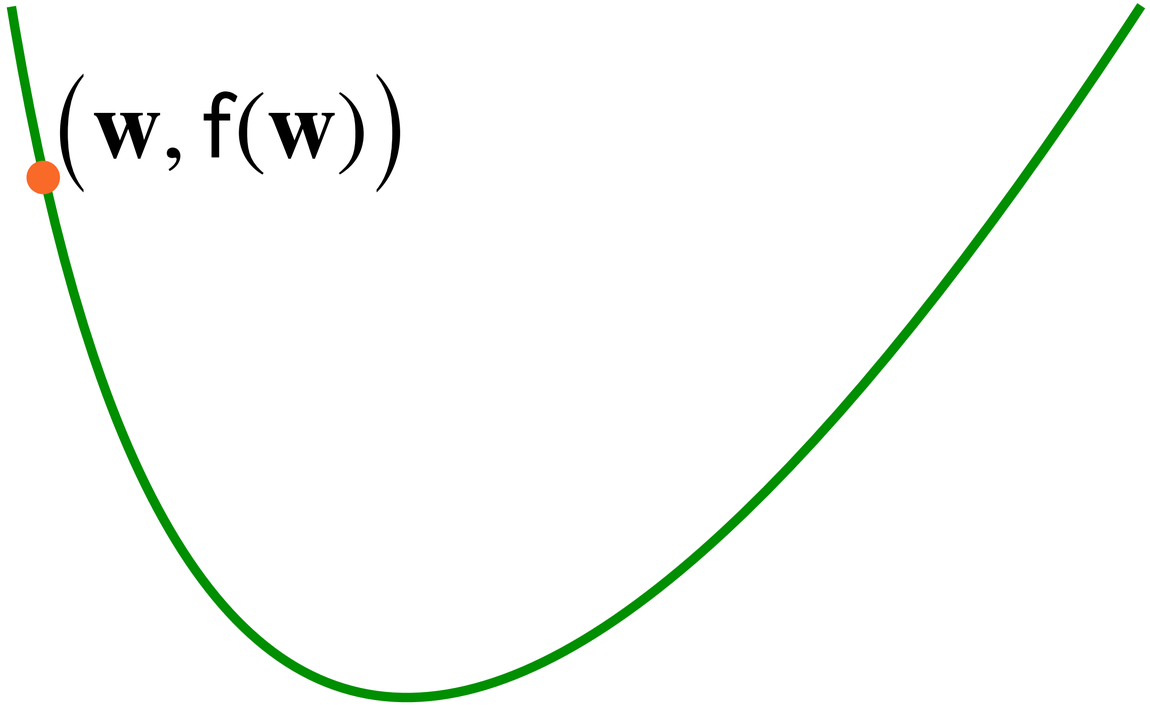
A: Not really unless you are interested in ML algo design & theory

$$\alpha \in [0,1]: \quad \mathrm{f}\big(\alpha\mathbf{u} + (1-\alpha)\mathbf{v}\big) \leq \alpha\mathrm{f}(\mathbf{u}) + (1-\alpha)\mathrm{f}(\mathbf{v})$$

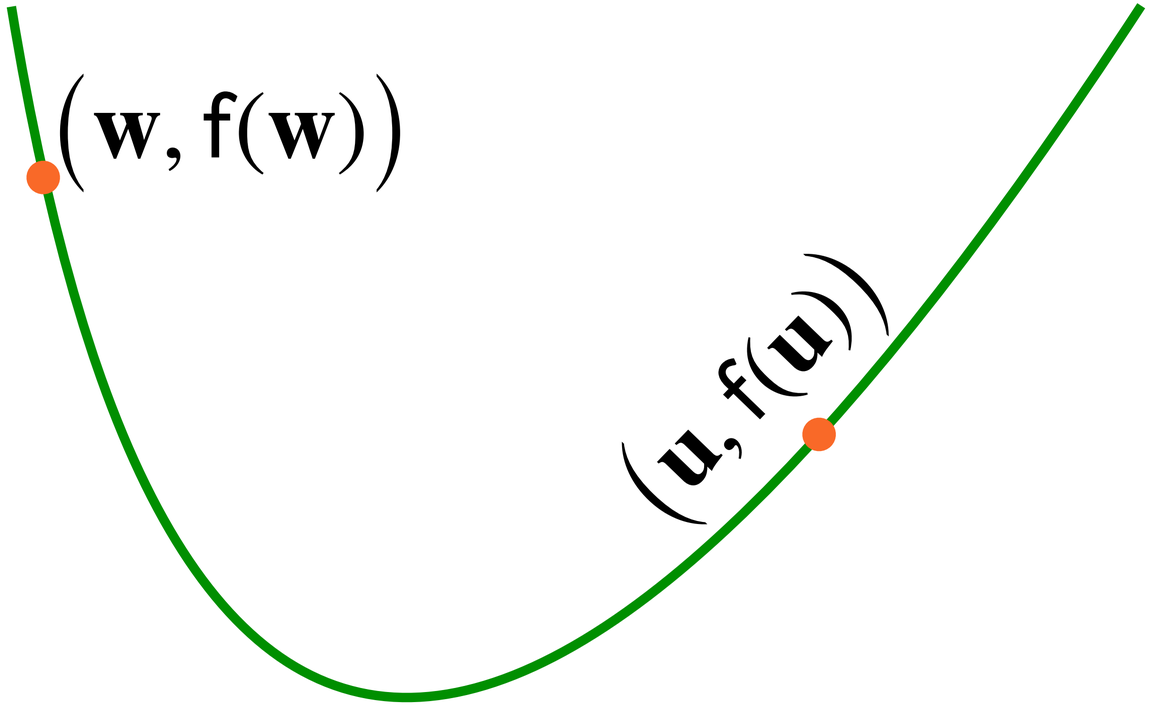
Convex Functions



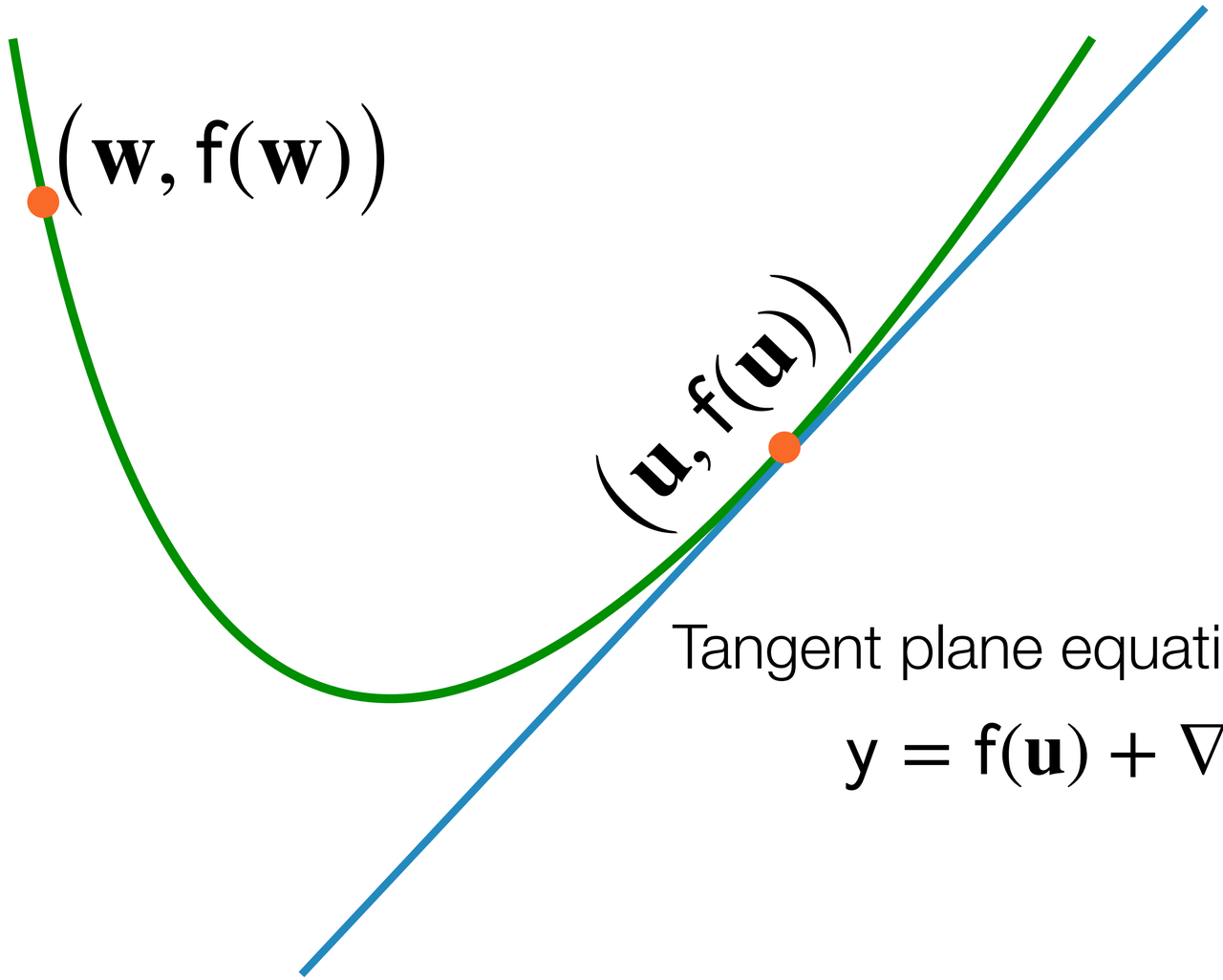
Multivariate Convex Functions



Multivariate Convex Functions



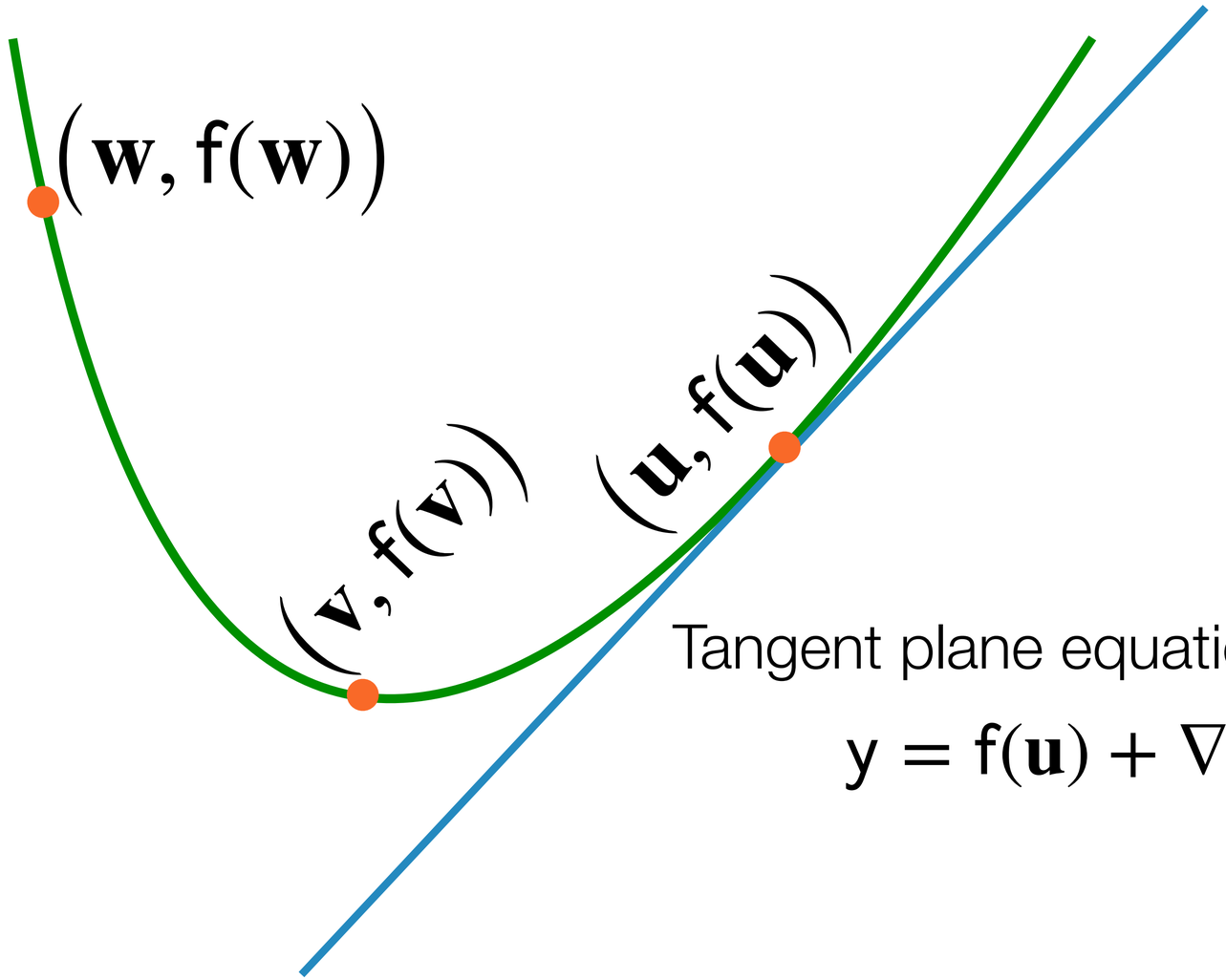
Multivariate Convex Functions



Tangent plane equation:

$$y = f(\mathbf{u}) + \nabla f(\mathbf{u}) \cdot (\mathbf{w} - \mathbf{u})$$

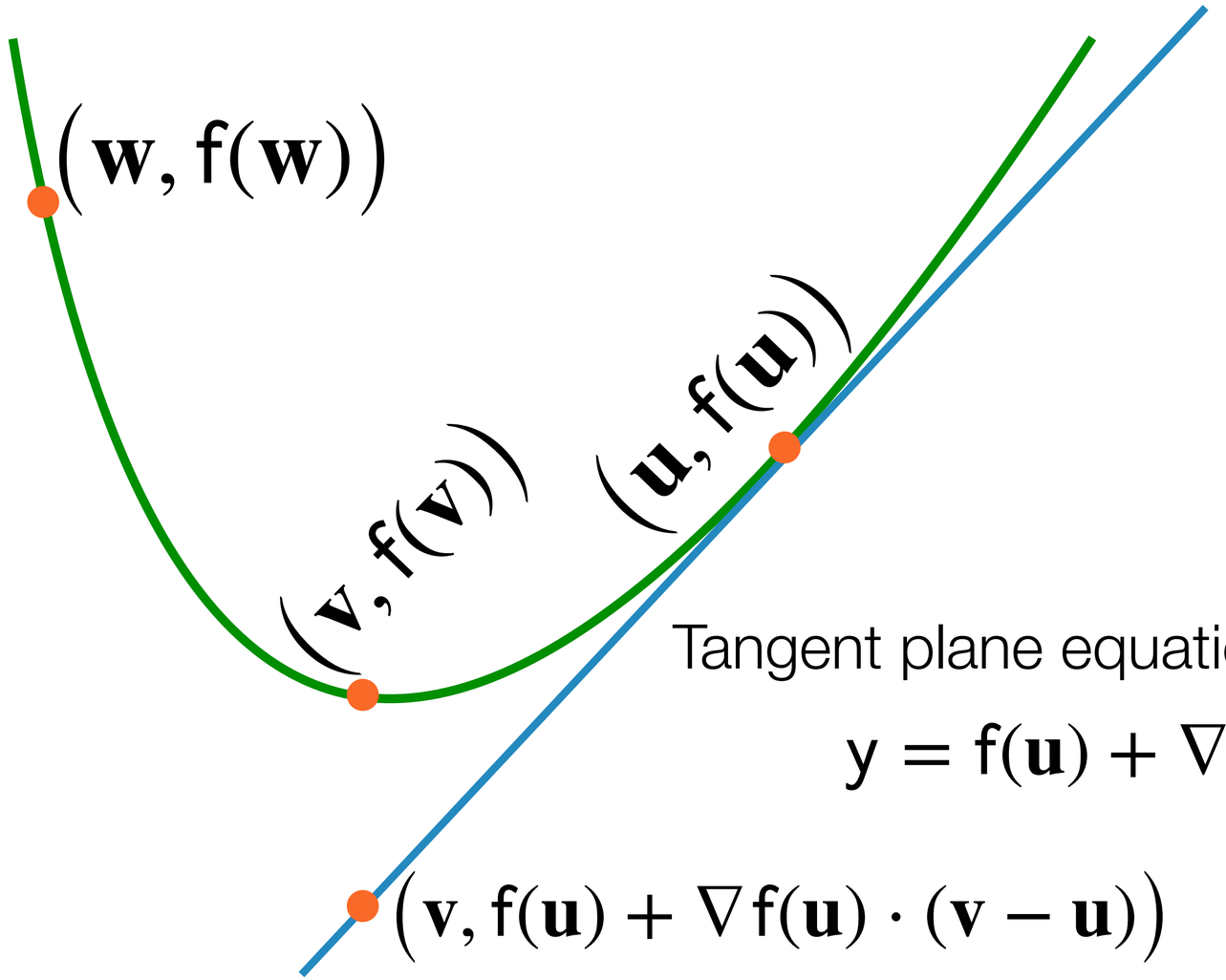
Multivariate Convex Functions



Tangent plane equation:

$$y = f(\mathbf{u}) + \nabla f(\mathbf{u}) \cdot (\mathbf{w} - \mathbf{u})$$

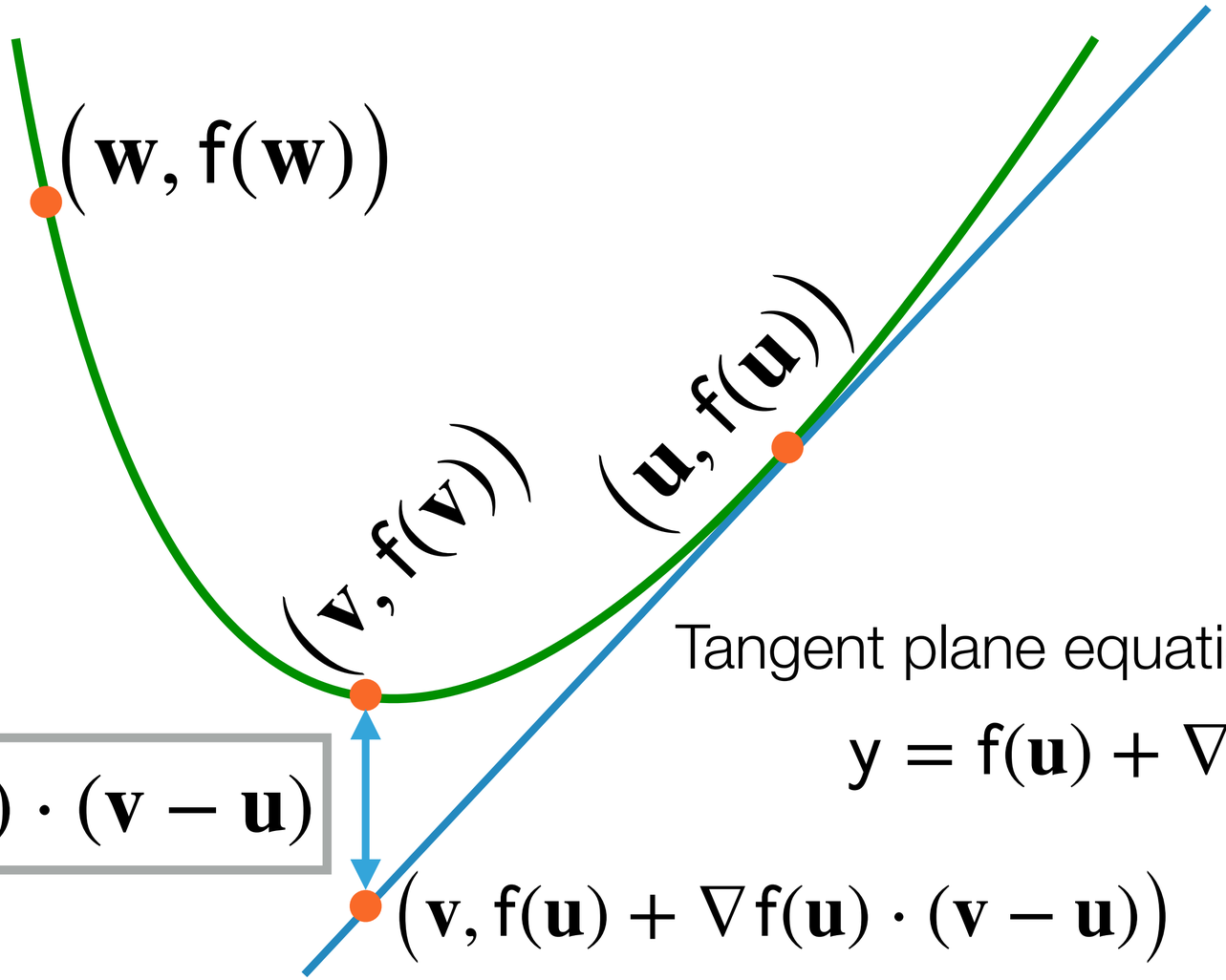
Multivariate Convex Functions



Tangent plane equation:

$$y = f(\mathbf{u}) + \nabla f(\mathbf{u}) \cdot (\mathbf{w} - \mathbf{u})$$

Multivariate Convex Functions



$$f(\mathbf{v}) \geq f(\mathbf{u}) + \nabla f(\mathbf{u}) \cdot (\mathbf{v} - \mathbf{u})$$

Tangent plane equation:

$$y = f(\mathbf{u}) + \nabla f(\mathbf{u}) \cdot (\mathbf{w} - \mathbf{u})$$

First Order Conditions

- For $\forall \alpha \in [0,1]$, $\mathbf{u} \in \mathbf{R}^d$, $\mathbf{v} \in \mathbf{R}^d$:

$$f\Big(\alpha \mathbf{u} + (1-\alpha)\mathbf{v}\Big) \leq \alpha f(\mathbf{u}) + (1-\alpha) f(\mathbf{v})$$

- $f(\mathbf{v}) \geq f(\mathbf{u}) + \nabla f(\mathbf{u}) \cdot (\mathbf{v} - \mathbf{u})$

- Fix $\mathbf{u} \in \mathbf{R}^d$, $\mathbf{v} \in \mathbf{R}^d$ and define $h : \mathbf{R} \rightarrow \mathbf{R}$ as $h(t) = f(\mathbf{u} + t\mathbf{v})$

  then $h(t)$ is a convex univariate function $\Big(h''(t) \geq 0 \Big)$

Second Order Conditions

Hessian of $f : \mathbf{R}^d \rightarrow \mathbf{R}$ is $d \times d$ matrix of second order derivatives $\nabla^2 f$:

$$H_{ij} = \frac{\partial^2 f}{\partial w_i \, \partial w_j}$$

f is convex iff:

$$\forall \mathbf{u} : \ \mathbf{u}^\top H \, \mathbf{u} \geq 0$$

This means that the smallest Eigen value of H is non-negative

COS234:

Introduction To Machine Learning

Prof. Yoram Singer

Topic: Stochastic Gradient Decent

Recap of Gradient Decent

Gradient Descent

- Input: function $f : \mathbf{R} \rightarrow \mathbf{R}_+$
- Goal: find $\hat{\mathbf{w}}$ such that

  $f(\hat{\mathbf{w}}) - f(\mathbf{w}^\star) \leq \epsilon$
- Choose initial value $\mathbf{w}_0$
- Loop:
  - $\mathbf{w}_{t+1} \leftarrow \mathbf{w}_t - \eta_t \nabla f(\mathbf{w}_t)$
  - Until …

```
def derivative_descent(w0, grad, eta):
    T, d = len(eta) + 1, len(w0)
    w = w0
    for t in range(1, T):
        g = grad(w)
        w = w - eta[t-1] * g
        if max(abs(g)) < 0.01: break
    return w
```

Learning Rate

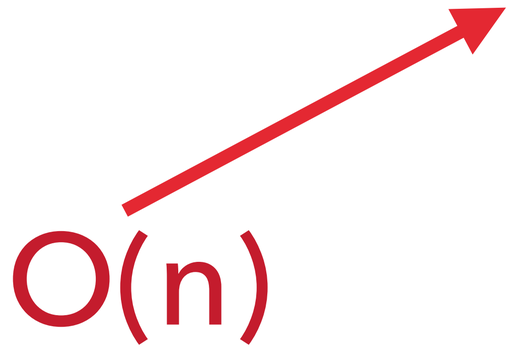
- Crucial in many learning problems
- Fixed learning-rate can be used in restricted circumstances
- Self-tuning procedure of learning-rate exist, notably AdaGrad
- In many applications:
  - Linear decrease $\eta_t = \frac{\eta_0}{b + st}$
  - Sub-linear decrease $\eta_t = \eta_0 / \sqrt{t}$
  - Where $\eta_0 \in [0.1, 1]$ (typically)

Runtime of GD

$$\mathbf{g}_t = \nabla \mathscr{L}(\mathbf{w}_t) = \frac{1}{n} \sum_{i=1}^{n} \nabla \ell_i(\mathbf{w})$$

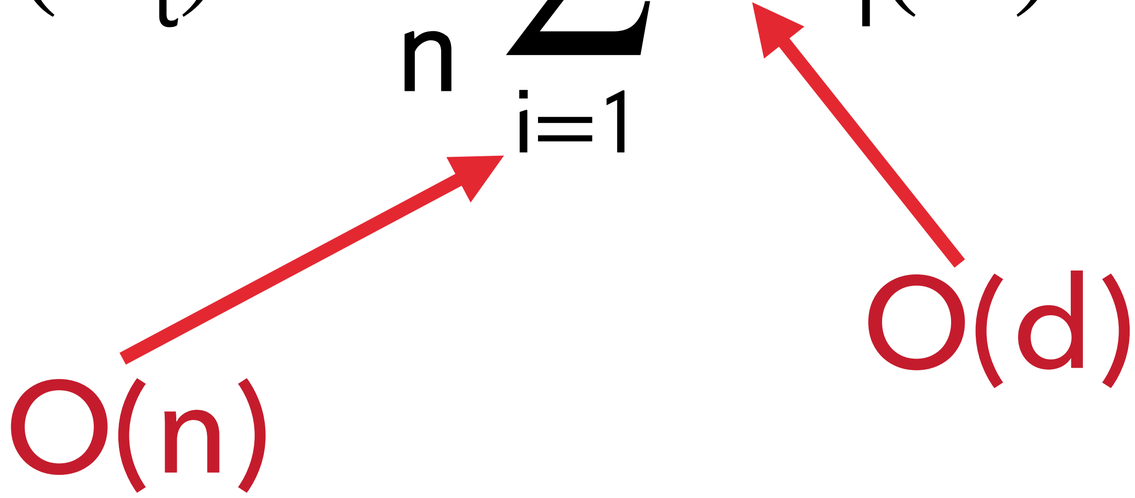
Runtime of GD

$$\mathbf{g}_t = \nabla \mathscr{L}(\mathbf{w}_t) = \frac{1}{n} \sum_{i=1}^{n} \nabla \ell_i(\mathbf{w})$$



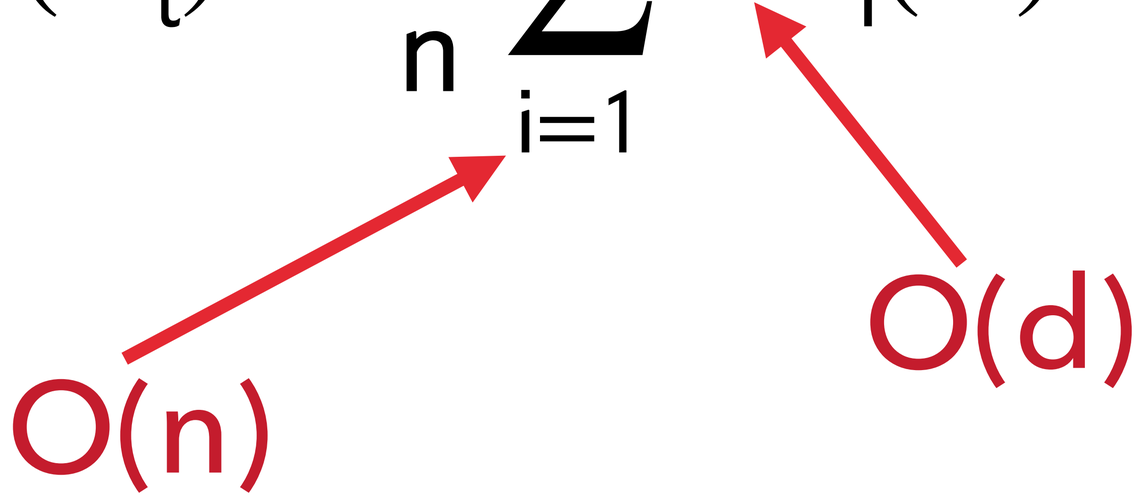
Runtime of GD

$$\mathbf{g}_t = \nabla \mathscr{L}(\mathbf{w}_t) = \frac{1}{n} \sum_{i=1}^{n} \nabla \ell_i(\mathbf{w})$$



Runtime of GD

$$\mathbf{g}_t = \nabla \mathscr{L}(\mathbf{w}_t) = \frac{1}{n} \sum_{i=1}^{n} \nabla \ell_i(\mathbf{w})$$



GD update takes O(nd) whereas

single CD update takes O(n+d)

Population Statistics

Population Statistics

Polls for how many votes each candidate will get

Population Statistics

Polls for how many votes each candidate will get

Ratings of books, products, movies, …

Population Statistics

Polls for how many votes each candidate will get

Ratings of books, products, movies, …

Likelihood of being admitted to Princeton

Population Statistics

Polls for how many votes each candidate will get

Ratings of books, products, movies, …

Likelihood of being admitted to Princeton

Estimate of likelihood of committing a second offense

Population Statistics

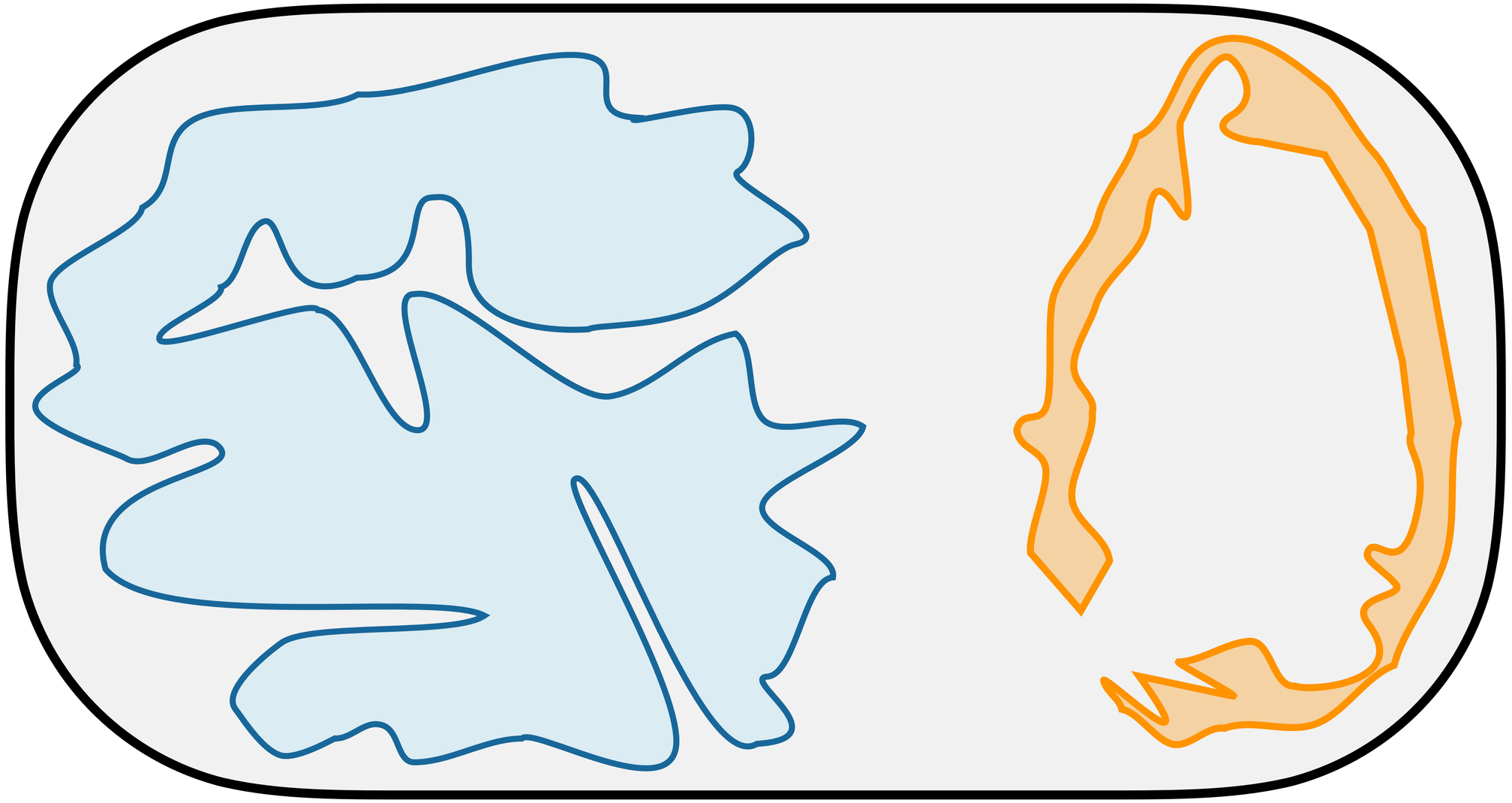
Polls for how many votes each candidate will get

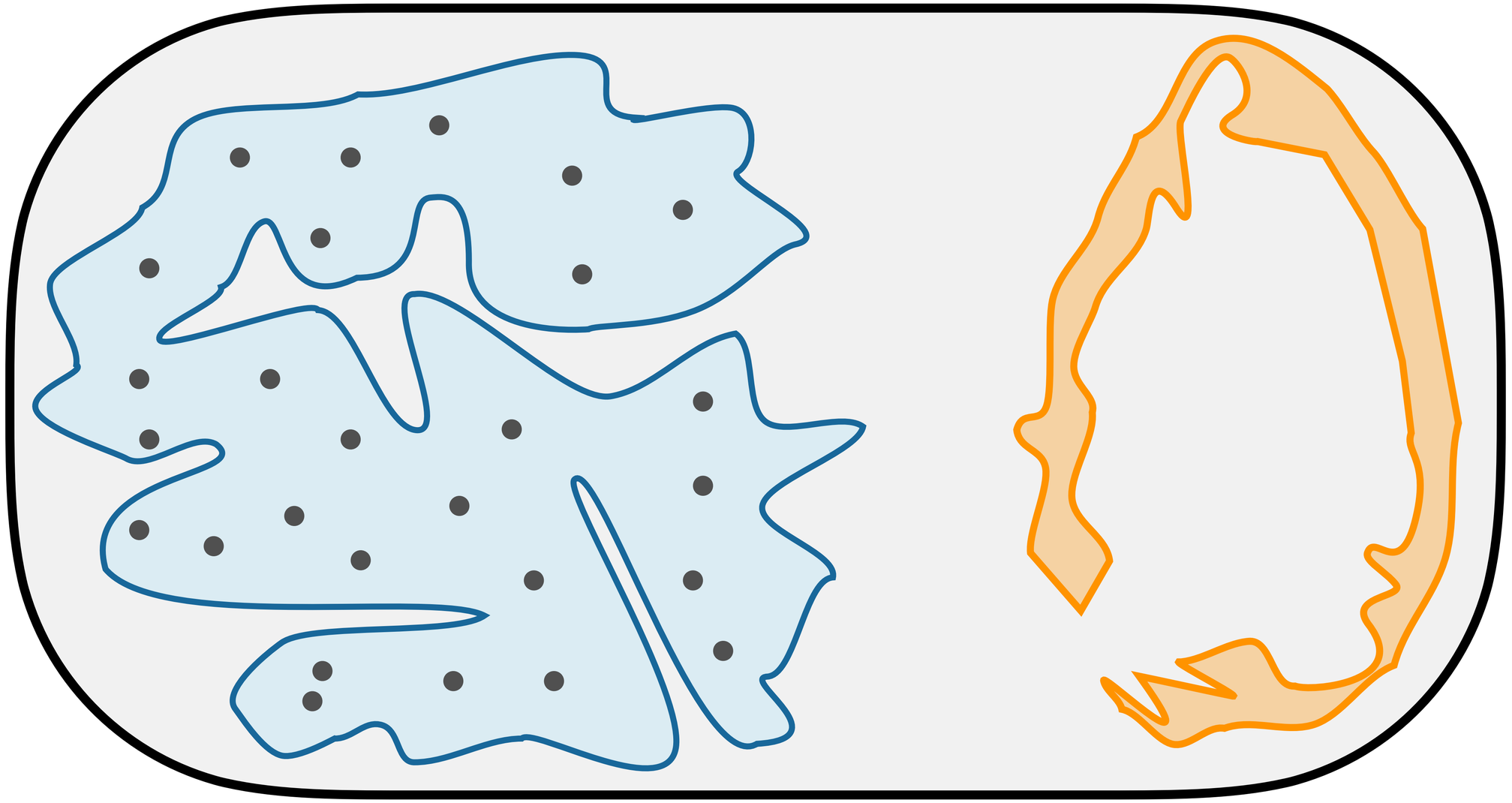
Ratings of books, products, movies, …

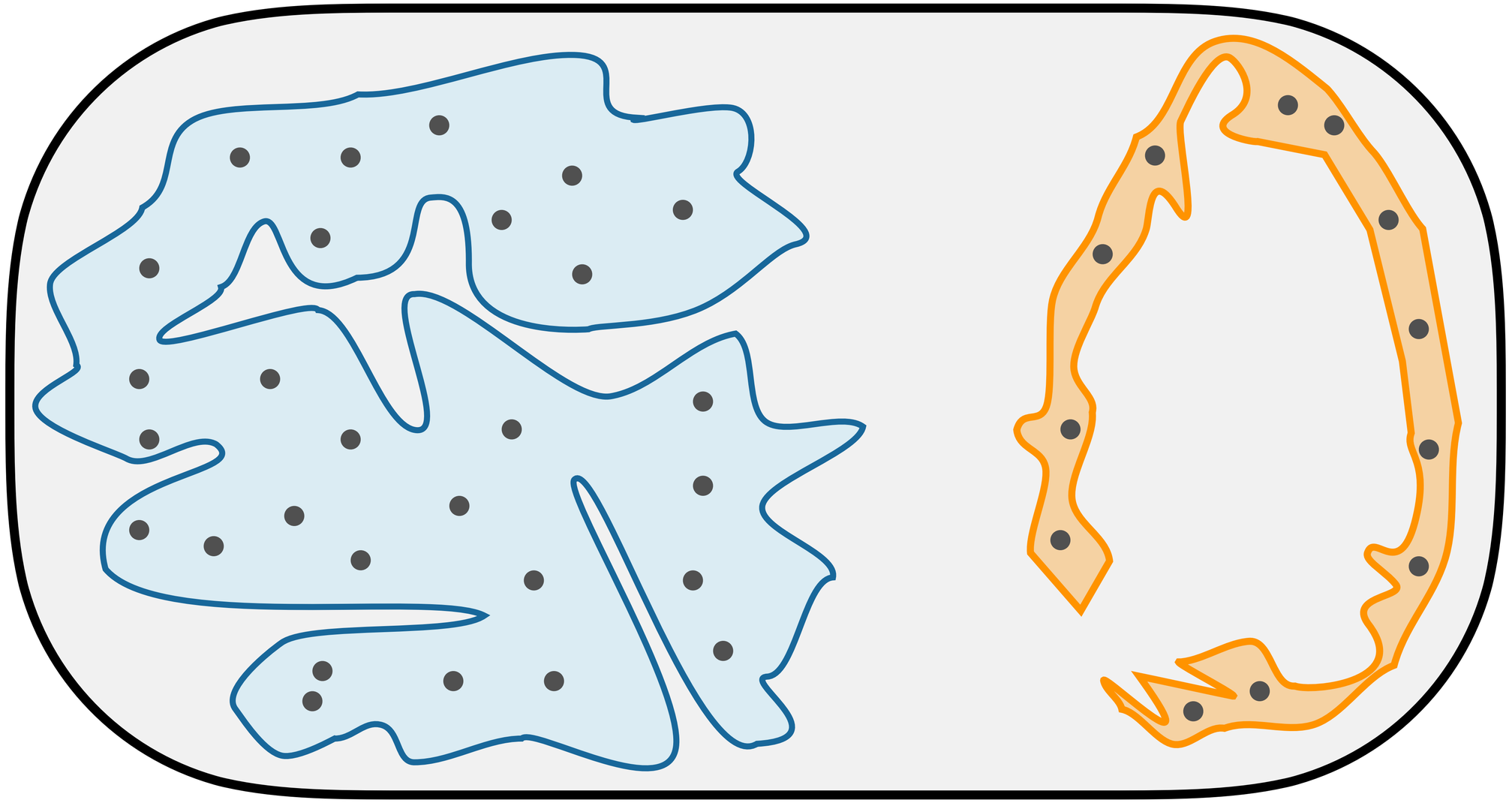
Likelihood of being admitted to Princeton

Estimate of likelihood of committing a second offense

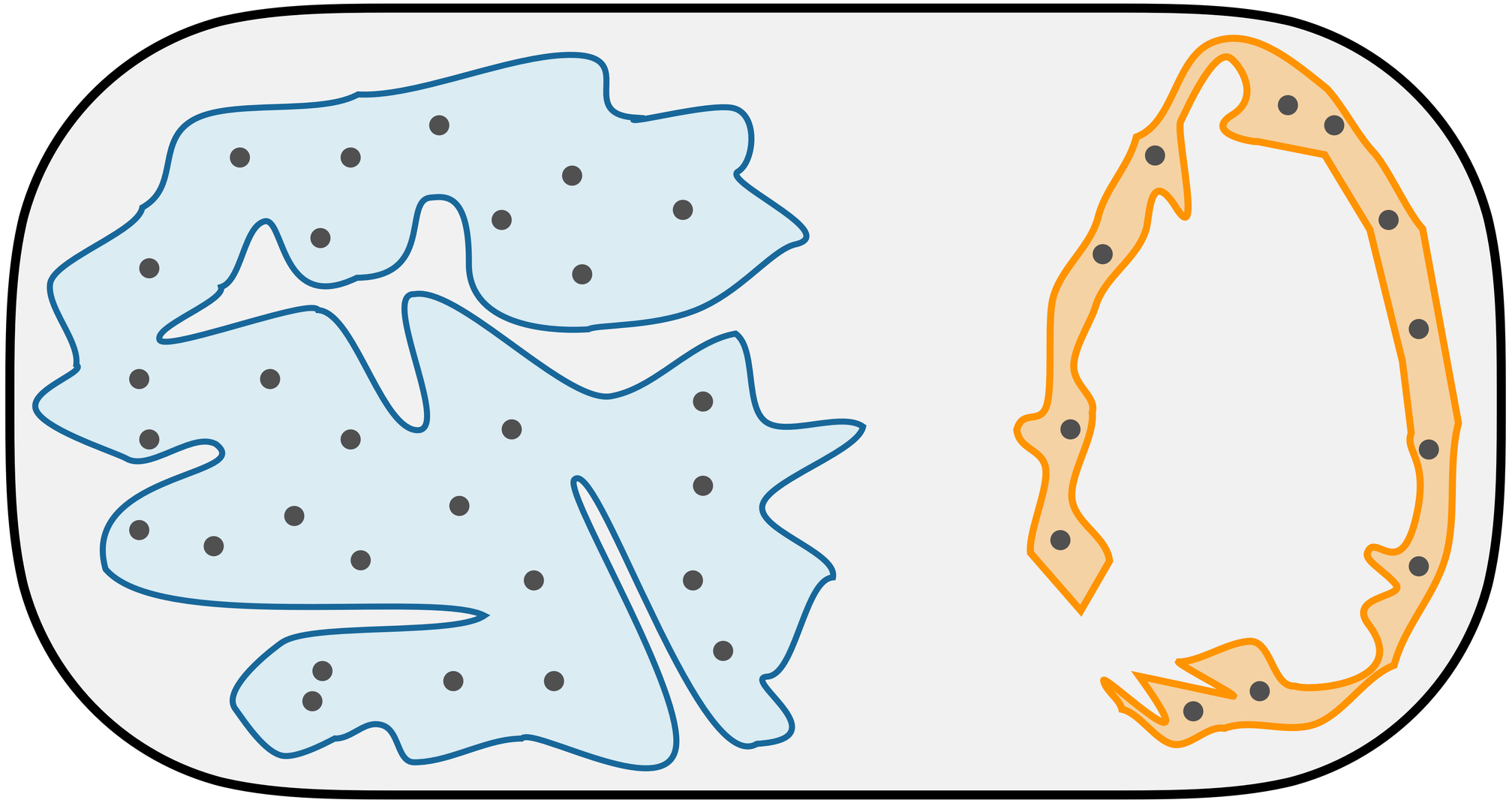
Estimate of a predicate of a large population by sampling

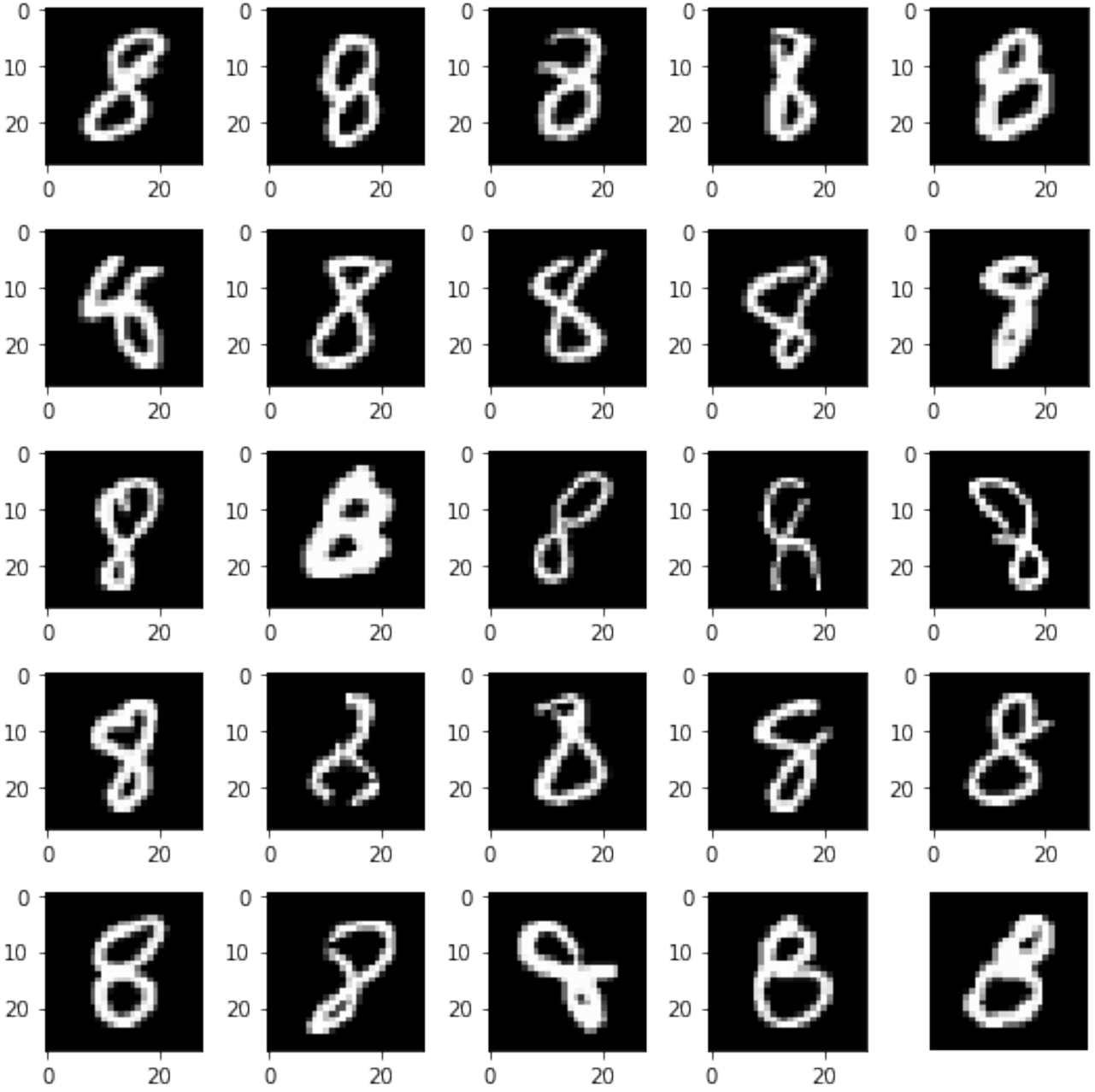


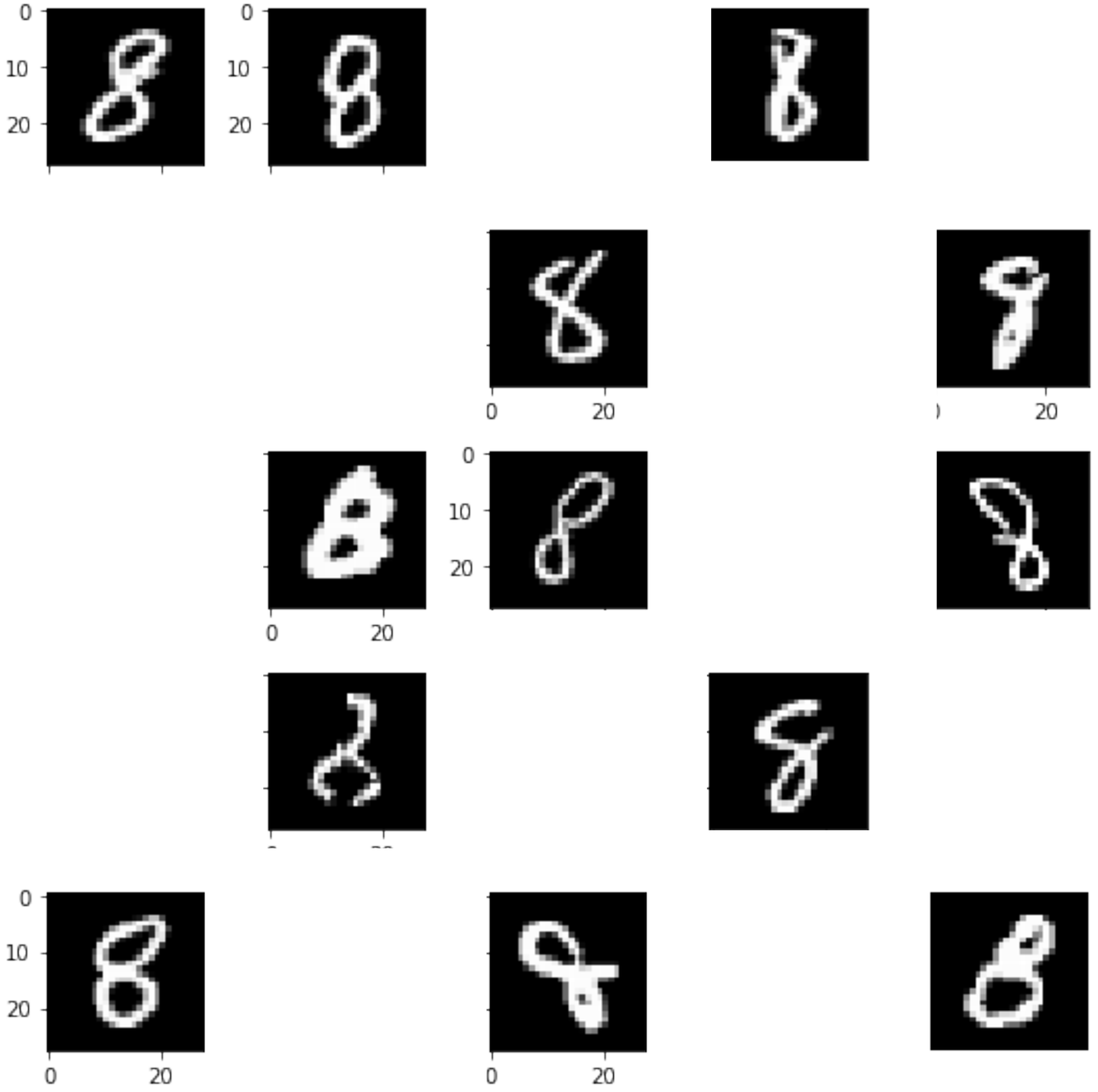




B: 28 votes **O**: 12 votes ⇒ 70% of population prefers **B** over **O**







(Stochastic) Estimate of Gradient

Empirical loss: $$\mathscr{L}(\mathbf{w}) = \frac{1}{n}\sum_{i=1}^{n} \ell(y_i, \mathbf{w}\cdot\mathbf{x}_i) \equiv \frac{1}{n}\sum_{i=1}^{n} \ell_i(\mathbf{w})$$

(Stochastic) Estimate of Gradient

Empirical loss: $$\mathscr{L}(\mathbf{w}) = \frac{1}{n}\sum_{i=1}^{n} \ell(y_i, \mathbf{w}\cdot\mathbf{x}_i) \equiv \frac{1}{n}\sum_{i=1}^{n} \ell_i(\mathbf{w})$$

Gradient of loss: $$\mathbf{g}_t = \nabla\mathscr{L}(\mathbf{w}_t) = \frac{1}{n}\sum_{i=1}^{n} \nabla\ell_i(\mathbf{w})$$

(Stochastic) Estimate of Gradient

Empirical loss: $$\mathscr{L}(\mathbf{w}) = \frac{1}{n}\sum_{i=1}^{n} \ell(y_i, \mathbf{w}\cdot\mathbf{x}_i) \equiv \frac{1}{n}\sum_{i=1}^{n} \ell_i(\mathbf{w})$$

Gradient of loss: $$\mathbf{g}_t = \nabla\mathscr{L}(\mathbf{w}_t) = \frac{1}{n}\sum_{i=1}^{n} \nabla\ell_i(\mathbf{w})$$

Pick random set $S \subset \{1,\ldots,n\}$ where $|S| \ll n$ and $\mathbf{P}\left[i \in S\right] = \frac{1}{n}$

(Stochastic) Estimate of Gradient

Empirical loss: $$\mathscr{L}(\mathbf{w}) = \frac{1}{n}\sum_{i=1}^{n} \ell(y_i, \mathbf{w}\cdot\mathbf{x}_i) \equiv \frac{1}{n}\sum_{i=1}^{n} \ell_i(\mathbf{w})$$

Gradient of loss: $$\mathbf{g}_t = \nabla\mathscr{L}(\mathbf{w}_t) = \frac{1}{n}\sum_{i=1}^{n} \nabla\ell_i(\mathbf{w})$$

Pick random set $S \subset \{1,\ldots,n\}$ where $|S| \ll n$ and $\mathbf{P}\left[i \in S\right] = \frac{1}{n}$

Estimates: $$\widehat{\mathscr{L}}(\mathbf{w}) = \frac{1}{|S|}\sum_{i\in S} \ell_i(\mathbf{w}) \qquad \widehat{\mathbf{g}}_t = \frac{1}{|S|}\sum_{i\in S} \nabla\ell_i(\mathbf{w})$$

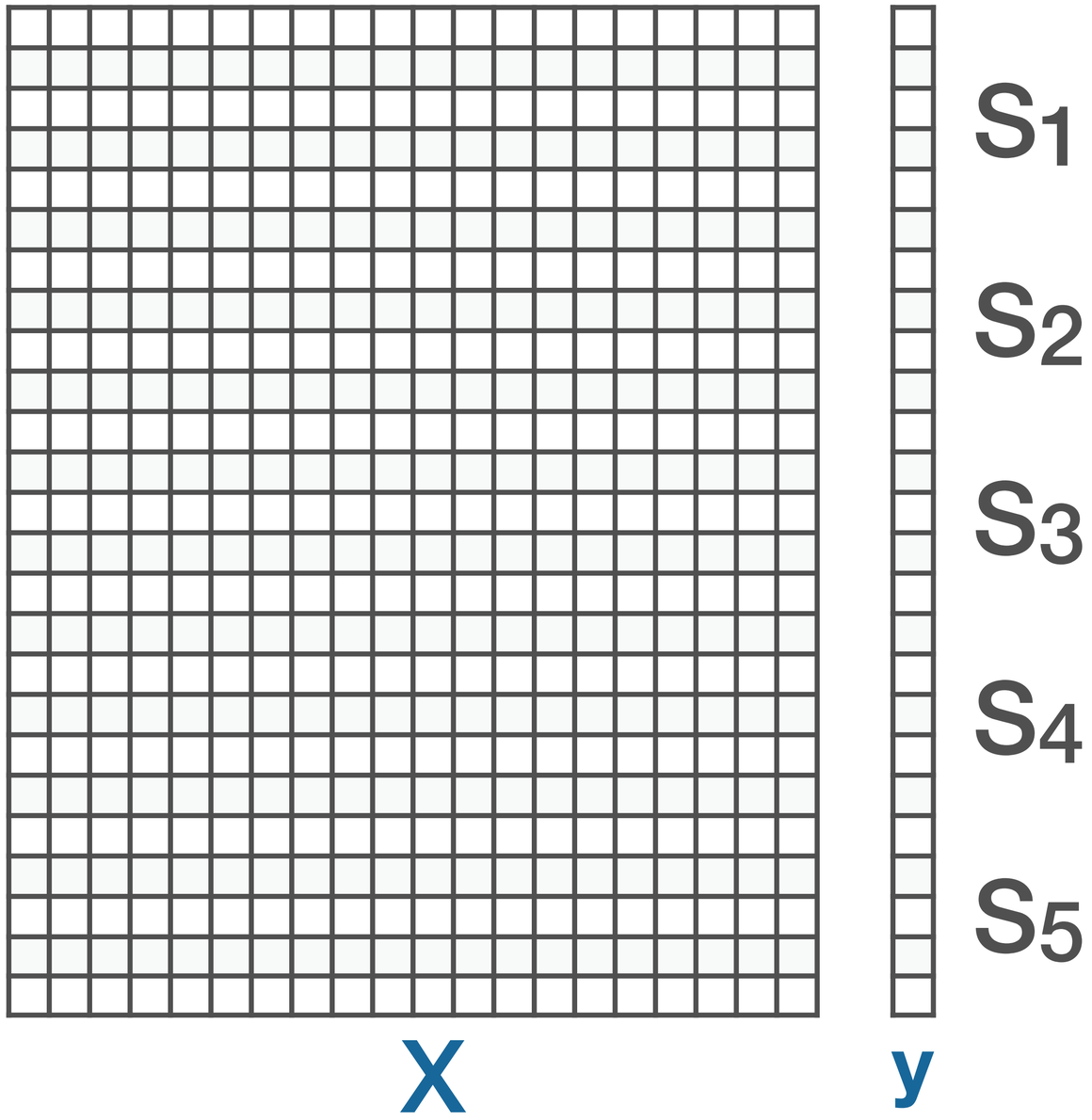
Properties of Estimates

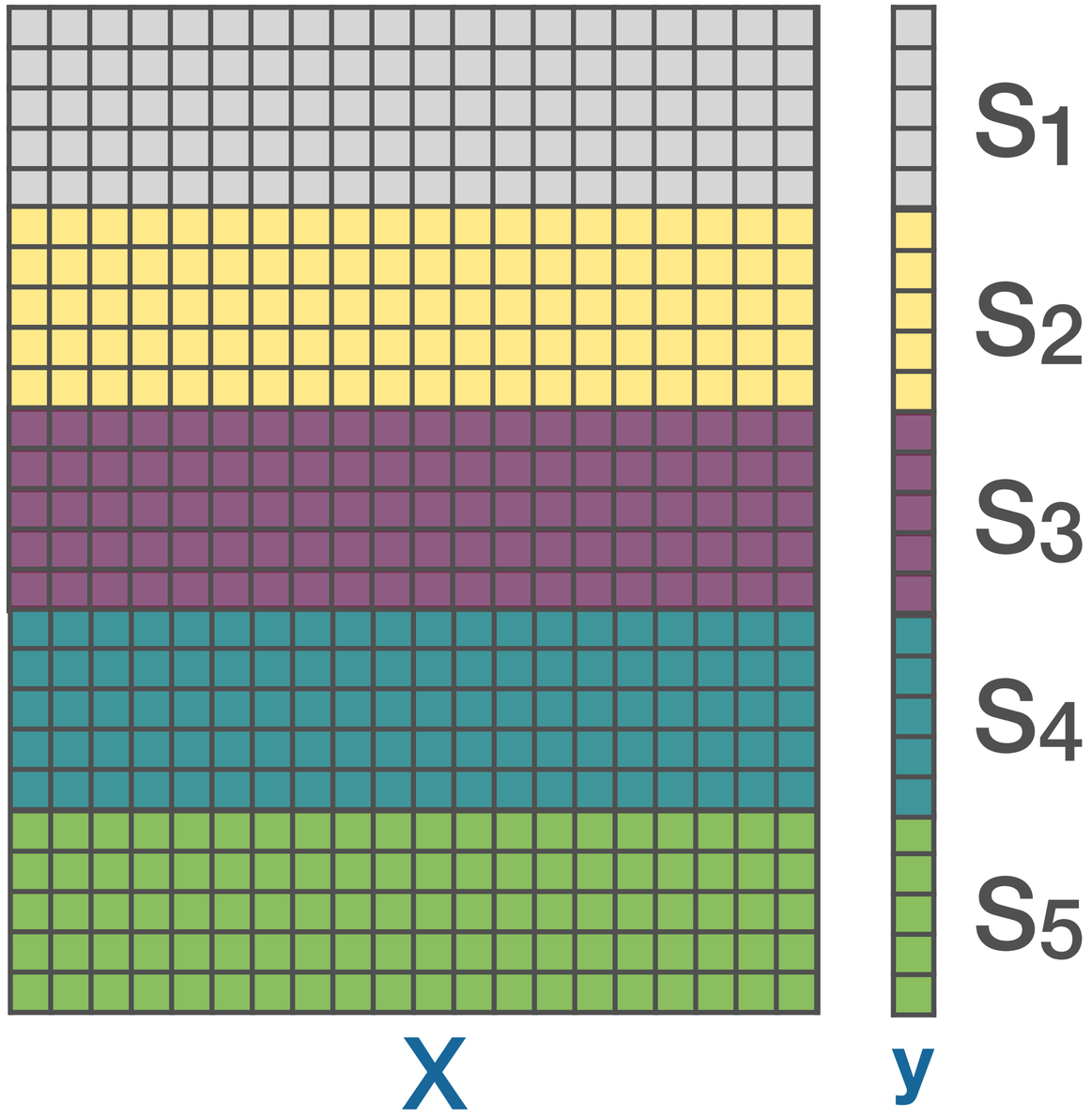
S generated at random by choosing indices from {1,…,n} w/ or w/o replacement

Then

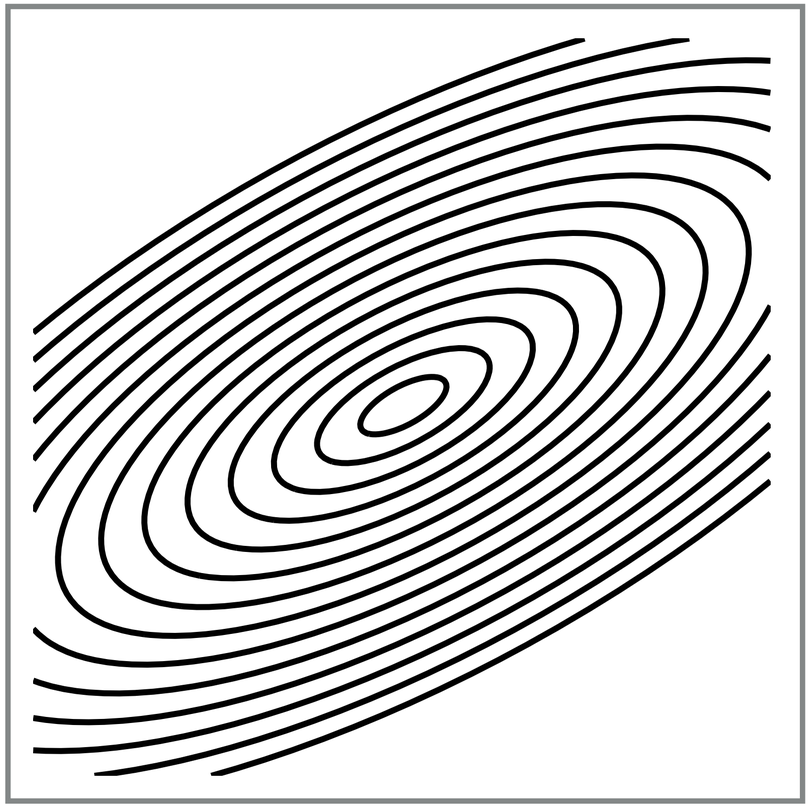
$$\forall \mathbf{w}: \quad \mathbf{E}\left[\widehat{\mathscr{L}}(\mathbf{w})\right] = \mathscr{L}(\mathbf{w}) \quad \text{and} \quad \mathbf{E}\left[\hat{\mathbf{g}}\right] = \nabla \mathscr{L}(\mathbf{w})$$

Variance of estimates proportional to $\dfrac{1}{\sqrt{|S|}}$

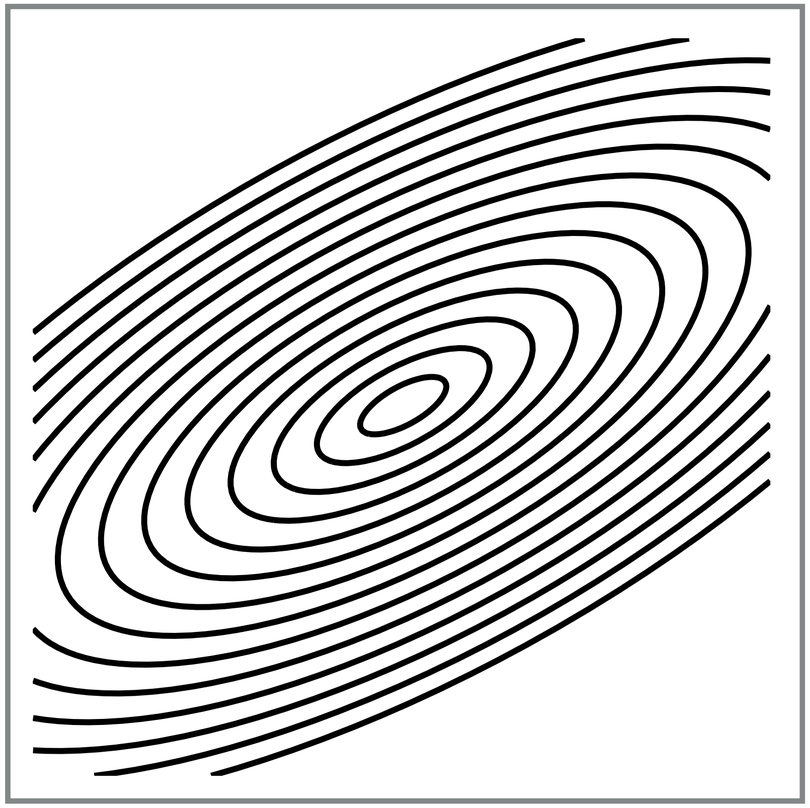




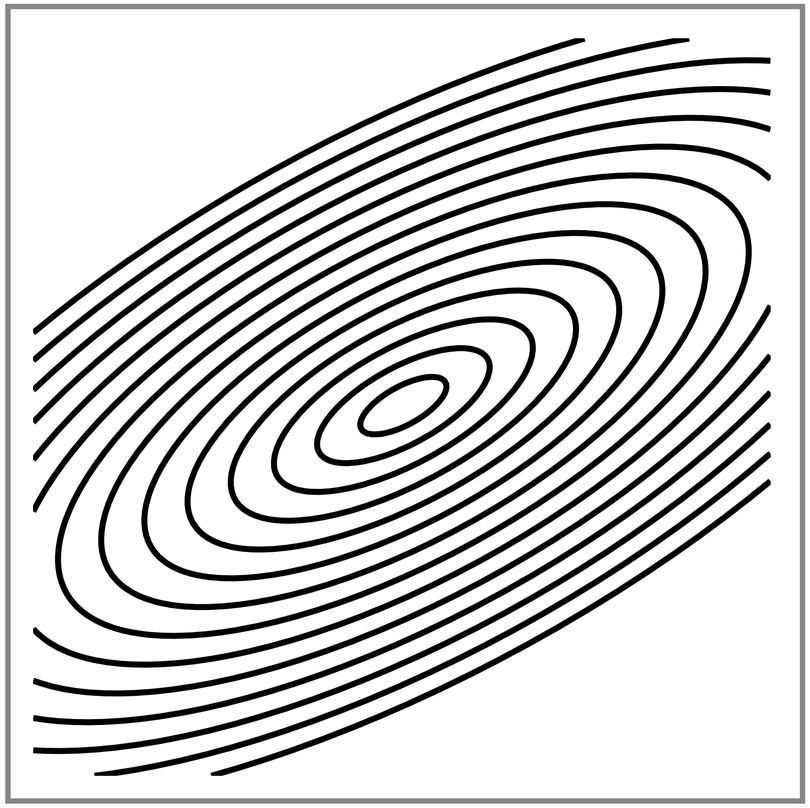
SGD



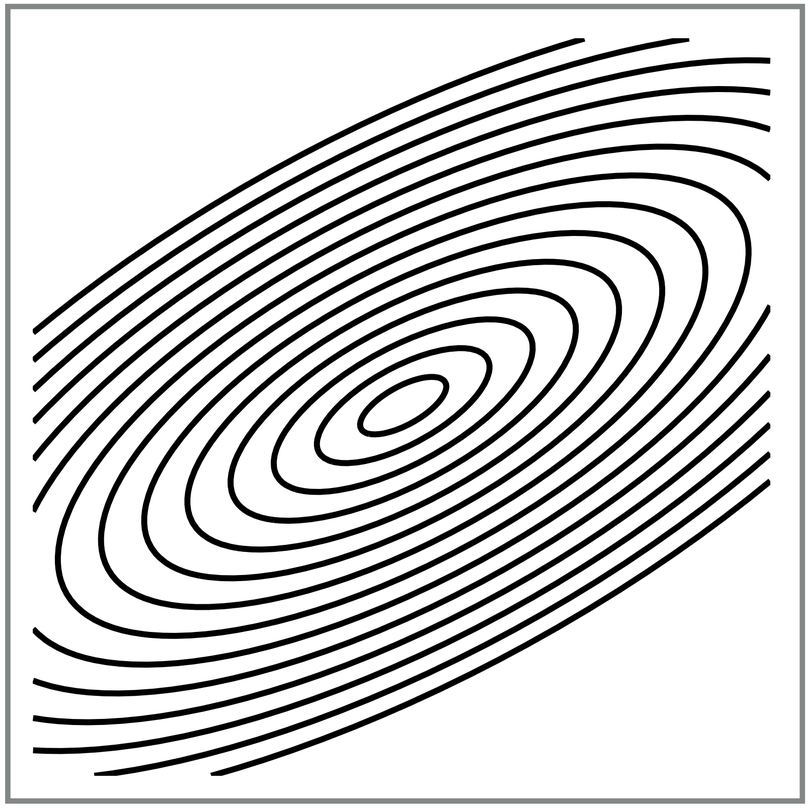
GD



SGD

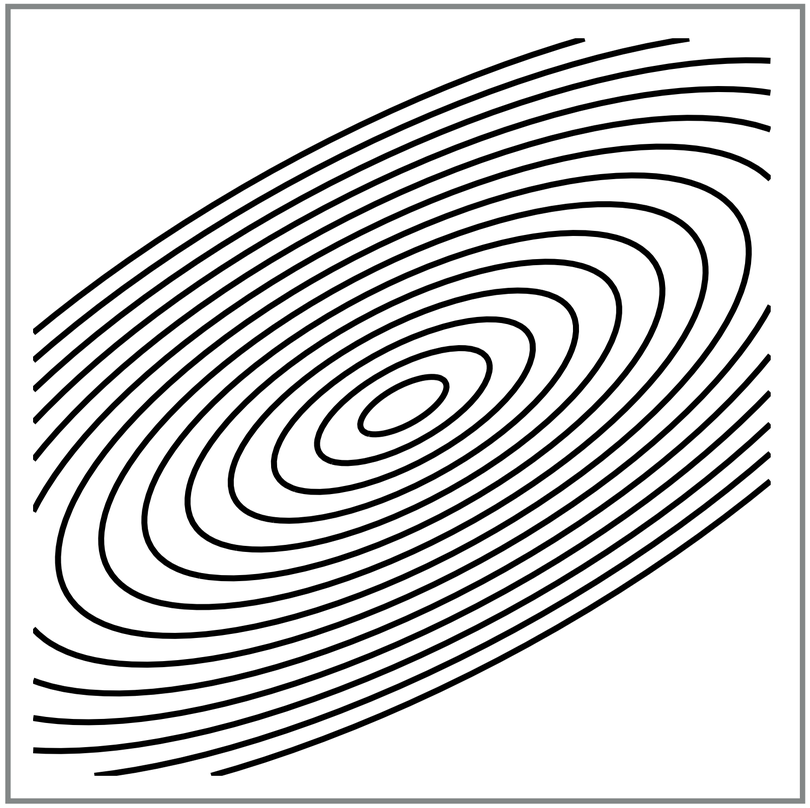


GD

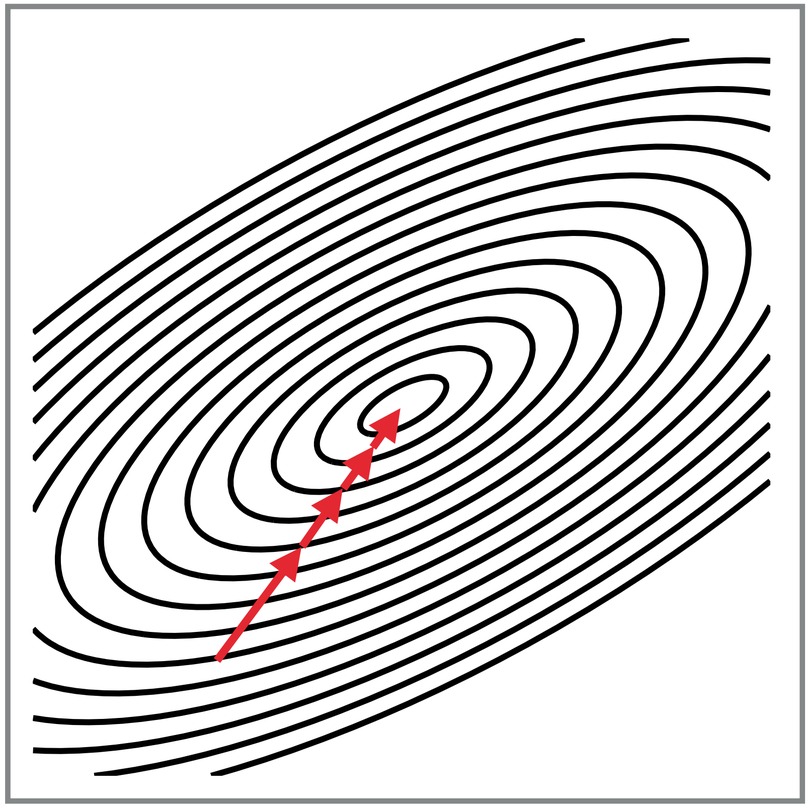


$$\mathbf{w}_{t+1} \leftarrow \mathbf{w}_t - \eta \nabla \mathsf{L}(\mathbf{w}_t)$$

SGD

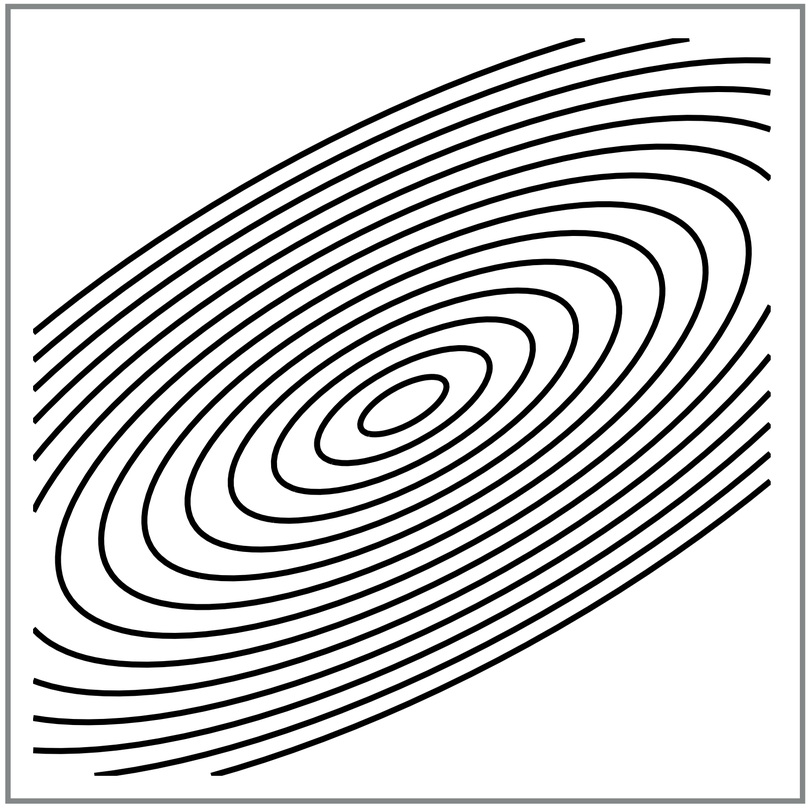


GD



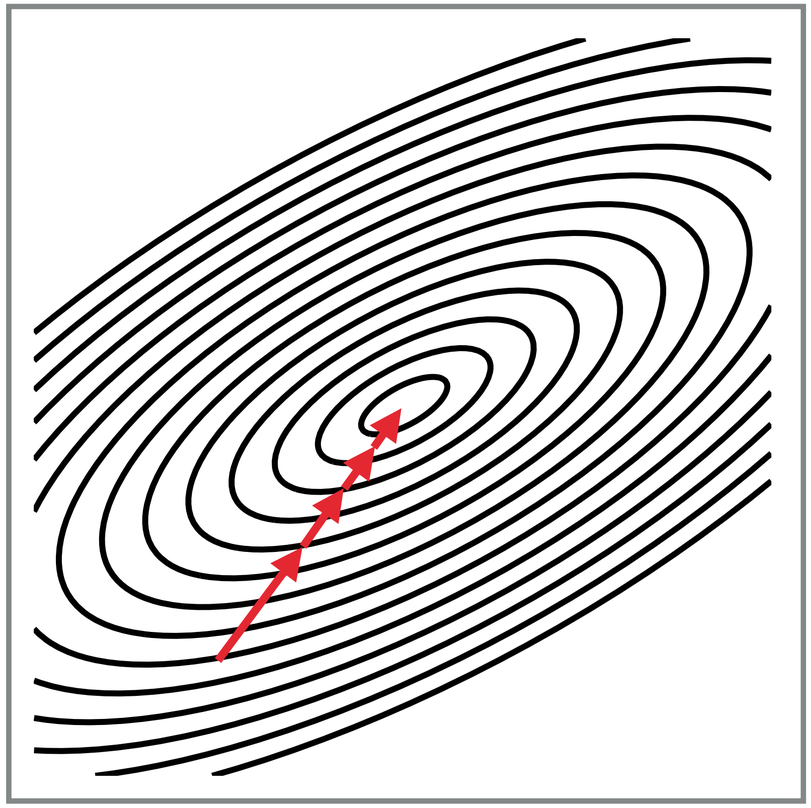
$$\mathbf{w}_{t+1} \leftarrow \mathbf{w}_t - \eta \nabla \mathsf{L}(\mathbf{w}_t)$$

SGD



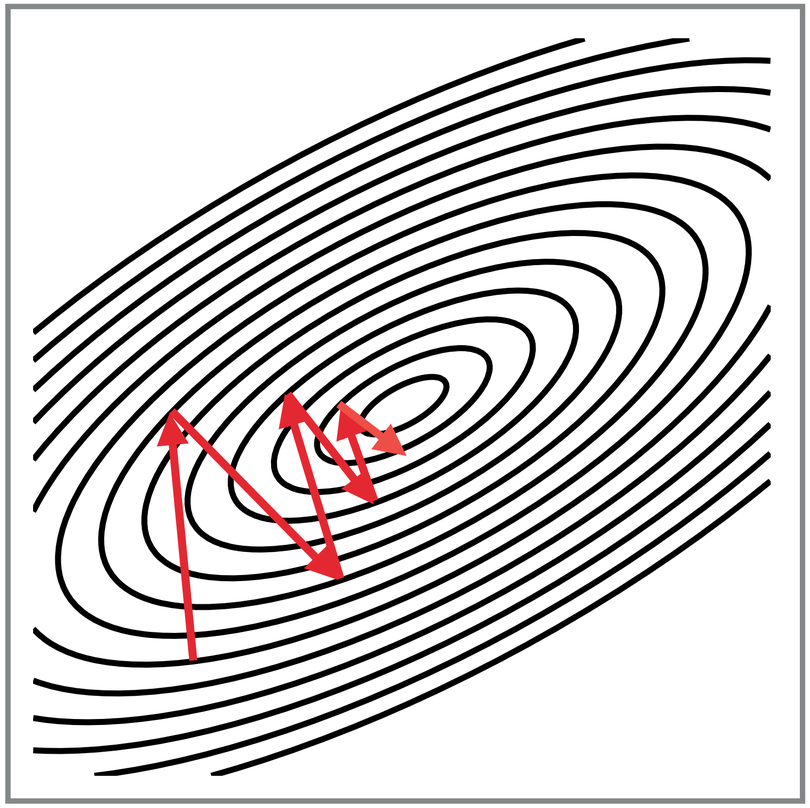
$$\mathbf{w}_{t+1} \leftarrow \mathbf{w}_t - \eta_t \widehat{\nabla \mathsf{L}}(\mathbf{w}_t)$$

GD



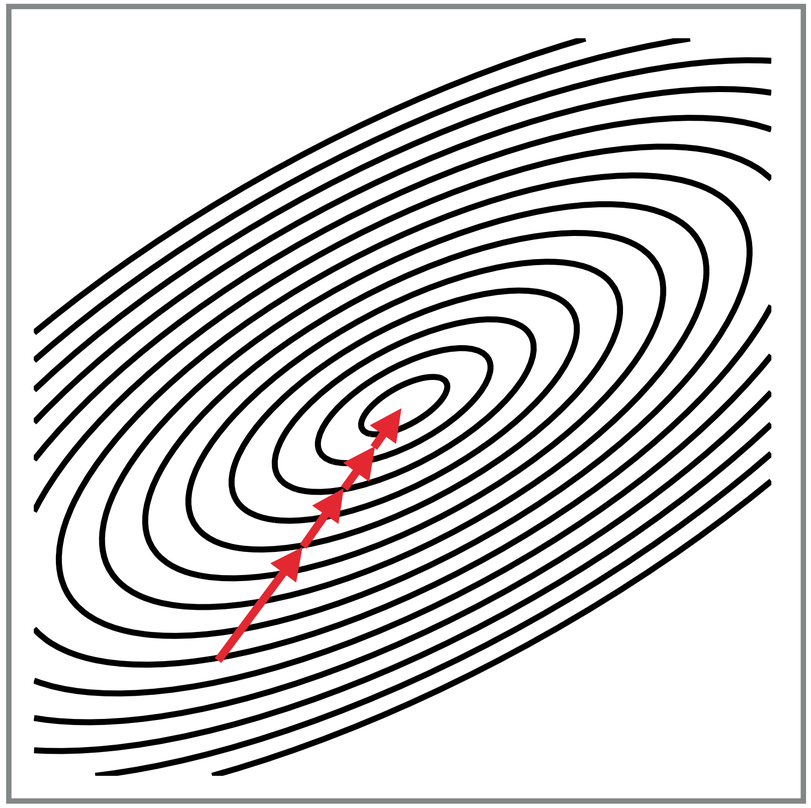
$$\mathbf{w}_{t+1} \leftarrow \mathbf{w}_t - \eta \nabla \mathsf{L}(\mathbf{w}_t)$$

SGD



$$\mathbf{w}_{t+1} \leftarrow \mathbf{w}_t - \eta_t \widehat{\nabla \mathsf{L}}(\mathbf{w}_t)$$

GD



$$\mathbf{w}_{t+1} \leftarrow \mathbf{w}_t - \eta \nabla \mathsf{L}(\mathbf{w}_t)$$

Parameter Averaging

Each $\mathbf{w}_t$ is a stochastic estimate

Parameter Averaging

Each $\mathbf{w}_t$ is a stochastic estimate

Different choices of $S_1, S_2, \ldots, S_t$ lead to different $\mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_t$

Parameter Averaging

Each $\mathbf{w}_t$ is a stochastic estimate

Different choices of $S_1, S_2, \ldots, S_t$ lead to different $\mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_t$

Need a "stable" estimate & convergence gurantees

Parameter Averaging

Each $\mathbf{w}_t$ is a stochastic estimate

Different choices of $S_1, S_2, \ldots, S_t$ lead to different $\mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_t$

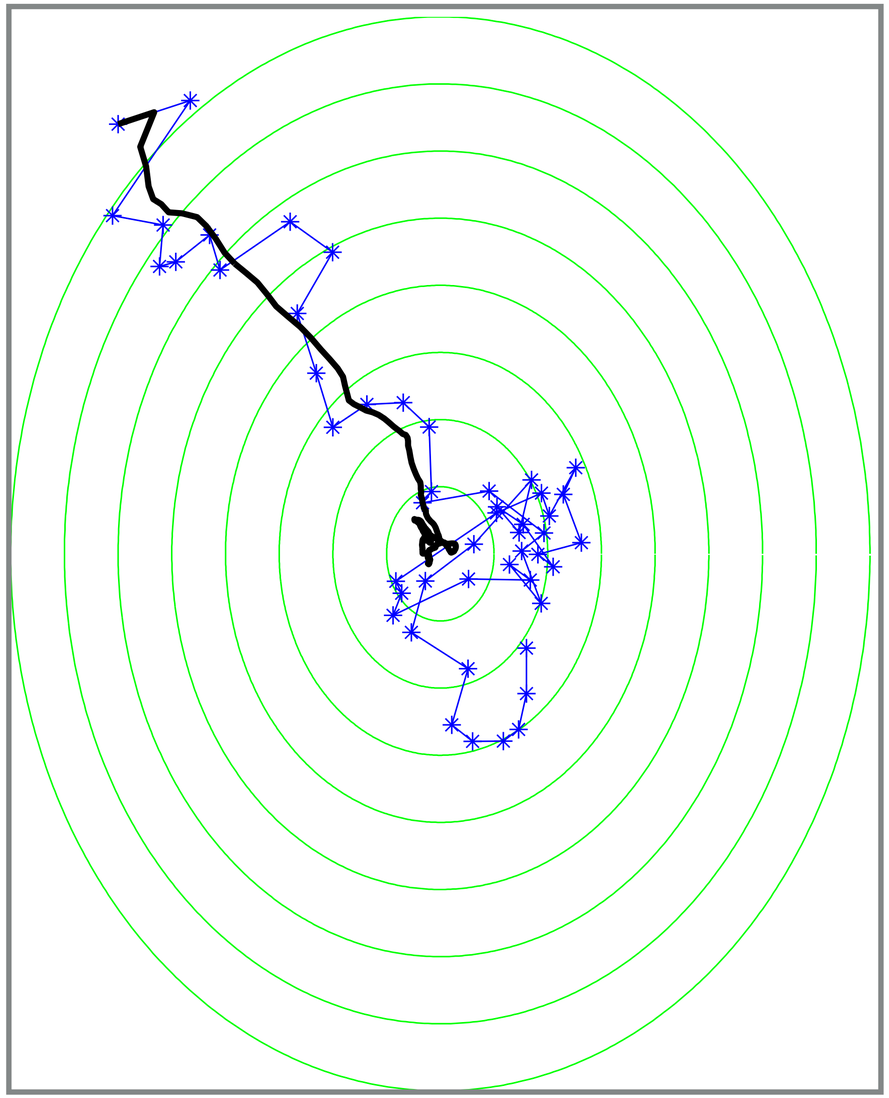
Need a "stable" estimate & convergence gurantees

Take weighted average of parameters:

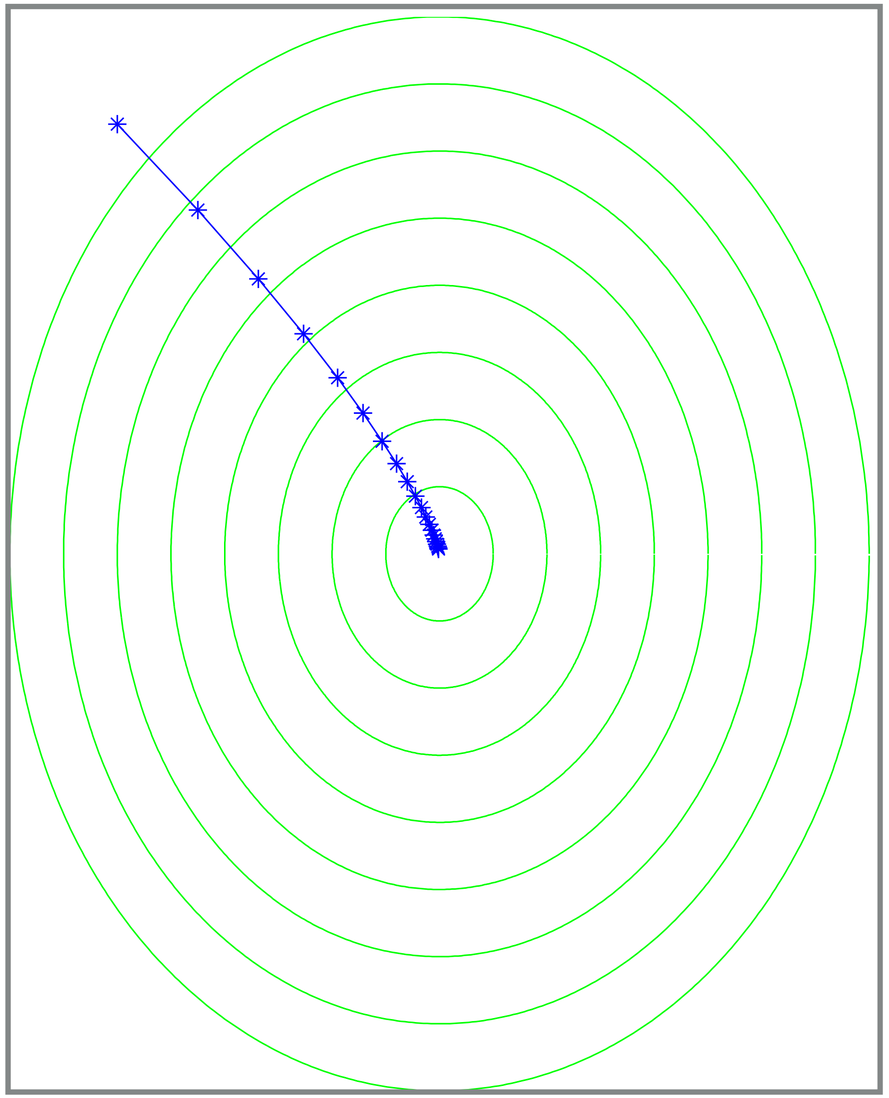
$$\bar{\mathbf{w}}_t = \sum_{i=1}^{t} \alpha_i \mathbf{w}_i \quad \text{where} \quad \sum_{i=1}^{t} \alpha_i = 1$$

Last	$\alpha_{i<T} = 0 \;\; \alpha_T = 1$	$\bar{\mathbf{w}}_T = \mathbf{w}_T$
Uniform	$\alpha_i = 1/T$	$\bar{\mathbf{w}}_T = \frac{1}{T} \sum_{t \le T} \mathbf{w}_t$
Suffix	$i \ge T - L : \;\; \alpha_i = \frac{1}{L}$	$\bar{\mathbf{w}}_T = \frac{1}{L} \sum_{T-L<t\le T} \mathbf{w}_t$
Geometric	$\alpha_i = \frac{\gamma^{T-i}}{1-\gamma}$	$\bar{\mathbf{w}}_{t+1} = (1-\gamma)\bar{\mathbf{w}}_t + \gamma \mathbf{w}_t$

SGD



GD



Learning Rate Scheduling

SGD is often used with a learning rate schedule $\frac{\eta_0}{\sqrt{t}}$ or $\frac{\eta_0}{1 + t/c}$

If η_0 is too small progress is slow or fail to end sufficiently close to optimum

If η_0 is too large we saw that even GD can diverge - move away from optimum

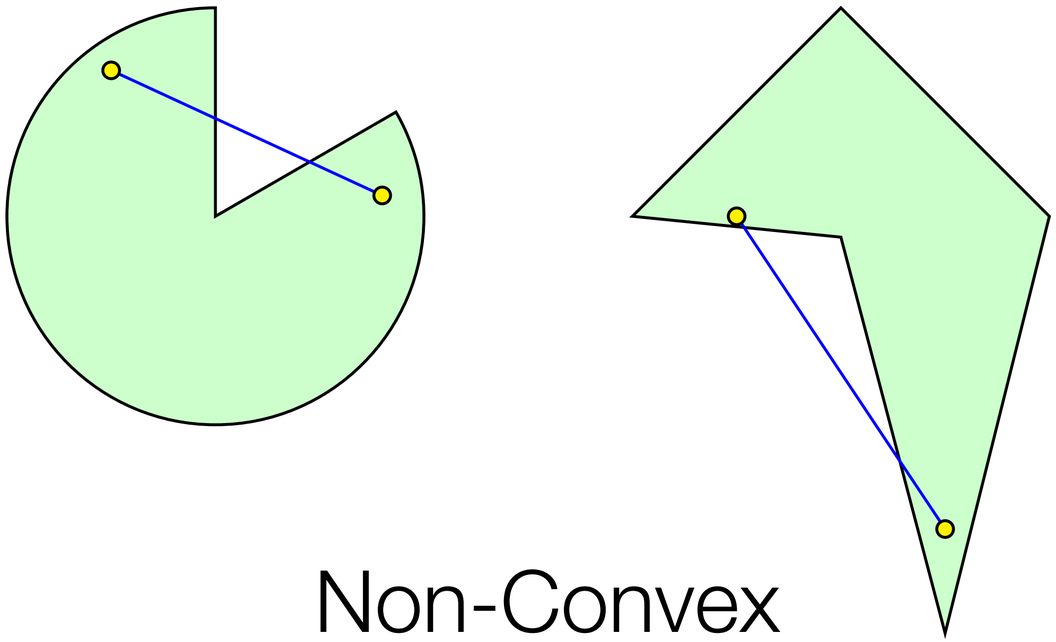
Since $\eta_t \searrow 0$ even if initial η_0 is too large eventually η_t becomes well suited

Start with η_0 which might (or might not) be too big safeguard from diverging ?

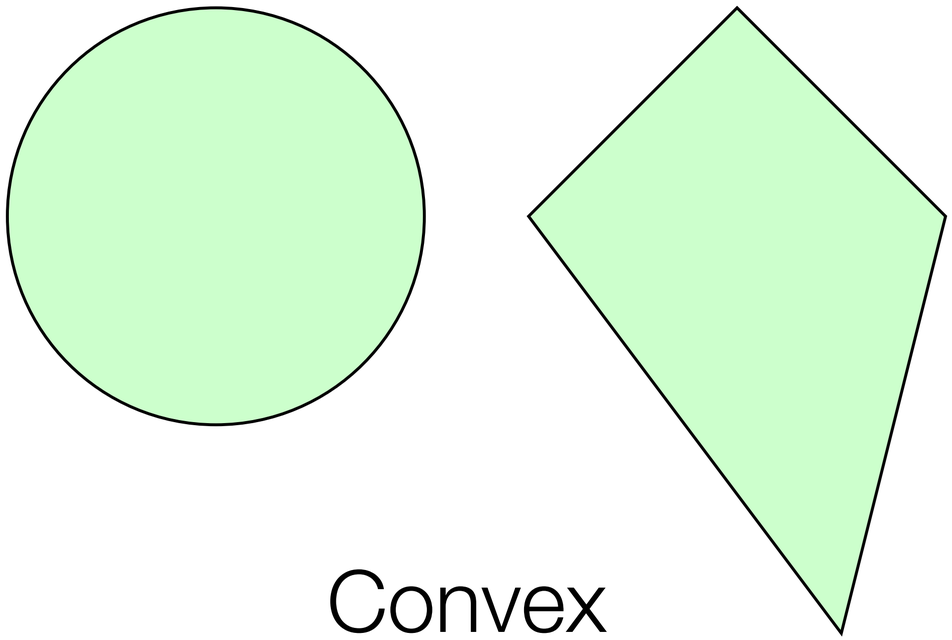
Convex Sets

Set Ω is convex if for any $\mathbf{u}, \mathbf{v} \in \Omega$ line segment between $\mathbf{u}$ and $\mathbf{v}$ is in Ω:

$$\forall \alpha \in [0,1] : \alpha \mathbf{u} + (1 - \alpha)\mathbf{v} \in \Omega$$

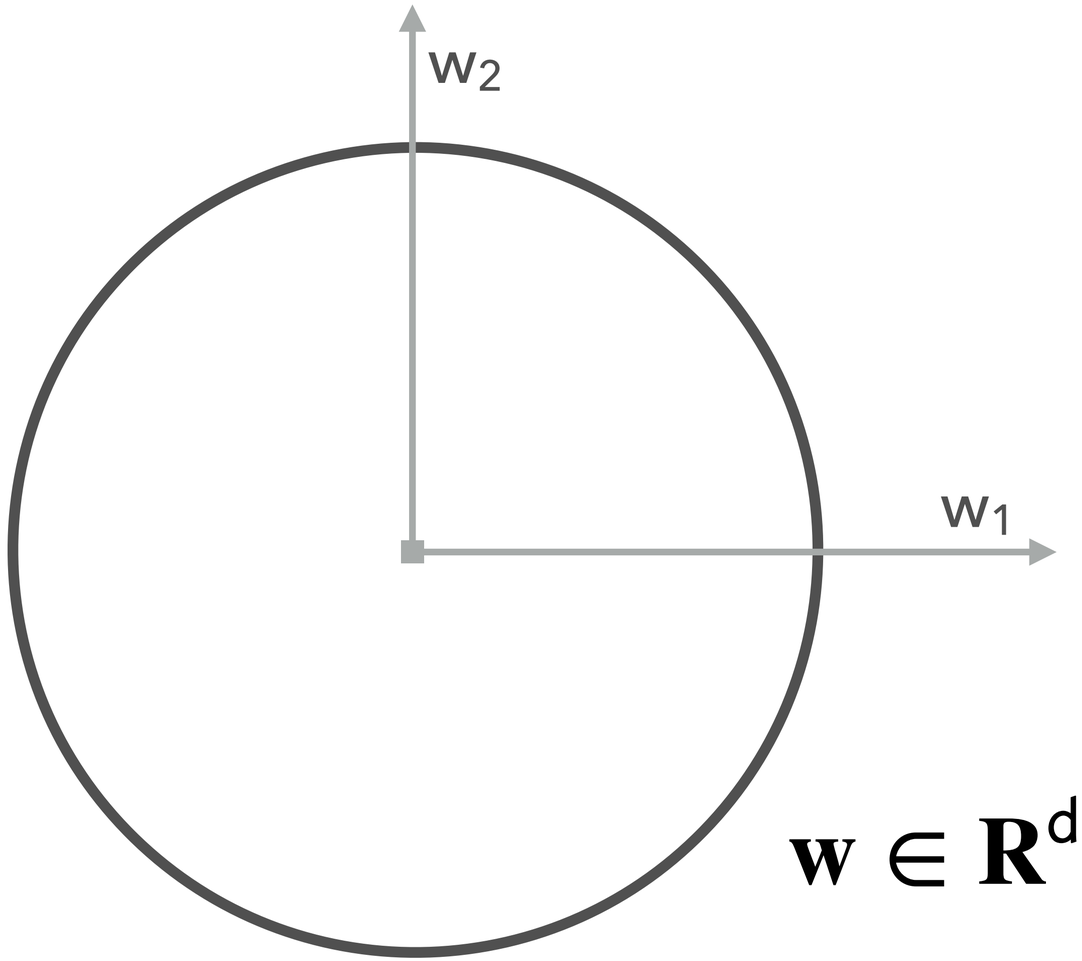


Non-Convex



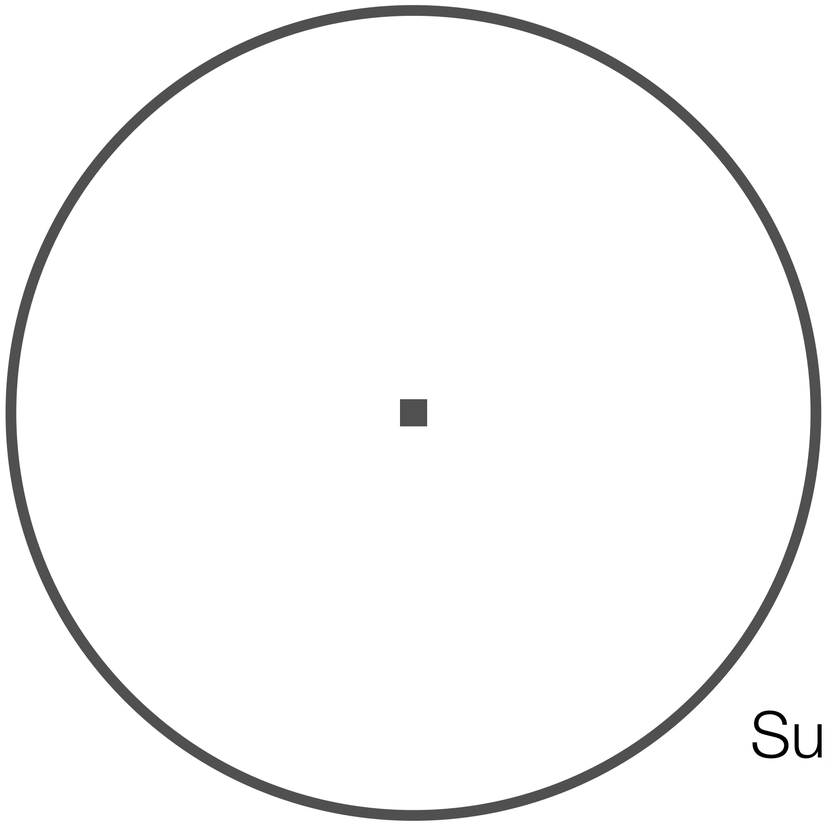
Convex

Euclidean Ball



$$\mathbf{w} \in \mathbf{R}^d : \quad \|\mathbf{w}\| \leq r \quad \Leftrightarrow \quad \sum_{j=1}^{d} w_j^2 \leq r^2$$

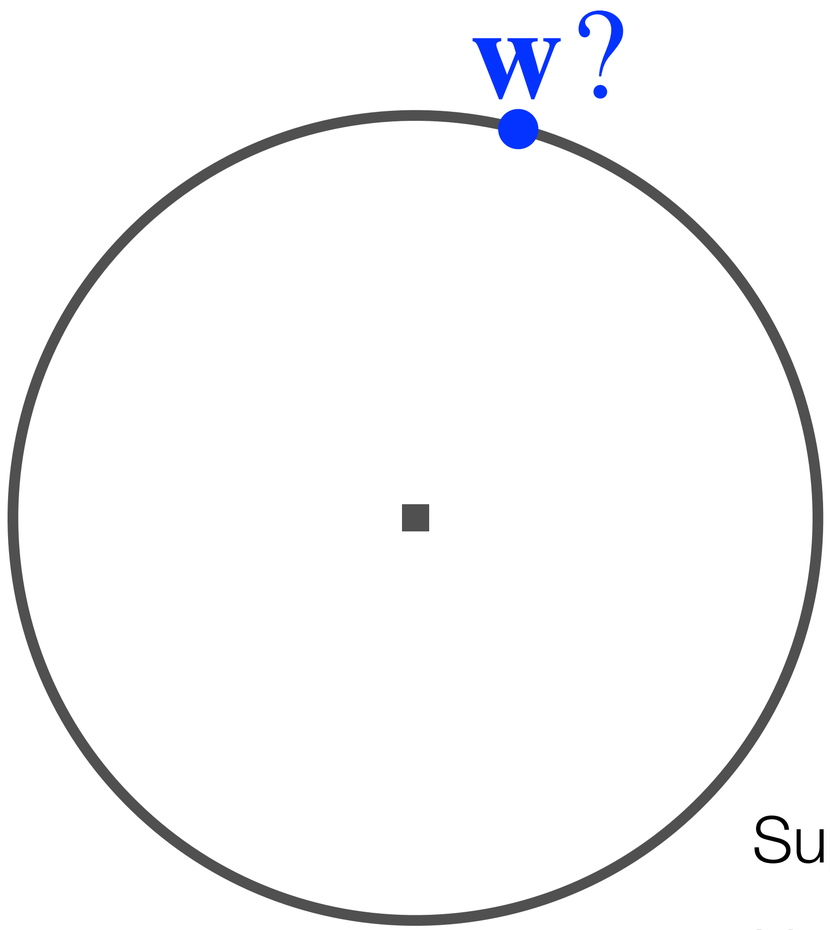
Projection onto Euclidean Ball



Suppose $\|\mathbf{u}\| > r$ hence outside the ball

Need to find closest vector $\mathbf{w}$ such that $\|\mathbf{w}\| \leq r$

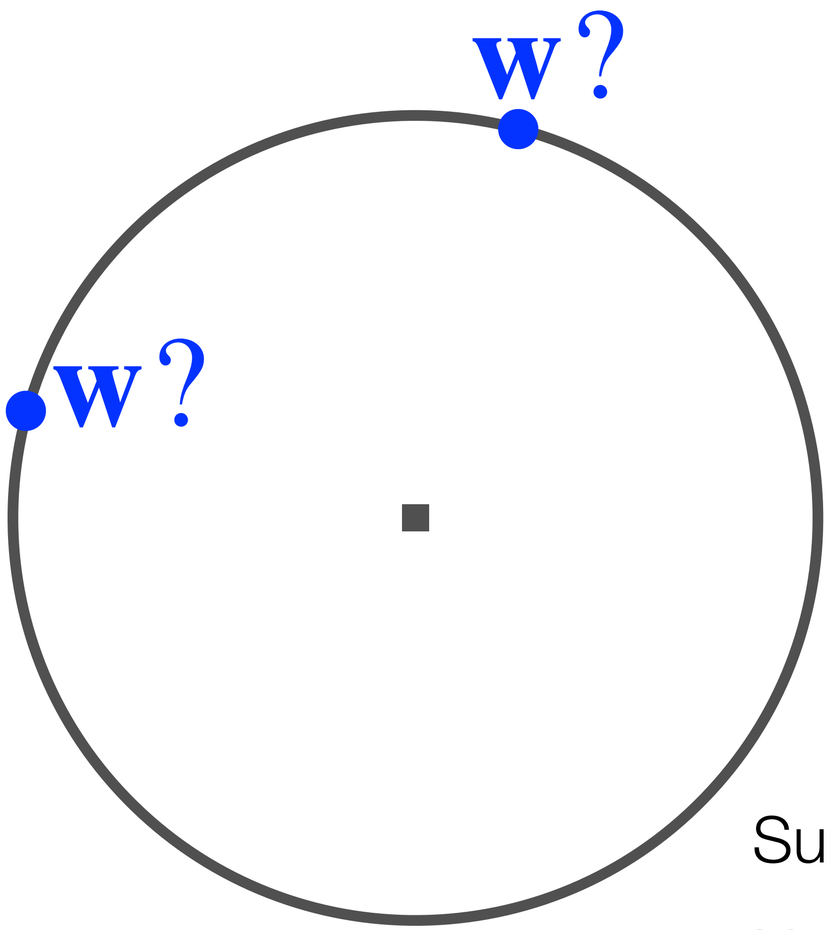
Projection onto Euclidean Ball



Suppose $\|\mathbf{u}\| > r$ hence outside the ball

Need to find closest vector $\mathbf{w}$ such that $\|\mathbf{w}\| \leq r$

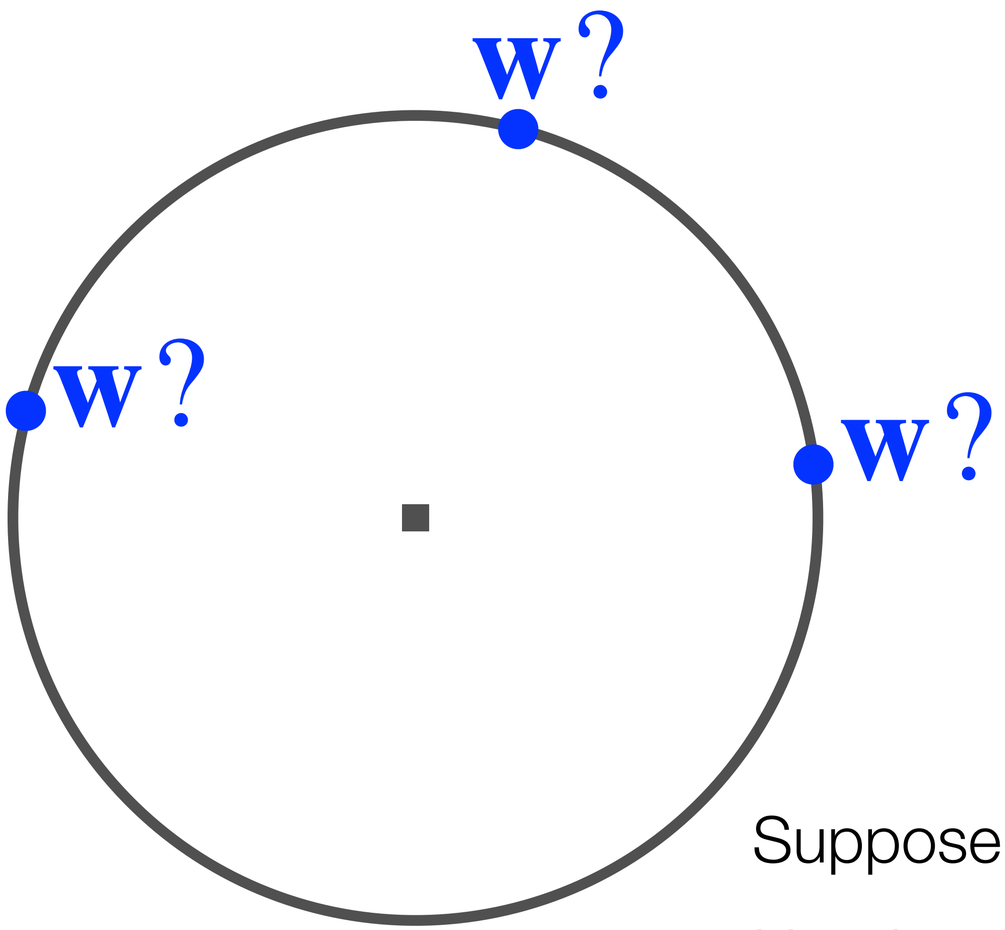
Projection onto Euclidean Ball



Suppose $\|\mathbf{u}\| > r$ hence outside the ball

Need to find closest vector $\mathbf{w}$ such that $\|\mathbf{w}\| \leq r$

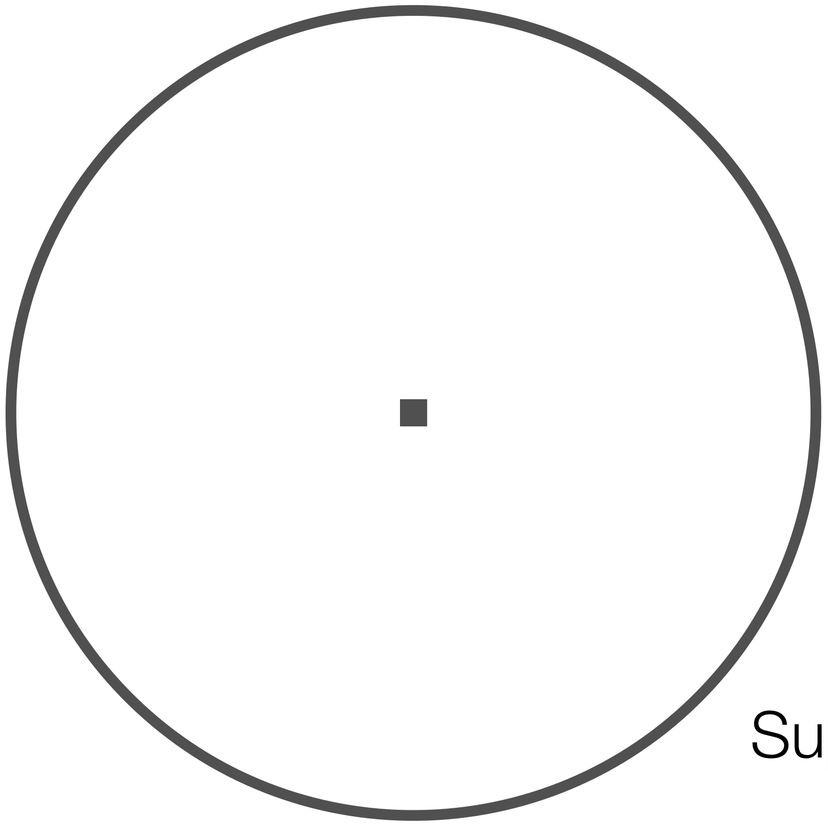
Projection onto Euclidean Ball



Suppose $\|\mathbf{u}\| > r$ hence outside the ball

Need to find closest vector $\mathbf{w}$ such that $\|\mathbf{w}\| \leq r$

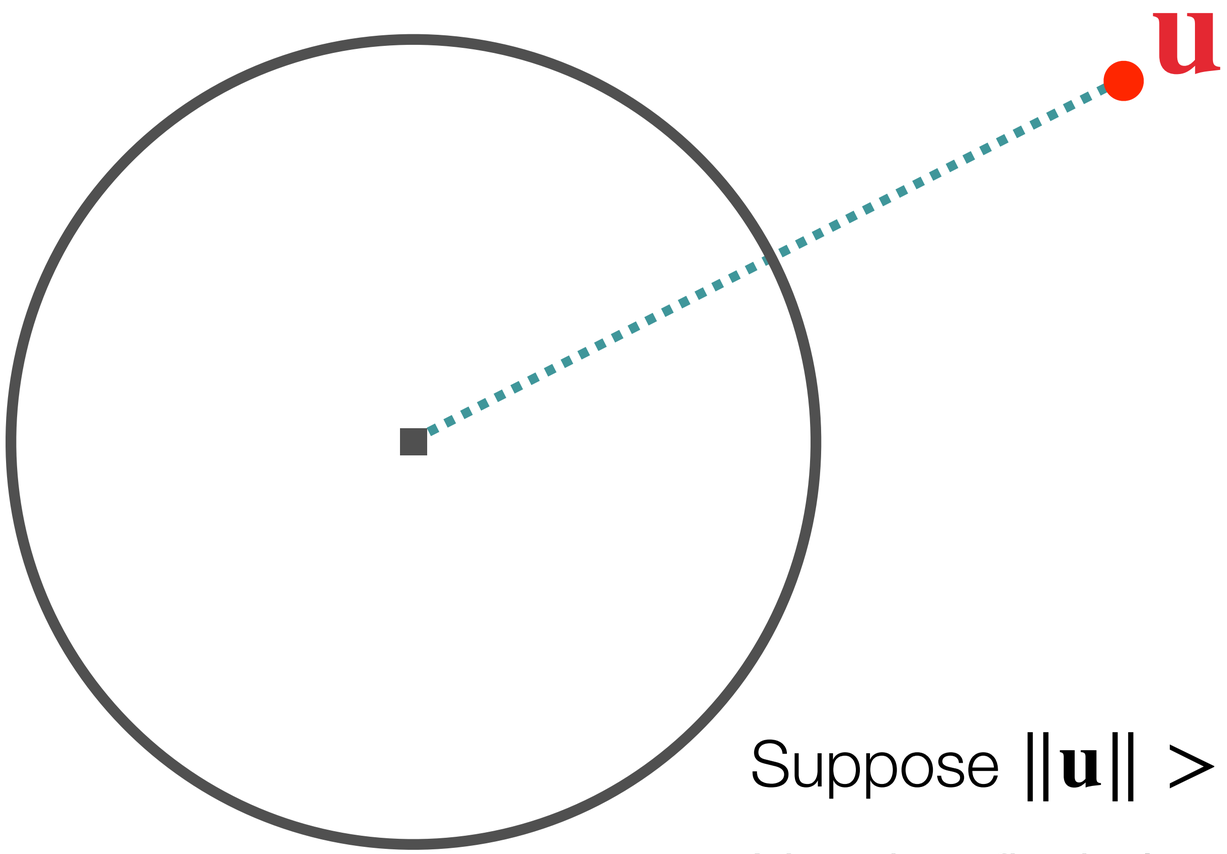
Projection onto Euclidean Ball



Suppose $\|\mathbf{u}\| > r$ hence outside the ball

Need to find closest vector $\mathbf{w}$ such that $\|\mathbf{w}\| \leq r$

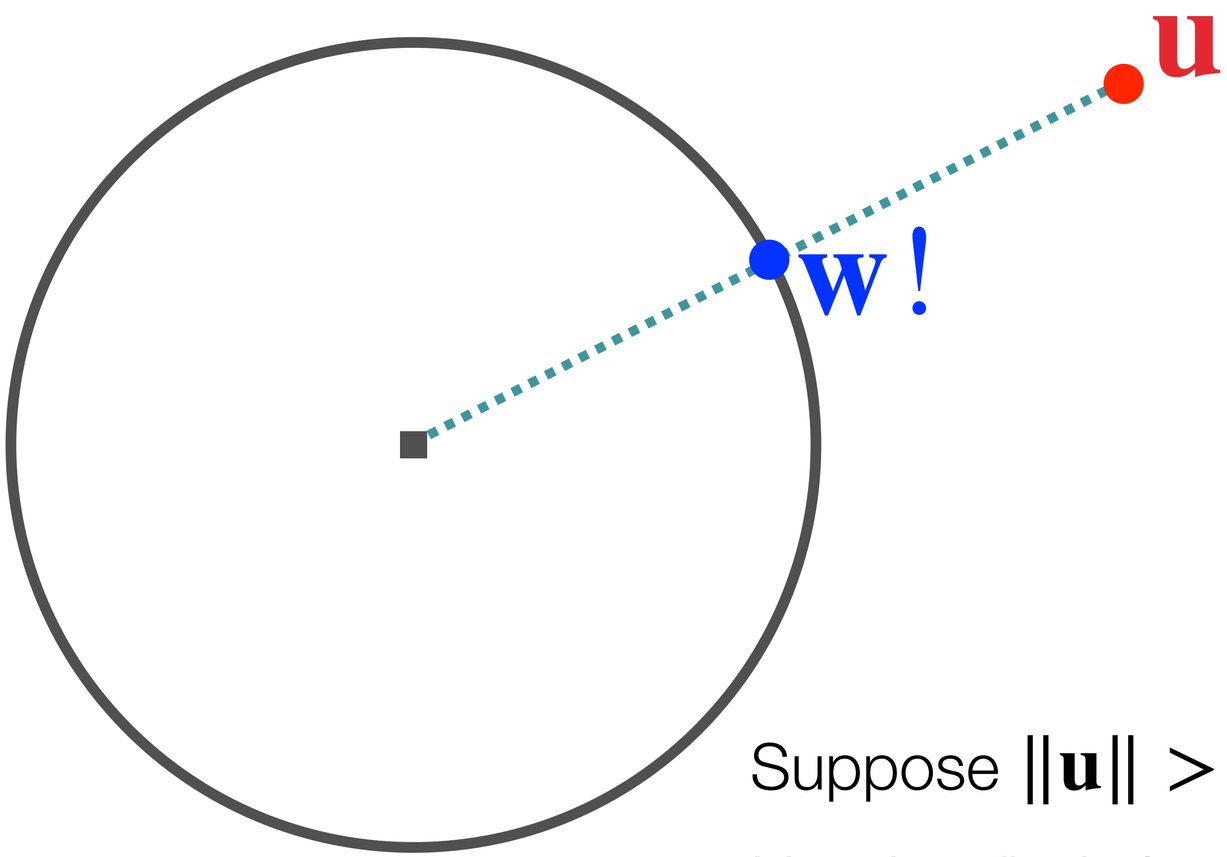
Projection onto Euclidean Ball



Suppose $\|\mathbf{u}\| > r$ hence outside the ball

Need to find closest vector $\mathbf{w}$ such that $\|\mathbf{w}\| \leq r$

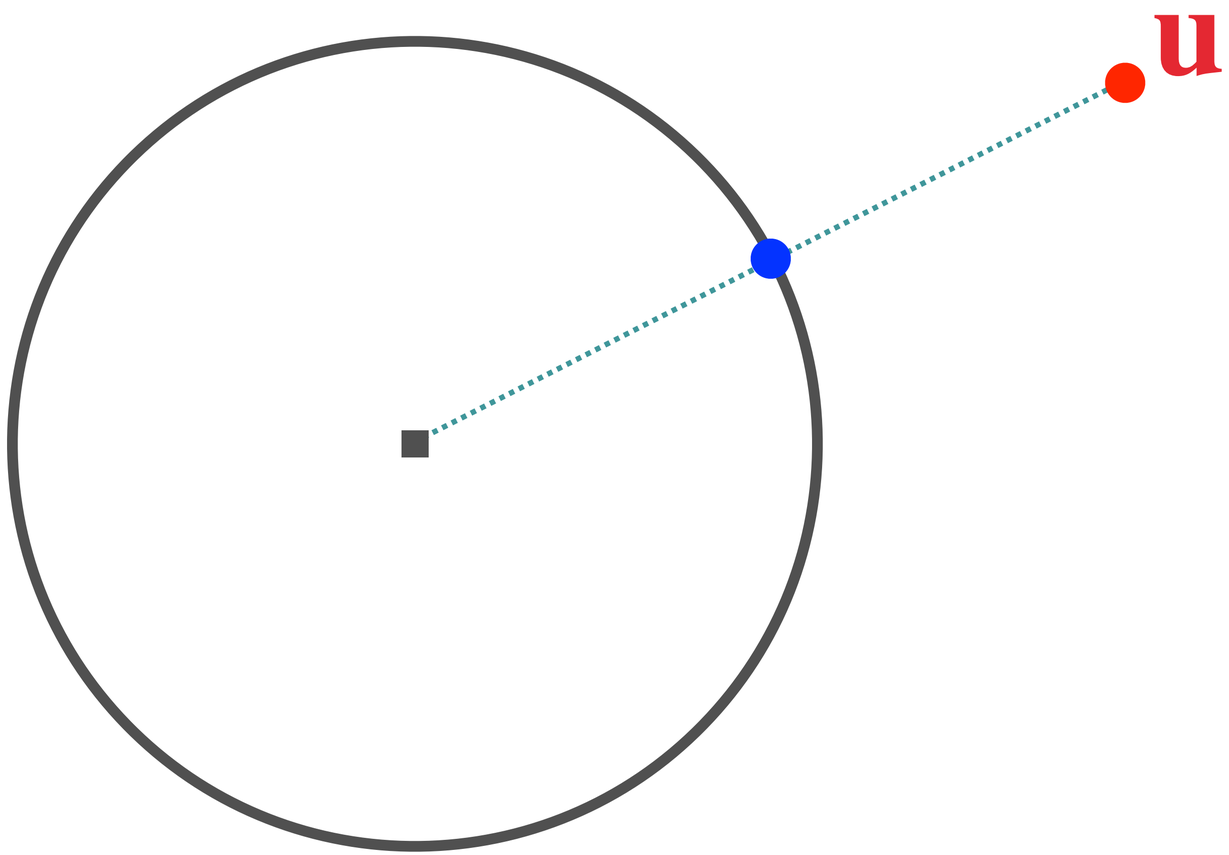
Projection onto Euclidean Ball



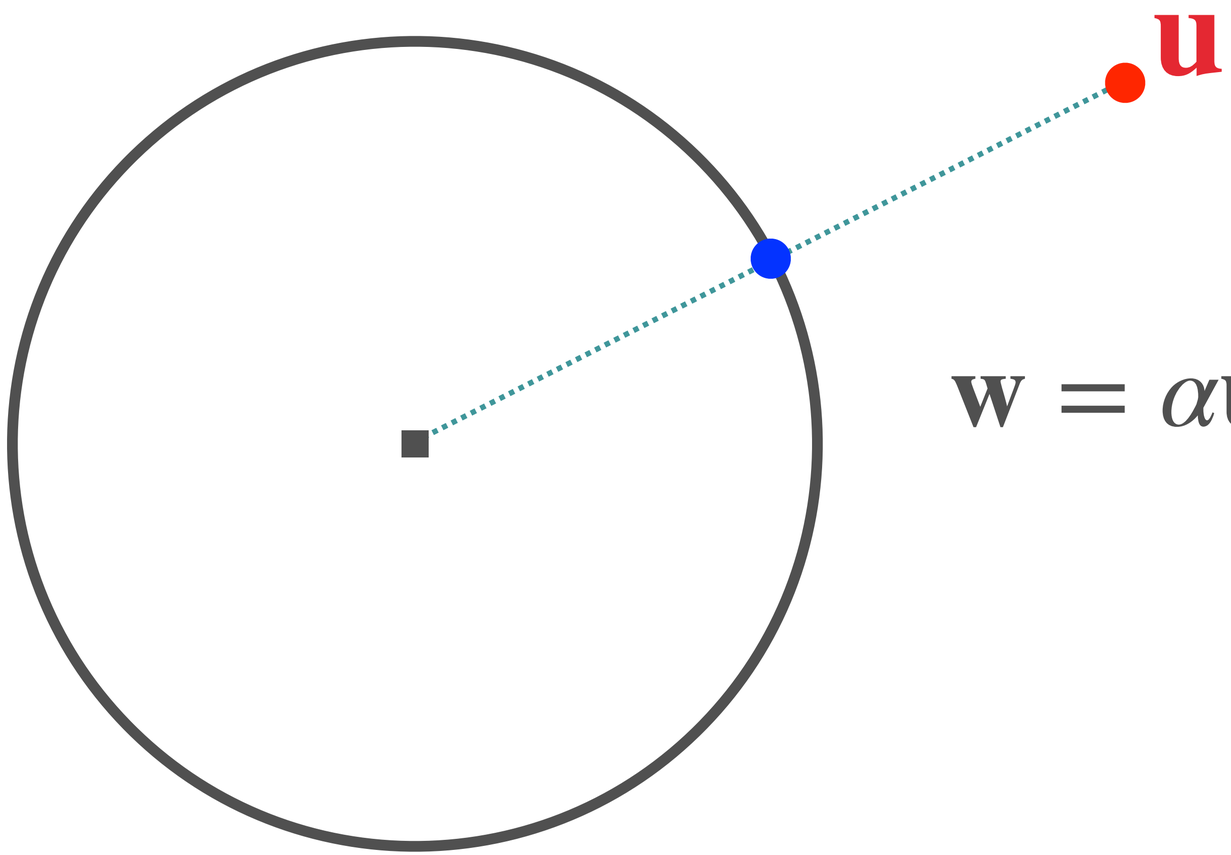
Suppose $\|\mathbf{u}\| > r$ hence outside the ball

Need to find closest vector $\mathbf{w}$ such that $\|\mathbf{w}\| \leq r$

Projection onto Euclidean Ball

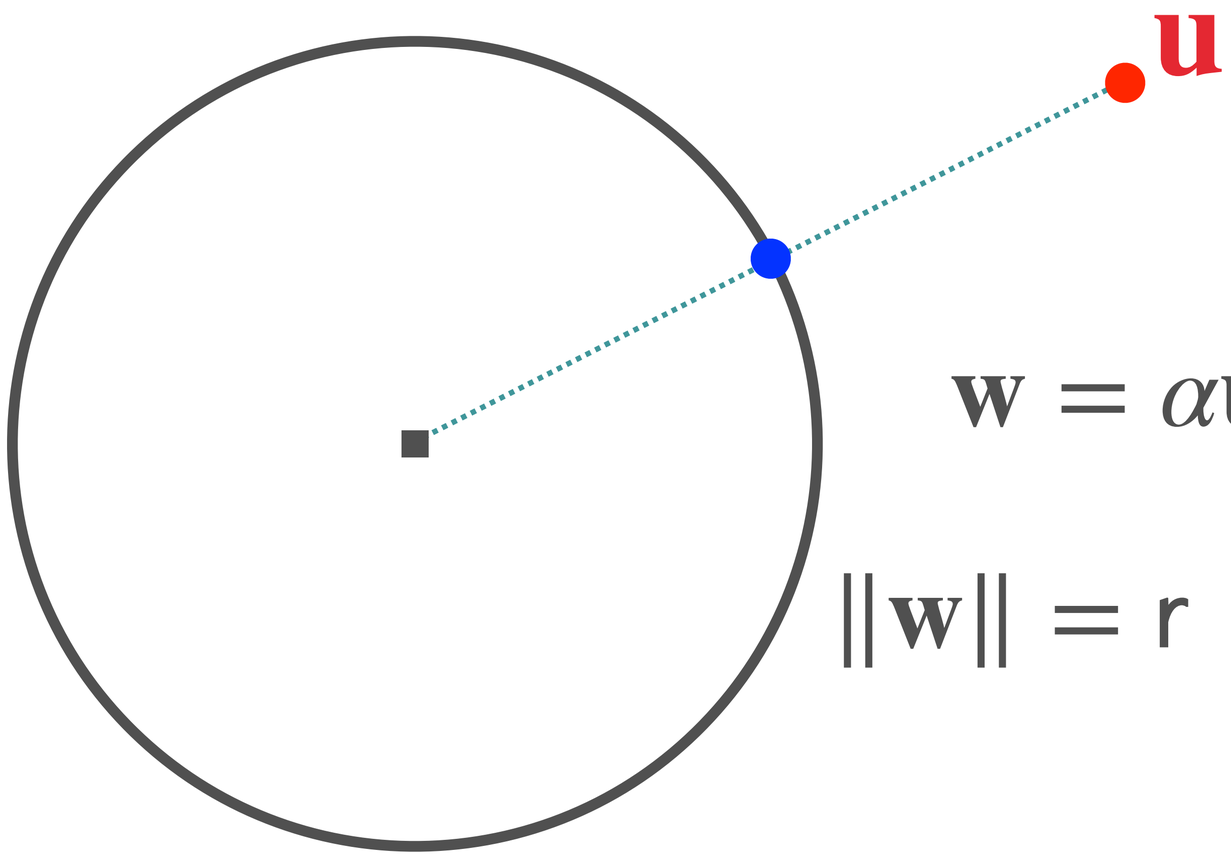


Projection onto Euclidean Ball



$$\mathbf{w} = \alpha \mathbf{u} \; + \; (1 - \alpha)\, \mathbf{0} = \alpha \mathbf{u}$$

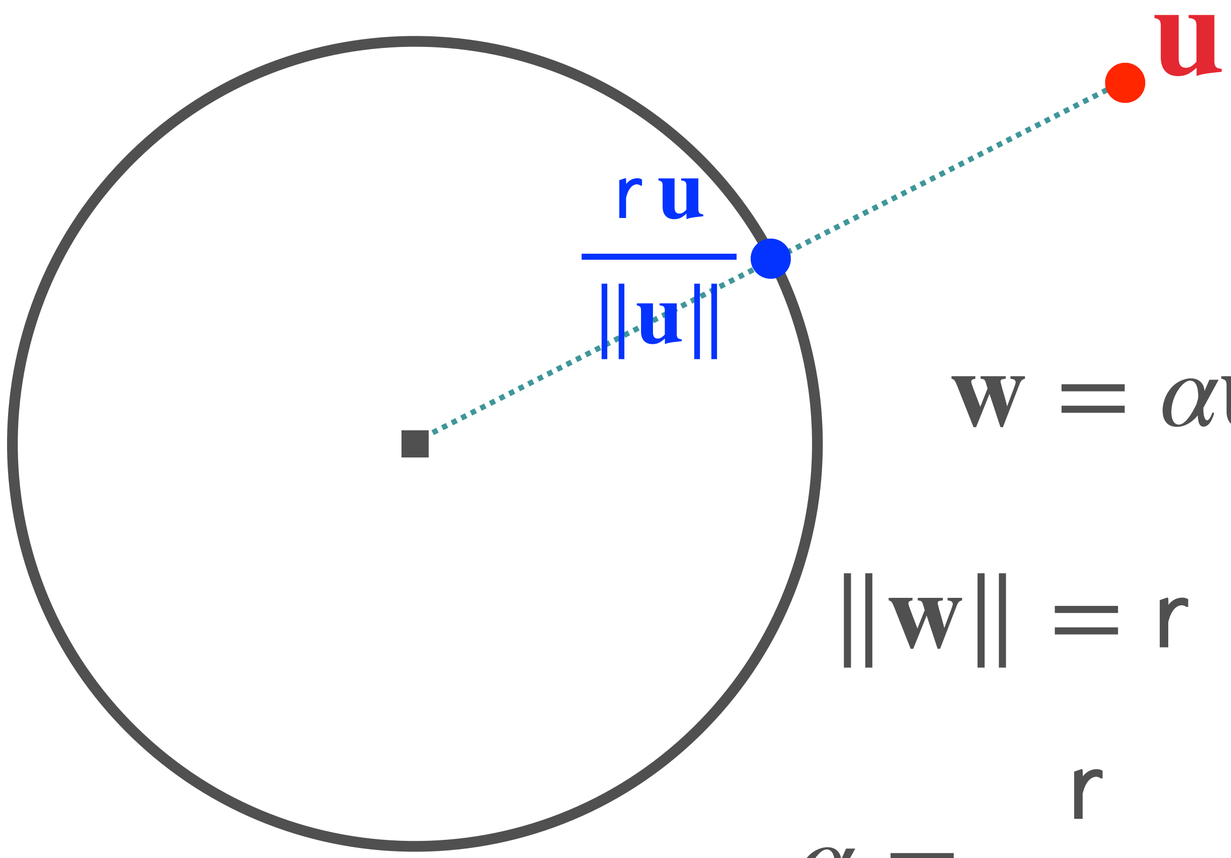
Projection onto Euclidean Ball



$$\mathbf{w} = \alpha \mathbf{u} \; + \; (1 - \alpha)\, \mathbf{0} = \alpha \mathbf{u}$$

$$\|\mathbf{w}\| = \mathsf{r} \quad \Rightarrow \quad \|\alpha \mathbf{u}\| = \alpha \|\mathbf{u}\| = \mathsf{r}$$

Projection onto Euclidean Ball

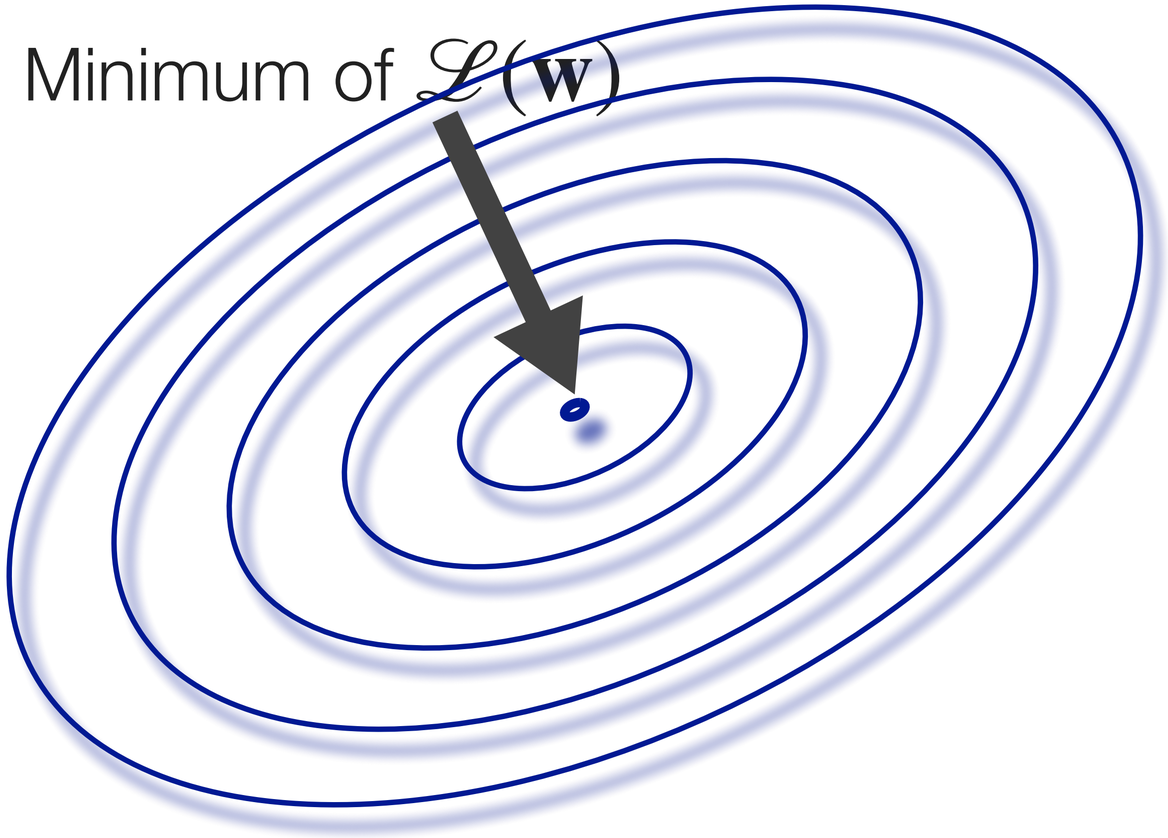


$$\mathbf{w} = \alpha\mathbf{u} + (1-\alpha)\mathbf{0} = \alpha\mathbf{u}$$

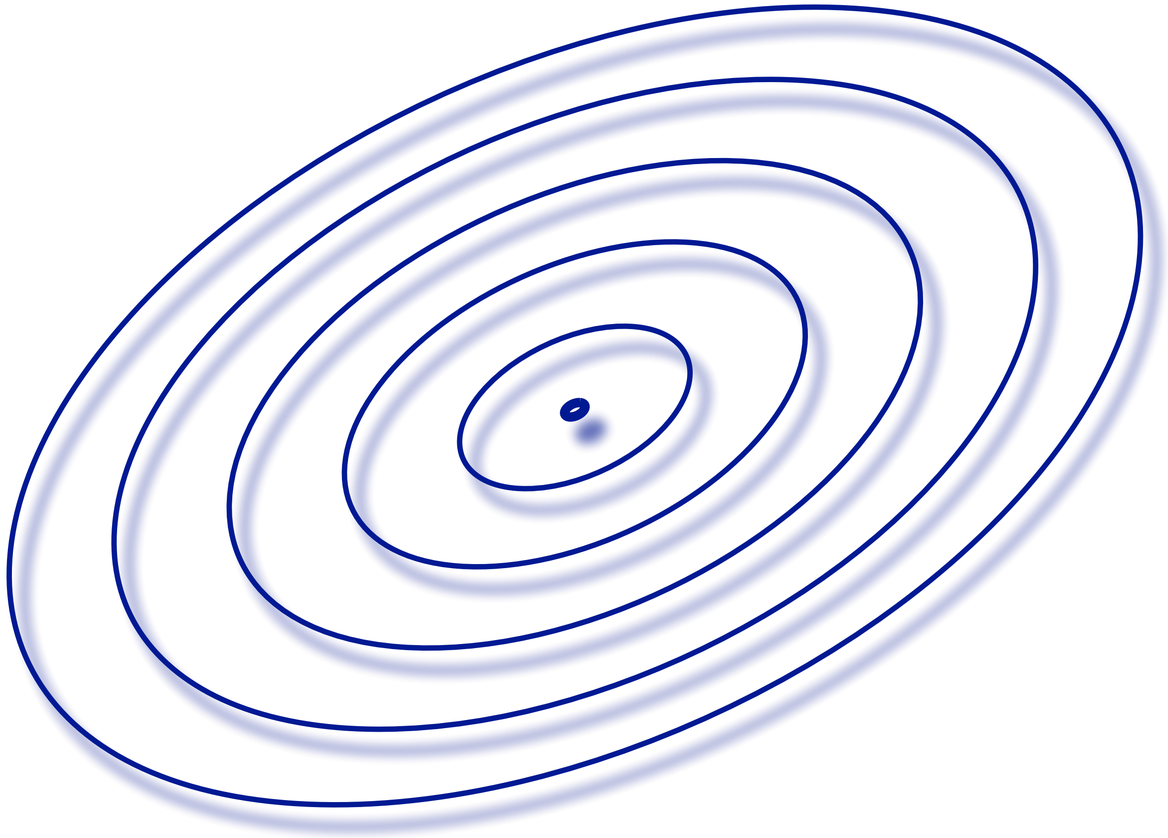
$$\|\mathbf{w}\| = \mathsf{r} \quad \Rightarrow \quad \|\alpha\mathbf{u}\| = \alpha\|\mathbf{u}\| = \mathsf{r}$$

$$\alpha = \frac{\mathsf{r}}{\|\mathbf{u}\|} \quad \Rightarrow \quad \mathbf{w} = \frac{\mathsf{r}\,\mathbf{u}}{\|\mathbf{u}\|}$$

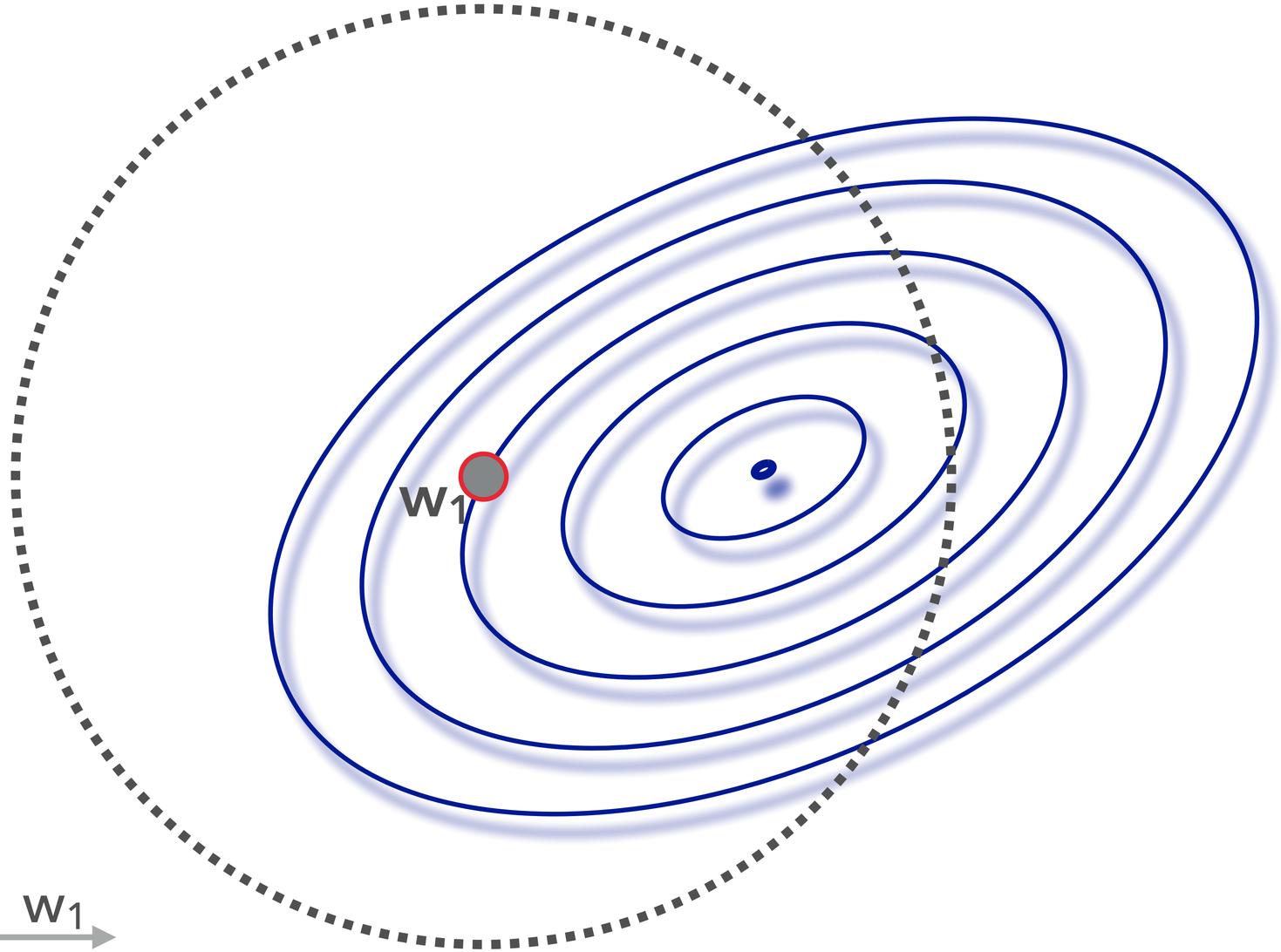
SGD
+
Projections



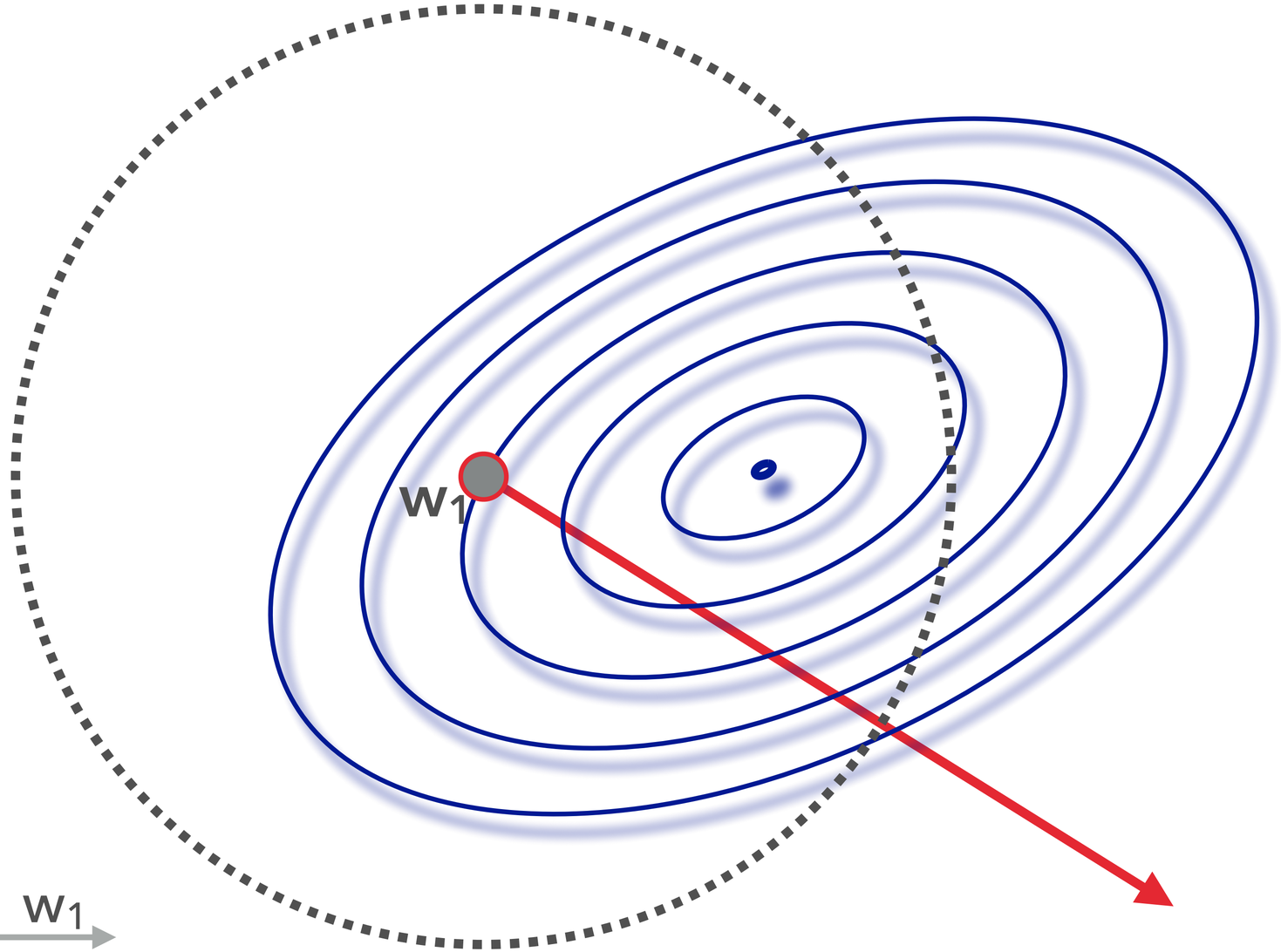
SGD
+
Projections



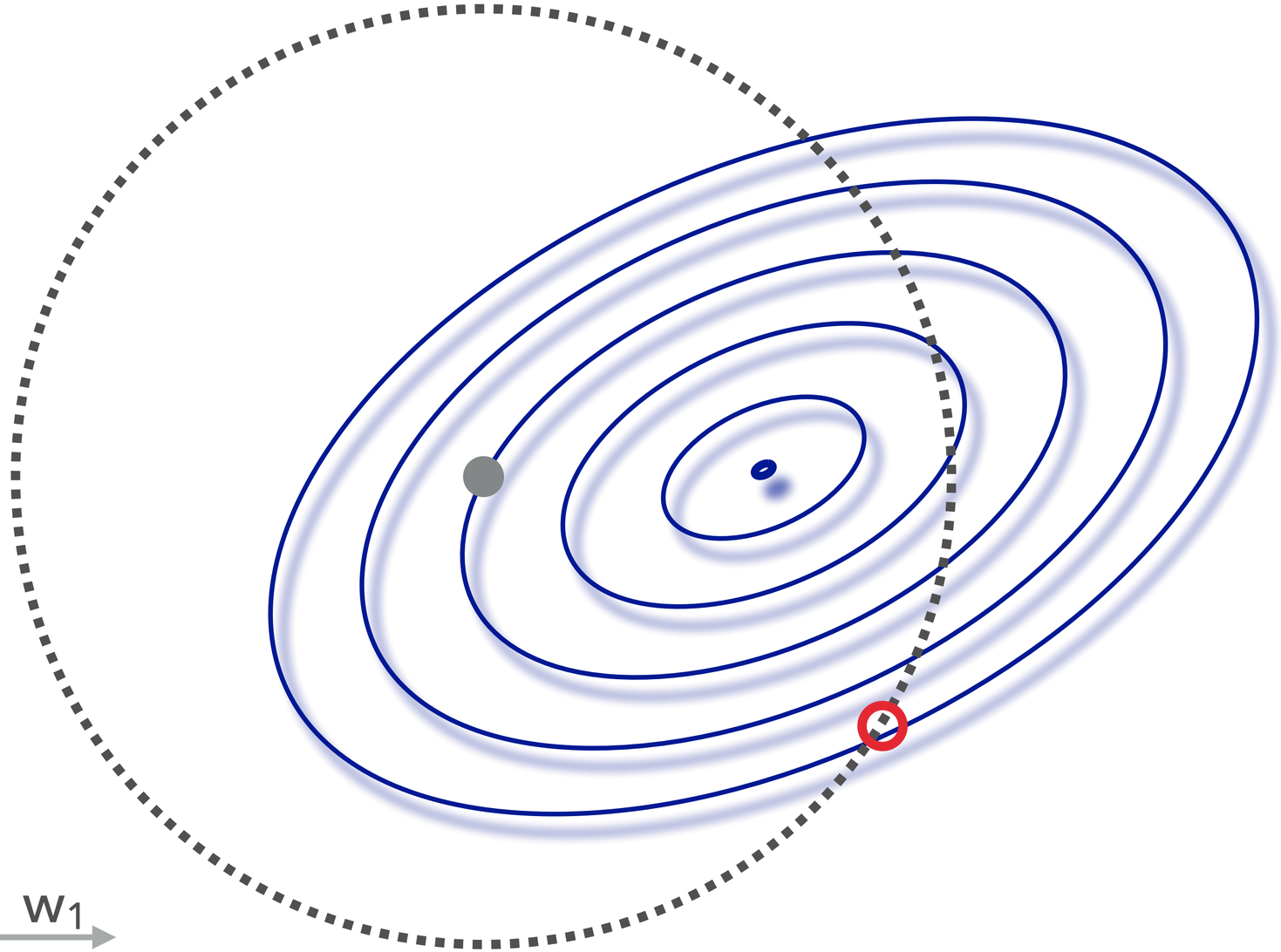
SGD + Projections



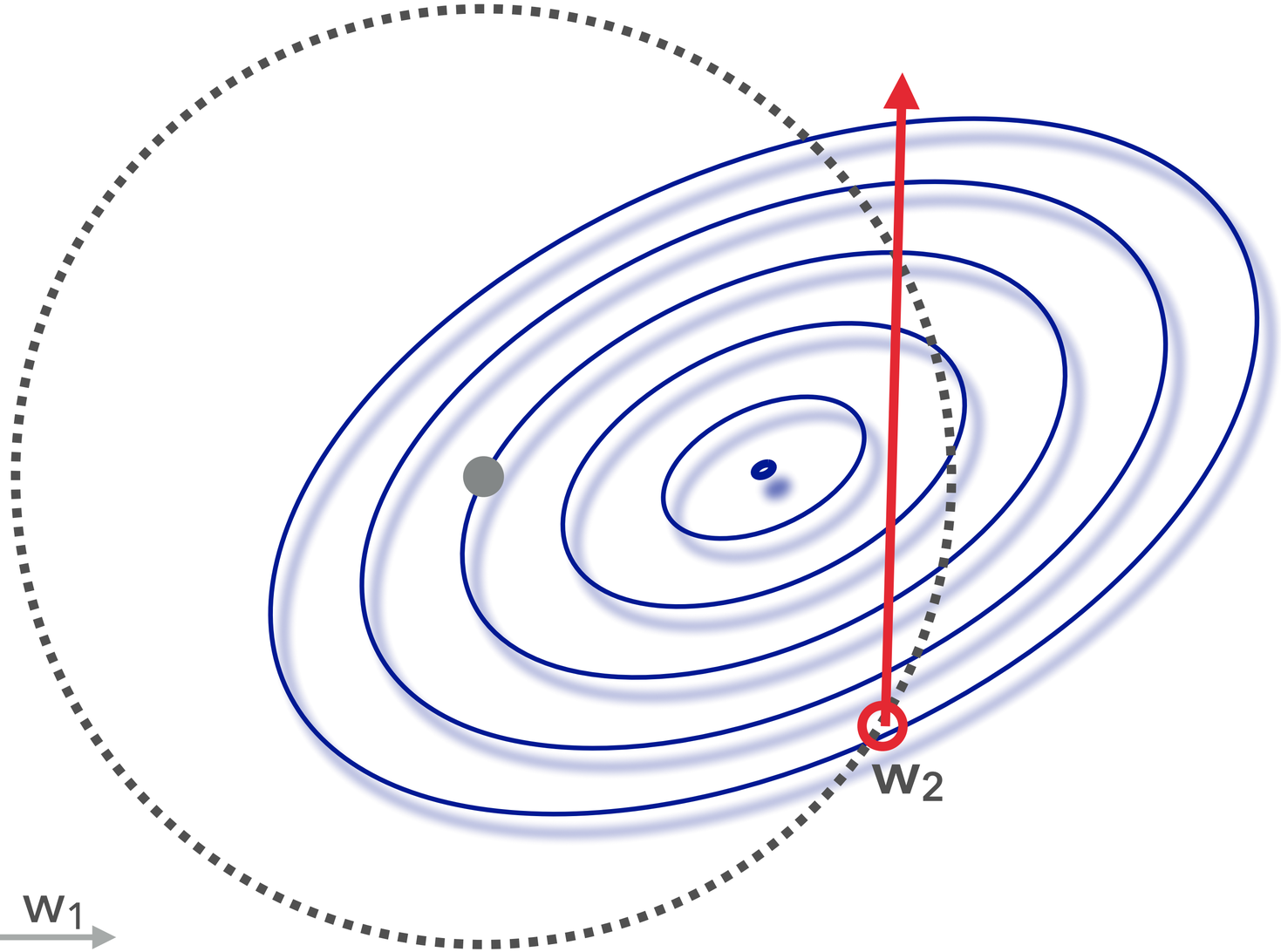
SGD + Projections



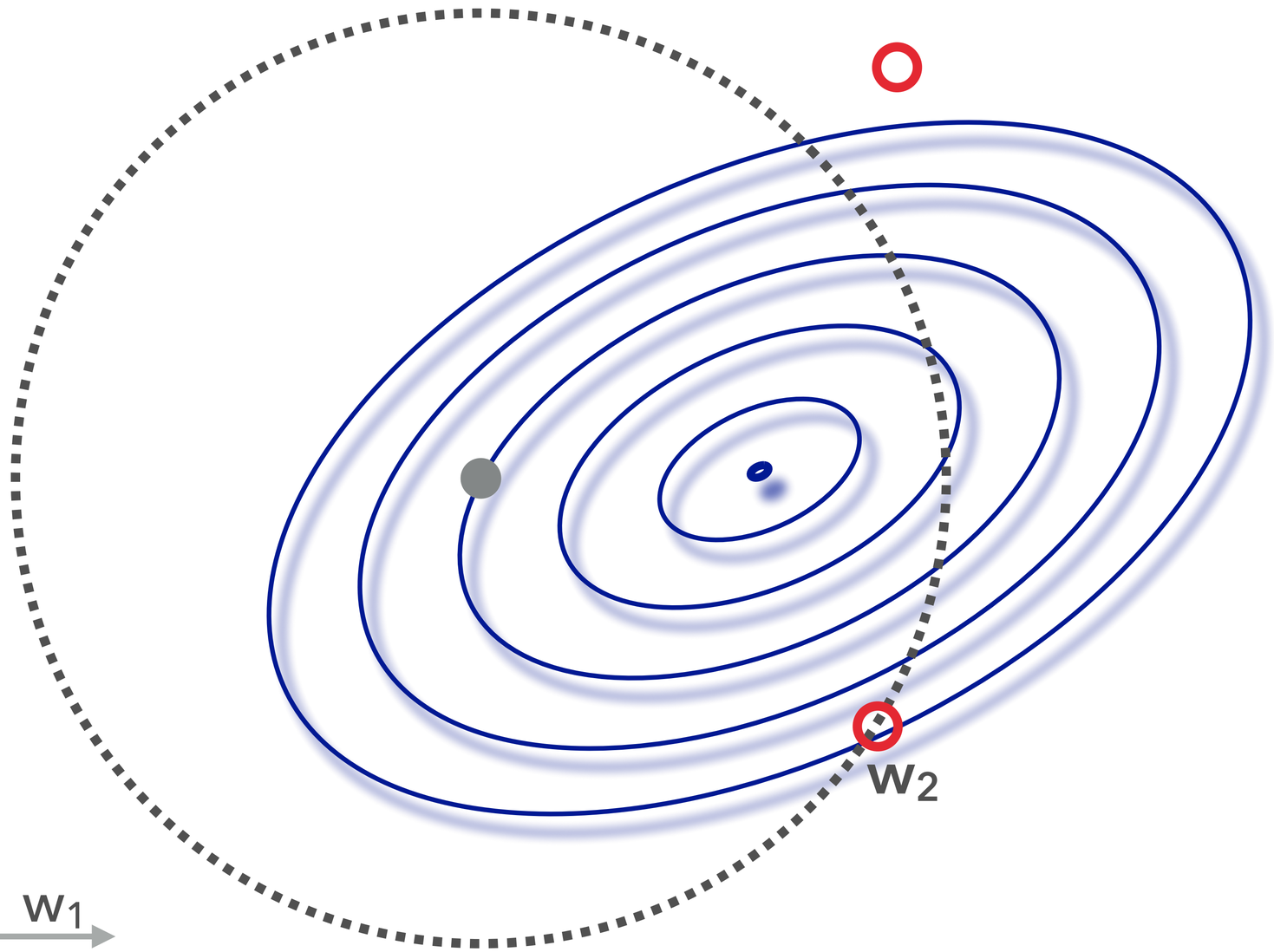
SGD + Projections



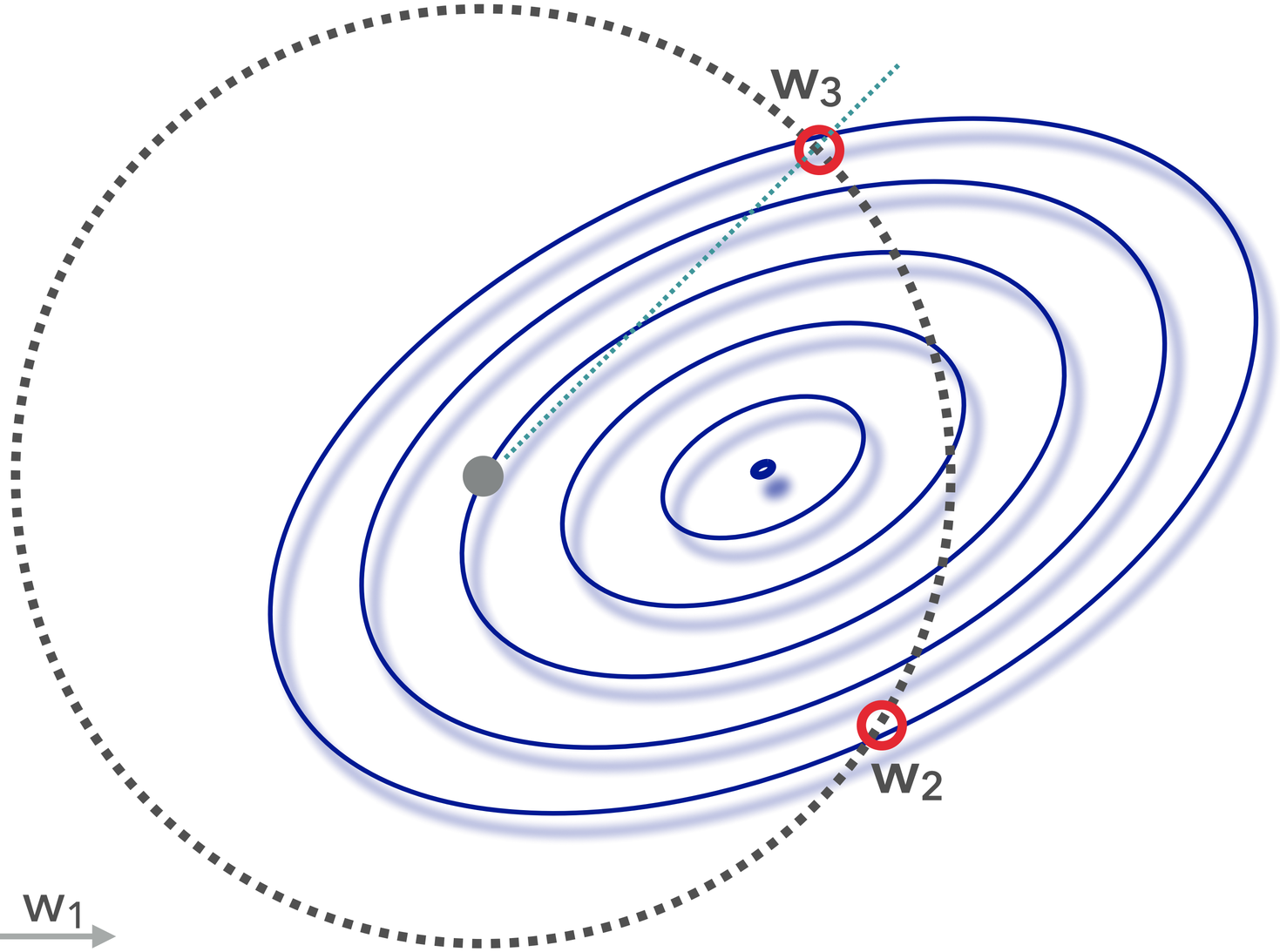
SGD + Projections



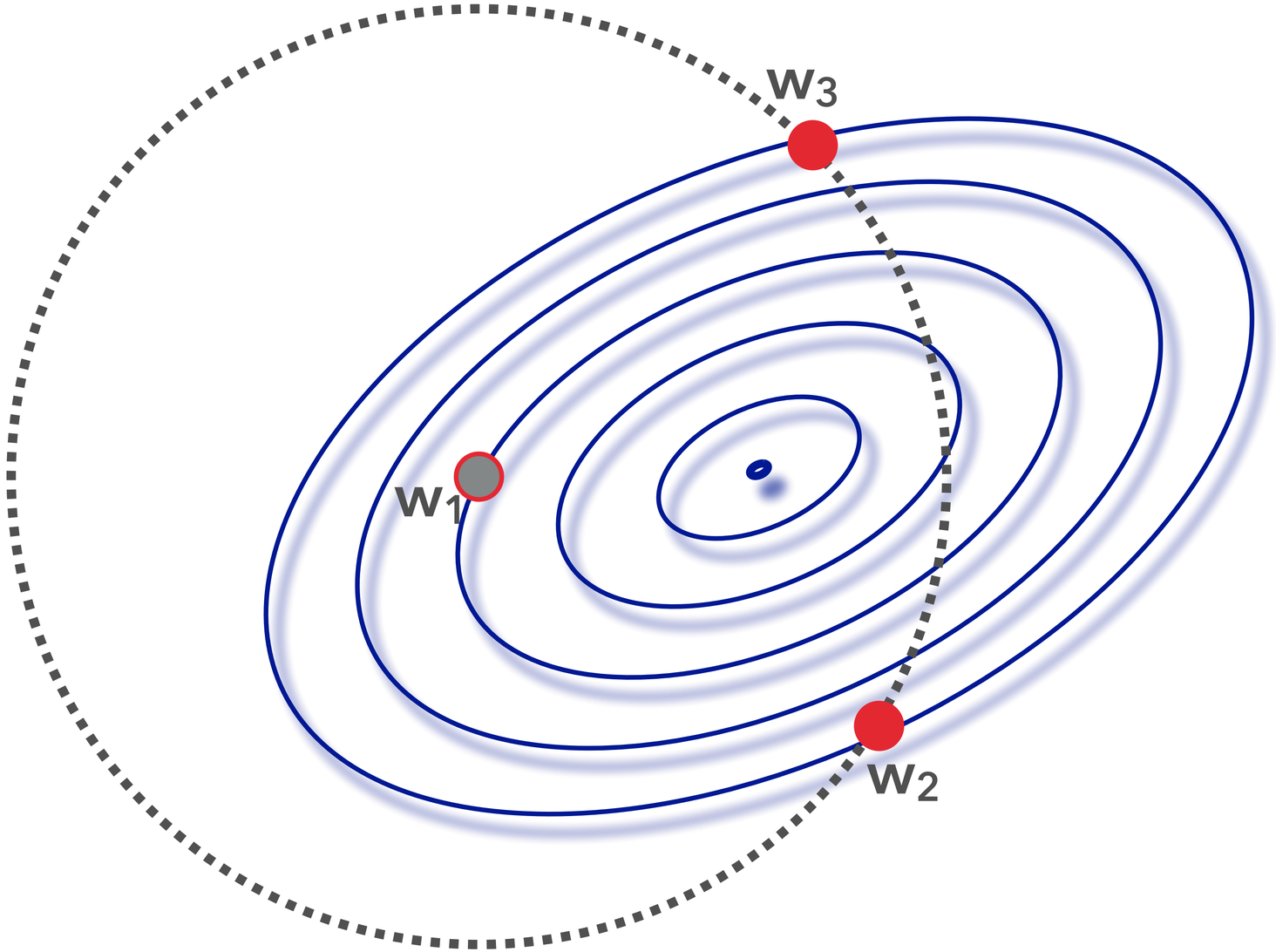
SGD + Projections



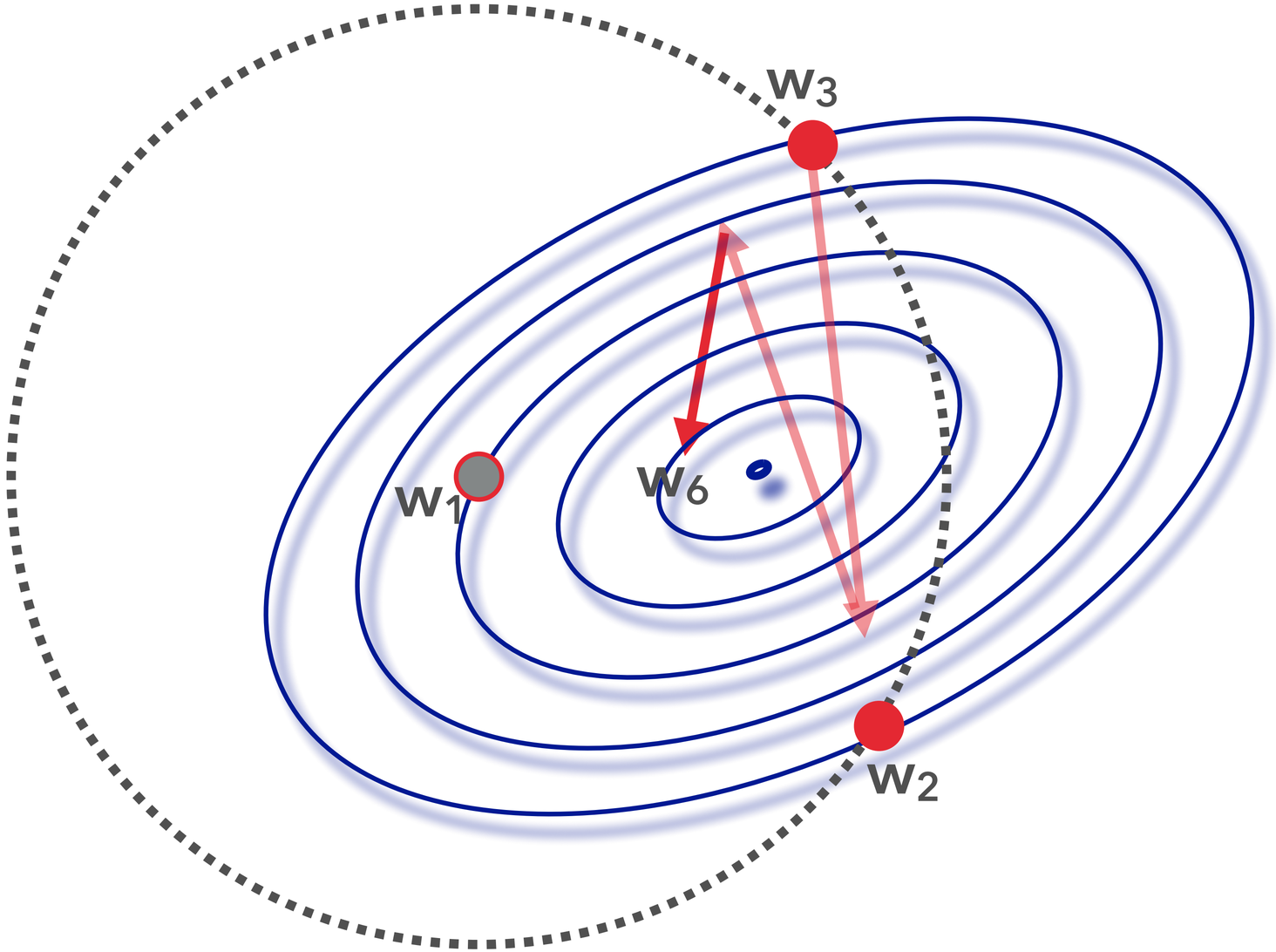
SGD + Projections



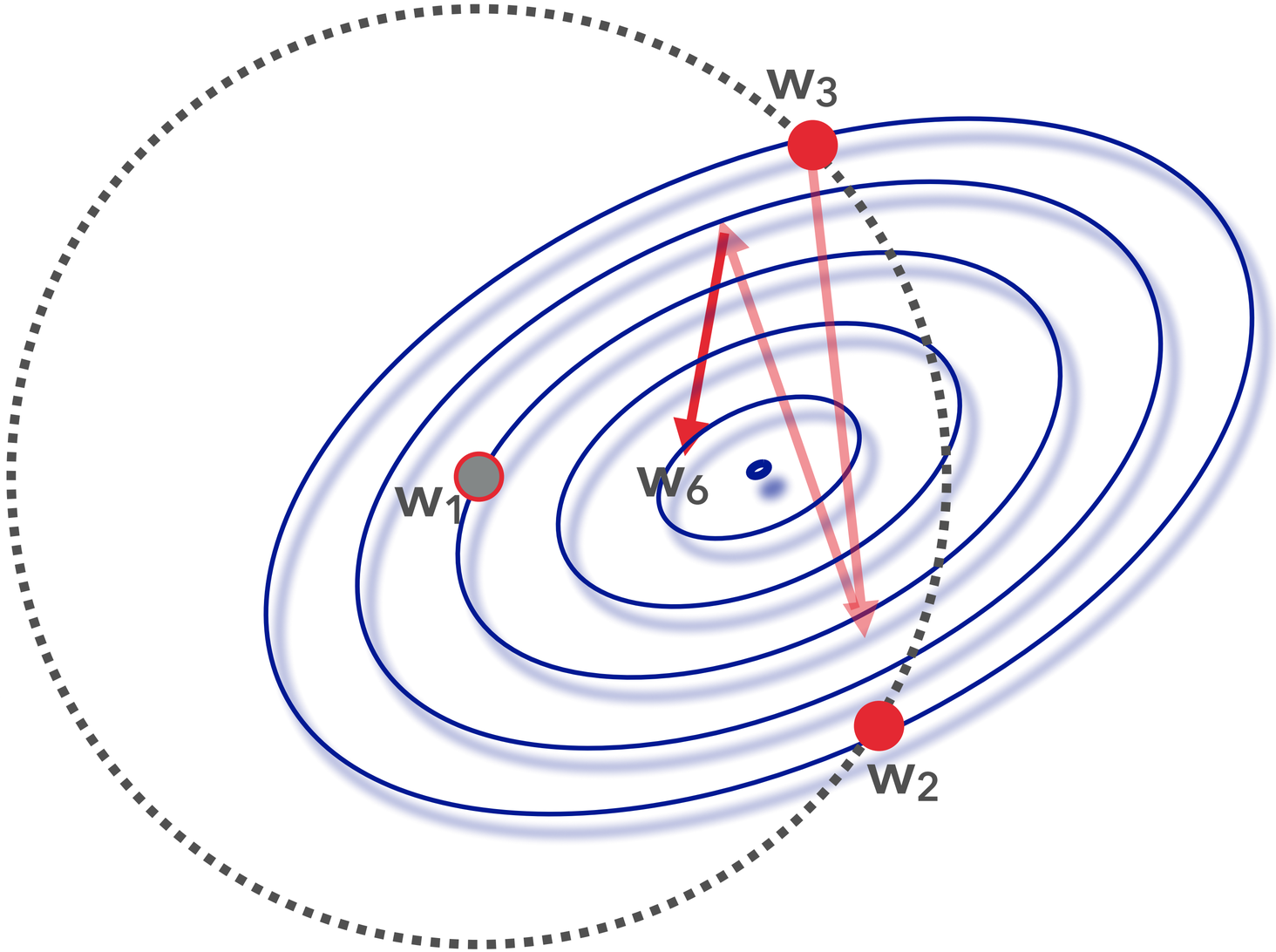
SGD + Projections



SGD + Projections



SGD + Projections



When learning rate too large
projection onto a ball
could prevent divergence

SGD + Projection & Averaging

```
def SGD(w0, ns, grad, eta, alph, rad):
  T, d = len(eta) + 1, len(w0)
  w, u = w0, w0   // u is averaged w
  for t in range(1, T):
     S  = random_set(ns)
     gs = sgrad(w, S)
     w = project(w - eta[t-1] * gs, rad)
     u = average(u, w, alph)
  return u
```

```
def project(w, rad):
  nr = np.sqrt(np.sum(w * w))
  s = min(1, rad / nr)
  return s* w
```

```
def average(u, w, alph):
  return (1-alph) * u + alph * w
```

	GD	SGD
Convergence	Guaranteed	In Expectation
Cost of Update	**O(nd)**	**O(|S|d)**
Learning Rate	Can be fixed	Decreasing
Termination	Grad Norm	Multiple Iterates
Output	Last $\mathbf{w}_T$	Averaged

Generalized Linear Models

Loss for linear predictors $\mathscr{L}(\mathbf{w}) = \frac{1}{n}\sum_{i=1}^{n} \ell(y_i, h(\mathbf{w} \cdot \mathbf{x_i}))$

h: transfer (activation) function $h : \mathbf{R} \rightarrow \mathbf{R}$ and $\hat{y}_i = h(\mathbf{w} \cdot \mathbf{x_i})$

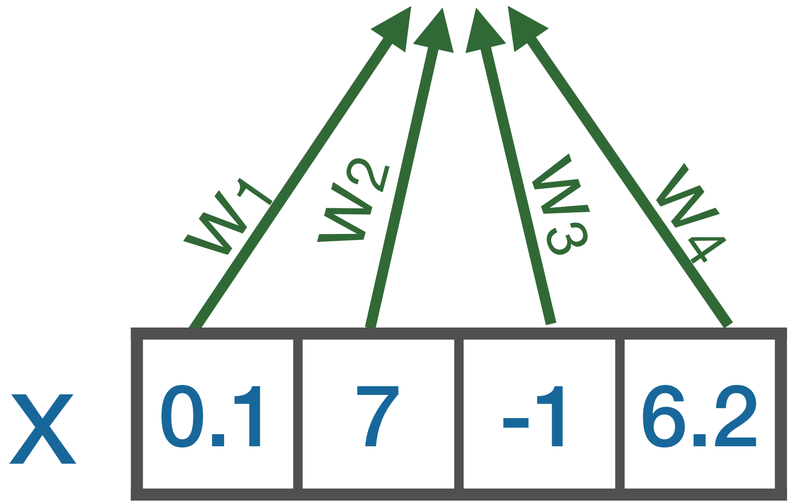
$\ell(\mathbf{y}, \hat{\mathbf{y}})$ binary function from to $\mathbf{R} \times \mathbf{R}$ to $\mathbf{R}_+$

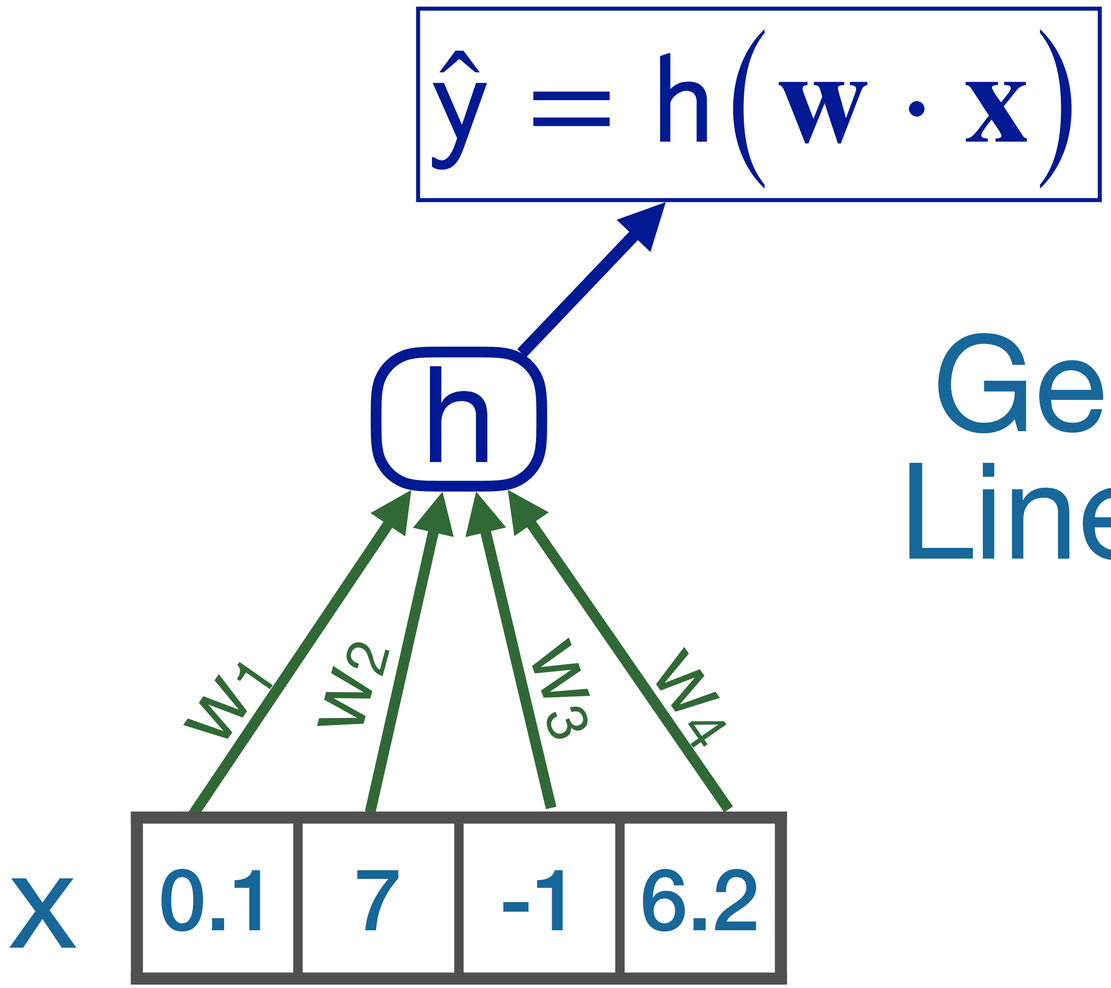
We can find $\mathbf{w}$ which (approximately) minimizes $\mathscr{L}(\mathbf{w})$ using GD and computing the gradient using chain rule

Generalized Linear Model

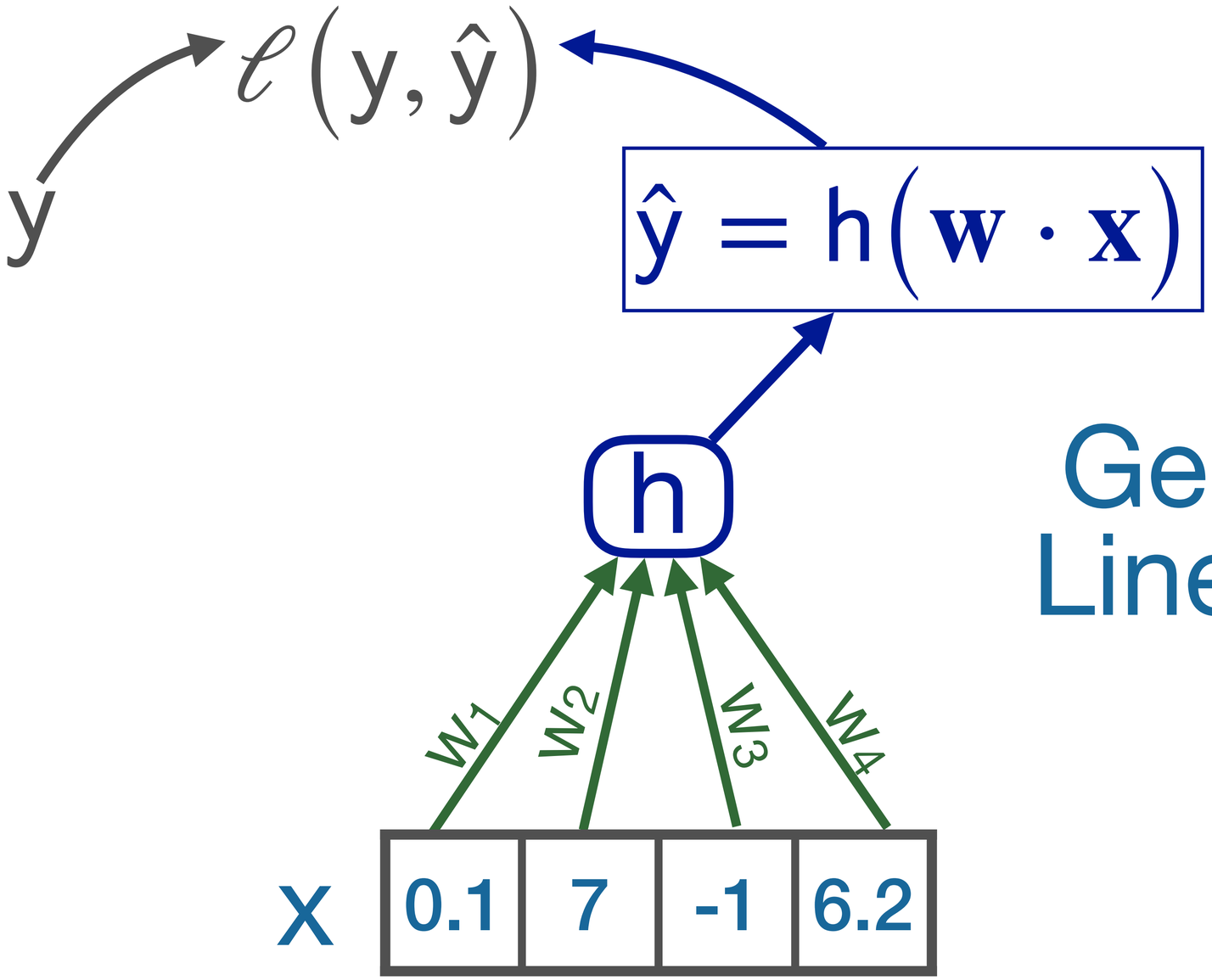
Generalized Linear Model

Generalized Linear Model





Generalized Linear Model



Generalized Linear Model

Gradients for GLM

$$\nabla \mathscr{L}(\mathbf{w}) = \frac{1}{n} \sum_{i=1}^{n} \nabla_{\mathbf{w}} \, \ell(y_i, h(\mathbf{w} \cdot \mathbf{x_i}))$$

Given y_i derivative w.r.t $\hat{y}_i$ is $\ell'(y_i, \hat{y}_i) = \dfrac{d\, \ell(y_i, \hat{y}_i)}{d\hat{y}_i}$

Define $z_i = \mathbf{w} \cdot \mathbf{x}_i$ then derivative of h w.r.t. z_i is $h'(z_i)$

Finally $\nabla_{\mathbf{w}} \, z_i = \mathbf{x}_i$

Thus $\nabla \mathscr{L}(\mathbf{w}) = \frac{1}{n} \sum_{i=1}^{n} \ell'(y_i, \hat{y}_i) \, h'(z_i) \, \mathbf{x_i}$

Gradients for GLM

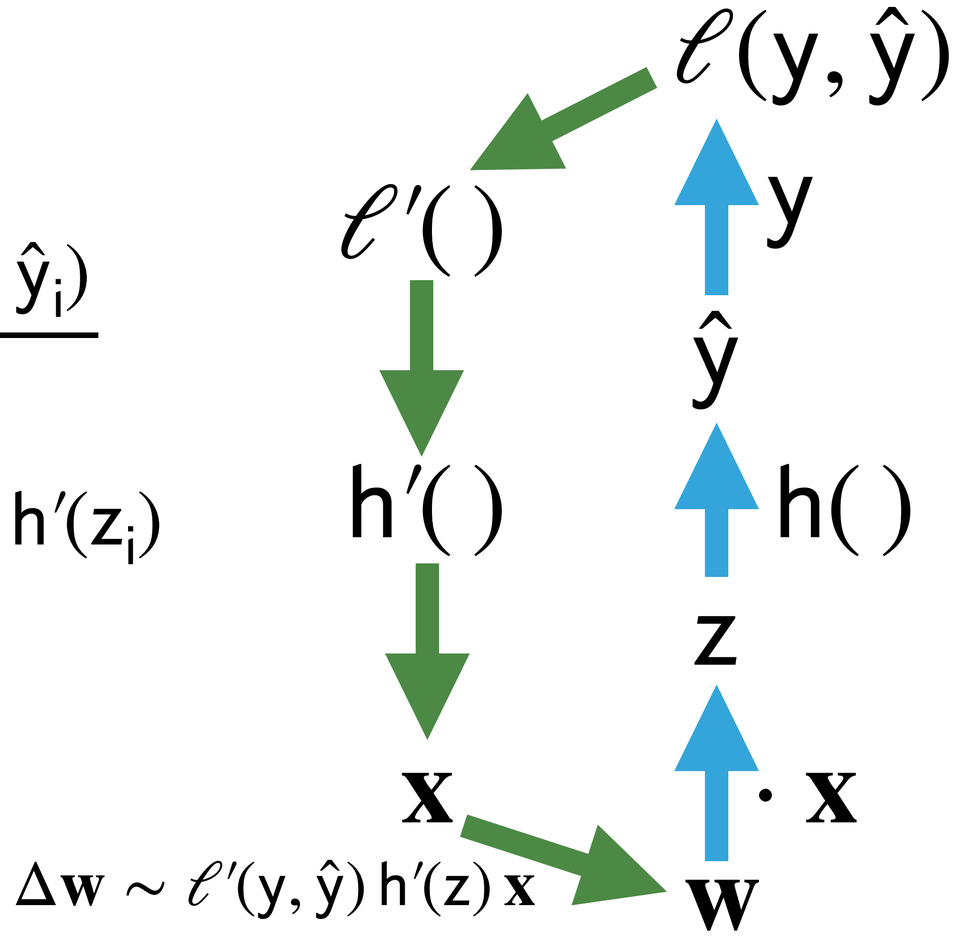
$$\nabla \mathscr{L}(\mathbf{w}) = \frac{1}{n} \sum_{i=1}^{n} \nabla_{\mathbf{w}} \, \ell(y_i, h(\mathbf{w} \cdot \mathbf{x_i}))$$

Given y_i derivative w.r.t $\hat{y}_i$ is $\ell'(y_i, \hat{y}_i) = \dfrac{d\,\ell(y_i, \hat{y}_i)}{d\hat{y}_i}$

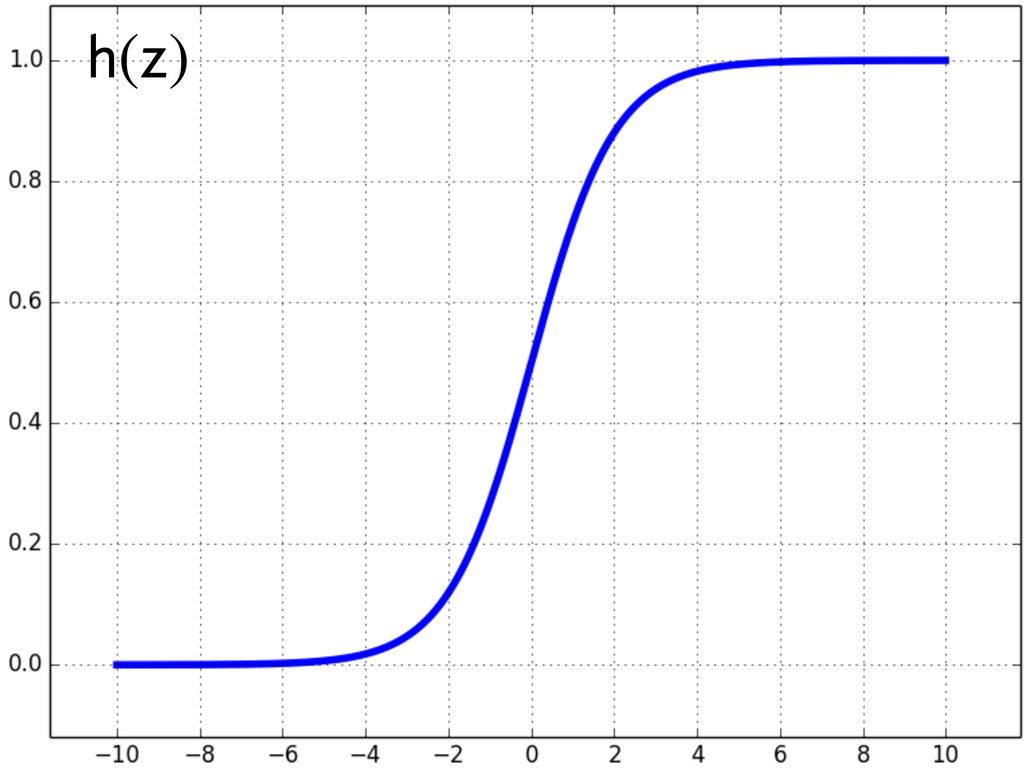
Define $z_i = \mathbf{w} \cdot \mathbf{x}_i$ then derivative of h w.r.t. z_i is $h'(z_i)$

Finally $\nabla_{\mathbf{w}} \, z_i = \mathbf{x}_i$

Thus $\nabla \mathscr{L}(\mathbf{w}) = \dfrac{1}{n} \sum_{i=1}^{n} \ell'(y_i, \hat{y}_i)\, h'(z_i)\, \mathbf{x_i}$

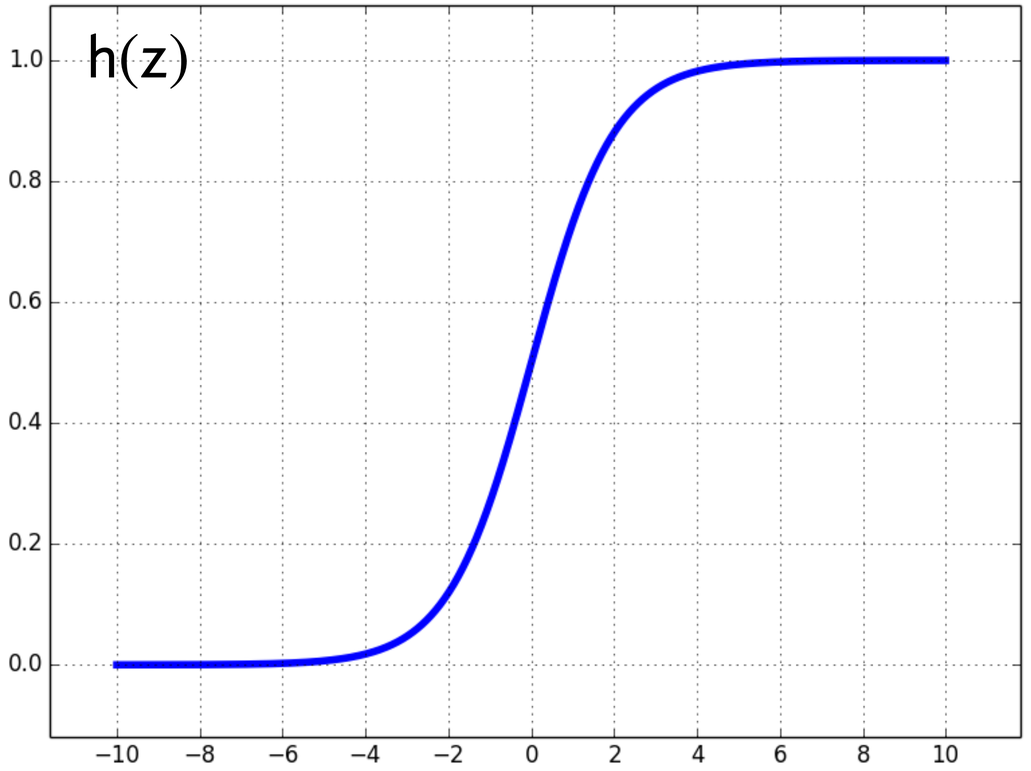


Logistic Regression



Logistic Regression

Labels: $y \in \{0, 1\}$ Targets: $\hat{y} \in [0, 1]$



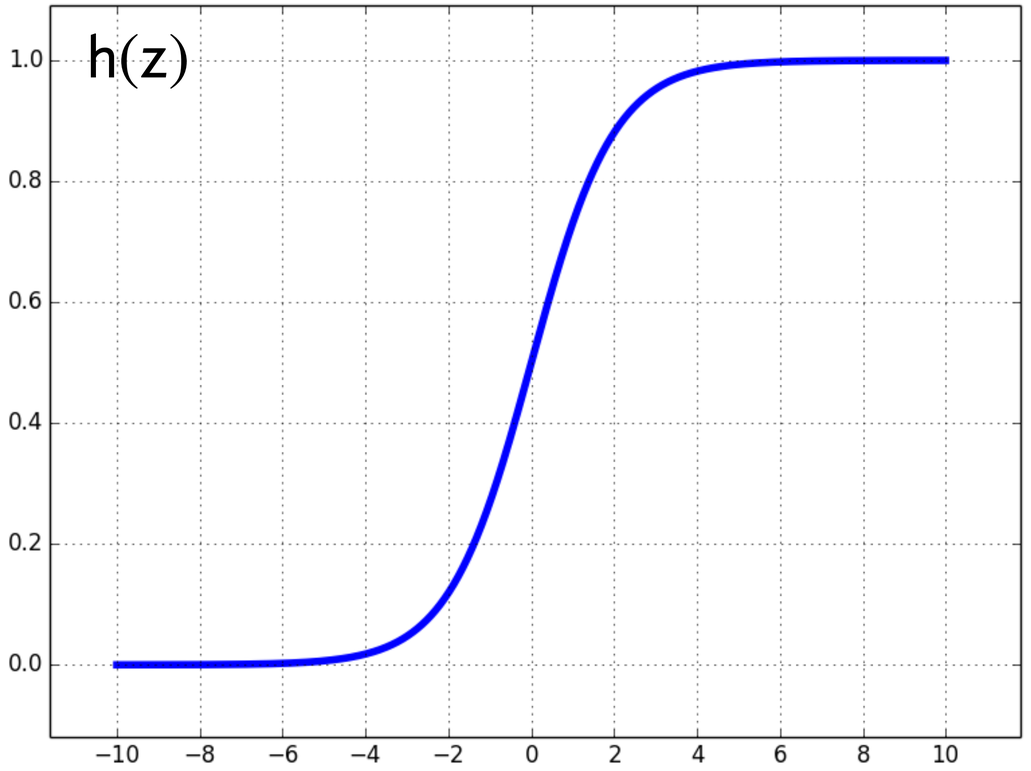
Logistic Regression

Labels: $y \in \{0, 1\}$ Targets: $\hat{y} \in [0, 1]$

Interpretation:

$\hat{y}$ probability estimate that outcome is 1

y actual outcome



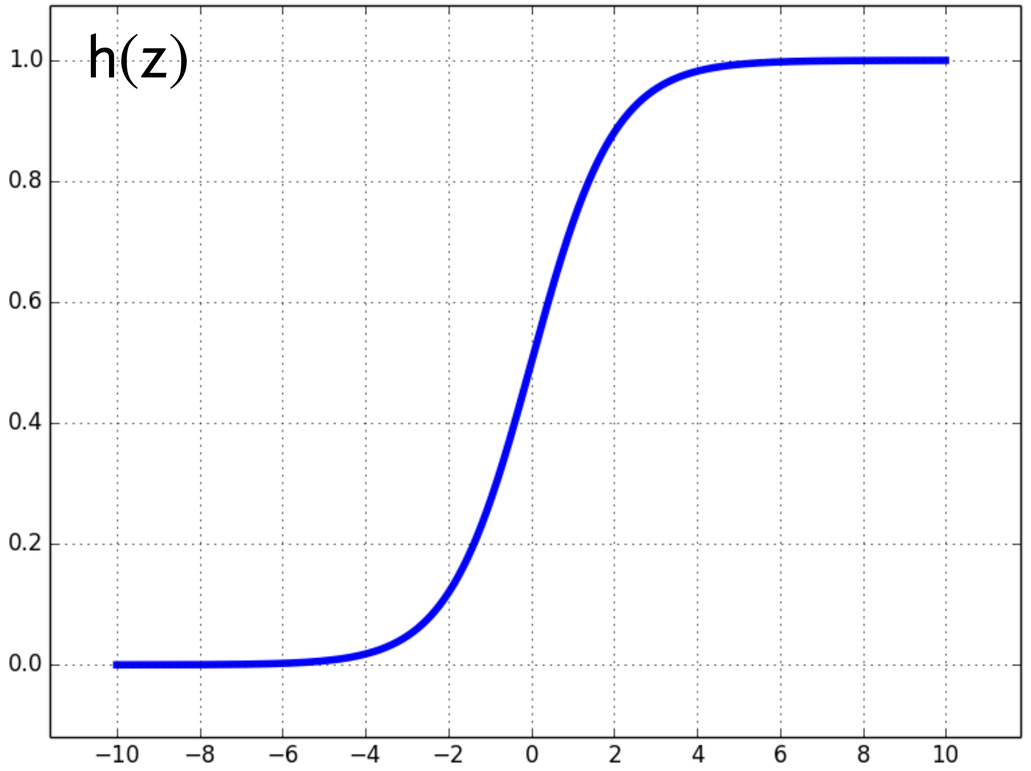
Logistic Regression

Labels: $y \in \{0, 1\}$ Targets: $\hat{y} \in [0, 1]$

Interpretation:

$\hat{y}$ probability estimate that outcome is 1

y actual outcome



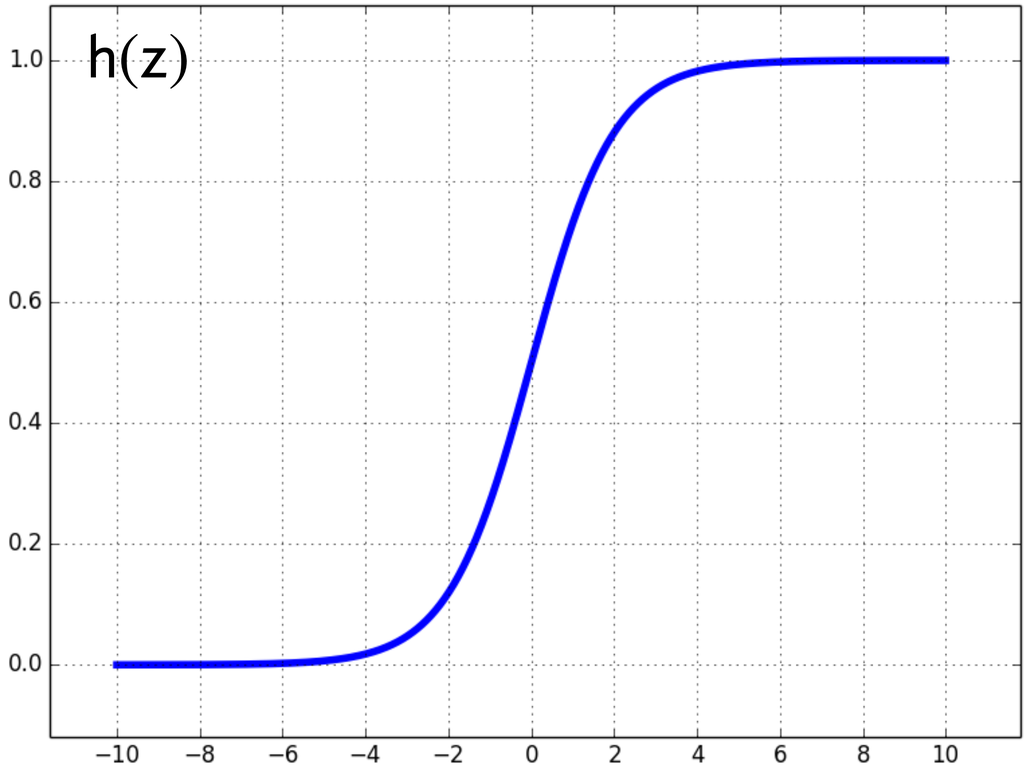
Logistic Regression

Labels: $y \in \{0, 1\}$ Targets: $\hat{y} \in [0, 1]$

Interpretation:

$\hat{y}$ probability estimate that outcome is 1

y actual outcome



$\hat{y} = h(\mathbf{w} \cdot \mathbf{x})$ where activation is:

$$h(z) = \frac{1}{1 + e^{-z}}$$

Loss for Logistic Regression

Loss for Logistic Regression

$\ell(y, \hat{y}) = -y\log(\hat{y}) - (1-y)\log(1-\hat{y})$ // works also when $y \in [0, 1]$

Loss for Logistic Regression

$\ell(y, \hat{y}) = -y\log(\hat{y}) - (1-y)\log(1-\hat{y})$ // works also when $y \in [0, 1]$

If $y == 1$:

$\ell(y, \hat{y}) = -\log(\hat{y})$

Else:

$\ell(y, \hat{y}) = -\log(1-\hat{y})$

Loss for Logistic Regression

$\ell(y, \hat{y}) = -y\log(\hat{y}) - (1-y)\log(1-\hat{y})$ // works also when $y \in [0, 1]$

If $y == 1$:

$\ell(y, \hat{y}) = -\log(\hat{y})$

Else:

$\ell(y, \hat{y}) = -\log(1-\hat{y})$

Loss for Logistic Regression

$\ell(y, \hat{y}) = -y\log(\hat{y}) - (1-y)\log(1-\hat{y})$ // works also when $y \in [0, 1]$

If y == 1:

$\ell(y, \hat{y}) = -\log(\hat{y})$

Else:

$\ell(y, \hat{y}) = -\log(1-\hat{y})$

Derivative of loss w.r.t prediction: $\frac{d\ell(y, \hat{y})}{d\hat{y}} = \frac{1-y}{1-\hat{y}} - \frac{y}{\hat{y}} = \frac{\hat{y}-y}{\hat{y}(1-\hat{y})}$

Activation Function

Activation Function

Recall: $h(z) = \frac{1}{1 + e^{-z}}$

Activation Function

Recall: $h(z) = \frac{1}{1 + e^{-z}}$

$$\frac{dh}{dz} = \frac{d}{dz}\left[\frac{1}{1 + e^{-z}}\right] = -\frac{-e^{-z}}{(1 + e^{-z})^2} = \frac{1}{1 + e^{-z}}\frac{e^{-z}}{1 + e^{-z}} = \frac{1}{1 + e^{-z}}\left(1 - \frac{1}{1 + e^{-z}}\right)$$

Activation Function

Recall: $h(z) = \frac{1}{1 + e^{-z}}$

$$\frac{dh}{dz} = \frac{d}{dz}\left[\frac{1}{1 + e^{-z}}\right] = -\frac{-e^{-z}}{(1 + e^{-z})^2} = \frac{1}{1 + e^{-z}}\frac{e^{-z}}{1 + e^{-z}} = \frac{1}{1 + e^{-z}}\left(1 - \frac{1}{1 + e^{-z}}\right)$$

h(z) 1-h(z)

Activation Function

Recall: $h(z) = \frac{1}{1 + e^{-z}}$

$$\frac{dh}{dz} = \frac{d}{dz}\left[\frac{1}{1 + e^{-z}}\right] = -\frac{-e^{-z}}{(1 + e^{-z})^2} = \frac{1}{1 + e^{-z}}\frac{e^{-z}}{1 + e^{-z}} = \frac{1}{1 + e^{-z}}\left(1 - \frac{1}{1 + e^{-z}}\right)$$

h(z) 1-h(z)

We get: $h'(z) = h(z)\,(1 - h(z)) = \hat{y}\,(1 - \hat{y})$

S.G. for Logistic Regression

S.G. for Logistic Regression

$$\frac{\partial}{\partial z}\ell\left(y, h(z)\right) = \frac{d\ell}{d\hat{y}}\frac{d\hat{y}}{dz} = \frac{\hat{y} - y}{\hat{y}(1-\hat{y})}\hat{y}\,(1-\hat{y}) = \hat{y} - y$$

S.G. for Logistic Regression

$$\frac{\partial}{\partial z}\ell\left(y, h(z)\right) = \frac{d\ell}{d\hat{y}}\frac{d\hat{y}}{dz} = \frac{\hat{y} - y}{\hat{y}(1-\hat{y})}\,\hat{y}\,(1-\hat{y}) = \hat{y} - y$$

$$\frac{\partial z}{\partial w_j} = x_j \quad \Rightarrow \quad \nabla_{\mathbf{w}} z = \mathbf{x} \quad \Rightarrow \quad \nabla_{\mathbf{w}}\ell\left(y, h(\mathbf{w}\cdot\mathbf{x})\right) = (\hat{y} - y)\,\mathbf{x}$$

S.G. for Logistic Regression

$$\frac{\partial}{\partial z}\ell\left(y, h(z)\right) = \frac{d\ell}{d\hat{y}}\frac{d\hat{y}}{dz} = \frac{\hat{y} - y}{\hat{y}(1-\hat{y})}\,\hat{y}\,(1-\hat{y}) = \hat{y} - y$$

$$\frac{\partial z}{\partial w_j} = x_j \quad \Rightarrow \quad \nabla_{\mathbf{w}} z = \mathbf{x} \quad \Rightarrow \quad \nabla_{\mathbf{w}}\ell\left(y, h(\mathbf{w}\cdot\mathbf{x})\right) = (\hat{y} - y)\,\mathbf{x}$$

$$\nabla\mathscr{L}(\mathbf{w}) = \frac{1}{|S|}\sum_{i\in S}(\hat{y}_i - y_i)\,\mathbf{x}_i \quad \Rightarrow \quad \mathbf{w}_{t+1} \leftarrow \mathbf{w}_t + \frac{\eta_t}{|S|}\sum_{i\in S}(y_i - \hat{y}_i)\,\mathbf{x}_i$$

Interpretation (single example)

Interpretation (single example)

Contribution to update is proportional to $y - \hat{y}$

Interpretation (single example)

Contribution to update is proportional to $y - \hat{y}$

When y=1 we want h(z) to be close to 1 which requires $z \gg 0$

If $\hat{y} \ll 1$ then $y - \hat{y} \approx 1$ & we add a vector (proportional to) $\mathbf{x}$ to $\mathbf{w}$

Resulting $\mathbf{w}_{t+1} \cdot \mathbf{x} = (\mathbf{w}_t + a\mathbf{x}) \cdot \mathbf{x} = \mathbf{w}_t \cdot \mathbf{x} + a\|\mathbf{x}\|^2 > \mathbf{w}_t \cdot \mathbf{x}$

Interpretation (single example)

Contribution to update is proportional to $y - \hat{y}$

When y=1 we want h(z) to be close to 1 which requires $z \gg 0$

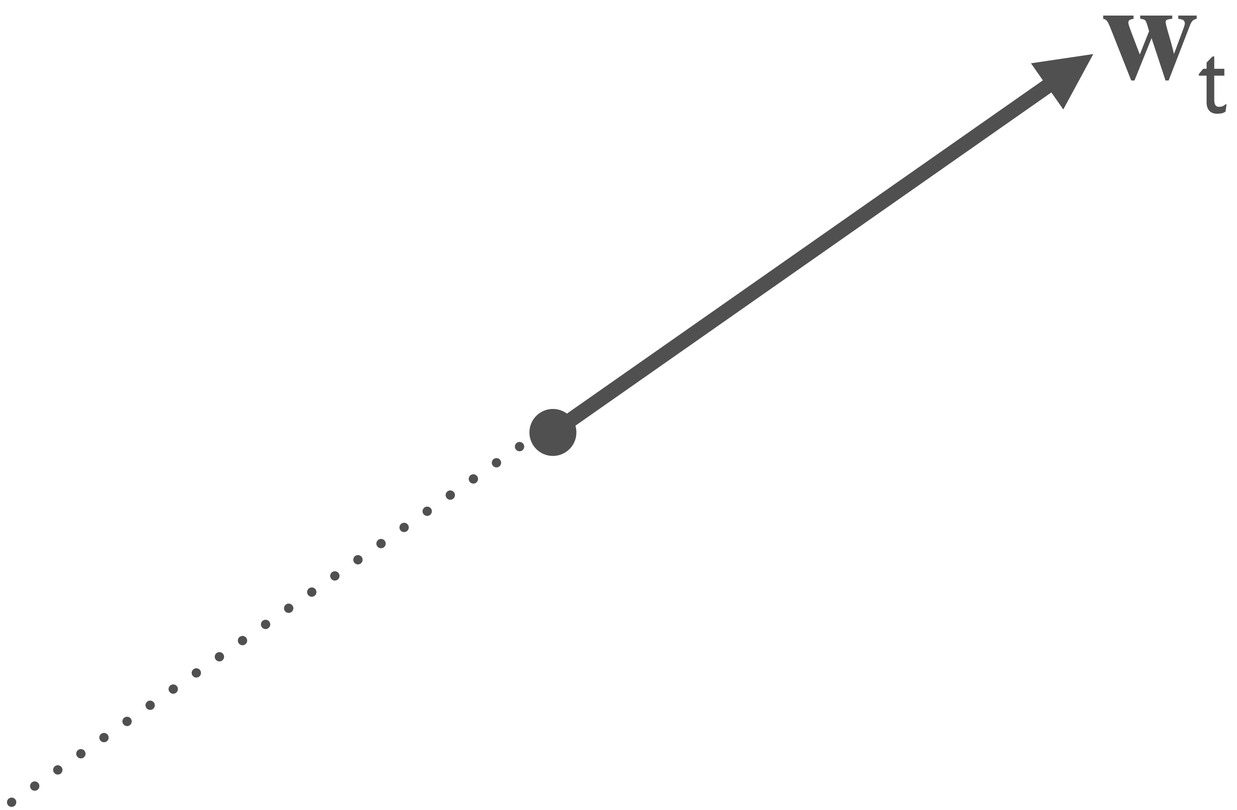
If $\hat{y} \ll 1$ then $y - \hat{y} \approx 1$ & we add a vector (proportional to) **x** to **w**

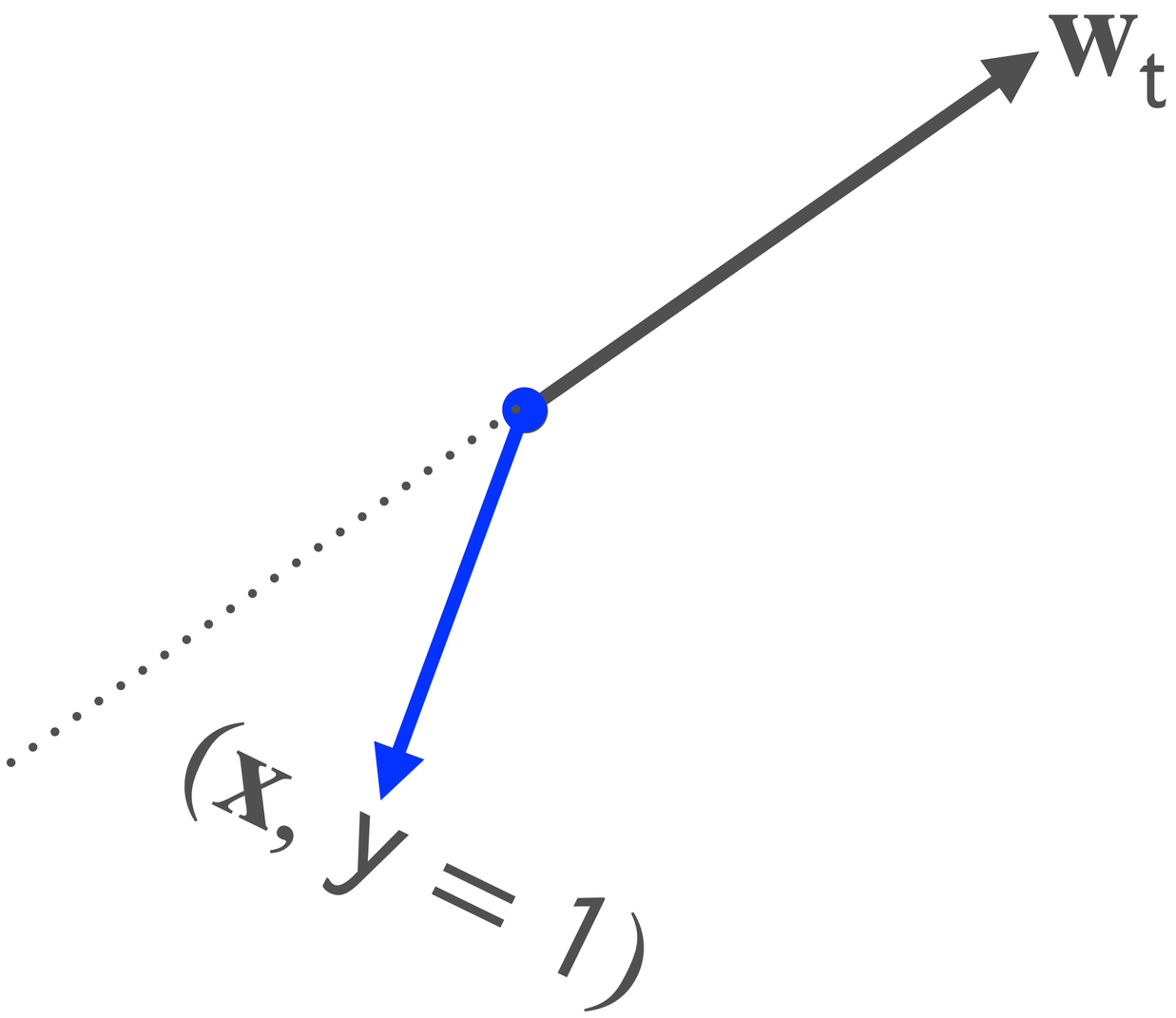
Resulting $\mathbf{w}_{t+1} \cdot \mathbf{x} = (\mathbf{w}_t + a\mathbf{x}) \cdot \mathbf{x} = \mathbf{w}_t \cdot \mathbf{x} + a\|\mathbf{x}\|^2 > \mathbf{w}_t \cdot \mathbf{x}$

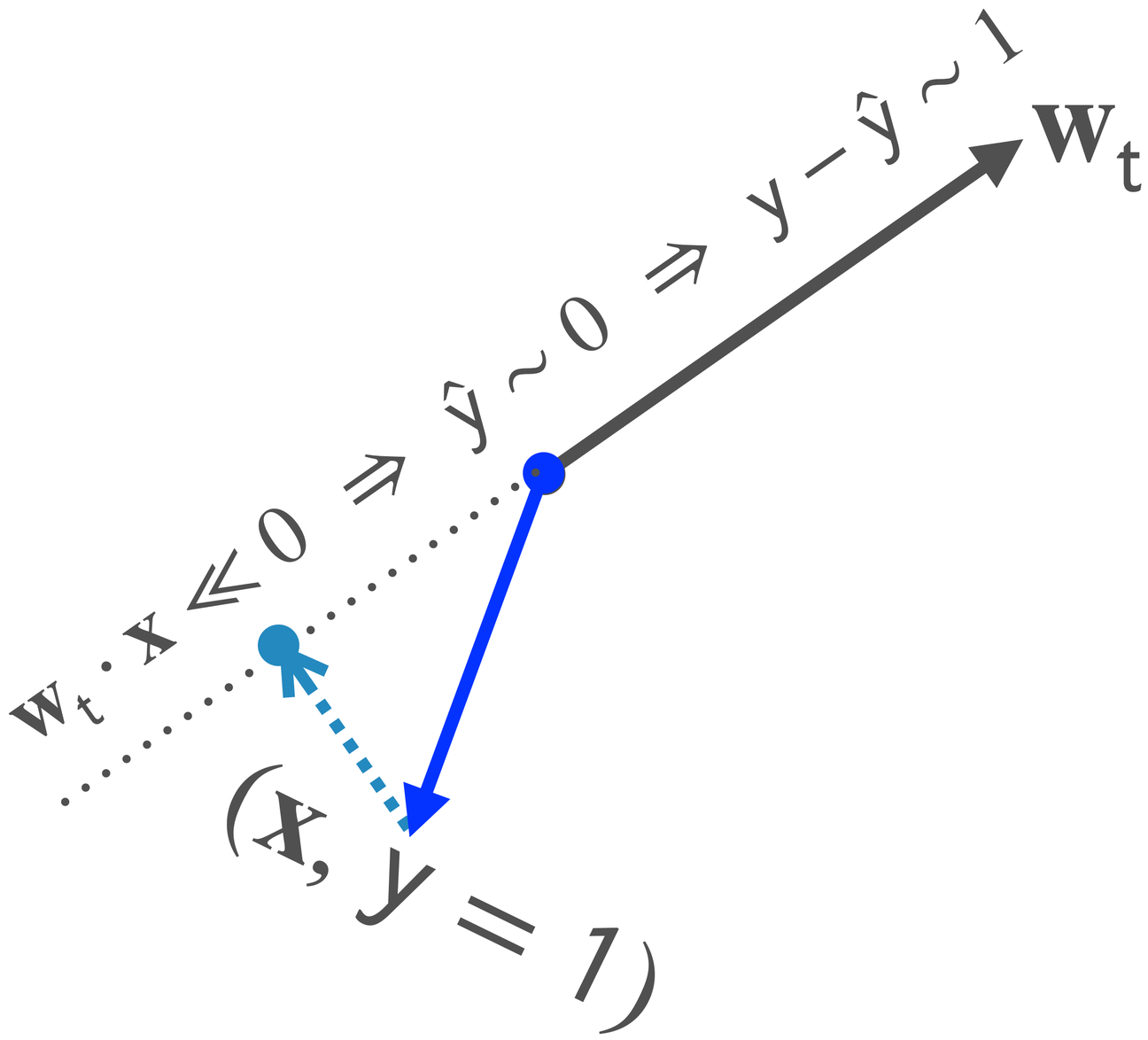
When y=0 we want h(z) to be close to 0 which requires $z \ll 0$

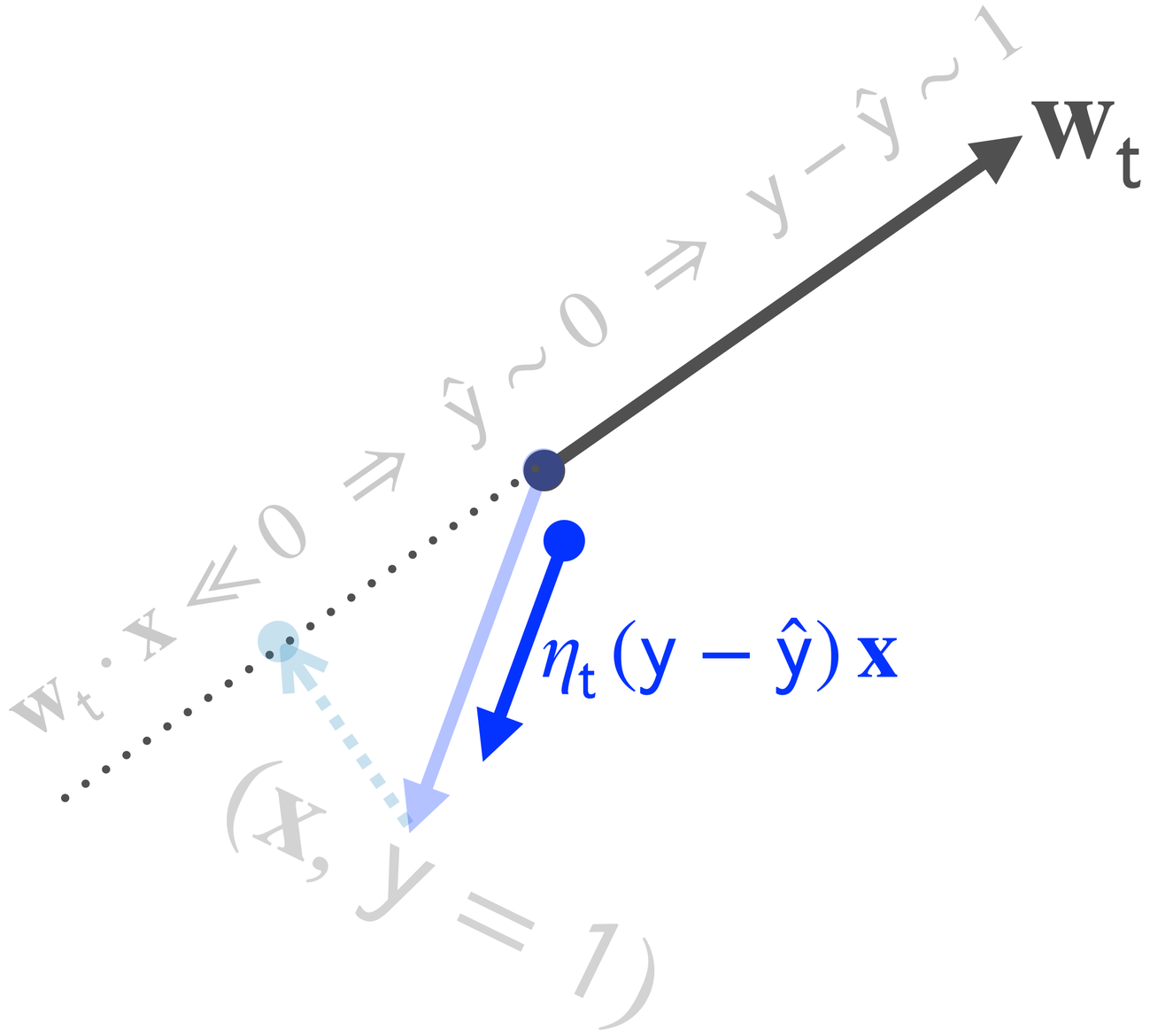
If $\hat{y} \gg 0$ then $y - \hat{y} \approx -1$ & we subtract (proportional) to -**x** from **w**

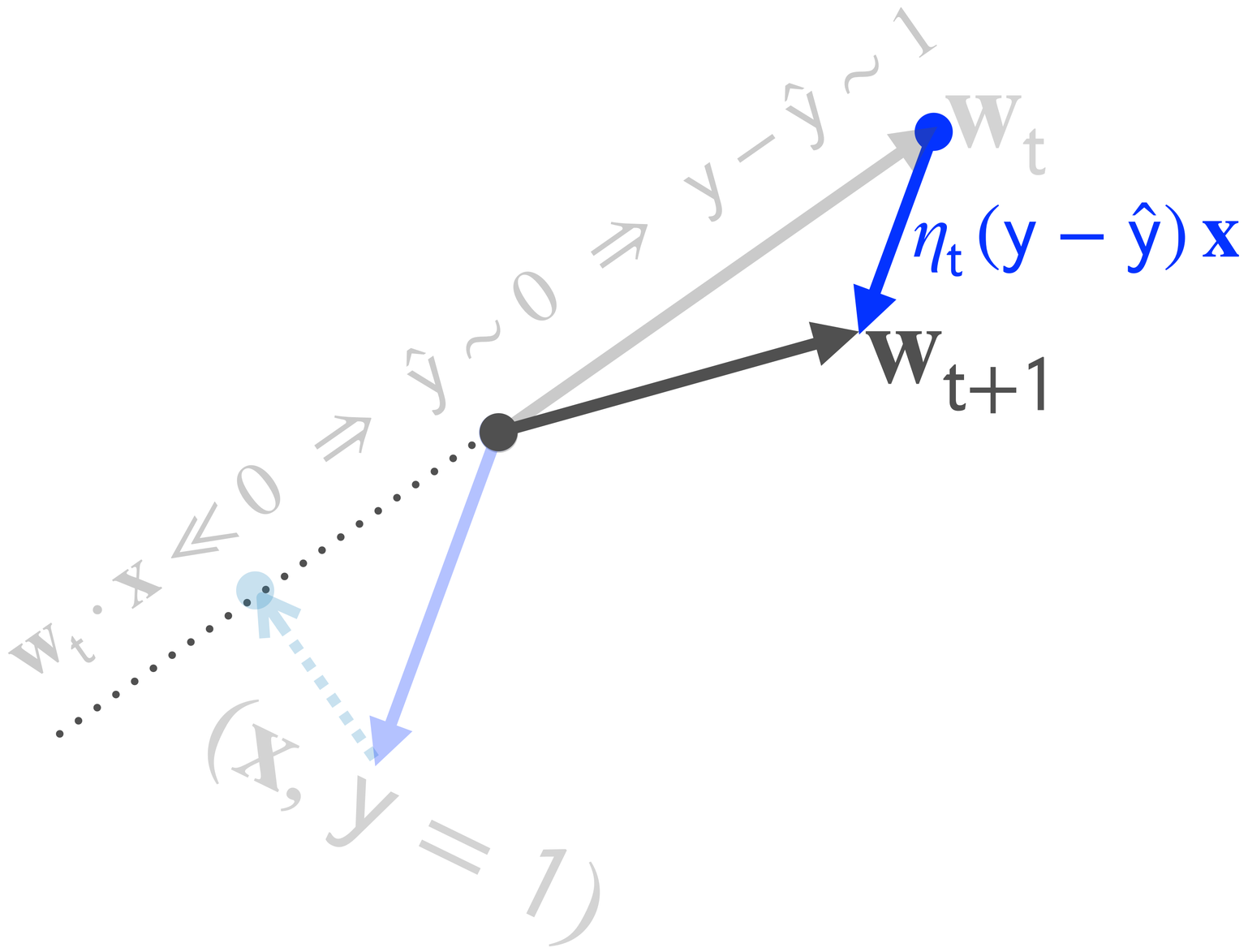
Resulting $\mathbf{w}_{t+1} \cdot \mathbf{x} = \mathbf{w}_t \cdot \mathbf{x} - a\|\mathbf{x}\|^2 < \mathbf{w}_t \cdot \mathbf{x}$











Next: before going deep...

Widen shallow learning: Multiclass Classification