6.036/6.862: Introduction to Machine Learning

Lecture: starts Tuesdays 9:35am (Boston time zone)

Course website: introml.odl.mit.edu

Who's talking? Leslie Kaelbling

Questions? On Piazza ("Lecture 3" topic)

Materials: Will all be available at course website

Last Time(s)

I. Linear classifiers
II. Perceptron algorithm
III. A more-complete ML analysis

Today's Plan

I. Linear logistic classification
II. Linear regression
III. Gradient descent

Reducing machine learning to optimization

Supply

- Hypothesis space $\mathcal{H}$
- Loss function $L(g, a)$
- Data $\mathcal{D}_n$

Define objective function

$$\mathcal{E}_n(h) = \frac{1}{n} \sum_{i=1}^{n} L(h(x^{(i)}), y^{(i)})$$

Optimization problem:

find h that minimizes the objective

Reducing machine learning to optimization

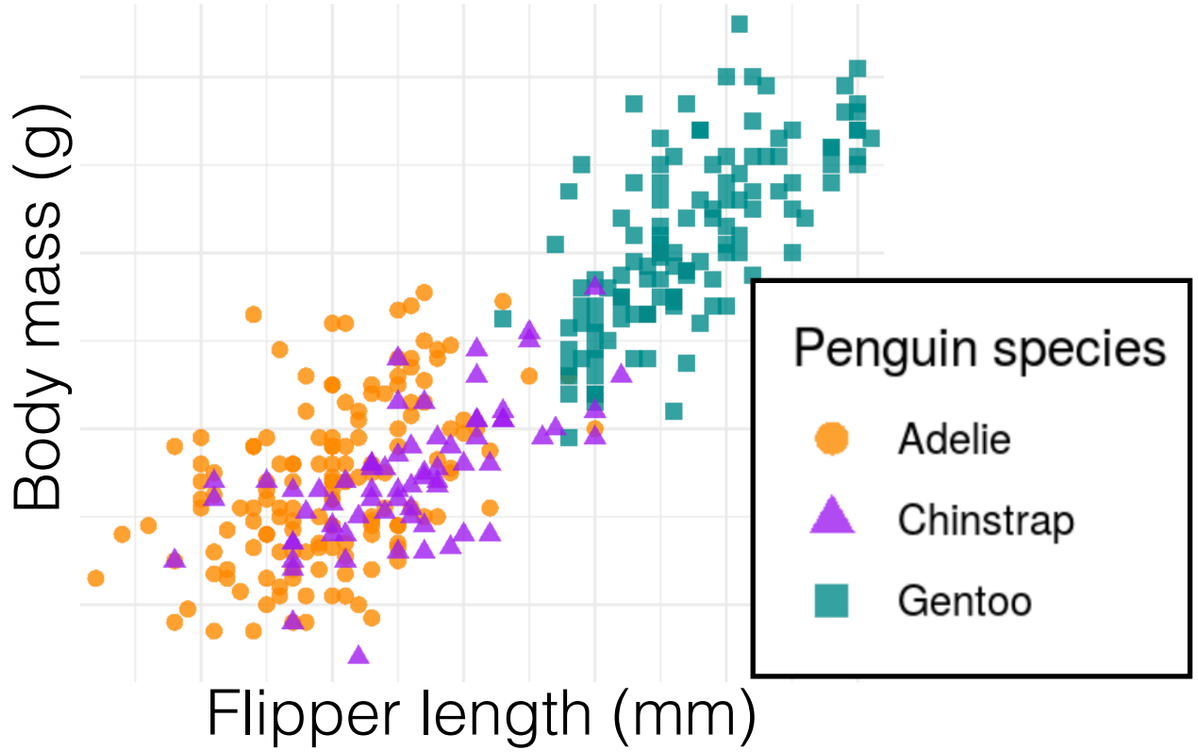
- Hypothesis space : linear models
- Loss functions :
  - classification: confidence in correct answer
  - regression: squared error
- Objective function:
  - training error
  - with added “complexity” penalty
- Optimization algorithms
  - gradient descent
  - stochastic gradient descent

Recall

- Perceptron struggles with data that's not linearly separable

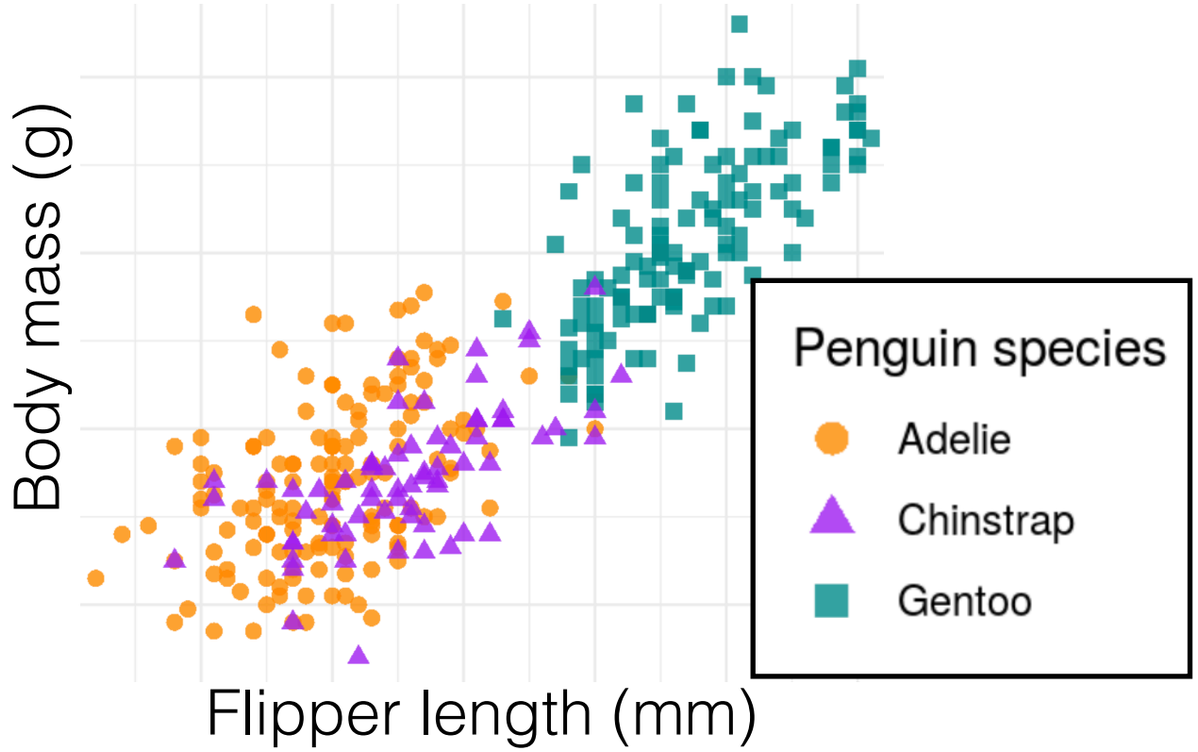
Recall

- Perceptron struggles with data that's not linearly separable



Recall

- Perceptron struggles with data that's not linearly separable

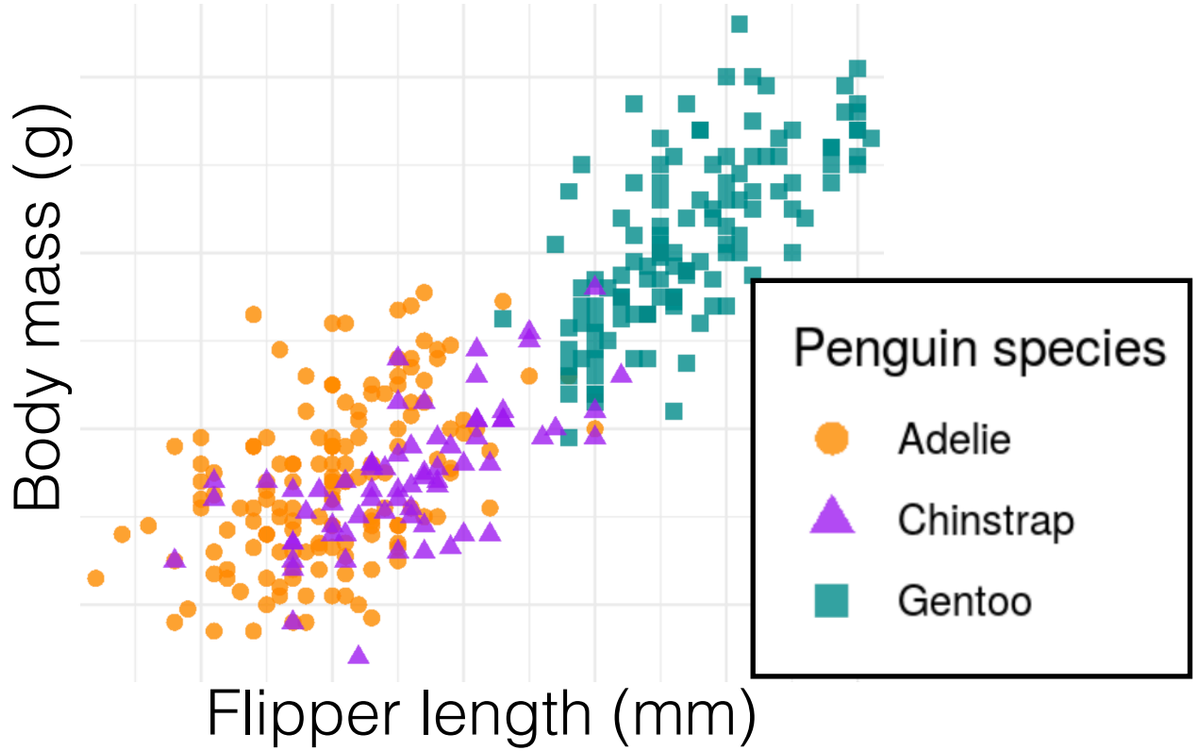


Notice

- Perceptron doesn't have a notion of uncertainty (how well do we know what we know?)

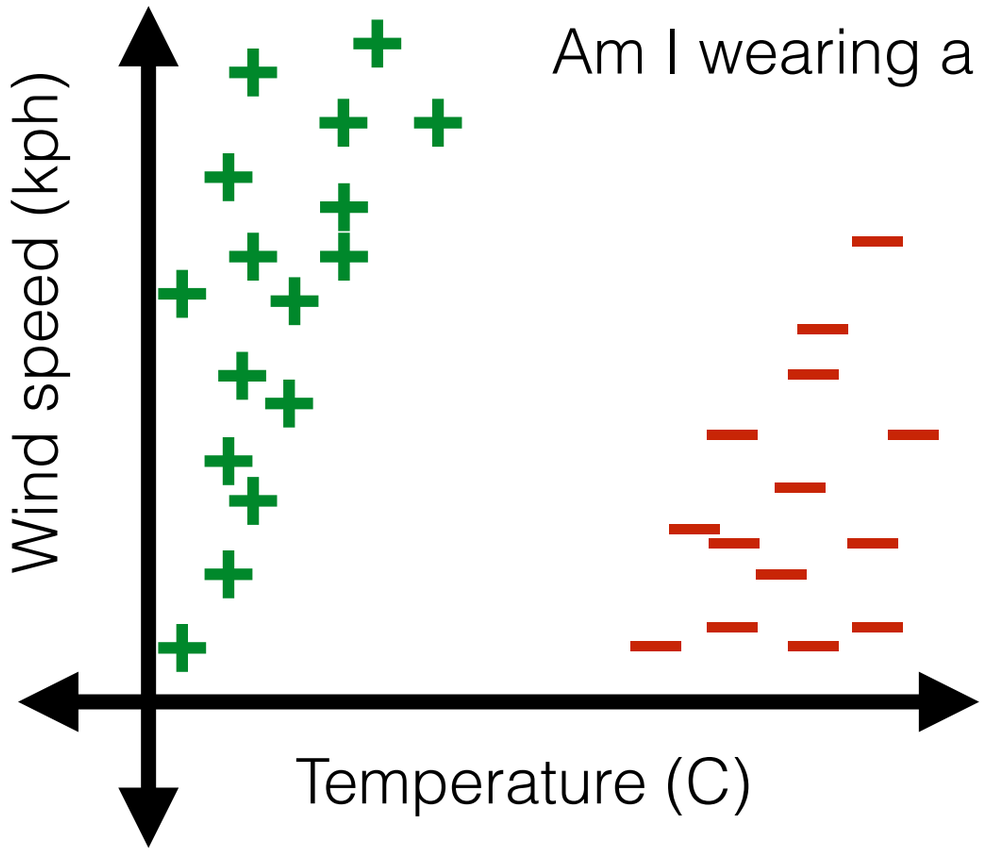
Recall

- Perceptron struggles with data that's not linearly separable



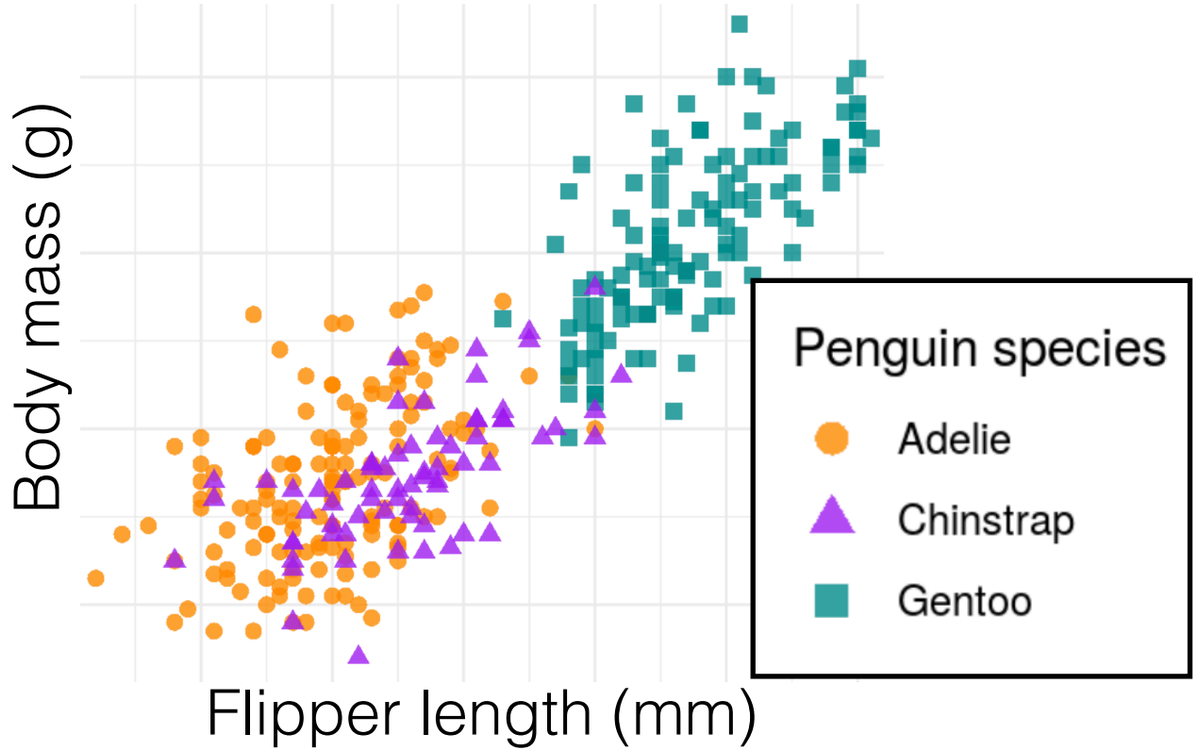
Notice

- Perceptron doesn't have a notion of uncertainty (how well do we know what we know?)



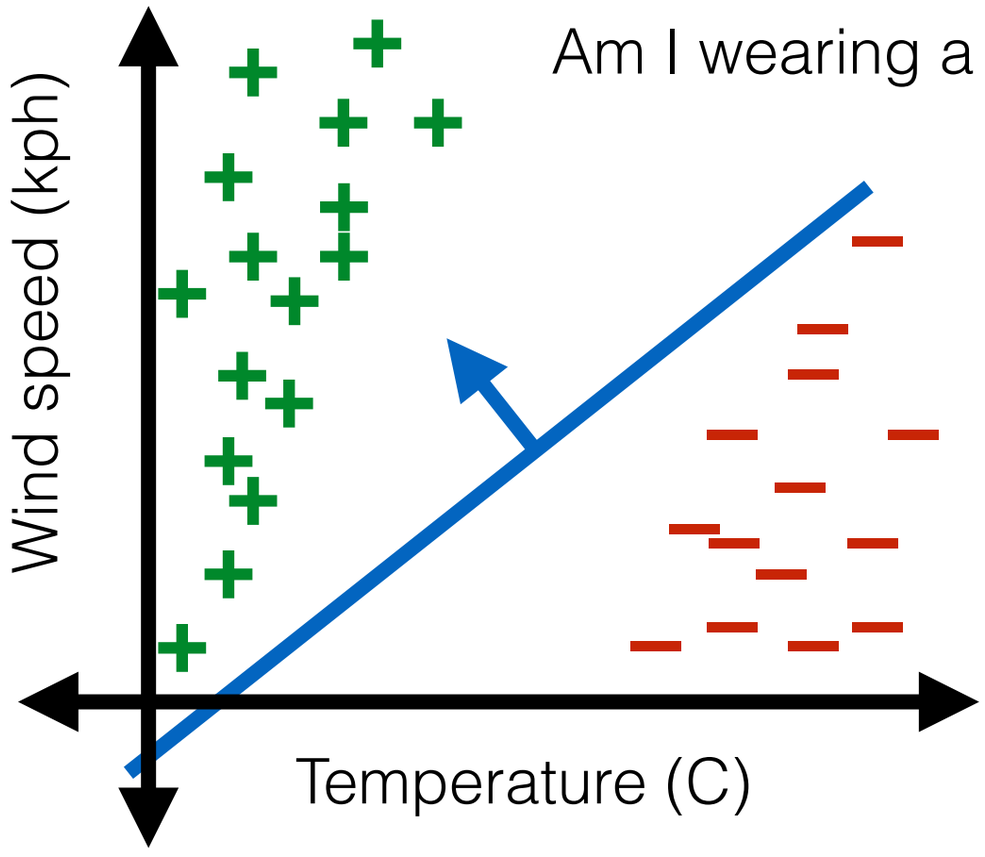
Recall

- Perceptron struggles with data that's not linearly separable



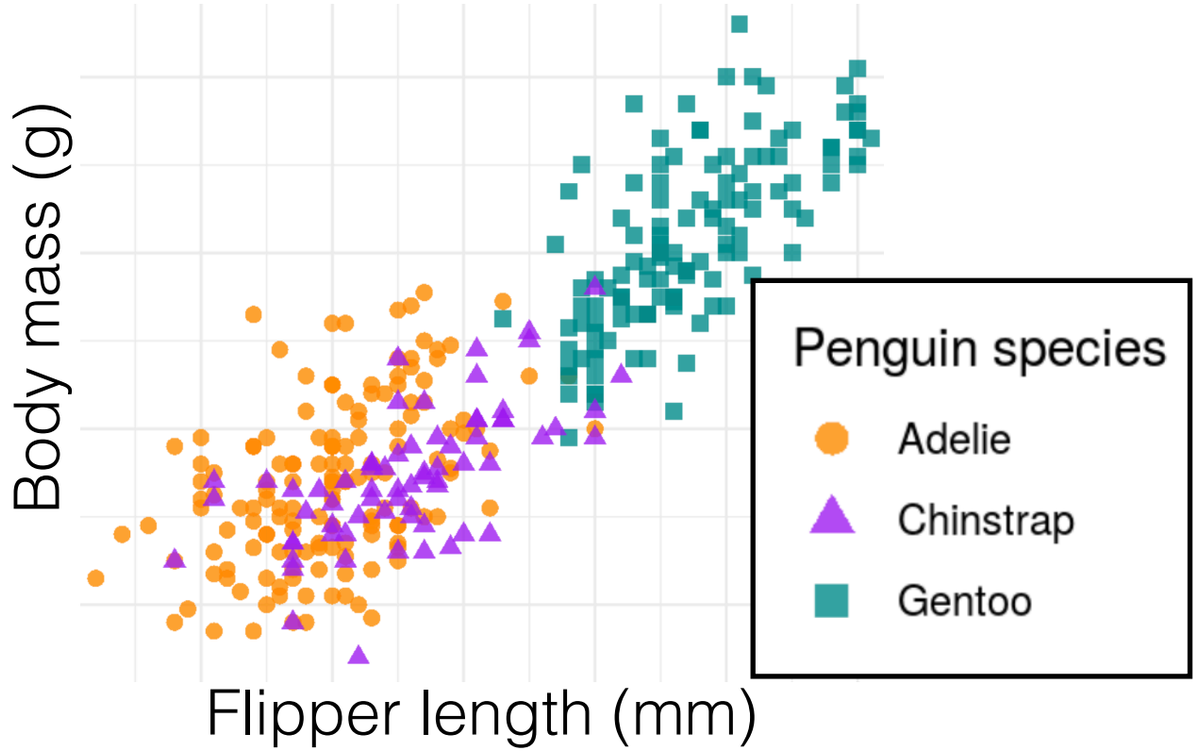
Notice

- Perceptron doesn't have a notion of uncertainty (how well do we know what we know?)



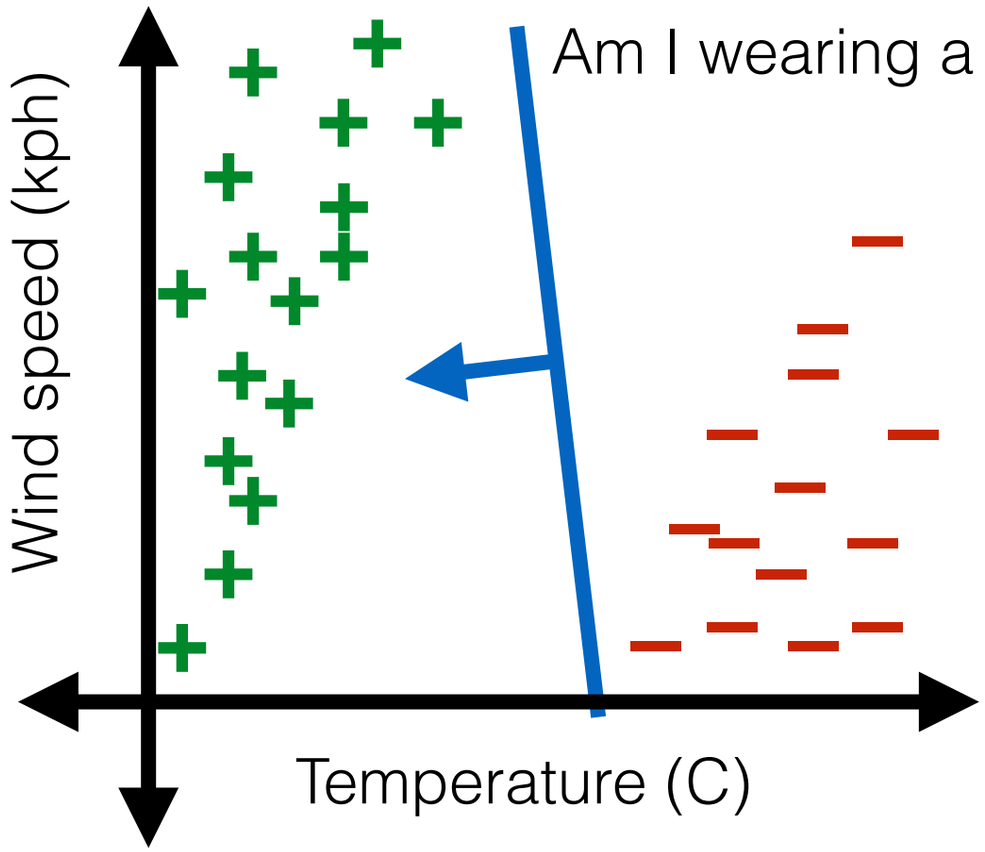
Recall

- Perceptron struggles with data that's not linearly separable



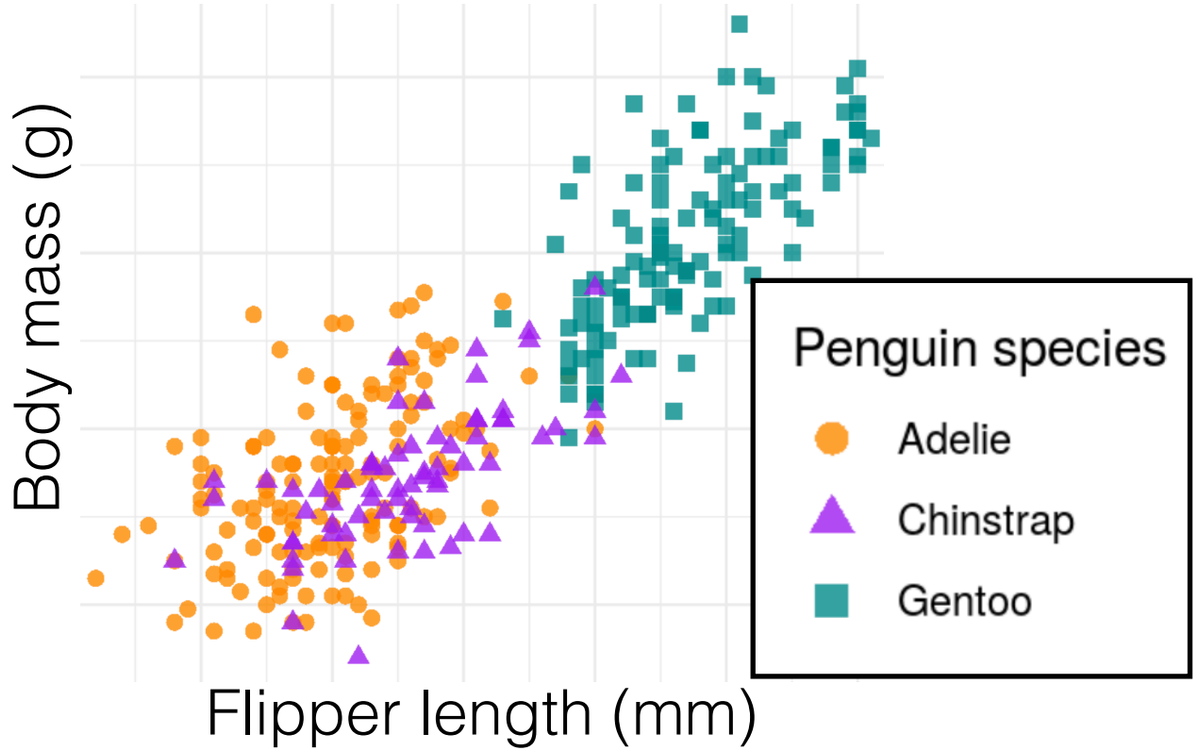
Notice

- Perceptron doesn't have a notion of uncertainty (how well do we know what we know?)



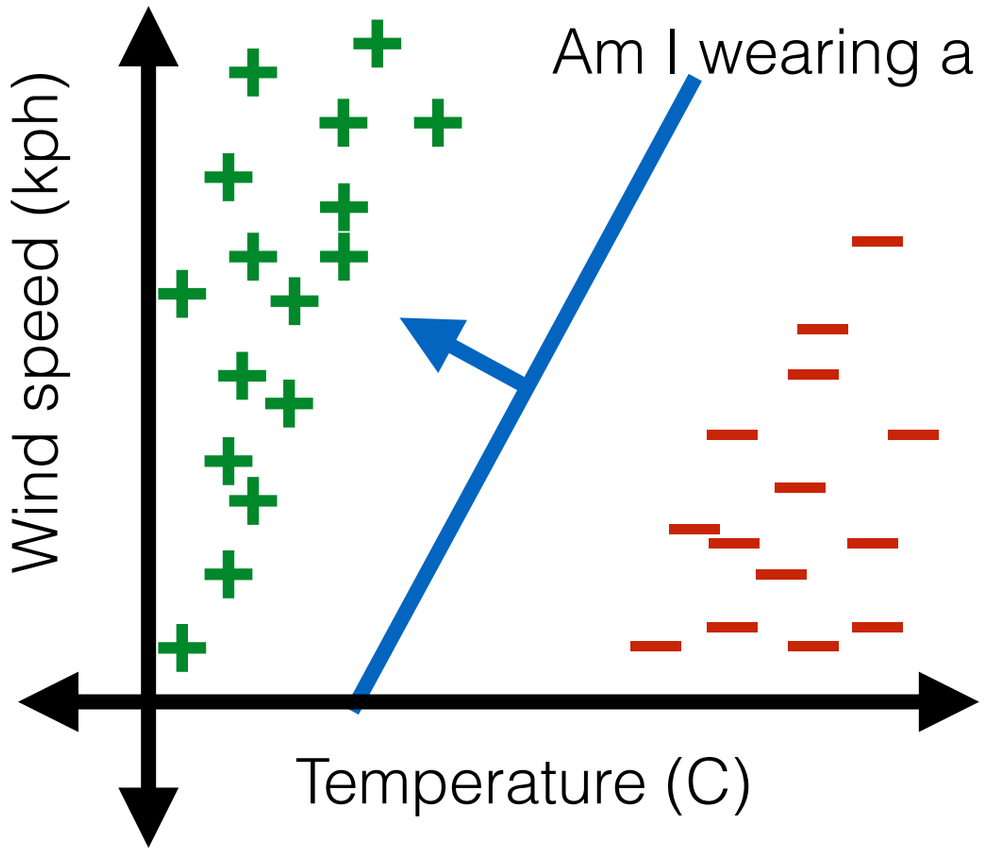
Recall

- Perceptron struggles with data that's not linearly separable



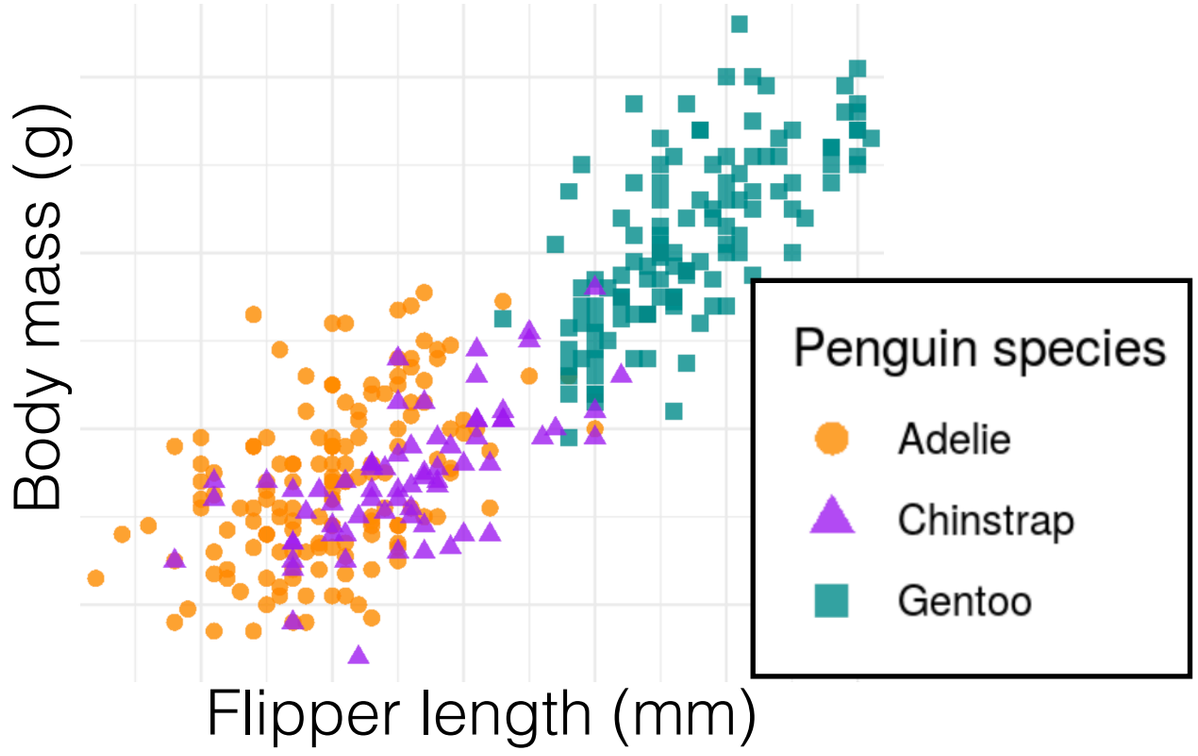
Notice

- Perceptron doesn't have a notion of uncertainty (how well do we know what we know?)



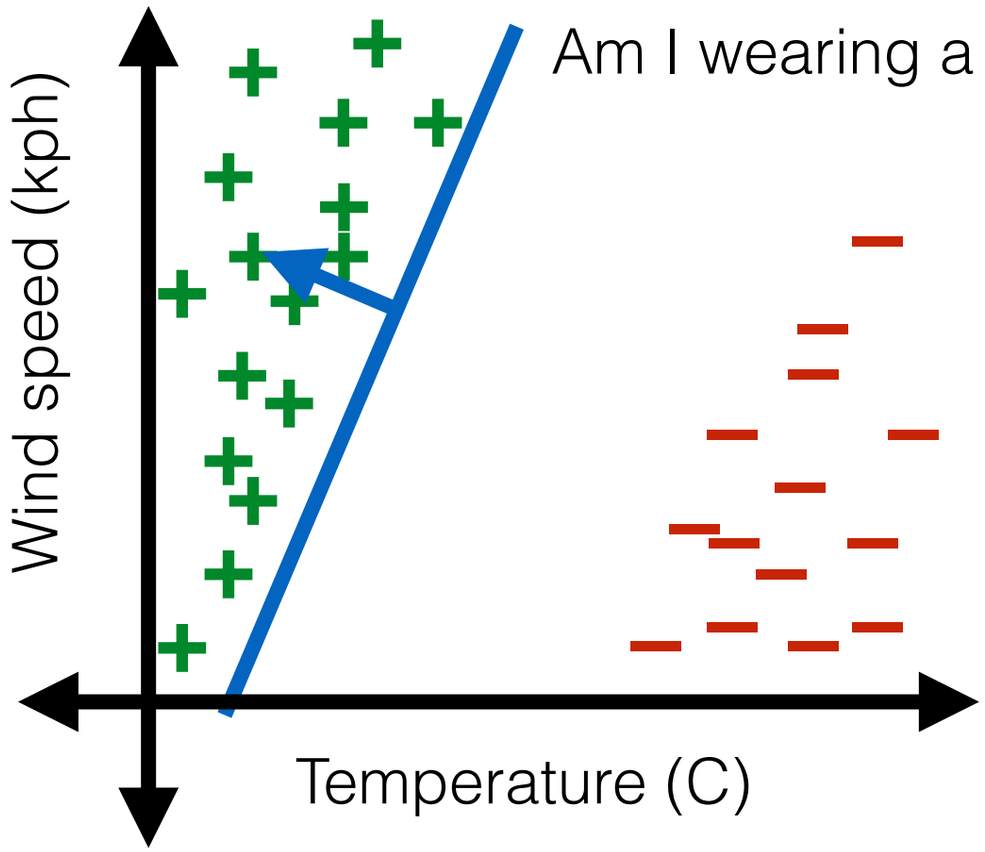
Recall

- Perceptron struggles with data that's not linearly separable



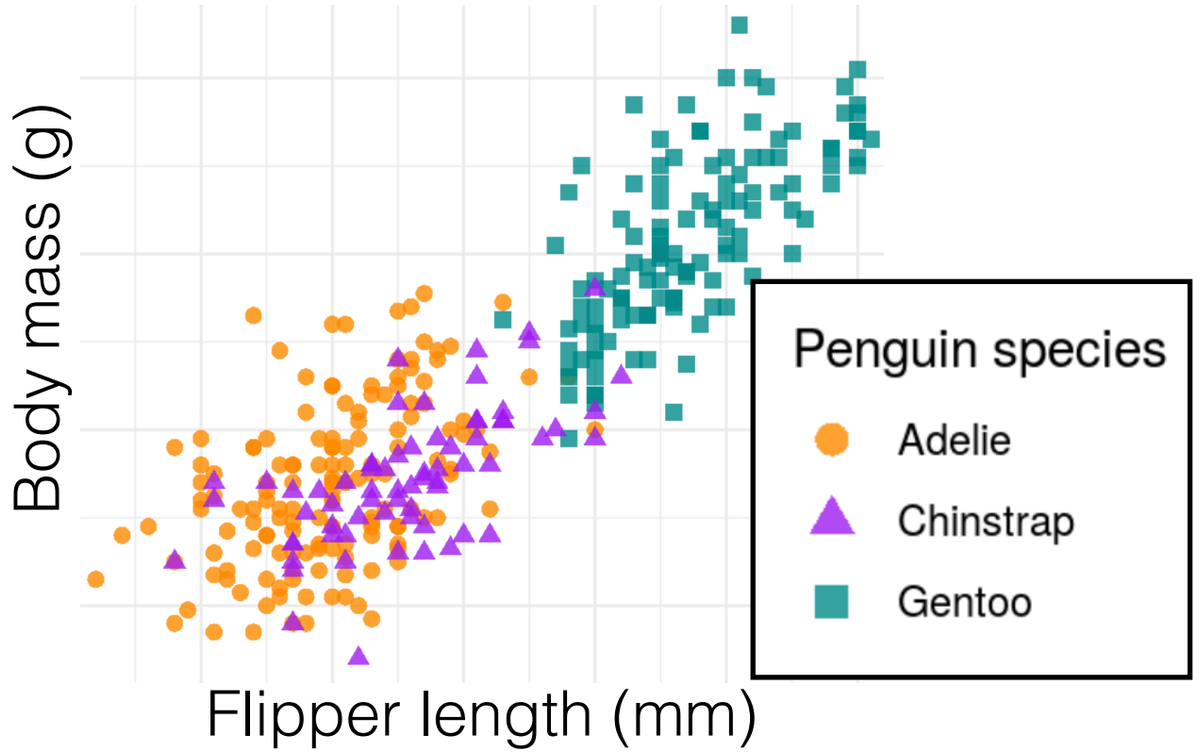
Notice

- Perceptron doesn't have a notion of uncertainty (how well do we know what we know?)



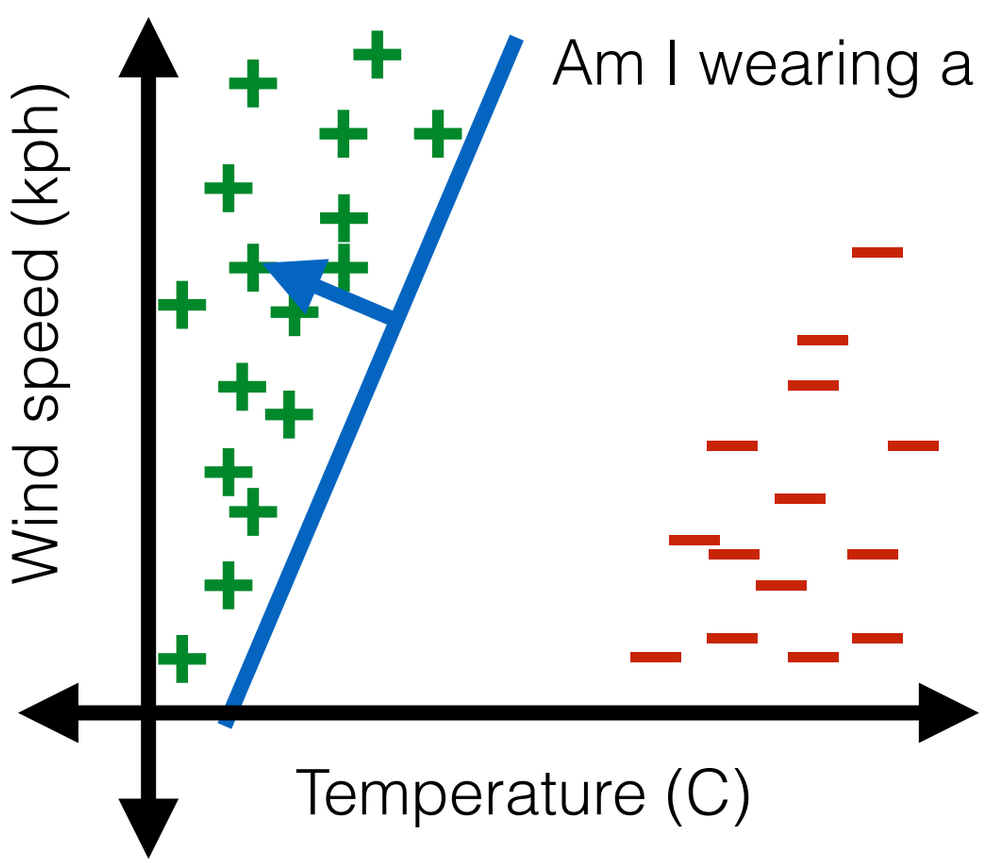
Recall

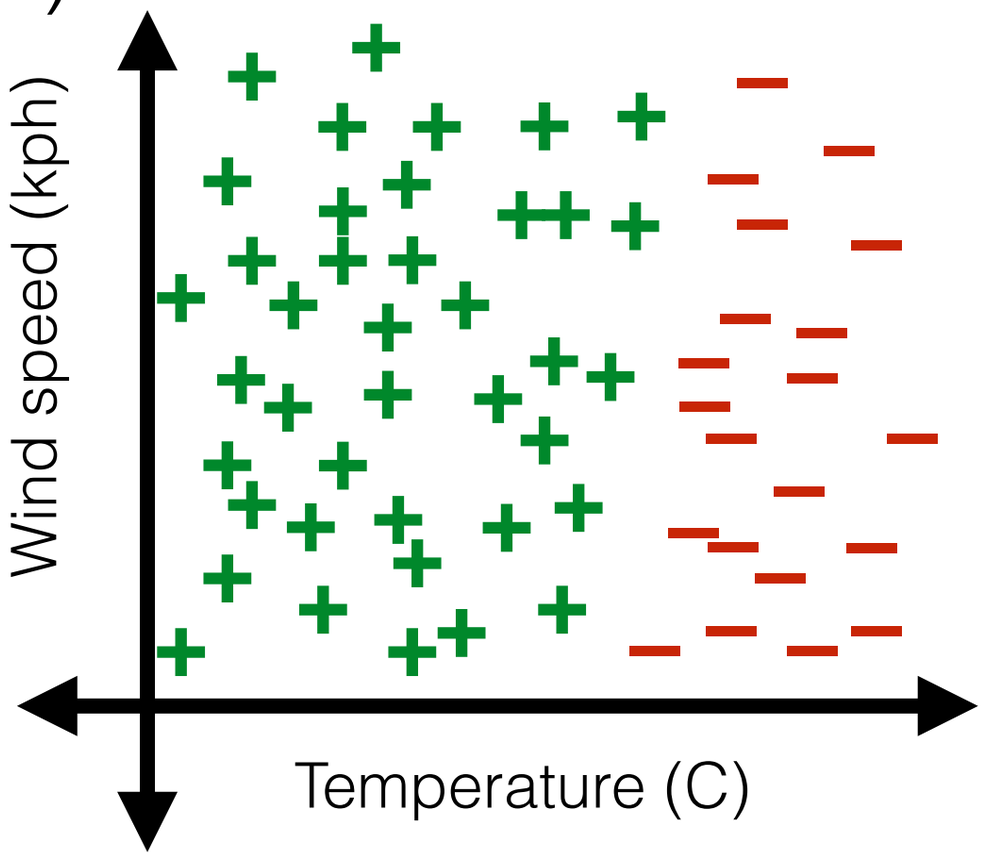
- Perceptron struggles with data that's not linearly separable



Notice

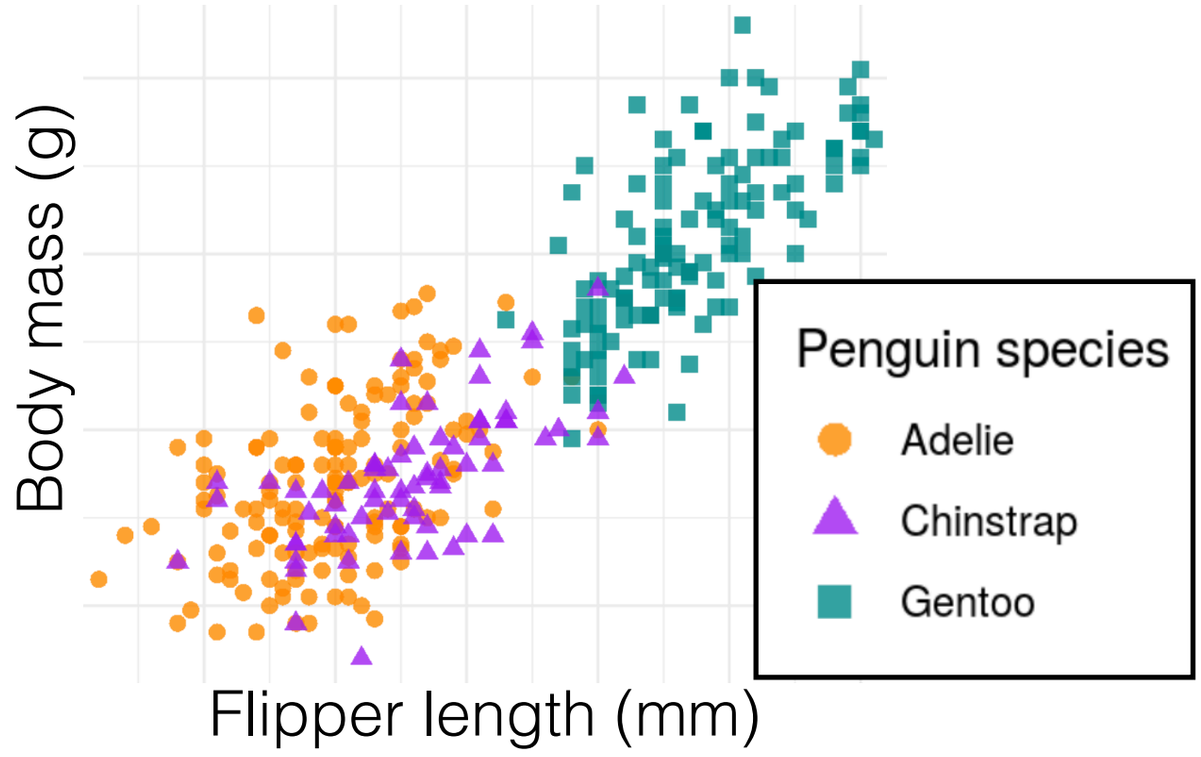
- Perceptron doesn't have a notion of uncertainty (how well do we know what we know?)





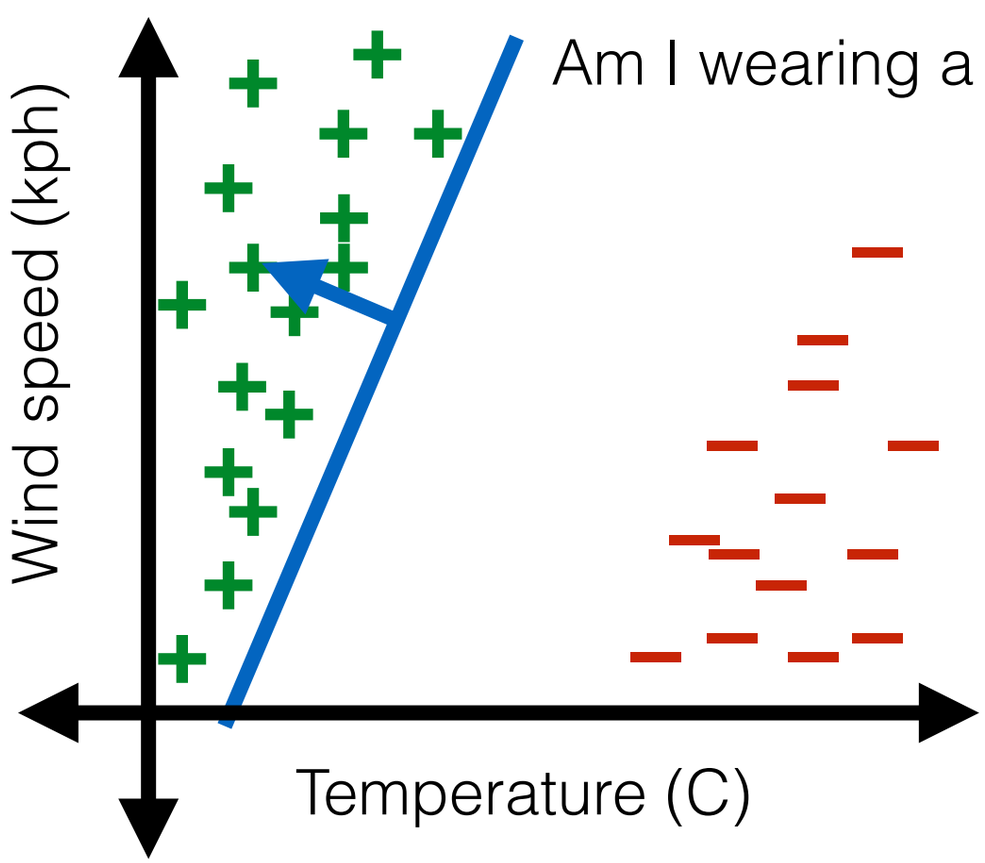
Recall

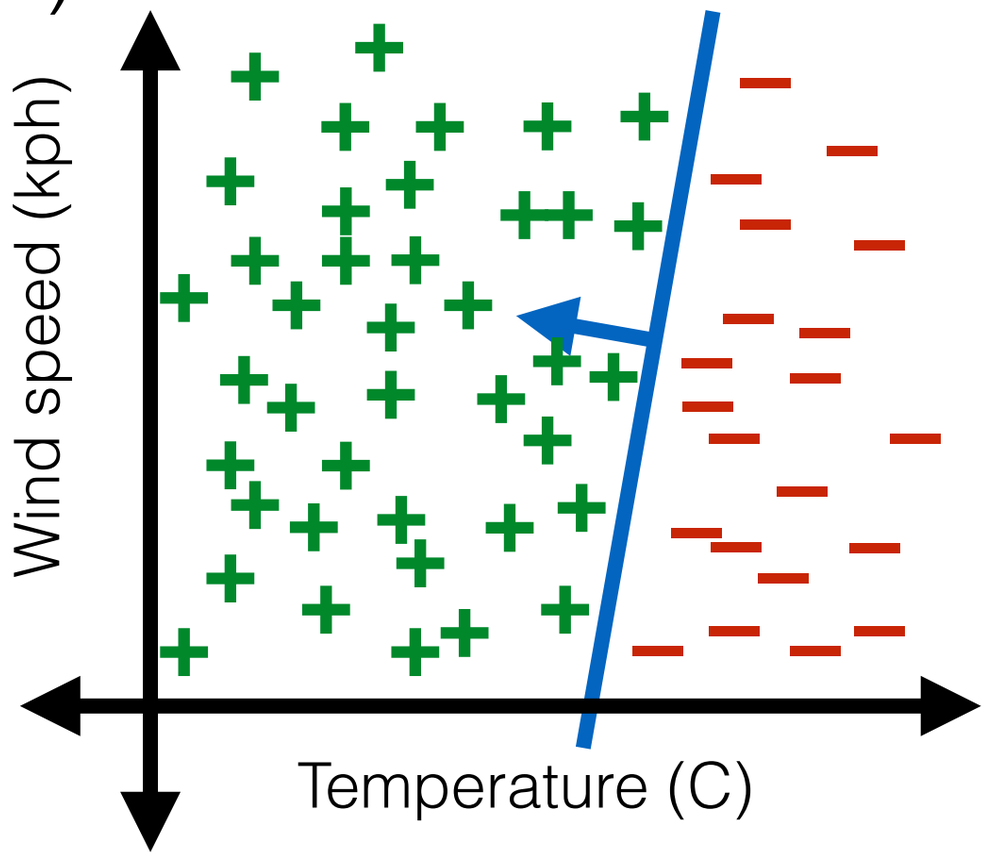
- Perceptron struggles with data that's not linearly separable



Notice

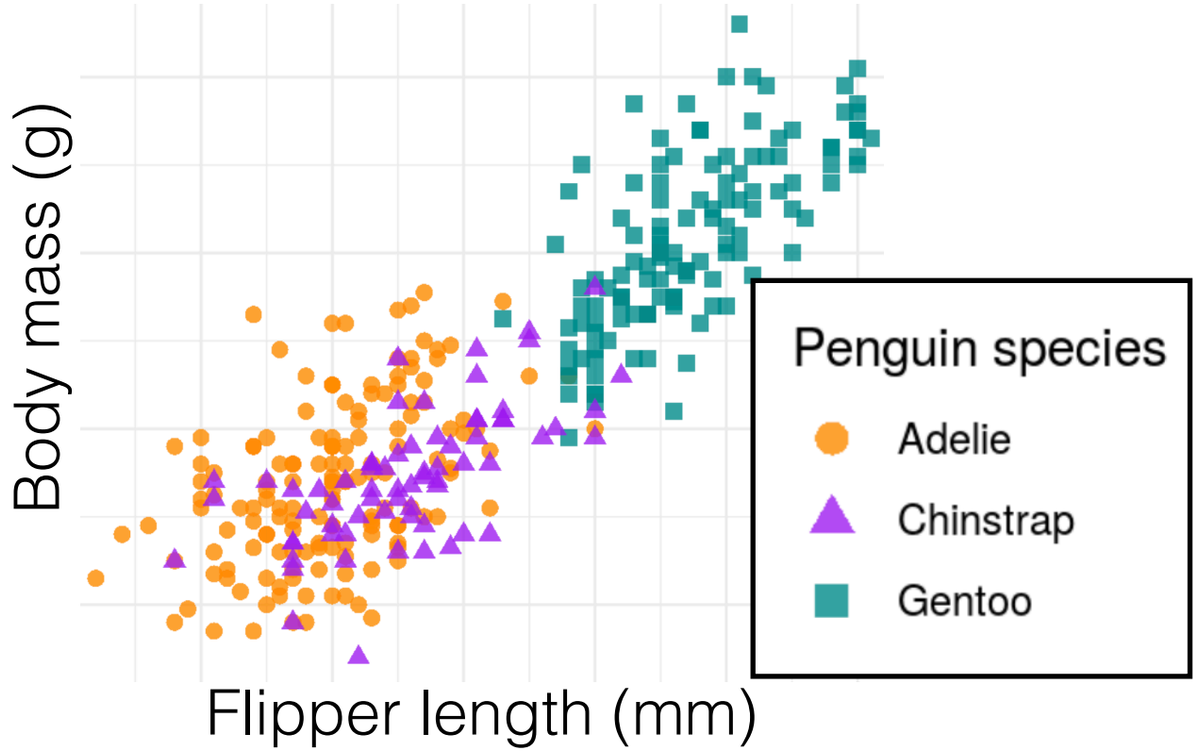
- Perceptron doesn't have a notion of uncertainty (how well do we know what we know?)





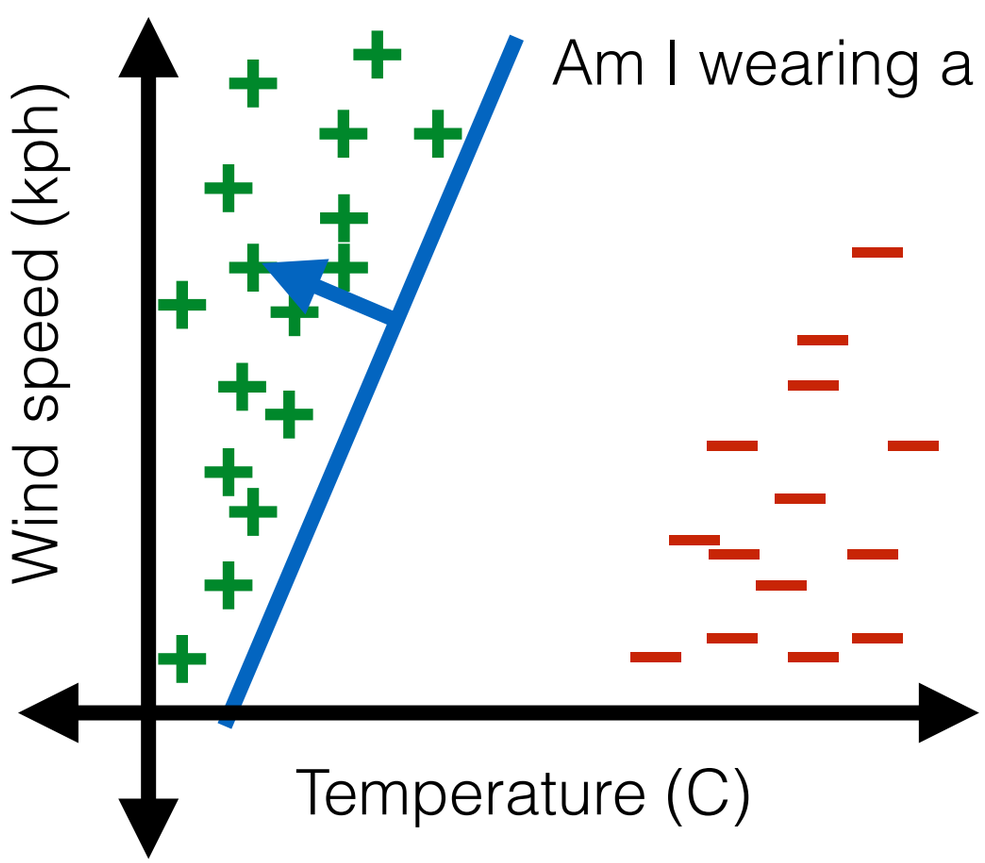
Recall

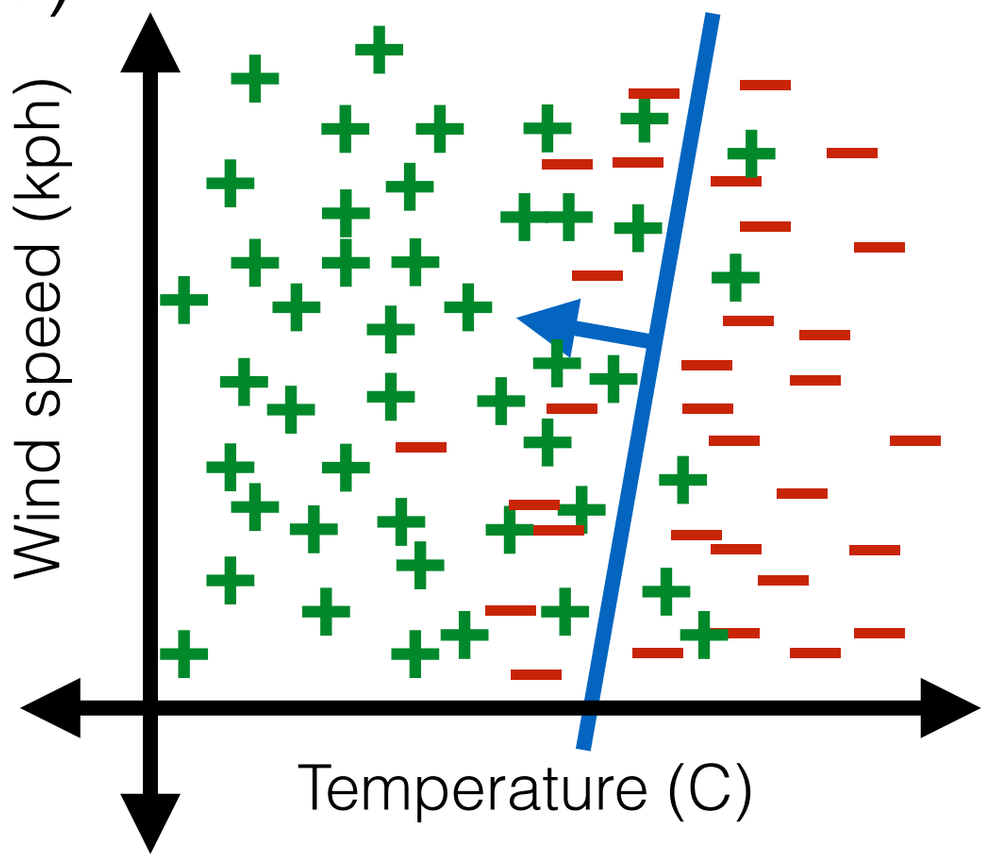
- Perceptron struggles with data that's not linearly separable



Notice

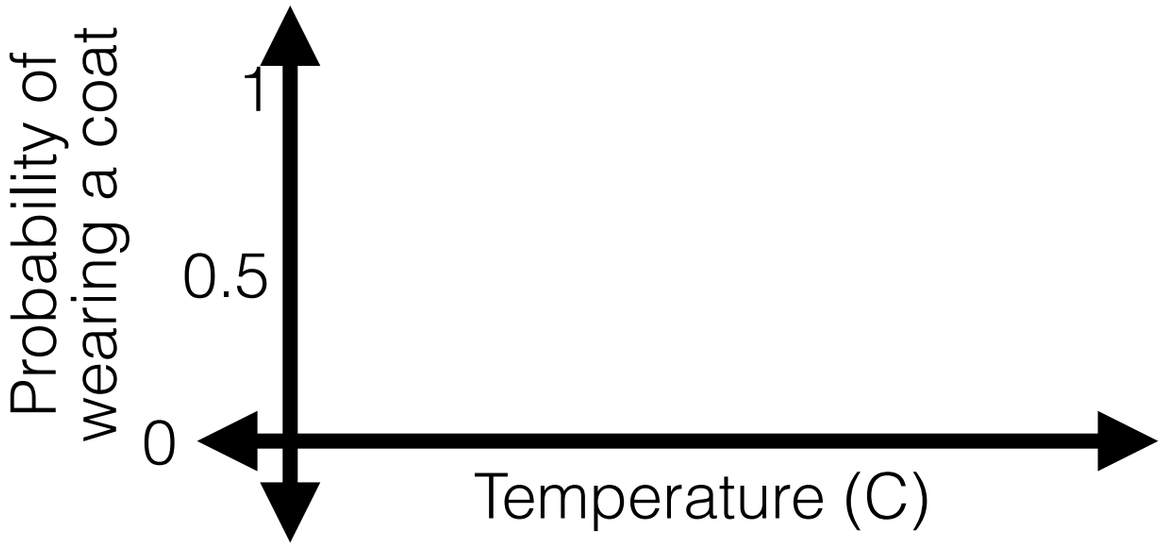
- Perceptron doesn't have a notion of uncertainty (how well do we know what we know?)



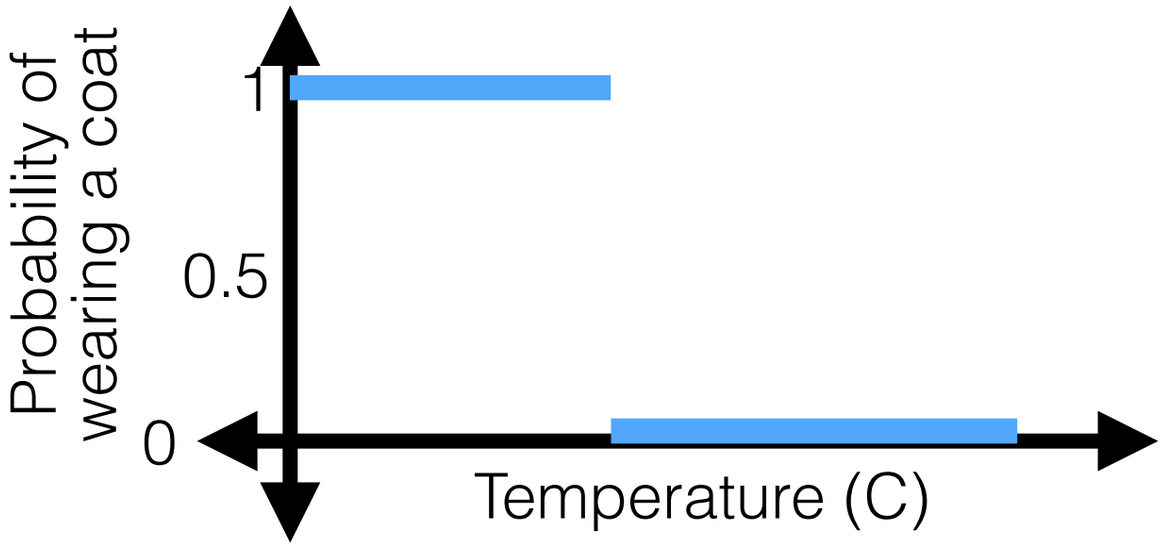


Capturing uncertainty

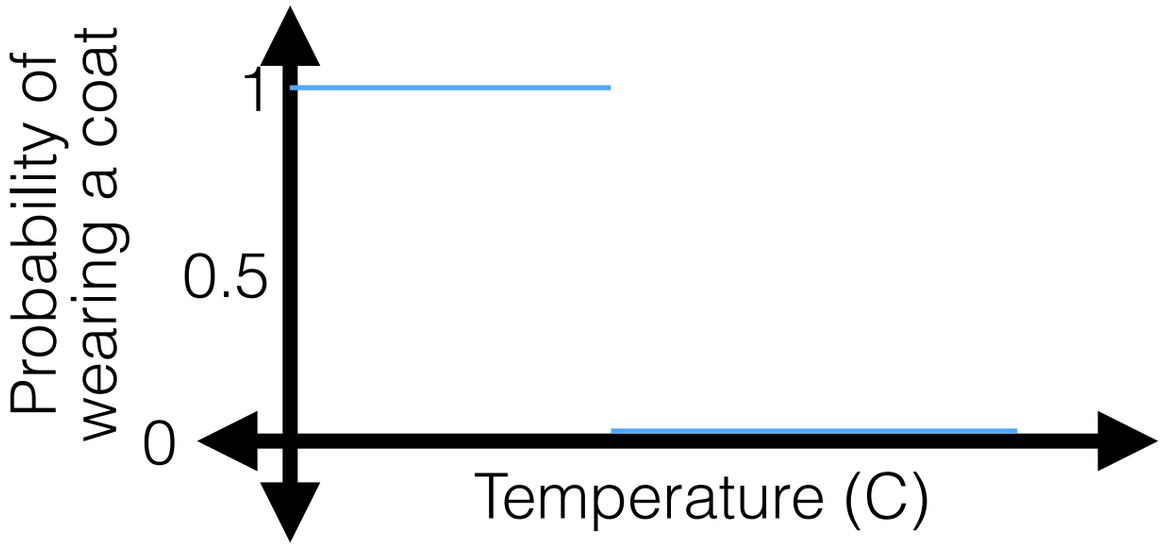
Capturing uncertainty



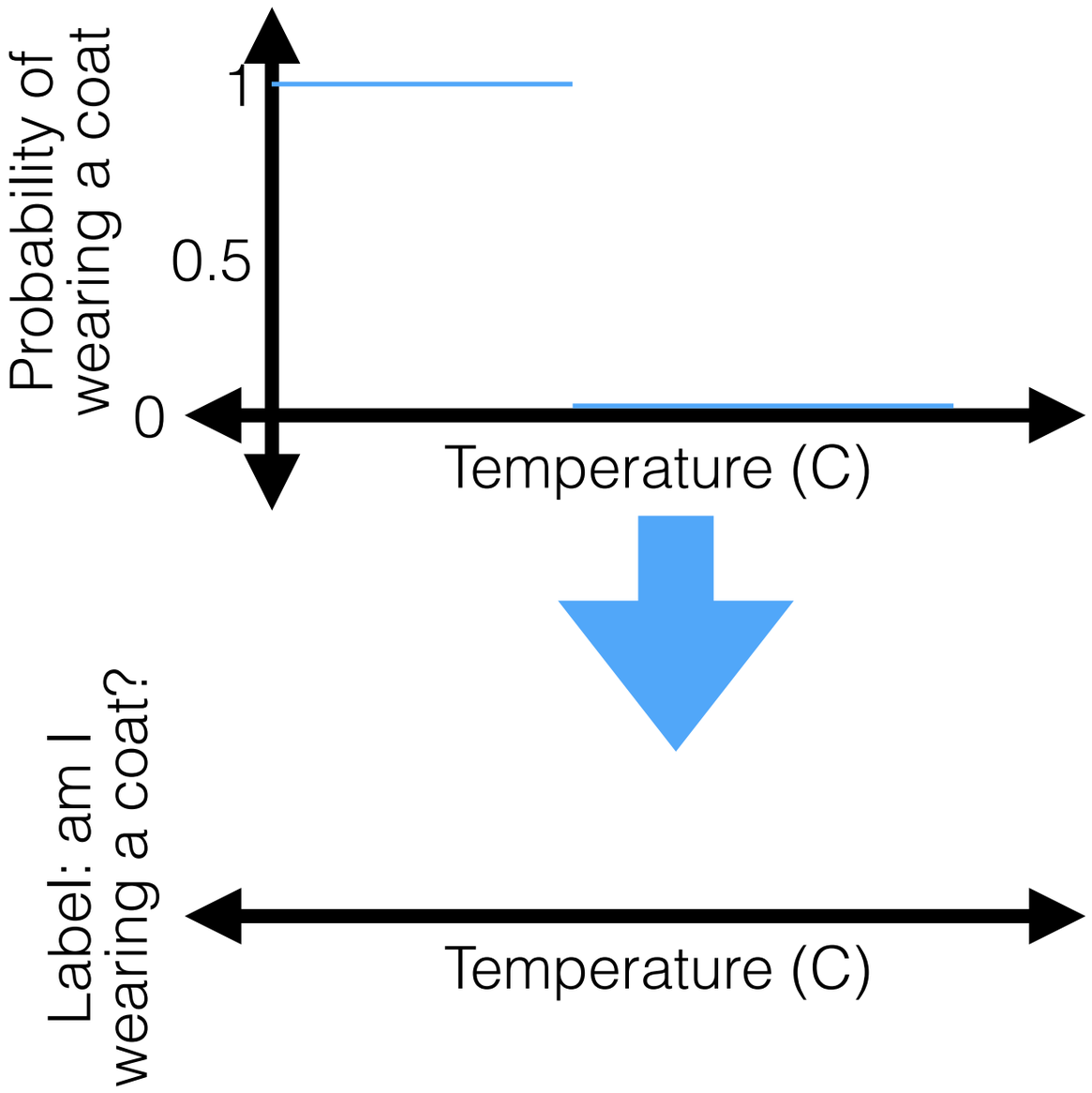
Capturing uncertainty



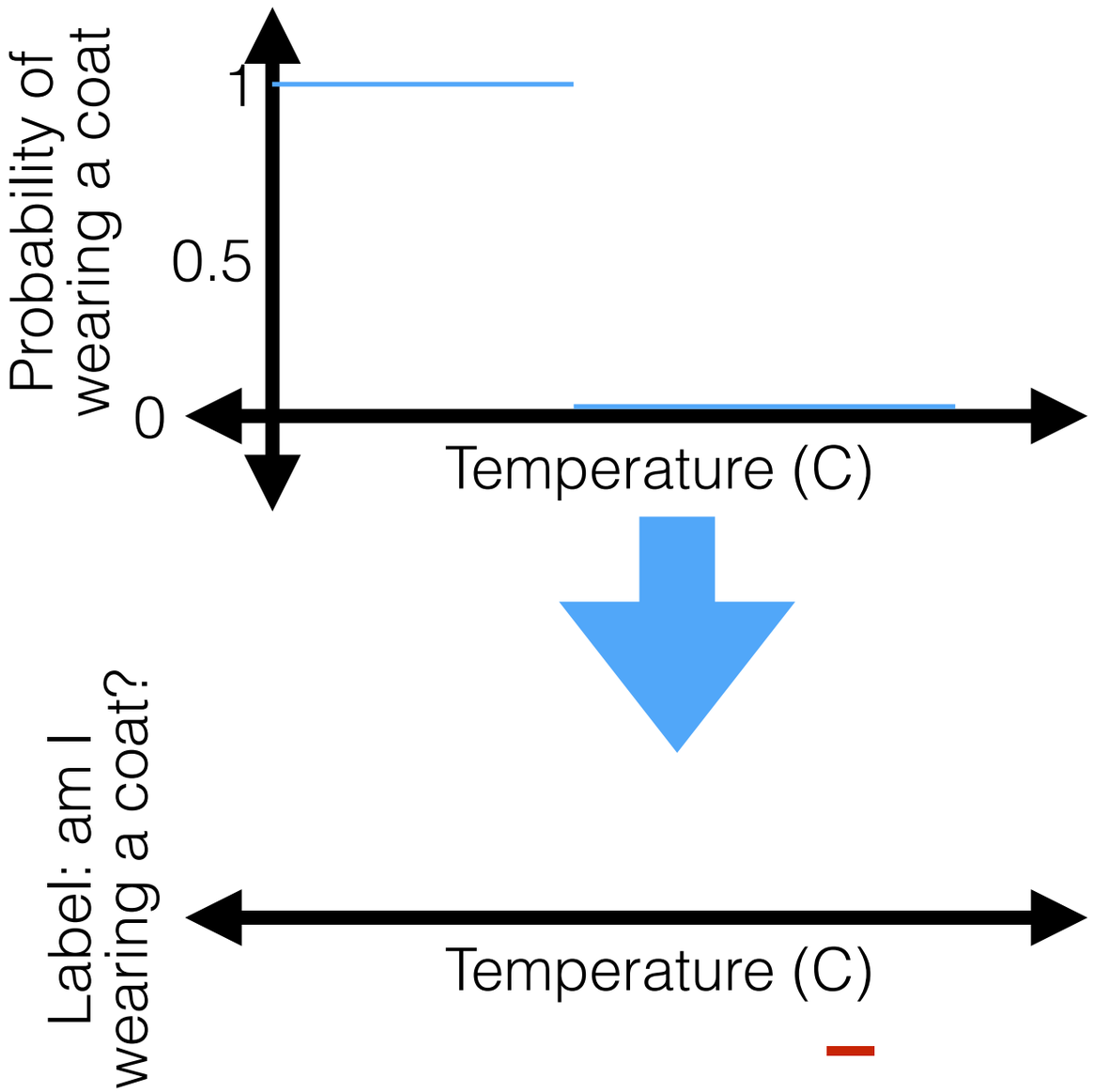
Capturing uncertainty



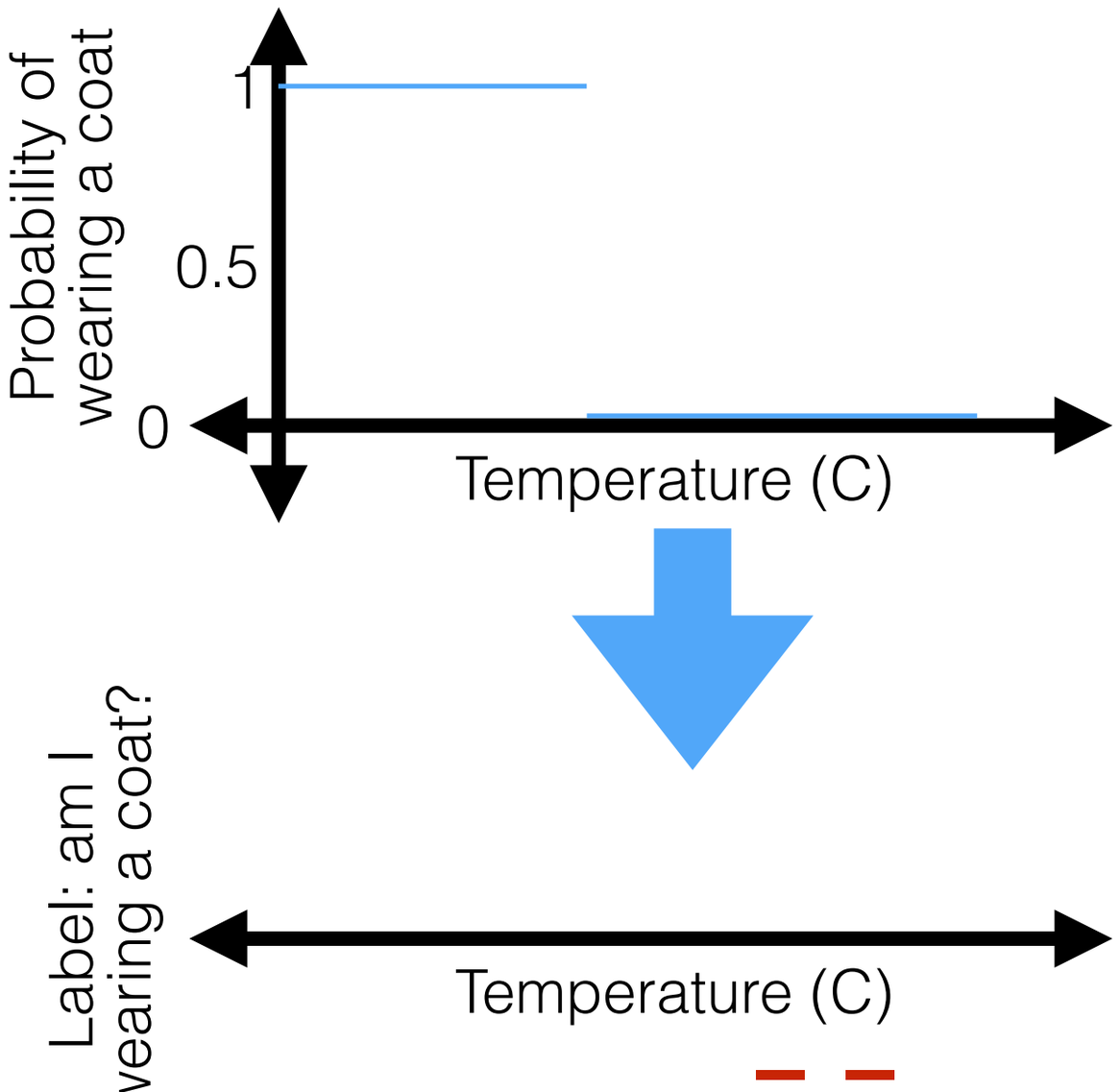
Capturing uncertainty



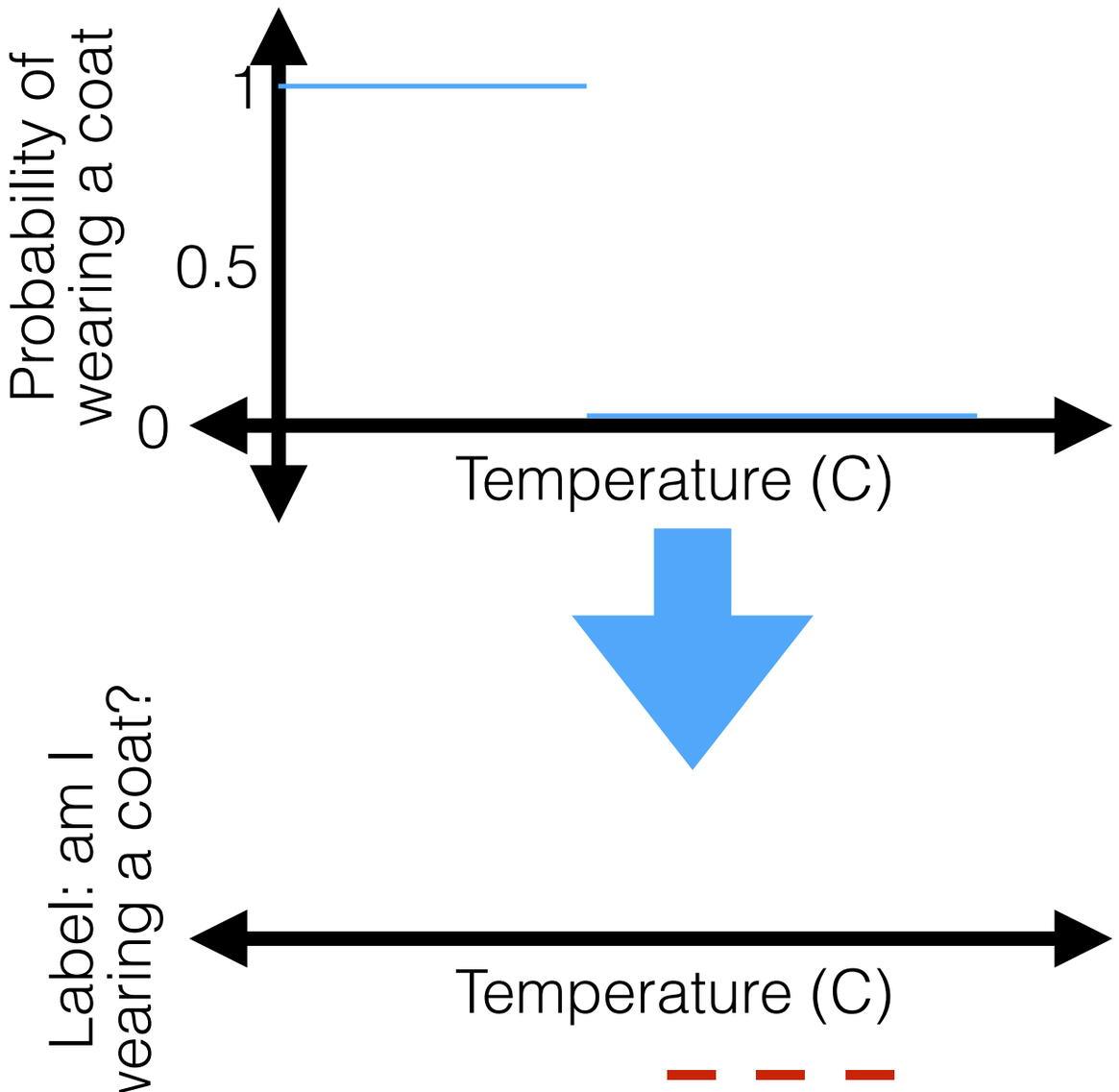
Capturing uncertainty



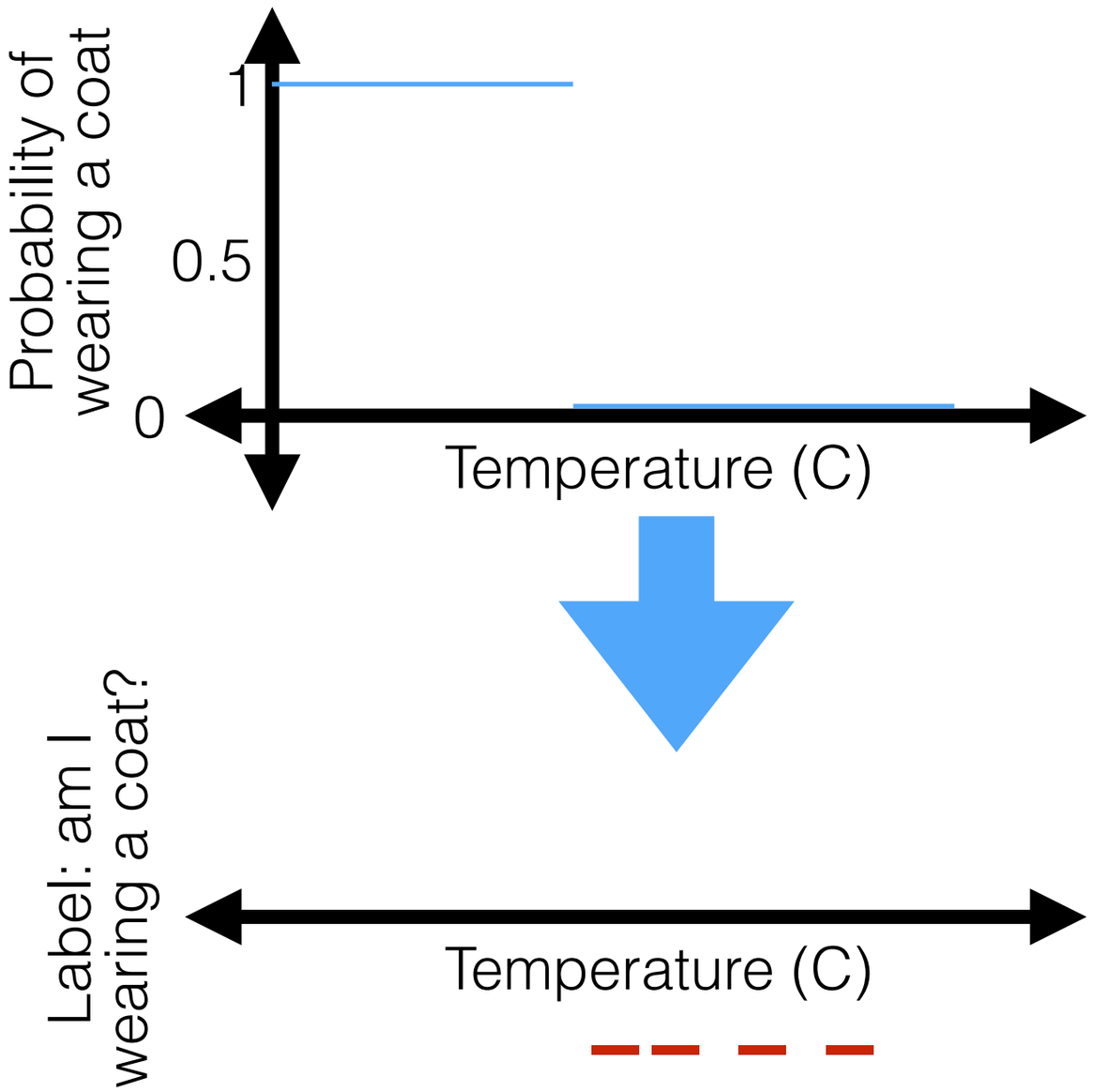
Capturing uncertainty



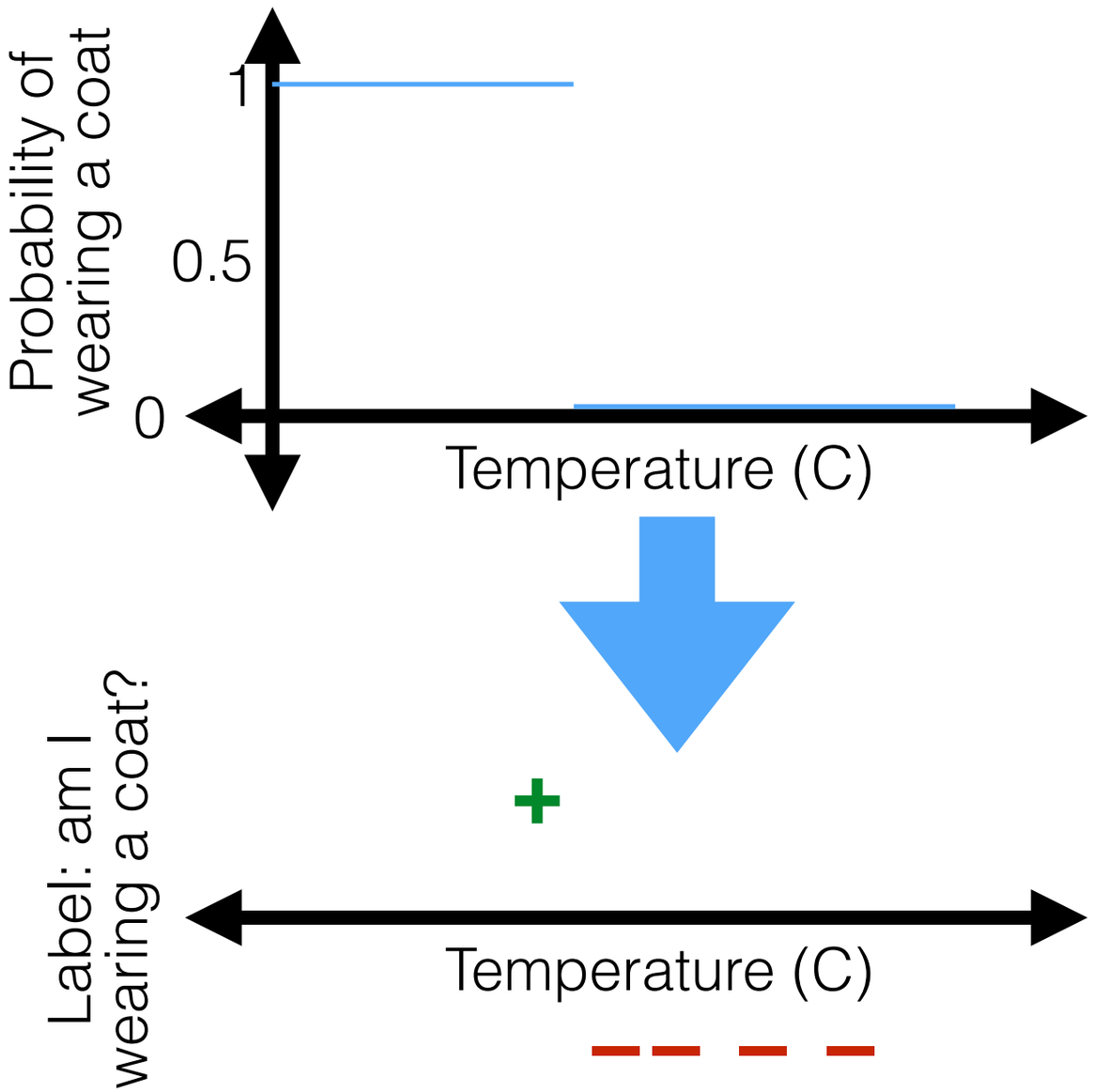
Capturing uncertainty



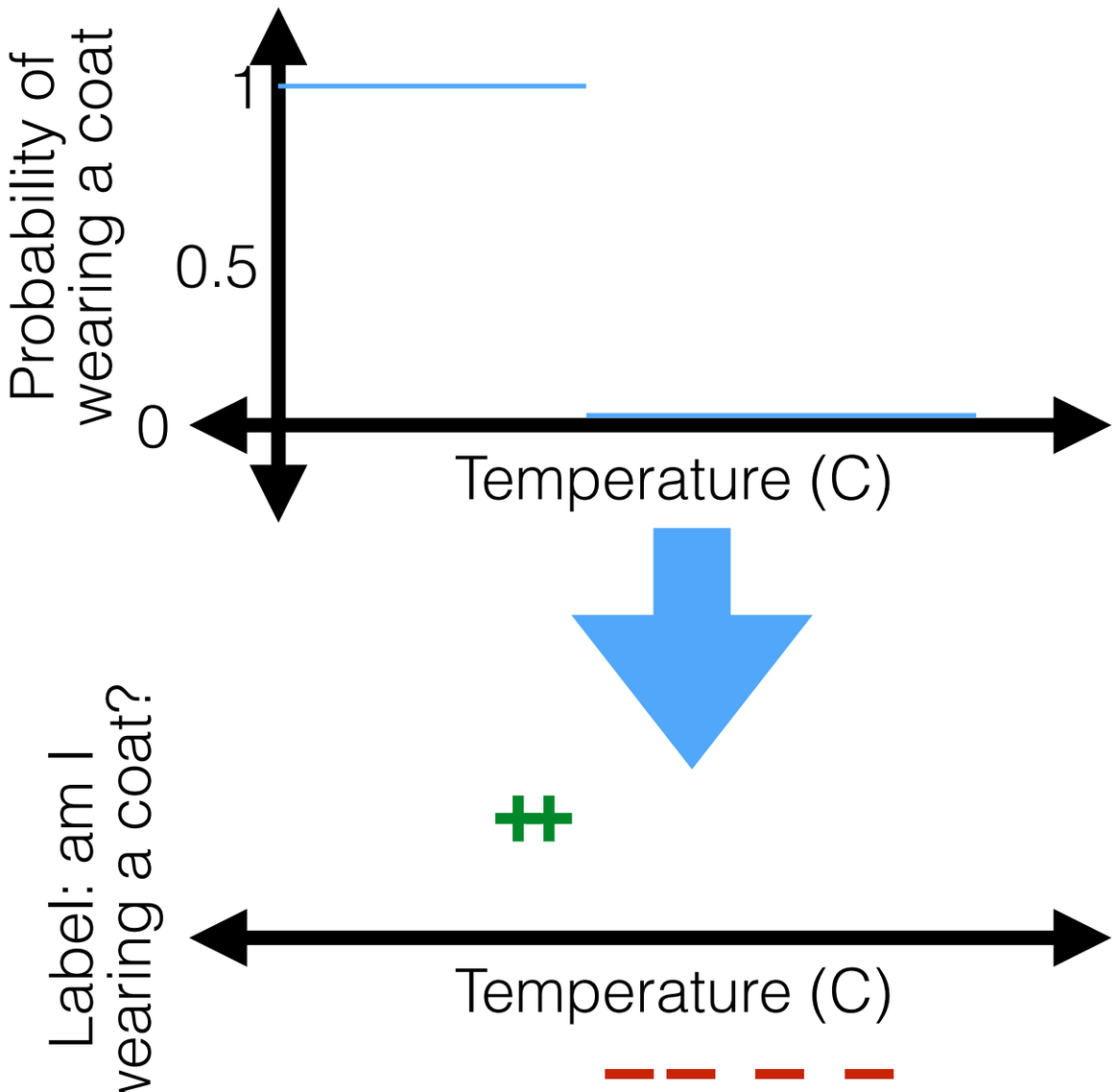
Capturing uncertainty



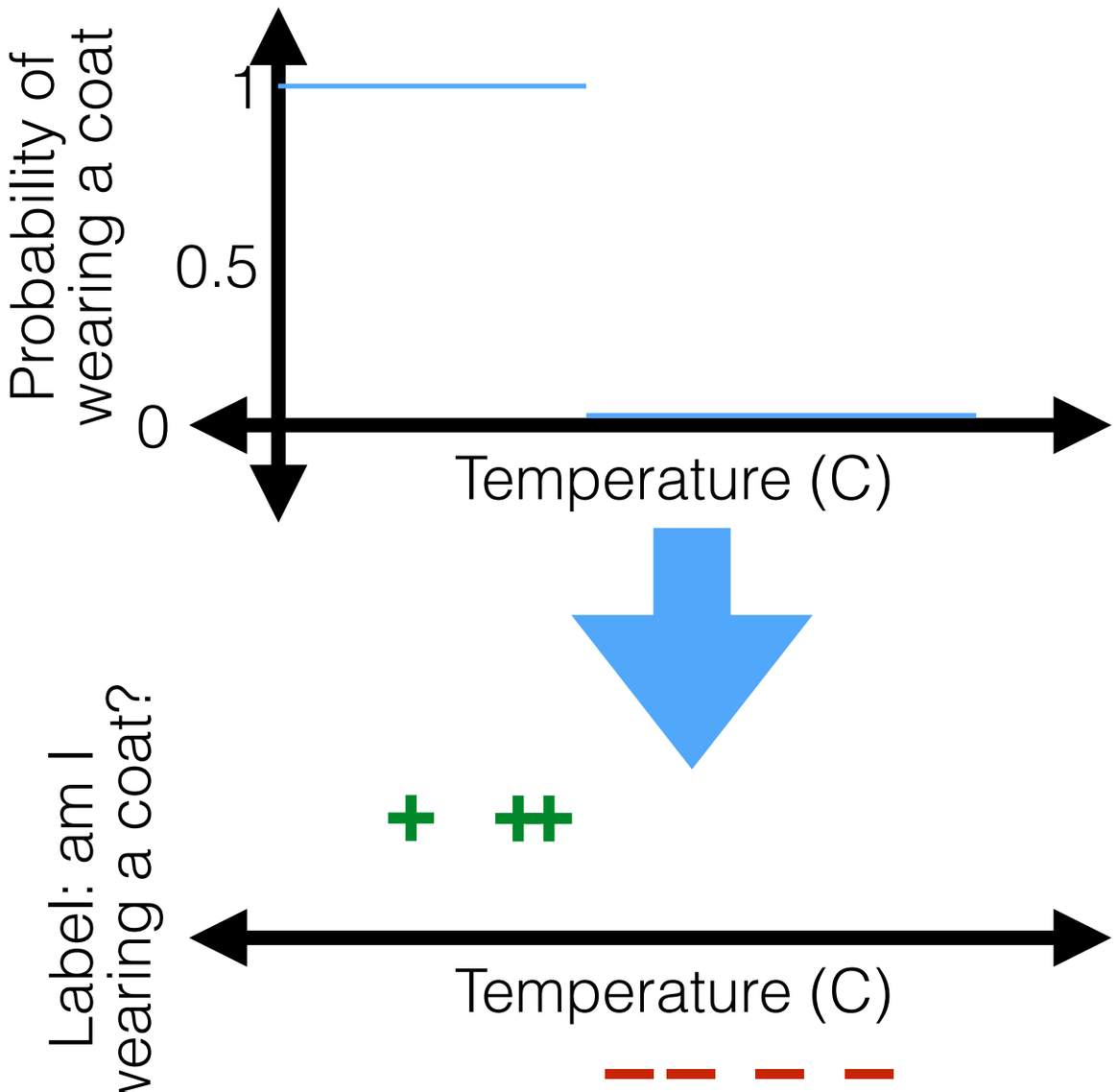
Capturing uncertainty



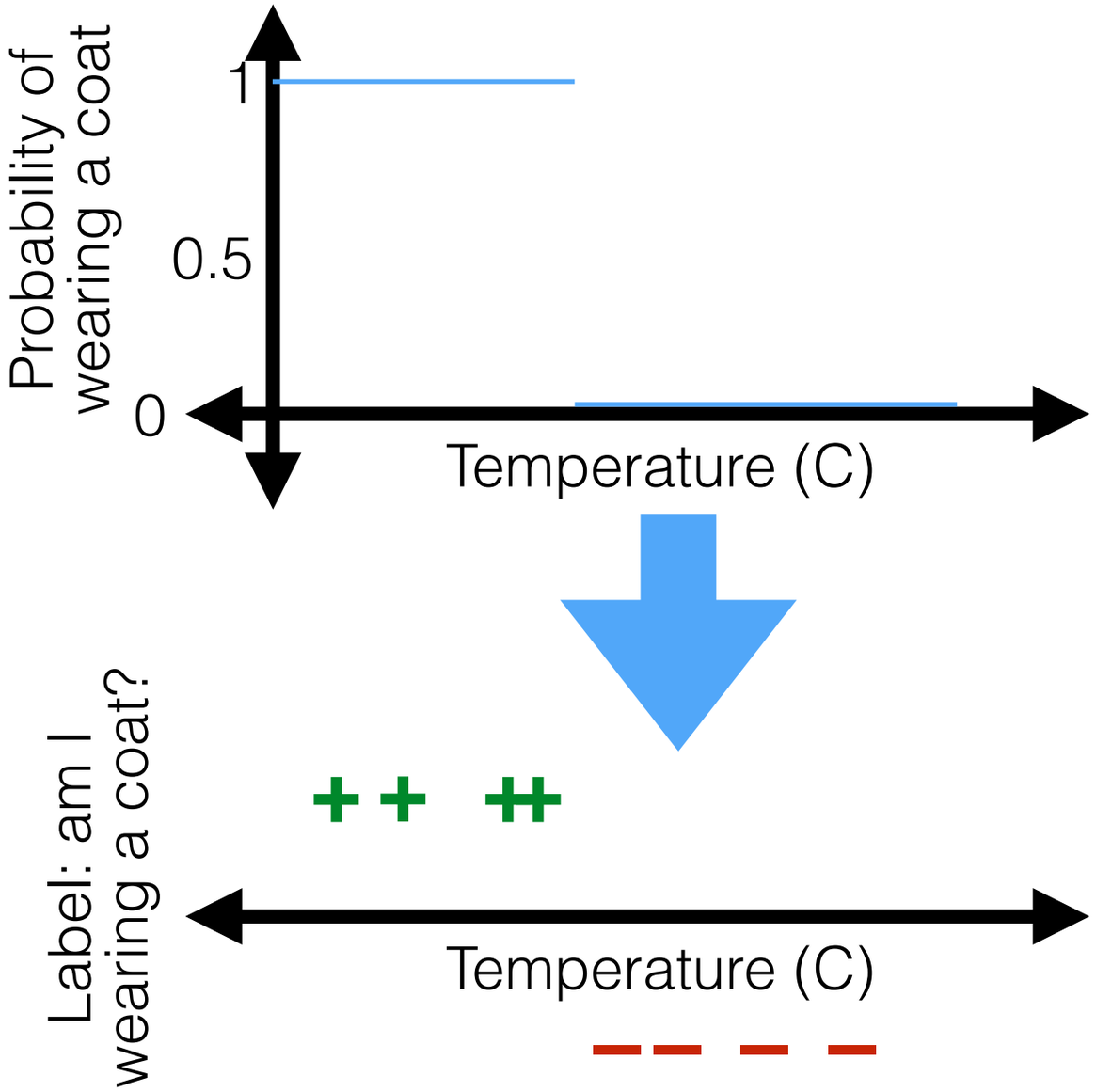
Capturing uncertainty



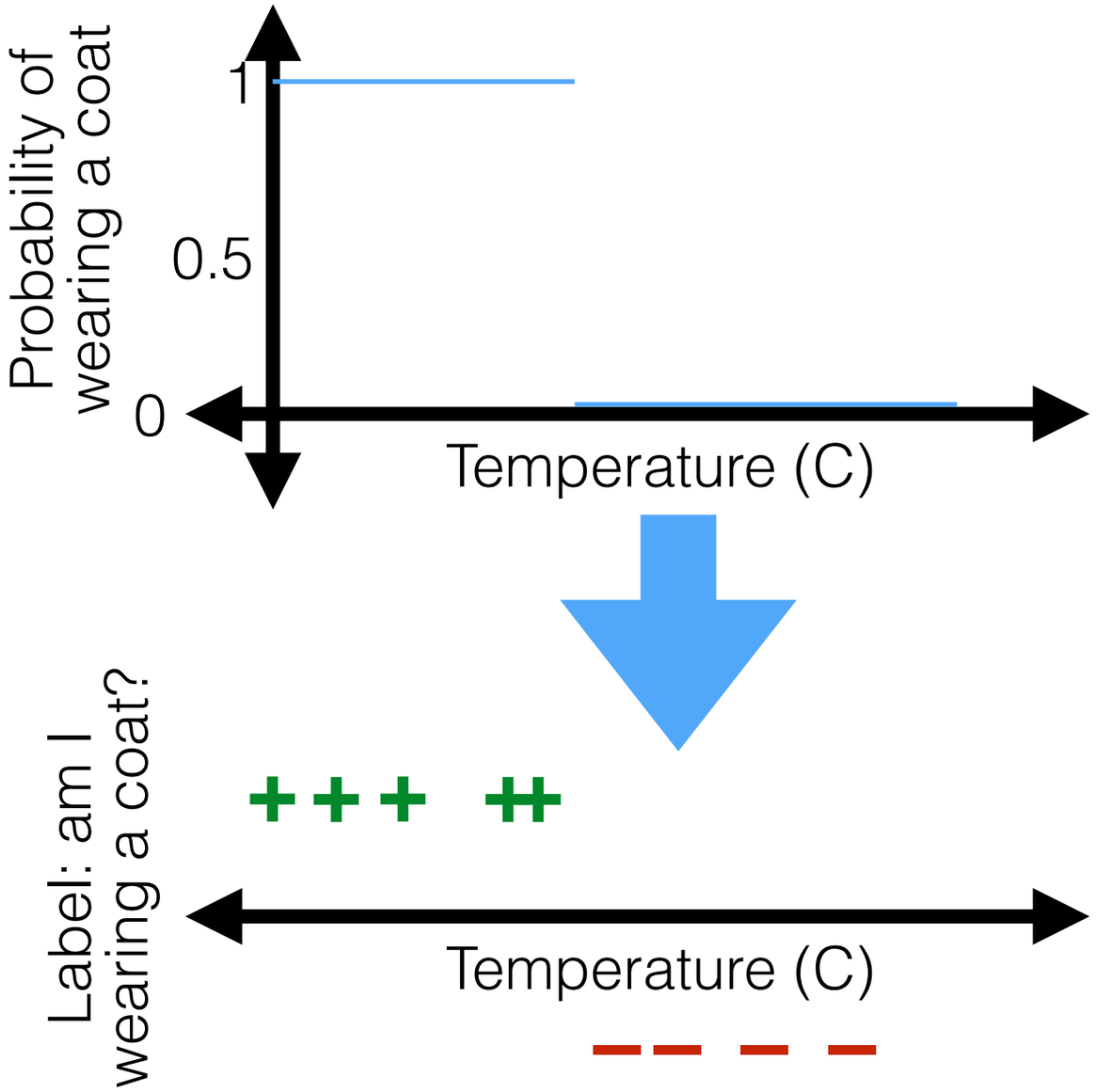
Capturing uncertainty



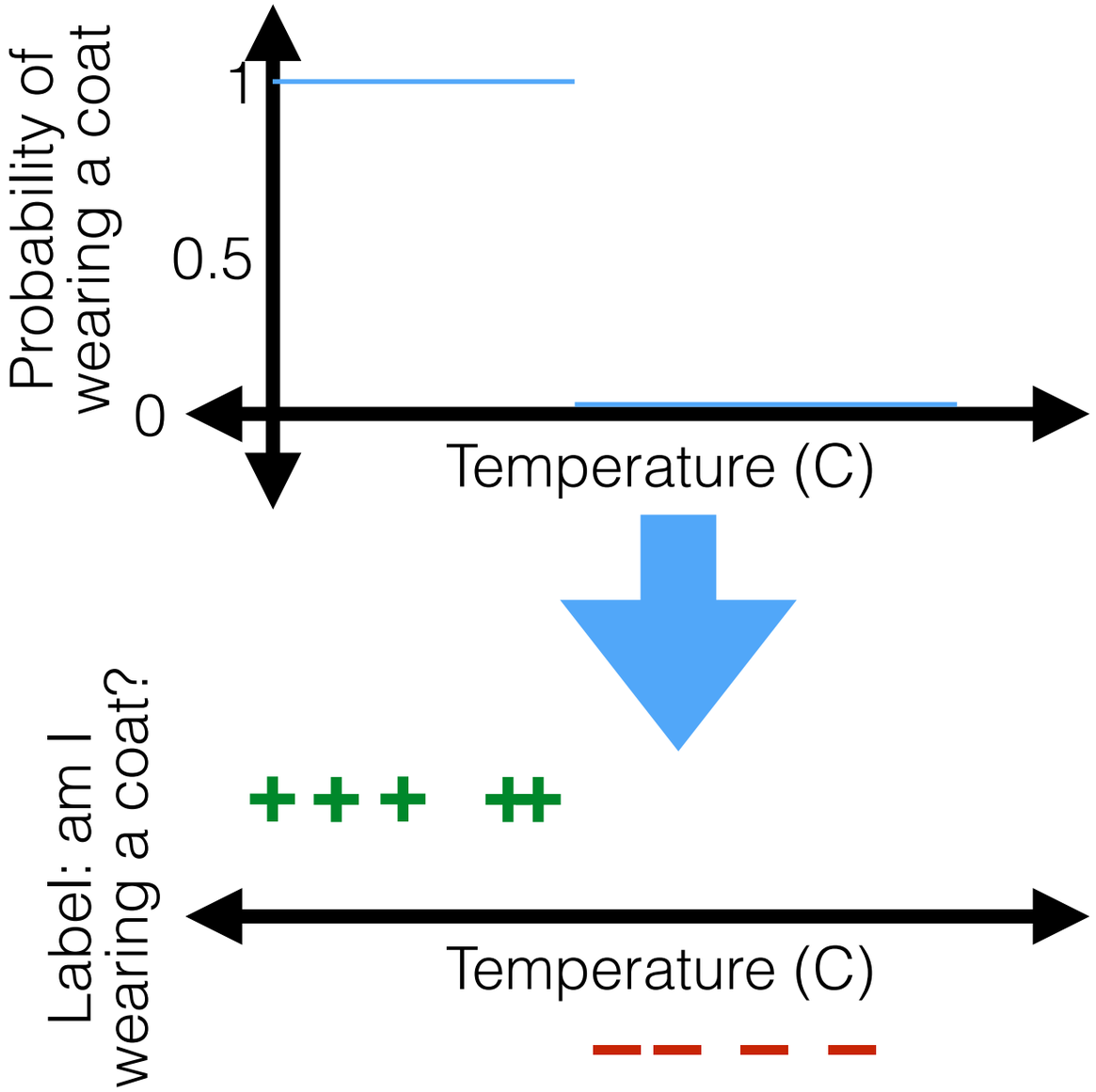
Capturing uncertainty



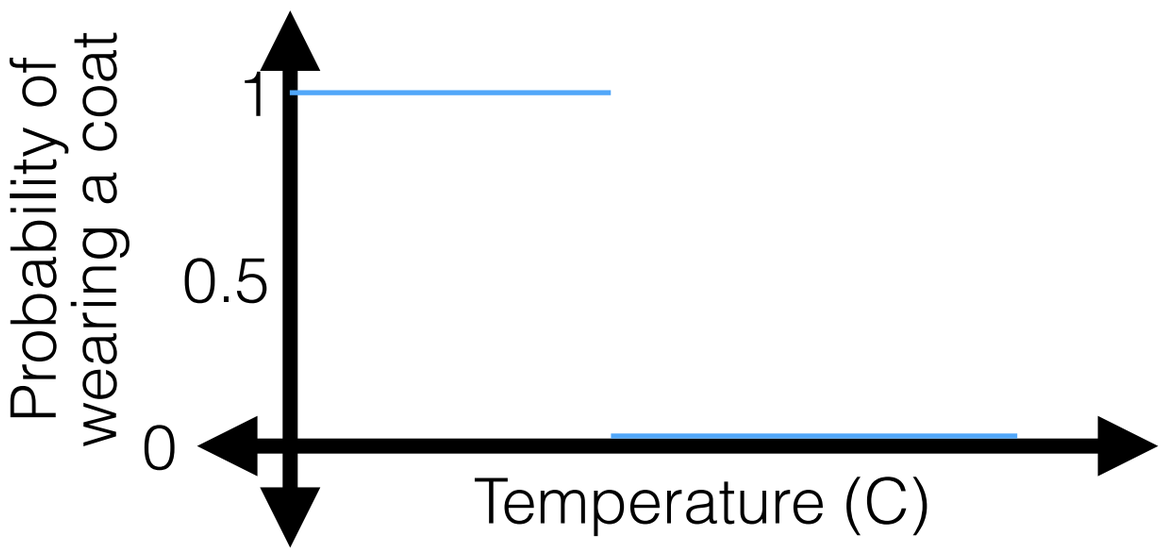
Capturing uncertainty

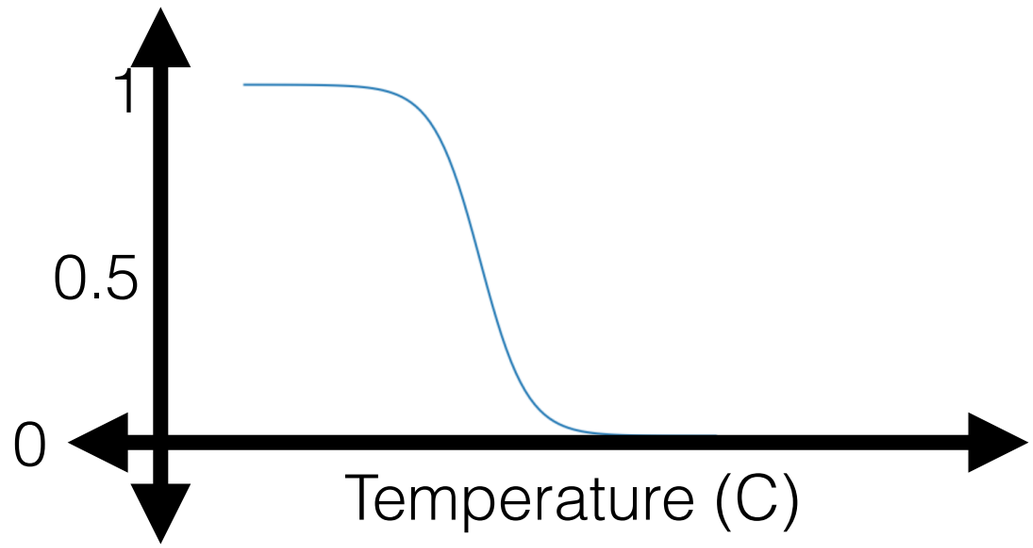


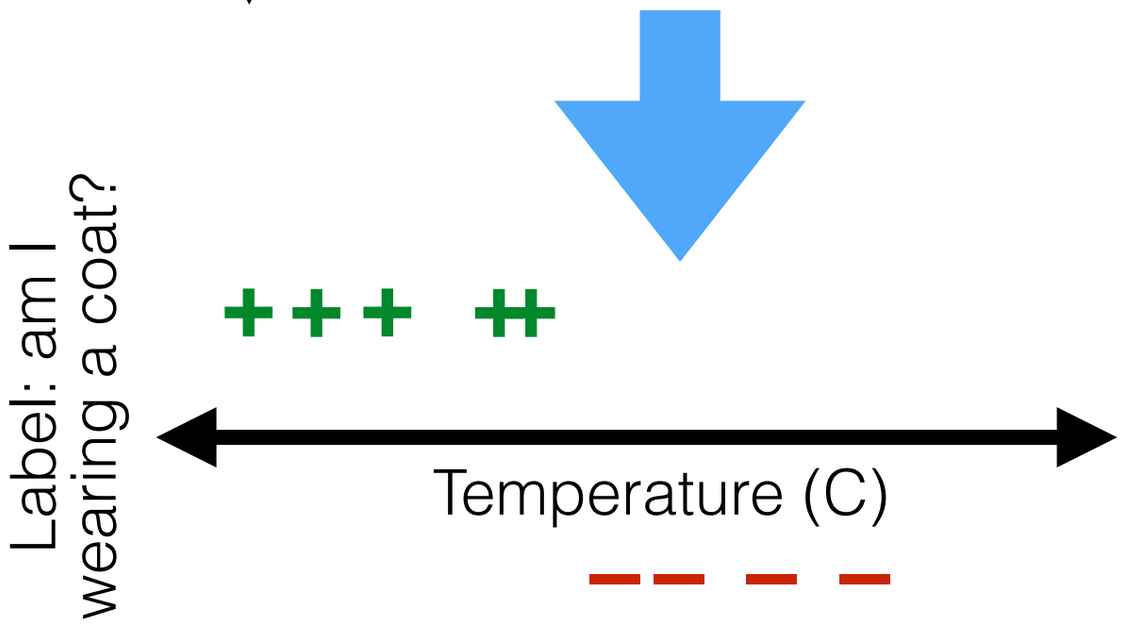
Capturing uncertainty



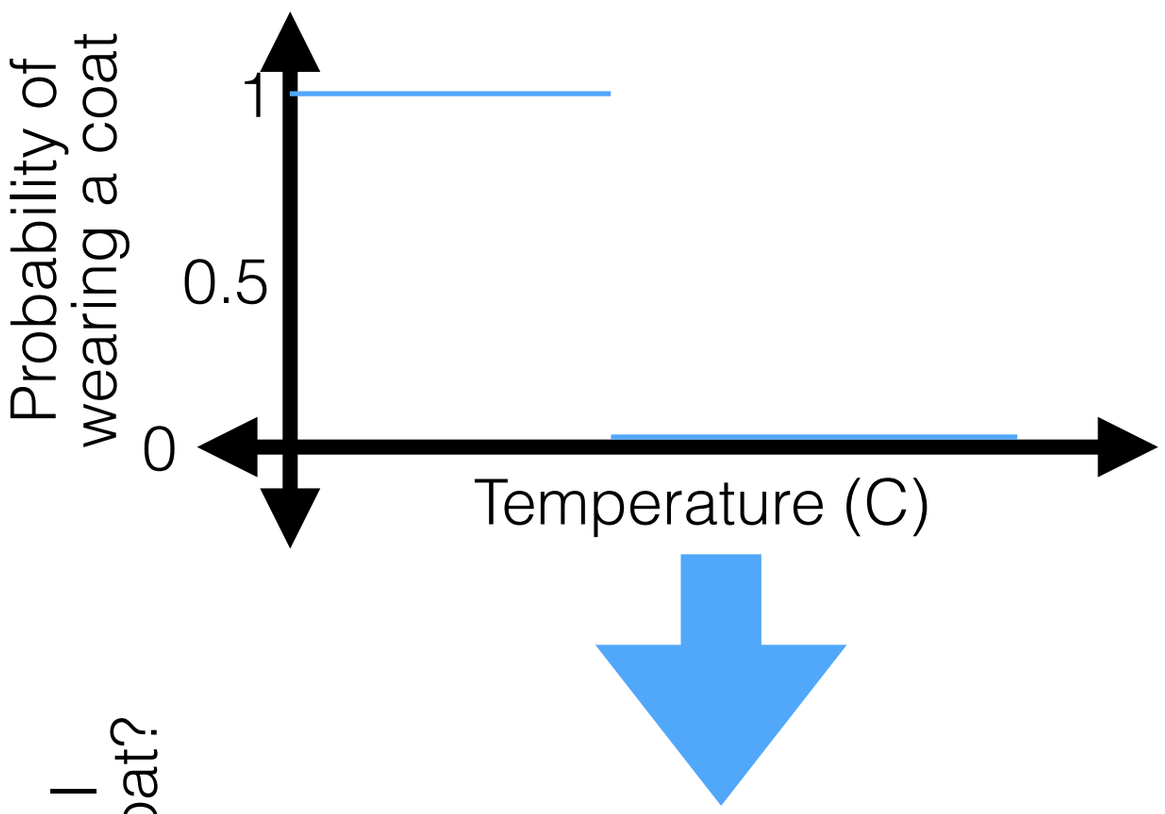
Capturing uncertainty

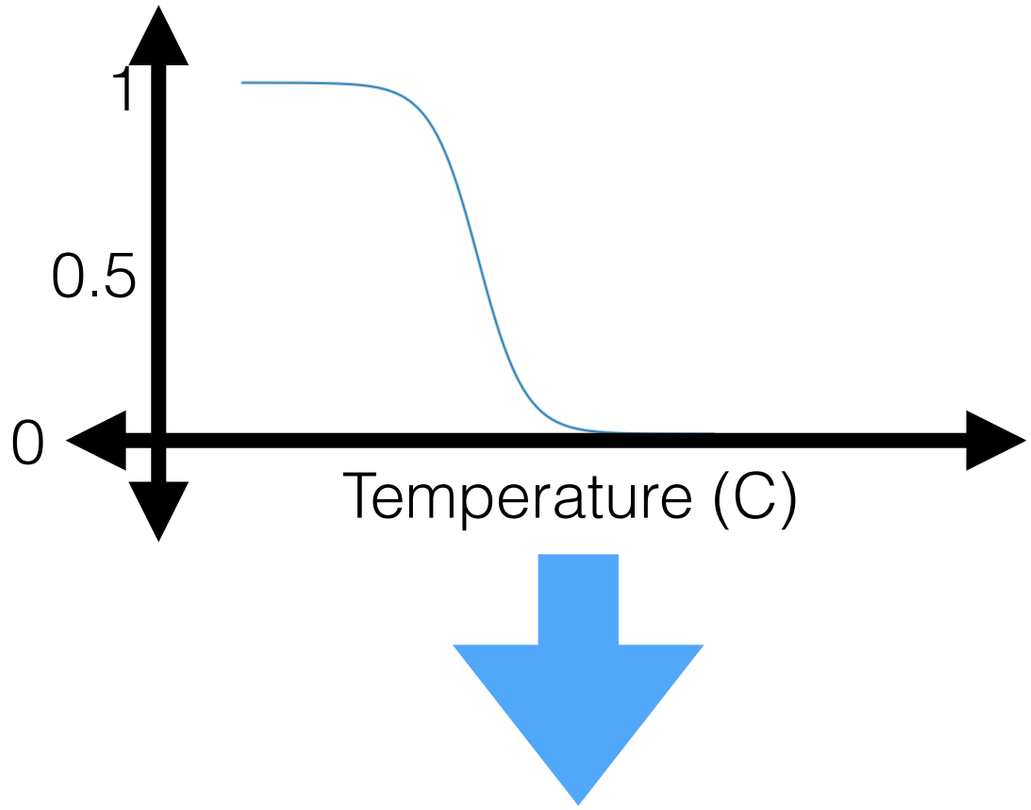


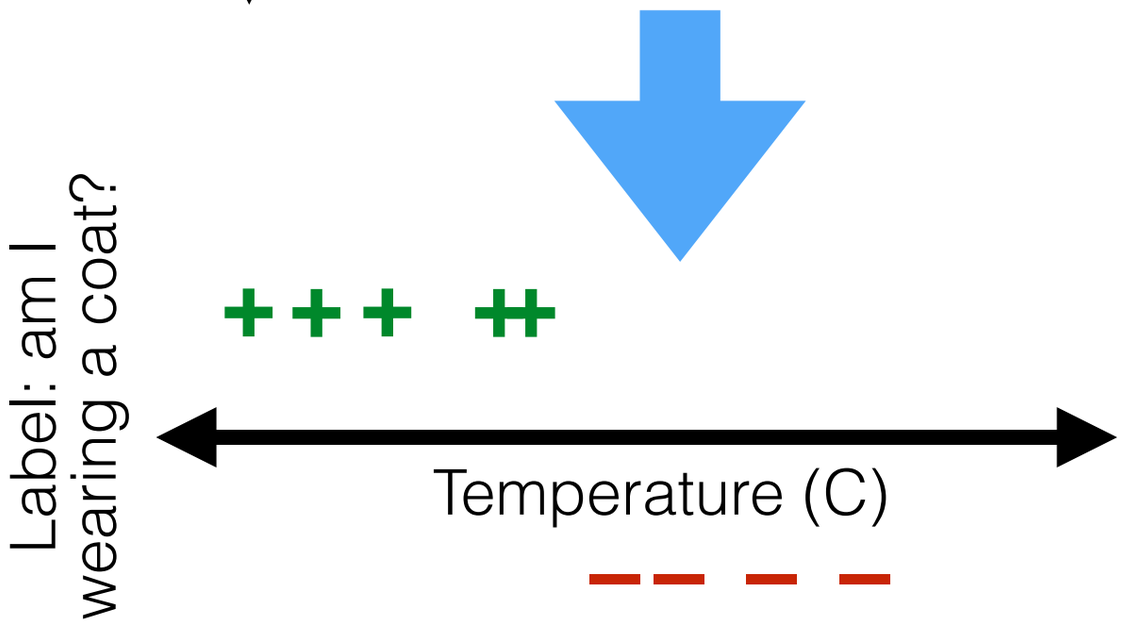


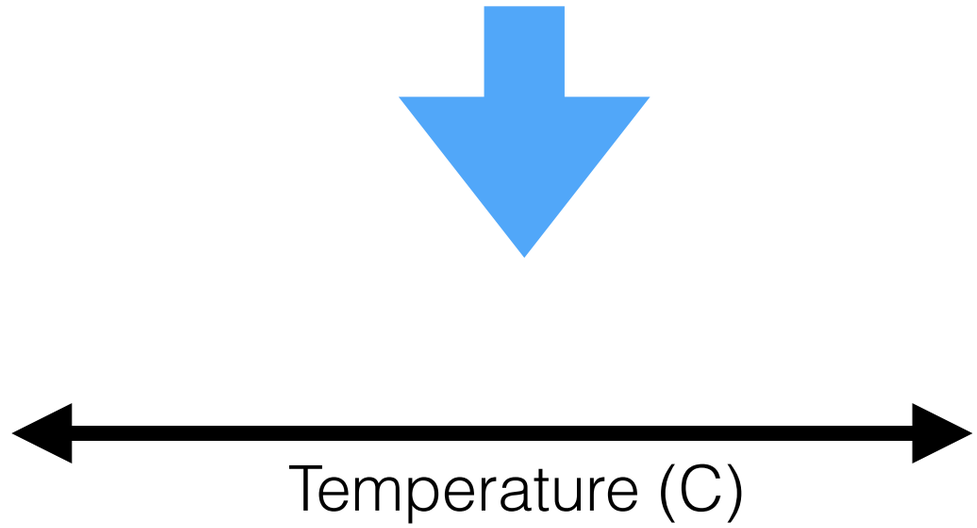


Capturing uncertainty

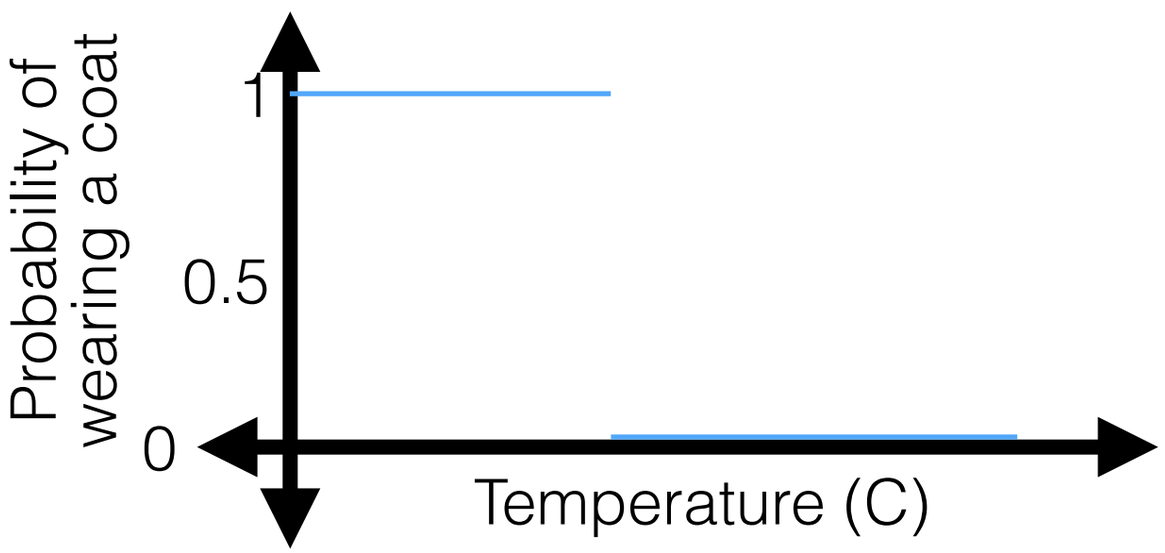


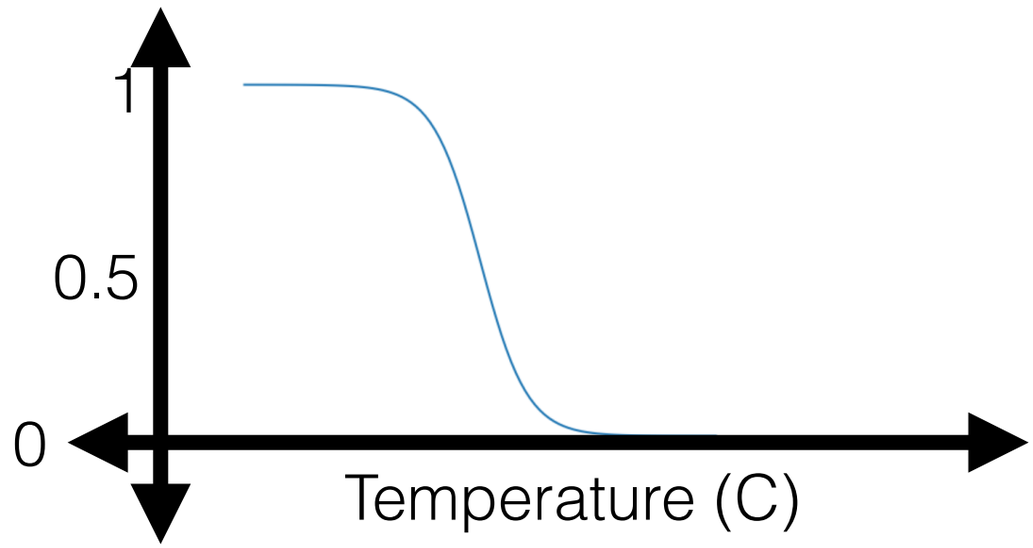


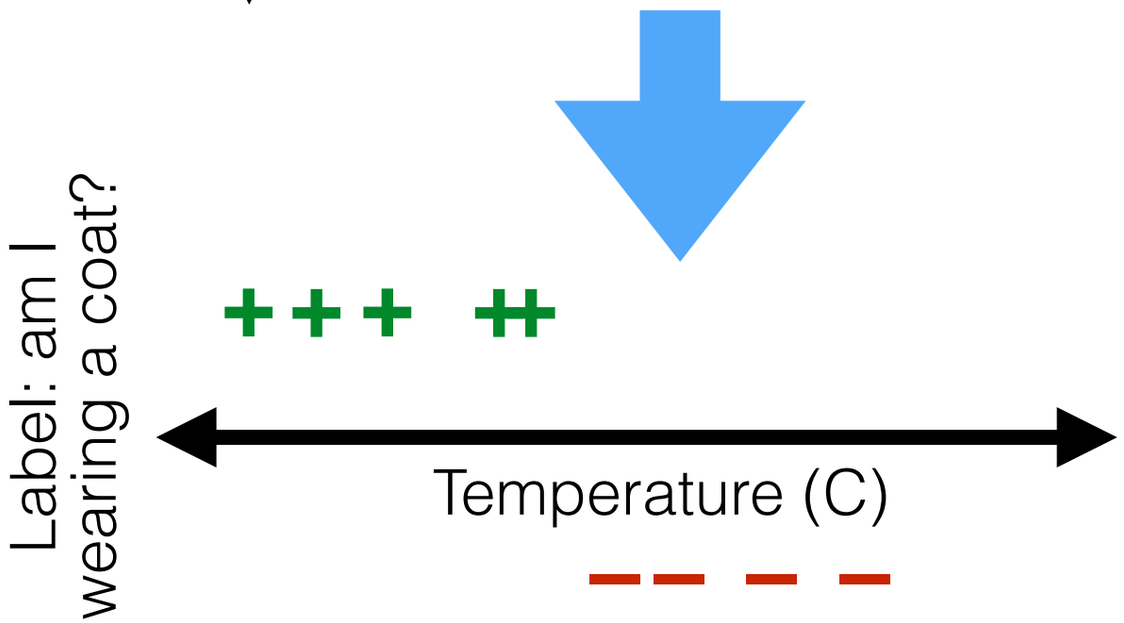


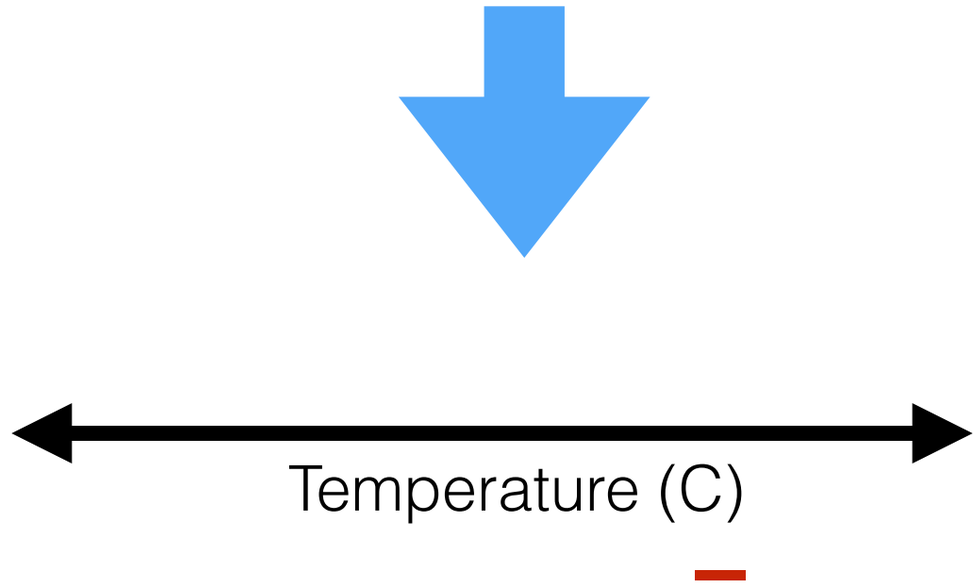


Capturing uncertainty

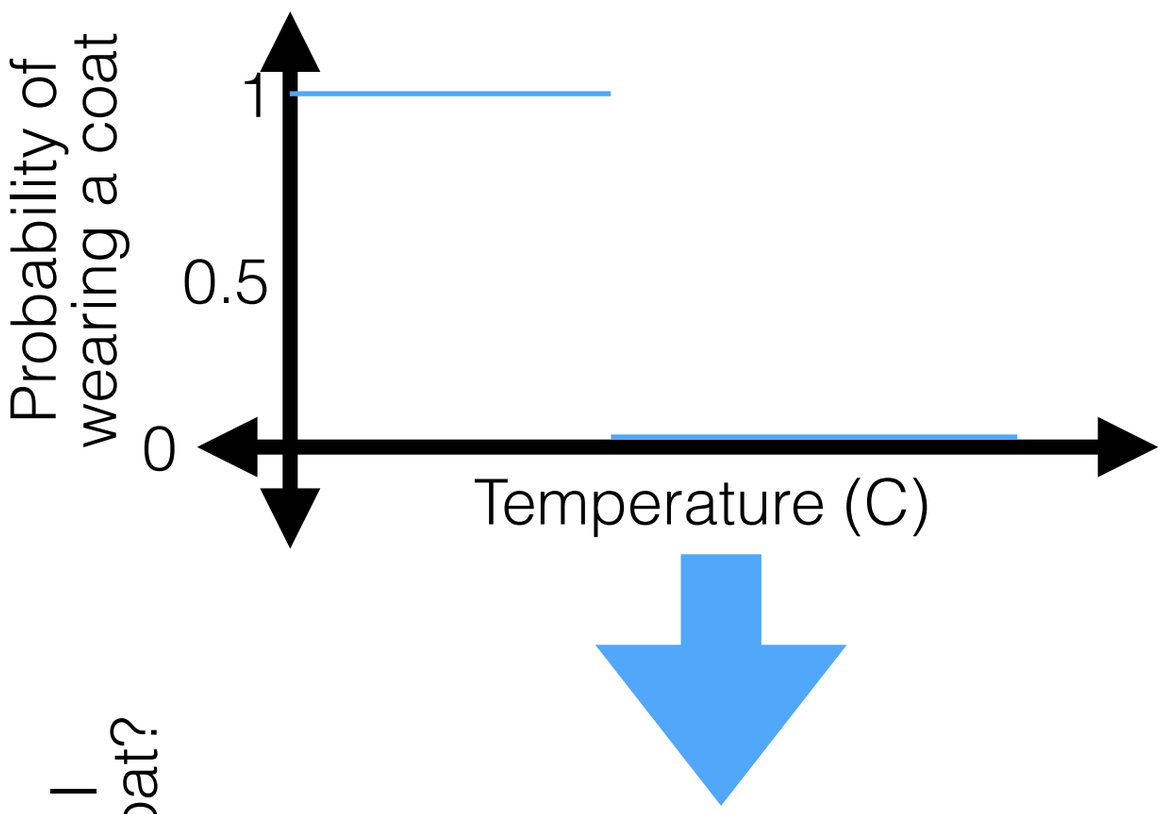


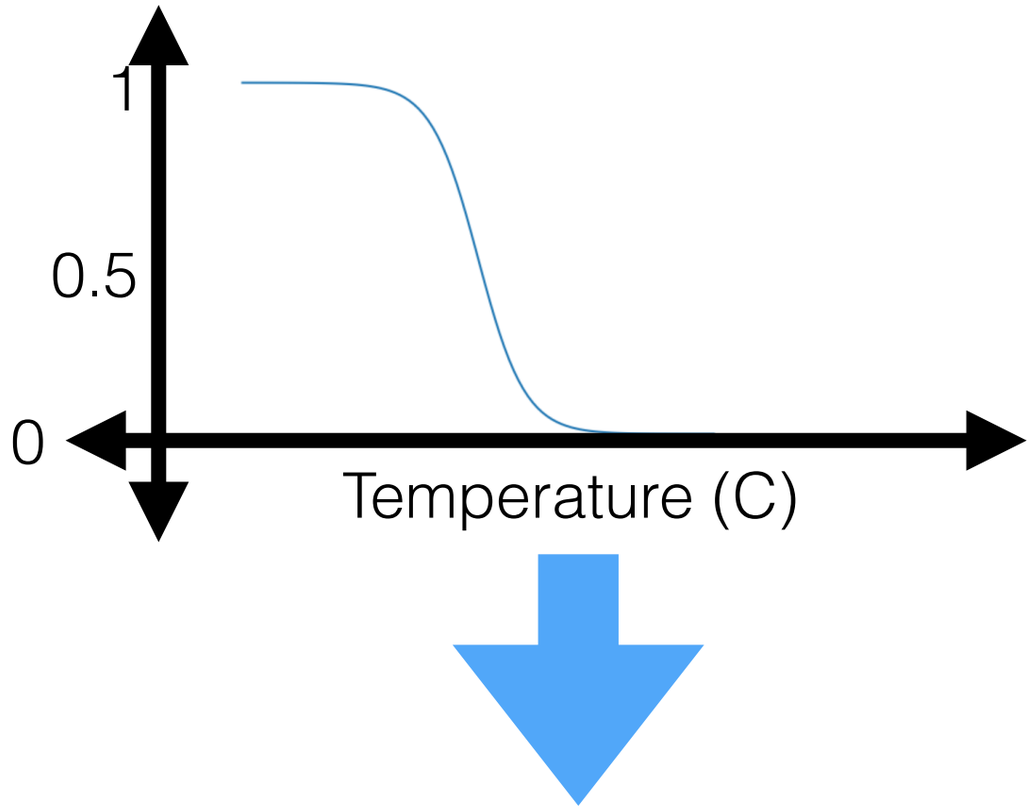


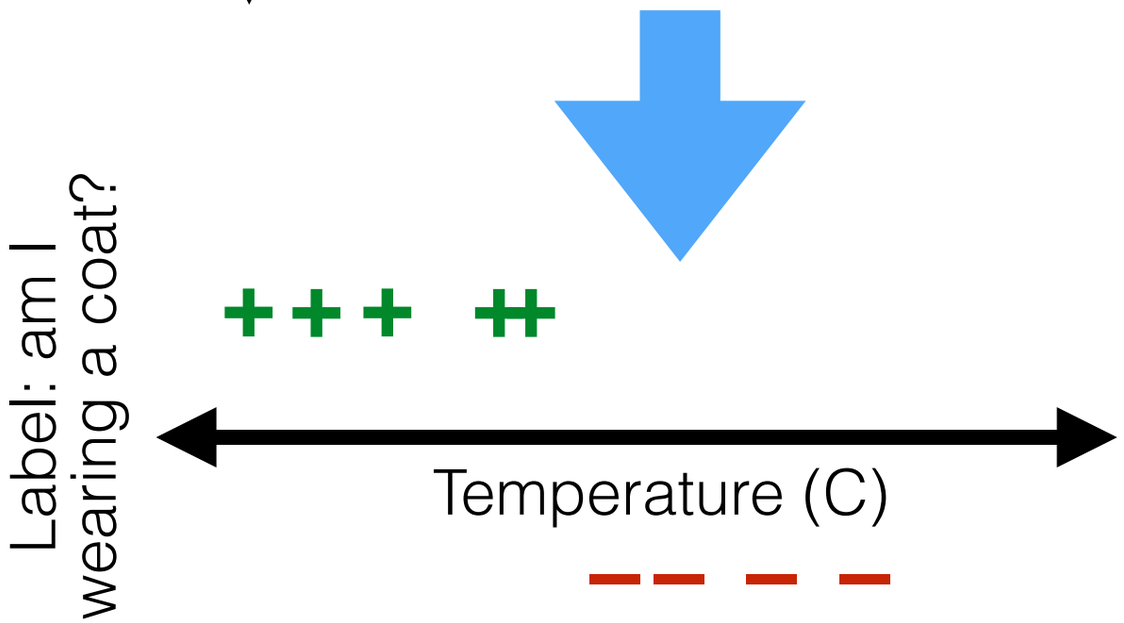


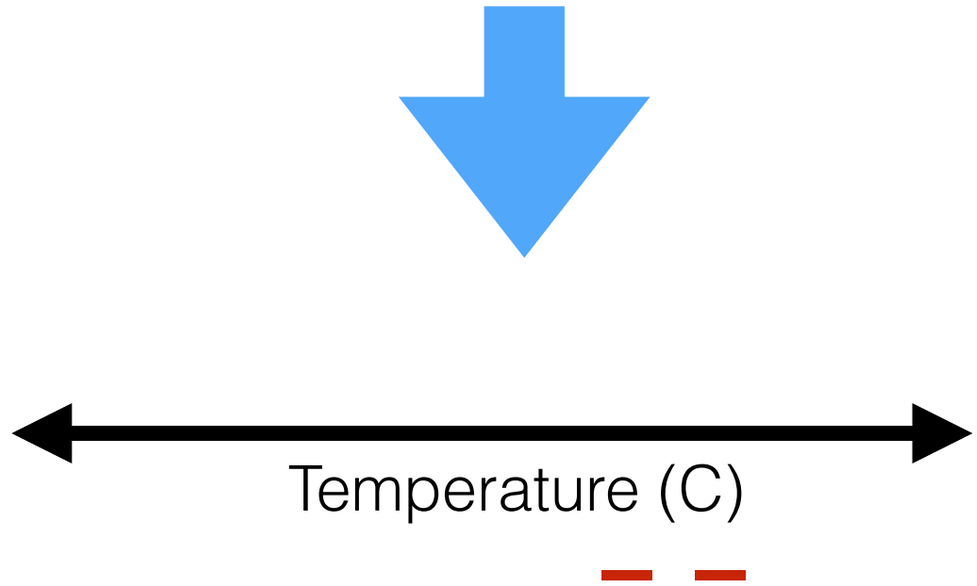


Capturing uncertainty

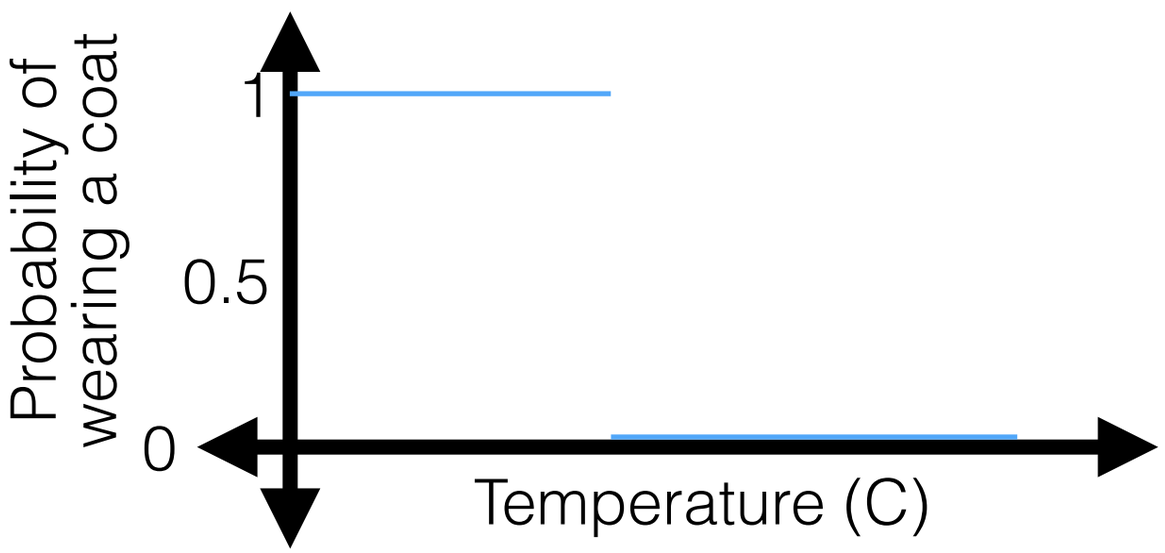


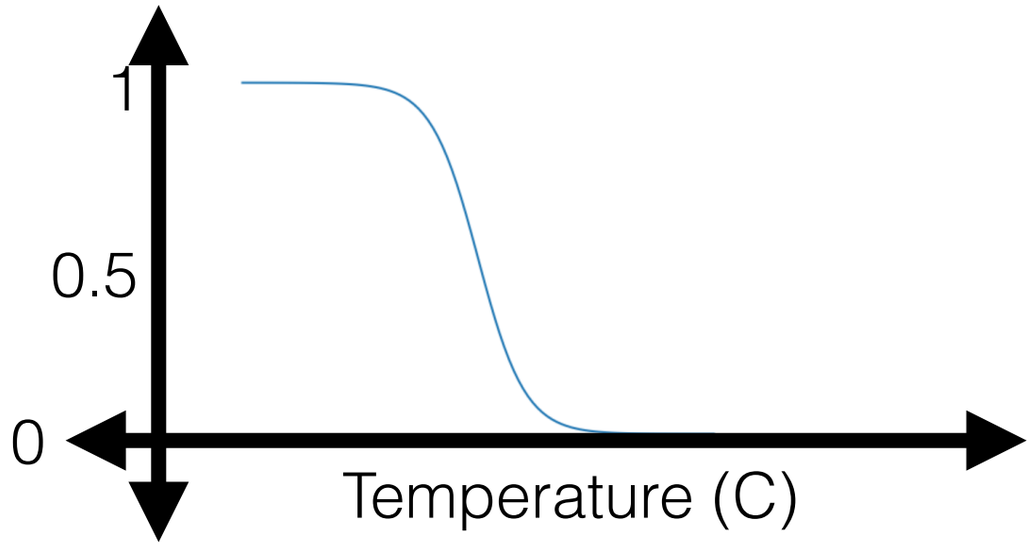


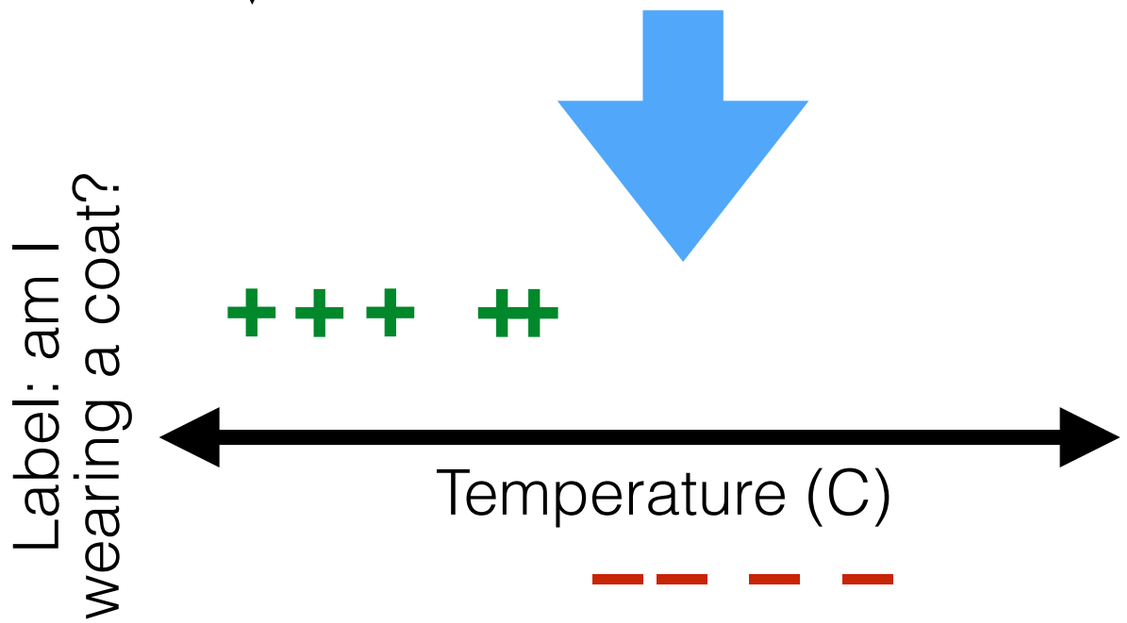


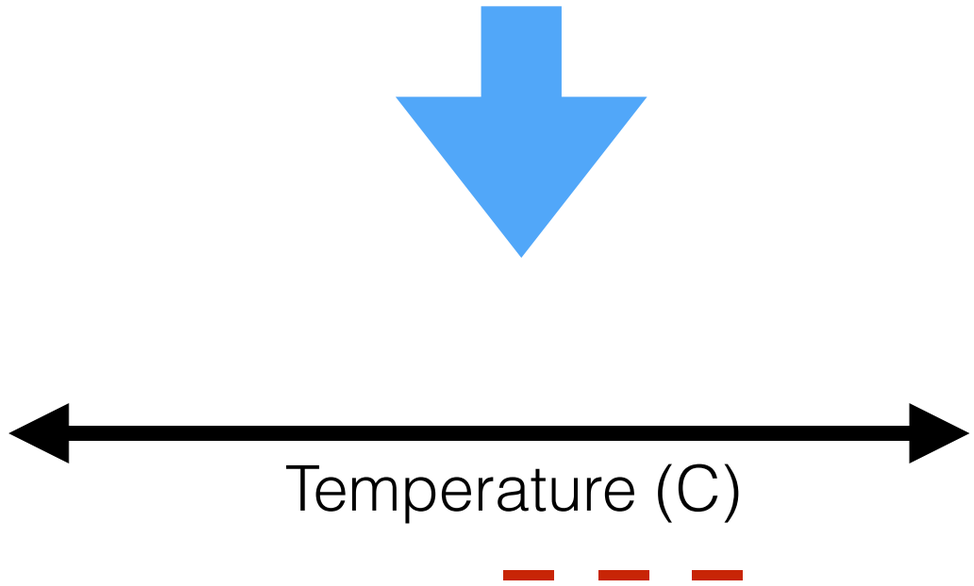


Capturing uncertainty

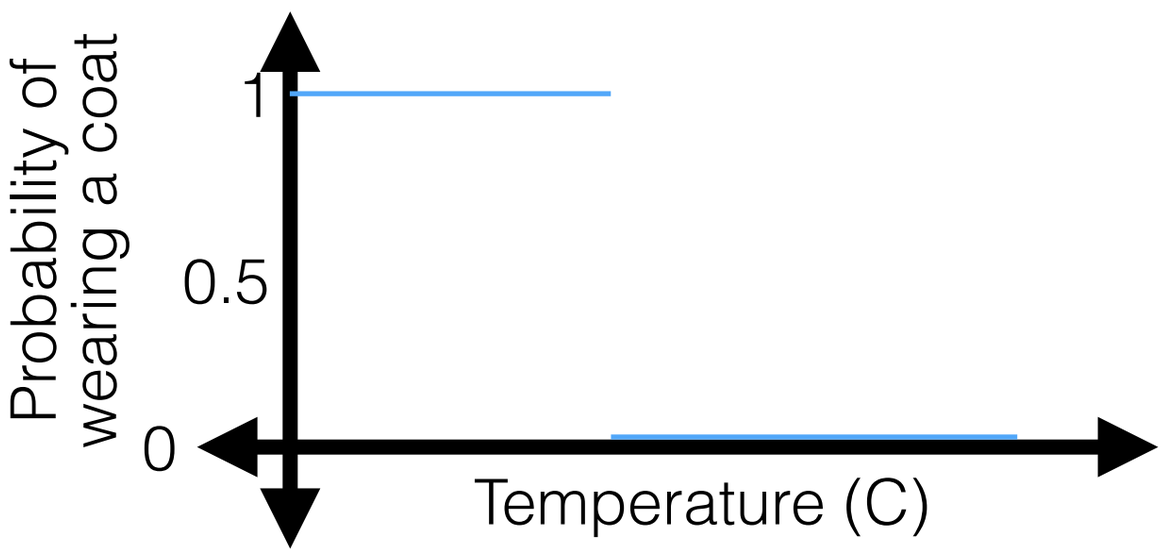


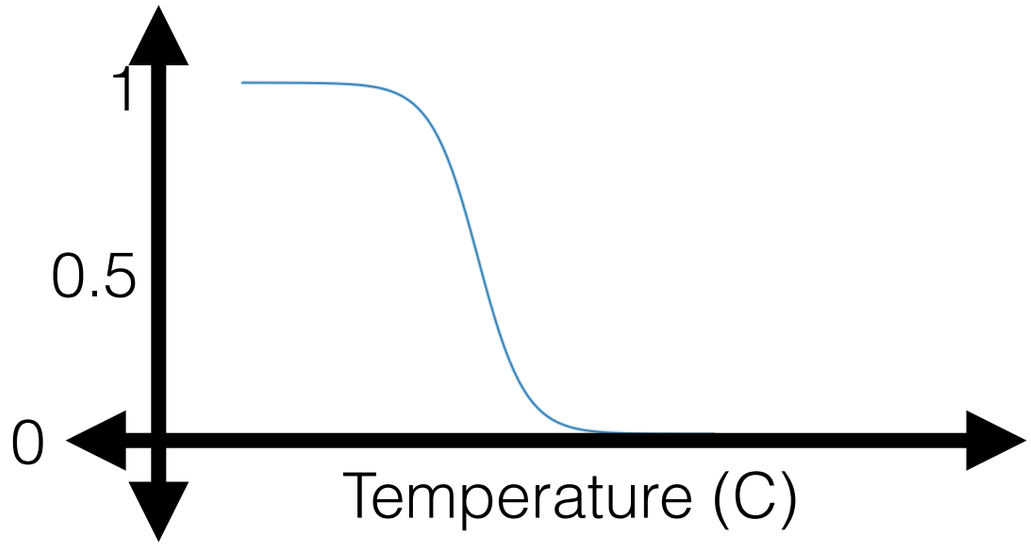


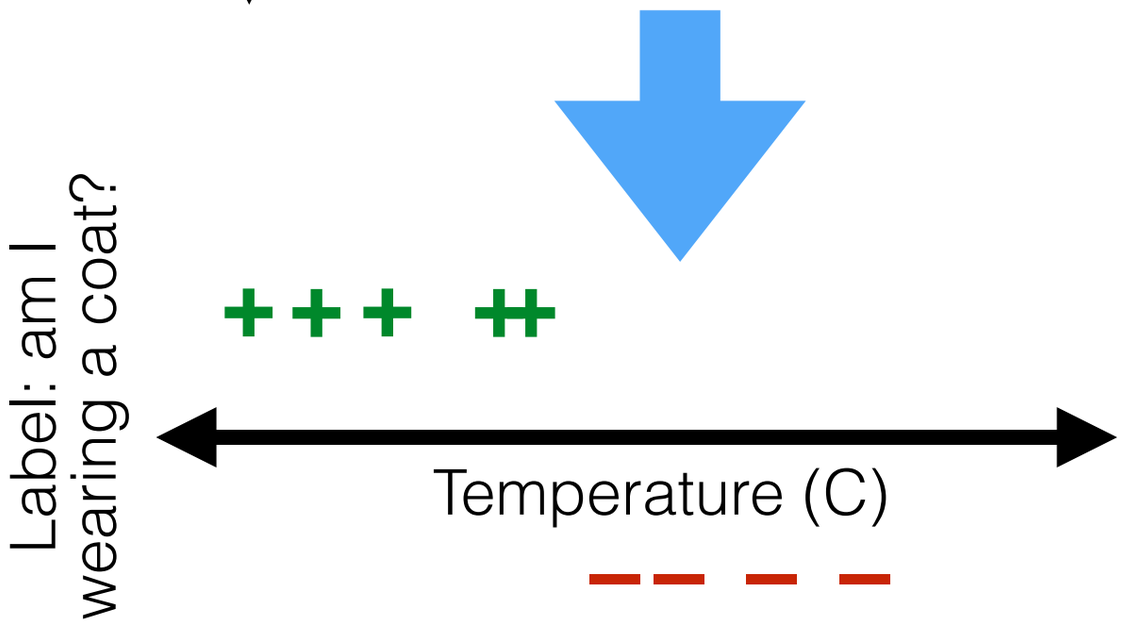


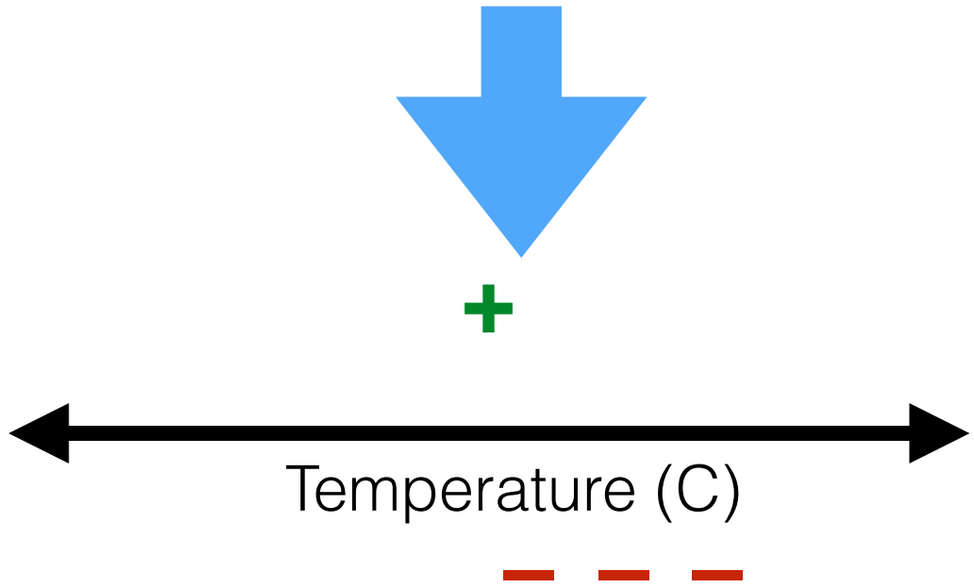


Capturing uncertainty

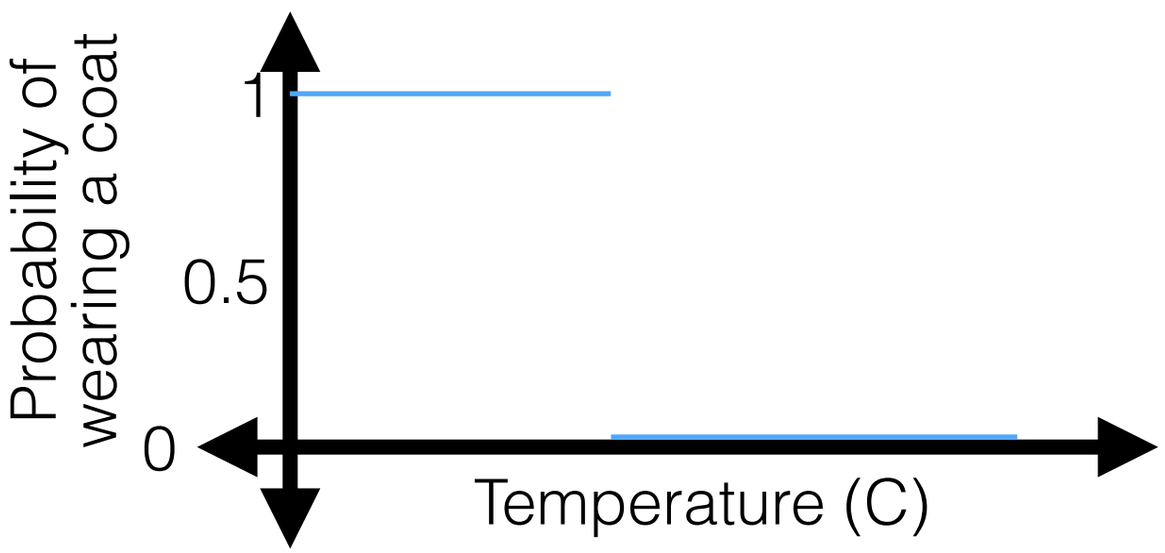


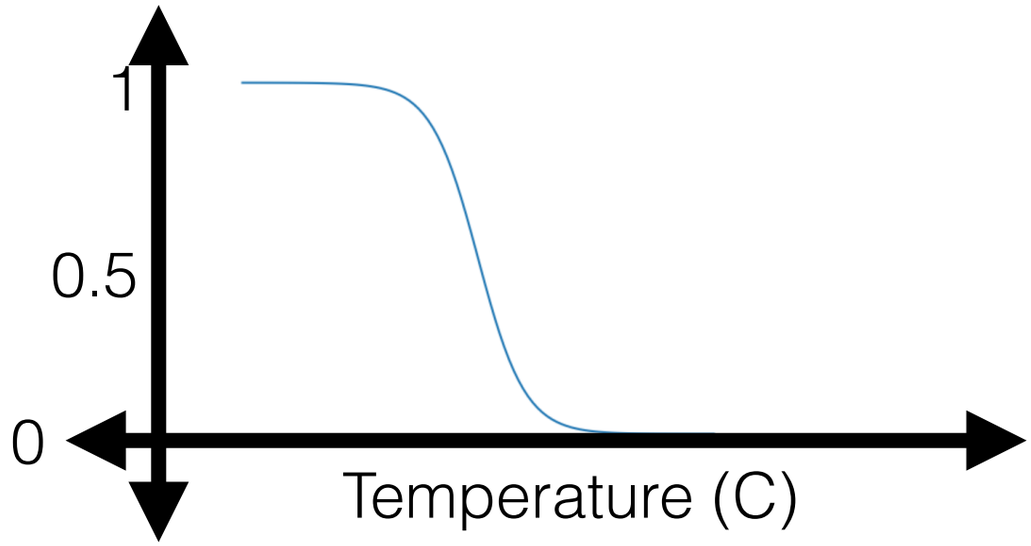


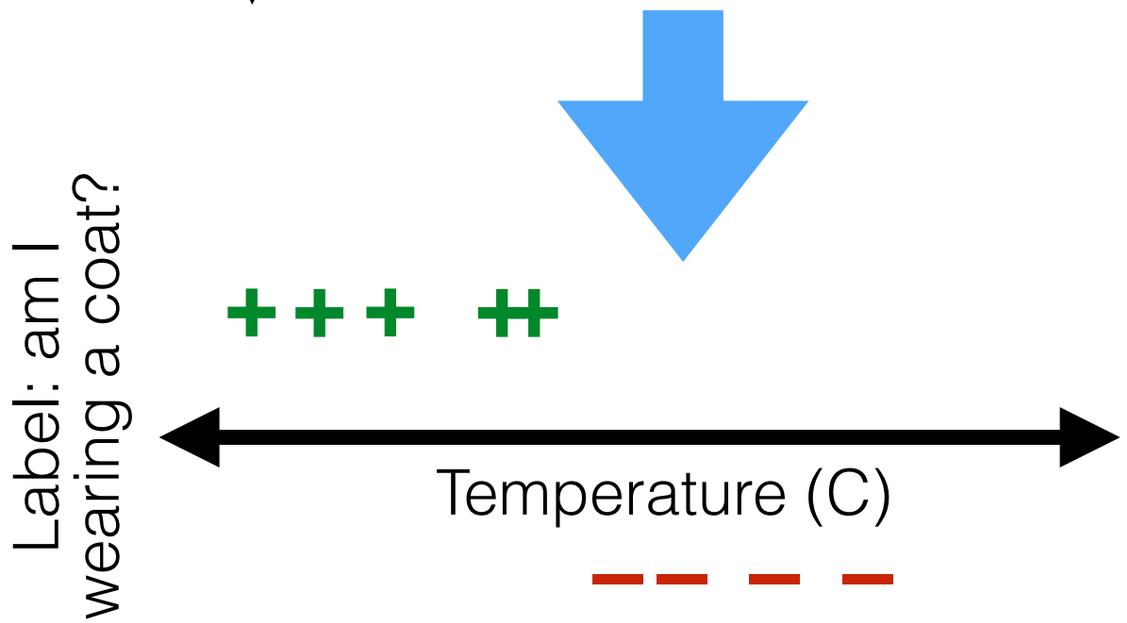


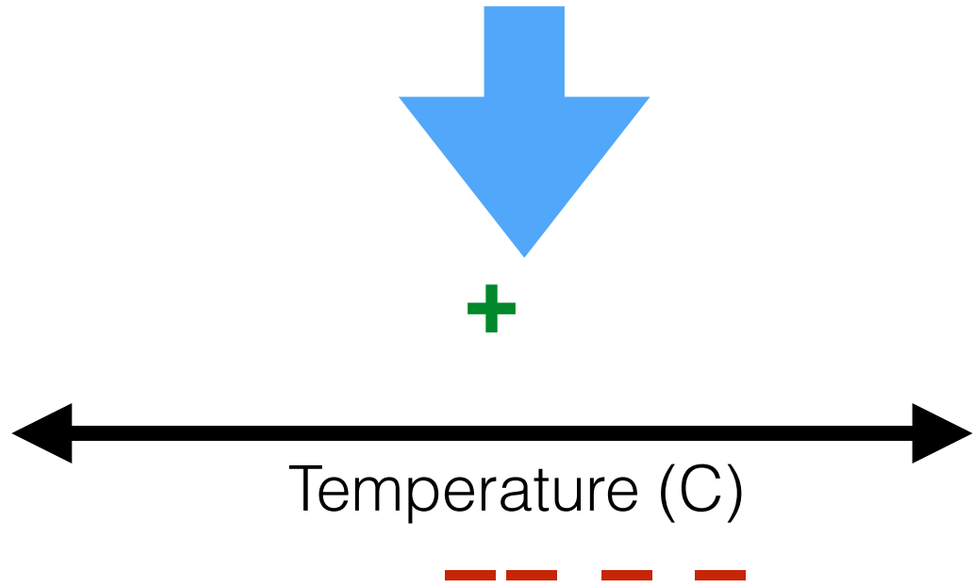


Capturing uncertainty

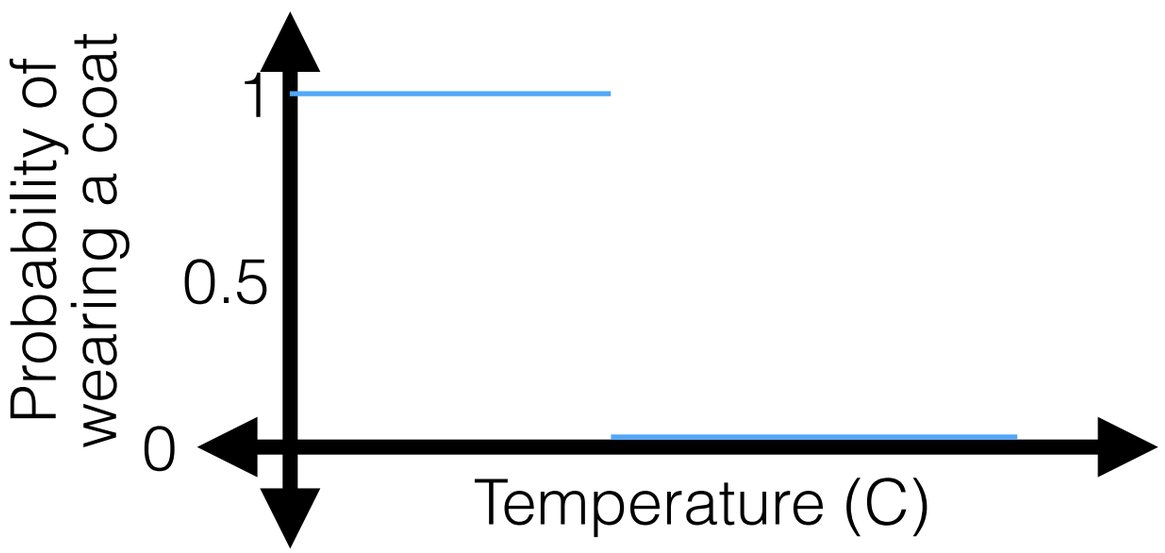


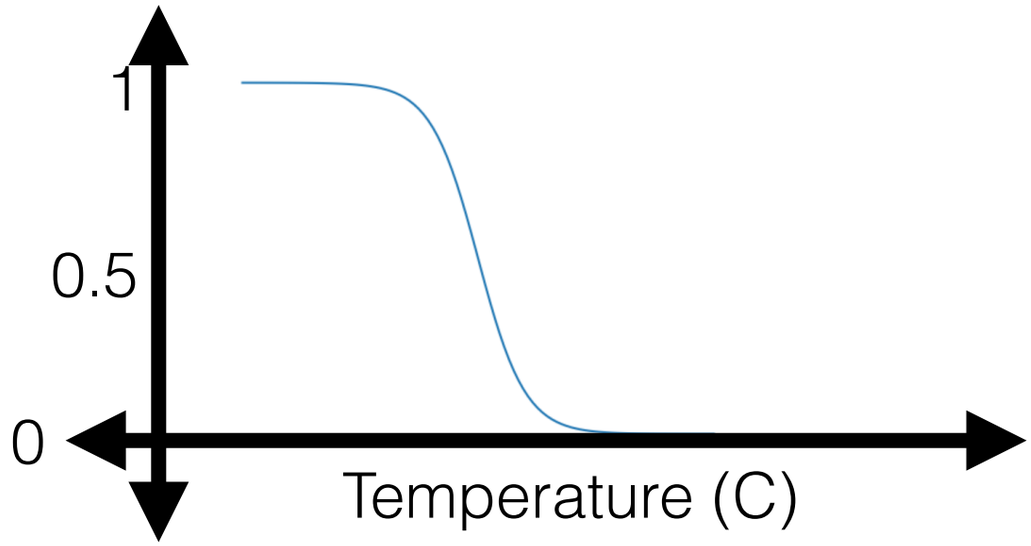


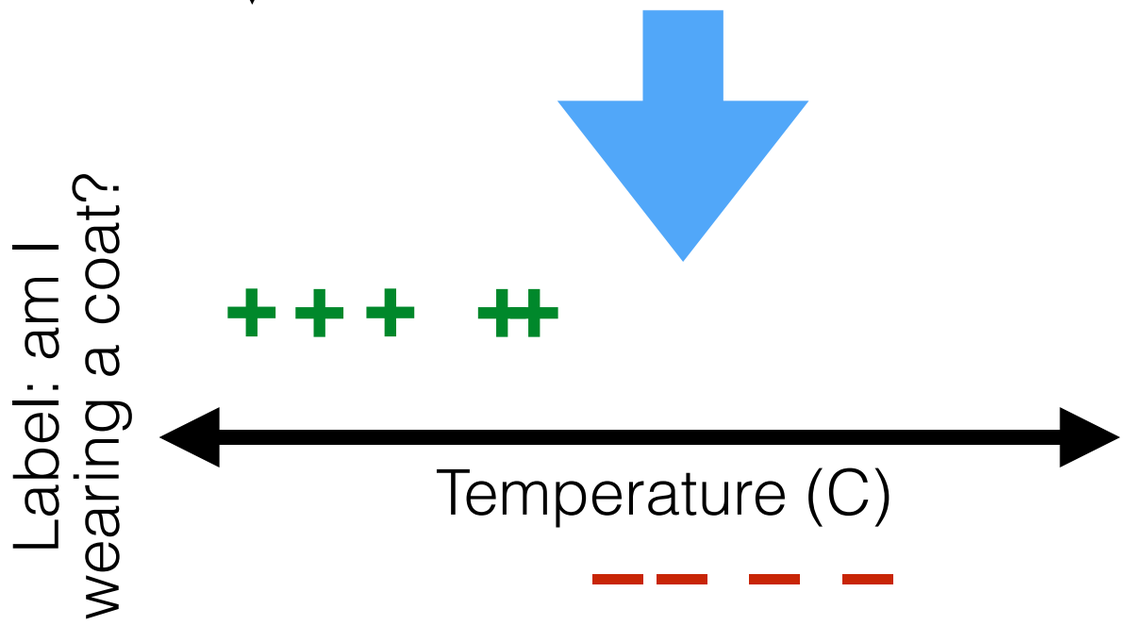


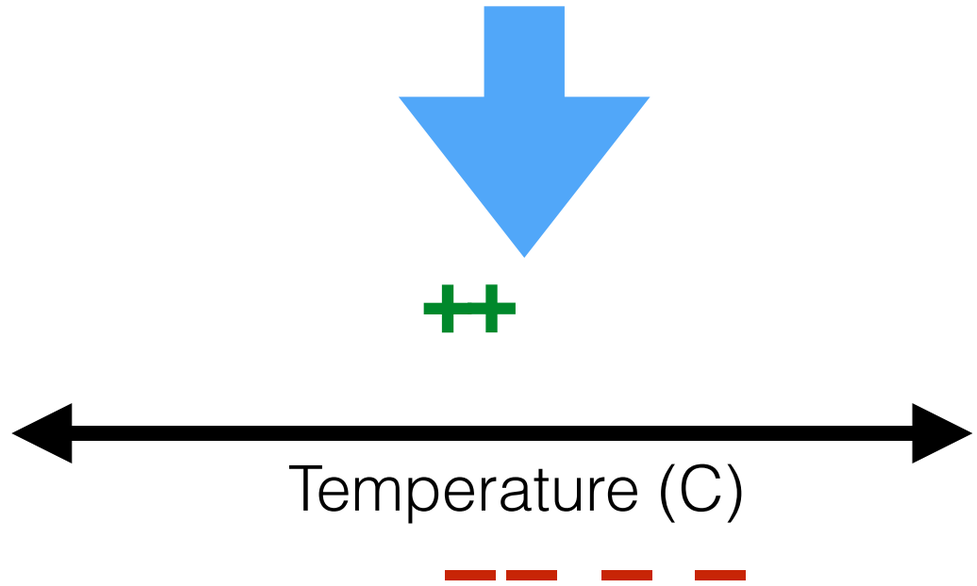


Capturing uncertainty

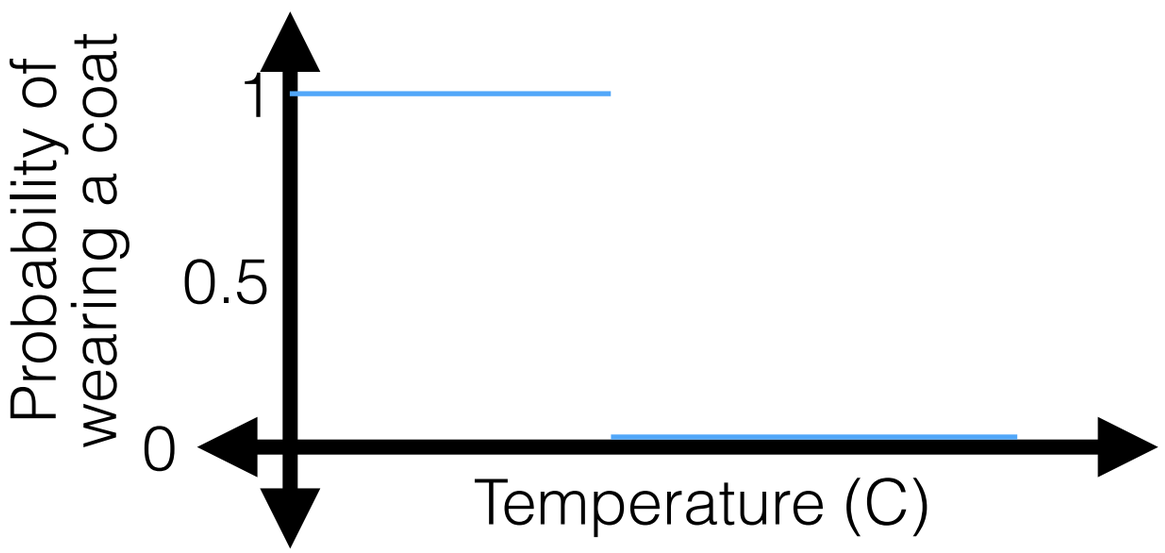


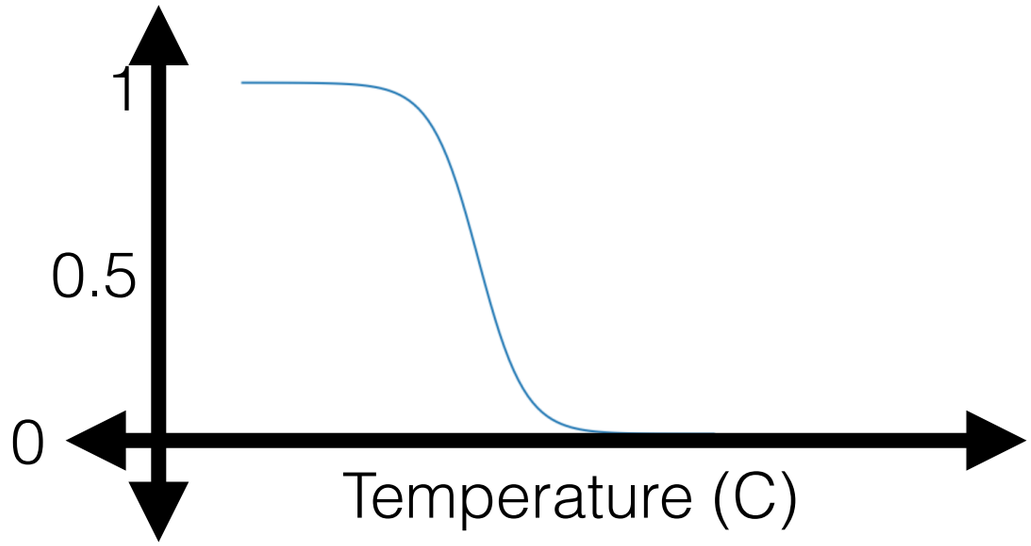


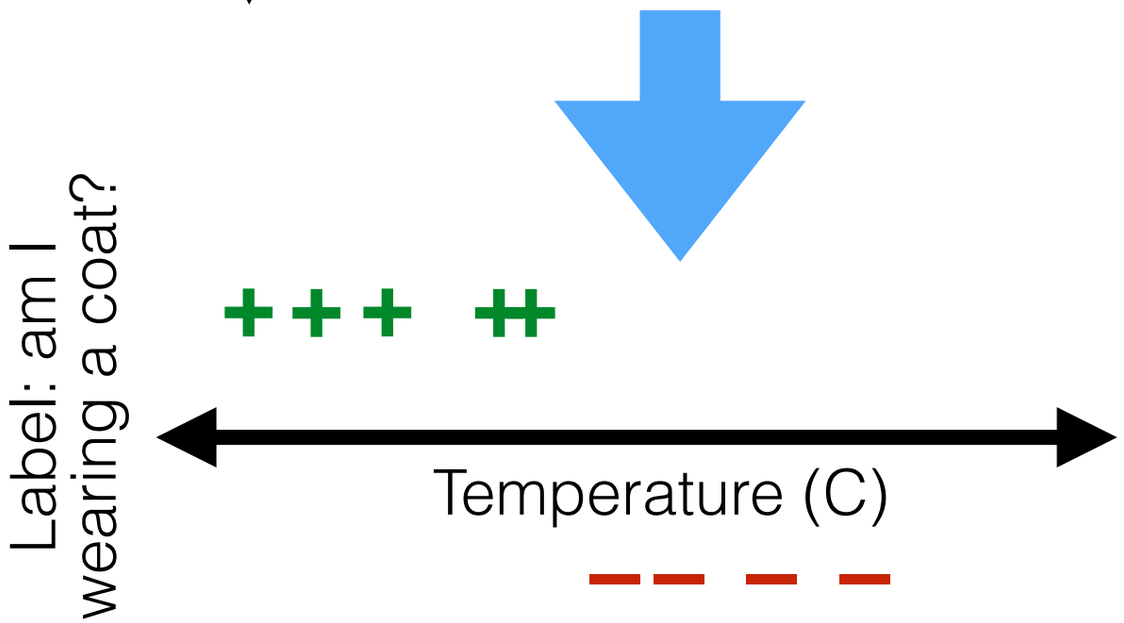


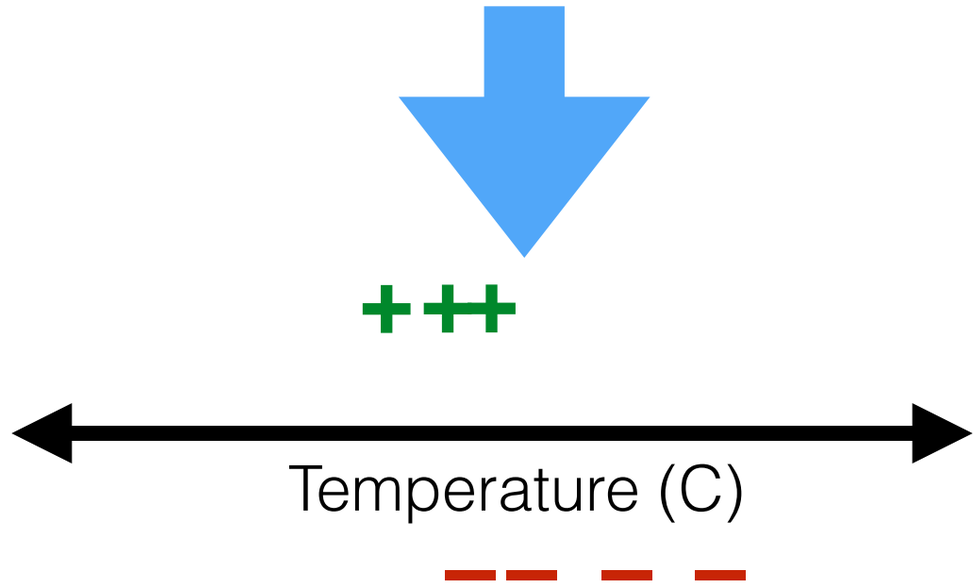


Capturing uncertainty

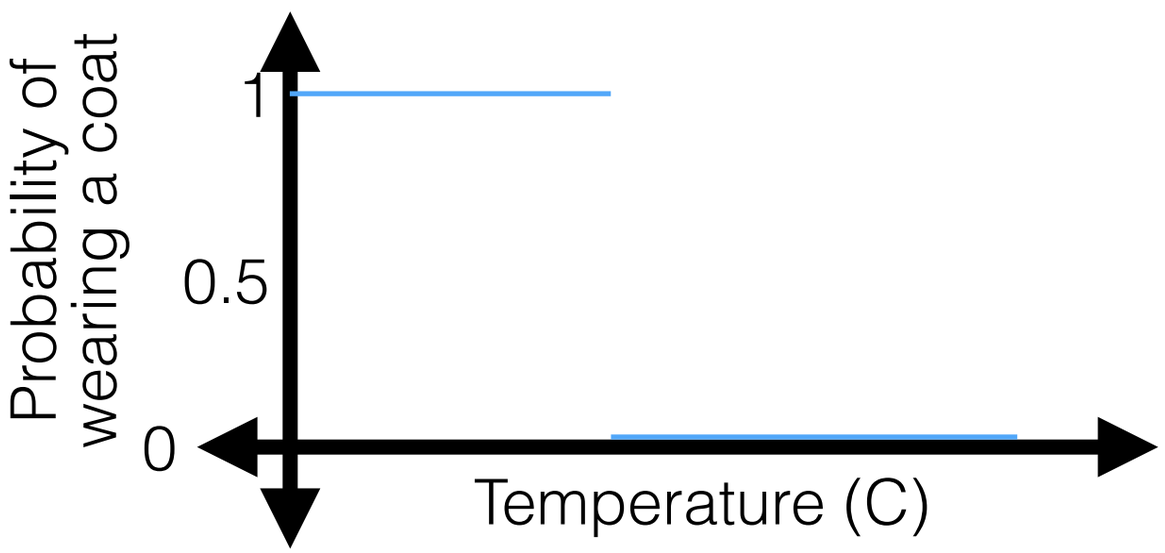


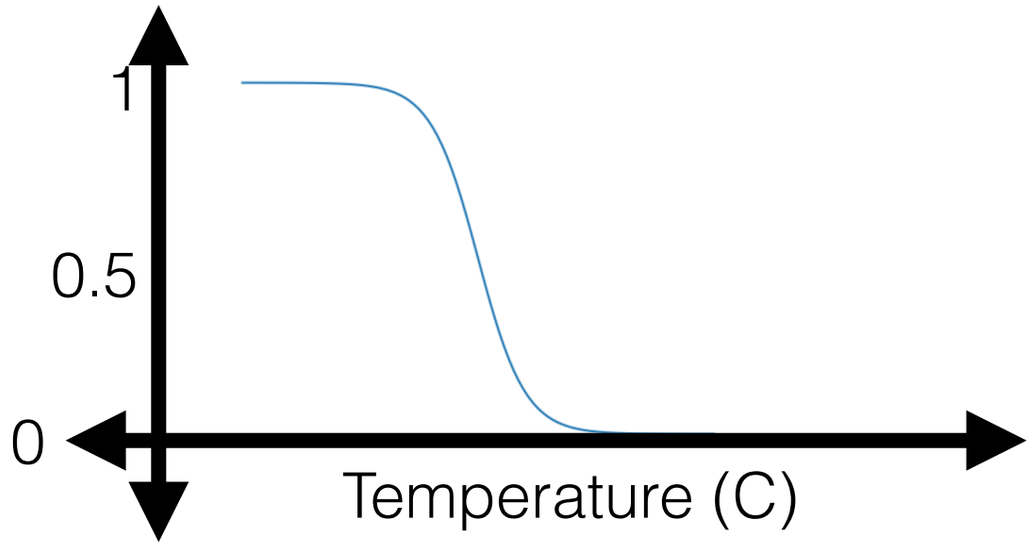


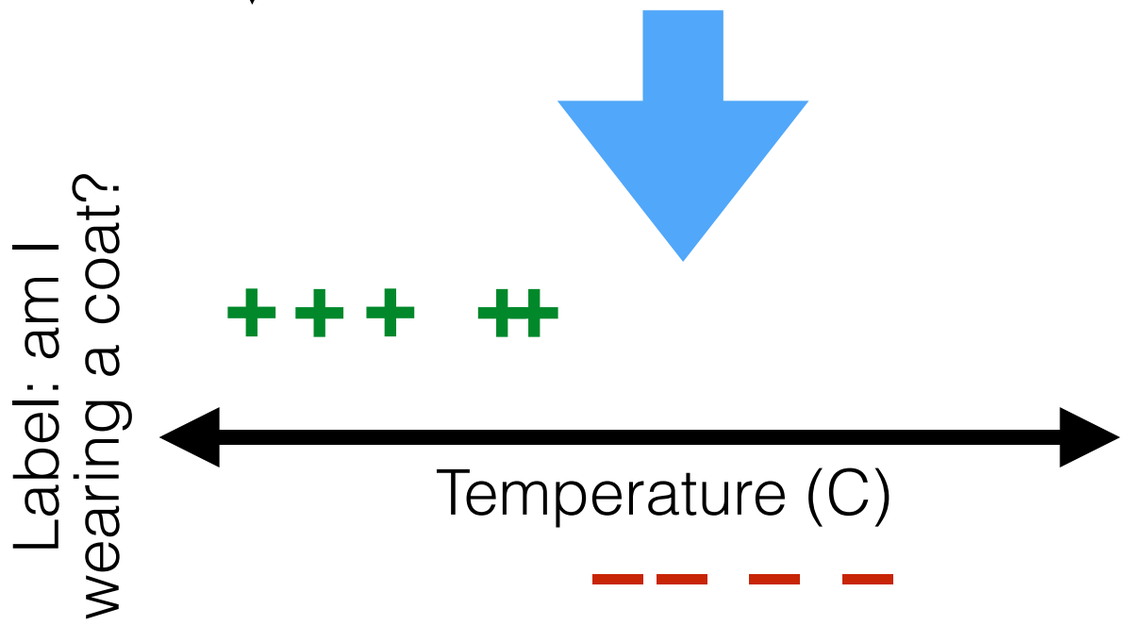


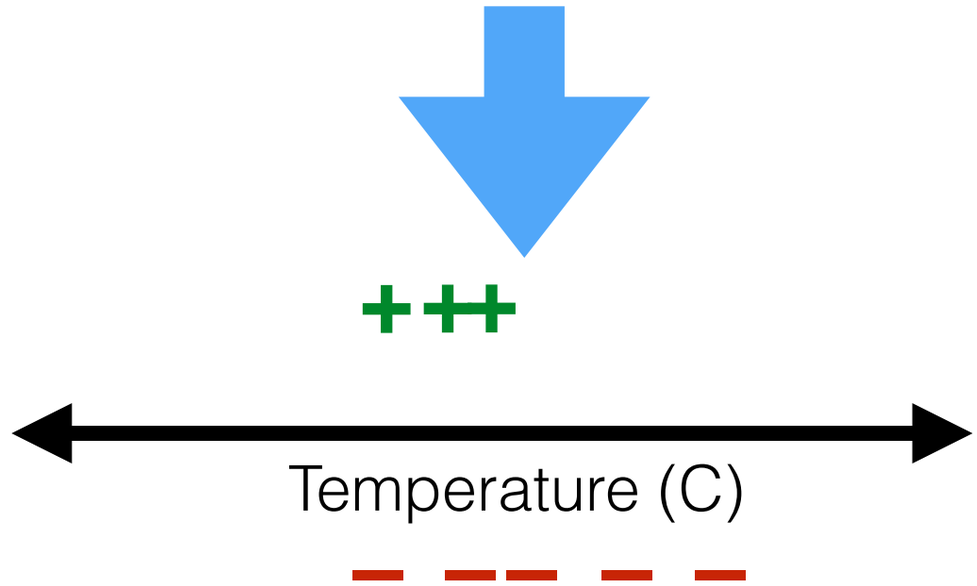


Capturing uncertainty

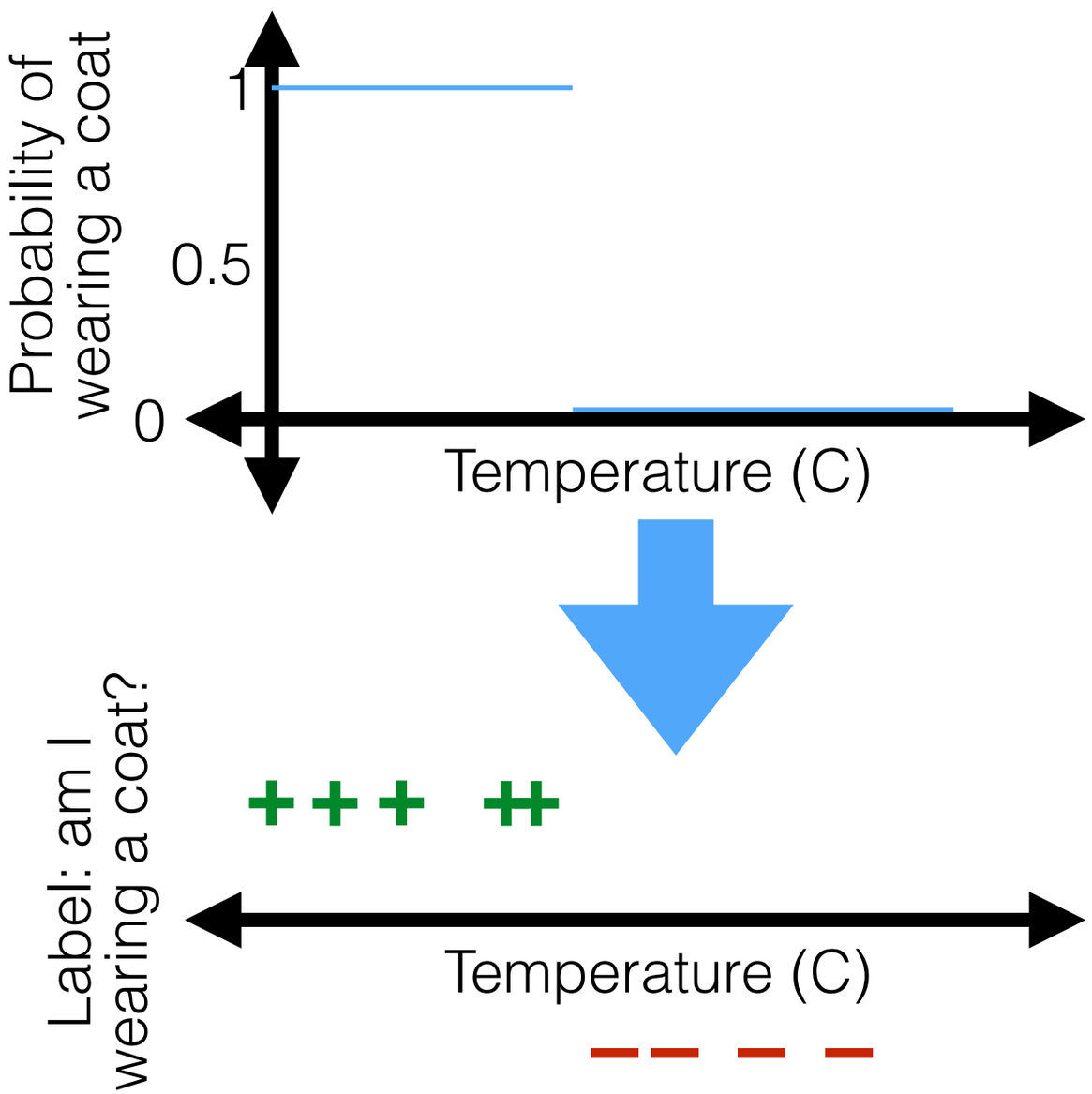


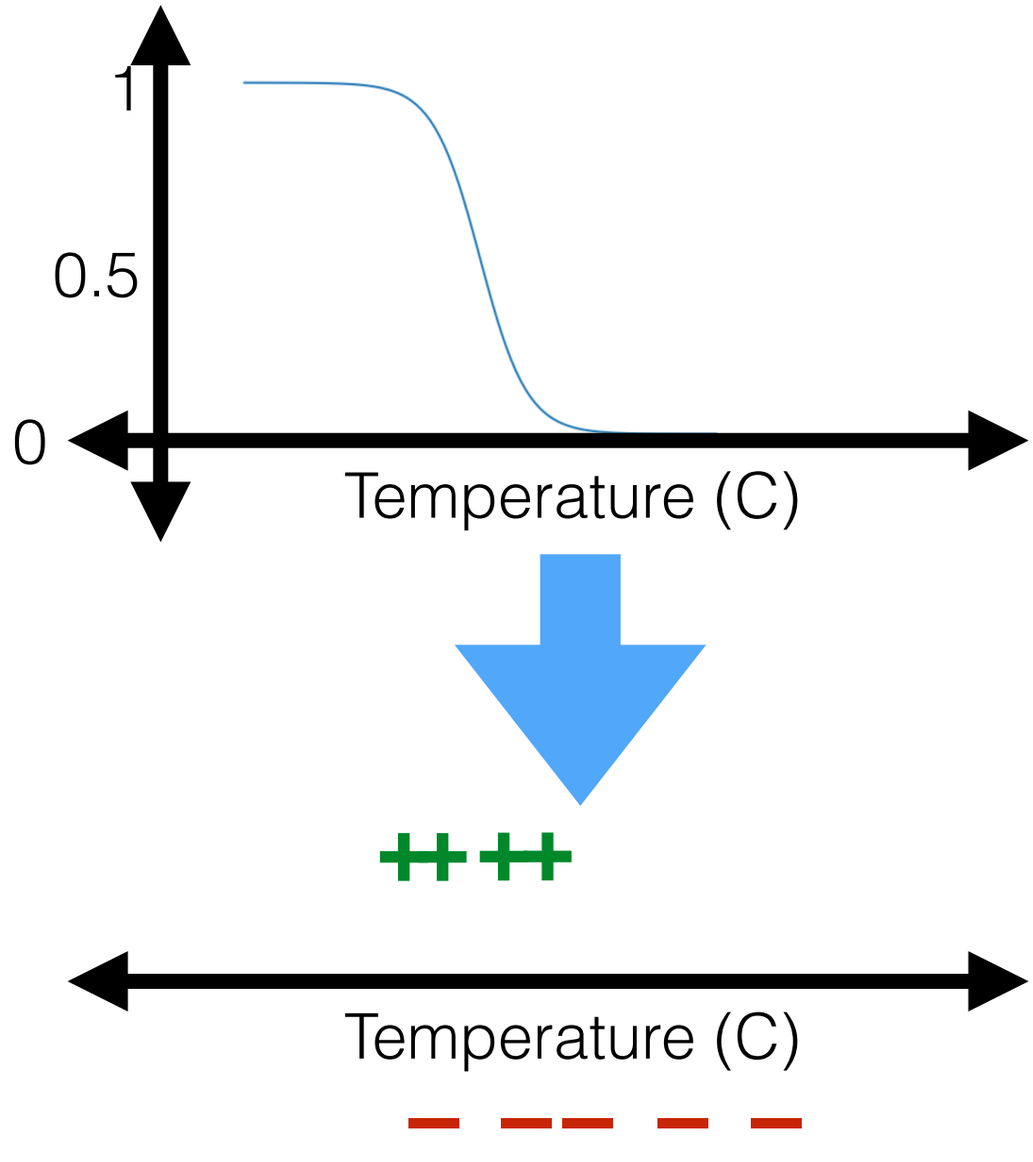




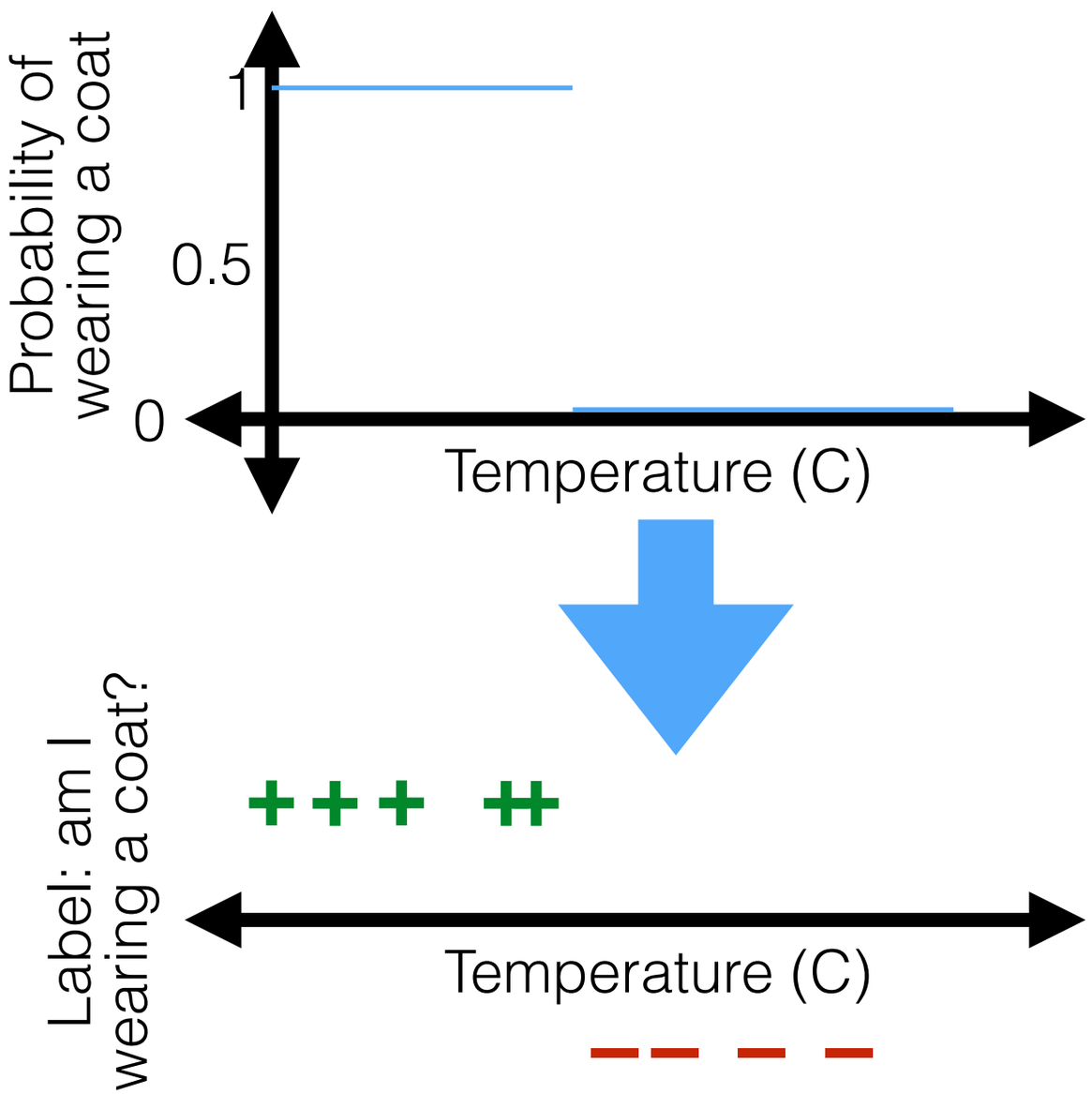


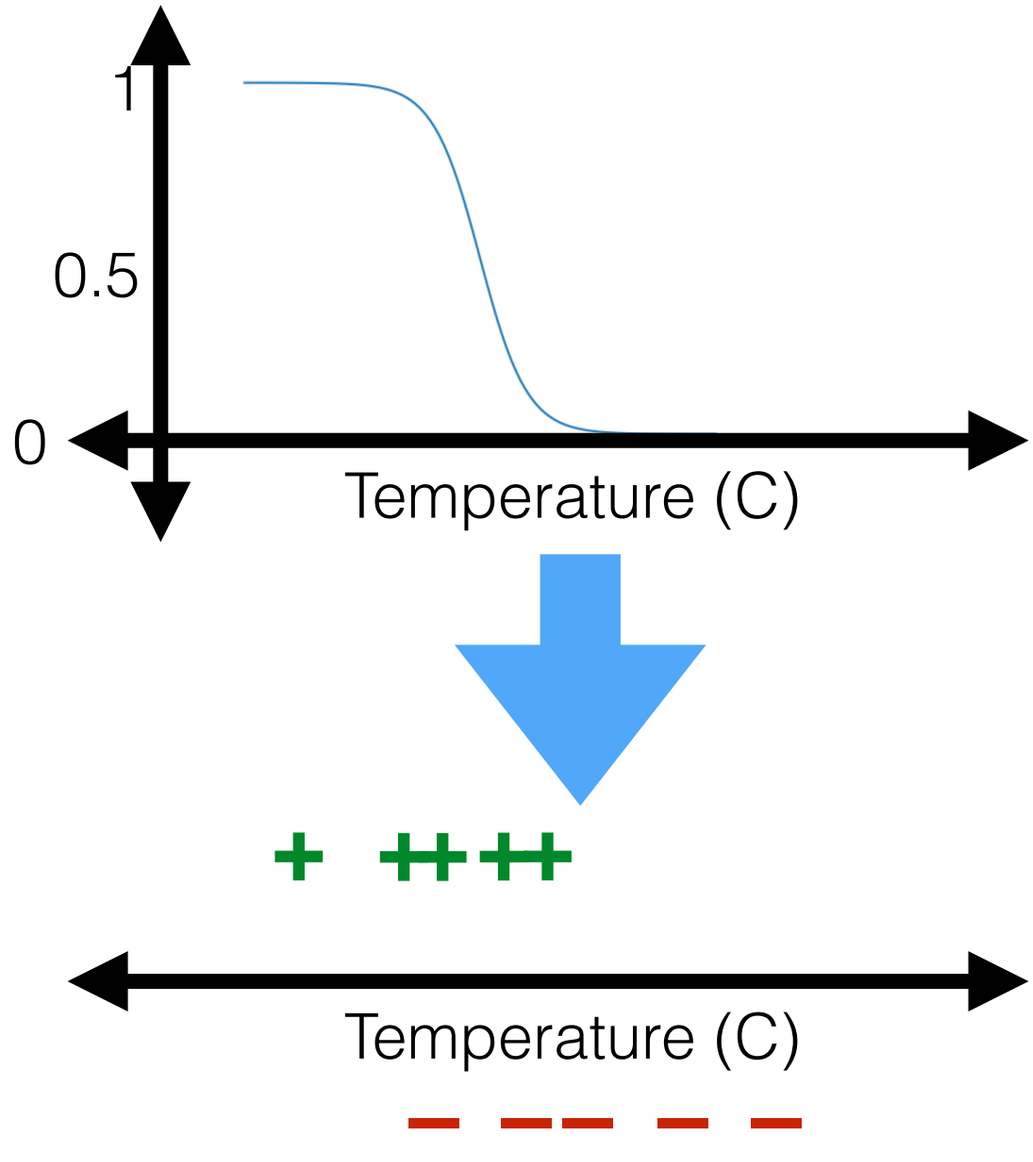
Capturing uncertainty



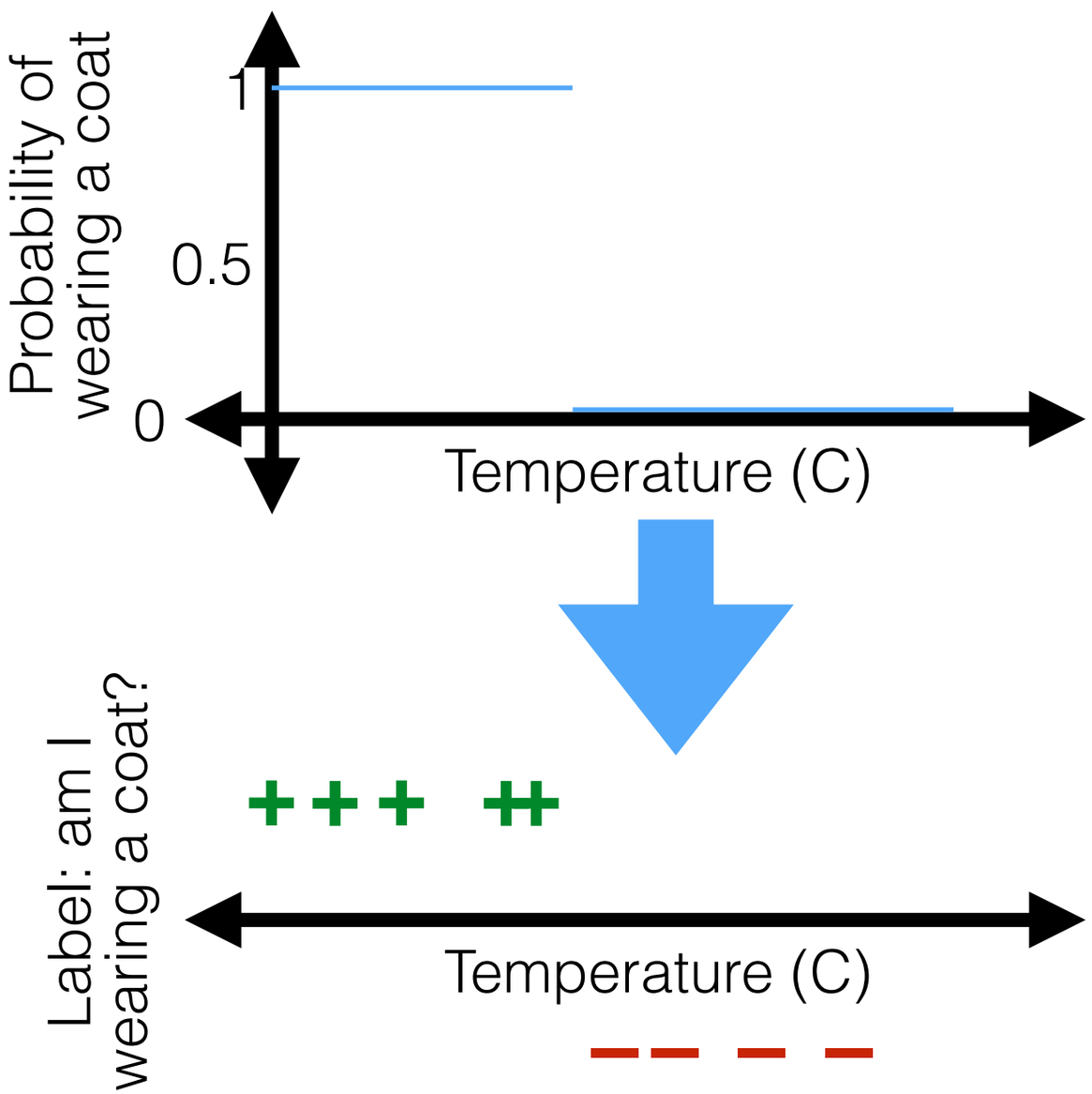


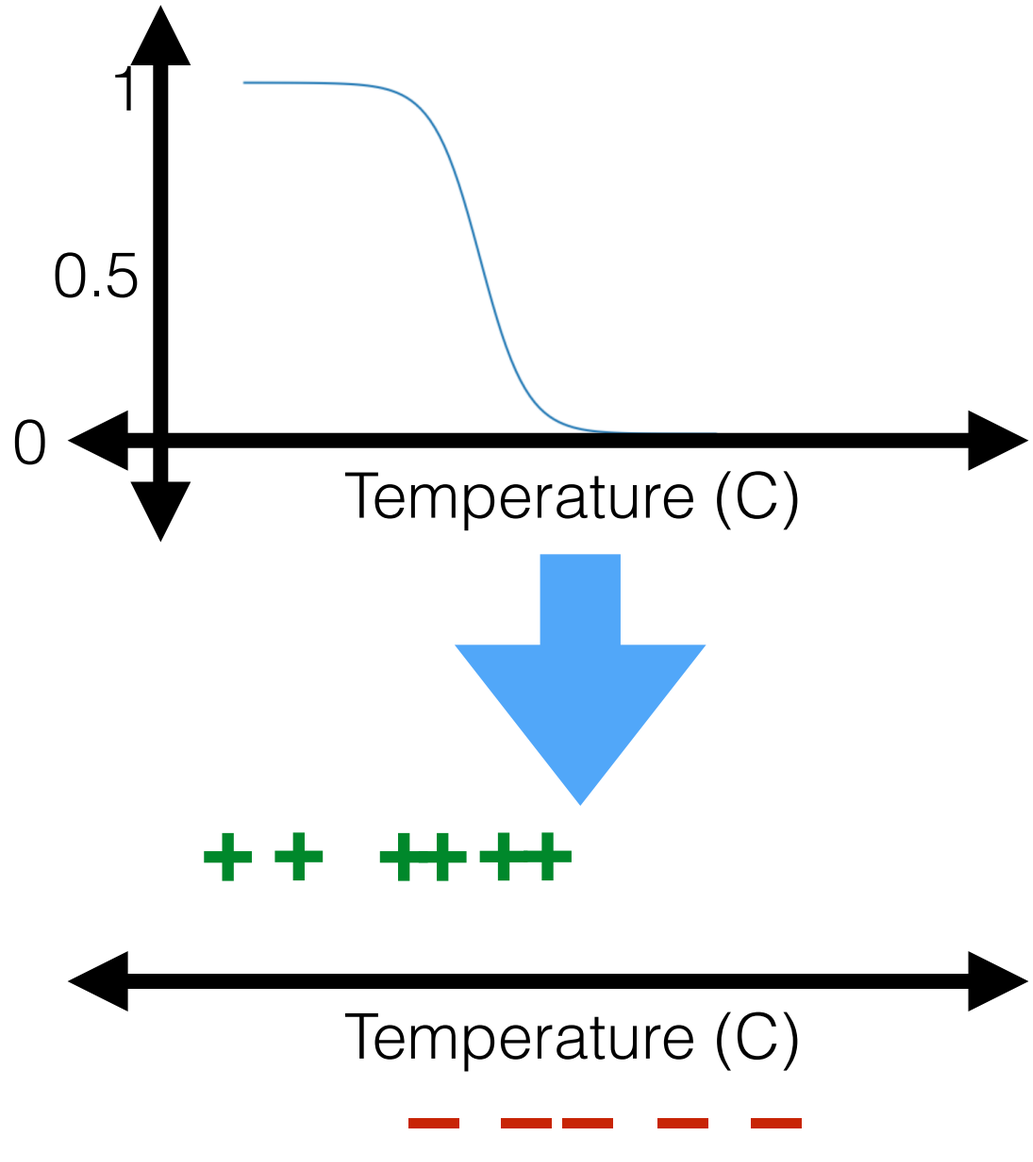
Capturing uncertainty



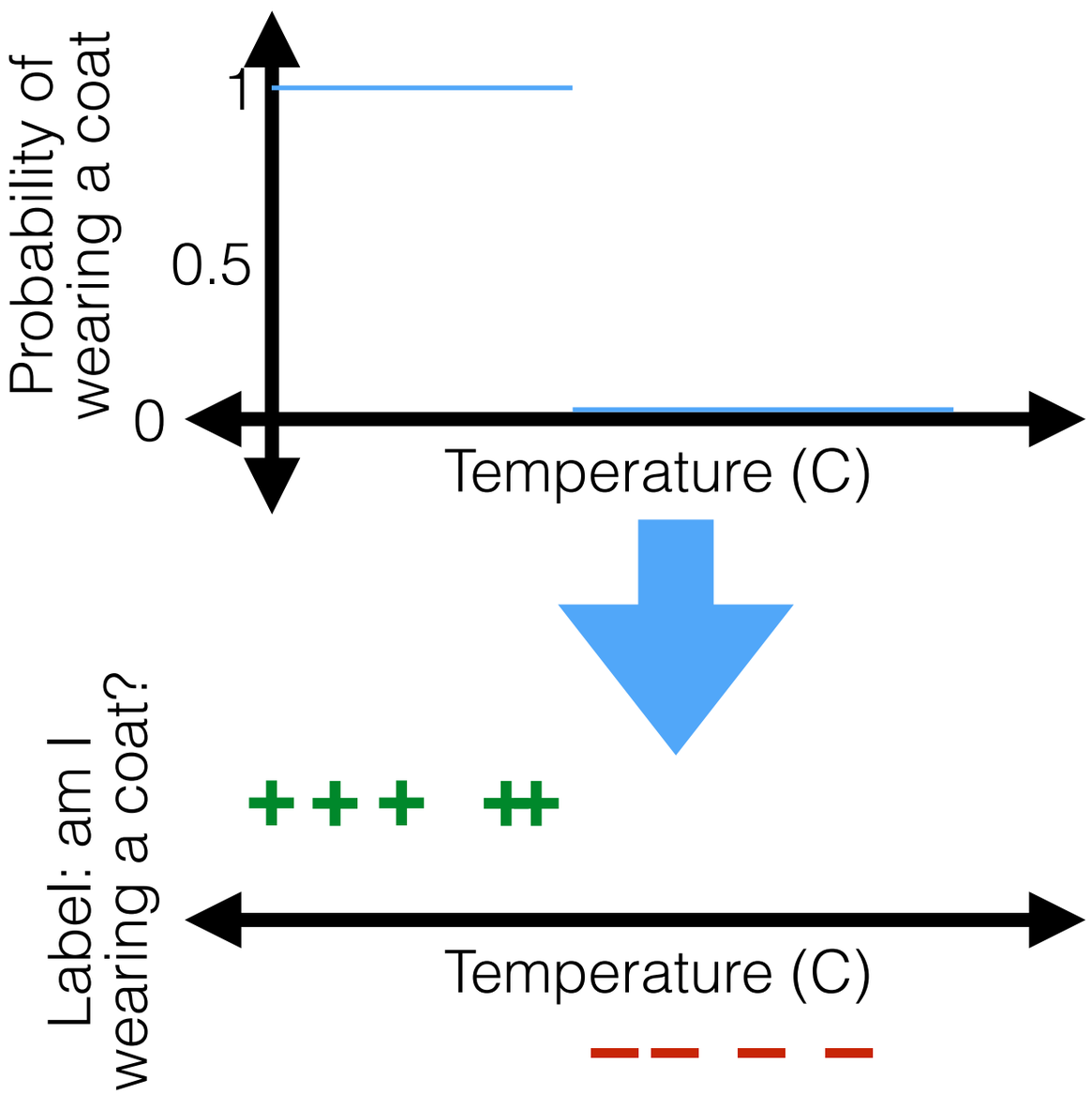


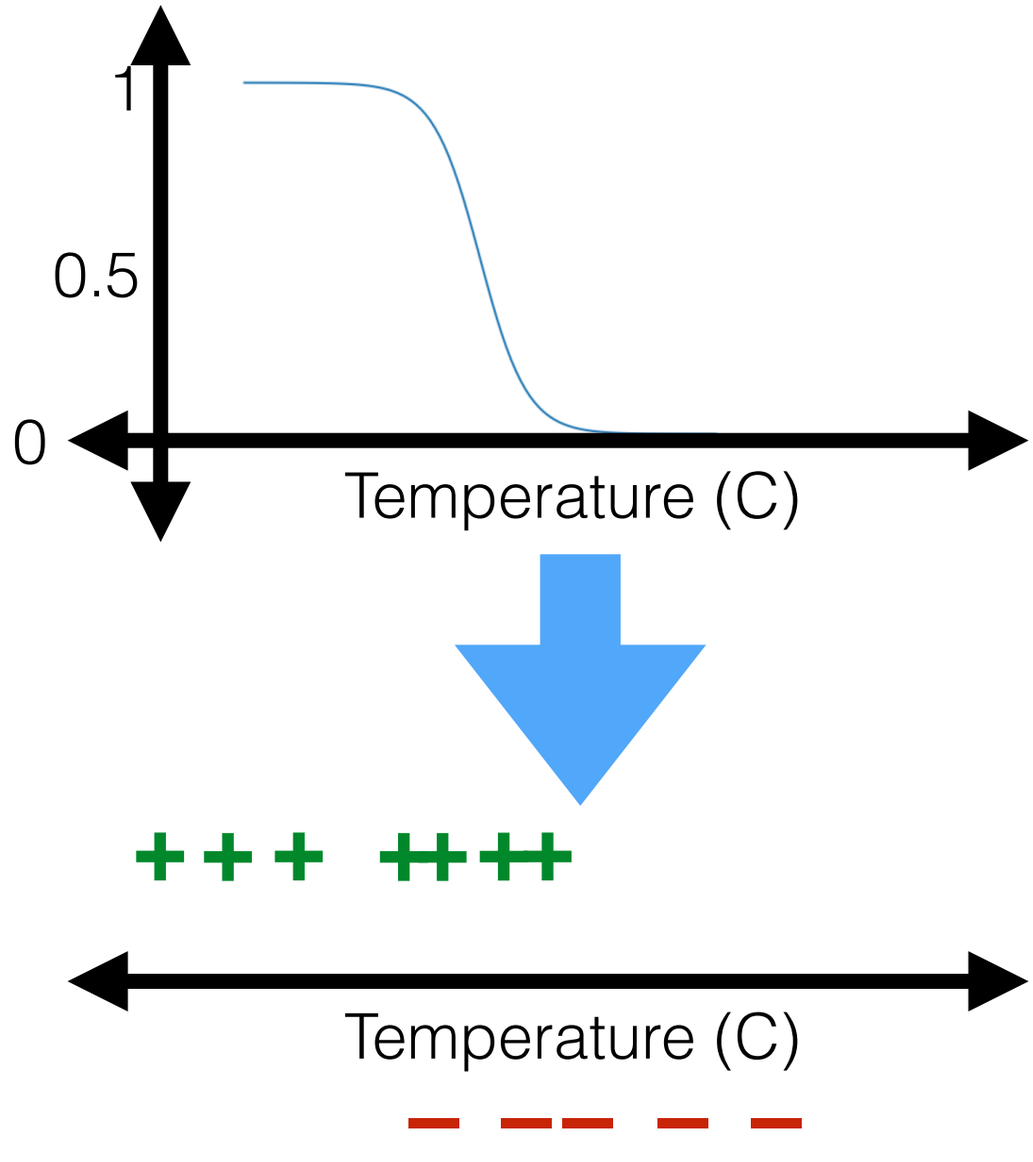
Capturing uncertainty



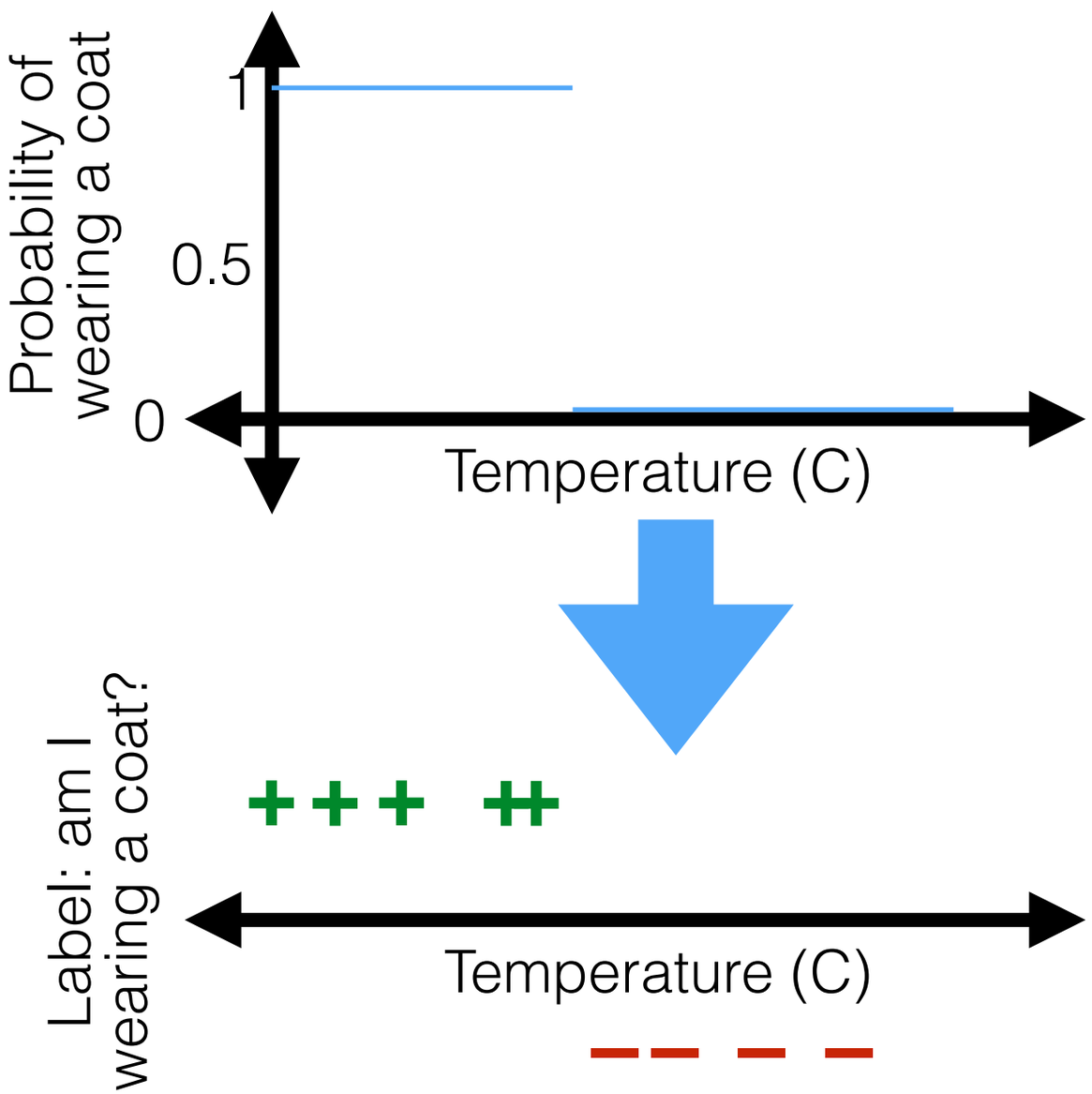


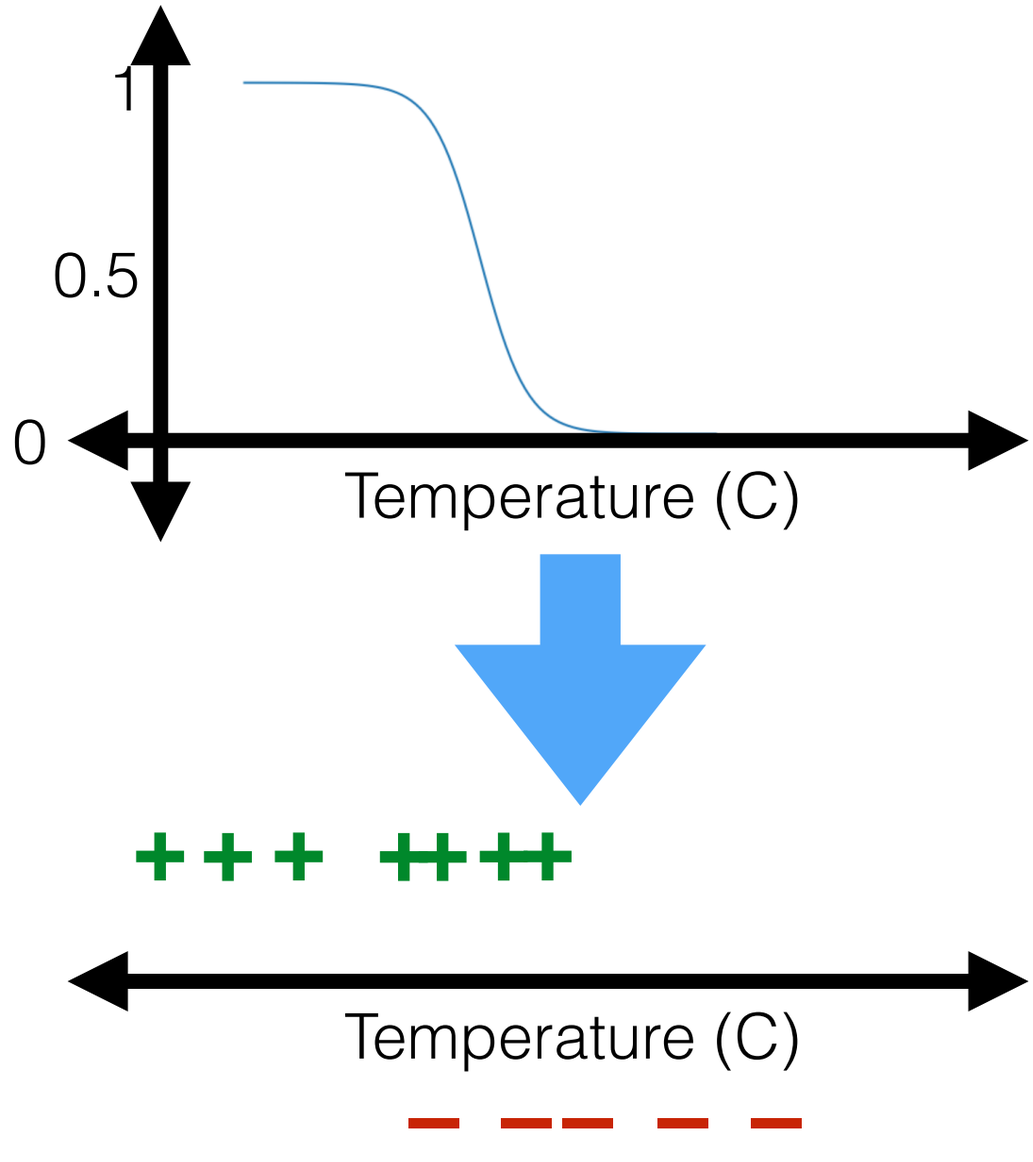
Capturing uncertainty





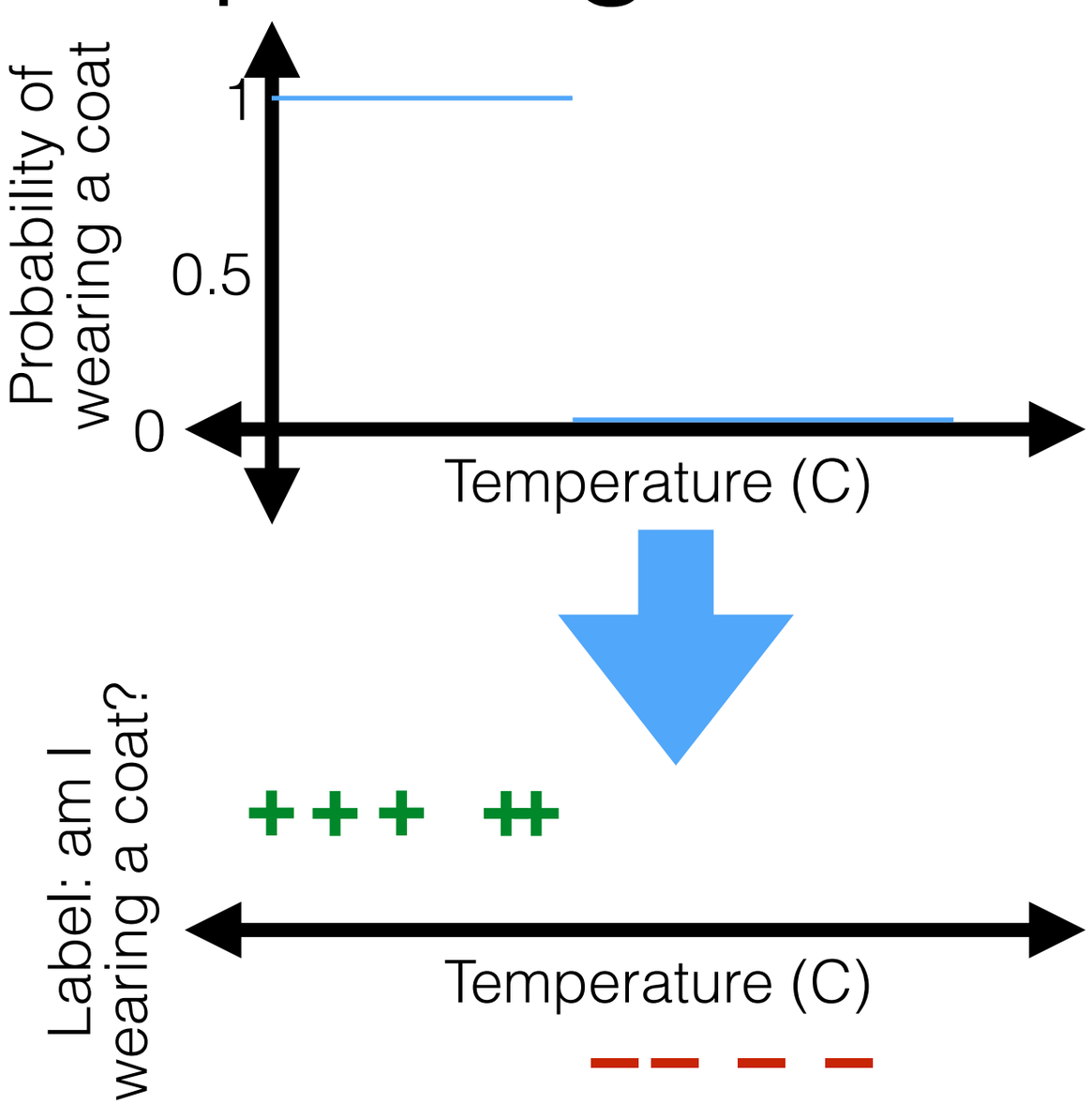
Capturing uncertainty

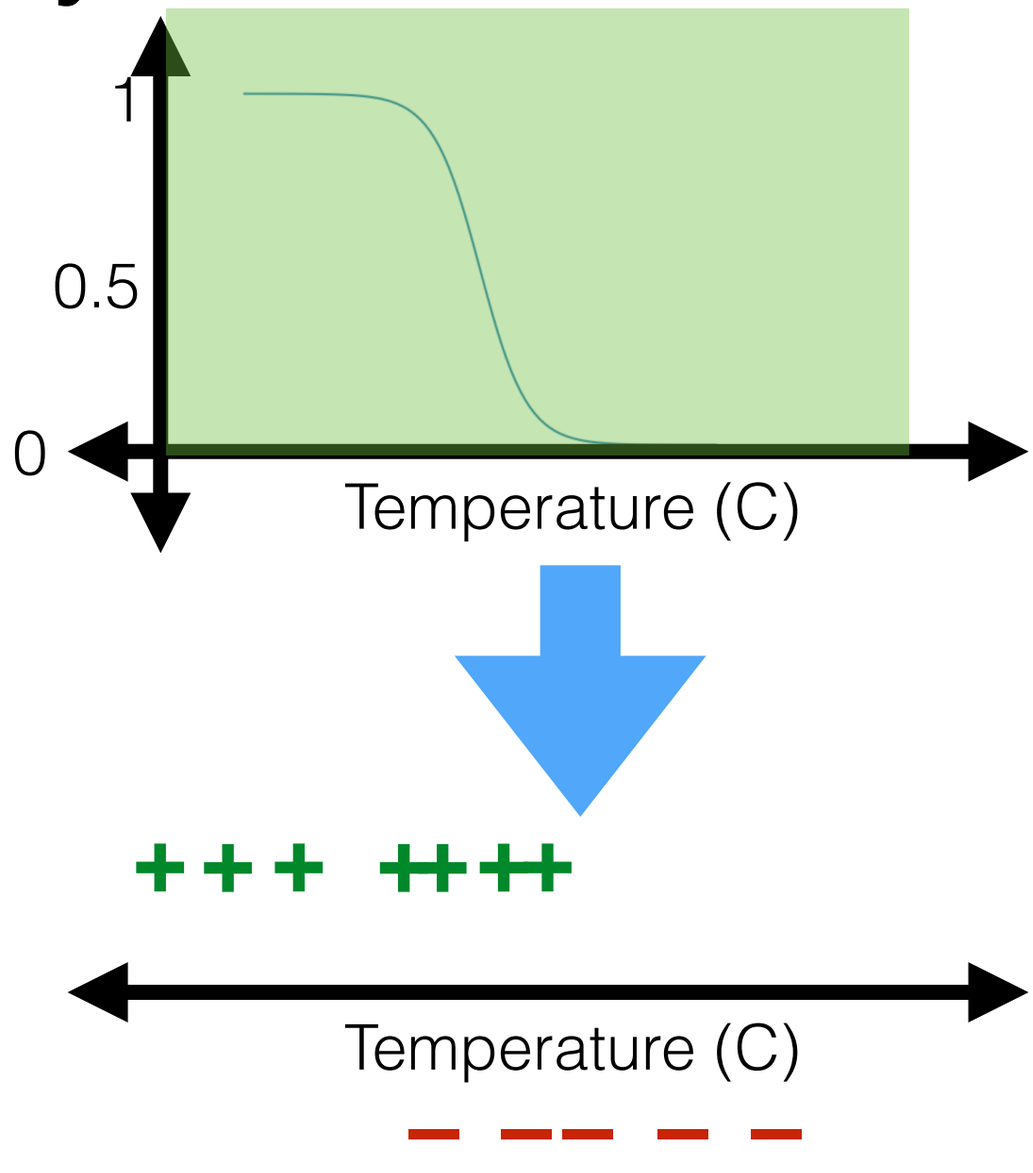




- How to make this shape?

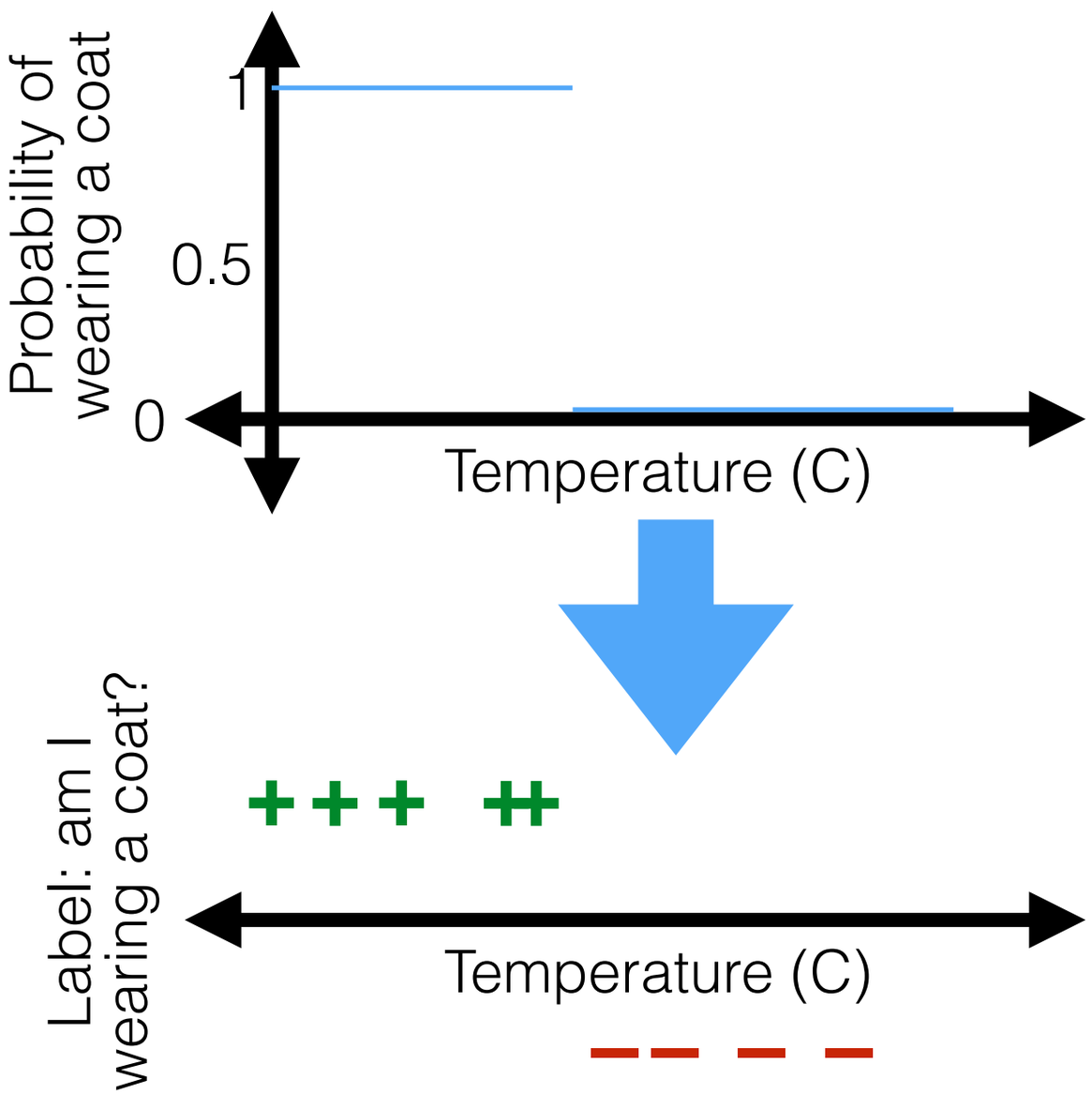
Capturing uncertainty

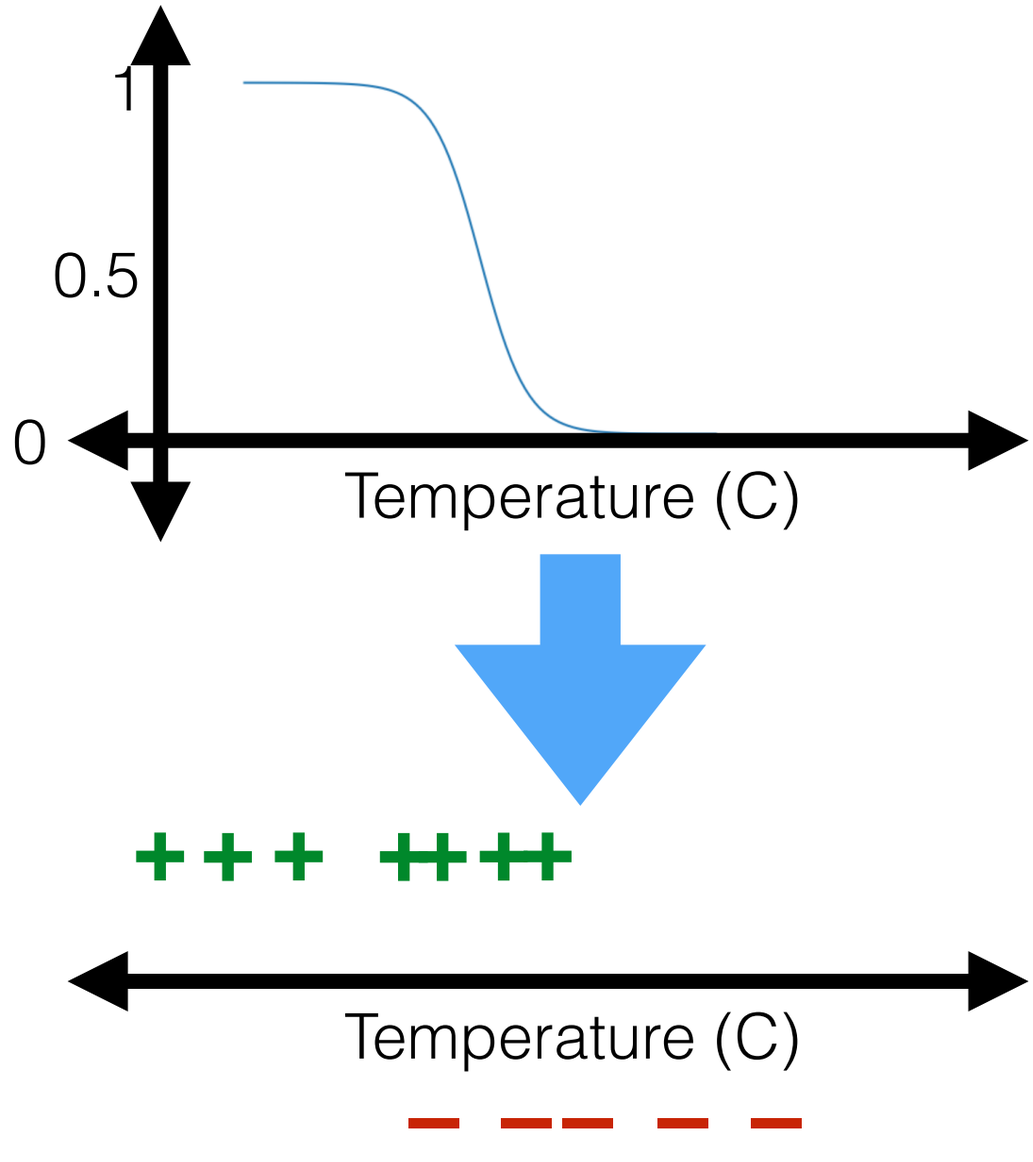




- How to make this shape?

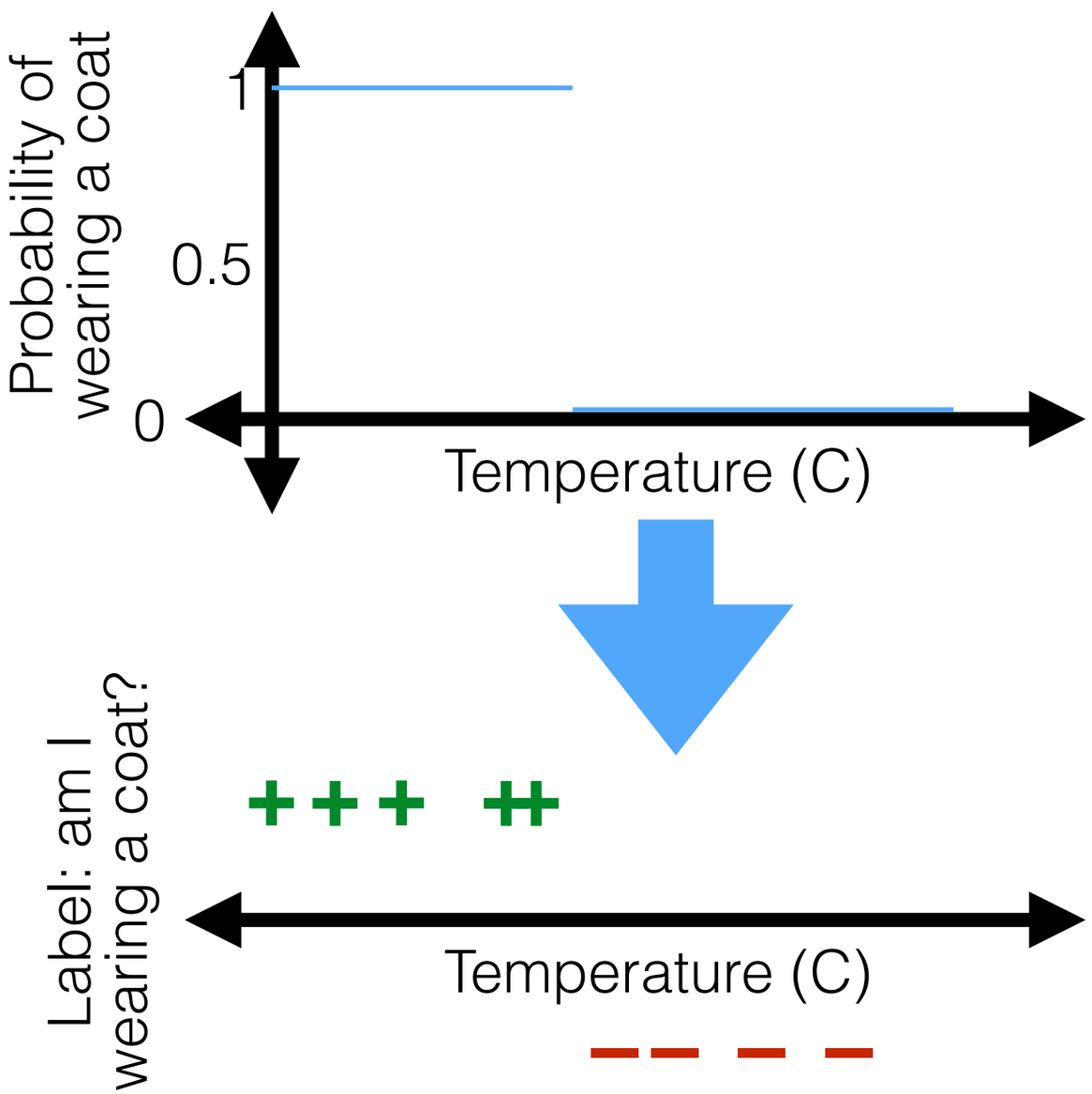
Capturing uncertainty

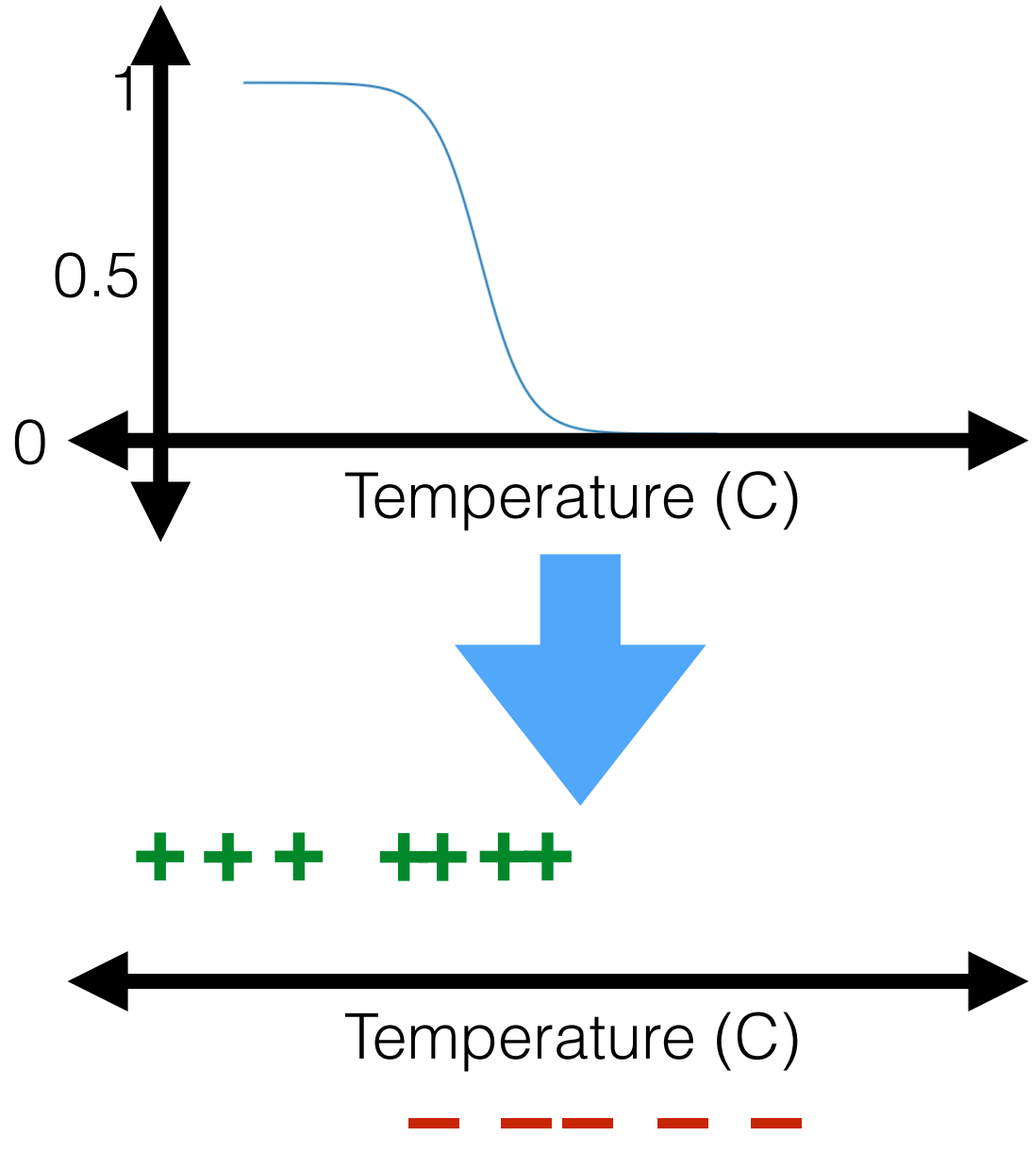




- How to make this shape?

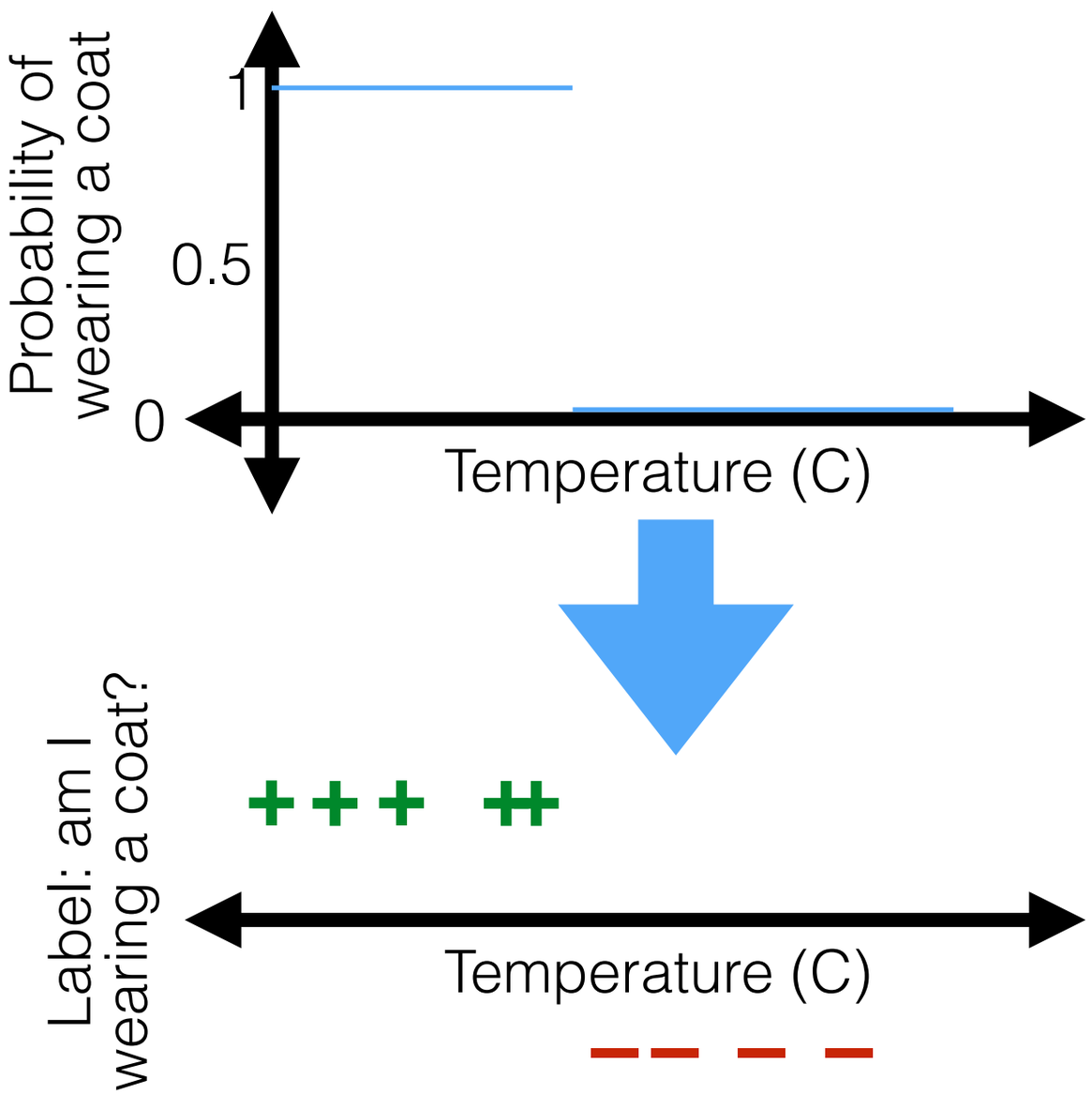
Capturing uncertainty

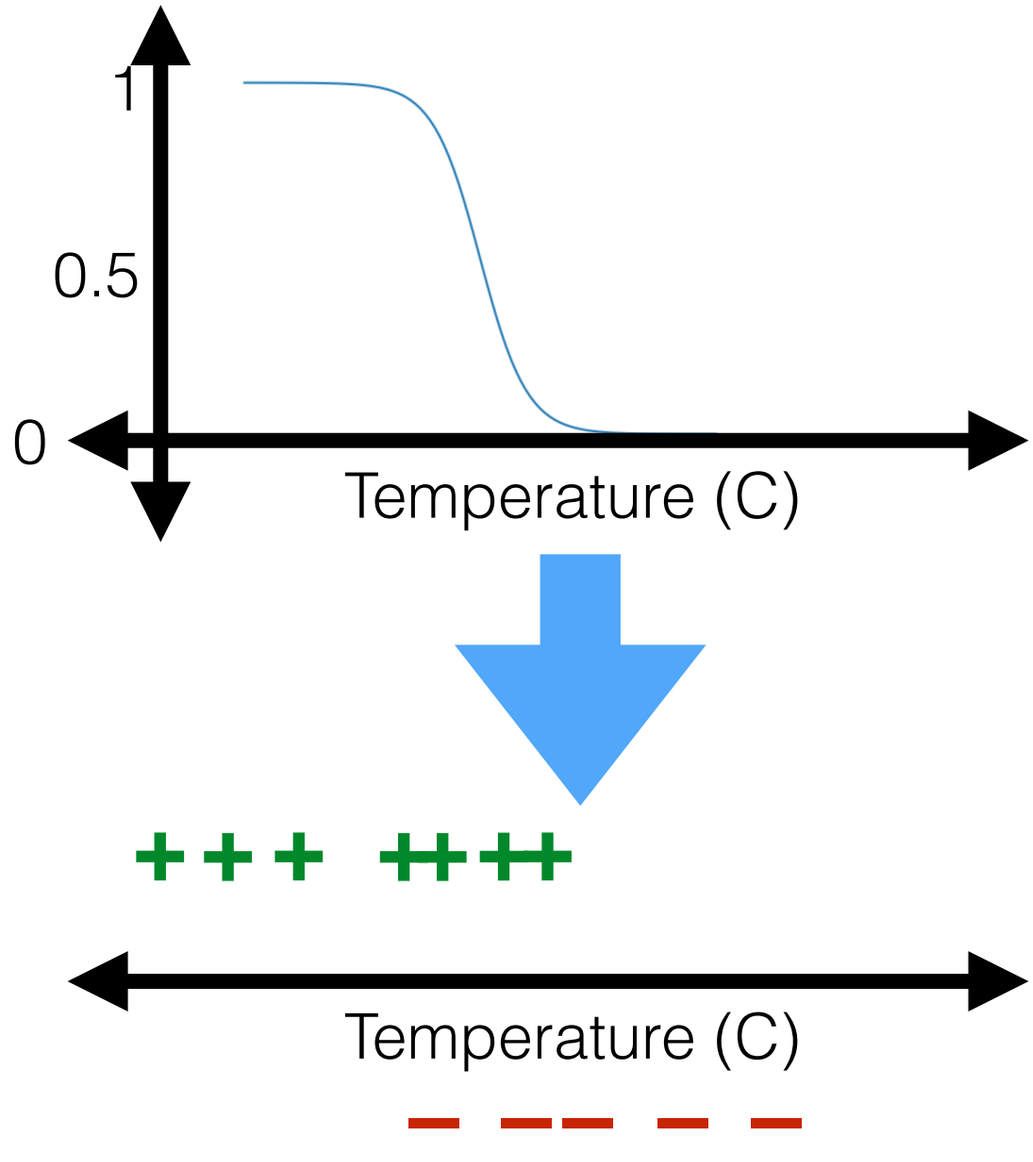




- How to make this shape?
  - Sigmoid/logistic function

Capturing uncertainty

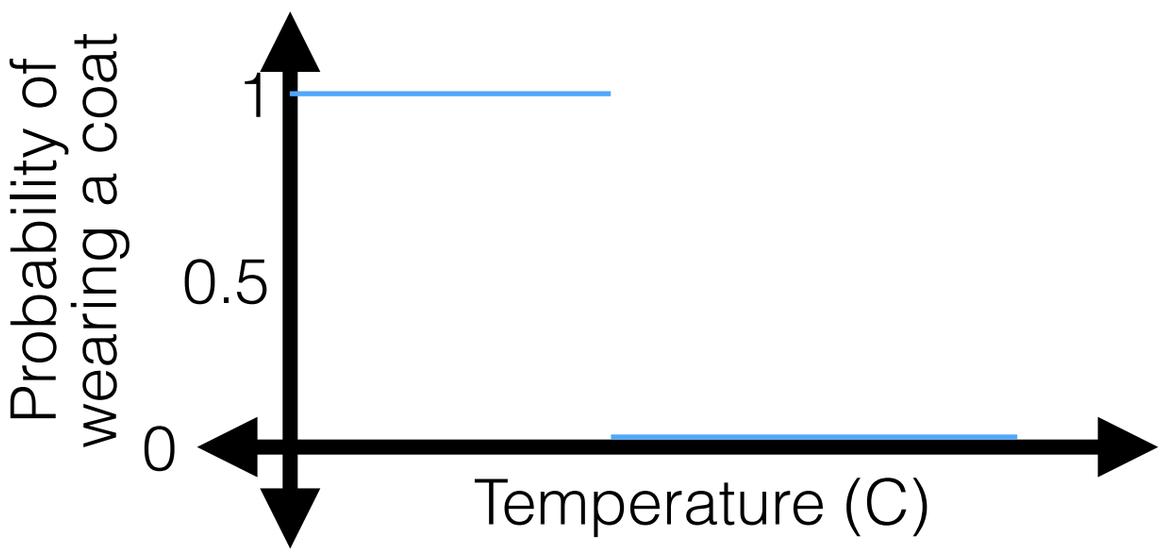


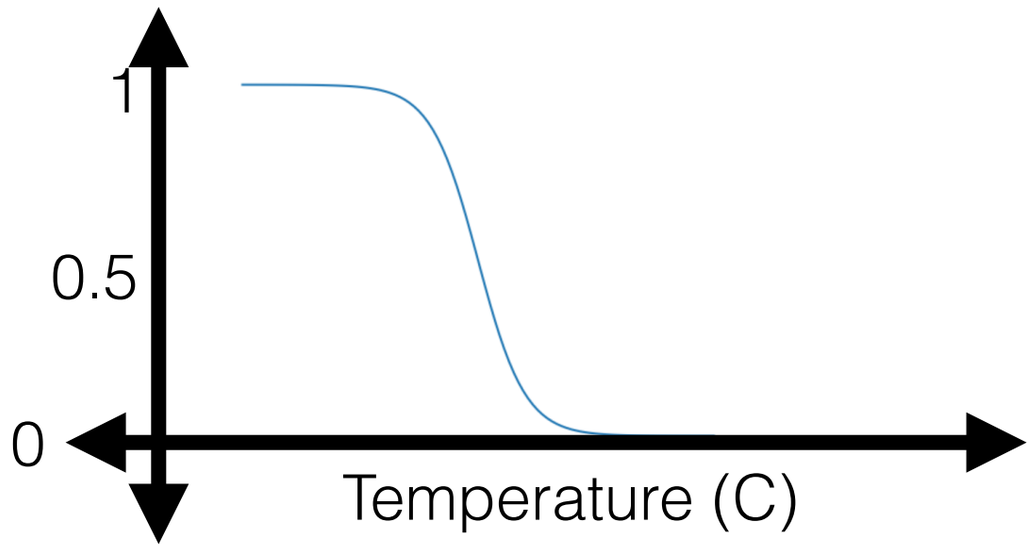


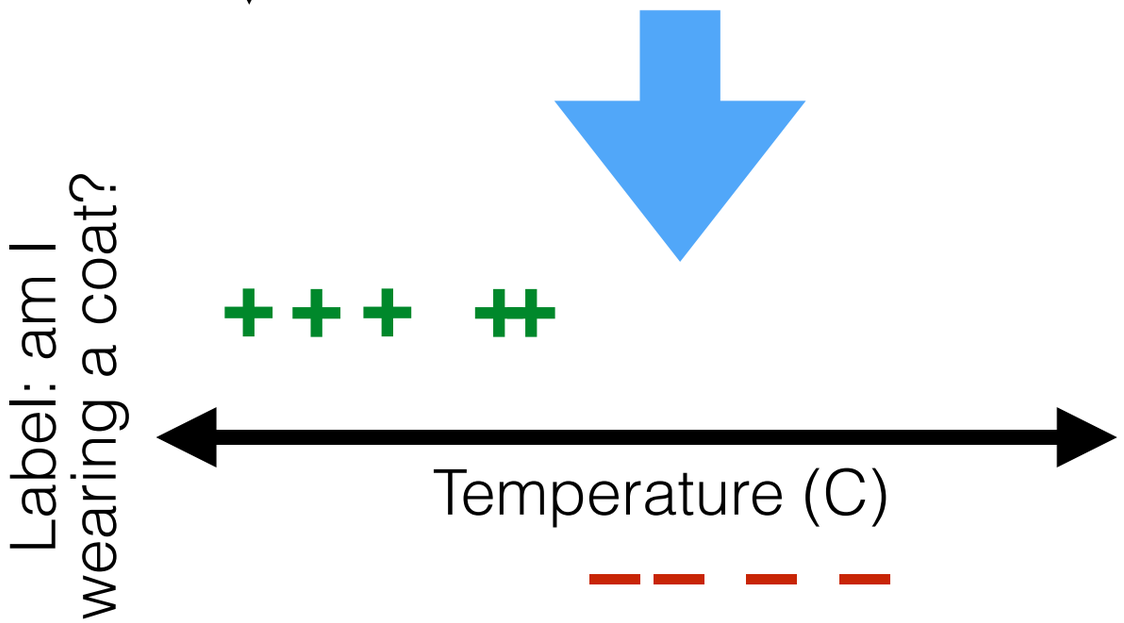
- How to make this shape?
  - Sigmoid/logistic function

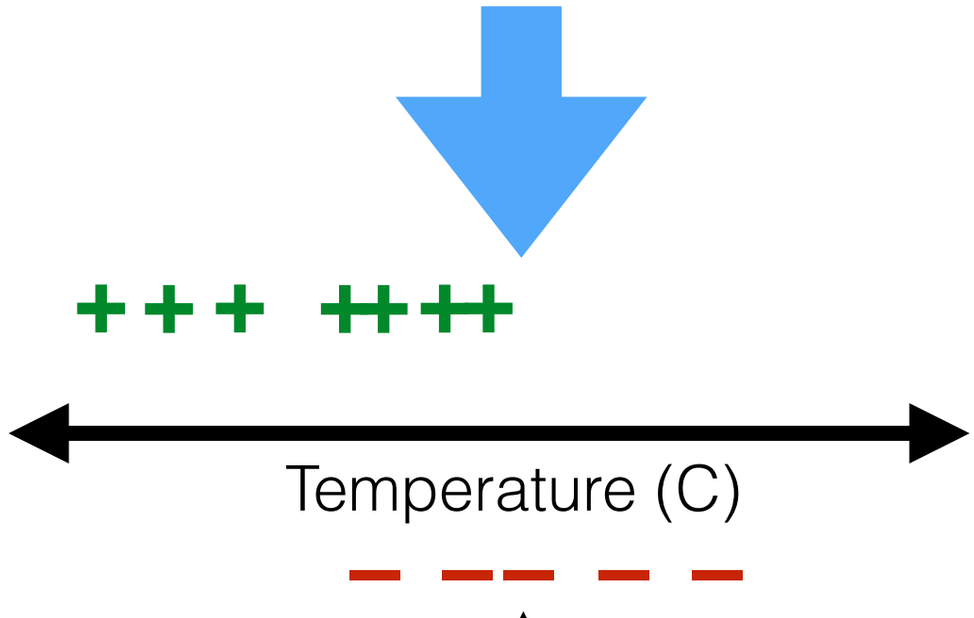
$$\sigma(z) = \frac{1}{1 + \exp(-z)}$$

Capturing uncertainty



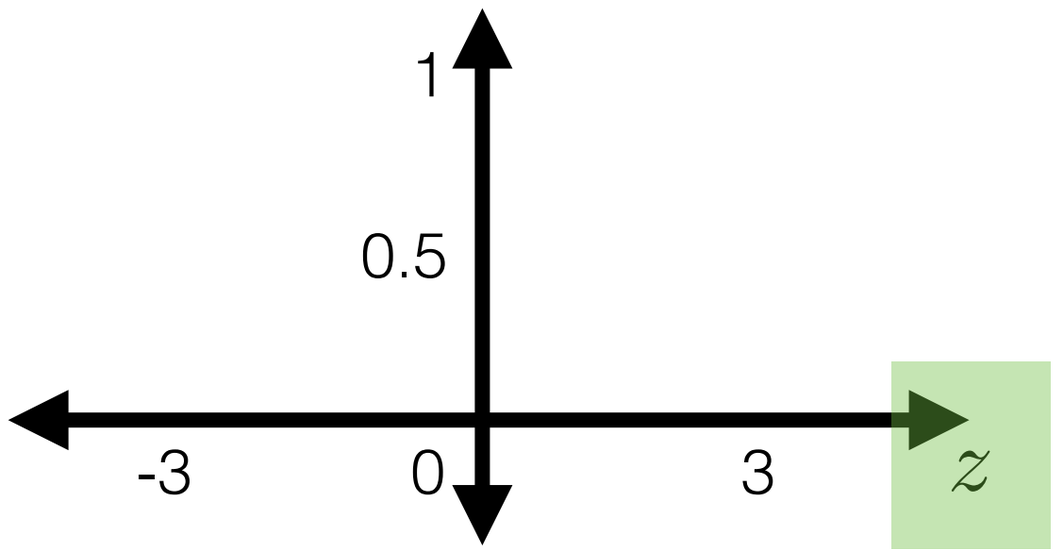




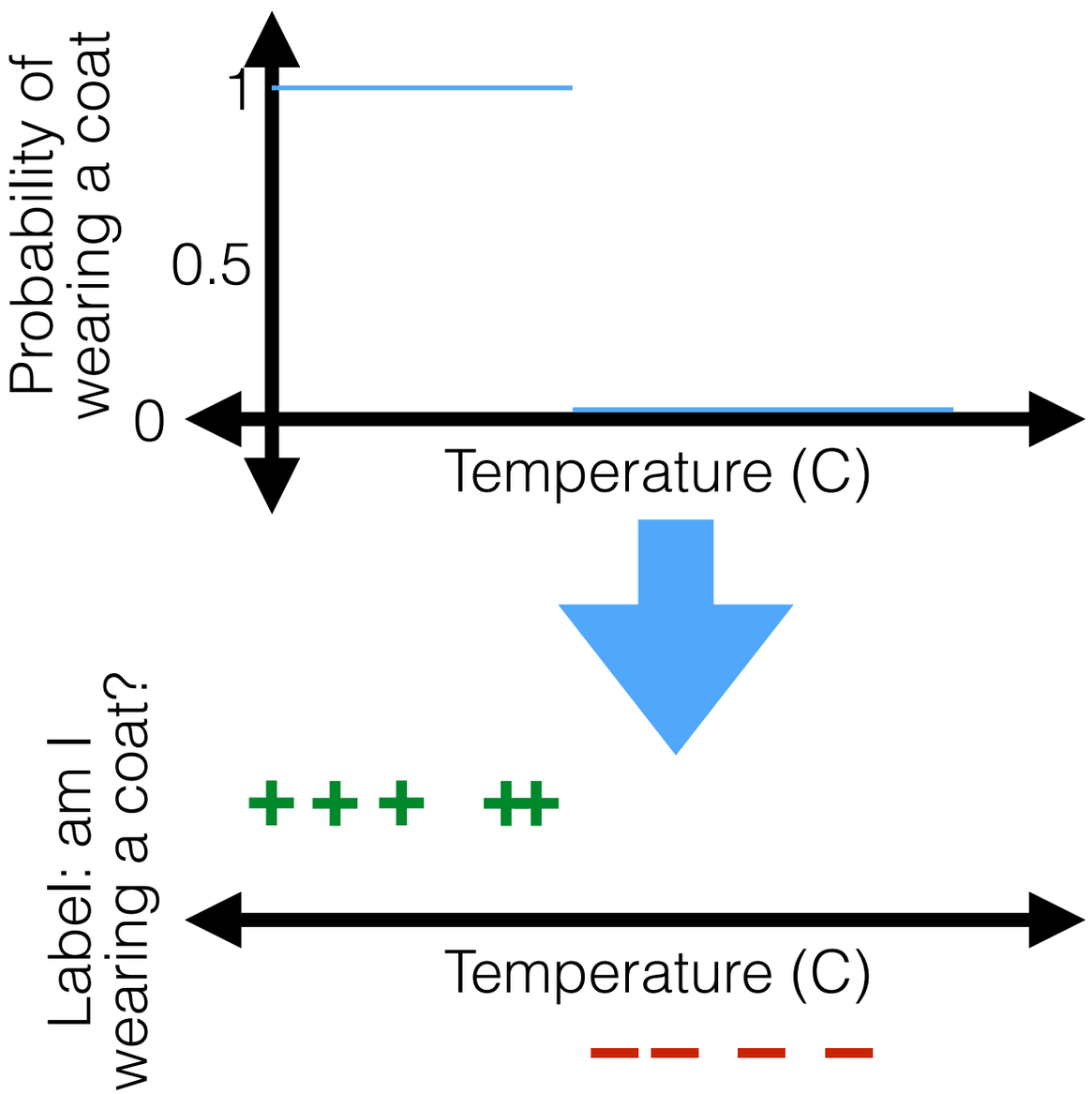


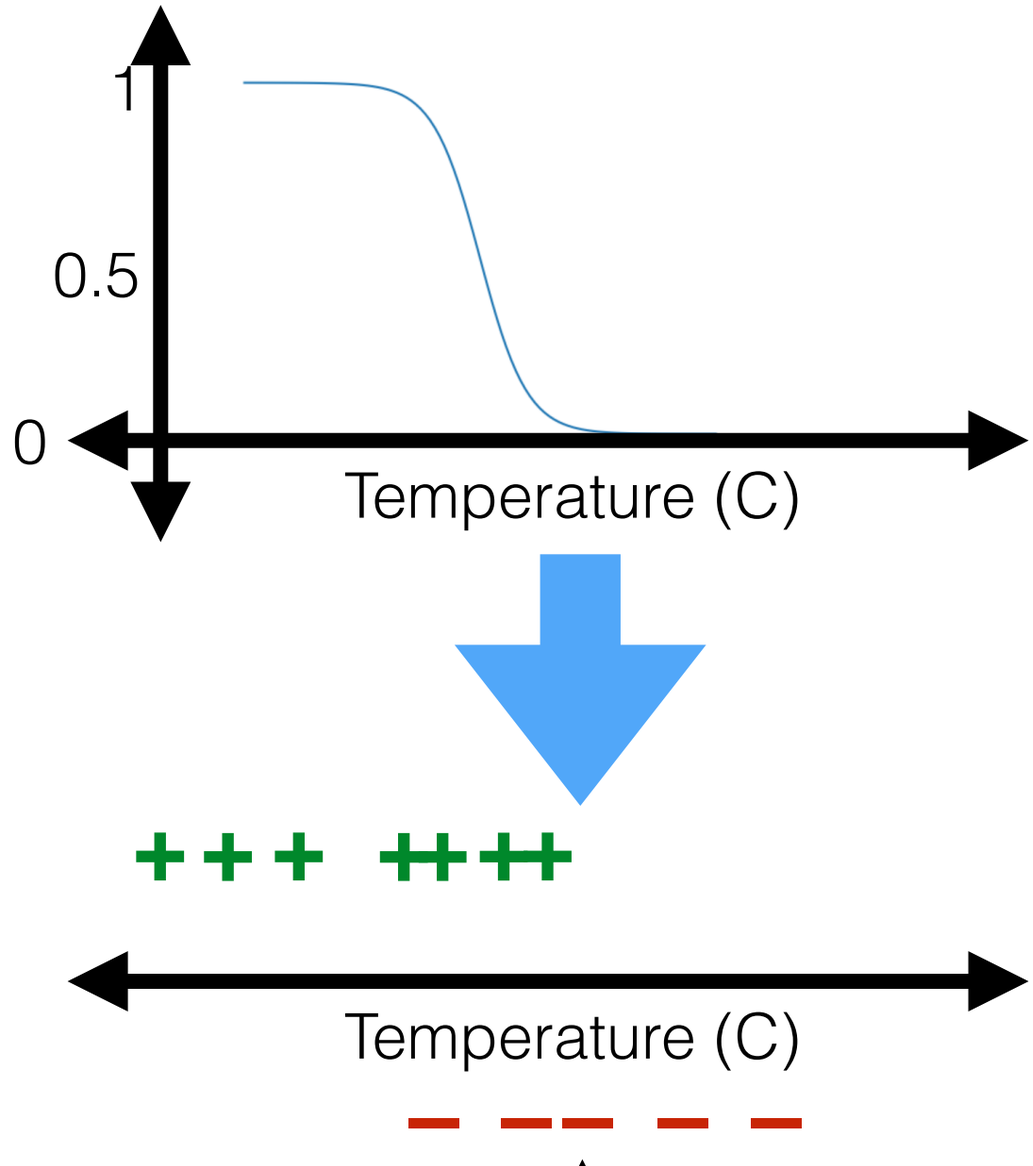
- How to make this shape?
  - Sigmoid/logistic function

$$\sigma(z) = \frac{1}{1 + \exp(-z)}$$



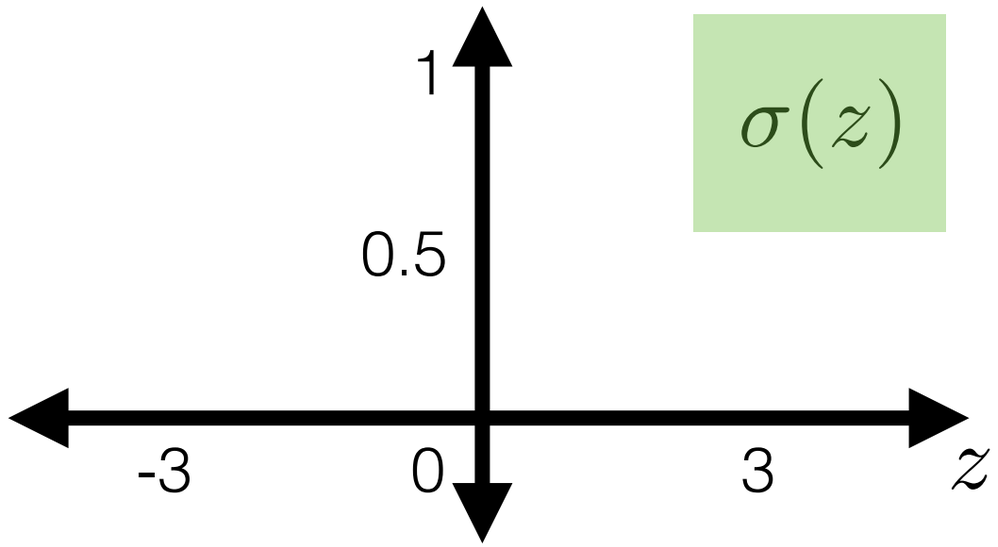
Capturing uncertainty



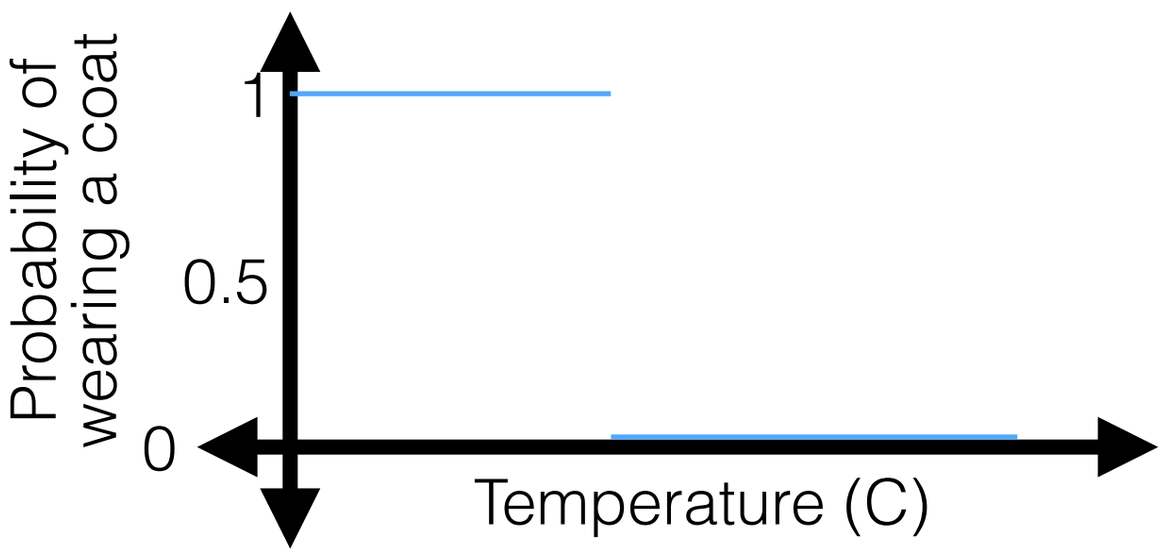


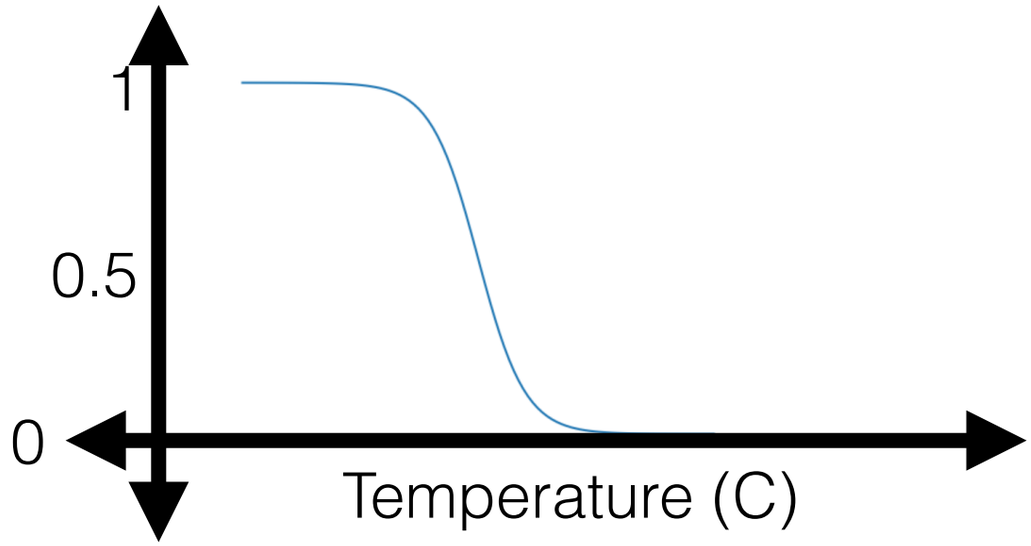
- How to make this shape?
  - Sigmoid/logistic function

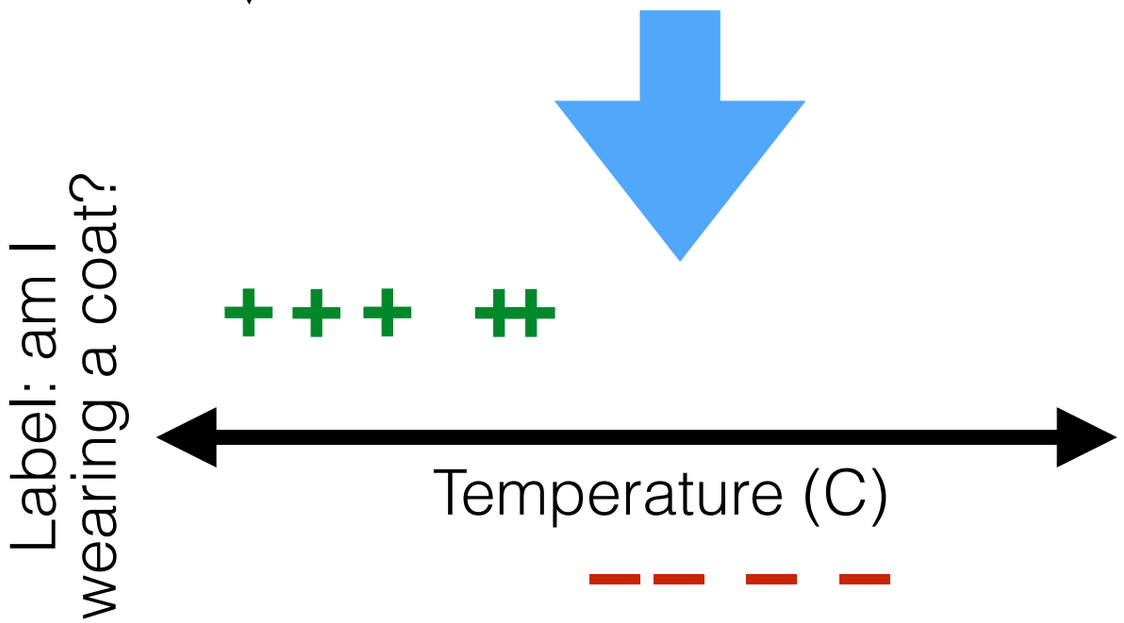
$$\sigma(z) = \frac{1}{1 + \exp(-z)}$$

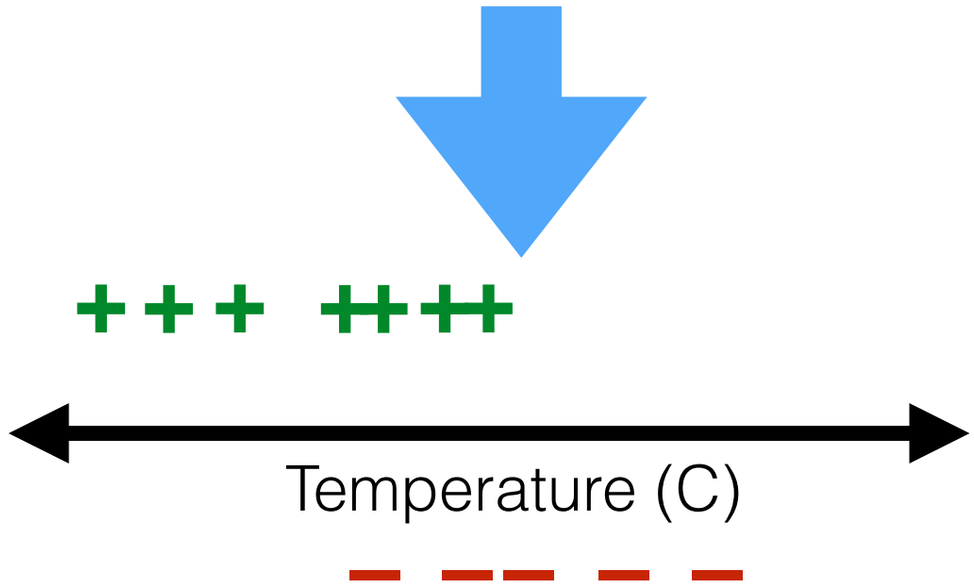


Capturing uncertainty



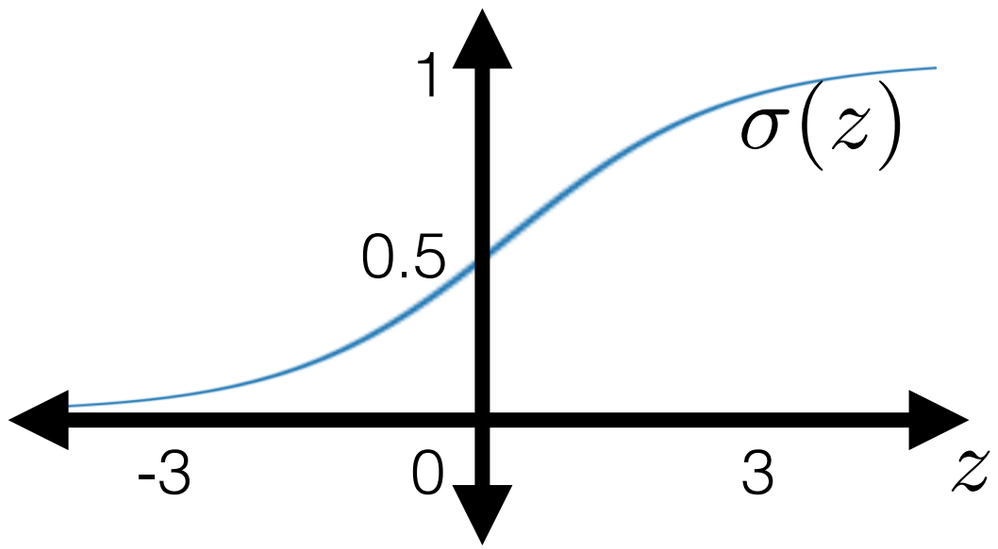




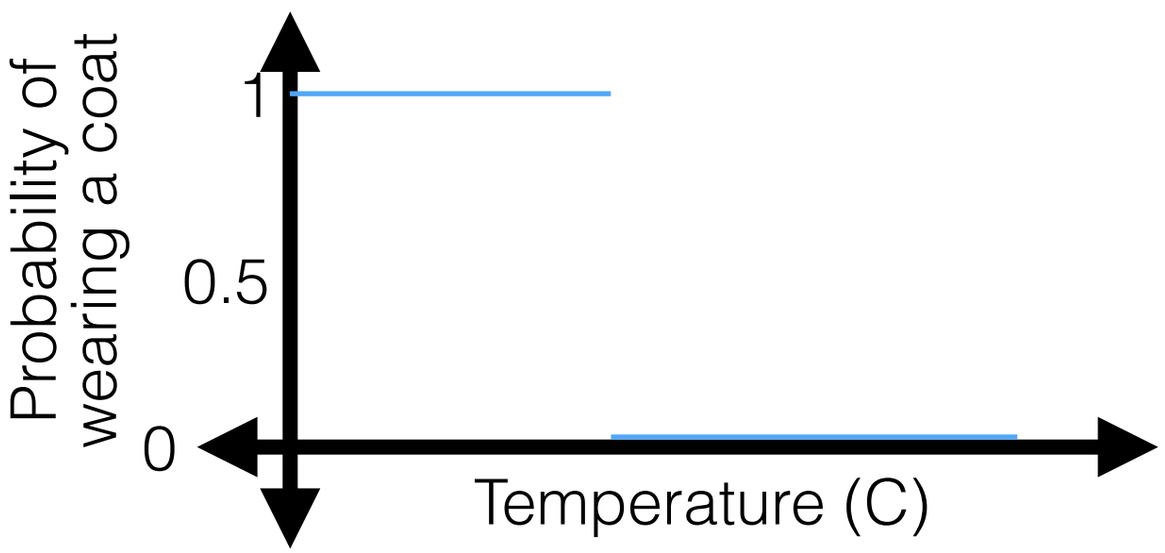


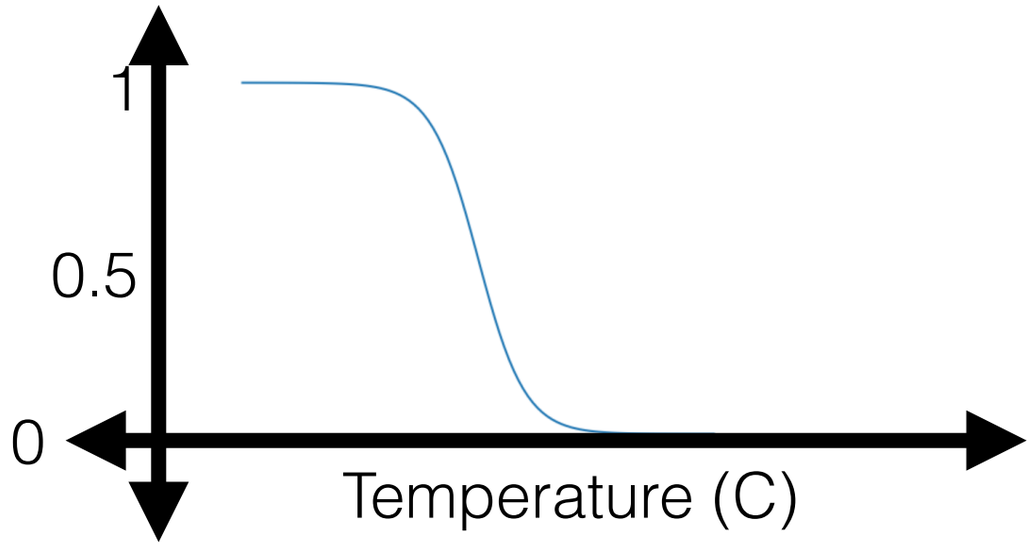
- How to make this shape?
  - Sigmoid/logistic function

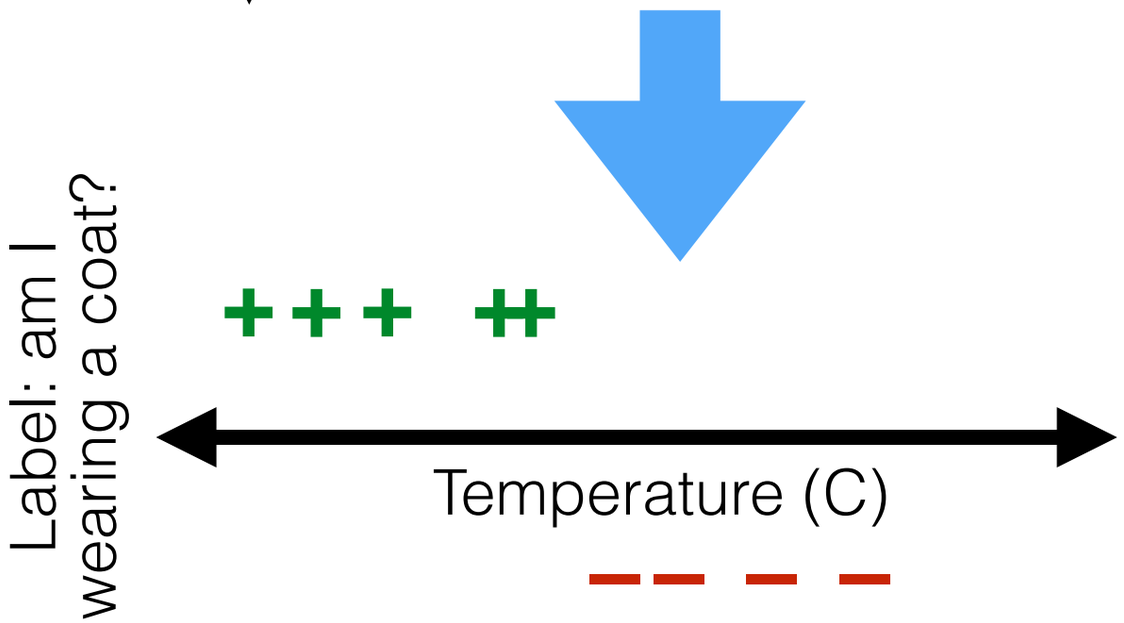
$$\sigma(z) = \frac{1}{1 + \exp(-z)}$$

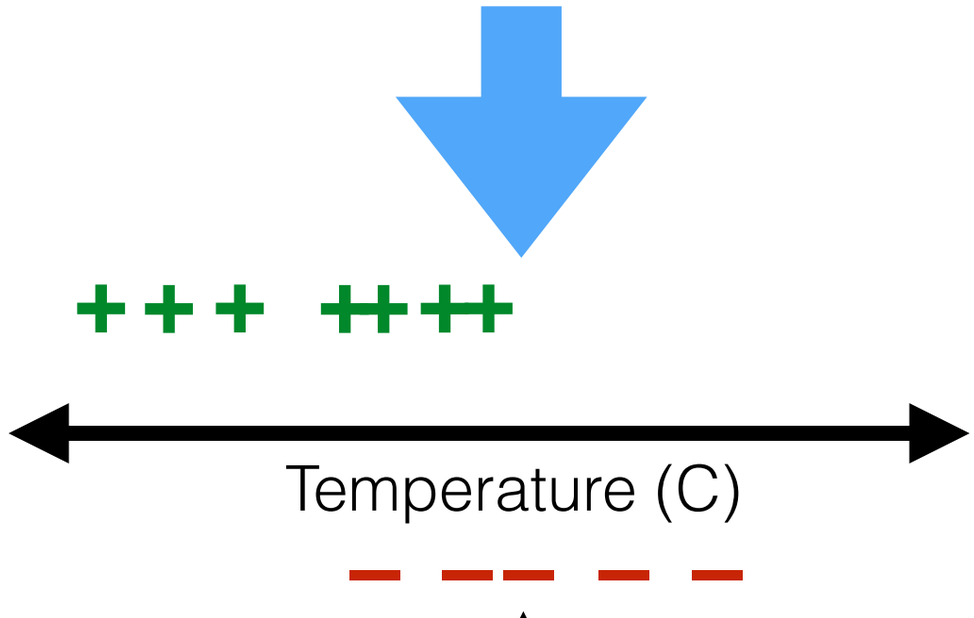


Capturing uncertainty



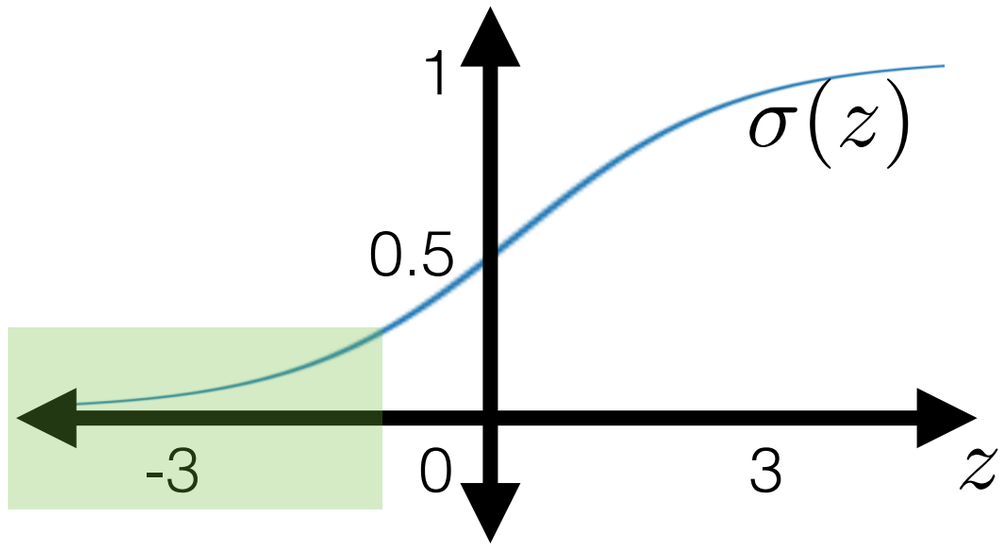




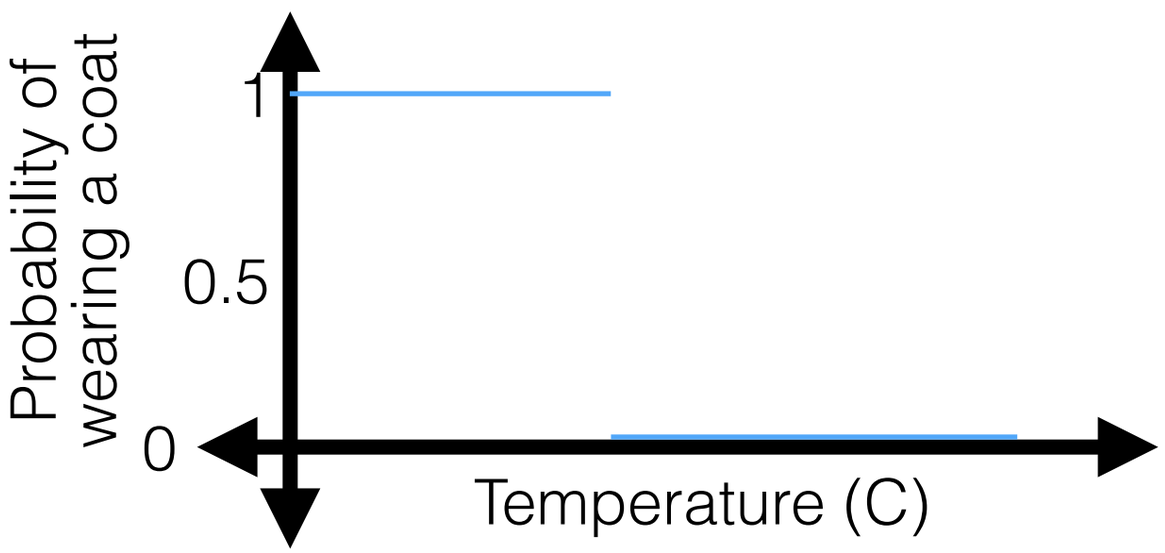


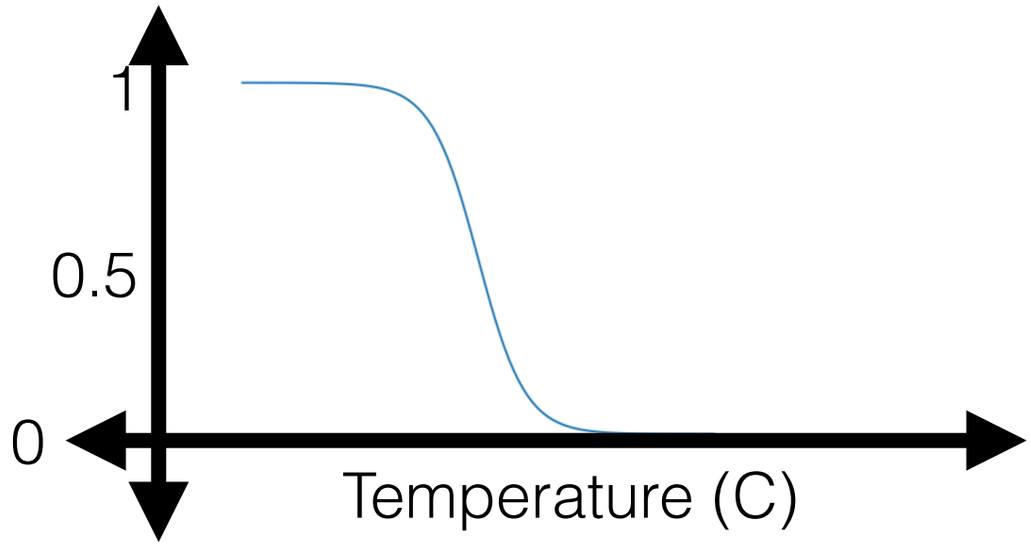
- How to make this shape?
  - Sigmoid/logistic function

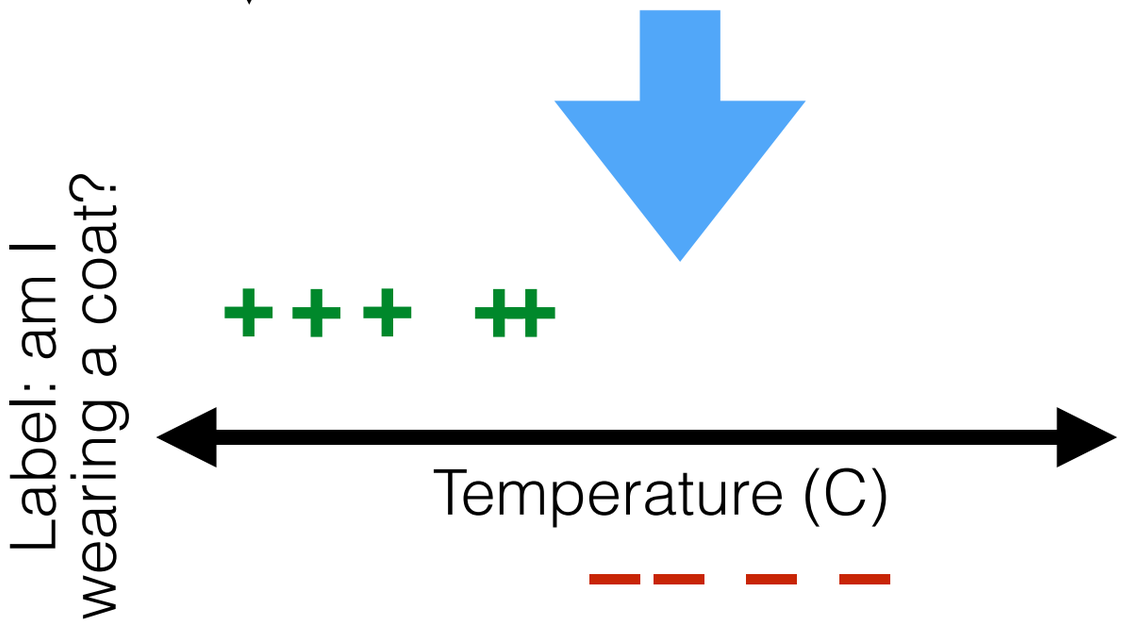
$$\sigma(z) = \frac{1}{1 + \exp(-z)}$$

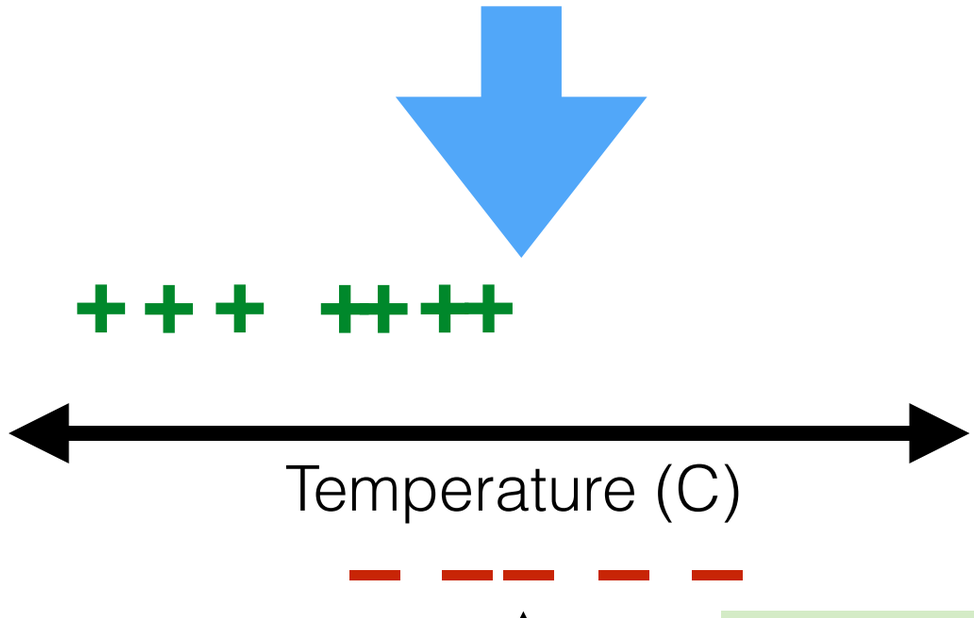


Capturing uncertainty



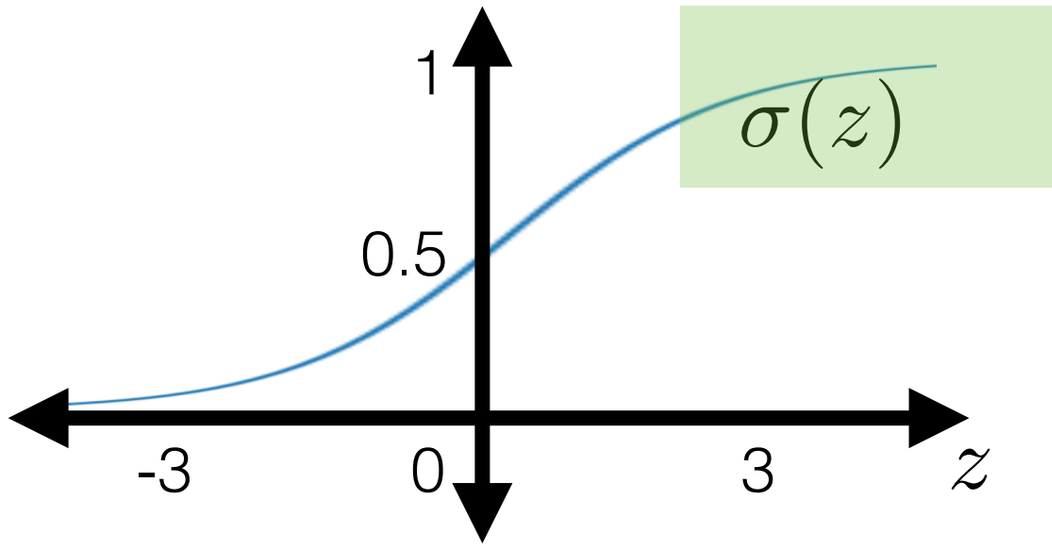




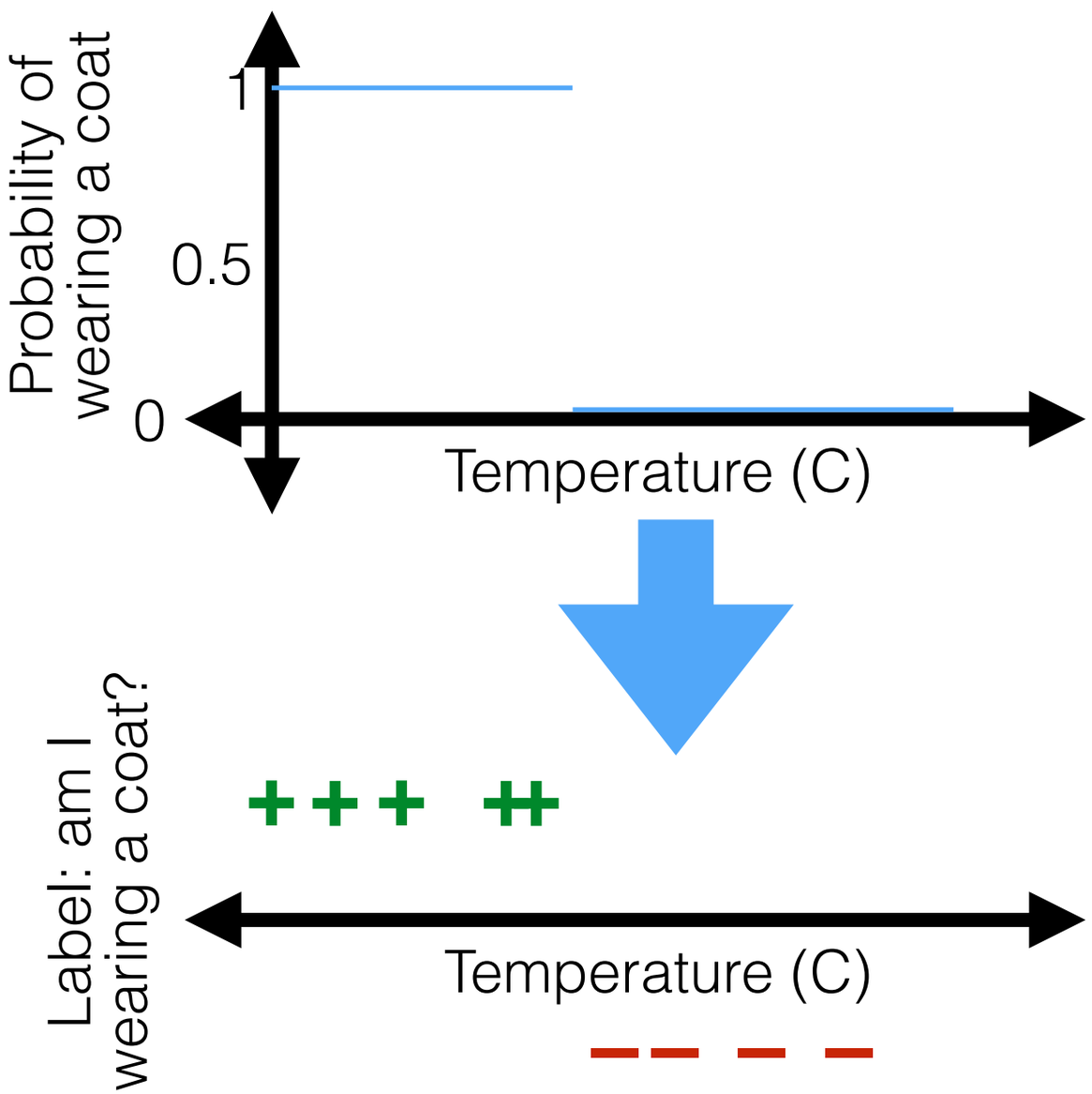


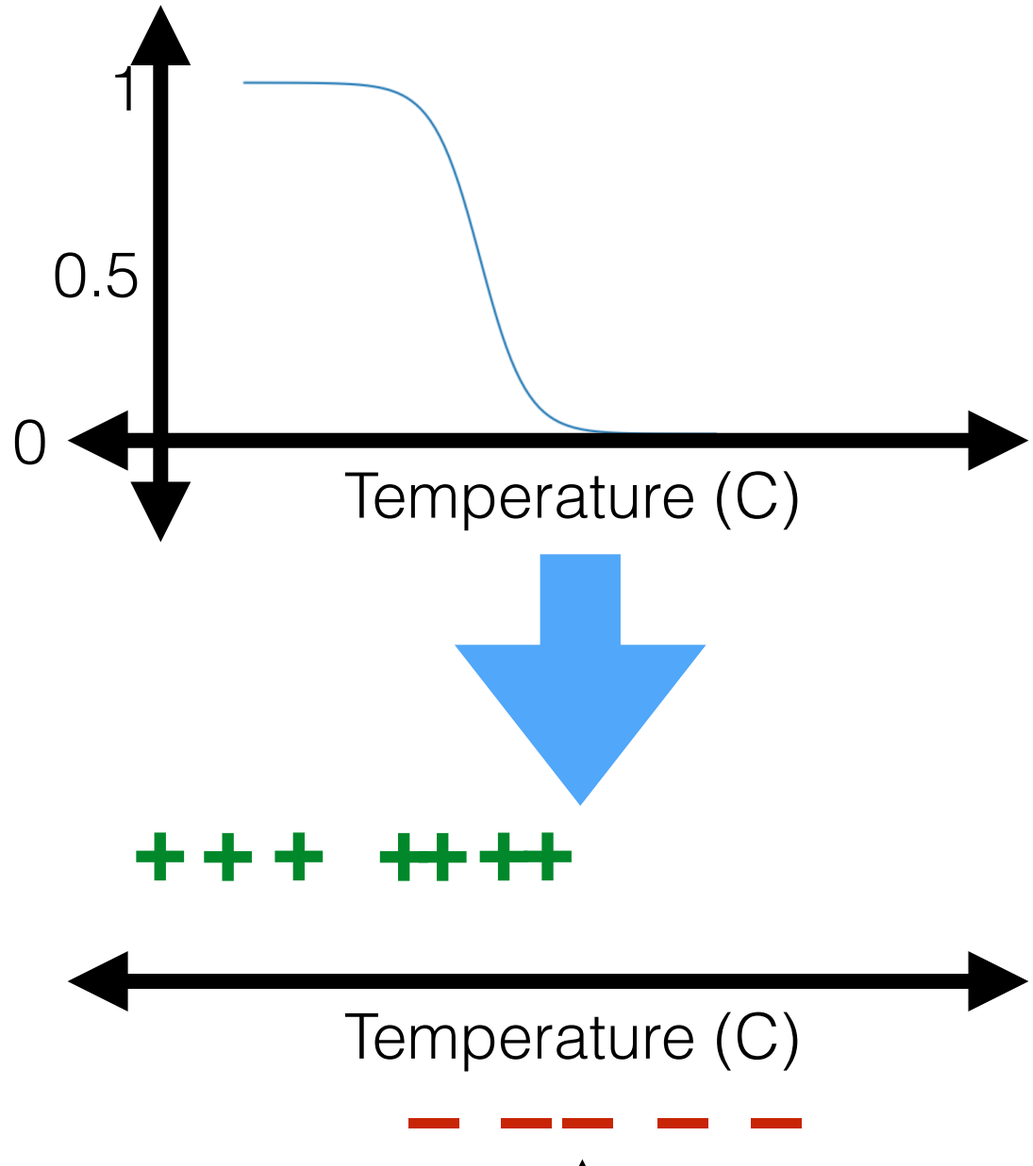
- How to make this shape?
  - Sigmoid/logistic function

$$\sigma(z) = \frac{1}{1 + \exp(-z)}$$



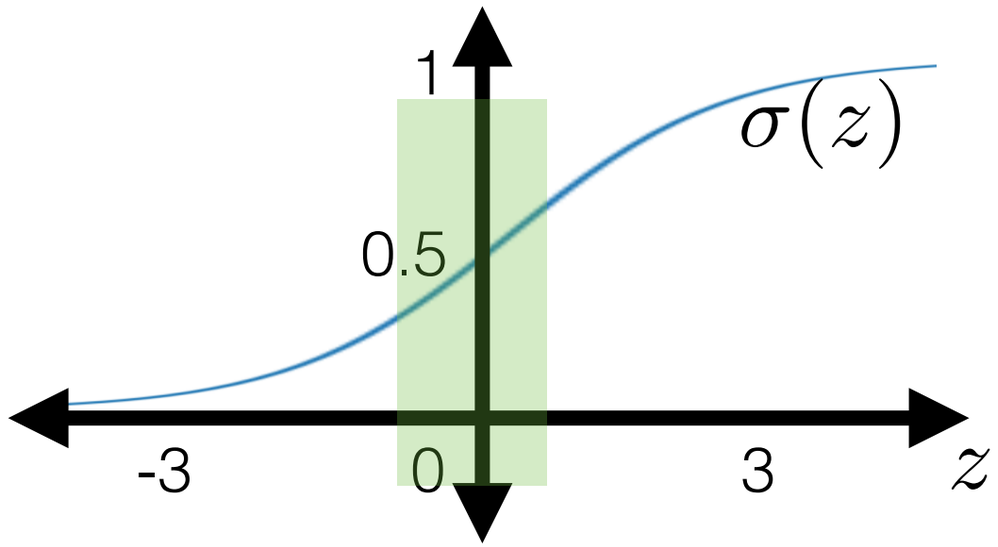
Capturing uncertainty



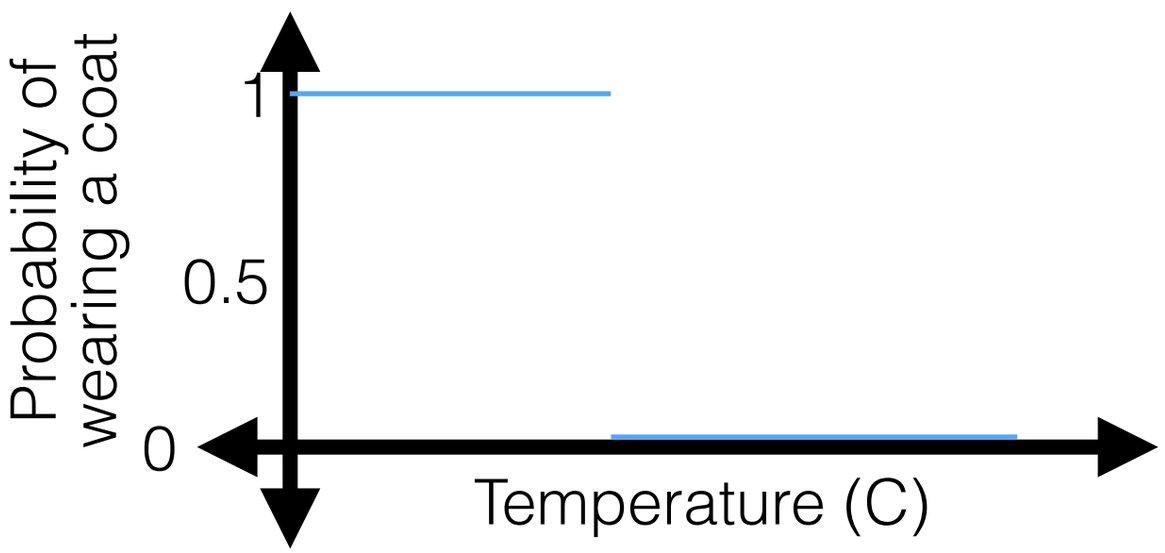


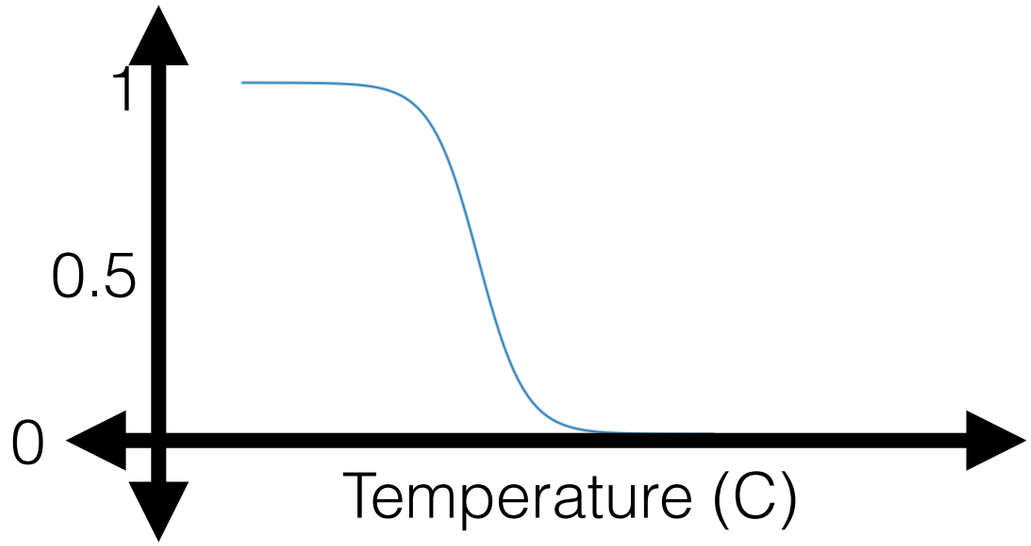
- How to make this shape?
  - Sigmoid/logistic function

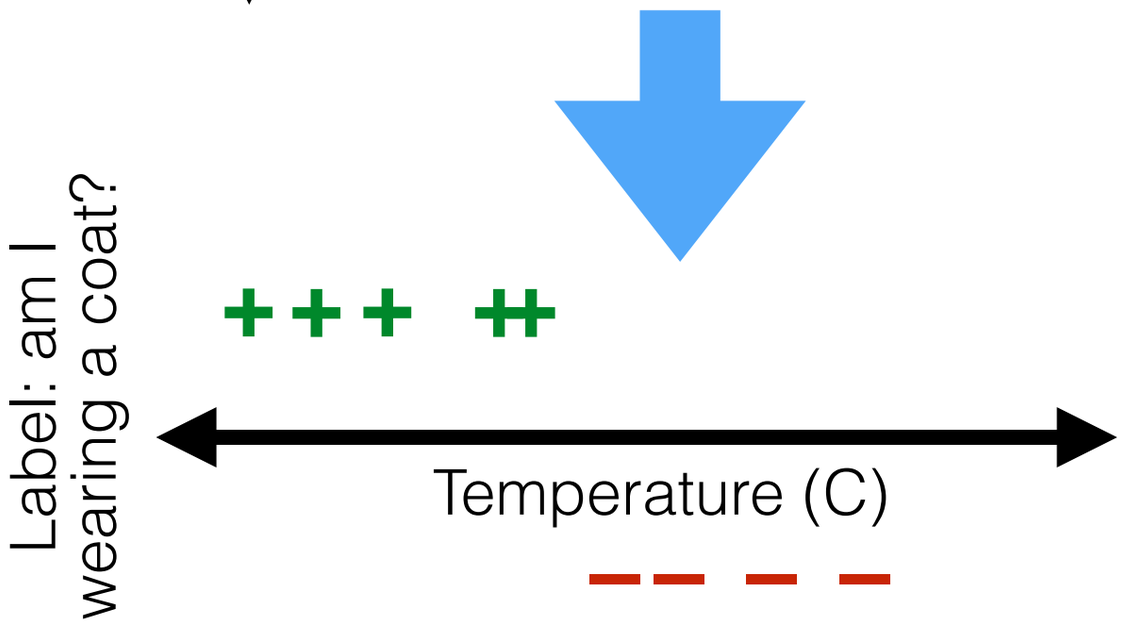
$$\sigma(z) = \frac{1}{1 + \exp(-z)}$$

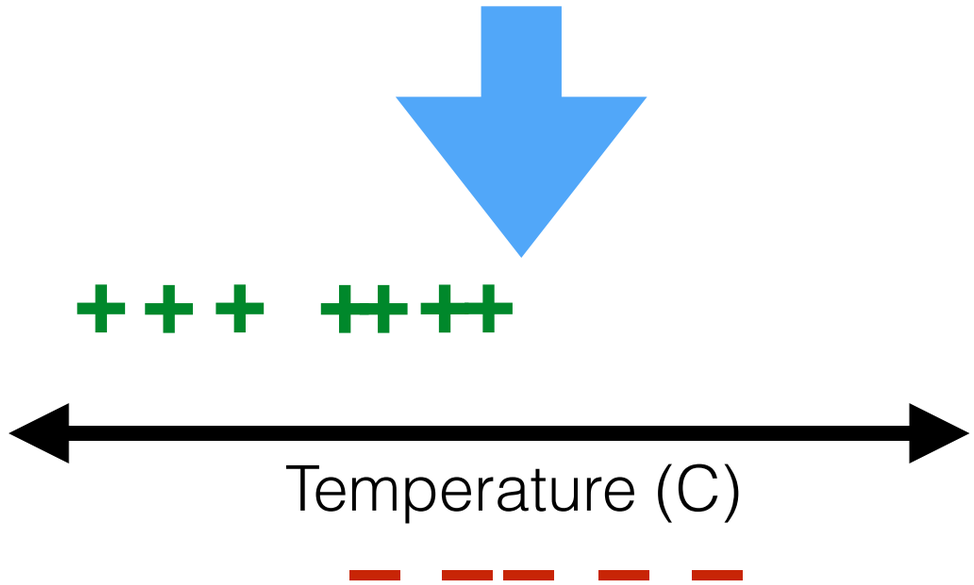


Capturing uncertainty



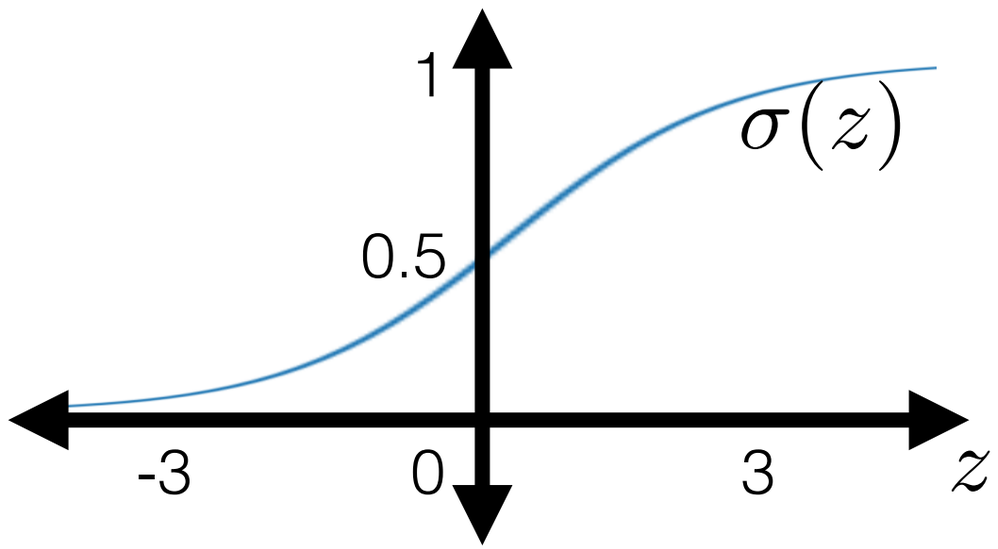




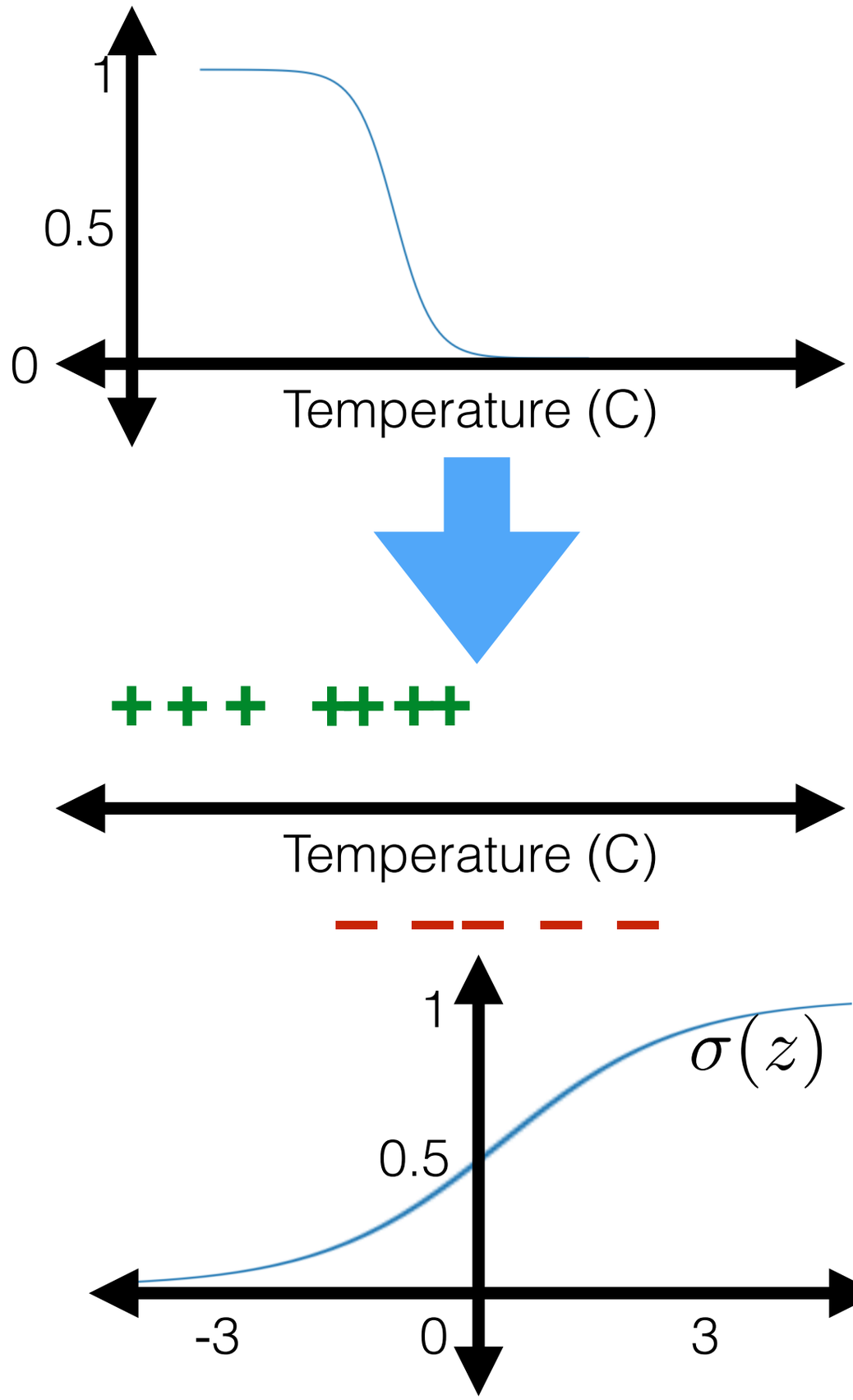


- How to make this shape?
  - Sigmoid/logistic function

$$\sigma(z) = \frac{1}{1 + \exp(-z)}$$



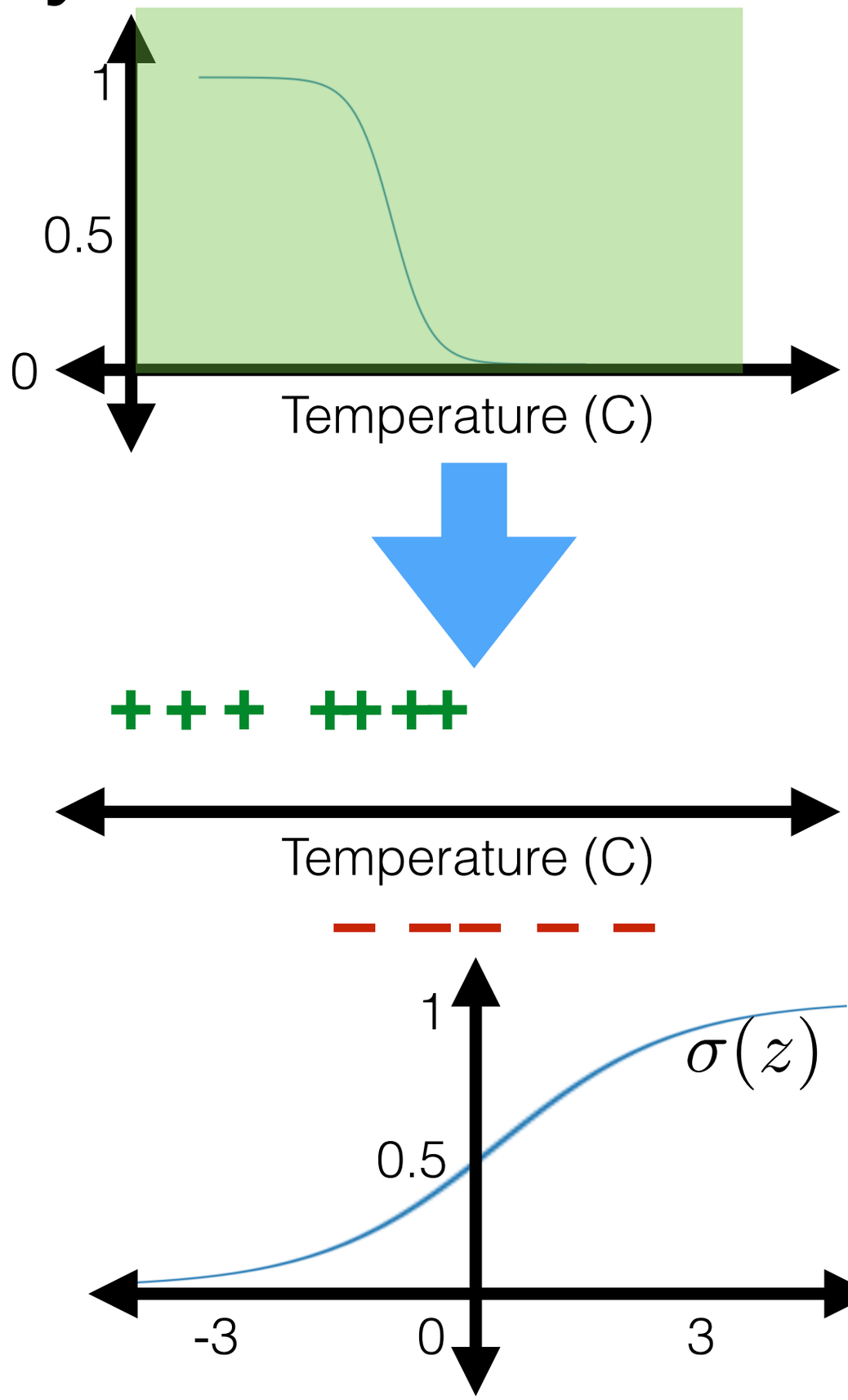
Capturing uncertainty



- How to make this shape?
  - Sigmoid/logistic function

$$\sigma(z) = \frac{1}{1 + \exp(-z)}$$

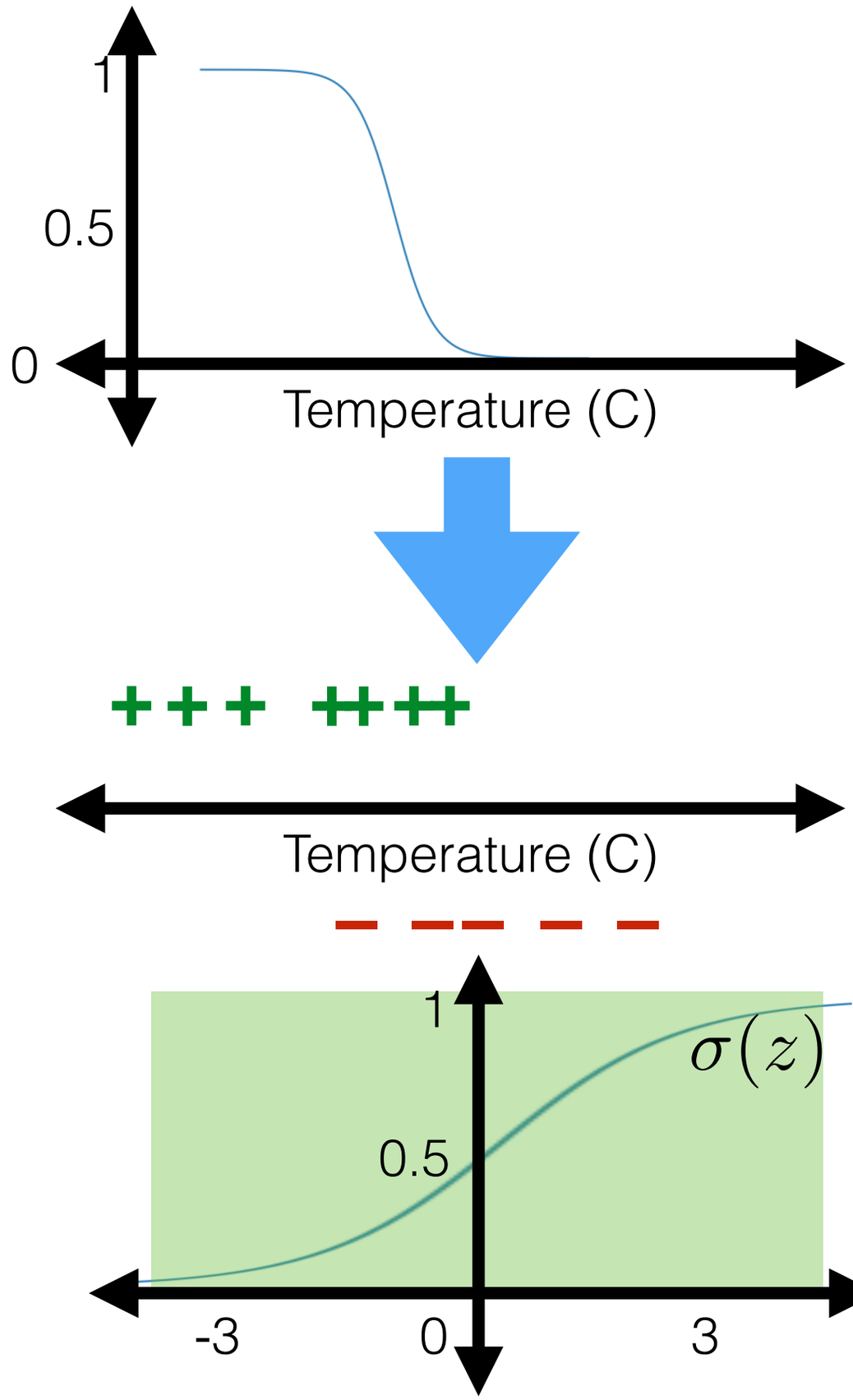
Capturing uncertainty



- How to make this shape?
  - Sigmoid/logistic function

$$\sigma(z) = \frac{1}{1 + \exp(-z)}$$

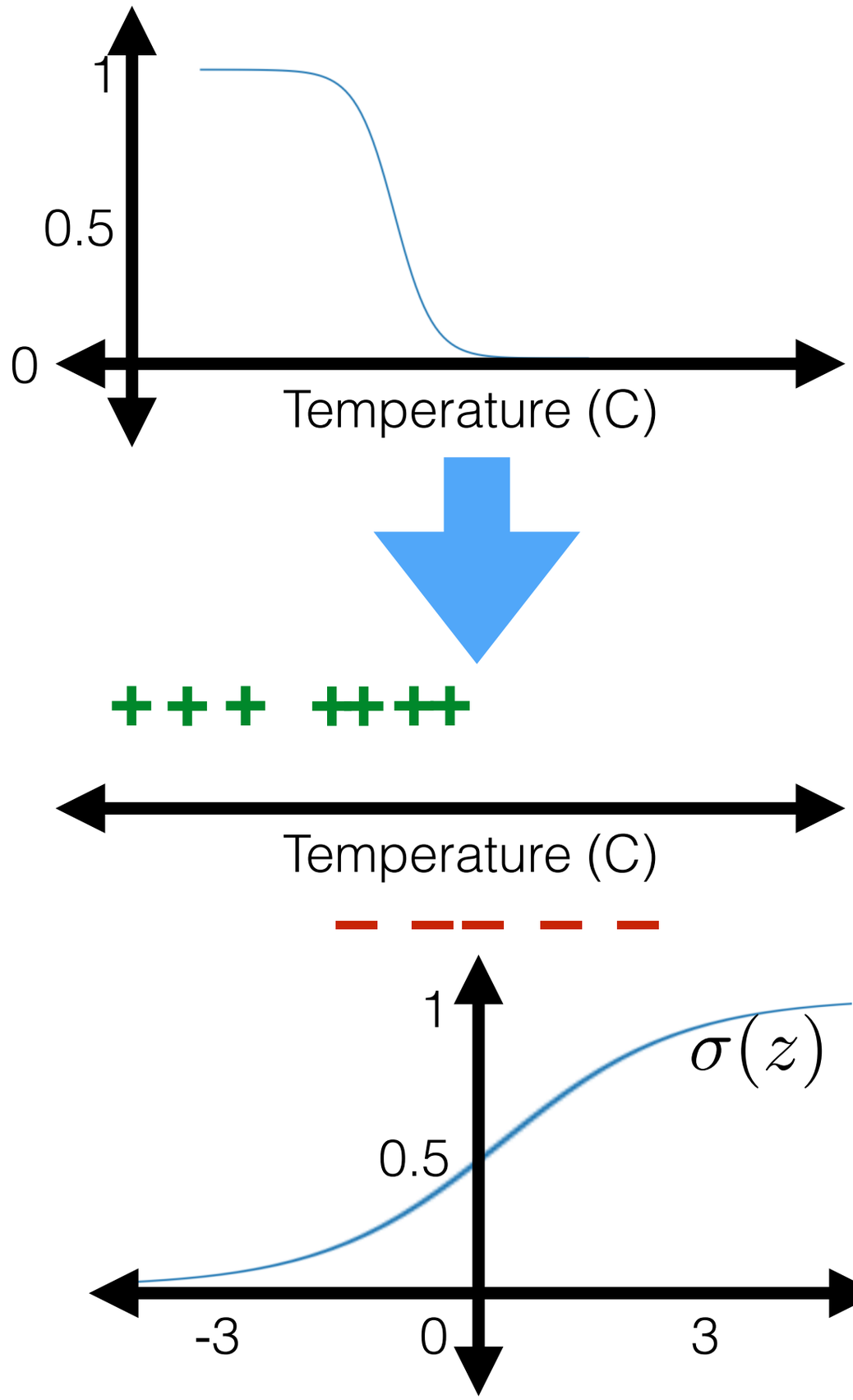
Capturing uncertainty



- How to make this shape?
  - Sigmoid/logistic function

$$\sigma(z) = \frac{1}{1 + \exp(-z)}$$

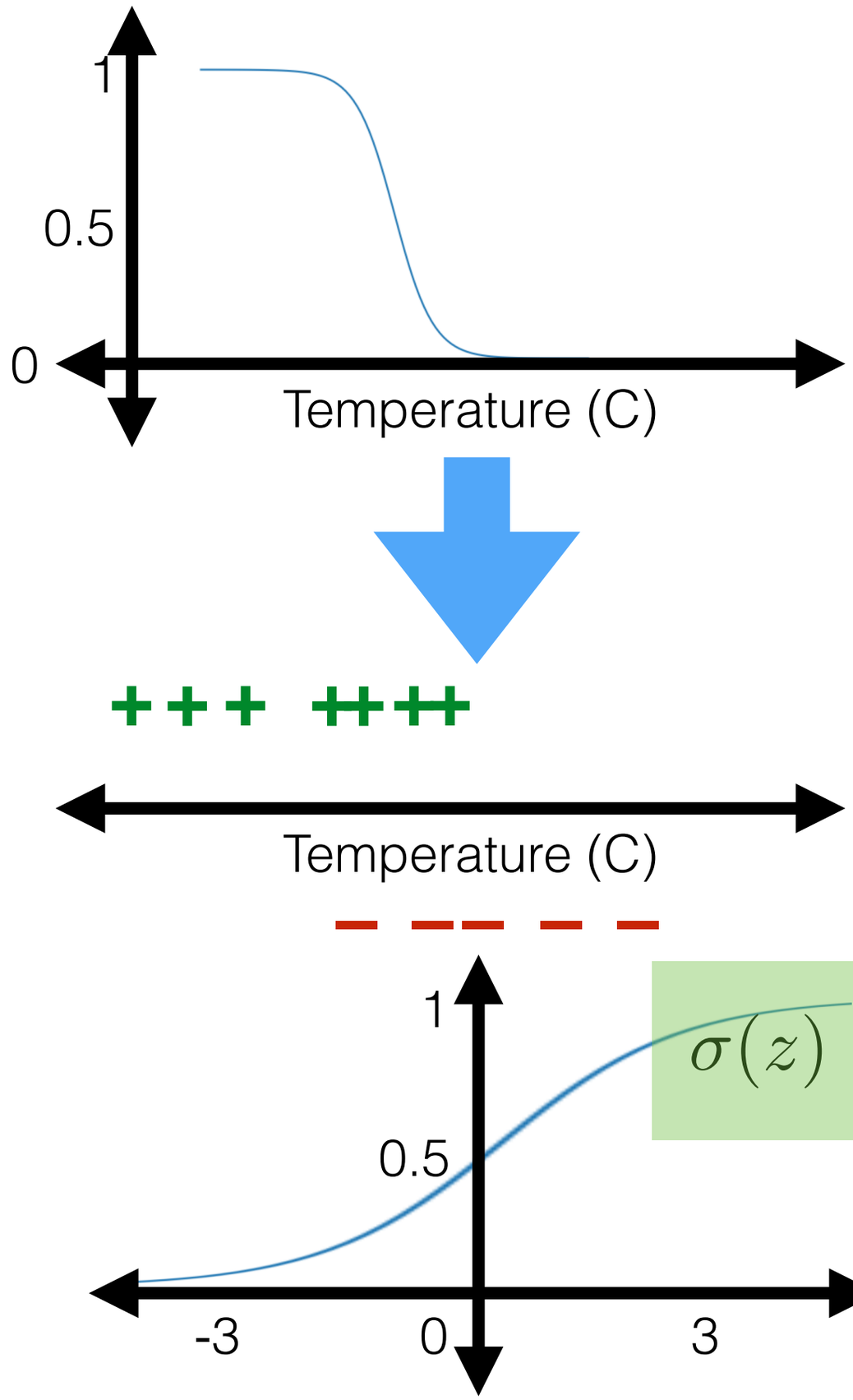
Capturing uncertainty



- How to make this shape?
  - Sigmoid/logistic function

$$\sigma(z) = \frac{1}{1 + \exp(-z)}$$

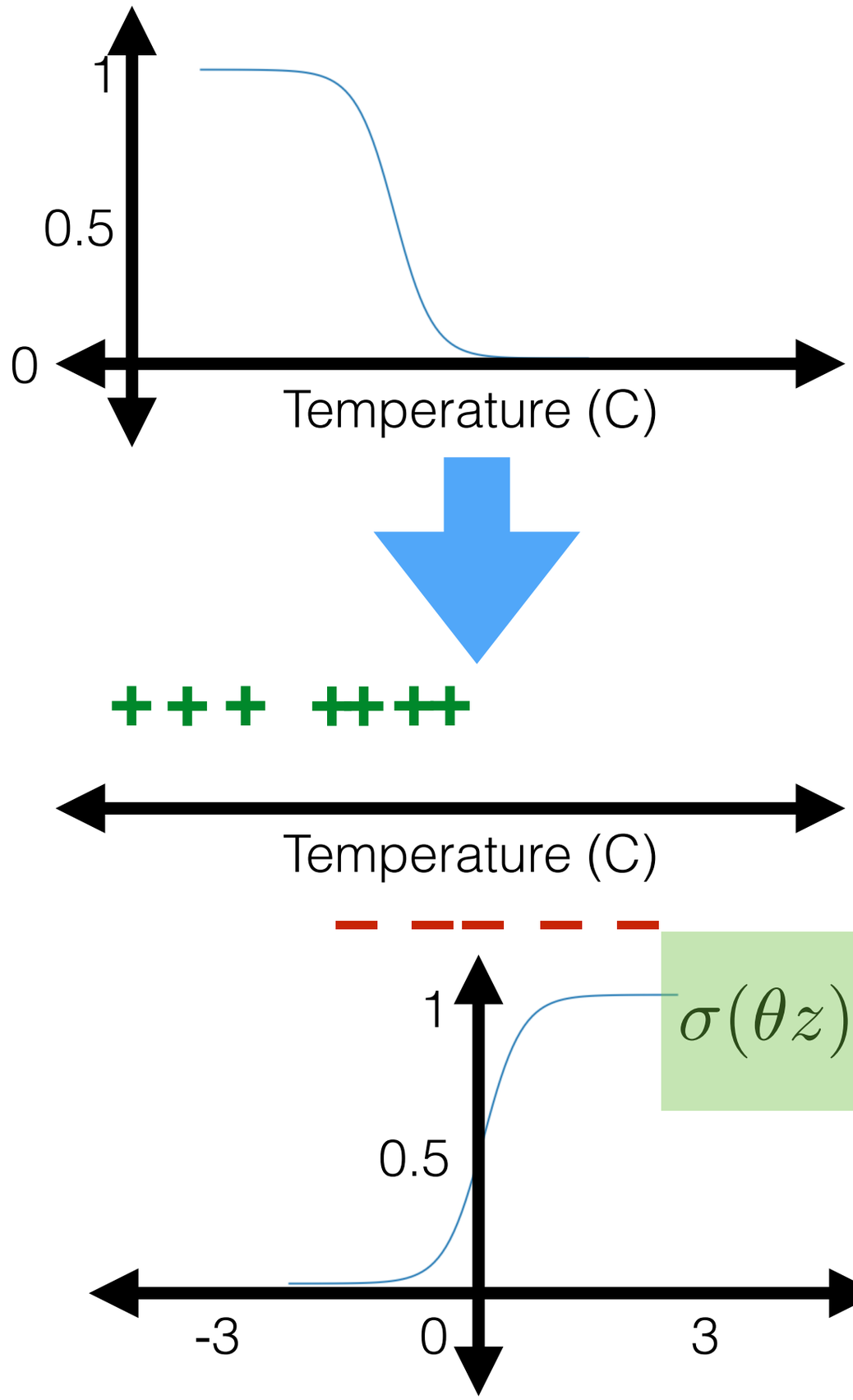
Capturing uncertainty



- How to make this shape?
  - Sigmoid/logistic function

$$\sigma(z) = \frac{1}{1 + \exp(-z)}$$

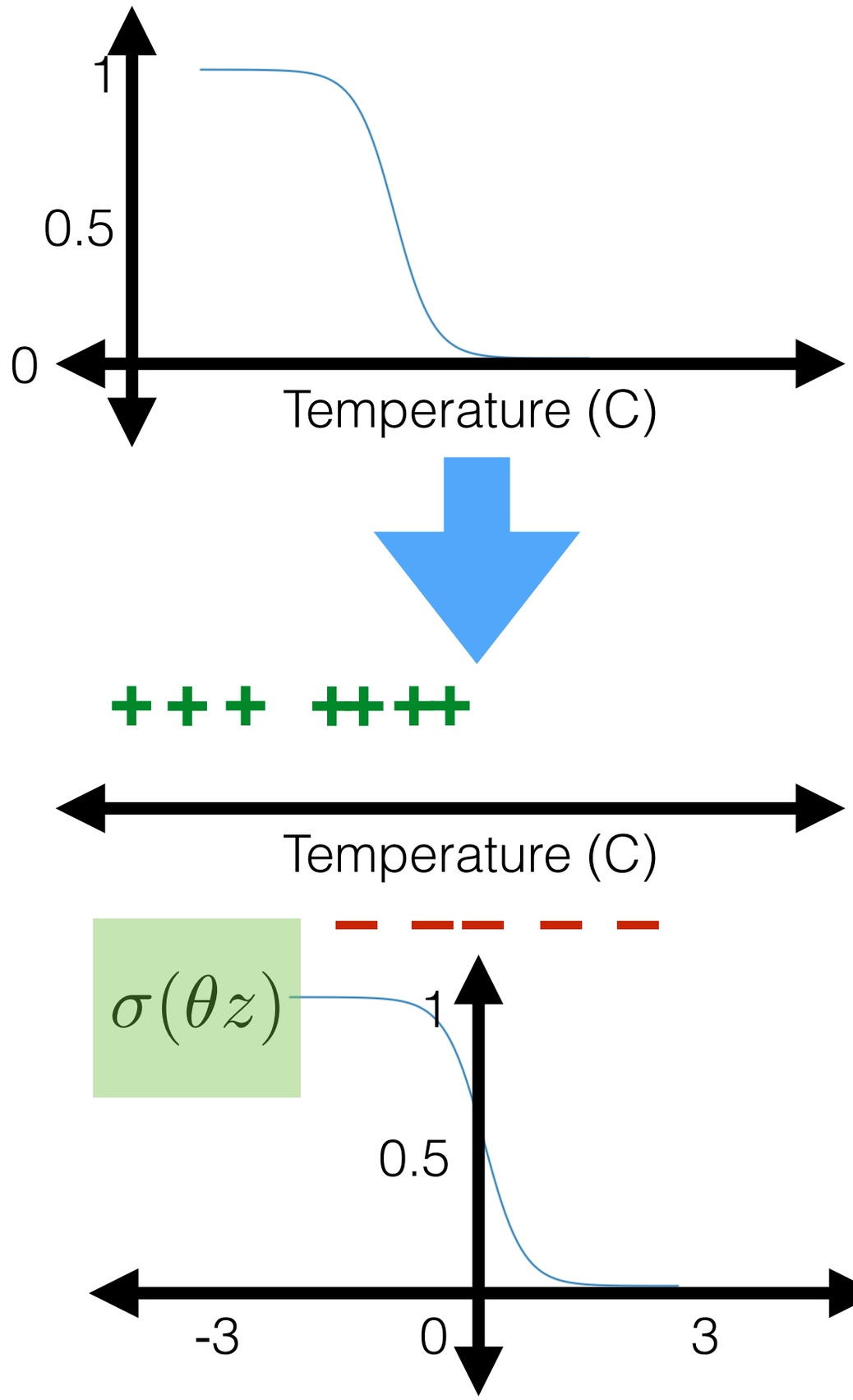
Capturing uncertainty



- How to make this shape?
  - Sigmoid/logistic function

$$\sigma(z) = \frac{1}{1 + \exp(-z)}$$

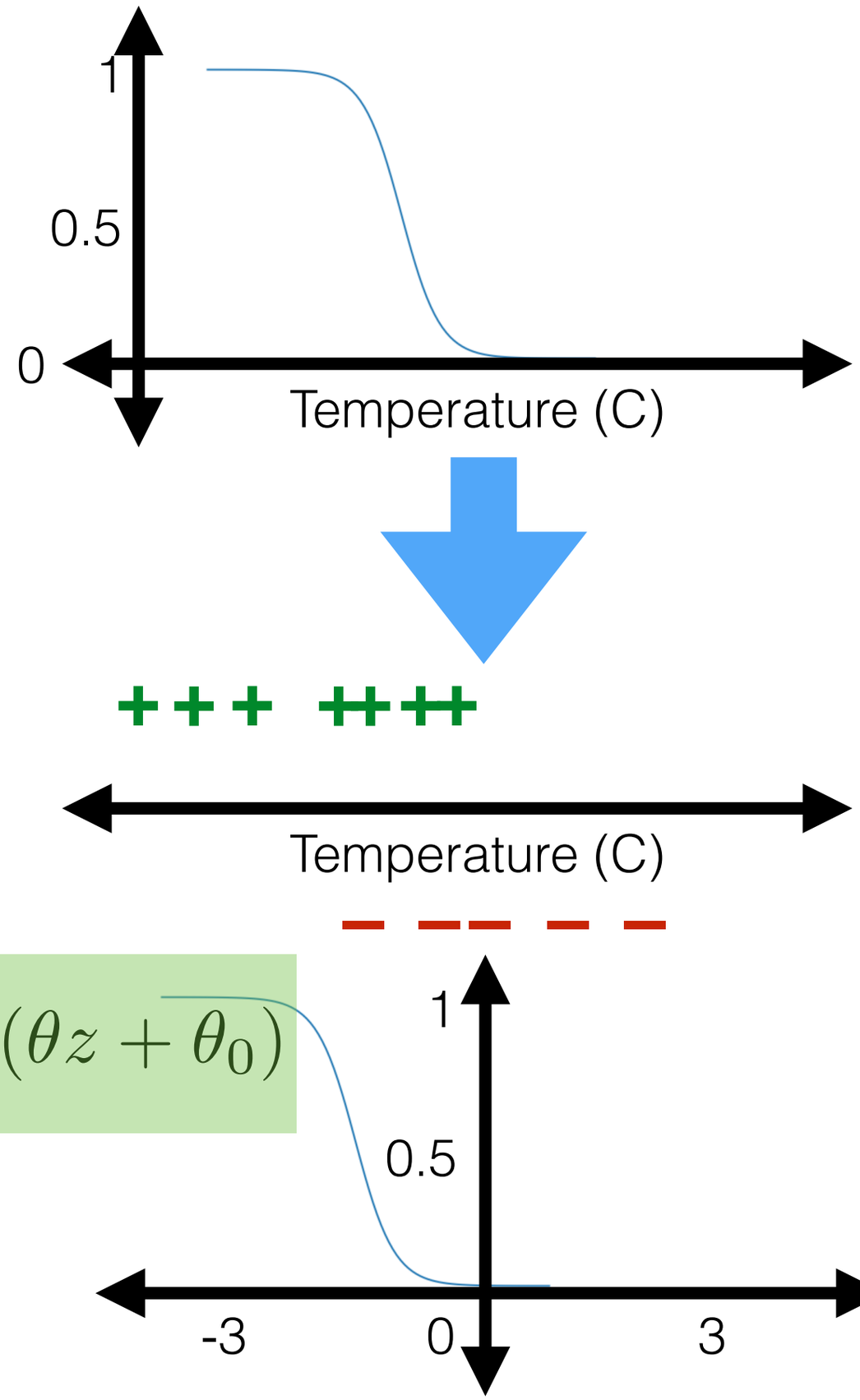
Capturing uncertainty



- How to make this shape?
  - Sigmoid/logistic function

$$\sigma(z) = \frac{1}{1 + \exp(-z)}$$

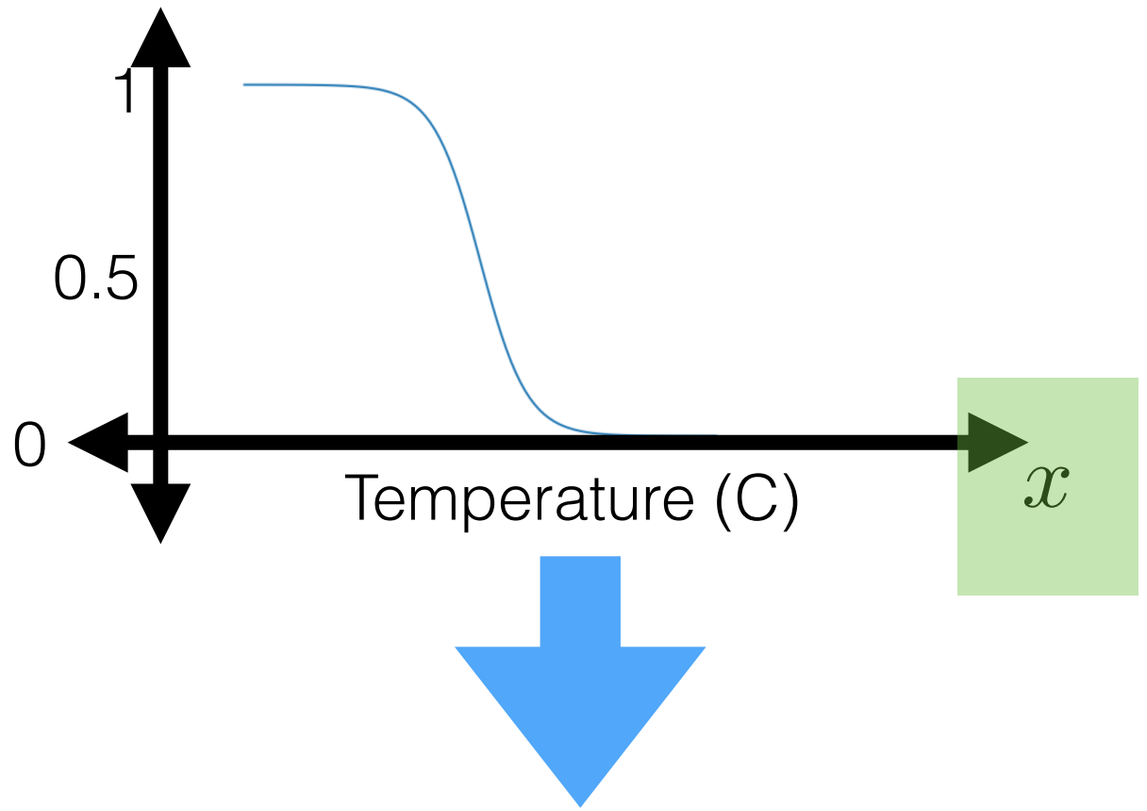
Capturing uncertainty

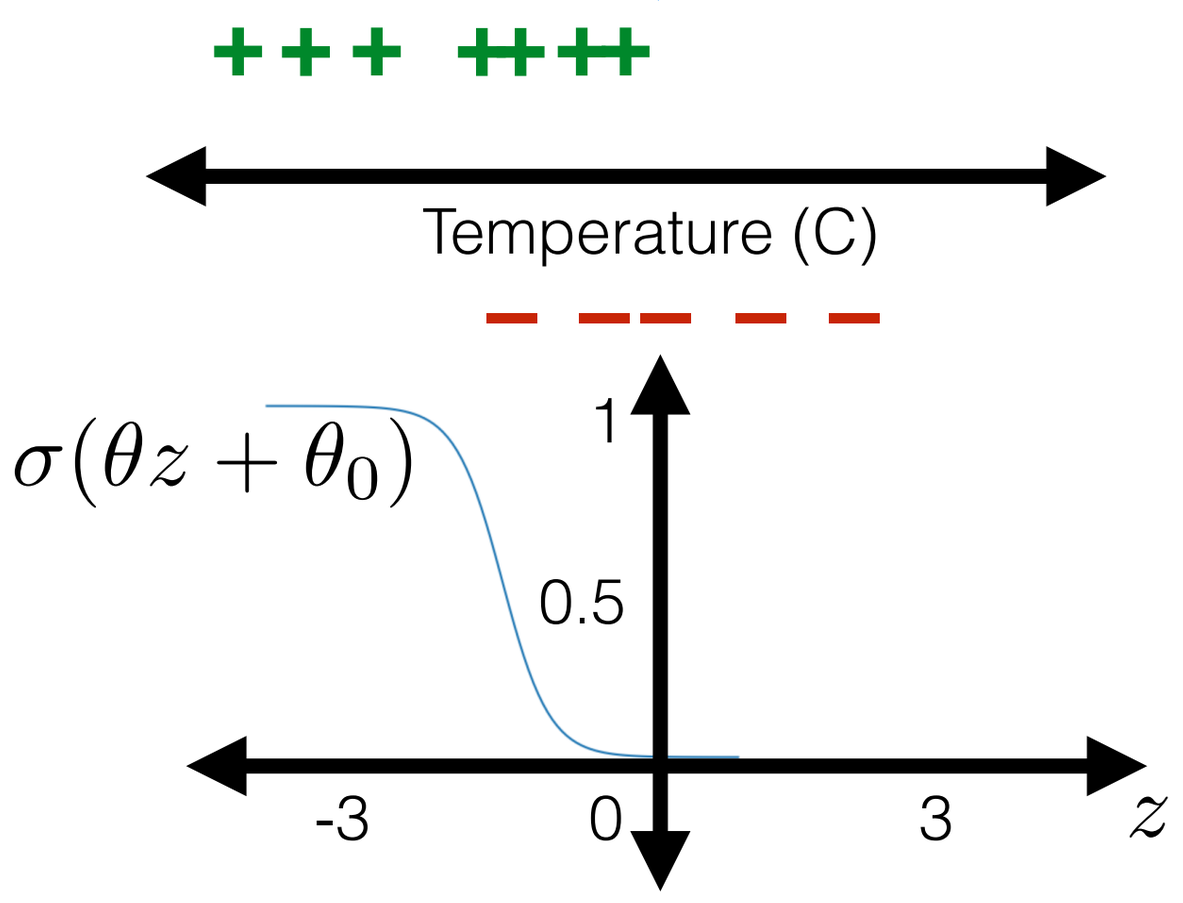


- How to make this shape?
  - Sigmoid/logistic function

$$\sigma(z) = \frac{1}{1 + \exp(-z)}$$

Capturing uncertainty

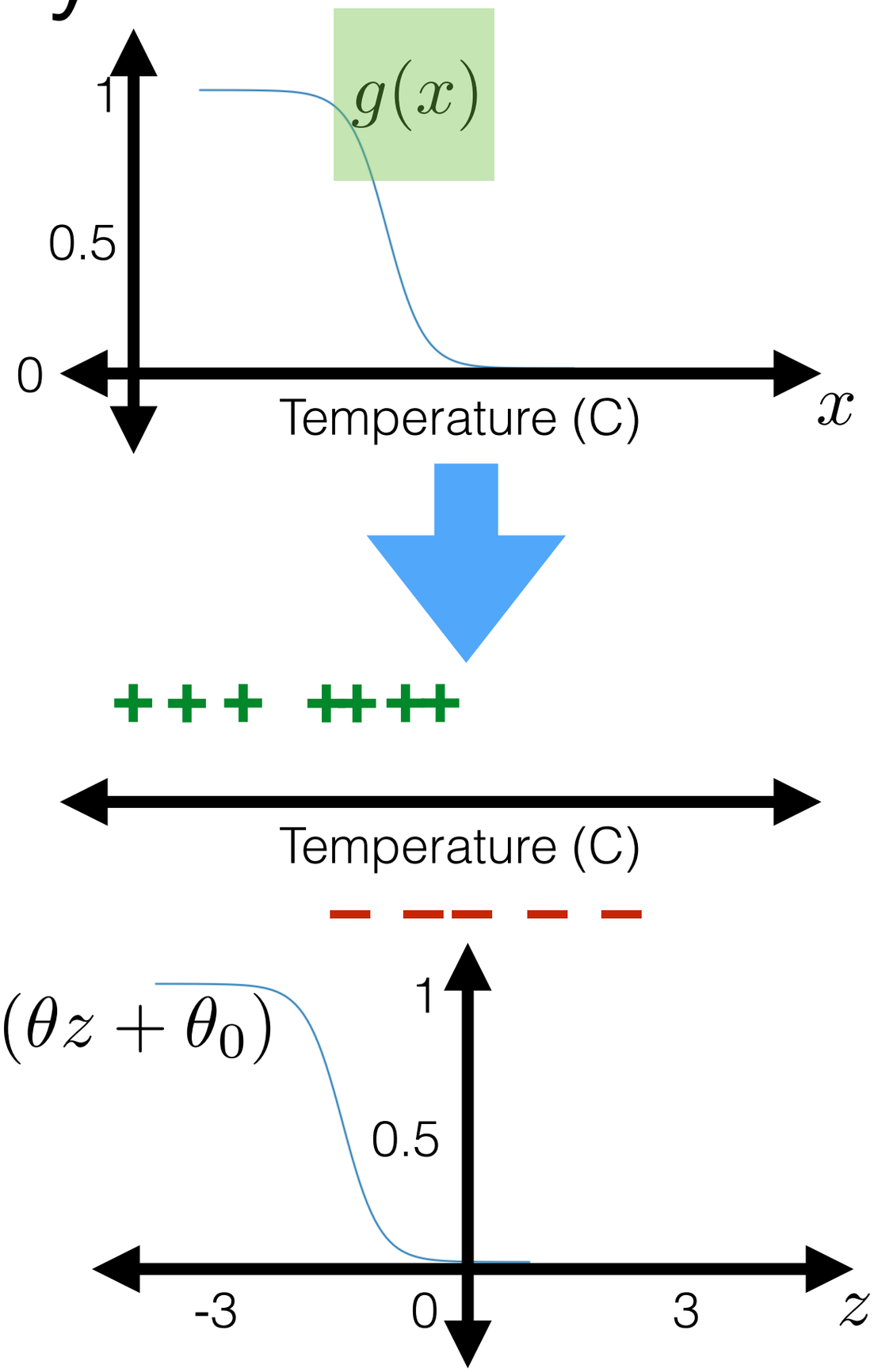




- How to make this shape?
  - Sigmoid/logistic function

$$\sigma(z) = \frac{1}{1 + \exp(-z)}$$

Capturing uncertainty

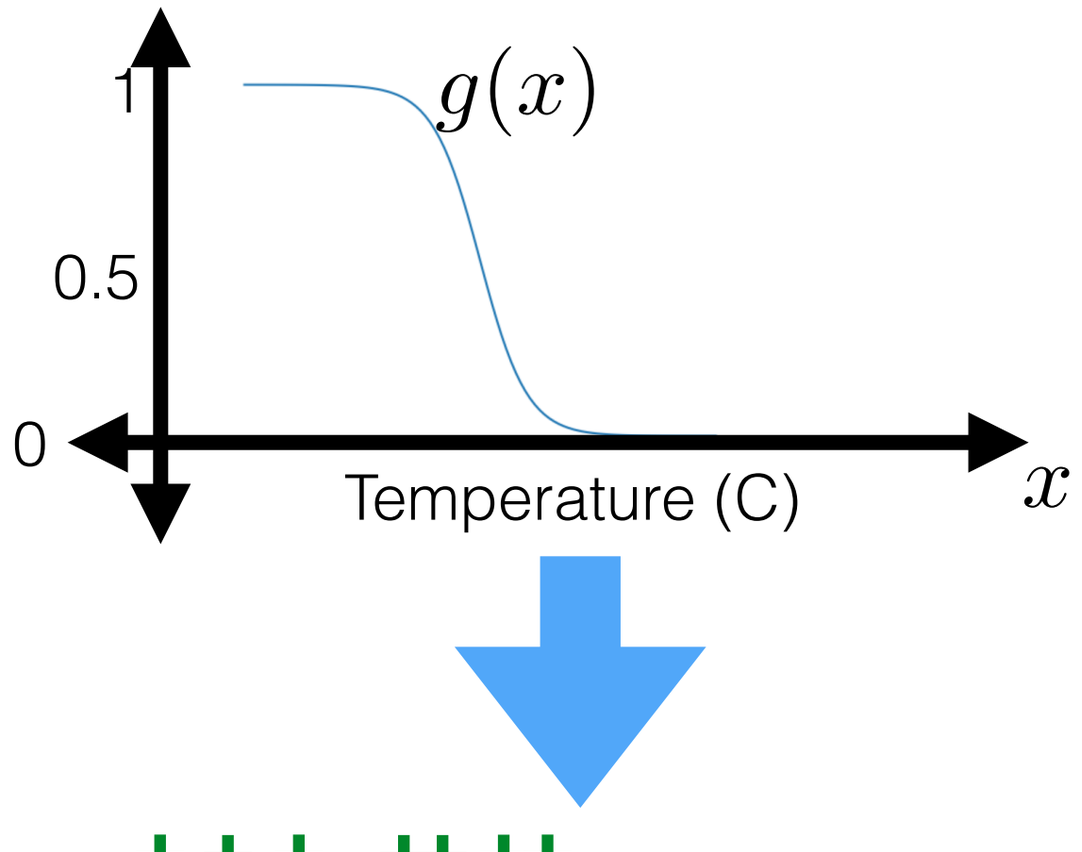


- How to make this shape?
  - Sigmoid/logistic function

$$\sigma(z) = \frac{1}{1 + \exp(-z)}$$

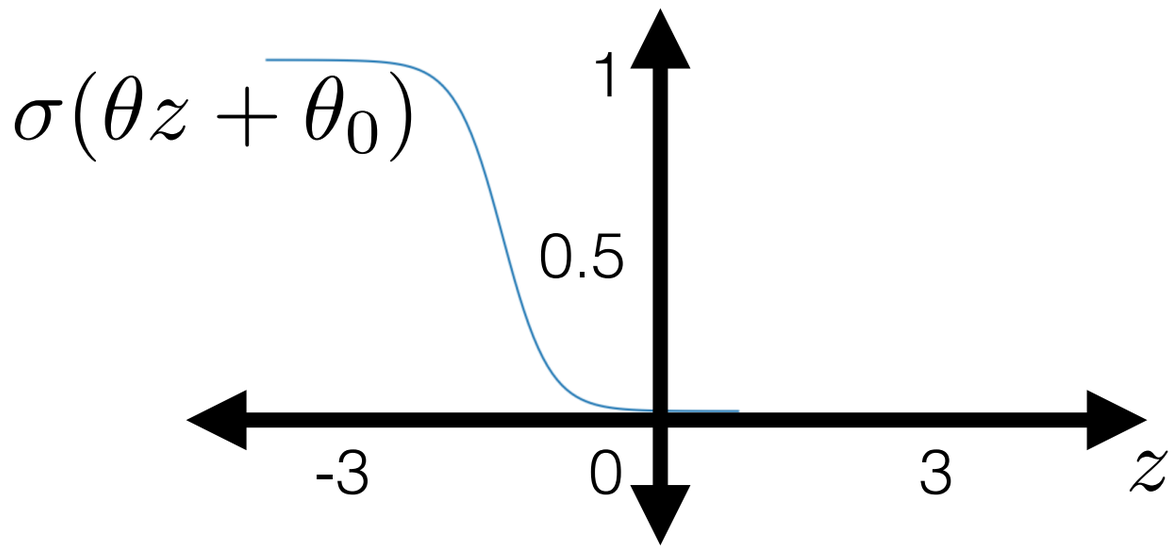
Capturing uncertainty

$$g(x) = \sigma(\theta x + \theta_0)$$



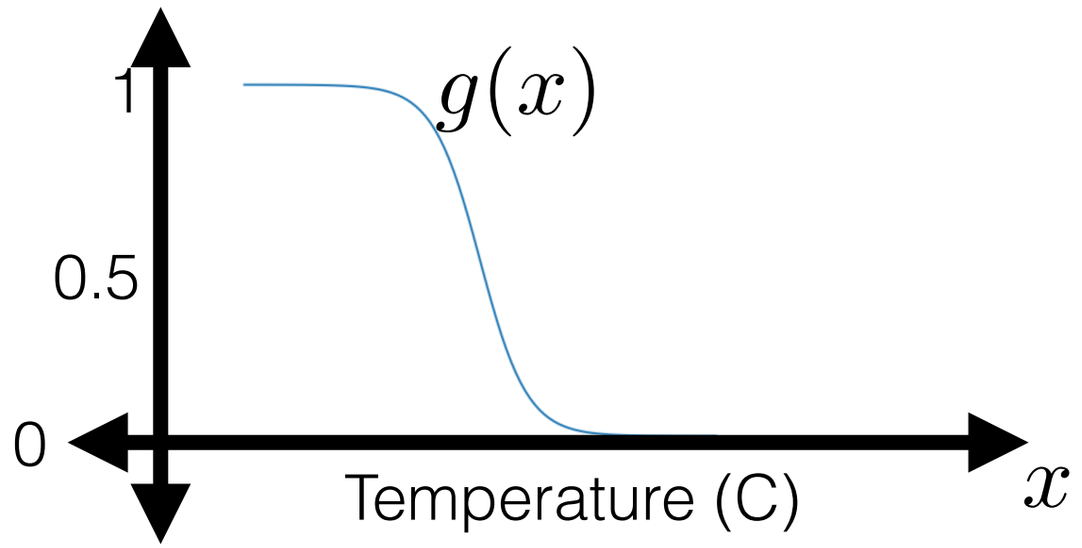
- How to make this shape?
  - Sigmoid/logistic function

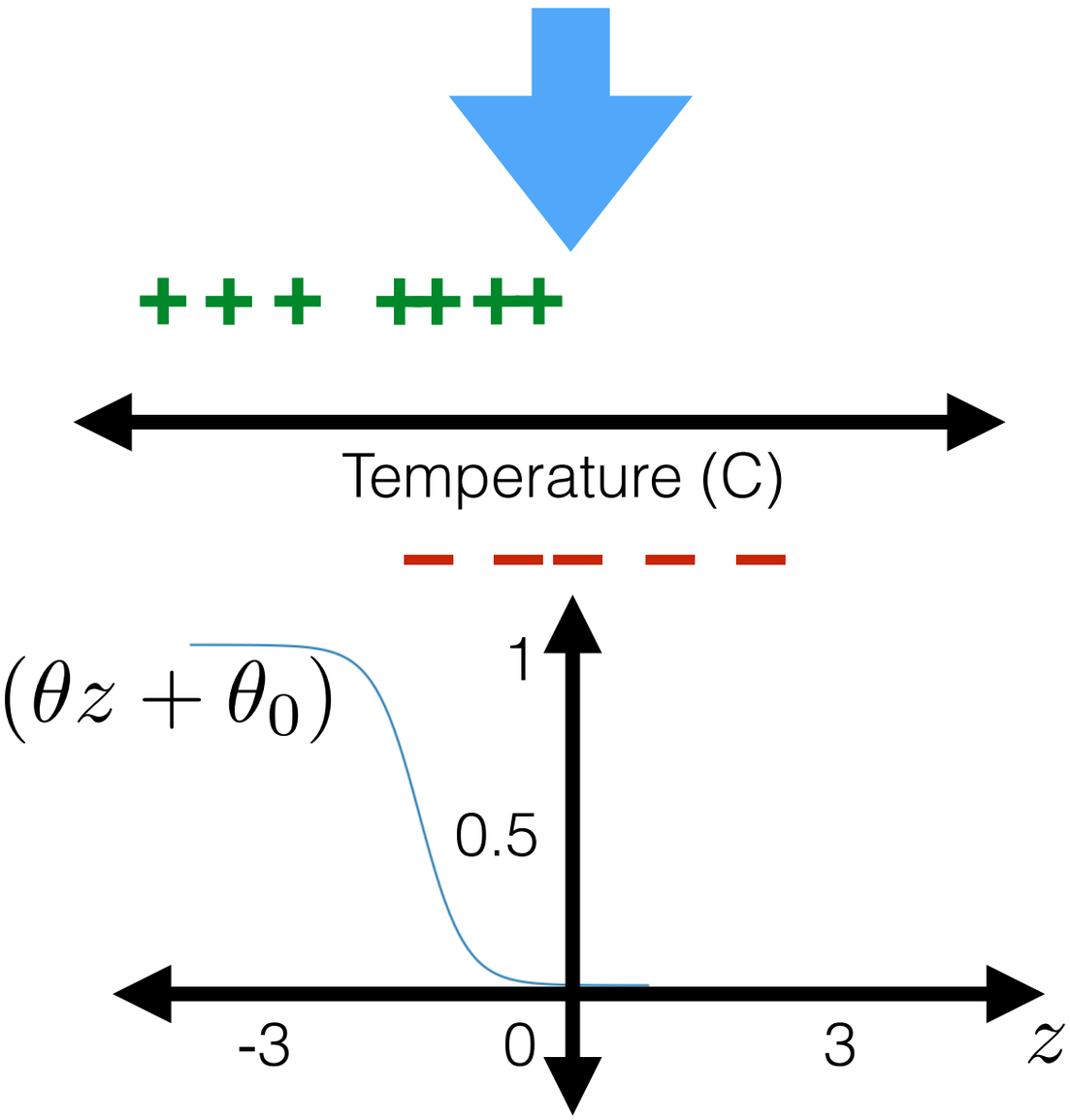
$$\sigma(z) = \frac{1}{1 + \exp(-z)}$$



Capturing uncertainty

$$g(x) = \sigma(\theta x + \theta_0)$$
$$= \frac{1}{1 + \exp\{-(\theta x + \theta_0)\}}$$





- How to make this shape?
  - Sigmoid/logistic function

$$\sigma(z) = \frac{1}{1 + \exp(-z)}$$

Capturing uncertainty

Capturing uncertainty

1 feature:

Capturing uncertainty

1 feature:

$$g(x) = \sigma(\theta x + \theta_0)$$

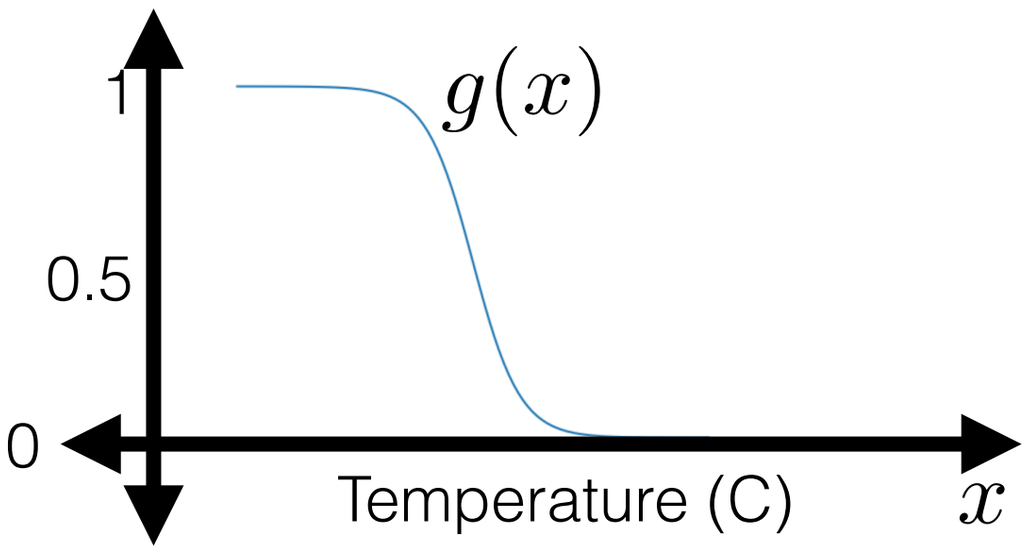
$$= \frac{1}{1 + \exp\{-(\theta x + \theta_0)\}}$$

Capturing uncertainty

1 feature:

$$g(x) = \sigma(\theta x + \theta_0)$$

$$= \frac{1}{1 + \exp\{-(\theta x + \theta_0)\}}$$

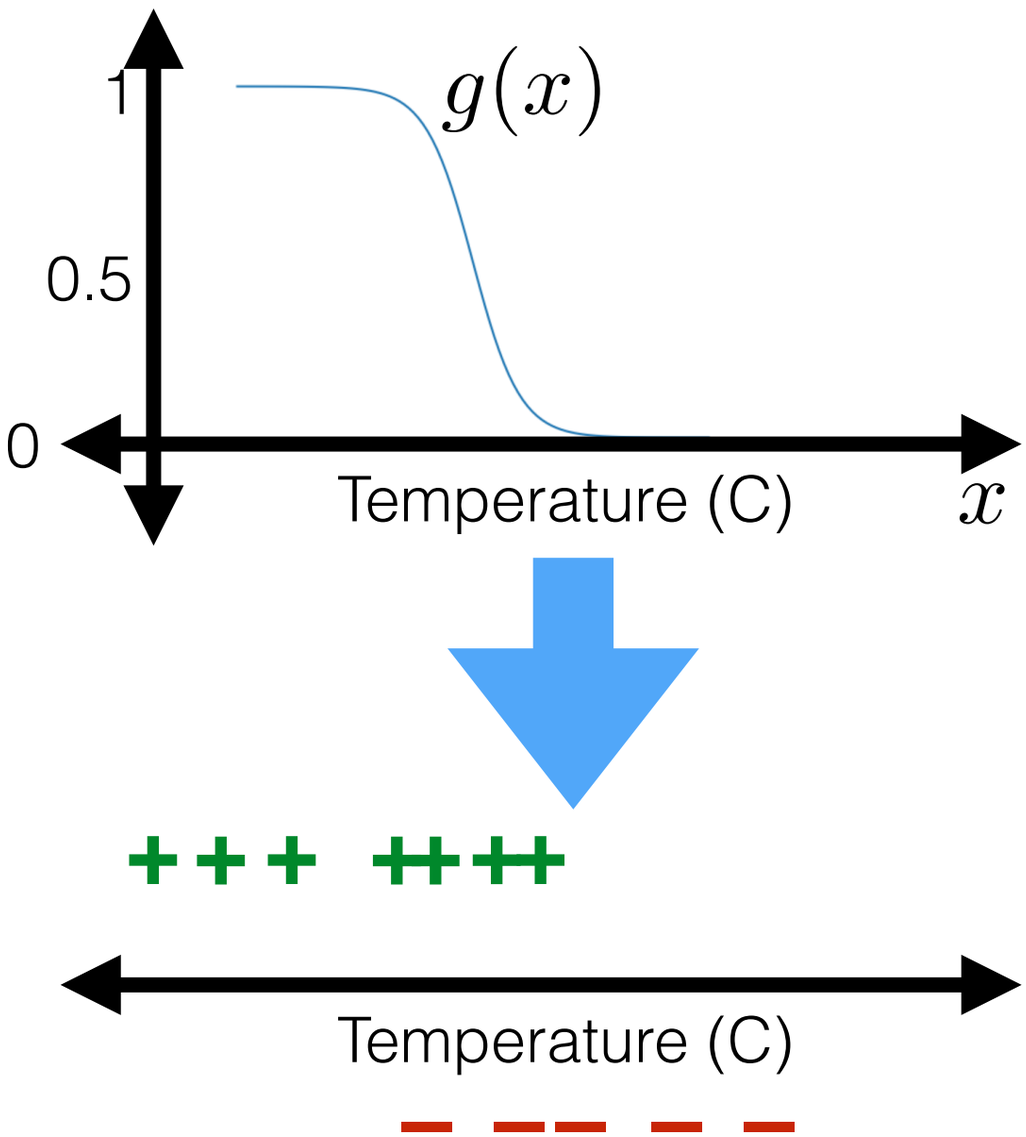


Capturing uncertainty

1 feature:

$$g(x) = \sigma(\theta x + \theta_0)$$

$$= \frac{1}{1 + \exp\{-(\theta x + \theta_0)\}}$$



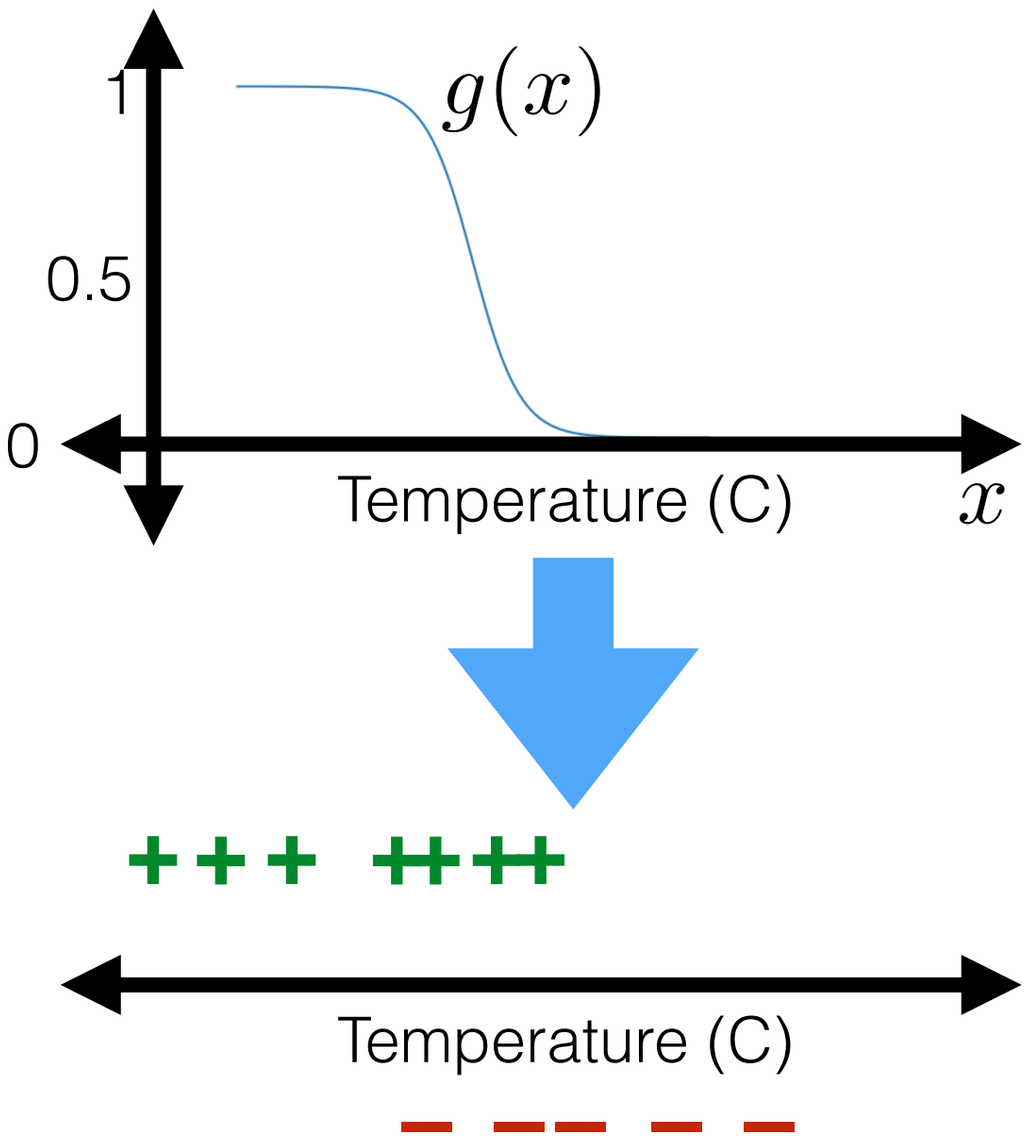
Capturing uncertainty

2 features:

1 feature:

$$g(x) = \sigma(\theta x + \theta_0)$$

$$= \frac{1}{1 + \exp\{-(\theta x + \theta_0)\}}$$

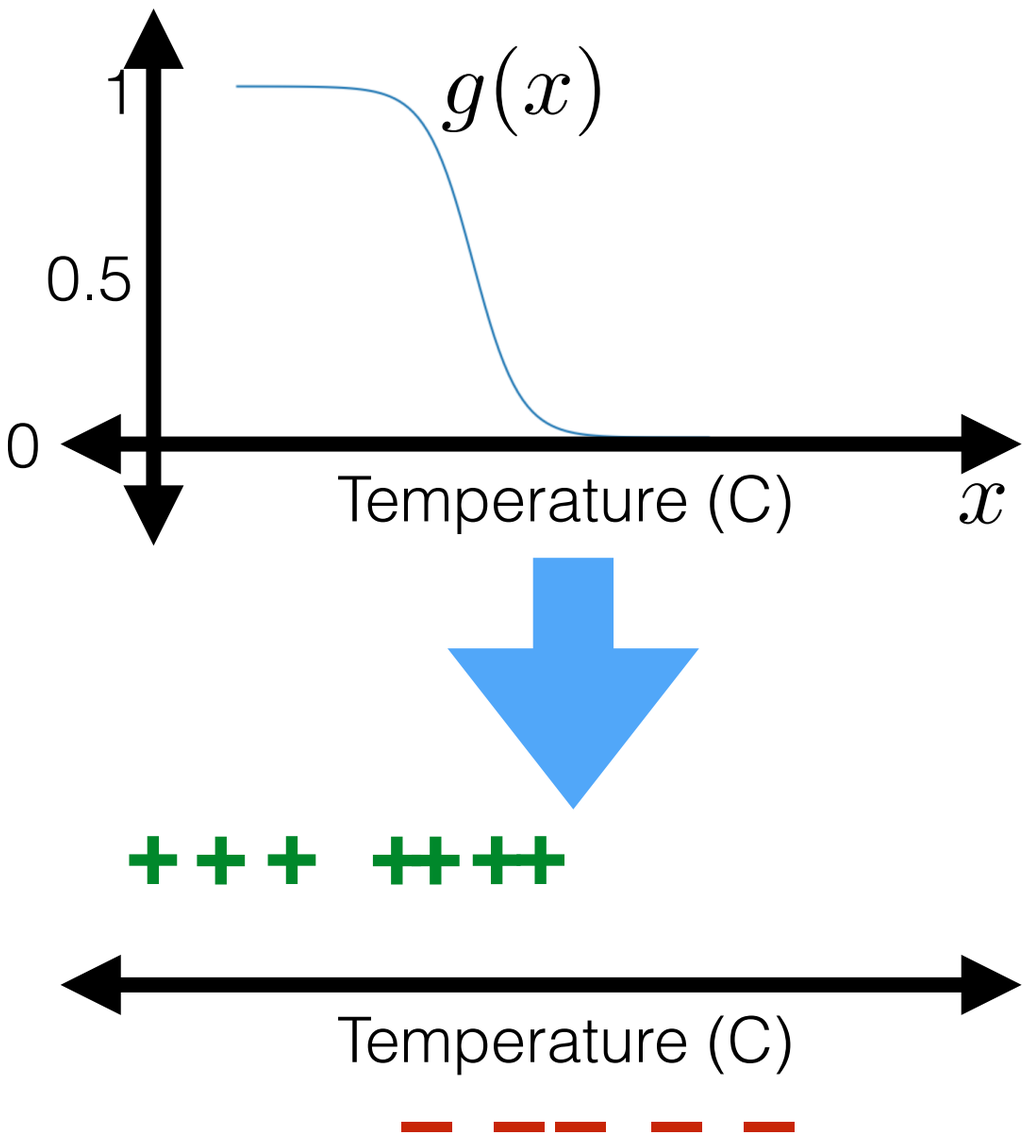


Capturing uncertainty

1 feature:

$$g(x) = \sigma(\theta x + \theta_0)$$

$$= \frac{1}{1 + \exp\{-(\theta x + \theta_0)\}}$$



2 features:

$$g(x) = \sigma(\theta^\top x + \theta_0)$$

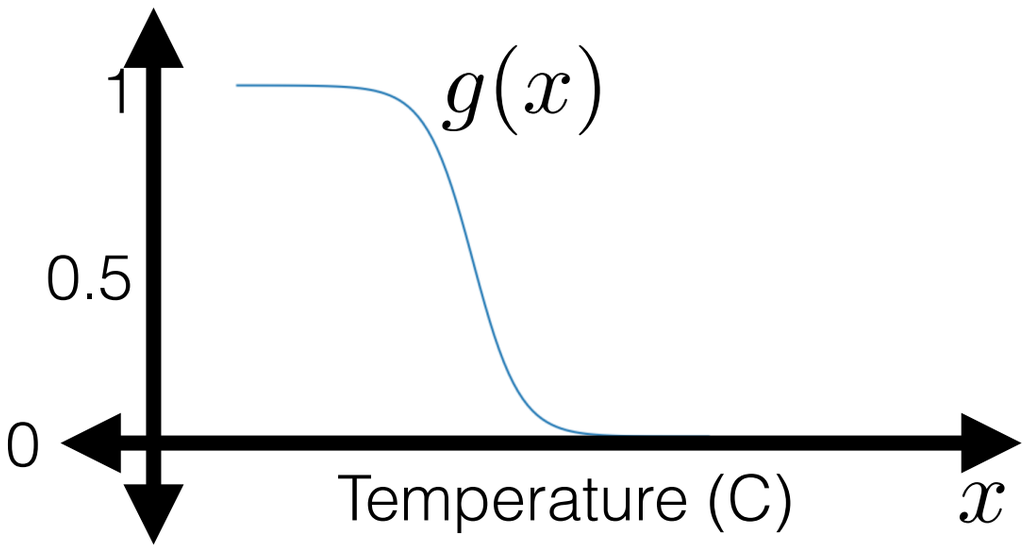
$$= \frac{1}{1 + \exp\{-(\theta^\top x + \theta_0)\}}$$

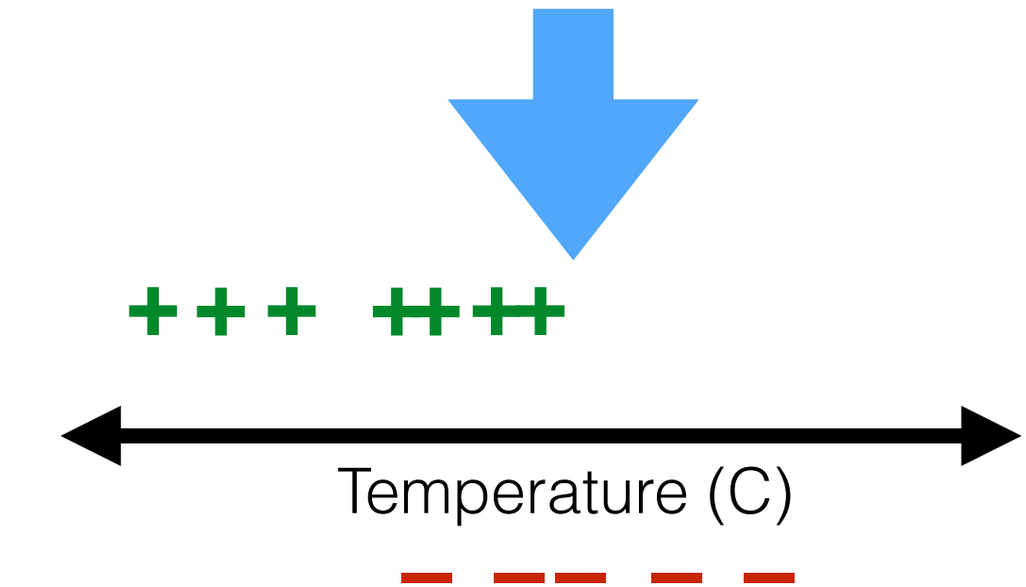
Capturing uncertainty

1 feature:

$$g(x) = \sigma(\theta x + \theta_0)$$

$$= \frac{1}{1 + \exp\{-(\theta x + \theta_0)\}}$$

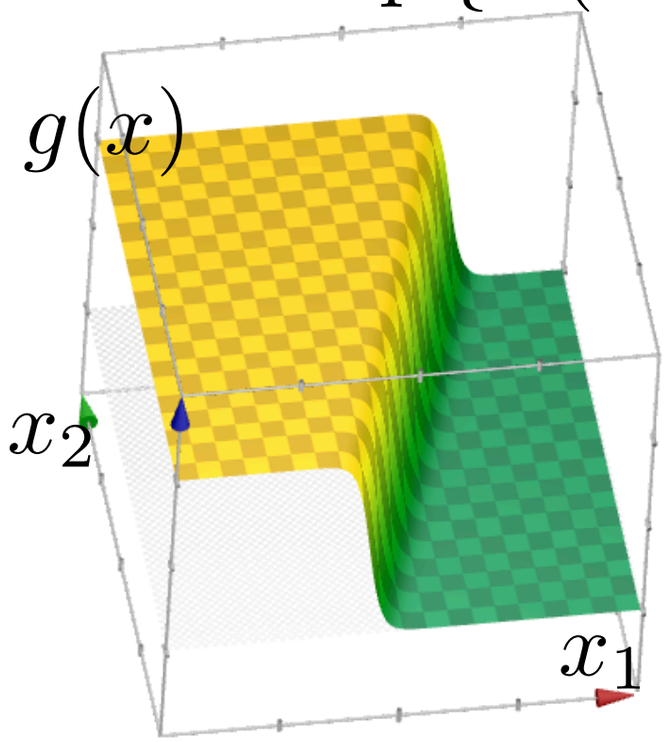




2 features:

$$g(x) = \sigma(\theta^\top x + \theta_0)$$

$$= \frac{1}{1 + \exp\{-(\theta^\top x + \theta_0)\}}$$

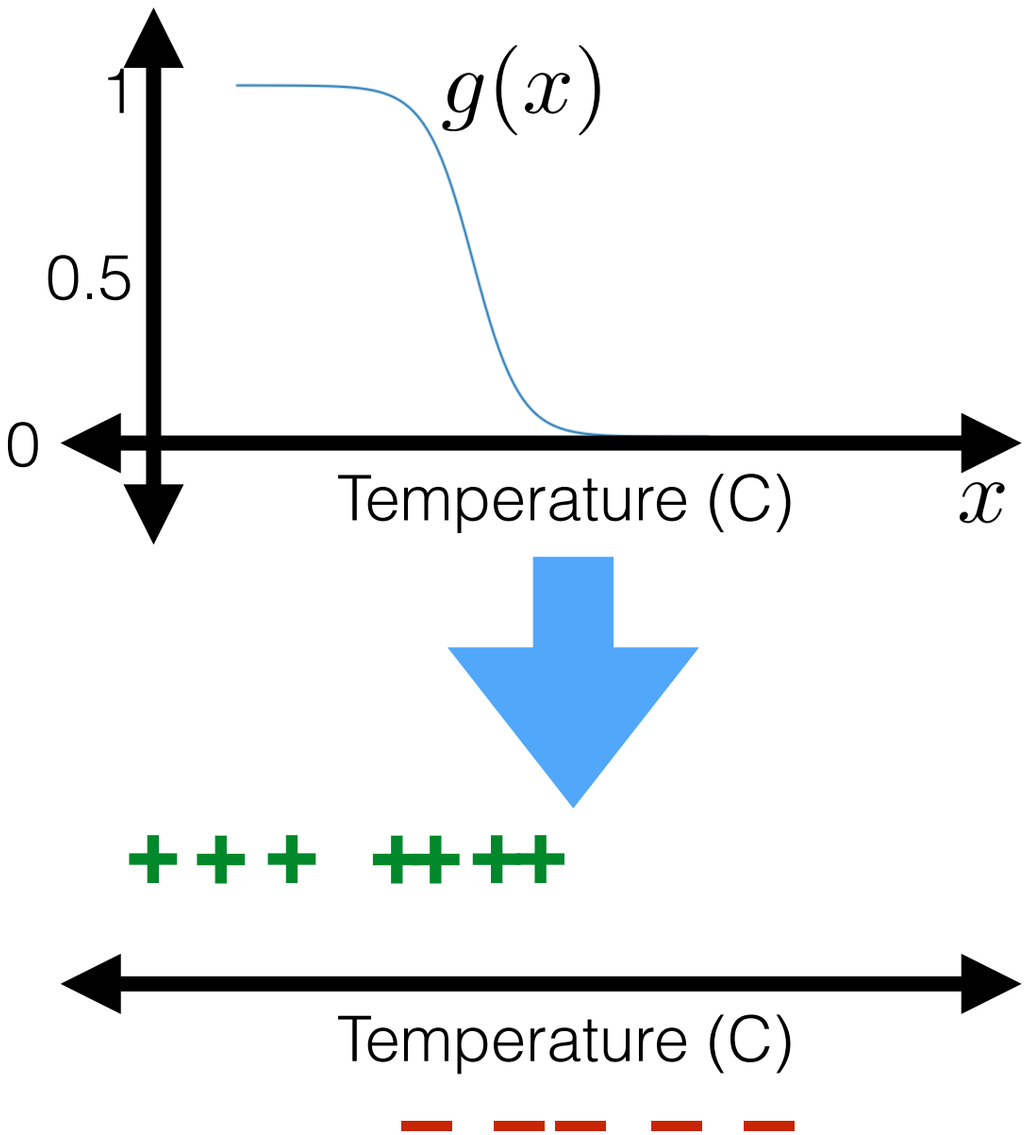


Capturing uncertainty

1 feature:

$$g(x) = \sigma(\theta x + \theta_0)$$

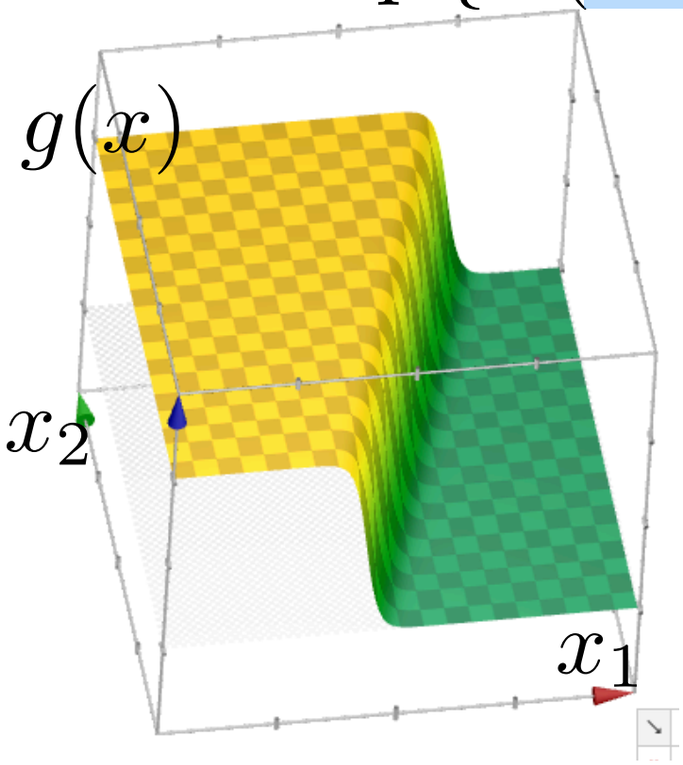
$$= \frac{1}{1 + \exp\{-(\theta x + \theta_0)\}}$$



2 features:

$$g(x) = \sigma(\theta^\top x + \theta_0)$$

$$= \frac{1}{1 + \exp\{-(\theta^\top x + \theta_0)\}}$$

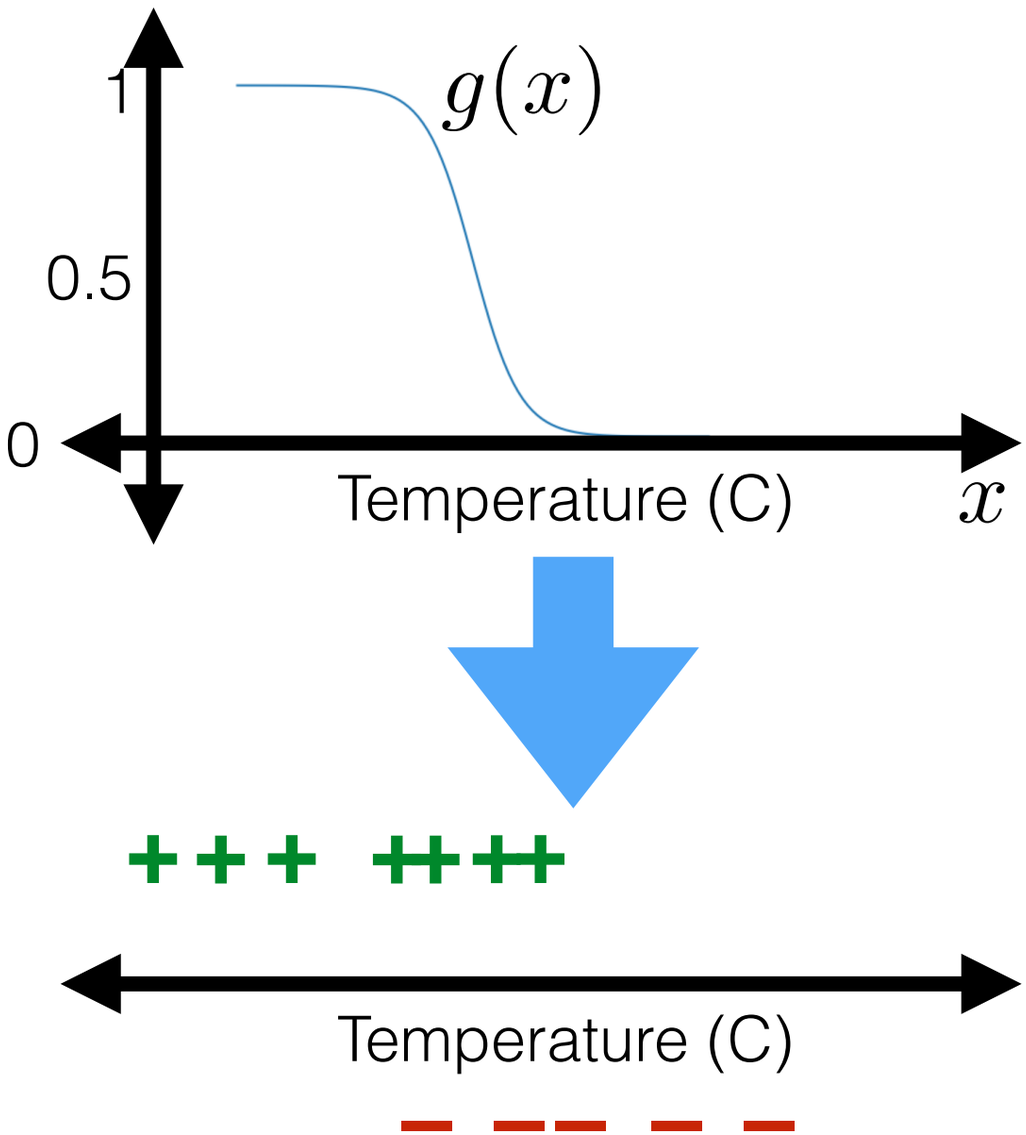


Capturing uncertainty

1 feature:

$$g(x) = \sigma(\theta x + \theta_0)$$

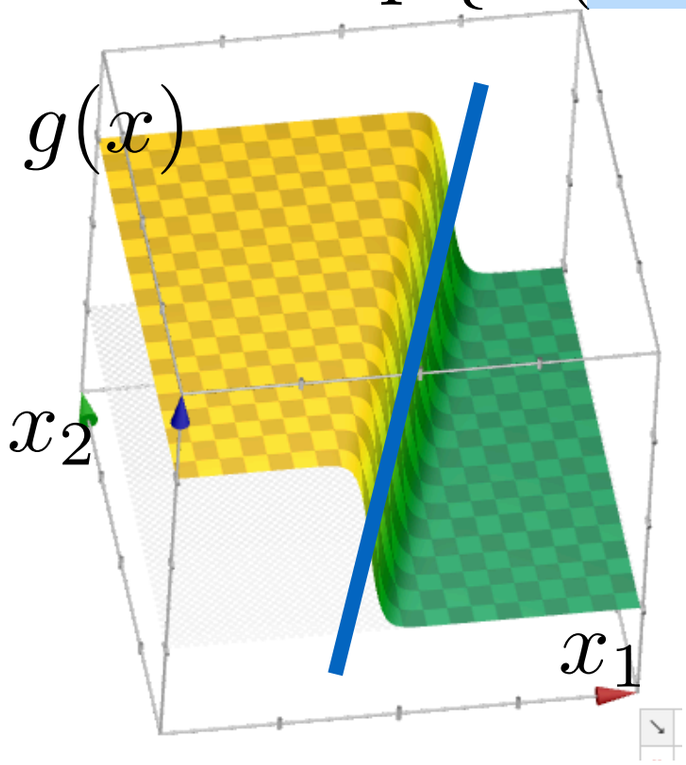
$$= \frac{1}{1 + \exp\{-(\theta x + \theta_0)\}}$$



2 features:

$$g(x) = \sigma(\theta^\top x + \theta_0)$$

$$= \frac{1}{1 + \exp\{-(\theta^\top x + \theta_0)\}}$$

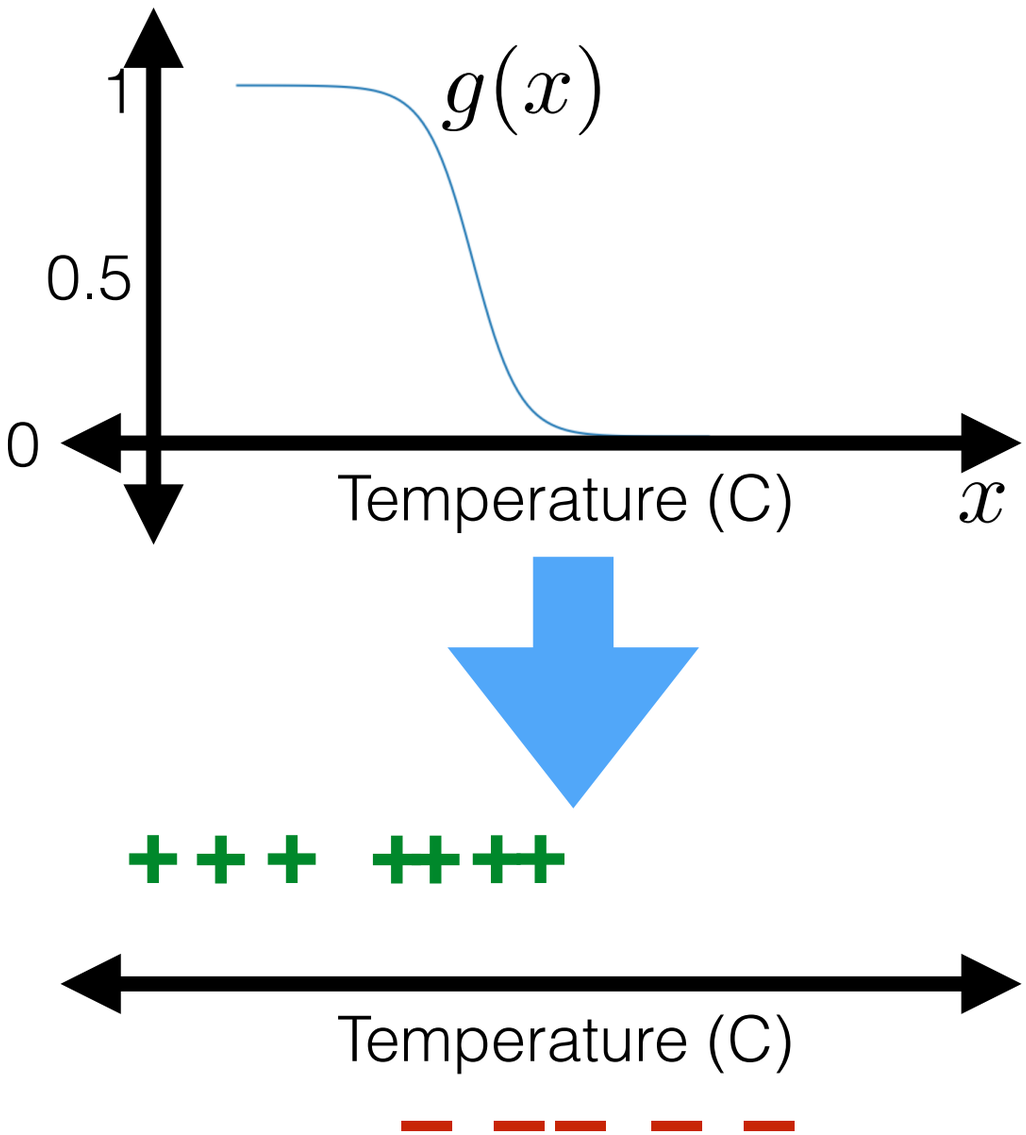


Capturing uncertainty

1 feature:

$$g(x) = \sigma(\theta x + \theta_0)$$

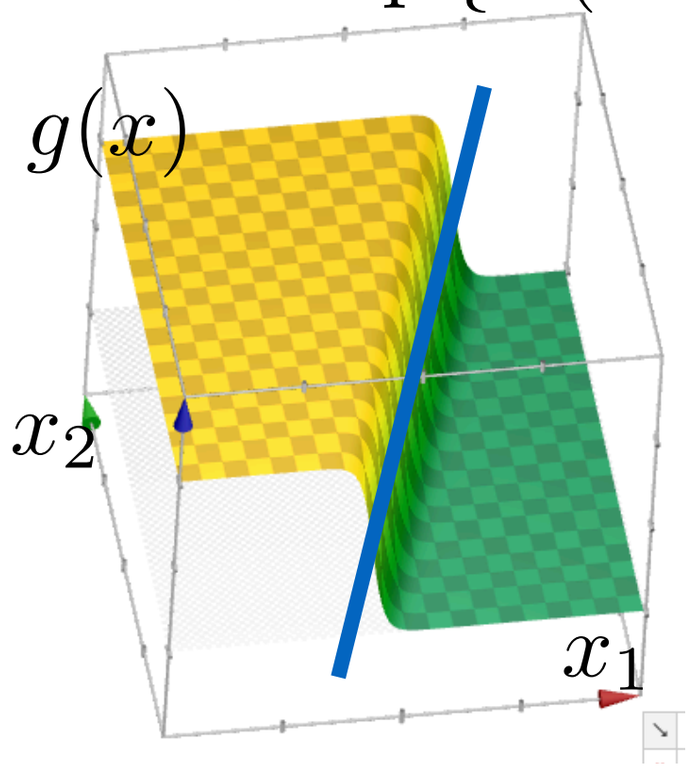
$$= \frac{1}{1 + \exp\{-(\theta x + \theta_0)\}}$$



2 features:

$$g(x) = \sigma(\theta^\top x + \theta_0)$$

$$= \frac{1}{1 + \exp\{-(\theta^\top x + \theta_0)\}}$$

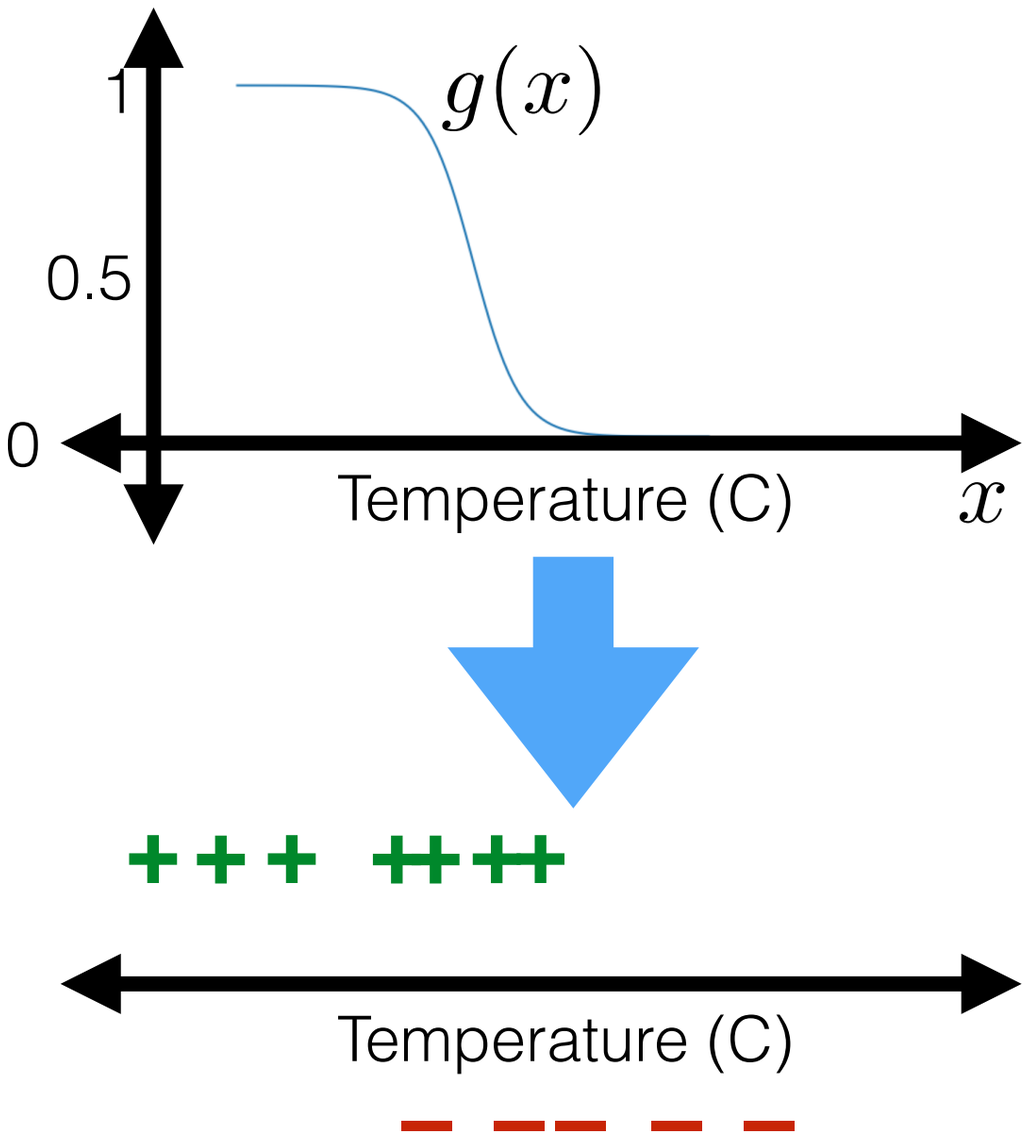


Capturing uncertainty

1 feature:

$$g(x) = \sigma(\theta x + \theta_0)$$

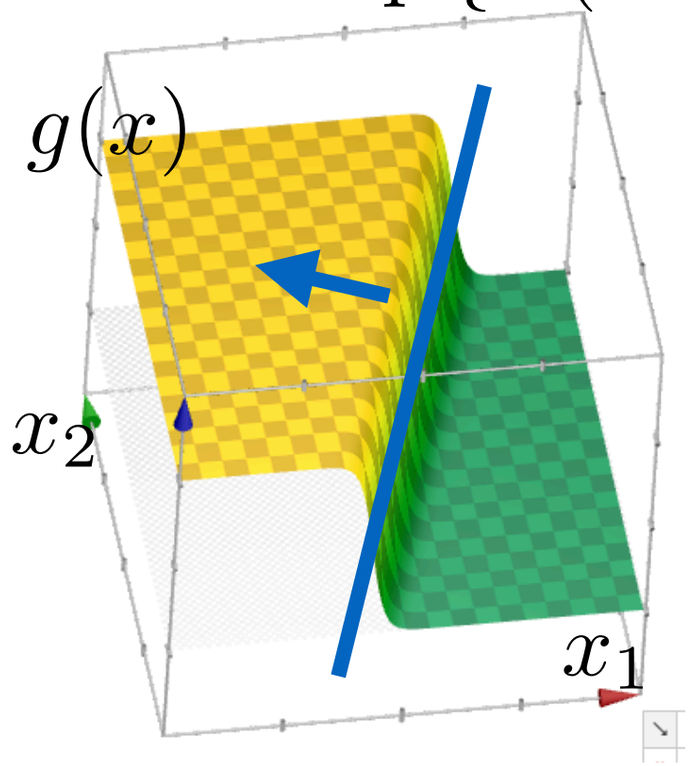
$$= \frac{1}{1 + \exp\{-(\theta x + \theta_0)\}}$$



2 features:

$$g(x) = \sigma(\theta^\top x + \theta_0)$$

$$= \frac{1}{1 + \exp\{-(\theta^\top x + \theta_0)\}}$$

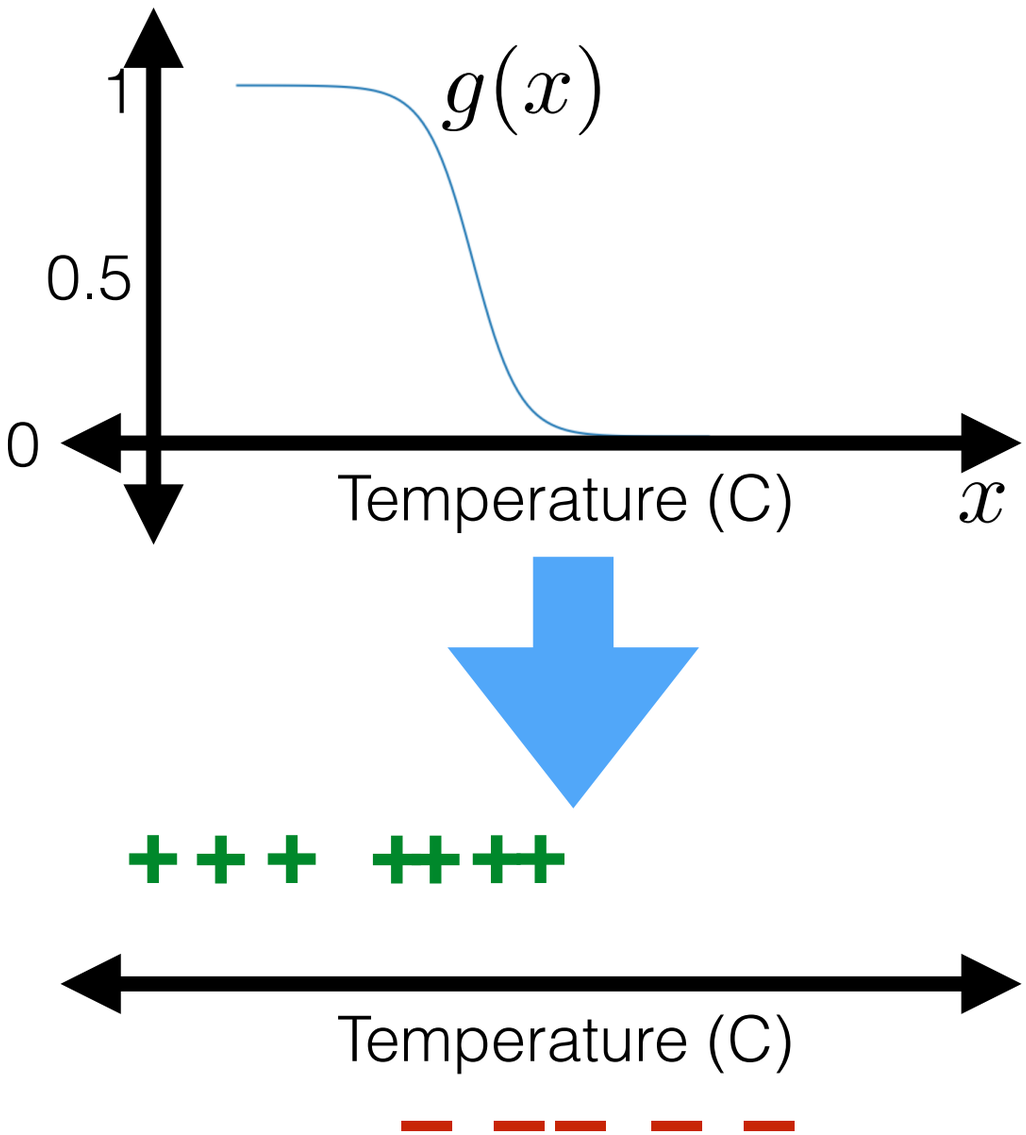


Capturing uncertainty

1 feature:

$$g(x) = \sigma(\theta x + \theta_0)$$

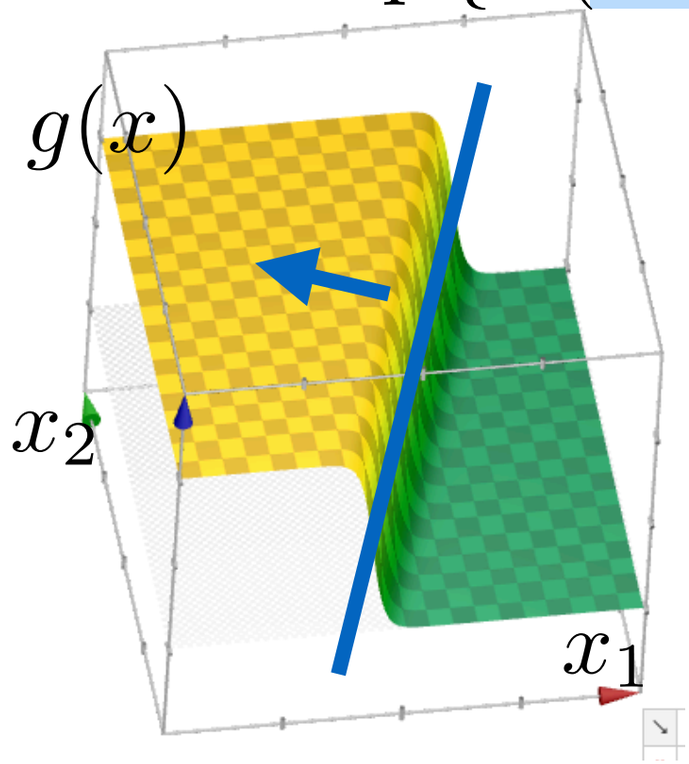
$$= \frac{1}{1 + \exp\{-(\theta x + \theta_0)\}}$$



2 features:

$$g(x) = \sigma(\theta^\top x + \theta_0)$$

$$= \frac{1}{1 + \exp\{-(\theta^\top x + \theta_0)\}}$$

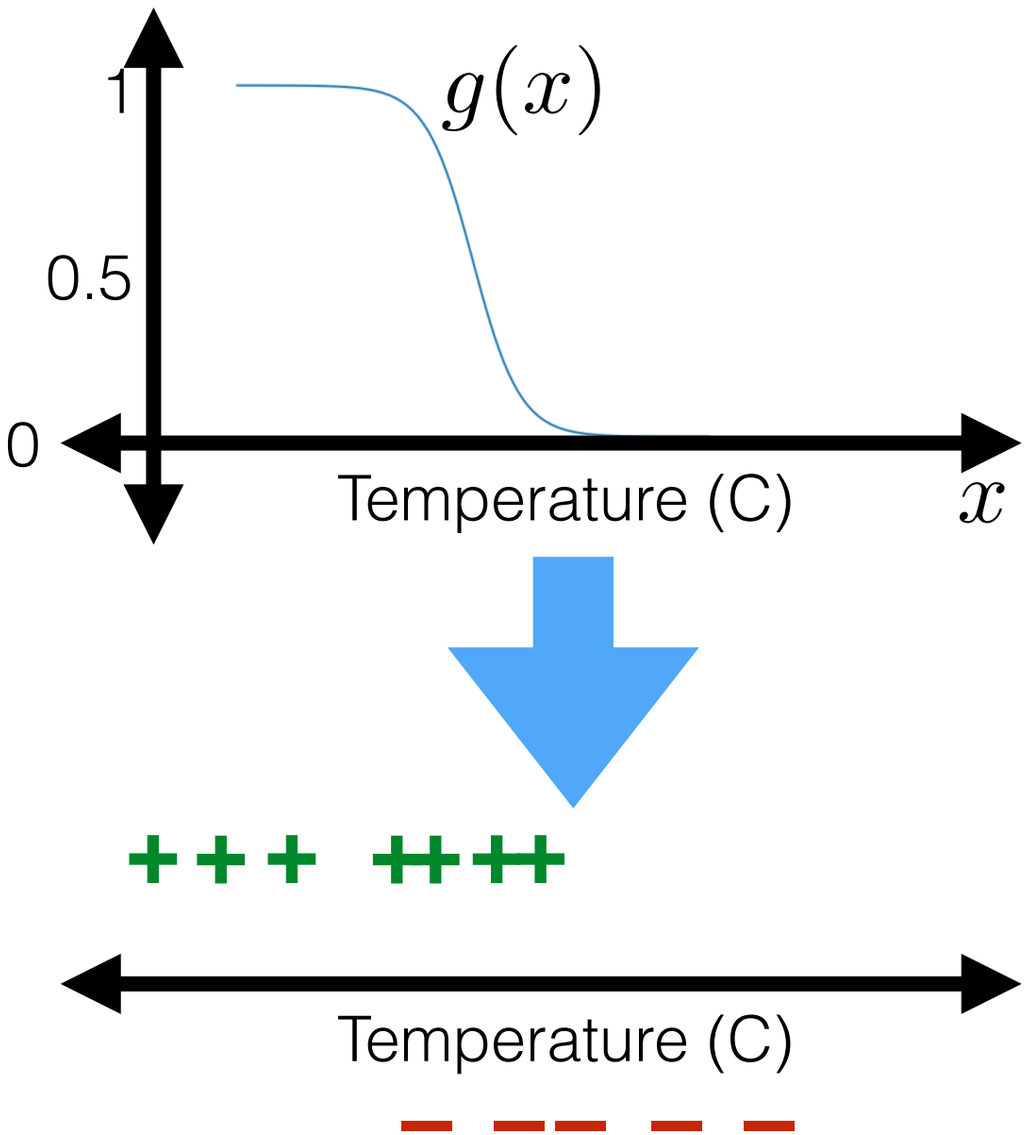


Capturing uncertainty

1 feature:

$$g(x) = \sigma(\theta x + \theta_0)$$

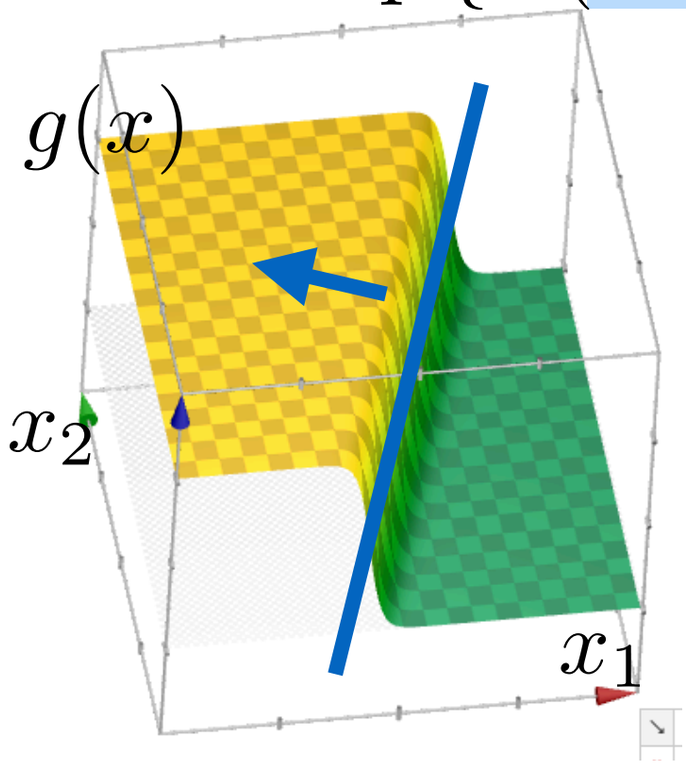
$$= \frac{1}{1 + \exp\{-(\theta x + \theta_0)\}}$$



2 features:

$$g(x) = \sigma(\theta^\top x + \theta_0)$$

$$= \frac{1}{1 + \exp\{-(\theta^\top x + \theta_0)\}}$$

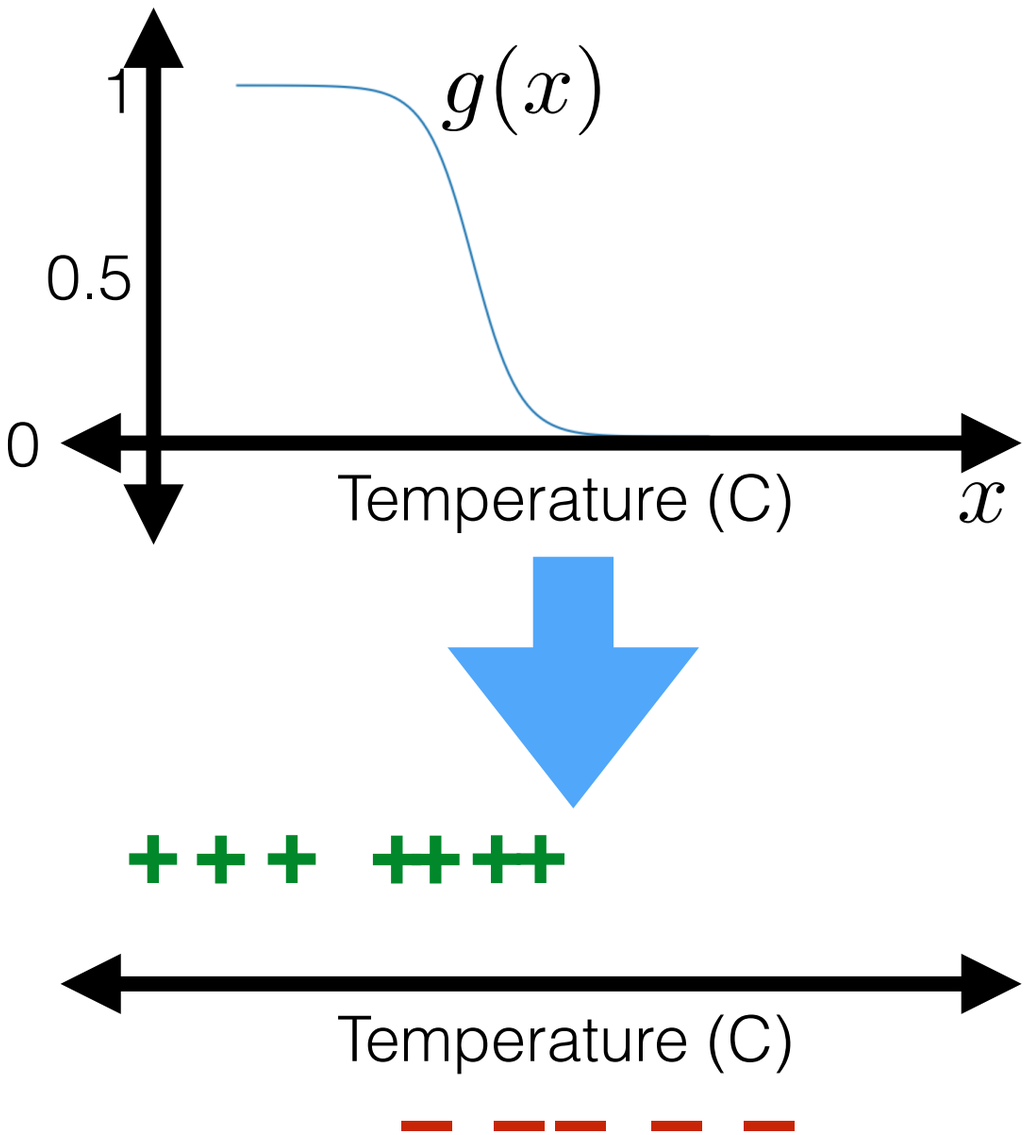


Capturing uncertainty

1 feature:

$$g(x) = \sigma(\theta x + \theta_0)$$

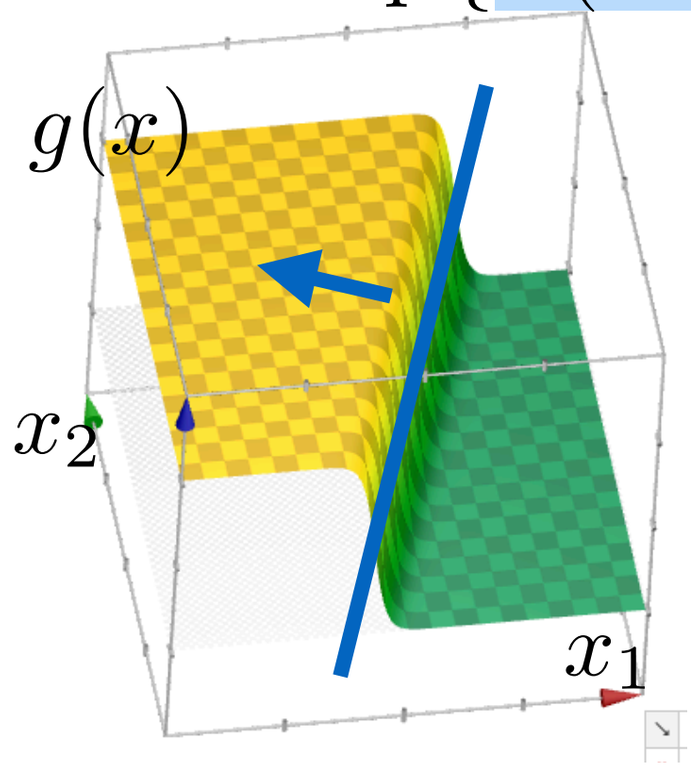
$$= \frac{1}{1 + \exp\{-(\theta x + \theta_0)\}}$$



2 features:

$$g(x) = \sigma(\theta^\top x + \theta_0)$$

$$= \frac{1}{1 + \exp\{-(\theta^\top x + \theta_0)\}}$$

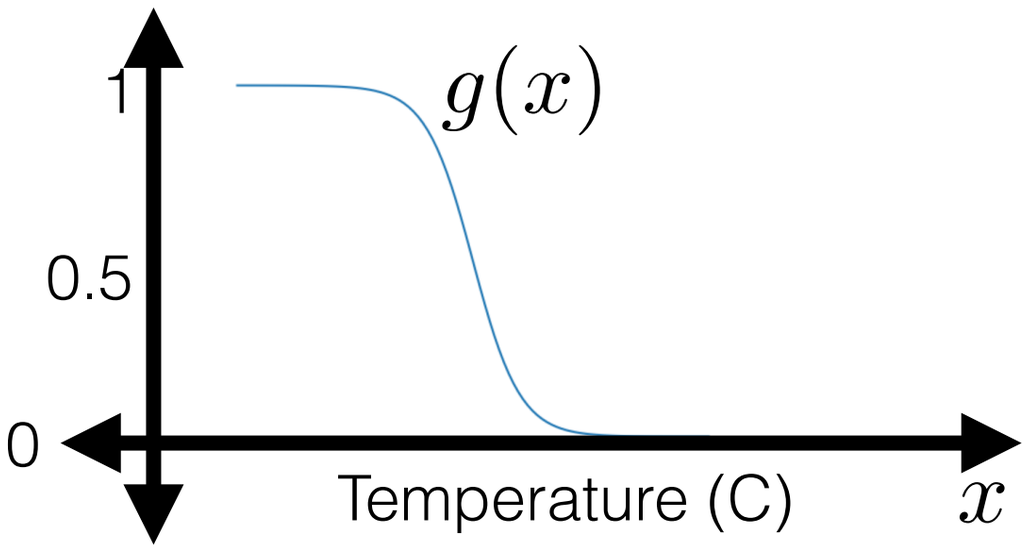


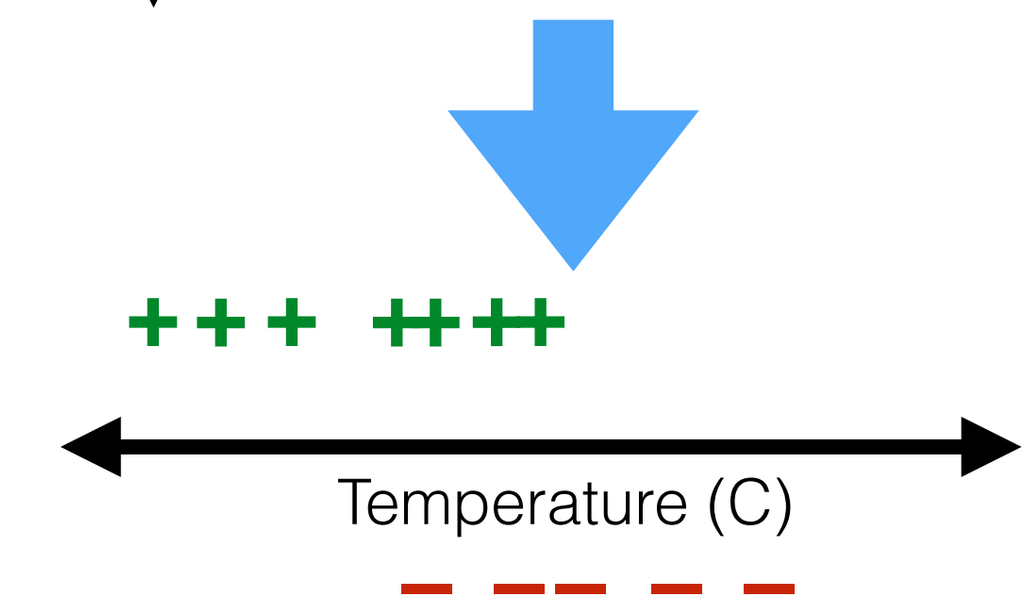
Capturing uncertainty

1 feature:

$$g(x) = \sigma(\theta x + \theta_0)$$

$$= \frac{1}{1 + \exp\{-(\theta x + \theta_0)\}}$$

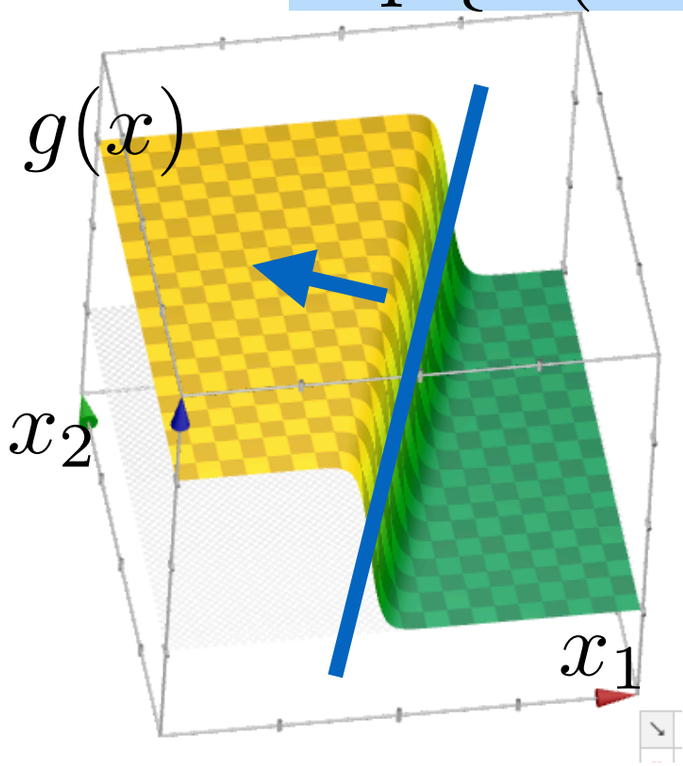




2 features:

$$g(x) = \sigma(\theta^\top x + \theta_0)$$

$$= \frac{1}{1 + \exp\{-(\theta^\top x + \theta_0)\}}$$

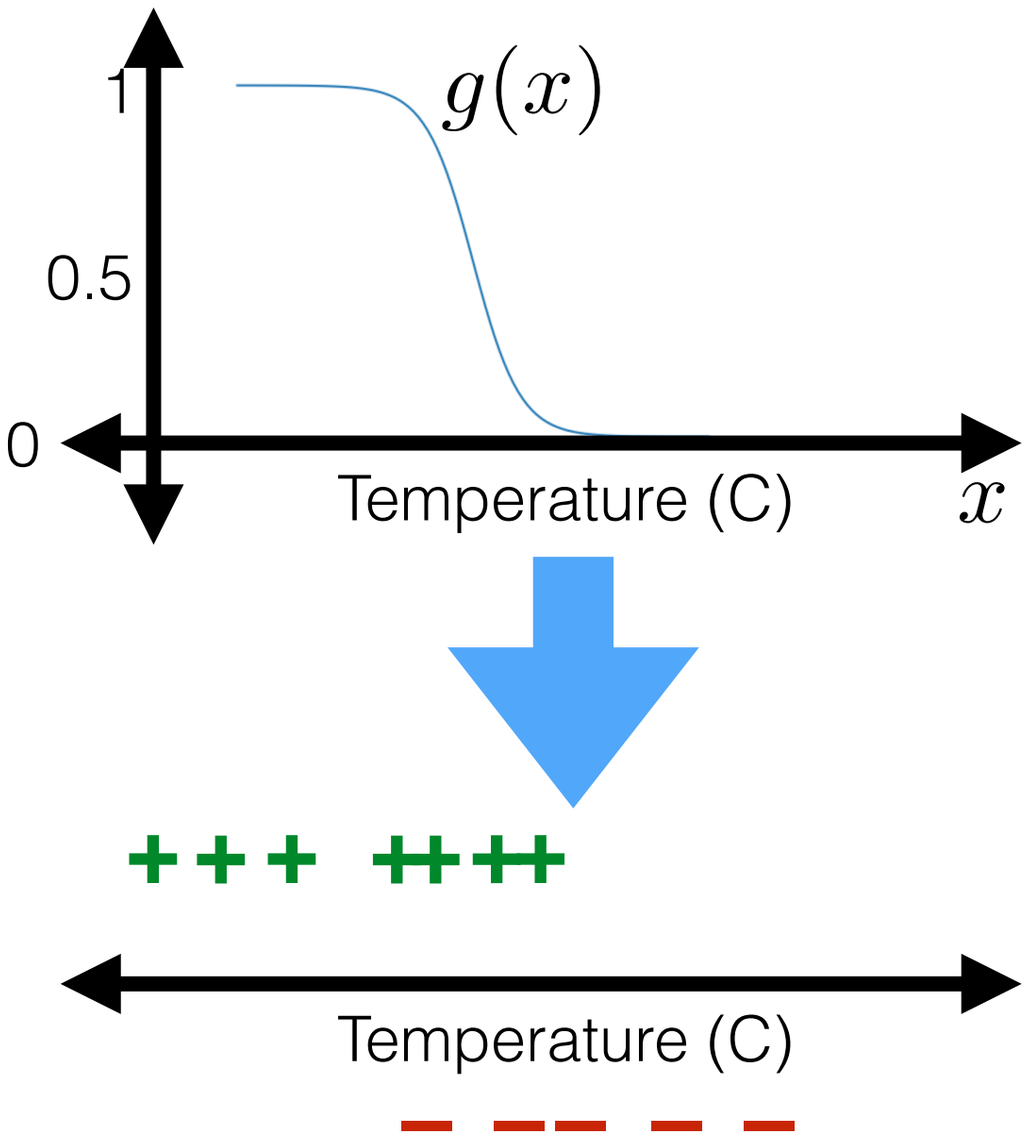


Capturing uncertainty

1 feature:

$$g(x) = \sigma(\theta x + \theta_0)$$

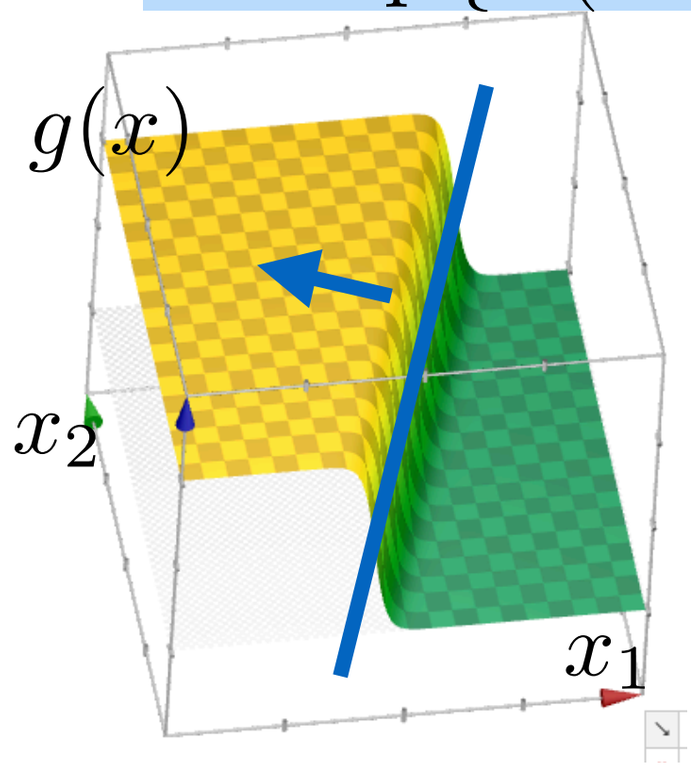
$$= \frac{1}{1 + \exp\{-(\theta x + \theta_0)\}}$$



2 features:

$$g(x) = \sigma(\theta^\top x + \theta_0)$$

$$= \frac{1}{1 + \exp\{-(\theta^\top x + \theta_0)\}}$$

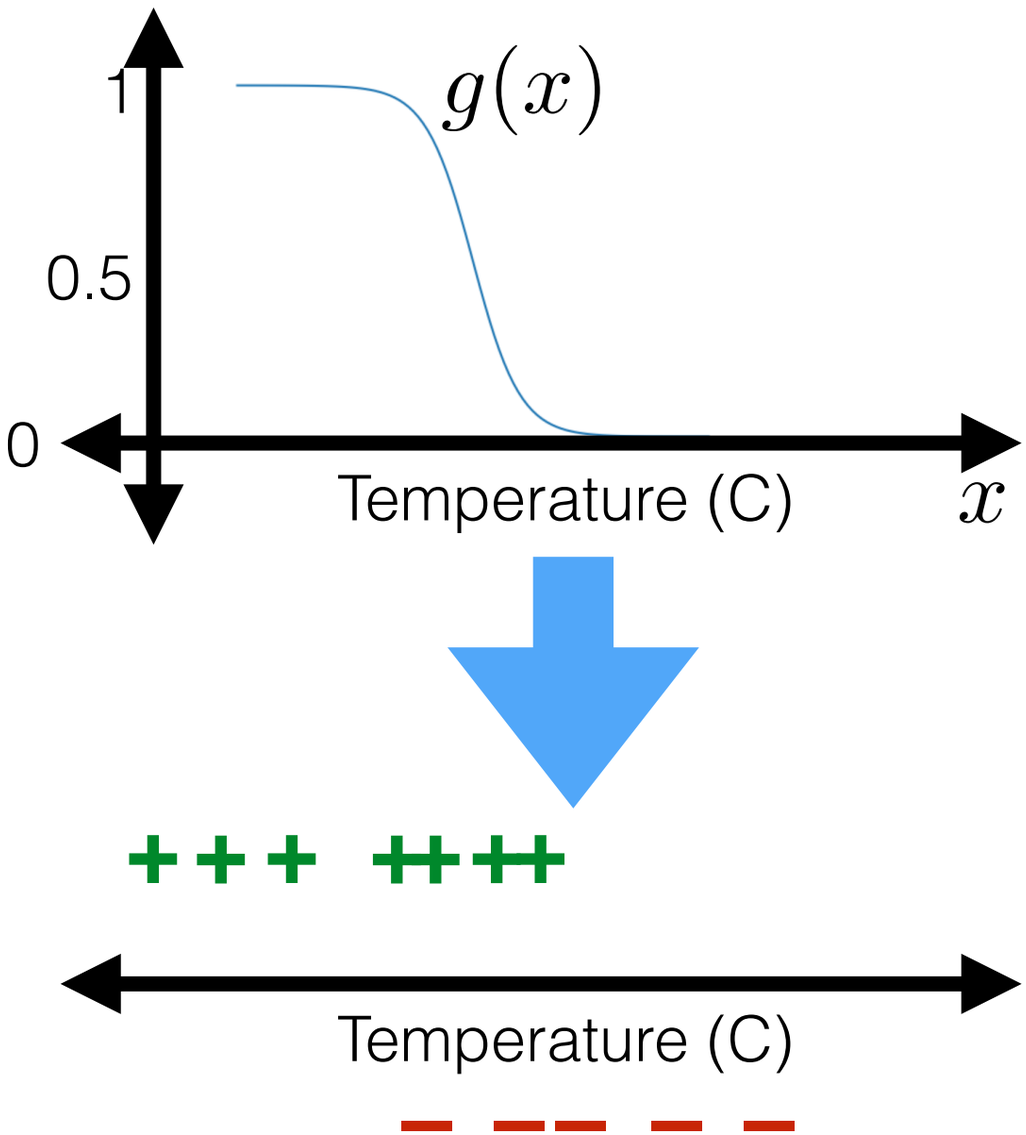


Capturing uncertainty

1 feature:

$$g(x) = \sigma(\theta x + \theta_0)$$

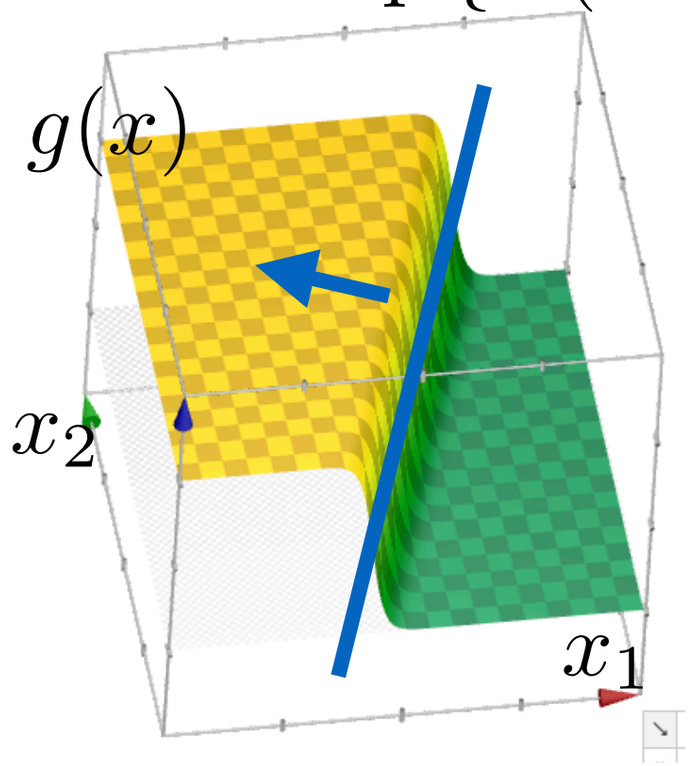
$$= \frac{1}{1 + \exp\{-(\theta x + \theta_0)\}}$$



2 features:

$$g(x) = \sigma(\theta^\top x + \theta_0)$$

$$= \frac{1}{1 + \exp\{-(\theta^\top x + \theta_0)\}}$$

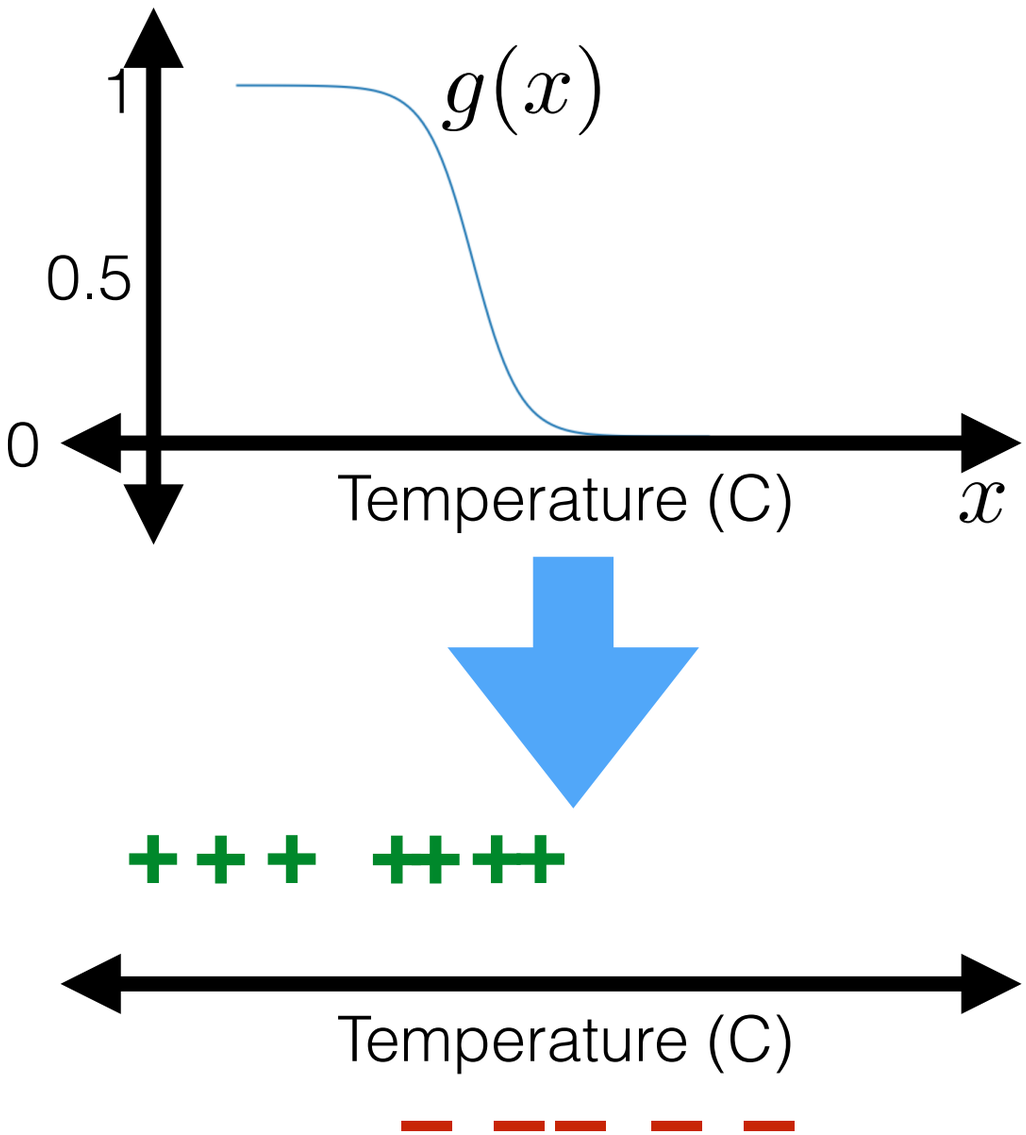


Capturing uncertainty

1 feature:

$$g(x) = \sigma(\theta x + \theta_0)$$

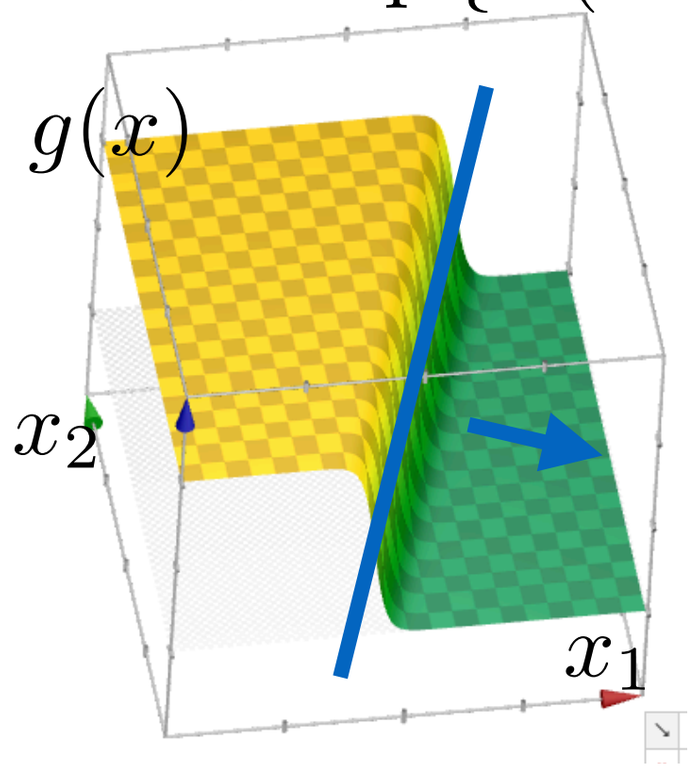
$$= \frac{1}{1 + \exp\{-(\theta x + \theta_0)\}}$$



2 features:

$$g(x) = \sigma(\theta^\top x + \theta_0)$$

$$= \frac{1}{1 + \exp\{-(\theta^\top x + \theta_0)\}}$$

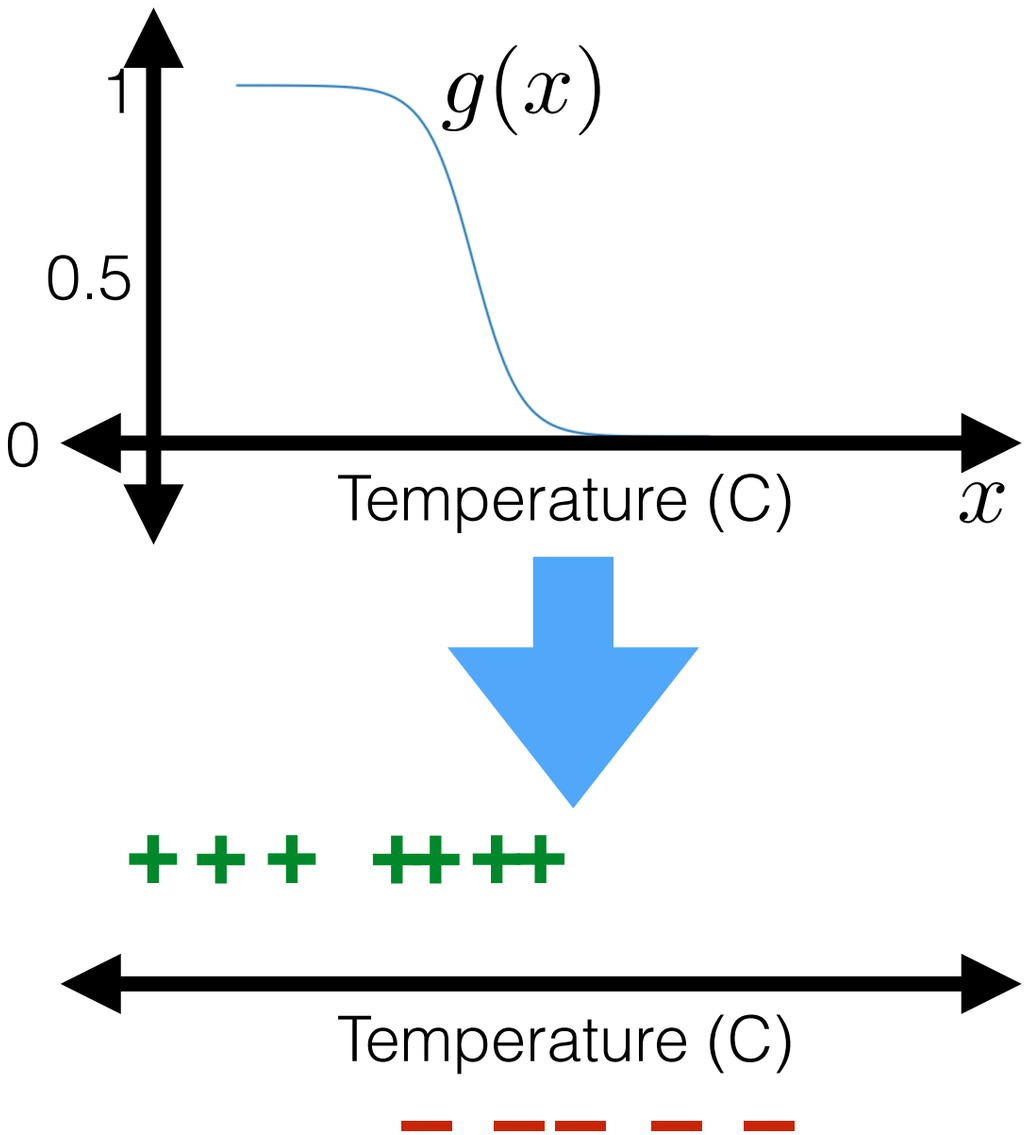


Capturing uncertainty

1 feature:

$$g(x) = \sigma(\theta x + \theta_0)$$

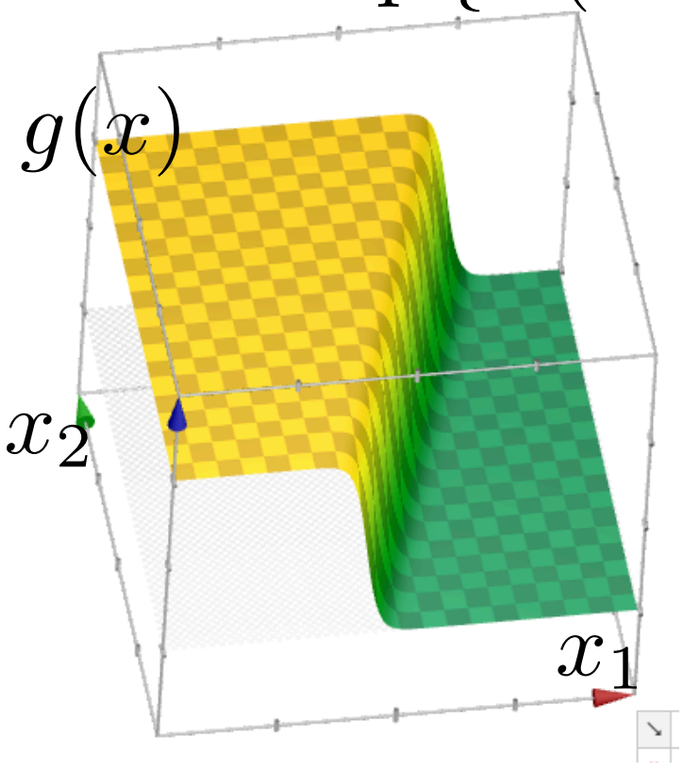
$$= \frac{1}{1 + \exp\{-(\theta x + \theta_0)\}}$$



2 features:

$$g(x) = \sigma(\theta^\top x + \theta_0)$$

$$= \frac{1}{1 + \exp\{-(\theta^\top x + \theta_0)\}}$$

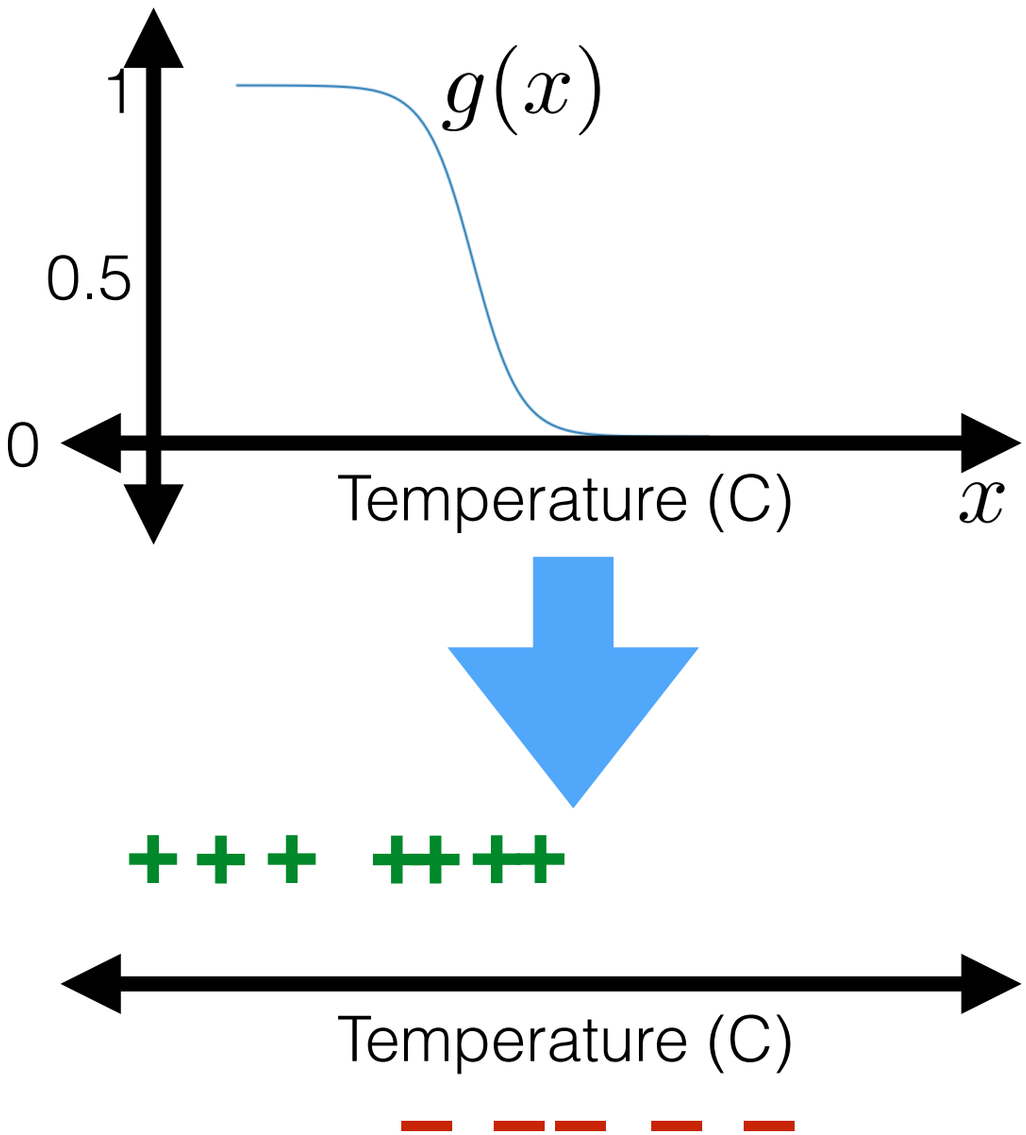


Capturing uncertainty

1 feature:

$$g(x) = \sigma(\theta x + \theta_0)$$

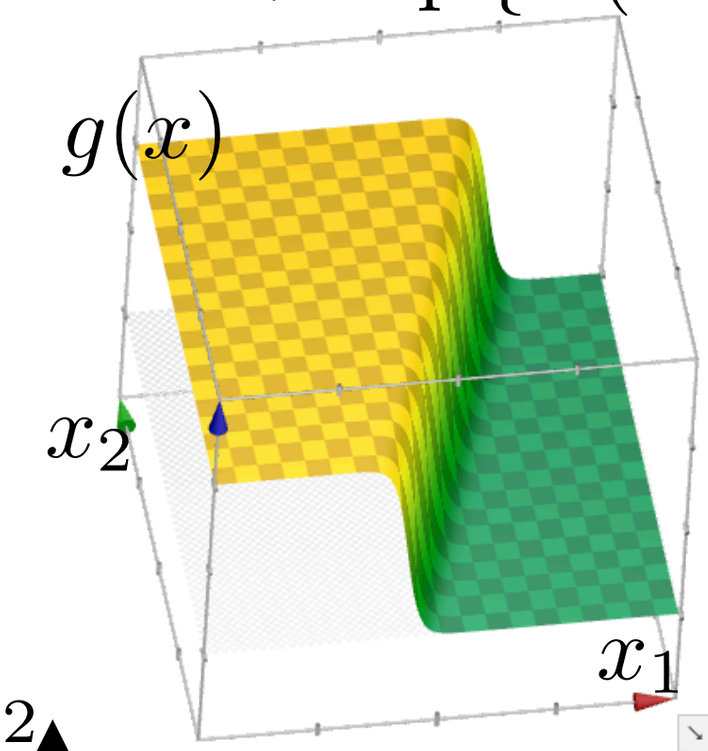
$$= \frac{1}{1 + \exp\{-(\theta x + \theta_0)\}}$$

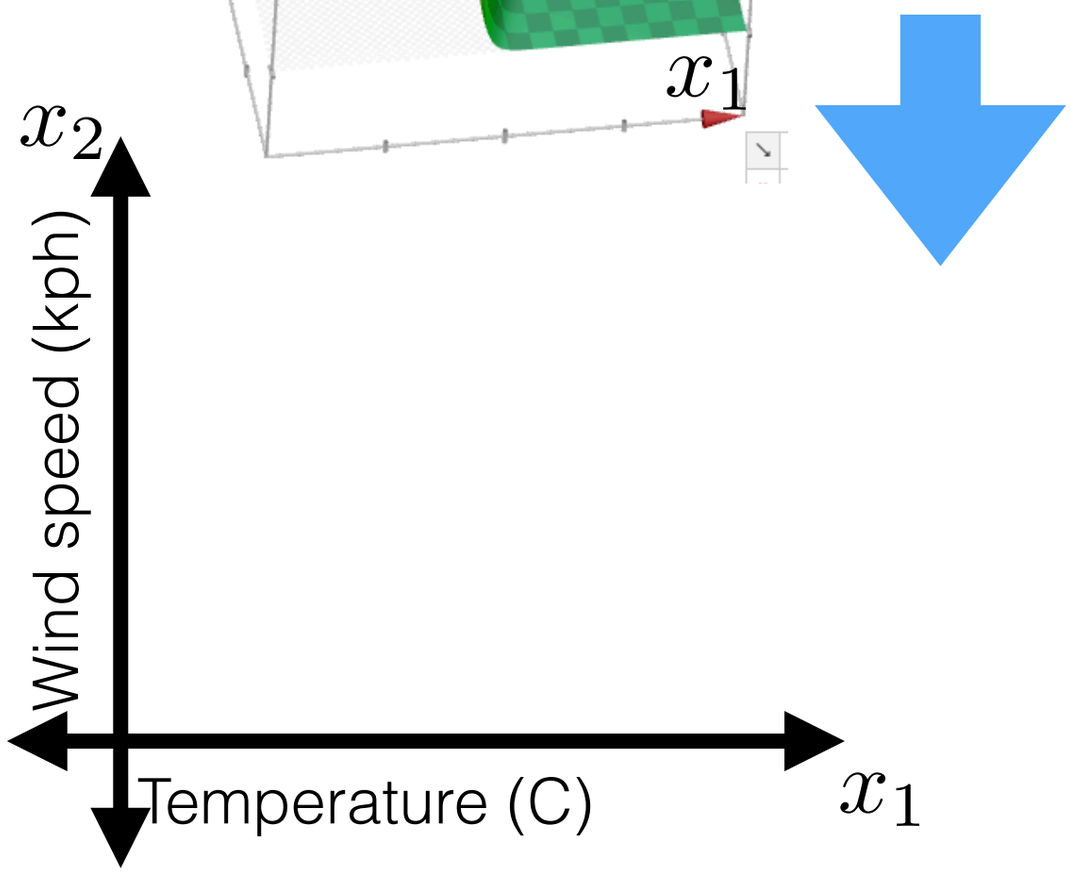


2 features:

$$g(x) = \sigma(\theta^\top x + \theta_0)$$

$$= \frac{1}{1 + \exp\{-(\theta^\top x + \theta_0)\}}$$



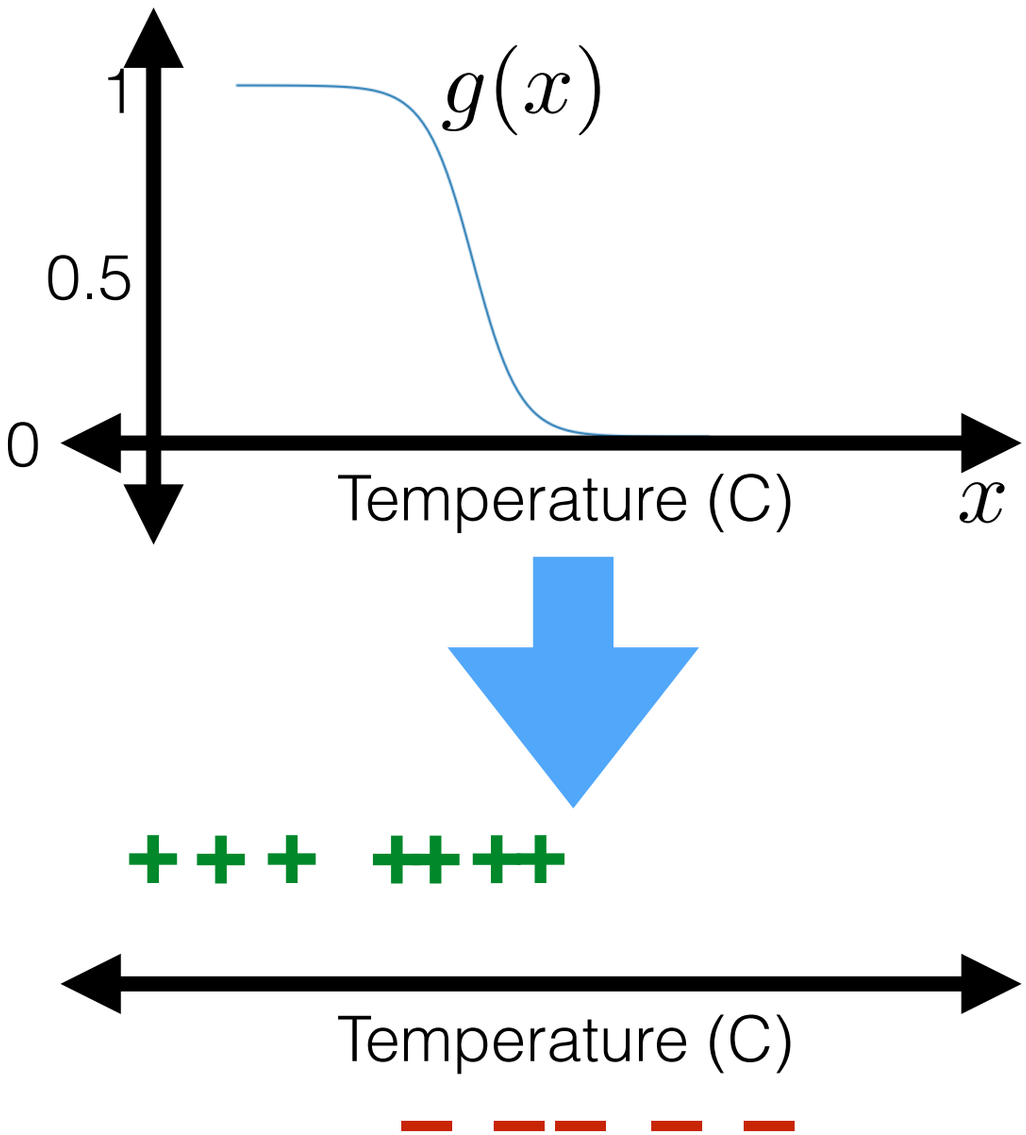


Capturing uncertainty

1 feature:

$$g(x) = \sigma(\theta x + \theta_0)$$

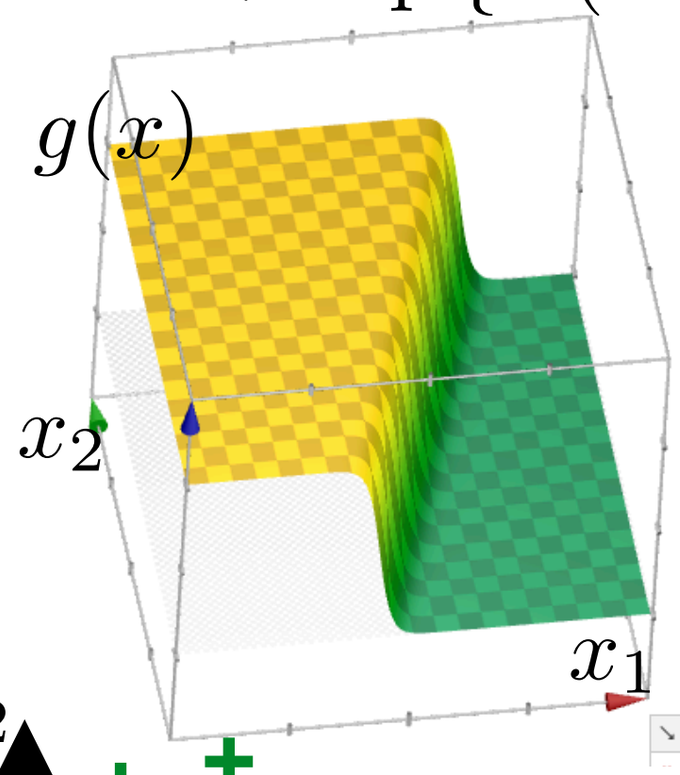
$$= \frac{1}{1 + \exp\{-(\theta x + \theta_0)\}}$$

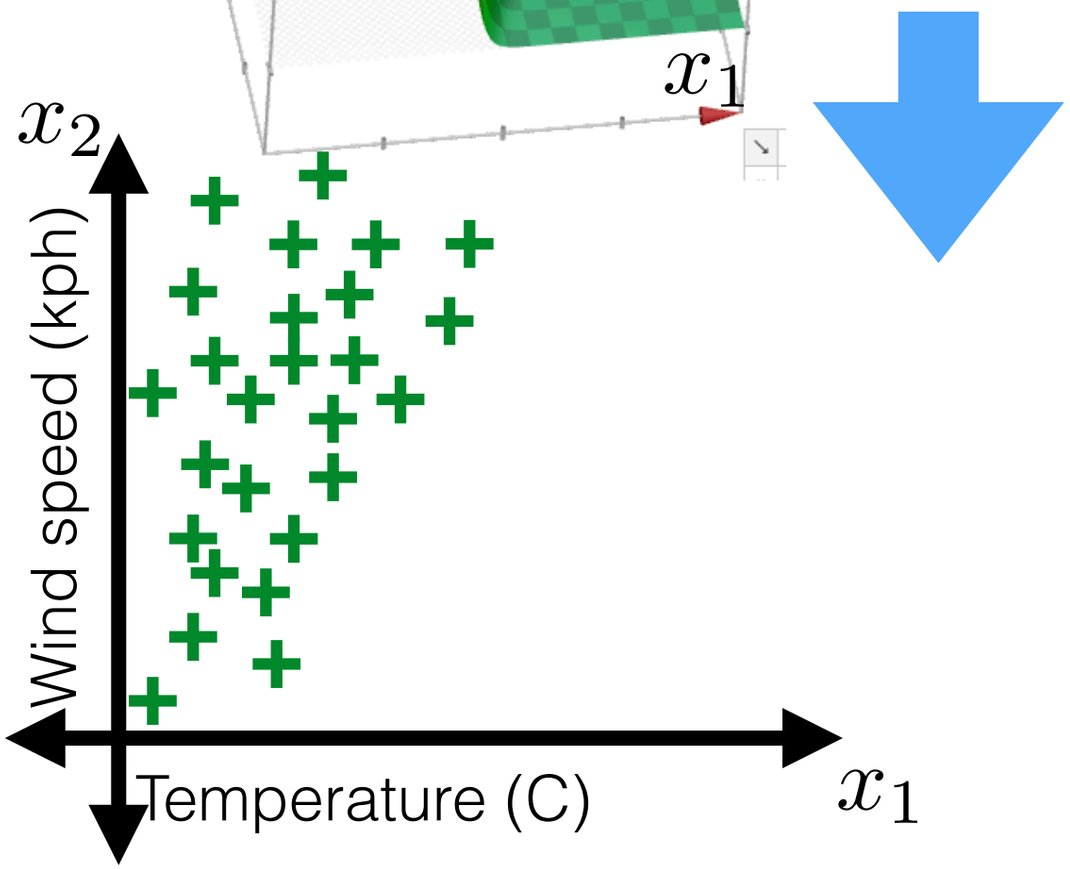


2 features:

$$g(x) = \sigma(\theta^\top x + \theta_0)$$

$$= \frac{1}{1 + \exp\{-(\theta^\top x + \theta_0)\}}$$



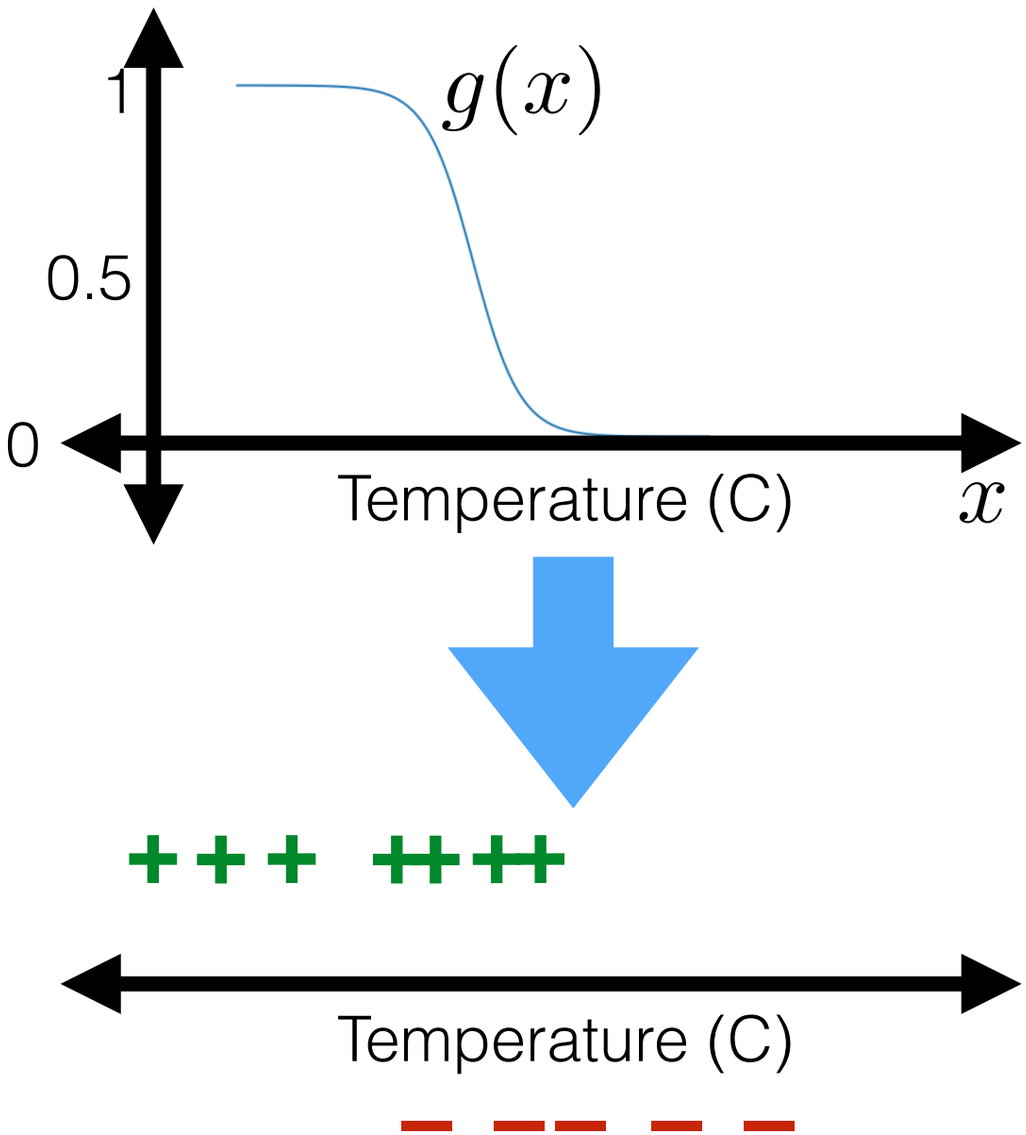


Capturing uncertainty

1 feature:

$$g(x) = \sigma(\theta x + \theta_0)$$

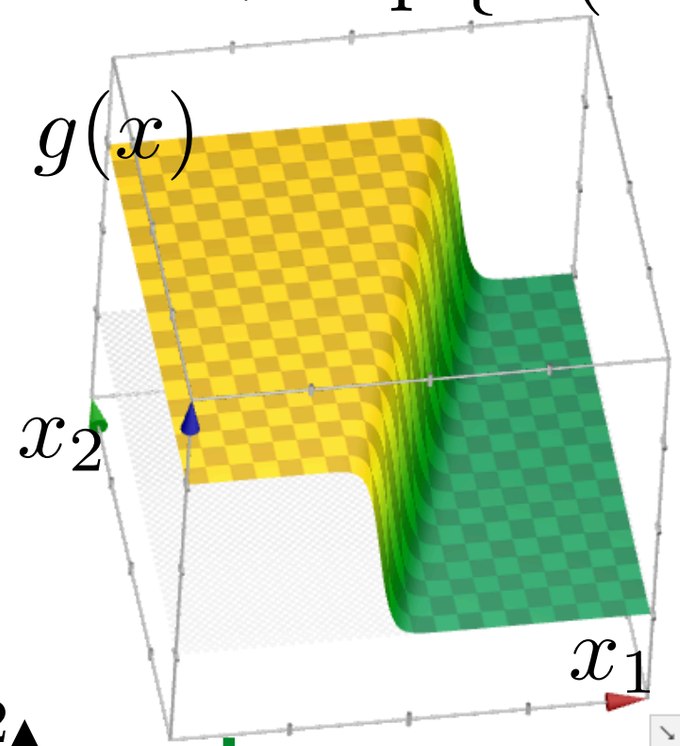
$$= \frac{1}{1 + \exp\{-(\theta x + \theta_0)\}}$$

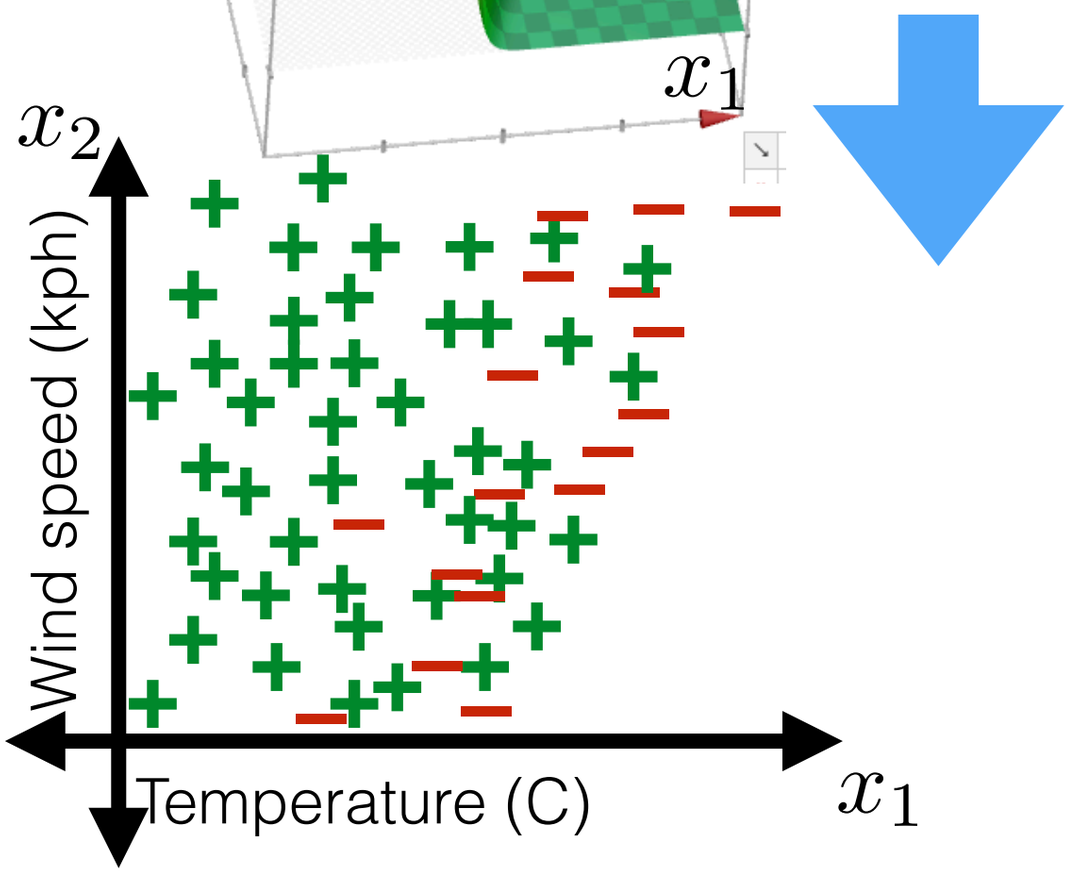


2 features:

$$g(x) = \sigma(\theta^\top x + \theta_0)$$

$$= \frac{1}{1 + \exp\{-(\theta^\top x + \theta_0)\}}$$



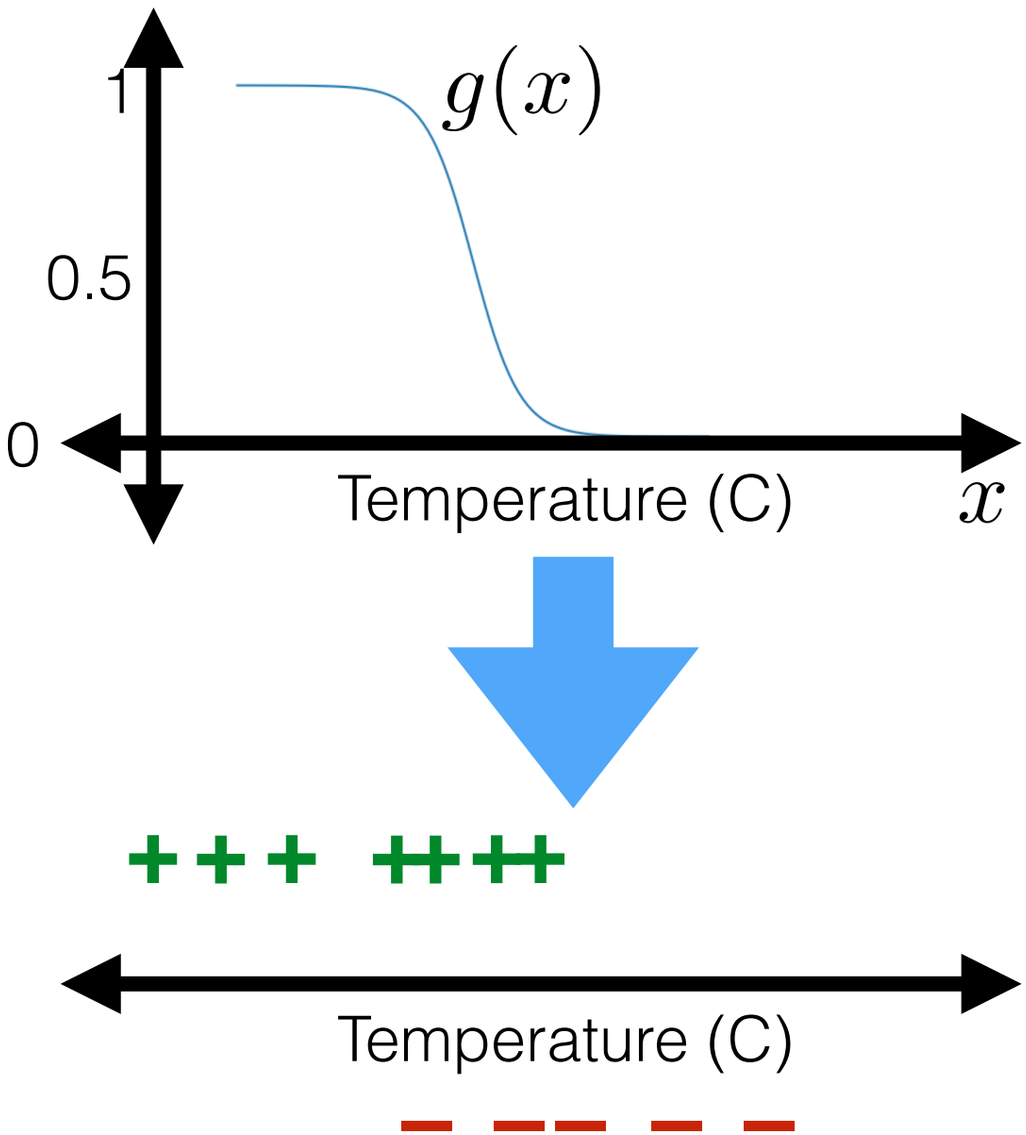


Capturing uncertainty

1 feature:

$$g(x) = \sigma(\theta x + \theta_0)$$

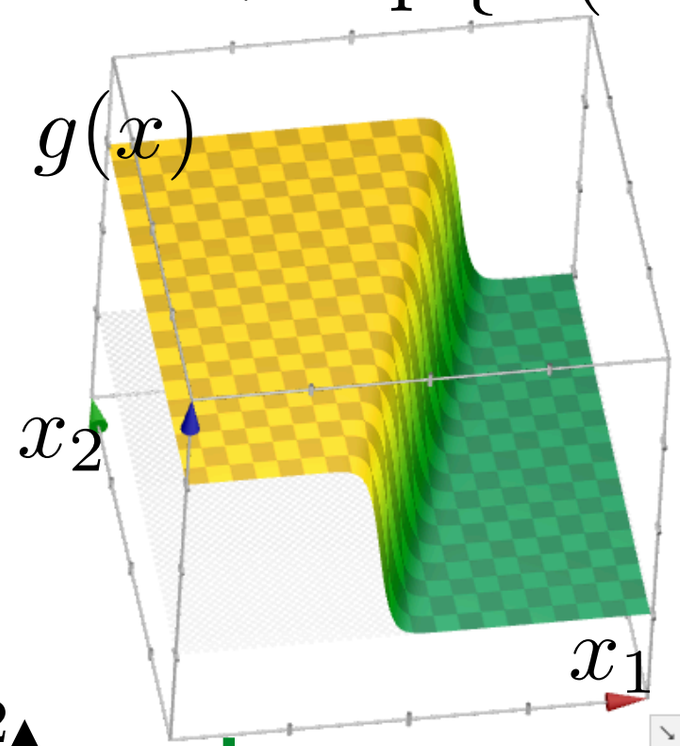
$$= \frac{1}{1 + \exp\{-(\theta x + \theta_0)\}}$$

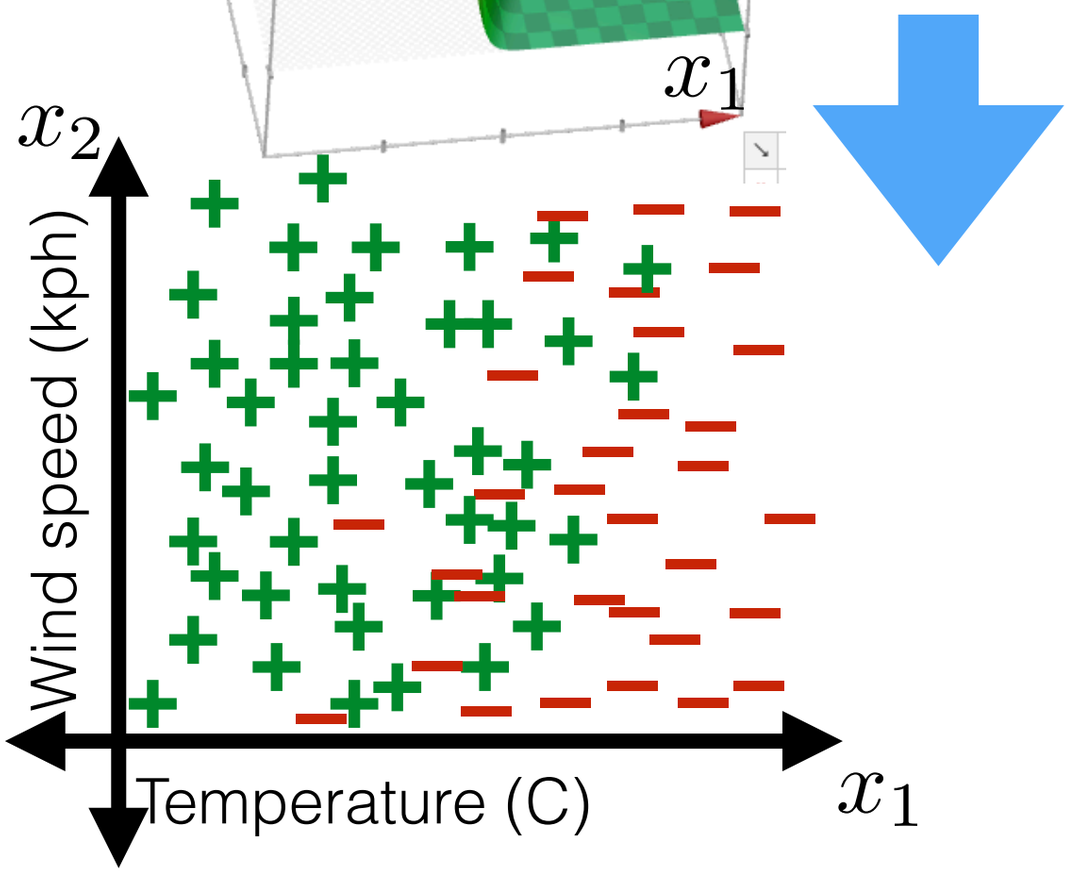


2 features:

$$g(x) = \sigma(\theta^\top x + \theta_0)$$

$$= \frac{1}{1 + \exp\{-(\theta^\top x + \theta_0)\}}$$





Linear logistic classification

aka logistic regression

Linear logistic classification

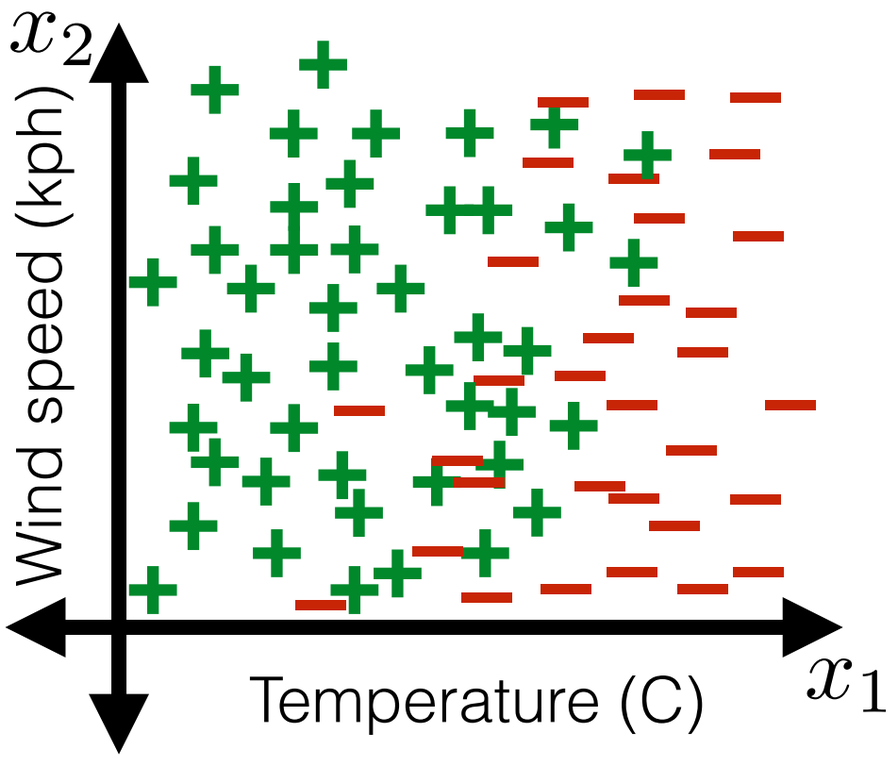
- How do we learn a classifier (i.e. learn θ, θ_0)?

aka logistic regression

Linear logistic classification

aka logistic regression

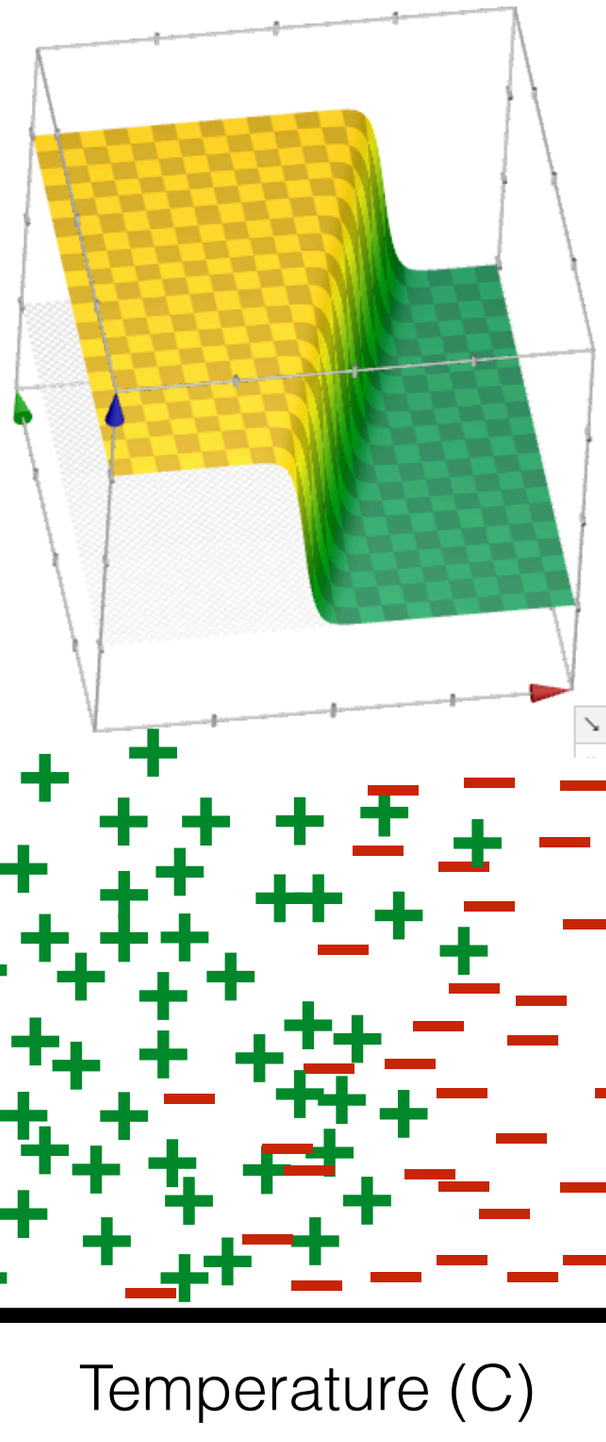
- How do we learn a classifier (i.e. learn θ, θ_0)?



Linear logistic classification

aka logistic regression

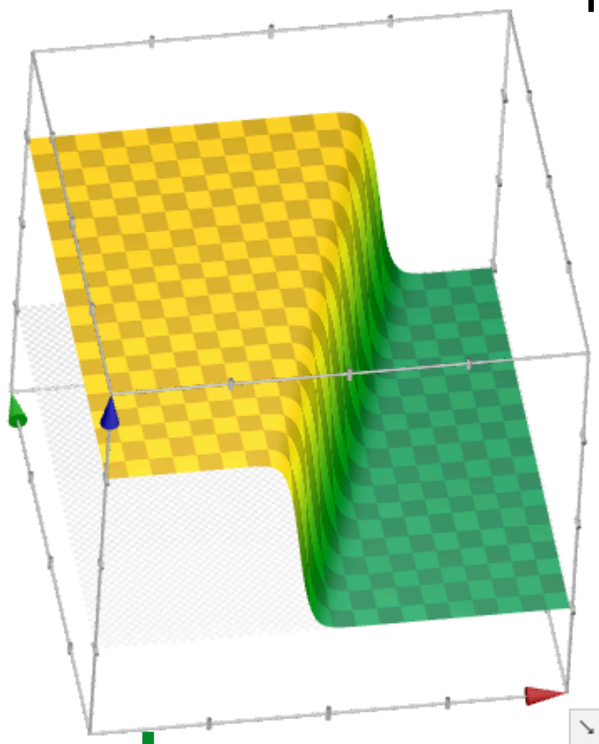
- How do we learn a classifier (i.e. learn θ, θ_0)?

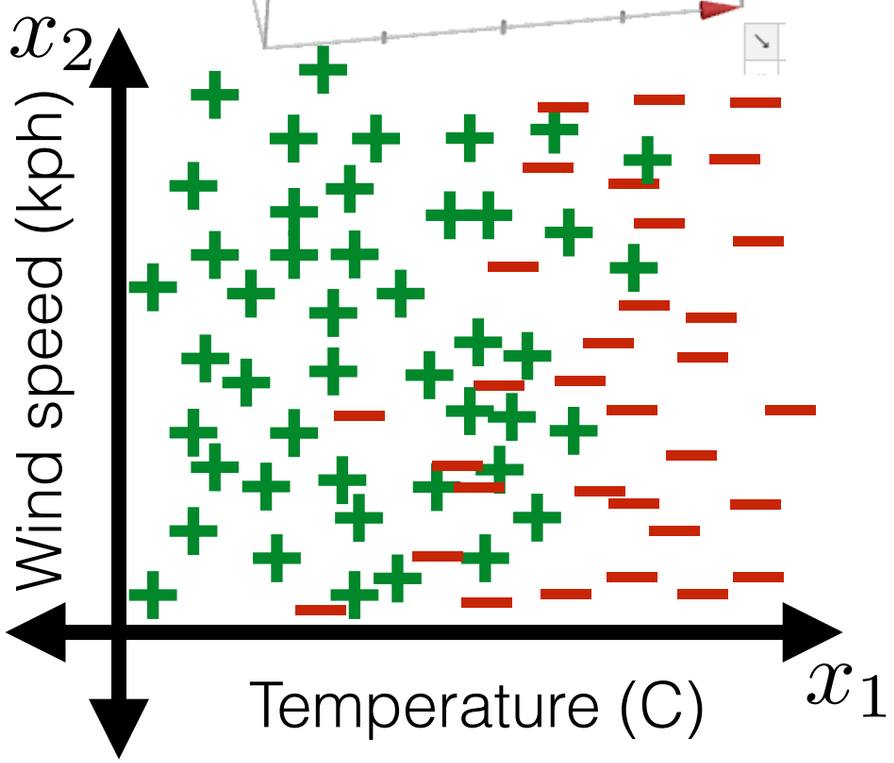


Linear logistic classification

aka logistic regression

- How do we learn a classifier (i.e. learn θ, θ_0)?
- How do we make predictions?

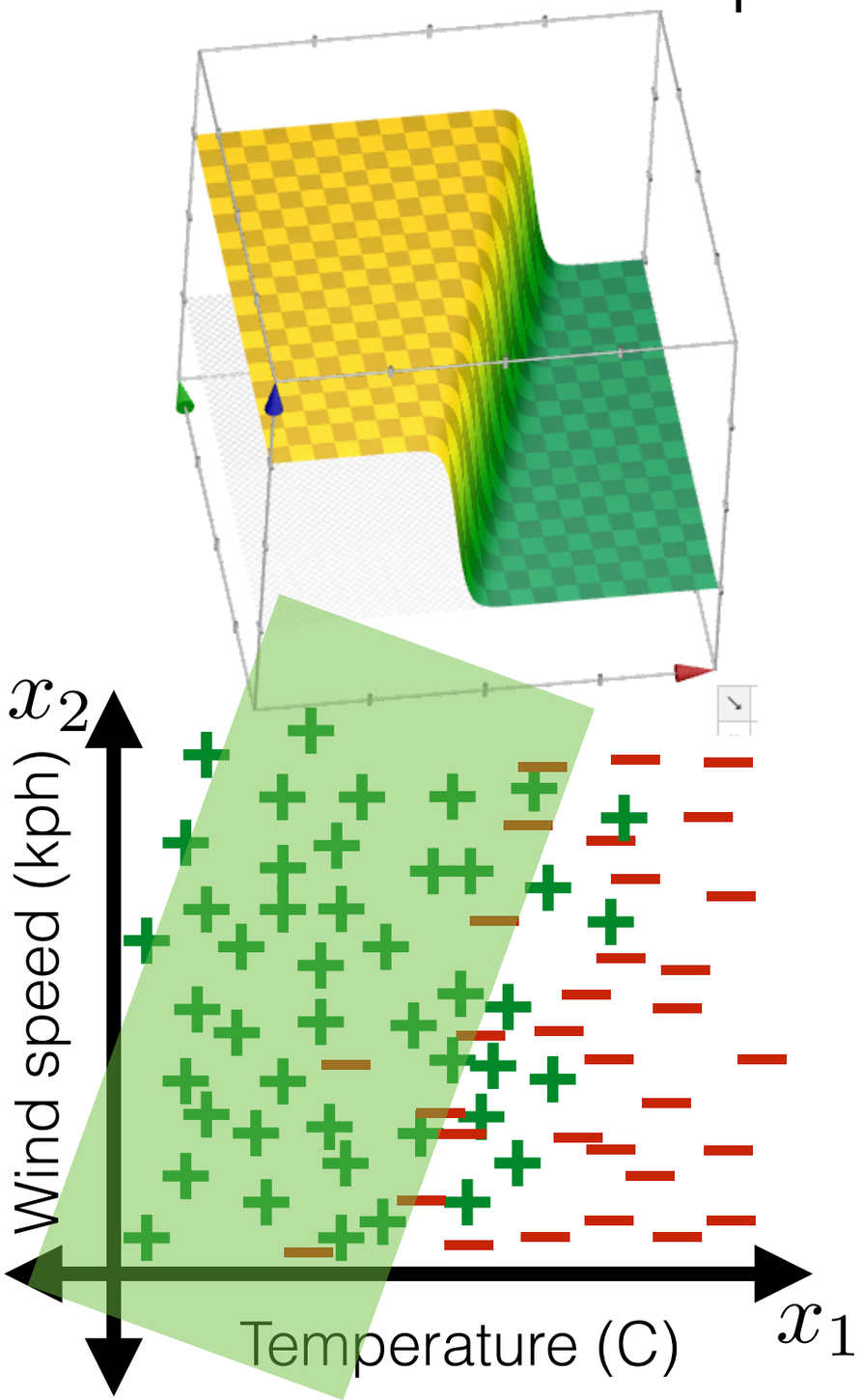




Linear logistic classification

aka logistic regression

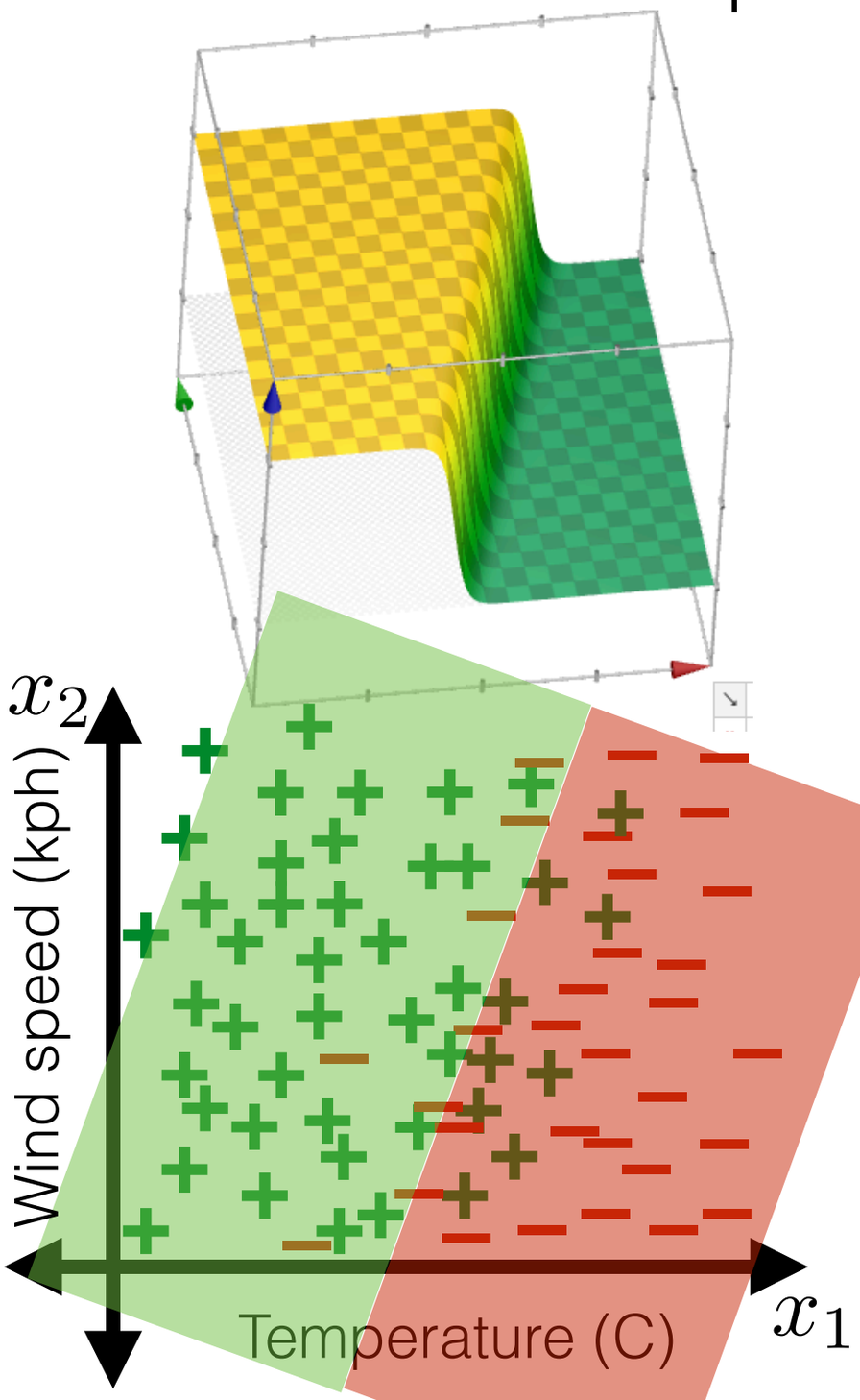
- How do we learn a classifier (i.e. learn θ, θ_0)?
- How do we make predictions?



Linear logistic classification

aka logistic regression

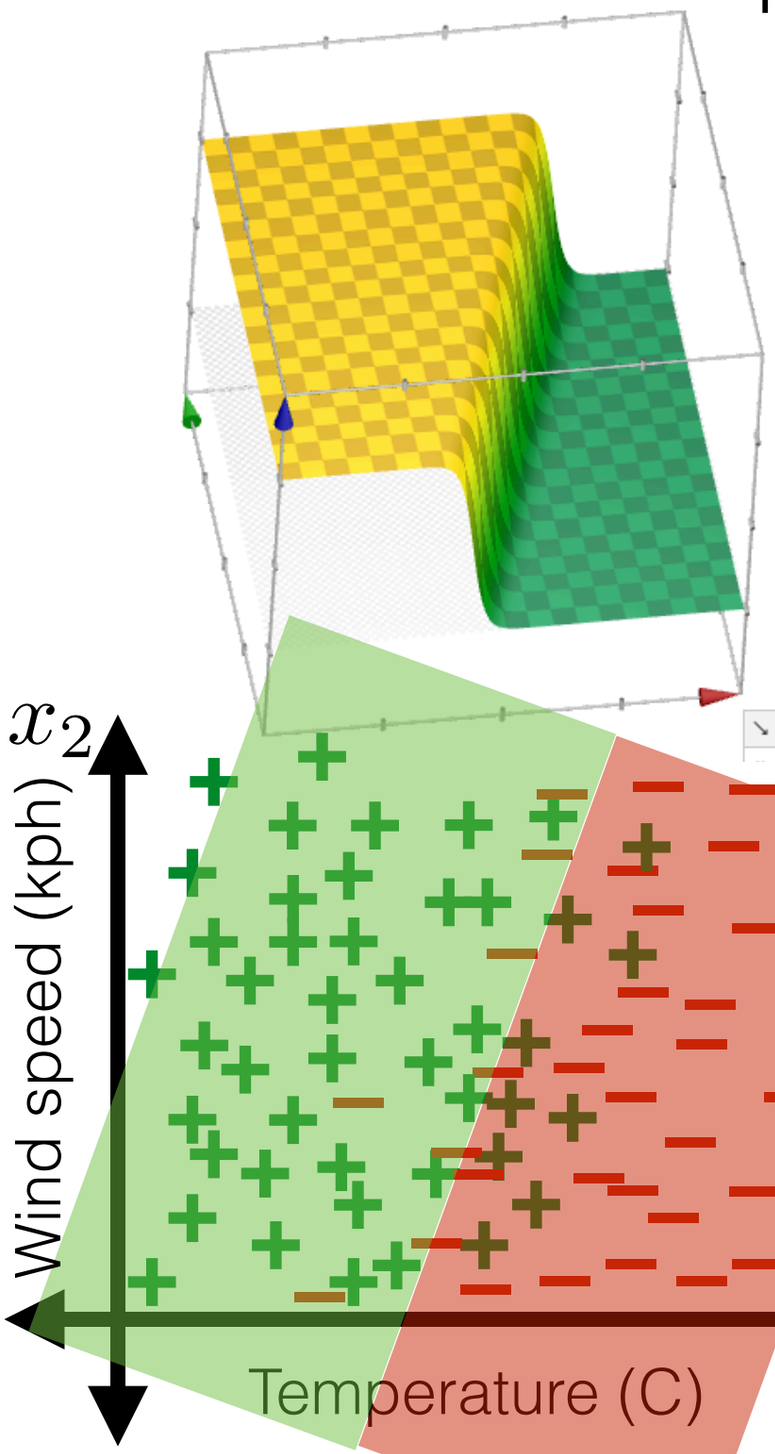
- How do we learn a classifier (i.e. learn θ, θ_0)?
- How do we make predictions?



Linear logistic classification

aka logistic regression

- How do we learn a classifier (i.e. learn θ, θ_0)?
- How do we make predictions?

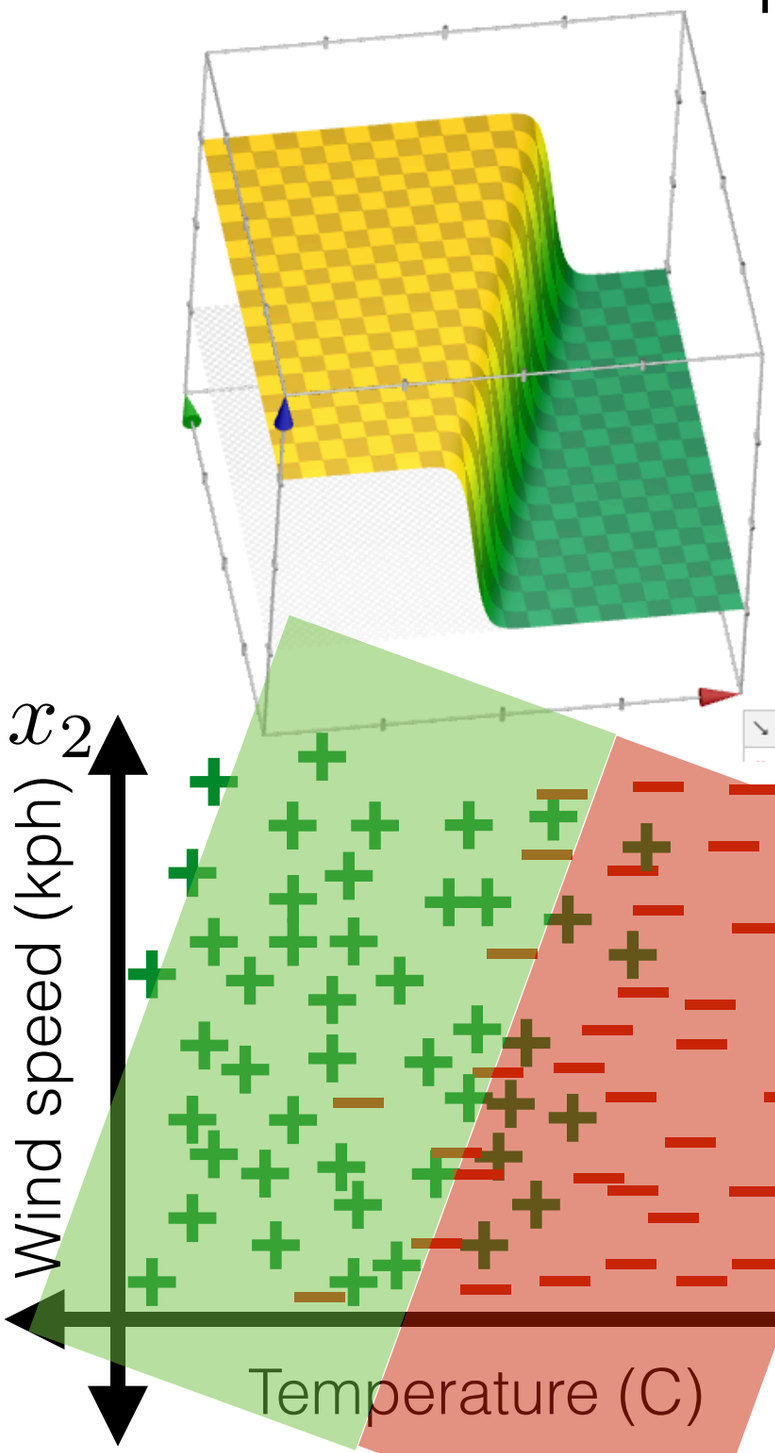


- Idea: predict +1 if:

Linear logistic classification

aka logistic regression

- How do we learn a classifier (i.e. learn θ, θ_0)?
- How do we make predictions?

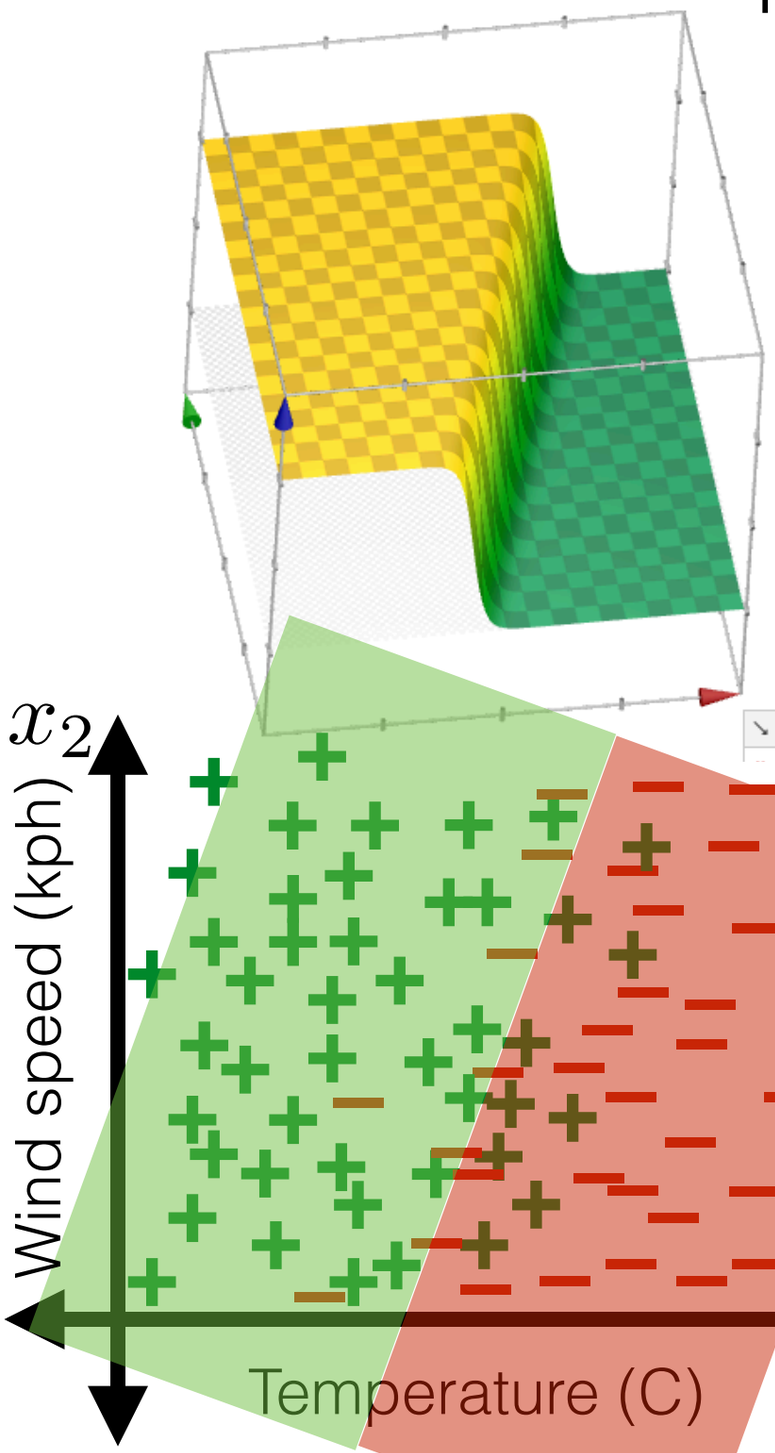


- Idea: predict +1 if: probability > 0.5

Linear logistic classification

aka logistic regression

- How do we learn a classifier (i.e. learn θ, θ_0)?
- How do we make predictions?



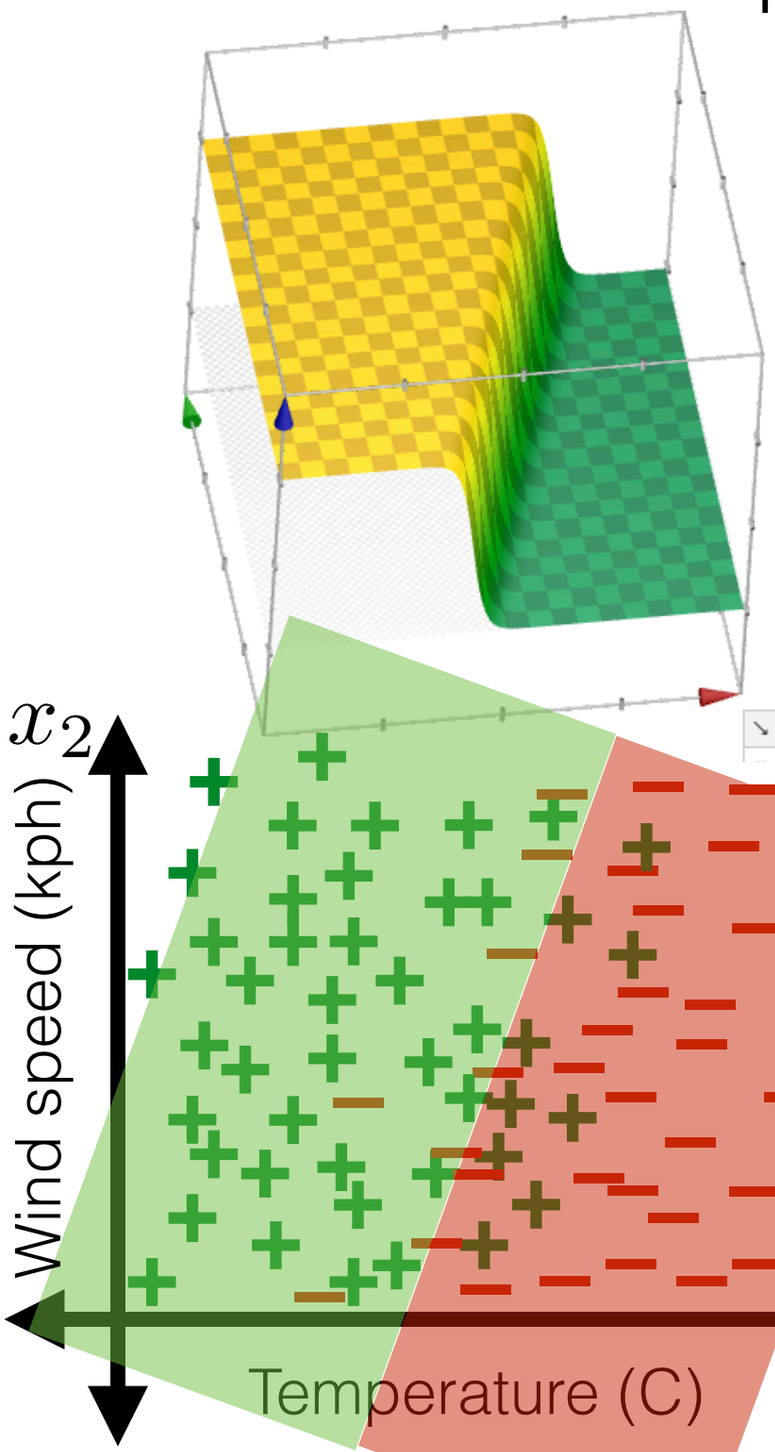
- Idea: predict +1 if: probability > 0.5

$$\sigma(\theta^\top x + \theta_0) > 0.5$$

Linear logistic classification

aka logistic regression

- How do we learn a classifier (i.e. learn θ, θ_0)?
- How do we make predictions?



- Idea: predict +1 if: probability > 0.5

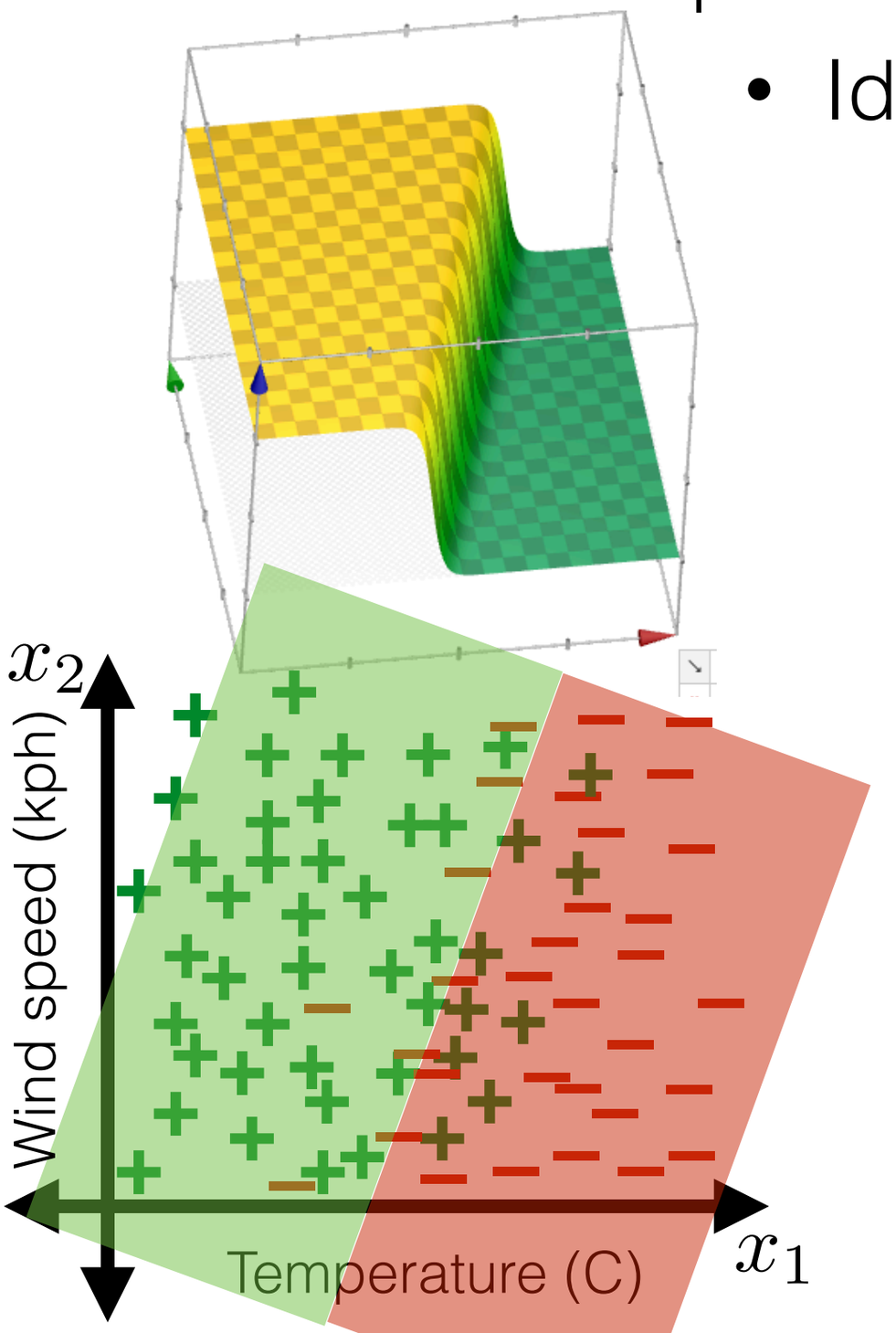
$$\sigma(\theta^\top x + \theta_0) > 0.5$$

$$\frac{1}{1 + \exp\{-(\theta^\top x + \theta_0)\}} > 0.5$$

Linear logistic classification

aka logistic regression

- How do we learn a classifier (i.e. learn θ, θ_0)?
- How do we make predictions?



- Idea: predict +1 if: probability > 0.5

$$\sigma(\theta^\top x + \theta_0) > 0.5$$

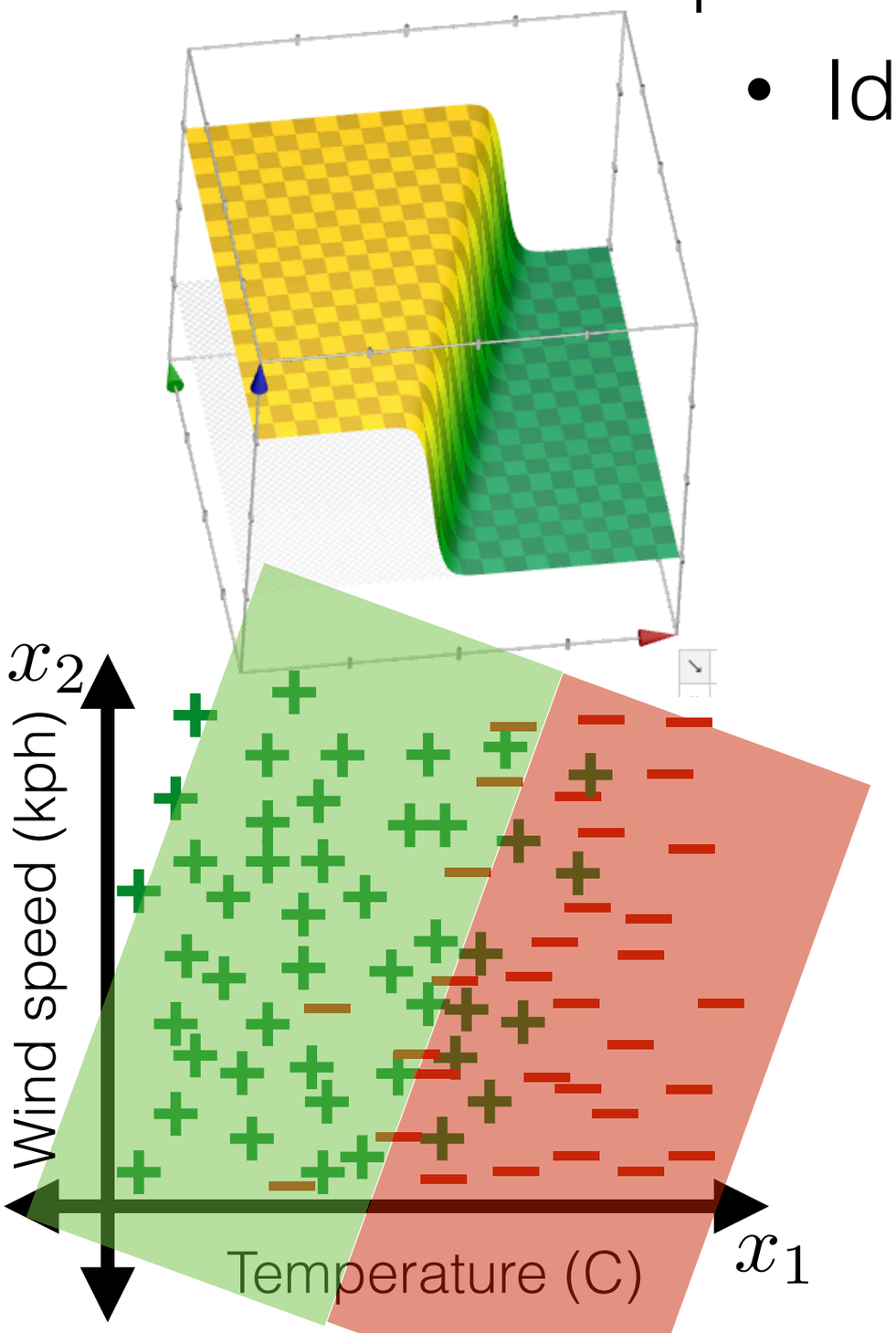
$$\frac{1}{1 + \exp\{-(\theta^\top x + \theta_0)\}} > 0.5$$

$$\exp\{-(\theta^\top x + \theta_0)\} < 1$$

Linear logistic classification

aka logistic regression

- How do we learn a classifier (i.e. learn θ, θ_0)?
- How do we make predictions?



- Idea: predict +1 if: probability > 0.5

$$\sigma(\theta^\top x + \theta_0) > 0.5$$

$$\frac{1}{1 + \exp\{-(\theta^\top x + \theta_0)\}} > 0.5$$

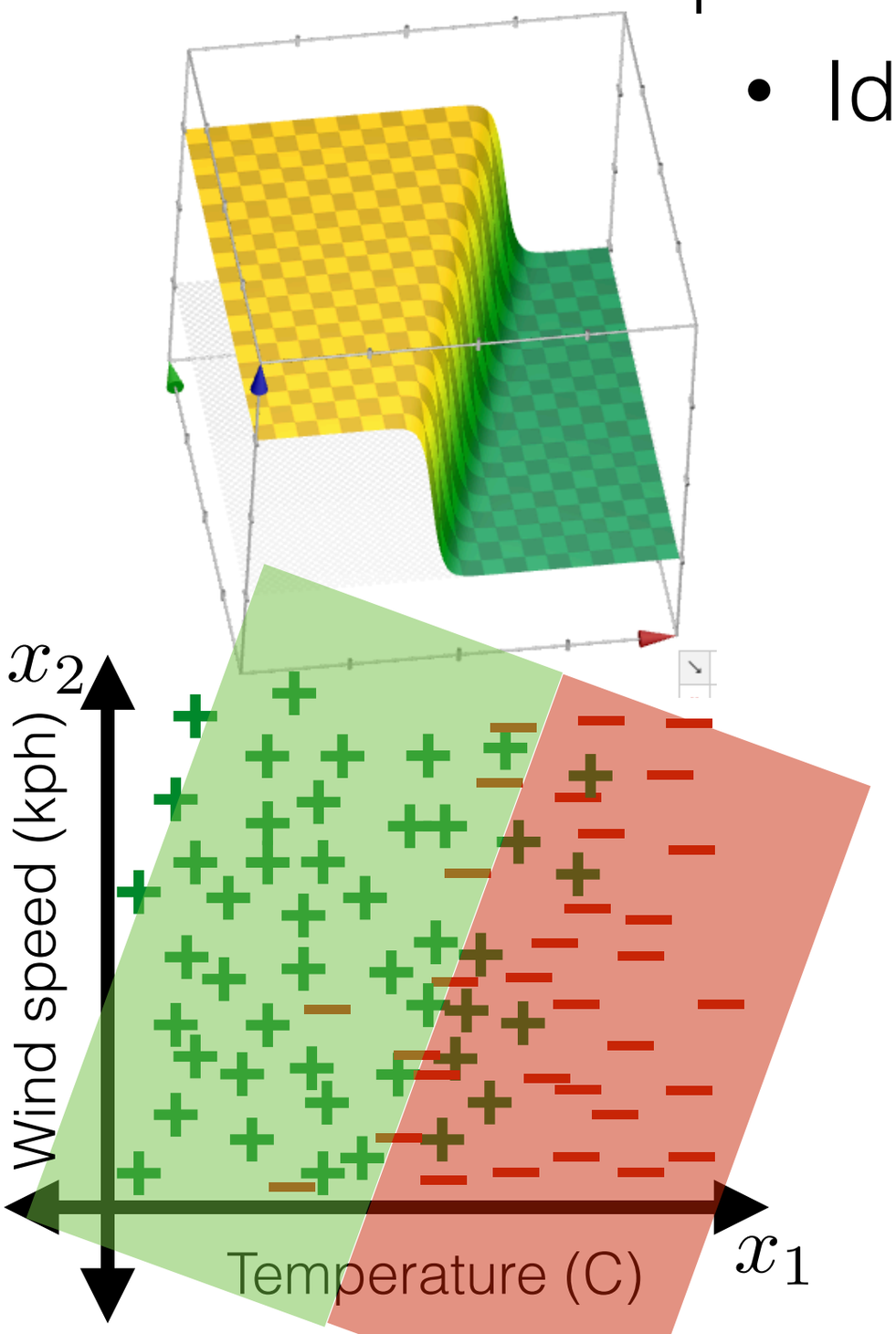
$$\exp\{-(\theta^\top x + \theta_0)\} < 1$$

$$\theta^\top x + \theta_0 > 0$$

Linear logistic classification

aka logistic regression

- How do we learn a classifier (i.e. learn θ, θ_0)?
- How do we make predictions?



- Idea: predict +1 if: probability > 0.5

$$\sigma(\theta^\top x + \theta_0) > 0.5$$

$$\frac{1}{1 + \exp\{-(\theta^\top x + \theta_0)\}} > 0.5$$

$$\exp\{-(\theta^\top x + \theta_0)\} < 1$$

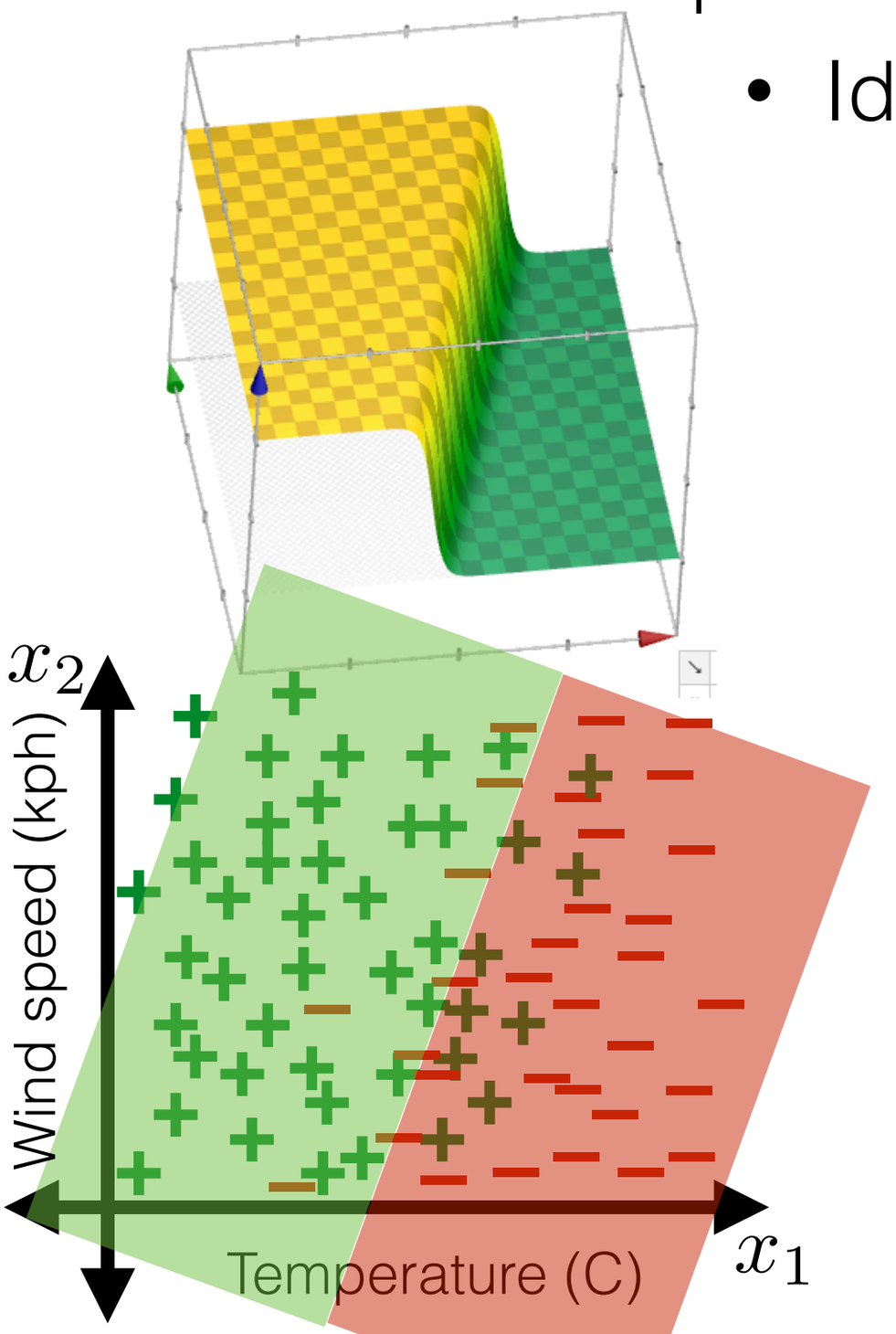
$$\theta^\top x + \theta_0 > 0$$

- Same hypothesis class as before!

Linear logistic classification

aka logistic regression

- How do we learn a classifier (i.e. learn θ, θ_0)?
- How do we make predictions?



- Idea: predict +1 if: probability > 0.5

$$\sigma(\theta^\top x + \theta_0) > 0.5$$

$$\frac{1}{1 + \exp\{-(\theta^\top x + \theta_0)\}} > 0.5$$

$$\exp\{-(\theta^\top x + \theta_0)\} < 1$$

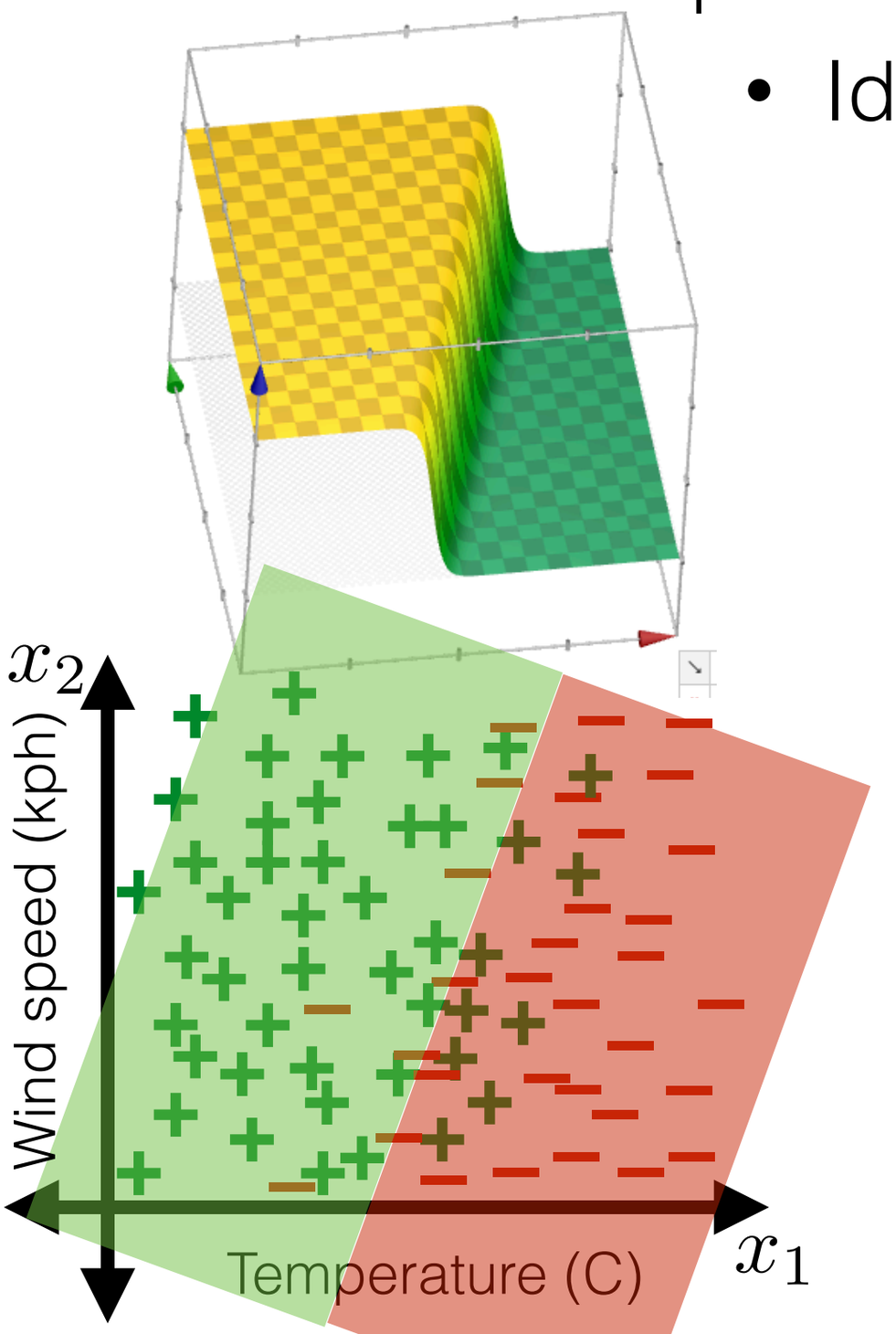
$$\theta^\top x + \theta_0 > 0$$

- Same hypothesis class as before! But we will get:

Linear logistic classification

aka logistic regression

- How do we learn a classifier (i.e. learn θ, θ_0)?
- How do we make predictions?



- Idea: predict +1 if: probability > 0.5

$$\sigma(\theta^\top x + \theta_0) > 0.5$$

$$\frac{1}{1 + \exp\{-(\theta^\top x + \theta_0)\}} > 0.5$$

$$\exp\{-(\theta^\top x + \theta_0)\} < 1$$

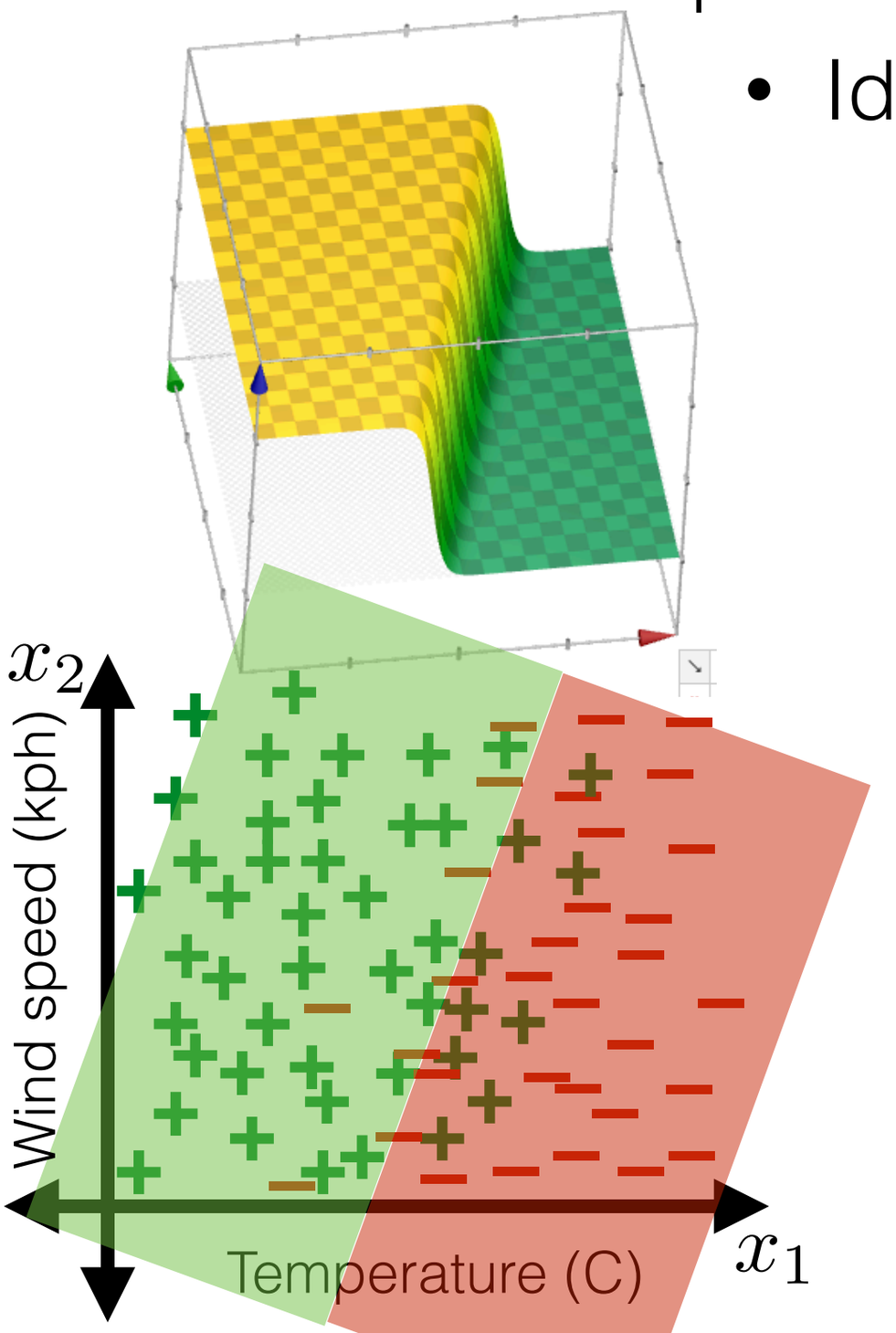
$$\theta^\top x + \theta_0 > 0$$

- Same hypothesis class as before! But we will get:
  - Uncertainties

Linear logistic classification

aka logistic regression

- How do we learn a classifier (i.e. learn θ, θ_0)?
- How do we make predictions?



- Idea: predict +1 if: probability > 0.5

$$\sigma(\theta^\top x + \theta_0) > 0.5$$

$$\frac{1}{1 + \exp\{-(\theta^\top x + \theta_0)\}} > 0.5$$

$$\exp\{-(\theta^\top x + \theta_0)\} < 1$$

$$\theta^\top x + \theta_0 > 0$$

- Same hypothesis class as before! But we will get:
  - Uncertainties
  - Quality guarantees when data not linearly separable

Linear logistic classification

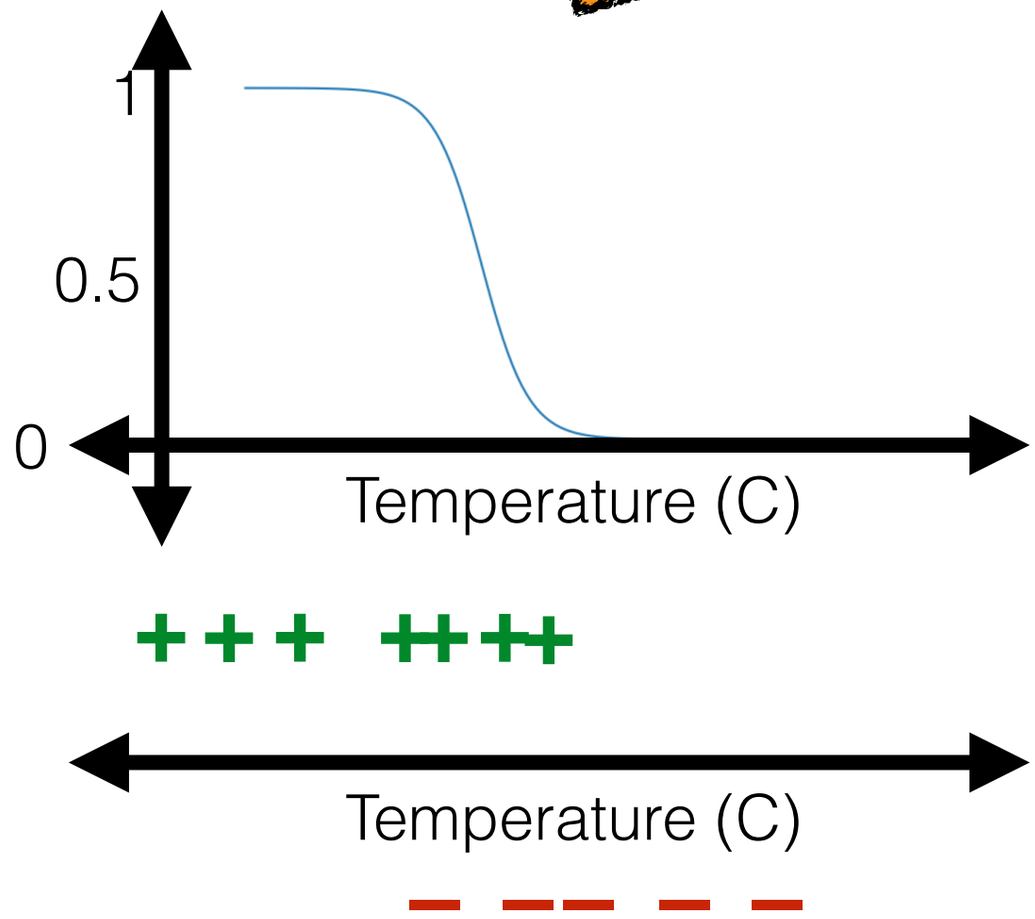
aka logistic regression

- How do we learn a classifier (i.e. learn θ, θ_0)?

Linear logistic classification

aka logistic regression

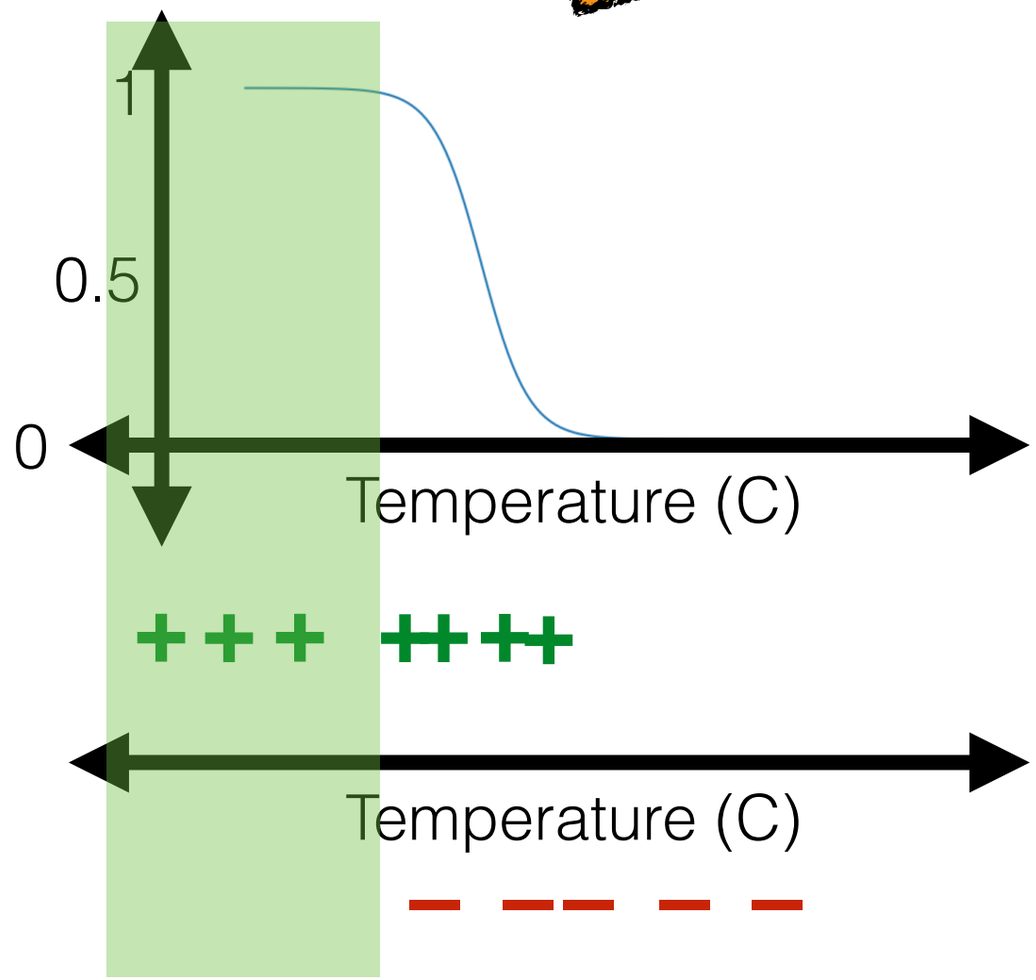
- How do we learn a classifier (i.e. learn θ, θ_0)?



Linear logistic classification

aka logistic regression

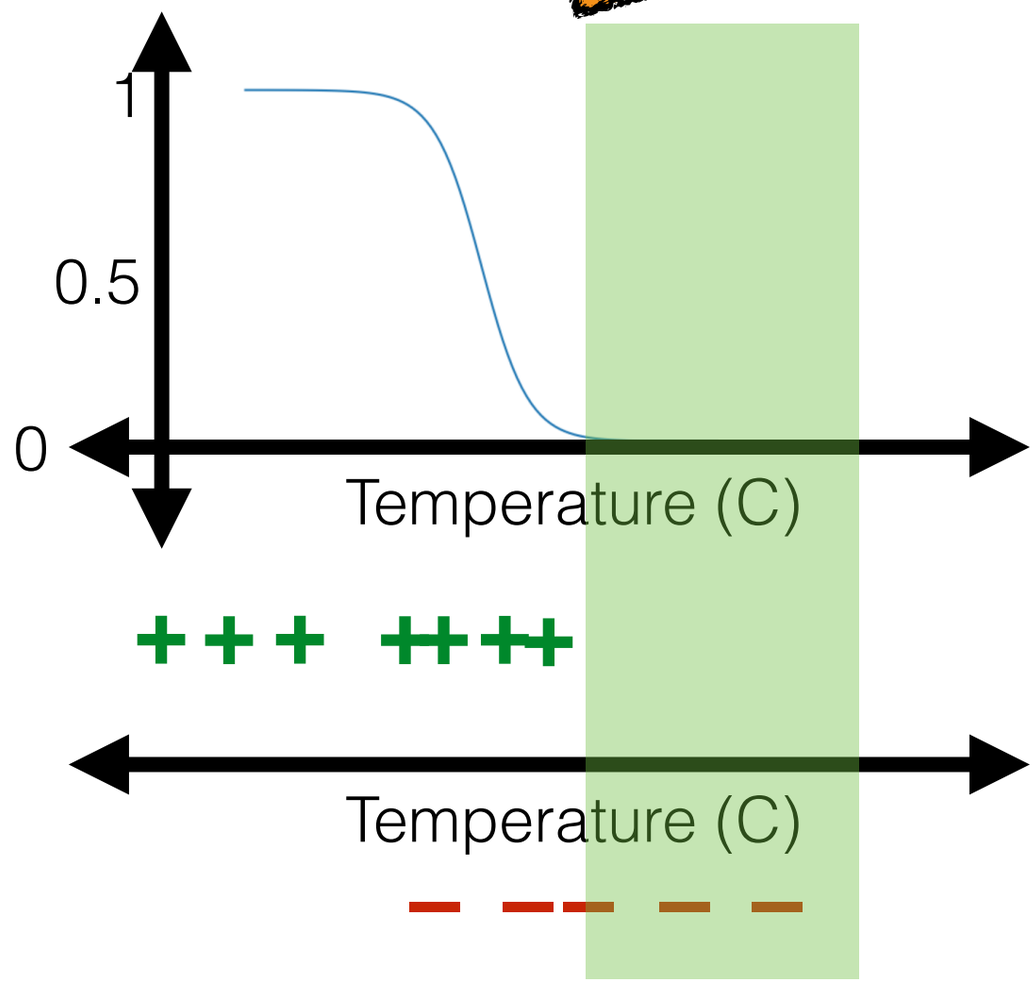
- How do we learn a classifier (i.e. learn θ, θ_0)?



Linear logistic classification

aka logistic regression

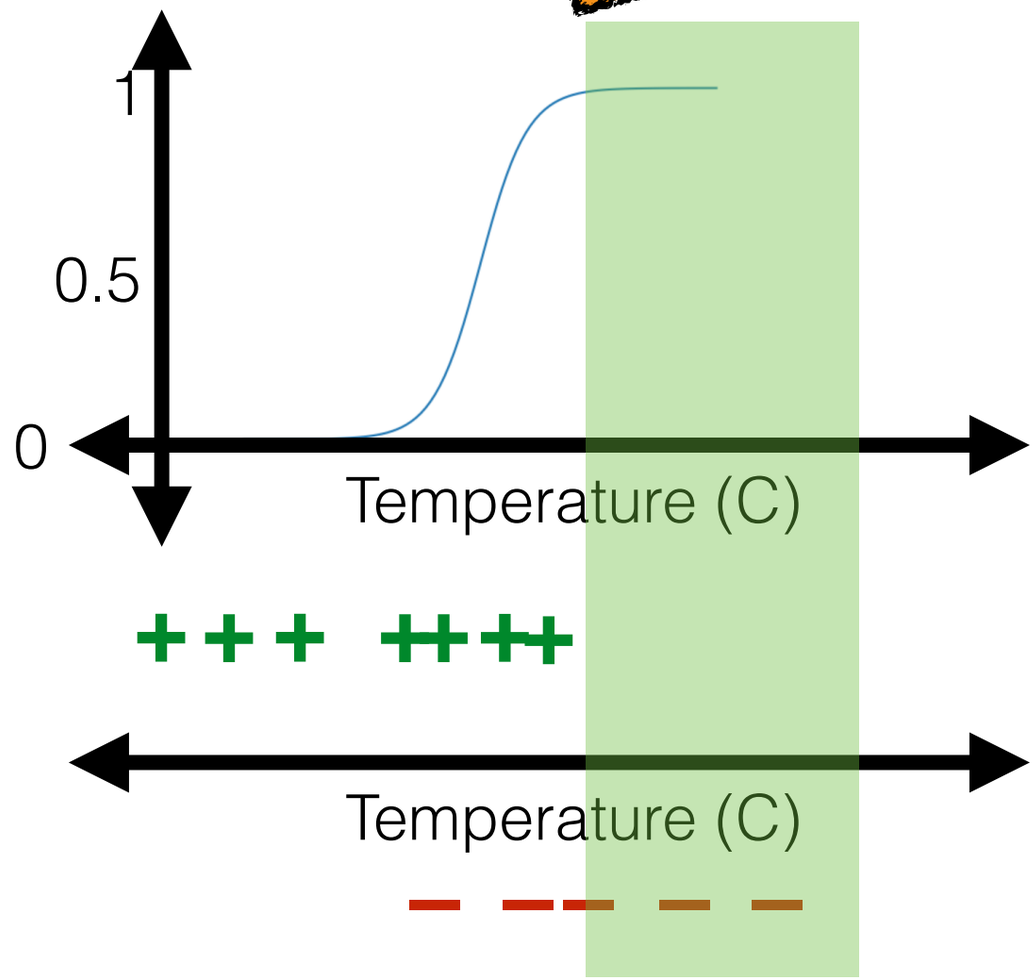
- How do we learn a classifier (i.e. learn θ, θ_0)?



Linear logistic classification

aka logistic regression

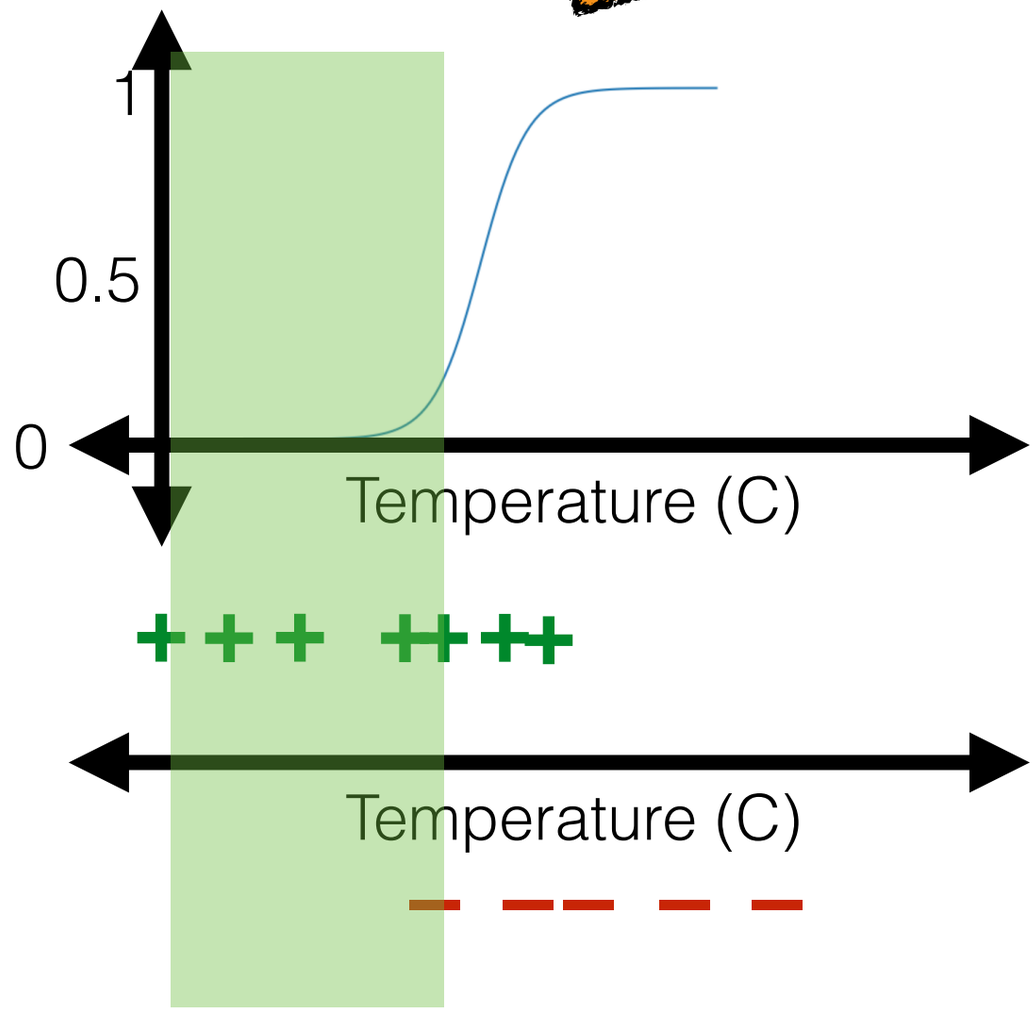
- How do we learn a classifier (i.e. learn θ, θ_0)?



Linear logistic classification

aka logistic regression

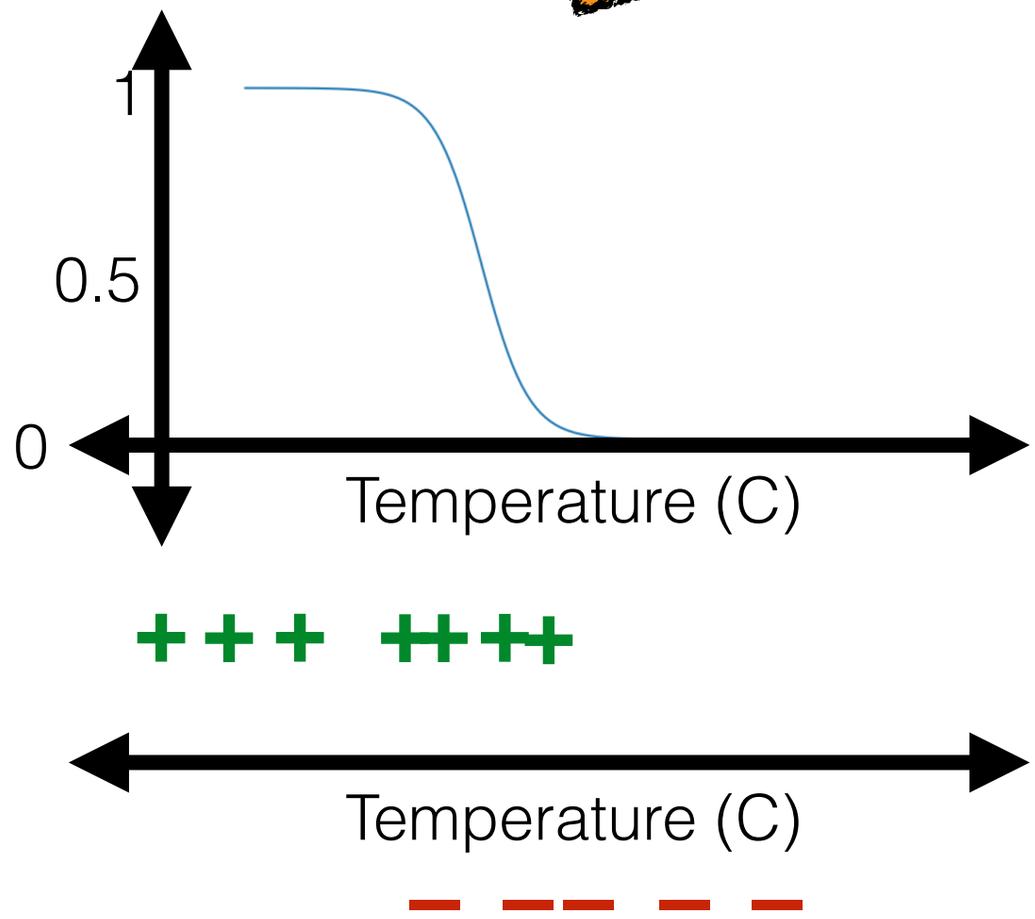
- How do we learn a classifier (i.e. learn θ, θ_0)?



Linear logistic classification

aka logistic regression

- How do we learn a classifier (i.e. learn θ, θ_0)?

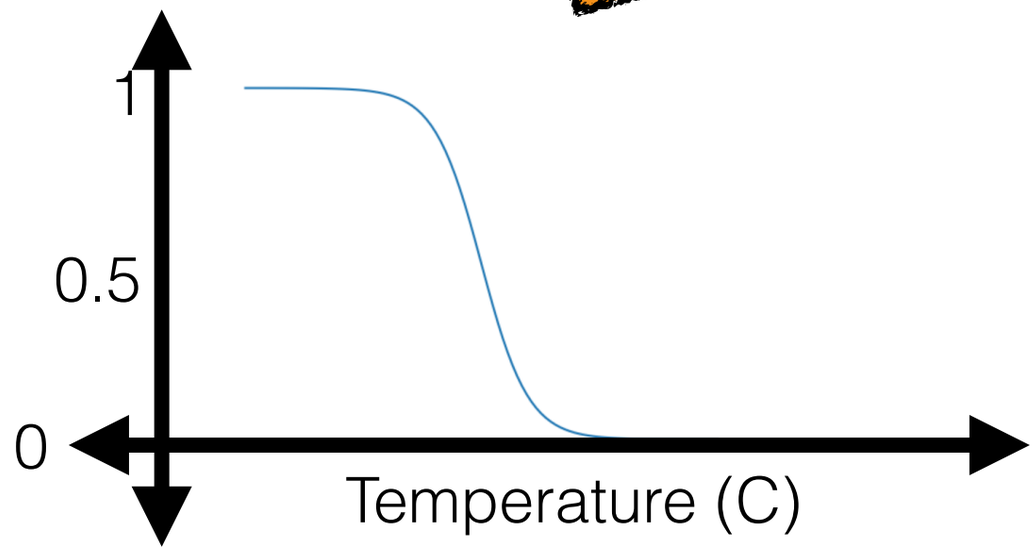


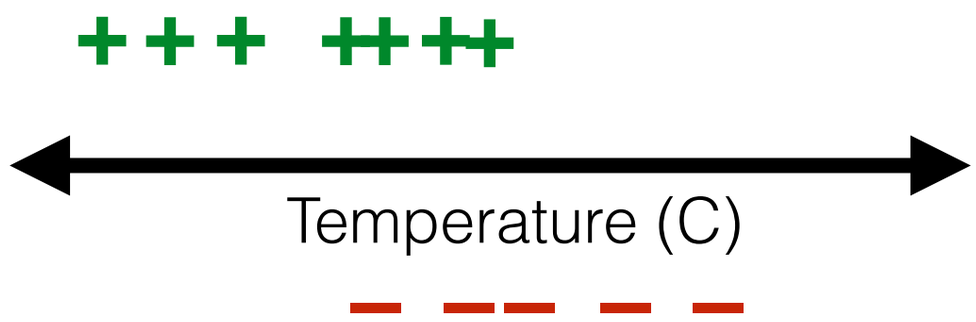
Linear logistic classification

aka logistic regression

- How do we learn a classifier (i.e. learn θ, θ_0)?

Probability(data)





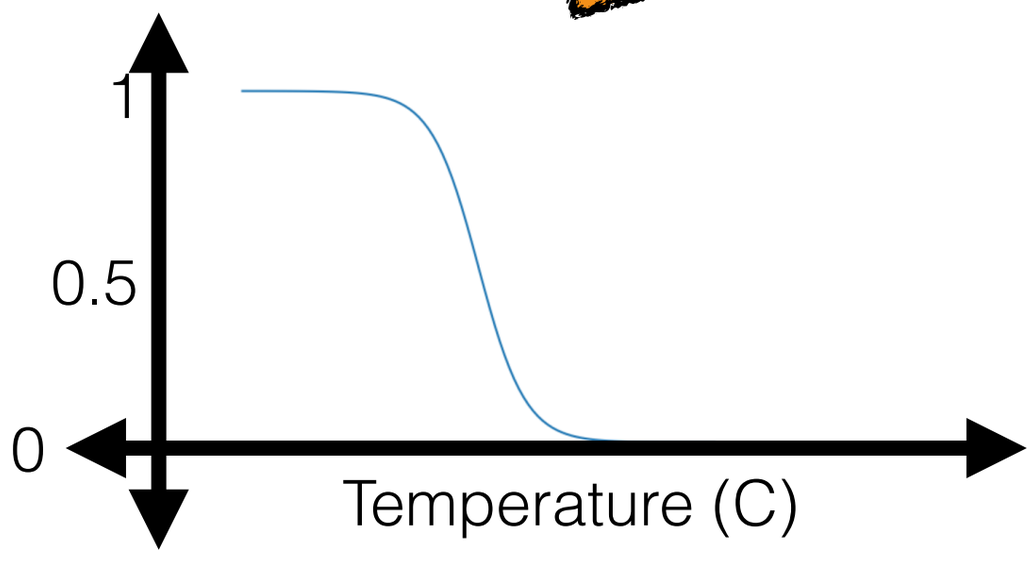
Linear logistic classification

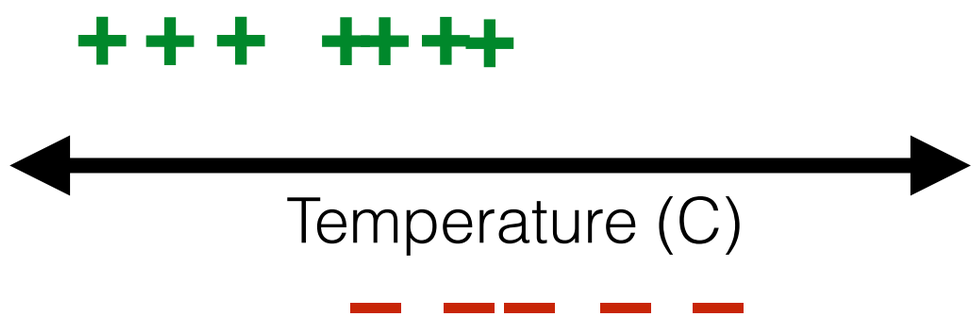
aka logistic regression

- How do we learn a classifier (i.e. learn θ, θ_0)?

Probability(data)

$$= \prod_{i=1}^{n} \text{Probability(data point } i)$$





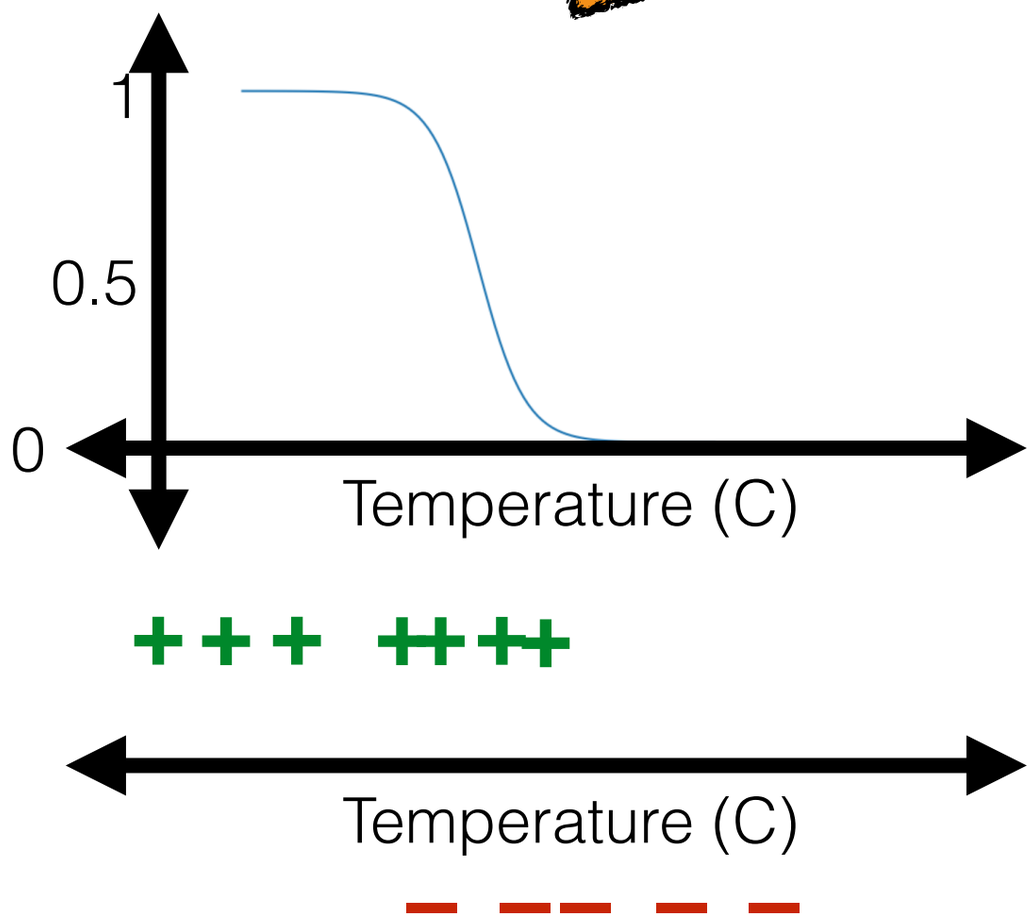
Linear logistic classification

aka logistic regression

- How do we learn a classifier (i.e. learn θ, θ_0)?

Probability(data)

$$= \prod_{i=1}^{n} \text{Probability(data point } i\text{)}$$



Linear logistic classification

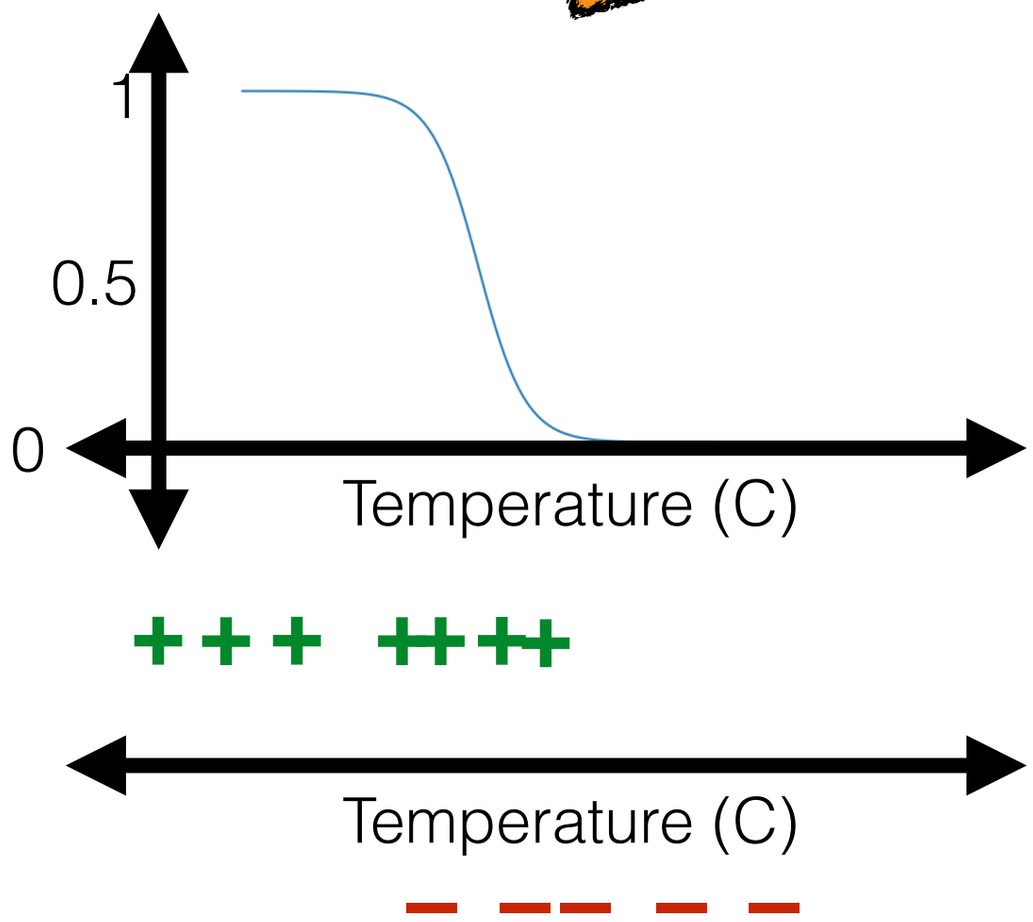
aka logistic regression

- How do we learn a classifier (i.e. learn θ, θ_0)?

Probability(data)

$$= \prod_{i=1}^{n} \text{Probability}(\text{data point } i)$$

[Let $g^{(i)} = \sigma(\theta^\top x^{(i)} + \theta_0)$]



Linear logistic classification

aka logistic regression

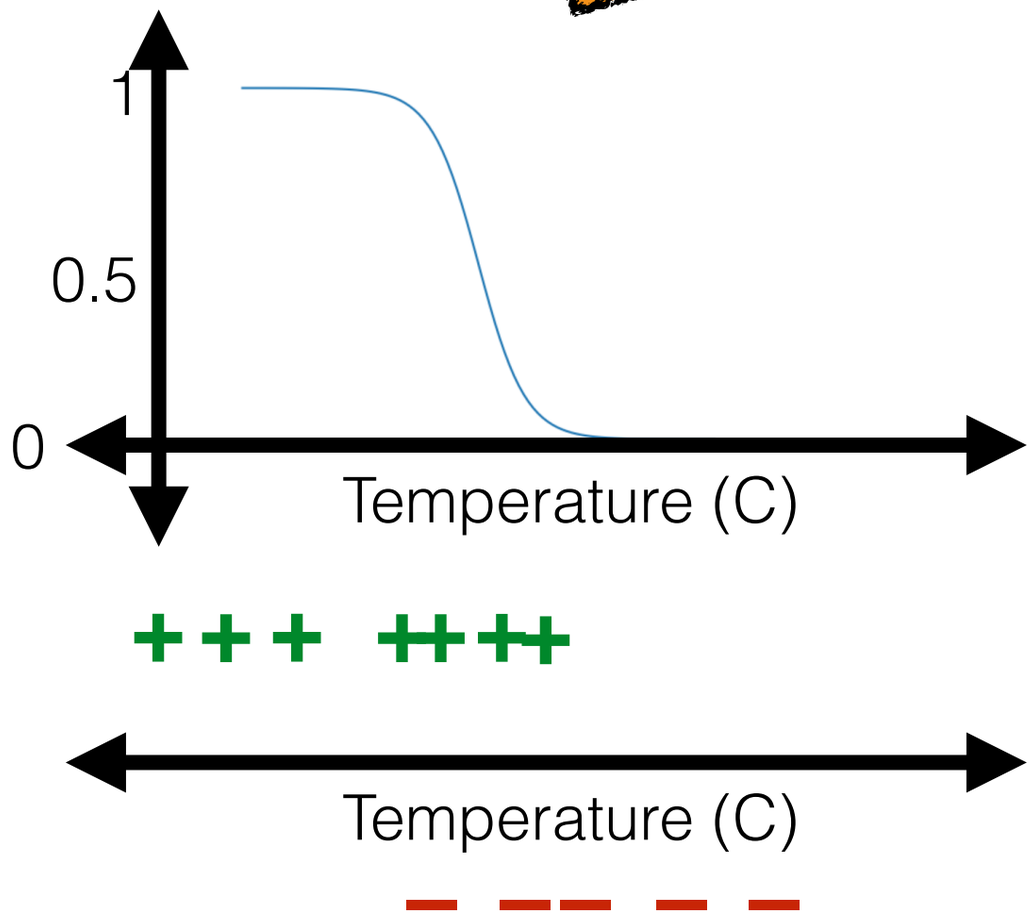
- How do we learn a classifier (i.e. learn θ, θ_0)?

Probability(data)

$$= \prod_{i=1}^{n} \text{Probability}(\text{data point } i)$$

$$[\text{Let } g^{(i)} = \sigma(\theta^\top x^{(i)} + \theta_0)\,]$$

$$= \prod_{i=1}^{n} \begin{cases} g^{(i)} \text{ if } y^{(i)} = +1 \\ (1 - g^{(i)}) \text{ else} \end{cases}$$



Linear logistic classification

aka logistic regression

- How do we learn a classifier (i.e. learn θ, θ_0)?

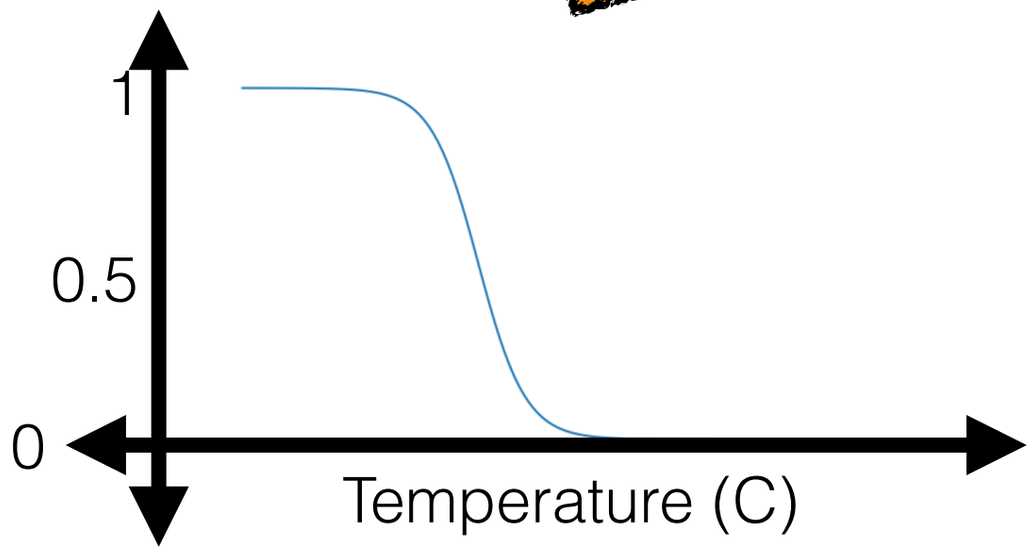
Probability(data)

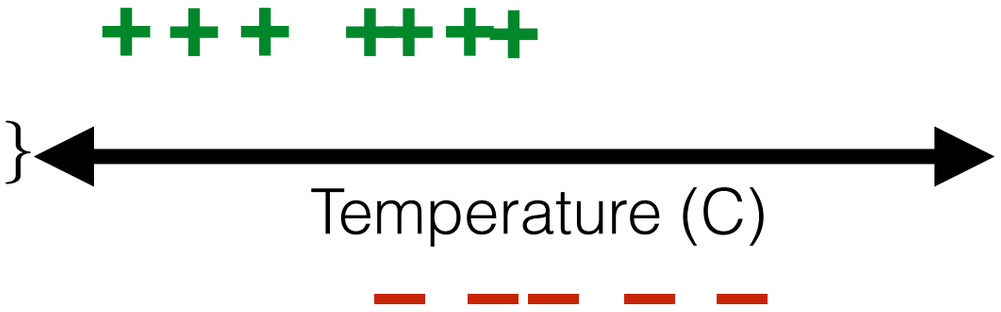
$$= \prod_{i=1}^{n} \text{Probability}(\text{data point } i)$$

$$[\text{Let } g^{(i)} = \sigma(\theta^\top x^{(i)} + \theta_0)\,]$$

$$= \prod_{i=1}^{n} \begin{cases} g^{(i)} \text{ if } y^{(i)} = +1 \\ (1 - g^{(i)}) \text{ else} \end{cases}$$

$$= \prod_{i=1}^{n} (g^{(i)})^{\mathbf{1}\{y^{(i)}=+1\}} (1 - g^{(i)})^{\mathbf{1}\{y^{(i)}\neq+1\}}$$





Linear logistic classification

aka logistic regression

- How do we learn a classifier (i.e. learn θ, θ_0)?

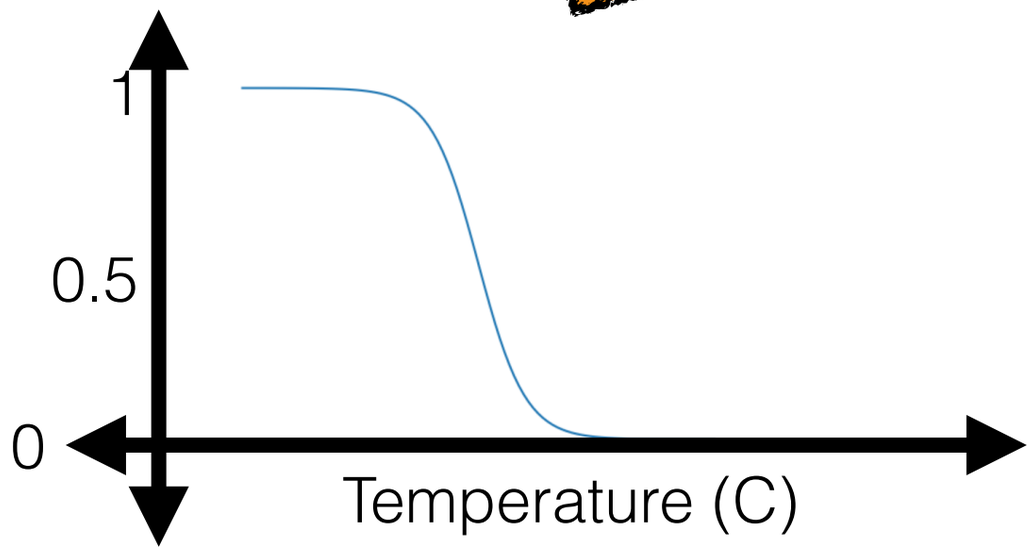
Probability(data)

$$= \prod_{i=1}^{n} \text{Probability}(\text{data point } i)$$

$$[\text{Let } g^{(i)} = \sigma(\theta^\top x^{(i)} + \theta_0)\,]$$

$$= \prod_{i=1}^{n} \begin{cases} g^{(i)} \text{ if } y^{(i)} = +1 \\ (1 - g^{(i)}) \text{ else} \end{cases}$$

$$= \prod_{i=1}^{n} (g^{(i)})^{\mathbf{1}\{y^{(i)}=+1\}} (1 - g^{(i)})^{\mathbf{1}\{y^{(i)}\neq+1\}}$$



Linear logistic classification

aka logistic regression

- How do we learn a classifier (i.e. learn θ, θ_0)?

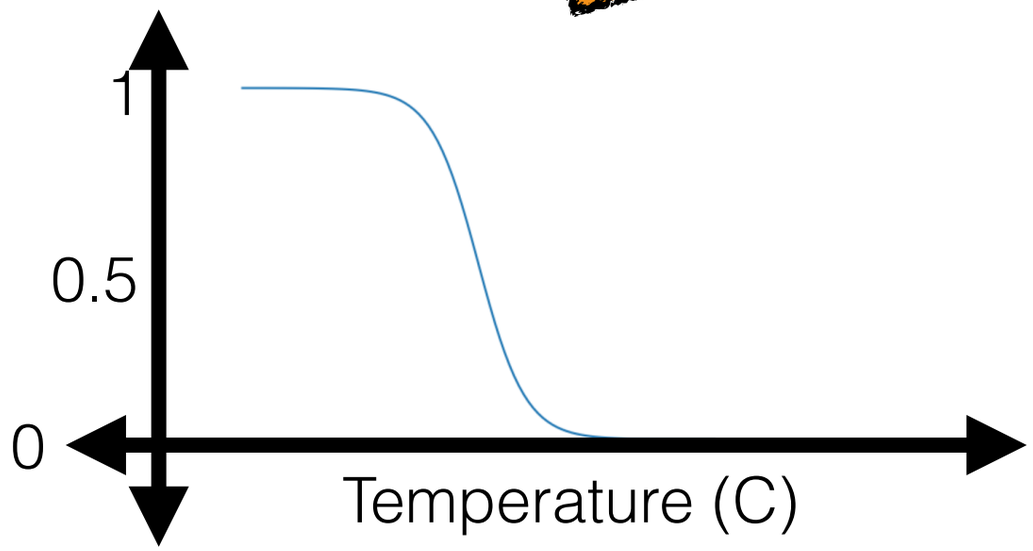
Probability(data)

$$= \prod_{i=1}^{n} \text{Probability}(\text{data point } i)$$

$$[\text{Let } g^{(i)} = \sigma(\theta^\top x^{(i)} + \theta_0)\,]$$

$$= \prod_{i=1}^{n} \begin{cases} g^{(i)} \text{ if } y^{(i)} = +1 \\ (1 - g^{(i)}) \text{ else} \end{cases}$$

$$= \prod_{i=1}^{n} (g^{(i)})^{\mathbf{1}\{y^{(i)}=+1\}} (1 - g^{(i)})^{\mathbf{1}\{y^{(i)}\neq+1\}}$$



Linear logistic classification

aka logistic regression

- How do we learn a classifier (i.e. learn θ, θ_0)?

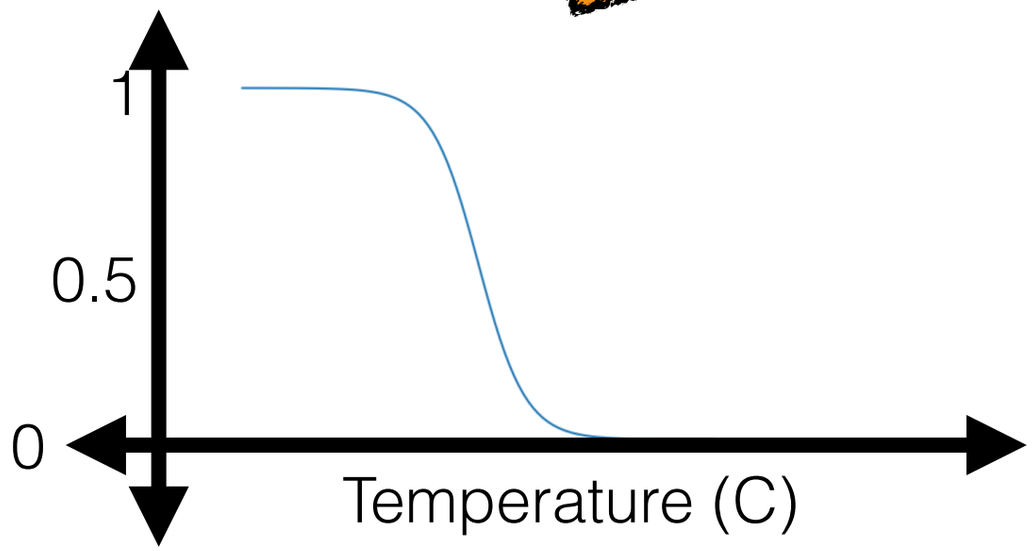
Probability(data)

$$= \prod_{i=1}^{n} \text{Probability}(\text{data point } i)$$

$$[\text{Let } g^{(i)} = \sigma(\theta^\top x^{(i)} + \theta_0)\,]$$

$$= \prod_{i=1}^{n} \begin{cases} g^{(i)} \text{ if } y^{(i)} = +1 \\ (1 - g^{(i)}) \text{ else} \end{cases}$$

$$= \prod_{i=1}^{n} (g^{(i)})^{\mathbf{1}\{y^{(i)}=+1\}} (1 - g^{(i)})^{\mathbf{1}\{y^{(i)}\neq+1\}}$$



Linear logistic classification

aka logistic regression

- How do we learn a classifier (i.e. learn θ, θ_0)?

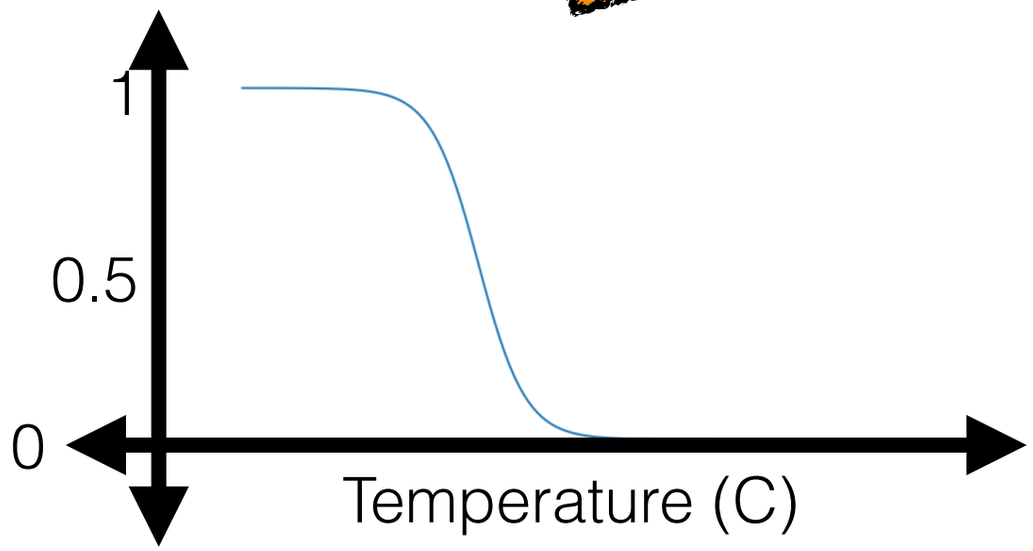
Probability(data)

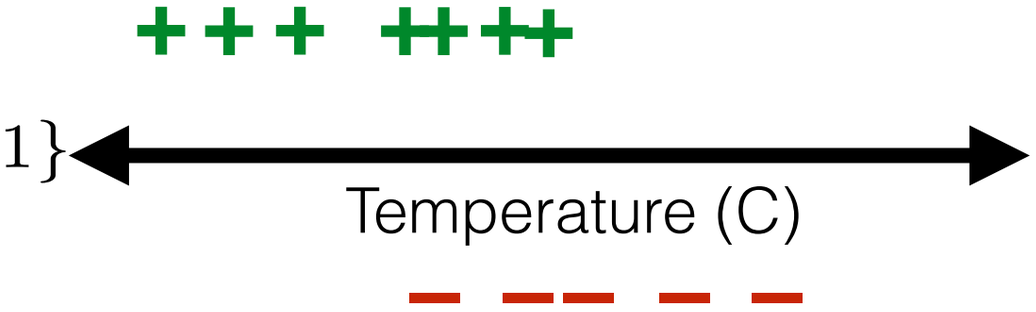
$$= \prod_{i=1}^{n} \text{Probability}(\text{data point } i)$$

$$[\text{Let } g^{(i)} = \sigma(\theta^\top x^{(i)} + \theta_0)\,]$$

$$= \prod_{i=1}^{n} \begin{cases} g^{(i)} \text{ if } y^{(i)} = +1 \\ (1 - g^{(i)}) \text{ else} \end{cases}$$

$$= \prod_{i=1}^{n} (g^{(i)})^{\mathbf{1}\{y^{(i)}=+1\}} (1 - g^{(i)})^{\mathbf{1}\{y^{(i)}\neq+1\}}$$





Linear logistic classification

aka logistic regression

- How do we learn a classifier (i.e. learn θ, θ_0)?

Probability(data)

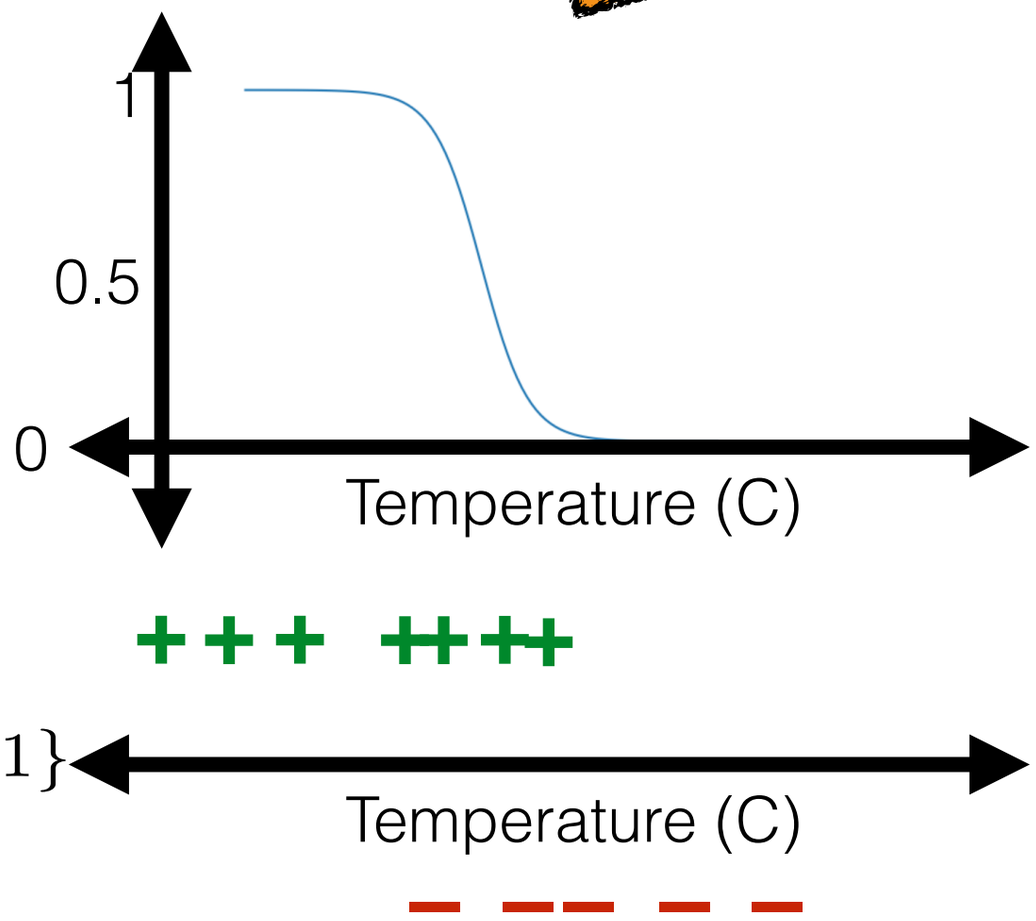
$$= \prod_{i=1}^{n} \text{Probability}(\text{data point } i)$$

[Let $g^{(i)} = \sigma(\theta^\top x^{(i)} + \theta_0)$]

$$= \prod_{i=1}^{n} \begin{cases} g^{(i)} \text{ if } y^{(i)} = +1 \\ (1 - g^{(i)}) \text{ else} \end{cases}$$

$$= \prod_{i=1}^{n} (g^{(i)})^{\mathbf{1}\{y^{(i)}=+1\}} (1 - g^{(i)})^{\mathbf{1}\{y^{(i)}\neq+1\}}$$

log probability(data)



Linear logistic classification

aka logistic regression

- How do we learn a classifier (i.e. learn θ, θ_0)?

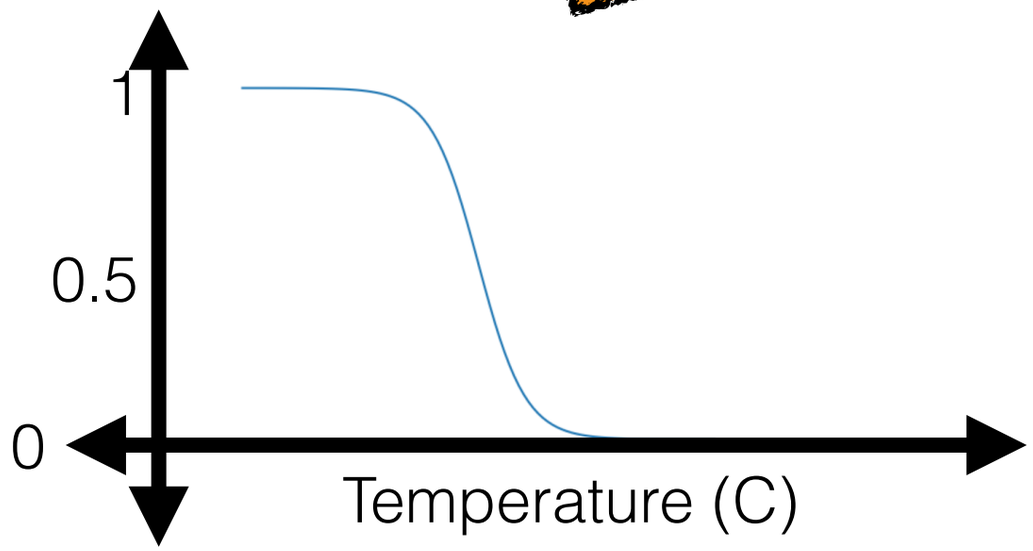
Probability(data)

$$= \prod_{i=1}^{n} \text{Probability}(\text{data point } i)$$

[Let $g^{(i)} = \sigma(\theta^\top x^{(i)} + \theta_0)$]

$$= \prod_{i=1}^{n} \begin{cases} g^{(i)} \text{ if } y^{(i)} = +1 \\ (1 - g^{(i)}) \text{ else} \end{cases}$$

$$= \prod_{i=1}^{n} (g^{(i)})^{\mathbf{1}\{y^{(i)}=+1\}} (1 - g^{(i)})^{\mathbf{1}\{y^{(i)}\neq+1\}}$$



Loss(data) = - log probability(data)

Linear logistic classification

aka logistic regression

- How do we learn a classifier (i.e. learn θ, θ_0)?

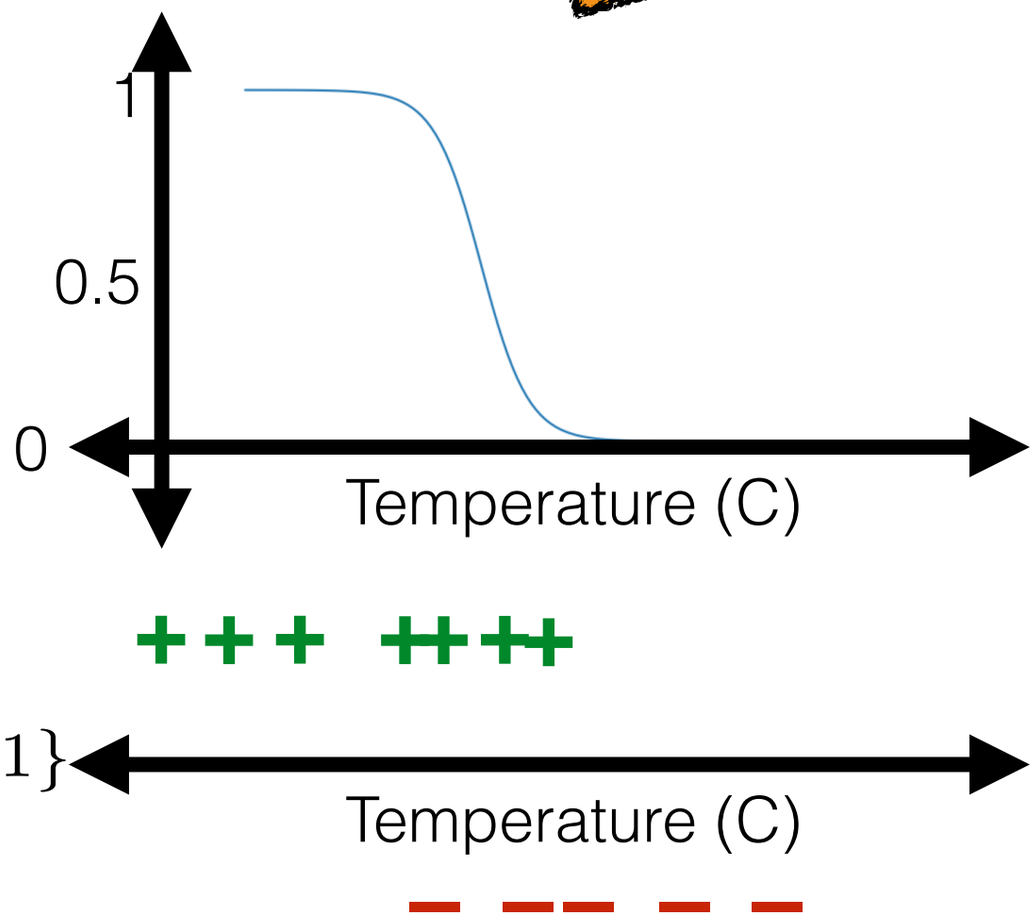
Probability(data)

$$= \prod_{i=1}^{n} \text{Probability}(\text{data point } i)$$

$$[\text{Let } g^{(i)} = \sigma(\theta^\top x^{(i)} + \theta_0)\,]$$

$$= \prod_{i=1}^{n} \begin{cases} g^{(i)} \text{ if } y^{(i)} = +1 \\ (1 - g^{(i)}) \text{ else} \end{cases}$$

$$= \prod_{i=1}^{n} (g^{(i)})^{\mathbf{1}\{y^{(i)}=+1\}} (1 - g^{(i)})^{\mathbf{1}\{y^{(i)}\neq+1\}}$$



Loss(data) = - log probability(data)

$$= \sum_{i=1}^{n} -\left(\mathbf{1}\{y^{(i)} = +1\} \log g^{(i)} + \mathbf{1}\{y^{(i)} \neq +1\} \log(1 - g^{(i)})\right.$$

Linear logistic classification

aka logistic regression

- How do we learn a classifier (i.e. learn θ, θ_0)?

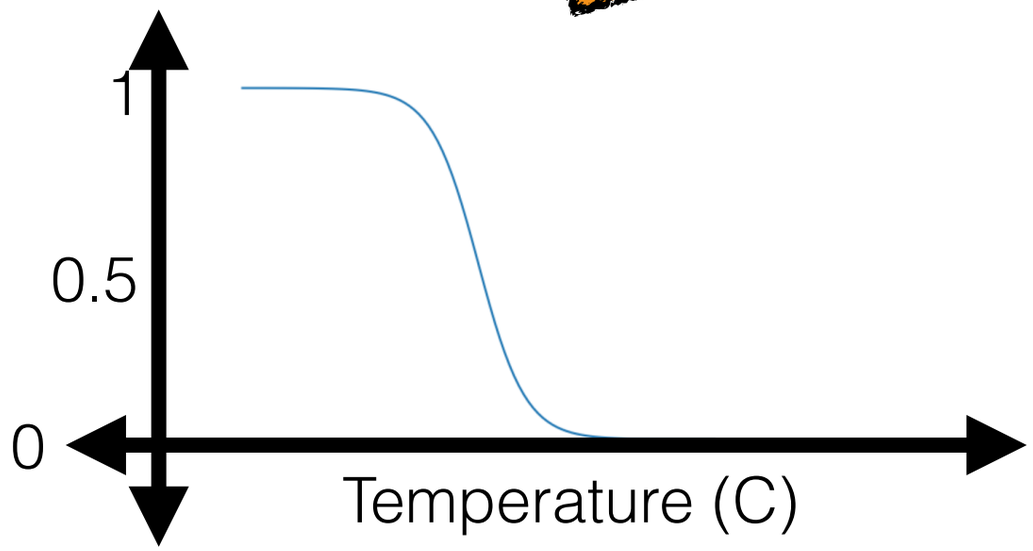
Probability(data)

$$= \prod_{i=1}^{n} \text{Probability}(\text{data point } i)$$

$$[\text{Let } g^{(i)} = \sigma(\theta^\top x^{(i)} + \theta_0)\,]$$

$$= \prod_{i=1}^{n} \begin{cases} g^{(i)} \text{ if } y^{(i)} = +1 \\ (1 - g^{(i)}) \text{ else} \end{cases}$$

$$= \prod_{i=1}^{n} (g^{(i)})^{\mathbf{1}\{y^{(i)}=+1\}} (1 - g^{(i)})^{\mathbf{1}\{y^{(i)}\neq+1\}}$$



Loss(data) = - log probability(data)

$$= \sum_{i=1}^{n} - \left(\mathbf{1}\{y^{(i)} = +1\} \log g^{(i)} + \mathbf{1}\{y^{(i)} \neq +1\} \log(1 - g^{(i)}) \right.$$

Linear logistic classification

aka logistic regression

- How do we learn a classifier (i.e. learn θ, θ_0)?

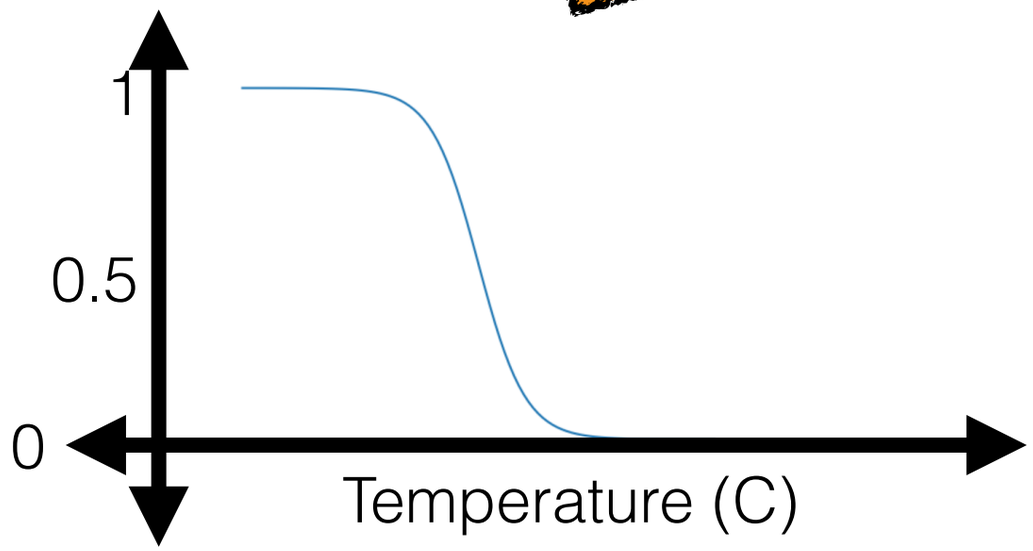
Probability(data)

$$= \prod_{i=1}^{n} \text{Probability}(\text{data point } i)$$

$$[\text{Let } g^{(i)} = \sigma(\theta^\top x^{(i)} + \theta_0)\,]$$

$$= \prod_{i=1}^{n} \begin{cases} g^{(i)} \text{ if } y^{(i)} = +1 \\ (1 - g^{(i)}) \text{ else} \end{cases}$$

$$= \prod_{i=1}^{n} (g^{(i)})^{\mathbf{1}\{y^{(i)}=+1\}} (1 - g^{(i)})^{\mathbf{1}\{y^{(i)}\neq+1\}}$$



Loss(data) = - log probability(data)

$$= \sum_{i=1}^{n} - \left(\mathbf{1}\{y^{(i)} = +1\} \log g^{(i)} + \mathbf{1}\{y^{(i)} \neq +1\} \log(1 - g^{(i)}) \right.$$

Linear logistic classification

aka logistic regression

- How do we learn a classifier (i.e. learn θ, θ_0)?

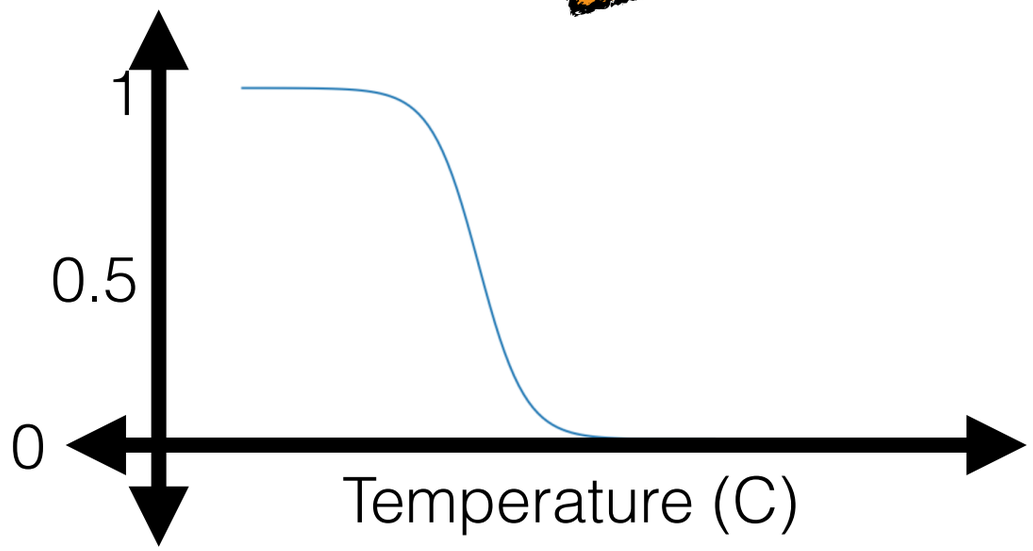
Probability(data)

$$= \prod_{i=1}^{n} \text{Probability}(\text{data point } i)$$

$$[\text{Let } g^{(i)} = \sigma(\theta^\top x^{(i)} + \theta_0)\,]$$

$$= \prod_{i=1}^{n} \begin{cases} g^{(i)} \text{ if } y^{(i)} = +1 \\ (1 - g^{(i)}) \text{ else} \end{cases}$$

$$= \prod_{i=1}^{n} (g^{(i)})^{\mathbf{1}\{y^{(i)}=+1\}} (1 - g^{(i)})^{\mathbf{1}\{y^{(i)}\neq+1\}}$$



Loss(data) = - log probability(data)

$$= \sum_{i=1}^{n} - \left(\mathbf{1}\{y^{(i)} = +1\} \log g^{(i)} + \mathbf{1}\{y^{(i)} \neq +1\} \log(1 - g^{(i)}) \right.$$

Linear logistic classification

aka logistic regression

- How do we learn a classifier (i.e. learn θ, θ_0)?

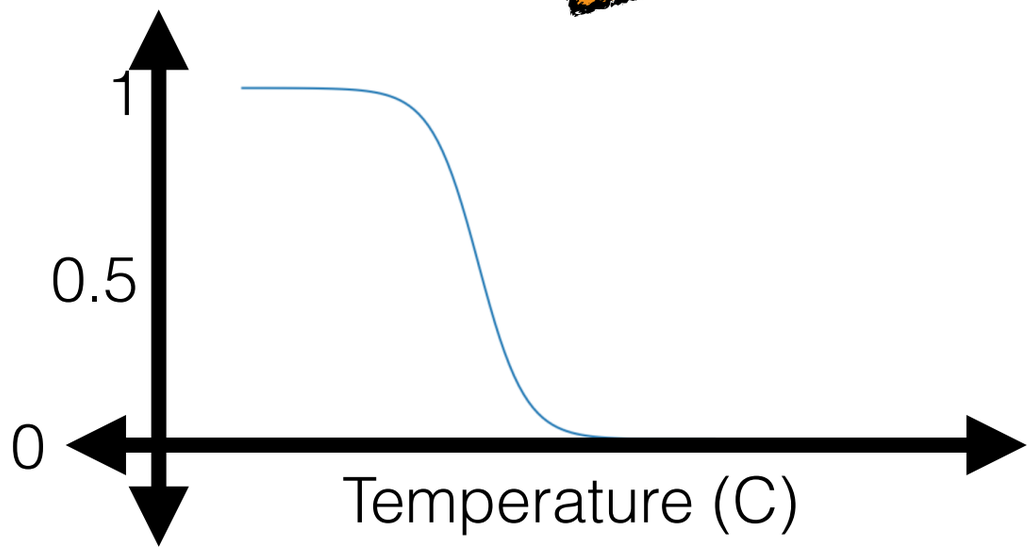
Probability(data)

$$= \prod_{i=1}^{n} \text{Probability}(\text{data point } i)$$

$$[\text{Let } g^{(i)} = \sigma(\theta^\top x^{(i)} + \theta_0)\,]$$

$$= \prod_{i=1}^{n} \begin{cases} g^{(i)} \text{ if } y^{(i)} = +1 \\ (1 - g^{(i)}) \text{ else} \end{cases}$$

$$= \prod_{i=1}^{n} (g^{(i)})^{\mathbf{1}\{y^{(i)}=+1\}} (1 - g^{(i)})^{\mathbf{1}\{y^{(i)}\neq+1\}}$$



Loss(data) = - log probability(data)

$$= \sum_{i=1}^{n} -\left(\mathbf{1}\{y^{(i)} = +1\} \log g^{(i)} + \mathbf{1}\{y^{(i)} \neq +1\} \log(1 - g^{(i)})\right.$$

Linear logistic classification

aka logistic regression

- How do we learn a classifier (i.e. learn θ, θ_0)?

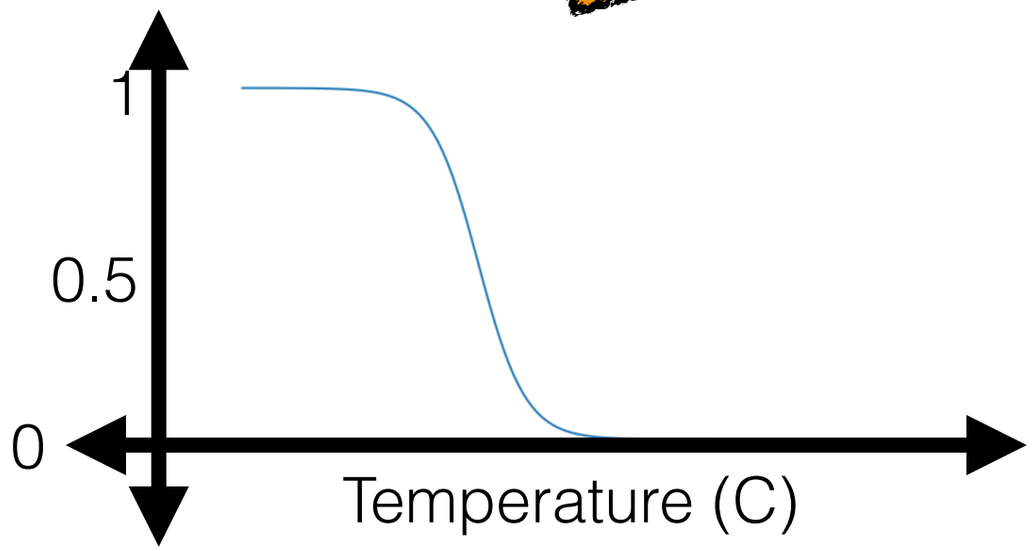
Probability(data)

$$= \prod_{i=1}^{n} \text{Probability}(\text{data point } i)$$

$$[\text{Let } g^{(i)} = \sigma(\theta^\top x^{(i)} + \theta_0)\,]$$

$$= \prod_{i=1}^{n} \begin{cases} g^{(i)} \text{ if } y^{(i)} = +1 \\ (1 - g^{(i)}) \text{ else} \end{cases}$$

$$= \prod_{i=1}^{n} (g^{(i)})^{\mathbf{1}\{y^{(i)}=+1\}} (1 - g^{(i)})^{\mathbf{1}\{y^{(i)}\neq+1\}}$$



Loss(data) = - log probability(data)

$$= \frac{1}{n} \sum_{i=1}^{n} - \left(\mathbf{1}\{y^{(i)} = +1\} \log g^{(i)} + \mathbf{1}\{y^{(i)} \neq +1\} \log(1 - g^{(i)}) \right.$$

Linear logistic classification

aka logistic regression

- How do we learn a classifier (i.e. learn θ, θ_0)?

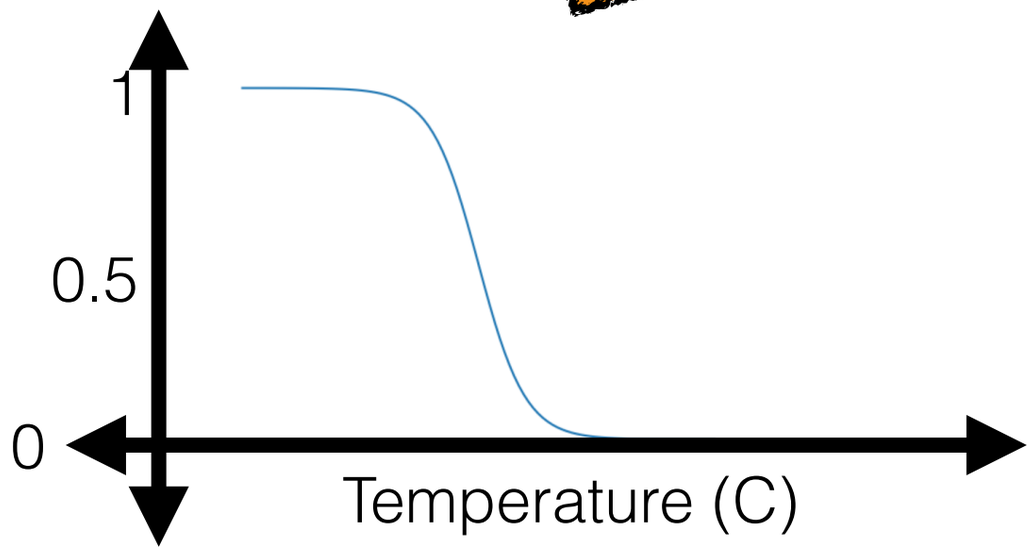
Probability(data)

$$= \prod_{i=1}^{n} \text{Probability}(\text{data point } i)$$

$$[\text{Let } g^{(i)} = \sigma(\theta^\top x^{(i)} + \theta_0)\,]$$

$$= \prod_{i=1}^{n} \begin{cases} g^{(i)} \text{ if } y^{(i)} = +1 \\ (1 - g^{(i)}) \text{ else} \end{cases}$$

$$= \prod_{i=1}^{n} (g^{(i)})^{\mathbf{1}\{y^{(i)}=+1\}} (1 - g^{(i)})^{\mathbf{1}\{y^{(i)}\neq+1\}}$$



Loss(data) = -(1/n) * log probability(data)

$$= \frac{1}{n} \sum_{i=1}^{n} - \left(\mathbf{1}\{y^{(i)} = +1\} \log g^{(i)} + \mathbf{1}\{y^{(i)} \neq +1\} \log(1 - g^{(i)}) \right.$$

Linear logistic classification

aka logistic regression

- How do we learn a classifier (i.e. learn θ, θ_0)?

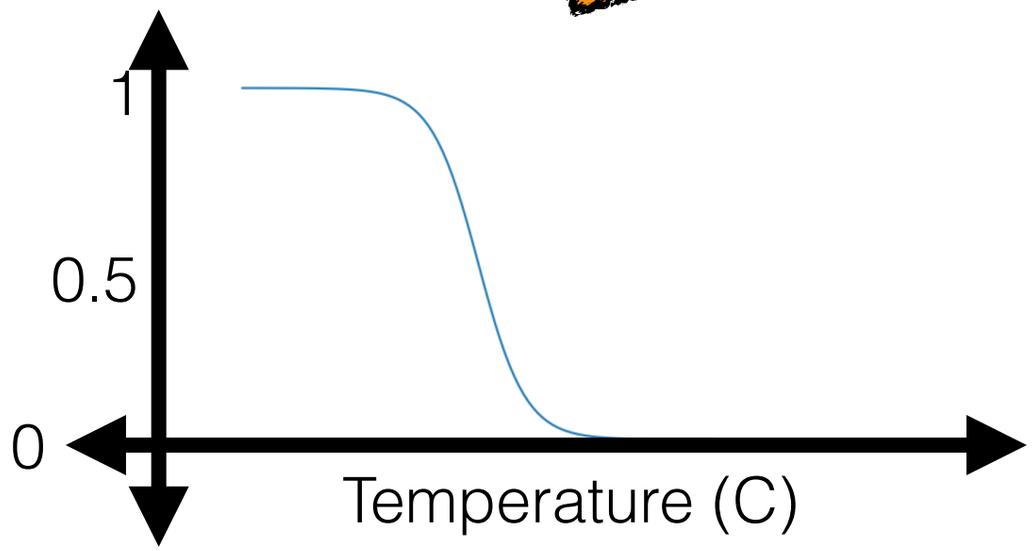
Probability(data)

$$= \prod_{i=1}^{n} \text{Probability}(\text{data point } i)$$

$$[\text{Let } g^{(i)} = \sigma(\theta^\top x^{(i)} + \theta_0)\,]$$

$$= \prod_{i=1}^{n} \begin{cases} g^{(i)} \text{ if } y^{(i)} = +1 \\ (1 - g^{(i)}) \text{ else} \end{cases}$$

$$= \prod_{i=1}^{n} (g^{(i)})^{\mathbf{1}\{y^{(i)}=+1\}} (1 - g^{(i)})^{\mathbf{1}\{y^{(i)}\neq+1\}}$$



Loss(data) = -(1/n) * log probability(data)

$$= \frac{1}{n} \sum_{i=1}^{n} -\left(\mathbf{1}\{y^{(i)} = +1\} \log g^{(i)} + \mathbf{1}\{y^{(i)} \neq +1\} \log(1 - g^{(i)}) \right)$$

Linear logistic classification

aka logistic regression

- How do we learn a classifier (i.e. learn θ, θ_0)?

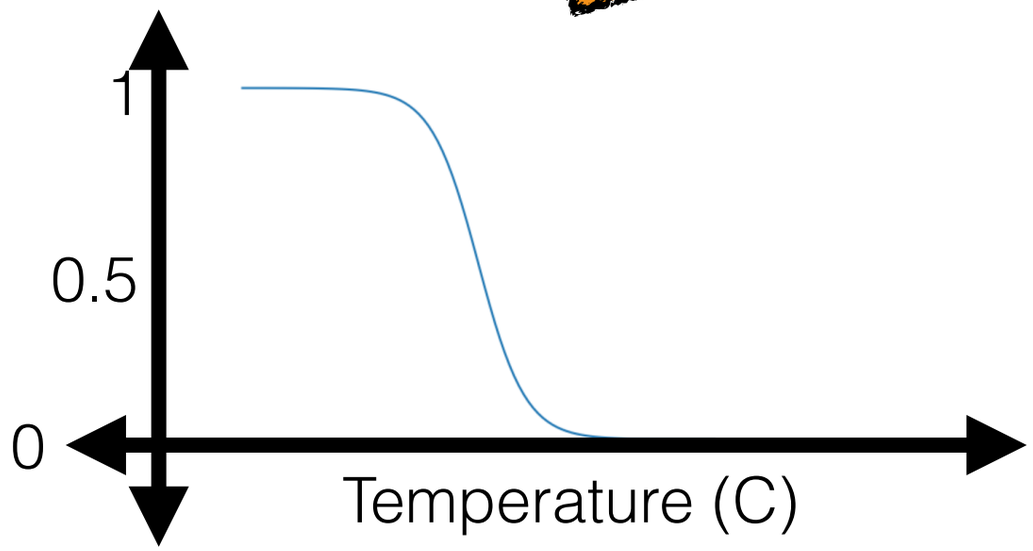
Probability(data)

$$= \prod_{i=1}^{n} \text{Probability}(\text{data point } i)$$

$$[\text{Let } g^{(i)} = \sigma(\theta^\top x^{(i)} + \theta_0)\,]$$

$$= \prod_{i=1}^{n} \begin{cases} g^{(i)} \text{ if } y^{(i)} = +1 \\ (1 - g^{(i)}) \text{ else} \end{cases}$$

$$= \prod_{i=1}^{n} (g^{(i)})^{\mathbf{1}\{y^{(i)}=+1\}} (1 - g^{(i)})^{\mathbf{1}\{y^{(i)}\neq+1\}}$$



Loss(data) = -(1/n) * log probability(data)

$$= \frac{1}{n} \sum_{i=1}^{n} -\left(\mathbf{1}\{y^{(i)} = +1\} \log g^{(i)} + \mathbf{1}\{y^{(i)} \neq +1\} \log(1 - g^{(i)})\right)$$

Negative log likelihood loss (*g* for guess, *a* for actual):

Linear logistic classification

aka logistic regression

- How do we learn a classifier (i.e. learn θ, θ_0)?

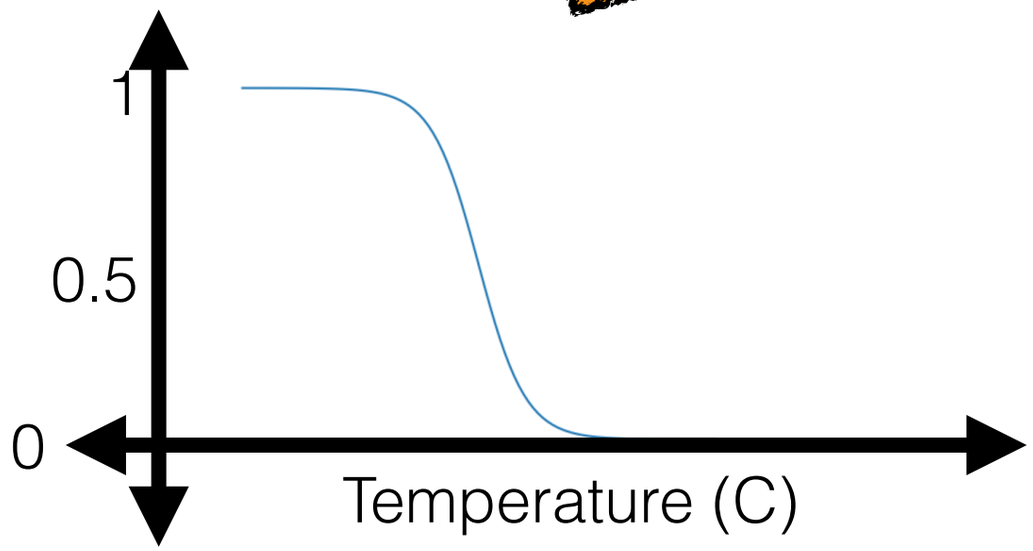
Probability(data)

$$= \prod_{i=1}^{n} \text{Probability}(\text{data point } i)$$

$$[\text{Let } g^{(i)} = \sigma(\theta^\top x^{(i)} + \theta_0)\,]$$

$$= \prod_{i=1}^{n} \begin{cases} g^{(i)} \text{ if } y^{(i)} = +1 \\ (1 - g^{(i)}) \text{ else} \end{cases}$$

$$= \prod_{i=1}^{n} (g^{(i)})^{\mathbf{1}\{y^{(i)}=+1\}} (1 - g^{(i)})^{\mathbf{1}\{y^{(i)}\neq+1\}}$$



Loss(data) = -(1/n) * log probability(data)

$$= \frac{1}{n} \sum_{i=1}^{n} -\left(\mathbf{1}\{y^{(i)} = +1\} \log g^{(i)} + \mathbf{1}\{y^{(i)} \neq +1\} \log(1 - g^{(i)})\right)$$

Negative log likelihood loss (*g* for guess, *a* for actual):

$$-L_{\text{nll}}(g, a) = (1\{a = +1\} \log g + 1\{a \neq +1\} \log(1 - g))$$

Linear logistic classification

- How do we learn a classifier (i.e. learn θ, θ_0)?

aka logistic regression

Linear logistic classification

aka logistic regression

- How do we learn a classifier (i.e. learn θ, θ_0)?
- Want to find parameter values to minimize average (negative log likelihood) loss across the data

Linear logistic classification

aka logistic regression

- How do we learn a classifier (i.e. learn θ, θ_0)?
- Want to find parameter values to minimize average (negative log likelihood) loss across the data

$$\frac{1}{n} \sum_{i=1}^{n} L_{\text{nll}}(\sigma(\theta^\top x^{(i)} + \theta_0), y^{(i)})$$

Linear logistic classification

aka logistic regression

- How do we learn a classifier (i.e. learn θ, θ_0)?
- Want to find parameter values to minimize average (negative log likelihood) loss across the data

$$J_{\text{lr}}(\theta, \theta_0) = \frac{1}{n} \sum_{i=1}^{n} L_{\text{nll}}(\sigma(\theta^\top x^{(i)} + \theta_0), y^{(i)})$$

Linear logistic classification

aka logistic regression

- How do we learn a classifier (i.e. learn θ, θ_0)?
- Want to find parameter values to minimize average (negative log likelihood) loss across the data

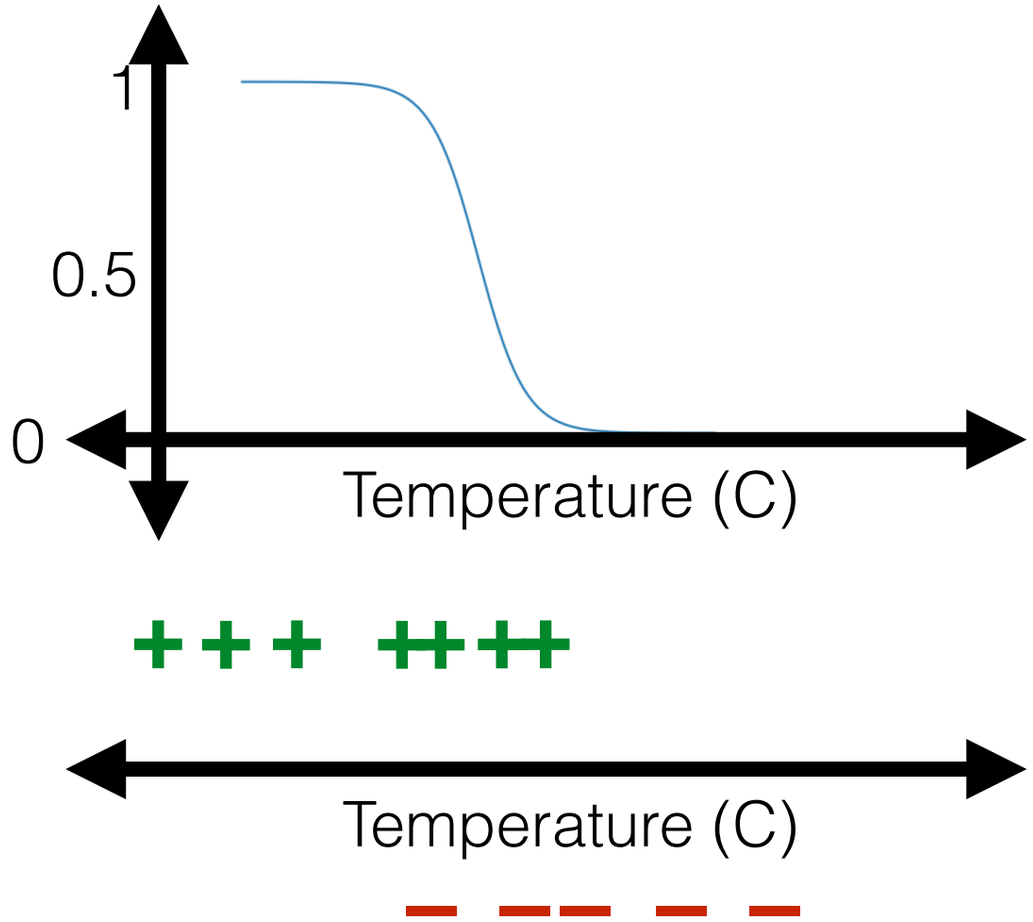
$$J_{\text{lr}}(\Theta) = J_{\text{lr}}(\theta, \theta_0) = \frac{1}{n} \sum_{i=1}^{n} L_{\text{nll}}(\sigma(\theta^\top x^{(i)} + \theta_0), y^{(i)})$$

Linear logistic classification

aka logistic regression

- How do we learn a classifier (i.e. learn θ, θ_0)?
- Want to find parameter values to minimize average (negative log likelihood) loss across the data

$$J_{\text{lr}}(\Theta) = J_{\text{lr}}(\theta, \theta_0) = \frac{1}{n} \sum_{i=1}^{n} L_{\text{nll}}(\sigma(\theta^\top x^{(i)} + \theta_0), y^{(i)})$$

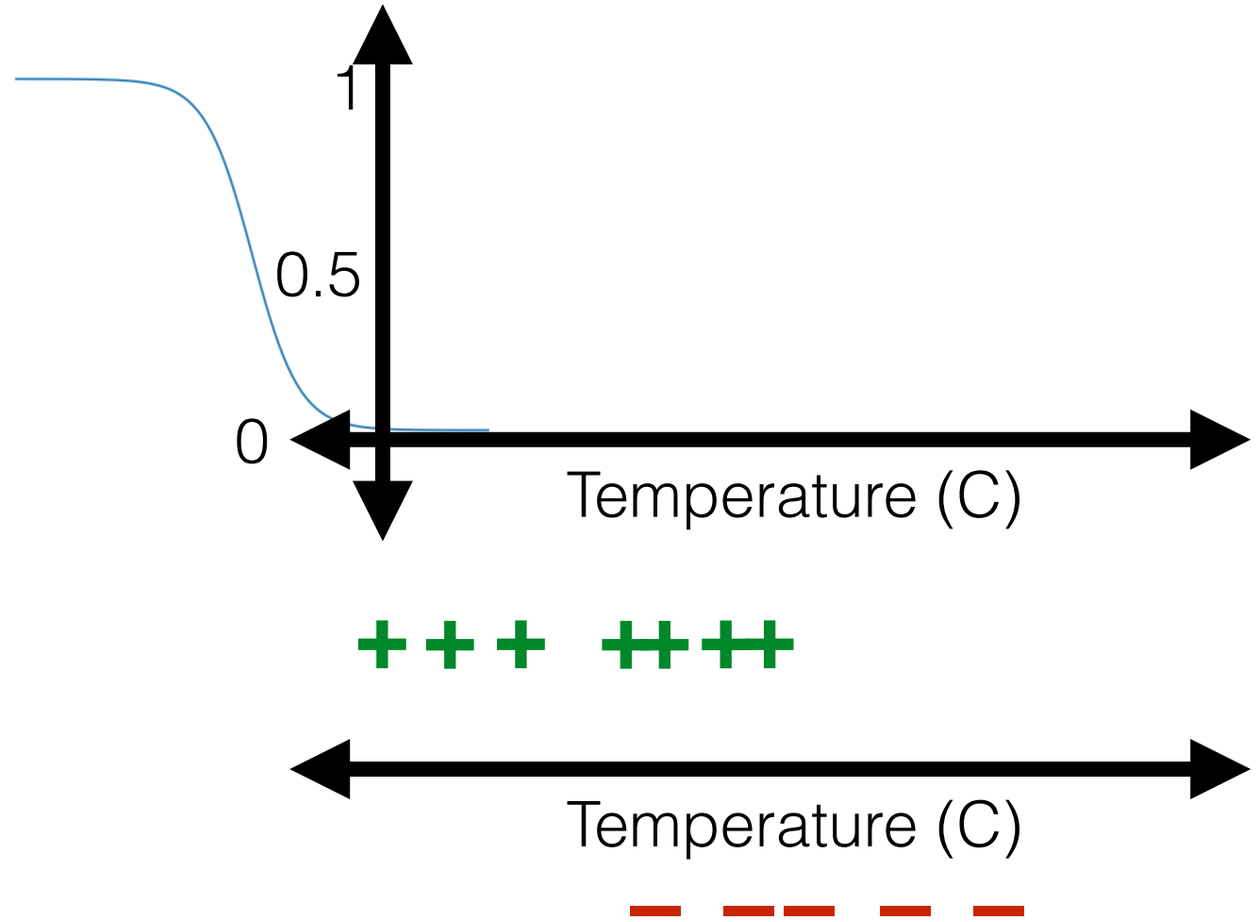


Linear logistic classification

aka logistic regression

- How do we learn a classifier (i.e. learn θ, θ_0)?
- Want to find parameter values to minimize average (negative log likelihood) loss across the data

$$J_{\text{lr}}(\Theta) = J_{\text{lr}}(\theta, \theta_0) = \frac{1}{n} \sum_{i=1}^{n} L_{\text{nll}}(\sigma(\theta^\top x^{(i)} + \theta_0), y^{(i)})$$

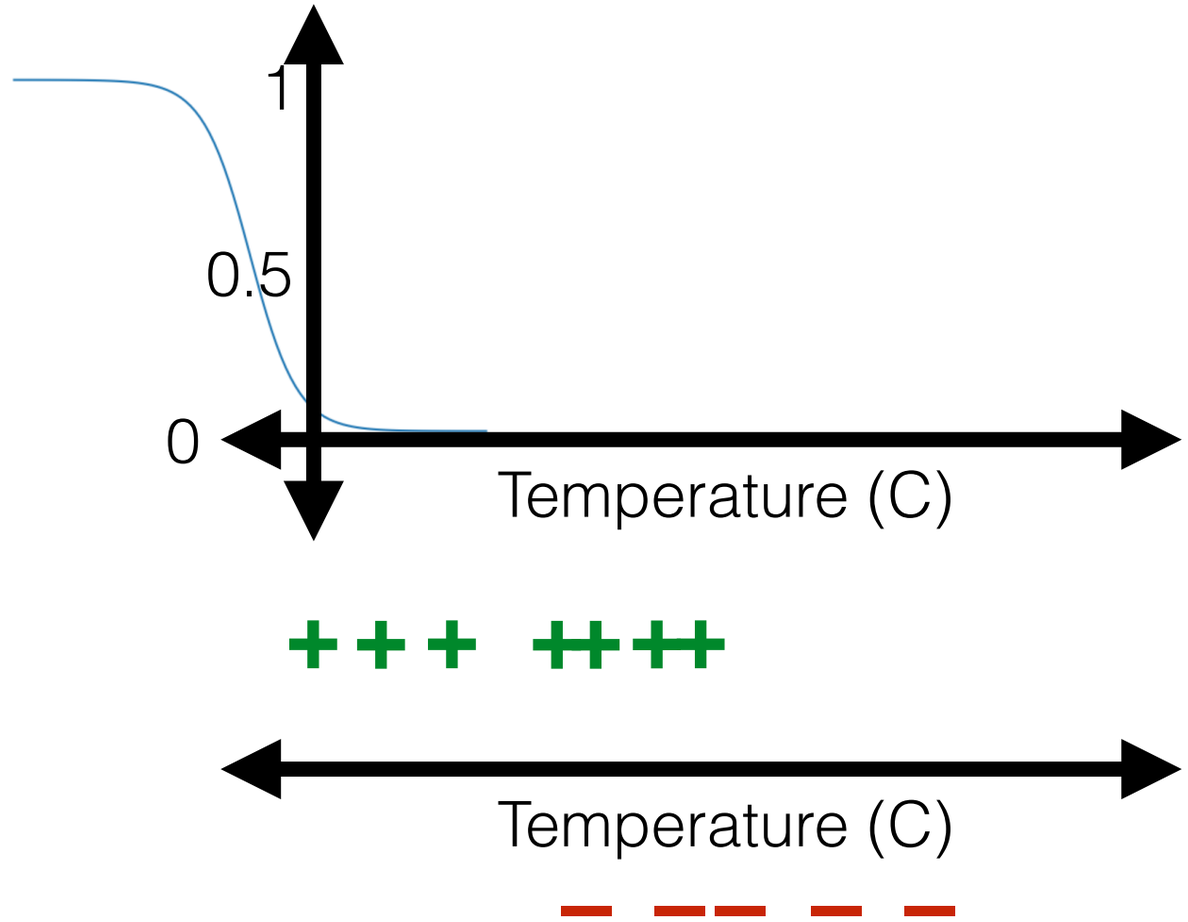


Linear logistic classification

aka logistic regression

- How do we learn a classifier (i.e. learn θ, θ_0)?
- Want to find parameter values to minimize average (negative log likelihood) loss across the data

$$J_{\text{lr}}(\Theta) = J_{\text{lr}}(\theta, \theta_0) = \frac{1}{n} \sum_{i=1}^{n} L_{\text{nll}}(\sigma(\theta^\top x^{(i)} + \theta_0), y^{(i)})$$

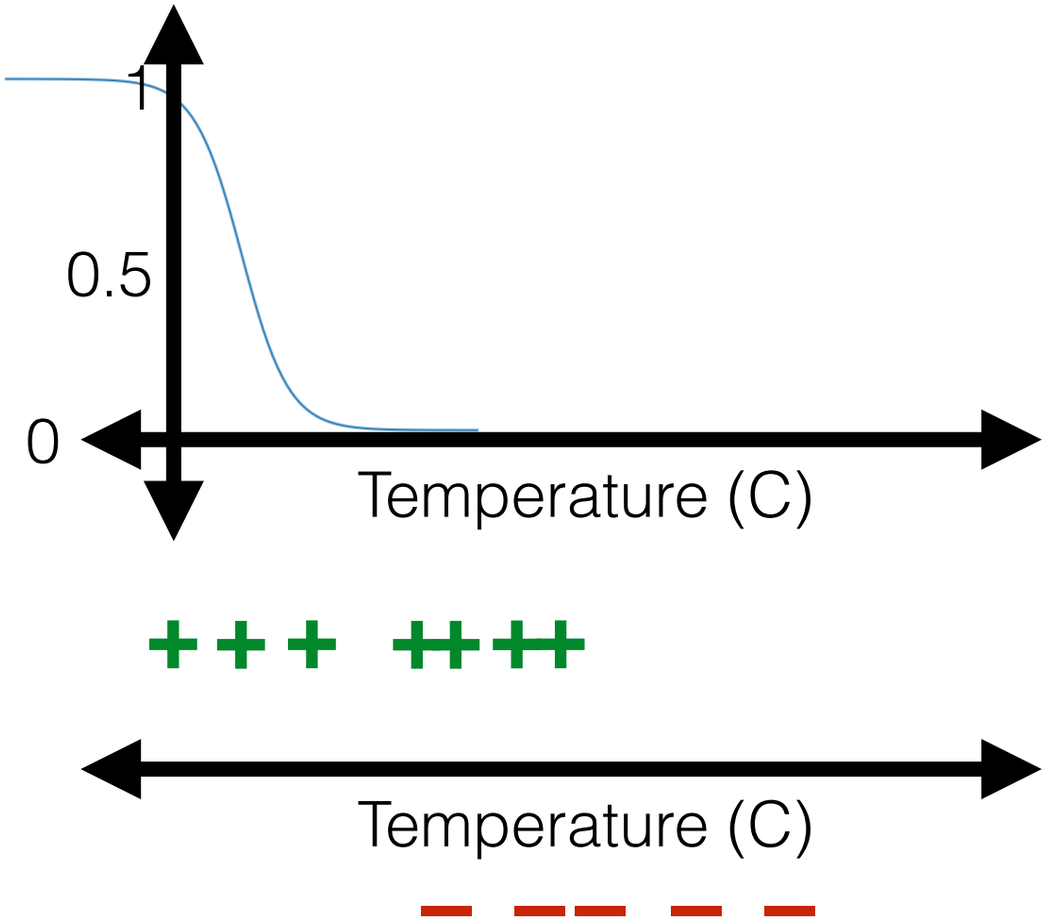


Linear logistic classification

aka logistic regression

- How do we learn a classifier (i.e. learn θ, θ_0)?
- Want to find parameter values to minimize average (negative log likelihood) loss across the data

$$J_{\text{lr}}(\Theta) = J_{\text{lr}}(\theta, \theta_0) = \frac{1}{n} \sum_{i=1}^{n} L_{\text{nll}}(\sigma(\theta^\top x^{(i)} + \theta_0), y^{(i)})$$

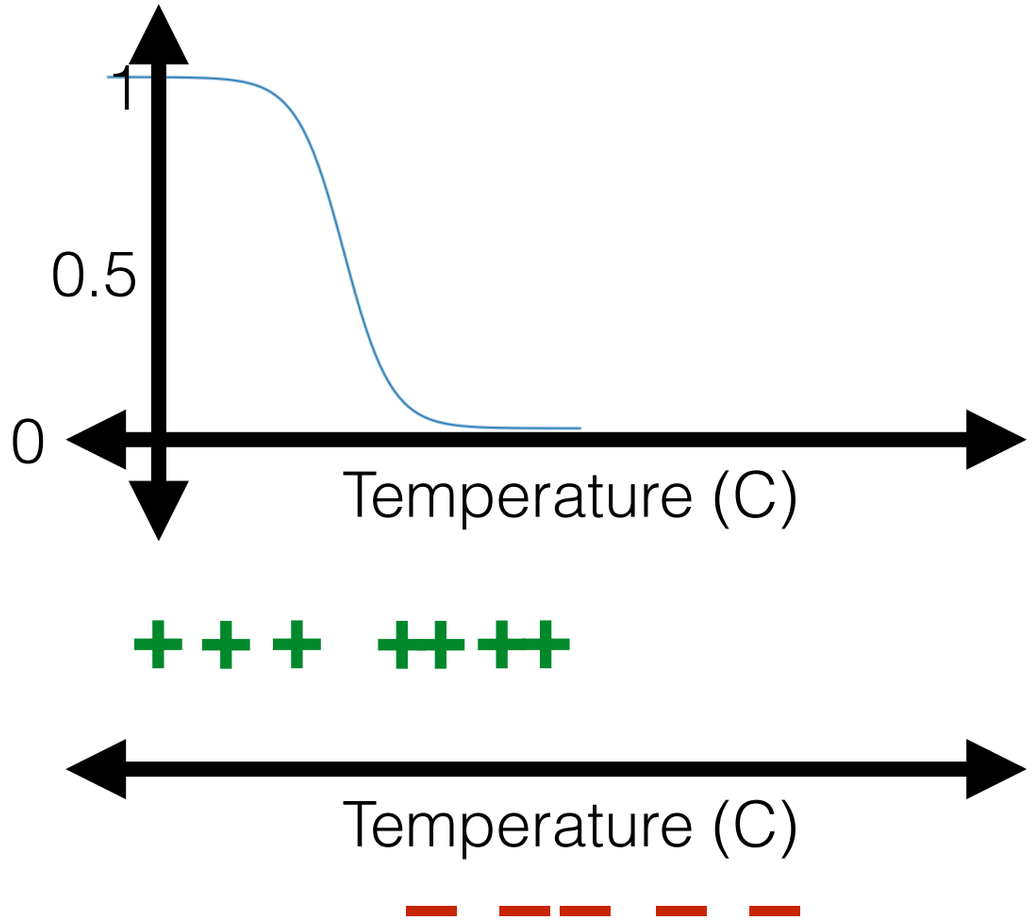


Linear logistic classification

aka logistic regression

- How do we learn a classifier (i.e. learn θ, θ_0)?
- Want to find parameter values to minimize average (negative log likelihood) loss across the data

$$J_{\text{lr}}(\Theta) = J_{\text{lr}}(\theta, \theta_0) = \frac{1}{n} \sum_{i=1}^{n} L_{\text{nll}}(\sigma(\theta^\top x^{(i)} + \theta_0), y^{(i)})$$

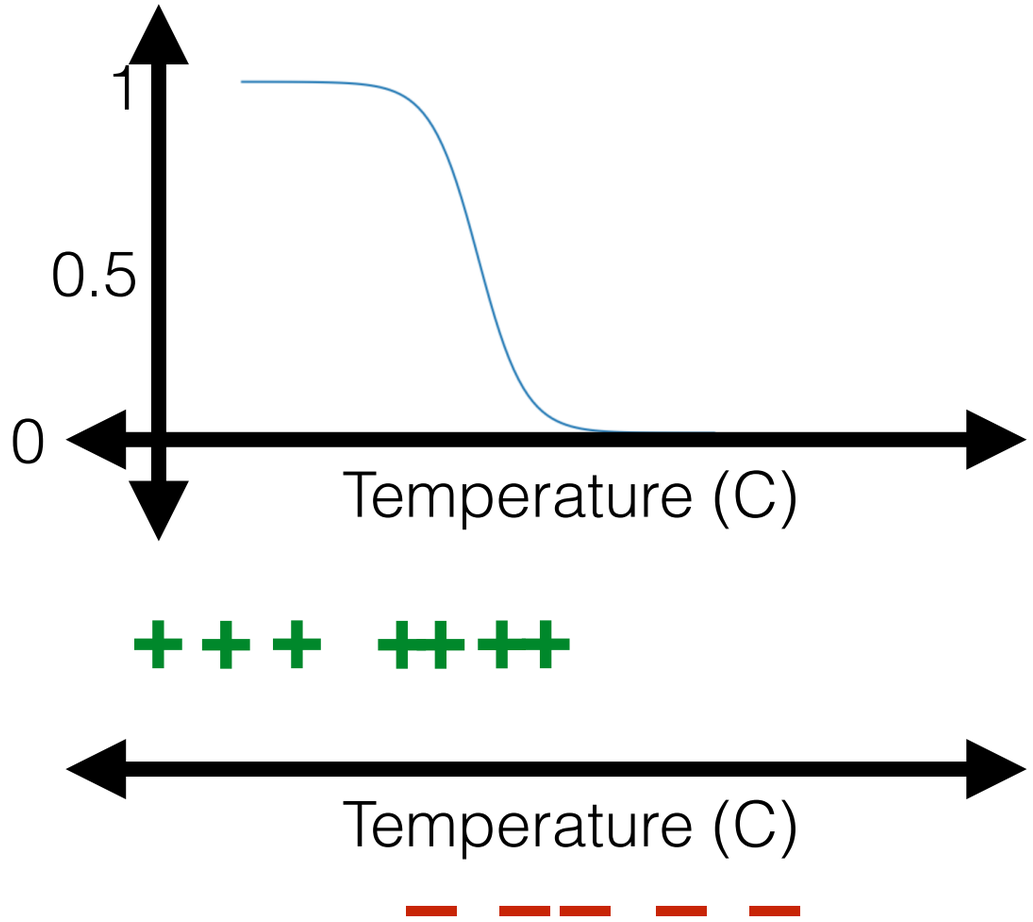


Linear logistic classification

aka logistic regression

- How do we learn a classifier (i.e. learn θ, θ_0)?
- Want to find parameter values to minimize average (negative log likelihood) loss across the data

$$J_{\text{lr}}(\Theta) = J_{\text{lr}}(\theta, \theta_0) = \frac{1}{n} \sum_{i=1}^{n} L_{\text{nll}}(\sigma(\theta^\top x^{(i)} + \theta_0), y^{(i)})$$

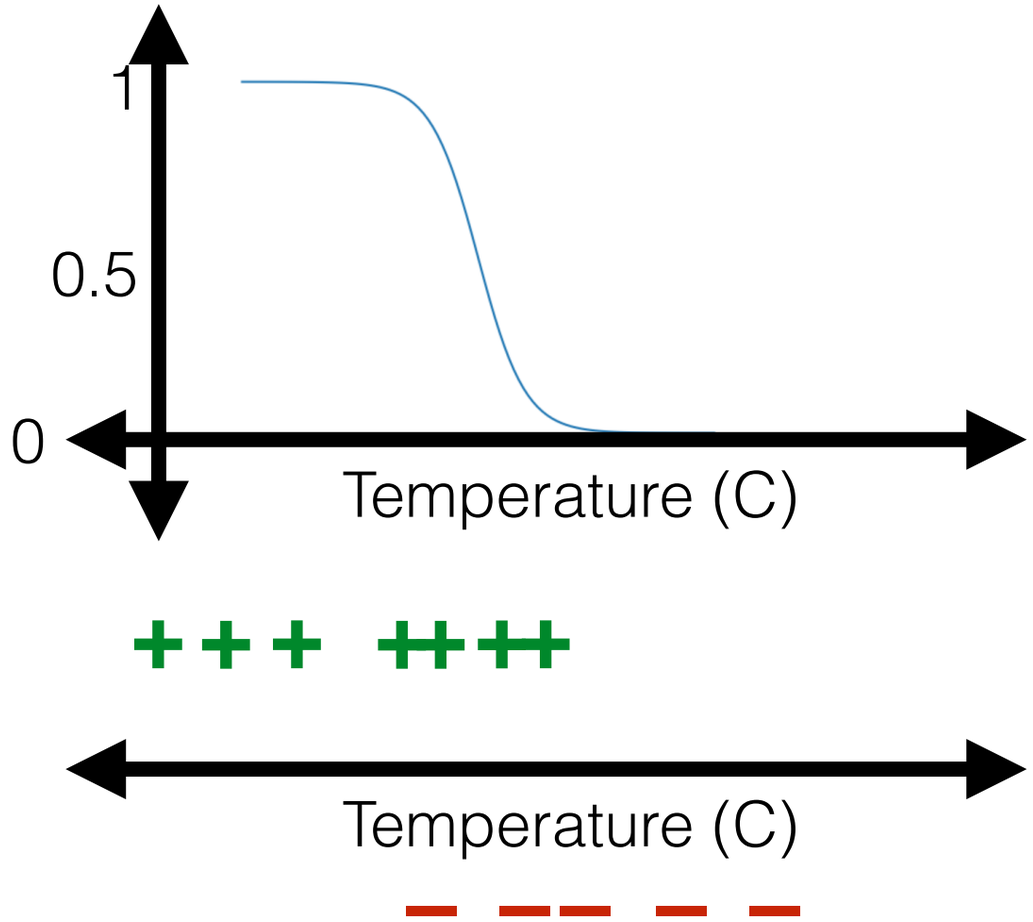


Linear logistic classification

aka logistic regression

- How do we learn a classifier (i.e. learn θ, θ_0)?
- Want to find parameter values to minimize average (negative log likelihood) loss across the data

$$J_{\text{lr}}(\Theta) = J_{\text{lr}}(\theta, \theta_0) = \frac{1}{n} \sum_{i=1}^{n} L_{\text{nll}}(\sigma(\theta^\top x^{(i)} + \theta_0), y^{(i)})$$



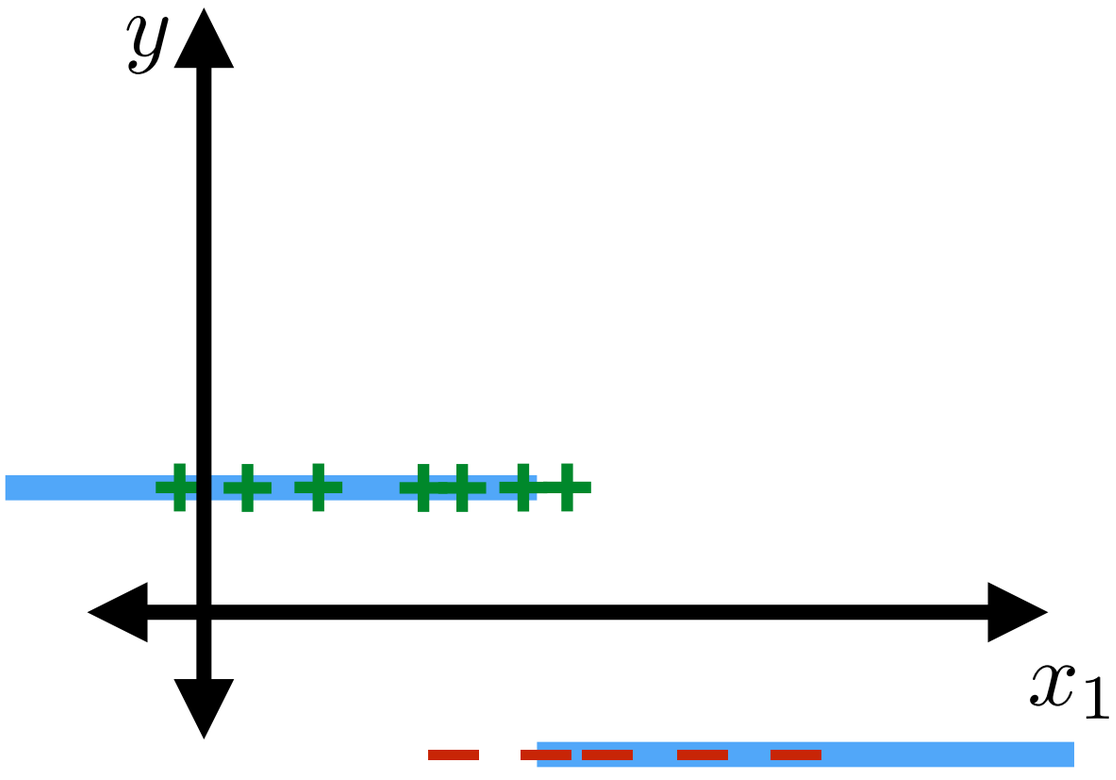
Recall

Classification

- Datum i: feature vector

$x^{(i)} = (x_1^{(i)}, \ldots, x_d^{(i)})^\top \in \mathbb{R}^d$

  - Label $y^{(i)} \in \{-1, +1\}$
- Hypothesis $h : \mathbb{R}^d \to \{-1, +1\}$
- Loss: 0-1, asymmetric, NLL
- Example: linear classification



Compare

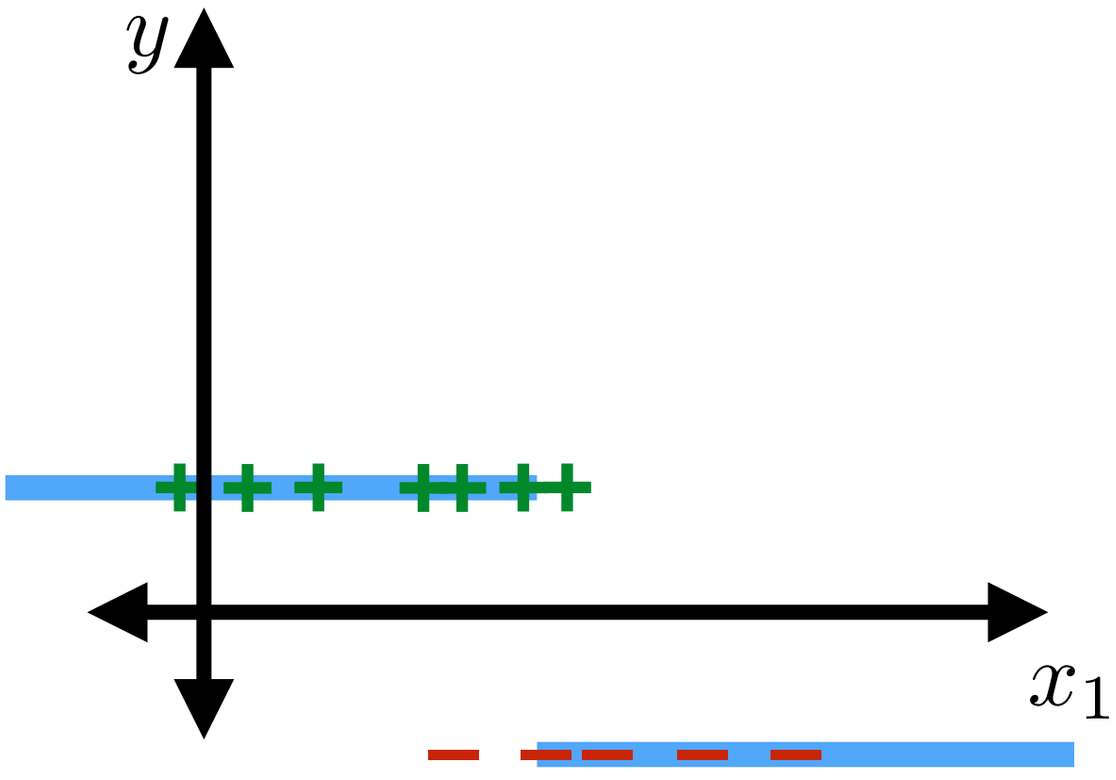
Recall

Classification

- Datum i: feature vector

$x^{(i)} = (x_1^{(i)}, \ldots, x_d^{(i)})^\top \in \mathbb{R}^d$

  - Label $y^{(i)} \in \{-1, +1\}$
- Hypothesis $h : \mathbb{R}^d \to \{-1, +1\}$
- Loss: 0-1, asymmetric, NLL
- Example: linear classification



Compare

Regression

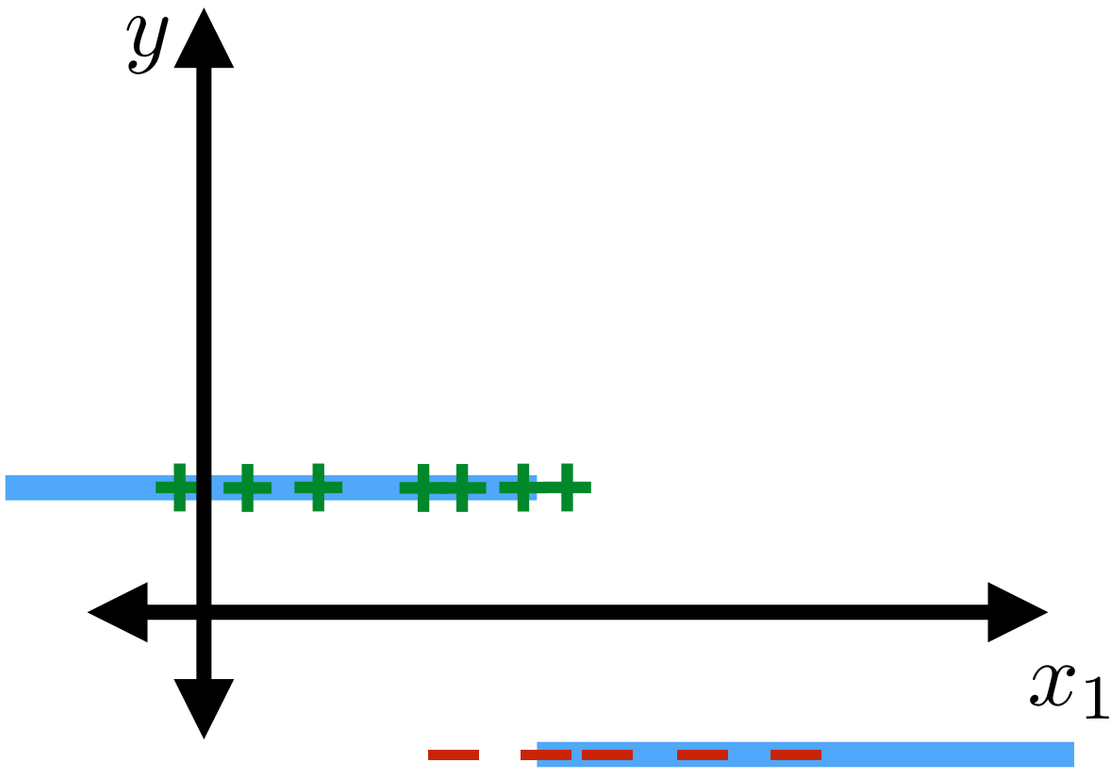
Recall

Classification

- Datum i: feature vector

$x^{(i)} = (x_1^{(i)}, \ldots, x_d^{(i)})^\top \in \mathbb{R}^d$

  - Label $y^{(i)} \in \{-1, +1\}$
- Hypothesis $h : \mathbb{R}^d \to \{-1, +1\}$
- Loss: 0-1, asymmetric, NLL
- Example: linear classification



Compare

Regression

- Datum i :

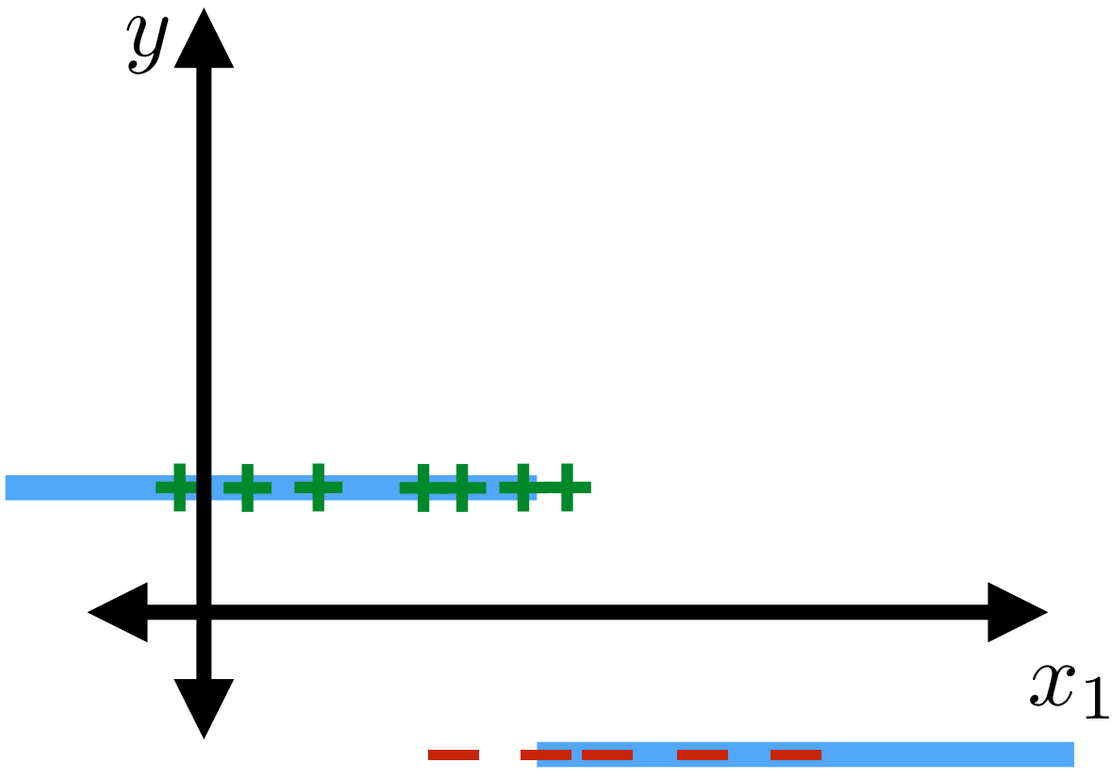
Recall

Classification

- Datum i: feature vector

$x^{(i)} = (x_1^{(i)}, \ldots, x_d^{(i)})^\top \in \mathbb{R}^d$

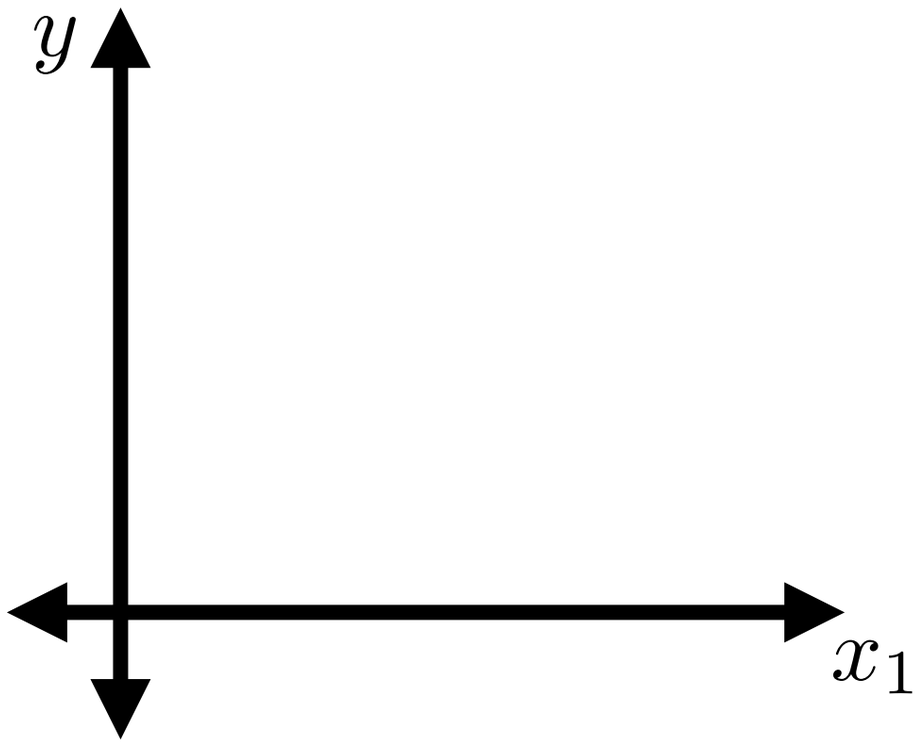
  - Label $y^{(i)} \in \{-1, +1\}$
- Hypothesis $h : \mathbb{R}^d \to \{-1, +1\}$
- Loss: 0-1, asymmetric, NLL
- Example: linear classification



Compare

Regression

- Datum i :



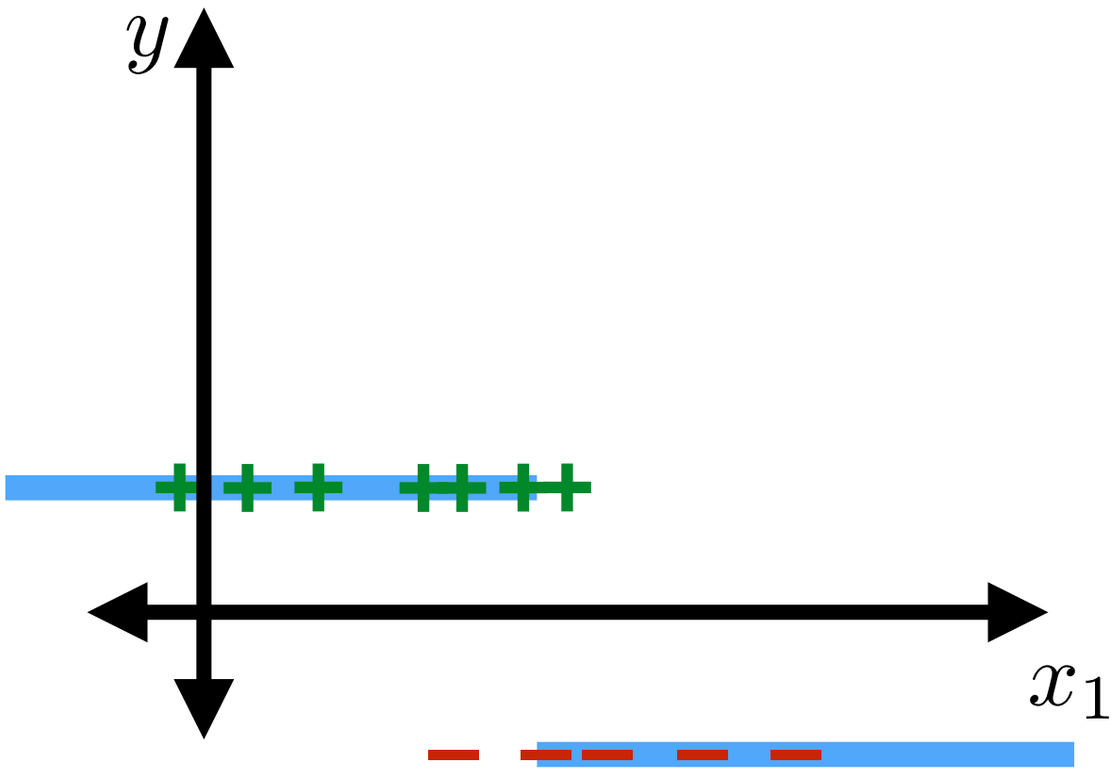
Recall

Classification

- Datum i: feature vector

$x^{(i)} = (x_1^{(i)}, \ldots, x_d^{(i)})^\top \in \mathbb{R}^d$

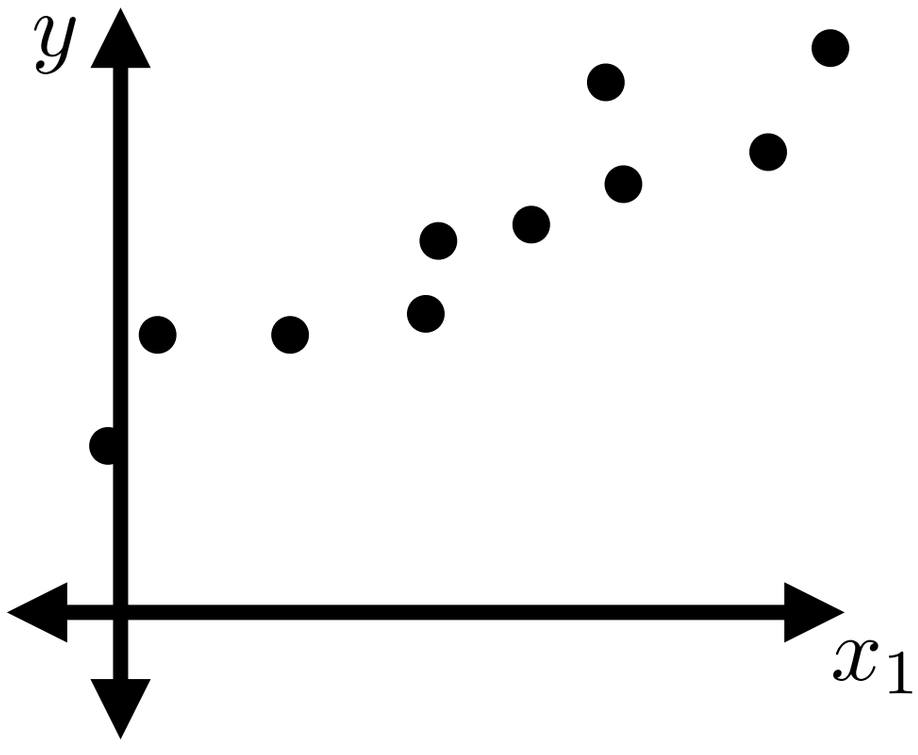
  - Label $y^{(i)} \in \{-1, +1\}$
- Hypothesis $h : \mathbb{R}^d \to \{-1, +1\}$
- Loss: 0-1, asymmetric, NLL
- Example: linear classification



Compare

Regression

- Datum i :



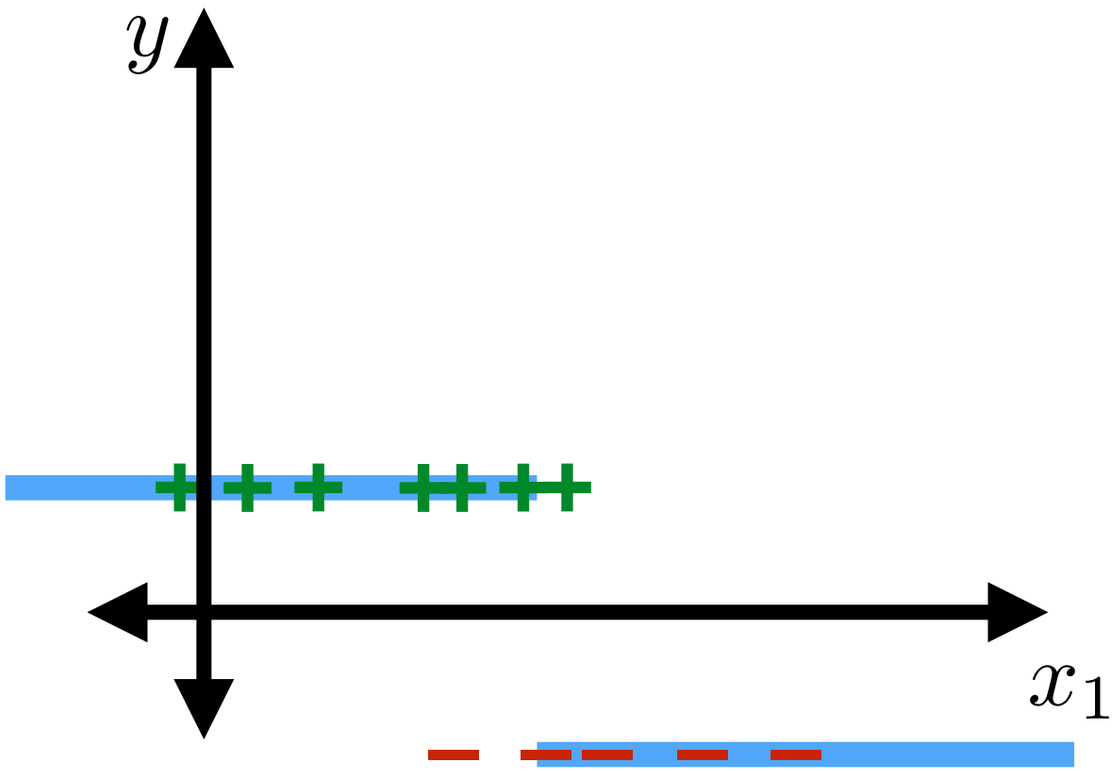
Recall

Classification

- Datum i: feature vector

$$x^{(i)} = (x_1^{(i)}, \ldots, x_d^{(i)})^\top \in \mathbb{R}^d$$

  - Label $y^{(i)} \in \{-1, +1\}$
- Hypothesis $h : \mathbb{R}^d \to \{-1, +1\}$
- Loss: 0-1, asymmetric, NLL
- Example: linear classification



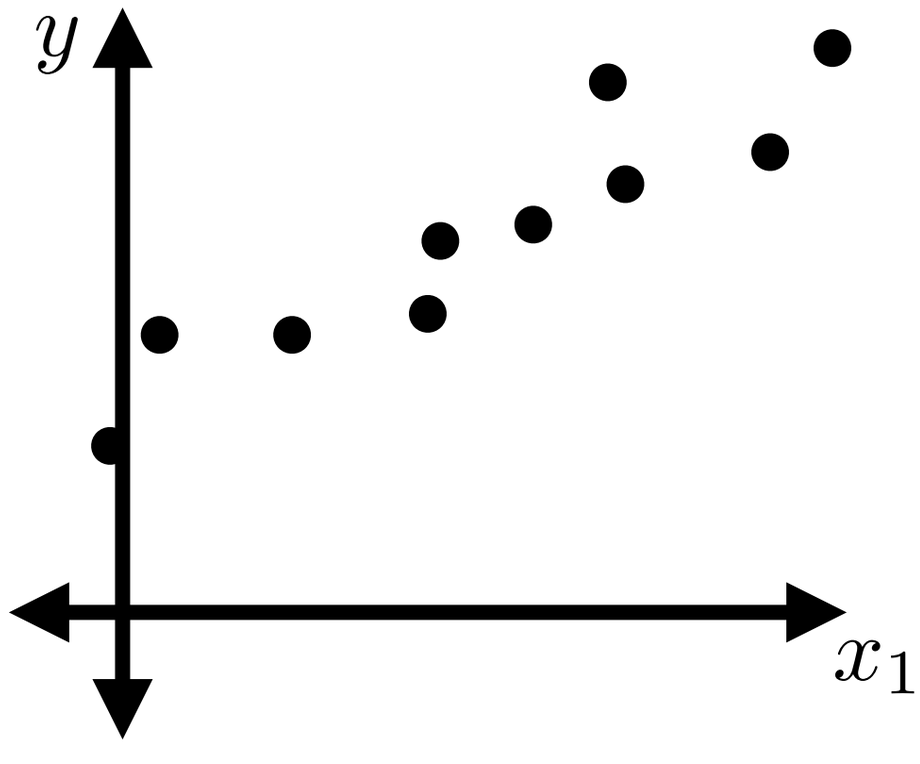
Compare

Regression

- Datum i : feature vector

$$x^{(i)} = (x_1^{(i)}, \ldots, x_d^{(i)})^\top \in \mathbb{R}^d$$

  - Label $y^{(i)} \in \mathbb{R}$



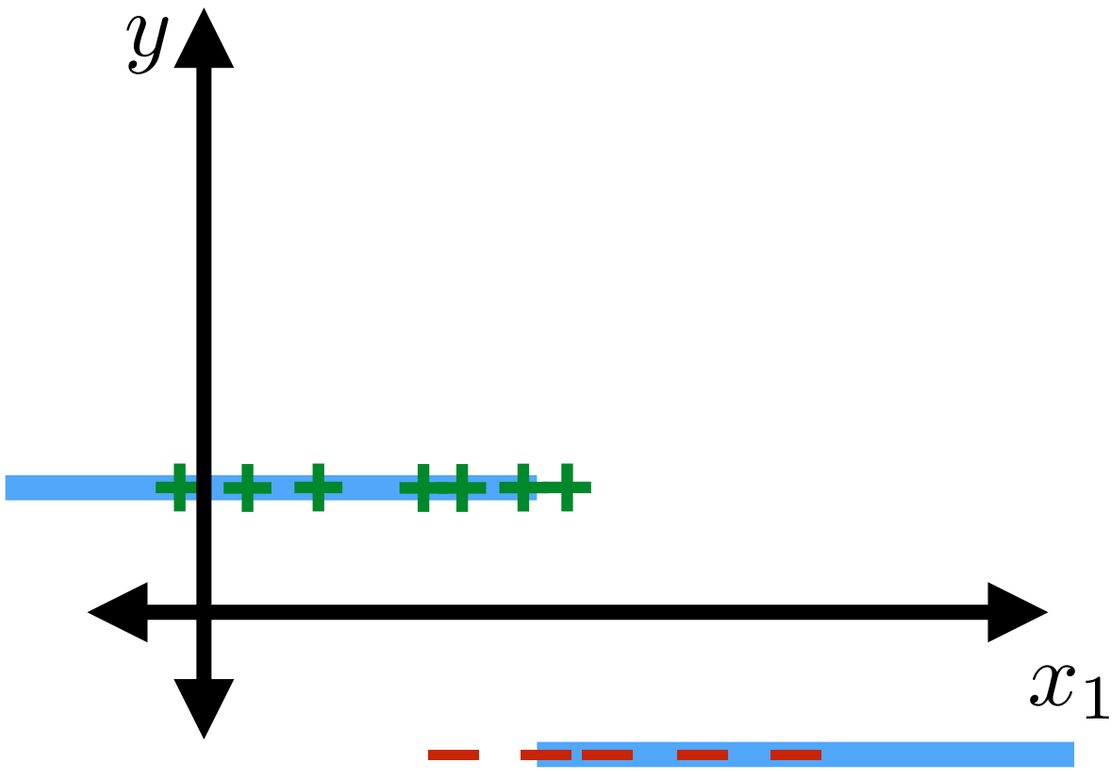
Recall

Classification

- Datum i: feature vector

$$x^{(i)} = (x_1^{(i)}, \ldots, x_d^{(i)})^\top \in \mathbb{R}^d$$

  - Label $y^{(i)} \in \{-1, +1\}$
- Hypothesis $h : \mathbb{R}^d \to \{-1, +1\}$
- Loss: 0-1, asymmetric, NLL
- Example: linear classification



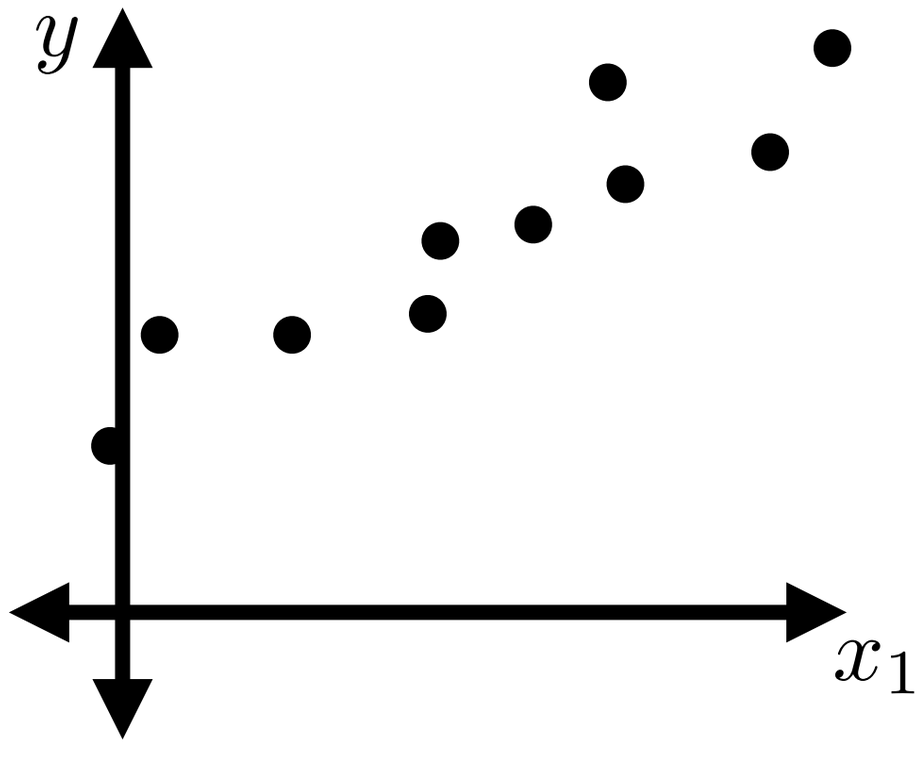
Compare

Regression

- Datum i : feature vector

$$x^{(i)} = (x_1^{(i)}, \ldots, x_d^{(i)})^\top \in \mathbb{R}^d$$

  - Label $y^{(i)} \in \mathbb{R}$
- Hypothesis $h : \mathbb{R}^d \to \mathbb{R}$



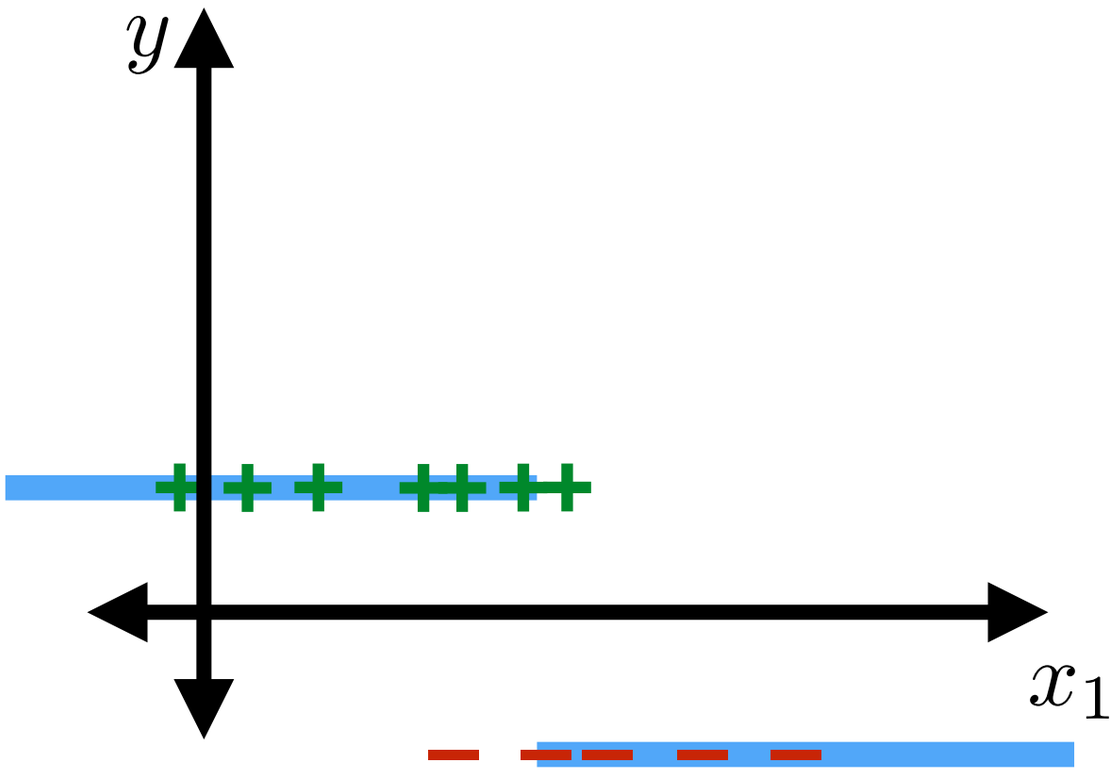
Recall

Classification

- Datum i: feature vector

$$x^{(i)} = (x_1^{(i)}, \ldots, x_d^{(i)})^\top \in \mathbb{R}^d$$

  - Label $y^{(i)} \in \{-1, +1\}$
- Hypothesis $h : \mathbb{R}^d \to \{-1, +1\}$
- Loss: 0-1, asymmetric, NLL
- Example: linear classification



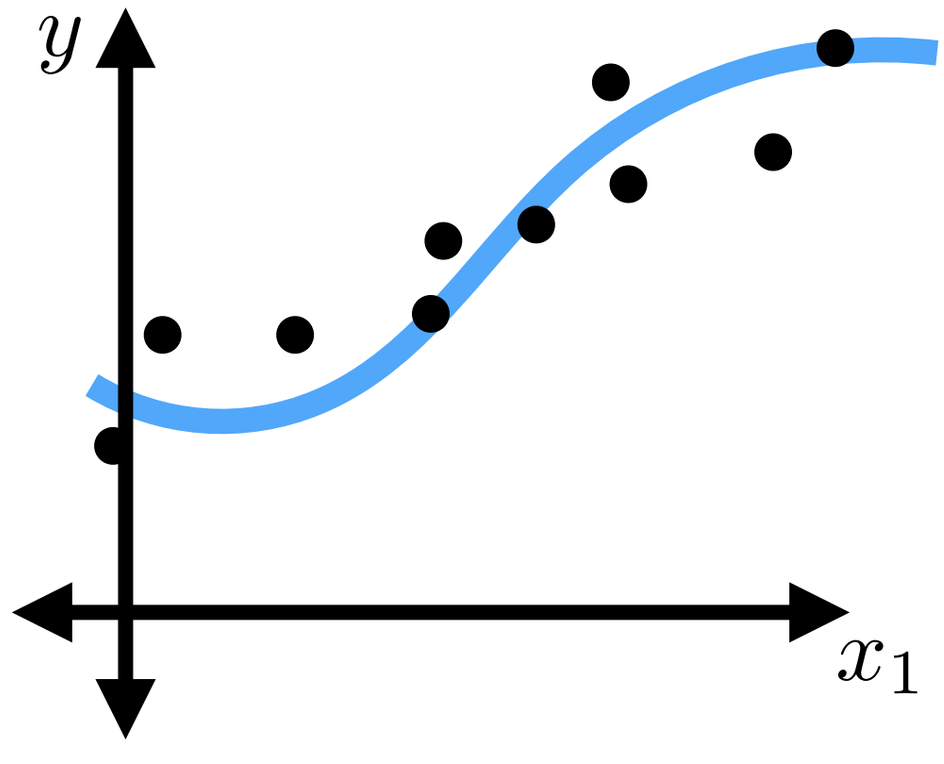
Compare

Regression

- Datum i : feature vector

$$x^{(i)} = (x_1^{(i)}, \ldots, x_d^{(i)})^\top \in \mathbb{R}^d$$

  - Label $y^{(i)} \in \mathbb{R}$
- Hypothesis $h : \mathbb{R}^d \to \mathbb{R}$



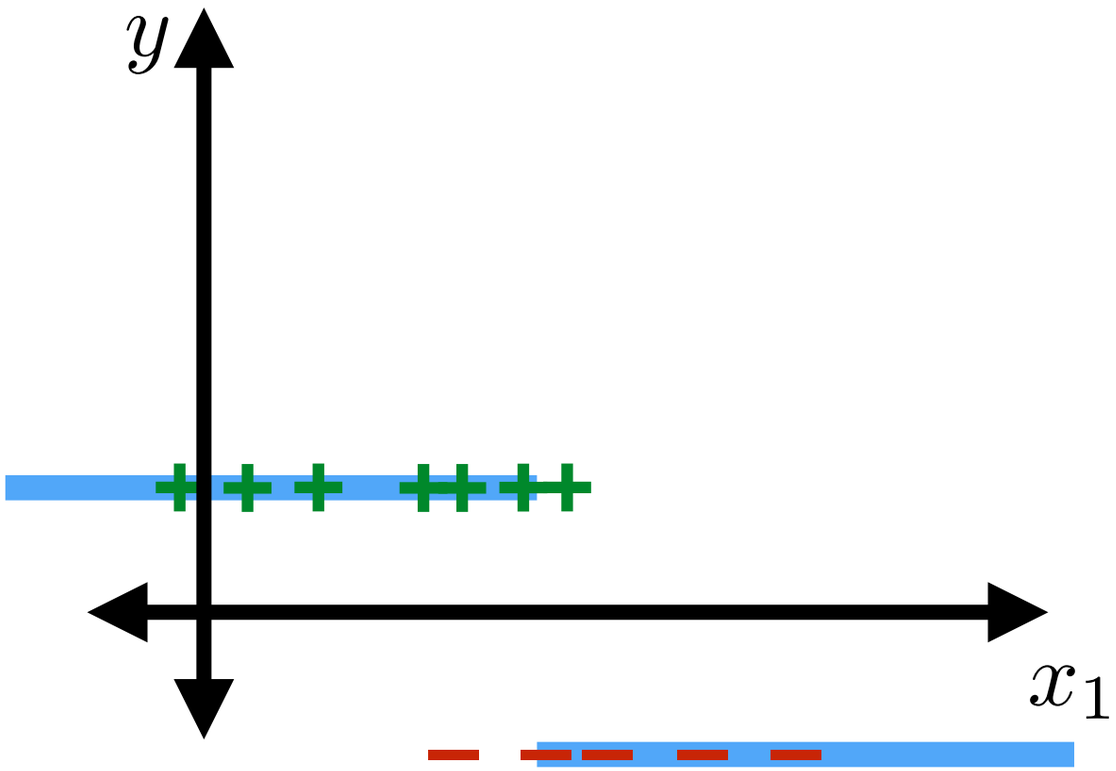
Recall

Classification

- Datum i: feature vector

$$x^{(i)} = (x_1^{(i)}, \ldots, x_d^{(i)})^\top \in \mathbb{R}^d$$

  - Label $y^{(i)} \in \{-1, +1\}$
- Hypothesis $h : \mathbb{R}^d \to \{-1, +1\}$
- Loss: 0-1, asymmetric, NLL
- Example: linear classification



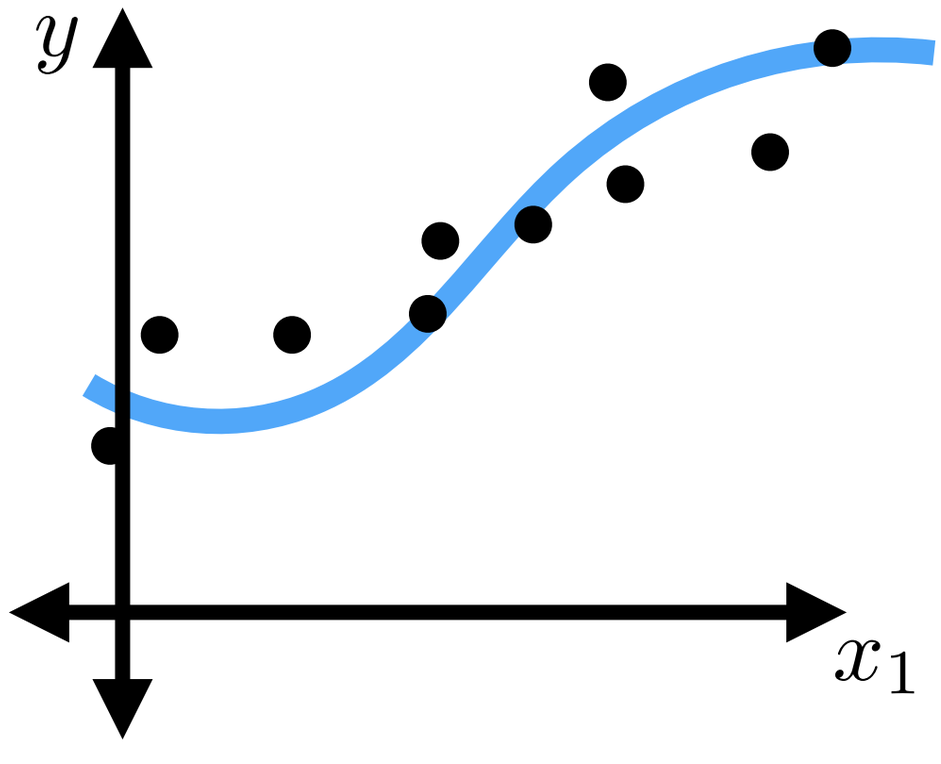
Compare

Regression

- Datum i : feature vector

$$x^{(i)} = (x_1^{(i)}, \ldots, x_d^{(i)})^\top \in \mathbb{R}^d$$

  - Label $y^{(i)} \in \mathbb{R}$
- Hypothesis $h : \mathbb{R}^d \to \mathbb{R}$
- Loss:



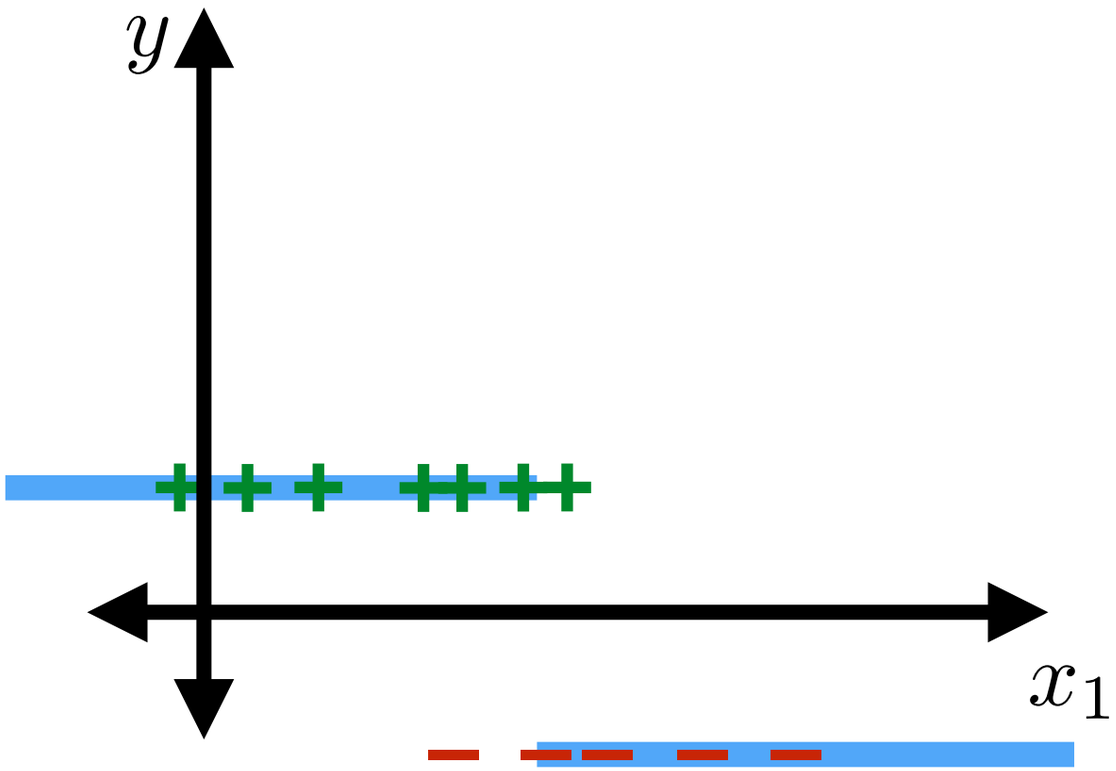
Recall

Classification

- Datum i: feature vector

$$x^{(i)} = (x_1^{(i)}, \ldots, x_d^{(i)})^\top \in \mathbb{R}^d$$

  - Label $y^{(i)} \in \{-1, +1\}$
- Hypothesis $h : \mathbb{R}^d \to \{-1, +1\}$
- Loss: 0-1, asymmetric, NLL
- Example: linear classification



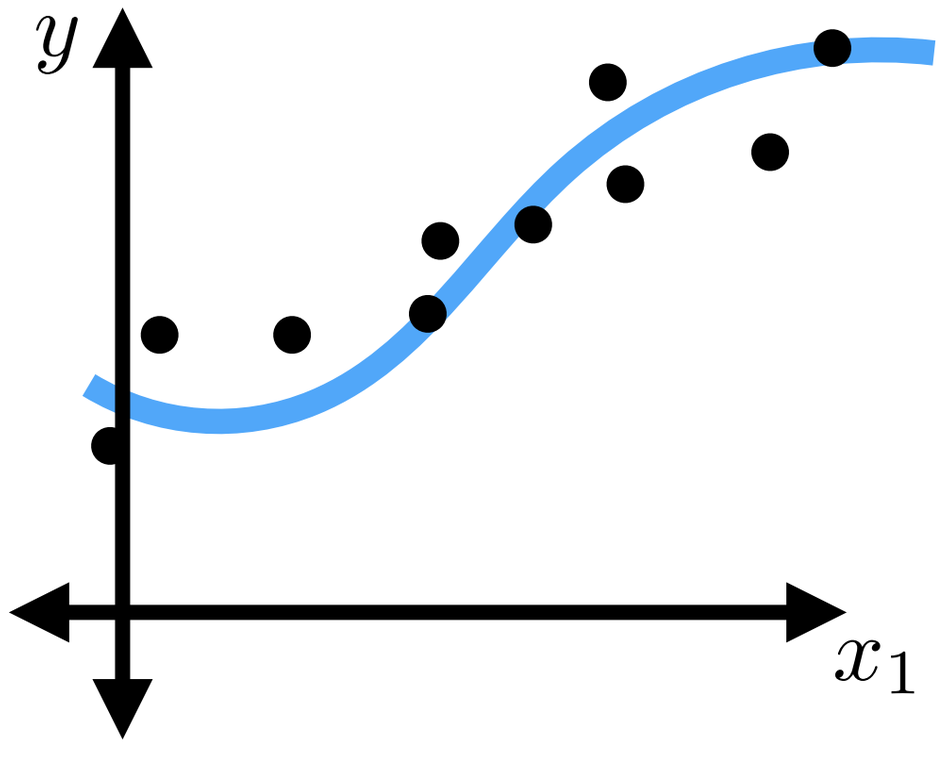
Compare

Regression

- Datum i : feature vector

$$x^{(i)} = (x_1^{(i)}, \ldots, x_d^{(i)})^\top \in \mathbb{R}^d$$

  - Label $y^{(i)} \in \mathbb{R}$
- Hypothesis $h : \mathbb{R}^d \to \mathbb{R}$
- Loss: $L(g, a) = (g - a)^2$



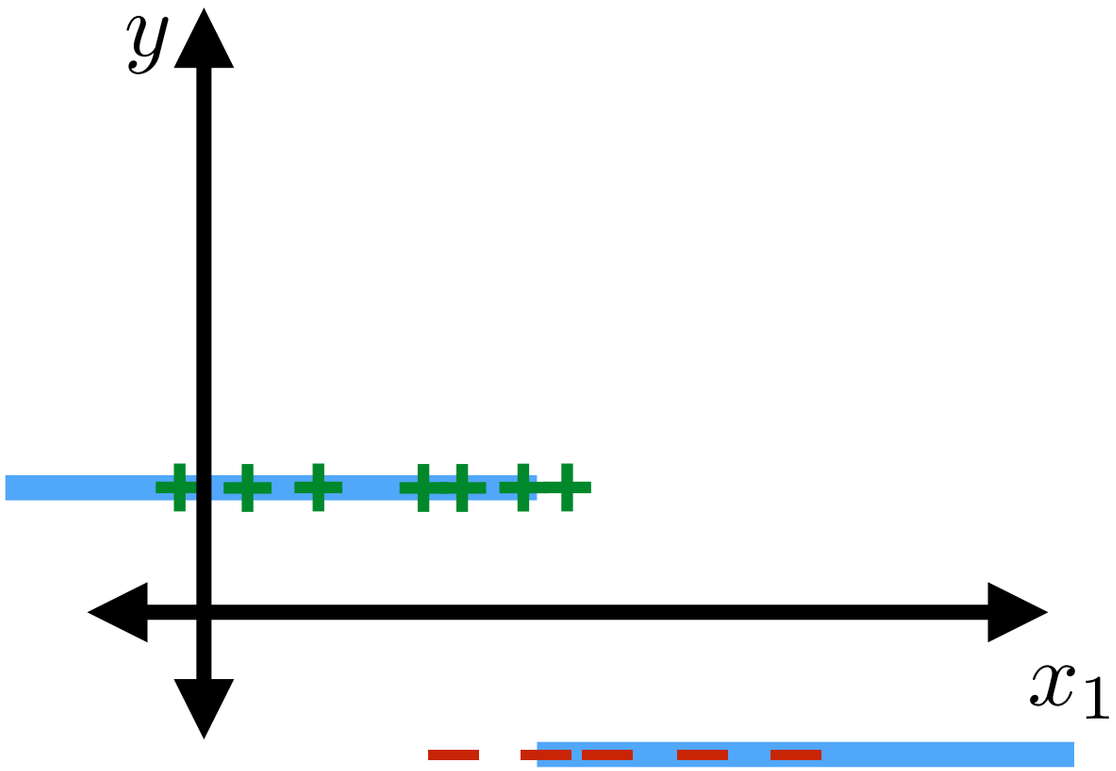
Recall

Classification

- Datum i: feature vector

$$x^{(i)} = (x_1^{(i)}, \ldots, x_d^{(i)})^\top \in \mathbb{R}^d$$

  - Label $y^{(i)} \in \{-1, +1\}$
- Hypothesis $h : \mathbb{R}^d \to \{-1, +1\}$
- Loss: 0-1, asymmetric, NLL
- Example: linear classification



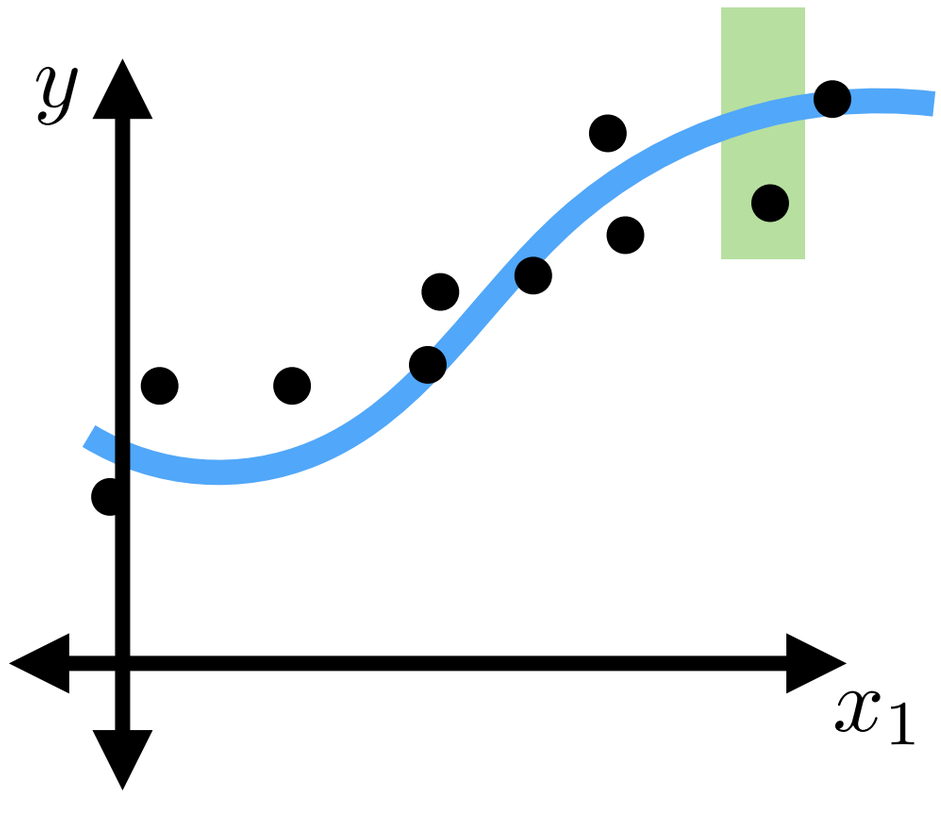
Compare

Regression

- Datum i : feature vector

$$x^{(i)} = (x_1^{(i)}, \ldots, x_d^{(i)})^\top \in \mathbb{R}^d$$

  - Label $y^{(i)} \in \mathbb{R}$
- Hypothesis $h : \mathbb{R}^d \to \mathbb{R}$
- Loss: $L(g, a) = (g - a)^2$



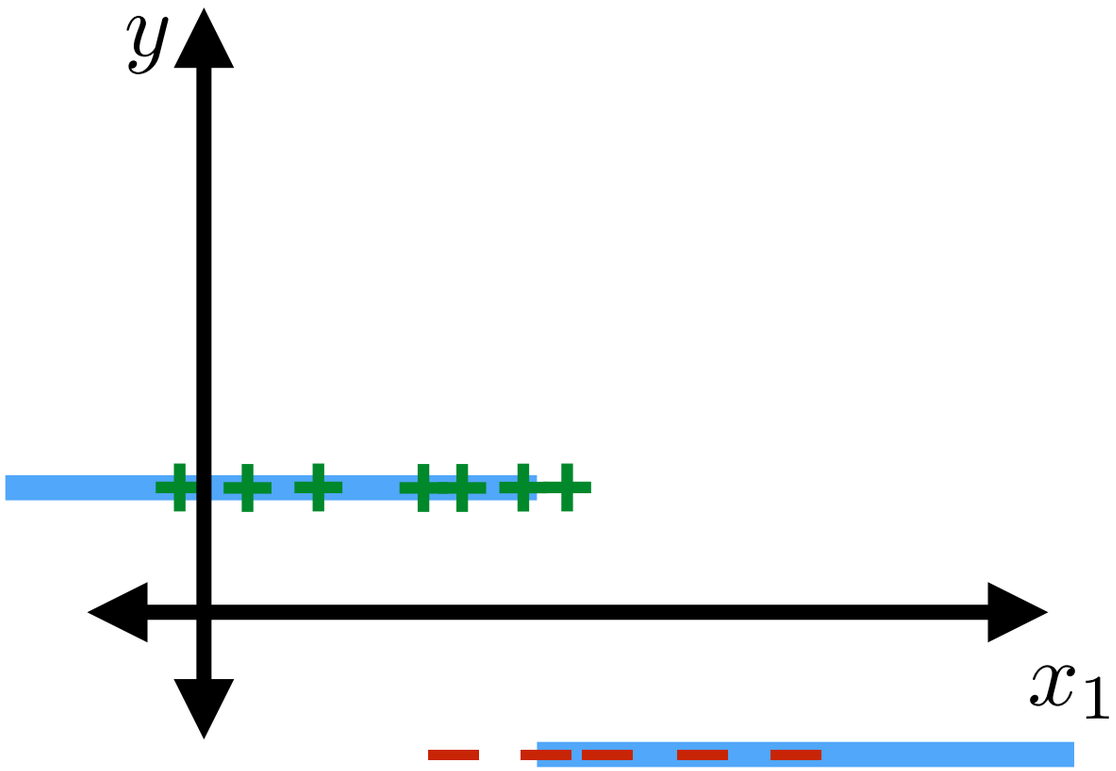
Recall

Classification

- Datum i: feature vector

$$x^{(i)} = (x_1^{(i)}, \ldots, x_d^{(i)})^\top \in \mathbb{R}^d$$

  - Label $y^{(i)} \in \{-1, +1\}$
- Hypothesis $h : \mathbb{R}^d \to \{-1, +1\}$
- Loss: 0-1, asymmetric, NLL
- Example: linear classification



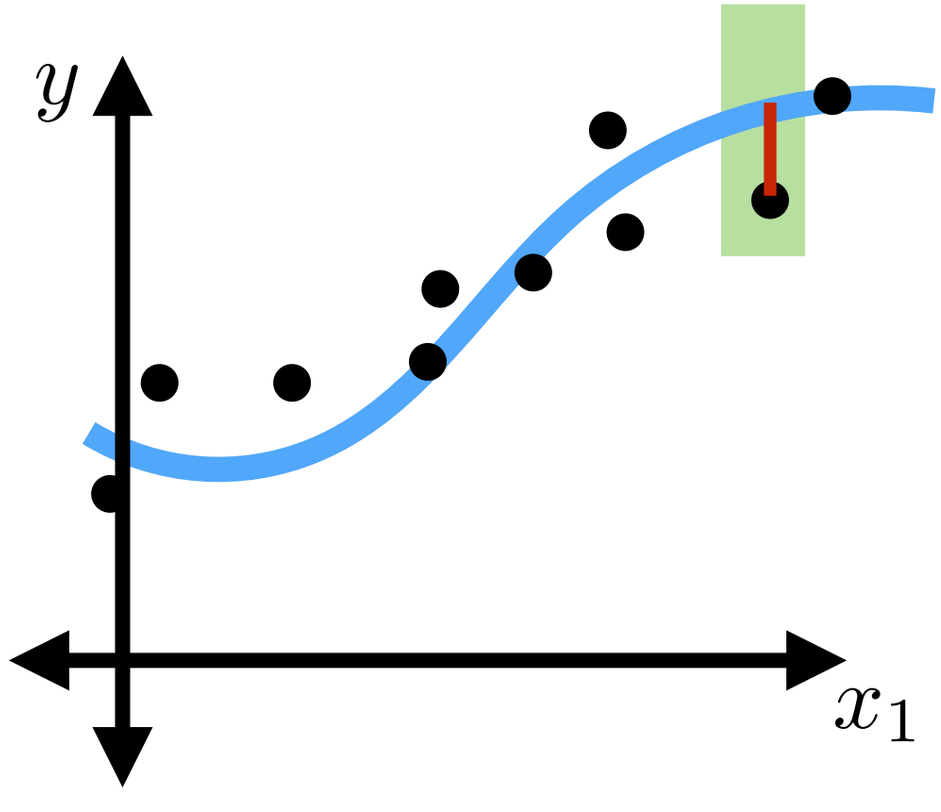
Compare

Regression

- Datum i : feature vector

$$x^{(i)} = (x_1^{(i)}, \ldots, x_d^{(i)})^\top \in \mathbb{R}^d$$

  - Label $y^{(i)} \in \mathbb{R}$
- Hypothesis $h : \mathbb{R}^d \to \mathbb{R}$
- Loss: $L(g, a) = (g - a)^2$



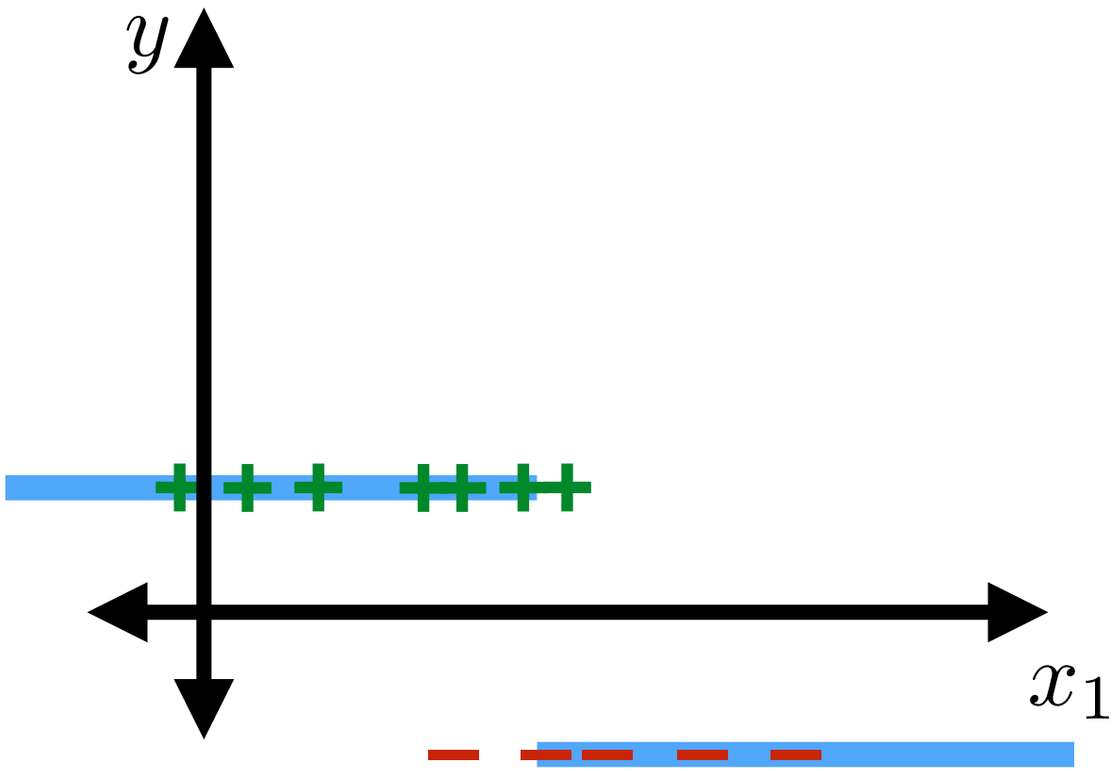
Recall

Classification

- Datum i: feature vector

$$x^{(i)} = (x_1^{(i)}, \ldots, x_d^{(i)})^\top \in \mathbb{R}^d$$

  - Label $y^{(i)} \in \{-1, +1\}$
- Hypothesis $h : \mathbb{R}^d \to \{-1, +1\}$
- Loss: 0-1, asymmetric, NLL
- Example: linear classification



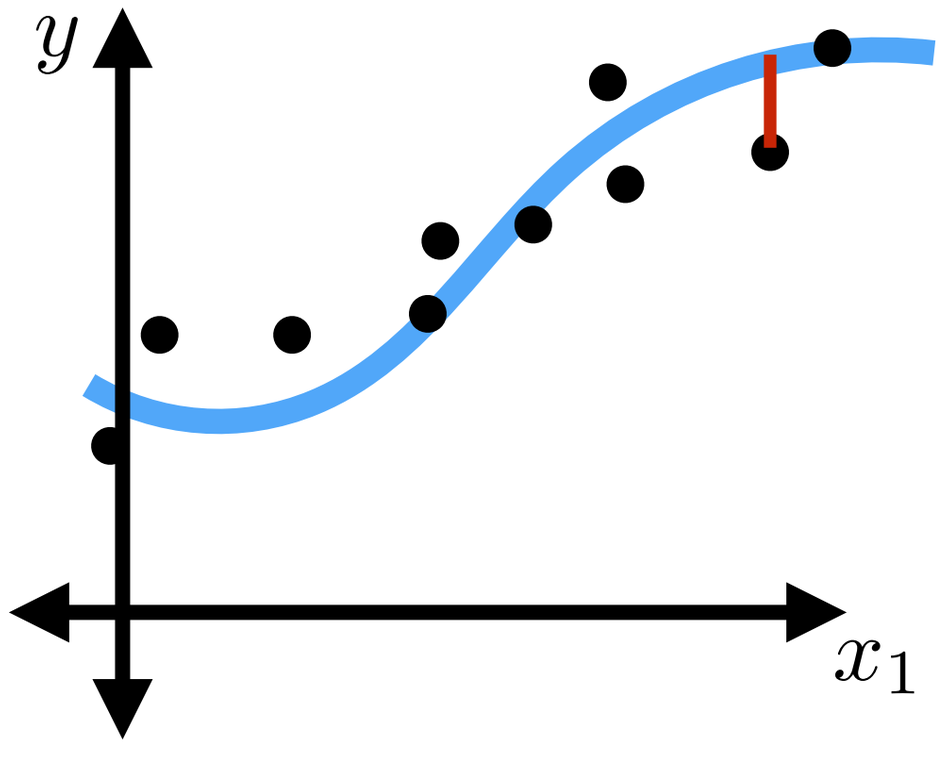
Compare

Regression

- Datum i : feature vector

$$x^{(i)} = (x_1^{(i)}, \ldots, x_d^{(i)})^\top \in \mathbb{R}^d$$

  - Label $y^{(i)} \in \mathbb{R}$
- Hypothesis $h : \mathbb{R}^d \to \mathbb{R}$
- Loss: $L(g, a) = (g - a)^2$



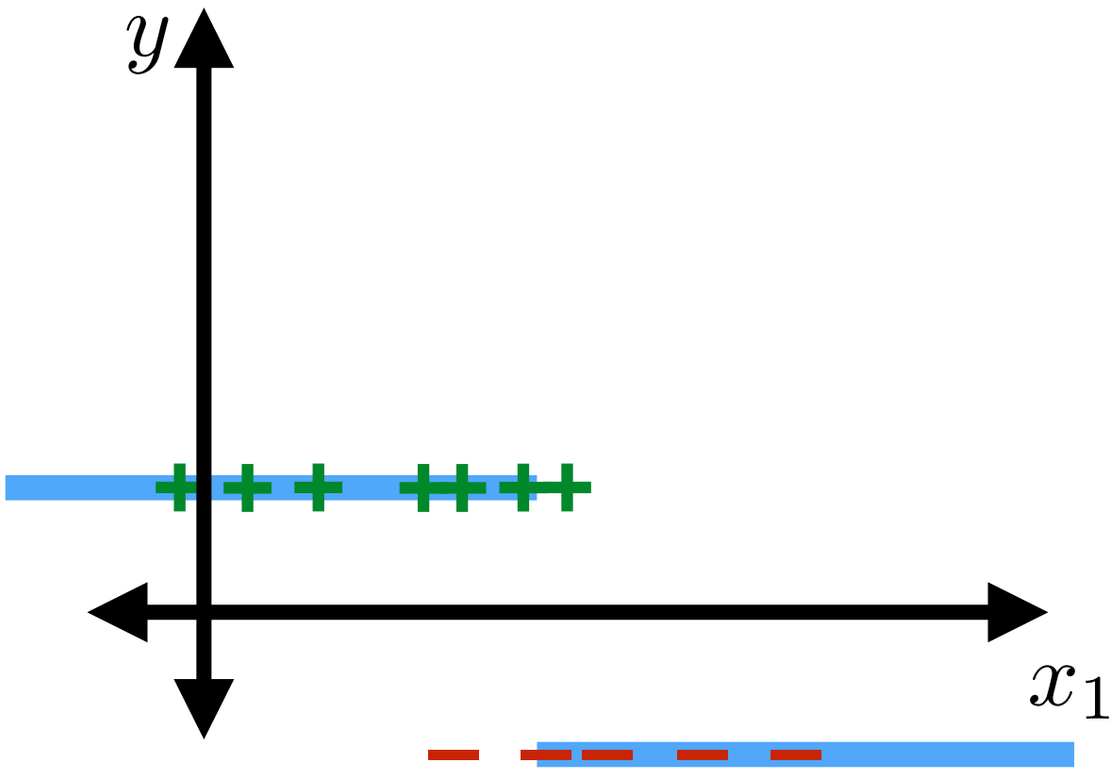
Recall

Classification

- Datum i: feature vector

$$x^{(i)} = (x_1^{(i)}, \ldots, x_d^{(i)})^\top \in \mathbb{R}^d$$

  - Label $y^{(i)} \in \{-1, +1\}$
- Hypothesis $h : \mathbb{R}^d \to \{-1, +1\}$
- Loss: 0-1, asymmetric, NLL
- Example: linear classification



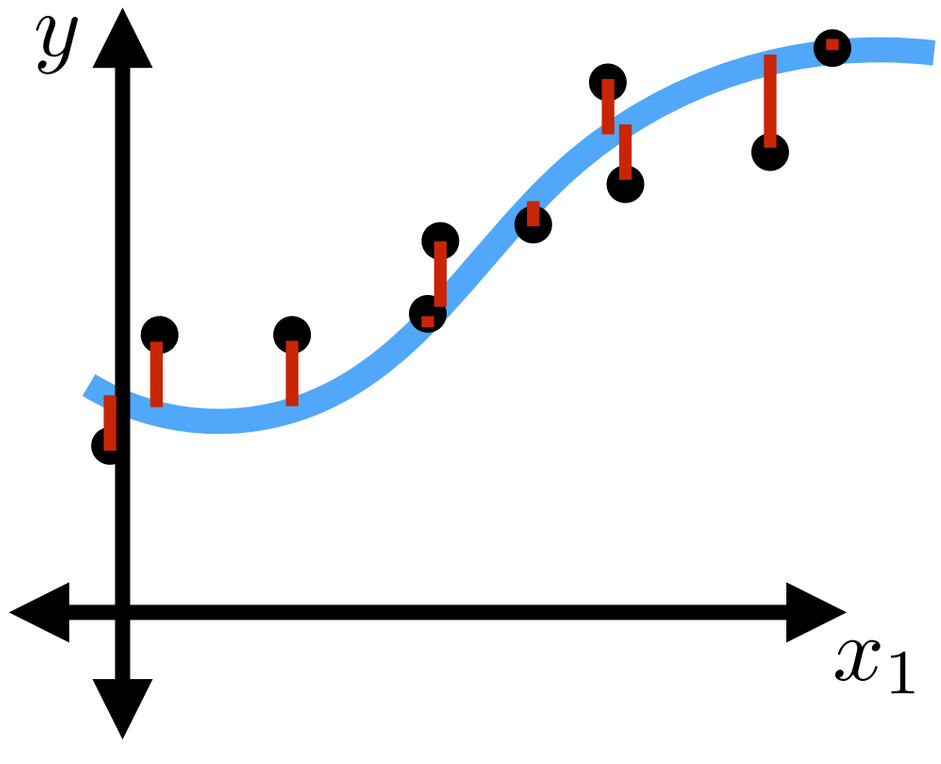
Compare

Regression

- Datum i : feature vector

$$x^{(i)} = (x_1^{(i)}, \ldots, x_d^{(i)})^\top \in \mathbb{R}^d$$

  - Label $y^{(i)} \in \mathbb{R}$
- Hypothesis $h : \mathbb{R}^d \to \mathbb{R}$
- Loss: $L(g, a) = (g - a)^2$



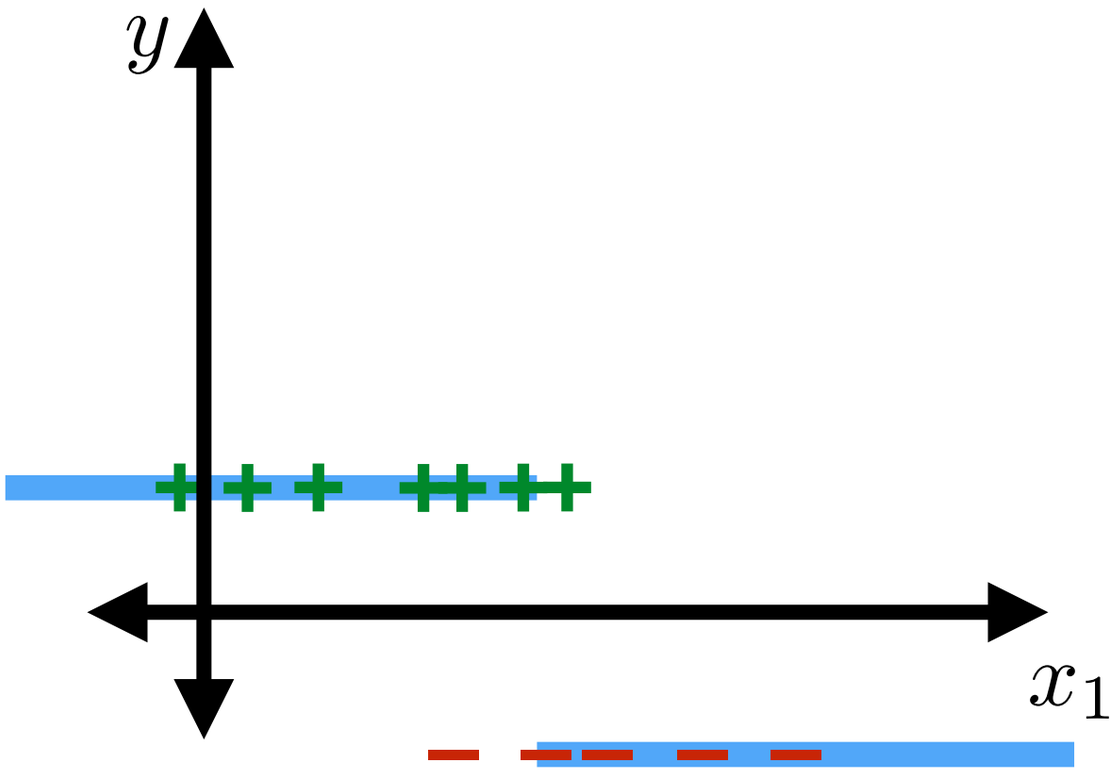
Recall

Classification

- Datum i: feature vector

$$x^{(i)} = (x_1^{(i)}, \ldots, x_d^{(i)})^\top \in \mathbb{R}^d$$

  - Label $y^{(i)} \in \{-1, +1\}$
- Hypothesis $h : \mathbb{R}^d \to \{-1, +1\}$
- Loss: 0-1, asymmetric, NLL
- Example: linear classification



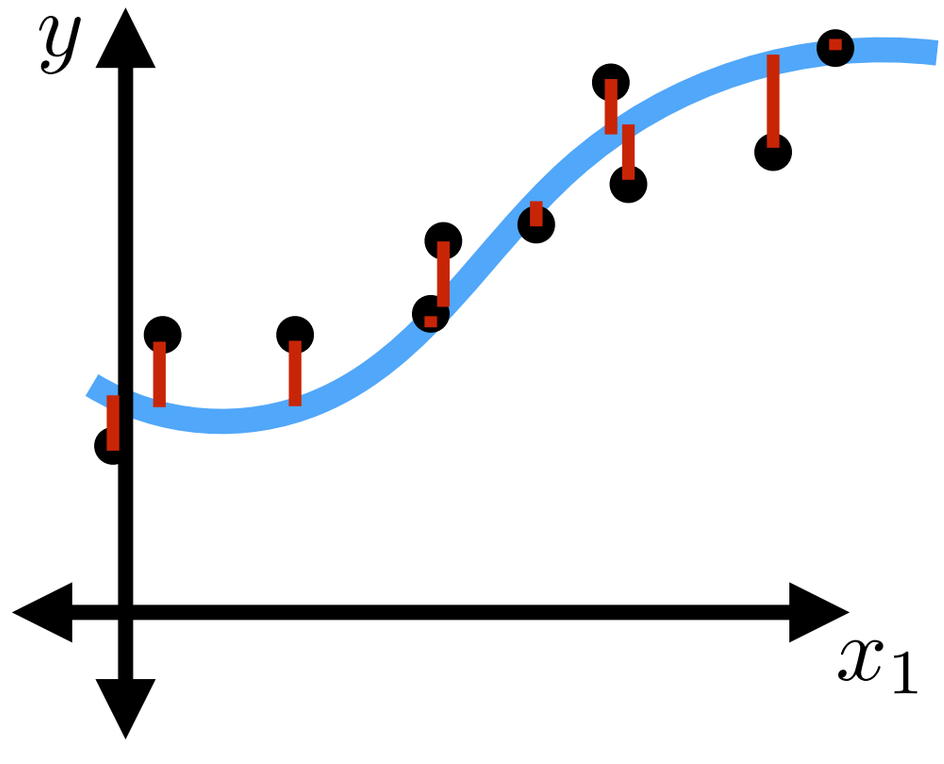
Compare

Regression

- Datum i : feature vector

$$x^{(i)} = (x_1^{(i)}, \ldots, x_d^{(i)})^\top \in \mathbb{R}^d$$

  - Label $y^{(i)} \in \mathbb{R}$
- Hypothesis $h : \mathbb{R}^d \to \mathbb{R}$
- Loss: $L(g, a) = (g - a)^2$
- Example: linear regression



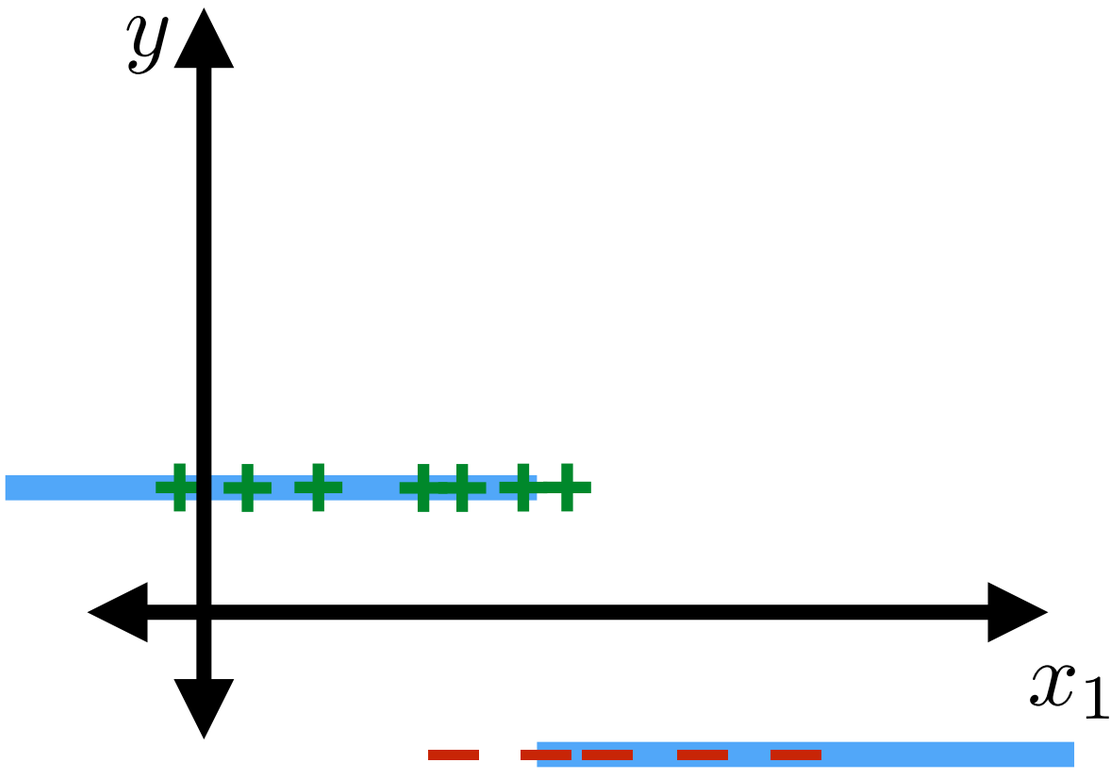
Recall

Classification

- Datum i: feature vector

$$x^{(i)} = (x_1^{(i)}, \ldots, x_d^{(i)})^\top \in \mathbb{R}^d$$

  - Label $y^{(i)} \in \{-1, +1\}$
- Hypothesis $h : \mathbb{R}^d \to \{-1, +1\}$
- Loss: 0-1, asymmetric, NLL
- Example: linear classification



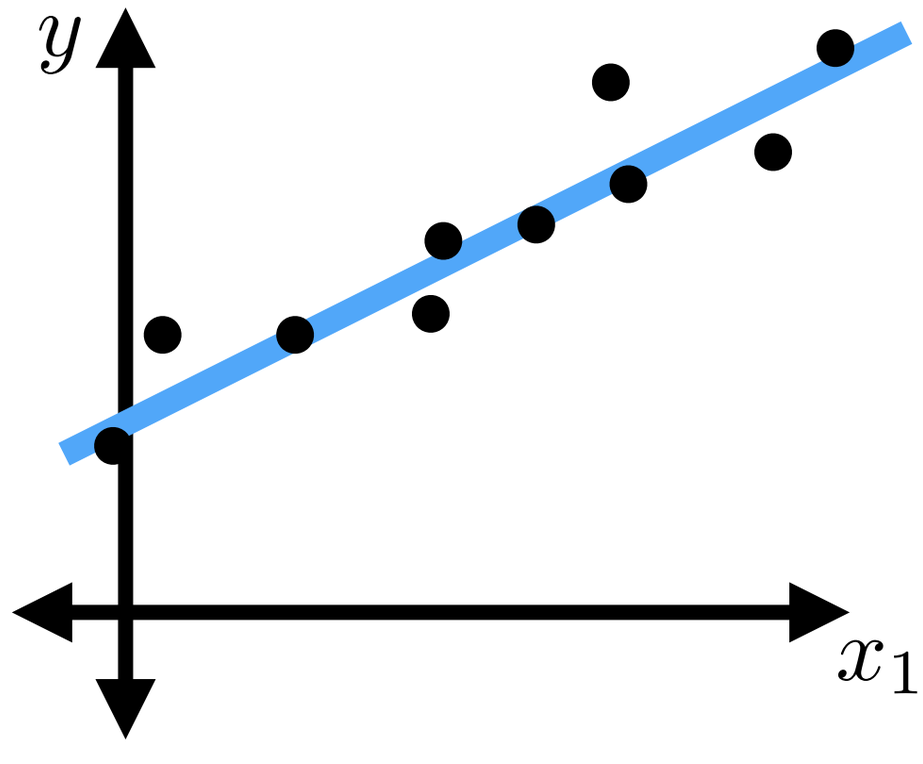
Compare

Regression

- Datum i : feature vector

$$x^{(i)} = (x_1^{(i)}, \ldots, x_d^{(i)})^\top \in \mathbb{R}^d$$

  - Label $y^{(i)} \in \mathbb{R}$
- Hypothesis $h : \mathbb{R}^d \to \mathbb{R}$
- Loss: $L(g, a) = (g - a)^2$
- Example: linear regression



Recall

Classification

- Datum i: feature vector

$$x^{(i)} = (x_1^{(i)}, \ldots, x_d^{(i)})^\top \in \mathbb{R}^d$$

  - Label $y^{(i)} \in \{-1, +1\}$
- Hypothesis $h : \mathbb{R}^d \to \{-1, +1\}$
- Loss: 0-1, asymmetric, NLL
- Example: linear classification

Compare

Regression

- Datum i : feature vector

$$x^{(i)} = (x_1^{(i)}, \ldots, x_d^{(i)})^\top \in \mathbb{R}^d$$

  - Label $y^{(i)} \in \mathbb{R}$
- Hypothesis $h : \mathbb{R}^d \to \mathbb{R}$
- Loss: $L(g, a) = (g - a)^2$
- Example: linear regression

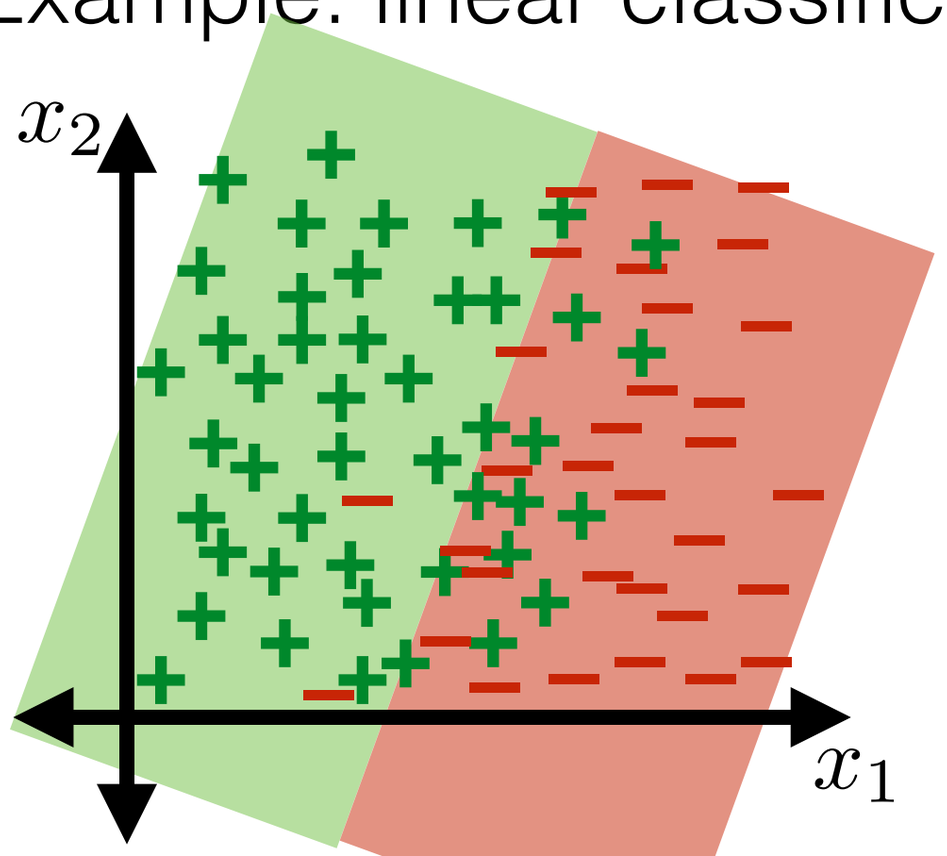
Recall

Classification

- Datum i: feature vector

$$x^{(i)} = (x_1^{(i)}, \ldots, x_d^{(i)})^\top \in \mathbb{R}^d$$

  - Label $y^{(i)} \in \{-1, +1\}$
- Hypothesis $h : \mathbb{R}^d \to \{-1, +1\}$
- Loss: 0-1, asymmetric, NLL
- Example: linear classification



Compare

Regression

- Datum i : feature vector

$$x^{(i)} = (x_1^{(i)}, \ldots, x_d^{(i)})^\top \in \mathbb{R}^d$$

  - Label $y^{(i)} \in \mathbb{R}$
- Hypothesis $h : \mathbb{R}^d \to \mathbb{R}$
- Loss: $L(g, a) = (g - a)^2$
- Example: linear regression

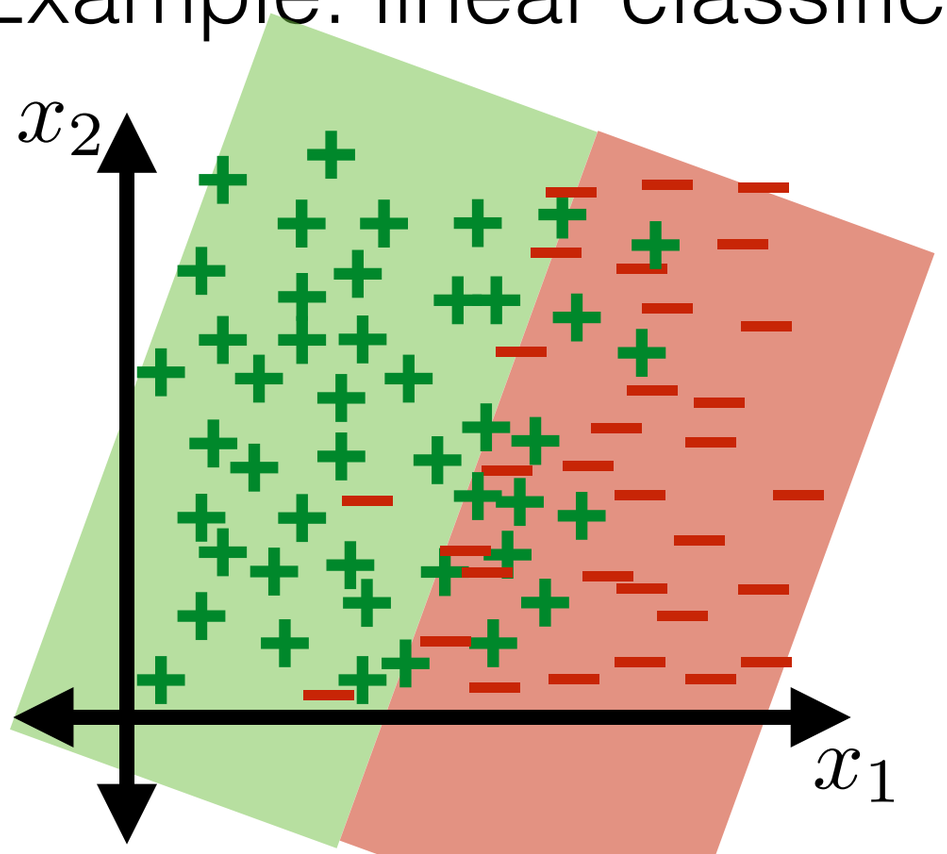
Recall

Classification

- Datum i: feature vector

$$x^{(i)} = (x_1^{(i)}, \ldots, x_d^{(i)})^\top \in \mathbb{R}^d$$

  - Label $y^{(i)} \in \{-1, +1\}$
- Hypothesis $h : \mathbb{R}^d \to \{-1, +1\}$
- Loss: 0-1, asymmetric, NLL
- Example: linear classification



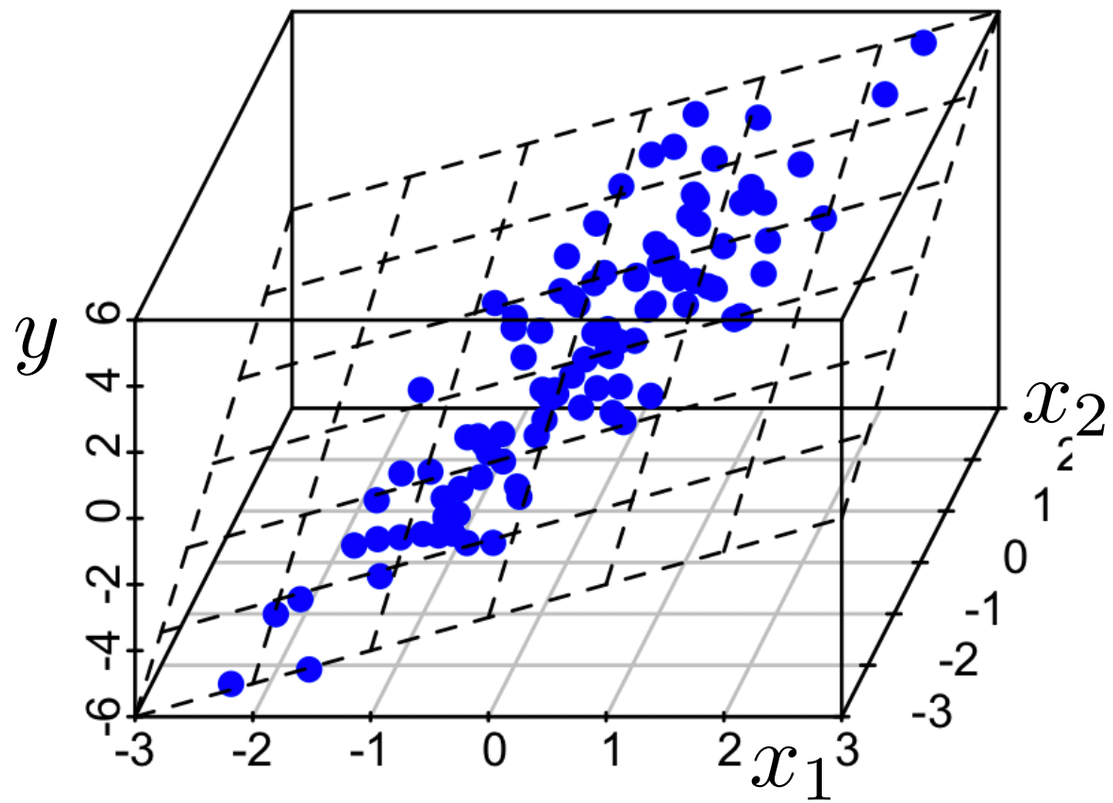
Compare

Regression

- Datum i : feature vector

$$x^{(i)} = (x_1^{(i)}, \ldots, x_d^{(i)})^\top \in \mathbb{R}^d$$

  - Label $y^{(i)} \in \mathbb{R}$
- Hypothesis $h : \mathbb{R}^d \to \mathbb{R}$
- Loss: $L(g, a) = (g - a)^2$
- Example: linear regression



Linear regression

- Hypotheses for linear regression: $h(x; \theta, \theta_0) = \theta^\top x + \theta_0$

Linear regression

- Hypotheses for linear regression: $h(x; \theta, \theta_0) = \theta^\top x + \theta_0$
- Training error (using squared error loss)

Linear regression

- Hypotheses for linear regression: $h(x; \theta, \theta_0) = \theta^\top x + \theta_0$
- Training error (using squared error loss)

$$J(\theta, \theta_0) = \frac{1}{n} \sum_{i=1}^{n} L(h(x^{(i)}; \theta, \theta_0), y^{(i)})$$

Linear regression

- Hypotheses for linear regression: $h(x; \theta, \theta_0) = \theta^\top x + \theta_0$
- Training error (using squared error loss)

$$J(\theta, \theta_0) = \frac{1}{n} \sum_{i=1}^{n} L(h(x^{(i)}; \theta, \theta_0), y^{(i)}) = \frac{1}{n} \sum_{i=1}^{n} (\theta^\top x^{(i)} + \theta_0 - y^{(i)})^2$$

Linear regression

- Hypotheses for linear regression: $h(x; \theta, \theta_0) = \theta^\top x + \theta_0$
- Training error (using squared error loss)

$$J(\theta, \theta_0) = \frac{1}{n} \sum_{i=1}^{n} L(h(x^{(i)}; \theta, \theta_0), y^{(i)}) = \frac{1}{n} \sum_{i=1}^{n} (\theta^\top x^{(i)} + \theta_0 - y^{(i)})^2$$

- Training error if we augment features with feature “1”

Linear regression

- Hypotheses for linear regression: $h(x; \theta, \theta_0) = \theta^\top x + \theta_0$
- Training error (using squared error loss)

$$J(\theta, \theta_0) = \frac{1}{n} \sum_{i=1}^{n} L(h(x^{(i)}; \theta, \theta_0), y^{(i)}) = \frac{1}{n} \sum_{i=1}^{n} (\theta^\top x^{(i)} + \theta_0 - y^{(i)})^2$$

- Training error if we augment features with feature "1"

$$J(\theta) = \frac{1}{n} \sum_{i=1}^{n} (\theta^\top x^{(i)} - y^{(i)})^2$$

Linear regression

- Hypotheses for linear regression: $h(x;\theta,\theta_0) = \theta^\top x + \theta_0$
- Training error (using squared error loss)

$$J(\theta,\theta_0) = \frac{1}{n}\sum_{i=1}^{n} L(h(x^{(i)};\theta,\theta_0), y^{(i)}) = \frac{1}{n}\sum_{i=1}^{n}(\theta^\top x^{(i)} + \theta_0 - y^{(i)})^2$$

- Training error if we augment features with feature "1"

$$J(\theta) = \frac{1}{n}\sum_{i=1}^{n}(\theta^\top x^{(i)} - y^{(i)})^2$$

1xd

Linear regression

- Hypotheses for linear regression: $h(x;\theta,\theta_0) = \theta^\top x + \theta_0$
- Training error (using squared error loss)

$$J(\theta,\theta_0) = \frac{1}{n}\sum_{i=1}^{n} L(h(x^{(i)};\theta,\theta_0), y^{(i)}) = \frac{1}{n}\sum_{i=1}^{n}(\theta^\top x^{(i)} + \theta_0 - y^{(i)})^2$$

- Training error if we augment features with feature "1"

$$J(\theta) = \frac{1}{n}\sum_{i=1}^{n}(\theta^\top x^{(i)} - y^{(i)})^2$$

1xd,dx1

Linear regression

- Hypotheses for linear regression: $h(x;\theta,\theta_0) = \theta^\top x + \theta_0$
- Training error (using squared error loss)

$$J(\theta,\theta_0) = \frac{1}{n}\sum_{i=1}^{n} L(h(x^{(i)};\theta,\theta_0), y^{(i)}) = \frac{1}{n}\sum_{i=1}^{n}(\theta^\top x^{(i)} + \theta_0 - y^{(i)})^2$$

- Training error if we augment features with feature "1"

$$J(\theta) = \frac{1}{n}\sum_{i=1}^{n}(\theta^\top x^{(i)} - y^{(i)})^2$$

1xd,dx1 1x1

Linear regression

- Hypotheses for linear regression: $h(x;\theta,\theta_0) = \theta^\top x + \theta_0$
- Training error (using squared error loss)

$$J(\theta,\theta_0) = \frac{1}{n}\sum_{i=1}^{n} L(h(x^{(i)};\theta,\theta_0), y^{(i)}) = \frac{1}{n}\sum_{i=1}^{n}(\theta^\top x^{(i)} + \theta_0 - y^{(i)})^2$$

- Training error if we augment features with feature "1"

$$J(\theta) = \frac{1}{n}\sum_{i=1}^{n}(\underbrace{\theta^\top x^{(i)}}_{\text{1xd,dx1}} - \underbrace{y^{(i)}}_{\text{1x1}})^2 = \frac{1}{n}\sum_{i=1}^{n}((x^{(i)})^\top\theta - y^{(i)})^2$$

Linear regression

- Hypotheses for linear regression: $h(x;\theta,\theta_0) = \theta^\top x + \theta_0$
- Training error (using squared error loss)

$$J(\theta,\theta_0) = \frac{1}{n}\sum_{i=1}^{n} L(h(x^{(i)};\theta,\theta_0), y^{(i)}) = \frac{1}{n}\sum_{i=1}^{n}(\theta^\top x^{(i)} + \theta_0 - y^{(i)})^2$$

- Training error if we augment features with feature "1"

$$J(\theta) = \frac{1}{n}\sum_{i=1}^{n}(\theta^\top x^{(i)} - y^{(i)})^2 = \frac{1}{n}\sum_{i=1}^{n}((\underbrace{x^{(i)}}_{\text{1xd}})^\top \theta - y^{(i)})^2$$

1xd,dx1 1x1

Linear regression

- Hypotheses for linear regression: $h(x; \theta, \theta_0) = \theta^\top x + \theta_0$
- Training error (using squared error loss)

$$J(\theta, \theta_0) = \frac{1}{n} \sum_{i=1}^{n} L(h(x^{(i)}; \theta, \theta_0), y^{(i)}) = \frac{1}{n} \sum_{i=1}^{n} (\theta^\top x^{(i)} + \theta_0 - y^{(i)})^2$$

- Training error if we augment features with feature "1"

$$J(\theta) = \frac{1}{n} \sum_{i=1}^{n} (\underbrace{\theta^\top x^{(i)}}_{\text{1xd,dx1}} - \underbrace{y^{(i)}}_{\text{1x1}})^2 = \frac{1}{n} \sum_{i=1}^{n} (\underbrace{(x^{(i)})^\top}_{\text{1xd,dx1}} \theta - y^{(i)})^2$$

Linear regression

- Hypotheses for linear regression: $h(x;\theta,\theta_0) = \theta^\top x + \theta_0$
- Training error (using squared error loss)

$$J(\theta,\theta_0) = \frac{1}{n}\sum_{i=1}^{n} L(h(x^{(i)};\theta,\theta_0), y^{(i)}) = \frac{1}{n}\sum_{i=1}^{n}(\theta^\top x^{(i)} + \theta_0 - y^{(i)})^2$$

- Training error if we augment features with feature "1"

$$J(\theta) = \frac{1}{n}\sum_{i=1}^{n}(\theta^\top x^{(i)} - y^{(i)})^2 = \frac{1}{n}\sum_{i=1}^{n}((\underbrace{x^{(i)}}_{\text{1xd,dx1}})^\top \theta - \underbrace{y^{(i)}}_{\text{1x1}})^2$$

Linear regression

- Hypotheses for linear regression: $h(x;\theta,\theta_0) = \theta^\top x + \theta_0$
- Training error (using squared error loss)

$$J(\theta,\theta_0) = \frac{1}{n}\sum_{i=1}^{n} L(h(x^{(i)};\theta,\theta_0), y^{(i)}) = \frac{1}{n}\sum_{i=1}^{n}(\theta^\top x^{(i)} + \theta_0 - y^{(i)})^2$$

- Training error if we augment features with feature "1"

$$J(\theta) = \frac{1}{n}\sum_{i=1}^{n}(\theta^\top x^{(i)} - y^{(i)})^2 = \frac{1}{n}\sum_{i=1}^{n}(\underbrace{(x^{(i)})^\top}_{\text{1xd,dx1}}\theta - \underbrace{y^{(i)}}_{\text{1x1}})^2$$

Linear regression

- Hypotheses for linear regression: $h(x;\theta,\theta_0) = \theta^\top x + \theta_0$
- Training error (using squared error loss)

$$J(\theta,\theta_0) = \frac{1}{n}\sum_{i=1}^{n} L(h(x^{(i)};\theta,\theta_0), y^{(i)}) = \frac{1}{n}\sum_{i=1}^{n}(\theta^\top x^{(i)} + \theta_0 - y^{(i)})^2$$

- Training error if we augment features with feature "1"

$$J(\theta) = \frac{1}{n}\sum_{i=1}^{n}(\theta^\top x^{(i)} - y^{(i)})^2 = \frac{1}{n}\sum_{i=1}^{n}((\underbrace{x^{(i)}}_{\text{1xd,dx1}})^\top \theta - \underbrace{y^{(i)}}_{\text{1x1}})^2$$

Linear regression

- Hypotheses for linear regression: $h(x;\theta,\theta_0) = \theta^\top x + \theta_0$
- Training error (using squared error loss)

$$J(\theta,\theta_0) = \frac{1}{n}\sum_{i=1}^{n} L(h(x^{(i)};\theta,\theta_0), y^{(i)}) = \frac{1}{n}\sum_{i=1}^{n}(\theta^\top x^{(i)} + \theta_0 - y^{(i)})^2$$

- Training error if we augment features with feature "1"

$$J(\theta) = \frac{1}{n}\sum_{i=1}^{n}(\theta^\top x^{(i)} - y^{(i)})^2 = \frac{1}{n}\sum_{i=1}^{n}((\underbrace{x^{(i)}}_{\text{1xd,dx1}})^\top \theta - \underbrace{y^{(i)}}_{\text{1x1}})^2$$

Define $\tilde{X} = \begin{bmatrix} x_1^{(1)} & \cdots & x_d^{(1)} \\ \vdots & \ddots & \vdots \\ x_1^{(n)} & \cdots & x_d^{(n)} \end{bmatrix}$

Linear regression

- Hypotheses for linear regression: $h(x;\theta,\theta_0) = \theta^\top x + \theta_0$
- Training error (using squared error loss)

$$J(\theta,\theta_0) = \frac{1}{n}\sum_{i=1}^{n} L(h(x^{(i)};\theta,\theta_0), y^{(i)}) = \frac{1}{n}\sum_{i=1}^{n}(\theta^\top x^{(i)} + \theta_0 - y^{(i)})^2$$

- Training error if we augment features with feature "1"

$$J(\theta) = \frac{1}{n}\sum_{i=1}^{n}(\theta^\top x^{(i)} - y^{(i)})^2 = \frac{1}{n}\sum_{i=1}^{n}((\underbrace{x^{(i)}}_{\text{1xd,dx1}})^\top \theta - \underbrace{y^{(i)}}_{\text{1x1}})^2$$

Define $\underset{\text{nxd}}{\tilde{X}} = \begin{bmatrix} x_1^{(1)} & \cdots & x_d^{(1)} \\ \vdots & \ddots & \vdots \\ x_1^{(n)} & \cdots & x_d^{(n)} \end{bmatrix}$

Linear regression

- Hypotheses for linear regression: $h(x;\theta,\theta_0) = \theta^\top x + \theta_0$
- Training error (using squared error loss)

$$J(\theta,\theta_0) = \frac{1}{n}\sum_{i=1}^{n} L(h(x^{(i)};\theta,\theta_0), y^{(i)}) = \frac{1}{n}\sum_{i=1}^{n}(\theta^\top x^{(i)} + \theta_0 - y^{(i)})^2$$

- Training error if we augment features with feature "1"

$$J(\theta) = \frac{1}{n}\sum_{i=1}^{n}(\underbrace{\theta^\top x^{(i)}}_{\text{1xd,dx1}} - \underbrace{y^{(i)}}_{\text{1x1}})^2 = \frac{1}{n}\sum_{i=1}^{n}((\underbrace{x^{(i)}}_{\text{1xd,dx1}})^\top\theta - \underbrace{y^{(i)}}_{\text{1x1}})^2$$

Define $\underbrace{\tilde{X}}_{\text{nxd}} = \begin{bmatrix} x_1^{(1)} & \cdots & x_d^{(1)} \\ \vdots & \ddots & \vdots \\ x_1^{(n)} & \cdots & x_d^{(n)} \end{bmatrix}$

Linear regression

- Hypotheses for linear regression: $h(x;\theta,\theta_0) = \theta^\top x + \theta_0$
- Training error (using squared error loss)

$$J(\theta,\theta_0) = \frac{1}{n}\sum_{i=1}^{n} L(h(x^{(i)};\theta,\theta_0), y^{(i)}) = \frac{1}{n}\sum_{i=1}^{n}(\theta^\top x^{(i)} + \theta_0 - y^{(i)})^2$$

- Training error if we augment features with feature "1"

$$J(\theta) = \frac{1}{n}\sum_{i=1}^{n}(\underbrace{\theta^\top x^{(i)}}_{\text{1xd,dx1}} - \underbrace{y^{(i)}}_{\text{1x1}})^2 = \frac{1}{n}\sum_{i=1}^{n}((\underbrace{x^{(i)}}_{\text{1xd,dx1}})^\top\theta - \underbrace{y^{(i)}}_{\text{1x1}})^2$$

Define $\underset{\text{nxd}}{\tilde{X}} = \begin{bmatrix} x_1^{(1)} & \cdots & x_d^{(1)} \\ \vdots & \ddots & \vdots \\ x_1^{(n)} & \cdots & x_d^{(n)} \end{bmatrix}$

Linear regression

- Hypotheses for linear regression: $h(x;\theta,\theta_0) = \theta^\top x + \theta_0$
- Training error (using squared error loss)

$$J(\theta,\theta_0) = \frac{1}{n}\sum_{i=1}^{n} L(h(x^{(i)};\theta,\theta_0), y^{(i)}) = \frac{1}{n}\sum_{i=1}^{n}(\theta^\top x^{(i)} + \theta_0 - y^{(i)})^2$$

- Training error if we augment features with feature "1"

$$J(\theta) = \frac{1}{n}\sum_{i=1}^{n}(\theta^\top x^{(i)} - y^{(i)})^2 = \frac{1}{n}\sum_{i=1}^{n}((\underbrace{x^{(i)}}_{\text{1xd,dx1}})^\top \theta - \underbrace{y^{(i)}}_{\text{1x1}})^2$$

Define $\underset{\text{nxd}}{\tilde{X}} = \begin{bmatrix} x_1^{(1)} & \cdots & x_d^{(1)} \\ \vdots & \ddots & \vdots \\ x_1^{(n)} & \cdots & x_d^{(n)} \end{bmatrix}$

Linear regression

- Hypotheses for linear regression: $h(x;\theta,\theta_0) = \theta^\top x + \theta_0$
- Training error (using squared error loss)

$$J(\theta,\theta_0) = \frac{1}{n}\sum_{i=1}^{n} L(h(x^{(i)};\theta,\theta_0), y^{(i)}) = \frac{1}{n}\sum_{i=1}^{n}(\theta^\top x^{(i)} + \theta_0 - y^{(i)})^2$$

- Training error if we augment features with feature "1"

$$J(\theta) = \frac{1}{n}\sum_{i=1}^{n}(\theta^\top x^{(i)} - y^{(i)})^2 = \frac{1}{n}\sum_{i=1}^{n}((\underbrace{x^{(i)}}_{\text{1xd,dx1}})^\top \theta - \underbrace{y^{(i)}}_{\text{1x1}})^2$$

Define $\underset{\text{nxd}}{\tilde{X}} = \begin{bmatrix} x_1^{(1)} & \cdots & x_d^{(1)} \\ \vdots & \ddots & \vdots \\ x_1^{(n)} & \cdots & x_d^{(n)} \end{bmatrix}$ $\tilde{Y} = \begin{bmatrix} y^{(1)} \\ \vdots \\ y^{(n)} \end{bmatrix}$

Linear regression

- Hypotheses for linear regression: $h(x;\theta,\theta_0) = \theta^\top x + \theta_0$
- Training error (using squared error loss)

$$J(\theta,\theta_0) = \frac{1}{n}\sum_{i=1}^{n} L(h(x^{(i)};\theta,\theta_0), y^{(i)}) = \frac{1}{n}\sum_{i=1}^{n}(\theta^\top x^{(i)} + \theta_0 - y^{(i)})^2$$

- Training error if we augment features with feature "1"

$$J(\theta) = \frac{1}{n}\sum_{i=1}^{n}(\underbrace{\theta^\top x^{(i)}}_{1\text{x}d,d\text{x}1} - \underbrace{y^{(i)}}_{1\text{x}1})^2 = \frac{1}{n}\sum_{i=1}^{n}((\underbrace{x^{(i)}}_{1\text{x}d,d\text{x}1})^\top\theta - \underbrace{y^{(i)}}_{1\text{x}1})^2$$

Define $\underset{n\text{x}d}{\tilde{X}} = \begin{bmatrix} x_1^{(1)} & \cdots & x_d^{(1)} \\ \vdots & \ddots & \vdots \\ x_1^{(n)} & \cdots & x_d^{(n)} \end{bmatrix}$ $\underset{n\text{x}1}{\tilde{Y}} = \begin{bmatrix} y^{(1)} \\ \vdots \\ y^{(n)} \end{bmatrix}$

Linear regression

- Hypotheses for linear regression: $h(x;\theta,\theta_0) = \theta^\top x + \theta_0$
- Training error (using squared error loss)

$$J(\theta,\theta_0) = \frac{1}{n}\sum_{i=1}^{n} L(h(x^{(i)};\theta,\theta_0), y^{(i)}) = \frac{1}{n}\sum_{i=1}^{n}(\theta^\top x^{(i)} + \theta_0 - y^{(i)})^2$$

- Training error if we augment features with feature "1"

$$J(\theta) = \frac{1}{n}\sum_{i=1}^{n}(\theta^\top x^{(i)} - y^{(i)})^2 = \frac{1}{n}\sum_{i=1}^{n}(\underbrace{(x^{(i)})^\top}_{\text{1xd,dx1}}\theta - \underbrace{y^{(i)}}_{\text{1x1}})^2$$

Define $\underset{\text{nxd}}{\tilde{X}} = \begin{bmatrix} x_1^{(1)} & \cdots & x_d^{(1)} \\ \vdots & \ddots & \vdots \\ x_1^{(n)} & \cdots & x_d^{(n)} \end{bmatrix}$ $\underset{\text{nx1}}{\tilde{Y}} = \begin{bmatrix} y^{(1)} \\ \vdots \\ y^{(n)} \end{bmatrix}$

Linear regression

- Hypotheses for linear regression: $h(x;\theta,\theta_0) = \theta^\top x + \theta_0$
- Training error (using squared error loss)

$$J(\theta,\theta_0) = \frac{1}{n}\sum_{i=1}^{n} L(h(x^{(i)};\theta,\theta_0), y^{(i)}) = \frac{1}{n}\sum_{i=1}^{n}(\theta^\top x^{(i)} + \theta_0 - y^{(i)})^2$$

- Training error if we augment features with feature "1"

$$J(\theta) = \frac{1}{n}\sum_{i=1}^{n}(\theta^\top x^{(i)} - y^{(i)})^2 = \frac{1}{n}\sum_{i=1}^{n}(\underbrace{(x^{(i)})^\top}_{\text{1xd,dx1}}\theta - \underbrace{y^{(i)}}_{\text{1x1}})^2$$

$$\tilde{X}\theta - \tilde{Y}$$

Define $\underset{\text{nxd}}{\tilde{X}} = \begin{bmatrix} x_1^{(1)} & \cdots & x_d^{(1)} \\ \vdots & \ddots & \vdots \\ x_1^{(n)} & \cdots & x_d^{(n)} \end{bmatrix}$ $\underset{\text{nx1}}{\tilde{Y}} = \begin{bmatrix} y^{(1)} \\ \vdots \\ y^{(n)} \end{bmatrix}$

Linear regression

- Hypotheses for linear regression: $h(x;\theta,\theta_0) = \theta^\top x + \theta_0$
- Training error (using squared error loss)

$$J(\theta,\theta_0) = \frac{1}{n}\sum_{i=1}^{n} L(h(x^{(i)};\theta,\theta_0), y^{(i)}) = \frac{1}{n}\sum_{i=1}^{n}(\theta^\top x^{(i)} + \theta_0 - y^{(i)})^2$$

- Training error if we augment features with feature "1"

$$J(\theta) = \frac{1}{n}\sum_{i=1}^{n}(\theta^\top x^{(i)} - y^{(i)})^2 = \frac{1}{n}\sum_{i=1}^{n}((x^{(i)})^\top \theta - y^{(i)})^2$$

1xd,dx1 1x1 1xd,dx1 1x1

$$\tilde{X}\theta - \tilde{Y}$$

Define $\underset{\text{nxd}}{\tilde{X}} = \begin{bmatrix} x_1^{(1)} & \cdots & x_d^{(1)} \\ \vdots & \ddots & \vdots \\ x_1^{(n)} & \cdots & x_d^{(n)} \end{bmatrix}$ $\underset{\text{nx1}}{\tilde{Y}} = \begin{bmatrix} y^{(1)} \\ \vdots \\ y^{(n)} \end{bmatrix}$

Linear regression

- Hypotheses for linear regression: $h(x;\theta,\theta_0) = \theta^\top x + \theta_0$
- Training error (using squared error loss)

$$J(\theta,\theta_0) = \frac{1}{n}\sum_{i=1}^{n} L(h(x^{(i)};\theta,\theta_0), y^{(i)}) = \frac{1}{n}\sum_{i=1}^{n}(\theta^\top x^{(i)} + \theta_0 - y^{(i)})^2$$

- Training error if we augment features with feature "1"

$$J(\theta) = \frac{1}{n}\sum_{i=1}^{n}(\theta^\top x^{(i)} - y^{(i)})^2 = \frac{1}{n}\sum_{i=1}^{n}((x^{(i)})^\top \theta - y^{(i)})^2$$

1xd,dx1 1x1 1xd,dx1 1x1

$$\tilde{X}\theta - \tilde{Y}$$

nxd

Define $\underset{\text{nxd}}{\tilde{X}} = \begin{bmatrix} x_1^{(1)} & \cdots & x_d^{(1)} \\ \vdots & \ddots & \vdots \\ x_1^{(n)} & \cdots & x_d^{(n)} \end{bmatrix}$ $\underset{\text{nx1}}{\tilde{Y}} = \begin{bmatrix} y^{(1)} \\ \vdots \\ y^{(n)} \end{bmatrix}$

Linear regression

- Hypotheses for linear regression: $h(x;\theta,\theta_0) = \theta^\top x + \theta_0$
- Training error (using squared error loss)

$$J(\theta,\theta_0) = \frac{1}{n}\sum_{i=1}^{n} L(h(x^{(i)};\theta,\theta_0), y^{(i)}) = \frac{1}{n}\sum_{i=1}^{n}(\theta^\top x^{(i)} + \theta_0 - y^{(i)})^2$$

- Training error if we augment features with feature "1"

$$J(\theta) = \frac{1}{n}\sum_{i=1}^{n}(\theta^\top x^{(i)} - y^{(i)})^2 = \frac{1}{n}\sum_{i=1}^{n}(\underbrace{(x^{(i)})^\top}_{\text{1xd,dx1}}\theta - \underbrace{y^{(i)}}_{\text{1x1}})^2$$

$$\tilde{X}\theta - \tilde{Y}$$

nxd,dx1

Define $\underset{\text{nxd}}{\tilde{X}} = \begin{bmatrix} x_1^{(1)} & \cdots & x_d^{(1)} \\ \vdots & \ddots & \vdots \\ x_1^{(n)} & \cdots & x_d^{(n)} \end{bmatrix}$ $\underset{\text{nx1}}{\tilde{Y}} = \begin{bmatrix} y^{(1)} \\ \vdots \\ y^{(n)} \end{bmatrix}$

Linear regression

- Hypotheses for linear regression: $h(x;\theta,\theta_0) = \theta^\top x + \theta_0$
- Training error (using squared error loss)

$$J(\theta,\theta_0) = \frac{1}{n}\sum_{i=1}^{n} L(h(x^{(i)};\theta,\theta_0), y^{(i)}) = \frac{1}{n}\sum_{i=1}^{n}(\theta^\top x^{(i)} + \theta_0 - y^{(i)})^2$$

- Training error if we augment features with feature "1"

$$J(\theta) = \frac{1}{n}\sum_{i=1}^{n}(\theta^\top x^{(i)} - y^{(i)})^2 = \frac{1}{n}\sum_{i=1}^{n}((x^{(i)})^\top \theta - y^{(i)})^2$$

1xd,dx1 1x1 1xd,dx1 1x1

$$\tilde{X}\theta - \tilde{Y}$$

nxd,dx1 nx1

Define $\underset{\text{nxd}}{\tilde{X}} = \begin{bmatrix} x_1^{(1)} & \cdots & x_d^{(1)} \\ \vdots & \ddots & \vdots \\ x_1^{(n)} & \cdots & x_d^{(n)} \end{bmatrix}$ $\underset{\text{nx1}}{\tilde{Y}} = \begin{bmatrix} y^{(1)} \\ \vdots \\ y^{(n)} \end{bmatrix}$

Linear regression

- Hypotheses for linear regression: $h(x;\theta,\theta_0) = \theta^\top x + \theta_0$
- Training error (using squared error loss)

$$J(\theta,\theta_0) = \frac{1}{n}\sum_{i=1}^{n} L(h(x^{(i)};\theta,\theta_0), y^{(i)}) = \frac{1}{n}\sum_{i=1}^{n}(\theta^\top x^{(i)} + \theta_0 - y^{(i)})^2$$

- Training error if we augment features with feature "1"

$$J(\theta) = \frac{1}{n}\sum_{i=1}^{n}(\theta^\top x^{(i)} - y^{(i)})^2 = \frac{1}{n}\sum_{i=1}^{n}((x^{(i)})^\top \theta - y^{(i)})^2$$

1xd,dx1 1x1 1xd,dx1 1x1

$$= \frac{1}{n}\|\tilde{X}\theta - \tilde{Y}\|^2$$

nxd,dx1 nx1

Define $\underset{\text{nxd}}{\tilde{X}} = \begin{bmatrix} x_1^{(1)} & \cdots & x_d^{(1)} \\ \vdots & \ddots & \vdots \\ x_1^{(n)} & \cdots & x_d^{(n)} \end{bmatrix}$ $\underset{\text{nx1}}{\tilde{Y}} = \begin{bmatrix} y^{(1)} \\ \vdots \\ y^{(n)} \end{bmatrix}$

Linear regression

- Hypotheses for linear regression: $h(x;\theta,\theta_0) = \theta^\top x + \theta_0$
- Training error (using squared error loss)

$$J(\theta,\theta_0) = \frac{1}{n}\sum_{i=1}^{n} L(h(x^{(i)};\theta,\theta_0), y^{(i)}) = \frac{1}{n}\sum_{i=1}^{n}(\theta^\top x^{(i)} + \theta_0 - y^{(i)})^2$$

- Training error if we augment features with feature "1"

$$J(\theta) = \frac{1}{n}\sum_{i=1}^{n}(\theta^\top x^{(i)} - y^{(i)})^2 = \frac{1}{n}\sum_{i=1}^{n}((\underbrace{x^{(i)}}_{\text{1xd,dx1}})^\top\theta - y^{(i)})^2$$

$$= \frac{1}{n}\|\tilde{X}\theta - \tilde{Y}\|^2$$

Define $\underset{\text{nxd}}{\tilde{X}} = \begin{bmatrix} x_1^{(1)} & \cdots & x_d^{(1)} \\ \vdots & \ddots & \vdots \\ x_1^{(n)} & \cdots & x_d^{(n)} \end{bmatrix}$ $\underset{\text{nx1}}{\tilde{Y}} = \begin{bmatrix} y^{(1)} \\ \vdots \\ y^{(n)} \end{bmatrix}$

Linear regression

- Hypotheses for linear regression: $h(x;\theta,\theta_0) = \theta^\top x + \theta_0$
- Training error (using squared error loss)

$$J(\theta,\theta_0) = \frac{1}{n}\sum_{i=1}^{n} L(h(x^{(i)};\theta,\theta_0), y^{(i)}) = \frac{1}{n}\sum_{i=1}^{n}(\theta^\top x^{(i)} + \theta_0 - y^{(i)})^2$$

- Training error if we augment features with feature "1"

$$J(\theta) = \frac{1}{n}\sum_{i=1}^{n}(\theta^\top x^{(i)} - y^{(i)})^2 = \frac{1}{n}\sum_{i=1}^{n}((\underbrace{x^{(i)}}_{\text{1xd,dx1}})^\top\theta - y^{(i)})^2$$

$$= \frac{1}{n}\underbrace{\|\tilde{X}\theta - \tilde{Y}\|^2}_{\text{nxd,dx1 nx1}} = \frac{1}{n}(\tilde{X}\theta - \tilde{Y})\ \ (\tilde{X}\theta - \tilde{Y})$$

?

Define $\underset{\text{nxd}}{\tilde{X}} = \begin{bmatrix} x_1^{(1)} & \cdots & x_d^{(1)} \\ \vdots & \ddots & \vdots \\ x_1^{(n)} & \cdots & x_d^{(n)} \end{bmatrix}$ $\underset{\text{nx1}}{\tilde{Y}} = \begin{bmatrix} y^{(1)} \\ \vdots \\ y^{(n)} \end{bmatrix}$

Linear regression

- Hypotheses for linear regression: $h(x;\theta,\theta_0) = \theta^\top x + \theta_0$
- Training error (using squared error loss)

$$J(\theta,\theta_0) = \frac{1}{n}\sum_{i=1}^{n} L(h(x^{(i)};\theta,\theta_0), y^{(i)}) = \frac{1}{n}\sum_{i=1}^{n}(\theta^\top x^{(i)} + \theta_0 - y^{(i)})^2$$

- Training error if we augment features with feature "1"

$$J(\theta) = \frac{1}{n}\sum_{i=1}^{n}(\theta^\top x^{(i)} - y^{(i)})^2 = \frac{1}{n}\sum_{i=1}^{n}((\underbrace{x^{(i)}}_{\text{1xd,dx1}})^\top\theta - y^{(i)})^2$$

$$= \frac{1}{n}\underbrace{\|\tilde{X}\theta - \tilde{Y}\|^2}_{\text{nxd,dx1 nx1}} = \frac{1}{n}(\tilde{X}\theta - \tilde{Y})^\top(\tilde{X}\theta - \tilde{Y})$$

Define $\underset{\text{nxd}}{\tilde{X}} = \begin{bmatrix} x_1^{(1)} & \cdots & x_d^{(1)} \\ \vdots & \ddots & \vdots \\ x_1^{(n)} & \cdots & x_d^{(n)} \end{bmatrix}$ $\underset{\text{nx1}}{\tilde{Y}} = \begin{bmatrix} y^{(1)} \\ \vdots \\ y^{(n)} \end{bmatrix}$

Linear regression

- Hypotheses for linear regression: $h(x;\theta,\theta_0) = \theta^\top x + \theta_0$
- Training error (using squared error loss)

$$J(\theta,\theta_0) = \frac{1}{n}\sum_{i=1}^{n} L(h(x^{(i)};\theta,\theta_0), y^{(i)}) = \frac{1}{n}\sum_{i=1}^{n}(\theta^\top x^{(i)} + \theta_0 - y^{(i)})^2$$

- Training error if we augment features with feature "1"

$$J(\theta) = \frac{1}{n}\sum_{i=1}^{n}(\theta^\top x^{(i)} - y^{(i)})^2 = \frac{1}{n}\sum_{i=1}^{n}((\underbrace{x^{(i)}}_{1\text{xd}})^\top \underbrace{\theta}_{\text{dx1}} - \underbrace{y^{(i)}}_{1\text{x1}})^2$$

$$= \frac{1}{n}\|\underbrace{\tilde{X}}_{\text{nxd}}\underbrace{\theta}_{\text{dx1}} - \underbrace{\tilde{Y}}_{\text{nx1}}\|^2 = \frac{1}{n}(\underbrace{\tilde{X}\theta - \tilde{Y}}_{\text{1xn}})^\top(\underbrace{\tilde{X}\theta - \tilde{Y}}_{\text{nx1}})$$

Define $\underset{\text{nxd}}{\tilde{X}} = \begin{bmatrix} x_1^{(1)} & \cdots & x_d^{(1)} \\ \vdots & \ddots & \vdots \\ x_1^{(n)} & \cdots & x_d^{(n)} \end{bmatrix}$ $\underset{\text{nx1}}{\tilde{Y}} = \begin{bmatrix} y^{(1)} \\ \vdots \\ y^{(n)} \end{bmatrix}$

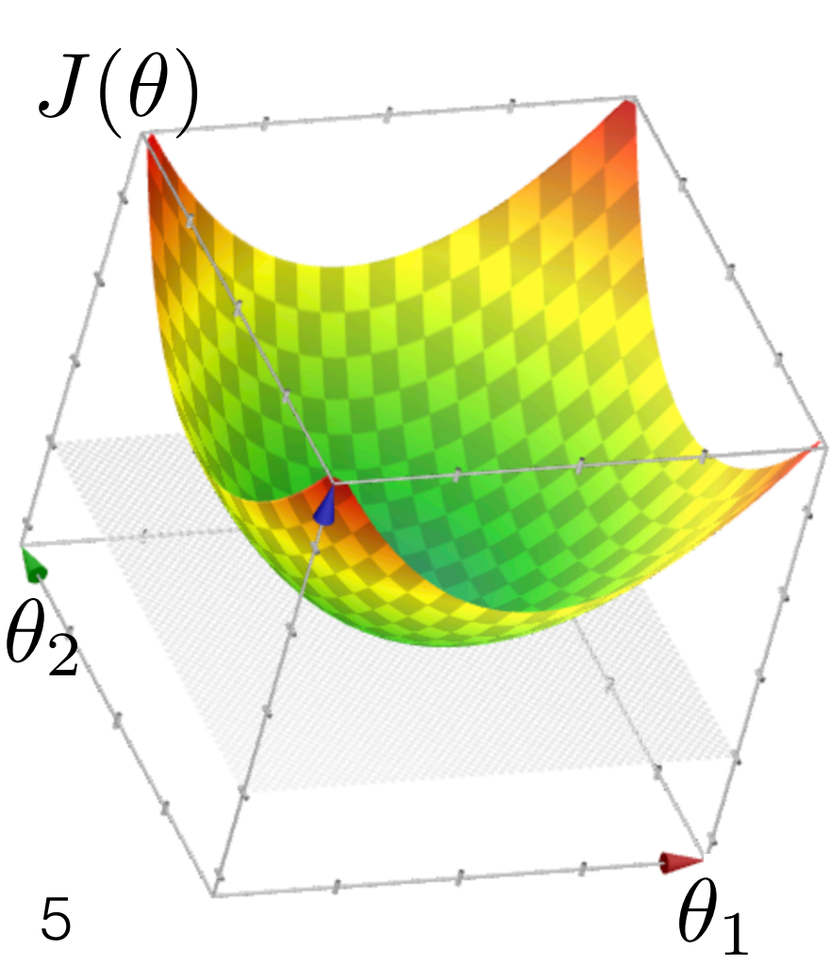
- Goal: minimize $J(\theta) = \frac{1}{n}(\tilde{X}\theta - \tilde{Y})^\top(\tilde{X}\theta - \tilde{Y})$

Linear regression: A Direct Solution

- Goal: minimize $J(\theta) = \frac{1}{n}(\tilde{X}\theta - \tilde{Y})^\top(\tilde{X}\theta - \tilde{Y})$

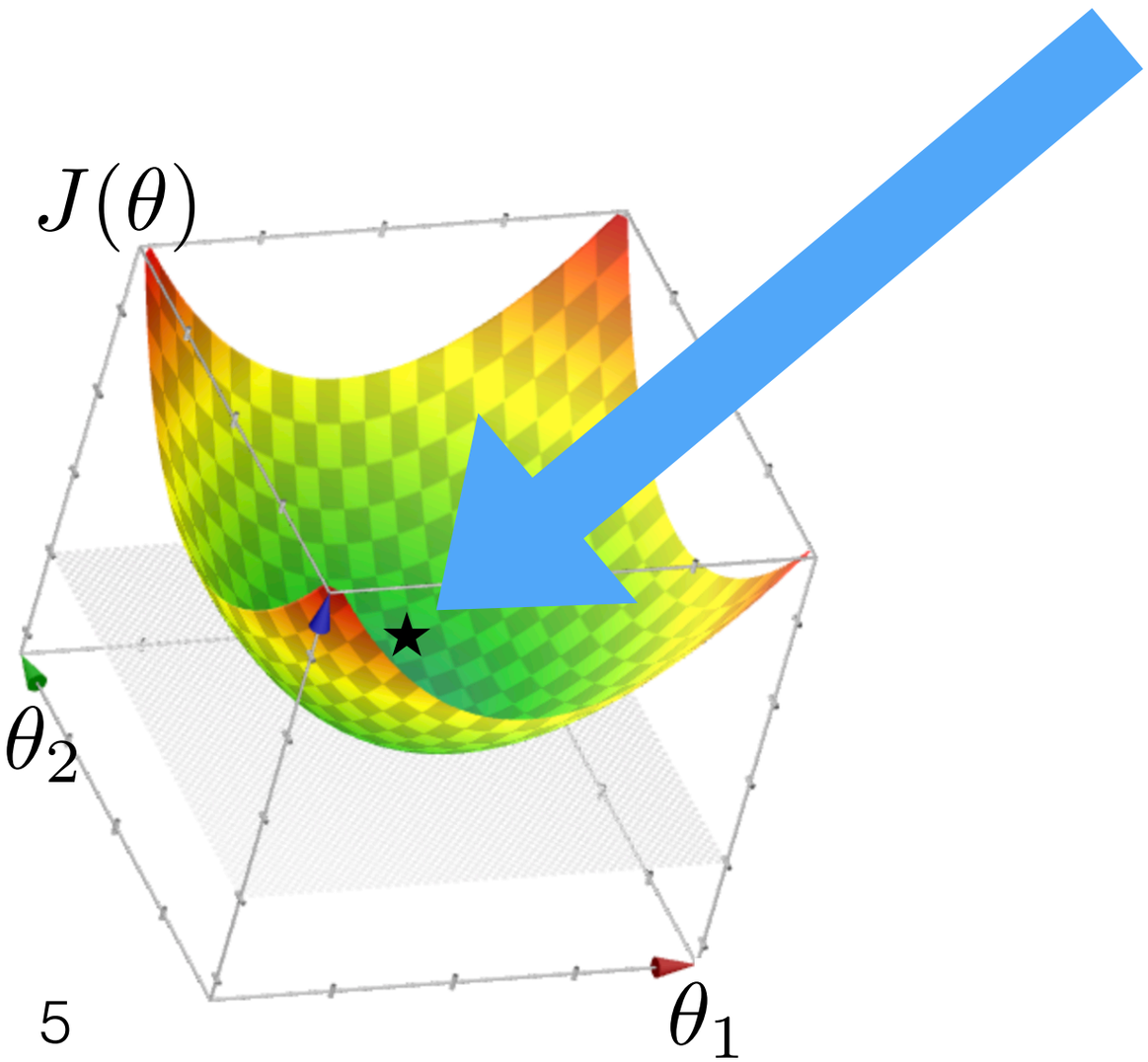
Linear regression: A Direct Solution

- Goal: minimize $J(\theta) = \frac{1}{n}(\tilde{X}\theta - \tilde{Y})^\top(\tilde{X}\theta - \tilde{Y})$



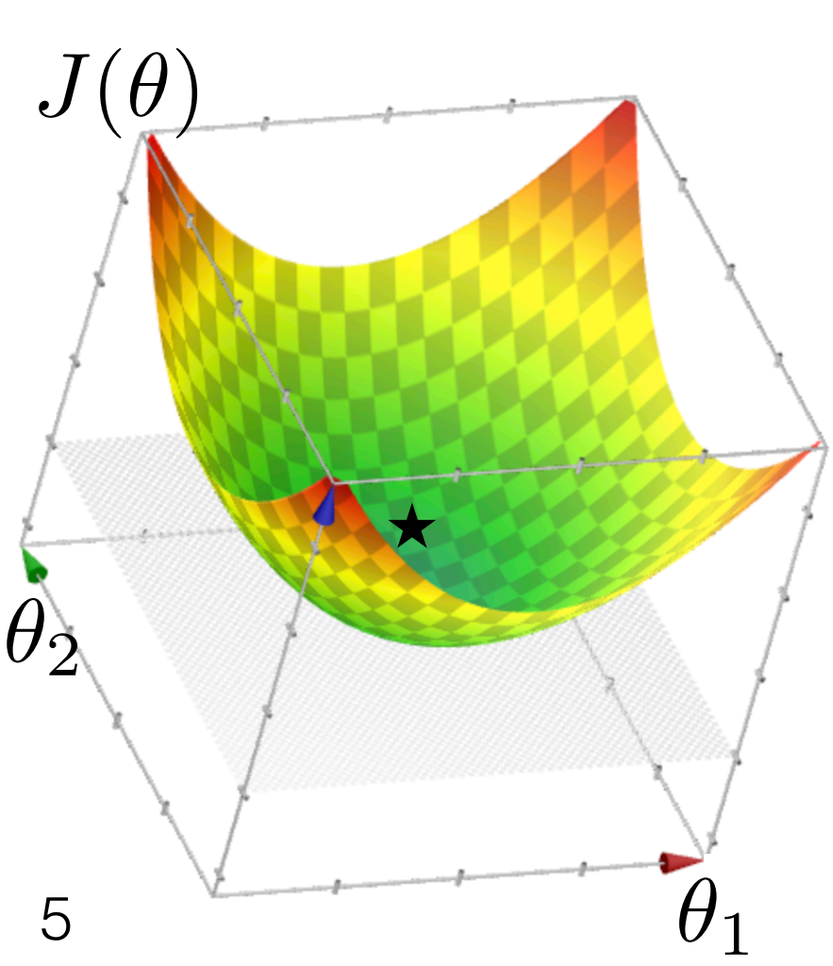
Linear regression: A Direct Solution

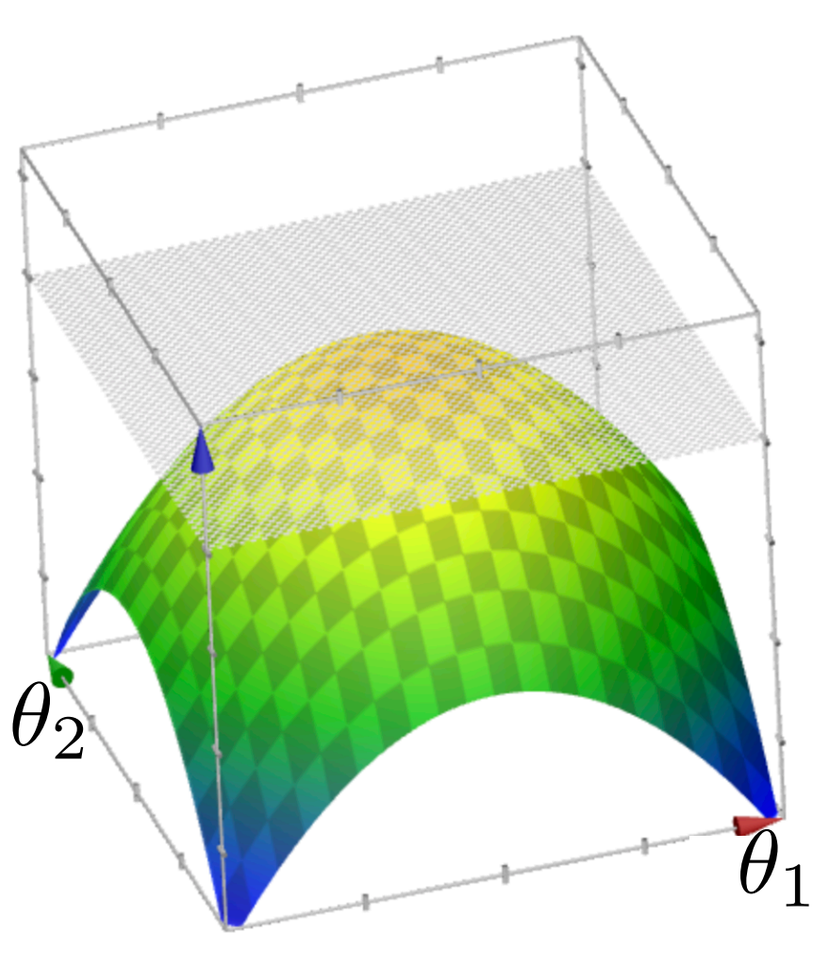
- Goal: minimize $J(\theta) = \frac{1}{n}(\tilde{X}\theta - \tilde{Y})^\top(\tilde{X}\theta - \tilde{Y})$



Linear regression: A Direct Solution

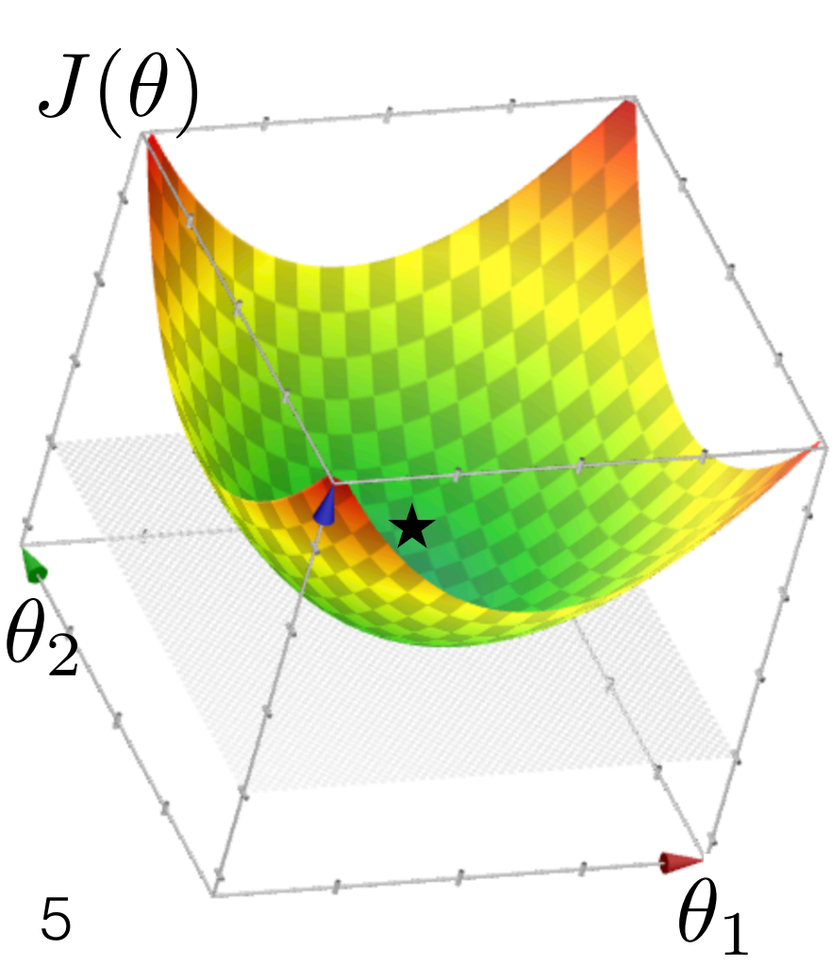
- Goal: minimize $J(\theta) = \frac{1}{n}(\tilde{X}\theta - \tilde{Y})^\top(\tilde{X}\theta - \tilde{Y})$

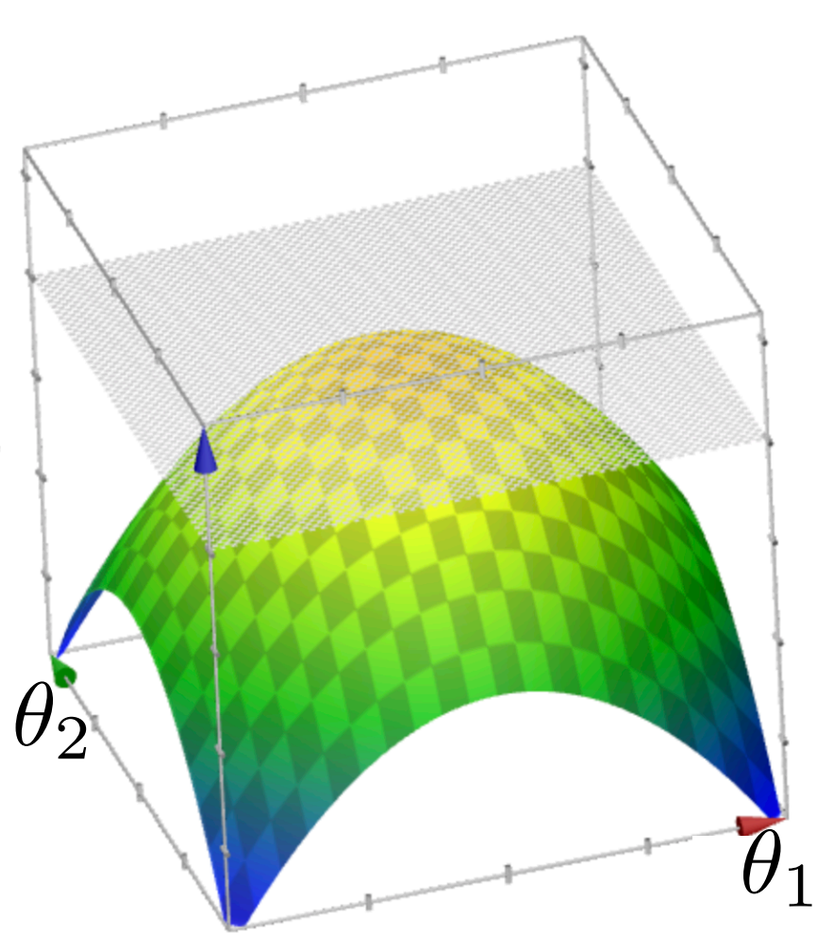


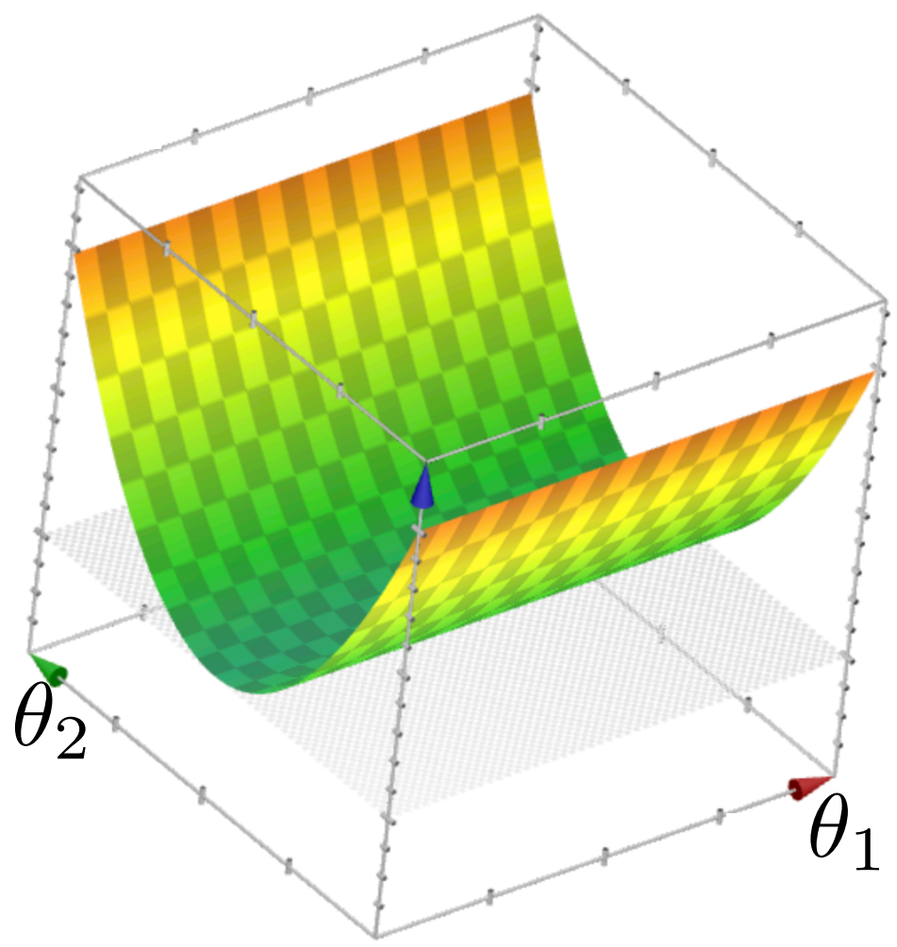


Linear regression: A Direct Solution

- Goal: minimize $J(\theta) = \frac{1}{n}(\tilde{X}\theta - \tilde{Y})^\top(\tilde{X}\theta - \tilde{Y})$

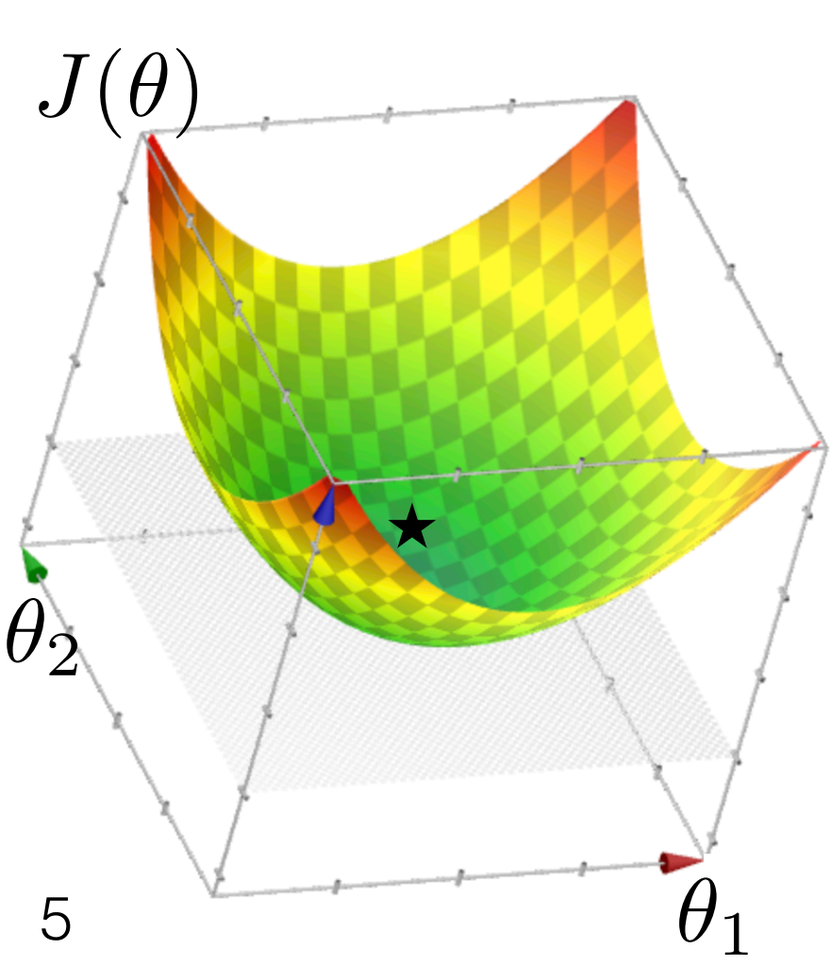


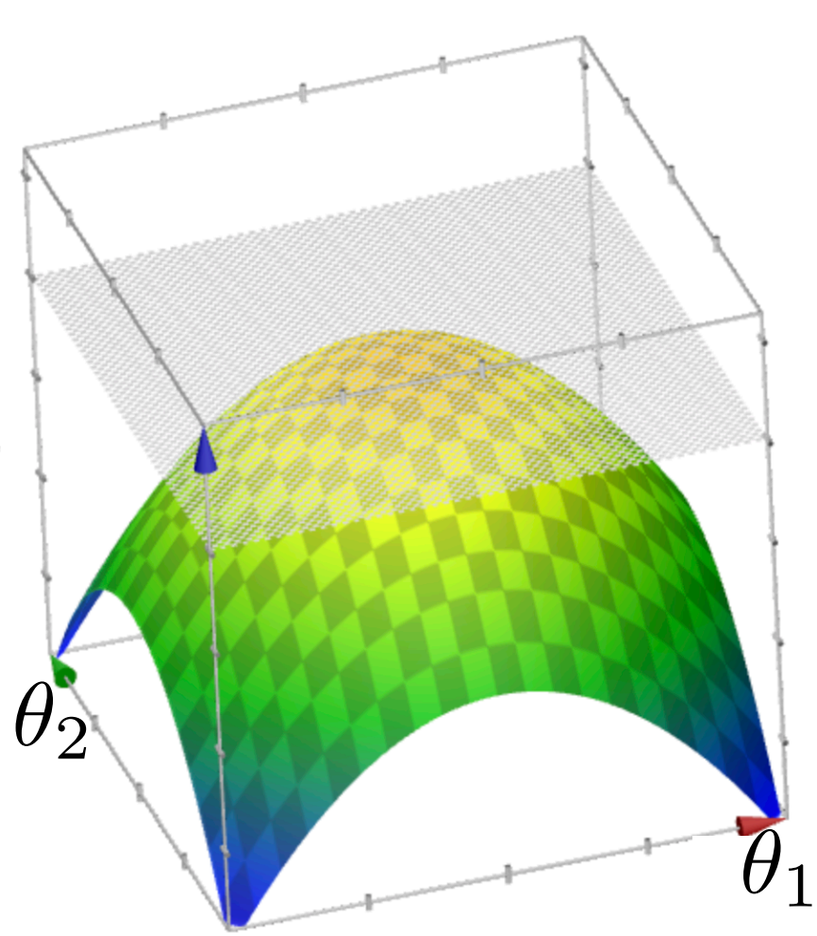


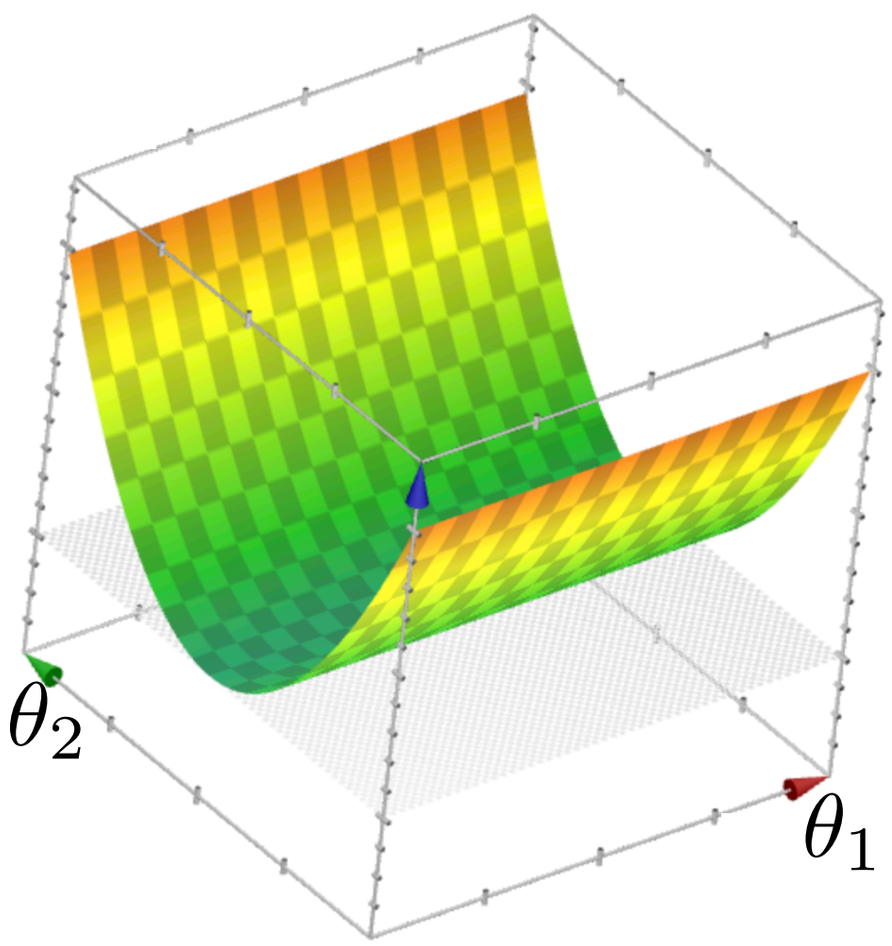


Linear regression: A Direct Solution

- Goal: minimize $J(\theta) = \frac{1}{n}(\tilde{X}\theta - \tilde{Y})^\top(\tilde{X}\theta - \tilde{Y})$
- Uniquely minimized at a point if gradient at that point is zero and function "curves up" [see linear algebra]







Linear regression: A Direct Solution

- Goal: minimize $J(\theta) = \frac{1}{n}(\tilde{X}\theta - \tilde{Y})^\top(\tilde{X}\theta - \tilde{Y})$
- Uniquely minimized at a point if gradient at that point is zero and function “curves up” [see linear algebra]

Linear regression: A Direct Solution

- Goal: minimize $J(\theta) = \frac{1}{n}(\tilde{X}\theta - \tilde{Y})^\top(\tilde{X}\theta - \tilde{Y})$
- Uniquely minimized at a point if gradient at that point is zero and function “curves up” [see linear algebra]
- Gradient

Linear regression: A Direct Solution

- Goal: minimize $J(\theta) = \frac{1}{n}(\tilde{X}\theta - \tilde{Y})^\top(\tilde{X}\theta - \tilde{Y})$
- Uniquely minimized at a point if gradient at that point is zero and function “curves up” [see linear algebra]
- Gradient $\nabla_\theta J(\theta)$

Linear regression: A Direct Solution

- Goal: minimize $J(\theta) = \frac{1}{n}(\tilde{X}\theta - \tilde{Y})^\top(\tilde{X}\theta - \tilde{Y})$
- Uniquely minimized at a point if gradient at that point is zero and function "curves up" [see linear algebra]
- Gradient $\nabla_\theta J(\theta)$

dx1

Linear regression: A Direct Solution

- Goal: minimize $J(\theta) = \frac{1}{n}(\tilde{X}\theta - \tilde{Y})^\top(\tilde{X}\theta - \tilde{Y})$
- Uniquely minimized at a point if gradient at that point is zero and function "curves up" [see linear algebra]
- Gradient $\underset{\text{dx1}}{\nabla_\theta J(\theta)} = \frac{2}{n}\tilde{X}^\top(\tilde{X}\theta - \tilde{Y})$

Linear regression: A Direct Solution

- Goal: minimize $J(\theta) = \frac{1}{n}(\tilde{X}\theta - \tilde{Y})^\top(\tilde{X}\theta - \tilde{Y})$
- Uniquely minimized at a point if gradient at that point is zero and function "curves up" [see linear algebra]
- Gradient $\underset{\text{dx1}}{\nabla_\theta J(\theta)} = \frac{2}{n}\tilde{X}^\top(\tilde{X}\theta - \tilde{Y})$

Exercise: check n,d=1

Linear regression: A Direct Solution

- Goal: minimize $J(\theta) = \frac{1}{n}(\tilde{X}\theta - \tilde{Y})^\top(\tilde{X}\theta - \tilde{Y})$
- Uniquely minimized at a point if gradient at that point is zero and function "curves up" [see linear algebra]
- Gradient $\nabla_\theta J(\theta) = \frac{2}{n}\tilde{X}^\top(\tilde{X}\theta - \tilde{Y})$

dx1

Exercise: check the vector elements

Exercise: check n,d=1

Linear regression: A Direct Solution

- Goal: minimize $J(\theta) = \frac{1}{n}(\tilde{X}\theta - \tilde{Y})^\top(\tilde{X}\theta - \tilde{Y})$
- Uniquely minimized at a point if gradient at that point is zero and function "curves up" [see linear algebra]
- Gradient $\nabla_\theta J(\theta) = \frac{2}{n}\tilde{X}^\top(\tilde{X}\theta - \tilde{Y})$

dx1 nxd

Exercise: check the vector elements

Exercise: check n,d=1

Linear regression: A Direct Solution

- Goal: minimize $J(\theta) = \frac{1}{n}(\tilde{X}\theta - \tilde{Y})^\top(\tilde{X}\theta - \tilde{Y})$
- Uniquely minimized at a point if gradient at that point is zero and function "curves up" [see linear algebra]
- Gradient $\nabla_\theta J(\theta) = \frac{2}{n}\tilde{X}^\top(\tilde{X}\theta - \tilde{Y})$

dx1 nxd,dx1

Exercise: check the vector elements

Exercise: check n,d=1

Linear regression: A Direct Solution

- Goal: minimize $J(\theta) = \frac{1}{n}(\tilde{X}\theta - \tilde{Y})^\top(\tilde{X}\theta - \tilde{Y})$
- Uniquely minimized at a point if gradient at that point is zero and function "curves up" [see linear algebra]
- Gradient $\nabla_\theta J(\theta) = \frac{2}{n}\tilde{X}^\top(\tilde{X}\theta - \tilde{Y})$

dx1 nxd,dx1 nx1

Exercise: check the vector elements

Exercise: check n,d=1

Linear regression: A Direct Solution

- Goal: minimize $J(\theta) = \frac{1}{n}(\tilde{X}\theta - \tilde{Y})^\top(\tilde{X}\theta - \tilde{Y})$
- Uniquely minimized at a point if gradient at that point is zero and function "curves up" [see linear algebra]
- Gradient $\nabla_\theta J(\theta) = \frac{2}{n}\tilde{X}^\top(\tilde{X}\theta - \tilde{Y})$

dx1 dxn nxd,dx1 nx1

Exercise: check the vector elements

Exercise: check n,d=1

Linear regression: A Direct Solution

- Goal: minimize $J(\theta) = \frac{1}{n}(\tilde{X}\theta - \tilde{Y})^\top(\tilde{X}\theta - \tilde{Y})$
- Uniquely minimized at a point if gradient at that point is zero and function "curves up" [see linear algebra]
- Gradient $\nabla_\theta J(\theta) = \frac{2}{n}\tilde{X}^\top(\tilde{X}\theta - \tilde{Y}) \overset{\text{set}}{=} 0$

dx1 dxn nxd,dx1 nx1

Exercise: check the vector elements

Exercise: check n,d=1

Linear regression: A Direct Solution

- Goal: minimize $J(\theta) = \frac{1}{n}(\tilde{X}\theta - \tilde{Y})^\top(\tilde{X}\theta - \tilde{Y})$
- Uniquely minimized at a point if gradient at that point is zero and function "curves up" [see linear algebra]
- Gradient $\nabla_\theta J(\theta) = \frac{2}{n}\tilde{X}^\top(\tilde{X}\theta - \tilde{Y}) \overset{\text{set}}{=} 0$

dx1 — dxn nxd,dx1 nx1

$$\tilde{X}^\top\tilde{X}\theta - \tilde{X}^\top\tilde{Y} = 0$$

Exercise: check the vector elements

Exercise: check n,d=1

Linear regression: A Direct Solution

- Goal: minimize $J(\theta) = \frac{1}{n}(\tilde{X}\theta - \tilde{Y})^\top(\tilde{X}\theta - \tilde{Y})$
- Uniquely minimized at a point if gradient at that point is zero and function "curves up" [see linear algebra]
- Gradient $\nabla_\theta J(\theta) = \frac{2}{n}\tilde{X}^\top(\tilde{X}\theta - \tilde{Y}) \overset{\text{set}}{=} 0$

dx1 — dxn nxd,dx1 nx1

$$\tilde{X}^\top\tilde{X}\theta - \tilde{X}^\top\tilde{Y} = 0$$

$$\tilde{X}^\top\tilde{X}\theta = \tilde{X}^\top\tilde{Y}$$

Exercise: check the vector elements

Exercise: check n,d=1

Linear regression: A Direct Solution

- Goal: minimize $J(\theta) = \frac{1}{n}(\tilde{X}\theta - \tilde{Y})^\top(\tilde{X}\theta - \tilde{Y})$
- Uniquely minimized at a point if gradient at that point is zero and function "curves up" [see linear algebra]
- Gradient $\nabla_\theta J(\theta) = \frac{2}{n}\tilde{X}^\top(\tilde{X}\theta - \tilde{Y}) \overset{\text{set}}{=} 0$

dx1 dxn nxd,dx1 nx1

$$\tilde{X}^\top\tilde{X}\theta - \tilde{X}^\top\tilde{Y} = 0$$

$$(\tilde{X}^\top\tilde{X})^{-1}\tilde{X}^\top\tilde{X}\theta = (\tilde{X}^\top\tilde{X})^{-1}\tilde{X}^\top\tilde{Y}$$

Exercise: check the vector elements

Exercise: check n,d=1

Linear regression: A Direct Solution

- Goal: minimize $J(\theta) = \frac{1}{n}(\tilde{X}\theta - \tilde{Y})^\top(\tilde{X}\theta - \tilde{Y})$
- Uniquely minimized at a point if gradient at that point is zero and function "curves up" [see linear algebra]
- Gradient $\nabla_\theta J(\theta) = \frac{2}{n}\tilde{X}^\top(\tilde{X}\theta - \tilde{Y}) \overset{\text{set}}{=} 0$

dx1 — dxn nxd,dx1 nx1

Exercise: check the vector elements

Exercise: check n,d=1

$$\tilde{X}^\top\tilde{X}\theta - \tilde{X}^\top\tilde{Y} = 0$$

$$(\tilde{X}^\top\tilde{X})^{-1}\tilde{X}^\top\tilde{X}\theta = (\tilde{X}^\top\tilde{X})^{-1}\tilde{X}^\top\tilde{Y}$$

$$\theta = (\tilde{X}^\top\tilde{X})^{-1}\tilde{X}^\top\tilde{Y}$$

Linear regression: A Direct Solution

- Goal: minimize $J(\theta) = \frac{1}{n}(\tilde{X}\theta - \tilde{Y})^\top(\tilde{X}\theta - \tilde{Y})$
- Uniquely minimized at a point if gradient at that point is zero and function "curves up" [see linear algebra]
- Gradient $\nabla_\theta J(\theta) = \frac{2}{n}\tilde{X}^\top(\tilde{X}\theta - \tilde{Y}) \overset{\text{set}}{=} 0$

dx1 dxn nxd,dx1 nx1

$$\tilde{X}^\top\tilde{X}\theta - \tilde{X}^\top\tilde{Y} = 0$$

$$(\tilde{X}^\top\tilde{X})^{-1}\tilde{X}^\top\tilde{X}\theta = (\tilde{X}^\top\tilde{X})^{-1}\tilde{X}^\top\tilde{Y}$$

$$\theta = (\tilde{X}^\top\tilde{X})^{-1}\tilde{X}^\top\tilde{Y}$$

Exercise: check the vector elements

Exercise: check n,d=1

- Matrix of second derivatives $\frac{2}{n}\tilde{X}^\top\tilde{X}$

Linear regression: A Direct Solution

- Goal: minimize $J(\theta) = \frac{1}{n}(\tilde{X}\theta - \tilde{Y})^\top(\tilde{X}\theta - \tilde{Y})$
- Uniquely minimized at a point if gradient at that point is zero and function "curves up" [see linear algebra]
- Gradient $\nabla_\theta J(\theta) = \frac{2}{n}\tilde{X}^\top(\tilde{X}\theta - \tilde{Y}) \overset{\text{set}}{=} 0$

dx1 dxn nxd,dx1 nx1

Exercise: check the vector elements

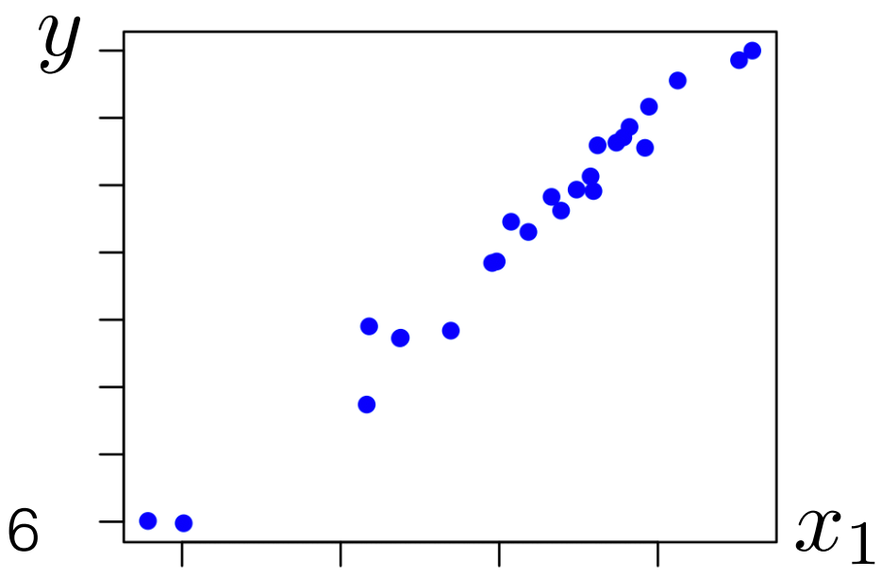
Exercise: check n,d=1

$$\tilde{X}^\top\tilde{X}\theta - \tilde{X}^\top\tilde{Y} = 0$$

$$(\tilde{X}^\top\tilde{X})^{-1}\tilde{X}^\top\tilde{X}\theta = (\tilde{X}^\top\tilde{X})^{-1}\tilde{X}^\top\tilde{Y}$$

$$\theta = (\tilde{X}^\top\tilde{X})^{-1}\tilde{X}^\top\tilde{Y}$$

- Matrix of second derivatives $\frac{2}{n}\tilde{X}^\top\tilde{X}$



Linear regression: A Direct Solution

- Goal: minimize $J(\theta) = \frac{1}{n}(\tilde{X}\theta - \tilde{Y})^\top(\tilde{X}\theta - \tilde{Y})$
- Uniquely minimized at a point if gradient at that point is zero and function "curves up" [see linear algebra]
- Gradient $\nabla_\theta J(\theta) = \frac{2}{n}\tilde{X}^\top(\tilde{X}\theta - \tilde{Y}) \overset{\text{set}}{=} 0$

dx1 dxn nxd,dx1 nx1

Exercise: check the vector elements

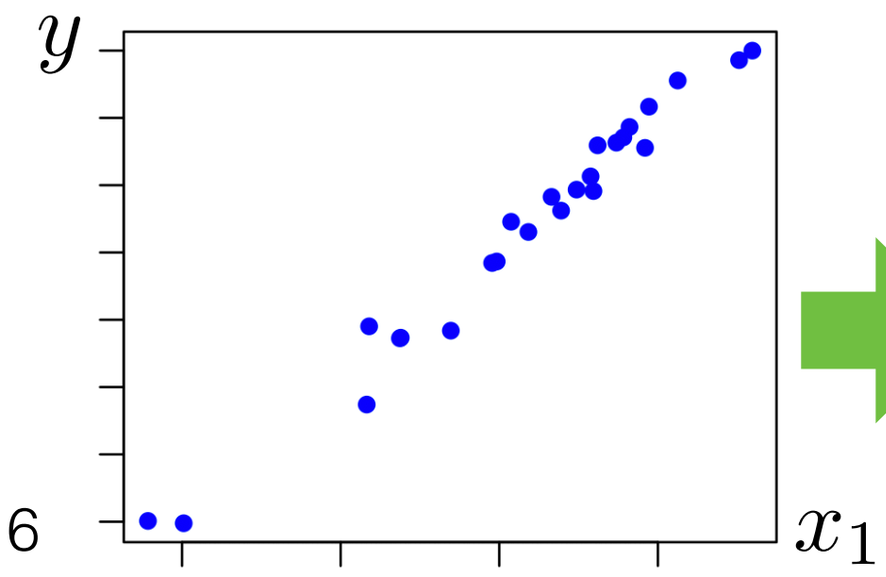
Exercise: check n,d=1

$$\tilde{X}^\top\tilde{X}\theta - \tilde{X}^\top\tilde{Y} = 0$$

$$(\tilde{X}^\top\tilde{X})^{-1}\tilde{X}^\top\tilde{X}\theta = (\tilde{X}^\top\tilde{X})^{-1}\tilde{X}^\top\tilde{Y}$$

$$\theta = (\tilde{X}^\top\tilde{X})^{-1}\tilde{X}^\top\tilde{Y}$$

- Matrix of second derivatives $\frac{2}{n}\tilde{X}^\top\tilde{X}$



Linear regression: A Direct Solution

- Goal: minimize $J(\theta) = \frac{1}{n}(\tilde{X}\theta - \tilde{Y})^\top(\tilde{X}\theta - \tilde{Y})$
- Uniquely minimized at a point if gradient at that point is zero and function "curves up" [see linear algebra]
- Gradient $\nabla_\theta J(\theta) = \frac{2}{n}\tilde{X}^\top(\tilde{X}\theta - \tilde{Y}) \overset{\text{set}}{=} 0$

dx1 dxn nxd,dx1 nx1

Exercise: check the vector elements

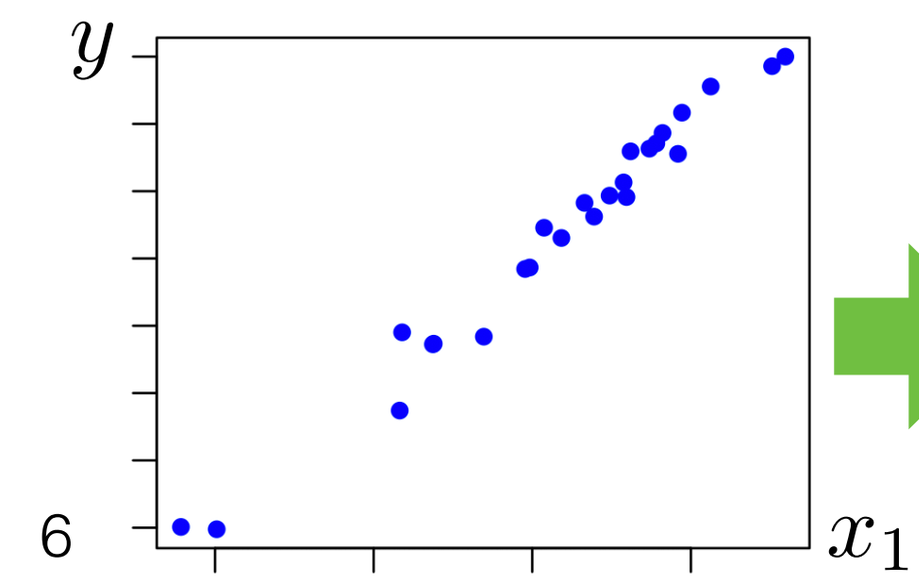
Exercise: check n,d=1

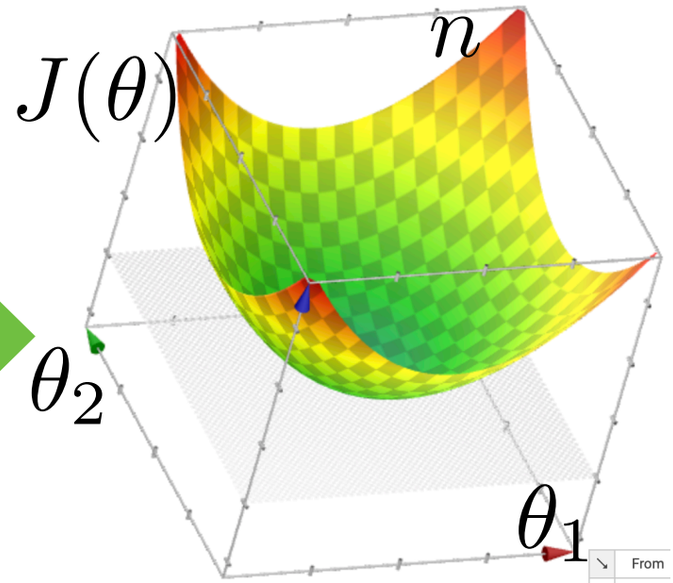
$$\tilde{X}^\top\tilde{X}\theta - \tilde{X}^\top\tilde{Y} = 0$$

$$(\tilde{X}^\top\tilde{X})^{-1}\tilde{X}^\top\tilde{X}\theta = (\tilde{X}^\top\tilde{X})^{-1}\tilde{X}^\top\tilde{Y}$$

$$\theta = (\tilde{X}^\top\tilde{X})^{-1}\tilde{X}^\top\tilde{Y}$$

- Matrix of second derivatives $\frac{2}{n}\tilde{X}^\top\tilde{X}$





Linear regression: A Direct Solution

- Goal: minimize $J(\theta) = \frac{1}{n}(\tilde{X}\theta - \tilde{Y})^\top(\tilde{X}\theta - \tilde{Y})$
- Uniquely minimized at a point if gradient at that point is zero and function "curves up" [see linear algebra]
- Gradient $\nabla_\theta J(\theta) = \frac{2}{n}\tilde{X}^\top(\tilde{X}\theta - \tilde{Y}) \overset{\text{set}}{=} 0$

dx1 dxn nxd,dx1 nx1

Exercise: check the vector elements

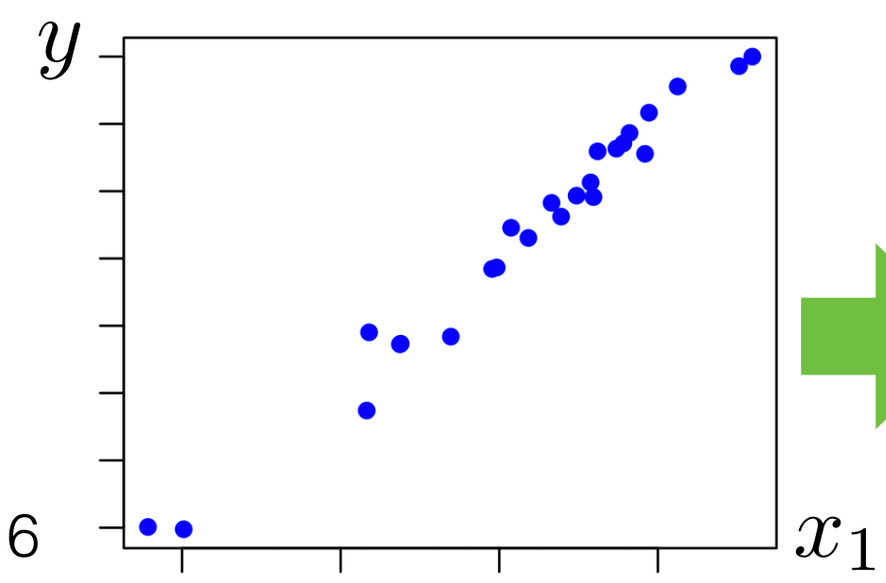
Exercise: check n,d=1

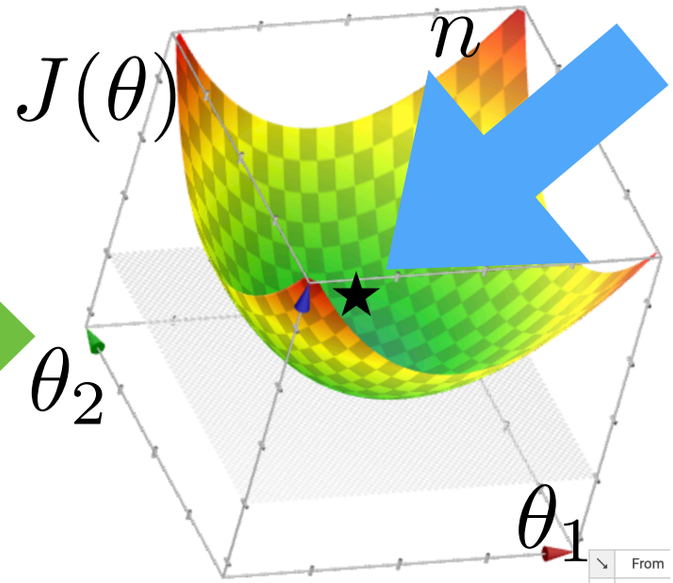
$$\tilde{X}^\top\tilde{X}\theta - \tilde{X}^\top\tilde{Y} = 0$$

$$(\tilde{X}^\top\tilde{X})^{-1}\tilde{X}^\top\tilde{X}\theta = (\tilde{X}^\top\tilde{X})^{-1}\tilde{X}^\top\tilde{Y}$$

$$\theta = (\tilde{X}^\top\tilde{X})^{-1}\tilde{X}^\top\tilde{Y}$$

- Matrix of second derivatives $\frac{2}{n}\tilde{X}^\top\tilde{X}$





Linear regression: A Direct Solution

- Goal: minimize $J(\theta) = \frac{1}{n}(\tilde{X}\theta - \tilde{Y})^\top(\tilde{X}\theta - \tilde{Y})$
- Uniquely minimized at a point if gradient at that point is zero and function "curves up" [see linear algebra]
- Gradient $\nabla_\theta J(\theta) = \frac{2}{n}\tilde{X}^\top(\tilde{X}\theta - \tilde{Y}) \overset{\text{set}}{=} 0$

dx1 dxn nxd,dx1 nx1

Exercise: check the vector elements

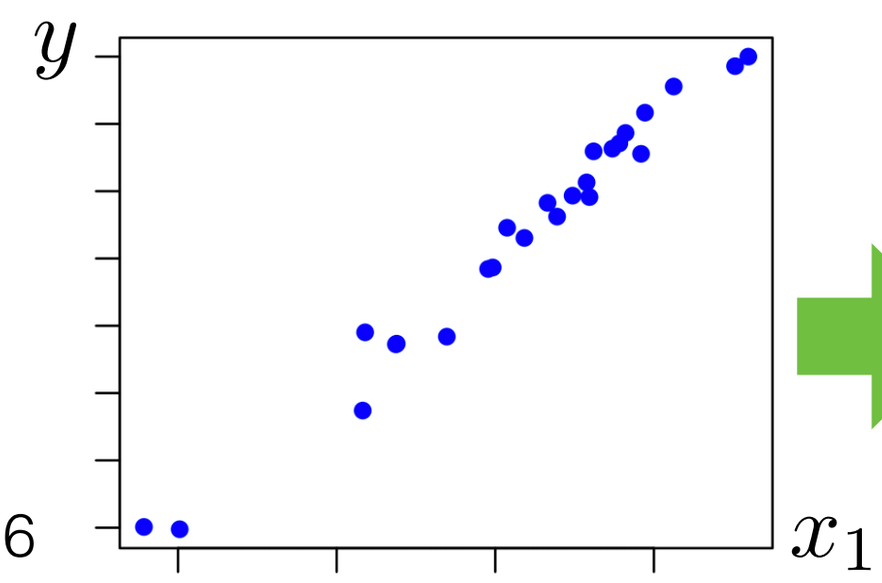
Exercise: check n,d=1

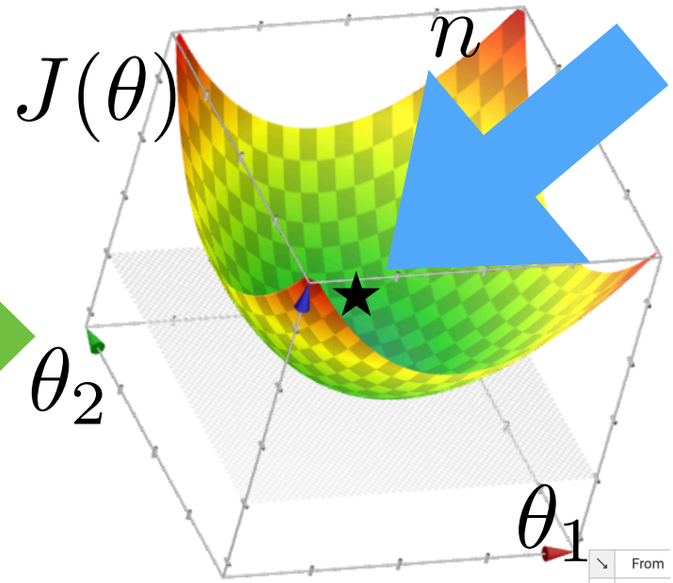
$$\tilde{X}^\top\tilde{X}\theta - \tilde{X}^\top\tilde{Y} = 0$$

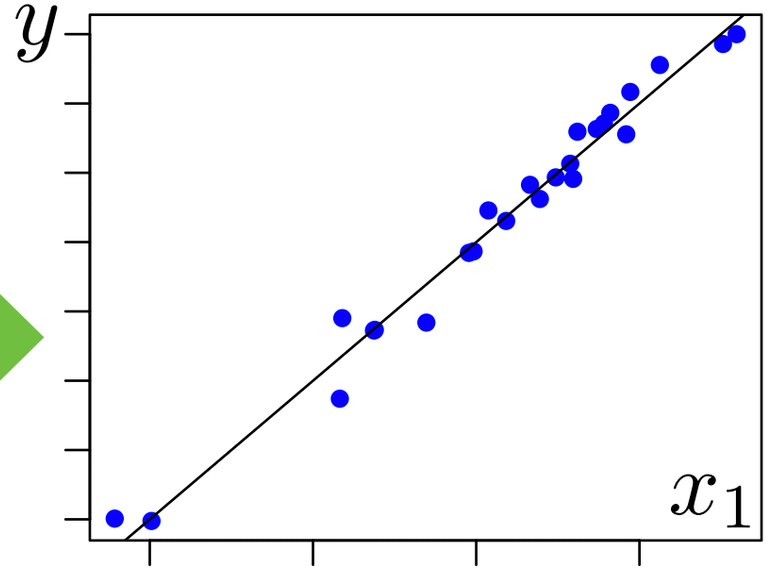
$$(\tilde{X}^\top\tilde{X})^{-1}\tilde{X}^\top\tilde{X}\theta = (\tilde{X}^\top\tilde{X})^{-1}\tilde{X}^\top\tilde{Y}$$

$$\theta = (\tilde{X}^\top\tilde{X})^{-1}\tilde{X}^\top\tilde{Y}$$

- Matrix of second derivatives $\frac{2}{n}\tilde{X}^\top\tilde{X}$







Linear regression: A Direct Solution

- Goal: minimize $J(\theta) = \frac{1}{n}(\tilde{X}\theta - \tilde{Y})^\top(\tilde{X}\theta - \tilde{Y})$
- Uniquely minimized at a point if gradient at that point is zero and function "curves up" [see linear algebra]
- Gradient $\nabla_\theta J(\theta) = \frac{2}{n}\tilde{X}^\top(\tilde{X}\theta - \tilde{Y}) \overset{\text{set}}{=} 0$

dx1 dxn nxd,dx1 nx1

Exercise: check the vector elements

Exercise: check n,d=1

$$\tilde{X}^\top\tilde{X}\theta - \tilde{X}^\top\tilde{Y} = 0$$

$$(\tilde{X}^\top\tilde{X})^{-1}\tilde{X}^\top\tilde{X}\theta = (\tilde{X}^\top\tilde{X})^{-1}\tilde{X}^\top\tilde{Y}$$

$$\theta = (\tilde{X}^\top\tilde{X})^{-1}\tilde{X}^\top\tilde{Y}$$

- Matrix of second derivatives $\frac{2}{n}\tilde{X}^\top\tilde{X}$

Linear regression: A Direct Solution

- Goal: minimize $J(\theta) = \frac{1}{n}(\tilde{X}\theta - \tilde{Y})^\top(\tilde{X}\theta - \tilde{Y})$
- Uniquely minimized at a point if gradient at that point is zero and function "curves up" [see linear algebra]
- Gradient $\nabla_\theta J(\theta) = \frac{2}{n}\tilde{X}^\top(\tilde{X}\theta - \tilde{Y}) \overset{\text{set}}{=} 0$

dx1 dxn nxd,dx1 nx1

Exercise: check the vector elements

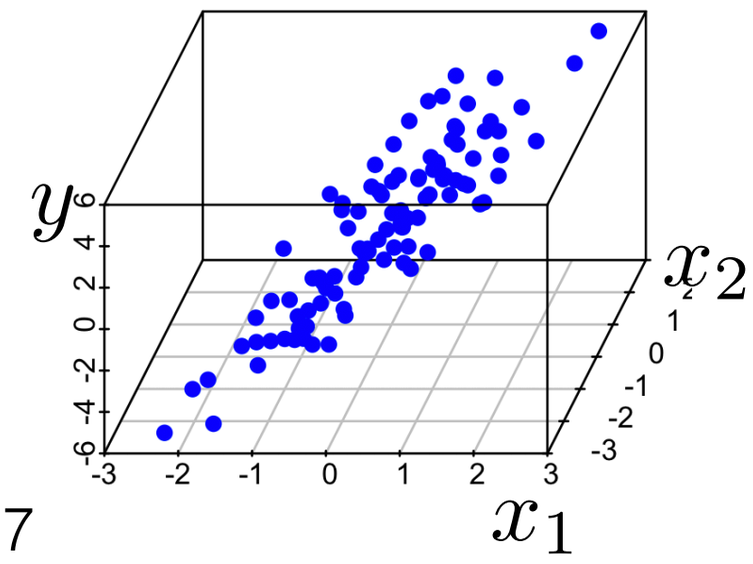
Exercise: check n,d=1

$$\tilde{X}^\top\tilde{X}\theta - \tilde{X}^\top\tilde{Y} = 0$$

$$(\tilde{X}^\top\tilde{X})^{-1}\tilde{X}^\top\tilde{X}\theta = (\tilde{X}^\top\tilde{X})^{-1}\tilde{X}^\top\tilde{Y}$$

$$\theta = (\tilde{X}^\top\tilde{X})^{-1}\tilde{X}^\top\tilde{Y}$$

- Matrix of second derivatives $\frac{2}{n}\tilde{X}^\top\tilde{X}$



Linear regression: A Direct Solution

- Goal: minimize $J(\theta) = \frac{1}{n}(\tilde{X}\theta - \tilde{Y})^\top(\tilde{X}\theta - \tilde{Y})$
- Uniquely minimized at a point if gradient at that point is zero and function "curves up" [see linear algebra]
- Gradient $\nabla_\theta J(\theta) = \frac{2}{n}\tilde{X}^\top(\tilde{X}\theta - \tilde{Y}) \overset{\text{set}}{=} 0$

dx1 dxn nxd,dx1 nx1

Exercise: check the vector elements

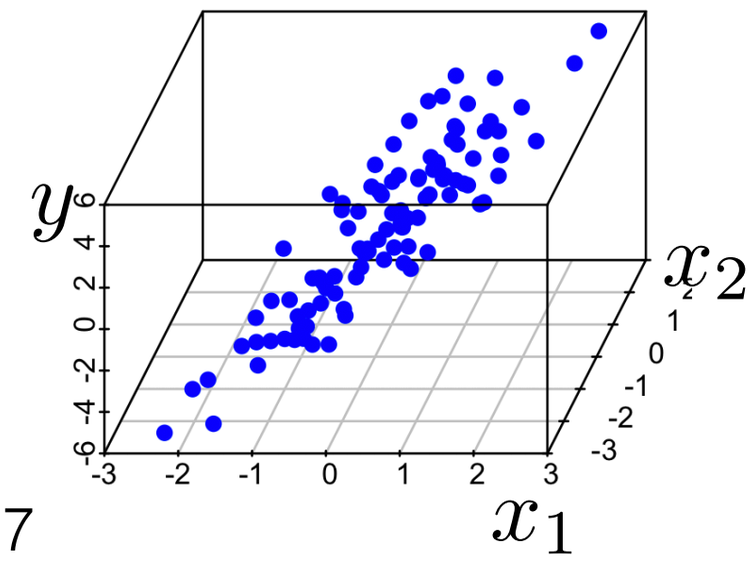
Exercise: check n,d=1

$$\tilde{X}^\top\tilde{X}\theta - \tilde{X}^\top\tilde{Y} = 0$$

$$(\tilde{X}^\top\tilde{X})^{-1}\tilde{X}^\top\tilde{X}\theta = (\tilde{X}^\top\tilde{X})^{-1}\tilde{X}^\top\tilde{Y}$$

$$\theta = (\tilde{X}^\top\tilde{X})^{-1}\tilde{X}^\top\tilde{Y}$$

- Matrix of second derivatives $\frac{2}{n}\tilde{X}^\top\tilde{X}$



Linear regression: A Direct Solution

- Goal: minimize $J(\theta) = \frac{1}{n}(\tilde{X}\theta - \tilde{Y})^\top(\tilde{X}\theta - \tilde{Y})$
- Uniquely minimized at a point if gradient at that point is zero and function "curves up" [see linear algebra]
- Gradient $\nabla_\theta J(\theta) = \frac{2}{n}\tilde{X}^\top(\tilde{X}\theta - \tilde{Y}) \overset{\text{set}}{=} 0$

dx1 dxn nxd,dx1 nx1

Exercise: check the vector elements

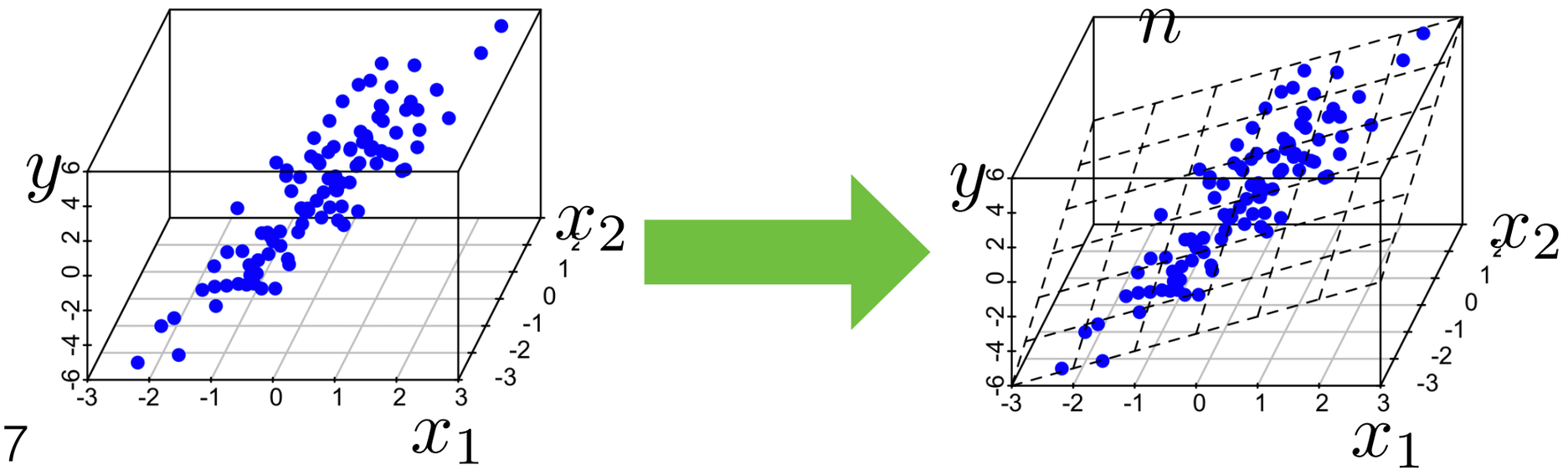
Exercise: check n,d=1

$$\tilde{X}^\top\tilde{X}\theta - \tilde{X}^\top\tilde{Y} = 0$$

$$(\tilde{X}^\top\tilde{X})^{-1}\tilde{X}^\top\tilde{X}\theta = (\tilde{X}^\top\tilde{X})^{-1}\tilde{X}^\top\tilde{Y}$$

$$\theta = (\tilde{X}^\top\tilde{X})^{-1}\tilde{X}^\top\tilde{Y}$$

- Matrix of second derivatives $\frac{2}{n}\tilde{X}^\top\tilde{X}$



Linear regression: A Direct Solution

- Goal: minimize $J(\theta) = \frac{1}{n}(\tilde{X}\theta - \tilde{Y})^\top(\tilde{X}\theta - \tilde{Y})$
- Uniquely minimized at a point if gradient at that point is zero and function "curves up" [see linear algebra]
- Gradient $\nabla_\theta J(\theta) = \frac{2}{n}\tilde{X}^\top(\tilde{X}\theta - \tilde{Y}) \overset{\text{set}}{=} 0$

dx1 dxn nxd,dx1 nx1

Exercise: check the vector elements

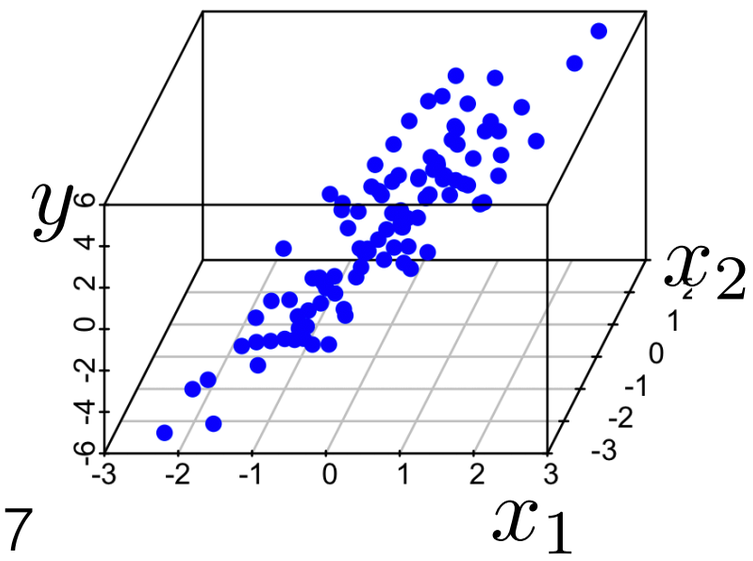
Exercise: check n,d=1

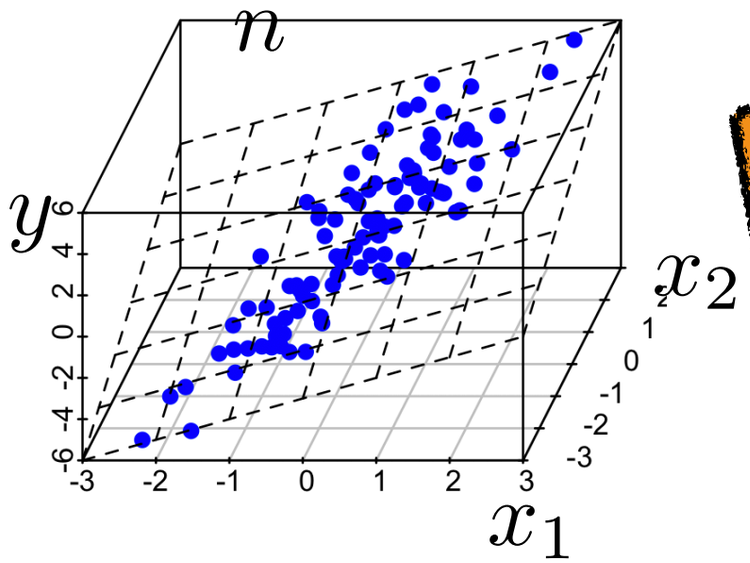
$$\tilde{X}^\top\tilde{X}\theta - \tilde{X}^\top\tilde{Y} = 0$$

$$(\tilde{X}^\top\tilde{X})^{-1}\tilde{X}^\top\tilde{X}\theta = (\tilde{X}^\top\tilde{X})^{-1}\tilde{X}^\top\tilde{Y}$$

$$\theta = (\tilde{X}^\top\tilde{X})^{-1}\tilde{X}^\top\tilde{Y}$$

- Matrix of second derivatives $\frac{2}{n}\tilde{X}^\top\tilde{X}$





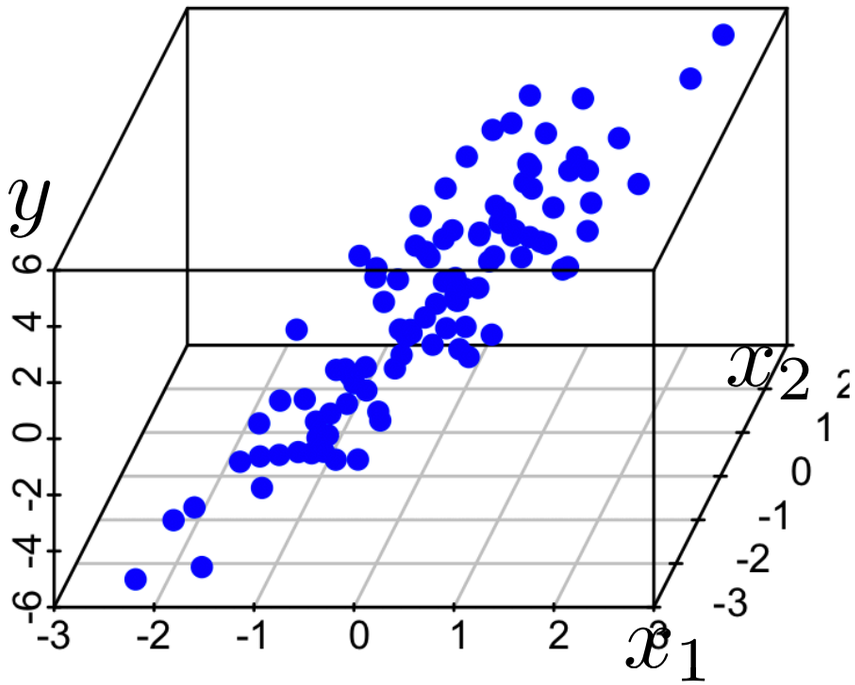
Note: hypothesis is a hyperplane!

What can go wrong in practice?

- Sometimes there isn't a unique best hyperplane

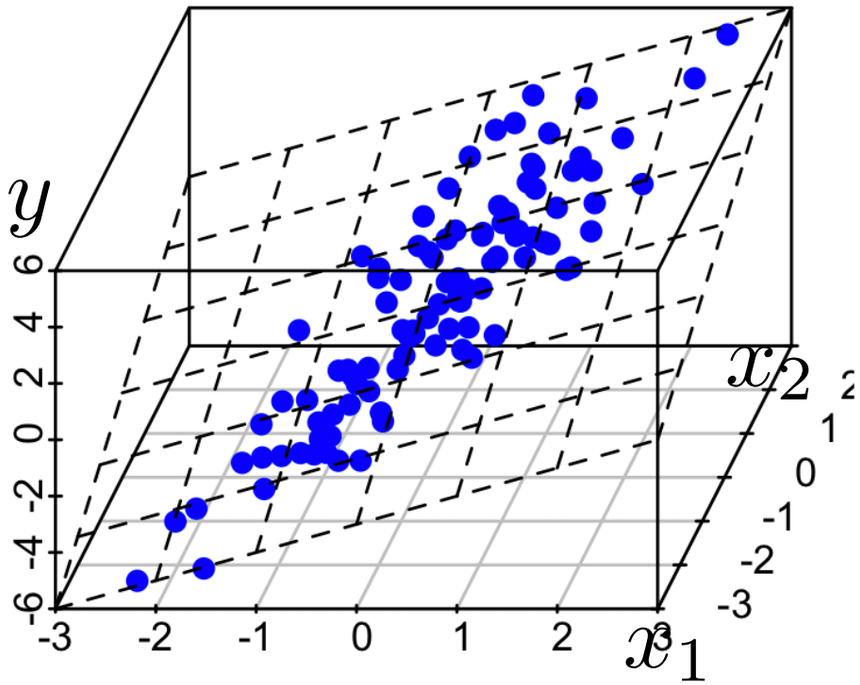
What can go wrong in practice?

- Sometimes there isn't a unique best hyperplane



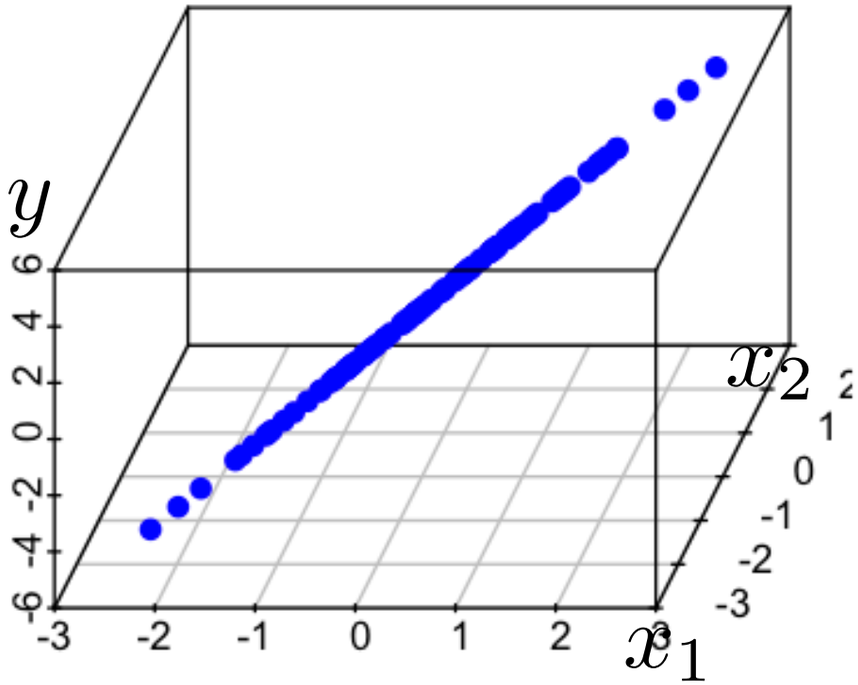
What can go wrong in practice?

- Sometimes there isn't a unique best hyperplane



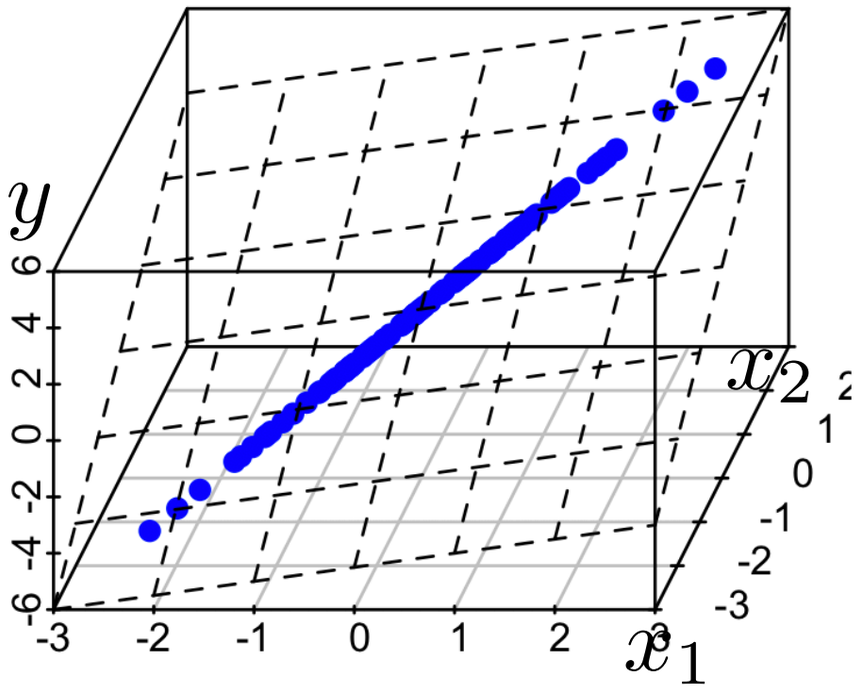
What can go wrong in practice?

- Sometimes there isn't a unique best hyperplane



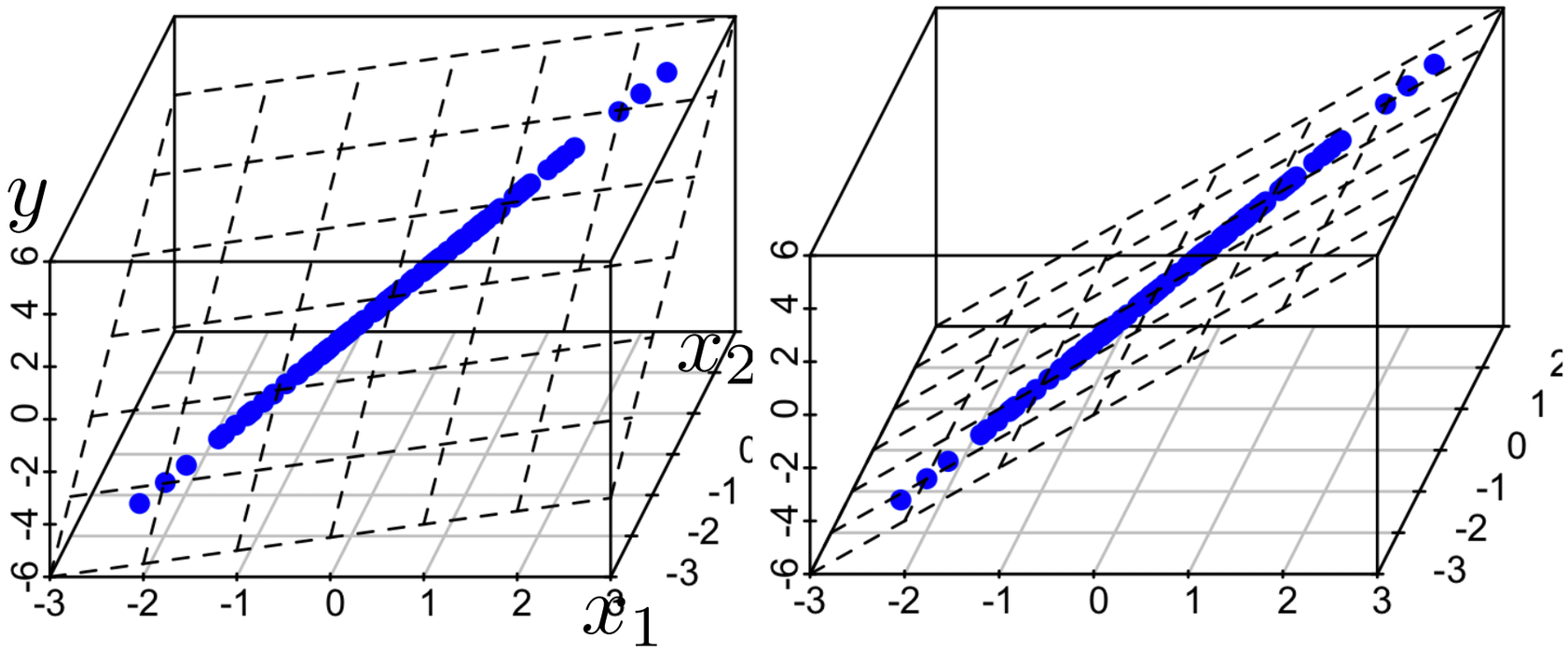
What can go wrong in practice?

- Sometimes there isn't a unique best hyperplane



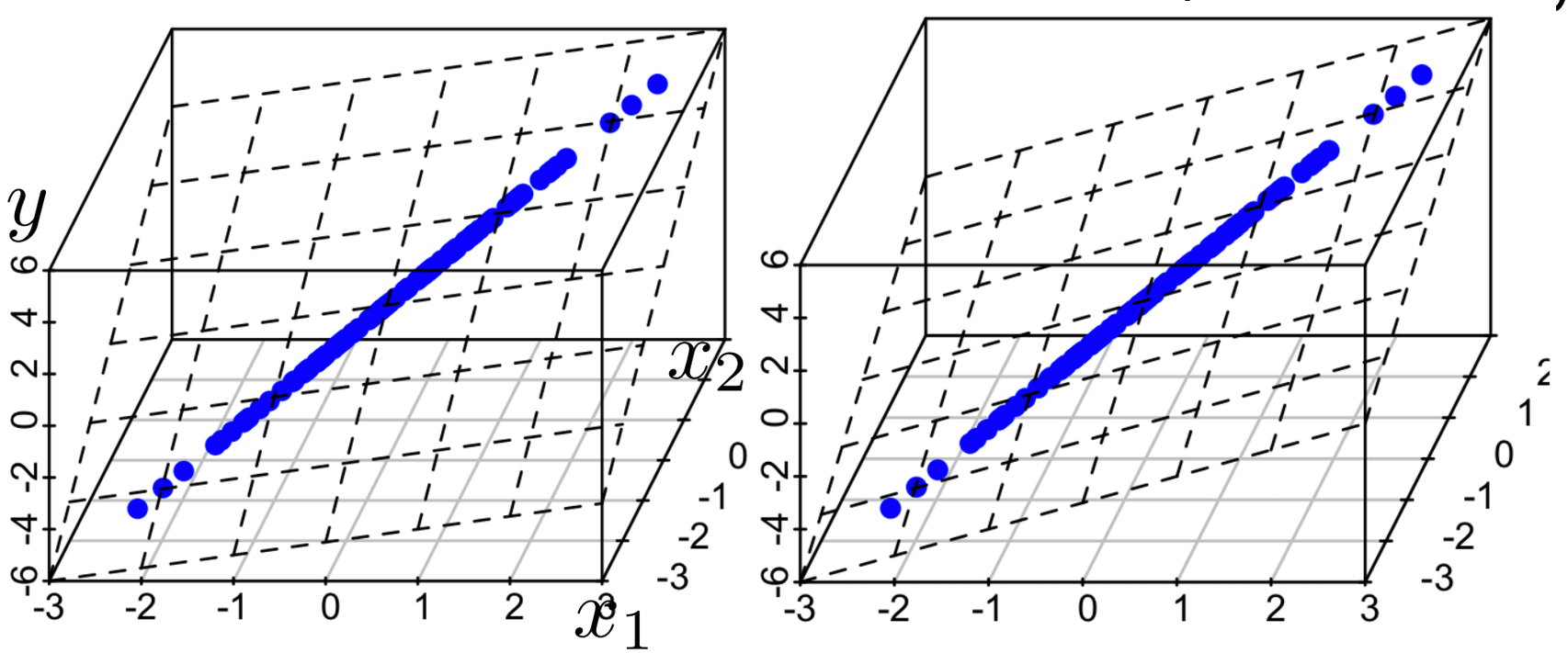
What can go wrong in practice?

- Sometimes there isn't a unique best hyperplane



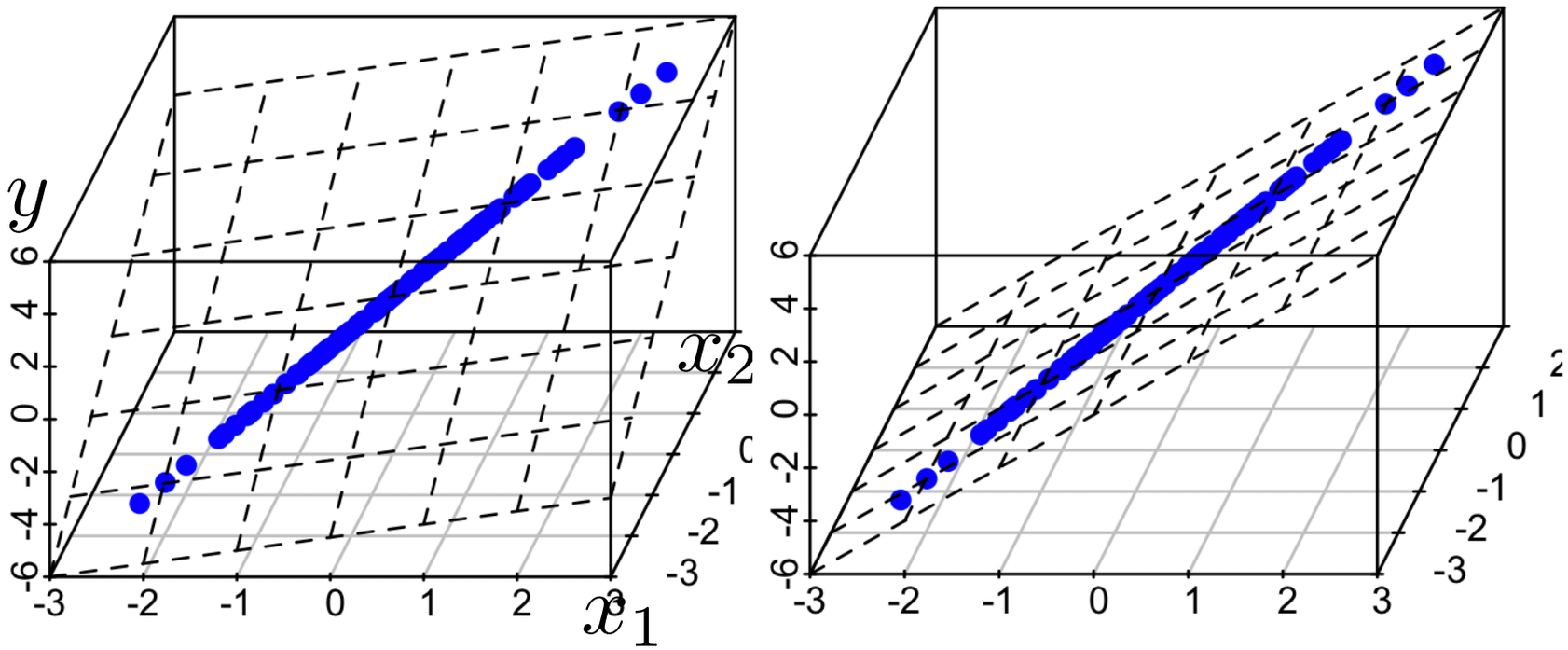
What can go wrong in practice?

- Sometimes there isn't a unique best hyperplane



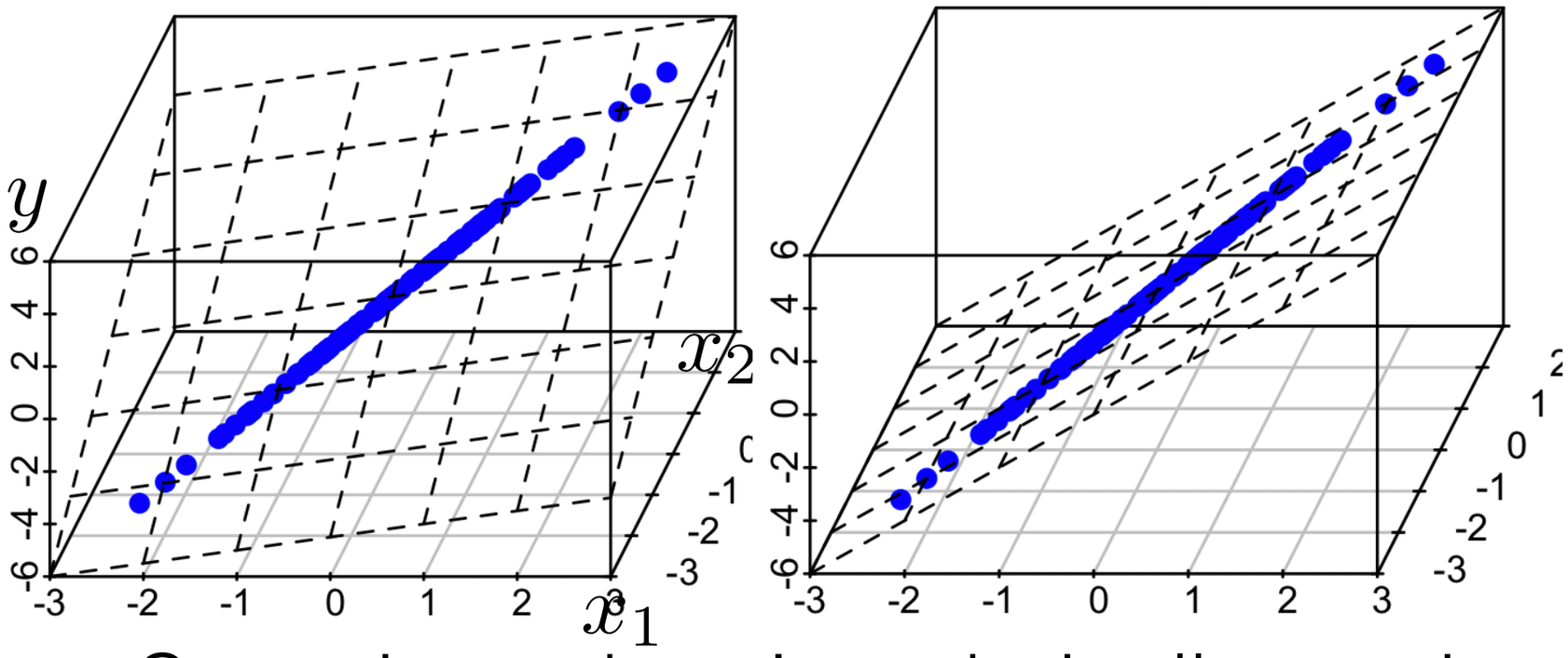
What can go wrong in practice?

- Sometimes there isn't a unique best hyperplane



What can go wrong in practice?

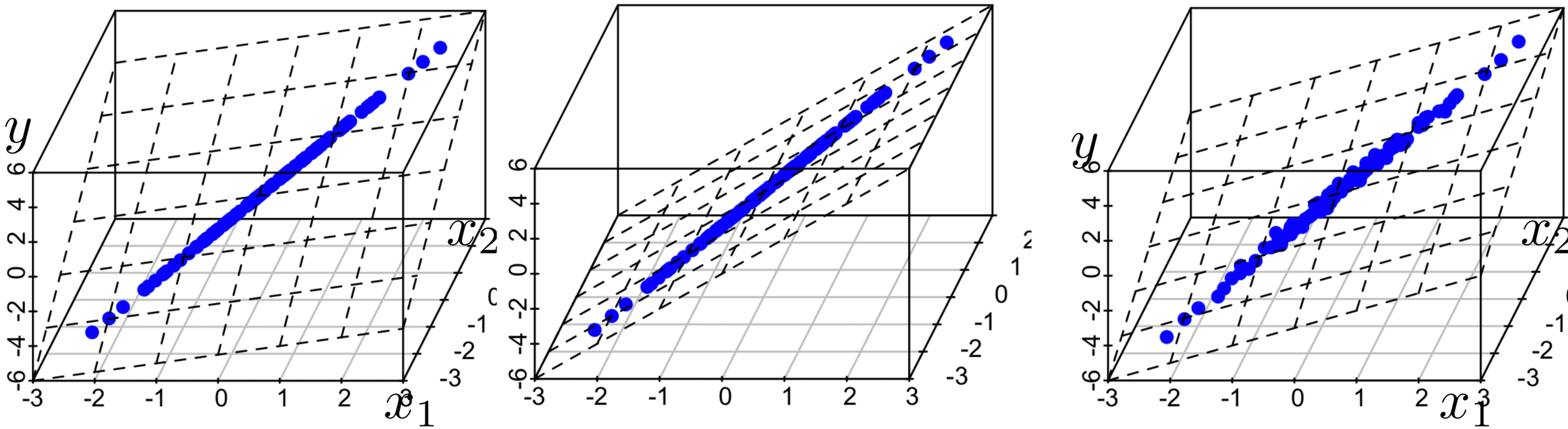
- Sometimes there isn't a unique best hyperplane



- Sometimes there's technically a unique best hyperplane, but just because of noise

What can go wrong in practice?

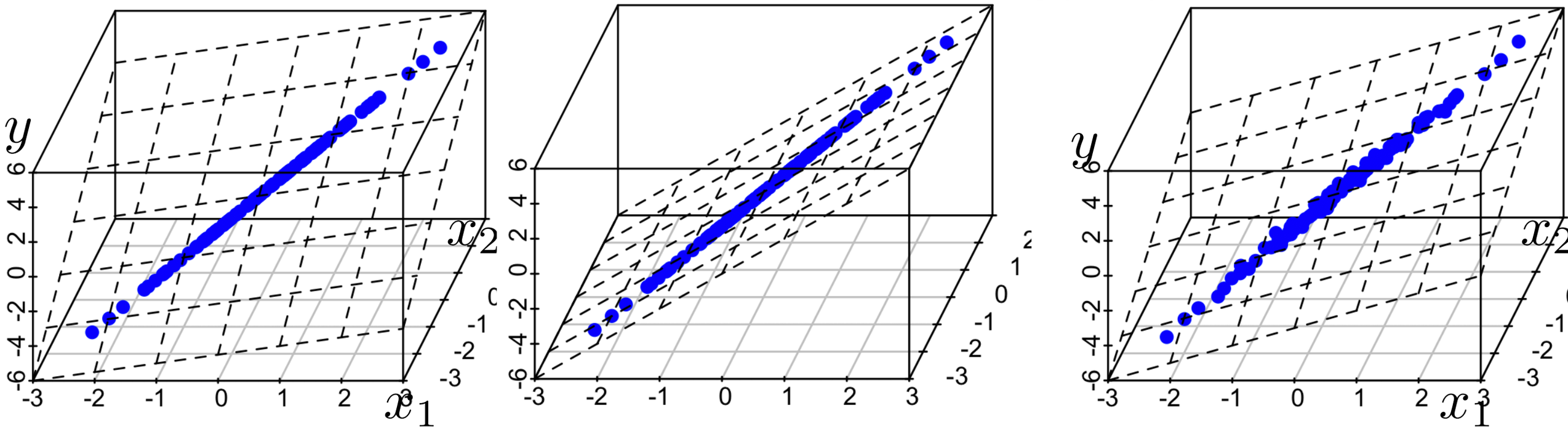
- Sometimes there isn't a unique best hyperplane



- Sometimes there's technically a unique best hyperplane, but just because of noise

What can go wrong in practice?

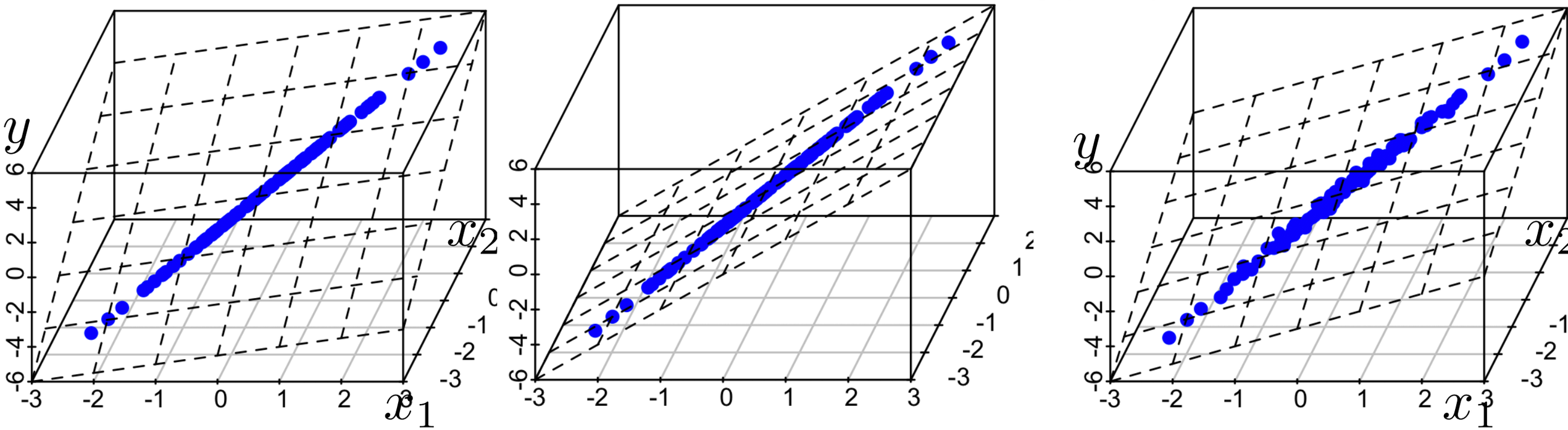
- Sometimes there isn't a unique best hyperplane



- Sometimes there's technically a unique best hyperplane, but just because of noise
- Feature encodings (cf. one-hot)

What can go wrong in practice?

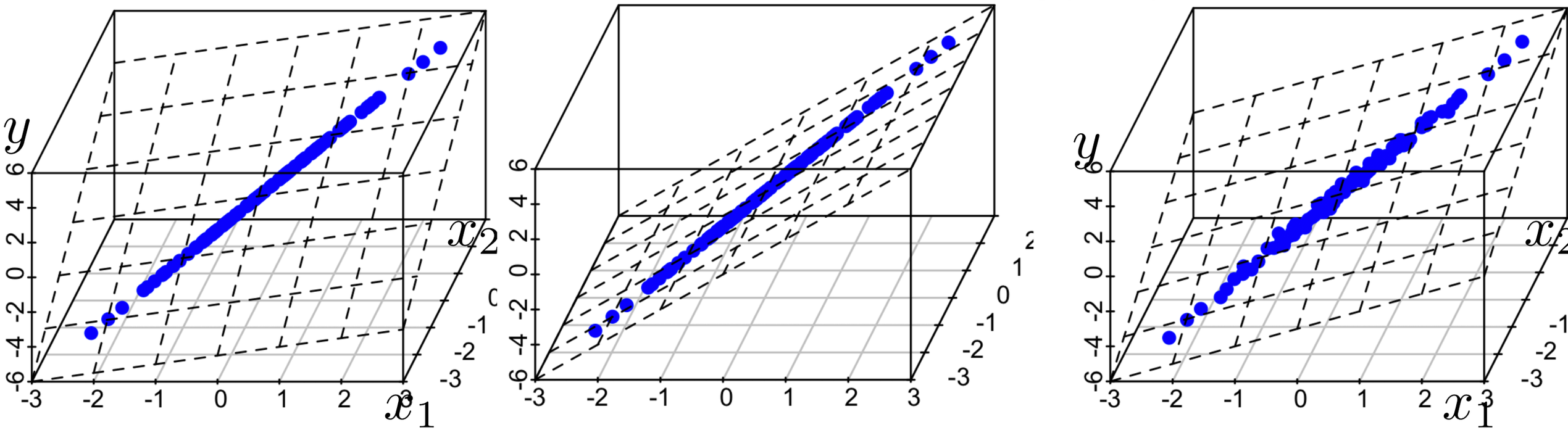
- Sometimes there isn't a unique best hyperplane



- Sometimes there's technically a unique best hyperplane, but just because of noise
- Feature encodings (cf. one-hot); real-life features often correlated

What can go wrong in practice?

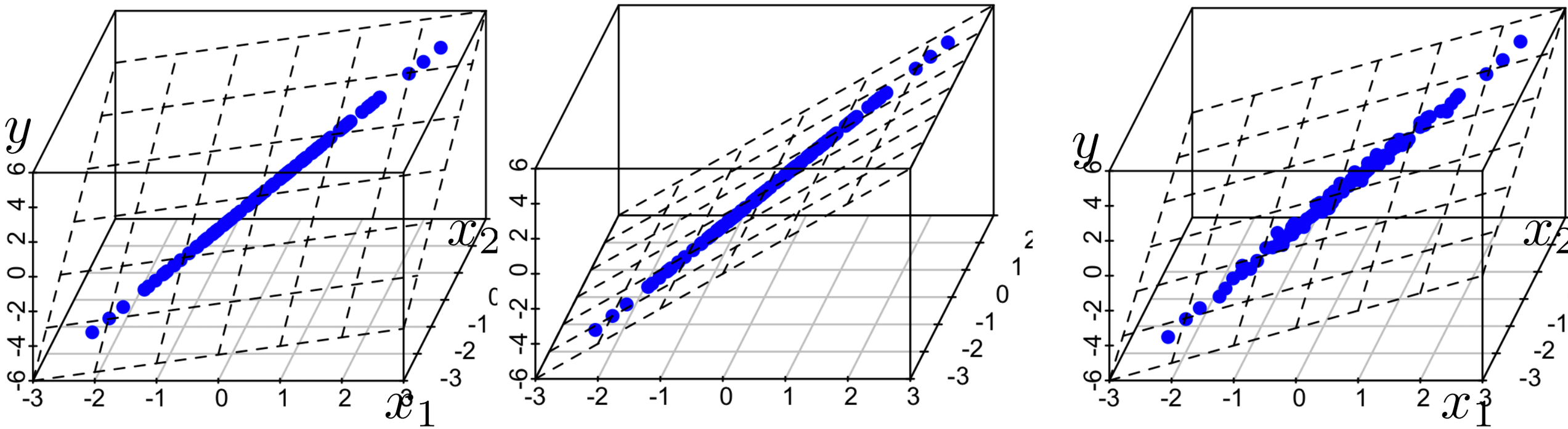
- Sometimes there isn't a unique best hyperplane



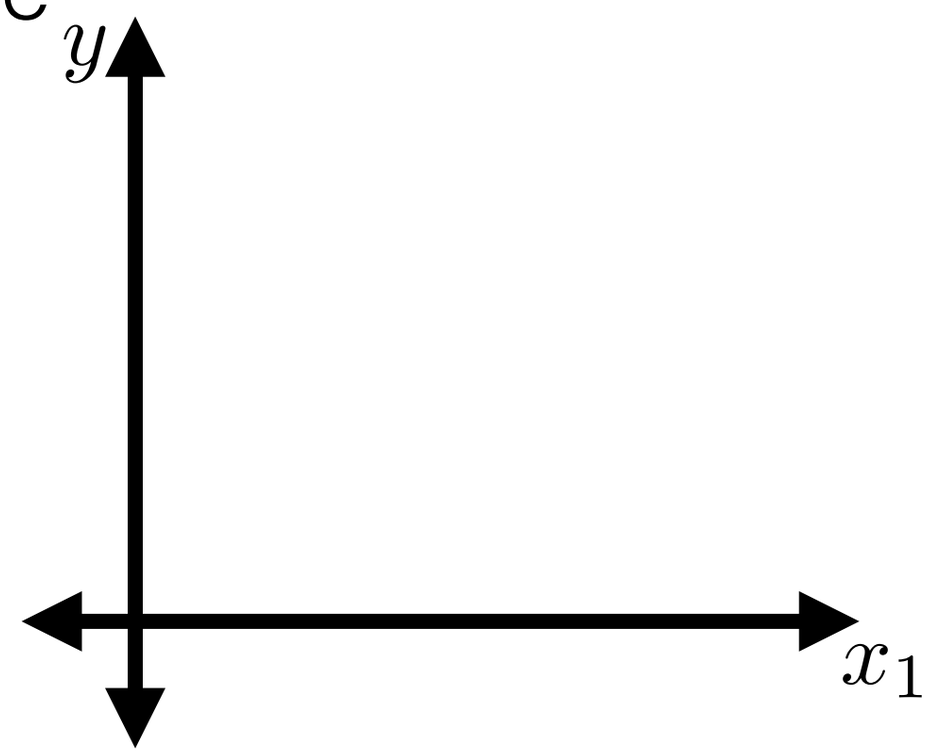
- Sometimes there's technically a unique best hyperplane, but just because of noise
- Feature encodings (cf. one-hot);
  real-life features often correlated;
  many feature dimensions

What can go wrong in practice?

- Sometimes there isn't a unique best hyperplane

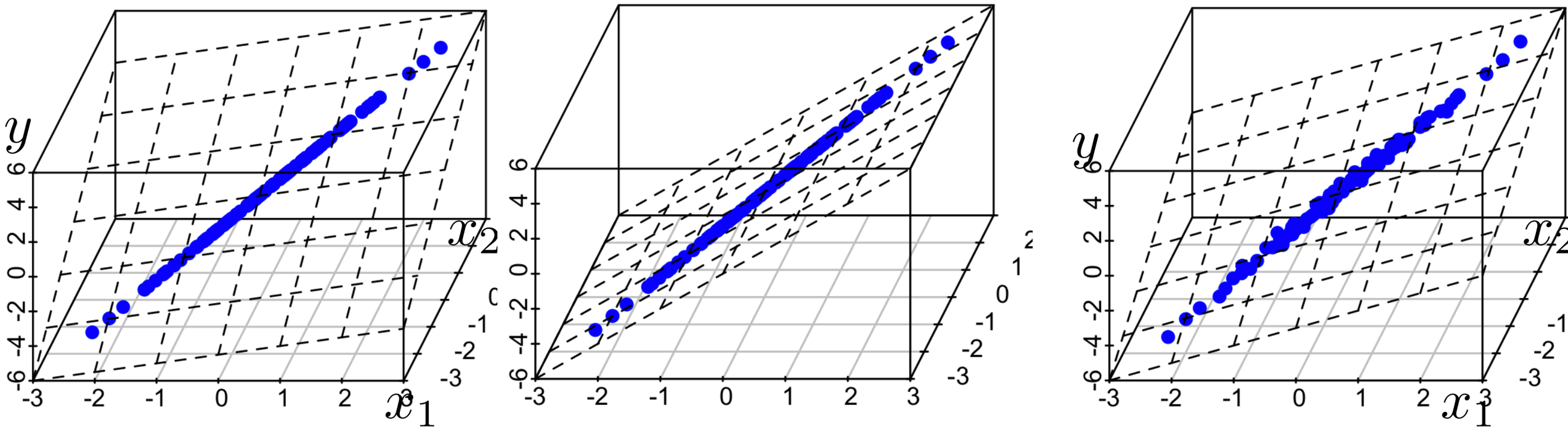


- Sometimes there's technically a unique best hyperplane, but just because of noise
- Feature encodings (cf. one-hot); real-life features often correlated; many feature dimensions

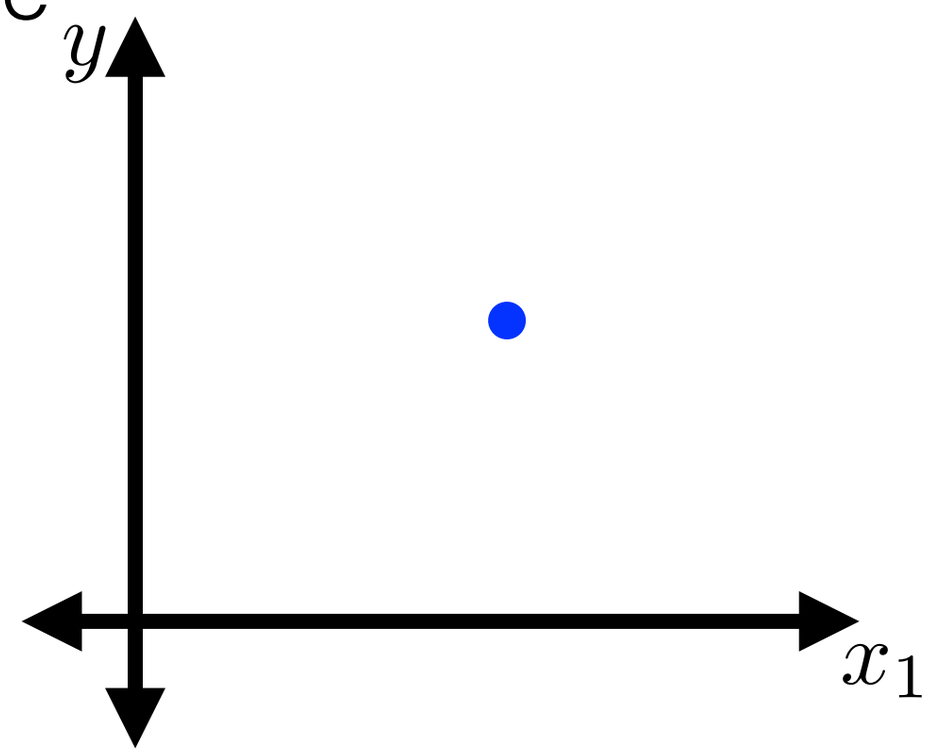


What can go wrong in practice?

- Sometimes there isn't a unique best hyperplane

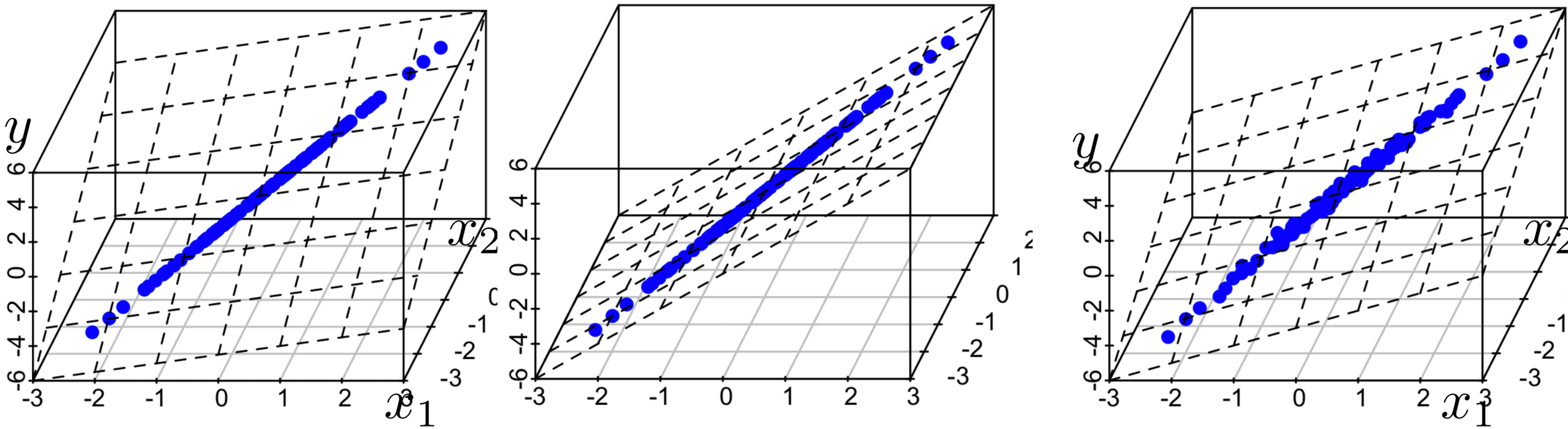


- Sometimes there's technically a unique best hyperplane, but just because of noise
- Feature encodings (cf. one-hot); real-life features often correlated; many feature dimensions

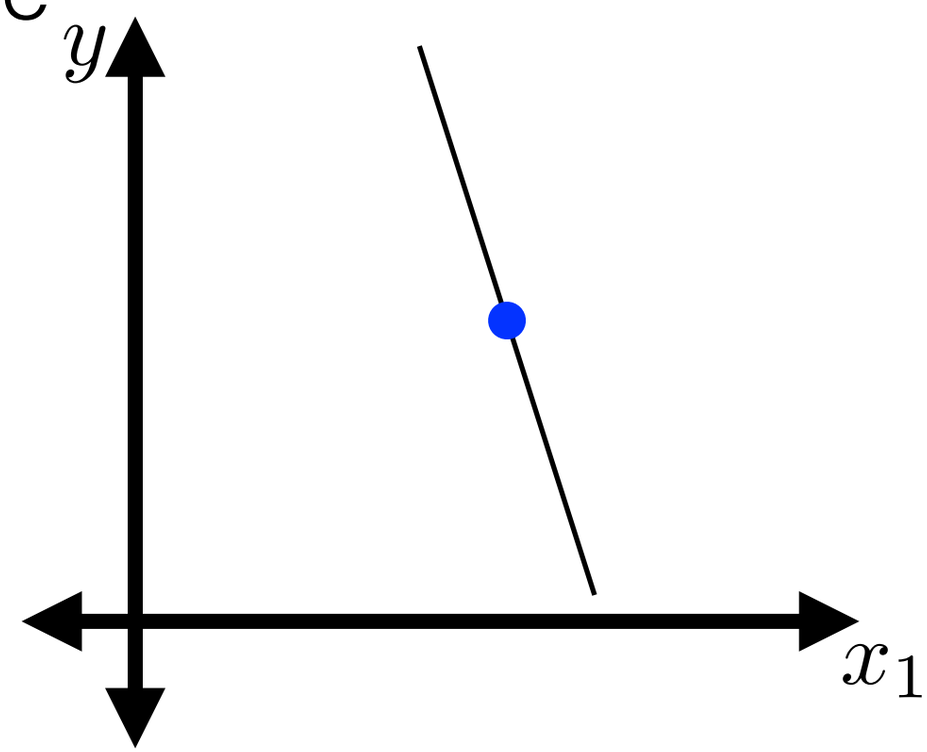


What can go wrong in practice?

- Sometimes there isn't a unique best hyperplane

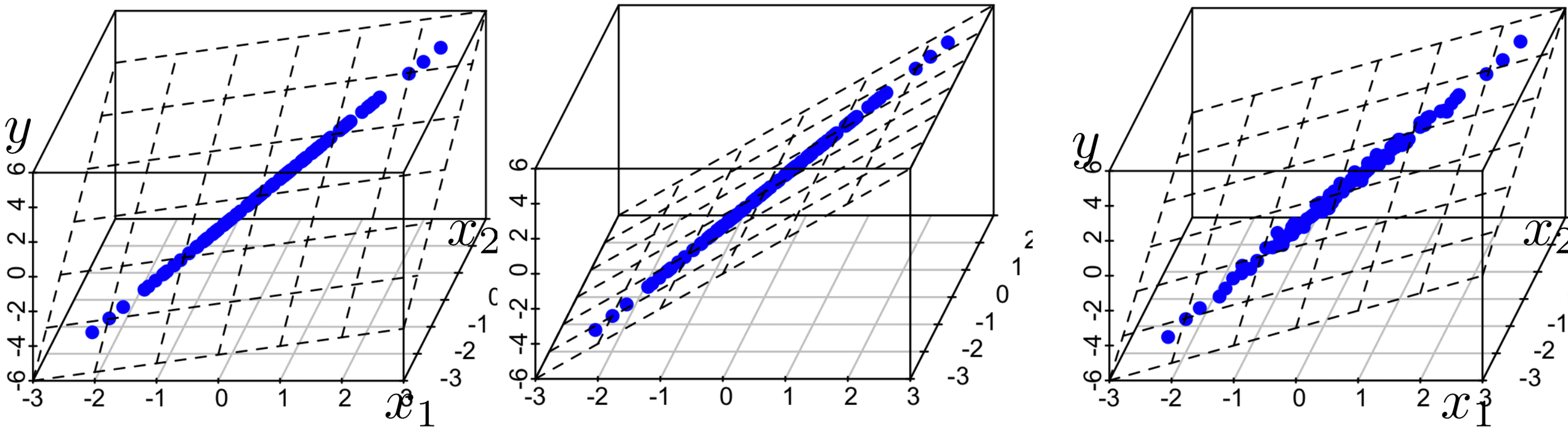


- Sometimes there's technically a unique best hyperplane, but just because of noise
- Feature encodings (cf. one-hot); real-life features often correlated; many feature dimensions

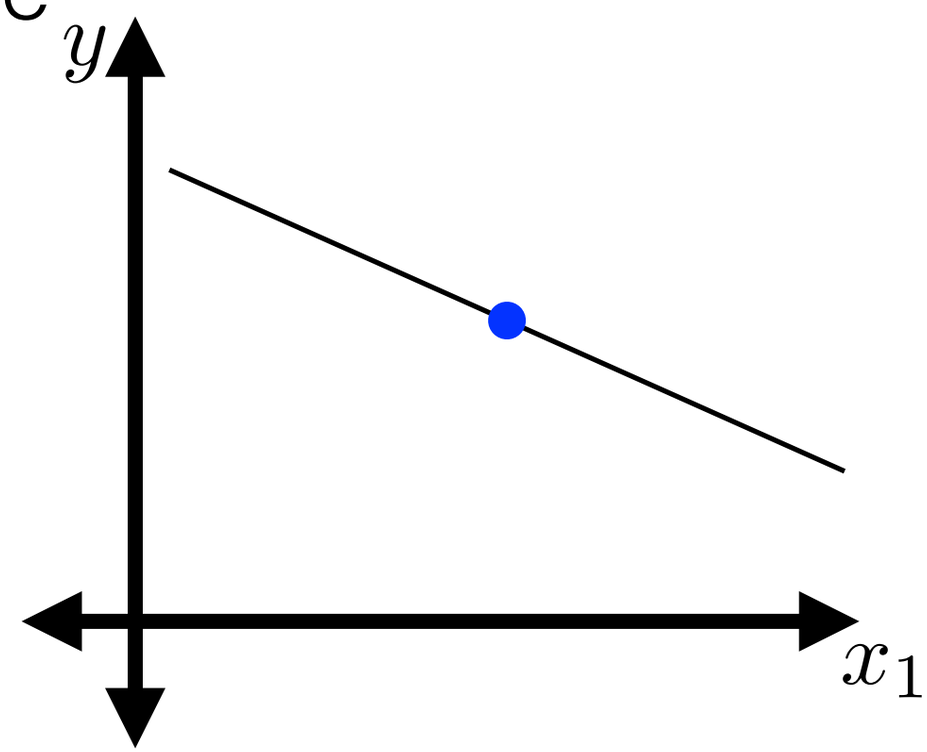


What can go wrong in practice?

- Sometimes there isn't a unique best hyperplane

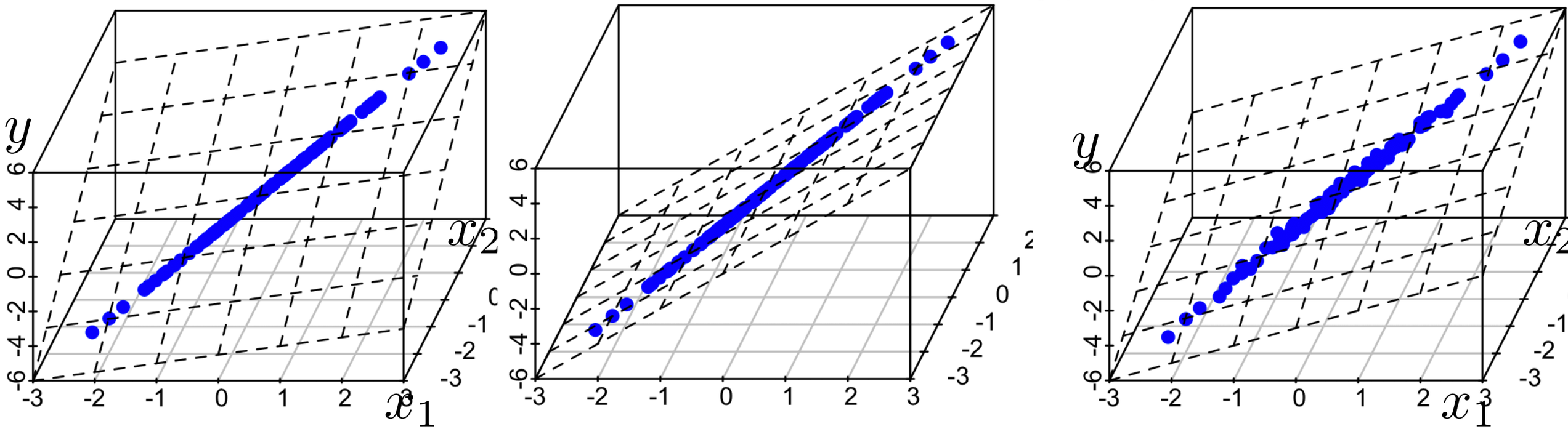


- Sometimes there's technically a unique best hyperplane, but just because of noise
- Feature encodings (cf. one-hot); real-life features often correlated; many feature dimensions

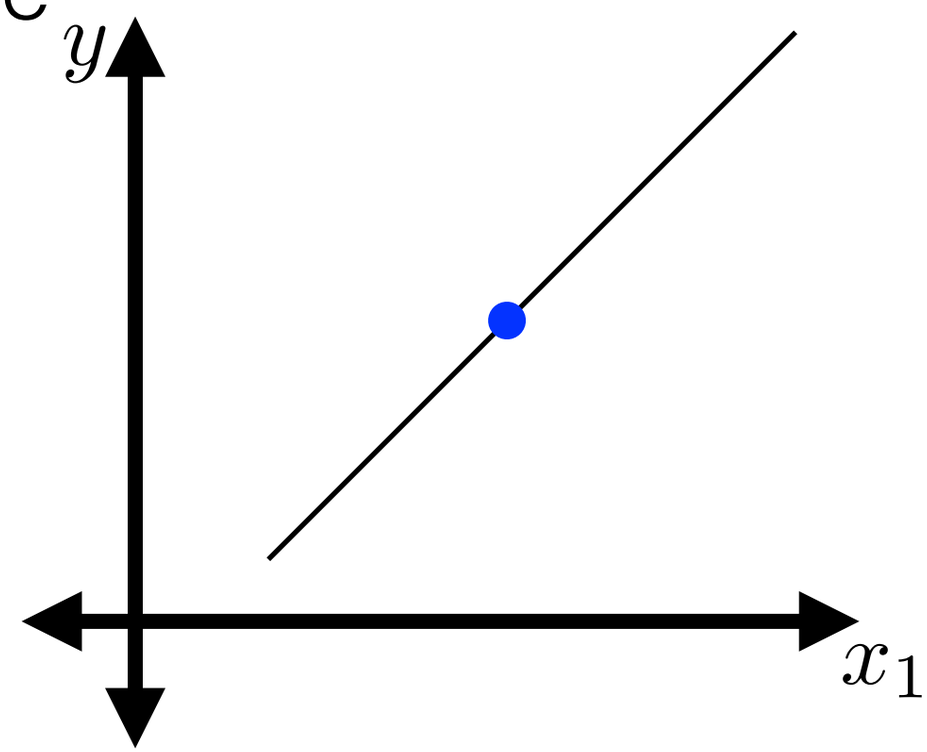


What can go wrong in practice?

- Sometimes there isn't a unique best hyperplane

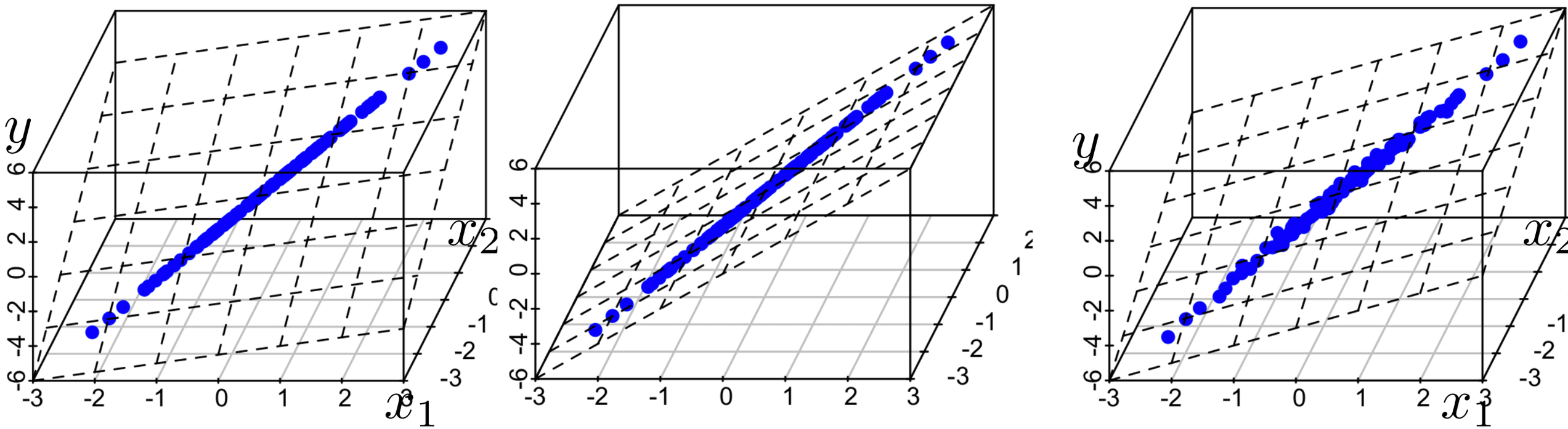


- Sometimes there's technically a unique best hyperplane, but just because of noise
- Feature encodings (cf. one-hot); real-life features often correlated; many feature dimensions

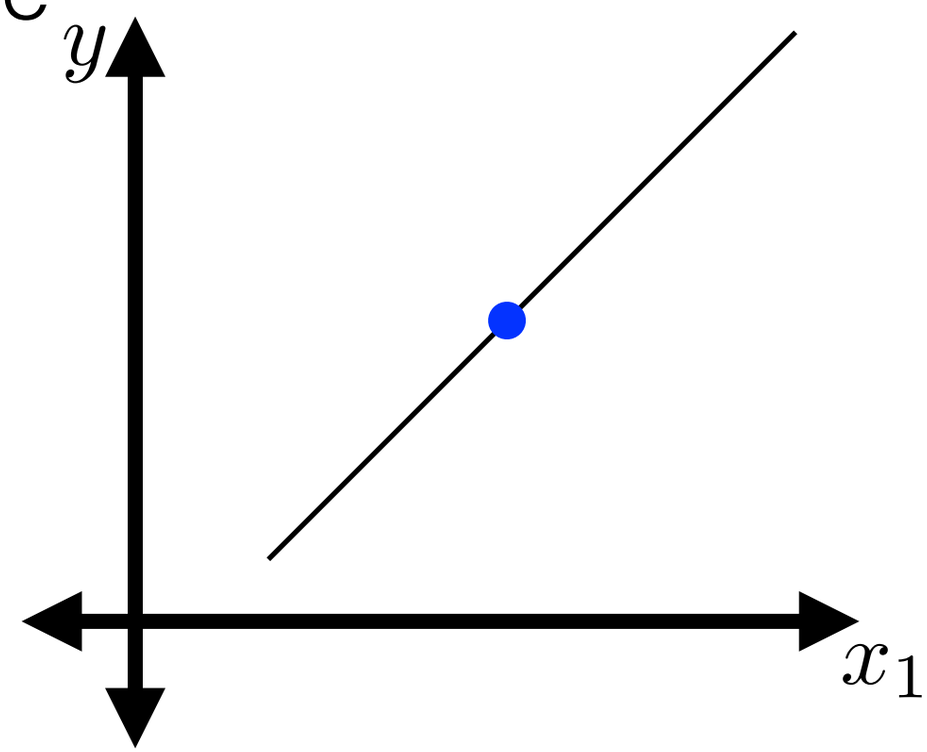


What can go wrong in practice?

- Sometimes there isn't a unique best hyperplane

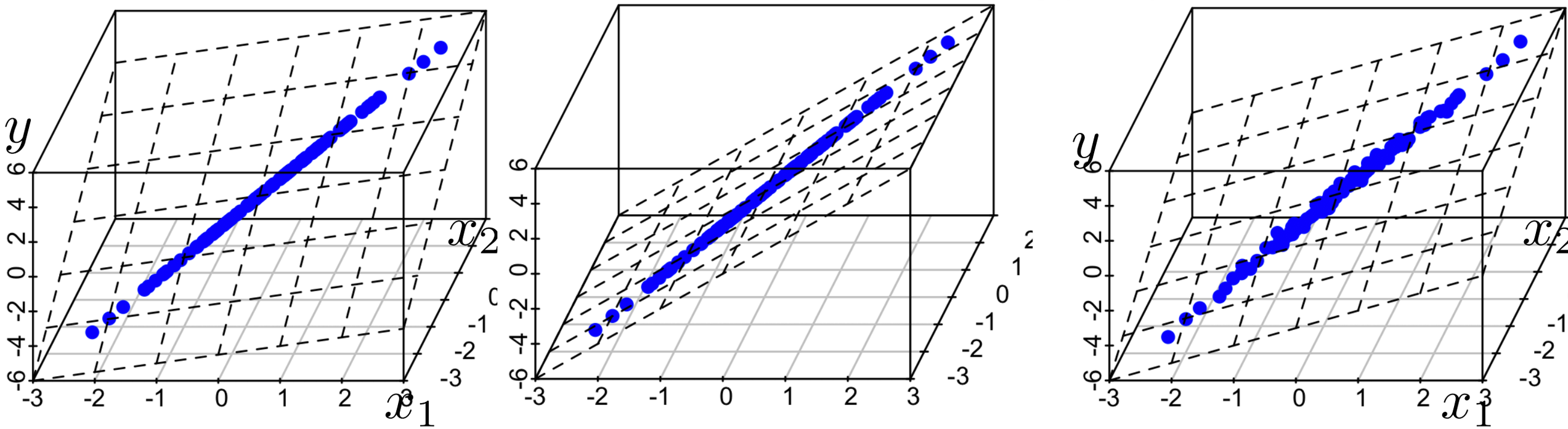


- Sometimes there's technically a unique best hyperplane, but just because of noise
- Feature encodings (cf. one-hot); real-life features often correlated; many feature dimensions
- How to choose among planes?

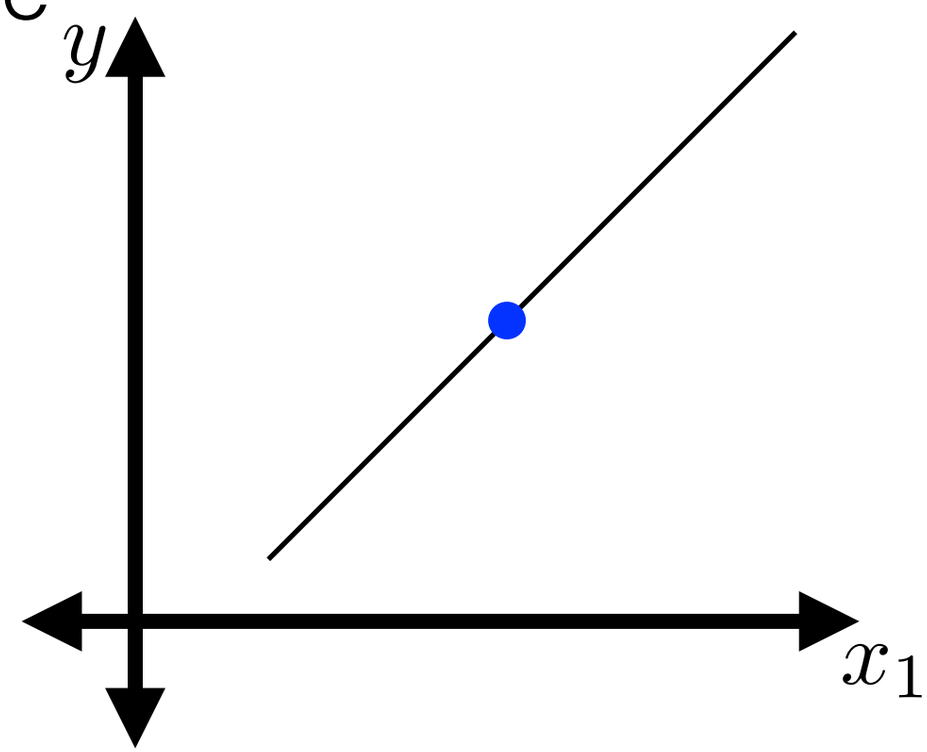


What can go wrong in practice?

- Sometimes there isn't a unique best hyperplane



- Sometimes there's technically a unique best hyperplane, but just because of noise
- Feature encodings (cf. one-hot); real-life features often correlated; many feature dimensions
- How to choose among planes? Preference for θ components being near zero



Regularizing linear regression

Regularizing linear regression

- Linear regression with square penalty: ridge regression

Regularizing linear regression

- Linear regression with square penalty: ridge regression

$$\frac{1}{n}\sum_{i=1}^{n}(\theta^\top x^{(i)} + \theta_0 - y^{(i)})^2$$

Regularizing linear regression

- Linear regression with square penalty: ridge regression

$$J_{\text{ridge}}(\theta, \theta_0) = \frac{1}{n} \sum_{i=1}^{n} (\theta^\top x^{(i)} + \theta_0 - y^{(i)})^2$$

Regularizing linear regression

- Linear regression with square penalty: ridge regression

$$J_{\text{ridge}}(\theta, \theta_0) = \frac{1}{n} \sum_{i=1}^{n} (\theta^\top x^{(i)} + \theta_0 - y^{(i)})^2 + \lambda \|\theta\|^2$$

Regularizing linear regression

- Linear regression with square penalty: ridge regression

$$J_{\text{ridge}}(\theta, \theta_0) = \frac{1}{n} \sum_{i=1}^{n} (\theta^\top x^{(i)} + \theta_0 - y^{(i)})^2 + \lambda \|\theta\|^2 \quad (\lambda > 0)$$

Regularizing linear regression

- Linear regression with square penalty: ridge regression

$$J_{\text{ridge}}(\theta, \theta_0) = \frac{1}{n} \sum_{i=1}^{n} (\theta^\top x^{(i)} + \theta_0 - y^{(i)})^2 + \lambda \|\theta\|^2 \quad (\lambda > 0)$$

What happens if $\lambda < 0$?

Regularizing linear regression

- Linear regression with square penalty: ridge regression

$$J_{\text{ridge}}(\theta, \theta_0) = \frac{1}{n} \sum_{i=1}^{n} (\theta^\top x^{(i)} + \theta_0 - y^{(i)})^2 + \lambda \|\theta\|^2 \quad (\lambda > 0)$$

- Special case: ridge regression with no offset

What happens if $\lambda < 0$?

Regularizing linear regression

- Linear regression with square penalty: ridge regression

$$J_{\text{ridge}}(\theta, \theta_0) = \frac{1}{n} \sum_{i=1}^{n} (\theta^\top x^{(i)} + \theta_0 - y^{(i)})^2 + \lambda \|\theta\|^2 \quad (\lambda > 0)$$

- Special case: ridge regression with no offset

$$J_{\text{ridge}}(\theta) = \frac{1}{n} (\tilde{X}\theta - \tilde{Y})^\top (\tilde{X}\theta - \tilde{Y}) + \lambda \|\theta\|^2$$

What happens if $\lambda < 0$?

Regularizing linear regression

- Linear regression with square penalty: ridge regression

$$J_{\text{ridge}}(\theta, \theta_0) = \frac{1}{n}\sum_{i=1}^{n}(\theta^\top x^{(i)} + \theta_0 - y^{(i)})^2 + \lambda\|\theta\|^2 \quad (\lambda > 0)$$

- Special case: ridge regression with no offset

$$J_{\text{ridge}}(\theta) = \frac{1}{n}(\tilde{X}\theta - \tilde{Y})^\top(\tilde{X}\theta - \tilde{Y}) + \lambda\|\theta\|^2$$

$$= \frac{1}{n}\|\tilde{X}\theta - \tilde{Y}\|^2 + \lambda\|\theta\|^2$$

What happens if $\lambda < 0$?

Regularizing linear regression

- Linear regression with square penalty: ridge regression

$$J_{\text{ridge}}(\theta, \theta_0) = \frac{1}{n}\sum_{i=1}^{n}(\theta^\top x^{(i)} + \theta_0 - y^{(i)})^2 + \lambda\|\theta\|^2 \quad (\lambda > 0)$$

- Special case: ridge regression with no offset

$$J_{\text{ridge}}(\theta) = \frac{1}{n}(\tilde{X}\theta - \tilde{Y})^\top(\tilde{X}\theta - \tilde{Y}) + \lambda\|\theta\|^2$$

$$= \frac{1}{n}\|\tilde{X}\theta - \tilde{Y}\|^2 + \lambda\|\theta\|^2$$

What happens if $\lambda < 0$?

- Min at: $\nabla_\theta J_{\text{ridge}}(\theta) = 0$

Regularizing linear regression

- Linear regression with square penalty: ridge regression

$$J_{\text{ridge}}(\theta, \theta_0) = \frac{1}{n}\sum_{i=1}^{n}(\theta^\top x^{(i)} + \theta_0 - y^{(i)})^2 + \lambda\|\theta\|^2 \quad (\lambda > 0)$$

- Special case: ridge regression with no offset

$$J_{\text{ridge}}(\theta) = \frac{1}{n}(\tilde{X}\theta - \tilde{Y})^\top(\tilde{X}\theta - \tilde{Y}) + \lambda\|\theta\|^2$$

$$= \frac{1}{n}\|\tilde{X}\theta - \tilde{Y}\|^2 + \lambda\|\theta\|^2$$

What happens if $\lambda < 0$?

- Min at: $\nabla_\theta J_{\text{ridge}}(\theta) = 0 \;\Rightarrow \theta = (\tilde{X}^\top\tilde{X} + n\lambda I)^{-1}\tilde{X}^\top\tilde{Y}$

Regularizing linear regression

- Linear regression with square penalty: ridge regression

$$J_{\text{ridge}}(\theta, \theta_0) = \frac{1}{n}\sum_{i=1}^{n}(\theta^\top x^{(i)} + \theta_0 - y^{(i)})^2 + \lambda\|\theta\|^2 \quad (\lambda > 0)$$

- Special case: ridge regression with no offset

$$J_{\text{ridge}}(\theta) = \frac{1}{n}(\tilde{X}\theta - \tilde{Y})^\top(\tilde{X}\theta - \tilde{Y}) + \lambda\|\theta\|^2$$

$$= \frac{1}{n}\|\tilde{X}\theta - \tilde{Y}\|^2 + \lambda\|\theta\|^2$$

What happens if $\lambda < 0$?

- Min at: $\nabla_\theta J_{\text{ridge}}(\theta) = 0 \;\Rightarrow \theta = (\tilde{X}^\top\tilde{X} + n\lambda I)^{-1}\tilde{X}^\top\tilde{Y}$

Regularizing linear regression

- Linear regression with square penalty: ridge regression

$$J_{\text{ridge}}(\theta, \theta_0) = \frac{1}{n}\sum_{i=1}^{n}(\theta^\top x^{(i)} + \theta_0 - y^{(i)})^2 + \lambda\|\theta\|^2 \quad (\lambda > 0)$$

- Special case: ridge regression with no offset

$$J_{\text{ridge}}(\theta) = \frac{1}{n}(\tilde{X}\theta - \tilde{Y})^\top(\tilde{X}\theta - \tilde{Y}) + \lambda\|\theta\|^2$$

$$= \frac{1}{n}\|\tilde{X}\theta - \tilde{Y}\|^2 + \lambda\|\theta\|^2$$

What happens if $\lambda < 0$?

- Min at: $\nabla_\theta J_{\text{ridge}}(\theta) = 0 \;\Rightarrow \theta = (\tilde{X}^\top \tilde{X} + n\lambda I)^{-1}\tilde{X}^\top \tilde{Y}$

nxd

Regularizing linear regression

- Linear regression with square penalty: ridge regression

$$J_{\text{ridge}}(\theta, \theta_0) = \frac{1}{n}\sum_{i=1}^{n}(\theta^\top x^{(i)} + \theta_0 - y^{(i)})^2 + \lambda\|\theta\|^2 \quad (\lambda > 0)$$

- Special case: ridge regression with no offset

$$J_{\text{ridge}}(\theta) = \frac{1}{n}(\tilde{X}\theta - \tilde{Y})^\top(\tilde{X}\theta - \tilde{Y}) + \lambda\|\theta\|^2$$

$$= \frac{1}{n}\|\tilde{X}\theta - \tilde{Y}\|^2 + \lambda\|\theta\|^2$$

What happens if $\lambda < 0$?

- Min at: $\nabla_\theta J_{\text{ridge}}(\theta) = 0 \;\Rightarrow \theta = (\tilde{X}^\top\tilde{X} + n\lambda I)^{-1}\tilde{X}^\top\tilde{Y}$

dxn,nxd

Regularizing linear regression

- Linear regression with square penalty: ridge regression

$$J_{\text{ridge}}(\theta, \theta_0) = \frac{1}{n}\sum_{i=1}^{n}(\theta^\top x^{(i)} + \theta_0 - y^{(i)})^2 + \lambda\|\theta\|^2 \quad (\lambda > 0)$$

- Special case: ridge regression with no offset

$$J_{\text{ridge}}(\theta) = \frac{1}{n}(\tilde{X}\theta - \tilde{Y})^\top(\tilde{X}\theta - \tilde{Y}) + \lambda\|\theta\|^2$$

$$= \frac{1}{n}\|\tilde{X}\theta - \tilde{Y}\|^2 + \lambda\|\theta\|^2$$

What happens if $\lambda < 0$?

- Min at: $\nabla_\theta J_{\text{ridge}}(\theta) = 0 \;\Rightarrow \theta = (\tilde{X}^\top\tilde{X} + n\lambda I)^{-1}\tilde{X}^\top\tilde{Y}$

dxn,nxd dxd

Regularizing linear regression

- Linear regression with square penalty: ridge regression

$$J_{\text{ridge}}(\theta, \theta_0) = \frac{1}{n}\sum_{i=1}^{n}(\theta^\top x^{(i)} + \theta_0 - y^{(i)})^2 + \lambda\|\theta\|^2 \quad (\lambda > 0)$$

- Special case: ridge regression with no offset

$$J_{\text{ridge}}(\theta) = \frac{1}{n}(\tilde{X}\theta - \tilde{Y})^\top(\tilde{X}\theta - \tilde{Y}) + \lambda\|\theta\|^2$$

$$= \frac{1}{n}\|\tilde{X}\theta - \tilde{Y}\|^2 + \lambda\|\theta\|^2$$

What happens if $\lambda < 0$?

- Min at: $\nabla_\theta J_{\text{ridge}}(\theta) = 0 \;\Rightarrow \theta = (\tilde{X}^\top\tilde{X} + n\lambda I)^{-1}\tilde{X}^\top\tilde{Y}$

dxn,nxd dxd

Regularizing linear regression

- Linear regression with square penalty: ridge regression

$$J_{\text{ridge}}(\theta, \theta_0) = \frac{1}{n}\sum_{i=1}^{n}(\theta^\top x^{(i)} + \theta_0 - y^{(i)})^2 + \lambda\|\theta\|^2 \quad (\lambda > 0)$$

- Special case: ridge regression with no offset

$$J_{\text{ridge}}(\theta) = \frac{1}{n}(\tilde{X}\theta - \tilde{Y})^\top(\tilde{X}\theta - \tilde{Y}) + \lambda\|\theta\|^2$$

$$= \frac{1}{n}\|\tilde{X}\theta - \tilde{Y}\|^2 + \lambda\|\theta\|^2$$

What happens if $\lambda < 0$?

- Min at: $\nabla_\theta J_{\text{ridge}}(\theta) = 0 \;\Rightarrow \theta = (\tilde{X}^\top\tilde{X} + n\lambda I)^{-1}\tilde{X}^\top\tilde{Y}$

dxn,nxd dxd

  - Matrix of second derivatives: $\tilde{X}^\top\tilde{X} + n\lambda I$ (always "curves up" & invertible when $\lambda > 0$)

Regularizing linear regression

- Linear regression with square penalty: ridge regression

$$J_{\text{ridge}}(\theta, \theta_0) = \frac{1}{n}\sum_{i=1}^{n}(\theta^\top x^{(i)} + \theta_0 - y^{(i)})^2 + \lambda\|\theta\|^2 \quad (\lambda > 0)$$

- Special case: ridge regression with no offset

$$J_{\text{ridge}}(\theta) = \frac{1}{n}(\tilde{X}\theta - \tilde{Y})^\top(\tilde{X}\theta - \tilde{Y}) + \lambda\|\theta\|^2$$

$$= \frac{1}{n}\|\tilde{X}\theta - \tilde{Y}\|^2 + \lambda\|\theta\|^2$$

What happens if $\lambda < 0$?

  - Min at: $\nabla_\theta J_{\text{ridge}}(\theta) = 0 \;\Rightarrow \theta = (\tilde{X}^\top\tilde{X} + n\lambda I)^{-1}\tilde{X}^\top\tilde{Y}$

dxn,nxd dxd

    - Matrix of second derivatives: $\tilde{X}^\top\tilde{X} + n\lambda I$ (always "curves up" & invertible when $\lambda > 0$)

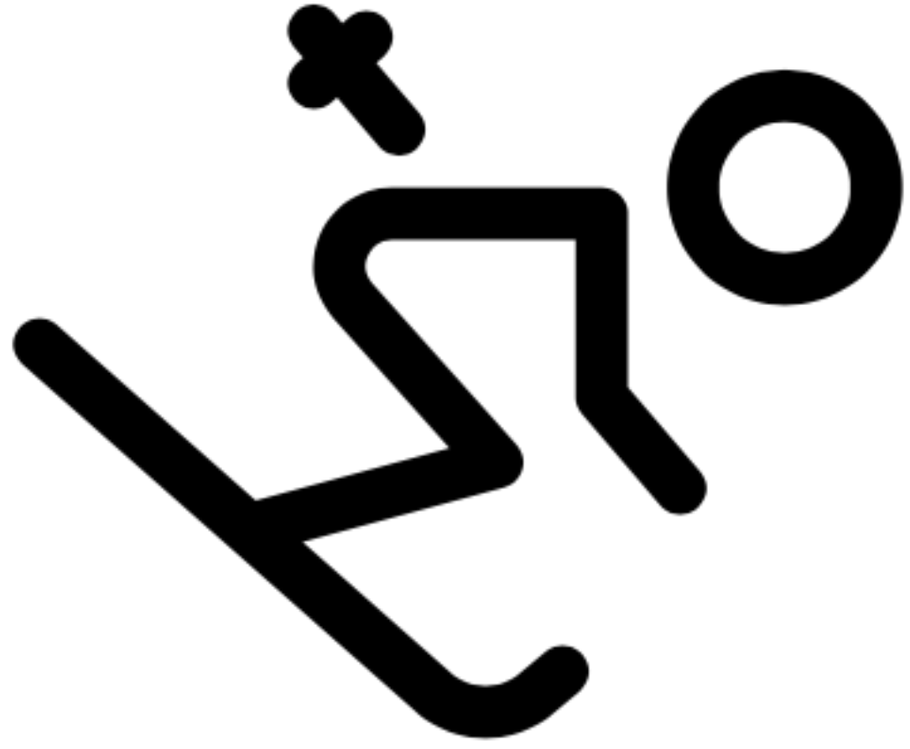
- Can also solve for minimizing parameters in case with offset; just a bit more math

- Linear regression with square penalty: ridge regression

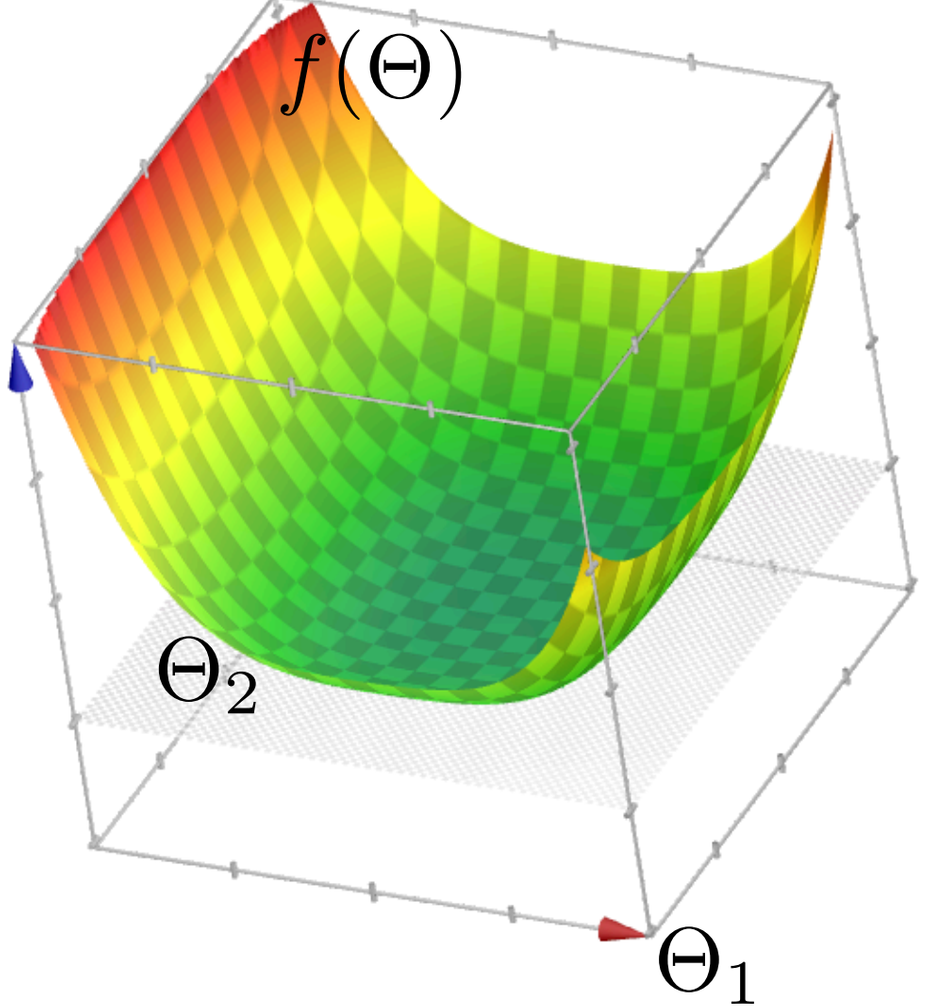
$$J_{\text{ridge}}(\theta, \theta_0) = \frac{1}{n} \sum_{i=1}^{n} (\theta^\top x^{(i)} + \theta_0 - y^{(i)})^2 + \lambda \|\theta\|^2 \quad (\lambda > 0)$$

Gradient descent

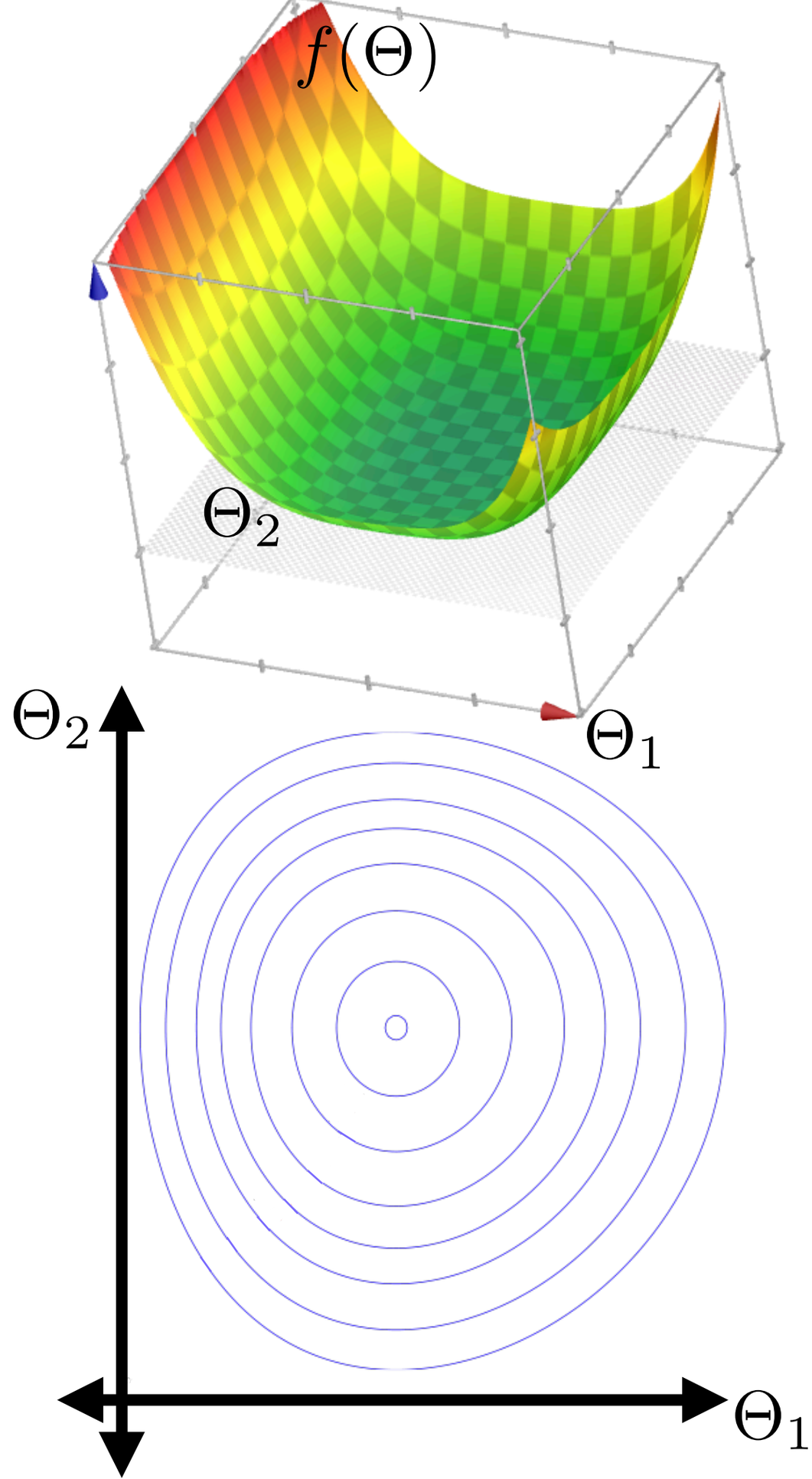
a simple and powerful optimization algorithm



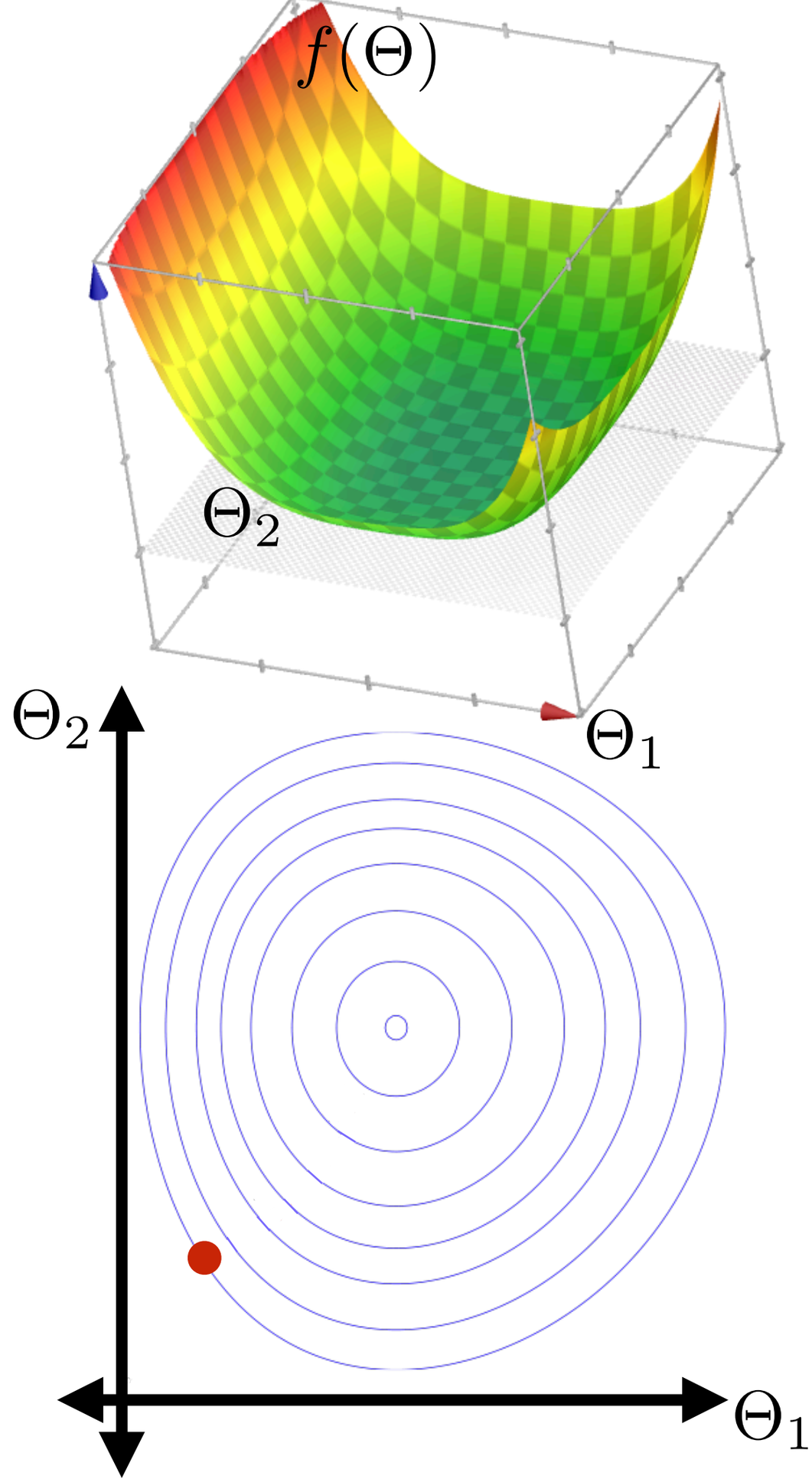
Gradient descent



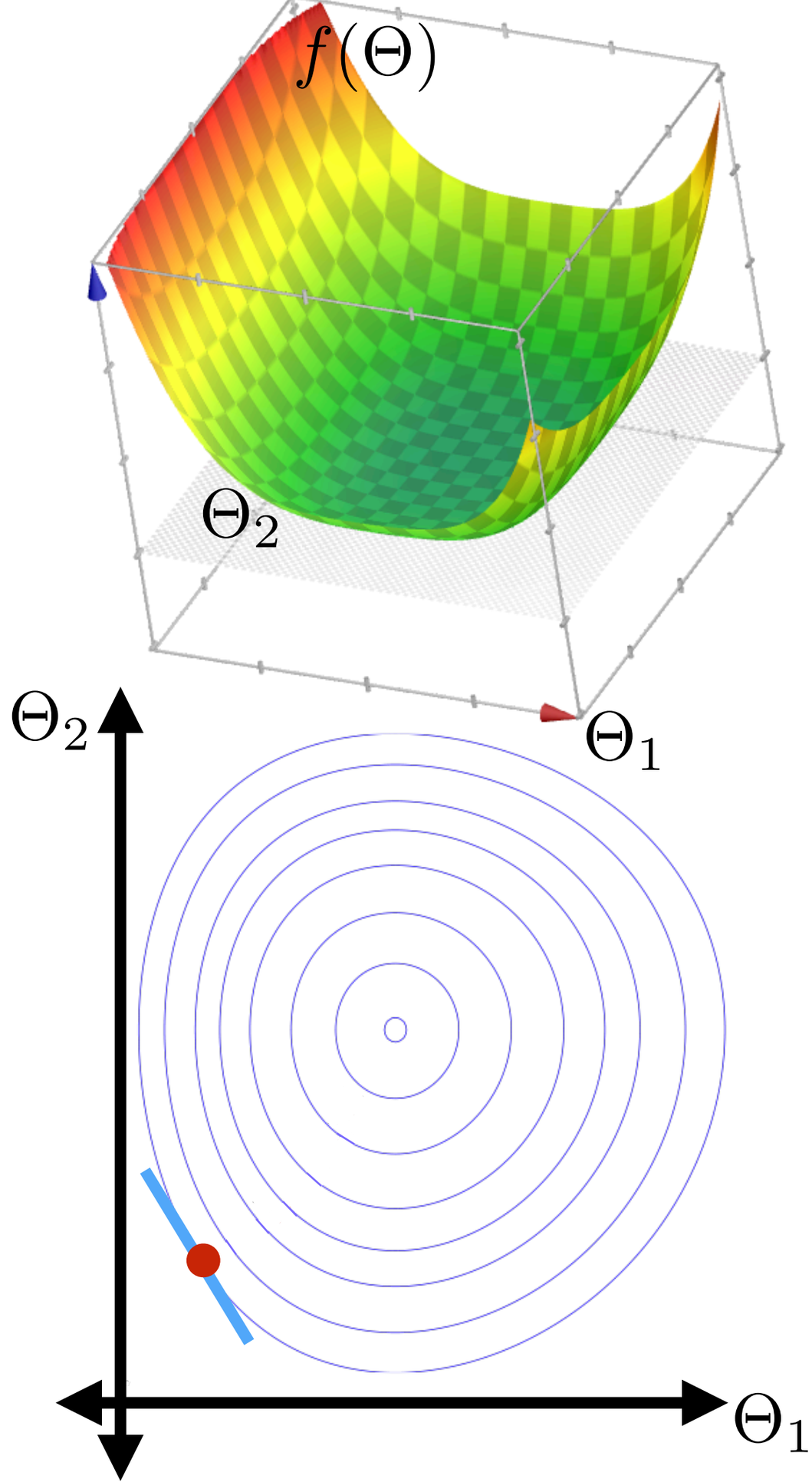
Gradient descent



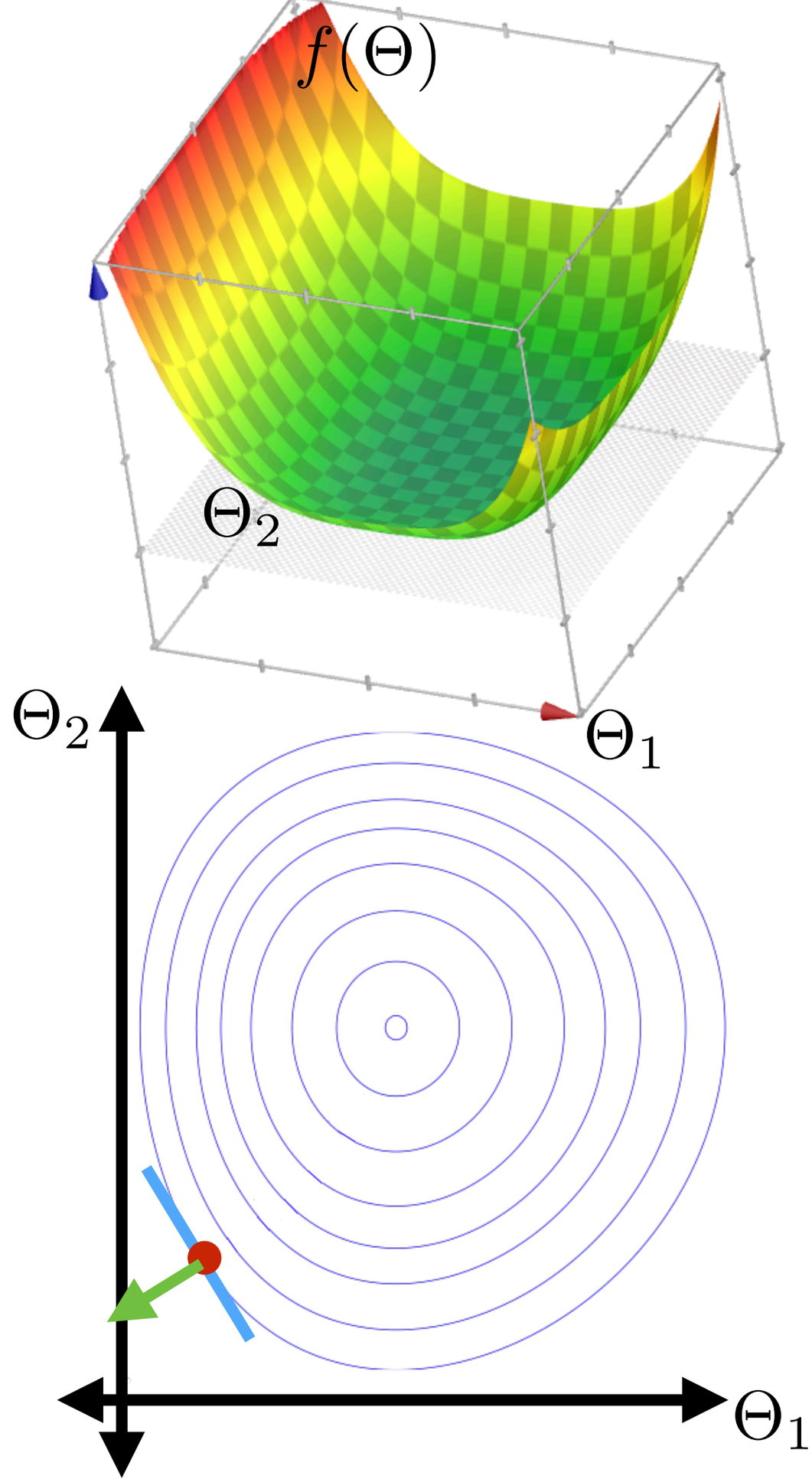
Gradient descent



Gradient descent

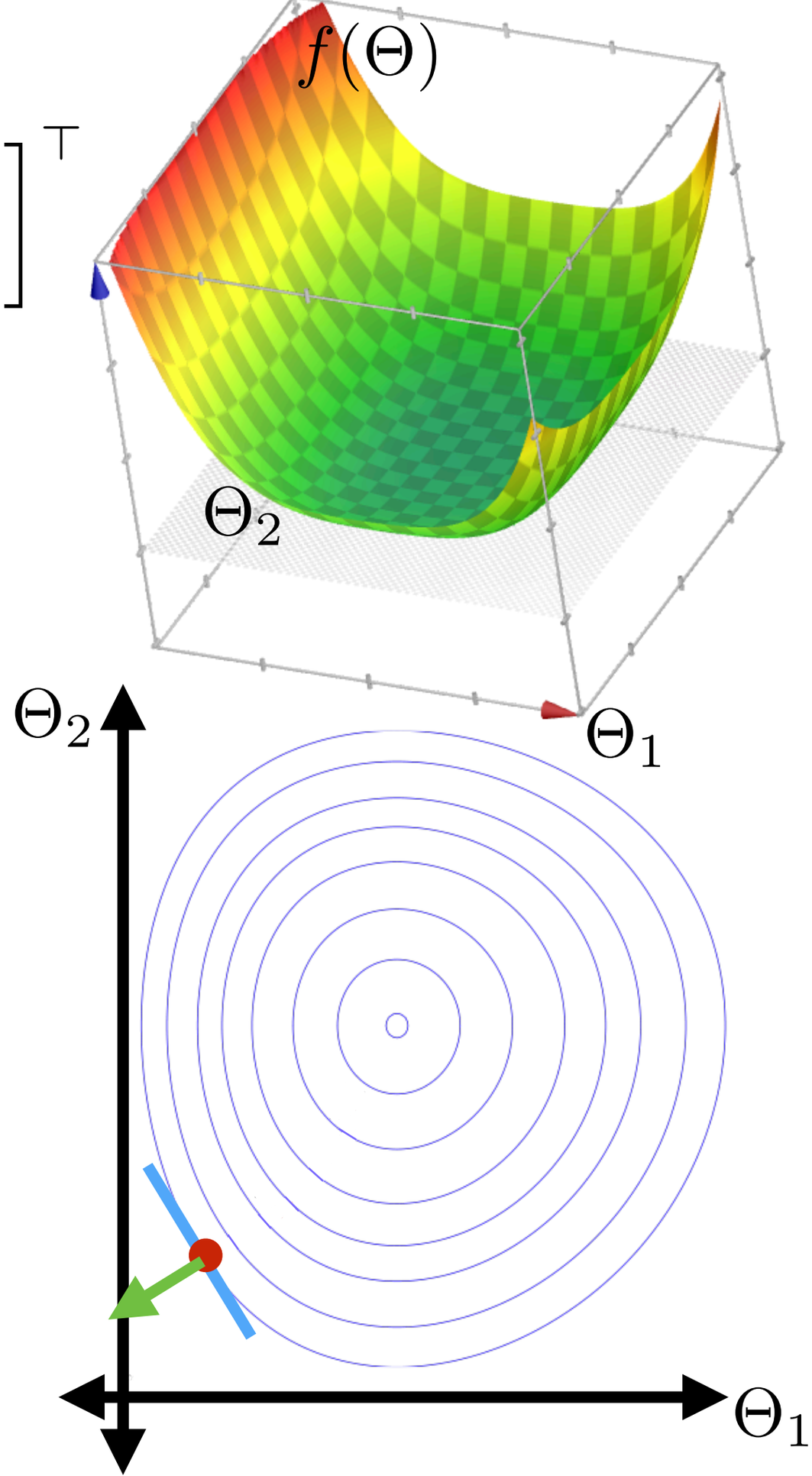


Gradient descent



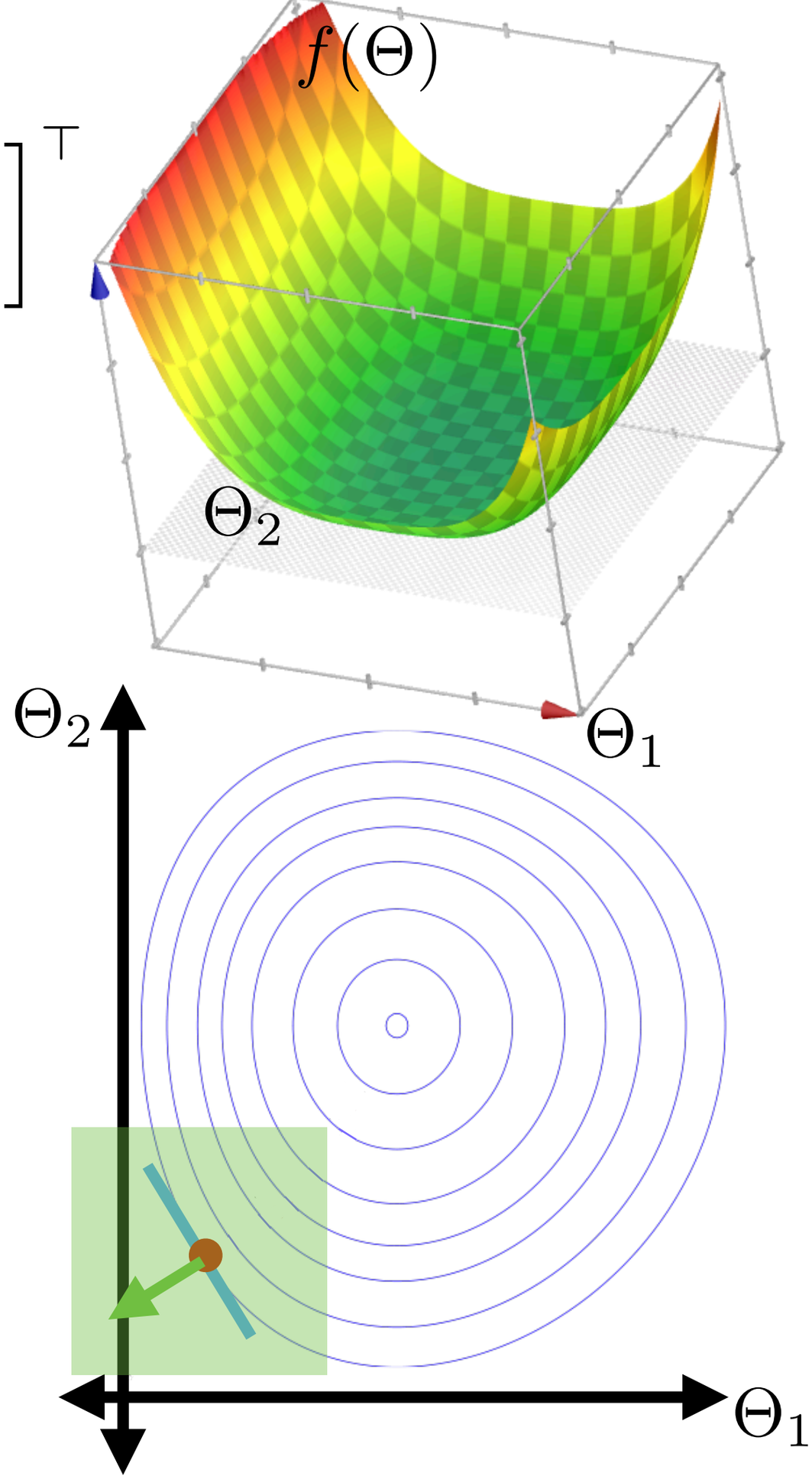
Gradient descent

- Gradient $\nabla_\Theta f = \left[\frac{\partial f}{\partial \Theta_1}, \ldots, \frac{\partial f}{\partial \Theta_m}\right]^\top$
  - with $\Theta \in \mathbb{R}^m$



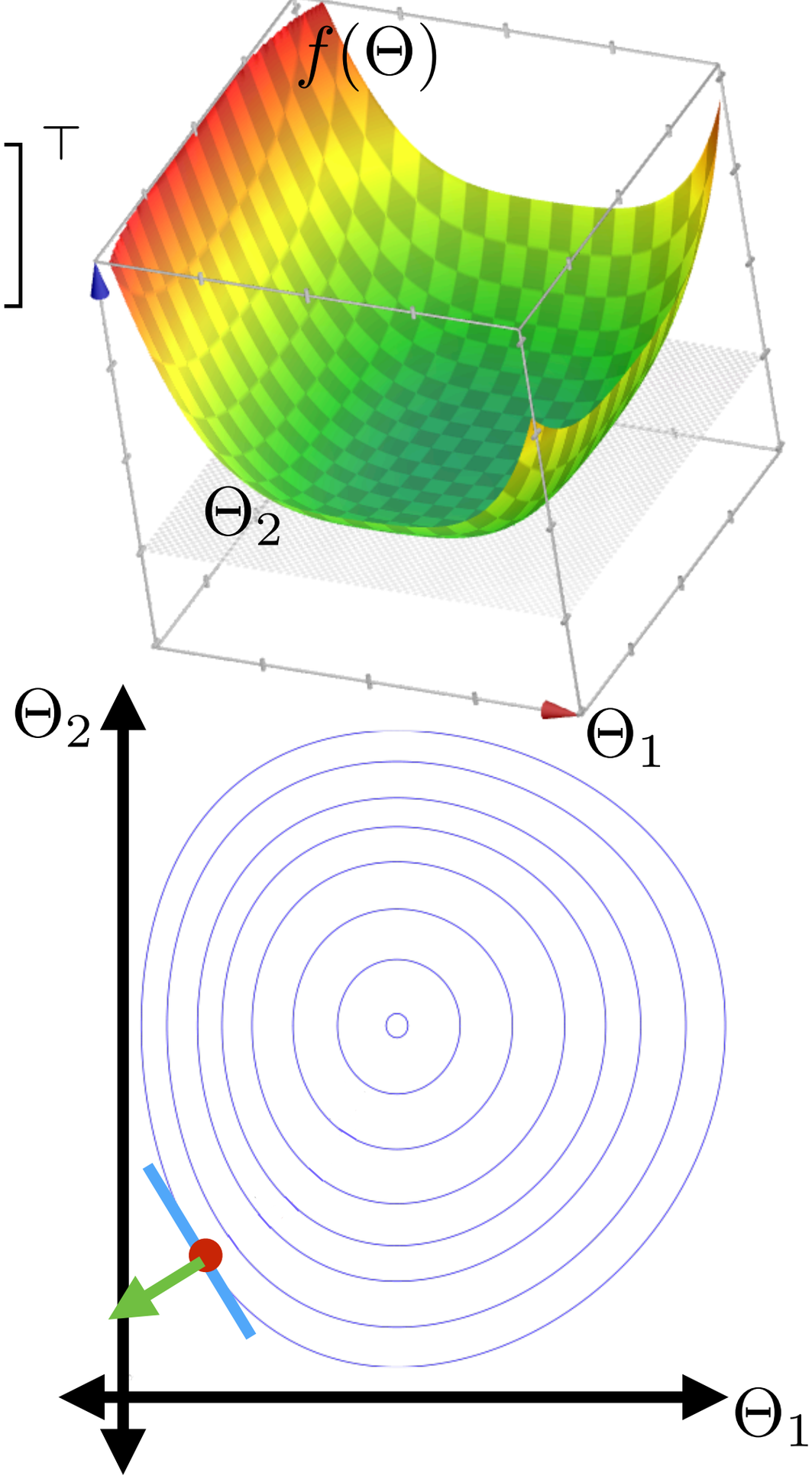
Gradient descent

- Gradient $\nabla_\Theta f = \left[\frac{\partial f}{\partial \Theta_1}, \ldots, \frac{\partial f}{\partial \Theta_m}\right]^\top$
  - with $\Theta \in \mathbb{R}^m$



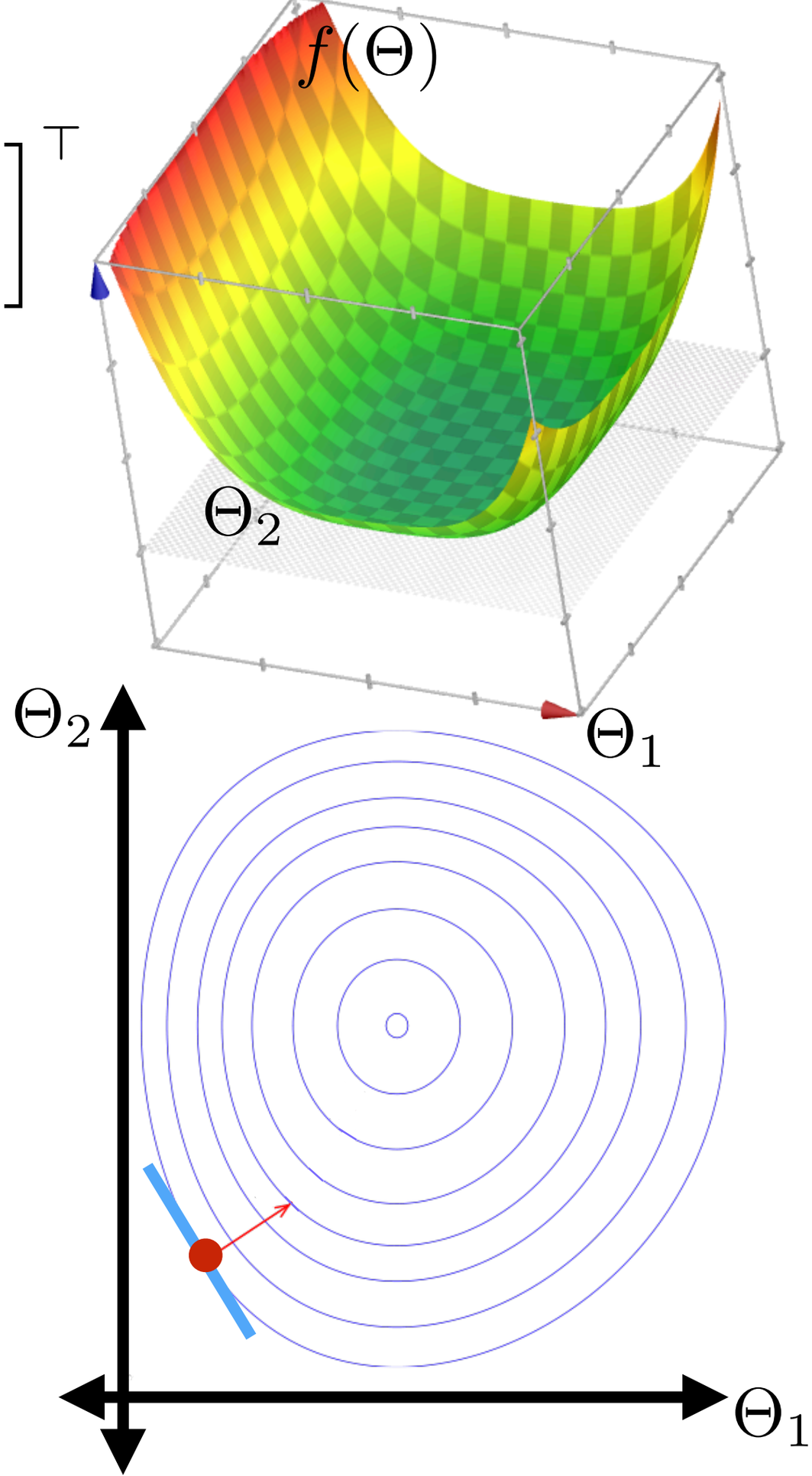
Gradient descent

- Gradient $\nabla_\Theta f = \left[\frac{\partial f}{\partial \Theta_1}, \ldots, \frac{\partial f}{\partial \Theta_m}\right]^\top$
  - with $\Theta \in \mathbb{R}^m$



Gradient descent

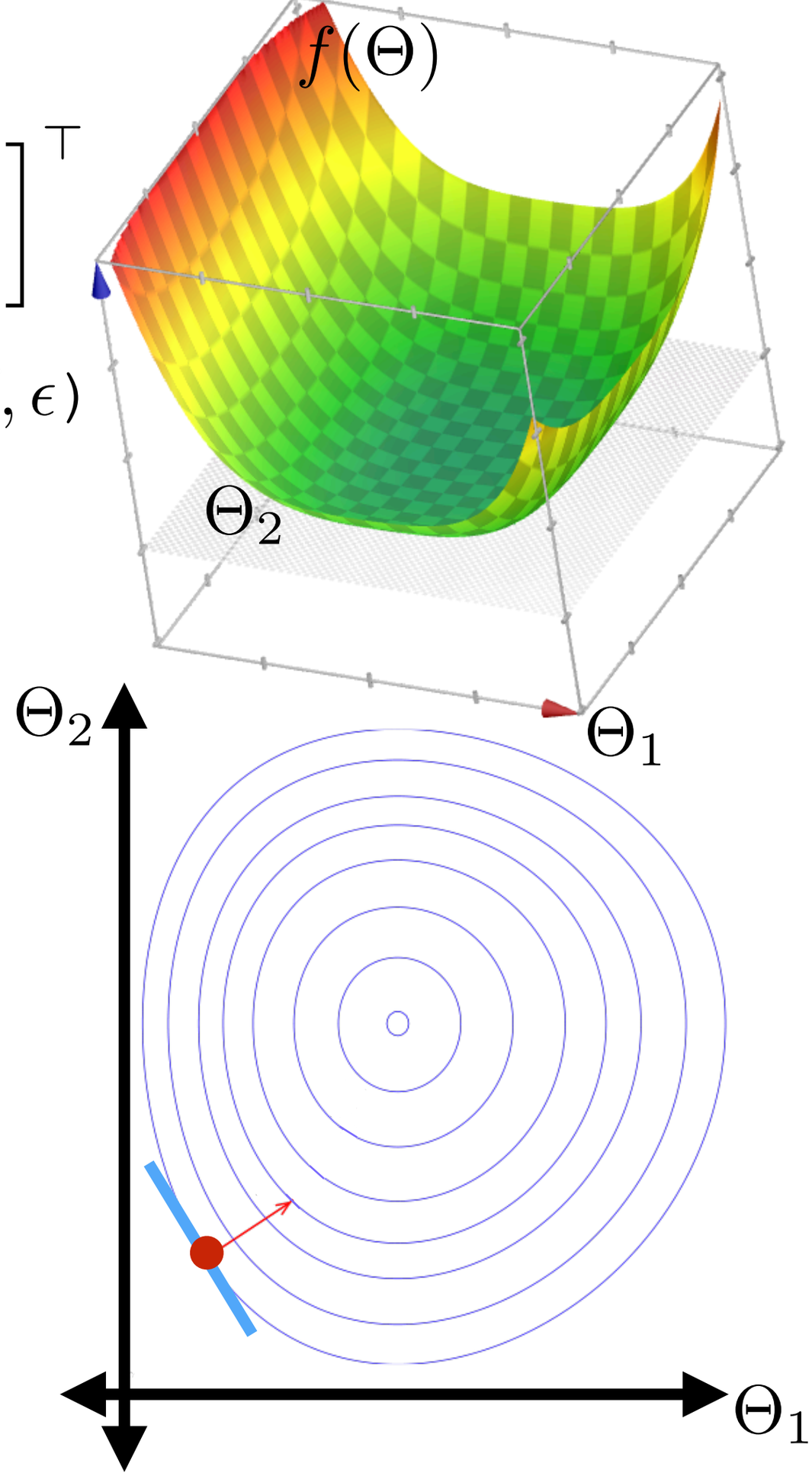
- Gradient $\nabla_\Theta f = \left[\frac{\partial f}{\partial \Theta_1}, \ldots, \frac{\partial f}{\partial \Theta_m}\right]^\top$
  - with $\Theta \in \mathbb{R}^m$



Gradient descent

- Gradient $\nabla_\Theta f = \left[\frac{\partial f}{\partial \Theta_1}, \ldots, \frac{\partial f}{\partial \Theta_m}\right]^\top$
  - with $\Theta \in \mathbb{R}^m$

`Gradient-Descent` $(\Theta_{\text{init}}, \eta, f, \nabla_\Theta f, \epsilon)$

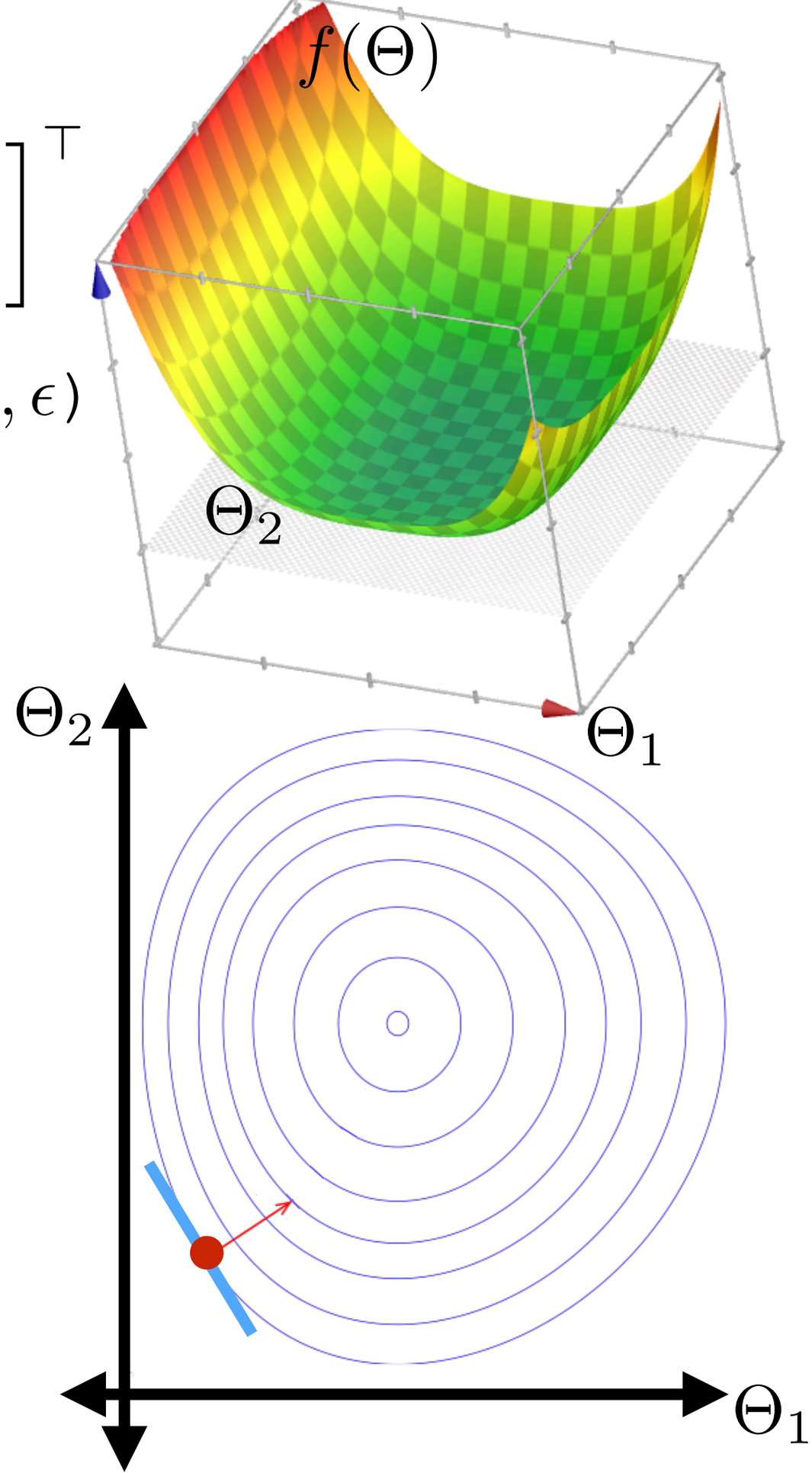


Gradient descent

- Gradient $\nabla_\Theta f = \left[\frac{\partial f}{\partial \Theta_1}, \ldots, \frac{\partial f}{\partial \Theta_m}\right]^\top$
  - with $\Theta \in \mathbb{R}^m$

`Gradient-Descent` $(\Theta_{\text{init}}, \eta, f, \nabla_\Theta f, \epsilon)$

  `Initialize` $\Theta^{(0)} = \Theta_{\text{init}}$



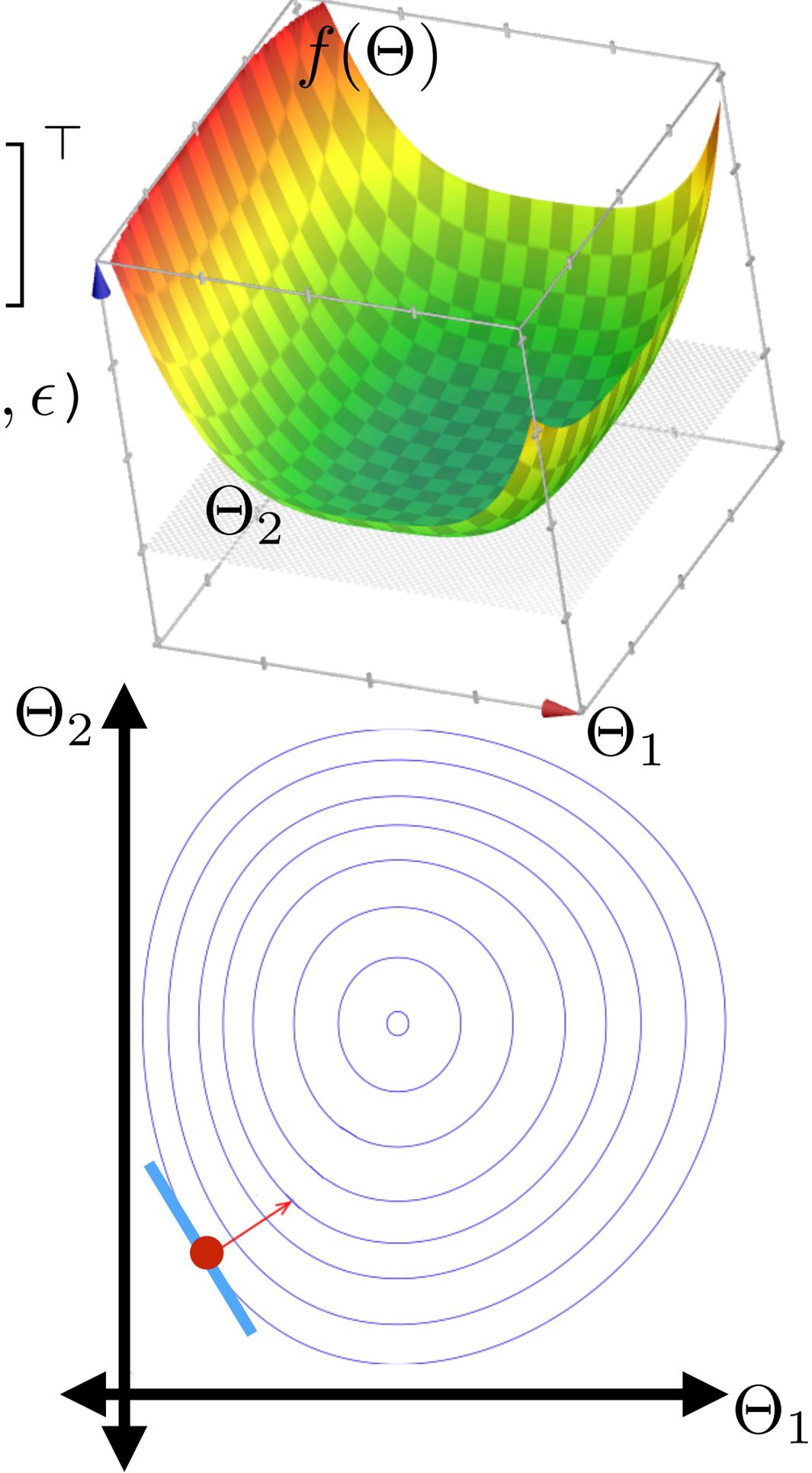
Gradient descent

- Gradient $\nabla_\Theta f = \left[\frac{\partial f}{\partial \Theta_1}, \ldots, \frac{\partial f}{\partial \Theta_m}\right]^\top$
  - with $\Theta \in \mathbb{R}^m$

`Gradient-Descent` $(\Theta_{\text{init}}, \eta, f, \nabla_\Theta f, \epsilon)$

  `Initialize` $\Theta^{(0)} = \Theta_{\text{init}}$

  `Initialize t = 0`



Gradient descent

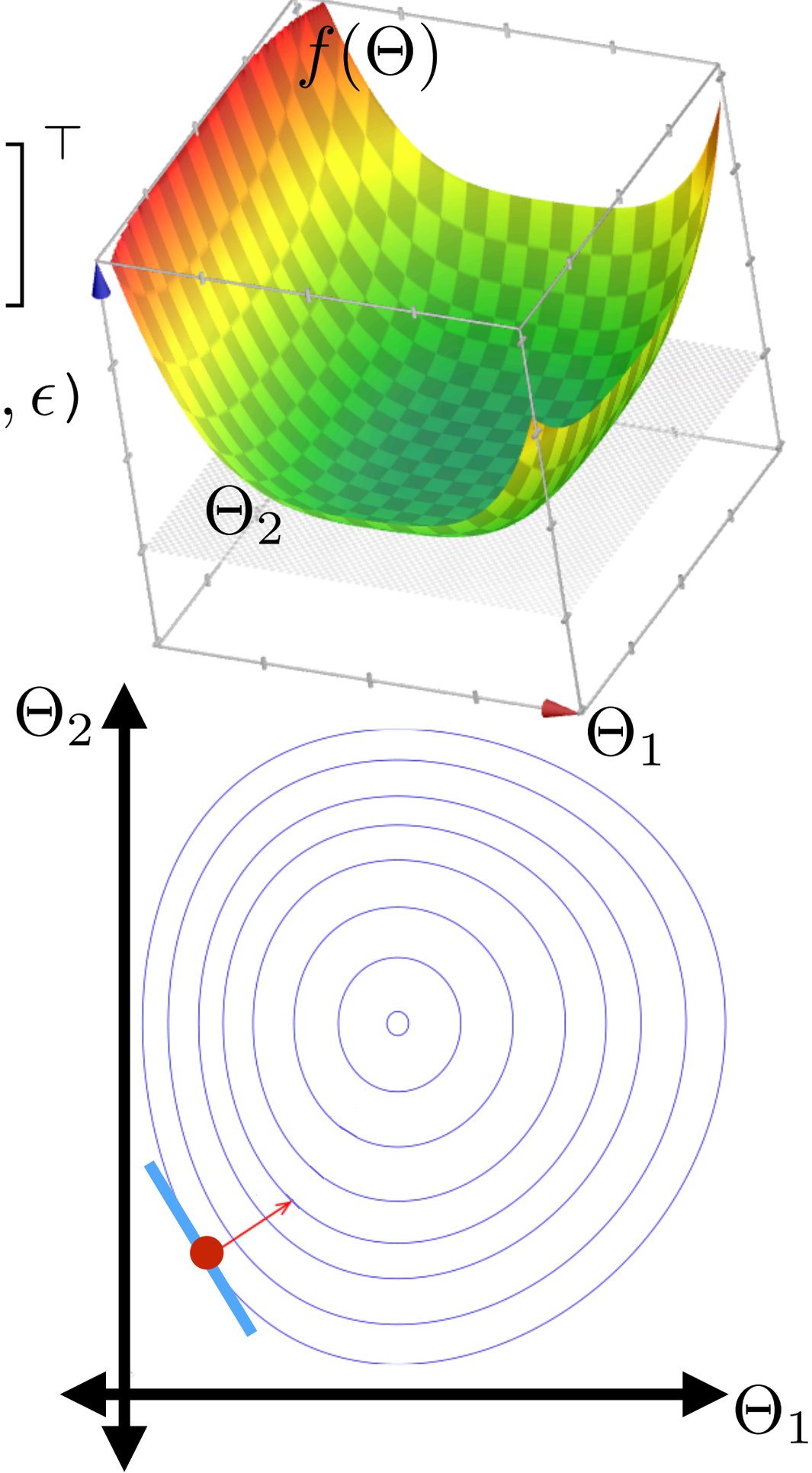
- Gradient $\nabla_\Theta f = \left[\frac{\partial f}{\partial \Theta_1}, \ldots, \frac{\partial f}{\partial \Theta_m}\right]^\top$
  - with $\Theta \in \mathbb{R}^m$

`Gradient-Descent` $(\Theta_{\text{init}}, \eta, f, \nabla_\Theta f, \epsilon)$

`Initialize` $\Theta^{(0)} = \Theta_{\text{init}}$

`Initialize t = 0`

`repeat`



Gradient descent

- Gradient $\nabla_\Theta f = \left[\frac{\partial f}{\partial \Theta_1}, \ldots, \frac{\partial f}{\partial \Theta_m}\right]^\top$
  - with $\Theta \in \mathbb{R}^m$

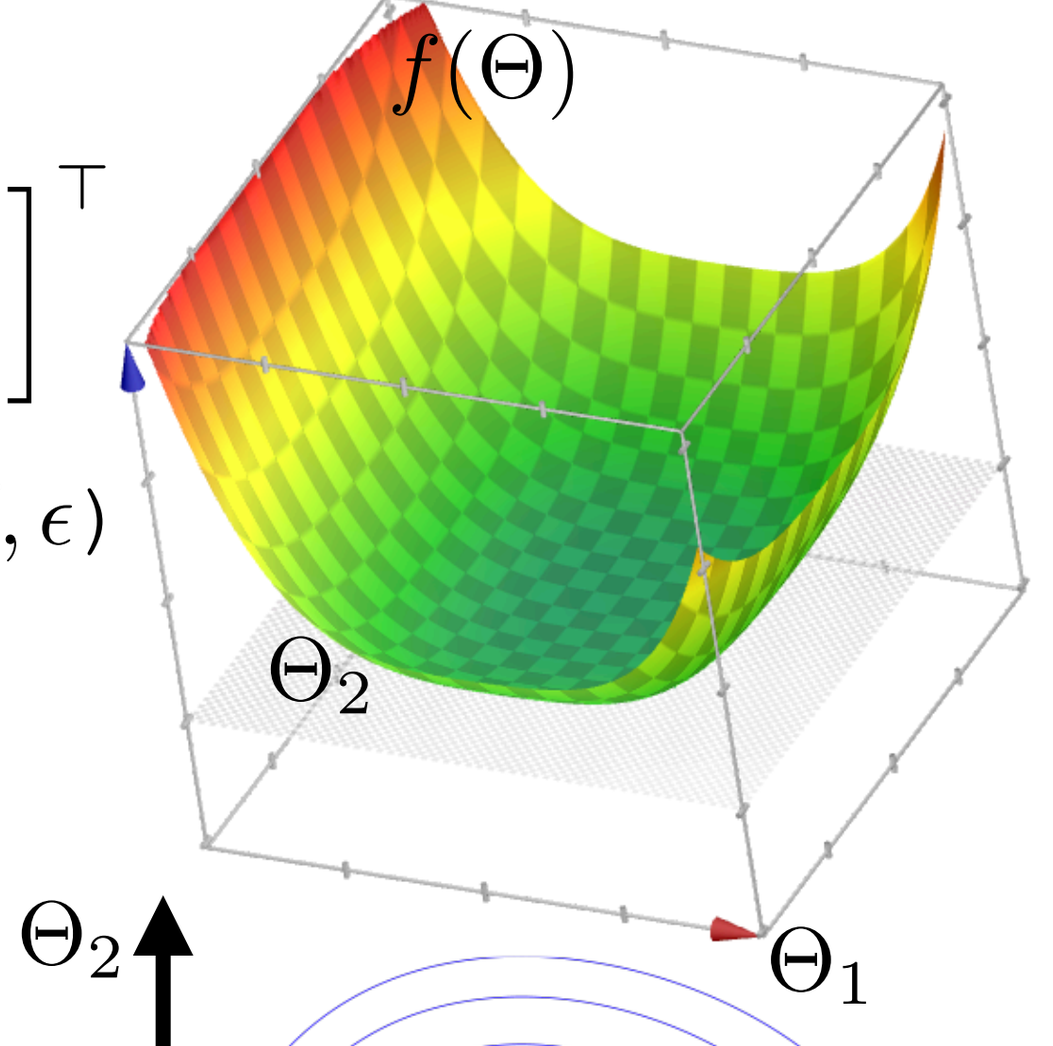
`Gradient-Descent` $(\Theta_{\text{init}}, \eta, f, \nabla_\Theta f, \epsilon)$

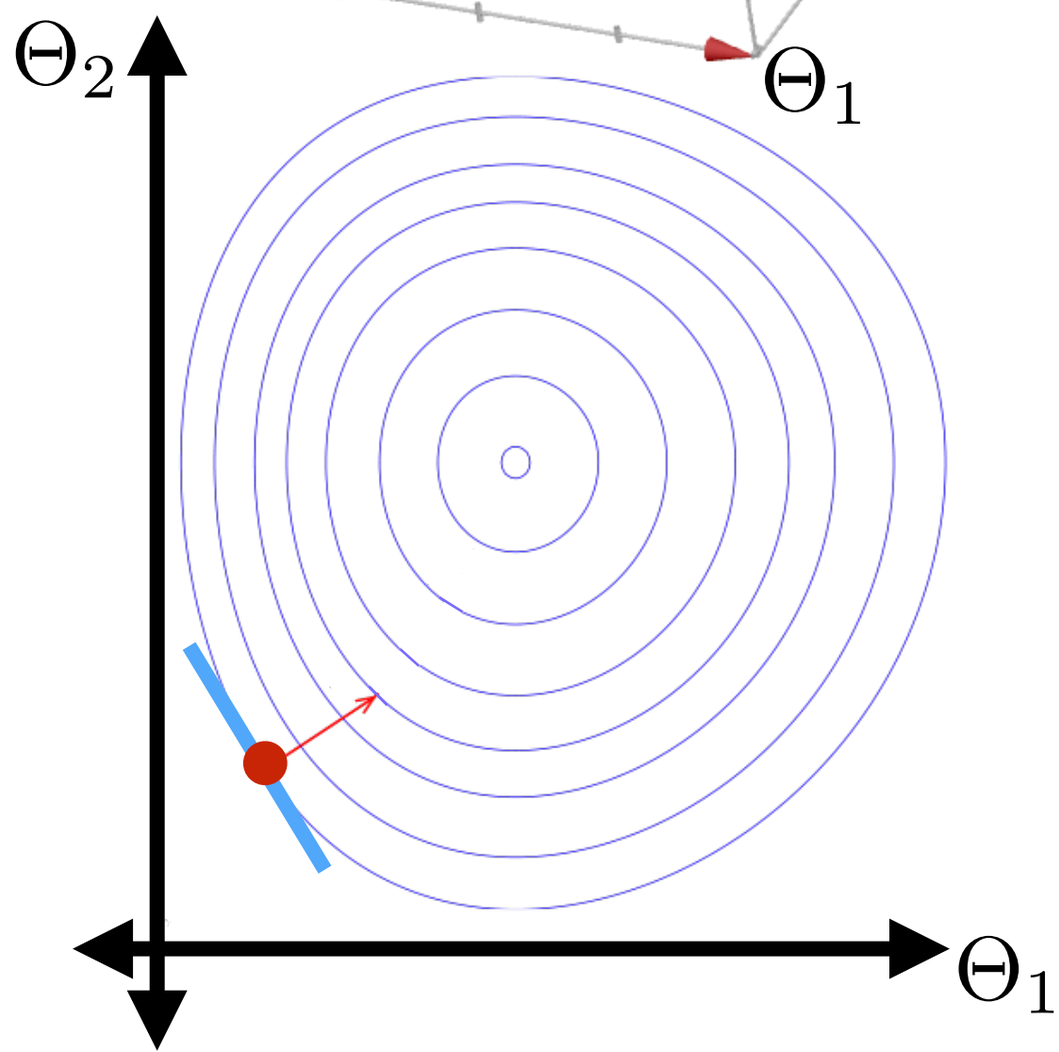
`Initialize` $\Theta^{(0)} = \Theta_{\text{init}}$

`Initialize t = 0`

`repeat`

`t = t + 1`





Gradient descent

- Gradient $\nabla_\Theta f = \left[\frac{\partial f}{\partial \Theta_1}, \ldots, \frac{\partial f}{\partial \Theta_m}\right]^\top$
  - with $\Theta \in \mathbb{R}^m$

`Gradient-Descent` $(\Theta_{\text{init}}, \eta, f, \nabla_\Theta f, \epsilon)$

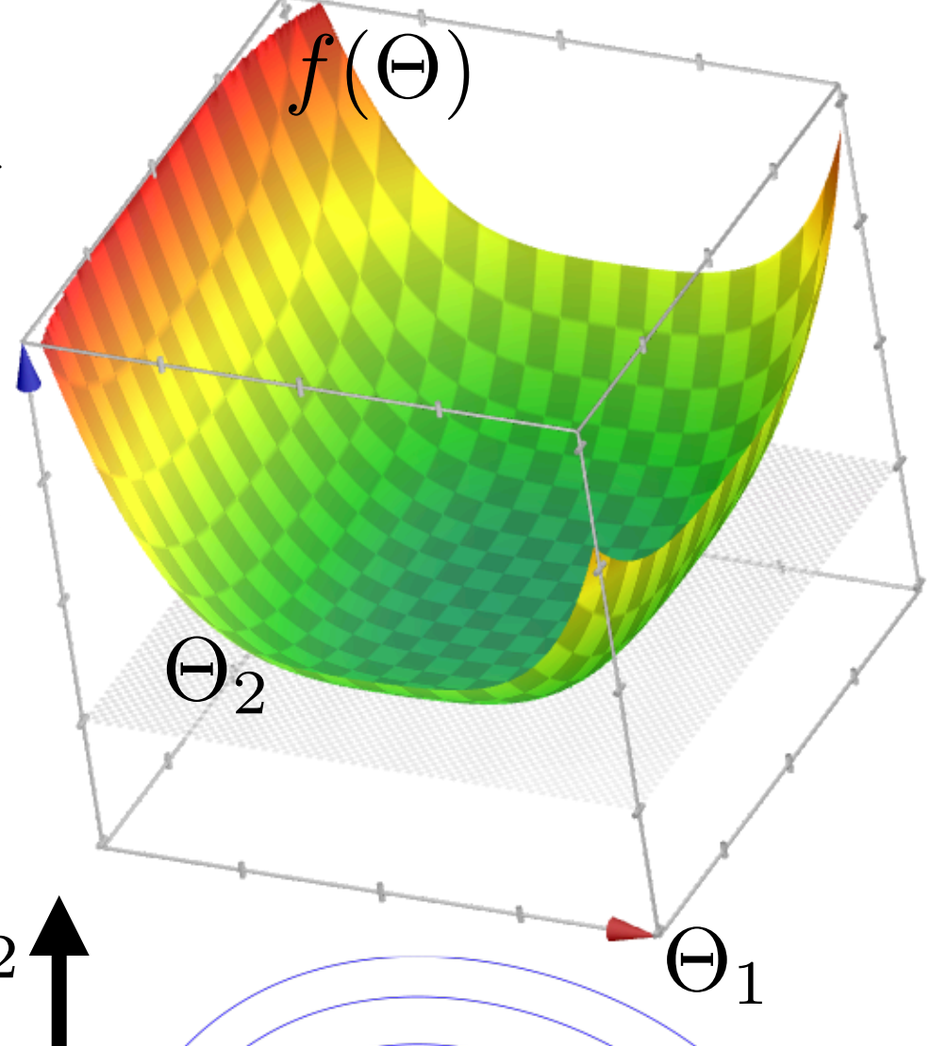
`Initialize` $\Theta^{(0)} = \Theta_{\text{init}}$

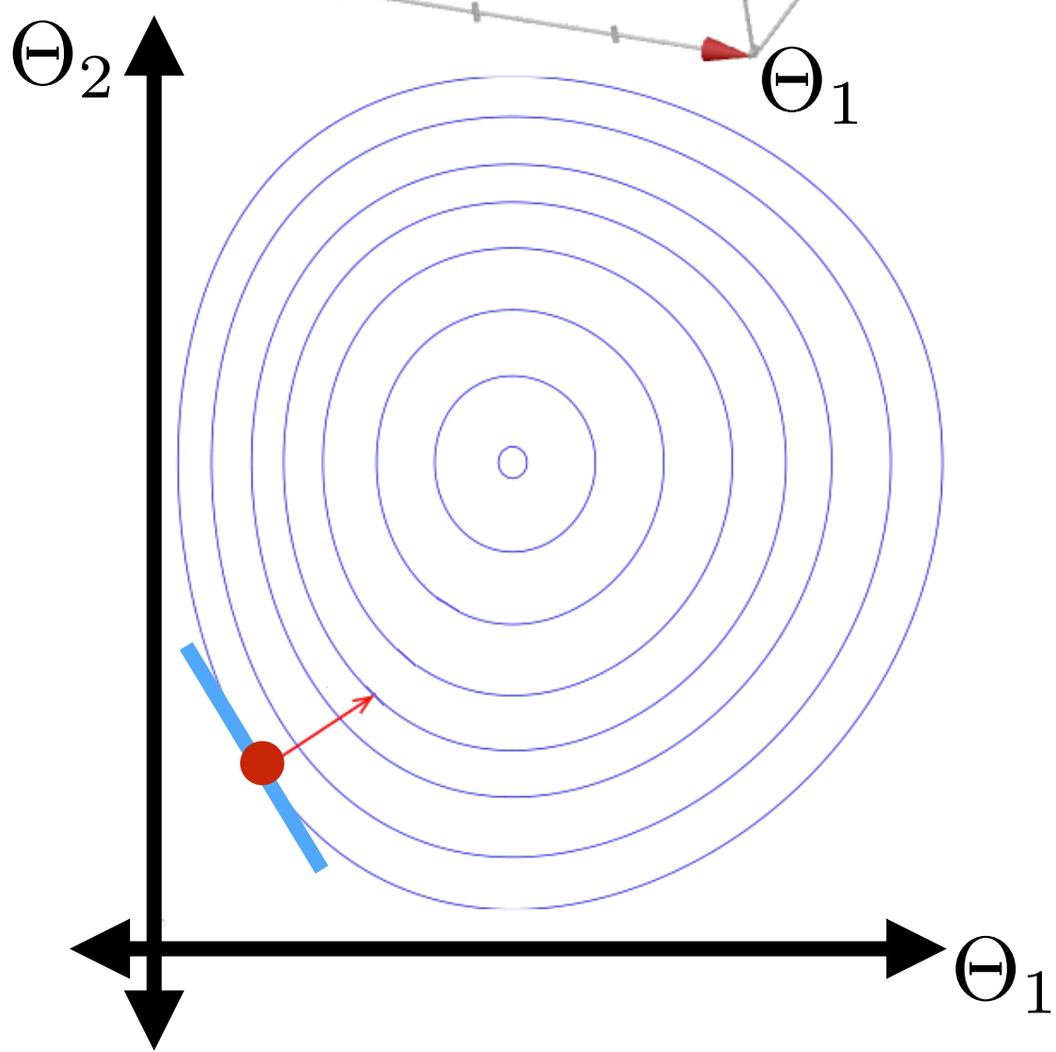
`Initialize t = 0`

`repeat`

`t = t + 1`

$\Theta^{(t)} = \Theta^{(t-1)} - \eta \nabla_\Theta f(\Theta^{(t-1)})$





Gradient descent

- Gradient $\nabla_\Theta f = \left[\frac{\partial f}{\partial \Theta_1}, \ldots, \frac{\partial f}{\partial \Theta_m}\right]^\top$
  - with $\Theta \in \mathbb{R}^m$

`Gradient-Descent` $(\Theta_{\text{init}}, \eta, f, \nabla_\Theta f, \epsilon)$

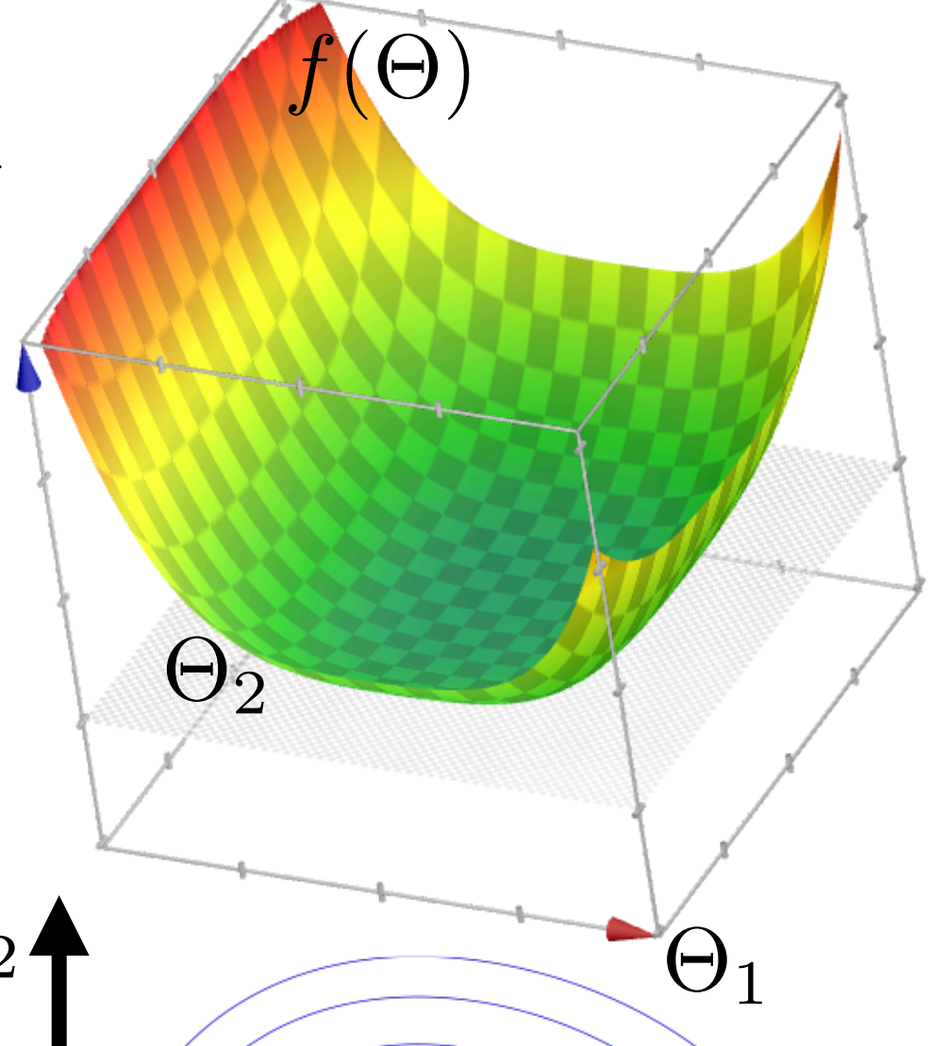
`Initialize` $\Theta^{(0)} = \Theta_{\text{init}}$

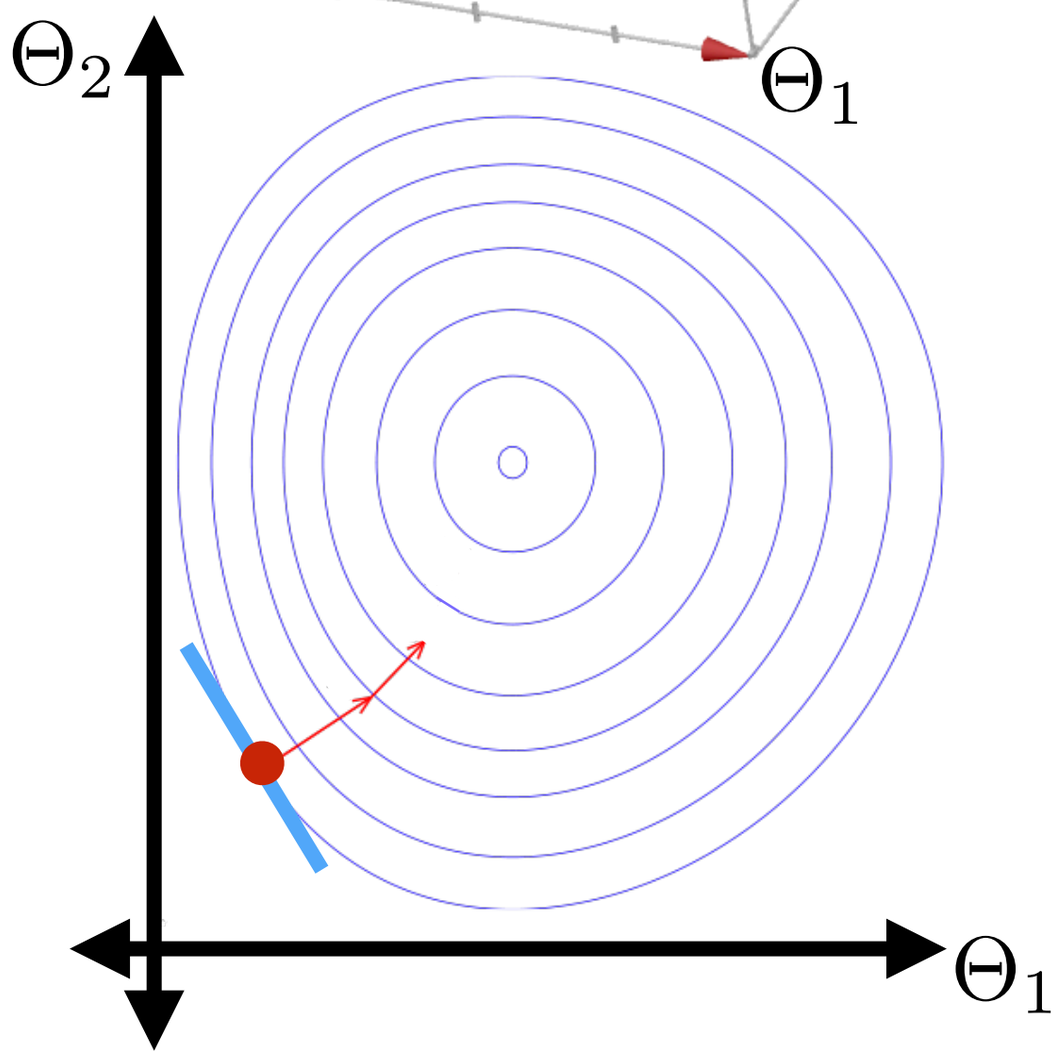
`Initialize t = 0`

`repeat`

`t = t + 1`

$\Theta^{(t)} = \Theta^{(t-1)} - \eta \nabla_\Theta f(\Theta^{(t-1)})$





Gradient descent

- Gradient $\nabla_\Theta f = \left[\frac{\partial f}{\partial \Theta_1}, \ldots, \frac{\partial f}{\partial \Theta_m}\right]^\top$
  - with $\Theta \in \mathbb{R}^m$

`Gradient-Descent` $(\Theta_{\text{init}}, \eta, f, \nabla_\Theta f, \epsilon)$

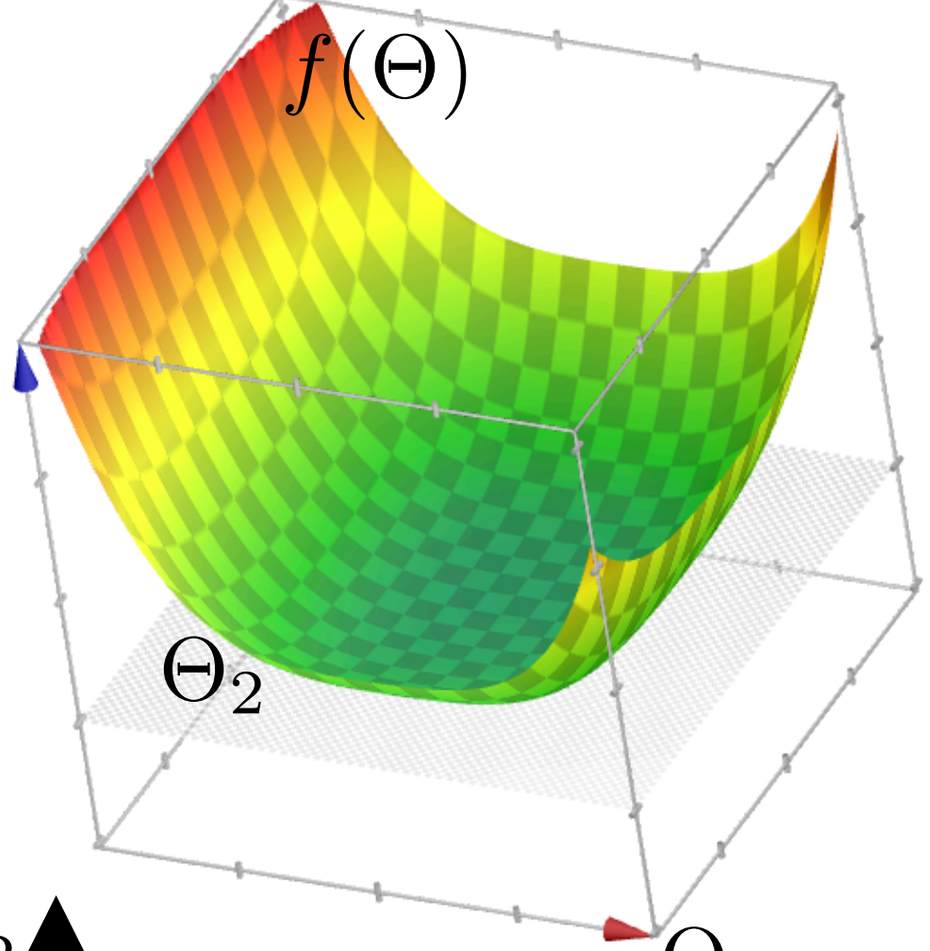
`Initialize` $\Theta^{(0)} = \Theta_{\text{init}}$

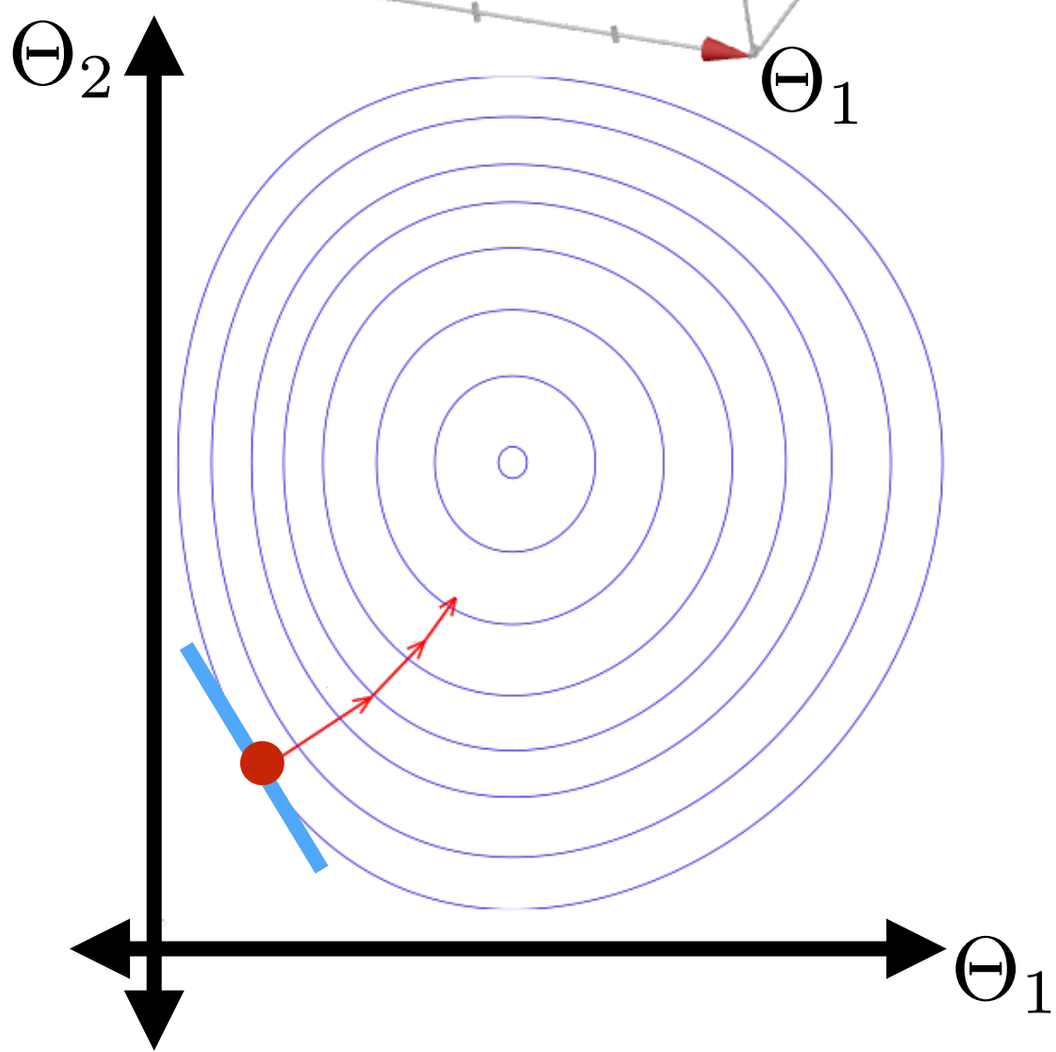
`Initialize t = 0`

`repeat`

`t = t + 1`

$\Theta^{(t)} = \Theta^{(t-1)} - \eta \nabla_\Theta f(\Theta^{(t-1)})$





Gradient descent

- Gradient $\nabla_\Theta f = \left[\frac{\partial f}{\partial \Theta_1}, \ldots, \frac{\partial f}{\partial \Theta_m}\right]^\top$
  - with $\Theta \in \mathbb{R}^m$

`Gradient-Descent` $(\Theta_{\text{init}}, \eta, f, \nabla_\Theta f, \epsilon)$

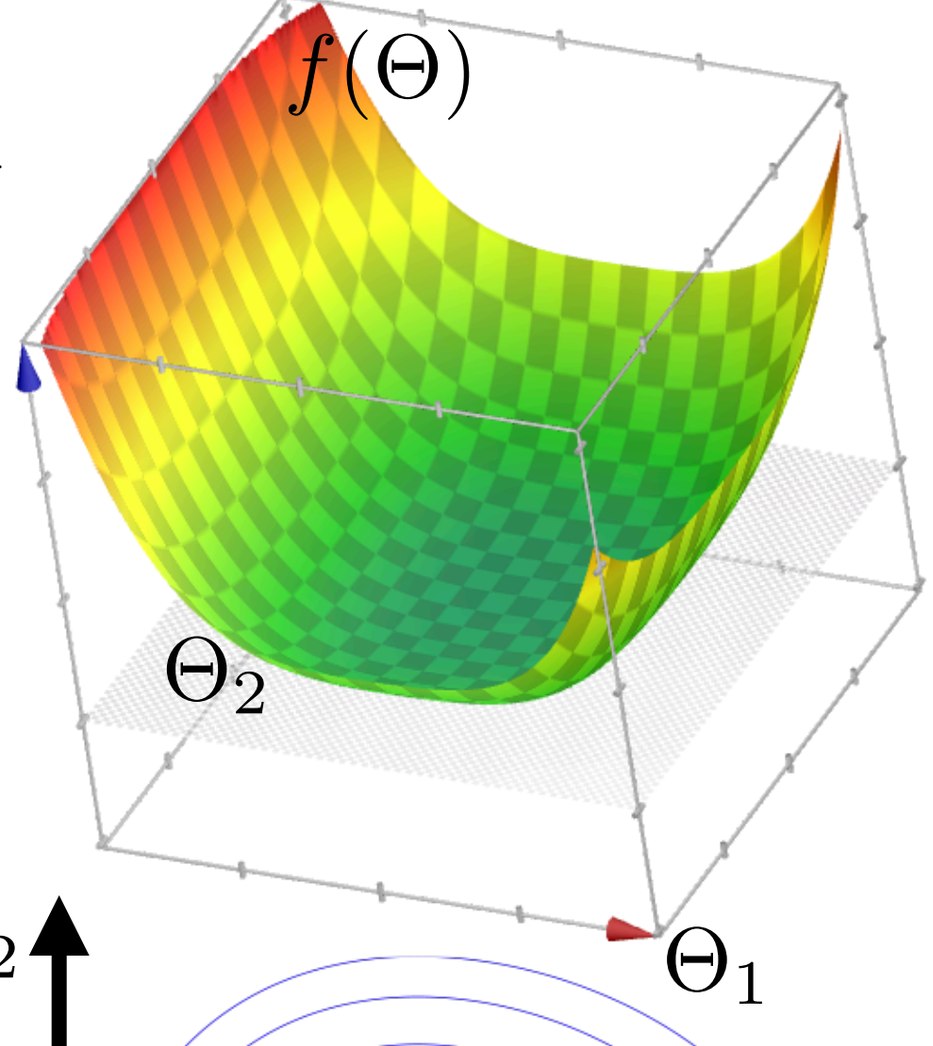
`Initialize` $\Theta^{(0)} = \Theta_{\text{init}}$

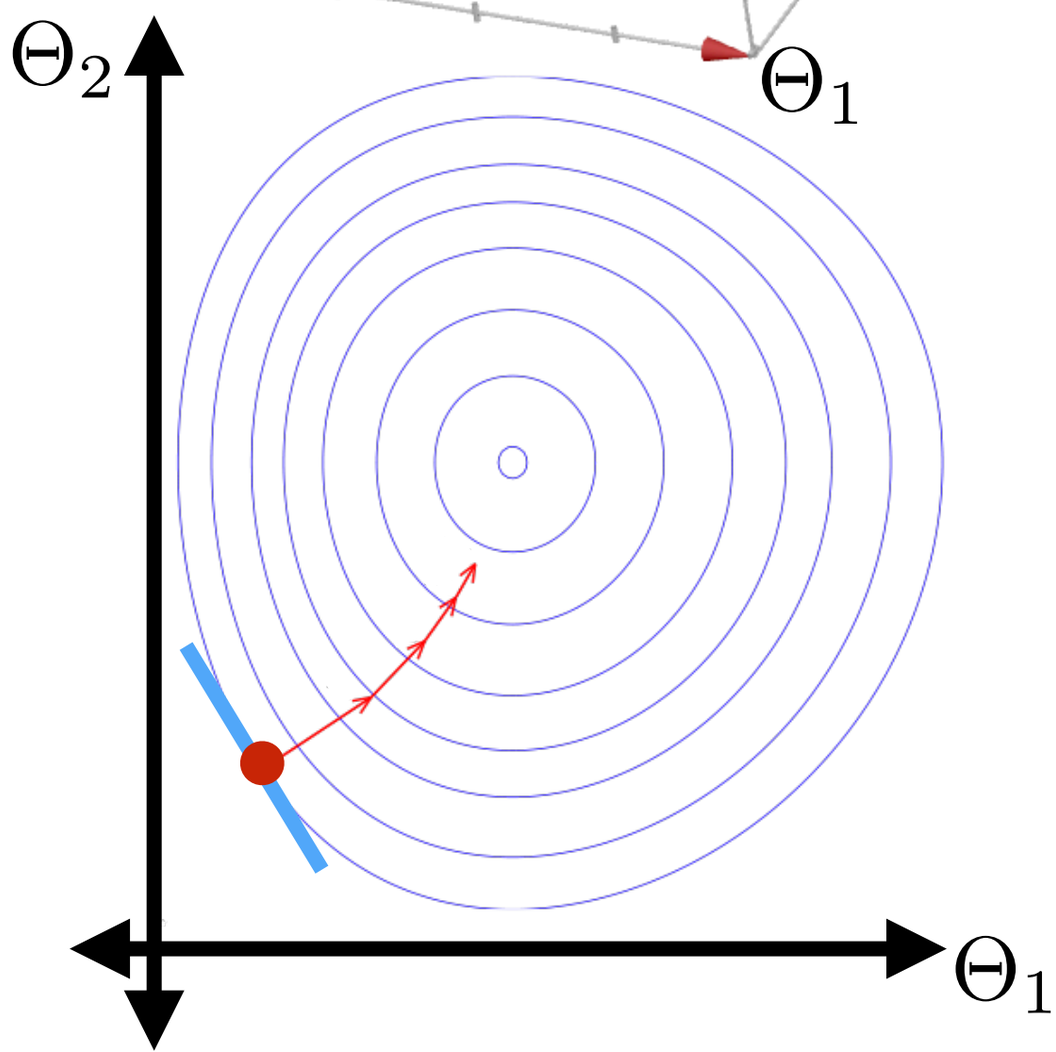
`Initialize t = 0`

`repeat`

`t = t + 1`

$\Theta^{(t)} = \Theta^{(t-1)} - \eta \nabla_\Theta f(\Theta^{(t-1)})$





Gradient descent

- Gradient $\nabla_\Theta f = \left[\frac{\partial f}{\partial \Theta_1}, \ldots, \frac{\partial f}{\partial \Theta_m}\right]^\top$
  - with $\Theta \in \mathbb{R}^m$

`Gradient-Descent` $(\Theta_{\text{init}}, \eta, f, \nabla_\Theta f, \epsilon)$

`Initialize` $\Theta^{(0)} = \Theta_{\text{init}}$

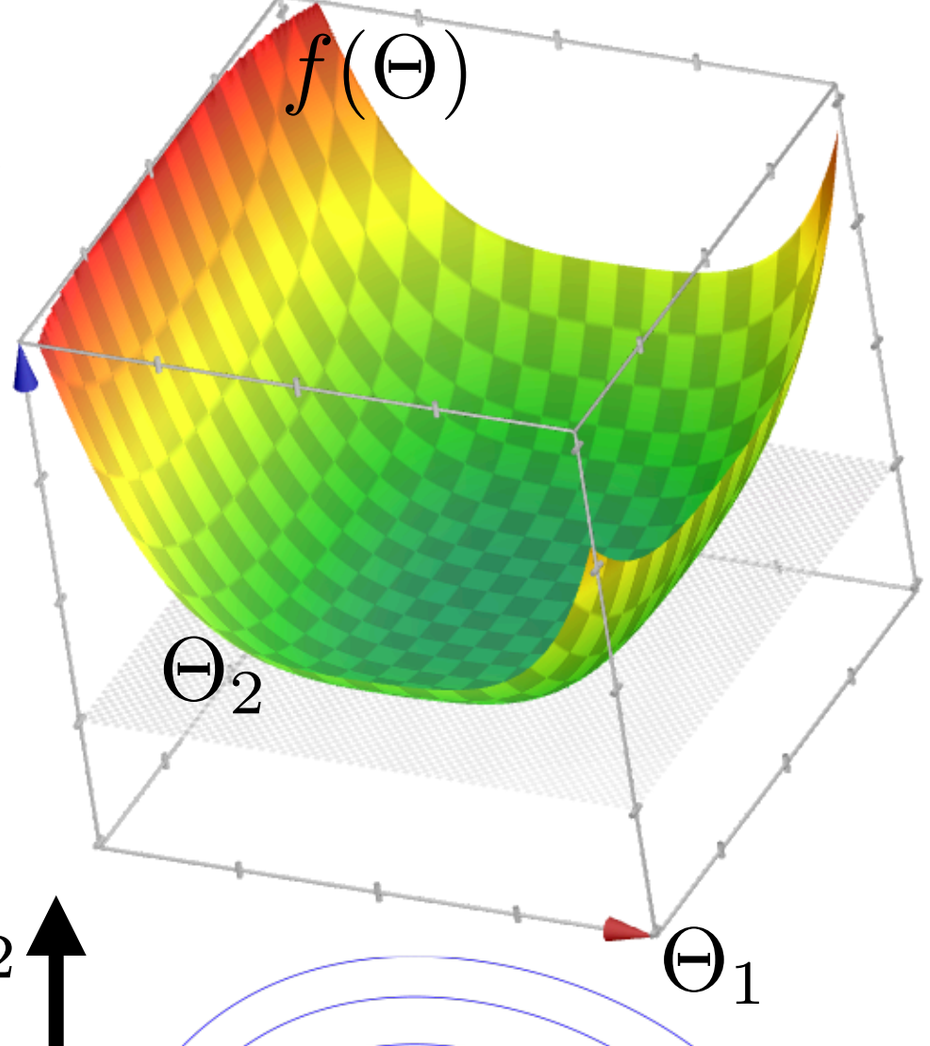
`Initialize t = 0`

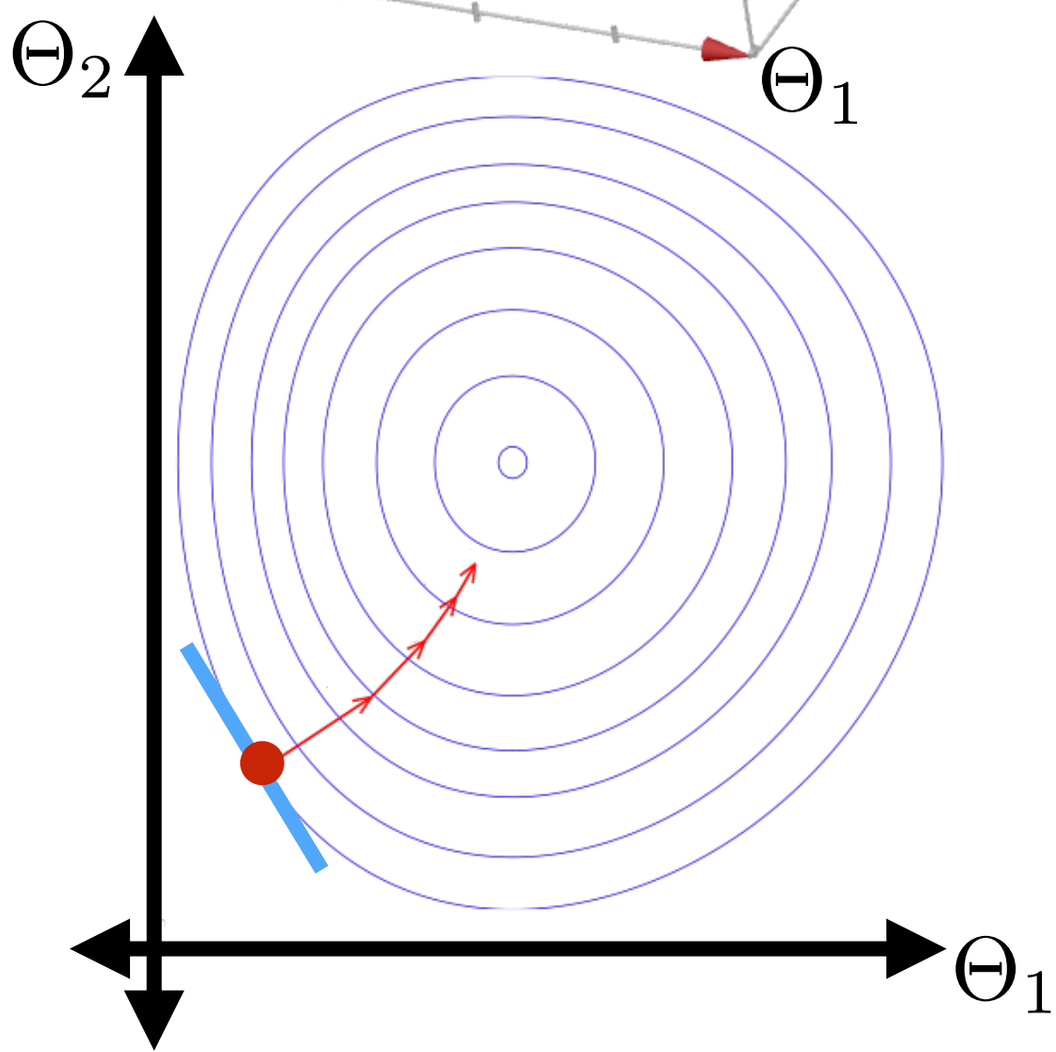
`repeat`

`t = t + 1`

$\Theta^{(t)} = \Theta^{(t-1)} - \eta \nabla_\Theta f(\Theta^{(t-1)})$

`until`





Gradient descent

- Gradient $\nabla_\Theta f = \left[\frac{\partial f}{\partial \Theta_1}, \ldots, \frac{\partial f}{\partial \Theta_m}\right]^\top$
  - with $\Theta \in \mathbb{R}^m$

Gradient-Descent $(\Theta_{\text{init}}, \eta, f, \nabla_\Theta f, \epsilon)$

  Initialize $\Theta^{(0)} = \Theta_{\text{init}}$

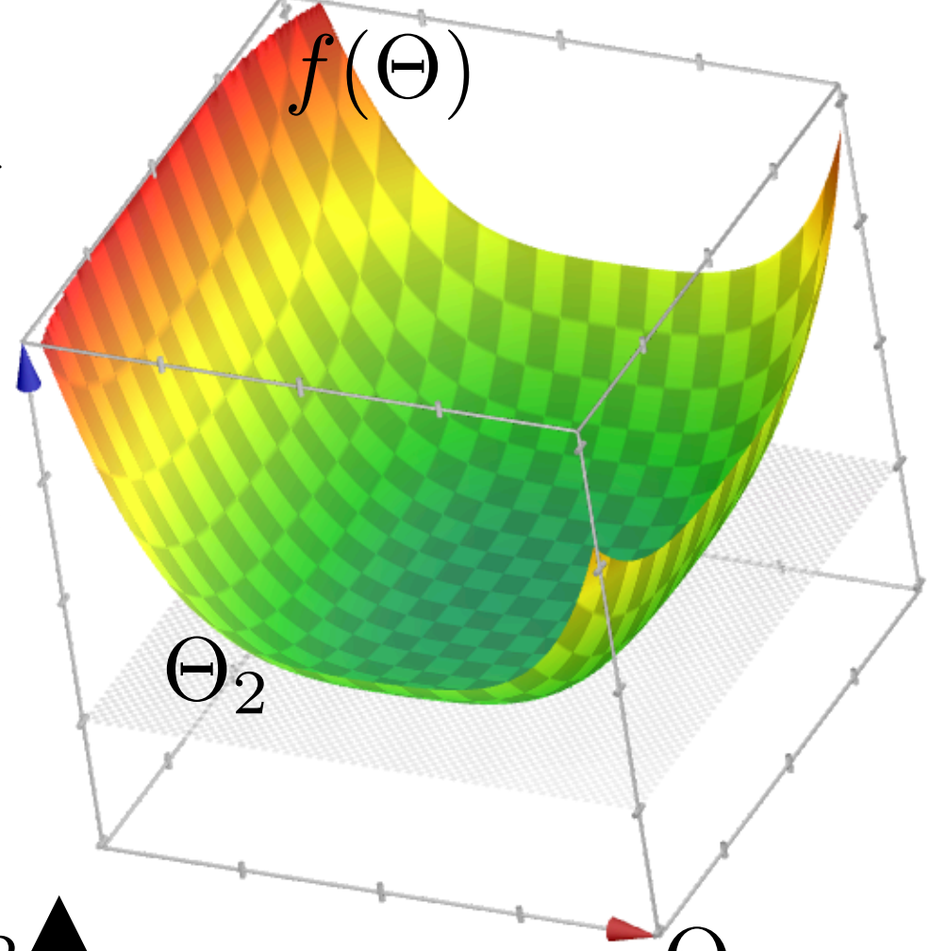
  Initialize t = 0

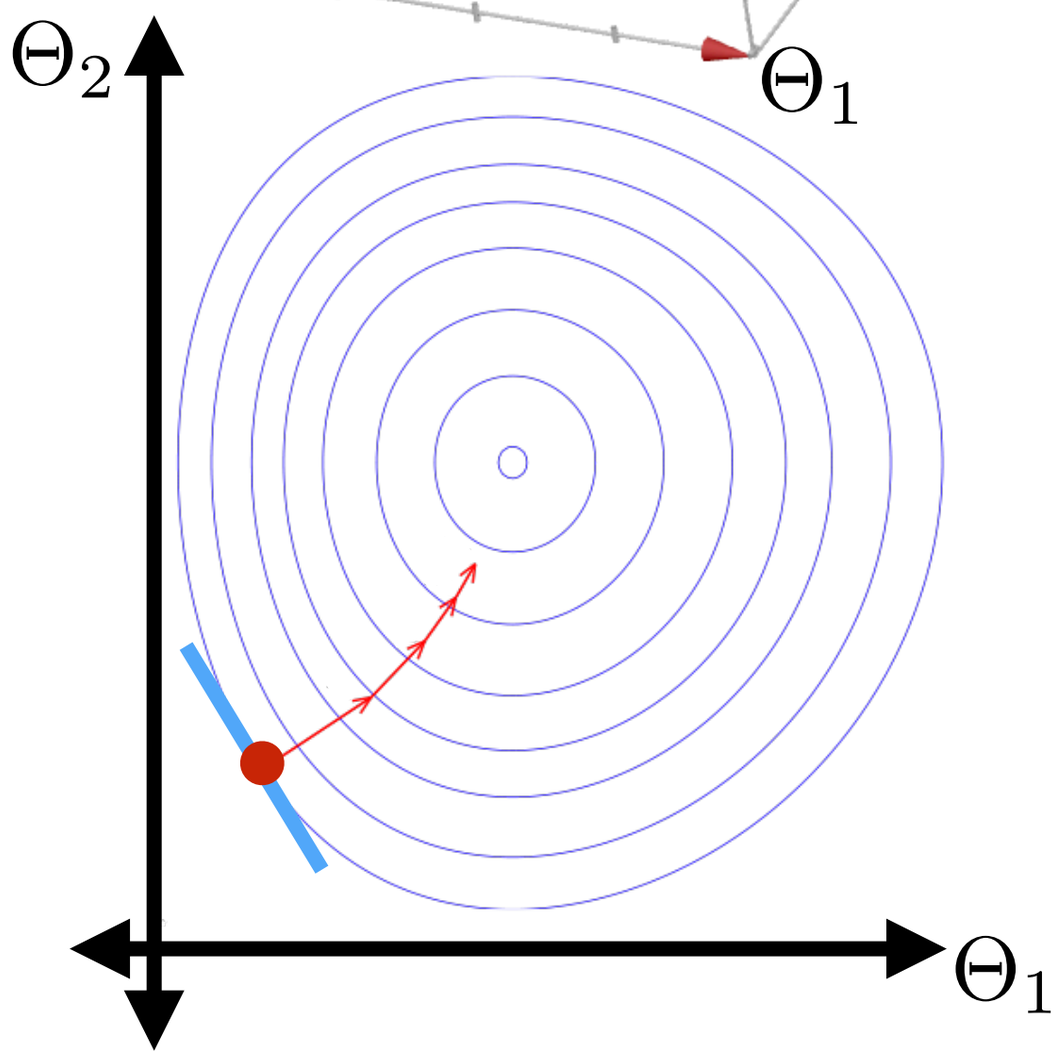
  repeat

    t = t + 1

    $\Theta^{(t)} = \Theta^{(t-1)} - \eta \nabla_\Theta f(\Theta^{(t-1)})$

  until $\left| f(\Theta^{(t)}) - f(\Theta^{(t-1)}) \right| < \epsilon$





Gradient descent

- Gradient $\nabla_\Theta f = \left[\frac{\partial f}{\partial \Theta_1}, \ldots, \frac{\partial f}{\partial \Theta_m}\right]^\top$
  - with $\Theta \in \mathbb{R}^m$

Gradient-Descent $(\Theta_{\text{init}}, \eta, f, \nabla_\Theta f, \epsilon)$

  Initialize $\Theta^{(0)} = \Theta_{\text{init}}$

  Initialize t = 0

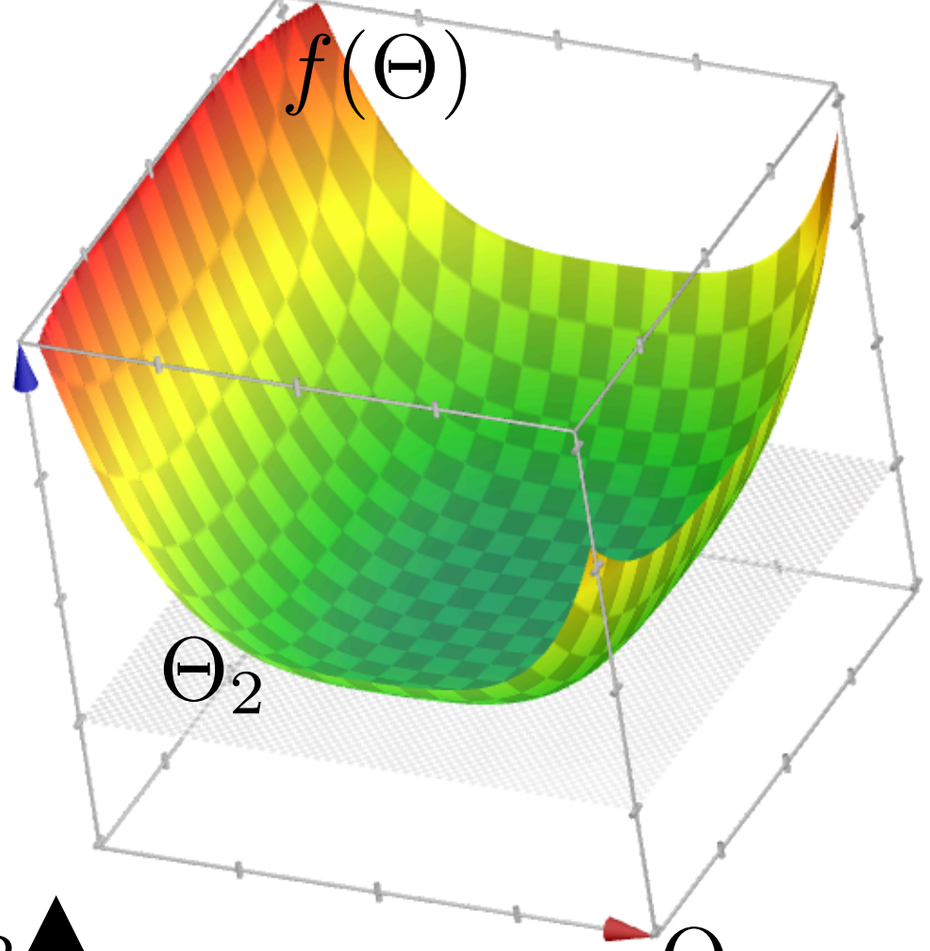
  repeat

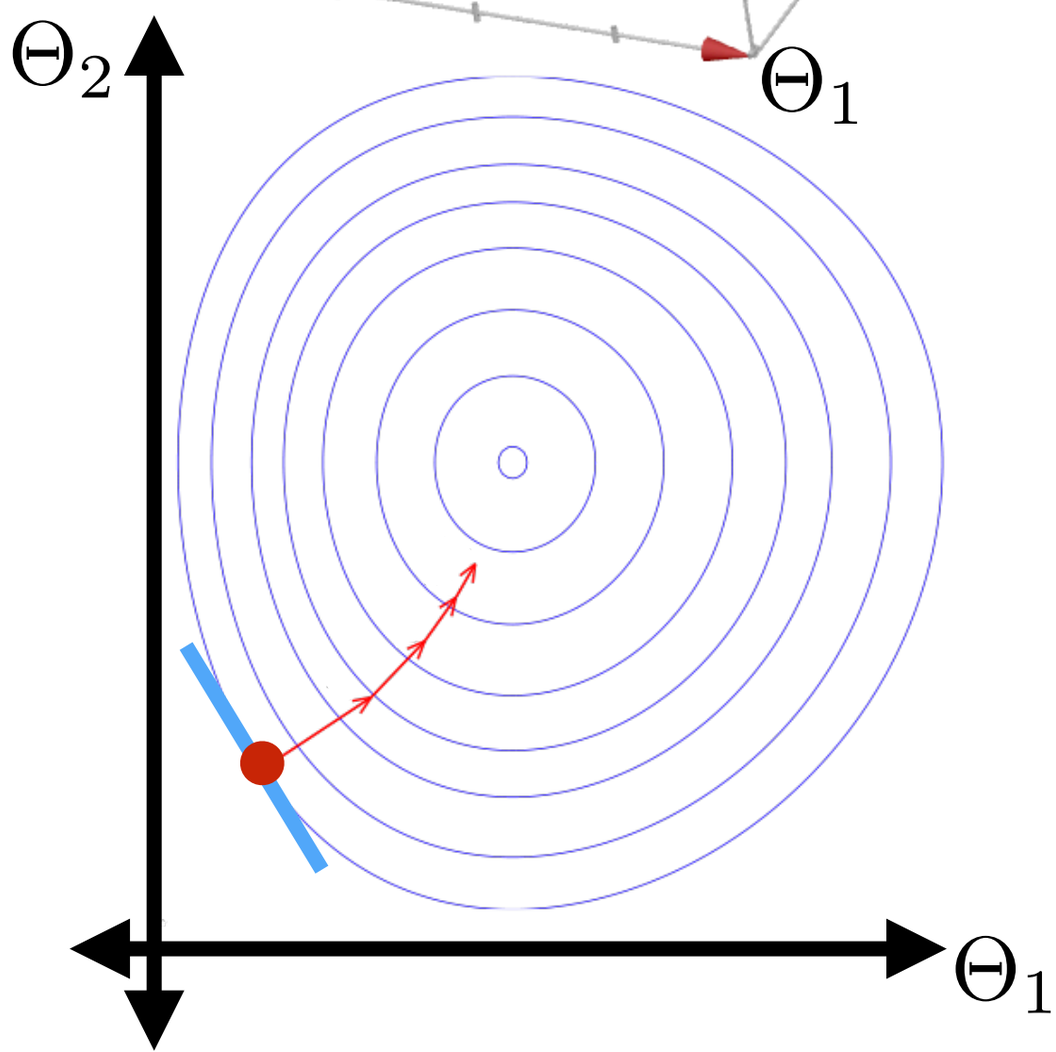
    t = t + 1

    $\Theta^{(t)} = \Theta^{(t-1)} - \eta \nabla_\Theta f(\Theta^{(t-1)})$

  until $\left| f(\Theta^{(t)}) - f(\Theta^{(t-1)}) \right| < \epsilon$

  Return $\Theta^{(t)}$





Gradient descent

- Gradient $\nabla_\Theta f = \left[\frac{\partial f}{\partial \Theta_1}, \ldots, \frac{\partial f}{\partial \Theta_m}\right]^\top$
  - with $\Theta \in \mathbb{R}^m$

Gradient-Descent $(\Theta_{\text{init}}, \eta, f, \nabla_\Theta f, \epsilon)$

  Initialize $\Theta^{(0)} = \Theta_{\text{init}}$

  Initialize t = 0

  repeat

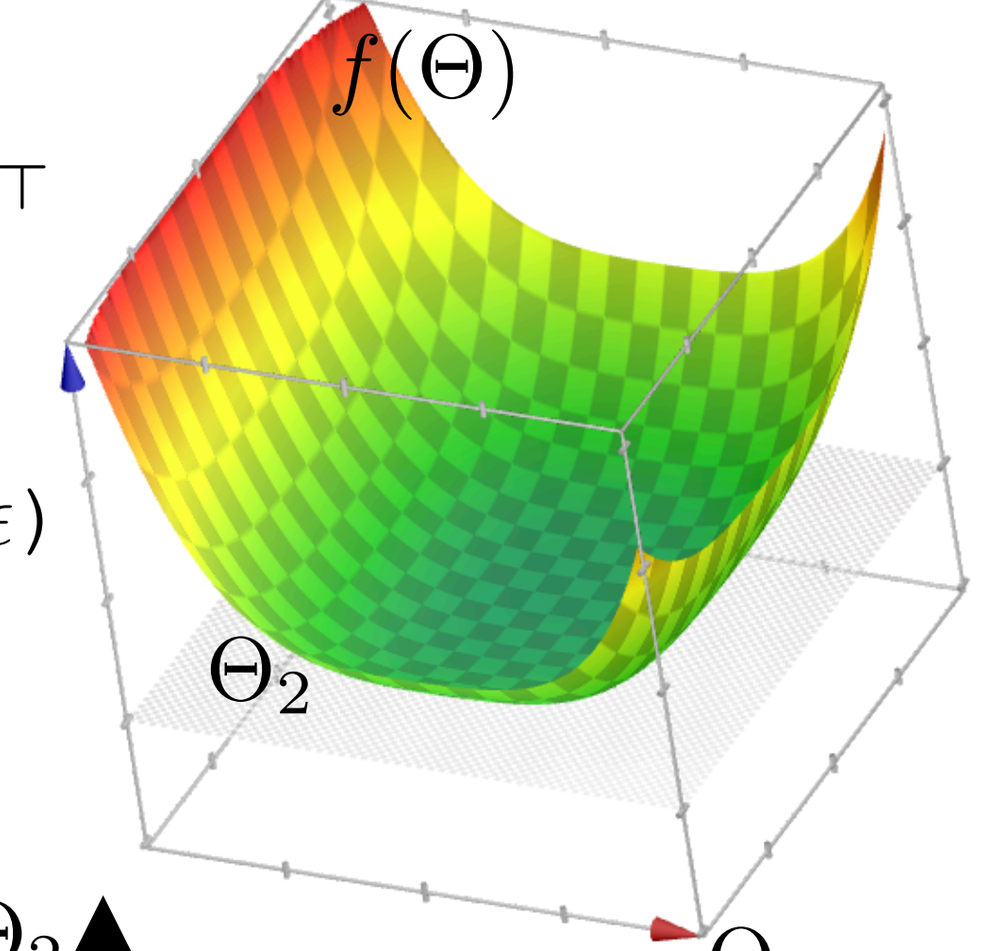
    t = t + 1

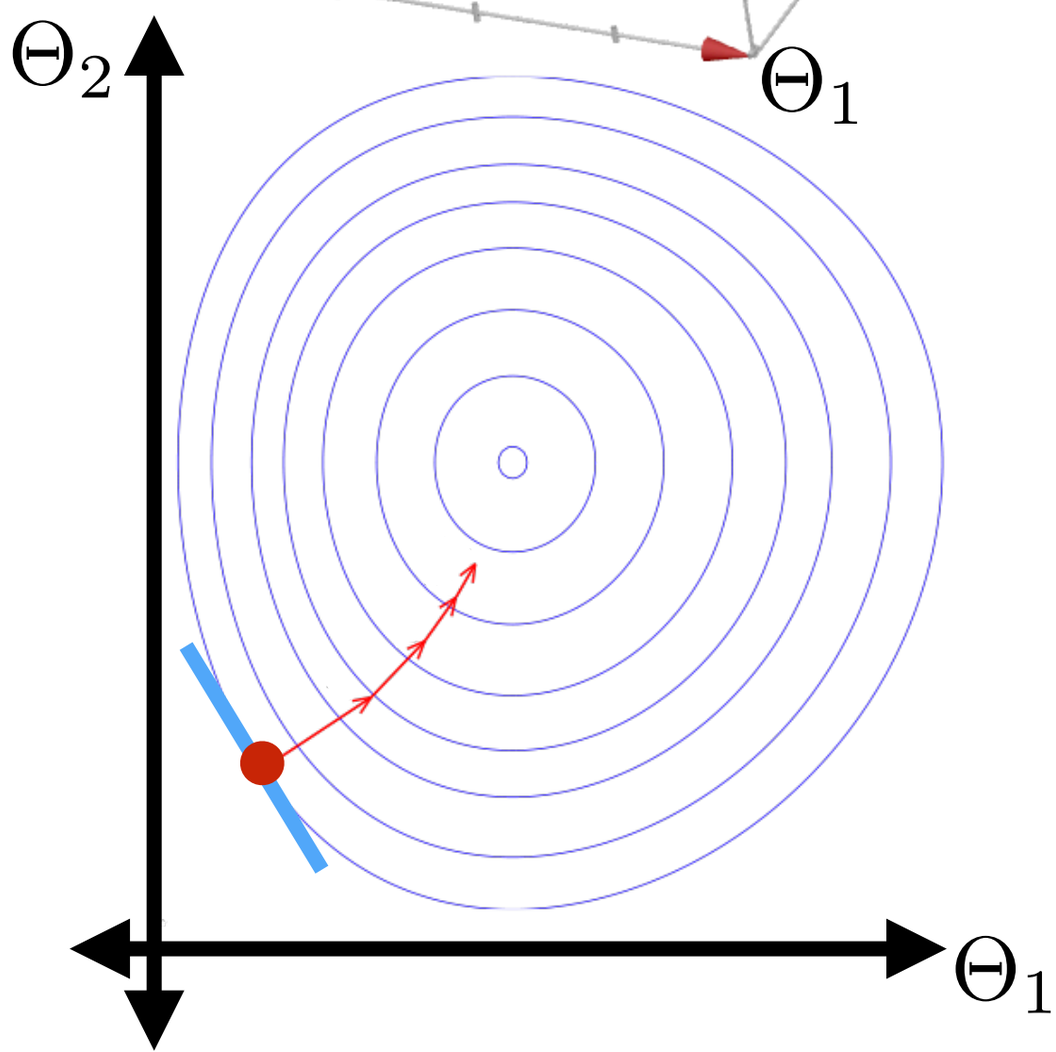
    $\Theta^{(t)} = \Theta^{(t-1)} - \eta \nabla_\Theta f(\Theta^{(t-1)})$

  until $\left| f(\Theta^{(t)}) - f(\Theta^{(t-1)}) \right| < \epsilon$

  Return $\Theta^{(t)}$

- Other possible stopping criteria:





Gradient descent

- Gradient $\nabla_\Theta f = \left[\frac{\partial f}{\partial \Theta_1}, \ldots, \frac{\partial f}{\partial \Theta_m}\right]^\top$
  - with $\Theta \in \mathbb{R}^m$

`Gradient-Descent` $(\Theta_{\text{init}}, \eta, f, \nabla_\Theta f, \epsilon)$

`Initialize` $\Theta^{(0)} = \Theta_{\text{init}}$

`Initialize t = 0`

`repeat`

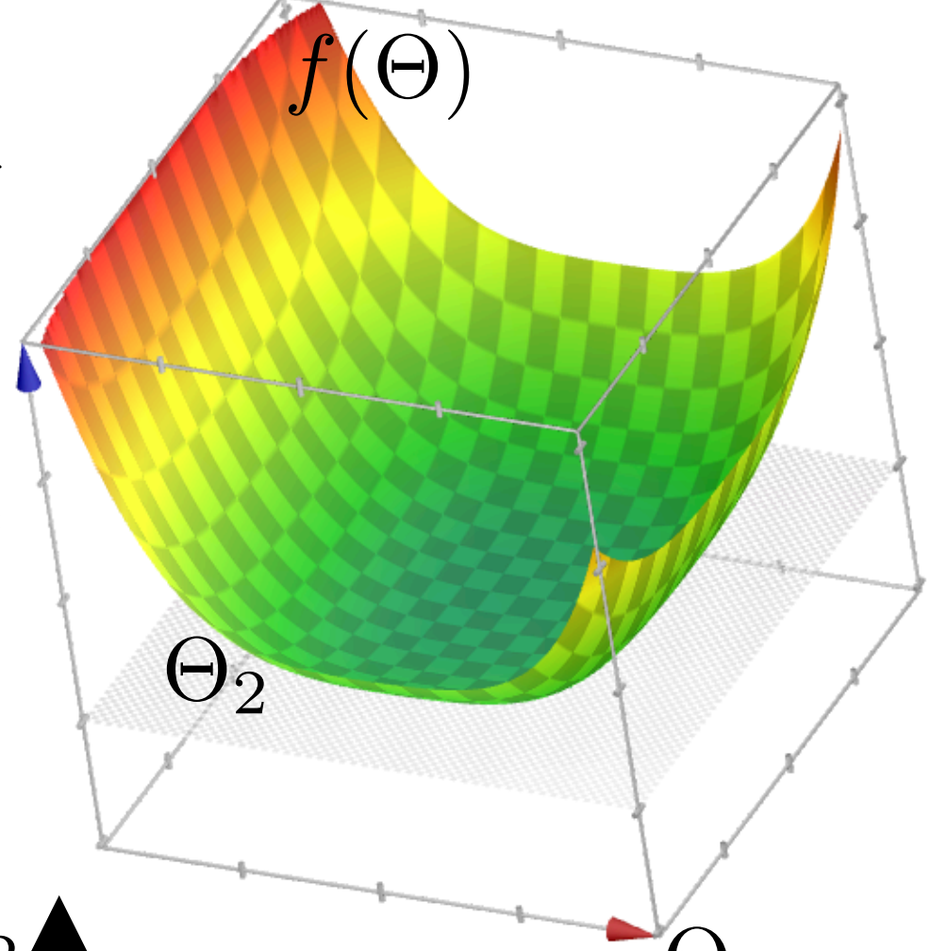
`t = t + 1`

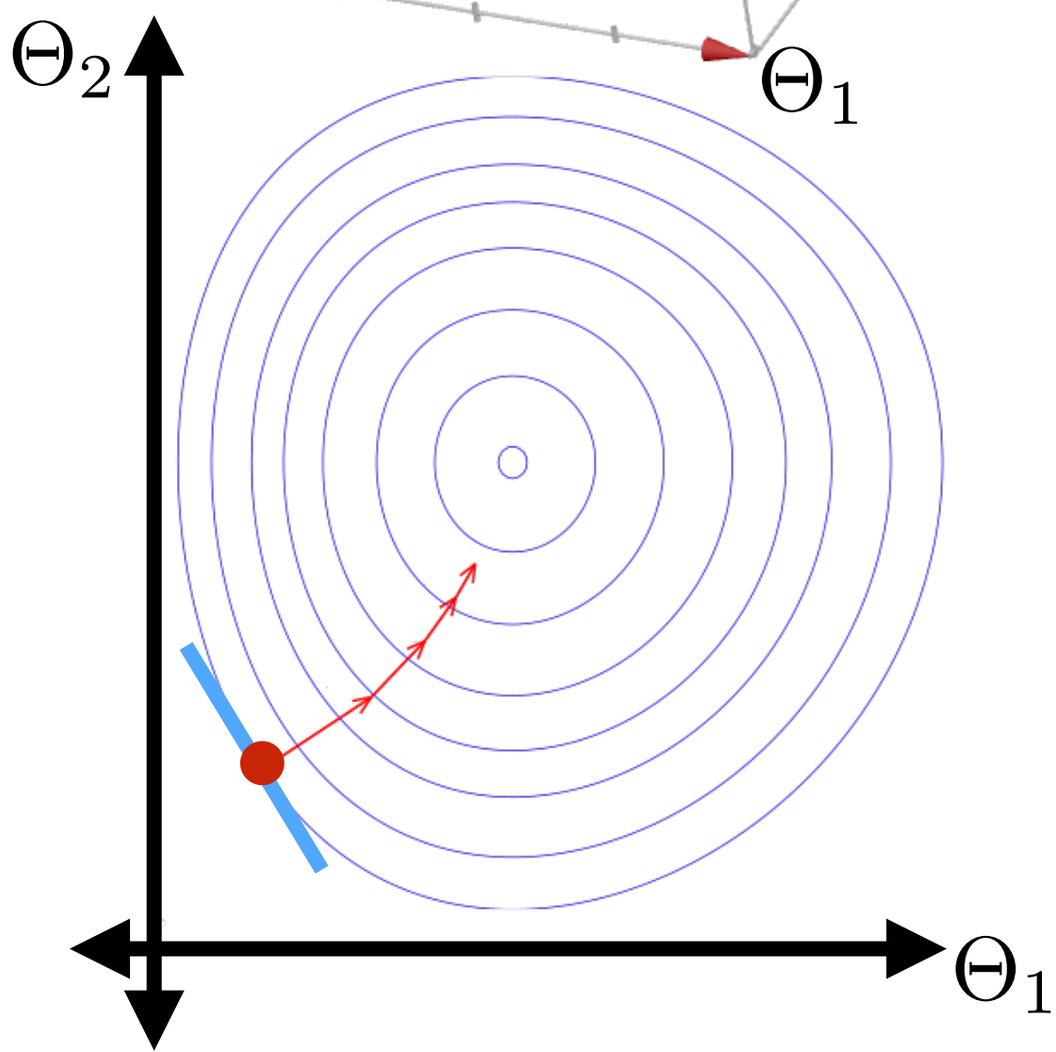
$$\Theta^{(t)} = \Theta^{(t-1)} - \eta \nabla_\Theta f(\Theta^{(t-1)})$$

`until` $\left| f(\Theta^{(t)}) - f(\Theta^{(t-1)}) \right| < \epsilon$

`Return` $\Theta^{(t)}$

- Other possible stopping criteria:
  - Max number of iterations *T*





Gradient descent

- Gradient $\nabla_\Theta f = \left[\frac{\partial f}{\partial \Theta_1}, \ldots, \frac{\partial f}{\partial \Theta_m}\right]^\top$
  - with $\Theta \in \mathbb{R}^m$

`Gradient-Descent` $(\Theta_{\text{init}}, \eta, f, \nabla_\Theta f, \epsilon)$

`Initialize` $\Theta^{(0)} = \Theta_{\text{init}}$

`Initialize t = 0`

`repeat`

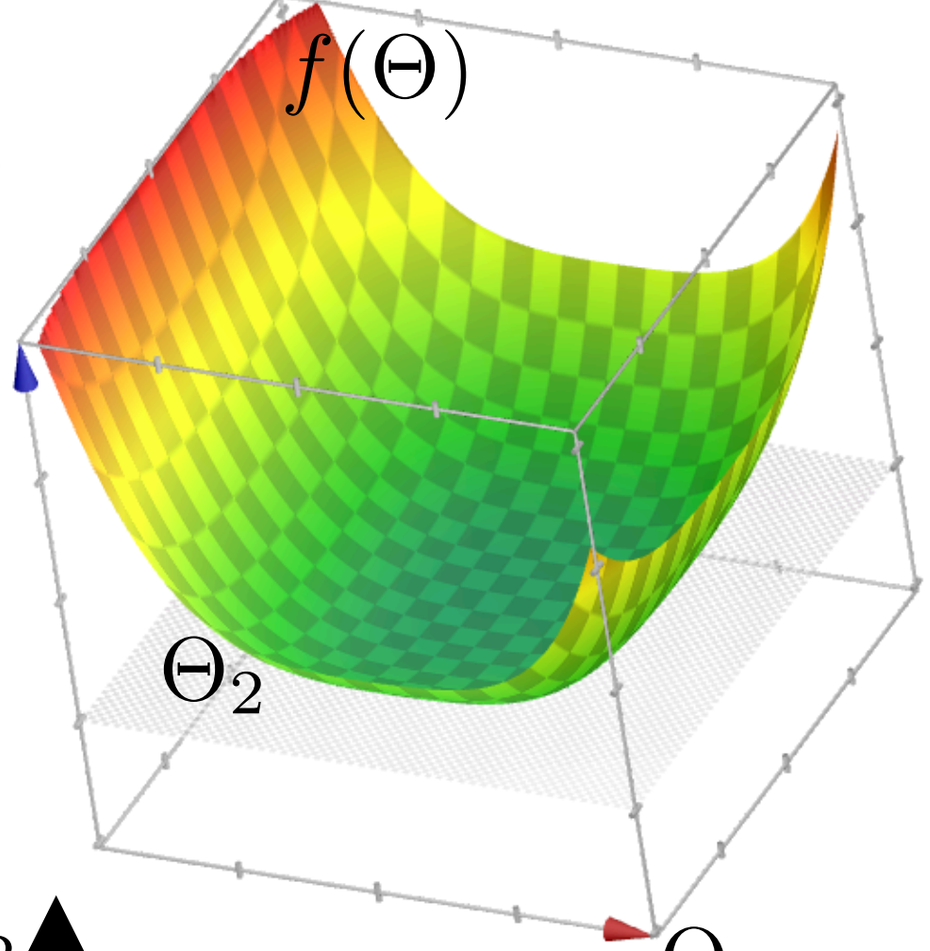
`t = t + 1`

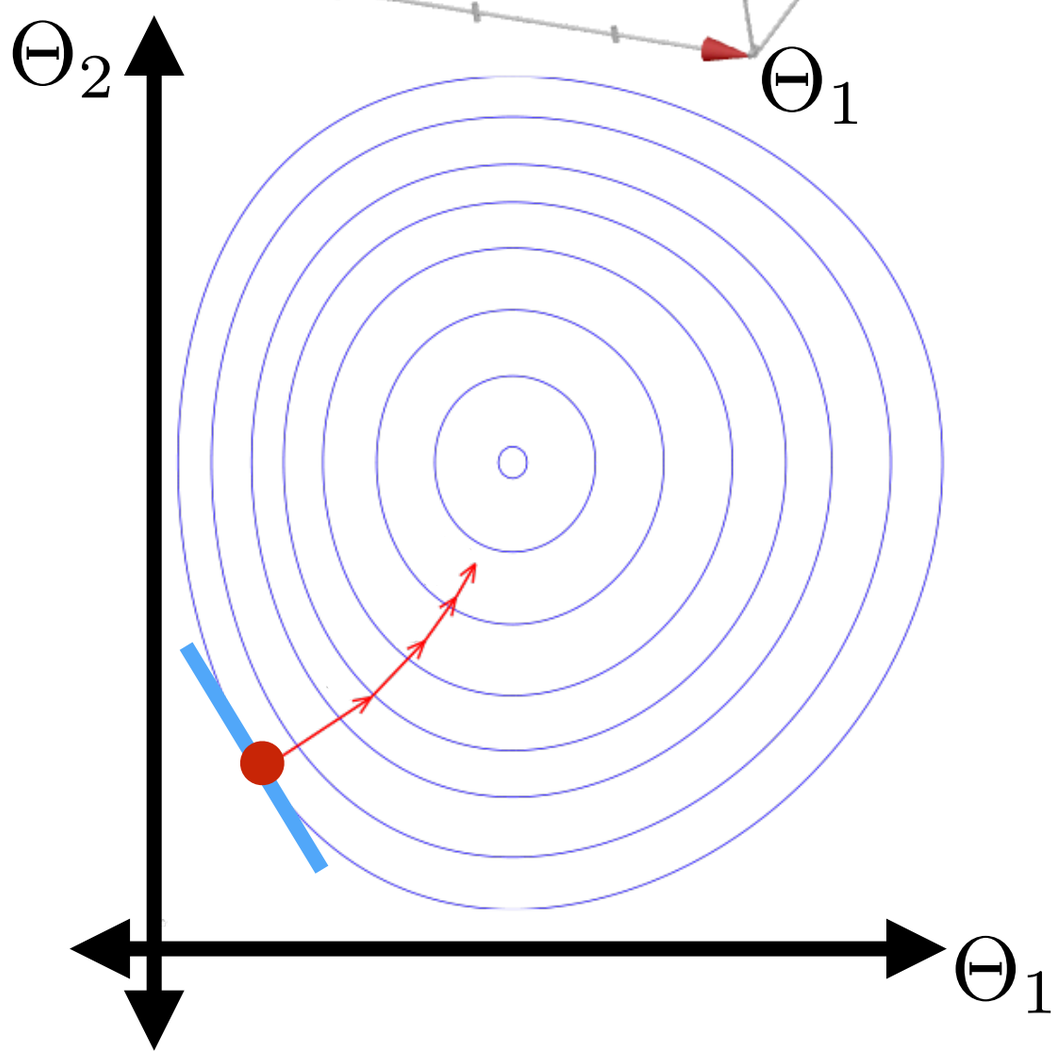
$$\Theta^{(t)} = \Theta^{(t-1)} - \eta \nabla_\Theta f(\Theta^{(t-1)})$$

`until` $\left| f(\Theta^{(t)}) - f(\Theta^{(t-1)}) \right| < \epsilon$

`Return` $\Theta^{(t)}$

- Other possible stopping criteria:
  - Max number of iterations *T*
  - $|\Theta^{(t)} - \Theta^{(t-1)}| < \epsilon$





Gradient descent

- Gradient $\nabla_\Theta f = \left[\frac{\partial f}{\partial \Theta_1}, \ldots, \frac{\partial f}{\partial \Theta_m}\right]^\top$
  - with $\Theta \in \mathbb{R}^m$

`Gradient-Descent` $(\Theta_{\text{init}}, \eta, f, \nabla_\Theta f, \epsilon)$

`Initialize` $\Theta^{(0)} = \Theta_{\text{init}}$

`Initialize t = 0`

`repeat`

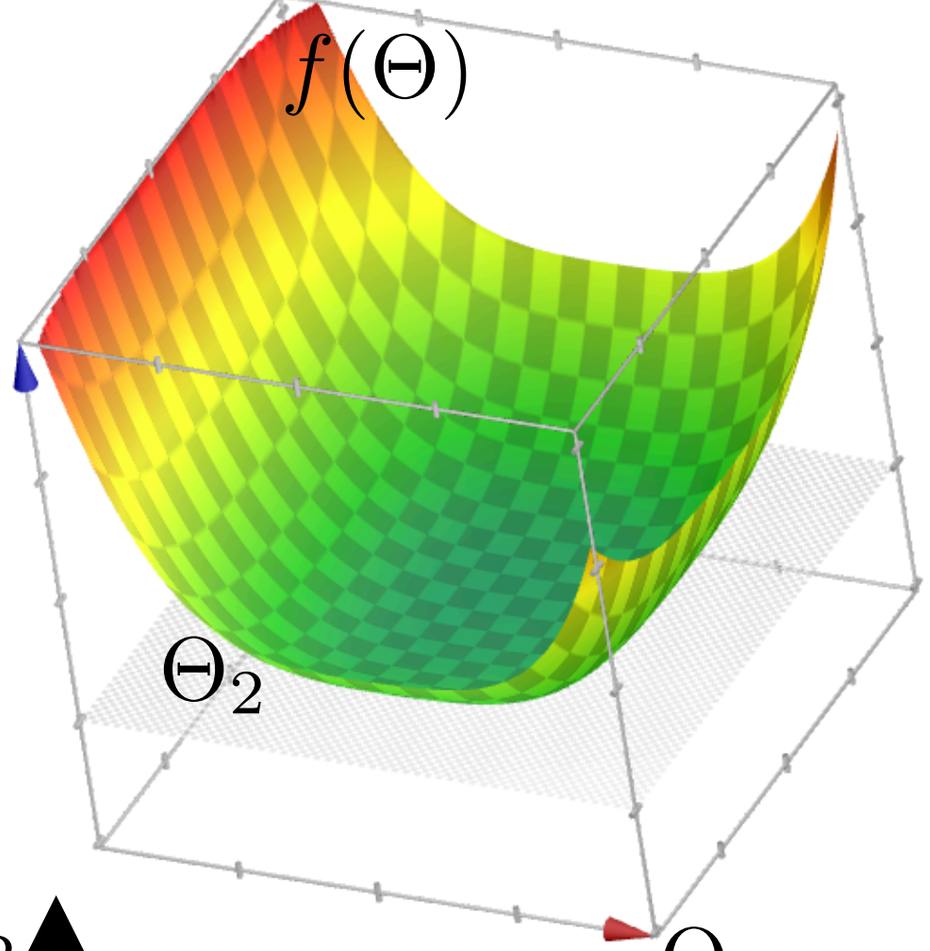
`t = t + 1`

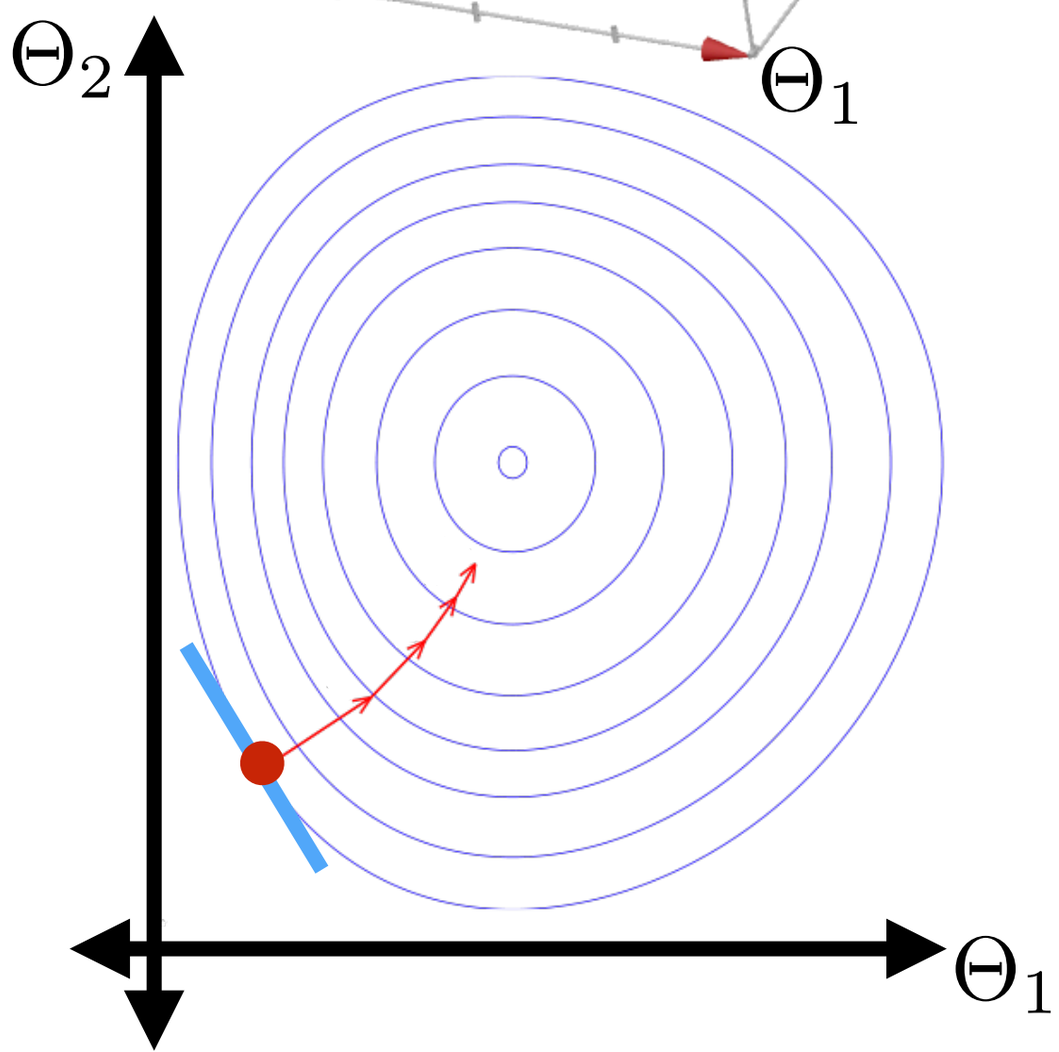
$$\Theta^{(t)} = \Theta^{(t-1)} - \eta \nabla_\Theta f(\Theta^{(t-1)})$$

`until` $\left| f(\Theta^{(t)}) - f(\Theta^{(t-1)}) \right| < \epsilon$

`Return` $\Theta^{(t)}$

- Other possible stopping criteria:
  - Max number of iterations *T*
  - $|\Theta^{(t)} - \Theta^{(t-1)}| < \epsilon$
  - $\|\nabla_\Theta f(\Theta^{(t)})\| < \epsilon$



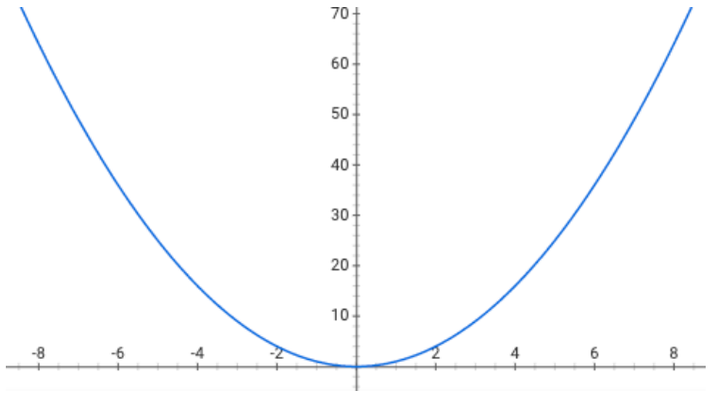


Gradient descent properties

- A function *f* on $\mathbb{R}^m$ is convex if any line segment connecting two points of the graph of *f* lies above or on the graph

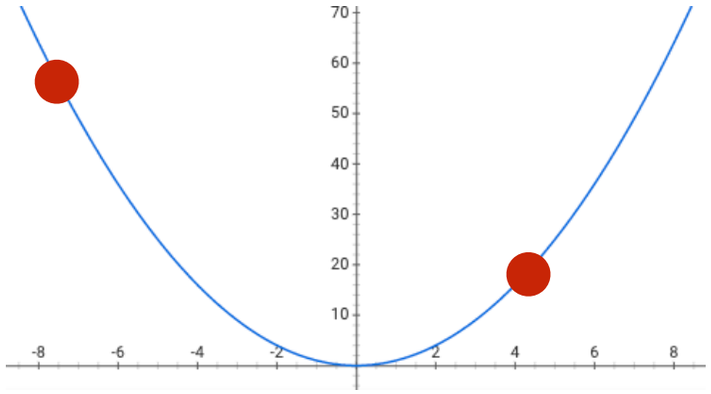
Gradient descent properties

- A function *f* on $\mathbb{R}^m$ is convex if any line segment connecting two points of the graph of *f* lies above or on the graph



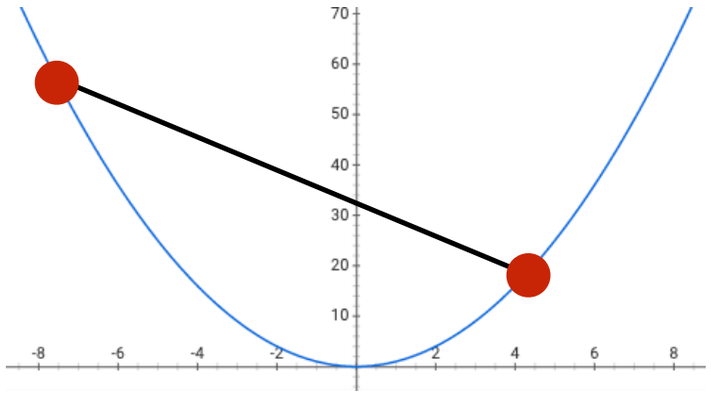
Gradient descent properties

- A function *f* on $\mathbb{R}^m$ is convex if any line segment connecting two points of the graph of *f* lies above or on the graph



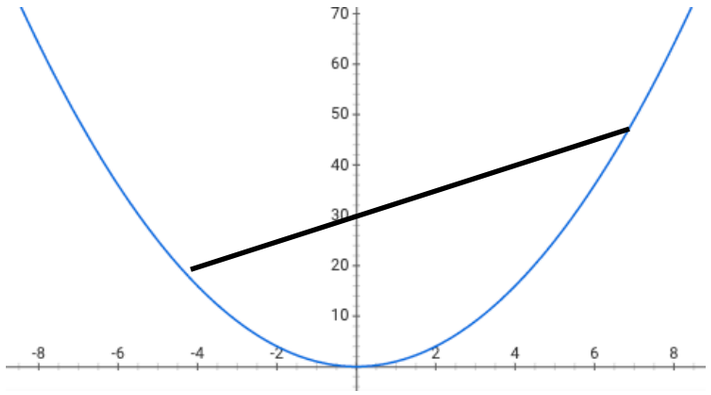
Gradient descent properties

- A function *f* on $\mathbb{R}^m$ is convex if any line segment connecting two points of the graph of *f* lies above or on the graph



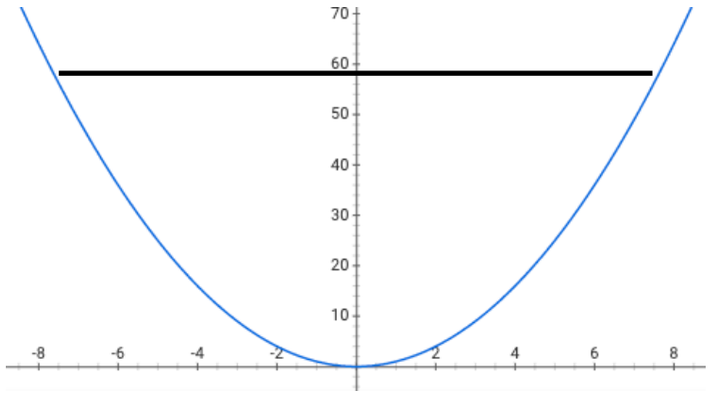
Gradient descent properties

- A function *f* on $\mathbb{R}^m$ is convex if any line segment connecting two points of the graph of *f* lies above or on the graph



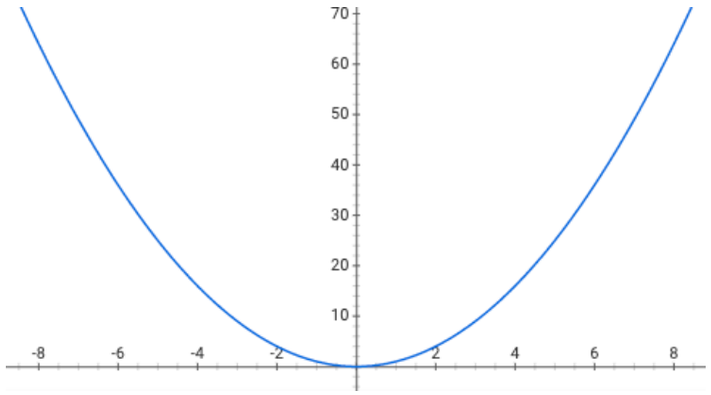
Gradient descent properties

- A function *f* on $\mathbb{R}^m$ is convex if any line segment connecting two points of the graph of *f* lies above or on the graph



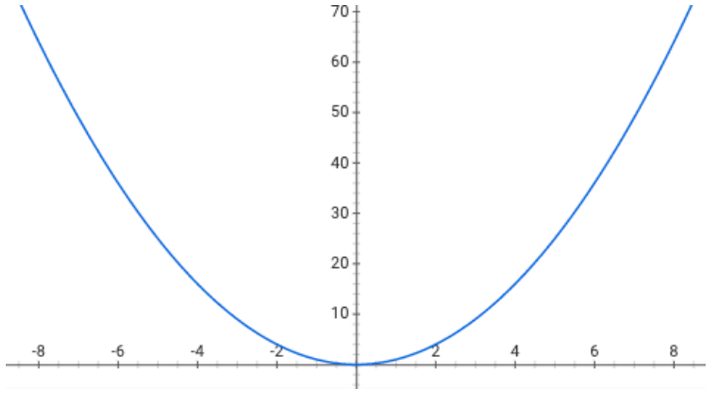
Gradient descent properties

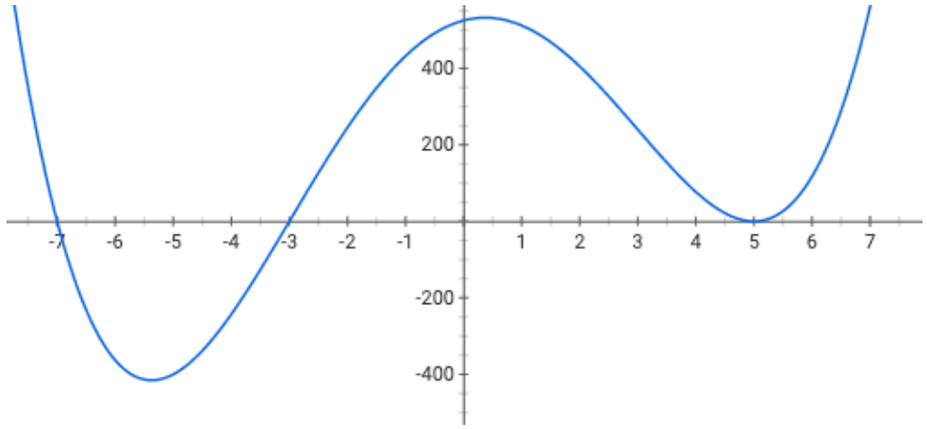
- A function *f* on $\mathbb{R}^m$ is convex if any line segment connecting two points of the graph of *f* lies above or on the graph



Gradient descent properties

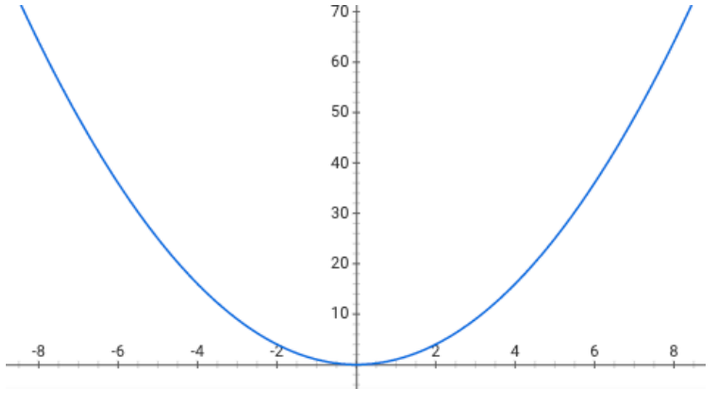
- A function *f* on $\mathbb{R}^m$ is convex if any line segment connecting two points of the graph of *f* lies above or on the graph

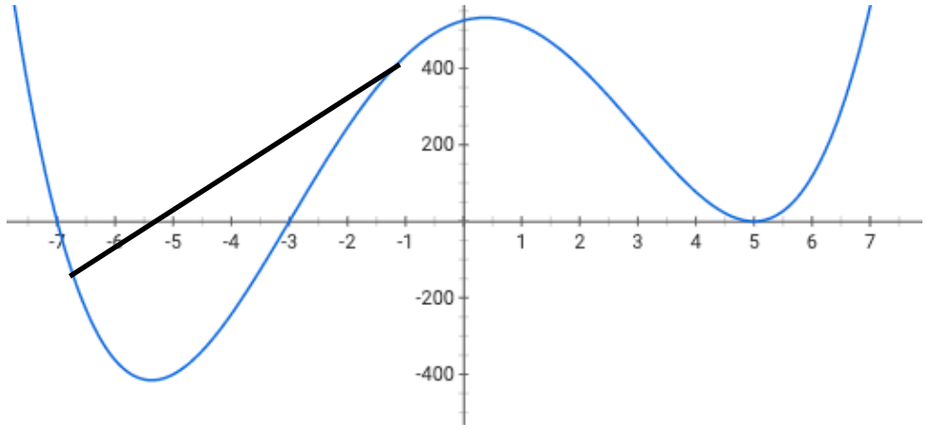




Gradient descent properties

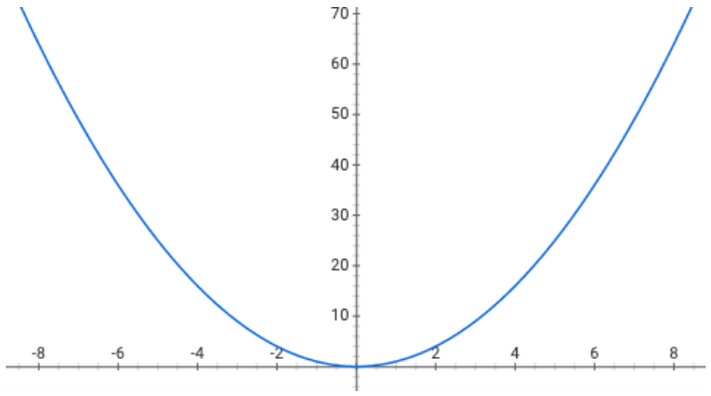
- A function *f* on $\mathbb{R}^m$ is convex if any line segment connecting two points of the graph of *f* lies above or on the graph

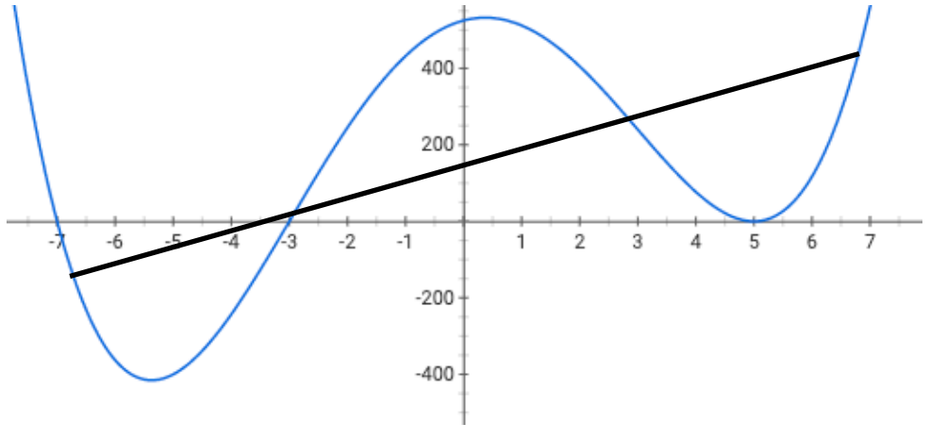




Gradient descent properties

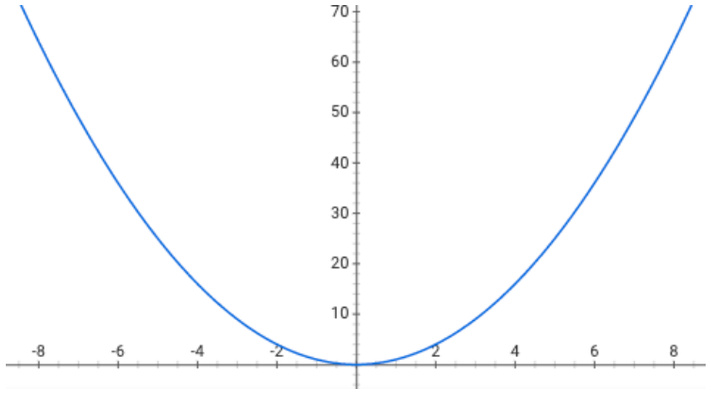
- A function *f* on $\mathbb{R}^m$ is convex if any line segment connecting two points of the graph of *f* lies above or on the graph

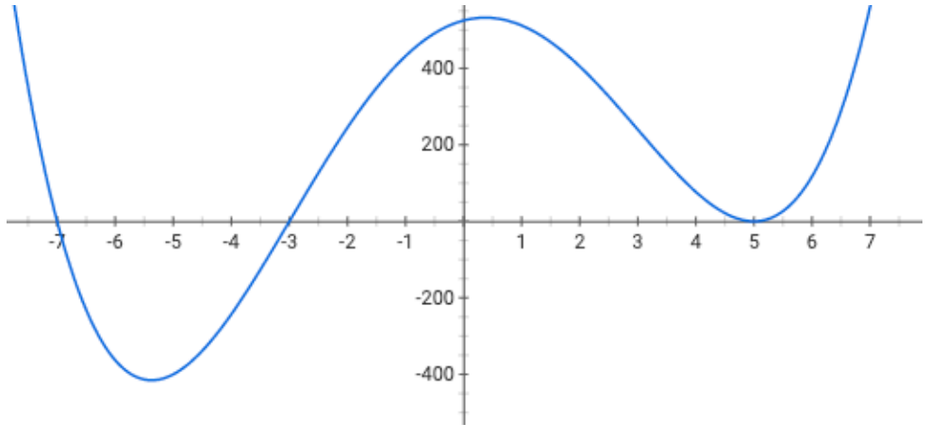




Gradient descent properties

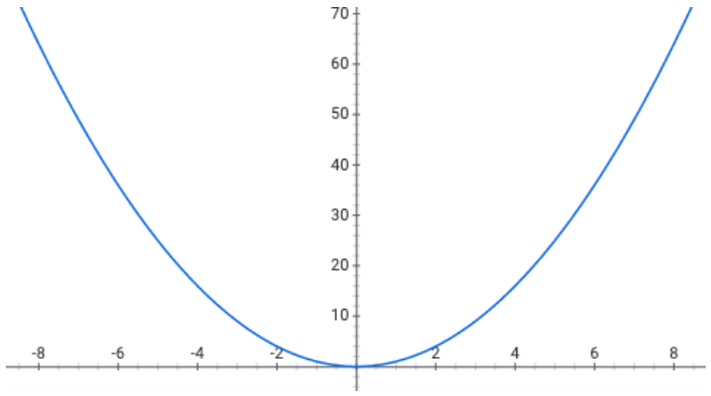
- A function *f* on $\mathbb{R}^m$ is convex if any line segment connecting two points of the graph of *f* lies above or on the graph

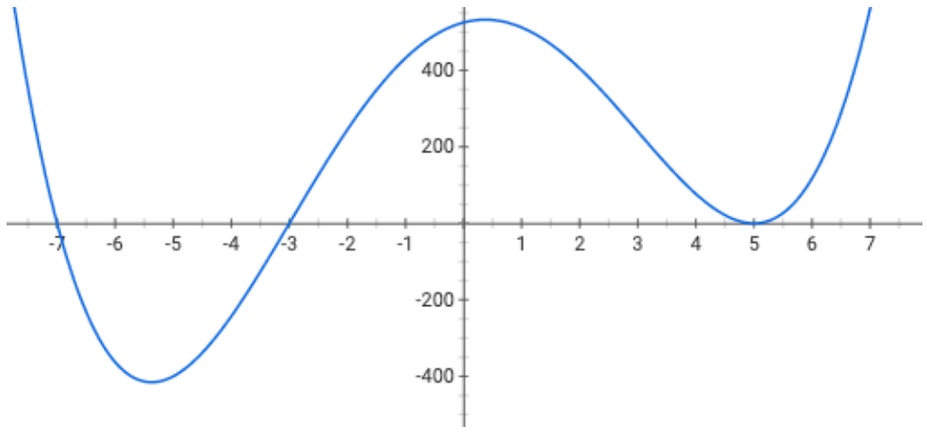


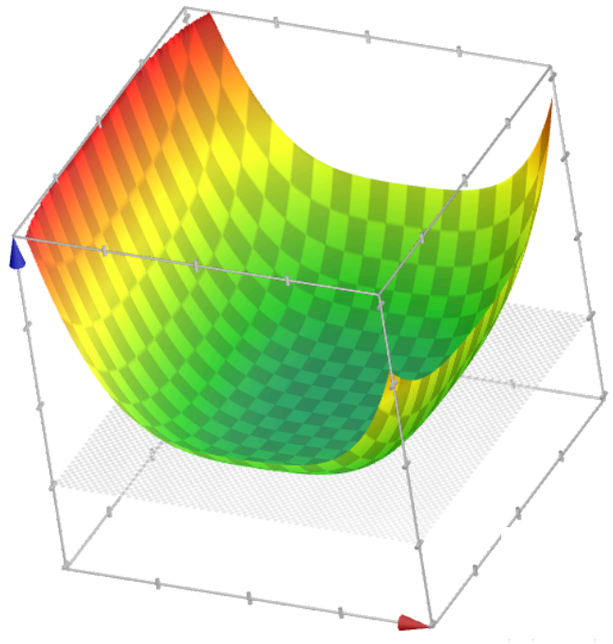


Gradient descent properties

- A function *f* on $\mathbb{R}^m$ is convex if any line segment connecting two points of the graph of *f* lies above or on the graph

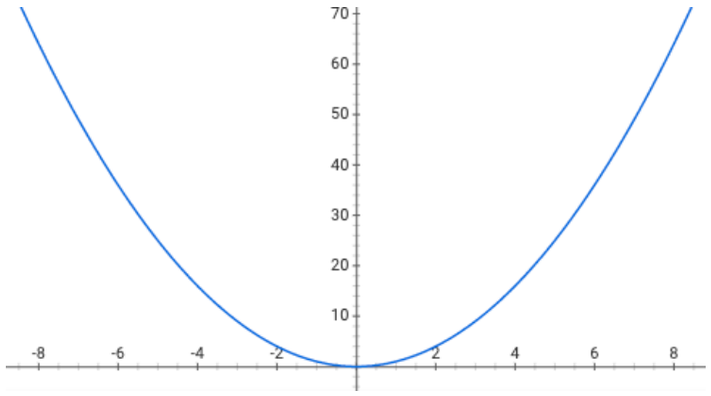


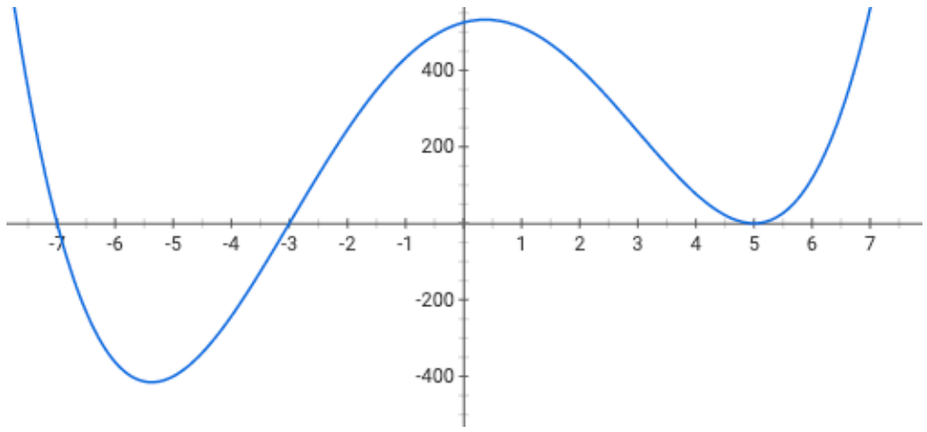


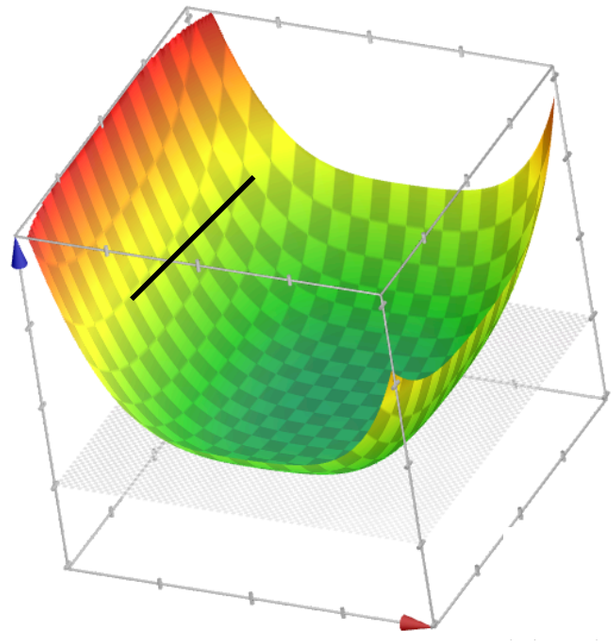


Gradient descent properties

- A function *f* on $\mathbb{R}^m$ is convex if any line segment connecting two points of the graph of *f* lies above or on the graph

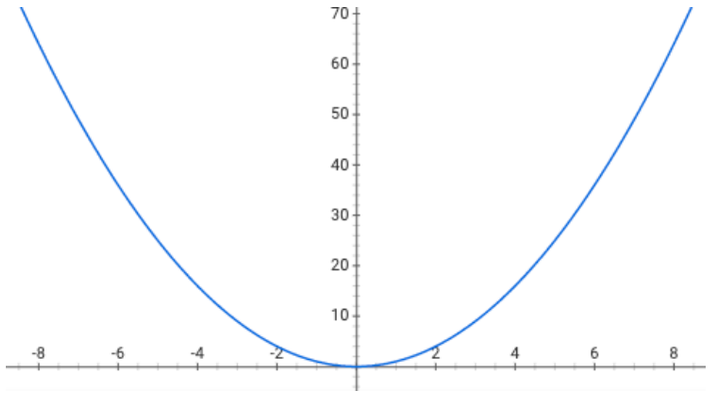


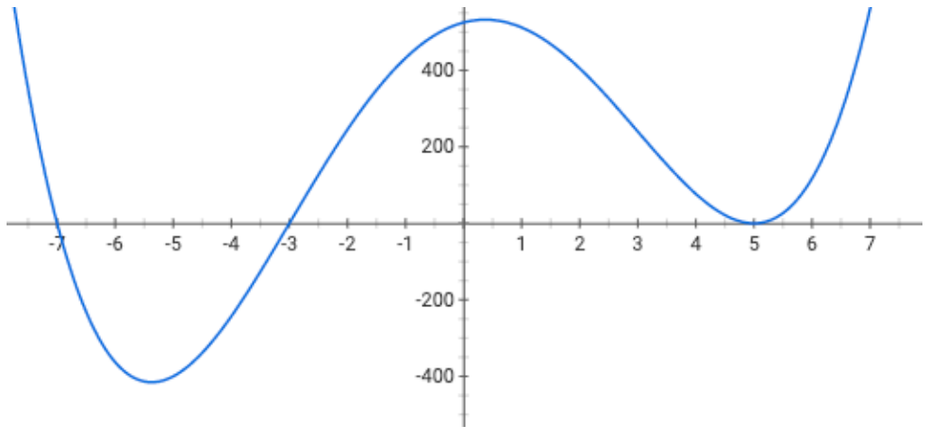


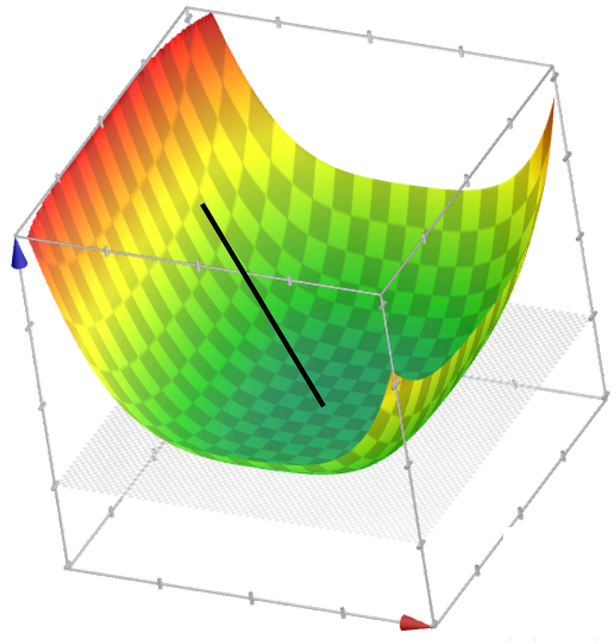


Gradient descent properties

- A function *f* on $\mathbb{R}^m$ is convex if any line segment connecting two points of the graph of *f* lies above or on the graph

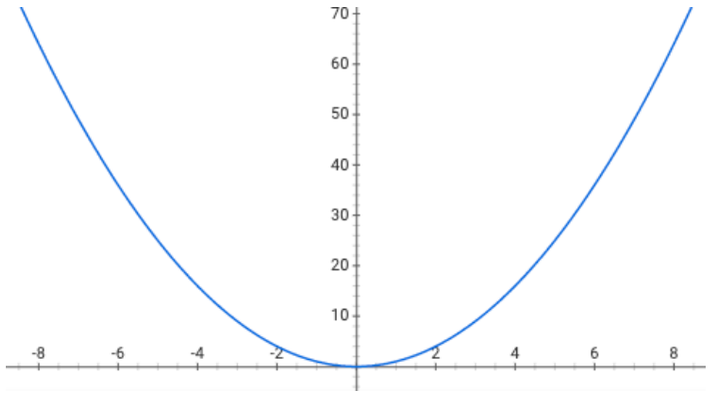


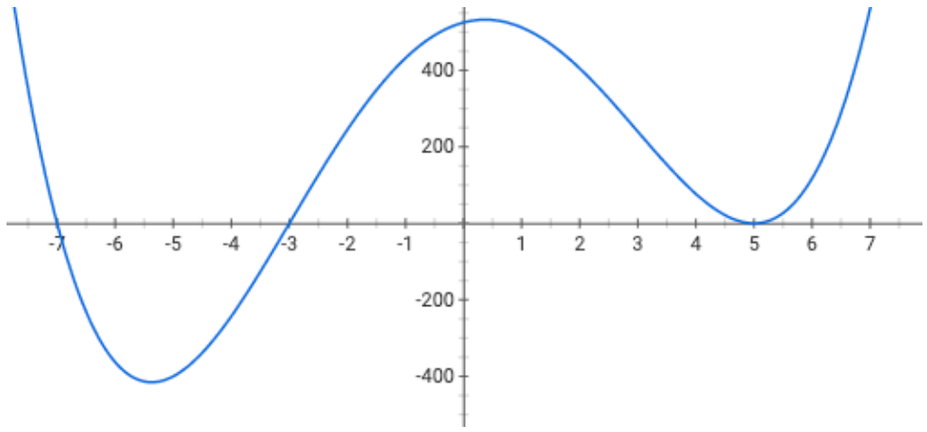


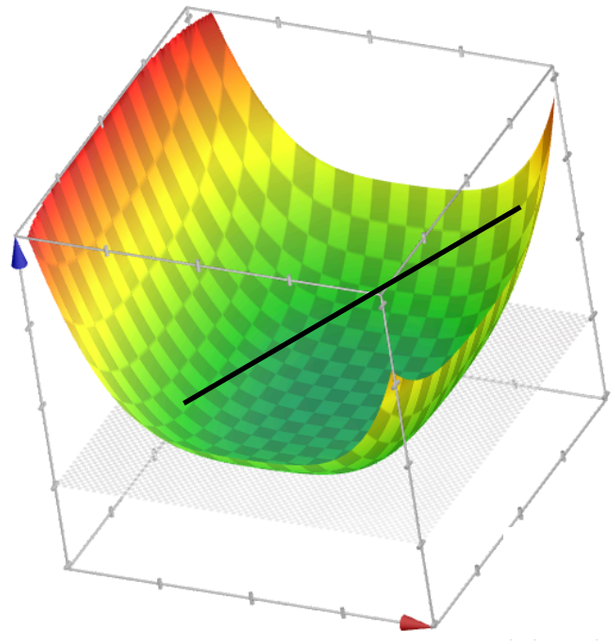


Gradient descent properties

- A function *f* on $\mathbb{R}^m$ is convex if any line segment connecting two points of the graph of *f* lies above or on the graph

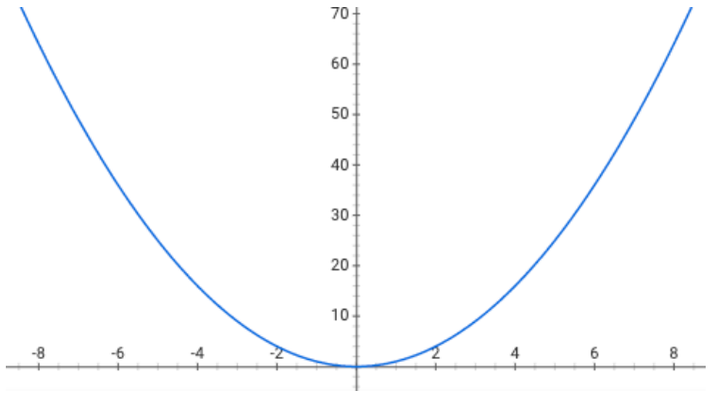


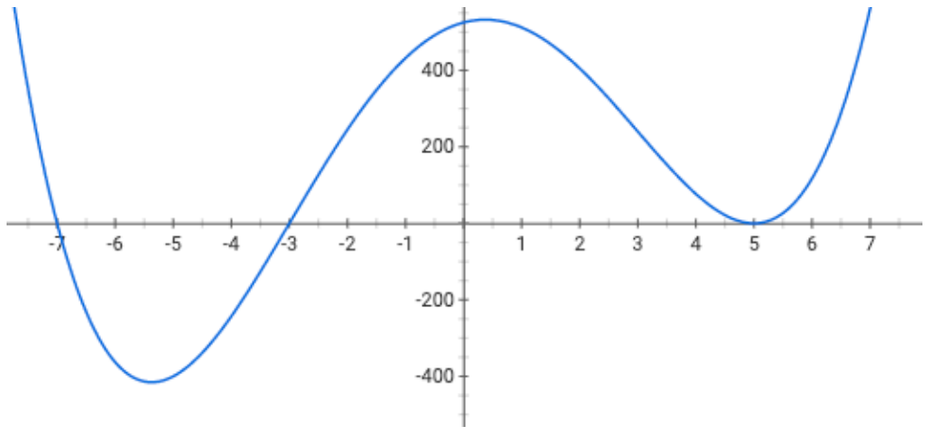


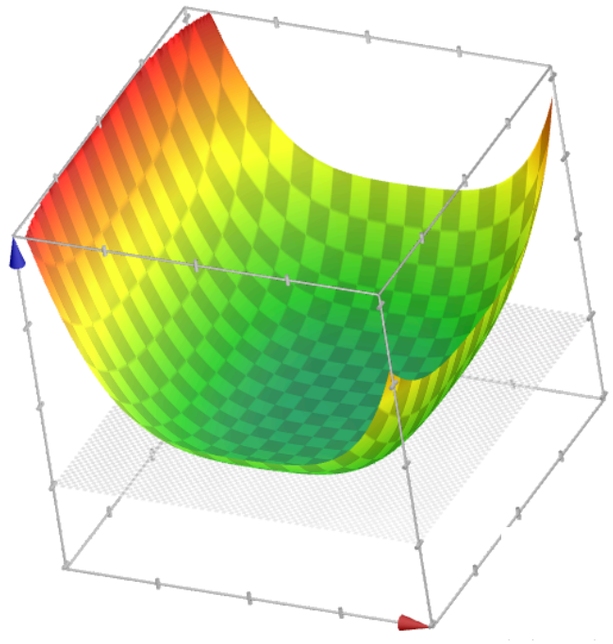


Gradient descent properties

- A function *f* on $\mathbb{R}^m$ is convex if any line segment connecting two points of the graph of *f* lies above or on the graph

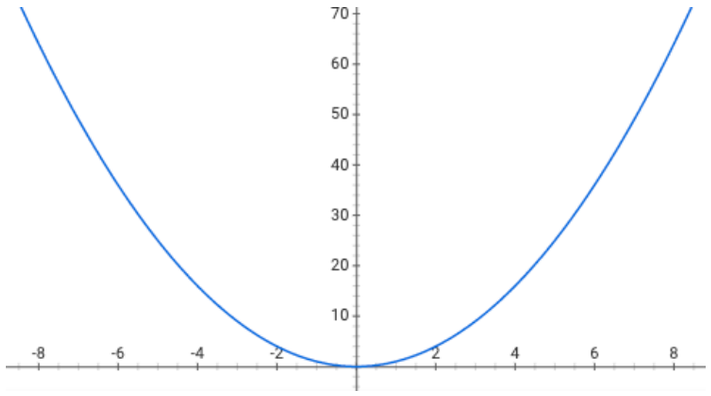


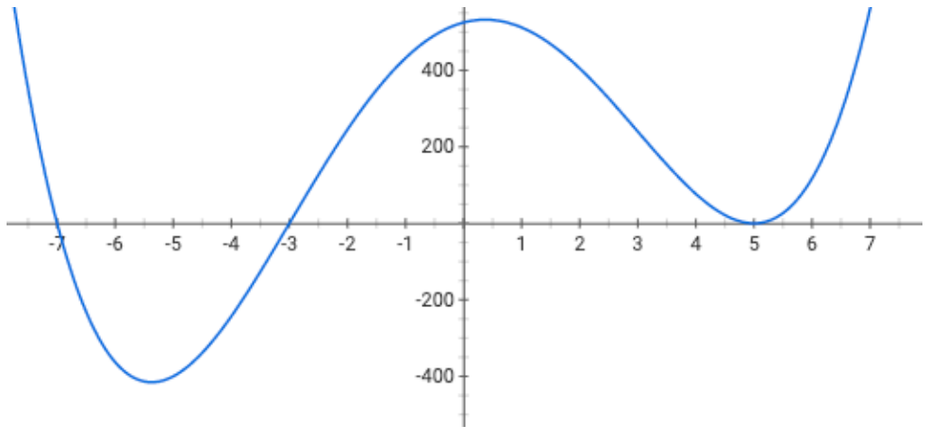


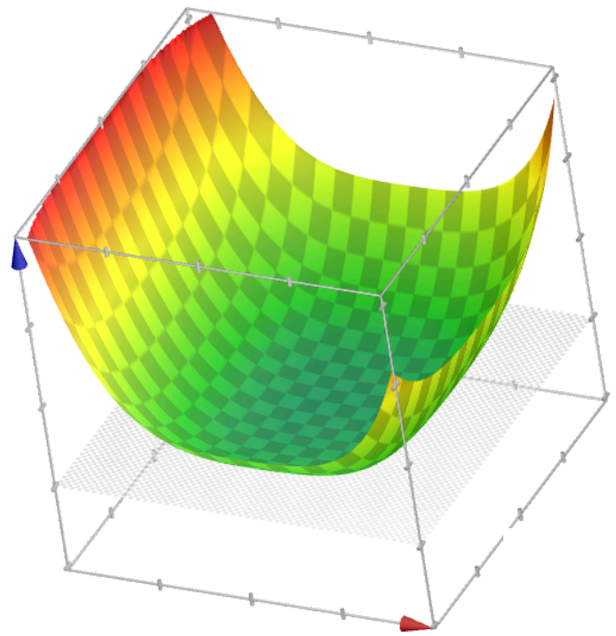


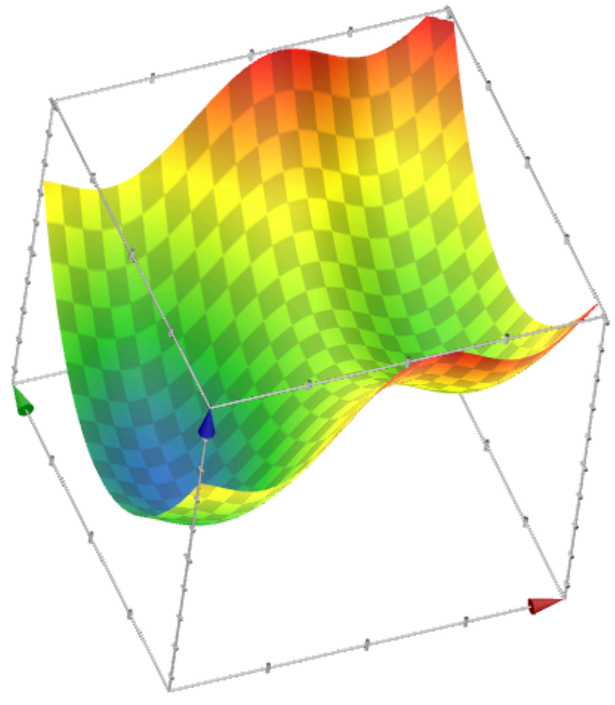
Gradient descent properties

- A function *f* on $\mathbb{R}^m$ is convex if any line segment connecting two points of the graph of *f* lies above or on the graph



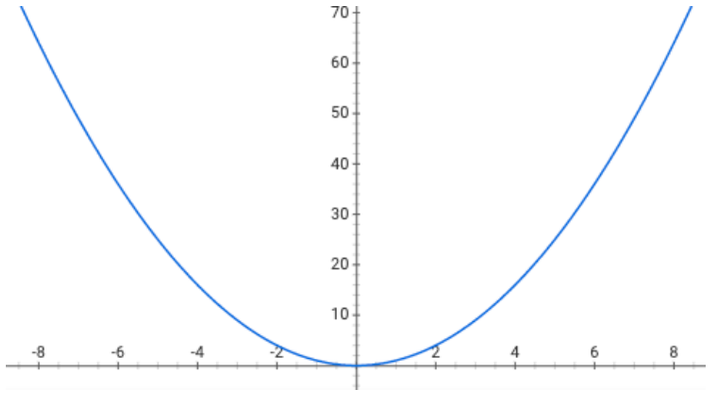


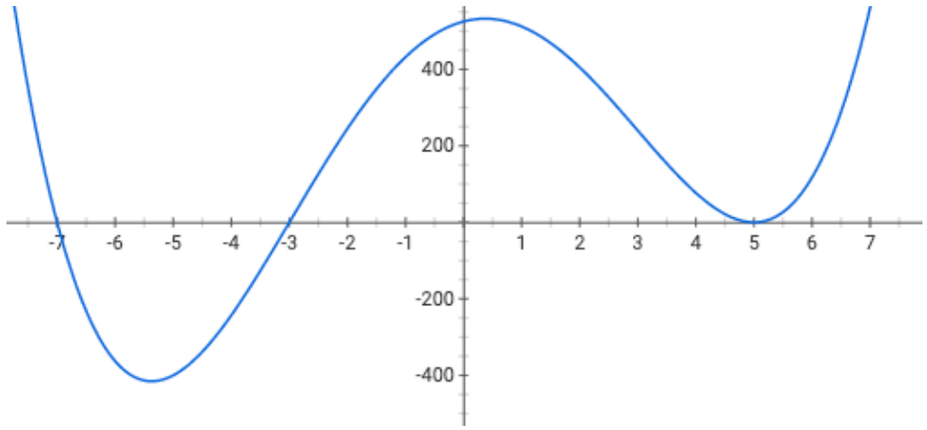


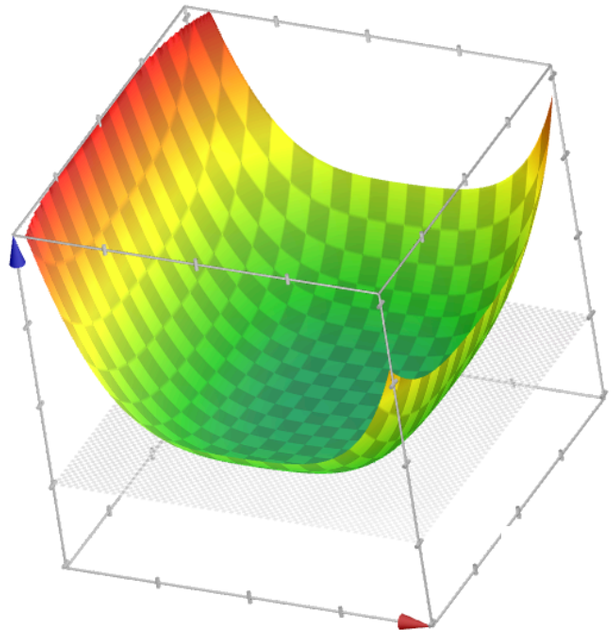


Gradient descent properties

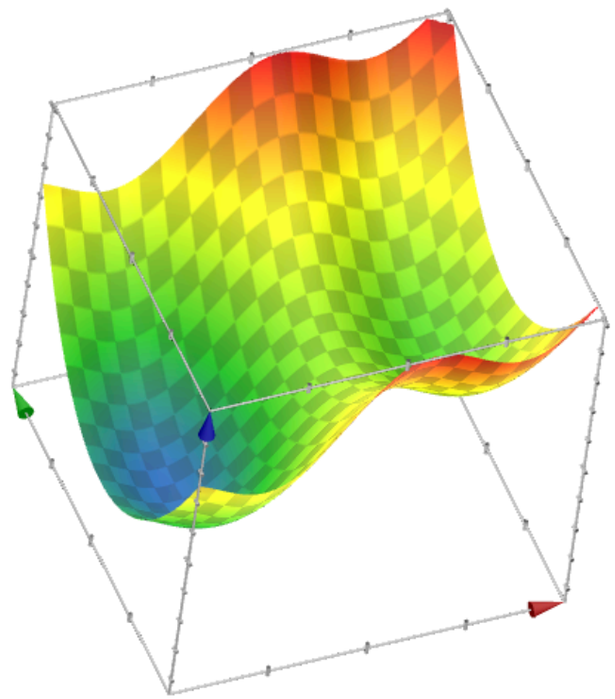
- A function *f* on $\mathbb{R}^m$ is convex if any line segment connecting two points of the graph of *f* lies above or on the graph





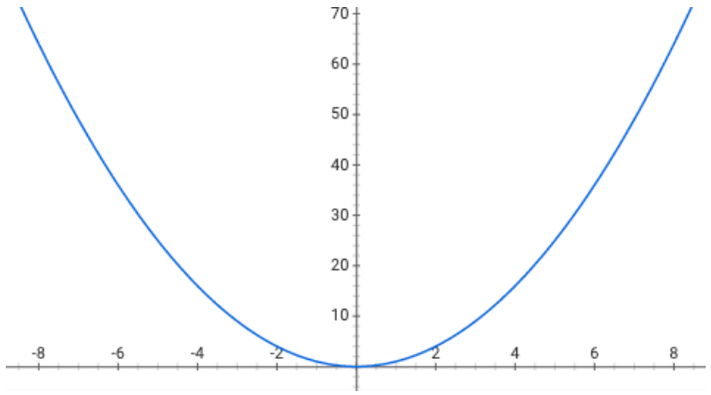


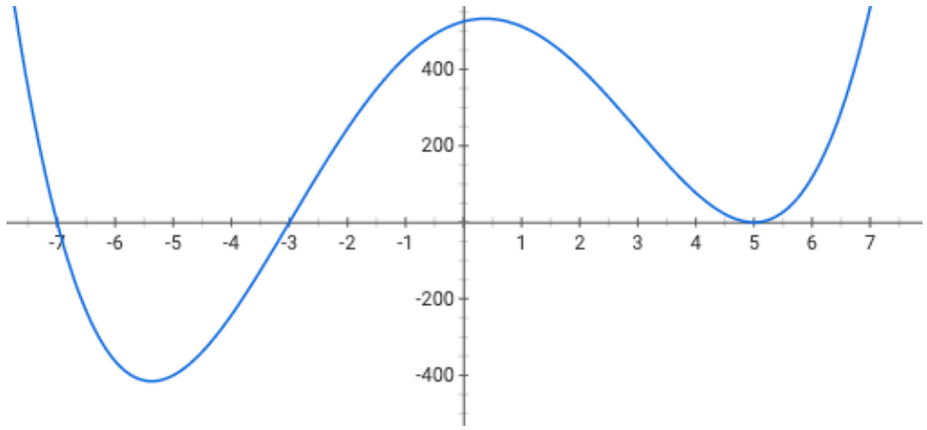
- **Theorem**: Gradient descent performance

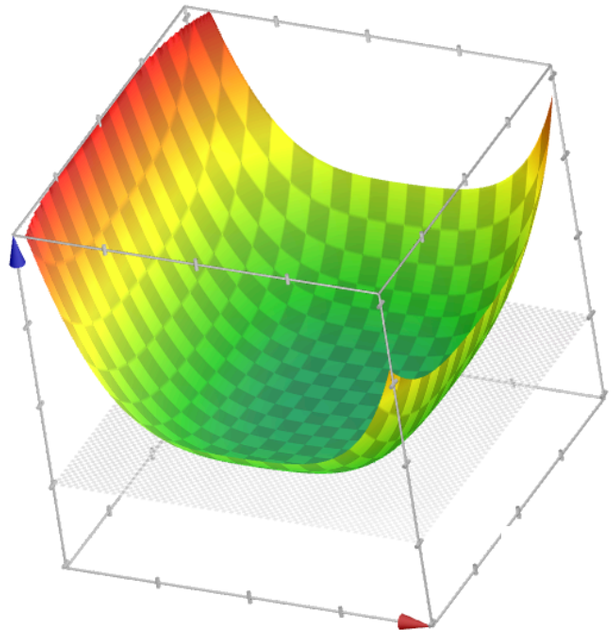


Gradient descent properties

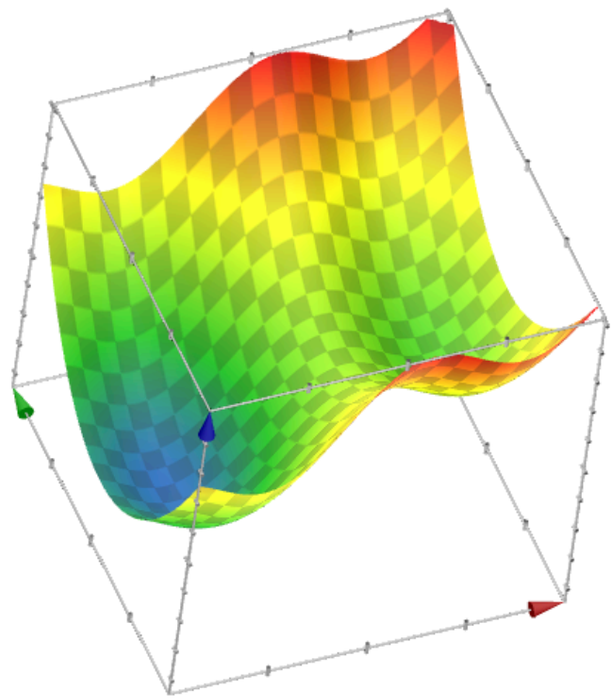
- A function *f* on $\mathbb{R}^m$ is convex if any line segment connecting two points of the graph of *f* lies above or on the graph





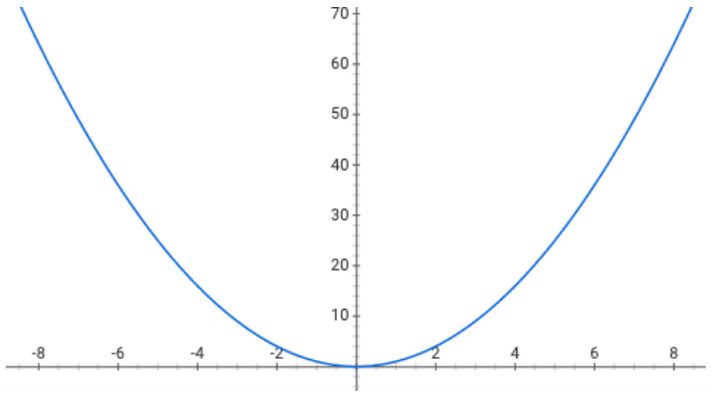


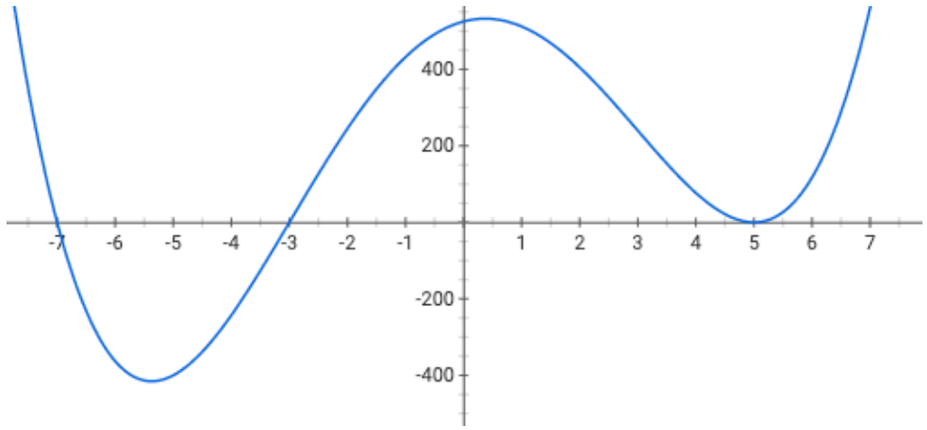
- **Theorem**: Gradient descent performance
  - **Assumptions**:

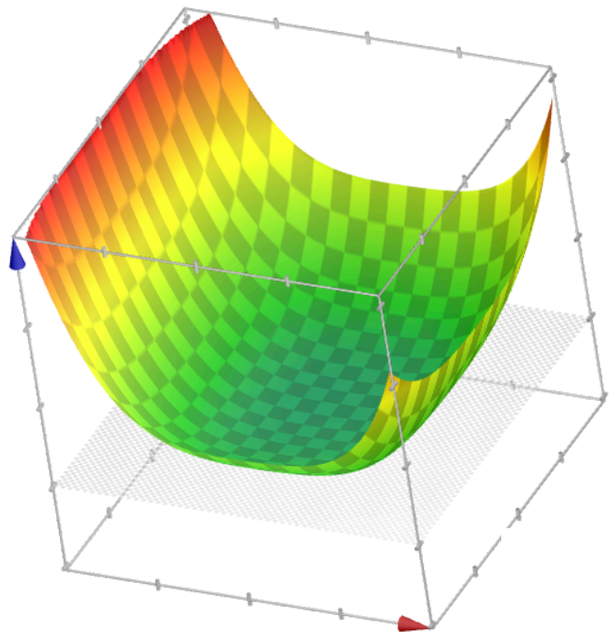


Gradient descent properties

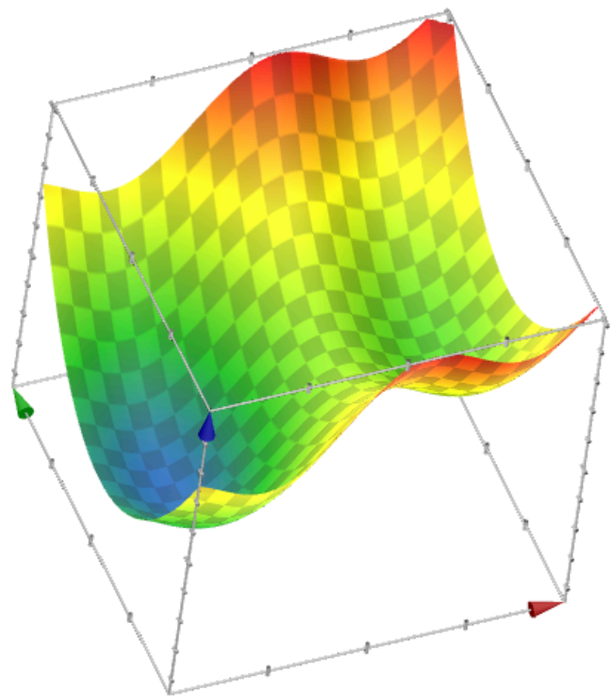
- A function *f* on $\mathbb{R}^m$ is convex if any line segment connecting two points of the graph of *f* lies above or on the graph





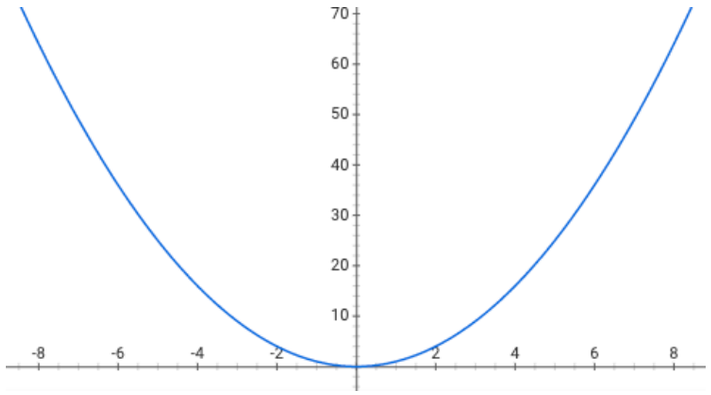


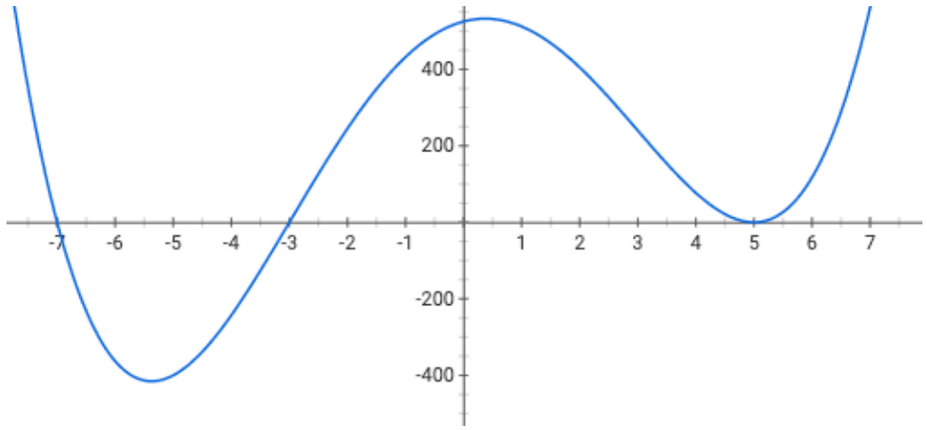
- **Theorem**: Gradient descent performance
  - **Assumptions**: (Choose any $\tilde{\epsilon} > 0$)

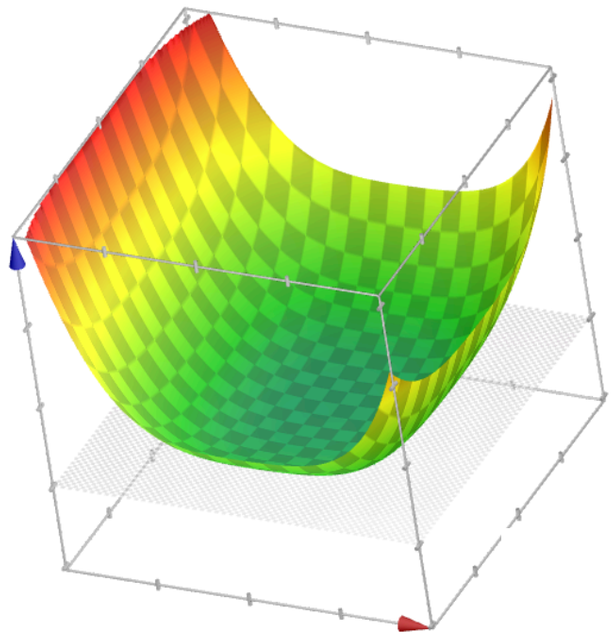


Gradient descent properties

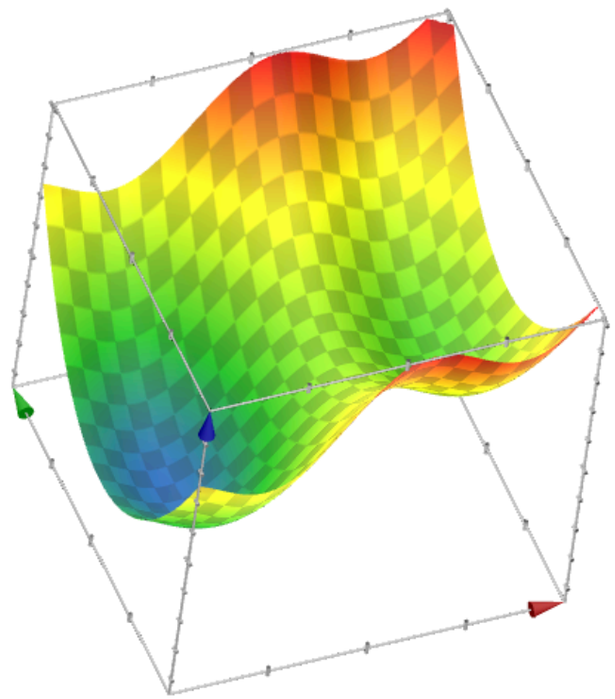
- A function *f* on $\mathbb{R}^m$ is convex if any line segment connecting two points of the graph of *f* lies above or on the graph





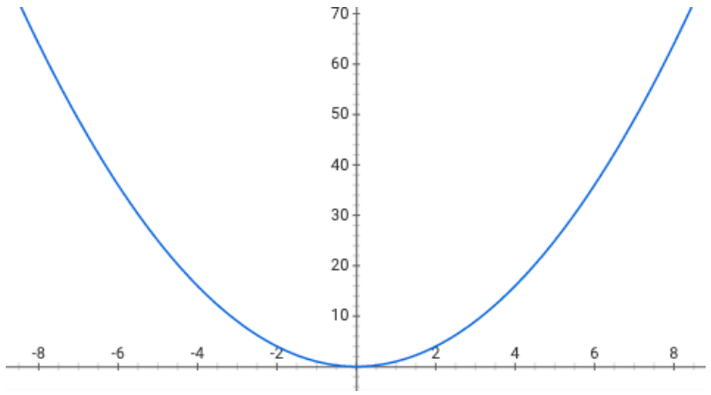


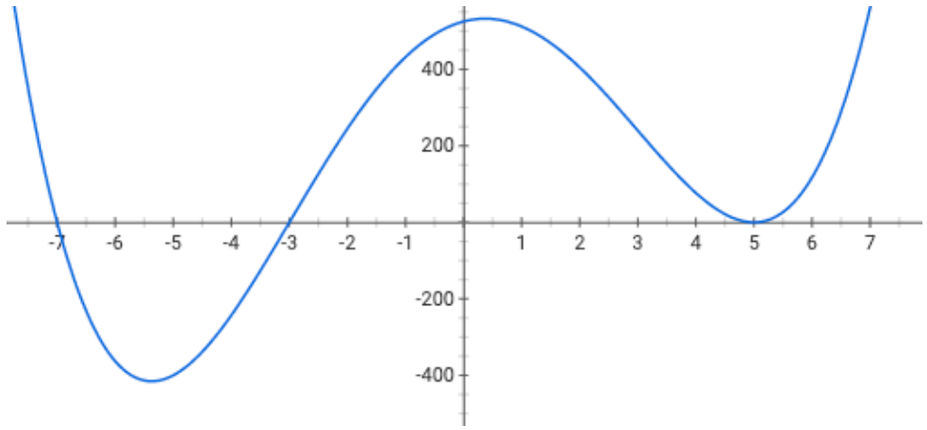
- **Theorem**: Gradient descent performance
  - **Assumptions**: (Choose any $\tilde{\epsilon} > 0$)
    - *f* is sufficiently "smooth" and convex

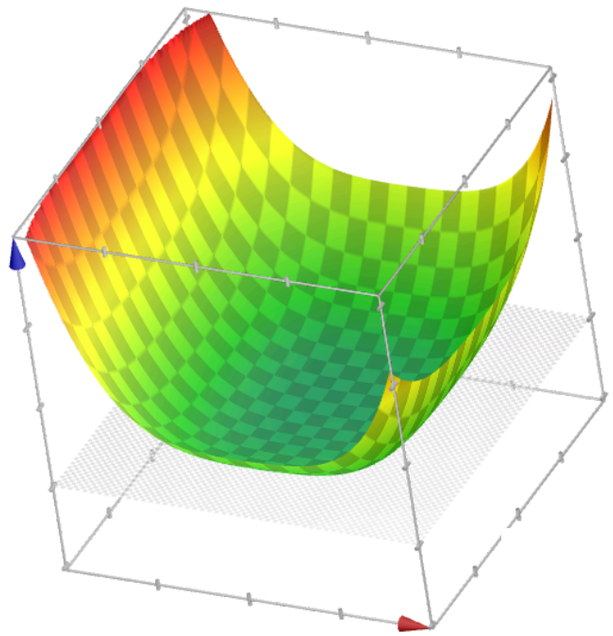


Gradient descent properties

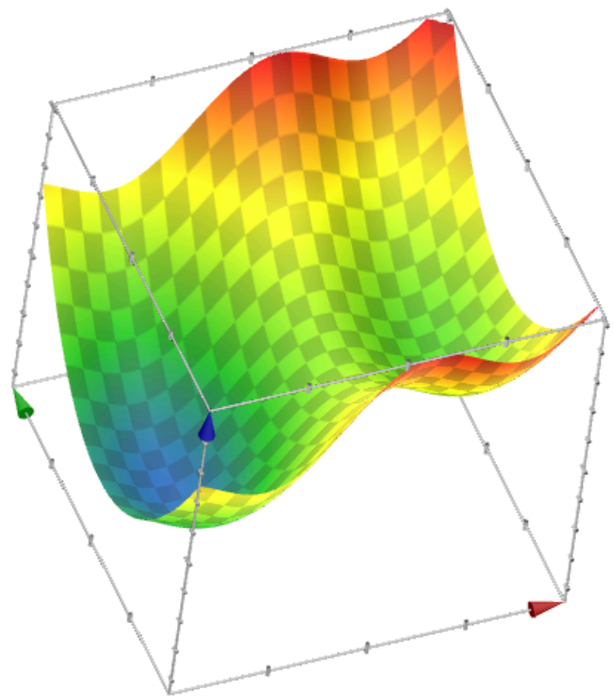
- A function *f* on $\mathbb{R}^m$ is convex if any line segment connecting two points of the graph of *f* lies above or on the graph





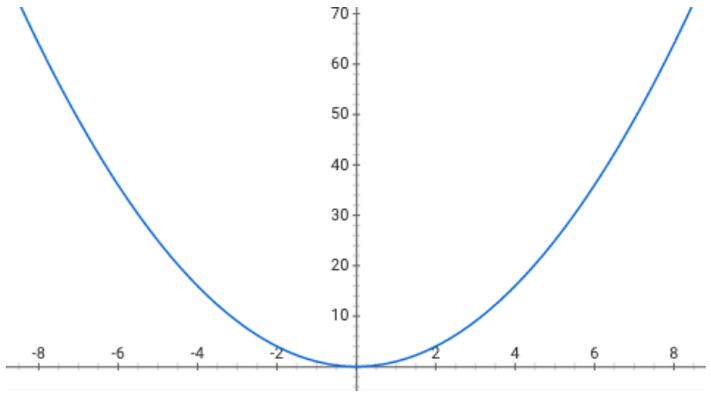


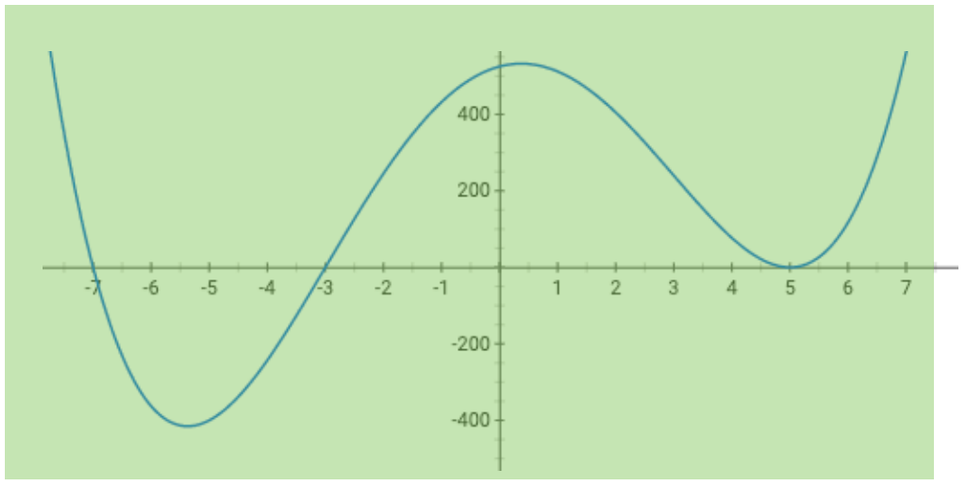
- **Theorem**: Gradient descent performance
  - **Assumptions**: (Choose any $\tilde{\epsilon} > 0$)
    - *f* is sufficiently "smooth" and convex
    - *f* has at least one global optimum

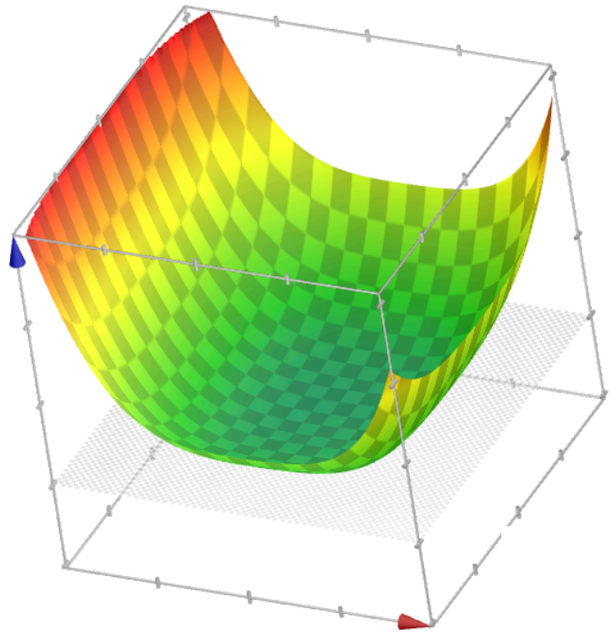


Gradient descent properties

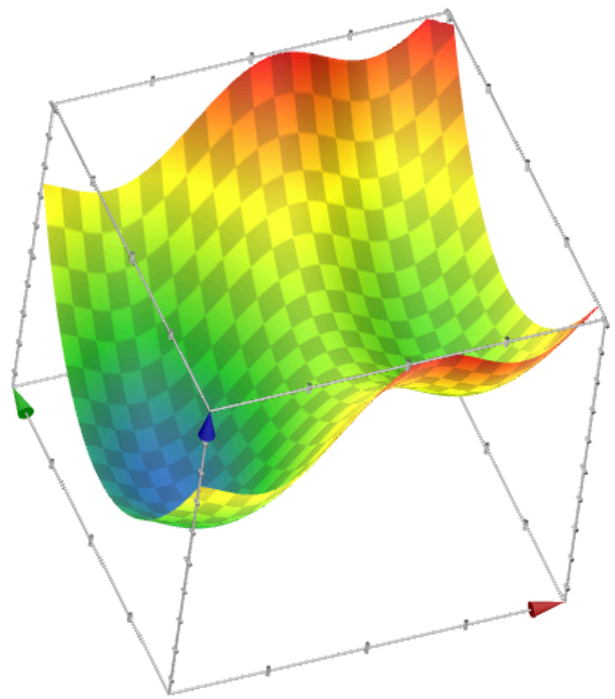
- A function *f* on $\mathbb{R}^m$ is convex if any line segment connecting two points of the graph of *f* lies above or on the graph





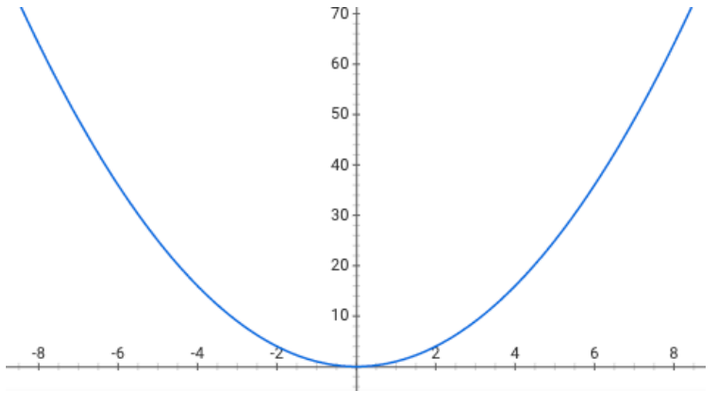


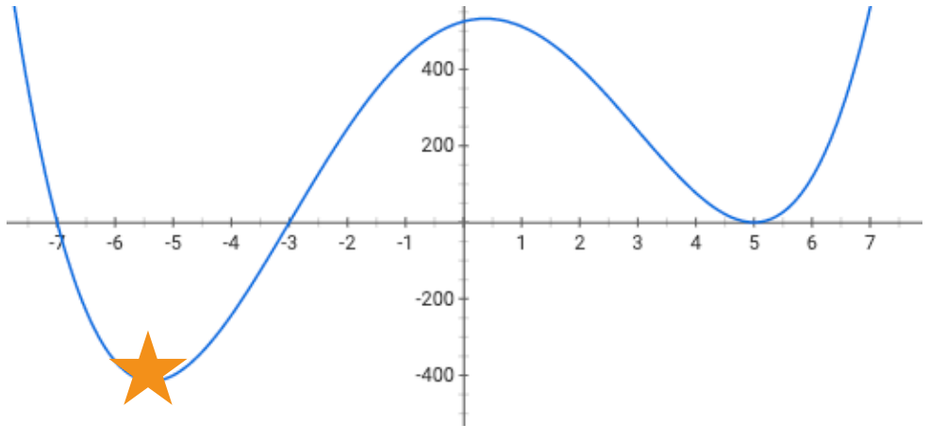
- **Theorem**: Gradient descent performance
  - **Assumptions**: (Choose any $\tilde{\epsilon} > 0$)
    - *f* is sufficiently "smooth" and convex
    - *f* has at least one global optimum

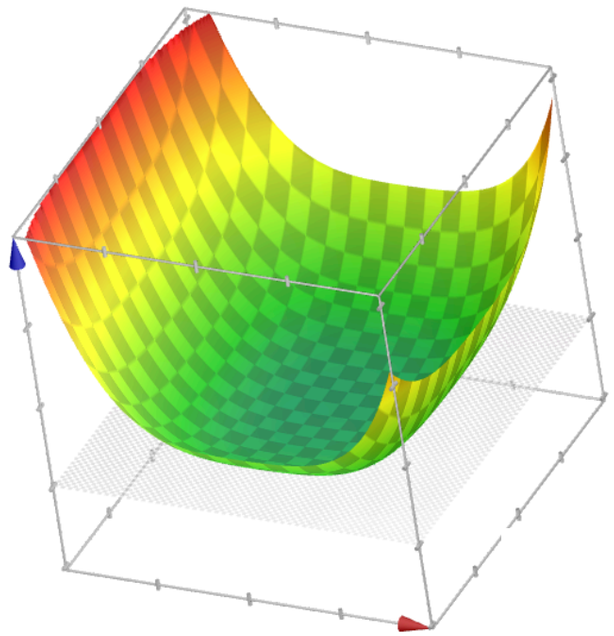


Gradient descent properties

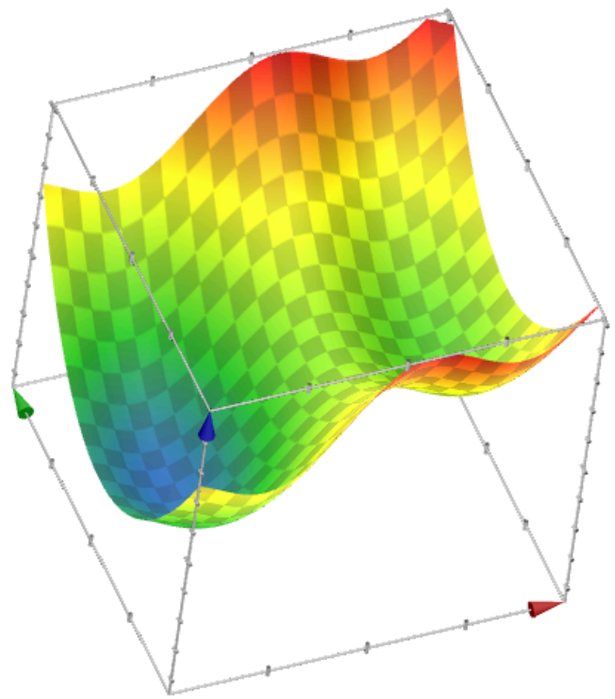
- A function *f* on $\mathbb{R}^m$ is convex if any line segment connecting two points of the graph of *f* lies above or on the graph





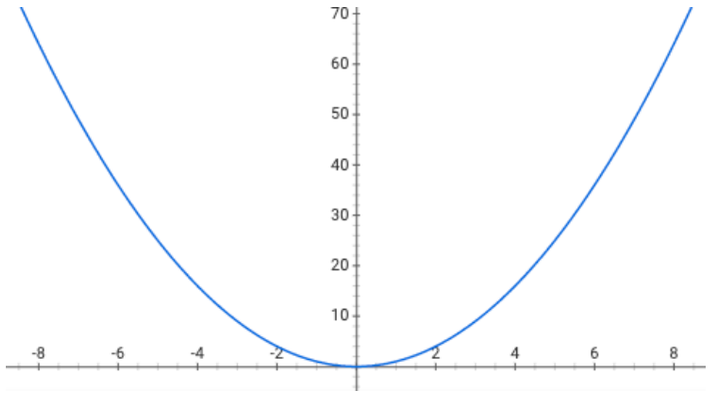


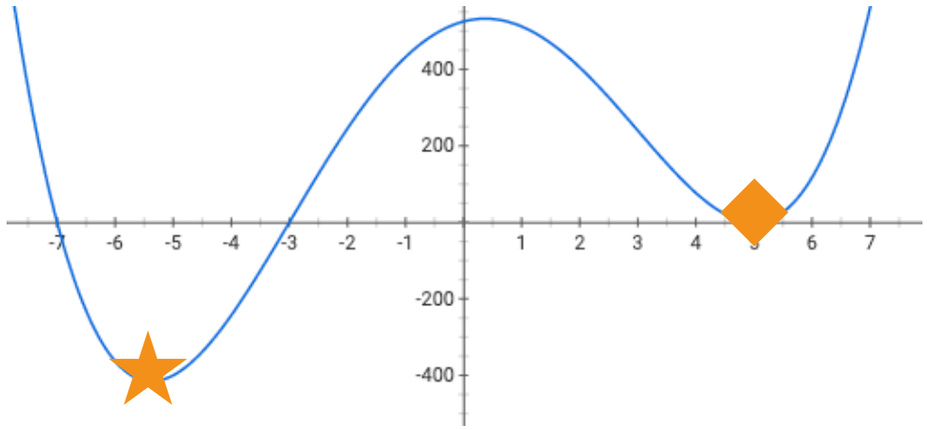
- **Theorem**: Gradient descent performance
  - **Assumptions**: (Choose any $\tilde{\epsilon} > 0$)
    - *f* is sufficiently "smooth" and convex
    - *f* has at least one global optimum

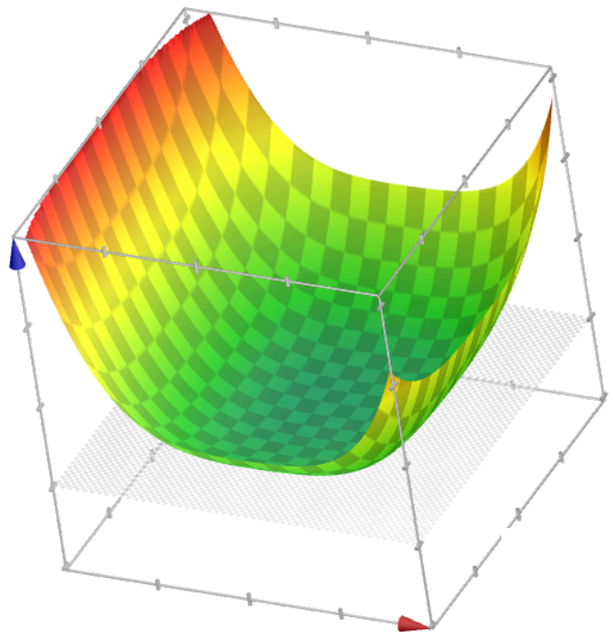


Gradient descent properties

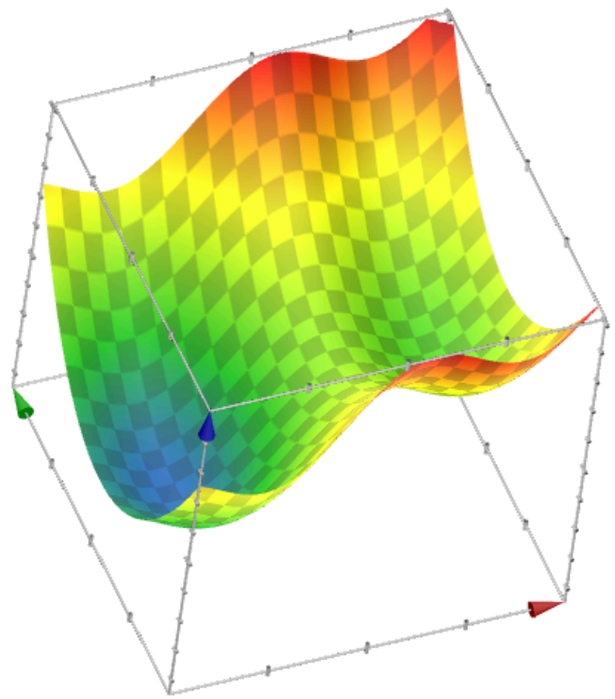
- A function *f* on $\mathbb{R}^m$ is convex if any line segment connecting two points of the graph of *f* lies above or on the graph





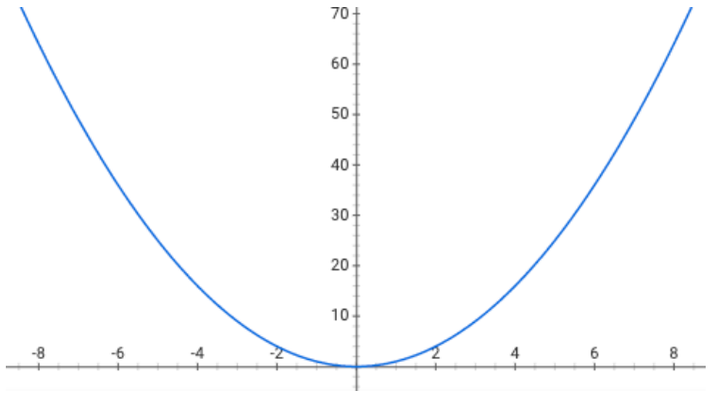


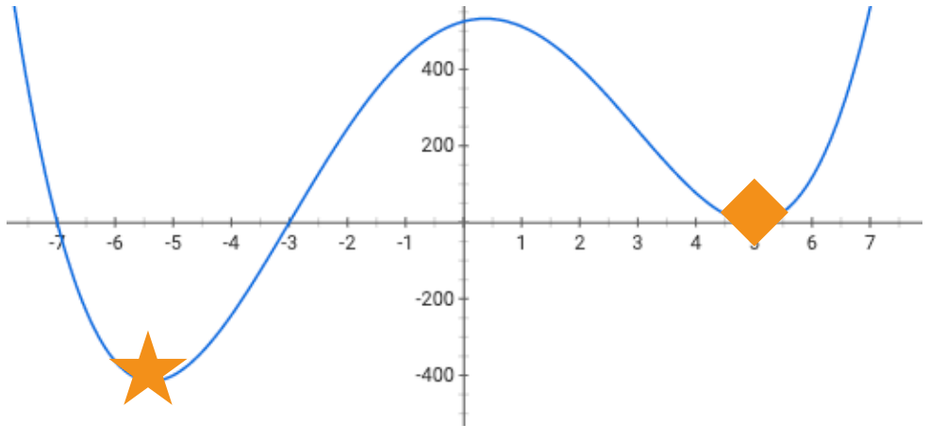
- **Theorem**: Gradient descent performance
  - **Assumptions**: (Choose any $\tilde{\epsilon} > 0$)
    - *f* is sufficiently "smooth" and convex
    - *f* has at least one global optimum

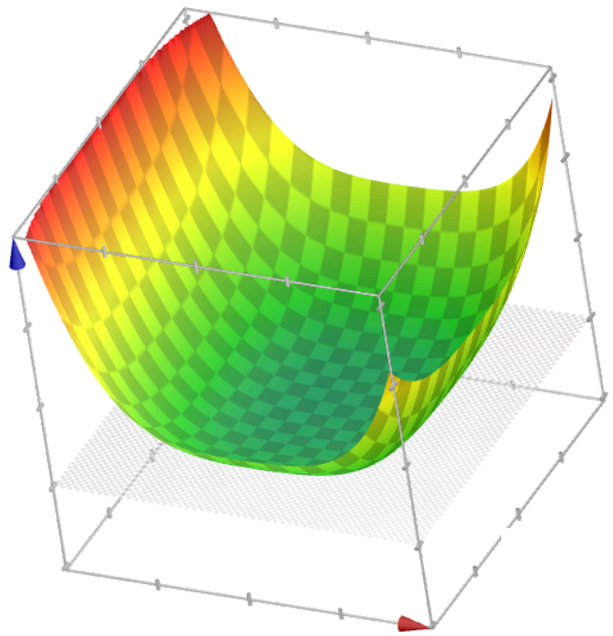


Gradient descent properties

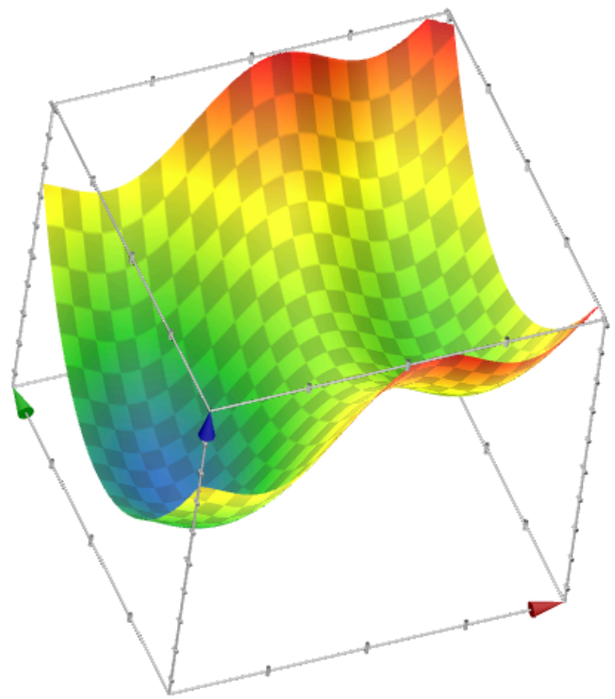
- A function *f* on $\mathbb{R}^m$ is convex if any line segment connecting two points of the graph of *f* lies above or on the graph





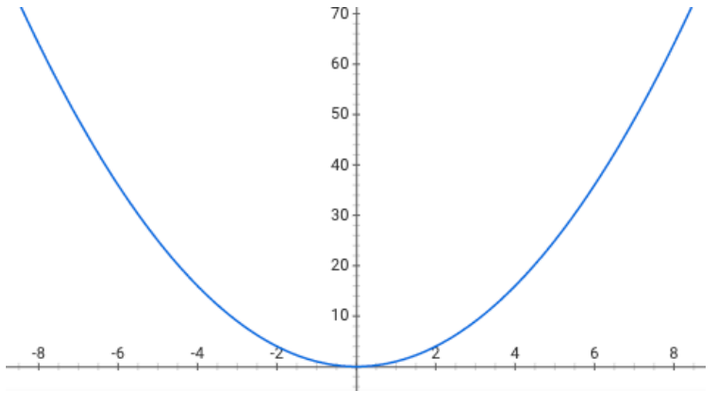


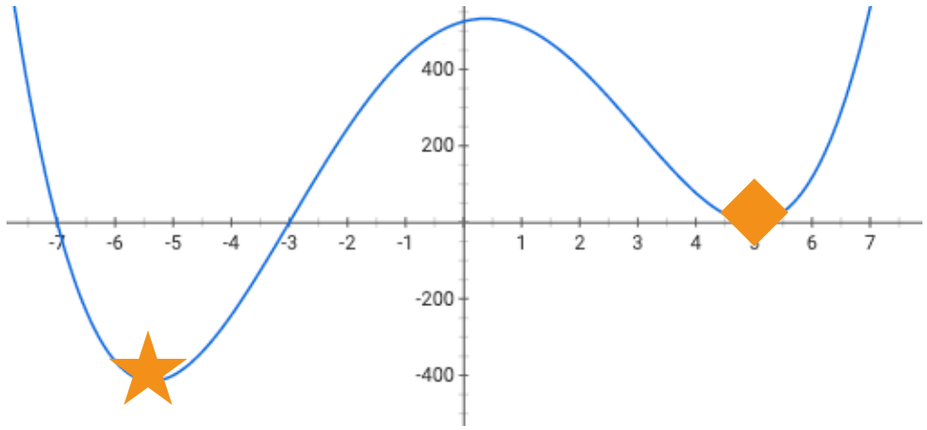
- **Theorem**: Gradient descent performance
  - **Assumptions**: (Choose any $\tilde{\epsilon} > 0$)
    - *f* is sufficiently "smooth" and convex
    - *f* has at least one global optimum
    - η is sufficiently small

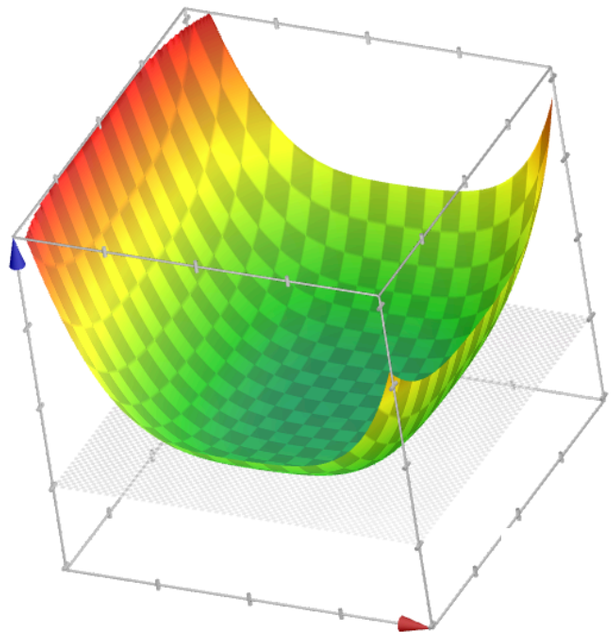


Gradient descent properties

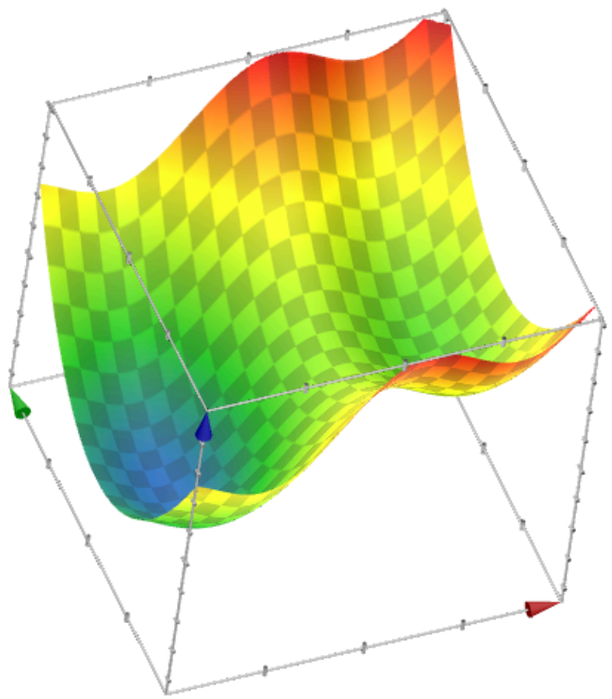
- A function *f* on $\mathbb{R}^m$ is convex if any line segment connecting two points of the graph of *f* lies above or on the graph







- **Theorem**: Gradient descent performance
  - **Assumptions**: (Choose any $\tilde{\epsilon} > 0$)
    - *f* is sufficiently "smooth" and convex
    - *f* has at least one global optimum
    - η is sufficiently small
  - **Conclusion**: If run long enough, gradient descent will return a value within $\tilde{\epsilon}$ of a global optimum Θ



Gradient descent for logistic regression

Gradient descent for logistic regression

- Loss $J_{\text{lr}}(\Theta) = J_{\text{lr}}(\theta, \theta_0)$ is differentiable

Gradient descent for logistic regression

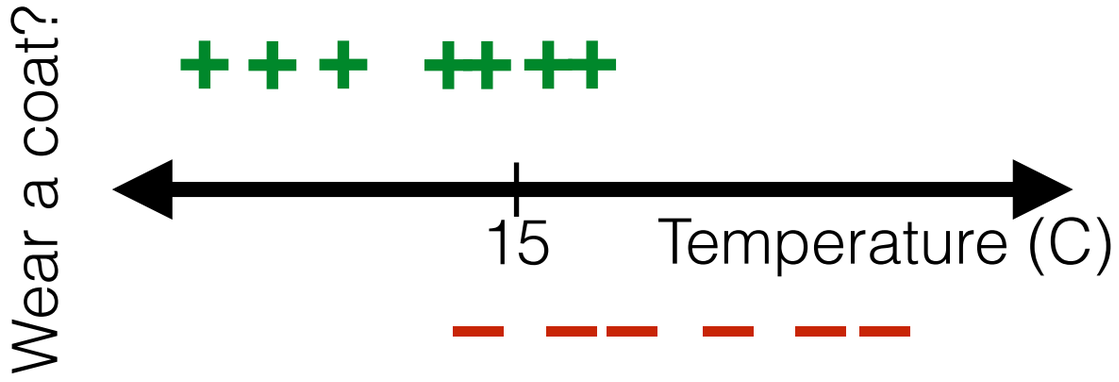
- Loss $J_{\text{lr}}(\Theta) = J_{\text{lr}}(\theta, \theta_0)$ is differentiable & convex

Gradient descent for logistic regression

- Loss $J_{\text{lr}}(\Theta) = J_{\text{lr}}(\theta, \theta_0)$ is differentiable & convex
- Run `Gradient-Descent(` $\Theta_{\text{init}}, \eta, J_{lr}, \nabla_\Theta J_{lr}, \epsilon$)

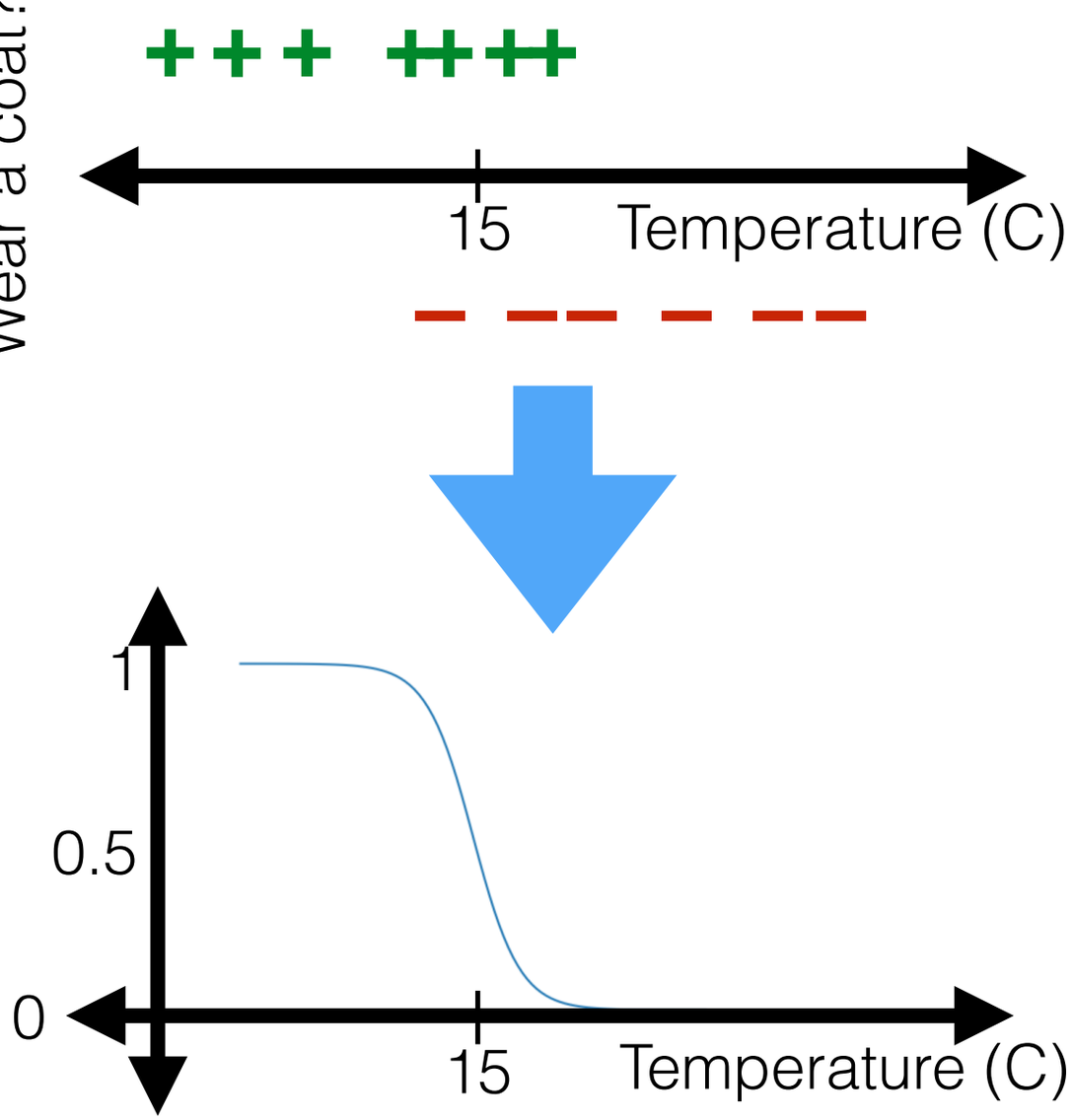
Gradient descent for logistic regression

- Loss $J_{\text{lr}}(\Theta) = J_{\text{lr}}(\theta, \theta_0)$ is differentiable & convex
- Run `Gradient-Descent(` $\Theta_{\text{init}}, \eta, J_{lr}, \nabla_\Theta J_{lr}, \epsilon$)



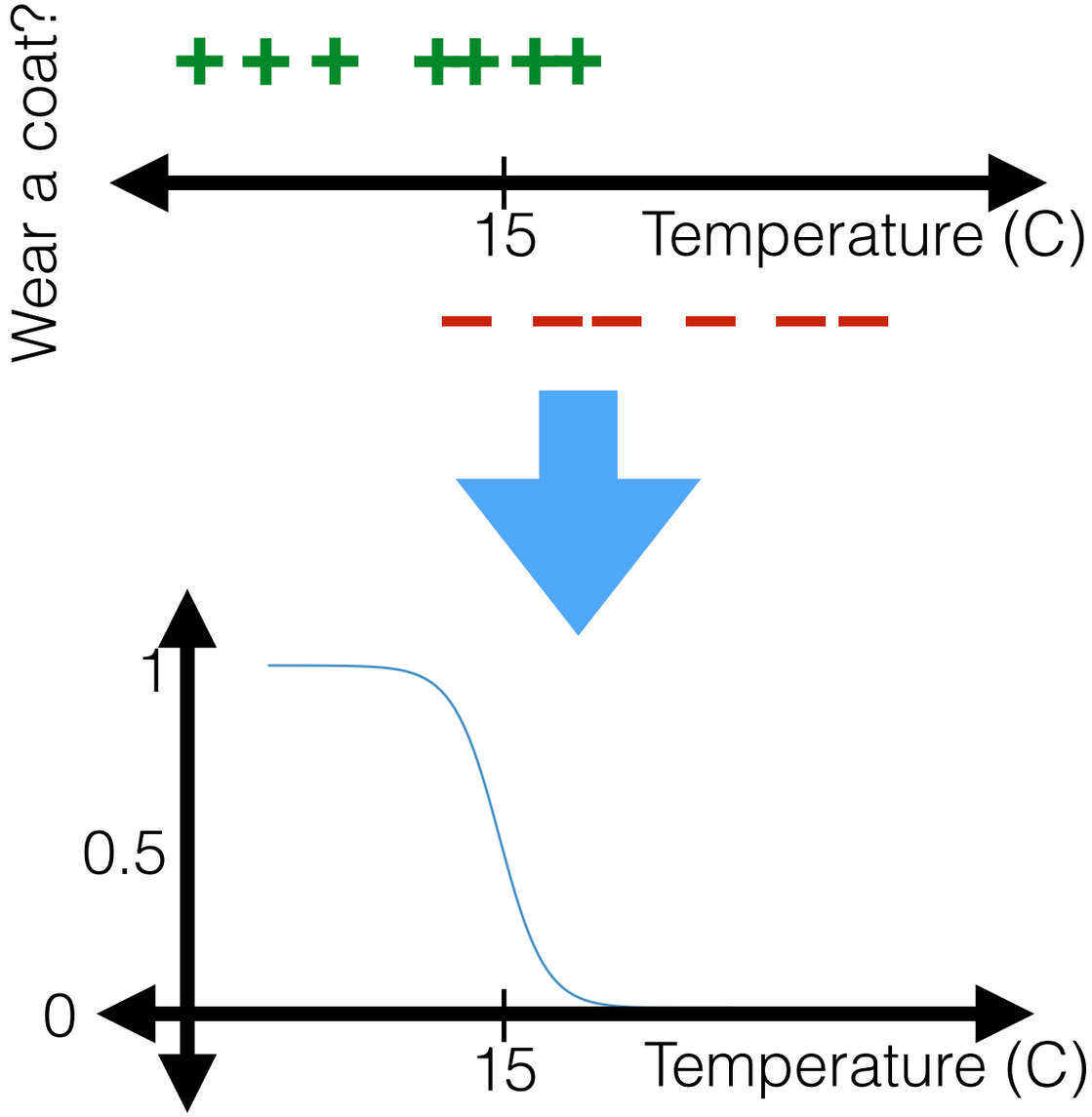
Gradient descent for logistic regression

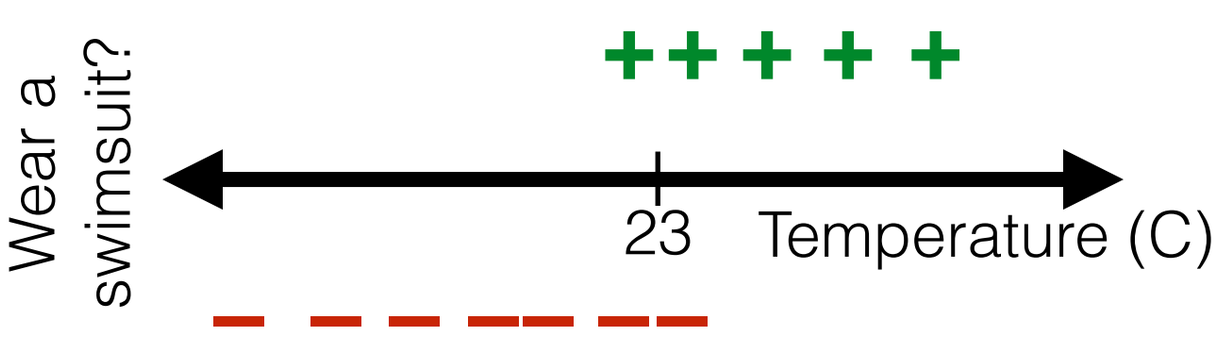
- Loss $J_{\text{lr}}(\Theta) = J_{\text{lr}}(\theta, \theta_0)$ is differentiable & convex
- Run `Gradient-Descent(` $\Theta_{\text{init}}, \eta, J_{lr}, \nabla_\Theta J_{lr}, \epsilon$)



Gradient descent for logistic regression

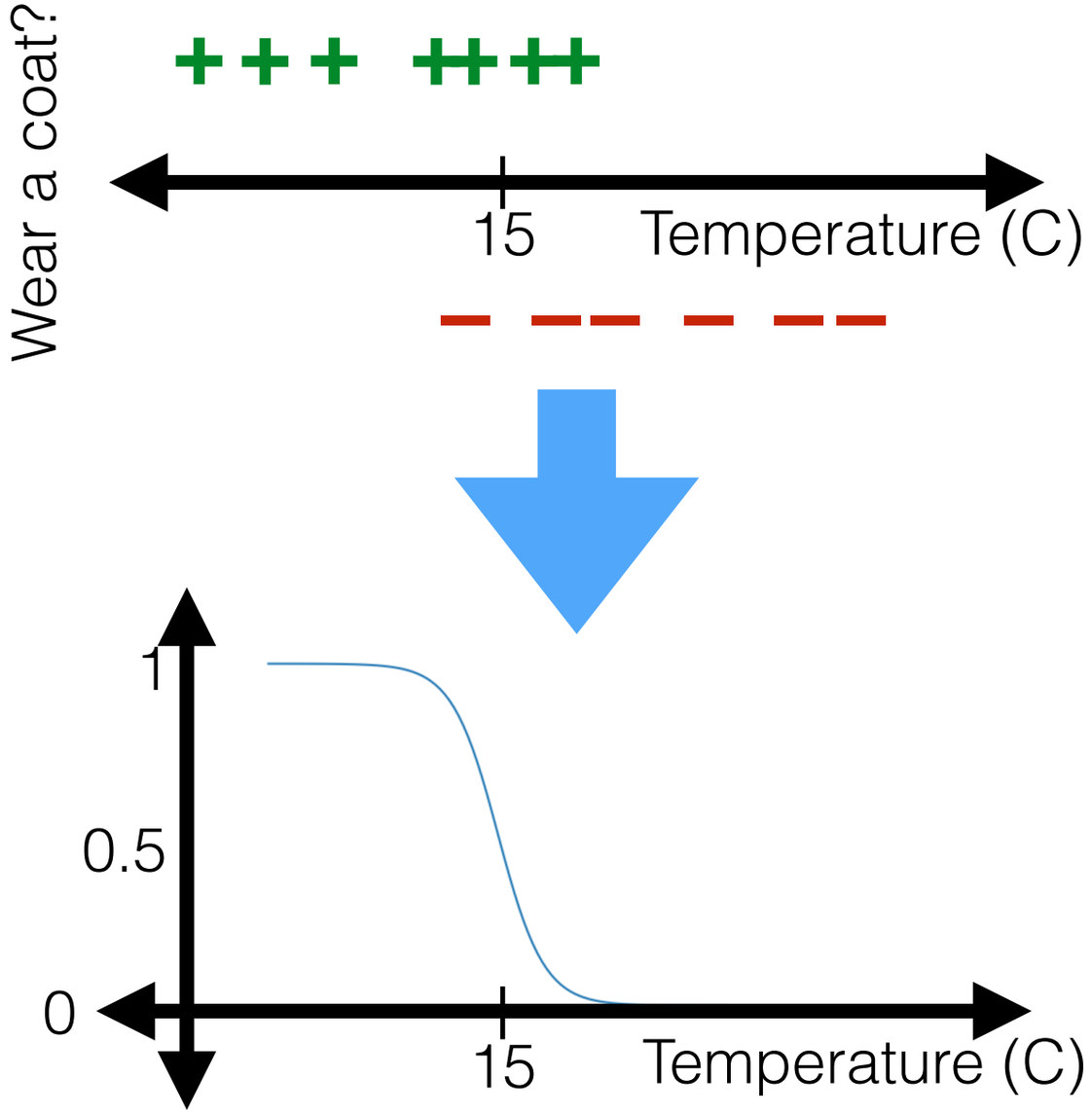
- Loss $J_{\text{lr}}(\Theta) = J_{\text{lr}}(\theta, \theta_0)$ is differentiable & convex
- Run `Gradient-Descent` ($\Theta_{\text{init}}, \eta, J_{lr}, \nabla_\Theta J_{lr}, \epsilon$)

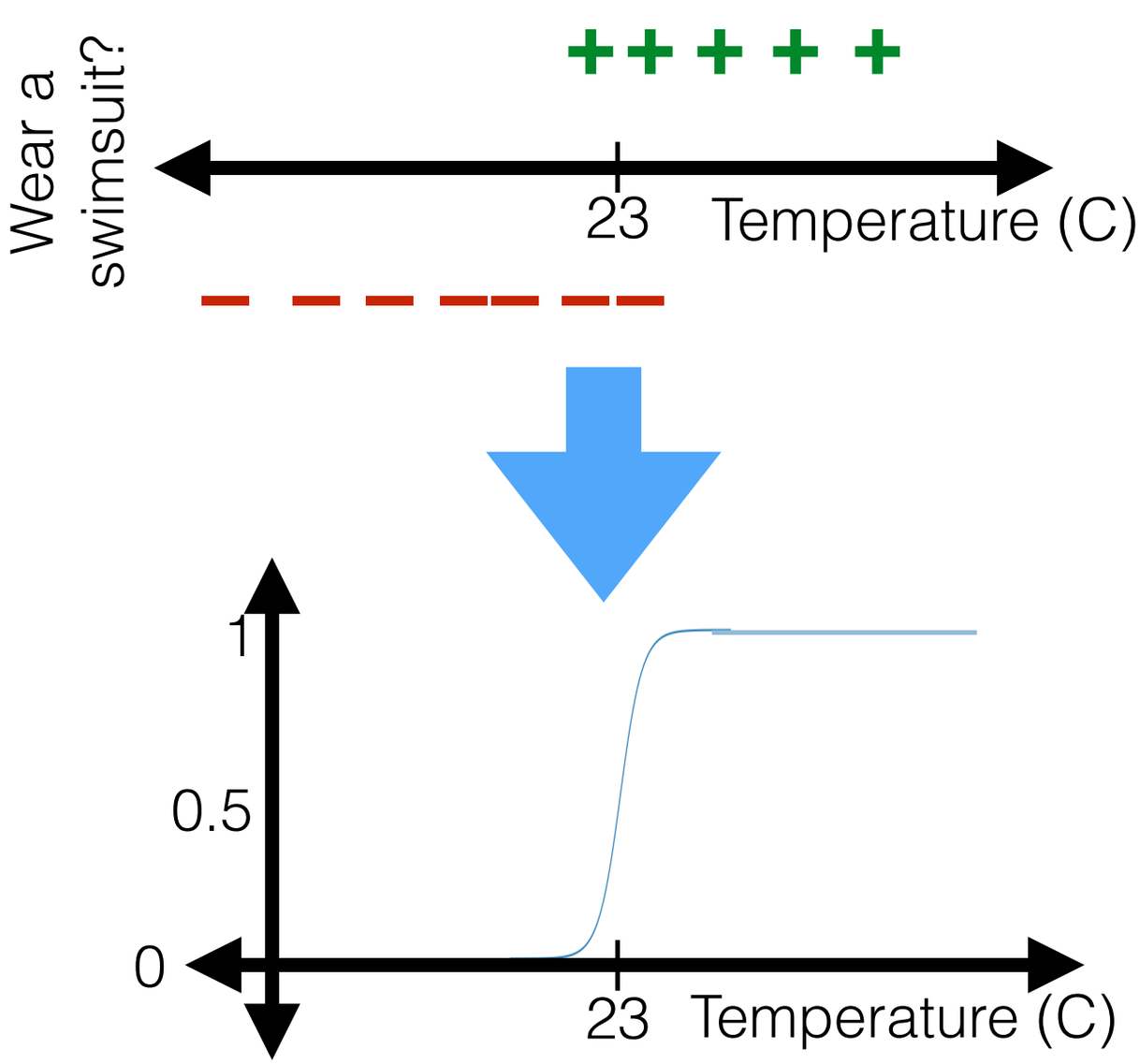




Gradient descent for logistic regression

- Loss $J_{\text{lr}}(\Theta) = J_{\text{lr}}(\theta, \theta_0)$ is differentiable & convex
- Run `Gradient-Descent` ($\Theta_{\text{init}}, \eta, J_{lr}, \nabla_\Theta J_{lr}, \epsilon$)



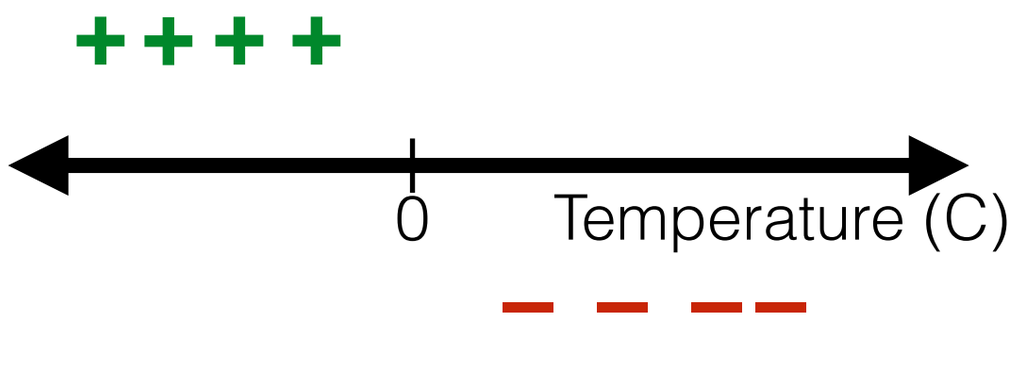


Gradient descent for logistic regression

- Loss $J_{\text{lr}}(\Theta) = J_{\text{lr}}(\theta, \theta_0)$ is differentiable & convex
- Run `Gradient-Descent(` $\Theta_{\text{init}}, \eta, J_{lr}, \nabla_\Theta J_{lr}, \epsilon$)

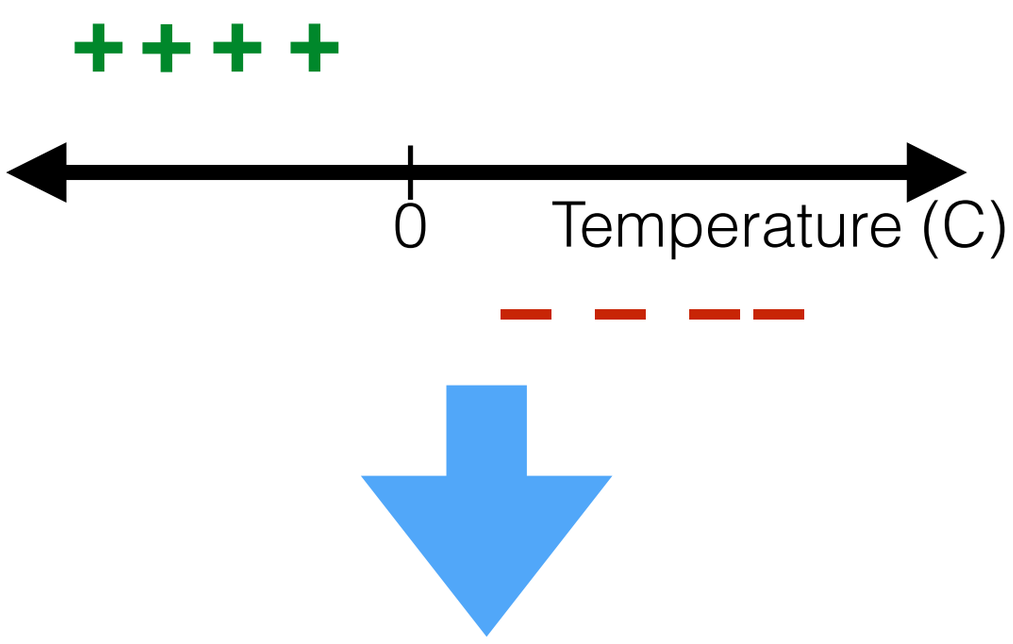
Gradient descent for logistic regression

- Loss $J_{\text{lr}}(\Theta) = J_{\text{lr}}(\theta, \theta_0)$ is differentiable & convex
- Run `Gradient-Descent` ($\Theta_{\text{init}}, \eta, J_{lr}, \nabla_\Theta J_{lr}, \epsilon$)



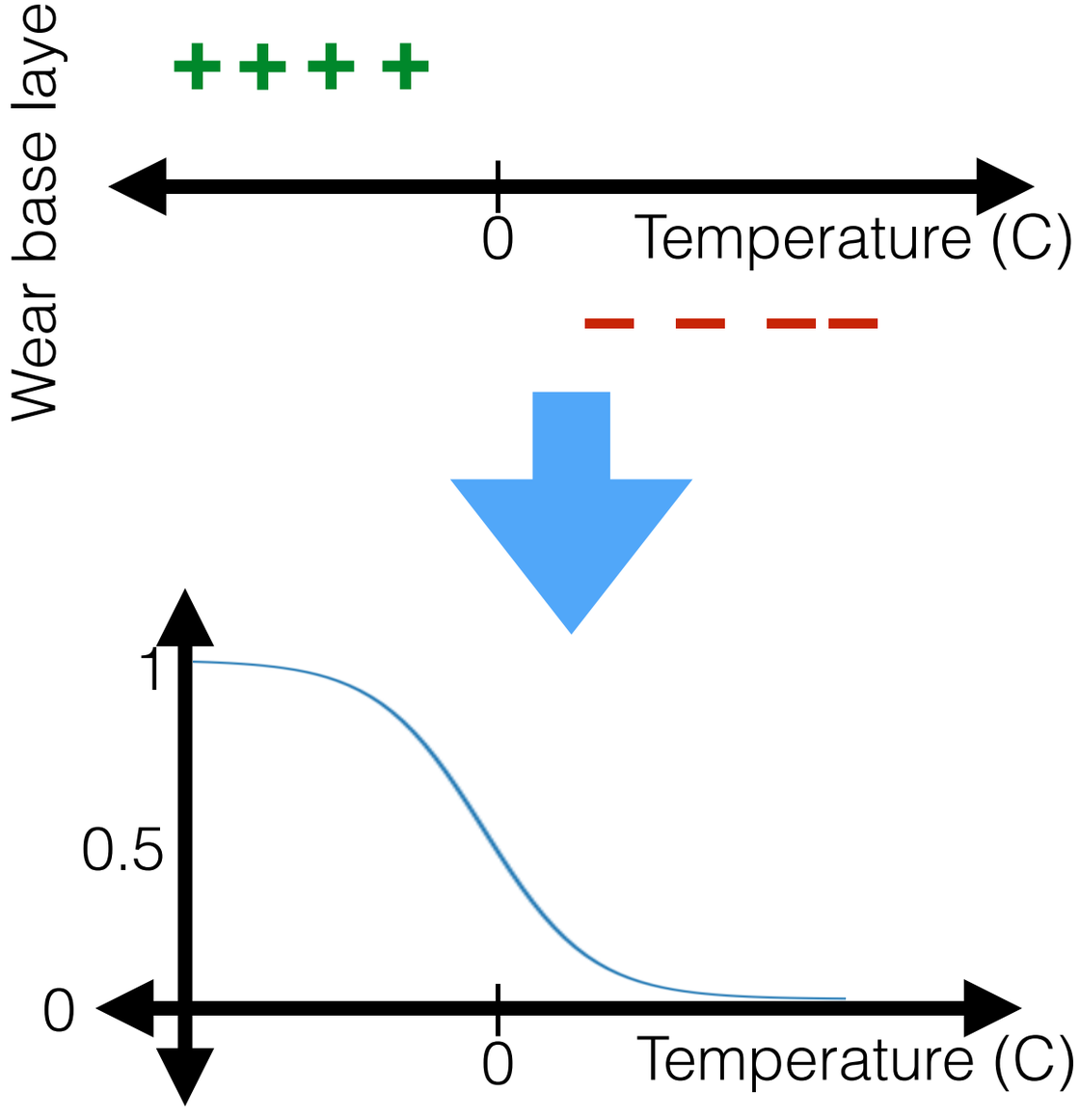
Gradient descent for logistic regression

- Loss $J_{\text{lr}}(\Theta) = J_{\text{lr}}(\theta, \theta_0)$ is differentiable & convex
- Run `Gradient-Descent(` $\Theta_{\text{init}}, \eta, J_{lr}, \nabla_\Theta J_{lr}, \epsilon$)



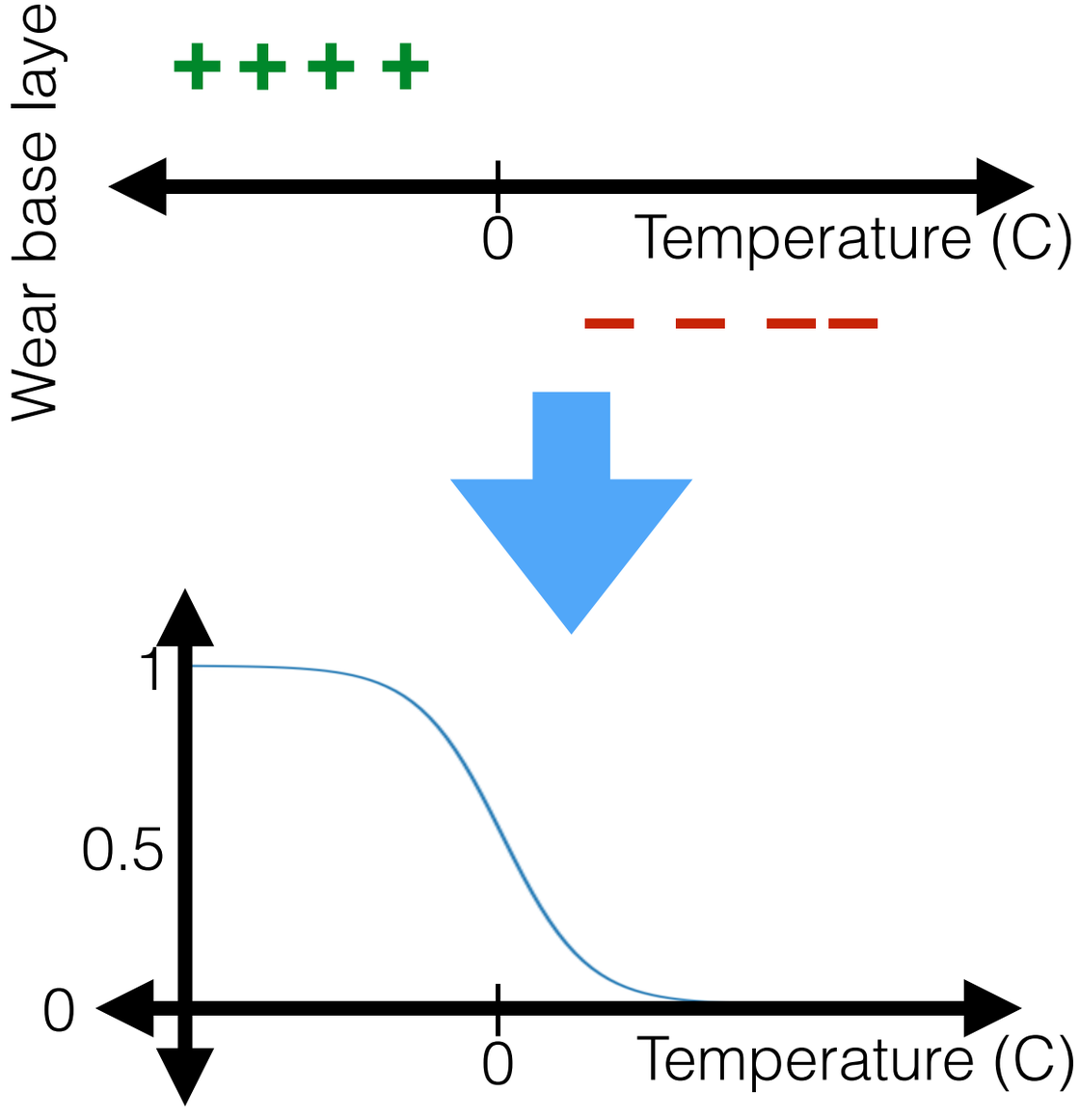
Gradient descent for logistic regression

- Loss $J_{\text{lr}}(\Theta) = J_{\text{lr}}(\theta, \theta_0)$ is differentiable & convex
- Run `Gradient-Descent(` $\Theta_{\text{init}}, \eta, J_{lr}, \nabla_\Theta J_{lr}, \epsilon$)



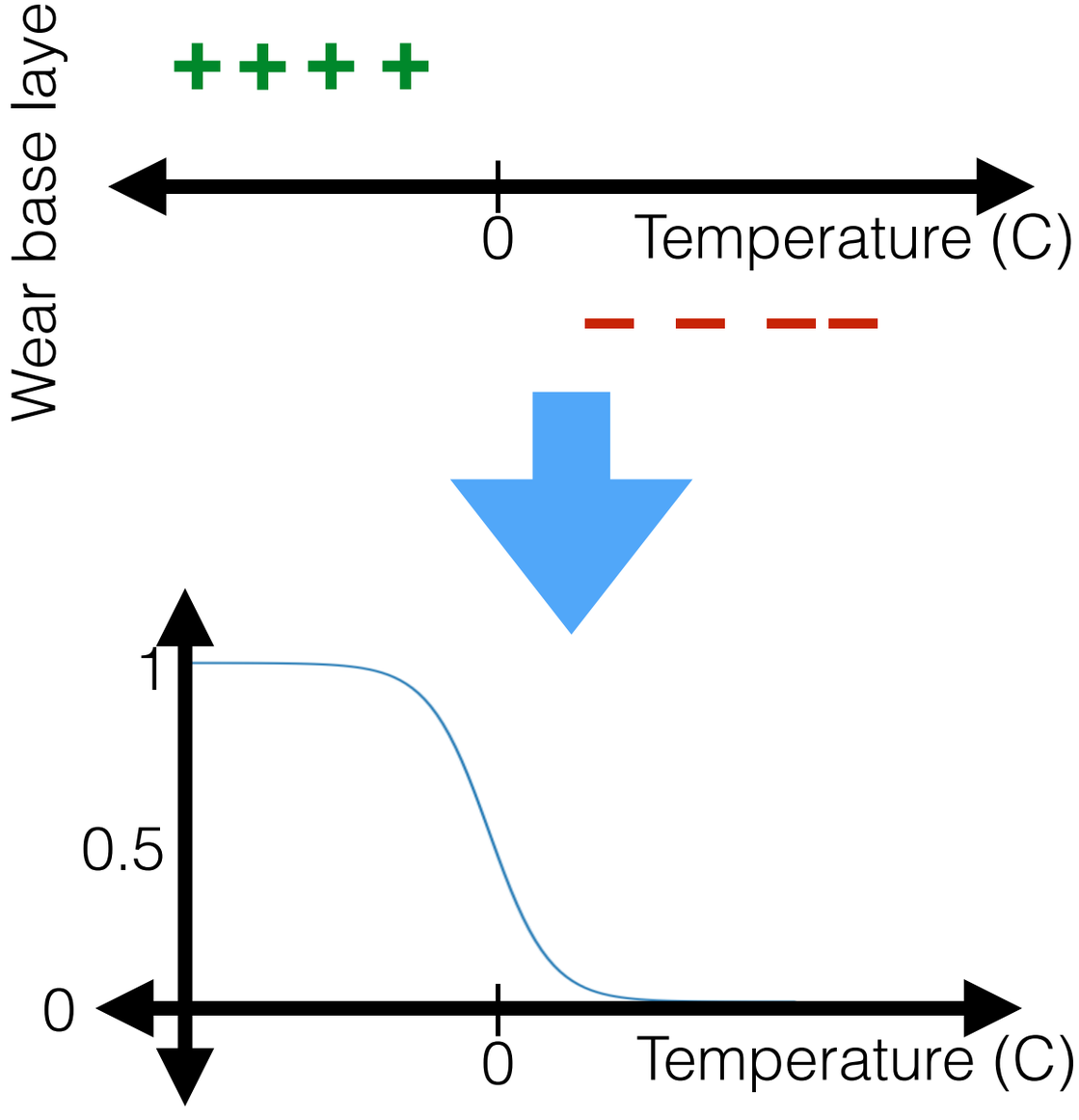
Gradient descent for logistic regression

- Loss $J_{\text{lr}}(\Theta) = J_{\text{lr}}(\theta, \theta_0)$ is differentiable & convex
- Run `Gradient-Descent(` $\Theta_{\text{init}}, \eta, J_{lr}, \nabla_\Theta J_{lr}, \epsilon$)



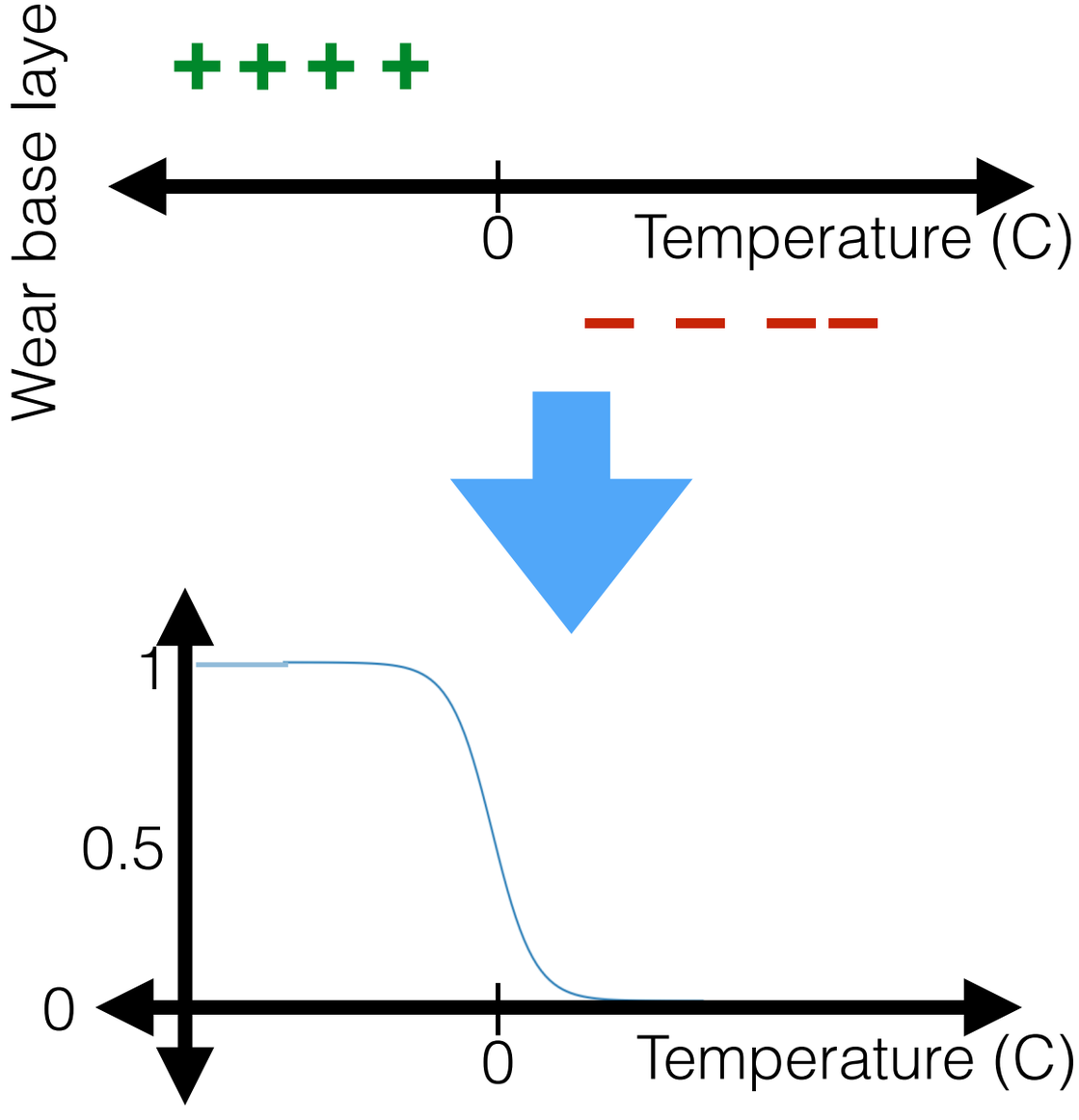
Gradient descent for logistic regression

- Loss $J_{\text{lr}}(\Theta) = J_{\text{lr}}(\theta, \theta_0)$ is differentiable & convex
- Run `Gradient-Descent(` $\Theta_{\text{init}}, \eta, J_{lr}, \nabla_\Theta J_{lr}, \epsilon$)



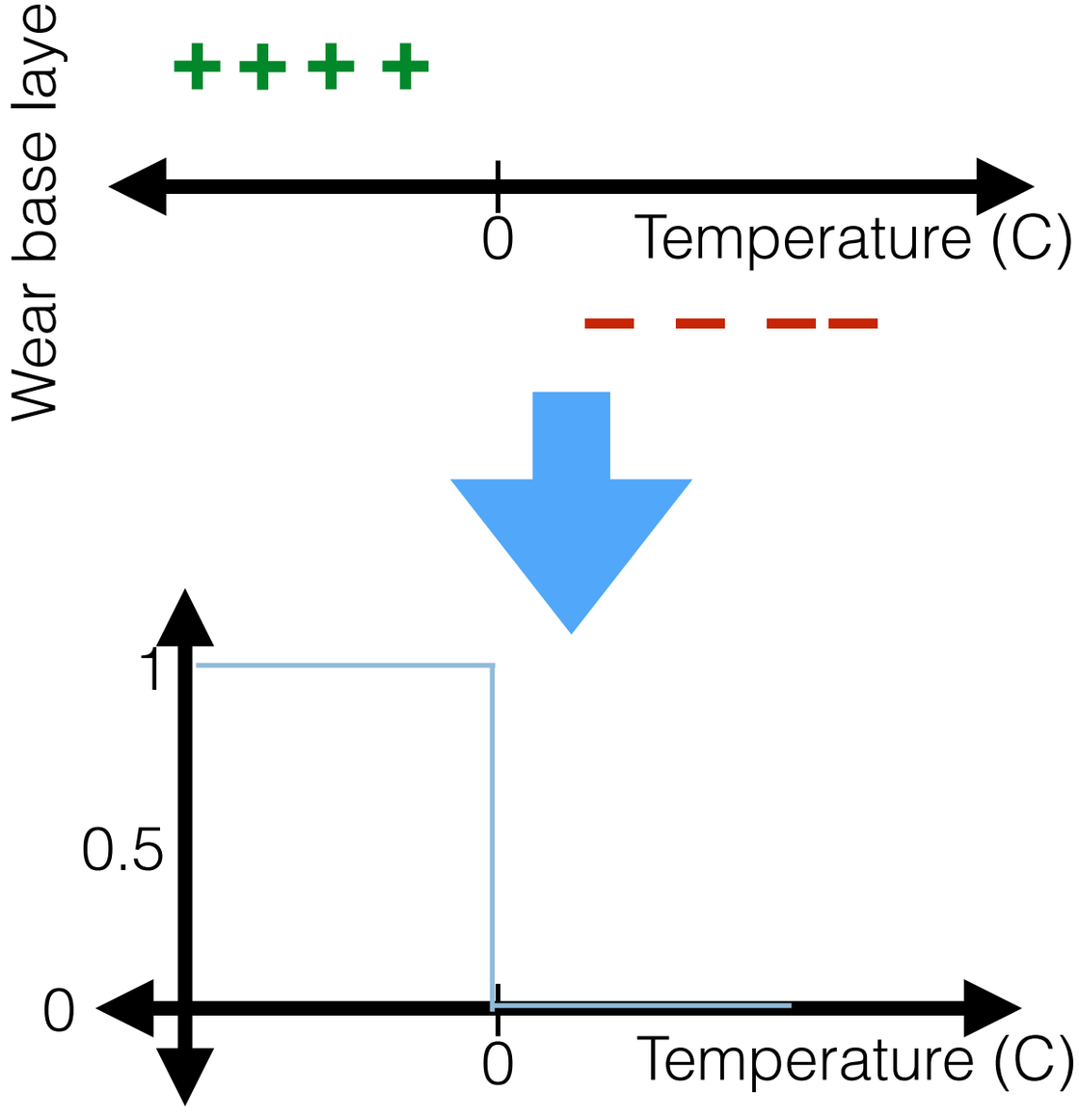
Gradient descent for logistic regression

- Loss $J_{\text{lr}}(\Theta) = J_{\text{lr}}(\theta, \theta_0)$ is differentiable & convex
- Run `Gradient-Descent(` $\Theta_{\text{init}}, \eta, J_{lr}, \nabla_{\Theta} J_{lr}, \epsilon$)



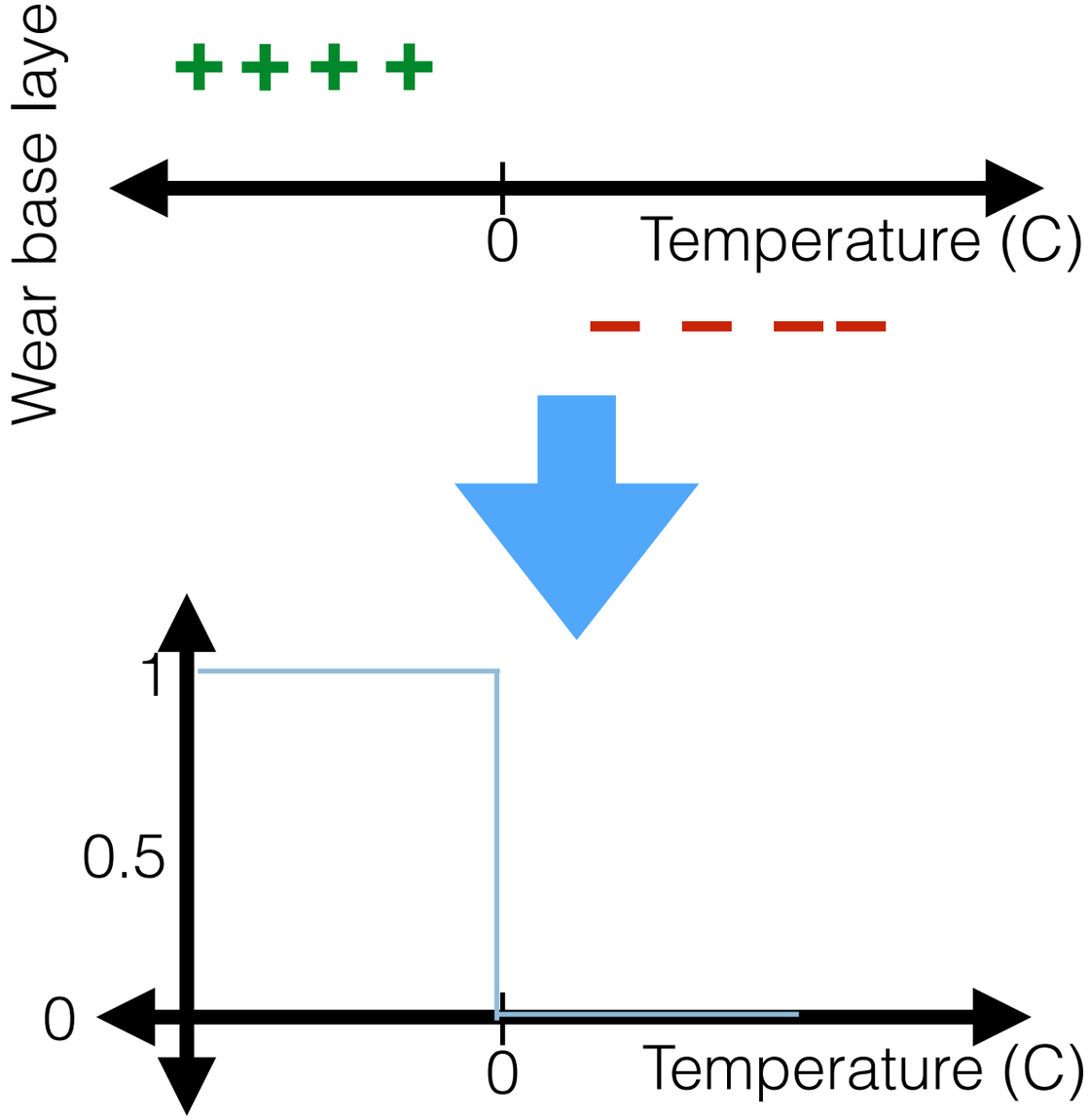
Gradient descent for logistic regression

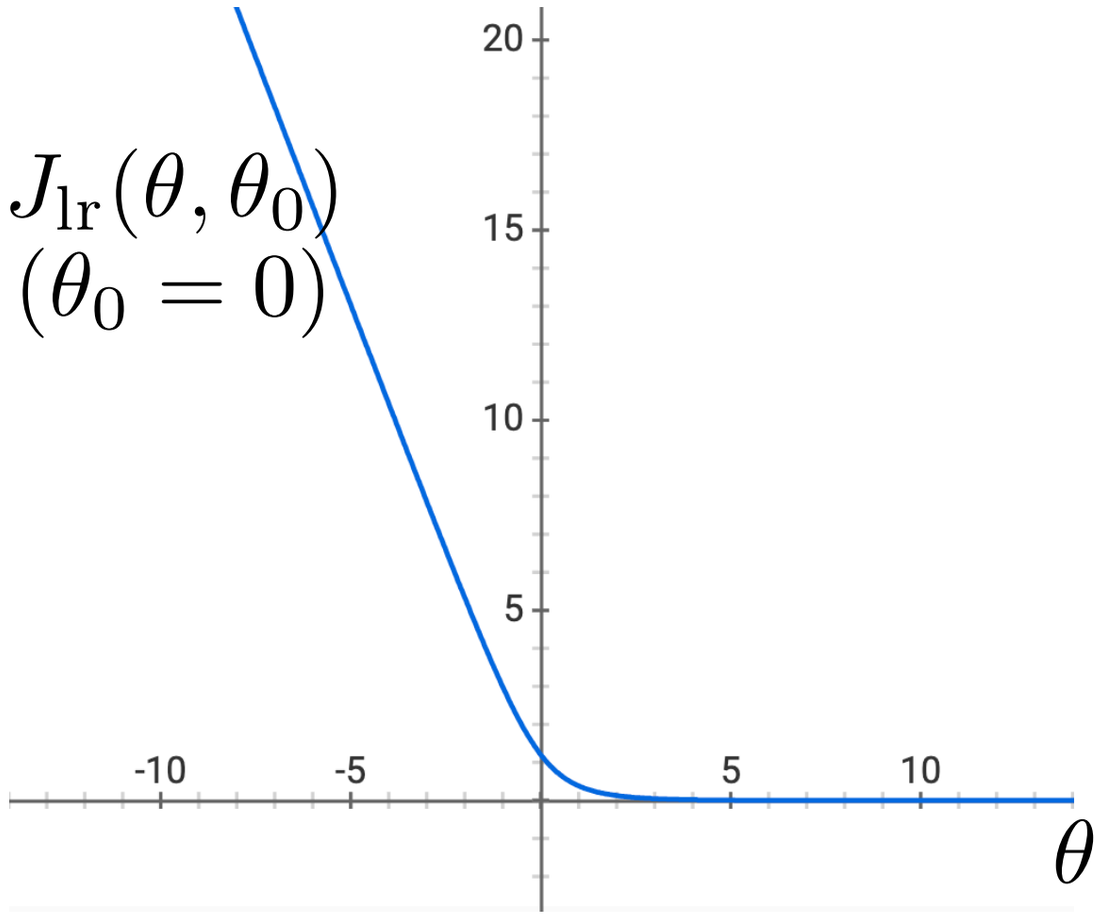
- Loss $J_{\text{lr}}(\Theta) = J_{\text{lr}}(\theta, \theta_0)$ is differentiable & convex
- Run `Gradient-Descent(` $\Theta_{\text{init}}, \eta, J_{lr}, \nabla_\Theta J_{lr}, \epsilon$)



Gradient descent for logistic regression

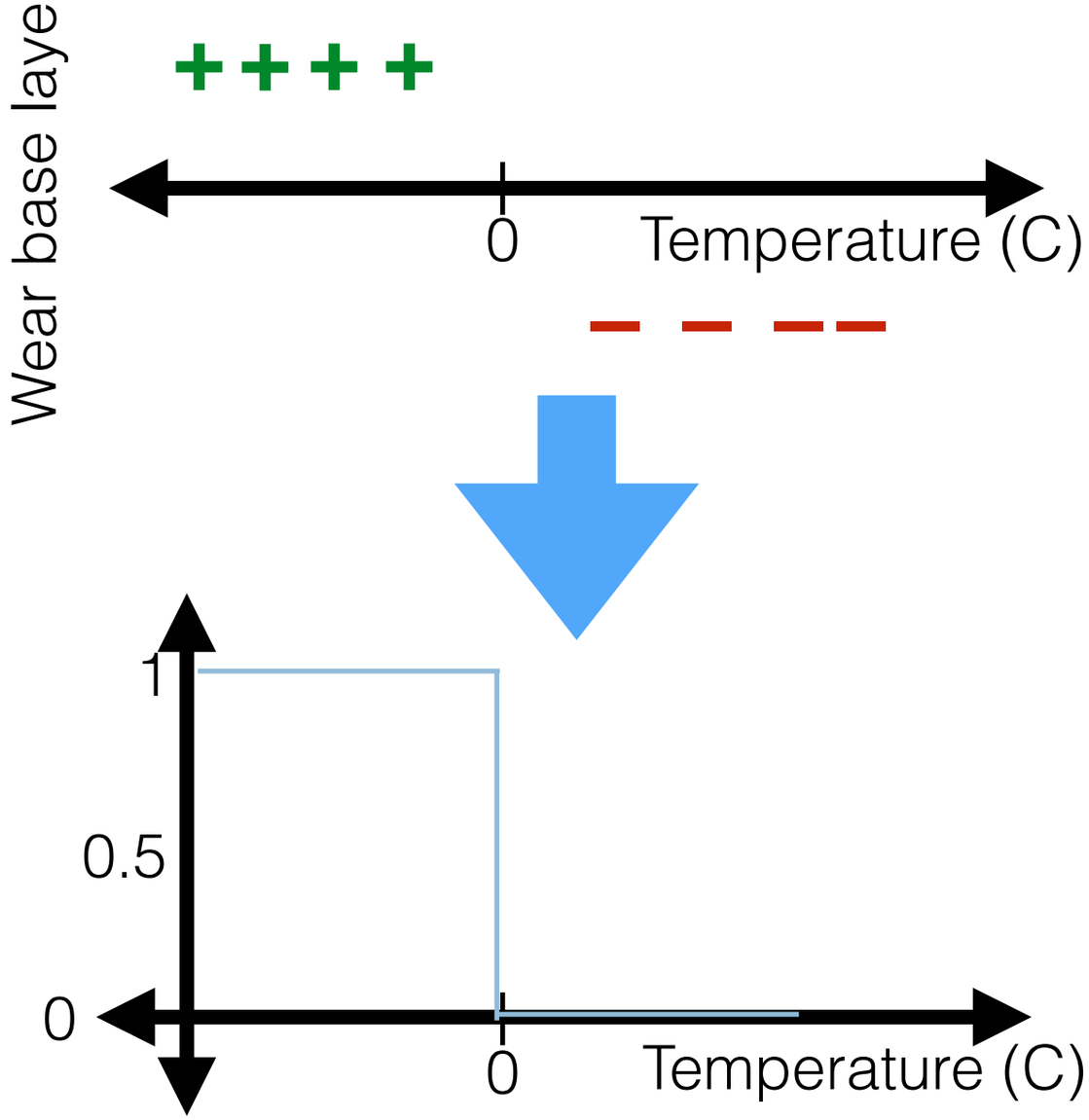
- Loss $J_{\text{lr}}(\Theta) = J_{\text{lr}}(\theta, \theta_0)$ is differentiable & convex
- Run `Gradient-Descent(` $\Theta_{\text{init}}, \eta, J_{lr}, \nabla_\Theta J_{lr}, \epsilon$)

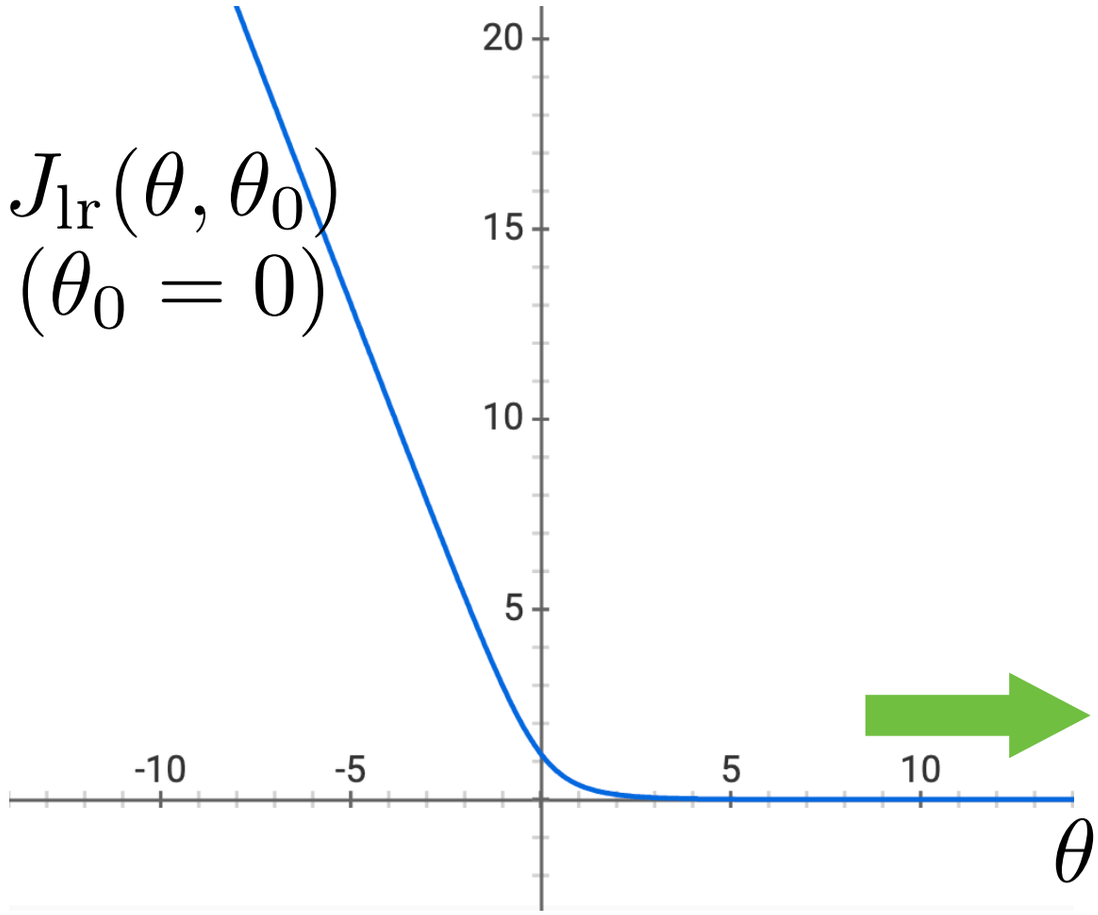




Gradient descent for logistic regression

- Loss $J_{\text{lr}}(\Theta) = J_{\text{lr}}(\theta, \theta_0)$ is differentiable & convex
- Run `Gradient-Descent` ($\Theta_{\text{init}}, \eta, J_{lr}, \nabla_\Theta J_{lr}, \epsilon$)





Logistic regression loss revisited

$$J_{\text{lr}}(\Theta) = J_{\text{lr}}(\theta, \theta_0)$$
$$= \frac{1}{n} \sum_{i=1}^{n} L_{\text{nll}}(\sigma(\theta^\top x^{(i)} + \theta_0), y^{(i)})$$

Logistic regression loss revisited

$$J_{\text{lr}}(\Theta) = J_{\text{lr}}(\theta, \theta_0)$$
$$= \frac{1}{n} \sum_{i=1}^{n} L_{\text{nll}}(\sigma(\theta^\top x^{(i)} + \theta_0), y^{(i)}) + \lambda \|\theta\|^2 \qquad (\lambda \geq 0)$$

Logistic regression loss revisited

$$J_{\text{lr}}(\Theta) = J_{\text{lr}}(\theta, \theta_0)$$
$$= \frac{1}{n} \sum_{i=1}^{n} L_{\text{nll}}(\sigma(\theta^\top x^{(i)} + \theta_0), y^{(i)}) + \lambda \|\theta\|^2 \qquad (\lambda \geq 0)$$

- A “regularizer” or “penalty” $R(\theta) = \lambda \|\theta\|^2$

Logistic regression loss revisited

$$J_{\text{lr}}(\Theta) = J_{\text{lr}}(\theta, \theta_0)$$
$$= \frac{1}{n} \sum_{i=1}^{n} L_{\text{nll}}(\sigma(\theta^\top x^{(i)} + \theta_0), y^{(i)}) + \lambda \|\theta\|^2 \quad (\lambda \geq 0)$$

- A "regularizer" or "penalty" $R(\theta) = \lambda \|\theta\|^2$
- Penalizes being overly certain

Logistic regression loss revisited

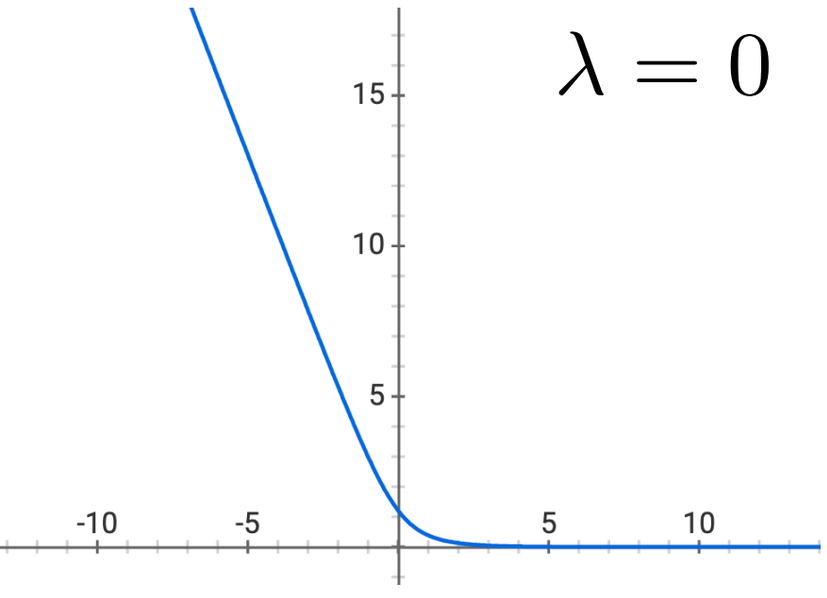
$$J_{\text{lr}}(\Theta) = J_{\text{lr}}(\theta, \theta_0)$$
$$= \frac{1}{n} \sum_{i=1}^{n} L_{\text{nll}}(\sigma(\theta^\top x^{(i)} + \theta_0), y^{(i)}) + \lambda \|\theta\|^2 \quad (\lambda \geq 0)$$

- A "regularizer" or "penalty" $R(\theta) = \lambda \|\theta\|^2$
- Penalizes being overly certain
- Objective is still differentiable & convex (gradient descent)

Logistic regression loss revisited

$$J_{\text{lr}}(\Theta) = J_{\text{lr}}(\theta, \theta_0)$$
$$= \frac{1}{n} \sum_{i=1}^{n} L_{\text{nll}}(\sigma(\theta^\top x^{(i)} + \theta_0), y^{(i)}) + \lambda \|\theta\|^2 \quad (\lambda \geq 0)$$

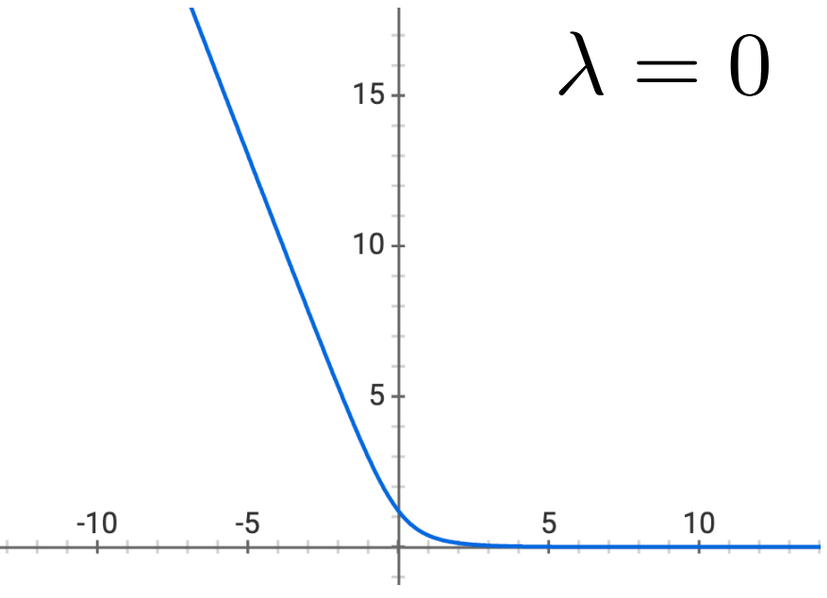
- A "regularizer" or "penalty" $R(\theta) = \lambda \|\theta\|^2$
- Penalizes being overly certain
- Objective is still differentiable & convex (gradient descent)

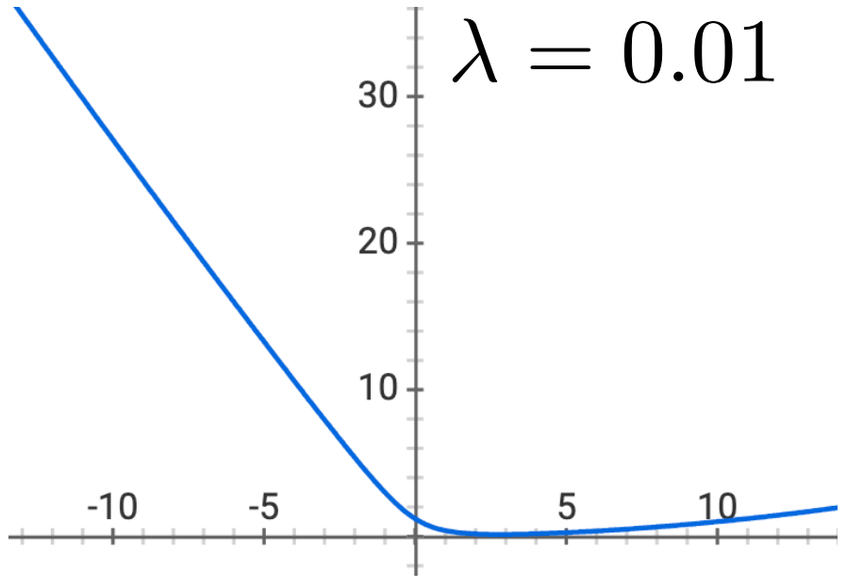


Logistic regression loss revisited

$$J_{\text{lr}}(\Theta) = J_{\text{lr}}(\theta, \theta_0)$$
$$= \frac{1}{n} \sum_{i=1}^{n} L_{\text{nll}}(\sigma(\theta^\top x^{(i)} + \theta_0), y^{(i)}) + \lambda \|\theta\|^2 \quad (\lambda \geq 0)$$

- A "regularizer" or "penalty" $R(\theta) = \lambda \|\theta\|^2$
- Penalizes being overly certain
- Objective is still differentiable & convex (gradient descent)

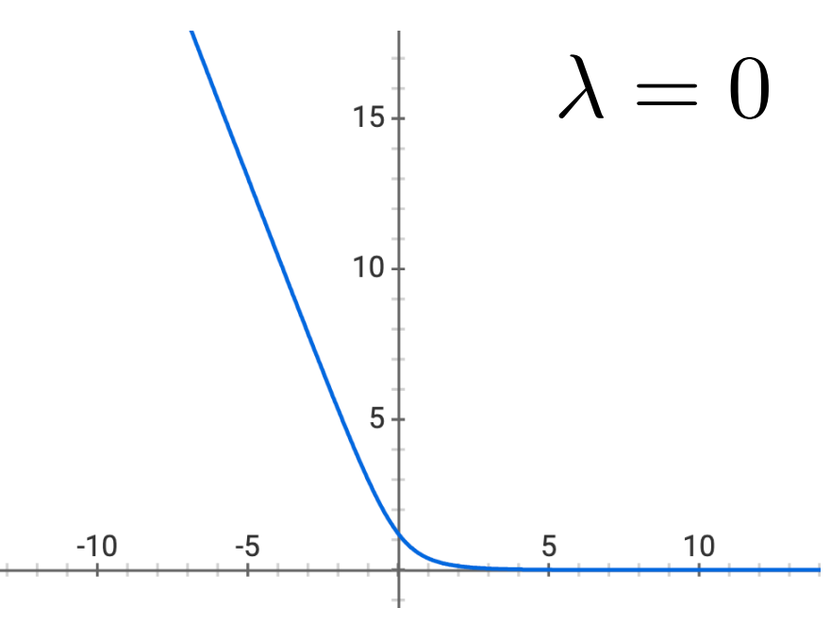


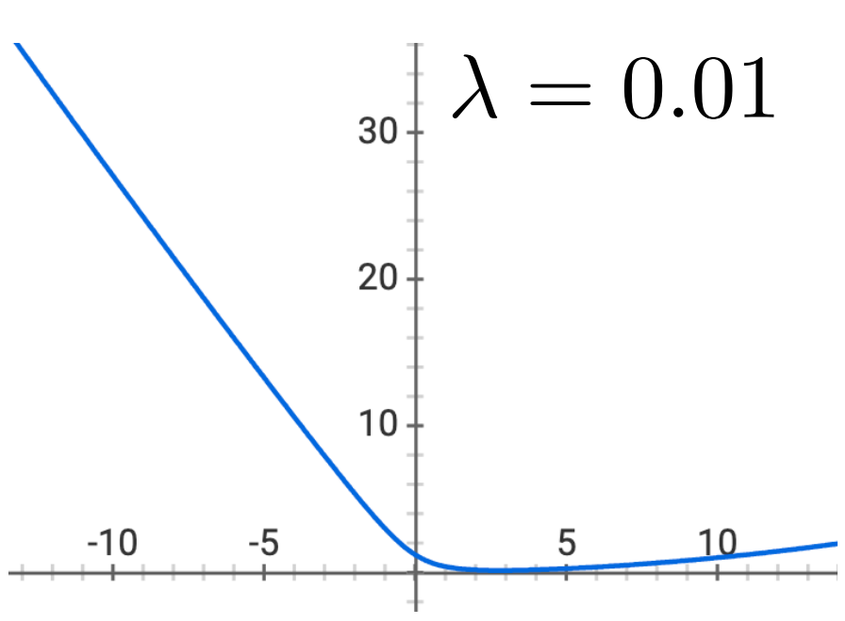


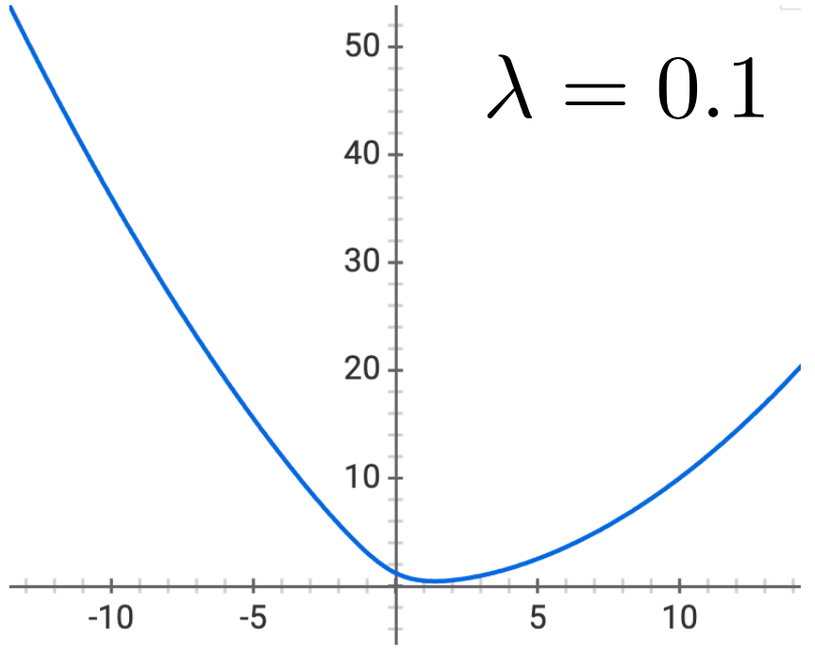
Logistic regression loss revisited

$$J_{\text{lr}}(\Theta) = J_{\text{lr}}(\theta, \theta_0)$$
$$= \frac{1}{n} \sum_{i=1}^{n} L_{\text{nll}}(\sigma(\theta^\top x^{(i)} + \theta_0), y^{(i)}) + \lambda \|\theta\|^2 \quad (\lambda \geq 0)$$

- A "regularizer" or "penalty" $R(\theta) = \lambda \|\theta\|^2$
- Penalizes being overly certain
- Objective is still differentiable & convex (gradient descent)



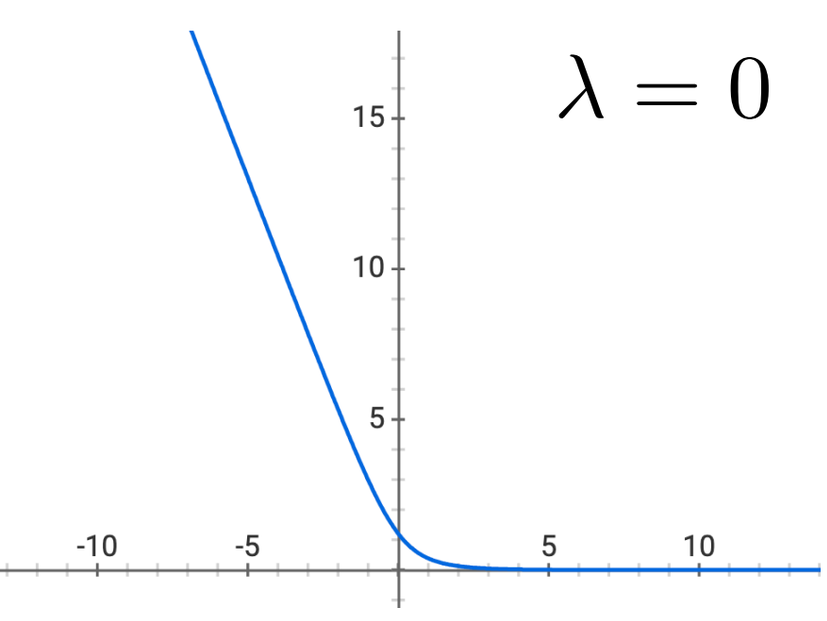


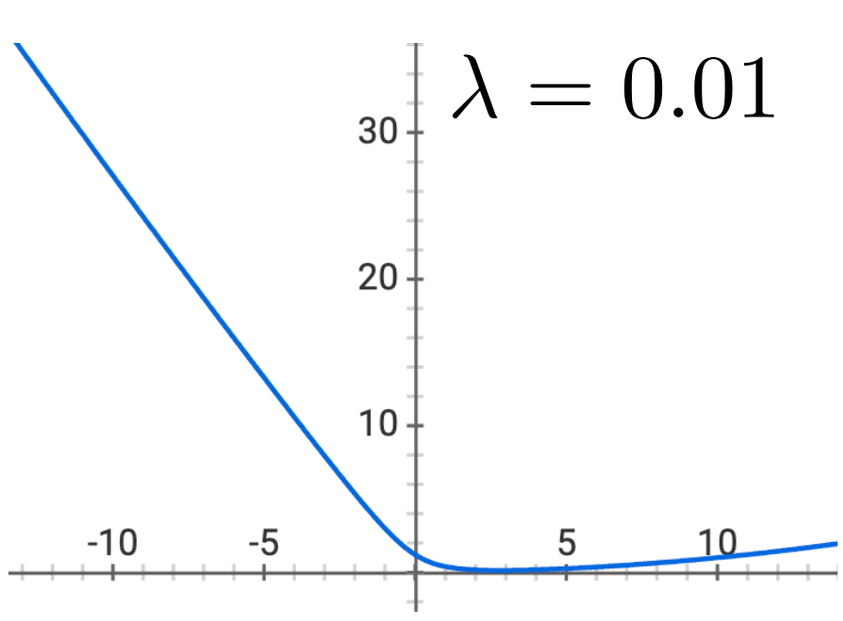


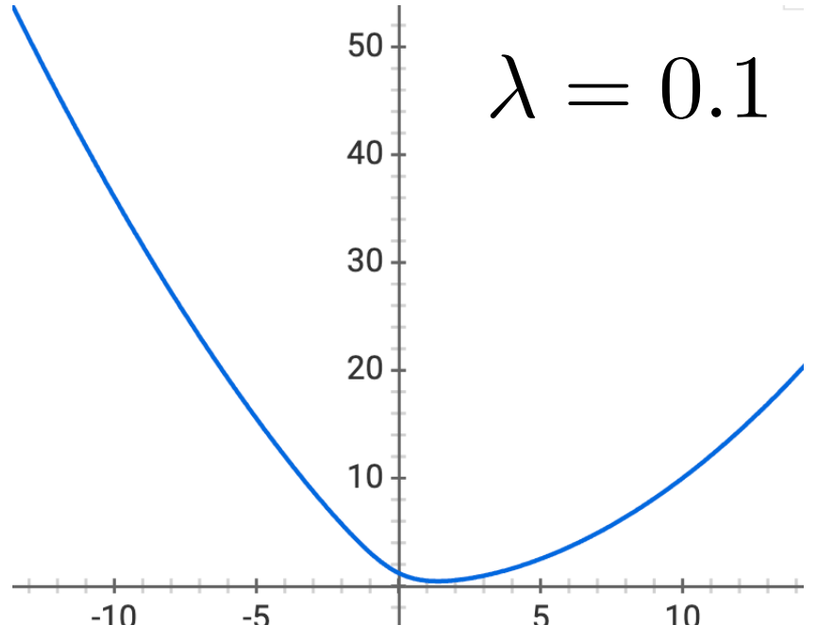
Logistic regression loss revisited

$$J_{\text{lr}}(\Theta) = J_{\text{lr}}(\theta, \theta_0)$$
$$= \frac{1}{n} \sum_{i=1}^{n} L_{\text{nll}}(\sigma(\theta^\top x^{(i)} + \theta_0), y^{(i)}) + \lambda \|\theta\|^2 \quad (\lambda \geq 0)$$

- A "regularizer" or "penalty" $R(\theta) = \lambda \|\theta\|^2$
- Penalizes being overly certain
- Objective is still differentiable & convex (gradient descent)







- How to choose hyperparameters? One option: consider a handful of possible values and compare via CV

Logistic regression learning algorithm

Exactly gradient descent with *f* given by logistic regression objective

Logistic regression learning algorithm

`LR-Gradient-Descent` $(\theta_{\text{init}}, \theta_{0,\text{init}}, \eta, \epsilon\ )$

Exactly gradient descent with *f* given by logistic regression objective

Logistic regression learning algorithm

LR-Gradient-Descent($\theta_{\text{init}}, \theta_{0,\text{init}}, \eta, \epsilon$)

  Initialize $\theta^{(0)} = \theta_{\text{init}}$

  Initialize $\theta_0^{(0)} = \theta_{0,\text{init}}$

Exactly gradient descent with *f* given by logistic regression objective

Logistic regression learning algorithm

LR-Gradient-Descent($\theta_{\text{init}}, \theta_{0,\text{init}}, \eta, \epsilon$)

  Initialize $\theta^{(0)} = \theta_{\text{init}}$

  Initialize $\theta_0^{(0)} = \theta_{0,\text{init}}$

  Initialize t = 0

Exactly gradient descent with *f* given by logistic regression objective

Logistic regression learning algorithm

LR-Gradient-Descent($\theta_{\text{init}}, \theta_{0,\text{init}}, \eta, \epsilon$)

  Initialize $\theta^{(0)} = \theta_{\text{init}}$

  Initialize $\theta_0^{(0)} = \theta_{0,\text{init}}$

  Initialize t = 0

  repeat

Exactly gradient descent with *f* given by logistic regression objective

Logistic regression learning algorithm

LR-Gradient-Descent($\theta_{\text{init}}, \theta_{0,\text{init}}, \eta, \epsilon$)

  Initialize $\theta^{(0)} = \theta_{\text{init}}$

  Initialize $\theta_0^{(0)} = \theta_{0,\text{init}}$

  Initialize t = 0

  repeat

    t = t + 1

Exactly gradient descent with *f* given by logistic regression objective

Logistic regression learning algorithm

LR-Gradient-Descent($\theta_{\text{init}}, \theta_{0,\text{init}}, \eta, \epsilon$)

Initialize $\theta^{(0)} = \theta_{\text{init}}$

Initialize $\theta_0^{(0)} = \theta_{0,\text{init}}$

Initialize t = 0

repeat

t = t + 1

$$\theta^{(t)} = \theta^{(t-1)} - \eta\left\{ \frac{1}{n} \sum_{i=1}^{n} \left[\sigma(\theta^{(t-1)\top} x^{(i)} + \theta_0^{(t-1)}) - y^{(i)}\right] x^{(i)} + 2\lambda\theta^{(t-1)} \right\}$$

$$\theta_0^{(t)} = \theta_0^{(t-1)} - \eta\left\{ \frac{1}{n} \sum_{i=1}^{n} \left[\sigma(\theta^{(t-1)\top} x^{(i)} + \theta_0^{(t-1)}) - y^{(i)}\right] \right\}$$

Exactly gradient descent with *f* given by logistic regression objective

Logistic regression learning algorithm

LR-Gradient-Descent($\theta_{\text{init}}, \theta_{0,\text{init}}, \eta, \epsilon$)

Initialize $\theta^{(0)} = \theta_{\text{init}}$

Initialize $\theta_0^{(0)} = \theta_{0,\text{init}}$

Initialize t = 0

repeat

t = t + 1

$$\theta^{(t)} = \theta^{(t-1)} - \eta\left\{ \frac{1}{n} \sum_{i=1}^{n} \left[\sigma(\theta^{(t-1)\top} x^{(i)} + \theta_0^{(t-1)}) - y^{(i)}\right] x^{(i)} + 2\lambda\theta^{(t-1)} \right\}$$

$$\theta_0^{(t)} = \theta_0^{(t-1)} - \eta\left\{ \frac{1}{n} \sum_{i=1}^{n} \left[\sigma(\theta^{(t-1)\top} x^{(i)} + \theta_0^{(t-1)}) - y^{(i)}\right] \right\}$$

until $|J_{\text{lr}}(\theta^{(t)}, \theta_0^{(t)}) - J_{\text{lr}}(\theta^{(t-1)}, \theta_0^{(t-1)})| < \epsilon$

Exactly gradient descent with *f* given by logistic regression objective

Logistic regression learning algorithm

LR-Gradient-Descent($\theta_{\text{init}}, \theta_{0,\text{init}}, \eta, \epsilon$)

Initialize $\theta^{(0)} = \theta_{\text{init}}$

Initialize $\theta_0^{(0)} = \theta_{0,\text{init}}$

Initialize t = 0

repeat

t = t + 1

$$\theta^{(t)} = \theta^{(t-1)} - \eta\left\{ \frac{1}{n} \sum_{i=1}^{n} \left[\sigma(\theta^{(t-1)\top} x^{(i)} + \theta_0^{(t-1)}) - y^{(i)}\right] x^{(i)} + 2\lambda\theta^{(t-1)} \right\}$$

$$\theta_0^{(t)} = \theta_0^{(t-1)} - \eta\left\{ \frac{1}{n} \sum_{i=1}^{n} \left[\sigma(\theta^{(t-1)\top} x^{(i)} + \theta_0^{(t-1)}) - y^{(i)}\right] \right\}$$

until $|J_{\text{lr}}(\theta^{(t)}, \theta_0^{(t)}) - J_{\text{lr}}(\theta^{(t-1)}, \theta_0^{(t-1)})| < \epsilon$

Return $\theta^{(t)}, \theta_0^{(t)}$

Exactly gradient descent with *f* given by logistic regression objective

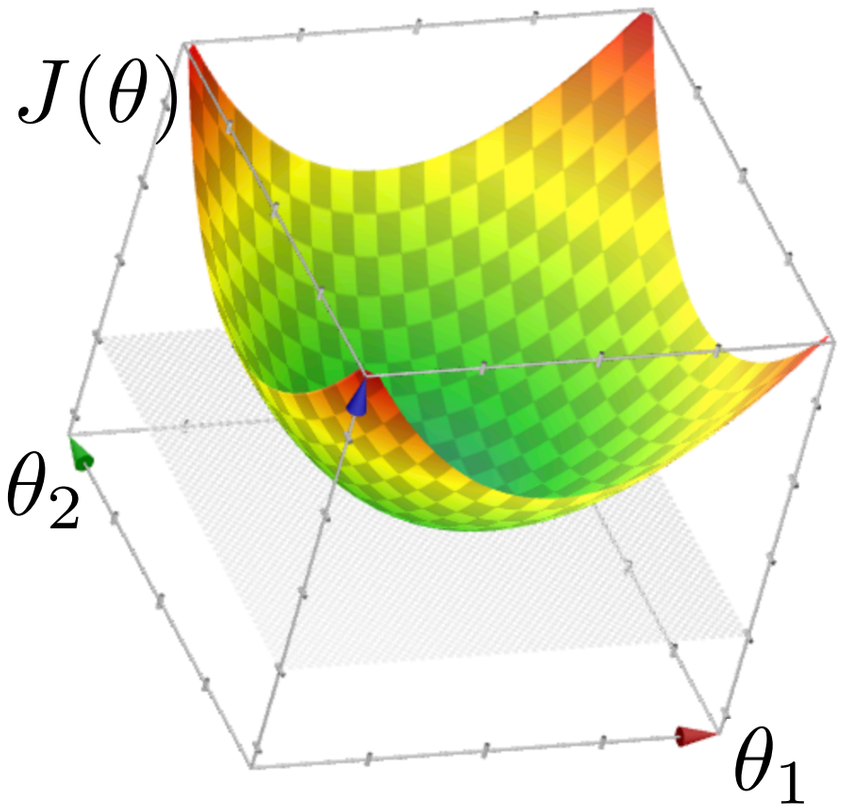
Optimizing linear regression

Optimizing linear regression

- Gradient descent

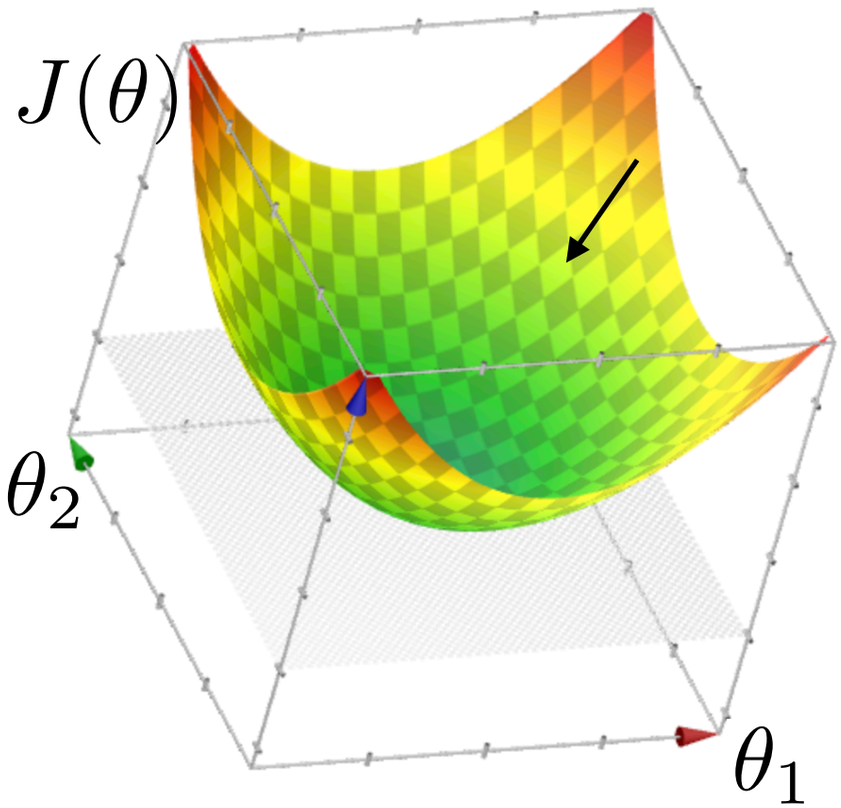
Optimizing linear regression

- Gradient descent



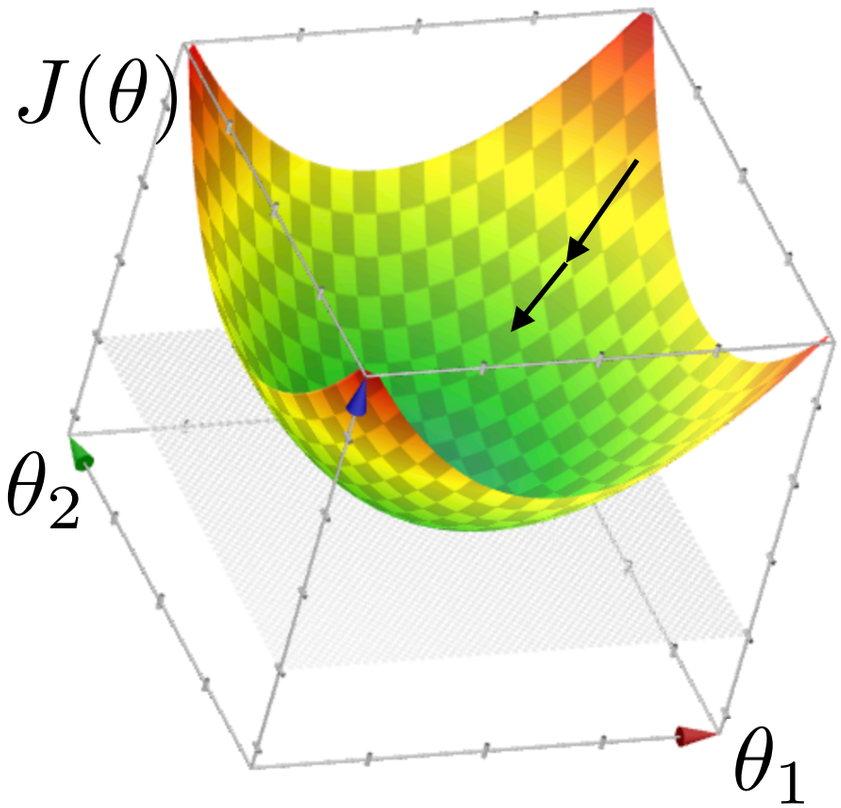
Optimizing linear regression

- Gradient descent



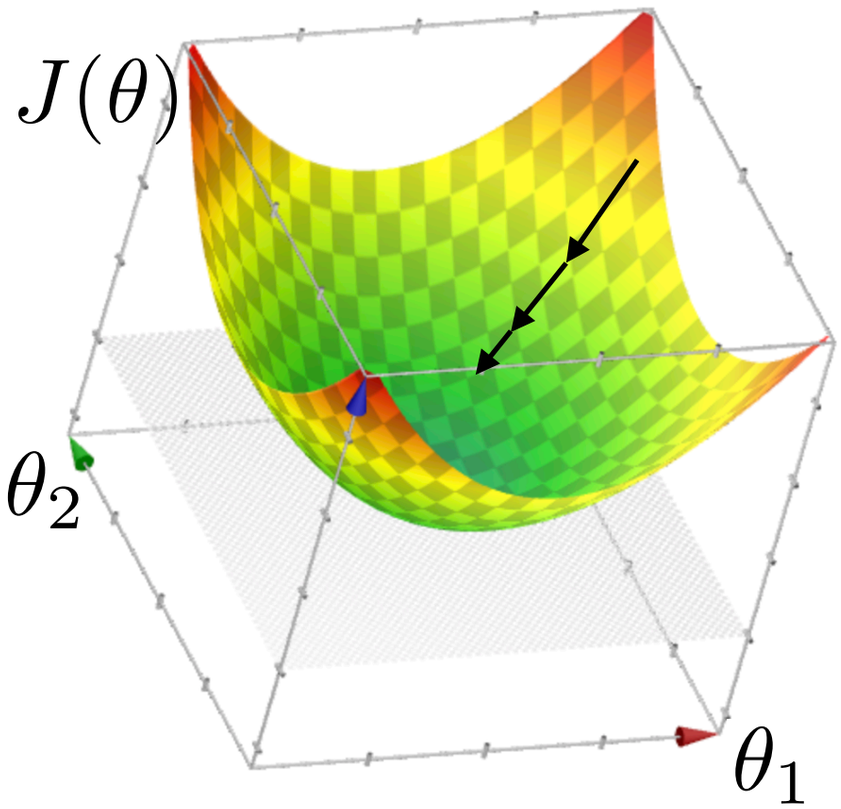
Optimizing linear regression

- Gradient descent



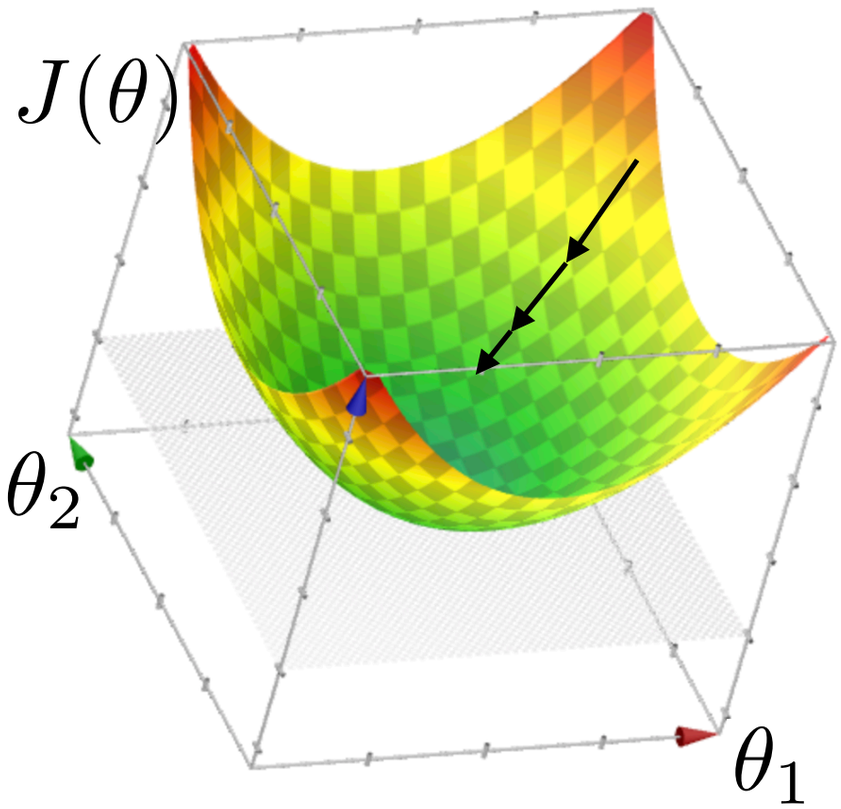
Optimizing linear regression

- Gradient descent



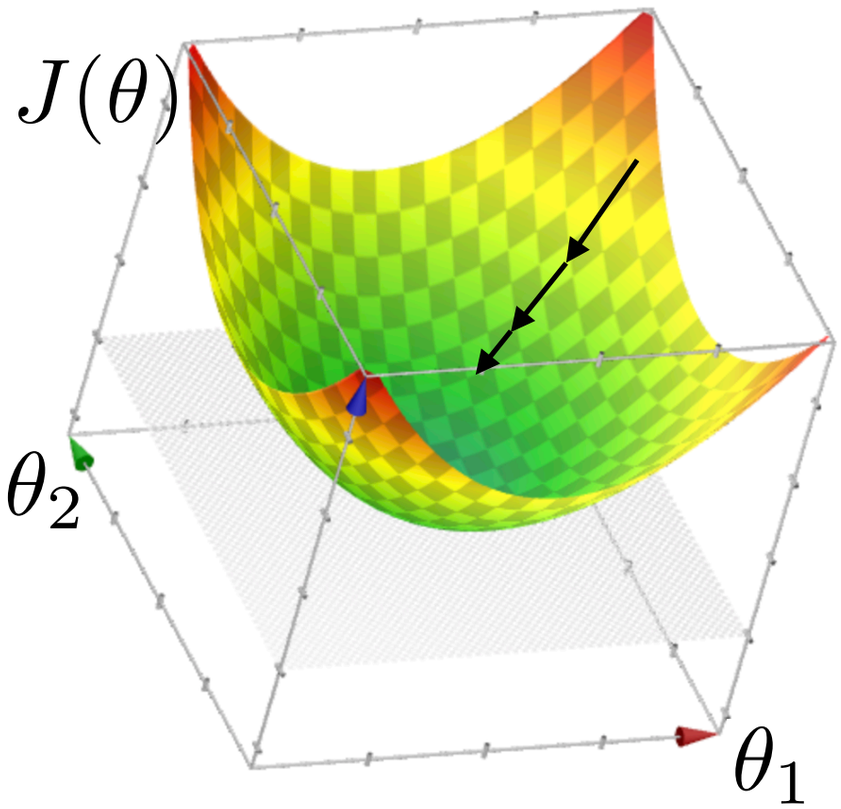
Optimizing linear regression

- Gradient descent



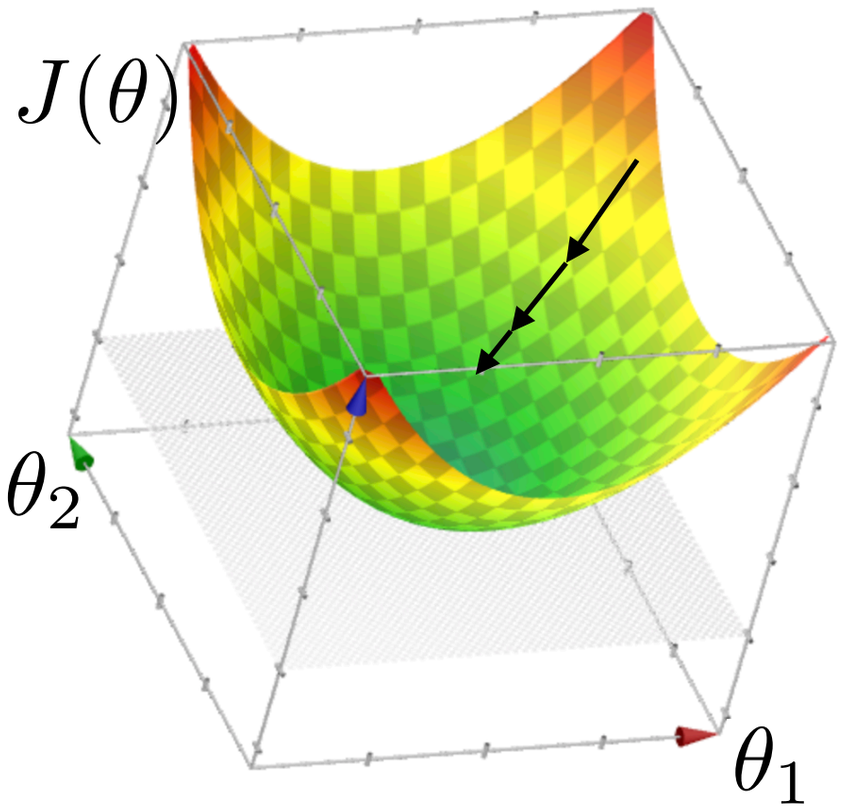
Optimizing linear regression

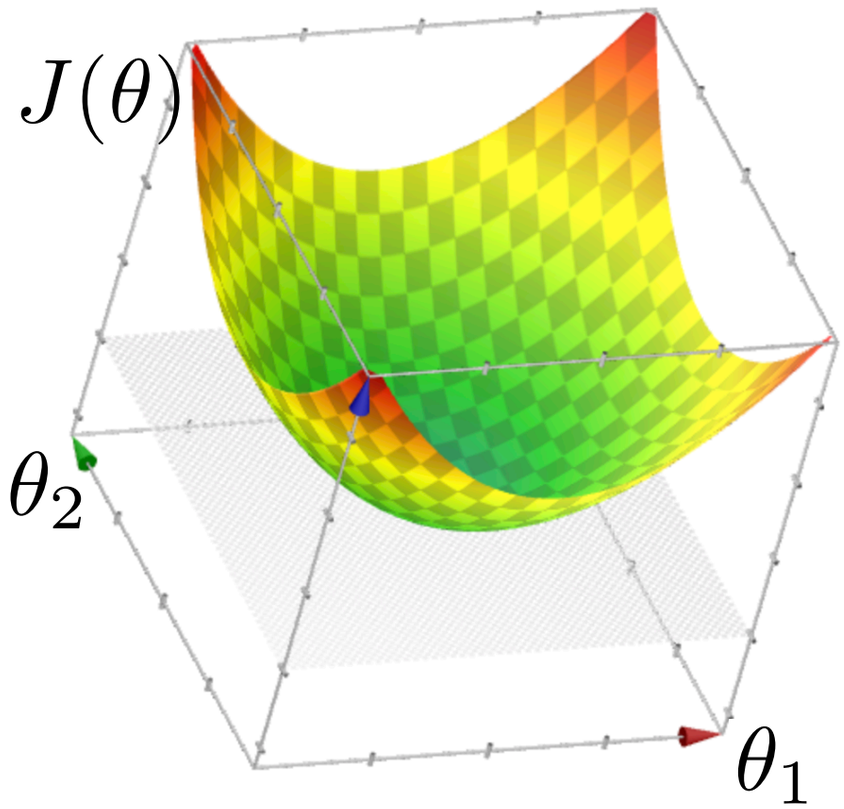
- Gradient descent vs. analytical/closed-form/direct solution



Optimizing linear regression

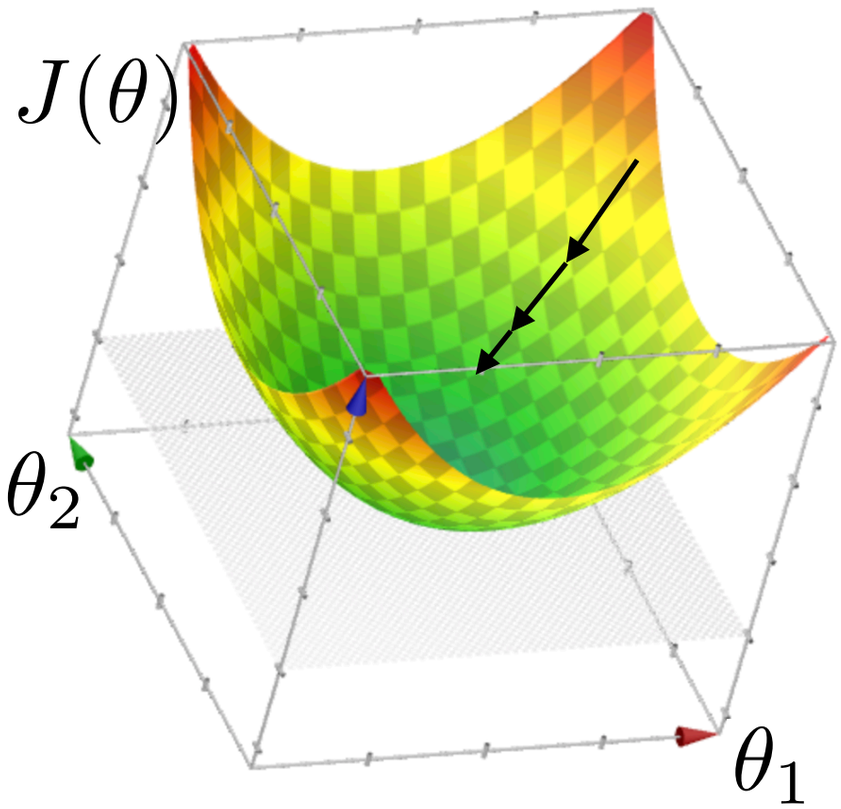
- Gradient descent vs. analytical/closed-form/direct solution

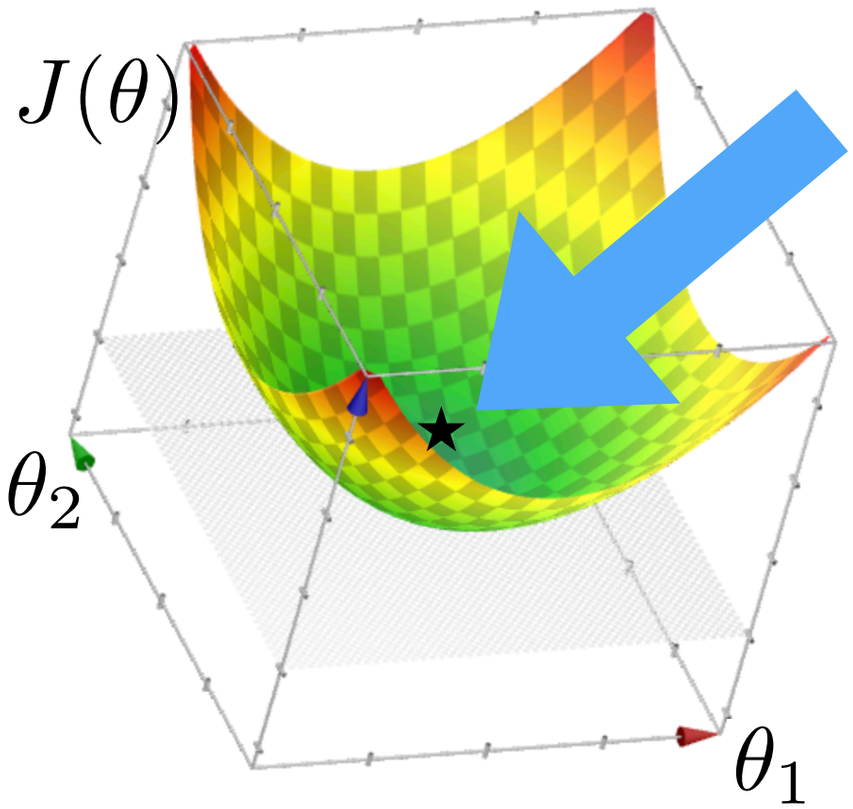




Optimizing linear regression

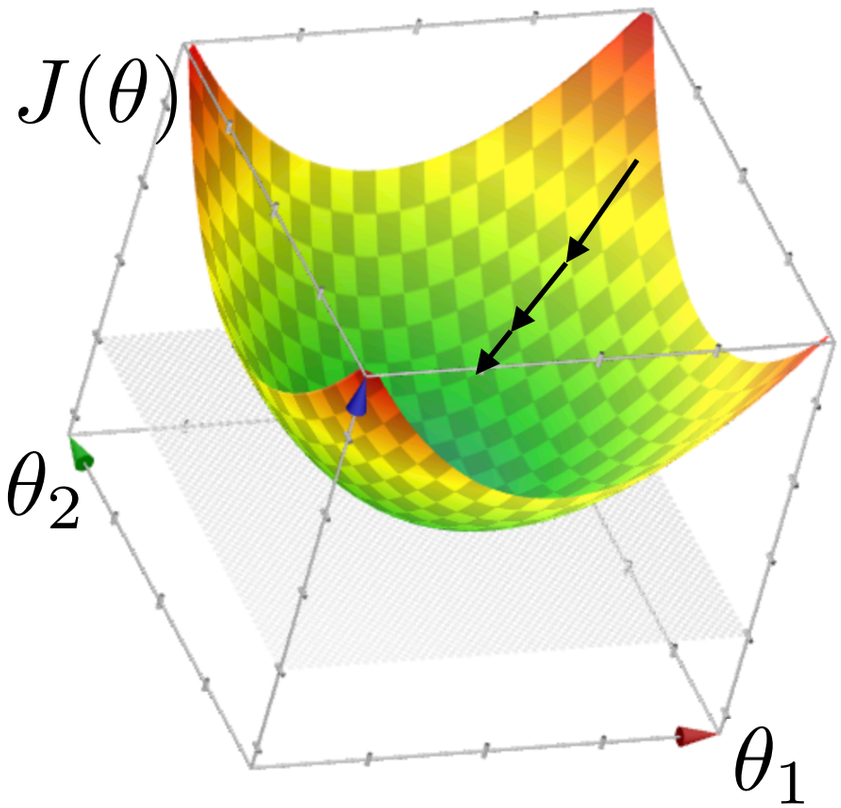
- Gradient descent vs. analytical/closed-form/direct solution

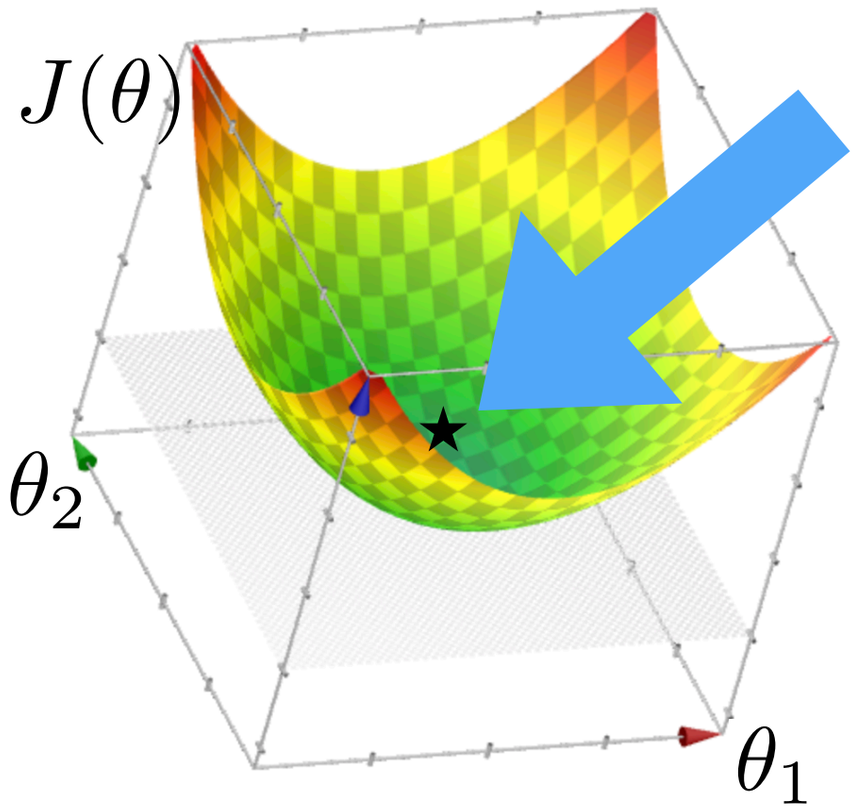




Optimizing linear regression

- Gradient descent vs. analytical/closed-form/direct solution

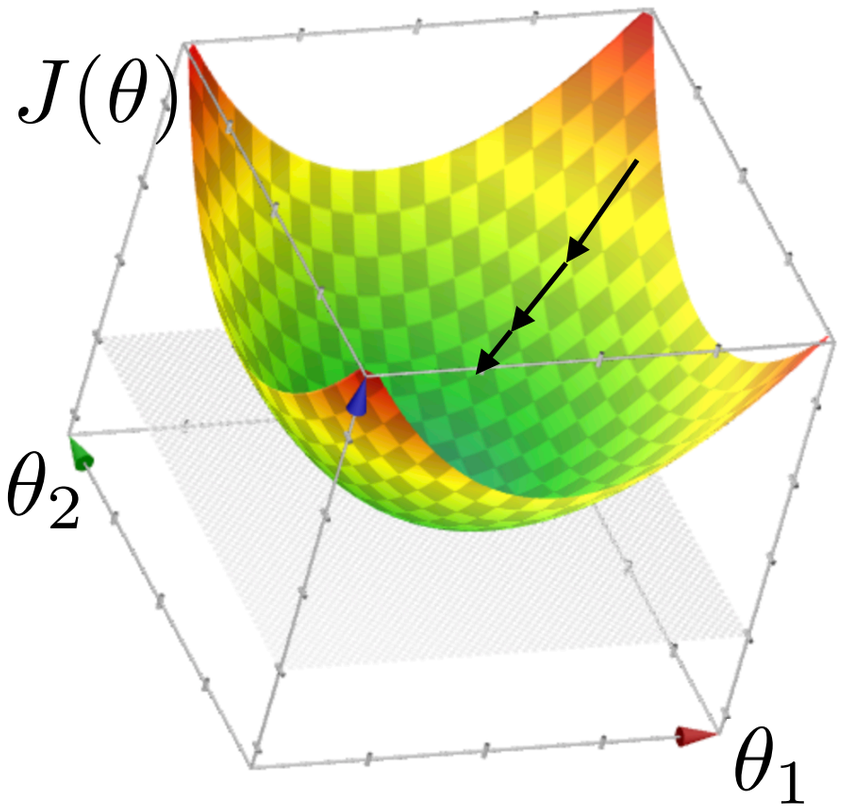


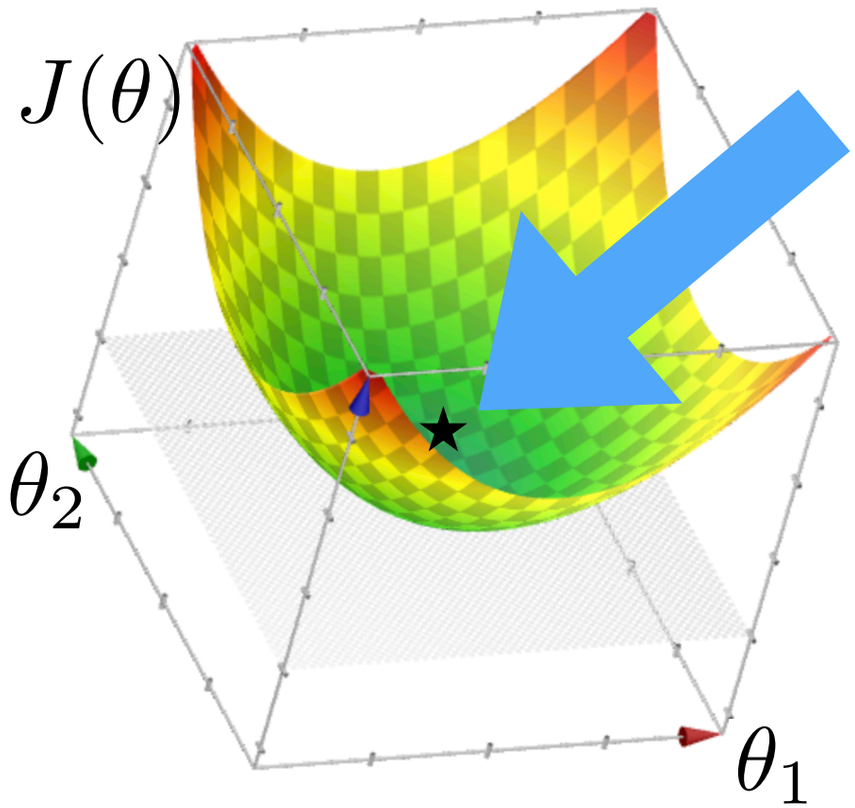


- Accuracy doesn't mean anything without running time

Optimizing linear regression

- Gradient descent vs. analytical/closed-form/direct solution

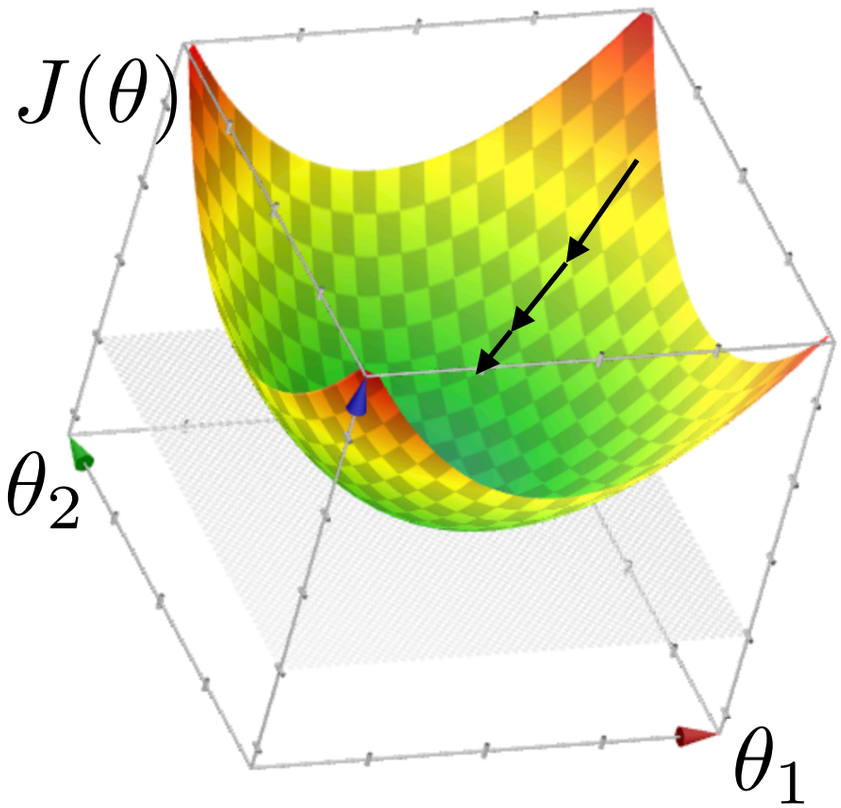


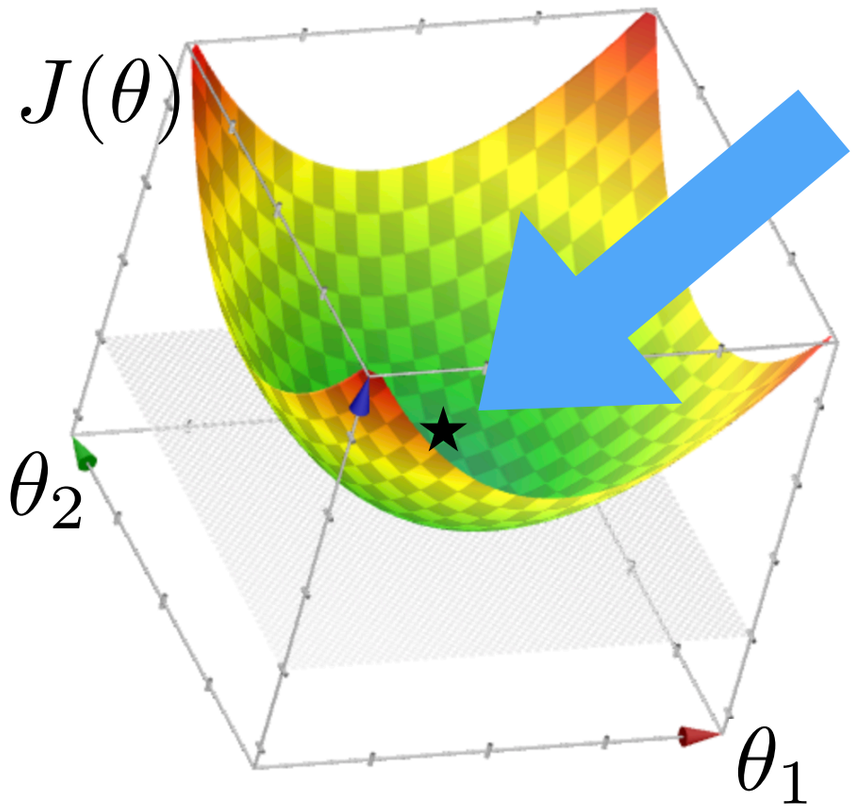


- Accuracy doesn't mean anything without running time
- Running time doesn't mean anything without accuracy

Optimizing linear regression

- Gradient descent vs. analytical/closed-form/direct solution

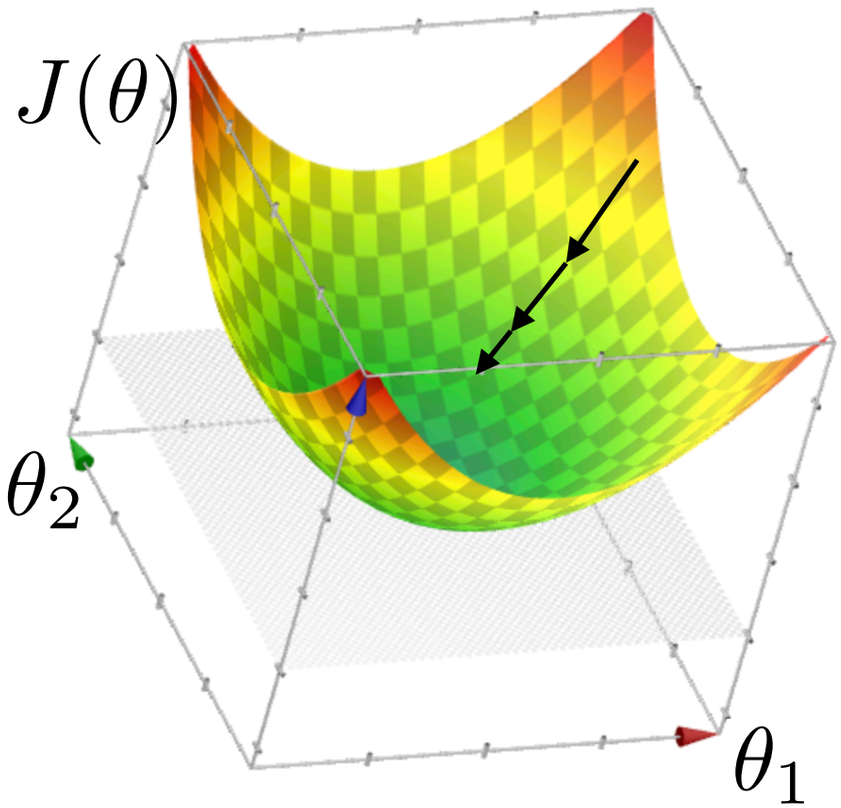


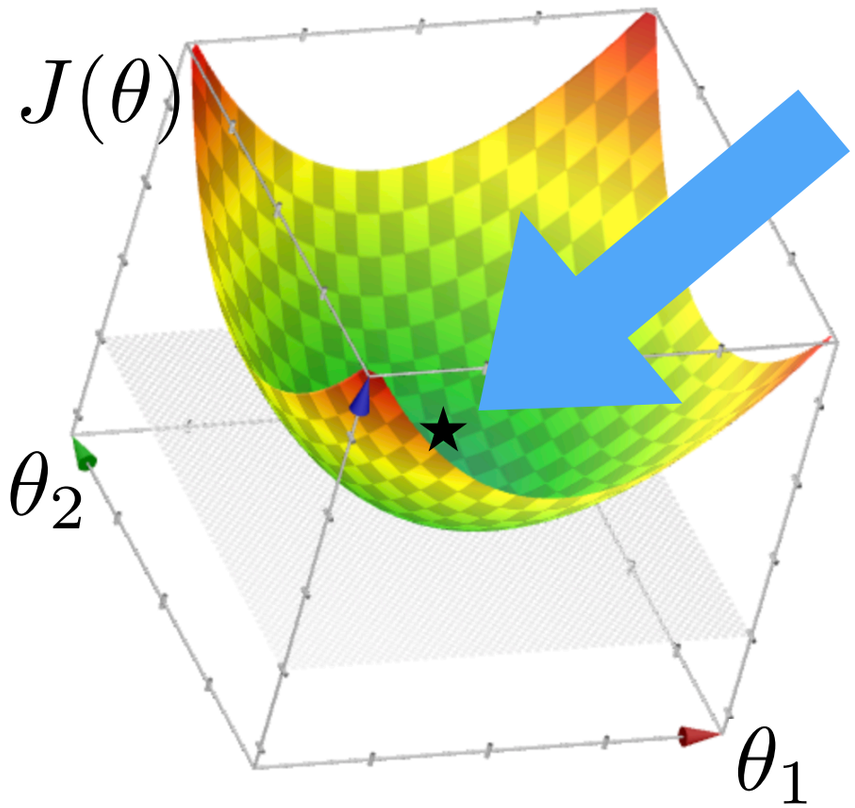


- Accuracy doesn't mean anything without running time
- Running time doesn't mean anything without accuracy
- Need to measure accuracy for the running time we have

Optimizing linear regression

- Gradient descent vs. analytical/closed-form/direct solution



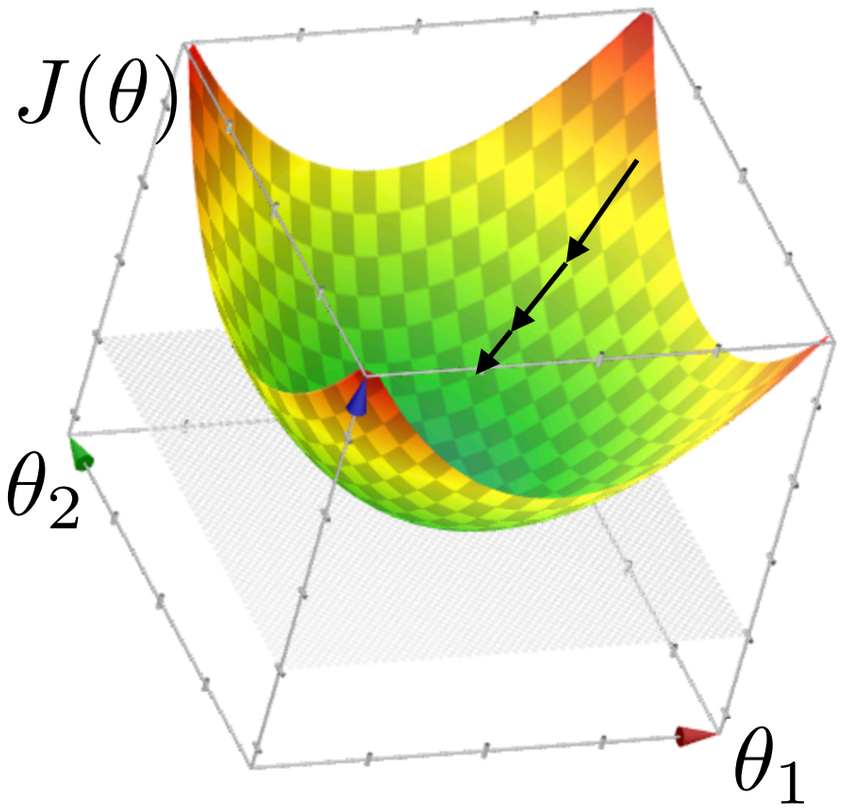


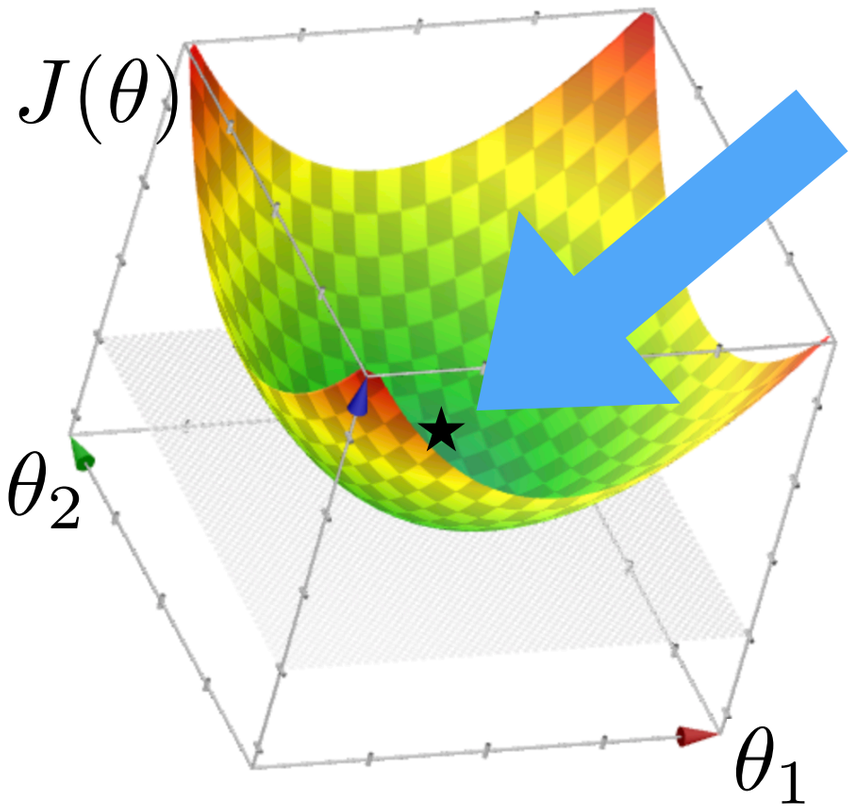
- Accuracy doesn't mean anything without running time
- Running time doesn't mean anything without accuracy
- Need to measure accuracy for the running time we have

$$\theta = (\tilde{X}^\top \tilde{X} + n\lambda I)^{-1} \tilde{X}^\top \tilde{Y}$$

Optimizing linear regression

- Gradient descent vs. analytical/closed-form/direct solution



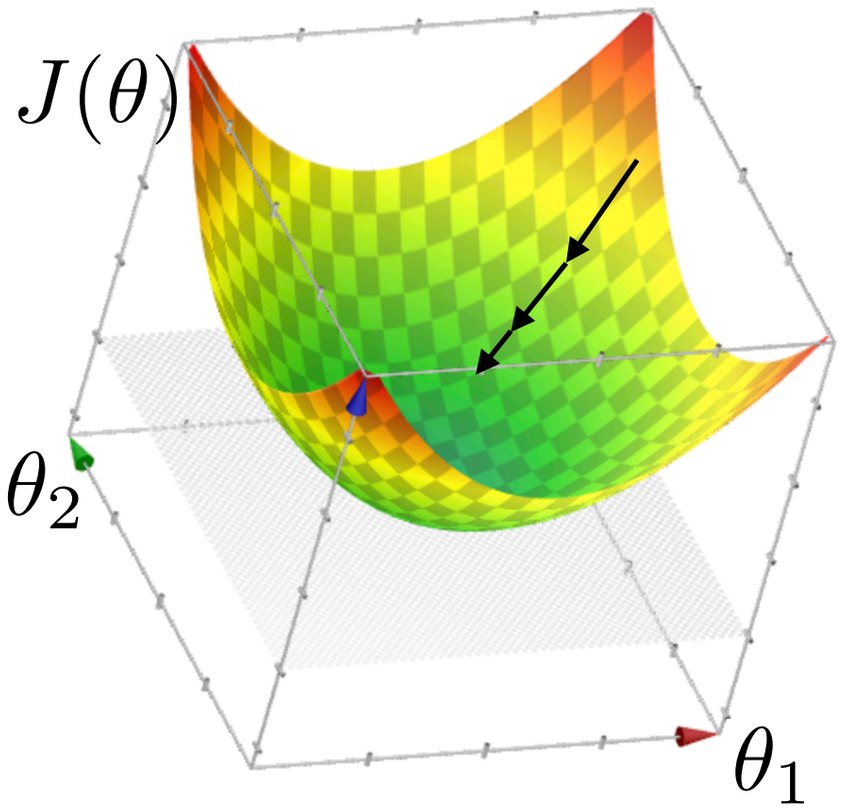


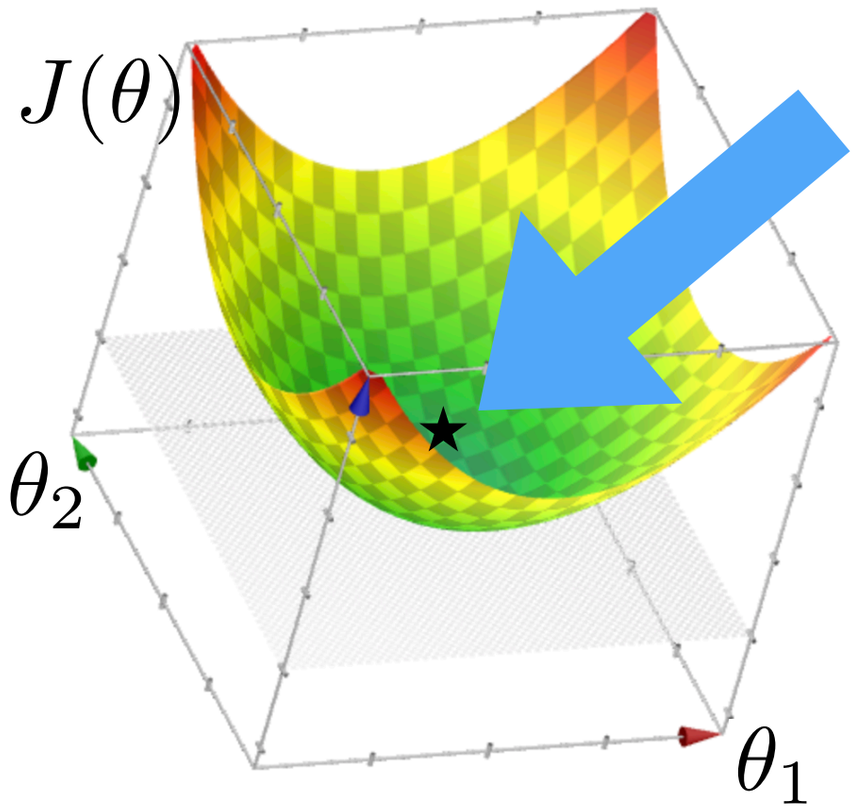
- Accuracy doesn't mean anything without running time
- Running time doesn't mean anything without accuracy
- Need to measure accuracy for the running time we have

$$\theta = \underbrace{(\tilde{X}^\top \tilde{X} + n\lambda I)}_{\text{dxd}}{}^{-1} \tilde{X}^\top \tilde{Y}$$

Optimizing linear regression

- Gradient descent vs. analytical/closed-form/direct solution





- Accuracy doesn't mean anything without running time
- Running time doesn't mean anything without accuracy
- Need to measure accuracy for the running time we have

$$\theta = \underbrace{(\tilde{X}^\top \tilde{X} + n\lambda I)}_{\text{dxd}}{}^{-1} \tilde{X}^\top \tilde{Y}$$

Matrix inversion: $O(d^3)$

Gradient descent for linear regression

Gradient descent for linear regression

`LinearRegression-Gradient-Descent(` $\theta_{\text{init}}, \theta_{0,\text{init}}, \eta, T$ `)`

Gradient descent for linear regression

`LinearRegression-Gradient-Descent(` $\theta_{\text{init}}, \theta_{0,\text{init}}, \eta, T$ `)`

Exactly gradient descent with *f* given by linear regression objective

Gradient descent for linear regression

`LinearRegression-Gradient-Descent(` $\theta_{\text{init}}, \theta_{0,\text{init}}, \eta, T$ `)`

`Initialize` $\theta^{(0)} = \theta_{\text{init}}$

`Initialize` $\theta_0^{(0)} = \theta_{0,\text{init}}$

Exactly gradient descent with *f* given by linear regression objective

Gradient descent for linear regression

LinearRegression-Gradient-Descent($\theta_{\text{init}}, \theta_{0,\text{init}}, \eta, T$)

  Initialize $\theta^{(0)} = \theta_{\text{init}}$

  Initialize $\theta_0^{(0)} = \theta_{0,\text{init}}$

  for t = 1 **to** T

Exactly gradient descent with *f* given by linear regression objective

Gradient descent for linear regression

LinearRegression-Gradient-Descent($\theta_{\text{init}}, \theta_{0,\text{init}}, \eta, T$)

Initialize $\theta^{(0)} = \theta_{\text{init}}$

Initialize $\theta_0^{(0)} = \theta_{0,\text{init}}$

for t = 1 **to** T

Exactly gradient descent with *f* given by linear regression objective

$$\theta^{(t)} = \theta^{(t-1)} - \eta \left\{ \frac{2}{n} \sum_{i=1}^{n} \left[\theta^{(t-1)\top} x^{(i)} + \theta_0^{(t-1)} - y^{(i)} \right] x^{(i)} + 2\lambda\theta^{(t-1)} \right\}$$

Gradient descent for linear regression

LinearRegression-Gradient-Descent($\theta_{\text{init}}, \theta_{0,\text{init}}, \eta, T$)

Initialize $\theta^{(0)} = \theta_{\text{init}}$

Initialize $\theta_0^{(0)} = \theta_{0,\text{init}}$

for t = 1 **to** T

Exactly gradient descent with *f* given by linear regression objective

$$\theta^{(t)} = \theta^{(t-1)} - \eta\left\{ \frac{2}{n}\sum_{i=1}^{n} \left[\theta^{(t-1)\top} x^{(i)} + \theta_0^{(t-1)} - y^{(i)}\right] x^{(i)} + 2\lambda\theta^{(t-1)} \right\}$$

$$\theta_0^{(t)} = \theta_0^{(t-1)} - \eta\left\{ \frac{2}{n}\sum_{i=1}^{n} \left[\theta^{(t-1)\top} x^{(i)} + \theta_0^{(t-1)} - y^{(i)}\right] \right\}$$

Gradient descent for linear regression

LinearRegression-Gradient-Descent($\theta_{\text{init}}, \theta_{0,\text{init}}, \eta, T$)

Initialize $\theta^{(0)} = \theta_{\text{init}}$

Initialize $\theta_0^{(0)} = \theta_{0,\text{init}}$

for t = 1 **to** T

Exactly gradient descent with *f* given by linear regression objective

$$\theta^{(t)} = \theta^{(t-1)} - \eta\left\{ \frac{2}{n} \sum_{i=1}^{n} \left[\theta^{(t-1)\top} x^{(i)} + \theta_0^{(t-1)} - y^{(i)}\right] x^{(i)} + 2\lambda\theta^{(t-1)} \right\}$$

$$\theta_0^{(t)} = \theta_0^{(t-1)} - \eta\left\{ \frac{2}{n} \sum_{i=1}^{n} \left[\theta^{(t-1)\top} x^{(i)} + \theta_0^{(t-1)} - y^{(i)}\right] \right\}$$

Return $\theta^{(t)}, \theta_0^{(t)}$

Stochastic gradient descent

Stochastic gradient descent

- Linear regression objective with $\lambda = 0$:

Stochastic gradient descent

- Linear regression objective with $\lambda = 0$:

$$J_{\text{linreg}}(\theta, \theta_0) = \frac{1}{n} \sum_{i=1}^{n} (\theta^\top x^{(i)} + \theta_0 - y^{(i)})^2$$

Stochastic gradient descent

- Linear regression objective with $\lambda = 0$:

$$J_{\text{linreg}}(\Theta) = J_{\text{linreg}}(\theta, \theta_0) = \frac{1}{n} \sum_{i=1}^{n} (\theta^\top x^{(i)} + \theta_0 - y^{(i)})^2$$

Stochastic gradient descent

- Linear regression objective with $\lambda = 0$:

$$J_{\text{linreg}}(\Theta) = J_{\text{linreg}}(\theta, \theta_0) = \frac{1}{n} \sum_{i=1}^{n} (\theta^\top x^{(i)} + \theta_0 - y^{(i)})^2$$

- A common machine learning objective:

$$f(\Theta) = \frac{1}{n} \sum_{i=1}^{n} f_i(\Theta)$$

Stochastic gradient descent

- Linear regression objective with $\lambda = 0$:

$$J_{\text{linreg}}(\Theta) = J_{\text{linreg}}(\theta, \theta_0) = \frac{1}{n} \sum_{i=1}^{n} (\theta^\top x^{(i)} + \theta_0 - y^{(i)})^2$$

- A common machine learning objective:

$$f(\Theta) = \frac{1}{n} \sum_{i=1}^{n} f_i(\Theta)$$

- Also recall logistic regression with $\lambda = 0$

Stochastic gradient descent

- Linear regression objective with $\lambda = 0$:

$$J_{\text{linreg}}(\Theta) = J_{\text{linreg}}(\theta, \theta_0) = \frac{1}{n} \sum_{i=1}^{n} (\theta^\top x^{(i)} + \theta_0 - y^{(i)})^2$$

- A common machine learning objective:

$$f(\Theta) = \frac{1}{n} \sum_{i=1}^{n} f_i(\Theta)$$

- Also recall logistic regression with $\lambda = 0$

`Stochastic-Gradient-Descent`$(\Theta_{\text{init}}, \eta, T)$

Stochastic gradient descent

- Linear regression objective with $\lambda = 0$:

$$J_{\text{linreg}}(\Theta) = J_{\text{linreg}}(\theta, \theta_0) = \frac{1}{n} \sum_{i=1}^{n} (\theta^\top x^{(i)} + \theta_0 - y^{(i)})^2$$

- A common machine learning objective:

$$f(\Theta) = \frac{1}{n} \sum_{i=1}^{n} f_i(\Theta)$$

- Also recall logistic regression with $\lambda = 0$

Stochastic-Gradient-Descent($\Theta_{\text{init}}, \eta, T$)

  Initialize $\Theta^{(0)} = \Theta_{\text{init}}$

Stochastic gradient descent

- Linear regression objective with $\lambda = 0$:

$$J_{\text{linreg}}(\Theta) = J_{\text{linreg}}(\theta, \theta_0) = \frac{1}{n} \sum_{i=1}^{n} (\theta^\top x^{(i)} + \theta_0 - y^{(i)})^2$$

- A common machine learning objective:

$$f(\Theta) = \frac{1}{n} \sum_{i=1}^{n} f_i(\Theta)$$

- Also recall logistic regression with $\lambda = 0$

Stochastic-Gradient-Descent($\Theta_{\text{init}}, \eta, T$)

  Initialize $\Theta^{(0)} = \Theta_{\text{init}}$

  for t = 1 **to** T

Stochastic gradient descent

- Linear regression objective with $\lambda = 0$:

$$J_{\text{linreg}}(\Theta) = J_{\text{linreg}}(\theta, \theta_0) = \frac{1}{n} \sum_{i=1}^{n} (\theta^\top x^{(i)} + \theta_0 - y^{(i)})^2$$

- A common machine learning objective:

$$f(\Theta) = \frac{1}{n} \sum_{i=1}^{n} f_i(\Theta)$$

- Also recall logistic regression with $\lambda = 0$

Stochastic-Gradient-Descent($\Theta_{\text{init}}, \eta, T$)
  Initialize $\Theta^{(0)} = \Theta_{\text{init}}$
  for t = 1 **to** T
    randomly select i from {1,…,n} (with equal probability)

Stochastic gradient descent

- Linear regression objective with $\lambda = 0$:

$$J_{\text{linreg}}(\Theta) = J_{\text{linreg}}(\theta, \theta_0) = \frac{1}{n} \sum_{i=1}^{n} (\theta^\top x^{(i)} + \theta_0 - y^{(i)})^2$$

- A common machine learning objective:

$$f(\Theta) = \frac{1}{n} \sum_{i=1}^{n} f_i(\Theta)$$

- Also recall logistic regression with $\lambda = 0$

Stochastic-Gradient-Descent($\Theta_{\text{init}}, \eta, T$)

  Initialize $\Theta^{(0)} = \Theta_{\text{init}}$

  for t = 1 **to** T

    randomly select i from {1,…,n} (with equal probability)

    $\Theta^{(t)} = \Theta^{(t-1)} - \eta(t) \nabla_\Theta f_i(\Theta^{(t-1)})$

Stochastic gradient descent

- Linear regression objective with $\lambda = 0$:

$$J_{\text{linreg}}(\Theta) = J_{\text{linreg}}(\theta, \theta_0) = \frac{1}{n}\sum_{i=1}^{n}(\theta^\top x^{(i)} + \theta_0 - y^{(i)})^2$$

- A common machine learning objective:

$$f(\Theta) = \frac{1}{n}\sum_{i=1}^{n} f_i(\Theta)$$

- Also recall logistic regression with $\lambda = 0$

Stochastic-Gradient-Descent($\Theta_{\text{init}}, \eta, T$)

  Initialize $\Theta^{(0)} = \Theta_{\text{init}}$

  for t = 1 **to** T

    randomly select i from {1,…,n} (with equal probability)

    $\Theta^{(t)} = \Theta^{(t-1)} - \eta(t)\nabla_\Theta f_i(\Theta^{(t-1)})$

Stochastic gradient descent

- Linear regression objective with $\lambda = 0$:

$$J_{\text{linreg}}(\Theta) = J_{\text{linreg}}(\theta, \theta_0) = \frac{1}{n} \sum_{i=1}^{n} (\theta^\top x^{(i)} + \theta_0 - y^{(i)})^2$$

- A common machine learning objective:

$$f(\Theta) = \frac{1}{n} \sum_{i=1}^{n} f_i(\Theta)$$

- Also recall logistic regression with $\lambda = 0$

Stochastic-Gradient-Descent($\Theta_{\text{init}}, \eta, T$)

  Initialize $\Theta^{(0)} = \Theta_{\text{init}}$

  for t = 1 **to** T

    randomly select i from {1,…,n} (with equal probability)

    $\Theta^{(t)} = \Theta^{(t-1)} - \eta(t) \nabla_\Theta f_i(\Theta^{(t-1)})$

Stochastic gradient descent

- Linear regression objective with $\lambda = 0$:

$$J_{\text{linreg}}(\Theta) = J_{\text{linreg}}(\theta, \theta_0) = \frac{1}{n} \sum_{i=1}^{n} (\theta^\top x^{(i)} + \theta_0 - y^{(i)})^2$$

- A common machine learning objective:

$$f(\Theta) = \frac{1}{n} \sum_{i=1}^{n} f_i(\Theta)$$

- Also recall logistic regression with $\lambda = 0$

Stochastic-Gradient-Descent($\Theta_{\text{init}}, \eta, T$)

  Initialize $\Theta^{(0)} = \Theta_{\text{init}}$

  for t = 1 **to** T

    randomly select i from {1,…,n} (with equal probability)

    $\Theta^{(t)} = \Theta^{(t-1)} - \eta(t) \nabla_\Theta f_i(\Theta^{(t-1)})$

Stochastic gradient descent

- Linear regression objective with $\lambda = 0$:

$$J_{\text{linreg}}(\Theta) = J_{\text{linreg}}(\theta, \theta_0) = \frac{1}{n}\sum_{i=1}^{n}(\theta^\top x^{(i)} + \theta_0 - y^{(i)})^2$$

- A common machine learning objective:

$$f(\Theta) = \frac{1}{n}\sum_{i=1}^{n} f_i(\Theta)$$

- Also recall logistic regression with $\lambda = 0$

Stochastic-Gradient-Descent($\Theta_{\text{init}}, \eta, T$)

  Initialize $\Theta^{(0)} = \Theta_{\text{init}}$

  for t = 1 **to** T

    randomly select i from {1,…,n} (with equal probability)

    $\Theta^{(t)} = \Theta^{(t-1)} - \eta(t)\nabla_\Theta f_i(\Theta^{(t-1)})$

  Return $\Theta^{(t)}$

Stochastic gradient descent

- Linear regression objective with $\lambda = 0$:

$$J_{\text{linreg}}(\Theta) = J_{\text{linreg}}(\theta, \theta_0) = \frac{1}{n}\sum_{i=1}^{n}(\theta^\top x^{(i)} + \theta_0 - y^{(i)})^2$$

- A common machine learning objective:

$$f(\Theta) = \frac{1}{n}\sum_{i=1}^{n} f_i(\Theta)$$

- Also recall logistic regression with $\lambda = 0$

Stochastic-Gradient-Descent($\Theta_{\text{init}}, \eta, T$)

  Initialize $\Theta^{(0)} = \Theta_{\text{init}}$

  for t = 1 **to** T

    randomly select i from {1,…,n} (with equal probability)

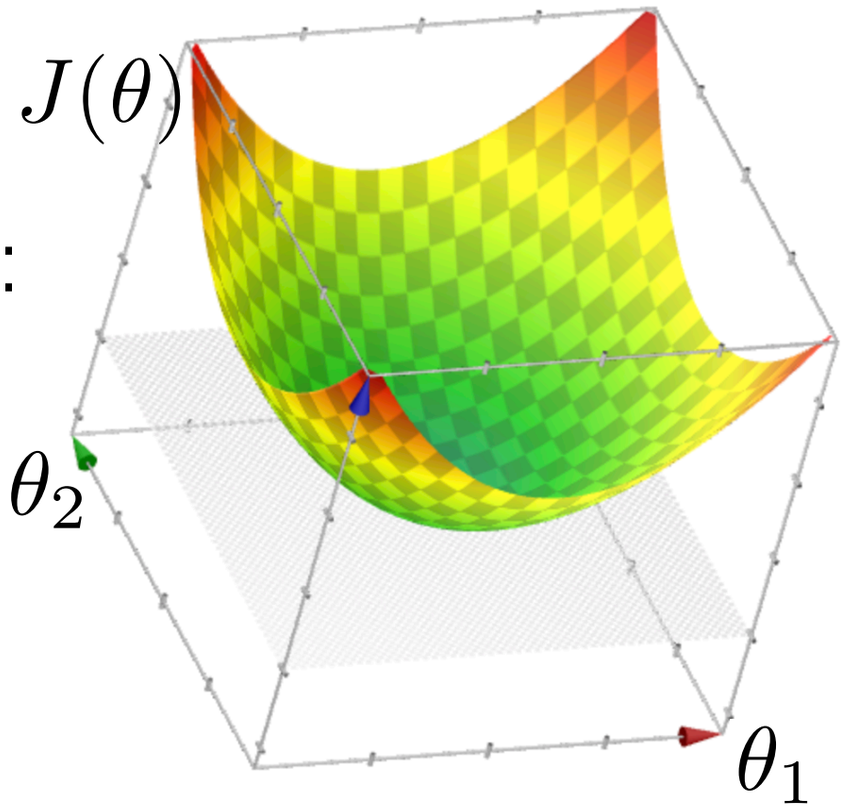
    $\Theta^{(t)} = \Theta^{(t-1)} - \eta(t)\nabla_\Theta f_i(\Theta^{(t-1)})$

  Return $\Theta^{(t)}$

- Commonly used with “minibatches”

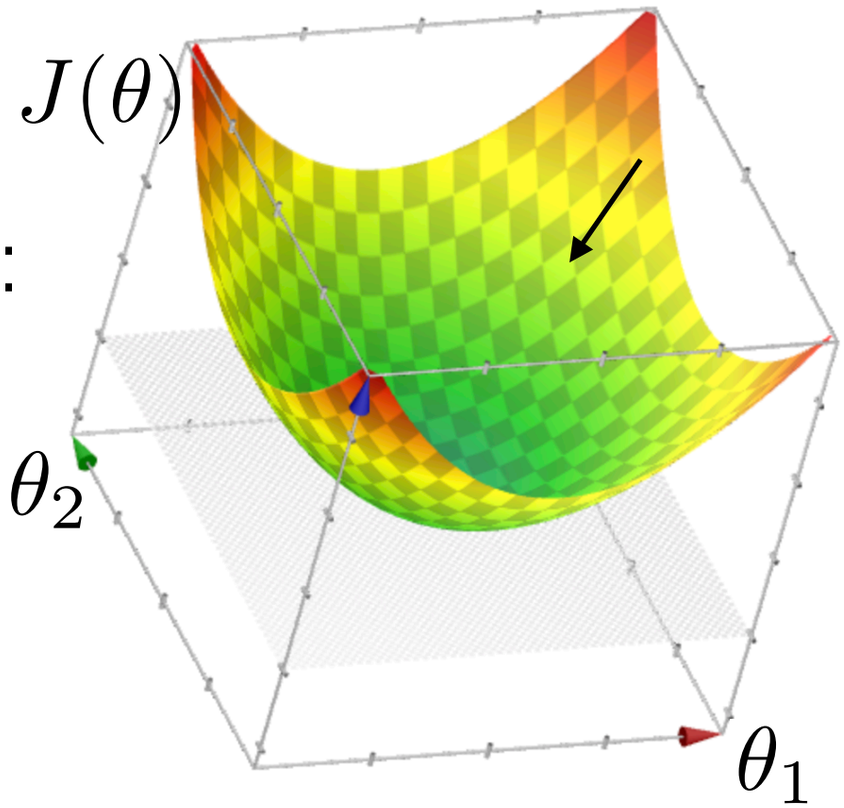
Stochastic gradient descent (SGD) properties

- GD:



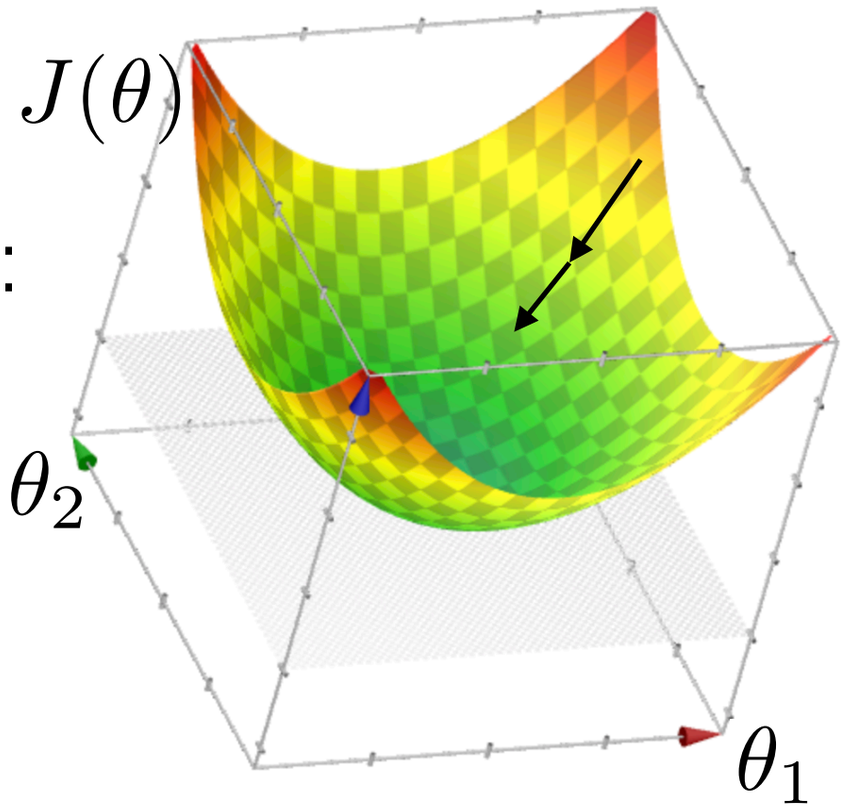
Stochastic gradient descent (SGD) properties

- GD:



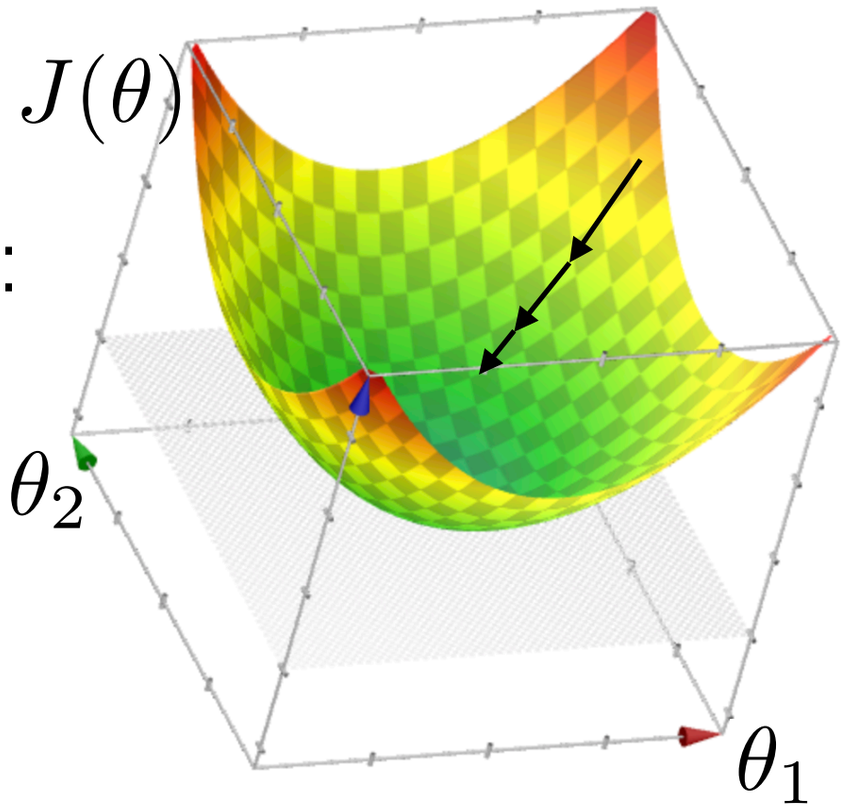
Stochastic gradient descent (SGD) properties

- GD:



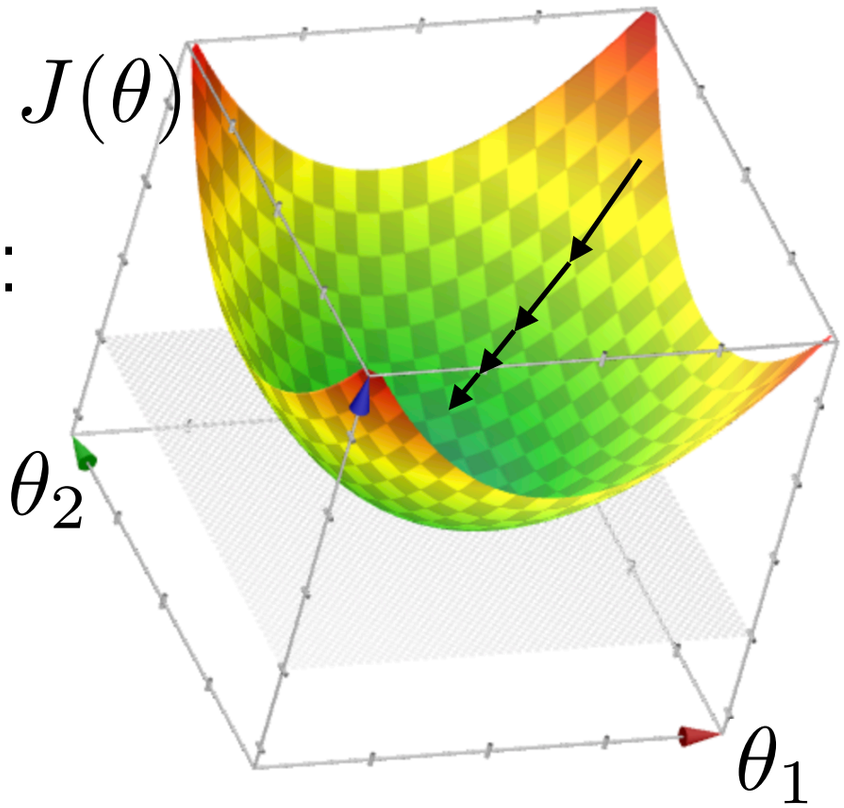
Stochastic gradient descent (SGD) properties

- GD:



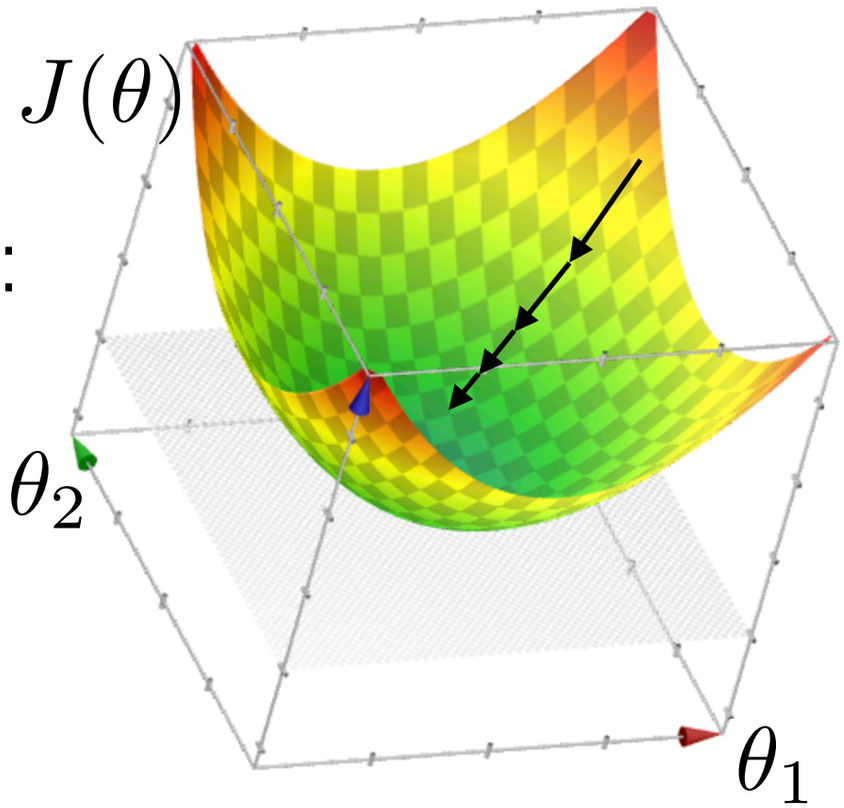
Stochastic gradient descent (SGD) properties

- GD:



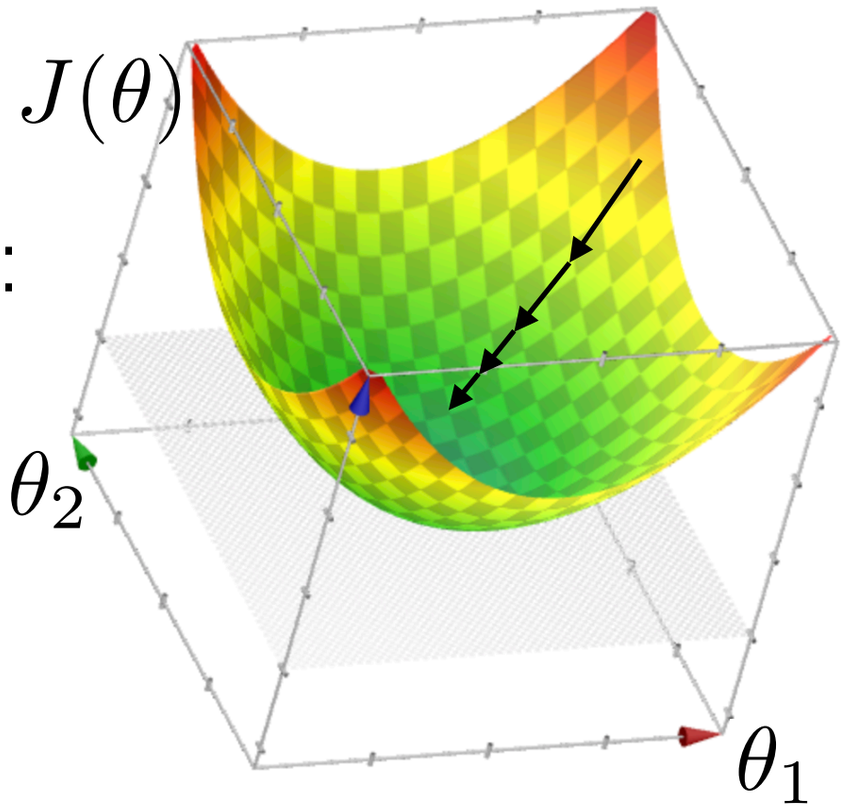
Stochastic gradient descent (SGD) properties

- GD:

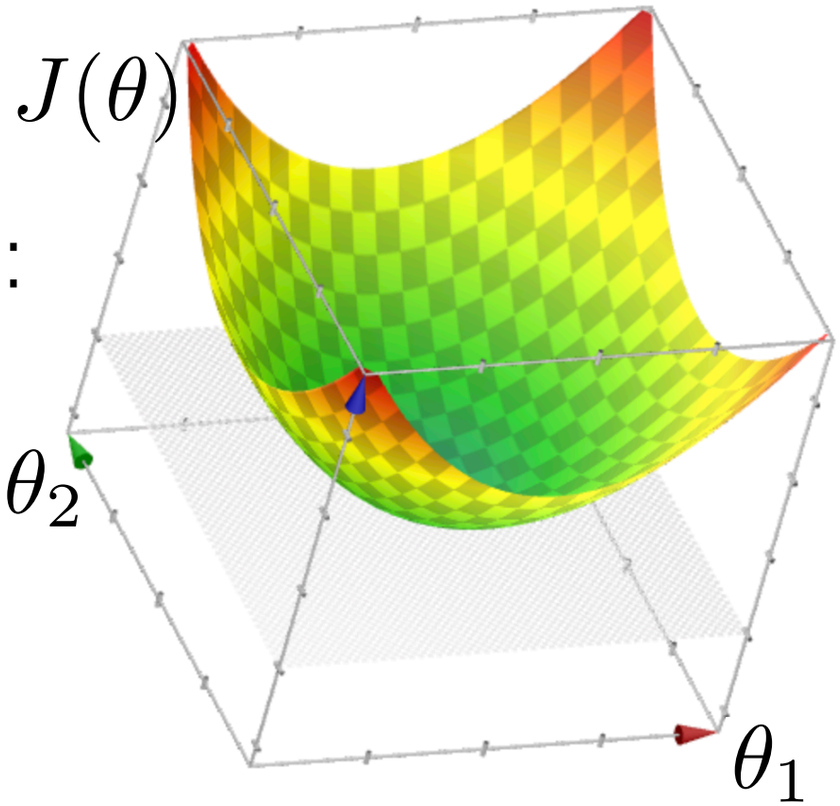


Stochastic gradient descent (SGD) properties

- GD:

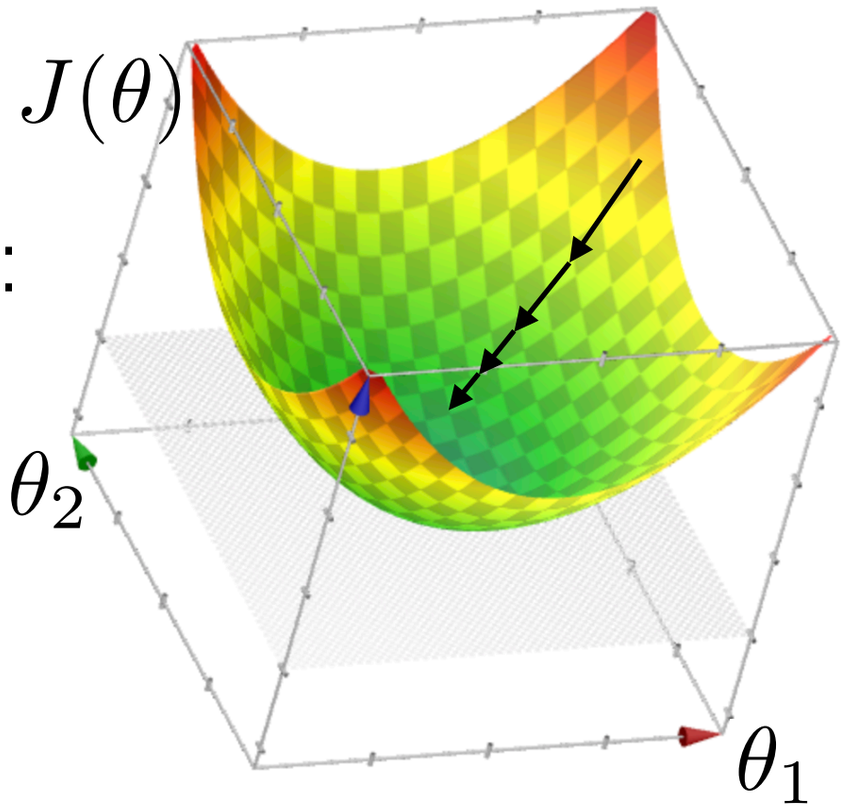


- SGD:

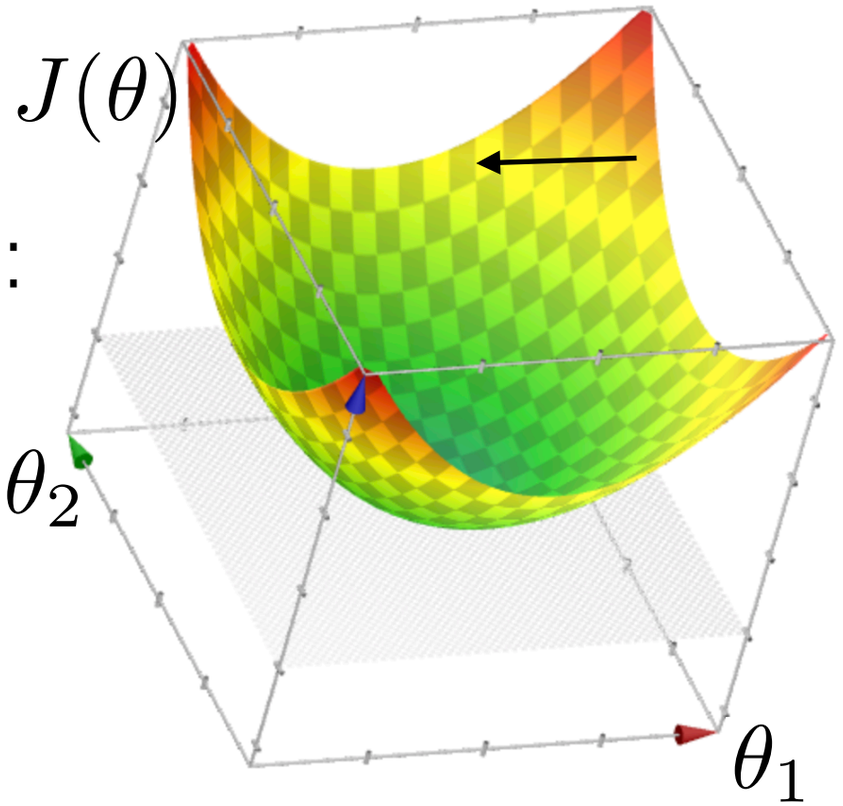


Stochastic gradient descent (SGD) properties

- GD:

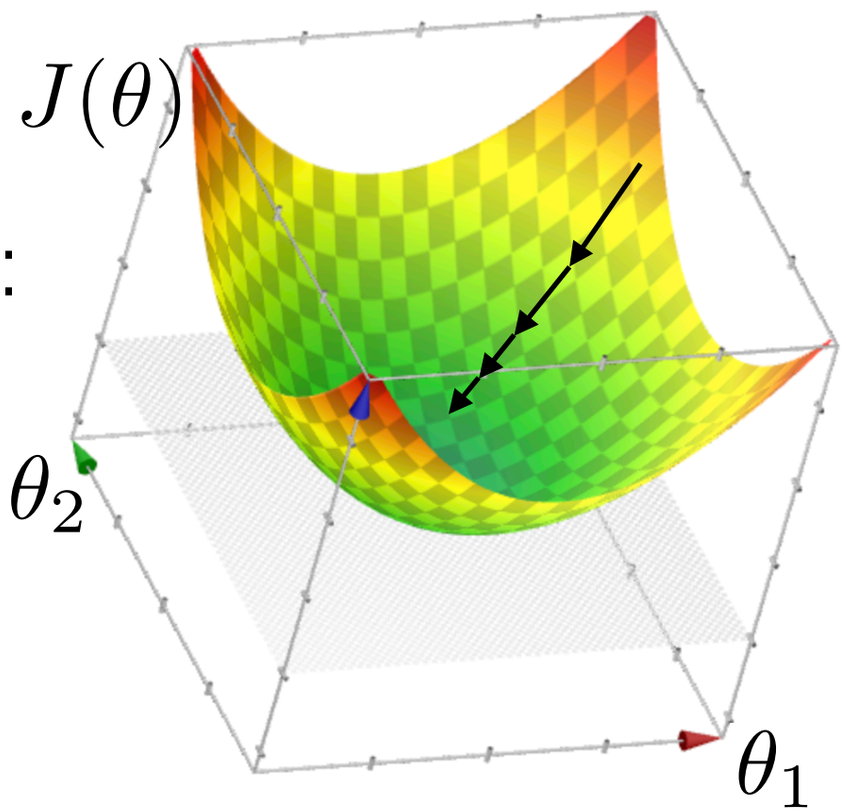


- SGD:

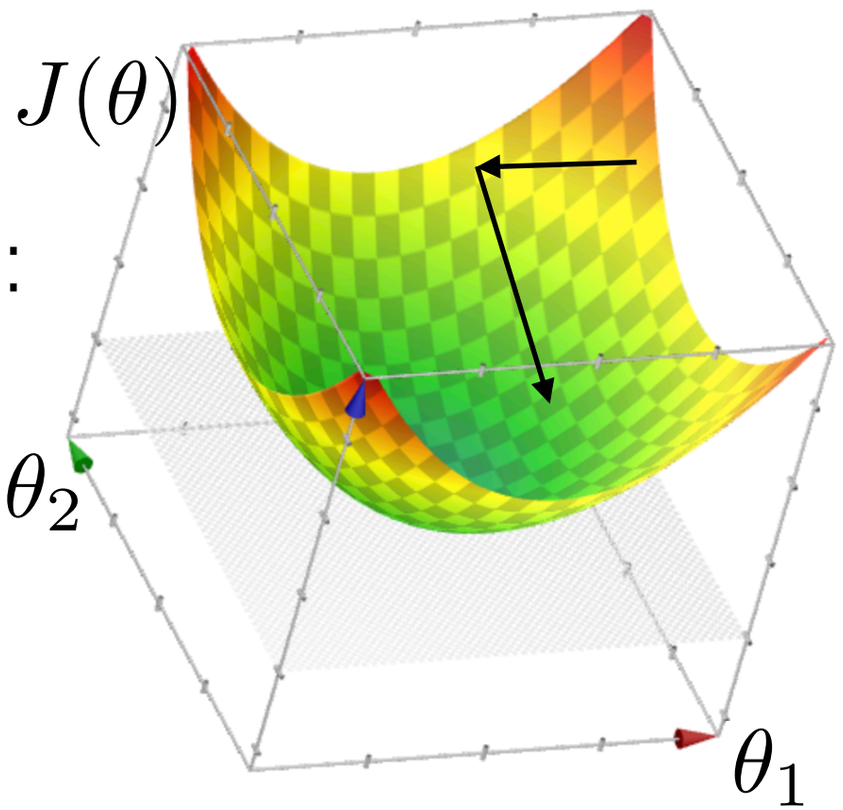


Stochastic gradient descent (SGD) properties

- GD:

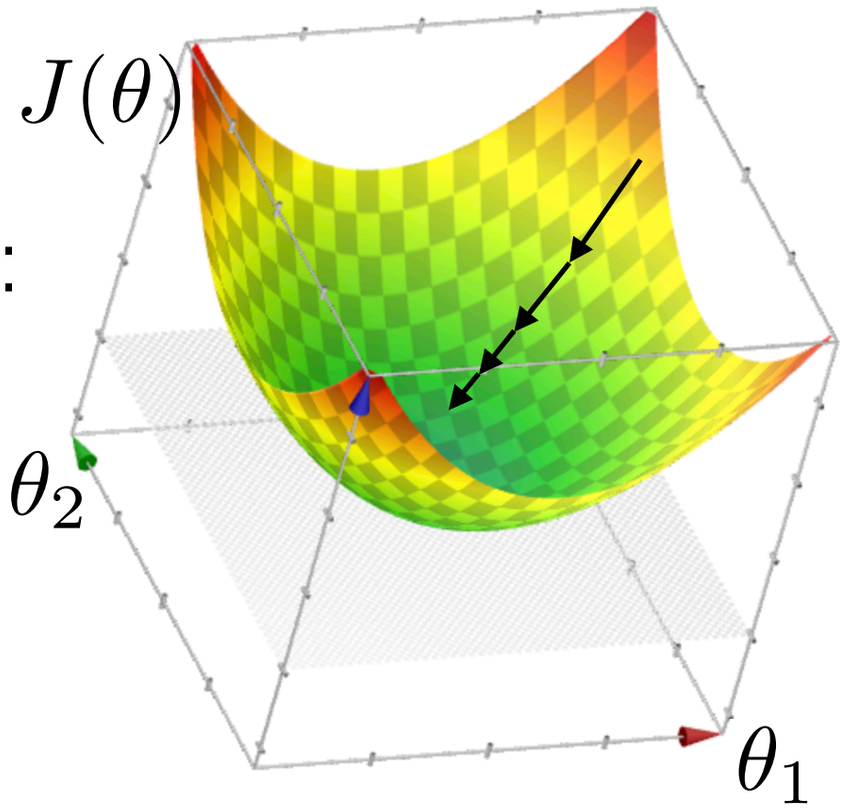


- SGD:

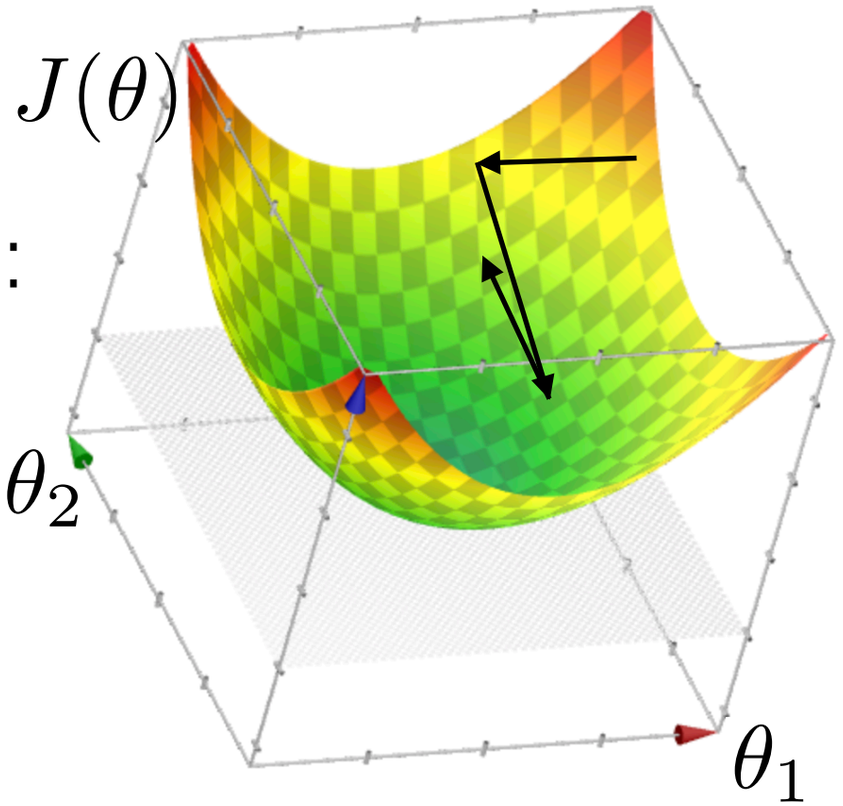


Stochastic gradient descent (SGD) properties

- GD:

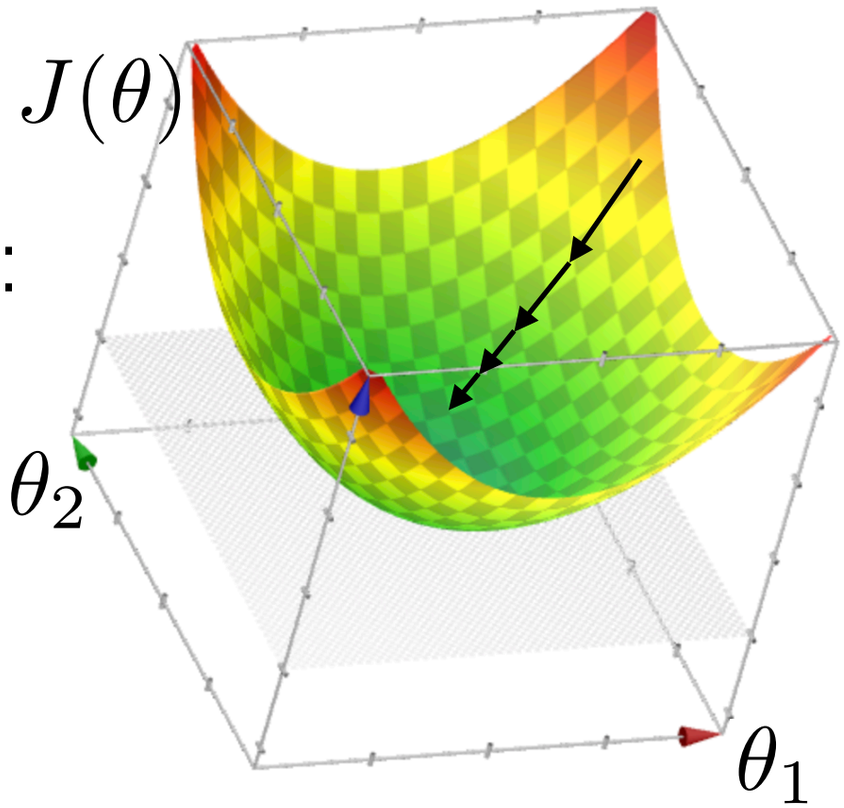


- SGD:

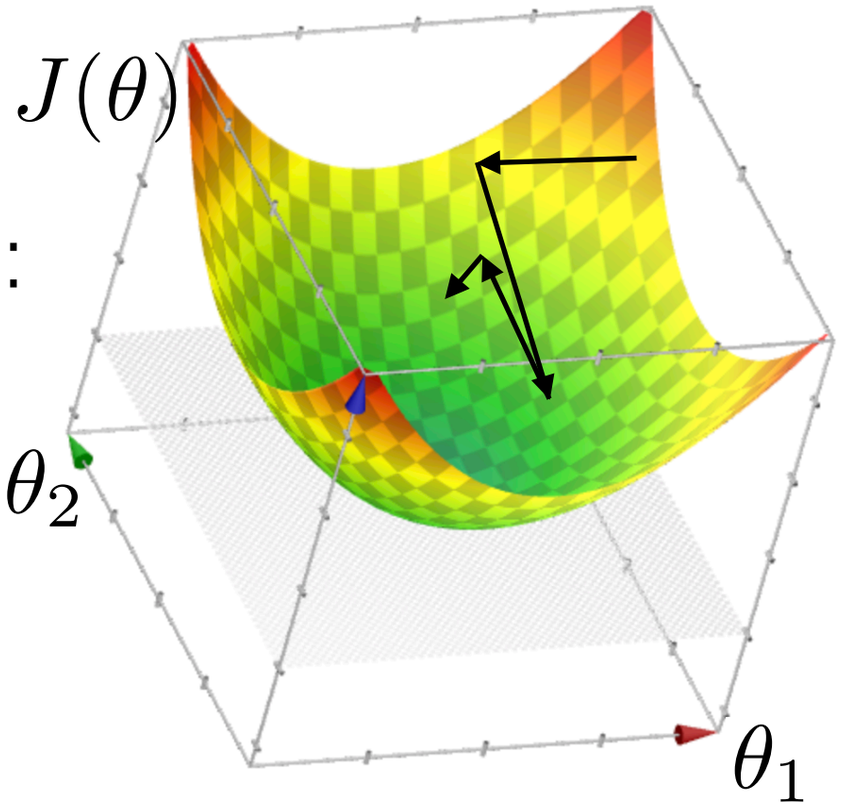


Stochastic gradient descent (SGD) properties

- GD:

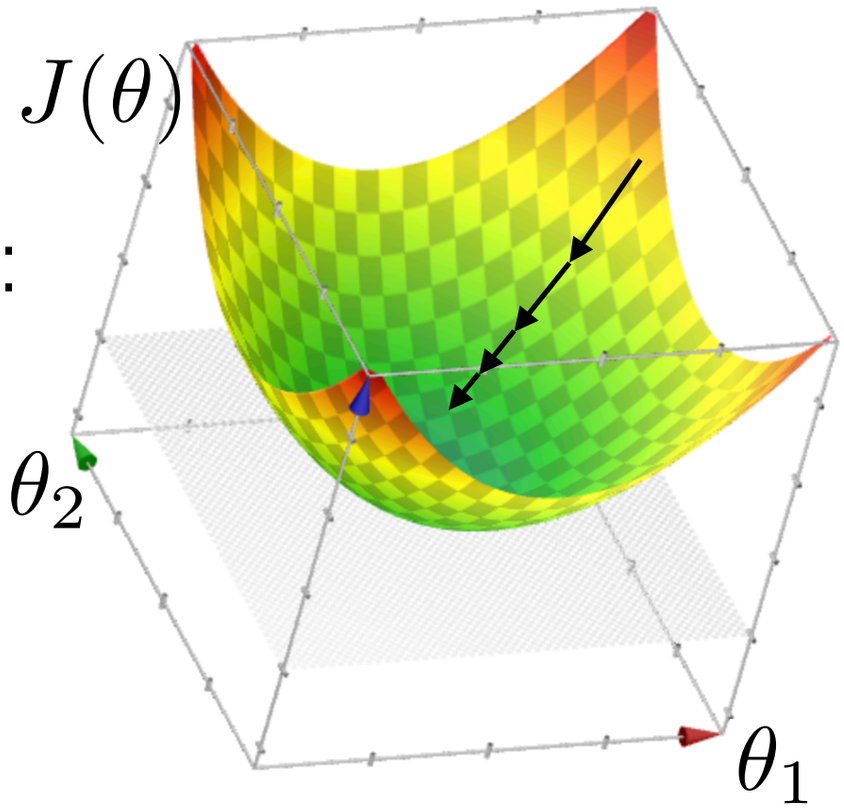


- SGD:

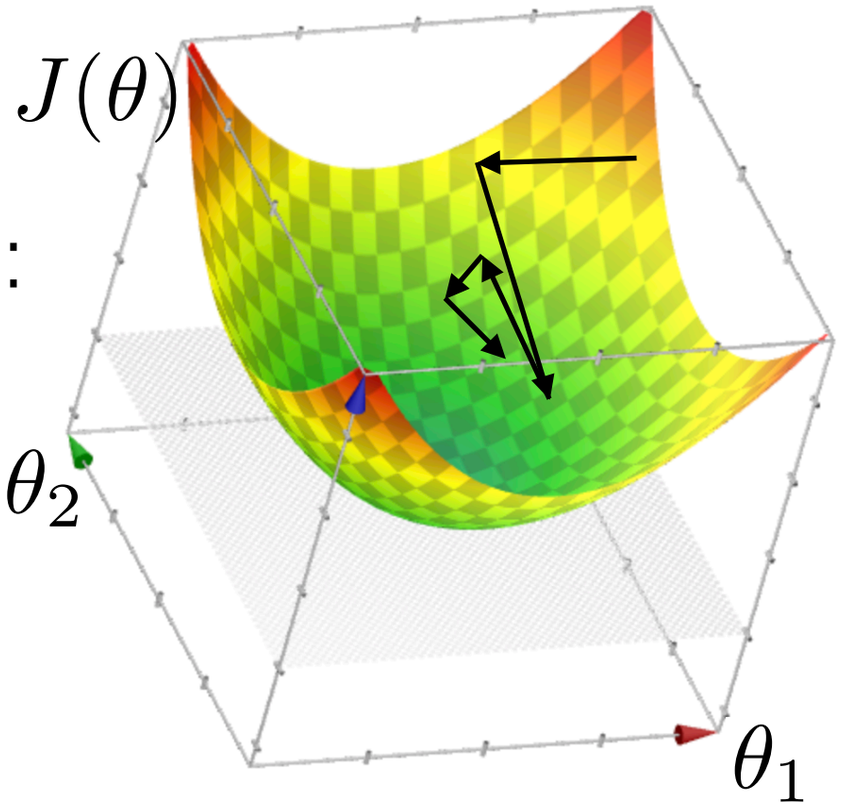


Stochastic gradient descent (SGD) properties

- GD:

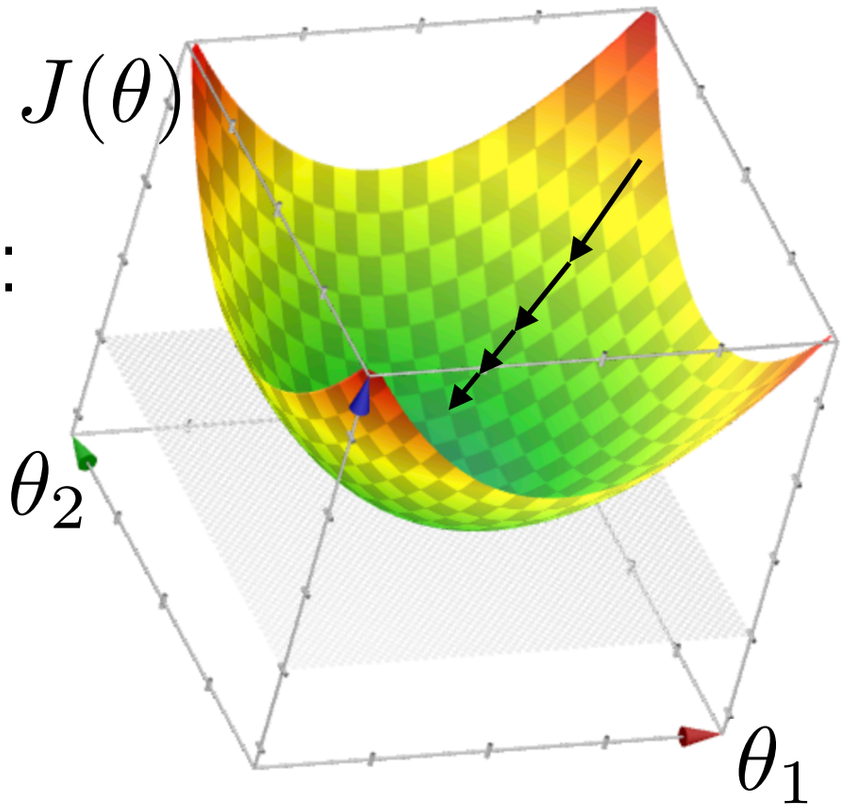


- SGD:

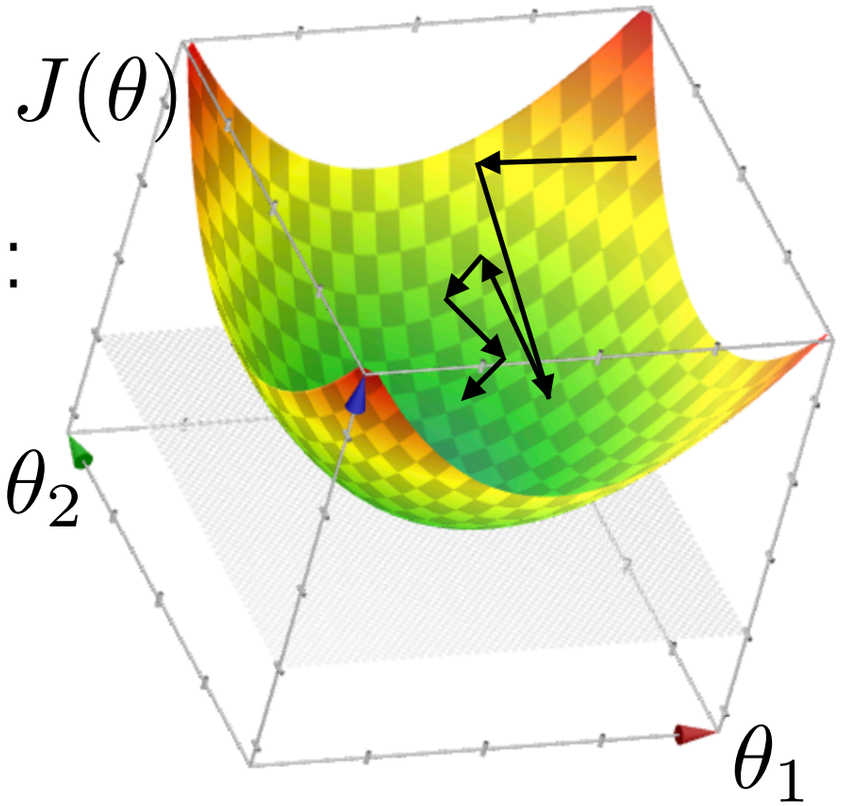


Stochastic gradient descent (SGD) properties

- GD:

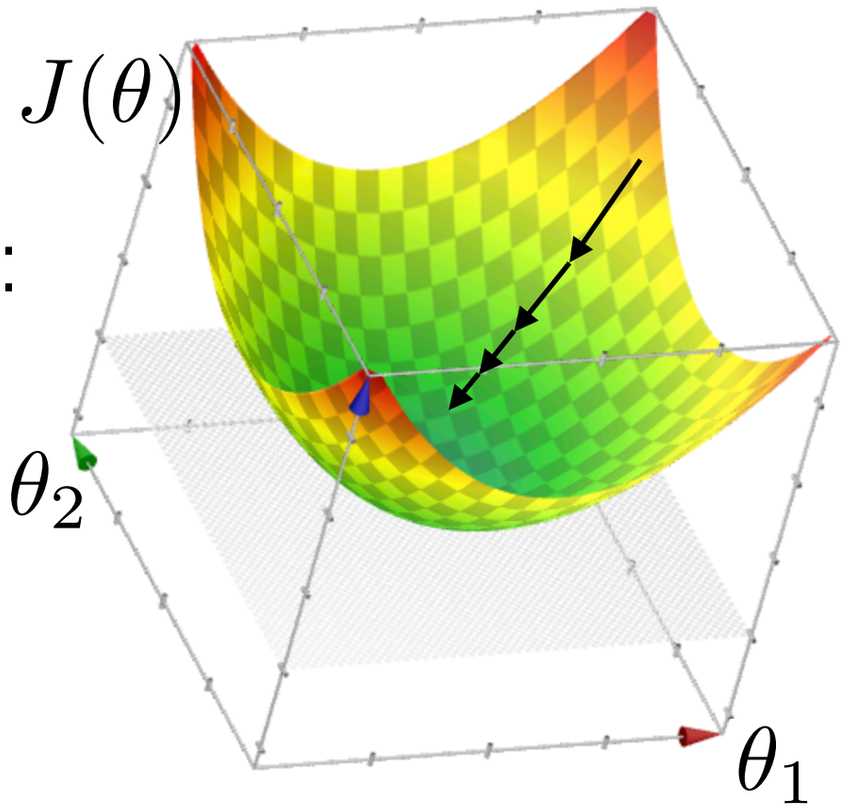


- SGD:

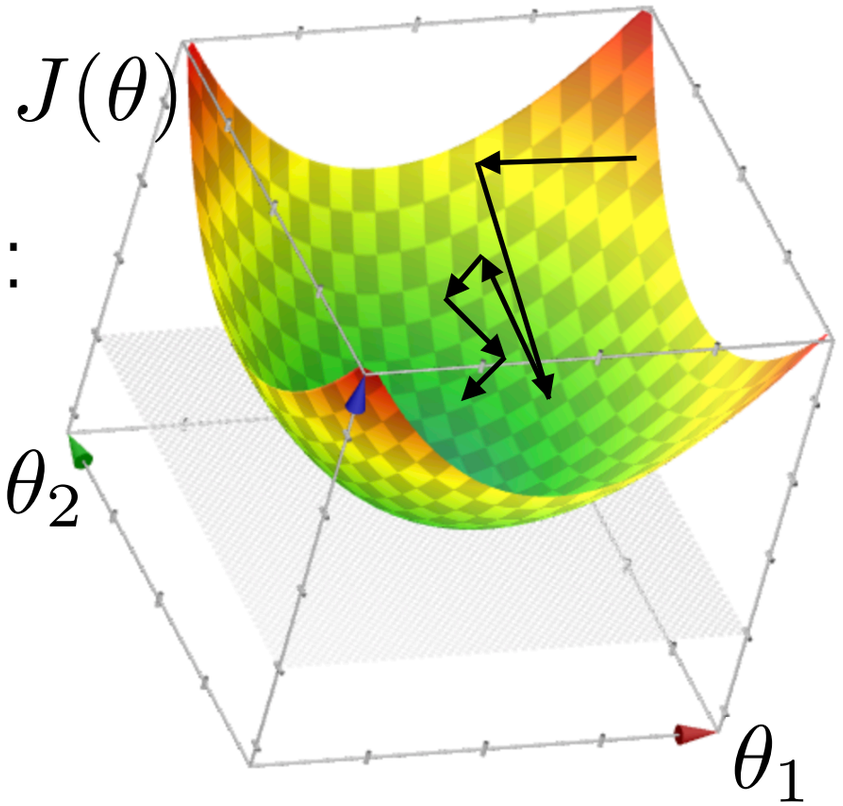


Stochastic gradient descent (SGD) properties

- GD:

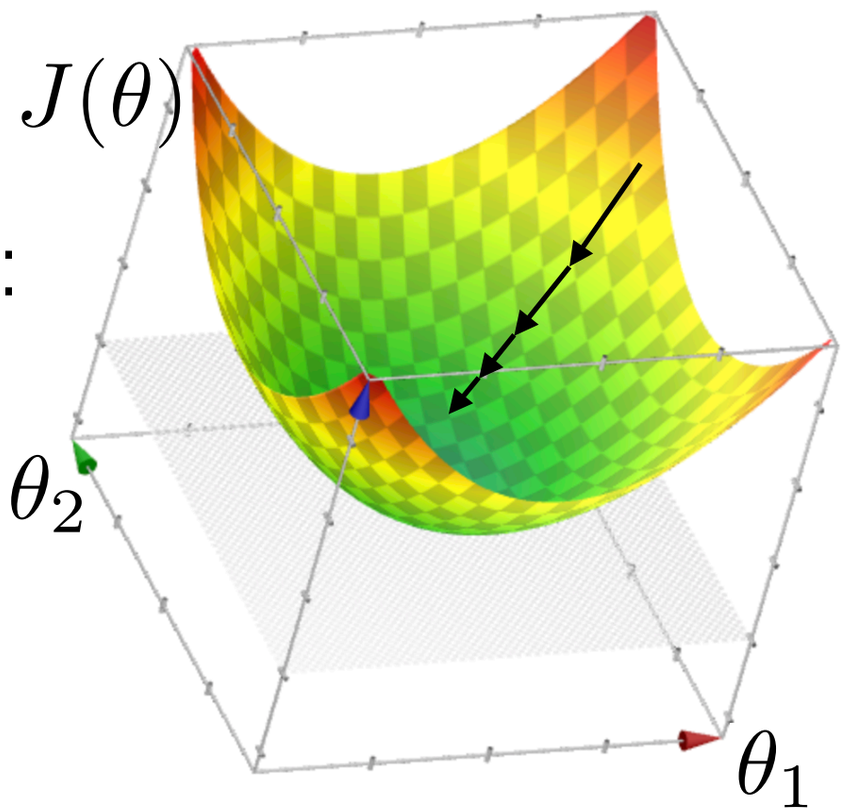


- SGD:

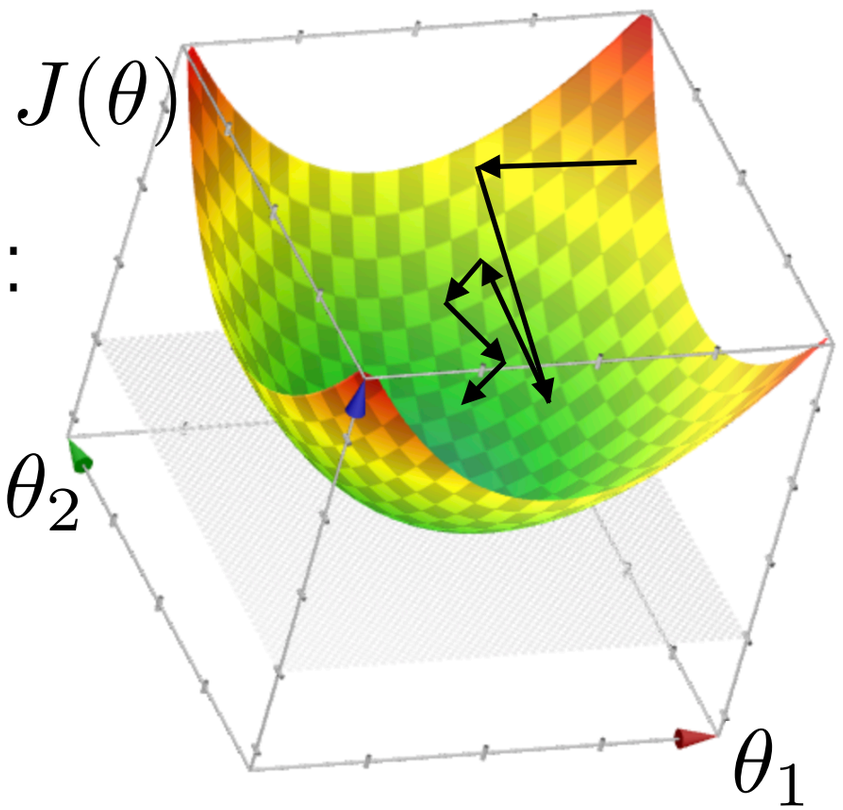


Stochastic gradient descent (SGD) properties

- GD:



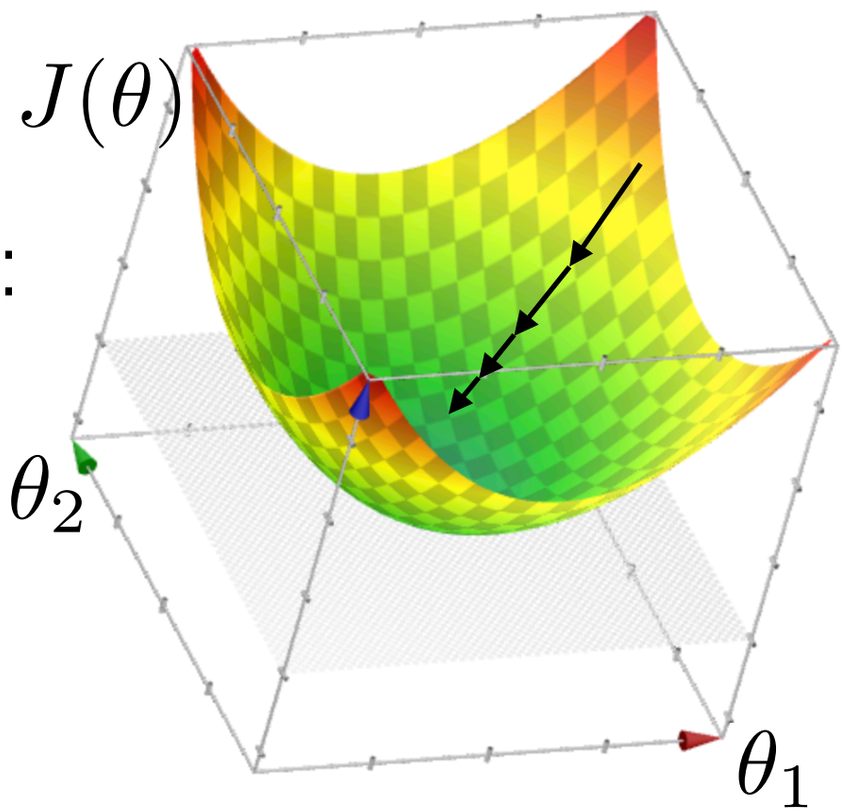
- SGD:



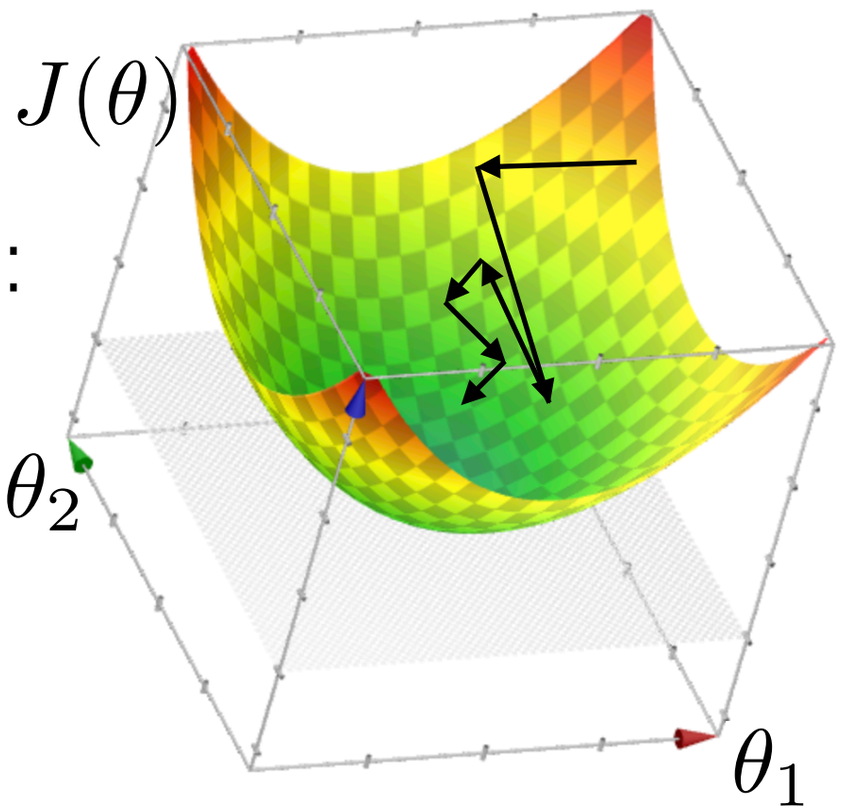
- **Theorem**: SGD performance

Stochastic gradient descent (SGD) properties

- GD:



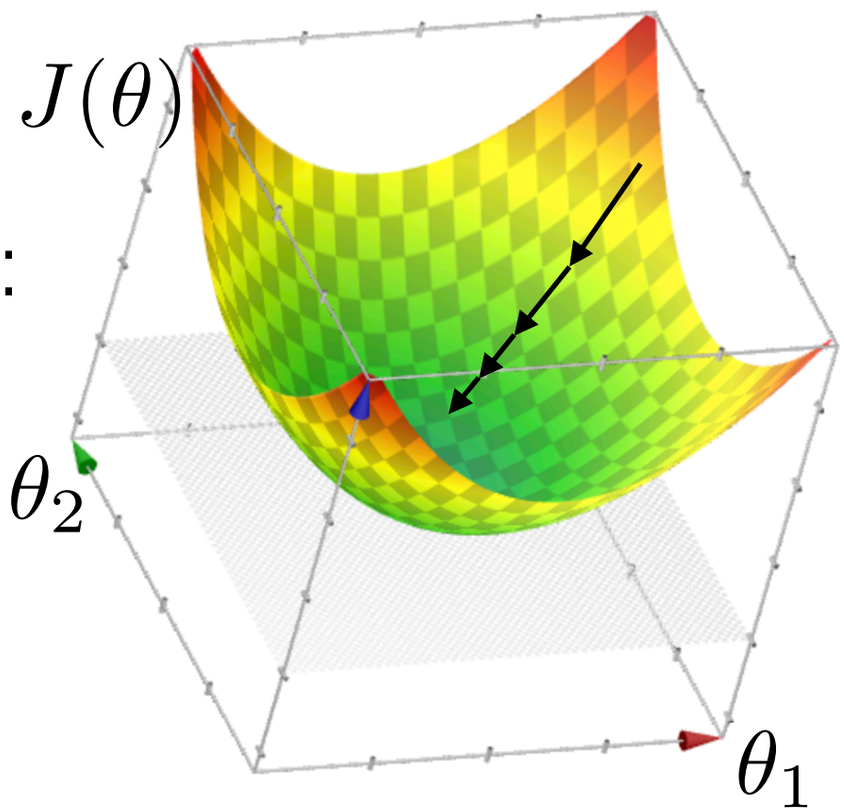
- SGD:



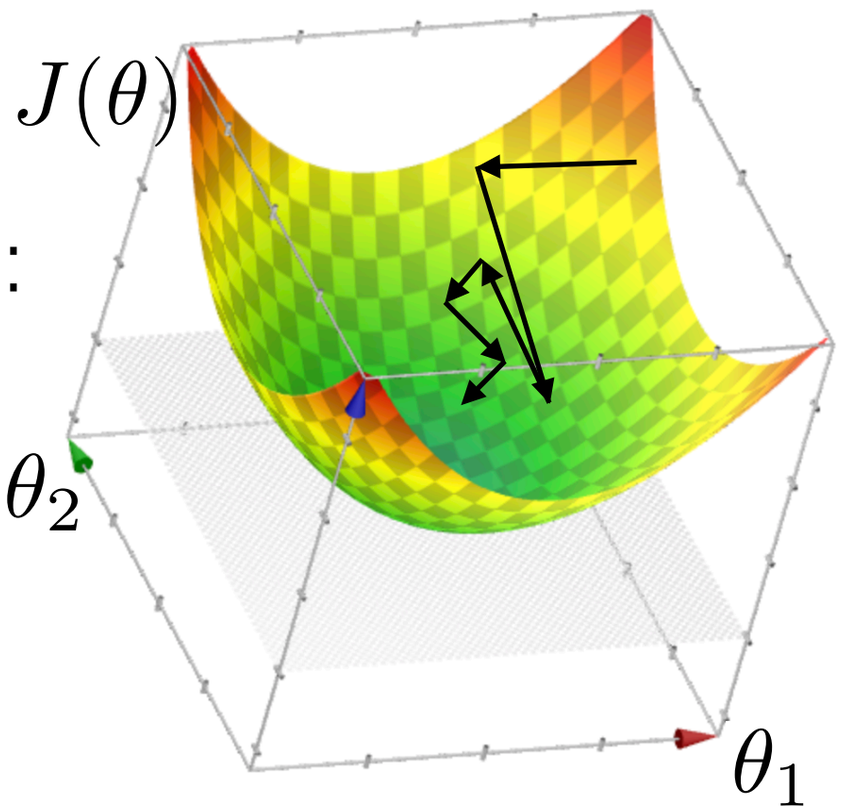
- **Theorem**: SGD performance
  - **Assumptions**:

Stochastic gradient descent (SGD) properties

- GD:



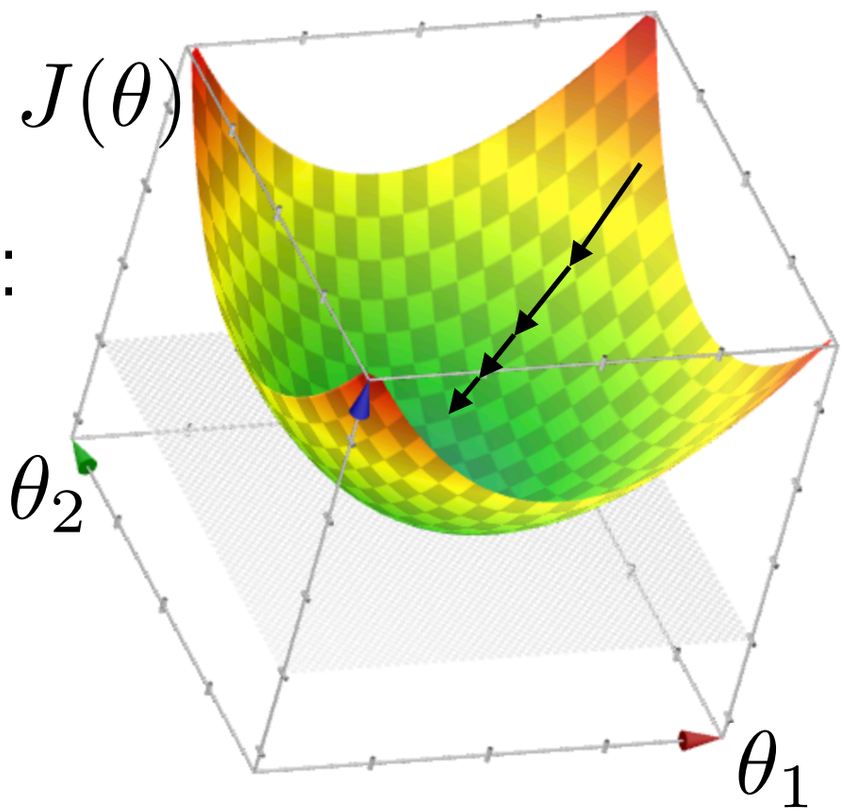
- SGD:



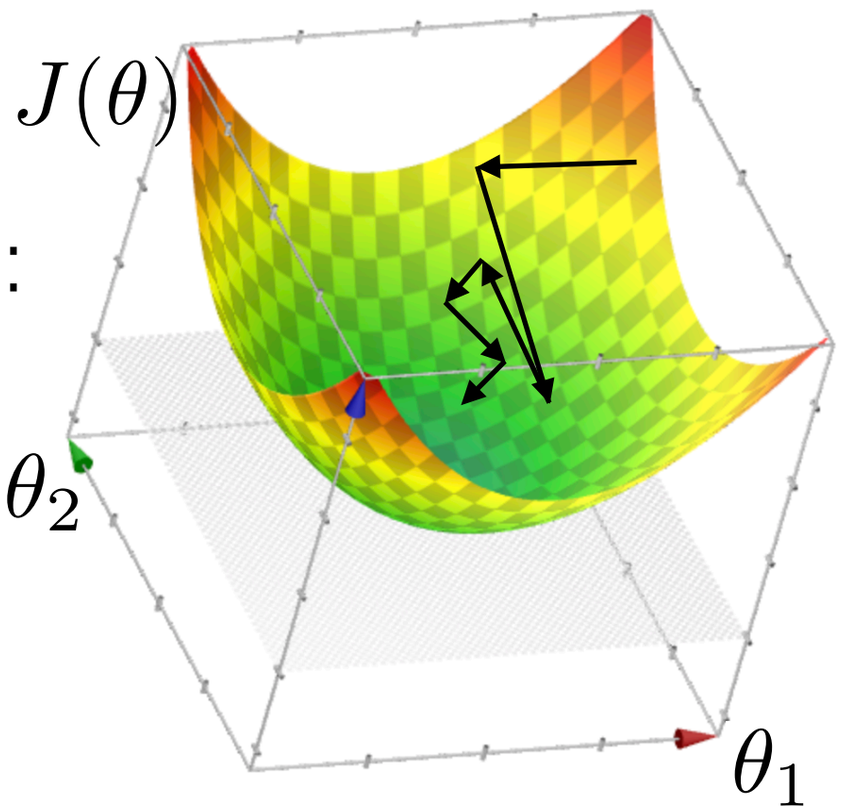
- **Theorem**: SGD performance
  - **Assumptions**: (Choose any $\tilde{\epsilon} > 0$)

Stochastic gradient descent (SGD) properties

- GD:



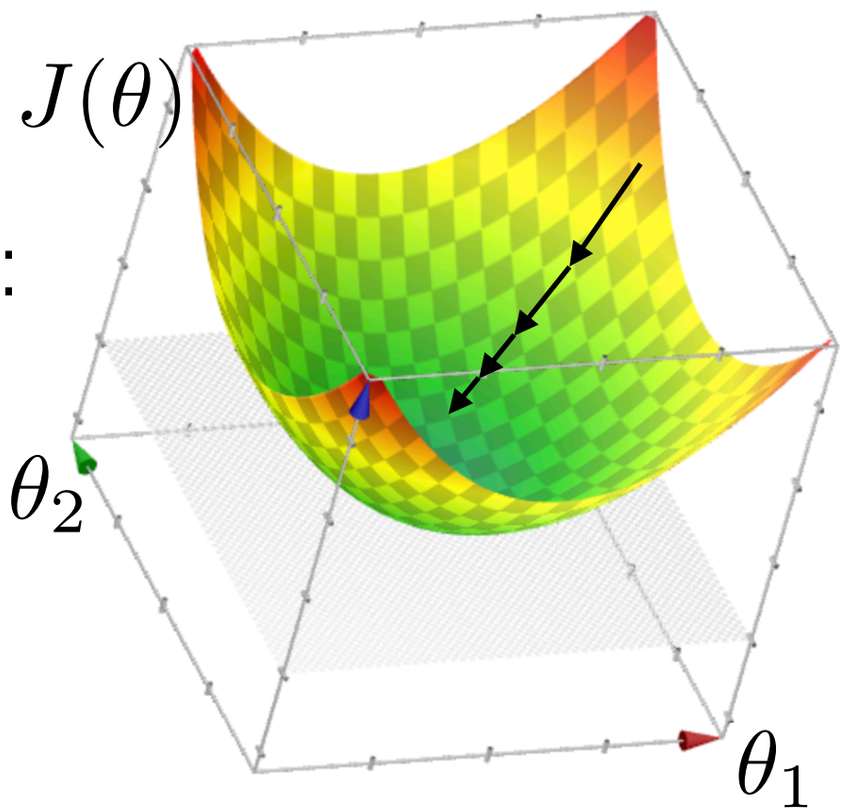
- SGD:



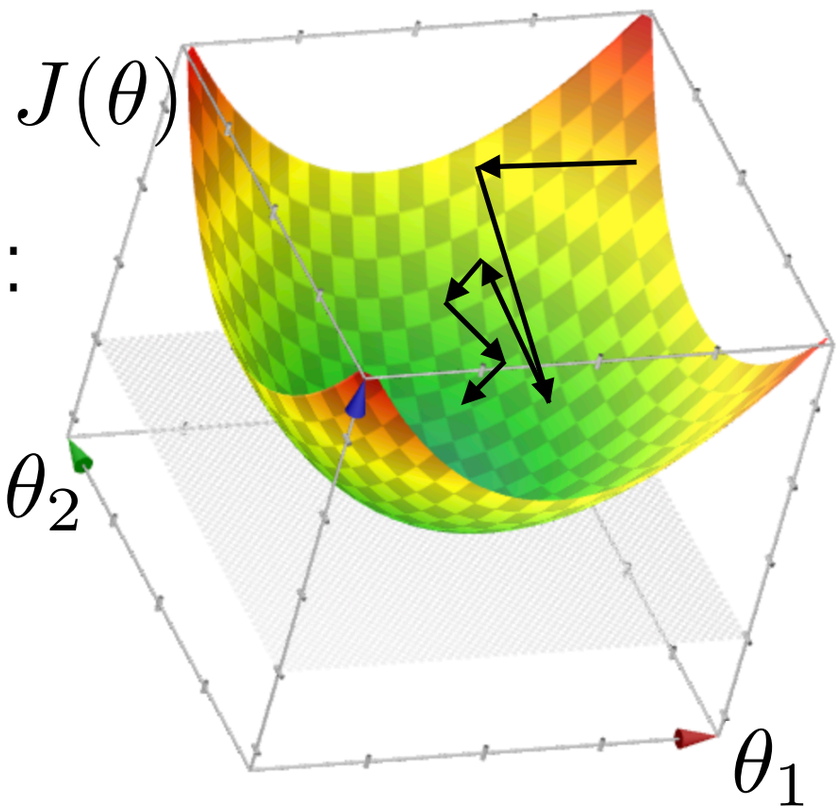
- **Theorem**: SGD performance
  - **Assumptions**: (Choose any $\tilde{\epsilon} > 0$)
    - *f* is “nice” & convex, has a unique global minimizer

Stochastic gradient descent (SGD) properties

- GD:



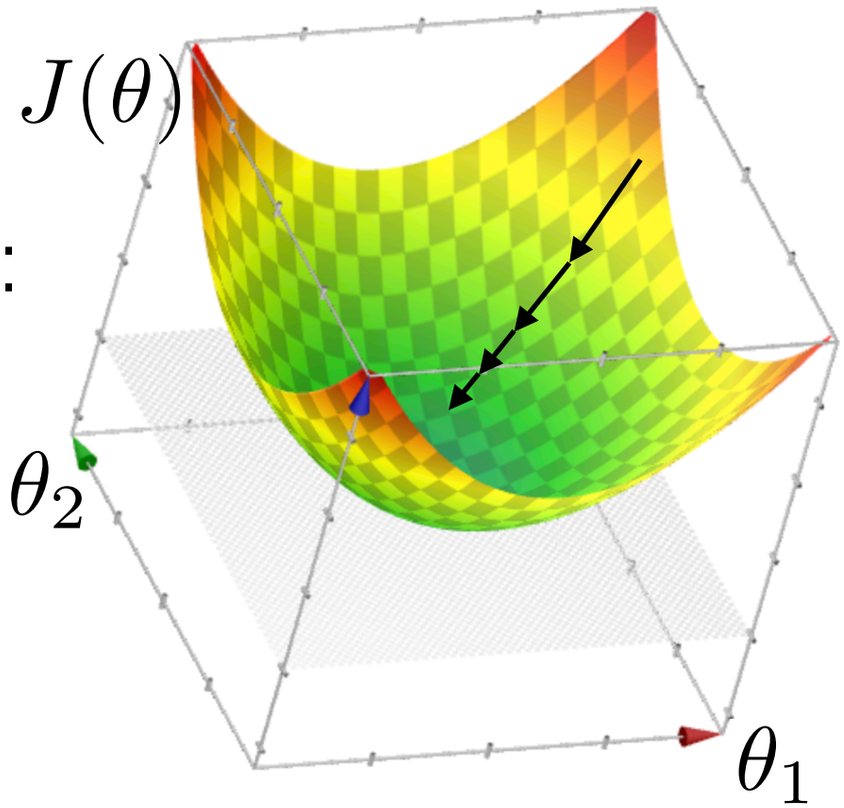
- SGD:



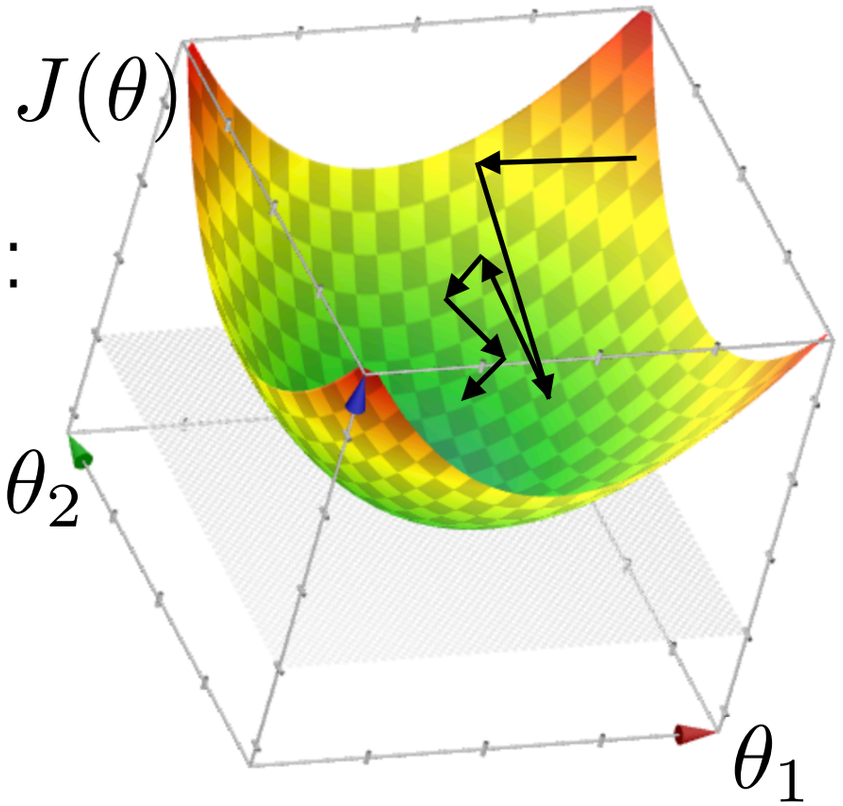
- **Theorem**: SGD performance
  - **Assumptions**: (Choose any $\tilde{\epsilon} > 0$)
    - f is "nice" & convex, has a unique global minimizer
    - $\sum_{t=1}^{\infty} \eta(t) = \infty, \sum_{t=1}^{\infty} (\eta(t))^2 < \infty$

Stochastic gradient descent (SGD) properties

- GD:



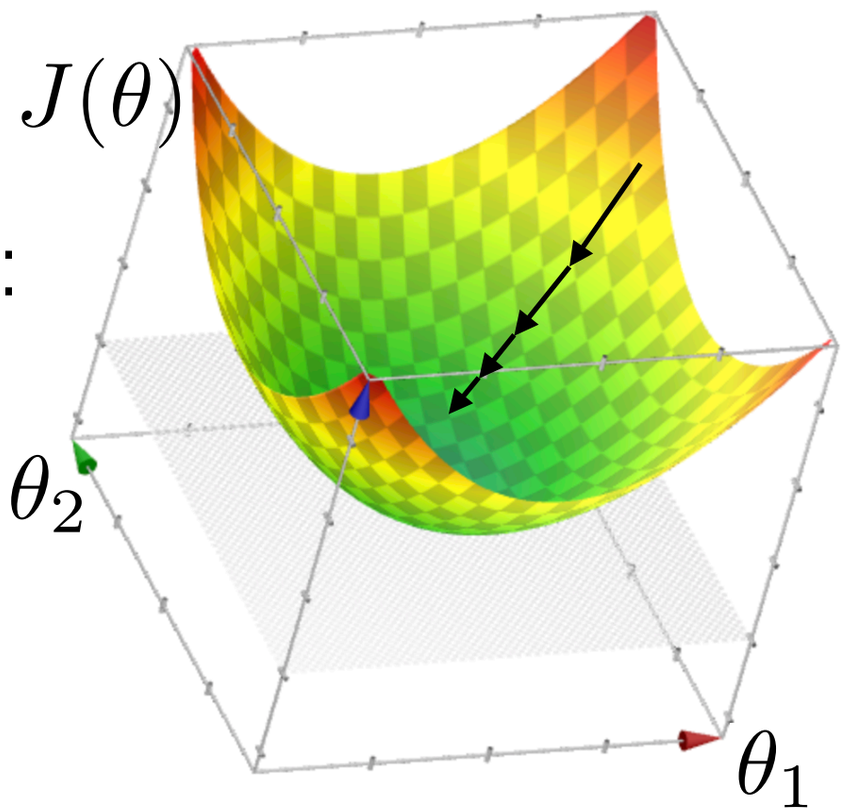
- SGD:



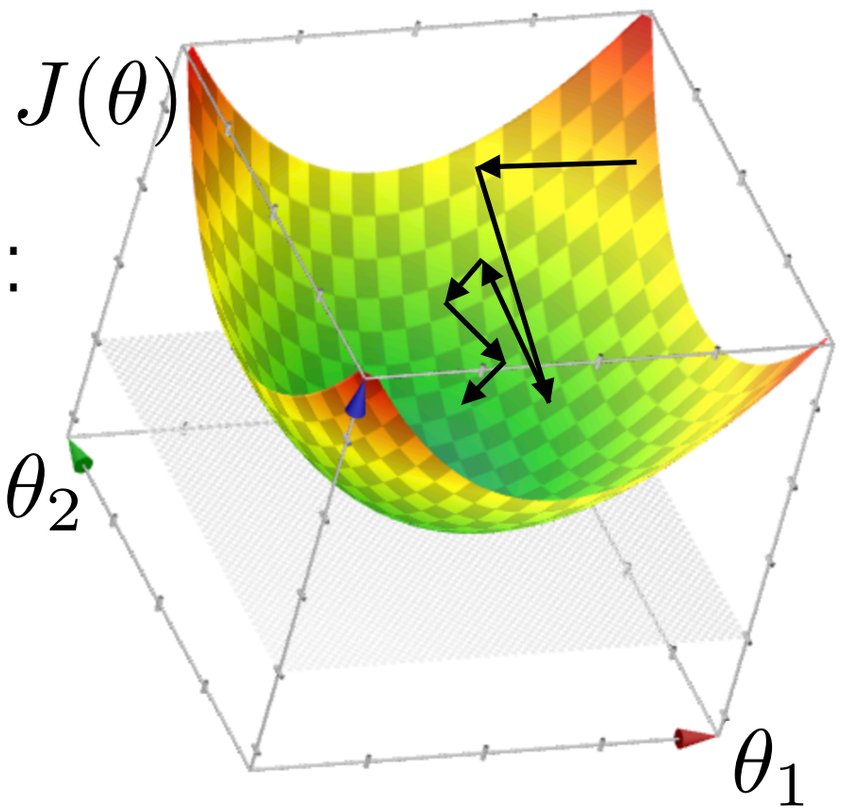
- **Theorem**: SGD performance
  - **Assumptions**: (Choose any $\tilde{\epsilon} > 0$)
    - f is "nice" & convex, has a unique global minimizer
    - $\sum_{t=1}^{\infty} \eta(t) = \infty, \sum_{t=1}^{\infty} (\eta(t))^2 < \infty$
      - e.g. $\eta(t) = \alpha(\tau_0 + t)^{-\kappa} (\kappa \in (0.5, 1])$

Stochastic gradient descent (SGD) properties

- GD:



- SGD:



- **Theorem**: SGD performance
  - **Assumptions**: (Choose any $\tilde{\epsilon} > 0$)
    - f is "nice" & convex, has a unique global minimizer
    - $\sum_{t=1}^{\infty} \eta(t) = \infty, \sum_{t=1}^{\infty} (\eta(t))^2 < \infty$
      - e.g. $\eta(t) = \alpha(\tau_0 + t)^{-\kappa} (\kappa \in (0.5, 1])$
  - **Conclusion**: If run long enough, stochastic gradient descent will return a value within $\tilde{\epsilon}$ of the global minimizer

WHEW!

NQ tomorrow

No lecture or exercises next week

Lab and HQ next week