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? Prof. Tamara Broderick

Questions? discourse.odl.mit.edu ("Lecture 4" category)

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/logistic regression
II. Gradient descent

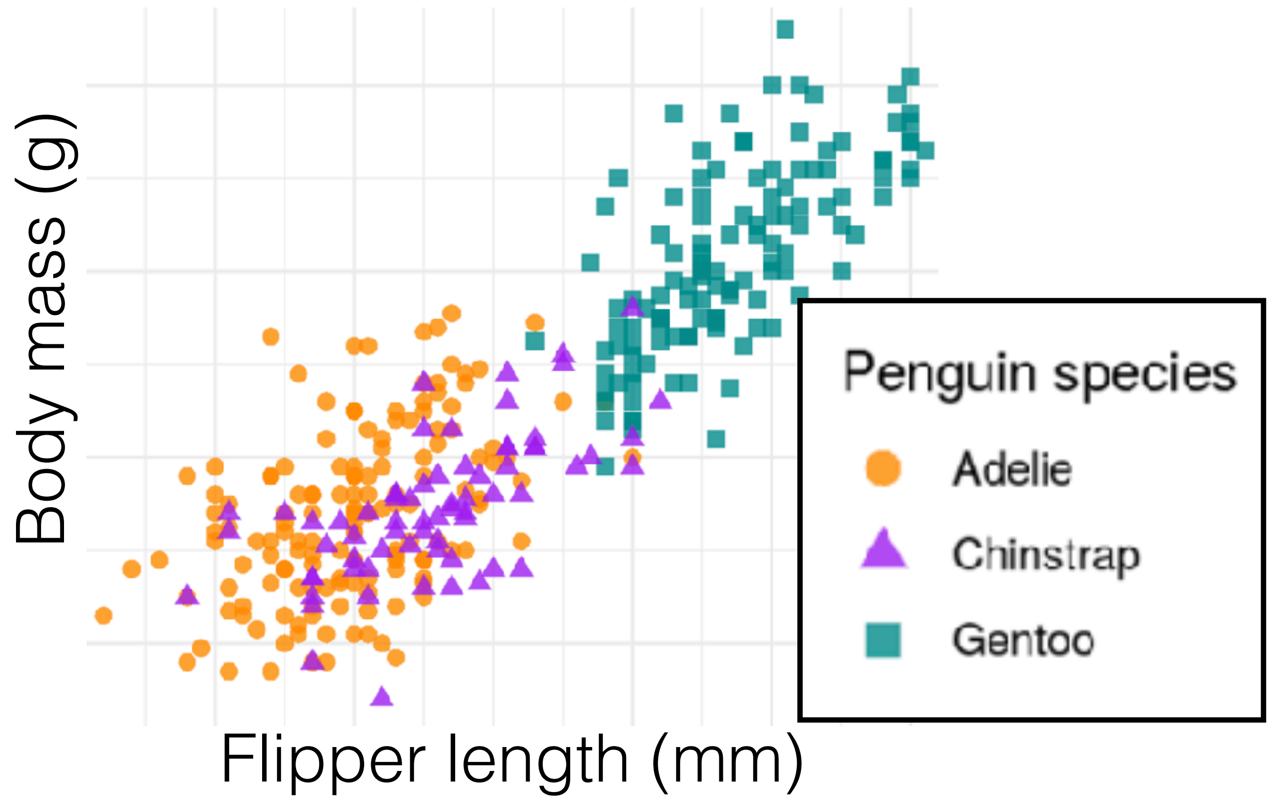
Recall

Recall

- Perceptron struggles with data that's not linearly separable

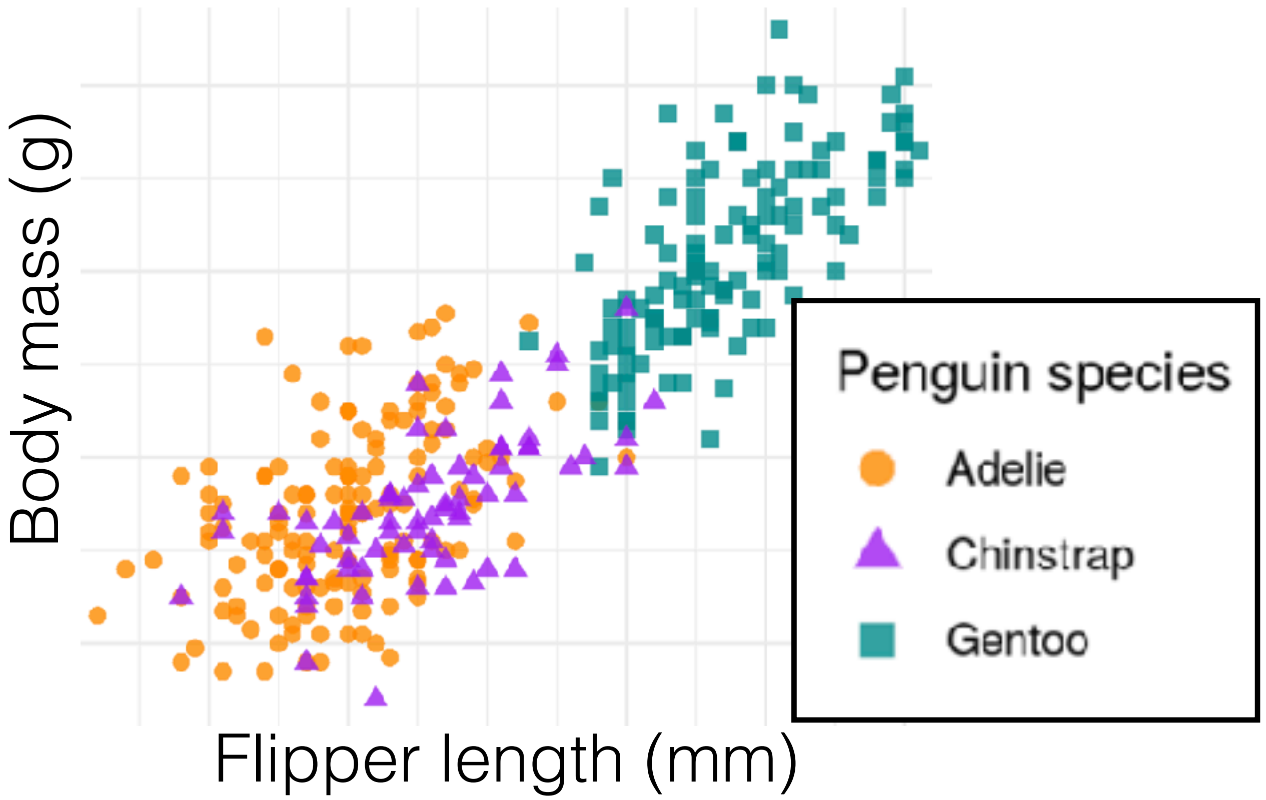
Recall

- Perceptron struggles with data that's not linearly separable



Recall

- Perceptron struggles with data that's not linearly separable

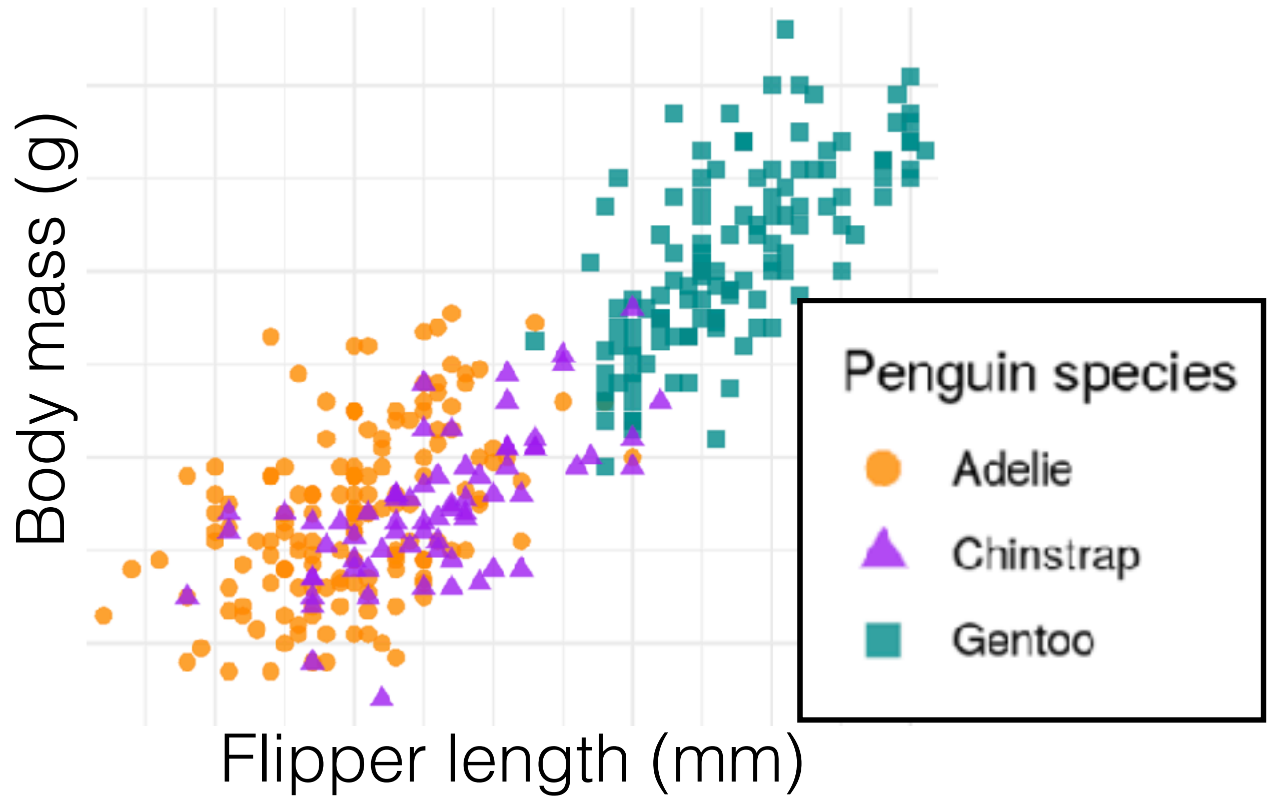


Notice

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

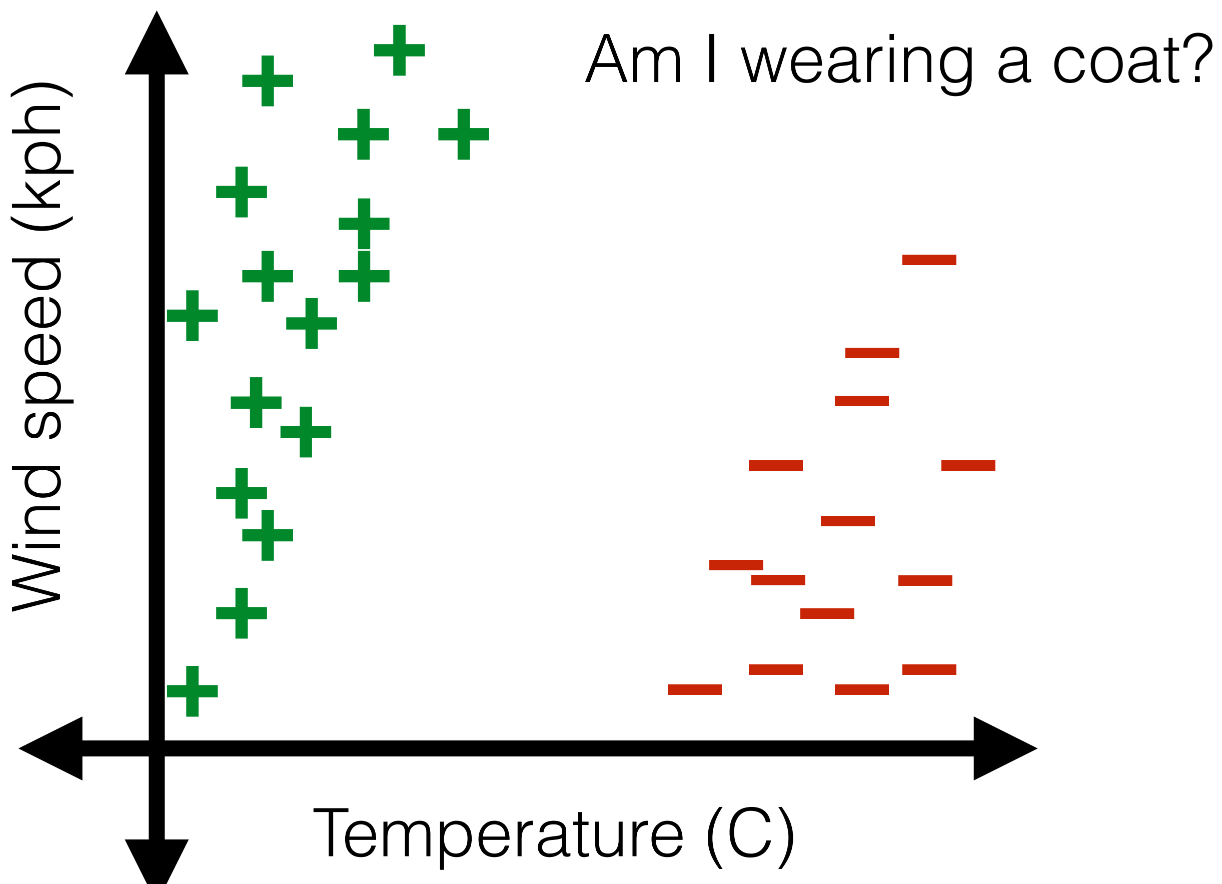
Recall

- Perceptron struggles with data that's not linearly separable



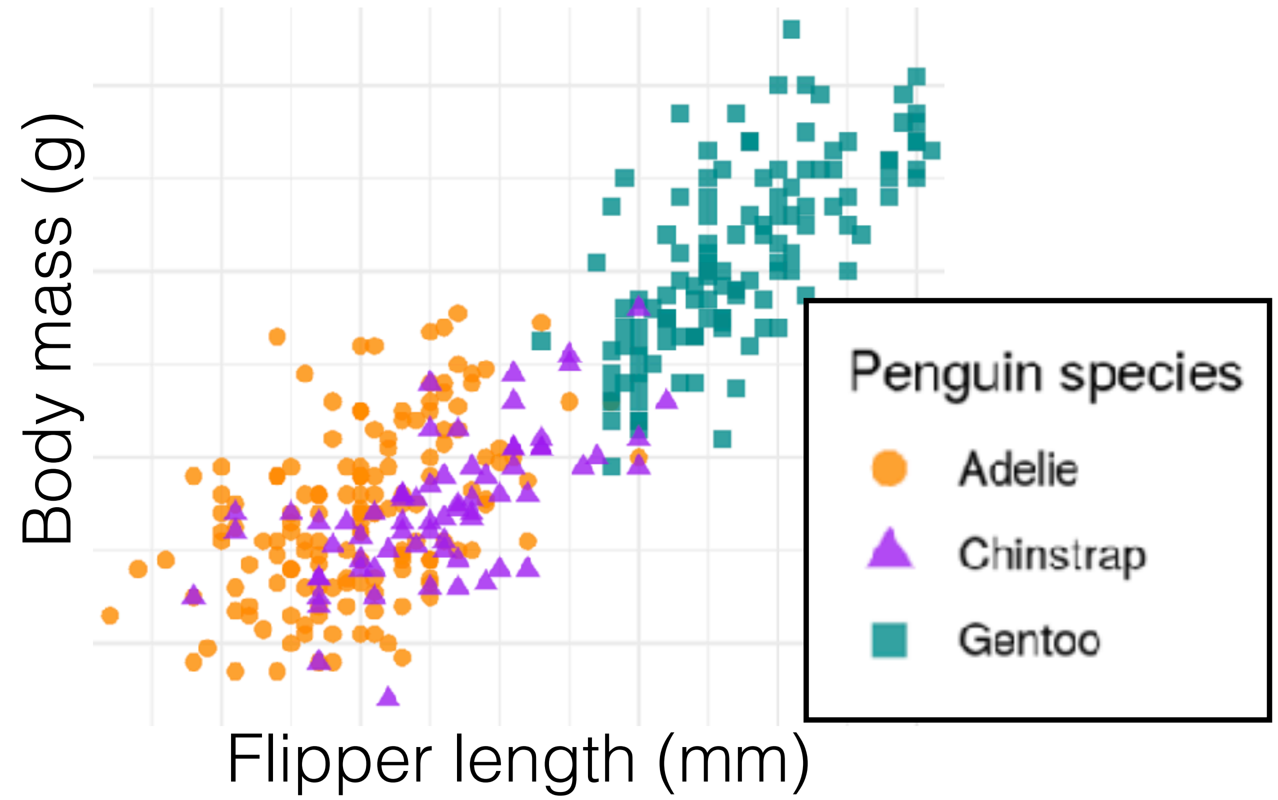
Notice

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



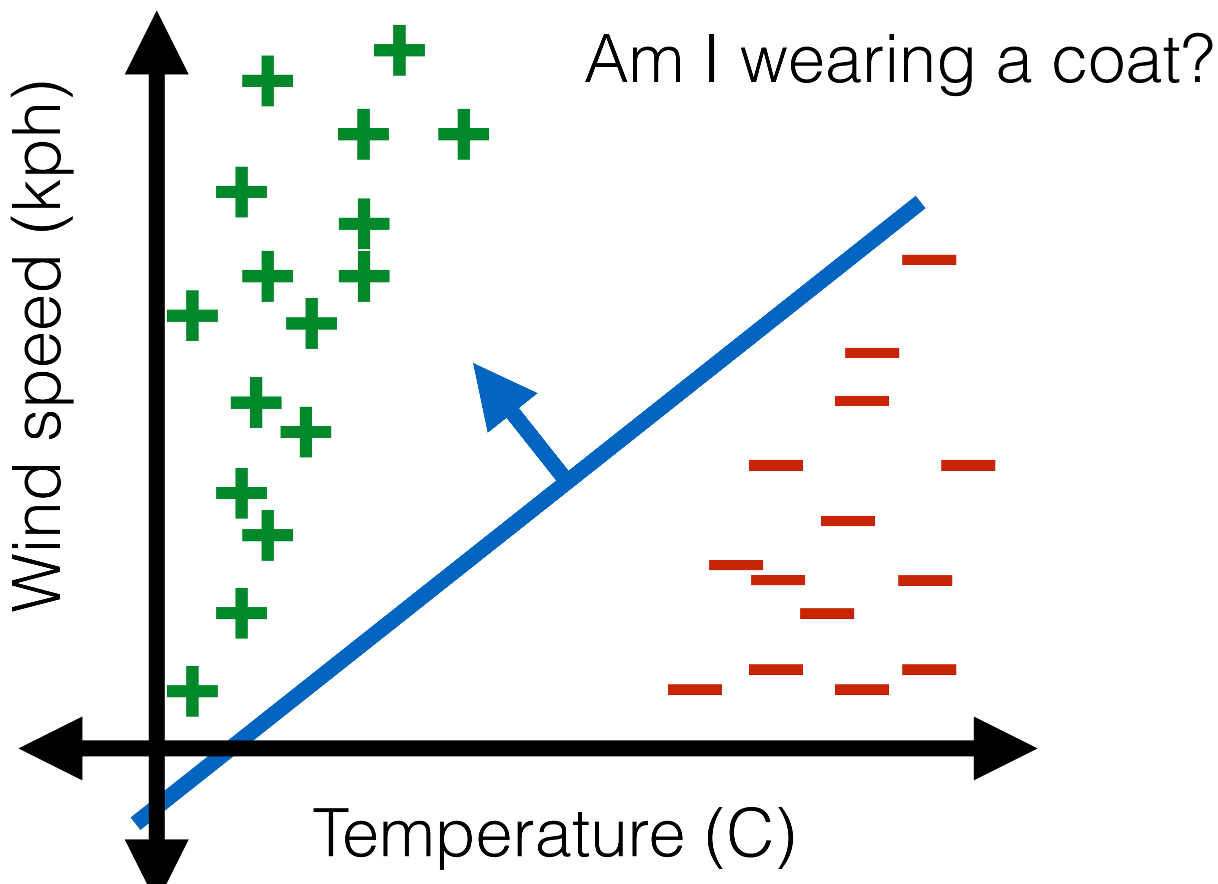
Recall

- Perceptron struggles with data that's not linearly separable



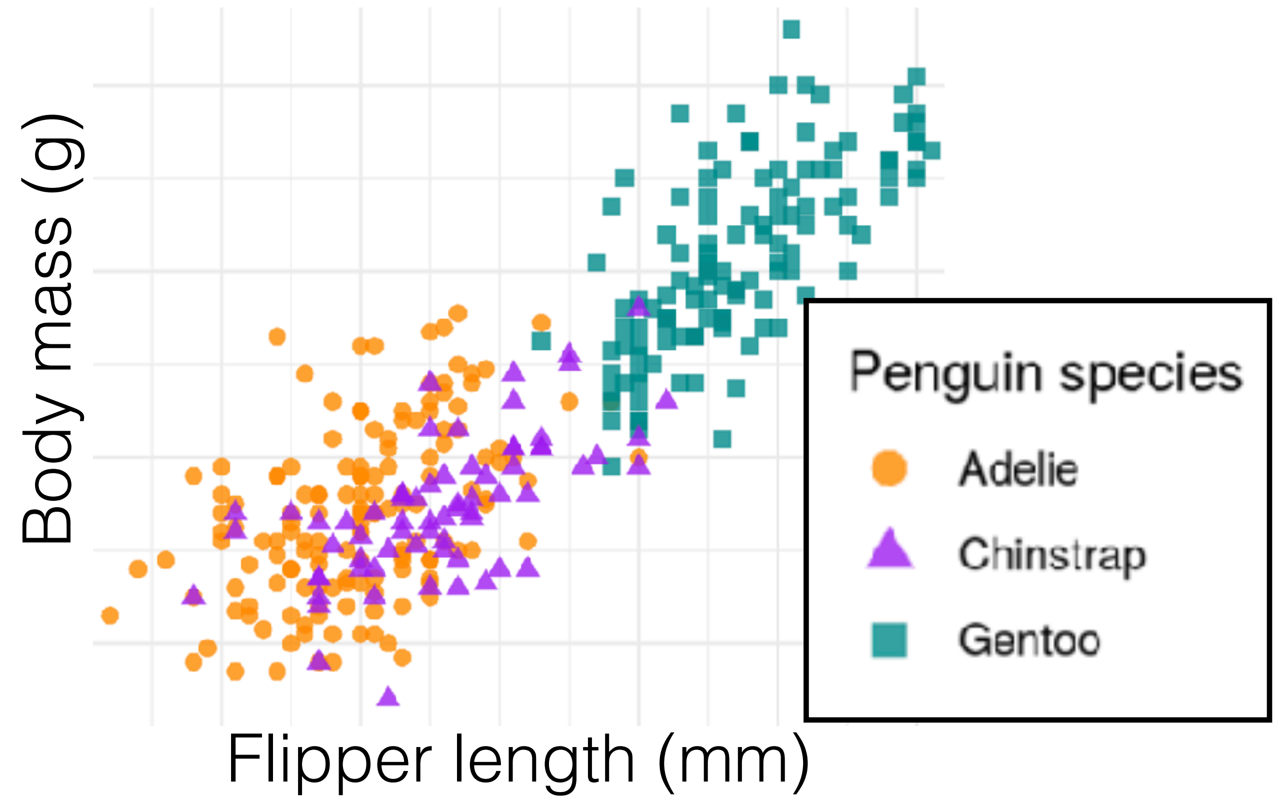
Notice

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



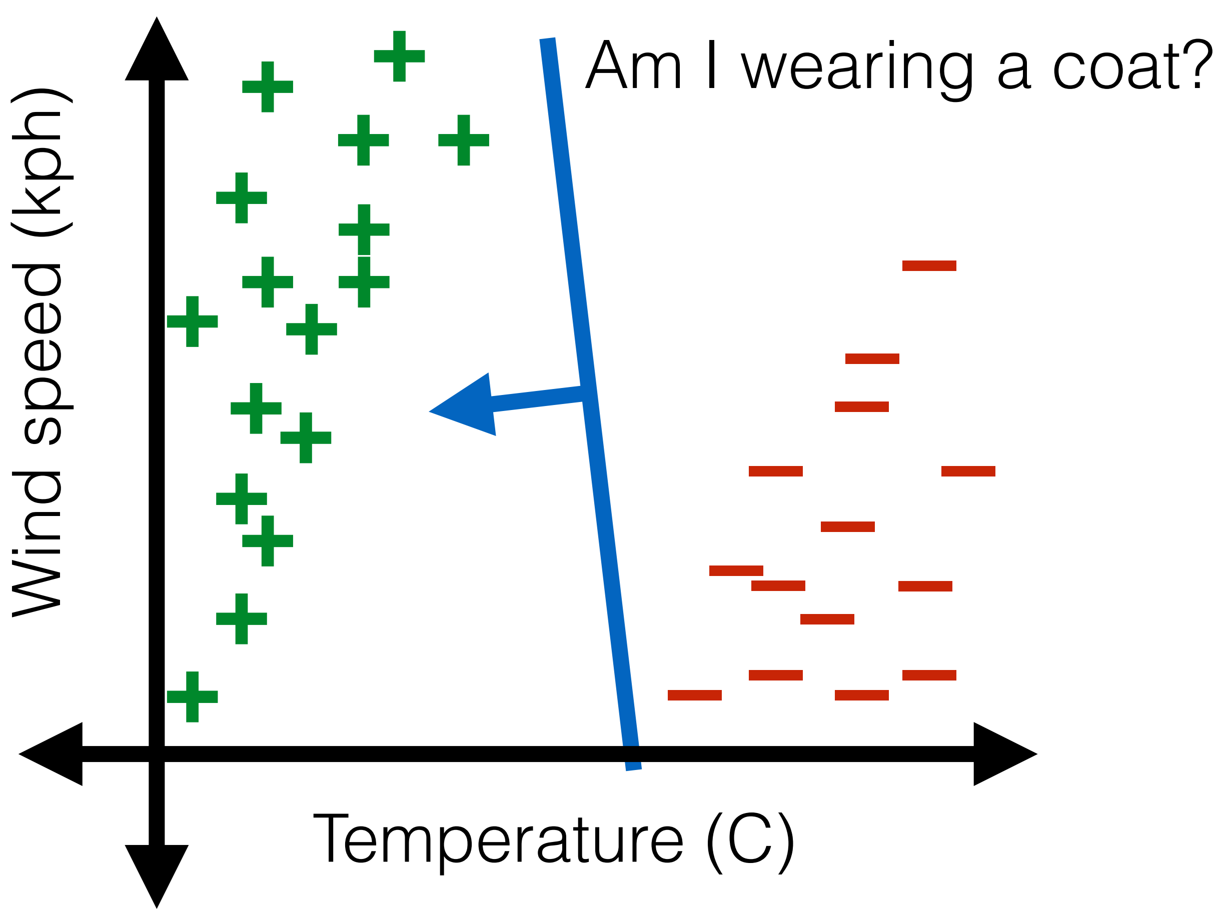
Recall

- Perceptron struggles with data that's not linearly separable



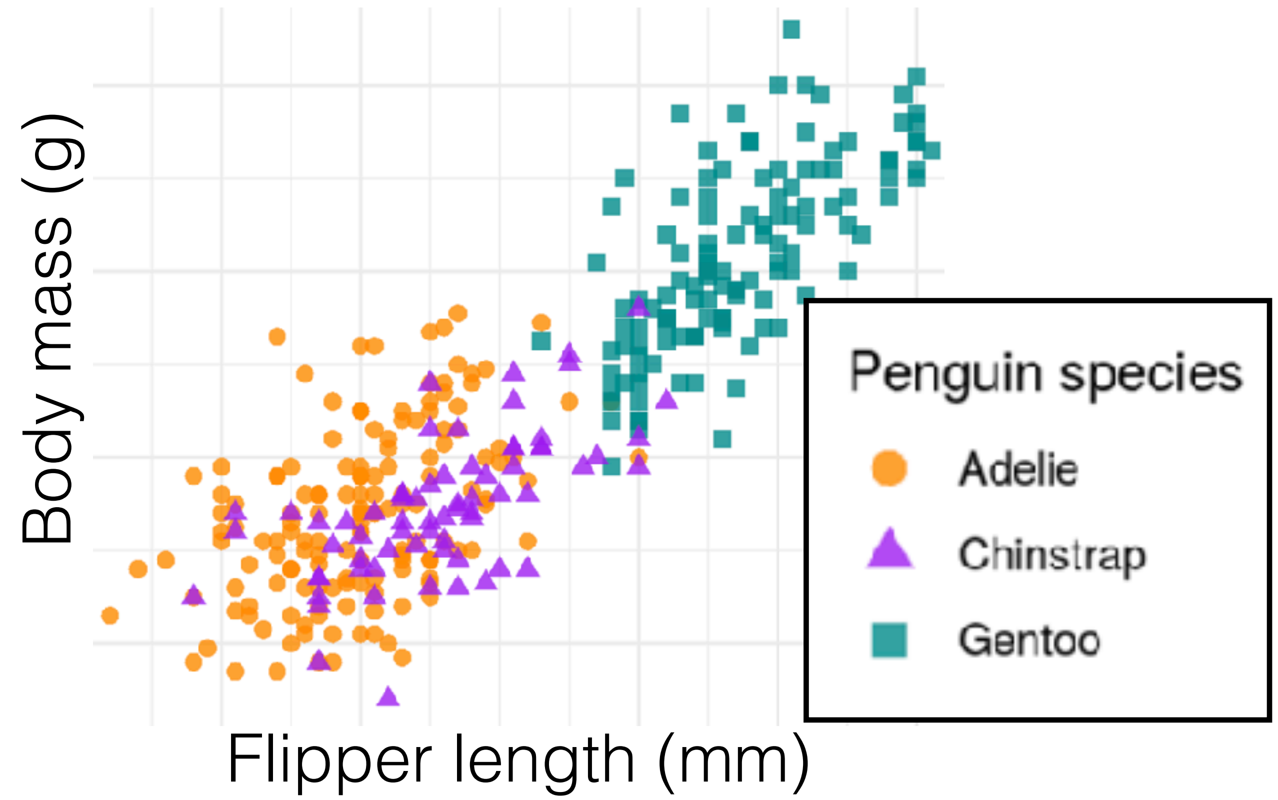
Notice

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



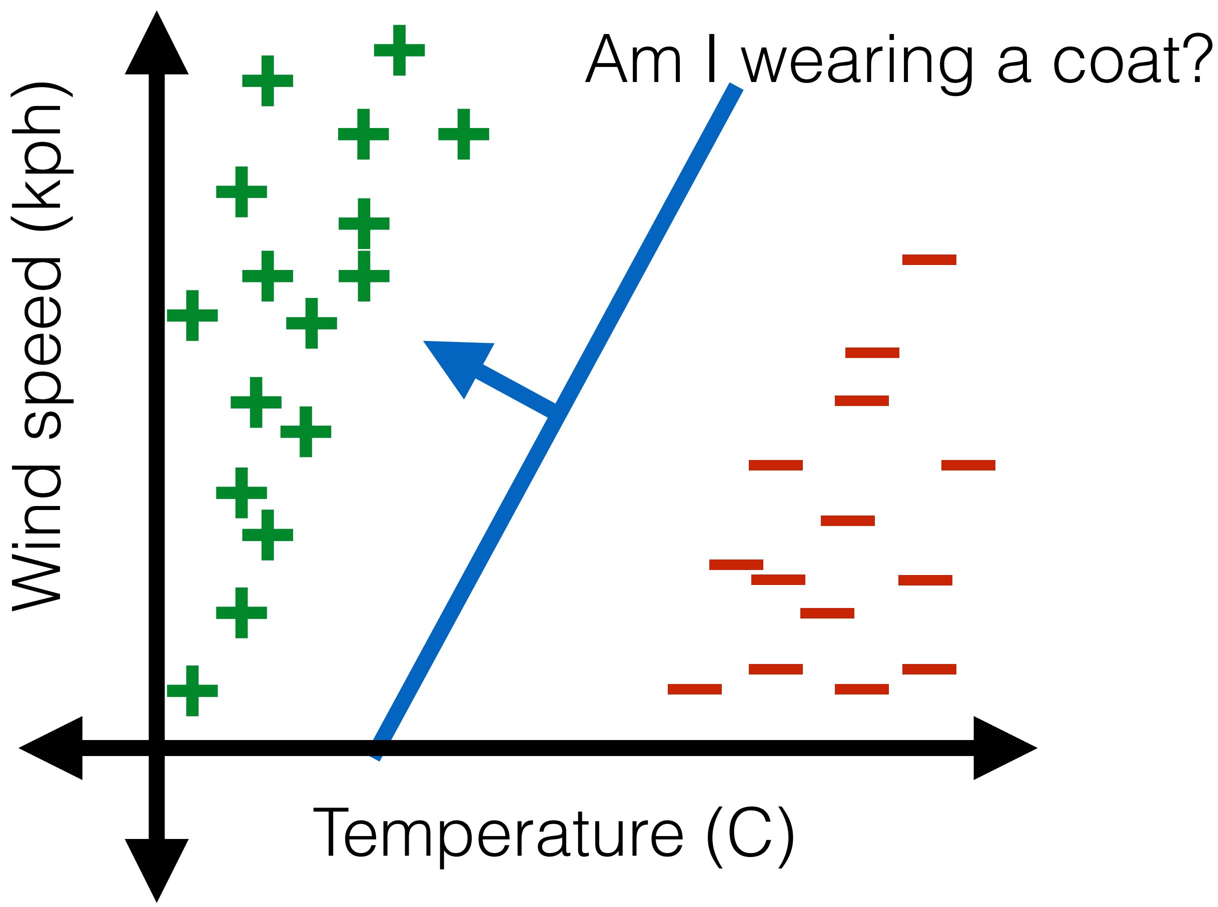
Recall

- Perceptron struggles with data that's not linearly separable



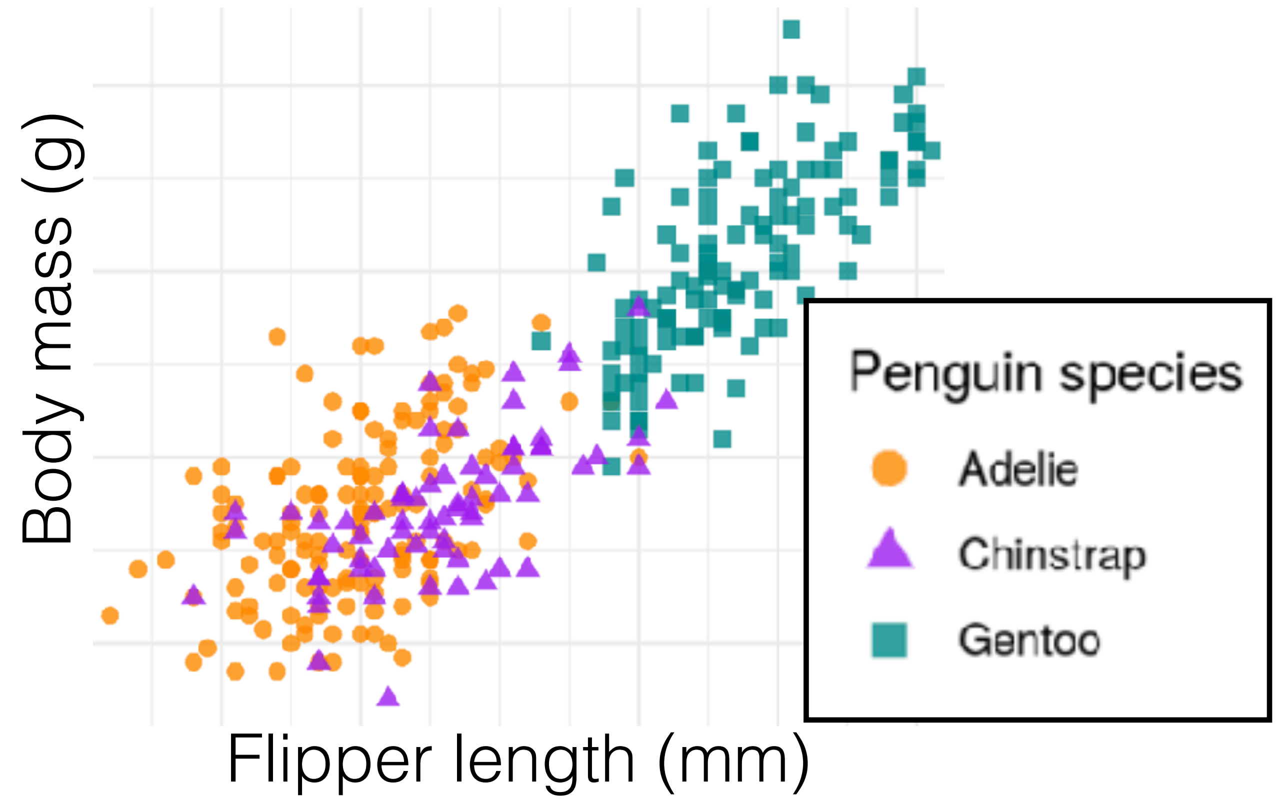
Notice

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



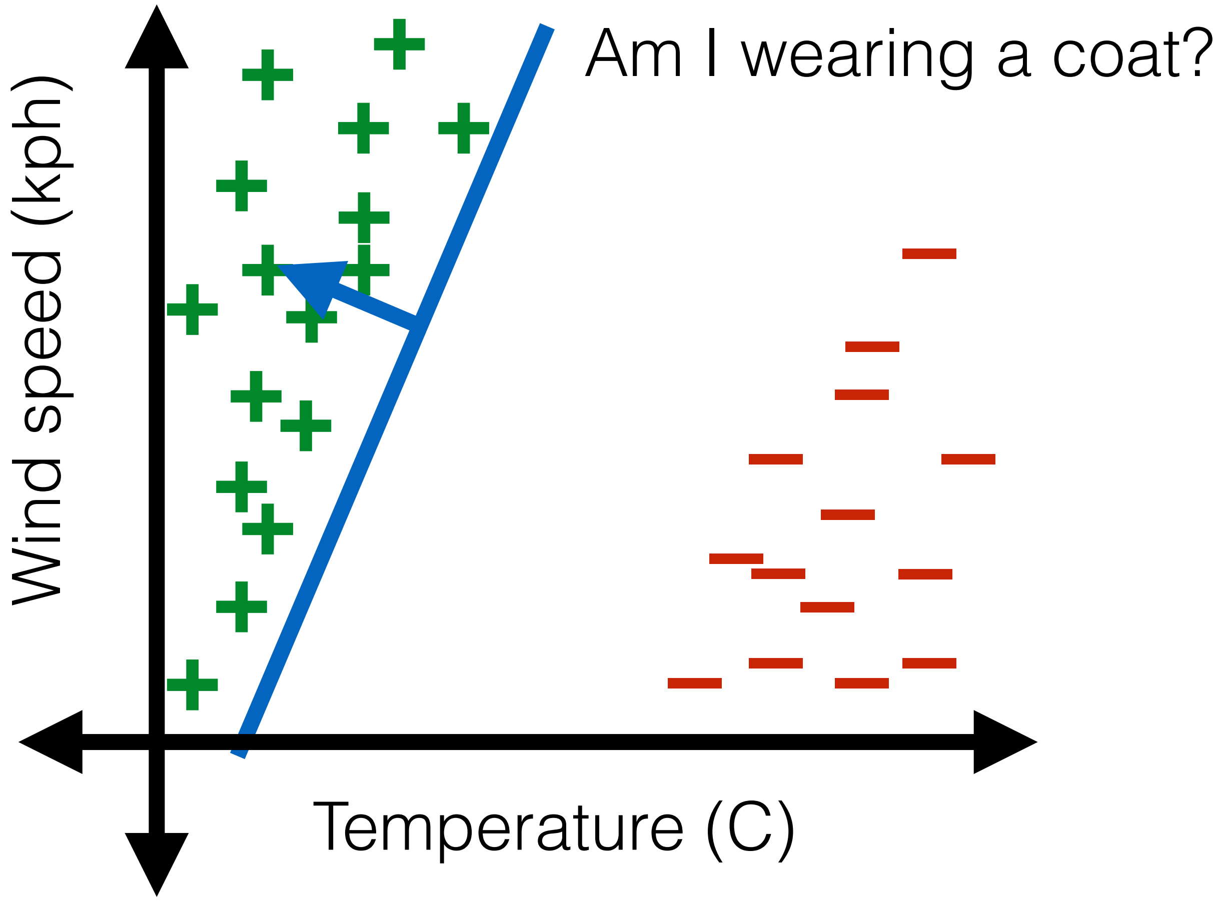
Recall

- Perceptron struggles with data that's not linearly separable



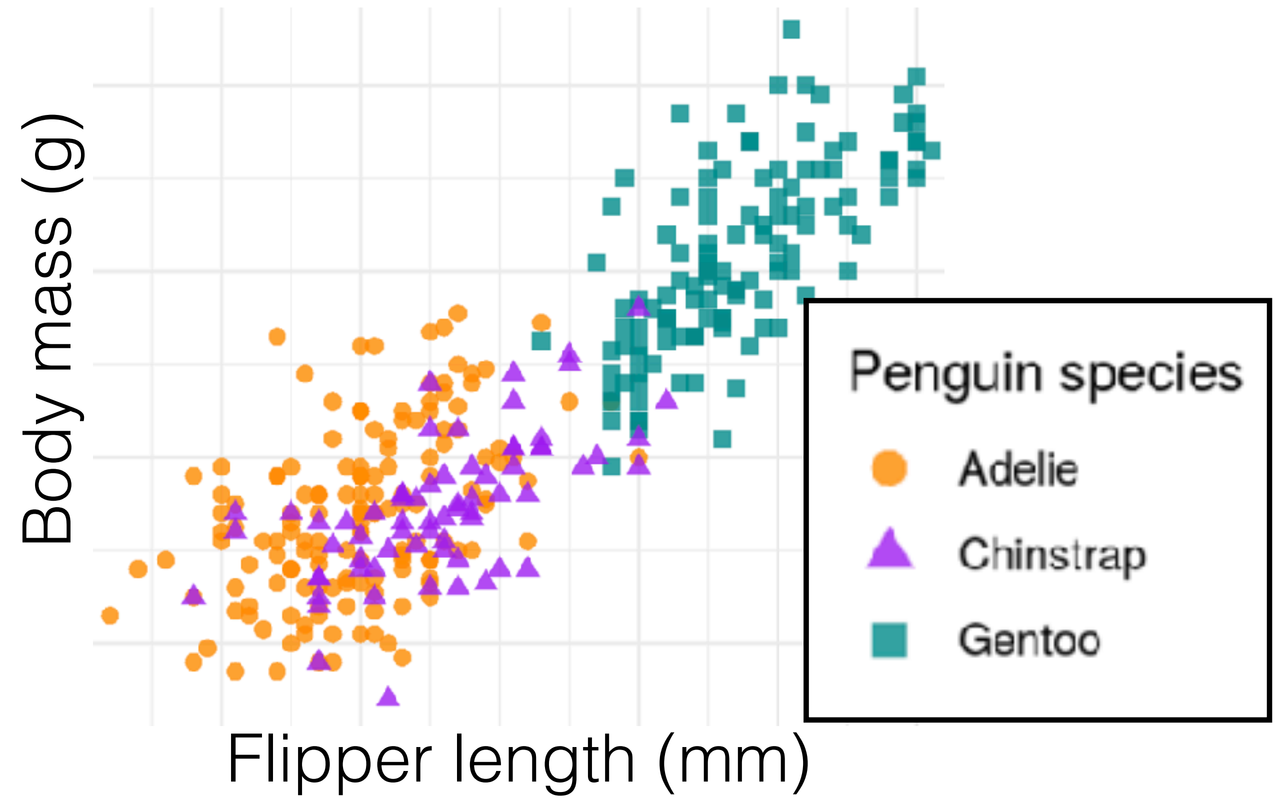
Notice

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



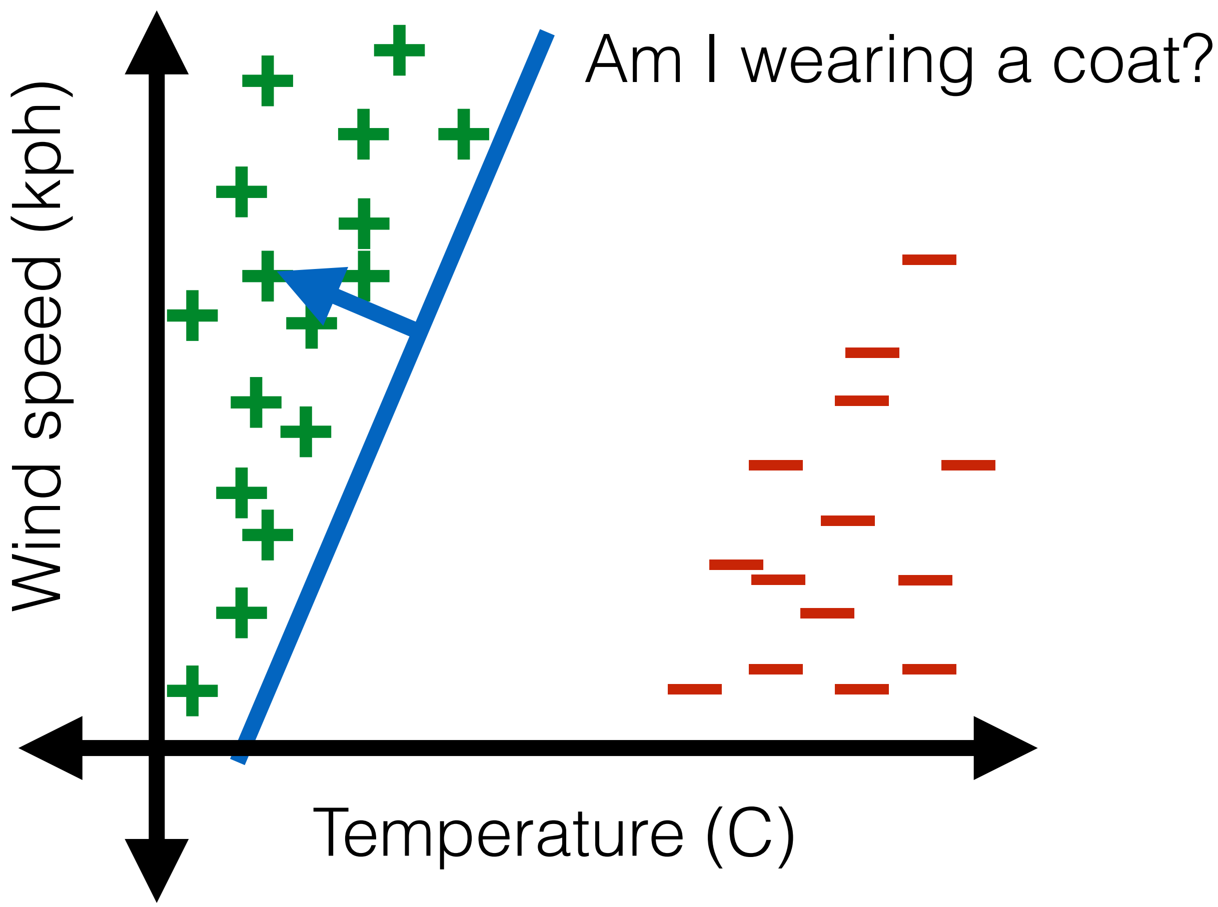
Recall

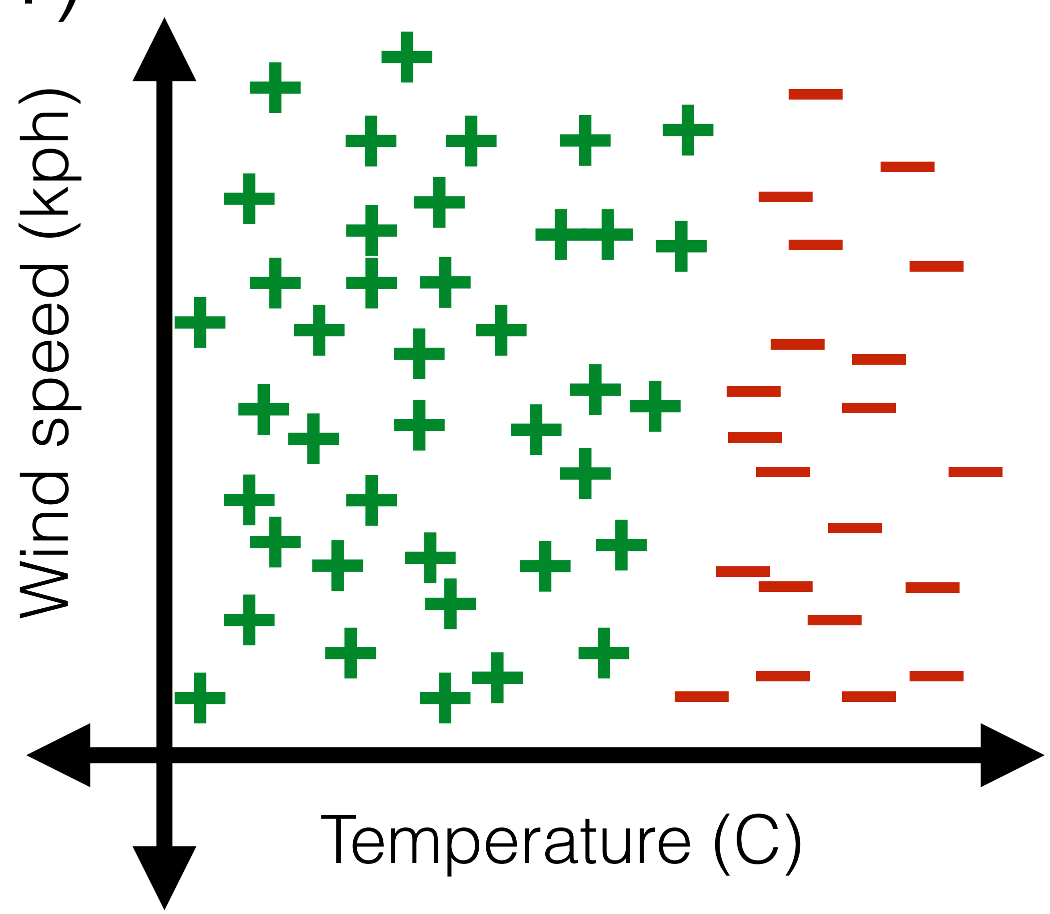
- Perceptron struggles with data that's not linearly separable



Notice

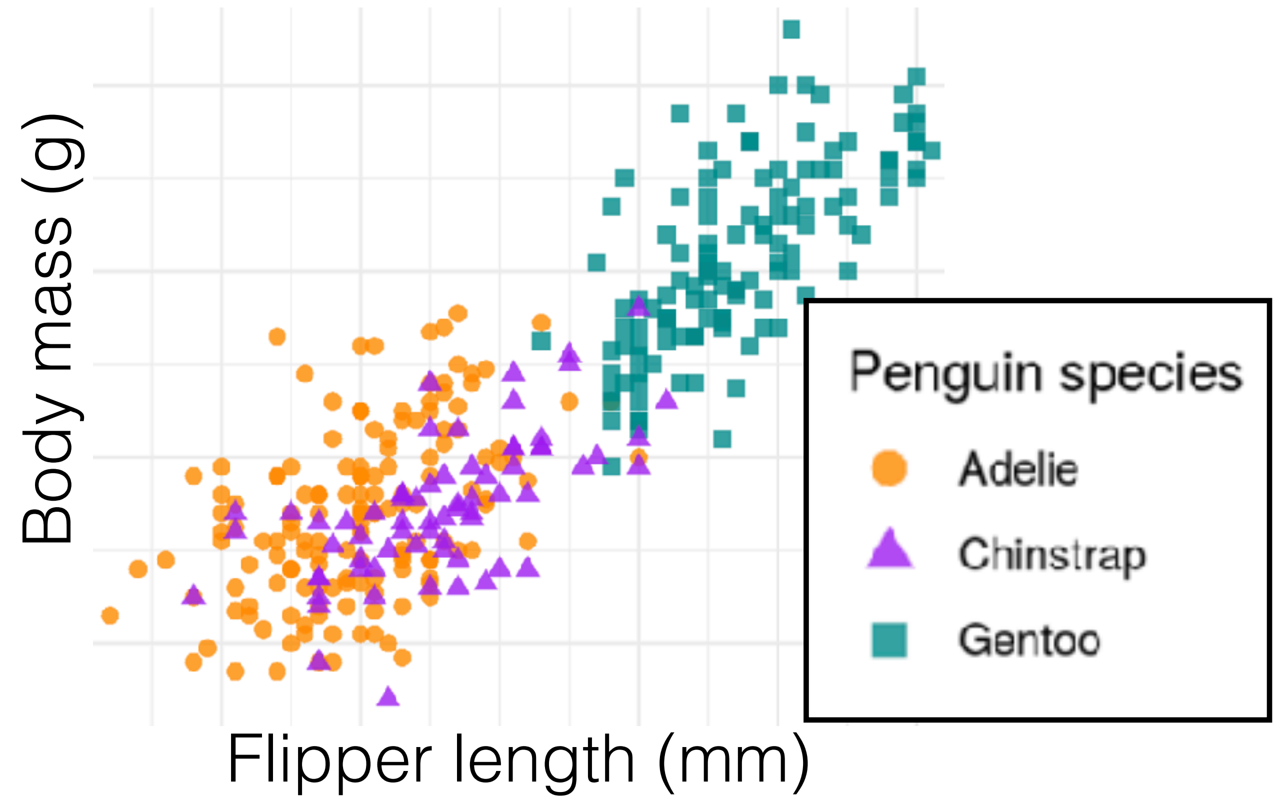
- Perceptron doesn't have a notion of uncertainty (how well do we know what we know?)





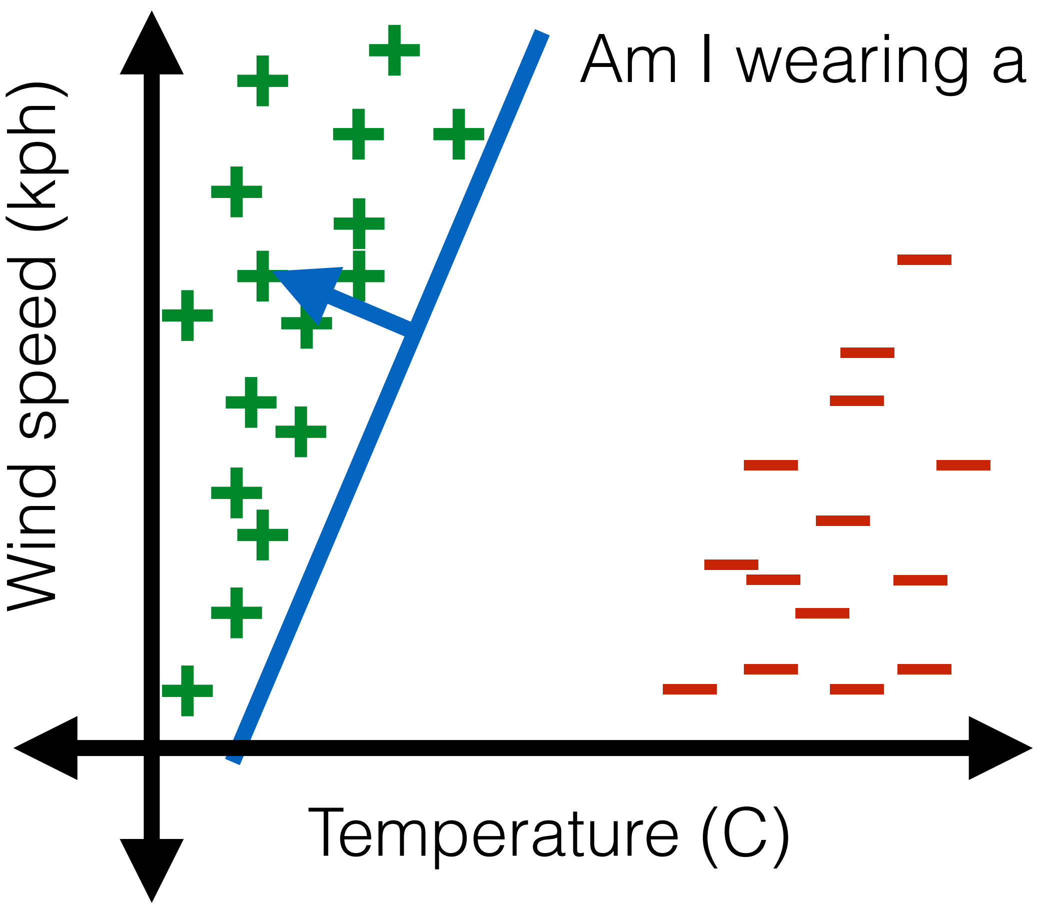
Recall

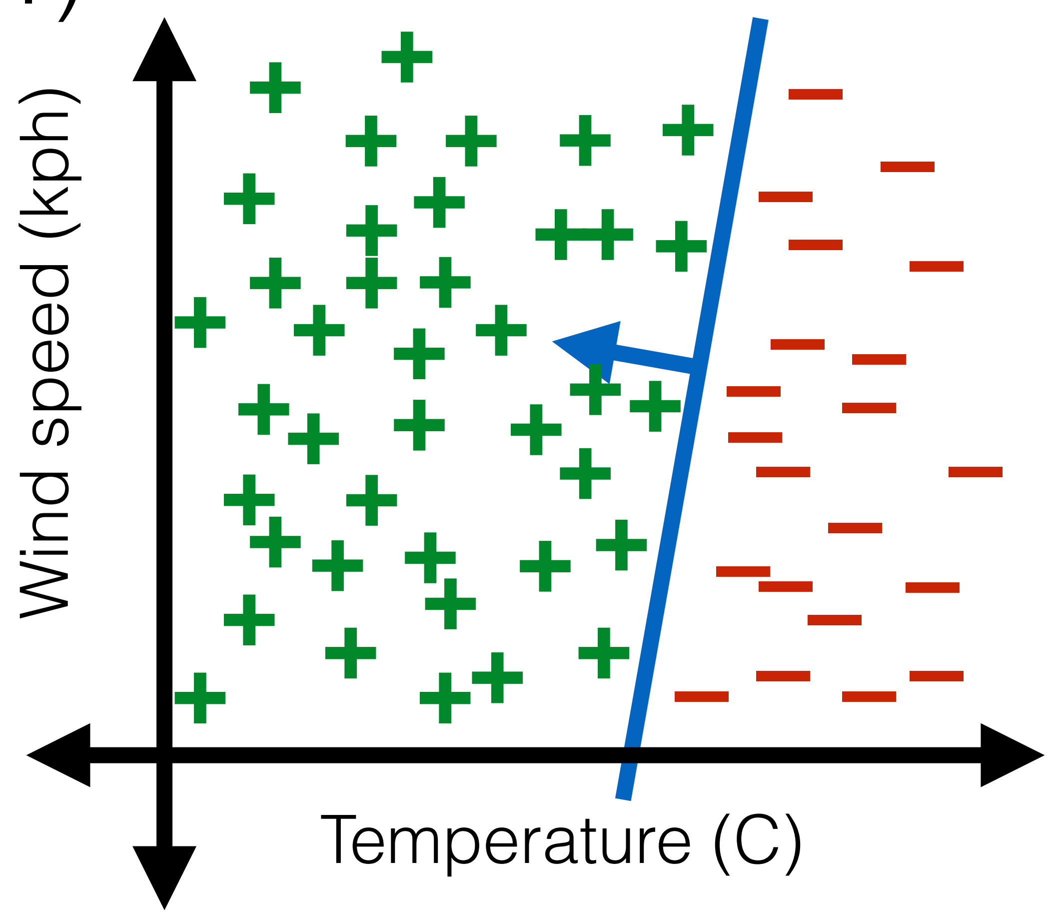
- Perceptron struggles with data that's not linearly separable



Notice

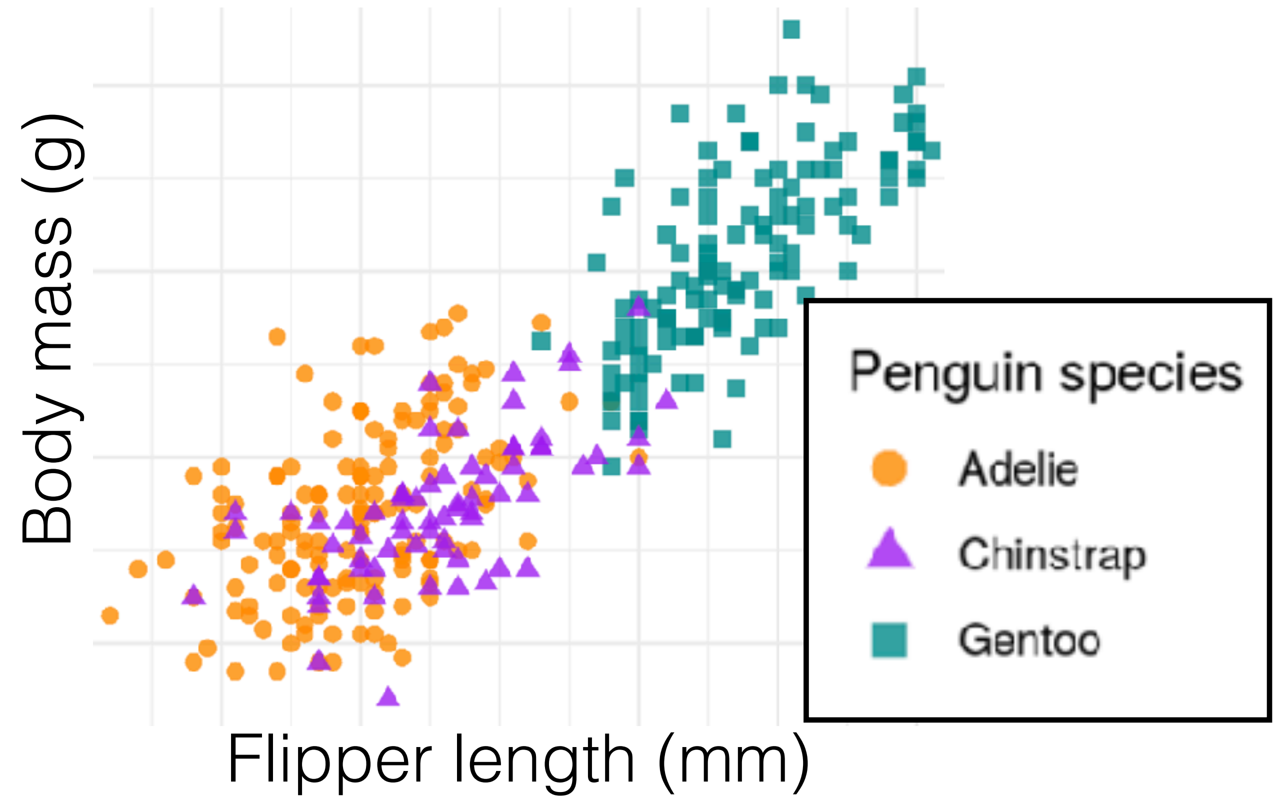
- Perceptron doesn't have a notion of uncertainty (how well do we know what we know?)





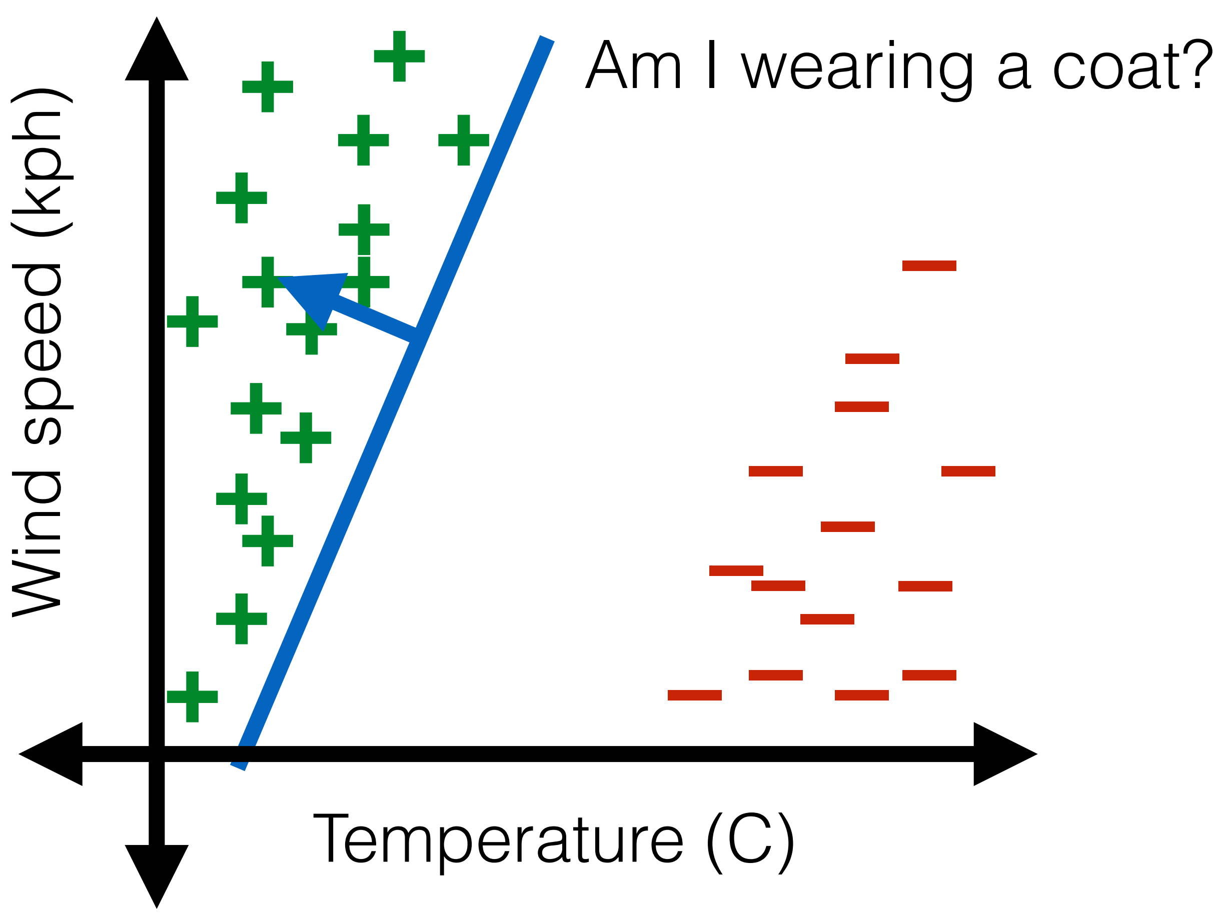
Recall

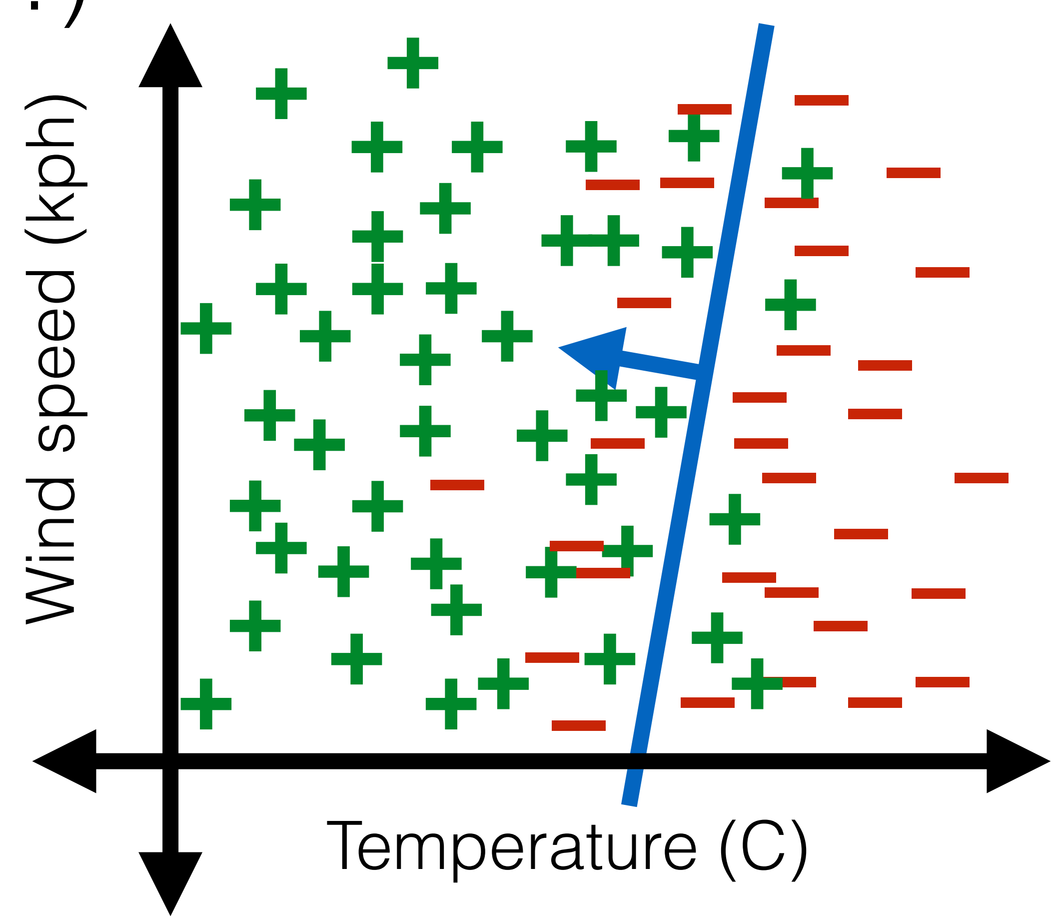
- Perceptron struggles with data that's not linearly separable



Notice

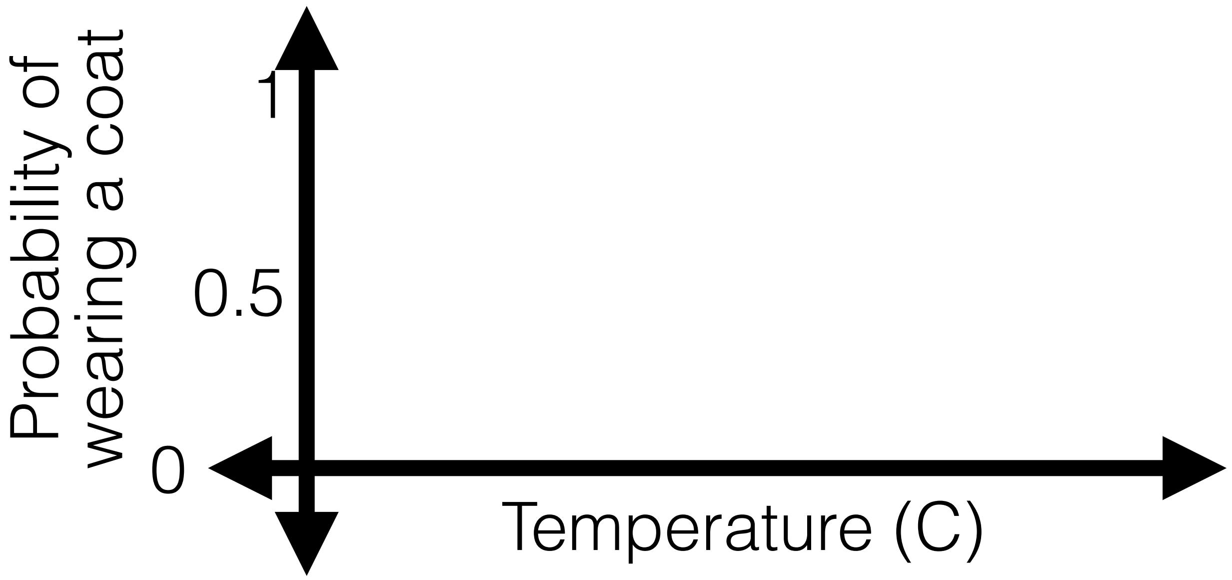
- Perceptron doesn't have a notion of uncertainty (how well do we know what we know?)



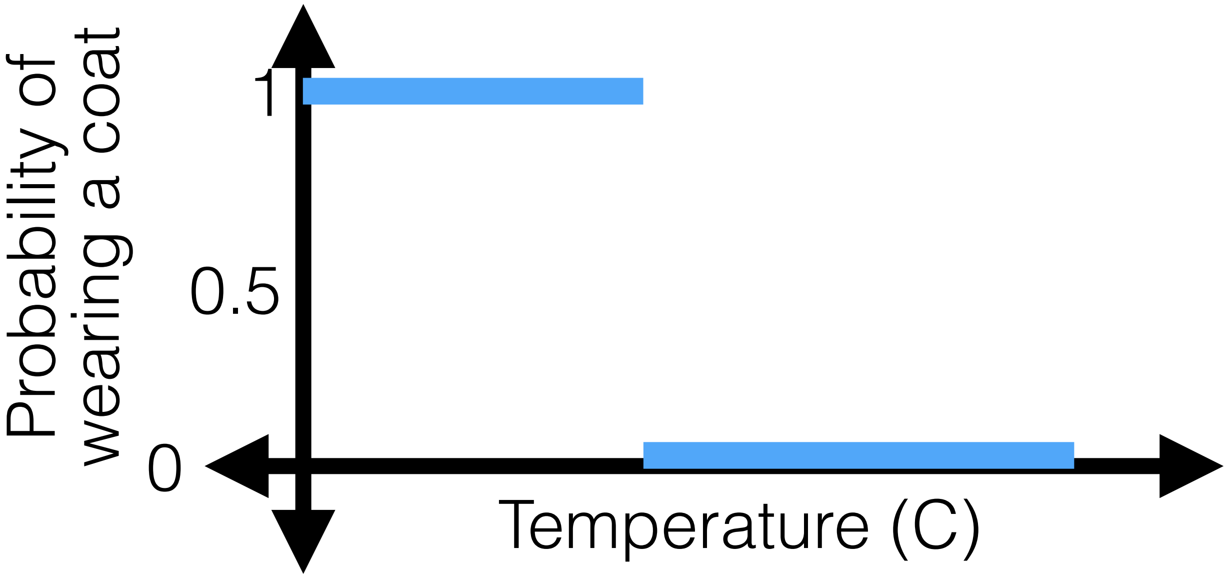


Capturing uncertainty

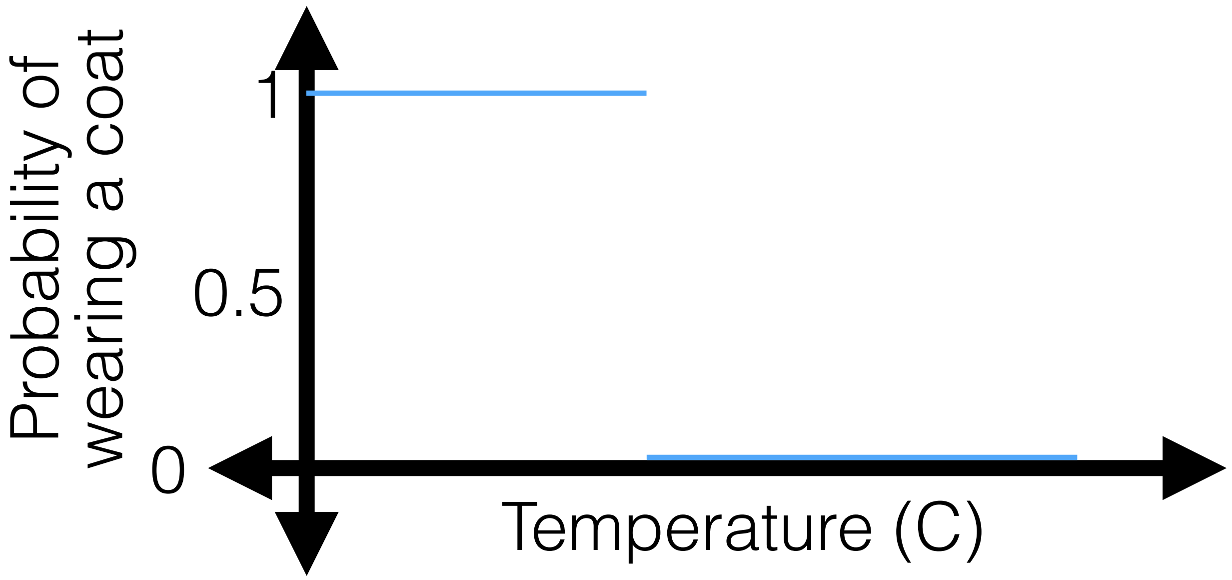
Capturing uncertainty



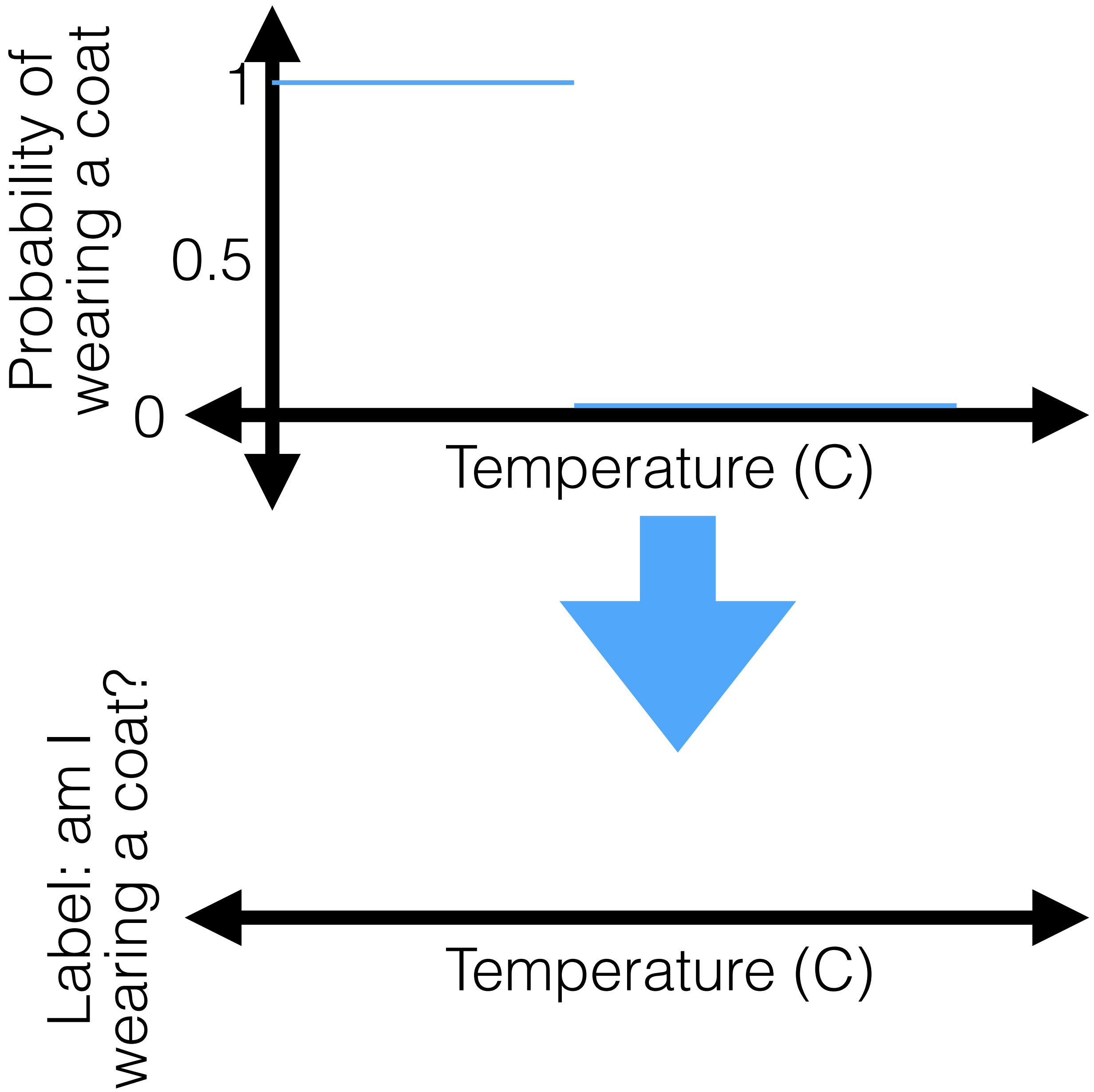
Capturing uncertainty



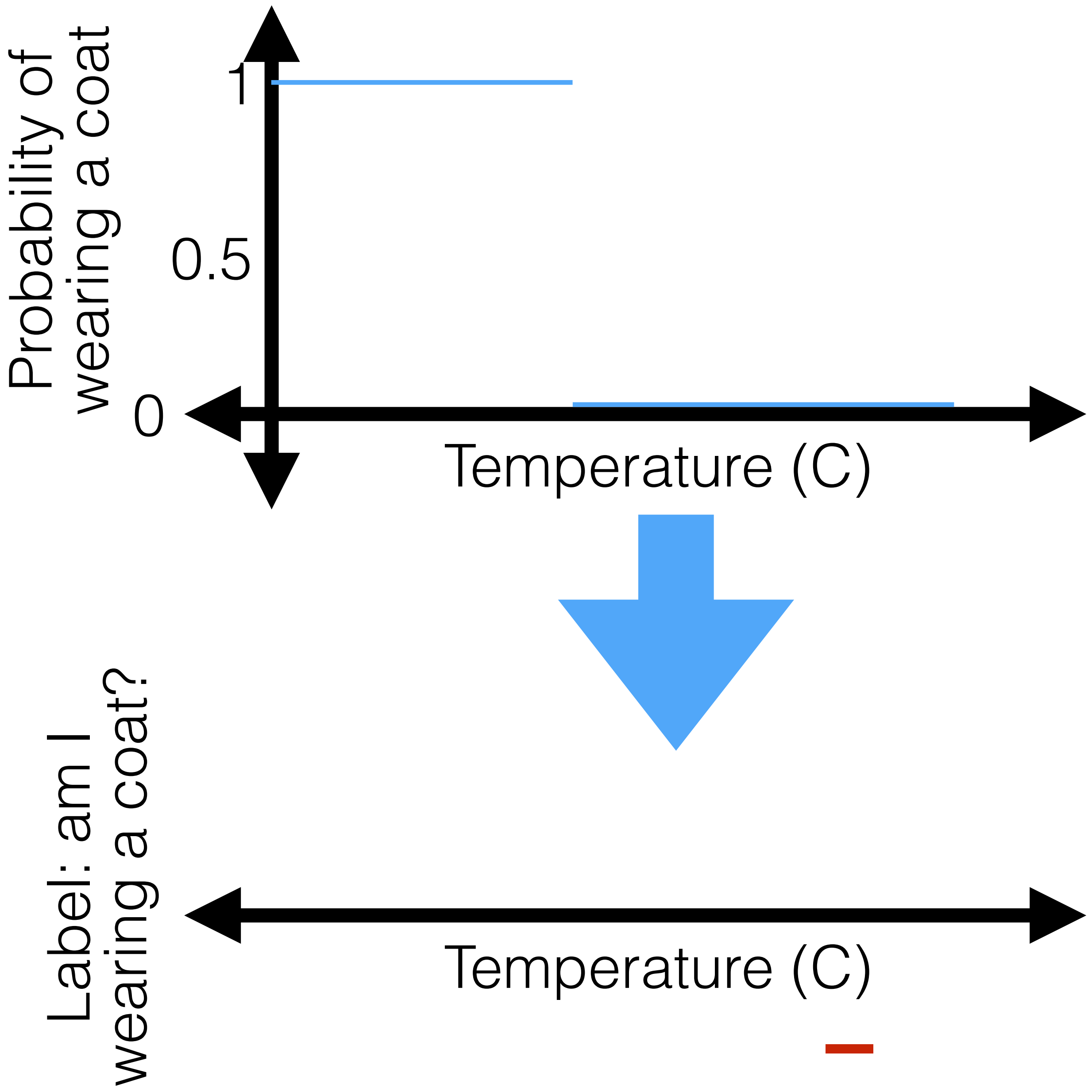
Capturing uncertainty



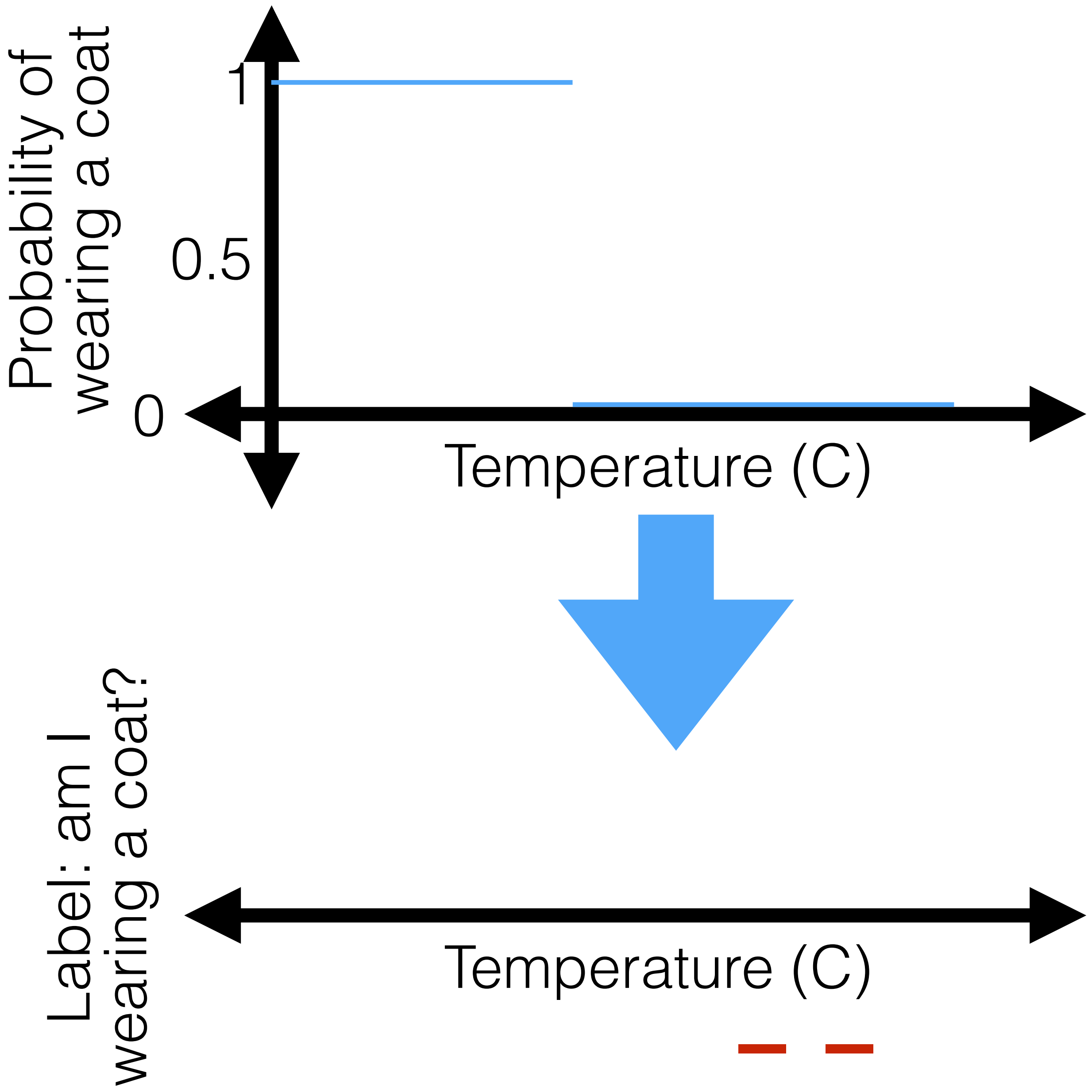
Capturing uncertainty



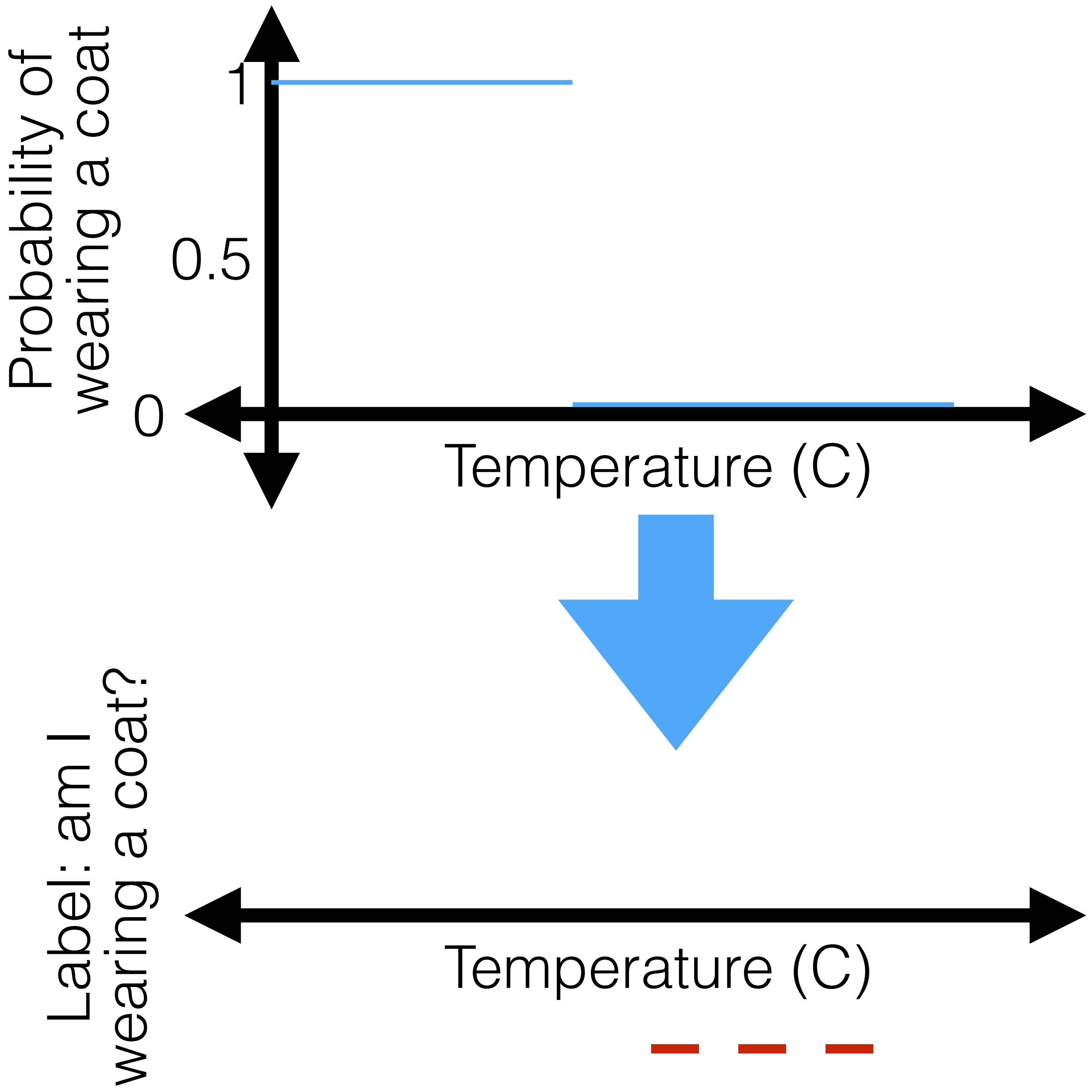
Capturing uncertainty



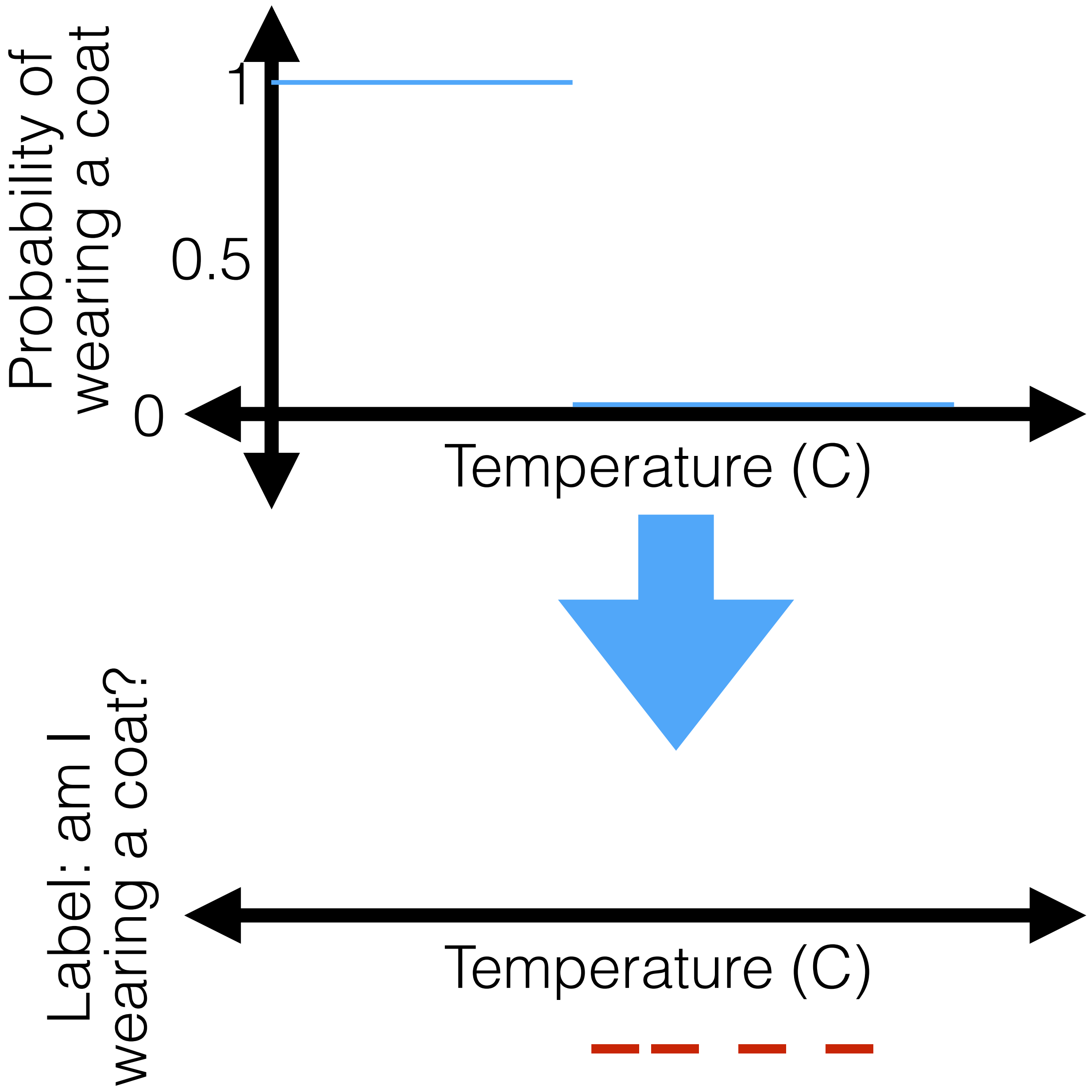
Capturing uncertainty



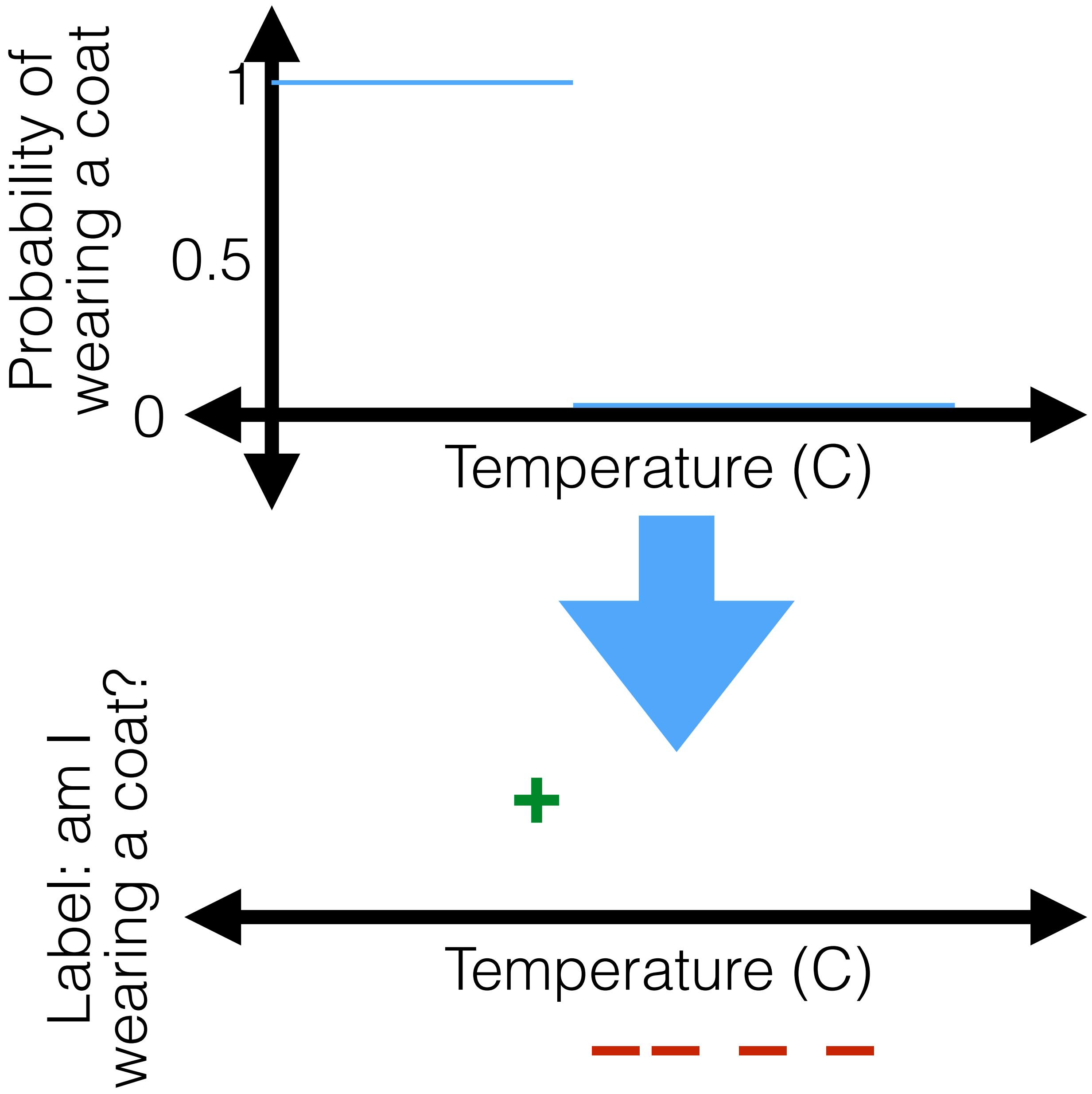
Capturing uncertainty



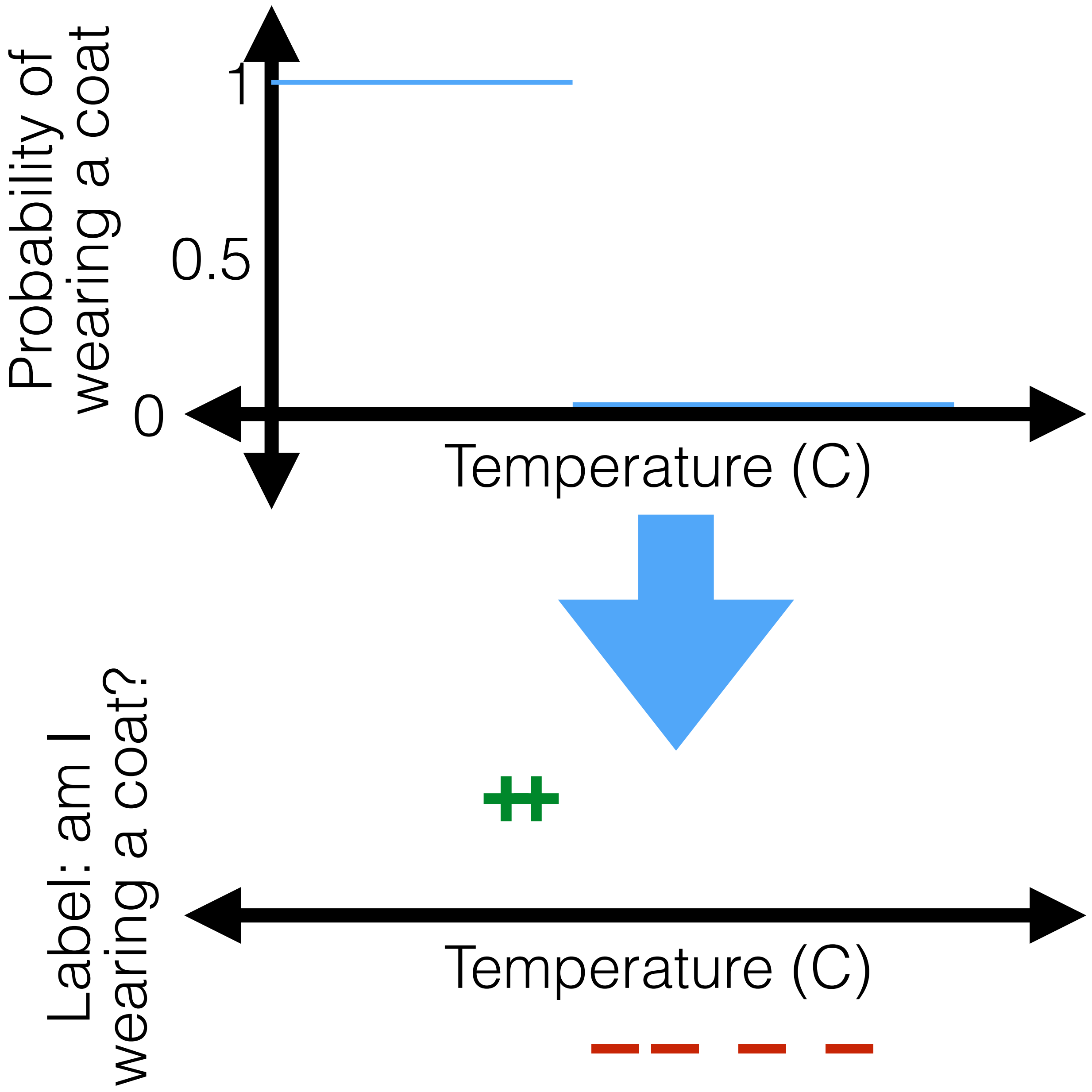
Capturing uncertainty



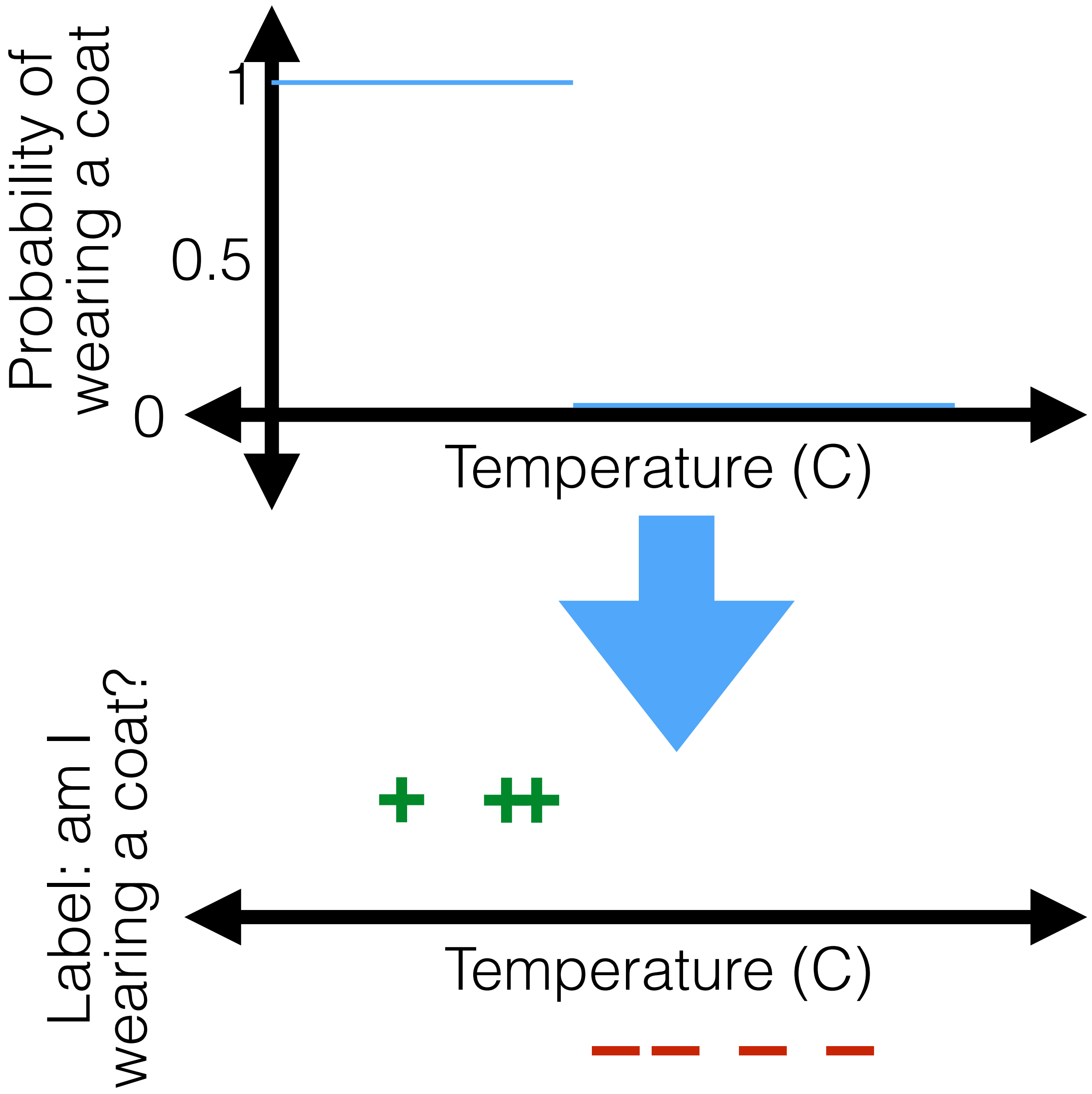
Capturing uncertainty



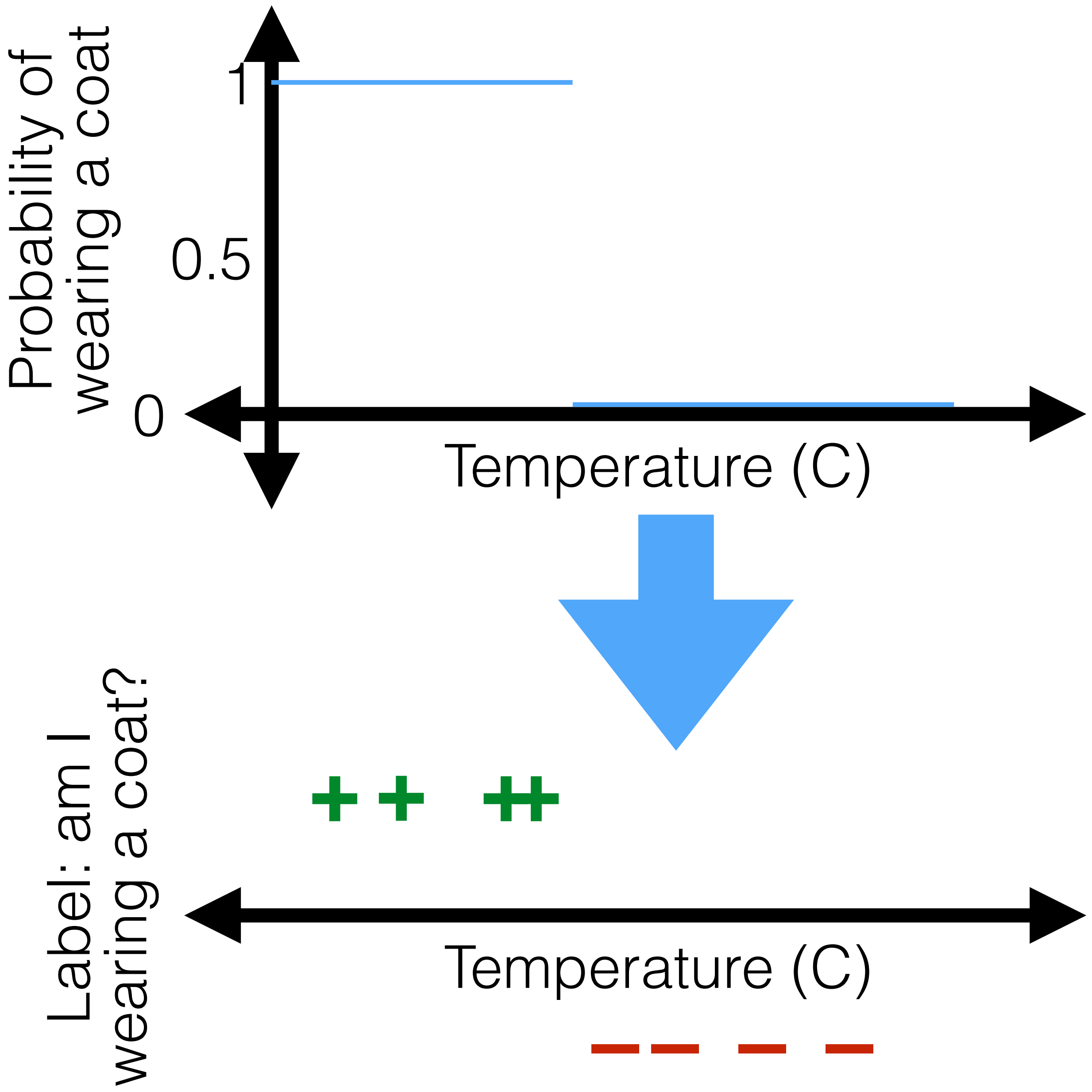
Capturing uncertainty



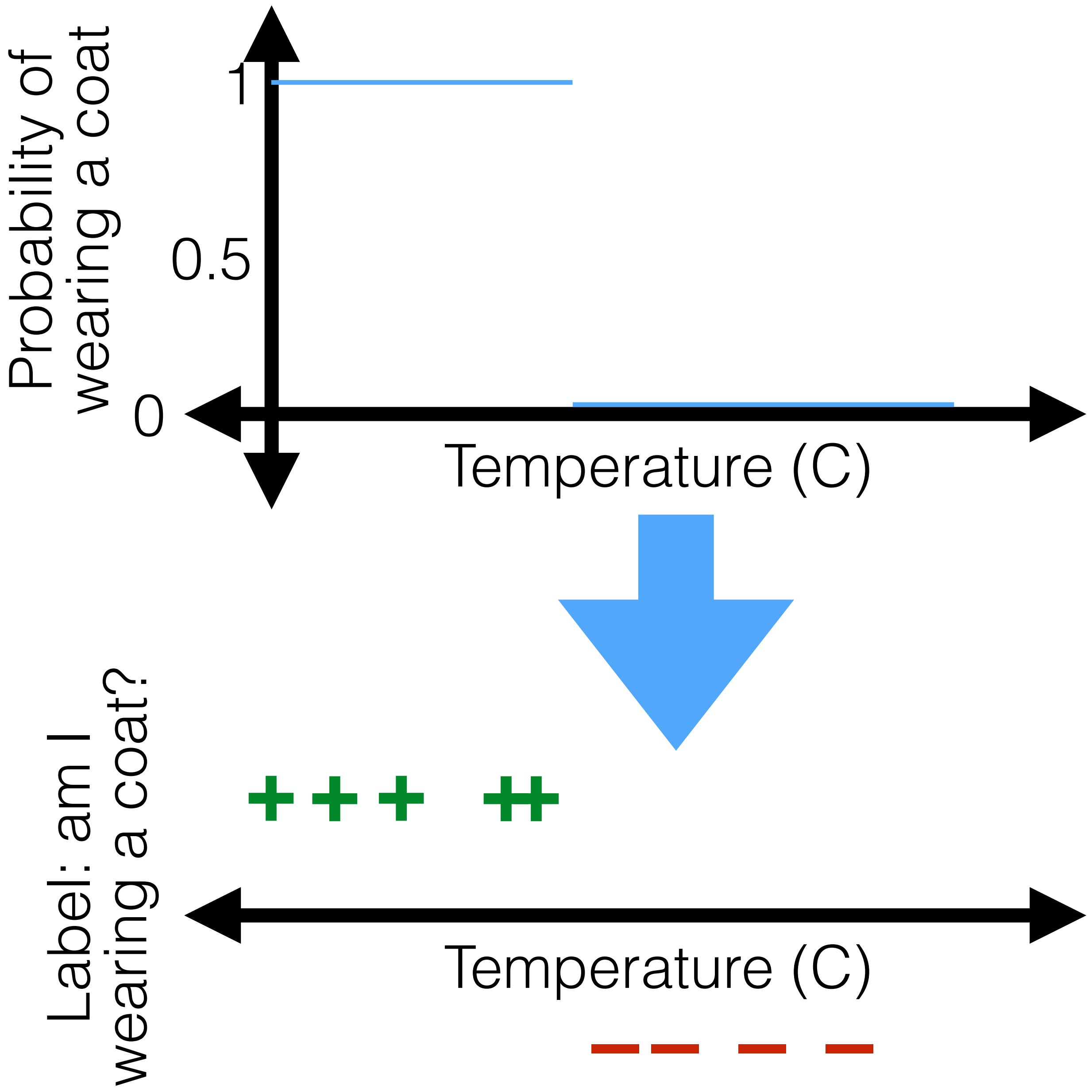
Capturing uncertainty



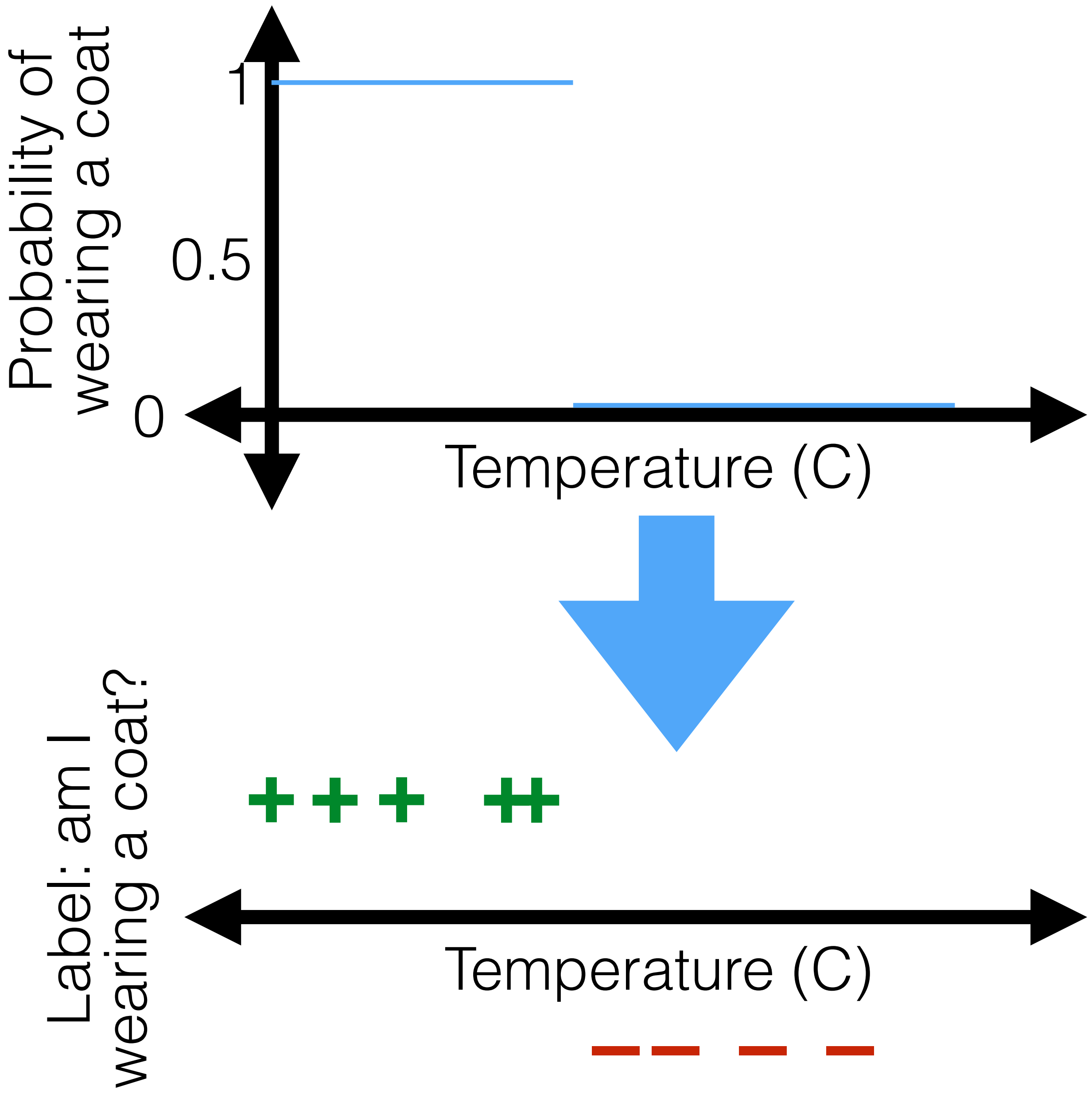
Capturing uncertainty



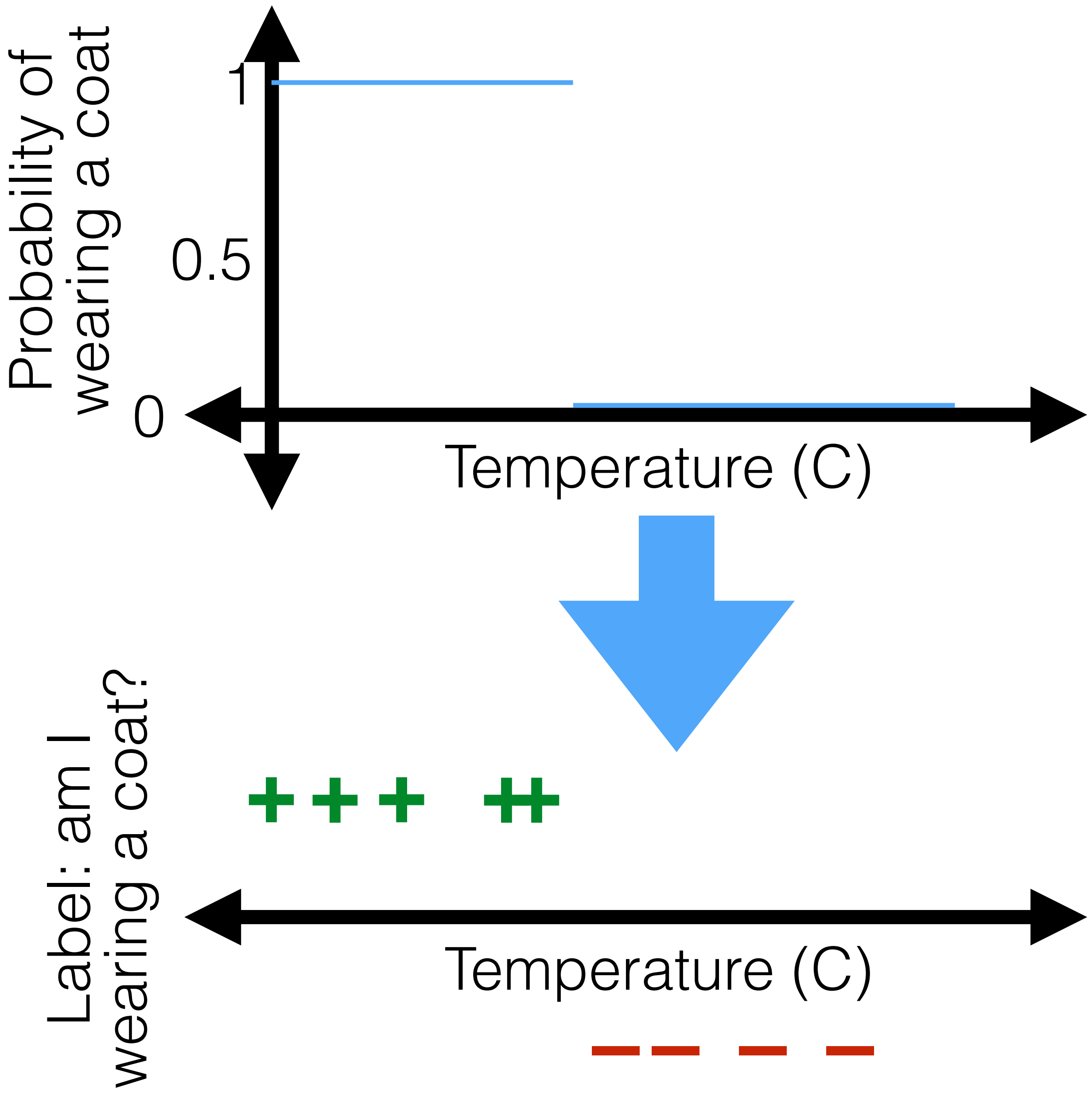
Capturing uncertainty

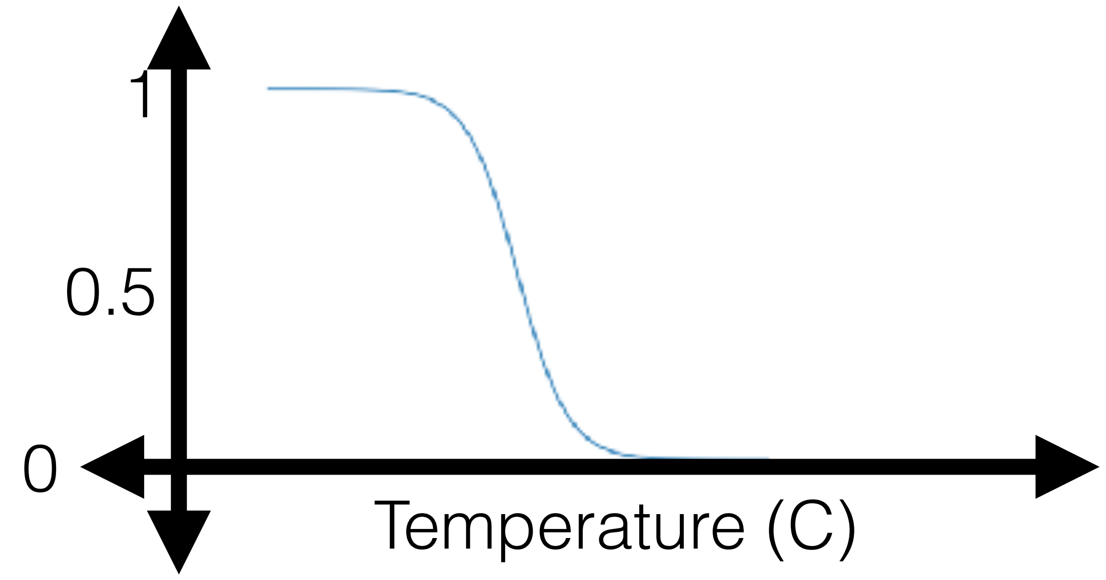


Capturing uncertainty

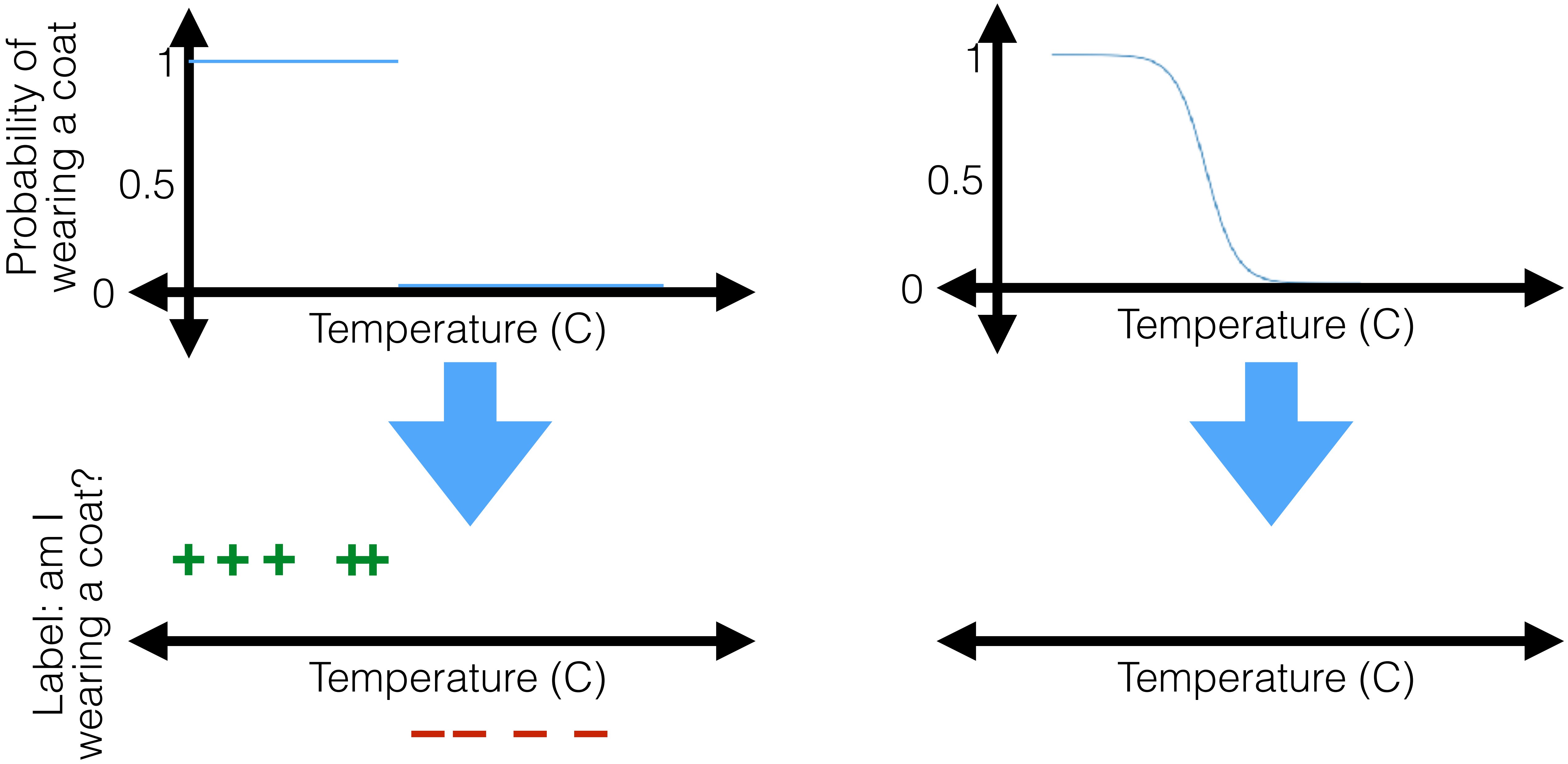


Capturing uncertainty

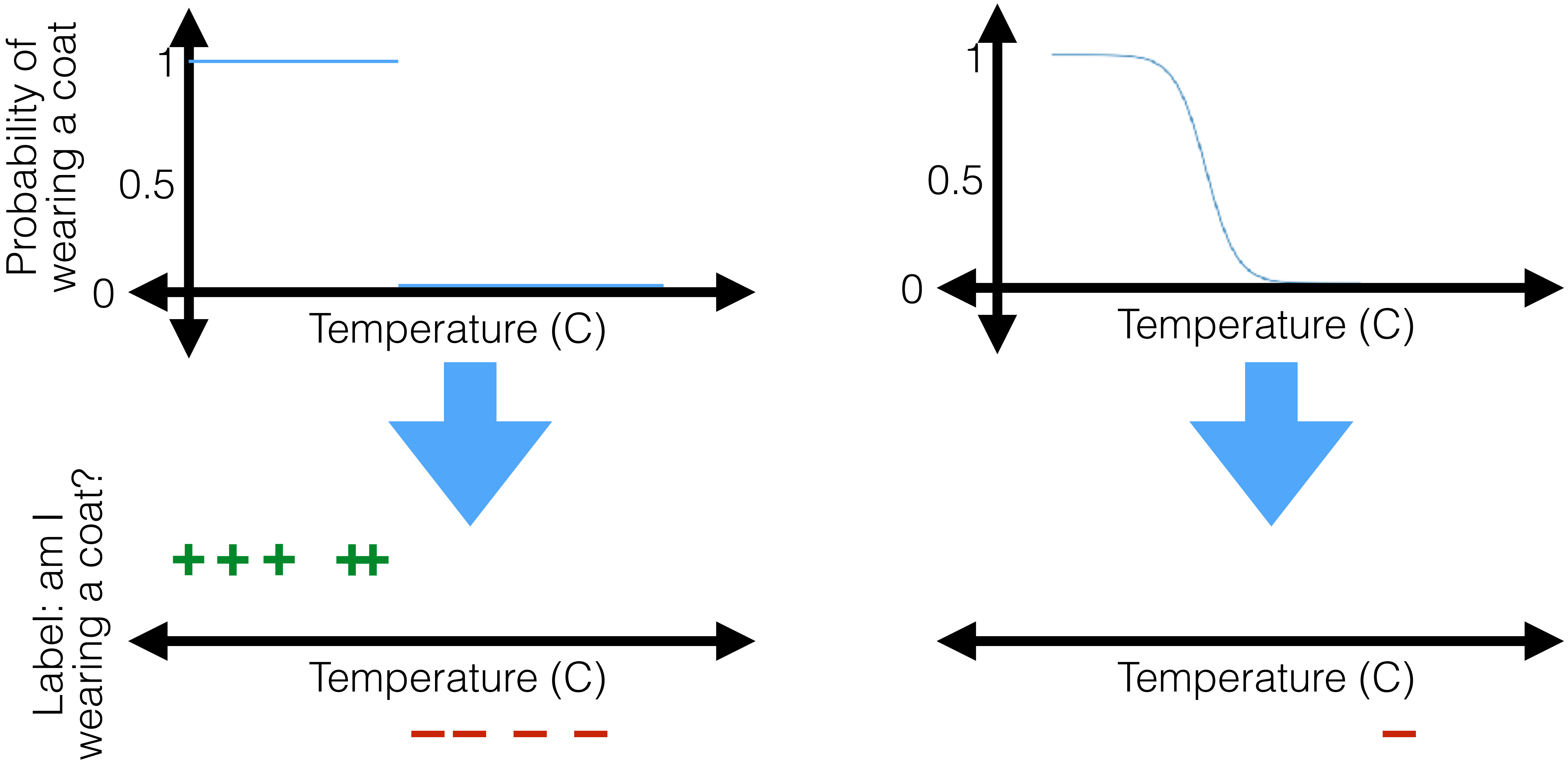




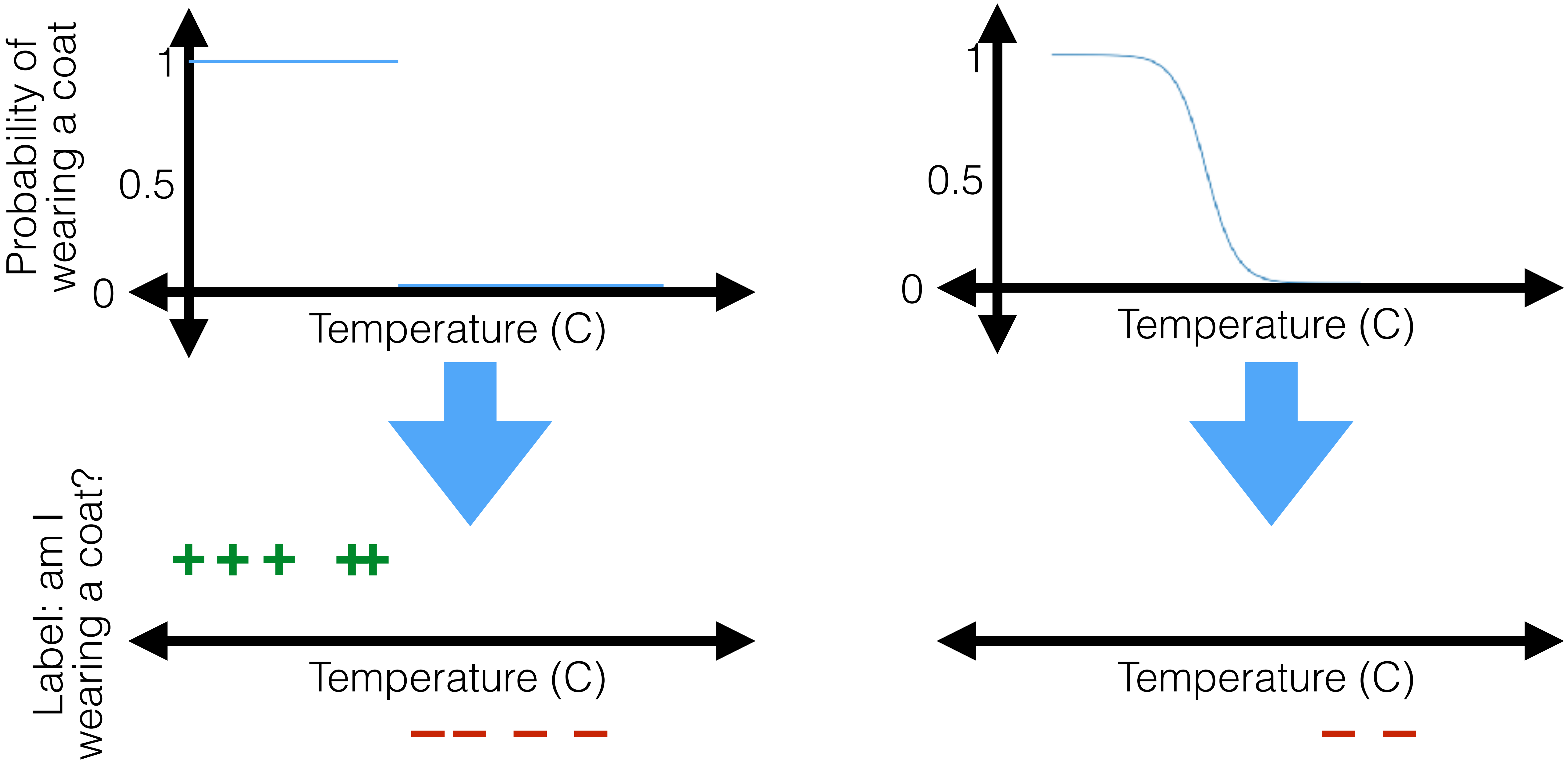
Capturing uncertainty



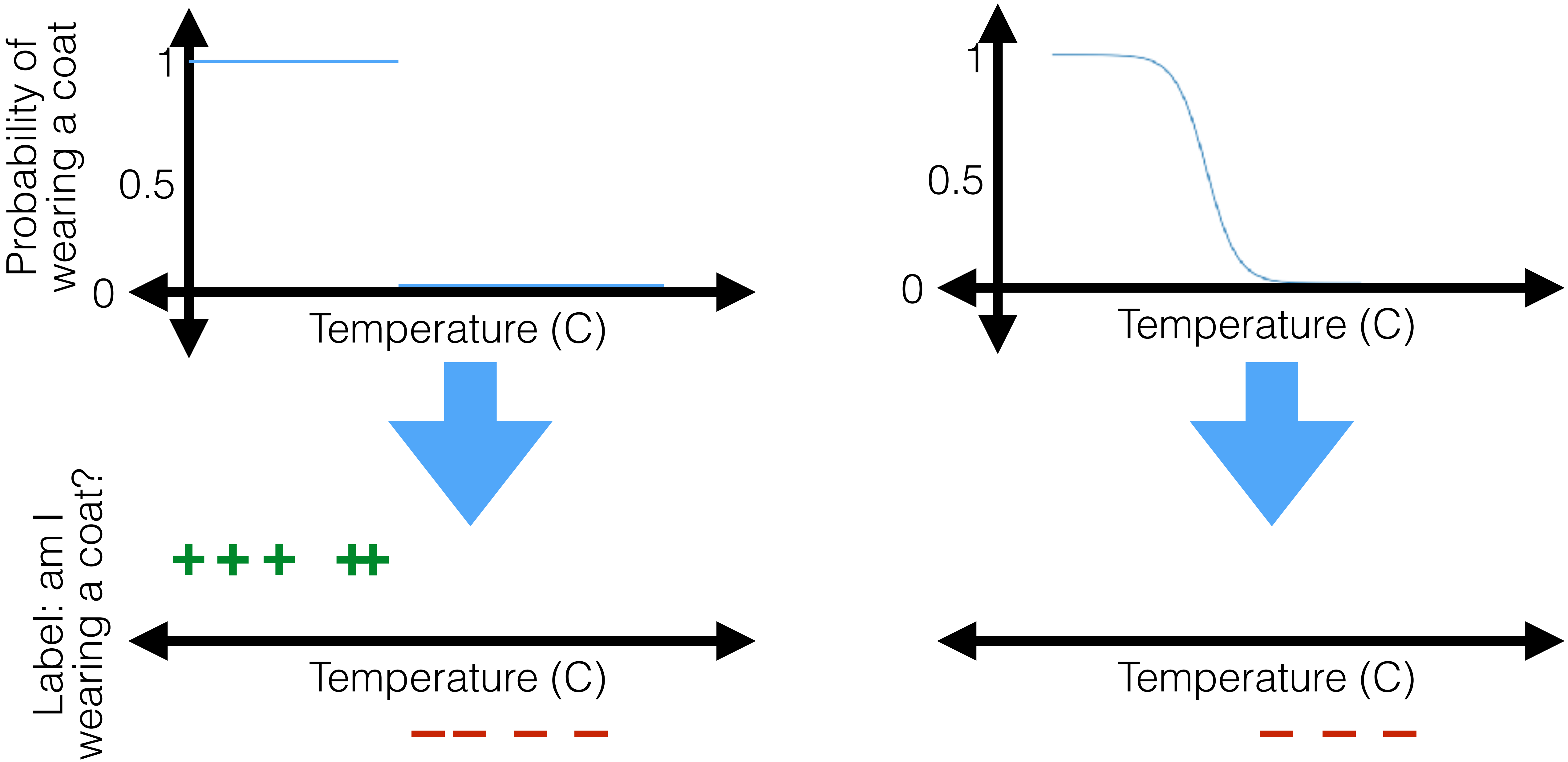
Capturing uncertainty



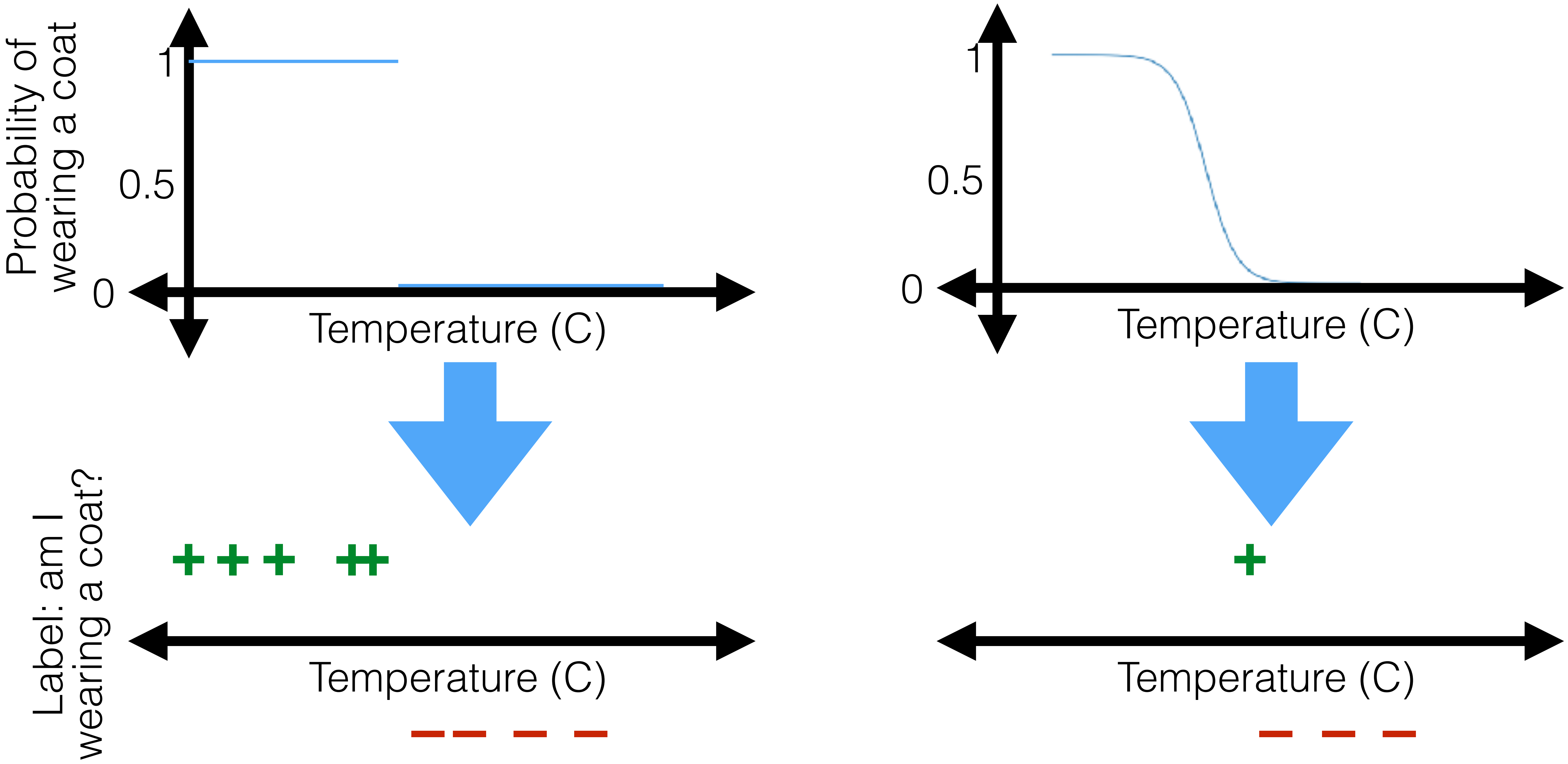
Capturing uncertainty



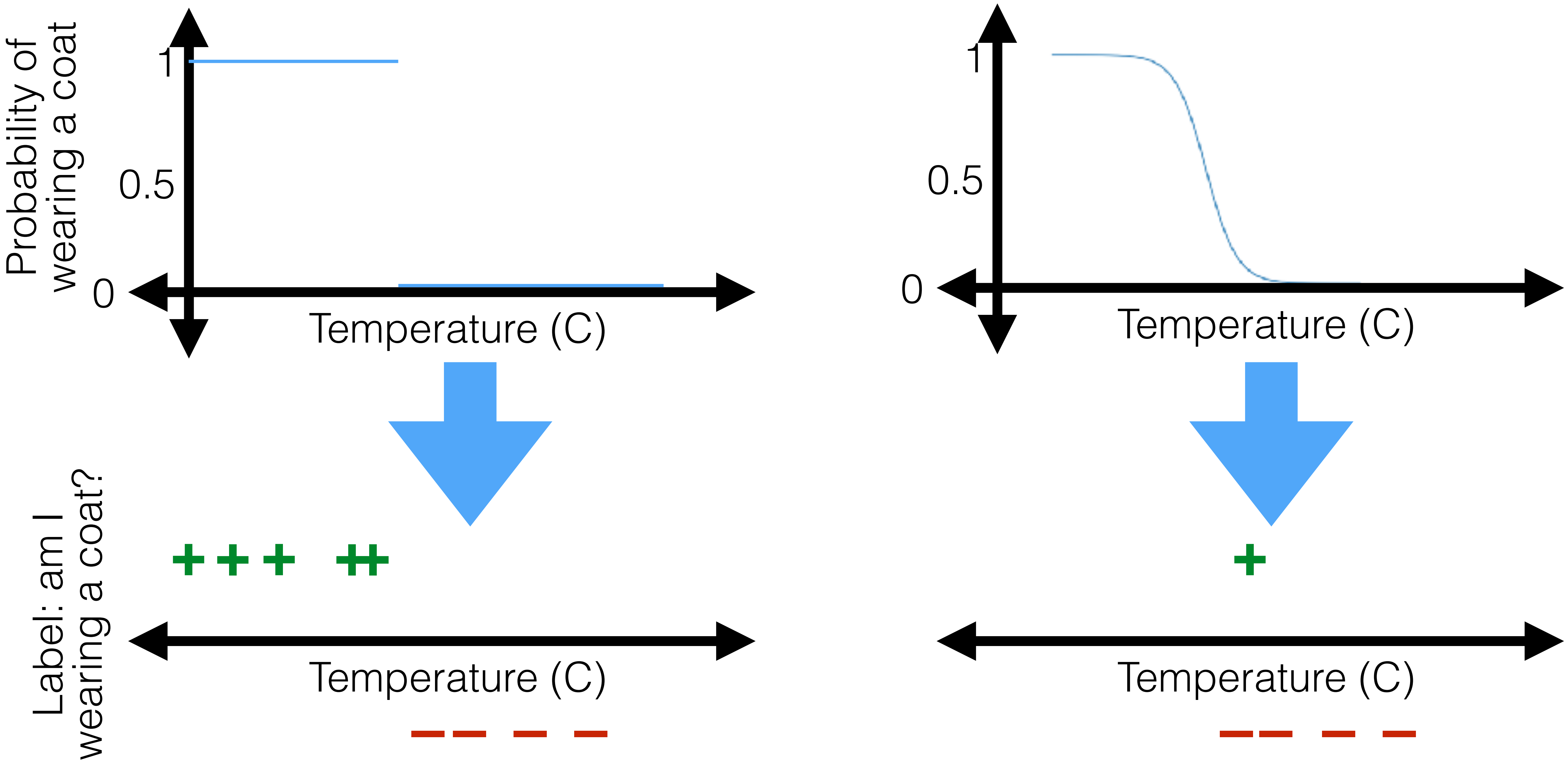
Capturing uncertainty



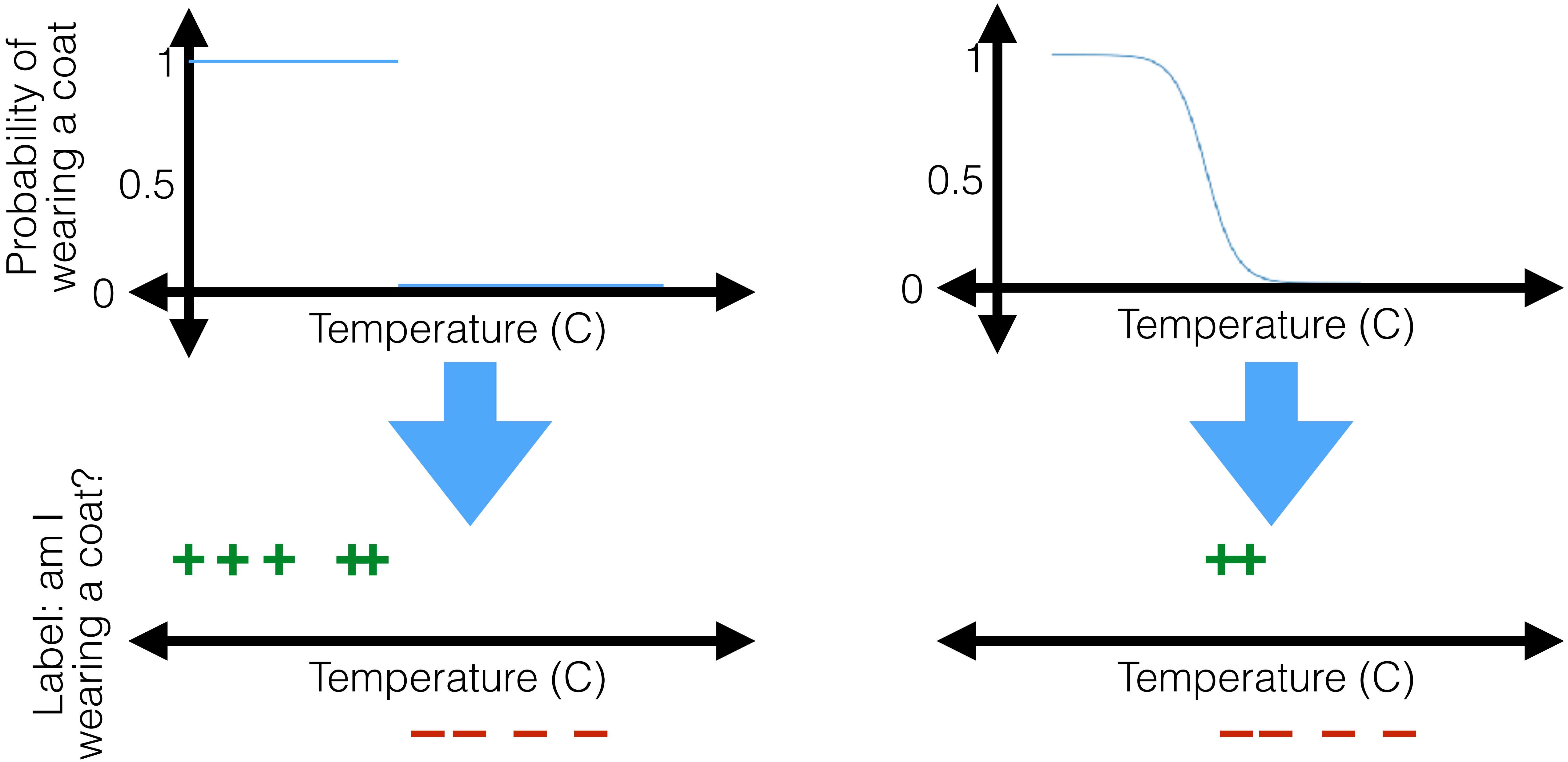
Capturing uncertainty



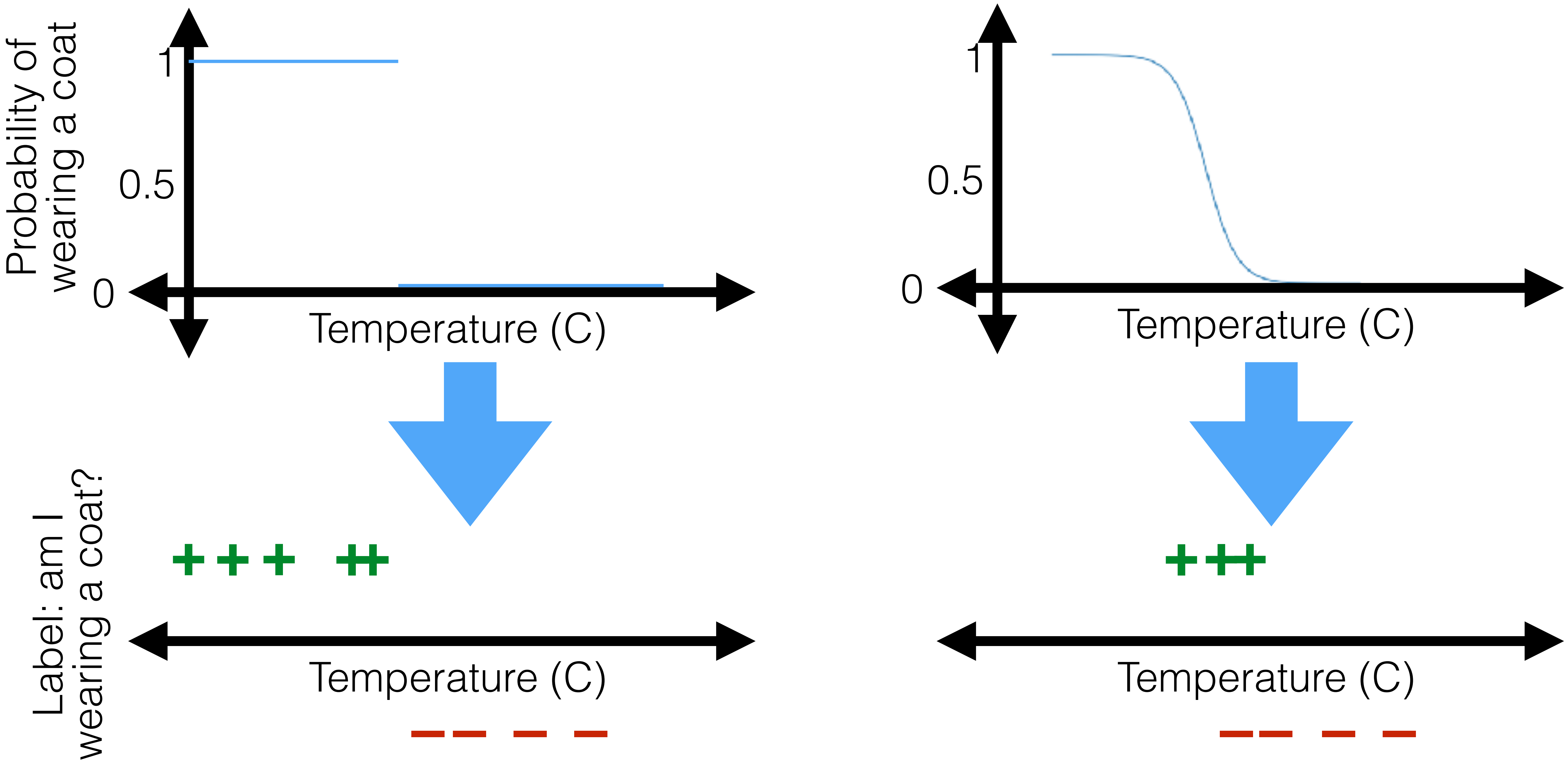
Capturing uncertainty



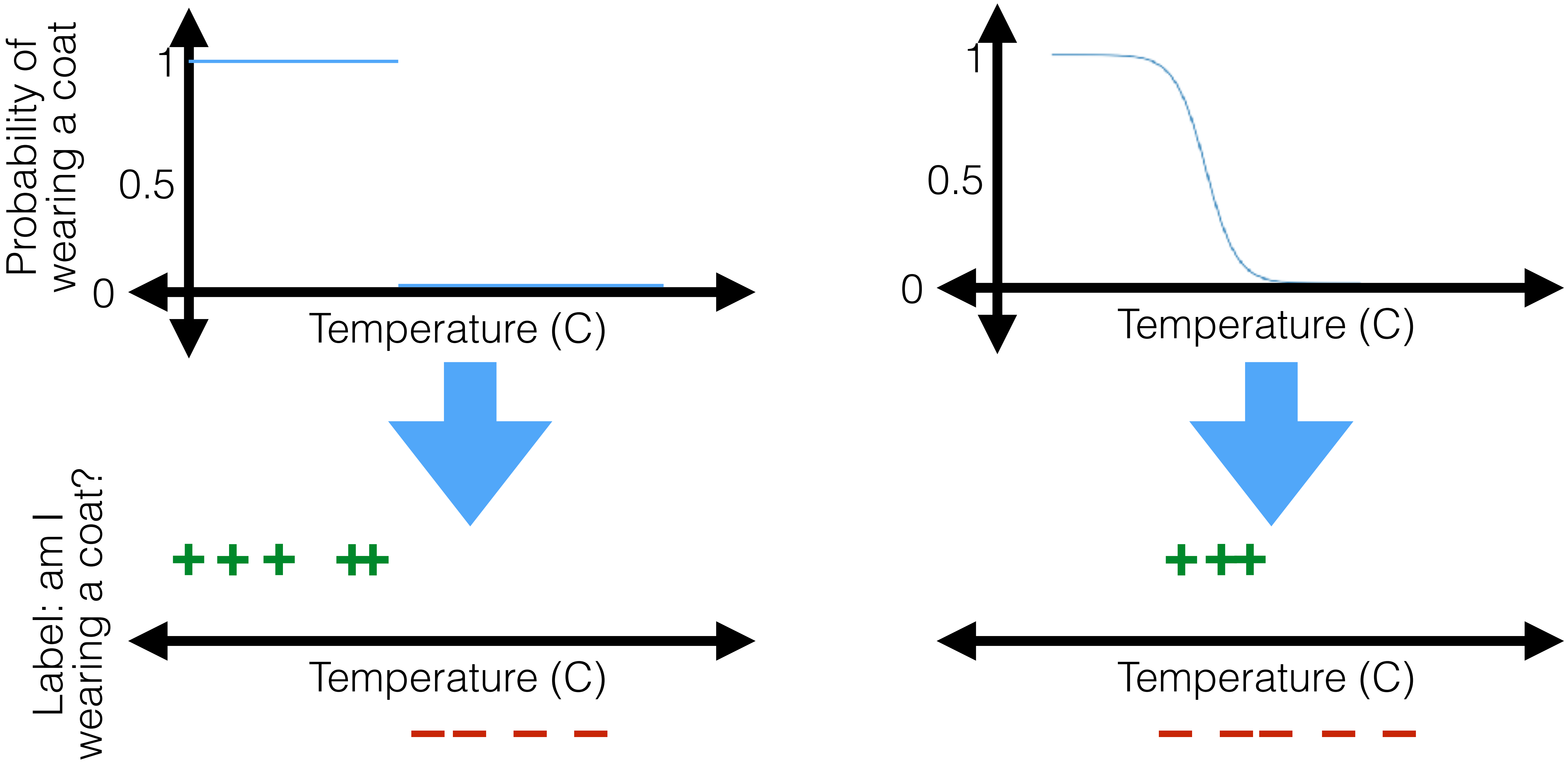
Capturing uncertainty



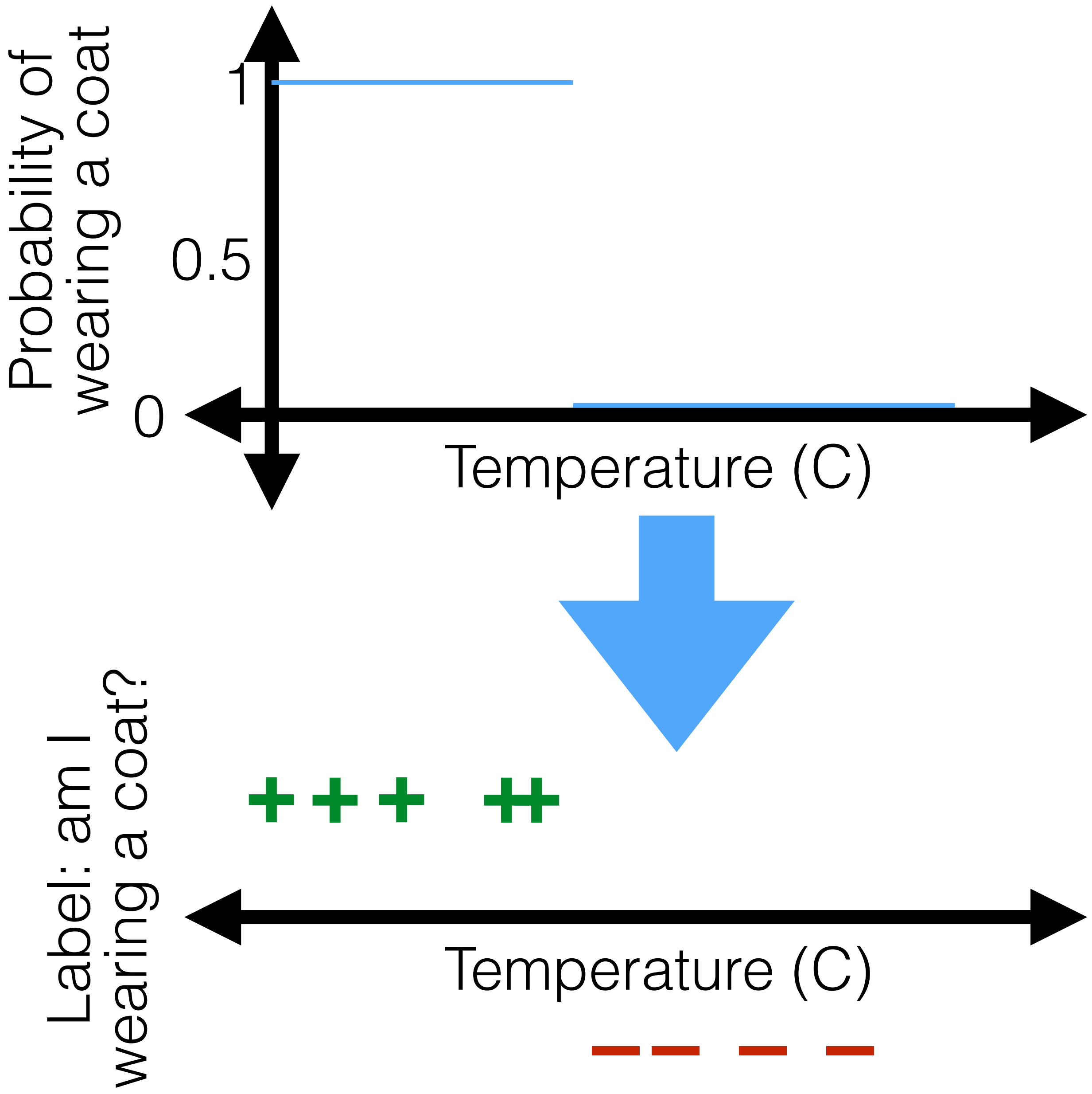
Capturing uncertainty

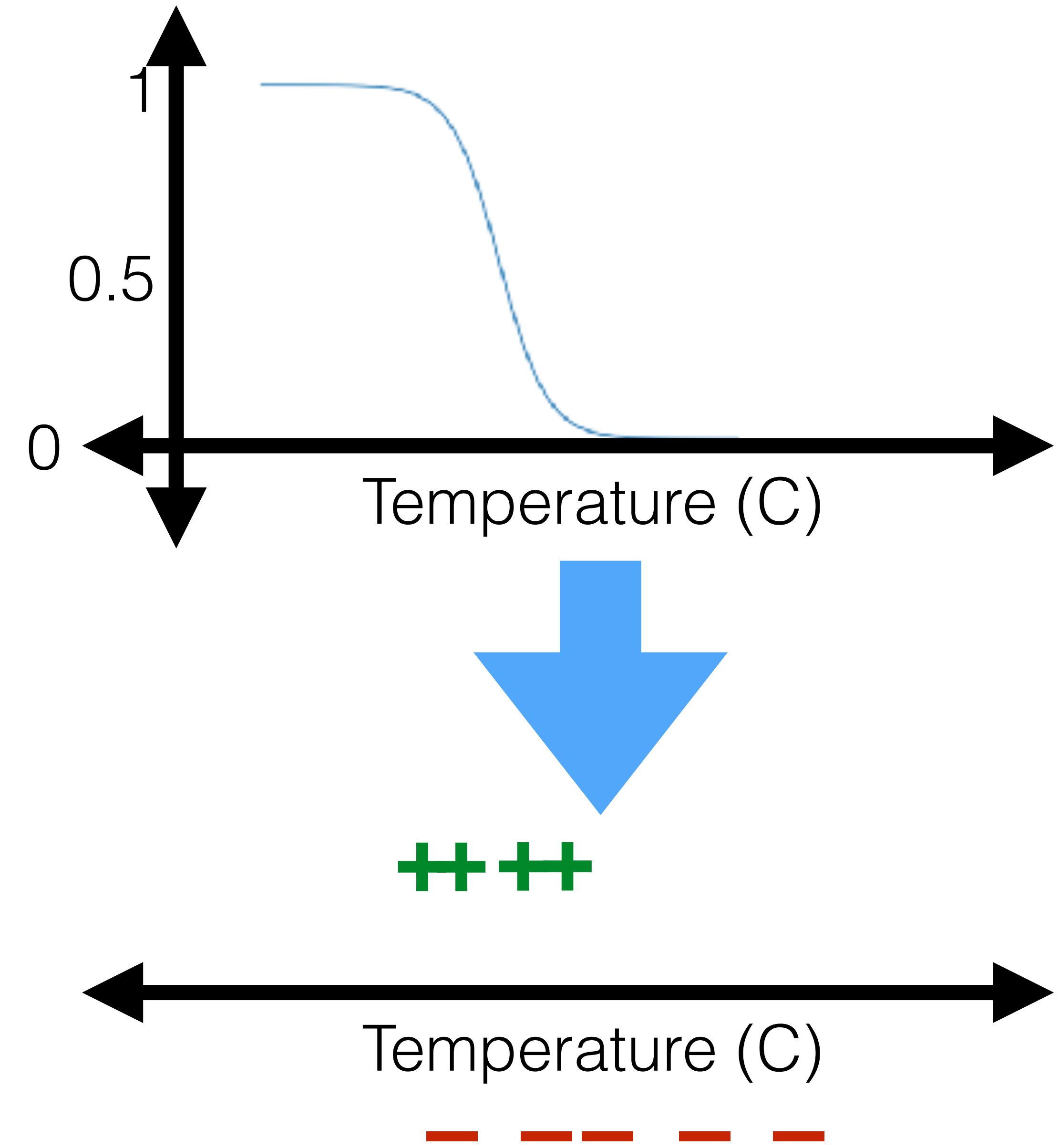


Capturing uncertainty

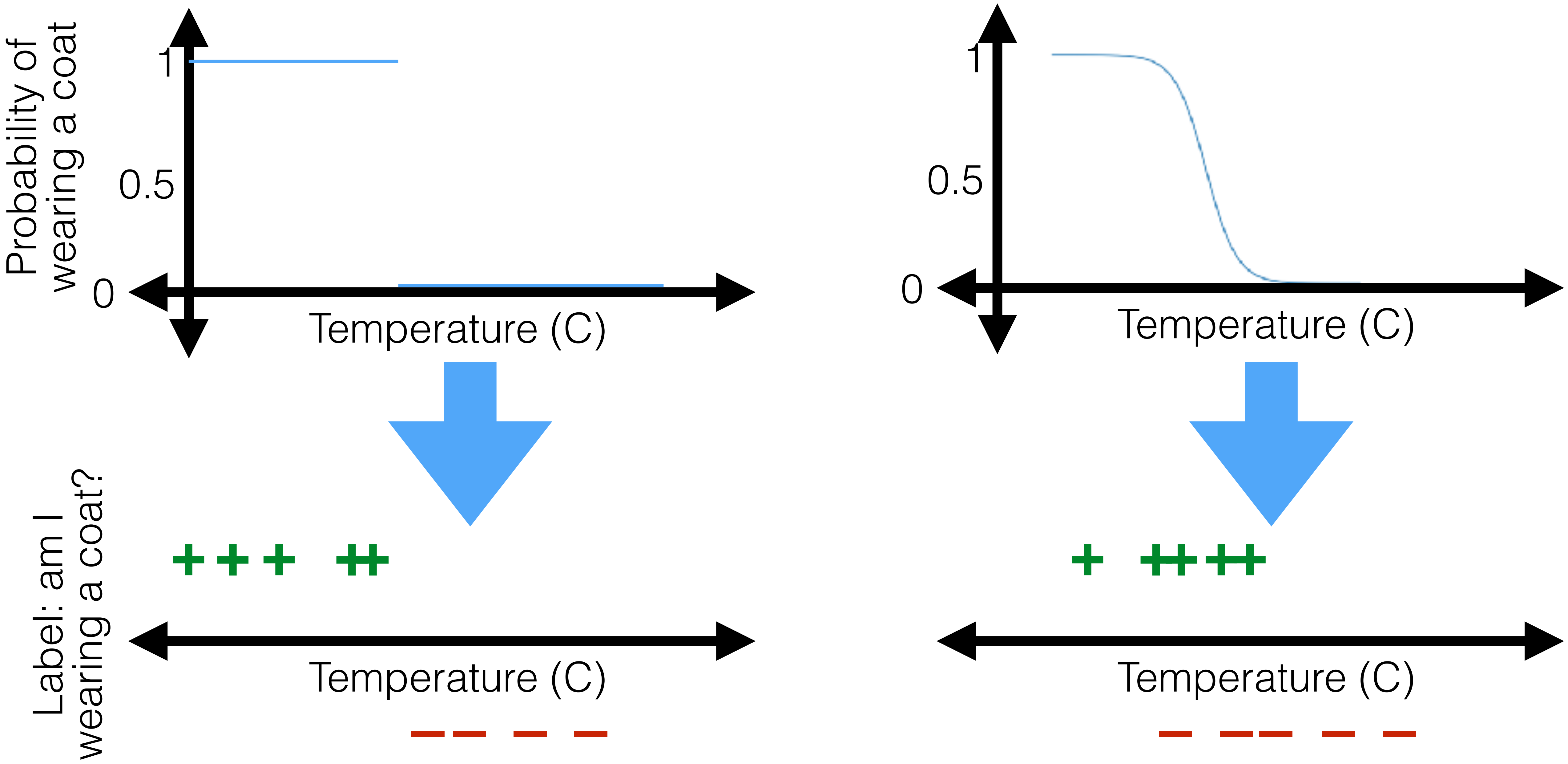


Capturing uncertainty

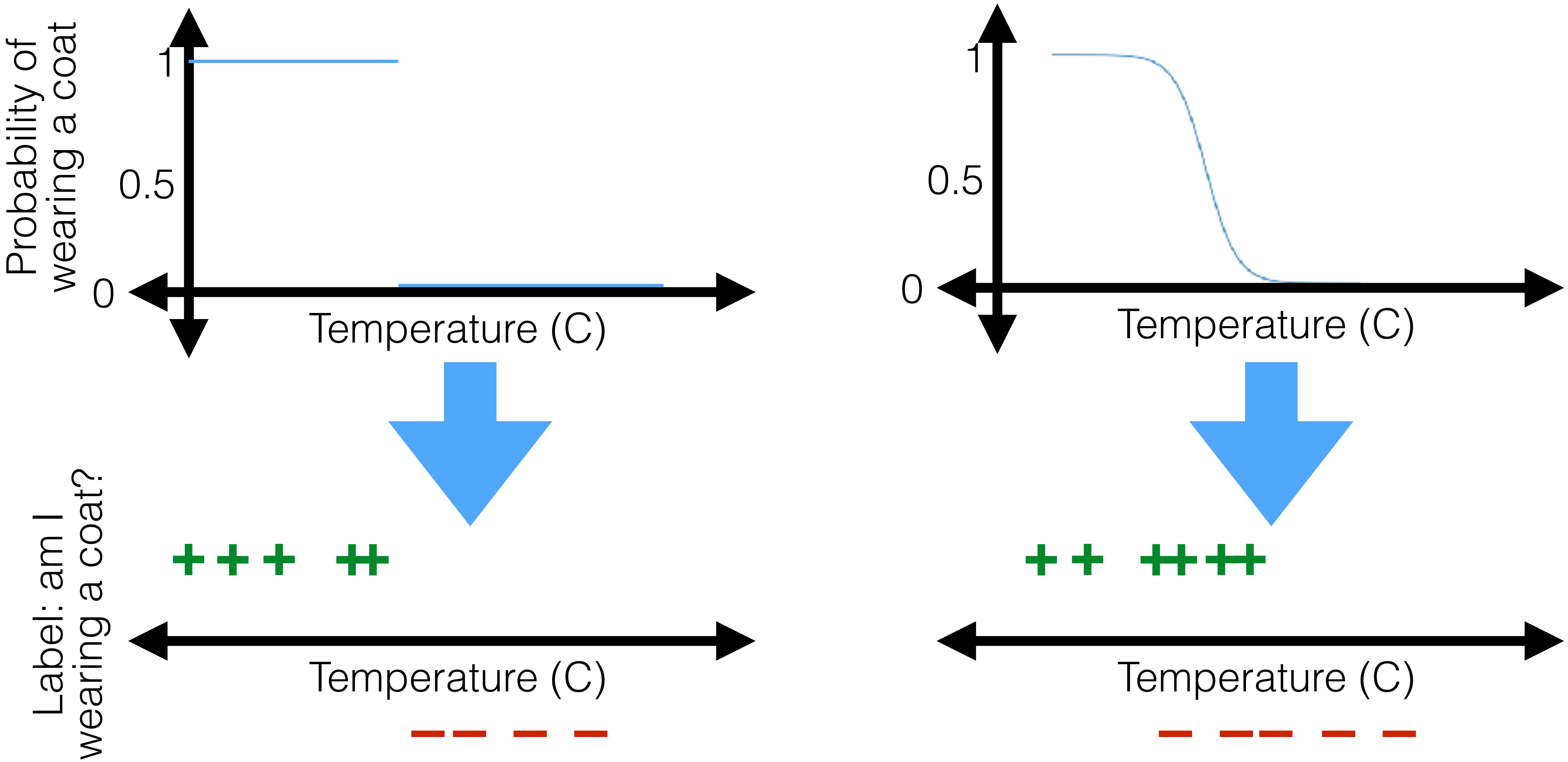




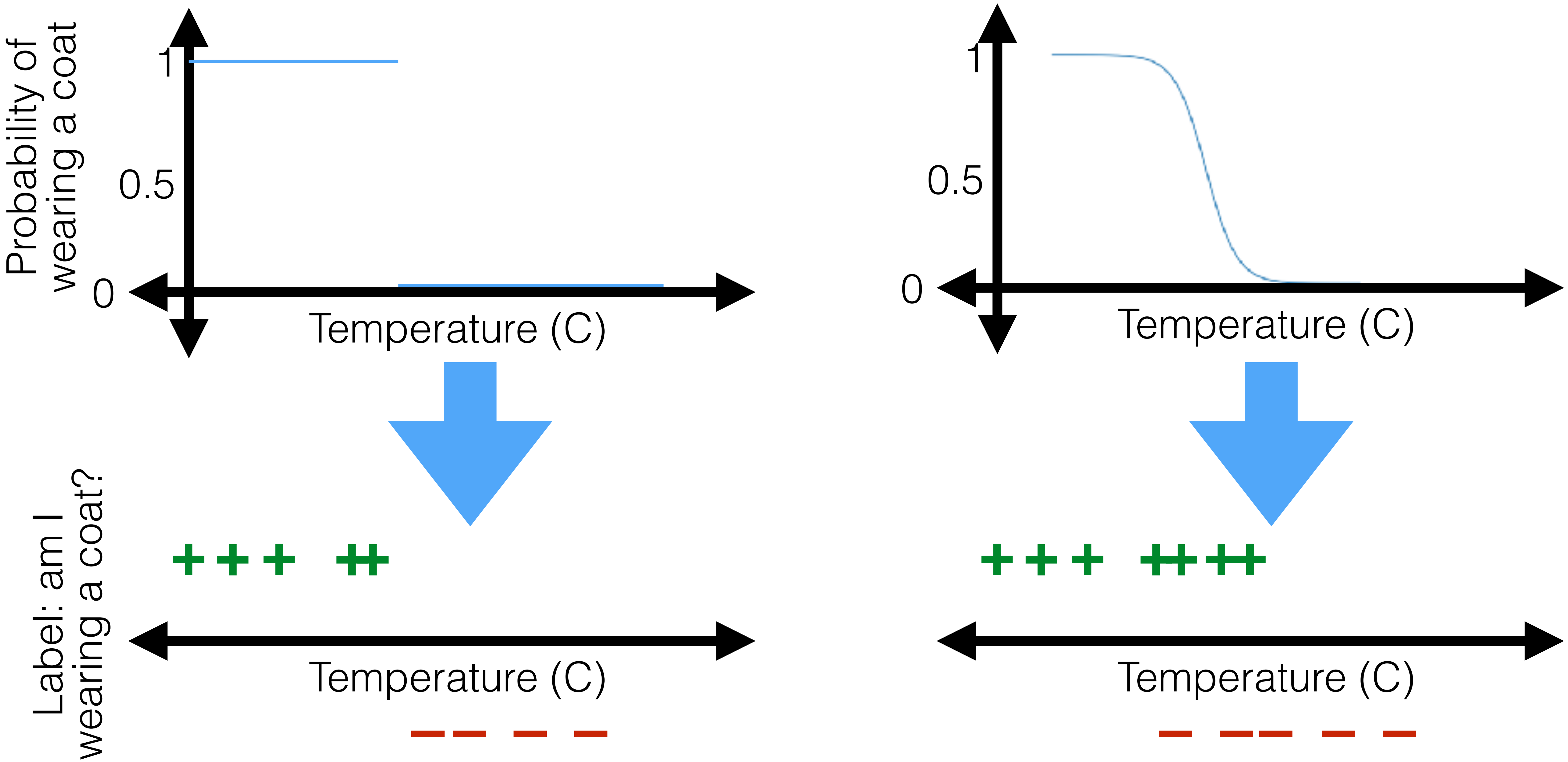
Capturing uncertainty



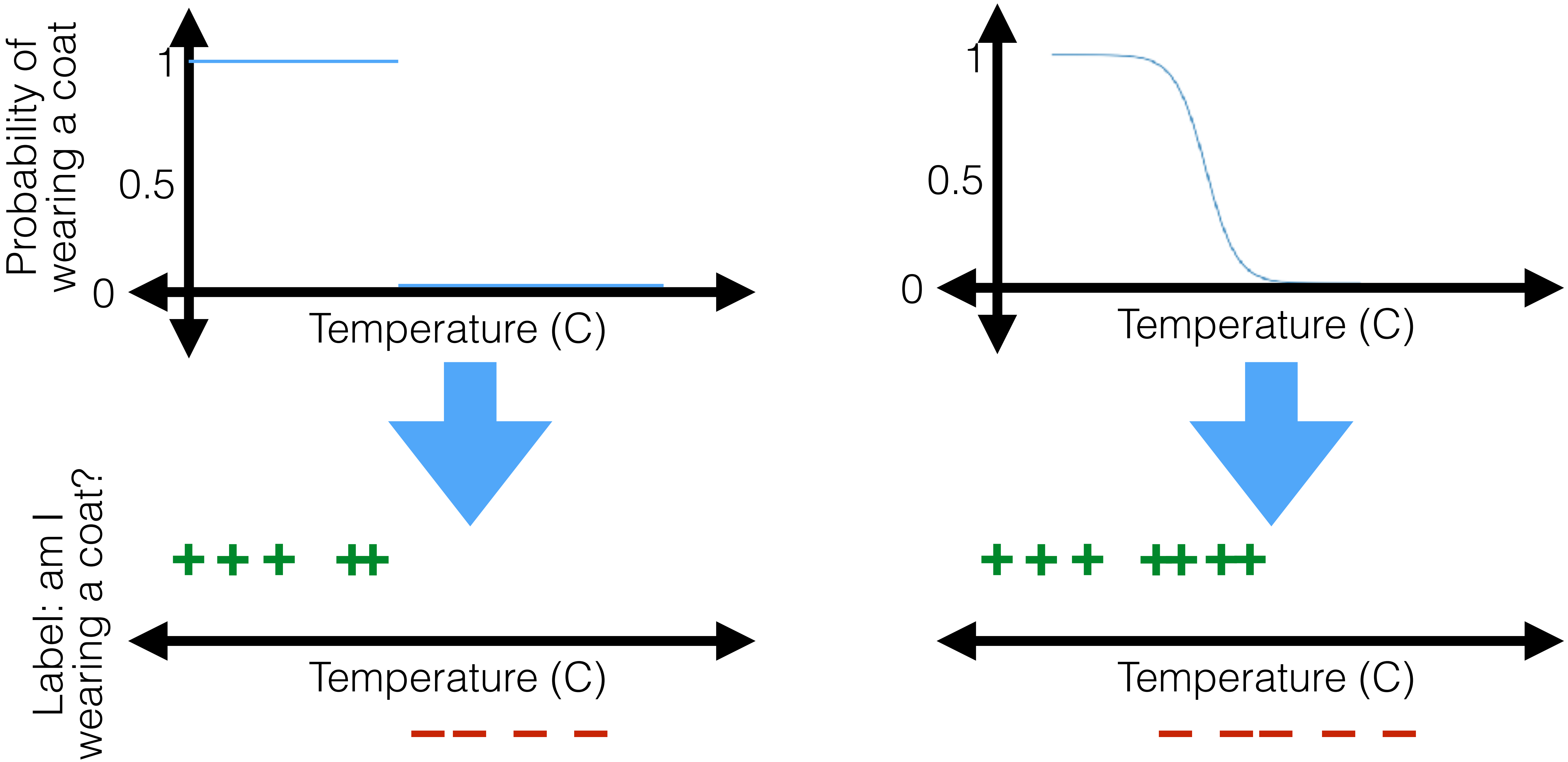
Capturing uncertainty



Capturing uncertainty

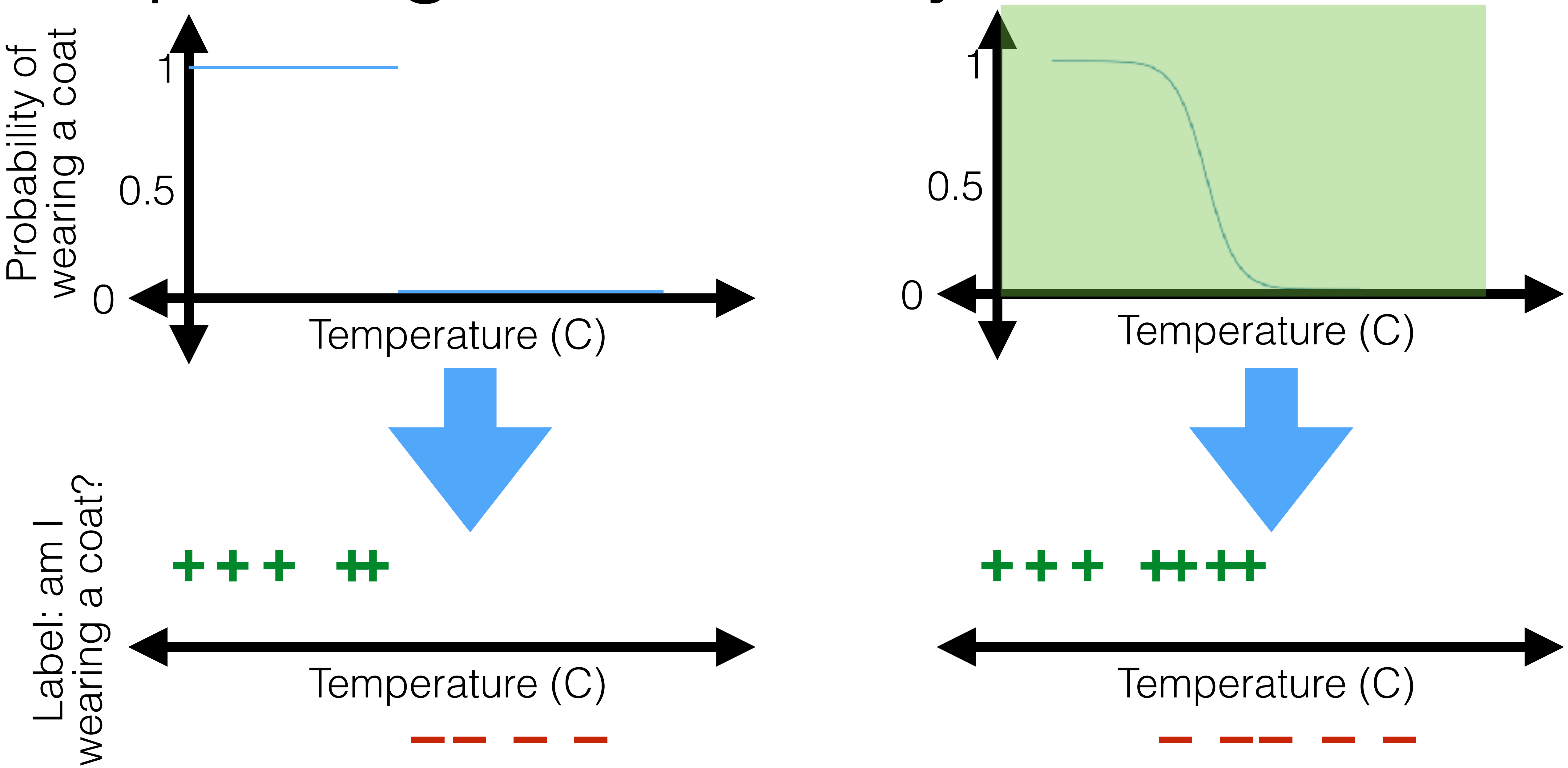


Capturing uncertainty



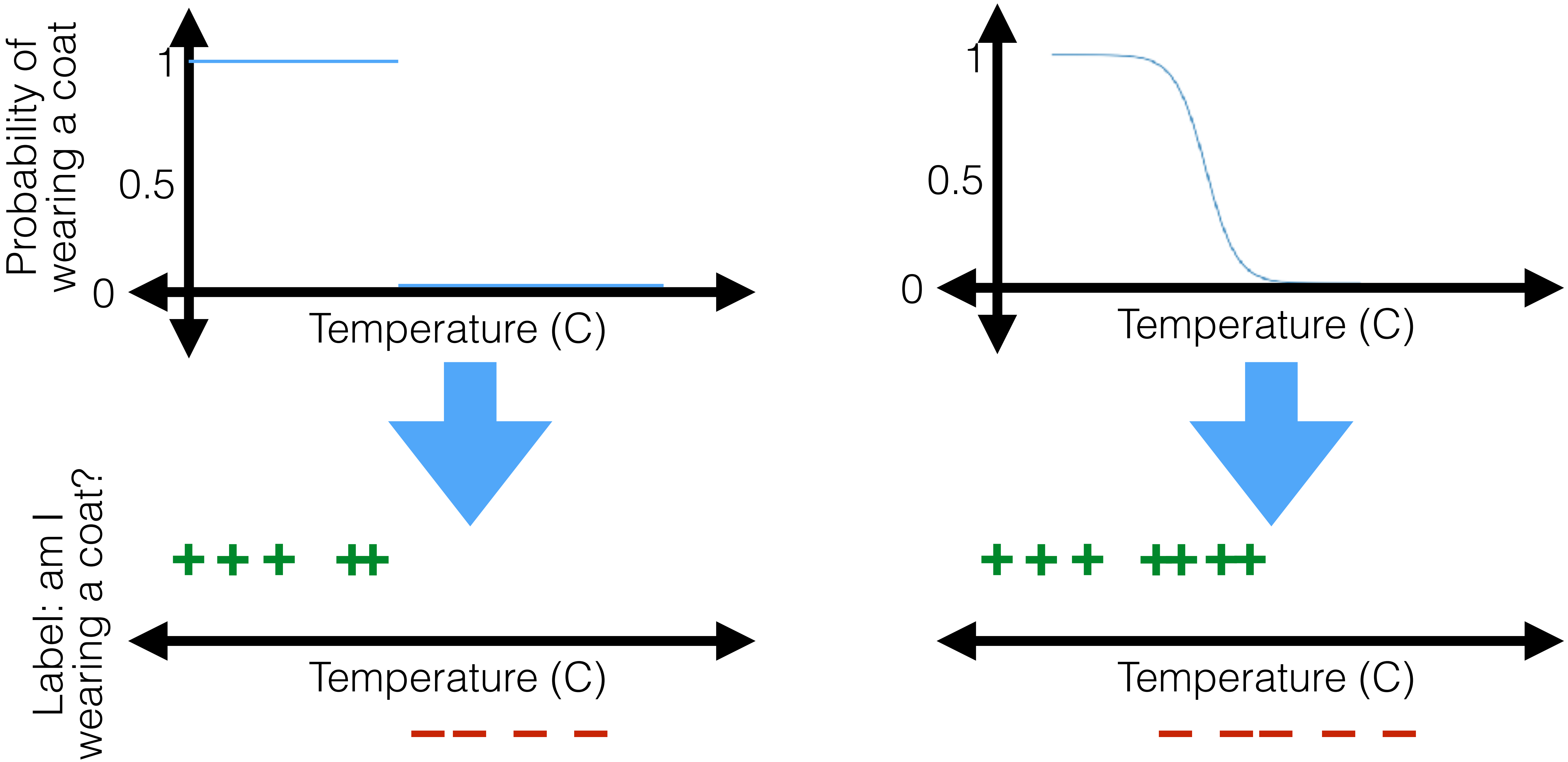
- How to make this shape?

Capturing uncertainty



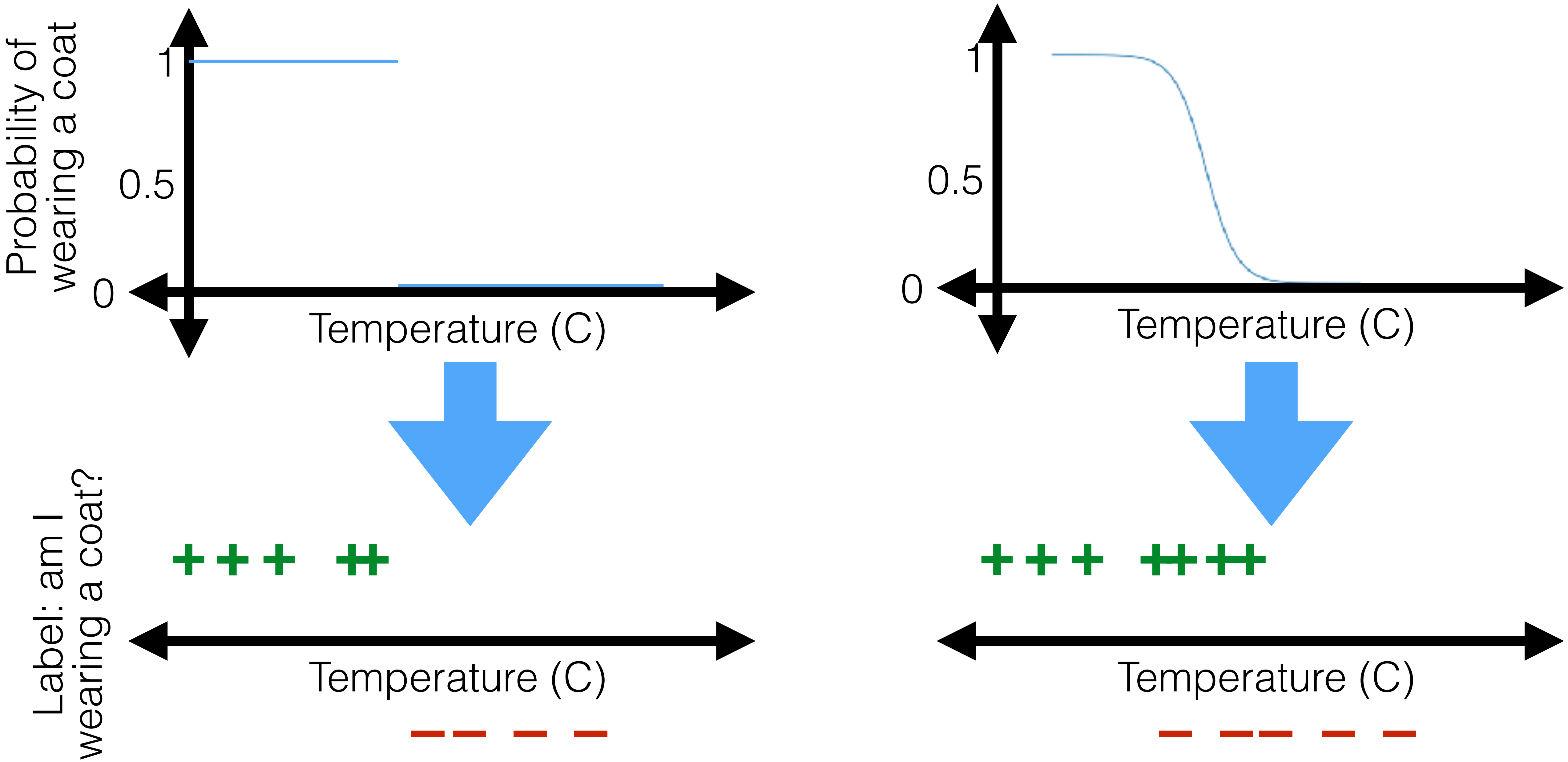
- How to make this shape?

Capturing uncertainty



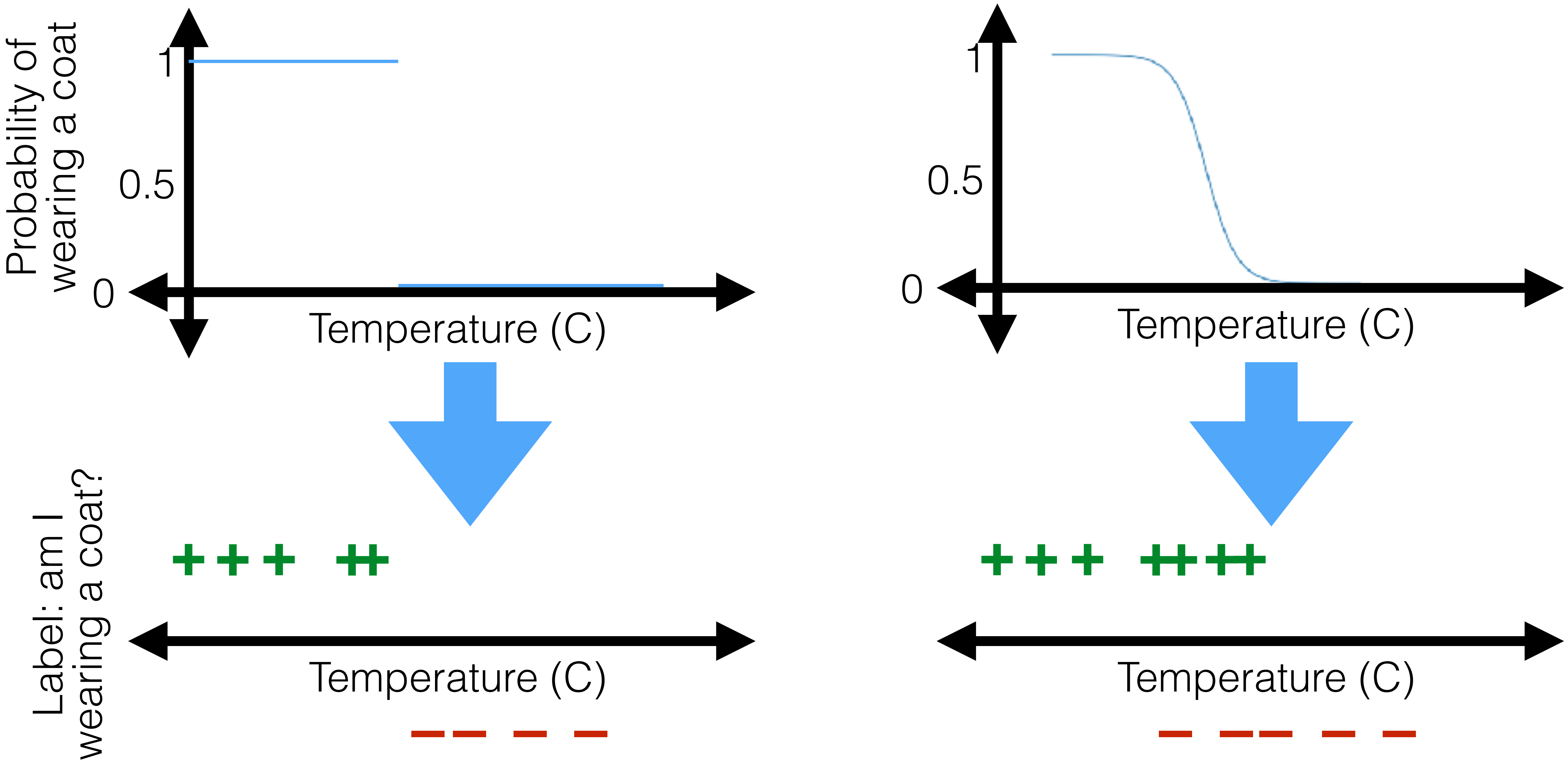
- How to make this shape?

Capturing uncertainty



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

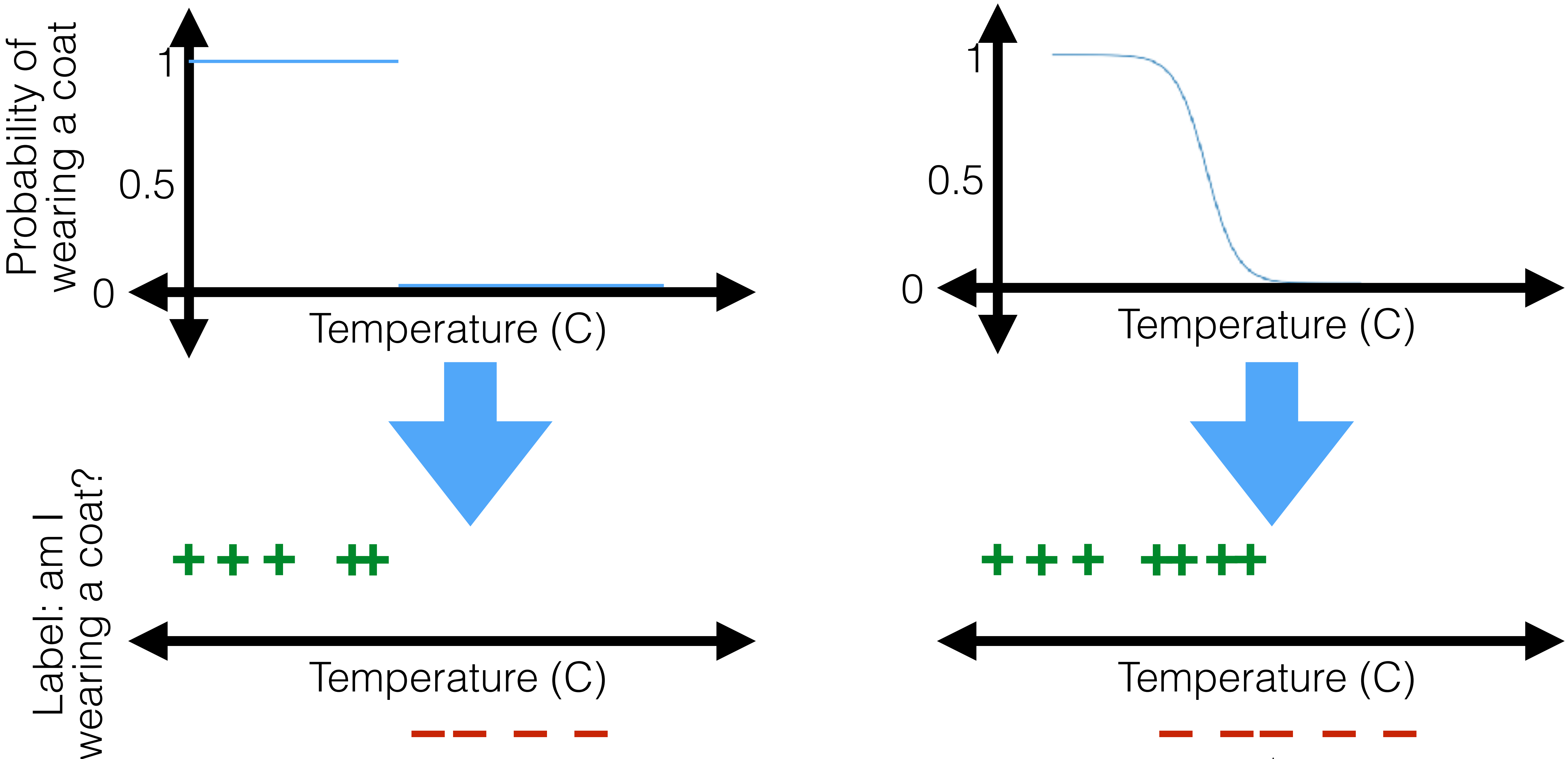
Capturing uncertainty



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

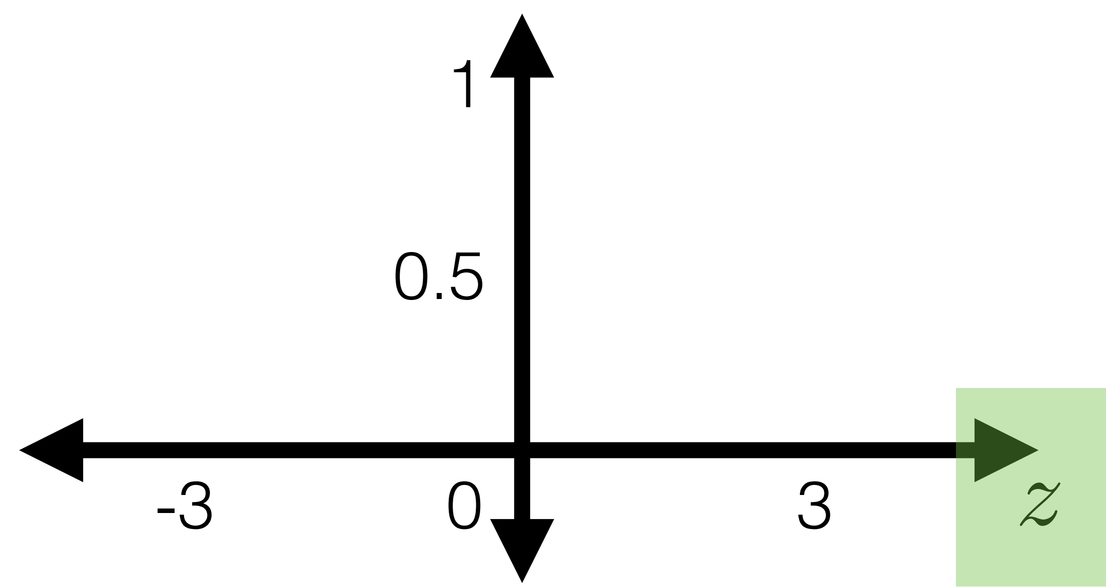
$$\sigma(z) = \frac{1}{1 + \exp(-z)}$$

Capturing uncertainty

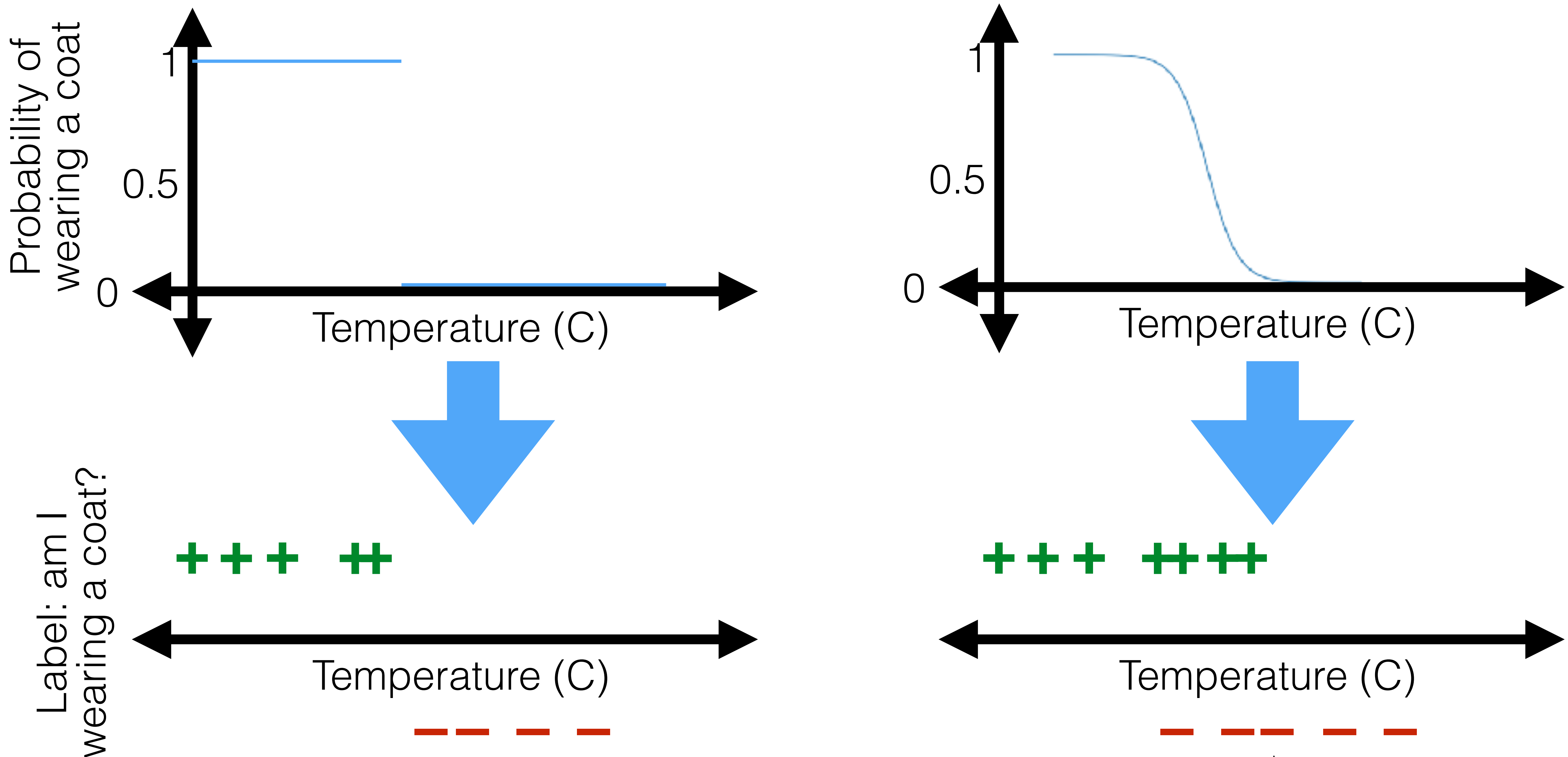


- How to make this shape?
  - Sigmoid/logistic function

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

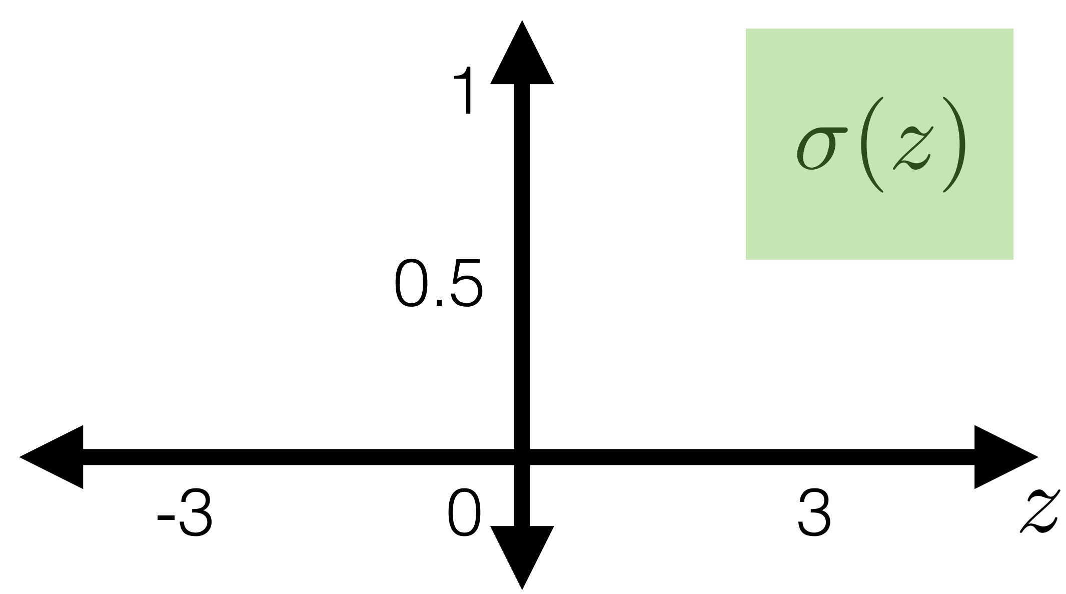


Capturing uncertainty

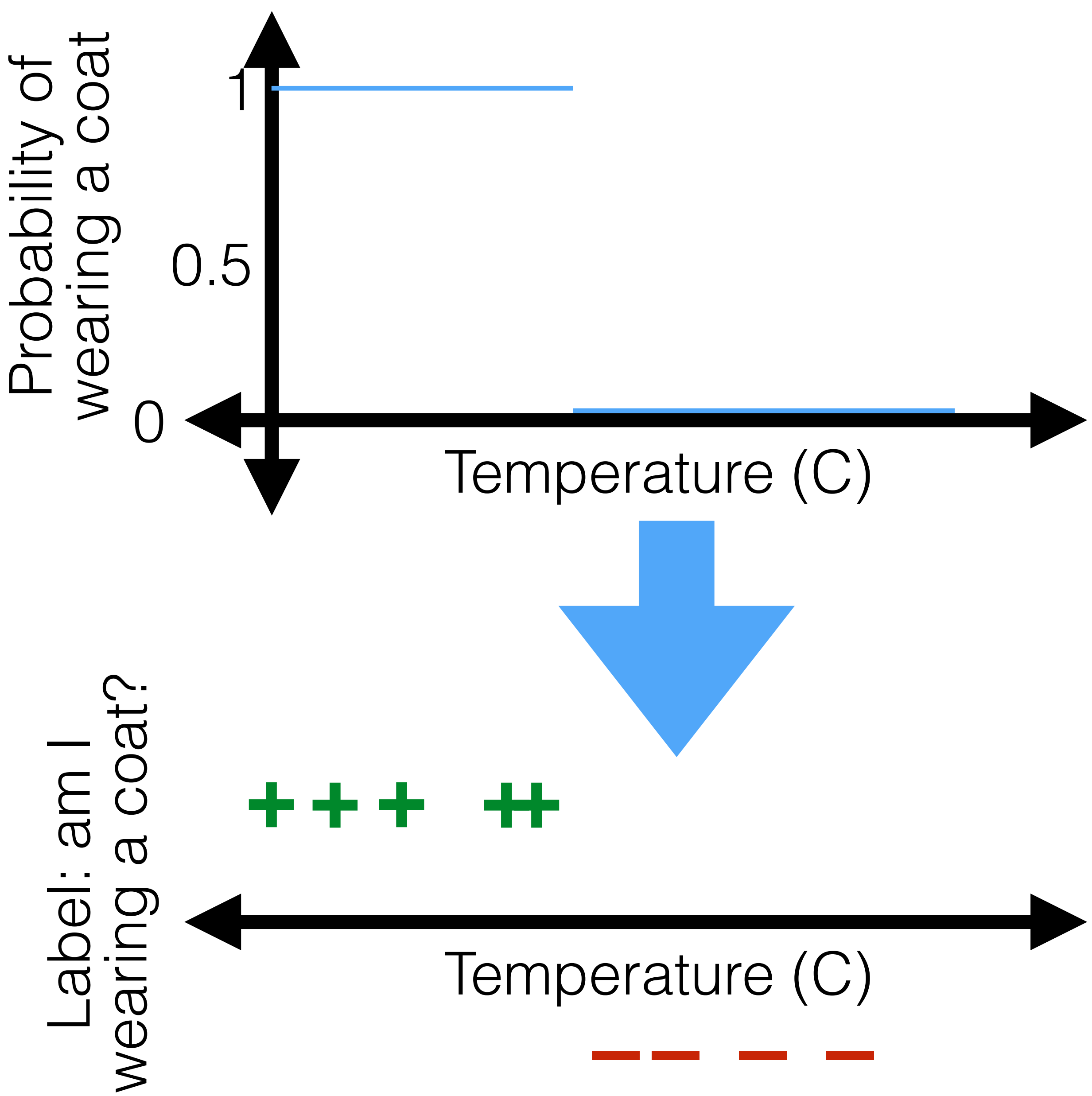


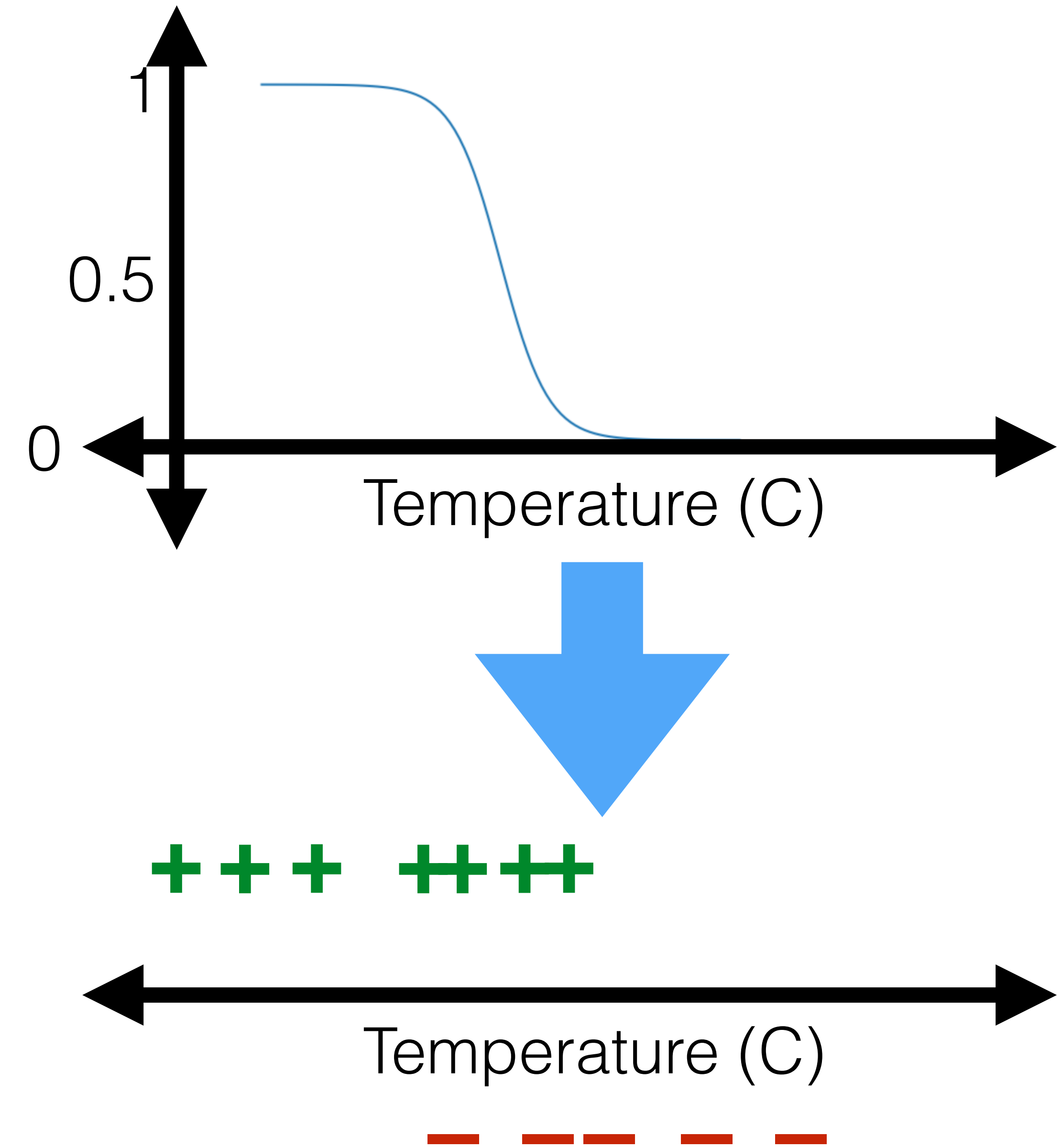
- How to make this shape?
  - Sigmoid/logistic function

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



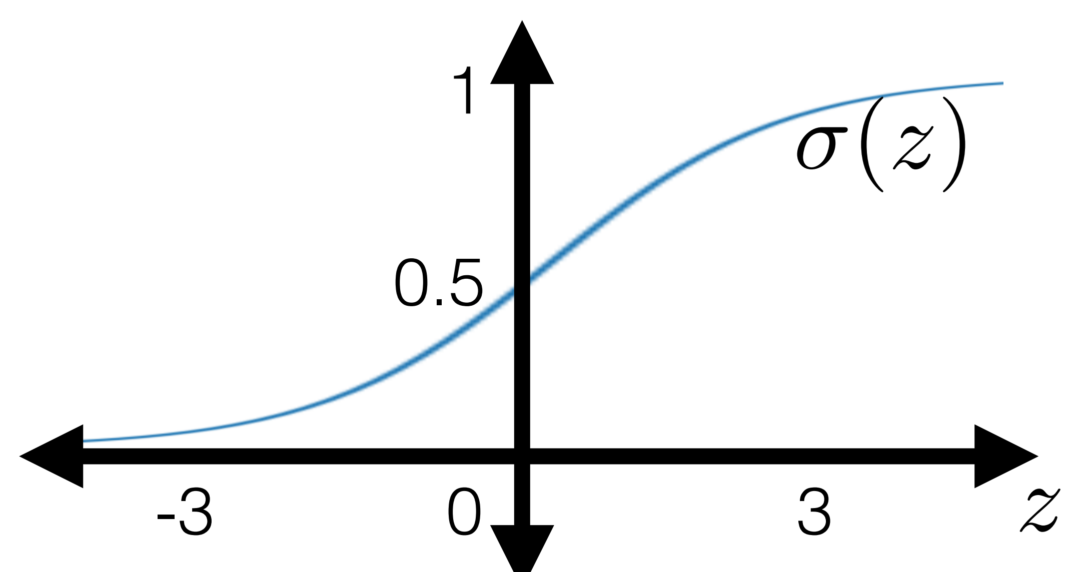
Capturing uncertainty



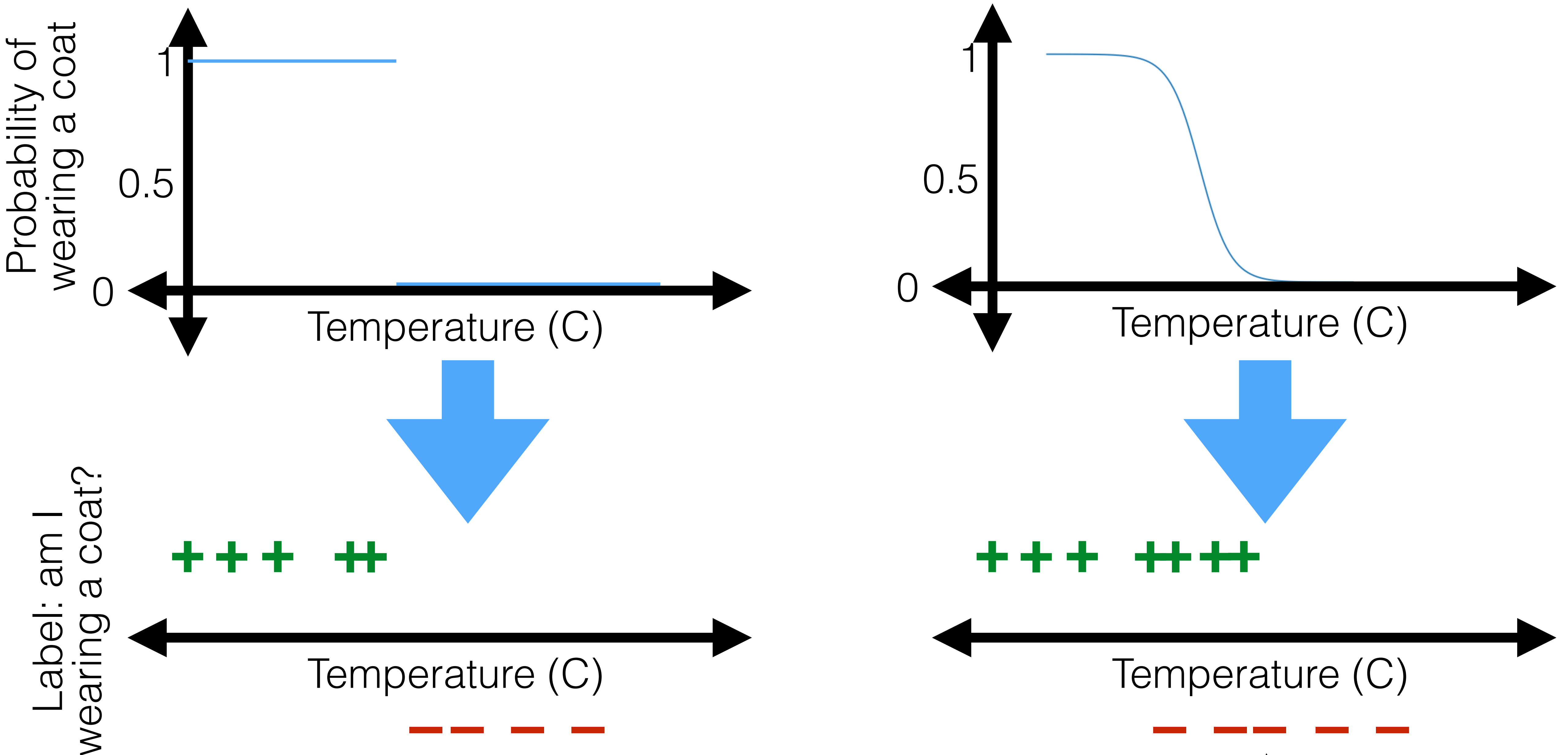


- How to make this shape?
  - Sigmoid/logistic function

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

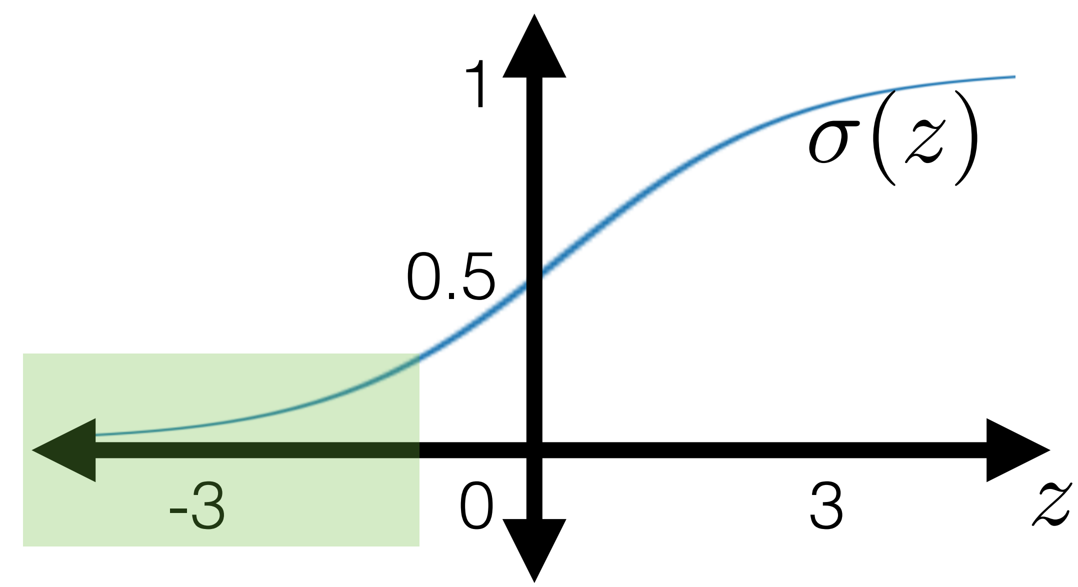


Capturing uncertainty

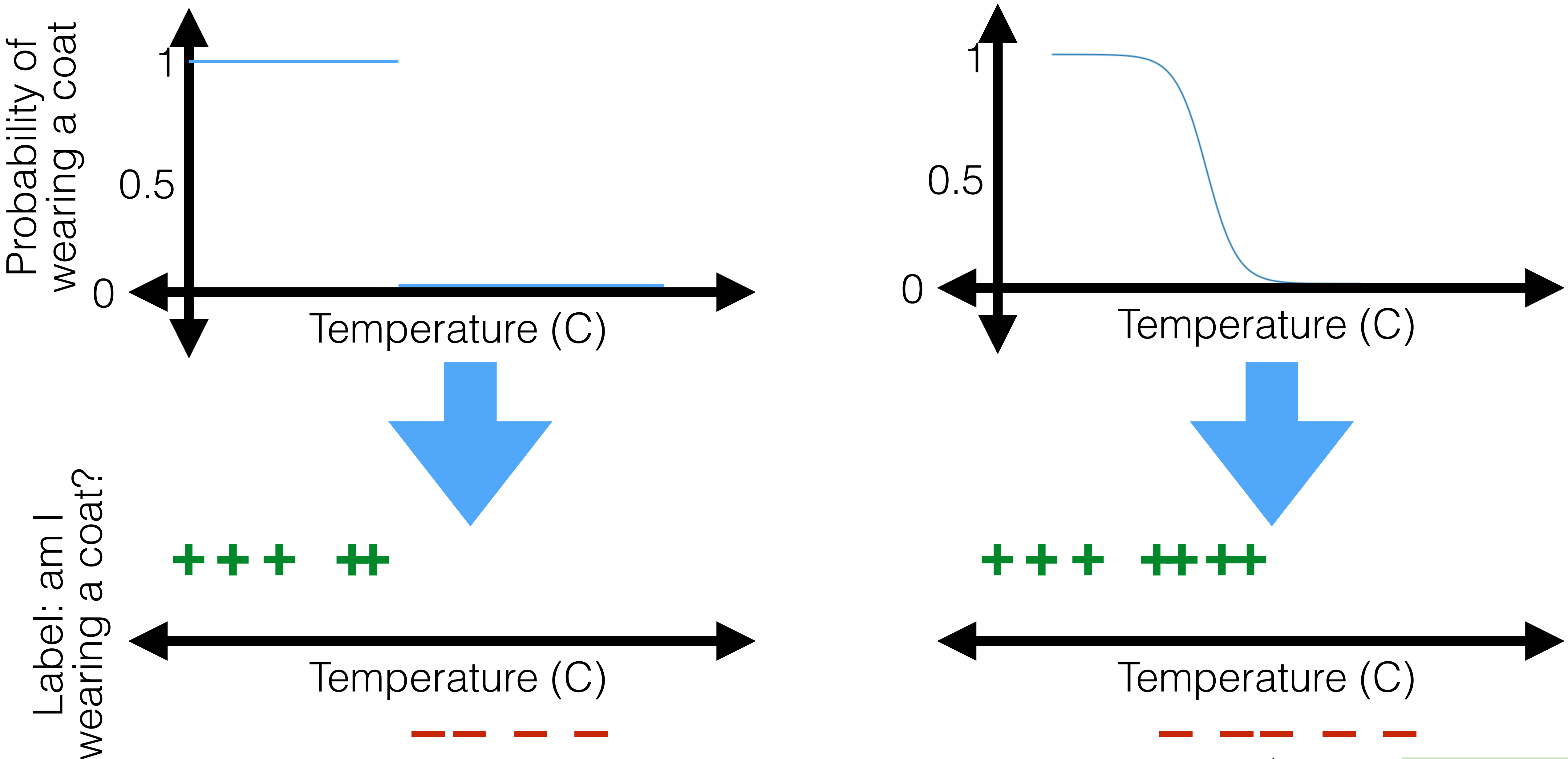


- How to make this shape?
  - Sigmoid/logistic function

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

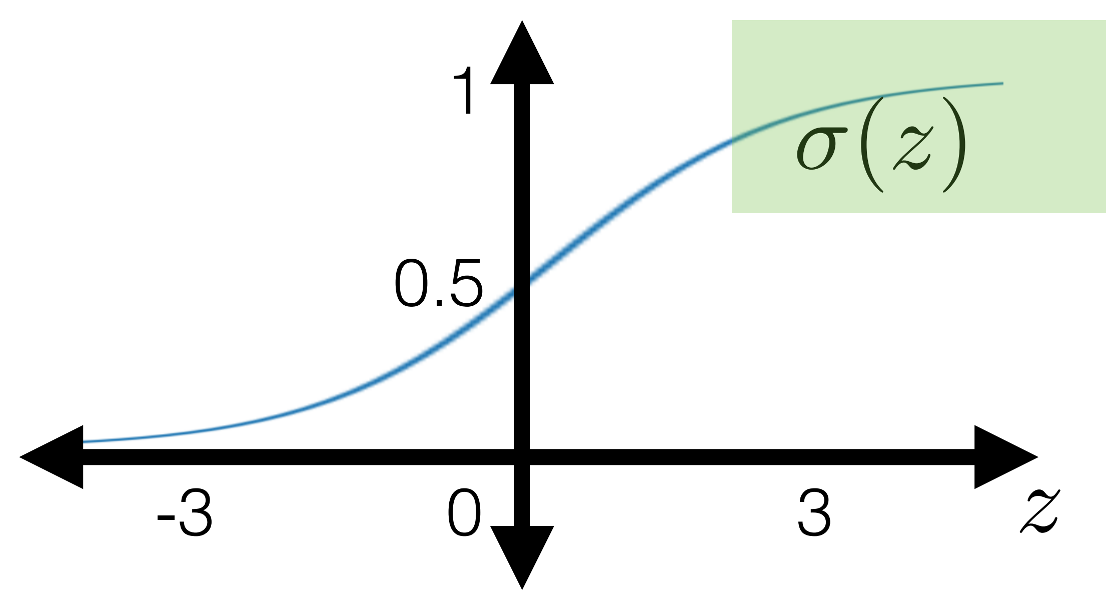


Capturing uncertainty

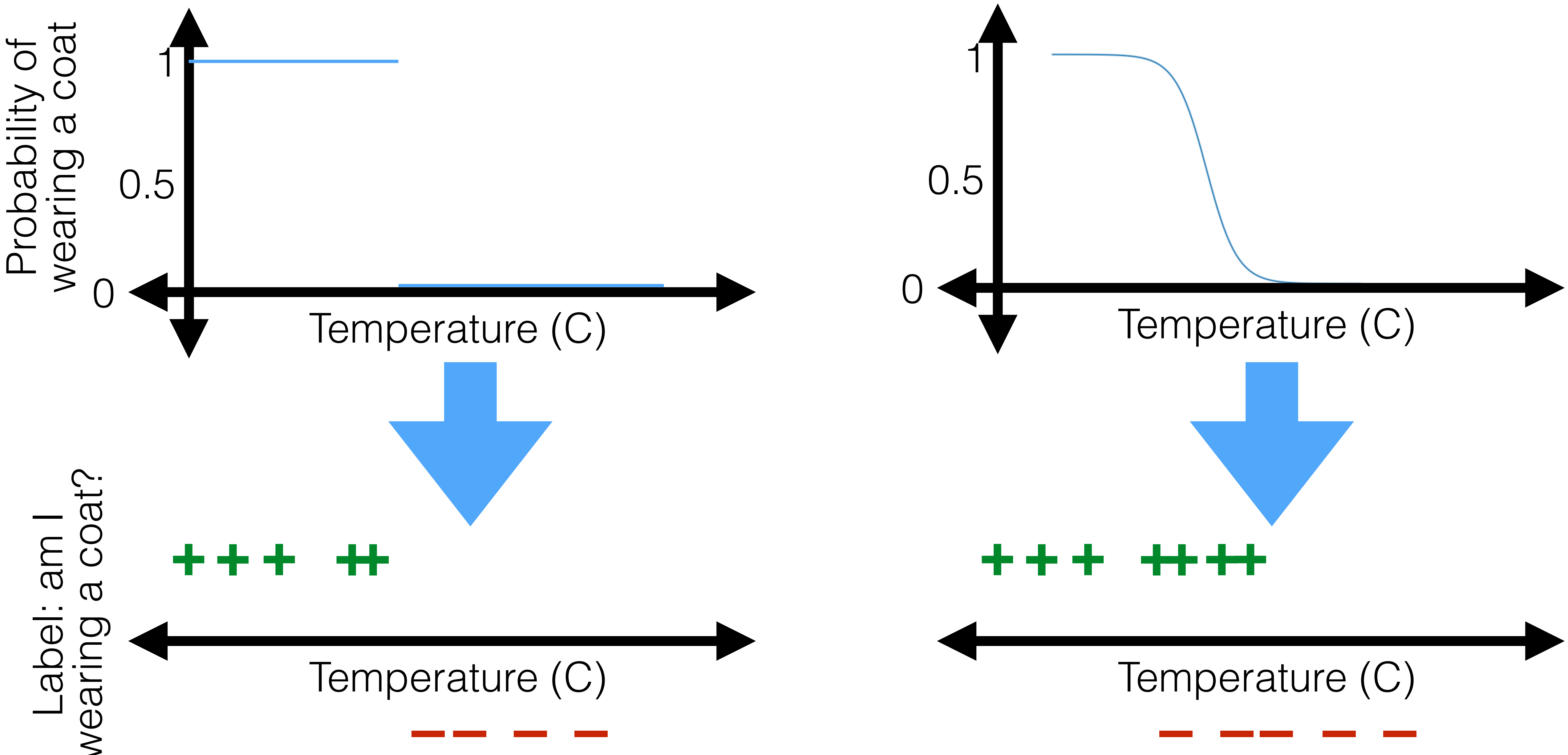


- How to make this shape?
  - Sigmoid/logistic function

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

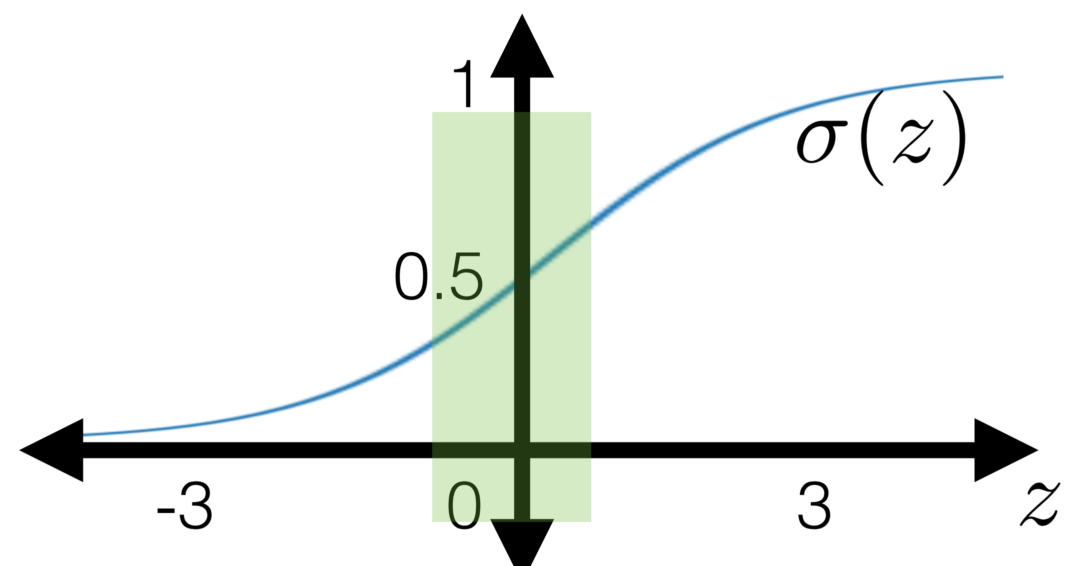


Capturing uncertainty

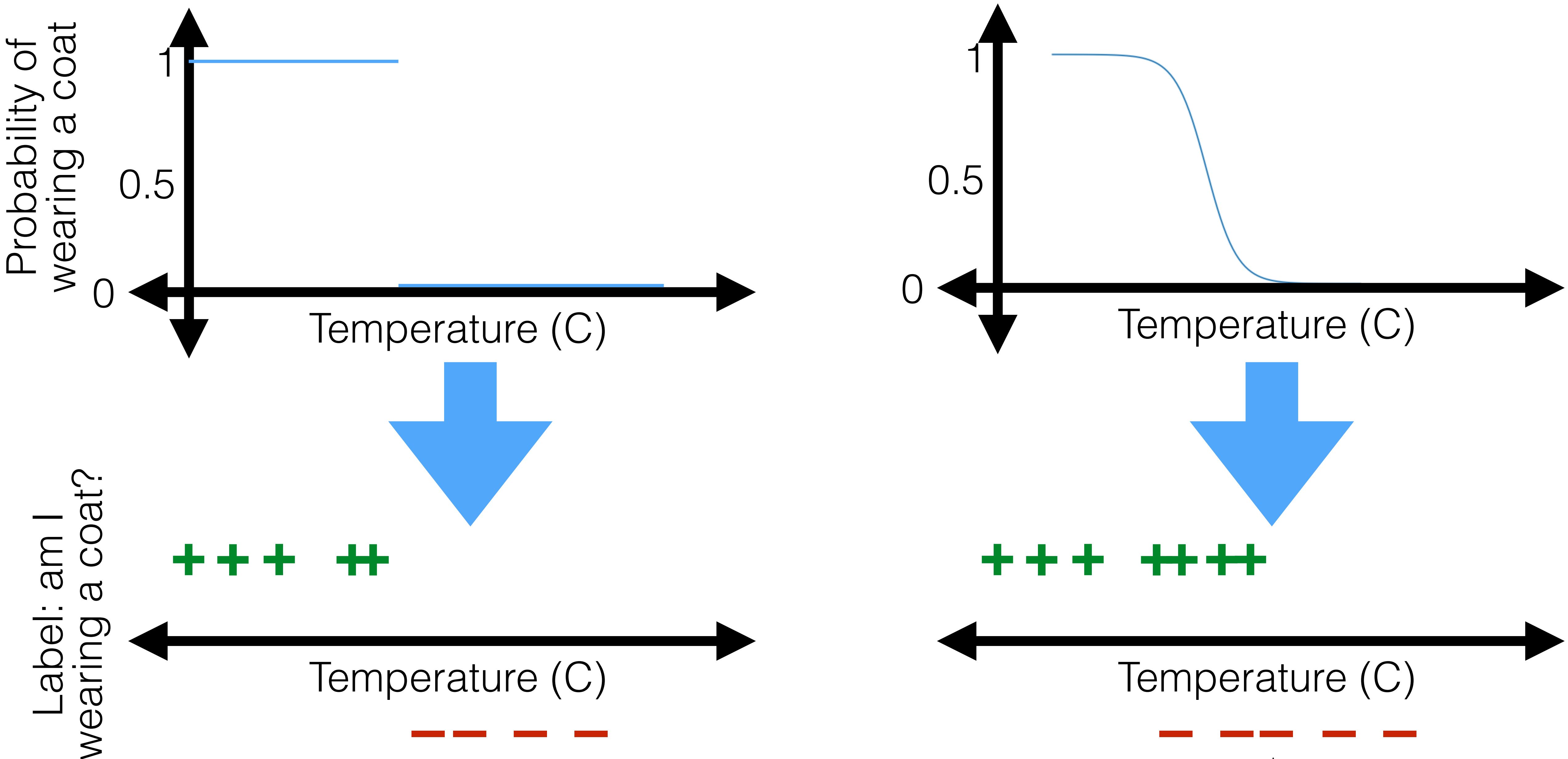


- How to make this shape?
  - Sigmoid/logistic function

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

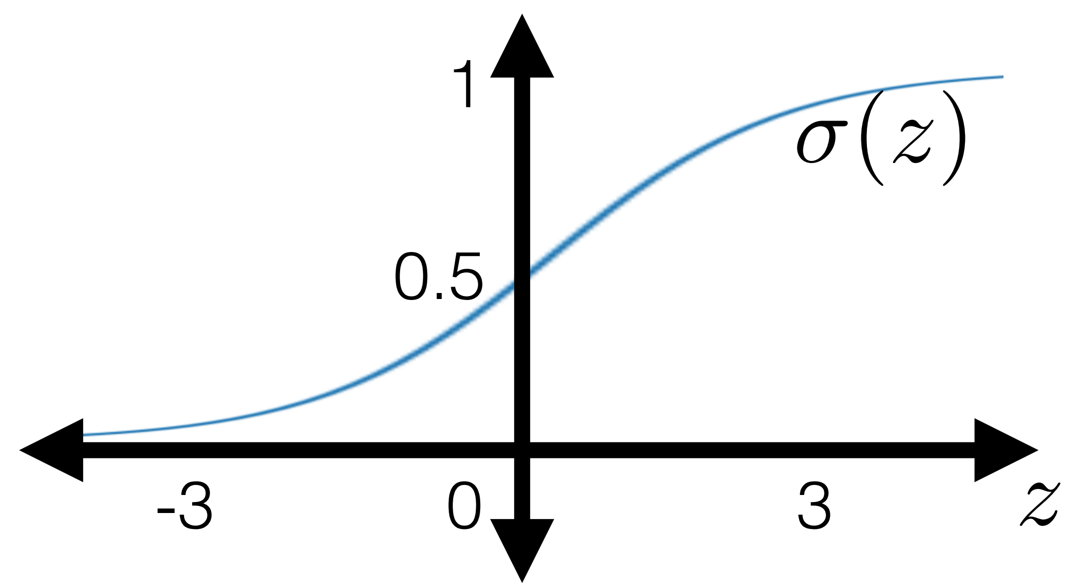


Capturing uncertainty

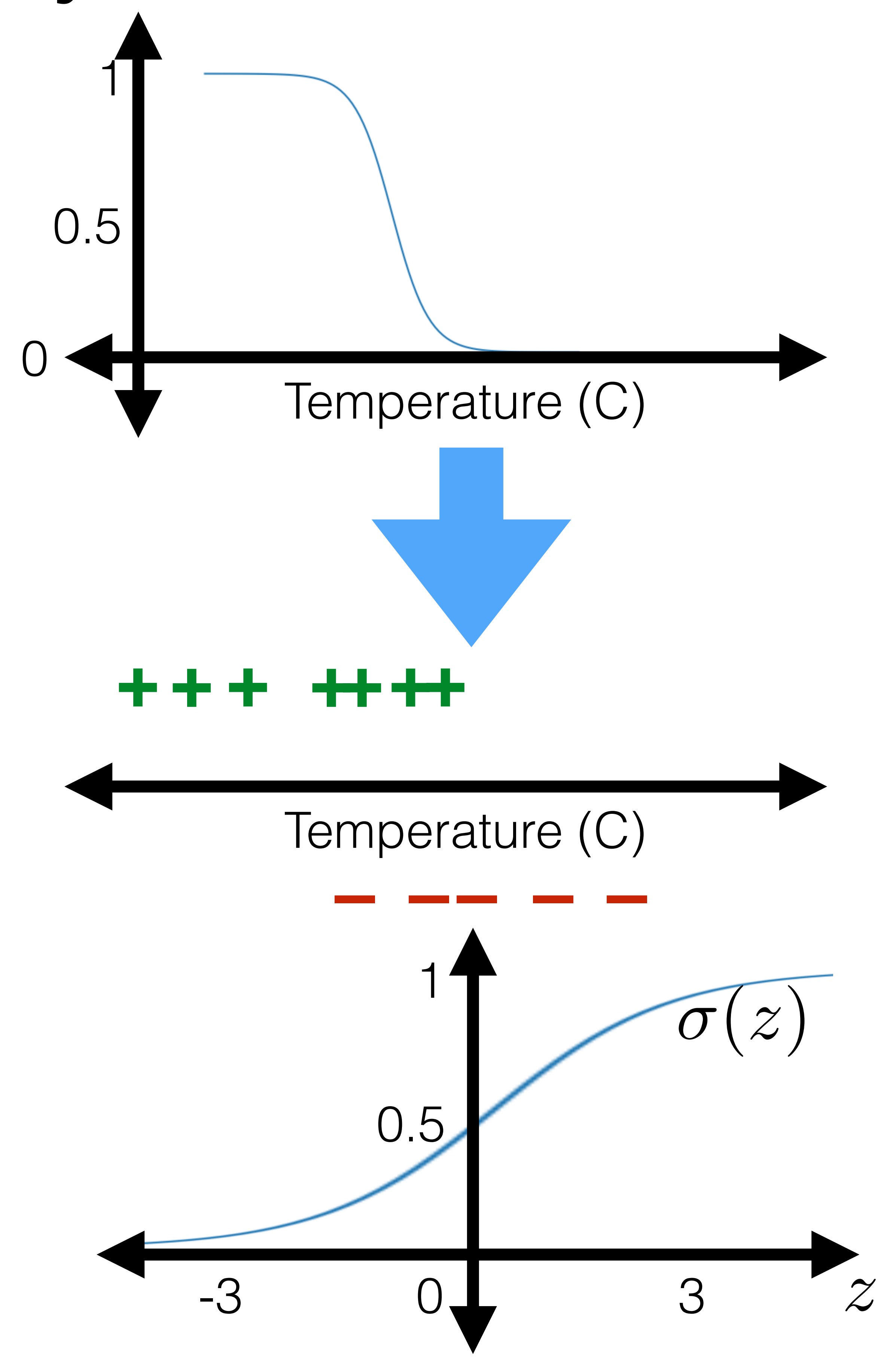


- How to make this shape?
  - Sigmoid/logistic function

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



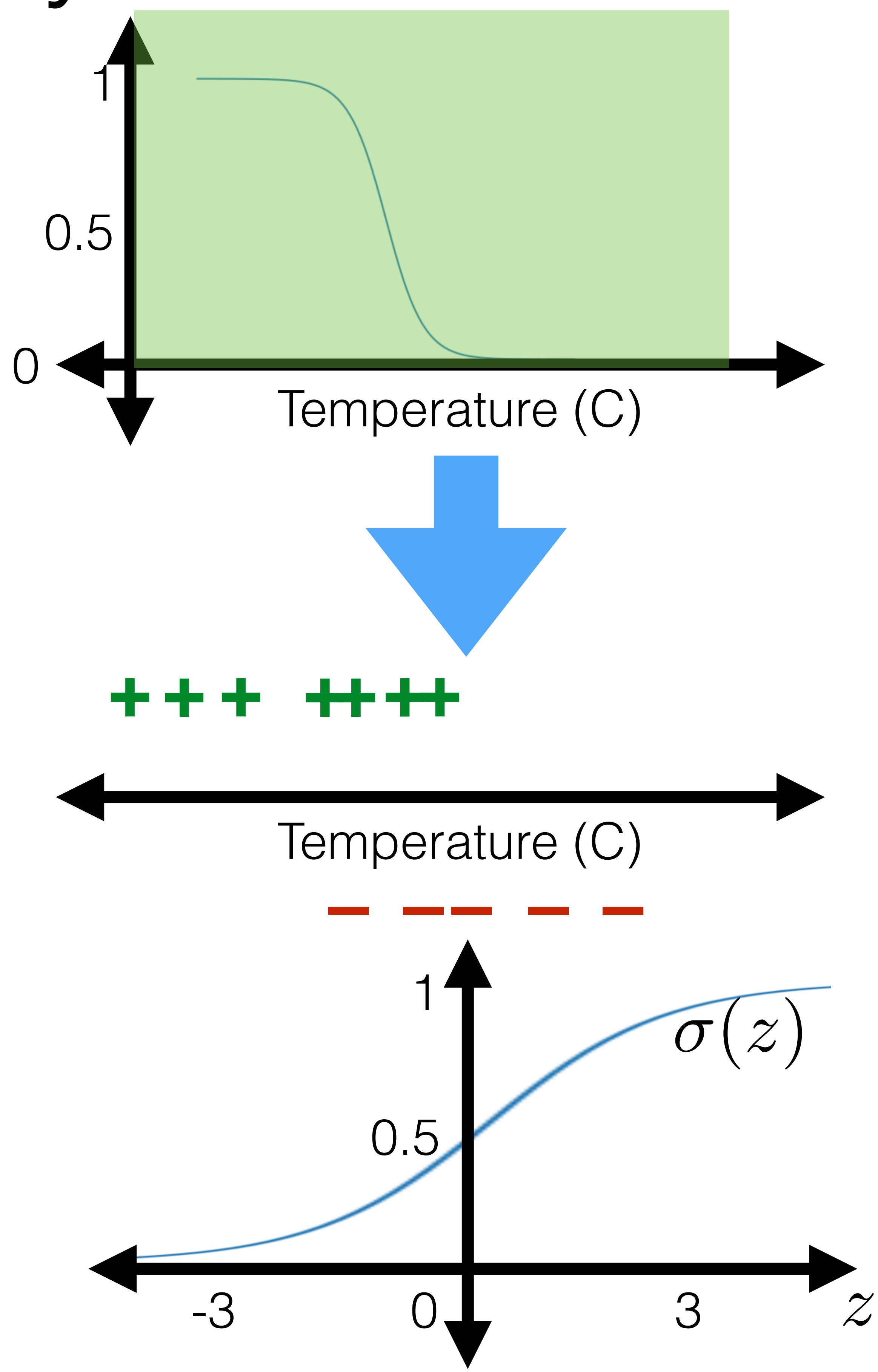
Capturing uncertainty



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

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

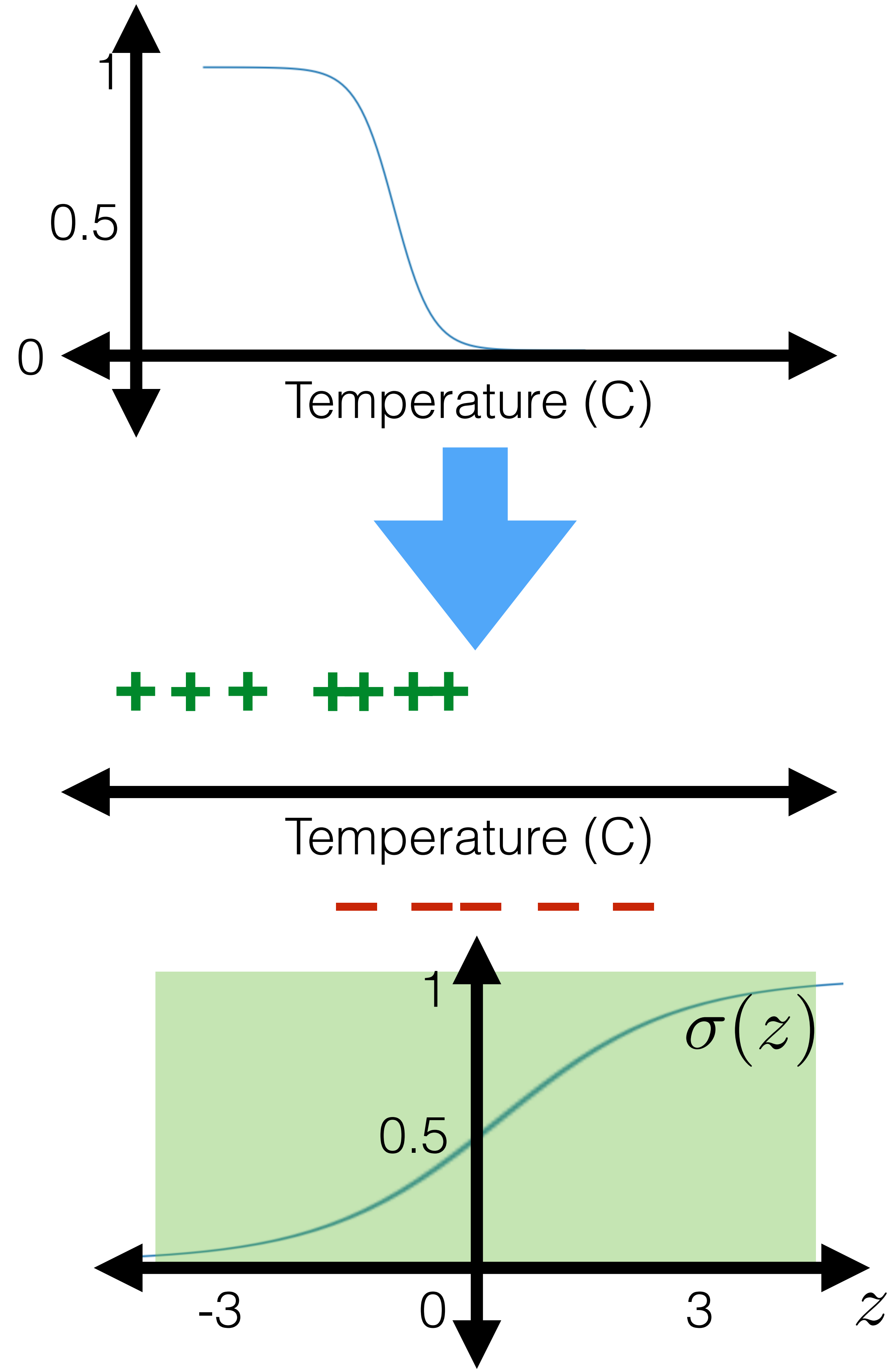
Capturing uncertainty



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

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

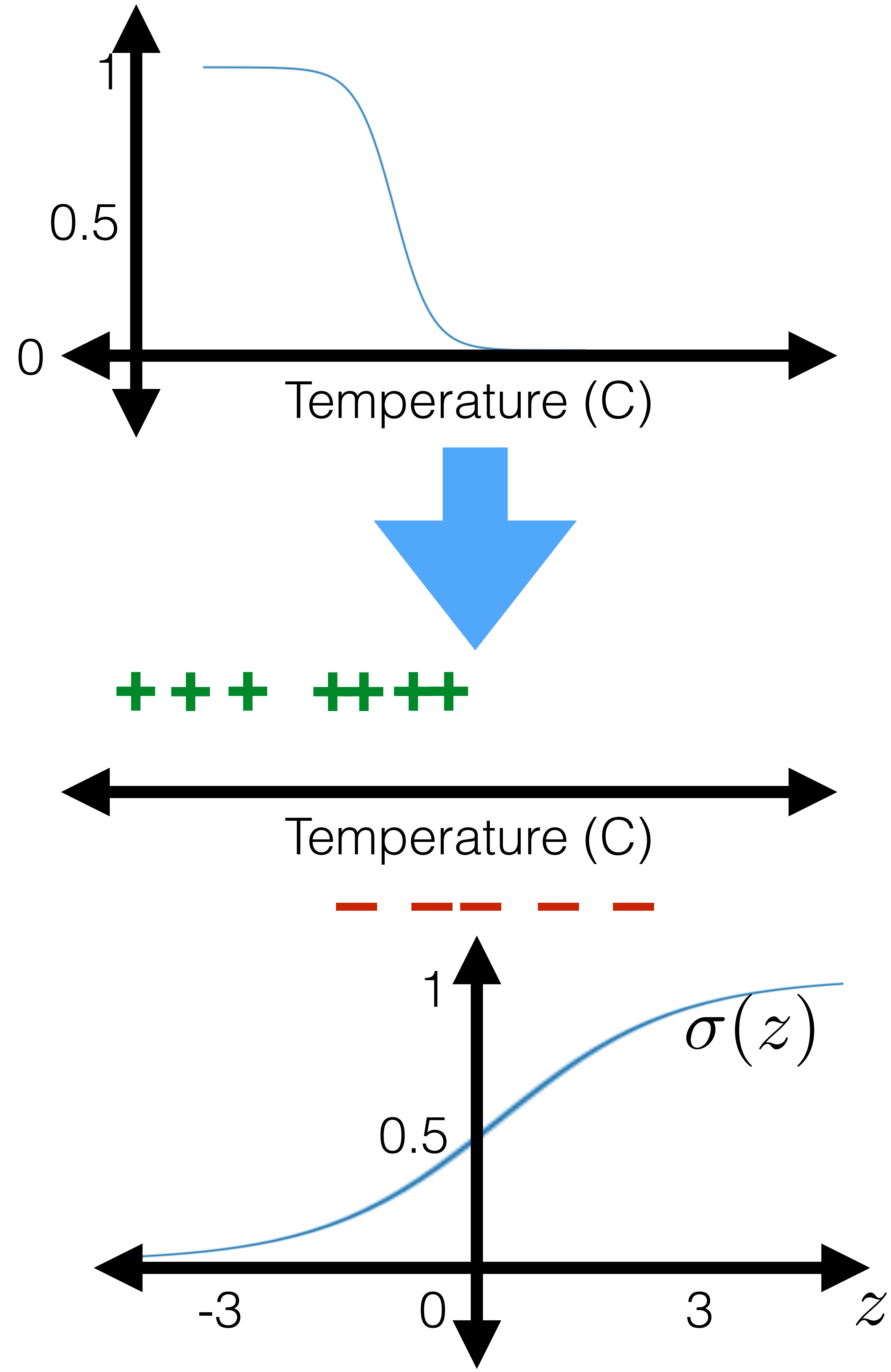
Capturing uncertainty



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

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

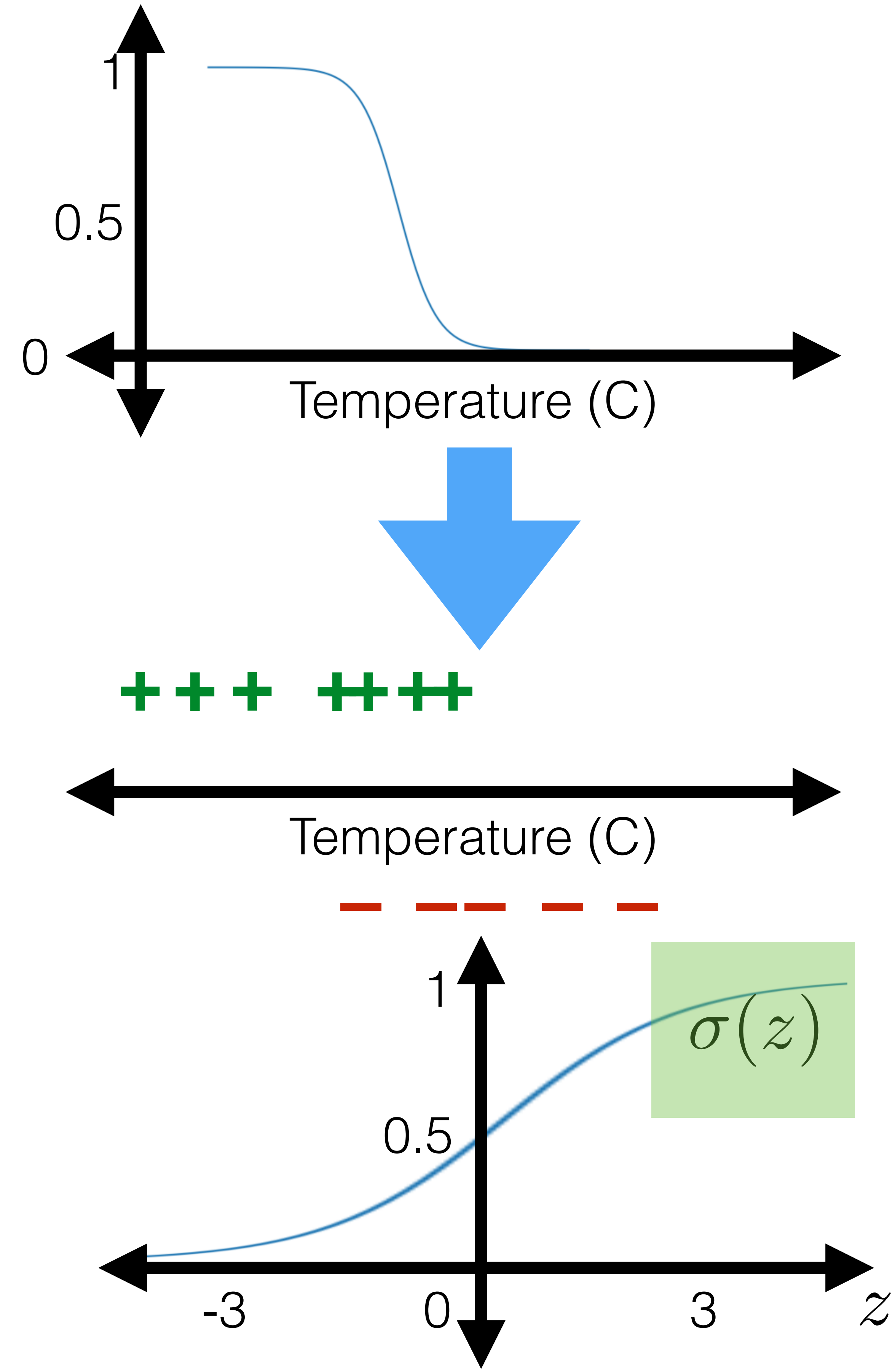
Capturing uncertainty



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

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

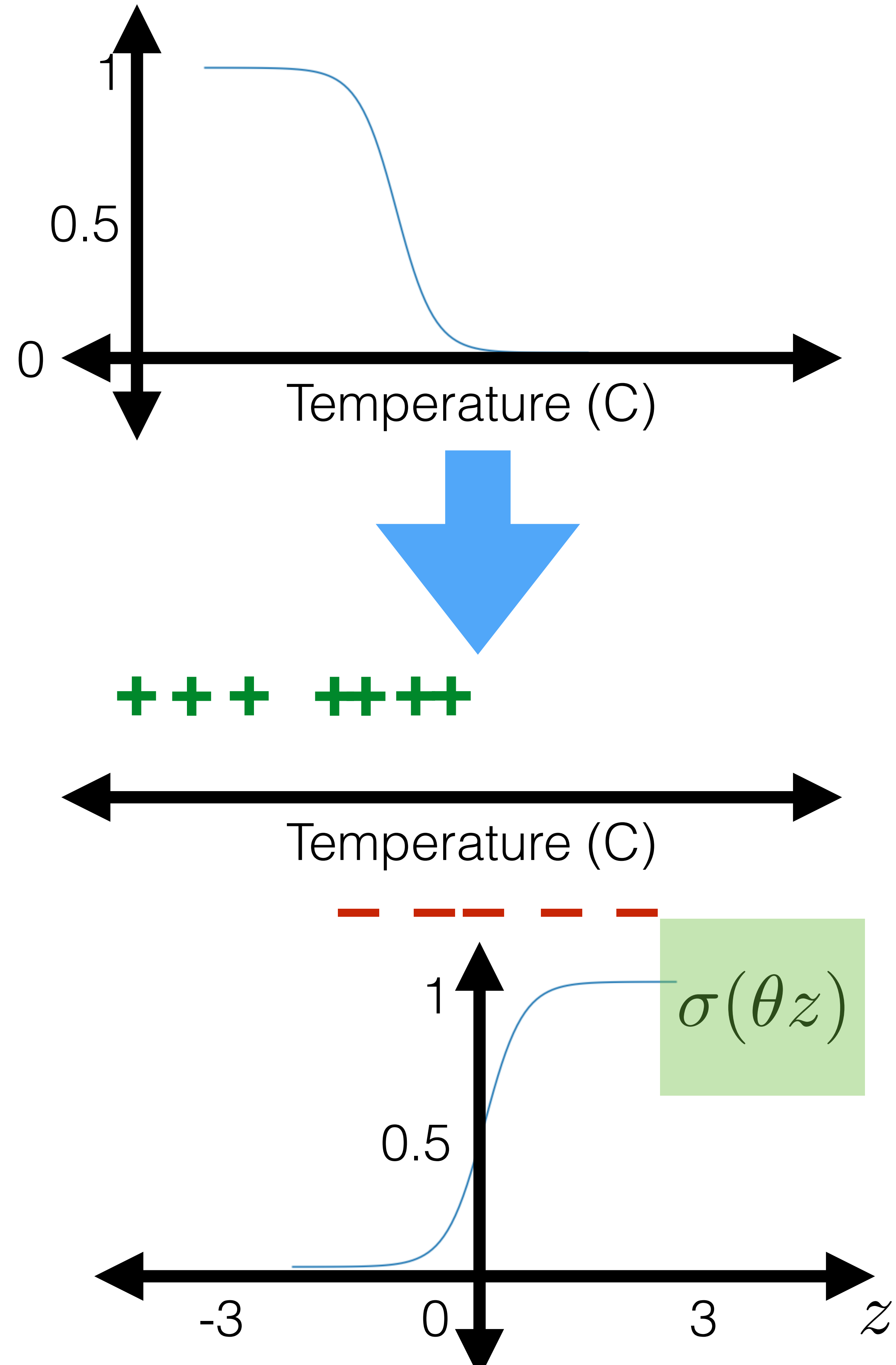
Capturing uncertainty



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

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

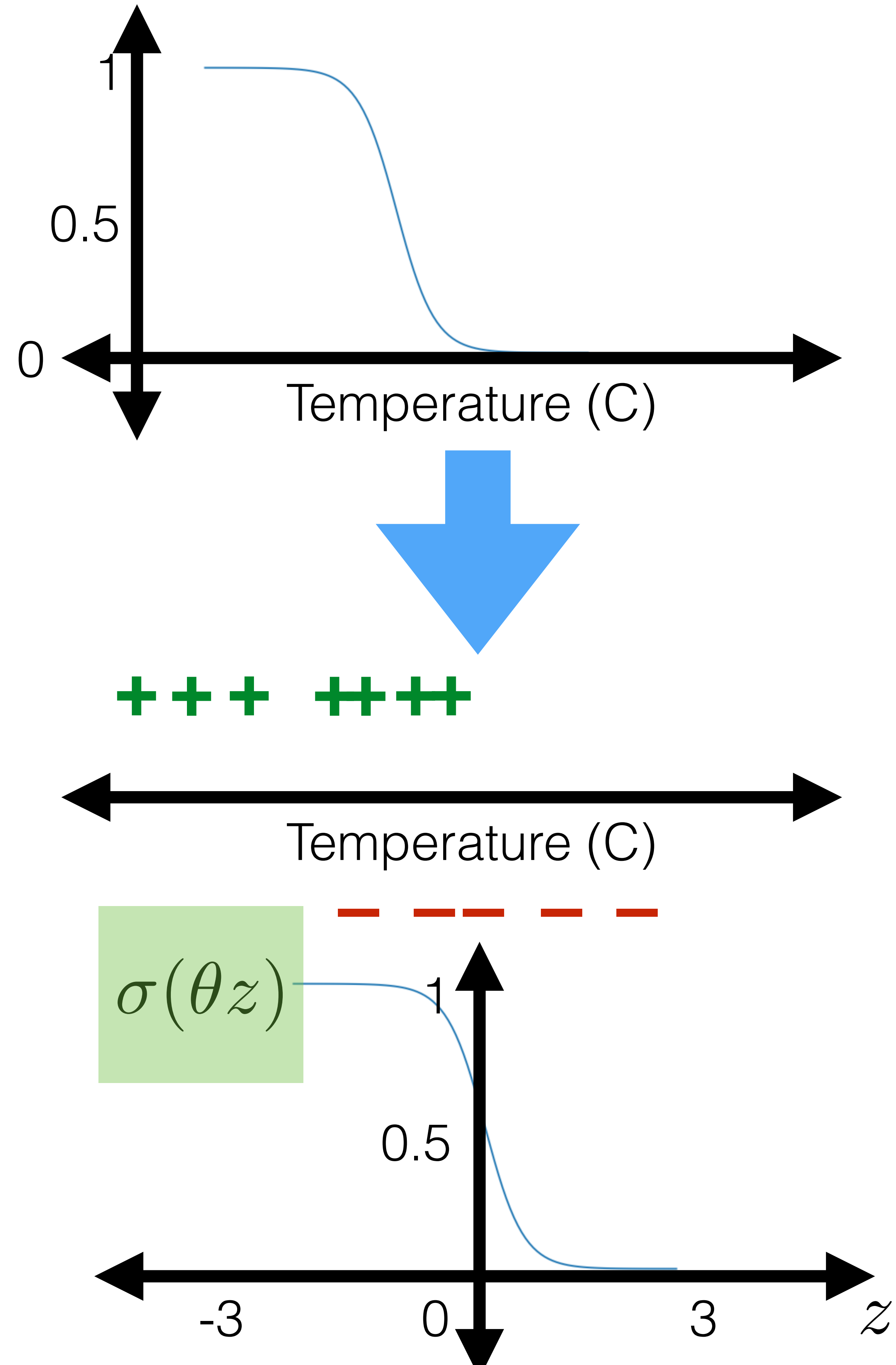
Capturing uncertainty



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

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

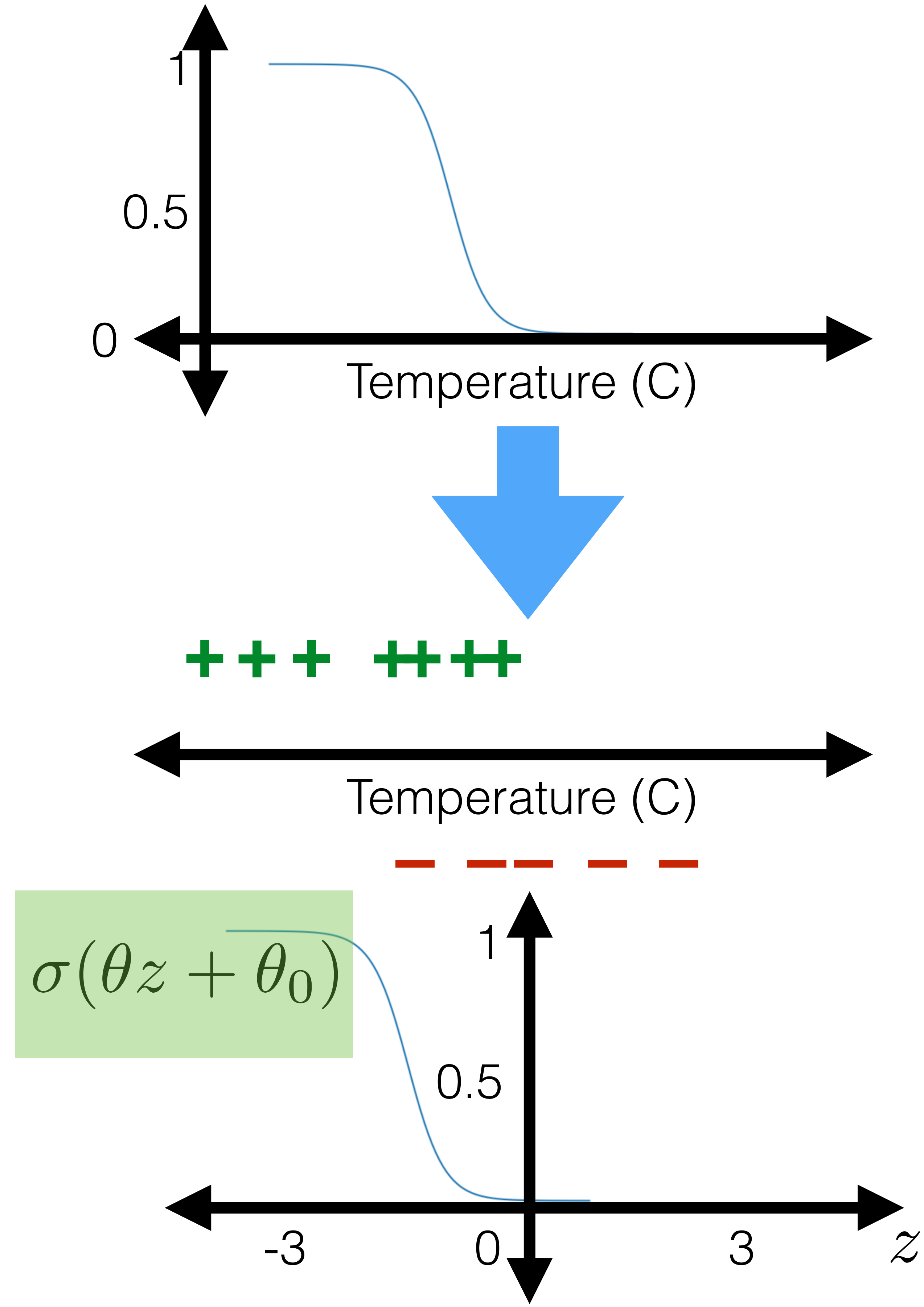
Capturing uncertainty



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

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

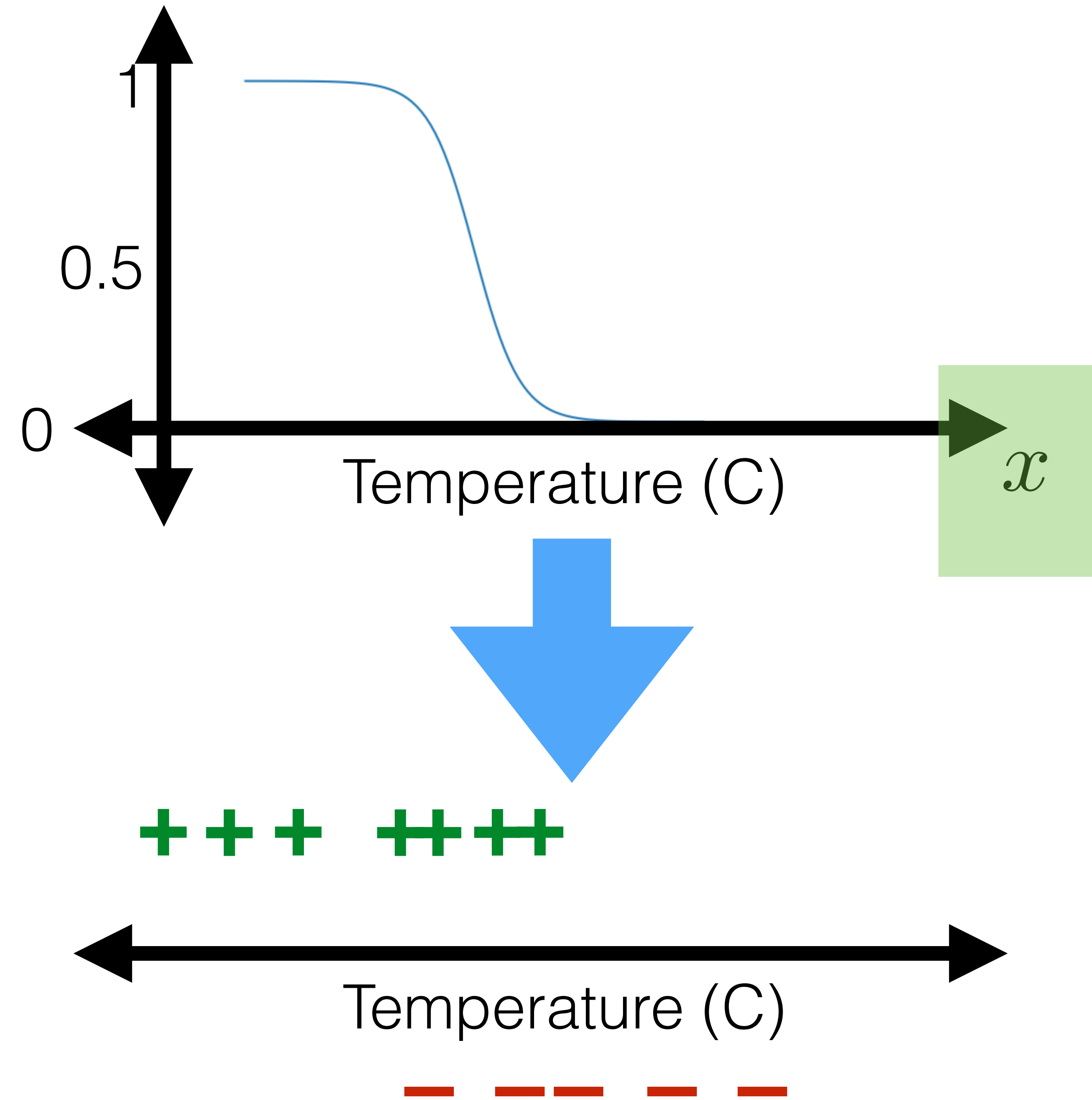
Capturing uncertainty



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

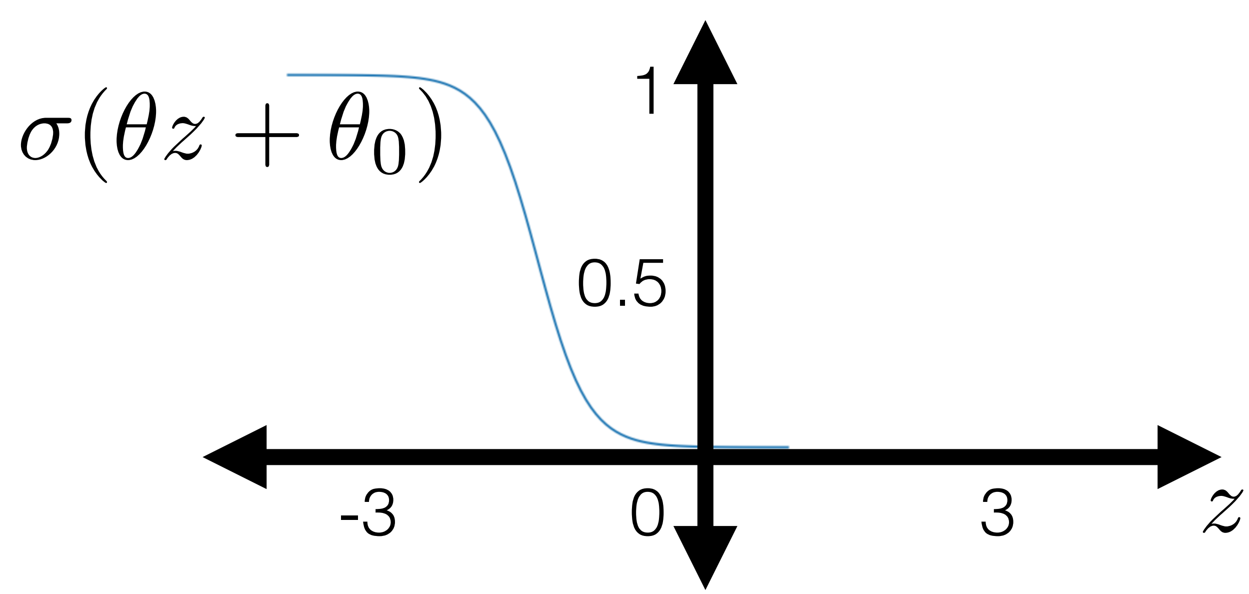
$$\sigma(z) = \frac{1}{1 + \exp(-z)}$$

Capturing uncertainty

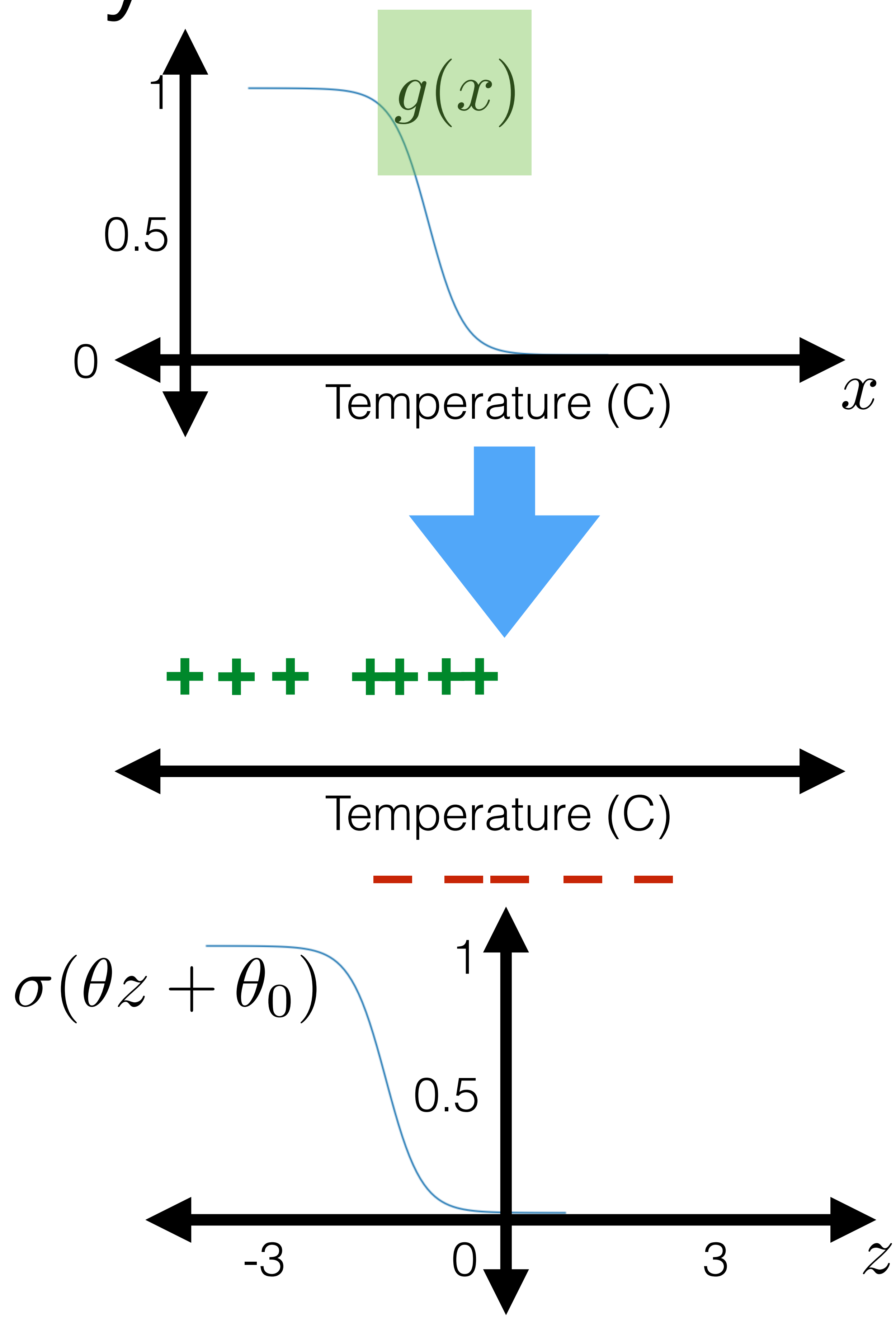


- How to make this shape?
  - Sigmoid/logistic function

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



Capturing uncertainty

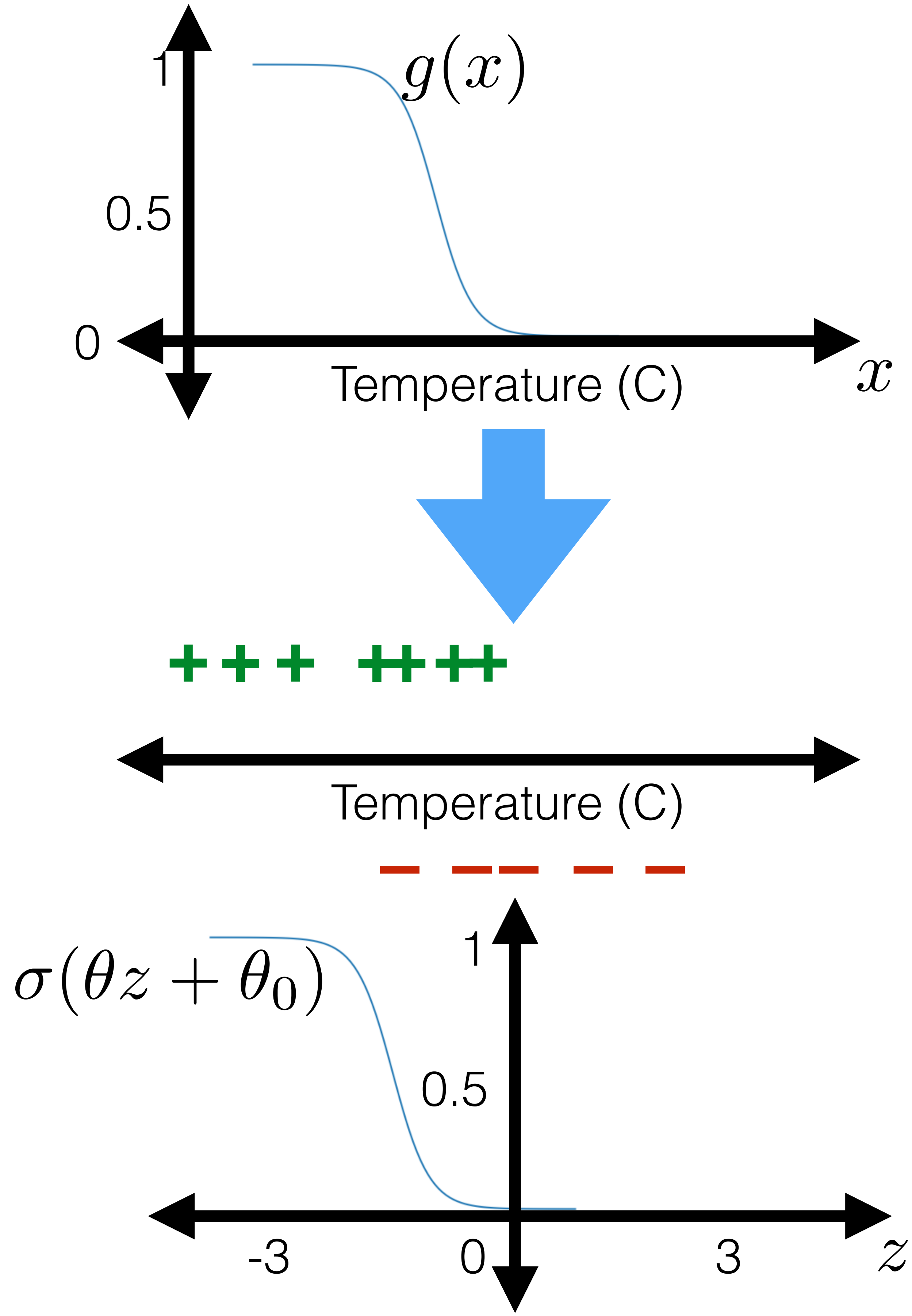


- How to make this shape?
  - Sigmoid/logistic function

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

Capturing uncertainty

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

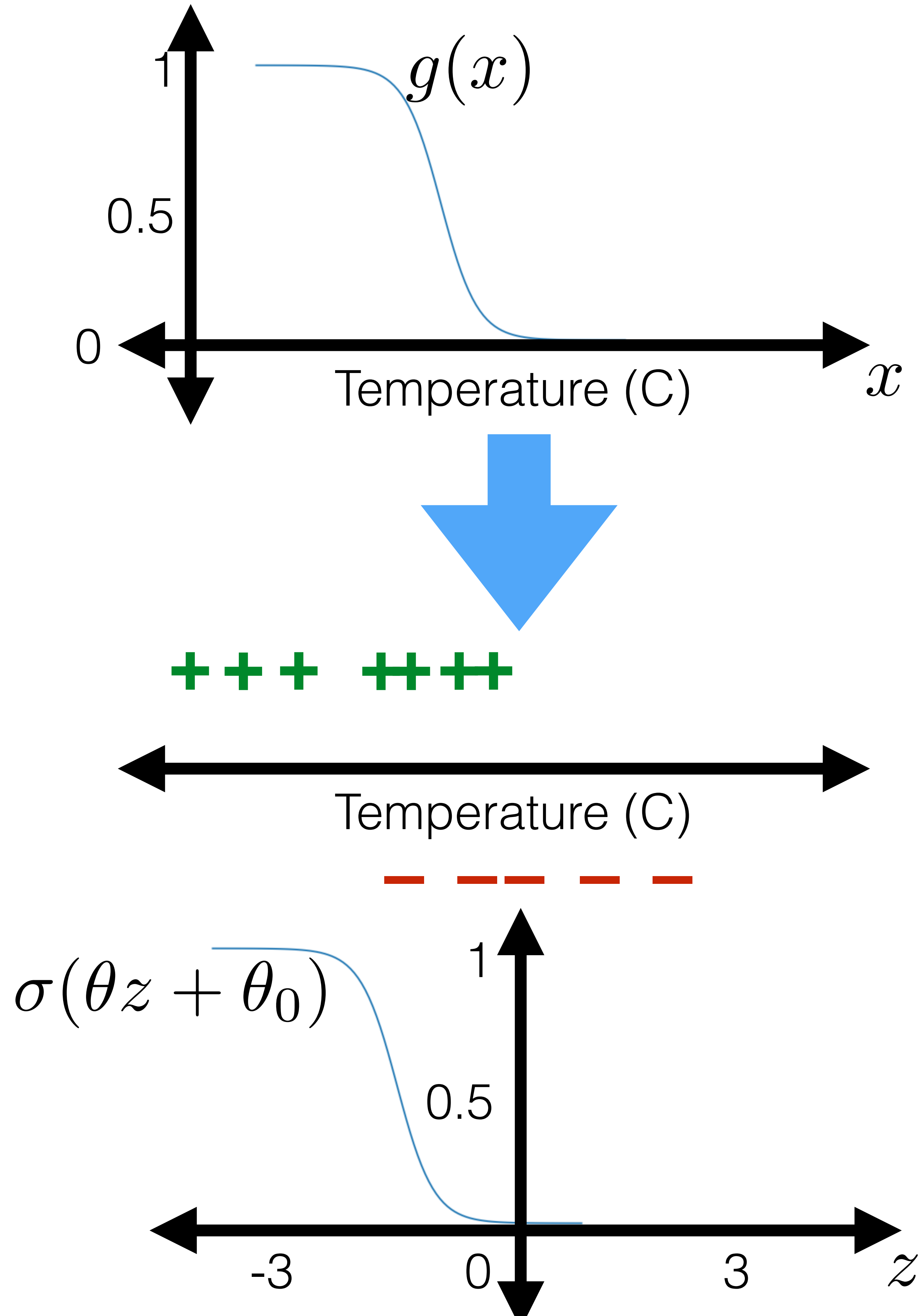


- How to make this shape?
  - Sigmoid/logistic function

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

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:

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

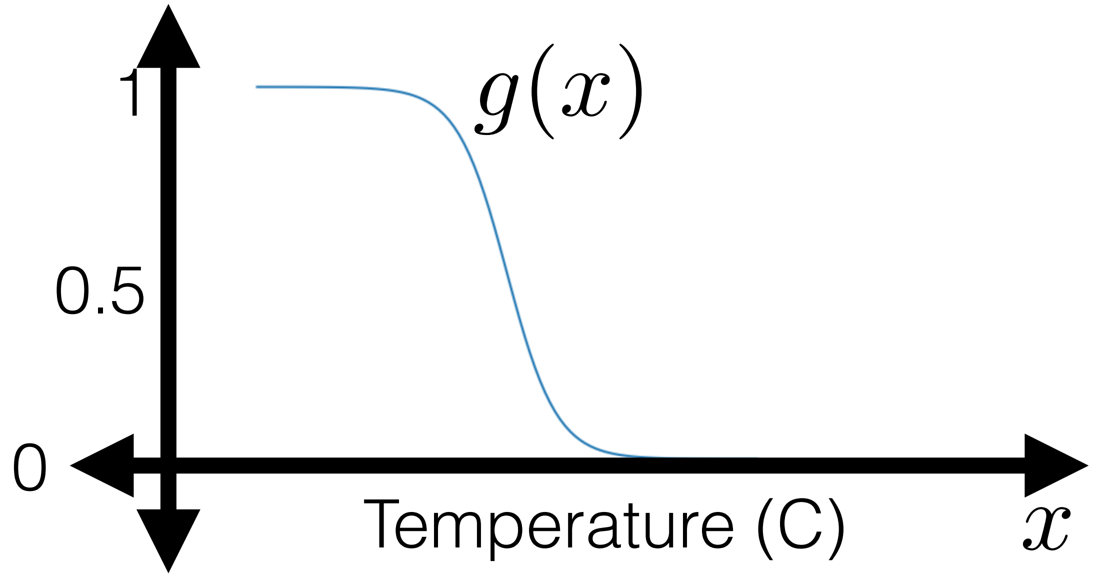
$$= \frac{1}{1 + \exp\{-(\theta x + \theta_0)\}}$$

Capturing uncertainty

1 feature:

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

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

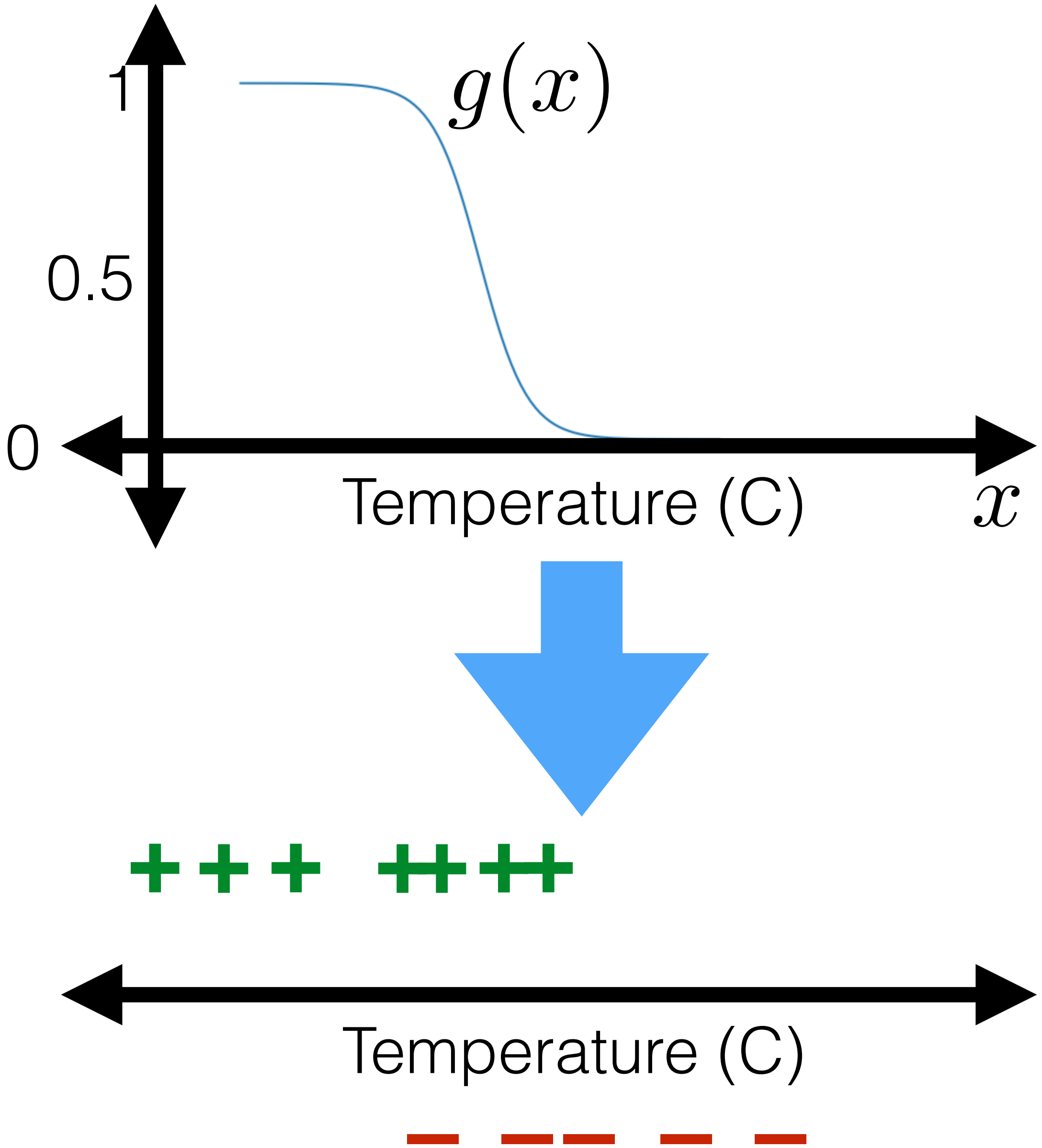


Capturing uncertainty

1 feature:

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

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



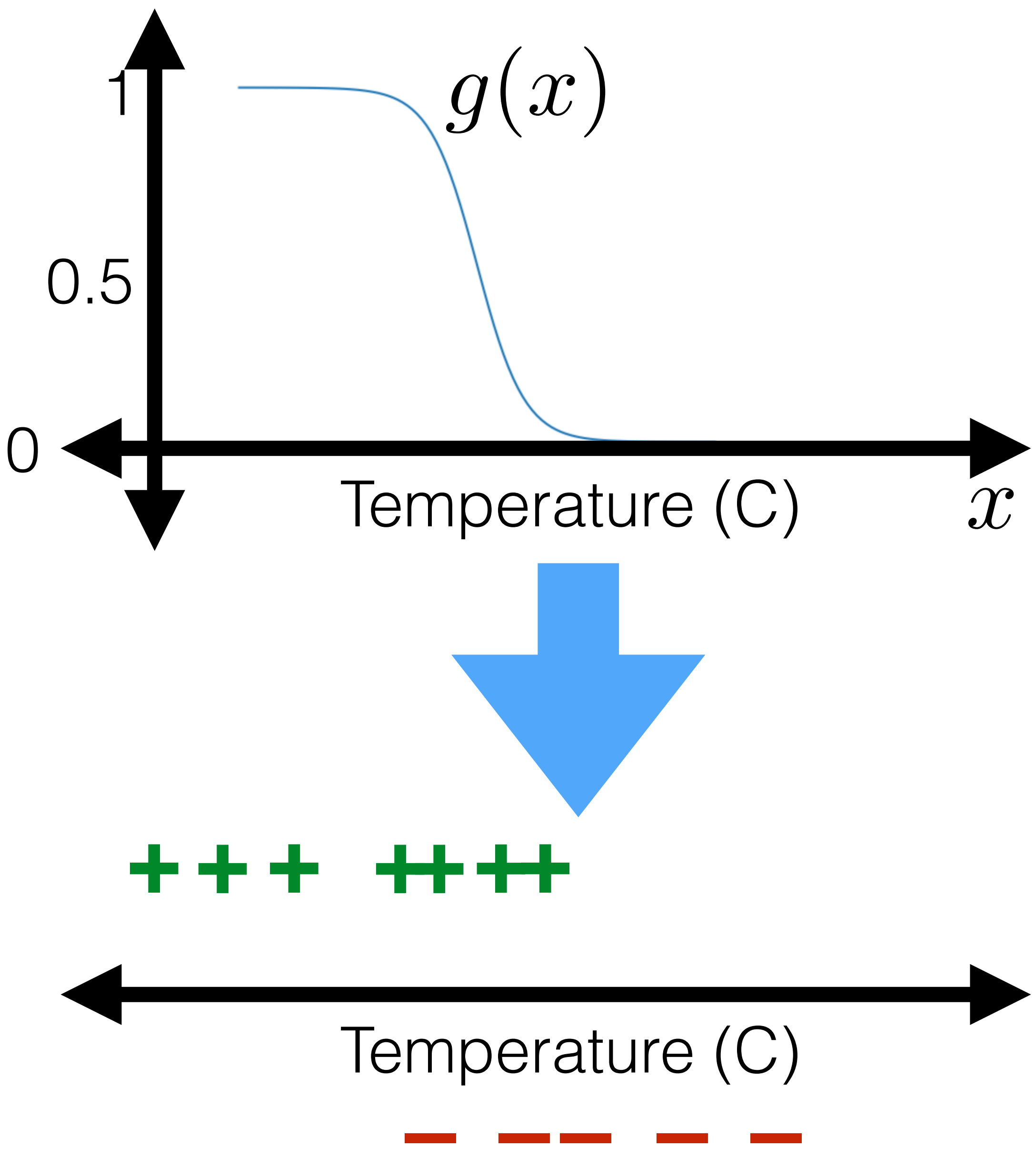
Capturing uncertainty

2 features:

1 feature:

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

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



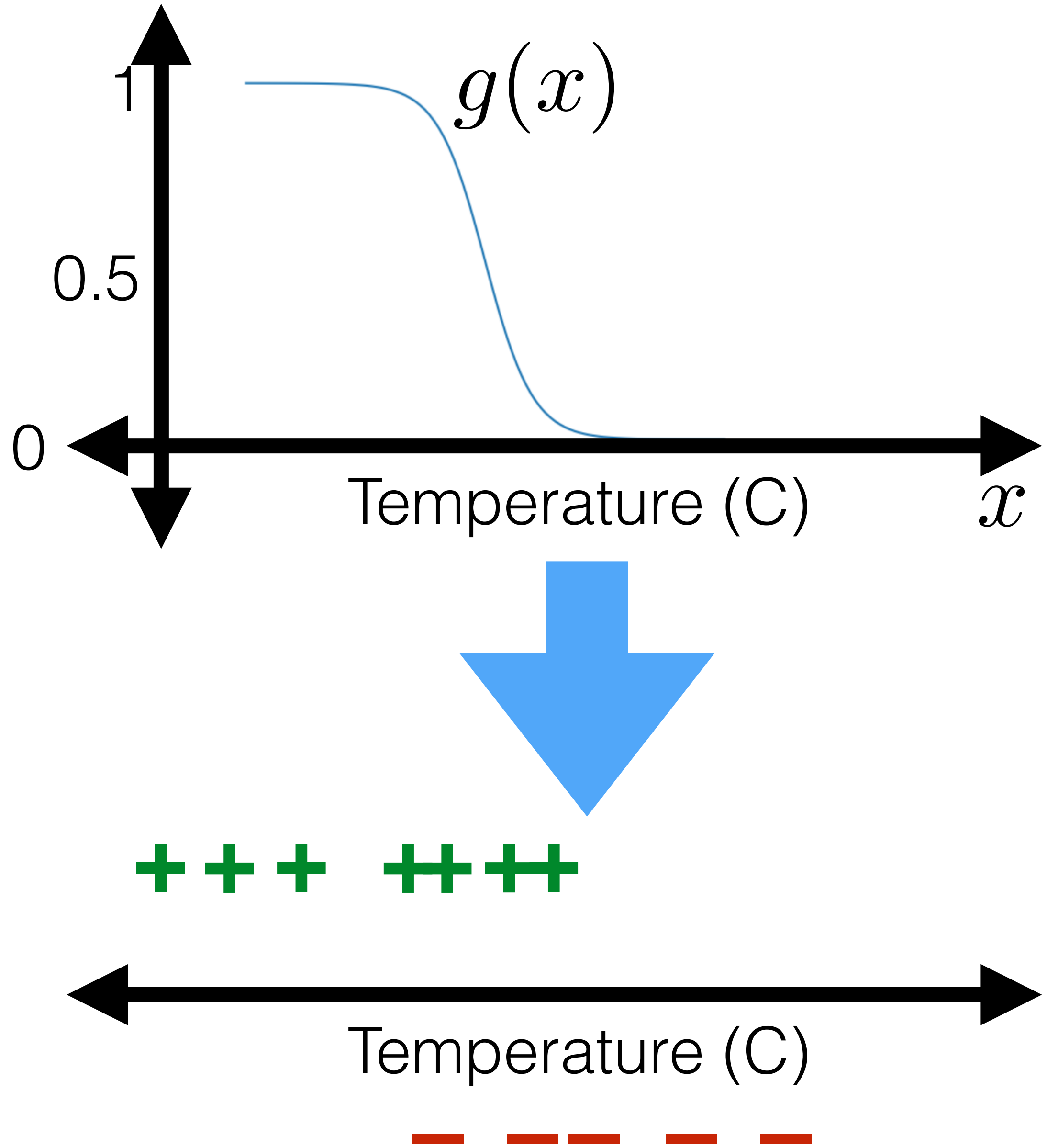
Capturing uncertainty

1 feature:

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

2 features:

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

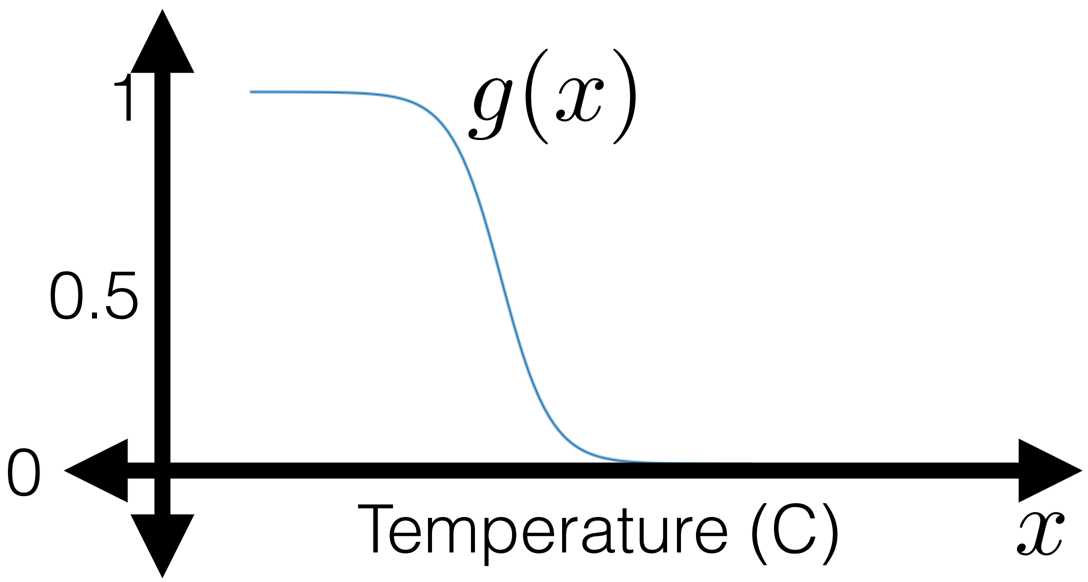


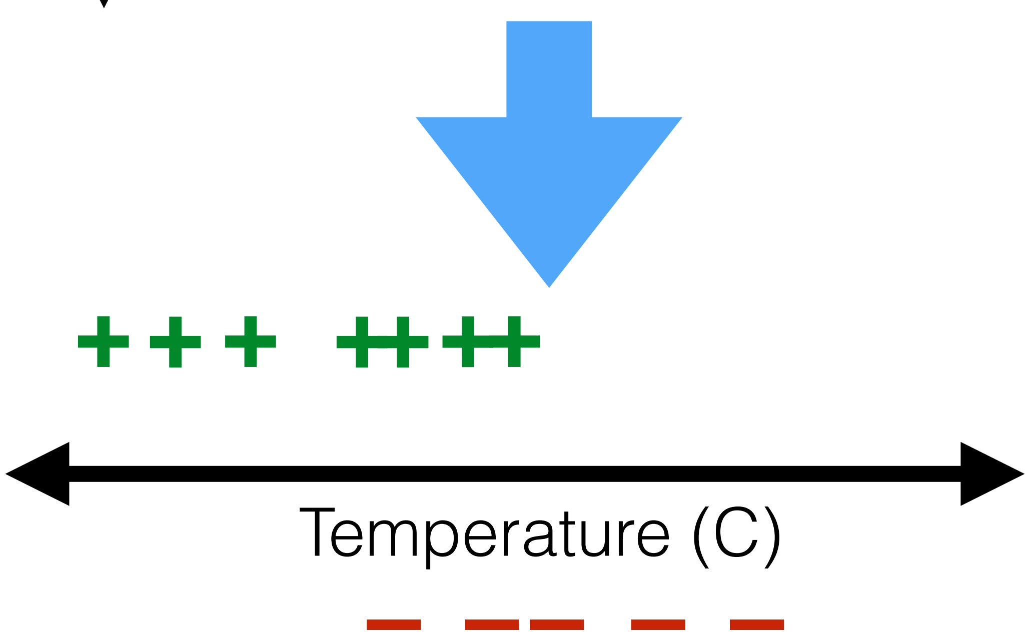
Capturing uncertainty

1 feature:

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

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

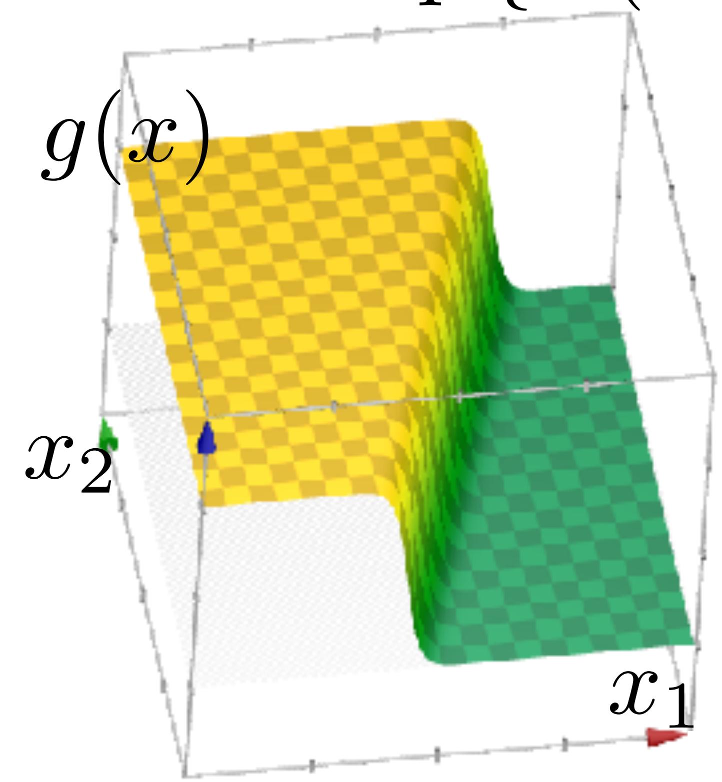




2 features:

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

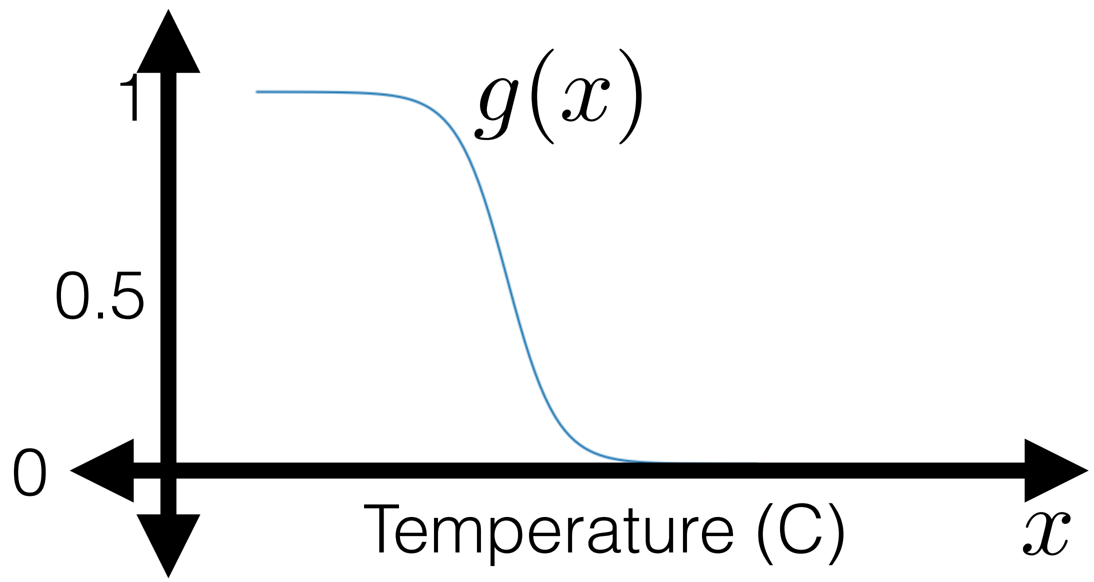
$$= \frac{1}{1 + \exp\{-(\theta^\top x + \theta_0)\}}$$

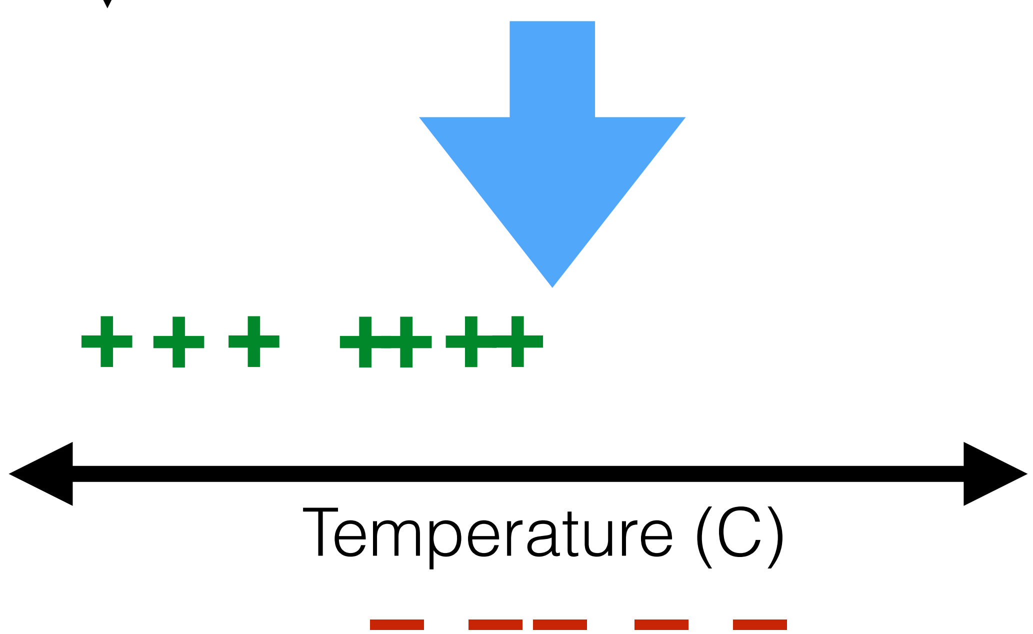


Capturing uncertainty

1 feature:

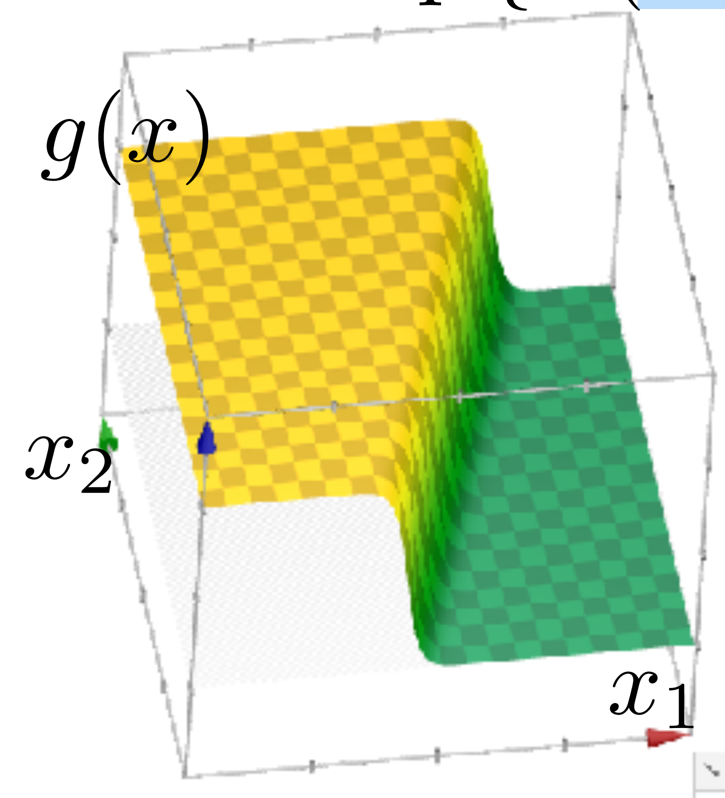
$$g(x) = \sigma(\theta x + \theta_0)$$
$$= \frac{1}{1 + \exp\{-(\theta x + \theta_0)\}}$$





2 features:

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

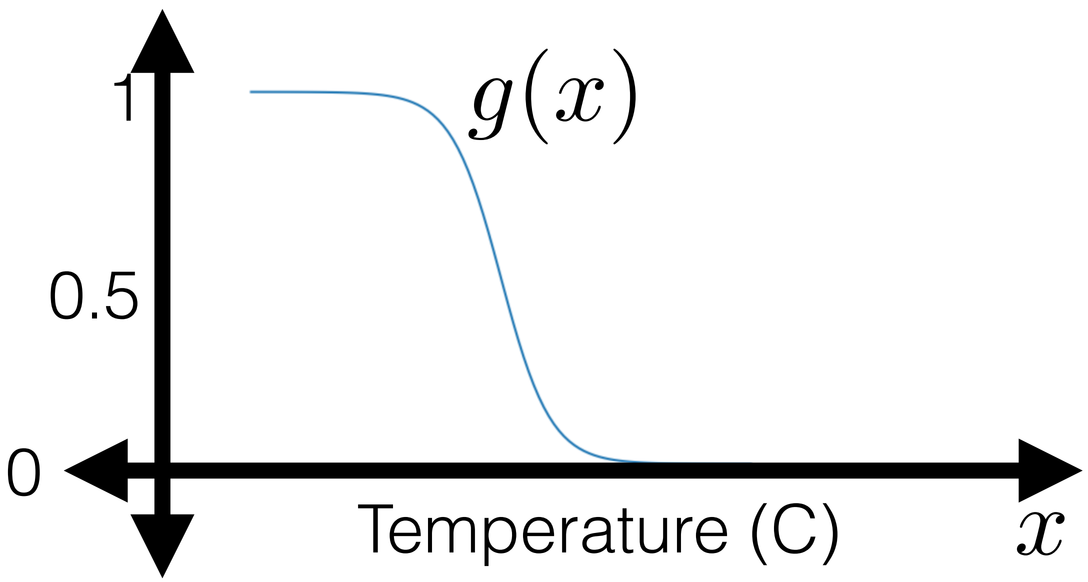


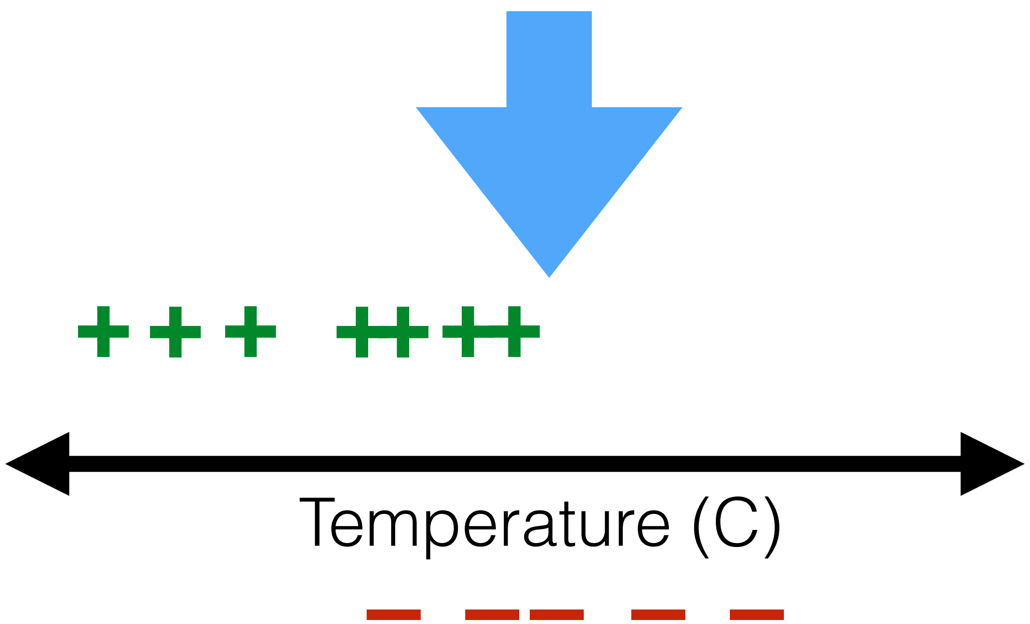
Capturing uncertainty

1 feature:

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

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

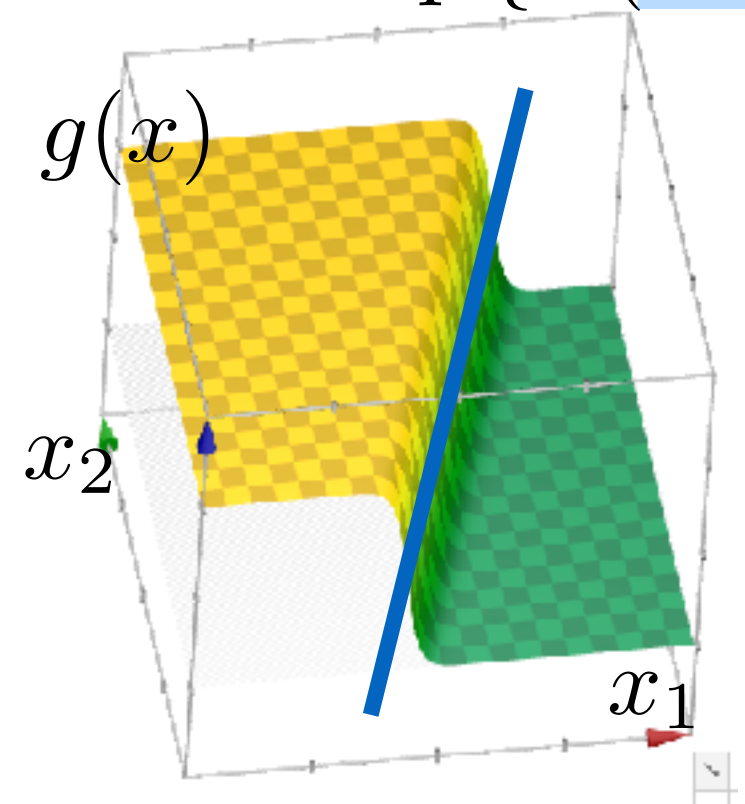




2 features:

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

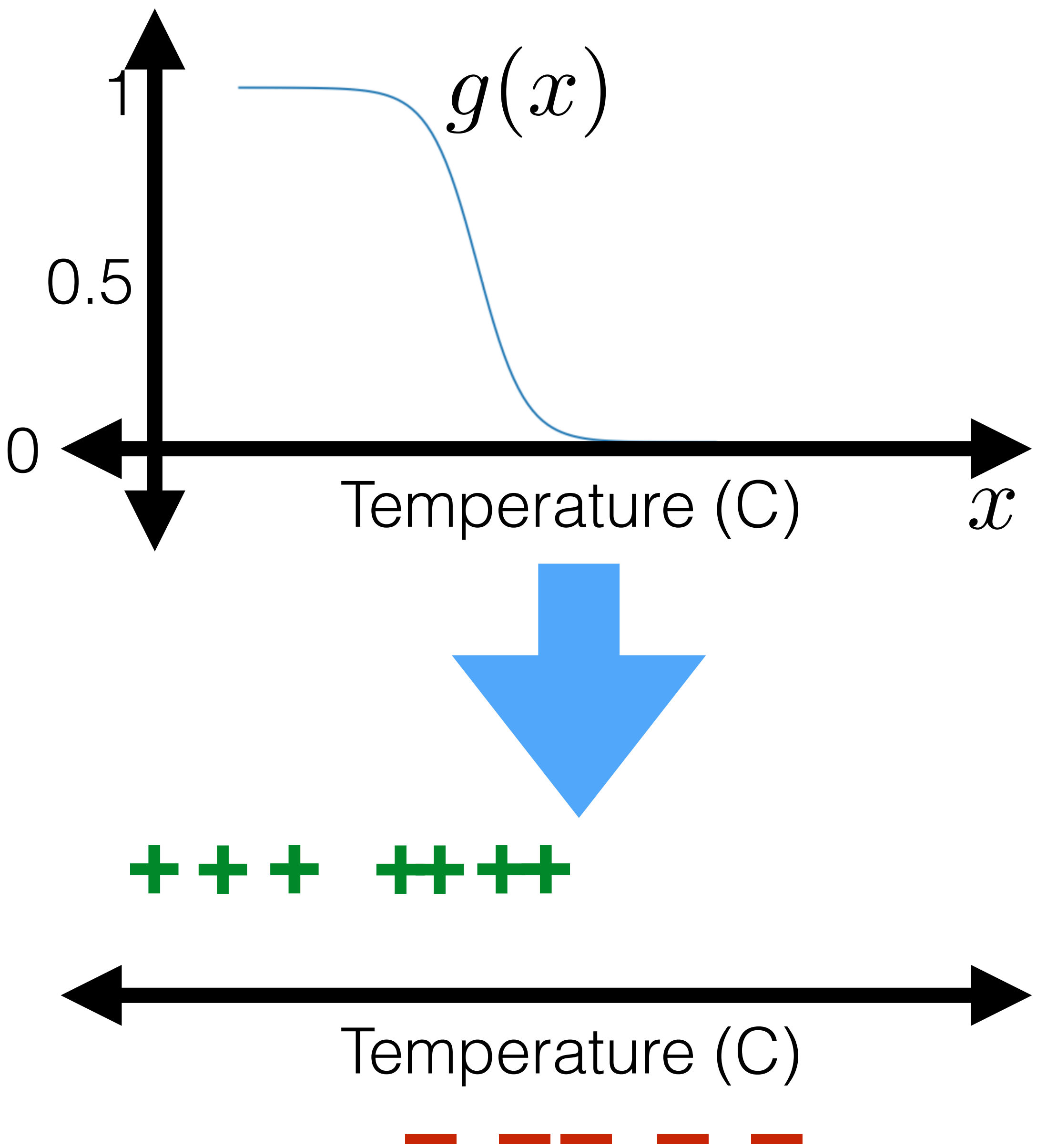
$$= \frac{1}{1 + \exp\{-(\theta^\top x + \theta_0)\}}$$



Capturing uncertainty

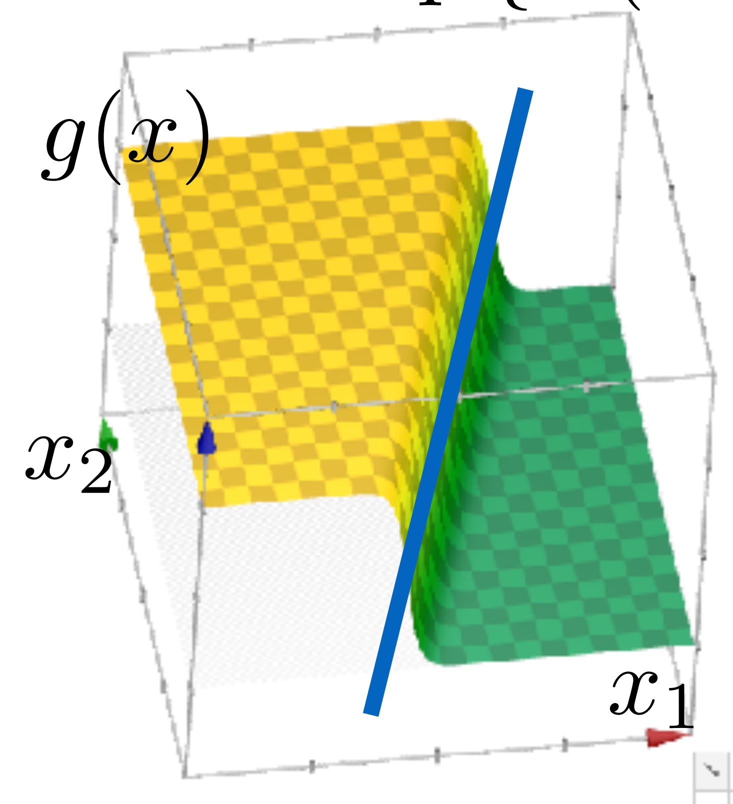
1 feature:

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



2 features:

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

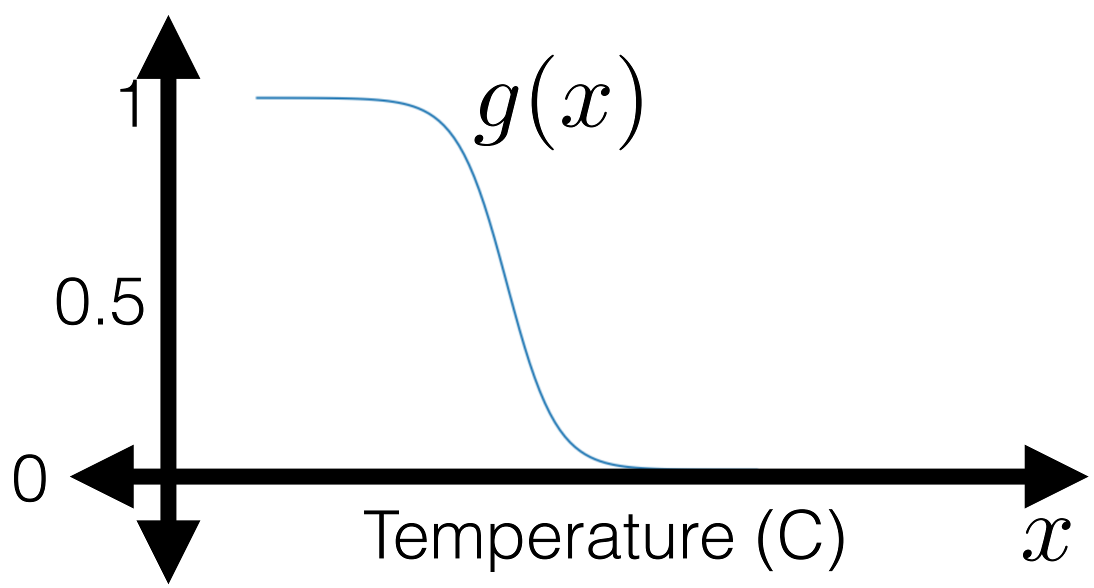


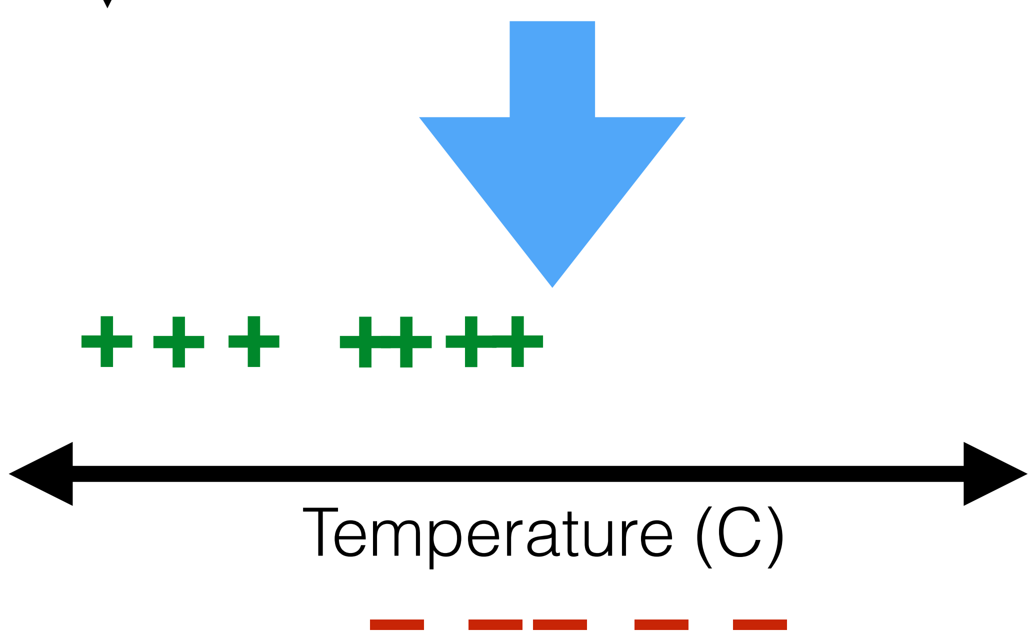
Capturing uncertainty

1 feature:

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

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

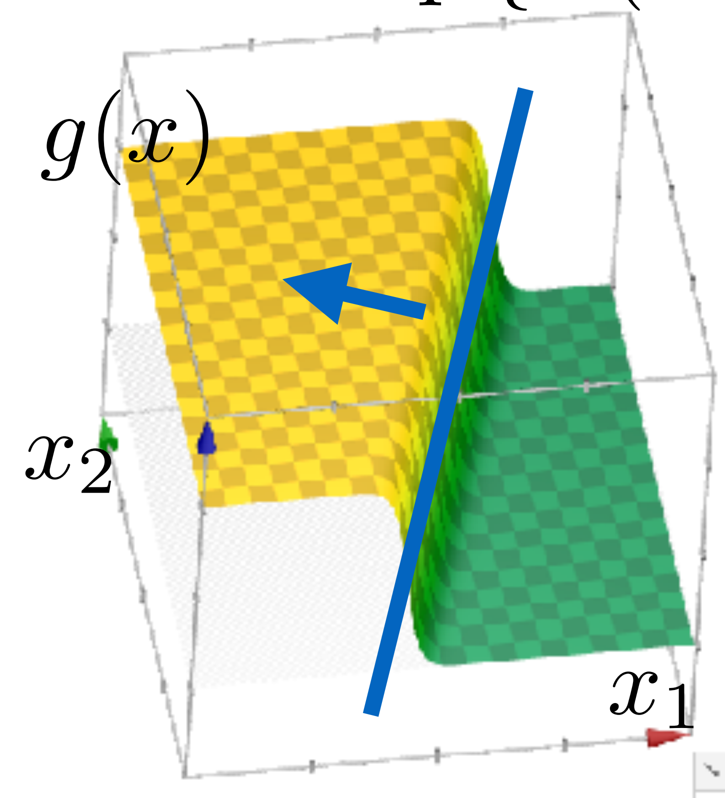




2 features:

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

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



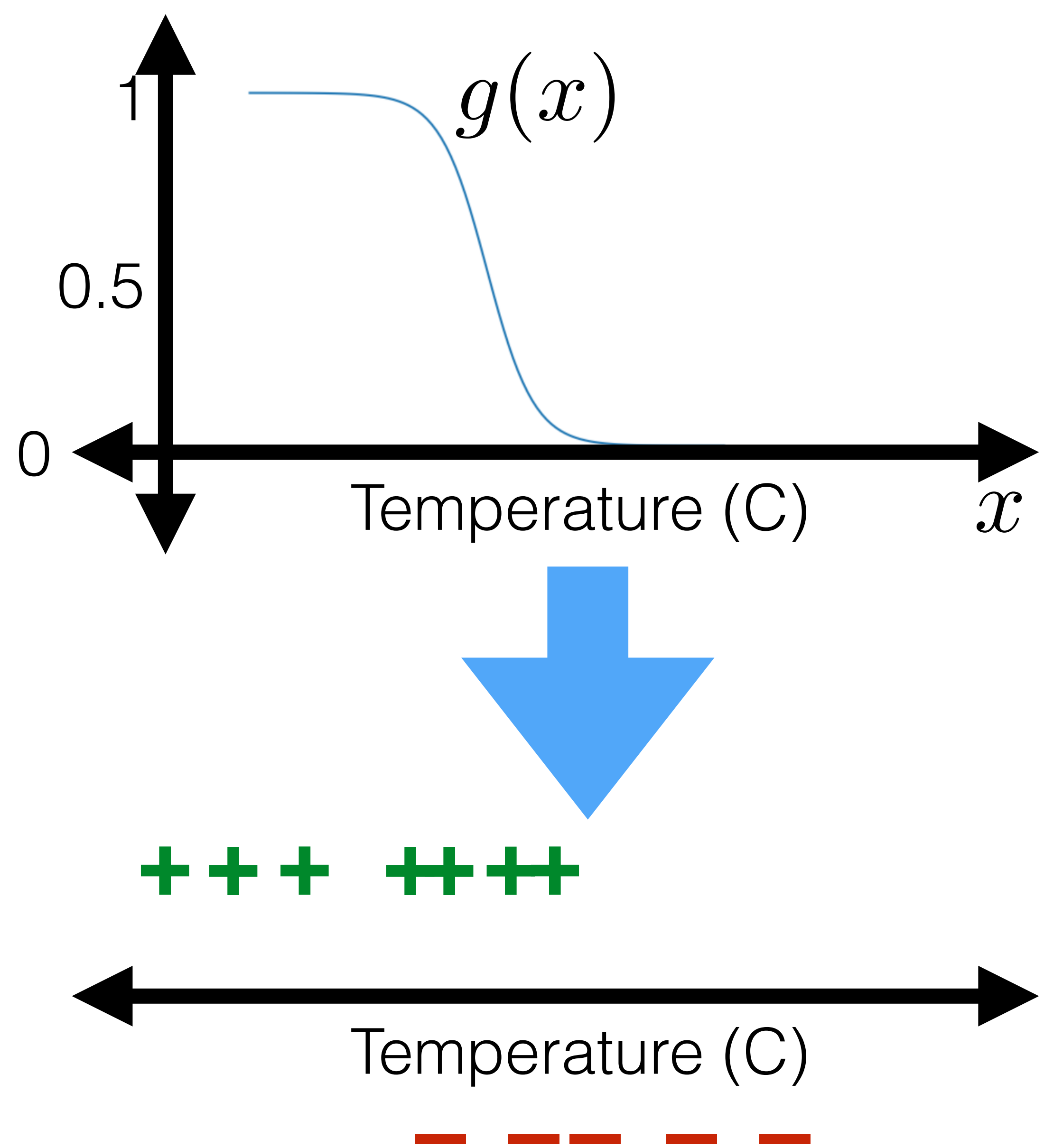
Capturing uncertainty

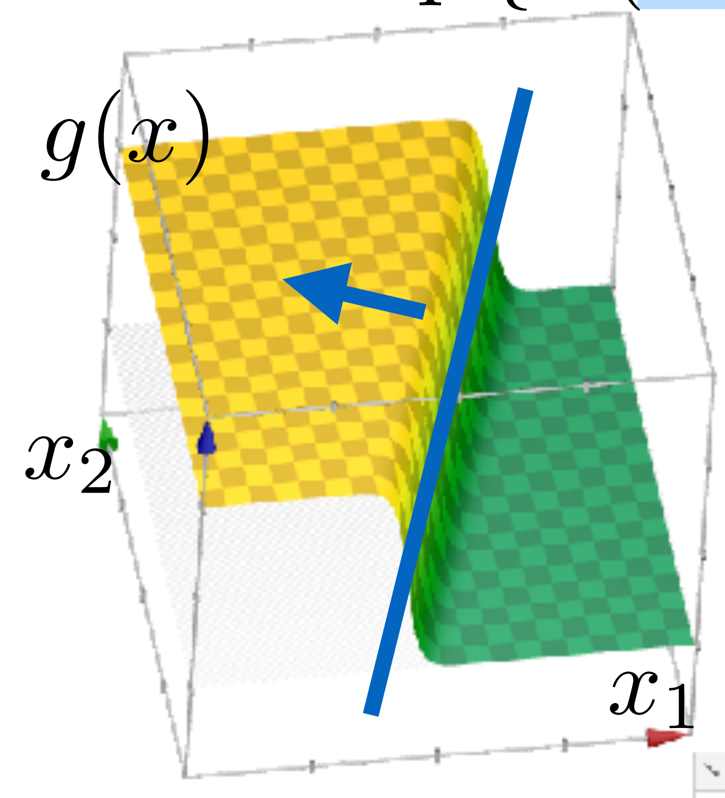
2 features:

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

1 feature:

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





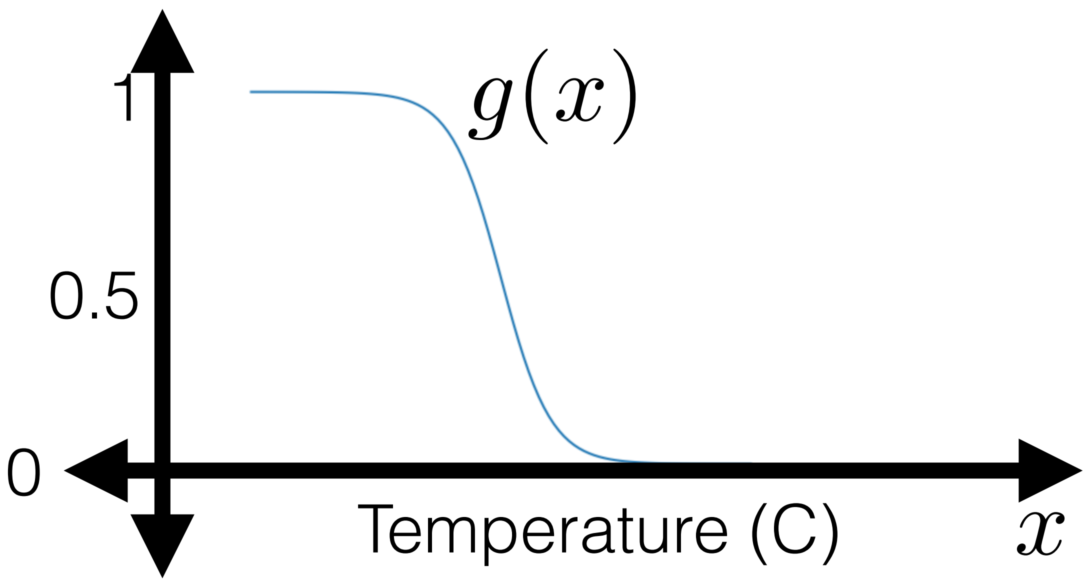
Capturing uncertainty

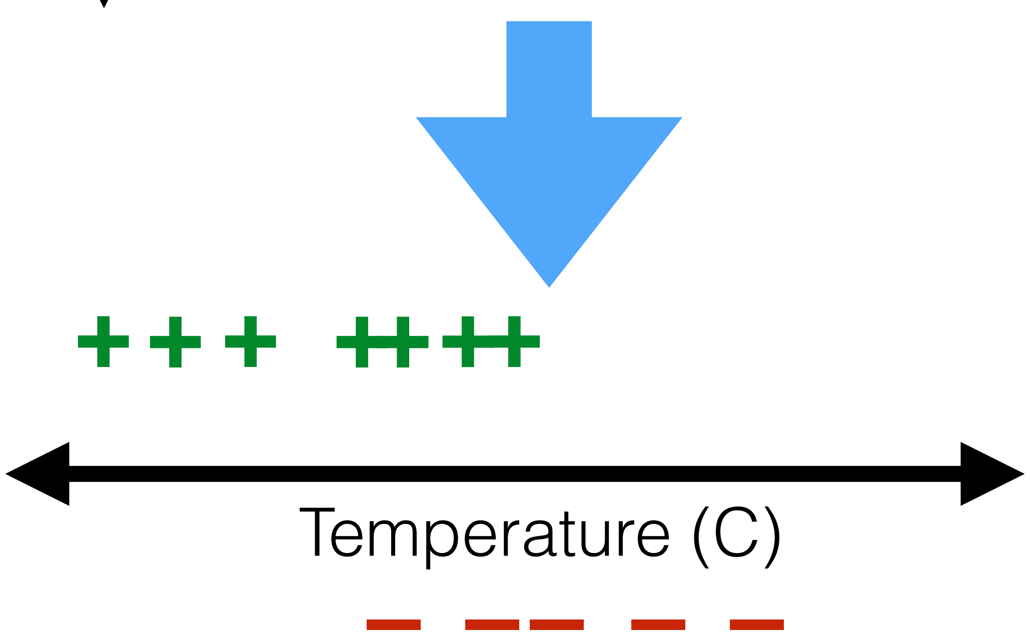
1 feature:

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

$g(x) =$

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

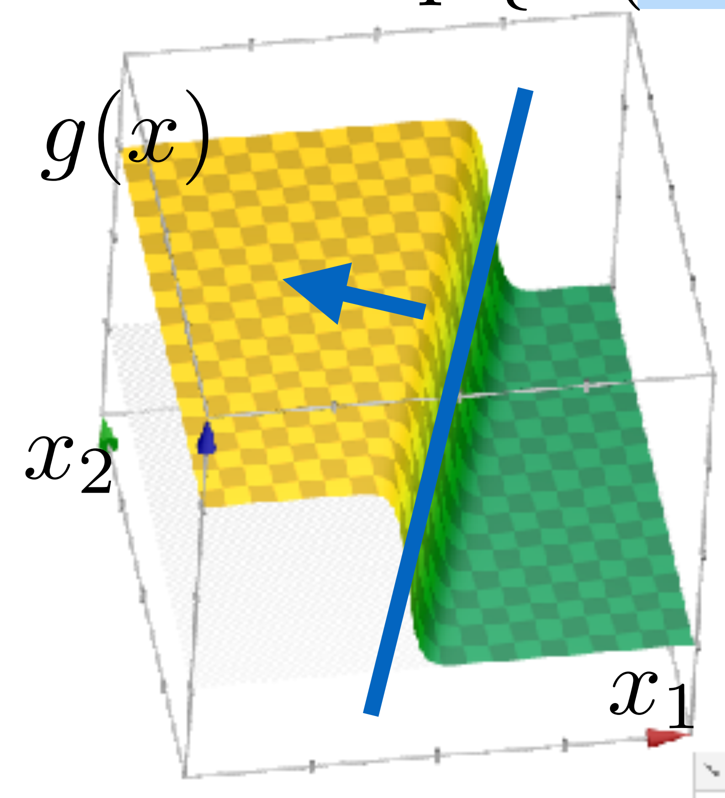




2 features:

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

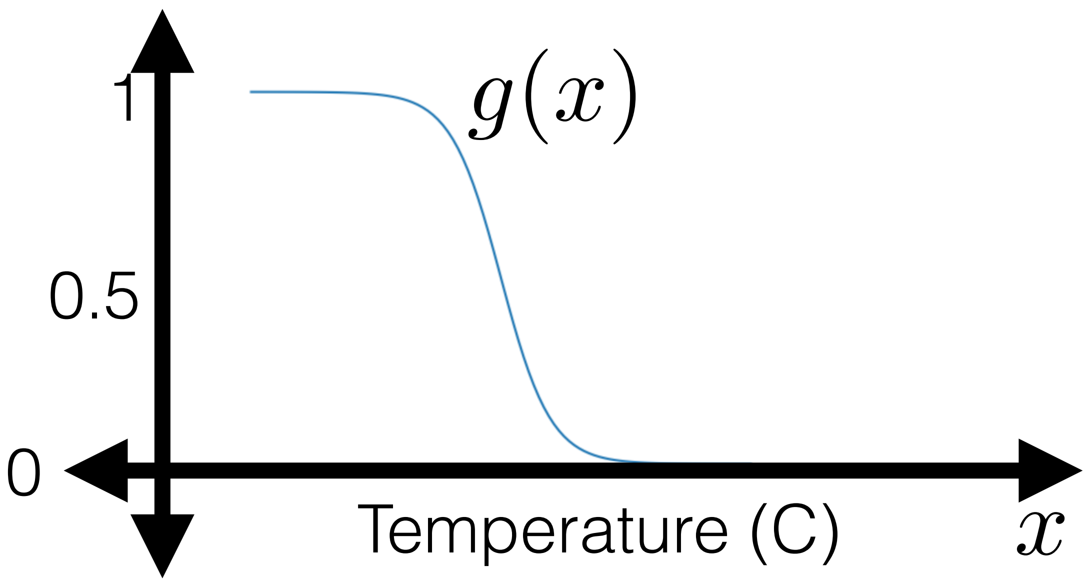
$$= \frac{1}{1 + \exp\{-(\theta^\top x + \theta_0)\}}$$

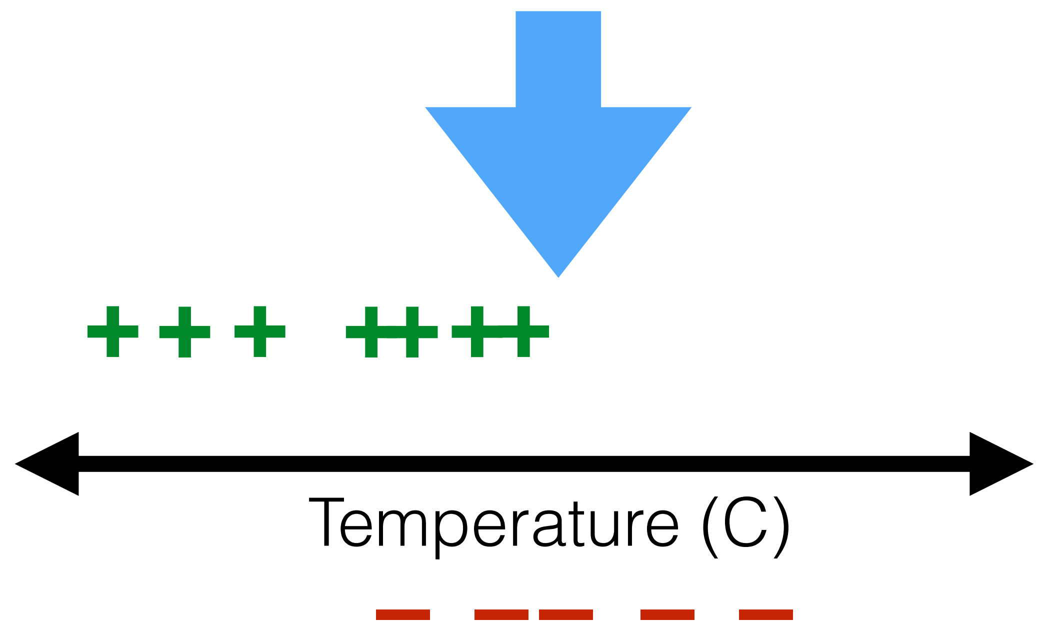


Capturing uncertainty

1 feature:

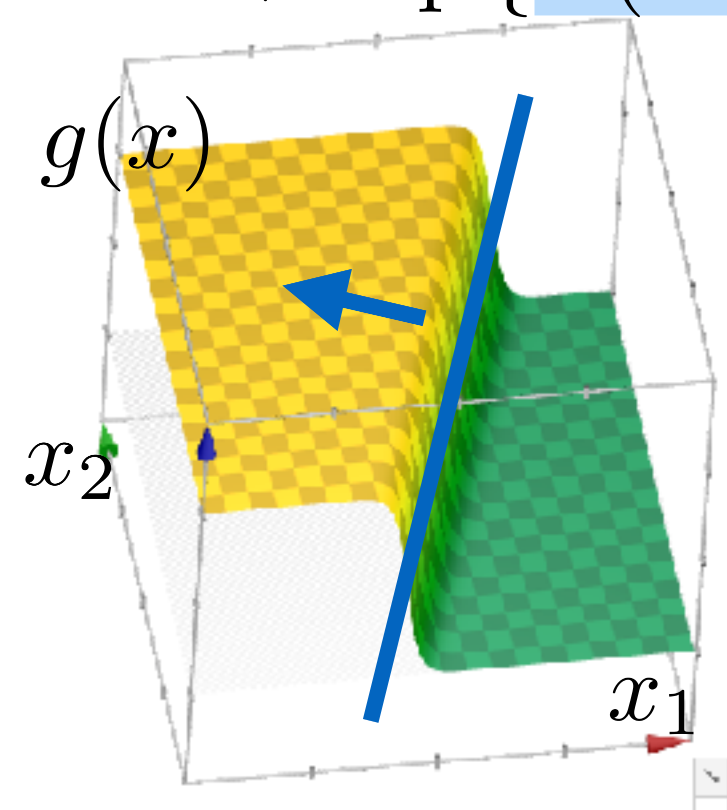
$$g(x) = \sigma(\theta x + \theta_0)$$
$$= \frac{1}{1 + \exp\{-(\theta x + \theta_0)\}}$$





2 features:

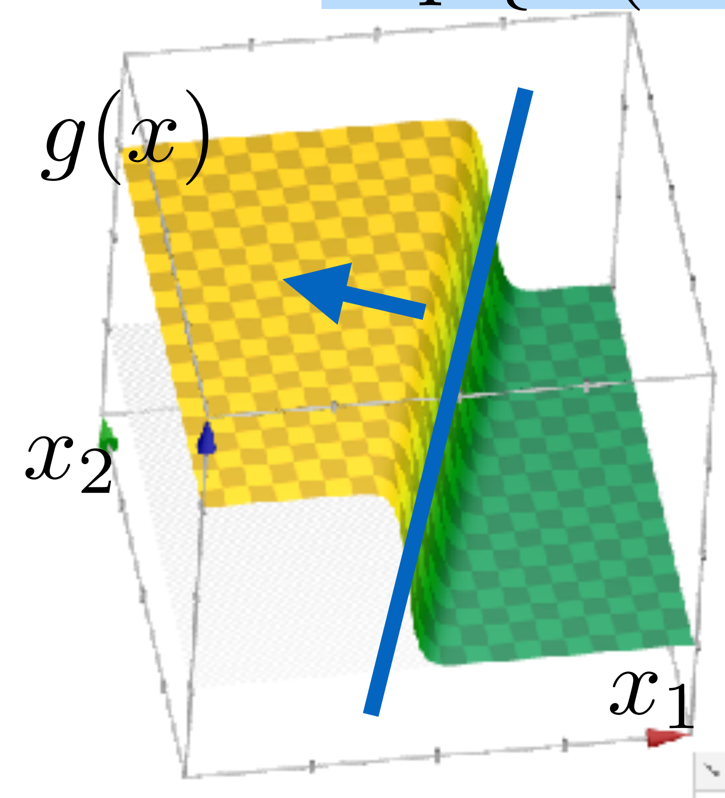
$$g(x) = \sigma(\theta^\top x + \theta_0)$$
$$= \frac{1}{1 + \exp\{-(\theta^\top x + \theta_0)\}}$$



Capturing uncertainty

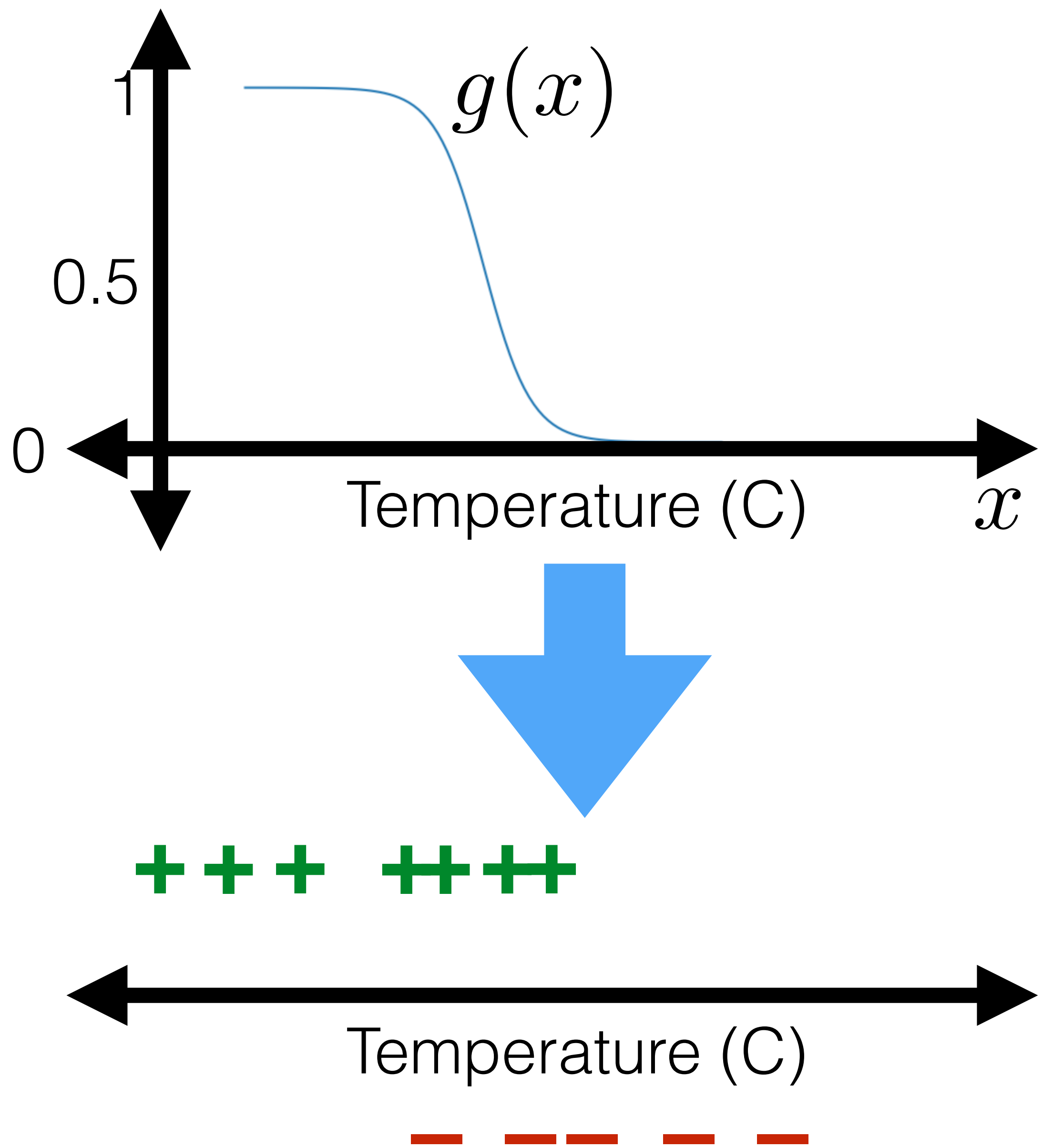
2 features:

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



1 feature:

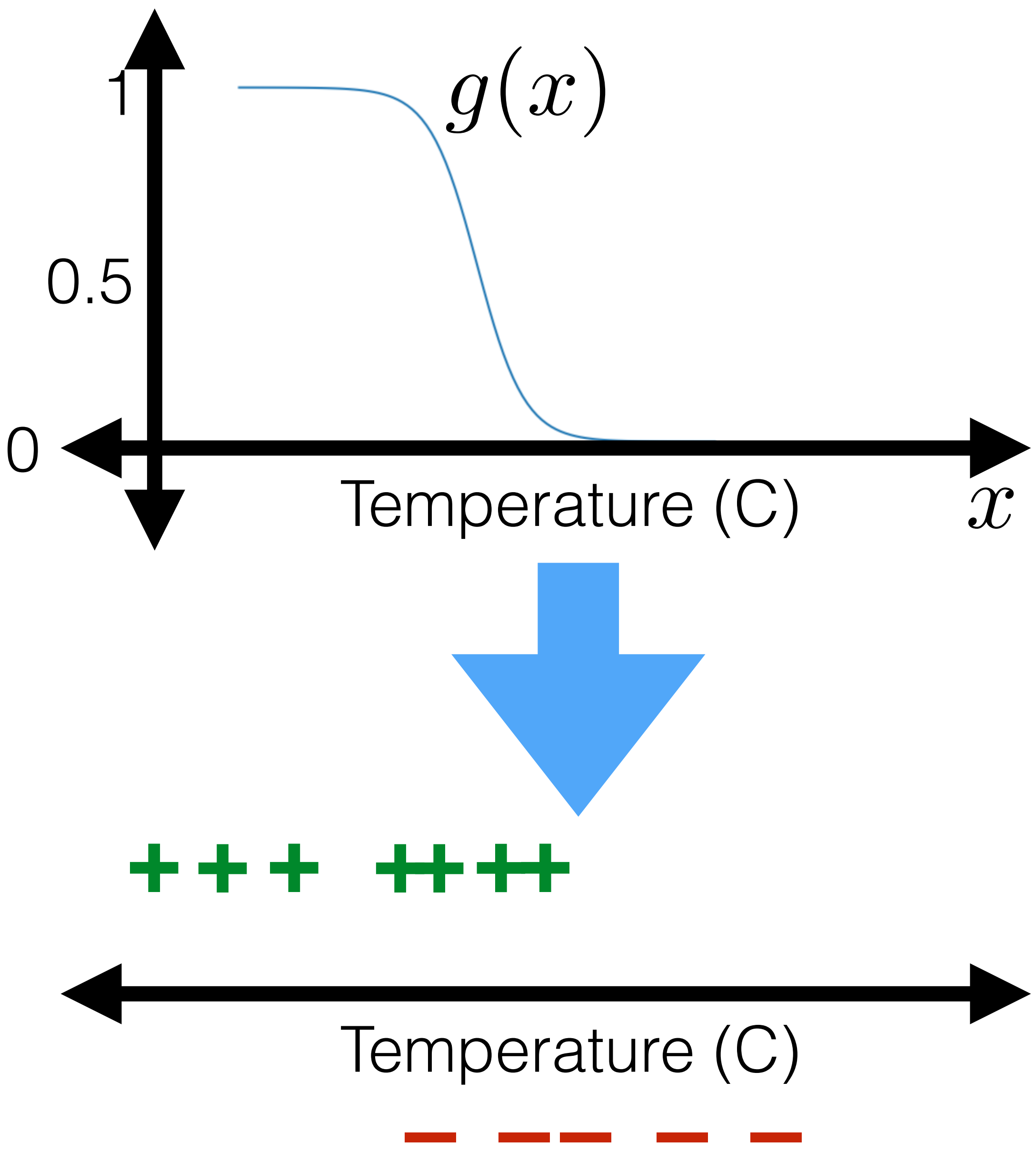
$$g(x) = \sigma(\theta x + \theta_0) = \frac{1}{1 + \exp\{-(\theta x + \theta_0)\}}$$



Capturing uncertainty

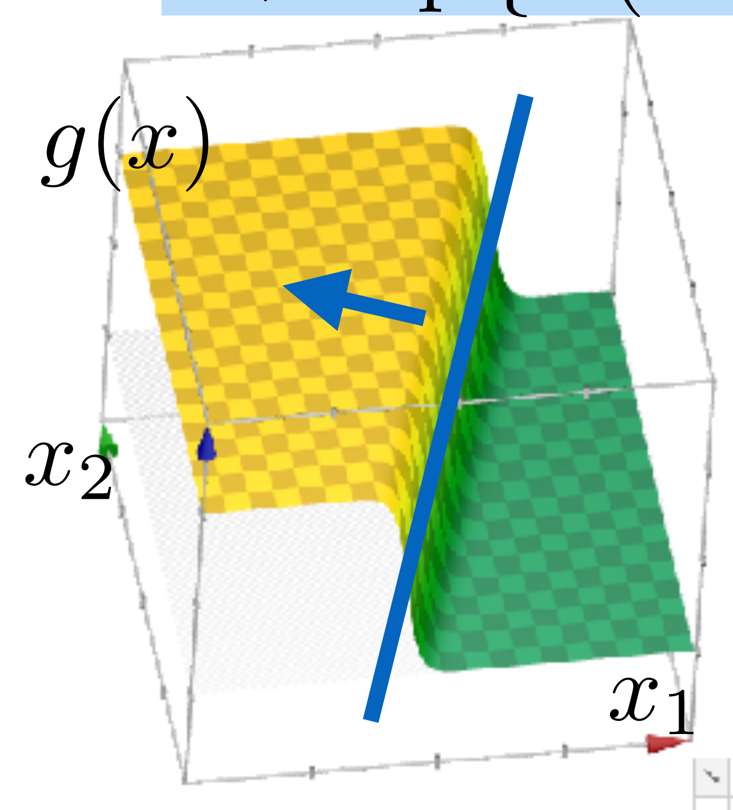
1 feature:

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



2 features:

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

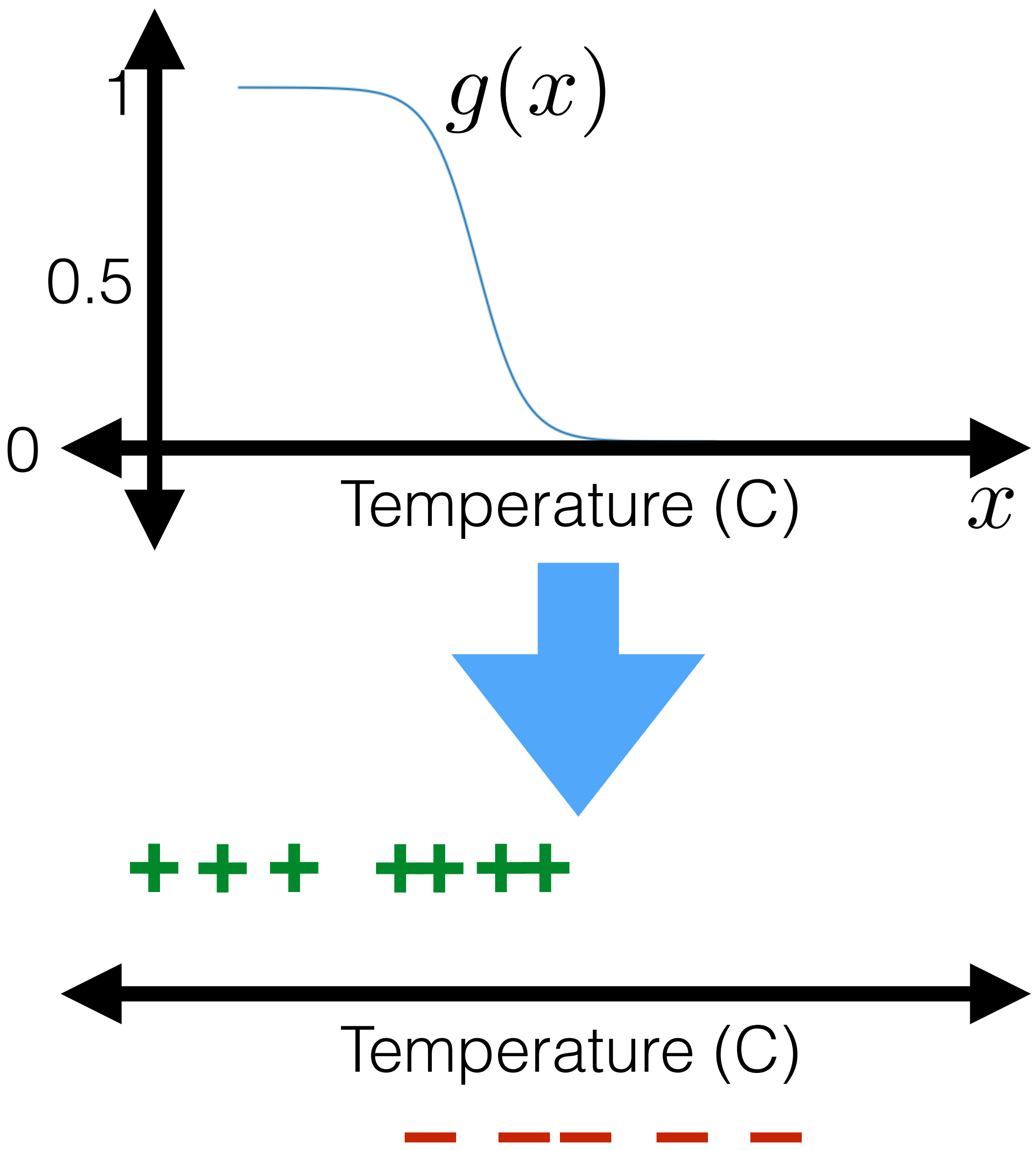


Capturing uncertainty

1 feature:

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

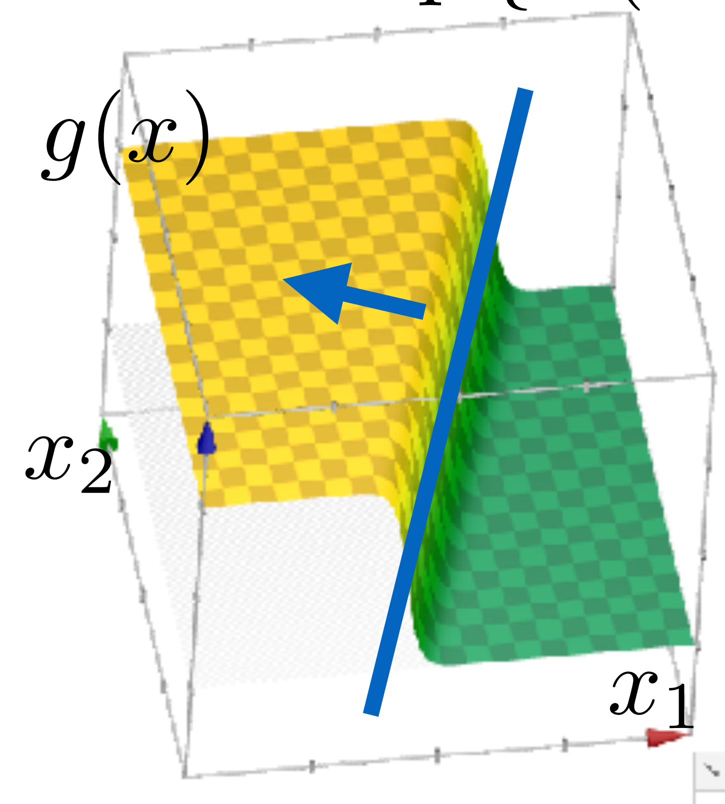
$$= \frac{1}{1 + \exp\{-(\theta x + \theta_0)\}}$$



2 features:

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

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

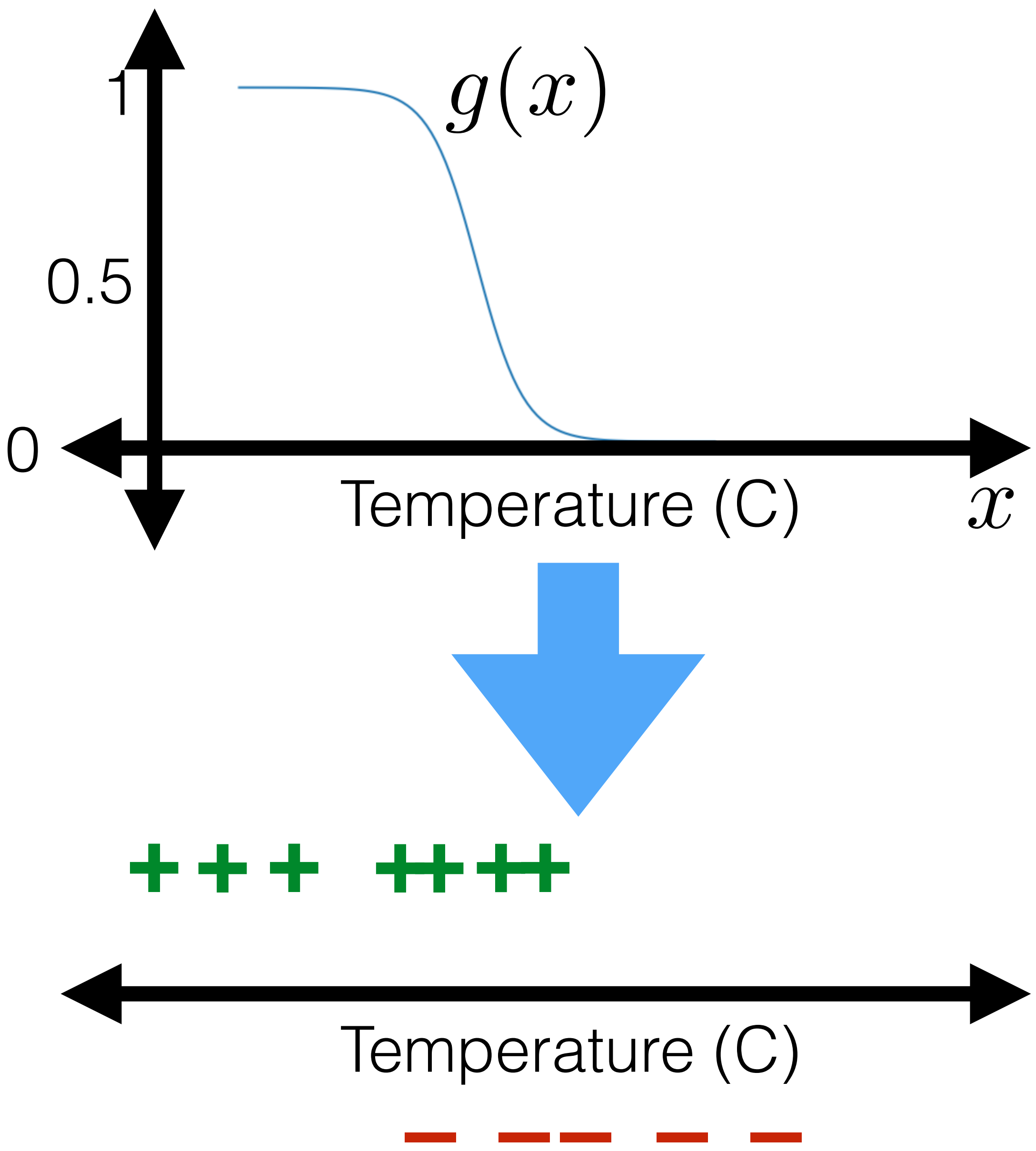


Capturing uncertainty

1 feature:

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

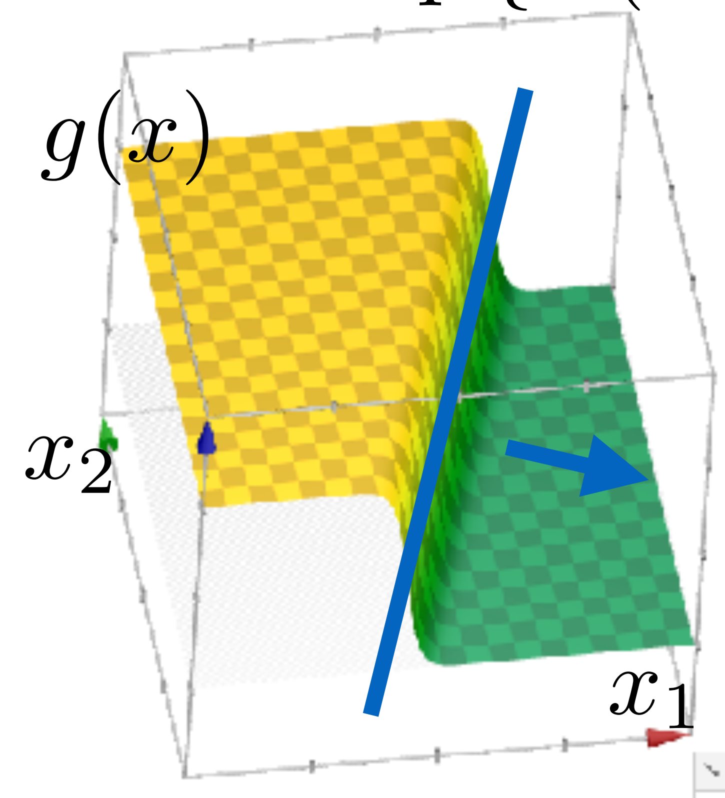
$$= \frac{1}{1 + \exp\{-(\theta x + \theta_0)\}}$$



2 features:

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

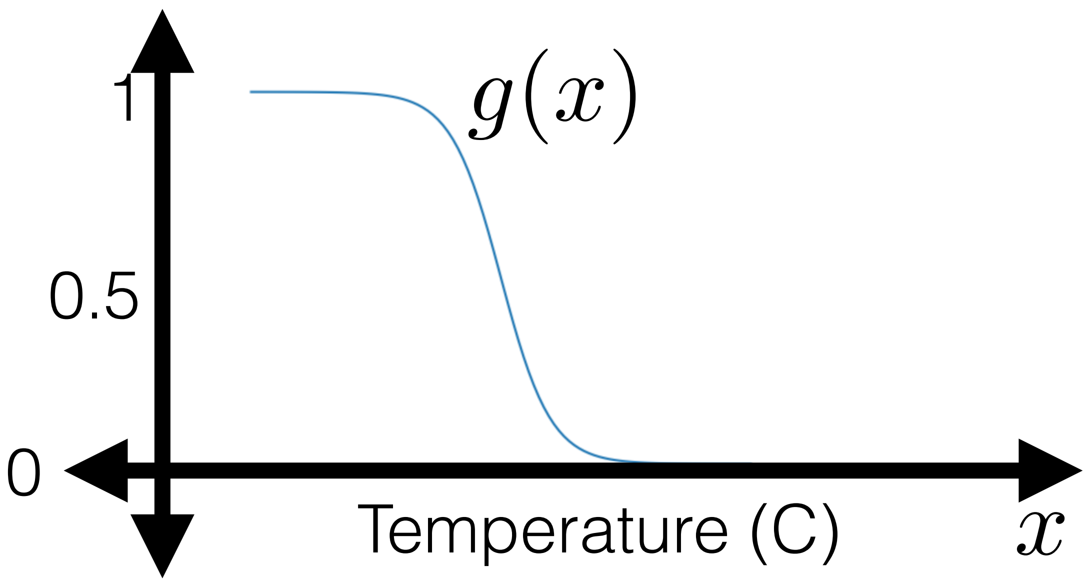
$$= \frac{1}{1 + \exp\{-(\theta^\top x + \theta_0)\}}$$

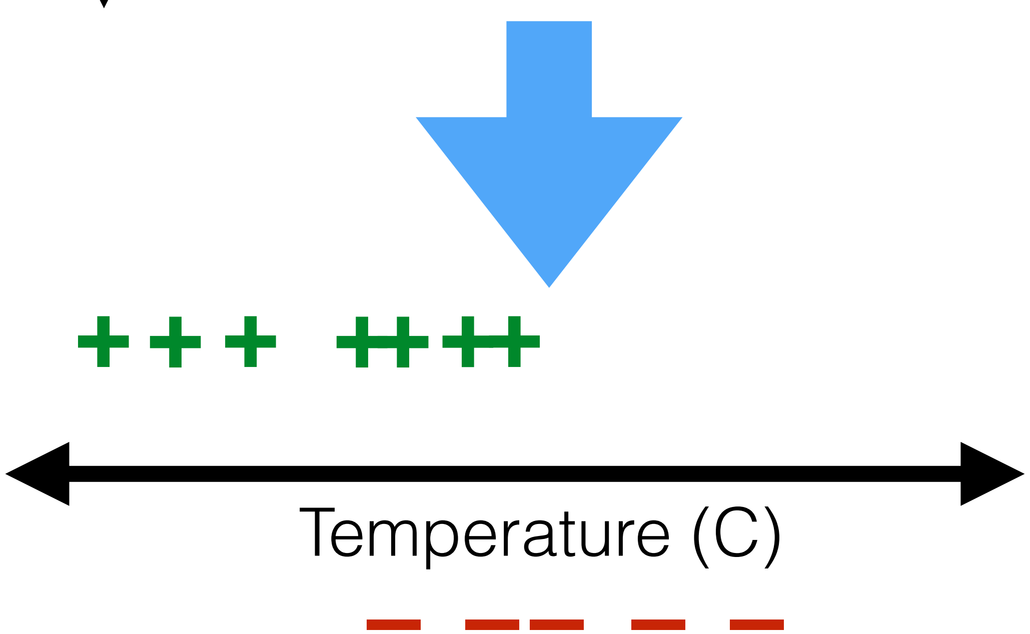


Capturing uncertainty

1 feature:

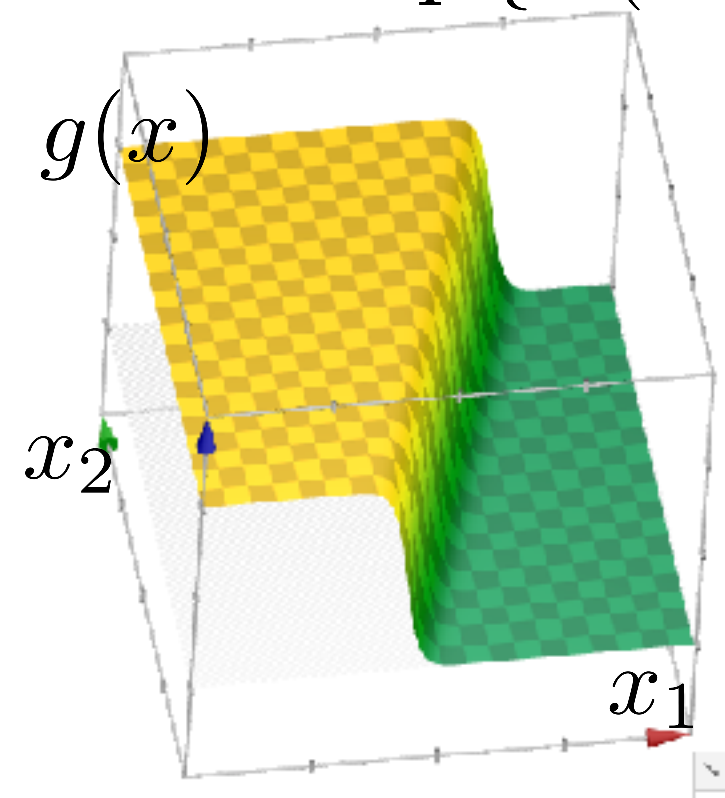
$$g(x) = \sigma(\theta x + \theta_0)$$
$$= \frac{1}{1 + \exp\{-(\theta x + \theta_0)\}}$$





2 features:

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

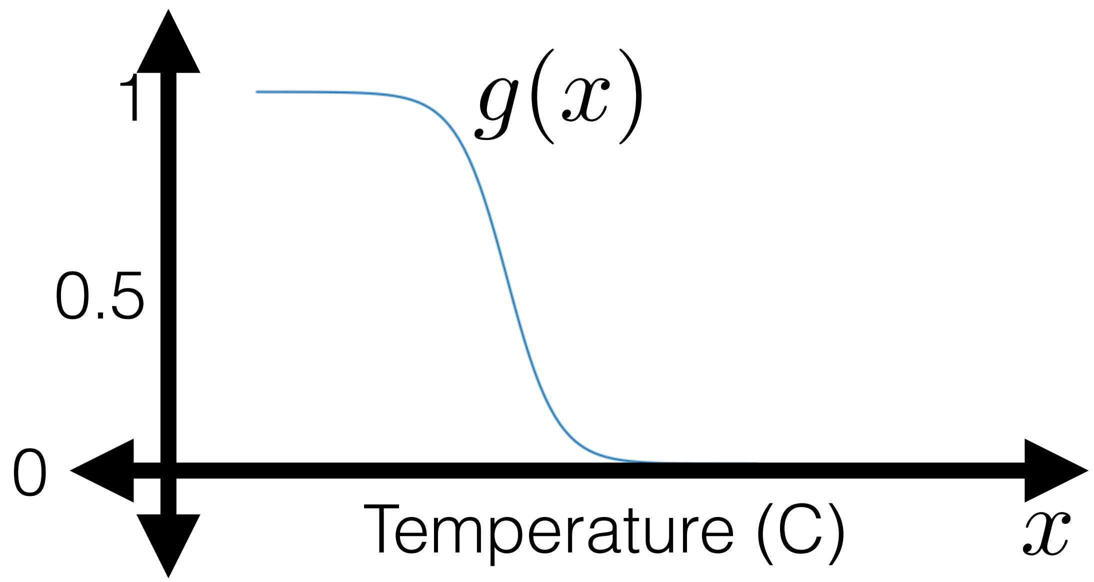


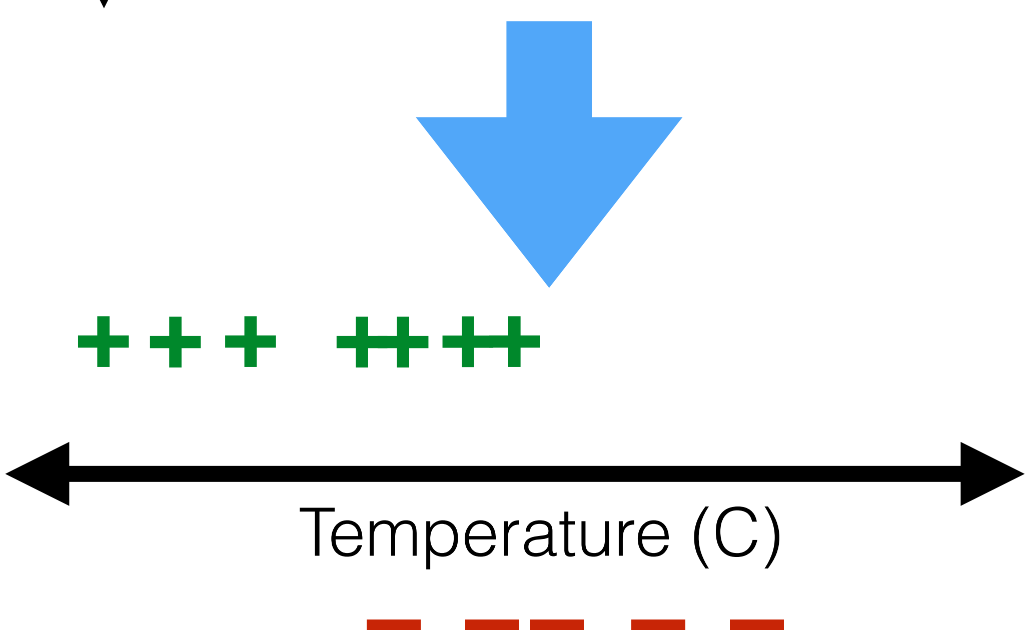
Capturing uncertainty

1 feature:

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

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

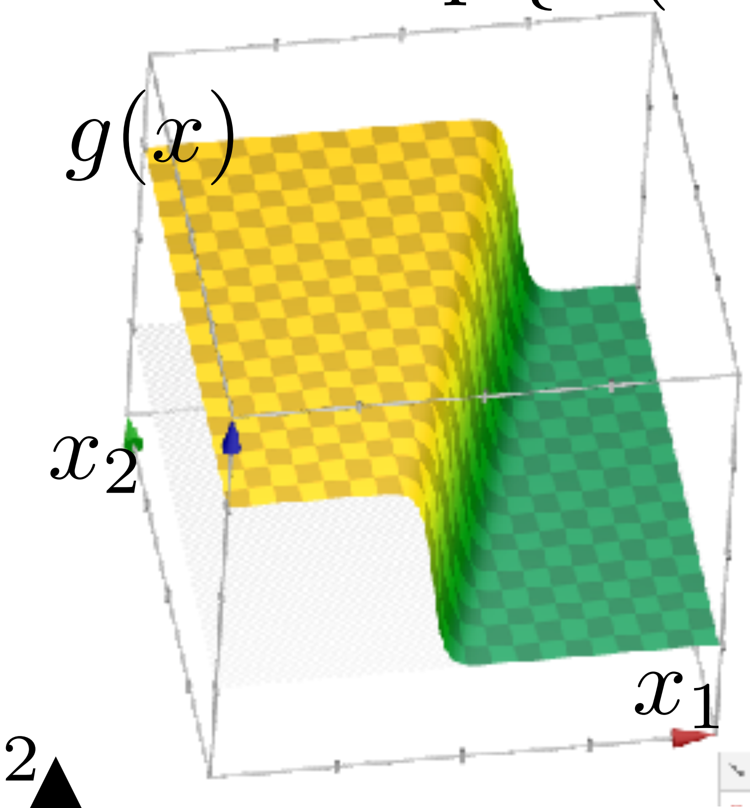


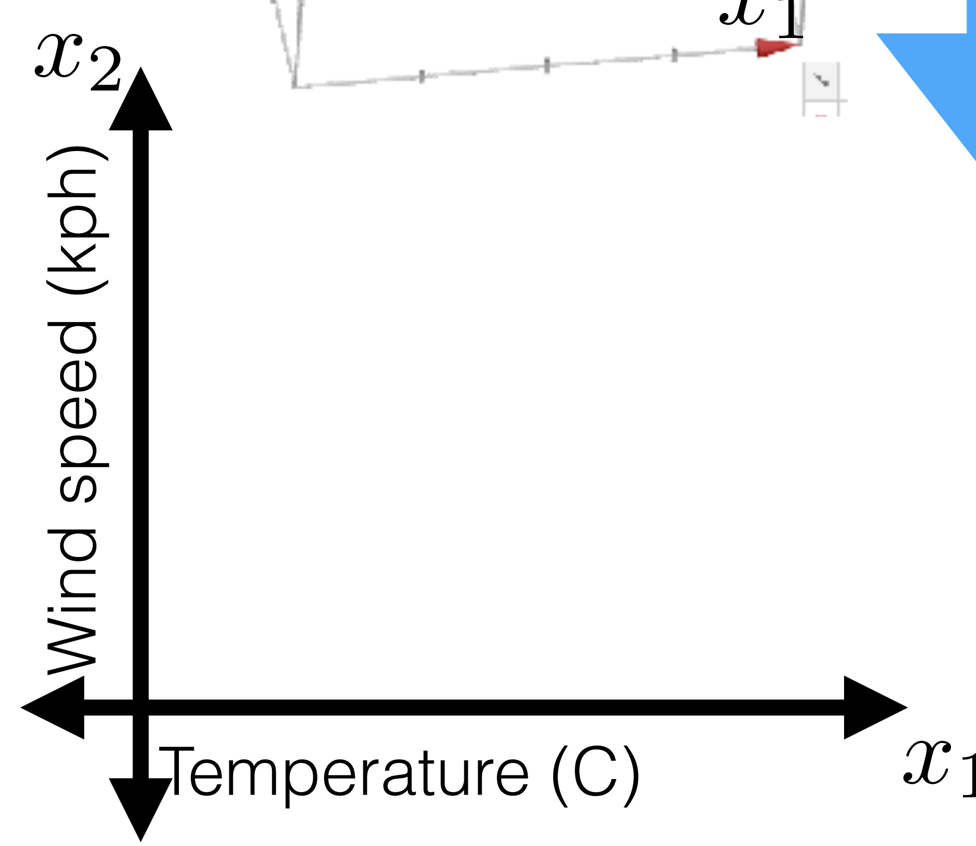


2 features:

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

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

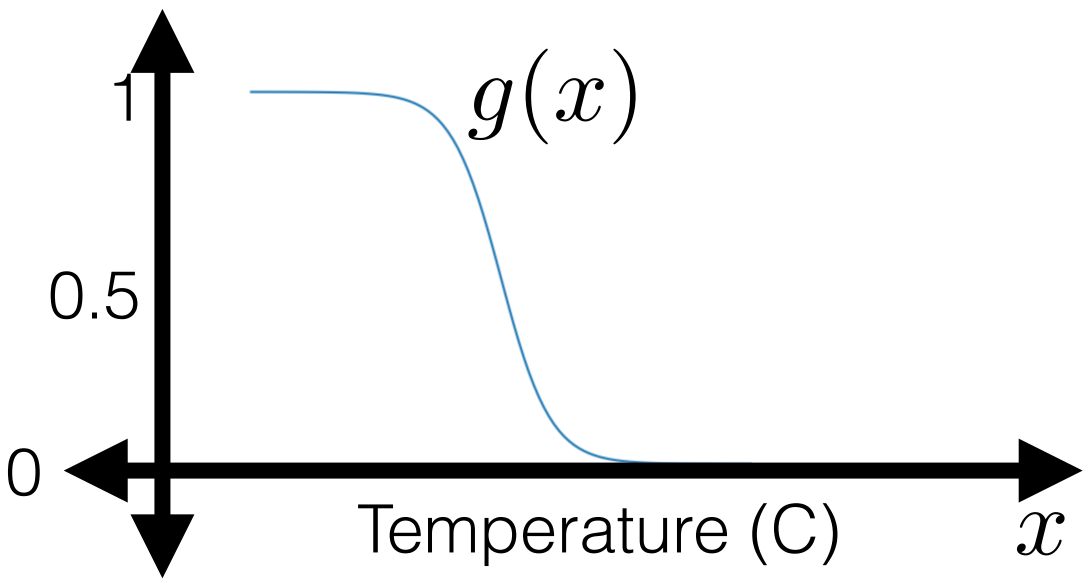


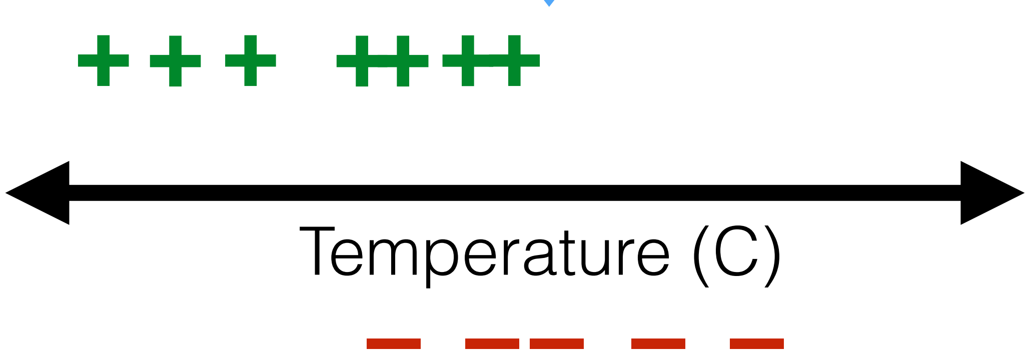


Capturing uncertainty

1 feature:

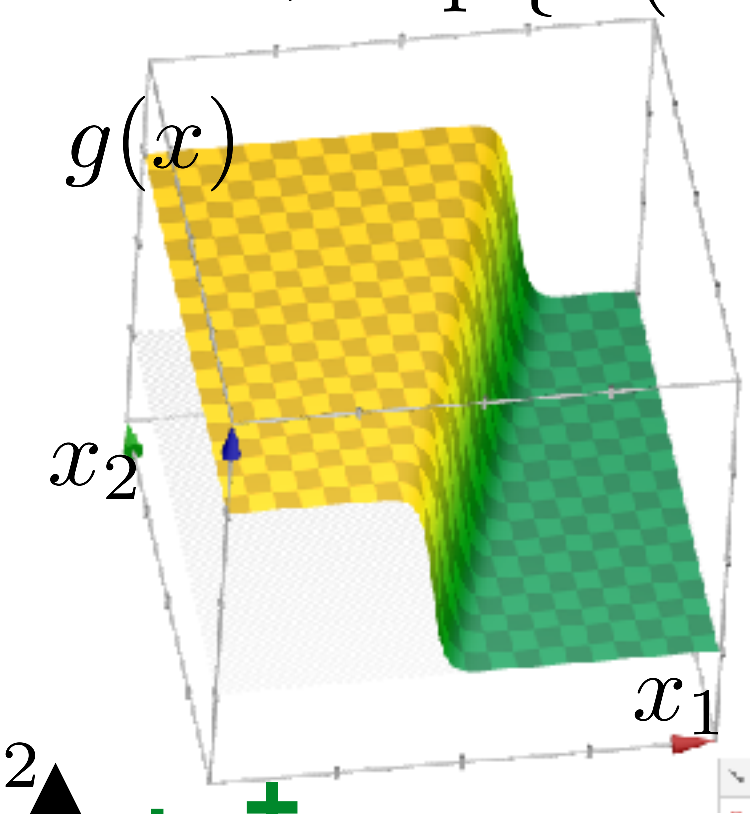
$$g(x) = \sigma(\theta x + \theta_0)$$
$$= \frac{1}{1 + \exp\{-(\theta x + \theta_0)\}}$$

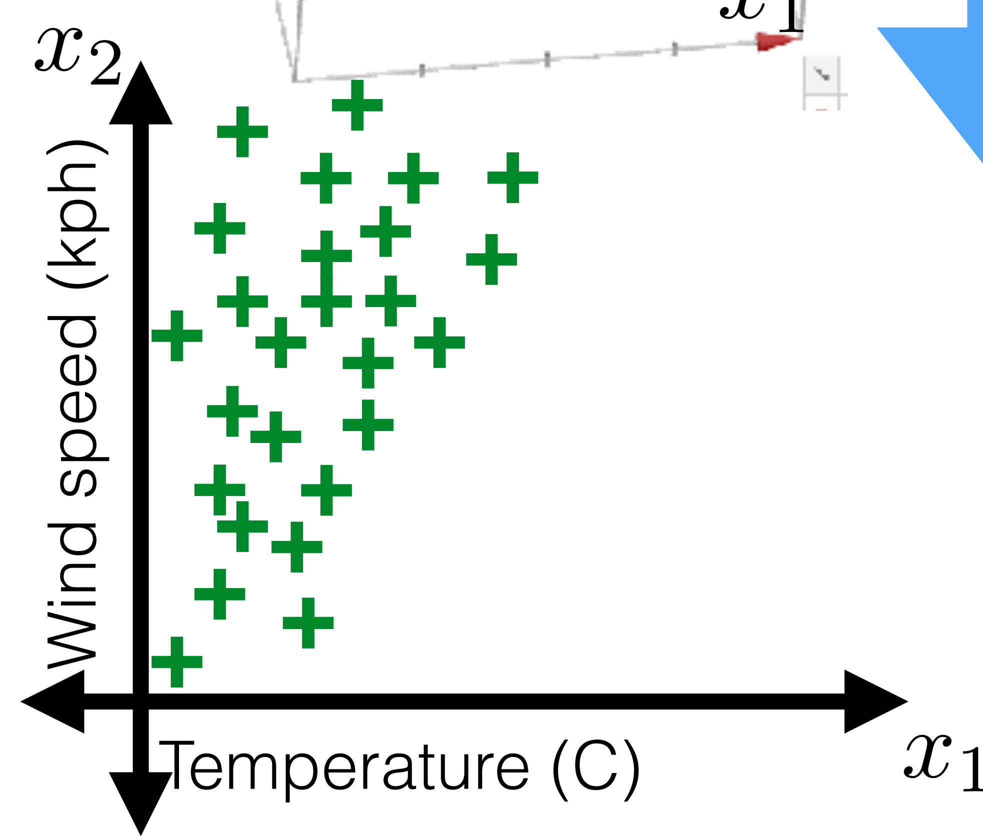




2 features:

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

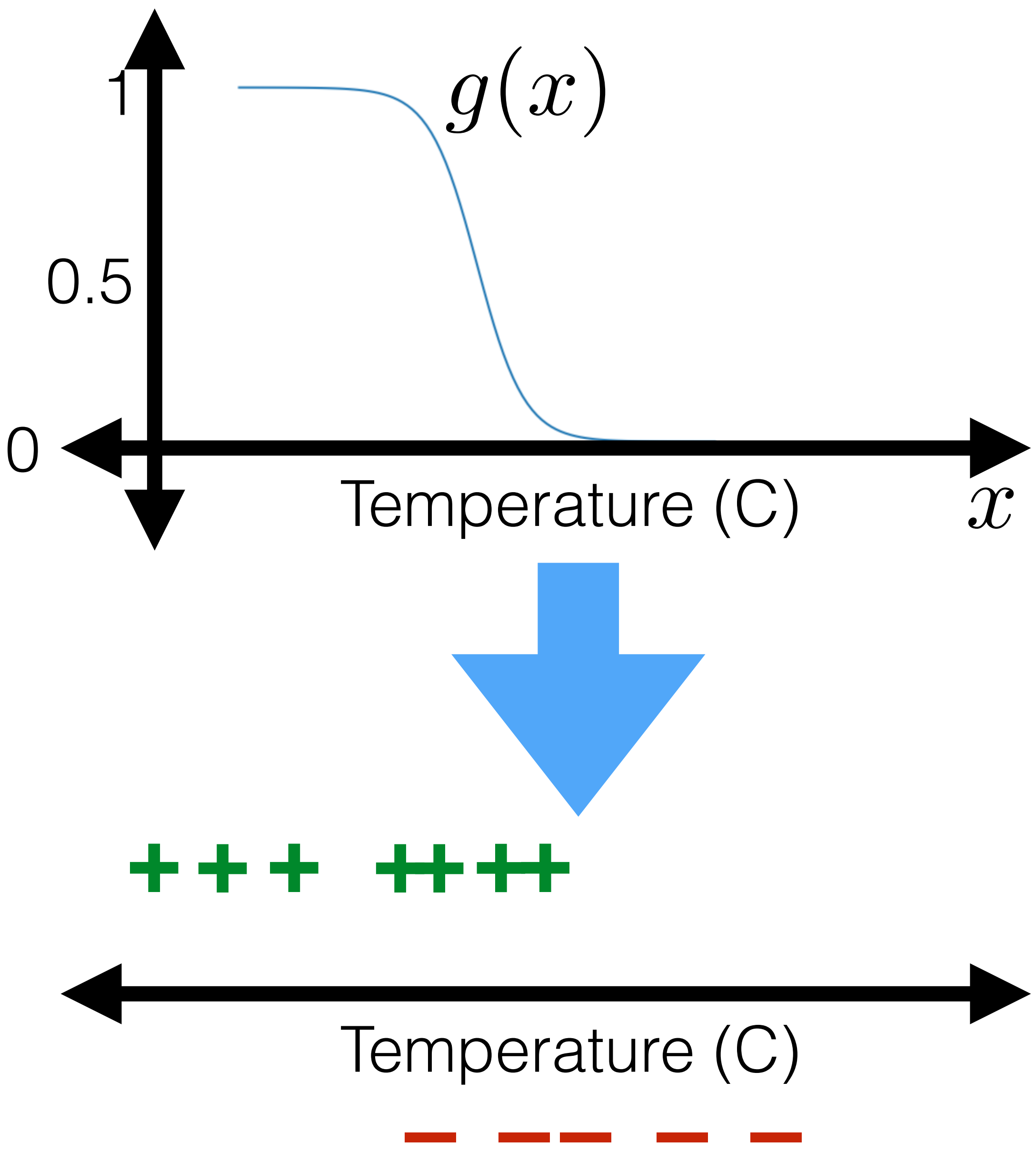




Capturing uncertainty

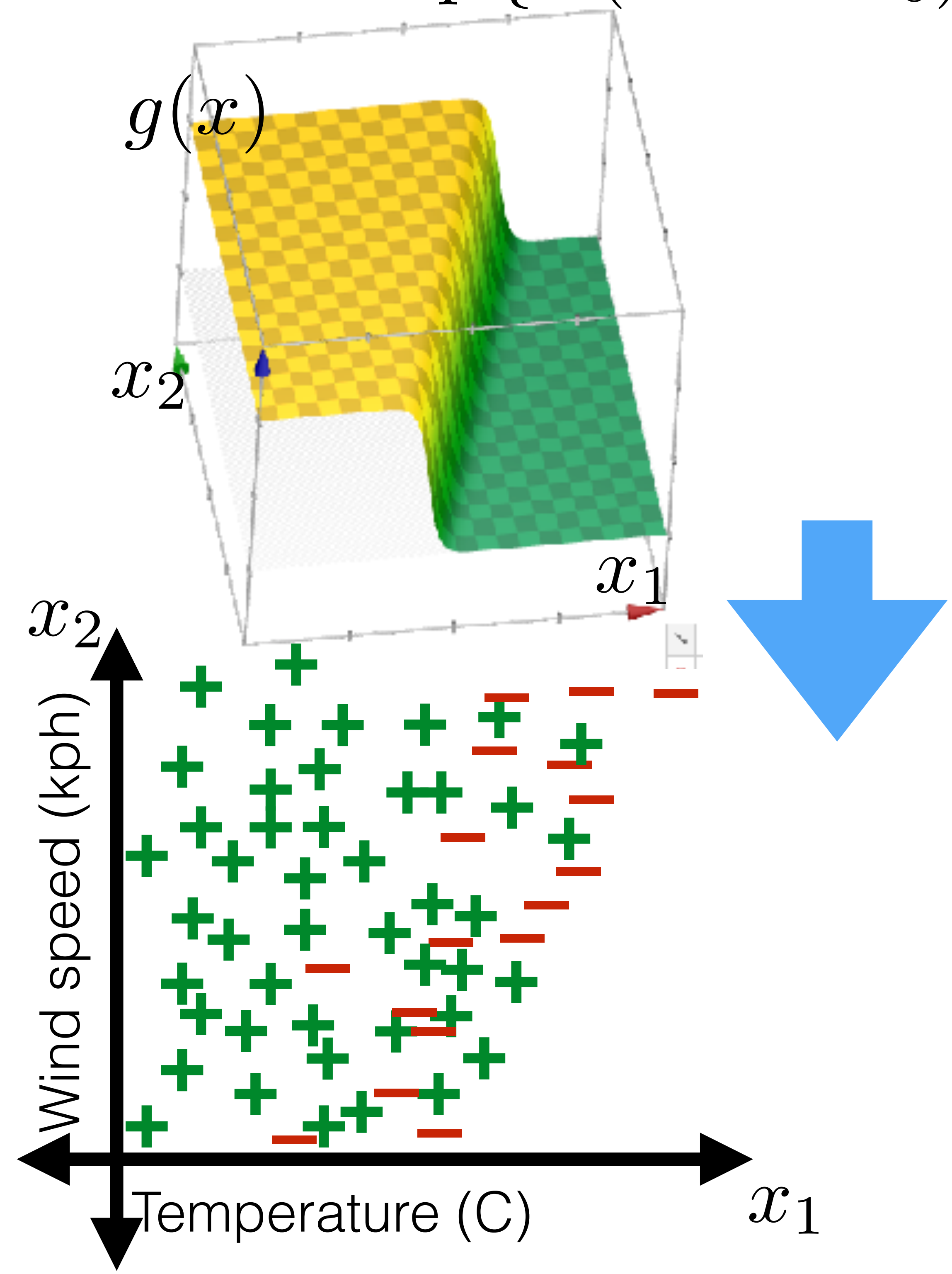
1 feature:

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



2 features:

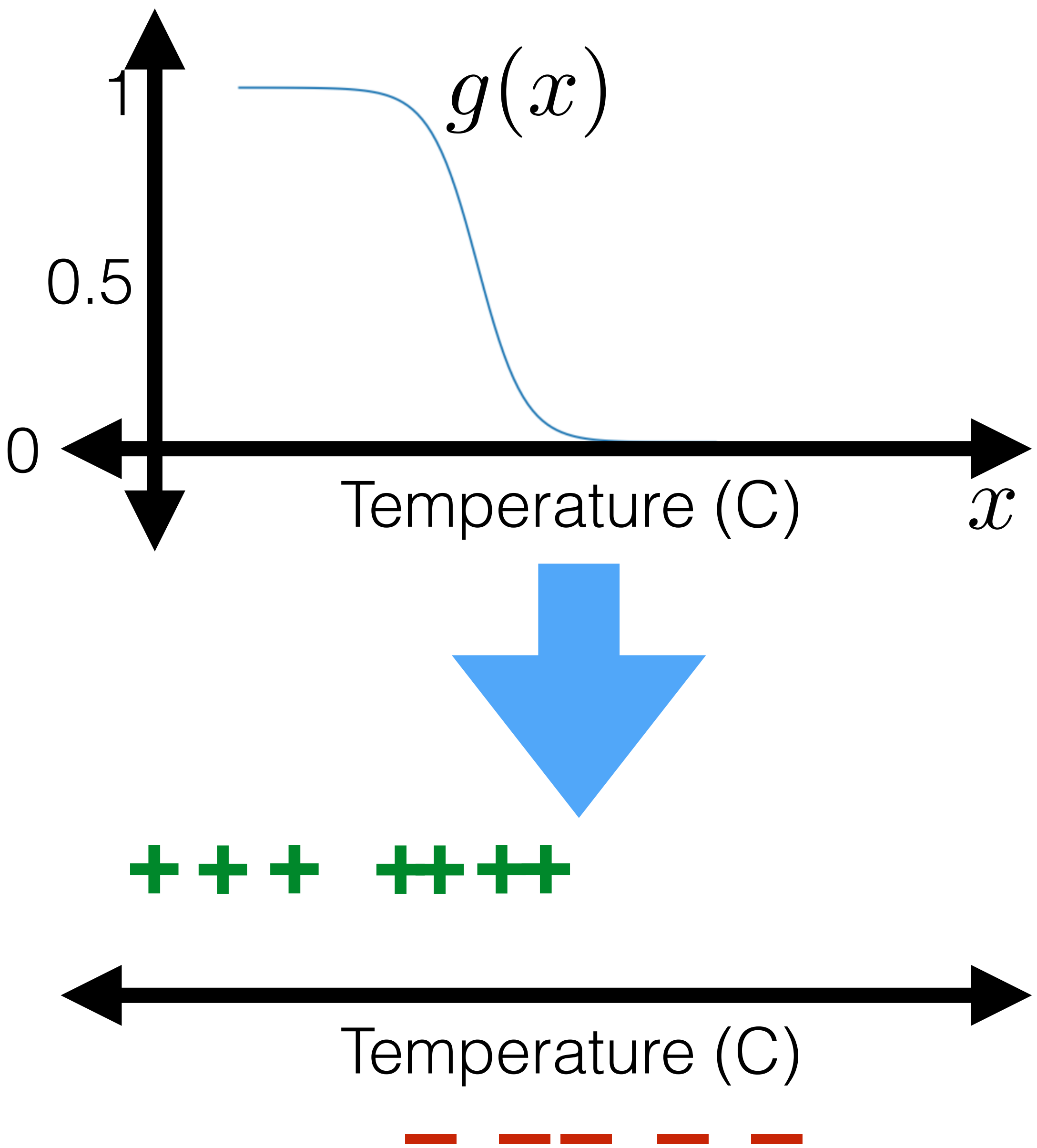
$$g(x) = \sigma(\theta^\top x + \theta_0)$$
$$= \frac{1}{1 + \exp\{-(\theta^\top x + \theta_0)\}}$$



Capturing uncertainty

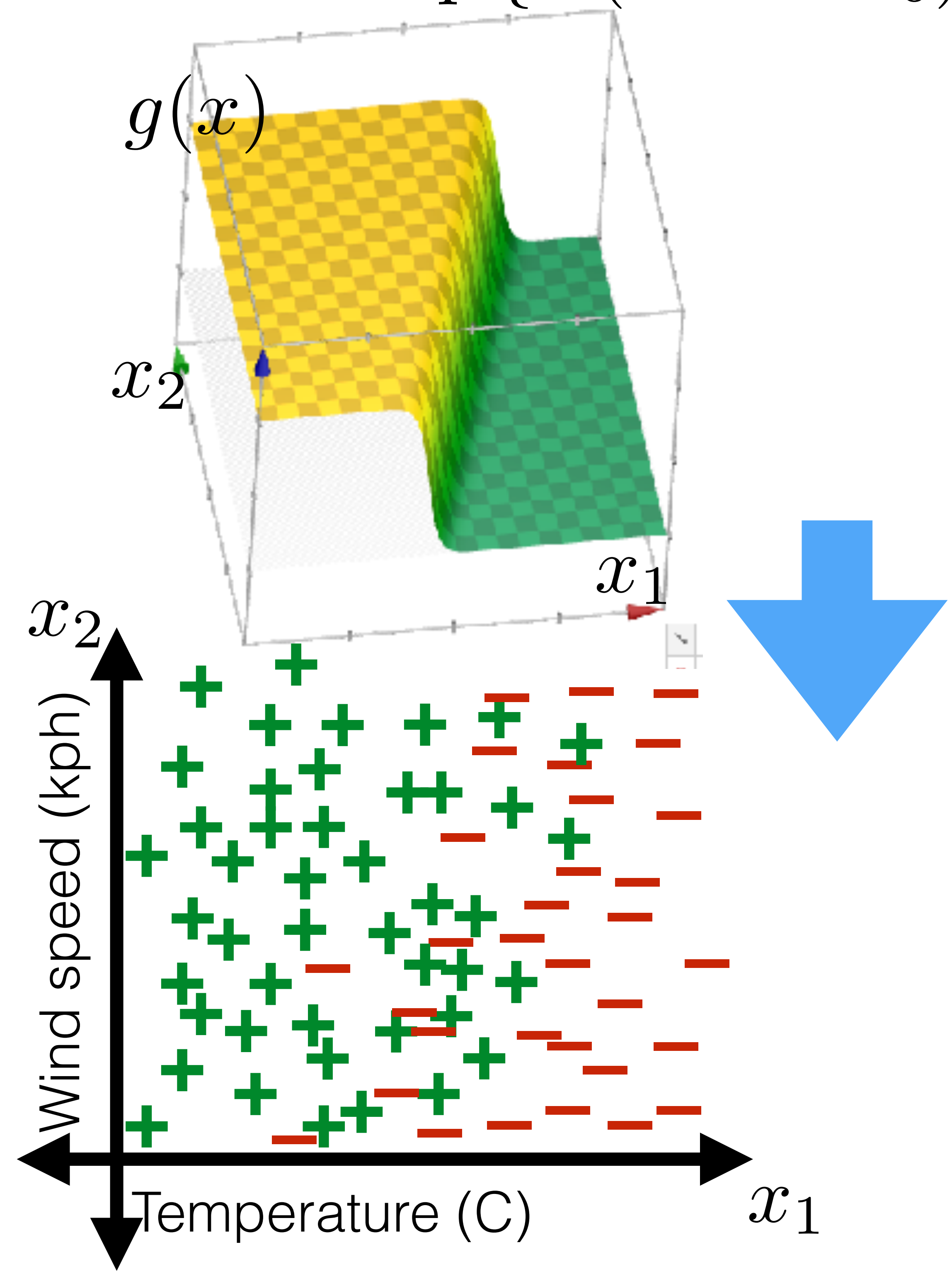
1 feature:

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



2 features:

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



Linear logistic classification

aka logistic regression

Linear logistic classification

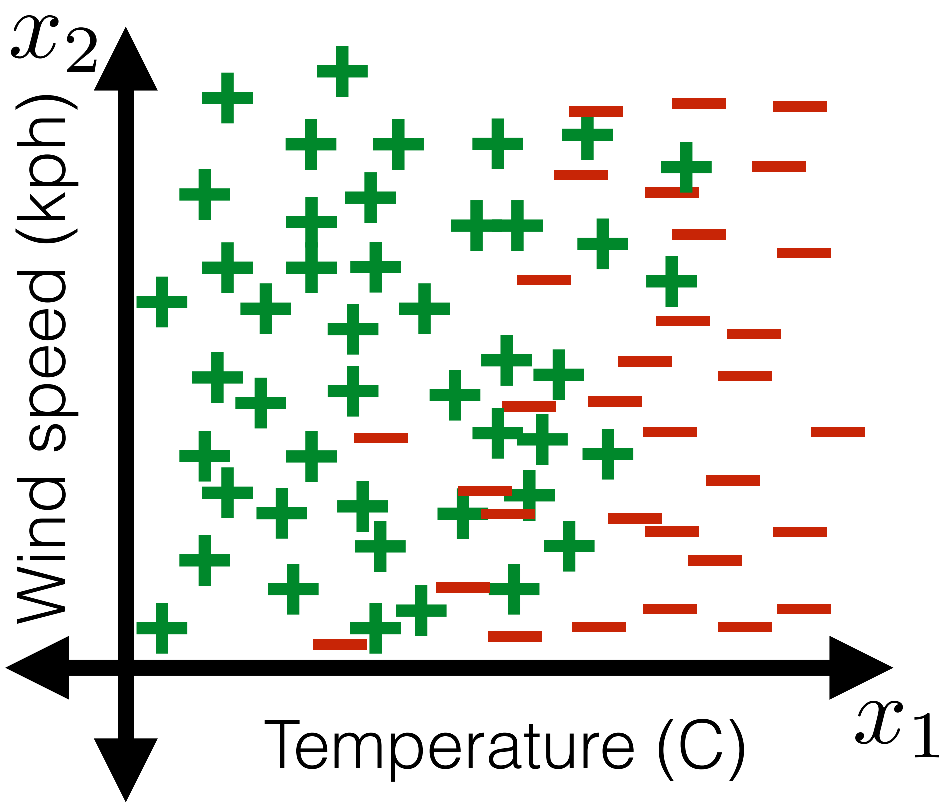
aka logistic regression

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

Linear logistic classification

aka logistic regression

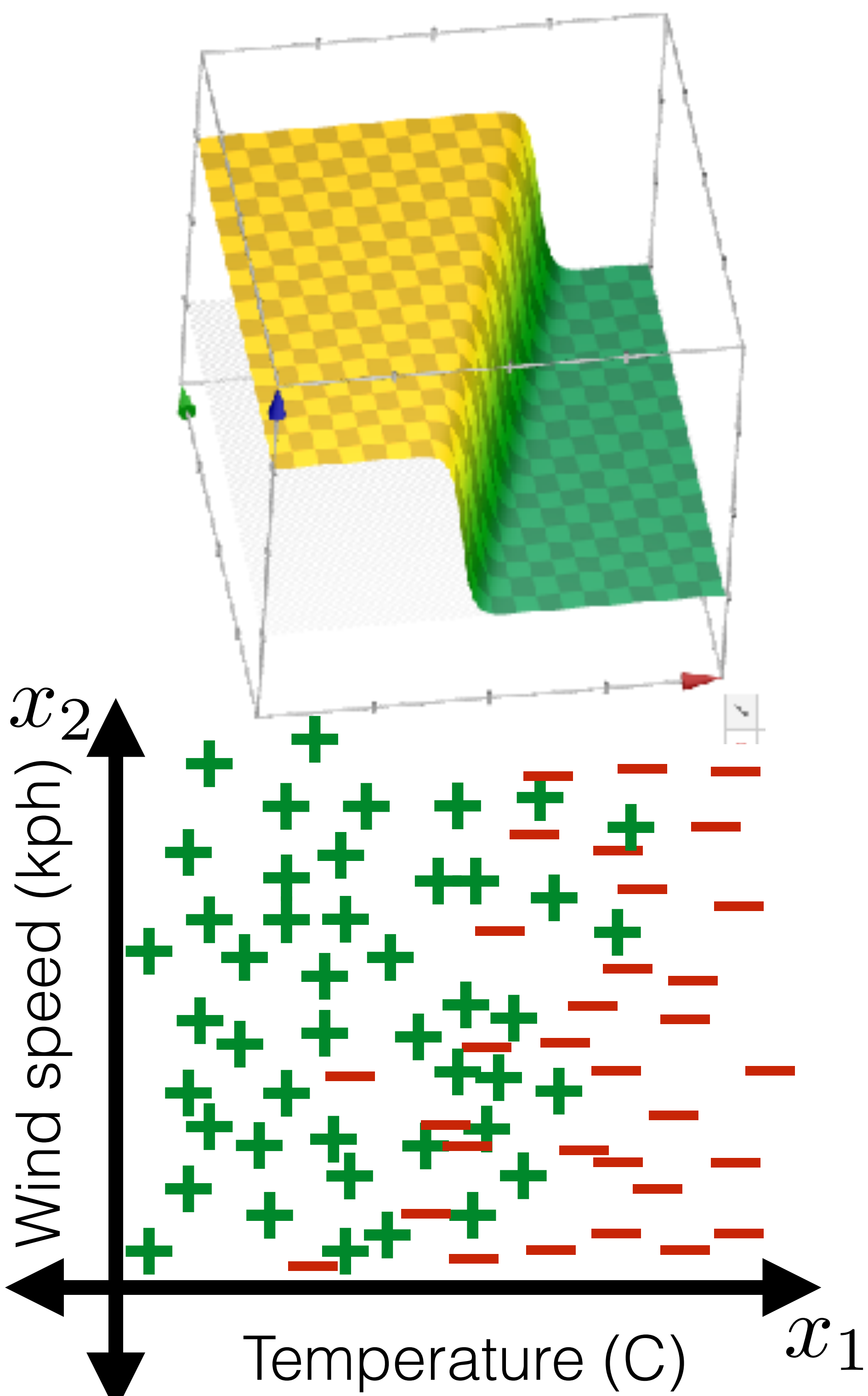
- How do we learn a classifier (i.e. learn θ, θ_0)?



Linear logistic classification

aka logistic regression

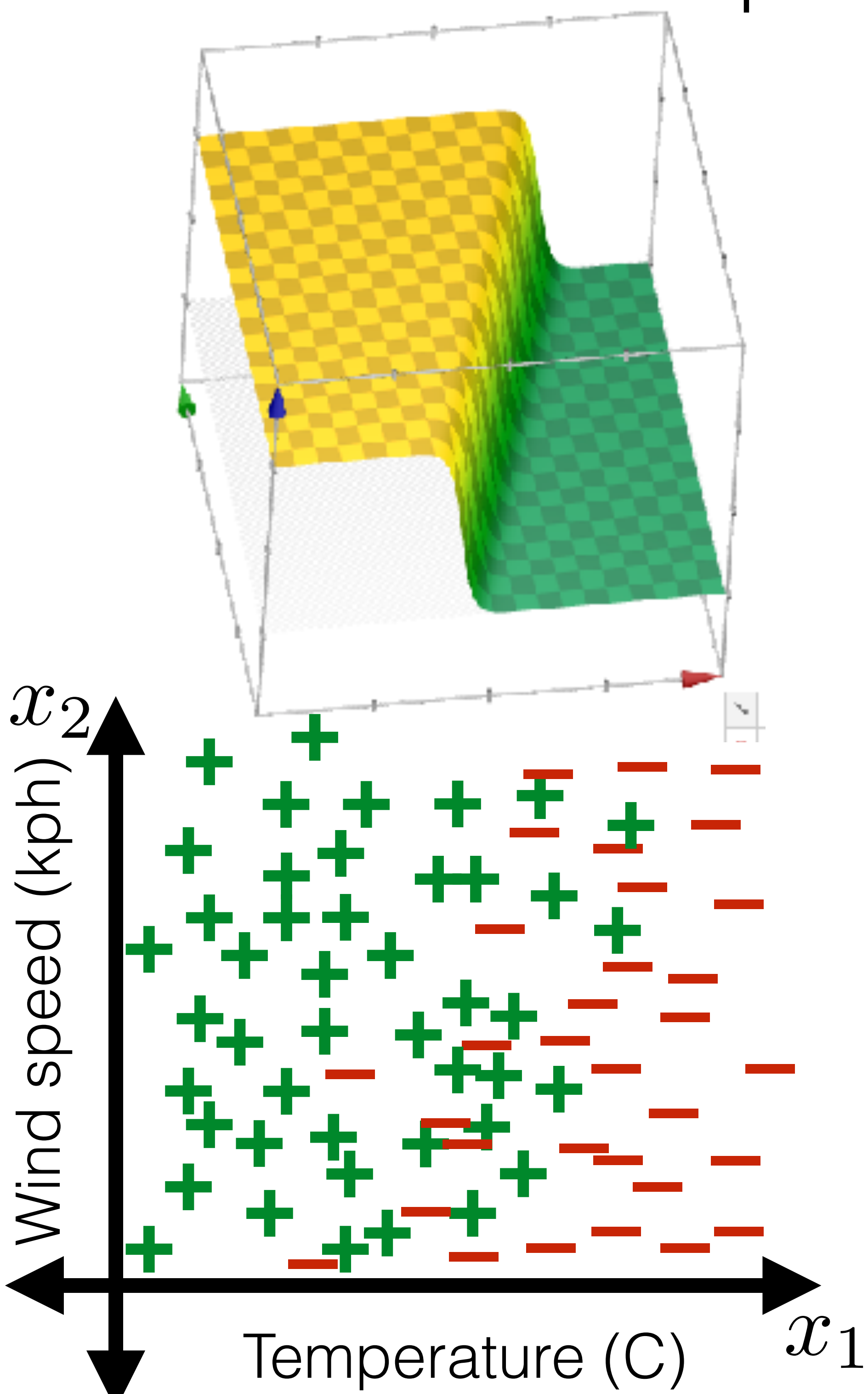
- How do we learn a classifier (i.e. learn θ, θ_0)?



Linear logistic classification

aka logistic regression

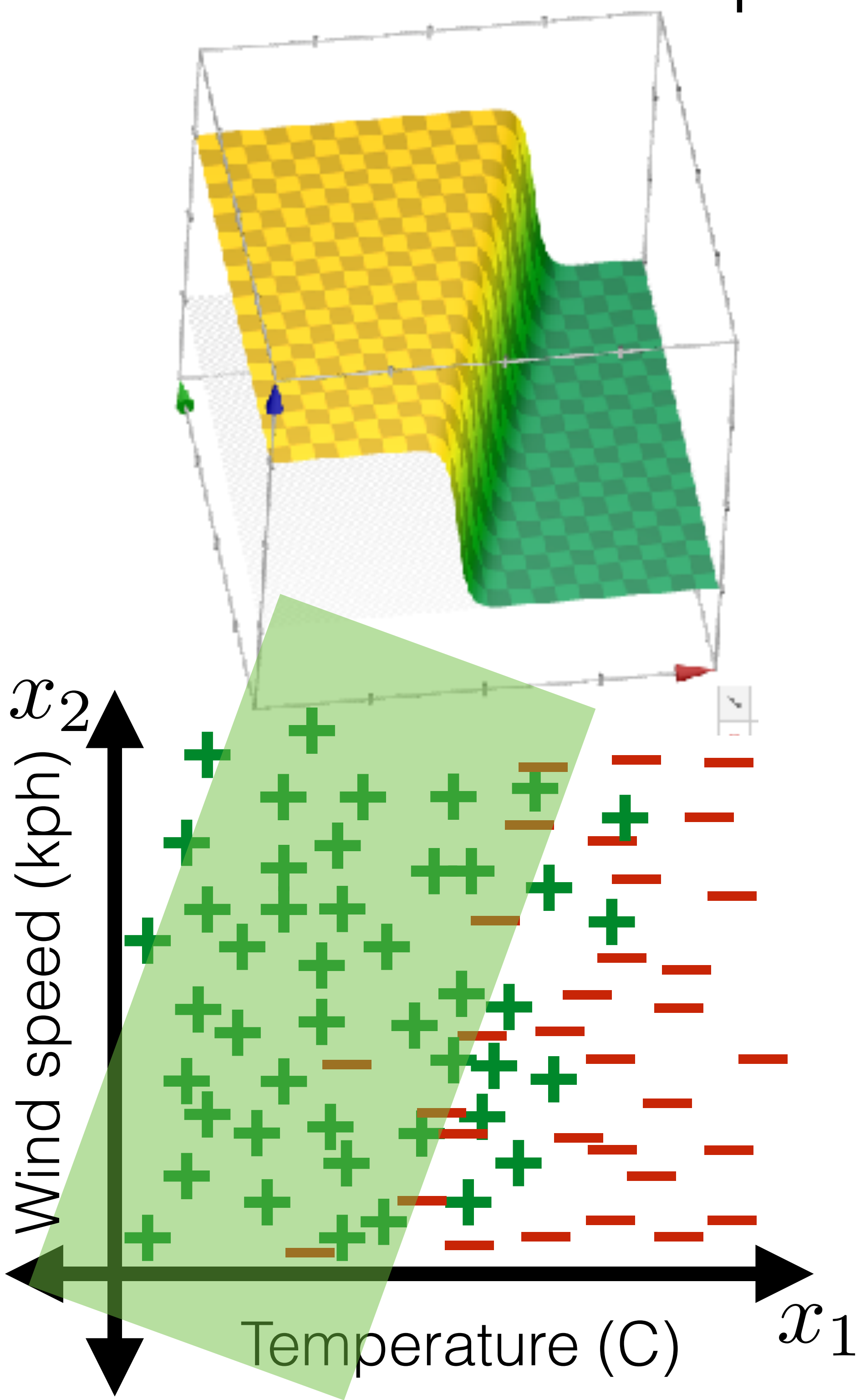
- How do we learn a classifier (i.e. learn θ, θ_0)?
- How do we make predictions?



Linear logistic classification

aka logistic regression

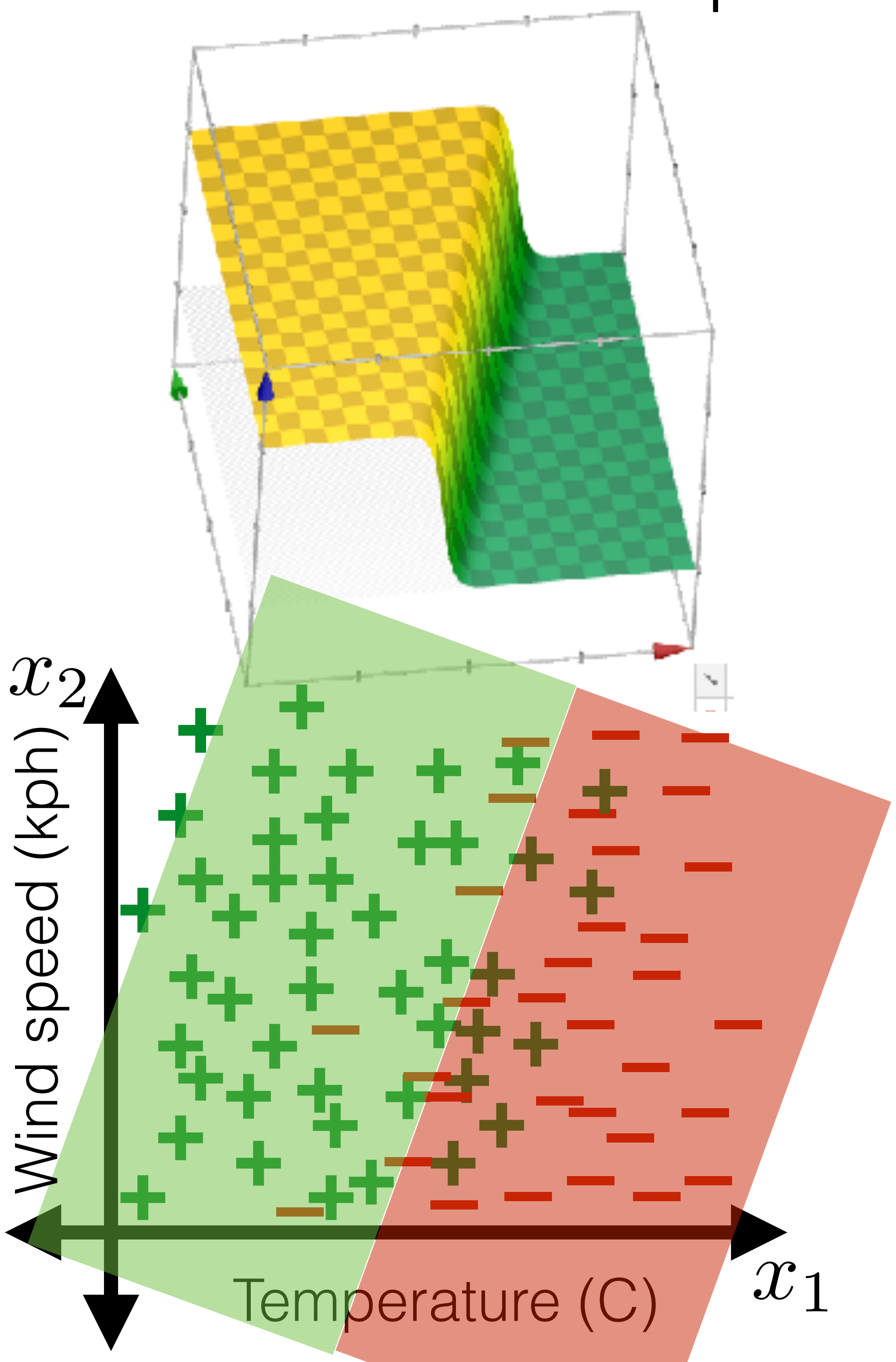
- How do we learn a classifier (i.e. learn θ, θ_0)?
- How do we make predictions?



Linear logistic classification

aka logistic regression

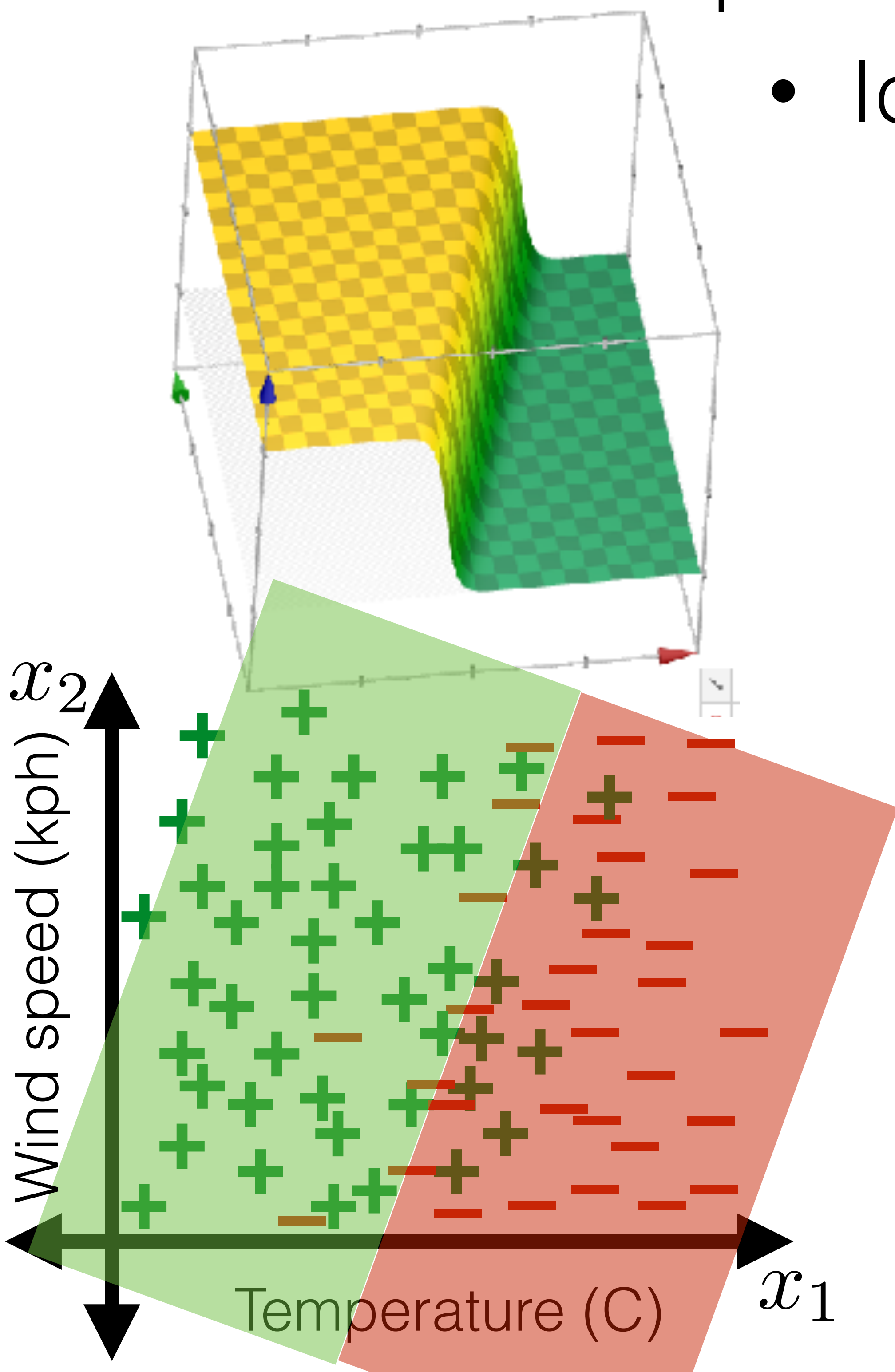
- How do we learn a classifier (i.e. learn θ, θ_0)?
- How do we make predictions?



Linear logistic classification

aka logistic regression

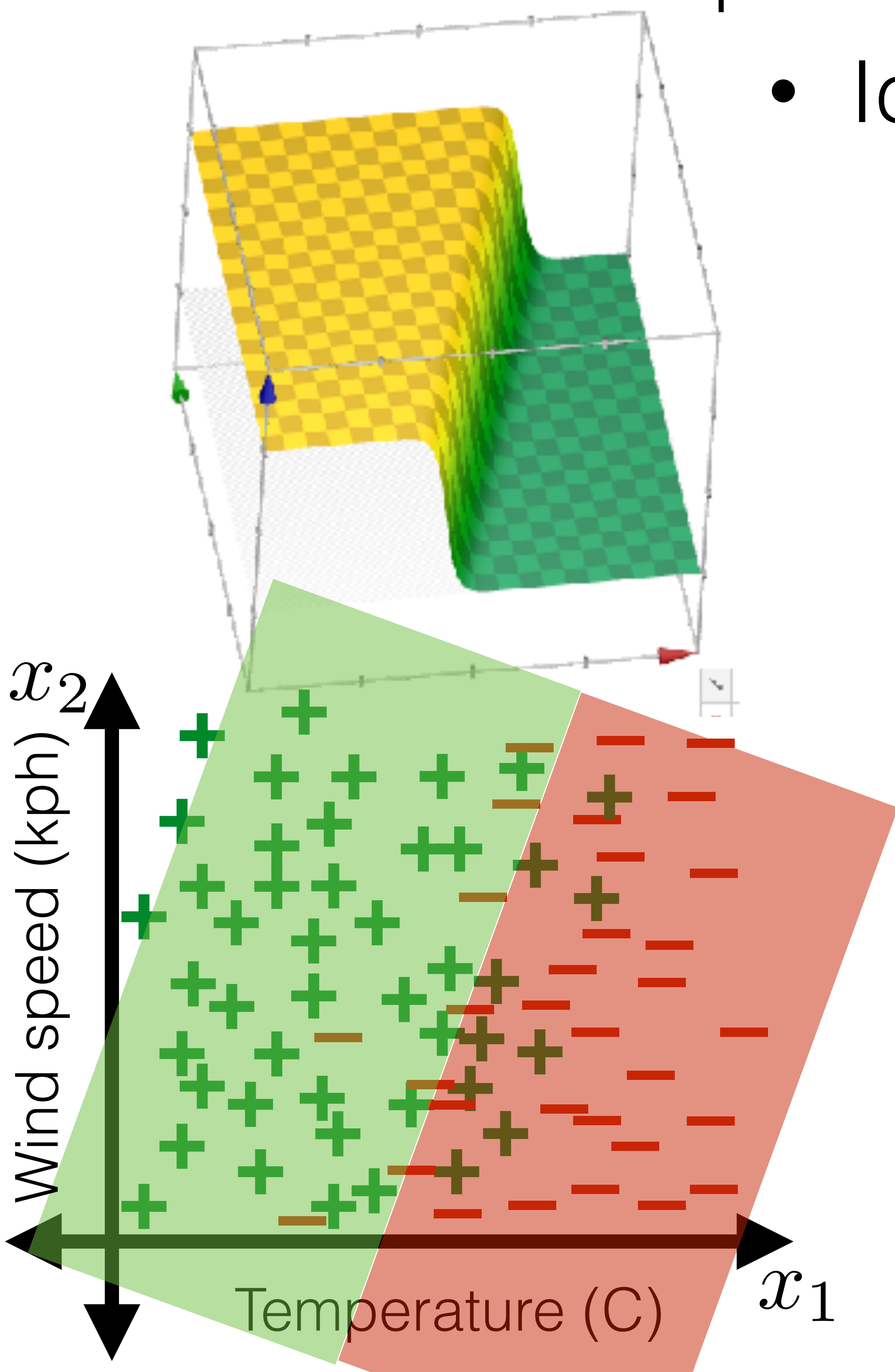
- How do we learn a classifier (i.e. learn θ, θ_0)?
- How do we make predictions?
  - Idea: predict +1 if:



Linear logistic classification

aka logistic regression

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

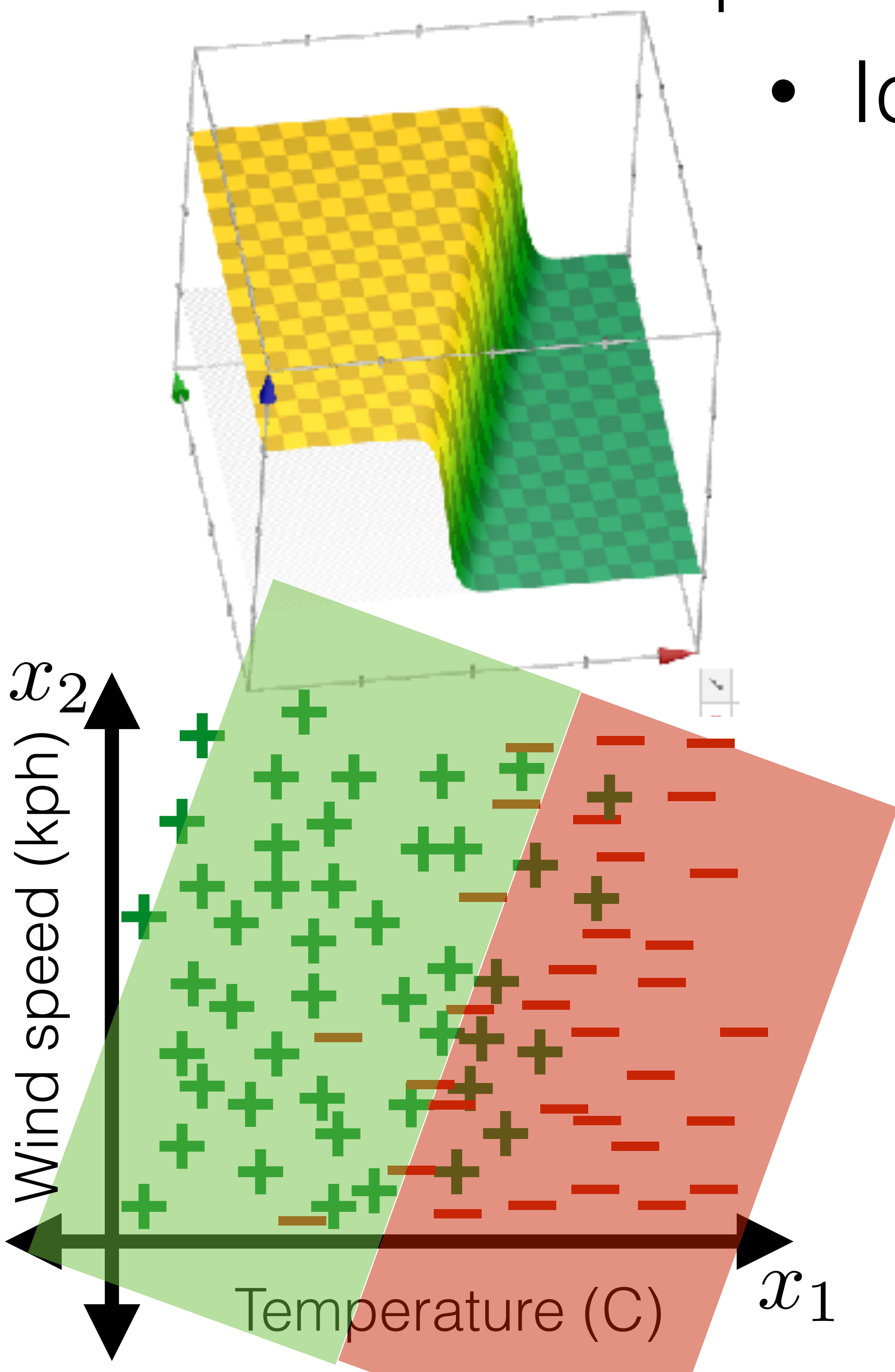


- Idea: predict +1 if: probability > 0.5

Linear logistic classification

aka logistic regression

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



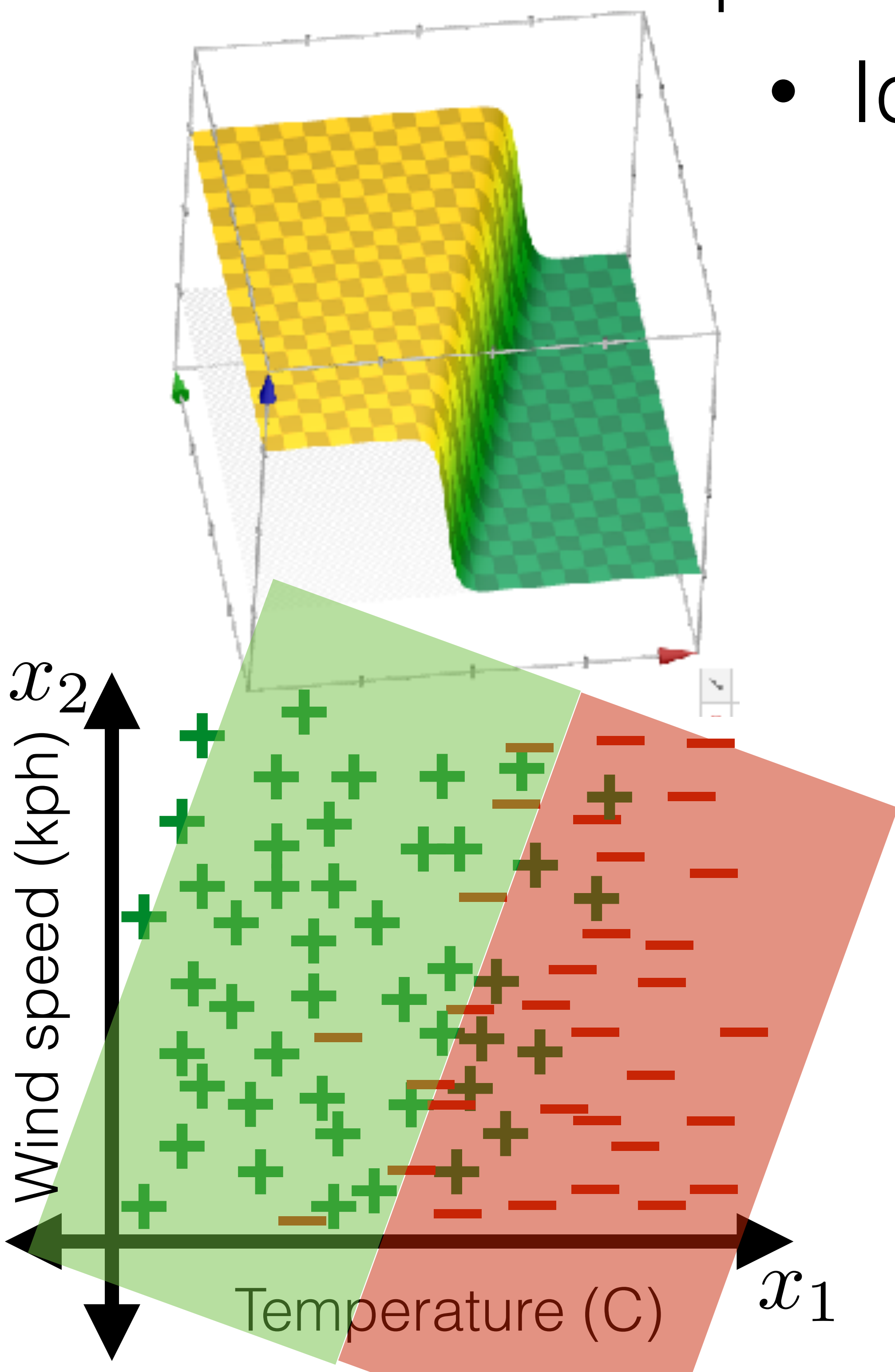
- Idea: predict +1 if: probability > 0.5

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

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

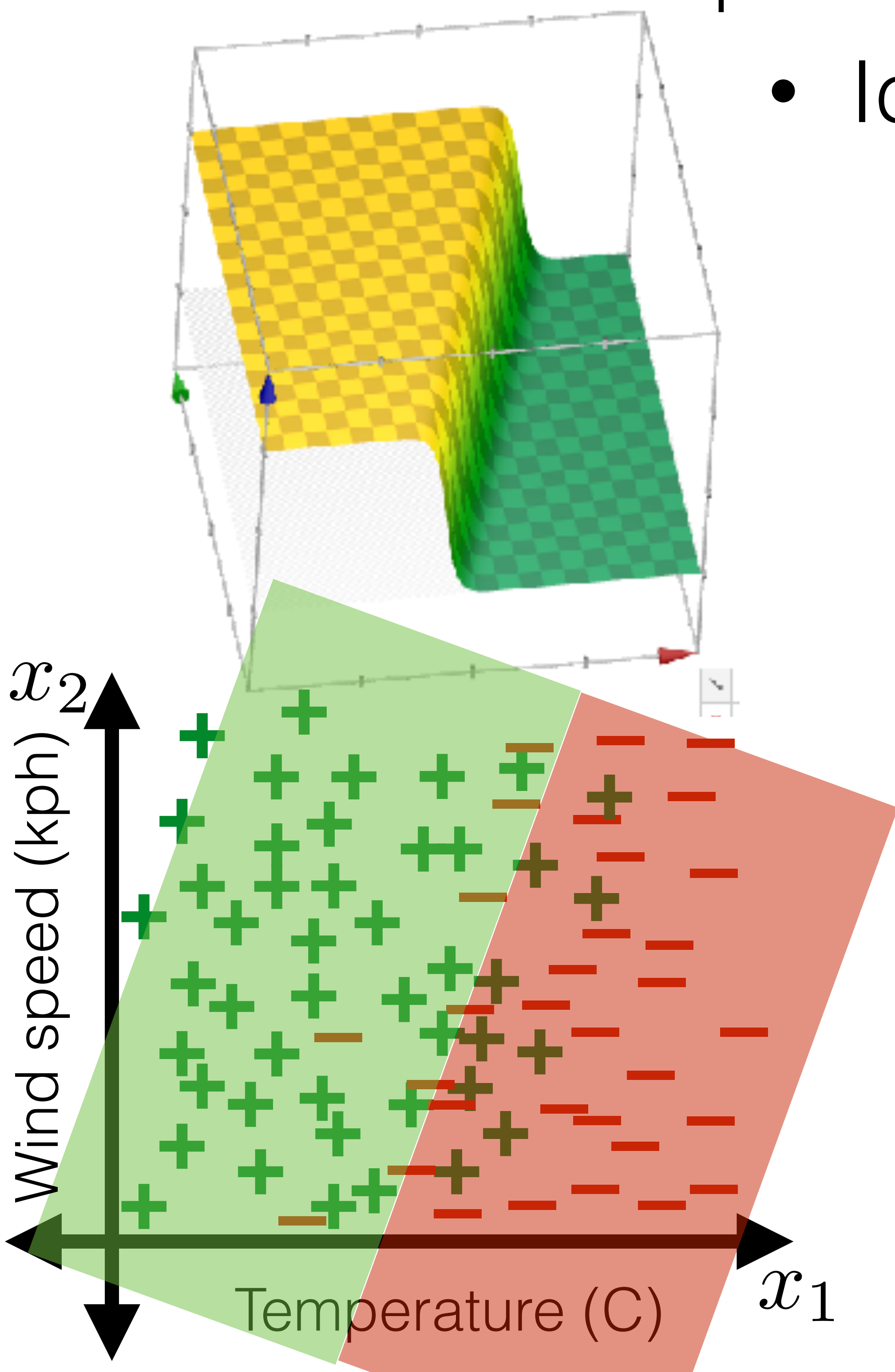
$$\sigma(\theta^\top x + \theta_0) > 0.5$$

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

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

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

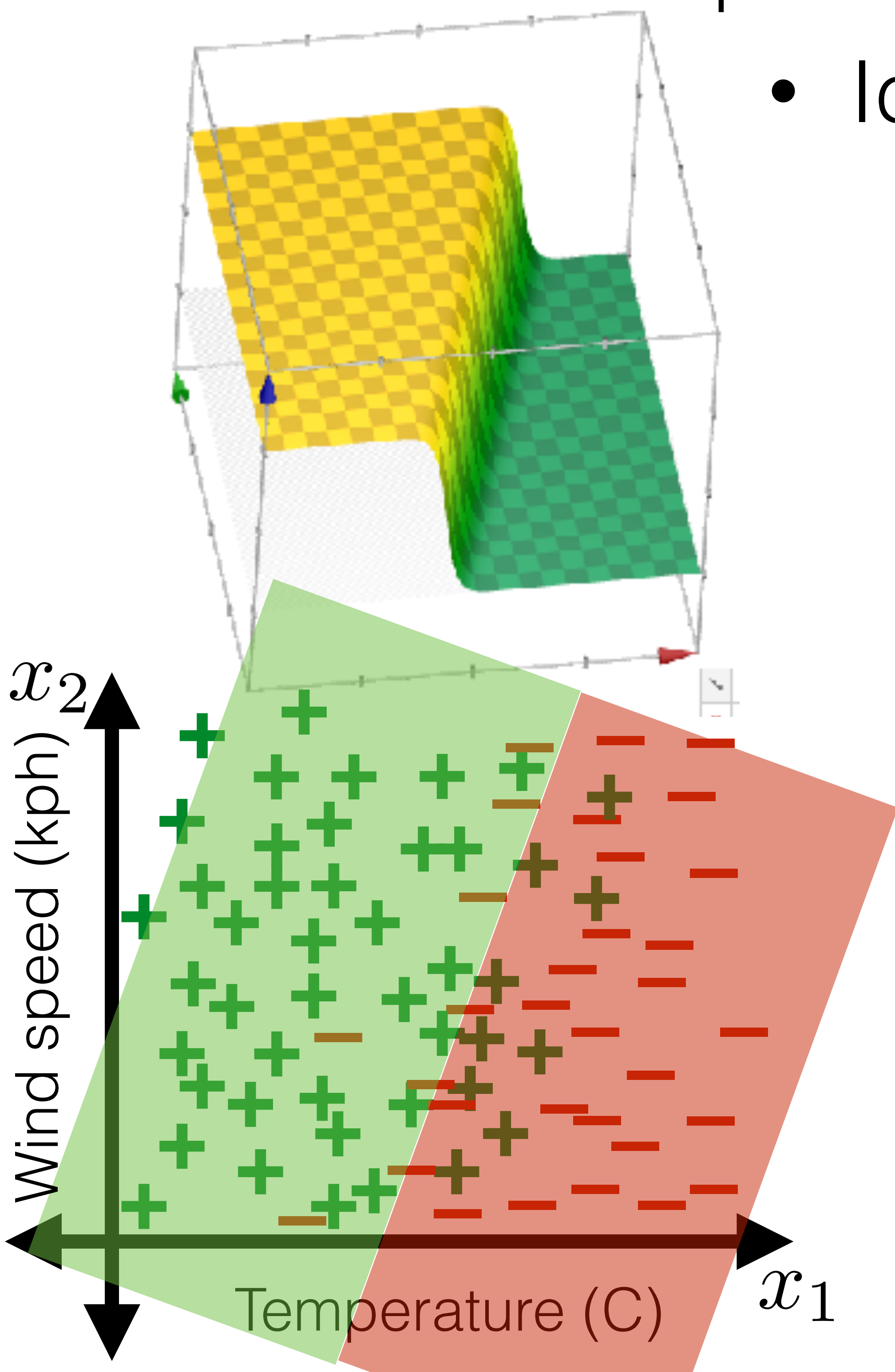
$$\frac{1}{1 + \exp\{-(\theta^\top x + \theta_0)\}} > 0.5$$

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

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

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

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

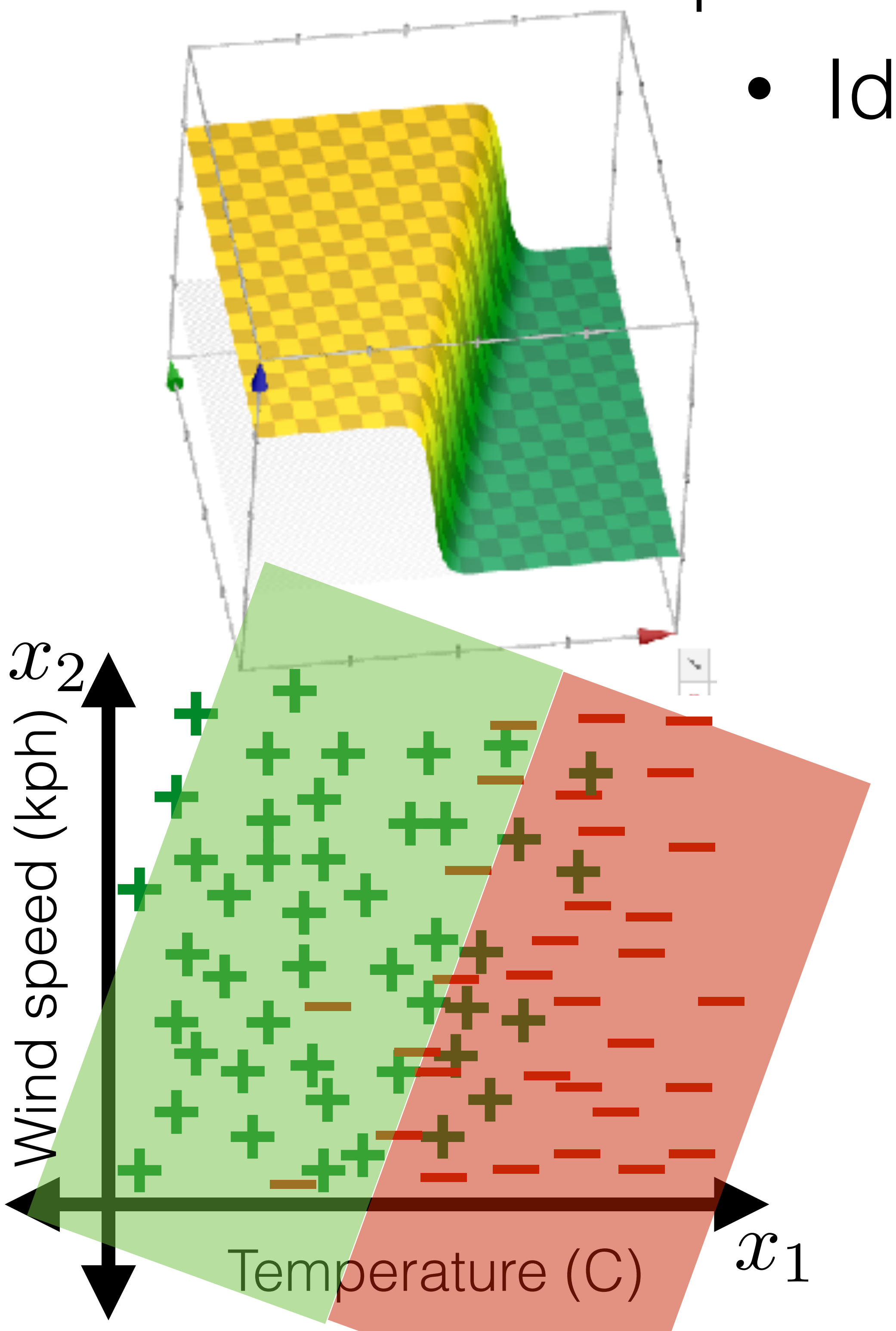
$$\exp\{-(\theta^\top x + \theta_0)\} < 1$$

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

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

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

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

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

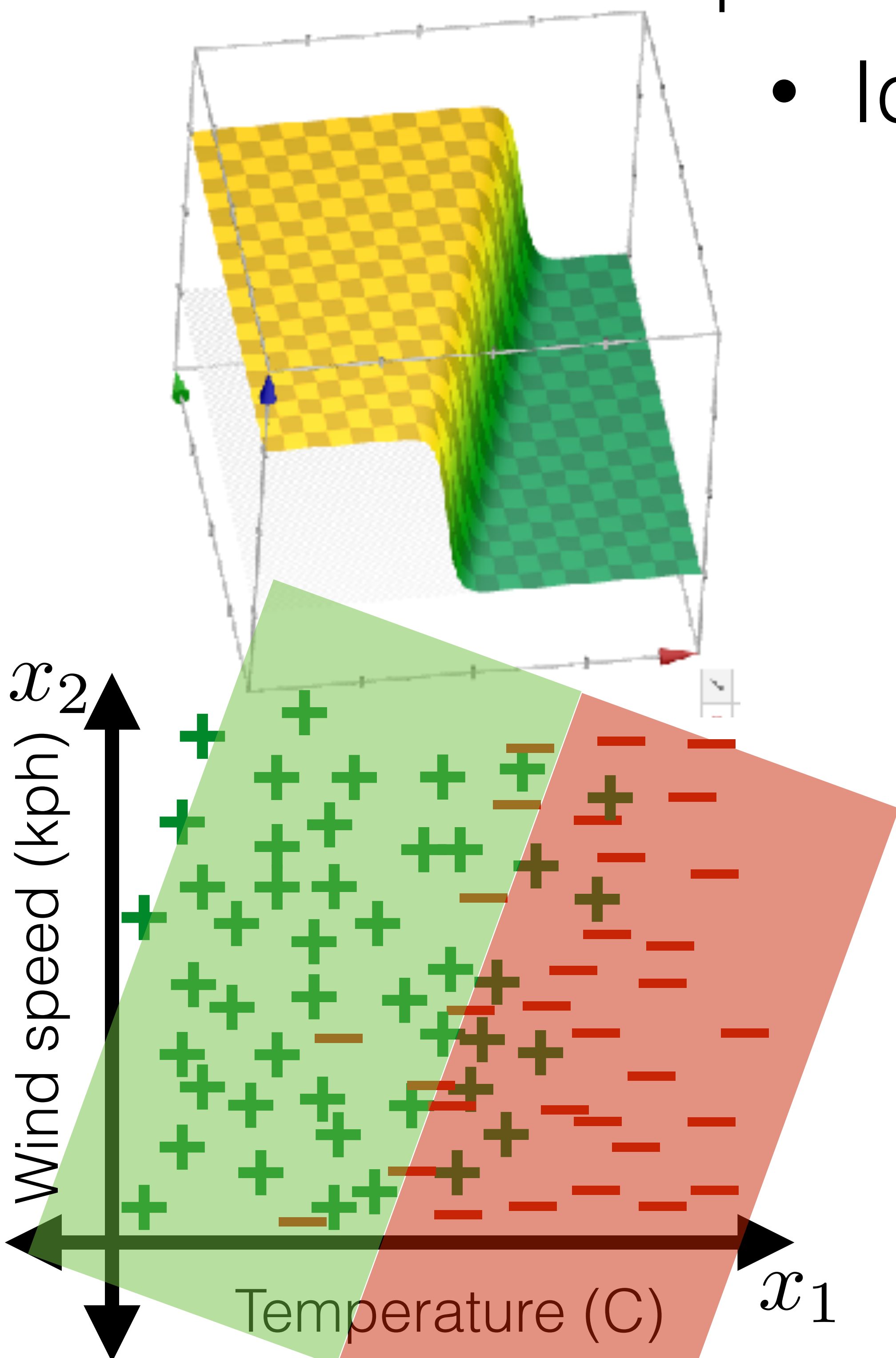
$$\theta^\top x + \theta_0 > 0$$

- 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

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

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

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

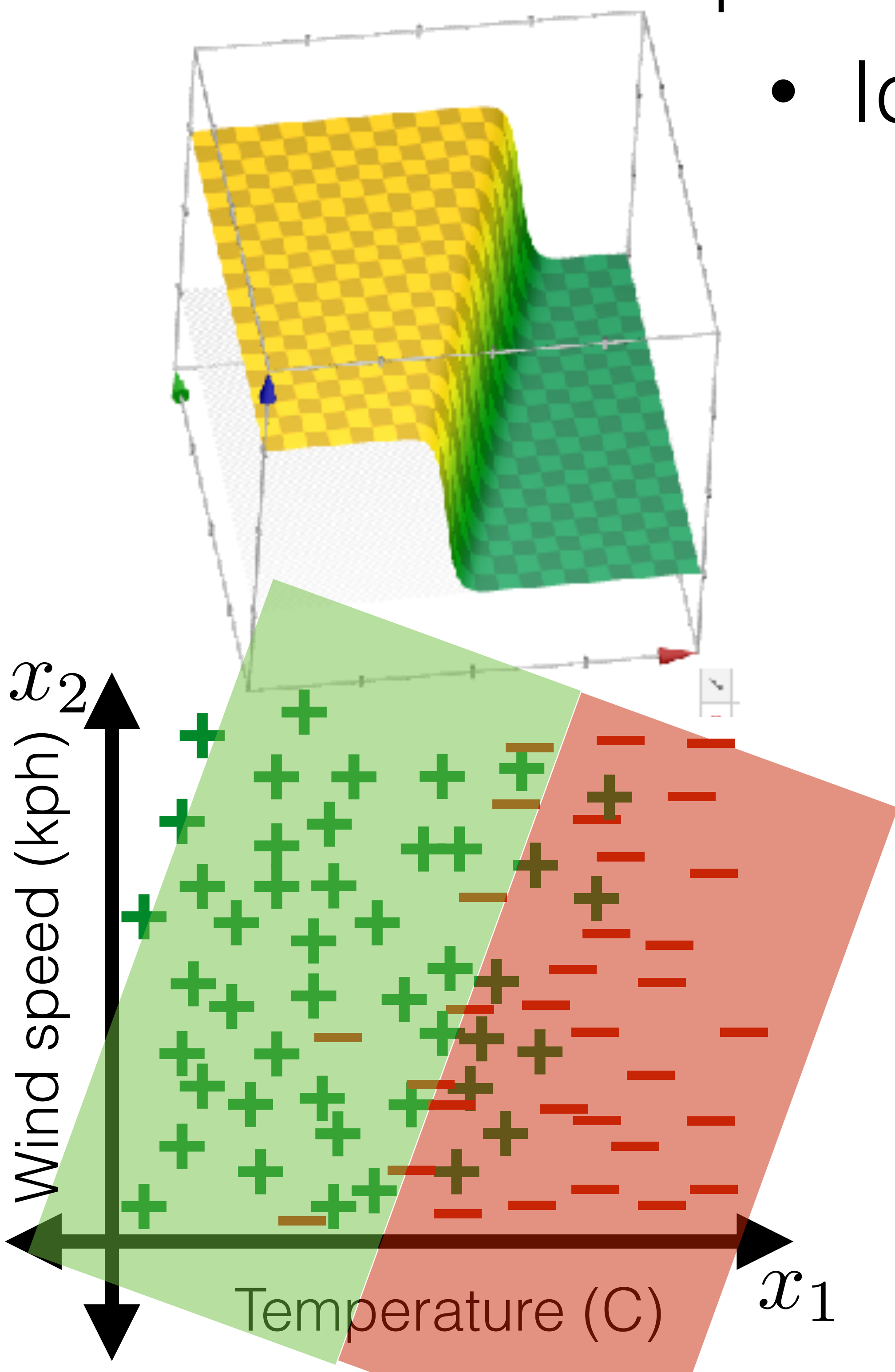
$$\theta^\top x + \theta_0 > 0$$

- 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

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

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

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

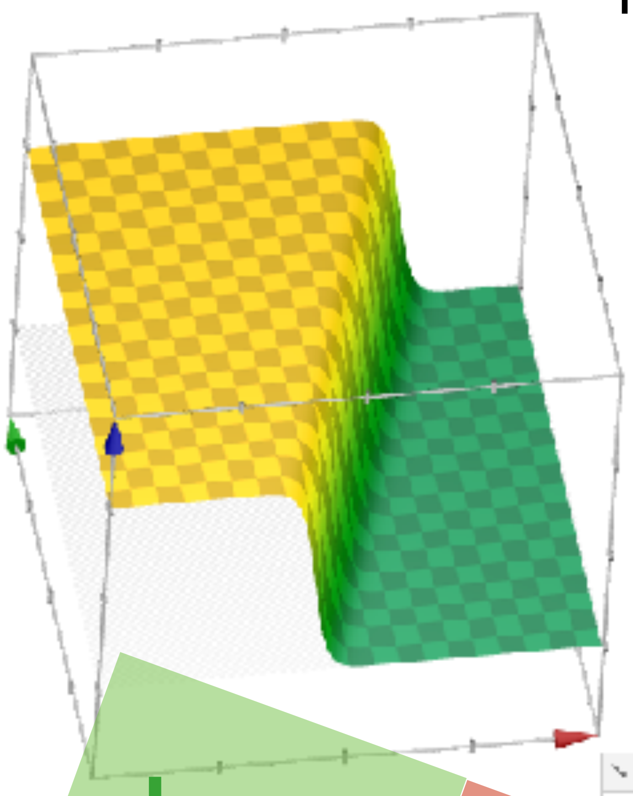
$$\theta^\top x + \theta_0 > 0$$

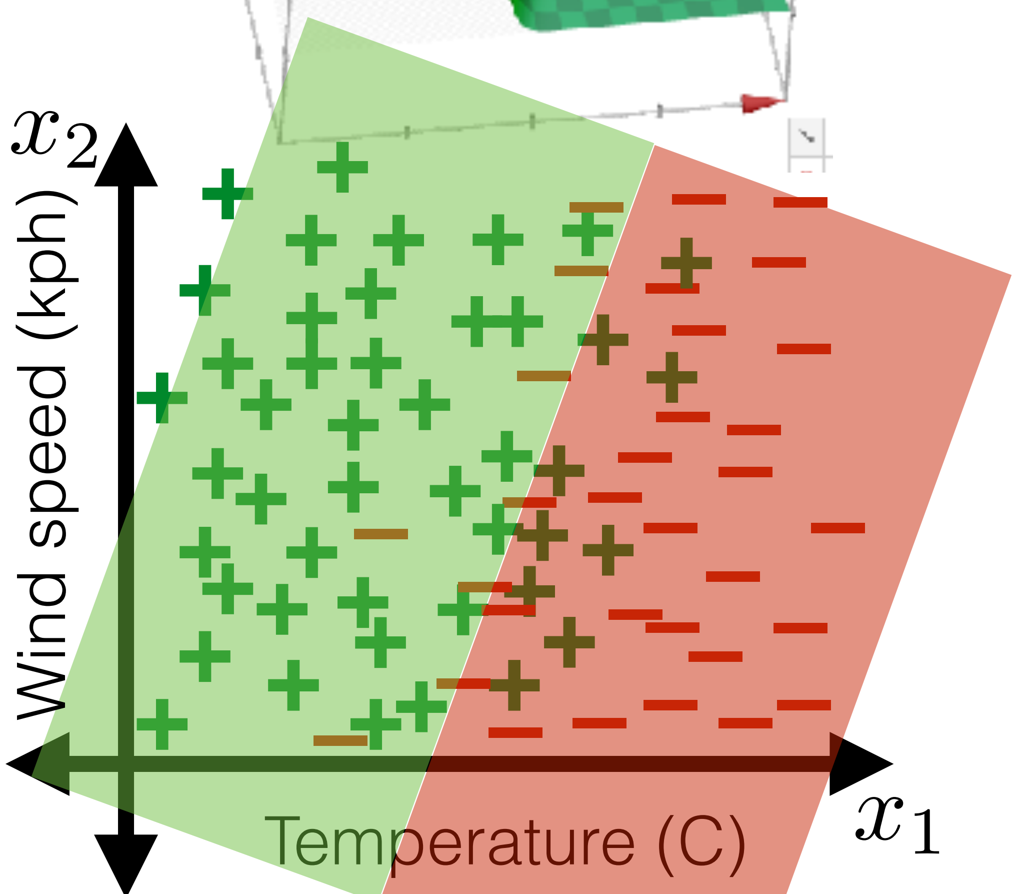
- Same hypothesis class as before! But we will get:
  - Uncertainties

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

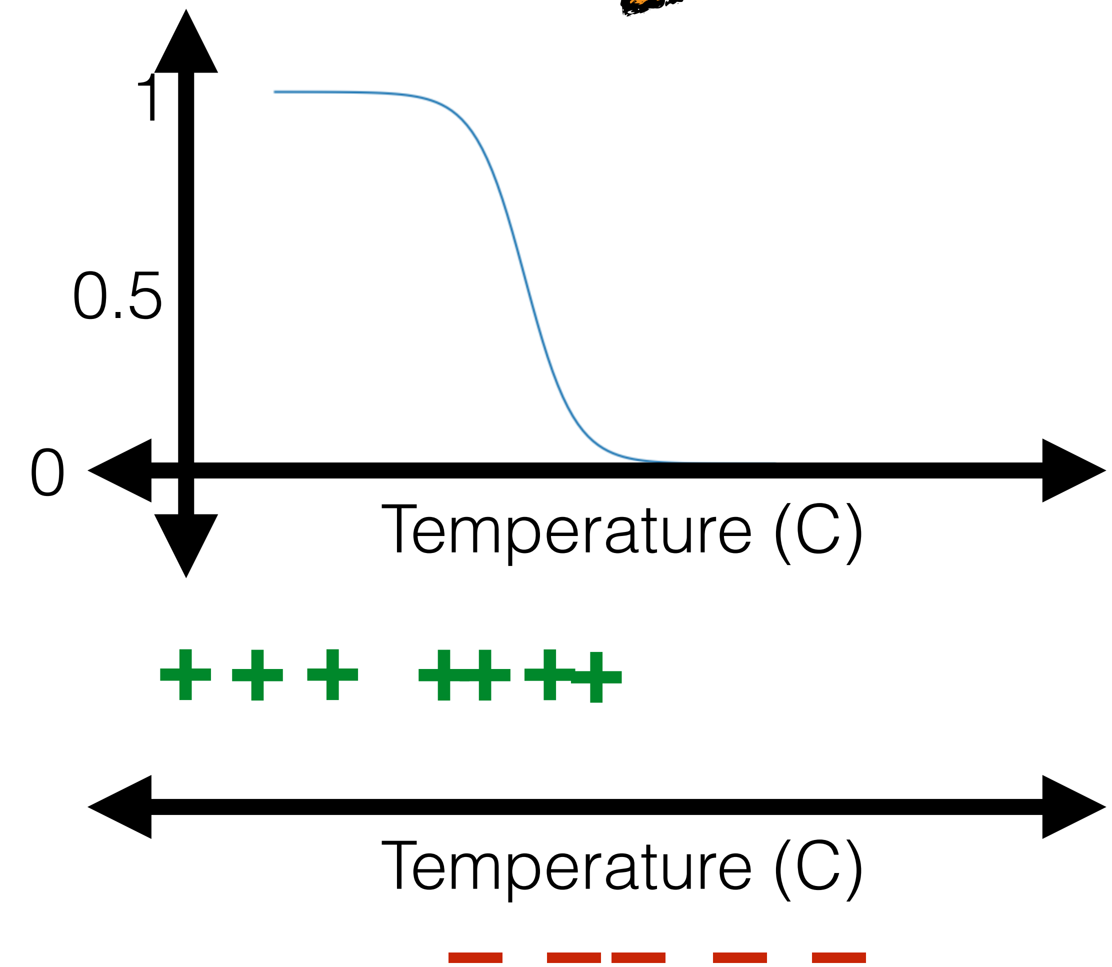
aka logistic regression

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

Linear logistic classification

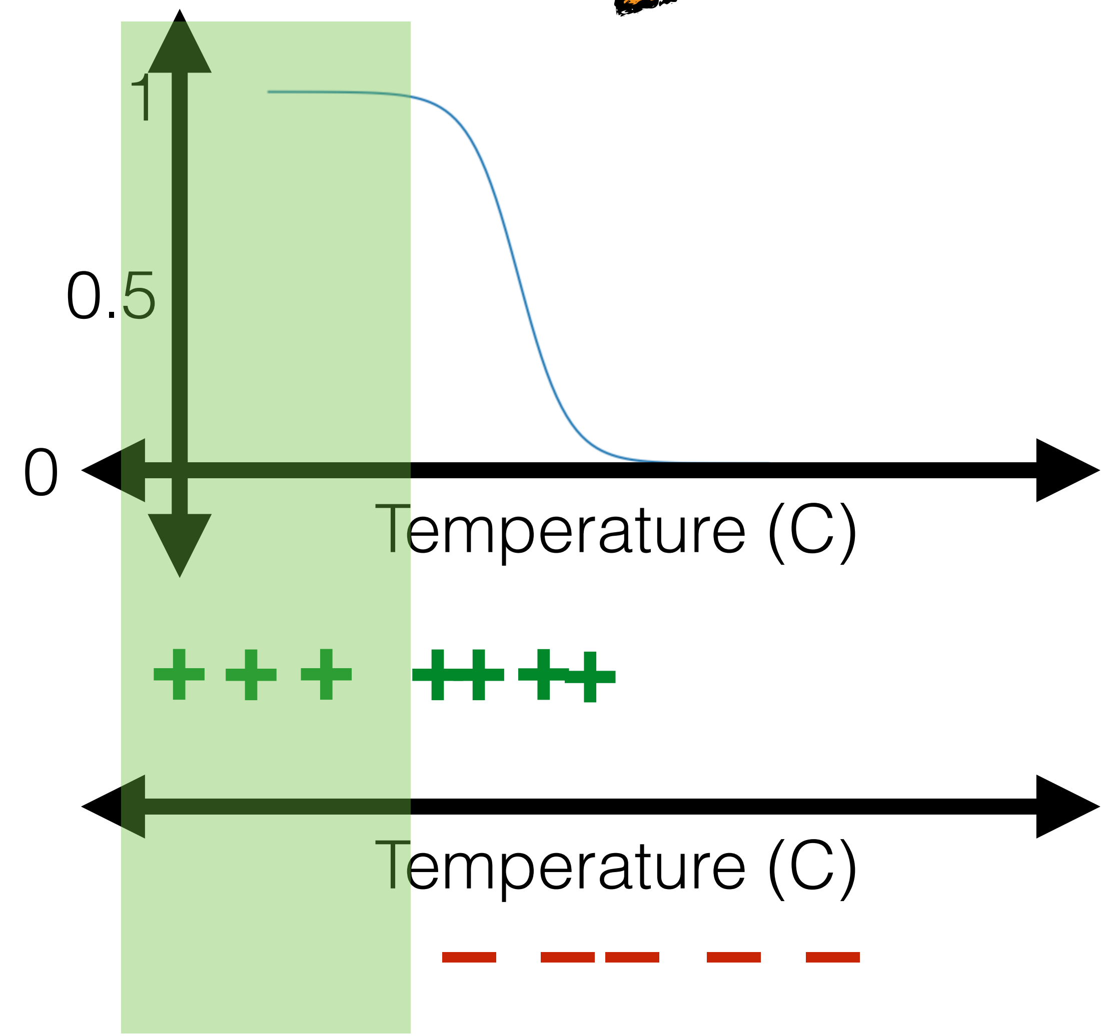
aka logistic regression

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



Linear logistic classification

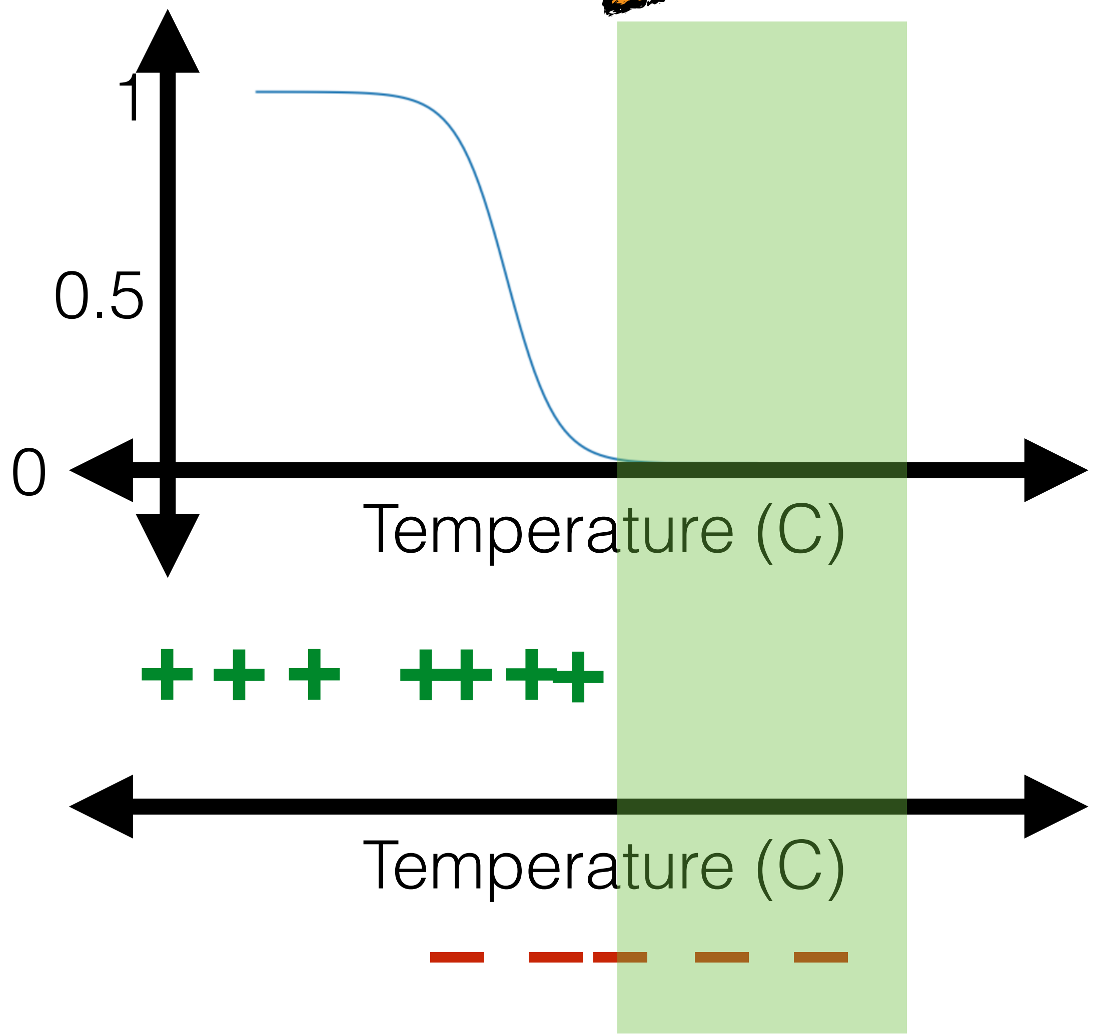
- How do we learn a classifier (i.e. learn θ, θ_0)?



Linear logistic classification

aka logistic regression

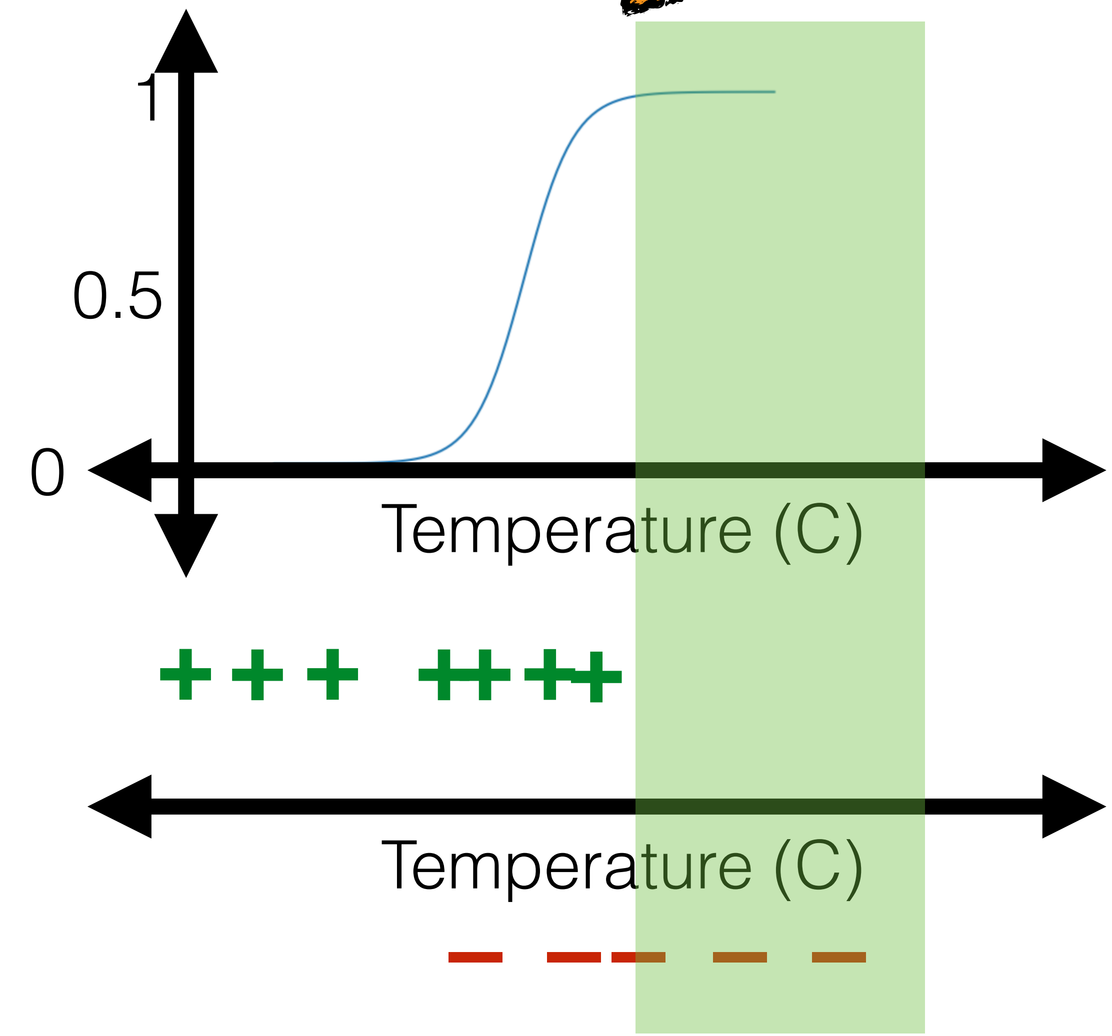
- How do we learn a classifier (i.e. learn θ, θ_0)?



Linear logistic classification

aka logistic regression

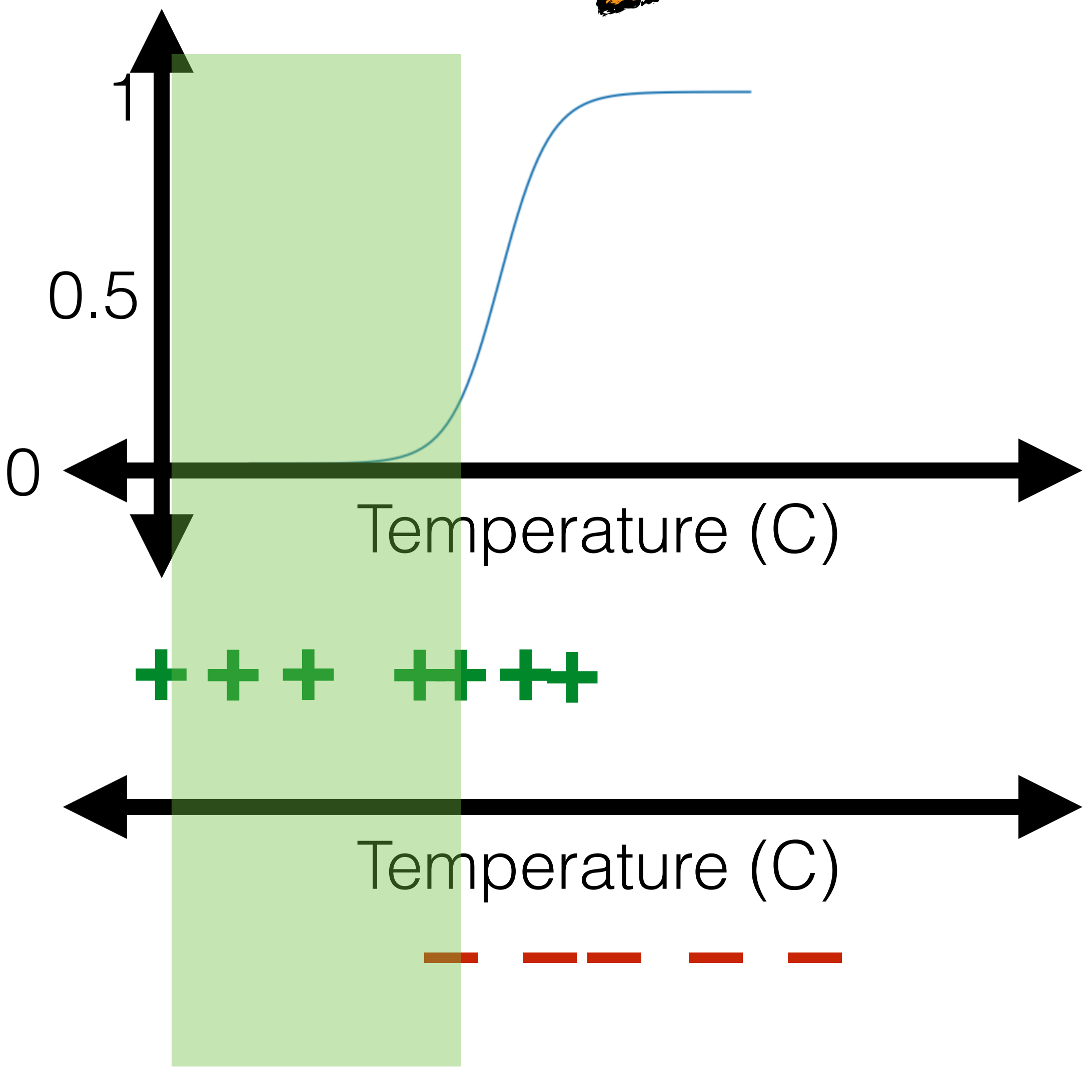
- How do we learn a classifier (i.e. learn θ, θ_0)?



Linear logistic classification

aka logistic regression

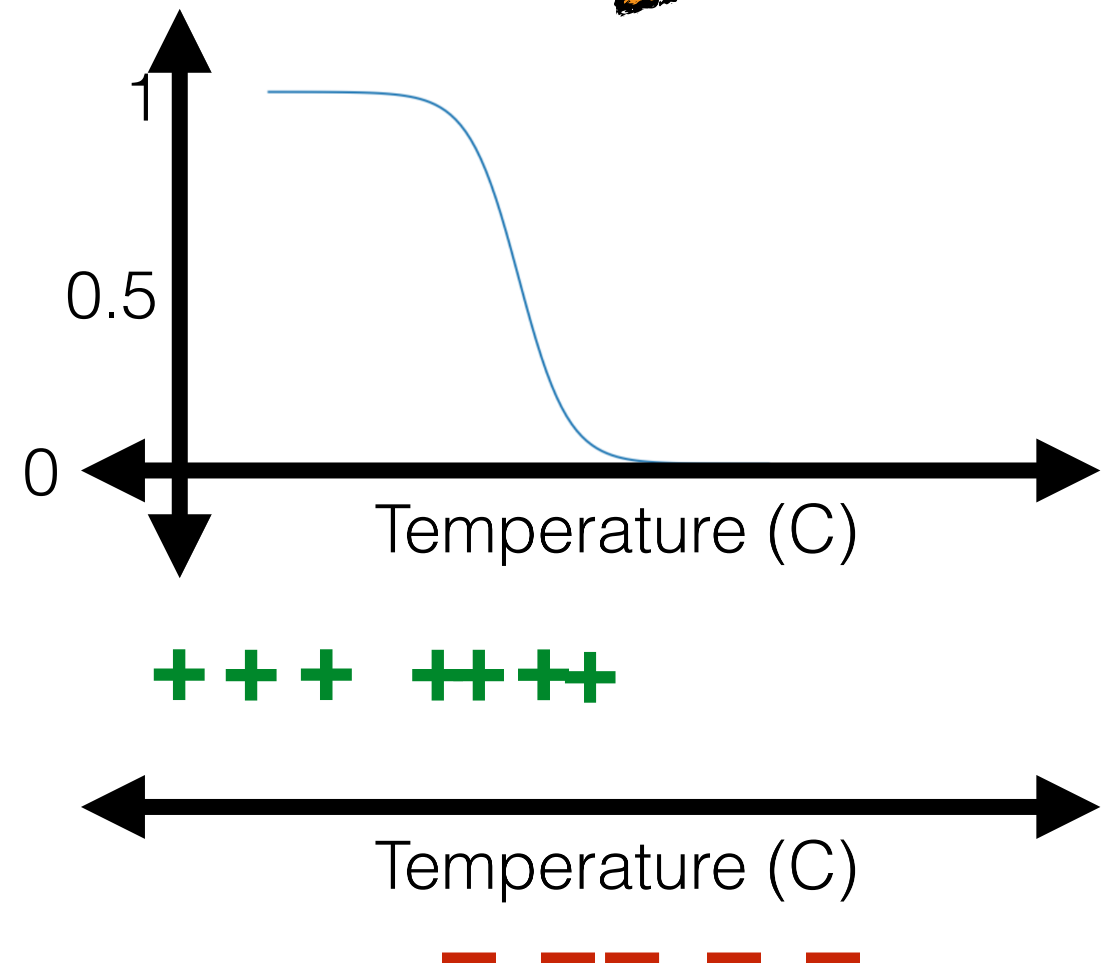
- How do we learn a classifier (i.e. learn θ, θ_0)?



Linear logistic classification

aka logistic regression

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

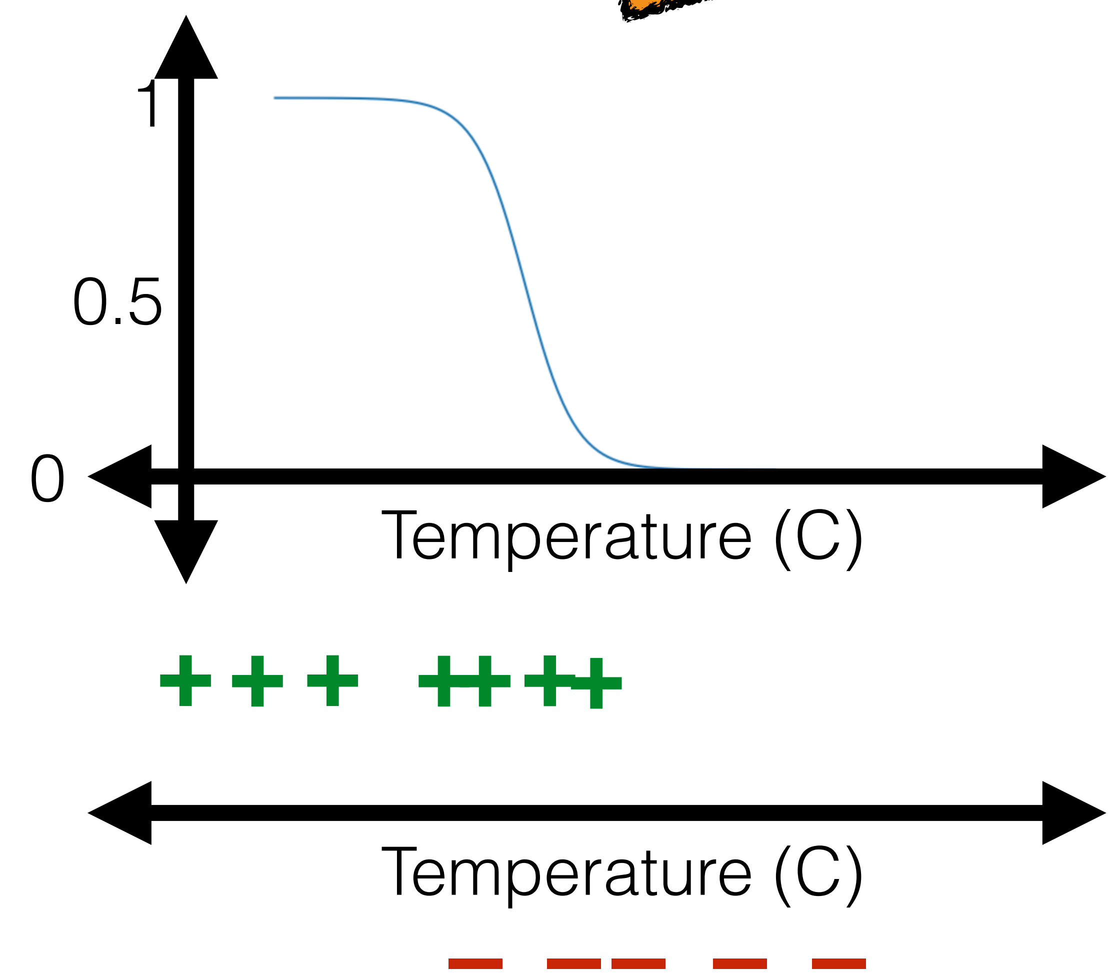


Linear logistic classification

aka logistic regression

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

Probability(data)



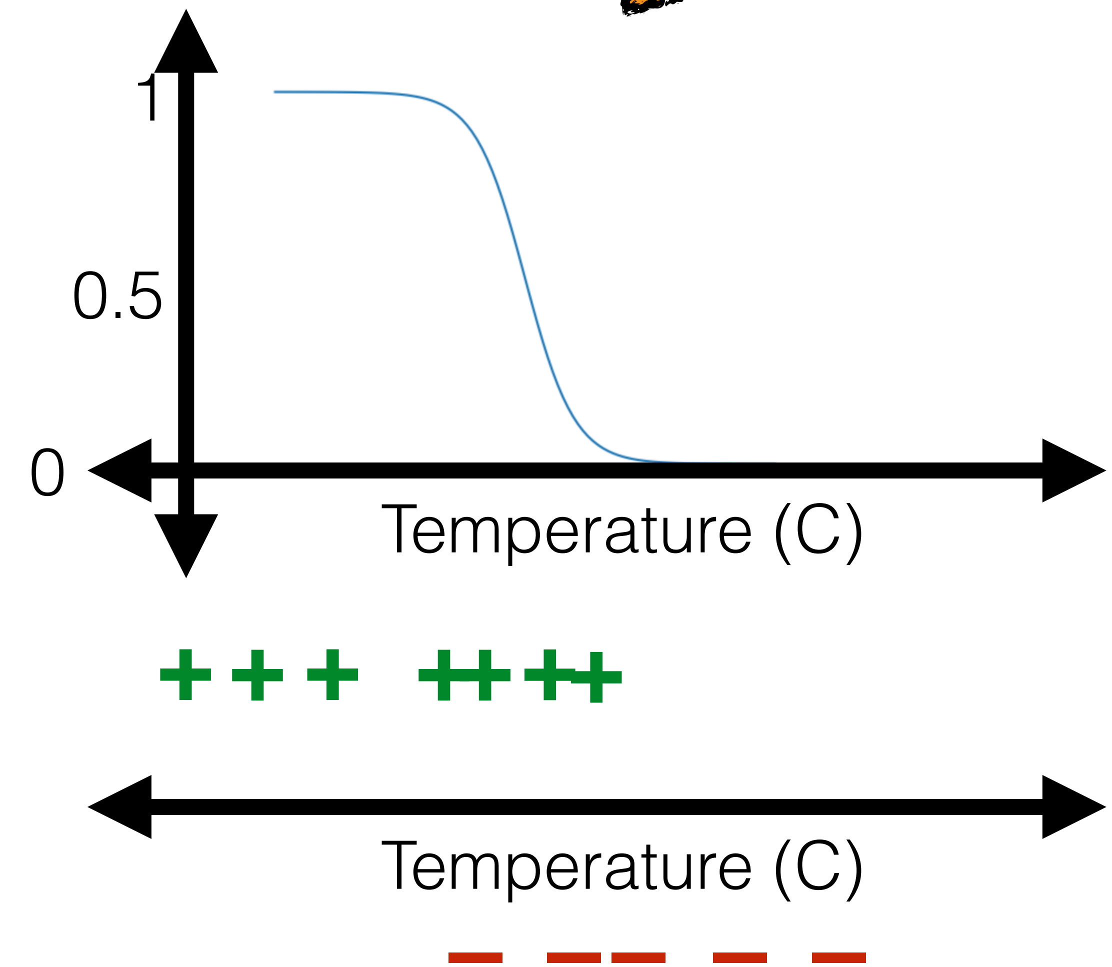
Linear logistic classification

aka logistic regression

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

Probability(data)

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



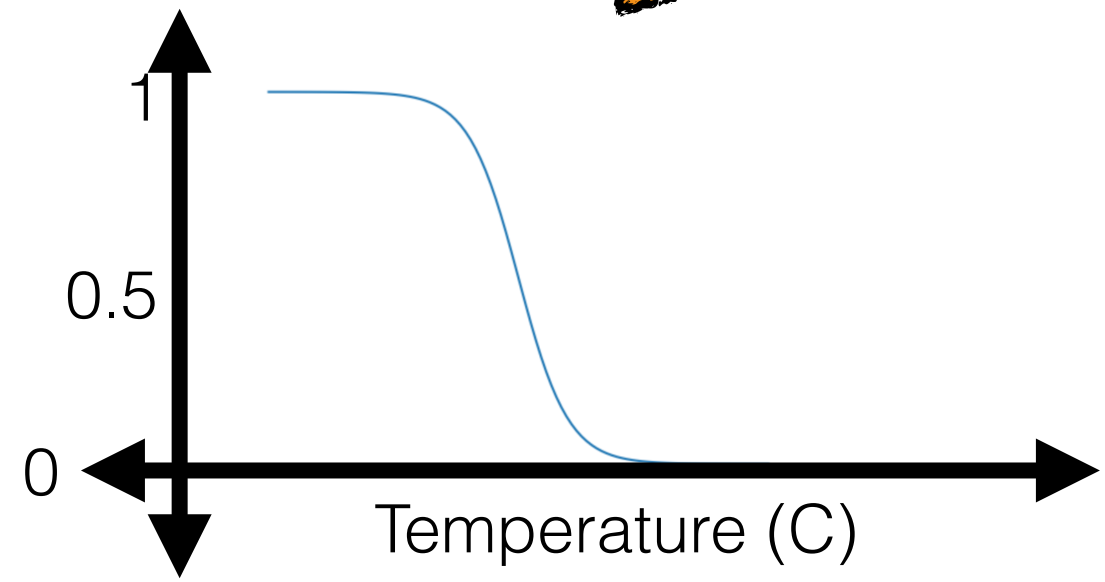
Linear logistic classification

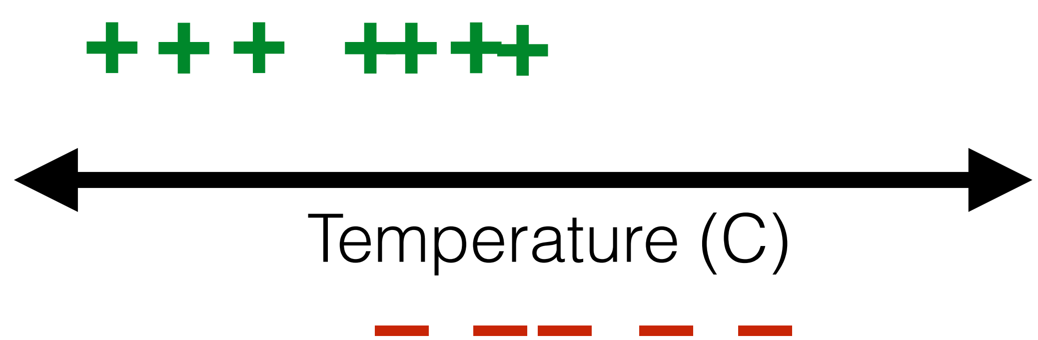
aka logistic regression

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

Probability(data)

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





Linear logistic classification

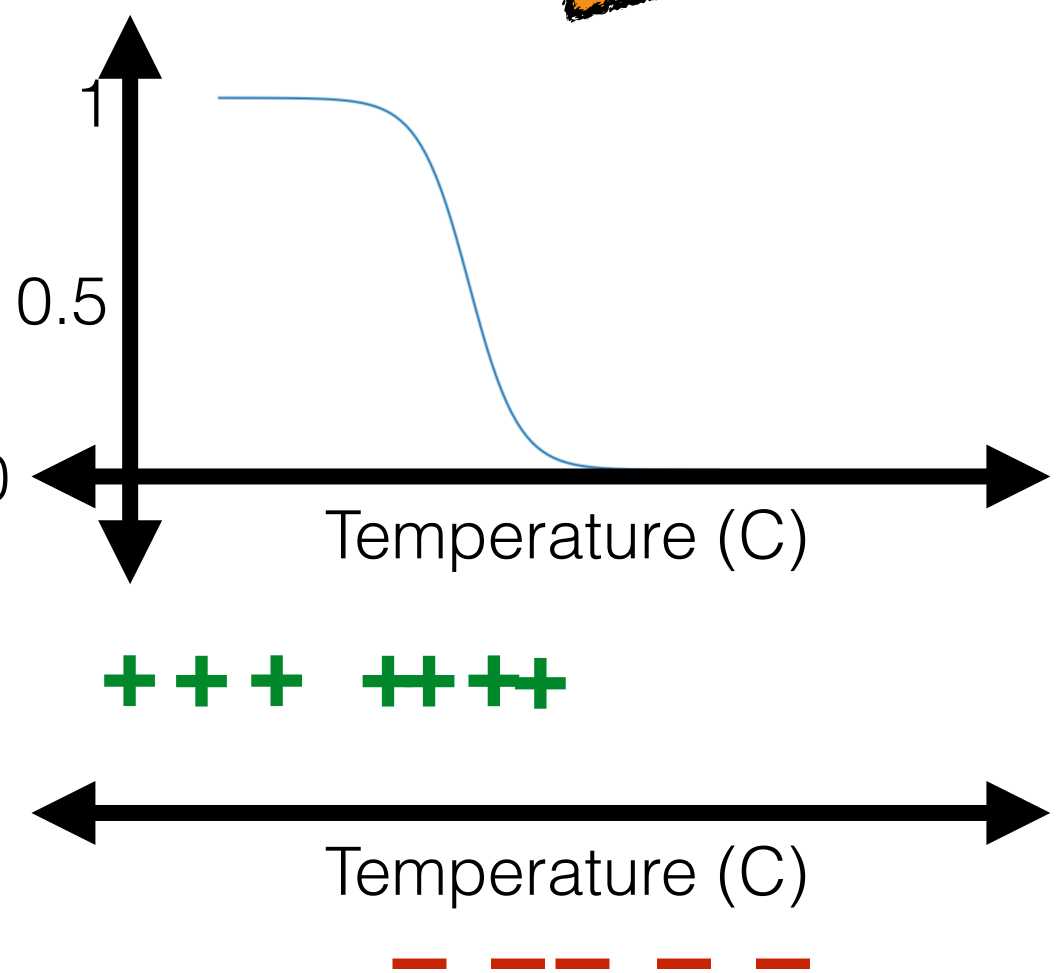
aka logistic regression

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

Probability(data)

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

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



Linear logistic classification

aka logistic regression

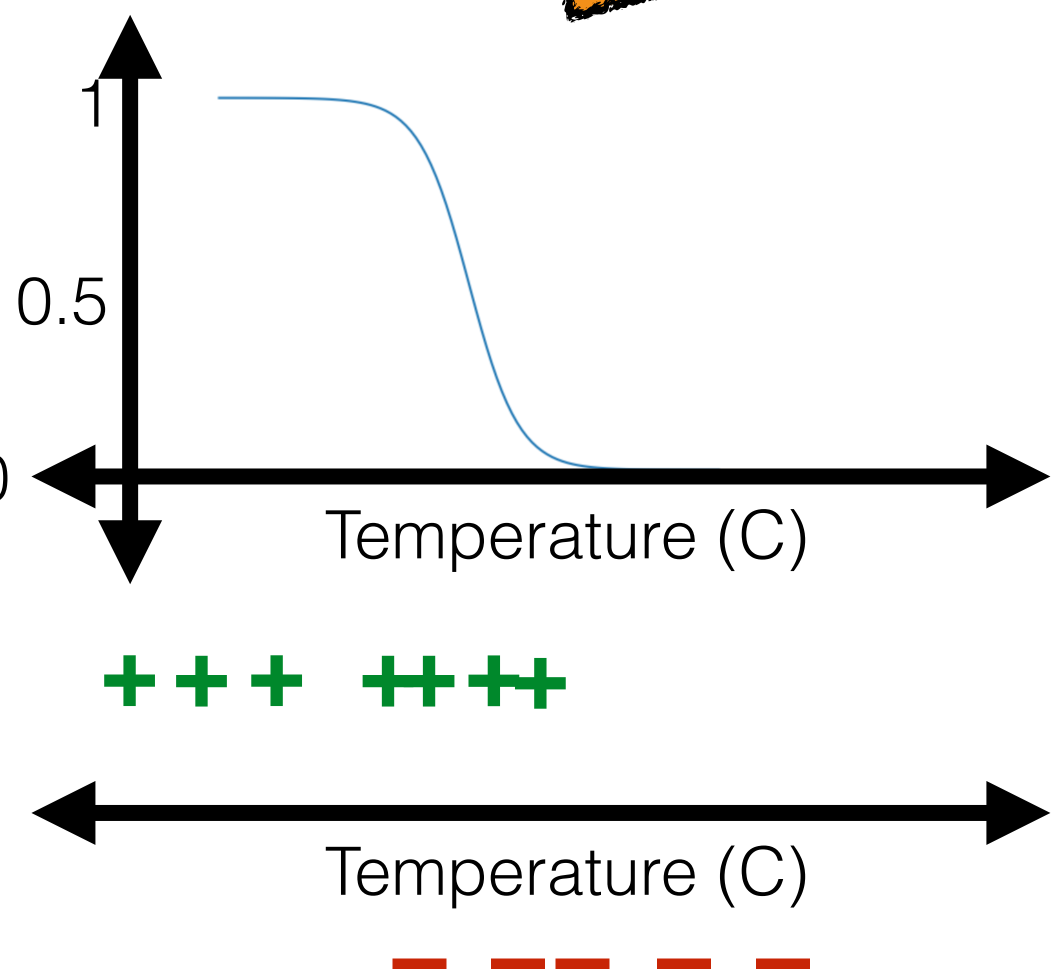
- How do we learn a classifier (i.e. learn θ, θ_0)?

Probability(data)

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

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

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



Linear logistic classification

aka logistic regression

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

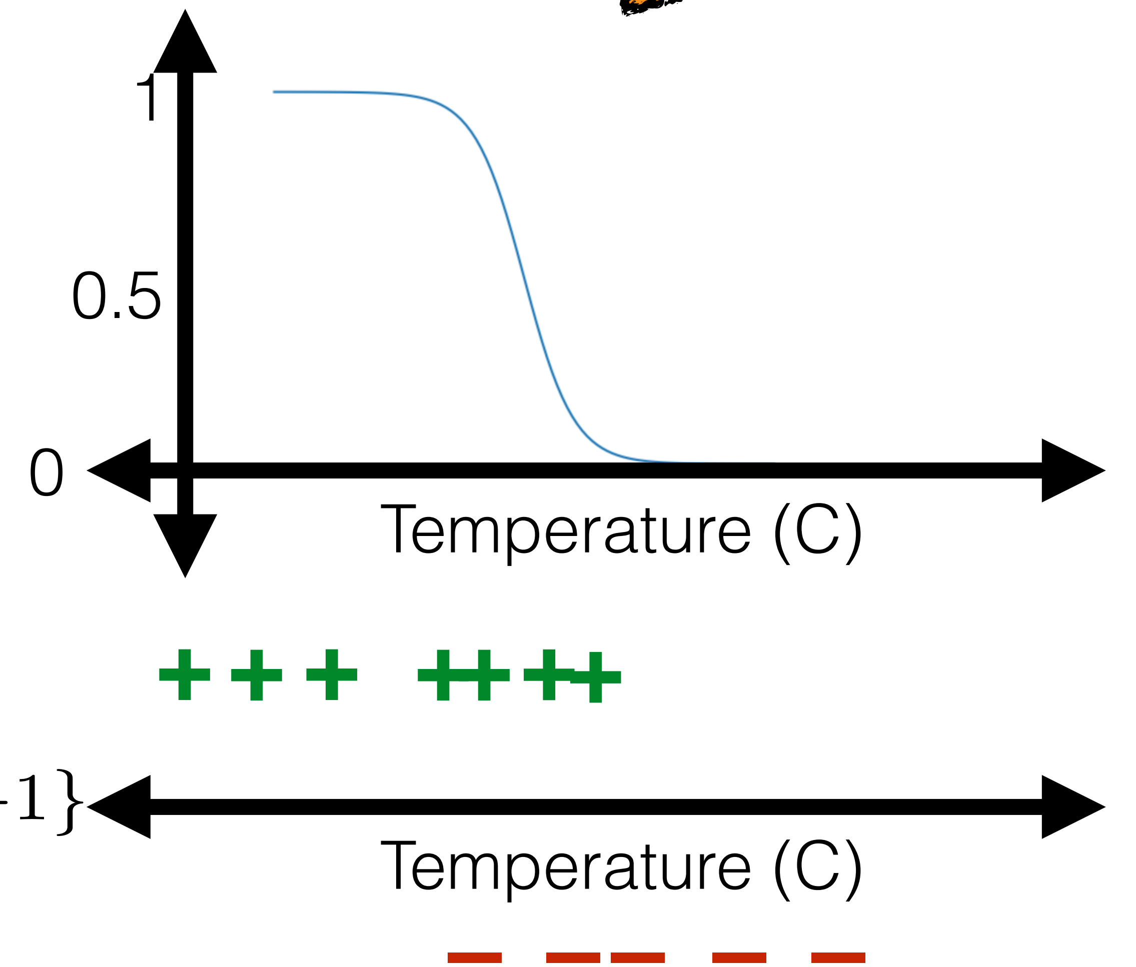
Probability(data)

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

[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)?

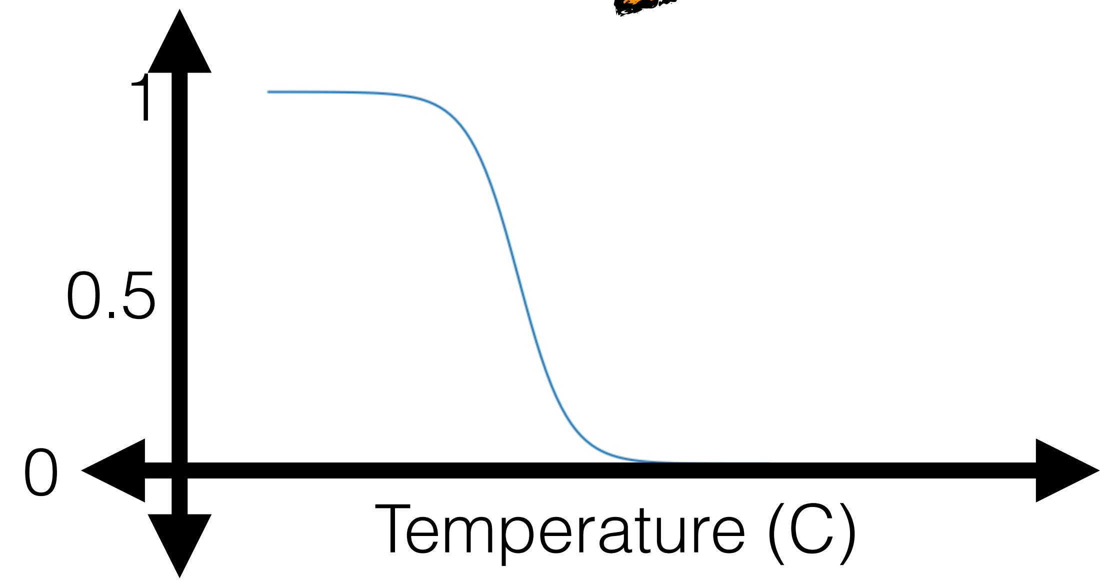
Probability(data)

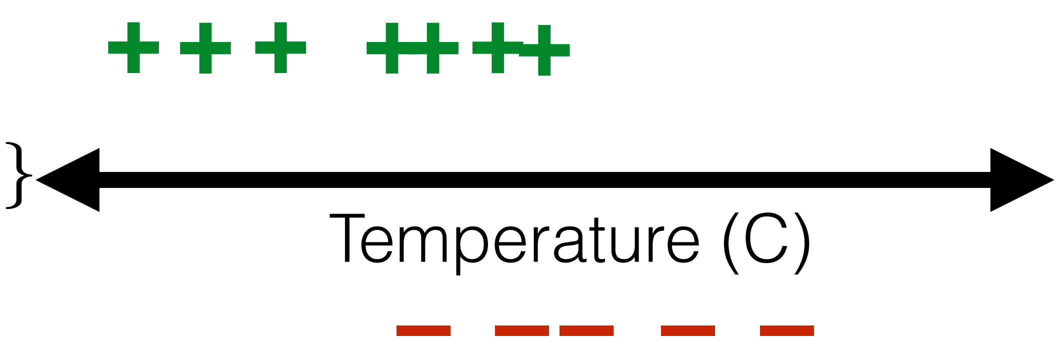
$$= \prod_{i=1}^{n} \text{Probability(data point } i)$$

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

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

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





Linear logistic classification

aka logistic regression

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

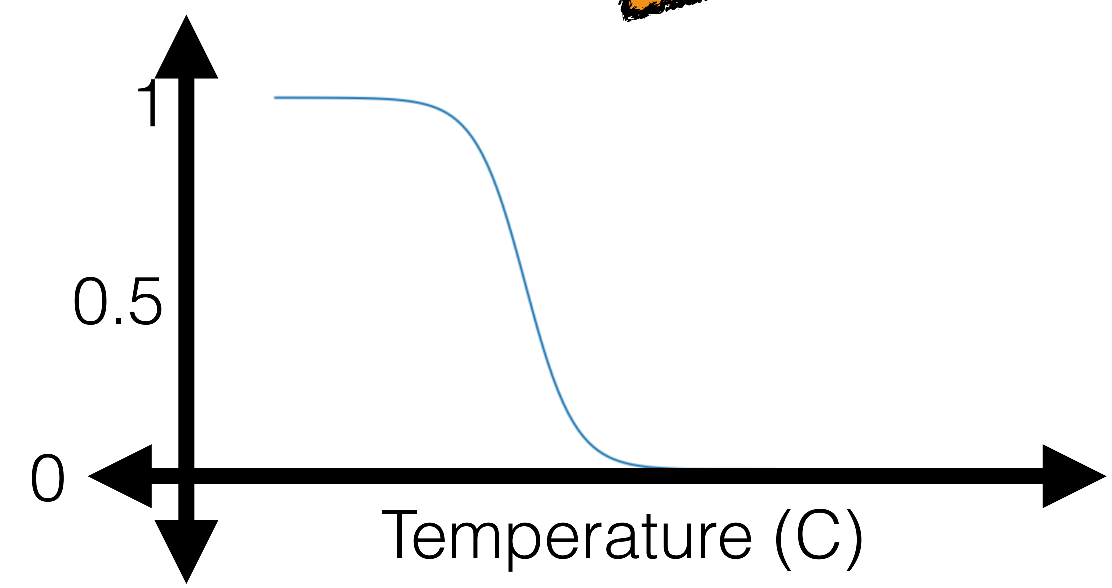
Probability(data)

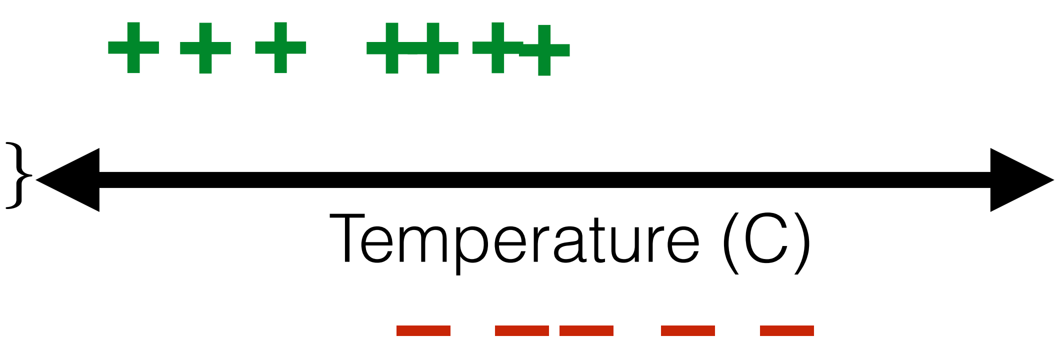
$$= \prod_{i=1}^{n} \text{Probability}(\text{data point } i)$$

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

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

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





Linear logistic classification

aka logistic regression

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

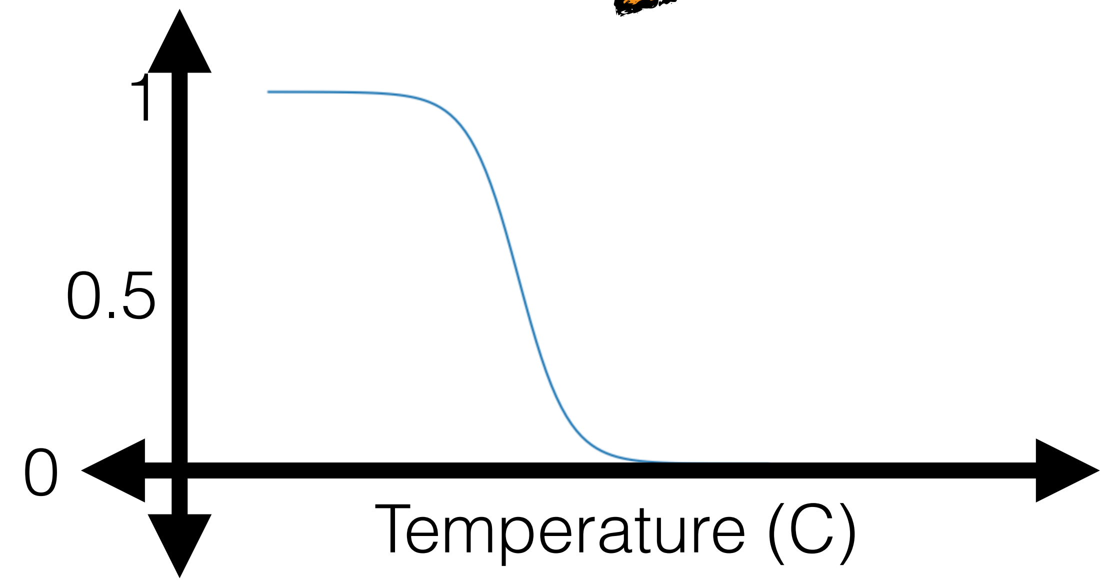
Probability(data)

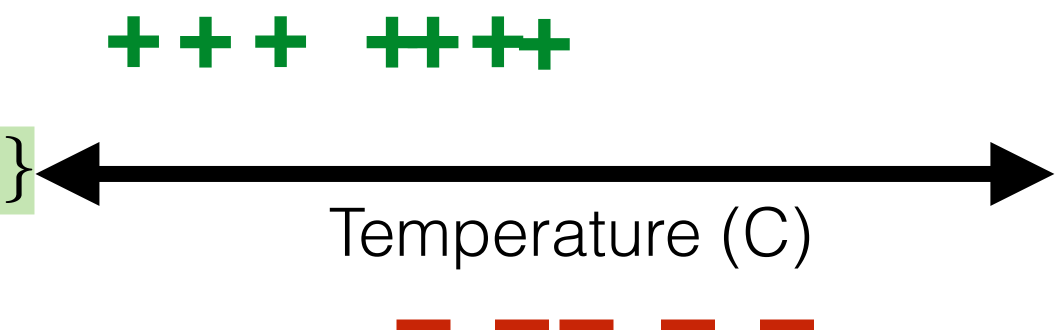
$$= \prod_{i=1}^{n} \text{Probability(data point } i)$$

[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)?

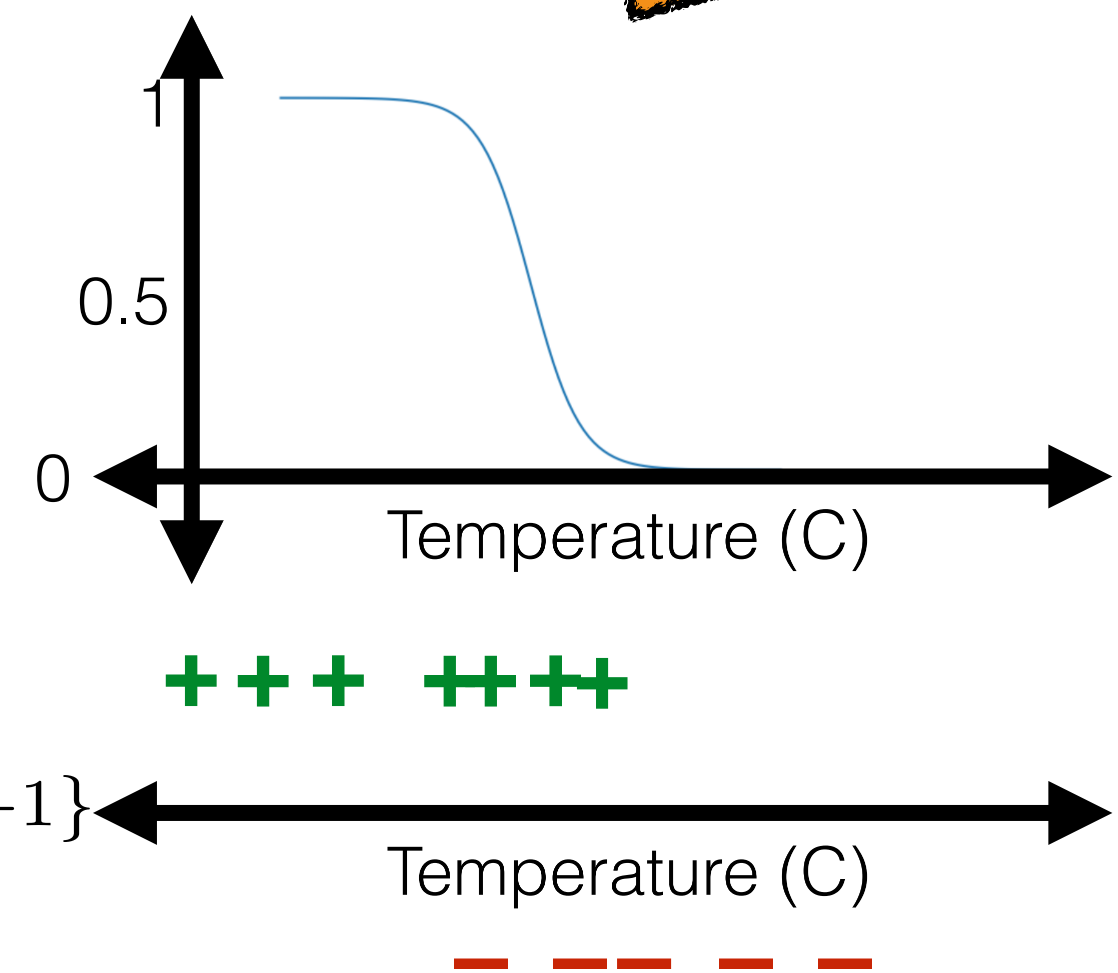
Probability(data)

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

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

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

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



Linear logistic classification

aka logistic regression

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

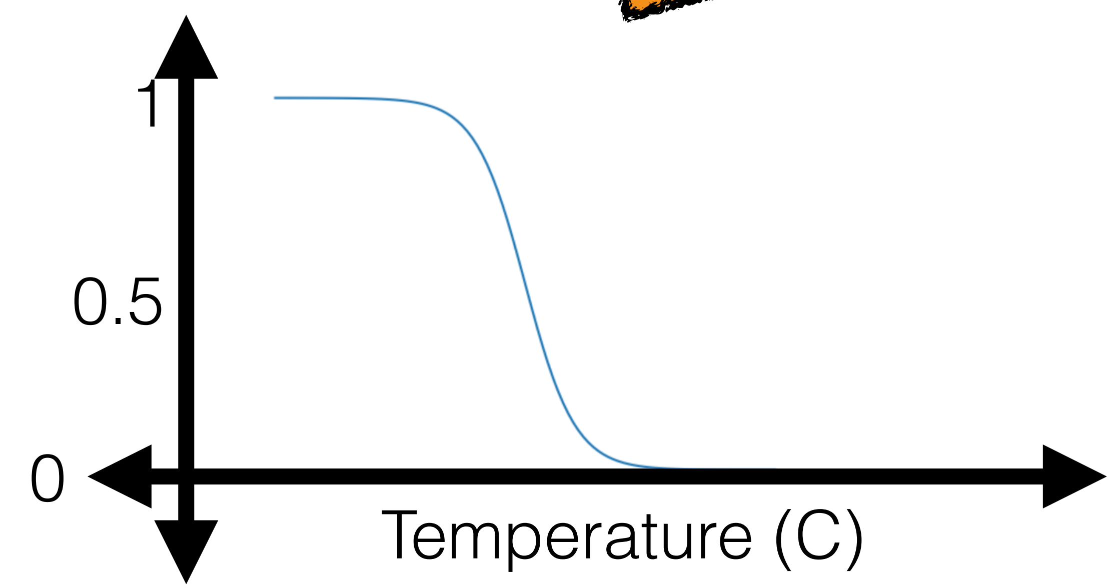
Probability(data)

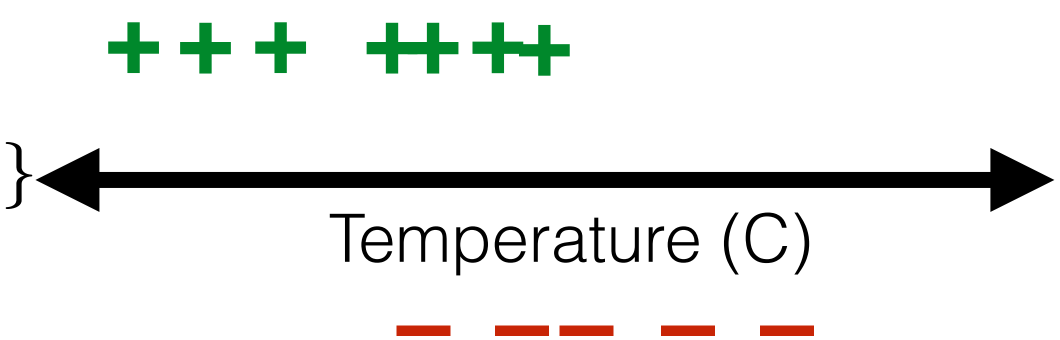
$$= \prod_{i=1}^{n} \text{Probability}(\text{data point } i)$$

[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

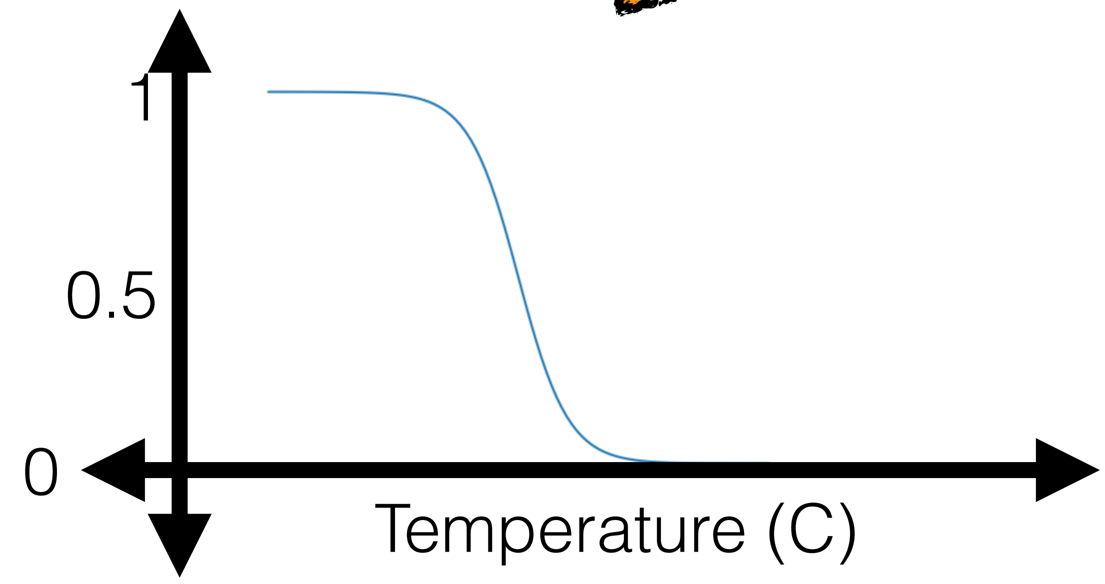
- How do we learn a classifier (i.e. learn θ, θ_0)?

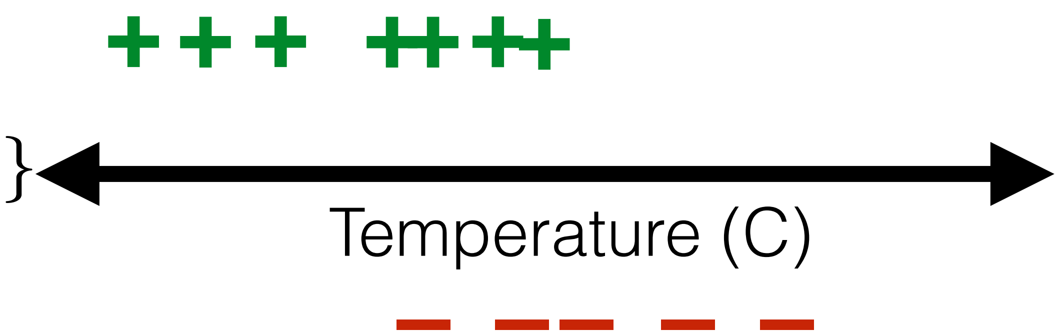
Probability(data)

$$= \prod_{i=1}^{n} \text{Probability}(\text{data point } i) \quad [\text{Let } g^{(i)} = \sigma(\theta^\top x^{(i)} + \theta_0)\,]$$

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

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





Loss(data) = -log probability(data)

Linear logistic classification

aka logistic regression

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

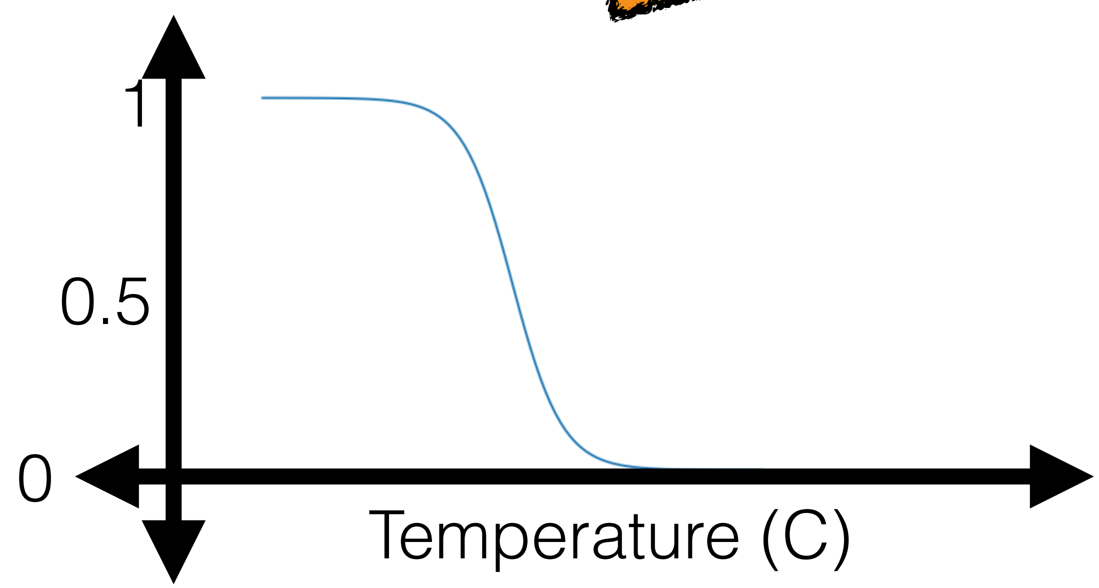
Probability(data)

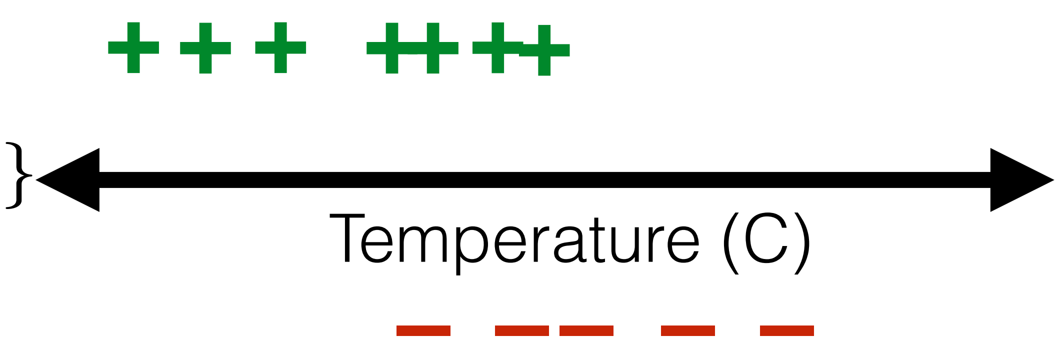
$$= \prod_{i=1}^{n} \text{Probability(data point } i)$$

[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)

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

Linear logistic classification

aka logistic regression

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

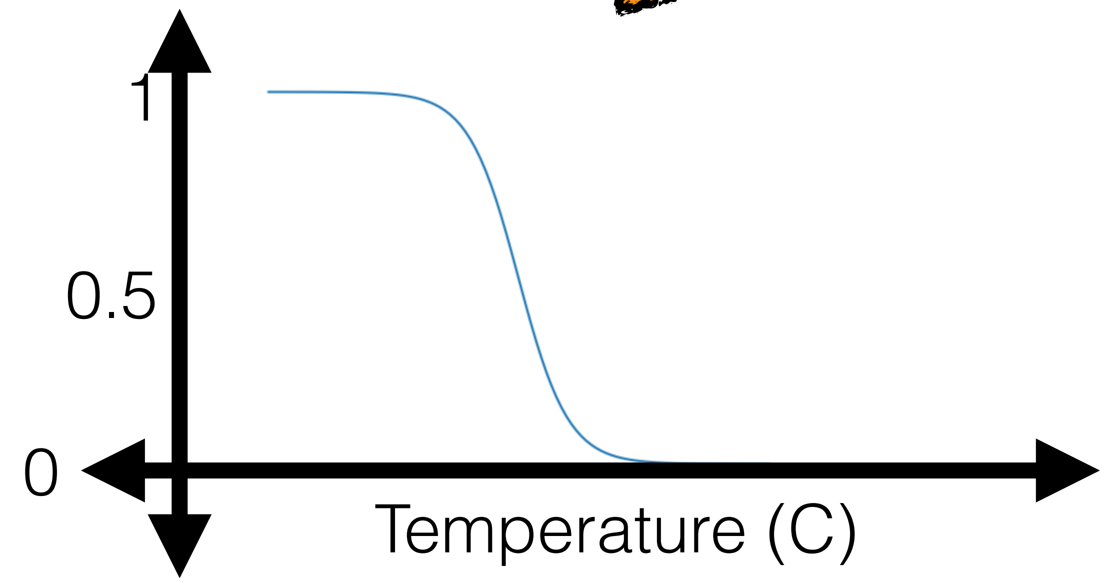
Probability(data)

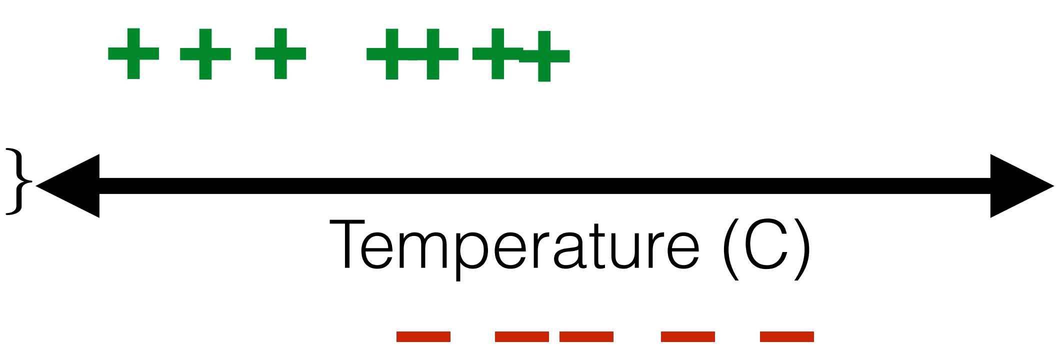
$$= \prod_{i=1}^{n} \text{Probability}(\text{data point } i)$$

[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)

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

Linear logistic classification

aka logistic regression

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

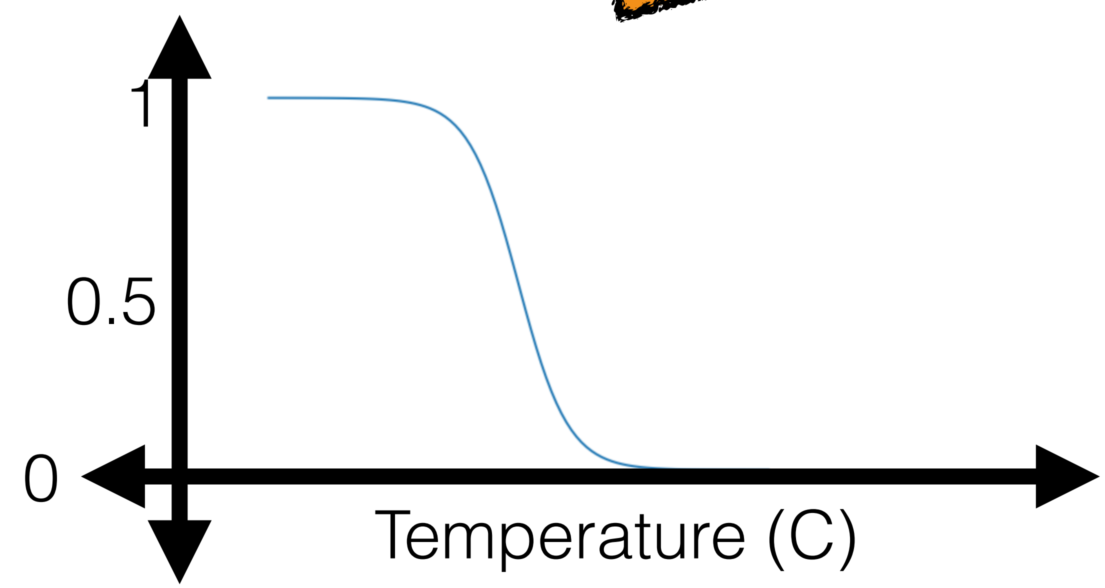
Probability(data)

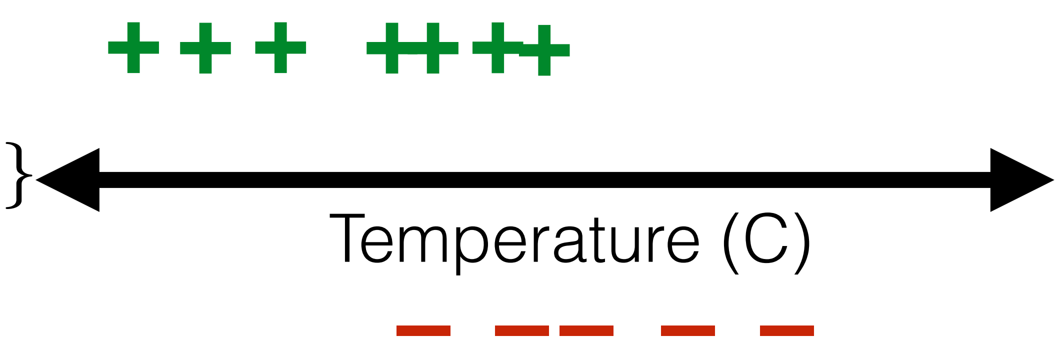
$$= \prod_{i=1}^{n} \text{Probability}(\text{data point } i)$$

[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)?

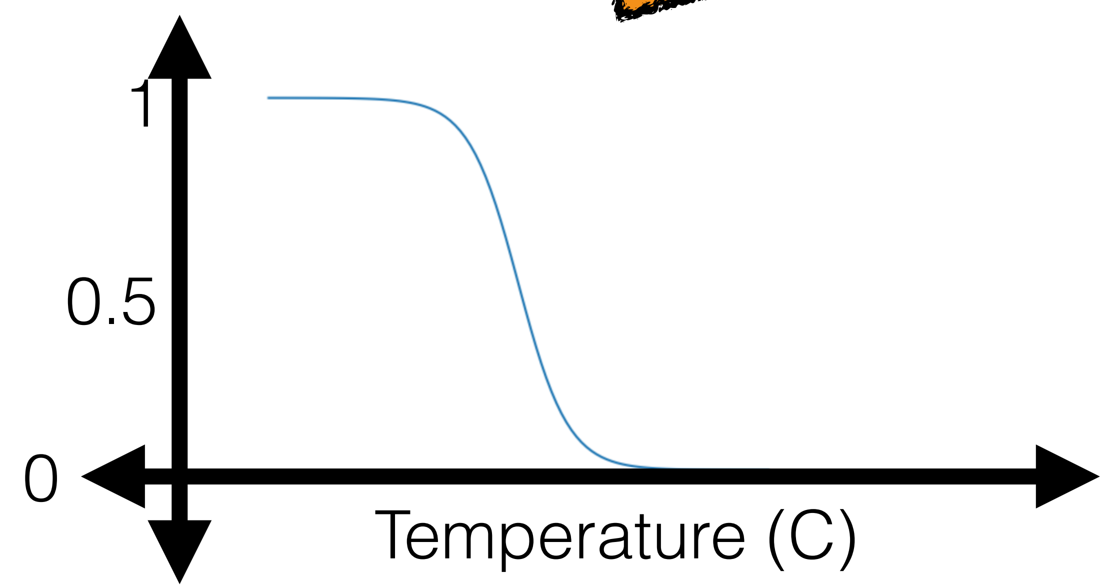
Probability(data)

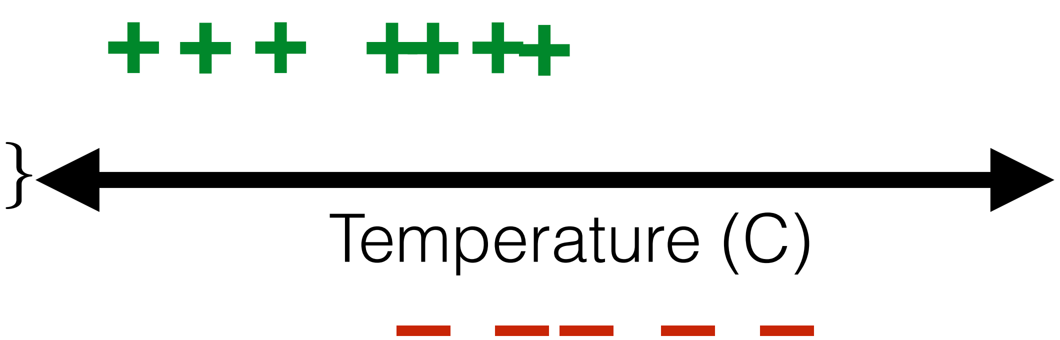
$$= \prod_{i=1}^{n} \text{Probability(data point } i)$$

[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)

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

Linear logistic classification

aka logistic regression

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

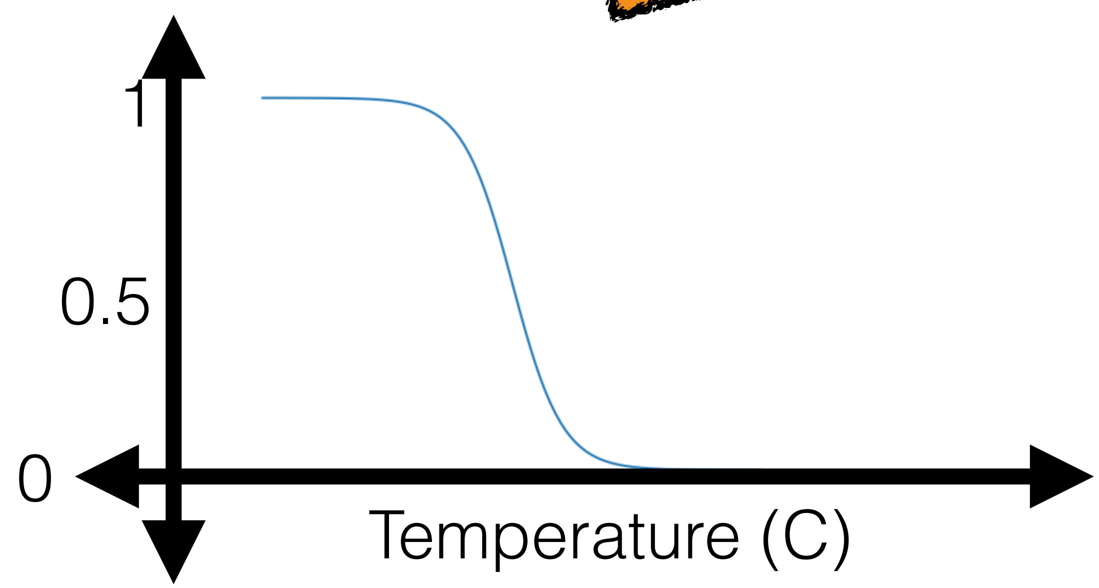
Probability(data)

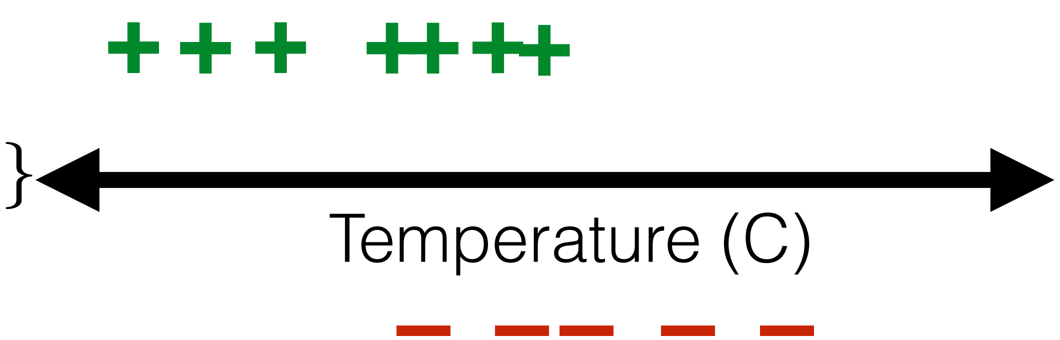
$$= \prod_{i=1}^{n} \text{Probability(data point } i)$$

[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)

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

Linear logistic classification

aka logistic regression

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

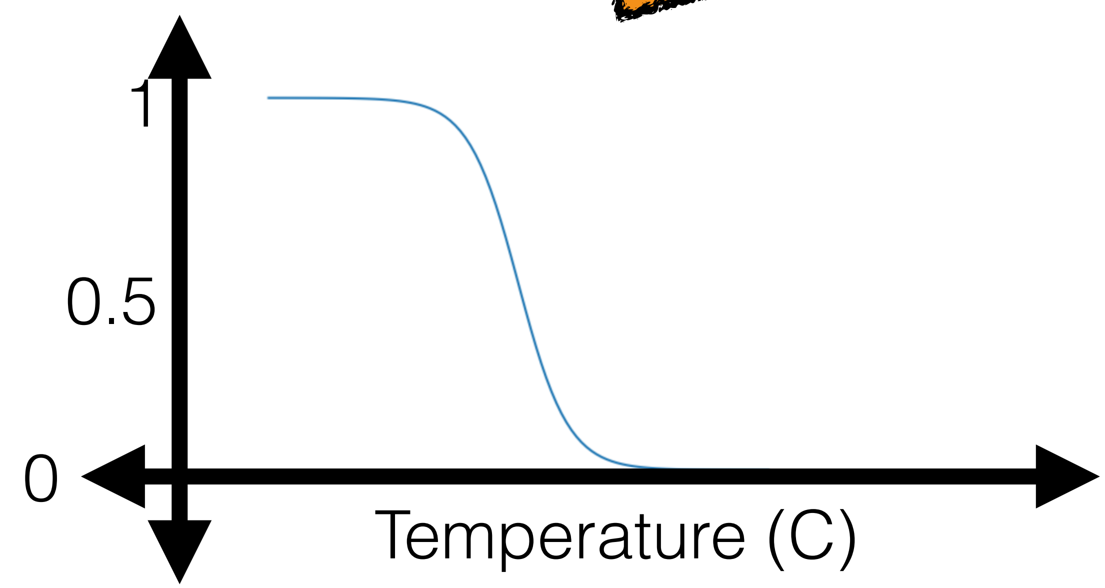
Probability(data)

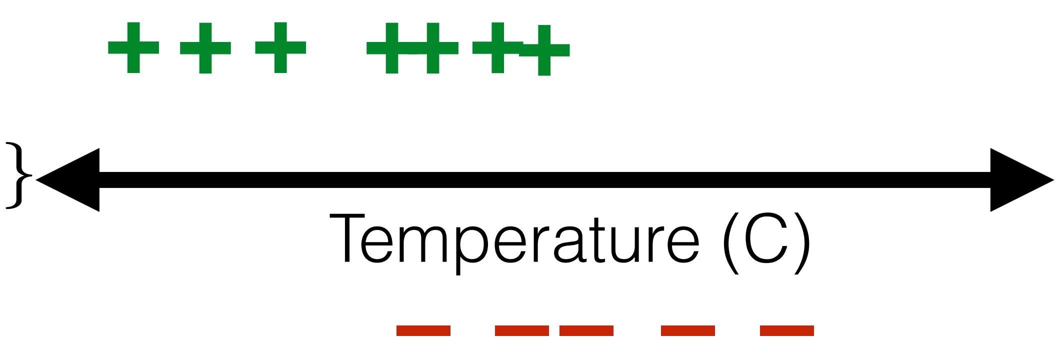
$$= \prod_{i=1}^{n} \text{Probability(data point } i)$$

[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)?

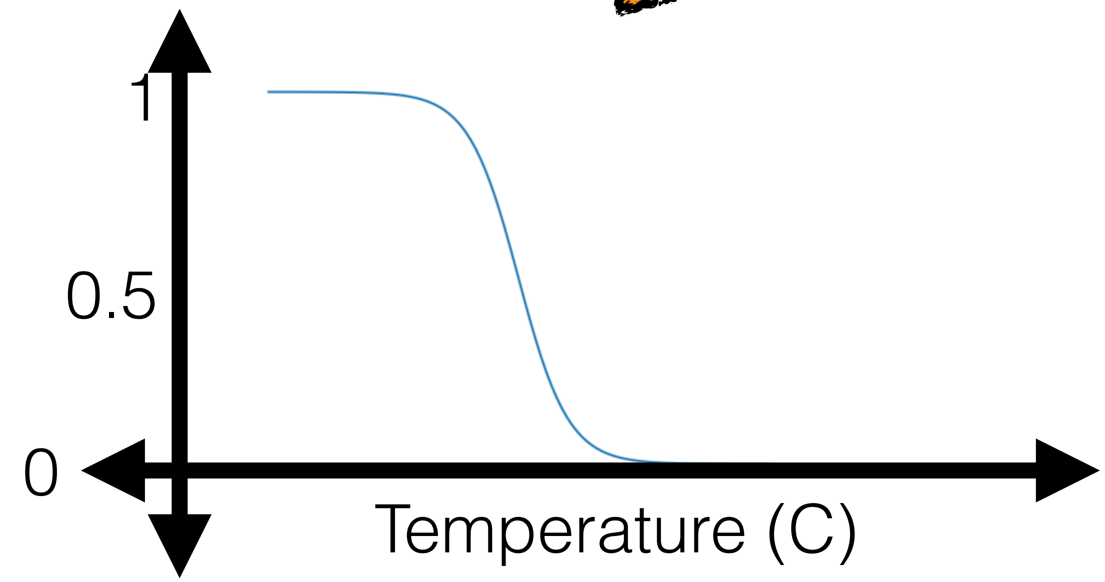
Probability(data)

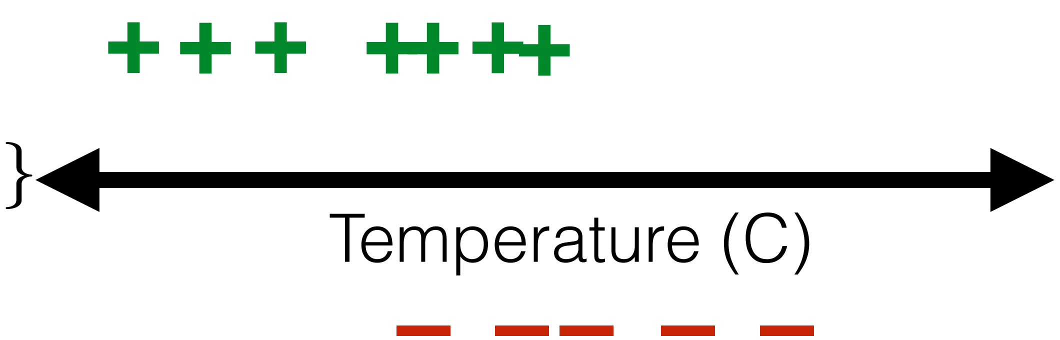
$$= \prod_{i=1}^{n} \text{Probability(data point } i)$$

[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) = -(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)?

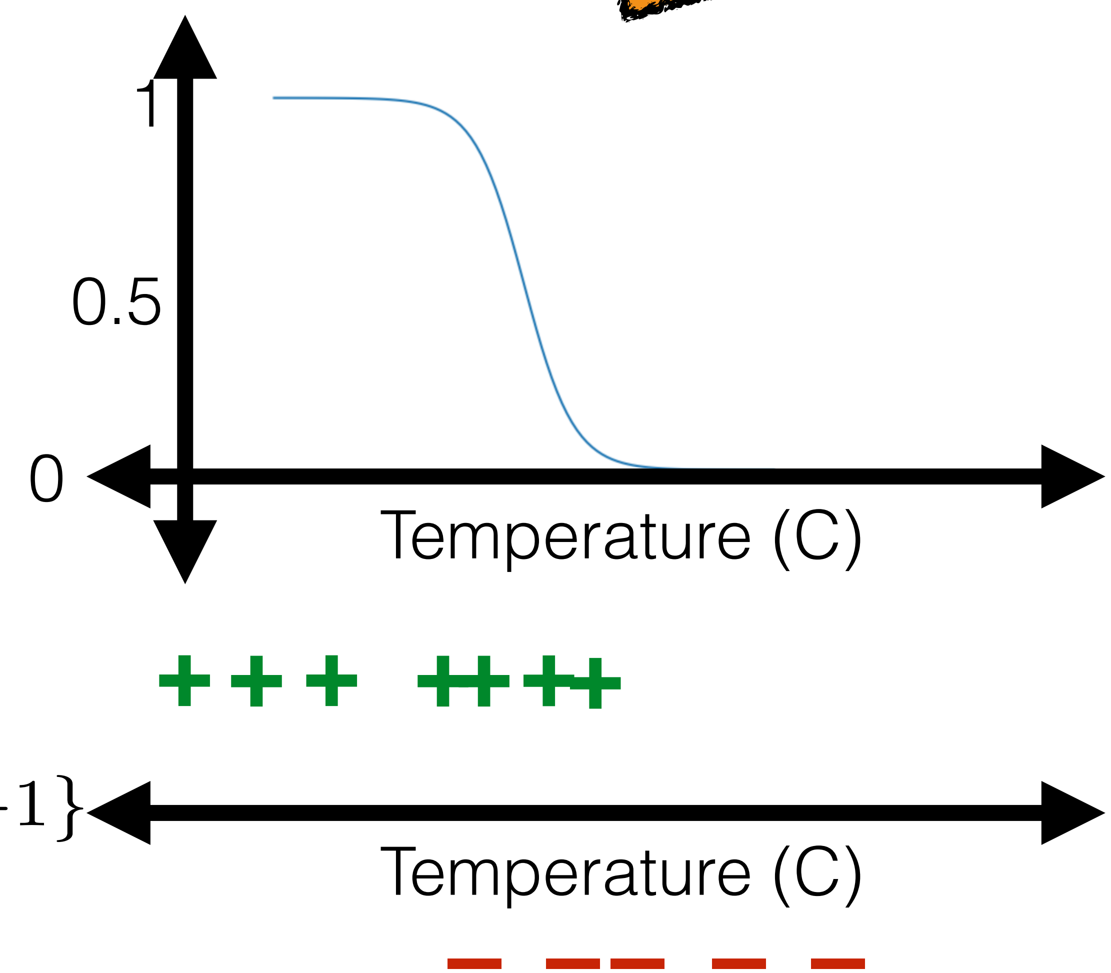
Probability(data)

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

[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) = -(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)?

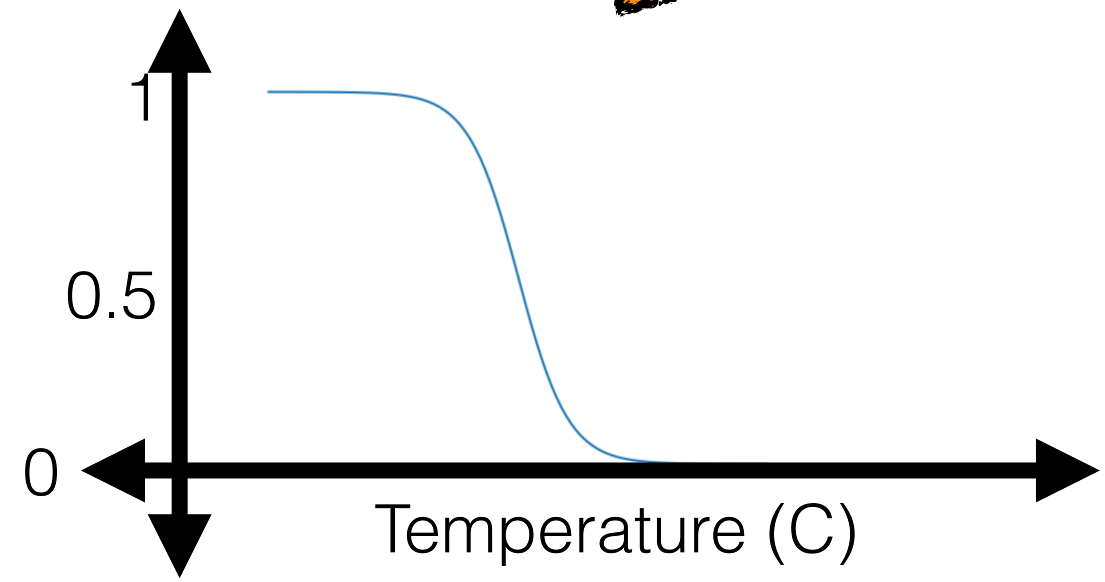
Probability(data)

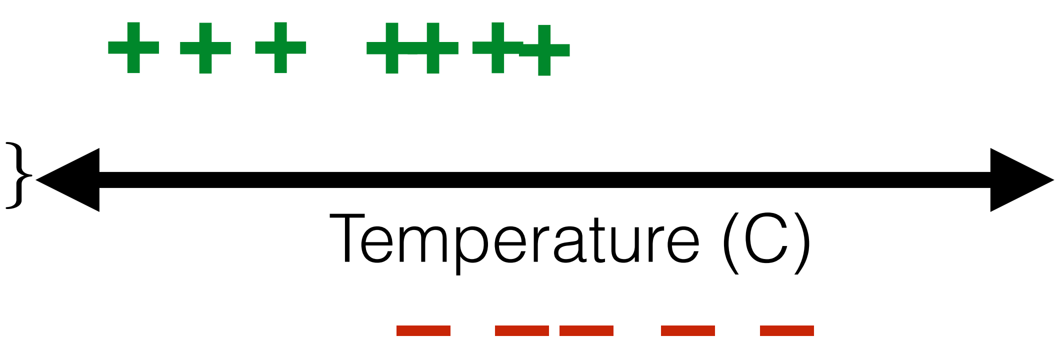
$$= \prod_{i=1}^{n} \text{Probability(data point } i)$$

[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) = -(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)?

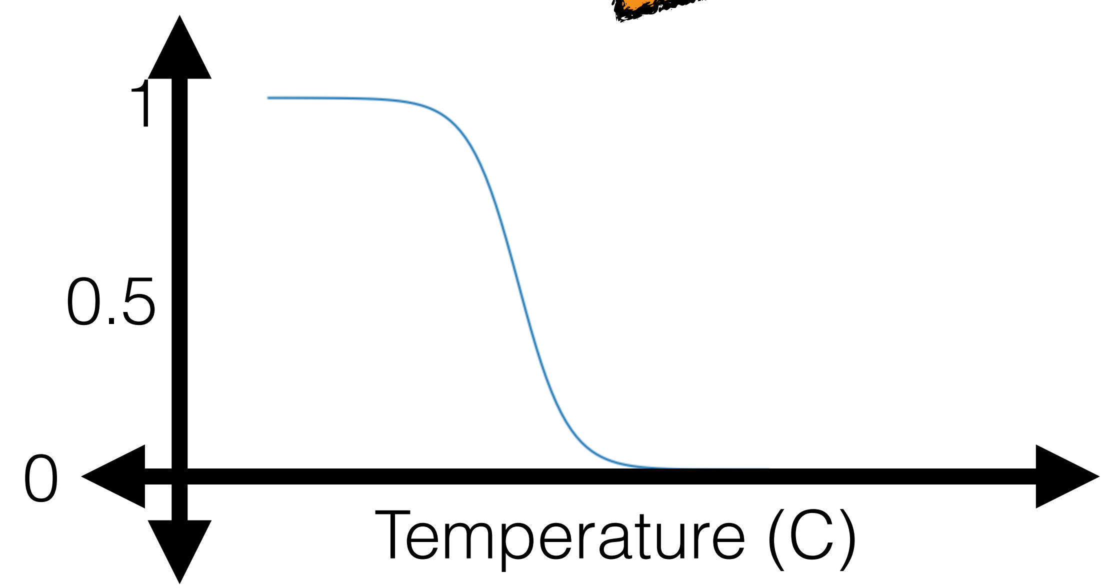
Probability(data)

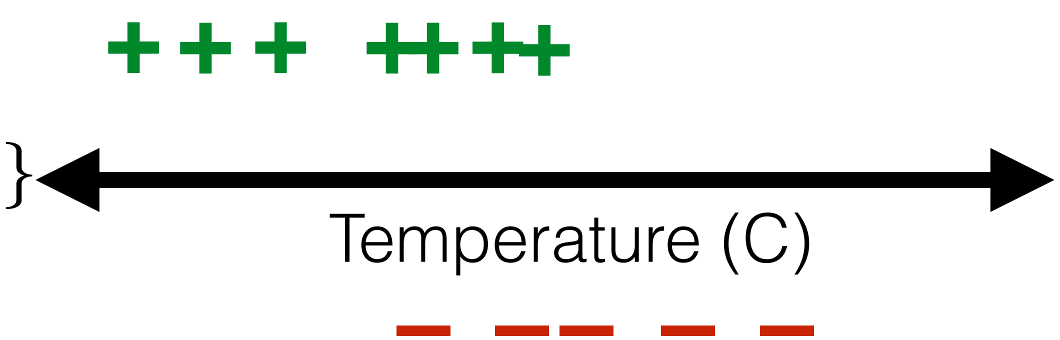
$$= \prod_{i=1}^{n} \text{Probability(data point } i)$$

[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) = -(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

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)?
- 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

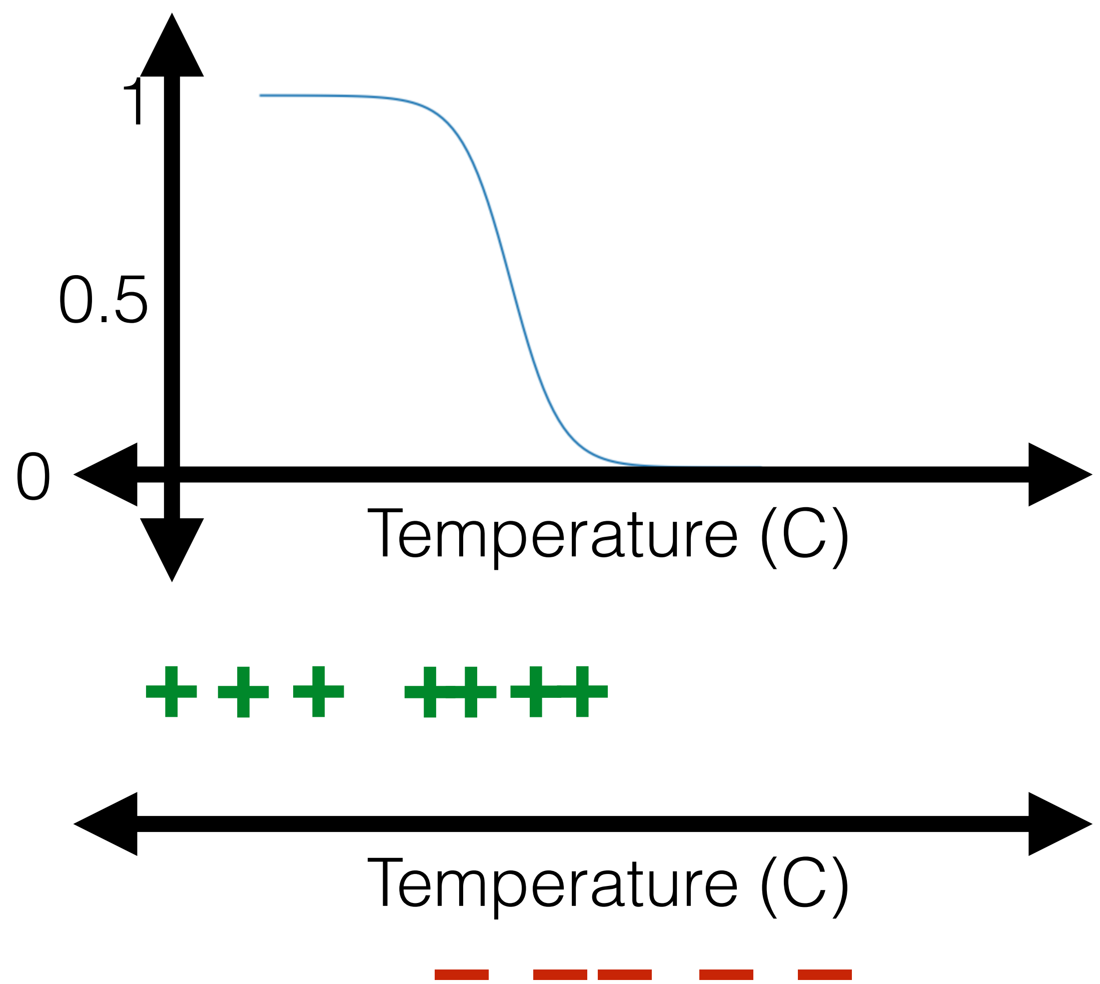
$$J_{\mathrm{lr}}(\Theta) = J_{\mathrm{lr}}(\theta, \theta_0) = \frac{1}{n}\sum_{i=1}^{n} L_{\mathrm{nll}}(\sigma(\theta^\top x^{(i)} + \theta_0), y^{(i)})$$

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

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

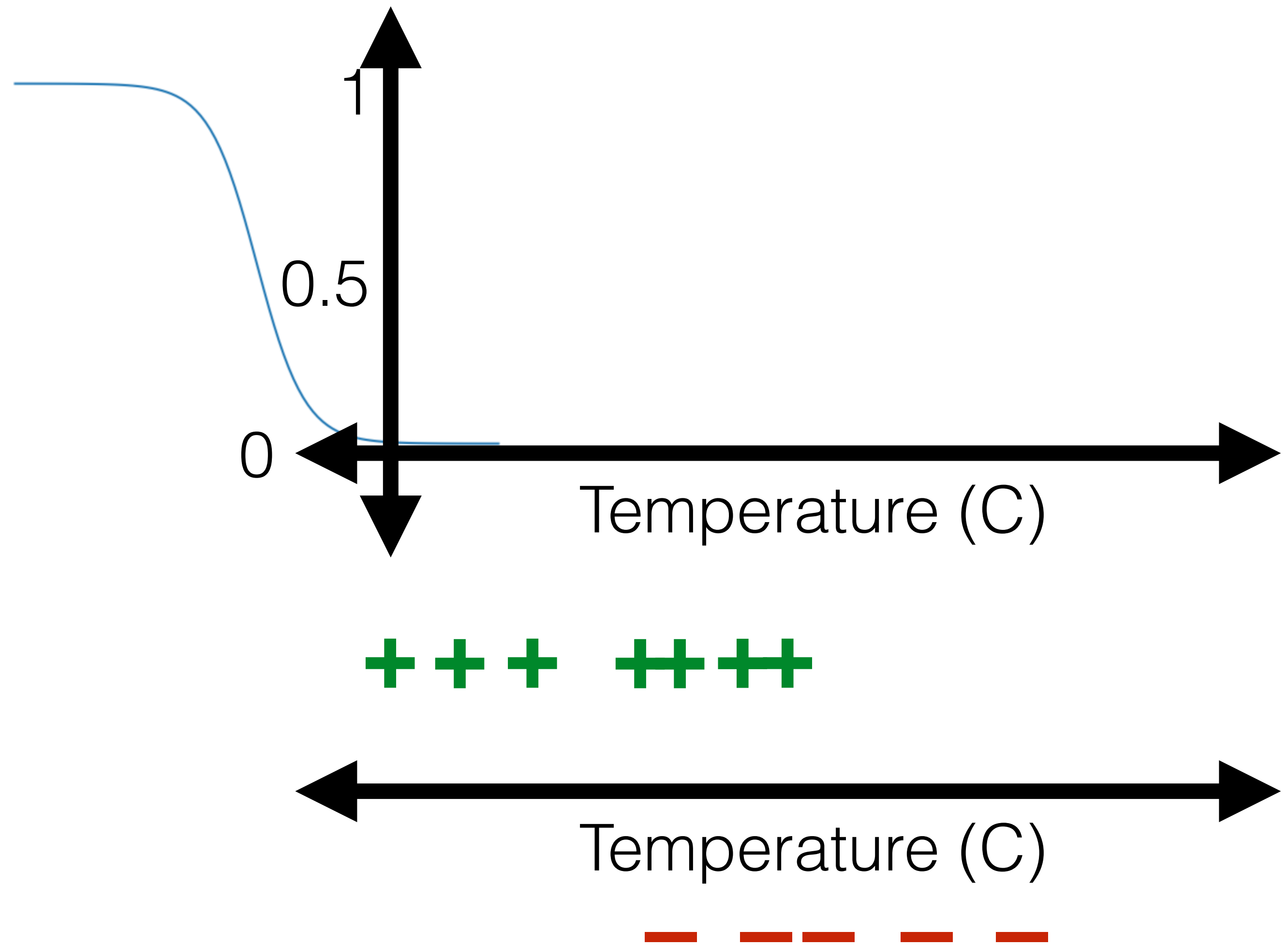


Linear logistic classification

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

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

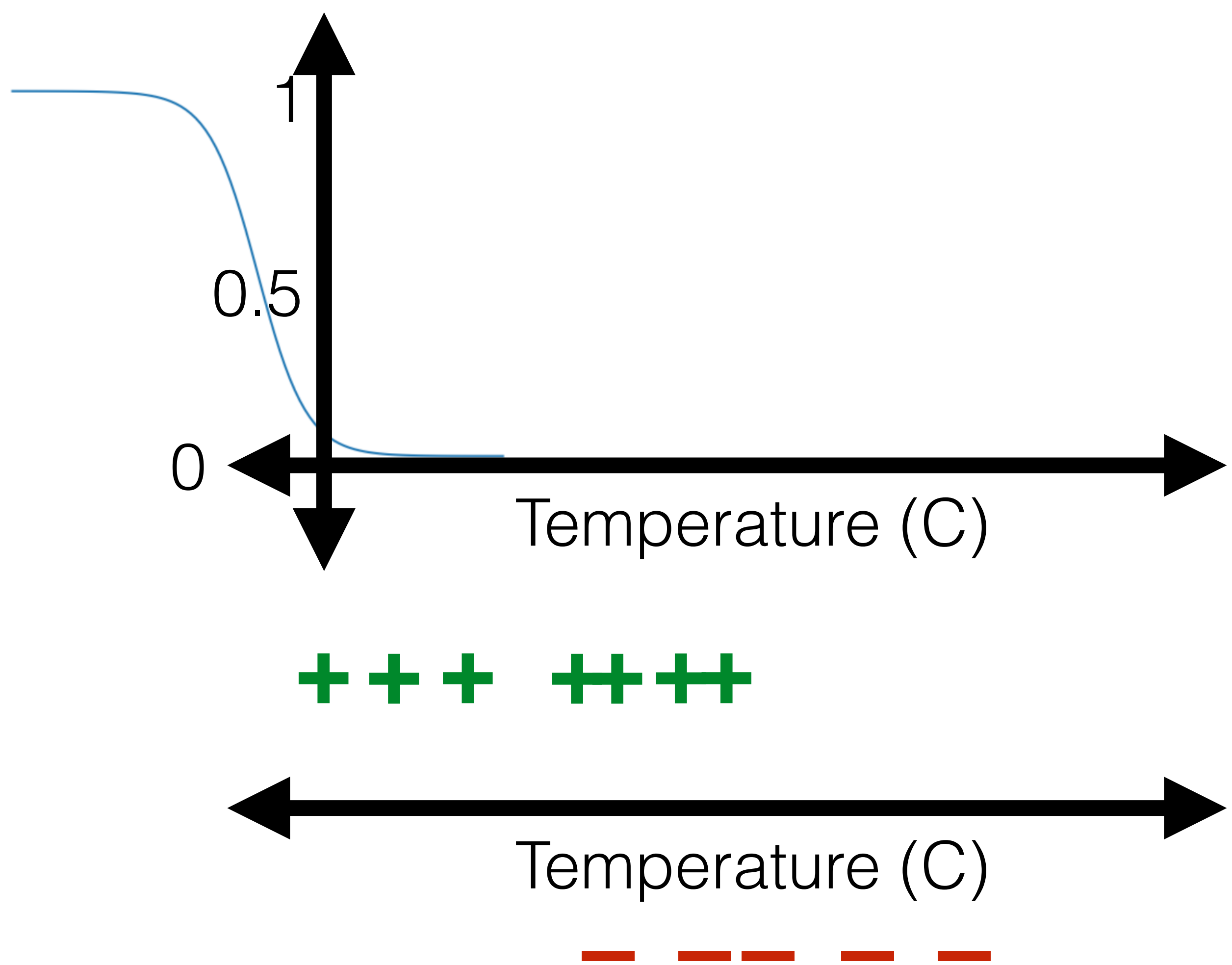


Linear logistic classification

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

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

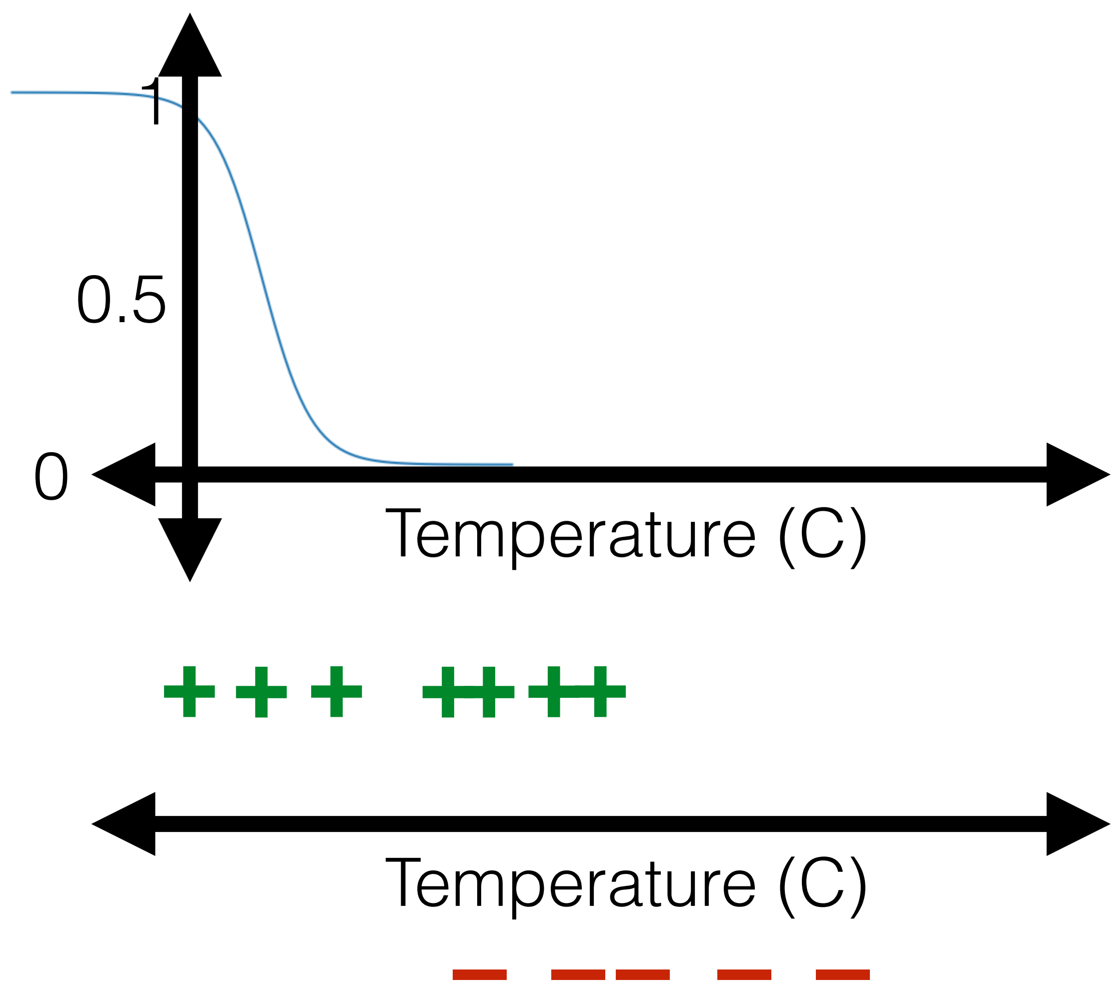


Linear logistic classification

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

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

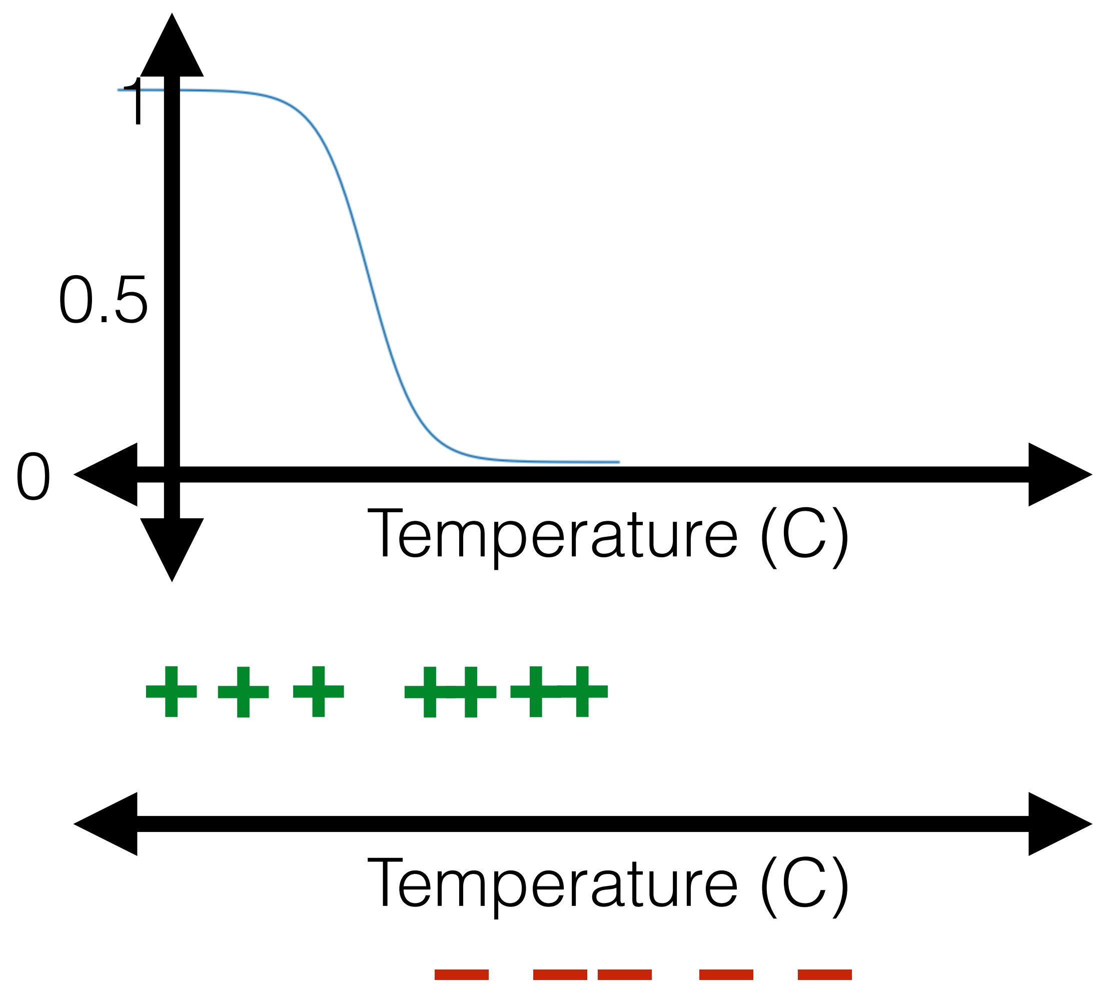


Linear logistic classification

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

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

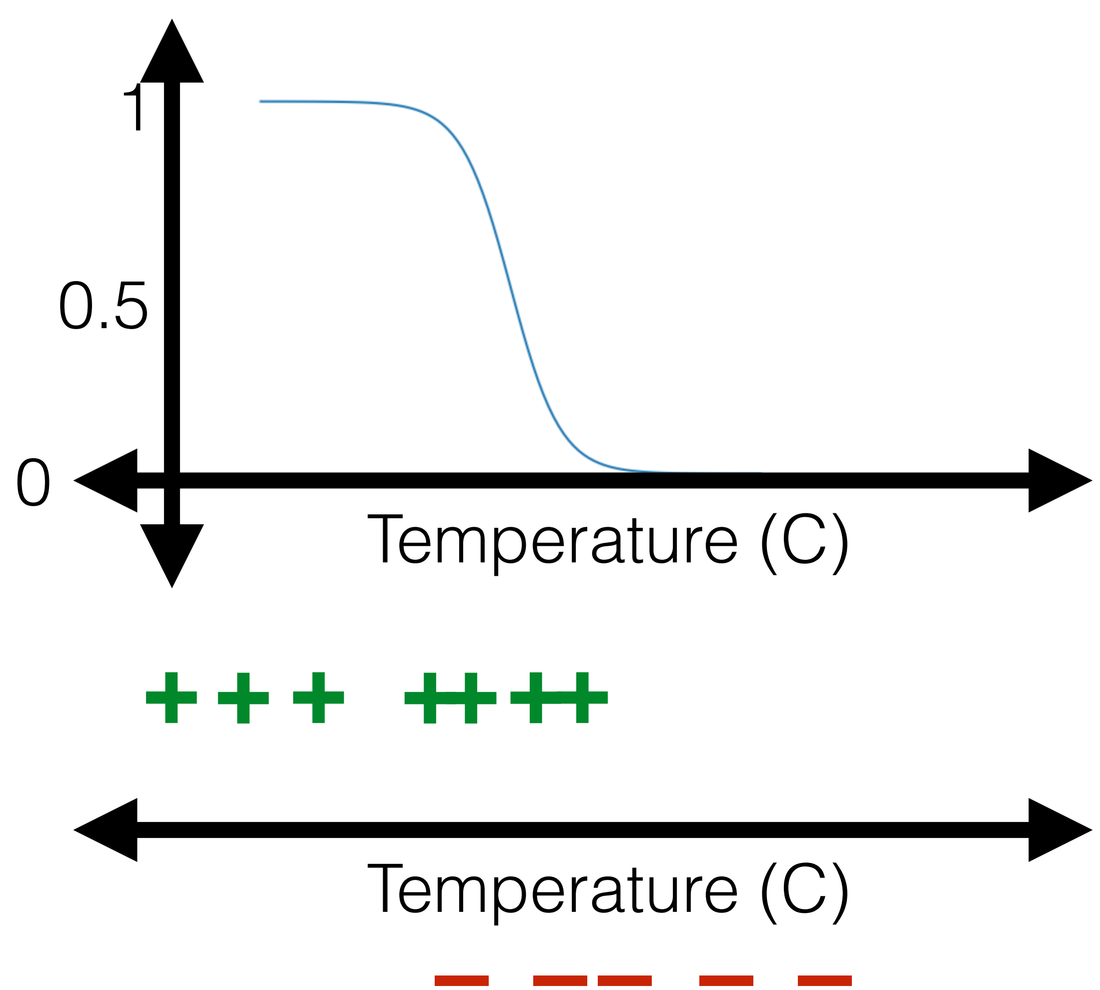


Linear logistic classification

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

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

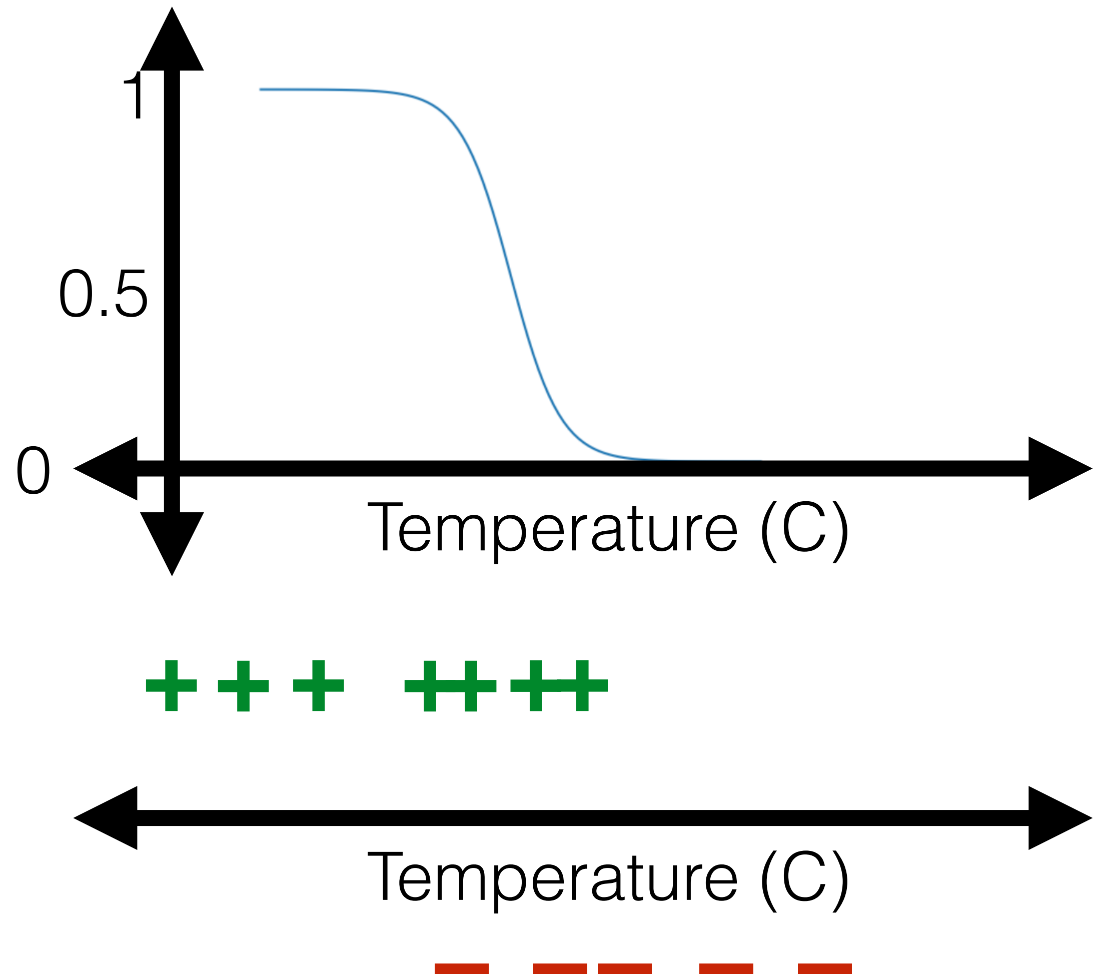


Linear logistic classification

aka logistic regression

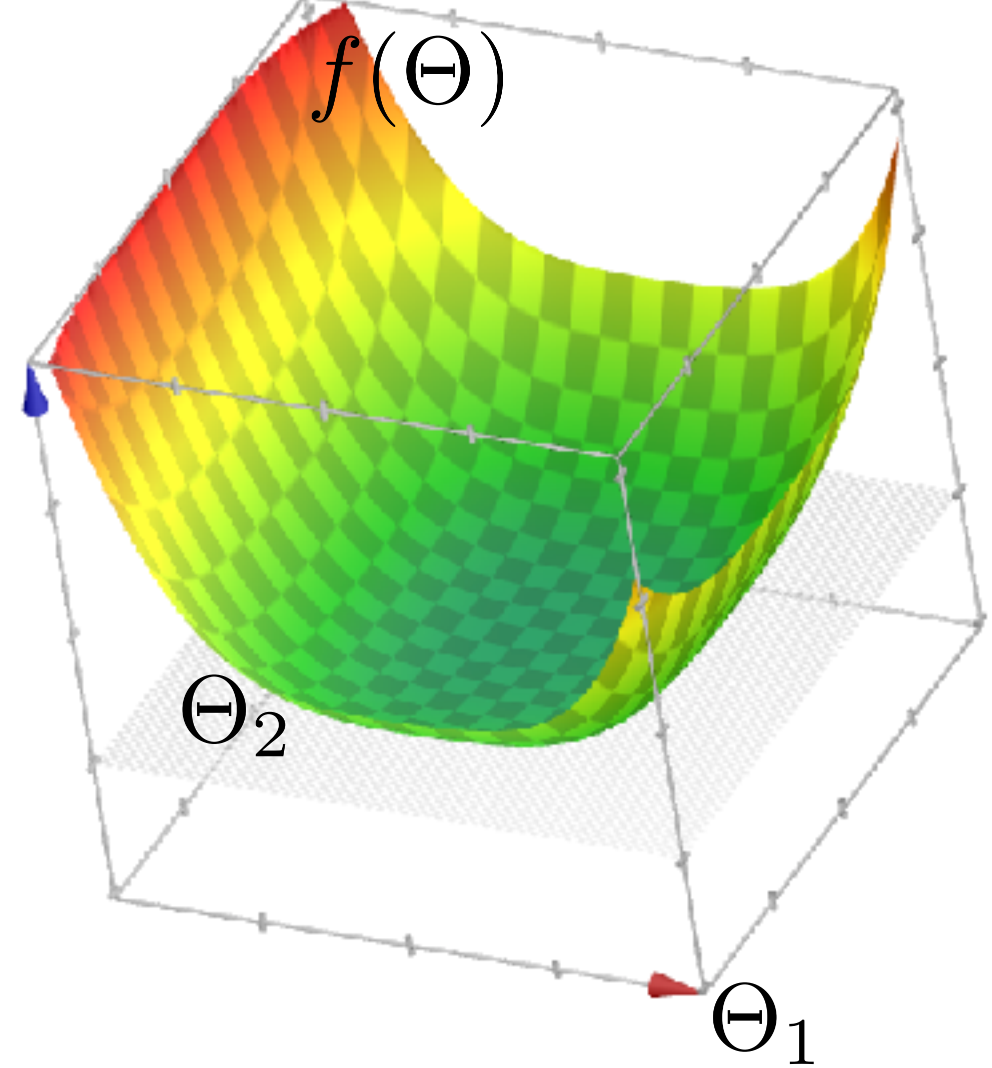
- How do we learn a classifier (i.e. learn θ, θ_0)?
- Want to find parameter values to minimize average (negative log likelihood) loss across the data

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

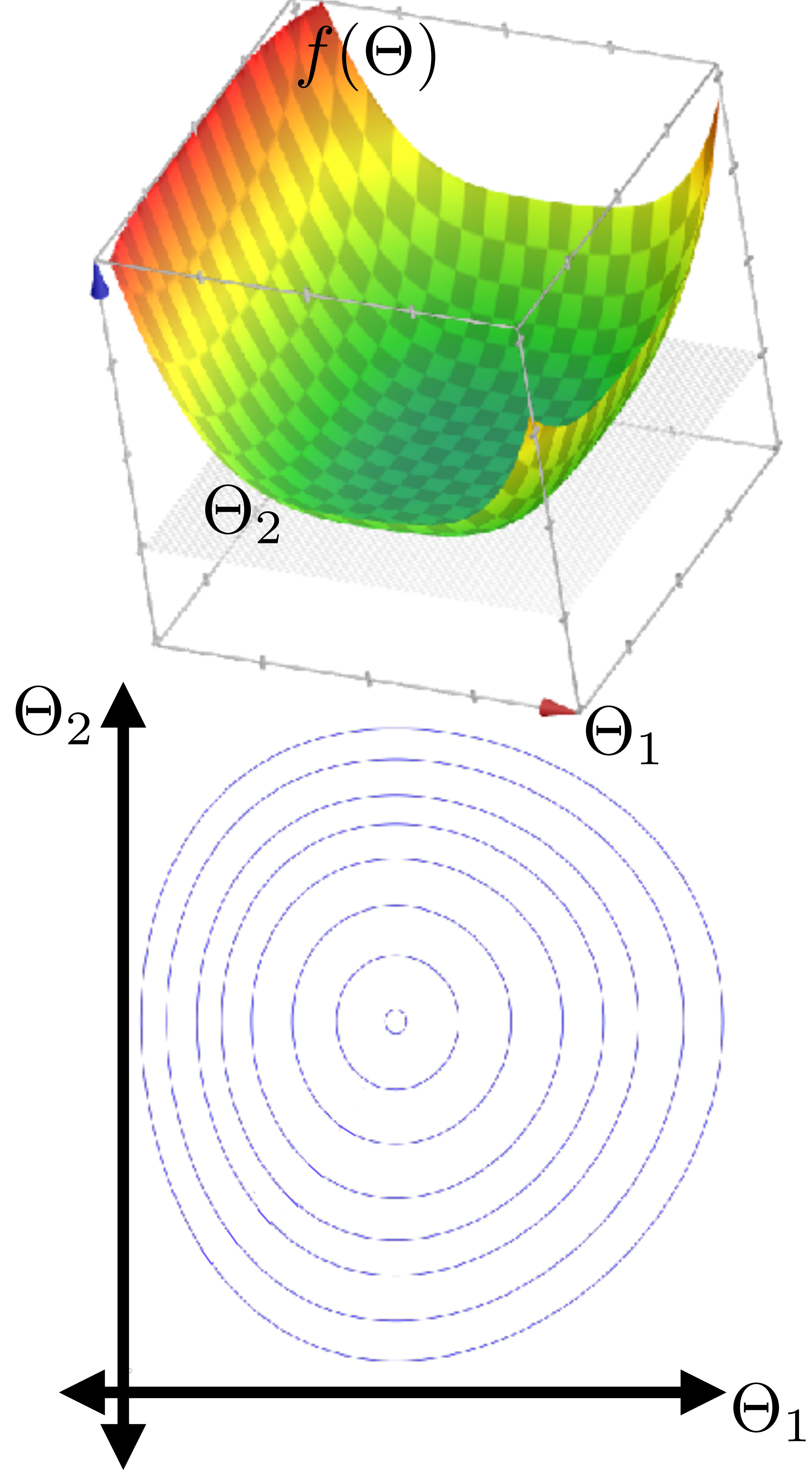


Gradient descent

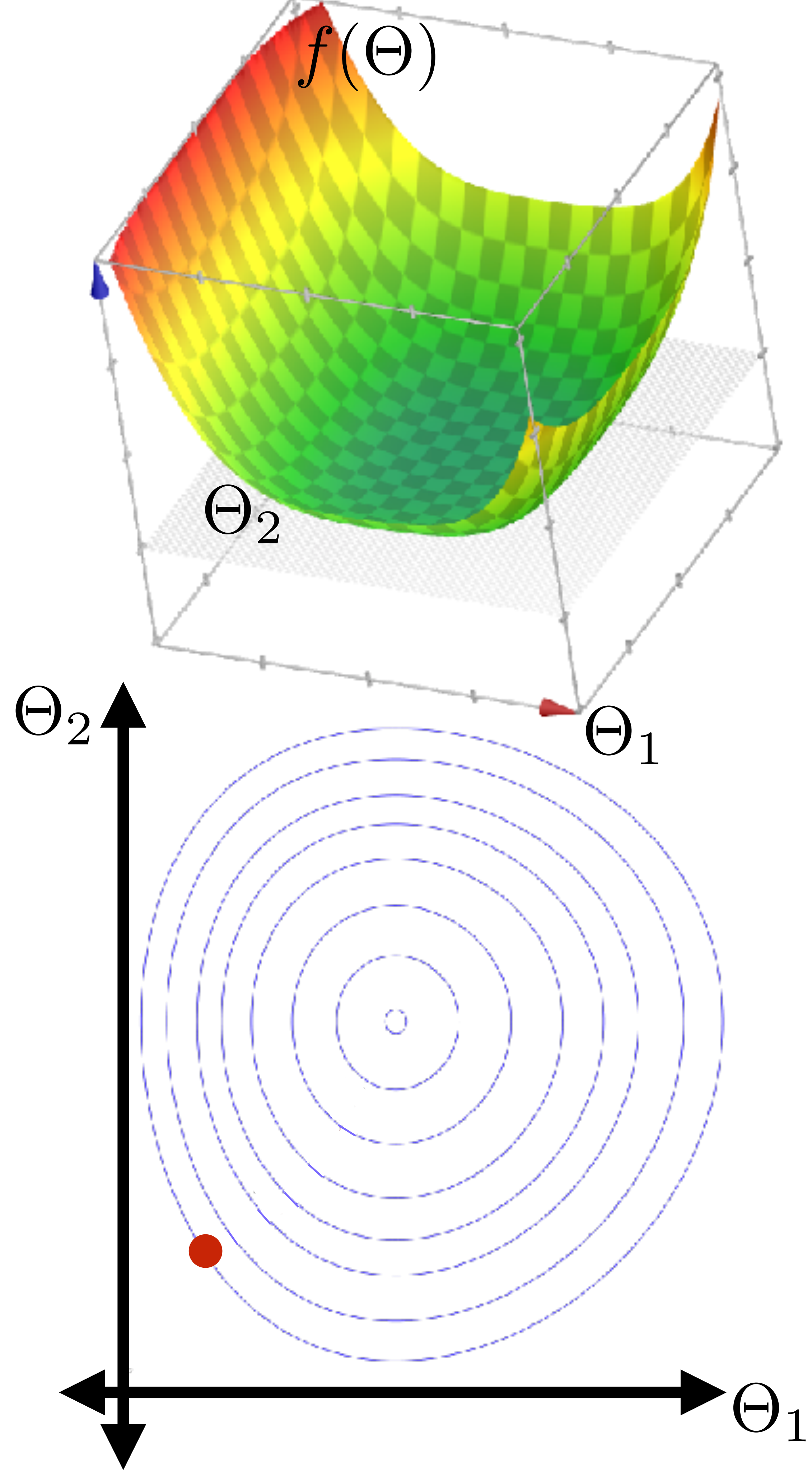
Gradient descent



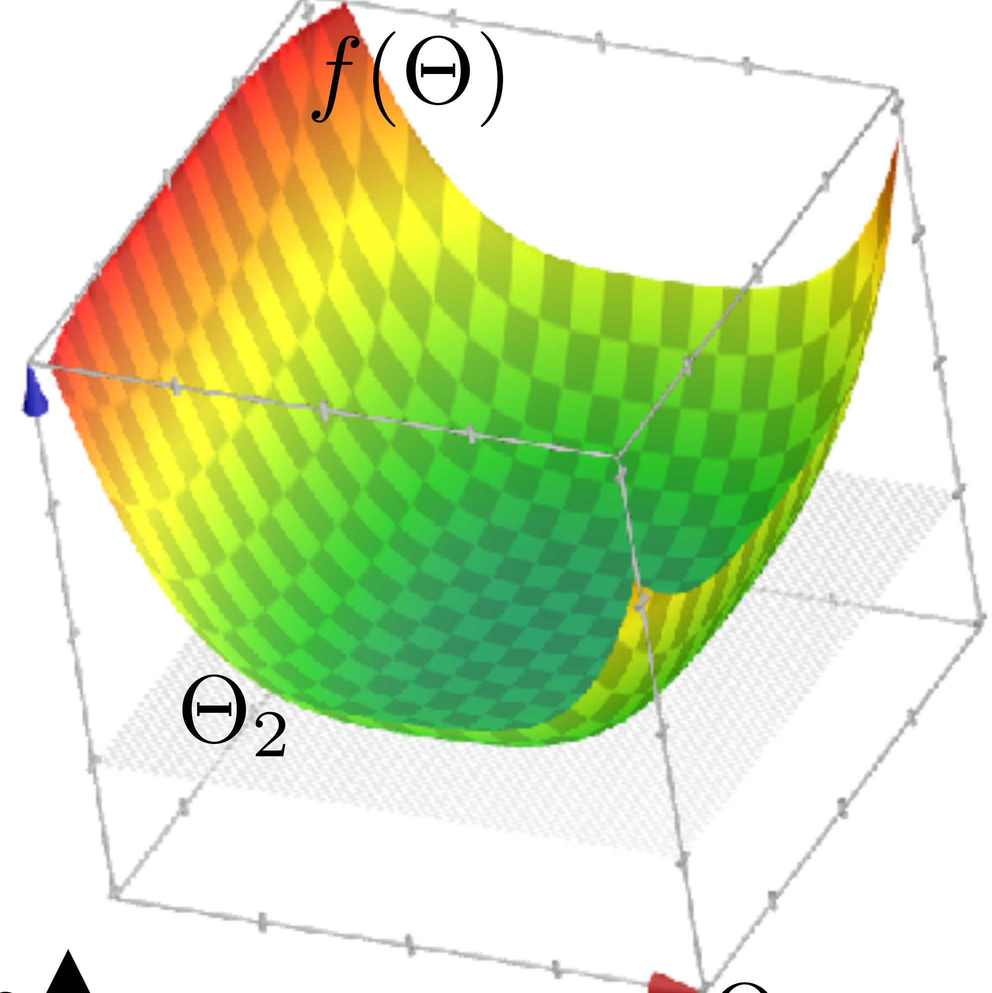
Gradient descent

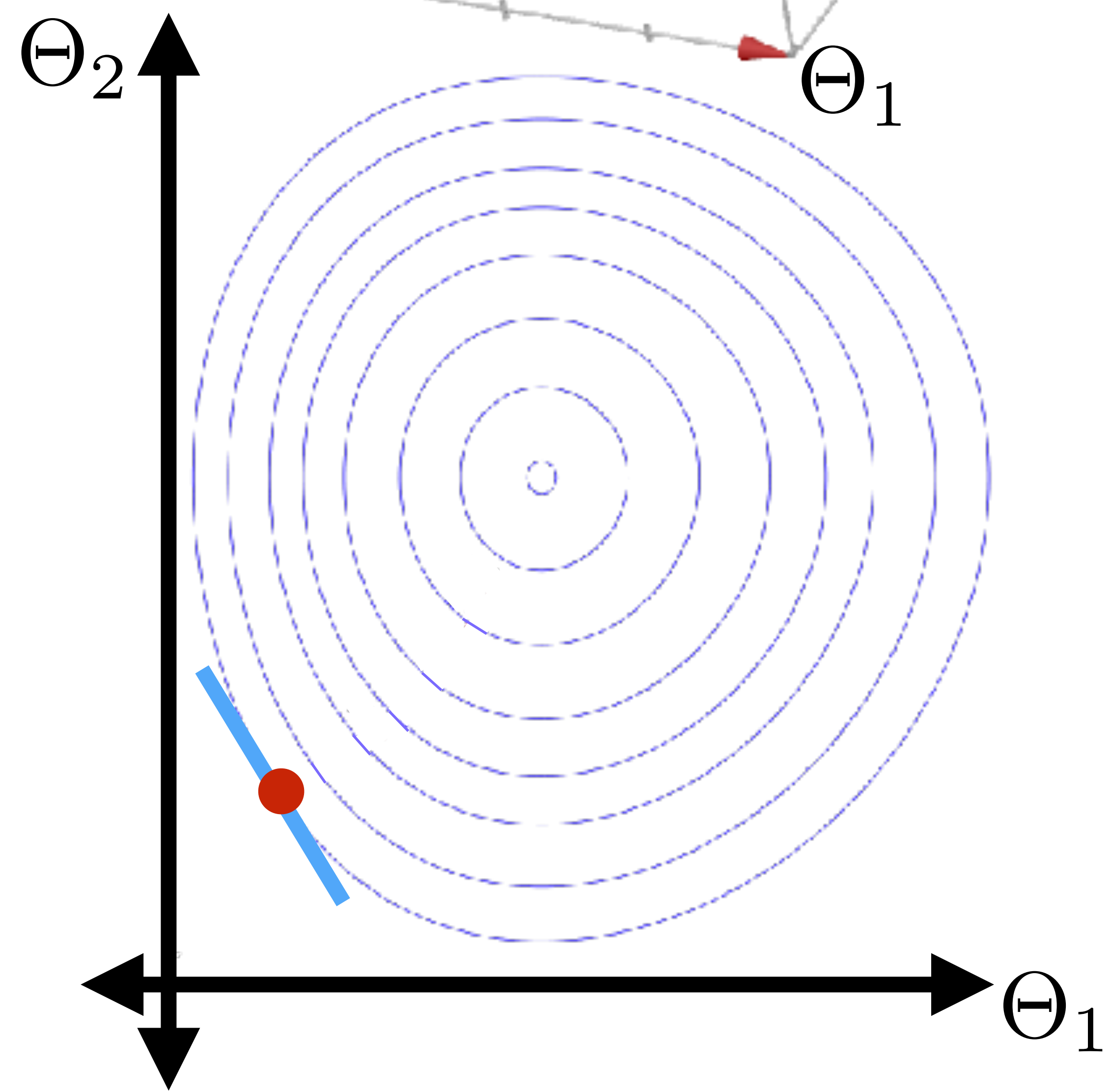


Gradient descent

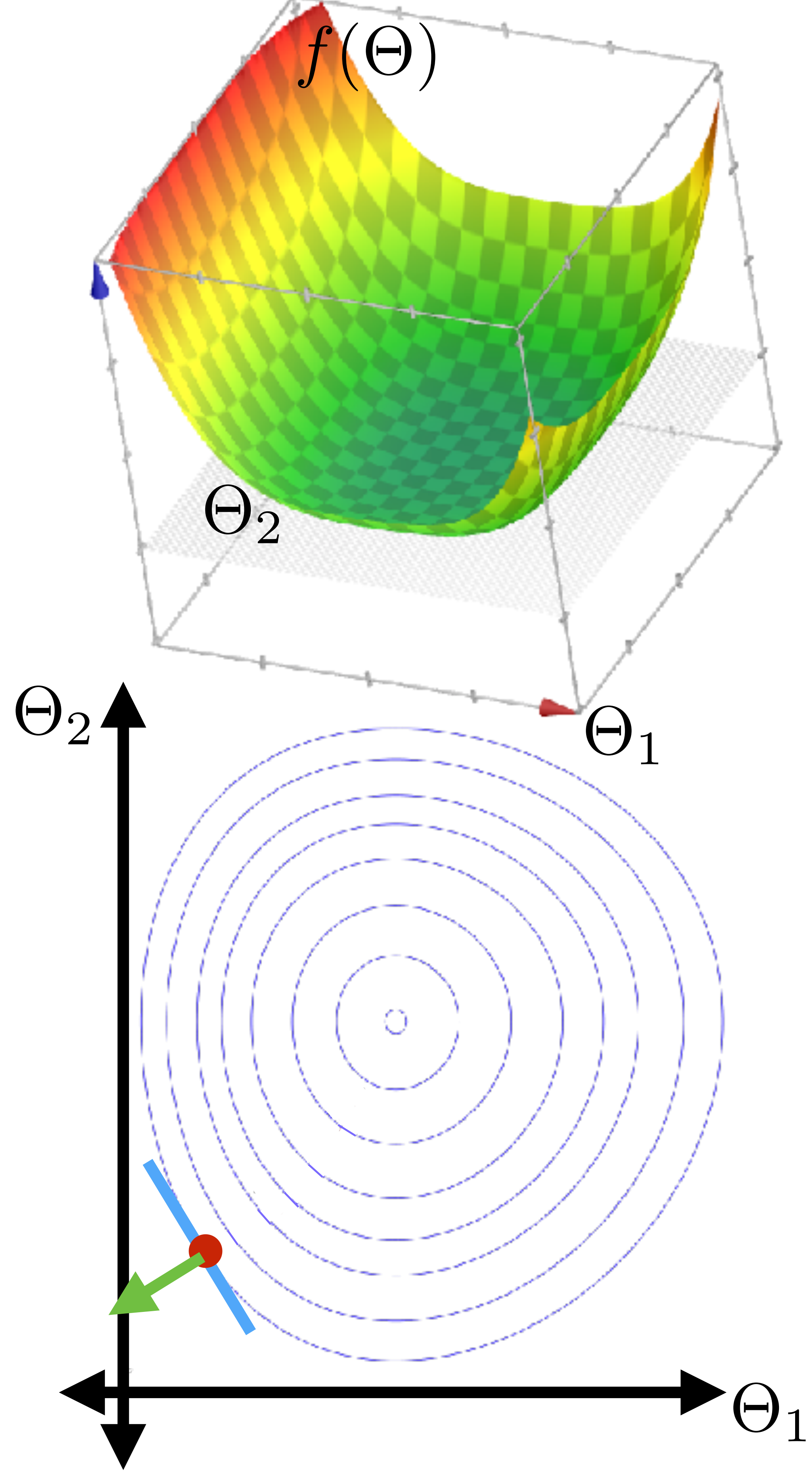


Gradient descent



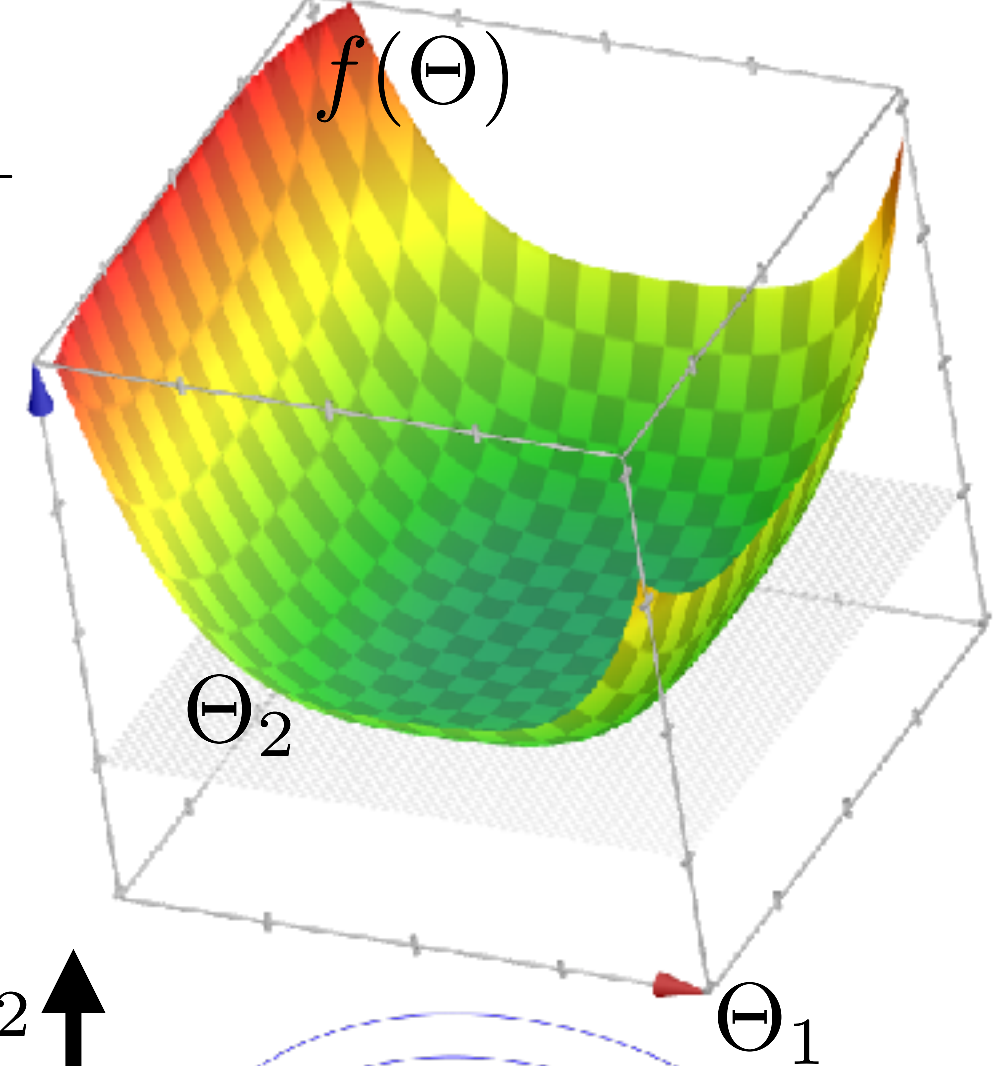


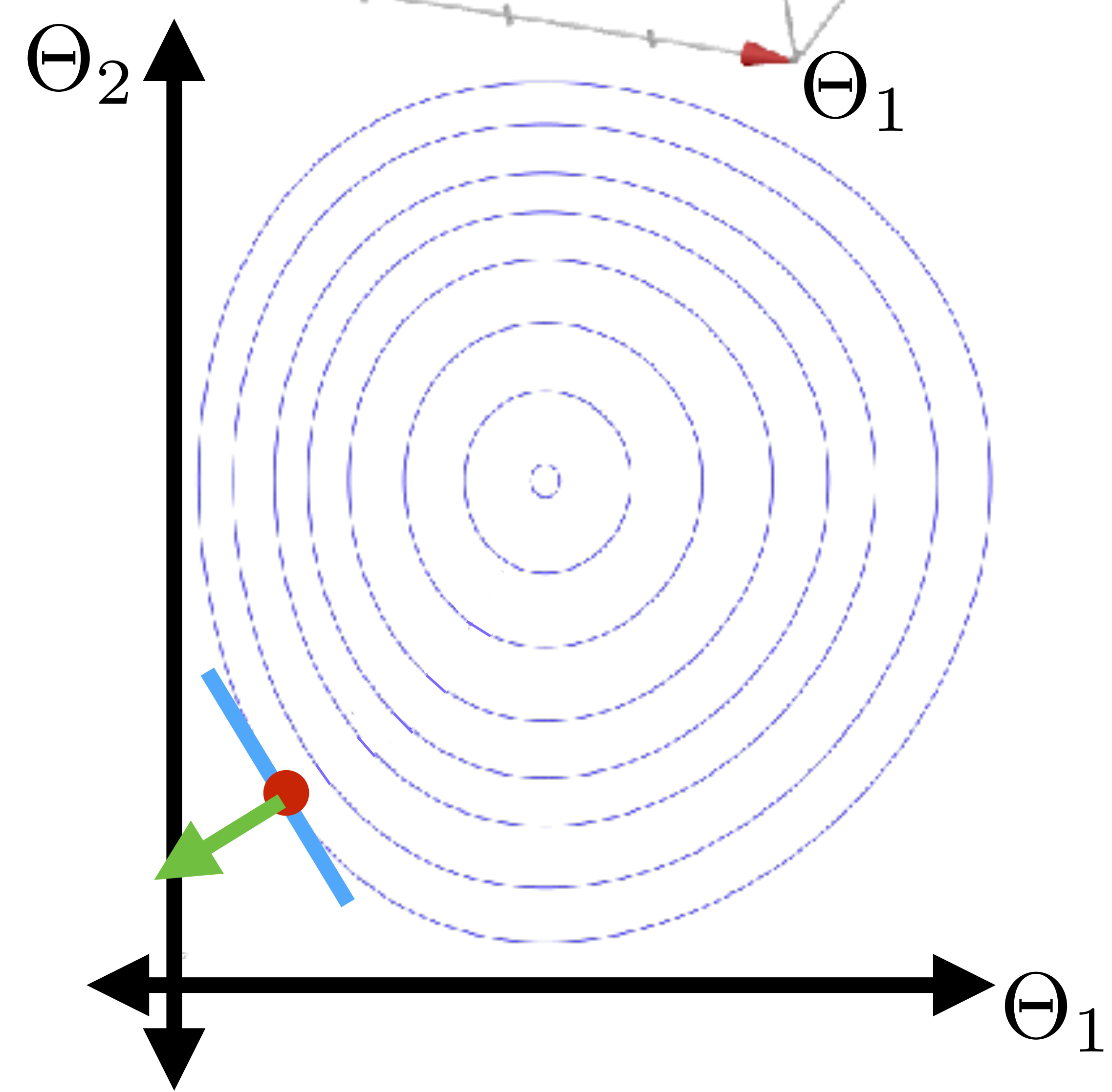
Gradient descent



Gradient descent

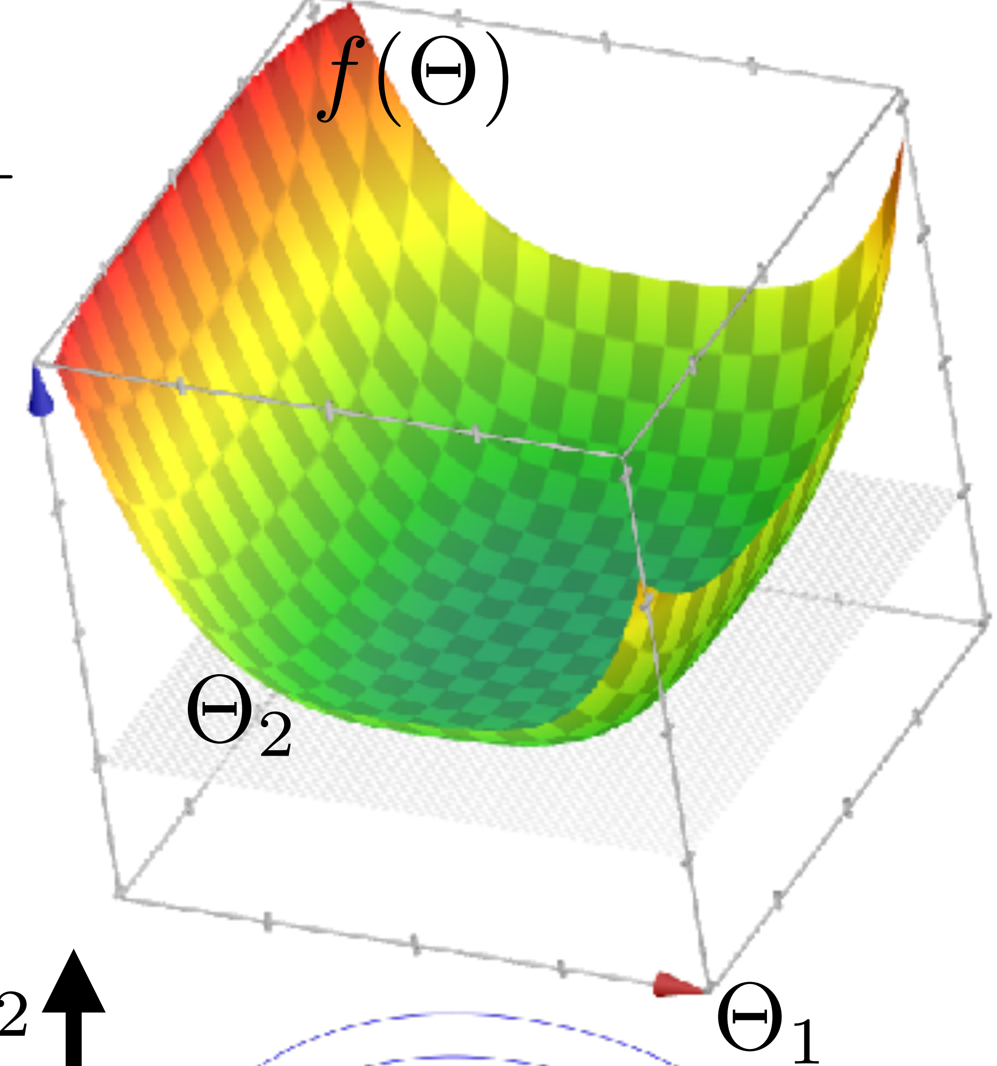
- Gradient $\nabla_\Theta f = \left[\frac{\partial f}{\partial \Theta_1}, \ldots, \frac{\partial f}{\partial \Theta_m} \right]^\top$
    - with $\Theta \in \mathbb{R}^m$

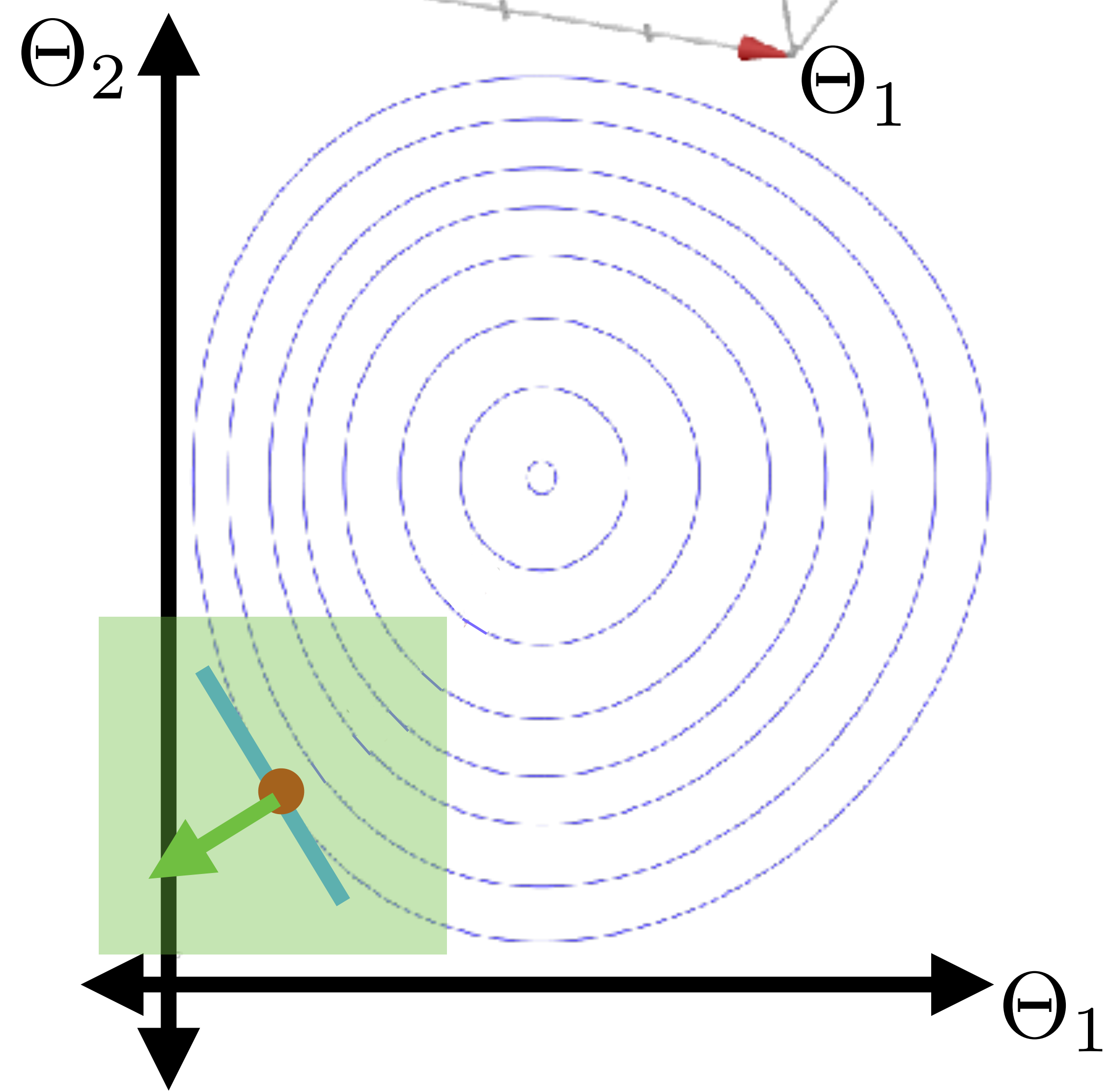




Gradient descent

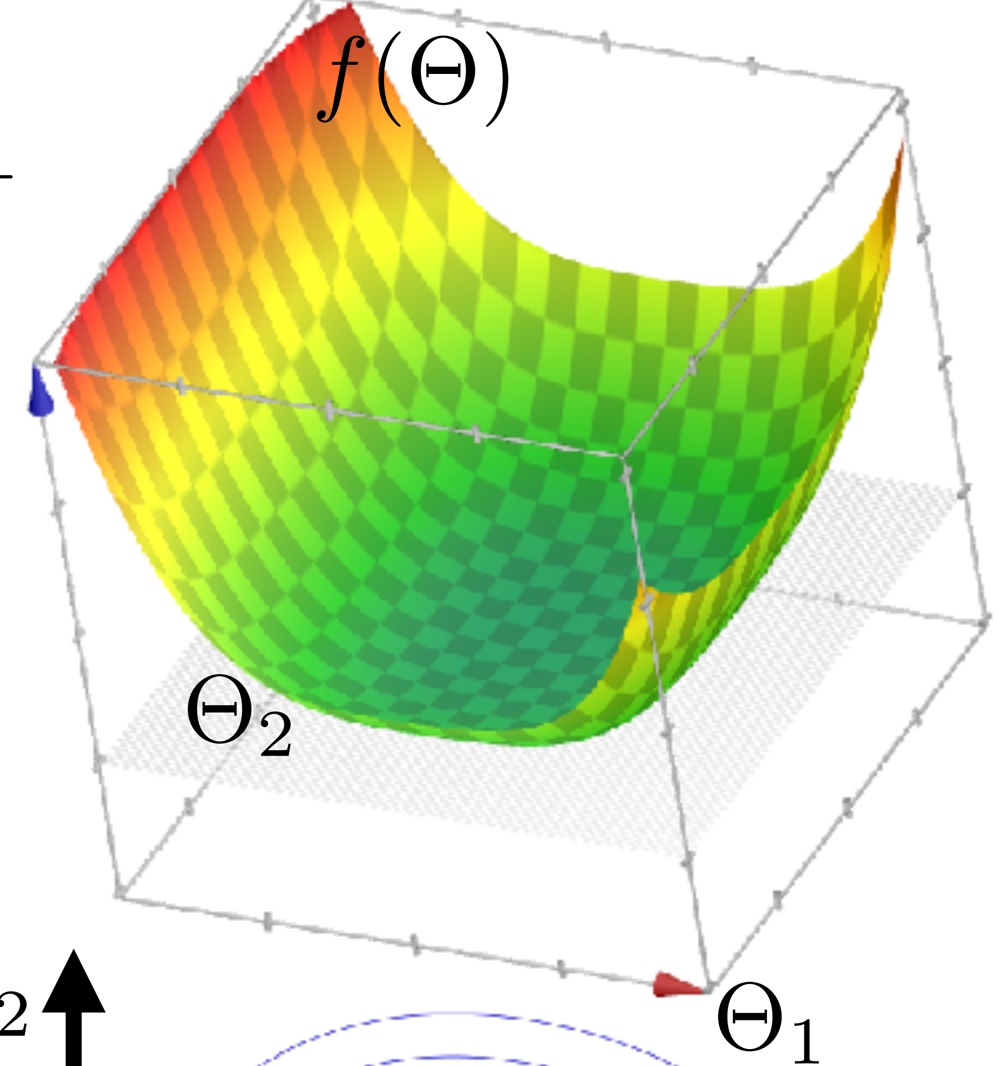
- Gradient $\nabla_\Theta f = \left[\frac{\partial f}{\partial \Theta_1}, \ldots, \frac{\partial f}{\partial \Theta_m}\right]^\top$
    - with $\Theta \in \mathbb{R}^m$

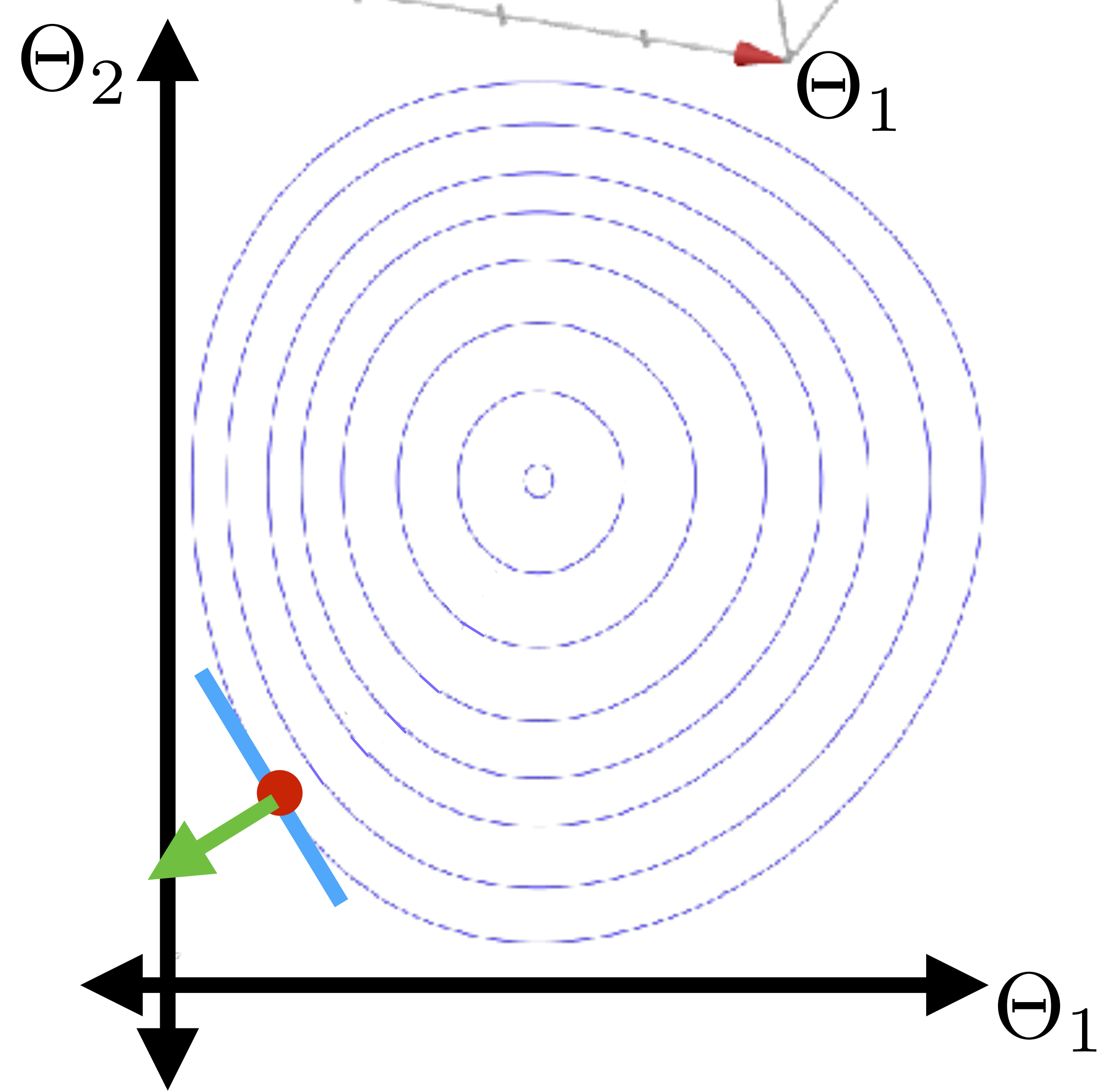




Gradient descent

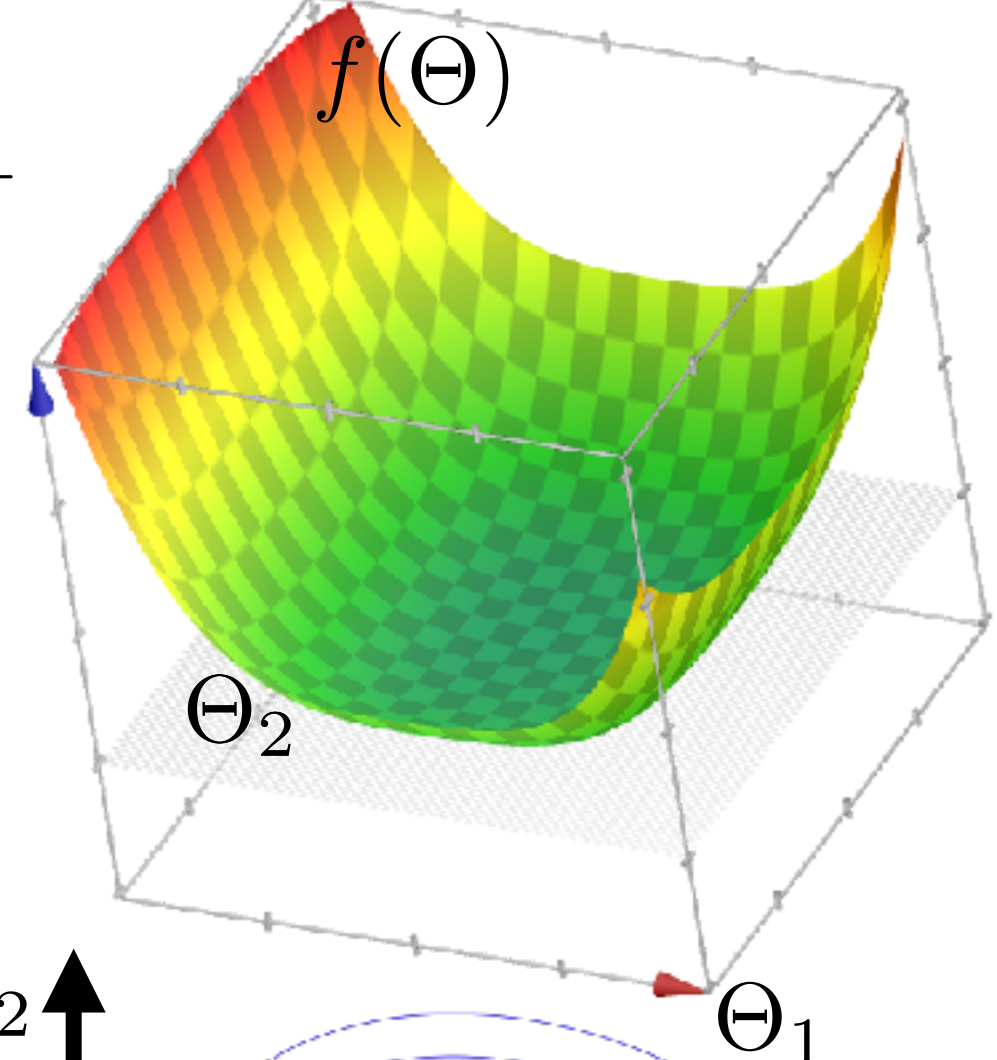
- Gradient $\nabla_\Theta f = \left[\frac{\partial f}{\partial \Theta_1}, \ldots, \frac{\partial f}{\partial \Theta_m}\right]^\top$
  - with $\Theta \in \mathbb{R}^m$

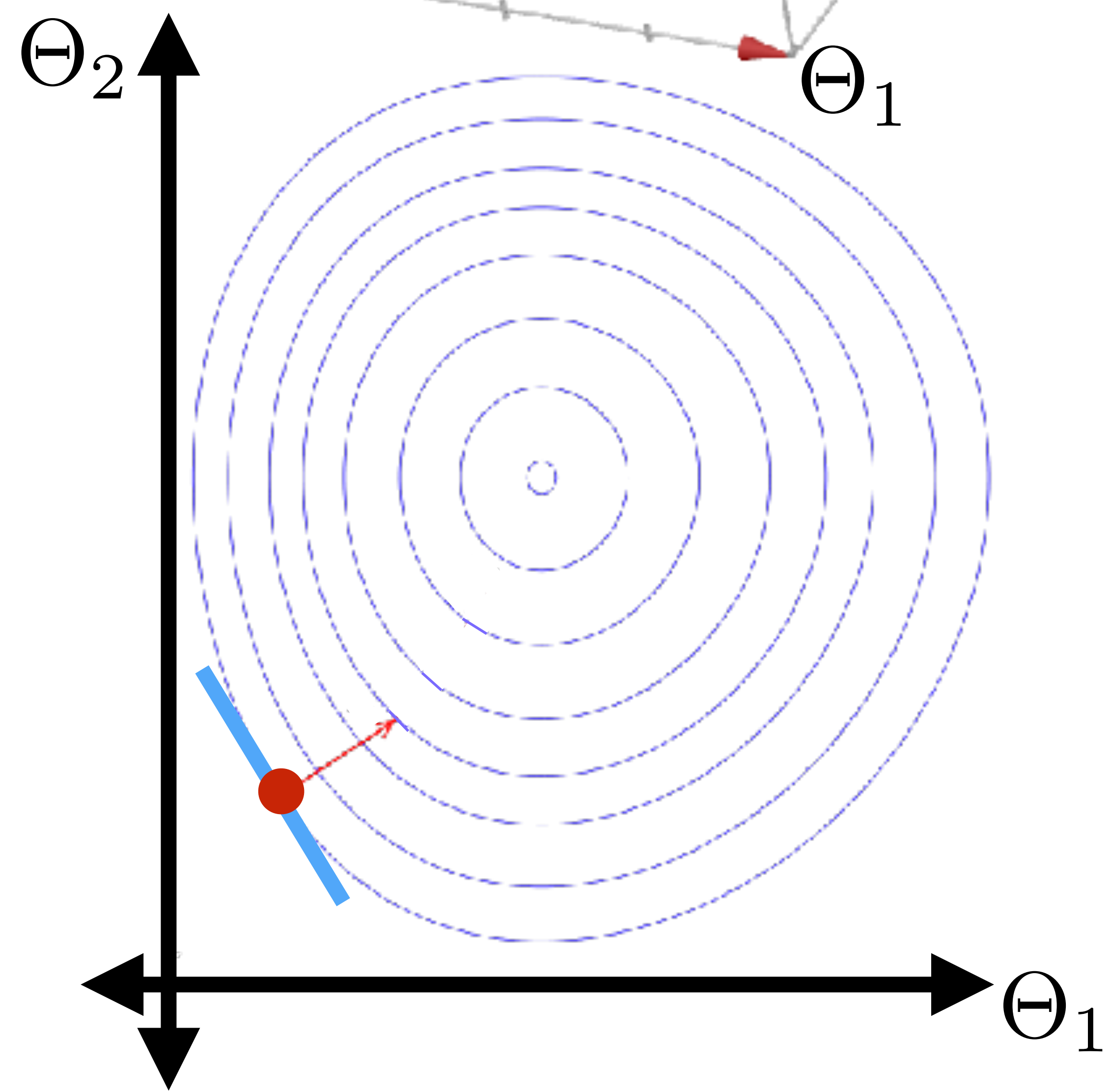




Gradient descent

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

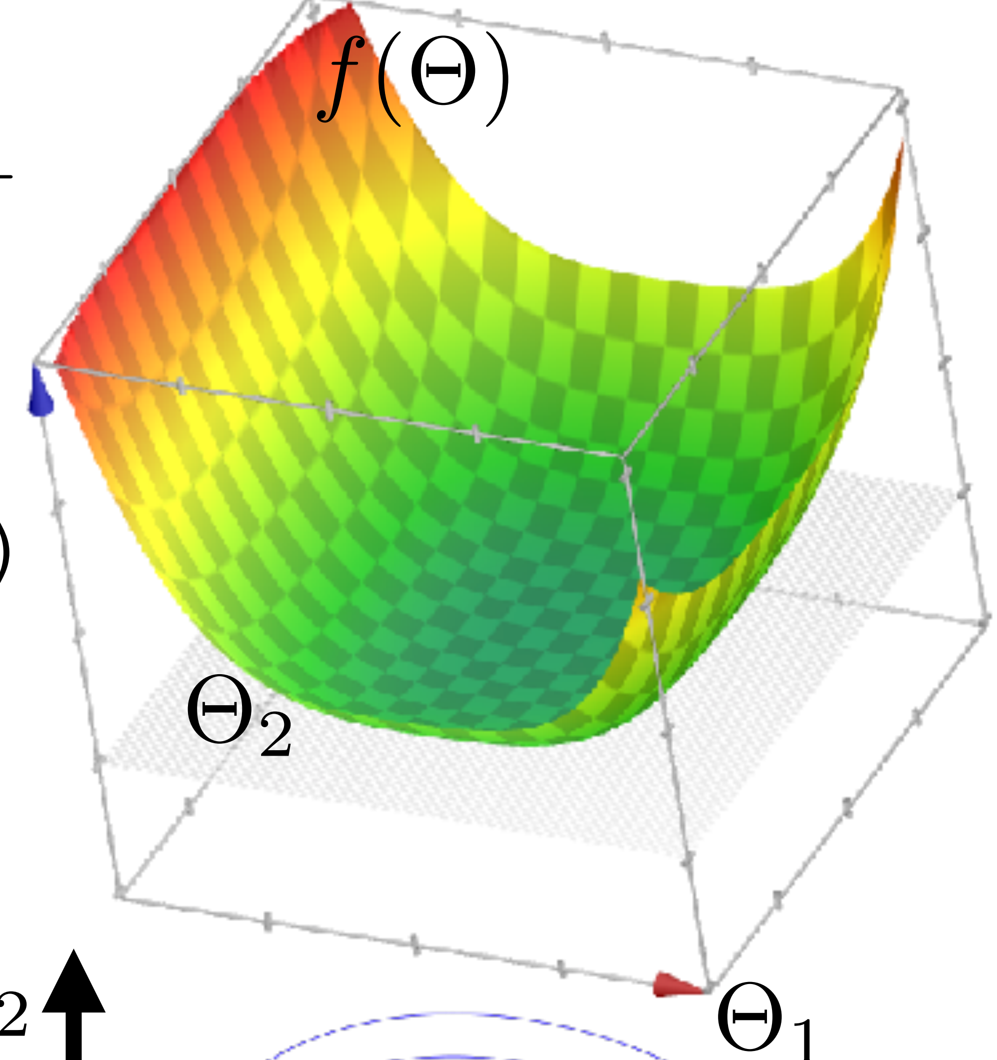


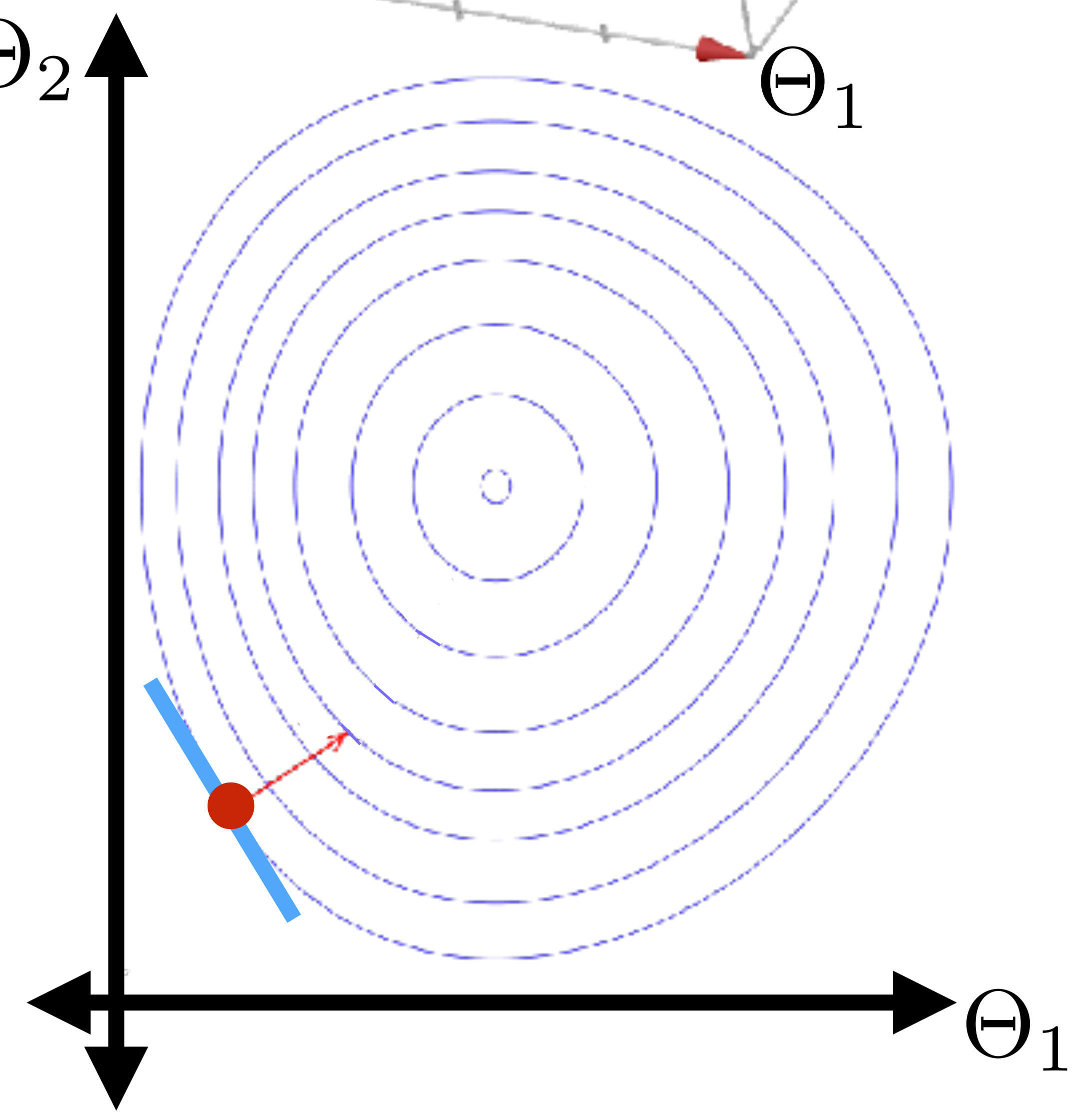


Gradient descent

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



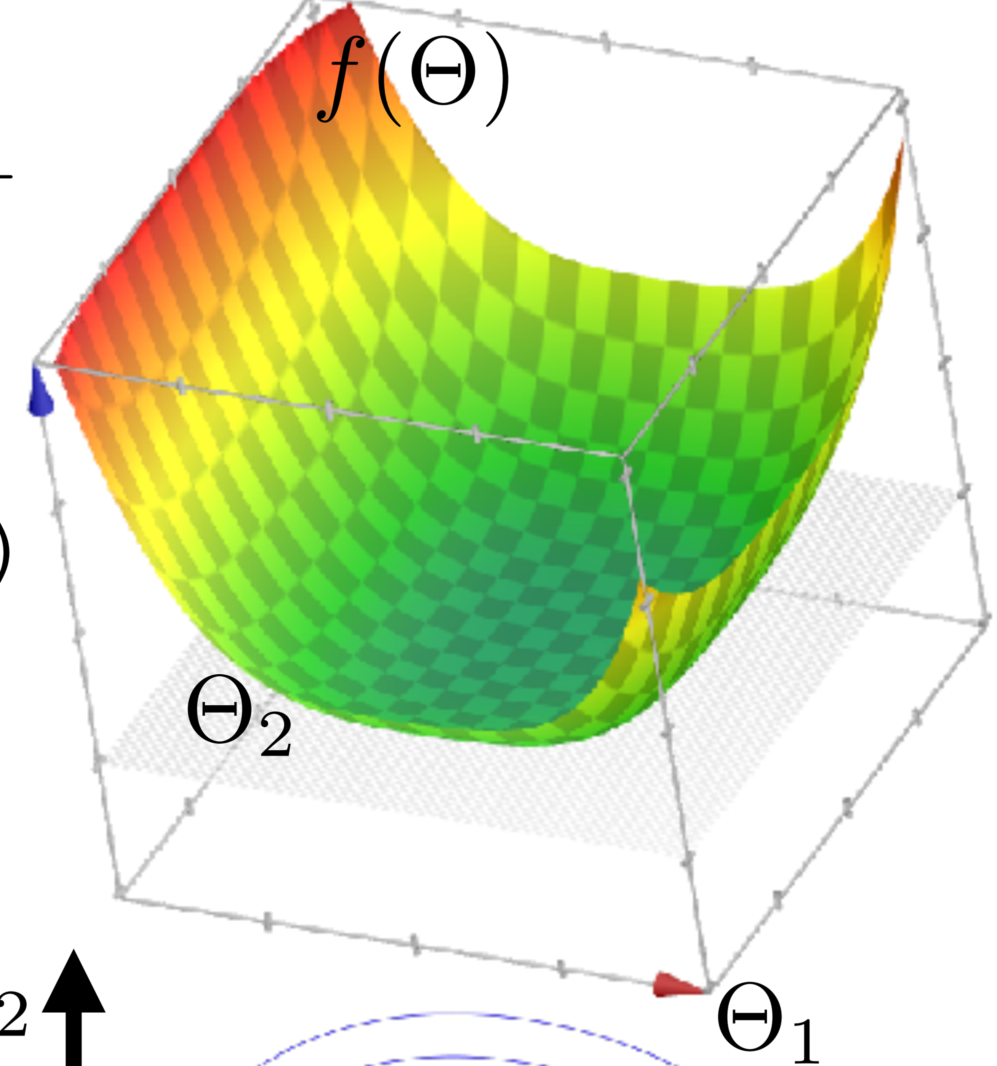


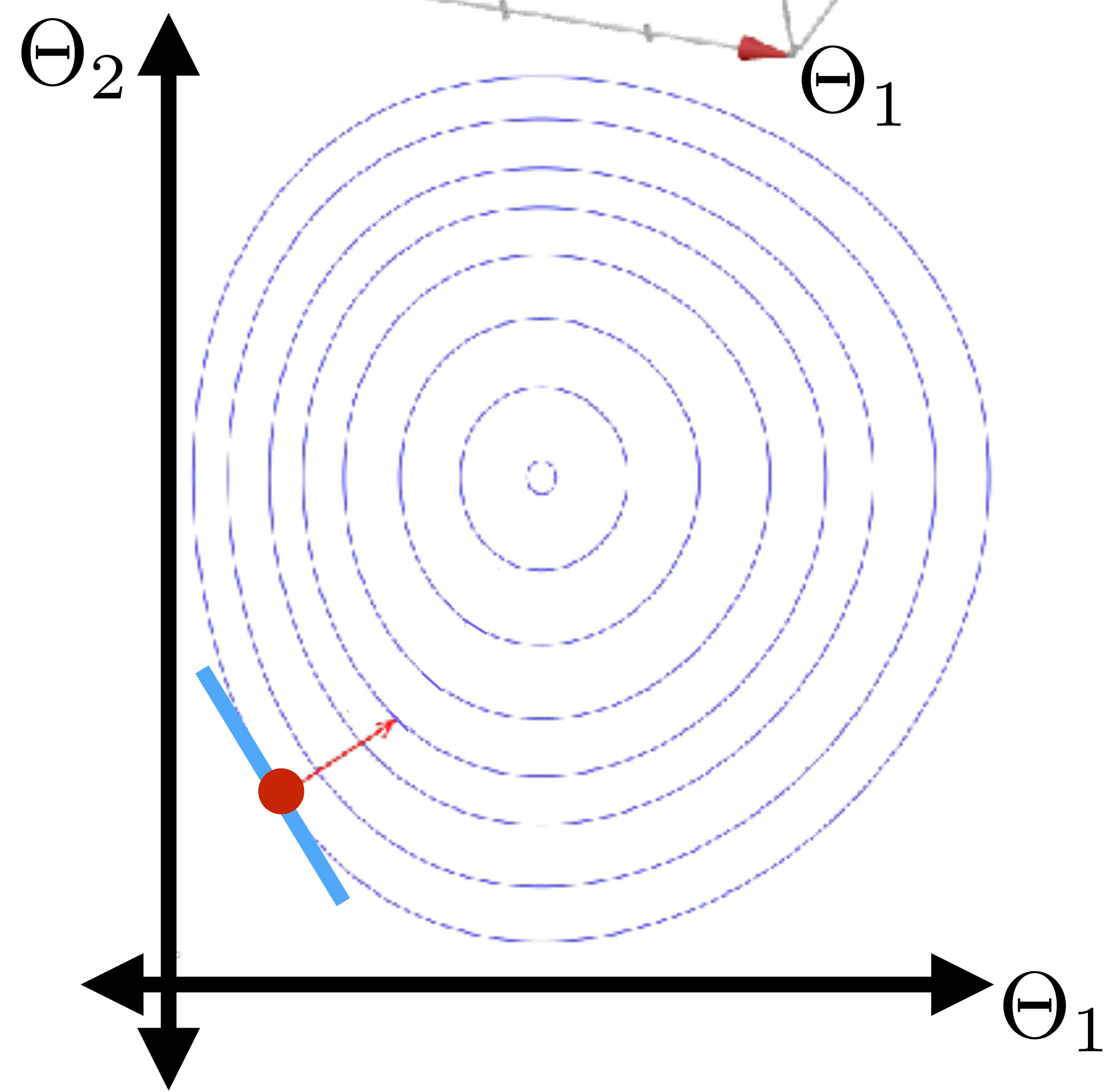
Gradient descent

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

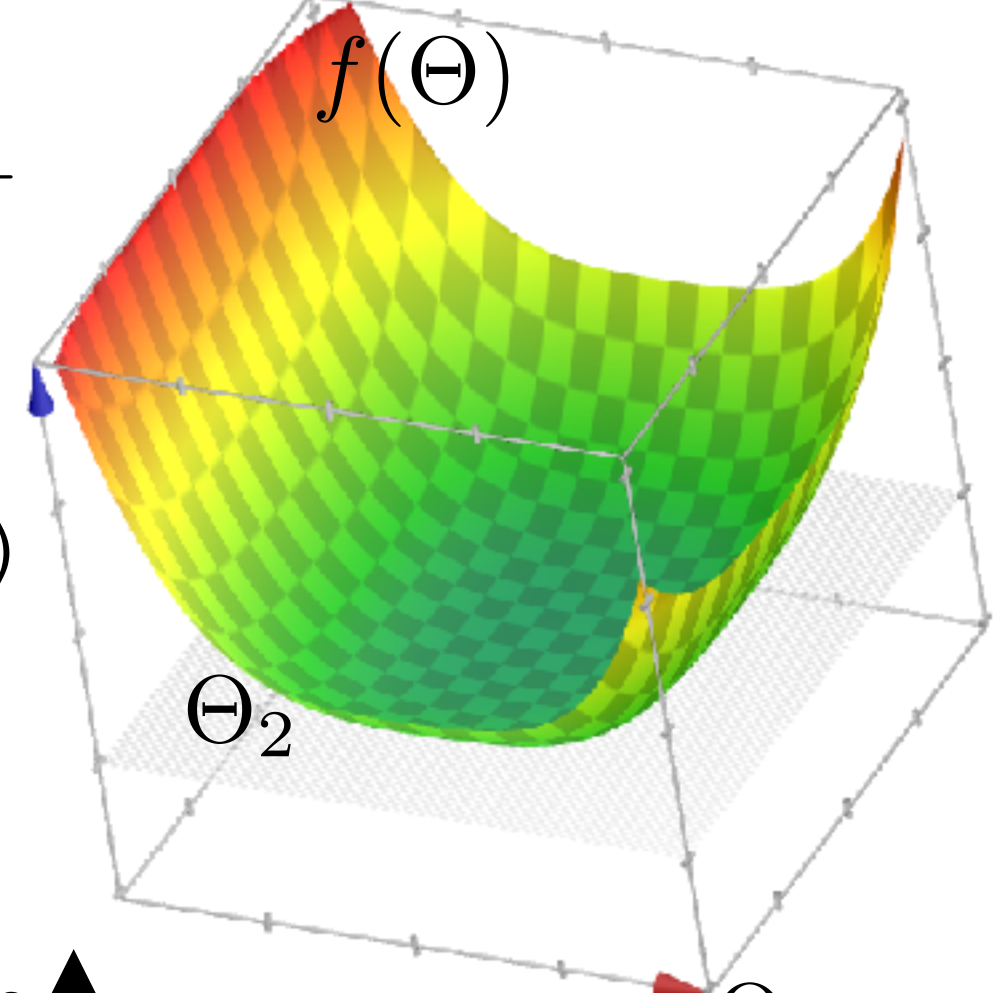
`Initialize` $\Theta^{(0)} = \Theta_{\text{init}}$





Gradient descent

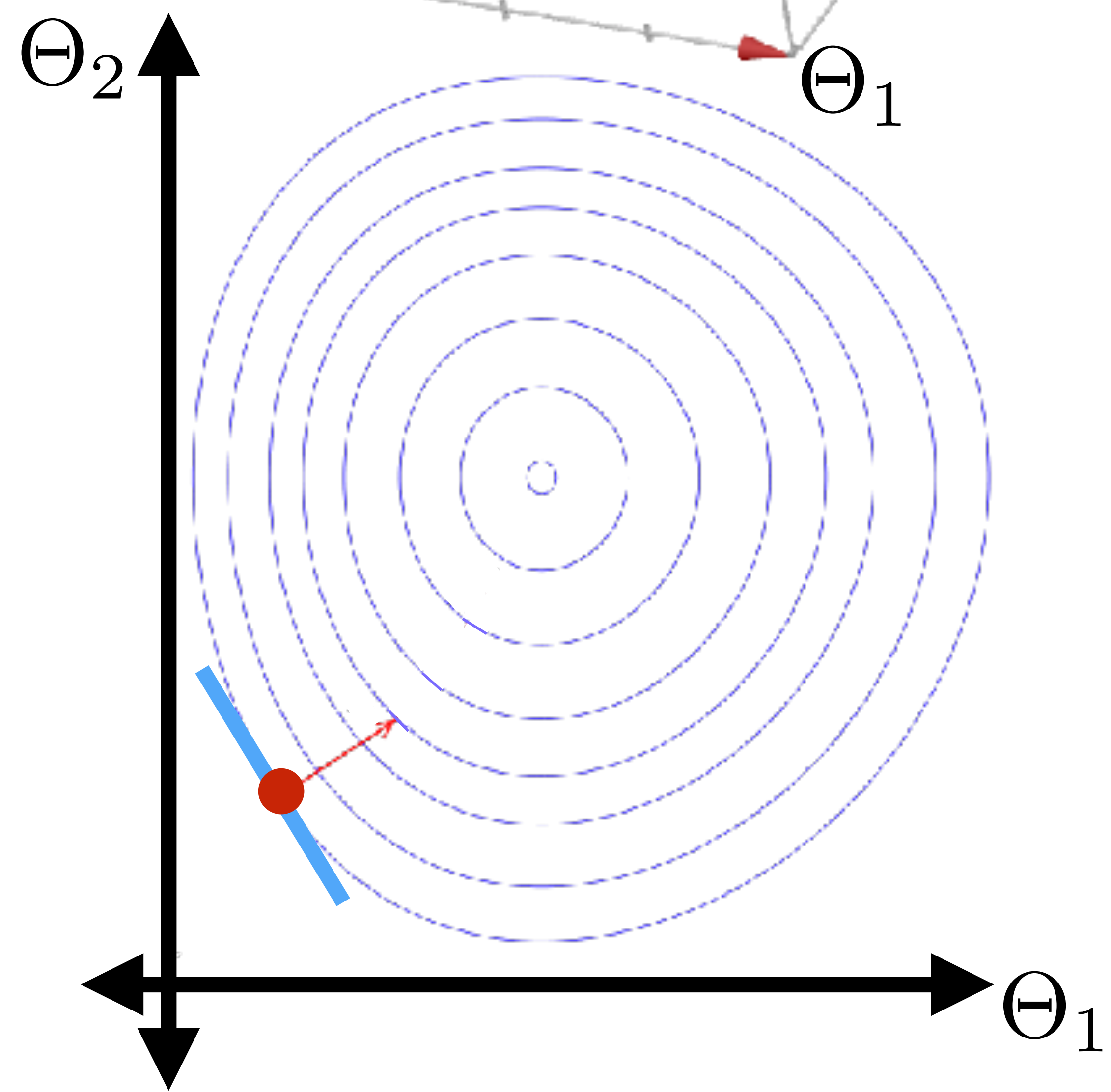
- Gradient $\nabla_{\Theta} f = \left[\frac{\partial f}{\partial \Theta_1}, \ldots, \frac{\partial f}{\partial \Theta_m} \right]^\top$
  - with $\Theta \in \mathbb{R}^m$



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

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

  Initialize t = 0



Gradient descent

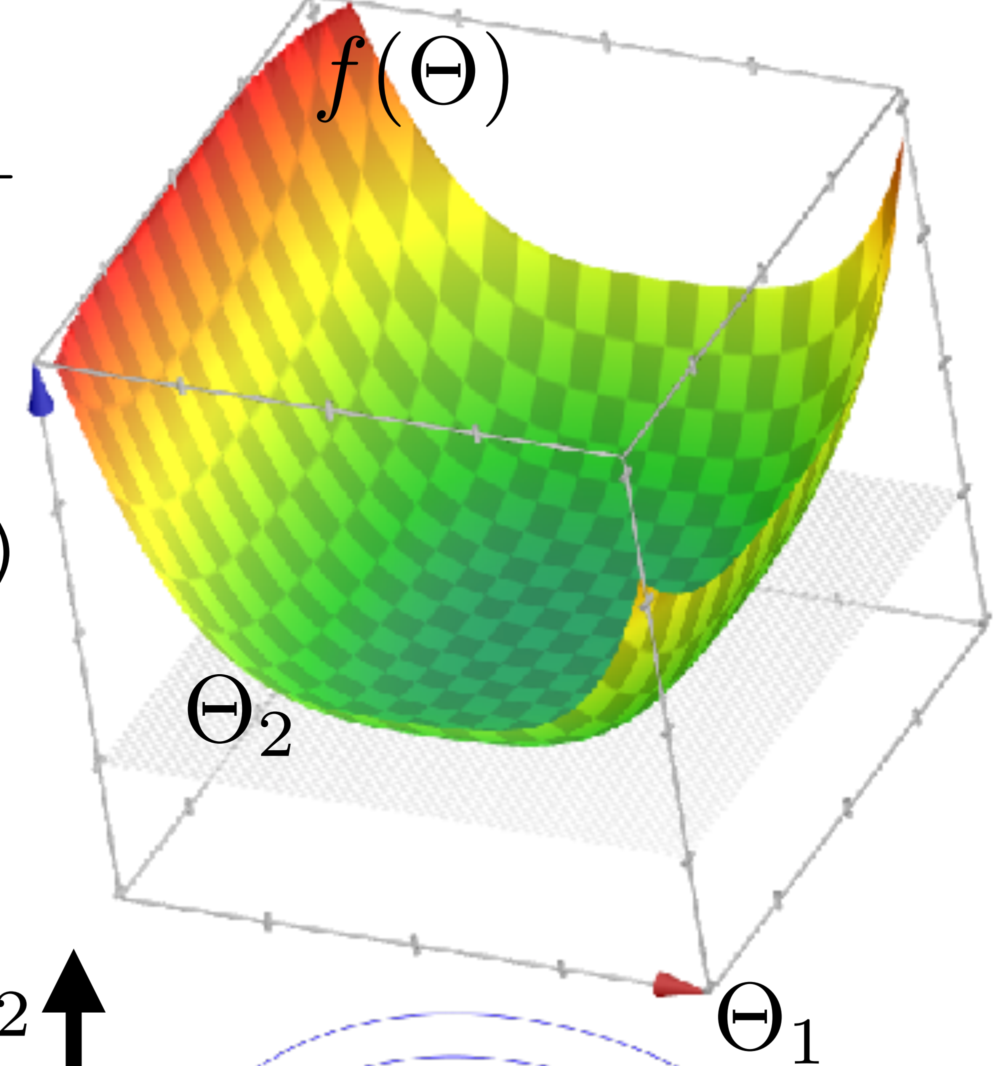
- Gradient $\nabla_\Theta f = \left[\frac{\partial f}{\partial \Theta_1}, \ldots, \frac{\partial f}{\partial \Theta_m} \right]^\top$
  - with $\Theta \in \mathbb{R}^m$

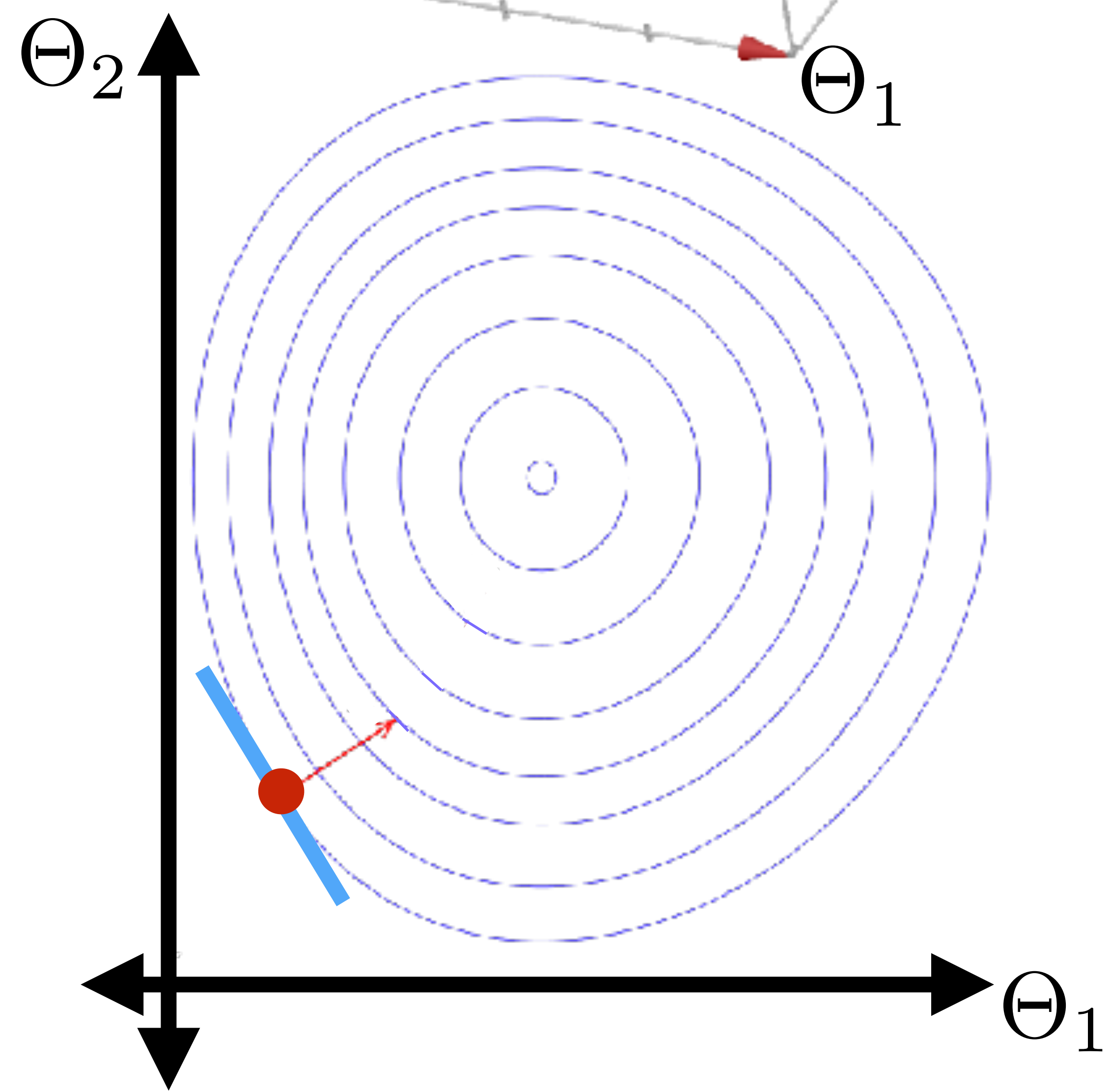
Gradient-Descent $(\Theta_{\text{init}}, \eta, f, \nabla_\Theta f, \epsilon)$

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

  Initialize t = 0

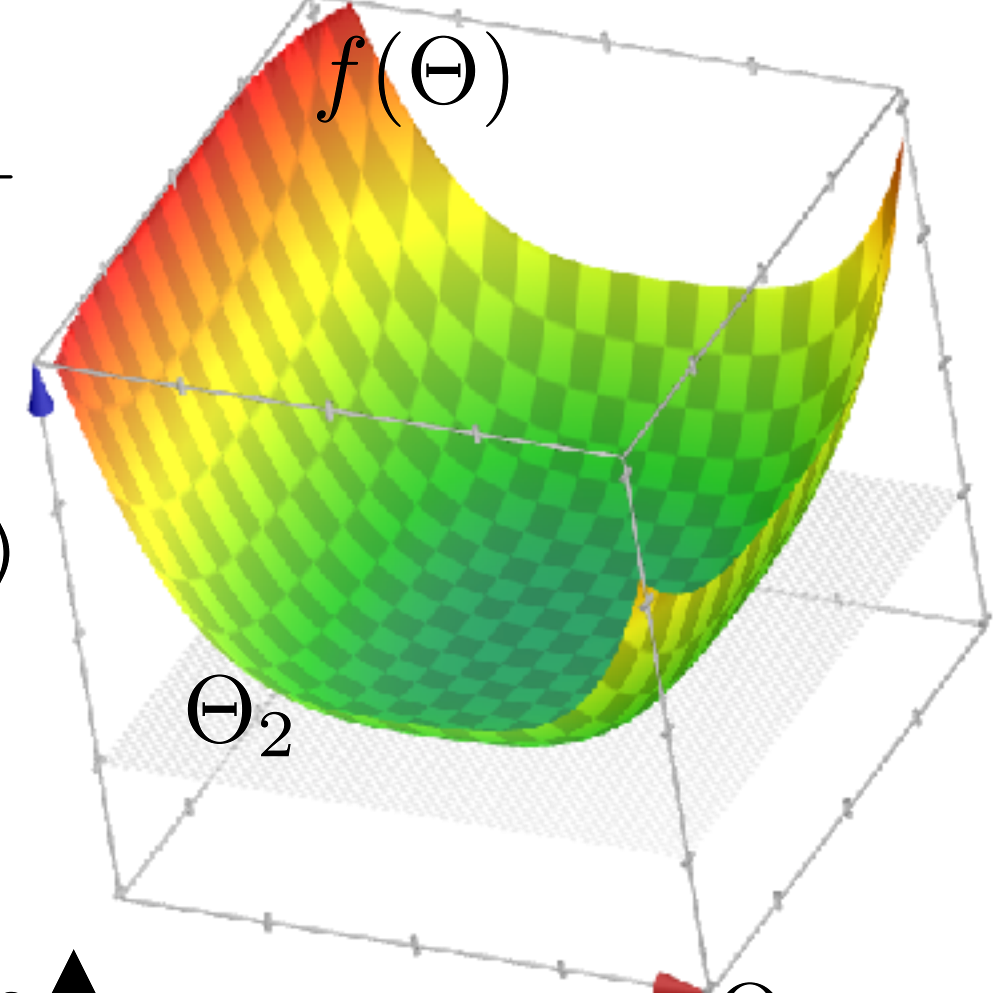
  repeat





Gradient descent

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



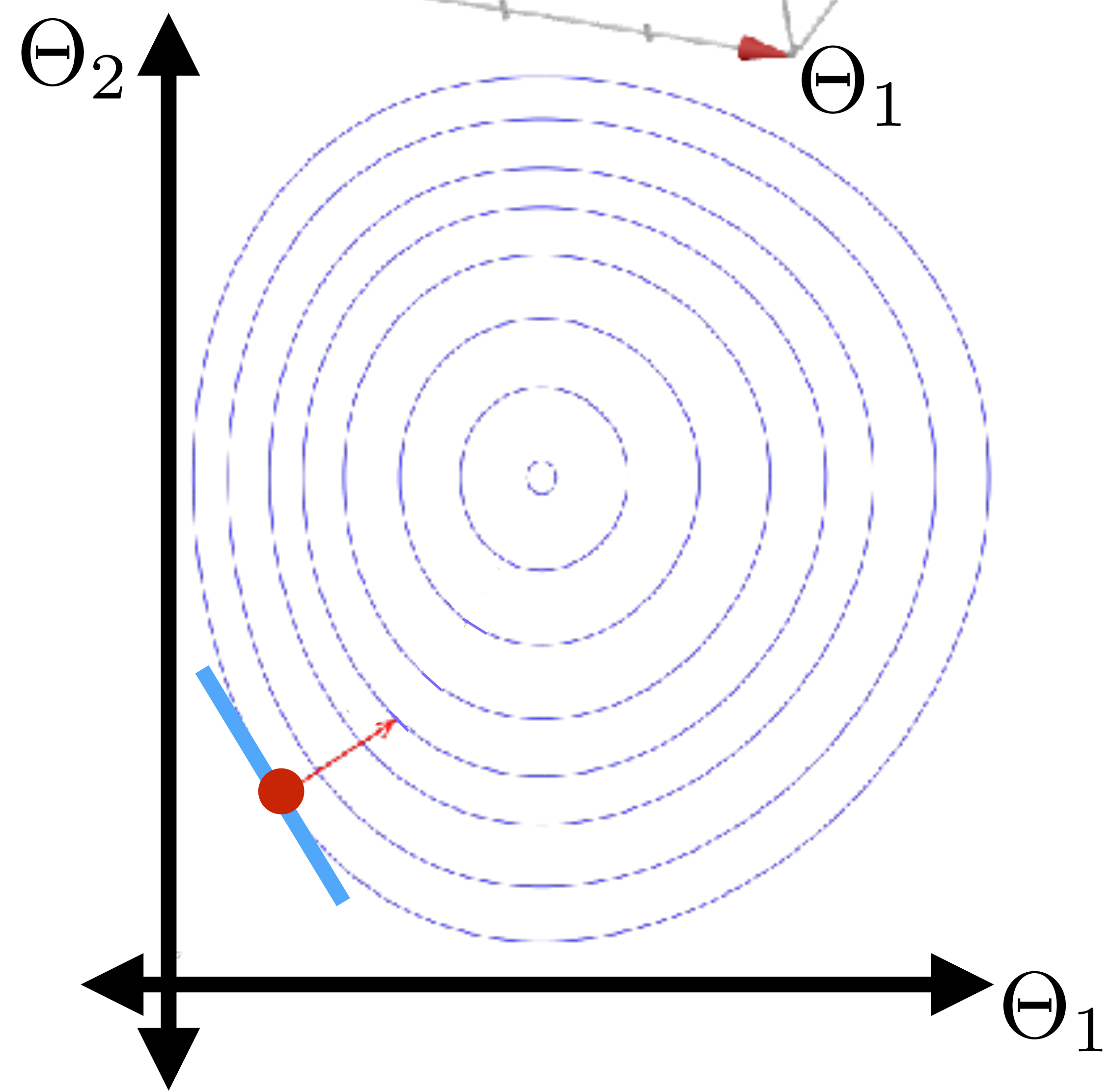
Gradient-Descent $(\Theta_{\text{init}}, \eta, f, \nabla_\Theta f, \epsilon)$

  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 (Θ_init, η, f, ∇_Θ f, ε)
  Initialize Θ^(0) = Θ_init
  Initialize t = 0
  repeat
    t = t + 1
    Θ^(t) = Θ^(t-1) − η ∇_Θ f(Θ^(t-1))
```

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

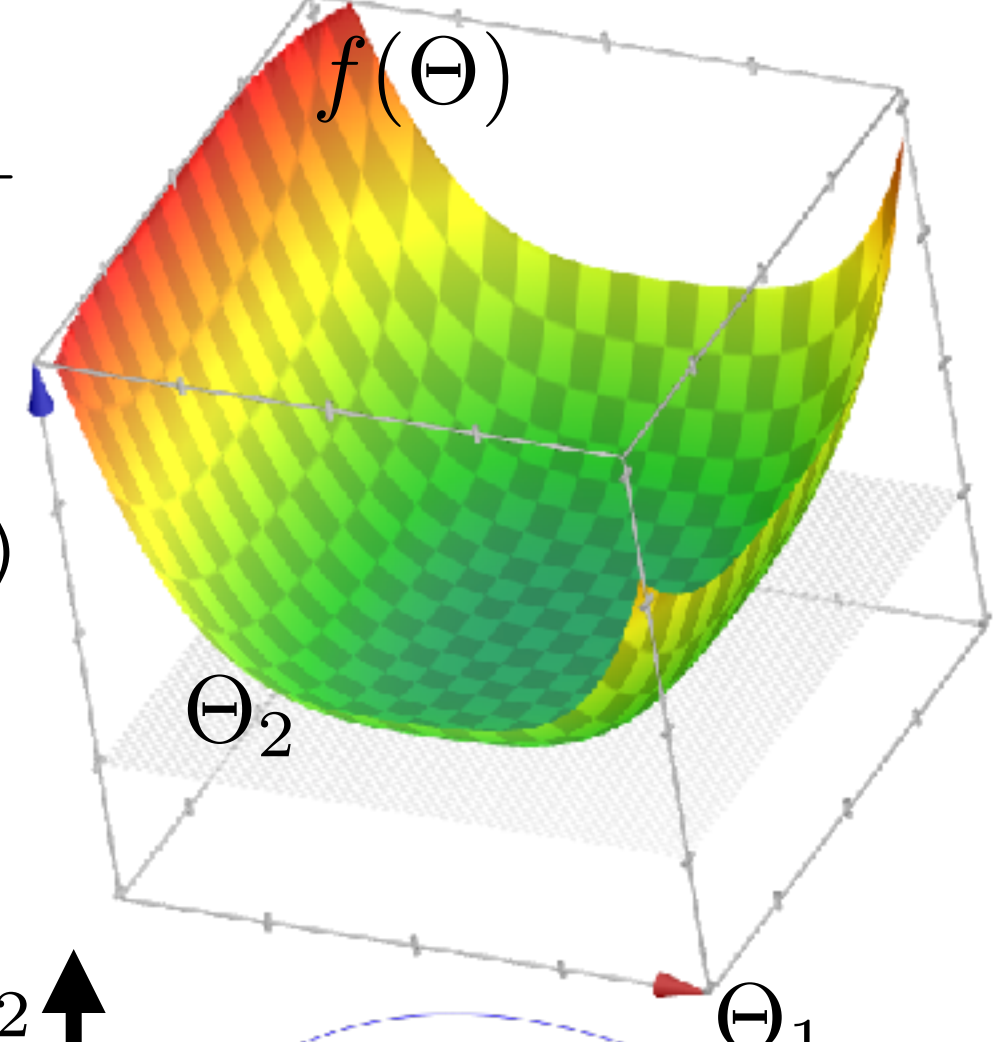
$\texttt{Initialize}\ \Theta^{(0)} = \Theta_{\text{init}}$

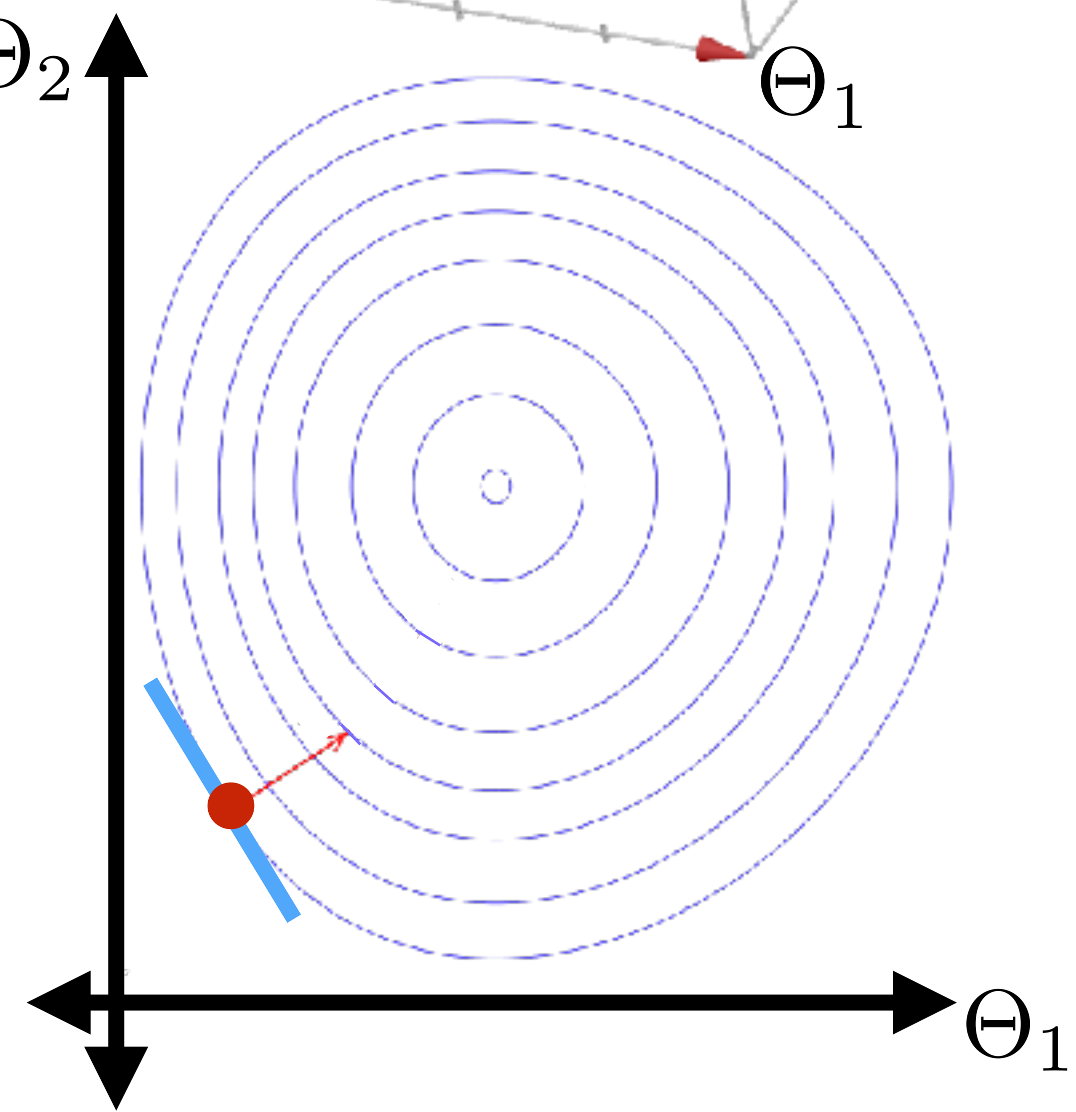
$\texttt{Initialize t = 0}$

repeat

$\texttt{t = t + 1}$

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





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 (Θ_init, η, f, ∇_Θ f, ε)
  Initialize Θ^(0) = Θ_init
  Initialize t = 0
  repeat
    t = t + 1
    Θ^(t) = Θ^(t-1) − η ∇_Θ f(Θ^(t-1))
```

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

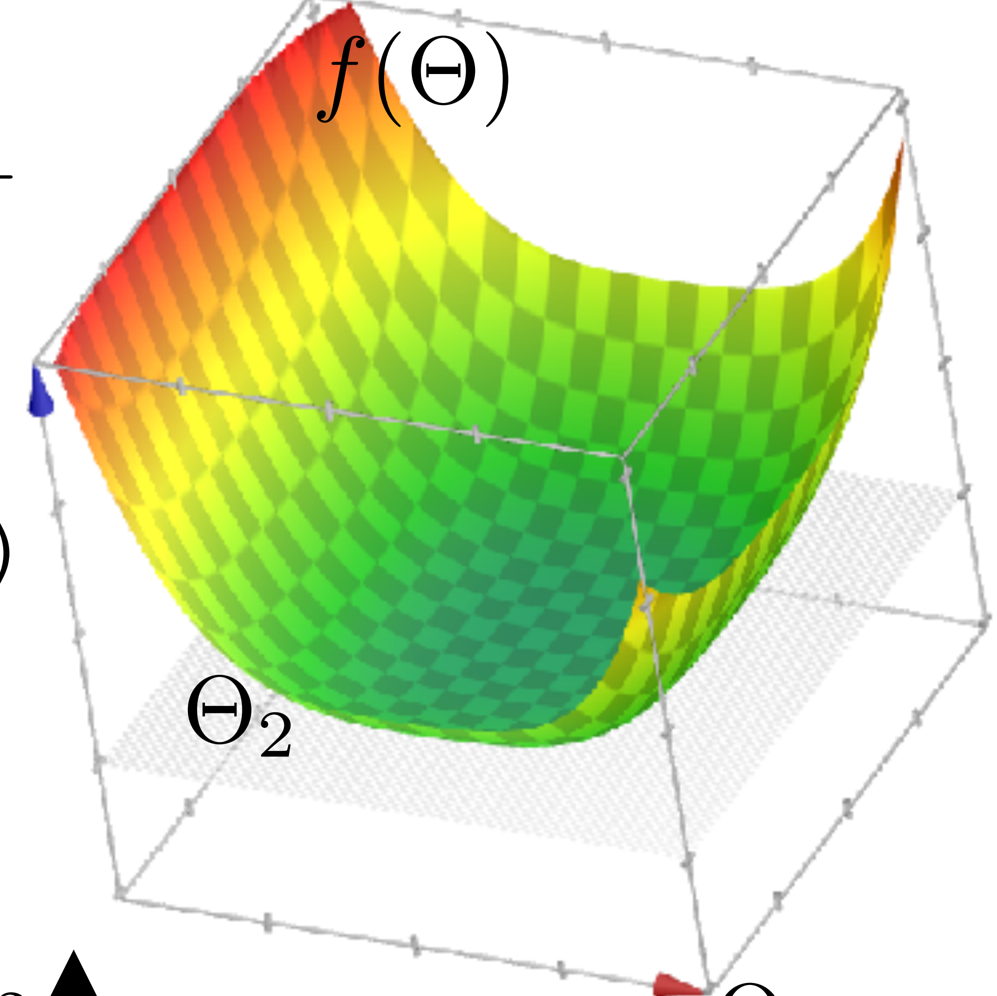
$\texttt{Initialize}\ \Theta^{(0)} = \Theta_{\text{init}}$

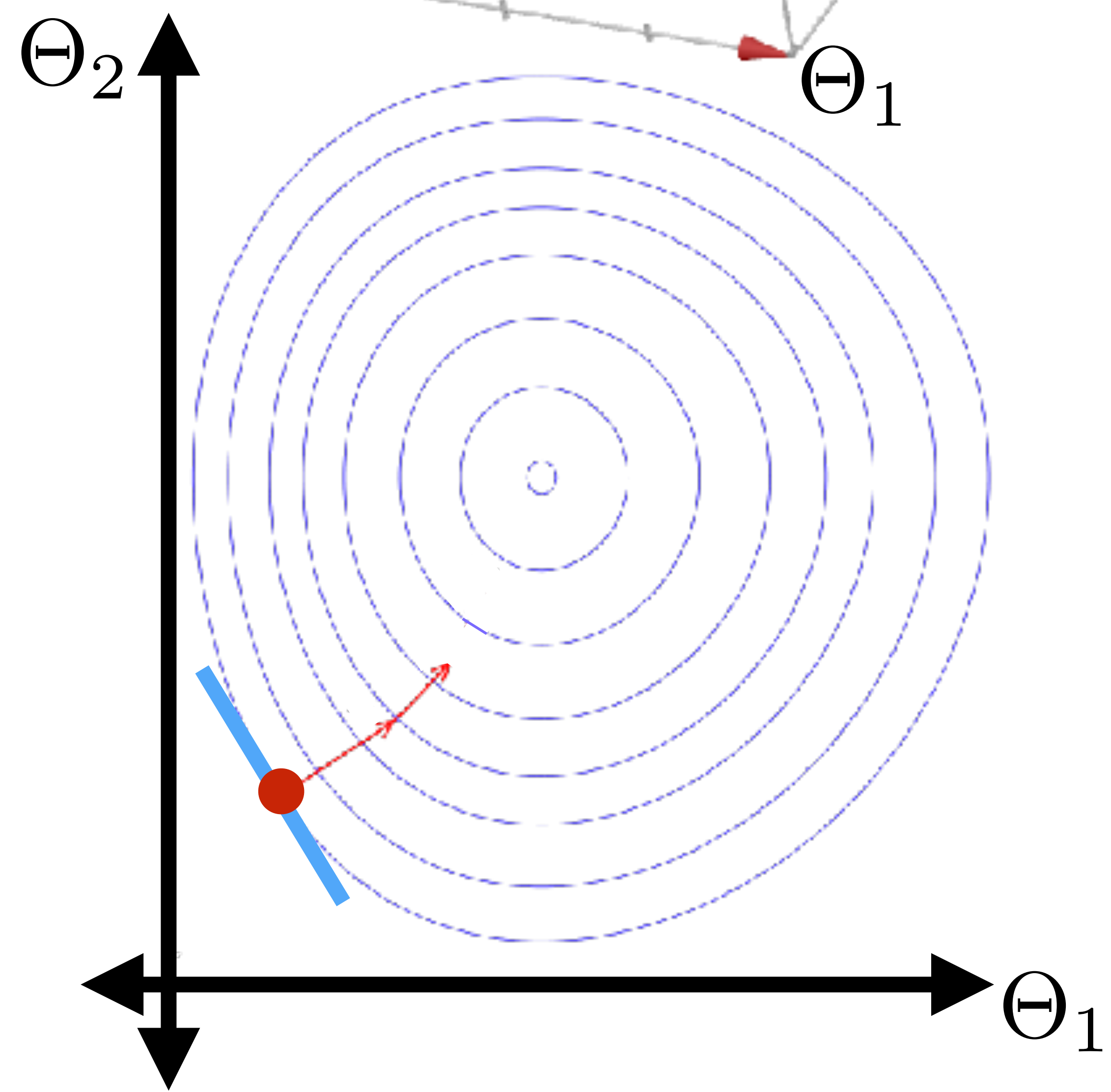
$\texttt{Initialize t = 0}$

repeat

$\texttt{t = t + 1}$

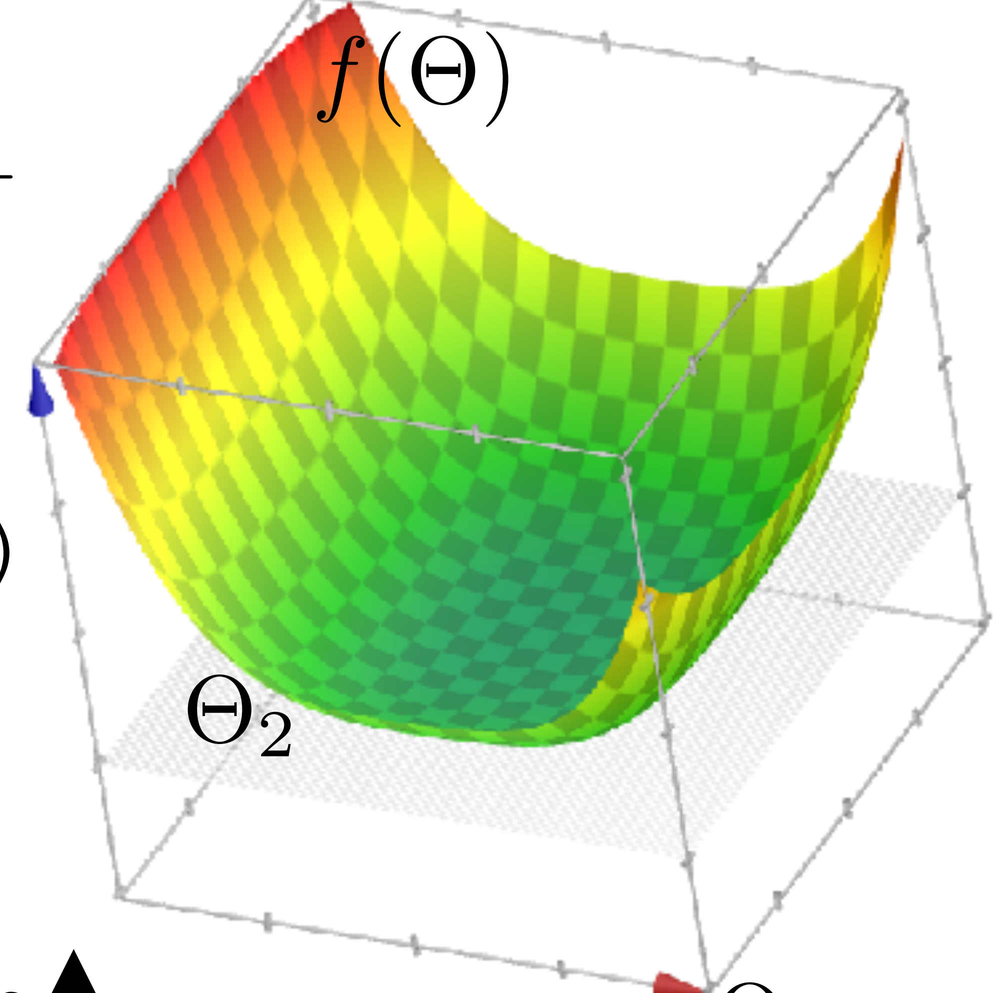
$\Theta^{(t)} = \Theta^{(t-1)} - \eta \nabla_\Theta f(\Theta^{(t-1)})$





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 (Θ_init, η, f, ∇_Θ f, ε)
  Initialize Θ^(0) = Θ_init
  Initialize t = 0
  repeat
    t = t + 1
    Θ^(t) = Θ^(t-1) - η ∇_Θ f(Θ^(t-1))
```

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

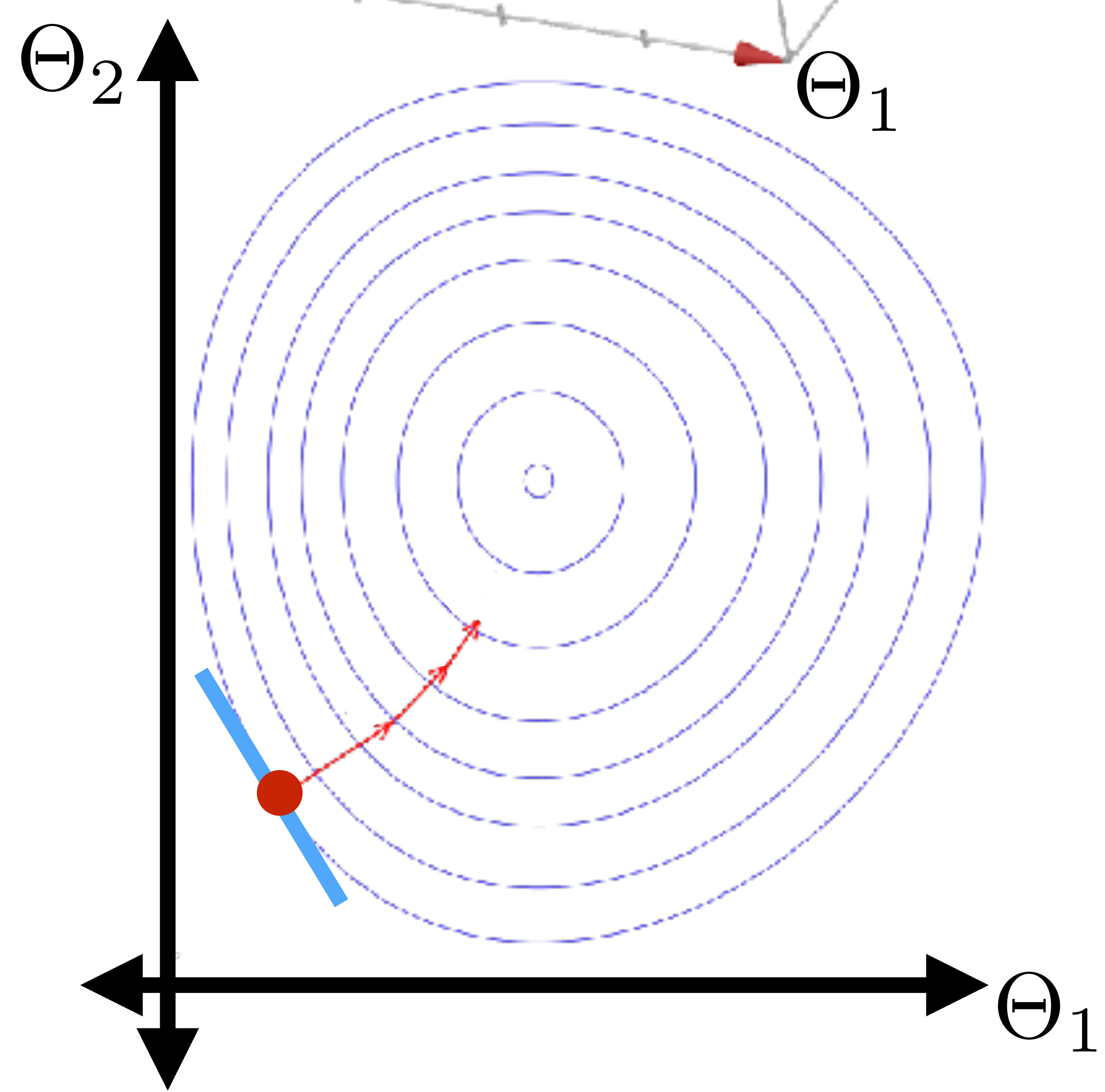
$\texttt{Initialize}\ \Theta^{(0)} = \Theta_{\text{init}}$

$\texttt{Initialize t = 0}$

repeat

$\texttt{t = t + 1}$

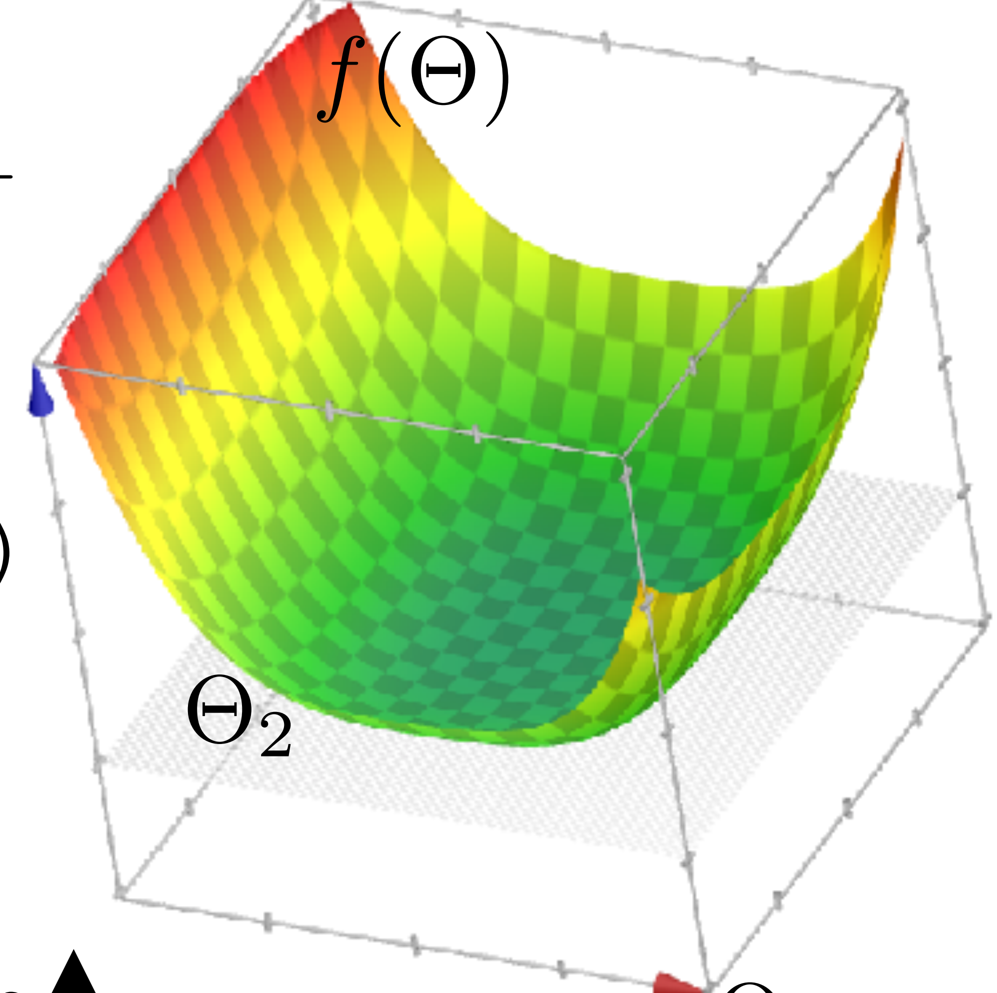
$\Theta^{(t)} = \Theta^{(t-1)} - \eta \nabla_\Theta f(\Theta^{(t-1)})$

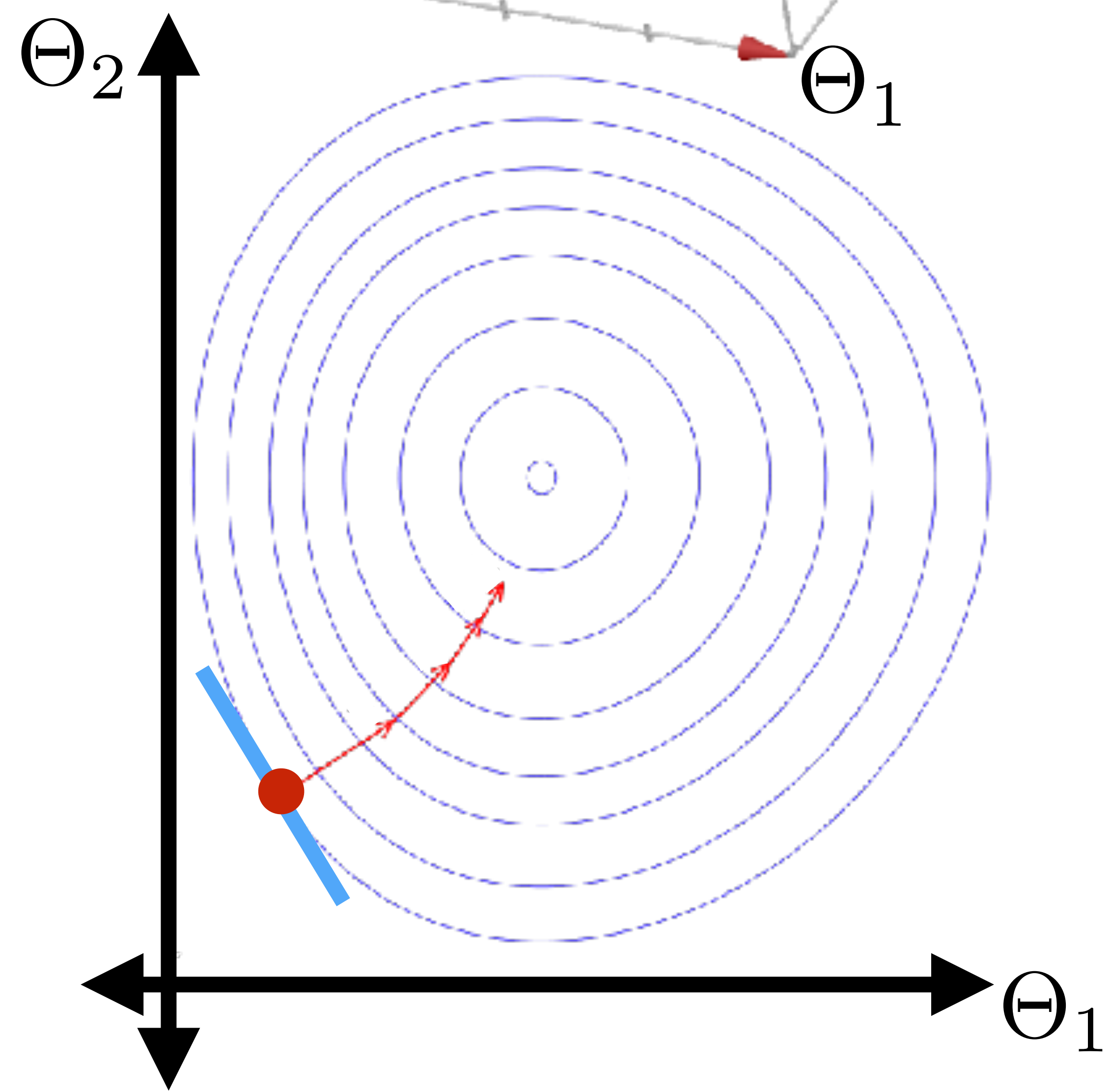


Gradient descent

- 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 (Θ_init, η, f, ∇_Θ f, ε)
  Initialize Θ^(0) = Θ_init
  Initialize t = 0
  repeat
    t = t + 1
    Θ^(t) = Θ^(t-1) − η ∇_Θ f(Θ^(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 $\Theta^{(0)} = \Theta_{\text{init}}$

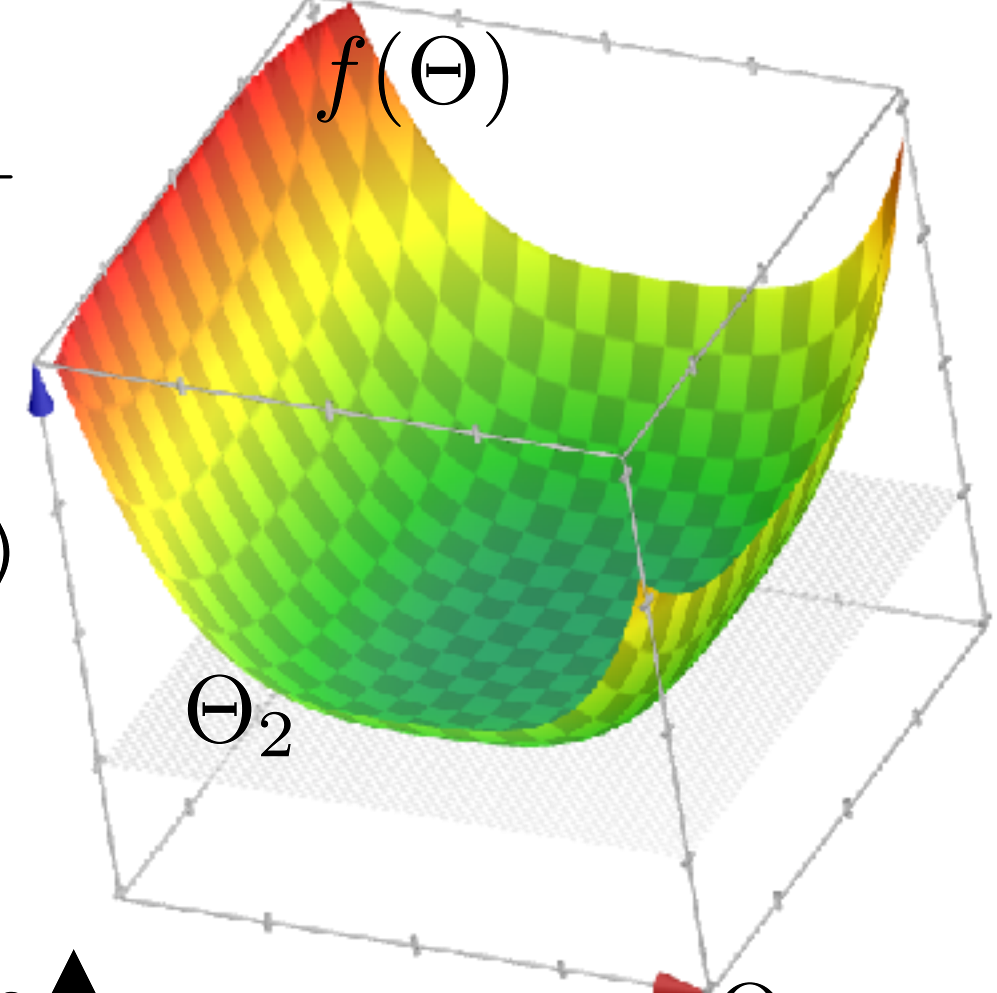
  Initialize t = 0

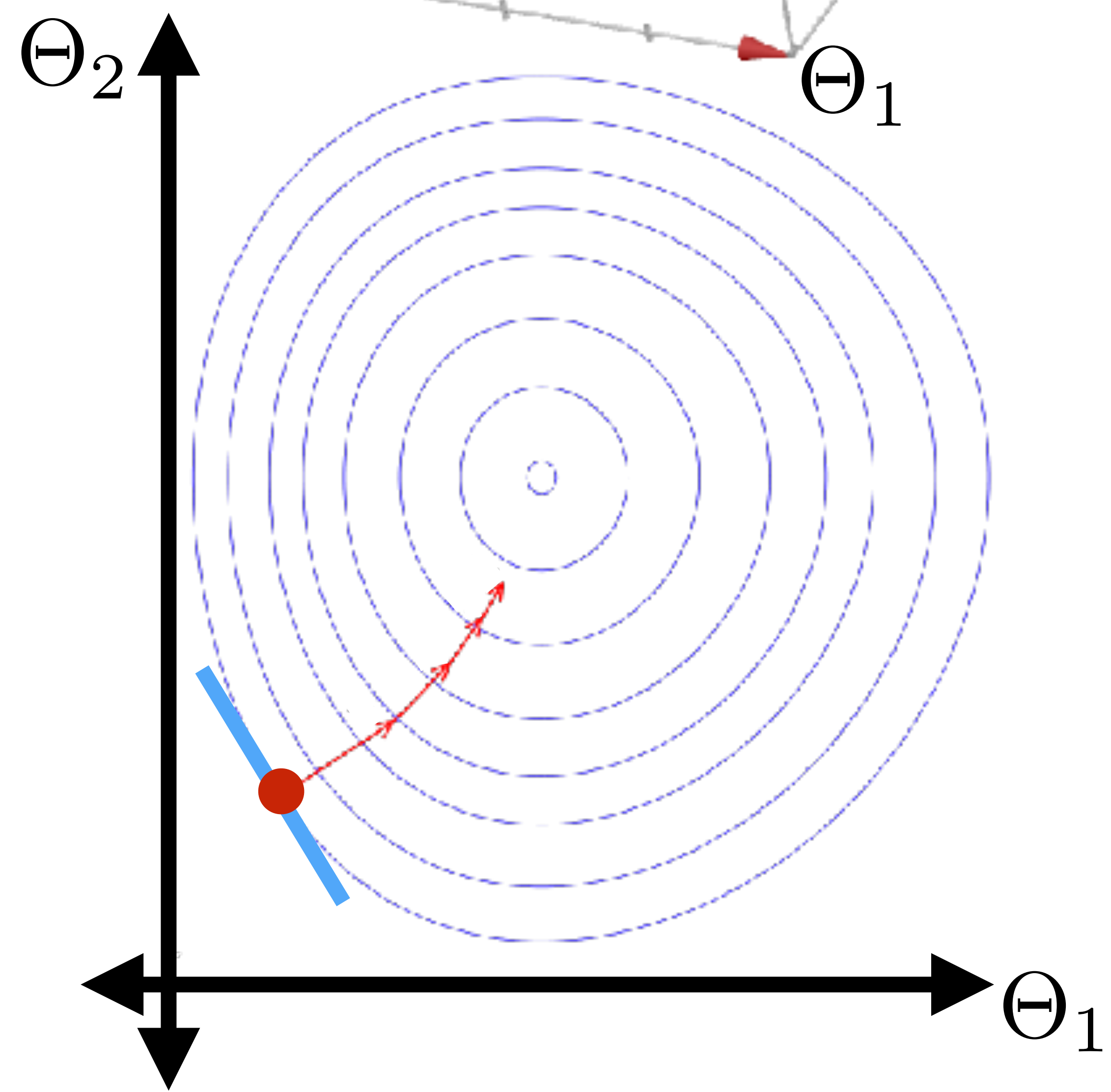
  repeat

    t = t + 1

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

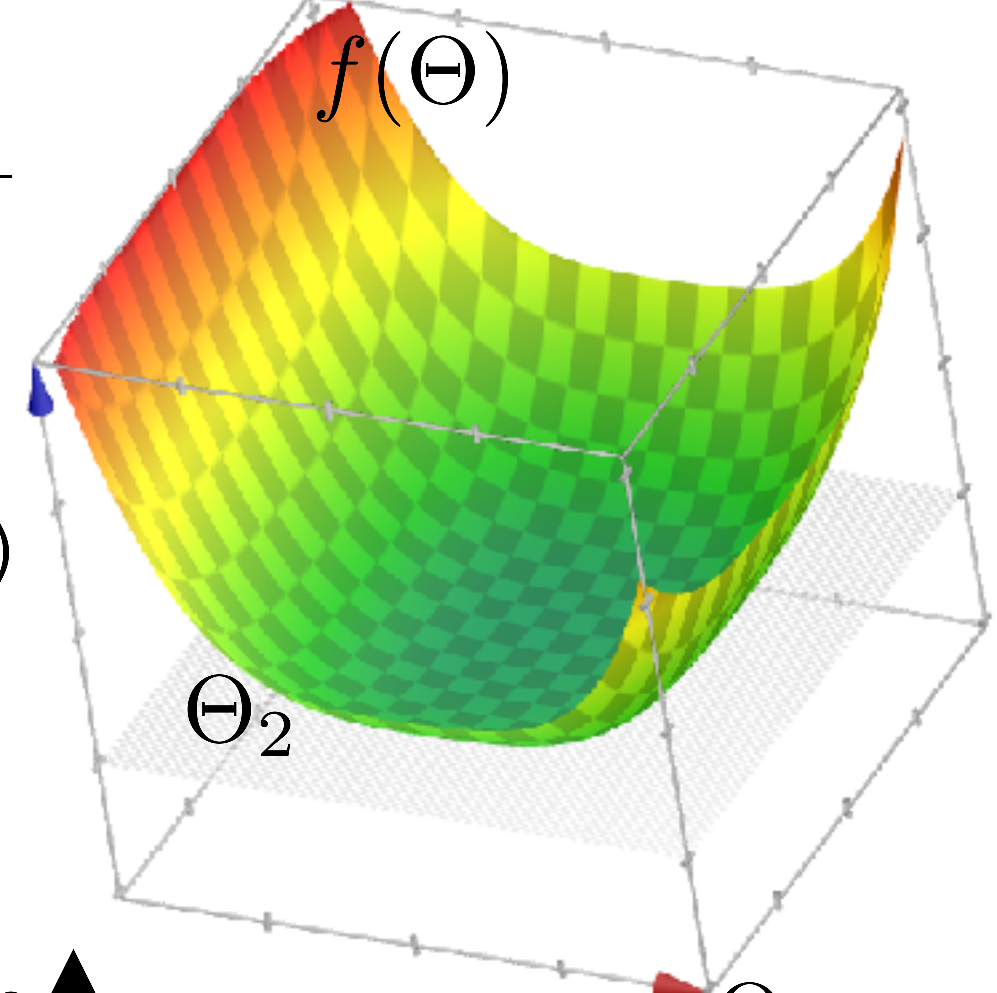
  until





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

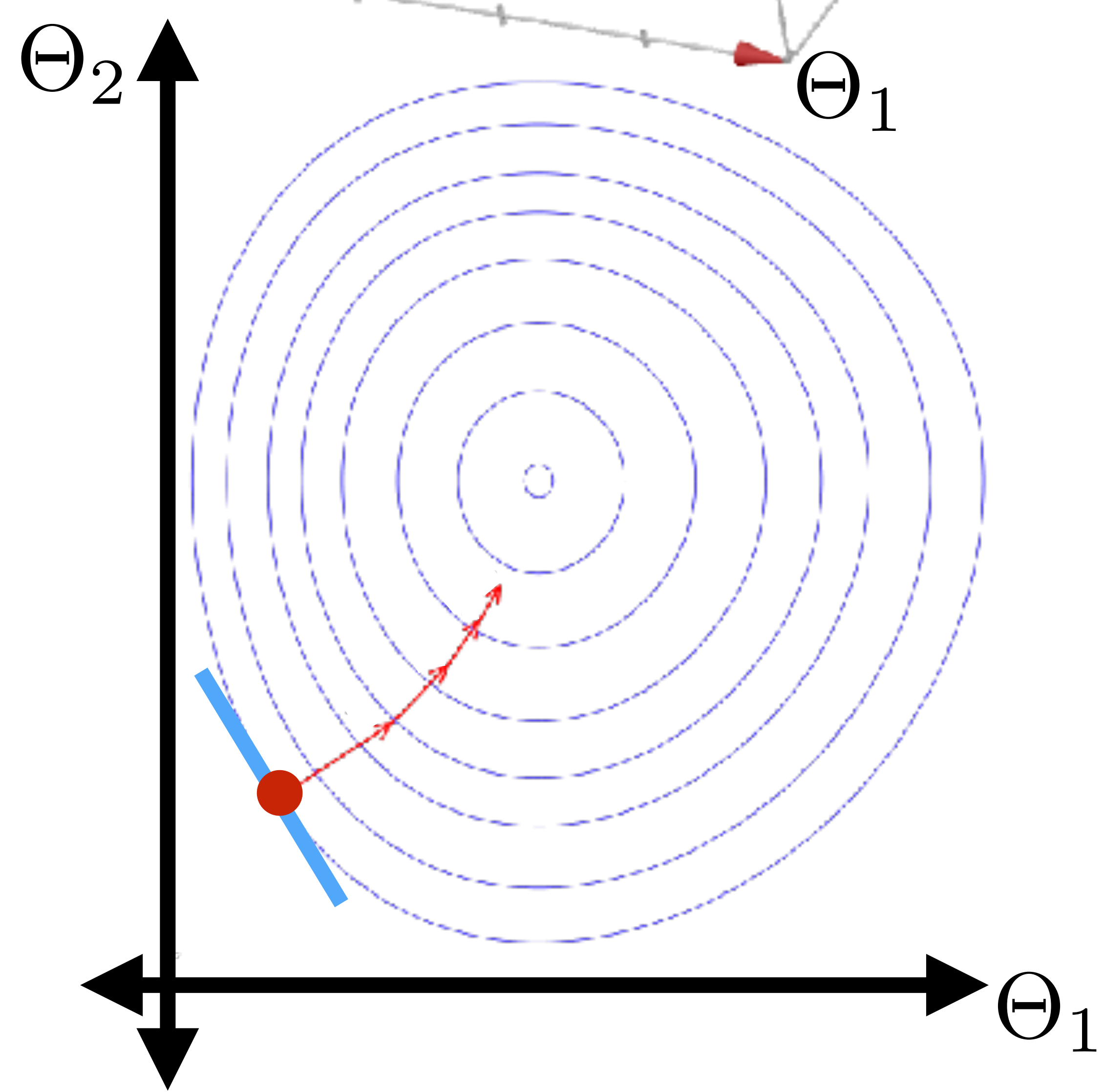
  Initialize t = 0

  repeat

    t = t + 1

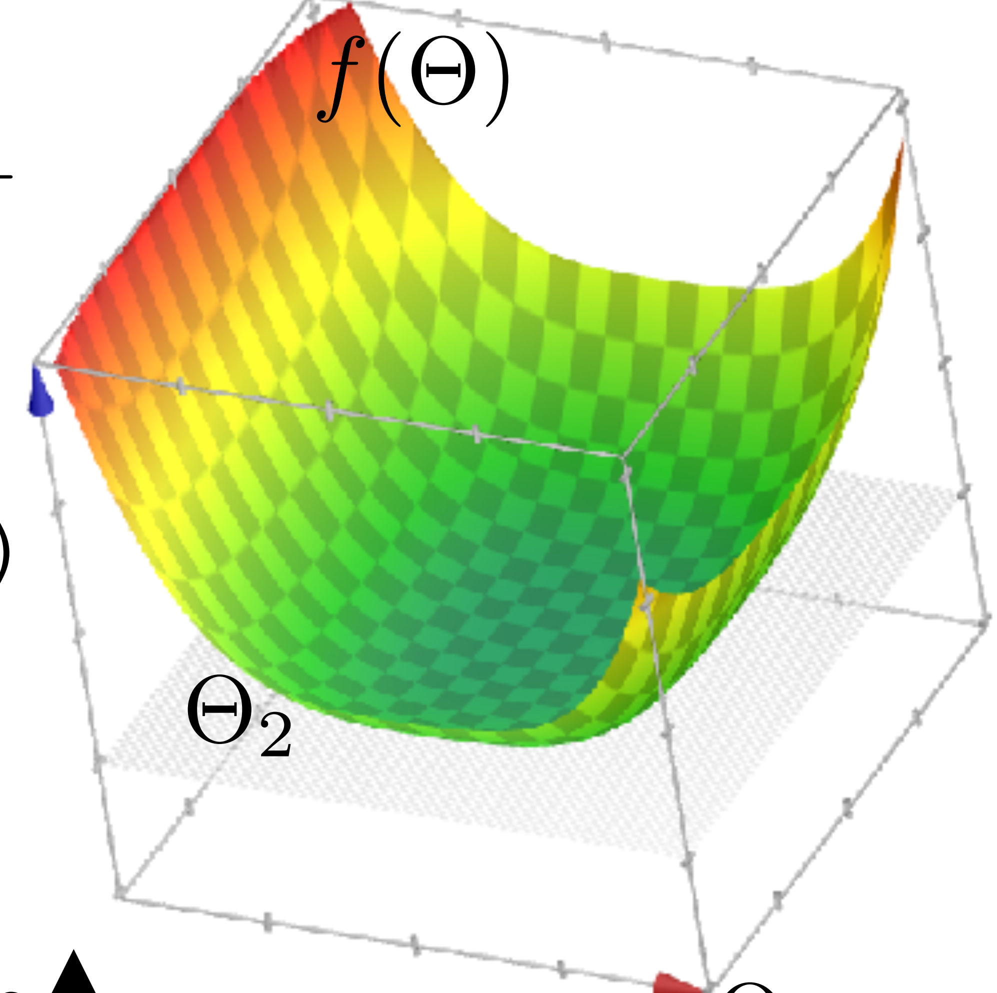
    $\Theta^{(t)} = \Theta^{(t-1)} - \eta \nabla_\Theta f(\Theta^{(t-1)})$

  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

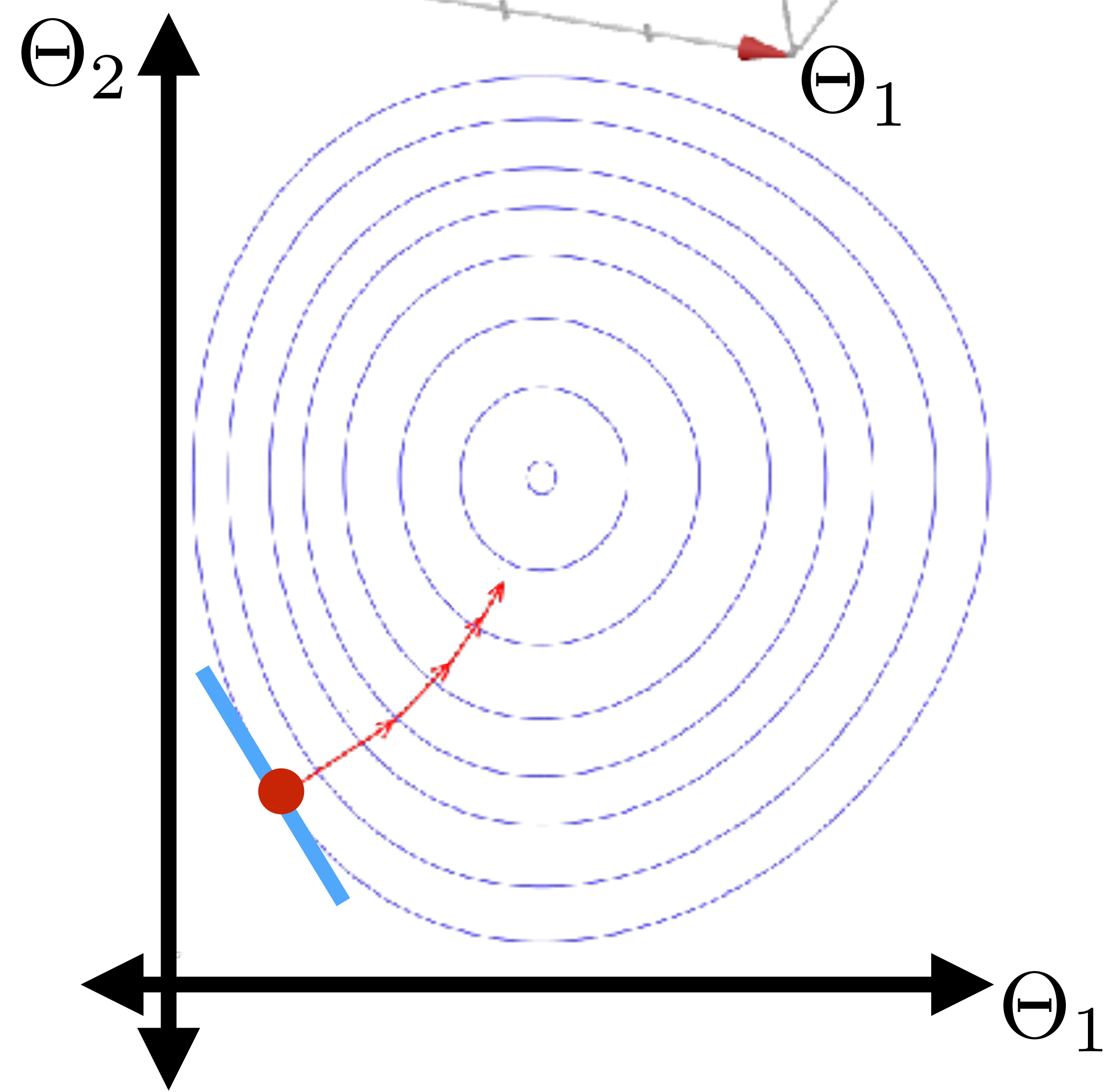
repeat

t = t + 1

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

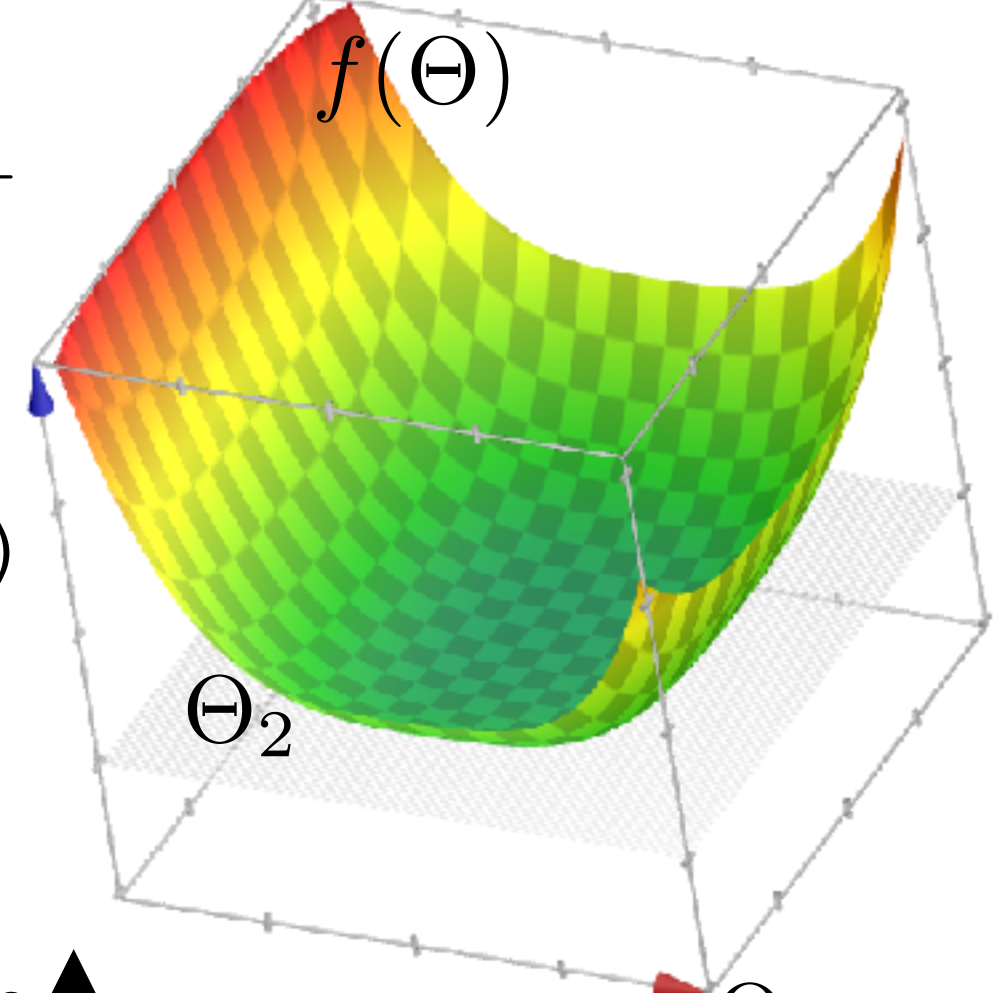
until $\left| f(\Theta^{(t)}) - f(\Theta^{(t-1)}) \right| < \epsilon$

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 $\Theta \in \mathbb{R}^m$

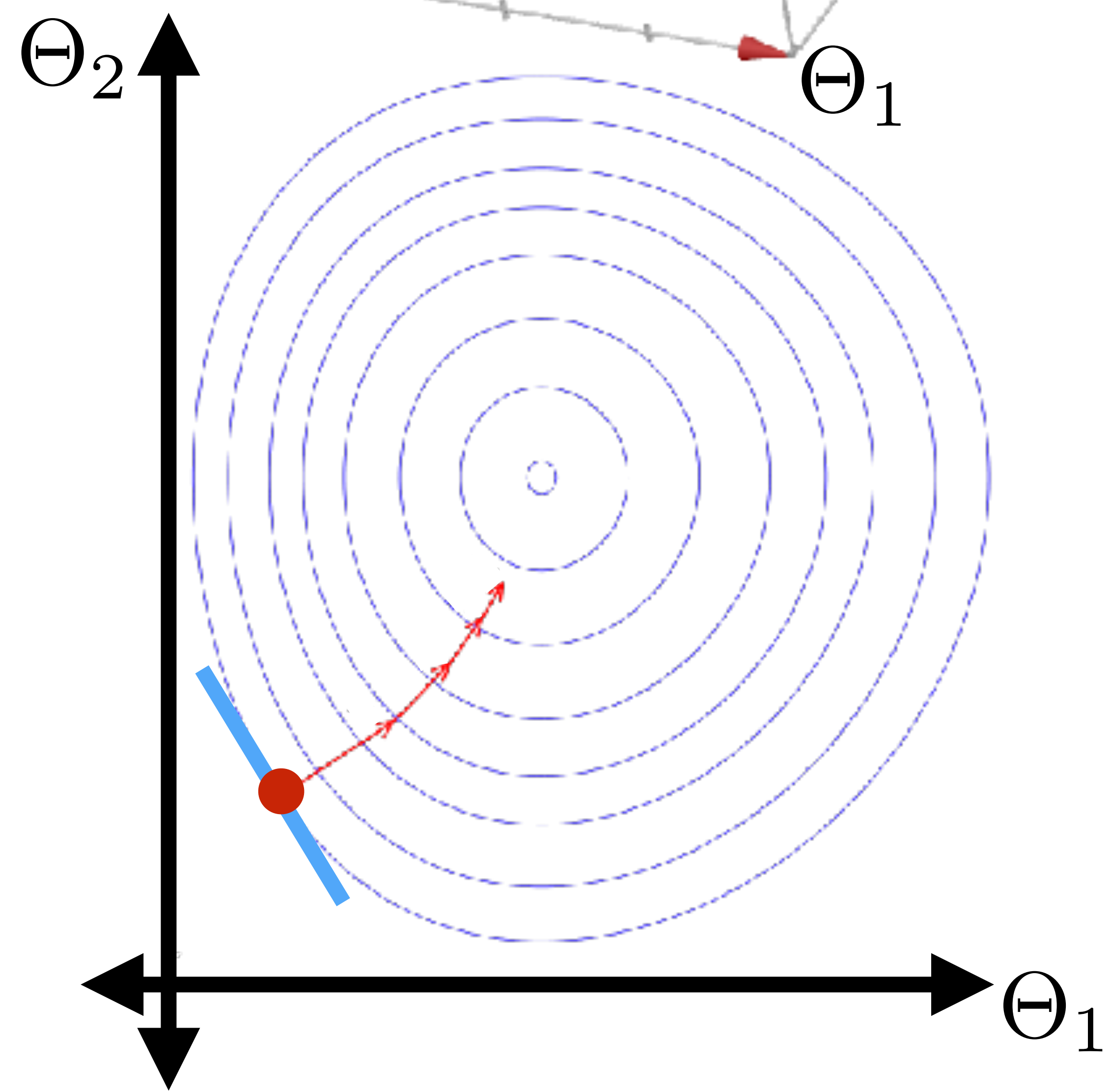



```
Gradient-Descent (Θ_init, η, f, ∇_Θ f, ε)
  Initialize Θ^(0) = Θ_init
  Initialize t = 0
  repeat
    t = t + 1
    Θ^(t) = Θ^(t-1) - η ∇_Θ f(Θ^(t-1))
  until |f(Θ^(t)) - f(Θ^(t-1))| < ε
  Return Θ^(t)
```

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

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

- Other possible stopping criteria:

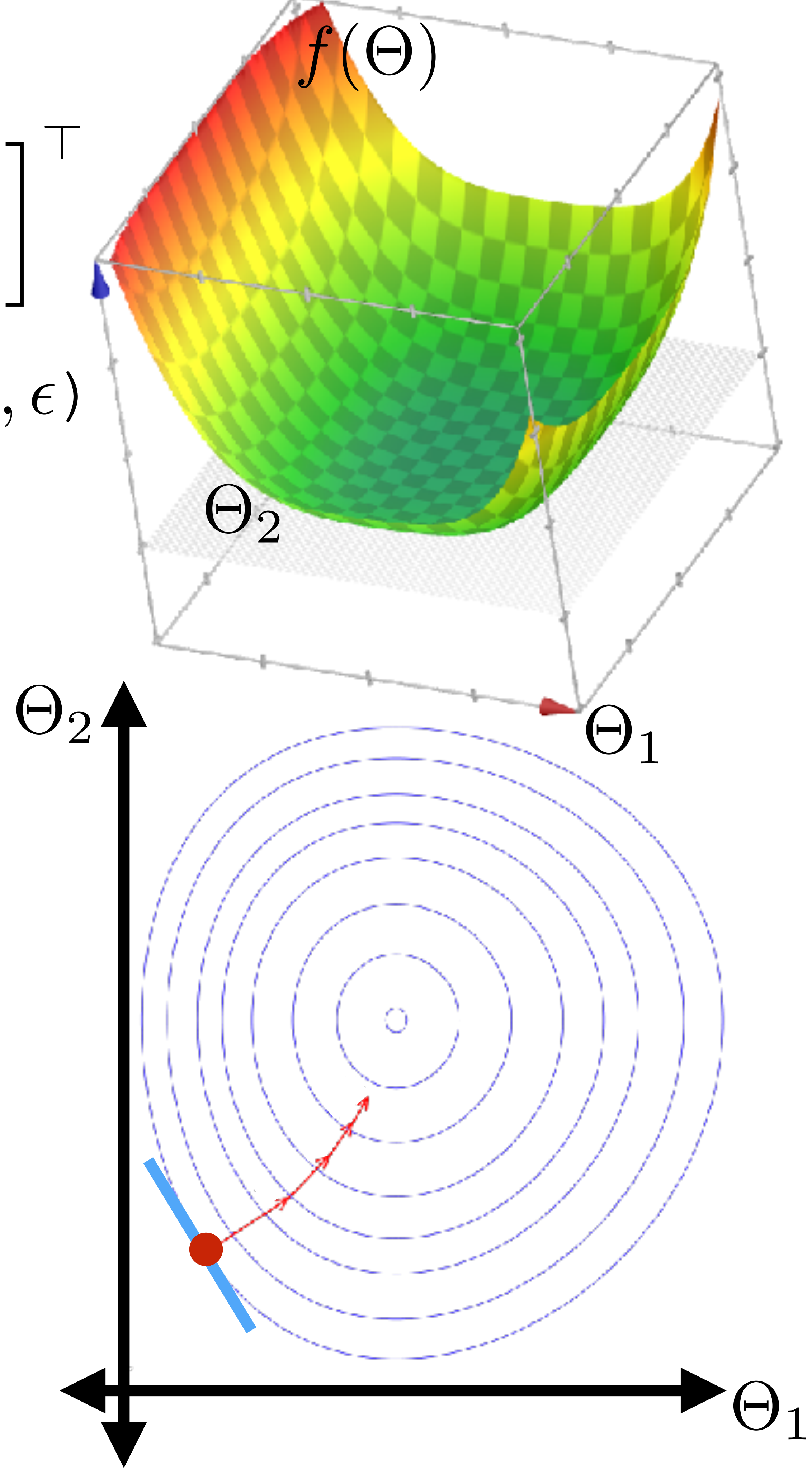


Gradient descent

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

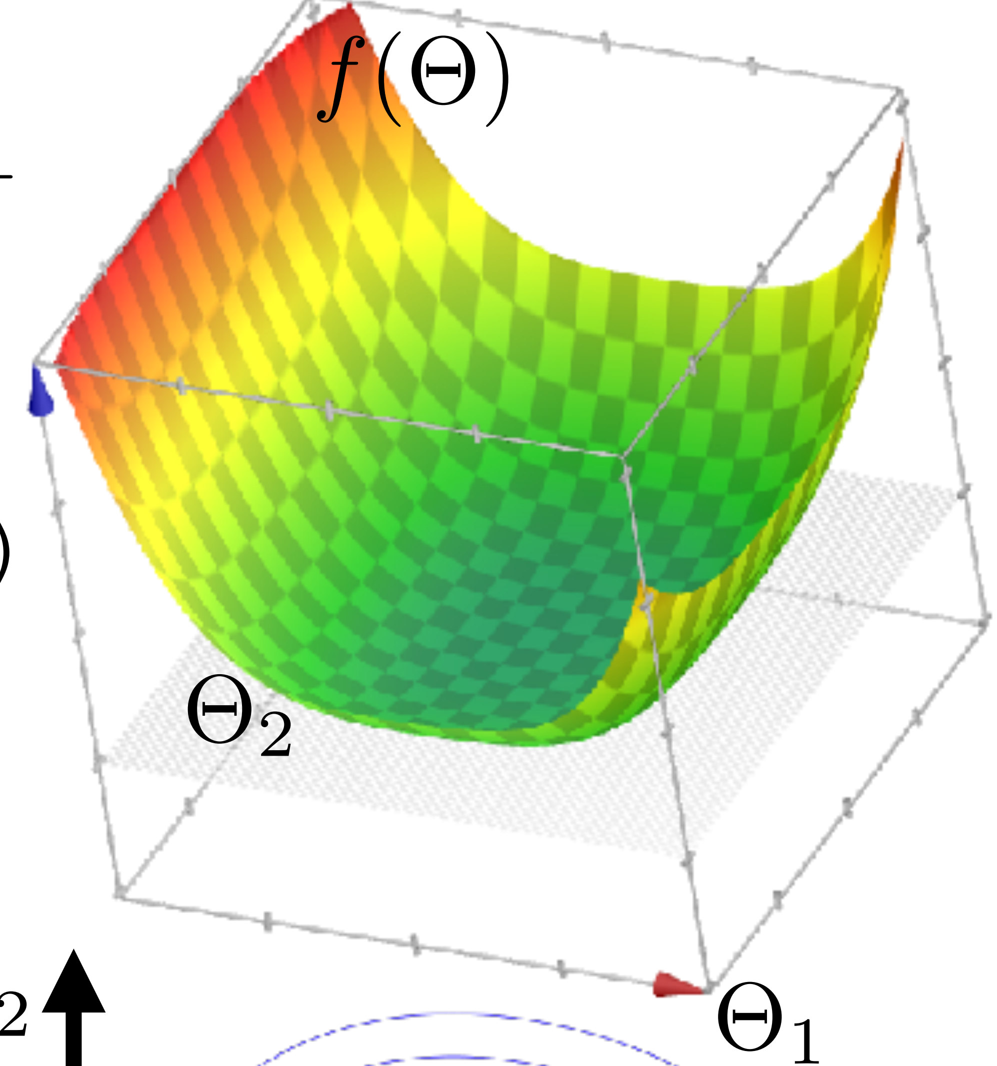
```
Gradient-Descent (Θ_init, η, f, ∇_Θ f, ε)
  Initialize Θ^(0) = Θ_init
  Initialize t = 0
  repeat
    t = t + 1
    Θ^(t) = Θ^(t-1) − η ∇_Θ f(Θ^(t-1))
  until |f(Θ^(t)) − f(Θ^(t-1))| < ε
  Return Θ^(t)
```

- 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`

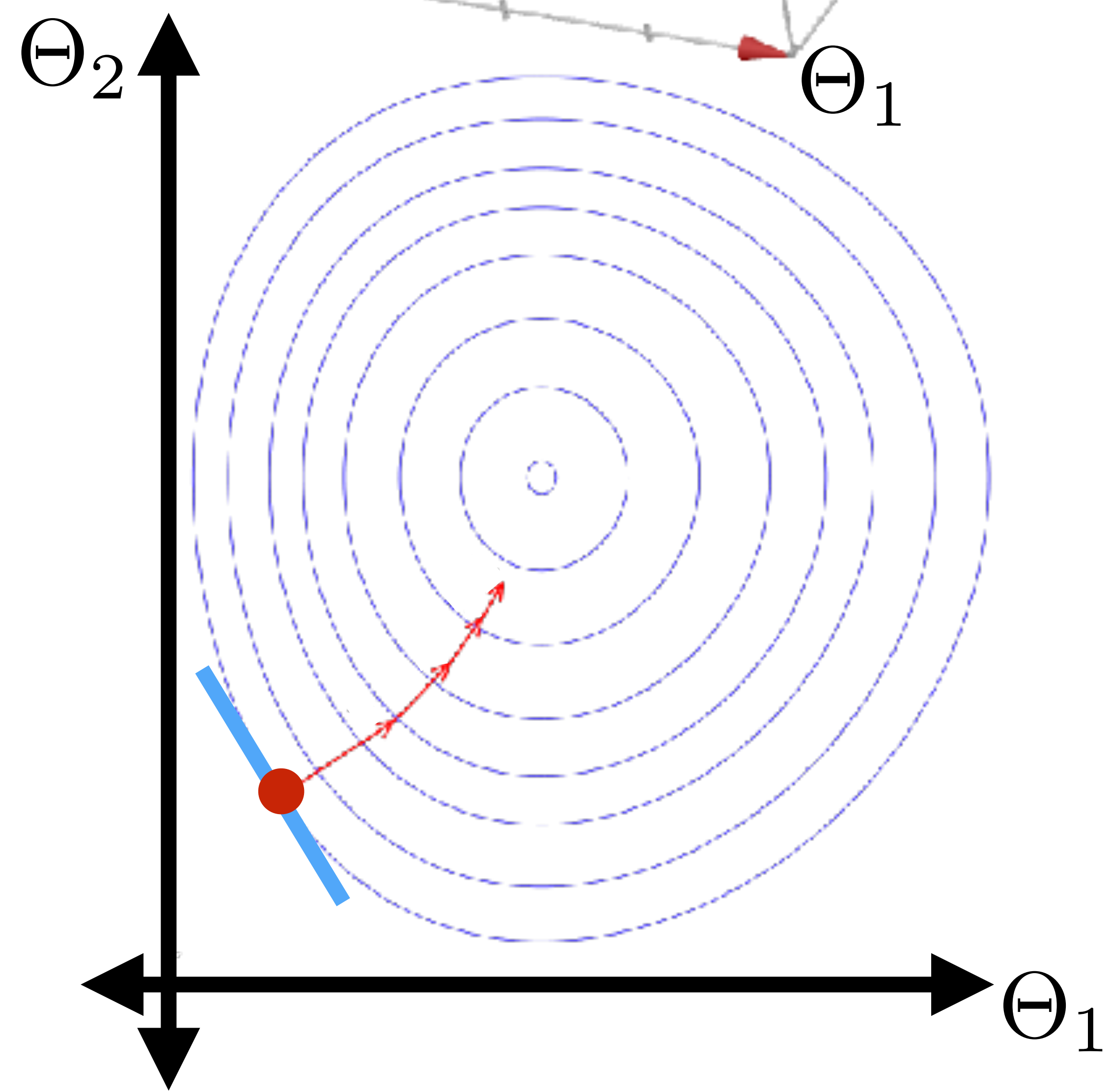
    `t = t + 1`

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

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

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

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

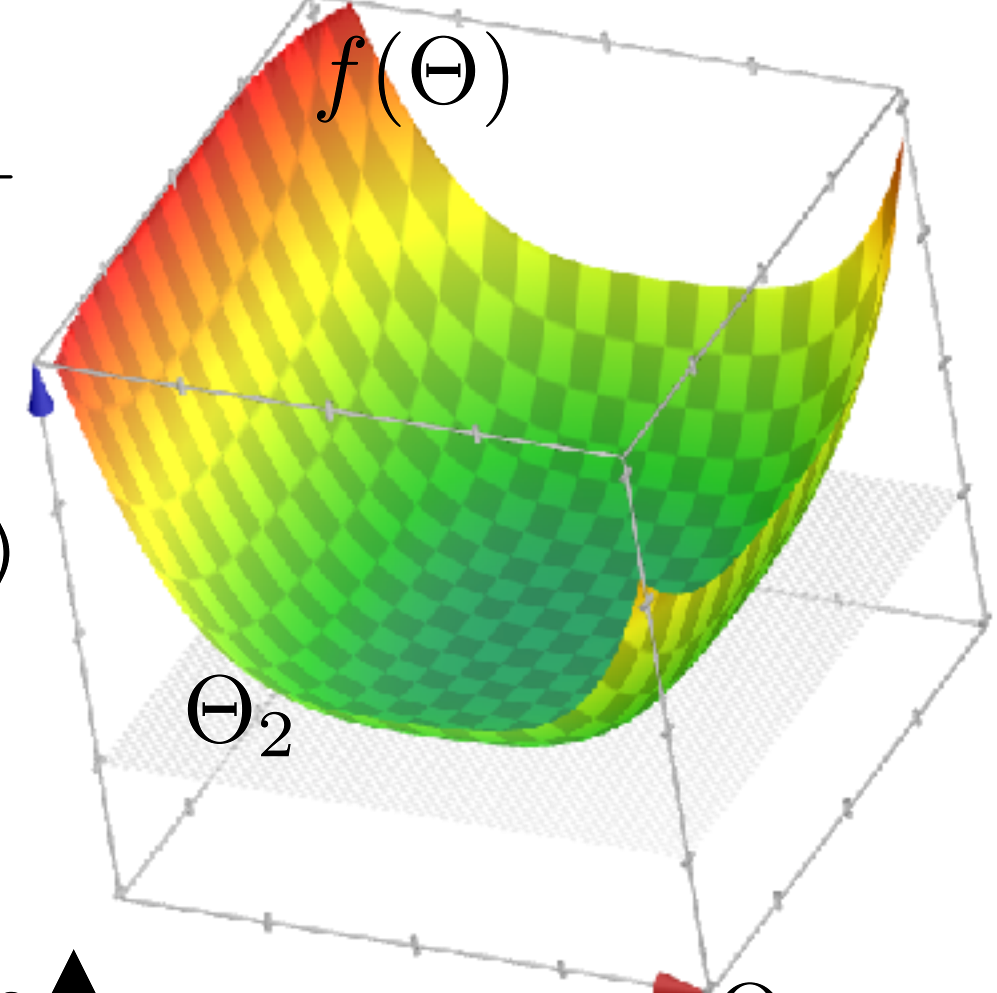


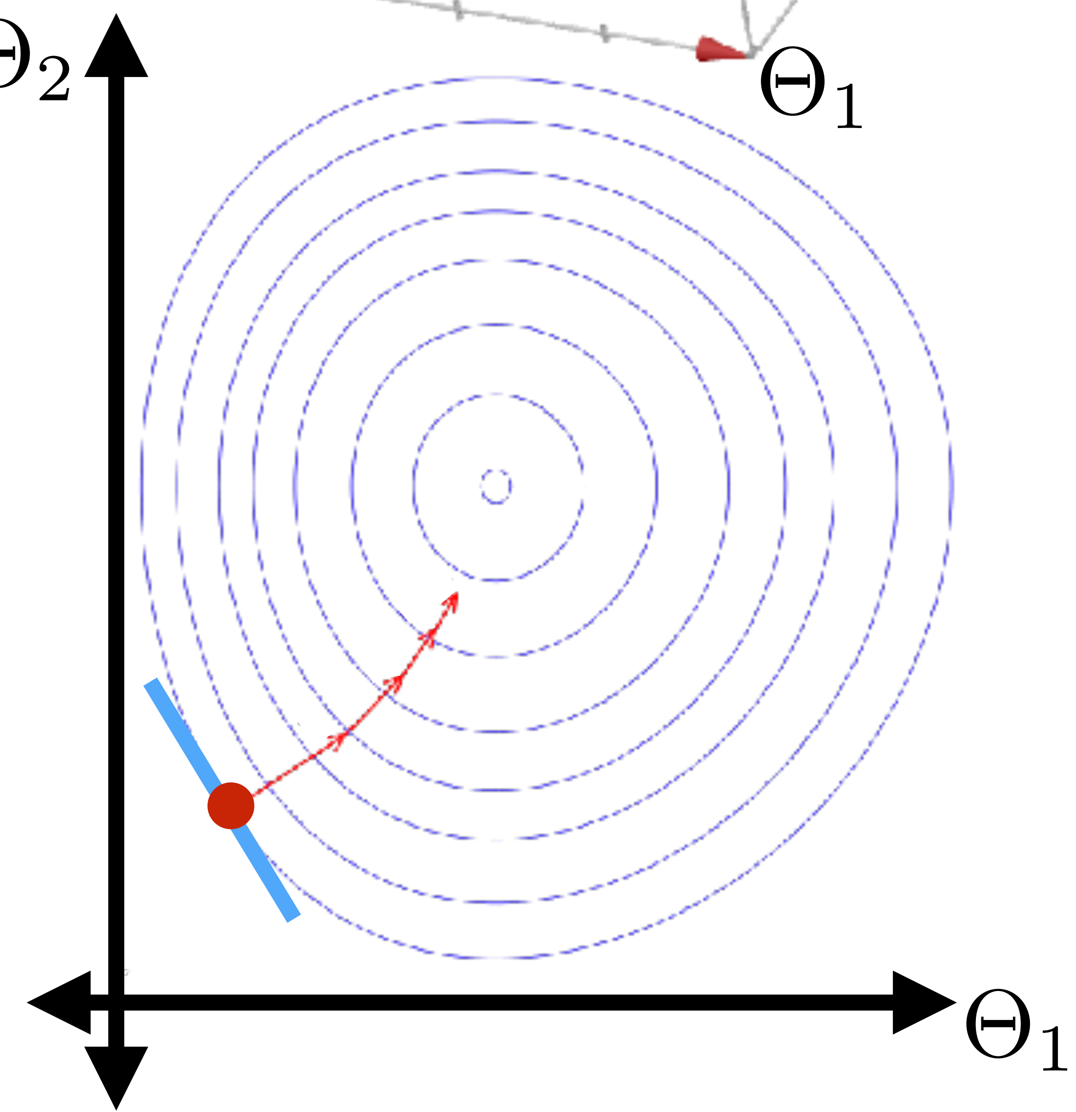
Gradient descent

- 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 (Θ_init, η, f, ∇_Θ f, ε)
  Initialize Θ^(0) = Θ_init
  Initialize t = 0
  repeat
    t = t + 1
    Θ^(t) = Θ^(t−1) − η∇_Θ f(Θ^(t−1))
  until |f(Θ^(t)) − f(Θ^(t−1))| < ε
  Return Θ^(t)
```

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



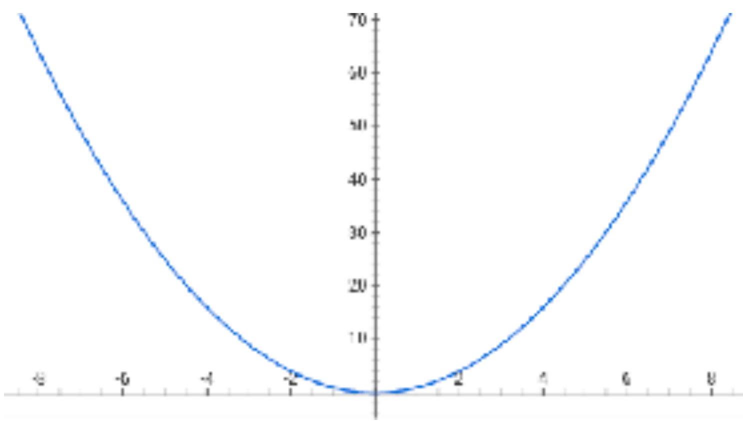


Gradient descent properties

- 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

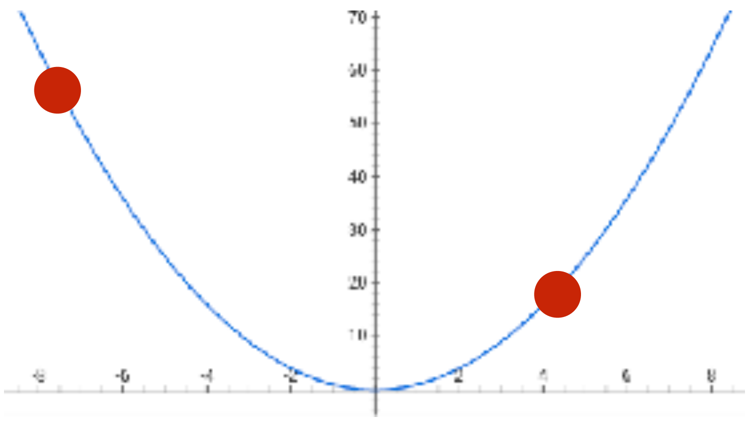
Gradient descent properties

- 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



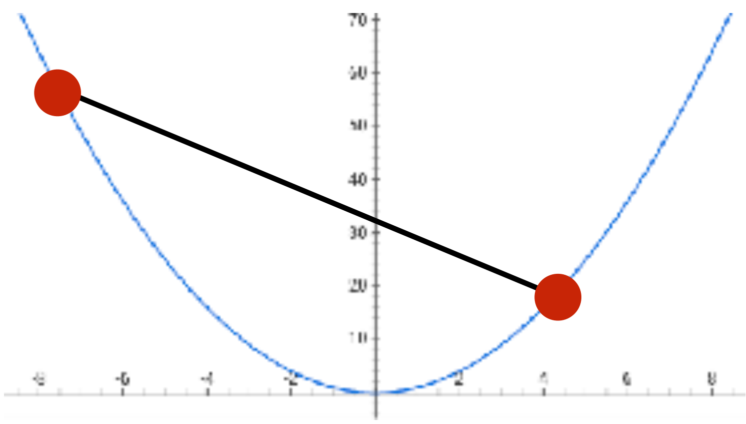
Gradient descent properties

- 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



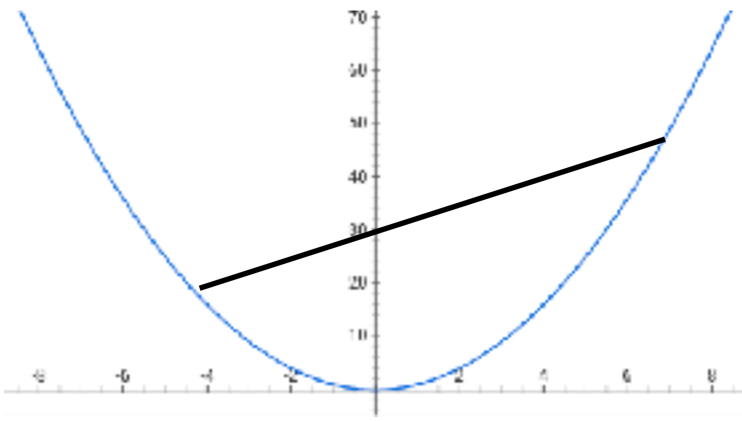
Gradient descent properties

- 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



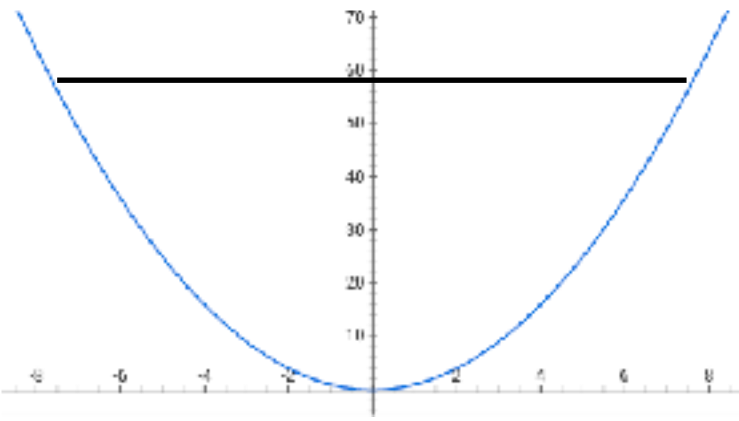
Gradient descent properties

- 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



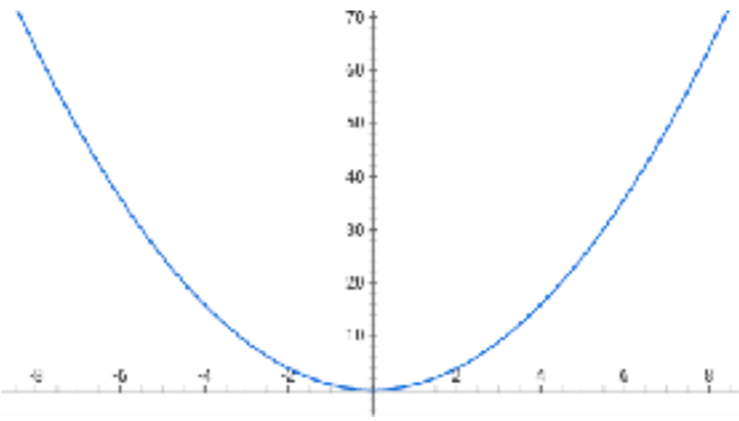
Gradient descent properties

- 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



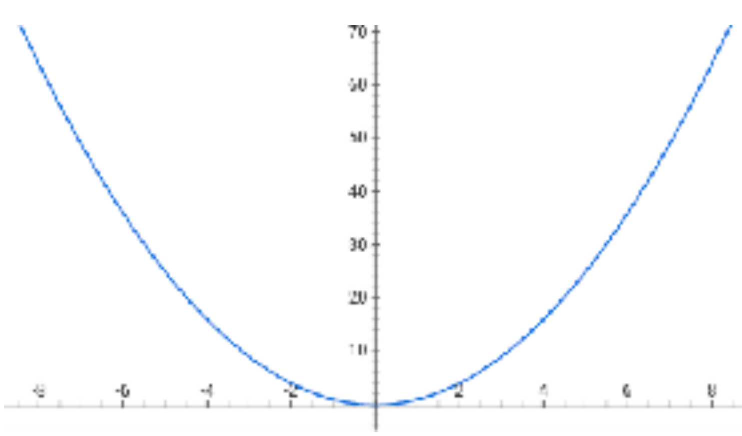
Gradient descent properties

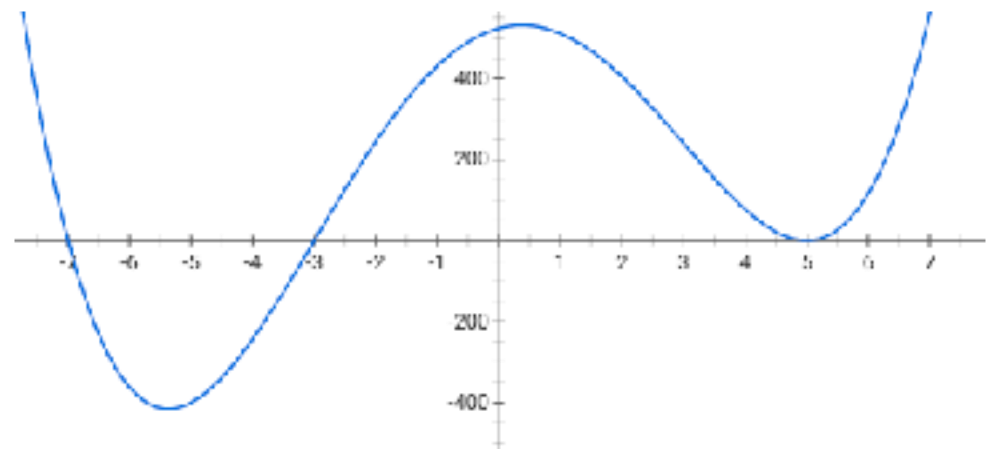
- A function *f* on $\mathbb{R}^m$ is convex if any line segment connecting two points of the graph of *f* lies above or on the graph



Gradient descent properties

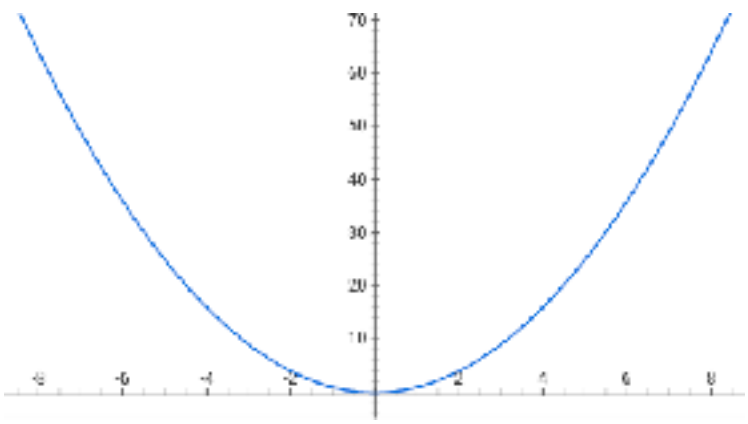
- A function *f* on $\mathbb{R}^m$ is convex if any line segment connecting two points of the graph of *f* lies above or on the graph

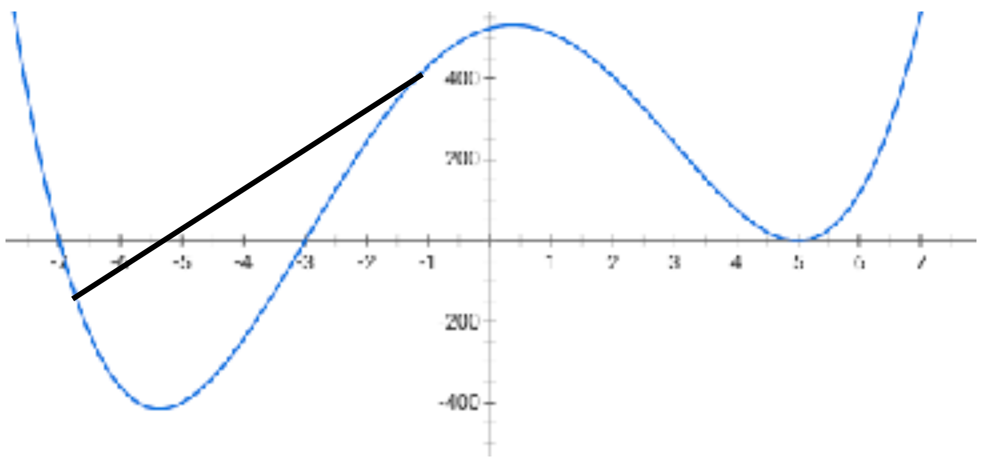




Gradient descent properties

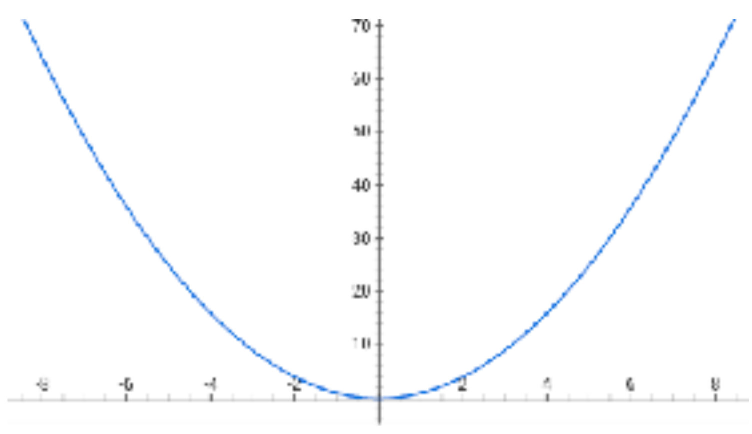
- A function *f* on $\mathbb{R}^m$ is convex if any line segment connecting two points of the graph of *f* lies above or on the graph

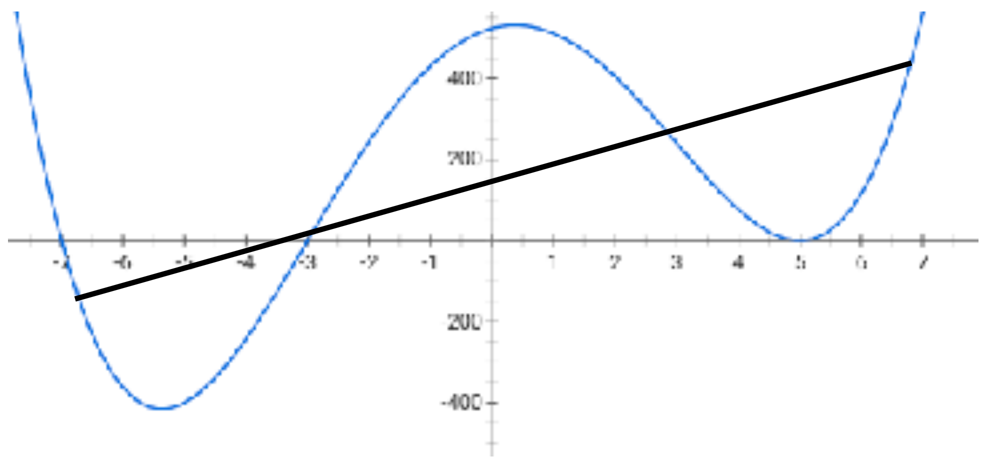




Gradient descent properties

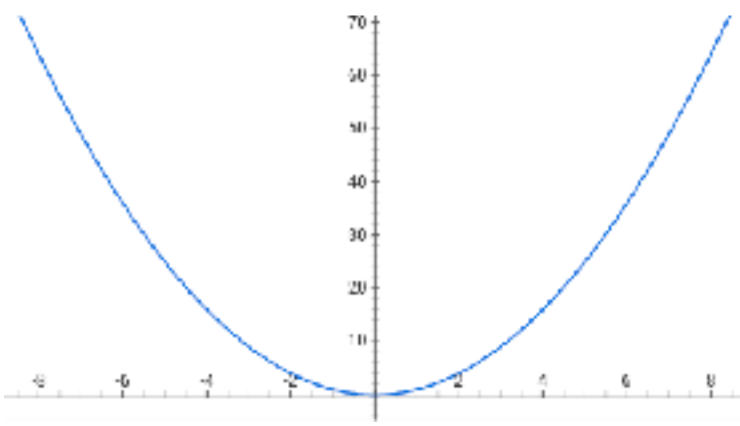
- A function *f* on $\mathbb{R}^m$ is convex if any line segment connecting two points of the graph of *f* lies above or on the graph

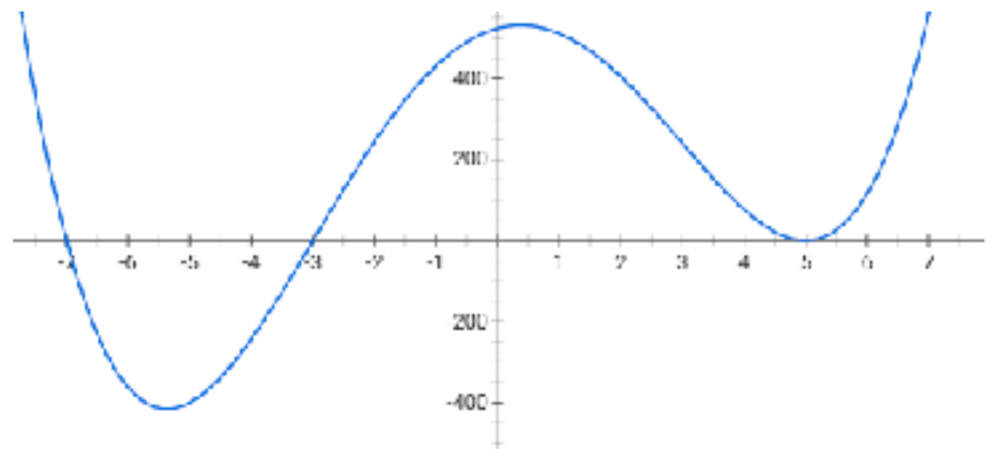




Gradient descent properties

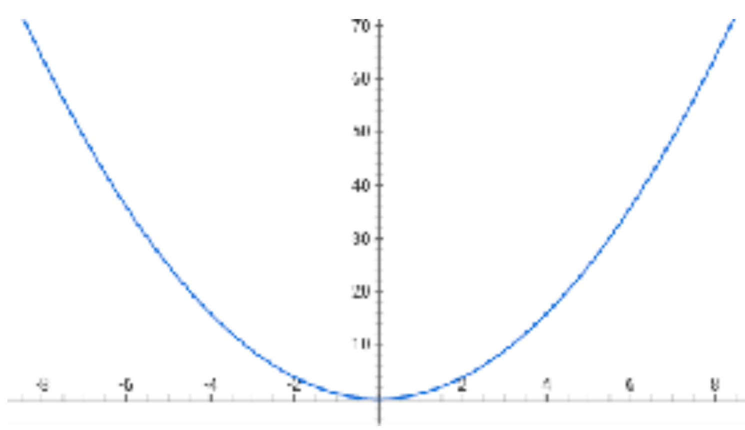
- A function *f* on $\mathbb{R}^m$ is convex if any line segment connecting two points of the graph of *f* lies above or on the graph

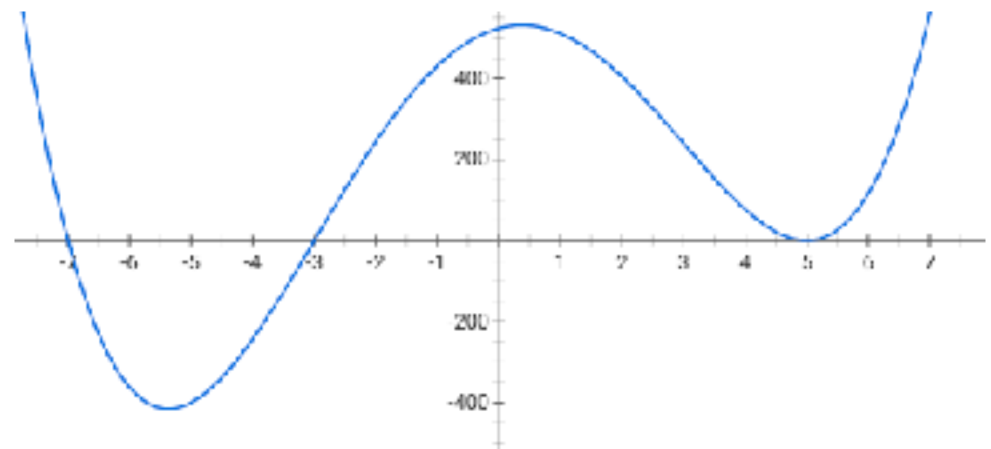


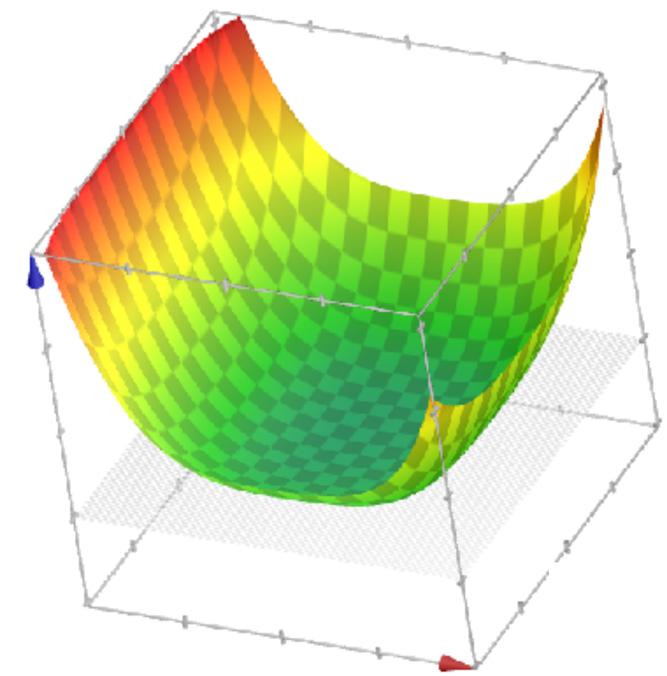


Gradient descent properties

- 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

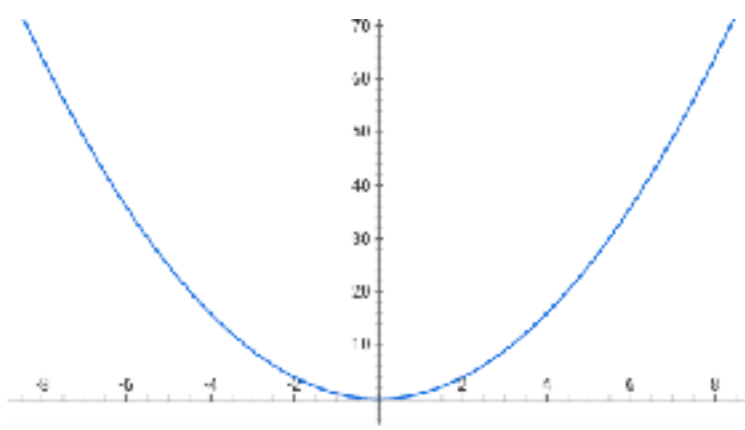


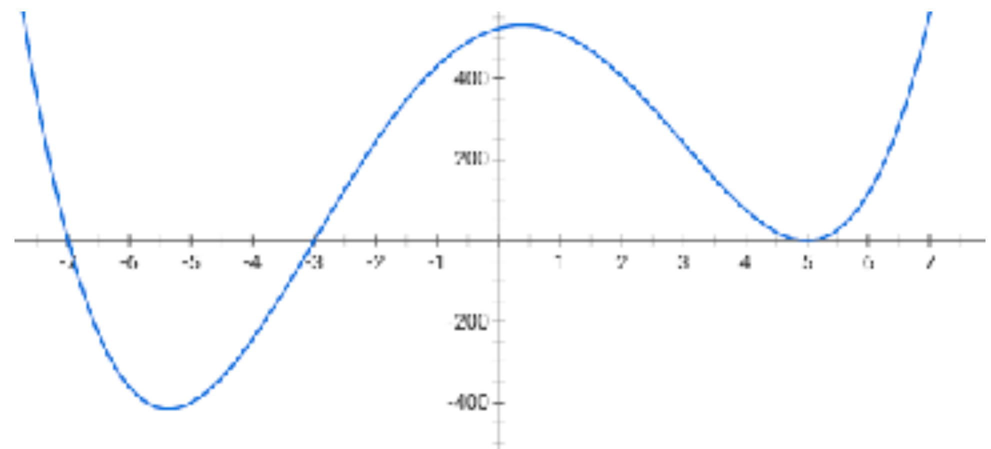


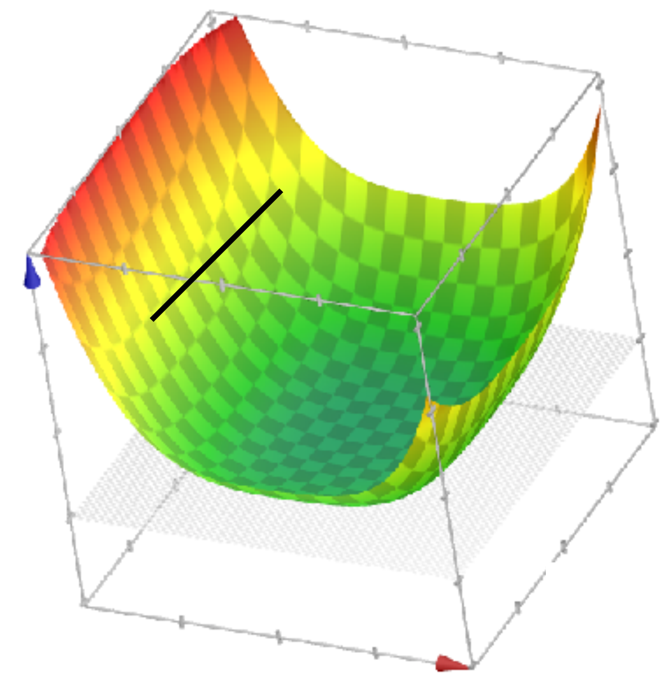


Gradient descent properties

- 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

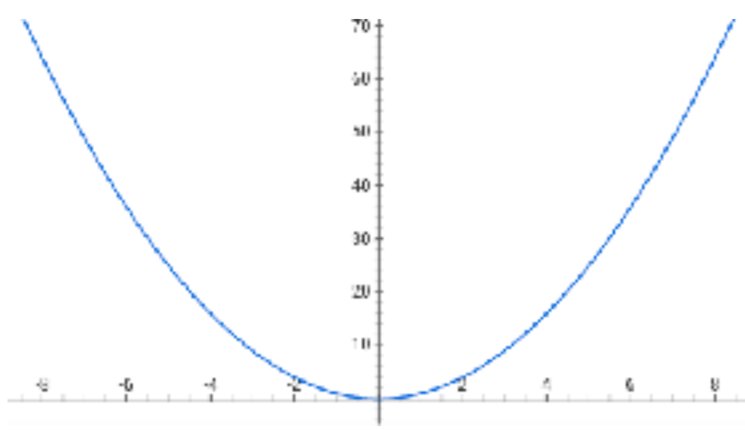


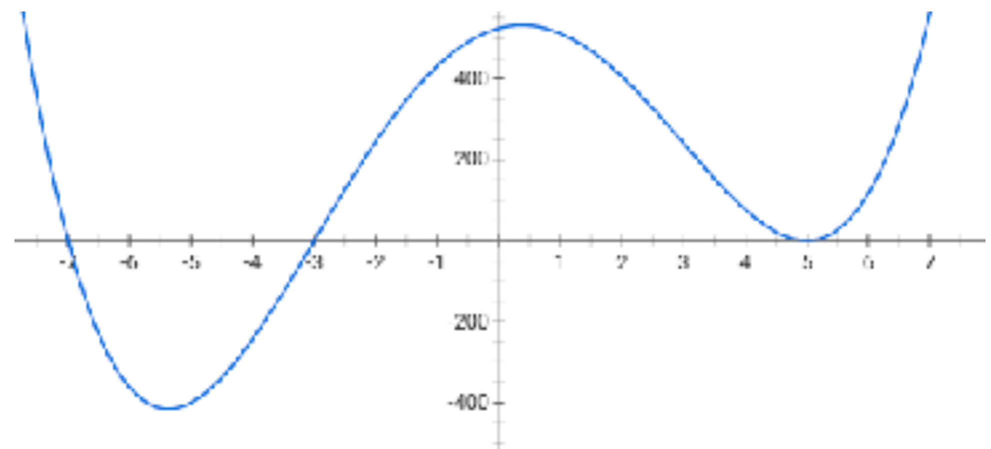


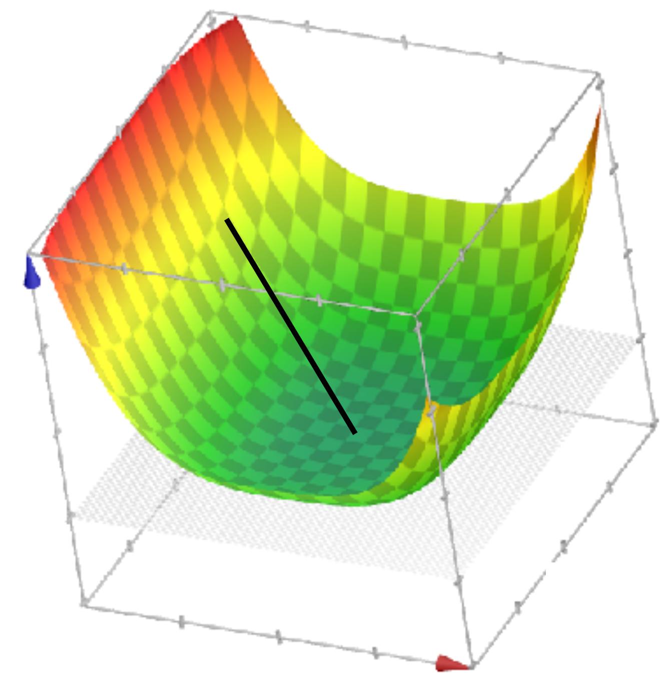


Gradient descent properties

- 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

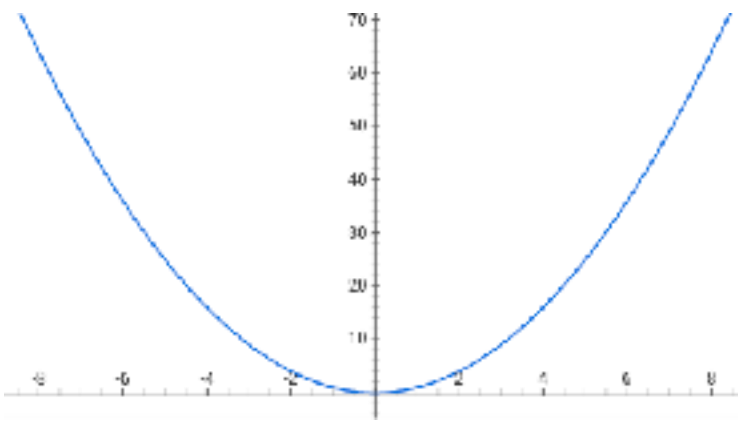


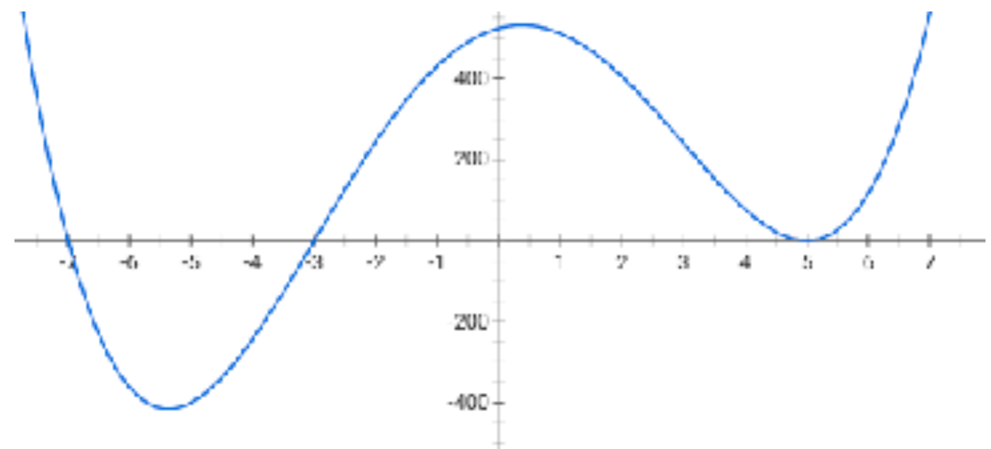


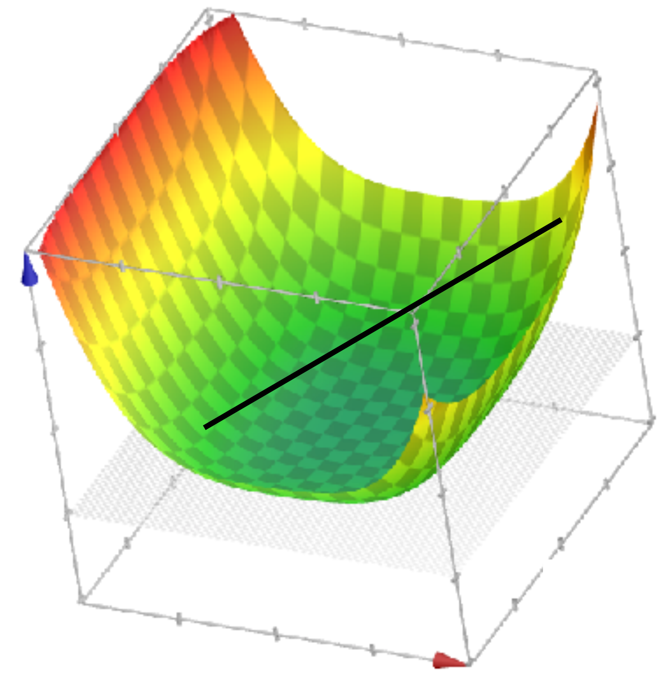


Gradient descent properties

- 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

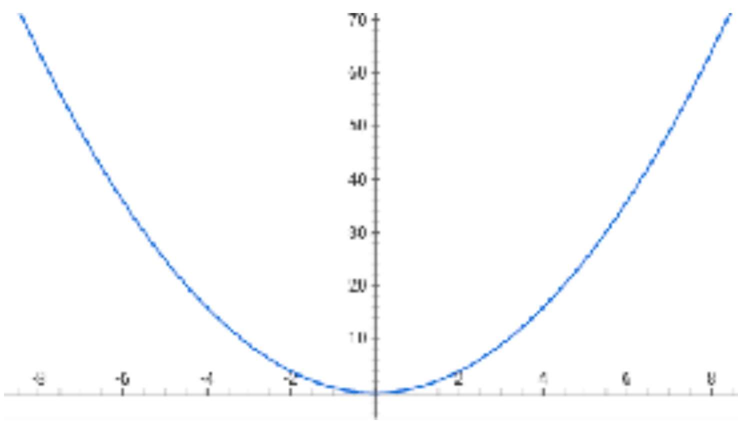


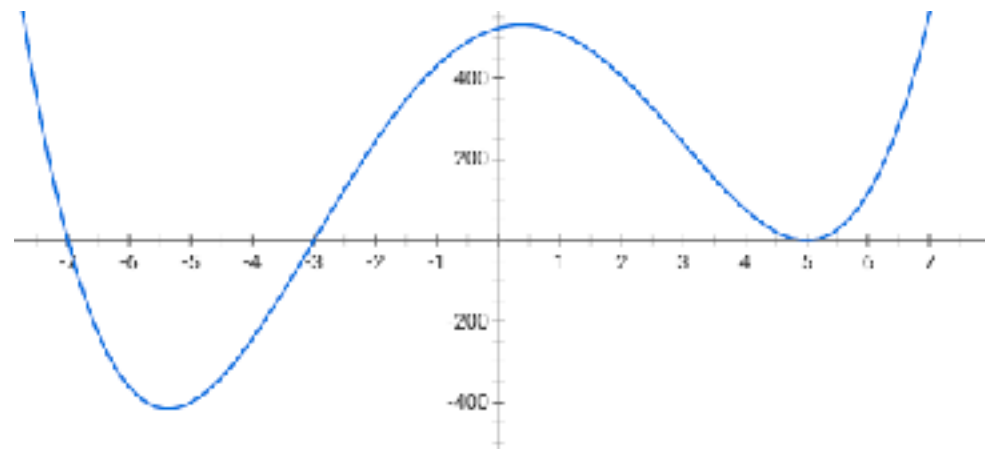


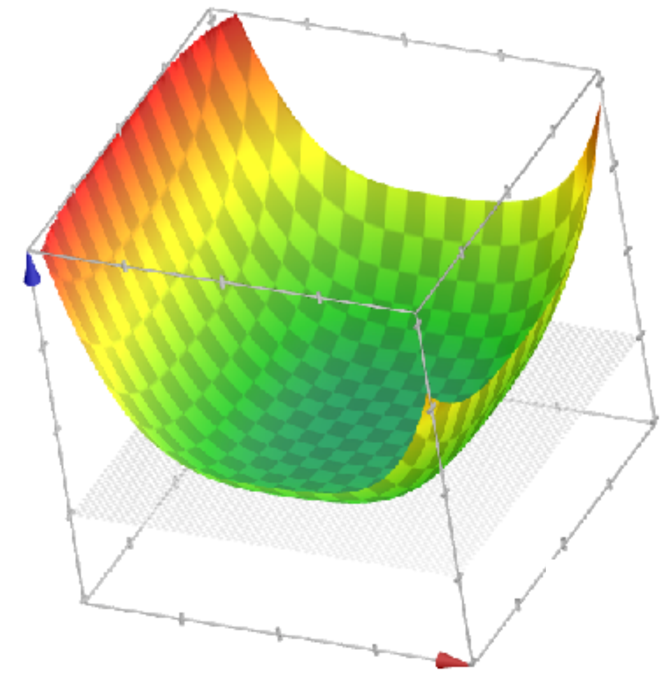


Gradient descent properties

- 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

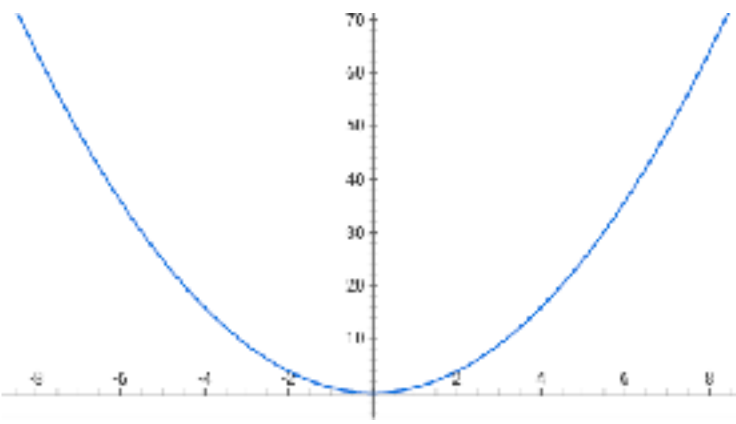


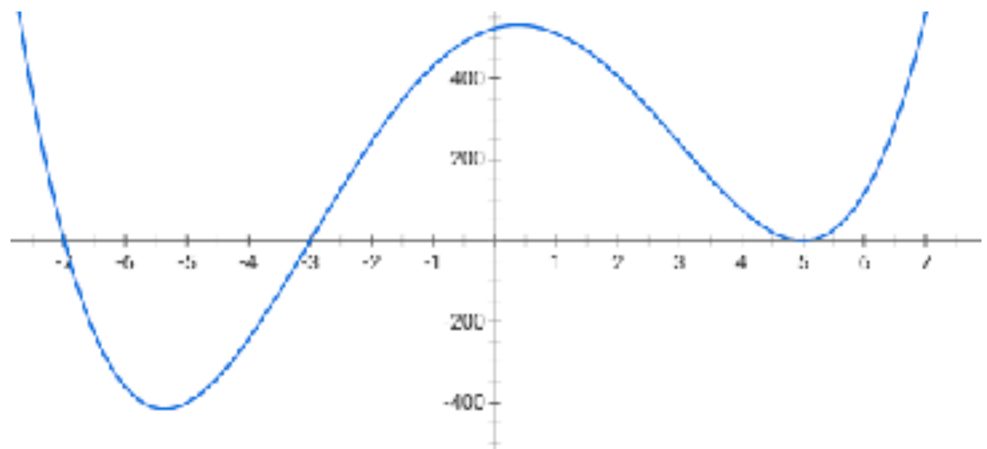


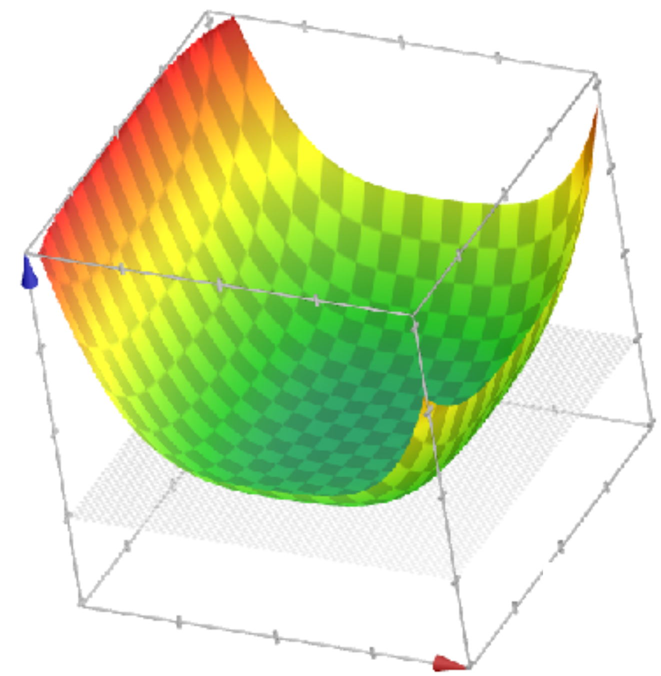


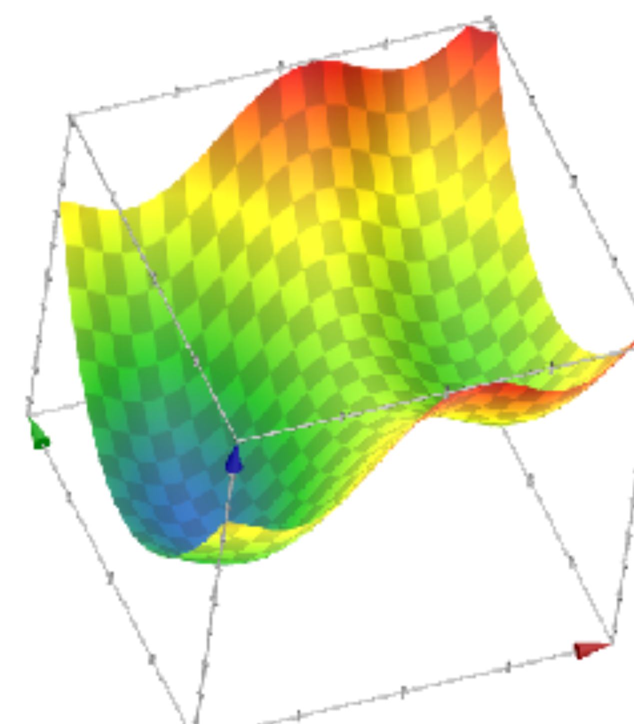
Gradient descent properties

- 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



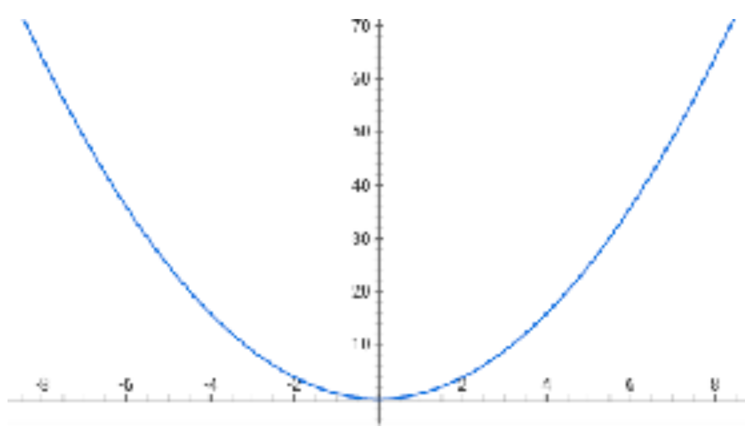


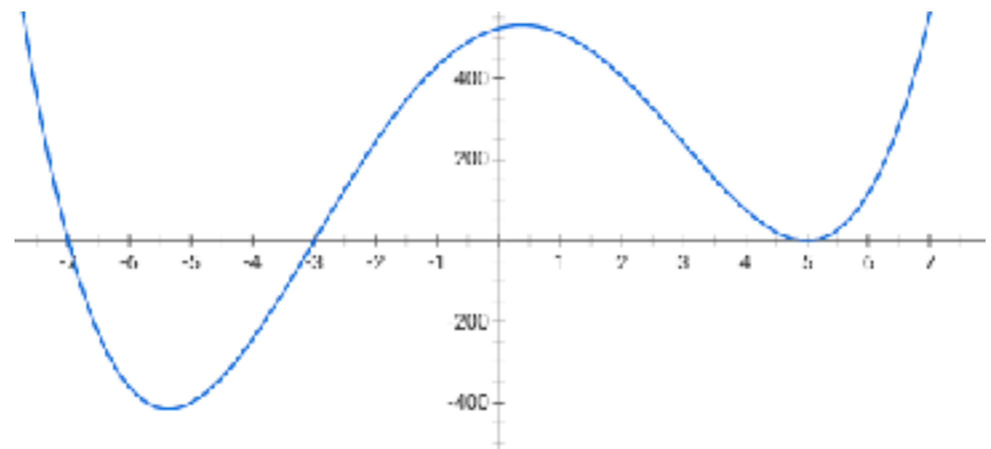


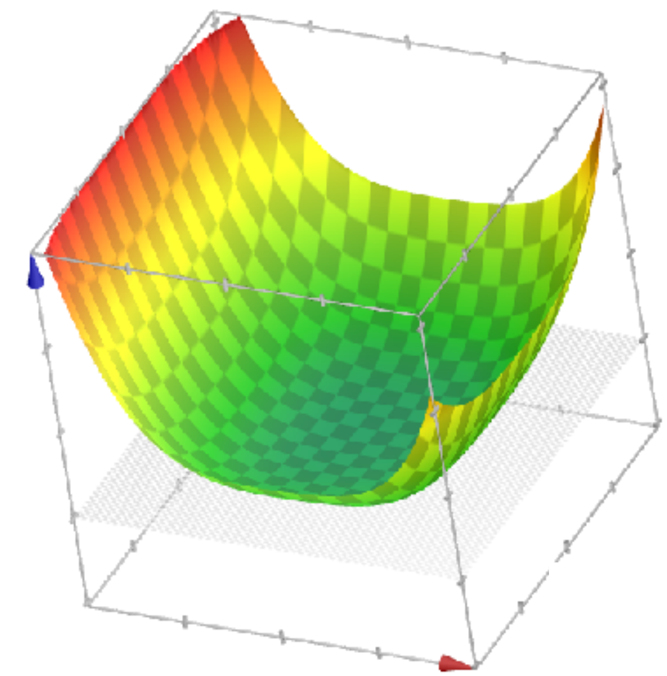


Gradient descent properties

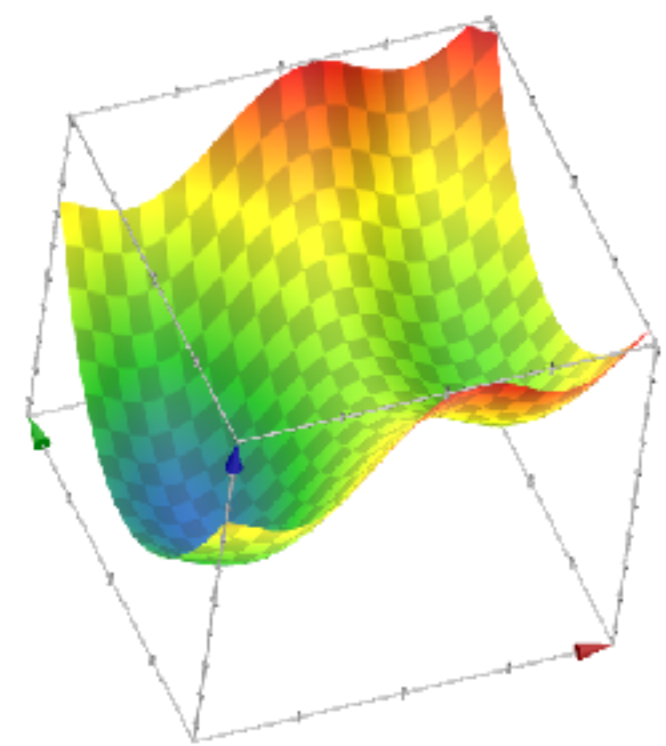
- A function *f* on $\mathbb{R}^m$ is convex if any line segment connecting two points of the graph of *f* lies above or on the graph





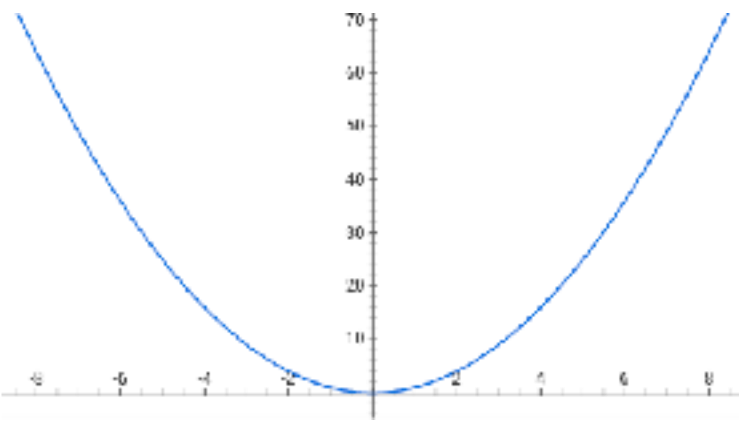


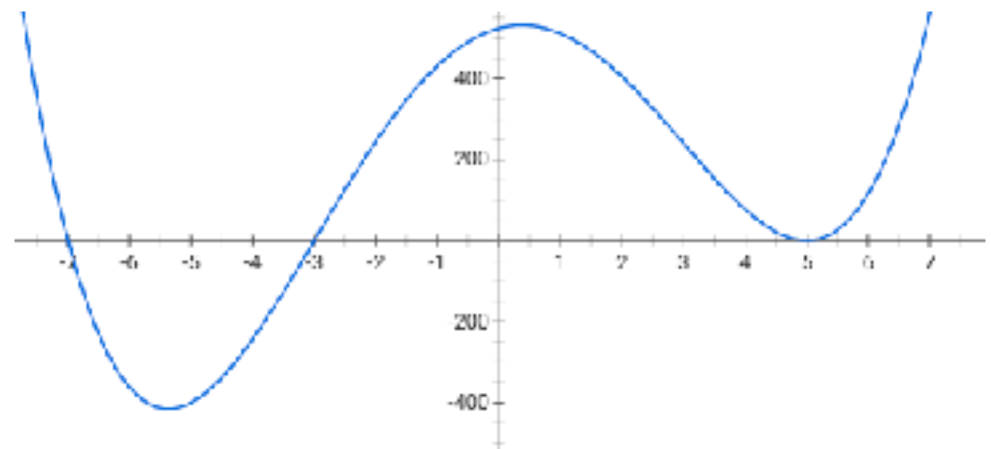
- **Theorem**: Gradient descent performance

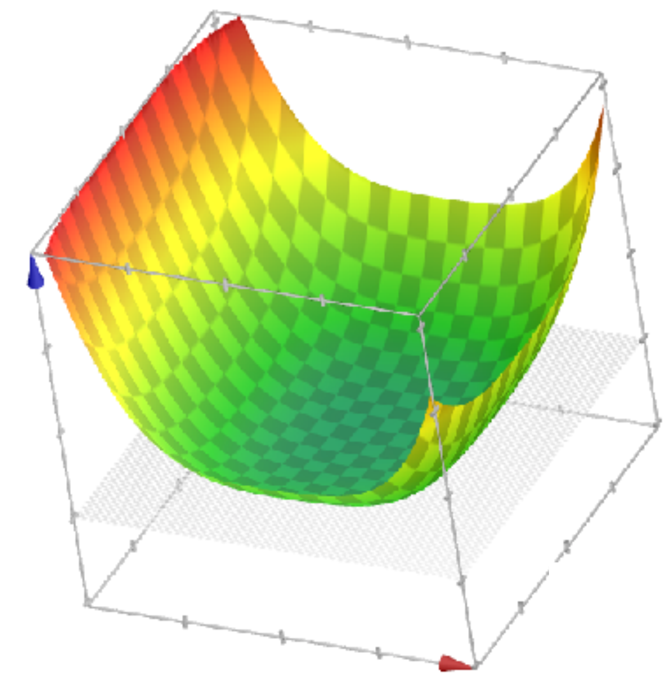


Gradient descent properties

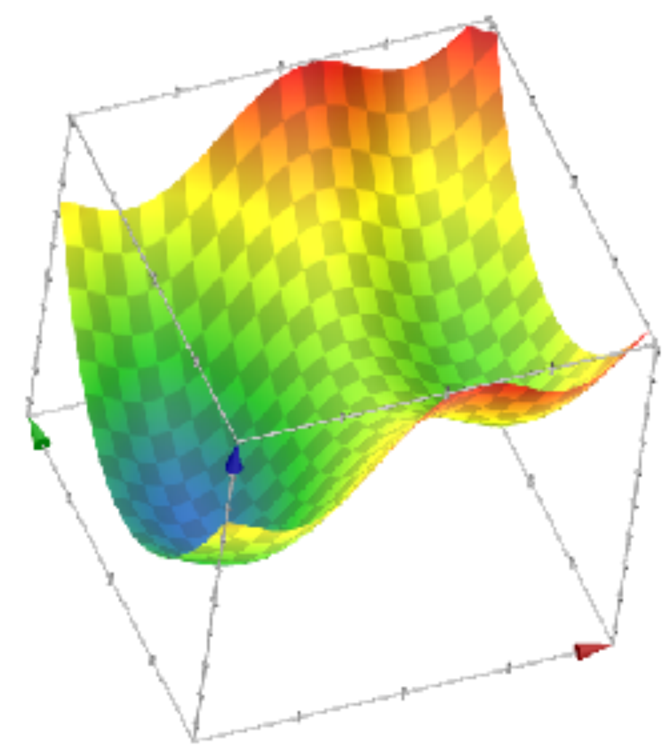
- A function *f* on $\mathbb{R}^m$ is convex if any line segment connecting two points of the graph of *f* lies above or on the graph





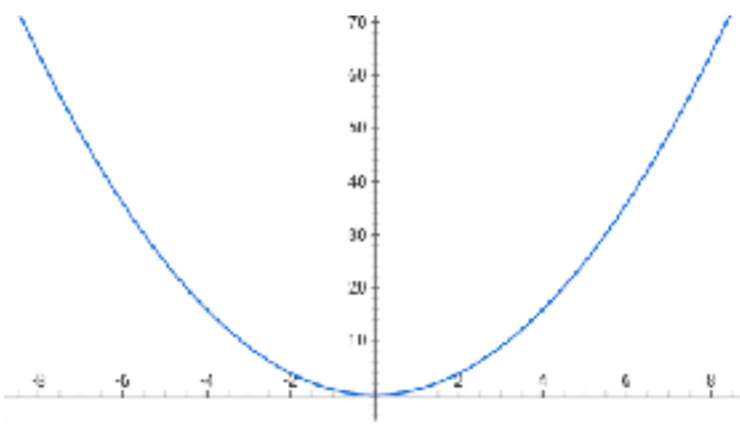
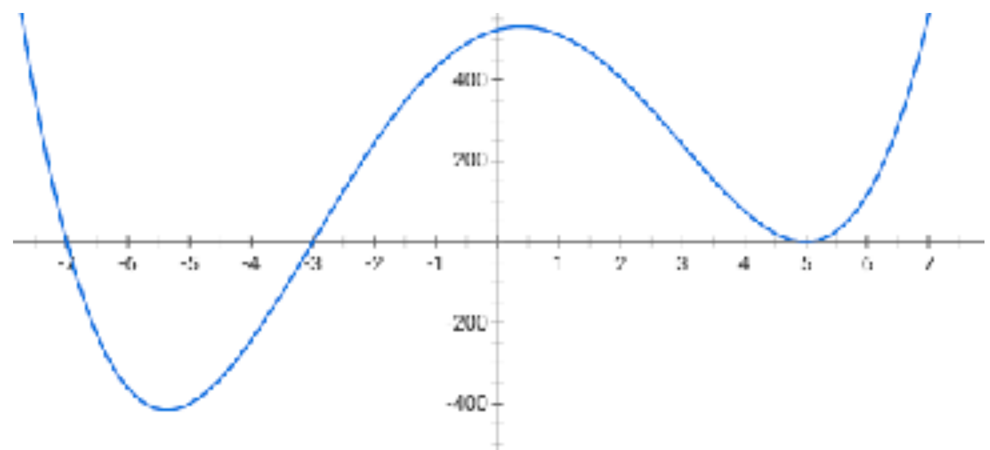
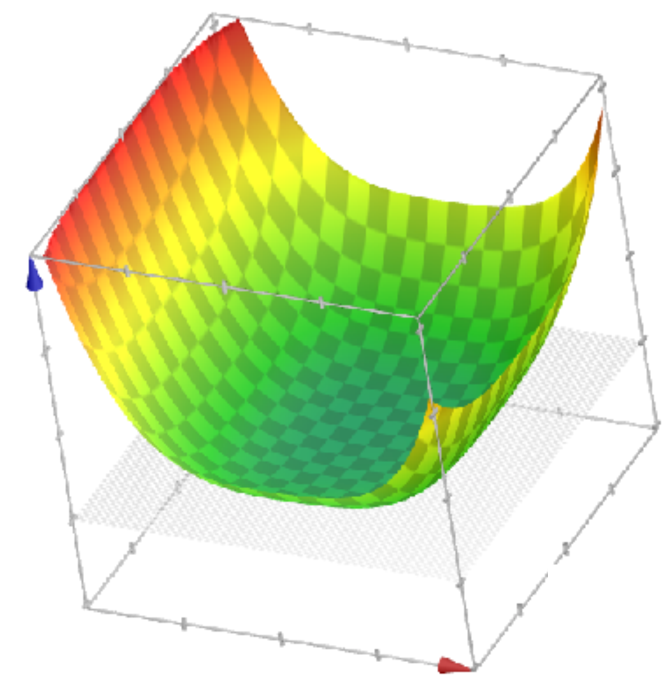


- **Theorem**: Gradient descent performance
  - **Assumptions**:

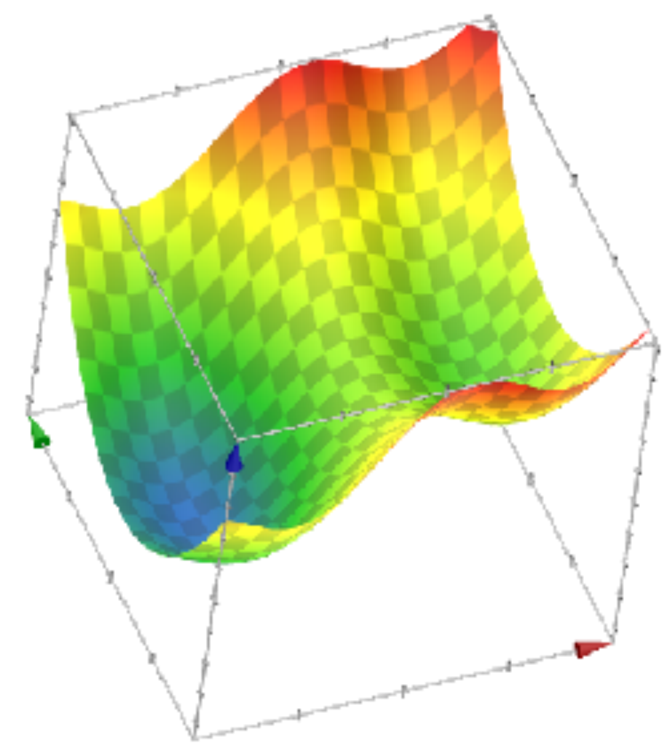


Gradient descent properties

- 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

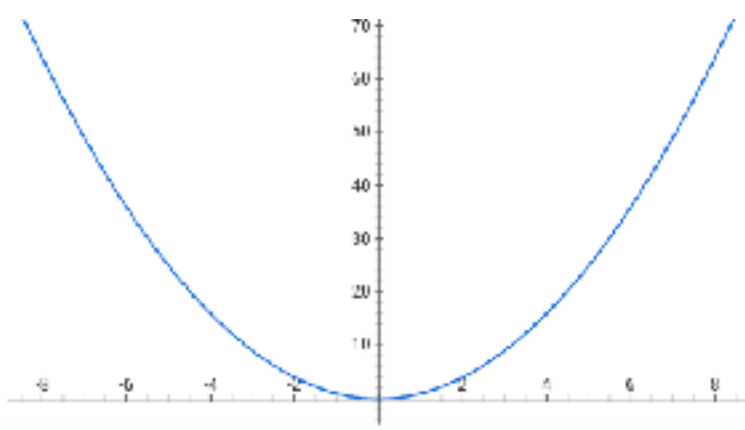
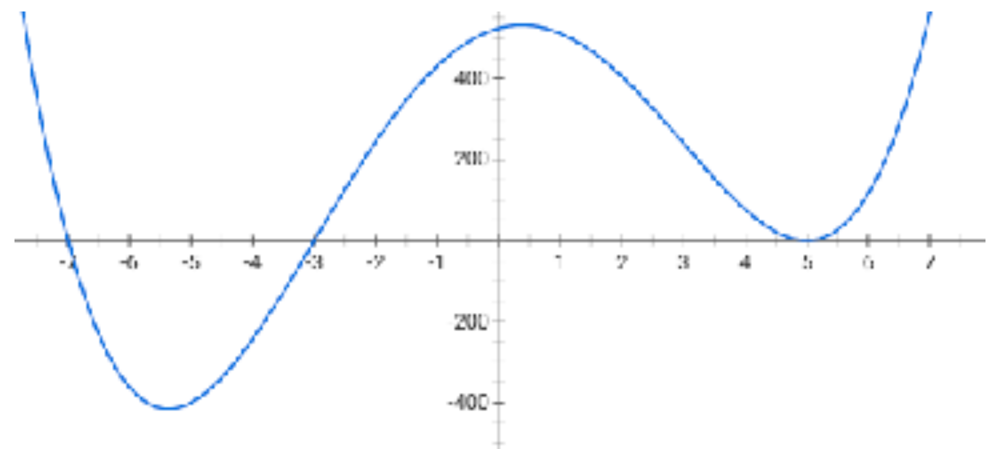
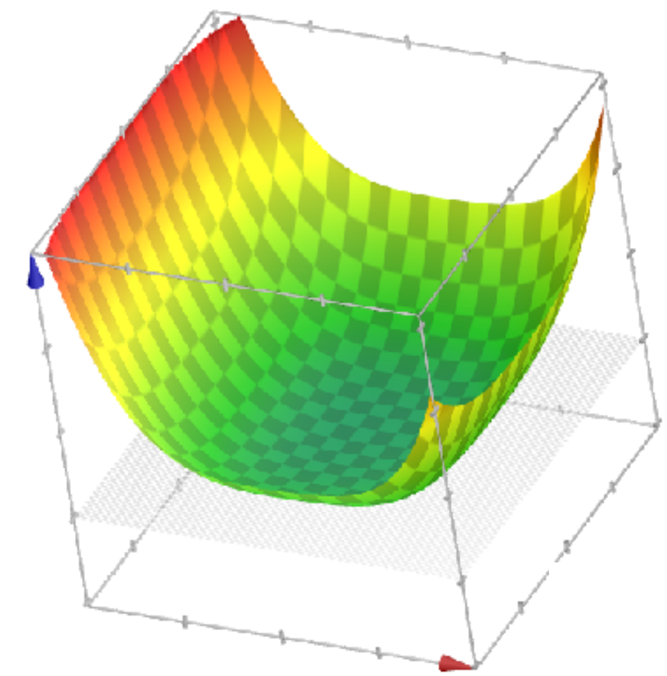




- **Theorem**: Gradient descent performance
  - **Assumptions**: (Choose any $\tilde{\epsilon} > 0$)

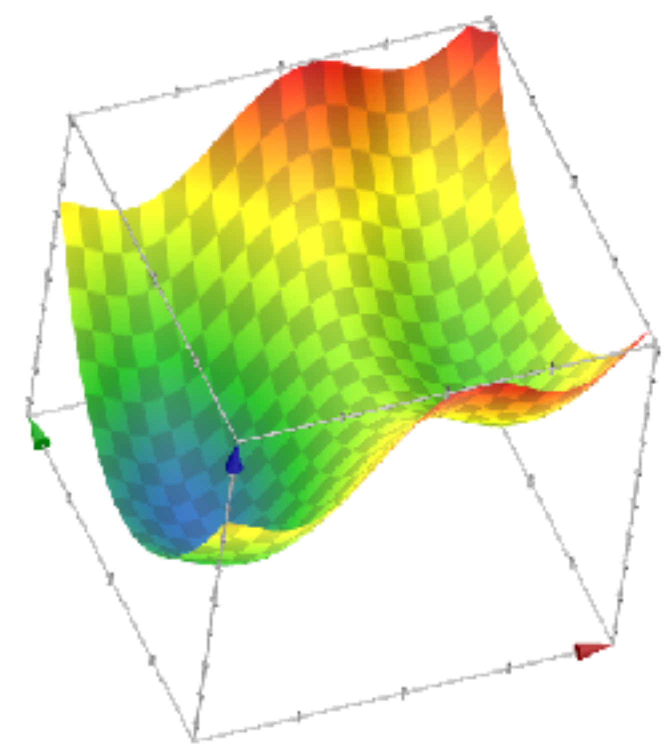


Gradient descent properties

- 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

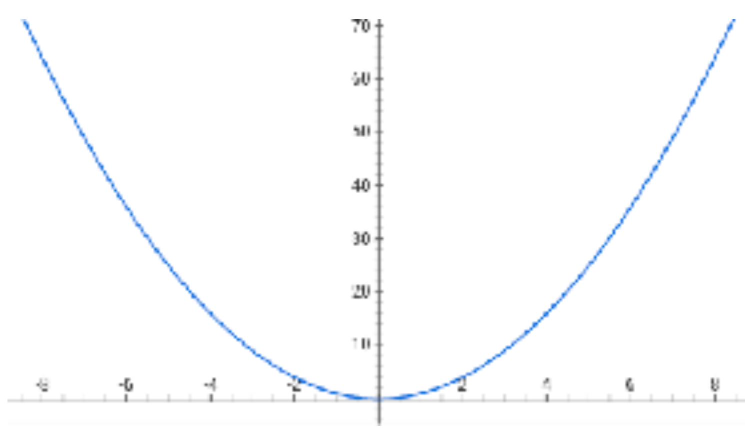




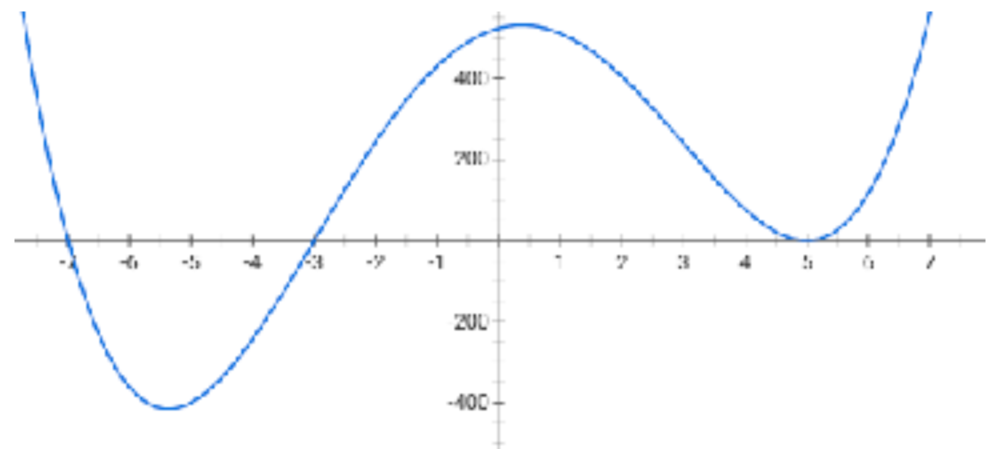
- **Theorem**: Gradient descent performance
  - **Assumptions**: (Choose any $\tilde{\epsilon} > 0$)
    - *f* is sufficiently "smooth" and convex

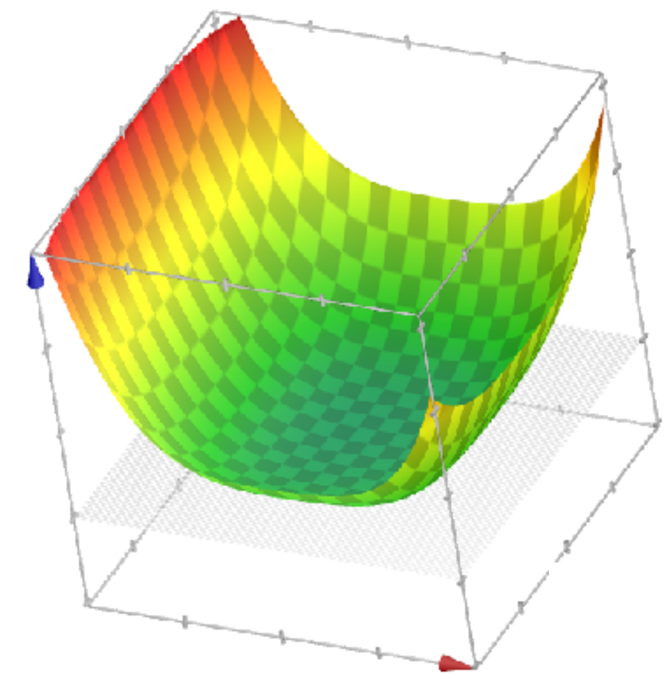


Gradient descent properties

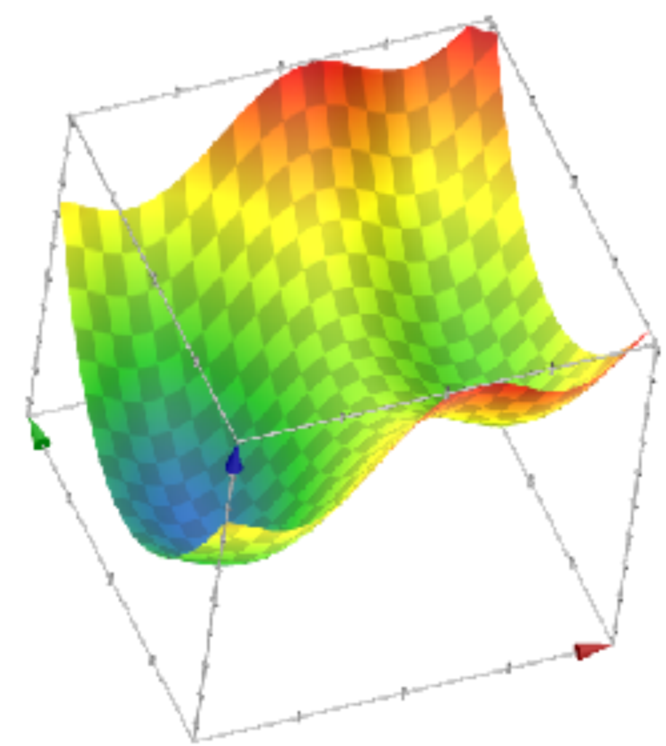
- A function *f* on $\mathbb{R}^m$ is convex if any line segment connecting two points of the graph of *f* lies above or on the graph





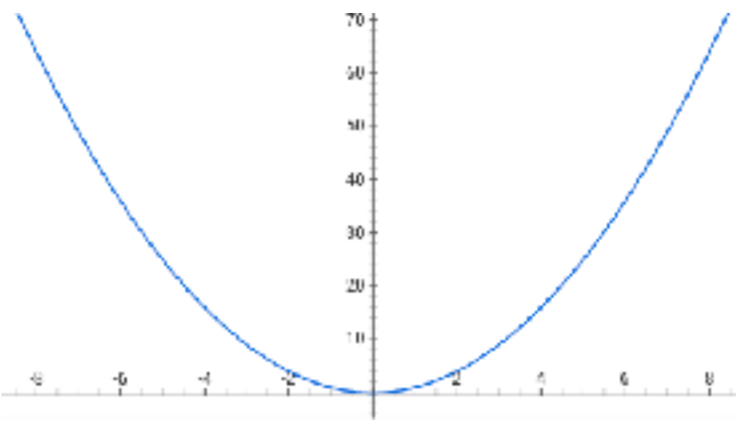


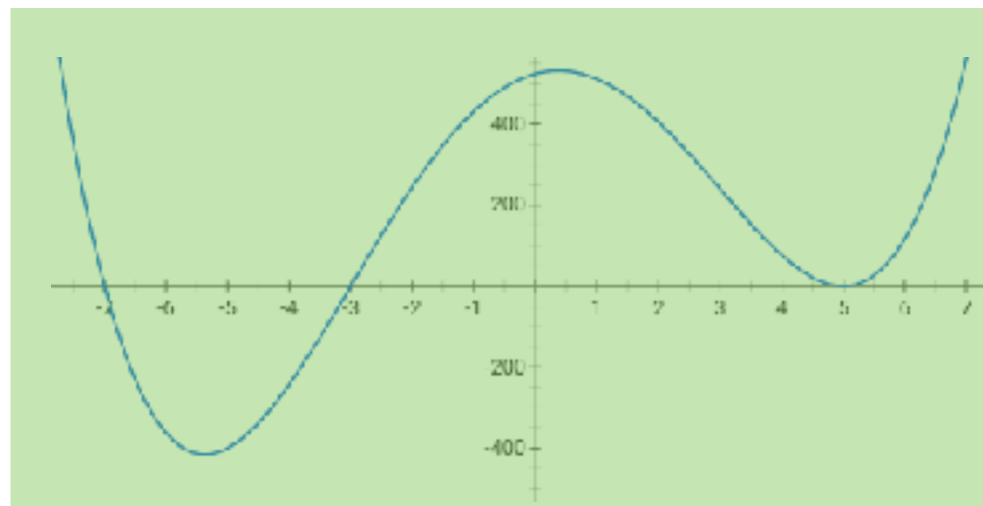
- **Theorem**: Gradient descent performance
  - **Assumptions**: (Choose any $\tilde{\epsilon} > 0$)
    - *f* is sufficiently "smooth" and convex
    - *f* has at least one global optimum

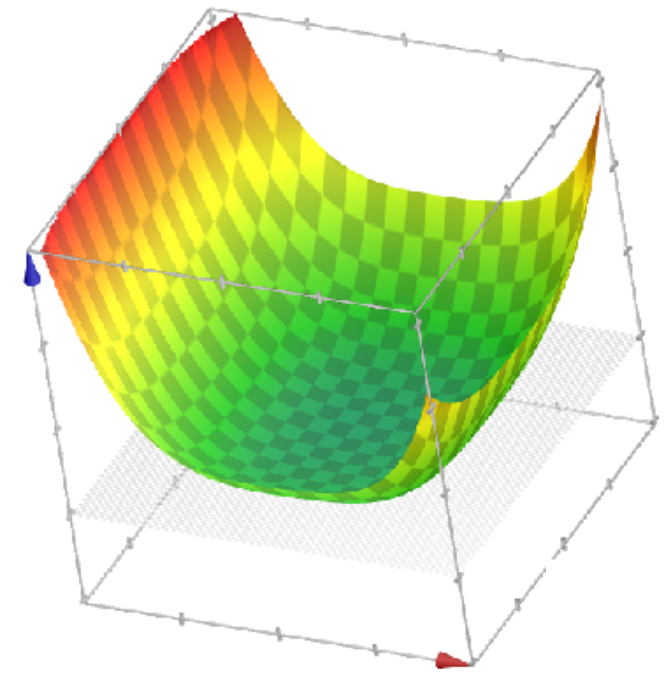


Gradient descent properties

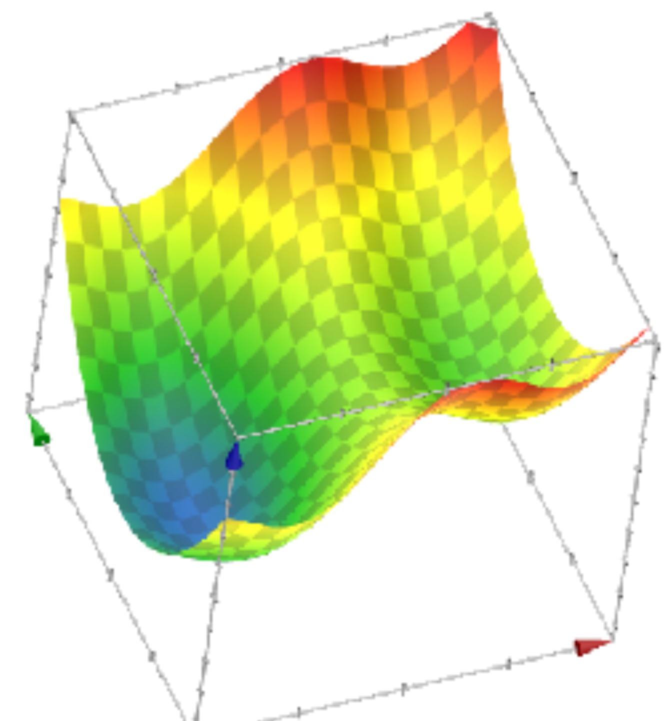
- A function *f* on $\mathbb{R}^m$ is convex if any line segment connecting two points of the graph of *f* lies above or on the graph





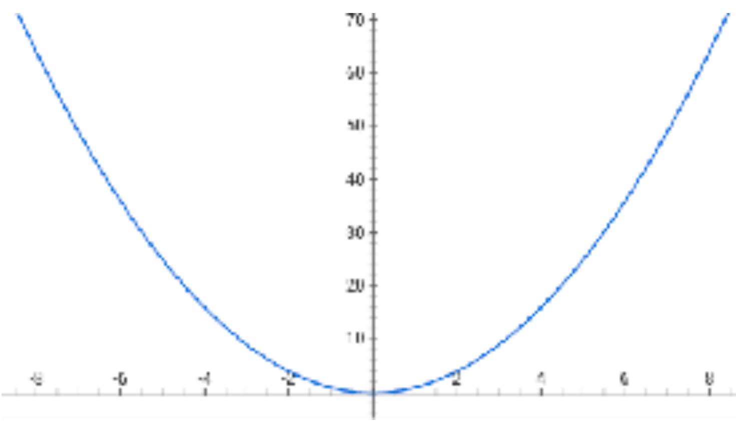


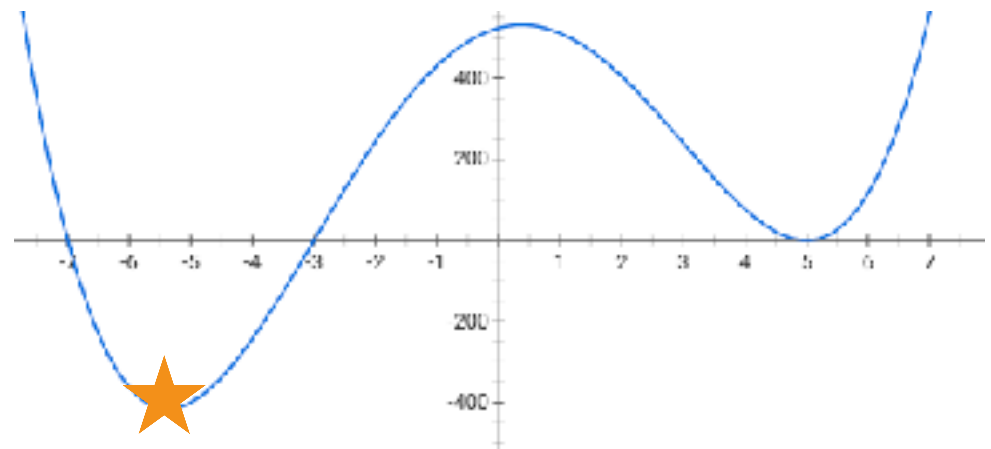
- **Theorem**: Gradient descent performance
  - **Assumptions**: (Choose any $\tilde{\epsilon} > 0$)
    - *f* is sufficiently "smooth" and convex
    - *f* has at least one global optimum

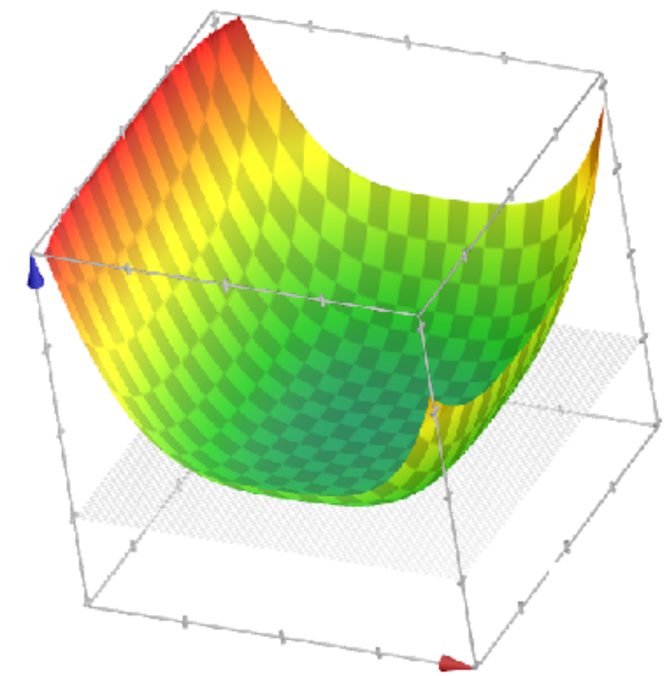


Gradient descent properties

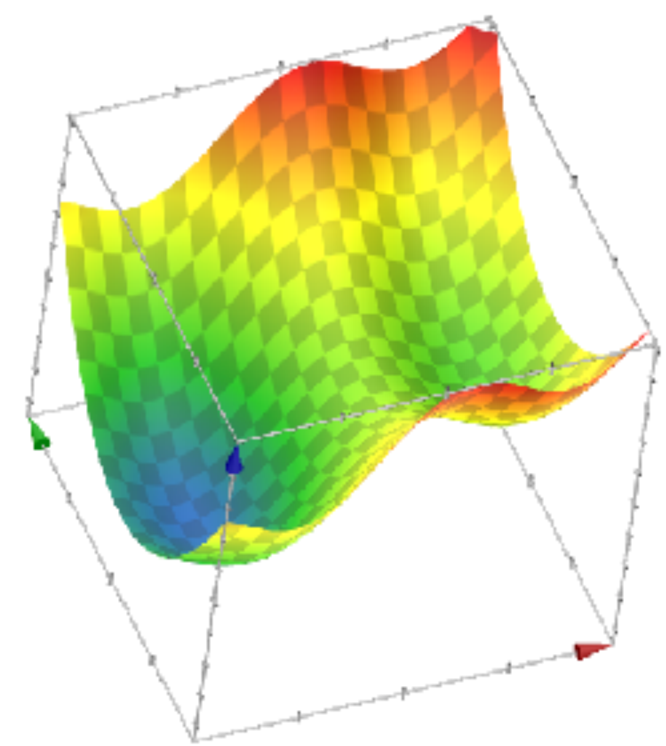
- A function *f* on $\mathbb{R}^m$ is convex if any line segment connecting two points of the graph of *f* lies above or on the graph





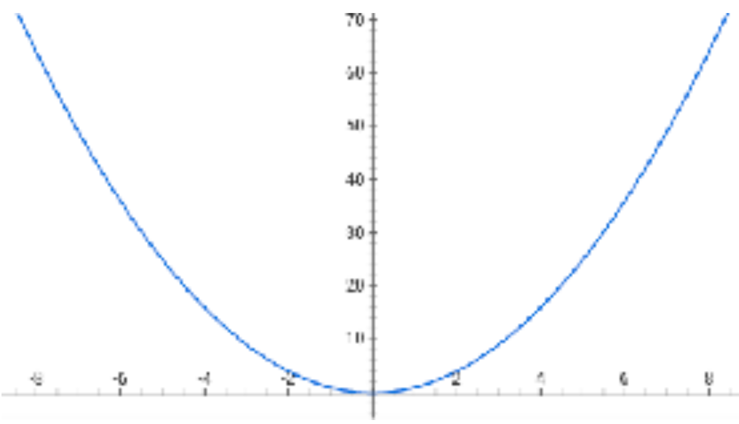


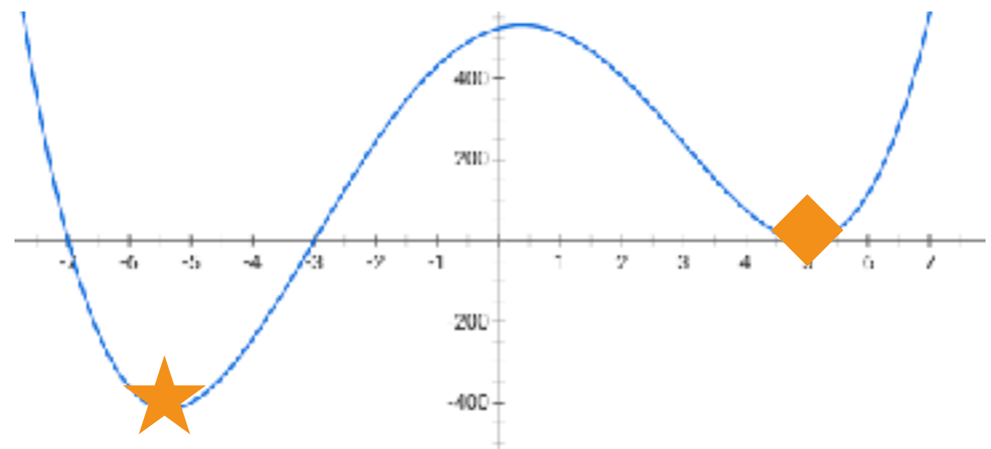
- **Theorem**: Gradient descent performance
  - **Assumptions**: (Choose any $\tilde{\epsilon} > 0$)
    - *f* is sufficiently "smooth" and convex
    - *f* has at least one global optimum

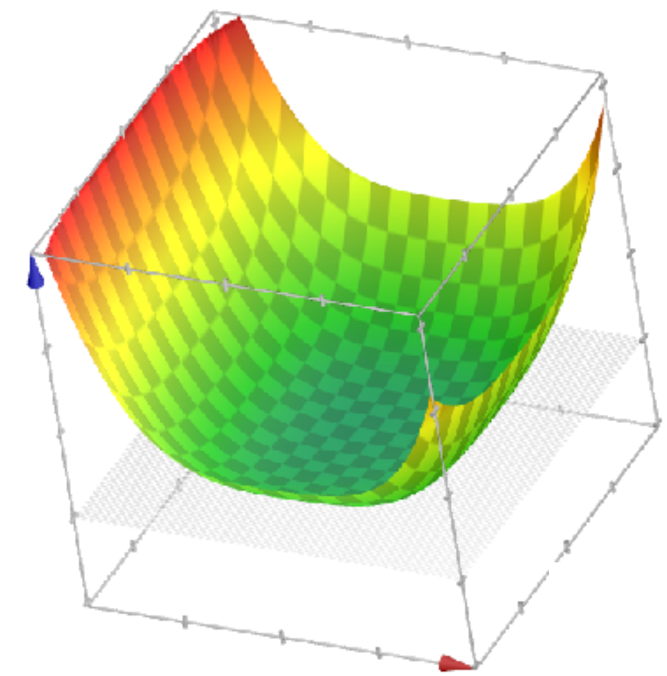


Gradient descent properties

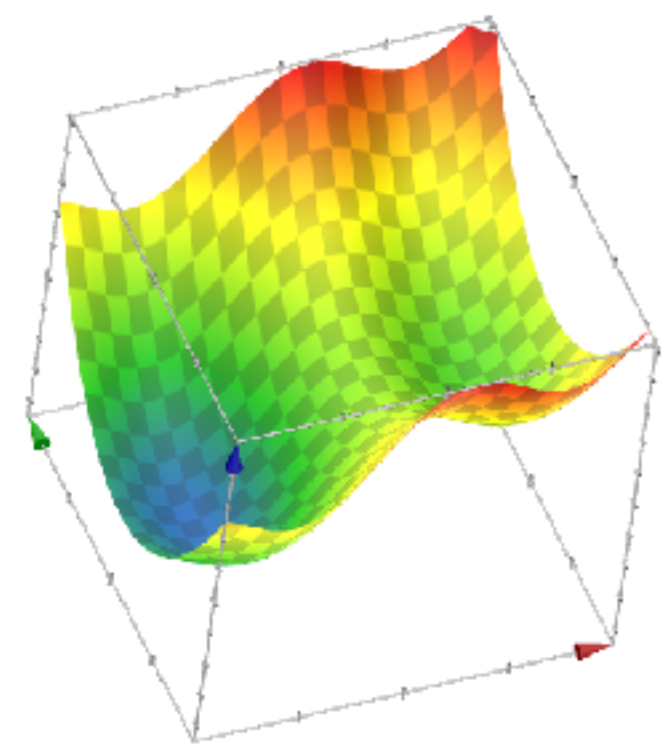
- A function *f* on $\mathbb{R}^m$ is convex if any line segment connecting two points of the graph of *f* lies above or on the graph





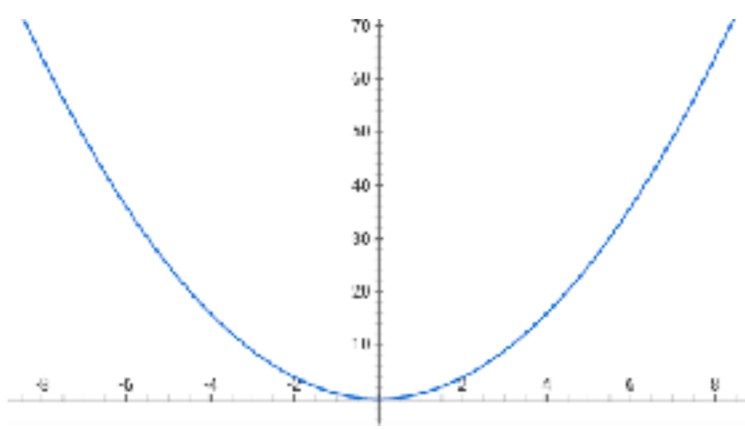


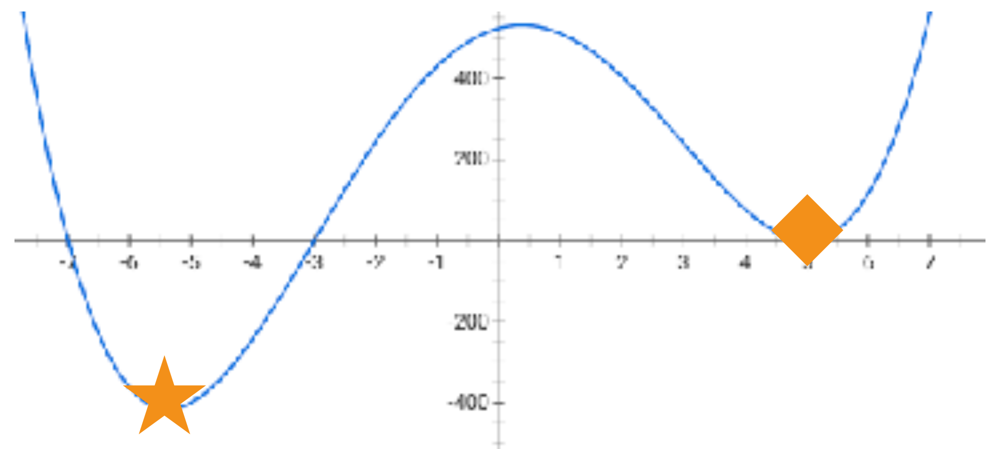
- **Theorem**: Gradient descent performance
  - **Assumptions**: (Choose any $\tilde{\epsilon} > 0$)
    - *f* is sufficiently "smooth" and convex
    - *f* has at least one global optimum

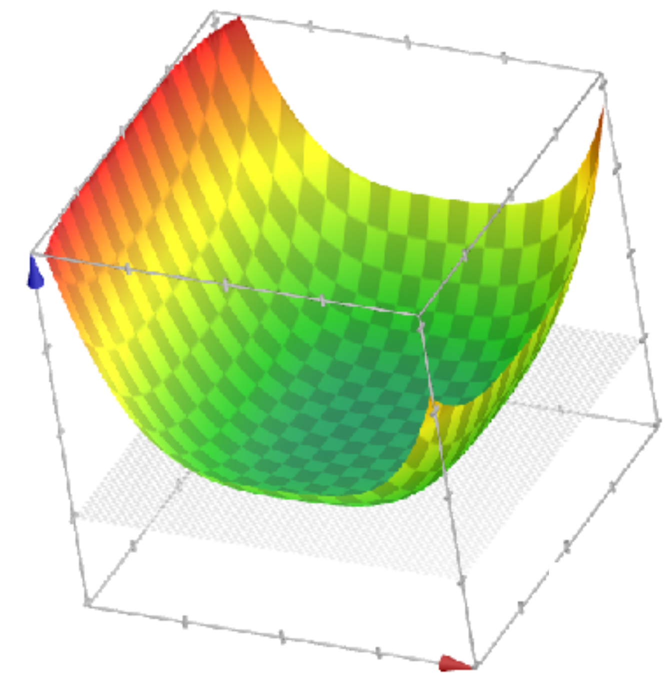


Gradient descent properties

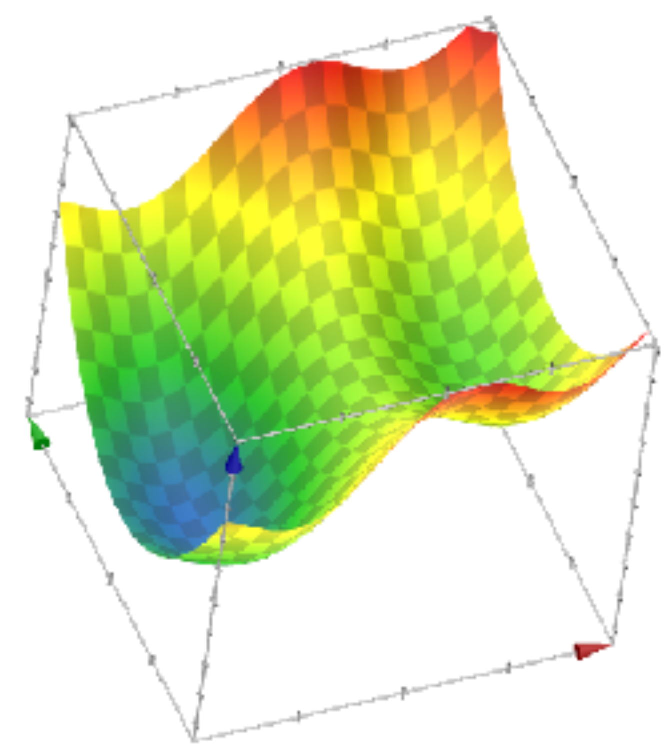
- A function *f* on $\mathbb{R}^m$ is convex if any line segment connecting two points of the graph of *f* lies above or on the graph





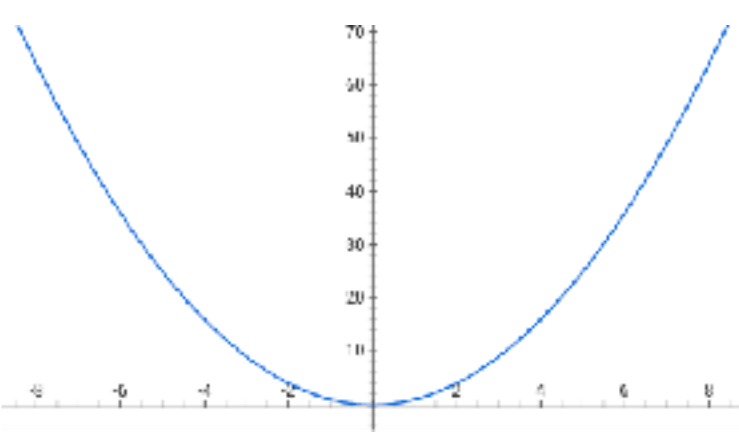


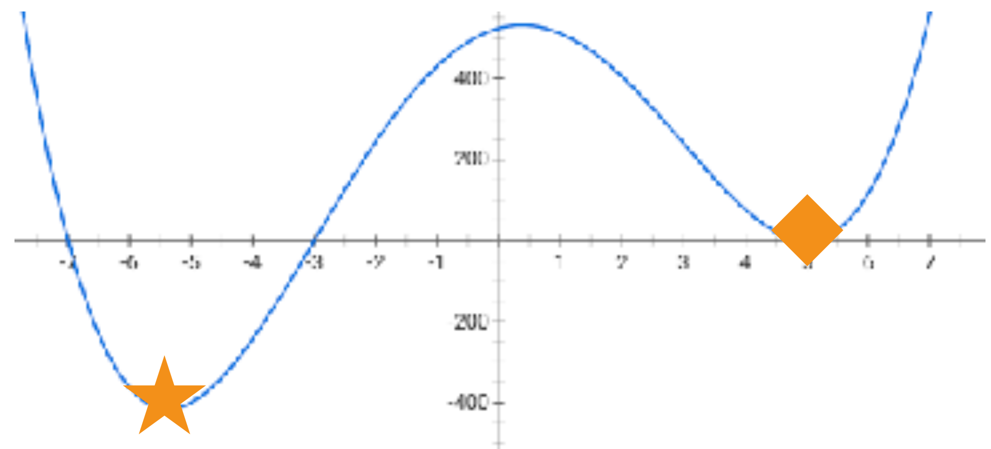
- **Theorem**: Gradient descent performance
  - **Assumptions**: (Choose any $\tilde{\epsilon} > 0$)
    - *f* is sufficiently "smooth" and convex
    - *f* has at least one global optimum
    - η is sufficiently small

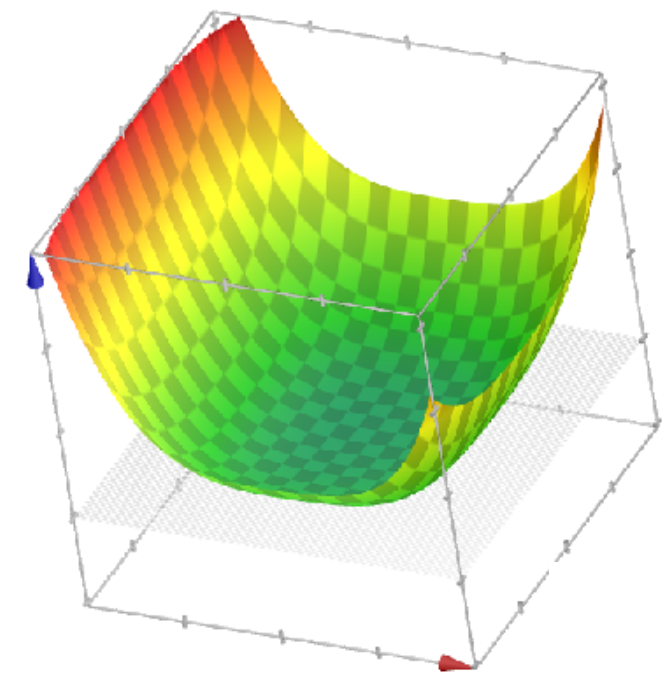


Gradient descent properties

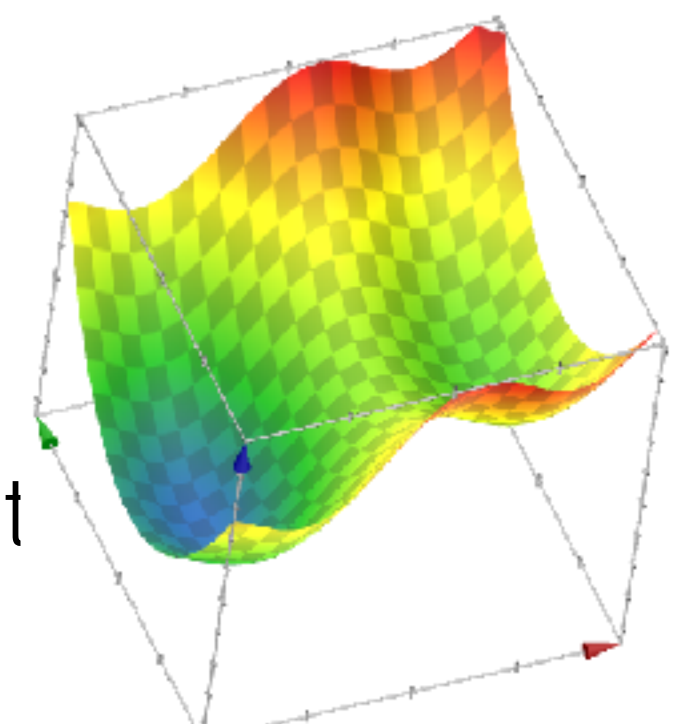
- A function *f* on $\mathbb{R}^m$ is convex if any line segment connecting two points of the graph of *f* lies above or on the graph







- **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

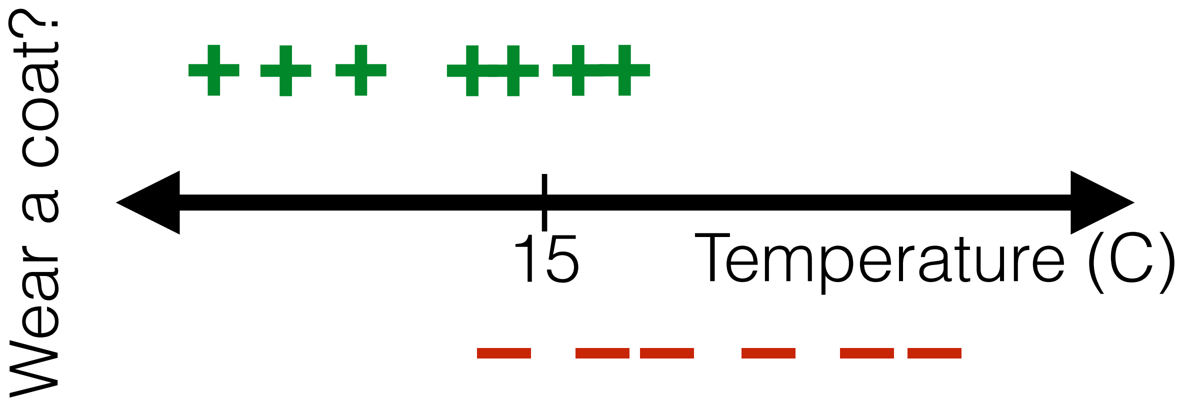
- Loss $J_{\text{lr}}(\Theta) = J_{\text{lr}}(\theta, \theta_0)$ is differentiable & convex

Gradient descent for logistic regression

- Loss $J_{\text{lr}}(\Theta) = J_{\text{lr}}(\theta, \theta_0)$ is differentiable & convex
- Run `Gradient-Descent(` $\Theta_{\text{init}}, \eta, J_{lr}, \nabla_{\Theta} J_{lr}, \epsilon$ `)`

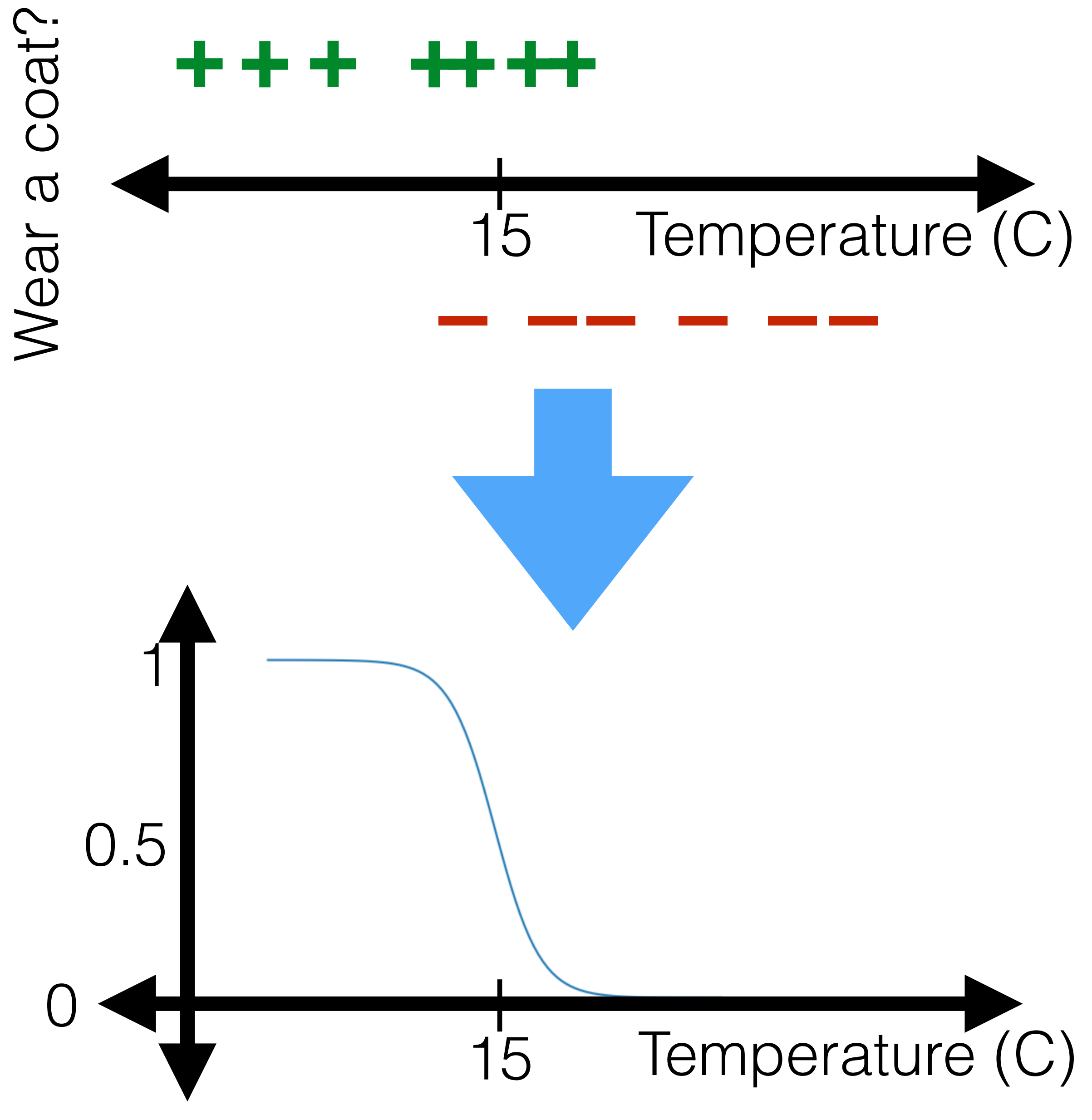
Gradient descent for logistic regression

- Loss $J_{\text{lr}}(\Theta) = J_{\text{lr}}(\theta, \theta_0)$ is differentiable & convex
- Run `Gradient-Descent(` $\Theta_{\text{init}}, \eta, J_{lr}, \nabla_\Theta J_{lr}, \epsilon$ `)`



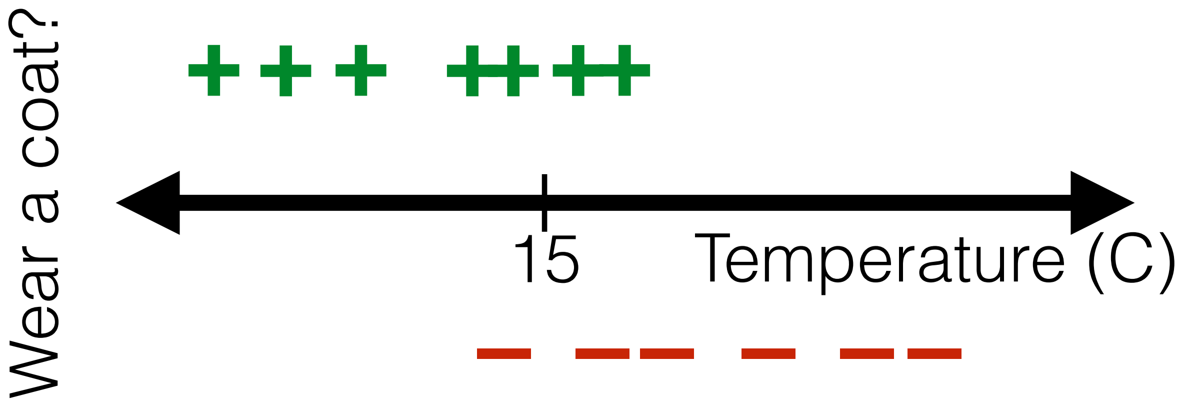
Gradient descent for logistic regression

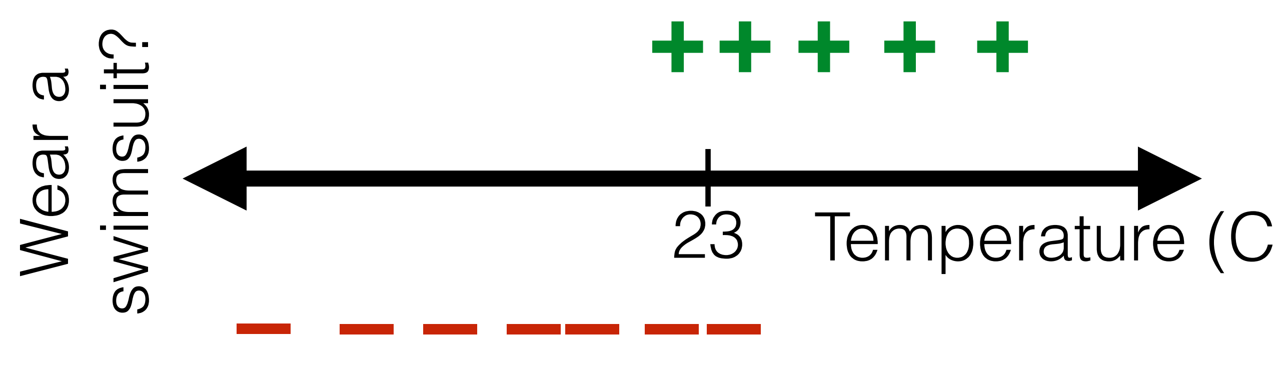
- Loss $J_{\text{lr}}(\Theta) = J_{\text{lr}}(\theta, \theta_0)$ is differentiable & convex
- Run `Gradient-Descent(` $\Theta_{\text{init}}, \eta, J_{lr}, \nabla_\Theta J_{lr}, \epsilon$ `)`

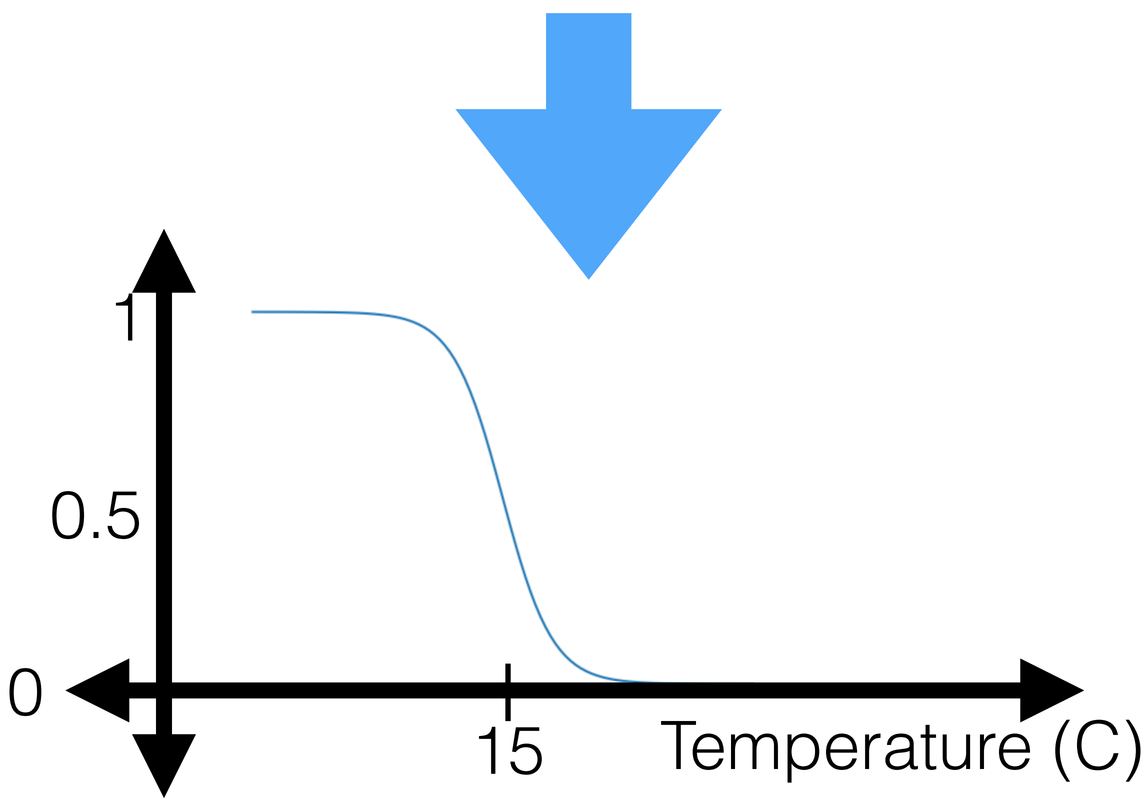


Gradient descent for logistic regression

- Loss $J_{\text{lr}}(\Theta) = J_{\text{lr}}(\theta, \theta_0)$ is differentiable & convex
- Run `Gradient-Descent(` $\Theta_{\text{init}}, \eta, J_{lr}, \nabla_\Theta J_{lr}, \epsilon$ `)`

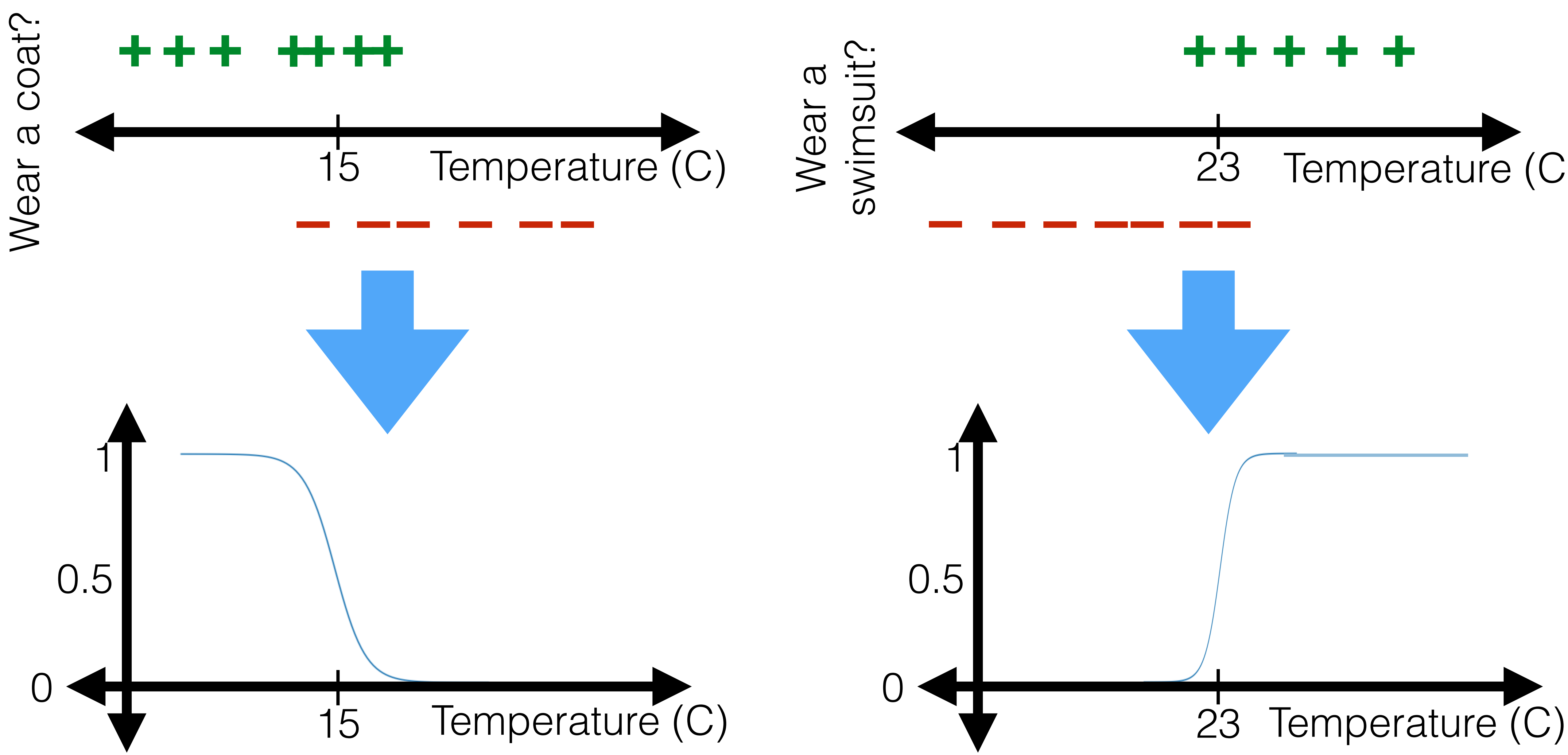






Gradient descent for logistic regression

- Loss $J_{\text{lr}}(\Theta) = J_{\text{lr}}(\theta, \theta_0)$ is differentiable & convex
- Run `Gradient-Descent(` $\Theta_{\text{init}}, \eta, J_{lr}, \nabla_\Theta J_{lr}, \epsilon$ `)`

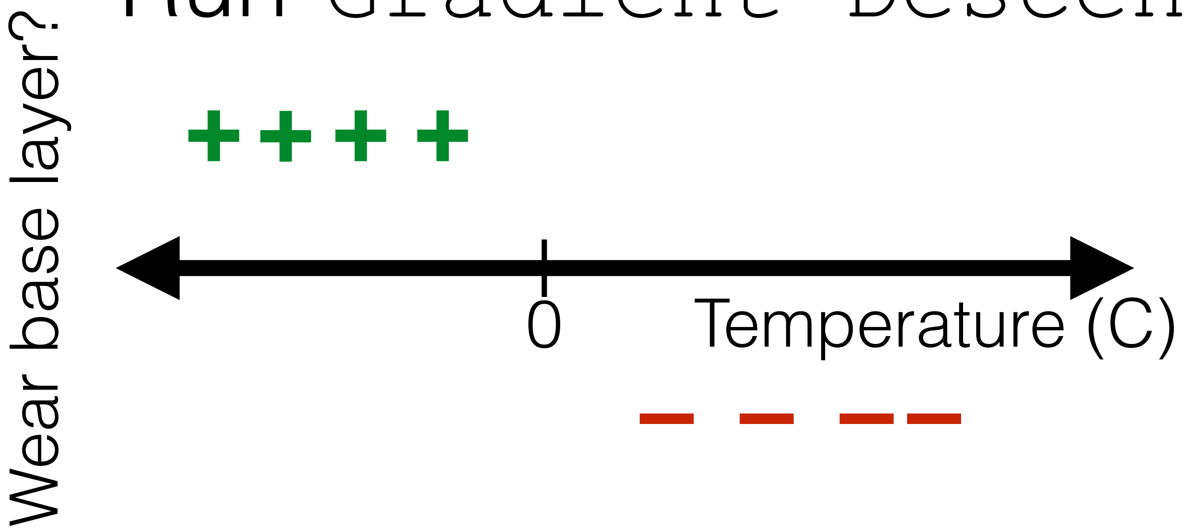


Gradient descent for logistic regression

- Loss $J_{\text{lr}}(\Theta) = J_{\text{lr}}(\theta, \theta_0)$ is differentiable & convex
- Run `Gradient-Descent(` $\Theta_{\text{init}}, \eta, J_{lr}, \nabla_\Theta J_{lr}, \epsilon$ `)`

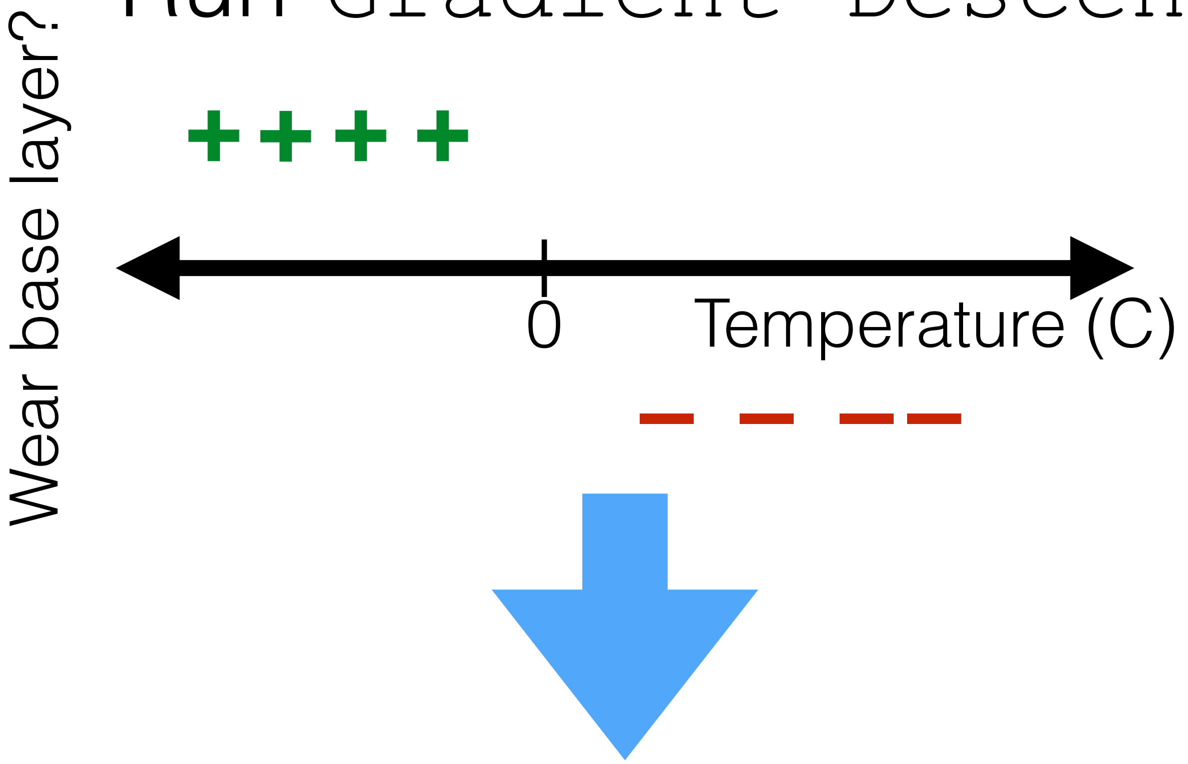
Gradient descent for logistic regression

- Loss $J_{\text{lr}}(\Theta) = J_{\text{lr}}(\theta, \theta_0)$ is differentiable & convex
- Run `Gradient-Descent(` $\Theta_{\text{init}}, \eta, J_{lr}, \nabla_{\Theta} J_{lr}, \epsilon$ `)`



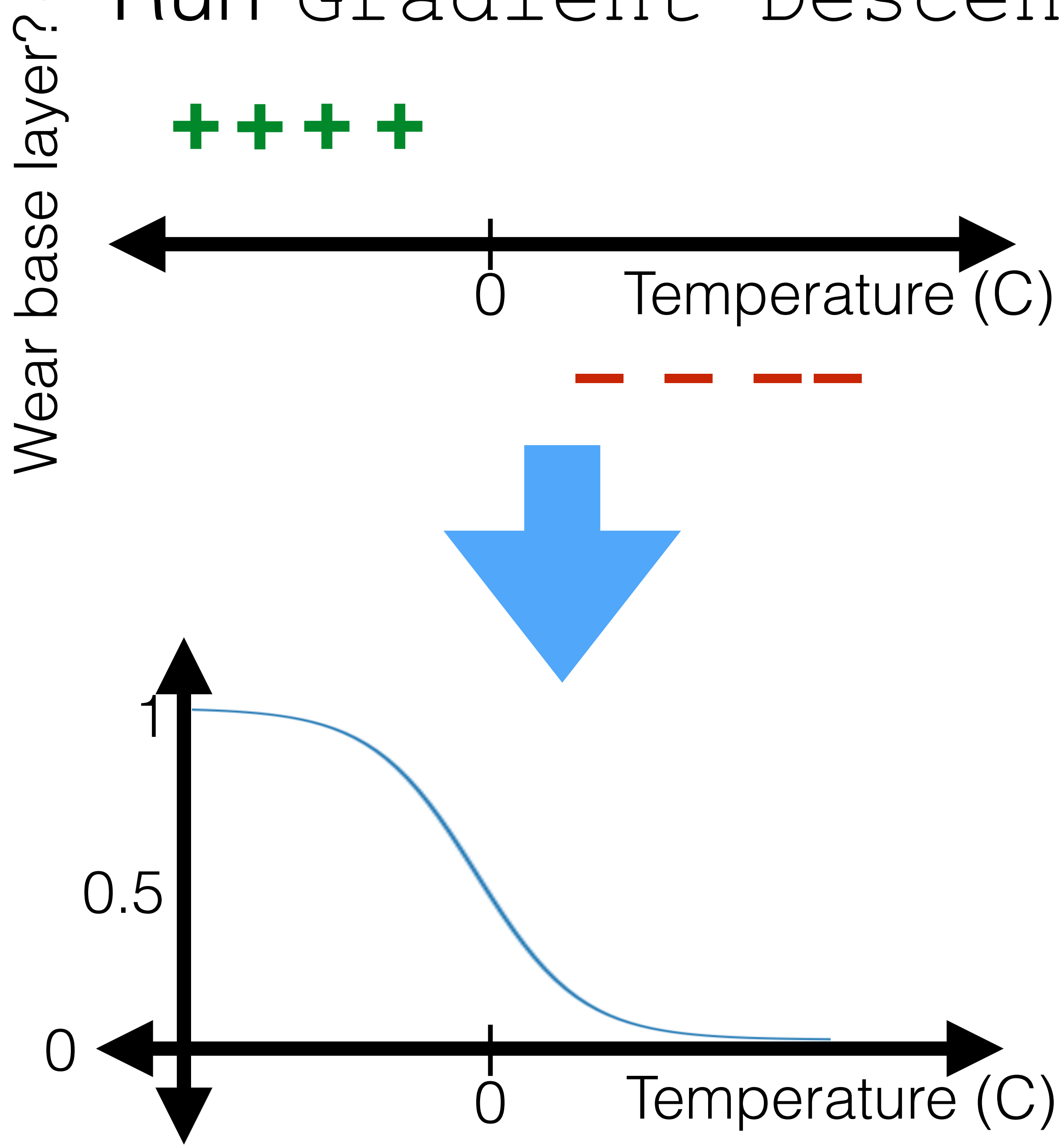
Gradient descent for logistic regression

- Loss $J_{\text{lr}}(\Theta) = J_{\text{lr}}(\theta, \theta_0)$ is differentiable & convex
- Run `Gradient-Descent(` $\Theta_{\text{init}}, \eta, J_{lr}, \nabla_\Theta J_{lr}, \epsilon$ `)`



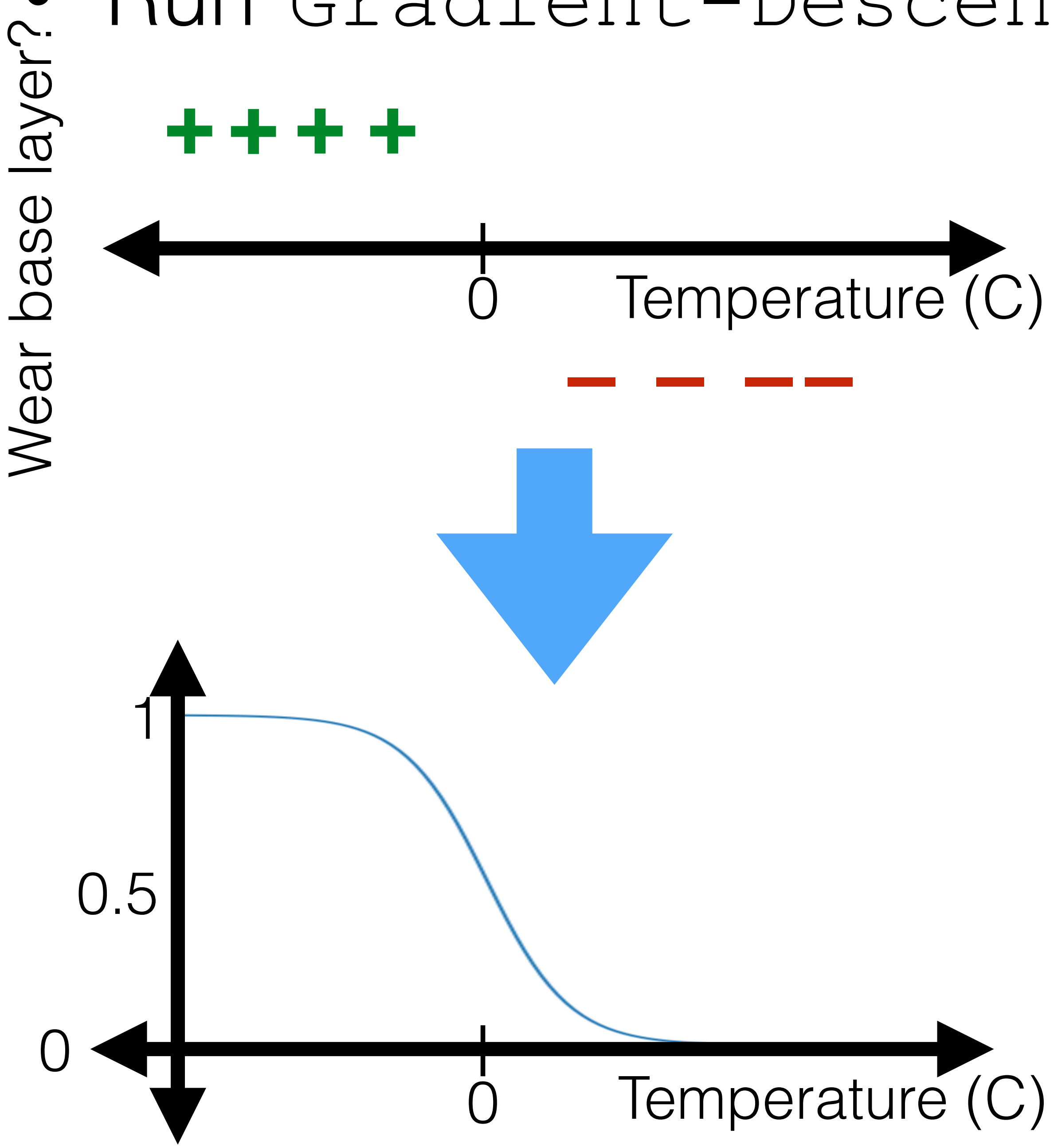
Gradient descent for logistic regression

- Loss $J_{\text{lr}}(\Theta) = J_{\text{lr}}(\theta, \theta_0)$ is differentiable & convex
- Run `Gradient-Descent(` $\Theta_{\text{init}}, \eta, J_{lr}, \nabla_{\Theta} J_{lr}, \epsilon$ `)`



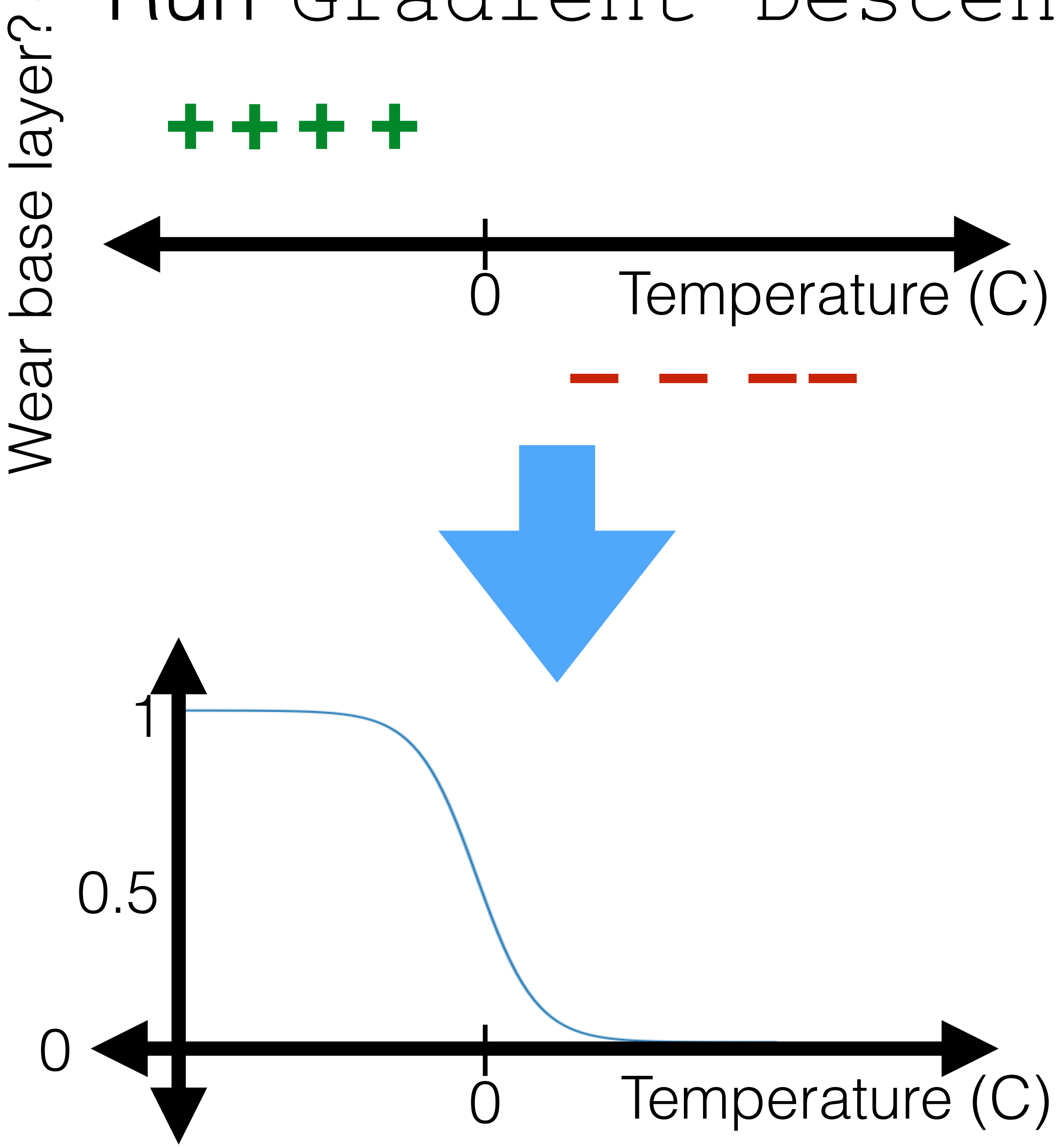
Gradient descent for logistic regression

- Loss $J_{\text{lr}}(\Theta) = J_{\text{lr}}(\theta, \theta_0)$ is differentiable & convex
- Run `Gradient-Descent(` $\Theta_{\text{init}}, \eta, J_{lr}, \nabla_\Theta J_{lr}, \epsilon$ `)`



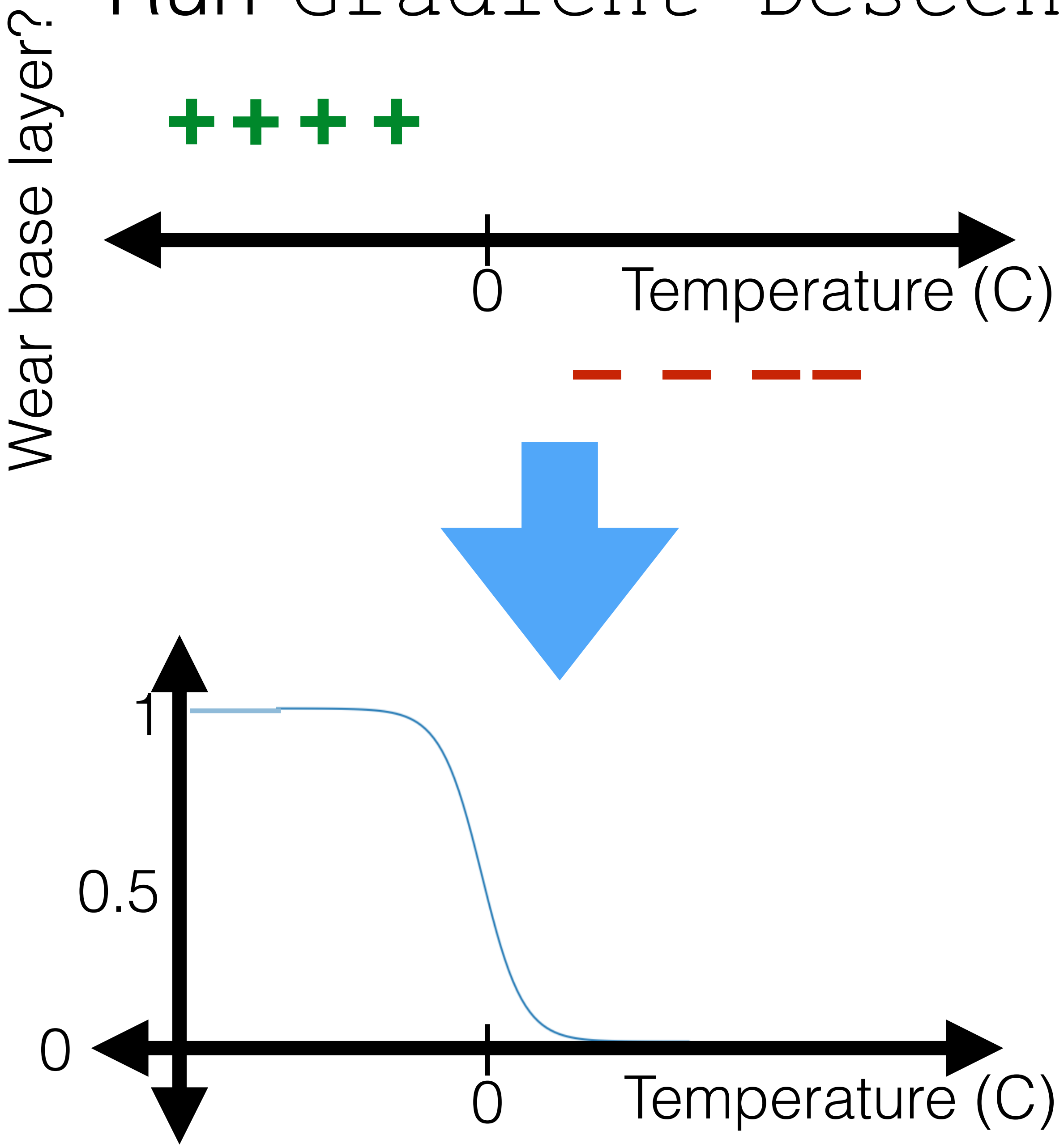
Gradient descent for logistic regression

- Loss $J_{\mathrm{lr}}(\Theta) = J_{\mathrm{lr}}(\theta, \theta_0)$ is differentiable & convex
- Run `Gradient-Descent(` $\Theta_{\mathrm{init}}, \eta, J_{lr}, \nabla_\Theta J_{lr}, \epsilon$ `)`



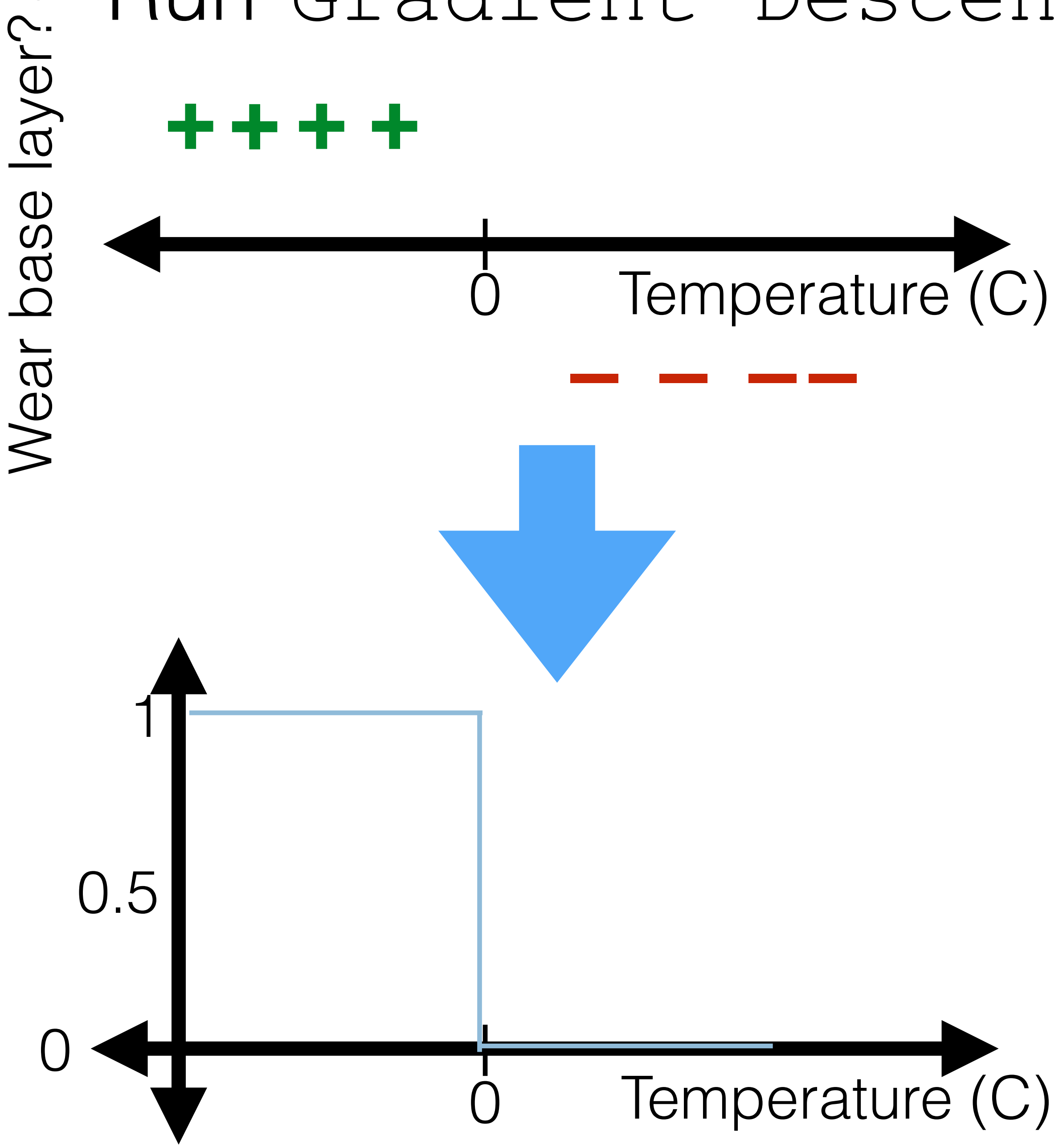
Gradient descent for logistic regression

- Loss $J_{\text{lr}}(\Theta) = J_{\text{lr}}(\theta, \theta_0)$ is differentiable & convex
- Run `Gradient-Descent(` $\Theta_{\text{init}}, \eta, J_{lr}, \nabla_\Theta J_{lr}, \epsilon$ `)`



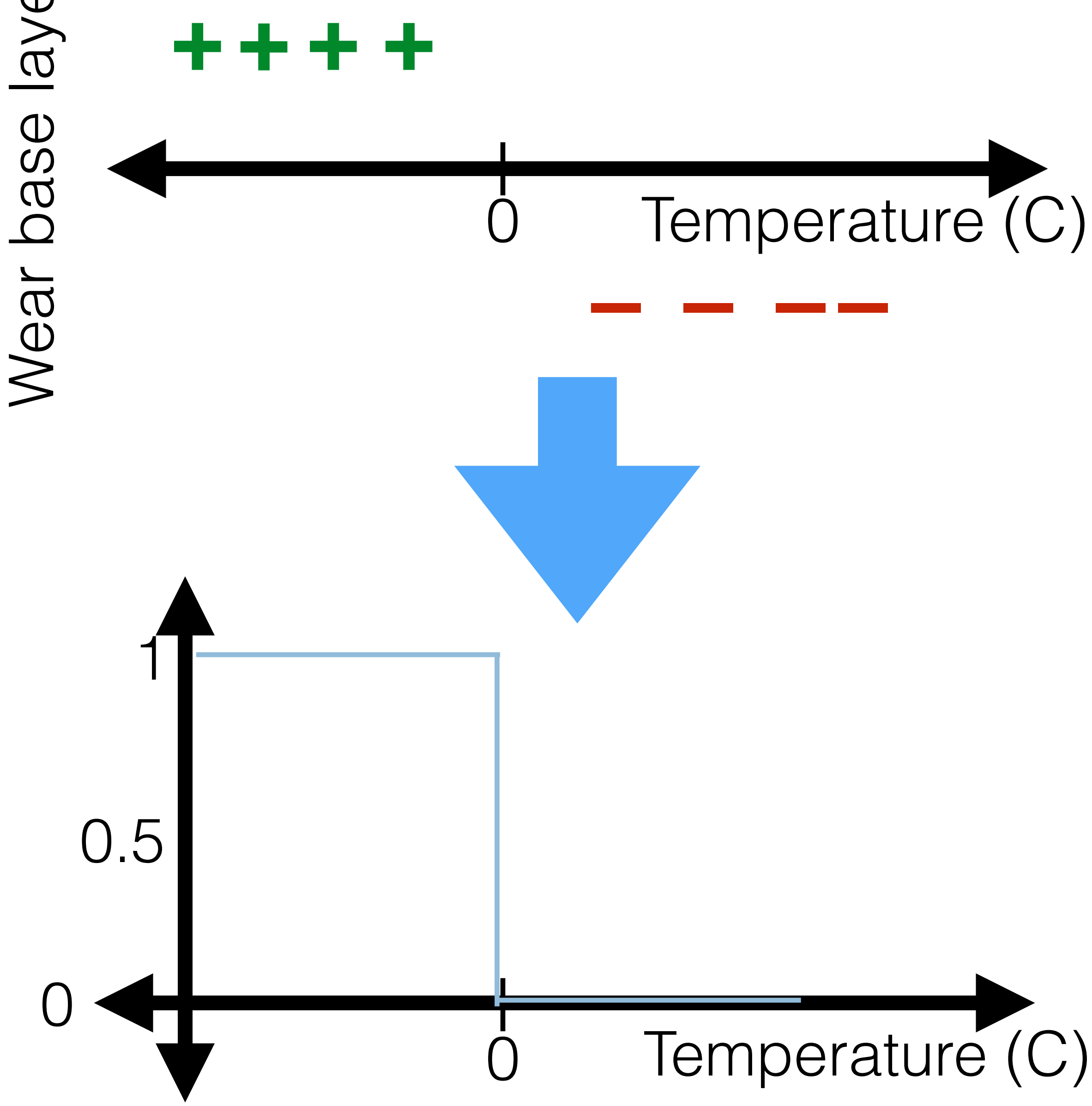
Gradient descent for logistic regression

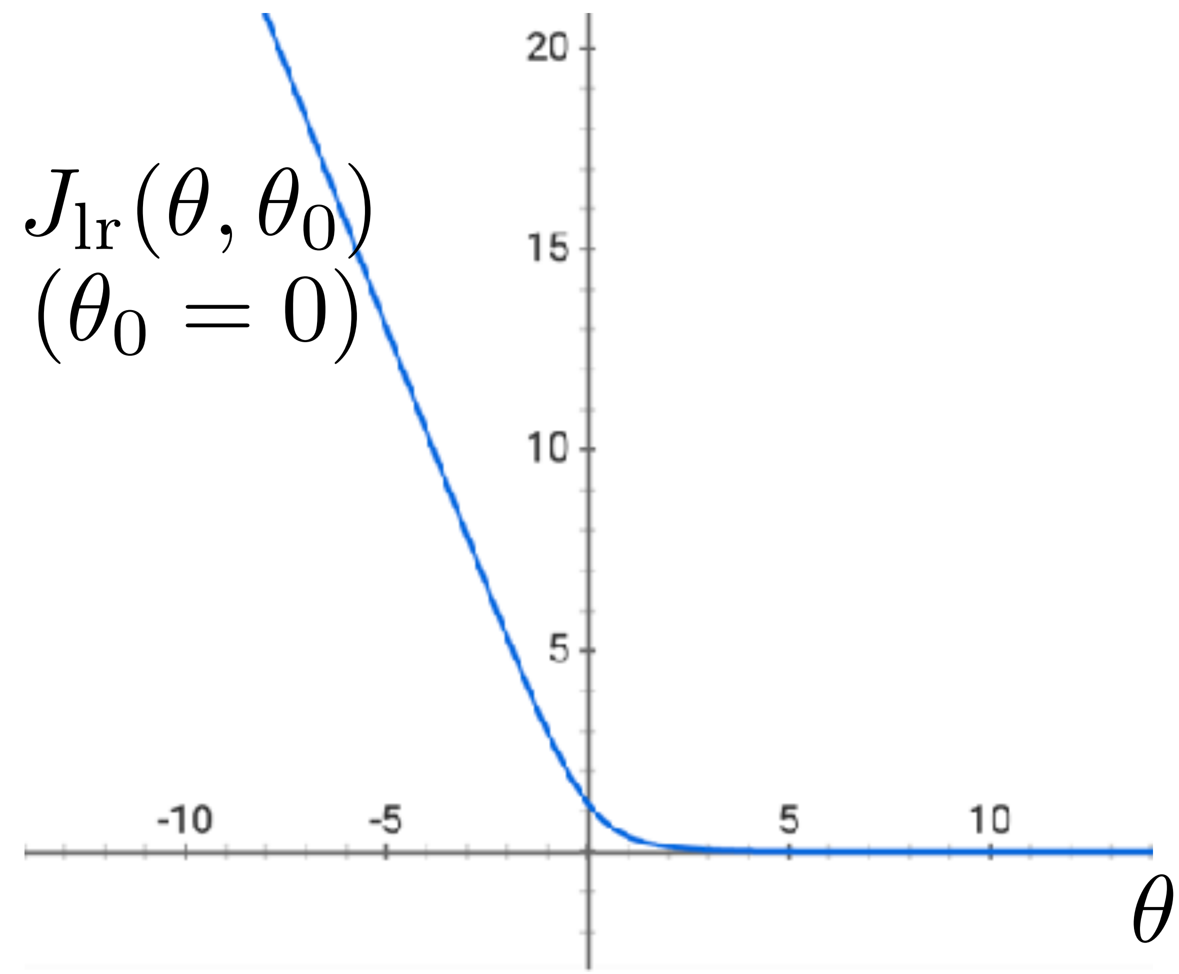
- Loss $J_{\text{lr}}(\Theta) = J_{\text{lr}}(\theta, \theta_0)$ is differentiable & convex
- Run `Gradient-Descent(` $\Theta_{\text{init}}, \eta, J_{lr}, \nabla_\Theta J_{lr}, \epsilon$ `)`



Gradient descent for logistic regression

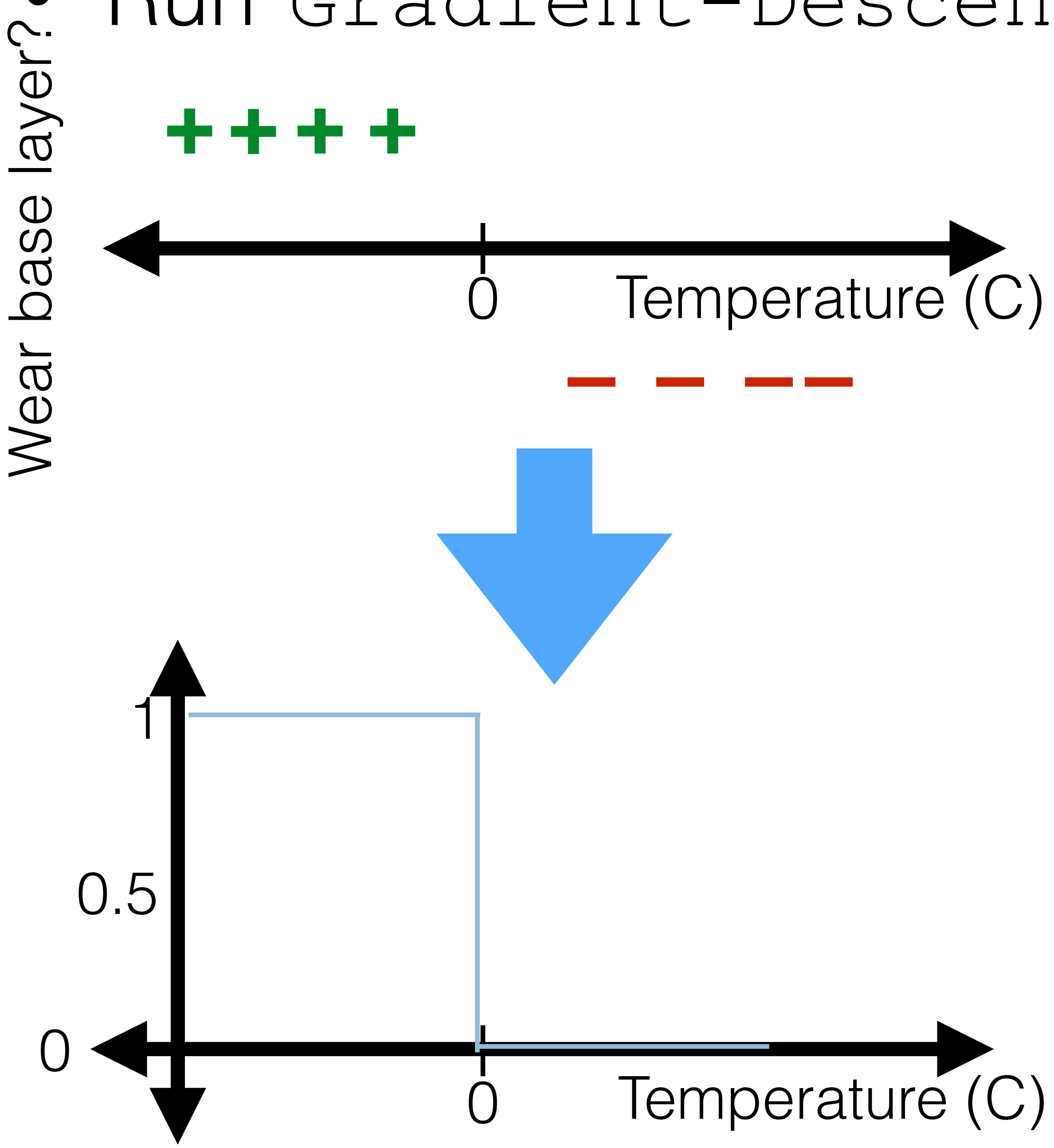
- Loss $J_{\text{lr}}(\Theta) = J_{\text{lr}}(\theta, \theta_0)$ is differentiable & convex
- Run `Gradient-Descent(` $\Theta_{\text{init}}, \eta, J_{lr}, \nabla_\Theta J_{lr}, \epsilon$ `)`

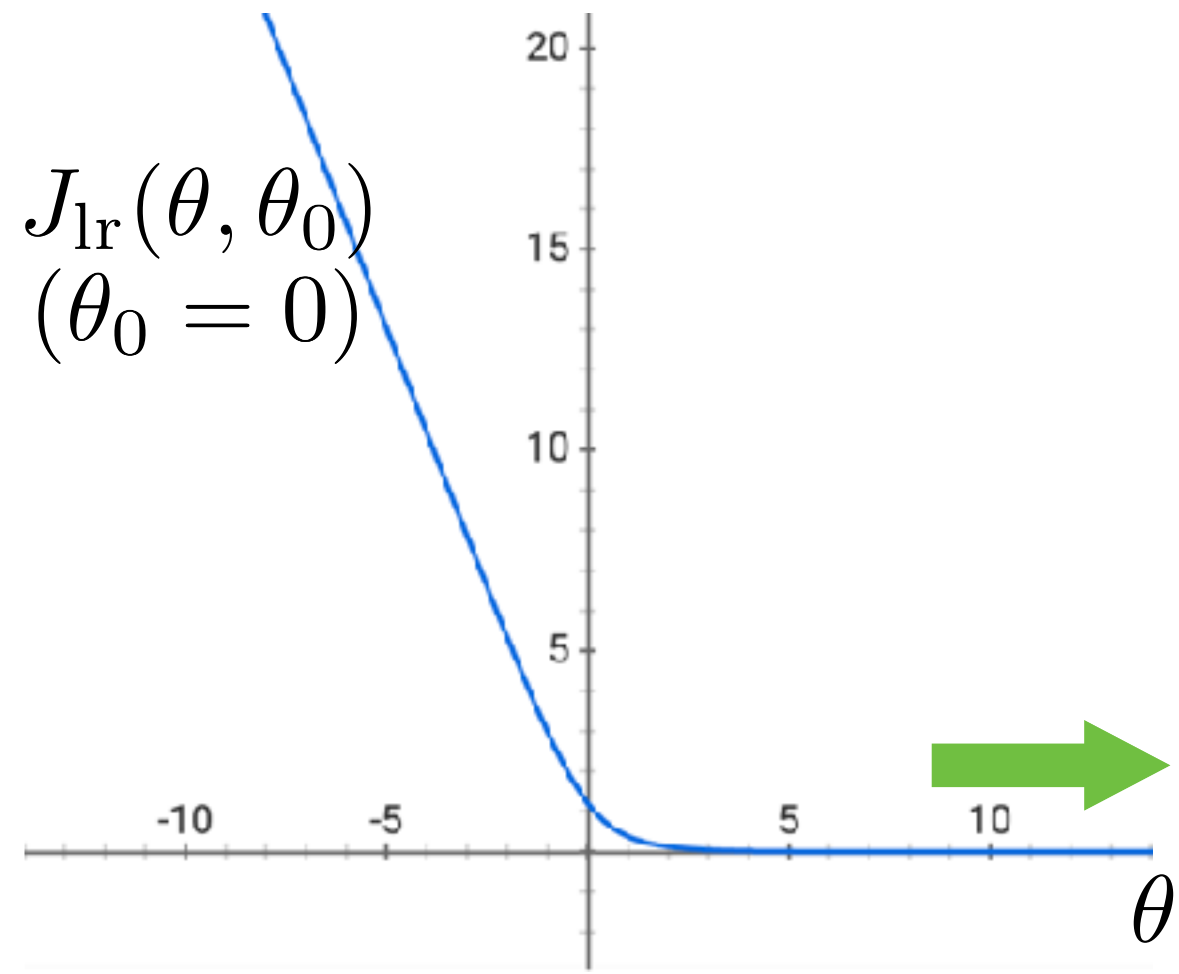




Gradient descent for logistic regression

- 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

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

Logistic regression loss revisited

$$
\begin{aligned}
J_{\mathrm{lr}}(\Theta) &= J_{\mathrm{lr}}(\theta, \theta_0) \\
&= \frac{1}{n} \sum_{i=1}^{n} L_{\mathrm{nll}}(\sigma(\theta^\top x^{(i)} + \theta_0), y^{(i)}) + \lambda \|\theta\|^2 \qquad (\lambda \geq 0)
\end{aligned}
$$

Logistic regression loss revisited

$$\begin{aligned} J_{\text{lr}}(\Theta) &= J_{\text{lr}}(\theta, \theta_0) \\ &= \frac{1}{n}\sum_{i=1}^{n} L_{\text{nll}}(\sigma(\theta^\top x^{(i)} + \theta_0), y^{(i)}) + \lambda\|\theta\|^2 \qquad (\lambda \geq 0) \end{aligned}$$

- A “regularizer” or “penalty” $R(\theta) = \lambda\|\theta\|^2$

Logistic regression loss revisited

$$J_{\text{lr}}(\Theta) = J_{\text{lr}}(\theta, \theta_0)$$
$$= \frac{1}{n}\sum_{i=1}^{n} L_{\text{nll}}(\sigma(\theta^\top x^{(i)} + \theta_0), y^{(i)}) + \lambda\|\theta\|^2 \qquad (\lambda \geq 0)$$

- A "regularizer" or "penalty" $R(\theta) = \lambda\|\theta\|^2$
- Penalizes being overly certain

Logistic regression loss revisited

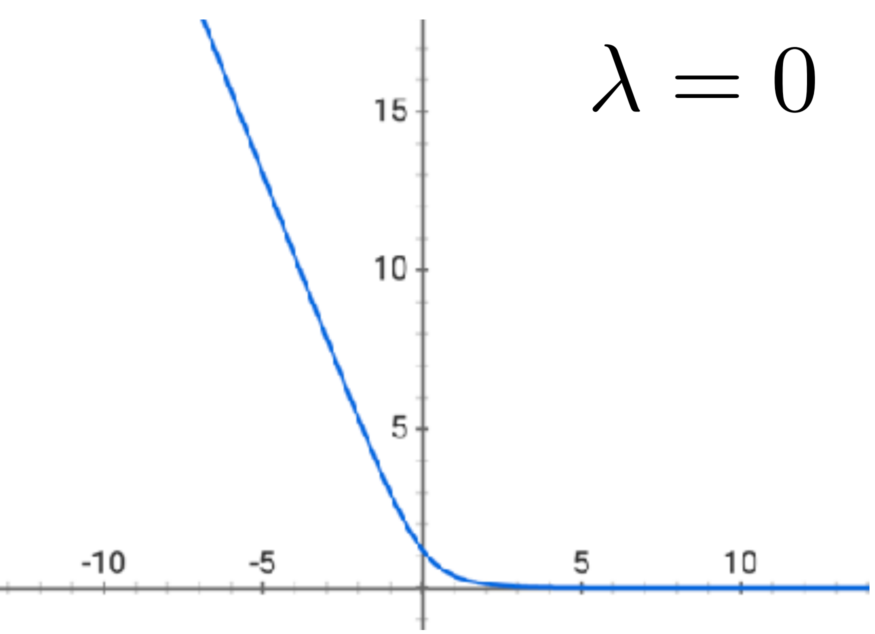
$$
\begin{aligned}
J_{\mathrm{lr}}(\Theta) &= J_{\mathrm{lr}}(\theta, \theta_0) \\
&= \frac{1}{n} \sum_{i=1}^{n} L_{\mathrm{nll}}(\sigma(\theta^\top x^{(i)} + \theta_0), y^{(i)}) + \lambda \|\theta\|^2 \qquad (\lambda \geq 0)
\end{aligned}
$$

- A "regularizer" or "penalty" $R(\theta) = \lambda \|\theta\|^2$
- Penalizes being overly certain
- Objective is still differentiable & convex (gradient descent)

Logistic regression loss revisited

$$J_{\text{lr}}(\Theta) = J_{\text{lr}}(\theta, \theta_0)$$
$$= \frac{1}{n} \sum_{i=1}^{n} L_{\text{nll}}(\sigma(\theta^\top x^{(i)} + \theta_0), y^{(i)}) + \lambda \|\theta\|^2 \qquad (\lambda \geq 0)$$

- A "regularizer" or "penalty" $R(\theta) = \lambda \|\theta\|^2$
- Penalizes being overly certain
- Objective is still differentiable & convex (gradient descent)

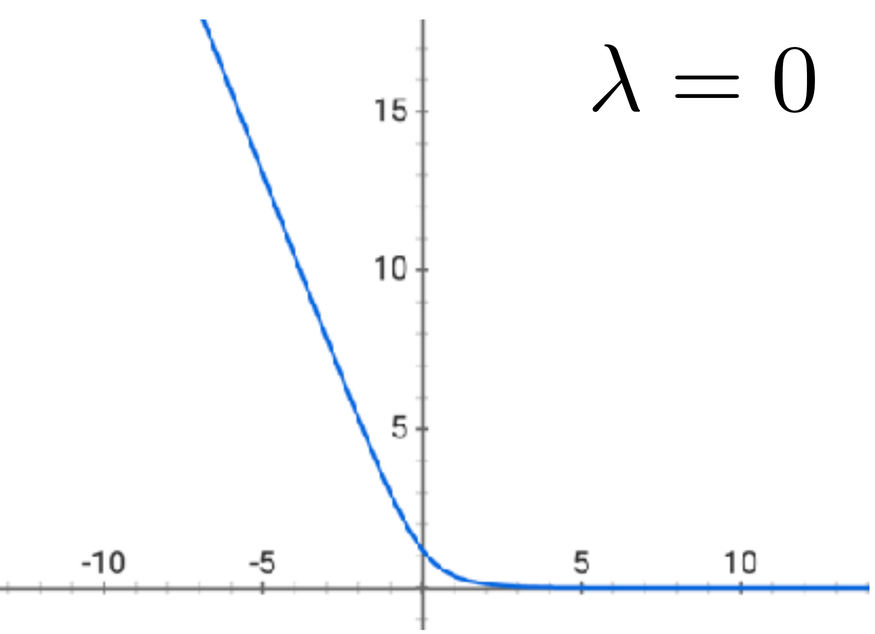


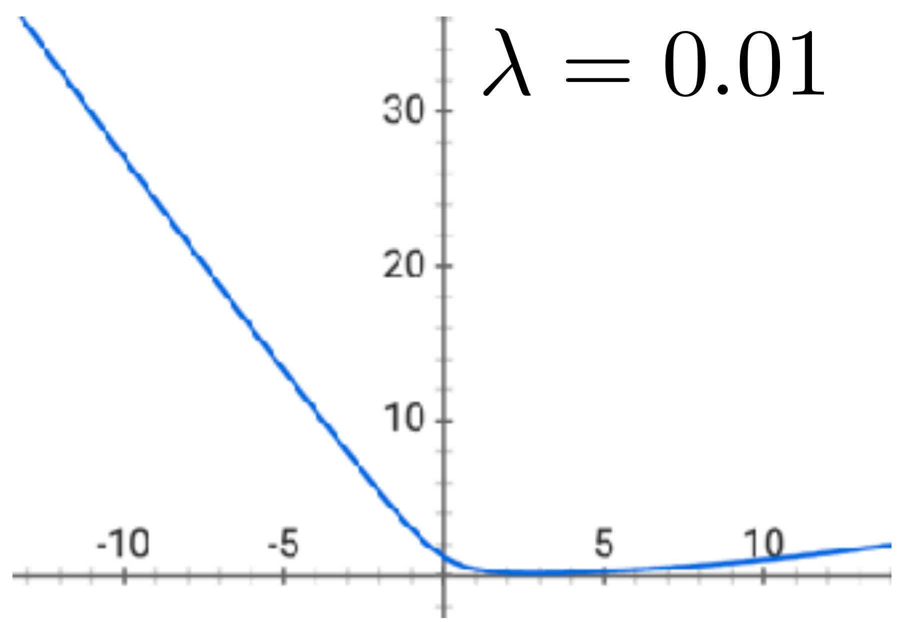
$\lambda = 0$

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” $R(\theta) = \lambda\|\theta\|^2$
- Penalizes being overly certain
- Objective is still differentiable & convex (gradient descent)

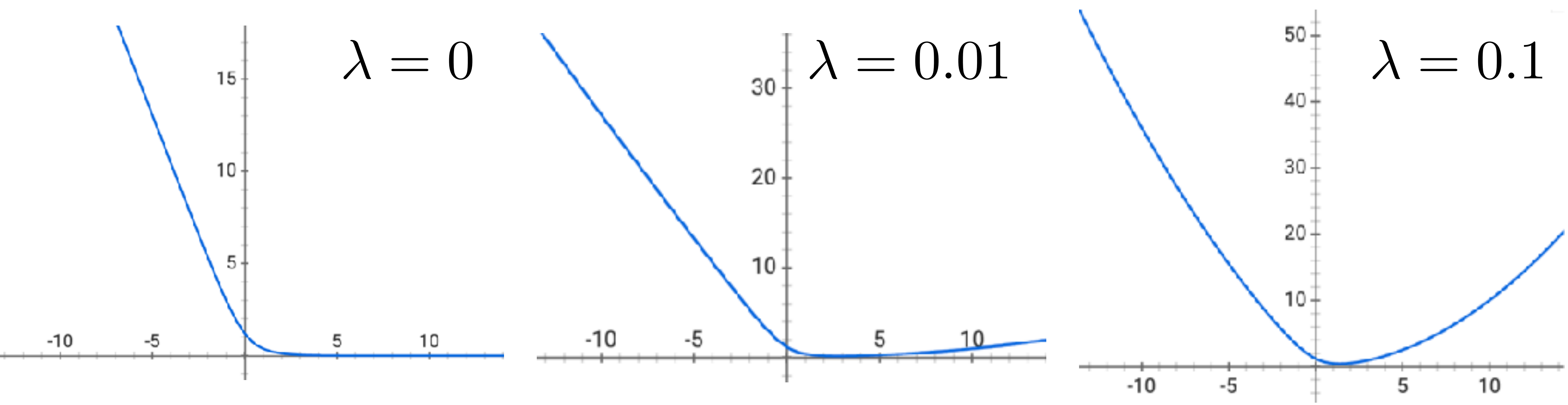




Logistic regression loss revisited

$$J_{\mathrm{lr}}(\Theta) = J_{\mathrm{lr}}(\theta, \theta_0)$$
$$= \frac{1}{n}\sum_{i=1}^{n} L_{\mathrm{nll}}(\sigma(\theta^\top x^{(i)} + \theta_0), y^{(i)}) + \lambda\|\theta\|^2 \qquad (\lambda \geq 0)$$

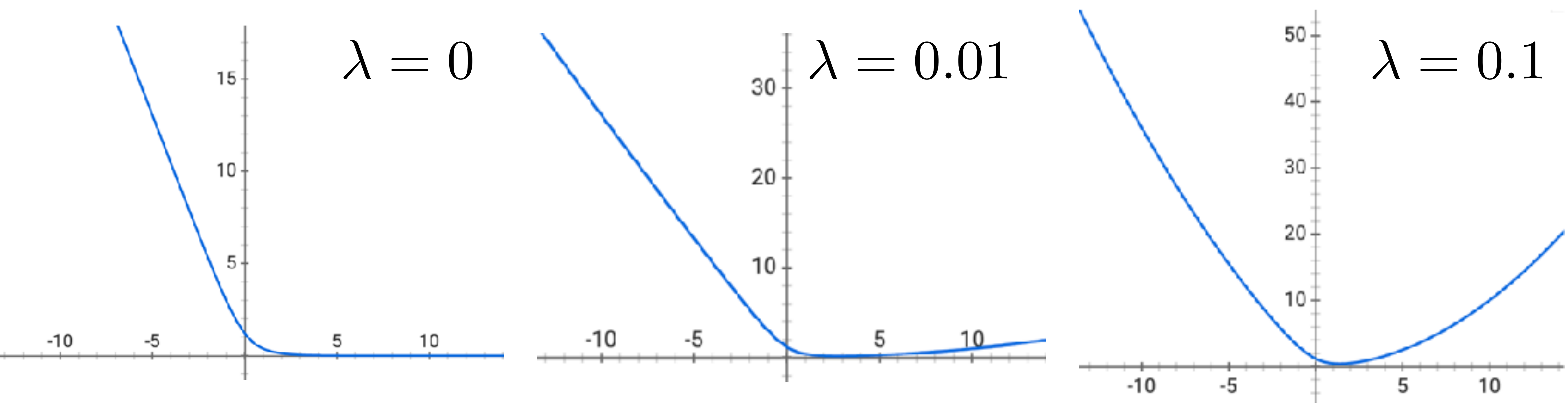
- A “regularizer” or “penalty” $R(\theta) = \lambda\|\theta\|^2$
- Penalizes being overly certain
- Objective is still differentiable & convex (gradient descent)



Logistic regression loss revisited

$$J_{\text{lr}}(\Theta) = J_{\text{lr}}(\theta, \theta_0)$$
$$= \frac{1}{n}\sum_{i=1}^{n} L_{\text{nll}}(\sigma(\theta^\top x^{(i)} + \theta_0), y^{(i)}) + \lambda\|\theta\|^2 \qquad (\lambda \geq 0)$$

- 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(θ_init, θ_0,init, η, ε)
  Initialize θ^(0) = θ_init
  Initialize θ_0^(0) = θ_0,init
  Initialize t = 0
```

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( θ_init, θ_0,init, η, ε )
  Initialize θ^(0) = θ_init
  Initialize θ_0^(0) = θ_0,init
  Initialize t = 0
  repeat
```

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(θ_init, θ_0,init, η, ε)
  Initialize θ^(0) = θ_init
  Initialize θ_0^(0) = θ_0,init
  Initialize t = 0
  repeat
    t = t + 1
```

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(θ_init, θ_0,init, η, ε)
  Initialize θ^(0) = θ_init
  Initialize θ_0^(0) = θ_0,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( θ_init, θ_0,init, η, ε )
  Initialize θ^(0) = θ_init
  Initialize θ_0^(0) = θ_0,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