Bayesian Neural Networks

Practical Deep Learning for Science
20 June, 2024

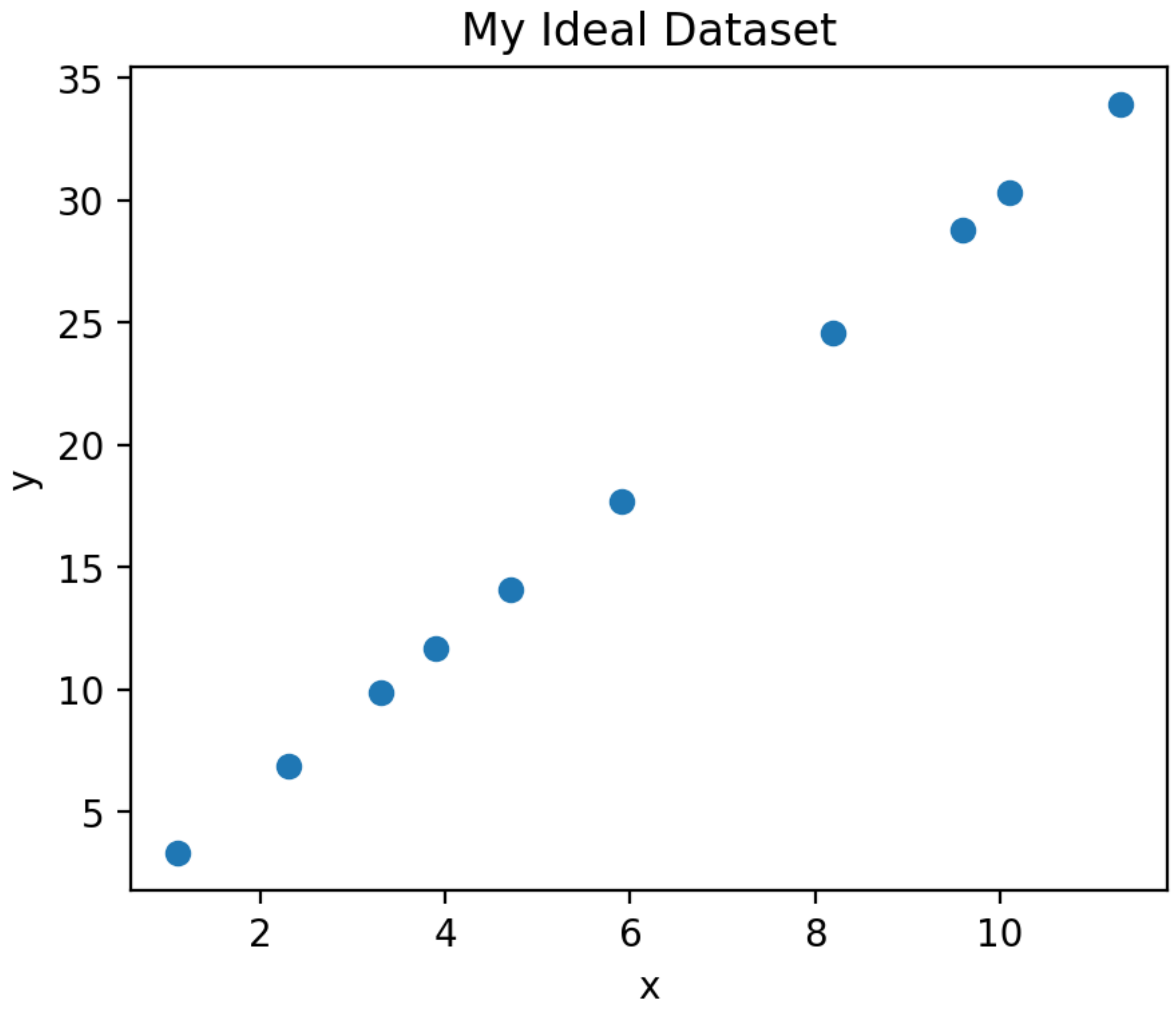
מכון ויצמן למדע
WEIZMANN INSTITUTE OF SCIENCE

- Nilotpal

Let’s say we are doing an experiment

X	Y
1.1	3.3
2.3	6.9
3.3	9.9
3.9	11.7
9.6	28.8
10.1	30.3
11.3	33.9
4.7	14.1
8.2	24.6
5.9	17.7

What is Y at x = 7?

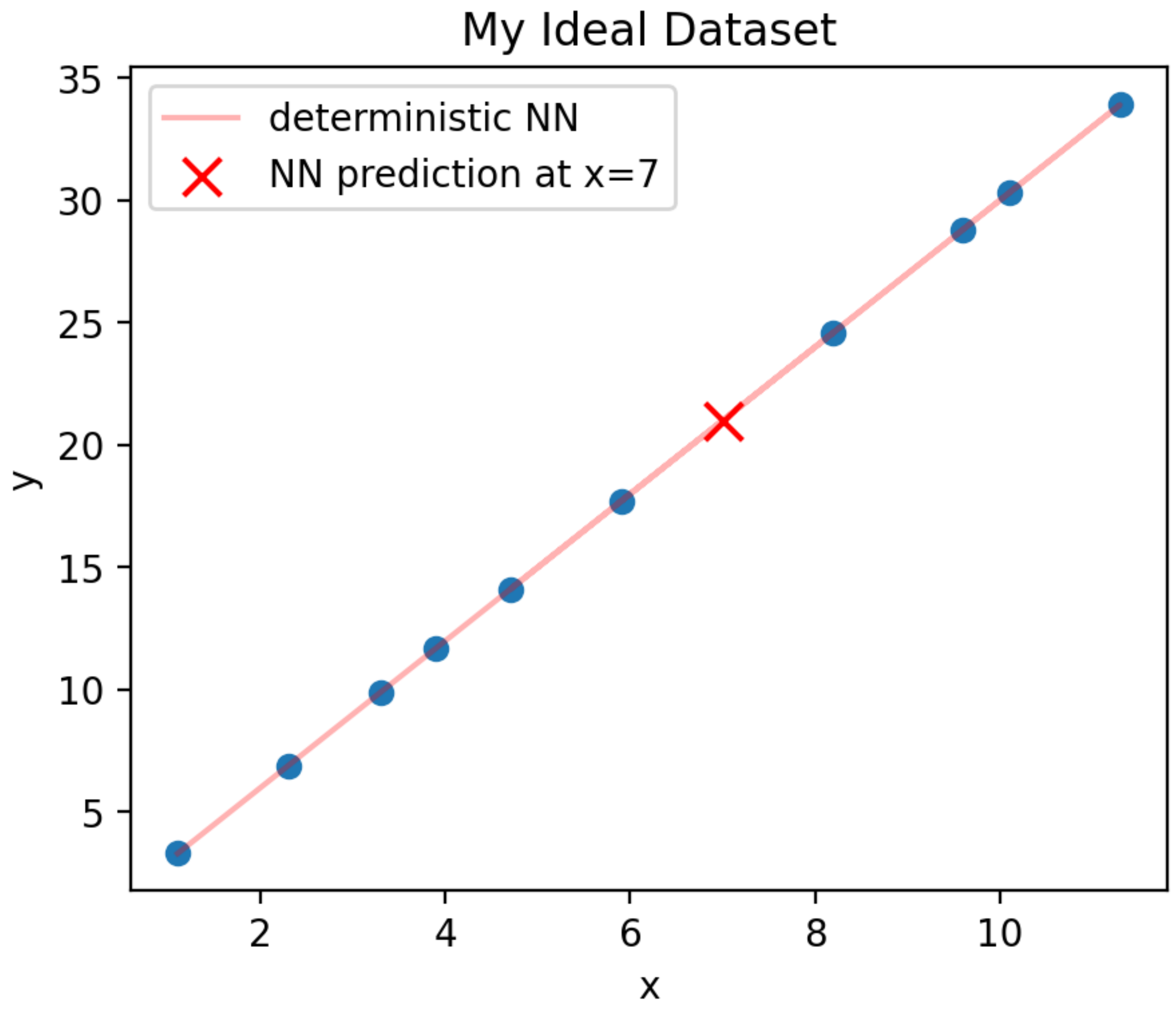


- We can still train an NN with **<u>only one weight</u>** to do the same

Let's say we are doing an experiment

X	Y
1.1	3.3
2.3	6.9
3.3	9.9
3.9	11.7
9.6	28.8
10.1	30.3
11.3	33.9
4.7	14.1
8.2	24.6
5.9	17.7

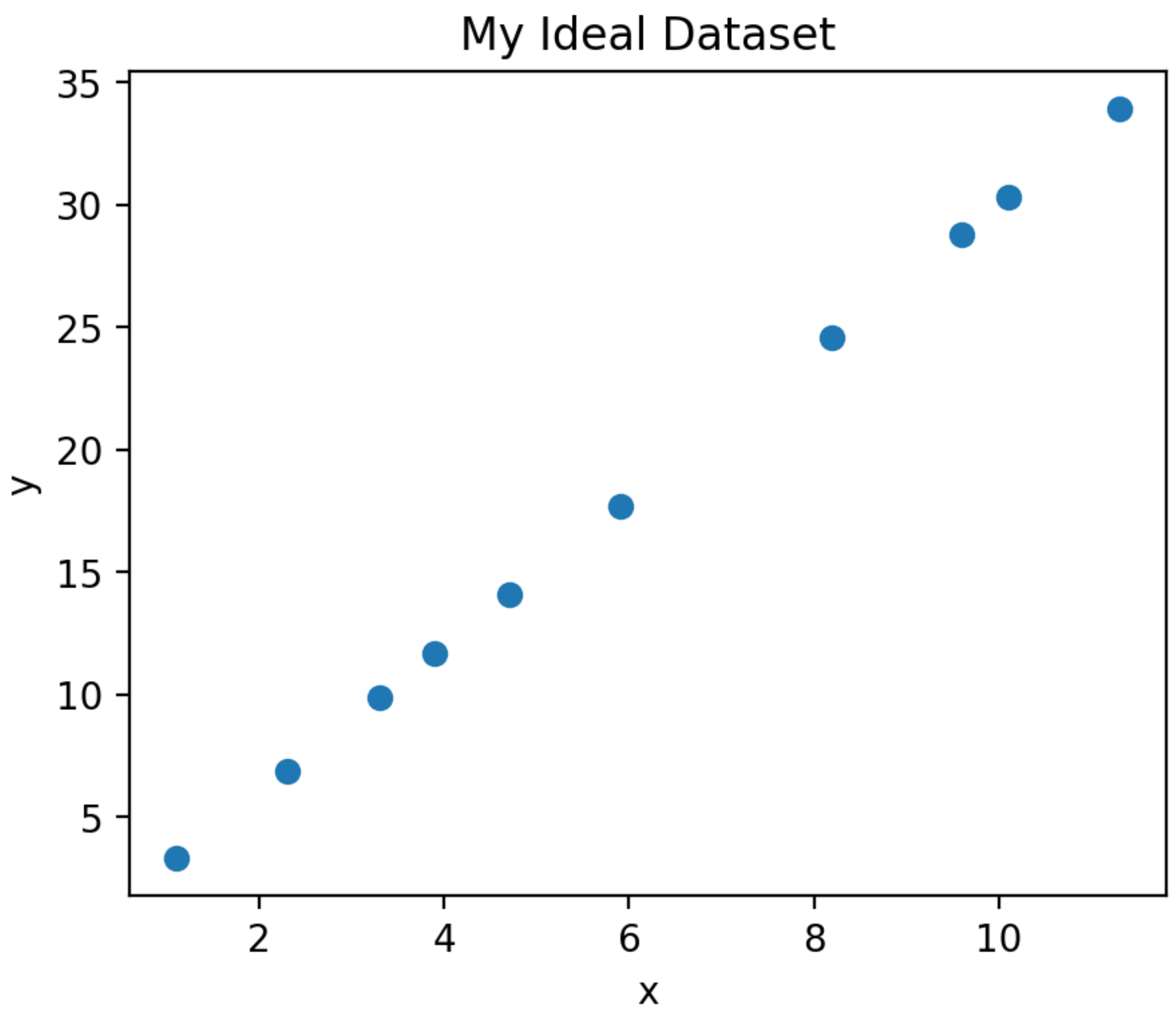
What is Y at x = 7?



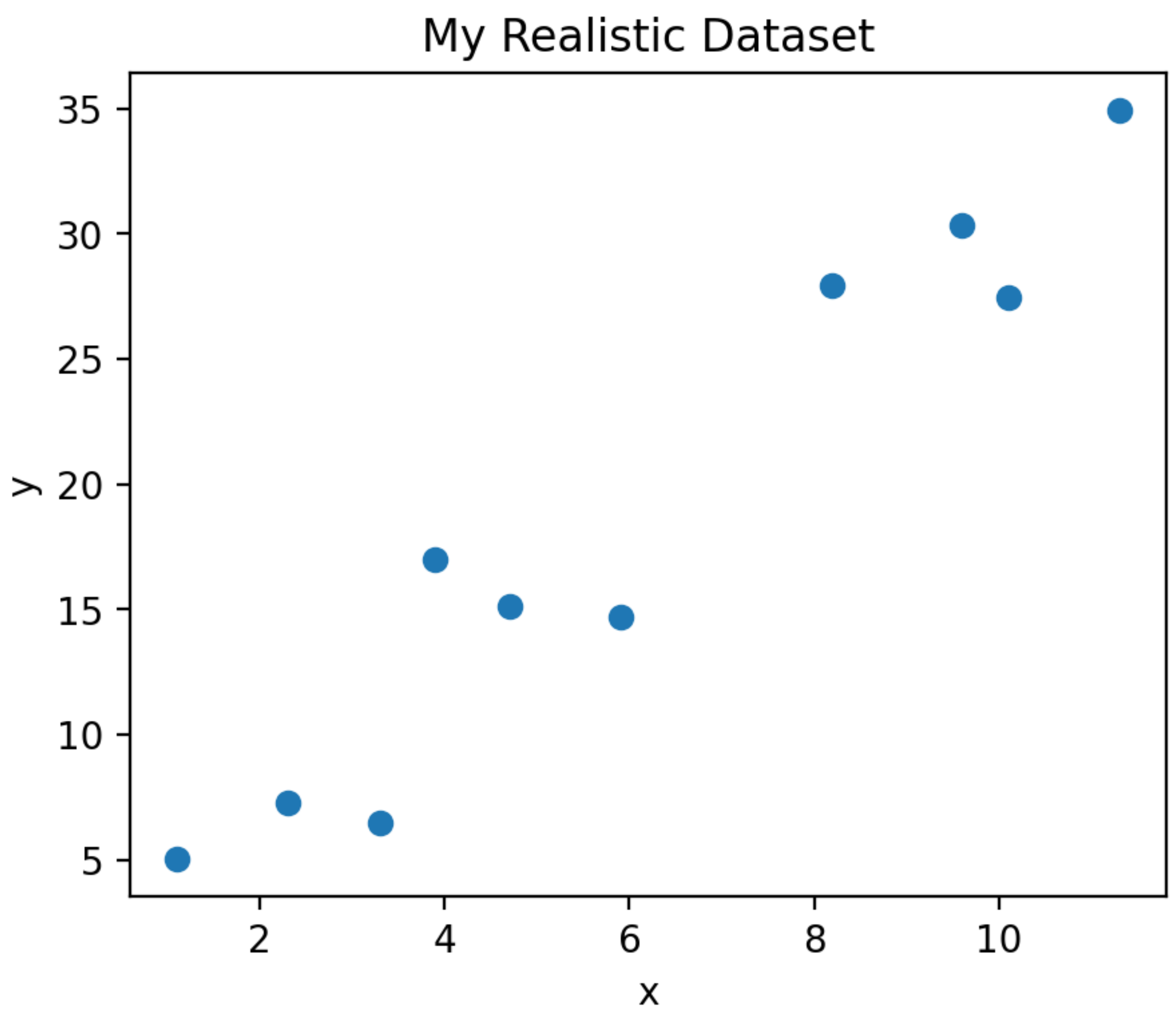
- We can still train an NN with **only one weight** to do the same

Let’s say we are doing an experiment

Ideal situation

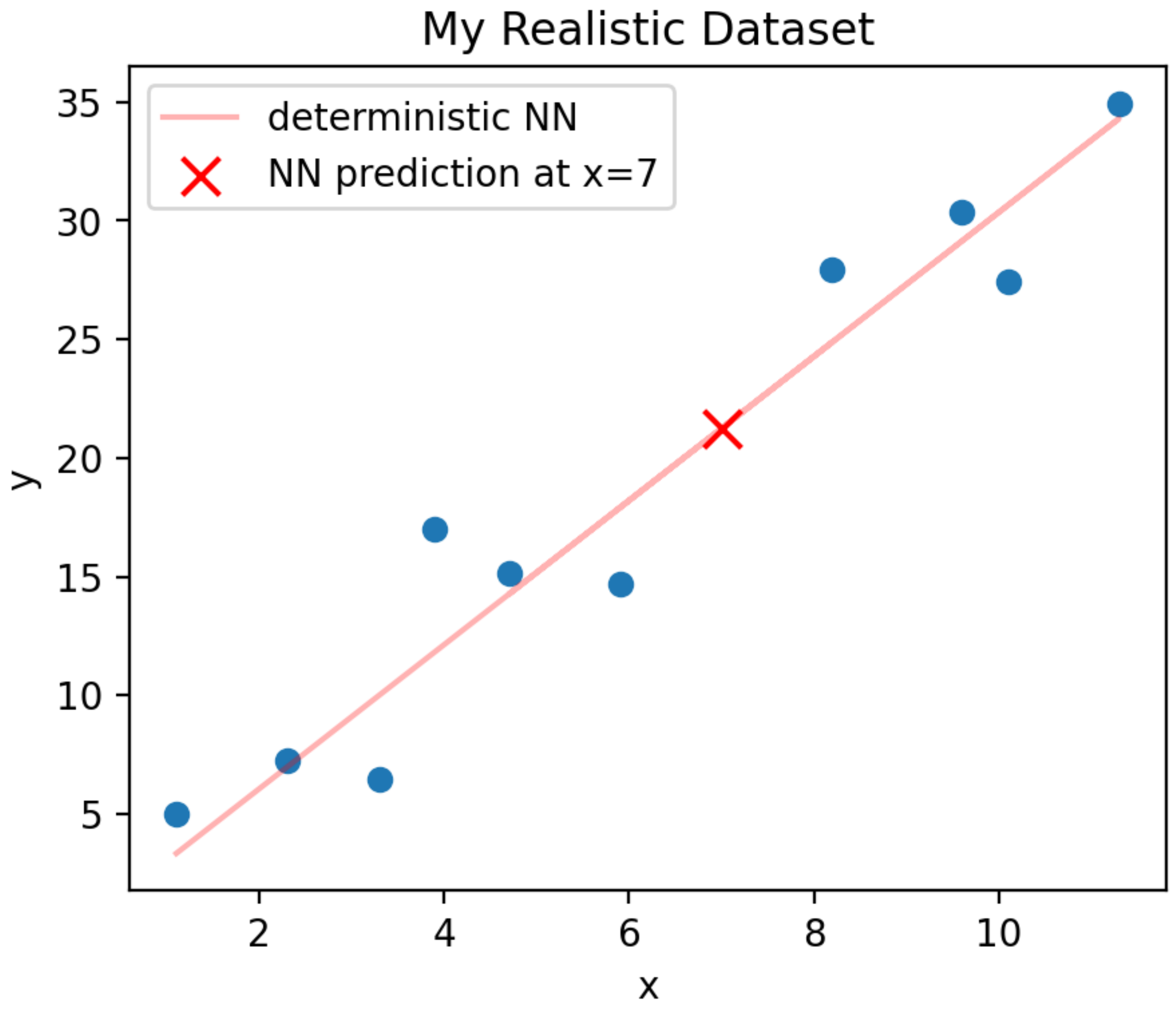


What we see

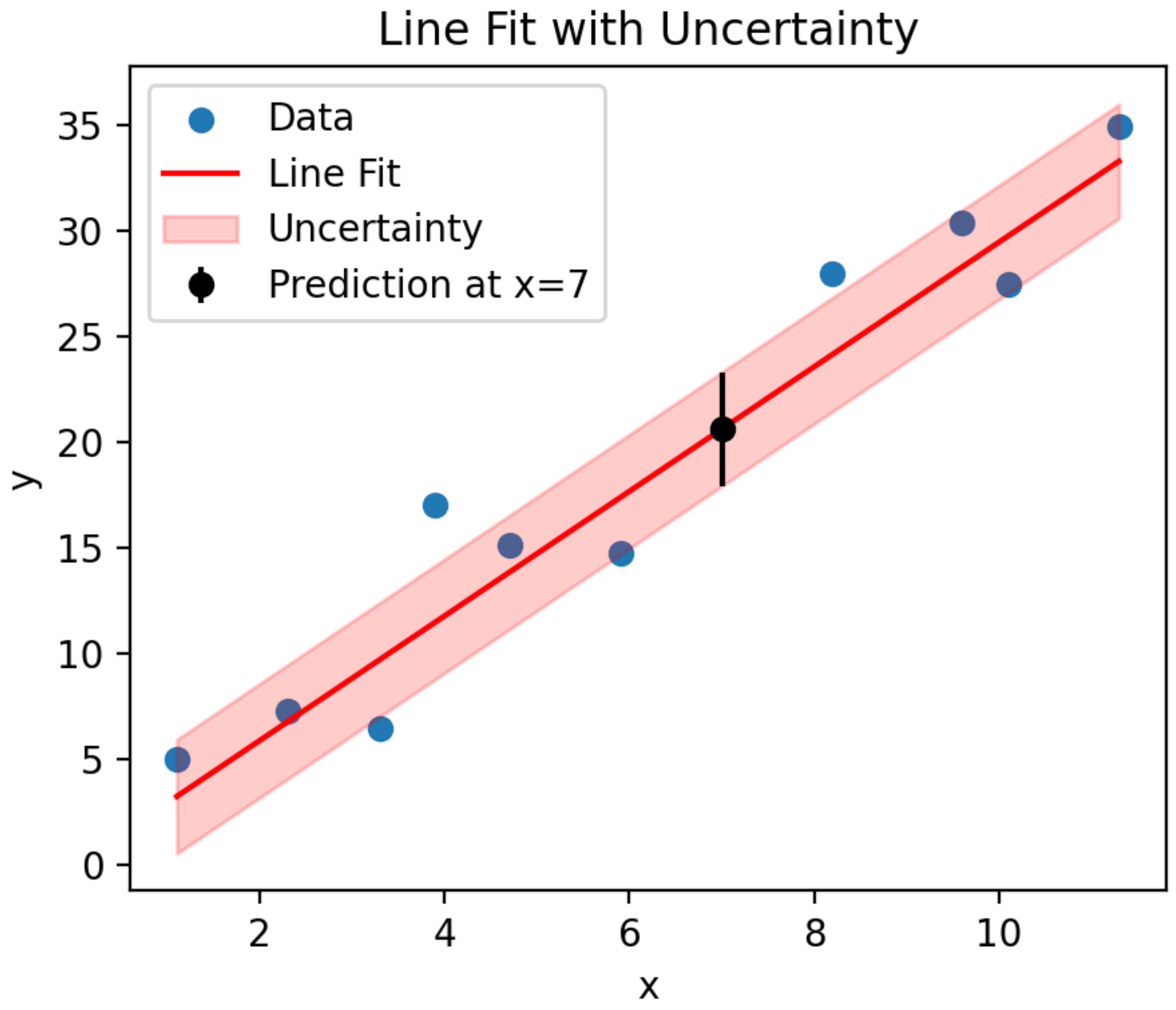


If we train a network

We will get



What we want



How do we quantify the uncertainty with NN

Aleatoric uncertainty

- Noise in the data
- More data won't help

vs

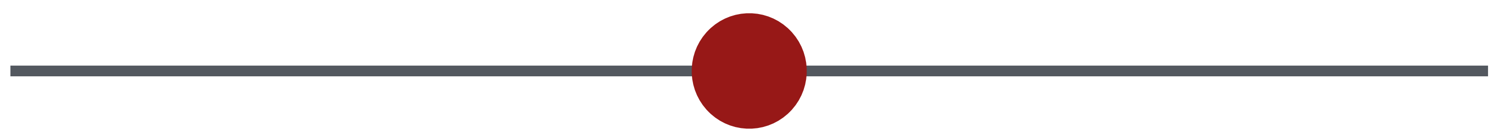
Epistemic uncertainty

- Uncertainty in the model parameters
- More data will help

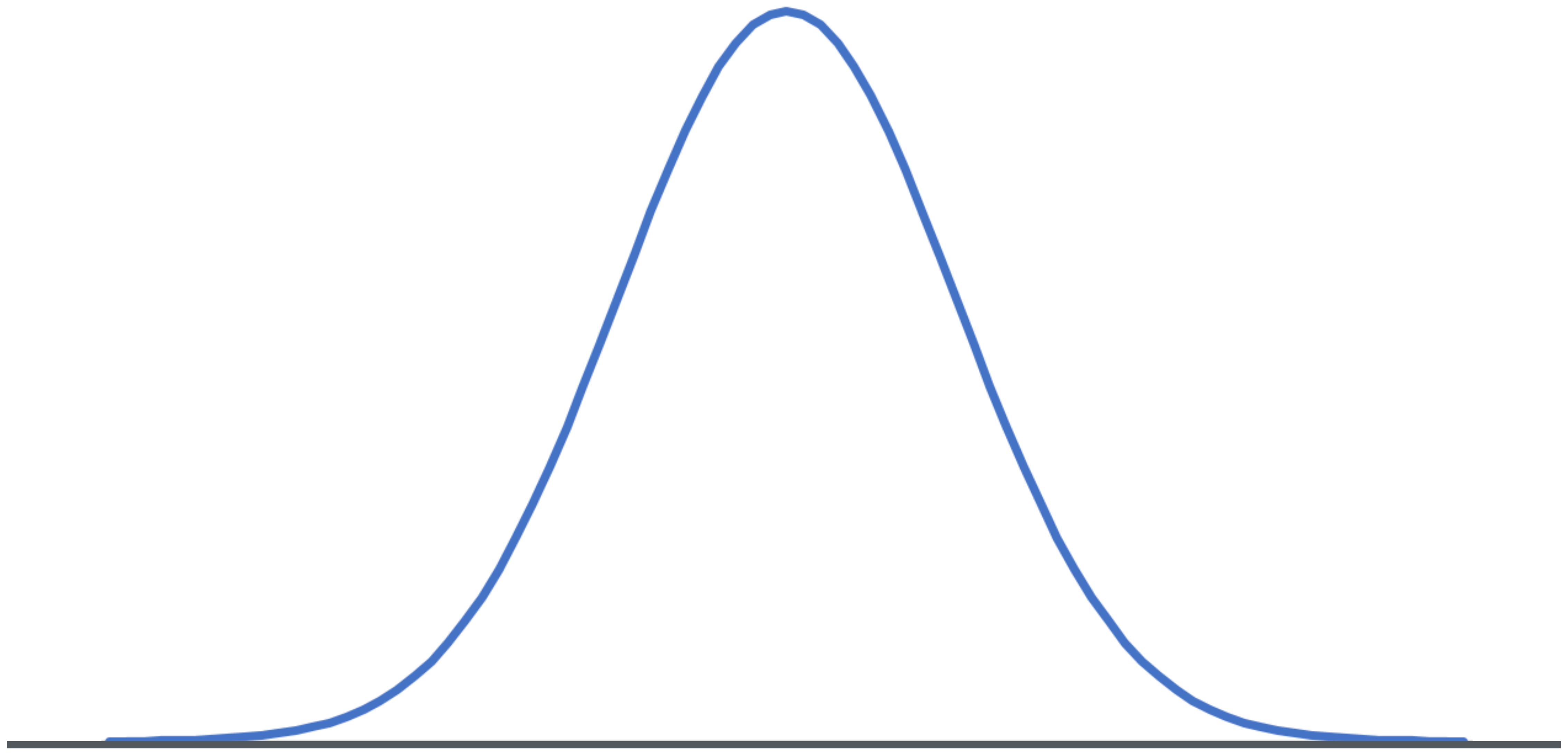
- We need to modify the network to accommodate these two uncertainties

Model uncertainty

Gaussian (μ, σ)



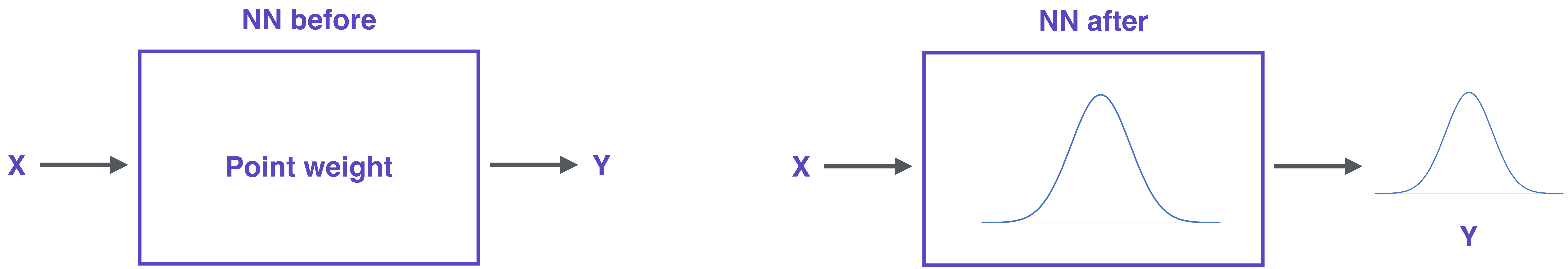
Model weight before



Model weight after

- Each weight, instead of being a number, will be two numbers - μ and σ
- During forward pass, we sample from the Gaussian (μ, σ)

Model uncertainty



How about the uncertainty from the data

- Let's add another Gaussian into the picture
- Mean = output of the NN (a gaussian)
- Sigma = another Gaussian we'll learn

NN output

Noise in data, we'll learn it

$\hat{Y} =$ Gaussian (mu, sigma)

We are drawing a Gaussian around the network prediction,
and we'll learn the width of this Gaussian

‘Bayes’ian NN

Li’l bit of extra maths :)

Bayesian networks

$p(x)$ **= distribution of** x

- We have
  - θ network parameters
  - D data

- We want to get the right network parameters, given the data (through training)
  - $p\left(\theta \mid D\right)$, we don't know how to compute it directly
  - Bayes' theorem gives us

$$p\left(\theta \mid D\right) = \frac{p\left(D \mid \theta\right) p\left(\theta\right)}{p\left(D\right)}$$

$p\left(y \mid x, \theta\right)$ (model output distribution given input and NN)

Prior (initial distribution of the NN weights; mostly Gaussian)

Distribution of data

$p\left(D\right)$ is expensive!

- $$p\left(D\right) = \int p\left(D \,|\, \theta\right) p\left(\theta\right) d\theta$$
  - Need to compute over all possible values of θ
  - Not possible!

$p(x)$ **= distribution of** x

- So, we approximate
  - Variational Inference (VI)

Approximate the NN weight distribution $p\left(\theta | D\right)$

- Let's assume $p\left(\theta | D\right)$ follows some distribution (say, Gaussian)
  - We try to approximate its parameters
  - $p\left(\theta | \phi\right)$ (how my NN weights are distributed, given the parameters)
    - ϕ will be μ and σ for our Gaussian p

- We want to make approximate posterior, $p\left(\theta | \phi\right)$ close to the true posterior, $p\left(\theta | D\right)$
  - So, ideally a KLD b/w the two, but we of course don't know the true posterior

- Turns out, this is equivalent to maximizing ELBO (Evidence Lower bound)

$$ELBO = Exp\left[logp\left(D | \theta\right)\right] - KL\left(p\left(\theta | \phi\right) \,||\, p(\theta)\right)$$

```
svi = SVI(model, guide, adam, loss=Trace_ELBO())
```