Regularization

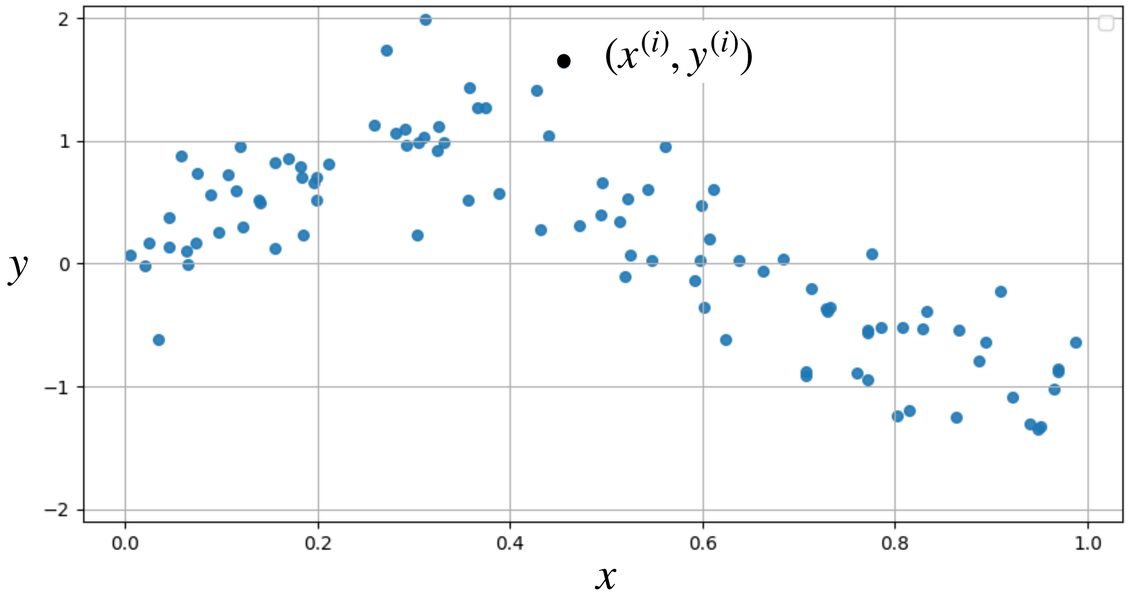
Elena Congeduti, 28-11-2024

Lecture’s Agenda

- Errors
- Overfitting and Model Capacity
- Regularization

Errors

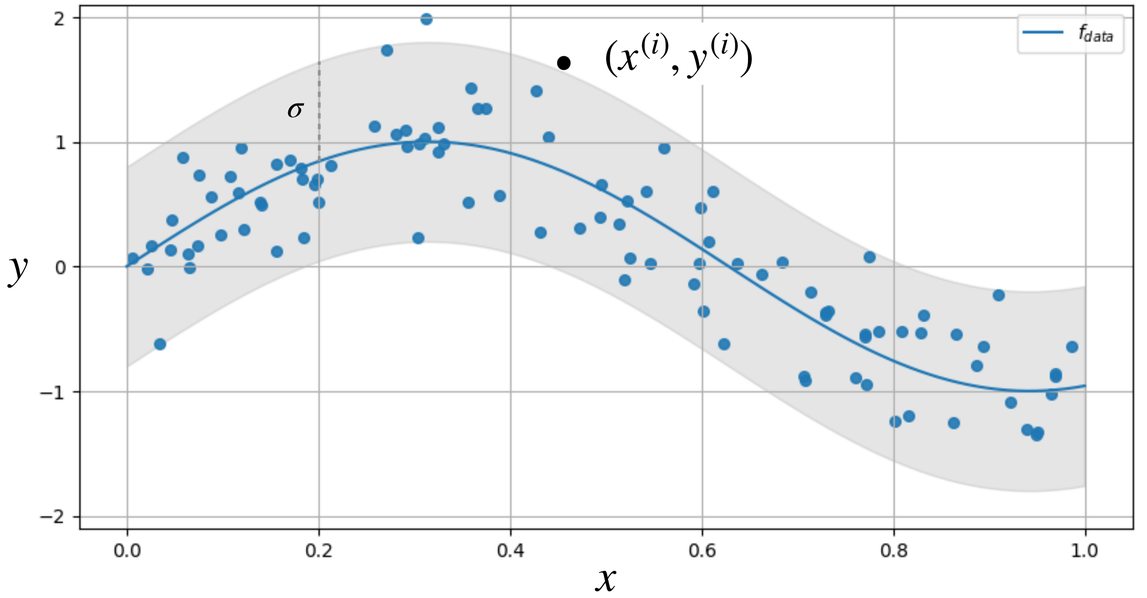
Regression problem



$\left(x^{(i)}, y^{(i)}\right)$ data points generated through f_{data} with some noise σ

Errors

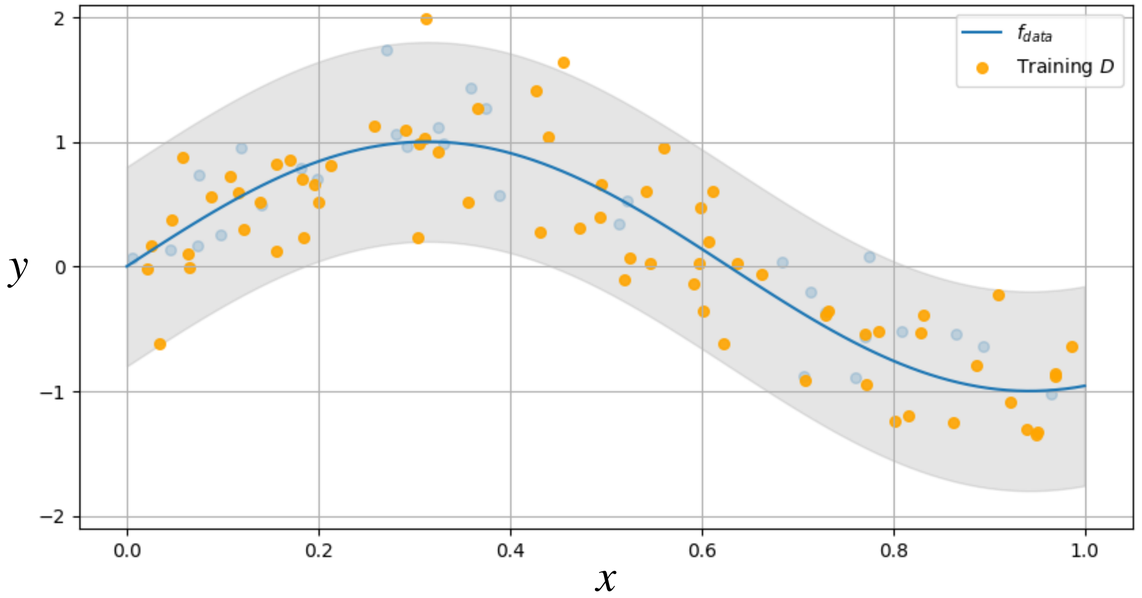
Regression problem



$\left(x^{(i)}, y^{(i)}\right)$ data points generated through f_{data} with some noise σ

Errors

Regression problem



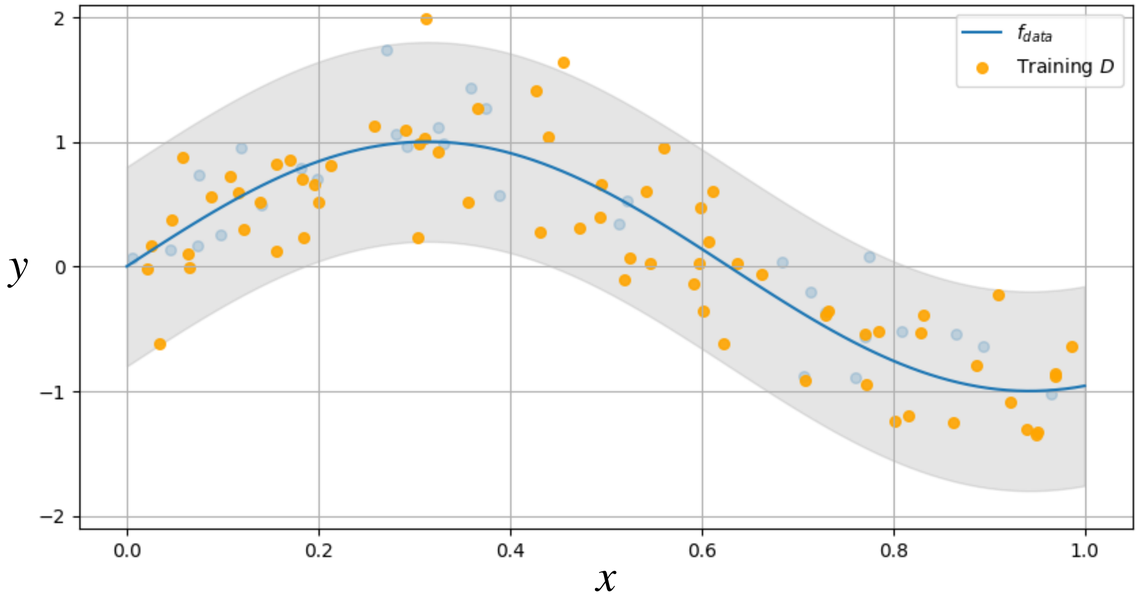
$\left(x^{(i)}, y^{(i)}\right)$ data points generated through f_{data} with some noise σ

Random split:

Training set D ● with size n

Errors

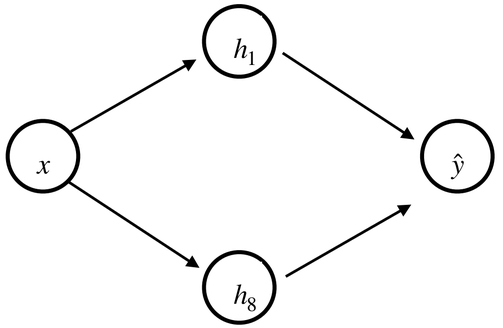
Regression problem



$\left(x^{(i)}, y^{(i)}\right)$ data points generated through f_{data} with some noise σ

Random split:

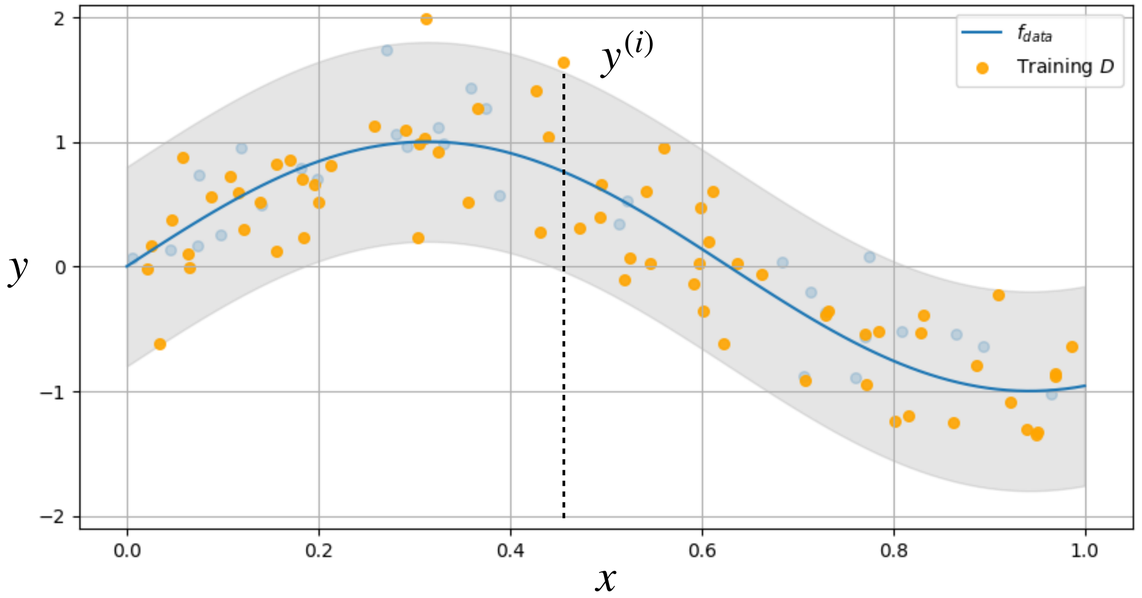
Training set D ● with size n



Fully connected layer, ReLU activation with 8 hidden units, $f_{model}(x;\theta) = \hat{y}$

Errors

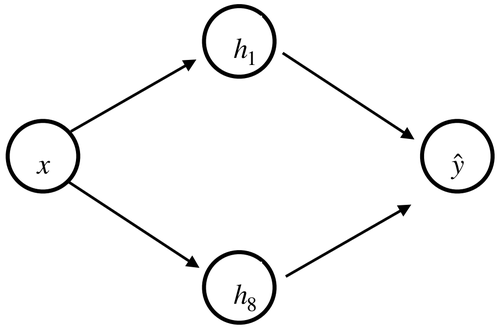
Regression problem



$\left(x^{(i)}, y^{(i)}\right)$ data points generated through f_{data} with some noise σ

Random split:

Training set D ● with size n

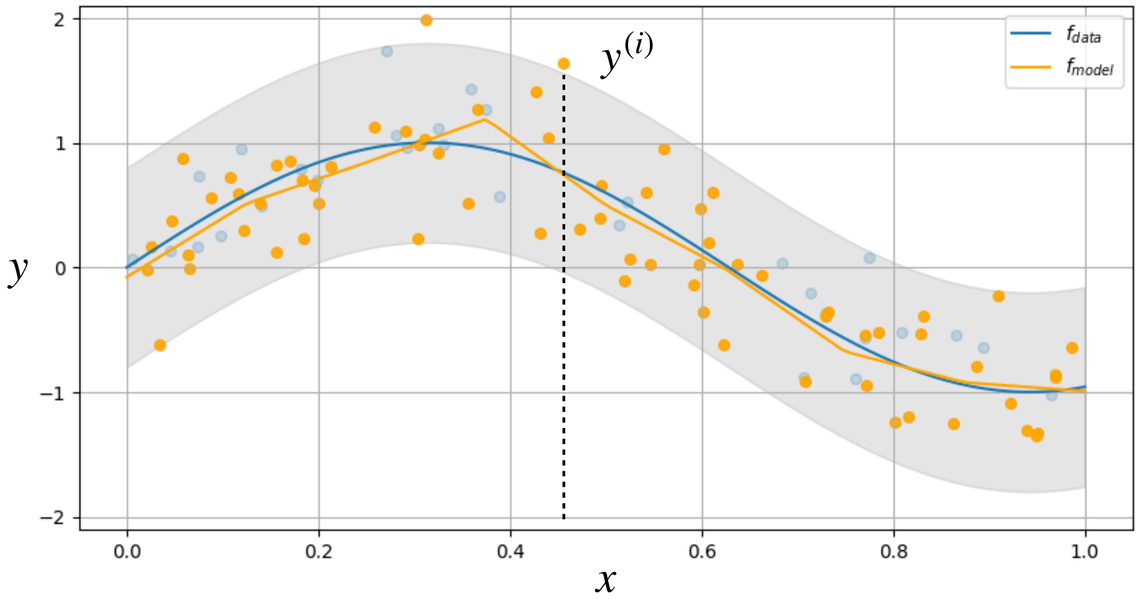


Fully connected layer, ReLU activation with 8 hidden units, $f_{model}(x;\theta) = \hat{y}$

Training loss $L_{SE}(\theta) = \frac{1}{n}\sum_{i=1}^{n}\frac{1}{2}(y^{(i)} - \hat{y}^{(i)})^2$

Errors

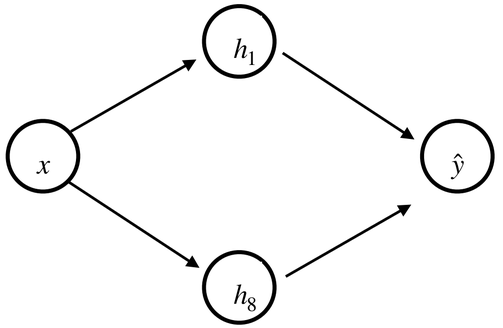
Regression problem



$(x^{(i)}, y^{(i)})$ data points generated through f_{data} with some noise σ

Random split:

Training set D ● with size n



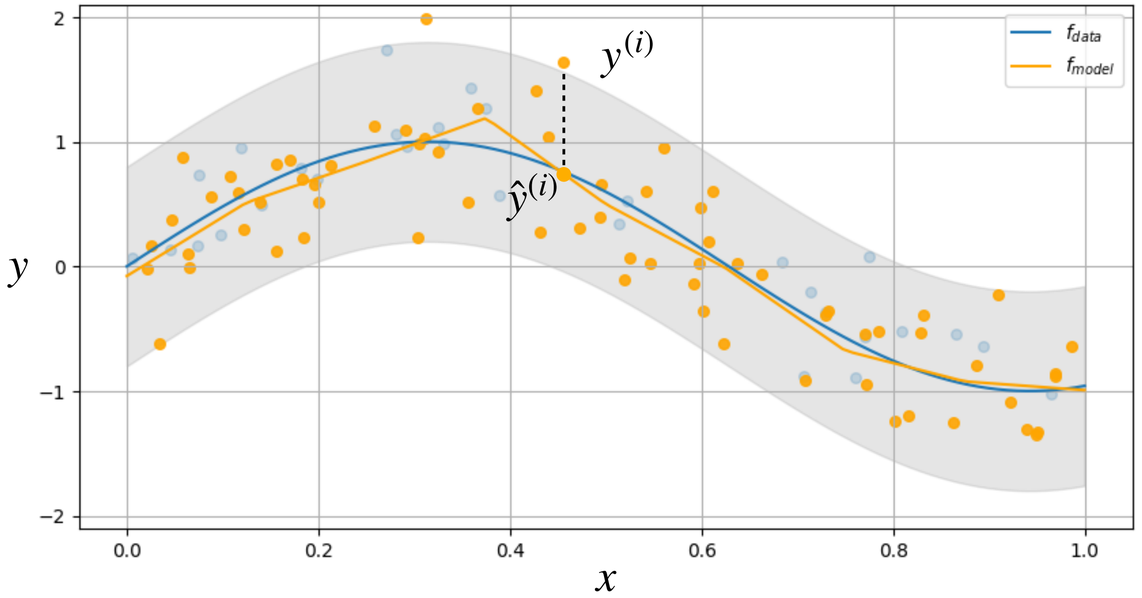
Fully connected layer, ReLU activation with 8 hidden units, $f_{model}(x;\theta) = \hat{y}$

Training loss $L_{SE}(\theta) = \frac{1}{n}\sum_{i=1}^{n}\frac{1}{2}(y^{(i)} - \hat{y}^{(i)})^2$

Optimal parameters $\hat{\theta} = argmin_{\theta} L(\theta)$

Errors

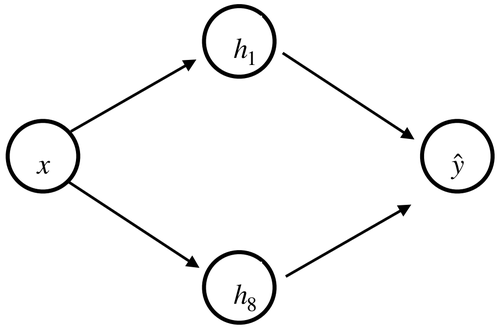
Regression problem



$(x^{(i)}, y^{(i)})$ data points generated through f_{data} with some noise σ

Random split:

Training set D ● with size n



Training error = $\frac{1}{n}\sum_{i=1}^{n}\frac{1}{2}(y^{(i)} - \hat{y}^{(i)})^2$ ● = 0.12

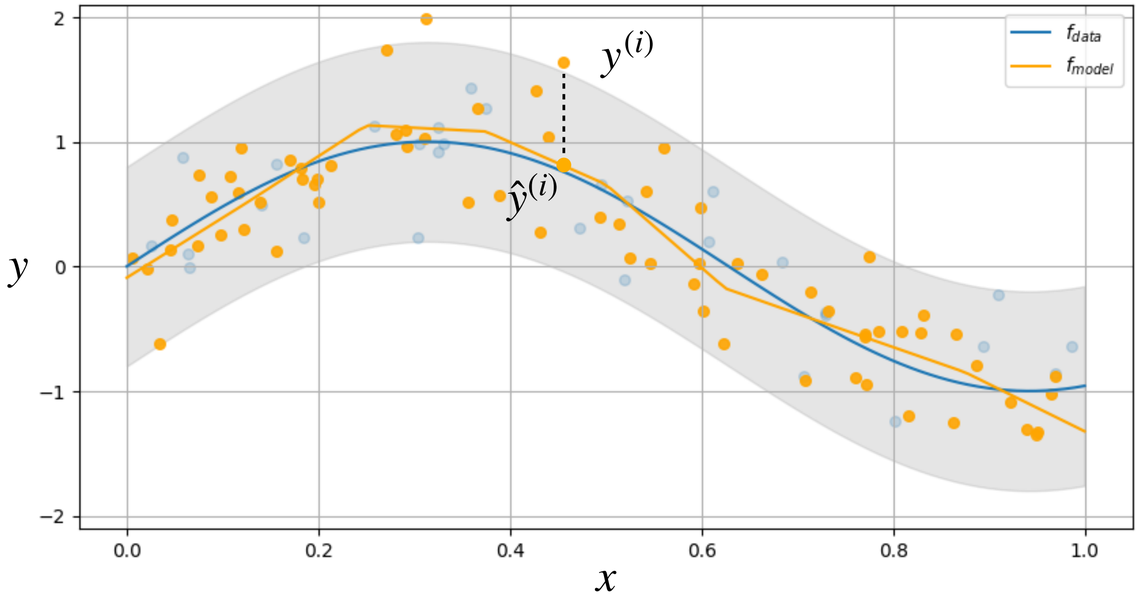
Fully connected layer, ReLU activation with 8 hidden units, $f_{model}(x;\theta) = \hat{y}$

Training loss $L_{SE}(\theta) = \frac{1}{n}\sum_{i=1}^{n}\frac{1}{2}(y^{(i)} - \hat{y}^{(i)})^2$

Optimal parameters $\hat{\theta} = argmin_{\theta} L(\theta)$

Errors

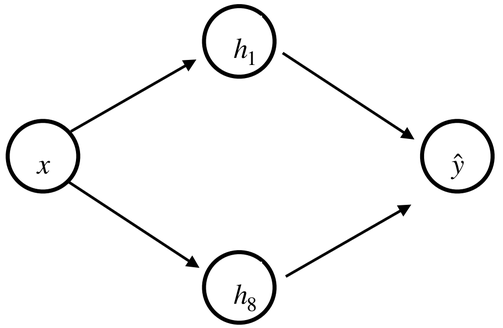
Regression problem



$(x^{(i)}, y^{(i)})$ data points generated through f_{data} with some noise σ

Different random split:

Training set D ● with size n



Training error = $\frac{1}{n}\sum_{i=1}^{n}\frac{1}{2}(y^{(i)} - \hat{y}^{(i)})^2$ ● = 0.1

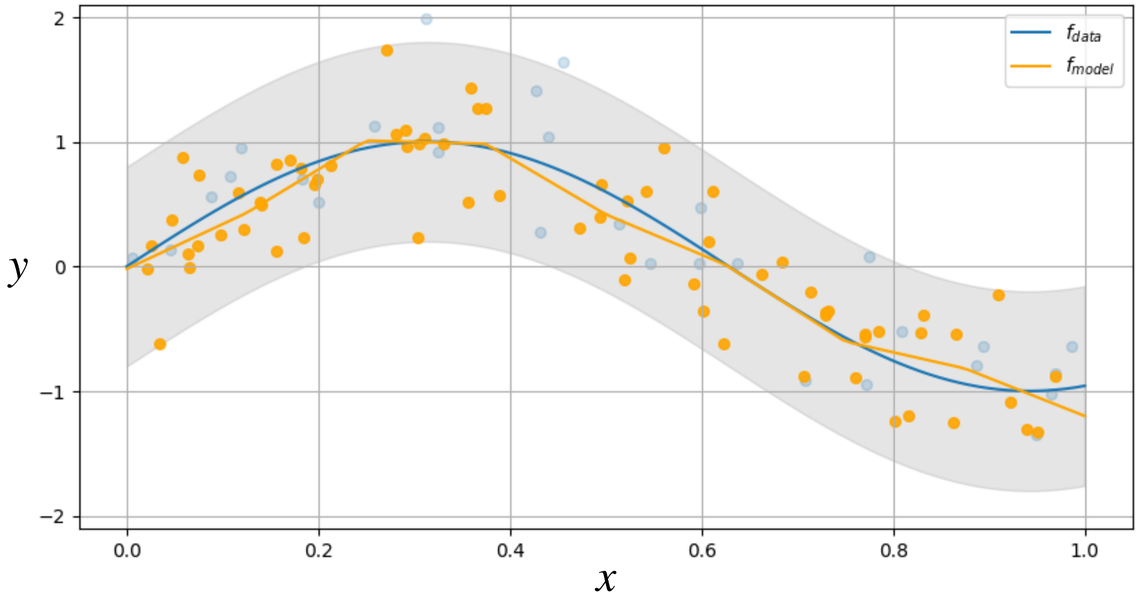
Fully connected layer, ReLU activation with 8 hidden units, $f_{model}(x;\theta) = \hat{y}$

Training loss $L_{SE}(\theta) = \frac{1}{n}\sum_{i=1}^{n}\frac{1}{2}(y^{(i)} - \hat{y}^{(i)})^2$

Optimal parameters $\hat{\theta} = argmin_{\theta} L(\theta)$

Errors

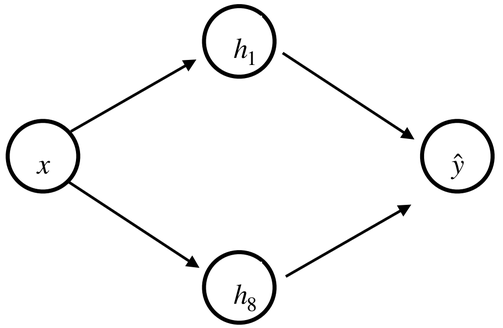
Regression problem



$(x^{(i)}, y^{(i)})$ data points generated through f_{data} with some noise σ

Different random split:

Training set D ● with size n



Training error = $\frac{1}{n}\sum_{i=1}^{n}\frac{1}{2}(y^{(i)} - \hat{y}^{(i)})^2$ ● = 0.09

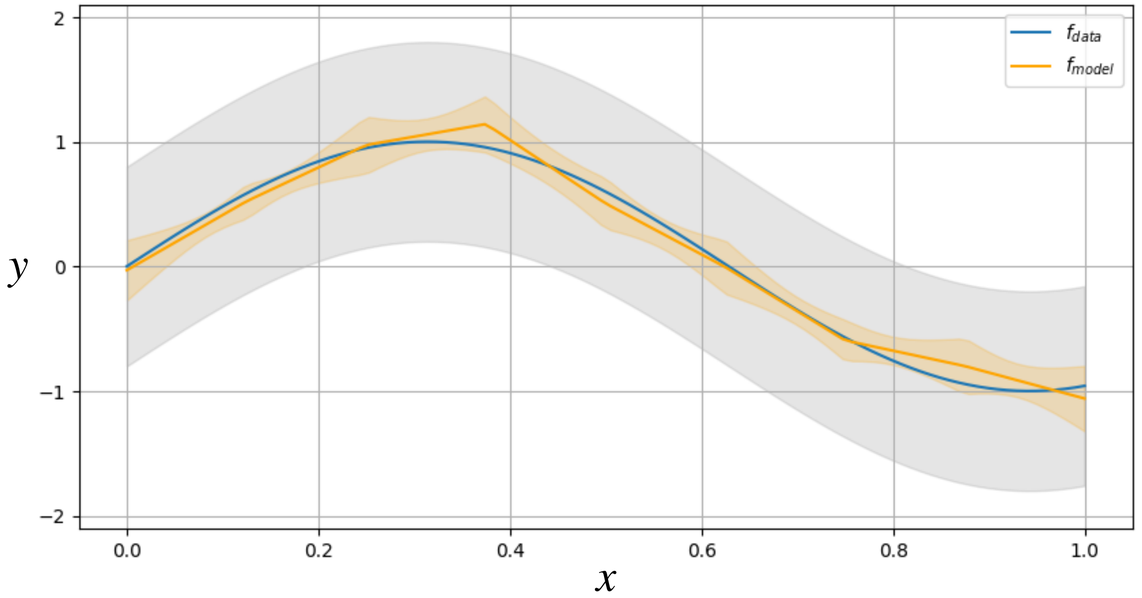
Fully connected layer, ReLU activation with 8 hidden units, $f_{model}(x;\theta) = \hat{y}$

Training loss $L_{SE}(\theta) = \frac{1}{n}\sum_{i=1}^{n}\frac{1}{2}(y^{(i)} - \hat{y}^{(i)})^2$

Optimal parameters $\hat{\theta} = argmin_{\theta} L(\theta)$

Errors

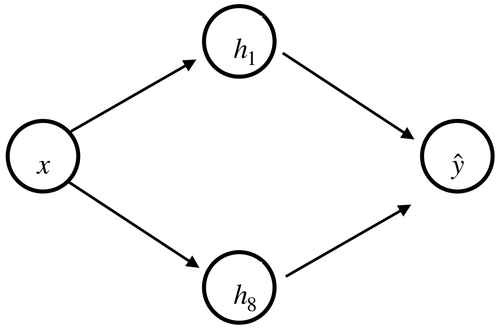
Regression problem



$(x^{(i)}, y^{(i)})$ data points generated through f_{data} with some noise σ

Average over different random split:

Training set D • with size n



Fully connected layer, ReLU activation with 8 hidden units, $f_{model}(x;\theta) = \hat{y}$

Training loss $L_{SE}(\theta) = \frac{1}{n}\sum_{i=1}^{n}\frac{1}{2}(y^{(i)} - \hat{y}^{(i)})^2$

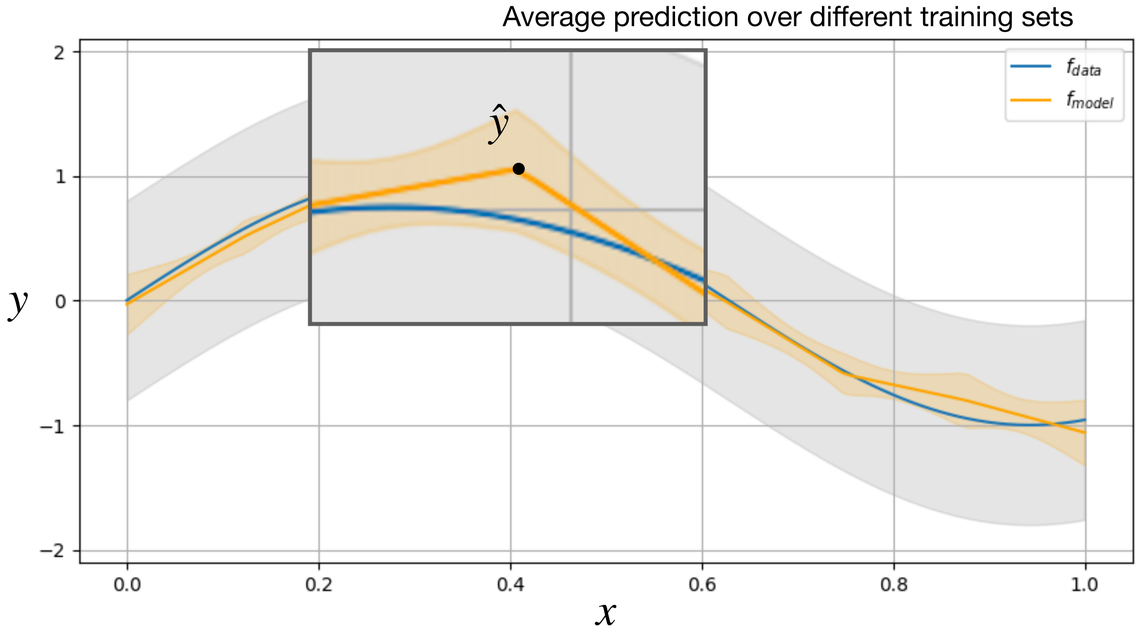
Optimal parameters $\hat{\theta} = argmin_\theta L(\theta)$

Generalization error

(over unseen data points, ≈ test error)

Errors

Regression problem



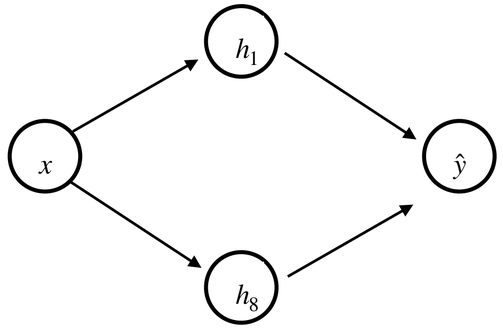
Generalization error

(over unseen data points, ≈ test error)

$(x^{(i)}, y^{(i)})$ data points generated through f_{data} with some noise σ

Average over different random split:

Training set D ● with size n



Fully connected layer, ReLU activation with 8 hidden units, $f_{model}(x;\theta) = \hat{y}$

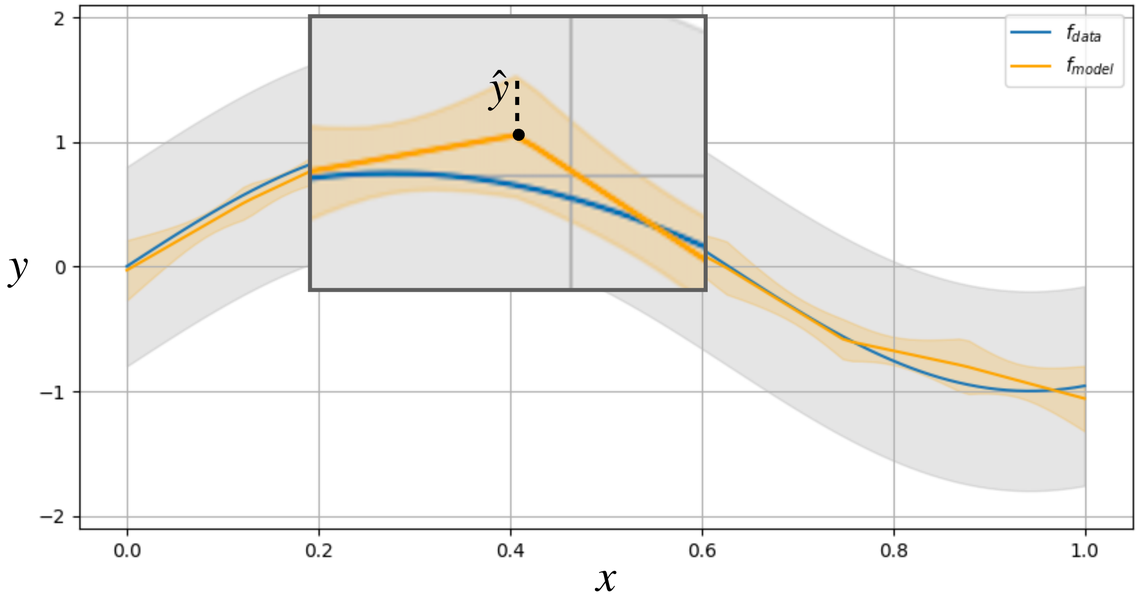
Training loss $L_{SE}(\theta) = \frac{1}{n}\sum_{i=1}^{n}\frac{1}{2}(y^{(i)} - \hat{y}^{(i)})^2$

Optimal parameters $\hat{\theta} = argmin_{\theta} L(\theta)$

Errors

Regression problem

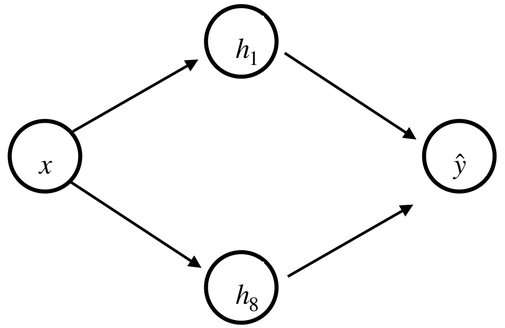
Standard deviation of the predictions over different training sets



$\left(x^{(i)}, y^{(i)}\right)$ data points generated through f_{data} with some noise σ

Average over different random split:

Training set D ● with size n



Fully connected layer, ReLU activation with 8 hidden units, $f_{model}(x;\theta) = \hat{y}$

Training loss $L_{SE}(\theta) = \frac{1}{n}\sum_{i=1}^{n}\frac{1}{2}(y^{(i)} - \hat{y}^{(i)})^2$

Optimal parameters $\hat{\theta} = argmin_{\theta} L(\theta)$

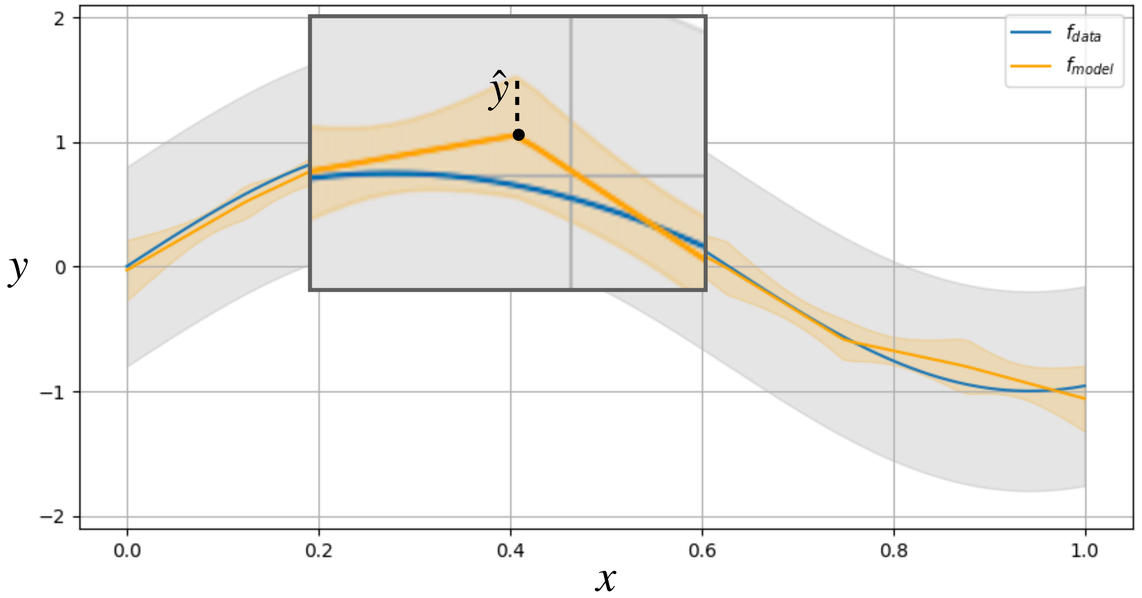
Generalization error

(over unseen data points, ≈ test error)

Errors

Regression problem

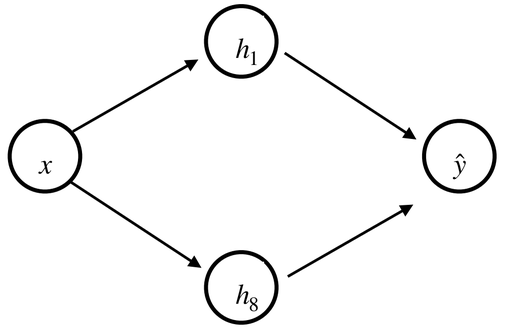
Standard deviation of the predictions over different training sets



$\left(x^{(i)}, y^{(i)}\right)$ data points generated through f_{data} with some noise σ

Average over different random split:

Training set D ● with size n



Fully connected layer, ReLU activation with 8 hidden units, $f_{model}(x;\theta) = \hat{y}$

Training loss $L_{SE}(\theta) = \frac{1}{n}\sum_{i=1}^{n}\frac{1}{2}(y^{(i)} - \hat{y}^{(i)})^2$

Optimal parameters $\hat{\theta} = argmin_{\theta} L(\theta)$

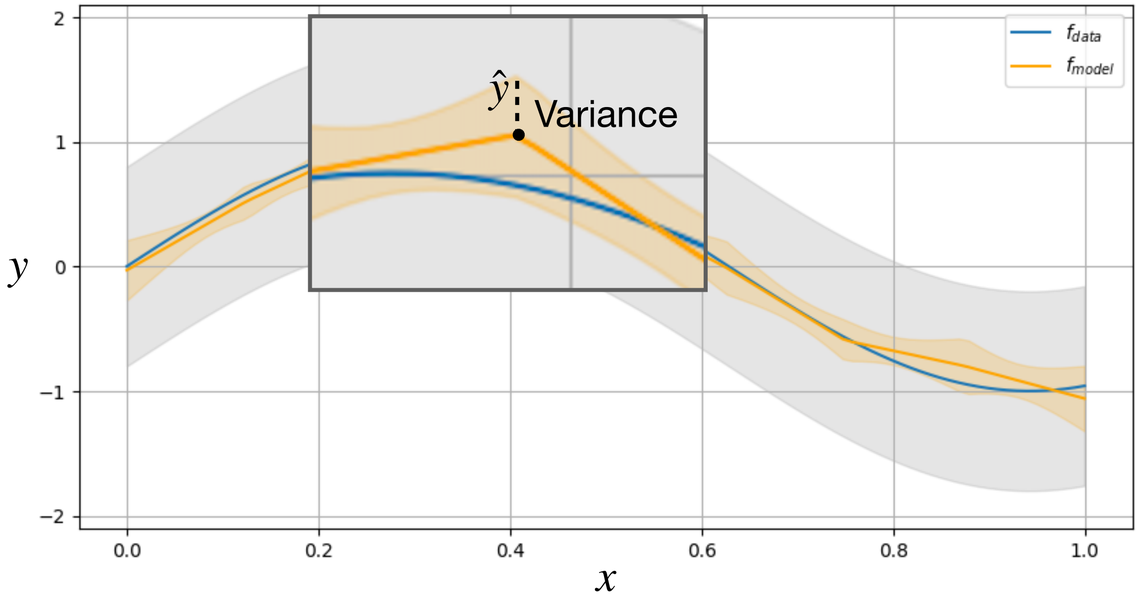
Generalization error

↓

Errors

Regression problem

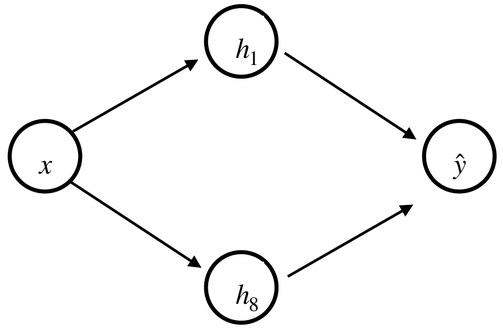
Standard deviation of the predictions over different training sets



$\left(x^{(i)}, y^{(i)}\right)$ data points generated through f_{data} with some noise σ

Average over different random split:

Training set D ● with size n



Fully connected layer, ReLU activation with 8 hidden units, $f_{model}(x;\theta) = \hat{y}$

Training loss $L_{SE}(\theta) = \frac{1}{n}\sum_{i=1}^{n}\frac{1}{2}(y^{(i)} - \hat{y}^{(i)})^2$

Optimal parameters $\hat{\theta} = argmin_{\theta} L(\theta)$

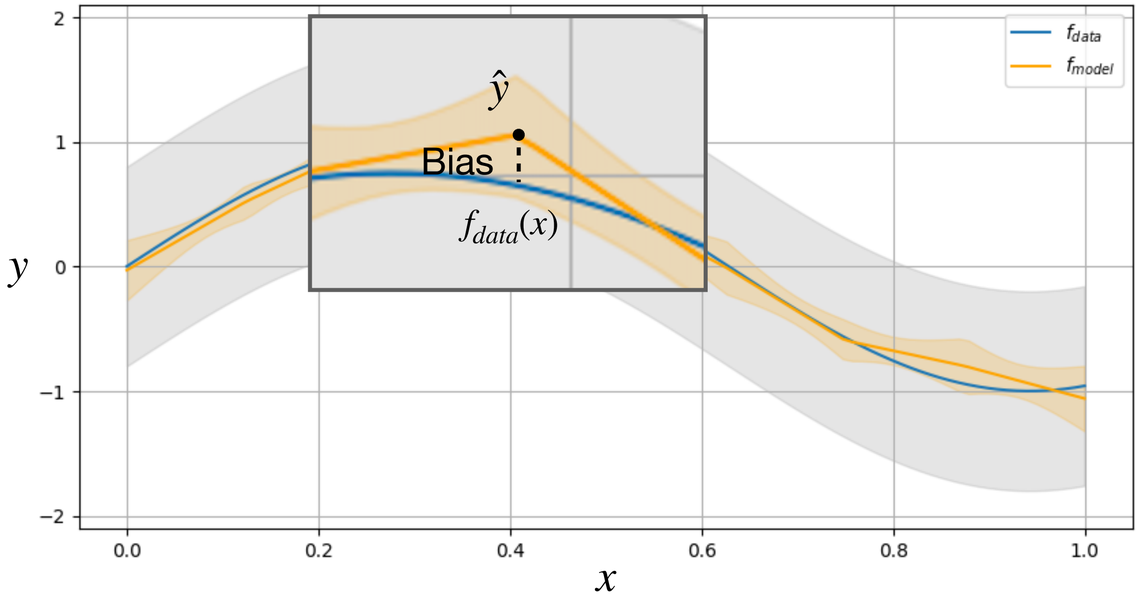
Generalization error

↓

Variance

Errors

Regression problem



Generalization error

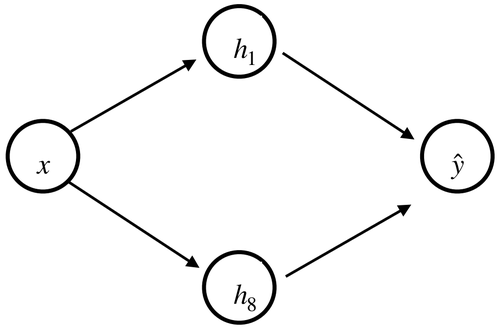
↓

Variance + Bias

$(x^{(i)}, y^{(i)})$ data points generated through f_{data} with some noise σ

Average over different random split:

Training set D ● with size n



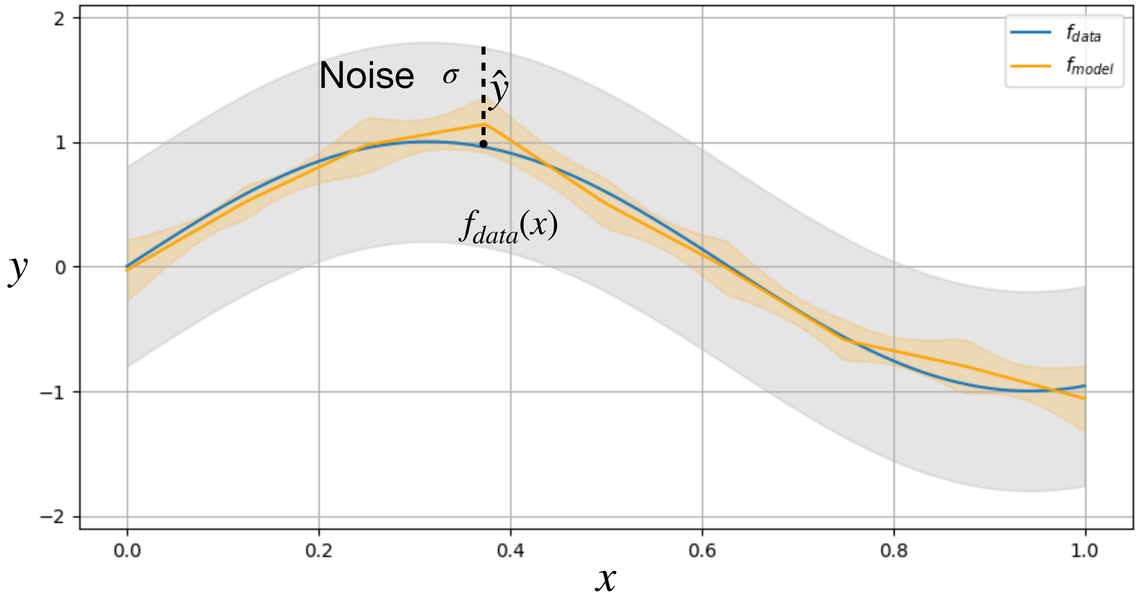
Fully connected layer, ReLU activation with 8 hidden units, $f_{model}(x;\theta) = \hat{y}$

Training loss $L_{SE}(\theta) = \frac{1}{n}\sum_{i=1}^{n}\frac{1}{2}(y^{(i)} - \hat{y}^{(i)})^2$

Optimal parameters $\hat{\theta} = argmin_{\theta} L(\theta)$

Errors

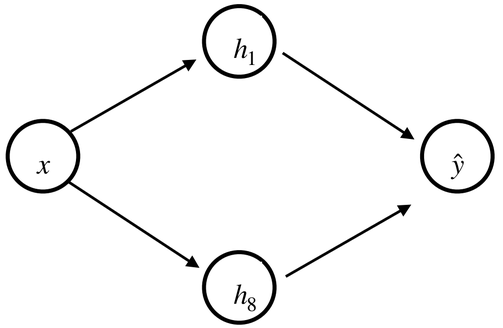
Regression problem



$\left(x^{(i)}, y^{(i)}\right)$ data points generated through f_{data} with some noise σ

Average over different random split:

Training set D • with size n



Fully connected layer, ReLU activation with 8 hidden units, $f_{model}(x;\theta) = \hat{y}$

Training loss $L_{SE}(\theta) = \frac{1}{n}\sum_{i=1}^{n}\frac{1}{2}(y^{(i)} - \hat{y}^{(i)})^2$

Optimal parameters $\hat{\theta} = argmin_{\theta} L(\theta)$

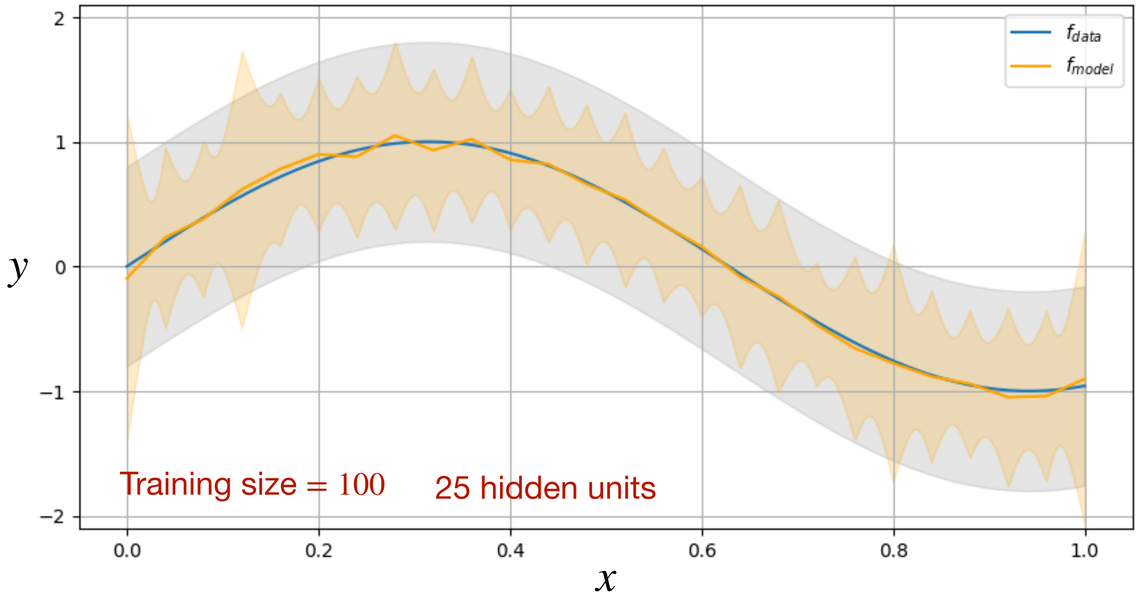
Generalization error

=

Variance + Bias + Noise

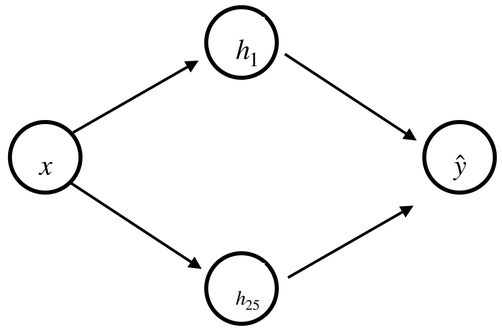
Errors: question

Regression problem



$\left(x^{(i)}, y^{(i)}\right)$ data points generated through f_{data} with some noise σ

Average over different training sets with size $n = 100$



Fully connected layer, ReLU activation with **25** hidden units, $f_{model}(x;\theta) = \hat{y}$

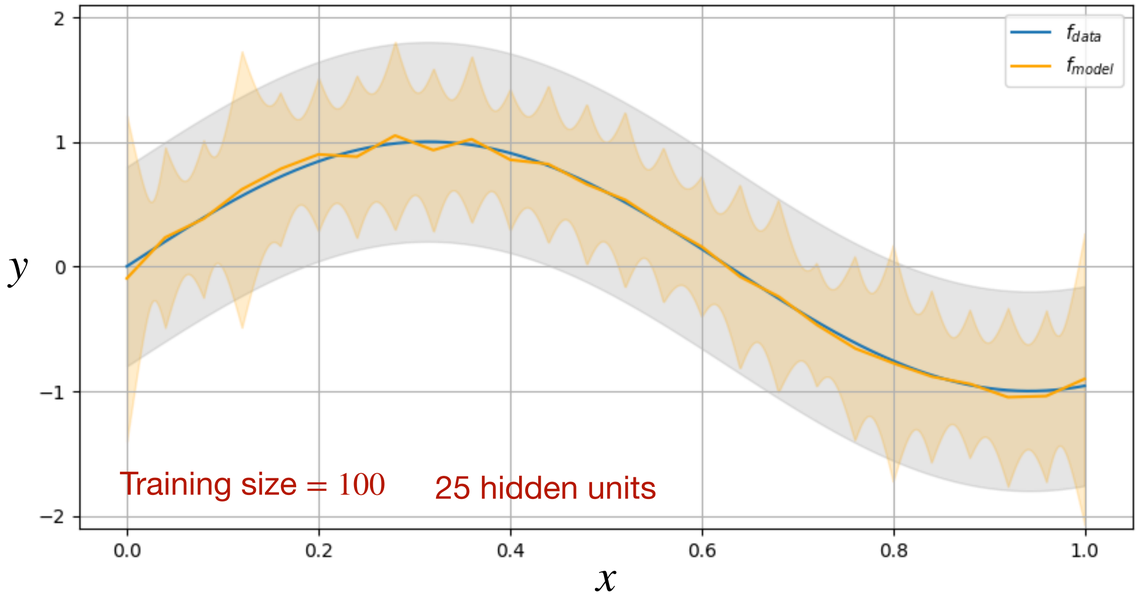
Generalization error

=

Variance + Bias + Noise

Errors: question

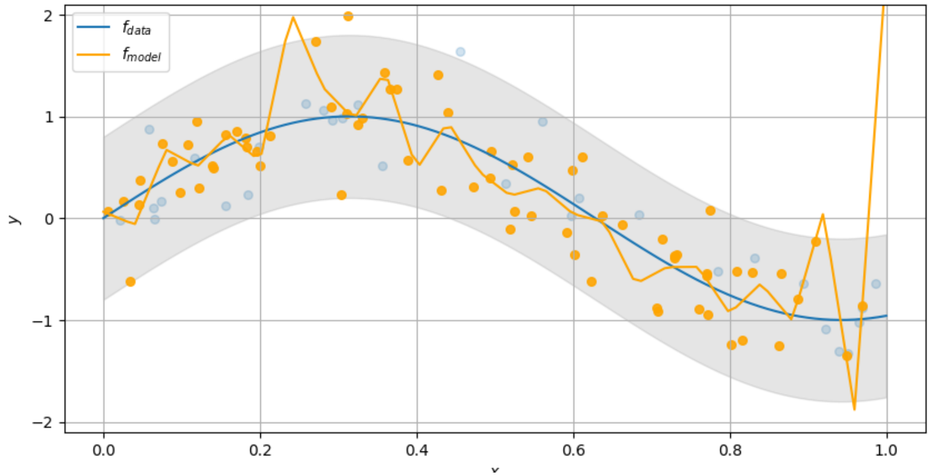
Regression problem



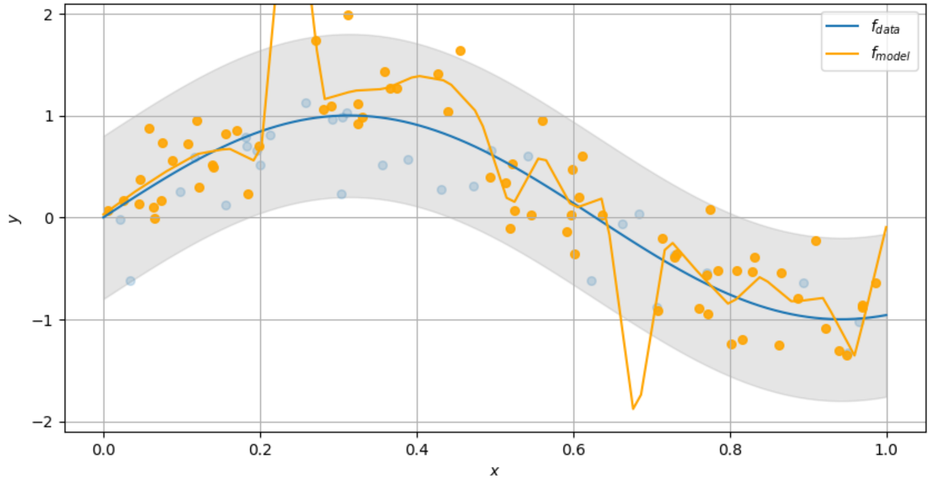
Generalization error

=

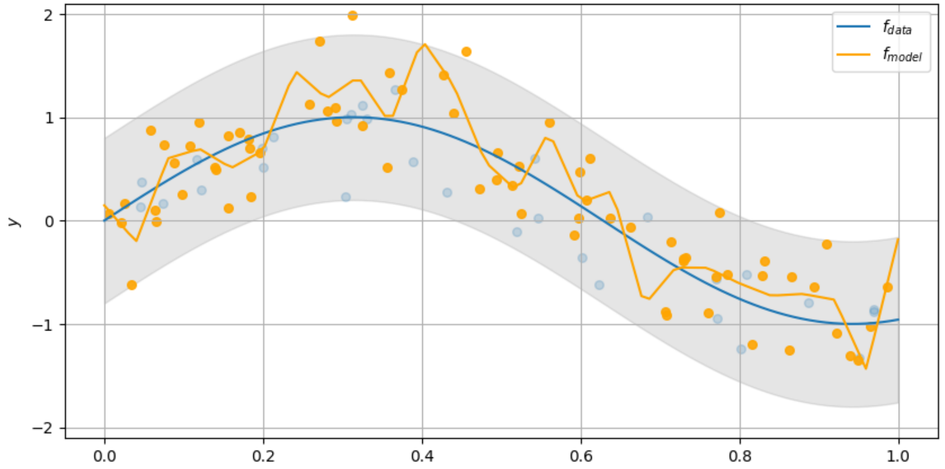
Variance + Bias + Noise



Training set D_1



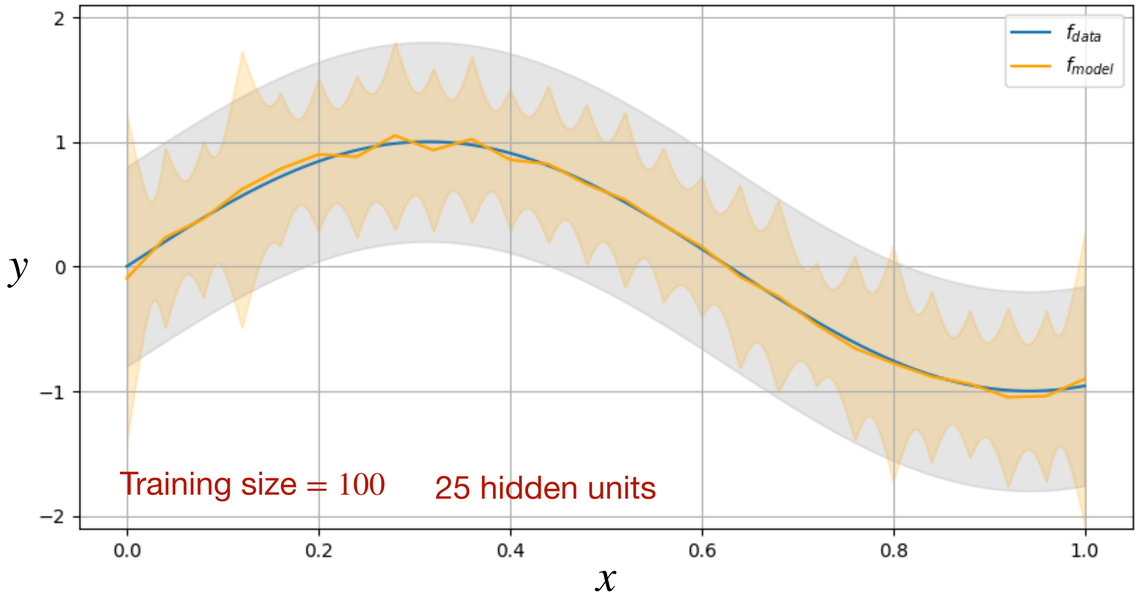
Training set D_2



Training set D_3

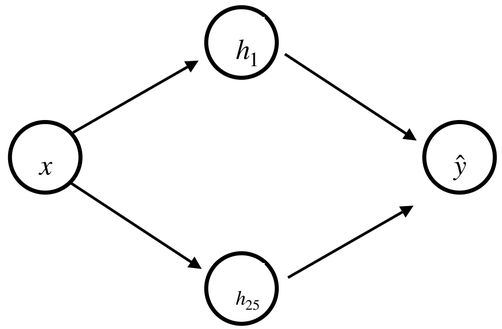
Errors: question

Regression problem



$\left(x^{(i)}, y^{(i)}\right)$ data points generated through f_{data} with some noise σ

Average over different training sets with size $n = 100$



Fully connected layer, ReLU activation with **25** hidden units, $f_{model}(x;\theta) = \hat{y}$

Generalization error

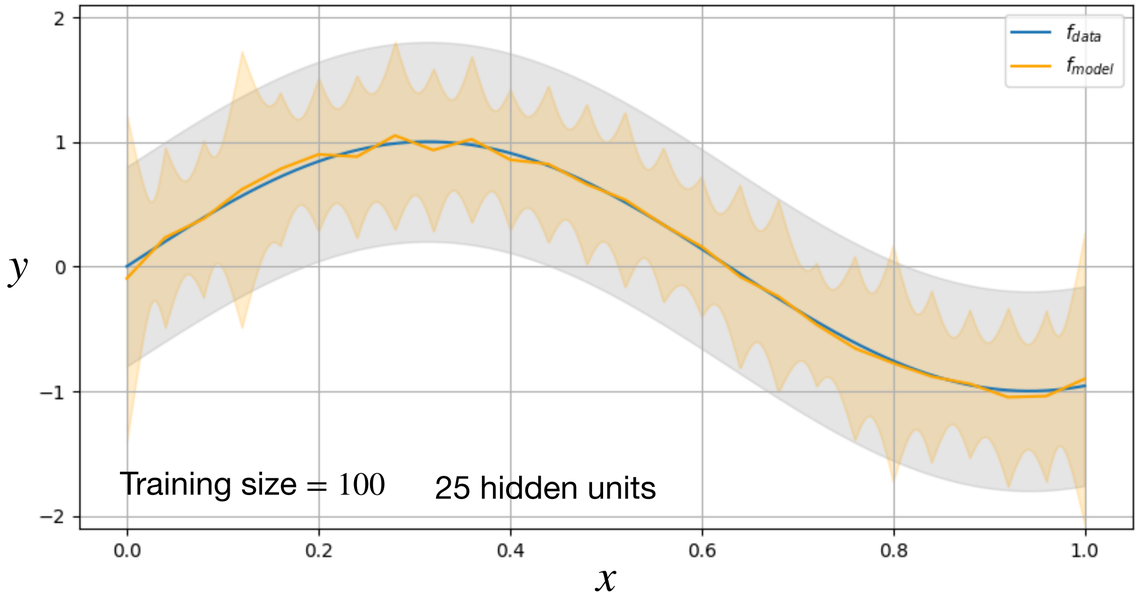
=

Variance + Bias + Noise

Which component is contributing the most to the generalization error?

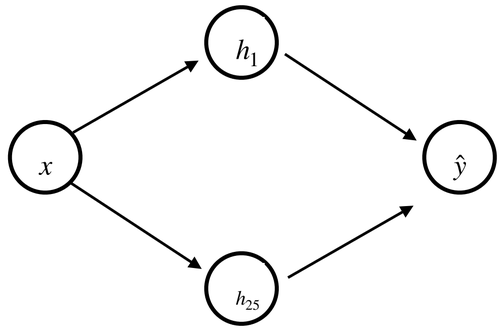
Errors: question

Regression problem



$\left(x^{(i)}, y^{(i)}\right)$ data points generated through f_{data} with some noise σ

Average over different training sets with size $n = 100$



Fully connected layer, ReLU activation with 25 hidden units, $f_{model}(x; \theta) = \hat{y}$

Generalization error

0.234

=

Variance + Bias + Noise

0.076 0.001 0.16

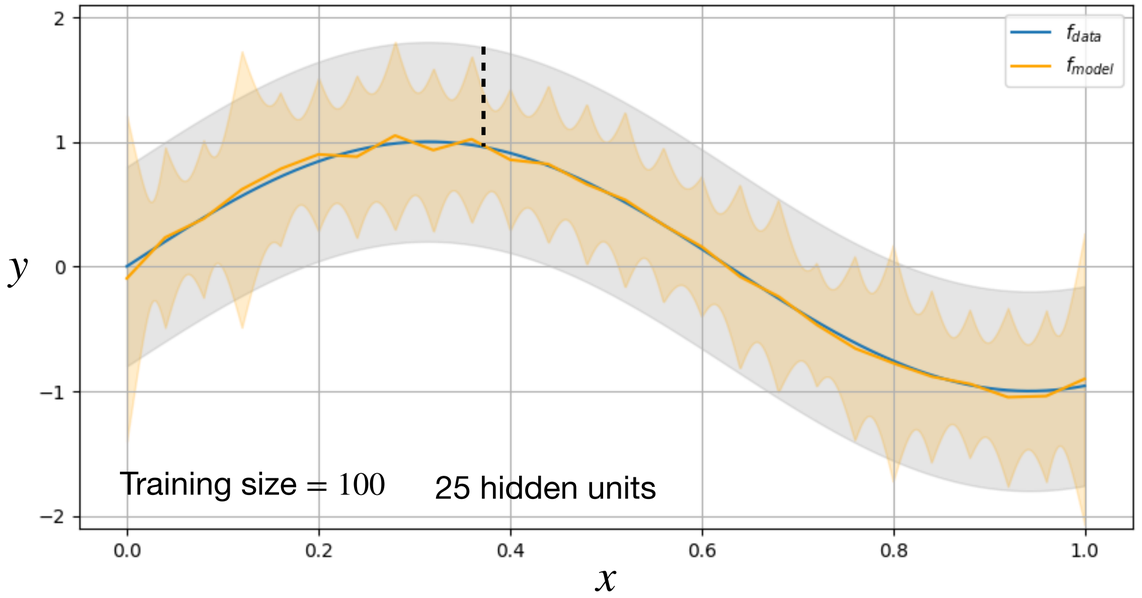
```
Training error 0.124
Test error 0.234

Variance 0.076
Bias 0.001
Noise 0.16
```

Which component is contributing the most to the generalization error?

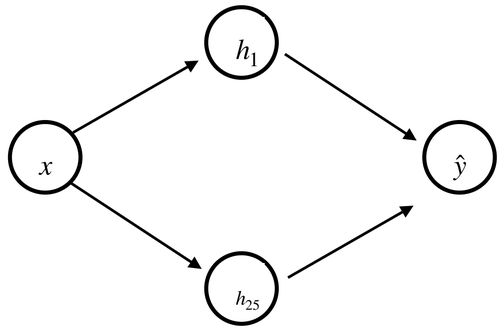
Errors: question

Regression problem



$\left(x^{(i)}, y^{(i)}\right)$ data points generated through f_{data} with some noise σ

Average over different training sets with size $n = 100$



Fully connected layer, ReLU activation with 25 hidden units, $f_{model}(x; \theta) = \hat{y}$

Generalization error

0.234

=

Variance + Bias + Noise

0.076 0.001 0.16

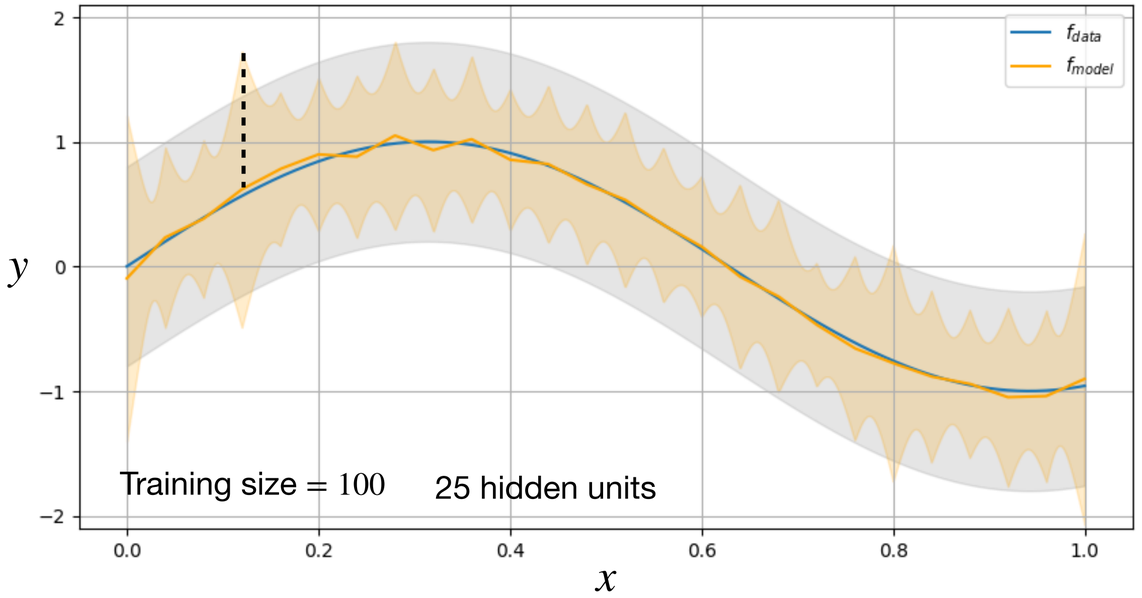
```
Training error 0.124
Test error 0.234

Variance 0.076
Bias 0.001
Noise 0.16
```

Which component is contributing the most to the generalization error?

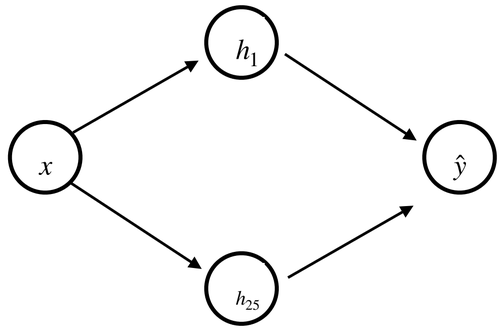
Errors: question

Regression problem



$\left(x^{(i)}, y^{(i)}\right)$ data points generated through f_{data} with some noise σ

Average over different training sets with size $n = 100$



Fully connected layer, ReLU activation with 25 hidden units, $f_{model}(x;\theta) = \hat{y}$

Generalization error

0.234

=

Variance + Bias + Noise

0.076 0.001 0.16

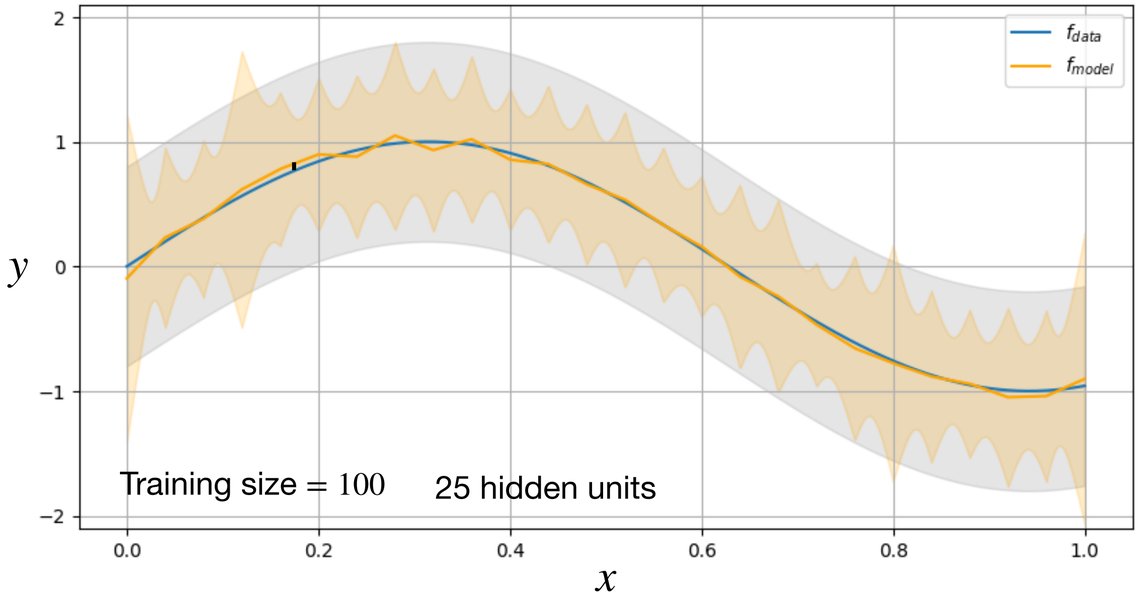
```
Training error 0.124
Test error 0.234

Variance 0.076
Bias 0.001
Noise 0.16
```

Which component is contributing the most to the generalization error?

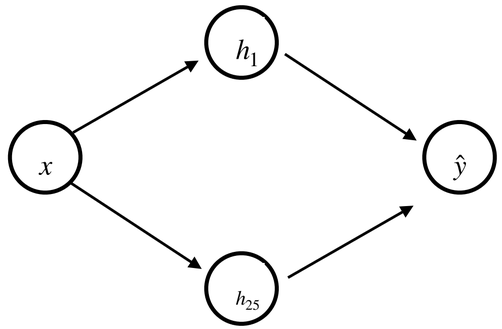
Errors: question

Regression problem



$\left(x^{(i)}, y^{(i)}\right)$ data points generated through f_{data} with some noise σ

Average over different training sets with size $n = 100$



Fully connected layer, ReLU activation with 25 hidden units, $f_{model}(x; \theta) = \hat{y}$

Generalization error

0.234

=

Variance + Bias + Noise

0.076 0.001 0.16

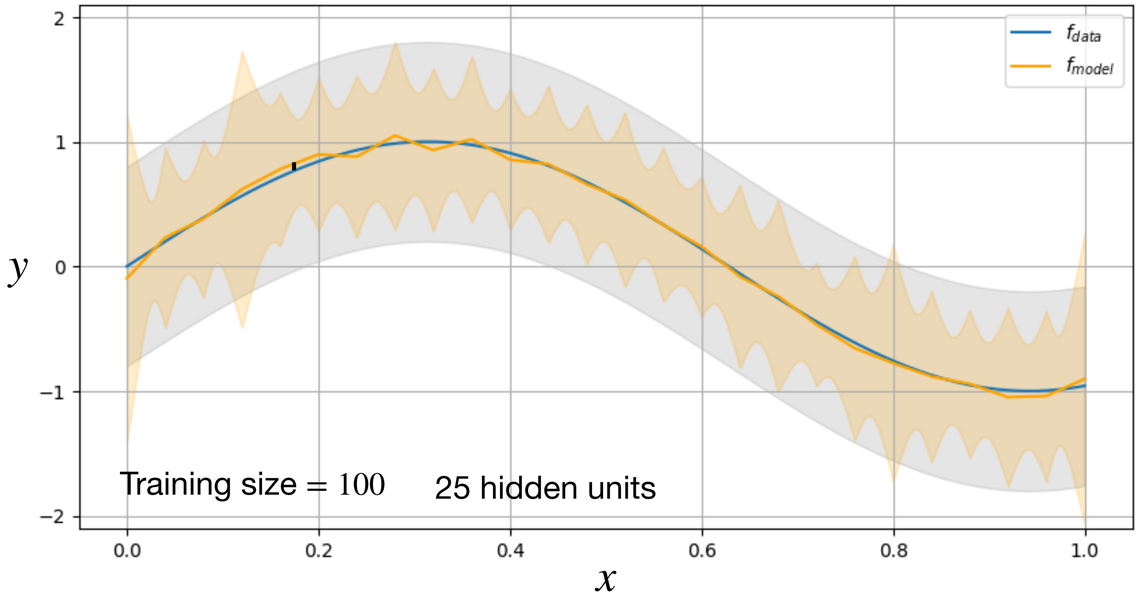
```
Training error 0.124
Test error 0.234

Variance 0.076
Bias 0.001
Noise 0.16
```

Which component is contributing the most to the generalization error?

Errors: question

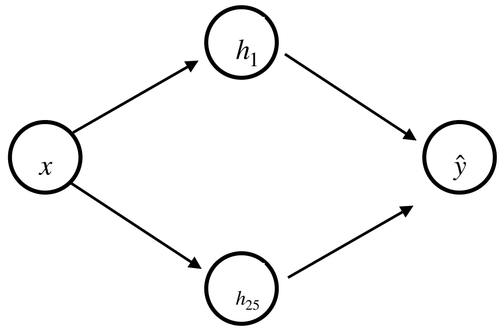
Regression problem



Underfitting, overfitting or proper fitting?

$\left(x^{(i)}, y^{(i)}\right)$ data points generated through f_{data} with some noise σ

Average over different training sets with size $n = 100$



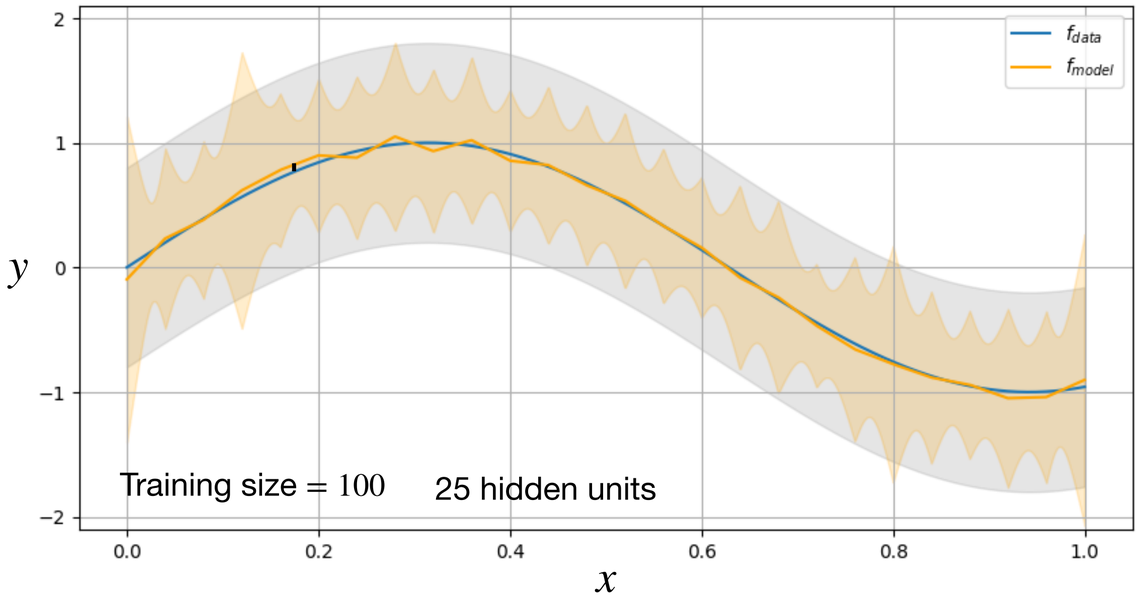
Fully connected layer, ReLU activation with 25 hidden units, $f_{model}(x;\theta) = \hat{y}$

```
Training error 0.124
Test error 0.234

Variance 0.076
Bias 0.001
Noise 0.16
```

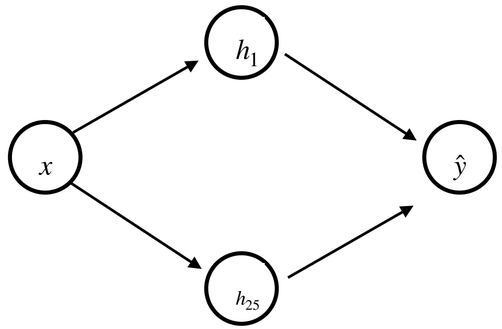
Errors: question

Regression problem



$\left(x^{(i)}, y^{(i)}\right)$ data points generated through f_{data} with some noise σ

Average over different training sets with size $n = 100$



Fully connected layer, ReLU activation with 25 hidden units, $f_{model}(x;\theta) = \hat{y}$

Underfitting, overfitting or proper fitting?

Generalization gap = test error - training error is large

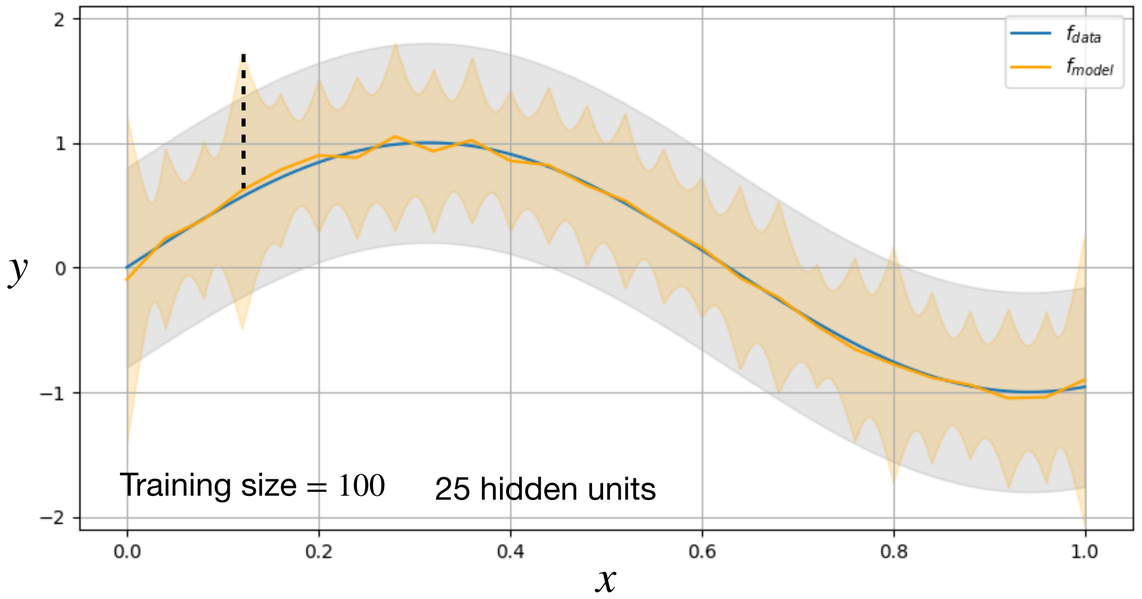
The model overfits the noise in the training data!

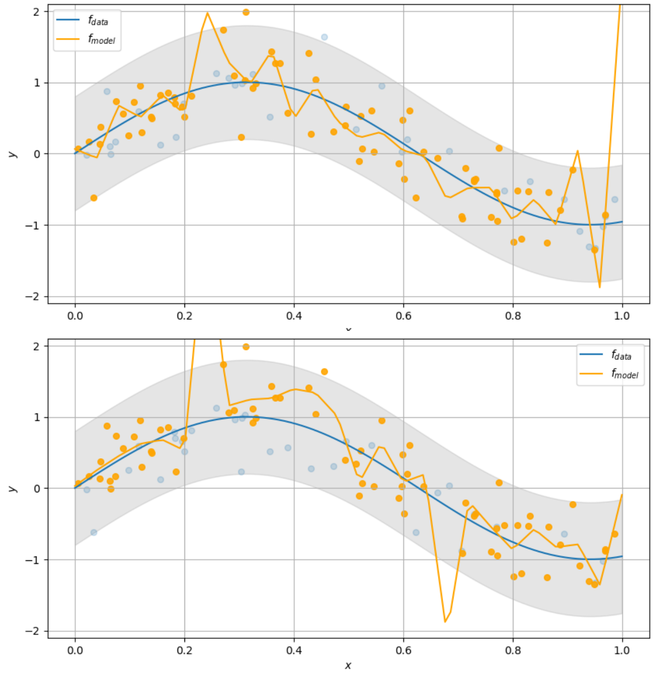
```
Training error 0.124   < 0.16 Noise
Test error 0.234       > 0.16 Noise

Variance 0.076
Bias 0.001
Noise 0.16
```

Overfitting

Regression problem





With 100 training samples, a model with 25 hidden units has too much representation capacity

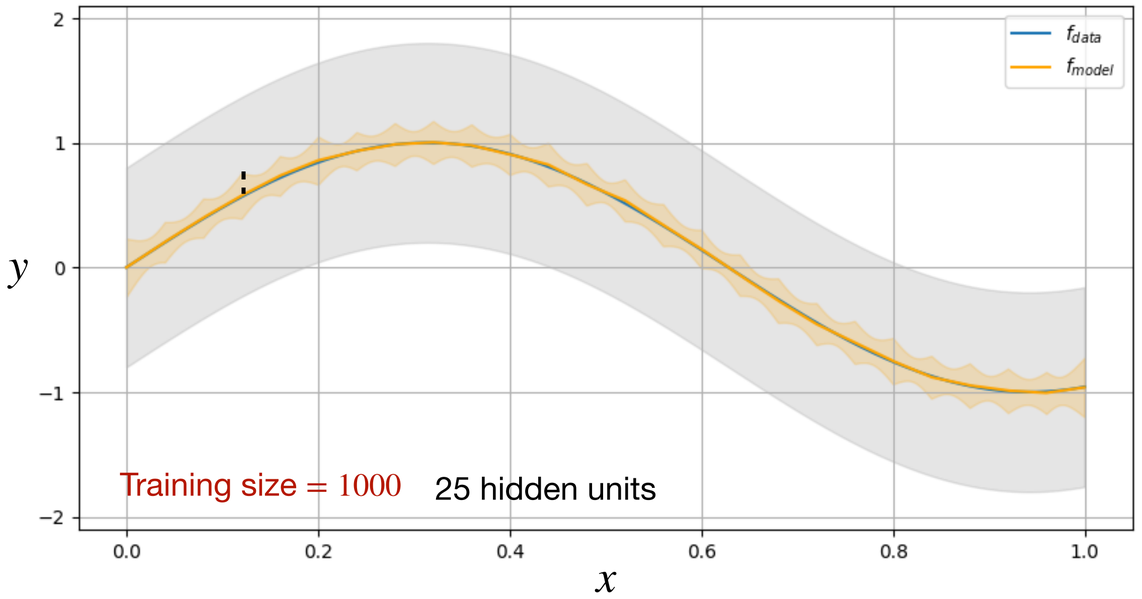
Generalization gap = 0.11

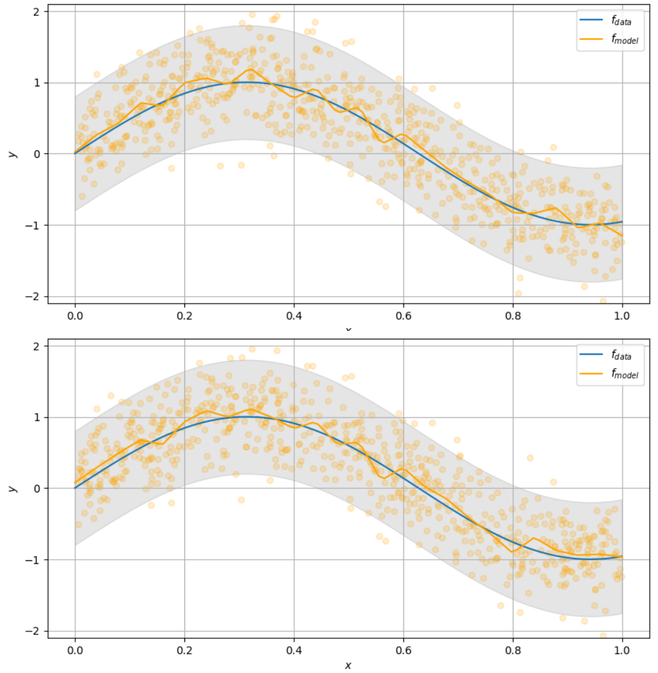
```
Training error 0.124
Test error 0.234

Variance 0.076
Bias 0.001
Noise 0.16
```

Overfitting

Regression problem





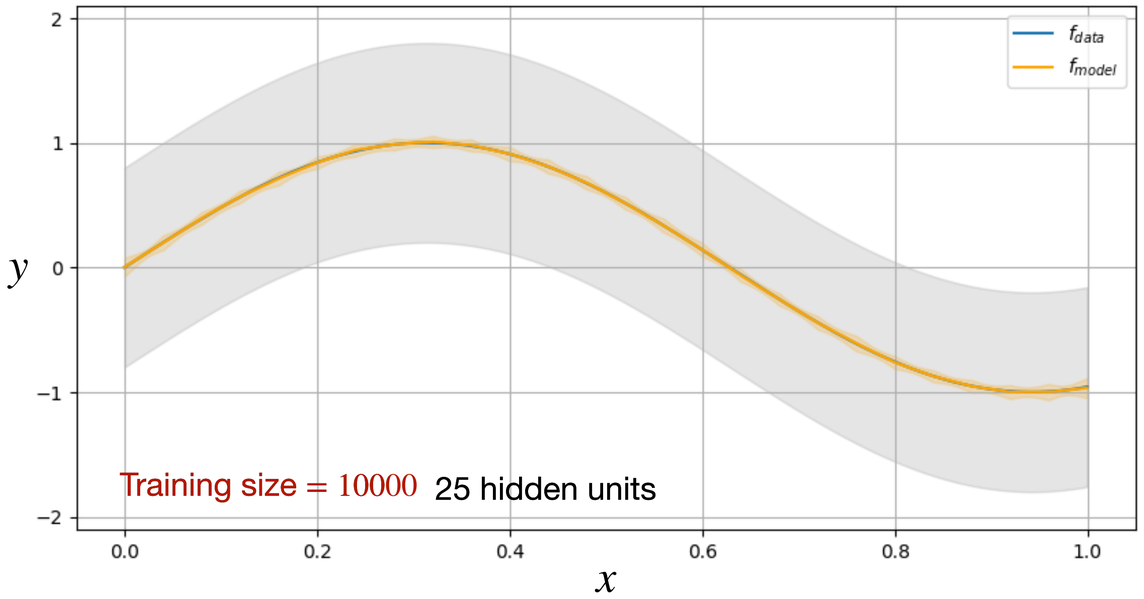
Increasing the number of training data improve always overfitting

Generalization gap = 0.006

```
Training error 0.156
Test error 0.162
```

Overfitting

Regression problem



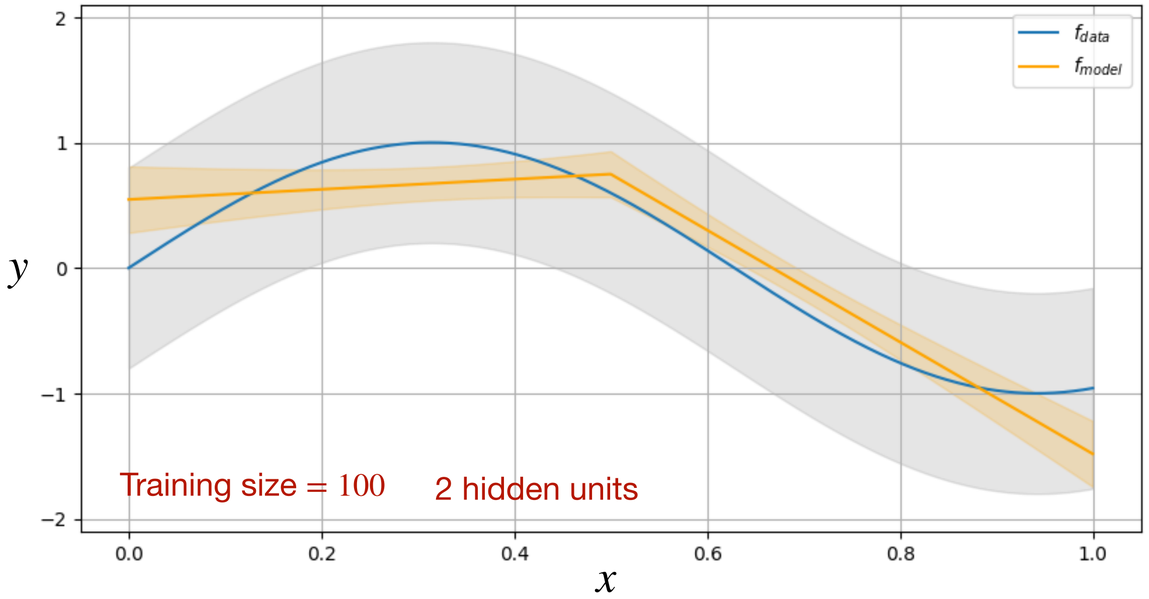
Increasing the number of training data improve always overfitting

Generalization gap ≈ 0

```
Training error 0.16
Test error 0.16
```

Fitting: question

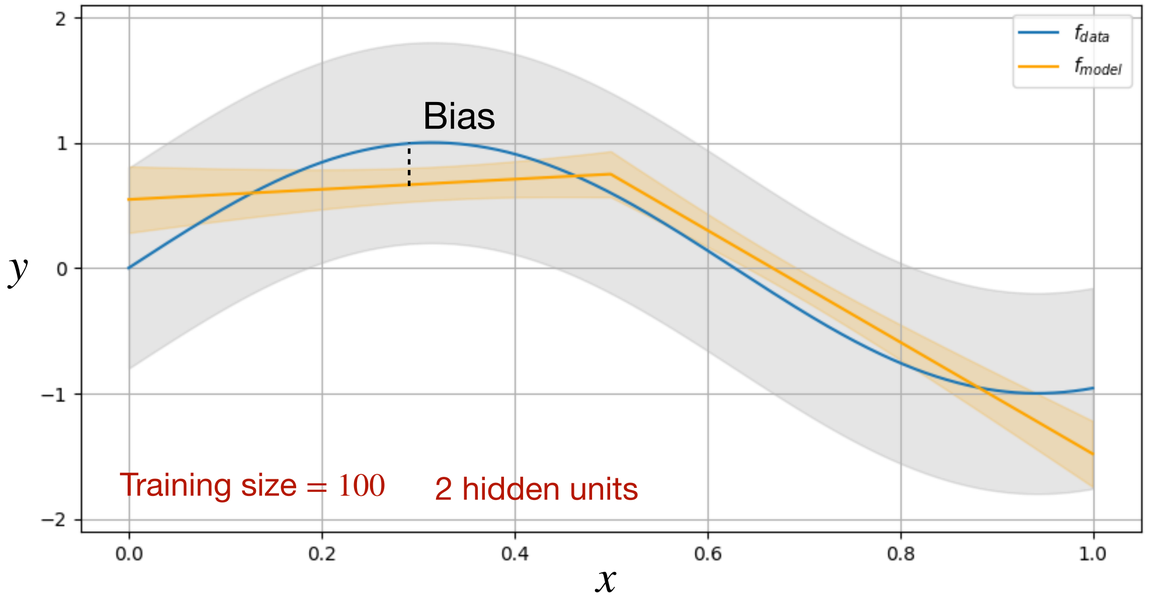
Regression problem



1. Which component is contributing the most to the generalisation error?
2. Underfitting, overfitting, proper fitting?

Fitting: question

Regression problem



1. Which component is contributing the most to the generalisation error?
2. **Underfitting**, overfitting, proper fitting?

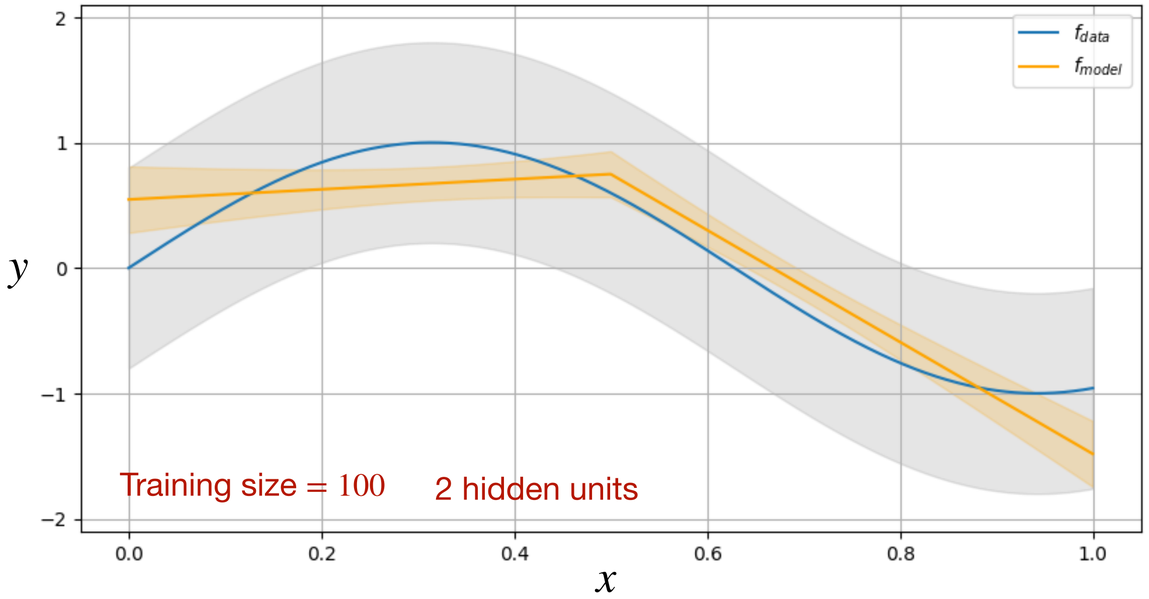
Generalization gap = 0.008

```
Training error 0.208
Test error 0.216

Variance 0.007
Bias 0.051
Noise 0.16
```

Fitting: question

Regression problem



1. Which component is contributing the most to the generalisation error?
2. **Underfitting**, overfitting, proper fitting?

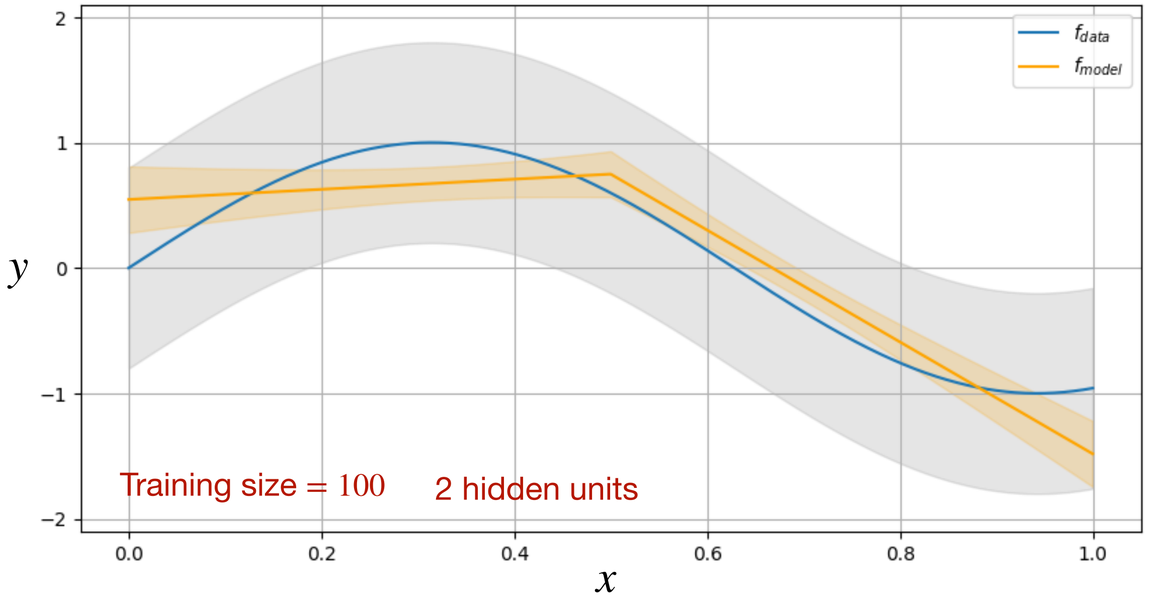
Generalization gap = 0.008

```
Training error 0.208 > 0.16 noise
Test error 0.216

Variance 0.007
Bias 0.051
Noise 0.16
```

Fitting: question

Regression problem



1. Which component is contributing the most to the generalisation error?
2. **Underfitting**, overfitting, proper fitting?

Generalization gap = 0.008

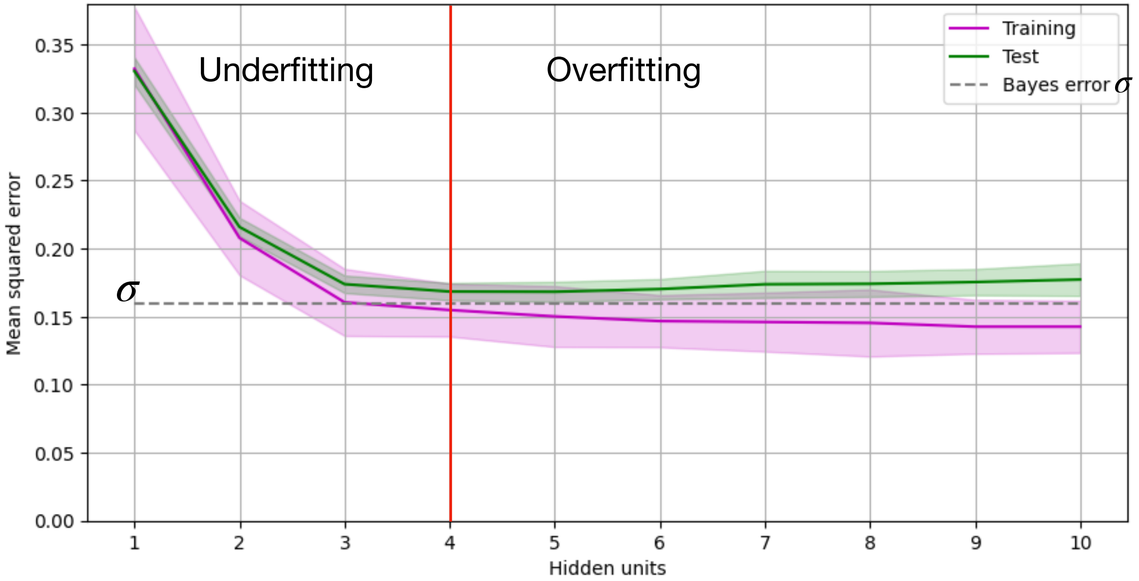
```
Training error 0.208 > 0.16 noise
Test error 0.216

Variance 0.007
Bias 0.051
Noise 0.16
```

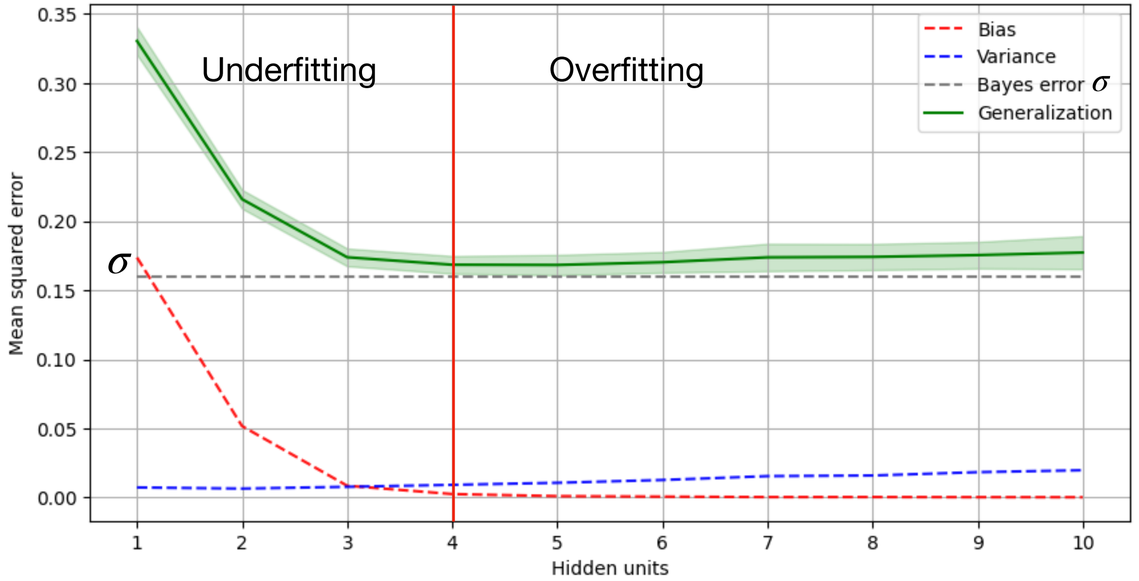
Underfitting: the model lacks capacity to predict well even the data it has been trained on!

Model Capacity

Regression problem

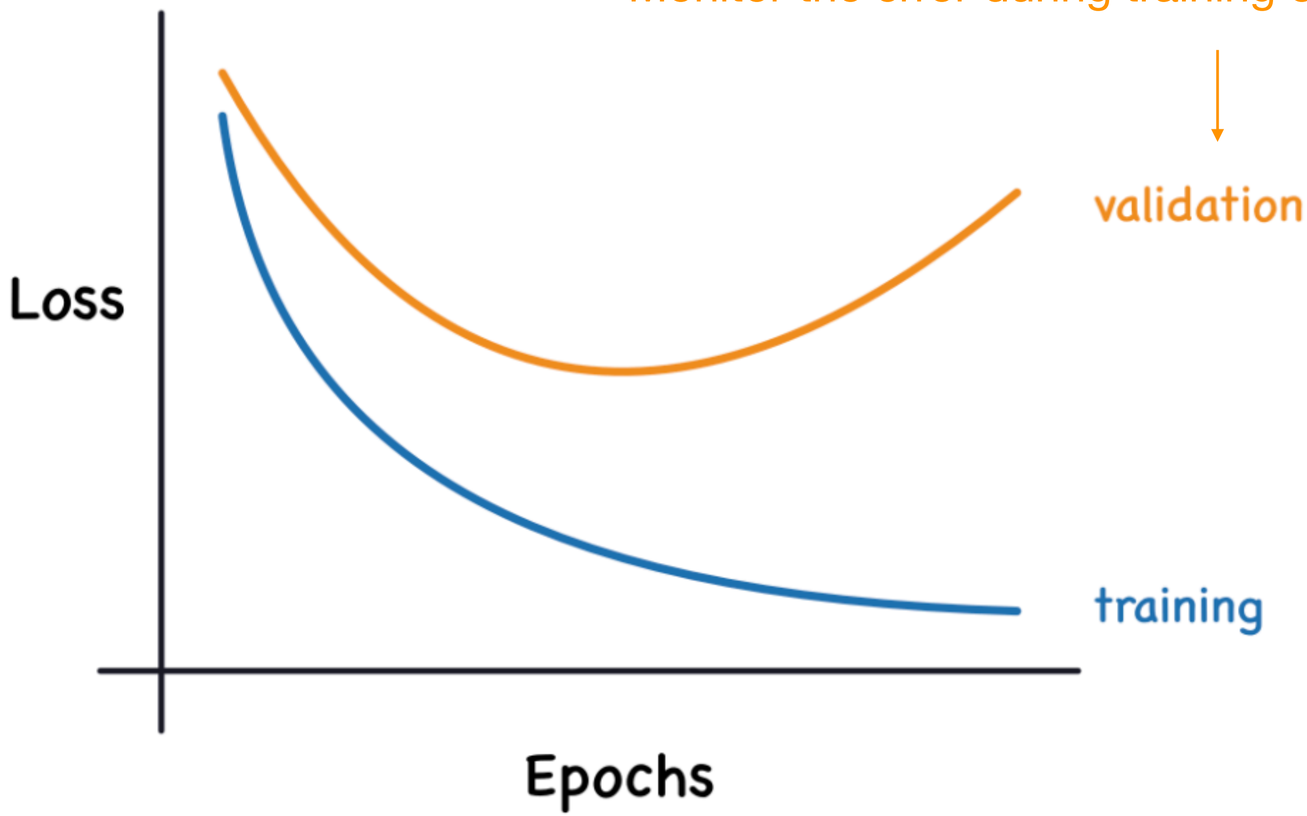


Training size $= 100$



Monitoring Fitting

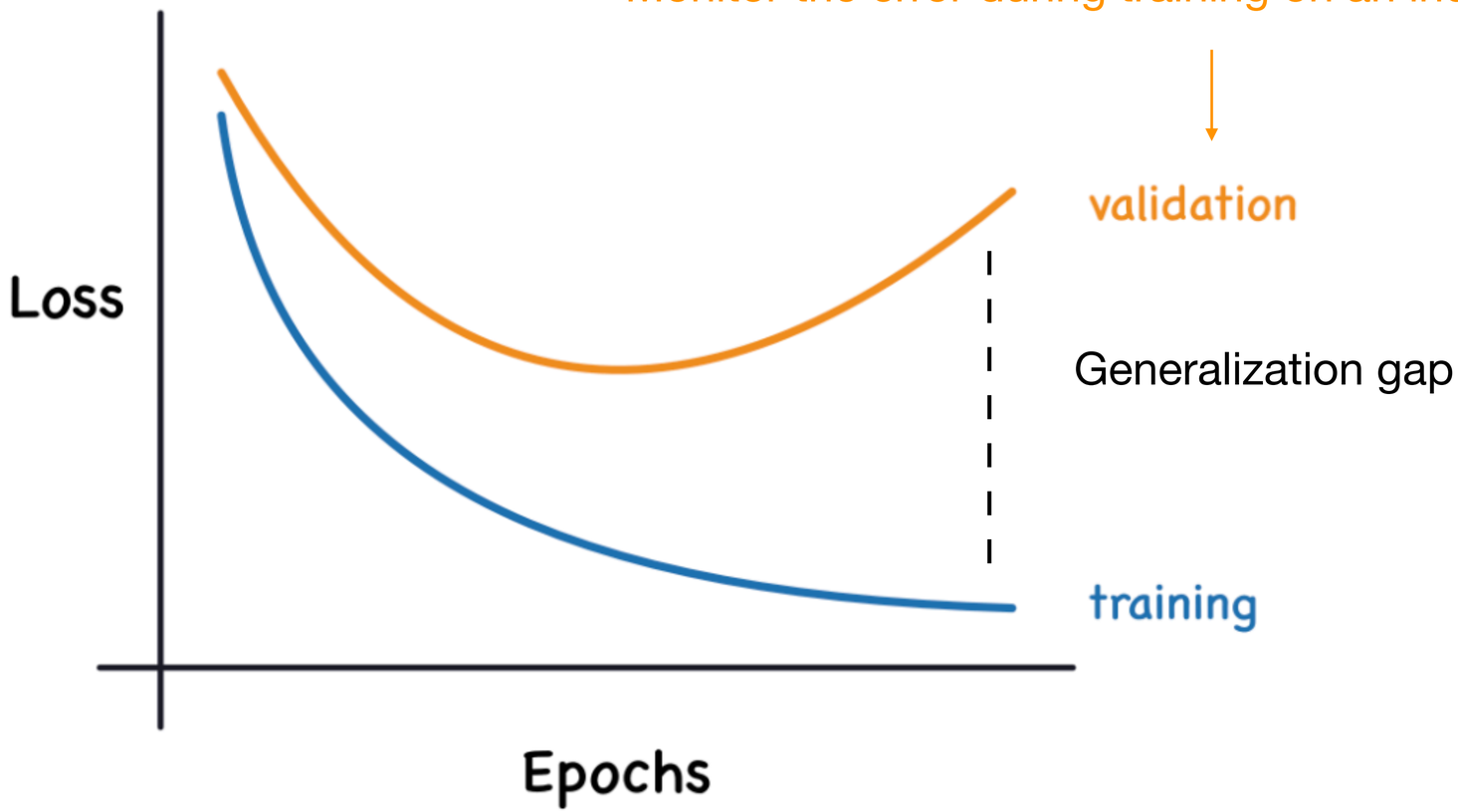
Monitor the error during training on an independent validation set



In practice, we only have access to training curves

Monitoring Fitting

Monitor the error during training on an independent validation set



In practice, we only have access to training curves

Generalization gap = training error - validation error

Regularization

Regularization are all those techniques that improve generalization: reducing the test error, possibly at the expense of increased training error

Regularization

Regularization are all those techniques that improve generalization: reducing the test error, possibly at the expense of increased training error

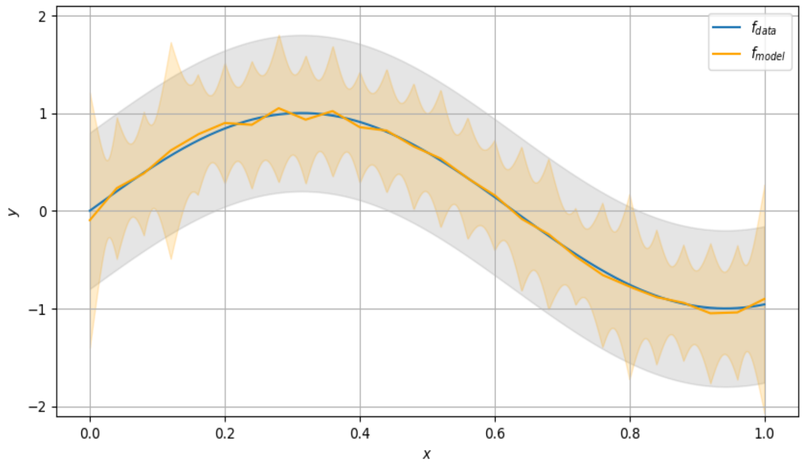
When to use it?

Regularization

Regularization are all those techniques that improve generalization: reducing the test error, possibly at the expense of increased training error

When to use it?

Overfitting (high generalization gap)



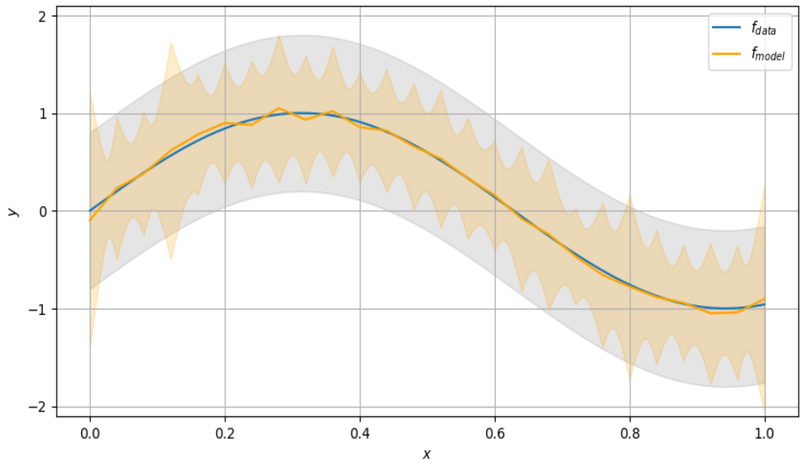
Training size $= 100$ 25 hidden units

Regularization

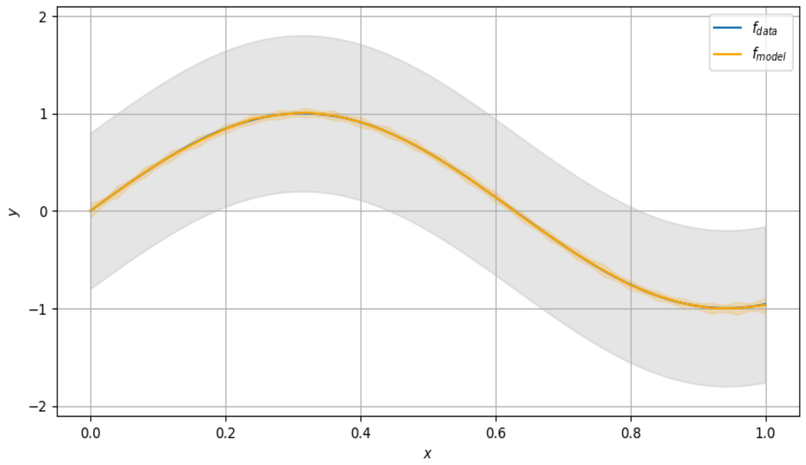
Regularization are all those techniques that improve generalization: reducing the test error, possibly at the expense of increased training error

When to use it?

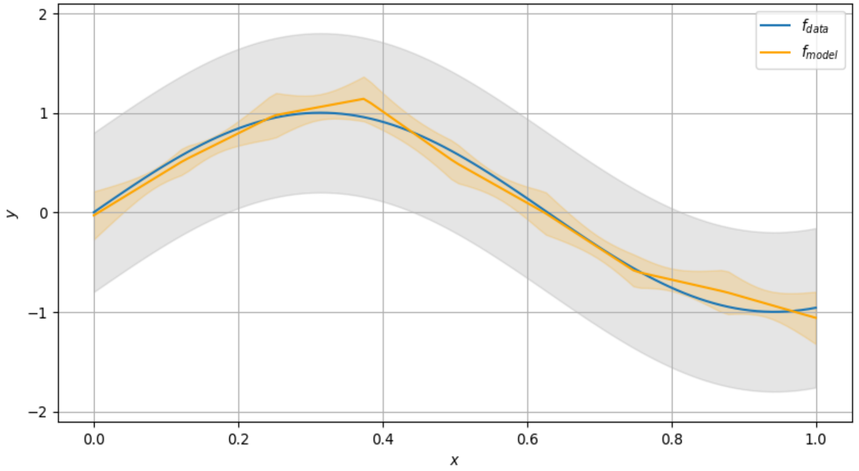
Overfitting (high generalization gap)



Training size $= 100$ 25 hidden units



Training size $= 10000$ 25 hidden units



Training size $= 100$ 8 hidden units

Use more data: amount of data is limited!

Reduce your model capacity

Regularization

Regularization are all those techniques that improve generalization: reducing the test error, possibly at the expense of increased training error

When to use it?

Overfitting (high generalization gap)

What to do?

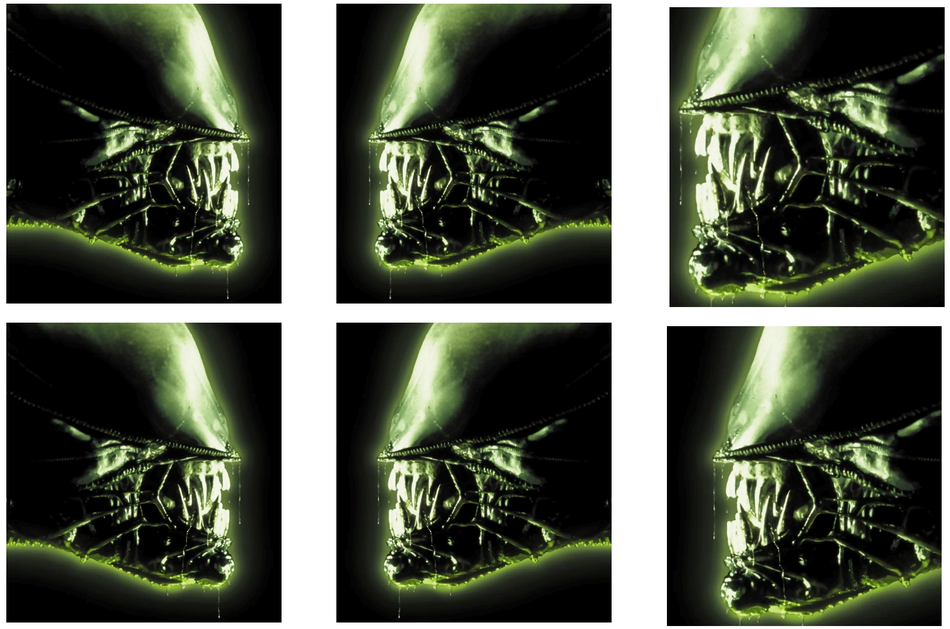
- Data Augmentation
- Weight decay/Parameter norm penalty
- Early stopping
- Dropout

Data Augmentation

Create synthetic data by applying transformations our model should be invariant to

distorted patterns with randomly picked distortion parameters. The distortions were combinations of the following planar affine transformations: horizontal and vertical translations, scaling, squeezing (simultaneous horizontal compression and vertical elongation, or the reverse), and horizontal shearing. Figure 7 shows examples of distorted patterns used for training.

'Gradient-based Learning Applied to Document Recognition', Y.LeCun, L.Bottou, Y.Bengio, P.Haffner, 1998



Only works when the invariances are known (images)

Only augment the training set!

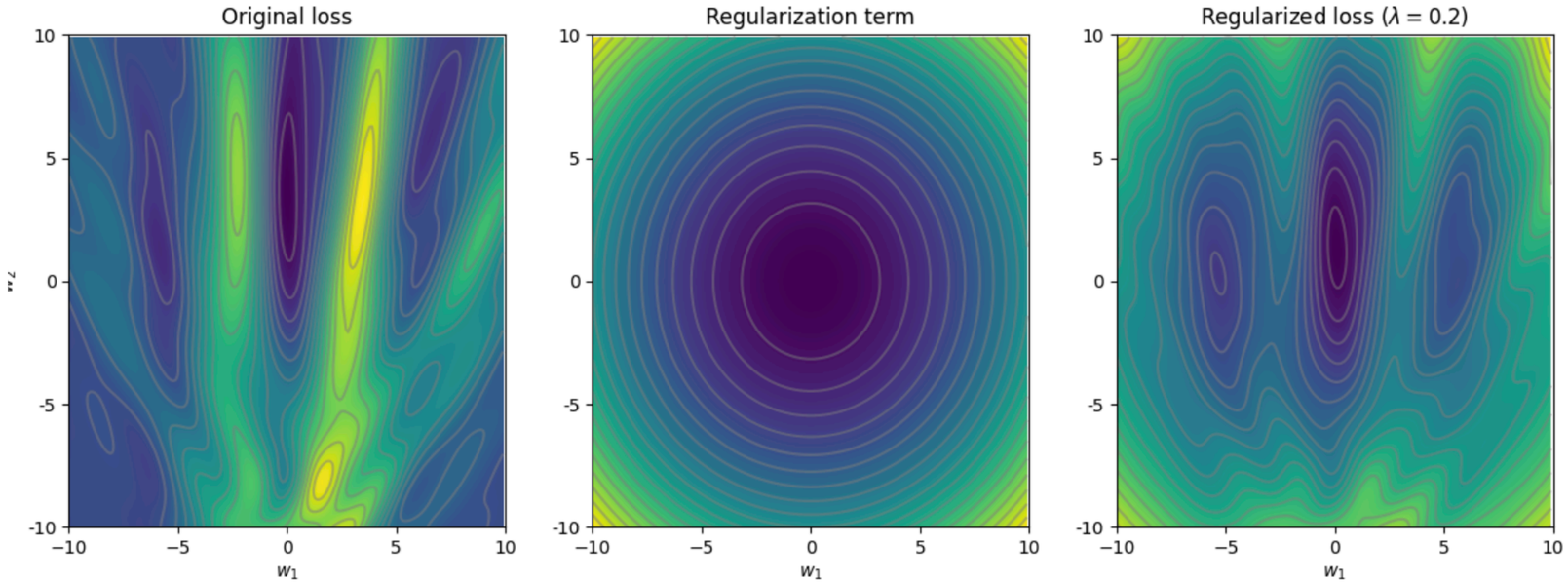
Weight Decay

Regularized loss = $L(\theta)$ + λ regularization term(θ)

Hyperparameter determines the importance of the regularization term

Weight Decay

Regularized loss = $L(\theta)$ + λ regularization term(θ)

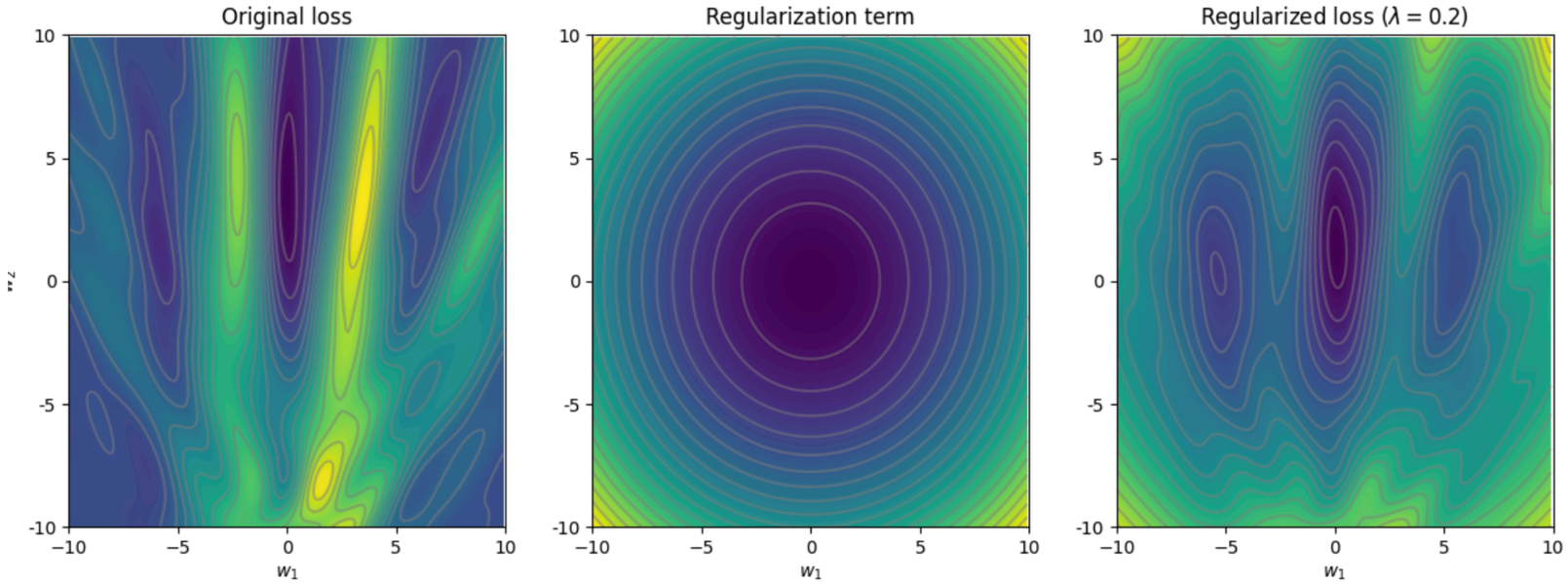


Which type of regularization term has been used and what is the effect?

Weight Decay

Regularized loss = $L(\theta)$ + λ regularization term(θ)

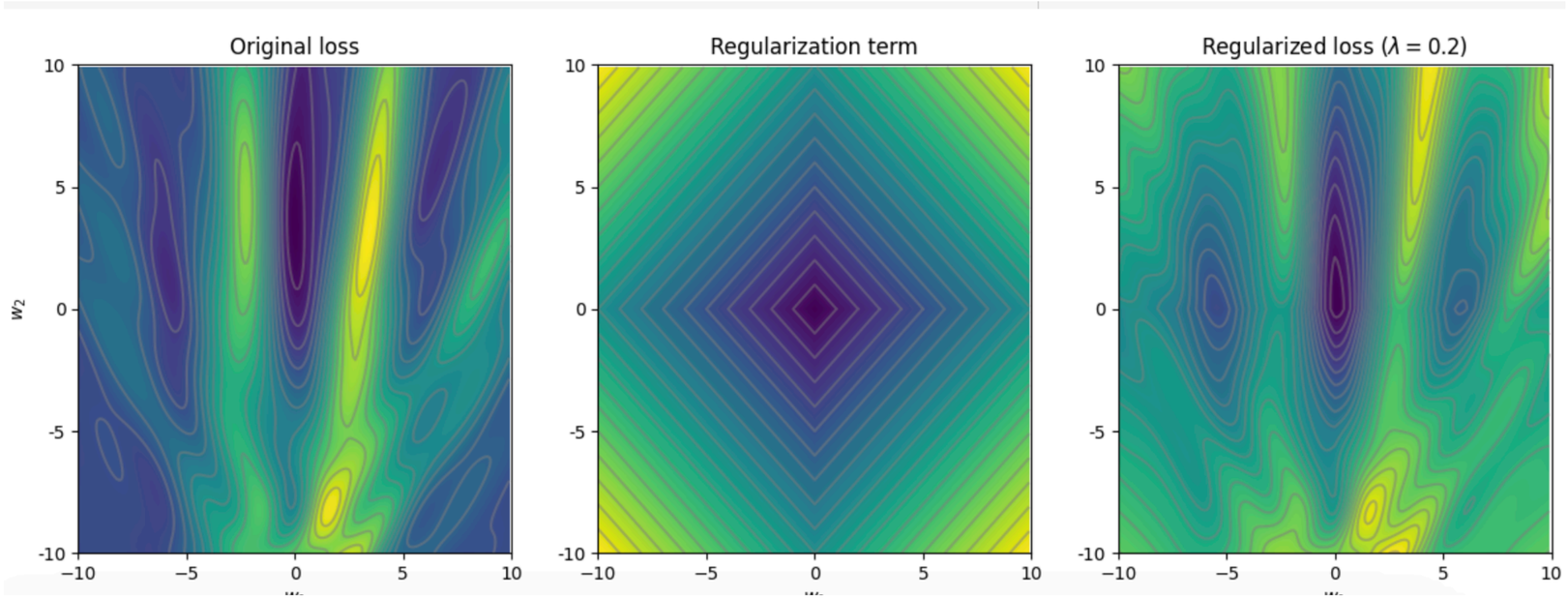
$$L \qquad + 0.2(w_1^2 + w_2^2)$$



Which type of regularization term has been used and what is the effect? **Ridge or L^2 norm penalty**

Weight Decay

Regularized loss = $L(\theta)$ + λ regularization term(θ)

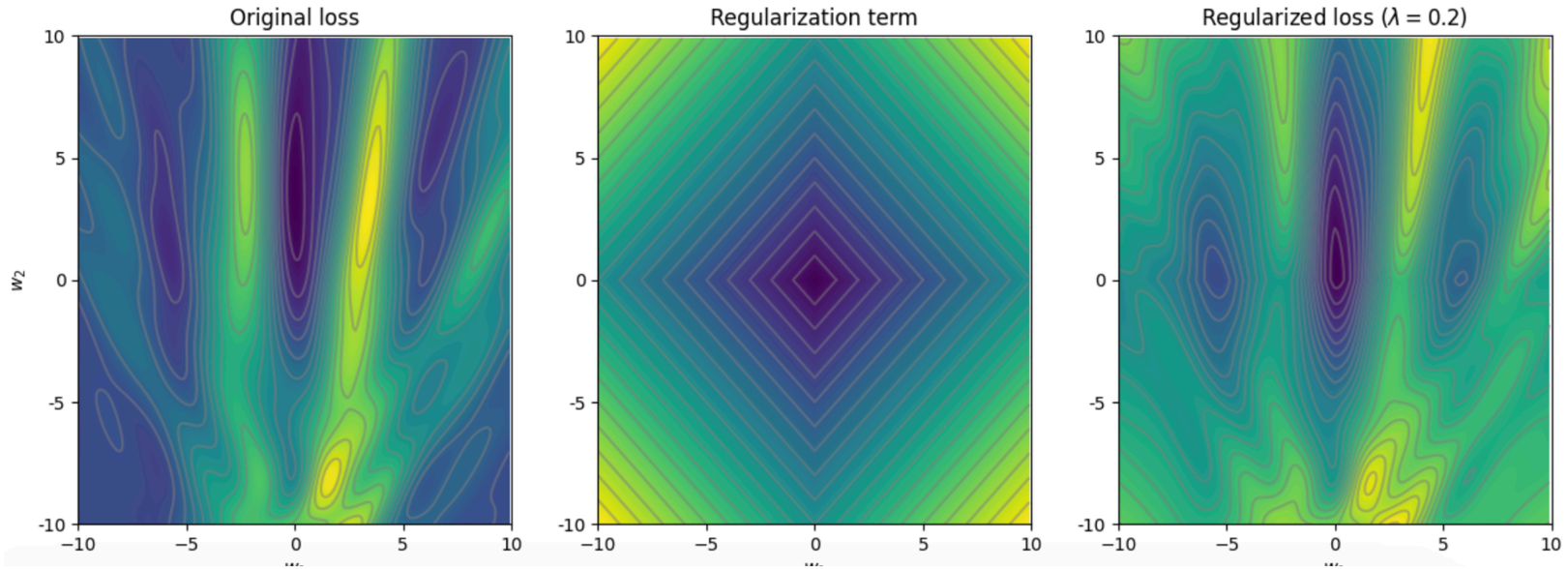


Which type of regularization term has been used and what is the effect?

Weight Decay

Regularized loss = $L(\theta)$ + λ regularization term(θ)

$$L \qquad + 0.2(|w_1| + |w_2|)$$



Which type of regularization term has been used and what is the effect? **Lasso or L^1 norm penalty**

Weight Decay: update

Regularized loss = $L(\theta)$ + λ regularization term(θ)

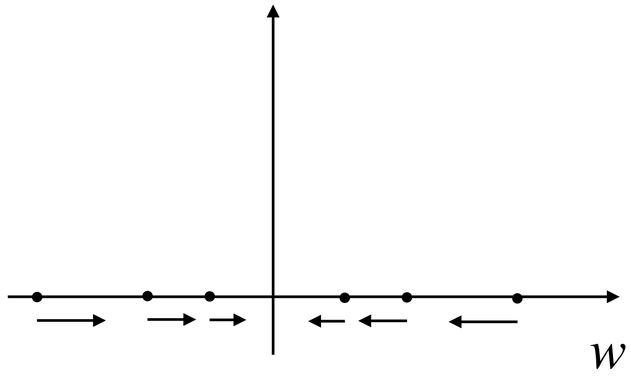
Update rule: $w \longleftarrow w - \alpha \frac{\partial L}{\partial w} - \alpha\lambda \frac{\partial \text{ regularization term}}{\partial w}$

Weight Decay: update

Regularized loss = $L(\theta)$ + λ regularization term(θ)

Update rule: $w \longleftarrow w - \alpha \frac{\partial L}{\partial w} - \alpha\lambda \frac{\partial \text{ regularization term}}{\partial w}$

1. Ridge or L^2 $\sum_i w_i^2$ $\quad w \longleftarrow w - \alpha \frac{\partial L}{\partial w} \boxed{- 2\alpha\lambda w}$



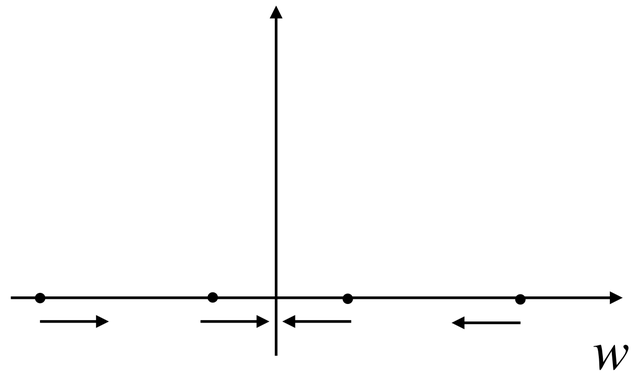
Weight Decay: update

Regularized loss = $L(\theta)$ + λ regularization term(θ)

Update rule: $w \longleftarrow w - \alpha \frac{\partial L}{\partial w} - \alpha\lambda \frac{\partial \text{ regularization term}}{\partial w}$

1. Ridge or L^2 $\sum_i w_i^2$
2. Lasso or L^1 $\sum_i |w_i|$

$$w \longleftarrow w - \alpha \frac{\partial L}{\partial w} \boxed{- \alpha\lambda \; sgn(w)}$$

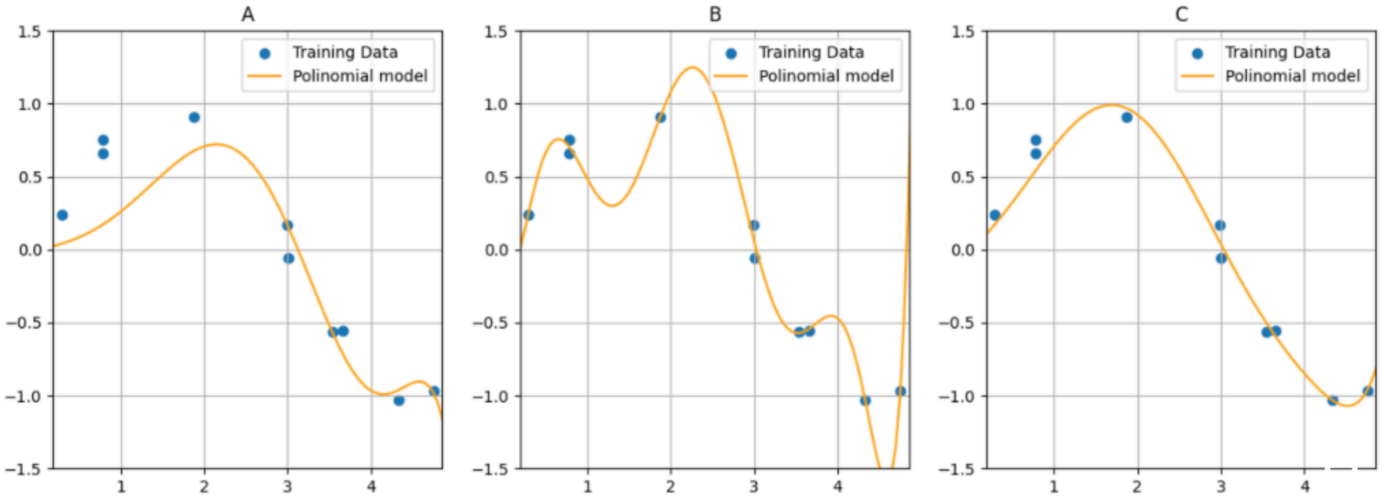


$$sgn(w) = \begin{cases} +1 & if \; w > 0 \\ -1 & if \; w < 0 \end{cases}$$

Weight Decay: coefficient λ

Polinomial model $f(x;\theta) = w_0 + w_1x^1 + w_2x^2 + w_3x^3 + \ldots + w_8x^8$ with 10 training data points

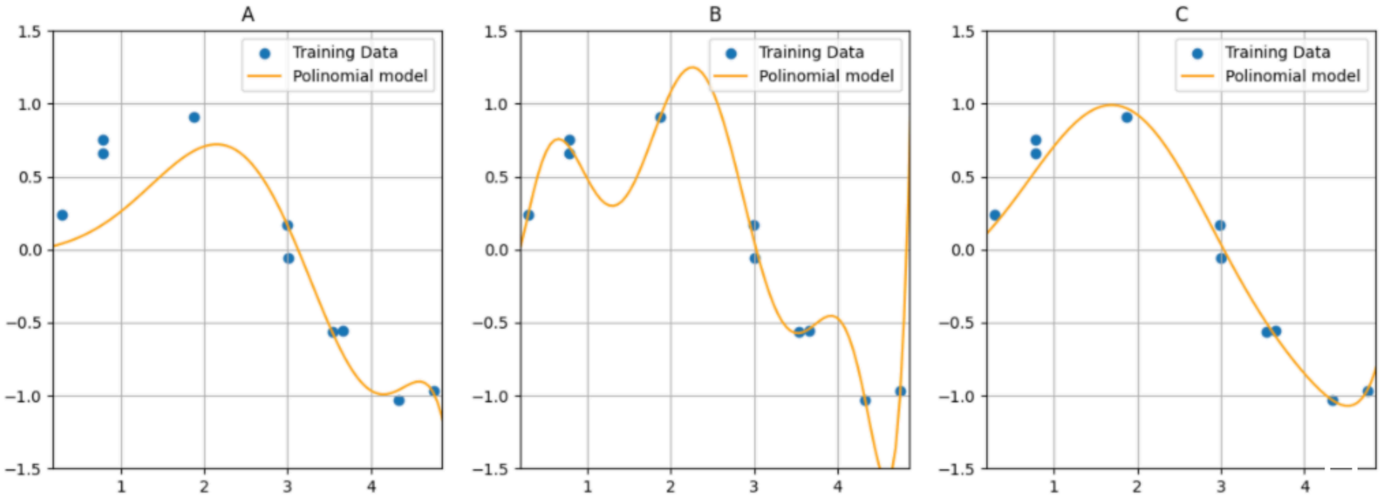
Regularized loss = $L_{SE}(w_0, w_1, \ldots, w_8) + \lambda \sum_{i=0}^{8} w_i^2$



Weight Decay: coefficient λ

Polinomial model $f(x;\theta) = w_0 + w_1x^1 + w_2x^2 + w_3x^3 + \ldots + w_8x^8$ with 10 training data points

Regularized loss = $L_{SE}(w_0, w_1, \ldots, w_8) + \lambda \sum_{i=0}^{8} w_i^2$

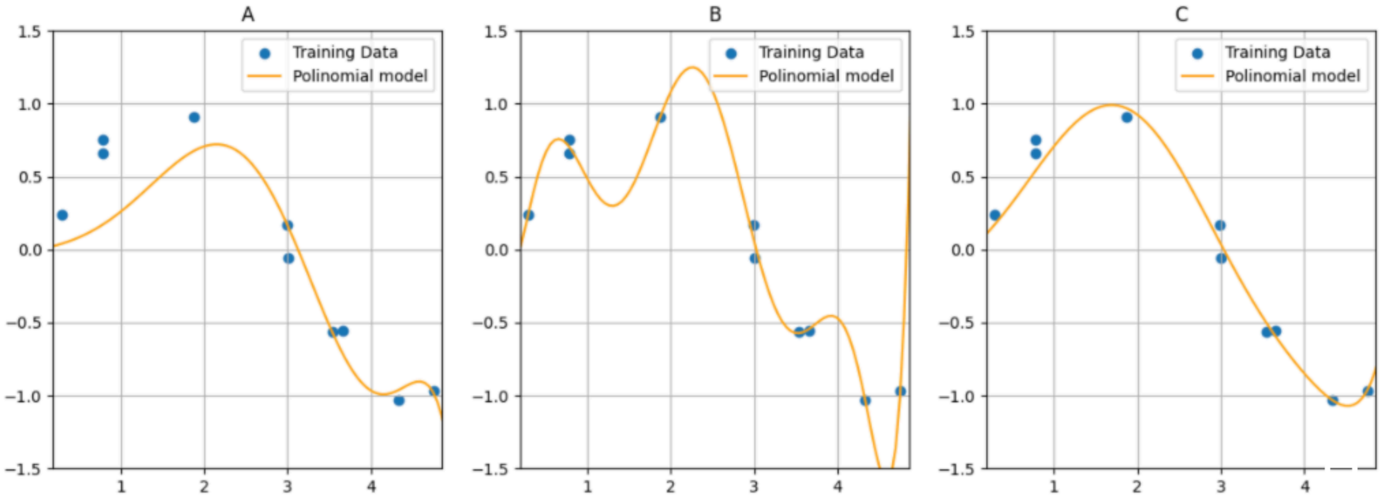


$\lambda = 0$?

Weight Decay: coefficient λ

Polinomial model $f(x;\theta) = w_0 + w_1x^1 + w_2x^2 + w_3x^3 + \ldots + w_8x^8$ with 10 training data points

Regularized loss = $L_{SE}(w_0, w_1, \ldots, w_8) + \lambda \sum_{i=0}^{8} w_i^2$



$\lambda = 0$?

$\lambda = 0$

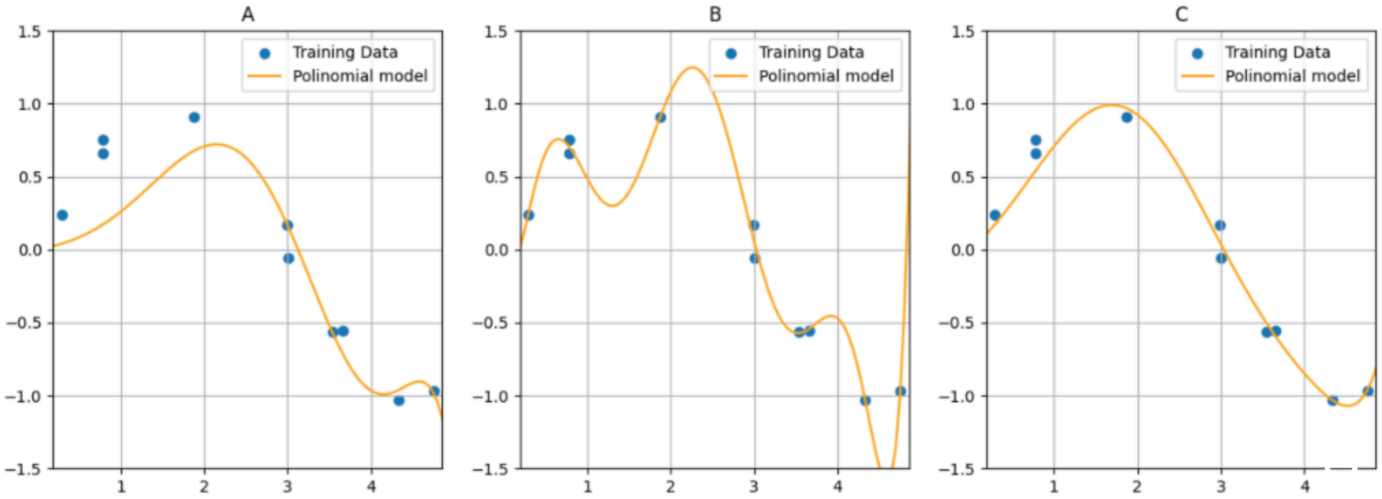
Overfitting

Loss = $L_{SE}(w_0, w_1, \ldots, w_8)$

Weight Decay: coefficient λ

Polinomial model $f(x;\theta) = w_0 + w_1x^1 + w_2x^2 + w_3x^3 + \ldots + w_8x^8$ with 10 training data points

Regularized loss = $L_{SE}(w_0, w_1, \ldots, w_8) + \lambda \sum_{i=0}^{8} w_i^2$



$\lambda = 0$

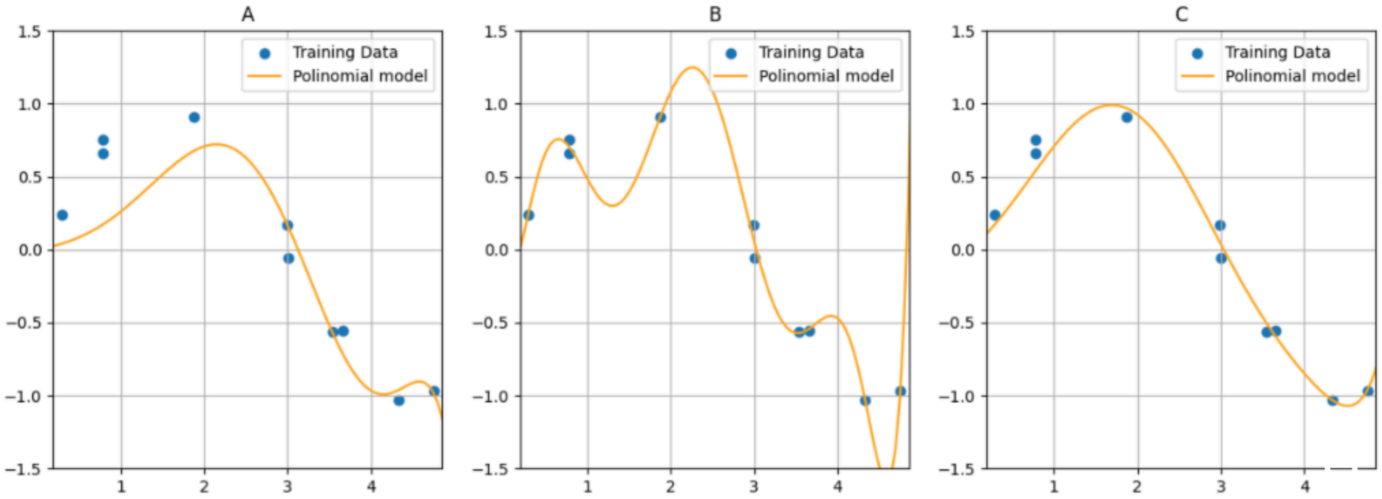
Overfitting

$\lambda = 0.3$ and $\lambda = 10$?

Weight Decay: coefficient λ

Polinomial model $f(x;\theta) = w_0 + w_1x^1 + w_2x^2 + w_3x^3 + \ldots + w_8x^8$ with 10 training data points

Regularized loss = $L_{SE}(w_0, w_1, \ldots, w_8) + \lambda \sum_{i=0}^{8} w_i^2$



$\lambda = 10$

Underfitting

$\lambda = 0$

Overfitting

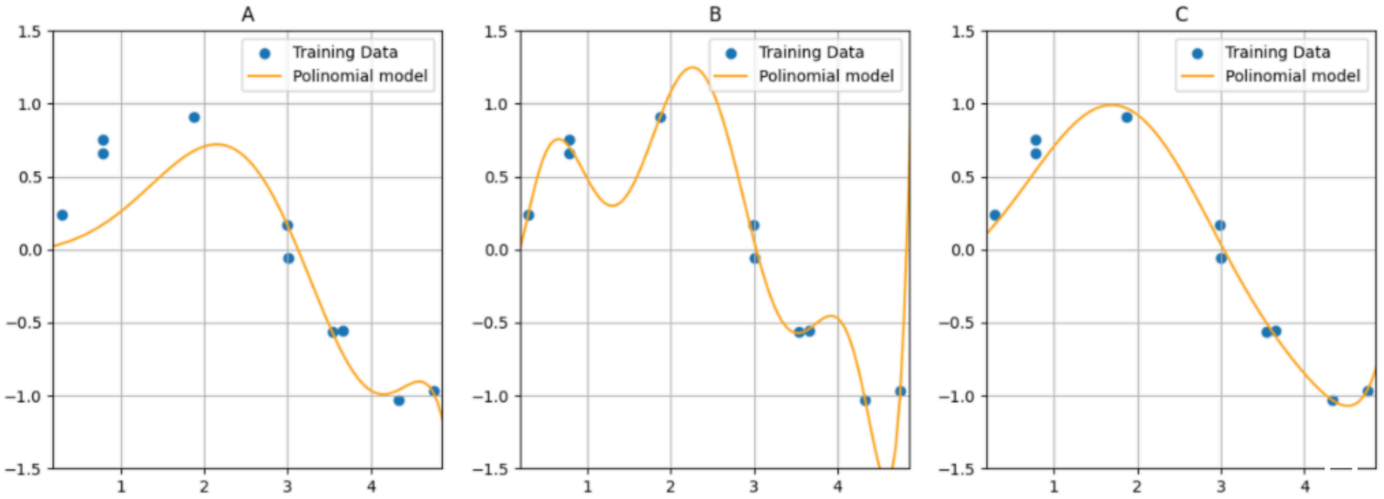
$\lambda = 0.3$ and $\lambda = 10$?

Loss = $L_{SE}(w_0, w_1, \ldots, w_8) + 10 \sum_{i=0}^{8} w_i^2$

Weight Decay: coefficient λ

Polinomial model $f(x;\theta) = w_0 + w_1x^1 + w_2x^2 + w_3x^3 + \ldots + w_8x^8$ with 10 training data points

Regularized loss = $L_{SE}(w_0, w_1, \ldots, w_8) + \lambda \sum_{i=0}^{8} w_i^2$



$\lambda = 10$
Underfitting

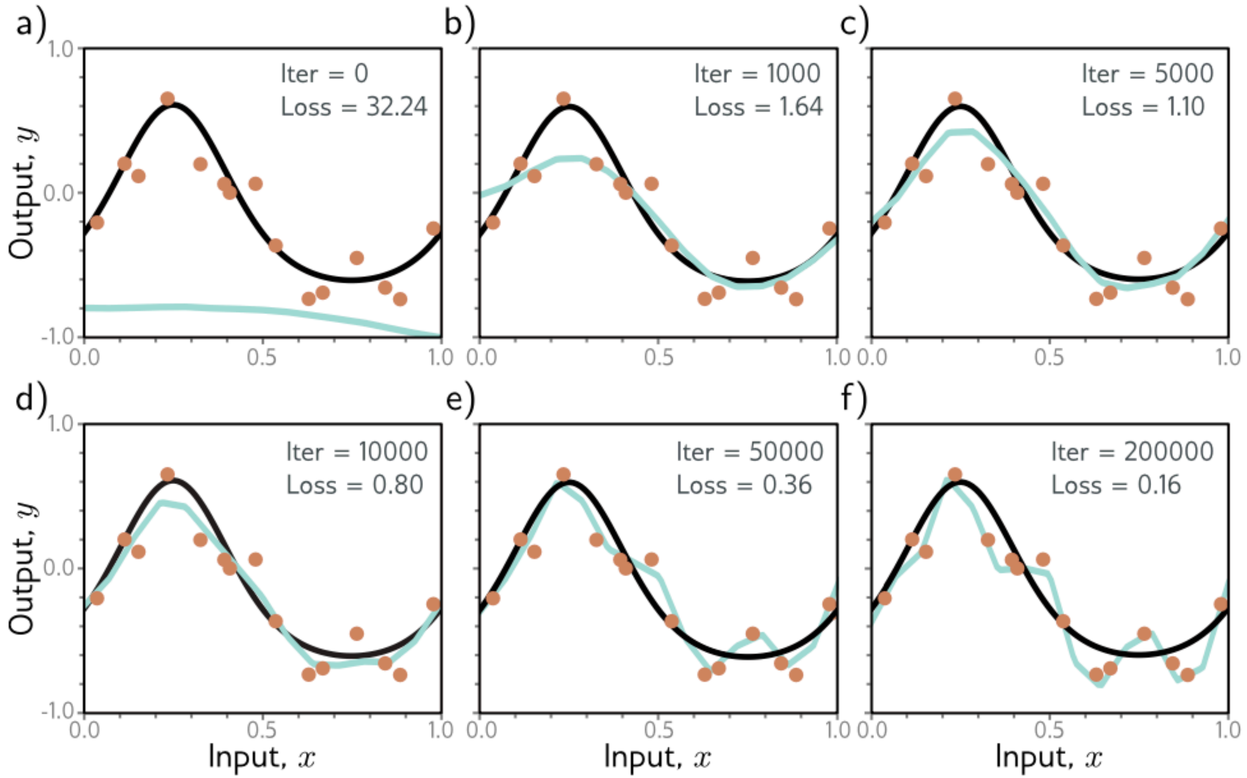
$\lambda = 0$
Overfitting

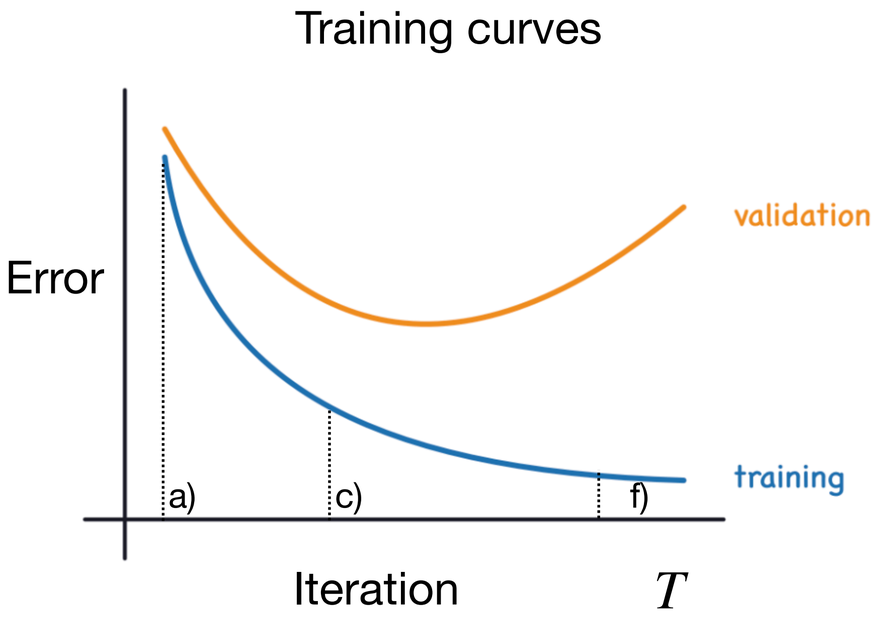
$\lambda = 0.3$
Proper fitting

$\lambda = 0.3$ and $\lambda = 10$?

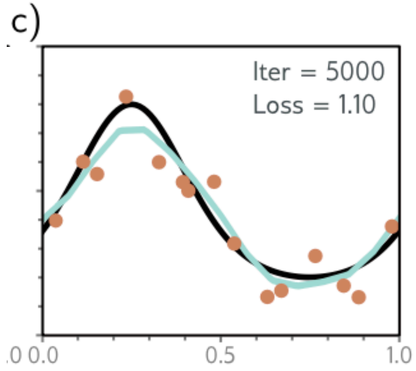
Loss = $L_{SE}(w_0, w_1, \ldots, w_8) + 0.3 \sum_{i=0}^{8} w_i^2$

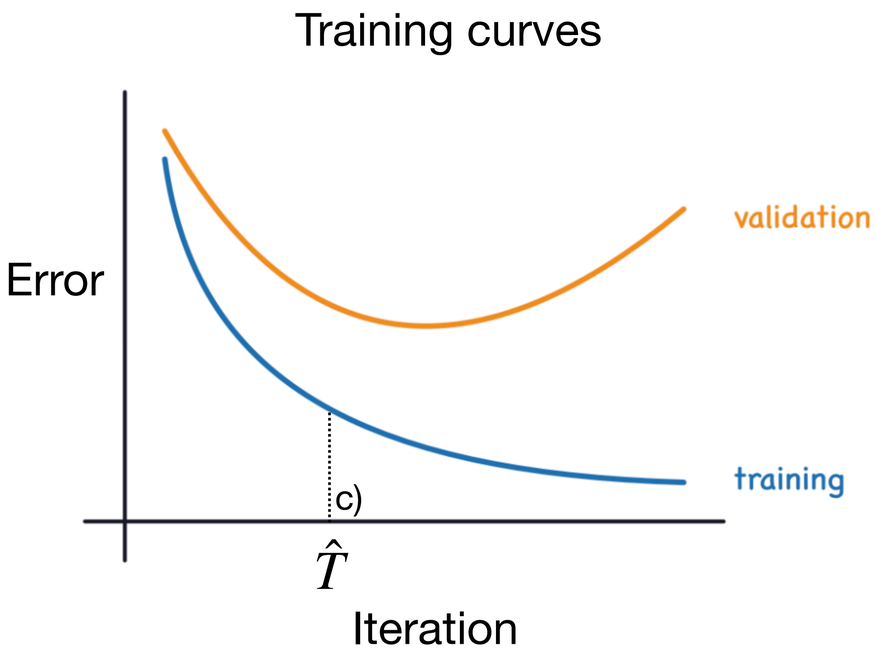
Early Stopping



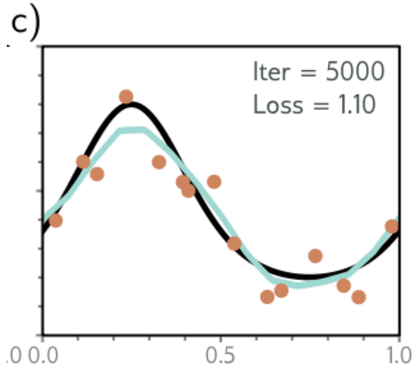


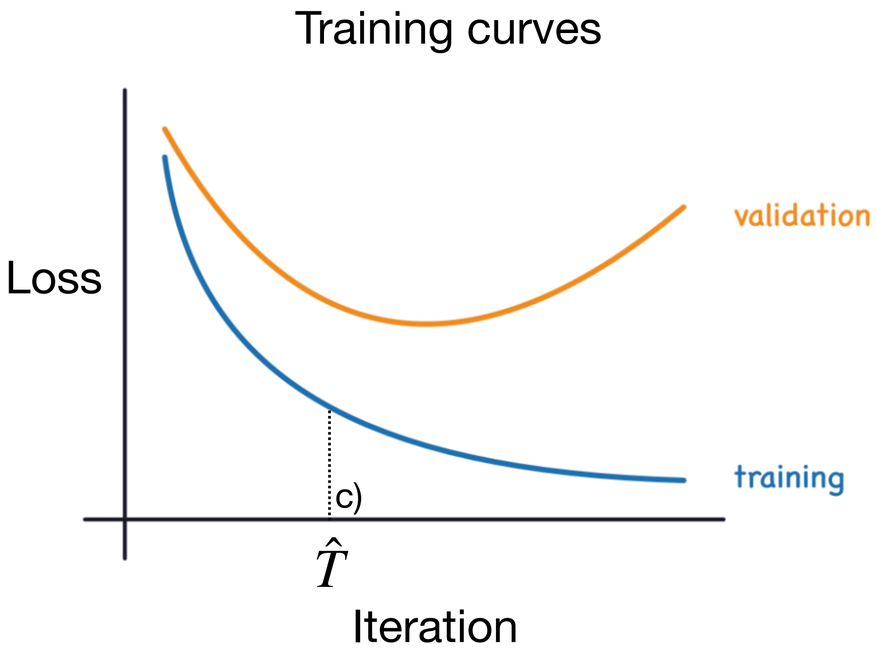
Early Stopping





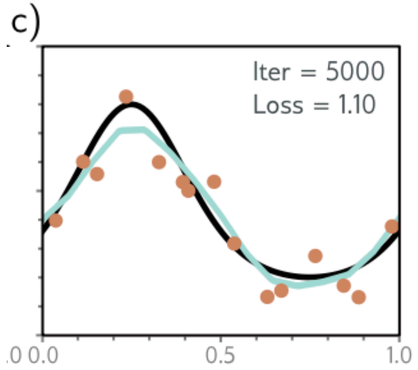
Early Stopping: true or false

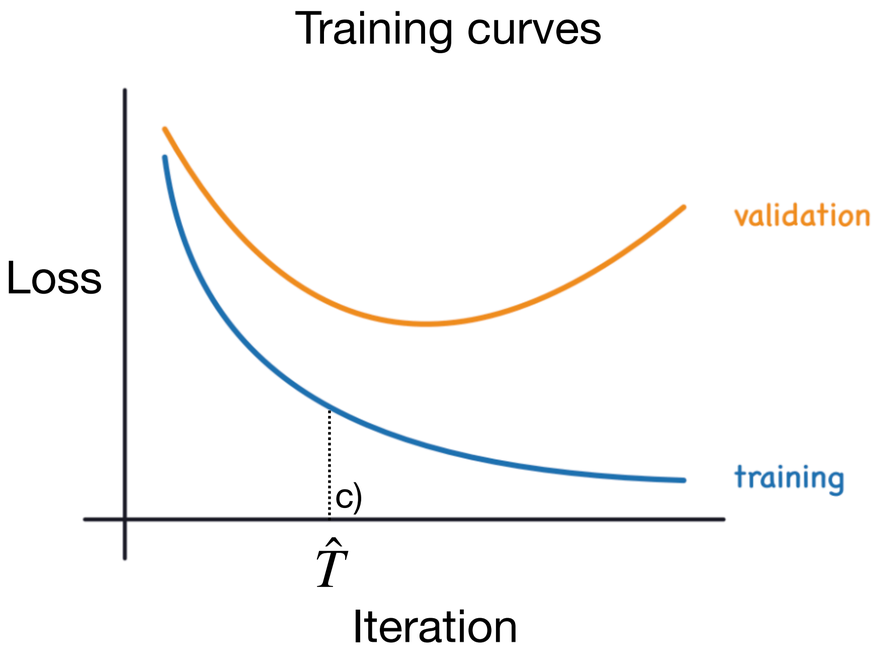




Early stopping introduces an additional hyperparameter: T the number of training steps after which learning is terminated. To find the optimal number of training steps $\hat{T}$, a neural network needs to be trained multiple times for different values of T

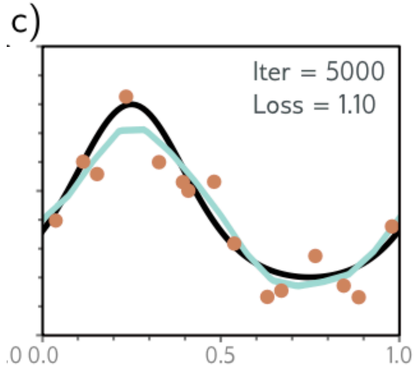
Early Stopping: true or false

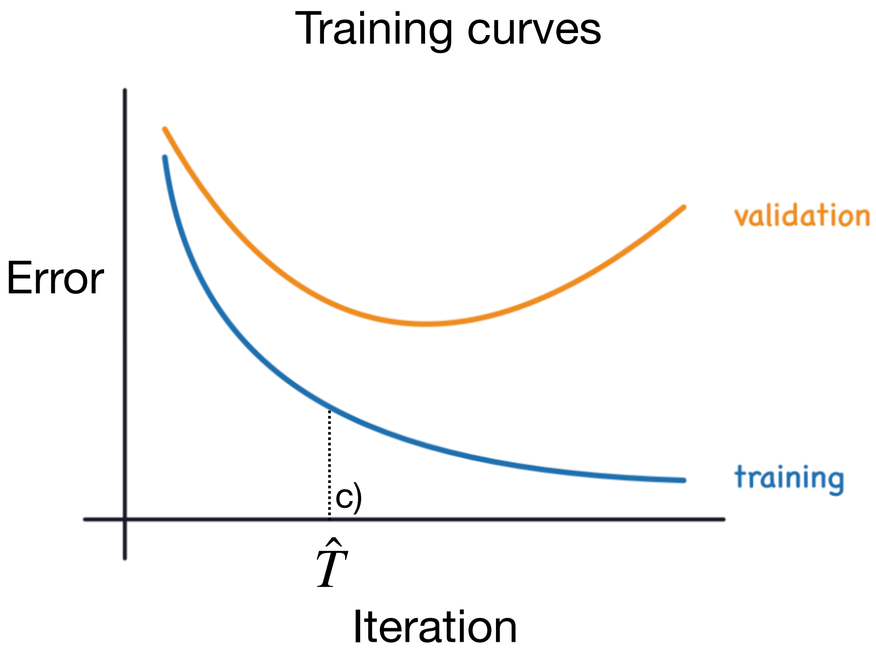




Early stopping introduces an additional hyperparameter: T the number of training steps after which learning is terminated. ~~To find the optimal number of training steps $\hat{T}$, a neural network needs to be trained multiple times for different values of T.~~

Early Stopping



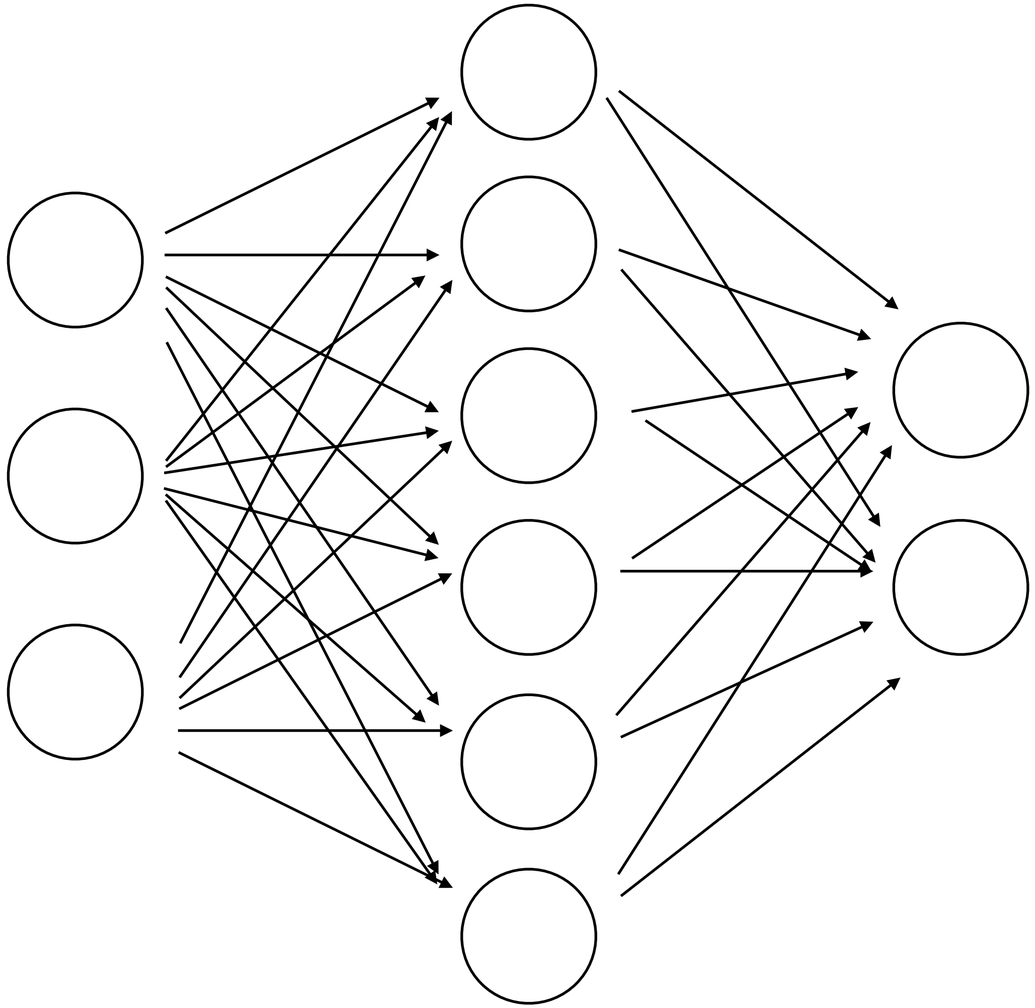


Patience value ⟶ number of epochs to wait after the last improvement in the validation error

Dropout

Mini - batch

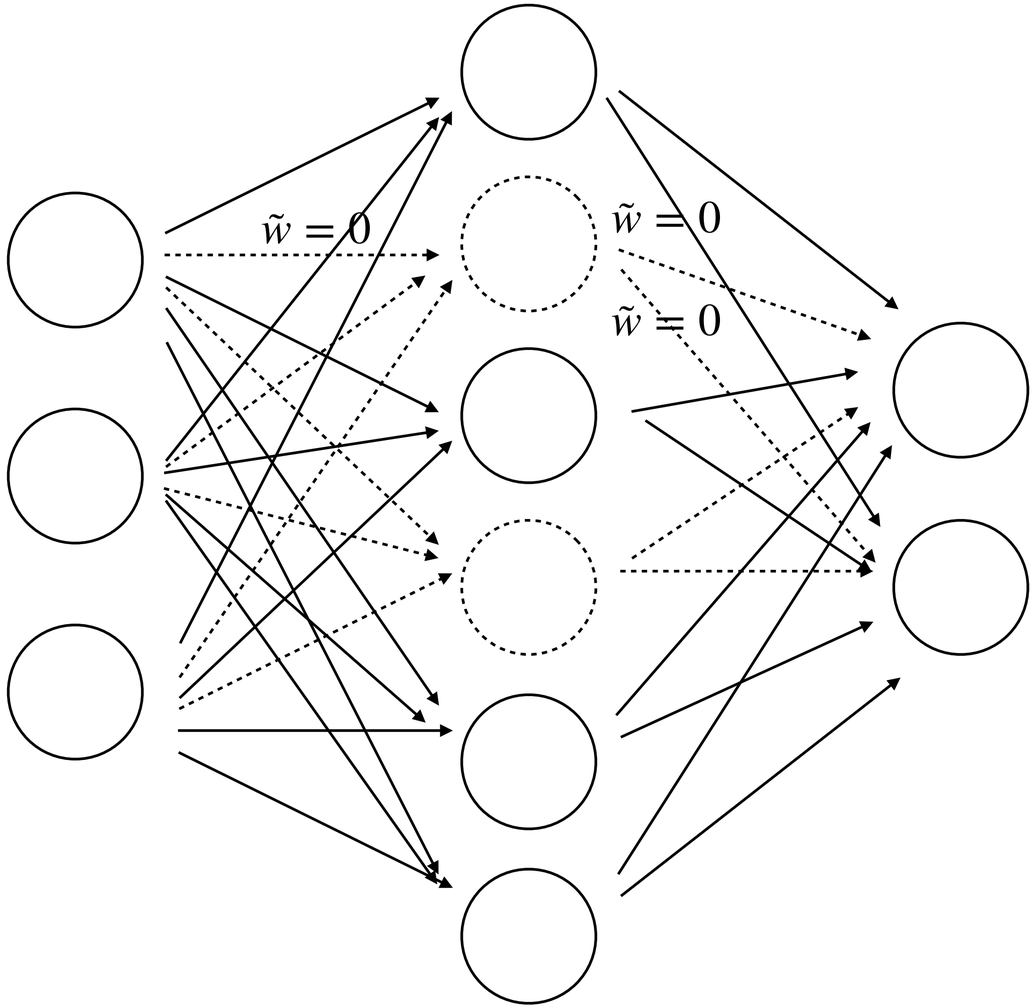
$x^{(1)}, x^{(2)}, \ldots x^{(m)}$



Dropout

Mini - batch

$x^{(1)}, x^{(2)}, \ldots x^{(m)}$

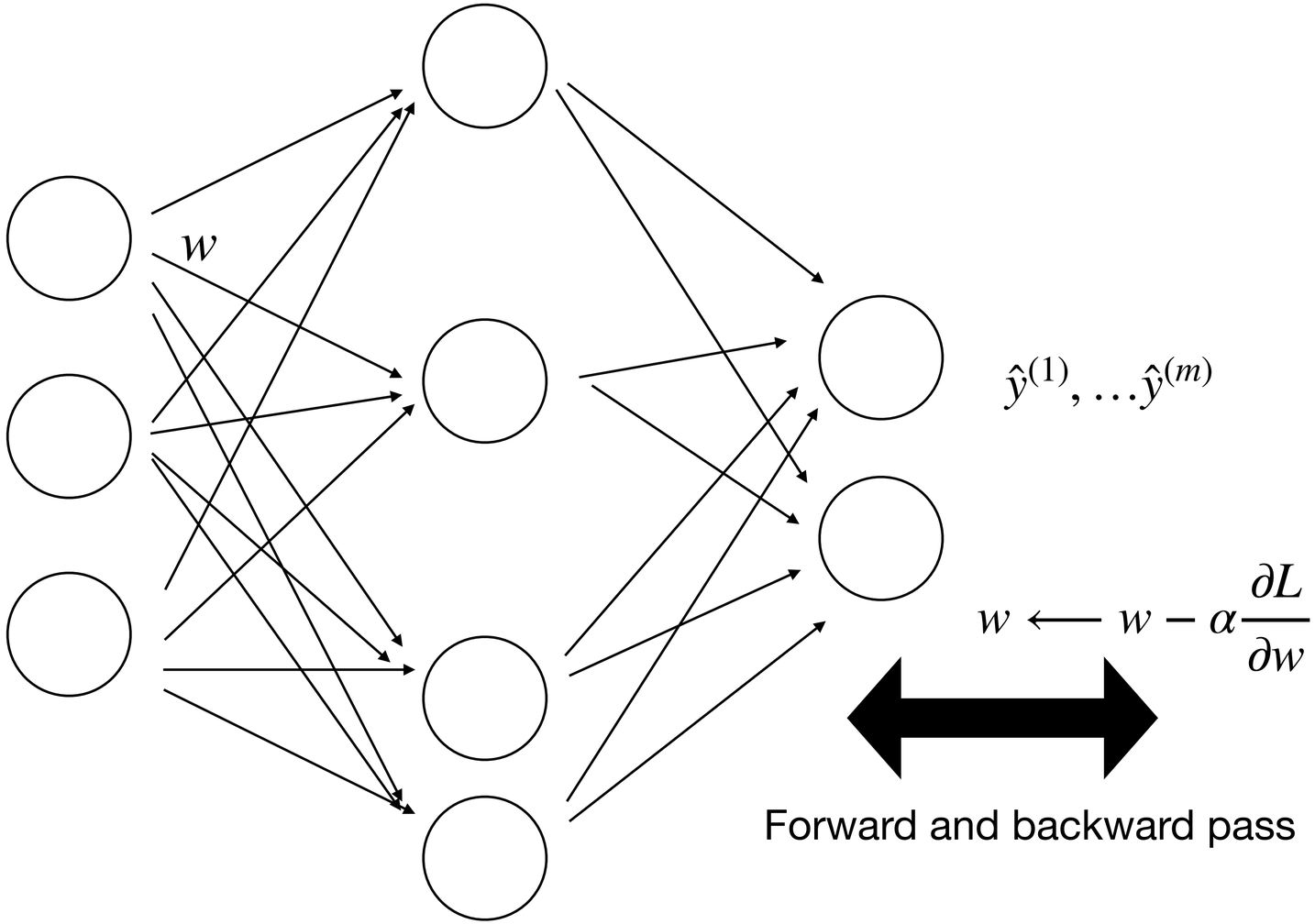


Before forward passing the mini-batch during training, switch off randomly some unit (with probability p)

Dropout

Mini - batch

$x^{(1)}, x^{(2)}, \ldots x^{(m)}$

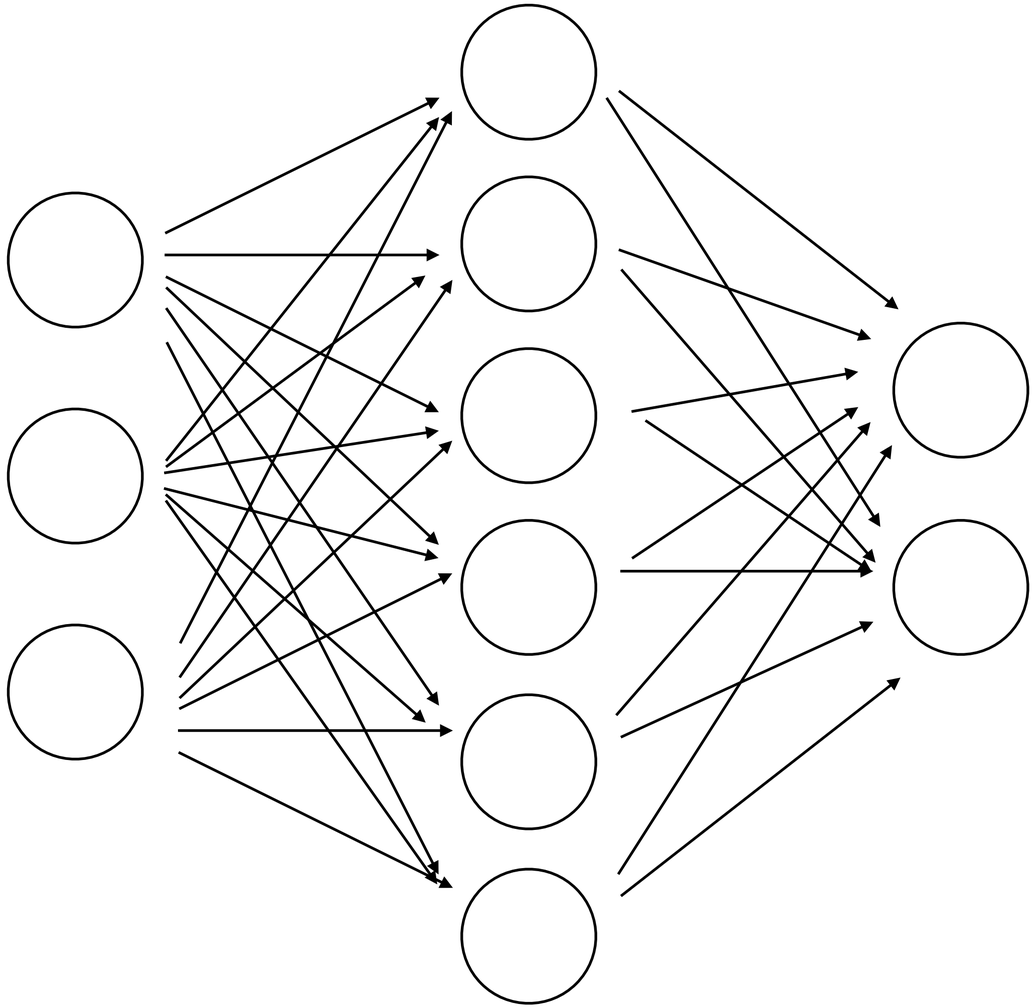


Before forward passing the mini-batch during training, switch off randomly some unit (with probability p)

Dropout

New Mini - batch

$x^{(m+1)}, \ldots, x^{(2m)}$

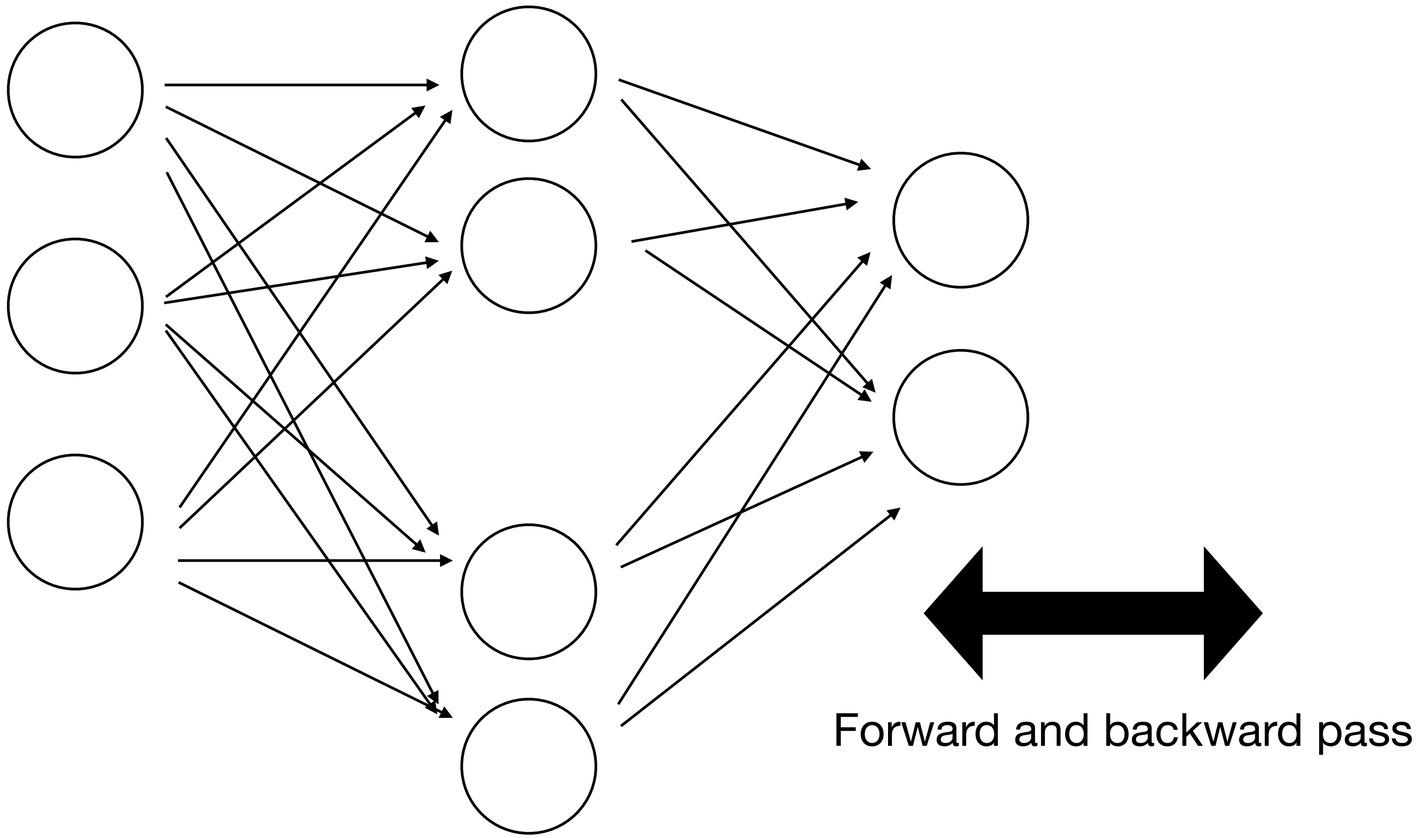


Before forward passing the mini-batch during training, switch off randomly some unit (with probability p)

Dropout

New Mini - batch

$x^{(m+1)}, \ldots, x^{(2m)}$

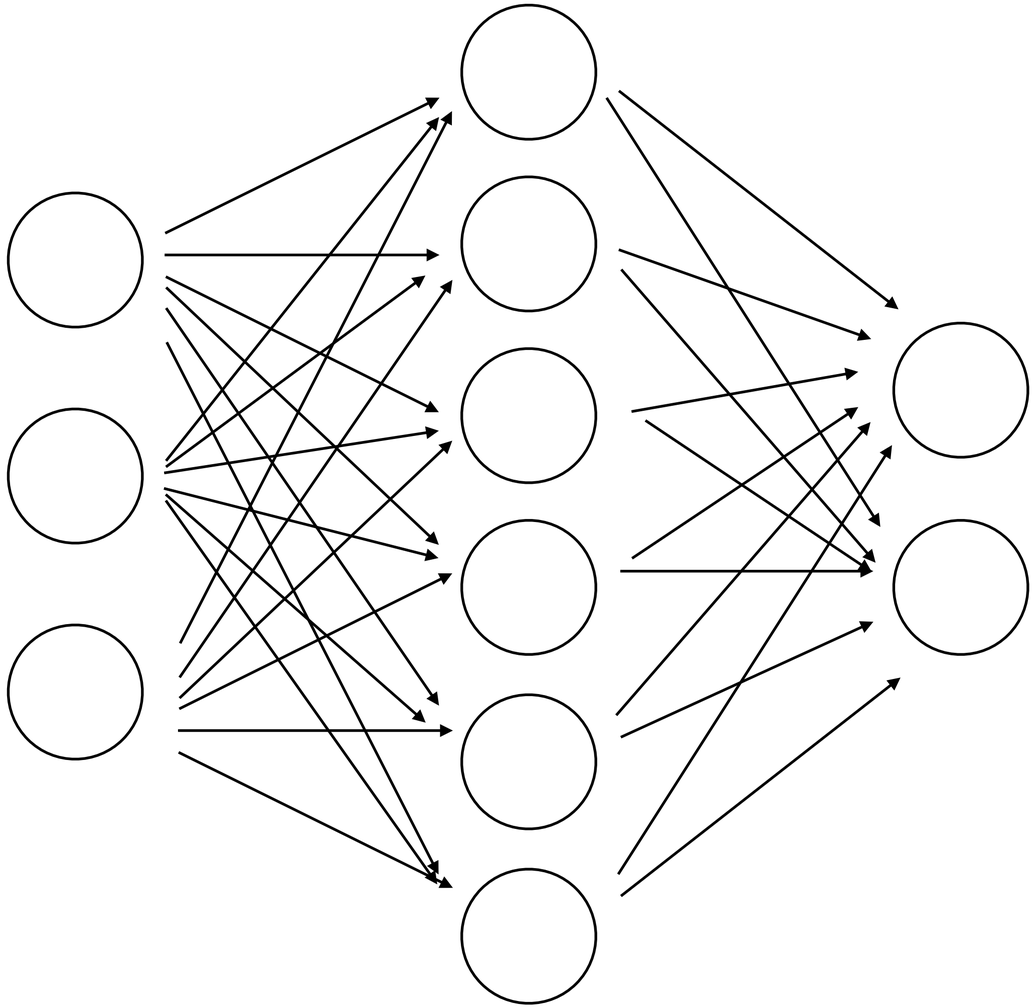


Forward and backward pass

Before forward passing the mini-batch during training, switch off randomly some unit (with probability p)

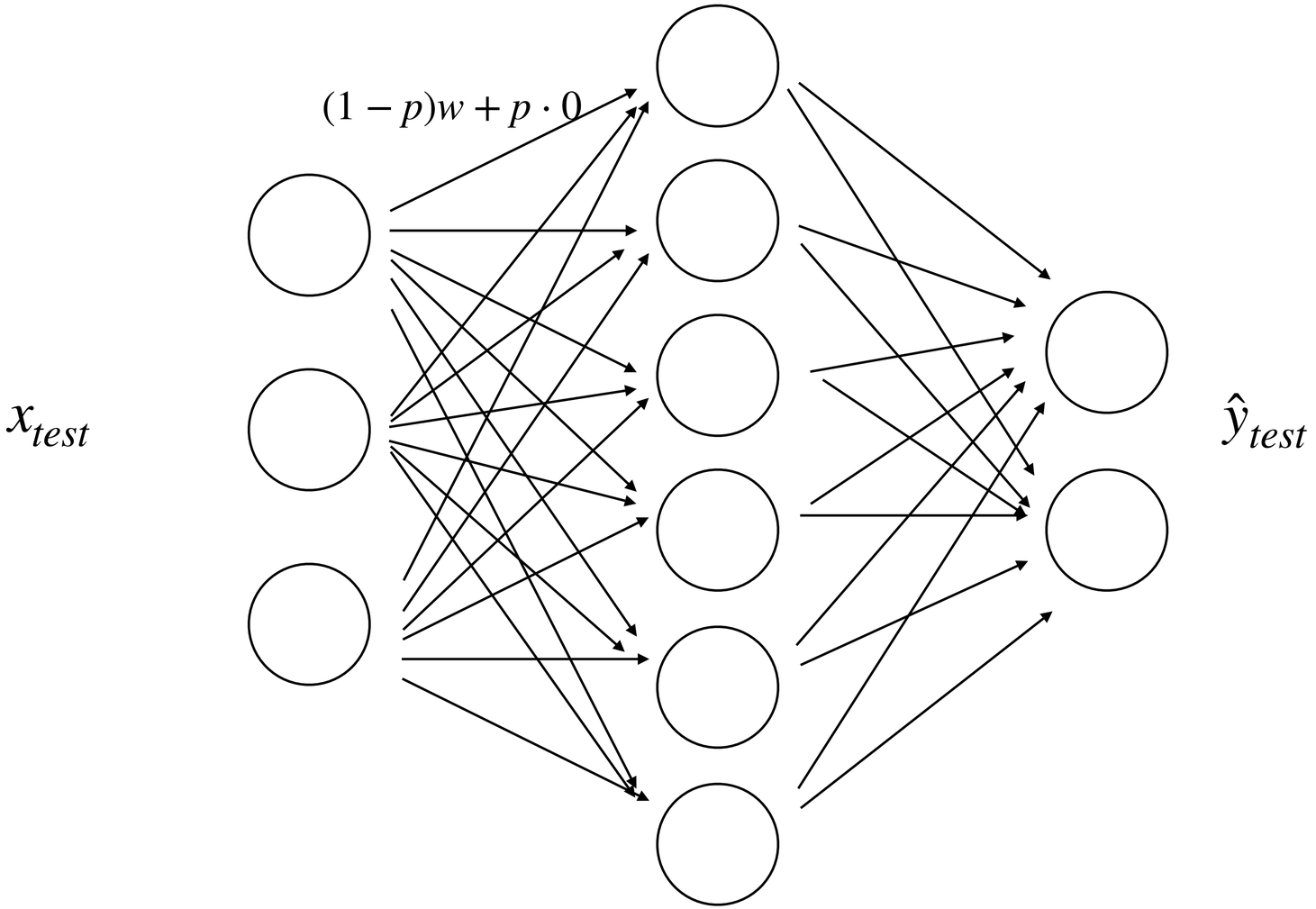
Dropout

x_{test}



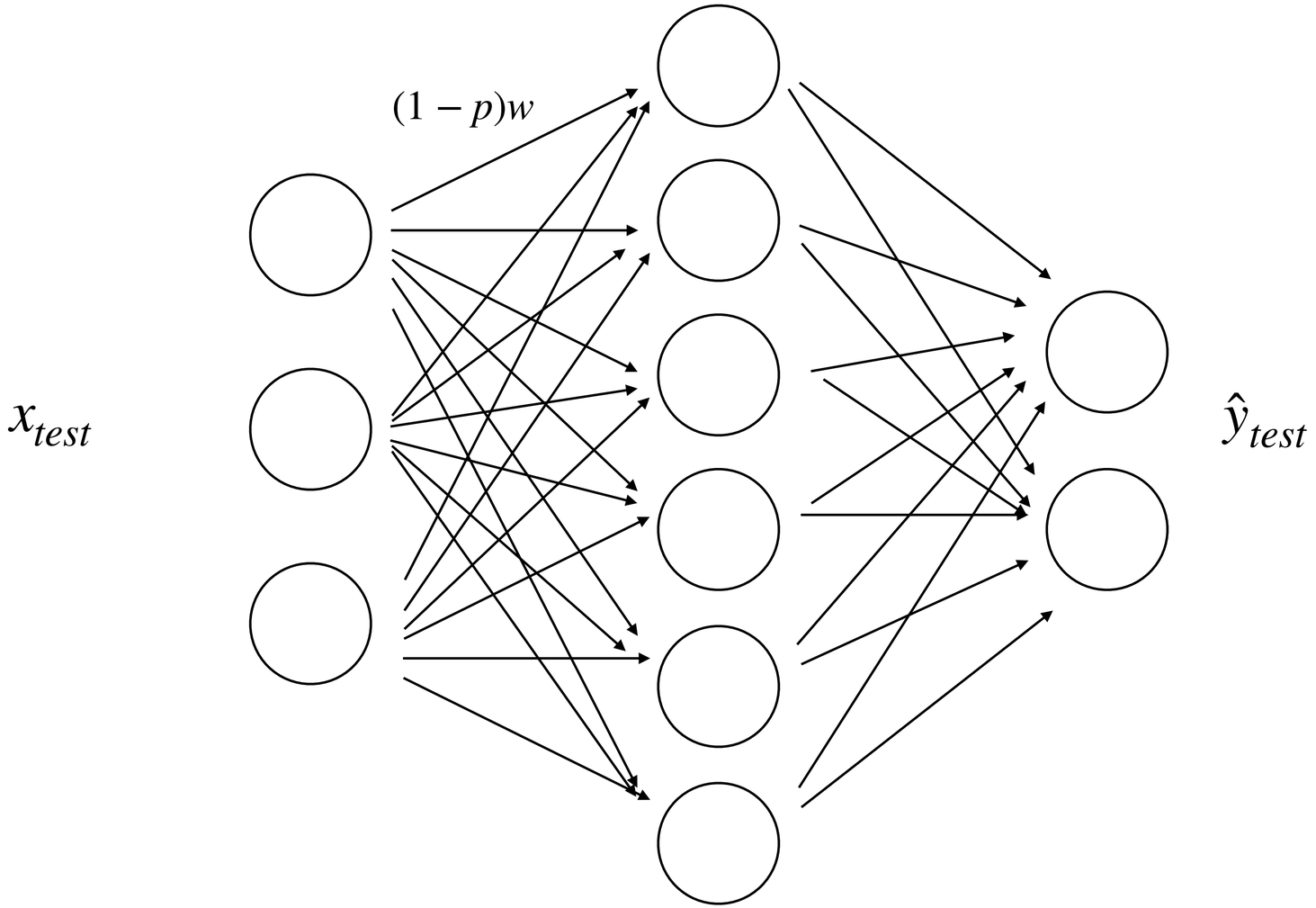
?

Dropout



Training several smaller neural networks to use at inference time an “average” network

Dropout



Each network might overfit the training data in different ways, averaging might help reducing overfitting

Questions

1. If your neural network seems to have high bias, what approach would you try to reduce it?

    a. Add regularization
    b. Increase the size of the network
    c. Get more training data
    d. None of the previous one

Questions

1. If your neural network seems to have high bias, what approach would you try to reduce it?

    a. Add regularization
    b. Increase the size of the network
    c. Get more training data
    d. None of the previous one

Questions

1. If your neural network seems to have high bias, what approach would you try to reduce it?

    a. Add regularization
    b. Increase the size of the network
    c. Get more training data
    d. None of the previous one

2. Can introducing noise in gradient descent as in stochastic gradient descent be considered an implicit regularization technique?

Questions

1. If your neural network seems to have high bias, what approach would you try to reduce it?

    a. Add regularization
    b. Increase the size of the network
    c. Get more training data
    d. None of the previous one

2. Can introducing noise in gradient descent as in stochastic gradient descent be considered an implicit regularization technique? **yes**

Questions

1. If your neural network seems to have high bias, what approach would you try to reduce it?

    a. Add regularization
    b. Increase the size of the network
    c. Get more training data
    d. None of the previous one

2. Can introducing noise in gradient descent as in stochastic gradient descent be considered an implicit regularization technique? yes

3. In which situations you might want to decrease the coefficient λ of the regularisation term?

Questions

1. If your neural network seems to have high bias, what approach would you try to reduce it?

    a. Add regularization
    b. Increase the size of the network
    c. Get more training data
    d. None of the previous one

2. Can introducing noise in gradient descent as in stochastic gradient descent be considered an implicit regularization technique? yes

3. In which situations you might want to decrease the coefficient λ of the regularisation term?
    - **Underfitting: if both training and test loss remain high, the model might underfit the training data.**
    - **Vainishing weights: check if the magnitude of your weights is too small.**

Questions

1. If your neural network seems to have high bias, what approach would you try to reduce it?

    a. Add regularization
    b. Increase the size of the network
    c. Get more training data
    d. None of the previous one

2. Can introducing noise in gradient descent as in stochastic gradient descent be considered an implicit regularization technique? yes

3. In which situations you might want to decrease the coefficient λ of the regularisation term?
    - Underfitting: if both training and test loss remain high, the model might underfit the training data.
    - Vainishing weights: check if the magnitude of your weights is too small.

4. Do you need regularization when you have infinite amount of training data?

Questions

1. If your neural network seems to have high bias, what approach would you try to reduce it?

    a. Add regularization
    b. Increase the size of the network
    c. Get more training data
    d. None of the previous one

2. Can introducing noise in gradient descent as in stochastic gradient descent be considered an implicit regularization technique? yes

3. In which situations you might want to decrease the coefficient λ of the regularisation term?
    - Underfitting: if both training and test loss remain high, the model might underfit the training data.
    - Vainishing weights: check if the magnitude of your weights is too small.

4. Do you need regularization when you have infinite amount of training data? **No**

Summary

Topics

- Generalization and errors
- Augmentation
- Regularization techniques

Reading material

- *Understanding Deep Learning* - Chapter 8 (excluded 8.2.2, 8.4), Chapter 9.1 (excluded 9.1.1), 9.3.1, 9.3.3, 9.3.8
- Neural Networks and Deep Learning: *Chapter 3: Regularization*