# Machine Learning

## Linear Regression 2

Hiroshi Shimodaira and Hao Tang

March 2026

*Ver. 1.0.1*

# Topics

- Linear regression with feature transformation
- Polynomial regression
- Overfitting in linear regression
- Linear regression with regularisation
- Ridge regression
- Kernel ridge regression

# Augmented vector representation

$$y = \boldsymbol{w}^\top \boldsymbol{x} + b = \begin{bmatrix} \boldsymbol{w}^\top & b \end{bmatrix} \begin{bmatrix} \boldsymbol{x} \\ 1 \end{bmatrix} = \begin{bmatrix} \boldsymbol{w} \\ b \end{bmatrix}^\top \begin{bmatrix} \boldsymbol{x} \\ 1 \end{bmatrix} = \boldsymbol{w}'^\top \boldsymbol{x}' \tag{1}$$

- Fitting $f(\boldsymbol{x}) = \boldsymbol{w}^\top \boldsymbol{x} + b$ is equivalent to appending 1 to $\boldsymbol{x}$ and fitting $f(\boldsymbol{x}') = \boldsymbol{w}'^\top \boldsymbol{x}'$.
- The 1 can be seen as a feature independent of the input.

# Feature functions

- Suppose we have a data point $\boldsymbol{x} = \begin{bmatrix} x_1 & x_2 & x_3 \end{bmatrix}^\top$.
- The data point after appending 1 becomes

$$\begin{bmatrix} 1 & x_1 & x_2 & x_3 \end{bmatrix}^\top \tag{2}$$

- The data point after appending 1 and quadratic terms becomes

$$\phi(\boldsymbol{x}) = \begin{bmatrix} 1 & x_1 & x_2 & x_3 & x_1x_2 & x_2x_3 & x_1x_3 & x_1^2 & x_2^2 & x_3^2 \end{bmatrix}^\top \tag{3}$$

- We call $\phi(\,)$ a *feature function*.
- Linear regression with polynomial feature transformation is called *polynomial regression*.

# Linear regression with feature transformation

- In general, $\phi : \mathbb{R}^d \mapsto \mathbb{R}^D$ can be any function.
- Instead of $f(\boldsymbol{x}) = \boldsymbol{w}^\top \boldsymbol{x} + b$, we now have $f(\boldsymbol{x}) = \boldsymbol{w}^\top \phi(\boldsymbol{x})$.
- Instead of $\mathbf{X} = \underbrace{\begin{bmatrix} \dot{\boldsymbol{x}}_1^\top \\ \vdots \\ \dot{\boldsymbol{x}}_N^\top \end{bmatrix}}_{N \times d}$, we have $\boldsymbol{\Phi} = \underbrace{\begin{bmatrix} \phi(\boldsymbol{x}_1)^\top \\ \vdots \\ \phi(\boldsymbol{x}_N)^\top \end{bmatrix}}_{N \times D}$
- The mean-squared error can be written as

$$\text{MSE} = \frac{1}{N} \sum_{i=1}^{N} (\boldsymbol{w}^\top \phi(\boldsymbol{x}_i) - y_i)^2 \tag{4}$$

$$= \frac{1}{N} \|\boldsymbol{\Phi} \boldsymbol{w} - \boldsymbol{y}\|_2^2. \tag{5}$$

# Linear regression with feature transformation (*cont.*)

- The cost function can be written compactly as

$$L = \|\boldsymbol{\Phi}\boldsymbol{w} - \boldsymbol{y}\|_2^2 \tag{6}$$

$$= (\boldsymbol{\Phi}\boldsymbol{w} - \boldsymbol{y})^\top (\boldsymbol{\Phi}\boldsymbol{w} - \boldsymbol{y}) \tag{7}$$

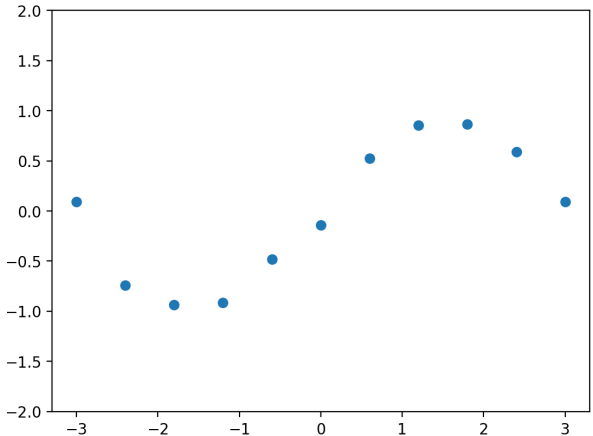
$$= \boldsymbol{w}^\top \boldsymbol{\Phi}^\top \boldsymbol{\Phi}\boldsymbol{w} - 2\boldsymbol{w}^\top \boldsymbol{\Phi}^\top \boldsymbol{y} + \boldsymbol{y}^\top \boldsymbol{y}. \tag{8}$$

- Solving the optimal solution gives

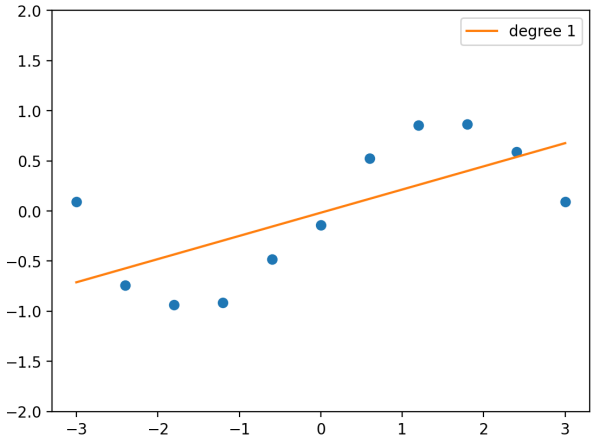
$$\frac{\partial L}{\partial \boldsymbol{w}}^\top = (\boldsymbol{\Phi}^\top \boldsymbol{\Phi} + (\boldsymbol{\Phi}^\top \boldsymbol{\Phi})^\top)\boldsymbol{w} - 2\boldsymbol{\Phi}^\top \boldsymbol{y} = \boldsymbol{0}. \tag{9}$$

$$\implies \boldsymbol{w} = (\boldsymbol{\Phi}^\top \boldsymbol{\Phi})^{-1} \boldsymbol{\Phi}^\top \boldsymbol{y}. \tag{10}$$

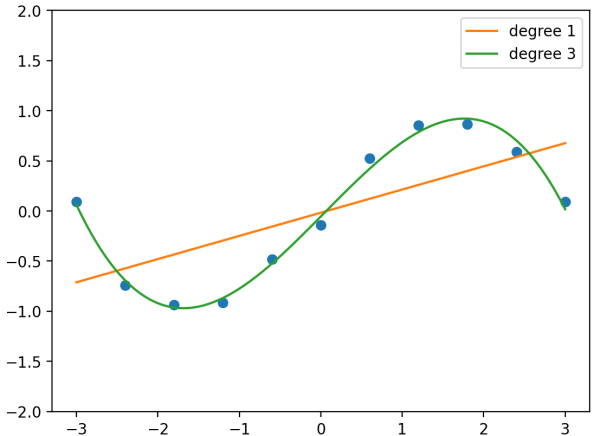
# Examples



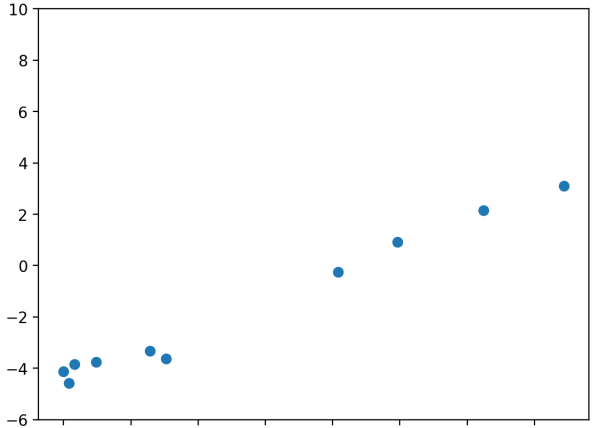
# Examples



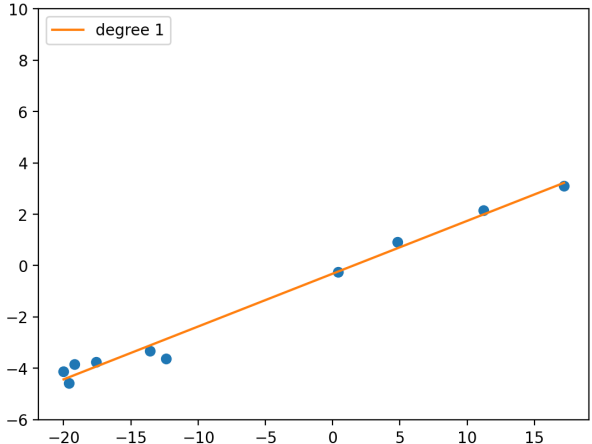
# Examples



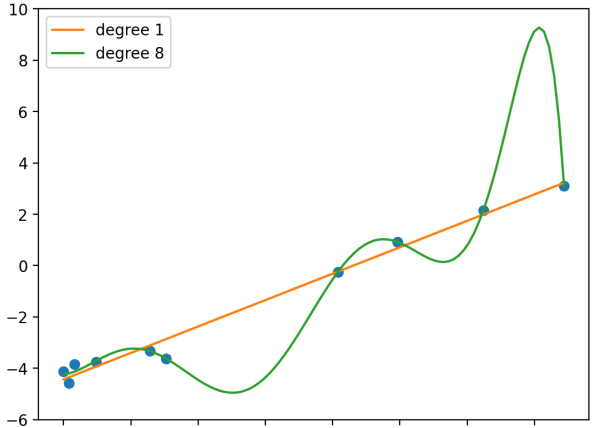
# Examples



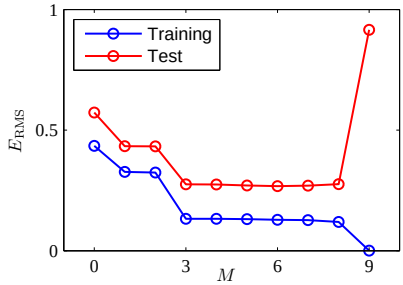
# Examples



# Examples



# Overfitting in linear regression




Credit: C. Bishop, PRML, Figure 1.5

- See slides on generalisation

# Linear regression with regularisation

- Optimisation problem without regularisation

$$\min_{\boldsymbol{w}} \frac{1}{N} \sum_{i=1}^{N} (\boldsymbol{w}^\top \phi(\boldsymbol{x}_i) - y_i)^2 \tag{11}$$

- Optimisation problem with regularisation

$$\min_{\boldsymbol{w}} \frac{1}{N} \sum_{i=1}^{N} (\boldsymbol{w}^\top \phi(\boldsymbol{x}_i) - y_i)^2 + \lambda \|\boldsymbol{w}\|^2, \quad \lambda > 0 \tag{12}$$

This is called '*ridge regression*'.

The second term is an $L^2$-regulariser.

# Ridge regression: training

$$L = \frac{1}{N}\sum_{i=1}^{N}(\boldsymbol{w}^\top\phi(\boldsymbol{x}_i) - y_i)^2 + \lambda\|\boldsymbol{w}\|^2 \tag{13}$$

$$= \frac{1}{N}\,(\boldsymbol{\Phi}\boldsymbol{w} - \boldsymbol{y})^\top(\boldsymbol{\Phi}\boldsymbol{w} - \boldsymbol{y}) + \lambda\boldsymbol{w}^\top\boldsymbol{w} \tag{14}$$

$$= \frac{1}{N}\,\boldsymbol{w}^\top\left(\boldsymbol{\Phi}^\top\boldsymbol{\Phi} + N\lambda\mathbf{I}\right)\boldsymbol{w} - 2\boldsymbol{w}^\top\boldsymbol{\Phi}^\top\boldsymbol{y} + \boldsymbol{y}^\top\boldsymbol{y} \tag{15}$$

$$\frac{\partial L}{\partial \boldsymbol{w}} = \frac{2}{N}\,(\boldsymbol{\Phi}^\top\boldsymbol{\Phi} + N\lambda\mathbf{I})\boldsymbol{w} - \boldsymbol{\Phi}^\top\boldsymbol{y}) = \mathbf{0} \tag{16}$$

$$\implies \quad (\boldsymbol{\Phi}^\top\boldsymbol{\Phi} + N\lambda\mathbf{I})\boldsymbol{w} = \boldsymbol{\Phi}^\top\boldsymbol{y} \tag{17}$$

$$\boldsymbol{w} = (\boldsymbol{\Phi}^\top\boldsymbol{\Phi} + N\lambda\mathbf{I})^{-1}\boldsymbol{\Phi}^\top\boldsymbol{y} \tag{18}$$

# Kernel ridge regression

Using the previous result, the prediction of $\hat{y}$ given $\boldsymbol{x}$ is expressed as

$$\hat{y} = \boldsymbol{w}^\top \phi(\boldsymbol{x}) \;=\; \boldsymbol{y}^\top \boldsymbol{\Phi}(\boldsymbol{\Phi}^\top \boldsymbol{\Phi} + N\lambda \mathbf{I})^{-1} \phi(\boldsymbol{x}) \tag{19}$$

Using the "push-through identity" $^{(\dagger)}$ yields:

$$= \boldsymbol{y}^\top (\boldsymbol{\Phi}\boldsymbol{\Phi}^\top + N\lambda \mathbf{I})^{-1} \boldsymbol{\Phi}\, \phi(\boldsymbol{x}) \tag{20}$$

Note that

$$\boldsymbol{\Phi}\boldsymbol{\Phi}^\top = \begin{bmatrix} \phi(\boldsymbol{x}_1)^\top \phi(\boldsymbol{x}_1) & \dots & \phi(\boldsymbol{x}_1)^\top \phi(\boldsymbol{x}_N) \\ & \vdots & \\ \phi(\boldsymbol{x}_N)^\top \phi(\boldsymbol{x}_1) & \dots & \phi(\boldsymbol{x}_N)^\top \phi(\boldsymbol{x}_N) \end{bmatrix}, \quad \boldsymbol{\Phi}\, \phi(\boldsymbol{x}) = \begin{bmatrix} \phi(\boldsymbol{x}_1)^\top \phi(\boldsymbol{x}) \\ \vdots \\ \phi(\boldsymbol{x}_N)^\top \phi(\boldsymbol{x}) \end{bmatrix} \tag{21}$$

We can use the kernel trick $k(\boldsymbol{u}, \boldsymbol{v}) = \phi(\boldsymbol{u})^\top \phi(\boldsymbol{v})$ !

$(\dagger)$ : $(AB + I)^{-1} A = A(BA + I)^{-1}$

https://en.wikipedia.org/wiki/Woodbury_matrix_identity

# Kernel ridge regression (*cont.*)

$$\hat{y} = \underbrace{\boldsymbol{y}^\top}_{1\times N}\underbrace{(\boldsymbol{\Phi}\boldsymbol{\Phi}^\top + N\lambda\mathbf{I})^{-1}}_{N\times N}\underbrace{\boldsymbol{\Phi}}_{N\times D}\underbrace{\phi(\boldsymbol{x})}_{D\times 1} \tag{22}$$

$$= \underbrace{\boldsymbol{y}^\top}_{1\times N}\underbrace{(\boldsymbol{K}(\mathbf{X},\mathbf{X}) + N\lambda\mathbf{I})^{-1}}_{N\times N}\underbrace{\boldsymbol{K}(\mathbf{X},\boldsymbol{x})}_{N\times 1} \tag{23}$$

where

$$\boldsymbol{K}(\mathbf{X},\mathbf{X}) = \begin{bmatrix} k(\boldsymbol{x}_1,\boldsymbol{x}_1) & \dots & k(\boldsymbol{x}_1,\boldsymbol{x}_N) \\ & \vdots & \\ k(\boldsymbol{x}_N,\boldsymbol{x}_1) & \dots & k(\boldsymbol{x}_N,\boldsymbol{x}_N) \end{bmatrix}, \qquad \boldsymbol{K}(\mathbf{X},\boldsymbol{x}) = \begin{bmatrix} k(\boldsymbol{x}_1,\boldsymbol{x}) \\ \vdots \\ k(\boldsymbol{x}_N,\boldsymbol{x}) \end{bmatrix} \tag{24}$$

NB: the computational complexity of matrix multiplication and matrix inversion for $n \times n$ matrices is approximately $O(n^3)$.

# Kernel ridge regression (*cont.*)

- With the kernel trick,

$$\hat{y} = \underbrace{\boldsymbol{y}^\top}_{1\times N} \underbrace{(\boldsymbol{K}(\mathbf{X},\mathbf{X}) + N\lambda\mathbf{I})^{-1}}_{N\times N} \underbrace{\boldsymbol{K}(\mathbf{X},\boldsymbol{x})}_{N\times 1} \tag{25}$$

$$= \underbrace{\boldsymbol{\alpha}^\top}_{1\times N} \underbrace{\boldsymbol{K}(\mathbf{X},\boldsymbol{x})}_{N\times 1} \tag{26}$$

# Kernel ridge regression (*cont.*)

- With the kernel trick,

$$\hat{y} = \underbrace{\boldsymbol{y}^\top}_{1\times N} \underbrace{(\boldsymbol{K}(\mathbf{X},\mathbf{X}) + N\lambda\mathbf{I})^{-1}}_{N\times N} \underbrace{\boldsymbol{K}(\mathbf{X},\boldsymbol{x})}_{N\times 1} \tag{25}$$

$$= \underbrace{\boldsymbol{\alpha}^\top}_{1\times N} \underbrace{\boldsymbol{K}(\mathbf{X},\boldsymbol{x})}_{N\times 1} \tag{26}$$

- Without the kernel trick,

$$\boldsymbol{w} = \underbrace{(\boldsymbol{\Phi}^\top\boldsymbol{\Phi} + N\lambda\mathbf{I})^{-1}}_{D\times D} \underbrace{\boldsymbol{\Phi}^\top}_{D\times N} \underbrace{\boldsymbol{y}}_{N\times 1} \tag{27}$$

$$\hat{y} = \underbrace{\boldsymbol{w}^\top}_{1\times D} \underbrace{\phi(\boldsymbol{x})}_{D\times 1} \tag{28}$$

# Kernel ridge regression (*cont.*)

- With the kernel trick,

$$\hat{y} = \underbrace{\boldsymbol{y}^\top}_{1\times N} \underbrace{(\boldsymbol{K}(\mathbf{X},\mathbf{X}) + N\lambda\mathbf{I})^{-1}}_{N\times N} \underbrace{\boldsymbol{K}(\mathbf{X},\boldsymbol{x})}_{N\times 1} \tag{25}$$

$$= \underbrace{\boldsymbol{\alpha}^\top}_{1\times N} \underbrace{\boldsymbol{K}(\mathbf{X},\boldsymbol{x})}_{N\times 1} \tag{26}$$

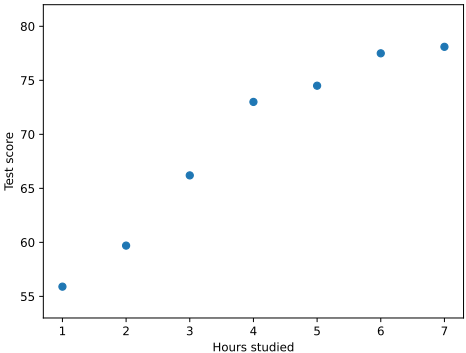
- Without the kernel trick,

$$\boldsymbol{w} = \underbrace{(\boldsymbol{\Phi}^\top\boldsymbol{\Phi} + N\lambda\mathbf{I})^{-1}}_{D\times D} \underbrace{\boldsymbol{\Phi}^\top}_{D\times N} \underbrace{\boldsymbol{y}}_{N\times 1} \tag{27}$$

$$\hat{y} = \underbrace{\boldsymbol{w}^\top}_{1\times D} \underbrace{\phi(\boldsymbol{x})}_{D\times 1} \tag{28}$$

- If $D \gg N$, it's faster and more efficient to use (26) than (28).

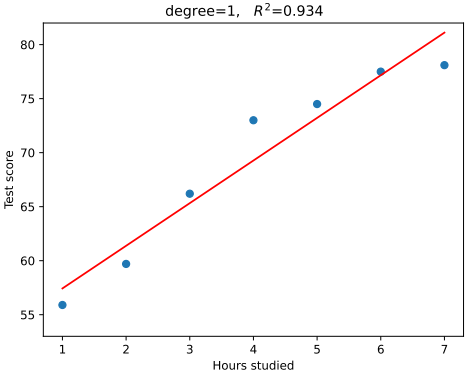
# Example – impact of outliers




Clean data

Data with an outlier

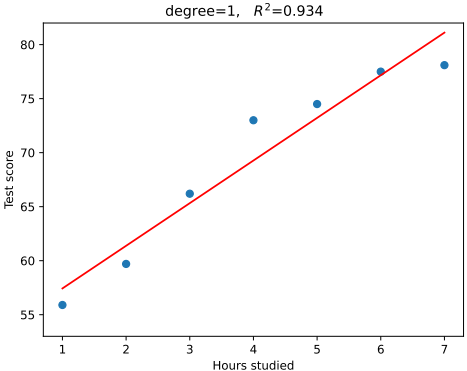
# Example – impact of outliers



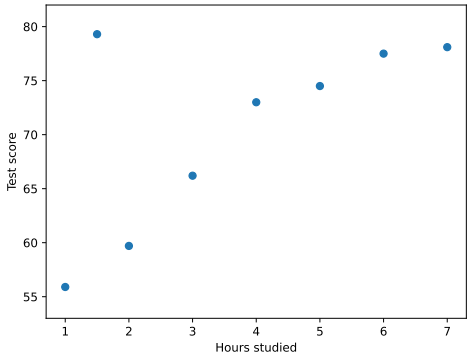

Clean data

Data with an outlier

# Example – impact of outliers

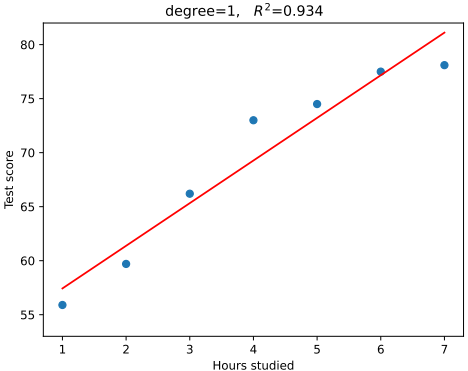



Clean data

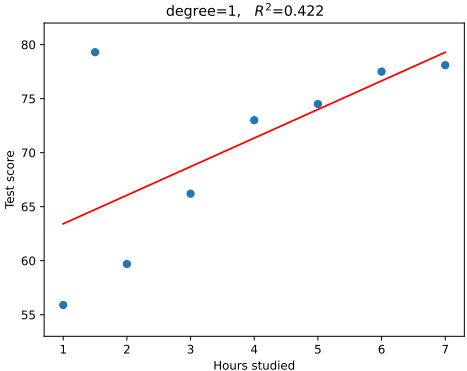



Data with an outlier

# Example – impact of outliers

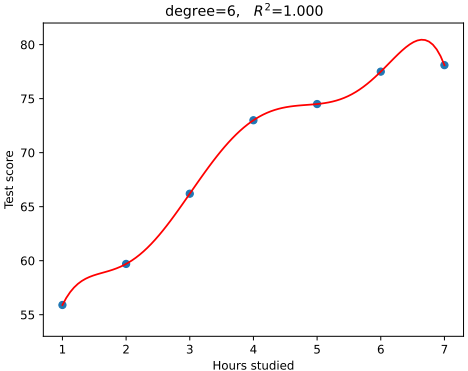



Clean data

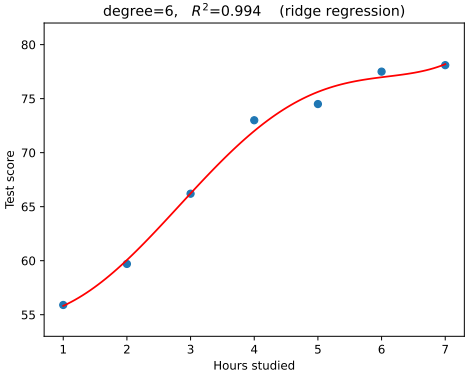



Data with an outlier

# Example – impact of outliers (*cont.*)

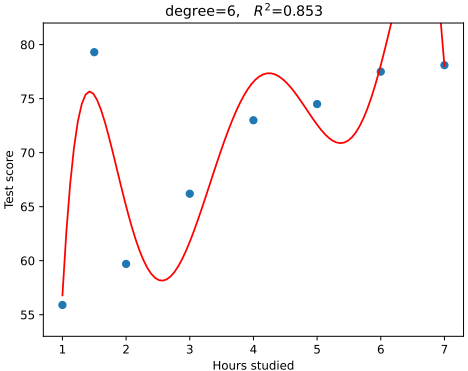



Polynomial regression

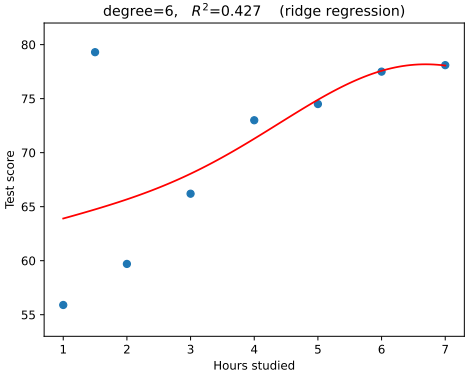



Ridge regression

# Example – impact of outliers (*cont.*)




Polynomial regression




Ridge regression

# Practical issues

- The runtime is not particularly suitable for large data sets.
- Is it right to evaluate the performance of linear regression on training data alone?
- What if there are outliers?
- What if the features are highly correlated?

# Topics not covered

- Choices of features $\boldsymbol{x}$ (feature selection)
- Interpretations of the model parameters $\boldsymbol{w}$
- Collinearity
- Heteroscedasticity
- Other linear regression models (e.g., LASSO, Bayesian linear regression)
- Multiple linear regression
- Relationships with neural networks
- Bias-variance trade-off
- Relationships with principal component analysis (PCA)

# Quizzes

1. Show the derivation of (4) to (5).
2. Show the derivation of (8).
3. Show the derivation of (9) to (10).
4. Show the derivation of (19) to (20).