Machine Learning in Precision Medicine

Linear Regression & Regularization

Prof. Dr. Christoph Lippert

Dr. Stefan Konigorski

Digital Health & Machine Learning

Formalities

All lecture and exercise materials are now available at
https://open.hpi.de/courses/MLPrecisionMedicineSoSe2019.

Further news and information through our mailing list, where you also submit your exercise solutions.

Further questions regarding formalities?

Contents of today: Linear Regression

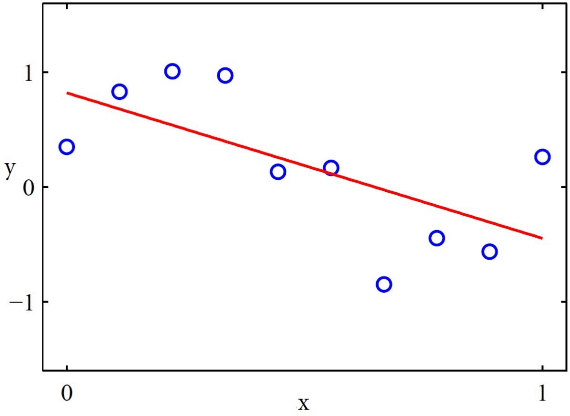
- Introduction
- Maximum likelihood estimation
- Polynomial curve fitting
- Solutions to overfitting
- Regularizes least squares

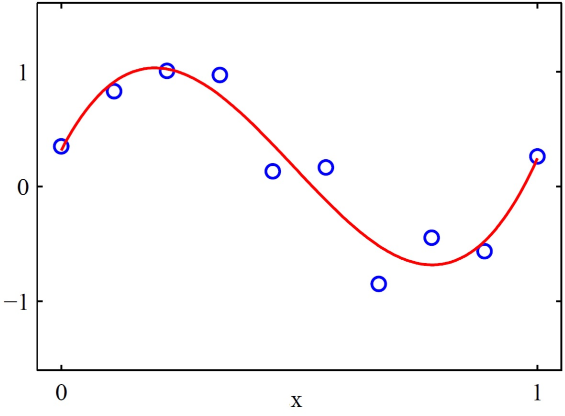
Recap & Quiz

What is a linear (regression) model?

Recap & Quiz

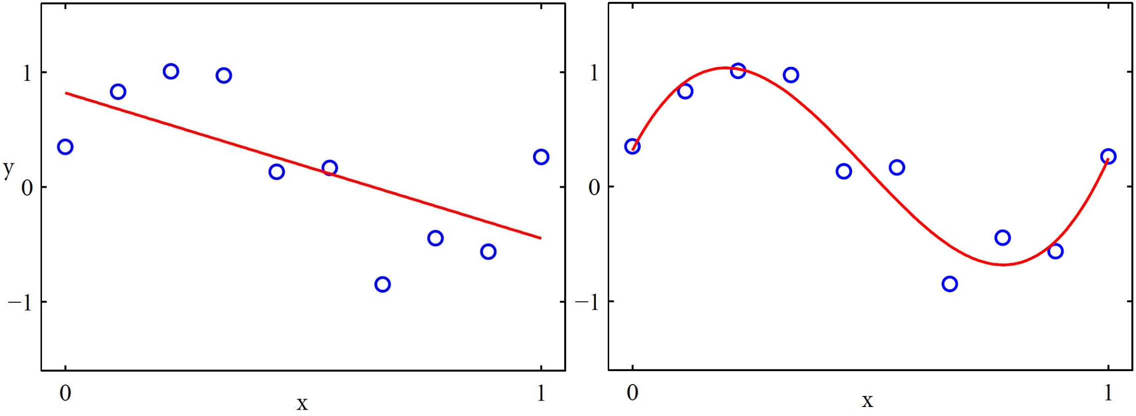
What is a linear (regression) model?





Recap & Quiz

What is a linear (regression) model?



$\longrightarrow$ linear in parameters (e.g. x^3 could be redefined as z).

Recap & Quiz

What is the likelihood?

Recap & Quiz

What is the likelihood?
Probability of data given parameters: $p(\mathcal{D}|\boldsymbol{\theta})$

Recap & Quiz

What is the likelihood?
Probability of data given parameters: $p(\mathcal{D}|\boldsymbol{\theta})$

What is the rationale behind using maximum likelihood as a criterion for finding the best line fitting 2D data points?

Recap & Quiz

What is the likelihood?
Probability of data given parameters: $p(\mathcal{D}|\boldsymbol{\theta})$

What is the rationale behind using maximum likelihood as a criterion for finding the best line fitting 2D data points?

How do you obtain the maximum likelihood values for your parameters ($=$ MLE, maximum likehood estimates)?

Recap & Quiz

What is the likelihood?
Probability of data given parameters: $p(\mathcal{D}|\boldsymbol{\theta})$

What is the rationale behind using maximum likelihood as a criterion for finding the best line fitting 2D data points?

How do you obtain the maximum likelihood values for your parameters ($=$ MLE, maximum likehood estimates)?
Set first derivative (w.r.t. each parameter of interest) of the log-likelihood to 0, solve.

Recap & Quiz

What do you need to know to write out the likelihood?

Recap & Quiz

What do you need to know to write out the likelihood?

- Model
- Distribution (pdf) of model, whether points are iid (i.e. whether pdf of all points factorizes into product of single points)
- and ...

Recap & Quiz

What do you need to know to write out the likelihood?

- Model
- Distribution (pdf) of model, whether points are iid (i.e. whether pdf of all points factorizes into product of single points)
- and ...
- that's all! Easy to get closed form, once you know (i.e. assume) a specific parametric distribution.

Maximum likelihood

- Setting: data $\mathcal{D}$ consisting of N data points $\{\mathbf{x}_n, y_n\}_{n=1}^N$, where $\mathbf{x}_n$ contains D features $\mathbf{x}_n = (x_{n1}, \dots, x_{nD})$.

Maximum likelihood

- Setting: data $\mathcal{D}$ consisting of N data points $\{\mathbf{x}_n, y_n\}_{n=1}^N$, where $\mathbf{x}_n$ contains D features $\mathbf{x}_n = (x_{n1}, \ldots, x_{nD})$.
- In matrix formulation: $\mathbf{X} = \begin{bmatrix} x_{11} & \ldots & x_{1D} \\ \vdots & \ddots & \vdots \\ x_{N1} & \ldots & x_{ND} \end{bmatrix} = \begin{bmatrix} \mathbf{x}_1 \\ \vdots \\ \mathbf{x}_N \end{bmatrix}.$

Maximum likelihood

- Setting: data $\mathcal{D}$ consisting of N data points $\{\mathbf{x}_n, y_n\}_{n=1}^{N}$, where $\mathbf{x}_n$ contains D features $\mathbf{x}_n = (x_{n1}, \ldots, x_{nD})$.
- In matrix formulation: $\mathbf{X} = \begin{bmatrix} x_{11} & \ldots & x_{1D} \\ \vdots & \ddots & \vdots \\ x_{N1} & \ldots & x_{ND} \end{bmatrix} = \begin{bmatrix} \mathbf{x}_1 \\ \vdots \\ \mathbf{x}_N \end{bmatrix}$.
- Assumption: linear model $y_n = \mathbf{x}_n \cdot \boldsymbol{\beta} + \epsilon_n$ with $\boldsymbol{\beta} \in \mathbb{R}^D$, $\epsilon_n \sim \mathcal{N}(0, \sigma^2)$.

Maximum likelihood

- Setting: data $\mathcal{D}$ consisting of N data points $\{\mathbf{x}_n, y_n\}_{n=1}^{N}$, where $\mathbf{x}_n$ contains D features $\mathbf{x}_n = (x_{n1}, \ldots, x_{nD})$.
- In matrix formulation: $\mathbf{X} = \begin{bmatrix} x_{11} & \ldots & x_{1D} \\ \vdots & \ddots & \vdots \\ x_{N1} & \ldots & x_{ND} \end{bmatrix} = \begin{bmatrix} \mathbf{x}_1 \\ \vdots \\ \mathbf{x}_N \end{bmatrix}$.
- Assumption: linear model $y_n = \mathbf{x}_n \cdot \boldsymbol{\beta} + \epsilon_n$ with $\boldsymbol{\beta} \in \mathbb{R}^D$, $\epsilon_n \sim \mathcal{N}(0, \sigma^2)$.
- Aim: Learn parameters by maximizing the likelihood.

Maximum likelihood

- Setting: data $\mathcal{D}$ consisting of N data points $\{\mathbf{x}_n, y_n\}_{n=1}^N$, where $\mathbf{x}_n$ contains D features $\mathbf{x}_n = (x_{n1}, \ldots, x_{nD})$.
- In matrix formulation: $\mathbf{X} = \begin{bmatrix} x_{11} & \ldots & x_{1D} \\ \vdots & \ddots & \vdots \\ x_{N1} & \ldots & x_{ND} \end{bmatrix} = \begin{bmatrix} \mathbf{x}_1 \\ \vdots \\ \mathbf{x}_N \end{bmatrix}$.
- Assumption: linear model $y_n = \mathbf{x}_n \cdot \boldsymbol{\beta} + \epsilon_n$ with $\boldsymbol{\beta} \in \mathbb{R}^D$, $\epsilon_n \sim \mathcal{N}(0, \sigma^2)$.
- Aim: Learn parameters by maximizing the likelihood.
- Just to make sure: what are the parameters to be learned?

Maximum likelihood

- Taking the logarithm, we obtain

$$
\begin{aligned}
\ln p(\mathbf{y}|\boldsymbol{\theta}, \sigma^2) &= \sum_{n=1}^{N} \ln \mathcal{N}\left(y_n \,\middle|\, \mathbf{x}_n \cdot \boldsymbol{\beta}, \sigma^2\right) \\
&= -\frac{N}{2} \ln 2\pi\sigma^2 - \frac{1}{2\sigma^2} \underbrace{\sum_{n=1}^{N} (y_n - \mathbf{x}_n \cdot \boldsymbol{\beta})^2}_{\text{Sum of squares}}
\end{aligned}
$$

Maximum likelihood

- Taking the logarithm, we obtain

$$
\begin{aligned}
\ln p(\mathbf{y}|\boldsymbol{\theta}, \sigma^2) &= \sum_{n=1}^{N} \ln \mathcal{N}\left(y_n \,\middle|\, \mathbf{x}_n \cdot \boldsymbol{\beta}, \sigma^2\right) \\
&= -\frac{N}{2} \ln 2\pi\sigma^2 - \frac{1}{2\sigma^2} \underbrace{\sum_{n=1}^{N} (y_n - \mathbf{x}_n \cdot \boldsymbol{\beta})^2}_{\text{Sum of squares}}
\end{aligned}
$$

- The likelihood is maximized when the squared error is minimized.

Maximum likelihood

- Taking the logarithm, we obtain

$$\ln p(\mathbf{y}|\boldsymbol{\theta}, \sigma^2) = \sum_{n=1}^{N} \ln \mathcal{N}\left(y_n \,\middle|\, \mathbf{x}_n \cdot \boldsymbol{\beta}, \sigma^2\right)$$

$$= -\frac{N}{2} \ln 2\pi\sigma^2 - \frac{1}{2\sigma^2} \underbrace{\sum_{n=1}^{N} (y_n - \mathbf{x}_n \cdot \boldsymbol{\beta})^2}_{\text{Sum of squares}}$$

- The likelihood is maximized when the squared error is minimized.
- Least squares and maximum likelihood are equivalent.

Maximum likelihood

- Taking the logarithm, we obtain

$$\begin{aligned}\ln p(\mathbf{y}|\boldsymbol{\theta},\sigma^2) &= \sum_{n=1}^{N} \ln \mathcal{N}\left(y_n \,\middle|\, \mathbf{x}_n \cdot \boldsymbol{\beta}, \sigma^2\right) \\ &= -\frac{N}{2}\ln 2\pi\sigma^2 - \frac{1}{2\sigma^2}\underbrace{\sum_{n=1}^{N}(y_n - \mathbf{x}_n \cdot \boldsymbol{\beta})^2}_{\text{Sum of squares}}\end{aligned}$$

- The likelihood is maximized when the squared error is minimized.
- Least squares and maximum likelihood are equivalent.
- Question: which assumption do you make using least squares to learn parameter values?

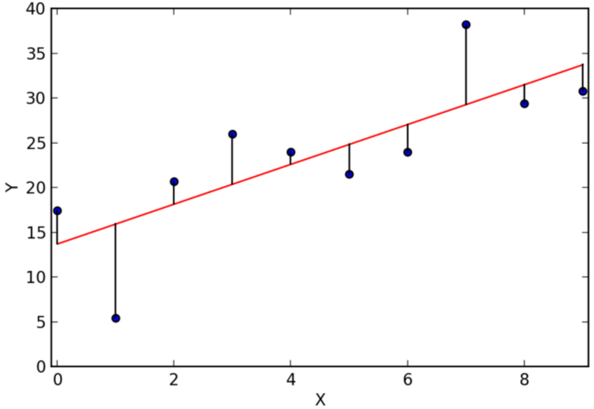
Maximum likelihood

- Taking the logarithm, we obtain

$$\ln p(\mathbf{y}|\boldsymbol{\theta}, \sigma^2) = \sum_{n=1}^{N} \ln \mathcal{N}\left(y_n \,\middle|\, \mathbf{x}_n \cdot \boldsymbol{\beta}, \sigma^2\right)$$
$$= -\frac{N}{2} \ln 2\pi\sigma^2 - \frac{1}{2\sigma^2} \underbrace{\sum_{n=1}^{N} (y_n - \mathbf{x}_n \cdot \boldsymbol{\beta})^2}_{\text{Sum of squares}}$$

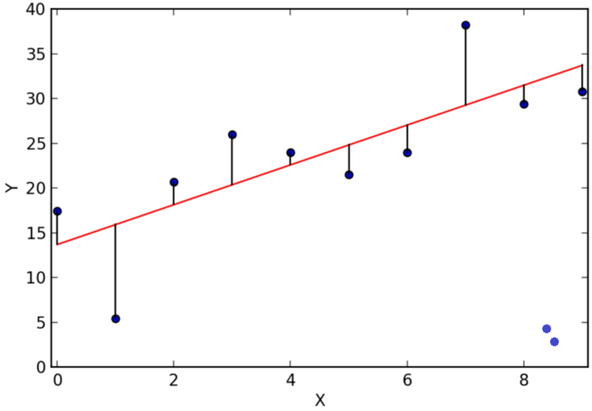
- The likelihood is maximized when the squared error is minimized.
- Least squares and maximum likelihood are equivalent.
- Question: which assumption do you make using least squares to learn parameter values?
- i.e. MLEs under linear model are (relatively) robust against wrong distribution assumption!

Maximum Likelihood and Least Squares



$$\underset{\boldsymbol{\beta}}{\operatorname{argmin}} \frac{1}{2} \sum_{n=1}^{N} (y_n - \mathbf{x}_n \cdot \boldsymbol{\beta})^2$$

Maximum Likelihood and Least Squares



How would the estimated regression line change?

Derive parameter values

- Derivative w.r.t. a single weight entry β_i

$$\frac{d}{\mathrm{d}\beta_d} \ln p(\mathbf{y}|\boldsymbol{\beta}, \sigma^2) = \frac{d}{\mathrm{d}\beta_d} \left[-\frac{N}{2} \ln 2\pi\sigma^2 - \frac{1}{2\sigma^2} \sum_{n=1}^{N} (y_n - \mathbf{x}_n \cdot \boldsymbol{\beta})^2 \right]$$

Derive parameter values

- Derivative w.r.t. a single weight entry β_i

$$\frac{d}{\mathrm{d}\beta_d} \ln p(\mathbf{y}|\boldsymbol{\beta}, \sigma^2) = \frac{d}{\mathrm{d}\beta_d} \left[-\frac{N}{2} \ln 2\pi\sigma^2 - \frac{1}{2\sigma^2} \sum_{n=1}^{N} (y_n - \mathbf{x}_n \cdot \boldsymbol{\beta})^2 \right]$$

$$= \left[\frac{1}{\sigma^2} \sum_{n=1}^{N} x_{nd} (y_n - \mathbf{x}_n \cdot \boldsymbol{\beta}) \right]$$

Derive parameter values

- Derivative w.r.t. a single weight entry β_i

$$\frac{d}{\mathrm{d}\beta_d} \ln p(\mathbf{y}|\boldsymbol{\beta}, \sigma^2) = \frac{d}{\mathrm{d}\beta_d} \left[-\frac{N}{2} \ln 2\pi\sigma^2 - \frac{1}{2\sigma^2} \sum_{n=1}^{N} (y_n - \mathbf{x}_n \cdot \boldsymbol{\beta})^2 \right]$$

$$= \left[\frac{1}{\sigma^2} \sum_{n=1}^{N} x_{nd} (y_n - \mathbf{x}_n \cdot \boldsymbol{\beta}) \right]$$

- Set gradient w.r.t. $\boldsymbol{\beta}$ to zero [vector holding the derivatives $\forall \beta_d$]

$$\nabla_{\boldsymbol{\beta}} \ln p(\mathbf{y}|\boldsymbol{\beta}, \sigma^2) =$$

Derive parameter values

- Derivative w.r.t. a single weight entry β_i

$$\frac{d}{\mathrm{d}\beta_d} \ln p(\mathbf{y}|\boldsymbol{\beta}, \sigma^2) = \frac{d}{\mathrm{d}\beta_d} \left[-\frac{N}{2} \ln 2\pi\sigma^2 - \frac{1}{2\sigma^2} \sum_{n=1}^{N} (y_n - \mathbf{x}_n \cdot \boldsymbol{\beta})^2 \right]$$

$$= \left[\frac{1}{\sigma^2} \sum_{n=1}^{N} x_{nd} (y_n - \mathbf{x}_n \cdot \boldsymbol{\beta}) \right]$$

- Set gradient w.r.t. $\boldsymbol{\beta}$ to zero [vector holding the derivatives $\forall \beta_d$]

$$\nabla_{\boldsymbol{\beta}} \ln p(\mathbf{y}|\boldsymbol{\beta}, \sigma^2) = \frac{1}{\sigma^2} \sum_{n=1}^{N} \mathbf{x}_n^{\mathrm{T}} (y_n - \mathbf{x}_n \cdot \boldsymbol{\beta}) = \mathbf{0} \qquad \text{(where } \mathbf{0} \text{ is a vector of 0s)}$$

Derive parameter values

- Set gradient w.r.t. $\boldsymbol{\beta}$ to zero [vector holding the derivatives $\forall \beta_d$]

$$\nabla_{\boldsymbol{\beta}} \ln p(\mathbf{y}|\boldsymbol{\beta}, \sigma^2) = \frac{1}{\sigma^2} \sum_{n=1}^{N} \mathbf{x}_n^{\mathrm{T}} (y_n - \mathbf{x}_n \cdot \boldsymbol{\beta}) = \mathbf{0} \qquad \text{(where } \mathbf{0} \text{ is a vector of 0s)}$$

Derive parameter values

- Set gradient w.r.t. $\boldsymbol{\beta}$ to zero [vector holding the derivatives $\forall \beta_d$]

$$\nabla_{\boldsymbol{\beta}} \ln p(\mathbf{y}|\boldsymbol{\beta}, \sigma^2) = \frac{1}{\sigma^2} \sum_{n=1}^{N} \mathbf{x}_n^{\mathrm{T}} (y_n - \mathbf{x}_n \cdot \boldsymbol{\beta}) = \mathbf{0} \qquad \text{(where } \mathbf{0} \text{ is a vector of 0s)}$$

- In matrix formulation: $\frac{1}{\sigma^2} \mathbf{X}^{\mathrm{T}} (\mathbf{y} - \mathbf{X} \cdot \boldsymbol{\beta}) = \mathbf{0}$.

Derive parameter values

- Set gradient w.r.t. $\boldsymbol{\beta}$ to zero [vector holding the derivatives $\forall \beta_d$]

$$\nabla_{\boldsymbol{\beta}} \ln p(\mathbf{y}|\boldsymbol{\beta}, \sigma^2) = \frac{1}{\sigma^2} \sum_{n=1}^{N} \mathbf{x}_n^{\mathrm{T}} (y_n - \mathbf{x}_n \cdot \boldsymbol{\beta}) = \mathbf{0} \qquad \text{(where } \mathbf{0} \text{ is a vector of 0s)}$$

- In matrix formulation: $\frac{1}{\sigma^2} \mathbf{X}^{\mathrm{T}} (\mathbf{y} - \mathbf{X} \cdot \boldsymbol{\beta}) = \mathbf{0}$.
- Solving for $\boldsymbol{\beta}$ yields: $\boldsymbol{\beta}_{\mathsf{ML}} =$

Derive parameter values

- Set gradient w.r.t. $\boldsymbol{\beta}$ to zero [vector holding the derivatives $\forall \beta_d$]

$$\nabla_{\boldsymbol{\beta}} \ln p(\mathbf{y}|\boldsymbol{\beta}, \sigma^2) = \frac{1}{\sigma^2} \sum_{n=1}^{N} \mathbf{x}_n^{\mathrm{T}} (y_n - \mathbf{x}_n \cdot \boldsymbol{\beta}) = \mathbf{0} \qquad \text{(where } \mathbf{0} \text{ is a vector of 0s)}$$

- In matrix formulation: $\frac{1}{\sigma^2} \mathbf{X}^{\mathrm{T}} (\mathbf{y} - \mathbf{X} \cdot \boldsymbol{\beta}) = \mathbf{0}$.
- Solving for $\boldsymbol{\beta}$ yields: $\boldsymbol{\beta}_{\mathsf{ML}} = \underbrace{(\mathbf{X}^{\mathrm{T}}\mathbf{X})^{-1}\mathbf{X}^{\mathrm{T}}}_{\text{Pseudo inverse of } \mathbf{X}} \mathbf{y}$

Derive parameter values

- Set gradient w.r.t. $\boldsymbol{\beta}$ to zero [vector holding the derivatives $\forall \beta_d$]

$$\nabla_{\boldsymbol{\beta}} \ln p(\mathbf{y}|\boldsymbol{\beta}, \sigma^2) = \frac{1}{\sigma^2} \sum_{n=1}^{N} \mathbf{x}_n^{\mathrm{T}} (y_n - \mathbf{x}_n \cdot \boldsymbol{\beta}) = \mathbf{0} \qquad \text{(where } \mathbf{0} \text{ is a vector of 0s)}$$

- In matrix formulation: $\frac{1}{\sigma^2} \mathbf{X}^{\mathrm{T}} (\mathbf{y} - \mathbf{X} \cdot \boldsymbol{\beta}) = \mathbf{0}$.
- Solving for $\boldsymbol{\beta}$ yields: $\boldsymbol{\beta}_{\text{ML}} = \underbrace{(\mathbf{X}^{\mathrm{T}} \mathbf{X})^{-1} \mathbf{X}^{\mathrm{T}}}_{\text{Pseudo inverse of } \mathbf{X}} \mathbf{y}$
- Question: what happened to $\frac{1}{\sigma^2}$?

Derive parameter values

- Set gradient w.r.t. $\boldsymbol{\beta}$ to zero [vector holding the derivatives $\forall \beta_d$]

$$\nabla_{\boldsymbol{\beta}} \ln p(\mathbf{y}|\boldsymbol{\beta}, \sigma^2) = \frac{1}{\sigma^2} \sum_{n=1}^{N} \mathbf{x}_n^{\mathrm{T}} (y_n - \mathbf{x}_n \cdot \boldsymbol{\beta}) = \mathbf{0} \qquad \text{(where } \mathbf{0} \text{ is a vector of 0s)}$$

- In matrix formulation: $\frac{1}{\sigma^2} \mathbf{X}^{\mathrm{T}} (\mathbf{y} - \mathbf{X} \cdot \boldsymbol{\beta}) = \mathbf{0}$.
- Solving for $\boldsymbol{\beta}$ yields: $\boldsymbol{\beta}_{\text{ML}} = \underbrace{(\mathbf{X}^{\mathrm{T}}\mathbf{X})^{-1}\mathbf{X}^{\mathrm{T}}}_{\text{Pseudo inverse of } \mathbf{X}} \mathbf{y}$
- Question: what happened to $\frac{1}{\sigma^2}$?
- Learn parameter values for σ^2 similarly.

Polynomial curve fitting

- The above can be extended to more general functions of $\mathbf{x}_n$ that are linear in $\boldsymbol{\beta}$. For simplicity, focus here on $\mathbf{x}_n = x$ (i.e. one feature).

  For example:

Polynomial curve fitting

- The above can be extended to more general functions of $\mathbf{x}_n$ that are linear in $\boldsymbol{\beta}$. For simplicity, focus here on $\mathbf{x}_n = x$ (i.e. one feature).

  For example:

- Use the polynomials up to degree M to construct new features

$$
\begin{aligned}
f(x, \boldsymbol{\beta}) &= \beta_0 + \beta_1 x + \beta_2 x^2 + \cdots + \beta_M x^M \\
&= \boldsymbol{\phi}(x) \cdot \boldsymbol{\beta}
\end{aligned}
$$

  with $\boldsymbol{\phi}(x) = (1, x, x^2, \ldots, x^M), \quad \phi_m(x) = x^m, \quad \boldsymbol{\beta} = (\beta_0, \beta_1, \ldots, \beta_M)^T \in \mathbb{R}^M$.

Polynomial curve fitting

- The above can be extended to more general functions of $\mathbf{x}_n$ that are linear in $\boldsymbol{\beta}$. For simplicity, focus here on $\mathbf{x}_n = x$ (i.e. one feature).

  For example:

- Use the polynomials up to degree M to construct new features

$$\begin{aligned} f(x, \boldsymbol{\beta}) &= \beta_0 + \beta_1 x + \beta_2 x^2 + \cdots + \beta_M x^M \\ &= \boldsymbol{\phi}(x) \cdot \boldsymbol{\beta} \end{aligned}$$

  with $\boldsymbol{\phi}(x) = (1, x, x^2, \ldots, x^M), \quad \phi_m(x) = x^m, \quad \boldsymbol{\beta} = (\beta_0, \beta_1, \ldots, \beta_M)^T \in \mathbb{R}^M$.

- Then, these polynomials can be used to predict $\mathbf{y}$ with $\mathbf{y} = \boldsymbol{\phi}(x) \cdot \boldsymbol{\beta} + \boldsymbol{\epsilon}$.

Polynomial curve fitting

- The above can be extended to more general functions of $\mathbf{x}_n$ that are linear in $\boldsymbol{\beta}$. For simplicity, focus here on $\mathbf{x}_n = x$ (i.e. one feature).

  For example:

- Use the polynomials up to degree M to construct new features

$$\begin{aligned} f(x, \boldsymbol{\beta}) &= \beta_0 + \beta_1 x + \beta_2 x^2 + \cdots + \beta_M x^M \\ &= \boldsymbol{\phi}(x) \cdot \boldsymbol{\beta} \end{aligned}$$

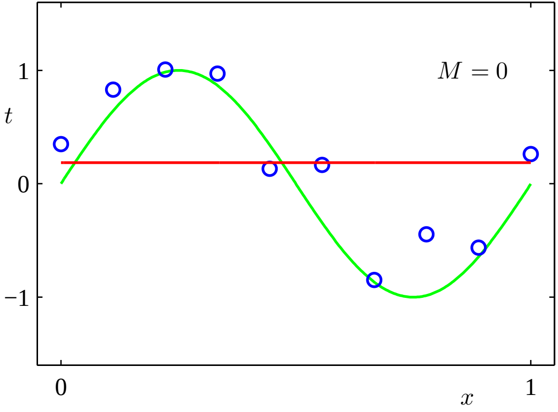
  with $\boldsymbol{\phi}(x) = (1, x, x^2, \ldots, x^M), \quad \phi_m(x) = x^m, \quad \boldsymbol{\beta} = (\beta_0, \beta_1, \ldots, \beta_M)^T \in \mathbb{R}^M$.

- Then, these polynomials can be used to predict $\mathbf{y}$ with $\mathbf{y} = \boldsymbol{\phi}(x) \cdot \boldsymbol{\beta} + \boldsymbol{\epsilon}$.

- Similarly, $\boldsymbol{\phi}$ can be any feature mapping ($\rightarrow$ call ϕ_m basis functions, see next lecture).

Polynomial curve fitting

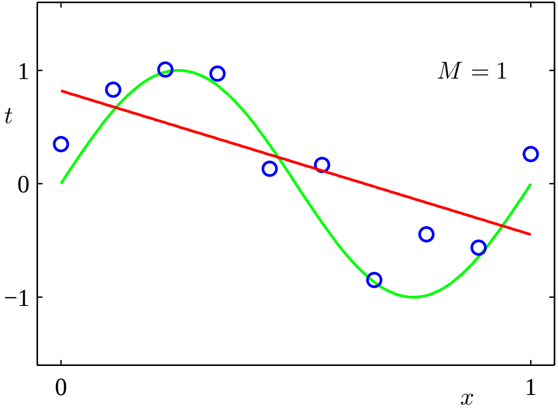
The degree M of the polynomial is crucial.



(C.M. Bishop, Pattern Recognition and Machine Learning)

Polynomial curve fitting

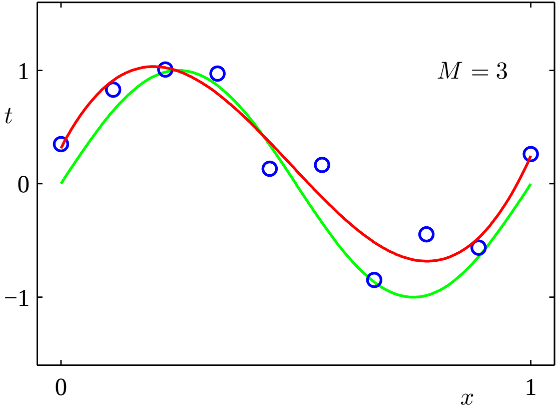
The degree M of the polynomial is crucial.



(C.M. Bishop, Pattern Recognition and Machine Learning)

Polynomial curve fitting

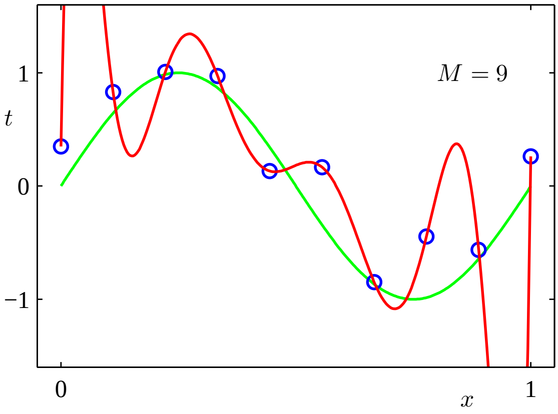
The degree M of the polynomial is crucial.



(C.M. Bishop, Pattern Recognition and Machine Learning)

Polynomial curve fitting

The degree M of the polynomial is crucial.



(C.M. Bishop, Pattern Recognition and Machine Learning)

Polynomial curve fitting

Question: So how should you choose the best M?

Polynomial curve fitting

Question: So how should you choose the best M?

Idea: Fit regression model for any M, compute mean squared error (MSE), choose M with lowest MSE.

Polynomial curve fitting

Question: So how should you choose the best M?

Idea: Fit regression model for any M, compute mean squared error (MSE), choose M with lowest MSE.

Problems?

Polynomial curve fitting

Question: So how should you choose the best M?

Idea: Fit regression model for any M, compute mean squared error (MSE), choose M with lowest MSE.

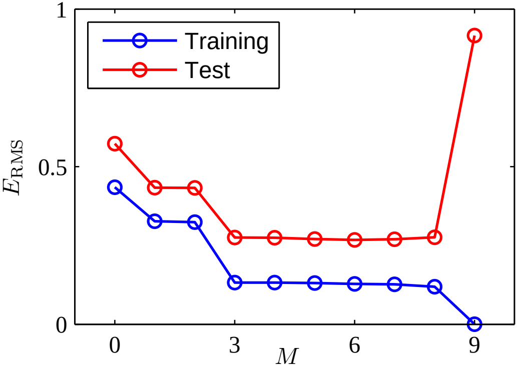
Problems? Overfitting!

Train vs. Test error

- Split sample in training and test set.
- Choose M based on test error.

Train vs. Test error

- Split sample in training and test set.
- Choose M based on test error.



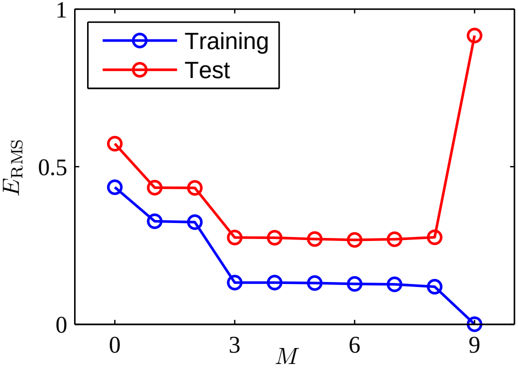
(C.M. Bishop, Pattern Recognition and Machine Learning)

- Intuition? Interpretation?

Train vs. Test error

How do you compute the test error?

- Split sample in training and test set.
- Choose M based on test error.

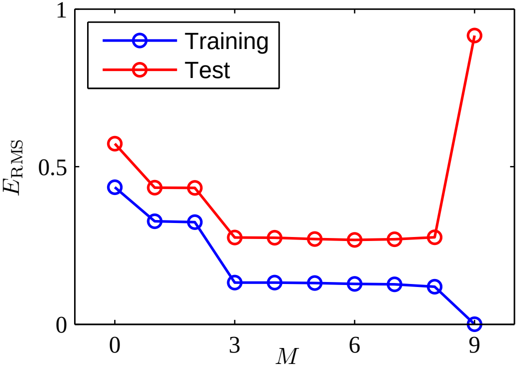


(C.M. Bishop, Pattern Recognition and Machine Learning)

- Intuition? Interpretation?

Train vs. Test error

- Split sample in training and test set.
- Choose M based on test error.



(C.M. Bishop, Pattern Recognition and Machine Learning)

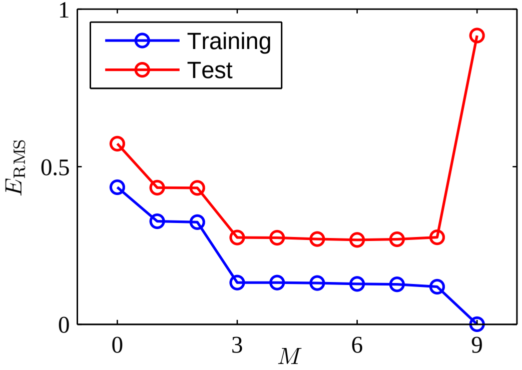
- Intuition? Interpretation?

How do you compute the test error?

Cross validation:

Train vs. Test error

- Split sample in training and test set.
- Choose M based on test error.



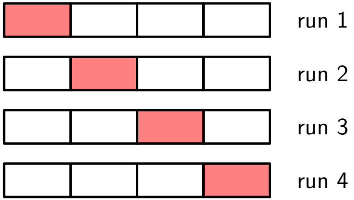
(C.M. Bishop, Pattern Recognition and Machine Learning)

- Intuition? Interpretation?

How do you compute the test error?

Cross validation:

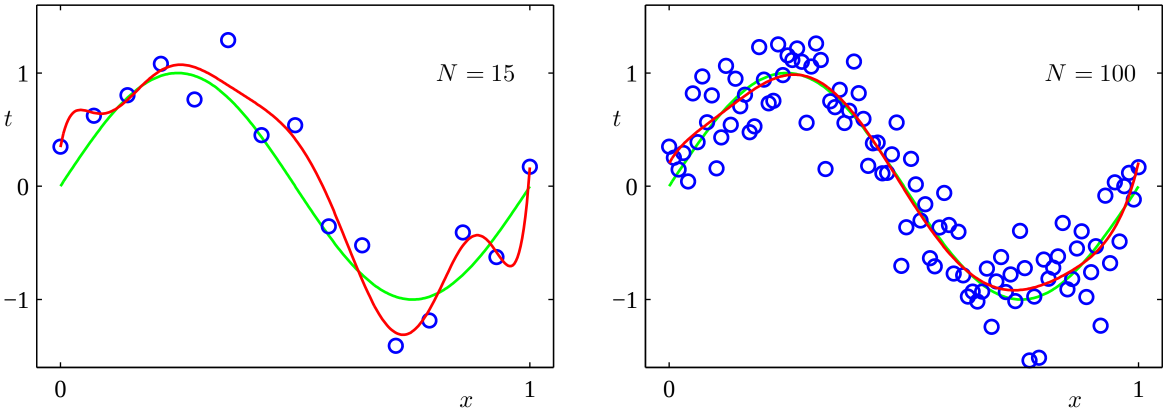
- randomly assign samples $(\mathbf{x}_n, y_n)$ to S sets of equal size
- for each set $s \in S$ (e.g. $S = 4$) and $m \in [1, \ldots, M]$:
  - train model on $S - 1$ remaining sets
  - predict on s and compute MSE or root MSE E_{RMS}.
- compute average MSE for degree m.
- pick m with lowest MSE.



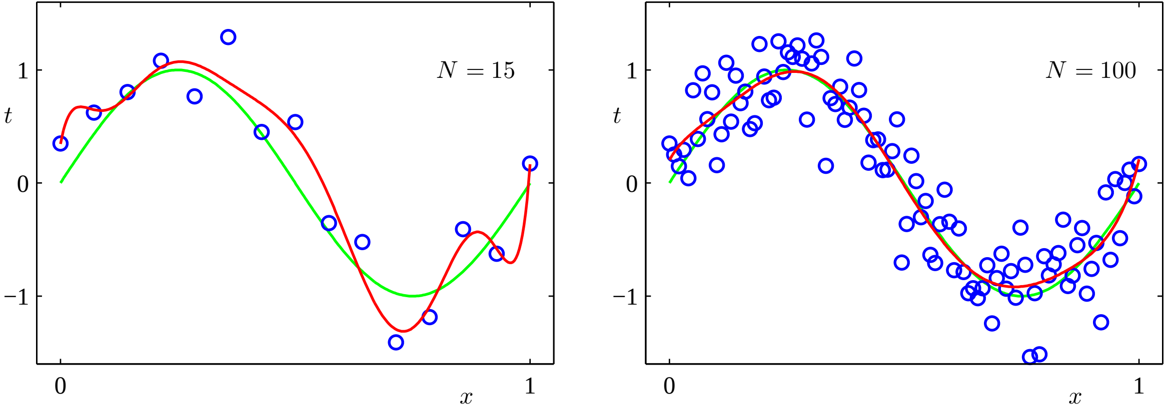
(C.M. Bishop, Pattern Recognition and Machine Learning)

What else can you do to improve model, i.e. learn better weights, find better curve fit, and be less prone to overfitting?

Get more data

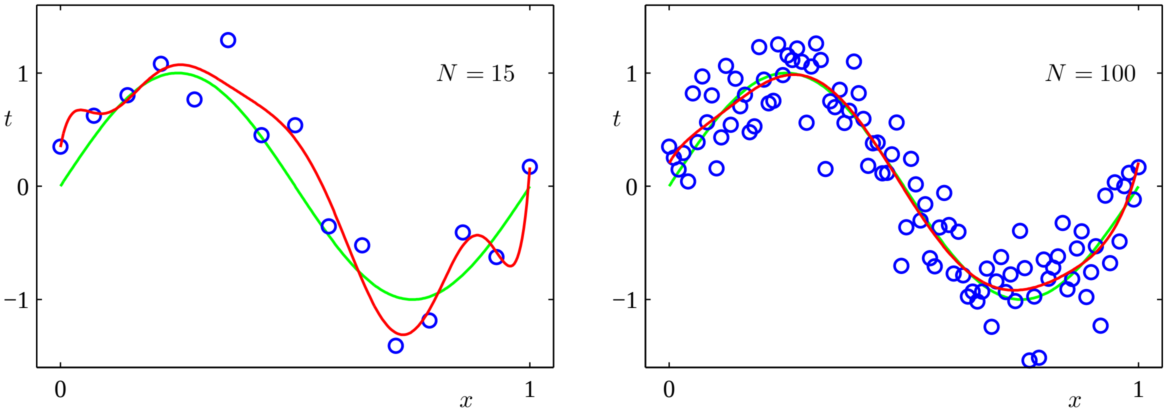


Get more data



Others ways to avoid overfitting and fit complex model for limited number of observations?

Get more data



Others ways to avoid overfitting and fit complex model for limited number of observations?

Regularization of weights

Regularization

Regularize the regression weights $\boldsymbol{\beta}$.

Loss function: $$\mathrm{E}_2(\boldsymbol{\beta}) = \underbrace{\frac{1}{2}\sum_{n=1}^{N}(y_n - \mathbf{x}_n \cdot \boldsymbol{\beta})^2}_{\text{squared error}} + \underbrace{\frac{\lambda}{2}\sum_{d=1}^{D}\beta_d^2}_{l_2\text{-norm regularizer}}$$

where $||\boldsymbol{\beta}||^2 = \sqrt{\sum_{d=1}^{D}\beta_d^2}$ is the l_2-norm of $\boldsymbol{\beta}$.

Regularization

Regularize the regression weights $\boldsymbol{\beta}$.

Loss function: $$\mathrm{E}_2(\boldsymbol{\beta}) = \underbrace{\frac{1}{2}\sum_{n=1}^{N}(y_n - \mathbf{x}_n \cdot \boldsymbol{\beta})^2}_{\text{squared error}} \quad + \quad \underbrace{\frac{\lambda}{2}\sum_{d=1}^{D}\beta_d^2}_{l_2\text{-norm regularizer}}$$

where $||\boldsymbol{\beta}||^2 = \sqrt{\sum_{d=1}^{D} \beta_d^2}$ is the l_2-norm of $\boldsymbol{\beta}$.

What effect does this have? How will the learned weights be different?

Regularization

Regularize the regression weights $\boldsymbol{\beta}$.

Loss function: $$\mathrm{E}_2(\boldsymbol{\beta}) = \underbrace{\frac{1}{2}\sum_{n=1}^{N}(y_n - \mathbf{x}_n \cdot \boldsymbol{\beta})^2}_{\text{squared error}} \quad + \quad \underbrace{\frac{\lambda}{2}\sum_{d=1}^{D}\beta_d^2}_{l_2\text{-norm regularizer}}$$

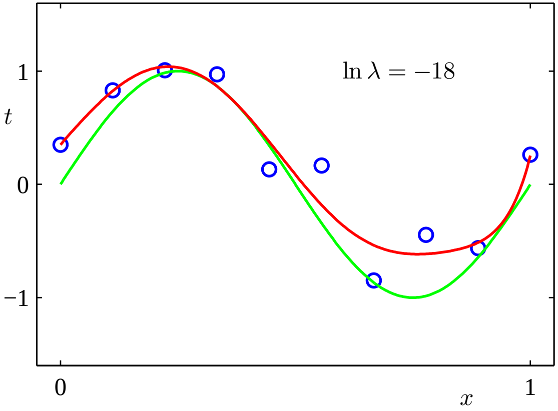
where $||\boldsymbol{\beta}||^2 = \sqrt{\sum_{d=1}^{D}\beta_d^2}$ is the l_2-norm of $\boldsymbol{\beta}$.

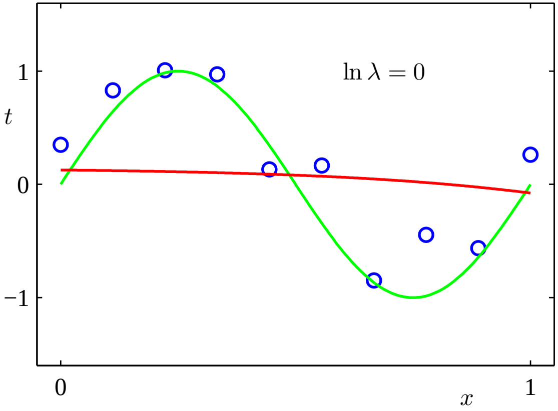
What effect does this have? How will the learned weights be different?

- Penalizes large weights.
- Reduces the complexity of the function that associates $\mathbf{x}$ with $\mathbf{y}$, i.e. learn parsimonious model.
- Also known as shrinkage methods.

Linear Regression
Regularization

Loss function: $$\mathrm{E}_2(\boldsymbol{\beta}) = \underbrace{\frac{1}{2}\sum_{n=1}^{N}(y_n - \mathbf{x}_n \cdot \boldsymbol{\beta})^2}_{\text{squared error}} + \underbrace{\frac{\lambda}{2}\sum_{d=1}^{D}\beta_d^2}_{l_2\text{-norm regularizer}}$$





Regularization

$$\mathrm{E}_2(\boldsymbol{\beta}) = \underbrace{\frac{1}{2}\sum_{n=1}^{N}(y_n - \mathbf{x}_n \cdot \boldsymbol{\beta})^2}_{\text{squared error}} \quad + \quad \underbrace{\frac{\lambda}{2}\sum_{d=1}^{D}\beta_d^2}_{l_2\text{-norm regularizer}}$$

Quiz: How to chose an optimal λ?

Regularization

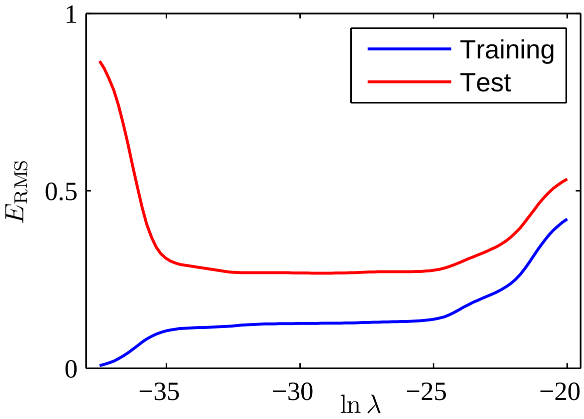
$$\mathrm{E}_2(\boldsymbol{\beta}) = \underbrace{\frac{1}{2}\sum_{n=1}^{N}(y_n - \mathbf{x}_n \cdot \boldsymbol{\beta})^2}_{\text{squared error}} \quad + \quad \underbrace{\frac{\lambda}{2}\sum_{d=1}^{D}\beta_d^2}_{l_2\text{-norm regularizer}}$$

Quiz: How to chose an optimal λ?

Answer: Look at the test error!

Regularization

$$\mathrm{E}_2(\boldsymbol{\beta}) = \underbrace{\frac{1}{2}\sum_{n=1}^{N}(y_n - \mathbf{x}_n \cdot \boldsymbol{\beta})^2}_{\text{squared error}} \quad + \quad \underbrace{\frac{\lambda}{2}\sum_{d=1}^{D}\beta_d^2}_{l_2\text{-norm regularizer}}$$



Quiz: How to chose an optimal λ?

Answer: Look at the test error!

Solving ridge regression

Minimize the loss function $\mathrm{E}_2(\boldsymbol{\beta}) = \underbrace{\frac{1}{2}\sum_{n=1}^{N}(y_n - \mathbf{x}_n \cdot \boldsymbol{\beta})^2}_{\text{squared error}} \quad + \quad \underbrace{\frac{\lambda}{2}\sum_{d=1}^{D}\beta_d^2}_{l_2\text{-norm regularizer}}$

Solving ridge regression

Minimize the loss function $\mathrm{E}_2(\boldsymbol{\beta}) = \underbrace{\frac{1}{2}\sum_{n=1}^{N}(y_n - \mathbf{x}_n \cdot \boldsymbol{\beta})^2}_{\text{squared error}} + \underbrace{\frac{\lambda}{2}\sum_{d=1}^{D}\beta_d^2}_{l_2\text{-norm regularizer}}$

- Derivative w.r.t. a single weight entry β_i

$$\frac{d}{\mathrm{d}\beta_d}\left[\frac{1}{2}\sum_{n=1}^{N}(y_n - \mathbf{x}_n \cdot \boldsymbol{\beta})^2 + \frac{\lambda}{2}\sum_{d=1}^{D}\beta_d^2\right]$$

Solving ridge regression

Minimize the loss function $\mathrm{E}_2(\boldsymbol{\beta}) = \underbrace{\frac{1}{2}\sum_{n=1}^{N}(y_n - \mathbf{x}_n \cdot \boldsymbol{\beta})^2}_{\text{squared error}} + \underbrace{\frac{\lambda}{2}\sum_{d=1}^{D}\beta_d^2}_{l_2\text{-norm regularizer}}$

- Derivative w.r.t. a single weight entry β_i

$$\frac{d}{\mathrm{d}\beta_d}\left[\frac{1}{2}\sum_{n=1}^{N}(y_n - \mathbf{x}_n \cdot \boldsymbol{\beta})^2 + \frac{\lambda}{2}\sum_{d=1}^{D}\beta_d^2\right] = \sum_{n=1}^{N} -x_{nd}(y_n - \mathbf{x}_n \cdot \boldsymbol{\beta}) + \lambda\beta_d$$

Solving ridge regression

Minimize the loss function $\mathrm{E}_2(\boldsymbol{\beta}) = \underbrace{\frac{1}{2}\sum_{n=1}^{N}(y_n - \mathbf{x}_n \cdot \boldsymbol{\beta})^2}_{\text{squared error}} + \underbrace{\frac{\lambda}{2}\sum_{d=1}^{D}\beta_d^2}_{l_2\text{-norm regularizer}}$

- Derivative w.r.t. a single weight entry β_i

$$\frac{d}{\mathrm{d}\beta_d}\left[\frac{1}{2}\sum_{n=1}^{N}(y_n - \mathbf{x}_n \cdot \boldsymbol{\beta})^2 + \frac{\lambda}{2}\sum_{d=1}^{D}\beta_d^2\right] = \sum_{n=1}^{N} -x_{nd}(y_n - \mathbf{x}_n \cdot \boldsymbol{\beta}) + \lambda\beta_d$$

- Set gradient w.r.t. $\boldsymbol{\beta}$ to zero [vector holding the derivatives $\forall\beta_d$]

$$\nabla_{\boldsymbol{\beta}}\mathrm{E}_2(\boldsymbol{\beta}) = \sum_{n=1}^{N}\mathbf{X}^{\mathrm{T}}(\mathbf{x}_n \cdot \boldsymbol{\beta} - y_n) + \lambda\boldsymbol{\beta} = \mathbf{0}$$

Solving ridge regression

Minimize the loss function $\mathrm{E}_2(\boldsymbol{\beta}) = \underbrace{\frac{1}{2}\sum_{n=1}^{N}(y_n - \mathbf{x}_n \cdot \boldsymbol{\beta})^2}_{\text{squared error}} + \underbrace{\frac{\lambda}{2}\sum_{d=1}^{D}\beta_d^2}_{l_2\text{-norm regularizer}}$

- Derivative w.r.t. a single weight entry β_i

$$\frac{d}{\mathrm{d}\beta_d}\left[\frac{1}{2}\sum_{n=1}^{N}(y_n - \mathbf{x}_n \cdot \boldsymbol{\beta})^2 + \frac{\lambda}{2}\sum_{d=1}^{D}\beta_d^2\right] = \sum_{n=1}^{N} -x_{nd}(y_n - \mathbf{x}_n \cdot \boldsymbol{\beta}) + \lambda\beta_d$$

- Set gradient w.r.t. $\boldsymbol{\beta}$ to zero [vector holding the derivatives $\forall\beta_d$]

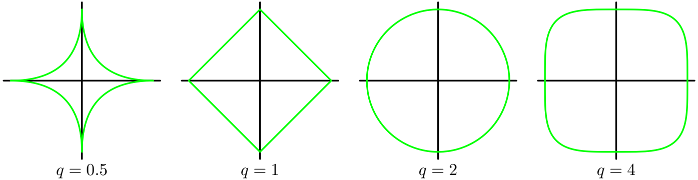
$$\nabla_{\boldsymbol{\beta}}\mathrm{E}_2(\boldsymbol{\beta}) = \sum_{n=1}^{N}\mathbf{X}^{\mathrm{T}}(\mathbf{x}_n \cdot \boldsymbol{\beta} - y_n) + \lambda\boldsymbol{\beta} = \mathbf{0}$$

$$\Longrightarrow \boldsymbol{\beta}_{Ridge} = \left(\mathbf{X}^{\mathrm{T}}\mathbf{X} + \lambda\mathbf{I}\right)^{-1}\mathbf{X}^{\mathrm{T}}\mathbf{y}$$

Regularization

A more general regularization:

$$\mathrm{E}_q(\boldsymbol{\beta}) = \underbrace{\frac{1}{2}\sum_{n=1}^{N}(y_n - \mathbf{x}_n \cdot \boldsymbol{\beta})^2}_{\text{squared error}} + \underbrace{\frac{\lambda}{2}\sum_{d=1}^{D}|\beta_d|^q}_{l_q\text{-norm regularizer}}$$



(C.M. Bishop, Pattern Recognition and Machine Learning)

Regularization

A more general regularization:

$$\mathrm{E}_q(\boldsymbol{\beta}) = \underbrace{\frac{1}{2}\sum_{n=1}^{N}(y_n - \mathbf{x}_n \cdot \boldsymbol{\beta})^2}_{\text{squared error}} + \underbrace{\frac{\lambda}{2}\sum_{d=1}^{D}|\beta_d|^q}_{l_q\text{-norm regularizer}}$$

- l_1: Lasso $\rightarrow$ feature selection, sparse solution
- l_2: Ridge $\rightarrow$ shrinks weights to 0
- Linear combination of l_1 and l_2: Elastic net

Summary

- Maximum likelihood in linear regression
- Polynomial linear regression
- Cross-classification
- Regularized regression with l_2-norm (ridge regression)

Questions?