TECHNISCHE UNIVERSITÄT DARMSTADT

Statistical Machine Learning

Lecture 08: Regression

Kristian Kersting

TU Darmstadt

Summer Term 2020

Today's Objectives

- Make you understand how to learn a continuous function
- Covered Topics
  - Linear Regression and its interpretations
  - What is overfitting?
  - Deriving Linear Regression from Maximum Likelihood Estimation
  - Bayesian Linear Regression

Outline

1. Introduction to Linear Regression

2. Maximum Likelihood Approach to Regression

3. Bayesian Linear Regression

4. Wrap-Up

Outline

TECHNISCHE UNIVERSITÄT DARMSTADT

Reminder

- Our task is to learn a mapping f from input to output

$$f : I \rightarrow O, \quad y = f(x; \theta)$$

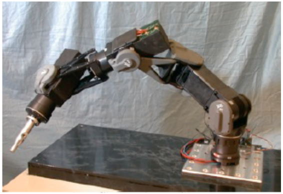
  - Input: $x \in I$ (images, text, sensor measurements, ...)
  - Output: $y \in O$
  - Parameters: $\theta \in \Theta$ (what needs to be "learned")
- **Regression**
  - Learn a mapping into a continuous space
    - $O = \mathbb{R}$
    - $O = \mathbb{R}^3$
    - $\ldots$

TECHNISCHE UNIVERSITÄT DARMSTADT

Motivation

- You want to predict the torques of a robot arm

$$
\begin{aligned}
y &= l\ddot{q} - \mu\dot{q} + mlg\sin(q) \\
&= \begin{bmatrix} \ddot{q} & \dot{q} & \sin(q) \end{bmatrix} \begin{bmatrix} l & -\mu & mlg \end{bmatrix}^{\intercal} \\
&= \phi(\mathbf{x})^{\intercal}\boldsymbol{\theta}
\end{aligned}
$$



- Can we do this with a data set?

$$\mathcal{D} = \left\{ (\mathbf{x}_i, y_i) \,\middle|\, i = 1 \cdots n \right\}$$

- A linear regression problem!

TECHNISCHE UNIVERSITÄT DARMSTADT

Least Squares Linear Regression

- We are given pairs of training data points and associated function values $(\mathbf{x}_i, y_i)$

$$X = \left\{\mathbf{x}_1 \in \mathbb{R}^d, \ldots, \mathbf{x}_n\right\}$$
$$Y = \{y_1 \in R, \ldots, y_n\}$$

  - Note: here we only do the case $y_i \in \mathbb{R}$. In general y_i can have more than one dimension, i.e., $y_i \in \mathbb{R}^f$ for some positive f

- Start with linear regressor

$$\mathbf{x}_i^\intercal \mathbf{w} + w_0 = y_i \quad \forall i = 1, \ldots, n$$

  - One linear equation for each training data point/label pair
  - Exactly the same basic setup as for least-squares classification! Only the values are continuous

TECHNISCHE UNIVERSITÄT DARMSTADT

Least Squares Linear Regression

$$\mathbf{x}_i^{\intercal}\mathbf{w} + w_0 = y_i \quad \forall i = 1, \ldots, n$$

- **Step 1**: Define

$$\hat{\boldsymbol{x}}_i = \begin{pmatrix} \mathbf{x}_i \\ 1 \end{pmatrix} \quad \hat{\mathbf{w}} = \begin{pmatrix} \mathbf{w} \\ w_0 \end{pmatrix}$$

TECHNISCHE UNIVERSITÄT DARMSTADT

Least Squares Linear Regression

$$\mathbf{x}_i^\intercal \mathbf{w} + w_0 = y_i \quad \forall i = 1, \ldots, n$$

- **Step 1**: Define

$$\hat{\boldsymbol{x}}_i = \begin{pmatrix} \mathbf{x}_i \\ 1 \end{pmatrix} \quad \hat{\mathbf{w}} = \begin{pmatrix} \mathbf{w} \\ w_0 \end{pmatrix}$$

- **Step 2**: Rewrite

$$\hat{\mathbf{x}}_i^\intercal \hat{\mathbf{w}} = y_i \quad \forall i = 1, \ldots, n$$

TECHNISCHE UNIVERSITÄT DARMSTADT

Least Squares Linear Regression

$$\mathbf{x}_i^\intercal \mathbf{w} + w_0 = y_i \quad \forall i = 1, \ldots, n$$

- **Step 1**: Define

$$\hat{\boldsymbol{x}}_i = \begin{pmatrix} \mathbf{x}_i \\ 1 \end{pmatrix} \quad \hat{\mathbf{w}} = \begin{pmatrix} \mathbf{w} \\ w_0 \end{pmatrix}$$

- **Step 2**: Rewrite

$$\hat{\mathbf{x}}_i^\intercal \hat{\mathbf{w}} = y_i \quad \forall i = 1, \ldots, n$$

- **Step 3**: Matrix-vector notation

$$\hat{\mathbf{X}}^\intercal \hat{\mathbf{w}} = \mathbf{y}$$

  - where $\hat{\mathbf{X}} = [\hat{\mathbf{x}}_1, \ldots, \hat{\mathbf{x}}_n]$ (each $\hat{\mathbf{x}}_i$ is a vector) and $\mathbf{y} = [y_1, \ldots, y_n]^\intercal$

TECHNISCHE UNIVERSITÄT DARMSTADT

Least Squares Linear Regression

- **Step 4**: Find the least squares solution

$$\hat{\mathbf{w}} = \arg\min_{\mathbf{w}} \left\| \hat{\mathbf{X}}^\intercal \mathbf{w} - \mathbf{y} \right\|^2$$

$$\nabla_{\mathbf{w}} \left\| \hat{\mathbf{X}}^\intercal \mathbf{w} - \mathbf{y} \right\|^2 = \mathbf{0}$$

$$\hat{\mathbf{w}} = \left(\hat{\mathbf{X}} \hat{\mathbf{X}}^\intercal \right)^{-1} \hat{\mathbf{X}} \mathbf{y}$$

- A closed form solution!

TECHNISCHE UNIVERSITÄT DARMSTADT

Least Squares Linear Regression

$$\hat{\mathbf{w}} = \left(\hat{\mathbf{X}}\hat{\mathbf{X}}^\intercal\right)^{-1}\hat{\mathbf{X}}\mathbf{y}$$

- Where is the costly part of this computation?

TECHNISCHE UNIVERSITÄT DARMSTADT

Least Squares Linear Regression

$$\hat{\mathbf{w}} = \left(\hat{\mathbf{X}}\hat{\mathbf{X}}^\intercal\right)^{-1} \hat{\mathbf{X}}\mathbf{y}$$

- Where is the costly part of this computation?
  - The inverse is a $\mathbb{R}^{D \times D}$ matrix
  - Naive inversion takes $O\left(D^3\right)$, but better methods exist
- What can we do if the input dimension D is too large?

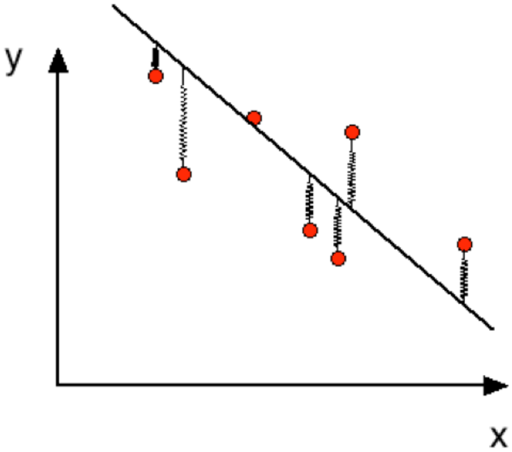
TECHNISCHE UNIVERSITÄT DARMSTADT

Least Squares Linear Regression

$$\hat{\mathbf{w}} = \left(\hat{\mathbf{X}}\hat{\mathbf{X}}^\intercal\right)^{-1}\hat{\mathbf{X}}\mathbf{y}$$

- Where is the costly part of this computation?
  - The inverse is a $\mathbb{R}^{D\times D}$ matrix
  - Naive inversion takes $O\left(D^3\right)$, but better methods exist
- What can we do if the input dimension D is too large?
  - Gradient descent
  - Work with fewer dimensions

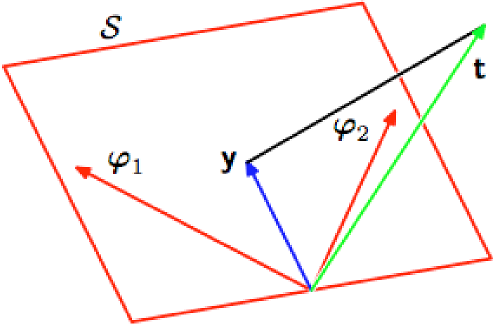
Mechanical Interpretation



TECHNISCHE UNIVERSITÄT DARMSTADT

Geometric Interpretation

- Predicted outputs are Linear Combinations of Features! Samples are projected in this Feature Space



TECHNISCHE UNIVERSITÄT DARMSTADT

Polynomial Regression

- How can we fit arbitrary polynomials using least-squares regression?
  - We introduce a feature transformation as before

$$
\begin{aligned}
y(\mathbf{x}) &= \mathbf{w}^{\mathsf{T}} \phi(\mathbf{x}) \\
&= \sum_{i=0}^{M} w_i \phi_i(\mathbf{x})
\end{aligned}
$$

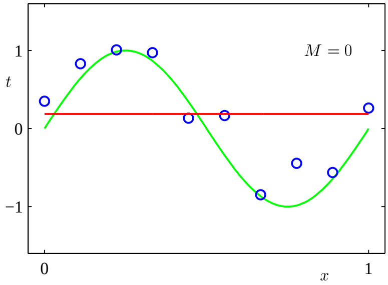
  - Assume $\phi_0(\mathbf{x}) = 1$
  - $\phi_i(.)$ are called the basis functions
  - Still a linear model in the parameters $\mathbf{w}$
- E.g. fitting a cubic polynomial

$$\phi(x) = \left(1, x, x^2, x^3\right)^{\mathsf{T}}$$

TECHNISCHE UNIVERSITÄT DARMSTADT

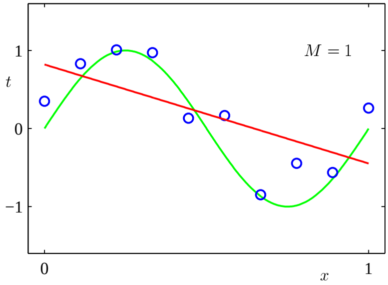
Polynomial Regression

- Polynomial of degree 0 (constant value)



Polynomial Regression

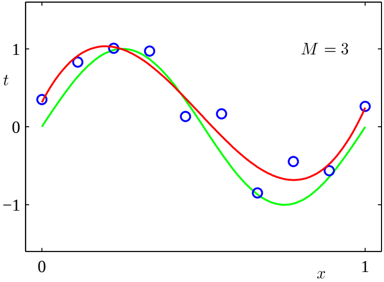
- Polynomial of degree 1 (line)



TECHNISCHE UNIVERSITÄT DARMSTADT

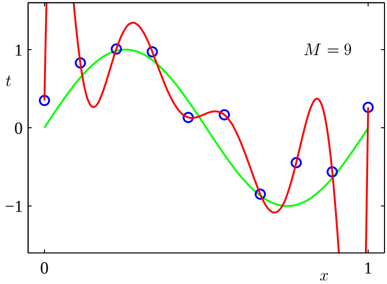
Polynomial Regression

- Polynomial of degree 3 (cubic)



Polynomial Regression

- Polynomial of degree 9



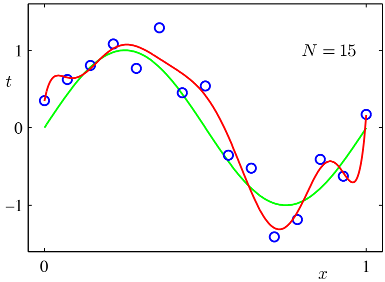
- Massive overfitting

TECHNISCHE UNIVERSITÄT DARMSTADT

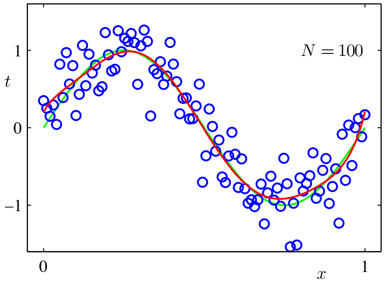
Outline

Overfitting

Relatively little data leads to overfitting



Enough data leads to a good estimate



Probabilistic Regression

- **Assumption 1**: Our target function values are generated by adding noise to the function estimate

$$y = f(\mathbf{x}, \mathbf{w}) + \epsilon$$

  - y - target function value; f - regression function; $\mathbf{x}$ - input value; $\mathbf{w}$ - weights or parameters; ϵ - noise

TECHNISCHE UNIVERSITÄT DARMSTADT

Probabilistic Regression

- **Assumption 1**: Our target function values are generated by adding noise to the function estimate

$$y = f(\mathbf{x}, \mathbf{w}) + \epsilon$$

  - y - target function value; f - regression function; $\mathbf{x}$ - input value; $\mathbf{w}$ - weights or parameters; ϵ - noise

- **Assumption 2**: The noise is a random variable that is Gaussian distributed

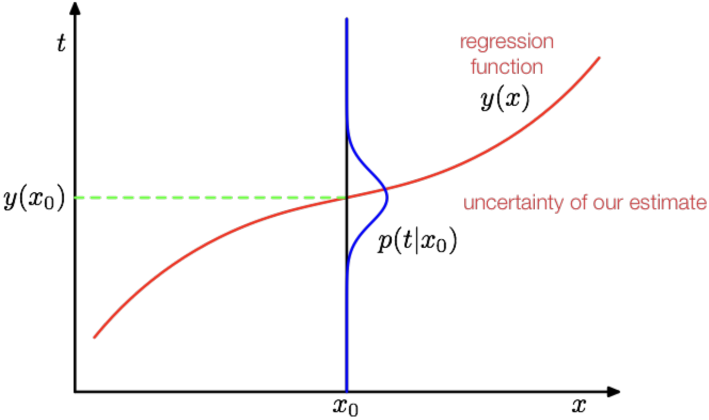
$$\epsilon \sim \mathcal{N}\left(0, \beta^{-1}\right)$$

$$p\left(y \,\middle|\, \mathbf{x}, \mathbf{w}, \beta\right) = \mathcal{N}\left(y \,\middle|\, f(\mathbf{x}, \mathbf{w}), \beta^{-1}\right)$$

  - $f(\mathbf{x}, \mathbf{w})$ is the mean; β^{-1} is the variance (β is the precision)

- Note that y is now a random variable with underlying probability distribution $p\left(y \,\middle|\, \mathbf{x}, \mathbf{w}, \beta\right)$

Probabilistic Regression



Probabilistic Regression

- **Given**
  - Training input data points $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_n] \in \mathbb{R}^{d \times n}$
  - Associated function values $\mathbf{Y} = [y_1, \ldots, y_n]^\intercal$
- **Conditional likelihood** (assuming the data is i.i.d.)

$$p\left(\mathbf{y} \,\middle|\, \mathbf{X}, \mathbf{w}, \beta\right) = \prod_{i=1}^{n} \mathcal{N}\left(y_i \,\middle|\, f\left(\mathbf{x}_i, \mathbf{w}\right), \beta^{-1}\right)$$

(with linear model)

$$= \prod_{i=1}^{n} \mathcal{N}\left(y_i \,\middle|\, \mathbf{w}^\intercal \phi\left(\mathbf{x}_i\right), \beta^{-1}\right)$$

  - $\mathbf{w}^\intercal \phi\left(\mathbf{x}_i\right)$ is the generalized linear regression function
- Maximize the likelihood w.r.t. (with respect to) $\mathbf{w}$ and β

TECHNISCHE UNIVERSITÄT DARMSTADT

Maximum Likelihood Regression

- Simplify using the **log**-likelihood

$$
\begin{aligned}
\log p\left(\mathbf{y} \,\middle|\, \mathbf{X}, \mathbf{w}, \beta\right) &= \sum_{i=1}^{n} \log \mathcal{N}\left(y_i \,\middle|\, \mathbf{w}^\intercal \phi\left(\mathbf{x}_i\right), \beta^{-1}\right) \\
&= \sum_{i=1}^{n} \left[\log\left(\frac{\sqrt{\beta}}{\sqrt{2\pi}}\right) - \frac{\beta}{2}\left(y_i - \mathbf{w}^\intercal \phi\left(\mathbf{x}_i\right)\right)^2\right] \\
&= \frac{n}{2}\log\beta - \frac{n}{2}\log\left(2\pi\right) - \frac{\beta}{2}\sum_{i=1}^{n}\left(y_i - \mathbf{w}^\intercal \phi\left(\mathbf{x}_i\right)\right)^2
\end{aligned}
$$

TECHNISCHE UNIVERSITÄT DARMSTADT

Maximum Likelihood Regression

- Gradient w.r.t. **w**

$$\begin{aligned} \nabla_{\mathbf{w}} \log p\left(\mathbf{y} \middle| \mathbf{X}, \mathbf{w}, \beta\right) &= \mathbf{0} \\ -\beta \sum_{i=1}^{n} \left(y_i - \mathbf{w}^\intercal \phi\left(\mathbf{x}_i\right)\right) \phi\left(\mathbf{x}_i\right) &= \mathbf{0} \end{aligned}$$

- Define

$$\mathbf{y} = \begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix}, \quad \mathbf{w} = \begin{bmatrix} w_1 \\ \vdots \\ w_n \end{bmatrix}, \quad \Phi = \begin{bmatrix} | & & | \\ \phi\left(\mathbf{x}_1\right) & \ldots & \phi\left(\mathbf{x}_n\right) \\ | & & | \end{bmatrix}$$

TECHNISCHE UNIVERSITÄT DARMSTADT

Maximum Likelihood Regression

$$\begin{aligned}
\sum_{i=1}^{n} y_i \phi(\mathbf{x}_i) &= \left[\sum_{i=1}^{n} \phi(\mathbf{x}_i)\,\phi(\mathbf{x}_i)^\intercal\right] \mathbf{w} \\
\Phi \mathbf{y} &= \Phi \Phi^\intercal \mathbf{w} \\
\mathbf{w}_{\mathrm{ML}} &= \left(\Phi \Phi^\intercal\right)^{-1} \Phi \mathbf{y}
\end{aligned}$$

- The same result as in least squares regression!

Maximum Likelihood Regression

- We obtain the same **w** as with least squares regression
    - Least-squares is equivalent to assuming the targets are Gaussian distributed
    - Note: The least squares method is not distribution-free!

TECHNISCHE UNIVERSITÄT DARMSTADT

Maximum Likelihood Regression

- We obtain the same **w** as with least squares regression
  - Least-squares is equivalent to assuming the targets are Gaussian distributed
  - Note: The least squares method is not distribution-free!
- However, the Maximum Likelihood approach is much more powerful!
  - We can also estimate β

$$\beta_{\text{ML}} = \left(\frac{1}{n} \sum_{i=1}^{n} \left(y_i - \mathbf{w}_{\text{ML}}^{\mathsf{T}} \phi\left(\mathbf{x}_i\right) \right)^2 \right)^{-1}$$

  - We can gauge the uncertainty of our estimate!

TECHNISCHE UNIVERSITÄT DARMSTADT

Loss Functions in Regression

- Given a new data point $\mathbf{x}_t$, in least squares regression the function value is $y_t = \hat{\mathbf{x}}_t^\top \hat{\mathbf{w}}$
- But in maximum likelihood regression we have a probability distribution over the function value $p\left(y \,\middle|\, \mathbf{x}, \mathbf{w}, \beta\right)$
- How do we actually estimate a function value y_t for a new data point $\mathbf{x}_t$?

Loss Functions in Regression

- Given a new data point $\mathbf{x}_t$, in least squares regression the function value is $y_t = \hat{\mathbf{x}}_t^\top \hat{\mathbf{w}}$
- But in maximum likelihood regression we have a probability distribution over the function value $p\left(y \middle| \mathbf{x}, \mathbf{w}, \beta\right)$
- How do we actually estimate a function value y_t for a new data point $\mathbf{x}_t$?
- We need a loss function, just as in the classification case

$$\begin{aligned} L : \mathbb{R} \times \mathbb{R} &\rightarrow \mathbb{R}^+ \\ (y_t, f(\mathbf{x}_t)) &\rightarrow L(y_t, f(\mathbf{x}_t)) \end{aligned}$$

Loss Functions in Regression

- Minimize the expected loss

$$\mathbb{E}_{\mathbf{x},y\sim p(\mathbf{x},y)}\left[L\right] = \int\int L\left(y, f\left(\mathbf{x}\right)\right) p\left(\mathbf{x}, y\right) \mathrm{d}\mathbf{x}\mathrm{d}y$$

TECHNISCHE UNIVERSITÄT DARMSTADT

Loss Functions in Regression

- Minimize the expected loss

$$\mathbb{E}_{\mathbf{x},y\sim p(\mathbf{x},y)}\left[L\right] = \int\int L\left(y, f\left(\mathbf{x}\right)\right) p\left(\mathbf{x}, y\right) \mathrm{d}\mathbf{x}\mathrm{d}y$$

- Simplest case: squared loss

$$L\left(y, f\left(\mathbf{x}\right)\right) = \left(y - f\left(\mathbf{x}\right)\right)^2$$

$$\mathbb{E}_{\mathbf{x},y\sim p(\mathbf{x},y)}\left[L\right] = \int\int \left(y - f\left(\mathbf{x}\right)\right)^2 p\left(\mathbf{x}, y\right) \mathrm{d}\mathbf{x}\mathrm{d}y$$

$$\frac{\partial \mathbb{E}\left[L\right]}{\partial f\left(\mathbf{x}\right)} = -2\int \left(y - f\left(\mathbf{x}\right)\right) p\left(\mathbf{x}, y\right) \mathrm{d}y = 0$$

$$\int y p\left(\mathbf{x}, y\right)\mathrm{d}y = f\left(\mathbf{x}\right) \int p\left(\mathbf{x}, y\right) \mathrm{d}y$$

TECHNISCHE UNIVERSITÄT DARMSTADT

Loss Functions in Regression

$$\int y p(\mathbf{x}, y)\, \mathrm{d}y = f(\mathbf{x}) \int p(\mathbf{x}, y)\, \mathrm{d}y$$

$$\int y p(\mathbf{x}, y)\, \mathrm{d}y = f(\mathbf{x})\, p(\mathbf{x})$$

$$f(\mathbf{x}) = \int y \frac{p(\mathbf{x}, y)}{p(\mathbf{x})} \mathrm{d}y = \int y p(y \mid \mathbf{x})\, \mathrm{d}y$$

$$f(\mathbf{x}) = \mathbb{E}_{y \sim p(y|\mathbf{x})}[y] = \mathbb{E}\left[y \middle| \mathbf{x}\right]$$

- Under squared loss, the optimal regression function is the mean $\mathbb{E}\left[y \middle| \mathbf{x}\right]$ of the posterior $p(y \mid \mathbf{x})$
- It is also called mean prediction

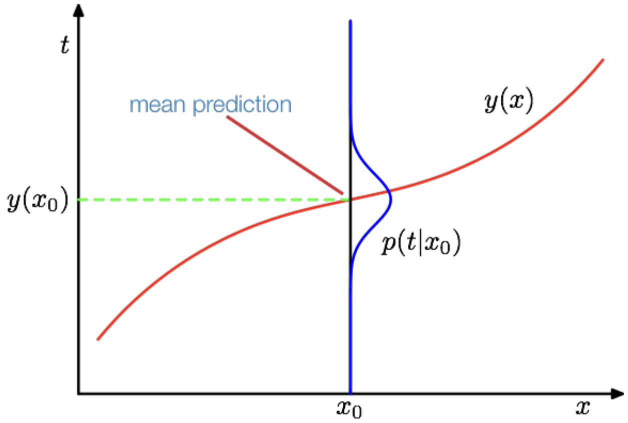
TECHNISCHE UNIVERSITÄT DARMSTADT

Loss Functions in Regression

- For our generalized linear regression function

$$f(\mathbf{x}) = \int y\mathcal{N}\left(y \,\middle|\, \mathbf{w}^{\intercal}\phi(\mathbf{x}), \beta^{-1}\right) \mathrm{d}y = \mathbf{w}^{\intercal}\phi(\mathbf{x})$$

Probabilistic Regression



Outline

Avoiding Overfitting

- Back to our original problem
  - We wanted to avoid overfitting and instabilities
  - Maximum likelihood also leads to overfitting (in the extreme case think if you only had one data point)
- What can we use to counter the problem?

TECHNISCHE UNIVERSITÄT DARMSTADT

Bayesian Linear Regression

- We place a prior on the parameters **w** to tame the instabilities

$$p\left(\mathbf{w}\middle|\mathbf{X},\mathbf{y}\right) \propto p\left(\mathbf{y}\middle|\mathbf{X},\mathbf{w}\right) p\left(\mathbf{w}\right)$$

  - Parameter prior: $p\left(\mathbf{w}\right)$
  - Likelihood of targets under the data and parameters (as before): $p\left(\mathbf{y}\middle|\mathbf{X},\mathbf{w}\right)$
  - Posterior over the parameters: $p\left(\mathbf{w}\middle|\mathbf{X},\mathbf{y}\right)$

TECHNISCHE UNIVERSITÄT DARMSTADT

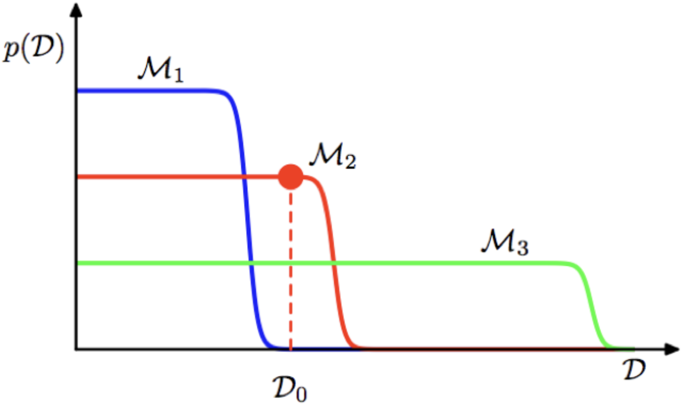
Bayesian Linear Regression

- We place a prior on the parameters **w** to tame the instabilities

$$p\left(\mathbf{w}\,\middle|\,\mathbf{X},\mathbf{y}\right) \propto p\left(\mathbf{y}\,\middle|\,\mathbf{X},\mathbf{w}\right)p\left(\mathbf{w}\right)$$

  - Parameter prior: $p\left(\mathbf{w}\right)$
  - Likelihood of targets under the data and parameters (as before): $p\left(\mathbf{y}\,\middle|\,\mathbf{X},\mathbf{w}\right)$
  - Posterior over the parameters: $p\left(\mathbf{w}\,\middle|\,\mathbf{X},\mathbf{y}\right)$
- Notice the VERY important difference: in this setting, you do not get anymore a single value for the parameters, but rather a probability distribution over the parameters

Basic Idea: Prior controls the Model Class and hence what Data Sets can be explained



Bayesian Regression

- Simple idea: Put a Gaussian prior on **w**
- It will put a “soft” limit on the coefficients and thus avoid instabilities

$$\mathbf{w} \sim p\left(\mathbf{w} \middle| \alpha\right) = \mathcal{N}\left(\mathbf{w} \middle| \mathbf{0}, \alpha^{-1}\mathbf{I}\right)$$

  - We use a zero mean Gaussian to keep the derivation compact, but you can use another mean
- Zero mean and spherical covariance (given by the diagonal covariance matrix)

TECHNISCHE UNIVERSITÄT DARMSTADT

Bayesian Regression

- Simple idea: Put a Gaussian prior on **w**
- It will put a “soft” limit on the coefficients and thus avoid instabilities

$$\mathbf{w} \sim p\left(\mathbf{w} \middle| \alpha\right) = \mathcal{N}\left(\mathbf{w} \middle| \mathbf{0}, \alpha^{-1}\mathbf{I}\right)$$

  - We use a zero mean Gaussian to keep the derivation compact, but you can use another mean
- Zero mean and spherical covariance (given by the diagonal covariance matrix)
- The posterior becomes

$$\begin{aligned} p\left(\mathbf{w} \middle| \mathbf{X}, \mathbf{y}, \alpha, \beta\right) &\propto p\left(\mathbf{y} \middle| \mathbf{X}, \mathbf{w}, \beta\right) p\left(\mathbf{w} \middle| \alpha\right) \\ &\propto p\left(\mathbf{y} \middle| \mathbf{X}, \mathbf{w}, \beta\right) \mathcal{N}\left(\mathbf{w} \middle| \mathbf{0}, \alpha^{-1}\mathbf{I}\right) \end{aligned}$$

Maximum A-Posteriori (MAP)

- First attempt to solve this problem: estimate **w** by maximizing the (log) posterior

$$
\begin{aligned}
\log p\left(\mathbf{w} \,\middle|\, \mathbf{X}, \mathbf{y}, \alpha, \beta\right) &= \log p\left(\mathbf{y} \,\middle|\, \mathbf{X}, \mathbf{w}, \beta\right) + \log \mathcal{N}\left(\mathbf{w} \,\middle|\, \mathbf{0}, \alpha^{-1}\mathbf{I}\right) + \text{const} \\
&= \sum_{i=1}^{n} \log \mathcal{N}\left(y_i \,\middle|\, \mathbf{w}^\intercal \phi\left(\mathbf{x}_i\right), \beta^{-1}\right) \\
&\quad + \log \mathcal{N}\left(\mathbf{w} \,\middle|\, \mathbf{0}, \alpha^{-1}\mathbf{I}\right) + \text{const} \\
&= -\frac{\beta}{2} \sum_{i=1}^{n} \left(y_i - \mathbf{w}^\intercal \phi\left(\mathbf{x}_i\right)\right)^2 - \frac{\alpha}{2} \mathbf{w}^\intercal \mathbf{w} + \text{const}
\end{aligned}
$$

TECHNISCHE UNIVERSITÄT DARMSTADT

Maximum A-Posteriori (MAP)

$$\nabla_{\mathbf{w}} \log p\left(\mathbf{w} \,\middle|\, \mathbf{X}, \mathbf{y}, \alpha, \beta\right) = \beta \sum_{i=1}^{n} \left(y_i - \mathbf{w}^\intercal \phi\left(\mathbf{x}_i\right)\right) \phi\left(\mathbf{x}_i\right) - \alpha \mathbf{w} = \mathbf{0}$$

$$\beta \sum_{i=1}^{n} y_i \phi\left(\mathbf{x}\right) = \beta \left[\sum_{i=1}^{n} \phi\left(\mathbf{x}_i\right) \phi\left(\mathbf{x}_i\right)^\intercal\right] \mathbf{w} + \alpha \mathbf{w}$$

$$\beta \sum_{i=1}^{n} y_i \phi\left(\mathbf{x}\right) = \beta \left[\sum_{i=1}^{n} \phi\left(\mathbf{x}_i\right) \phi\left(\mathbf{x}_i\right)^\intercal + \alpha\right] \mathbf{w}$$

$$\beta \Phi \mathbf{y} = \left(\beta \Phi \Phi^\intercal + \alpha \mathbf{I}\right) \mathbf{w}$$

$$\mathbf{w}_{\mathrm{MAP}} = \left(\Phi \Phi^\intercal + \frac{\alpha}{\beta} \mathbf{I}\right)^{-1} \Phi \mathbf{y}$$

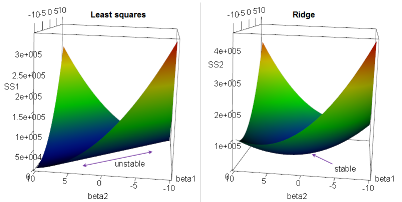
- What is the role of α / β in the expression?

Maximum A-Posteriori (MAP)

$$\mathbf{w}_{\text{MAP}} = \left(\Phi\Phi^{\mathsf{T}} + \frac{\alpha}{\beta}\mathbf{I} \right)^{-1} \Phi\mathbf{y}$$

- The prior has the effect that it regularizes the pseudo-inverse
- Also called ridge regression

Intuition for the term "ridge", although these are not the historical reasons : If there is multi-collinearity, we get a "ridge" in the likelihood function. This in turn yields a long "valley" in the RSS. Ridge regression "fixes" the ridge. It adds a penalty that turns the ridge into a nice peak in likelihood space.



Maximum A-Posteriori (MAP) vs Regularized Least-squares Linear Regression

- There is another way to look at the MAP result
- Let us add a regularization term to our objective from Least-squares Linear Regression

$$\hat{\mathbf{w}} = \arg\min_{\mathbf{w}} \frac{1}{2} \left\| \hat{\mathbf{X}}^{\mathsf{T}} \mathbf{w} - \mathbf{y} \right\|^2 + \frac{\lambda}{2} \|\mathbf{w}\|^2$$

- Solving for **w** we get a new estimate

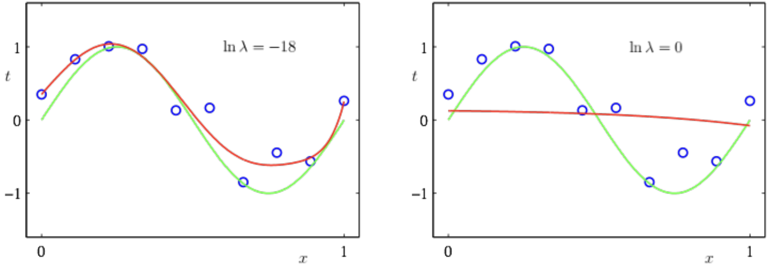
$$\hat{\mathbf{w}} = \left(\hat{\mathbf{X}} \hat{\mathbf{X}}^{\mathsf{T}} + \lambda \mathbf{I} \right)^{-1} \hat{\mathbf{X}} \mathbf{y}$$

  - where $\lambda = \alpha / \beta$

- When you place a regularizer λ in least-squares linear regression, you are assuming the targets have Gaussian distributed noise, but also that your parameters are Gaussian distributed

Bayesian Regression

- Polynomial of degree 9 with prior on **w**



- $\lambda = \alpha/\beta$ controls the complexity of the model and determines the degree of overfitting

Bayesian Regression

Table of the coefficients $\mathbf{w}^\star$ for $M = 9$ polynomials with various values for the regularization parameter λ. Note that $\ln \lambda = -\infty$ corresponds to a model with no regularization, i.e., to the graph at the bottom right in Figure 1.4. We see that, as the value of λ increases, the typical magnitude of the coefficients gets smaller.

	$\ln \lambda = -\infty$	$\ln \lambda = -18$	$\ln \lambda = 0$
$w_0^\star$	0.35	0.35	0.13
$w_1^\star$	232.37	4.74	-0.05
$w_2^\star$	-5321.83	-0.77	-0.06
$w_3^\star$	48568.31	-31.97	-0.05
$w_4^\star$	-231639.30	-3.89	-0.03
$w_5^\star$	640042.26	55.28	-0.02
$w_6^\star$	-1061800.52	41.32	-0.01
$w_7^\star$	1042400.18	-45.95	-0.00
$w_8^\star$	-557682.99	-91.53	0.00
$w_9^\star$	125201.43	72.68	0.01

[Bishop]

TECHNISCHE UNIVERSITÄT DARMSTADT

Full Bayesian Regression

- We can go further than MAP estimation
- Observation: We do not actually need to know $\mathbf{w}$, all we want to do is to predict a function value based on the training data
- Idea: "Remove" $\mathbf{w}$ by marginalizing over it

$$p\left(y_t \,\middle|\, \mathbf{x}_t, \mathbf{X}, \mathbf{y}\right) = \int p\left(y_t, \mathbf{w} \,\middle|\, \mathbf{x}_t, \mathbf{X}, \mathbf{y}\right) \mathrm{d}\mathbf{w}$$

  - y_t - predicted value; $\mathbf{x}_t$ - test input; $\mathbf{X}$ - training data points; $\mathbf{y}$ - training function values

TECHNISCHE UNIVERSITÄT DARMSTADT

Full Bayesian Regression

$$\underbrace{p\left(y_t \,\middle|\, \mathbf{x}_t, \mathbf{X}, \mathbf{y}\right)}_{\text{predictive distribution}} = \int p\left(y_t, \mathbf{w} \,\middle|\, \mathbf{x}_t, \mathbf{X}, \mathbf{y}\right) \mathrm{d}\mathbf{w}$$

$$= \int p\left(y_t \,\middle|\, \mathbf{w}, \mathbf{x}_t, \mathbf{X}, \mathbf{y}\right) p\left(\mathbf{w} \,\middle|\, \mathbf{x}_t, \mathbf{X}, \mathbf{y}\right) \mathrm{d}\mathbf{w}$$

$$= \int \underbrace{p\left(y_t \,\middle|\, \mathbf{w}, \mathbf{x}_t\right)}_{\text{regression model}} \underbrace{p\left(\mathbf{w} \,\middle|\, \mathbf{X}, \mathbf{y}\right)}_{\text{posterior distribution}} \mathrm{d}\mathbf{w}$$

- For Gaussian distributions, this can be done in closed form, leading to so-called Gaussian Processes

TECHNISCHE UNIVERSITÄT DARMSTADT

Full Bayesian Regression

- We can also do that in closed form: integrate out all possible parameters

$$p\left(y_* \,\middle|\, \mathbf{x}_*, \mathbf{X}, \mathbf{y}\right) = \int \underbrace{p\left(y_* \,\middle|\, \mathbf{x}_*, \theta\right)}_{\text{likelihood}} \underbrace{p\left(\theta \,\middle|\, \mathbf{X}, \mathbf{y}\right)}_{\text{parameter posterior}} \, \mathrm{d}\theta$$

  - y_* - predicted value; $\mathbf{x}_*$ - test input; $\mathbf{X}, \mathbf{y}$ - training data

TECHNISCHE UNIVERSITÄT DARMSTADT

Full Bayesian Regression

- We can also do that in closed form: integrate out all possible parameters

$$p\left(y_* \,\middle|\, \mathbf{x}_*, \mathbf{X}, \mathbf{y}\right) = \int \underbrace{p\left(y_* \,\middle|\, \mathbf{x}_*, \theta\right)}_{\text{likelihood}} \underbrace{p\left(\theta \,\middle|\, \mathbf{X}, \mathbf{y}\right)}_{\text{parameter posterior}} \, \mathrm{d}\theta$$

  - y_* - predicted value; $\mathbf{x}_*$ - test input; $\mathbf{X}, \mathbf{y}$ - training data

- The predictive distribution is again a Gaussian

$$p\left(y_* \,\middle|\, \mathbf{x}_*, \mathbf{X}, \mathbf{y}\right) = \mathcal{N}\left(\mathbf{y}_* \,\middle|\, \mu\left(\mathbf{x}_*\right), \sigma^2\left(\mathbf{x}_*\right)\right)$$

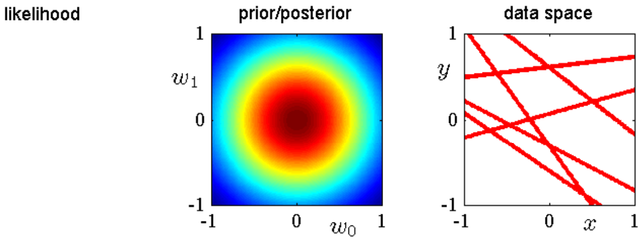
$$\mu\left(\mathbf{x}_*\right) = \phi^T\left(\mathbf{x}_*\right)\left(\frac{\alpha}{\beta}\mathbf{I} + \Phi\Phi^\mathsf{T}\right)^{-1} \Phi^\mathsf{T}\mathbf{y}$$

$$\sigma^2\left(\mathbf{x}_*\right) = \frac{1}{\beta} + \phi^\mathsf{T}\left(\mathbf{x}_*\right)\left(\alpha\mathbf{I} + \beta\Phi\Phi^\mathsf{T}\right)^{-1}\phi\left(\mathbf{x}_*\right)$$

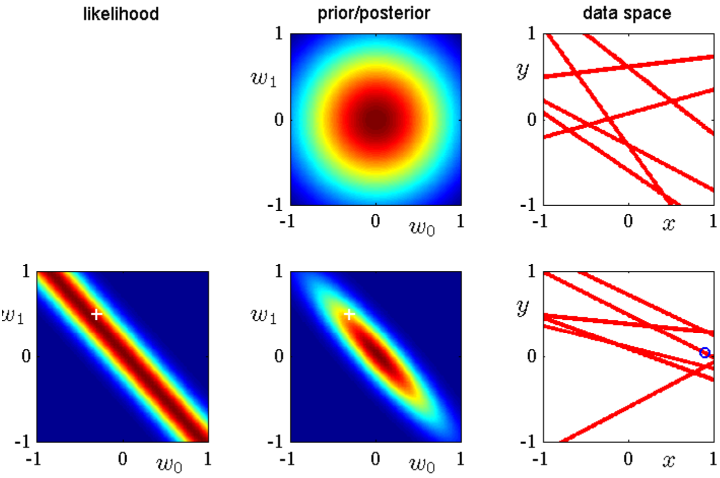
- The variance is state dependent

TECHNISCHE UNIVERSITÄT DARMSTADT

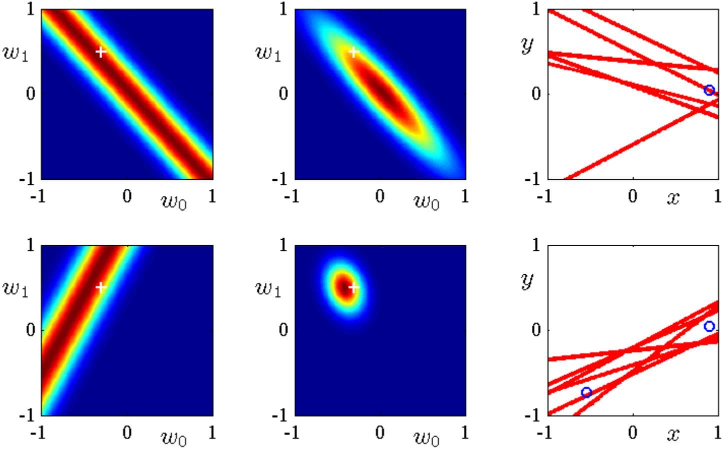
Bayesian (Linear) Regression



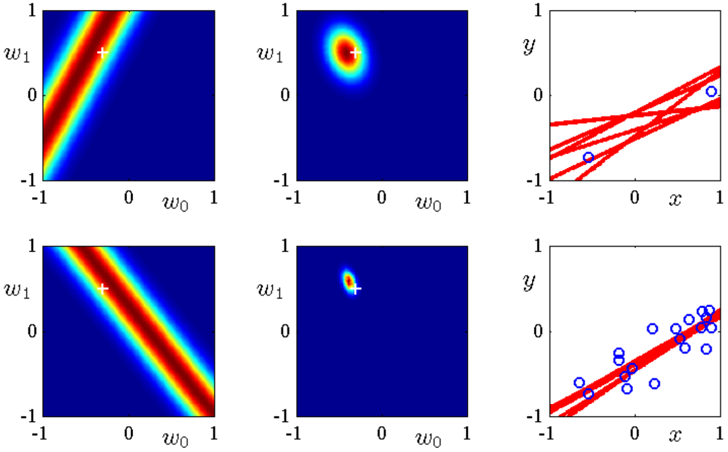
Bayesian (Linear) Regression



Bayesian (Linear) Regression

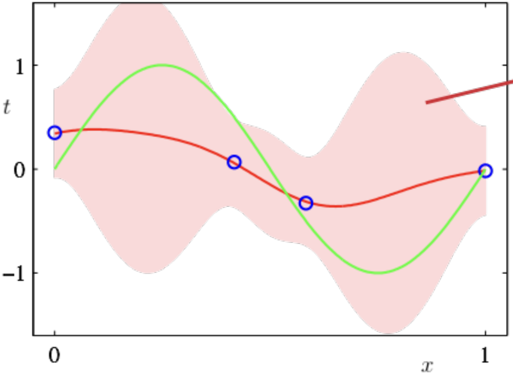


Bayesian (Linear) Regression



Gaussian Processes - Quick Preview

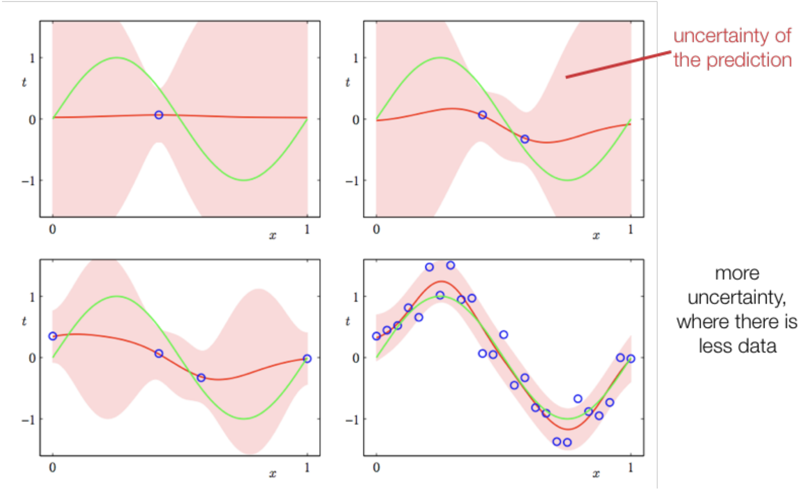
- Essentially Kernelized Bayesian Ridge Regression is equivalent to Gaussian Processes. We will not cover them now, but here is a quick preview of what they can do



uncertainty of the prediction

more uncertainty, where
there is less data

Gaussian Processes - Quick Preview



TECHNISCHE UNIVERSITÄT DARMSTADT

Outline

4. Wrap-Up

You know now:

- How to formulate a linear regression problem
- The different methods to perform linear regression: least-squares, maximum likelihood and bayesian
- Derive the equations for the parameters using the different methods
- Why introducing a prior distribution over the parameters can combat overfitting

TECHNISCHE UNIVERSITÄT DARMSTADT

Self-Test Questions

- What is regression (in general) and linear regression (in particular)?
- What is the cost function of regression and how can I interpret it?
- What is overfitting?
- How can I derive a Maximum-Likelihood Estimator for Regression?
- Why are Bayesian methods important?
- What is MAP and how is it different to full Bayesian regression?

TECHNISCHE UNIVERSITÄT DARMSTADT

Homework

- Reading Assignment for next lecture
  - Murphy ch. 8
  - Bishop ch. 4