TECHNISCHE UNIVERSITÄT
KAISERSLAUTERN

8.3 Non-linear Regression

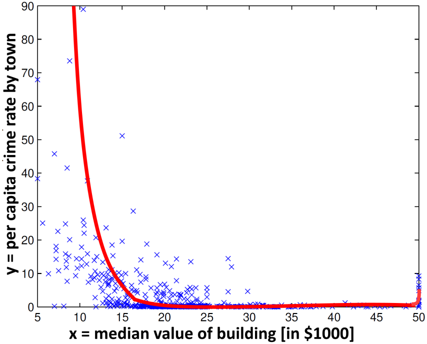
Machine Learning 1: Foundations

Marius Kloft *(TUK)*

1. Linear Regression
2. LOOCV
3. Non-linear Regression
   - Kernel Ridge Regression
   - Deep Regression
4. Unifying Loss View of Regression and Classification
   - Kernel Ridge Regression
   - Deep Regression

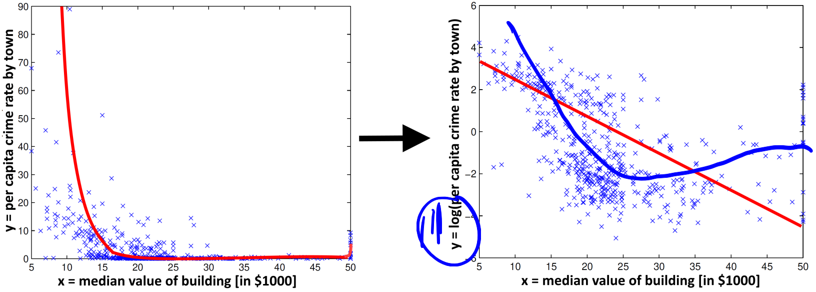
Non-linear Regression

Motivation:

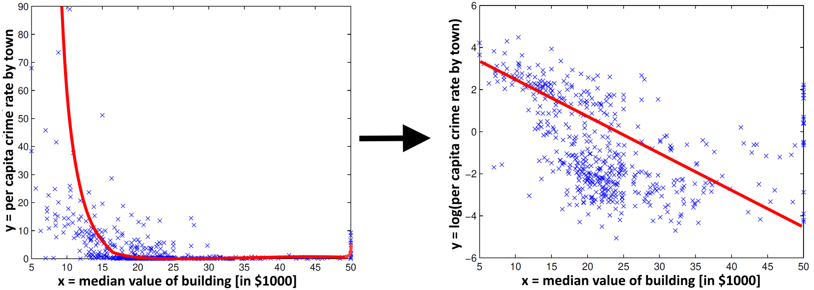


Linear regression not adequate if ground truth is non-linear

Trick: Apply a Log Transformation to the Label

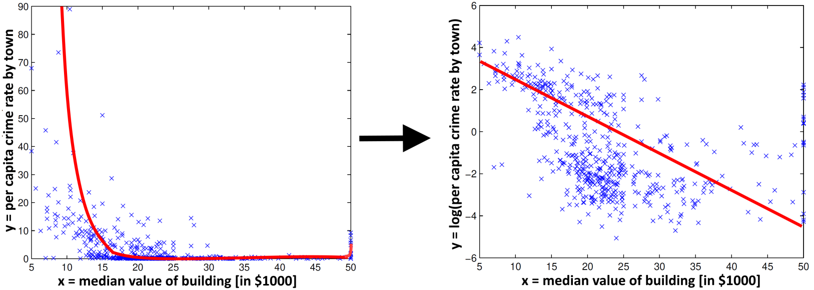


Trick: Apply a Log Transformation to the Label



The fit looks better now...

Trick: Apply a Log Transformation to the Label



The fit looks better now...

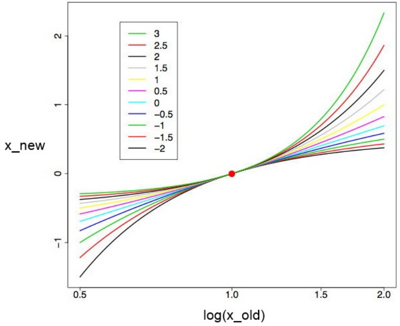
... but is still not optimal.

Trick: Box-Cox Transformation

Definition

The **Box-Cox transformation** (or 'power transformation') with parameter λ is:

$$x_{\text{new}} \leftarrow \begin{cases} \log(x_{\text{old}}) & \text{if } \lambda = 0 \\ \frac{x_{\text{old}}^{\lambda} - 1}{\lambda} & \text{if } \lambda \neq 0 \end{cases}$$

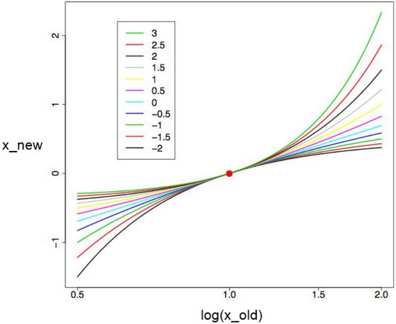


Trick: Box-Cox Transformation

Definition

The **Box-Cox transformation** (or 'power transformation') with parameter λ is:

$$x_{\text{new}} \leftarrow \begin{cases} \log(x_{\text{old}}) & \text{if } \lambda = 0 \\ \frac{x_{\text{old}}^{\lambda} - 1}{\lambda} & \text{if } \lambda \neq 0 \end{cases}$$



Interpretation:

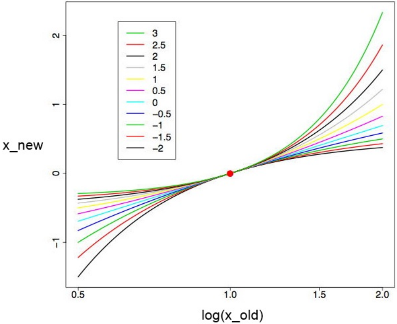
$$\text{for} \begin{cases} \lambda > 1: & \text{data is „stretched"} \quad (\text{e.g. } \lambda = 2\text{: quadratically}) \\ 0 < \lambda < 1: & \text{data is concentrated (e.g. } \lambda = 0.5\text{: square root)} \\ \lambda = 0: & \text{log transform} \\ \lambda < 0: & \text{analogously, with the order of data reversed} \end{cases}$$

Trick: Box-Cox Transformation

Definition

The **Box-Cox transformation** (or 'power transformation') with parameter λ is:

$$x_{\text{new}} \leftarrow \begin{cases} \log(x_{\text{old}}) & \text{if } \lambda = 0 \\ \frac{x_{\text{old}}^{\lambda} - 1}{\lambda} & \text{if } \lambda \neq 0 \end{cases}$$

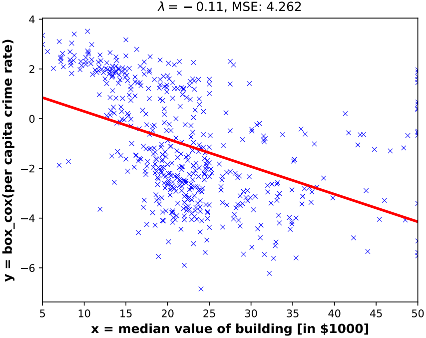


Interpretation:

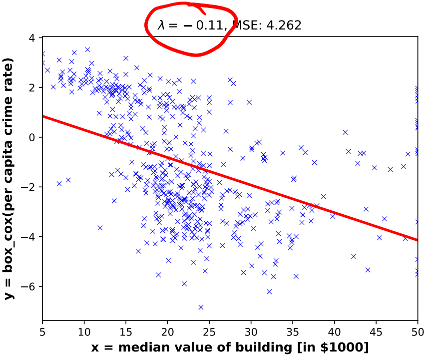
$$\text{for} \begin{cases} \lambda > 1: & \text{data is stretched (e.g. } \lambda = 2\text{: quadratically)} \\ 0 < \lambda < 1: & \text{data is concentrated (e.g. } \lambda = 0.5\text{: square root)} \\ \lambda = 0: & \text{log transform} \\ \lambda < 0: & \text{analogously, with the order of data reversed} \end{cases}$$

The optimal parameter λ can be chosen automatically (omitted)

Our Data After the Box-Cox Transformation:



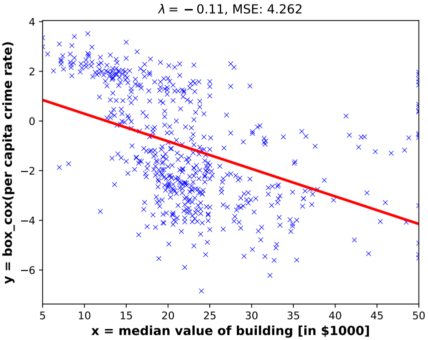
Our Data After the Box-Cox Transformation:



Rule of thumb:

- Always apply Box-Cox transformation to the labels
- If the number of features is small, you can applying it also to the features

Our Data After the Box-Cox Transformation:



Rule of thumb:

- Always apply Box-Cox transformation to the labels
- If the number of features is small, you can applying it also to the features

If this still not helps or the number of features is high, we need a **non-linear regression** method.

Non-linear Regression

We will discuss two powerful **non-linear** regression methods:

1. Kernel ridge regression
2. Deep regression

Kernel Ridge Regression (KRR)

KRR is just the kernelized version of linear regression:

Definition

Let k be a kernel with associated feature map $\phi : \mathbb{R}^d \to \mathcal{H}$, i.e., $k(\mathbf{x}, \tilde{\mathbf{x}}) = \langle \phi(\mathbf{x}), \phi(\tilde{\mathbf{x}}) \rangle$. Then **kernel ridge regression (KRR)** is defined as:

$$\mathbf{w}_{KRR} := \underset{\mathbf{w} \in \mathcal{H}}{\arg\min} \quad \frac{1}{2} \|\mathbf{w}\|^2 + C \sum_{i=1}^{n} \|y_i - \langle \mathbf{w}, \underline{\phi}(x_i) \rangle \|^2.$$

Kernel Ridge Regression (KRR)

KRR is just the kernelized version of linear regression:

Definition

Let k be a kernel with associated feature map $\phi : \mathbb{R}^d \to \mathcal{H}$, i.e., $k(\mathbf{x}, \tilde{\mathbf{x}}) = \langle \phi(\mathbf{x}), \phi(\tilde{\mathbf{x}}) \rangle$. Then **kernel ridge regression (KRR)** is defined as:

$$\mathbf{w}_{KRR} := \operatorname*{arg\,min}_{\mathbf{w} \in \mathcal{H}} \ \frac{1}{2} \|\mathbf{w}\|^2 + C \sum_{i=1}^{n} \| y_i - \langle \mathbf{w}, \phi(x_i) \rangle \|^2 .$$

Recall the kernel trick:

- re-formulate the problem in terms of inner products between data points
- replace inner products by kernel

To Formulate KRR in Terms of Inner Products...

... we apply the representer theorem (see kernel lecture):

$$\exists \boldsymbol{\alpha} \in \mathbb{R}^n : \quad \mathbf{w}_{KRR} = \sum_{i=1}^{n} \alpha_i \phi(\mathbf{x}_i) = \phi(X)\boldsymbol{\alpha} ,$$

where we employ the notation:

- $\phi(X) := (\phi(\mathbf{x}_1), \ldots, \phi(\mathbf{x}_n))$

Derivation of KRR

Using the representer theorem, we can write:

$$\min_{\mathbf{w} \in \mathbb{R}^d} \quad \frac{1}{2}\|\mathbf{w}\|^2 + C\left\|\mathbf{y} - \phi(X)^\top \mathbf{w}\right\|^2$$

Derivation of KRR

Using the representer theorem, we can write:

$$
\begin{aligned}
&\min_{\mathbf{w} \in \mathbb{R}^d} \quad \frac{1}{2}\|\mathbf{w}\|^2 + C\left\|\mathbf{y} - \phi(X)^\top \mathbf{w}\right\|^2 \\
= &\min_{\boldsymbol{\alpha} \in \mathbb{R}^n} \quad \frac{1}{2} \underbrace{\|\phi(X)\boldsymbol{\alpha}\|^2}_{\boldsymbol{\alpha}^\top \underbrace{\phi(X)^\top \phi(X)}_{=K} \boldsymbol{\alpha}} + C\left\|\mathbf{y} - \underbrace{\phi(X)^\top \phi(X)}_{=K} \boldsymbol{\alpha}\right\|^2
\end{aligned}
$$

Derivation of KRR

Using the representer theorem, we can write:

$$
\begin{aligned}
&\min_{\mathbf{w} \in \mathbb{R}^d} \frac{1}{2}\|\mathbf{w}\|^2 + C\left\|\mathbf{y} - \phi(X)^\top \mathbf{w}\right\|^2 \\
= &\min_{\boldsymbol{\alpha} \in \mathbb{R}^n} \frac{1}{2} \underbrace{\|\phi(X)\boldsymbol{\alpha}\|^2}_{\boldsymbol{\alpha}^\top \underbrace{\phi(X)^\top \phi(X)}_{=K} \boldsymbol{\alpha}} + C\left\|\mathbf{y} - \underbrace{\phi(X)^\top \phi(X)}_{=K} \boldsymbol{\alpha}\right\|^2 \\
= &\min_{\boldsymbol{\alpha} \in \mathbb{R}^n} \frac{1}{2}\boldsymbol{\alpha}^\top K \boldsymbol{\alpha} + C\left\|\mathbf{y} - K\boldsymbol{\alpha}\right\|^2
\end{aligned}
$$

Derivation of KRR

Using the representer theorem, we can write:

$$\begin{aligned}
&\min_{\mathbf{w} \in \mathbb{R}^d} \frac{1}{2}\|\mathbf{w}\|^2 + C\left\|\mathbf{y} - \phi(X)^\top \mathbf{w}\right\|^2 \\
= &\min_{\boldsymbol{\alpha} \in \mathbb{R}^n} \frac{1}{2} \underbrace{\|\phi(X)\boldsymbol{\alpha}\|^2}_{\boldsymbol{\alpha}^\top \underbrace{\phi(X)^\top \phi(X)}_{=K} \boldsymbol{\alpha}} + C\left\|\mathbf{y} - \underbrace{\phi(X)^\top \phi(X)}_{=K} \boldsymbol{\alpha}\right\|^2 \\
= &\min_{\boldsymbol{\alpha} \in \mathbb{R}^n} \frac{1}{2}\boldsymbol{\alpha}^\top K \boldsymbol{\alpha} + C\left\|\mathbf{y} - K\boldsymbol{\alpha}\right\|^2
\end{aligned}$$

One can show (homework) that the optimal solution $\boldsymbol{\alpha}^*$ is:

$$\boxed{\boldsymbol{\alpha}^* = \left(K + \tfrac{1}{2C} I_{n \times n}\right)^{-1} y}$$

KRR Solution

Theorem

The solution of KRR is given by

$$f(\mathbf{x}) = \sum_{i=1}^{n} \alpha_i k(\mathbf{x}_i, \mathbf{x}) \quad \text{with} \quad \boldsymbol{\alpha} = \left(K + \frac{1}{2C} I_{n \times n}\right)^{-1} y.$$

It can thus be computed in $O(n^3)$.

KRR Solution

Theorem

The solution of KRR is given by

$$f(\mathbf{x}) = \sum_{i=1}^{n} \alpha_i k(\mathbf{x}_i, \mathbf{x}) \quad \text{with} \quad \boldsymbol{\alpha} = \left(K + \frac{1}{2C} I_{n \times n}\right)^{-1} y.$$

It can thus be computed in $O(n^3)$.

Usually we use KRR together with a Gaussian kernel,

KRR Solution

Theorem

The solution of KRR is given by

$$f(\mathbf{x}) = \sum_{i=1}^{n} \alpha_i k(\mathbf{x}_i, \mathbf{x}) \quad \text{with} \quad \boldsymbol{\alpha} = \left(K + \frac{1}{2C} I_{n\times n}\right)^{-1} y.$$

It can thus be computed in $O(n^3)$.

Usually we use KRR together with a Gaussian kernel,
but sometimes it can make sense to use a linear kernel:

- KRR with linear kernel is the same as LS regression (LSR)
- but KRR's complexity is $O(n^3)$, while LSR's is $O(d^3)$
  - advantageous when $n < d$.

KRR Solution

Theorem

The solution of KRR is given by

$$f(\mathbf{x}) = \sum_{i=1}^{n} \alpha_i k(\mathbf{x}_i, \mathbf{x}) \quad \text{with} \quad \boldsymbol{\alpha} = \left(K + \frac{1}{2C} I_{n \times n}\right)^{-1} y.$$

It can thus be computed in $O(n^3)$.

Usually we use KRR together with a Gaussian kernel,
but sometimes it can make sense to use a linear kernel:

- KRR with linear kernel is the same as LS regression (LSR)
- but KRR's complexity is $O(n^3)$, while LSR's is $O(d^3)$
  - advantageous when $n < d$.

Note that the LOOCV trick can be adapted to KRR

- thus we can compute the LOOCV RMSE of KRR in $O(n^3)$

Now Say In Addition to Our Housing Data ...

... we are given images of the houses:

Now Say In Addition to Our Housing Data ...

... we are given images of the houses:

How to estimate the value of such a house from the image
(and maybe additional other features)?

Deep Regression

State of the art in image processing: deep CNNs

How can we use ANNs (such as deep CNNs) for regression?

Deep Regression

State of the art in image processing: deep CNNs

How can we use ANNs (such as deep CNNs) for regression?

Just take our classification ANN and change the loss to least squares:

Deep Regression (DR)

Predict $f(\mathbf{x}) = \mathbf{w}_*^\top \phi_{W_*}(\mathbf{x})$, where:

$$(\mathbf{w}_*, W_*) := \min_{\mathbf{w}, W} \frac{1}{2} \|\mathbf{w}\|^2 + \sum_{l=1}^{L} \|W_l\|_{\mathrm{Fro}}^2 + C \sum_{i=1}^{n} \left(y_i - \mathbf{w}^\top \phi_W(\mathbf{x}_i) \right)^2$$