Regression

Sibylle Hess

Technische Universiteit Eindhoven University of Technology

1

Informal Problem Description

Example for Regression: Prediction of House Prices

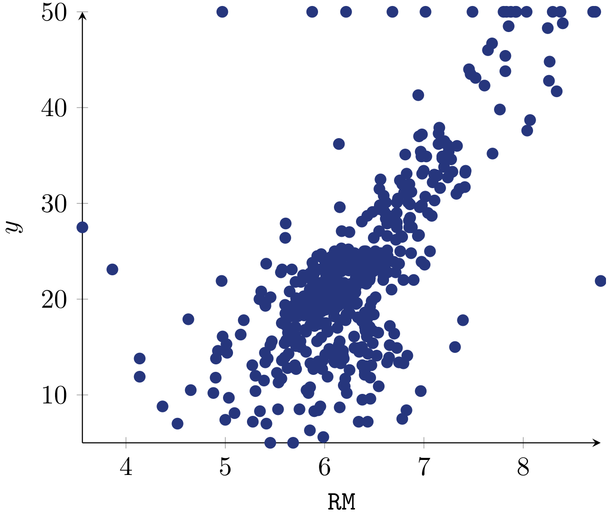
The Boston House Prices Dataset

The Boston house prices dataset has 12 features and a target variable y describing the average price in a neighborhood in 1000\$.

ID	`RM`	`LSTAT`	...	y
1	6.5	4.98	...	24.0
2	6.4	9.14	...	21.6
3	7.2	4.03	...	34.7
⋮	⋮	⋮	⋮	⋮

Particularly relevant for prediction are the features `RM`, denoting the average number of rooms in houses in a neighborhood and `LSTAT`, describing the percentage of homeowners considered as *lower class*.

How can we Predict the Price of a House?



The Data Representation for Regression Problems

ID	F_1	F_2	...	F_d	y
1	D_{11}	D_{12}	...	D_{1d}	y_1
2	D_{21}	D_{22}	...	D_{2d}	y_2
$\vdots$	$\vdots$	$\vdots$	$\vdots$		$\vdots$
n	D_{n1}	D_{n2}	...	D_{nd}	y_n

The goal is to predict target y given a feature vector $\mathbf{x}$ by means of a function

$$f(\mathbf{x}) \approx y$$

2

Derive the Formal Problem Definition

Formalizing the Regression Task

Given a dataset consisting of n observations

$$\mathcal{D} = \left\{(D_{i\cdot}, y_i) | D_{i\cdot} \in \mathbb{R}^{1\times d}, y_i \in \mathbb{R}, 1 \leq i \leq n\right\}$$

Find $f : \mathbb{R}^d \rightarrow \mathbb{R}$, $f \in \mathcal{F}$ such that $f(D_{i\cdot}^\top) \approx y_i$ for all $1 \leq i \leq n$

The underlying assumption is that every observation $(D_{i\cdot}, y_i)$ is generated by the true model function f^* and noise:

$$y_i = f^*(D_{i\cdot}^\top) + \epsilon_i$$

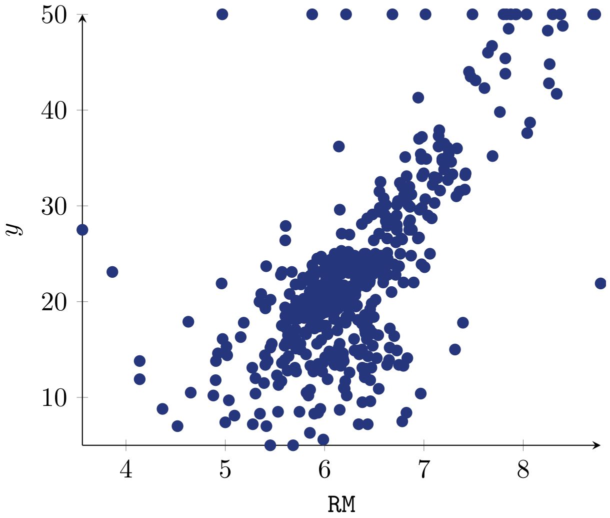
Okay, so regression is to find a function which fits the (noisy) function values we know from the data.

Two questions arise:

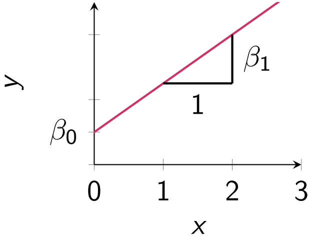
What kind of functions are we looking for?

What does fit actually mean?

Function Families



Affine Functions in Two Dimensions (d=1)

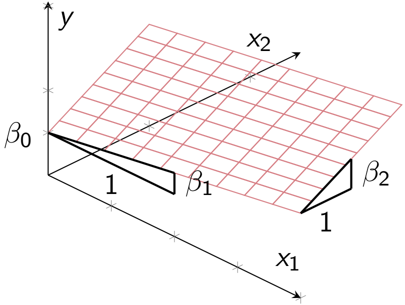


$f : \mathbb{R} \to \mathbb{R}$

$$
\begin{aligned}
f(x) &= \beta_1 x + \beta_0 \\
&= \begin{pmatrix} 1 & x \end{pmatrix} \begin{pmatrix} \beta_0 \\ \beta_1 \end{pmatrix} \\
&= \phi(x)^\top \boldsymbol{\beta} \qquad \text{(inner product)}
\end{aligned}
$$

where $\phi(x) = \begin{pmatrix} 1 \\ x \end{pmatrix}$, $\boldsymbol{\beta} \in \mathbb{R}^2$

Affine Functions in Three Dimensions (d=2)



$$f : \mathbb{R}^2 \to \mathbb{R}$$

$$\begin{aligned} f(\mathbf{x}) &= \beta_2 x_2 + \beta_1 x_1 + \beta_0 \\ &= \begin{pmatrix} 1 & x_1 & x_2 \end{pmatrix} \begin{pmatrix} \beta_0 \\ \beta_1 \\ \beta_2 \end{pmatrix} \\ &= \phi(\mathbf{x})^\top \boldsymbol{\beta}, \text{ where} \end{aligned}$$

$$\phi(\mathbf{x}) = \begin{pmatrix} 1 \\ x_1 \\ x_2 \end{pmatrix}, \boldsymbol{\beta} \in \mathbb{R}^3$$

Generalization for affine functions:

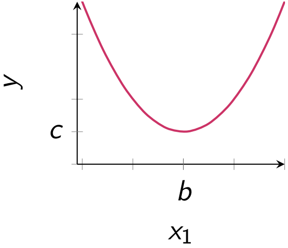
$$\phi_{aff}(\mathbf{x}) = \begin{pmatrix} 1 \\ \mathbf{x} \end{pmatrix} \in \mathbb{R}^{d+1}$$

Function Classes

1. Affine functions:

$$\mathcal{F}_{aff} = \left\{ f : \mathbb{R}^d \to \mathbb{R}, f(\mathbf{x}) = \phi_{aff}(\mathbf{x})^\top \boldsymbol{\beta} \,\middle|\, \boldsymbol{\beta} \in \mathbb{R}^{d+1} \right\}$$

Polynomials of Degree k=2 (d=1)

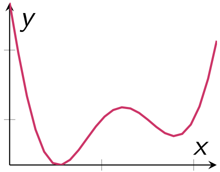


$$f : \mathbb{R} \to \mathbb{R}$$

$$\begin{aligned} f(x) &= a(x-b)^2 + c \\ &= ax^2 - 2abx + ab^2 + c \\ &= \beta_2 x^2 + \beta_1 x + \beta_0 \\ &= \begin{pmatrix} 1 & x & x^2 \end{pmatrix} \begin{pmatrix} \beta_0 \\ \beta_1 \\ \beta_2 \end{pmatrix} \\ &= \phi(x)^\top \boldsymbol{\beta}, \text{ where} \end{aligned}$$

$$\phi(x) = \begin{pmatrix} 1 \\ x \\ x^2 \end{pmatrix}, \boldsymbol{\beta} \in \mathbb{R}^3$$

Polynomials of Degree k (d=1)



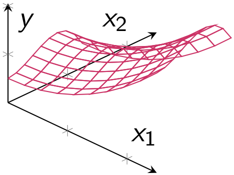
$$f : \mathbb{R} \to \mathbb{R}$$

$$\begin{aligned} f(x) &= \beta_k x^k + \ldots + \beta_1 x + \beta_0 \\ &= \begin{pmatrix} 1 & \ldots & x^k \end{pmatrix} \begin{pmatrix} \beta_0 \\ \vdots \\ \beta_k \end{pmatrix} \\ &= \phi(x)^\top \boldsymbol{\beta}, \text{ where} \end{aligned}$$

$$\phi(x) = \begin{pmatrix} 1 \\ x \\ \vdots \\ x^k \end{pmatrix}, \boldsymbol{\beta} \in \mathbb{R}^{k+1}$$

Multivariate Polynomials of Degree k (d=2)

$$f : \mathbb{R}^2 \to \mathbb{R}$$



$$f(\mathbf{x}) = \sum_{i_1=0}^{k} \sum_{i_2=0}^{k} \beta_{i_1 i_2} x_1^{i_1} x_2^{i_2}$$

$$= \underbrace{\begin{pmatrix} 1 & \ldots & x_1^k x_2^{k-1} & x_1^k x_2^k \end{pmatrix}}_{=:\phi(\mathbf{x})^\top} \begin{pmatrix} \beta_{00} \\ \vdots \\ \beta_{k(k-1)} \\ \beta_{kk} \end{pmatrix}$$

$$= \phi(\mathbf{x})^\top \boldsymbol{\beta}, \text{ where } \phi(\mathbf{x}), \boldsymbol{\beta} \in \mathbb{R}^{(k+1)^2}.$$

Generalization for polynomials of degree k:

$$\phi_{pk}(\mathbf{x}) \in \mathbb{R}^{(k+1)^d}, \text{ for } \mathbf{x} \in \mathbb{R}^d$$

Function Classes

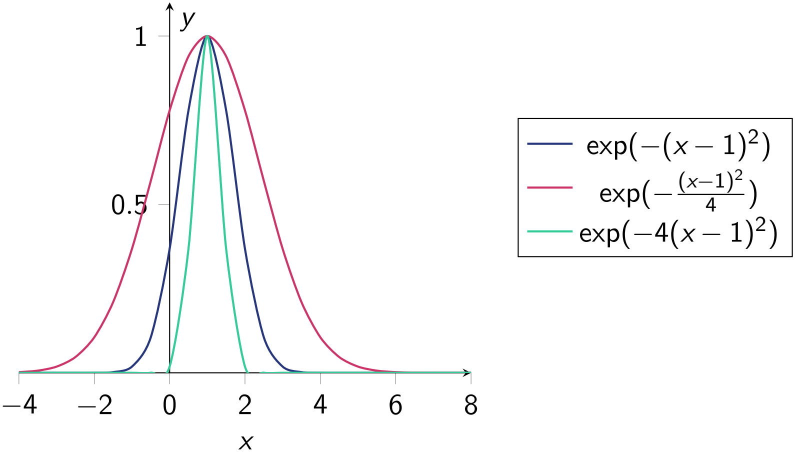
1. Affine functions:

$$\mathcal{F}_{aff} = \left\{ f : \mathbb{R}^d \to \mathbb{R}, f(\mathbf{x}) = \phi_{aff}(\mathbf{x})^\top \boldsymbol{\beta} \middle| \boldsymbol{\beta} \in \mathbb{R}^{d+1} \right\}$$

2. Polynomials of degree k:

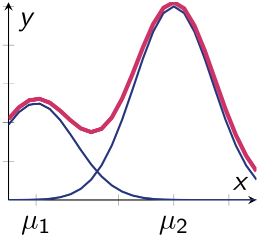
$$\mathcal{F}_{pk} = \left\{ f : \mathbb{R}^d \to \mathbb{R}, f(\mathbf{x}) = \phi_{pk}(\mathbf{x})^\top \boldsymbol{\beta} \middle| \boldsymbol{\beta} \in \mathbb{R}^{(k+1)^d} \right\}$$

The Gaussian Function



$$\kappa(\mathbf{x}) = \exp\left(-\gamma\|\mathbf{x}-\boldsymbol{\mu}\|^2\right)$$

Local Gaussian Radial Basis Functions



$f : \mathbb{R} \to \mathbb{R}$

$$f(x) = \sum_{i=1}^{k} \beta_i \exp\left(-\frac{\|x - \mu_i\|^2}{2\sigma^2}\right)$$

$$= \begin{pmatrix} \kappa_1(x) & \dots & \kappa_k(x) \end{pmatrix} \begin{pmatrix} \beta_1 \\ \vdots \\ \beta_k \end{pmatrix}$$

$$= \phi(x)^\top \boldsymbol{\beta}, \text{ where } \phi(x), \boldsymbol{\beta} \in \mathbb{R}^k$$

Generalization for the sum of k Gaussians:

$$\phi_{Gk}(\mathbf{x}) = \left(\exp(-\gamma\|\mathbf{x} - \boldsymbol{\mu}_1\|^2) \dots \exp(-\gamma\|\mathbf{x} - \boldsymbol{\mu}_k\|^2)\right)$$

Function Classes

1. Affine functions:

$$\mathcal{F}_{aff} = \left\{ f : \mathbb{R}^d \to \mathbb{R}, f(\mathbf{x}) = \phi_{aff}(\mathbf{x})^\top \beta \middle| \beta \in \mathbb{R}^{d+1} \right\}$$

2. Polynomials of degree k:

$$\mathcal{F}_{pk} = \left\{ f : \mathbb{R}^d \to \mathbb{R}, f(\mathbf{x}) = \phi_{pk}(\mathbf{x})^\top \beta \middle| \beta \in \mathbb{R}^{(k+1)^d} \right\}$$

3. Sum of k Gaussians:

$$\mathcal{F}_{Gk} = \left\{ f : \mathbb{R}^d \to \mathbb{R}, f(\mathbf{x}) = \phi_{Gk}(\mathbf{x})^\top \beta \middle| \beta \in \mathbb{R}^{k} \right\}$$

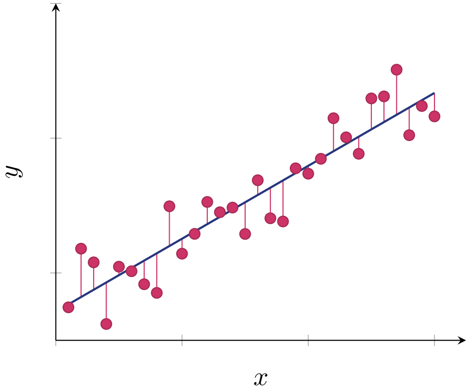
Ok, so we know now of three function families which we can use to fit our model...

.. and they can all be defined by the inner product of a basis function ϕ and a vector β.

But how do we fit our model?

Minimize the Residual Sum of Squares

Measuring the Fit of a Function



The Residual Sum of Squares

We want to minimize the approximation error of our function f to the target values y:

$$\begin{aligned} RSS(\beta) &= \sum_{i=1}^{n} (y_i - f(D_{i\cdot}))^2 \\ &= \sum_{i=1}^{n} (y_i - \phi(D_{i\cdot}^\top)^\top \beta)^2 \\ &= \sum_{i=1}^{n} (y_i - X_{i\cdot}\beta)^2 \\ &= \|\mathbf{y} - X\beta\|^2. \end{aligned}$$

The function $RSS(\beta)$ is known as the Residual Sum of Squares.

The Design Matrix

Our function class is given for a specified basis function ϕ as:

$$\mathcal{F} = \{f : \mathbb{R}^d \rightarrow \mathbb{R}, f(\mathbf{x}) = \phi(\mathbf{x})^\top \boldsymbol{\beta} | \boldsymbol{\beta} \in \mathbb{R}^p\}.$$

We gather the (transformed) feature vectors $\phi(D_{i\cdot}^\top)$ in the design matrix X and the target values in the vector $\mathbf{y}$:

$$X = \begin{pmatrix} -- & \phi(D_{1\cdot}^\top)^\top & -- \\ & \vdots & \\ -- & \phi(D_{n\cdot}^\top)^\top & -- \end{pmatrix} \in \mathbb{R}^{n \times p}, \qquad \mathbf{y} = \begin{pmatrix} y_1 \\ \vdots \\ y_n \end{pmatrix} \in \mathbb{R}^n$$

The Regression Task

Given a dataset consisting of n observations

$$\mathcal{D} = \left\{ (D_{i\cdot}, y_i) | D_{i\cdot} \in \mathbb{R}^{1\times d}, y_i \in \mathbb{R}, 1 \leq i \leq n \right\}$$

Choose a basis function $\phi : \mathbb{R}^d \to \mathbb{R}^p$, and create the design matrix $X \in \mathbb{R}^{n\times p}$, where $X_{i\cdot} = \phi(D_{i\cdot}^\top)^\top$

Find the regression vector β, solving following objective

$$\min_{\beta} RSS(\beta) = \|\mathbf{y} - X\beta\|^2 \qquad \text{s.t. } \beta \in \mathbb{R}^p.$$

Return the predictor function $f : \mathbb{R}^d \to \mathbb{R}$, $f(\mathbf{x}) = \phi(\mathbf{x})^\top \beta$.

3

Optimization

The RSS is a Convex Function

Theorem

The function $RSS(\beta) = \|\mathbf{y} - X\beta\|^2$ is convex.

Proof (Sketch): We show that the squared L_2-norm $\|\cdot\|^2$ is a convex function.

The composition of the affine function $g(\beta) = \mathbf{y} - X\beta$ with the convex function $\|\cdot\|^2$, given by the $RSS(\beta) = \|g(\beta)\|^2$ is then also convex.

Regression is a Convex Optimization Problem

The optimization problem

$$\min_{\boldsymbol{\beta}} RSS(\boldsymbol{\beta}) \qquad \text{s.t. } \boldsymbol{\beta} \in \mathbb{R}^p$$

is convex:

1. $RSS(\alpha\boldsymbol{\beta}_1 + (1-\alpha)\boldsymbol{\beta}_2) \leq \alpha RSS(\boldsymbol{\beta}_1) + (1-\alpha)RSS(\boldsymbol{\beta}_2)$ for every $\alpha \in [0,1]$, $\boldsymbol{\beta}_1, \boldsymbol{\beta}_2 \in \mathbb{R}^p$
2. $\mathbb{R}^p$ is convex.

So, we have an unconstrained optimization problem with a smooth objective function. How do we solve this problem?

With FONC!

Solving the Regression Problem

We compute the stationary points, setting the gradient to zero.

$$RSS(\beta) = \|\mathbf{y} - X\beta\|^2 \qquad \nabla_\beta RSS(\beta) = -2X^\top(\mathbf{y} - X\beta)$$

$$-2(X^\top(\mathbf{y} - X\beta)) = 0 \Leftrightarrow X^\top X\beta = X^\top \mathbf{y}$$

According to FONC the set of possible minimizers of the regression problem are given by the set of regression vectors

$$\{\beta \in \mathbb{R}^p \mid X^\top X\beta = X^\top \mathbf{y}\}.$$

Since RSS is convex, all stationary points are global minima.

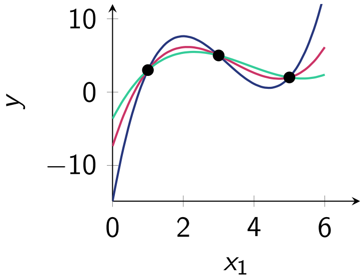
Regression Minimizers

The global minimizers of the regression problem are given by

$$\{\boldsymbol{\beta} \in \mathbb{R}^p \mid X^\top X\boldsymbol{\beta} = X^\top \mathbf{y}\}.$$

If the matrix $X^\top X$ is invertible, then there is only one minimizer:

$$\boldsymbol{\beta} = (X^\top X)^{-1} X^\top \mathbf{y}$$



However, there also might be infinitely many global minimizers of $RSS(\boldsymbol{\beta})$.

So, if I have a regression problem, then I choose a basis function and determine the solution by solving that system of linear equations.

But how do I know if I chose a good basis function?

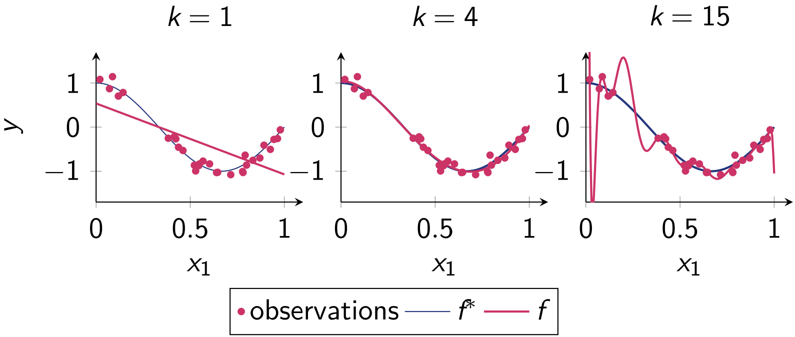
Bias-Variance Tradeoff

3

Evaluation

Finding the Right Basis Function

Assume we want to approximate the true function f^* with polynomials of degree k:



What is the best k?

Evaluate on a Test Set

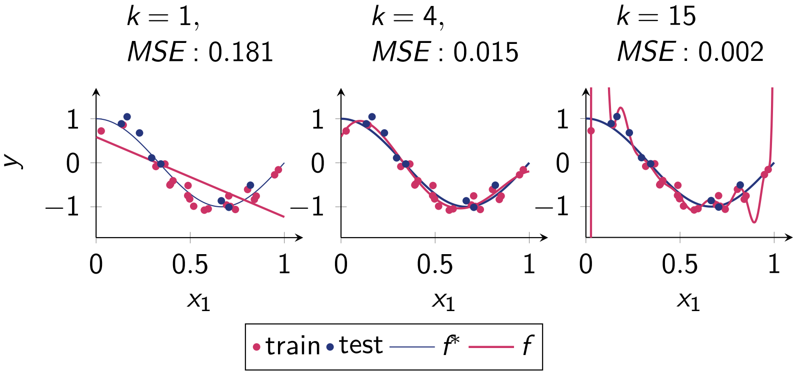
If the model assumption is correct then the regression model should be able to predict y for unobserved $\mathbf{x}$.

Idea: Hold out a test set, indicated by $\mathcal{I} \subseteq \{1, \dots n\}$ from the n training data points and compute the error on the test data.

The Mean Squared Error (MSE) returns the average squared prediction error:

$$MSE(\beta, \mathcal{I}) = \frac{1}{|\mathcal{I}|} \sum_{i \in \mathcal{I}} (y_i - \phi(D_{i\cdot}^\top)^\top \beta)^2$$

Computing the MSE on a Test Set



So, the best k is $k = 15$??

Ok, what did just happen? Can that happen often? How can I make my evaluation reliable?

Selecting the Best Model in Theory: Minimizing EPE

In theory, the MSE results from the following process:

- sample the (finite) training data $\mathcal{D}_j \subset \mathbb{R}^{1\times d} \times \mathbb{R}$
- learn a model $f_j(\mathbf{x}) = \phi(\mathbf{x})^\top \boldsymbol{\beta}_j$ based on the training data,
- sample a (finite) test set $\mathcal{T}_j \subset \mathbb{R}^{1\times d} \times \mathbb{R}$
- compute MSE_j

If we repeat this sampling process k times, obtaining scores $MSE_1, \ldots, MSE_k$, we could approximate the Expected squared Prediction Error (EPE)

$$\frac{1}{k}\sum_{j=1}^{k} MSE_j = \frac{1}{k}\sum_{j=1}^{k} \frac{1}{|\mathcal{T}_j|} \sum_{(\mathbf{x},y)\in\mathcal{T}_j} (y - f_j(\mathbf{x}))^2 \approx \mathbb{E}_{\mathbf{x},y,\mathcal{D}}[(y - f_{\mathcal{D}}(\mathbf{x}))^2].$$

The Random Variables of EPE

EPE has three random variables:

$$\mathbb{E}_{\mathbf{x},y,\mathcal{D}}[(y - f_{\mathcal{D}}(\mathbf{x}))^2],$$

where

- $\mathbf{x}$ is the random variable of a feature vector in the test set.
- y is the random variable of the target of $\mathbf{x}$.
- $\mathcal{D}$ is the random variable of the training data.

Interpreting Targets are Samples of a Random Variable

Assumption: the process generating the i-th (noisy) target is

$$y_i = f^*(D_{i \cdot}) + \epsilon_i,$$

where f^* is the true regression function and ϵ_i is a sample of a random variable ϵ with mean $\mu = 0$ and variance σ^2.

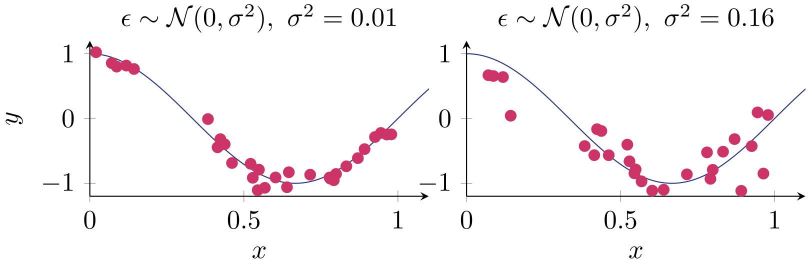
As a result, the targets are samples of the random variable $y = f^*(\mathbf{x}) + \epsilon$ such that

$$\mathbb{E}_y[y|\mathbf{x}] = f^*(\mathbf{x}) \qquad Var_y(y|\mathbf{x}) = \mathbb{E}_y[(y - f^*(\mathbf{x}))^2|\mathbf{x}] = \sigma^2$$

Sampling Training Data With Gaussian Noise: Effect of σ

For example, one-dimensional training and test data points could be sampled by the process

1. Sample $x_i \in [0, 1]$ for $1 \leq i \leq n$
2. Sample ϵ_i from $\mathcal{N}(0, \sigma^2)$
3. Set $y_i = f^*(x_i) + \epsilon_i$



The Bias-Variance Tradeoff

$$\mathbb{E}_{\mathbf{x},y,\mathcal{D}}[(y - f_{\mathcal{D}}(\mathbf{x}))^2] = \mathbb{E}_{\mathbf{x}}[\mathbb{E}_{y,\mathcal{D}}[(y - f_{\mathcal{D}}(\mathbf{x}))^2|\mathbf{x}]]$$

We fix the random variable $\mathbf{x}$ to a value $\mathbf{x}_0$ and get

$$\begin{aligned}\mathbb{E}_{y,\mathcal{D}}[(y - f_{\mathcal{D}}(\mathbf{x}_0))^2] = \sigma^2 &+ \underbrace{(f^*(\mathbf{x}_0) - \mathbb{E}_{\mathcal{D}}[f_{\mathcal{D}}(\mathbf{x}_0)])^2}_{bias^2} \\ &+ \underbrace{\mathbb{E}_{\mathcal{D}}[(\mathbb{E}_{\mathcal{D}}[f_{\mathcal{D}}(\mathbf{x}_0)] - f_{\mathcal{D}}(\mathbf{x}_0))^2]}_{variance}\end{aligned}$$

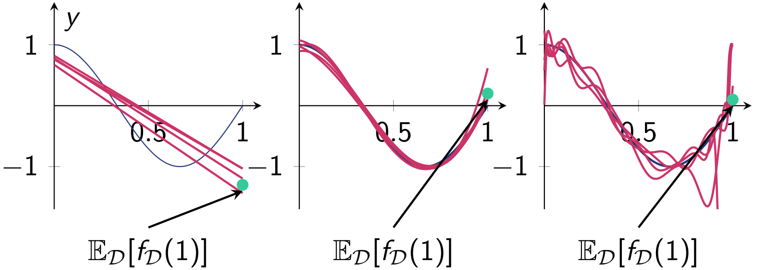
Hence, the expected squared prediction error is minimized for functions having a low variance and low bias.

Bias and Variance of Models

$k = 1$, high bias, low variance

$k = 4$, low variance, low bias

$k = 15$, high variance, low bias



The red lines are the regression functions trained on three training data sets.

Selecting the Best Model in Practice: Cross-Validation

k-fold CV: divide the data set into k disjunctive chunks indicated by

$$\{1,\ldots,n\} = \mathcal{I} = \mathcal{I}_1 \cup \ldots \cup \mathcal{I}_k,\ \mathcal{I}_j \cap \mathcal{I}_l = \emptyset \text{ for } j \neq l$$

Train k models where model $f_j(\mathbf{x}) = \phi(\mathbf{x})^\top \boldsymbol{\beta}_j$ is trained on the datapoints $\mathcal{I} \setminus \mathcal{I}_j$ and evaluated on the datapoints $\mathcal{I}_j$.

The cross-validation MSE is then given as

$$\frac{1}{k}\sum_{j=1}^{k} MSE(\boldsymbol{\beta}_j, \mathcal{I}_j) = \frac{1}{k}\sum_{j=1}^{k} \frac{1}{|\mathcal{I}_j|}\sum_{i\in\mathcal{I}_j}(y_i - f_j(D_{i\cdot}))^2$$

Question: Is the cross-validation MSE in general a good approximation of EPE?

Selecting the Best Model in Practice: Cross-Validation

k-fold CV: divide the data set into k disjunctive chunks indicated by

$$\{1, \ldots, n\} = \mathcal{I} = \mathcal{I}_1 \cup \ldots \cup \mathcal{I}_k, \ \mathcal{I}_j \cap \mathcal{I}_l = \emptyset \text{ for } j \neq l$$

Train k models where model $f_j(\mathbf{x}) = \phi(\mathbf{x})^\top \boldsymbol{\beta}_j$ is trained on the datapoints $\mathcal{I} \setminus \mathcal{I}_j$ and evaluated on the datapoints $\mathcal{I}_j$.

The cross-validation MSE is then given as

$$\frac{1}{k}\sum_{j=1}^{k} MSE(\boldsymbol{\beta}_j, \mathcal{I}_j) = \frac{1}{k}\sum_{j=1}^{k} \frac{1}{|\mathcal{I}_j|} \sum_{i \in \mathcal{I}_j} (y_i - f_j(D_{i\cdot}))^2$$

Question: Is the cross-validation MSE in general a good approximation of EPE?

Selecting the Best Model in Practice: Cross-Validation

k-fold CV: divide the data set into k disjunctive chunks indicated by

$$\{1, \dots, n\} = \mathcal{I} = \mathcal{I}_1 \cup \dots \cup \mathcal{I}_k,\ \mathcal{I}_j \cap \mathcal{I}_l = \emptyset \text{ for } j \neq l$$

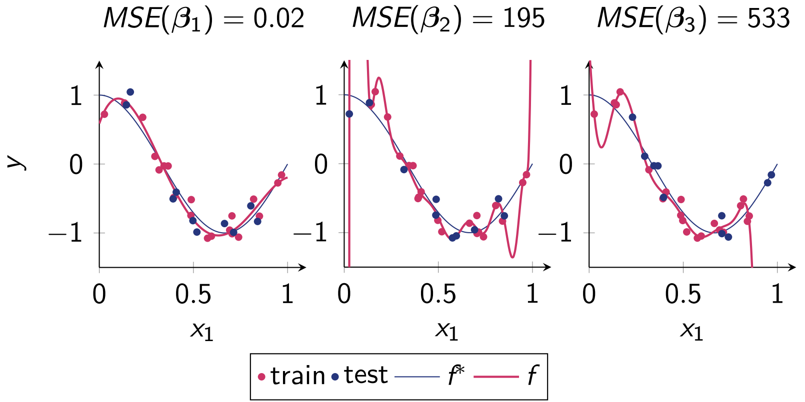
Train k models where model $f_j(\mathbf{x}) = \phi(\mathbf{x})^\top \boldsymbol{\beta}_j$ is trained on the datapoints $\mathcal{I} \setminus \mathcal{I}_j$ and evaluated on the datapoints $\mathcal{I}_j$.

The cross-validation MSE is then given as

$$\frac{1}{k}\sum_{j=1}^{k} MSE(\boldsymbol{\beta}_j, \mathcal{I}_j) = \frac{1}{k}\sum_{j=1}^{k}\frac{1}{|\mathcal{I}_j|}\sum_{i\in\mathcal{I}_j}(y_i - f_j(D_{i\cdot}))^2$$

Question: Is the cross-validation MSE in general a good approximation of EPE?

A 3-fold CV for training a Polynomial with $k = 15$



The 3-fold CV-MSE is the given as $\frac{1}{3}(0.02 + 195 + 533) = 242.7$

The

Bias-Variance Tradeoff

is a Theoretic Measure of

Over- and Underfit

in a Regression Model