Regression

- Given: Dataset $\mathcal{D} = \{(\mathbf{x}_i, Y_i) | i = 1, ..., n\}$ with n tuples
  - $\mathbf{x}$: Object description
  - Y: Numerical target attribute $\Rightarrow$ regression problem
- Find a function $f : \text{dom}(X_1) \times \ldots \times \text{dom}(X_k) \rightarrow Y$ minimizing the error $e(f(x_1, \ldots, x_k), y)$ for all given data objects $(x_1, \ldots, x_k, y)$.

Remember

- Instead of finding structure in a data set, we are now focusing on methods that find explanations for an unknown dependency within the data.
- Supervised (because we know the desired outcome)
- Descriptive (because we care about explanation)

Regression line

Given: A data set for two continuous attributes x and y.

It is assumed that there is an approximate linear dependency between x and y:

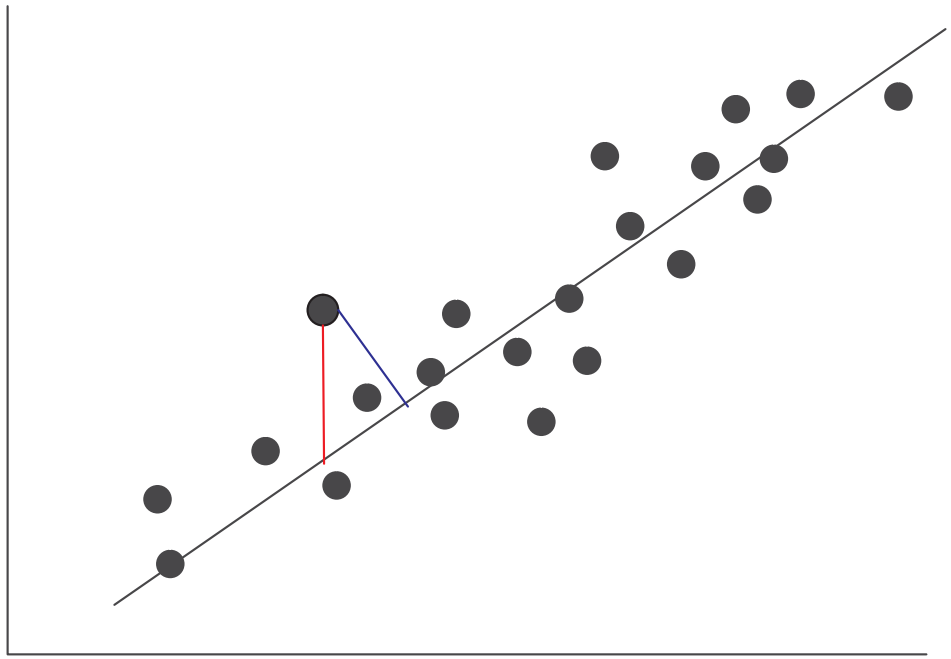
$$y \approx a + bx$$

Find a regression line (i.e. determine the parameters a and b) such that the line fits the data as good as possible.

Example

- Trend estimation (e.g. oil price over time)
- Epidemiology (e.g. cigarette smoking vs. lifespan)
- Finance (e.g. return on investment vs. return on all risky assets)
- Economics (e.g. consumption vs. available income)

Regression Line



What is a good fit?

Cost functions

Usually, the sum of square errors in y-direction is chosen as cost function (to be minimized).

Other reasonable cost functions:

- mean absolute distance in y-direction
- mean Euclidean distance
- maximum absolute distance in y-direction (or equivalently: the maximum squared distance in y-direction)
- maximum Euclidean distance
- ...

Construction

Given data (x_i, y_i) $(i = 1, \ldots, n)$, the least squares cost function is

$$F(a, b) \;=\; \sum_{i=1}^{n} \left((a + bx_i) - y_i\right)^2 .$$

Goal

The y-values that are computed with the linear equation should (in total) deviate as little as possible from the measured values.

Finding the minimum

A necessary condition for a minimum of the cost function $F(a,b) = \sum_{i=1}^{n} \left((a + bx_i) - y_i\right)^2$ is that the partial derivatives of this function w.r.t the parameters a and b vanish, that is

$$\frac{\partial F}{\partial a} = \sum_{i=1}^{n} 2(a + bx_i - y_i) = 0 \quad \text{and} \quad \frac{\partial F}{\partial b} = \sum_{i=1}^{n} 2(a + bx_i - y_i)x_i = 0$$

As a consequence, we obtain the so-called normal equations

$$na + \left(\sum_{i=1}^{n} x_i\right) b = \sum_{i=1}^{n} y_i \quad \text{and} \quad \left(\sum_{i=1}^{n} x_i\right) a + \left(\sum_{i=1}^{n} x_i^2\right) b = \sum_{i=1}^{n} x_i y_i$$

that is, a two-equation system with two unknowns a and b which has a unique solution (if at least two different x-values exist).

Least squares and MLE

A regression line can be interpreted as a maximum likelihood estimator (MLE):

Assumption: The data generation process can be described well by the model

$$y = a + bx + \xi,$$

where ξ is normally distributed random variable with mean 0 and (unknown) variance σ^2.

The parameter that minimizes the sum of squared deviations (in y-direction) from the data points maximizes the probability of the data given this model class.

Least squares and MLE

Therefore,

$$f(y \mid x) = \frac{1}{\sqrt{2\pi\sigma^2}} \cdot \exp\left(-\frac{(y-(a+bx))^2}{2\sigma^2}\right),$$

leading to the **likelihood function**

$$\begin{aligned} & L((x_1, y_1), \ldots (x_n, y_n); a, b, \sigma^2) \\ = & \prod_{i=1}^{n} f(y_i \mid x_i) \\ = & \prod_{i=1}^{n} \cdot \frac{1}{\sqrt{2\pi\sigma^2}} \cdot \exp\left(-\frac{(y_i-(a+bx_i))^2}{2\sigma^2}\right). \end{aligned}$$

Least squares and MLE

To simplify the computation of derivatives for finding the maximum, we compute the logarithm:

$$
\begin{aligned}
&\ln L((x_1, y_1), \ldots (x_n, y_n); a, b, \sigma^2) \\
&= \ln \prod_{i=1}^{n} \frac{1}{\sqrt{2\pi\sigma^2}} \cdot \exp\left(-\frac{(y_i - (a + bx_i))^2}{2\sigma^2}\right) \\
&= \sum_{i=1}^{n} \ln \frac{1}{\sqrt{2\pi\sigma^2}} - \frac{1}{2\sigma^2} \sum_{i=1}^{n} (y_i - (a + bx_i))^2
\end{aligned}
$$

Least squares and MLE

$$\sum_{i=1}^{n} \ln \frac{1}{\sqrt{2\pi\sigma^2}} \; - \; \frac{1}{2\sigma^2} \sum_{i=1}^{n} (y_i - (a + bx_i))^2$$

From this expression it becomes clear by computing the derivatives w.r.t. the parameters a and b that maximizing the likelihood function is equivalent to minimizing

$$F(a, b) = \sum_{i=1}^{n} (y_i - (a + bx_i))^2.$$

Regression polynomials

The least squares method can be extended to regression polynomials (e.g. x = time, y = distance by constant acceleration)

$$y = p(x) = a_0 + a_1 x + \ldots + a_m x^m$$

with a given fixed degree m.

We have to minimize the error function

$$\begin{aligned} F(a_0, \ldots, a_m) &= \sum_{i=1}^{n} (p(x_i) - y_i)^2 \\ &= \sum_{i=1}^{n} ((a_0 + a_1 x_i + \ldots + a_m x_i^m) - y_i)^2 \end{aligned}$$

In analogy to the linear case, we form the partial derivatives of this function w.r.t. the parameters $a_k, 0 \leq k \leq m$, and equate them to zero.

Multilinear regression

- Given: A data set $((\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n))$ with
  - input vectors $\mathbf{x}_i$ and
  - corresponding responses y_i,
  - $1 \leq i \leq n$.

for which we want to determine the linear regression function

$$y = f(x_1, \ldots, x_m) = a_0 + \sum_{k=1}^{m} a_k x_k.$$

Example

- Price of a house depending on its size (x_1) and age (x_2)
- Ice cream consumption based on the temperature (x_1), the price (x_2) and the family income (x_3)
- Electric consumption based on the number of flats with one (x_1), two (x_2), three (x_3) and four or more persons (x_4) living in them

Multilinear regression

$$
\begin{aligned}
F(a_0, \ldots, a_m) &= \sum_{i=1}^{n} (f(\mathbf{x_i}) - y_i)^2 \\
&= \sum_{i=1}^{n} \left(a_0 + a_1 x_1^{(i)} + \ldots + a_m x_m^{(i)} - y_i \right)^2
\end{aligned}
$$

In order to derive the normal equations, it is convenient to write the functional to minimize in matrix form

$$
F(\mathbf{a}) = (\mathbf{Xa} - \mathbf{y})^\top (\mathbf{Xa} - \mathbf{y})
$$

where

$$
\mathbf{a} = \begin{pmatrix} a_0 \\ \vdots \\ a_m \end{pmatrix}, \mathbf{X} = \begin{pmatrix} 1 & x_{1,1} & \cdots & x_{1,m} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & x_{n,1} & \cdots & x_{n,m} \end{pmatrix} \text{ and } \mathbf{y} = \begin{pmatrix} y_1 \\ \vdots \\ y_n \end{pmatrix}
$$

Multilinear regression

Again a necessary condition for a minimum is that the partial derivatives of this function w.r.t the coefficients $a_k, 0 \leq k \leq m$, vanish.
Using the differential operator ∇, we can write these conditions as

$$\nabla_a F(\mathbf{a}) = \frac{d}{d\mathbf{a}} F(\mathbf{a}) = \left(\frac{\partial}{\partial a_0} F(\mathbf{a}), \frac{\partial}{\partial a_1} F(\mathbf{a}), \ldots, \frac{\partial}{\partial a_m} F(\mathbf{a}) \right) = 0$$

Whereas the differential operator behaves like a vector

$$\nabla_a = \left(\frac{\partial}{\partial a_0}, \frac{\partial}{\partial a_1}, \ldots, \frac{\partial}{\partial a_m} \right)$$

Multilinear regression

$$F(\mathbf{a}) = (\mathbf{Xa} - \mathbf{y})^\top (\mathbf{Xa} - \mathbf{y})$$

to find the minimum we use the differential operator ∇

$$\begin{aligned}
0 &= \nabla_a (\mathbf{Xa} - \mathbf{y})^\top (\mathbf{Xa} - \mathbf{y}) \\
&= (\nabla_a (\mathbf{Xa} - \mathbf{y}))^\top (\mathbf{Xa} - \mathbf{y}) + \Big((\mathbf{Xa} - \mathbf{y})^\top (\nabla_a (\mathbf{Xa} - \mathbf{y})) \Big)^\top \\
&= (\nabla_a (\mathbf{Xa} - \mathbf{y}))^\top (\mathbf{Xa} - \mathbf{y}) + (\nabla_a (\mathbf{Xa} - \mathbf{y}))^\top (\mathbf{Xa} - \mathbf{y}) \\
&= 2\mathbf{X}^\top (\mathbf{Xa} - \mathbf{y}) \\
&= 2\mathbf{X}^\top \mathbf{Xa} - 2\mathbf{X}^\top \mathbf{y}
\end{aligned}$$

from which we obtain the system of normal equations

$$\mathbf{X}^\top \mathbf{Xa} = \mathbf{X}^\top \mathbf{y}.$$

Multilinear regression

$$\mathbf{X}^\top \mathbf{X}\mathbf{a} = \mathbf{X}^\top \mathbf{y}$$

The system is (uniquely) solvable iff $\mathbf{X}^\top \mathbf{X}$ is invertible (nonsingular).
In this case we have

$$\mathbf{a} = (\mathbf{X}^\top \mathbf{X})^{-1}\mathbf{X}^\top \mathbf{y} = \mathbf{X}^+ \mathbf{y}.$$

Moore-Penrose pseudo-inverse

The expression $(\mathbf{X}^\top \mathbf{X})^{-1}\mathbf{X}^\top = \mathbf{X}^+$ is also known as the (Moore-Penrose) pseudo-inverse of the matrix $\mathbf{X}$.
Pseudo-inverse matrices are used to compute the inverse of none quadratic and singular matrices.
They provides a least square solution to a system of linear equations without a unique solution.

Nonlinear regression

Minimization of the error function based on partial derivatives w.r.t. the parameters does not work in the other examples of error functions, since

- the absolute value and the maximum are not everywhere differentiable and
- the distance in the case of the Euclidean distance leads to system of nonlinear equation for which no analytical solution is known.

Nonlinear regression

Example

For nonlinear dependencies taking partial derivatives leads to nonlinear equations: $y = ae^{bx}$ (radioactive decay, growth of bacteria,...)

$$
\begin{aligned}
E(a,b) &= \sum_{i=1}^{n} \left(ae^{bx_i} - y_i \right)^2 \\
\frac{\partial E}{\partial a} &= 2 \sum_{i=1}^{n} \left(ae^{bx_i} - y_i \right) e^{bx_i} \\
\frac{\partial E}{\partial b} &= 2 \sum_{i=1}^{n} \left(ae^{bx_i} - y_i \right) a x_i e^{bx_i}
\end{aligned}
$$

Possible solutions

- Iterative methods (e.g. gradient descent)
- Transformation of the regression function

Transformation

More complex regression functions can be transformed to the problem of finding a regression line or regression polynomial.

Example

$$y = ax^b$$

can be transformed by taking the logarithm of the equation

$$\ln y = \ln a + b \cdot \ln x$$

Notice

The sum of squared errors is only minimized in the transformed space (coordinates $x' = \ln x$ and $y' = \ln y$), but not necessary also in the original space. Nevertheless, this approach often yields good results and can be used as a starting point for a subsequent gradient descent in the original space.

Logit-transformation

Example

For practical purpose, it is important that one can transform the logistic function,

$$y = \frac{y_{max}}{1 + e^{a+bx}}$$

which describes limited growth processes and is also often used as the activation function of the neurons in artificial neural networks.

$$\frac{1}{y} = \frac{1 + e^{a+bx}}{y_{max}}$$

$$\frac{y_{max} - y}{y} = e^{a+bx}$$

$$\ln\left(\frac{y_{max} - y}{y}\right) = a + bx$$

We only need to transform the data points according to the left-hand side of the equation.

Model vs. black box

When the principal functional dependency between the predictor variables Y and the predictor variables $x_1, \ldots, x_p$ is known, an explicit parameterized (possibly nonlinear) regression function can be specified.

If such a model is not known, one can still try to construct a suitable regression function.

Model vs. black box

When the functional dependency between the predictor variables $X_1, \ldots, X_k$ is not known, one can try a

- linear $y = a_0 + a_1 x_1 + \ldots + a_k x_k$
- quadratic

$$\begin{aligned} y &= a_0 + a_1 \cdot x_1 + \ldots + a_k \cdot x_k + \\ & a_{k+1} \cdot x_1^2 + \ldots + a_{2k} \cdot x_k^2 + \\ & a_{2k+1} \cdot x_1 x_2 + \ldots + a_{2k+k(k-1)/2} \cdot x_{k-1} x_k \end{aligned}$$

- or cubic approach.

Model vs. black box

The coefficients a_i can be interpreted as weighting factors, at least when the predictor variables $X_1, \ldots, X_k$ have been normalised.

They also provide information of a positive or negative correlation of the predictor variables with the dependent variable Y.

Usually, complex regression functions yield black box models, which might provide a good approximation of the data, but do not admit a useful interpretation (of the coefficients).

Generalisation

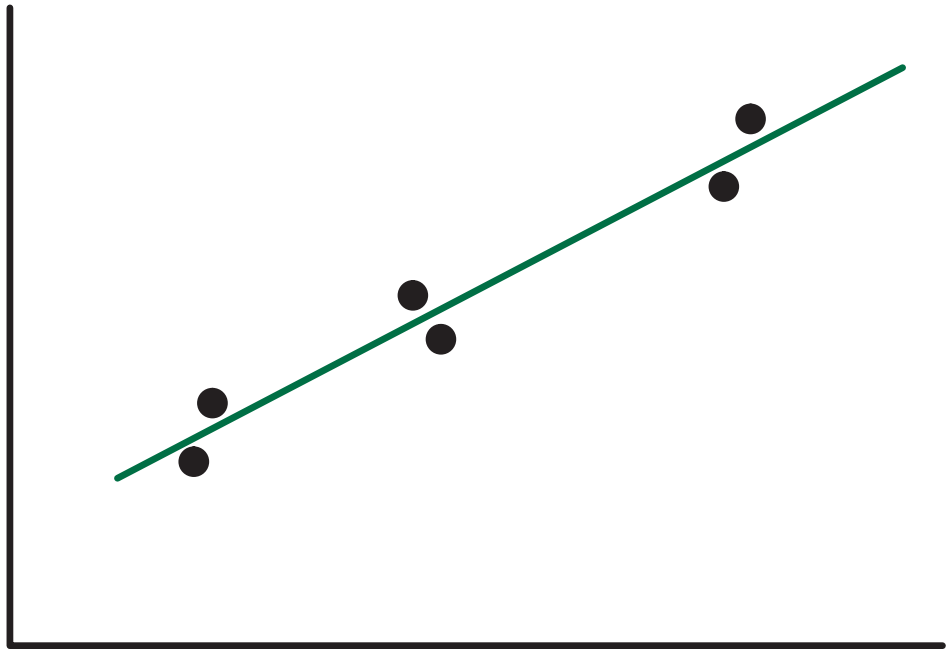
Considering a data set as a collection of examples, describing the dependency between the predictor variables and the dependent variable, the regression function should "learn" this dependency from the data to generalize it in order to make correct predictions on new data.

To achieve this, the regression function must be universal (flexible) enough to be able to learn the dependency.

This does not mean that a more complex regression function with more parameters leads to better results than a simple one.

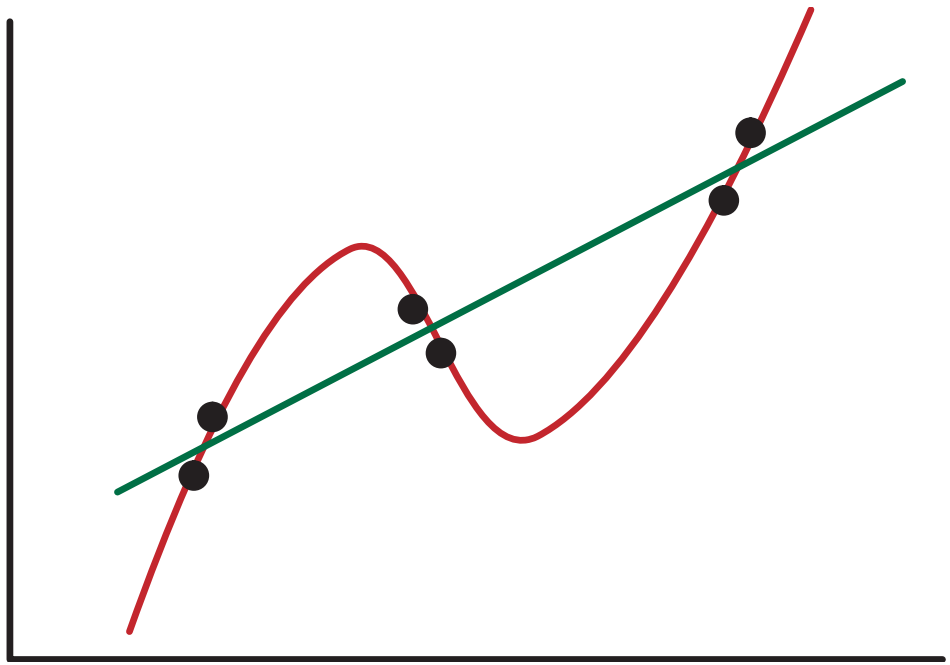
Overfitting

Complex regression functions can lead to overfitting:



Overfitting

Complex regression functions can lead to overfitting:



Keep it simple

The regression function "learns" a description of the data, not of the structure inherent in the data.
The prediction using a complex function can be worse than for a simpler regression function.

Robust regression

Rewrite the error function to be minimized in the form

$$F(\mathbf{a}) = (\mathbf{Xa} - \mathbf{y})^\top (\mathbf{Xa} - \mathbf{y}) = \sum_{i=1}^{n} \rho(e_i) \; = \; \sum_{i=1}^{n} \rho(\mathbf{x}_i^\top \mathbf{a} - y_i)$$

with e_i as the (signed) error of the regression function at ith point and least squares method $\rho(e) = e^2$.

For other choices of ρ, ρ should satisfy at least

- $\rho(e) \geq 0$,
- $\rho(0) = 0$,
- $\rho(e) = \rho(-e)$,
- $\rho(e_i) \geq \rho(e_j)$, if $|e_i| \geq |e_j|$.

M-estimators

Parameter estimation based on an objective function of the given form and an error measure satisfying the above mentioned criteria is called an M-estimator.

M-estimators

Computing the derivatives w.r.t. to the parameters a_i of

$$\sum_{i=1}^{n} \rho(e_i) \;=\; \sum_{i=1}^{n} \rho(\mathbf{x}_i^\top \mathbf{a} - y_i)$$

with $\psi = \rho'$ as the outer derivative leads to the system of $(m+1)$ linear equations

$$\sum_{i=1}^{n} \psi_i(\mathbf{x}_i^\top \mathbf{a} - y_i)\mathbf{x}_i^\top = 0.$$

Defining $w(e) = \psi(e)/e$ and $w_i = w(e_i)$ leads to

$$\sum_{i=1}^{n} \frac{\psi(\mathbf{x}_i^\top \mathbf{a} - y_i)}{e_i} \cdot e_i \cdot \mathbf{x}_i^\top \;=\; \sum_{i=1}^{n} w_i \cdot (\mathbf{x}_i^\top \mathbf{a} - y_i) \cdot \mathbf{x}_i^\top \;=\; \mathbf{0}.$$

Solution of this system of equations is the same as for the standard least squares problem with (nonfixed) weights

$$\sum_{i=1}^{n} w_i e_i^2.$$

M-estimators

Problem:

- The weights w_i depend on the errors e_i and
- the errors e_i depend on the weights w_i.

Solution strategy: Alternating optimization.

1. Choose and initial solution $\mathbf{a}^{(0)}$, for instance, the standard least square solution setting all weights to $w_i = 1$.
2. In each iteration step t, calculate the residuals $e^{(t-1)}$ and the corresponding weights $w^{(t-1)} = w(e^{(t-1)})$ determined by the previous step.
3. Compute the solution of the weighted least square problem $\sum_{i=1}^{n} w_i e_i^2$ which leads to

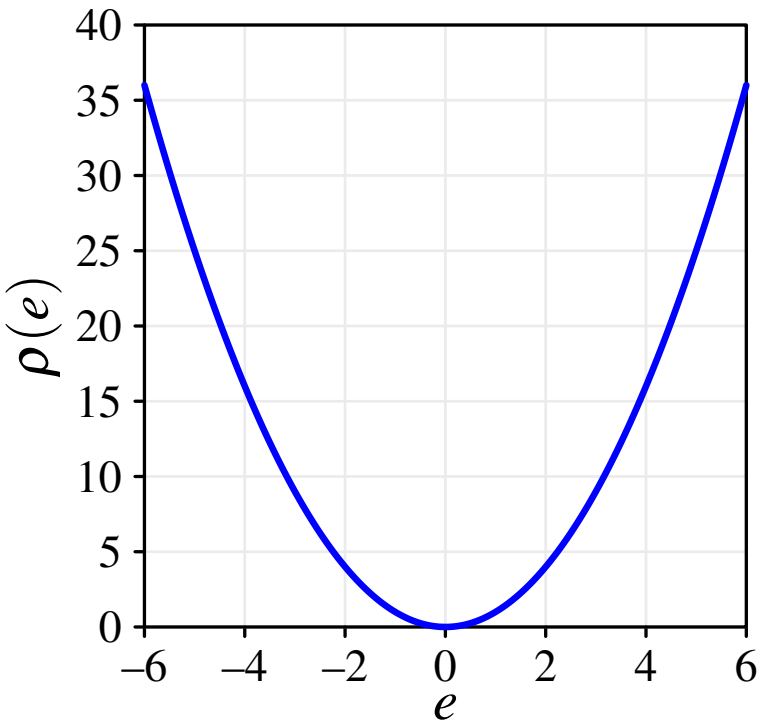
$$\mathbf{a}^{(t)} = (\mathbf{X}^\top \mathbf{W}^{(t-1)} \mathbf{X})^{-1} \mathbf{X}^\top \mathbf{W}^{(t-1)} \mathbf{y},$$

where $\mathbf{W}$ stands for the diagonal matrix with weights w_i on the diagonal.

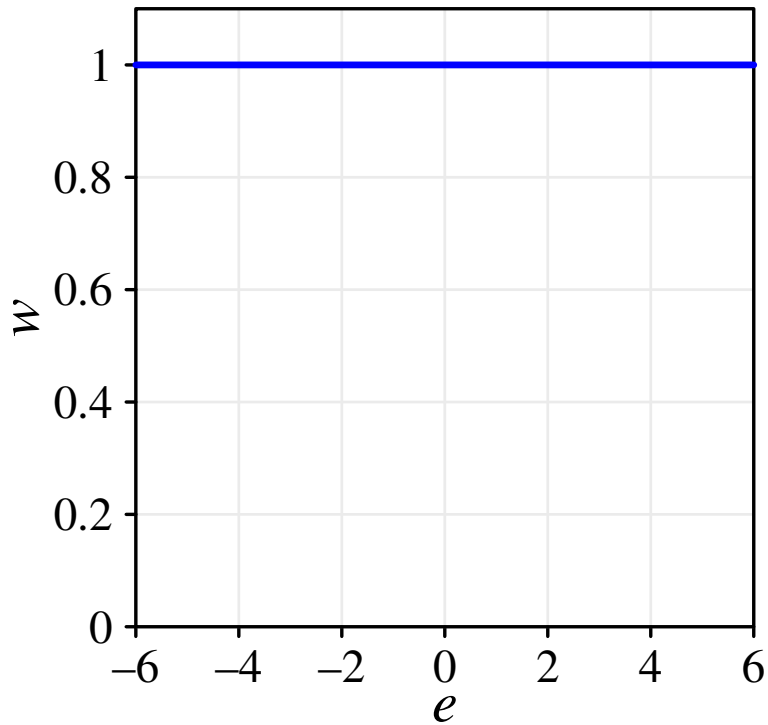
Robust regression

Method	$\rho(e)$
least squares	e^2
Huber	$\begin{cases} \frac{1}{2}e^2, & \text{if } \lvert e\rvert \le k \\ k\lvert e\rvert - \frac{1}{2}k^2, & \text{if } \lvert e\rvert > k \end{cases}$
Tukey	$\begin{cases} \frac{k^2}{6}\left(1-\left(1-\left(\frac{e}{k}\right)^2\right)^3\right)e^2, & \text{if } \lvert e\rvert \le k \\ \frac{k^2}{6}, & \text{if } \lvert e\rvert > k \end{cases}$

Least squares



$$\rho(e) = e^2$$

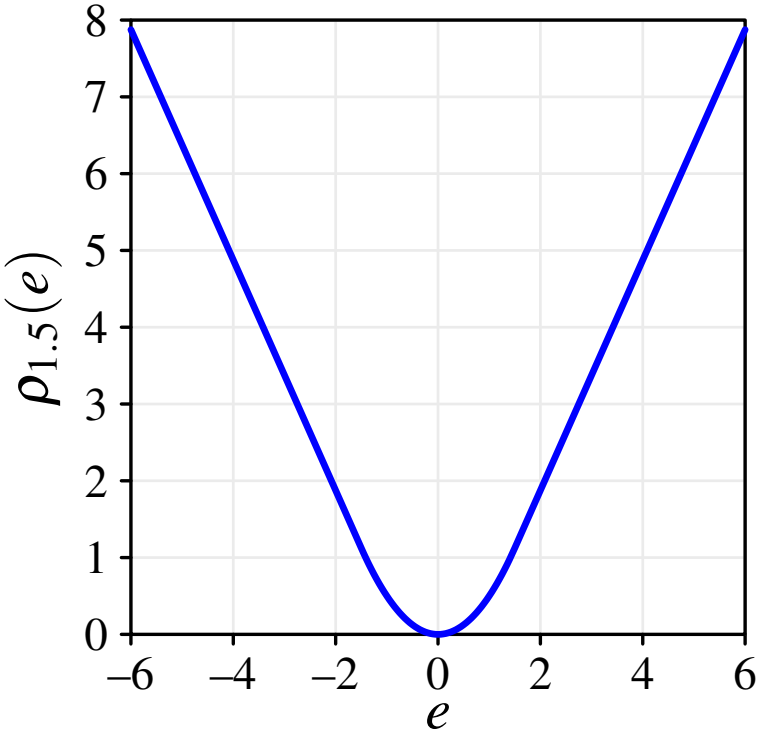


$$\omega(e) = 1$$

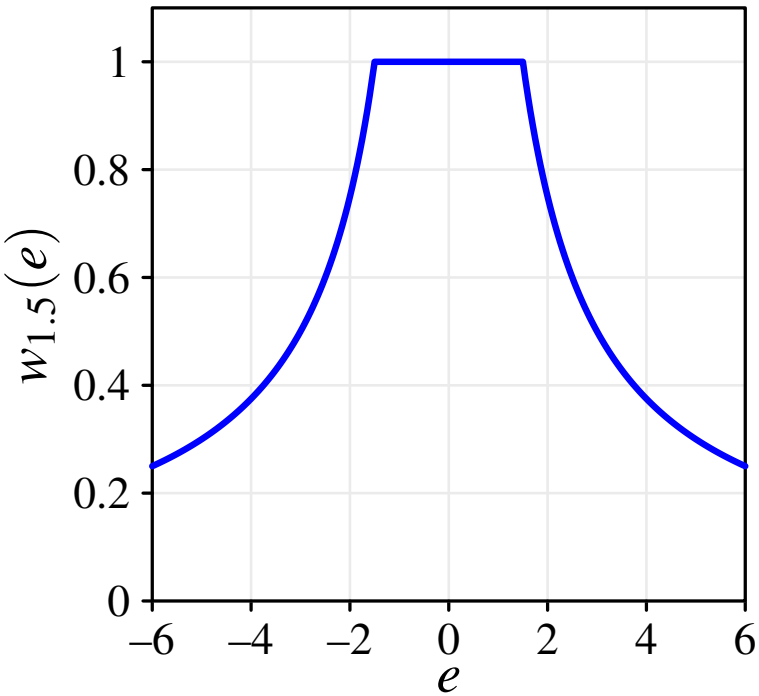
The error measure ρ increases in a quadratic manner with increasing distance.

$\Rightarrow$ Extreme outliers have full influence.

Huber



$\rho(e)$	
$\frac{1}{2}e^2$	if $\lvert e\rvert \leq k$
$k\lvert e\rvert - \frac{1}{2}k^2$	if $\lvert e\rvert > k$

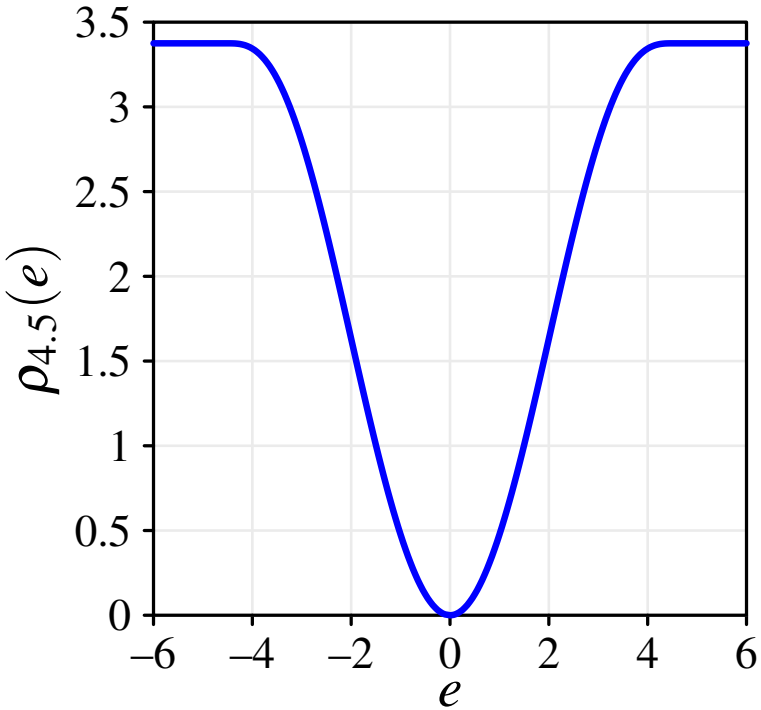


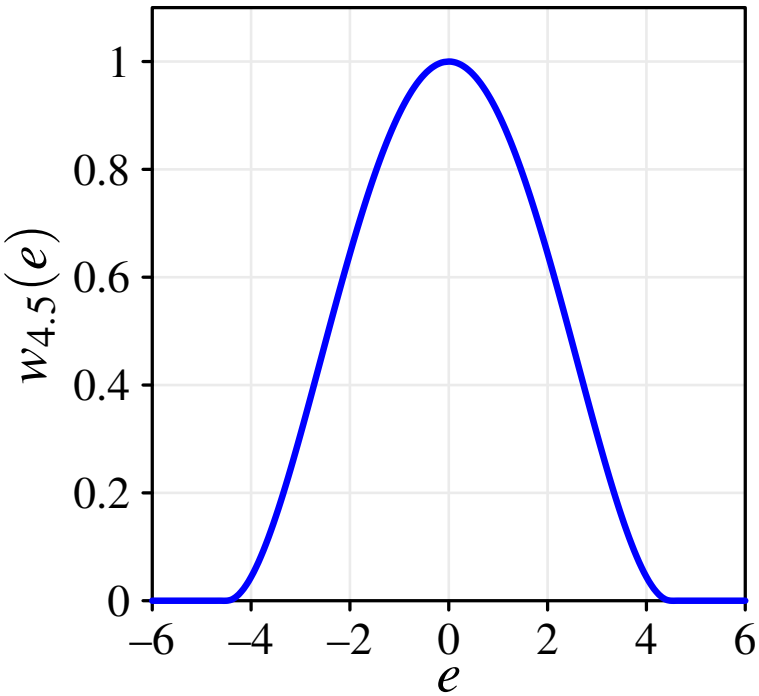
$\omega(e)$	
1	if $\lvert e\rvert \leq k$
$\frac{k}{\lvert e\rvert}$	if $\lvert e\rvert > k$

The error measure ρ switches from a quadratic increase for small errors to a linear increase for larger errors.

$\Rightarrow$ Only data points with a small error have full influence.

Tukey's biweight





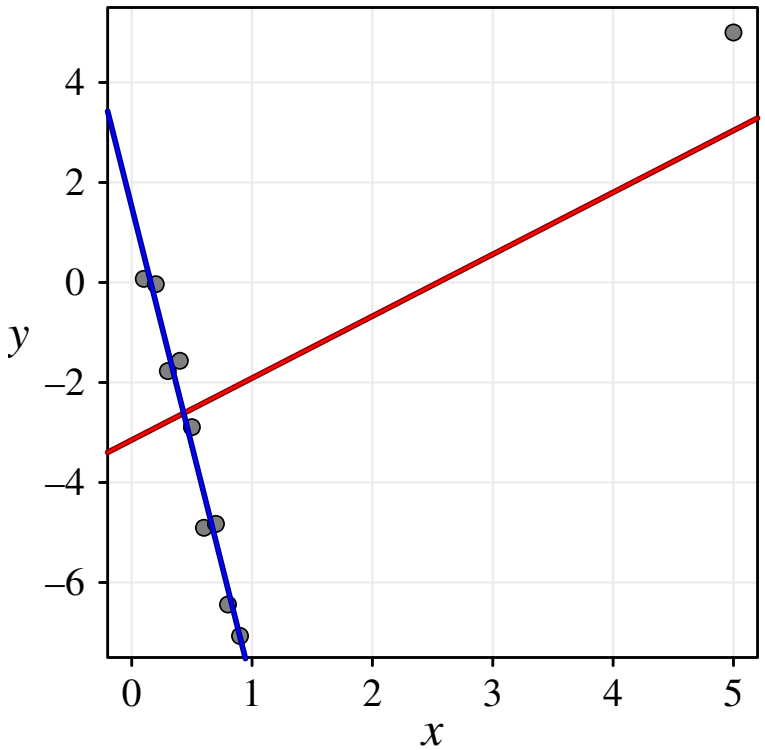
$\rho(e)$	
$\frac{k^2}{6}\left(1-\left(1-\left(\frac{e}{k}\right)^2\right)^3\right)$	if $\lvert e\rvert \leq k$
$\frac{k^2}{6}$	if $\lvert e\rvert > k$

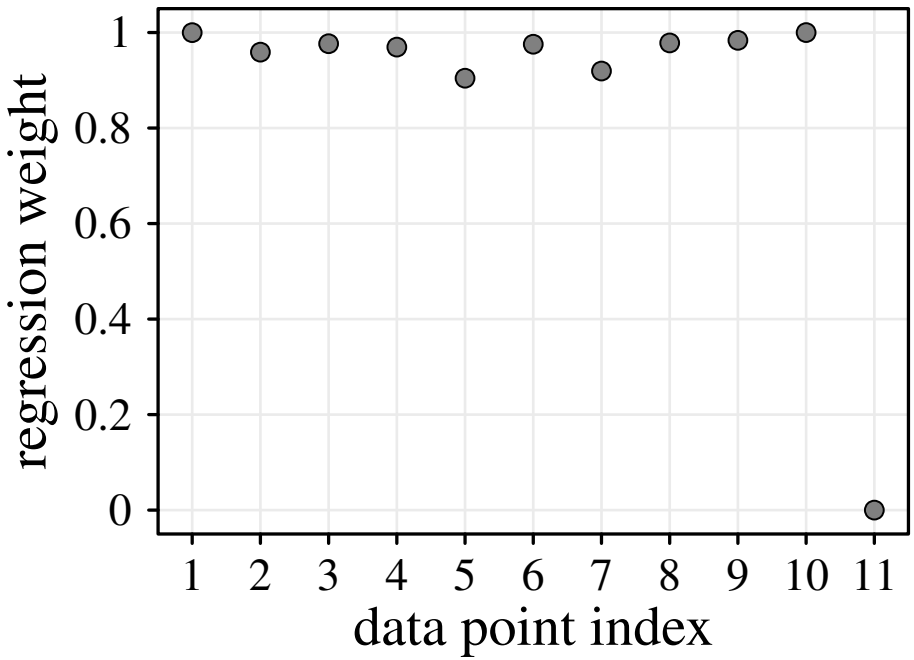
$\omega(e)$	
$(1-(\frac{e}{k})^2)^2$	if $\lvert e\rvert \leq k$
0	if $\lvert e\rvert > k$

For larger errors, the error measure ρ does not increase at all but remains constant.

$\Rightarrow$ Weights of extreme outliers drop to zero.

Least squares vs. robust regression





least squares and robust regression

Regression & nominal attributes

If most of the predictor variables are numerical and the few nominal attributes have small domains, a regression function can be constructed for each possible combination of the values of the nominal attributes, given that the data set is sufficiently large and covers all combinations.

Example

Attribute	Type/Domain
Sex	F/M
Vegetarian	Yes/No
Age	numerical
Height	numerical
Weight	numerical

Task: Predict the weight based on the other attributes.
Possible solution: Construct four separate regression functions for (F,Yes),(F,No),(M,Yes),(M,No) using only age and height as predictor variables.

Regression & nominal attributes

Alternative approach: Encode the nominal attributes as numerical attributes.

- Binary attributes can be encoded as $0/1$ or $-1/1$
- For nominal attributes with more than two values, a $0/1$ or $-1/1$ numerical attribute should be introduced for each possible value of the nominal attribute.
- Do not encode nominal attributes with more than two values in one numerical attribute, unless the nominal attribute is actually ordinal.

Classification as regression

A two-class classification problem (with classes 0 and 1) can be viewed as regression problem.

The regression function will usually not yield exact outputs 0 and 1, but the classification decision can be made by considering 0.5 as a cut-off value.

Problem: The objective functions aims at minimizing the function approximation error (for example, the mean squared error), but not misclassifications.

Classification as regression

Example

1000 data objects, 500 belonging to class 0, 500 to class 1.

- Regression function f yields 0.1 for all data from class 0 and 0.9 for all data from class 1.
- Regression function g always yields the exact and correct values 0 and 1, except for 9 data objects where it yields 1 instead of 0 and vice versa.

Regression function	Misclassifications	MSQE
f	0	0.01
g	9	0.009

From the viewpoint of regression g is better than f, from the viewpoint of misclassifications f should be preferred.

Logistic regression

Two class problem:

- Y: class attribute, $\mathrm{dom}(Y) = \{c_1, c_2\}$
  $\mathbf{X} = (X_1, \ldots, X_m)$ m-dimensional random vector
  $P(C = c_1 \mid \mathbf{X} = \mathbf{x}) = p(\mathbf{x})$,
  $P(C = c_2 \mid \mathbf{X} = \mathbf{x}) = 1 - p(\mathbf{x})$.

- **Given:** A set of data points $\{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$ each of which belongs to one of the two classes c_1 and c_2.

- **Desired:** A simple description of the probability function $p(\mathbf{x})$ for the given dataset $\mathbf{X}$.

- **Approach:** Describe the probability p by the logistic function:

$$p(\mathbf{x}) = \frac{1}{1 + e^{a_0 + \mathbf{ax}}} = \frac{1}{1 + \exp\left(a_0 + \sum_{i=1}^{m} a_i x_i\right)}$$

Classification: Logistic regression

By applying the logit-transformation we obtain

$$\ln\left(\frac{1-p(\mathbf{x})}{p(\mathbf{x})}\right) \quad = \quad a_0 + \mathbf{a}^\top \mathbf{x} \quad = \quad a_0 + \sum_{i=1}^{m} a_i x_i$$

that is, a multilinear regression problem, which can be solved with the introduced techniques.

But how do we determine the values $p(\mathbf{x})$ that enter the above equation?

- For a small data space with sufficient many realizations for every possible point the class probability can be estimated by the relative frequencies of the classes.
- If this is not the case, we may rely on an approach known as kernel estimation.

Kernel estimation

- **Idea:** Define an "influence function" (kernel), which describes how strongly a data point influences the probability estimate for neighboring points.

- **Gaussian kernel**

$$K(\mathbf{x}, \mathbf{y}) \quad = \quad \frac{1}{(2\pi\sigma^2)^{\frac{m}{2}}} \exp\left(-\frac{(\mathbf{x}-\mathbf{y})^\top(\mathbf{x}-\mathbf{y})}{2\sigma^2}\right)$$

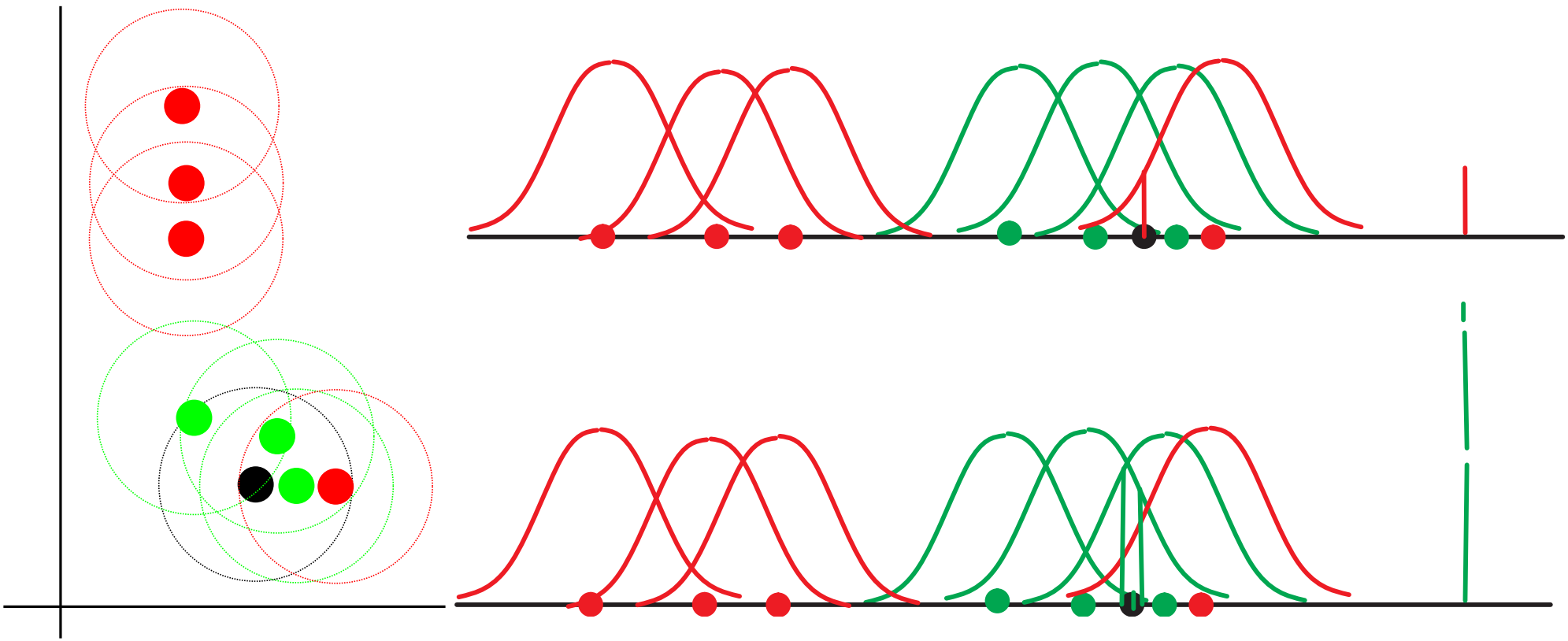
  where the variance σ^2 has to be chosen by a user.

Classification: Logistic regression

Kernel estimation applied to a two class problem:

$$\hat{p}(\mathbf{x}) = \frac{\sum_{i=1}^{n} c(\mathbf{x}_i) K(\mathbf{x}, \mathbf{x}_i)}{\sum_{i=1}^{n} K(\mathbf{x}, \mathbf{x}_i)}.$$

$$c(\mathbf{x}_i) = \begin{cases} 1 & \text{if } \mathbf{x_i} \text{ belongs to class } c_1 \\ 0 & \text{else} \end{cases}$$



Summary

- **Pros:**
  - Strong mathematical foundation
  - Simple to calculate and to understand (for a moderate number of dimensions)
  - High predictive accuracy

- **Cons:**
  - Many dependencies are non-linear
  - Global model does not adapt to locally different data distributions
    $\Rightarrow$ Locally weighted regression