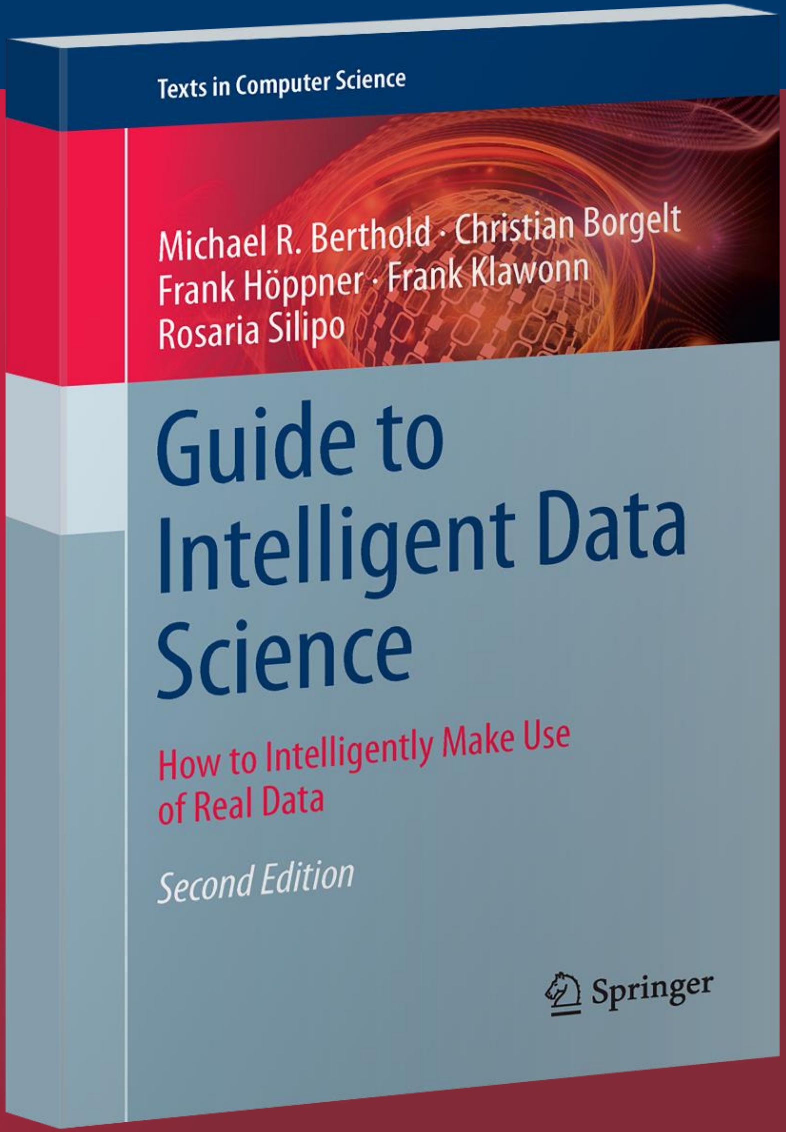
Regressions



Texts in Computer Science

Michael R. Berthold · Christian Borgelt
Frank Höppner · Frank Klawonn
Rosaria Silipo

Guide to Intelligent Data Science

How to Intelligently Make Use of Real Data

Second Edition

Springer

Caption

Summary of this lesson

“All models are approximations. Essentially, all models are wrong, but some are useful.”
-George Box

How can we model the data?

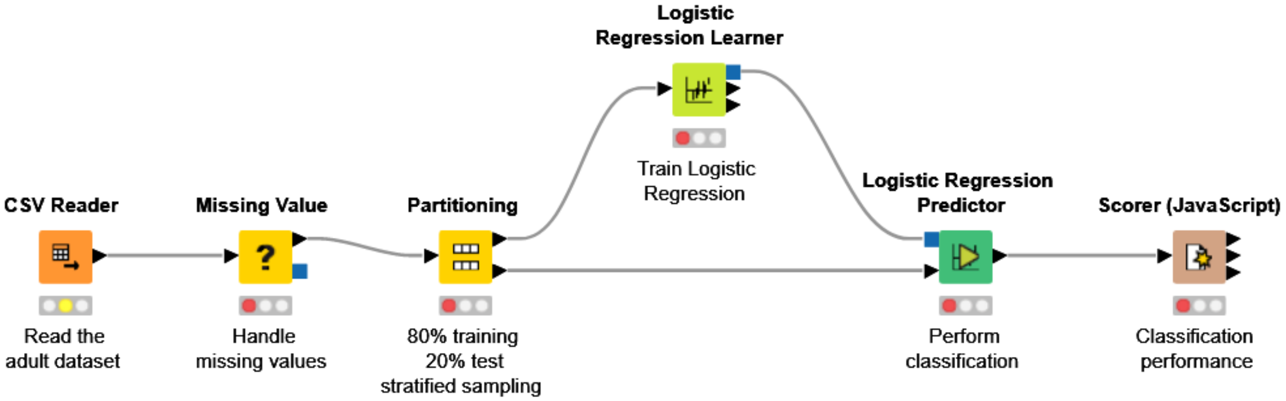
**This lesson refers to chapter 8 of the GIDS book*

Content of this Lesson

- Regression
  - The Regression Task
  - Linear Regression
  - Other Regressions
  - Logistic Regression
  - Robust Regression
  - Regression for Classification
  - Practical Example

Datasets

- Datasets used : adult dataset
- Example Workflow:
  - „Logistic regression“ https://kni.me/w/LWHdcrt_DFIepk0p
    - Missing value handling
    - Logistic regression


Logistic Regression Learner
Train Logistic Regression
CSV Reader
Read the adult dataset
Missing Value
Handle missing values
Partitioning
80% training 20% test stratified sampling
Logistic Regression Predictor
Perform classification
Scorer (JavaScript)
Classification performance

The Regression Task

Note

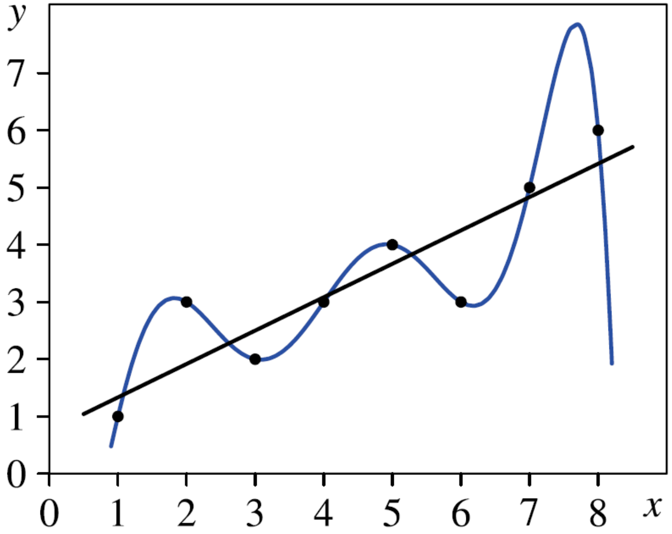
- We are focusing on methods that find explanations for an unknown dependency within the data.

- **Supervised** (because we know the desired outcome)
- **Descriptive** (because we care about explanation)

The Regression Task

- Goal: Explain how target attribute depends on descripitive attributes
  - Target attributes ➔ ***Response variable***
  - Descriptive attributes ➔ ***Regressor variables***

- As a parameterized function class *f*
  - Estimate parameters to describe the relationship
  - Must be simple enough for interpolation and extrapolation purposes
  - Example: Line (black) v.s. Polynomial (blue) with degree 7



The Regression Task: formally

Given a dataset $D = \{(\boldsymbol{x}_i, y_i) \mid i = 1, \dots, n\}$ with n tuples

- $\boldsymbol{x}$: Object description $[x_1, \dots, x_k]$
- y: Numerical target attribute

Find a function

$$f: \mathrm{dom}(x_1) \times \dots \times \mathrm{dom}(x_k) \rightarrow y \in \mathbb{R}$$

minimizing the error

$$E(f(x_1, \dots, x_k), y)$$

for all given n data objects $(\boldsymbol{x}_i, y_i)$.

Linear Regression

Regression Line

- Given a data set with two continuous attributes, x and y
- There is an approximate linear dependency between x and y

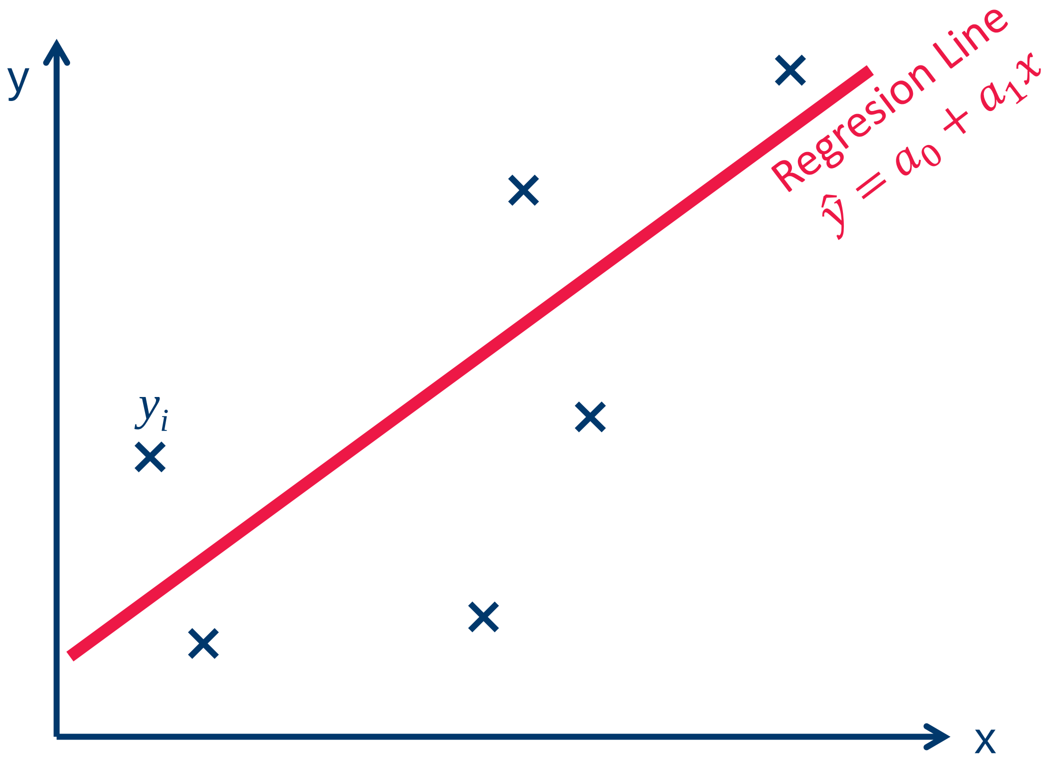
$$y \approx a + bx$$

- We find a **regression line** (i.e., determine the parameters a and b) such that the fits the data as well as possible

- Examples:
  - 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., spending vs. available income)

Regression Line

- What is a **good** fit?

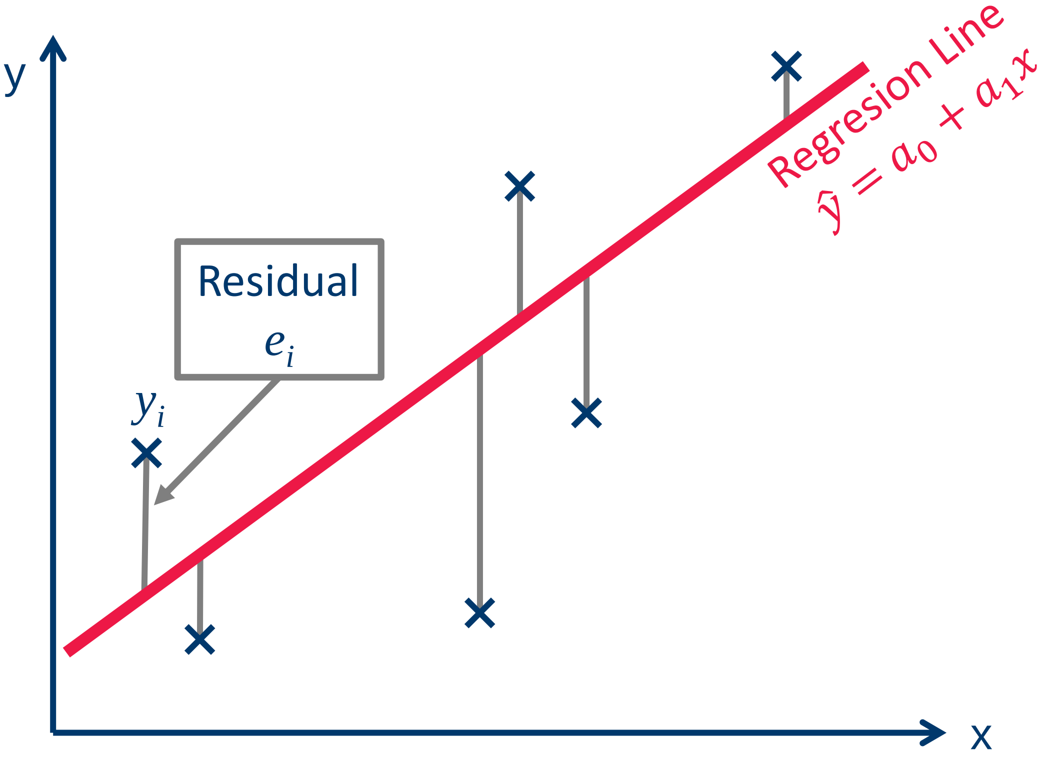


Cost Function

- The error, or the ***residual***, is calculated at each data point
- The sum of square errors (SSE) is chosen as cost function (to be minimized)

$$\sum_{i=1}^{n} e_i^2 = \sum_{i=1}^{n} (\widehat{y}_i - y_i)^2$$

- Referred as the ***least square method***



Cost Functions

- Sum of square errors

- Other reasonable cost functions
  - mean absolute distance
  - mean Euclidean distance
  - maximum absolute distance in *y*-direction (or equivalently: the
  - maximum squared distance in *y*-direction)
  - maximum Euclidean distance
  - . . .

Construction

- Think of a straight line $\hat{y} = f(x) = a + bx$
- Find a and b to model all observations (x_i, y_i) as close as possible
- ➔ SSE $F(a, b) = \sum_{i=1}^{n}(f(x) - y_i)^2 = \sum_{i=1}^{n}(a + bx_i - y_i)^2$ should be minimal
- **Goal**: The y-values that are computed with the linear equation should (squared and in total) deviate as little as possible from the measured values.

Construction

- SSE

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

is minimal if the partial derivatives w.r.t. a and b are 0

- That is:

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

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

Construction

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

$$na + \left(\sum_{i=1}^{n} x_i\right) b = \sum_{i=1}^{n} y_i$$

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

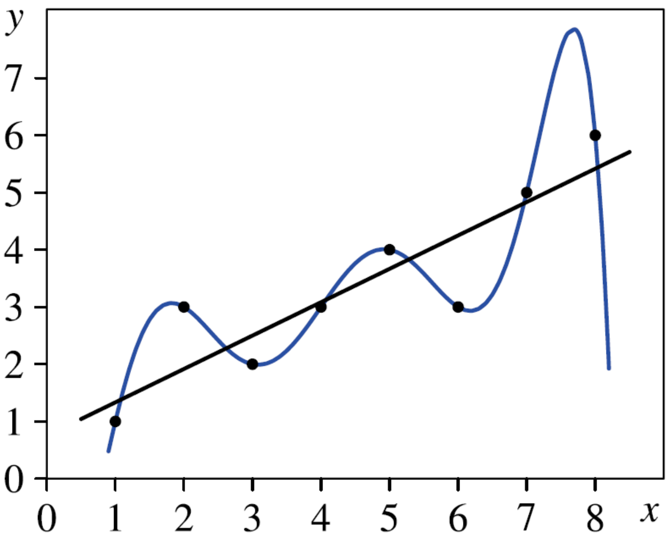
- that is, a two-equation system with two unknowns *a* and *b* which has a unique solution (if at least two different *x*-values exist).
- => A unique solution exists for a and b

Example – Regression Line

- Example: data

x	1	2	3	4	5	6	7	8
y	1	3	2	3	4	3	5	6

- Resulting regression line: $y = \frac{3}{4} + \frac{7}{12}x$



Least Squares and MLE

- The straight line determined in this way is called **regression line** for the data set *D.*
- A regression line can be interpreted as a **maximum likelihood estimator** (MLE):
- **Assumption:** The data generation process can be described by the model

$$f(x) = a + bx + \xi$$

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

- *The parameters that minimize 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:

$$f(y|x) = \frac{1}{2\pi\sigma^2} \exp\left(-\frac{\big(y-(a+bx)\big)^2}{2\sigma^2}\right)$$

- Leading to the likelihood function:

$$L\big((x_1,y_1),\ldots,(x_n,y_n);a,b,\sigma^2\big)$$

$$= \prod_{i=1}^{n} f(y_i|x_i)$$

$$= \prod_{i=1}^{n} \frac{1}{2\pi\sigma^2} \exp\left(-\frac{\big(y_i-(a+bx_i)\big)^2}{2\sigma^2}\right)$$

Least Squares and MLE

- To simplify the calculation of the derivatives to find the maximum, we compute the **logarithm**.

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

- After computing the derivatives w.r.t. the parameters *a* and *b*, we realize that maximizing the likelihood function is equivalent to minimizing

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

Other Regressions

Polynomial Regression

- Least square method can be extended to **polynomials** of degree m

$$y = p(x) = a_0 + a_1 + a_2x^2 + \cdots + a_mx^m$$

- Find a_i's that minimize the error function

$$F(a_0, a_1, \dots, a_m) = \sum_{i=1}^{n} (p(x) - y_i)^2$$
$$= \sum_{i=1}^{n} (a_0 + a_1 + a_2x^2 + \cdots + a_mx^m - y_i)^2$$

- We form the partial derivatives of this function w.r.t. the parameters $a_k, k = 1, 2, \cdots, m$, and equate them to zero

Multivariate Regression

- Given a dataset $D = \{(\boldsymbol{x}_i, y_i) \mid i = 1, \dots, n\}$ with n tuples
  - Input vector $\boldsymbol{x}_i = (x_{i1}, x_{i2}, \dots, x_{im})$ with multiple regressors
  - And corresponding response y_i
- For which we want to determine the linear regression function

$$y = f(x_1, x_2, \dots, x_m) = a_0 + \sum_{k=1}^{m} a_k x_k$$

- Examples:
  - Price of a house (y) depending on its size (x_1) and age (x_2)
  - Ice cream consumption (y) based on the temperature (x_1), the price (x_2), and the family income (x_3)
  - Electric consumption (y) 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

Multivariate Regression

- The cost function can be written as:

$$F(a_0, a_1, \dots, a_m) = \sum_{i=1}^{n} (f(\boldsymbol{x}_i) - y_i)^2$$

$$= \sum_{i=1}^{n} (a_0 + a_1 x_{i1} + a_2 x_{i2} + \cdots + a_m x_{im} - y_i)^2$$

Multivariate Regression

- It is convenient to write in the matrix form:

$$F(\boldsymbol{a}) = (\boldsymbol{X}\boldsymbol{a} - \boldsymbol{y})^T(\boldsymbol{X}\boldsymbol{a} - \boldsymbol{y})$$

- where

$$\boldsymbol{a} = \begin{pmatrix} a_0 \\ a_1 \\ \vdots \\ a_m \end{pmatrix} \qquad \boldsymbol{X} = \begin{pmatrix} 1 & x_{11} & \cdots & x_{1m} \\ 1 & x_{21} & \cdots & x_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & x_{n1} & \cdots & x_{nm} \end{pmatrix} = \begin{pmatrix} \boldsymbol{x}_1 \\ \boldsymbol{x}_2 \\ \vdots \\ \boldsymbol{x}_n \end{pmatrix} \qquad \boldsymbol{y} = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix}$$

$$\boldsymbol{x}_i = (1, x_{i1}, x_{i2}, \cdots, x_{im})$$

Multivariate Regression

- Find the minimum with the differential operator $\nabla_{\boldsymbol{a}}$

$$\nabla_{\boldsymbol{a}} = \left(\frac{\partial}{\partial a_0}, \frac{\partial}{\partial a_1}, \dots, \frac{\partial}{\partial a_m}\right)$$

- And find the solution to the equation

$$0 = \nabla_{\boldsymbol{a}} F(\boldsymbol{a}) = \nabla_{\boldsymbol{a}} (\boldsymbol{Xa} - \boldsymbol{y})^T (\boldsymbol{Xa} - \boldsymbol{y})$$

$$= \left(\nabla_{\boldsymbol{a}} (\boldsymbol{Xa} - \boldsymbol{y})\right)^T (\boldsymbol{Xa} - \boldsymbol{y}) + \left((\boldsymbol{Xa} - \boldsymbol{y})^T (\nabla_{\boldsymbol{a}} (\boldsymbol{Xa} - \boldsymbol{y}))\right)^{\boldsymbol{T}}$$

$$= \left(\nabla_{\boldsymbol{a}} (\boldsymbol{Xa} - \boldsymbol{y})\right)^T (\boldsymbol{Xa} - \boldsymbol{y}) + \left(\nabla_{\boldsymbol{a}} (\boldsymbol{Xa} - \boldsymbol{y})\right)^T (\boldsymbol{Xa} - \boldsymbol{y})$$

$$= 2\boldsymbol{X}^T (\boldsymbol{Xa} - \boldsymbol{y}) = 2\boldsymbol{X}^T \boldsymbol{Xa} - 2\boldsymbol{X}^T \boldsymbol{y}$$

- From which we obtain the system of normal equations:

$$\boldsymbol{X}^T \boldsymbol{Xa} = \boldsymbol{X}^T \boldsymbol{y}$$

Multivariate Regression

$$\boldsymbol{X}^T\boldsymbol{X}\boldsymbol{a} = \boldsymbol{X}^T\boldsymbol{y}$$

- The system is uniquely solvable iff $\boldsymbol{X}^T\boldsymbol{X}$ is invertible (nonsingular)
- In this case we have:

$$\boldsymbol{a} = (\boldsymbol{X}^T\boldsymbol{X})^{-1}\ \boldsymbol{X}^T\boldsymbol{y} = \boldsymbol{X}^+\boldsymbol{y}$$

- Moore-Penrose pseudo-inverse
  - The expression $(\boldsymbol{X}^T\boldsymbol{X})^{-1}\ \boldsymbol{X}^T = \boldsymbol{X}^+$ is also known as the (Moore-Penrose) pseudo-inverse of the matrix $\boldsymbol{X}$.
  - Pseudo-inverse matrices are used to compute the inverse of singular matrices.
  - They provide a least square solution to a system of linear equations without a unique solution.

Regression vs. Time Series Analysis

- **Regression**
  - Targets y & set of input features
  - No time order information
  - Describing the relationship between the target and input features
  - Model → interpolation

- **Time series analysis**
  - **Time** ordered sequence of observations
  - Predicting future observations from:
    - Past values in time series
    - Accompanying time series
  - Model → extrapolation

Nonlinear Regression

Solving equations based on partial derivatives of the cost function does not work in some cases with:

- Non-differentiable cost function (absolute value, maximum, etc)
- No analytical solution for equations

Nonlinear Regression

Example

- Nonlinear model $y = ae^{bx}$ (radioactive decay, growth of bacteria, …)
- Then the cost function and their partial derivatives are

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

$$\frac{\partial F}{\partial a} = 2\sum_{i=1}^{n} \left(ae^{bx_i} - y_i\right)e^{bx_i}$$

$$\frac{\partial F}{\partial b} = 2\sum_{i=1}^{n} \left(ae^{bx_i} - y_i\right)ax_ie^{bx_i}$$

Possible solutions:

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

Logistic Regression

Transformation

- Nonlinear regression functions can be transformed, and solved as a linear regression
- Example:

$$y = ax^b$$

- Can be transformed by taking the natural log of the equation

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

- Notice the sum of squared error is minimized only in the log-transformed space (i.e., $x' = \ln x$, $y' = \ln y$)

Logit Transformation

Let’s consider another transformation

- **Logistic functions** describe limited growth processes, and defined as

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

- The inverse of this function (**logit function**) produces a linear model

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

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

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

Logit Transformation

- **logit function**

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

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

- Fitting the data to this model is often referred as **logistic regression**

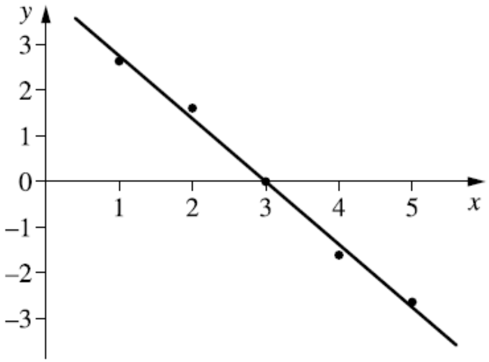
Example – Logit Transformation

- The data

x	1	2	3	4	5
y	0.4	1.0	3.0	5.0	5.6

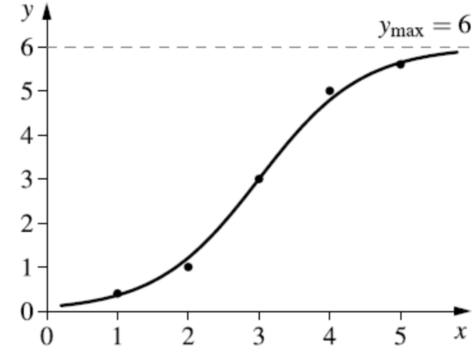
- Can be transformed with a logit-transformation, and the linear regression line is fitted to

$$z = logit(y) = 4.133 - 1.3775x$$



- We can transform y back with the logistic function, and obtain the logistic regression curve

$$y = \frac{6}{1 + e^{4.133 - 1.3775x}}$$



Model vs. Black-box

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

- 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).

Generalization

- 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
- The same function should also be able to generalize it to make correct predictions on new data.

- The regression function “learns” a description of the data, not of the structure of the data.
- The prediction using a complex regression function can be worse than the prediction using a simpler regression function (overfitting).

Robust Regression

Robust Regression

- Ordinary regression – sensitive to outliers
- Solution: ***robust regression***
- Let’s re-write the cost function as

$$F(\boldsymbol{a}) = (\boldsymbol{X}\boldsymbol{a} - \boldsymbol{y})^T(\boldsymbol{X}\boldsymbol{a} - \boldsymbol{y}) = \sum_{i=1}^{n} \rho\,(e_i) = \sum_{i=1}^{n} \rho\left(\boldsymbol{x}_i^T \boldsymbol{a} - y_i\right)$$

- For the least square method, the function ρ is a square function
- (i.e., $\rho(e) = e^2$)

Robust Regression

- More generally, the ρ function can be any function satisfying the following:

$$\rho(e) \geq 0,$$

$$\rho(0) = 0,$$

$$\rho(e) = \rho(-e),$$

$$\rho(e_i) \geq \rho(e_j) \quad \text{if } |e_i| \geq |e_j|.$$

- Parameter estimation with a cost function with a ρ function satisfying these conditions are called an **M-estimator**.

M-Estimators

- Calculate the derivatives w.r.t. the parameters a_i in

$$\sum_{i=1}^{n} \rho\left(e_i\right) = \sum_{i=1}^{n} \rho\left(\boldsymbol{x}_i^T \boldsymbol{a} - y_i\right)$$

- We find the solution to the system of linear equations

$$\sum_{i=1}^{n} \psi_i\left(\boldsymbol{x}_i^T \boldsymbol{a} - y_i\right)\boldsymbol{x}_i^T = 0$$

- Where $\psi = \rho'$. If we define $w(e) = \psi(e)/e$ and $w_i = w(e_i)$,

$$\sum_{i=1}^{n} \frac{\psi_i\left(\boldsymbol{x}_i^T \boldsymbol{a} - y_i\right)}{e_i} \cdot e_i \cdot \boldsymbol{x}_i^T = \sum\nolimits_{i=1}^{n} w_i e_i^2 \boldsymbol{x}_i^T = 0$$

- The solution is the same as the standard least squares problem with weights $\sum_{i=1}^{n} w_i e_i^2$

M-Estimators

Problem in finding the solution:

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

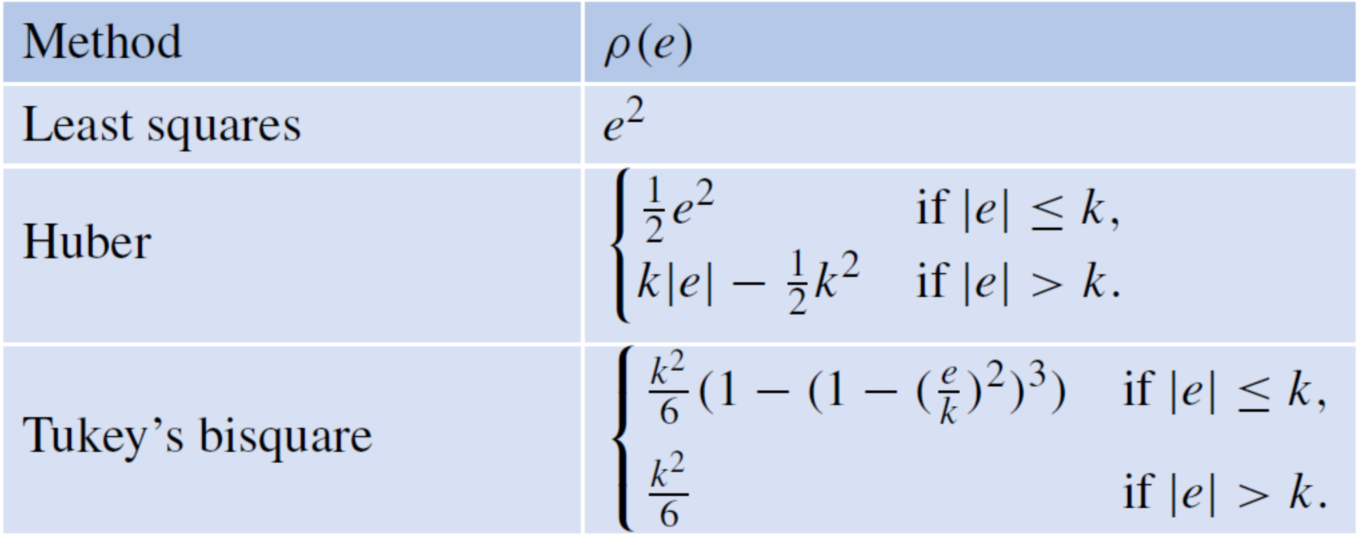
Strategy: alternating optimization

1. Choose an initial solution $\boldsymbol{a}^{(0)}$, (e.g., standard least squares solution) and set all weights to $w_i = 1$
2. At step t, calculate the residuals $e^{(t-1)}$ and the corresponding weights $w^{(t-1)} = w(e^{(t-1)})$
3. Compute the solution to the weighted least squared problem

$$\boldsymbol{a}^{(0)} = \left(\boldsymbol{X}^T \boldsymbol{W}^{(t-1)} \boldsymbol{X}\right)^{-1} \boldsymbol{X}^T \boldsymbol{W}^{(t-1)} \boldsymbol{y}$$

- Where $\boldsymbol{W}$ is a diagonal matrix with weights w_i on the main diagonal

Methods of Robust Regression



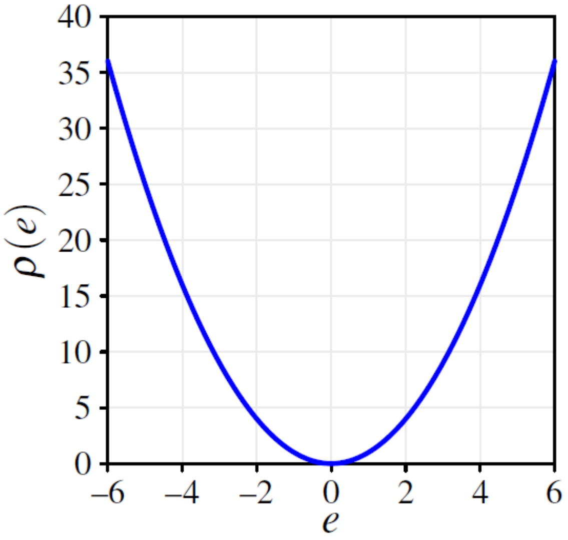
Method	$\rho(e)$
Least squares	e^2
Huber	$\begin{cases} \frac{1}{2}e^2 & \text{if } \lvert e\rvert \leq k, \\ k\lvert e\rvert - \frac{1}{2}k^2 & \text{if } \lvert e\rvert > k. \end{cases}$
Tukey's bisquare	$\begin{cases} \frac{k^2}{6}(1-(1-(\frac{e}{k})^2)^3) & \text{if } \lvert e\rvert \leq k, \\ \frac{k^2}{6} & \text{if } \lvert e\rvert > k. \end{cases}$

- Where parameter k needs to be chosen for Huber and Tukey’s bisquare

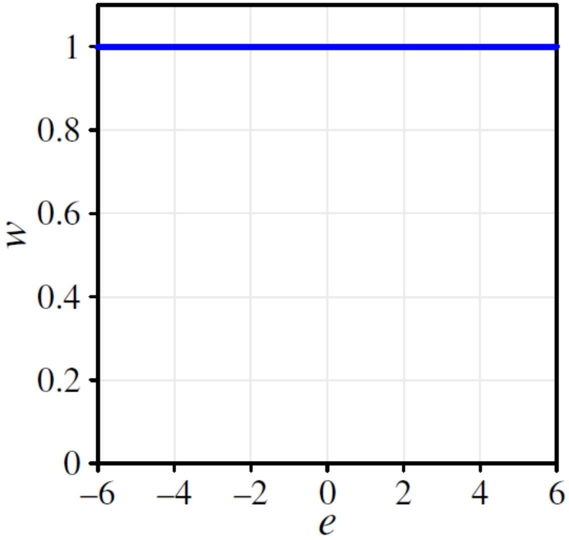
Least Squares

- The error measure ρ increases in a quadratic manner with increasing deviation

→ Extreme outliers have full influence



$\rho(e) = e^2$

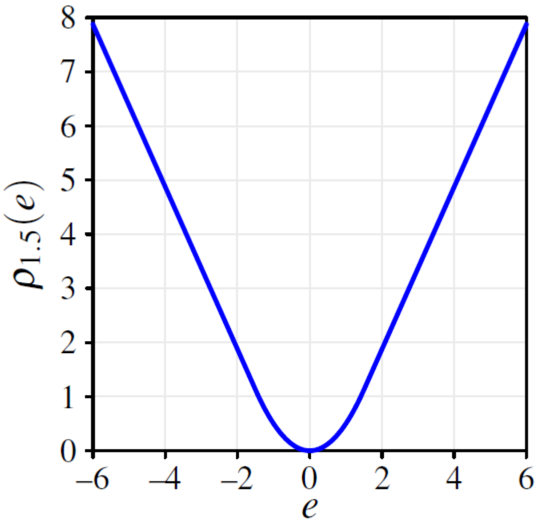


$\omega(e) = 1$

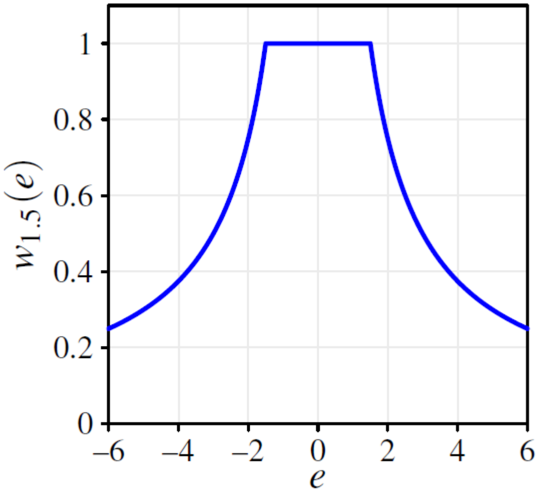
Huber (k=1.5)

- The error measure ρ switches from quadratic (for small errors) to linear (for large errors)

→ Only data points with small error have full influence



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

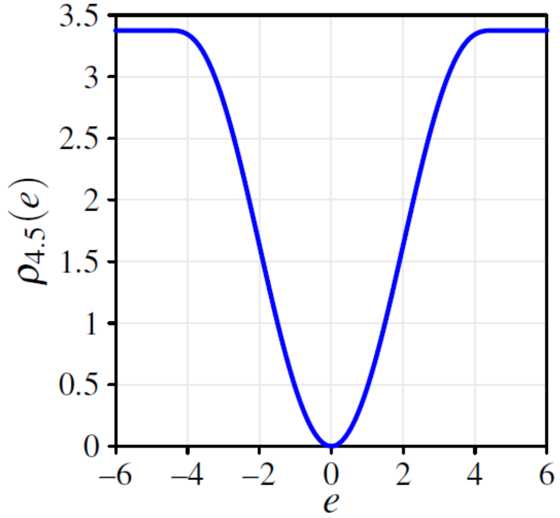


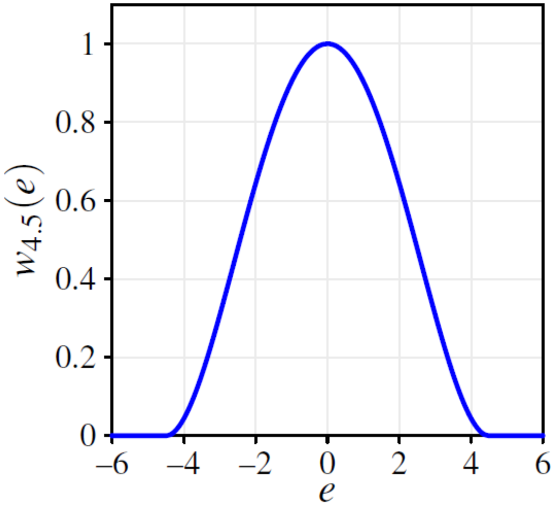
$\omega(e)$	
1	if $\lvert e\rvert \leq k$
$\frac{k}{\lvert e\rvert}$	if $\lvert e\rvert > k$

Tukey’s Bisquare (k=4.5)

- The error measure ρ does not increase for large errors

→ Weights of extreme outliers drop to zero



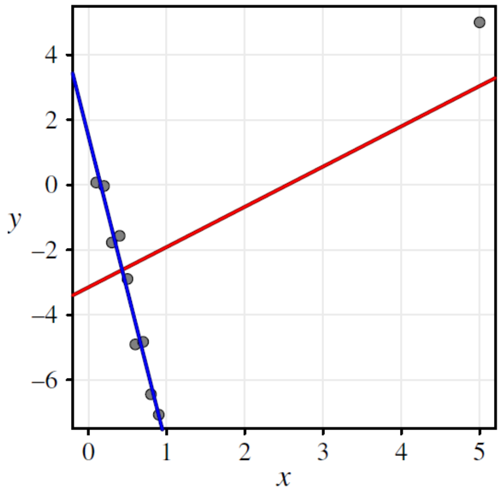


$\rho(e)$	
$\frac{k^2}{6}\left(1-\left(1-\left(\frac{e}{k}\right)^2\right)^3\right)$	if $\lvert e\rvert \leq k$
$\frac{k^2}{6}$	if $\lvert e\rvert > k$

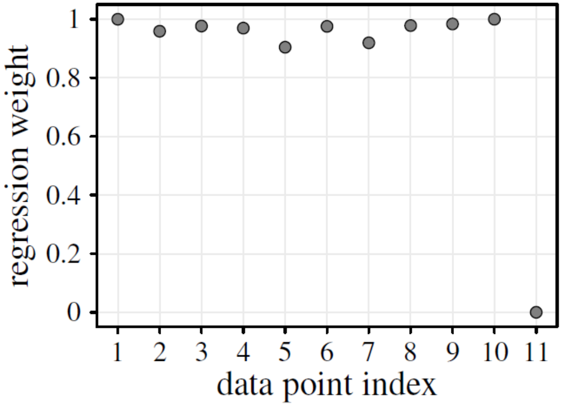
$\omega(e)$	
$(1-(\frac{e}{k})^2)^2$	if $\lvert e\rvert \leq k$
0	if $\lvert e\rvert > k$

Least Squares vs. Robust Regression

- An extreme outlier influences the regression line in least squares
- The influence of the outlier is attenuated in robust regression



- Reduced weight is apparent in a plot of regression weights in robust regression



Regression for Classification

Regression and nominal attributes

If:

- most of the predictor variables are numerical,
- and the few nominal attributes have small domains, and
- the data set is sufficiently large and covers all combinations.

then we can construct a regression function for each possible combination of the values of the nominal attributes.

Example:

Attribute	Type / Domain
sex	F/M
vegetarian	yes/no
Age	numerical
Height	numerical
Weight	numerical

Possible solution to predict weight: four regression functions for (F,Yes),(F,No),(M,Yes),(M,No) using only age and height as predictor variables.

Regression and 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 (1-of-*n* coding).
- 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 (classes 0 vs. 1) can be viewed as a regression problem

Challenges:

- A regression function usually cannot produce outcomes 0 or 1
- The cost functions aim to reduce the numerical error (measured as squared residuals, for example), not misclassification

Solution:

- A regression model for the probability of belonging to a certain class
- A probability cut-off (e.g, probability > 0.5) can be used for classification

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	Mis-classifications	MSE
f	0	0.01
g	9	0.009

- From the viewpoint of regression g is better than f (smaller MSE), from the viewpoint of misclassifications f should be preferred.

Logistic Regression: Two-class problem

- Two-Class Problem:
- If Y belongs to one of two classes $\{c_1, c_2\}$, then we can model the probability for one class only

$$P(Y = c_1 \mid X = \boldsymbol{x}) = p(\boldsymbol{x})$$

$$P(Y = c_2 \mid X = \boldsymbol{x}) = 1 - p(\boldsymbol{x})$$

- **Given**: A set of data points $\{\boldsymbol{x}_1, .., \boldsymbol{x}_n\}$ each assigned to one of the two classes c_1 and c_2.
- **Desired:** Train a function, which gives us the probability $p(\boldsymbol{x})$ for each class (0 and 1) based on the input features for the given dataset.

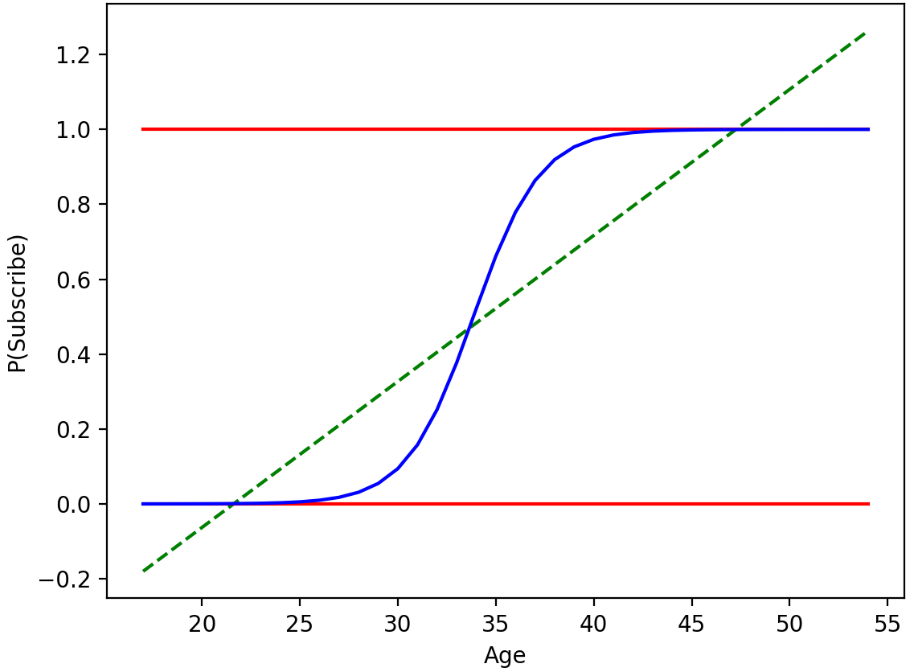
Linear Regression vs. Logistic Regression

	Linear Regression	**Logistic Regression**
Target variable *y*	Numeric $y \in (-\infty, \infty)/[a, b]$	***Nominal*** $y \in \{0, 1, 2, 3\}/\{red, white\}$
Functional relationship between features and…	… target value y $y = f(x_1, \dots, x_n, \beta_0, \dots, \beta_n)$ $y = \beta_0 + \beta_1 x_1 + \cdots + \beta_n x_n$	… ***class probability P (y = class i)*** $P(y = c_i) = f(x_1, \dots, x_n, \beta_0, \dots, \beta_n)$

Goal: Find the regression coefficients $\beta_0, \dots, \beta_n$

Example where Linear Regression Fails

- **Result:** $p(subscribe) = -0.84 + 0.04\, age$
- **Problem:** $p(subscribe) < 0$ for age $= 20$ and $p(subscribe) > 1\; for\; age = 50$

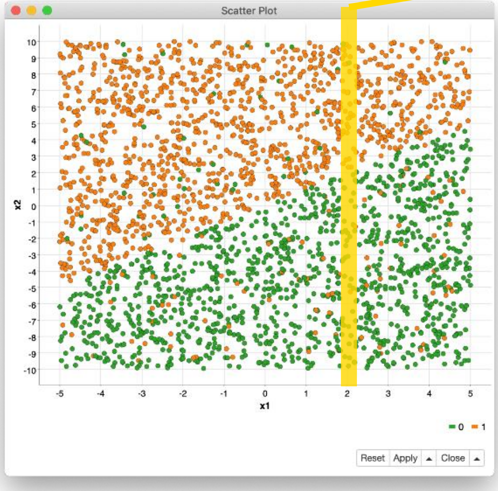


Age

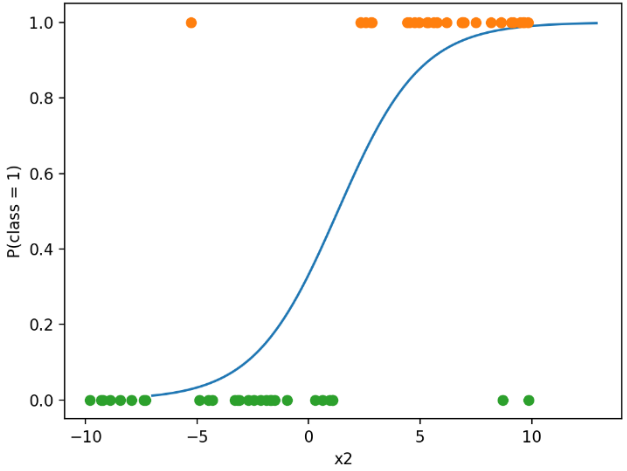
Let’s Find Out How Binary Logistic Regression Works!

$$P(y = 1) = f(x_1, x_2; \beta_0, \beta_1, \beta_2) := \frac{1}{1 + e^{-(\beta_0+\beta_1 x_1+\beta_2 x_2)}}$$

Feature space



Probability function given $x_1 = 2$



Logistic Regression: two-class problem

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

$$p(\boldsymbol{x}) = \frac{1}{1 + \exp(a_0 + \sum_{j=1}^{m} a_j x_j)}$$

- By applying the logit-transformation, we have a multivariate regression problem

$$\ln\left(\frac{1 - p(\boldsymbol{x})}{p(\boldsymbol{x})}\right) = a_0 + \sum_{j=1}^{m} a_j x_j$$

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

Logistic Regression: Two-class problem

How do we determine class probability $p(\boldsymbol{x})$ for this regression problem?

– If we have sufficiently *many* realizations for all possible data points

→ $p(\boldsymbol{x})$ can be estimated by the relative frequencies of the classes

– If there aren’t many realizations, we rely on ***kernel estimation***

Kernel Estimation

- **Idea**: Define an “influence function” (kernel), which describes how strongly a data point influences the probability estimate for neighboring points.
- The “influence” is stronger from a closer point, weaker for a distant point
- The “influence” is modeled by a kernel function
- Example: **Gaussian kernel**

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

- Where $\boldsymbol{y}$ is a neighbor of $\boldsymbol{x}$
- Higher (or lower) influence if $\boldsymbol{x}$ and $\boldsymbol{y}$ are closer (or farther)
- Variance σ^2 has to be chosen by the user.

Kernel Estimation

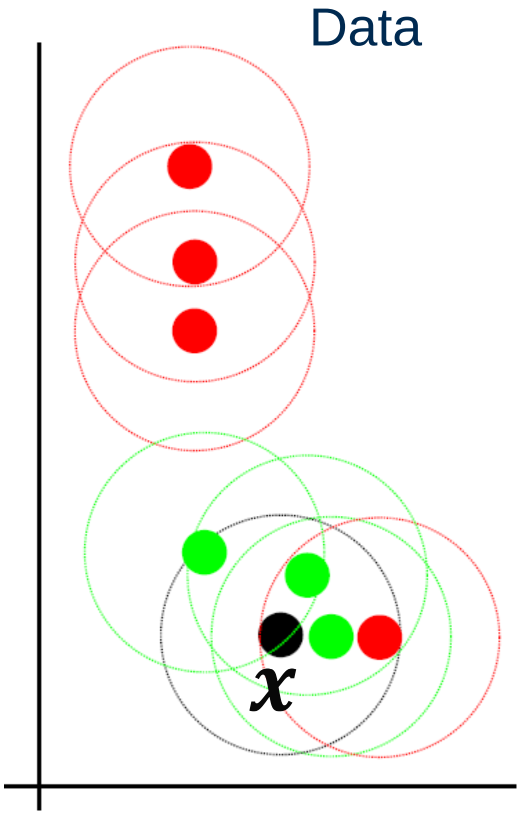
- Kernel estimation for a two-class problem

→ $p(\boldsymbol{x})$ is estimated as the sum of $k(\cdot,\cdot)$ between $\boldsymbol{x}$ and all other data points belonging to class c_1

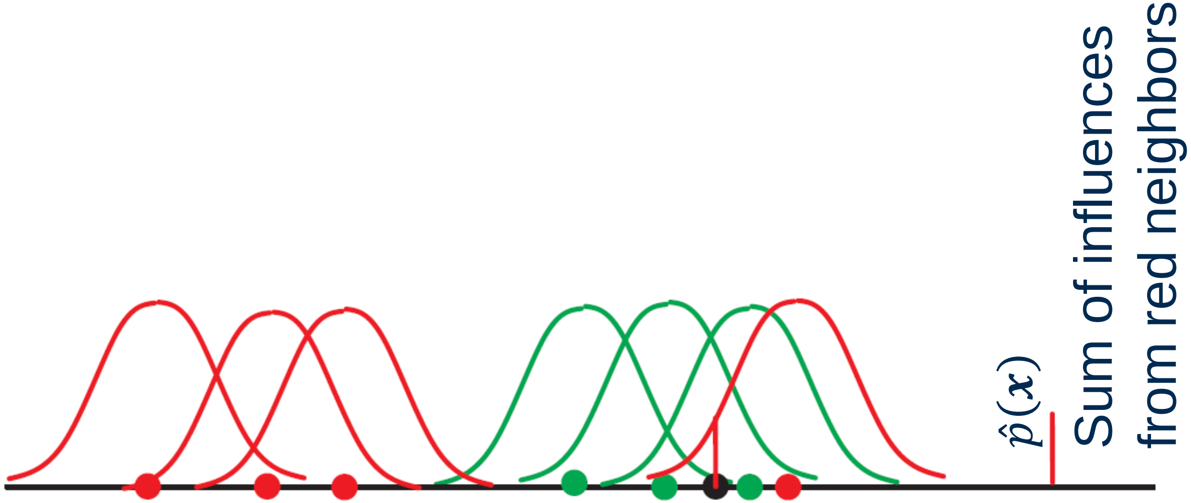
$$\hat{p}(\boldsymbol{x}) = \frac{\sum_{i=1}^{n} c(\boldsymbol{x}_i) K(\boldsymbol{x}, \boldsymbol{x}_i)}{\sum_{i=1}^{n} K(\boldsymbol{x}, \boldsymbol{x}_i)}$$

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

Example – Kernel Estimation

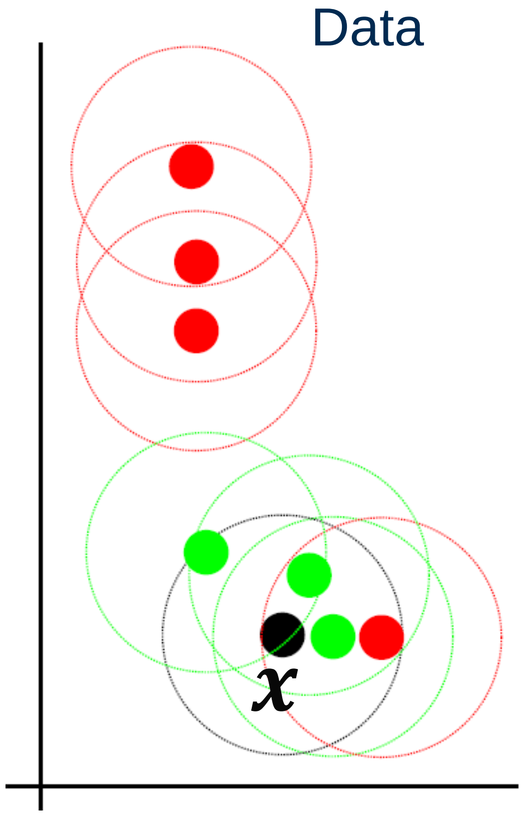


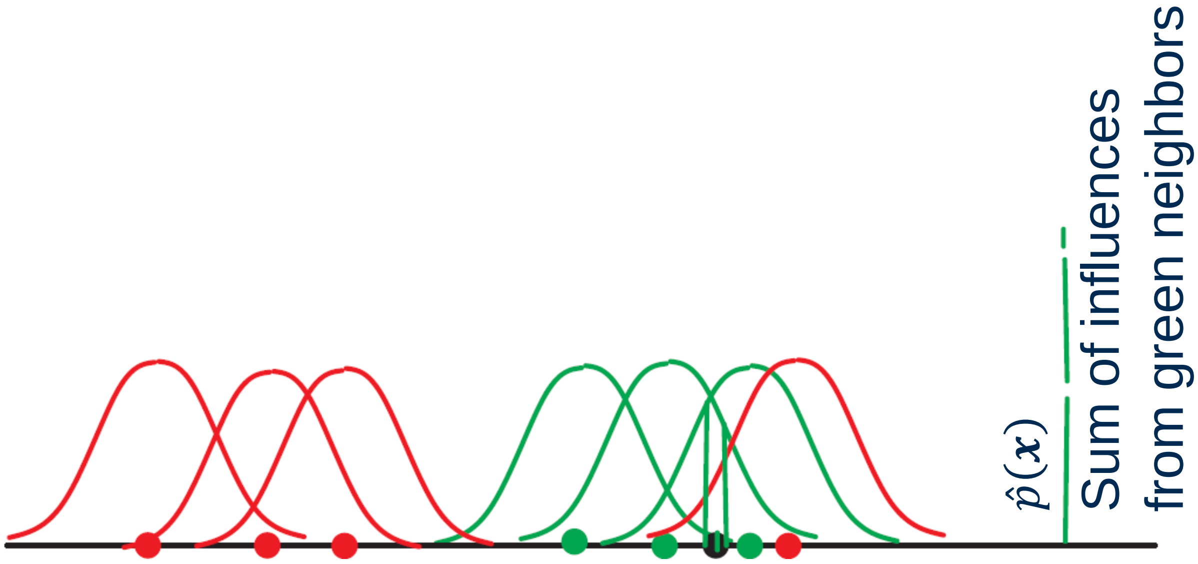
- If red≡c_1, we calculate the sum of kernel functions between $\boldsymbol{x}$ and all red neighbors


$\hat{p}(\boldsymbol{x})$
Sum of influences from red neighbors

Example – Kernel Estimation

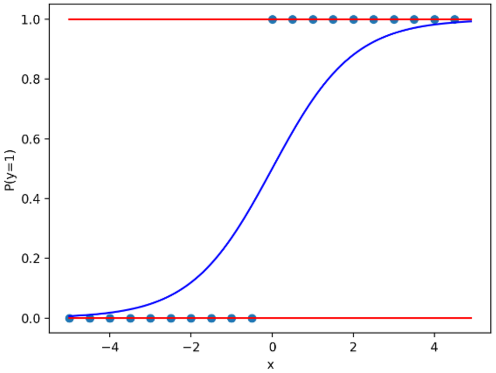
- If green≡c_1, we calculate the sum of kernel functions between $\boldsymbol{x}$ and all green neighbors



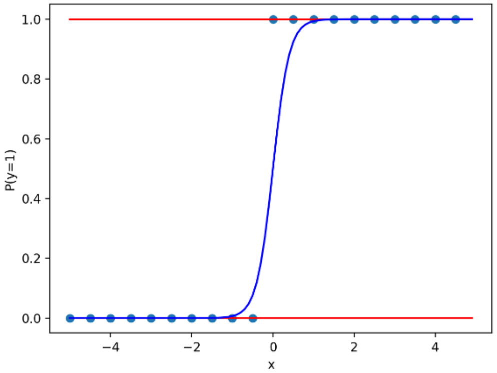

$\hat{p}(\boldsymbol{x})$
Sum of influences from green neighbors

Regularization in Logistic Regression

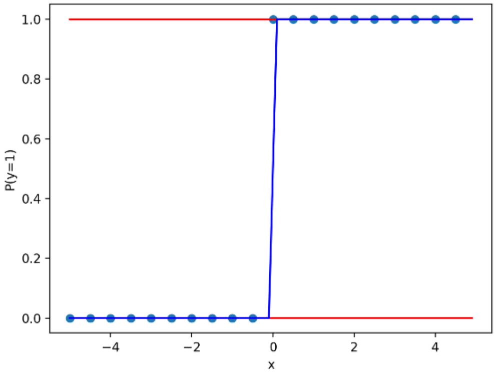
- Is there a way to handle overfitting?



$$P(y=1)=\frac{1}{1+e^{-x}}$$



$$P(y=1)=\frac{1}{1+e^{-5x}}$$



$$P(y=1)=\frac{1}{1+e^{-100x}}$$

- If data are linearly separable, coefficients becomes extremely large

→ *Overfitting*

Regularization in Logistic Regression

- The parameters in a logistic regression model is determined by maximizing the likelihood function
- Or equivalently, minimizing the (negative) log-likelihood function
- To avoid overfitting: add regularization by penalizing large coefficients
- Estimate of coefficient vector β obtained by:

$$\hat{\beta} = \min_{\beta}\{-LL(\beta, y, \boldsymbol{x}) + \lambda R(\beta)\}$$

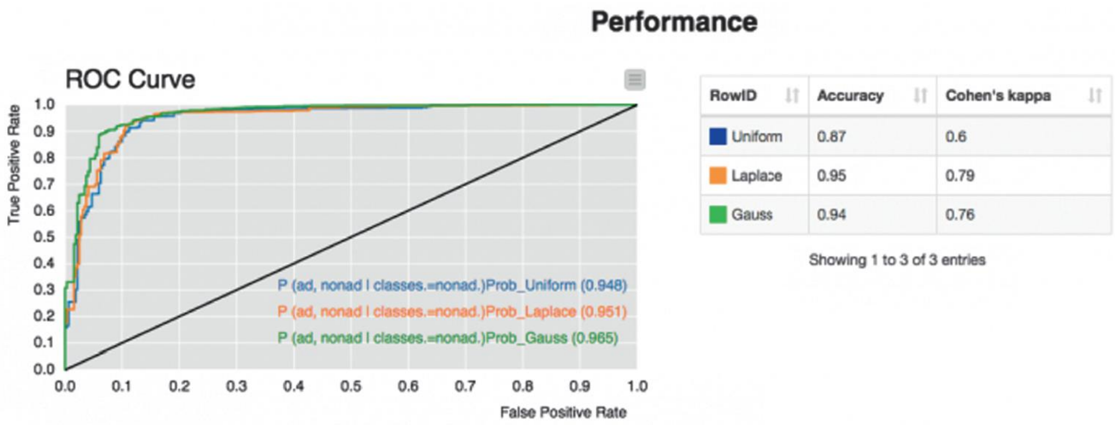
- The choice of the regularization term $R(\beta)$: **Gauss**, **Laplace**, **L1**, **L2**, etc.

Regularization Example

- Internet Advertisement Data – UCI Machine Learning Repository
- More features (680) than samples (n=120)

→ Prone to overfitting

- Logistic regression with no regularization (uniform) (blue), Laplace (orange), and Gauss (green)



Performance

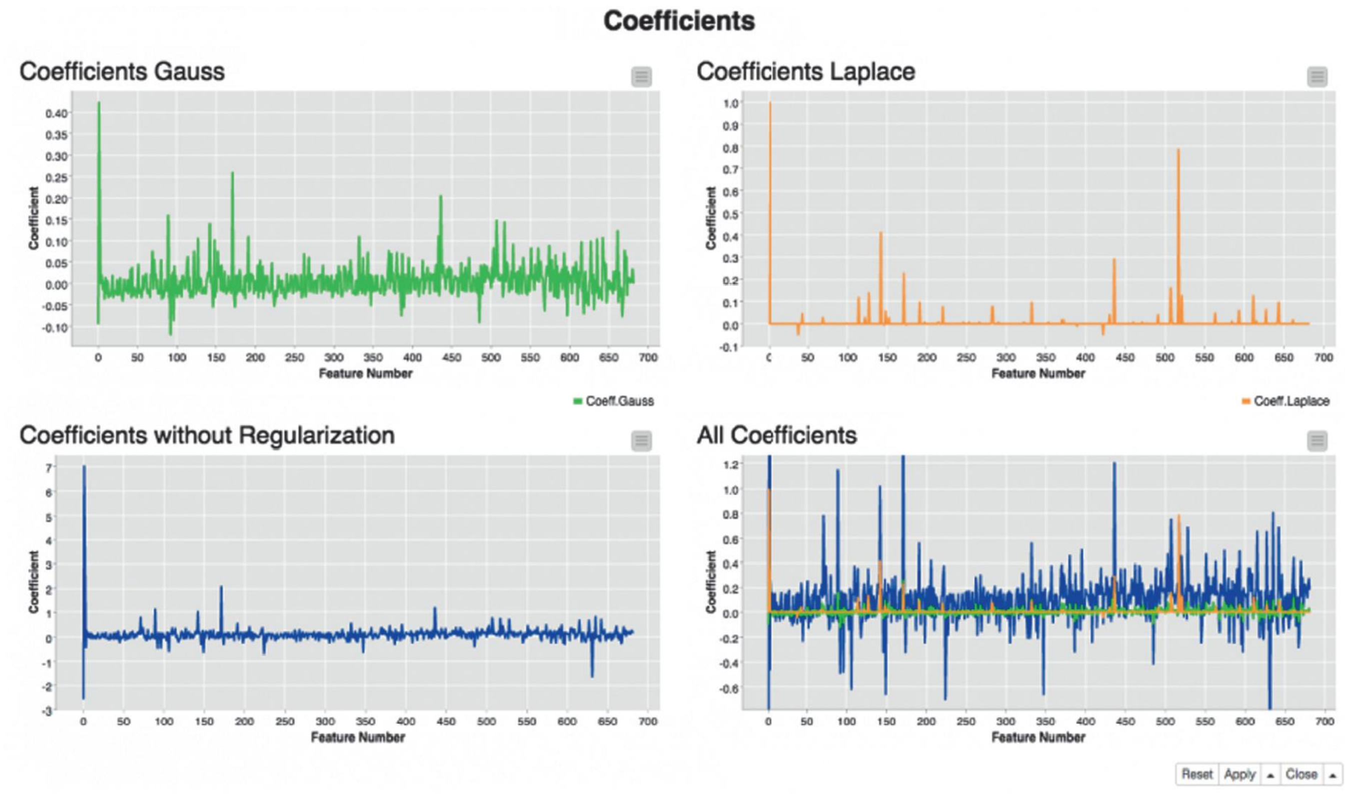
ROC Curve
True Positive Rate
1.0
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0.0
False Positive Rate
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1.0
P (ad, nonad | classes.=nonad.)Prob_Uniform (0.948)
P (ad, nonad | classes.=nonad.)Prob_Laplace (0.951)
P (ad, nonad | classes.=nonad.)Prob_Gauss (0.965)

RowID	Accuracy	Cohen's kappa
🟦 Uniform	0.87	0.6
🟧 Laplace	0.95	0.79
🟩 Gauss	0.94	0.76

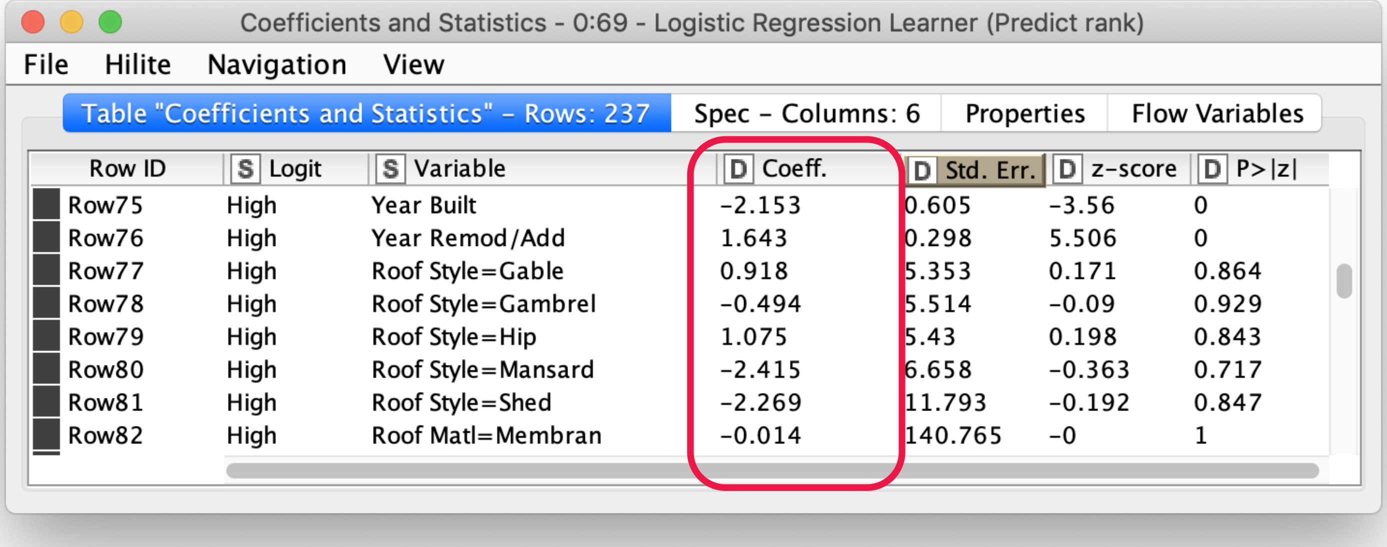
Showing 1 to 3 of 3 entries

Regularization Example

- Without regularization → large regression coefficients



Interpretation of the Coefficients

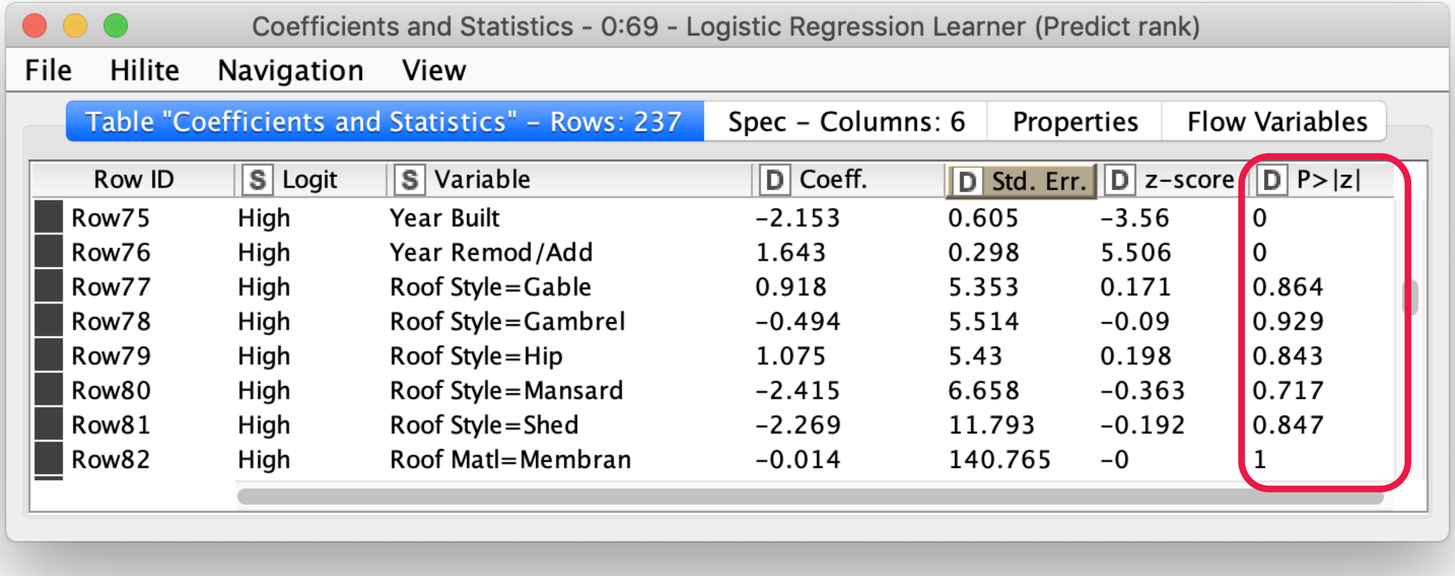


Coefficients and Statistics - 0:69 - Logistic Regression Learner (Predict rank)
File
Hilite
Navigation
View
Table "Coefficients and Statistics" – Rows: 237
Spec – Columns: 6
Properties
Flow Variables

Row ID	S Logit	S Variable	D Coeff.	D Std. Err.	D z-score	D P>|z|
Row75	High	Year Built	-2.153	0.605	-3.56	0
Row76	High	Year Remod/Add	1.643	0.298	5.506	0
Row77	High	Roof Style=Gable	0.918	5.353	0.171	0.864
Row78	High	Roof Style=Gambrel	-0.494	5.514	-0.09	0.929
Row79	High	Roof Style=Hip	1.075	5.43	0.198	0.843
Row80	High	Roof Style=Mansard	-2.415	6.658	-0.363	0.717
Row81	High	Roof Style=Shed	-2.269	11.793	-0.192	0.847
Row82	High	Roof Matl=Membran	-0.014	140.765	-0	1

- Interpretation of the sign
  - $\beta_i > 0$: Bigger x_i lead to higher probability
  - $\beta_i < 0$: Bigger x_i lead to smaller probability

Interpretation of the p Value



Coefficients and Statistics - 0:69 - Logistic Regression Learner (Predict rank)
File
Hilite
Navigation
View
Table "Coefficients and Statistics" – Rows: 237
Spec – Columns: 6
Properties
Flow Variables

Row ID	S Logit	S Variable	D Coeff.	D Std. Err.	D z-score	D P>|z|
Row75	High	Year Built	-2.153	0.605	-3.56	0
Row76	High	Year Remod/Add	1.643	0.298	5.506	0
Row77	High	Roof Style=Gable	0.918	5.353	0.171	0.864
Row78	High	Roof Style=Gambrel	-0.494	5.514	-0.09	0.929
Row79	High	Roof Style=Hip	1.075	5.43	0.198	0.843
Row80	High	Roof Style=Mansard	-2.415	6.658	-0.363	0.717
Row81	High	Roof Style=Shed	-2.269	11.793	-0.192	0.847
Row82	High	Roof Matl=Membran	-0.014	140.765	-0	1

– p- value < α: input feature has a significant impact on the dependent variable.

Summary – Regression

Pros:

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

Cons:

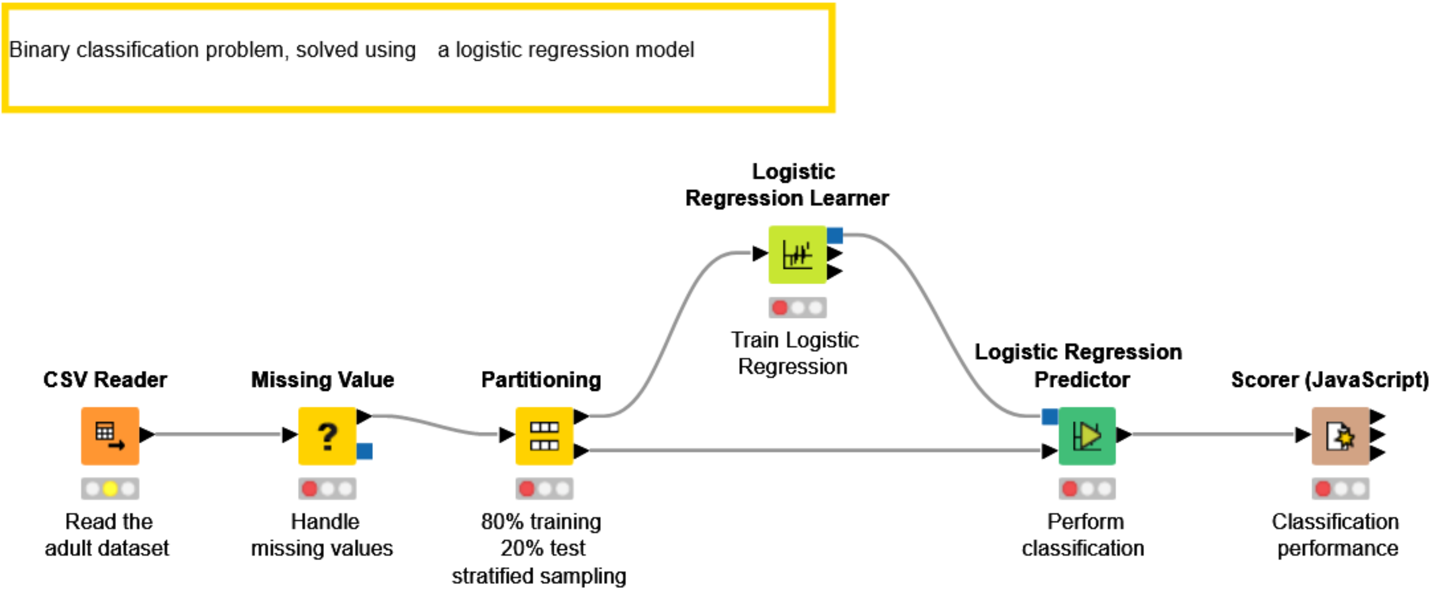
- Many dependencies are non-linear
- Global model does not adapt to locally different data distributions

Summary - Regression

- Logistic regression is used for classification problems
- The regression coefficients are calculated by maximizing the likelihood function, which has no closed form solution, hence iterative methods are used.
- Regularization can be used to avoid overfitting.
- The p-value shows us whether an independent variable is significant

Practical Example with KNIME Analytics Platform

Logistic Regression



Binary classification problem, solved using a logistic regression model

CSV Reader
Read the adult dataset
Missing Value
Handle missing values
Partitioning
80% training 20% test stratified sampling
Logistic Regression Learner
Train Logistic Regression
Logistic Regression Predictor
Perform classification
Scorer (JavaScript)
Classification performance

- Training and application of a logistic regression model. Notice the Missing Value node to fix possible missing values in the data

For any questions please contact: email@email.com