Linear Regression

Machine Learning 1 — Lecture 4

9th April 2024

Robert Peharz
Institute of Theoretical Computer Science
Graz University of Technology

Regression: Predict a *continuous* **target variable** from one or more **input variables**. Hence, a form of **supervised learning.**

input variables (also **independent variables**, **features**):

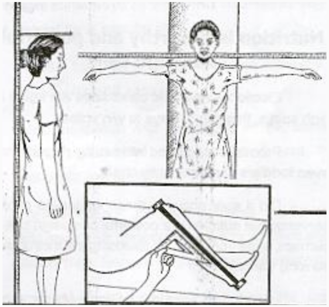
$$x_1, x_2, \ldots, x_D, \text{ collected in a vector } \boldsymbol{x} = (x_1, x_2, \ldots, x_D)^T$$

target variable (also **dependent variable**, **output variable**, or **ground truth**):

$$y$$

Height Regression

Example



knee height [cm]	arm span [cm]	body height [cm]
57	181	185
55	177	177
53	167	176
...	...	...

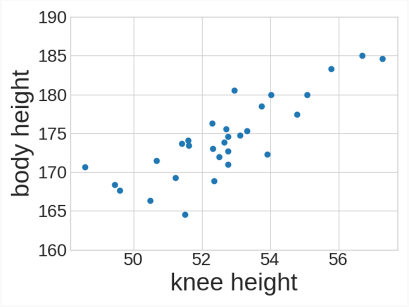
- You need to measure the body height of a hospitalized patient, who is bed tied and cannot stand to take a measurement
- Say you have collected data about the knee length, arm span and height of N persons
- You might now learn to predict patients' height from their arm span and knee height, thus "indirectly measure" the height of the hospitalized patient

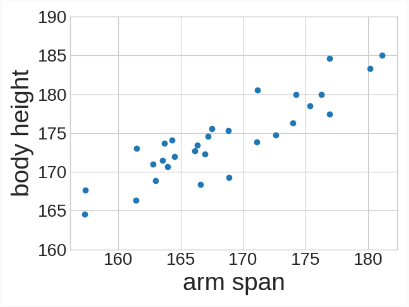
Height Regression – Data

Example

knee height [cm]	arm span [cm]	height [cm]
57	181	185
55	177	177
53	167	176
...	...	...

$N = 30$ data points

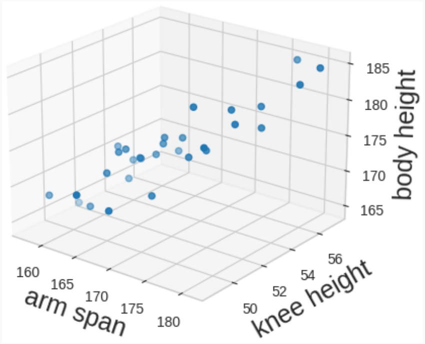




Height Regression – Data

knee height [cm]	arm span [cm]	height [cm]
57	181	185
55	177	177
53	167	176
...	...	...

$N = 30$ data points



Regression

The target y is a **"noisy" function** of the input variables $\boldsymbol{x}$:

$$y = f(\boldsymbol{x}) + e \qquad\qquad e: \text{ noise term}$$

Assume we have a **training data set** with N examples:

$$\mathcal{D} = \left\{(\boldsymbol{x}^{(1)}, y^{(1)}), (\boldsymbol{x}^{(2)}, y^{(2)}), \ldots, (\boldsymbol{x}^{(N)}, y^{(N)})\right\}$$

$\boldsymbol{x}^{(i)} \in \mathbb{R}^D$, $\boldsymbol{x}^{(i)} = (x_1^{(i)}, \ldots, x_D^{(i)})^T$: D-dimensional vector
$y^{(i)} \in \mathbb{R}$: target value
$(\boldsymbol{x}^{(i)}, y^{(i)})$: i^{th} **training example** (input-output pair)

Goal: Learn the function f from $\mathcal{D}$.

Hypothesis Space

Searching over **all possible** functions $f\colon \mathbb{R}^D \mapsto \mathbb{R}$ is challenging.

Thus, one usually considers a **restricted** class of functions, the so-called **hypothesis space** $\mathcal{H}$ (also called **model class**, **function space**, etc.).

Often, the hypothesis space is given via some **parametrized function**

$$f_{\boldsymbol{\theta}}(\boldsymbol{x}) = f(\boldsymbol{x}, \boldsymbol{\theta})$$

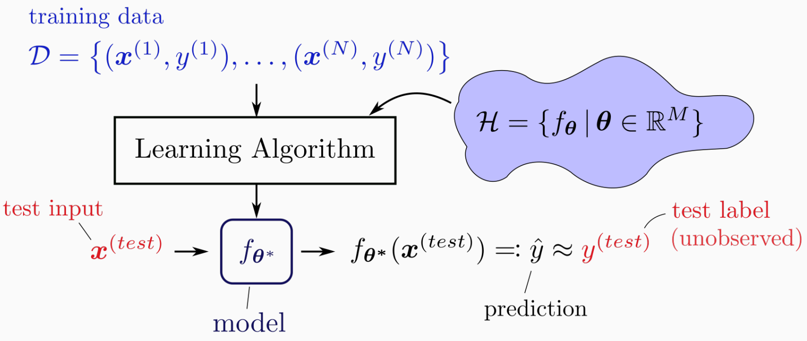
where $\boldsymbol{\theta} = (\theta_1, \ldots, \theta_M)^T$ is an M-dimensional **parameter vector**.

Each $\boldsymbol{\theta} \in \mathbb{R}^M$ describes some function. Formally, the hypothesis space is the **set of functions**

$$\mathcal{H} = \{f_{\boldsymbol{\theta}} \,|\, \boldsymbol{\theta} \in \mathbb{R}^M\}$$

where $\boldsymbol{\theta}$ ranges over all possible M-dimensional vectors.

The Overall Picture


training data
$\mathcal{D} = \{(\boldsymbol{x}^{(1)}, y^{(1)}), \dots, (\boldsymbol{x}^{(N)}, y^{(N)})\}$
Learning Algorithm
$\mathcal{H} = \{f_{\boldsymbol{\theta}} \mid \boldsymbol{\theta} \in \mathbb{R}^M\}$
test input
$\boldsymbol{x}^{(test)}$
$f_{\boldsymbol{\theta}^*}$
model
$f_{\boldsymbol{\theta}^*}(\boldsymbol{x}^{(test)}) =: \hat{y} \approx y^{(test)}$
prediction
test label
(unobserved)

(Same picture generally applies to **supervised learning**, not only regression)

Linear Regression

When the hypothesis space is restricted to all **affine functions**

$$\mathcal{H} = \left\{ f_{\boldsymbol{\theta}}(\boldsymbol{x}) := \boldsymbol{w}^T \boldsymbol{x} + b \,|\, \boldsymbol{w} \in \mathbb{R}^D, b \in \mathbb{R} \right\}$$

we speak of **linear regression**.[*] Thus, each function is of the form

$$f_{\boldsymbol{\theta}}(\boldsymbol{x}) = \boldsymbol{w}^T \boldsymbol{x} + b = \left(\sum_{d=1}^{D} w_d x_d \right) + b$$

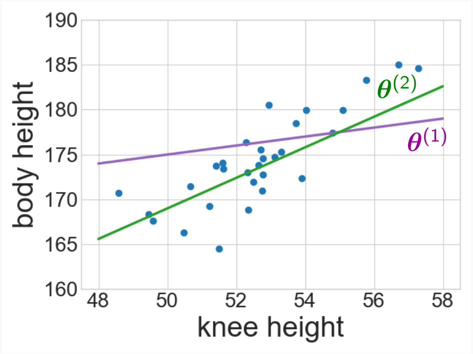
The parameter vector $\boldsymbol{\theta}$ consists of **bias** b and **weight vector** $\boldsymbol{w} = (w_1, w_2, \ldots, w_D)^T$:

$$\boldsymbol{\theta} = (b, w_1, w_2, \ldots, w_D)^T \in \mathbb{R}^{D+1}$$

[*] The distinction between *affine* and *linear* function is sloppy at times.

How to pick suitable parameters? Example

knee height [cm]	height [cm]
57	185
55	177
53	176
...	...



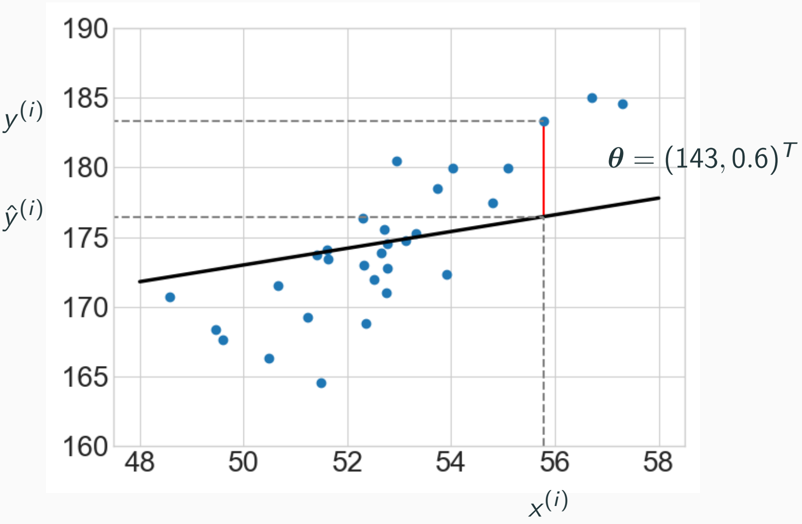
- Predict height from knee height (1-d input)
- Hypothesis class consists of functions $f_{\theta}(x) = wx + b$
- Parameters: $\theta = (b, w)^T \in \mathbb{R}^2$
- Consider $\theta^{(1)} = (150, 0.5)^T$ and $\theta^{(2)} = (84, 1.7)^T$
- Which one is better? In what sense it is better?

Squared Error

- Consider an arbitrary training point $\boldsymbol{x}^{(i)}$, $y^{(i)}$
- Consider parameters $\boldsymbol{\theta} = (b, \overbrace{w_1, \dots, w_D}^{\boldsymbol{w}^T})^T$
- The **predicted target** is $\hat{y}^{(i)} =: f_{\boldsymbol{\theta}}(\boldsymbol{x}^{(i)}) = \boldsymbol{w}^T \boldsymbol{x}^{(i)} + b$
- The **squared error** (ℓ_2 **error**) between **predicted** and **true** target is

$$\big(\underbrace{\hat{y}^{(i)} - y^{(i)}}_{\text{prediction error}}\big)^2 = \left(f_{\boldsymbol{\theta}}(\boldsymbol{x}^{(i)}) - y^{(i)}\right)^2 = \left(\boldsymbol{w}^T \boldsymbol{x}^{(i)} + b - y^{(i)}\right)^2$$

Squared Error



Least Squares Loss Function

- We want the squared error to be small on all training examples
- Idea: average the squared error over the whole data set
- This yields the **least-squares loss function** (**ℓ_2 loss**):

$$\begin{aligned}\mathcal{L}(\boldsymbol{\theta}) &= \frac{1}{N}\sum_{i=1}^{N}\left(\hat{y}^{(i)} - y^{(i)}\right)^2 \\ &= \frac{1}{N}\sum_{i=1}^{N}\left(f_{\boldsymbol{\theta}}(\boldsymbol{x}^{(i)}) - y^{(i)}\right)^2 \\ &= \frac{1}{N}\sum_{i=1}^{N}\left(\boldsymbol{w}^T\boldsymbol{x}^{(i)} + b - y^{(i)}\right)^2\end{aligned}$$

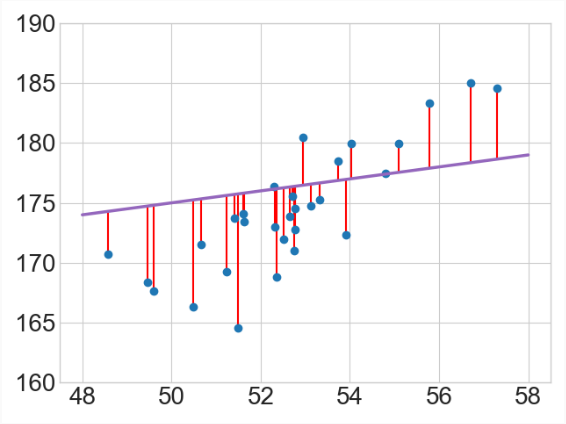
- A **loss function** (**objective**, **cost function**) is a notion of **fitness**
- The lower $\mathcal{L}(\boldsymbol{\theta})$, the better $\boldsymbol{\theta}$

Least Squares Loss

$$\mathcal{L}(\boldsymbol{\theta}) = \frac{1}{N}\sum_{i=1}^{N}\left(f_{\boldsymbol{\theta}}(\boldsymbol{x}^{(i)}) - y^{(i)}\right)^2$$

$\boldsymbol{\theta}^{(1)} = (150, 0.5)^T$

$\mathcal{L}(\boldsymbol{\theta}^{(1)}) = 22.67$



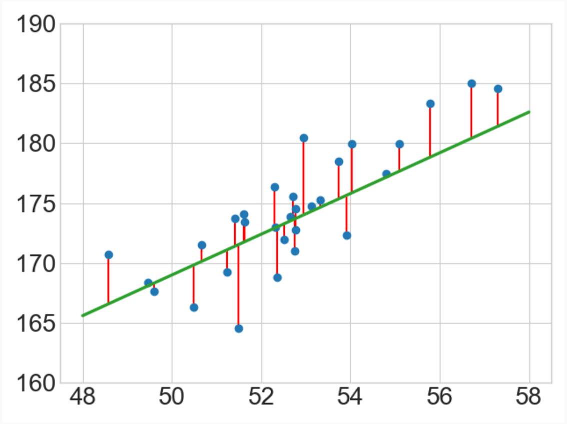
Least Squares Loss

Example

$$\mathcal{L}(\boldsymbol{\theta}) = \frac{1}{N} \sum_{i=1}^{N} \left(f_{\boldsymbol{\theta}}(\boldsymbol{x}^{(i)}) - y^{(i)} \right)^2$$

$\boldsymbol{\theta}^{(2)} = (84, 1.7)^T$

$\mathcal{L}(\boldsymbol{\theta}^{(2)}) = 9.28$

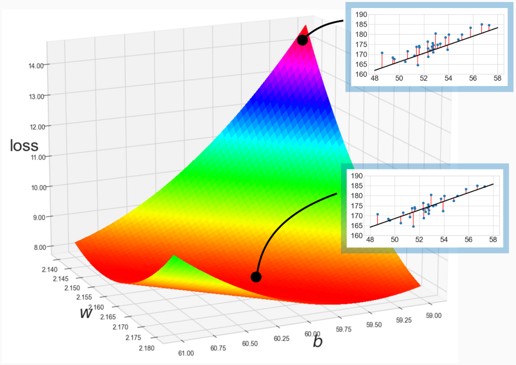


Loss Function Illustrated

Example

- ranging over all $\boldsymbol{\theta}$ yields a "**loss landscape**"
- least squares: **quadratic function**
- **convex** ("bowl shaped")
- for convex functions, every **local minimum** is a **global minimum**

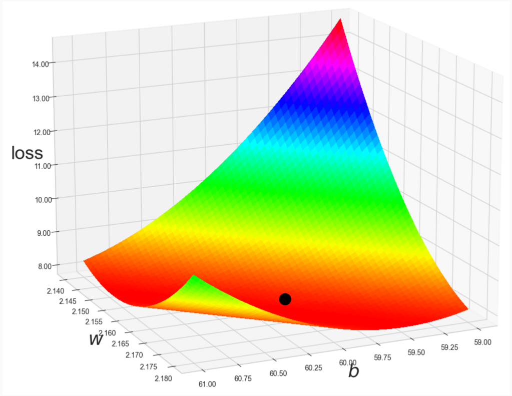
$$\mathcal{L}(\boldsymbol{\theta}) = \frac{1}{N}\sum_{i=1}^{N}\left(f_{\boldsymbol{\theta}}(\boldsymbol{x}^{(i)}) - y^{(i)}\right)^2$$



Least Squares — Analytic Solution

Minimum of $\mathcal{L}(\theta)$ (Quadratic Function)

The regression loss $\mathcal{L}(\boldsymbol{\theta})$ has (under regularity conditions) a **unique** stationary point, which is its **unique global minimum**.



Thus, we find the **least-squares solution** $\boldsymbol{\theta}^*$ by setting $\nabla_{\boldsymbol{\theta}}\mathcal{L} = 0$.

Absorbing the Bias

The model is an affine function

$$f_{\boldsymbol{\theta}}(\boldsymbol{x}) = \boldsymbol{w}^T \boldsymbol{x} + b$$

We can simplify notation a bit, by expanding the input vector by a **constant** (**"dummy feature"**): $\boldsymbol{x} := (1, x_1, x_2, \ldots, x_D)^T$

$$f_{\boldsymbol{\theta}}(\boldsymbol{x}) = \sum_{i=1}^{D} \theta_i x_i + \theta_0 \, 1 = \boldsymbol{\theta}^T \boldsymbol{x},$$

where $\boldsymbol{\theta} = (b, w_1, \ldots, w_D)^T$.

Least-Squares in Matrix Form

- Collect all $\boldsymbol{x}^{(i)}$ as rows in a $N \times (D+1)$ matrix:

$$\boldsymbol{X} = \begin{pmatrix} \boldsymbol{x}^{(1)\,T} \\ \boldsymbol{x}^{(2)\,T} \\ \vdots \\ \boldsymbol{x}^{(N)\,T} \end{pmatrix} = \begin{pmatrix} 1 & x_1^{(1)} & x_2^{(1)} & \dots & x_D^{(1)} \\ 1 & x_1^{(2)} & x_2^{(2)} & \dots & x_D^{(2)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_1^{(N)} & x_2^{(N)} & \dots & x_D^{(N)} \end{pmatrix}$$

- $\boldsymbol{X}$ is called the **design matrix**
- The vector of predictions is

$$\hat{\boldsymbol{y}} = \begin{pmatrix} \hat{y}^{(1)} \\ \hat{y}^{(2)} \\ \vdots \\ \hat{y}^{(N)} \end{pmatrix} = \begin{pmatrix} \boldsymbol{x}^{(1)\,T}\boldsymbol{\theta} \\ \boldsymbol{x}^{(2)\,T}\boldsymbol{\theta} \\ \vdots \\ \boldsymbol{x}^{(N)\,T}\boldsymbol{\theta} \end{pmatrix} = \begin{pmatrix} \boldsymbol{x}^{(1)\,T} \\ \boldsymbol{x}^{(2)\,T} \\ \vdots \\ \boldsymbol{x}^{(N)\,T} \end{pmatrix} \boldsymbol{\theta} = \boldsymbol{X}\boldsymbol{\theta}$$

Least-Squares in Matrix Form cont'd

- Collect all targets $y^{(i)}$ in a vector:

$$\boldsymbol{y} = \begin{pmatrix} y^{(1)} \\ y^{(2)} \\ \vdots \\ y^{(N)} \end{pmatrix}$$

- The vector of **prediction errors** is given as

$$\boldsymbol{e} := \begin{pmatrix} \hat{y}^{(1)} - y^{(1)} \\ \hat{y}^{(2)} - y^{(2)} \\ \vdots \\ \hat{y}^{(N)} - y^{(N)} \end{pmatrix} = \hat{\boldsymbol{y}} - \boldsymbol{y} = \boldsymbol{X\theta} - \boldsymbol{y}$$

Least-Squares in Matrix Form cont'd

- The least-squares objective can now be written

$$\begin{aligned}\mathcal{L}(\boldsymbol{\theta}) = \frac{1}{N}\sum_{i=1}^{N}\left(\hat{y}^{(i)} - y^{(i)}\right)^2 &= \frac{1}{N}\sum_{i=1}^{N} e^{(i)}e^{(i)} \\ &= \frac{1}{N}\boldsymbol{e}^T\boldsymbol{e} \\ &= \frac{1}{N}\left(\boldsymbol{X}\boldsymbol{\theta} - \boldsymbol{y}\right)^T\left(\boldsymbol{X}\boldsymbol{\theta} - \boldsymbol{y}\right)\end{aligned}$$

- **Quadratic function** in $\boldsymbol{\theta}$
- The gradient can be shown to be

$$\nabla_{\boldsymbol{\theta}}\mathcal{L} = \frac{1}{N}2\boldsymbol{X}^T\left(\boldsymbol{X}\boldsymbol{\theta} - \boldsymbol{y}\right)$$

- Compare this with the 1-d case: $\frac{d}{\mathrm{d}\theta}(x\theta - y)^2 = 2x(x\theta - y)$

Closed Form Solution

Setting the gradient to zero yields the minimum:

$$\begin{aligned}\nabla_{\boldsymbol{\theta}}\mathcal{L}(\boldsymbol{\theta}) =& \frac{1}{N}2\boldsymbol{X}^T(\boldsymbol{X}\boldsymbol{\theta} - \boldsymbol{y}) \overset{!}{=} 0 \\ & \boldsymbol{X}^T(\boldsymbol{X}\boldsymbol{\theta} - \boldsymbol{y}) = 0\end{aligned}$$

Hence, we have to solve a linear system of equations:

$$\begin{aligned}\boldsymbol{X}^T\boldsymbol{X}\theta^* &= \boldsymbol{X}^T\boldsymbol{y} \\ \theta^* &= \underbrace{(\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T}_{\text{Moore-Penrose Inverse}}\boldsymbol{y}\end{aligned}$$

$(\boldsymbol{X}^T\boldsymbol{X})^{-1}$ exist, if the columns of $\boldsymbol{X}$ are **linearly independent** (i.e., when there are no “redundant features”).

Closed Form Solution

Example

n	knee height	height
1	56.7	185.0
2	54.8	177.4
3	52.7	175.5
4	51.5	164.6
5	51.6	173.4
6	52.3	173.0
7	52.8	174.5
8	50.7	171.5
9	53.1	174.8
10	53.9	172.3
11	49.6	167.6
12	55.1	180.0
13	54.0	179.9
14	57.3	184.6
15	53.3	175.3
16	52.8	172.7
17	52.5	171.9
18	50.5	166.3
19	52.9	180.5
20	52.4	168.8
21	49.5	168.4
22	52.7	173.8
23	55.8	183.3
24	52.3	176.3
25	51.2	169.3
26	52.8	171.0
27	53.7	178.5
28	51.6	174.1
29	48.6	170.7
30	51.4	173.7

Closed Form Solution

Example

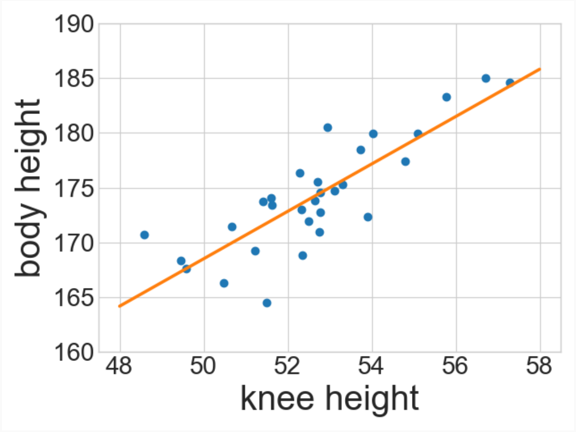
$$\boldsymbol{X} = \underbrace{\begin{pmatrix} 1 & 56.7 \\ 1 & 54.8 \\ 1 & 52.7 \\ \vdots & \vdots \end{pmatrix}}_{30\times2} \quad \boldsymbol{y} = \underbrace{\begin{pmatrix} 185 \\ 177.4 \\ 175.5 \\ \vdots \end{pmatrix}}_{30\times1}$$

$$\boldsymbol{X}^T\boldsymbol{X} = \begin{pmatrix} 30 & 1580.1 \\ 1580.1 & 83339.25 \end{pmatrix} \qquad (\boldsymbol{X}^T\boldsymbol{X})^{-1} = \begin{pmatrix} 24.076 & -0.456 \\ -0.456 & 0.009 \end{pmatrix}$$

$$\boldsymbol{X}^T\boldsymbol{y} = \begin{pmatrix} 5228.7 \\ 275645.13 \end{pmatrix} \qquad \boldsymbol{\theta}^* = (\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{y} = \begin{pmatrix} 60.4 \\ 2.16 \end{pmatrix}$$

Closed Form Solution

Example



Optimal in least-squares sense: $\theta^* = \begin{pmatrix} 60.4 \\ 2.16 \end{pmatrix}$ $\mathcal{L}(\boldsymbol{\theta}^*) = 7.95$

Duality to Maximum Likelihood

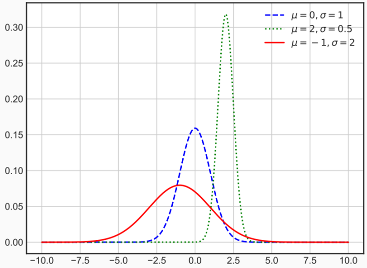
- least-squares has an interesting connection to **Gaussian density estimation** and the **maximum likelihood principle**
- we will first review Gaussian density estimation, an **unsupervised** technique, and connect it then to linear regression, a **supervised** technique

Univariate Gaussian Distribution

Consider the univariate **Gaussian distribution** with density

$$p(x; \mu, \sigma) = \frac{1}{\sqrt{2\pi}\,\sigma} e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2}$$

with parameters **mean** μ and **standard deviation** $\sigma > 0$, which might be written as parameter vector $\boldsymbol{\theta} = (\mu, \sigma)^T$.

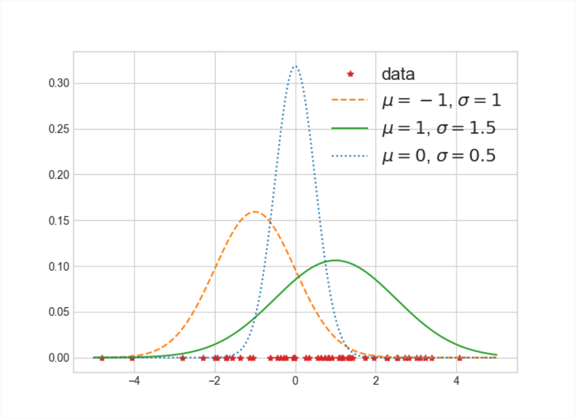


Carl Friedrich Gauss 1777–1855

Image: wikipedia

How to fit a Gaussian?

Say we have observed data $\mathcal{D} = \{x^{(1)}, \ldots, x^{(N)}\}$? Which $\boldsymbol{\theta} = (\mu, \sigma)^T$ explain this data best? What is a good objective function for density estimation?



Note that there are only inputs now – density estimation is **unsupervised**.

Maximum Likelihood Estimation

When $\mathcal{D} = \left\{x^{(1)}, x^{(2)}, \ldots, x^{(N)}\right\}$ are drawn **independently and identically distributed (i.i.d.)**, the probability density of the whole dataset $\mathcal{D}$ is

$$p(\mathcal{D}; \boldsymbol{\theta}) = \prod_{i=1}^{N} p(x^{(i)}; \boldsymbol{\theta})$$

Note: Generally, multiple random variables $X_1, X_2, \ldots, X_N$ are **statistically independent** if their distribution factorizes. Hence, the probability density $p(\mathcal{D}; \boldsymbol{\theta})$ factorizes into **sample-wise** densities $p(x^{(i)}; \boldsymbol{\theta})$ due to the i.i.d. assumption.

Maximum Likelihood Estimation cont'd

$p(\mathcal{D};\boldsymbol{\theta})$ depends on the dataset $\mathcal{D}$ and the parameters $\boldsymbol{\theta}$.

- When seen as function of data: probability density of $\mathcal{D}$
- When seen as function of parameters: **likelihood** of $\boldsymbol{\theta}$

Maximum Likelihood Estimator

Parameters $\boldsymbol{\theta}^*$ which maximize the likelihood

$$\boldsymbol{\theta}^* := \arg\max_{\boldsymbol{\theta}} \; p(\mathcal{D};\boldsymbol{\theta}) = \arg\max_{\boldsymbol{\theta}} \; \prod_{i=1}^{N} p(x^{(i)};\boldsymbol{\theta})$$

Generally, for any parametric model (not only Gaussians), maximum likelihood is a **consistent estimator,** i.e. for $N \to \infty$ it recovers the distribution closest to the true data distribution.

The Log-Likelihood

Instead of likelihood, we can maximize the **log-likelihood**:

$$\log p(\mathcal{D}; \boldsymbol{\theta}) := \log \prod_{i=1}^{N} p(x^{(i)}; \boldsymbol{\theta}) = \sum_{i=1}^{N} \log p(x^{(i)}; \boldsymbol{\theta})$$

Since the log is a **strictly increasing** function, the log-likelihood has exactly the same maxima as the likelihood, i.e. maximizing log-likelihood is equivalent to maximizing likelihood.

Optimizing the log-likelihood is usually easier and numerically more stable than optimizing likelihood (product of many small factors).

Maximum Likelihood for Gaussians

- Gaussian likelihood: $p(\mathcal{D}; \mu, \sigma) = \prod_{i=1}^{N} \frac{1}{\sqrt{2\pi}\,\sigma}\, e^{-\frac{1}{2}\left(\frac{x^{(i)}-\mu}{\sigma}\right)^2}$
- Thus, the log-likelihood is

$$\log p(\mathcal{D}; \mu, \sigma) = \sum_{i=1}^{N} - \log(\sqrt{2\pi}\,\sigma) - \frac{1}{2}\left(\frac{x^{(i)} - \mu}{\sigma}\right)^2$$

- **(Negative) quadratic function** in μ
- Deriving after μ delivers:

$$\frac{\partial \log p}{\partial \mu} = \sum_{i=1}^{N} -\frac{1}{2}\, 2 \left(\frac{x^{(i)} - \mu}{\sigma}\right)\left(-\frac{1}{\sigma}\right) = \sum_{i=1}^{N} \frac{x^{(i)} - \mu}{\sigma^2}$$

- At maximum, $\frac{\partial \log p(\mathcal{D};\mu,\sigma)}{\partial \mu} \overset{!}{=} 0$

Maximum Likelihood for Gaussians cont'd

$$\frac{\partial \log p(\mathcal{D}; \mu, \sigma)}{\partial \mu} = \sum_{i=1}^{N} \frac{x^{(i)} - \mu}{\sigma^2} \overset{!}{=} 0$$

Hence,

$$\sum_{i=1}^{N} x^{(i)} = \sum_{i=1}^{N} \mu^* = N\mu^* \quad \Rightarrow \quad \mu^* = \frac{1}{N} \sum_{i=1}^{N} x^{(i)}$$

- Maximum likelihood μ^* is simply the **average** over data
- Solution independent of σ
- Thus, we can now fix μ^* and maximize w.r.t. σ

Maximum Likelihood for Gaussians: σ^*

$$\log p(\mathcal{D}; \mu^*, \sigma) = \sum_{i=1}^{N} - \log(\sqrt{2\pi}\,\sigma) - \frac{1}{2}\left(\frac{x^{(i)} - \mu^*}{\sigma}\right)^2$$

Deriving after σ:

$$\frac{\partial \log p}{\partial \sigma} = \sum_{i=1}^{N} -\frac{\sqrt{2\pi}}{\sqrt{2\pi}\,\sigma} - \frac{1}{2}\,2\left(\frac{x^{(i)} - \mu^*}{\sigma}\right)(-1)\left(\frac{x^{(i)} - \mu^*}{\sigma^2}\right)$$

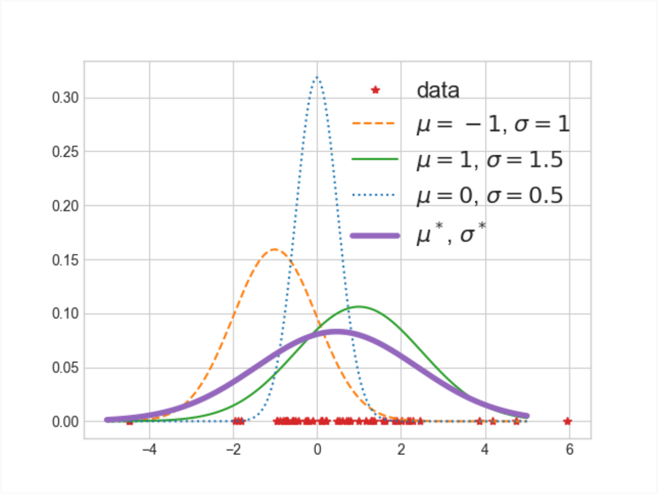
$$= \sum_{i=1}^{N} -\frac{1}{\sigma} + \left(\frac{(x^{(i)} - \mu^*)^2}{\sigma^3}\right) \overset{!}{=} 0$$

Hence, we get the **empirical standard deviation** as ML solution:

$$\sum_{i=1}^{N} \sigma^2 = \sum_{i=1}^{N} (x^{(i)} - \mu^*)^2 \quad \Rightarrow \quad \sigma^* = \sqrt{\frac{1}{N} \sum_{i=1}^{N} (x^{(i)} - \mu^*)^2}$$

Maximum Likelihood Fit

The maximum likelihood solution



Connection to Linear Regression

Connection to Linear Regression

- Assume now a **regression problem** with data

$$\mathcal{D} = \left\{(\boldsymbol{x}^{(1)}, y^{(1)}), (\boldsymbol{x}^{(2)}, y^{(2)}), \ldots, (\boldsymbol{x}^{(N)}, y^{(N)})\right\}$$

- We model now **the output** $y^{(i)}$ as univariate Gaussian with
    - **fixed** standard deviation σ,
    - but whose **mean depends on** $\boldsymbol{x}^{(i)}$:

$$\mu^{(i)} = \boldsymbol{w}^T \boldsymbol{x}^{(i)} + b$$

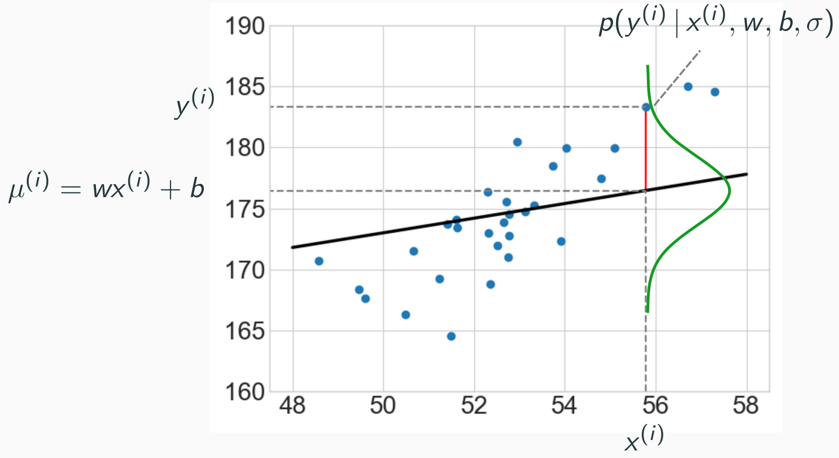
- Thus, $y^{(i)}$ is drawn from a **conditional Gaussian**:

$$p(y^{(i)} \mid \boldsymbol{x}^{(i)}, \boldsymbol{w}, b, \sigma) = \frac{1}{\sqrt{2\pi}\,\sigma}\, e^{-\frac{1}{2}\left(\frac{y^{(i)} - \boldsymbol{w}^T \boldsymbol{x}^{(i)} + b}{\sigma}\right)^2}$$

- Equivalently, the prediction error $e^{(i)} = y^{(i)} - \boldsymbol{w}^T \boldsymbol{x}^{(i)} + b$ is assumed to be Gaussian with $\mu = 0$, and fixed σ

Connection to Linear Regression

Example



Maximum Likelihood for Linear Regression

- Let $\boldsymbol{y} = (y^{(1)}, y^{(2)}, \ldots, y^{(N)})^T$ be the target vector
- $\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(N)}$ are fixed, including a "dummy feature" for the bias
- $\boldsymbol{\theta} = (b, w_1, \ldots, w_D)^T$ are model parameters to be learned
- Thus, we can simply write $\mu^{(i)} = \boldsymbol{\theta}^T \boldsymbol{x}^{(i)}$
- Standard deviation σ is fixed
- We want to maximize the log-likelihood:

$$\boldsymbol{\theta}^* = \arg\max_{\boldsymbol{\theta}} \log p(\boldsymbol{y} \mid \boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(N)}, \boldsymbol{\theta}, \sigma)$$

Maximum Likelihood for Linear Regression

$$
\begin{aligned}
&\log p(\boldsymbol{y} \mid \boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(D)}, \boldsymbol{\theta}, \sigma) \\
&\quad = \log \prod_{i=1}^{N} p(y^{(i)} \mid \boldsymbol{x}^{(i)}, \boldsymbol{\theta}, \sigma) \\
&\quad = \sum_{i=1}^{N} \log \left(\frac{1}{\sqrt{2\pi}\,\sigma} e^{-\frac{1}{2}\left(\frac{y^{(i)} - \boldsymbol{\theta}^T \boldsymbol{x}^{(i)}}{\sigma}\right)^2} \right) \\
&\quad = \sum_{i=1}^{N} \overbrace{\cancel{-\log \sqrt{2\pi}\,\sigma}}^{\text{const.}} - \overbrace{\frac{1}{2\sigma^2}}^{\textit{scaling}} \left(y^{(i)} - \boldsymbol{\theta}^T \boldsymbol{x}^{(i)} \right)^2
\end{aligned}
$$

Compare this with the least squares objective:

$$
\overbrace{\frac{1}{N}}^{\textit{scaling}} \sum_{i=1}^{N} \left(y^{(i)} - \boldsymbol{\theta}^T \boldsymbol{x}^{(i)} \right)^2
$$

Maximum Likelihood for Linear Regression

Clearly,

$$\max_{\boldsymbol{\theta}} \quad -\frac{1}{2\sigma^2}\sum_{i=1}^{N}\left(y^{(i)} - \boldsymbol{\theta}^T\boldsymbol{x}^{(i)}\right)^2$$

is equivalent to

$$\min_{\boldsymbol{\theta}} \quad \frac{1}{N}\sum_{i=1}^{N}\left(y^{(i)} - \boldsymbol{\theta}^T\boldsymbol{x}^{(i)}\right)^2$$

Thus, minimizing least squares is equivalent to maximum likelihood under Gaussian output assumption, with fixed σ!

Equivalence to Probabilistic Model

Why does this matter?

- Understanding methods in different ways
- Probabilistic analysis and justification
- Making the implicit model assumptions explicit
- Generalizing the model by playing with probabilistic assumptions
- Generally, many apparently non-probabilistic machine learning algorithms can be explained with probabilistic arguments