Supervised Learning

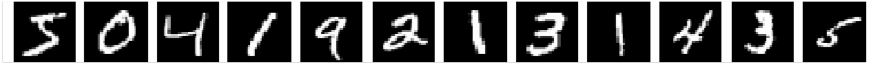
Johanni Brea

Einführung Machine Learning

GYMINF 2022

Table of Contents

Handwritten Digit Classification (MNIST)



our goal: assign the correct digit class to images

5 0 4 1 9 2 1 3 1 4 3 5

input X: 28x28 = 784 pixels with values between 0 (black) and 1 (white)

output Y: digit class 0, 1, . . ., 9

Spam Detection with the Enron Dataset

spam

Subject: follow up
here ’ s a question i ’ ve been wanting to ask you , are you feeling down but too embarrassed to go to the doc to get your m / ed ’ s ?
here ’ s the answer , forget about your local p harm . acy and the long waits , visits and embarassments . . do it all in the privacy of your own home , right now . http : / / chopin . manilamana . com / p / test / duet it ’ s simply the best and most private way to obtain the stuff you need without all the red tape .

ham

Subject: darrin presto
amy :
please follow up as soon as possible with darrin presto regarding a real time interview . i forwarded his resume to you last week . he can be reached at 509 - 946 - 7879
thanks
greg

Our goal: classify new emails as spam or “ham“ (not spam).

input X: sequences of characters (emails), output Y: label spam or ham

Wind Speed Prediction

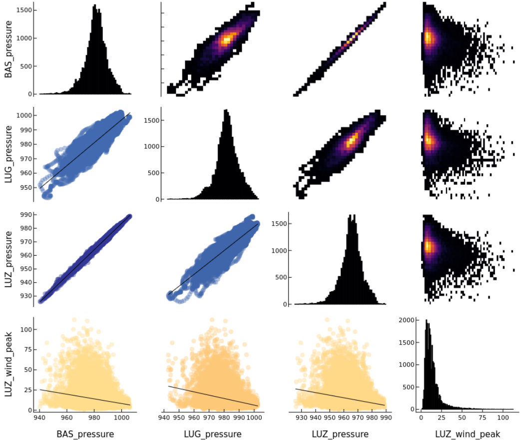
- SwissMeteo data: hourly measurements for 5 years from different stations (Bern, Basel, Luzern, Lugano, etc.).
- Our goal: given measurements at different stations, predict wind speed in Luzern 5 hours later.

Wind Speed Prediction

time	BAS_pressure	LUG_pressure	...	LUZ_pressure	LUZ_wind_peak
$x_{11} = 2015010100$	$x_{12} = 997.1$	$x_{13} = 998.6$	$\dots$	$x_{1p} = 980.0$	$y_1 = 13.0$
$x_{21} = 2015010101$	$x_{22} = 997.3$	$x_{23} = 998.8$	$\dots$	$x_{2p} = 979.9$	$y_2 = 6.8$
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$	
$x_{n1} = 2017123123$	$x_{n2} = 972.7$	$x_{n3} = 981.5$	$\dots$	$x_{np} = 957.5$	$y_n = 11.9$

- p **input** variables $X = (X_1, X_2, \dots, X_p)$
  e.g. X_1 `time`, X_2 `BAS_pressure`, X_3 `LUG_pressure`
  also called: **predictors, independent variables, features**
- **output** variable Y e.g. `LUZ_wind_peak`
  also called: **response, dependent variable**
- n **measurements** or **data points**

Always Look at Raw Data!

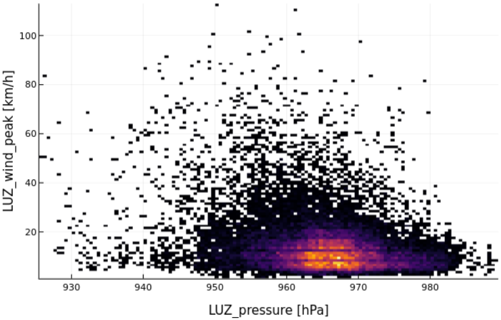


- **on diagonal**: 1D histogram
- **lower triangle**: scatter plot & trend line
- **upper triangle**: 2D histogram

Observations

1. LUZ_wind_peak has a long tail.
2. For low pressures there are outliers of strong wind.
3. Pressure in Basel and Luzern is highly correlated.
4. ...

Wind Speed Prediction



- The higher the pressure in Luzern, the less probable it is to have strong winds.
- There is no function `LUZ_wind_peak` = f(`LUZ_pressure`) that can describe this data; instead we use conditional probability densities p(`LUZ_wind_peak` | `LUZ_pressure`).

Table of Contents

Data Generating Processes

It is useful to think of our datasets as samples from **data generating processes** for the input X and the conditional output Y|X.

- **MNIST** X: people write digits $\to$ people take standardized photos thereof.
  Y|X: different people label the same photo X.
- **Spam** X: people write emails.
  Y|X: different people classify the same email X as spam or not.
- **Weather** X: the weather acts on sensors in weather stations.
  Y|X: the weather evolves from X and is measured again 5 hours later.

Using samples from these data generating processes, supervised learning aims at learning something about the conditional processes, i.e how Y depends on X.

Where Does Noise Come From?

For most data generating processes we **cannot measure all factors** that determine the outcome.

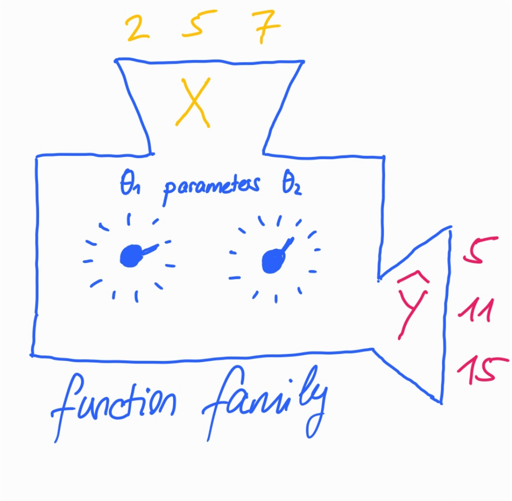
$\Rightarrow$ **same values of the measured factors can cause different outcomes**.

- **MNIST** Different persons may label the same handwritten digit differently.
- **Spam** What is spam for somebody, may not be spam for someone else.
- **Weather** Even when all considered weather stations measure exactly the same values at time t_1 and t_2, the full state of the weather at t_1 differs most likely from the one at t_2.

In machine learning we treat the effect of unmeasured factors as noise with certain probability distributions.

Table of Contents

How Does Supervised Learning Work?



Function Family

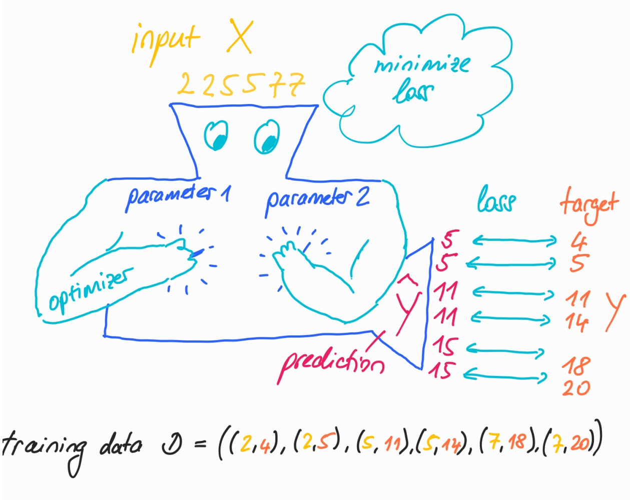
- We change the parameters.
- The machine computes $\hat{y}$ given parameters θ and x.

For example

$$\hat{y} = f_\theta(x) = \theta_0 + \theta_1 x$$

When we change the parameters θ_0 and θ_1, we change the way $\hat{y}$ depends on x.

How Does Supervised Learning Work?



Loss Minimizing Machine

- We specify
  1. the training data
  2. the function family (model)
  3. the loss function $L(y, \hat{y})$
  4. the optimizer
- The machine changes the parameters with the help of the optimizer until the loss is minimal.

For example: linear regression

Training Loss and Test Loss

- **Training Set** $\mathcal{D}$: Data used by the machine to tune the parameters.
- **Training Loss of Function** f: $\mathcal{L}(f, \mathcal{D}) = \frac{1}{n}\sum_{i=1}^{n} L(y_i, f(x_i))$
- **Test Loss of Function** f **at** x **for a Conditional Data Generating Process**: $\mathrm{E}_{Y|x}[L(Y, f(x))]$ = expected loss under the conditional generating process.
- **Test Loss of Function** f **for a Joint Data Generating Process**: $\mathrm{E}_{X,Y}[L(Y, f(X))]$ = expected loss under the joint generating process.
- **Test Set** $\mathcal{D}_{\text{test}}$: Data from the same generating process as the training set, not used for parameter tuning.
- **Test Loss of Function** f **for a Test Set** $\mathcal{D}_{\text{test}}$: $\mathcal{L}(f, \mathcal{D}_{\text{test}})$ = same computation as for the training loss but for a test set.

Blackboard: Linear Regression as a Loss Minimizing Machine

Data Generating Process

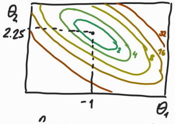
$y = 2x - 1 + \varepsilon$, $\quad E[\varepsilon] = 0 \quad Var[\varepsilon] = \sigma^2$

Training Data

$((x_1 = 0, y_1 = -1), (x_2 = 2, y_2 = 4), (x_3 = 2, y_3 = 3))$

Function Family

$\hat{y} = \theta_0 + \theta_1 \cdot x$



Loss Function

$$L(\theta) = L(\theta_0, \theta_1) = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 = \frac{1}{n}\sum_{i=1}^{n}(y_i - \theta_0 - \theta_1 x_i)^2$$

$$= \frac{1}{3}\left((-1 - \theta_0)^2 + (4 - \theta_0 - 2\theta_1)^2 + (3 - \theta_0 - 2\theta_1)^2\right)$$

Optimizer: Default

Solution: $\hat{\theta}_0 = -1$, $\hat{\theta}_1 = 2.25$, $L(\hat{\theta}) = \frac{2}{3} \cdot 0.5^2$

Test Loss at x_0:

$$E[(\underbrace{2x_0 - \cancel{1} + \varepsilon}_{y} \underbrace{+ \cancel{1} - 2.25x_0}_{-\hat{y}})^2] = (0.25x_0)^2 + \sigma^2$$

Test Data:

$((x_1 = 1, y_1 = 0), (x_2 = 2, y_2 = 3), (x_3 = 3, y_3 = 5), (x_4 = 0, y_4 = -1))$

$\Rightarrow$ (Empirical) Test Loss $= \frac{1}{4}(0.25^2 + 0.5^2 + 0.75^2 + 0^2)$

Quiz

Suppose we have training data $\mathcal{D} = ((0,1),(2,9))$ and test data $\mathcal{D}_{\text{test}} = ((0,0),(3,20))$, define a function family $f(x) = \theta_0 + \theta_1 x^2$ and loss function $L(y,\hat{y}) = |y - \hat{y}|$.

Correct or wrong?

1. The training loss is minimal for $\hat{\theta}_0 = 1$ and $\hat{\theta}_1 = 2$.
2. The test loss of $f(x) = 1 + 2x^2$ at $x = 0$ for the conditional data generating process is 1.
3. The test loss of $f(x) = 1 + 2x^2$ for the test set is 1.

Supervised Learning with MLJ

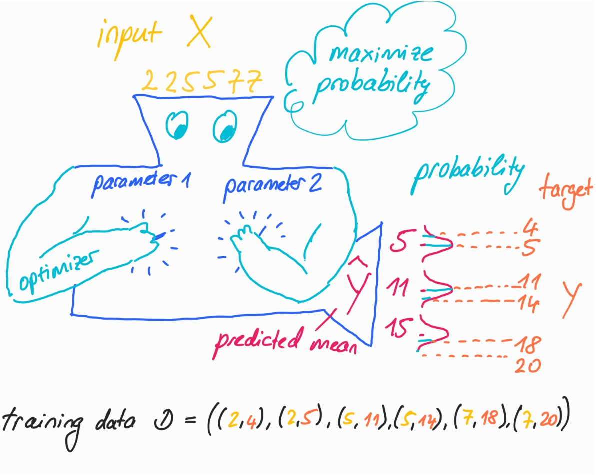
```
model = LinearRegressor()     # function family, loss function and optimizer
mach = machine(model, X, y)  # training data with input X and output y
fit!(mach)                    # fit the machine
fitted_params(mach)           # inspect the fitted parameters
ŷ = predict(mach)             # make predictions on the training data
ŷtest = predict(mach, Xtest) # make predictions on the test data
```

Which Loss Functions Should We Use?

- Is the mean squared error always a good loss?
- What kind of loss would be good in a classification setting (e.g. MNIST)?
- How should we choose the loss when we know something about the noise distribution?

All these questions have a straight-forward answer, if we use a **family of probability distributions** (instead of a family of functions) and estimate the parameters with a **maximum likelihood approach** (instead of minimizing a hand-picked loss).

How Does Supervised Learning Work?



Likelihood Maximizing Machine

- We specify
  1. the training data
  2. the family of probability distributions (model)
  3. the optimizer
- The machine changes the parameters with the help of the optimizer until the likelihood of the parameters is maximal.

For example: linear regression

The Likelihood Function

For a family of conditional probability distributions $P(y|x,\theta)$ and training data $\mathcal{D} = ((x_1, y_1), (x_2, y_2), \dots, (x_n, y_n))$ the **likelihood function** is defined as

$$\ell(\theta) = \prod_{i=1}^{n} P(y_i|x_i, \theta)\,.$$

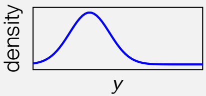
This is the probability of all the responses y_i given all the inputs x_i for a given value of the parameters θ.

In practice it is usually more convenient to work with the **log-likelihood function**

$$\log \ell(\theta) = \sum_{i=1}^{n} \log P(y_i|x_i, \theta)$$

The Normal, Bernoulli and Categorical Distribution

Normal



$$p(y|x) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(y-f(x))^2}{2\sigma^2}}$$

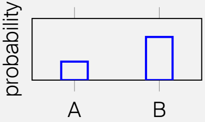
$f(x)$: a number

mean: $f(x)$

variance: σ^2

mode: $f(x)$

Bernoulli



$$p(\mathrm{A}|x) = p_\mathrm{A} = \sigma(f(x))$$

$f(x)$: a number

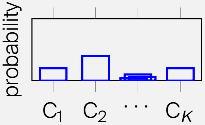
sigmoid/logistic function

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

$$p(\mathrm{B}|x) = 1 - p_\mathrm{A} = \sigma(-f(x))$$

mode: A if $p_\mathrm{A} > p_\mathrm{B}$

Categorical



$$p(\mathrm{C}_i|x) = p_{\mathrm{C}_i} = s(f(x))_i$$

$f(x)$: a vector of K numbers

softmax function

$$s(x)_i = \frac{e^{x_i}}{\sum_{j=1}^K e^{x_j}}$$

mode: X with largest p_X.

Blackboard: The Normal, Bernoulli and Categorical Distribution

Normal

$f(x) = 1\ ,\ \sigma = 2$

$$Pr(y \in [-0.05, 0.05]) \approx 0.1 \cdot \frac{1}{\sqrt{2\pi} \cdot 2} e^{-\frac{(0-1)^2}{2 \cdot 2^2}} \approx 0.017$$

Bernoulli

$f(x) = 3$

$$Pr(y = A) = \frac{1}{1+e^{-3}} \approx 0.95$$

$$Pr(y = B) \approx \frac{1}{1+e^{3}} \approx 0.05$$

Categorical

$$f(x) = \begin{pmatrix} 3 \\ -2 \\ 1 \\ 0 \end{pmatrix}$$

$$Pr(y = A) = \frac{e^3}{e^3 + e^{-2} + e^1 + e^0} \approx 0.84$$

$$Pr(y = D) = \frac{e^0}{e^3 + e^{-2} + e^1 + e^0} \approx 0.04$$

Blackboard: Maximum Likelihood Estimation

Data Generating Process

$P(y=A|x) = \text{Bernoulli}(2x-1)$

$y = A$ if $\sigma(2x-1) > \varepsilon$, $\varepsilon \sim \text{Uniform}([0,1])$

Training Data

$((x_1 = 0, y_1 = B), (x_2 = 2, y_2 = A), (x_3 = 3, y_3 = B))$

Family of Distributions

$P(y=A|x,\theta) = \sigma(\theta_0 + \theta_1 x)$

Log-Likelihood Function

$\log \ell(\theta) = \log \ell(\theta_0, \theta_1) = \sum_{i=1}^{n} \log P(y_i | x_i, \theta)$

$= \log \sigma(-\theta_0) + \log \sigma(\theta_0 + 2\theta_1) + \log \sigma(-\theta_0 - 3\theta_1)$

Optimizer: Default

Solution: $\hat{\theta}_0 \approx -1.3$ $\hat{\theta}_1 \approx 0.3$ $\log \ell(\hat{\theta}) \approx -1.9$

Test Log-Likelihood at x_0: $E[\log P(Y|x_0)]$

$\underbrace{\sigma(2x_0-1)}_{P(Y=A|x_0)} \cdot \log \underbrace{\sigma(0.3x_0 - 1.3)}_{P(Y=A|x_0,\hat{\theta})} + \underbrace{\sigma(-2x_0+1)}_{P(Y=B|x_0)} \log \underbrace{\sigma(-0.3x_0+1.3)}_{P(Y=B|x_0,\hat{\theta})}$

Supervised Learning with MLJ

```julia
model = LogisticClassifier() # distribution family and optimizer
mach = machine(model, X, y)  # training data with input X and output y
fit!(mach)                   # fit the machine
fitted_params(mach)          # inspect the fitted parameters
p̂ = predict(mach)            # predicted probabilities on the training data
p̂ₐ = pdf.(p̂, "A")            # predicted probabilities of class "A"
ŷ = predict_mode(mach)       # class with highest predicted probability
```

Summary

We use a training set to find a conditional distribution that captures some regularities of the conditional data generation process. The goal is to find a conditional distribution that minimizes the test loss of the joint data generation process. With a test set we can assess how close we are at reaching this goal.

Supervised Learning as Loss Minimization	Supervised Learning as Likelihood Maximization
We provide 1. training data 2. function family 3. loss function 4. optimizer	We provide 1. training data 2. probability distribution family 3. optimizer The negative log-likelihood function of the parameters implicitly defines a loss function.
It is not (always) obvious what kind of loss function to take for classification problems or regression problems with a specific noise distributions	We take the binomial for binary classification problems and the categorical for other classification problems. In regression with a specific noise distribution this can easily be used.