HEINZ NIXDORF INSTITUT
UNIVERSITÄT PADERBORN

MASCHINELLES LERNEN FÜR INTELLIGENTE SYSTEME: ÜBER DATEN, WISSEN UND ALGORITHMEN

Eyke Hüllermeier

Fachgebiet Intelligente Systeme und Maschinelles Lernen

Fachgruppe Industrie 4.0 meets Fachgruppe Systems Engineering, Miele, Gütersloh, 27. November 2018

MEILENSTEINE DER KI

AlphaGo schlägt Lee Sedol (2016)

Watson gewinnt Jeopardy! (2011)

Deep Blue schlägt Garry Kasparov (1997)

MEILENSTEINE DER KI

Daten
+ Lernen

AlphaGo schlägt Lee Sedol (2016)

Wissen
+ Retrieval

Watson gewinnt Jeopardy! (2011)

Algorithmik
+ Rechenleistung

Deep Blue schlägt Garry Kasparov (1997)

MEILENSTEINE DER KI

Von der Welt der Spiele in die Realität ...

Daten
+ Lernen

AlphaGo schlägt Lee Sedol (2016)

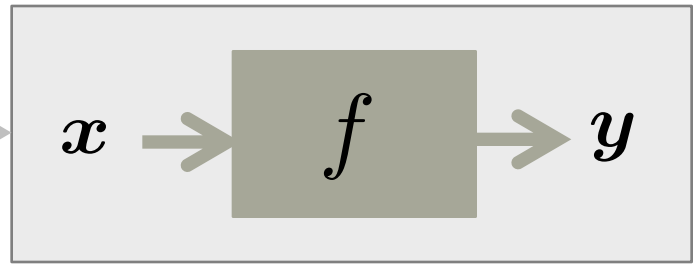
Wissen
+ Retrieval

Watson gewinnt Jeopardy! (2011)

Algorithmik
+ Rechenleistung

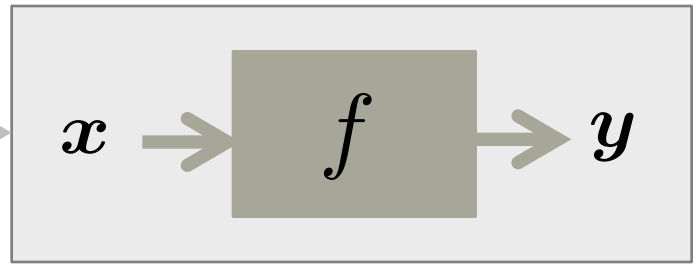
Deep Blue schlägt Garry Kasparov (1997)

DER ALGORITHMISCHE ANSATZ

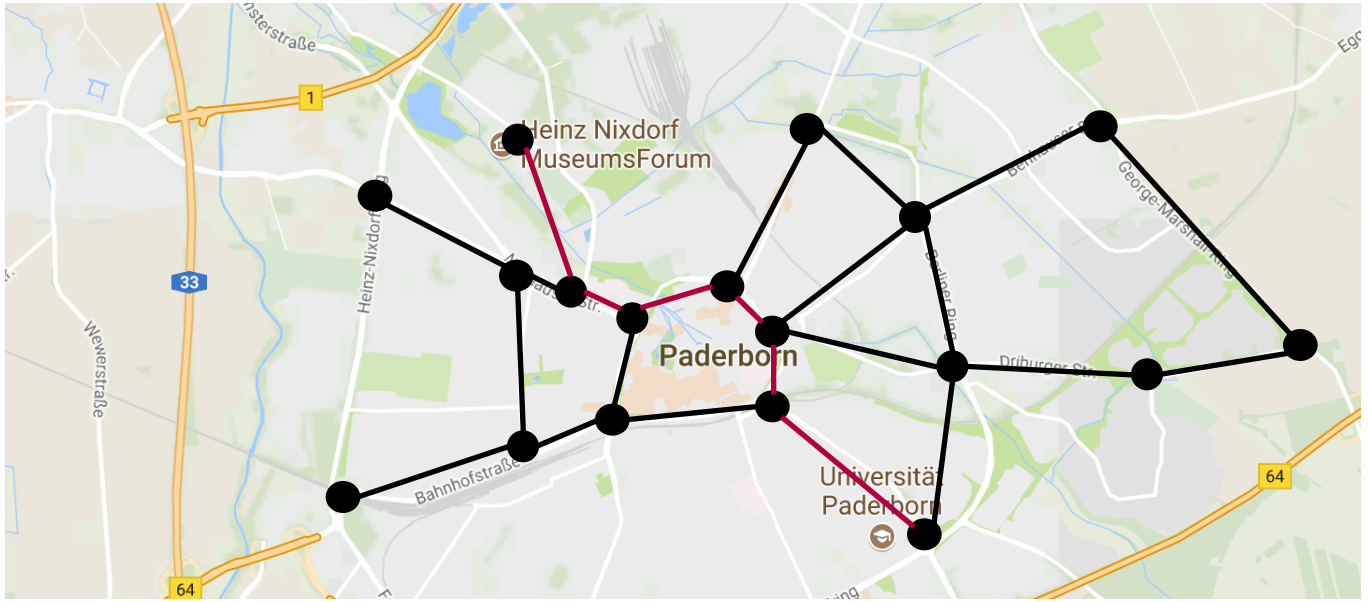


Algorithmus

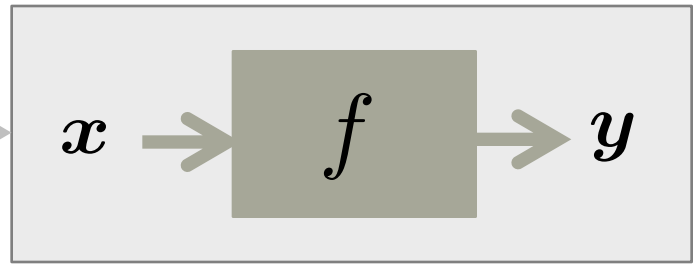
DER ALGORITHMISCHE ANSATZ



Algorithmus



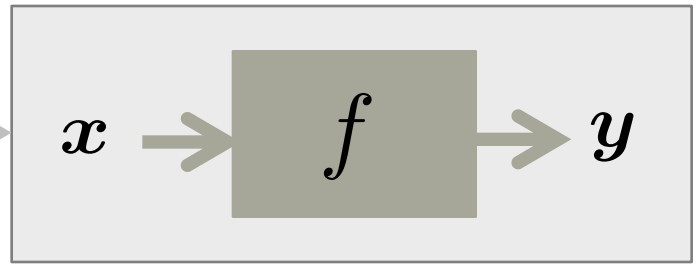
DER ALGORITHMISCHE ANSATZ



Algorithmus

```
ALGORITHM shortest-path(V,T)
 W := {v1}
 ShortDist[v1] :=0
 FOR each u in V - {v1}
     ShortDist[u] := T[v1,u]
 WHILE W /= V
       MinDist := INFINITE
       FOR each v in V - W
            IF ShortDist[v] < MinDist
               MinDist = ShortDist[v]
               w := v
            END {if}
       END {for}
       W := W U {w}
       FOR each u in V - W
            ShortDist[u] := Min(ShorDis[u],ShortDist[w] + T[w,u])
 END {while}
```

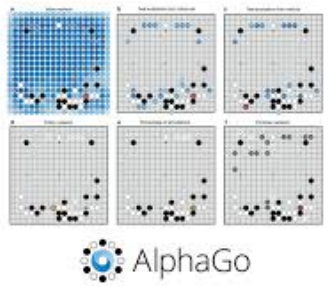
DER ALGORITHMISCHE ANSATZ



Algorithmus

*Erfordert nicht nur ein **umfassendes Verständnis** des Problems, sondern auch des **Lösungsprozesses**.*

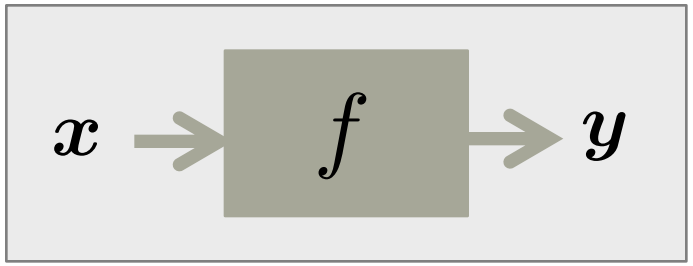
KOMPLEXE PROBLEME



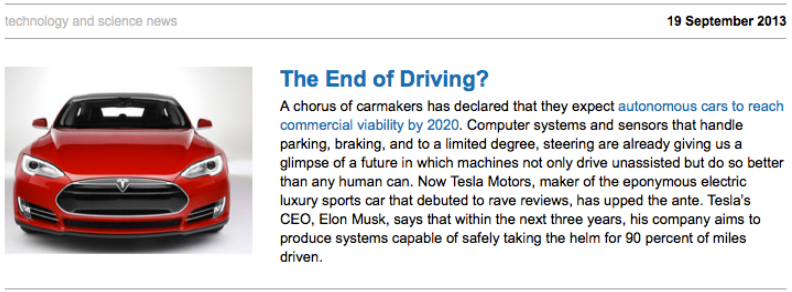
SPIELE, GAMING

ROBOTER-FUSSBALL

Zustandsvektor
beschreibt die Umgebung



Aktion


technology and science news **19 September 2013**

The End of Driving?

A chorus of carmakers has declared that they expect autonomous cars to reach commercial viability by 2020. Computer systems and sensors that handle parking, braking, and to a limited degree, steering are already giving us a glimpse of a future in which machines not only drive unassisted but do so better than any human can. Now Tesla Motors, maker of the eponymous electric luxury sports car that debuted to rave reviews, has upped the ante. Tesla's CEO, Elon Musk, says that within the next three years, his company aims to produce systems capable of safely taking the helm for 90 percent of miles driven.

AUTONOMES FAHREN

DIE EVOLUTION INTELLIGENTER SYSTEME

```
function GetMin(var a: TList)
var
    i, min, mini: integer;
begin
    min := MaxInt;
    mini := 0;
    for i := 1 to a.len do
        if a.arr[i].G < min t
        begin
            min := a.arr[i].G
            mini := i;
        end;

    GetMin := mini;
end;
```

klassische Programmierung

```
mann(adam).
mann(tobias).
mann(frank).
frau(eva).
frau(daniela).
frau(ulrike).
vater(adam,tobias).
vater(tobias,frank).
vater(tobias,ulrike).
mutter(eva,tobias).
mutter(daniela,frank).
mutter(daniela,ulrike).
```

Wissensbasierte Systeme

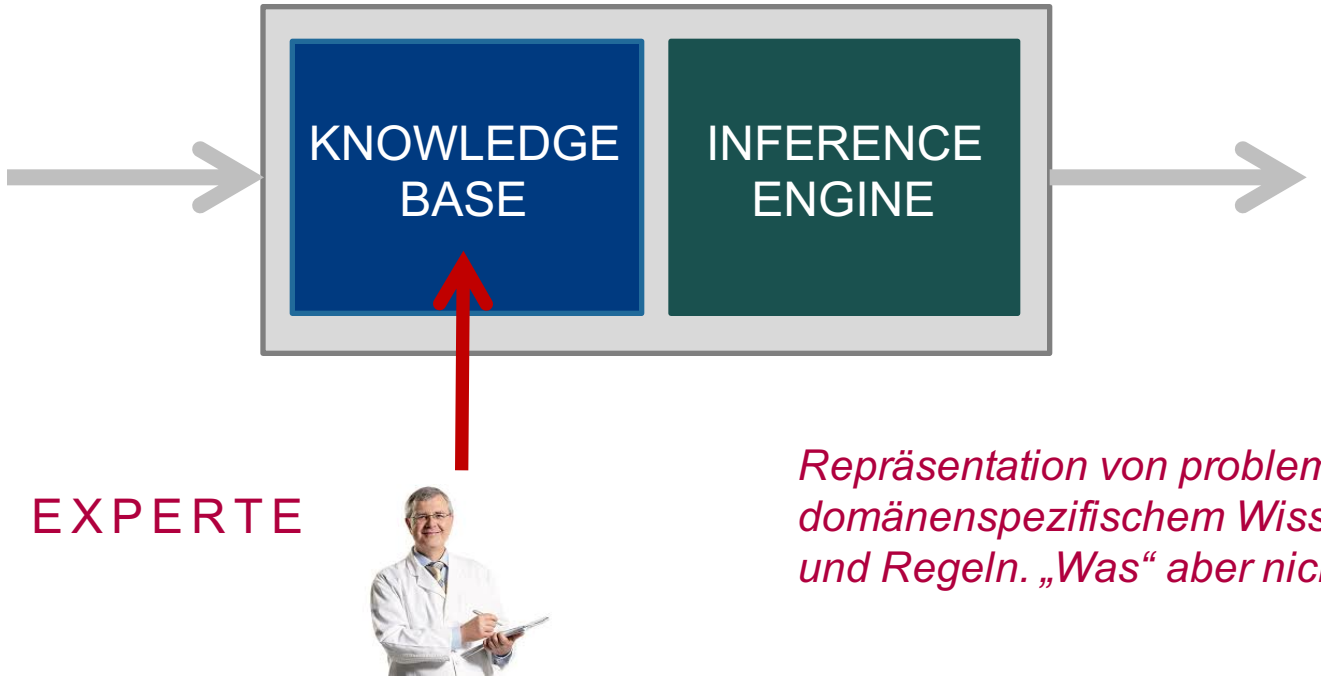
```
# Spot Check Algorithms
models = []
models.append(('LR', Logis
models.append(('LDA', Line
models.append(('KNN', KNei
models.append(('CART', Dec
models.append(('NB', Gauss
models.append(('SVM', SVC(
# evaluate each model in t
results = []
names = []
for name, model in models:
    kfold = model_selectio
    cv_results = model_sel
    results.append(cv_resu
    names.append(name)
    msg = "%s: %f (%f)" %
    print(msg)
```

“implizite” Programmierung

automatisiertes Machine Learning

… ist schwierig für komplexe Probleme

WISSENSBASIERTE SYSTEME



Repräsentation von problem- und domänenspezifischem Wissen wie Fakten und Regeln. „Was“ aber nicht „wie“!

- Generische Kontrollstrukturen implementiert durch eine Infernzmaschine
- Programme = Theorien in formaler Logik, Berechnung = Deduktion
- Eng verwandt zu deklarativen Programmiersprachen wie PROLOG

DIE EVOLUTION INTELLIGENTER SYSTEME

```
function GetMin(var a: TList)
var
    i, min, mini: integer;
begin
    min := MaxInt;
    mini := 0;
    for i := 1 to a.len do
        if a.arr[i].G < min t
        begin
            min := a.arr[i].G
            mini := i;
        end;

    GetMin := mini;
end;
```

klassische Programmierung

… ist schwierig für komplexe Probleme

```
mann(adam).
mann(tobias).
mann(frank).
frau(eva).
frau(daniela).
frau(ulrike).
vater(adam,tobias).
vater(tobias,frank).
vater(tobias,ulrike).
mutter(eva,tobias).
mutter(daniela,frank).
mutter(daniela,ulrike).
```

Wissensbasierte Systeme

… leidet unter dem Flaschenhals des Wissenserwerbs

```
# Spot Check Algorithms
models = []
models.append(('LR', Logis
models.append(('LDA', Line
models.append(('KNN', KNei
models.append(('CART', Dec
models.append(('NB', Gauss
models.append(('SVM', SVC(
# evaluate each model in t
results = []
names = []
for name, model in models:
    kfold = model_selectio
    cv_results = model_sel
    results.append(cv_resu
    names.append(name)
    msg = "%s: %f (%f)" %
    print(msg)
```

“implizite” Programmierung

automatisiertes Machine Learning

AUTOMATISIERUNG MENSCHLICHER FÄHIGKEITEN

Menschliche Fähigkeiten sind nicht immer einfach zu beschreiben!

MANN
oder
FRAU

$\boldsymbol{x} \in \mathbb{R}^N$

$\boldsymbol{y} \in \{0, 1\}$

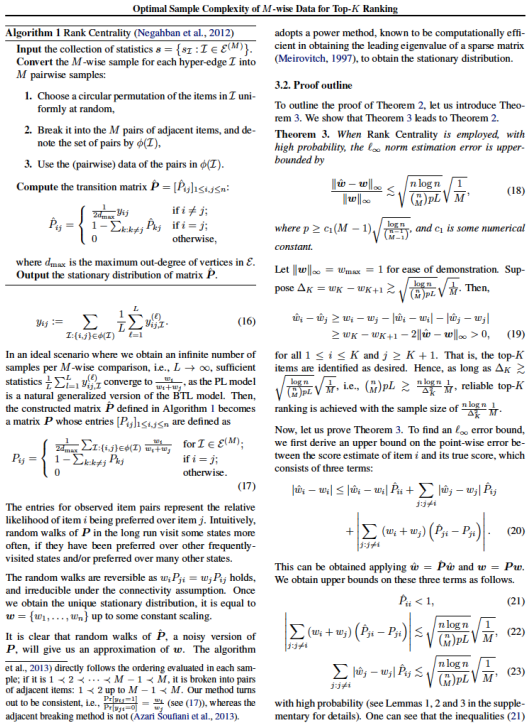
AUTOMATISIERUNG MENSCHLICHER FÄHIGKEITEN

Menschliche Fähigkeiten sind nicht immer einfach zu beschreiben!

For example, a reduction of the search space does not immediately imply better solutions.

Eine Beschränkung des Suchraums führt beispielsweise nicht unmittelbar zu besseren Lösungen.

AUTOMATISIERUNG MENSCHLICHER FÄHIGKEITEN



Optimal Sample Complexity of M-wise Data for Top-K Ranking

Algorithm 1 Rank Centrality (Negahban et al., 2012)

Input the collection of statistics $\boldsymbol{s} = \{s_{\mathcal{I}} : \mathcal{I} \in \mathcal{E}^{(M)}\}$.
Convert the M-wise sample for each hyper-edge $\mathcal{I}$ into M pairwise samples:

1. Choose a circular permutation of the items in $\mathcal{I}$ uniformly at random,
2. Break it into the M pairs of adjacent items, and denote the set of pairs by $\phi(\mathcal{I})$,
3. Use the (pairwise) data of the pairs in $\phi(\mathcal{I})$.

Compute the transition matrix $\hat{\boldsymbol{P}} = [\hat{P}_{ij}]_{1\le i,j\le n}$:

$$\hat{P}_{ij} = \begin{cases} \frac{1}{2d_{\max}} y_{ij} & \text{if } i \neq j; \\ 1 - \sum_{k:k\neq j} \hat{P}_{kj} & \text{if } i = j; \\ 0 & \text{otherwise,} \end{cases}$$

where $d_{\max}$ is the maximum out-degree of vertices in $\mathcal{E}$.
Output the stationary distribution of matrix $\hat{\boldsymbol{P}}$.

$$y_{ij} := \sum_{\mathcal{I}:(i,j)\in\phi(\mathcal{I})} \frac{1}{L} \sum_{\ell=1}^{L} y_{ij,\mathcal{I}}^{(\ell)}. \tag{16}$$

In an ideal scenario where we obtain an infinite number of samples per M-wise comparison, i.e., $L \to \infty$, sufficient statistics $\frac{1}{L}\sum_{\ell=1}^{L} y_{ij,\mathcal{I}}^{(\ell)}$ converge to $\frac{w_i}{w_i+w_j}$, as the PL model is a natural generalized version of the BTL model. Then, the constructed matrix $\hat{\boldsymbol{P}}$ defined in Algorithm 1 becomes a matrix $\boldsymbol{P}$ whose entries $[P_{ij}]_{1\le i,j\le n}$ are defined as

$$P_{ij} = \begin{cases} \frac{1}{2d_{\max}} \sum_{\mathcal{I}:(i,j)\in\phi(\mathcal{I})} \frac{w_i}{w_i+w_j} & \text{for } \mathcal{I} \in \mathcal{E}^{(M)}; \\ 1 - \sum_{k:k\neq j} P_{kj} & \text{if } i = j; \\ 0 & \text{otherwise.} \end{cases} \tag{17}$$

The entries for observed item pairs represent the relative likelihood of item i being preferred over item j. Intuitively, random walks of $\boldsymbol{P}$ in the long run visit some states more often, if they have been preferred over other frequently-visited states and/or preferred over many other states.

The random walks are reversible as $w_i P_{ji} = w_j P_{ij}$ holds, and irreducible under the connectivity assumption. Once we obtain the unique stationary distribution, it is equal to $\boldsymbol{w} = \{w_1, \dots, w_n\}$ up to some constant scaling.

It is clear that random walks of $\hat{\boldsymbol{P}}$, a noisy version of $\boldsymbol{P}$, will give us an approximation of $\boldsymbol{w}$. The algorithm adopts a power method, known to be computationally efficient in obtaining the leading eigenvalue of a sparse matrix (Meirovitch, 1997), to obtain the stationary distribution.

et al., 2013) directly follows the ordering evaluated in each sample; if it is $1 \prec 2 \prec \cdots \prec M-1 \prec M$, it is broken into pairs of adjacent items: $1 \prec 2$ up to $M-1 \prec M$. Our method turns out to be consistent, i.e., $\frac{\mathbb{P}[y_{ij}=1]}{\mathbb{P}[y_{ij}=0]} = \frac{w_i}{w_j}$ (see (17)), whereas the adjacent breaking method is not (Azari Soufiani et al., 2013).

3.2. Proof outline

To outline the proof of Theorem 2, let us introduce Theorem 3. We show that Theorem 3 leads to Theorem 2.

Theorem 3. *When* Rank Centrality *is employed, with high probability, the ℓ_∞ norm estimation error is upper-bounded by*

$$\frac{\|\hat{\boldsymbol{w}} - \boldsymbol{w}\|_\infty}{\|\boldsymbol{w}\|_\infty} \lesssim \sqrt{\frac{n \log n}{\binom{n}{M} pL}} \sqrt{\frac{1}{M}}, \tag{18}$$

where $p \ge c_1 (M-1) \sqrt{\frac{\log n}{\binom{n}{M-1}}}$, and c_1 is some numerical constant.

Let $\|\boldsymbol{w}\|_\infty = w_{\max} = 1$ for ease of demonstration. Suppose $\Delta_K = w_K - w_{K+1} \gtrsim \sqrt{\frac{\log n}{\binom{n}{M} pL}} \sqrt{\frac{1}{M}}$. Then,

$$\begin{aligned} \hat{w}_i - \hat{w}_j &\ge w_i - w_j - |\hat{w}_i - w_i| - |\hat{w}_j - w_j| \\ &\ge w_K - w_{K+1} - 2\|\hat{\boldsymbol{w}} - \boldsymbol{w}\|_\infty > 0, \end{aligned} \tag{19}$$

for all $1 \le i \le K$ and $j \ge K+1$. That is, the top-K items are identified as desired. Hence, as long as $\Delta_K \gtrsim \sqrt{\frac{\log n}{\binom{n}{M} pL}} \sqrt{\frac{1}{M}}$, i.e., $\binom{n}{M} pL \gtrsim \frac{n \log n}{\Delta_K^2} \frac{1}{M}$, reliable top-$K$ ranking is achieved with the sample size of $\frac{n \log n}{\Delta_K^2} \frac{1}{M}$.

Now, let us prove Theorem 3. To find an ℓ_∞ error bound, we first derive an upper bound on the point-wise error between the score estimate of item i and its true score, which consists of three terms:

$$\begin{aligned} |\hat{w}_i - w_i| \le{}& |\hat{w}_i - w_i| \hat{P}_{ii} + \sum_{j:j\neq i} |\hat{w}_j - w_j| \hat{P}_{ij} \\ &+ \left| \sum_{j:j\neq i} (w_i + w_j) \left(\hat{P}_{ji} - P_{ji} \right) \right|. \end{aligned} \tag{20}$$

This can be obtained applying $\hat{\boldsymbol{w}} = \hat{\boldsymbol{P}}\hat{\boldsymbol{w}}$ and $\boldsymbol{w} = \boldsymbol{P}\boldsymbol{w}$. We obtain upper bounds on these three terms as follows.

$$\hat{P}_{ii} < 1, \tag{21}$$

$$\left| \sum_{j:j\neq i} (w_i + w_j) \left(\hat{P}_{ji} - P_{ji} \right) \right| \lesssim \sqrt{\frac{n \log n}{\binom{n}{M} pL}} \sqrt{\frac{1}{M}}, \tag{22}$$

$$\sum_{j:j\neq i} |\hat{w}_j - w_j| \hat{P}_{ij} \lesssim \sqrt{\frac{n \log n}{\binom{n}{M} pL}} \sqrt{\frac{1}{M}}, \tag{23}$$

with high probability (see Lemmas 1, 2 and 3 in the supplementary for details). One can see that the inequalities (21)

Abstract

Given a sample of instances with binary labels, the top ranking problem is to produce a ranked list of instances where the *head* of the list is dominated by positives. Popular existing approaches to this problem are based on surrogates to a performance measure known as the fraction of positives of the top (PTop). In this paper, we show that the measure and its surrogates have an undesirable property: for certain noisy distributions, it is optimal to trivially predict *the same score for all instances*. We propose a simple rectification of the measure which avoids such trivial solutions, while still focussing on the head of the ranked list and being as easy to optimise.

AUS DATEN LERNEN

Statt eine Lösung direkt (algorithmisch oder wissensbasiert) zu entwickeln, ist es einfacher ...

- **Beispiele** zu zeigen und diese vom System **generalisieren** zu lassen

→ MANN

AUS DATEN LERNEN

Statt eine Lösung direkt (algorithmisch oder wissensbasiert) zu entwickeln, ist es einfacher ...

- **Beispiele** zu zeigen und diese vom System **generalisieren** zu lassen

→ MANN

- das System **probieren** zu lassen und **Feedback** zu geben

AUS DATEN LERNEN

Statt eine Lösung direkt (algorithmisch oder wissensbasiert) zu entwickeln, ist es einfacher ...

- **Beispiele** zu zeigen und diese vom System **generalisieren** zu lassen

→ MANN

- das System **probieren** zu lassen und **Feedback** zu geben

- etwas **vorzuführen** und das System **imitieren** zu lassen

AUS DATEN LERNEN

Statt eine Lösung direkt (algorithmisch oder wissensbasiert) zu entwickeln, ist es einfacher ...

- **Beispiele** zu zeigen und diese vom System **generalisieren** zu lassen

→ MANN

→ *supervised learning*

- das System **probieren** zu lassen und **Feedback** zu geben

→ *reinforcement learning*

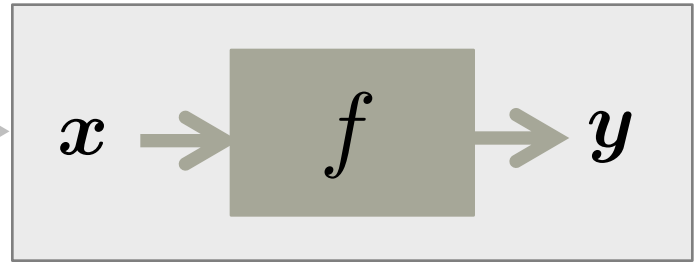
- etwas **vorzuführen** und das System **imitieren** zu lassen

→ *imitation learning*

TRAINIEREN STATT PROGRAMMIEREN

Spiegel Online,
5. Februar 2017

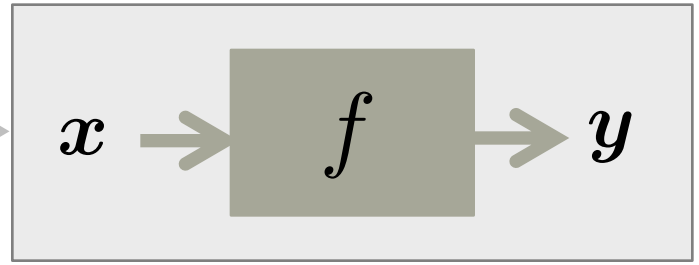
AUS DATEN LERNEN

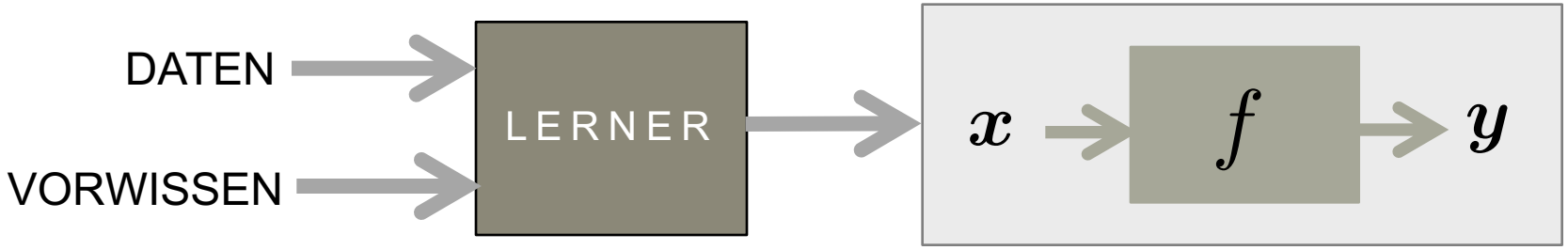


“Machine learning is the science of getting computers to act without being explicitly programmed.”

Andrew Ng, 2013

AUS DATEN LERNEN



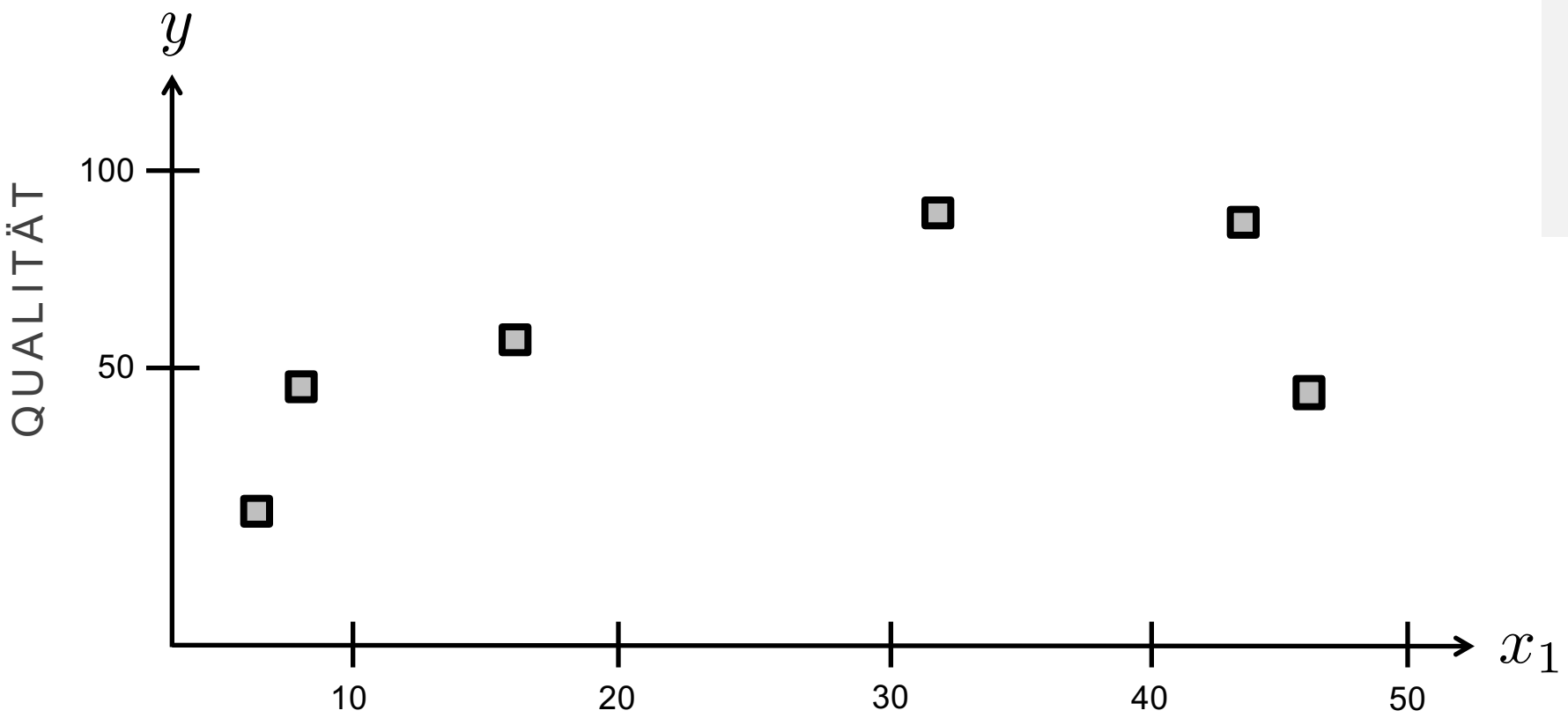


DATEN UND WISSEN

Einstellungen

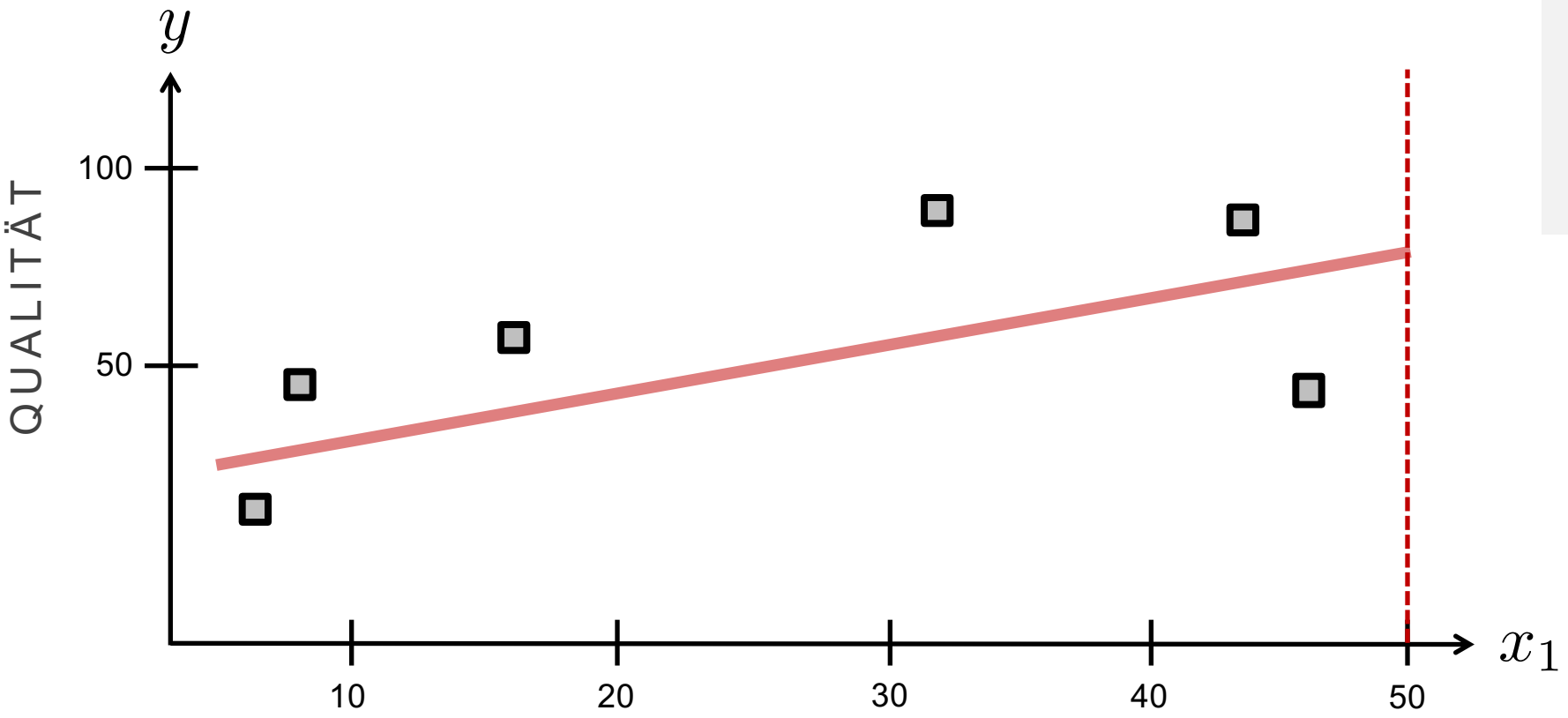
Produktqualität

DATEN UND WISSEN

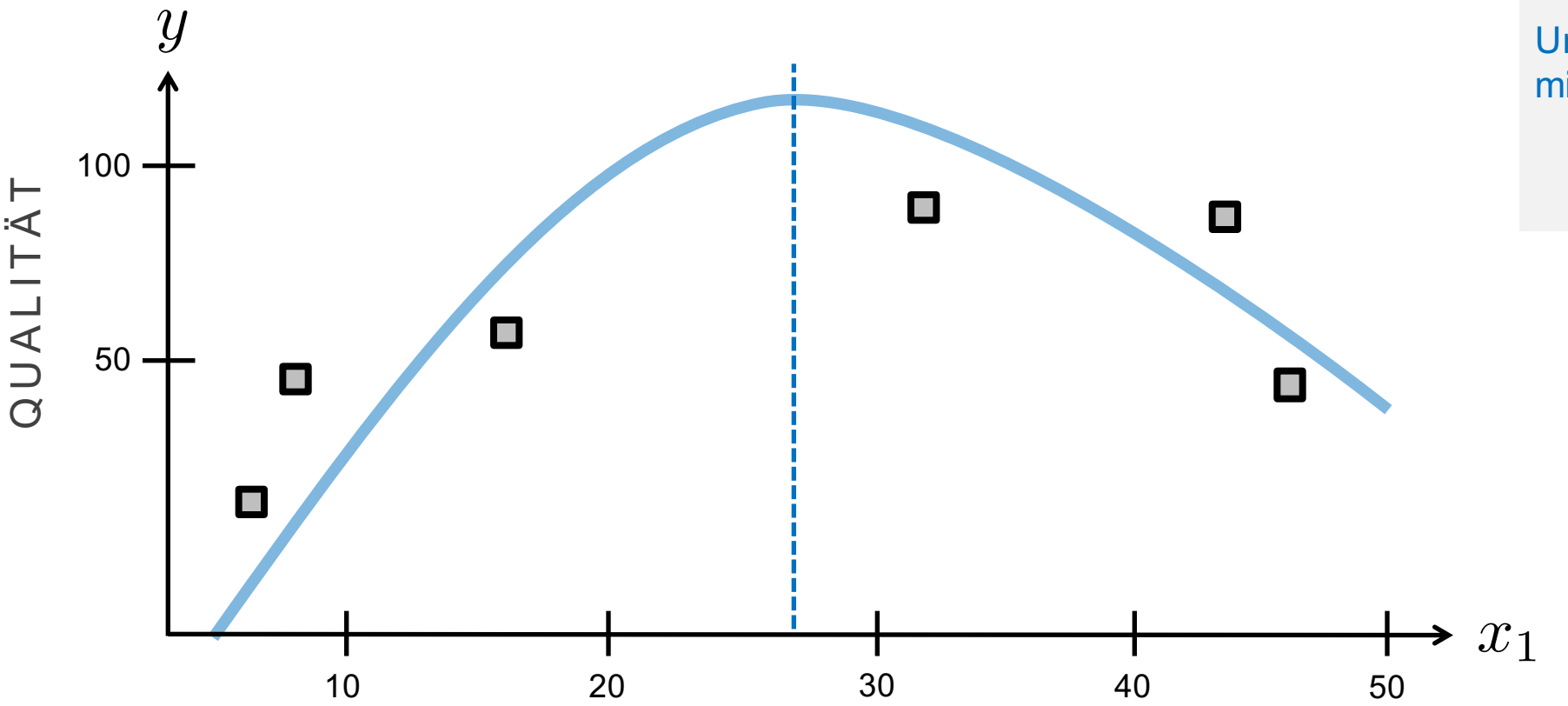


DATEN UND WISSEN

Einfacher Zusammenhang, hohe Messfehler

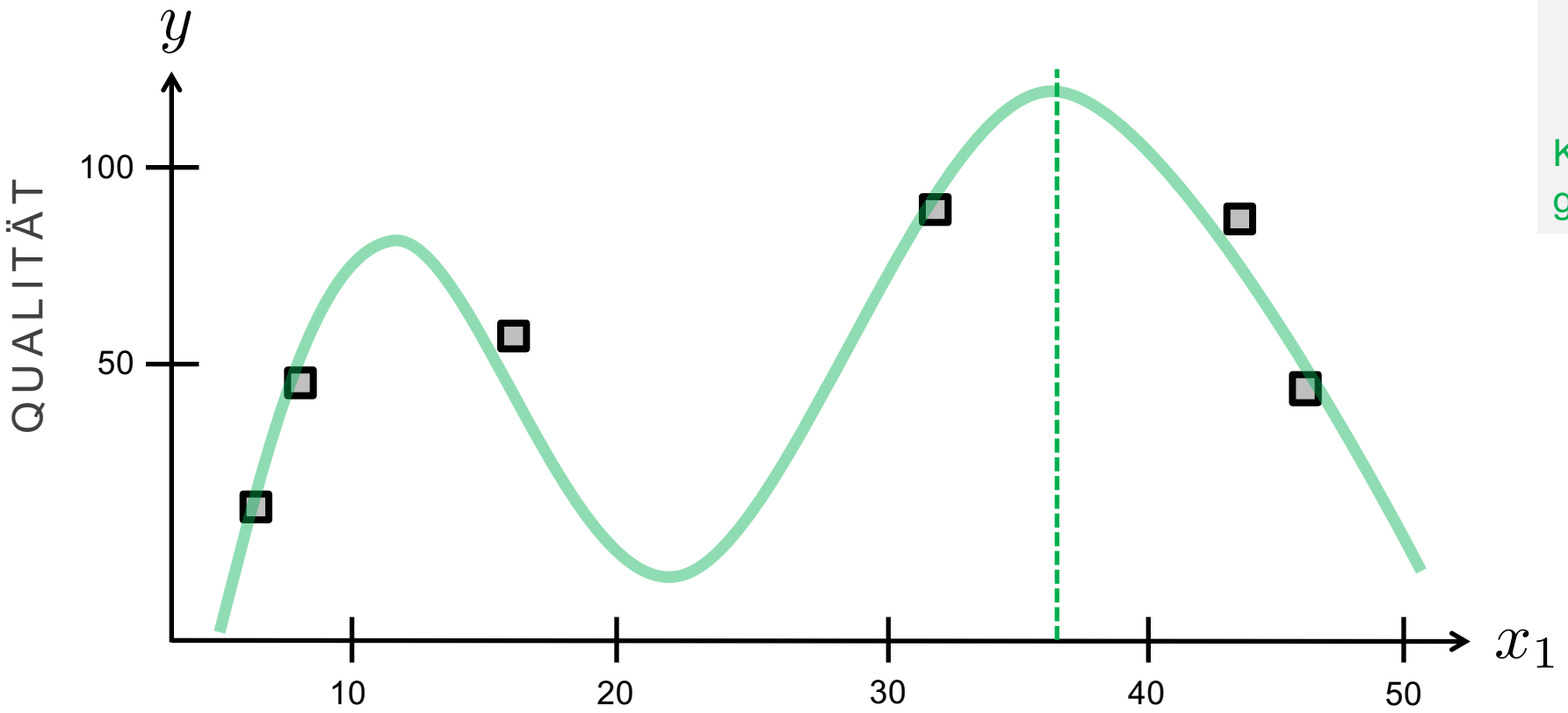


DATEN UND WISSEN



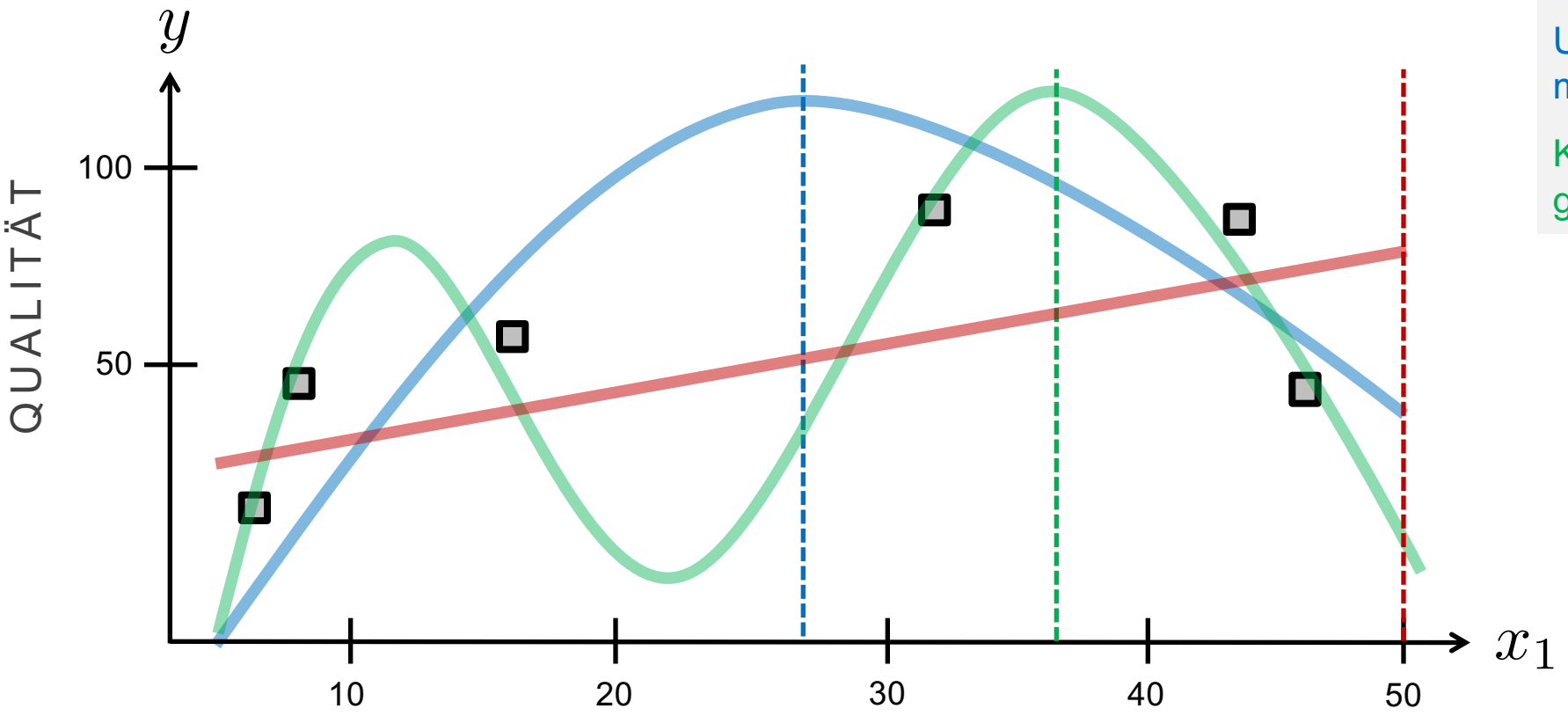
Unimodaler Zusammenhang, mittlere Messfehler

DATEN UND WISSEN



Komplexer Zusammenhang, geringe Messfehler

DATEN UND WISSEN

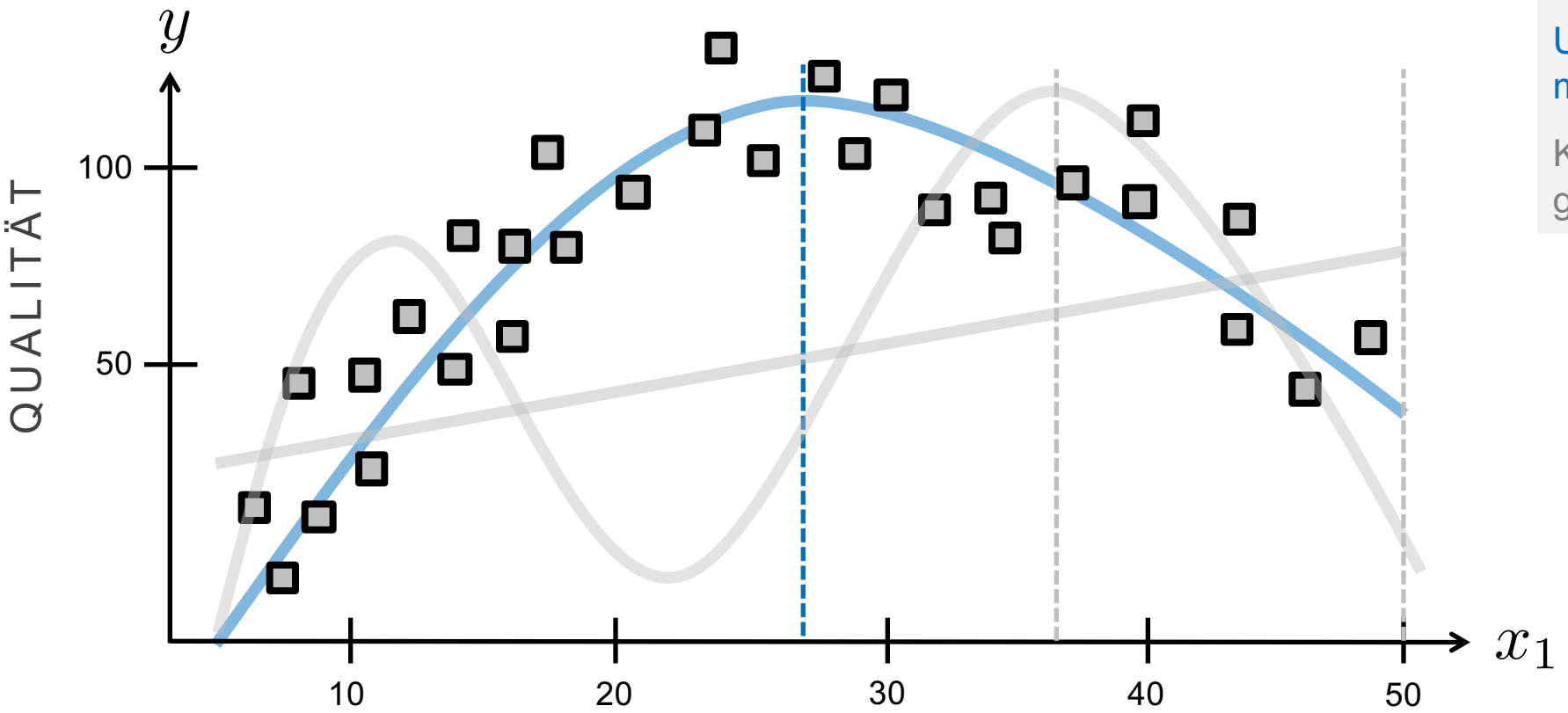


Einfacher Zusammenhang, hohe Messfehler

Unimodaler Zusammenhang, mittlere Messfehler

Komplexer Zusammenhang, geringe Messfehler

VIEL HILFT VIEL ...



Einfacher Zusammenhang, hohe Messfehler

Unimodaler Zusammenhang, mittlere Messfehler

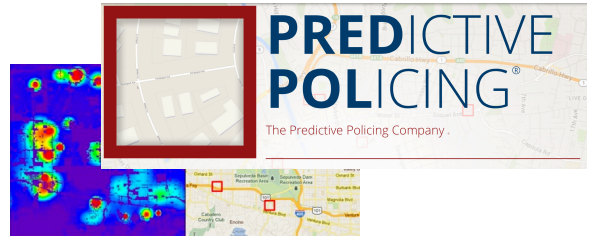
Komplexer Zusammenhang, geringe Messfehler

ANWENDUNGEN DES MASCHINELLEN LERNENS

Gesundheit, Healthcare

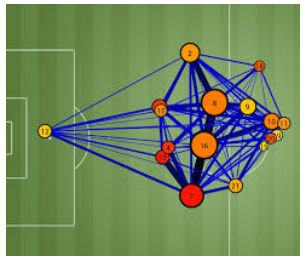
Google twitter facebook Instagram

Social Networks



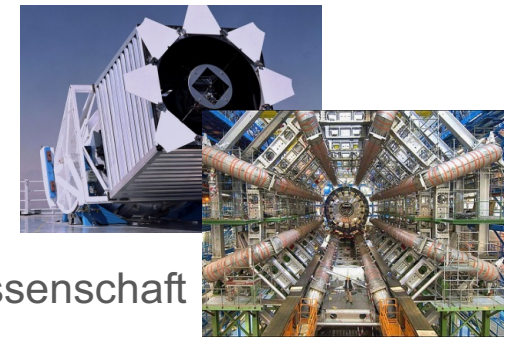
Umwelt, Smart Energy Management

Economics, Finance



Sports Analytics

CPS, Internet-of-Things



Wissenschaft

Smart Traffic

INDUSTRIELLE DIGITALISIERUNG

Predictive Maintenance und Condition Monitoring:
Die Digitalisierung und sensorische Ausstattung von Anlagen ermöglicht eine intelligente Wartung und Zustandsüberwachung von Maschinen.

INDUSTRIELLE DIGITALISIERUNG

Predictive Maintenance und Condition Monitoring:
Die Digitalisierung und sensorische Ausstattung von Anlagen ermöglicht eine intelligente Wartung und Zustandsüberwachung von Maschinen.

Datengetriebene Optimierung:
Das kontinuierliche Erfassen umfangreicher digitaler Informationen bildet die Grundlage der datengetriebenen Optimierung von Prozessen und Anlagen.

INDUSTRIELLE DIGITALISIERUNG

Predictive Maintenance und Condition Monitoring:
Die Digitalisierung und sensorische Ausstattung von Anlagen ermöglicht eine intelligente Wartung und Zustandsüberwachung von Maschinen.

Datengetriebene Optimierung:
Das kontinuierliche Erfassen umfangreicher digitaler Informationen bildet die Grundlage der datengetriebenen Optimierung von Prozessen und Anlagen.

Datengetriebenes Qualitätsmanagement:
Auf ähnliche Weise kann die Qualität von Produkten überwacht werden, sowohl frühzeitig im Prozess als auch nach der Auslieferung.

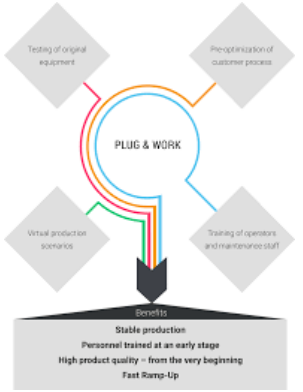
INDUSTRIELLE DIGITALISIERUNG

Lernende Assistenzsysteme:
Moderne Assistenzsysteme erhalten laufend digitales Feedback ihrer Benutzer, wodurch sich Möglichkeiten zur Personalisierung solcher Systeme eröffnen.

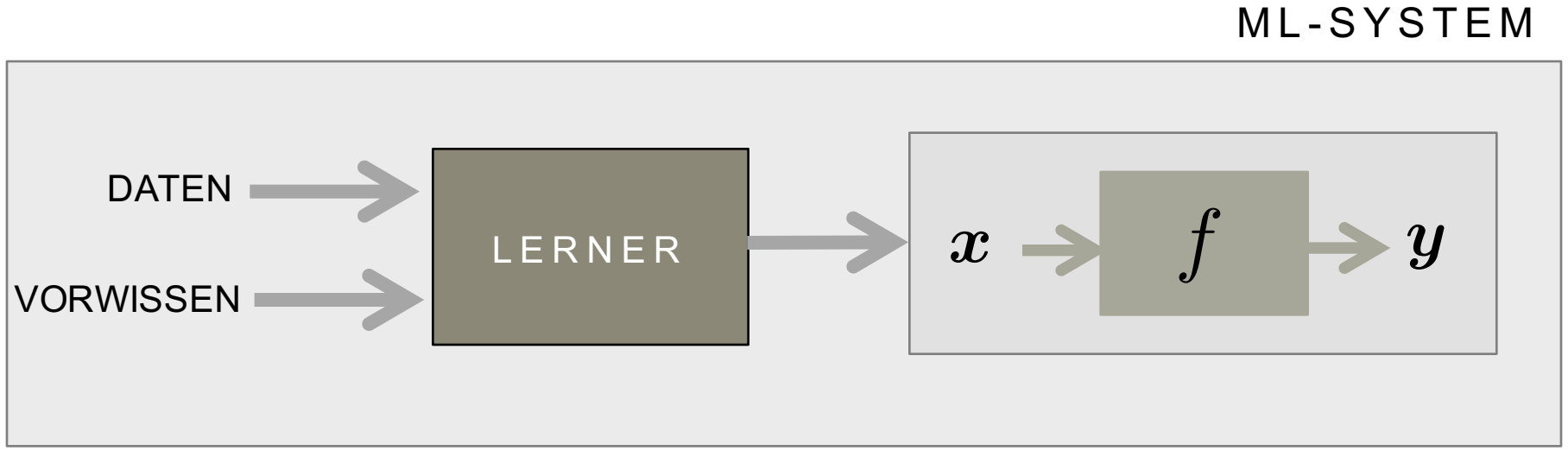
INDUSTRIELLE DIGITALISIERUNG

Lernende Assistenzsysteme:
Moderne Assistenzsysteme erhalten laufend digitales Feedback ihrer Benutzer, wodurch sich Möglichkeiten zur Personalisierung solcher Systeme eröffnen.

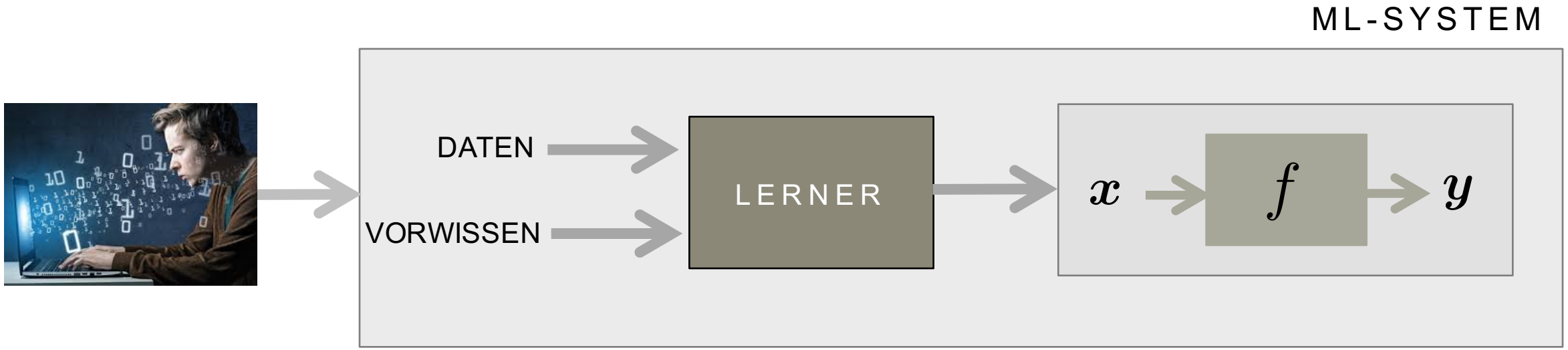
Plug and Work:
Datengetriebene Kalibrierung und Selbstadaptation der Schnittstellen zwischen Komponenten in komplexen verteilten Anlagen.



ML-BASIERTES PROBLEMLÖSEN

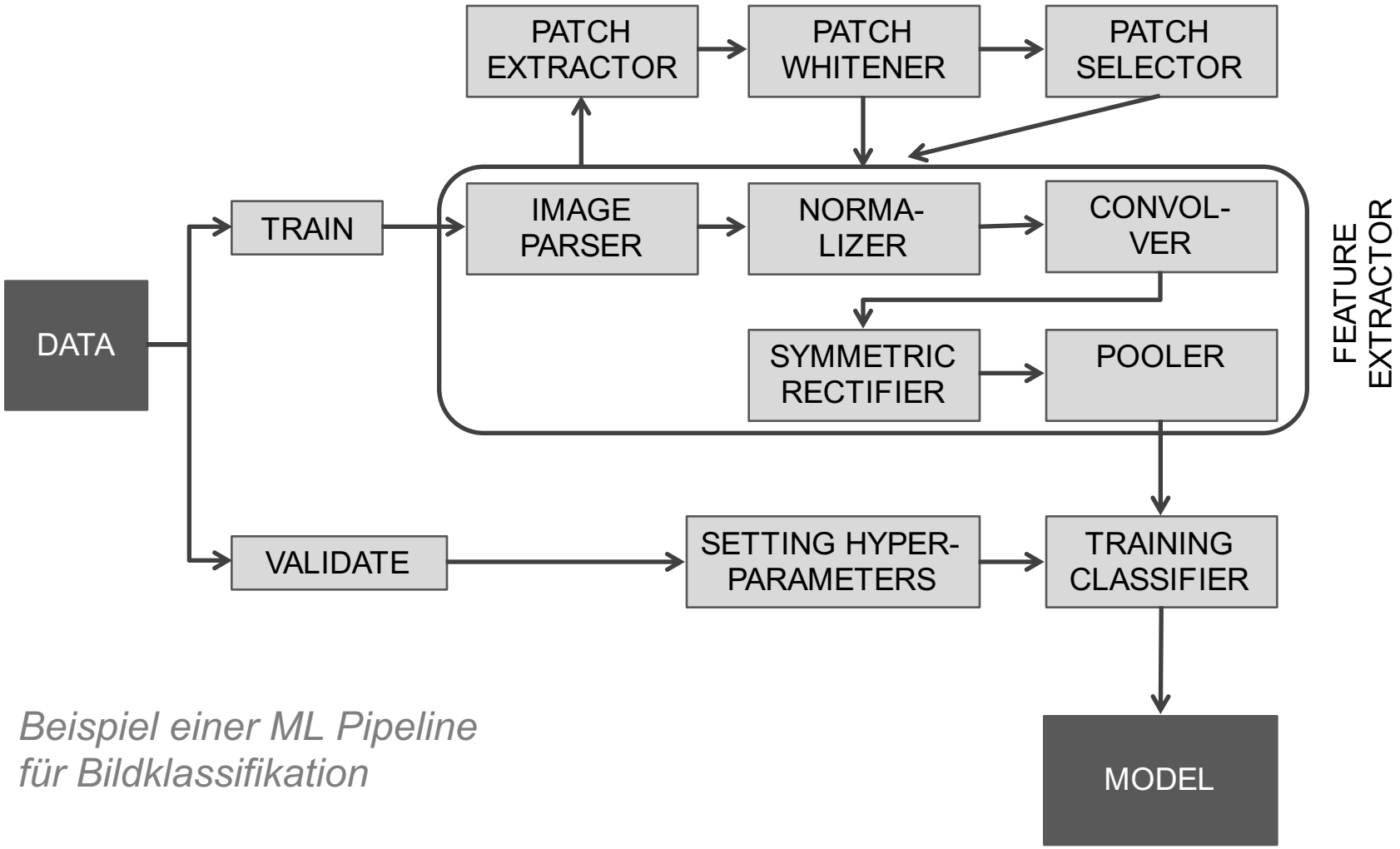


ML-BASIERTES PROBLEMLÖSEN



- Algorithmus wird nicht mehr für das ursprüngliche Problem benötigt, aber für das **Problem, wie man lernt, dieses Problem zu lösen**.
- Beinhaltet adäquate Modellierung und Formalisierung, Datenvorverarbeitung, Induktionsprinzip, algorithmische Umsetzung dieses Prinzips, etc.

ML-BASIERTES PROBLEMLÖSEN

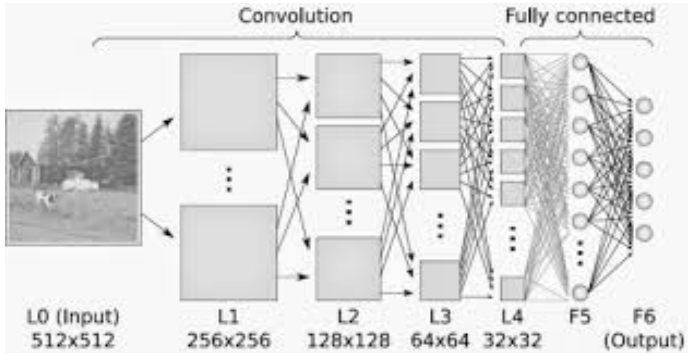


Beispiel einer ML Pipeline für Bildklassifikation

ML-BASIERTES PROBLEMLÖSEN

Ein tiefes neuronales Netz (CNN) kann leicht mehr als 40 Hyper-Parameter haben:

- number of hidden units
- activation function
- convolution kernel width
- implicit zero padding
- weight decay coefficient
- loss function
- weight initialization
- learning rate
- batch size
- dropout rate
- …



Das praktische Lösen eines ML Problems geht einher mit der expliziten oder impliziten Festlegung tausender Freiheitsgrade …

DIE EVOLUTION INTELLIGENTER SYSTEME

```
function GetMin(var a: TList)
var
    i, min, mini: integer;
begin
    min := MaxInt;
    mini := 0;
    for i := 1 to a.len do
        if a.arr[i].G < min t
        begin
            min := a.arr[i].G
            mini := i;
        end;

    GetMin := mini;
end;
```

klassische Programmierung

… ist schwierig für komplexe Probleme

```
mann(adam).
mann(tobias).
mann(frank).
frau(eva).
frau(daniela).
frau(ulrike).
vater(adam,tobias).
vater(tobias,frank).
vater(tobias,ulrike).
mutter(eva,tobias).
mutter(daniela,frank).
mutter(daniela,ulrike).
```

Wissensbasierte Systeme

… leidet unter dem Flaschenhals des Wissenserwerbs

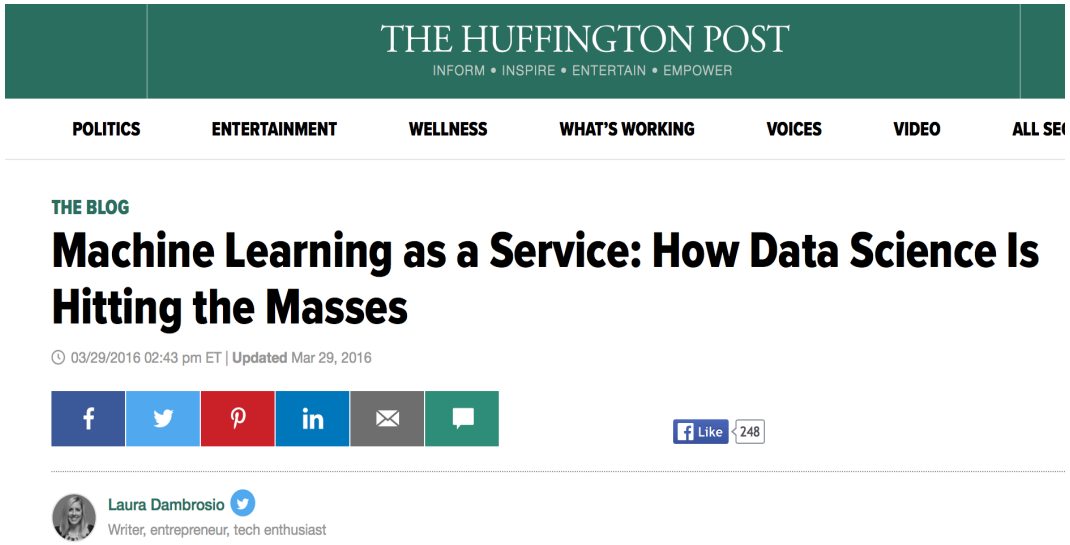
```
# Spot Check Algorithms
models = []
models.append(('LR', Logis
models.append(('LDA', Line
models.append(('KNN', KNei
models.append(('CART', Dec
models.append(('NB', Gauss
models.append(('SVM', SVC(
# evaluate each model in t
results = []
names = []
for name, model in models:
    kfold = model_selectio
    cv_results = model_sel
    results.append(cv_resu
    names.append(name)
    msg = "%s: %f (%f)" %
    print(msg)
```

“implizite” Programmierung

… erfordert hohes Maß an Expertise in ML

automatisiertes Machine Learning

AUTOMATISIERTES MASCHINELLES LERNEN



THE HUFFINGTON POST

INFORM • INSPIRE • ENTERTAIN • EMPOWER

POLITICS ENTERTAINMENT WELLNESS WHAT'S WORKING VOICES VIDEO ALL SE

THE BLOG

Machine Learning as a Service: How Data Science Is Hitting the Masses

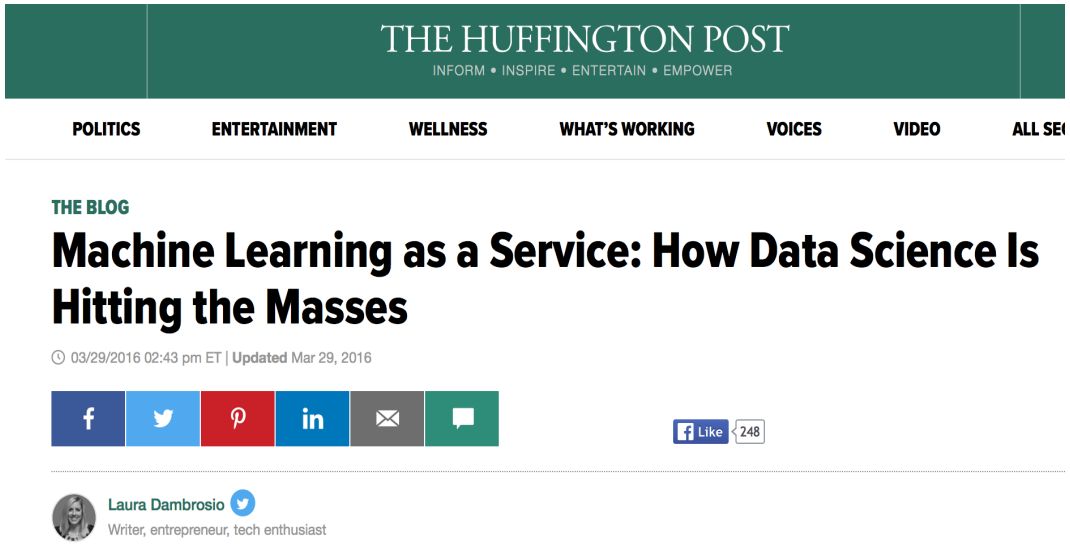
03/29/2016 02:43 pm ET | **Updated** Mar 29, 2016

Like 248

Laura Dambrosio

Writer, entrepreneur, tech enthusiast

AUTOMATISIERTES MASCHINELLES LERNEN



THE HUFFINGTON POST
INFORM • INSPIRE • ENTERTAIN • EMPOWER

POLITICS ENTERTAINMENT WELLNESS WHAT'S WORKING VOICES VIDEO ALL SE

THE BLOG

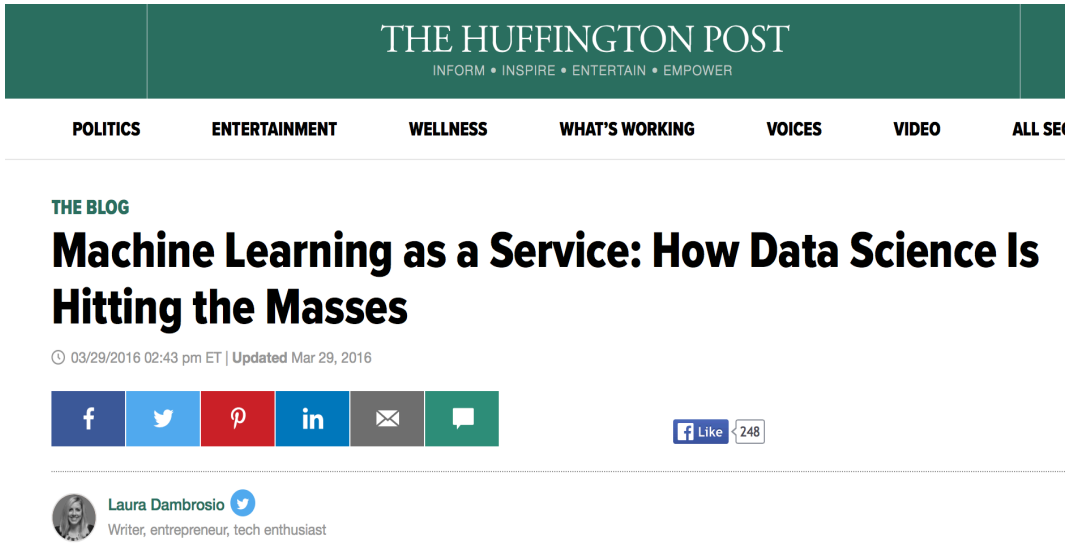
Machine Learning as a Service: How Data Science Is Hitting the Masses

03/29/2016 02:43 pm ET | **Updated** Mar 29, 2016

Like 248

Laura Dambrosio
Writer, entrepreneur, tech enthusiast

AUTOMATISIERTES MASCHINELLES LERNEN



THE HUFFINGTON POST
INFORM • INSPIRE • ENTERTAIN • EMPOWER

POLITICS ENTERTAINMENT WELLNESS WHAT'S WORKING VOICES VIDEO ALL SE

THE BLOG

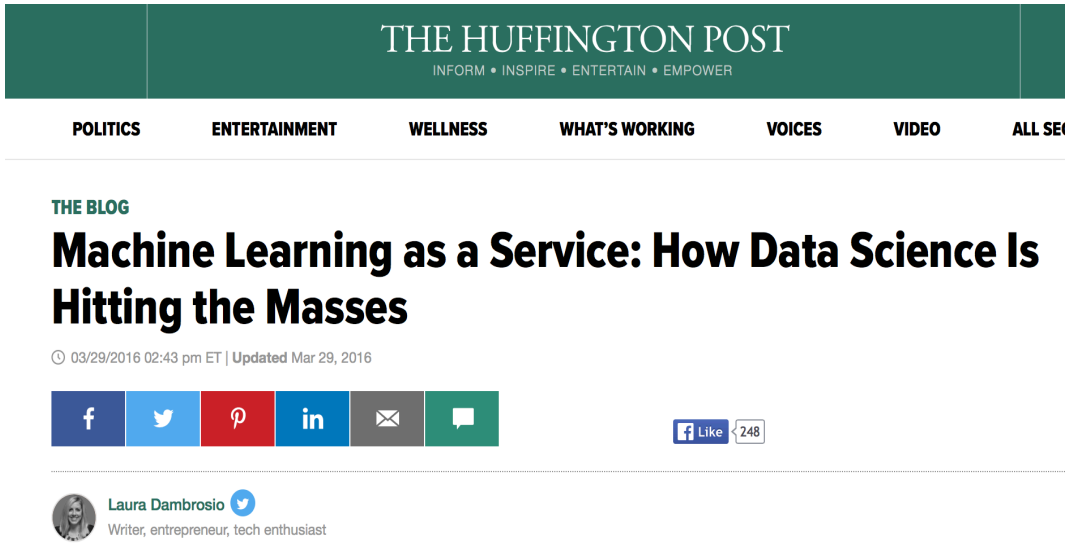
Machine Learning as a Service: How Data Science Is Hitting the Masses

03/29/2016 02:43 pm ET | **Updated** Mar 29, 2016

Like 248

Laura Dambrosio
Writer, entrepreneur, tech enthusiast

AUTOMATISIERTES MASCHINELLES LERNEN



SFB 901
ON - THE - FLY COMPUTING

- Einige **AutoML** Tools existieren bereits, andere werden derzeit entwickelt.
- Diese Tools realisieren im Wesentlichen eine systematische Suche und Bewertung im Raum der ML Pipelines.

ML, AUTO-ML UND META-LERNEN

Instruiere den Computer, wie das Problem zu lösen ist.

`{MALE, FEMALE}`

Daten

Instruiere den Computer, wie man lernt, das Problem zu lösen.

Algorithmus (Prädiktor)

ML, AUTO-ML UND META-LERNEN

Instruiere den Computer, wie das Problem zu lösen ist. → {MALE, FEMALE}

Daten → *Instruiere den Computer, wie man lernt, das Problem zu lösen.* → Algorithmus (Prädiktor)

ML Problem (Daten) → *Instruiere den Computer, wie man einen guten Weg findet, um zu lernen, wie man man das Problem löst.* → ML Algorithmus (Lerner)

ML, AUTO-ML UND META-LERNEN

Instruiere den Computer, wie das Problem zu lösen ist. → {MALE, FEMALE}

Daten → *Instruiere den Computer, wie man lernt, das Problem zu lösen.* → Algorithmus (Prädiktor)

ML Problem (Daten) → *Instruiere den Computer, wie man einen guten Weg findet, um zu lernen, wie man man das Problem löst.* → ML Algorithmus (Lerner)

ML Problem (Daten) → *Instruiere den Computer, wie man lernt zu lernen, das Problem zu lösen.* → AutoML Algorithmus

ZUSAMMENFASSUNG

```
function GetMin(var a: TList)
var
    i, min, mini: integer;
begin
    min := MaxInt;
    mini := 0;
    for i := 1 to a.len do
        if a.arr[i].G < min t
        begin
            min := a.arr[i].G
            mini := i;
        end;

    GetMin := mini;
end;
```

klassische Programmierung

```
mann(adam).
mann(tobias).
mann(frank).
frau(eva).
frau(daniela).
frau(ulrike).
vater(adam,tobias).
vater(tobias,frank).
vater(tobias,ulrike).
mutter(eva,tobias).
mutter(daniela,frank).
mutter(daniela,ulrike).
```

Wissensbasierte Systeme

```
# Spot Check Algorithms
models = []
models.append(('LR', Logis
models.append(('LDA', Line
models.append(('KNN', KNei
models.append(('CART', Dec
models.append(('NB', Gauss
models.append(('SVM', SVC(
# evaluate each model in t
results = []
names = []
for name, model in models:
    kfold = model_selectio
    cv_results = model_sel
    results.append(cv_resu
    names.append(name)
    msg = "%s: %f (%f)" %
    print(msg)
```

"implizite" Programmierung

automatisiertes Machine Learning

- hohes Maß an Domänen- und Lösungswissen
- direkte algorithmische Lösung (Informatiker = Experte) bzw.
- generische algorithmische Lösung (Experte vs. Informatiker)

- Daten können Domänenwissen kompensieren (aber nicht komplett ersetzen)
- Expertise/Wissen in ML
- ML bzw. Auto-ML bzw. Meta-ML Algorithmen

→ *mehr **Daten**, **Wissen** wird abstrakter und **Algorithmen** werden generischer*