Online Machine Learning

Tim Pietz

Agenda

1. Negativbeispiel: Offline Learning
2. Online Learning
3. Herausforderungen & Algorithmen
4. Implementationen
5. Charakteristiken

Negativbeispiel: Stock Market Prediction

Basierend auf vergangenen Verlauf

Zukünftige Entwicklung vorhersagen

Negativbeispiel: Stock Market Prediction

Quelle: google.de

- Auf alten Daten trainieren
- Genauigkeit degradiert mit der Zeit
- Aktualisierung notwendig

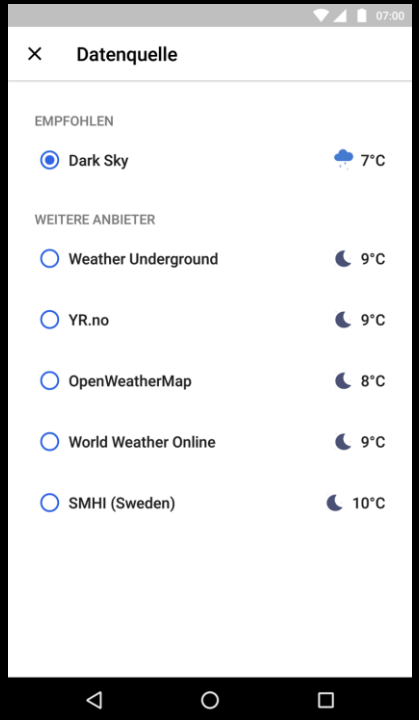
Beispiel: Wetterdienste

Gegeben Zugriff auf 5 Wetterdienste

Temperatur vorhersagen

Beispiel: Wetterdienste

a) Einen auswählen
   - Degradierung
b) Nutzer auswählen lassen
c) Dienste dynamisch integrieren
   - Jedem Dienst wird ein Gewicht zugewiesen
   - Kontinuierliche Neugewichtung



Quelle: Screenshot von
“Weather Timeline- Forecast”

Beispiel: Wetterdienste

Given n predictors

Initialize:

$w_i \leftarrow 1$

Monitor stream:

Receive predictions $x_1, ..., x_n$

$\hat{y} \leftarrow \frac{1}{L} \Sigma(w_i \cdot x_i)$ with $L = \Sigma(w_i)$

Submit prediction $\hat{y}$

Receive correct value y

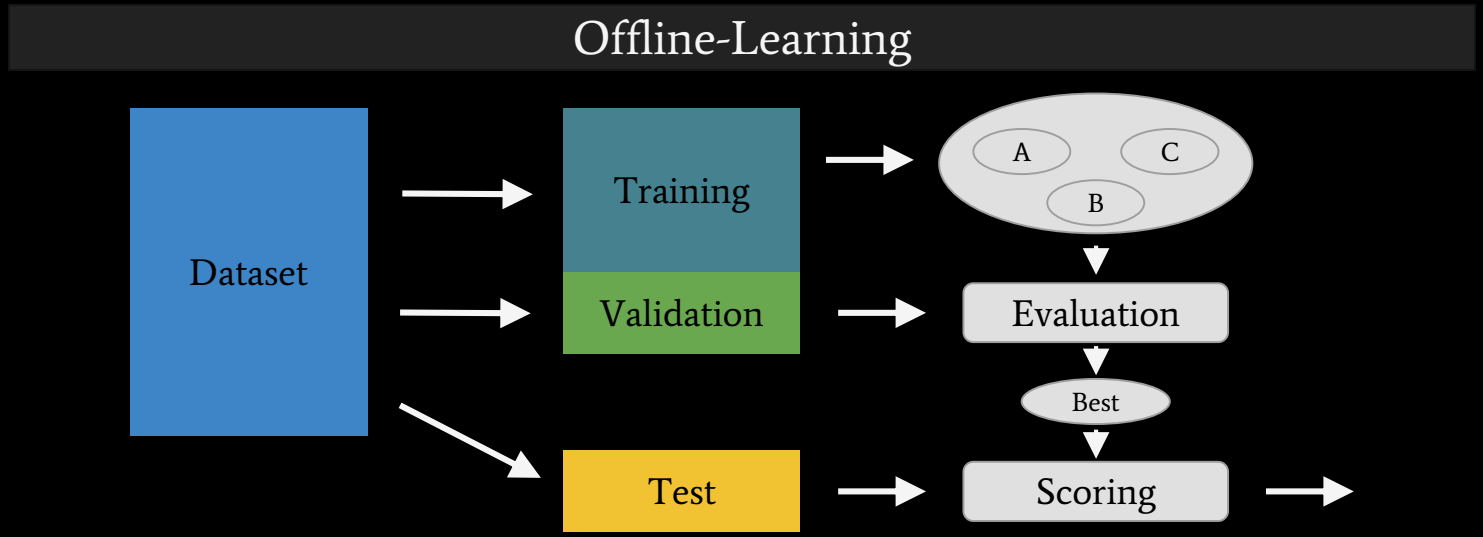
For predictors i with $x_i \neq y$:

$w_i \leftarrow w_i / (1 + \alpha \cdot \text{Loss}(x_i, y))$ with $\text{Loss}(a, b) = | a - b |$

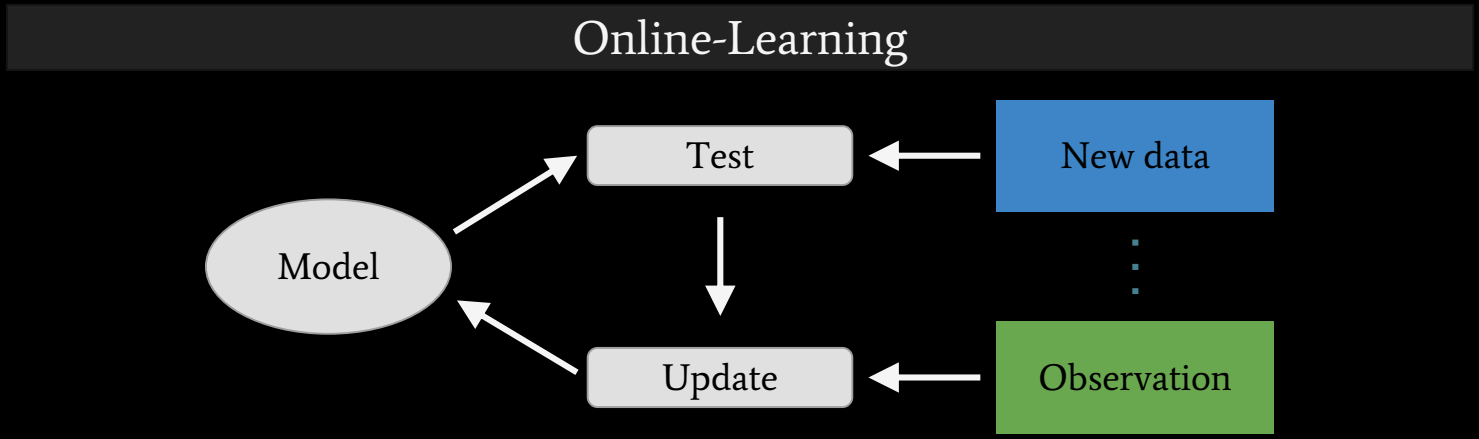
Beispiel: Wetterdienste

- Algorithmus: Expert Advice / Voting
- Online-Learning zur kontinuierlichen Verbesserung
- Fluss an Trainingsdaten ausgenutzt
- Daten werden nicht gespeichert

Offline-Learning vs Online-Learning



Offline-Learning vs Online-Learning

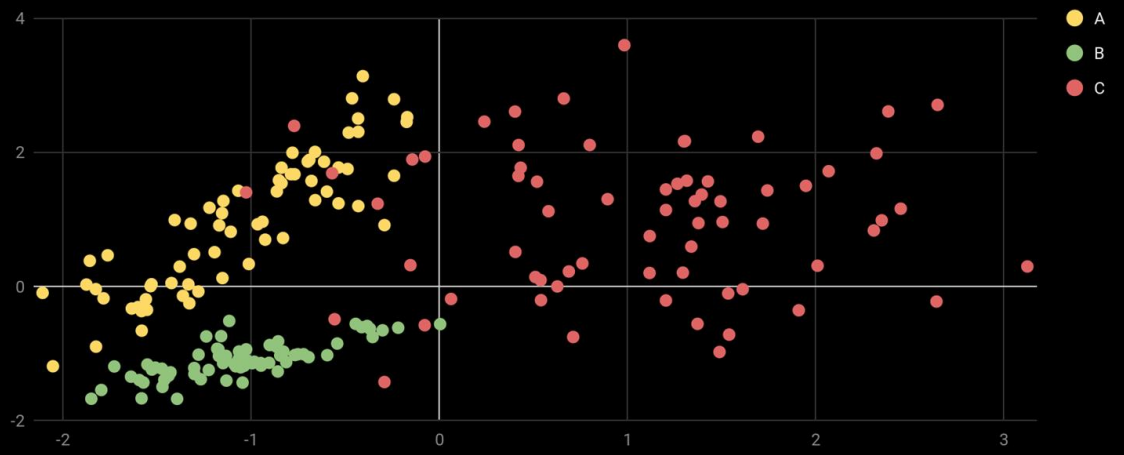


Offline-Learning vs Online-Learning

	Offline Learning	Online Learning
Training	Vor Nutzung	Kontinuierlich
Zugriff auf Trainingsdaten	(Mehrfach) während Trainingsphase	Einmalig
Zielfunktion	Stationär	Dynamisch

Herausforderungen: Catastrophic Forgetting

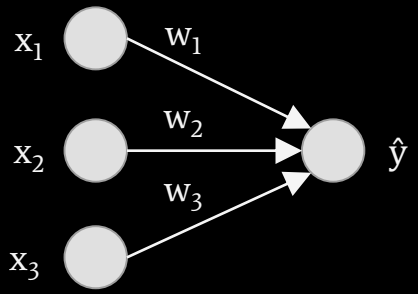
- Vollständiges, abruptes Vergessen bereits gelernter Informationen
- Erst A und B, danach C



Stabilität ↔ Reaktivität

Algorithmus: Perzeptron

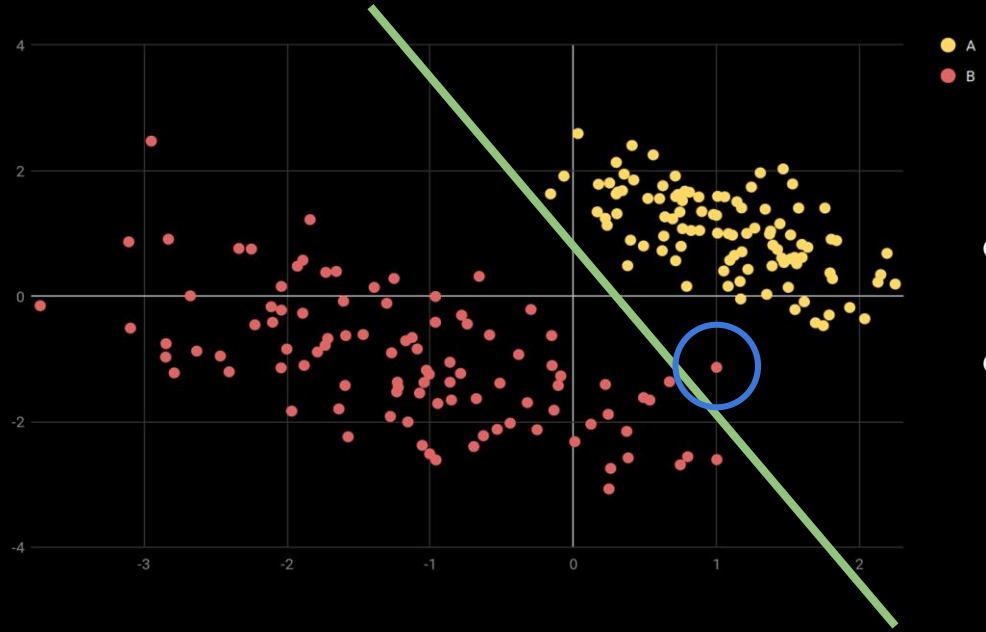
- Eingabevektor x
- Gewichtungsvektor w
- Ausgabewert $\hat{y}$



$$\hat{y} = 1 \quad \text{gdw.} \quad \Sigma(w_i \cdot x_i) \geq \Theta \quad \text{sonst } -1$$

$$\hat{y} = \text{sign}(\Sigma(w_i \cdot x_i) - \Theta) = \text{sign}(w \cdot x - \Theta)$$

Algorithmus: Perzeptron

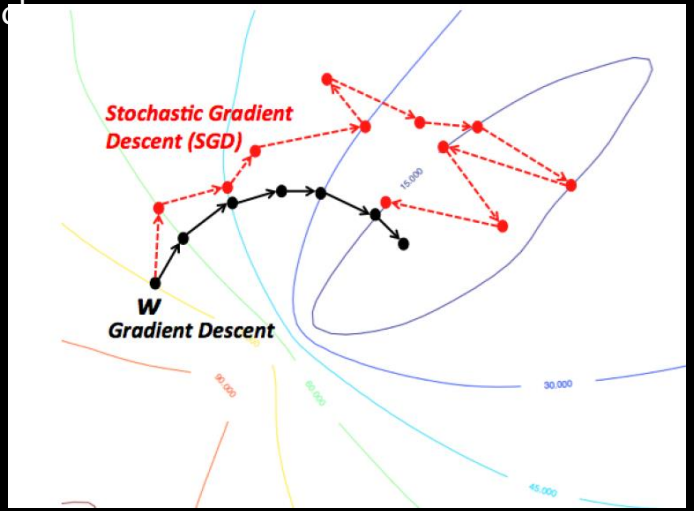


- Anpassung der Gewichte notwendig
- Minimierung der Verlustfunktion

Weight Optimization

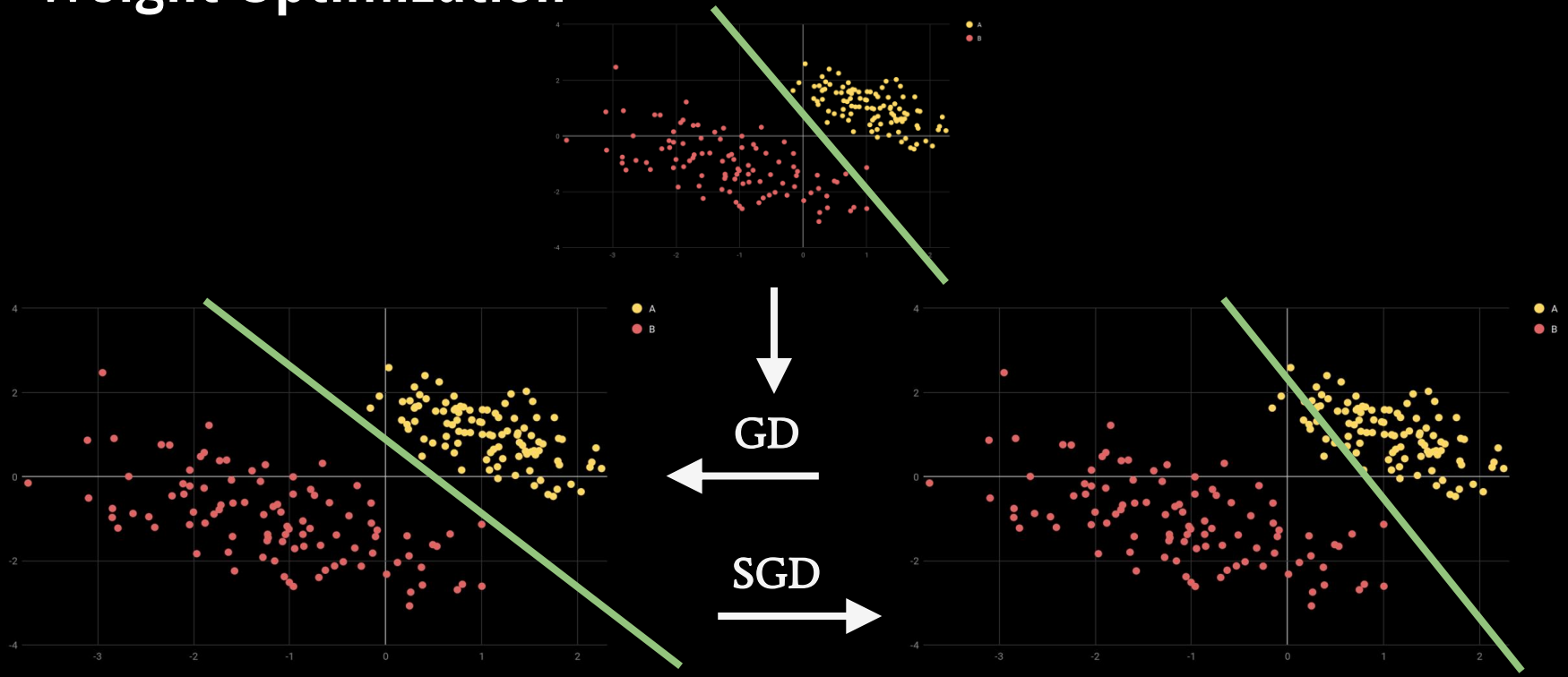
- Gradient Descent
  - Betrachtet alle Datenpunkte, um beste Änderung zu berec
  - Aufwendig in Big Data Größenordnungen

- Stochastic Gradient Descent
  - Betrachtet nur aktuellen Datenpunkt
  - Datenpunkte können nach Einlesen verworfen werden
  - Fehleranfälliger

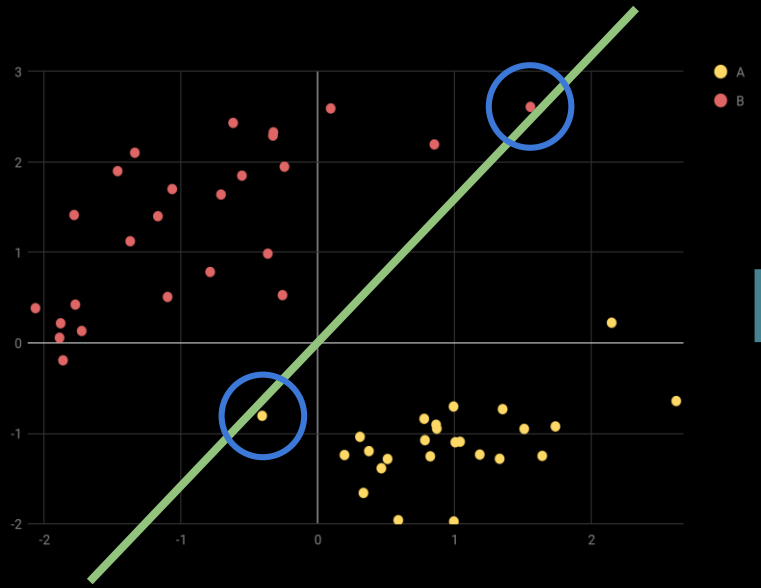


Quelle: wikidocs.net/3413

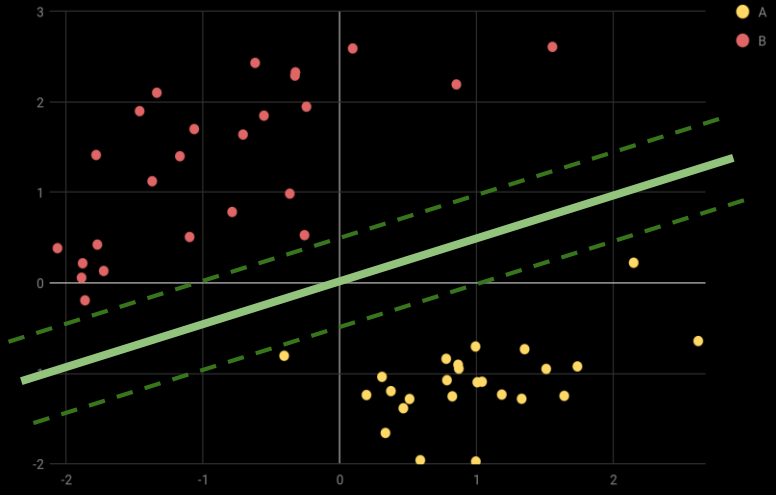
Weight Optimization



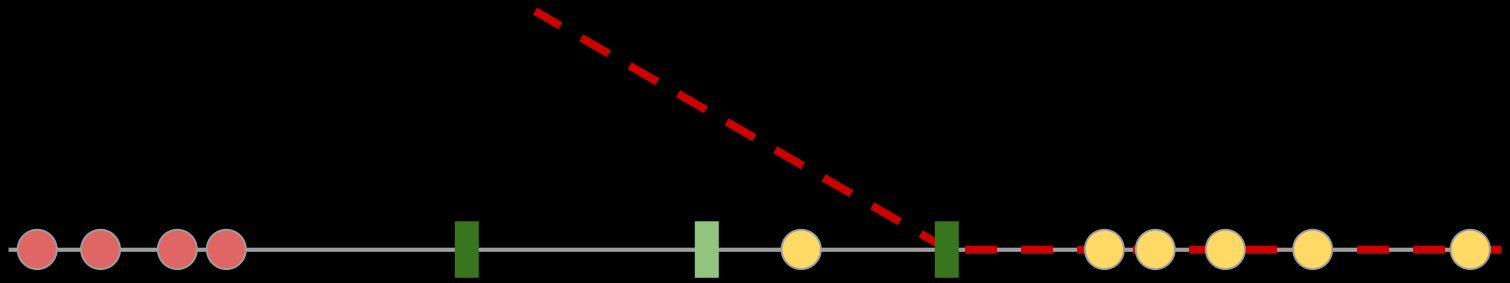
Algorithmus: Passive Aggressive



Large Margin Classifier

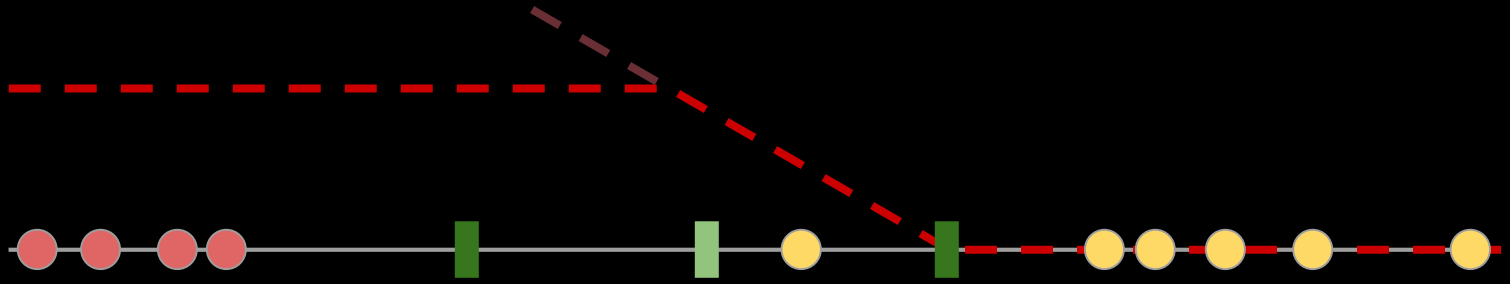


Algorithmus: Passive Aggressive



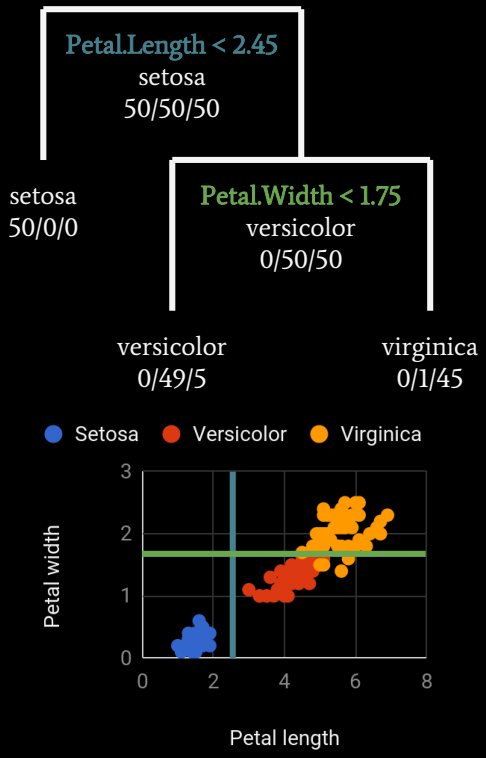
- Verlust > 0, wenn falsch klassifiziert oder innerhalb Pufferzone
- Update aktualisiert Gewichte so, dass der Punkt an der Grenze liegt

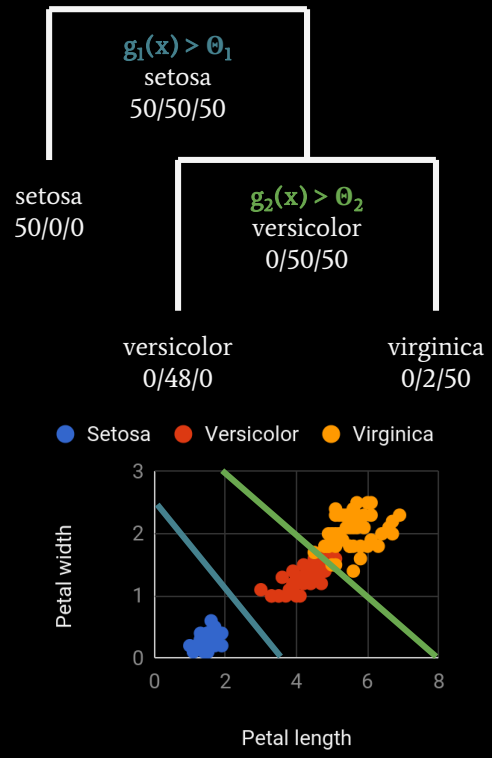
Algorithmus: Passive Aggressive-I



- Verlustfunktion begrenzen
- Beschränkt max. Gewichtsänderung
- Geringere Sensibilität gegen Outlier

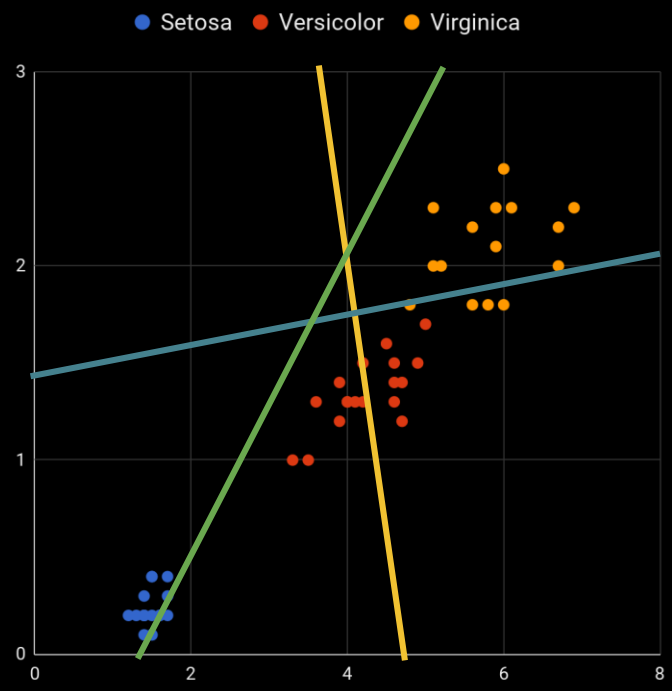
Algorithmus: Online Random Forests





Quelle: Nach [2] Machine Learning, Lecture BigData Analytics

Algorithmus: Online Random Forests



- Start mit einzelnen Knoten
- Mögliche Testfunktionen zufällig wählen
- Evaluierung mit jedem Datenpunkt

17/18/15

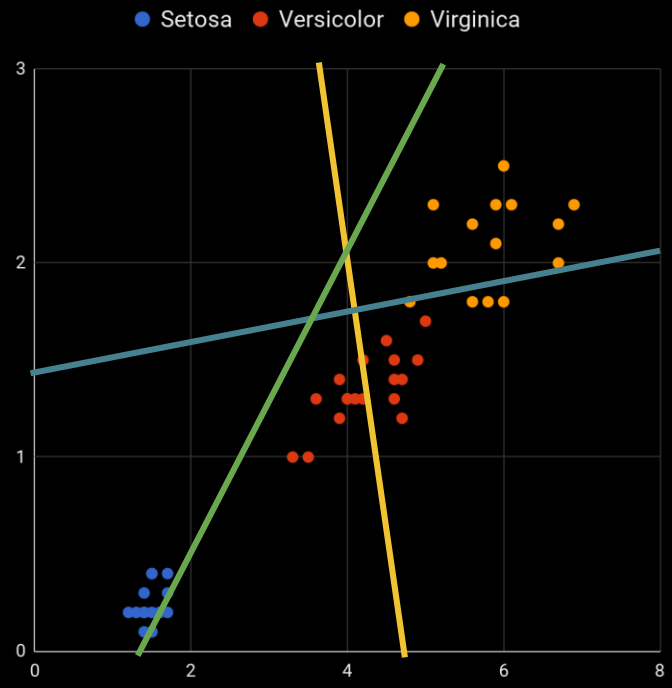
Mögliche Splits:

14/0/0 und 3/18/15

17/9/0 und 0/9/15

0/0/12 und 17/18/3

Algorithmus: Online Random Forests



Mögliche Splits:

14/0/0 und 3/18/15

17/9/0 und 0/9/15

0/0/12 und 17/18/3

Wenn

Datenpunkte ausgewertet > α
und Informationsgewinn > β

Dann entsprechend Aufsplitten

Algorithmus: Online Random Forests

Entscheidungswald:
Mehrere Entscheidungsbäume trainieren
Um vereinzelte Fehlklassifikation auszugleichen

- Datenpunkte zufällig häufig eintrainieren ⇒ Variation
- Voting: Häufigste Vorhersage wird ausgegeben

Algorithmus: Online Random Forests

Nicht-Statische Zielfunktion:

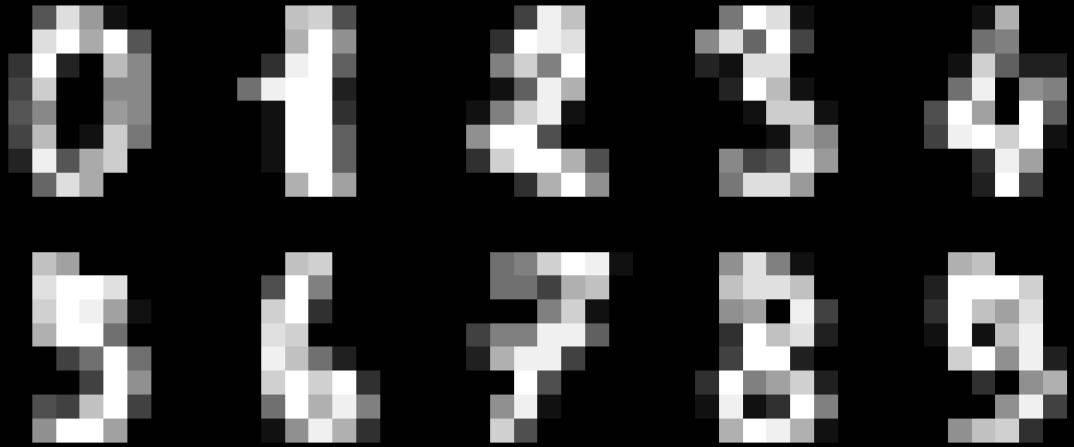
- Nicht-Trainingsdaten zum Testen verwenden
- Bäume mit hoher Fehlerquote zurücksetzen

Herausforderung: Modellkomplexität

- Online Random Forests
  - Anzahl Datenpunkte für Split
  - Informationsgewinn durch Split
  - Anzahl der zufälligen Tests
  - Anzahl Bäume

- Metaparameter bei offline Learning manuell gewählt
  - Generalisierung durch Modellkomplexität
- Datensatz bei Online Learning unbekannt

Implementation



8px × 8px 64 Dimensionen Wertebereich 0-16 1800 Datenpunkte

Implementation

```
stream = DigitsStream()
w = np.zeros(stream.get_dimensionality())

while stream.has_next():

    d = stream.get_next_data()
    prediction = 1 if np.dot(d, w) > 0 else -1

    y = stream.get_actual_value()
    loss = min(1, max(0, 1 - y * np.dot(d, w)))

    w = w + y * loss * d
```

Implementation

Quelle: scikit-learn.org

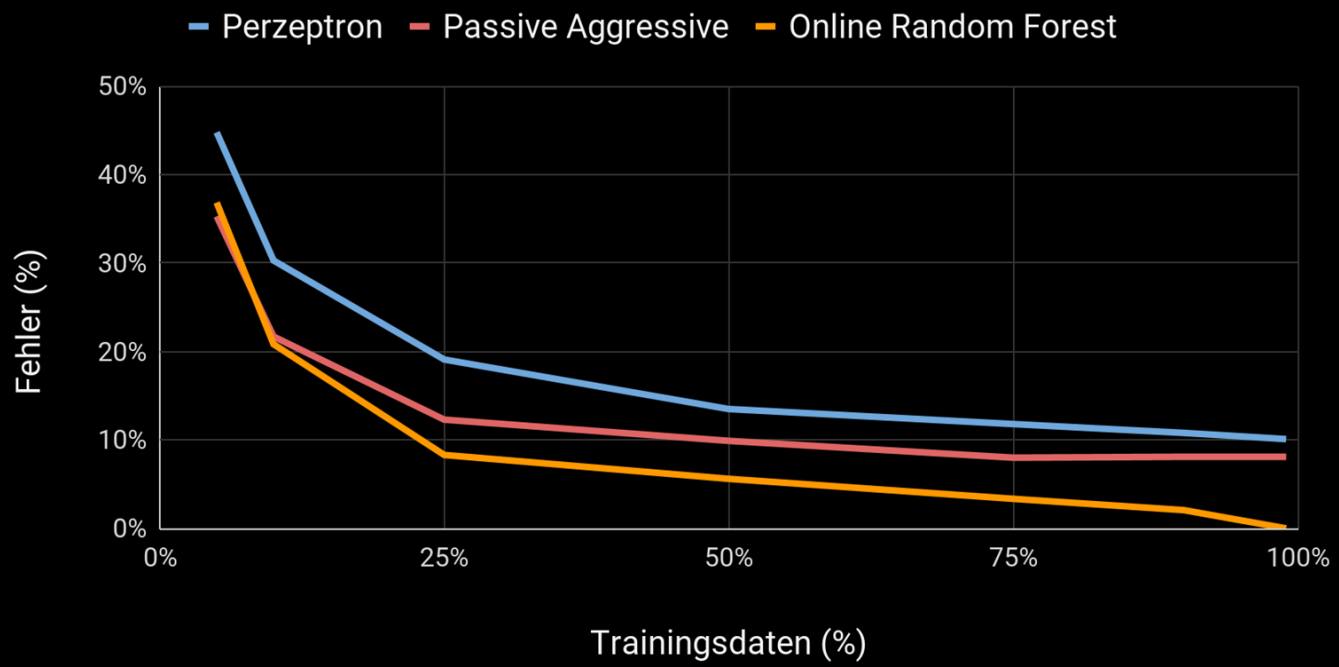
```
digits = datasets.load_digits()
X, y = digits.data, digits.target
X_train, X_test, y_train, y_test = \
    train_test_split(X, y, test_size=0.7)

clf = PassiveAggressiveClassifier(C=1.0, max_iter=1)

clf.fit(X_train, y_train)

score = clf.score(X_test, y_test)
```

Implementation



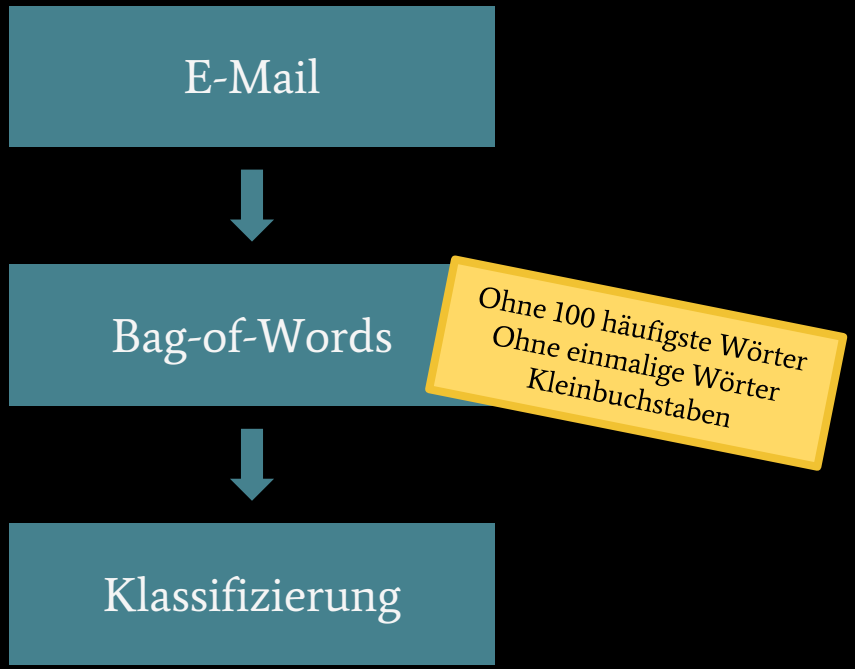
ORF: 100 Bäume / max. Tiefe 20 / 5 Tests

Charakteristiken

E-Mail Klassifikation in nutzerspezifizierte Ordner

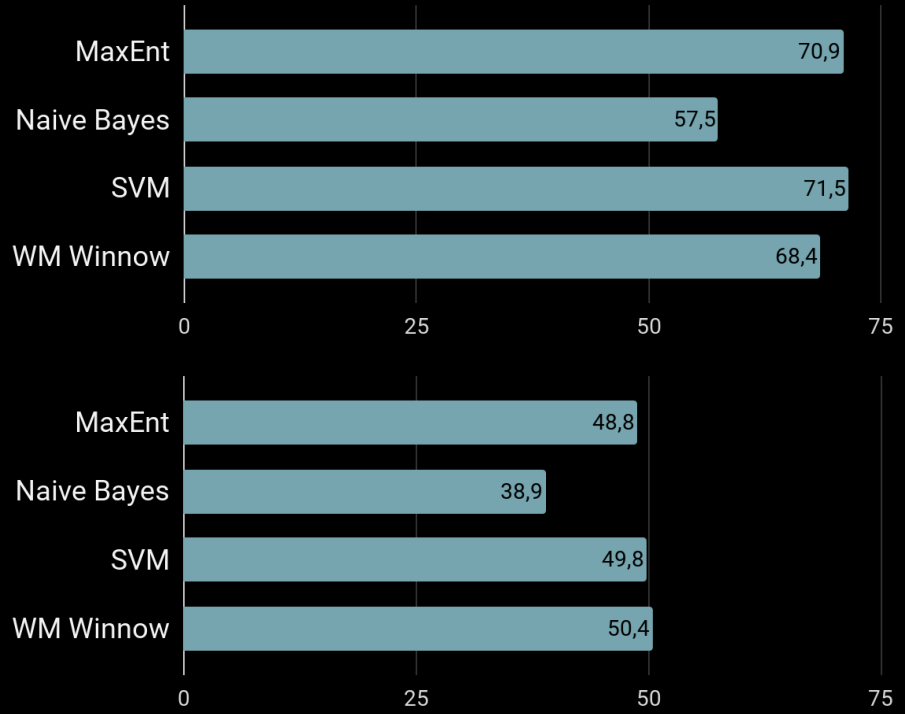
- Kontinuierliches Erstellen und Entfernen von Ordnern
- Inhaltliche Änderungen
- Keine eindeutige Thematik: “Todo” oder “Antwort erforderlich”
- Konversationen häufig gemeinsames Thema

Charakteristiken



Charakteristiken

Charakteristiken



Enron

- 30 Ordner pro Person
- 100 Mails pro Ordner

SRI

- 20 Ordner pro Person
- 25 Mails pro Ordner

Quelle: Nach [2] Automatic Categorization of Email into Folders: Benchmark Experiments on Enron and SRI Corpora

Charakteristiken

Enron user	MaxEnt	Naive Bayes	SVM	WM Winnow
beck-s	55,8	32	**56,4**	49,9
farmer-d	76,6	64,8	**77,5**	74,6
kaminski-v	55,7	46,1	**57,4**	51,6
kitchen-l	58,4	35,6	**59,1**	54,6
lokay-m	**83,6**	75	82,7	81,8
sanders-r	71,6	56,8	**73**	72,1
williams-w3	94,4	92,2	**94,6**	94,5
	70,9	57,5	**71,5**	68,4

Quelle: Nach [2] Automatic Categorization of Email into Folders: Benchmark Experiments on Enron and SRI Corpora

Charakteristiken

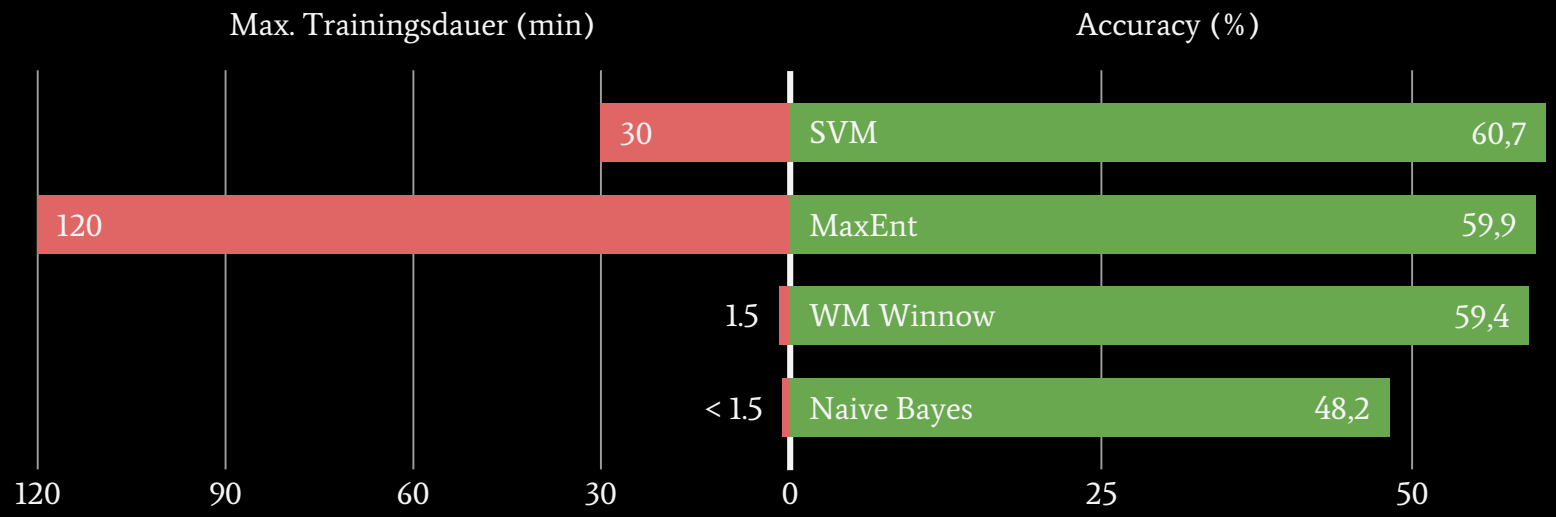
SRI user	MaxEnt	Naive Bayes	SVM	WM Winnow
acheyer	37,1	27	**38,5**	37
bmark	23,3	13,5	**24,3**	23,9
disrael	36,5	36,5	40,7	**41,4**
mgervasio	42,2	31,3	43,5	**50,3**
mgondek	75,2	55,8	**75,2**	74,9
rperrault	66,4	49,8	**68,1**	62,7
vchaudhri	61,1	58,7	58,6	**62,7**
	48,8	38,9	49,8	**50,4**

Quelle: Nach [2] Automatic Categorization of Email into Folders: Benchmark Experiments on Enron and SRI Corpora

Charakteristiken

Quelle: Nach [2] Automatic Categorization of Email into Folders: Benchmark Experiments on Enron and SRI Corpora

Charakteristiken



Quelle: Nach [2] Automatic Categorization of Email into Folders: Benchmark Experiments on Enron and SRI Corpora

Zusammenfassung

- **Online Learning:** Kontinuierliches Lernen veränderlicher Zielfunktionen
  - Vergleich Online Learning vs Batch Learning
- Herausforderungen: **Catastrophic Forgetting** und **Modellkomplexität**
- 3 Algorithmen: **Expert Advice, Perzeptron** und **Passive-Aggressive**
- **Gradient Descent** und **Stochastic Gradient Descent**
- **Online Random Forests**
- **Implementationen**
- **Charakteristiken**

Quellen

- Tavish Srivastava: “Introduction to Online Machine Learning: Simplified”, 27.01.2015, https://www.analyticsvidhya.com/blog/2015/01/introduction-online-machine-learning-simplified-2/
- Felipe Almeida: Online Machine Learning: Introduction, overview and examples, 2016, https://de.slideshare.net/queirozfcom/online-machine-learning-introduction-and-examples
- “Out-of-core algorithm”, 20.11.2017, https://en.wikipedia.org/w/index.php?title=Out-of-core_algorithm&oldid=790467914
- “Online machine learning”, 23.10.2017, https://en.wikipedia.org/w/index.php?title=Online_machine_learning&oldid=795662704
- “Winnow (algorithm)”, 23.10.2017, https://en.wikipedia.org/w/index.php?title=Winnow_(algorithm)&oldid=796943968
- Pavel Laskov, Blaine Nelson: Online and Incremental Learning, Advanced Topics in Machine Learning, 2012, http://www.ra.cs.uni-tuebingen.de/lehre/ss12/advanced_ml/lecture8.pdf
- Vitor R. Carvalho, William W. Cohen: Notes on Single-Pass Online Learning Algorithms, 2006, http://www.cs.cmu.edu/~vitor/papers/onlinetechreport.pdf
- [2] Ron Bekkerman: Automatic Categorization of Email into Folders: Benchmark Experiments on Enron and SRI Corpora, 2004, http://scholarworks.umass.edu/cgi/viewcontent.cgi?article=1217&context=cs_faculty_pubs

Quellen

- Burlutskiy, Petridis, Fish, Chernov, Ali: An Investigation on Online versus Batch Learning in Predicting User Behaviour, 2016, http://bura.brunel.ac.uk/bitstream/2438/15408/1/Fulltext.pdf
- “Catastrophic interference”, 25.11.2017, https://en.wikipedia.org/w/index.php?title=Catastrophic_interference&oldid=796150734
- Alexander Gepperth, Barbara Hammer: Incremental learning algorithms and applications, 2016, https://www.elen.ucl.ac.be/Proceedings/esann/esannpdf/es2016-19.pdf
- Parth Shah, Riju Pahwa: Lecture 1 Notes, Urban Data Analytics and Machine Learning, 2017, http://web.mit.edu/6.S097/www/resources/L01.pdf
- Victor Lavrenko: IR20.3 Passive-aggressive algorithm (PA), 2015, https://www.youtube.com/watch?v=uxGDwyPWNkU
- “Enabling Continual Learning in Neural Networks”, 29.11.2017, https://deepmind.com/blog/enabling-continual-learning-in-neural-networks/
- “Perzeptron”, 09.12.2017, https://de.wikipedia.org/w/index.php?title=Perzeptron&oldid=165056401
- “Gradientenverfahren”, 09.12.2017, https://de.wikipedia.org/w/index.php?title=Gradientenverfahren&oldid=158180650

Quellen

- Saffari, Leistner, Santner, Godec, Bischof: On-line Random Forests, 2009, http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.150.1671&rep=rep1&type=pdf
- “Random Forest”, 08.12.2017, https://en.wikipedia.org/w/index.php?title=Random_forest&oldid=810361658
- [1] Julian M. Kunkel: Machine Learning, Lecture BigData Analytics, 2017, https://wr.informatik.uni-hamburg.de/_media/teaching/wintersemester_2017_2018/bd1718-5-machine-learning.pdf