„Grundwassersysteme und Numerik“

Veranstaltung im Modul Hydrosystemanalyse

- Vorlesung: Datenbasierte Methoden

Prof. Dr. Olaf Kolditz
Dr. Erik Nixdorf

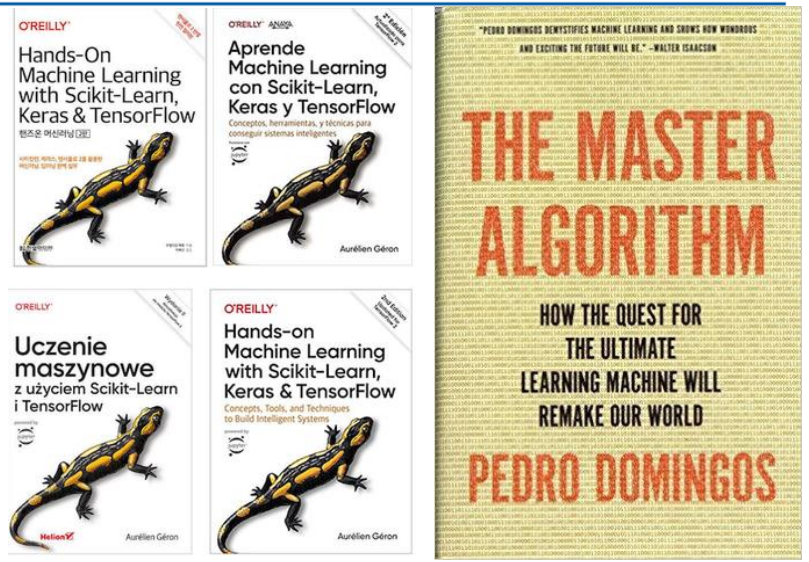
09.07.2021

Einleitung

- Die Vorlesung soll Ihnen einen ganz groben Einblick in datenbasierte Methoden („Machine Learning“) geben

- 2 Teile: 1) Einführung in die Methodik
  2) Anwendung in der Hydrogeologie am konkreten Beispiel

  - Viele Teile dieser Vorlesung sind inspiriert von dem Buch : *Hands-on Machine Learning with Scikit-Learn, Keras, and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems* von Aurelien Geron
  - Für ein tieferes (sinn-stiftendes) Verständnis der Konzepte empfiehlt sich auch z.B: *„The Master Algorithm*“ von Pedro Domingos



https://tinyurl.com/2x2tn2rk

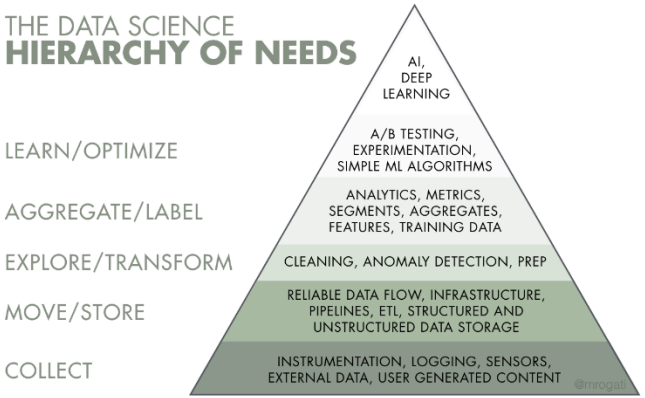
Einführung: ~~~Data Science, Big Data, AI, Deep Learning, …~~~

https://ouzor.github.io/blog/2015/04/29/tty-datascience-lecture.html

Einführung Data Science & Machine Learning

- Was ist denn jetzt Data Science?
- Data Science ist ein interdisziplinäres Wissenschaftsfeld, welches wissenschaftlich fundierte Methoden, Prozesse, Algorithmen und Systeme zur Extraktion von Erkenntnissen, Mustern und Schlüssen aus Daten ermöglicht [wiki]

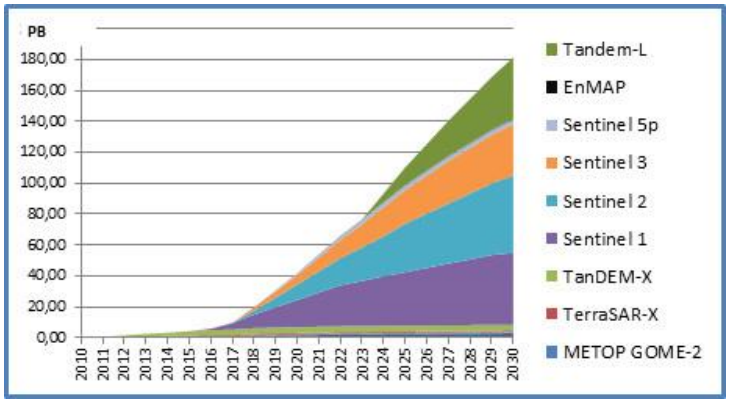
- "Data Science is statistics on a Mac" -Big Data Borat on Twitter
- "Big data is like teenage sex: everyone talks about it, nobody really knows how to do it, everyone thinks everyone else is doing it, so everyone claims they are doing it…" -Dan Ariely on Facebook



https://tinyurl.com/3tjx3hrw

Einführung Big Data

- Big Data bezeichnet Datenmengen, welche beispielsweise zu groß, zu komplex, zu schnelllebig oder zu schwach strukturiert sind, um sie mit manuellen und herkömmlichen Methoden der Datenverarbeitung auszuwerten [wiki]

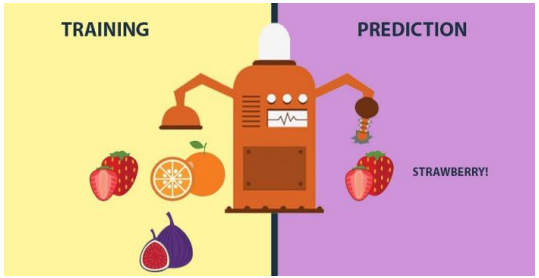


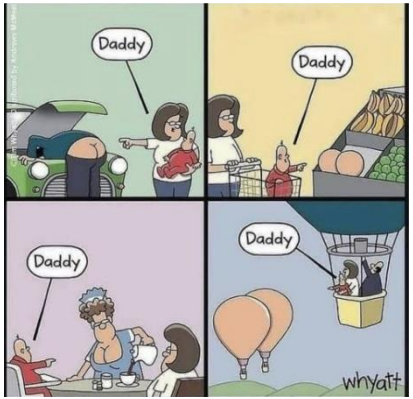
Auflaufende Satellitendaten am DLR [https://tinyurl.com/546rmf2s]

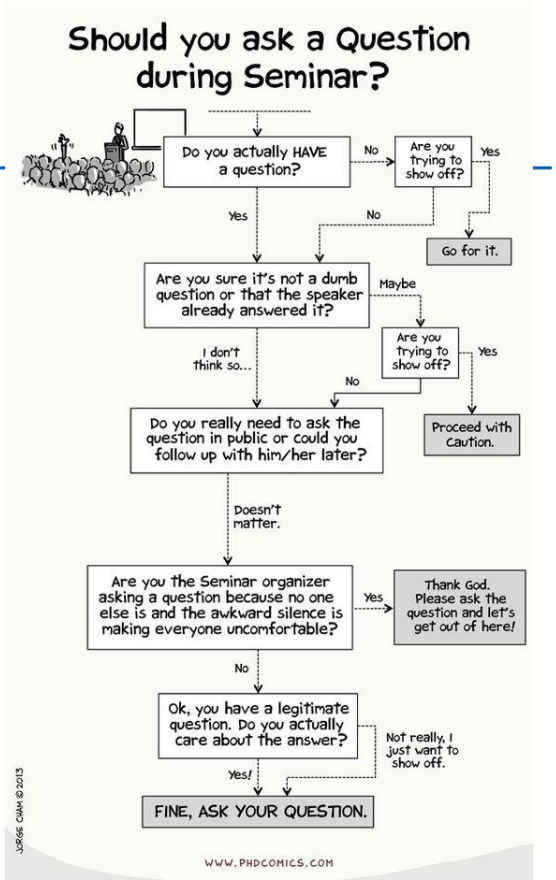
Einführung Maschinelles Lernen

- Maschinelles Lernen ist ein Oberbegriff für die „künstliche“ Generierung von Wissen aus Erfahrung: Ein künstliches System lernt aus Beispielen ohne explizite Anweisungen und kann diese nach Beendigung der Lernphase verallgemeinern [Wiki]

- „Ein Computerprogramm lernt aus Erfahrung E in Bezug auf eine Aufgabe T und ein Leistungsmaß P, wenn sich seine Leistung bei T, gemessen an P, mit der Erfahrung E verbessert." (Mitchel 1998)







Kein Maschinelles Lernen

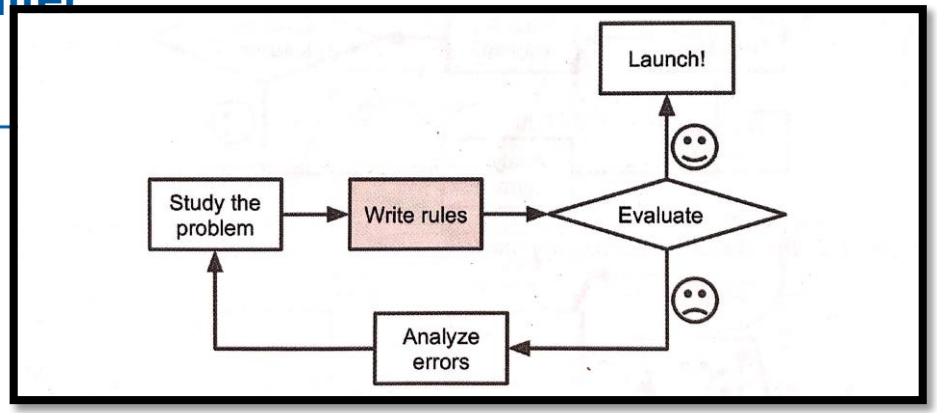
Einleitung Maschinelles Lernen: Warum

- Beispiel Spam Filter:
- Ein Spam Filter **klassifiziert** Emails in 2 Kategorien: Spam/ nicht Spam
- Typischweise hat ein Spam Filter 4 Ansätze zur Kontrolle

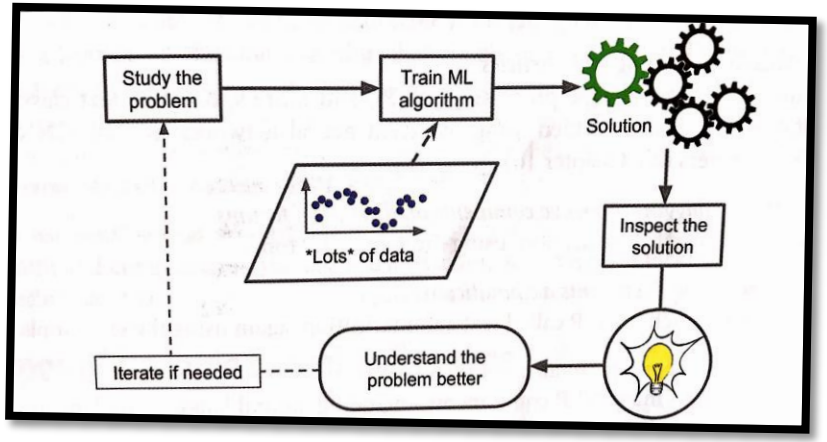
1. Überprüfung des Absenders anhand seiner E-Mail-Adresse oder URL
2. Kontrolle der Server, die den Inhalt versenden, weiterleiten oder zur V
3. Aussortieren nach dem Header
4. Aussortieren anhand des Textes (Contentfilter)

Einleitung Maschinelles Lernen: Spamfilter

- Ein „klassisch“ programmierter Spamfilter würde nach Strukturen in den Emails suchen und explizite Regeln aufstellen.
- Ergebnisse der Klassifierung werden evaluiert und die Regellliste angepasst bis die gewünschte Präzision erreicht ist
- Es entsteht eine lange Liste von Regeln die schwer zu warten ist
- Spammer ändern vlt die Formulierungen um dem Spamfilter zu entgehen („For you“ statt „4you“)
- Ein (Überwachter) ML Ansatz lernt aus bekannten Daten automatisch welche Muster gut für das Klassifizieren von Emails sind ->Kurz, wartungsarm, akkurater



Traditioneller Ansatz



ML Ansatz

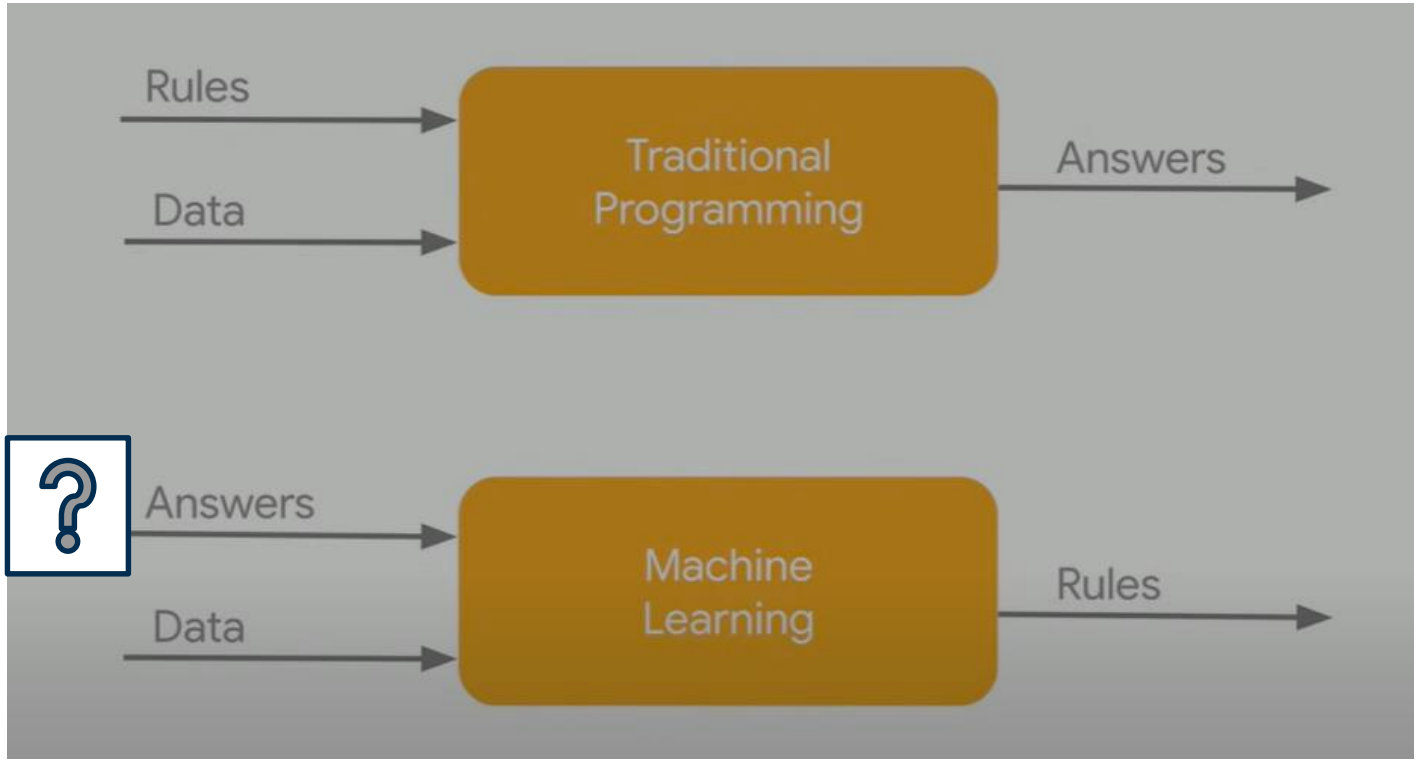
Einführung Maschinelles Lernen

- „Ein Computerprogramm lernt aus Erfahrung E in Bezug auf eine Aufgabe T und ein Leistungsmaß P, wenn sich seine Leistung bei T, gemessen an P, mit der Erfahrung E verbessert." (Mitchel 1998)

- Was bedeutet die Definition angewandt auf das Email Klassifizierungsproblem?

- E-Mails als Spam oder nicht als Spam zu klassifizieren.
- Beobachtet, wie Sie E-Mails als Spam oder nicht als Spam kennzeichnen.
- Die Anzahl (oder der Anteil) der E-Mails, die korrekt als Spam/Nicht-Spam klassifiziert wurden.

Einleitung Maschinelles Lernen



https://tinyurl.com/ejdfx8xd

Einführung Maschinelles Lernen

- Maschinelles Lernen könnte sinnvoll sein bei:

1. Problemen, die viel fine-tuning und eine lange Liste von Regeln erfordern
2. Sich schnell verändernde Probleme, bei denen die Anpassungsfähigkeit generischer Algorithmen vorteilhaft sein kann
3. Komplexe nicht-determinische Probleme für die traditionelle(re) Lsöungen unbefriedigende Ergebnisse bringen
4. Probleme die mit traditionelle(re)n Lösungen extreme Rechenressourcen benötigen

Kategorien Maschinellen Lernen Algorithmen

- ML Algorithmen werden üblicherweise nach dem Maß der menschlichen Überwachung („human supervision“ eingeteilt und dann weiter in die Anwendungsfälle:
- überwachtes Lernen (Supervised Learning)
- unüberwachtes Lernen (Unsupervised Learning)
- teilüberwachtes Lernen (Semi-Supervised Learning)
- verstärkendes Lernen (Reinforcement Learning)
- Kategorisierung nach anderen Kriterien ist möglich wie z.B.:

1. Instanz und Modellbasierte Lerner
2. Batch und Online Lerner

Reinforcement Learning (Deep Q playing Atari Breakout https://tinyurl.com/a3s5zzke

Überwachtes Lernen

- Beim Überwachten Lernen enthält der Datensatz Informationen zur Zielvariablen, er ist „gelabelt“:
- Jede Instanz wird durch die Werte defniert, die vorgegebene Features und die Zielvariable annehmen
- Zwei Grundaufgaben ergeben sich für überwachtes Lernen:

1. **Klassifikation**: Zu jeder Instanz gehört eine Klasse aus einer endlichen Menge; fnde für neue Instanzen die richtige Klasse. •
2. **Regression**: Zu jeder Instanz gehört eine Zahl; fnde für neue Instanzen die (ungefähr) richtige Zahl.

Kovariablen (Features) Zielvariable („Label“)

Instanz

```
# Load datasets
train = pd.read_csv('https://raw.githubusercontent.com/selva86/datasets/master/BostonHousing.csv')
train
```

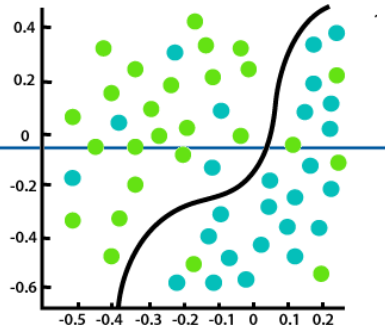
	crim	zn	indus	chas	nox	rm	age	dis	rad	tax	ptratio	b	lstat	medv
0	0.00632	18.0	2.31	0	0.538	6.575	65.2	4.0900	1	296	15.3	396.90	4.98	24.0
1	0.02731	0.0	7.07	0	0.469	6.421	78.9	4.9671	2	242	17.8	396.90	9.14	21.6
2	0.02729	0.0	7.07	0	0.469	7.185	61.1	4.9671	2	242	17.8	392.83	4.03	34.7
3	0.03237	0.0	2.18	0	0.458	6.998	45.8	6.0622	3	222	18.7	394.63	2.94	33.4
4	0.06905	0.0	2.18	0	0.458	7.147	54.2	6.0622	3	222	18.7	396.90	5.33	36.2
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
501	0.06263	0.0	11.93	0	0.573	6.593	69.1	2.4786	1	273	21.0	391.99	9.67	22.4
502	0.04527	0.0	11.93	0	0.573	6.120	76.7	2.2875	1	273	21.0	396.90	9.08	20.6
503	0.06076	0.0	11.93	0	0.573	6.976	91.0	2.1675	1	273	21.0	396.90	5.64	23.9
504	0.10959	0.0	11.93	0	0.573	6.794	89.3	2.3889	1	273	21.0	393.45	6.48	22.0
505	0.04741	0.0	11.93	0	0.573	6.030	80.8	2.5050	1	273	21.0	396.90	7.88	11.9

506 rows × 14 columns

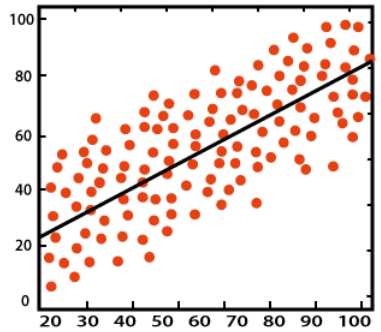
Boston House Datensatz

Überwachtes Lernen: Klassifikation Regression

- Gängige ML Algorithmen für überwachten Lernen sind:
  - k nearest neighbor
  - Linear Regression
  - Logistische Regression
  - Support Vector Machines
  - Random Forest und Decision Trees
  - Neurale Netze



Classification



Regression

https://tinyurl.com/cusxxhec

Unüberwachtes Lernen

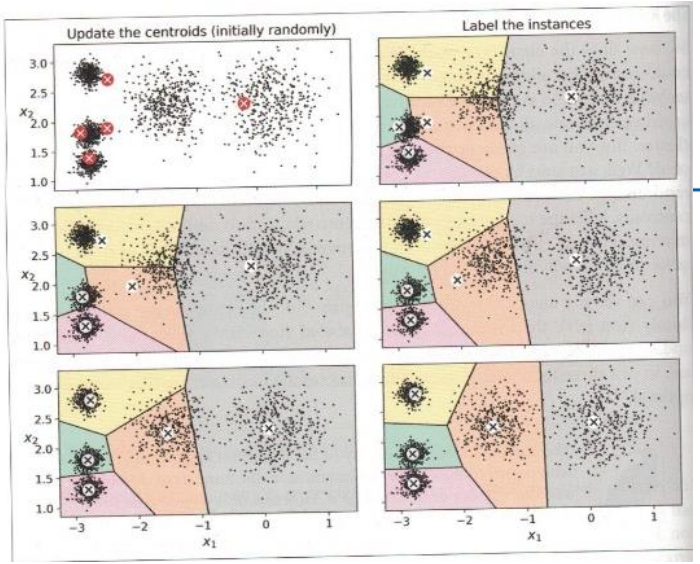
- Unüberwachtes Lernen bedeutet, dass die Daten kein Label (Information über die Zielvariable) enthalten. Der Fokus liegt auf Datenexploration und Mustererkennung
- Ergebnis des unüberwachten Lernens ist z.B Zuschreibung eines Gleichheitsmaßes (z.B. Index für jede Instanz)
- Beispielalgorithmen sind:
- Clustering (z.B K-Means,Hierarchical Clustering Analysis)
- Anomaly detection (Isolation Forest, One-class SVM Algorithm)
- Dimensionalitätsreduzierung (Principal Component Analysis, Local Linear Embedding)

Unüberwachtes Lernen: K-Mean Clustering

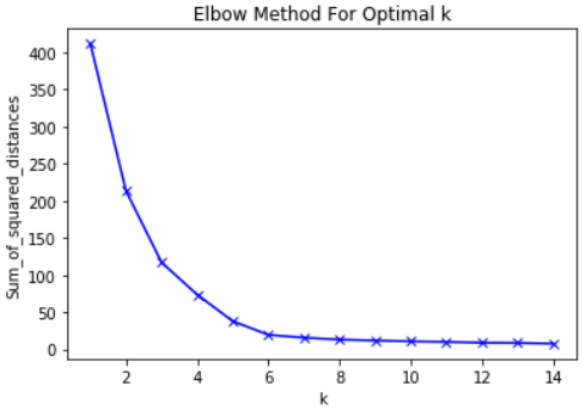
- Ziel von *k*-Means ist es, den Datensatz so in *k* Partitionen zu teilen, dass die Summe der quadrierten Abweichungen von den Cluster-Schwerpunkten minimal ist [wiki]

- 3 Arbeitsschritte

1. **Initialisierung**: Wähle k zufällige Mittelwerte
2. **Zuordnung:** Jedes Datenobjekt wird dem nächstgelegenene Knotenschwerpunkt zugeordnet
3. **Aktualisieren:** Berechne die Mittelpunkte der Cluster neu
4. **Auswertung** des Iterationsergebnisses

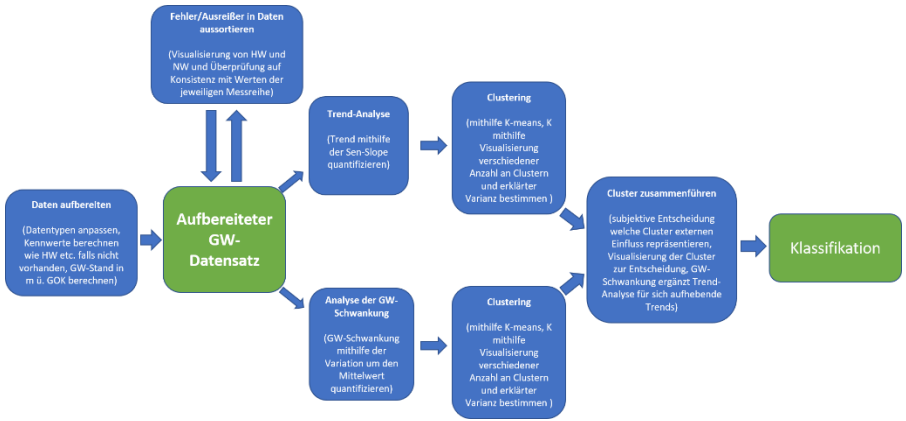


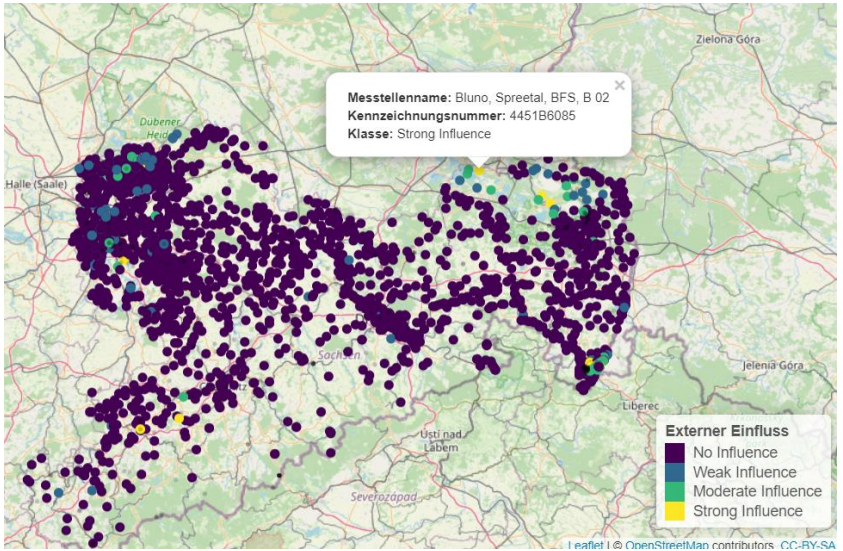
K-Mean Clustering (Geron, 2020)



Unüberwachtes Lernen: Beispiel aus der hydrogeologischen Praxis

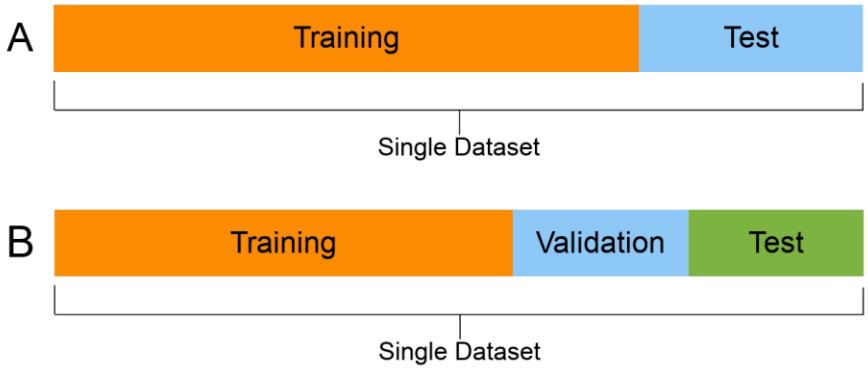
- Clustering von GWMs nach Einflussgrößen (J.Schalla) mit k-mean Clustering Verfahren

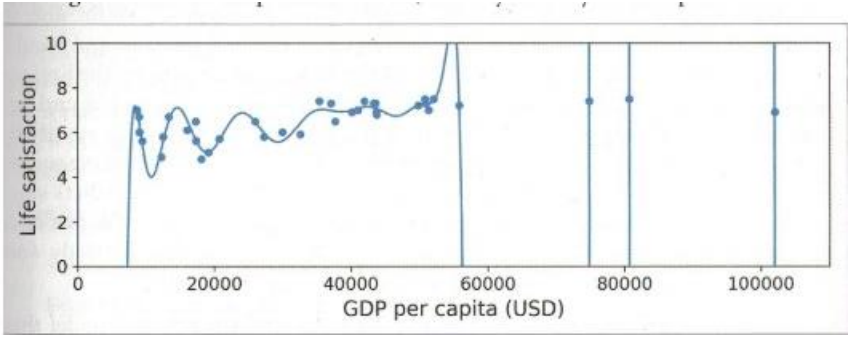




Maschinelles Lernen: Overfitting

- Überanpassung (Overfitting) bedeutet, dass der Lerner zu Empfindlich auf kleinere Schwankungen in den Daten reagiert (high variance problem)
- Aufteilen der Daten in Trainings und Testdaten ermöglicht eine Quantifizerung des Problems

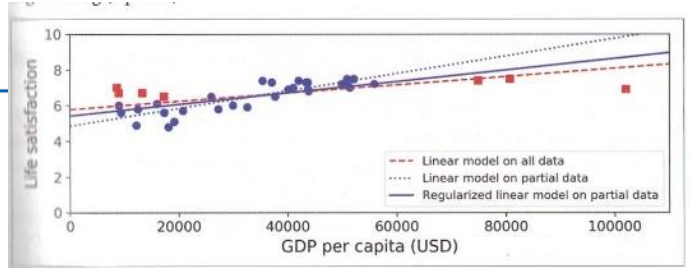




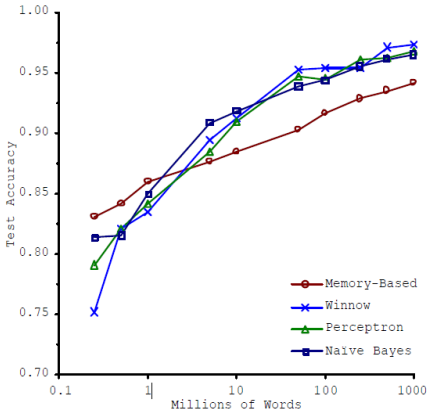
Überanpassung durch Polynomfit (Geron, 2020)

Maschinelles Lernen: Overfitting

- Overfitting kann folgende **Ursachen** haben:
- Der Algorithmus ist zu komplex für das Problem
- Es gibt zuviele Features bzw zu wenige Daten-Instanzen
- Die Daten sind fehlerbehaftet (Rauschen, Ausreißer)
- Lösungen:
  - Mehr Daten/Erhöhung der Datenqualität (outlier Detection..)
  - Dimensionsreduzierung (PCA, …)
  - Simplifzierung/Regularisierung des verwendeten Lerners



Reguliarisierung einer linearen Regression (Geron, 2020)

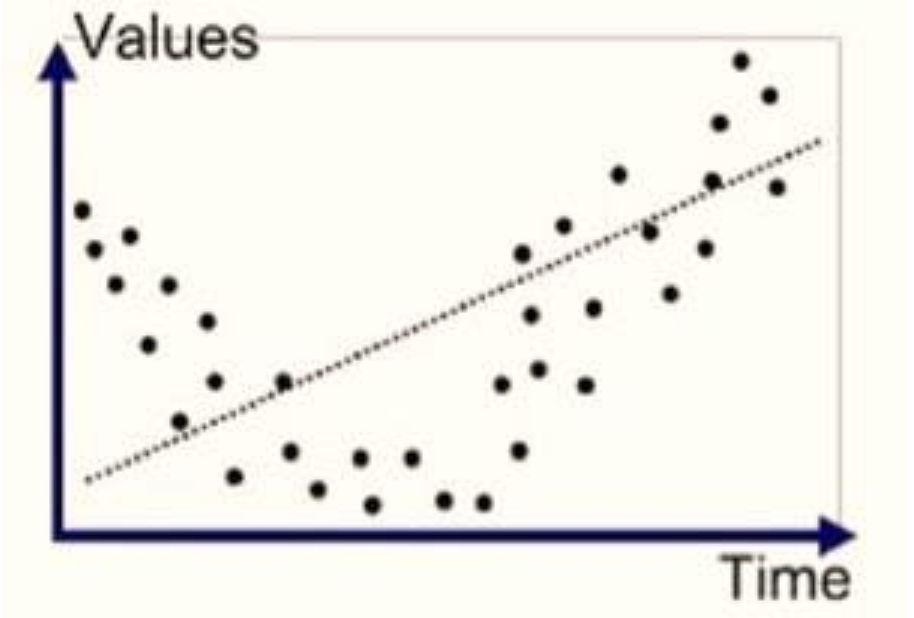


Google's Research Director Peter Norvig : "We don't have better algorithms. We just have more data." (Bank and Brill (2001)

Maschinelles Lernen: Unteranpassung (underfitting)

- Unfähigkeit des Modells die zugrunde liegende Struktur der Daten abzubilden und Informationen aus den Daten abzuleiten (high-bias Problem)
- Ursachen:

1. Zu einfaches Modell gewählt (z.B. lineares Modell)
2. Zu wenige /nicht relevante Features ausgewählt



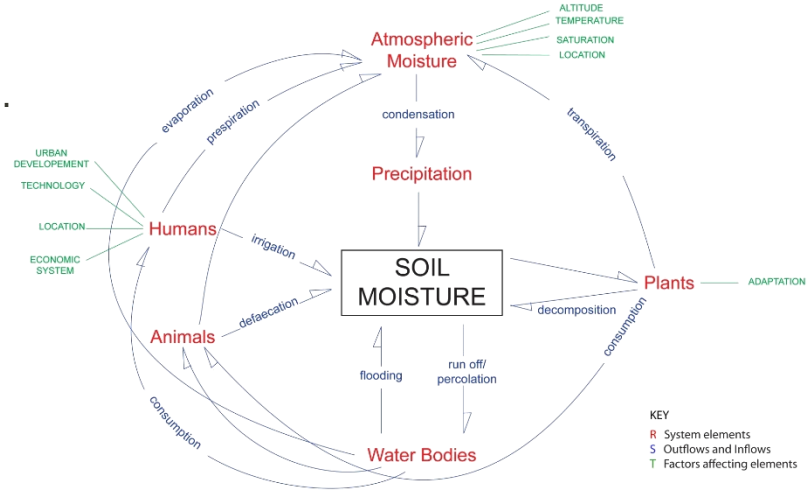
End to End Machine Learning Projekt

- Ein typischer Ablauf eines Projekts bei dem ML Algorithmen verwendet werden könnte so aussehen:

1. Einen konzeptionellen Überblick verschaffen („Das große Ganze überblicken“)
2. Datenaquisation
3. Exploration und Visualisierung der Daten
4. Aufbereitung der Daten für den ausgewählen/die ausgewählten ML Algorithmen
5. Auswahl und Trainieren der Lerner
6. Hyper-parameter Optimierung
7. Ergebnisvisualisierung und Evaluierung
8. Dokumentation und „Wartung“

Beispielprojekt: Bodenfeuchte Vorhersage im Müglitz Einzugsgebiet

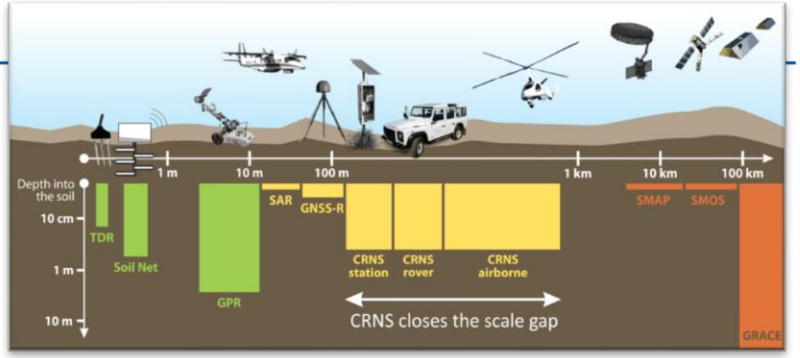
- Bodenfeuchte = kritische Zustandsgröße
- Die Kenntnis der räumlich-zeitlichen Verteilung der Bodenfeuchte ist für verschiedene Modelle wichtig, z. B. für ereignisbasierte hydrologische Modelle und Klimamodelle
- Die Berechnung/Messung/Vorhersage von zeitlich und räumlich hoch (<1km²) Bodenfeuchte ist komplex und hängt von einer Mehrzahl statischer und dynamischer Faktoren ab → Grundvorlesung Bodenkunde



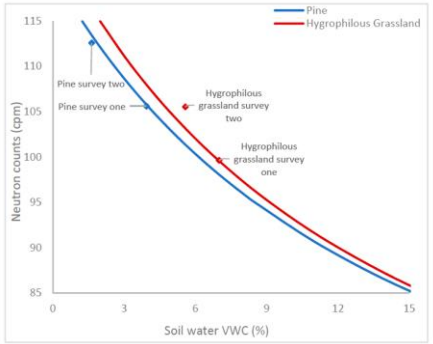
Koedel, UFZ

Beispielprojekt: Bodenfeuchte --Messverfahren

- Jedes Messverfahren der Bodenfeuchte denkt ein bestimmtes integrales Messvolumen ab und hat ein differenziertes Messintervall
- Neutronen basierten Beobachtungen (Cosmic Ray Neutron Sensing, CRNS) ermöglichen ein portables, nicht invasives Messen der Bodenfeuchte auf der Mesoskala



Skalen für Bodenfeuchtemessverfahren



Valther et al, 2019

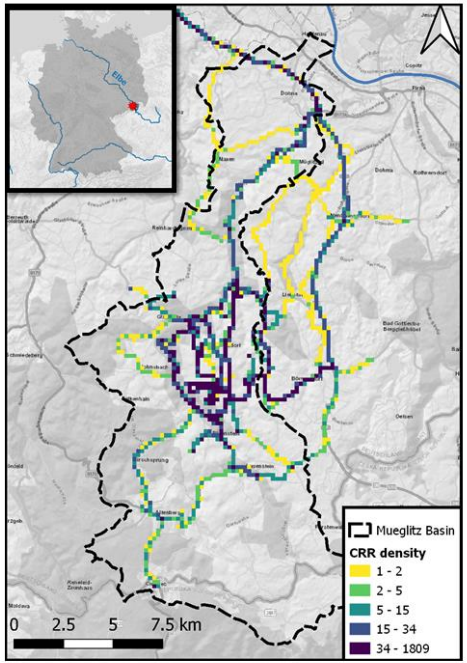
Beispielprojekt: Bodenfeuchte --Mueglitz

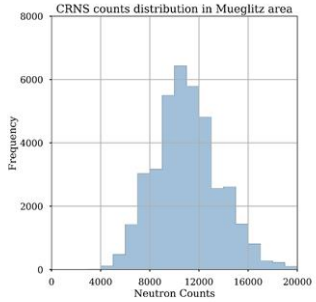
- Müglitzeinzugsgebiet als Testregion (210km² südöstlich von Dresden)

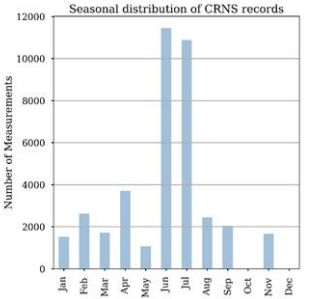
- ~30 Fahren mit dem CRNS Rover in 2 Jahren mit 40 000 Messungen (Instanzen)

- Aufgabe für den datenbasierten Ansatz: Kann auf Basis des verfügbaren Datensatzes mit Hilfe eines überwachten Lerners die Bodenfeuchte in den **nichtbefahrenen** Gebieten zu den **nichtbefahrenen Zeiten plausibel** berechnet werden?

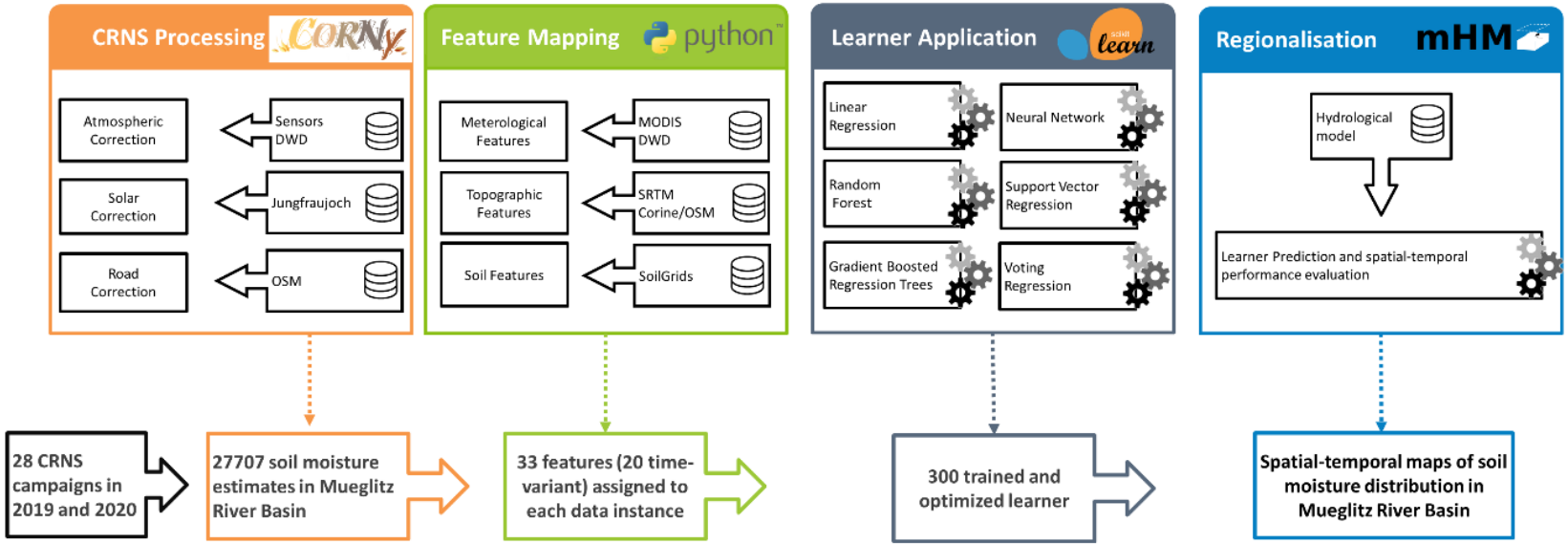
- Produkt: Räumlic-zeitliche Karten der Bodenfeuchte im Müglitztal in einer Auflösung von 200m (~Messvolumen CRNS)







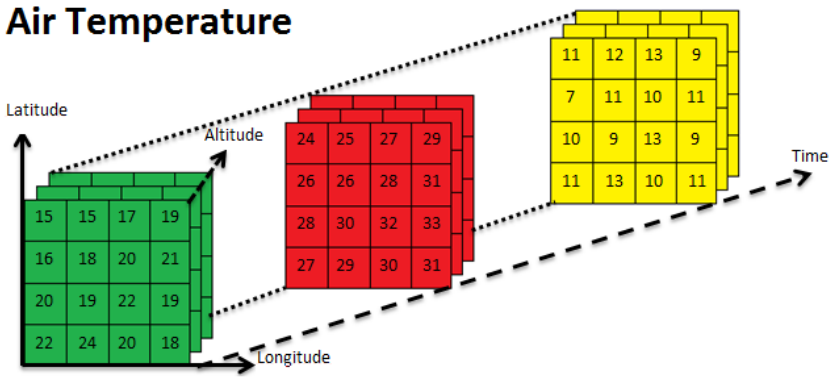
Beispielprojekt Bodenfeuchte: -- Ansatz



Warum nicht gleich ein Hydrologisches Modell nehmen?

Beispielprojekt Bodenfeuchte: -- Kovariablen

- Datenprozessierung aus öffentlichen Datenbanken (SRTM DEM, DWD open Data, SoilGrids, MODIS, OpenStreetMap)
- Datenkonvertierung auf ein 200x200m Gitter des Müglitz Einzugsgebiets und das Format NetCDF
- Bereinigung, Skalierung und Umwandlung kategorischer in kontinuierliche Features



Time-independent features

Topographic features
- Elevation*
- Aspect
- Slope
- Topographic wetness index
- Waterways**
- Water bodies**
- Landcover*
- Road type

Soil features
- Silt content
- Sand content
- Clay content*
- Soil organic carbon**
- Soil bulk density**

Time-dependent features

Meteorological features
- Air temperature**
- Humidity**
- Air pressure**
- Sum of rain (1*,7,14 d)
- No. of rain events (7,14 d)

Land Surface features
- Surface reflectance (4 bands)
- Vegetation Index (NDVI,EVI)*
- Land surface temperature (day,night)
- Leaf area index

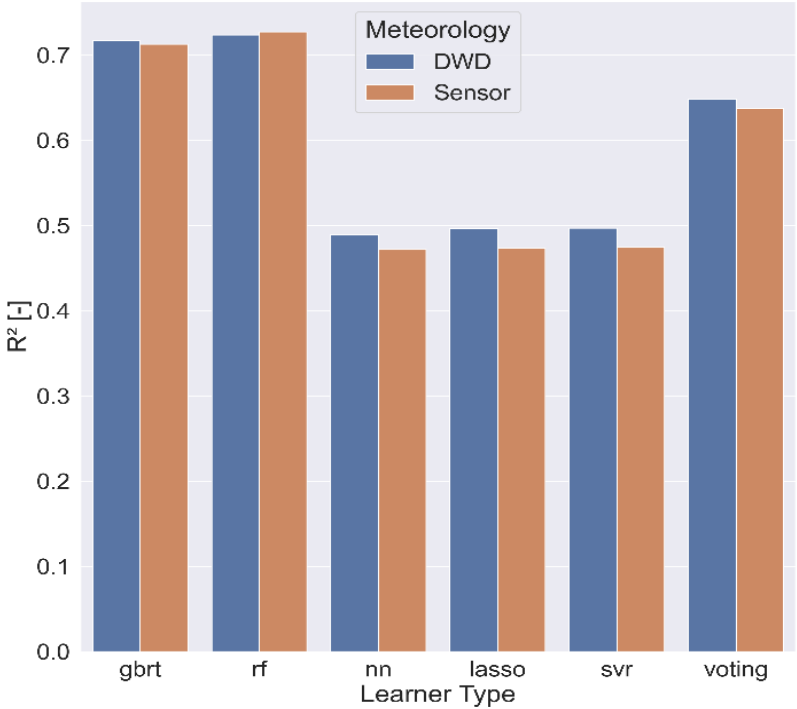
Other features
- Day of the year
- Incoming neutron radiation**

Beispielprojekt Bodenfeuchte: Trainieren der Algorithmen

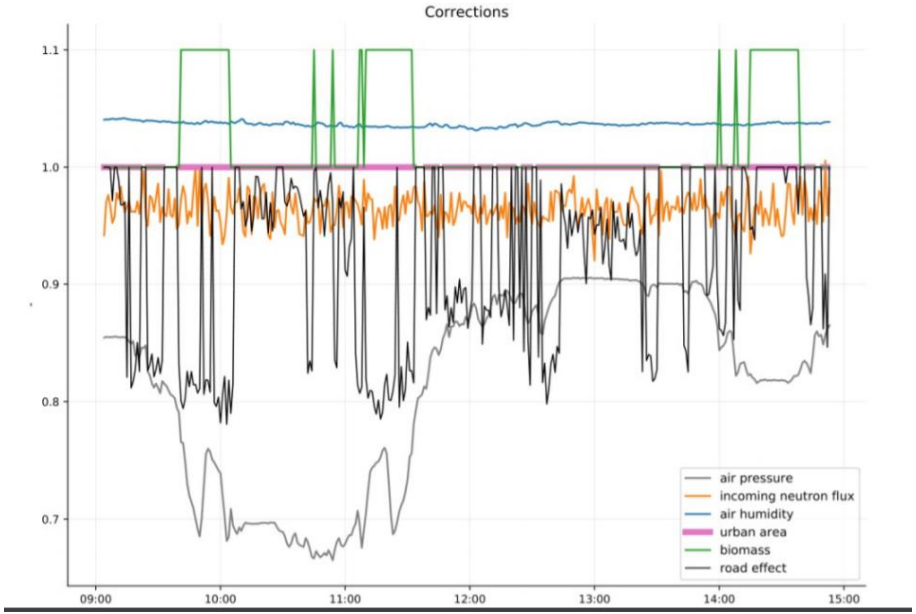
- 5 verschiedene Regression Lerner auf die CRNS Daten getestet:

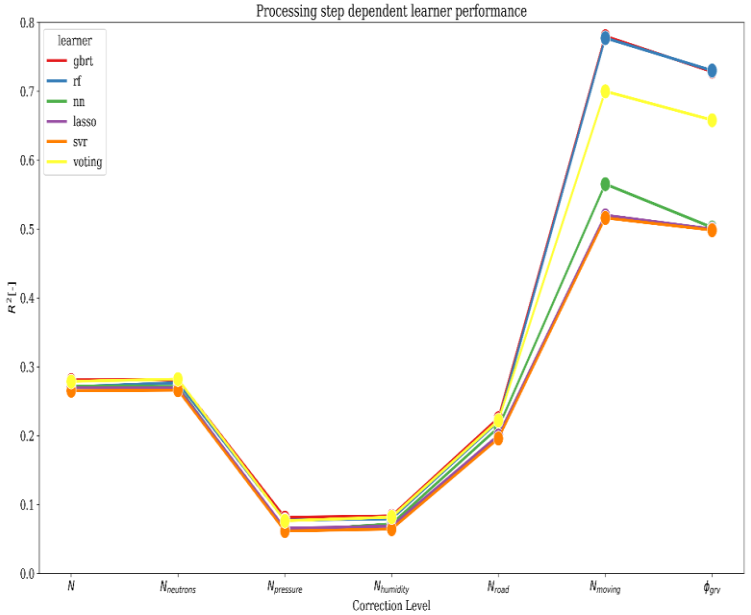
1. Gradient Boosted Regression Trees (gbrt),
2. random forest (rf),
3. artifical neural networks (nn),
4. lasso linear regression (lasso)
5. Support-Vector Regression
6. Voting Regressor

- Hyper-parameter Optimierung über systematische Kombinierung (GridSearch)



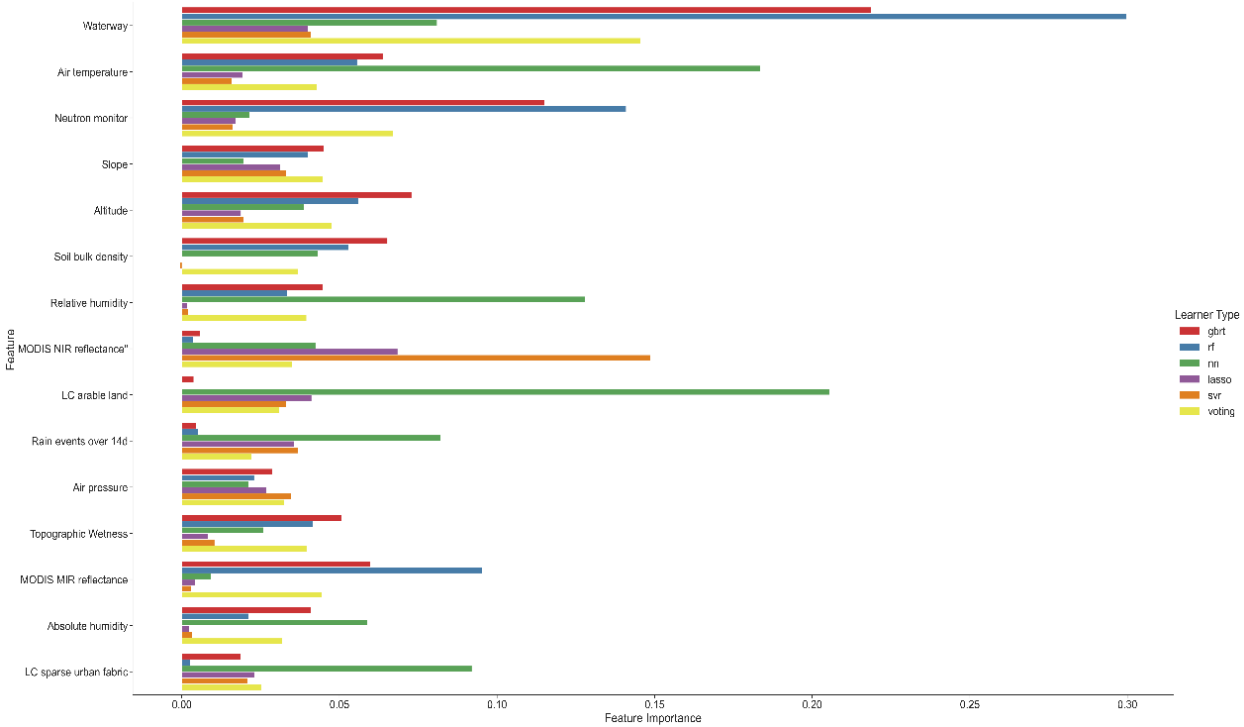
Beispielprojekt Bodenfeuchte: CRNS Datenkorrektur



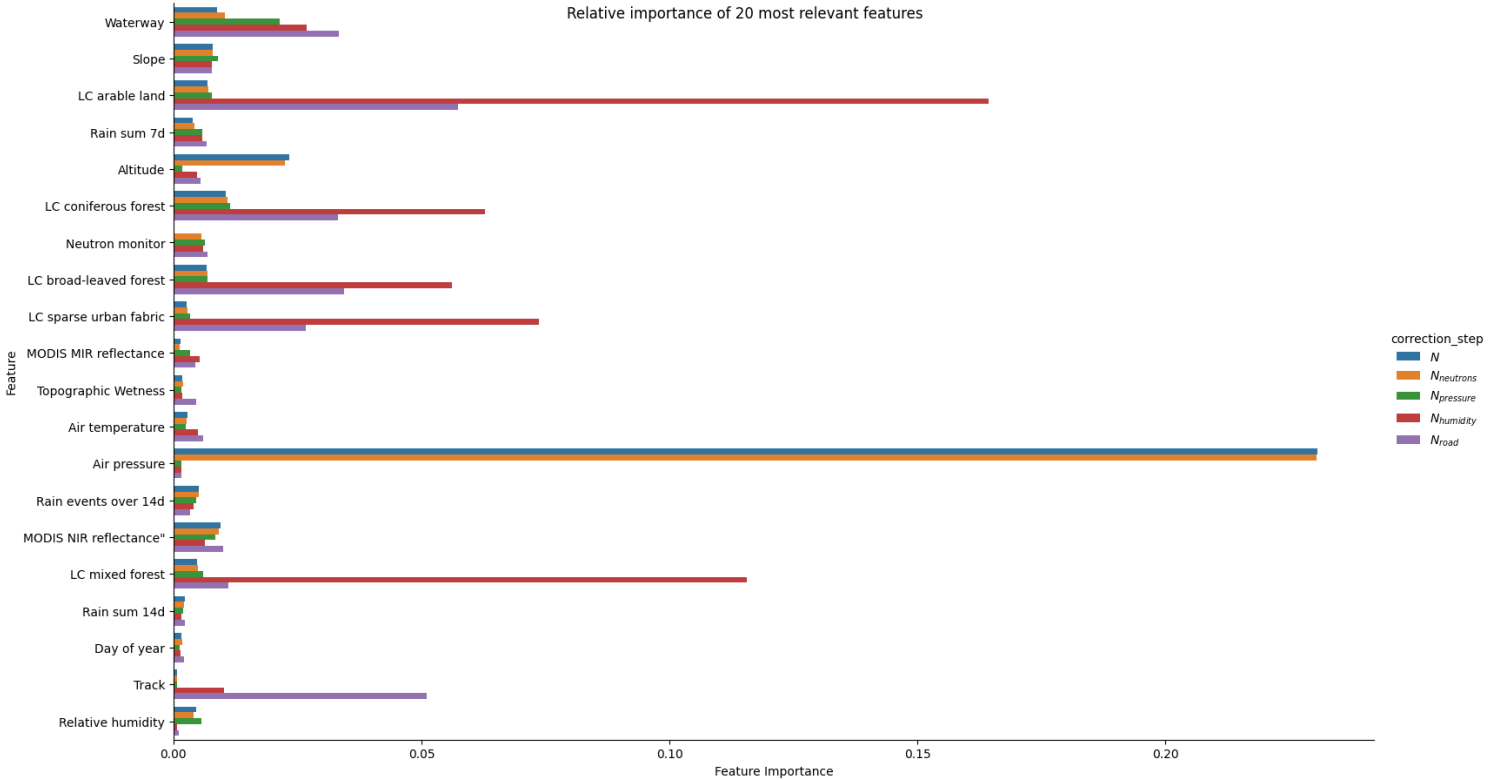


Beispielprojekt Bodenfeuchte: Feature Importance

- Der Wert für die **Feature Importance** beschreibt, bei zufälliger Verteilung des Faktors, wie er sich auf die Gesamtleistung des Modells auswirkt.
- Wenn der Wert nahe Null liegt, bedeutet dies, dass der Faktor fast keinen Einfluss auf das Modell hat.
- Je höher der Wert, desto wichtiger ist der Faktor.



Beispielprojekt Bodenfeuchte: Feature Importance nach Korrekturschritten



Beispielprojekt Bodenfeuchte: Karten