**CAT**

Beim Machine Learning gibt es auf der Suche nach den möglichst besten Modellen verschiedene Stellen an den man ansetzen kann. Neben *Feature Engineering* stellt *Hyperparameteroptimierung* eine weitere Möglichkeit dar, die genutzten Modelle zu verbessern.

**Data Science Kernaussage:**

Mithilfe von Verfahren wie *Hyperband* können ML-Modelle hinsichtlich ihrer Hyperparameter (und somit insgesamt) effizient und automatisiert optimiert werden.

**(P) roblem):**

Bei der Modellierung mit Machine Learning im Allgemeinen und neuronalen Netzen im Speziellen gibt es viele unterschiedliche Parameter bzw. Hyperparameter zu wählen, die Einfluss auf Performance, Konvergenzverhalten und erzielten Ergebnissen haben können. So muss man bei neuronalen Netzen z.B. die Anzahl an Hidden Layern und die Anzahl an Neuronen pro Layer mitsamt Aktivierungsfunktion bestimmen. Außerdem gilt es Algorithmen für die Gewichteinitialisierung und Berechnung der Gradienten auszuwählen. Bei einigen Hyperparametern haben sich bestimmte Optionen bewährt, wie z.B. *ReLu* als Aktivierungsfunktionen für tiefe Netze, so dass man auf Erfahrungen und Empfehlungen aufbauen kann. Allerdings hängt es immer von den vorliegenden Daten und der jeweiligen Problemstellung ab, welche Parameter sich (am Besten) eignen. Eine „händische“ Suche bzw. Hyperparameteroptimierung durch Ausprobieren verschiedener Optionen ist aufgrund der vielen Freiheitsgrade aufwändig und ineffizient.

**(I) ntervention:**

Es gibt verschiedene Ansätze und Verfahren mit denen die Hyperparameteroptimierung algorithmisch und somit automatisiert gelöst werden kann. Eine gute Wahl ist z.B. der Hyperband-Algorithmus, den das Python-Paket *KerasTuner* für Tensorflow zur Verfügung stellt.

**(K) ontrollintervention**

KerasTuner stellt noch weitere Algorithmen bzw. Klassen zur Hyperparameteroptimierung bereit, nämlich *RandomSearch*, *BayesianOptimization* und *SklearnTuner*. Letzterer kann für Modelle von Scikit-Learn verwendet werden. Scikit-Learn wiederum stellt die Klassen *GridSearchCV*, *RandomizedSearchCV*, *HalvingGridSearchCV* und *HalvingRandomSearchCV* (auf dessen Ansatz Hyperband aufsetzt) zur Verfügung, welche auch bei der Modellierung mit Tensorflow genutzt werden können.

**(E) rgebnismaß (Zielgröße(n)) – Die Evidence**

Durch die Hyperparameteroptimierung soll eine optimale Auswahl an Parametern für die Modelle gefunden werden. Es werden am Ende diejenigen Parameter gewählt mit denen beim Training, die besten Werte der Validierungsmetrik (*Accuracy* bei Klassifizierung und *MSE* bzw. *MAE* bei Regression) erzielt wurden.

**Anmerkungen:**

Für Hyperparameteroptimierung sind die englischen Begriffe *Hyperparameter Optimization* und (*Hyper-)Parameter Tuning* weit verbreitet.

**Literaturhinweise:**

* *Getting started with KerasTuner (Developer Guide)* <https://keras.io/guides/keras_tuner/getting_started/>
* *Visualize the hyperparameter tuning process (Developer Guide)* <https://keras.io/guides/keras_tuner/visualize_tuning/>
* *KerasTuner API*  
   <https://keras.io/api/keras_tuner/>
* *Hyperband: A Novel Bandit-Based Approach to Hyperparameter Optimization (Paper)* <https://jmlr.org/papers/v18/16-558.html>
* *Tuning the hyper-parameters of an estimator (User Guide)* <https://scikit-learn.org/stable/modules/grid_search.html>

**Die Suche nach der besten Evidenz**

**1. Problem**

**2. Definition einer wichtigen suchbaren Frage**

**3. Auswahl der wahrscheinlichsten Quelle für diese Evidenz**

**4. Erstellung einer Suchstrategie**

**5.0 Zusammenstellung der Evidenzausbeute 5.1 Falle Ausbeute schlecht**

**Auswahl der zweit-wahrscheinlichsten Quelle für diese Evidenz**

**Erstellung einer Suchstrategie**

**Zusammenfassung der Evidenz**

**Anwendung der Evidenz**

**6. Anwendung der Evidenz**