

Efficient and Robust Automated Machine Learning

Robert Feldhans

12. Juli 2018

Seminar Musterklassifikation

1. Motivation
2. Automated Machine Learning in a Nutshell
3. Meta Learning
4. Ensembles
5. Fragerunde

Motivation

Interlude: Hyperparameter I

Was sind Hyperparameter?

- Werden *vor* dem Lernen definiert
- Sind in der Regel Zahlen oder Funktionen

Interlude: Hyperparameter I

Was sind Hyperparameter?

- Werden *vor* dem Lernen definiert
- Sind in der Regel Zahlen oder Funktionen

Allgemein

Alles was in irgendeiner Art austauschbar ist in einem speziellen ML-Verfahren und während des Trainings konstant bleibt

Beispiele für Hyperparameter

- Lernrate
- Gewichte jeglicher Form
- Anzahl der Cluster in k-means clustering
- Aktivierungsfunktionen
- Anzahl der Hidden Layers in einem Netz
- Breite der Layers in einem Netz

- Ein gutes neuronales Netz zu trainieren ist schwer, braucht viel Arbeitszeit und Erfahrung
- Jeder sollte in der Lage sein NN zu trainieren (im besten Fall sogar Maschinen!)

- Ein gutes neuronales Netz zu trainieren ist schwer, braucht viel Arbeitszeit und Erfahrung
- Jeder sollte in der Lage sein NN zu trainieren (im besten Fall sogar Maschinen!)

Lösung: Ein automatisches (und effizientes) System, welches gute Hyperparameter auswählt, muss her!

Automated Machine Learning in a Nutshell

Grundlegende Idee

- Starten mit *irgendwie* ausgewählten Hyperparametern
- Classifier trainieren
- Classifier evaluieren
- Hyperparametertuning mithilfe eines Bayesian optimizer
- Wiederholung bis zu einem zufriedenstellenden Ergebnis

Auto-ML II

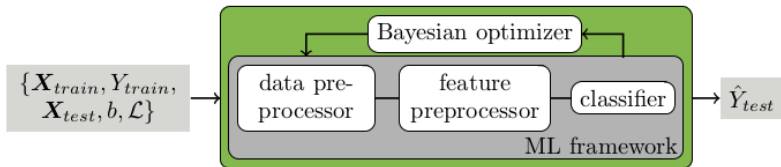


Bild zu problemen mit der initialisierung von hyperparametern

Rapidly Exploring Random Tree (RTT) I

Idee

- Werte zufällig wählen
- So oft wiederholen, bis man einen guten Überblick über den Searchspace hat

Rapidly Exploring Random Tree (RTT) I

Idee

- Werte zufällig wählen
- So oft wiederholen, bis man einen guten Überblick über den Searchspace hat

Vorteil

Bietet beliebig guten Überblick über den Searchspace

Rapidly Exploring Random Tree (RTT) I

Idee

- Werte zufällig wählen
- So oft wiederholen, bis man einen guten Überblick über den Searchspace hat

Vorteil

Bietet beliebig guten Überblick über den Searchspace

Achtung

RTT bietet einige Fallstricke. Think before use!

Rapidly Exploring Random Tree (RTT) II



- Ausgesprochen Rechenintensiv
- Unterschiedliche Lernverfahren?
- Es gibt kein “best” Lernverfahren, nur “best at”
- Manche ML-Verfahren erfordern intensive Hyperparameteroptimisierung
- Bayes optimization sollte sich jedoch um dieses Problem kümmern

Meta Learning

Idee

- Das richtige ML-Verfahren für ein bestimmtes Datenset hängt vom Datenset selbst ab
- Ein bestimmter Klassifikator sollte auf ähnlichen Datensets ähnlich gute Ergebnisse liefern

Idee

- Das richtige ML-Verfahren für ein bestimmtes Datenset hängt vom Datenset selbst ab
- Ein bestimmter Klassifikator sollte auf ähnlichen Datensets ähnlich gute Ergebnisse liefern

⇒ Also bauen wir uns einen Klassifikator, der uns anhand eines neuen Datensets sagt, welche Art von Klassifikator wir trainieren sollten

Erstellung des Meta-Klassifikators

- Für eine (große) Menge an bereits bekannten Sets:
Metafeatures berechnen
⇒ class-probability, categorical-numerical-ratio, number of classes/features/instances (missing)
- Einen Klassifikator (SMAC) auf diesen Meta-Features trainieren

Erstellung des Meta-Klassifikators

- Für eine (große) Menge an bereits bekannten Sets:
Metafeatures berechnen
⇒ class-probability, categorical-numerical-ratio, number of classes/features/instances (missing)
- Einen Klassifikator (SMAC) auf diesen Meta-Features trainieren

Auswertung

- Für ein neues Datenset werden anhand der L_1 Distanz zu den bereits bekannten Datensets Klassifikatoren ausgewählt

Ensembles

- Mehrere vielversprechende ML-Verfahren ausgewählt
- Jedes davon aufwändig mit Bayesian optimizer im Hinblick auf Hyperparameter getunt
- Das beste der Besten herausgepickt und die anderen weggeworfen

- Mehrere vielversprechende ML-Verfahren ausgewählt
- Jedes davon aufwändig mit Bayesian optimizer im Hinblick auf Hyperparameter getunt
- Das beste der Besten herausgepickt und die anderen weggeworfen

Warum eigentlich?

Idee

- Anstatt teuer optimierte Klassifikatoren wegzuwerfen, Kombination der Besten

Idee

- Anstatt teuer optimierte Klassifikatoren wegzuwerfen, Kombination der Besten

Aber wie kombinieren?

- Alle ungewichtet aufsummieren?
- Stacking?
- gradient-free numerical optimization?

Wie baut man ein Ensemble?

- Starte mit einem leeren Ensemble
- Füge den Klassifikator dem Ensemble hinzu, der das Ensemble am besten ergänzt
- Wiederhole bis alle Klassifikatoren enthalten sind oder X mal
- Durchschnitt über alle Predictions bilden für Resultat

Wie baut man ein Ensemble?

- Starte mit einem leeren Ensemble
- Füge den Klassifikator dem Ensemble hinzu, der das Ensemble am besten ergänzt
- Wiederhole bis alle Klassifikatoren enthalten sind oder X mal
- Durchschnitt über alle Predictions bilden für Resultat
- Alle Einträge sind ungewichtet
- Duplikationen sind erlaubt

Vielen Dank für eure Aufmerksamkeit!

Fragerunde



M. Feurer, A. Klein, K. Eggenberger, J. Springenberg, M. Blum, and F. Hutter, “Efficient and robust automated machine learning,” in *Advances in Neural Information Processing Systems 28* (C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, eds.), pp. 2962–2970, Curran Associates, Inc., 2015.