Insert Company Logo here

Patricia Kaufmann

Relevanztuning ist keine Magie – Es ist Machine Learning

Search/Recommendations

code.talks commerce

SHI

SEARCH **ANALYTICS** BIG DATA

Consulting • Software • Development • Training

SEARCH
ANALYTICS
BIG DATA

WAS WIR TUN

Wir realisieren Lösungen zur optimalen Nutzung von Daten.

- E-Commerce Search
- Intranet Portal Search
- Website Search
- Search Knowledge Management
- Cross Analytics
- Visualisierung von Daten

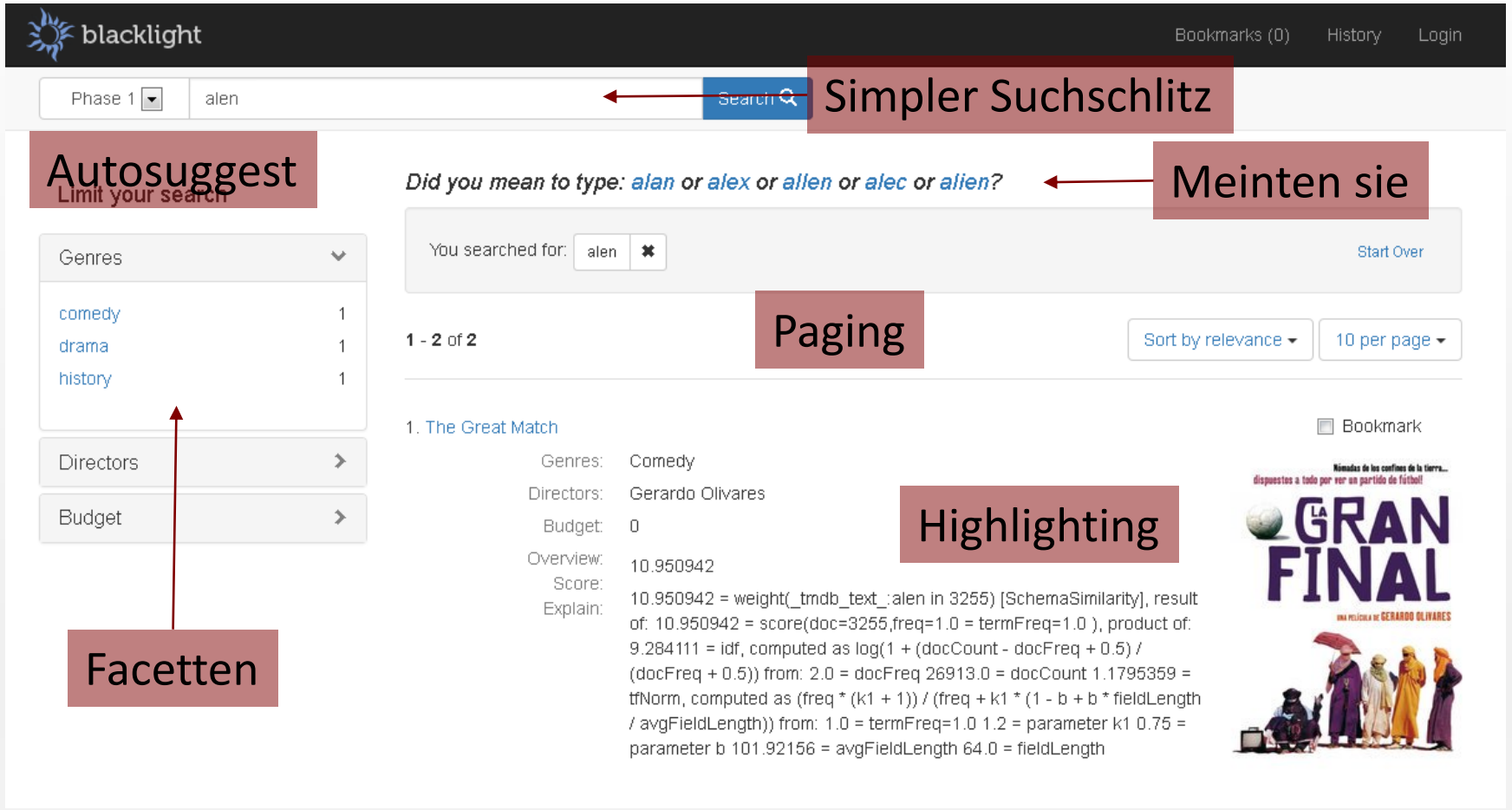
Patricia Kaufmann

- Consultant Search, Analytics & Vizualisation
- patricia.kaufmann@shi-gmbh.com

Agenda

- Praktisches Beispiel: Streaming-Plattform – Gute Usability & Miese Relevanz
- Klassische Relevanztuning-Möglichkeiten
- Learning to Rank
  - Motivation
  - Beispiele aus der Praxis
  - Umsetzung in Solr
  - Features
  - Modelle
  - Fachliche und Technische Herausforderungen
  - Besonderheiten bei Solrs LTR Plugin

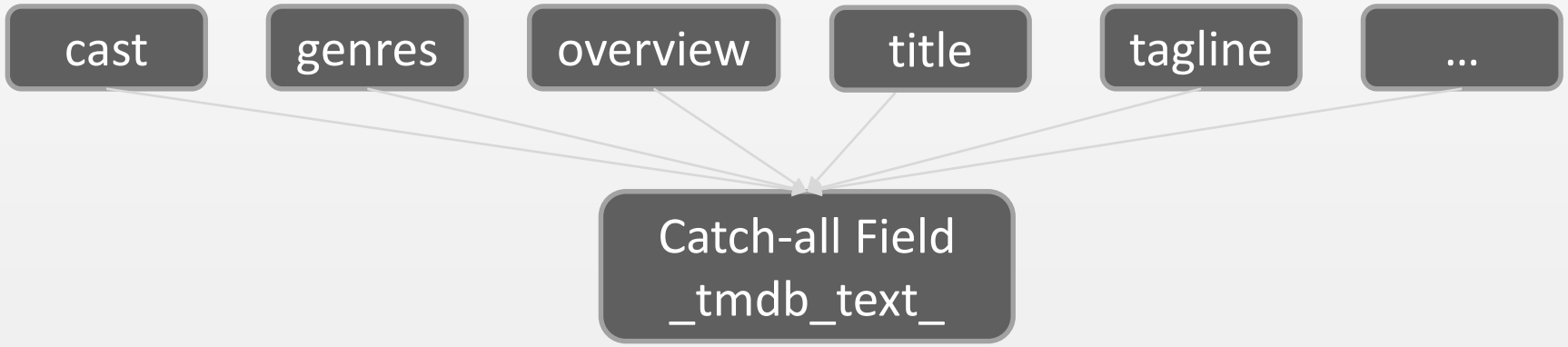
Praktisches Beispiel - Setting



Powered by Data from THE MOVIE DB

Phase 1 – Standardsuche

- Suche in einem Feld (Catch-all Field)
- Kopieren aller zu durchsuchenden Inhalte in Catch-all Field
- Beispiel: `q=_tmdb_text_:pirates of the caribbean`



👎 Identische Gewichtung unterschiedlicher Inhalte

Phase 2 – eDisMax mit Gewichtung

- Mehrere Felder, die einzeln gewichtet sind
- Beispiel:
  `q=pirates of the caribbean&qf=title^5 overview^2&defType=edismax`


Applikationsseitig einfach umsetzbar


Keine komplizierten Suchanfragen für den Benutzer, wie

`q=title:pirates^5 OR title:caribbean^5 … OR overview:pirates^2 OR …`


Jeder Term wird separat betrachtet

Phase 3 – eDisMax mit Phrasen

- Eine Folge von getroffenen Termen wird höher bewertet als separate Treffer
- Beispiel:

```
q=pirates of the caribbean&qf=title^5 overview^2&pf=title^10
overview^4&ps=1&defType=edismax
```

Bessere Ergebnisse bei Suche nach zwei oder mehr Termen

Noch mehr Tuning-Parameter

Phase 4 – eDisMax mit Boost

- Function Queries: Bestimmte Feldwerte der Treffer als Relevanzkriterium
- Beispiel Release-Datum: aktuellere Treffer sollen besser bewertet werden

```
q=pirates of the caribbean&qf=title^5 overview^2&pf=title^15
overview^4&ps=1&defType=edismax&
boost=recip(ms(NOW,release_date),3.16e-11,1,1)
```

Berücksichtigung numerischer Werte aus dem Index möglich

Performance-Verluste bei vielen Treffern und komplexen Funktionen

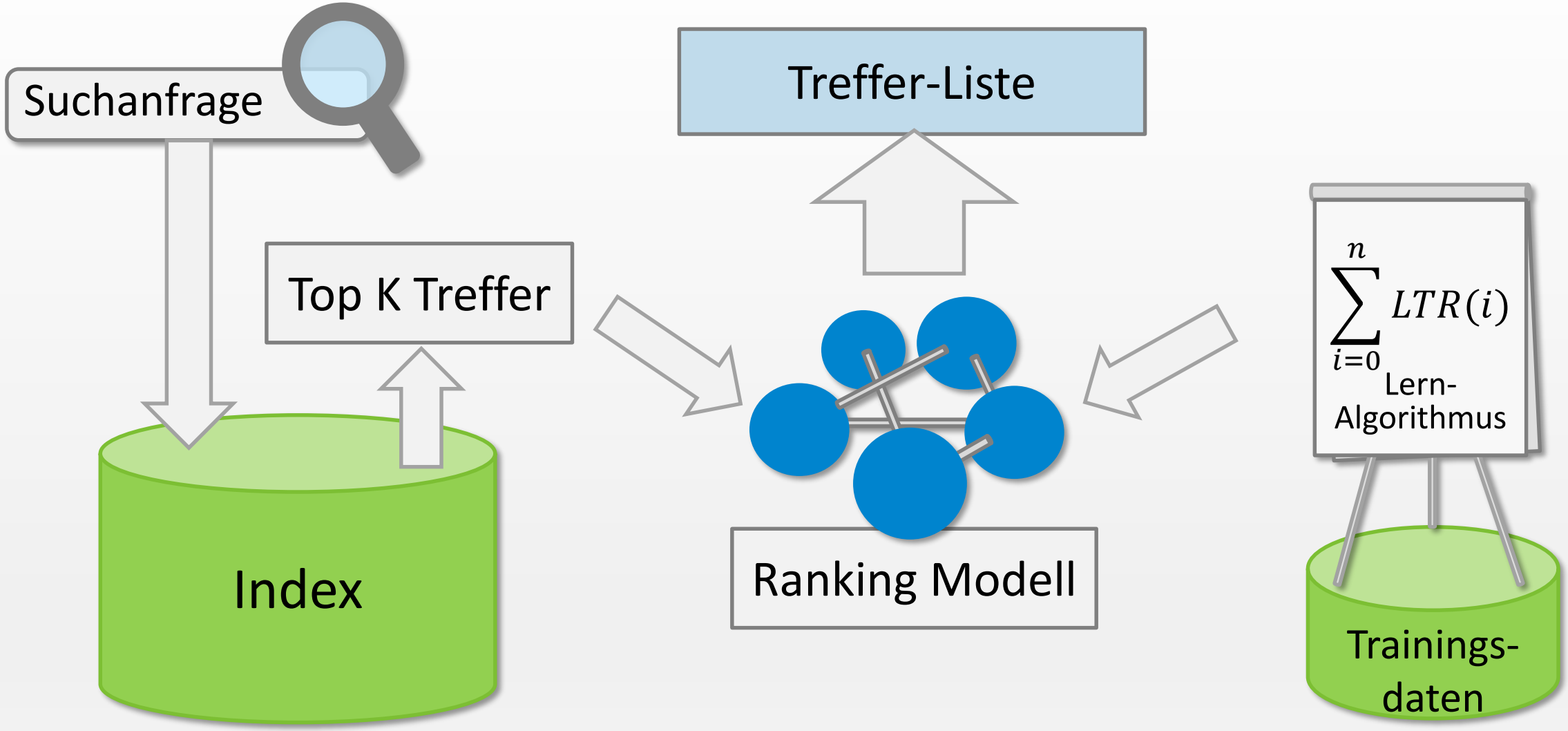
Weiterer zu testender Faktor

LTR - Motivation

- Phase 1 -4 helfen, haben aber **Grenzen**:
  - **Hoher** zeitlicher **Aufwand** durch **manuelles Finetuning**
  - **Hoher Testaufwand** durch Seiteneffekte: Änderungen bei den Parametern können andere Suchanfragen schlechter machen.
  - **Unübersichtlich**: Fehlender Überblick bei vielen Parametern
  - Domänenspezifisches **Expertenwissen nötig**

→ Lösungsansatz: **Automatisiertes Tuning** der Parameter mit **Machine Learning**!

LTR – Learning To Rank



LTR – Verwendungen in der Praxis

Bekannte Verwendungen in der Praxis:

- 2003 AltaVista
- 2005 Bing (RankNet von Microsoft Research)
- 2009 Yandex (russische Suchmaschine) mit MatrixNet
- 2017 Integration von LTR in Apache Solr

Quelle: Wikipedia - https://en.wikipedia.org/wiki/Learning_to_rank

LTR - Umsetzung in Solr (I)

Zweistufig, da es sonst Probleme mit der Performance geben kann*

* (LTR-Scoring kann rechenaufwendig sein und muss für jedes Dokument einzeln angewendet werden)

LTR - Umsetzung in Solr (II)

- Der durch LTR berechnete „Boost“ basiert auf sogenannten Features
- Features sind die Merkmale, die zum Maschinellen Lernen benutzt werden. Sie müssen aber auch auf die Daten in Solr abgebildet werden:
  - Solr Queries
  - Feldwerte aus den Dokumenten
  - Externe Werte (unabhängig vom Index)
  - Ursprünglicher Relevanz-Score

LTR - Features

```
{
  "name":"originalScore",
  "class":"org.apache.solr.ltr.feature.OriginalScoreFeature",
  "params":{}
},
{
  "name": "isShort",
  "class": "org.apache.solr.ltr.feature.SolrFeature",
  "params":{ "fq": ["runtime:[1 TO 20]"] }
},
{
  "name" : "isMobileUser",
  "class" : "org.apache.solr.ltr.feature.ValueFeature",
  "params" : { "value" : "${isMobileUser:<default>}", "required":true }
}
```

LTR - Features

- Relevanz der Features muss ermittelt werden
  - durch Experten
  - durch Benutzeraktionen und Zuweisung von Experten

Herausforderungen:

- Wenige Experten: Teuer und/oder zeitaufwendig, um genügend Daten zu sammeln
- Noisy Data: Verwendung von Userdaten ohne fachliche Prüfung
- Abhängigkeit von Anwendungsdomäne

SEARCH ANALYTICS BIG DATA

LTR - Features

```
{ "1": "isShort",
"2": "isMobileUser",
"3": "isPrimetime",
"4": "isAfternoon",
"5": "isNight",
"6": "isFullLength",
"7": "isAdult",
"8": "isComedy",
"9": "isDrama",
"10": "isAction",
"11": "isFrenchUser",
"12": "isFrenchMovie",
"13": "isFrenchLocation"}
```

Relevanz	Query	isShort	isMobile	Features									
0.0625	qid:1	1:0	2:1	3:0	4:0	5:1	6:0	7:0	8:0	9:1	10:0	11:1	12:0	13:0
1.0	qid:1	1:0	2:1	3:0	4:0	5:1	6:0	7:0	8:1	9:0	10:1	11:1	12:0	13:0
0.25	qid:1	1:0	2:1	3:0	4:0	5:1	6:0	7:0	8:0	9:1	10:1	11:1	12:0	13:0
1.0	qid:2	1:1	2:1	3:1	4:0	5:0	6:0	7:1	8:0	9:1	10:1	11:0	12:0	13:0
2.0	qid:2	1:1	2:1	3:1	4:0	5:0	6:0	7:0	8:1	9:1	10:0	11:0	12:0	13:0
0.5	qid:2	1:1	2:0	3:1	4:0	5:0	6:0	7:0	8:1	9:1	10:1	11:0	12:0	13:0

LTR - Features

- Features müssen definiert, normalisiert und gegebenenfalls quantisiert werden:
  - Zu viele unnötige Features können die relevanten ausblenden.
  - Nicht normalisierte Features können andere Features überschatten.
- Nur numerische Features
  - Kategorische Features (one-hot-encoding, Problem bei hoher Kardinalität)
  - Binäre Features (Problem bei der Normalisierung mit anderen numerischen Features)
- Wie geht man mit fehlenden Features um?

ML-Ranking Algorithmen

- Fülle an Algorithmen und Implementierungen
  - Ranklib: bekannteste Implementierung von mehreren Algorithmen (unter anderem LambdaMART)
  - SVMrank
  - libsvm
- Modelle für Solrs LTR
  - Standardimplementierung: LinearModel und MultipleAdditiveTreesModel; alle Algorithmen, die diese Modelle erzeugen können (evtl. noch umwandeln in JSON)
  - Für andere Modelle gibt es eine Basisklasse, die man erweitern kann.
- Mit unterschiedlichen Parametern zum Finetuning

LinearModel

- Auch mit großem Feature-Raum möglich
- Einfaches Ergebnis (quasi eine Liste von automatisch optimierten Boost-Faktoren)

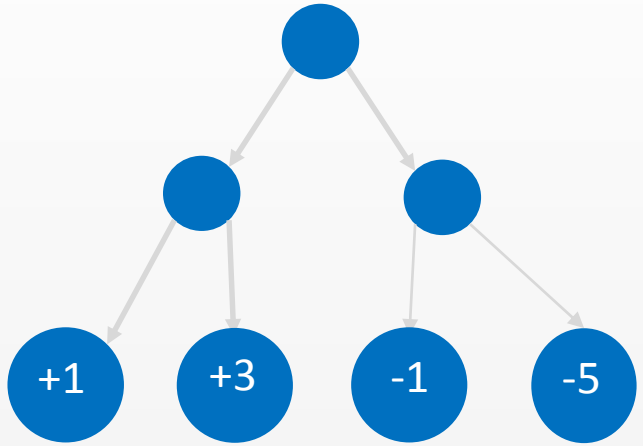
- Nur linear-separierbare Probleme, z.B. eine Untergruppe (obwohl die zum Erstellen verwendeten SVMs das Problem nicht haben)
- Für Menschen schwierig zu lesen

Tools

- RankSVM
- Pranking

MultipleAdditiveTreesModel

- Auch linear-separierbare Probleme
- Einzelne Regeln sind für Menschen nachvollziehbar
- Anfällig für Overfitting
- Summe der Regeln nicht nachvollziehbar/überschaubar für Menschen

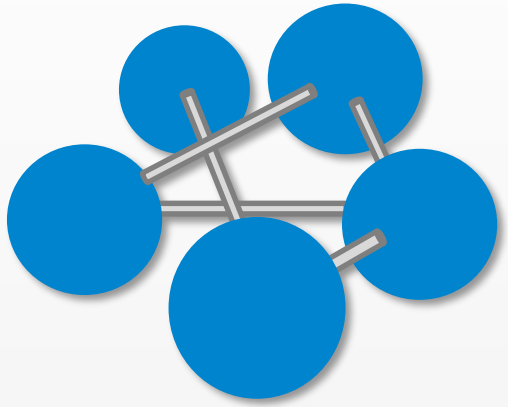


Tools

- LambdaMART
- Gradient Boosted Regression Trees (GBRT)

LTR – Modell-Herausforderungen

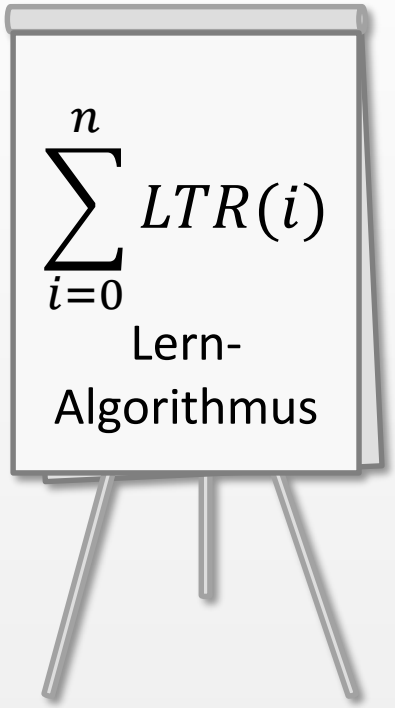
- Overfitting: Falsche, zu viele und/oder nicht-normalisierte Features
- Underfitting: nicht genug oder biased Daten



➔ Muss im Einzelnen geprüft werden, da es vom Algorithmus, den Daten und den Features abhängt. Testen, testen, testen…

LTR – Fachliche Herausforderungen

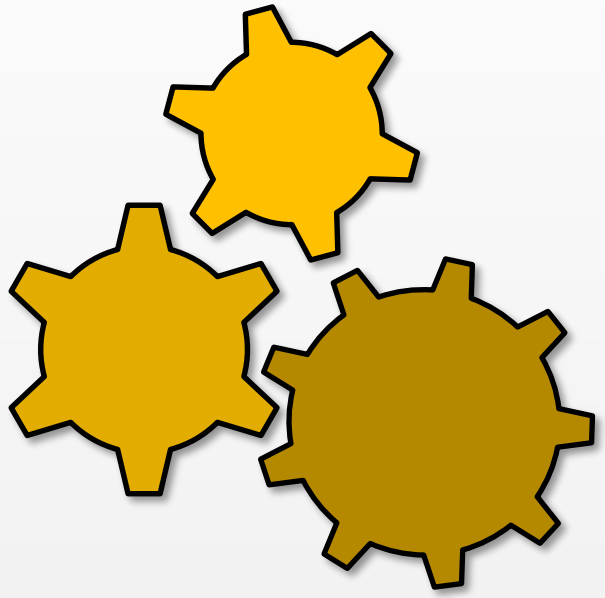
- Das automatische Tuning ist nicht geschenkt
  - Für den Lernalgorithmus braucht es Daten (von Nutzern oder von Experten)
  - Relevanz muss ermittelt werden (Expertenwissen)
  - Features müssen definiert und normalisiert werden (Expertenwissen)
  - Der Lernalgorithmus muss konfiguriert werden (Expertenwissen)



➔ Nicht einfacher, sondern schwieriger als manuelles Tuning

LTR - Technische Herausforderungen

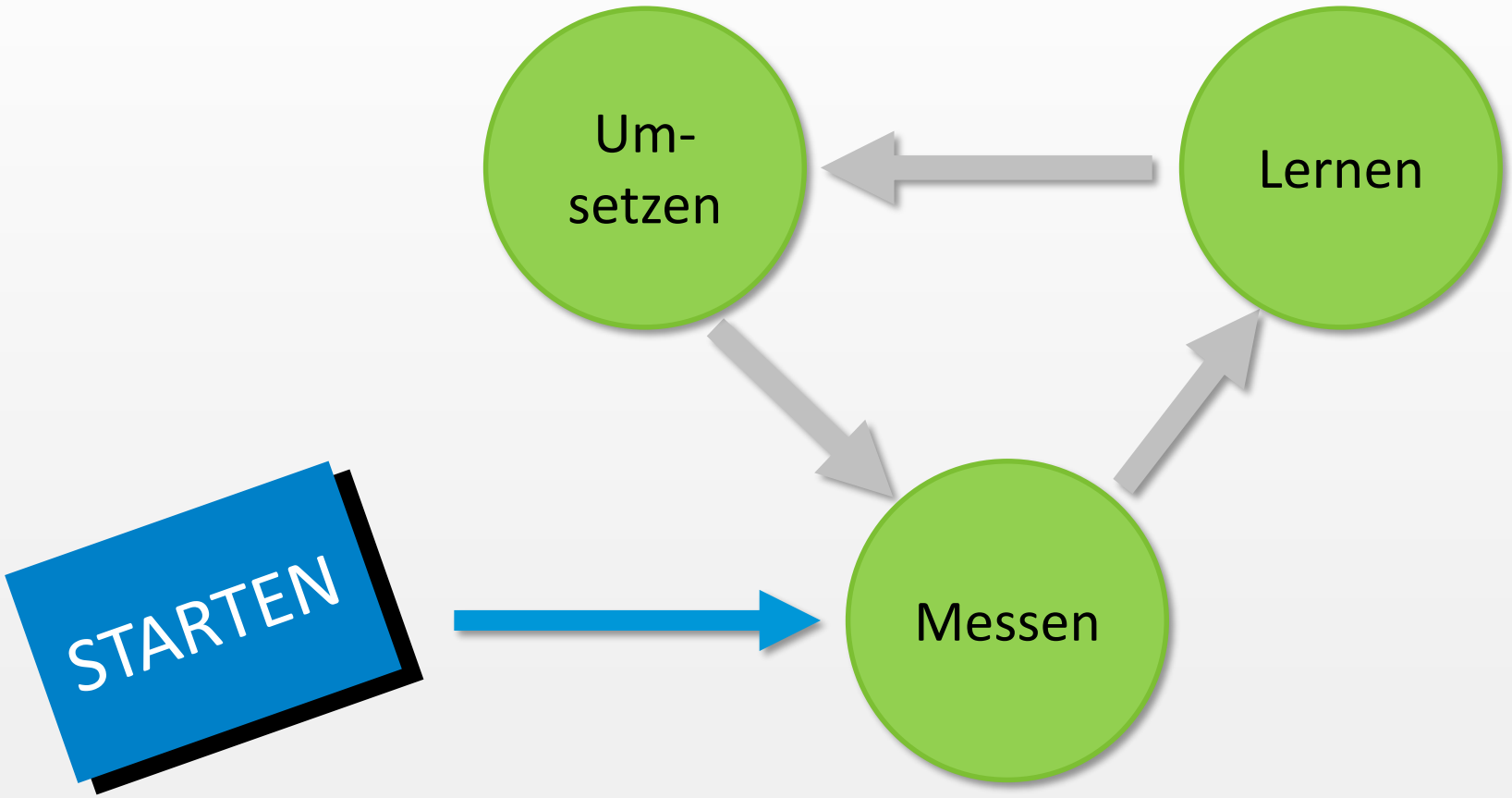
- Überführung der Ausgabe des Learning Algorithmus in Solr-JSON Modell
  - Nicht automatisch (aber automatisierbar)
  - Fehleranfällig
- Analyse der ausgewählten Features
  - Schwierig zu debuggen (Tools nötig)
  - Schwierig zu verstehen



Solr Besonderheiten

- Performance bei Query Re-Ranking
  - Je mehr Dokumente komplexe Berechnungsprozesse durchlaufen, desto größer die Auswirkungen.
- ZooKeeper Limit für Größe von Modellen (vor Version 7.2)
  - ZooKeeper für Dateien kleinerer Größe gedacht (zNode Limit 1MB)
  - Anpassung des Limits grundsätzlich möglich, muss für ZK-Server und Clients (Solr) definiert werden.

Ausblick



KONTAKT

SHI GmbH
Konrad-Adenauer-Allee 15
D - 86150 Augsburg

info@shi-gmbh.com
+49 821 - 74 82 633 - 0
@SHIEngineers

Referenzen

- https://en.wikipedia.org/wiki/Machine_learning
- https://en.wikipedia.org/wiki/Learning_to_rank
- http://alexbenedetti.blogspot.de
- https://sematext.com/blog/2012/01/06/relevance-tuning-and-competitive-advantage-via-search-analytics/
- https://lucene.apache.org/solr/guide/6_6/learning-to-rank.html
- http://opensourceconnections.com/blog/2017/02/14/elasticsearch-learning-to-rank/
- https://www.slideshare.net/lucidworks/learning-to-rank-in-solr-presented-by-michael-nilsson-diego-ceccarelli-bloomberg-lp
- https://github.com/o19s/elasticsearch-learning-to-rank
- https://berlinbuzzwords.de/17/session/we-built-elasticsearch-learning-rank-plugin-then-came-hard-part