FRIEDRICH-SCHILLER-
UNIVERSITÄT
JENA

Entwicklung eines Benutzermodells zur nutzeradaptiven Suche von Datensätzen

Masterarbeit

zur Erlangung des akademischen Grades
Master of Science (M. Sc.)
im Studiengang Informatik
Friedrich-Schiller-Universität Jena
Fakultät für Mathematik und Informatik

eingereicht von Tim Surber
geb. am 10.07.1996 in Jena

Themenverantwortliche: Prof. Birgitta König-Ries, FSU Jena
Betreuer: Sirko Schindler, DLR-Institut für Datenwissenschaften, Jena
Zweitgutachterin: Dr. Friederike Klan, DLR-Institut für Datenwissenschaften, Jena

Jena, 17. März 2021

Kurzfassung

Datenportale stellen eine große Menge an Datensätzen zur Verfügung. Das Finden des gesuchten Datensatzes stellt dabei eine Herausforderung dar. Das Ziel dieser Arbeit ist es, den Nutzer gezielter zum gewünschten Datensatz zu führen. Die gewählte Methode ist das Erstellen eines Benutzermodells, um die Suche nutzeradaptiv zu gestalten. Als eine mögliche Adaption wird in dieser Arbeit das Unterstützen des Nutzers bei der Verwendung von Suchfiltern betrachtet. Durch Vorhersage zukünfiger Interaktionen können Suchfiltervorschläge erzeugt werden. Dazu wurde ein Modell unter Verwendung von partiell sortierten Sequenzregeln anhand vorheriger Nutzersitzungen trainiert.

Das Modell wurde anhand eines vorliegenden Datensatzes, den Aufzeichnungen eines Geodatenportals, geprüft. Es konnten die Suchfilter des nächsten Schrittes mit einer Genauigkeit von 51% vorhergesagt werden, was eine deutliche Verbesserung gegenüber eine zufälligen Vorhersage unter gleichen Bedinungen mit 11% darstellt. Die Ergebnisse zeigen, dass die Entwicklung eines Benutzermodells ein geeignetes Vorgehen ist, um Suchfilter vorherzusagen. Dies kann eingesetzt werden, um das Suchen in Datenportalen zu verbessern.

Abstract

Data portals host a large amount of data records. Finding the right data set can be challenging. The goal of this work is to guide the user to the correct data set. This is implemented by creating a user model to make the search user-adaptive. This work focuses on supporting the user in using search filters. By predicting future interactions, search filter suggestions can be generated. For this purpose, a model is trained using partially-ordered sequential rules with previous user sessions.

The model was tested on an existing dataset, the records of a geospatial portal. It was able to predict the search filters of the next step with an accuracy of 51%, which is a significant improvement over a random prediction under the same conditions with 11%. The results show that the development of a user model is a suitable process to predict search filters. This can be used to improve searching in data portals.

Inhaltsverzeichnis

1 Einleitung **1**

2 Stand der Forschung **4**

2.1 Sequenzmusteranalyse 6

2.1.1 Transaktionsdatenbank 6

2.1.2 Bewertung der Signifikanz 8

2.1.3 Vorhersage 8

2.1.4 Anwendungen 9

2.1.5 Implementationen 9

2.2 Sequenzregelanalyse 10

2.2.1 Bewertung der Signifikanz 10

2.2.2 Varianten 11

2.2.3 Anwendungen 13

2.2.4 Implementationen 13

2.3 Markov-Ketten 14

2.3.1 Anwendungen 15

2.3.2 Implementierungen 15

2.4 Künstliche Neuronale Netze 16

2.4.1 Anwendungen 17

2.4.2 Implementierungen 17

2.5 EOWEB®GeoPortal 18

2.6 Benutzermodelle 22

3 Konzept **24**

3.1 Anforderungen an den Datensatz 25

3.2 Anforderungen an das Vorhersageverfahren 27

3.3 Auswahl des Vorhersageverfahrens 27

3.4 Definition der Elemente 30

4 Datensatz **31**

4.1 Inhalt 31

4.2 Merkmale des Datensatzes 34

5 Umsetzung **38**

5.1 Datenvorverarbeitung 38
5.1.1 Aggregation 39
5.1.2 Generieren der Transaktionsdatenbank aus Feldern des Datensatzes 39
5.1.3 Filterung 40
5.2 Vorhersage durch Anwendung der Sequenzregelanalyse 41
5.2.1 Vorhersage unter Anwendung partiell sortierter Sequenzregeln (PSSR) 42
5.2.2 Priorisierte partiell sortierte Sequenzregeln (PPSSR) 46
5.3 Technische Umsetzung 48

6 Evaluation **49**

6.1 Ergebnisse der Vorfilterung 49
6.2 Parametrisierung 50
6.2.1 Hyperparameter 50
6.2.2 Konfiguration 51
6.2.3 Verwendete Parametergrößen 52
6.2.4 Parameterkombinationen bilden 53
6.3 Prüfen der Vorhersage 53
6.4 Auswahl des Modells 54
6.4.1 Auswahlkriterum 54
6.4.2 Vorgehen 55
6.5 Vergleichsverfahren 57
6.6 Auswertung der Vorhersageergebnisse 58
6.7 Einfluss der Parameter 59
6.7.1 Umfang der Trainingsdaten 60
6.7.2 Optimale Hyperparameter 61

7 Fazit **63**

Literatur **65**

Abbildungsverzeichnis **70**

Tabellenverzeichnis **71**

1 Einleitung

Ein Datenportal stellt eine Sammlung von Datensätzen zur Verfügung. Es bietet Funktionen zum gezielten Auffinden von Datensätzen, dazu werden üblicherweise Datensätze mit Metadaten versehen, um die Suche zu erleichtern.

Eine Umfrage unter Wissenschaftlern zeigt, dass das Finden von passenden Datensätzen in Datenportalen zum Einsatz in der Forschung eine anspruchsvolle (73%) oder sogar schwierige (19%) Aufgabe ist. Ein häufig genannter Grund dafür ist, dass die zur Verfügung stehenden Suchwerkzeuge nicht adäquat sind (Gregory u. a. 2020).

Der Prozess des Suchens lässt sich in zwei Formen von Suchanwendungen einteilen (Heyer u. a. 2011). Die nachschlagende Suche wird etwa zum Aufsuchen von Fakten oder zum Beantworten einer konkreten Frage genutzt. Ein übliches Bedienelement der nachschlagenden Suche ist eine Suchleiste zur Eingabe von Stichwörtern (Marchionini 2006).

Bei einem Datenportal wird dem Nutzer hingegen eine andere Suchform zur Verfügung gestellt. Ein Grund dafür ist, dass die nachschlagende Suche sich auf Datenportalen als wenig effektiv herausgestellt hat. Eine Studie anhand eines biomedizinischen Datenportals zeigt, dass die Verwendung einer klassischen Suchleiste in vielen Fällen nicht zum gewünschten Ergebnis führt (Dixit u. a. 2017). Diese Form des Suchens wird auch als exploratives Suchen bezeichnet. Als Bedienelement der explorativen Suche können Suchfilter verwendet werden, um die Ergebnismenge immer weiter einzuschränken, bis schließlich zu dem gesuchten Datensatz navigiert werden kann.

Die Auswahl der Suchfilter durch den Nutzer ist entscheidend für den Erfolg der Suche. Eine gezielte Auswahl der Suchfilter führt in wenigen Schritten zum korrekten Ergebnis, was bereits für den Spezialfall einer Filtersuche, der Facettensuche, untersucht wurde (Basu Roy u. a. 2008). Eine unvorteilhafte Wahl der Suchfilter hingegen

führt zu einem längeren Suchprozess, was sich negativ auf die Nutzerzufriedenheit auswirkt (Y. Xu und Mease 2009).

Wenn der Nutzer jedoch bei der Auswahl der Suchfilter unterstützt wird, kann er gezielter und häufer zu seinen gewünschten Suchergebnissen geführt werden. Diese Unterstützung lässt sich etwa durch gezieltes Priorisieren bei der Anzeige der Suchfilter realisieren. Zur Priorisierung der Suchfilter kann eine Vorhersage der nächsten Nutzerinteraktion dienen.

Die Vorhersage von folgenden Nutzerinteraktion soll in dieser Arbeit betrachtet werden. Dazu soll ein eine Benutzermodell anhand von historischen Nutzungsdaten trainiert werden. Hierfür wird in dieser Arbeit ein Prozess entwickelt und angewendet. Dieser besteht aus der Vorverarbeitung der Daten, der Wahl eines passenden Mining- bzw. Analyseverfahrens und abschließender Verifizierung der Korrektheit der Ergebnisse. Die Vorhersage geschieht nutzerspezifisch, die Adaption erfolgt anhand des Benutzermodells.

Der Vorhersageprozess soll unabhängig von einem spezifischen Datensatz sein und für alle Datensätze, die in ein bestimmtes Schema passen, möglich sein. Die Evaluierung der getätigten Vorhersage soll anhand eines Beispieldatensatzes erfolgen. Dieser stammt aus dem EOWEB®GeoPortal (EGP) welches vom Deutschen Zentrum für Luft- und Raumfahrt (DLR) unterhalten wird.

Dieses Datenportal stellt Erdbeobachtungsdaten zur Verfügung. Das EGP hat Suchfilter integriert. Diese erlauben unter anderem eine zeitliche Einschränkung der Ergebnisse (das Aufnahmedatum) oder eine räumliche Einschränkung (die abgebildete Region). Derzeit ist im EGP noch keine Nutzermodellierung enthalten. Aus den Aufzeichnungen des EGP lässt sich schließen, dass ein beträchtlicher Teil der Nutzer den Suchprozess abbricht, bevor ein Ergebnis gefunden werden kann. Dies gibt einen Hinweis darauf, dass die Optimierung des Suchprozesses für konkret diese Plattform hilfreich ist.

Für das EGP liegen Aufzeichnungen über mehrere Monate in Form von Logdateien vor. Aus diesen lassen sich Nutzersitzungen als eine Sequenz von Interaktionen rekonstruieren. Neben einem Nutzeridentifikator und einem Zeitstempel enthält jede Interaktion auch Informationen über Art und Inhalt der verwendeten Filter der Suche. Dies kann als Ausgang für eine Mustererkennung mithilfe eines Data-

Mining-Prozesses dienen. Darauf aufbauend kann dann eine Vorhersage für zukünftige Interaktionen ermöglicht werden. Durch die Aufteilung des Datensatzes in Trainings- und Testdaten kann eine Prüfung der Vorhersagegenauigkeit durchgeführt werden.

Ziel dieser Arbeit ist es, Nutzer der Suchfunktion eines Datenportals gezielter zu dem gewünschten Datensatz zu führen. In dieser Arbeit soll ein Teilaspekt behandelt werden, der verwendet werden kann, um dieses Ziel zu erreichen: Das Priorisieren von Suchfiltern. Dabei soll ein Vorgehen entwickelt werden, um zukünftige Nutzerinteraktionen des Suchprozesses vorherzusagen. Durchgeführt wird die Vorhersage ausschließlich anhand von Logaufzeichnungen. Es werden keine Umfragen mit Nutzergruppen durchgeführt. Der tatsächliche Einsatz der vorhergesagten Suchfiltern in der Benutzeroberfläche und die zugehörige Auswertung dessen bleibt Gegenstand der zukünftigen Forschung.

Bewertet wird das Erreichen des Ziels in dieser Arbeit durch Evaluation des Vorhersageergebnis, wobei die Vorhersagegenauigkeit der wichtigste Messwert ist. Dies erfolgt, in dem der EGP Datensatz in Test- und Trainingsdaten gesplittet wird.

Im folgenden Kapitel wird der Stand der Forschung erläutert, ebenso findet eine detaillierte Erklärung des EGP statt. In Kapitel 3 wird das Konzept der Vorhersage vorgestellt. Anschließend folgt eine Informationen über den verwendeten Datensatz. Kapitel 5 zeigt die zugehörige Implementation, anschließend erfolgt eine Diskussion der Vorhersageergebnisse. Zum Schluss findet ein Ausblick auf mögliche zukünftige Entwicklungen statt.

2 Stand der Forschung

Es existieren bereits Veröffentlichungen, welche eine Vorhersage von Filtern mit dem Ziel, das Suchergebnis zu verbessern, durchführen.

In (Chantamunee, Wong und Fung 2020) werden künstliche neuronale Netze zum Bilden von personalisierten Empfehlungen von Suchfacetten eingesetzt. Das entwickelte Verfahren wird anhand drei öffentlich verfügbarer Datensätze geprüft, diese Datensätze kommen aus den Bereichen Tourismus und Filmempfehlungen.

Durch (Niu, Fan und Zhang 2019) wird gezeigt, wie Entscheidungsbäume für die Vorhersage von Suchfacetten genutzt werden können. Dies geschieht anhand eines Datensatzes, welcher aus Benutzersitzungen eines Online-Katalogs einer Bibliothek gewonnen wurde. Zusätzlich wird gezeigt, dass die Facettennutzung stark von vorherigen Interaktionen des Nutzers abhängig ist.

Diese Werke unterscheiden sich jedoch in zwei entscheidenden Punkten von dieser Arbeit:

Die vorgestellten Werke beziehen sich auf eine Spezialform der Suchfilter, die Facetten. Diese ähneln den in dieser Arbeit behandelten klassischen Suchfiltern in gewisser Weise, zeigen aber dennoch entscheidende Unterschiede auf. Die wichtigsten Unterschiede sind in Tabelle 2.1 dargestellt. Der verschiedene Aufbau von Suchfiltern und Facetten resultiert zwangsläufig in einem abweichenden Vorgehen des Nutzers.

Der zweite Unterschied betrifft die Art der untersuchten Plattform. In dieser Arbeit wird die Suche von Datensätzen in einem Datenportal behandelt. In (Kern und Mathiak 2015) wird gezeigt, dass die Suche nach Datensätzen sich von sonstigen Suchsituationen (es wird ein Vergleich mit der Suche nach Literatur durchgeführt) unterscheidet. Die Nutzer von Datenportalen stellen andere Anforderungen an die

Suchfilter	Facetten
Ändern sich nicht während der Suche	Ändern sich abhängig von den Suchergebnissen
Können vor und nach einer Suchanfrage selektiert werden	Können nur nach einer Suchanfrage selektiert werden
Hauptsächlich genutzt in informationsreichen Seiten	Hauptsächlich in E-Commerce Anwendungen genutzt
Leere Ergebnismenge möglich	Ergebnismenge zeigt immer mindestens ein Resultat

Tabelle 2.1: Vergleich Suchfilter mit Facetten. Quelle: (Kennedy 2020)

Ergebnisse, so ist das Vorkommen und die Qualität von Metadaten besonders wichtig. Auch die Suchvorgehen unterscheiden sich, eine Suche auf Datenportalen wird aufgrund der Wichtigkeit für die Nutzer intensiver durchgeführt.

Die Einschränkung auf Datenportale und die Vorhersage von klassischen Suchfiltern unterscheidet diese Arbeit von bisherigen Publikationen und behandelt somit neue Aspekte. Dem Autor ist keine Publikation bekannt, welche Vorhersagen von Suchfiltern in einem Datenportal durchführt.

Die Vorhersage von zukünftigen Benutzerinteraktionen mit enthaltenen Suchfiltern erfolgt auf Basis vorheriger Benutzerinteraktionen. Die aufeinanderfolgenden Interaktionen können als sequenzielle Daten betrachtet werden. Es folgt eine Vorstellung von in der Literatur bekannten Verfahren, welche eine Vorhersage anhand vorliegender sequenzieller Daten durchführen können.

Bei dem Finden von Mustern in sequenziellen Daten ist das Kernproblem, zu untersuchen, welche Elemente häufig nacheinander auftreten. Es unterscheidet sich dabei von Verfahren, welche die Reihenfolge von Elementen nicht beachten, wie etwa Frequent Itemset Mining oder die Assoziationsanalyse. Diese werden genutzt, um häufig zusammen auftretende Elemente zu finden (Brückner und Scheffer 2013).

Nach der Vorstellung der Vorhersageverfahren werden in diesem Kapitel relevante Hintergrundinformationen des EOWEB GeoPortals genannt. Dieses ist der Ursprung des in dieser Arbeit verwendeten Datensatzes. Zum Schluss werden Informationen über Benutzermodelle vorgestellt.

2.1 Sequenzmusteranalyse

Die Sequenzmusteranalyse ist ein Verfahren, welches als Eingabe eine Transaktionsdatenbank benötigt. Aus der Transaktionsdatenbank werden Sequenzmuster extrahiert. Sequenzmuster zeigen nacheinander auftretende Elemente einer Sequenz, wobei die signifikanten Sequenzmuster am relevantesten sind.

2.1.1 Transaktionsdatenbank

In Abbildung 2.1 wird ein Beispiel einer Transaktionsdatenbank in zwei verschiedenen Darstellungsformen gezeigt. In Abbildung 2.1a ist ein Ausschnitt der Transaktionsdatenbank in der ausführlichen Schreibweise gezeigt. In Abbildung 2.1b werden die gleichen Daten dagegen in einer verkürzten Schreibweise gezeigt, wie sie auch im Rest der Arbeit verwendet wird. Diese Form eignet sich auch zur Repräsentation der Transaktionsdatenbank in Textdateien.

Das Beispiel zeigt die Einkäufe von sechs Kunden, jeder Kunde wird durch eine Kunden-ID bezeichnet. Jedem Kunden lassen sich Einkäufe zuordnen. Ein Einkauf enthält die erworbenen Produkte A bis E. Die Einkäufe sind sortiert in der Transaktionsdatenbank eingefügt, dass bedeutet Einkauf 2 von Kunde 1 fand zu einem späteren Zeitpunkt als Einkauf 1 von Kunde 1 statt. Jeder Einkauf lässt sich eindeutig durch das Tupel (Kunden-ID, Einkaufs-ID) bestimmen.

Das Beispiel zeigt etwa, dass Kunde 1 im ersten Einkauf die Produkte A und C erwarb. Zu einem späteren Einkauf wurde nur Produkt B erworben, zu einem dritten Einkauf C und E. Kunde 2 erwarb im Beispiel die Produkte A, B und C einzeln in drei getrennten Einkäufen.

Eine Transaktionsdatenbank wird durch folgende Bestandteile bestimmt:

- **Sequenz** Die Transaktionsdatenbank enthält Sequenzen. In Abbildung 2.1a wird eine Sequenz durch die gleiche Kunden-ID gekennzeichnet. In Abbildung 2.1b steht jede neue Zeile für eine Sequenz. Im Beispiel entspricht die Sequenz der Einkaufshistorie eines einzelnen Kundens.

Kunden-ID	Einkaufs-ID	A	B	C	D	E
1	1	✓	×	✓	×	×
1	2	×	✓	×	×	×
1	3	×	×	✓	×	✓
2	1	✓	×	×	×	×
2	2	×	✓	×	×	×
2	3	×	×	✓	×	×
	...					

(a) Ausführliche Schreibweise

1: $< \{A, C\}, \{B\}, \{C, E\} >$
2: $< \{A\}, \{B\}, \{C\} >$
3: $< \{A, C\}, \{B\}, \{D\} >$
4: $< \{B, D\}, \{E\}, \{A\} >$
5: $< \{E\}, \{D\}, \{B\}, \{A\} >$
6: $< \{B, D, E\}, \{A\} >$

(b) Verkürzte Schreibweise

Abbildung 2.1: Beispiel für eine Transaktionsdatenbank

- **Ereignis** Eine Sequenz erhält Ereignisse. Die Ereignisse einer Sequenz sind zeitlich sortiert. In Abbildung 2.1a wird jede Sequenz durch die Kombination von Kunden-ID und Einkaufs-ID definiert. In Abbildung 2.1b wird eine Sequenz durch eine Einfassung in geschweifte Klammern {} begrenzt. Die Einkäufe aus dem Beispiel entsprechen den Ereignissen.

- **Element** Ein Ereignis enthält eine Menge von Elementen. In dieser Menge herrscht keine Ordnungsrelation. Die Elemente entsprechen im Beispiel den Artikeln A bis E des Einkaufs .

Eine mögliches Sequenzmuster, welches aus Abbildung 2.1 abgeleitet werden kann, ist beispielsweise $< \{A\}, \{B\}, \{C\} >$. Es ist in den Sequenzen 1 und 2 vorhanden. Die Trennung der Mengen aus Elementen durch ein Komma impliziert eine zeitliche Abfolge; $\{B\}$ tritt somit zu einem späteren Zeitpunkt als $\{A\}$. Dabei ist es keine Anforderung, dass die Elemente der Sequenzmuster in direkt aufeinanderfolgenden Ereignissen auftreten. Es können auch beliebig viele Ereignisse dazwischen auftreten.

Ein anderes Sequenzmuster ist $< \{B, D\}, \{A\} >$, welches in den Sequenzen 4 und 6 auftritt. Wenn mehrere Elemente innerhalb der geschweiften Klammern {} vorhanden sind, dann treten diese zum gleichen Zeitpunkt auf. Dieses Muster gibt somit an, dass B und D zum gleichen Zeitpunkt erworben werden und $\{A\}$ zu einem späteren Zeitpunkt.

2.1.2 Bewertung der Signifikanz

Bei der Sequenzmusteranalyse an einem umfangreichen Datensatz entstehen entsprechend viele Sequenzmuster. Um zu bewerten, wie interessant ein Sequenzmuster ist, kann der Messwert „Support“ verwendet werden.

Support
Der Support eines Sequenzmusters m gibt an, wie groß der Anteil der Sequenzen ist, die dieses Muster enthalten.

$$sup(m) = \frac{\text{Anzahl der Sequenzen, die } m \text{ enthalten}}{\text{Anzahl aller Sequenzen}}$$

Das Sequenzmuster $< \{A\}, \{B\}, \{C\} >$ tritt in den Sequenzen 1 und 2 von insgesamt sechs Sequenzen auf, der zugehörige Support-Wert ist somit $\frac{2}{6} = 33\%$.

2.1.3 Vorhersage

Sequenzmuster mit dem zugehörigen Support-Wert sind geeignet um häufig nacheinander auftretende Elemente zu finden. Wie unter anderem durch (Pitman und Zanker 2011) und (Fournier Viger, Faghihi u. a. 2012) analysiert wurde, eignen sich Sequenzmuster nur begrenzt zur Vorhersage.

Die Vorhersage durch Sequenzmuster erfordert einen Trainingsdatensatz in Form einer Transaktionsdatenbank. In dem Trainingsdatensatz werden häufig auftretende Sequenzmuster gefunden. Anschließend kann anhand von neuen beziehungsweise unbekannten Sequenzen eine Vorhersage des folgenden Elements durchgeführt werden. Dabei wird ein Sequenzmuster gewählt, welches eine große Übereinstimmung mit der neuen Sequenz besitzt. Die verbleibenden Elemente des gewählten Sequenzmusters entsprechen der Vorhersage.

Die Einschränkungen der Vorhersage durch Sequenzmuster werden an einem Beispiel verdeutlicht, welches die Transaktionsdatenbank Abbildung 2.1 als Trainingsdaten verwendet. Dazu wird die Vorhersage einer neuen Sequenz betrachtet. Es sind bereits zwei Ereignisse in dieser Sequenz enthalten, wobei das erste Ereignis das Element $\{A\}$ und das zweite Ereignis $\{B\}$ enthält. Ziel ist die Vorhersage des folgenden

Elements. Dies soll unter der Bedingung durchgeführt werden, dass die Vorhersage zu mindestens 80% korrekt ist, ansonsten soll keine Vorhersage abgegeben werden.

Diese Information ist jedoch nicht aus dem Sequenzmuster oder dem zugehörigen Supportwert abzulesen. In der Transaktionsdatenbank lässt sich erkennen, dass auf $\{A\}, \{B\}$ in zwei von drei Fällen C folgt, in Sequenz 1 und 2. In einem von drei Fällen folgt auf $\{A\}, \{B\}$ stattdessen D (in Sequenz 3). Unter der Annahme, dass die neu eintreffenden Sequenzen gleich wie die Trainingsdaten verteilt sind, ist die Vorhersage zu 67% korrekt. Damit unterschreitet die Verwendung dieses Musters den geforderten Anspruch von 80% Korrektheit.

Es lässt sich eine für die Vorhersage nützliche Eigenschaft erkennen, welche in den Sequenzmustern fehlt: Wenn $\{A\}, \{B\}$ auftreten, dann folgt in 67% der Fälle $\{C\}$. Diese Eigenschaft wird von den Sequenzregeln genutzt, welche in Abschnitt 2.2 vorgestellt werden.

2.1.4 Anwendungen

Insbesondere im medizinischen Bereich existieren Forschungsprojekte, welche Sequenzmuster verwenden. Sequenzmuster sind geeignet, um etwa Kombinationen von Risikofaktoren einer Krankheit zu ermitteln. Beispiele dafür sind das Erkennen von Risikofaktoren einer HIV-Erkrankung (Velez u. a. 2013) oder der Vogelgrippe (Z. Xu 2016). Neben dem Einsatz im medizinischen Bereich können Sequenzmustern zur Vorhersage des Verkehrsflusses (Ibrahim und Shafiq 2019) verwendet werden.

2.1.5 Implementationen

Das Paket `arulesSequence` (Hornik, Grün und Hahsler 2005) für die Programmiersprache R ermöglicht das Erstellen von Sequenzmustern. Dies verwendet den CSPADE-Algorithmus (Zaki 2000b), welcher eine Erweiterung des SPADE-Algorithmus (Zaki 2000a) um zusätzliche Parameter ist.

2.2 Sequenzregelanalyse

Die Sequenzregelanalyse ist ebenso ein Verfahren, welches eine Transaktionsdatenbank verwendet. Anhand dieser werden Sequenzregeln erzeugt. Sequenzregeln beschreiben nacheinander auftretende Elemente einer Sequenz. Dabei verwenden Sequenzregeln eine andere Darstellungsform als Sequenzmuster. Es werden die auftretenden Elemente auf zwei Seiten unterteilt und üblicherweise durch einen Pfeil getrennt dargestellt, wie etwa $X \rightarrow Y$. Die linke, oder auch vorausgehende Seite der Regel (X) stellt die Bedingung für das Eintreffen der Regel dar. Die rechte, oder auch nachfolgende Seite der Regel (Y) zeigt die Elemente, welche danach auftreten (nicht zwangsläufig direkt danach). Ein Beispiel für Sequenzregeln aus Abbildung 2.1 ist etwa $\{A\}, \{B\} \rightarrow \{C\}$ aus den Sequenzen 1 und 2.

2.2.1 Bewertung der Signifikanz

Für die Anwendung der Sequenzregeln ist entscheidend, aus der großen Menge der generierten Regeln diese auszuwählen, die sich besonders gut für zukünftige Vorhersagen eignen. Diese Regeln werden als interessant oder signifikant bezeichnet.

Neben dem aus den Sequenzmustern bekannten Support-Wert exisitiert bei Sequenzregeln zusätzlich der Konfidenz-Wert.

Support
Für eine Sequenzregel $r = X \rightarrow Y$ ist der Support folgendermaßen definiert:

$$sup(r) = \frac{\text{Anzahl der Sequenzen, die zuerst } X \text{ und anschließend auch Y enthalten}}{\text{Gesamtzahl der Sequenzen}}$$

Der Support gibt somit an, wie groß der Anteil der Sequenzen ist, die diese Regel enthalten.

Konfidenz
Für eine Sequenzregel $r = X \rightarrow Y$ ist die Konfidenz folgendermaßen definiert:

$$conf(r) = \frac{\text{Anzahl der Sequenzen, die zuerst } X \text{ und anschließend auch Y enthalten}}{\text{Anzahl der Sequenzen, die } X \text{ enthalten}}$$

Die Konfidenz gibt an, wie wahrscheinlich es ist, dass auf ein Antreffen der linke Seiten X in einer Sequenz, später die rechte Seite Y folgt. Durch die Konfidenz wird somit angegeben, wie häufig eine Regel korrekt ist.

Zur Auswahl einer interessanten Regel ist das Betrachten von sowohl Support als auch Konfidenz relevant. Bei Betrachten des Supports als einzigen Wert besteht die Gefahr, dass Regeln, welche häufig korrekt sind, aber selten auftreten, nicht als interessant erkannt werden. Trotzdem müssen auch Regeln, welche sich durch einen hohen Konfidenzwert und einen niedrigen Supportwert auszeichnen mit Vorsicht behandelt werden. Der hohe Konfidenzwert kann bei einem geringen Support ein Indiz für einen „Zufallstreffer“ sein.

2.2.2 Varianten

Im Folgenden werden zwei Varianten der Sequenzregeln vorgestellt, es sind sowohl vollständig sortierte Sequenzregeln als auch partiell sortierte Sequenzregeln bekannt (Fournier Viger, C.-W. Wu, Tseng und Nkambou 2012).

Vollständig sortierte Sequenzregeln
Diese Variante wird auch „sequential rules between sequential patterns“ genannt. Sowohl die linke und rechte Seite der Regel ist jeweils ein vollständig sortiertes Sequenzmuster.

Das Betrachten der Sequenzen 4 bis 6 in Abbildung Abbildung 2.1 erzeugt unter anderem folgende drei verschiedene Sequenzregeln, welche sehr spezifisch sind, da sie jeweils nur für eine Sequenz gelten:

$$\{B, D\}, \{E\} \rightarrow \{A\}$$
$$\{E\}, \{D\}, \{B\} \rightarrow \{A\}$$
$$\{B, D, E\}, \rightarrow \{A\}$$

Zusätzlich können noch weitere, recht allgemeine Sequenzregeln gebildet werden, wie etwa die Folgende:

$$\{B\} \rightarrow \{A\}$$

Angenommen, es tritt eine neue Sequenz auf, deren weiterer Verlauf anhand der vorhandenen Sequenzregeln vorhergesagt werden soll. Diese Sequenz ist $\{B\}, \{D, E\}$. Diese Sequenz den Mustern ähnelt den Mustern, welche sich auf der linken Seite der drei spezifischen Regeln befinden. Dennoch passt sie zu keiner der Regeln. Stattdessen kann nur die allgemeine Regel $\{B\} \rightarrow \{A\}$ verwendet werden. Die Verwendung von ausschließlich allgemeinen Regeln führt aber dazu, dass verschiedene Ausgangssituationen nicht hinreichend differenziert werden können. So ist möglicherweise für das Auftreten der neuen Sequenz $\{C\}, \{B\}$, welche mit $\{C\}$ ein komplett neues Element enthält, die Regel $\{B\} \rightarrow \{A\}$ möglicherweise keine ausreichende Unterscheidung mehr.

Partiell sortierte Sequenzregeln
In (Fournier Viger, C.-W. Wu, Tseng, Cao u. a. 2015) wird das Problem der vollständig sortierten Sequenzregeln erkannt: Die große Menge an erstellten Regeln sind häufig zu spezifisch, um sie für Vorhersagen von neuen Sequenzen nutzen zu können.

Deswegen wurde von Fournier-Viger et al. eine andere Definiton der Sequenzregeln entwickelt. Diese werden als partiell sortierte Sequenzregeln oder auch „common to several sequences“ bezeichnet.

Partiell sortierte Sequenzregeln zeichnen sich durch folgende Eigenschaften aus:

- Die Sequenzrelation zwischen linker und rechter Seite der Regel bleibt erhalten – Die rechte Seite folgt weiterhin der linken Seite.

- Die Unterteilung der linken und rechten Seite in jeweils einzelne Ereignisse wird aufgehoben – Alle Elemente stehen innerhalb eines geschweiften Klammer {}.

- Die Muster, welche sich auf der linken und rechten Seite der Regel befinden, sind nicht sortiert.

- Linke und rechte Seite der Regel sind immer disjunkte Mengen.

Ein erneutes Betrachten der Sequenzen 4 bis 6 aus Abbildung 2.1 zeigt folgendes Muster:
Nach Auftreten von B, D und E in beliebiger Reihenfolge folgt danach A.

Diese drei Sequenzen lassen sich alle durch die folgende partiell sortierte Sequenzregel beschreiben:

$$\{B, D, E\} \rightarrow \{A\}$$

Untersuchungen zeigen, dass partiell sortierte Sequenzregeln bei den betrachteten Datensätzen stets bessere Vorhersageergebnisse erzielte als die vollständig sortierten Sequenzregeln (Fournier Viger, C.-W. Wu, Tseng, Cao u. a. 2015).

2.2.3 Anwendungen

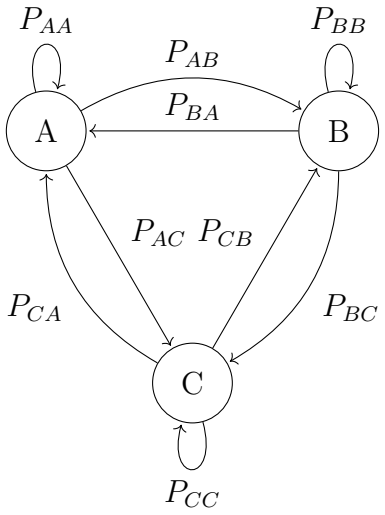
Sequenzregeln wurden eingesetzt um Kursverläufe an der Börse (Yang, Hsieh und J. Wu 2006) oder Besuche von Webseiten vorherzusagen (Singh, M. Kaur und P. Kaur 2017). Ein weitere Anwendung der Sequenzregeln ist es, Abhängigkeiten zwischen verschiedenen Sensormessswerten eines Gebäude-Automatisierungssystems zu finden (Stinner u. a. 2019).

2.2.4 Implementationen

Mit dem bereits erwähnten Paket `arulesSequence` der Programmiersprache R können mit der Funktion `ruleInduction` aus vorher erstellten Sequenzmustern vollständig sortierte Sequenzregeln generiert werden.

SPMF (Fournier-Viger, Lin u. a. 2016), was sowohl als eigenständiges Programm als auch als Java-Softwarebibliothek verwendet werden kann, implementiert verschiedene Data-Mining-Algorithmen. Es können aus einer Transaktionsdatenbank sowohl vollständig sortierte (`RuleGen`-Algorithmus) als auch partiell sortierte Sequenzregeln (u.a. `TRuleGrowth`-Algorithmus) erstellt werden.

2.3 Markov-Ketten



(a) Darstellung als Graph

	A	B	C
A	P_{AA}	P_{BA}	P_{CA}
B	P_{AB}	P_{BB}	P_{CB}
C	P_{AC}	P_{BC}	P_{CC}

(b) Übergangsmatrix

Abbildung 2.2: Beispiel Markov-Kette

Eine Markov-Kette ist eine weitere Möglichkeit, eine Vorhersage aus sequenziellen Daten zu erzeugen. Die Markov-Kette ist eine Beschreibung eines Zufallsprozesses. Eine übliche Darstellungsform ist ein Diagramm, wie in Abbildung 2.2a dargestellt. Die gezeigte Markov-Kette besteht aus den Zuständen (A, B, C). Der Prozess befindet sich immer in einem Zustand. Es ist ein Übergang in andere Zustände möglich. Es werden sowohl zwischen verschiedenen Zuständen Übergangswahrscheinlichkeiten (u.a. P_{BA}, P_{AC}) angegeben, als auch Wahrscheinlichkeiten, im gleichen Zustand zu bleiben (u.a. P_{AA}, P_{BB}). Die Wahrscheinlichkeiten können in einer Übergangsmatrix angegeben werden (siehe: Abbildung 2.2b).

Um Markov-Ketten zur Vorhersage zu nutzen, wird ein Trainingsdatensatz eingesetzt, um die Übergangswahrscheinlichkeiten bestimmen zu können. Die in Abbildung 2.2 dargestellte Form besitzt kein Gedächtnis. Dies bedeutet, dass der nächste Zustand nur vom derzeitigen Zustand abhängig ist.

Bei Verwendung von Markov-Ketten zur Vorhersage sequenzieller Daten führt die Betrachtung der vorherigen Zustände zu genaueren Vorhersagen (Rosvall u. a. 2014), (Chierichetti u. a. 2012). Dazu werden Markov-Ketten höherer beziehungsweise n-ter Ordnung benötigt. Die Ordnung beschreibt, von wie vielen Zuständen die Vorhersage abhängig ist. Abbildung 2.2 zeigt eine Markov-Kette erster Ordnung, da sie insgesamt

von einem Zustand abhängig ist.

Wenn das gezeigte Beispiel stattdessen eine Markov-Kette zweiter Ordnung wäre, dann wäre die Vorhersage vom derzeitigen Zustand und vom vorherigen Zustand abhängig. Dies wird erreicht, in dem Zustände erzeugt werden, welche eine Aneinanderreihung der Zustände der Markov-Kette erster Ordnung sind. Die Zustände sind in diesem Fall alle Variationen von $\{A, B, C\}$ mit einer Länge von zwei: $(\{AA\}, \{AB\}, \{AC\}, \{BA\} \ldots)$. Bei Markov-Ketten höherer Ordnung steigt die Anzahl der benötigten Zustände und die benötigte Rechenleistung somit schnell an (Deshpande und Karypis 2004).

2.3.1 Anwendungen

Markov-Ketten können eingesesetzt werden, um algorithmisch Texte oder Musik zu generieren (McAlpine, Miranda und Hoggar 1999). Eine weitere Einsatzmöglichkeit ist die Vorhersage des Klickpfades von Nutzern einer Webseite (Sarukkai 2000). Die Vorhersage des Klickpfades konnte bessere Ergebnisse erzielen, in dem mehrere Übergangsmatrizen aus verschiedenen Zeiträumen erstellt wurden, wobei neuere stärker gewichtet werden (Jayalal, Hawksley und Brereton 2007).

2.3.2 Implementierungen

Es existiert unter anderem die Python-Bibliothek `mchmm`, welche genutzt werden kann, um aus einer gegebenen Sequenz eine Übergangsmatrix mit den zugehörigen Wahrscheinlichkeiten zu erstellen (Terpilowski 2021).

2.4 Künstliche Neuronale Netze

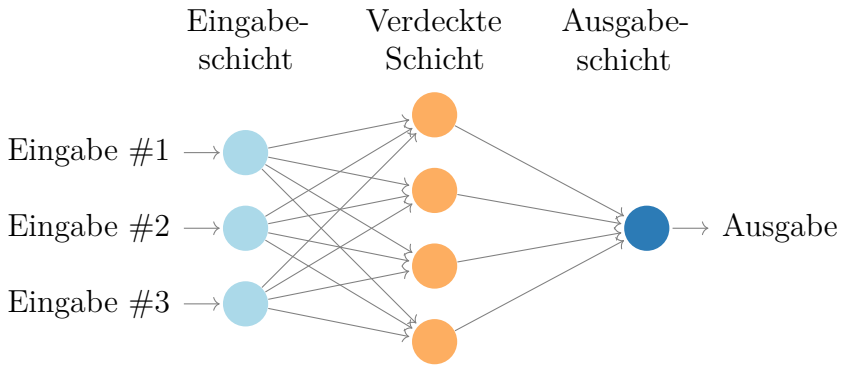


Abbildung 2.3: Schematische Darstellung eines Künstlichen Neuronalen Netzes

Ein künstliches Neuronales Netz (KNN) besteht aus künstlichen Neuronen, welche untereinander verbunden sind. In Abbildung 2.3 ist ein einfaches Feedforward-Netz dargestellt. Dies bedeutet, dass die Ausgaben der Neuronen ausschließlich in eine Richtung weitergeführt werden, eine Rückkopplung findet in dem Fall nicht statt. Die Neuronen sind in einzelnen Schichten angeordnet. Die dargestellte Eingabeschicht kann drei verschiedene Eingaben verarbeiten. Häufig werden als Eingaben normierte Zahlenwerte zwischen 0 und 1 verwendet.

Es existieren Verbindungen beziehungsweise Kanten zwischen den Neuronen. Jeder Verbindung wird eine bestimmte Gewichtung zugeordnet. Das Netz lernt, in dem die Gewichte der Verbindungen angepasst werden. Dazu sind Trainingsdaten notwendig.

Die Entwicklung von Trainingsalgorithmen, welche auf leistungsfähigen Grafikprozessoren ausgeführt werden können, erlaubt die Verwendung sehr großer Datenmengen als Trainingsdaten (Paine u. a. 2013).

Die Ausgabe der einzelnen Neuronen wird von Schicht zu Schicht bis zu einer Ausgabeschicht geleitet. An dieser kann das Ergebnis observiert werden.

Die Zustände und Gewichtungen innerhalb eines KNN sind für den Menschen schwer interpretierbar, insbesondere wenn deutlich mehr Neuronen und Schichten als in dem gezeigten Beispiel vorhanden sind. Ein fertig trainiertes KNN erzeugt zu einer Eingabe eine Ausgabe, es ist jedoch schlecht nachzuvollziehen, wie die Ausgabe

erzeugt wurde. Dieses Verhalten wird oft als Black-Box-Verhalten bezeichnet. Es gibt Bestrebungen den inneren Zustand der KNN durch Visualisierungen verständlicher darzustellen (Murdoch und Szlam 2017), diese befinden sich jedoch noch in einem recht frühen Stadium.

Es existieren viele Arten der KNN. Zur Vorhersage sequenzieller Daten werden häufig Long short-term memory („langes Kurzzeitgedächtnis“) Netze eingesetzt. Erstmals in (Hochreiter und Schmidhuber 1997) vorgestellt, ist es heutzutage ein weit verbreitetes Verfahren. Im Gegensatz zum Feedforward-Netz in Abbildung 2.3 nutzt ein Long short-term memory Netz zusätzlich Neuronenverbindungen, welche entgegen der Verabeitungsrichtung verlaufen, um bessere Vorhersageergebnisse erzielen zu können.

2.4.1 Anwendungen

Im letzten Jahrzehnt wurde viel an KNN geforscht, so steigt die Anzahl der Publikationen, welche sich auf KNN beziehen, im letzten Jahrzehnt jährlich an[1].

So wird beispielsweise ein Long short-term memory Netz eingesetzt, um Kunden- und Einkaufsverhalten im Versandhandel zu analysieren, das gezeigte Vorgehen ist besonders geeignet für große Datenmengen (Jamshed, Mallick und Kumar 2020). Auch die Handschrifterkennung (Graves u. a. 2009) oder die Spracherkennung (Sak, Senior und Beaufays 2014) sind häufige Anwendungsgebiete. In (Park u. a. 2018) wird ein KNN genutzt um die Fahrbahn von Fahrzeugen vorherzusagen.

2.4.2 Implementierungen

Es existieren viele Softwarepakete um KNN nutzen zu können. Populäre Beispiele sind `scikit-lern` (Pedregosa u. a. 2011) und `TensorFlow` (Martín Abadi u. a. 2015).

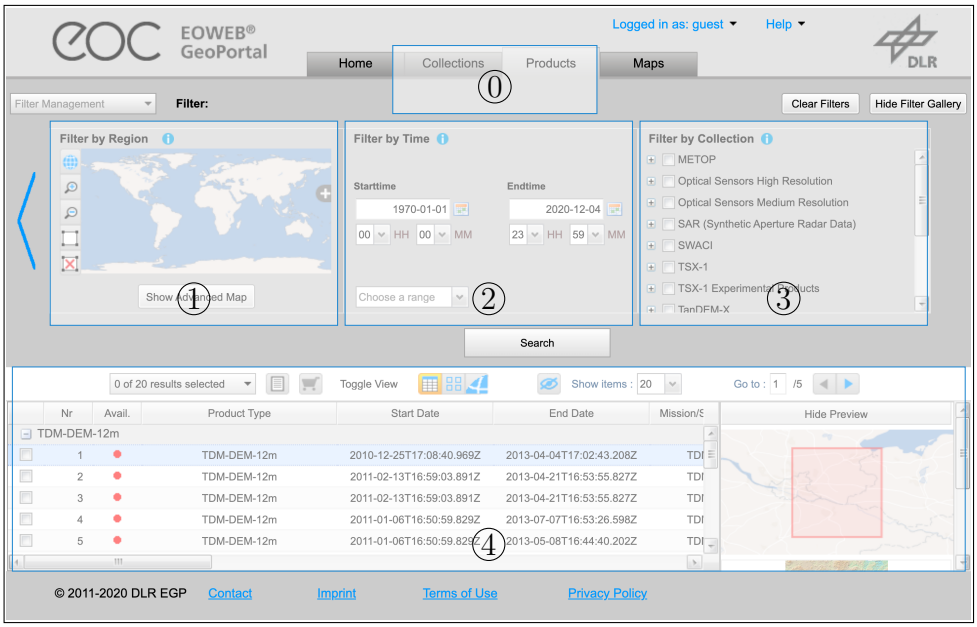
[1] Anhand der Suchfunktion der `https://www.sciencedirect.com` Datenbank bestimmt

2.5 EOWEB®GeoPortal

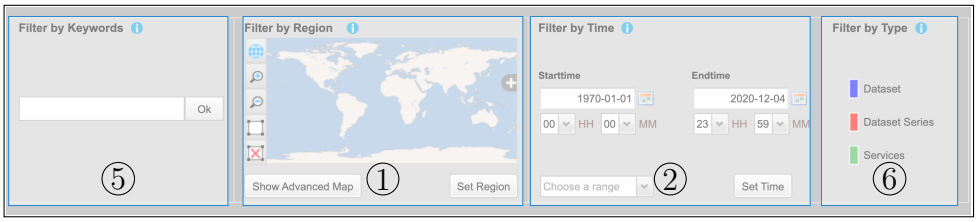
Das EOWEB® GeoPortal (EGP) ist ein Datenportal des Deutschen Zentrums für Luft- und Raumfahrt e.V. Auf diesem Datenportal stehen Erdbeobachtungsdaten aus dem Deutschen Satellitendaten-Archiv zur Verfügung (Deutsches Zentrum für Luft- und Raumfahrt e.V. 2020). Das Portal ist online[2] verfügbar.

Um einen Datensatz zu erwerben, durchläuft ein Nutzer üblicherweise mehrere Schritte. Zuerst erfolgt die Anmeldung auf dem Datenportal. Die Verwendung eines Gastzugangs ist ebenso möglich. Im nächsten Schritt existieren Such- und Filterfunktionen, um den Nutzer beim Auffinden von Datensätzen zu unterstützen. In der Ergebnisliste erscheinen Einträge mit einer Vorschau der Ergebnisse. Falls ein passender Datensatz gefunden wurde, kann dieser erworben und anschließend heruntergeladen werden. Dazu wird auf ein weiteres Portal weitergeleitet.

[2] https://eoweb.dlr.de



(a) *Products*-Ansicht



(b) Suchoptionen der *Collections*-Ansicht

Abbildung 2.4: EOWEB Benutzeroberfläche.
⓪ Tab um zwischen Ansichten zu wechseln
① Räumlicher Filter
② Zeitlicher Filter
③ Kollektionsfilter
④ Ergebnisliste
⑤ Stichwortsuche
⑥ Filterung nach Typ

Die Benutzeroberfläche des EGP ist in der Abbildung 2.4 dargestellt. Die angebotenen Beobachtungsdaten sind in Kollektionen unterteilt. Eine Kollektion fasst verschiedene, ähnliche Beobachtungen zusammen. Ein Beispiel für eine Kollektion ist *METOP GOME-2 - Nitrogen Dioxide (NO2) - Global*. Dabei handelt es sich um Aufzeichnungen der METOP Satelliten. Gemessen wurde die Stickstoffdioxid-Konzentration weltweit. Die Kollektionen sind mit Metadaten versehen. Mögliche Informationen können unter anderem eine Kurzbeschreibung, Auflösung der Messwerte oder die Nennung eines Ansprechpartners sein.

Eine Kollektion enthält Produkte. Im Fall der *METOP GOME-2* Kollektion ist ein Produkt ein Datensatz zu einem konkreten Zeitpunkt. Eine Vorschau eines *METOP GOME-2* Datensatzes wird in Abbildung 2.5 gezeigt.

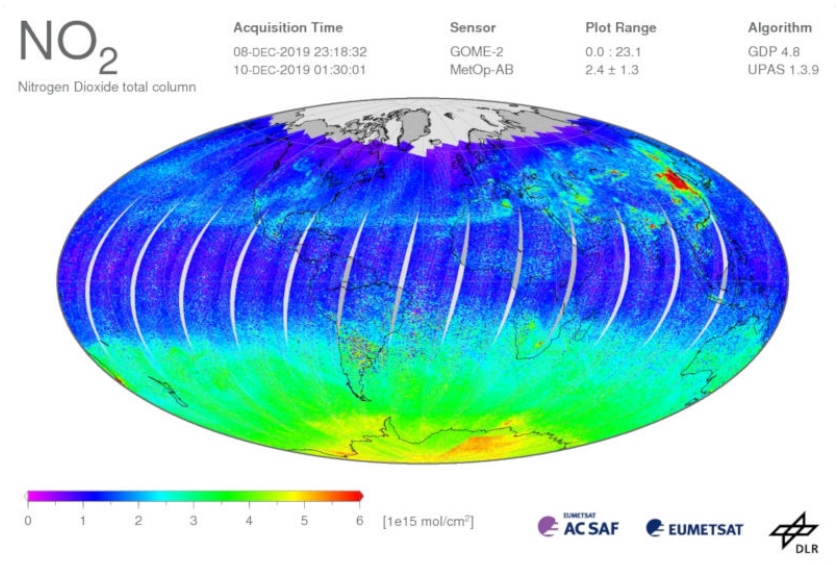


Abbildung 2.5: Vorschau der Stickstoffdioxid-Messwerte des METOP-Satelliten. Quelle: (DLR EGP 2020)

Die Benutzeroberfläche wird anhand der Anmerkungen in Abbildung 2.4 erläutert, sie lässt sich in Menü, Suchelemente und Ergebnisliste unterteilen.

Über das Menü lässt sich zwischen zwei Ansichten umschalten. Je nachdem, ob der Nutzer entweder nach Kollektionen oder nach Produkten sucht, stehen verschiedene Ansichten zur Verfügung, zwischen denen über Tabs (⓪) in der Kopfzeile umgeschaltet werden kann. Die Ansichten unterscheiden sich in der Anzeige der Ergebnisse und in den verfügbaren Suchfiltern.

Folgende Suchfilter werden im EGP angeboten:

- **Räumlich** (①) Der räumliche Filter ermöglicht die Spezifizierung der Region, zu der Erdbeobachtungsdaten abgerufen werden sollen. Dies kann durch Zeichnen eines Begrenzungsrahmens auf einer Weltkarte erfolgen, durch Auswahl eines Ortes aus einer vordefinierten Liste oder durch Hochladen eines *shape file*, welches eine räumliche Einschränkung definiert.

- **Zeitlich** (②) Durch den zeitlichen Filter kann die Auswahl auf Beobach-

tungsdaten eingeschränkt werden, die innerhalb eines bestimmten Zeitfensters aufgenommen wurden. Es besteht sowohl die Möglichkeit vorgeschlagene Intervalle wie *Letzter Monat* zu verwenden, als auch individuell ein Start- sowie Enddatum festzulegen.

- **Bestandteil einer Kollektion** (③) Dieser Filter legt fest, zu welcher Kollektion die angezeigten Produkte gehören. Eine Auswahl von mehreren Kollektionen zur gleichen Zeit wird unterstützt.

- **Eingabe eines Schlüsselworts** (⑤) Dieser Filter ermöglicht eine freie Suche in den Metadaten der Kollektionen. Es existiert eine Suchvervollständigung. Bei Eingabe eines Suchbegriffes werden Vorschläge angezeigt.

- **Typ** (⑥) Der Typ einer Kollektion kann drei verschiedene Werte annehmen. Ein *Datensatz* enthält ein einzelnes Produkt, eine *Datensatzserie* hingegen mehrere zusammenhängende Aufzeichnungen. Dies kann etwa die Betrachtung des gleichen Messwertes zu verschiedenen Zeitpunkten sein. Die dritte Möglichkeit, ein *Service*, enthält keine Produkte, sondern bietet Zugriff auf einen Webservice an, der weitere Daten zur Verfügung stellt. Es können verschiedene Kollektionstypen zeitgleich gewählt werden.

Die *Collections*-Ansicht (siehe: Abbildung 2.4b) zeigt in der Ergebnisliste ausschließlich Kollektionen an. Diese Ansicht ermöglicht die Verwendung der räumlichen und zeitlichen Suchfilter, die Nutzung der Schlüsselwortsuche oder die Wahl des Typs.

Bei Verwendung der *Products*-Ansicht hingegen (siehe: Abbildung 2.4a) enthält die Ergebnisliste nur Einträge vom Typ Produkt. Die Schlüsselwortsuche und Typfilterung ist in dieser Ansicht nicht verfügbar, stattdessen ist eine Filterung nach Zugehörigkeit zu einer Kollektion möglich.

Die Ergebnisliste (④) verwendet Paginierung. Bei vielen Suchresultaten werden die Ergebnisse auf mehrere Seiten aufgeteilt, der Nutzer kann die angezeigte Seite und die Anzahl der Ergebnisse pro Seite anpassen.

Eine Analyse der EOWEB-Logdaten und damit der gleiche Datensatz, welcher in dieser Arbeit verwendet wurde, wurde in (Schindler, Paradies und Twele 2019) durchgeführt. Darin wurde eine Übersicht über die Verwendungshäufigkeit der einzelnen

Filter erstellt. Es wurden die Eingaben der Nutzer in den Suchfeldern detailliert analysiert. Dazu erfolgte eine Einteilung der Bedeutung der Suchbegriffe in einzelne Klassen.

Die Autoren konnten feststellen, dass die vorhandenen Begriffe der Suchvervollständigung oft nicht ausreichend waren, um die Intentionen der Nutzer abzubilden. Außerdem wurde gezeigt, dass in der vorhandenen Stichwortsuche Schreibfehler oder alternative Schreibweisen eines Begriffes selten zum Erfolg führten. Es wurden Vorschläge formuliert, um die Suchleistung in Zukunft verbessern zu können. Dazu gehört die Verwendung eines Stemming-Algorithmus, um abweichende Schreibweisen eines Wortes zu erkennen und die Verwendung einer OpenStreetMaps-Schnittstelle, um lokalisierte Ortsbezeichnungen nutzen zu können.

Die durchgeführten Analysen behandelten die Suchanfragen als einzelne Ereignisse. Eine Betrachtung der Anfragen als Sequenz in einer Sitzung eines Nutzers fand nicht statt. Eine Vorhersage zukünftiger Anfragen, wie es Bestandteil dieser Arbeit ist, wird nicht durchgeführt.

2.6 Benutzermodelle

Ein Benutzermodell ist eine synthetische Repräsentation einer Person. Das Modell wird durch verschiedene Attribute (ein Schema) bestimmt (Thierry 2005). Ein Benutzermodell kann als *offen* oder auch als *geschlossen* klassifiziert werden. Offen bedeutet hierbei, dass der Nutzer einsehen kann, welche Informationen über ihn existieren. Zusätzlich ist ein gezieltes, aktives Anpassen des Modells direkt durch den Nutzer möglich. Ein Beispiel ist in Abbildung 2.6 zu sehen. Dort dargestellt ist *Grapevine* (Rahdari, Brusilovsky und Babichenko 2020). Grapevine ist ein System zum Finden von Forschungsberatern, in dem der Anwender durch verschiedene Slider seine Interessen spezifizieren kann.

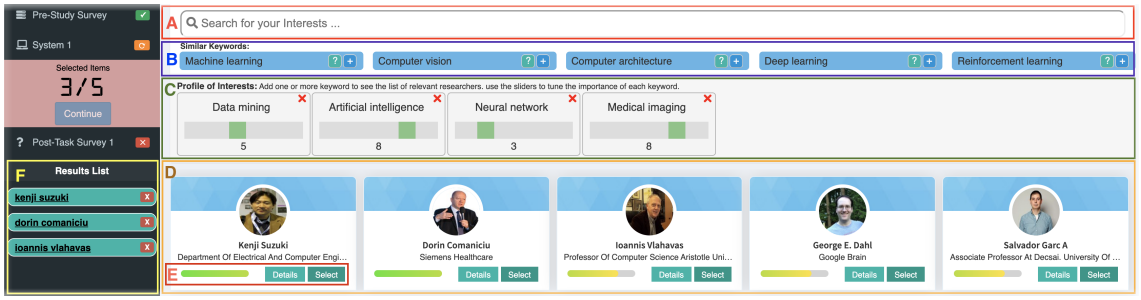


Abbildung 2.6: Beispiel eines Systems mit offenem Nutzermodell.
A: Stichworteingabe, B: Verwandte Stichworte, C: Interessen-Slider, D: Ergebnisse
Quelle: nach Rahdari, Brusilovsky und Babichenko 2020, S. 4

Ein offenes Benutzermodell erhöht die Transparenz für den Nutzer und erlaubt eine gewisse Kontrolle über die Adaption (Frasincar, Borsje und Levering 2009). Dies ist allerdings auch nicht ohne Risiken. Es ist durchaus möglich, dass ein geringes Vorwissen der Nutzer dazu führt, dass irreführende Pfade eingeschlagen werden (Ahn u. a. 2007). Ein offenes Benutzermodell kann zur Bestimmung der Risikobereitschaft von Nutzern im Kontext der mobilen Datennutzung (Molnar und Muntean 2019) verwendet werden. Eine weitere mögliche Nutzung ist der Einsatz im interaktiven Schulunterricht (Molnar, Virseda und Frias-Martinez 2015). Ein geschlossenes Nutzermodell hingegen bietet keine Möglichkeit der aktiven Anpassung durch den Nutzer. Der Nutzer beeinflusst die Adaption zwar durch seine Interaktionen, es besteht aber keine Möglichkeit der Beeinflussung durch ein zur Verfügung gestelltes Bedienelement.

Anhand eines Benutzermodells können drei grundlegende Arten der Personalisierung (Molnar und Muntean 2019) durchgeführt werden:

- *Personalisierung anhand des Nutzers* Die Ausgabe passt sich den vorherigen Aktionen der Nutzer an. Alternativ können Informationen von anderen Systemen importiert werden (Wongchokprasitti u. a. 2015).

- *Personalisierung anhand des Geräts* Die Ausgabe passt sich dem verwendeten Gerät an. So kann die Darstellung von Webseiten abhängig von der Bildschirmauflösung des Endgeräts sein (Anam, Ho und Lim 2014).

- *Personalisierung anhand des Kontexts* Die Ausgabe passt sich an den Kontext des Nutzers an. Dies kann zum Beispiel der derzeitige Standort sein.

3 Konzept

In diesem Kapitel wird das Konzept beschrieben, um eine Vorhersage von Benutzerinteraktionen durchzuführen. Um eine Vorhersage zu treffen, wird ein Verfahren namens Kollaboratives Filtern angewendet. Dabei werden vergangene Interaktionen vieler anderer Nutzer verwendet, um die Interaktionen eines neuen Nutzers vorherzusagen (Resnick und Varian 1997). Kollaboratives Filtern arbeitet nach folgendem Prinzip (Hansen 2008):

1. Aus einem vorliegenden Trainingsdatensatz, welcher eine Sammlung vergangener Nutzerinteraktionen enthält, werden Muster extrahiert und gespeichert.

2. Anhand der bisherigen Interaktionen eines neues Nutzers wird ein vergleichbares Muster aus dem Speicher gesucht.

3. Die folgende Interaktion des Vergleichsmusters wird als Vorhersage gewählt.

Kollaboratives Filtern wird bereits vielfach eingesetzt, etwa um anhand der Interessen des Nutzers News-Artikel zu zeigen (Konstan u. a. 1997) oder um bei einem Streamingdienst Musiktitel zu empfehlen (Baer 2015). Eine Einschränkung dieses Verfahren ist jedoch, dass die Empfehlung, welche von der Auswahl der anderen Nutzer abhängig ist, zwar eine beliebte, aber nicht zwingend die optimale Wahl ist. Ein großer Vorteil dieses Verfahrens ist, dass es einfach möglich ist, an einen großen Datensatz zu gelangen. Es müssen keine Umfragen durchgeführt werden, stattdessen können Informationen aus der üblichen Benutzung des Systems gewonnen werden.

Um die Vorhersage durchzuführen wird ein Benutzermodell trainiert. Die Personalisierung findet anhand der bisherigen Interaktionen dieses Nutzers statt. Das Bennutzermodell ist geschlossen: Der Nutzer kann nicht aktiv die persönlichen Präferenzen festlegen.

Das Konzept beschränkt die Form des zu verarbeitenden Datensatzes auf ein Format, welches häufig verwendet wird, um Nutzerinteraktionen abzubilden. Als Quelle des Datensatzes können etwa Logdateien dienen, wie sie zum Beispiel von Serveranwendungen erstellt werden. Um diese Art des Datensatzes genauer zu definieren, werden die Anforderungen, die durch den Datensatz erfüllt sein müssen, im Folgenden beschrieben. Danach wird ein Überblick über das Vorhersageverfahren gegeben, anschließend wird die Wahl des Vorhersagealgorithmus begründet. Später wird vorgestellt, wie aus den Feldern des Datensatzes Elemente für die Vorhersage gewonnen werden können.

Das Konzept wurde in einem iterativen Prozess entwickelt. Dazu wurden unter Anwendung des Konzepts die Vorhersageergebnisse stets übeprüft. Dies geschah einerseits anhand eines als „Zaki“ bezeichneten Datensatz (Zaki 2001), welcher oft zur Demonstration von Sequenzmustern und Sequenzregeln verwendet wird. Andererseits anhand des EGP-Datensatzes, welcher Aufzeichnungen über Nutzerinteraktionen eines Geodatenportals enthält und in Kapitel 4 vorgestellt wird. Anschließend flossen die so gewonnen Erkenntnisse in die Aktualisierung des Konzepts ein.

3.1 Anforderungen an den Datensatz

Das in diesem Konzept vorgestellte Vorhersageverfahren ist für verschiedene Datensätze anwendbar. Im Folgenden werden abstrakte Anforderungen an den verwendeten Datensatz vorgestellt, damit eine Vorhersage nach dem vorgestellten Konzept möglich ist. Dazu wird eine Klasse an Datensätzen definiert, für die das vorgestellte Konzept verwendet werden kann. Ziel dieser Definition ist es, allgemein genug zu sein, so dass eine große Menge an Datensätzen existiert, die für dieses Verfahren verwendbar sind. Gleichzeitig soll die Definition spezifisch genug sein, so dass alle Datensätze dieser Klasse mit wenigen Anpassungen verwendbar sind.

Diese Klasse an Datensätzen wird bestimmt durch gewisse Anforderungen an den Inhalt des Datensatzes. Um die Allgemeinheit der Definition zu unterstreichen, erfolgt bei Beschreibung der Anforderungen eine Zuordnung zu Beispielen aus verschiedenen Situationen: Die Suche auf einer Webseite (①), Einkäufe im Geschäft (②) und das Vorliegen einer Medikamentenhistorie (③).

- Der Datensatz muss eine Menge aus Sitzungen enthalten. Eine Sitzung kann etwa eine Sitzung auf einer Webseite (①), der Einkaufsverlauf eines Kunden (②) oder die Medikamentenhistorie eines Patienten (③) sein.

- Die Sitzungen bestehen aus Interaktionen. Die Interaktionen sind zeitlich geordnet. Eine Interaktion kann beispielsweise das Starten einer Suchanfrage (①), ein einzelner Einkauf (②) oder die Verordnung eines Medikaments (③) sein.

- Interaktionen enthalten eine unsortierte Menge an Feldern. Diese Felder können beispielsweise die verwendeten Suchfilter (①), die gekauften Artikel eines Einkaufs (②), oder die verschriebenen Medikamente eines Arztbesuchs (③) sein. Dies ist eine wichtige Unterscheidung, da viele Datensätze nur ein Feld pro Interaktion enthalten können, zum Beispiel der Klickpfad von Nutzern auf einer Webseite. Auch einige Vorhersageverfahren sind nur auf ein Feld pro Interaktion ausgelegt sind, wie etwa `CPT` (T. Gueniche, Fournier Viger und Tseng 2013).

- Es existiert eine beschränkte Menge an möglichen Werten für ein Feld. Eine Möglichkeit ist es, zu unterscheiden ob ein Feld einen Inhalt hat oder alternativ leer ist (zwei mögliche Werte). Falls der Inhalt des Feldes relevant ist und für die Vorhersage betrachtet werden soll, können Strategien wie Binning für Zahlenwerte oder das Zuordnen zu einzelnen Klassen bei Zeichenketten vorgenommen werden.

Falls die Quelle des Datensatzes eine Logdatei in einem unstrukturierten Textformat ist, muss vorher eine Zerlegung der relevanten Information der Logeinträge in einzelne Felder vorgenommen werden. Ein mögliches Vorgehen wäre das Zerlegen durch reguläre Ausdrücke und das Speichern in entweder einer JSON-Datei oder einer Datenbank.

3.2 Anforderungen an das Vorhersageverfahren

Abbildung 3.1: Vorhersageverfahren

Abbildung 3.1 zeigt in einer abstrakten Darstellung, welche Eingabe das Vorhersageverfahren erwartet und was die Ausgabe ist. Um eine Vorhersage durchzuführen zu können, erfolgt zuerst ein Training des Benutzermodells anhand eines Trainingsdatensatzes. Der Trainingsdatensatz muss den vorgestellten Anforderungen eines Datensatzes entsprechen. Weiterhin muss bereits eine Vorverarbeitung des Datensatzes vorgenommen wurden sein.

Nachdem das Modell trainiert wurde, kann eine Vorhersage von neuen, unbekannten Sitzungen vollzogen werden. Anhand des trainierten Benutzermodells kann entschieden werden, welche Interaktion am wahrscheinlichsten folgt. Diese wird dann als Vorhersage ausgegeben.

3.3 Auswahl des Vorhersageverfahrens

Im Kapitel „Stand der Forschung“ konnten verschieden Vorhersageverfahren anhand einer Literaturrecherche vorgestellt werden. Die vorgestellten Algorithmen erhalten nun eine Bewertung auf Eignung für die Vorhersage an einem Datensatz, welcher den vorgestellten Anforderungen entspricht. Es ist nicht als allgemeine Empfehlung zur Vorhersage von Daten zu werten.

Künstliche neuronale Netze

Aufgrund der intensiven Forschung zur Nutzung von künstlichen neuronalen Netzen (KNN) existieren für viele Anwendungsfälle Algorithmen, welche eine hohe Vorhersagegenauigkeit erreichen können. Ebenso eignen sich Neuronale Netze für sehr große Datenmengen, oft als Big-Data bezeichnet.

Eine grundlegende Herausforderung bei der Verwendung von KNN ist das Verhalten, welches sich als „Black Box“ beschreiben lässt. Das Durchführen von Optimierungen, um die Vorhersageergebnisse zu beeinflussen, ist komplex. Grund dafür ist die schlechte Interpretierbarkeit der Verbindungen der Neuronen mit den zugehörigen Gewichtungen. Weiterhin benötigt das Training viel Rechenleistung und die Komplexität einer korrekten Implementation ist hoch.

Aus diesen Gründen werden KNN in diesem Fall nicht zur Vorhersage verwendet.

Markov-Ketten

Ein wichtiger Schritt, um Markov-Ketten anwenden zu können, ist das Definieren der mögliche Zustände des Prozesses. Aus den Zuständen wird anschließend die Transaktionsmatrix abgeleitet. Um eine Vorhersage eines Datensatz des vorgestellten Formats durchzuführen können, werden bei Verwendung von Markov-Ketten eine große Anzahl an Zuständen benötigt. Dies wird an dem folgenden Beispiel erläutert.

Angenommen es soll abgebildet werden, dass eine Interaktion acht verschiedene Felder $A, B, \ldots H$ enthält. Basis der Vorhersage soll ausschließlich sein, ob ein Feld vorhanden (A) oder nicht vorhanden ($\bar{A}$) ist. Es müssen alle Kombinationen „von Feld vorhanden“ oder „Feld nicht vorhanden“ als eigene Zustände abgebildet werden: $< ABCDEFGH >, < \bar{A}BCDEFGH >, < A\bar{B}CDEFGH > \ldots < \bar{A}\bar{B}\bar{C}\bar{D}\bar{E}\bar{F}\bar{G}\bar{H} >$
Die Gesamtzahl der benötigten Zustände beträgt in diesem Fall:

$$2^8 = 256$$

Dies beschreibt jedoch nur die nötigen Zustände für eine Markov-Kette erster Ordnung. Bei einer Markov-Kette neunter Ordnung, welche notwendig ist, um acht Elemente aus der Vergangenheit in die Vorhersage einzubeziehen, tritt eine drastische

Steigerung der benötigten Zustände auf: Alle Variationen der Länge 9 müssen aus den 256 vorherigen Zuständen gebildet werden. Somit werden insgesamt deutlich mehr Zustände benötigt:

$$256^9 = 5e{+}21$$

Auch wenn für dieses Beispiel die Dimensionen noch recht gering gewählt wurden, werden somit so viele Zustände benötigt, dass keine praktikable Anwendung mehr möglich ist.

Es zeigt sich, dass Markov-Ketten ungeeignet sind für eine Vorhersage, welche abhängig von mehreren vorherigen Interaktion ist, in Kombination mit mehreren Feldern pro Interaktion. Genau dies ist aber erwartet und gefordert in dem gewählten Vorgehen, weshalb auch Markov-Ketten nicht eingesetzt werden.

Sequenzregeln

Da es sich um eine Vorhersageaufgabe handelt, werden Sequenzregeln gegenüber den ähnlichen Sequenzmustern bevorzugt. Der Einsatz von Sequenzregeln erfordert, dass die Eingabedaten als Transaktionsdatenbank vorliegen. Eine Transaktionsdatenbank besteht aus einzelnen Sequenzen, welche Ereignisse enthalten. Ein Ereignis wird durch eine Menge von Elementen bestimmt. In Tabelle 3.1 werden die Begriffe der Transaktionsdatenbank und die äquivalenten Begriffe aus den geforderten Elementen eines Datensatzes gegenübergestellt. Es zeigt sich, dass sich die Bestandteile eines Datensatzes, welcher den beschriebenen Anforderungen entspricht, gut auf die Bestandteile einer Transaktionsdatenbank abbilden lassen.

Transaktionsdatenbank	Datensatz
Sequenz	Sitzung
Ereignis	Interaktion
Element	Feld

Tabelle 3.1: Zuordnung der Begriffe der Transkationsdatenbank zu den Anforderung des Datensatzes

Die einfache Interpretierbarkeit der Regeln eignet sich auch zum Analysieren der Datensätze. Erkannte Muster können betrachtet werden, es erfolgt nicht nur die Ausgabe eines Vorhersageergebnisses.

Das Trainieren des Modells, in dem Fall das Erstellen der Sequenzregeln, nimmt verhältnisweise wenig Rechenleistung in Anspruch nimmt. Dieser Vorteil kommt insbesondere bei Verwendung mehrerer Hyperparameter zum Tragen, da der Optimierungsprozess deutlich verkürzt wird. Ein weiterer Vorteil der Sequenzregeln ist, dass sie partiell sortiert sein können. Somit müssen neue Muster nicht umbedingt dem trainierten Modell bekannt sein, sondern es wird eine Art Fuzzy-Logic beim Vergleich der Muster angewendet.

Aus diesen Gründen werden in dieser Arbeit Sequenzregeln eingesetzt um eine Vorhersage durchzuführen. Es existieren verschiedene Algorithmen um Sequenzregeln zu erstellen. Die Wahl des Algorithmus der Sequenzregelanalyse wird in Kapitel 5 beschrieben.

3.4 Definition der Elemente

In dem Datensatz sind Interaktionen enthalten, die aus mehreren Feldern bestehen. Eine Möglichkeit wäre, das Vorhandensein der Felder direkt als Elemente in der Transaktionsdatenbank zu verwenden. Ein mögliches Ereignis wäre bei Existenz der Felder A und B etwa $\{A, B\}$. Dieses Vorgehen kann jedoch Probleme verursachen:

- Es ist ausschließlich die Vorhersage von neu hinzugefügten Elementen möglich
- Durch eine geringe Anzahl an möglichen Elementen ist die Anzahl an Sequenzregeln gering, was die Vorhersagegenauigkeit negativ beeinflussen kann.

Stattdessen kann eine Transformation auf die Felder angewendet werden. Dabei wird der Unterschied zwischen den Feldern zwei aufeinanderfolgenden Logeinträgen als Element verwendet. Ein mögliches Ereignis wäre etwa $\{A_{\text{hinzugefügt}}, B_{\text{entfernt}}\}$. Mit diesem Vorgehen lässt sich auch das Entfernen von Elementen vorhersagen. Außerdem vergrößert sich durch die größere Anzahl an möglichen Elementen auch die Anzahl der Sequenzregeln, was zu einer erhöhten Vorhersagegenauigkeit führen kann.

Das Erzeugen der Elemente beeinflusst maßgeblich die Regebildung und somit auch das Vorhersageergebnis. Das genaue Vorgehen muss an den Datensatz angepasst werden.

4 Datensatz

Der Datensatz wurde aus Logdateien des EGP-Webservers erstellt. Es werden ausschließlich die Aufzeichnungen über getätigte Suchanfragen und durchgeführte Bestellungen in dieser Arbeit verwendet.

Ein öffentlicher Zugriff auf den Datensatz besteht aus Datenschutzgründen nicht. Er wurde von den EGP-Betreibern zur Durchführung der Analyse in dieser Arbeit zur Verfügung gestellt.

4.1 Inhalt

Der Inhalt des Datensatzes entspricht den im Konzept vorgestellten Anforderungen an einen Datensatz. Die EGP-Logdatei enthält Interaktionen. Eine Interaktion entspricht einer Anfrage des Nutzers an den Server. Jede Interaktion lässt sich einer Sitzung zuordnen.

Der vorliegende Datensatz enthält Interaktionen des Typs Suchanfrage und Bestellung. Eine Interaktion des Typs Suchanfrage ist in Listing 1 dargestellt. In diesem Beispiel setzte der Nutzer zwei Suchfilter, einen Filter auf die Kollektion (1-1) und einen zur Beschränkung der Zeit (1-2), ein. Ein Filter wird nicht immer nur durch ein Feld definiert, so wird etwa der zeitliche Filter aus zwei Feldern zusammengesetzt. Alle Suchfilter können am Präfix `search-` erkannt werden. Ein weiteres Feld (1-5) identifiziert diese Interaktion als Suchanfrage.

Um eine Zuordnung der Interaktionen zu Sitzungen zu ermöglichen, fügt der Server jeder Interaktion eine generierte Sitzungs-ID (1-4). hinzu[1]. Eine Sitzungs-ID ist

[1] Genauere Erläuterung in Unterabschnitt 5.1.1

gültig vom initialen Aufruf der Seite bis zu einem Überschreiten eines Zeitlimits bei Inaktivität des Nutzers. Dies ermöglicht es, in Verbindung mit dem zugehörigen Zeitstempel(1-3), eine Sitzung eines Nutzers zu rekonstruieren.

```
{
    1-1 "search-collection":
    ↪  ["urn:eop:DLR:EOWEB:TDM-CoSSC-Experimental",
    ↪  "urn:eop:DLR:EOWEB:TDM-CoSSC-DEM"],
    1-2 "search-temporalConstraint-start":
    ↪  "2011-10-20T00:00:00Z",
    1-3 "@timestamp": "2018-08-07T11:17:34.760Z",
    1-2 "search-temporalConstraint-stop": "2012-04-20T23:59:00Z",
    1-4 "sessionid": "C253267FCE4063C56FE033XXXXXXXXXX",
    1-5 "search": "true"
}
```

Listing 1: Ausschnitt aus Logdatei: Suchanfrage. Der tatsächliche Eintrag enthält mehr Felder als dargestellt.
1-1 Suchfilter Kollektion, 1-2 Suchfilter Zeit, 1-3 Zeitstempel, 1-4 Sitzungs-ID, 1-5 Typzuweisung

Von Nutzern durchgeführte Produktbestellungen werden ebenso aufgezeichnet. Ein Beispiel für den Logeintrag einer Bestellung wird in Listing 2 gezeigt. Enthalten sind 2-1 Informationen über bestellte Artikel und Art der Auslieferung, in diesem Fall ein Download über das SFTP-Protokoll. Ebenso wie bei den Suchanfragen werden 2-2 Zeitstempel, die 2-3 Sitzungs-ID und eine Typzuweisung 2-4 aufgezeichnet.

Die Bestellinformationen sind wertvolle Informationen, welche für zahlreiche Analysen eingesetzt werden können. So können Informationen über Bestellungen dazu verwendet werden, die Qualität der Suchergebnisse zu bewerten. Wenn ein Nutzer ein Produkt erwirbt, so ist das eine eindeutige Aussage darüber, dass die vorherige Suche als erfolgreich zu werten ist. Es existieren auch andere Methoden, um die Qualität von Suchergebnissen zu bewerten, etwa die Seitenverweildauer (Kim u. a. 2014).

Die Aufzeichnungen der Bestellungen können auch dazu dienen, um typische Interaktionsmuster von Bots zu identifizieren. Dies können zum Beispiel Crawler von Suchmaschinen sein. Das Vorhandensein einer kostenpflichtigen Bestellung in einer

Sitzung des EGP ist mit hoher Wahrscheinlichkeit ein Ausschlusskriterium für Bots[2]. Daraus folgt nicht, dass eine Bestellung eine notwendige Bedingung für einen menschlichen Nutzer ist. Auch sind nicht sämtliche Unterschiede im Verhalten zwischen bestellenden und nicht bestellenden Kunden durch Bots zu erklären. Es kann jedoch als ein mögliches Indiz verwendet werden, um zu identifizieren, ob bestimmte Muster durch Bots ausgelöst werden.

In den vorliegenden Daten sind nicht alle Bestellungen gelistet, tatsächlich ist der Anteil an Sitzungen mit Bestellungen höher. Es existieren alternative Kanäle über E-Mail oder Telefon, über die Nutzer eine Bestellung aufgeben können, diese treten jedoch nicht in den vorliegenden Logdateien auf.

```
{
    (2-1)"order-details": {
        "deliveryOption": [
            {
                "deliveryMethod": "sftp"
            }
        ],
        "productType": "TSX-1.SAR.L1b-StripMap-L0"
    },
    (2-2)"@timestamp": "2019-03-27T09:47:29.394Z",
    (2-3)"sessionid": "3E365A42457E57D7562472XXXXXXXXXX",
    (2-4)"order": "true"
}
```

Listing 2: Ausschnitt aus Logdatei: Bestellung. Der tatsächliche Eintrag enthält mehr Felder als dargestellt.
(2-1) Bestellinformationen, (2-2) Zeitstempel, (2-3) Sitzungs-ID, (2-4) Typzuweisung

[2] In anderen Onlineshops mit begrenzt verfügbaren und begehrten Artikeln ist es durchaus möglich, dass Bots zum Kauf von Produkten eingesetzt werden

4.2 Merkmale des Datensatzes

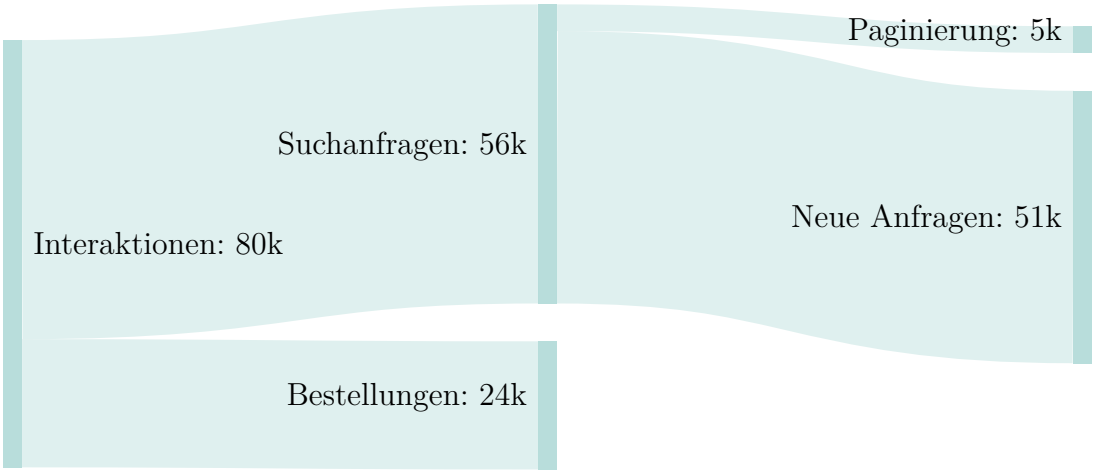


Abbildung 4.1: Verteilung der Interaktionstypen

Der vorliegende Datensatz wurde in einem Zeitraum beginnend im August 2018 bis zum Mai 2019 aufgezeichnet. Im Datensatz sind über 80 000 Interaktionen aus insgesamt 8 740 Sitzungen enthalten. Somit können im EGP täglich durchschnittlich knapp 30 neue Sitzungen verzeichnet werden. Jede Sitzung enthält im Durchschnitt zehn Interaktionen.

Interaktionen können in Suchanfragen oder Bestellung unterschieden werden. Der Großteil der Interaktionen ist, wie Abbildung 4.1 zeigt, eine Suchanfrage. Es ist eine weitere Unterteilung der Interaktionen vom Typ Suchanfrage möglich:

- Eine Paginierungs-Anfrage enthält die gleichen Filter mit den gleichen Werten wie die Suchanfrage davor. Der Nutzer navigierte nur die Ergebnis-Seiten oder passte die Anzahl der Ergebnisse auf einer Seite an (oder beides).

- Neue Anfragen hingegen zeichnen sich durch geänderte Filter im Vergleich zur vorherigen Suchanfrage aus.

Paginierung tritt deutlich seltener auf als neue Anfragen. Einerseits liegt es daran, dasss eine Anwendung der Suchfilter die Ergebnisse deutlich einschränkt und nur noch eine Seite vorhanden ist. Weiterhin ist es ein oft untersuchtes Phänomen von Suchmaschinen, dass Paginierung nur selten verwendet wird: So wird bei der

Suchmaschine Google nur in etwa 5% der Fälle eine zweite Seite mit Ergebnissen aufgerufen (Petrescu 2014).

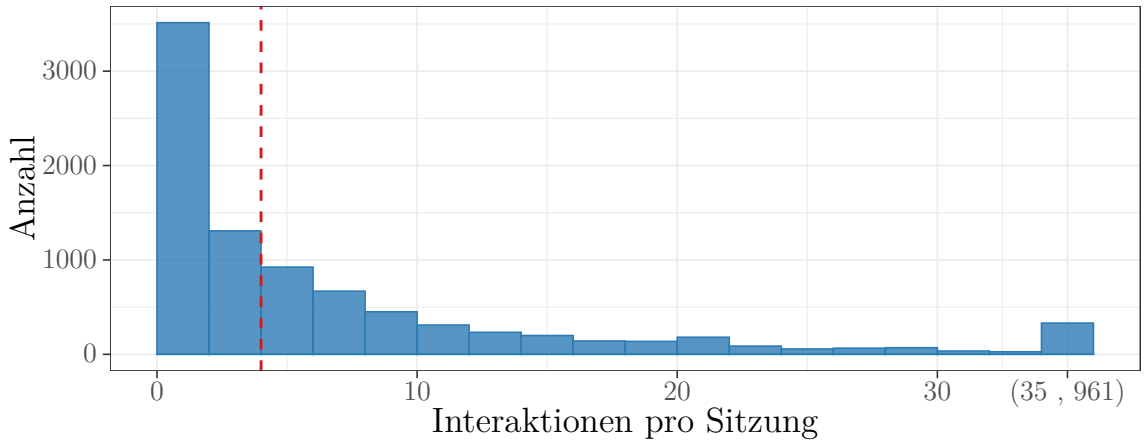


Abbildung 4.2: Histogramm Interaktionen pro Sitzung. Mit eingezeichnetem Median. Die Klassenbreite beträgt zwei. Die letzte Klasse zeigt alle Ausreißer mit über 35 Interaktionen pro Sitzung

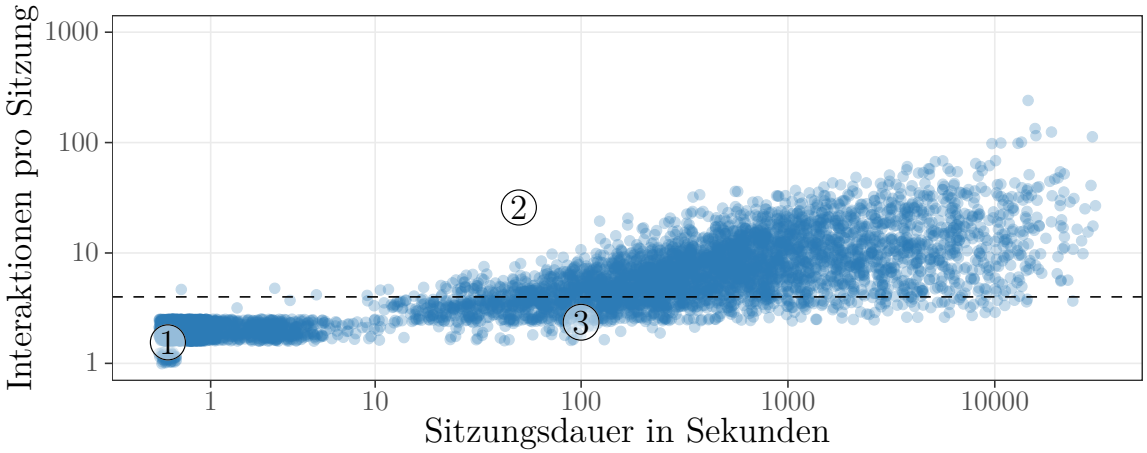
Die Abbildung 4.2 zeigt in einem Histogramm die Verteilung der Interaktionen pro Sitzung. Es existiert ein eindeutiger Trend: Sitzungen mit wenigen Interaktionen sind deutlich häufiger als Sitzungen mit vielen Interaktionen. Es ist abzulesen, dass die Hälfte der Sitzungen aus vier oder weniger Interaktionen besteht.

Ein Blick in die Daten zeigt, dass 40% der Sitzungen weniger als zehn Sekunden andauern und dabei nur aus einer einzelnen oder zwei Interaktion bestehen, der Nutzer besucht somit ausschließlich die Startseite. Bei der zweiten Interaktion, die kurz nach der ersten Interaktion auftritt, handelt es sich um eine technische Eigenheit des Systems, und keine direkt vom Nutzer verursachte Interaktion. Dieser Umstand wird in Unterabschnitt 5.1.3 behandelt.

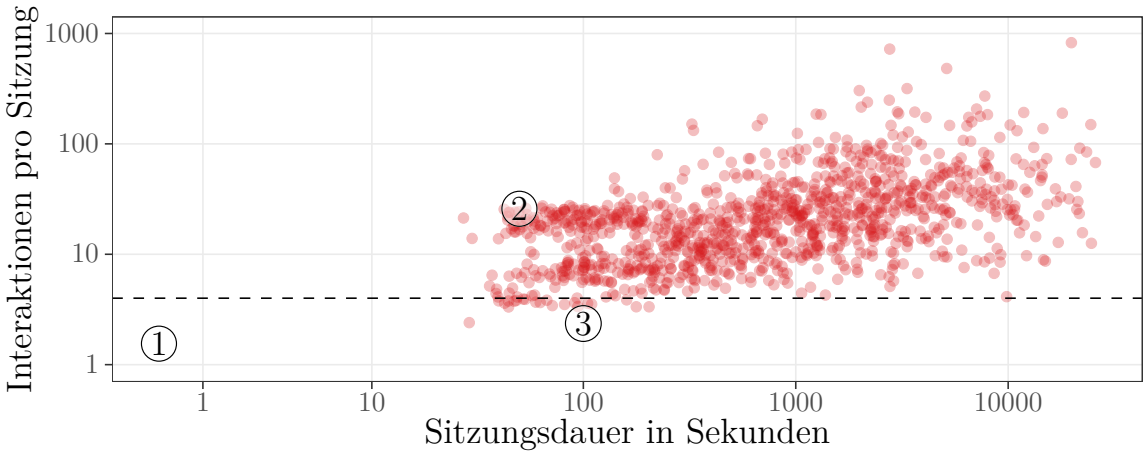
Der Anteil der Nutzer, die nach dem Betrachtem der Startseite die Seite verlassen, wird auch als Absprungrate bezeichnet und ist ein wichtiger Messwert bei der Analyse von Web Traffic (Poulos, Korfiatis und Papavlassopoulos 2019).

Es ist für diese Arbeit nicht relevant, die Gründe für den schnellen Absprung einiger Nutzer zu finden. Doch Sitzungen mit nur sehr wenigen oder gar keinen Interaktionen bieten nur eine unzureichende Datenbasis für Vorhersagen von folgenden Interaktio-

nen. Der Abbruch der Sitzung erfolgte schon, bevor eine Suche durchgeführte wurde oder Filter gesetzt wurden.



(a) ● Sitzungen ohne Bestellung



(b) ● Sitzungen mit Bestellung

Abbildung 4.3: Verteilung der Anzahl von Interaktionen pro Sitzung abhängig von der Sitzungsdauer. Beide Achsen sind logarithmisch skaliert. Verwendung von Jittering um Überlagerungen der Punkte zu verringern. Orientierungslinie bei drei Interaktionen pro Sitzung eingefügt. Unterschiede zwischen den Plots mit ①-③ markiert

Abbildung 4.3 zeigt die Verteilung der Interaktionen pro Sitzung abhängig von der Sitzungsdauer. Dabei wird zwischen zwei Ausprägungen einer Sitzung unterscheiden: Abbildung 4.3a zeigt Sitzungen ohne eine ausgeführte Bestellung, Abbildung 4.3b zeigt Sitzungen mit mindestens einer Bestellung. Die eingefügten Clustermarkierungen (①-③) sind bei beiden Ausprägungen an der jeweils gleichen Position. Sie zeigen

Cluster auf, an denen bedeutende Unterschiede zwischen den Ausprägungen liegen. An den sonstigen Stellen existieren keine signifikanten Unterschiede.

Es lässt sich bei beiden Ausprägungen eine Gemeinsamkeit als Trend erkennen: Längere Sitzungen korrelieren mit vielen Interaktionen pro Sitzung. Aus den markierten Clustern lässt sich schließen:

- ① Ein Cluster, bei dem alle Sitzungen sich durch eine Interaktionsanzahl von eins oder zwei auszeichnen, tritt ausschließlich bei Sitzungen ohne Bestellung auf. Kennzeichnend ist weiterhin eine sehr kurze Sitzungsdauer. Dieser Cluster repräsentiert diese Sitzungen, die in die Absprungrate einfließen. Das Aufgeben einer Bestellung ist eine Interaktion und erfordert mindestens eine weitere Interaktion vorher, das Laden der Startseite. In den meisten Fällen, sofern der gewünschte Datensatz nicht gleich auf der Startseite gelistet ist, auch noch eine weitere Suchanfrage. Somit existieren keine Sitzungen, welche Bestellungen enthalten und gleichzeitig sehr geringe Interaktionszahlen aufweisen.

- ② Bei einer kurzen Sitzungsdauer von unter einer Minute und einer vergleichsweise hohen Interaktionsanzahl bildet sich bei den Sitzungen mit Bestellung ein Cluster, welches bei den Sitzungen ohne Bestellungen nicht existiert. Ein Grund dafür sind die Bestellungen selbst, welche zu mindestens einer zusätzlichen Interaktion in einer Sitzung beitragen. Dies reicht aber alleine nicht, um den Unterschied zu erklären. Möglicherweise handelt es sich in diesem Cluster um Nutzer, die mit dem EGP sehr vertraut sind. Diese Nutzer zeichnen sich durch eine genaue Vorstellung aus, welchen Datensatz sie benötigen und können in dem bekannten System gezielt navigieren.

- ③ Sitzungen bestehend aus drei oder vier Interaktionen, in Kombination mit einer verhältnismäßig langen Sitzungsdauer, treten fast ausschließlich ohne enthaltene Bestellungen auf. Bei den Sitzungen mit Bestellungen erscheinen sie nur sehr vereinzelt.

Eine Vorhersage zukünftiger Interaktionen wird anhand dieses Datensatzes im folgenden Kapitel durchgeführt.

5 Umsetzung

In diesem Kapitel erfolgt die Umsetzung des vorgestellten Konzepts. Die Umsetzung erfolgt unter Verwendung des EGP-Datensatzes. Zuerst wird die für den Datensatz erforderliche Datenvorverarbeitung beschrieben. Anschließend wird die Funktionsweise und vorgenommene Implementation der gewählten Vorhersageverfahren dargelegt. Schließlich wird die vorgenommene technische Umsetzung erläutert.

Das vorgestellte Vorgehen ist für alle Datensätze anwendbar, welche die in Abschnitt 3.1 vorgestellten Anforderungen erfüllen. Es sind nur wenige Anpassungen des Vorgehens an den Datensatz notwendig. So verwendet das hier vorgestellte Vorgehen einen Abruf der Daten aus Elasticsearch, es wäre aber auch ein Abruf von anderen Datenbanken oder aus Textdateien möglich. Ebenso muss das Generieren der Transaktionsdatenbank an den Datensatz angepasst werden.

5.1 Datenvorverarbeitung

Insbesondere Datensätze, welche nicht unter kontrollierten Bedingungen gewonnen werden, sind anfällig für fehlerhafte Daten (Han, Kamber und Pei 2012, S. 83). Dies können unter anderem fehlerhafte, rauschende, oder inkonsistente Einträge sein. Diese können die Erfolge von Data-Mining Prozessen erheblich schmälern, in vielen Fällen ist deshalb eine Datenvorverarbeitung notwendig. Eine gezielte Vorverarbeitung der Daten vor der Anwendung eines Vorhersagealgorithmus führt zu einem besseren Vorhersageergebnis (Han, Kamber und Pei 2012, S. 87).

5.1.1 Aggregation

Im Rahmen von vorherigen Arbeiten an diesem Datensatzes durch (Schindler, Paradies und Twele 2019) wurden die textbasierten Logdateien des EGP in eine Elasticsearch-Datenbank eingelesen und jeder Logeintrag durch reguläre Ausdrücke in einzelne Felder zerlegt.

Um die vorgestellte Analyse durchzuführen, werden die Daten aus der Elasticsearch-Datenbank extrahiert. Es wird eine Aggregation durchgeführt. Dazu werden Interaktionen mit gleicher Session-ID zu einer Sitzung zusammengefasst. Innerhalb einer Sitzung werden die Interaktionen anhand des Zeitstempels aufsteigend sortiert.

5.1.2 Generieren der Transaktionsdatenbank aus Feldern des Datensatzes

Um aus den Feldern der Interaktionen des Datensatzes Elemente für die Transaktionsdatenbank zu gewinnen, wird eine Transformation angewendet. Dazu wird die im Konzept vorgestellte Methode genutzt: Es wird ein Vergleich der Felder der derzeitigen Interaktion I_c mit den Feldern der vorherigen Interaktion I_{c-1} durchgeführt. Als Felder werden nur die verwendeten, verfügbaren Suchfilter betrachtet und keine Bestellungen oder Paginierungsinformationen. Es können folgende Elementzustände auftreten:

- Hinzugefügt – Der Filter existierte in I_{c-1} noch nicht, aber in I_c.
- Geändert – Der Filter existiert sowohl in I_{c-1} als auch in I_c, wobei der Wert des Filters sich ändert.
- Existierend – Der Filter existiert sowohl in I_{c-1} als auch in I_c, wobei der Wert des Filters gleich bleibt.
- Entfernt – Der Filter existiert in I_{c-1} aber nicht in I_c.

Das resultierende Element ergibt sich aus der Konkatenation von der Art des Filters mit dem zugehörigen Zustand. Somit entsteht etwa `Zeitlich_Geändert` als Element,

wenn sich der zeitliche Filter im Vergleich zur vorherigen Interaktion geändert hat.

Weiterhin wird am Beginn jeder Sitzung ein „Start“-Element eingefügt. Dies dient dazu, Muster am Beginn einer Sitzung erkennen zu können und somit an dieser Stelle eine angepasste Vorhersage zu ermöglichen. Eine Sequenz in der Transaktionsdatenbank einer einzelnen Sitzung des EGP könnte folgendermaßen aussehen:
`{Start}, {Kollektion_Hinzugefügt}, {Zeitlich_Hinzugefügt, Kollektion_Entfernt}`

In diesem Beispiel fügte der Nutzer im ersten Schritt einen Kollektionsfilter hinzu. Im zweiten Schritt wurde der Kollektionsfilter wieder entfernt und ein zeitlicher Filter hinzugefügt.

5.1.3 Filterung

Die Filterung kann zum Ausschluss einzelner Interaktionen (Filterung der Interaktionen) einer Sitzung oder zum Ausschluss kompletter Sitzungen (Filterung der Sitzungen) führen, mit dem Ziel, das Vorhersageergebnis zu verbessern. Zuerst wird die Filterung der Interaktionen ausgeführt, danach die Filterung der Sitzungen.

Filterung der Interaktionen
Ein häufig zu erkennnendes Muster ist das *Auftreten von zwei Interaktionen innerhalb eines sehr kurzem Zeitabstands*. Die Interaktionen ereignen sich innerhalb einer Sekunde – oft sogar im Zehntelsekundenbereich. Dieses Muster tritt häufig am Beginn einer Sitzung auf.

Auf Nachfrage konnten die Betreiber des EGP den Sachverhalt klären. Es ist technisch bedingt, dass ein Nutzer beim Laden der Seite neben der eigentlichen Anfrage eine weitere Anfrage im Hintergrund abschickt und somit einen weiteren Eintrag in den Logdateien auslöst. Da somit eine der Anfragen nicht durch eine tatsächliche Benutzerinteraktion ausgelöst wird, wird bei Vorliegen dieses Musters eine der Interaktionen entfernt.

Als *leere Interaktionen* werden Interaktionen bezeichnet, bei denen sich im Vergleich zur vorherigen Interaktion kein Feld verändert. Unter dieser Voraussetzung können für den verwendeten Datensatz folgende Interaktionen als „leer“ bezeichnet werden und somit entfernt werden: Die Nutzung der Paginierung um die Ergebnisseite zu wechseln, die Nutzung der Paginierung um die Anzahl der Ergebnisse auf einer Seite zu ändern oder bei dem Bestellen von Produkten.

Filterung der Sitzungen
Einige Interaktionen enthalten eine *Leere Sitzungs-ID*. Dies äußert sich darin, dass das Feld gar nicht existiert oder der Wert `null` ist. Zu Beginn der Aufzeichnungen war das EGP noch nicht konfiguriert, um Sitzungs-IDs zu speichern. Bei der Aggregation werden alle Interaktionen mit leerer Sitzungs-ID zusammengeführt, was in diesem Fall missleitend ist. Deshalb werden diese Interaktionen entfernt.

Wenn die Seite schon nach *wenigen Interaktionen* verlassen wird, reichen die vorhandenen Daten nicht aus, um eine verlässliche Vorhersage zu tätigen. Aufgrunddessen werden Sitzungen mit nur einer oder zwei Interaktionen herausgefiltert. Dabei wird die bereinigte Interaktionszahl nach dem vorherigen Schritt „Filterung der Interaktionen“ betrachtet.

5.2 Vorhersage durch Anwendung der Sequenzregelanalyse

Es folgt die Beschreibung der Implementation des verwendeten Verfahrens um die Sequenzregelanalyse zu nutzen, um die Vorhersage zu tätigen.

Das Bilden von vollständig sortierten Sequenzregeln mithilfe des CSPADE-Algorithmus konnte die Erkenntnisse aus (Fournier-Viger, T. T. Gueniche und S. 2012) bestätigen: Die erstellten Regeln waren zu spezifisch, um sie für eine Vorhersage nutzen zu können. Kennzeichnende Merkmale der erstellten Regeln waren Variationen, der Form $\{A, B\} \rightarrow \{A\}$ oder $\{A\}, \{C\} \rightarrow \{A\}$. Genauer betrachtet, zeigt sich, dass diese Regeln lediglich eine Aussage darüber liefern, dass A auf A folgt. Dabei existieren vielfältige Varianten mit verschiedenen Elementen auf der linken Seite der Regel, welche sich oft nur minimal unterscheiden. Kennzeichnend für diese Varianten waren niedrige Supportwerte der Regeln. All dies sind Indikatoren dafür, dass eine Vorhersage mit diesem Verfahren nicht zielführend ist. Der Einsatz vollständig sortierter Sequenzregeln wurde nicht weiter verfolgt. Stattdessen wurden partiell sortierte Sequenzregeln eingesetzt.

5.2.1 Vorhersage unter Anwendung partiell sortierter Sequenzregeln (PSSR)

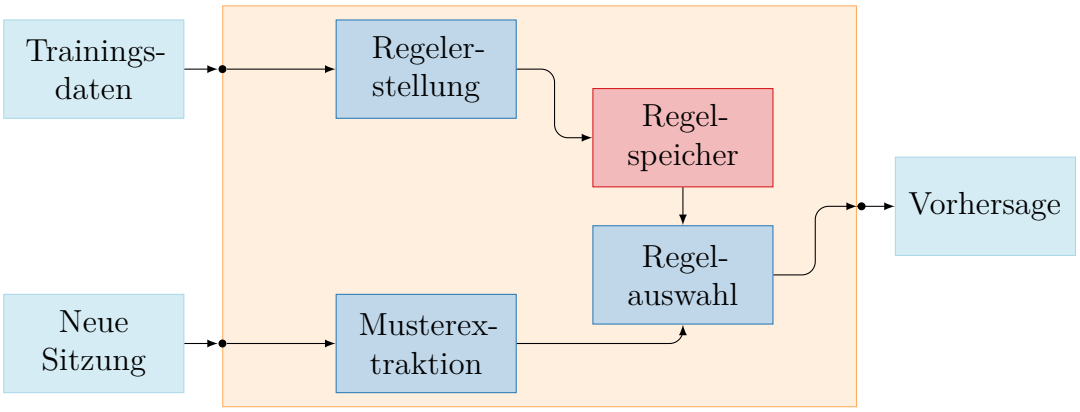


Abbildung 5.1: Schema PSSR

In Abbildung 5.1 wird ein Überblick über die Vorhersage unter Verwendung von partiell sortierten Sequenzregeln gegeben. Die Implementation erfolgte so, dass der vorgegebene Rahmen mit Ein- und Ausgabedaten aus Abschnitt 3.2 eingehalten wird.

Es werden die einzelnen Schritte, um eine Vorhersage durchzuführen, vorgestellt. Am Anfang jedes Unterpunktes steht in *kursiv* eine kurze Zusammenfassung, anschließend folgt die ausführliche Beschreibung.

Regelerstellung

Anhand der Trainingsdaten werden die Sequenzregeln mit dem TRuleGrowth-Algorithmus erstellt.

Der erste Schritt ist die Regelerstellung anhand der Trainingsdaten. Dazu findet der in (Fournier Viger, C.-W. Wu, Tseng und Nkambou 2012) vorgestellte *TRuleGrowth* Algorithmus Verwendung. Dieser Algorithmus ist durch die große Zahl an möglichen Hyperparametern flexibel nutzbar. Im Folgenden werden die verfügbaren Hyperparameter vorgestellt.

Minimaler Support (`min_sup`)
Von allen erkannten Regeln werden ausschließlich die ausgegeben, deren Support größer als `min_sup` ist. Für das vorgestellte Vorgehen ist eine genaue Bestimmung dieses Wertes

nicht entscheidend, da Regeln mit geringem Support ohnehin später durch das Scoring selten oder gar nicht verwendet werden. Ein zu hoher Wert von `min_sup` kann dagegen die Vorhersagegenauigkeit reduzieren, da nur sehr wenige Regeln erstellt werden können.

Minimale Konfidenz (`min_conf`)
Von allen erkannten Regeln werden ausschließlich die ausgegeben, deren Konfidenz größer als `min_conf` ist. Ein Erhöhen des Wertes von `min_conf` verringert die Regelverfügbarkeit, da weniger Regeln erstellt werden.

Fenstergröße (`window_size`)
Die Fenstergröße gibt die maximale Distanz zwischen dem Ereignis, welche das erste Element der linken Seite einer Regel enthält und dem Ereignis, welches das letzte Element der rechten Seite der Regel enthält, an. Somit bestimmt die `window_size` über wie viele Interaktionen sich ein Muster erstrecken kann.

Maximale Länge der linken und rechten Seite der Regel (`lhs_size`, `rhs_size`)
Diese beiden Parameter legen die maximale Anzahl der Elemente auf der linken und rechten Seite der Regel fest. Nimmt der Wert beider Parameter beispielsweise eins an, sind ausschließlich Regeln der Form $\{A\} \rightarrow \{B\}$ möglich, aber nicht Regeln der Form $\{A, C\} \rightarrow \{B\}$ oder $\{A\} \rightarrow \{B, C\}$.

Die erstellten Regeln werden als $r_1 \dots r_n$ in dem Regelspeicher hinterlegt und können dort für mehrere Vorhersagen verwendet werden.

Musterextraktion

Es werden aus einer neuen Sitzung die Interaktionen ausgewählt, anhand denen eine Vorhersage durchgeführt wird.

Es tritt eine neue Sitzung auf, welche die Interaktionen $I_1 \dots I_c$ enthält. Der Index der aktuellsten Interaktion wird mit c gekennzeichnet. Ziel ist eine Vorhersage der unbekannten Interaktion I_{c+1}.

Um eine Regel zur Vorhersage bestimmen zu können, ist der erste Schritt das Auswählen der vergangenen Interaktionen, welche als Basis für die Vorhersage dienen. Dies sind alle Interaktionen in einem Intervall von $[I_{lower}, I_{upper}]$. Die Größe dieses Intervalls ist abhängig von einem Parameter:

Präfixgröße (`prefix_size`)
Die Präfixgröße gibt an, wie viele der vergangenen Interaktionen einer Sitzung genutzt werden, um eine zugehörige Sequenzregel zu finden.Eine Präfixgröße von eins ist dabei der kleinstmögliche Wert, in diesem Fall werden nur die Elemente der letzten Interaktion I_c genutzt, um eine Regel auszuwählen.

Dabei sind die Intervallgrenzen von $[I_{lower}, I_{upper}]$ folgendermaßen bestimmt:

$$lower = max(1, c - \texttt{prefix\_size})$$
$$upper = c$$

Es wird eine Funktion $elem(I)$ für eine Interaktion definiert, welche die Elemente dieser Interaktion liefert. Für jede Interaktion im Intervall $[I_{lower}, I_{upper}]$ wird die Funktion $elem(I)$ aufgerufen. Alle Elemente, die dadurch erhalten werden, werden einer gemeinsamen Menge $M_{history}$ hinzugefügt. Die enthaltenen Elemente sind die Elemente, die verwendet werden, um eine Vorhersage zu erstellen.

Regelauswahl

Es erfolgt eine Auswahl der Regel, die zum bisherigen Verlauf der Sitzung passt. Es erfolgt zuerst eine Filterung nach Regeln, welche mit dem bisherigen Interaktionsverlauf übereinstimmen. Anschließend werden diese Regeln anhand eines Scorings gewichtet um die relevanteste Regel zu identifizieren. Die rechte Seite der Regel wird für die Vorhersage eingesetzt.

Es werden für eine Regel r die Funktionen $lhs(r)$ und $rhs(r)$ definiert, welche jeweils die Elemente der linken beziehungsweise der rechten Seite der Regel zurückgeben. Weiterhin geben die Funktionen $conf(r)$ und $sup(r)$ die Konfidenz und den Support einer Regel aus.

Es erfolgt eine Suche nach Regeln, welche zur Vorhersage geeignet sind. Dazu wird für jede Regel r in $r_1 \dots r_n$ geprüft, ob die linke Seite der Regel eine Untermenge von $M_{history}$ ist: $lhs(r) \subseteq M_{history}$

Falls keine einzige Regel existiert, die diese Bedingung erfüllt, kann keine Vorhersage getätigt werden. Die Regelverfügbarkeit ist nicht gegeben. Falls mehrere Regeln existieren, die diese Bedingung erfüllen, erfolgt eine Selektion anhand des Scoring.

Die Berechnung des Scores erfolgt für jede Regel r in $r_1 \dots r_n$ beim Einlesen der Sequenzregeln. Dazu wird eine Formel angewendet, welche auf einem Vorgehen aus (Fournier-Viger, T. T. Gueniche und S. 2012) basiert. Der Score jeder Regel wird im Folgenden anhand dieser Formel bestimmt:

$$score(r) = (c_1 \times conf(r) + c_2 \times supp(r)) \times |lhs(r)|$$

$|lhs(r)|$ bezeichnet die Anzahl der Elemente der linken Seite der Sequenzregel r. Weiterhin werden die an den Datensatz angepassten Konstanten c_1 und c_2 verlangt[1]. Das Verhältnis dieser Konstanten gewichtet den Konfidenz- und Supportwert für die Berechnung des Scores. Bei diesen Konstanten handelt es sich auch um Hyperparameter.

Gewichtung von Support und Konfidenz (`c1`)
Der absolute Wert des Scores einer Regel ist irrelevant. Es reicht aus, den Score einer Regel im Vergleich zu anderen Regeln zu betrachten. Deshalb ist es ausreichend, das Verhältnis zwischen c_1 und c_2 zu variieren. Somit wird c_2 konstant auf 0.5 festgesetzt[2] und ausschließlich c_1 untersucht. Dadurch wird der Suchraum eingeschränkt und die Rechenzeit zur Modellbestimmung kann reduziert werden.

Das vorgestellte Verfahren wählt eine Regel nur als geeignet zur Vorhersage aus, wenn die linke Seite der Regel eine Untermenge (beziehungsweise komplett enthalten) in $M_{history}$ ist. Das Verfahren von (Fournier-Viger, T. T. Gueniche und S. 2012) hingegen erlaubt auch eine partielle Übereinstimmung. Dort wird auch in der Scoring Formel nicht $|lhs(r)|$ verwendet. Stattdessen wird bei jeder Vorhersage bestimmt, wieviele Elemente sowohl in $M_{history}$ als auch in $|lhs(r)|$ enthalten sind. Dies kann auch als $|lhs(r) \bigcap M_{history}|$ beschrieben werden.

Die Verwendung der partiellen Übereinstimmung, wie in (Fournier-Viger, T. T. Gueniche und S. 2012) gezeigt, sorgte bei dem vorliegenden Datensatz für schlechtere Vorhersageergebnisse und wurde deswegen nicht verwendet. Das hier verwendete Verfahren erlaubt, das Scoring der Regeln einmal nach dem Trainingsprozess durchzuführen und im Regelspeicher zu hinterlegen. Der errechnete Score kann für jede Vorhersage wiederverwendet werden. Dies verringert die Rechenzeit im Vergleich zu (Fournier-Viger, T. T. Gueniche und S. 2012), dort muss der Score einer Regel bei jeder Vorhersage neu berechnet werden.

Es erfolgt die Wahl der Regel mit dem höchsten Score, diese Regel wird als r_{pref} vermerkt. Anschließend werden die Elemente der rechten Seite dieser Regel $rhs(r_{pref})$ als Vorhersage ausgegeben.

[1] In (Fournier-Viger, T. T. Gueniche und S. 2012) werden mit $c_1 = 0,7$ und $c_2 = 0,3$ die besten Vorhersagen erzielt.
[2] Es wäre auch jeder andere Wert möglich.

5.2.2 Priorisierte partiell sortierte Sequenzregeln (PPSSR)

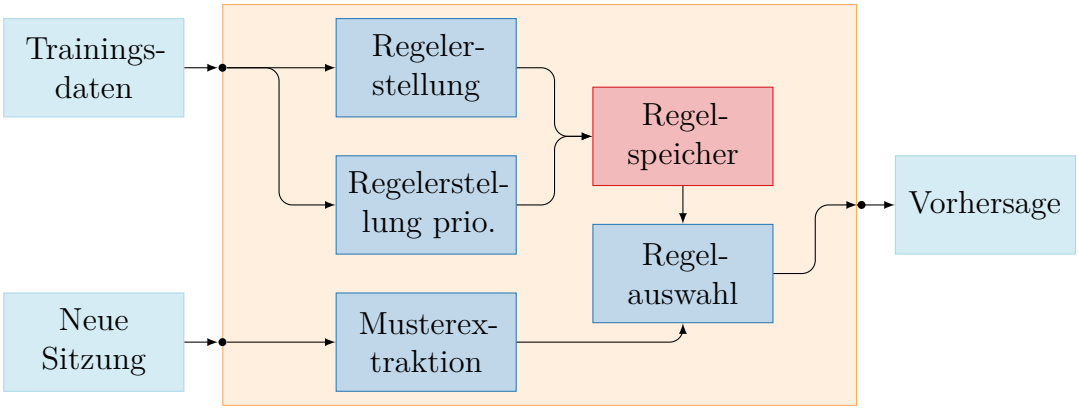


Abbildung 5.2: Schema PPSSR

Die Vorhersage unter der Verwendung priorisierter partiell sortierte Sequenzregeln, dargestellt in Abbildung 5.2, ähnelt dem vorher vorgestellten Verfahren. Der einzige Unterschied ist, dass eine zweite Regelerstellung durchgeführt wird.

Dieses Verfahren wurde im Rahmen dieser Arbeit entwickelt. Die Analyse der Ergebnisse des vorher beschriebenen PSSR-Verfahrens zeigte positive Auswirkungen auf die Vorhersagegenauigkeit, wenn der `window_size` Parameter so angepasst wird, dass Muster gefunden werden, welche viele Interaktionen umspannen. Das resultiert jedoch darin, dass Muster, die innerhalb weniger Interaktionen auftreten, schlechter erkannt werden[3].

Die Muster, welche wenige Interaktionen umspannen, werden im Folgenden als „kurze“ Muster bezeichnet, die, welche viele Interaktionen umspannen, als „ausgedehnte“. Die Analyse zeigte weiterhin, dass einige kurze Muster existieren, deren Vorhersage sehr zuverlässig ist. Das gilt insbesondere, wenn der Datensatz Elemente enthält, deren Vorkommen überdurchschnittlich häufig im Vergleich zu anderen Elementen ist.

Anhand dieser Erkenntnisse wurde ein Vorgehen entwickelt, welches sowohl kurze als auch ausgedehnte Muster erkennen und vorhersagen kann. Dazu wird eine Erweiterung der partiell sortierten Sequenzregeln eingeführt und als *priorisierte* partiell sortierte Sequenzregeln bezeichnet. Dazu werden zwei Sätze von Sequenzregeln mit verschiedenen `window_size`-Parametern getrennt bestimmt. Bei der Wahl der Sequenzregel zur Vorhersage werden die Regeln, welche kurze Muster erkennen, priorisiert – daher auch die Namensgebung.

[3] Das diese Analyse so detailliert möglich ist, unterstreicht die Vorteile der Nachvollziehbarkeit und Interpretierbarkeit bei Verwendung von Sequenzregeln im Gegensatz zu anderen Verfahren.

Im Folgenden werden die Unterschiede dieses Verfahrens im Vergleich zu PSSR beschrieben, das restliche Vorgehen bleibt gleich.

Erstellen der priorisierten Regeln

Es werden Regeln erstellt, die mit hoher Zuverlässigkeit kurze Muster erkennen können.

Zur Erstellung der priorisierten Regeln, welche verwendet werden, um kurze Muster zu identifizieren, werden angepasste Hyperparameter verwendet. Die `window_size`, `lhs_size`, `rhs_size` und `prefix_size` wird auf eins festgelegt. Die minimale Konfidenz einer Regel `min_conf` wird bei den priorisierten Regeln immer höher gewählt als bei den normalen Regeln. Da die Prioritätsregeln stets bevorzugt werden, müssen sie auch mit einer größeren Wahrscheinlichkeit das korrekte Ergebnis vorhersagen. Das Erstellen der normalen Sequenzregeln erfolgt genauso wie im PSSR Verfahren, es erfolgt eine Optimierung der Hyperarameter.

Die normalen Sequenzregeln werden als $r_1 \dots r_n$ im Regelspeicher abgegelegt. Die priorisierten Regeln hingegen als $rp_1 \dots rp_n$. Somit können die unter verschiedenen Parametern erstellten Regeln im weiteren Vorgehen unterschieden werden.

Regelauswahl

Wenn eine priorisierte, passende Sequenzregel existiert wird diese verwendet, ansonsten eine normale Sequenzregel.

Nach dem Bilden der Menge an Elementen, die für die Vorhersage verwendet werden $M_{history}$, erfolgt zunächst die Prüfung, ob eine passende priorisierte Regel existiert:

Dazu wird für jede priorisierte Regel rp in $rp_1 \dots rp_n$ zunächst geprüft, ob sie geeignet zur Vorhersage ist, die Bedingung ist äquivalent zur Prüfung bei PSSR:

$$lhs(rp) \subseteq M_{history}$$

Dabei wird die priorisierte Regel mit dem höchsten Score, welche diese Bedingung erfüllt, als r_{pref} gekennzeichnet und deren rechte Seite ausgegeben. Existiert keine passende priorisierte Regel wird das Auswahlverfahren mit dem normalen Regeln analog zu PSSR ausgeführt.

5.3 Technische Umsetzung

Der vorgestellte Ablauf zum Durchführen der Vorhersage besteht aus mehreren Schritten, welche in verschiedenen Modulen der Programmiersprache Python abgebildet wurden. Es existieren Module zum Erstellen der Transaktionsdatenbank aus dem Datensatz, zum Erstellen der Regeln und schließlich zum Durchführen der Vorhersage. Weiterhin gibt es einen Wrapper, der einen vorhandenen Datensatz in Trainings- und Testdaten splittet, um die Vorhersage zu prüfen und das Modell mit den geeignetesten Hyperparametern zu wählen. Durch den modularen Aufbau können die Schritte einzeln durchgeführt werden. Dies ist zur Analyse des Prozesses geeignet, etwa wenn es nötig ist, temporäre Dateien wie die Transaktionsdatenbank oder die erstellten Regeln zu inspizieren. Ebenso ist es geeignet um die Auswirkungen von Parametern zu untersuchen und manuell anzupassen.

Das Aufrufen von SPMF (Fournier-Viger, Lin u. a. 2016) zur Erstellung der Regeln erfolgt als Aufruf eines Subprozesses, da SPMF in der Programmiersprache Java geschrieben ist. Der Datenaustausch zwischen den Python-Modulen und SPMF, bestehend aus Transaktionsdatenbank als Eingabe für SPMF und den erzeugten Regeln als Ausgabe, geschieht über temporäre Textdateien.

An vielen Stellen wurden Funktionen der Python-Bibliothek pandas(team 2020) verwendet. Diese bietet mit `dataframes` eine Datenstruktur zur Abbildung von tabellarischen Daten. Viele Aktionen lassen sich spalten- oder zeilenweise durchführen. Durch flexible Möglichkeiten zur Indizierung, Gruppierung und Filterung von Daten war es möglich den Implementationsaufwand deutlich zu reduzieren.

Der Quellcode ist unter `https://gitlab.dlr.de/surb_ti/predict_user_interactions` verfügbar.

6 Evaluation

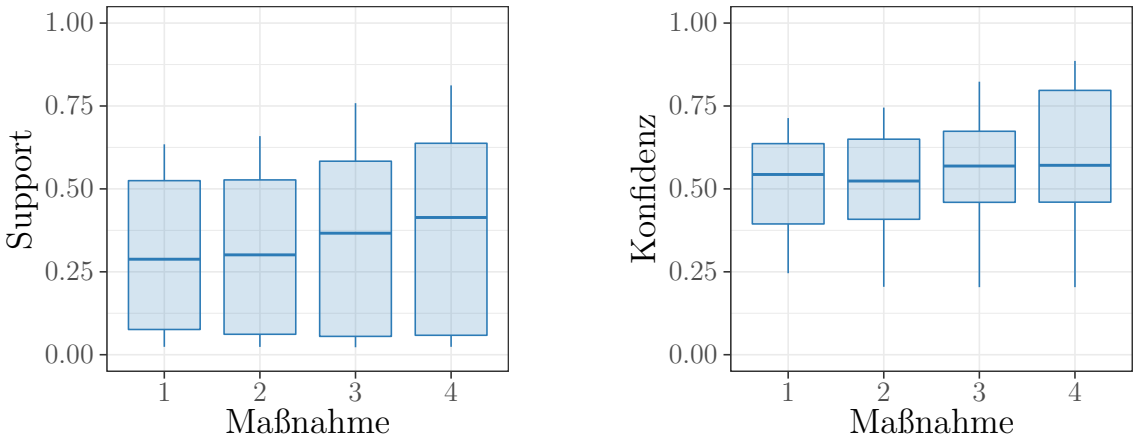
Die in diesem Kapitel gezeigten Ergebnisse entstanden, indem die beiden Vorhersageverfahren durch eine Aufteilung des EGP-Datensatzes in Test- und Trainingsdaten geprüft wurden. Weiterhin folgt ein Vergleich mit zwei weiteren Vorhersageverfahren.

Zuerst werden die Auswirkungen der durchgeführten Vorfilterung demonstriert. Danach wird beschrieben, wie aus den zahlreich durchgeführten Versuchen mit verschieden Parametern das geeigneteste Modell selektiert wurde. Es folgt eine Diskussion der erzielten Vorhersageergebnisse. Schließlich werden die Resultate der Hyperparameteroptimierung analysiert.

6.1 Ergebnisse der Vorfilterung

Eine Vorfilterung hat das Ziel, die Vorhersageergebnisse zu verbessern. Um zu beweisen, dass das gewählte Verfahren zielführend war, werden die Auswirkungen der einzelnen Filterschritte in Abbildung 6.1 gezeigt. Um diese Abbildung zu erstellen, wurden verschiedene Schritte der Filterung auf den Datensatz angewendet. Nach jedem Schritt wurden Sequenzregeln erzeugt und die Kennwerte Support und Konfidenz jeder Regel bestimmt.

In der Grafik ist ein Anstieg von sowohl Support als auch der Konfidenz mit fortschreitender Filterung zu erkennen. Insbesondere in den oberen zwei Quartilen ist diese Erscheinung ausgeprägt. Der dortige Anstieg ist am relevantesten für die Vorhersage, da aufgrund des Scoring-Verfahrens Regeln mit hohen Konfidenz- und Supportwerten bevorzugt eingesetzt werden. Dies zeigt, dass die verwendeten Filterungsmaßnahmen zielführend sind und die Bildung einer hochwertigen Datenbasis stützen, die es ermöglicht, Vorhersagen mit großer Genauigkeit zu tätigen.



(a) Entwicklung des Support-Wertes (b) Entwicklung des Konfidenz-Wertes

Abbildung 6.1: Einfluss der Vorfilter-Schritte der Interaktionen auf Konfidenz und Support der Sequenzregeln.
1) Ohne Filterung
2) Entfernen der Sitzungen mit kurzem Zeitabstand
3) Zusätzlich: Entfernen des Pagings
4) Zusätzlich: Entfernen der Bestellungen

6.2 Parametrisierung

Die Prüfung der Vorhersage wurde auf verschiedene Weisen parametrisiert. Einerseits werden für die sequenzregelbasierten Verfahren die optimalen Hyperparameter bestimmt.

Auch der Prüfungpsrozess der Vorhersage an sich wird parametrisiert, um die verwendeten Vorhersageverfahren in verschiedenen Situationen evaluieren zu können. Die Kombination der Prüfungsprozessparameter wird im Folgenden als *Konfiguration* bezeichnet.

6.2.1 Hyperparameter

Die Vorhersage unter Verwendung der Sequenzregelanalyse ist von mehreren Hyperparametern abhängig, sie beeinflussen die Regelerstellung und Regelauswahl. Die Hyperparameter wurden in Abschnitt 5.2 beschrieben.

Die korrekten Werte der Hyperparameter lassen sich nicht aus dem Datensatz ablesen oder vorhersagen, sie müssen experimentell bestimmt werden (Claesen und De Moor 2015). Dabei muss beachtet werden, dass ein Hyperparameter nicht unabhängig von anderen

Hyperparametern ist. Wenn ein Hyperarameter positiv mit der Vorhersagegenauigkeit korreliert, lässt sich nicht zwangsläufig daraus schließen, dass das in Kombination mit anderen Parametern genauso ist.

Anhand von durchgeführten Stichproben wurde das Intervall der Werte eines jeden Hyperparameters geschätzt. Falls sich Top-Kandidaten nahe der Grenze eines Parameterintervalls befanden, wurde der Versuch erneut mit erweitertem Wertebereich durchgeführt.

6.2.2 Konfiguration

Die Prüfung der Vorhersageergebnisse der Algorithmen anhand verschiedener Parameter erlaubt eine qualifiziertere Aussage über die Eignung der Algorithmen. Die Konfiguration wird genutzt um eine Prüfung in verschiedenen Situationen durchzuführen. Eine Vergleichbarkeit der Algorithmen untereinander ist nur gegeben, falls die Konfiguration gleich ist.

Ausschluss von Elementzuständen (`excl_elem`)
Der `excl_elem` Parameter wird verwendet, um das Verhalten der Vorhersage auf sich ändernde Datensätze zu prüfen. Durch gezieltes Entfernen einzelner Elemente aus der Transakationsdatenbank ist es möglich, die Verteilung der Elemente zu beeinflussen.

Der `excl_elem` Parameter kann genutzt werden, um gezielt die Häufigkeitsverteilung der Elemente zu beeinflussen. Wird das am häufigsten verwendete Element aus der Transaktionsdatenbank entfernt, so ist die Häufigkeit der verbleibenden Elemente gleichverteilter. Der Parameter ist somit ein Instrument, um das Verhalten in verschiedenen Vorhersagesituationen zu prüfen, obwohl nur ein Datensatz existiert.

Bei der Vorhersage von Interaktionsdaten aus etwa einem Web-Portal kann sich im Laufe der Zeit die Verteilung der Elemente stets ändern. Das Verhalten des Vorhersageverfahrens bei sich ändernder Datenbasis ist somit eine wertvolle Aussage.

Der Standardfall bei leerem `excl_elem` Parameter ist die Vorhersage anhand aller möglichen Elementzustände (Hinzugefügt, Geändert, Existierend, Entfernt). Bei Angabe eines Elementzustandes im Parameter wird dieser Elementzustand aus der Transaktionsdatenbank vor der Vorhersage entfernt.

Suffixgröße (`suffix_size`)
Ein weiterer Parameter, dessen Einfluss geprüft wird, ist die Suffixgröße. Eine Suffixgrö-

ße von n bedeutet, dass die Vorhersage als korrekt gewertet wird, falls die Vorhersage innerhalb der nächsten n Interaktionen eintritt. Bei einer Suffixgröße von eins muss die Vorhersage innerhalb der nächsten Interaktion eintreten, was für ein Vorhersageverfahren den schwierigsten Fall darstellt.

Der praktische Nutzen der Vorhersage lässt mit steigender Suffixgröße nach. Die Vorhersage, dass eine Nutzertätigkeit sicher in der nächsten Interaktion stattfindet, ist wertvoller und genauer, als dass die Nutzertätigkeit in entweder der nächsten oder der übernächsten Interaktion stattfindet.

Ein Vergrößern der Suffixgröße wird bei jedem Vorhersageverfahren zu einer Erhöhung des Vorhersageerfolgs führen. Die Chance auf einen zufällig richtigen Treffer steigt. Dennoch ist die Betrachtung dieses Parameters wichtig, da er Aussage darüber gibt, ob häufig korrekte Vorhersagen abgegeben wurden, die bloß zu einem zu frühen Zeitpunkt ausgegeben wurden und sich erst später bewahrheiten.

Die Bezeichnung „Suffixgröße“ wurde aus (Fournier-Viger, T. T. Gueniche und S. 2012) übernommen.

6.2.3 Verwendete Parametergrößen

Parameter	Minimaler Wert (inkl.)	Maximaler Wert (inkl.)	Schrittgröße
excl_elem ●	Vorhersage existierender Elemente: ✓/✗		
suffix_size ●	1	3	1
min_support ▲	1	1	0
min_confidence ▲	40	60	5
window_size ▲	1	9	1
lhs_size ▲	1	9	1
rhs_size ▲	1	3	1
c1 ▲	0.3	0.7	0.1
prefix_size ▲	1	9	1

Tabelle 6.1: Verwendete Parametergrößen.
● Konfigurationsparameter
▲ Hyperparameter

Tabelle 6.1 zeigt, mit welchen Parametern die Vorhersageverfahren geprüft wurden. Bei dem `excl_elem` Parameter, welcher nicht numerisch ist, wurden zwei Werte geprüft: Einmal ist der Parameter leer, es werden alle Elemente aus der Transaktionsdatenbank verwendet.

Die andere Variante ist, dass der Parameter den Wert „Existierend“ erhält, somit wird der meistgenutzte Elementzustand des EGP-Datensatzes ausgeschlossen. Es ergeben sich für diesen Datensatz insgesamt sechs Konfiguration welche bei der Vorhersage unterschieden werden (2 `excl_elem` × 3 `suffx_size`).

6.2.4 Parameterkombinationen bilden

Die Vorhersageverfahren werden mit den verschiedenen Hyperparametern und Konfigurationen geprüft. Es wird eine Aufgabenliste erstellt, welche Aufgaben enthält. Eine Aufgabe beinhaltet den Names des zu verwendenden Vorhersageverfahrens, eine Liste der zu verwendenden Parameter sowie die zu verwendenden Test- und Trainingsdaten. Die Vorhersageverfahren durchlaufen alle Aufgaben.

Eine Methode, welche garantiert, die optimale Kombination von Hyperparametern zu finden, ist das Testen jeder möglichen Hyperparameterkombination (Hapke und Nelson 2020, S. 96). Dies wird auch als Rastersuche bezeichnet. Es existieren auch andere Methoden, etwa die zufällige Auswahl gewisser Parameterkombinationen, um die Rechenzeit zu reduzieren. Dabei ist allerdings nicht garantiert, die optimale Lösung zu finden. Für die vorliegenden Parameter wurde sich auf die Rastersuche beschränkt.

Die Erstellung der Aufgabenliste orientiert sich an folgenden Kriterien:

- Jeder Algorithmus muss jede Konfiguration durchlaufen.
- Die auf Sequenzregeln basierenden Algorithmen, welche Hyperparameter nutzen, benötigen eine Optimierung dieser durch die Rastersuche.
- Es erfolgt eine fünffache Prüfung mit jeweils abweichenden Test- und Trainingsdaten.

6.3 Prüfen der Vorhersage

Die Vorhersageverfahren geben eine Vorhersage aus. Es wird mit dem Testdatensatz verglichen, ob die Vorhersage korrekt ist.

Um die Vorhersageergebnisse evaluieren zu können, wird die getätigte Vorhersage $M_{prediction}$

mit den Elementen aus den tatsächlich folgenden Interaktionen der Trainingsdaten verglichen. Dazu bestimmt der Parameter `suffix_size` wie viele Interaktionen der Testdaten betrachtet werden, bis die getätigte Vorhersage in Erfüllung tritt.

Es existieren Interaktion $I_1 \dots I_n$ einer Sitzung in den Testdaten. I_c ist die derzeitig betrachtete Interaktion. Die folgende Interaktion, die vorhergesagt werden soll, ist I_{c+1}. Diese Interaktion ist dem Prüfprozess aber nicht dem Vorhersageverfahren bekannt, da sie aus den Testdaten stammt.

Der Raum der zu betrachtenden Interaktionen, für die geprüft wird ob die Vorhersage zutrifft, wird durch $[I_{lower}, I_{upper}]$ eingeschränkt. Die Intervallgrenzen sind wie folgt definiert:

$$lower = c + 1$$
$$upper = max(n, c + \texttt{suffix\_size})$$

Für jede Interaktion im Intervall $[I_{lower}, I_{upper}]$ wird die Funktion $elem(I)$ aufgerufen. Alle Elemente, die dadurch erhalten werden, werden einer gemeinsamen Menge M_{actual} hinzugefügt. Die Vorhersage wird als korrekt gewertet, wenn die vorhergesagten Elemente in der Menge der tatsächlichen, zukünftigen Elemente der Trainingsdaten enthalten sind $(M_{prediction} \subseteq M_{actual})$.

6.4 Auswahl des Modells

Die Rastersuche prüft mehrere tausend Parameterkombinationen. In diesem Abschnitt sollen folgende Fragen beantwortet werden:
Nach welchem Kriterium erfolgt die Auswahl des geeignetesten Modells?
Wie wird bei der Auswahl vorgegangen?

6.4.1 Auswahlkriterum

Als Auswahlkriterium werden Qualitätsmetriken eingesetzt, die Aussagen über die Vorhersagequalität geben. In (Fournier-Viger, T. T. Gueniche und S. 2012) werden zwei Metriken zur Beurteilung der Vorhersagequalität von Sequenzregeln eingesetzt. Diese werden für diese Auswertung ebenso verwendet.

Die erste Metrik ist die *Vorhersagegenauigkeit*. Eine Vorhersage ist als erfolgreich zu verstehen, wenn die Vorhersage eine Teilmenge der tatsächlich folgenden Elemente ist. Die Vorhersagegenauigkeit ist eine Rate und wird bestimmt durch die Anzahl der erfolgreichen Vorhersagen geteilt durch die Gesamtzahl aller Vorhersagesituationen. Im Folgenden wird sie als Prozentwert angegeben. Die Vorhersagegenauigkeit ist universell für jeden Vorhersagealgorithmus verwendbar.

Die *Vorhersageverfügbarkeit* gibt an, wie häufig es möglich war, in einer Vorhersagesituation tatsächlich eine Vorhersage abzugeben. Die Korrektheit der Vorhersage wird für diese Metrik außer Acht gelassen. Bei der Vorhersage durch Sequenzregeln gibt die Vorhersageverfügbarkeit an, dass mindestens eine Sequenzregel existiert, die zur Vorhersage genutzt werden kann. Die Vorhersageverfügbarkeit ist eine Rate und wird bestimmt durch die Anzahl der Vorhersagesituationen, zu denen eine Vorhersage abgegeben werden kann, geteilt durch die Gesamtzahl aller Vorhersagesituationen. Im Folgenden wird sie als Prozentwert angegeben.

Die Vorhersagegenauigkeit ist zur allgemeinen Bewertung der Vorhersage am Besten geeignet: Es ist die wichtigste Kenngröße einer Vorhersage und für alle Verfahren anwendbar.

6.4.2 Vorgehen

Eine mögliche Strategie um Hyperparameter zu optimieren und somit das geeigneteste Modell zu finden, ist eine einmalige Aufteilung des vorliegenden Datensatz in Test- und Trainingsdaten. Anschließend erfolgt eine Wahl des Modells, welches die höchste Vorhersagegenauigkeit erzielt. Diese Strategie birgt zwei grundsätzliche Risiken:

1. Es erfolgt eine Überanpassung. Das gewählte Modell ist geeignet für genau die gewählte Aufteilung von Test- und Trainingsdaten. Die Eignung für neue Daten ist aber nur eingeschränkt, das Modell ist nicht allgemein genug.

2. Es kommt zu einer Stichprobenverzerrung. Dies kann passieren, wenn die gewählten Testdaten zufällig besonders gut oder besonders schlecht zur Vorhersage geeignet sind.

Eine Methode, um diese Probleme zu minimieren ist die Kreuzvalidierung. Bei einer n-fachen Kreuzvalidierung, wird der Datensatz in n möglichst gleich große Teile aufgesplittet, $n-1$ davon sind Trainingsdaten und der verbleibende Teil wird für Testdaten verwendet. Es folgen n Durchläufe in denen stets die Vorhersagegenauigkeit bestimmt wird.

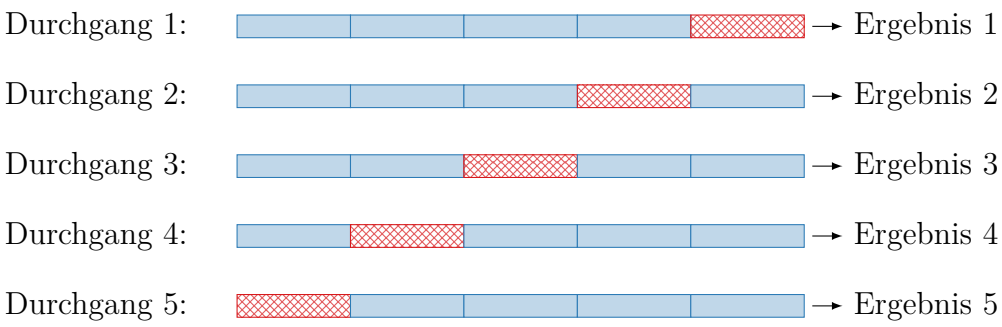


Abbildung 6.2: Kreuzvalidierungsverfahren.
Aufteilung in Trainingsdaten und Testdaten

In jedem Durchlauf besteht der Testdatensatz aus anderen Sitzungen als in den Durchläufen davor. Das Vorgehen wird am Beispiel einer fünffachen Kreuzvalidierung, in Abbildung 6.2 dargestellt.

Anschließend werden die Ergebnisse aus den n Durchläufen gemittelt. Dadurch werden die Probleme der Überanpassung und Stichprobenverzerrung minimiert. Ein Nachteil ist, dass die Rechenzeit zur Bestimmung des Modells n-fach länger dauert.

Um den vorhandenen Datensatz in Test- und Trainingsdaten zu teilen, wird das `KFold` Modul aus der `sklearn` (Pedregosa u. a. 2011) Bibliothek verwendet. Listing 3 zeigt einen Ausschnitt des verwendeten Quellcodes. Der `n_splits` Parameter legt dabei fest, dass eine fünffache Kreuzvalidierung verwendet wird. Der `shuffle` Parameter sortiert die Sitzungen vor der Teilung zufälllig neu. Durch den `random_state` Parameter wird der Zufallsgenerator mit einem festen Wert initialisiert, um reproduzierbare Ergebnisse zu erhalten.

```
cv_split = KFold(n_splits=5, shuffle=True, random_state=123)
for train_indices, test_indices in cv_split.split(dataset):
    ...
```

Listing 3: Split des Datensatz für Kreuzvalidierung

Nach den fünf Durchläufen werden Vorhersagegenauigkeit und Vorhersageverfügbarkeit gemittelt und die Standardabweichung berechnet. Das Modell mit der höchsten Vorhersagegenauigkeit wird selektiert. Zusätzlich werden Statistiken geführt über die Verwendungshäufigkeiten und Korrektheit der Vorhersage jeder einzelnen Regel. Dies trägt zum Datenverständnis bei und erlaubt, gezielte Anpassungen an dem Vorhersageprozess durchzuführen.

6.5 Vergleichsverfahren

Um die Ergebnisse der Verfahren PSSR und PPSSR einordnen zu können und den Erfolg derer zu bewerten, werden zwei Algorithmen implementiert, welche als Vergleichswert dienen. Diese verwenden keine Sequenzregeln. Die Betrachtung der Vorhersageverfügbarkeit ist für diese Verfahren nicht aussagekräftig, da eine Vorhersage immer möglich ist.

Zufallsverteilung
Als Eingabe erhält der Algorithmus eine Liste aller möglichen Elemente, die im Datensatz auftreten können. Anhand eines Zufallsgenerators wird eines der Elemente ausgewählt und zur Vorhersage genutzt. Der Algorithmus ist somit nach der Initialisierung komplett unabhängig von den verwendeten Daten.

Häufigkeitsverteilung
Bei diesem Algorithmus wird in der gesamten Menge der Trainingsdaten die Häufigkeit der vorkommenden Elemente analysiert. Das Element, welches am häufigsten auftritt, wird ausgewählt und in jedem Fall zur Vorhersage genutzt.

Die folgenden Algorithmen dienen als Maßstab. Um den Einsatz eines Vorhersagealgorithmus unter Verwendung von Sequenzregeln rechtfertigen zu können, müssen diese Vergleichswerte bezüglich der Vorhersagegenauigkeit übertroffen werden. Während die Zufallsverteilung eine eher geringe Hürde darstellt, kann der auf Häufigkeitsverteilung basierende Algorithmus insbesondere bei Datensätzen, bei denen einzelne Elemente häufig auftreten, eine hohe Vorhersagegenauigkeit erzielen.

6.6 Auswertung der Vorhersageergebnisse

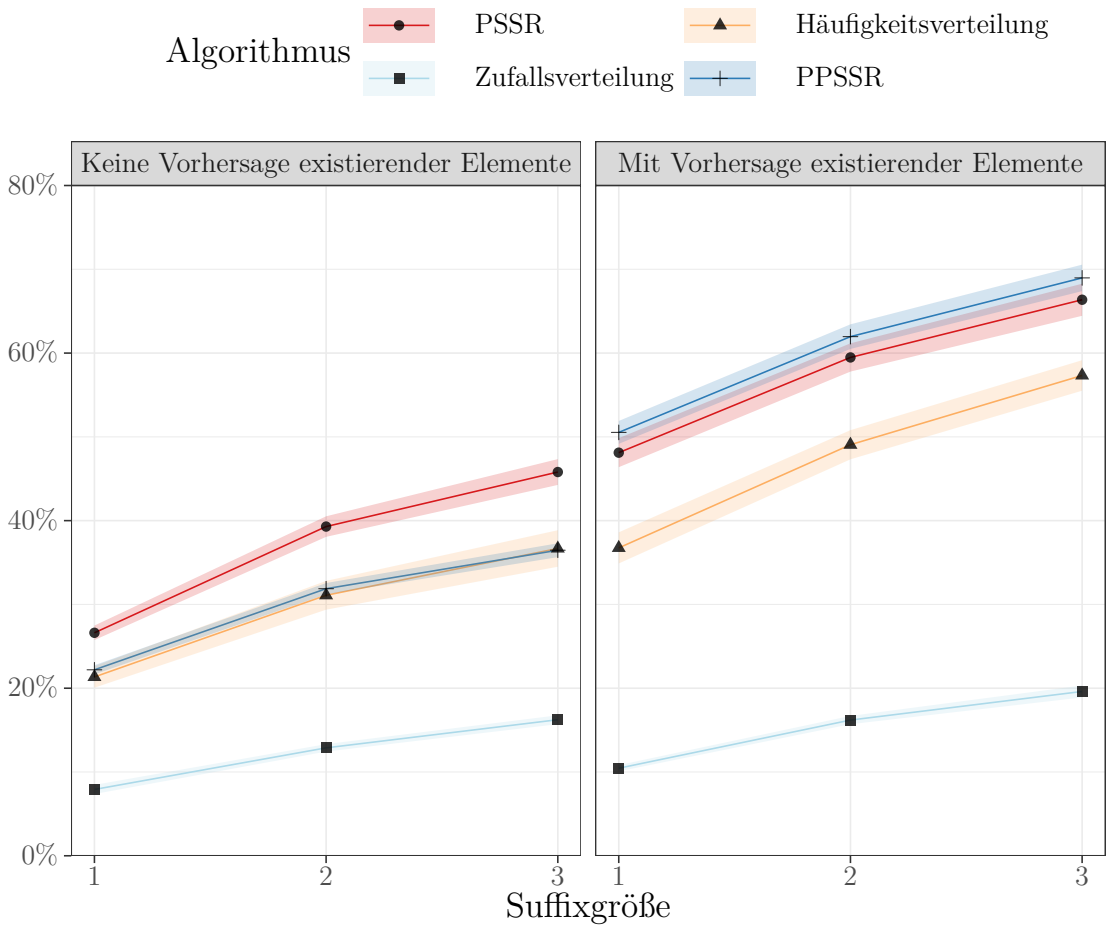


Abbildung 6.3: Vorhersagegenauigkeit der einzelnen Algorithmen. Die Linie stellt den Mittelwert nach der Kreuzvalidierung dar, die hinterlegte Fläche die Standardabweichung.
Auch wenn die Daten auf der x-Achse diskret sind, wurde aufgrund der besseren Ablesbarkeit ein Liniendiagramm zur Darstellung gewählt.

In der Abbildung 6.3 sind die Vorhersagegenauigkeiten der vier betrachteten Algorithmen für die sechs verschiedenen Vorhersagefälle dargestellt. Folgende Sachverhalte können beobachtet werden:

- Allen Algorithmen gemeinsam ist, dass die Vorhersage eines spezifischen Algorithmus bei gleicher Suffixgröße im Falle „Mit Vorhersage existierender Elemente“ stets in einer größeren Genauigkeit als im Falle „Keine Vorhersage existierender Elemente“ resultiert. Dadurch zeigt sich, dass das Vorhandensein von überdurchschnittlich

häufig vorkommenden Elementen in der Transaktionsdatenbank zu einer höheren Vorhersagegenauigkeit führt.

- Das Verhalten bei wachsender Suffixgröße ist bei allen Algorithmen gleich. Dadurch zeigt sich, dass bei keinem der Algorithmen überdurchschnittlich häufig die korrekte Vorhersage zu einem zu frühen Zeitpunkt eintrifft.

- Der Algorithmus, welcher eine Zufallsverteilung nutzt, wird in allen Fällen von den anderen Algorithmen deutlich geschlagen, was auf eine korrekte und zielführende Implementation von PSSR und PPSSR hinweist.

- Während PPSSR im Fall „Mit Vorhersage existierender Elemente“ knapp die höchste Vorhersagegenauigkeit erzielt, so schneidet er im Fall „Keine Vorhersage existierender Elemente“ deutlich schlechter ab und befindet sich nur auf dem Niveau der Vorhersage durch eine Häufigkeitsverteilung. Dies zeigt die Bedeutsamkeit der Einbeziehung des `excl_elem` Parameters: PPSSR kann bessere Vorhersageergebnisse PSSR erzielen, aber nicht in jeder Situation.

- Die Standardabweichung ist über alle Fälle und Algorithmen verteilt ähnlich gering, es spielt folglich eine geringe Rolle, aus welcher Stelle im Datensatz eine Stichprobe genommen wird.

Die Vorhersageverfügbarkeit (nicht abgebildet), welche nur für PSSR und PPSSR betrachtet wird, ist stets sehr hoch. Im Fall „Mit Vorhersage existierender Elemente“ ist sie immer bei 100%. In dem Fall „Keine Vorhersage existierender Elemente“ liegt sie sowohl für PSSR als auch PPSSR zwischen 90%–95%. Dies zeigt, dass die gewählte Strategie, möglichst viele Regeln zu generieren und die Auswahl der Regel über das Scoring vorzunehmen, funktionierte.

Es konnte bewiesen werden, dass die gezeigte Implementation der Verfahren unter Verwendung von Sequenzregeln eine geeignete Methode ist, um Interaktionen mit enthaltenen Suchfiltern für Besucher eines Datenportals vorherzusagen.

6.7 Einfluss der Parameter

Es wird der Einfluss der verwendeten Parameter auf das Vorhersageergebnis betrachtet. Dabei wird einerseits betrachtet, wie mehr oder weniger Trainingsdaten die Vorhersage

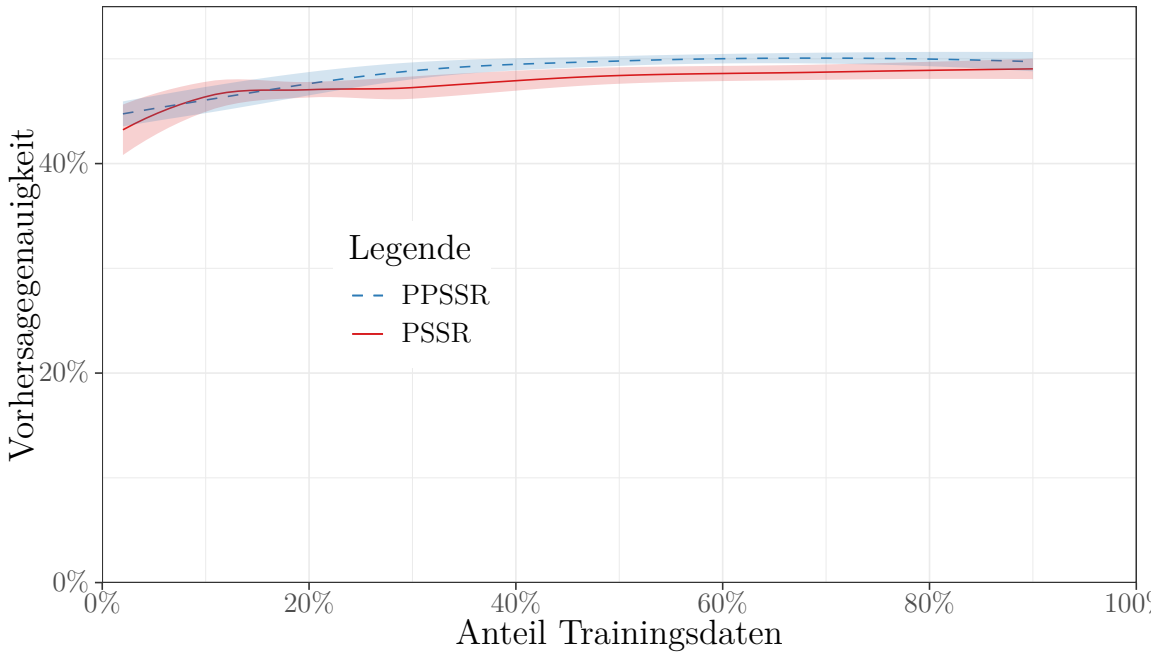


Abbildung 6.4: Abhängigkeit der Vorhersagegenauigkeit von der Menge der Trainingsdaten. Die hinterlegte Fläche zeigt die Standardabweichung. Die Grafik besteht aus 20 Einzelmesswerten, welche leicht geglättet wurden.

beeinflussen. Dies ist wichtig um auszuschließen, dass die Algorithmen durch einen zu kleinen Trainingsdatensatz limitiert wurden.

Weiterhin erfolgt eine Vorstellung der Hyperparameter, welche sich als optimal für die jeweiligen Modelle herausstellten.

6.7.1 Umfang der Trainingsdaten

Es ist zu prüfen, ob die Vorhersagegenauigkeit durch eine mangelnde Größe des Datensatzes beschränkt wurde. Abbildung 6.4 zeigt die Vorhersagegenauigkeit abhängig vom Anteil der Trainingsdaten am gesamten Datensatz. Um die Grafik zu erzeugen, wurden die Vorhersagealgorithmen mit der Konfiguration „Keine Vorhersage bestehender Elemente“ und einer Suffixgröße von eins mit den jeweils optimalen Hyperparametern ausgeführt. Dabei wurde für jeden Durchlauf der Anteil der Trainingsdaten vom Gesamtdatensatz variiert. Es wurde für jeden Durchlauf eine Kreuzvalidierung durchgeführt.

Die Kurven steigen anfangs bei einem Anteil der Trainingsdaten von 2% bis etwa 15%

deutlich an. In diesem Bereich ist auch die Streuung am größten. Ab etwa 20% Trainingsdatenanteil verbessert sich die Vorhersagegenauigkeit nur noch wenig, ab 50% fast gar nicht mehr. Daraus lässt sich schließen, dass der gewählte Anteil der Trainingsdaten bei dieser Evaluation mit 80% groß genug war, um eine verwertbare Aussage zu erstellen. Weiterhin zeigt sich, dass ein längerer Aufzeichnungszeitraum des EGP-Datensatzes keinen Einfluss auf die Ergebnisse hätte[1].

6.7.2 Optimale Hyperparameter

algorithm	excl_elem ●	suffix_size ●	min_confidence ▲	window_size ▲	lhs_size ▲	rhs_size ▲	c1 ▲	prefix_size ▲	mean ■	std ■
PPSSR	✓	1	50	6	1	1	0.5	8	50.53	1.36
		2	50	7	1	1	0.5	7	61.96	1.47
		3	50	6	1	1	0.5	8	68.97	1.58
	×	1	50	8	3	1	0.5	2	22.20	0.55
		2	50	8	3	1	0.5	2	31.88	0.68
		3	50	8	3	1	0.5	2	36.47	0.83
PSSR	✓	1	50	5	4	1	0.5	8	48.11	1.73
		2	50	5	4	1	0.5	8	59.48	1.69
		3	50	5	4	1	0.5	8	66.36	1.91
	×	1	50	7	2	1	0.5	5	26.61	0.86
		2	50	8	2	1	0.5	5	39.29	1.23
		3	50	8	2	1	0.5	5	45.80	1.54

Tabelle 6.2: Optimale Hyperparameter und Vorhersageergebnisse.
● Konfigurationsparameter
▲ Hyperparameter
■ Vorhersagegenauigkeit nach Kreuzvalidierung

In Tabelle 6.2 sind die Modelle mit optimalen Hyperparametern ▲, unterschieden nach den Konfigurationsparametern ● und dem verwendeten Vorhersageverfahren, gelistet. Weiterhin wird die erzielte Vorhersagegenauigkeit ■ nach der Kreuzvalidierung, unterteilt in Mittelwert

[1] Unter der Annahme, dass keine saisonalen Effekte bei der Benutzung des EGP auftreten

und die Standardabweichung, gezeigt[2].

Folgende Aussagen lassen sich anhand der vorliegenden Daten treffen:

- Die optimalen Hyperparameter der Vorhersageverfahren weichen zum Teil deutlich voneinander ab. Wenn PPSSR mit PSSR bei gleicher Konfiguration verglichen wird, zeigt sich, dass andere Hyperparameter benötigt werden. Dies bedeutet auch, dass bei Wechsel des Vorhersagealgorithmus von PPSSR auf PSSR (oder umgekehrt) eine erneute Optimierung der Hyperparameter notwendig ist.

- Eine Variation des `excl_elem` Parameters, welcher eine Veränderung der Verteilung der Elemente im Datensatz simuliert, erfordert andere Hyperparameter.

- Eine Variation der Suffixgröße `suffix_size` hat keinen oder nur minimalen Einfluss auf die optimalen Hyperparameter. Dies bedeutet, dass bei Ändern der Suffixgröße keine erneute Suche nach optimalen Hyperparametern durchgeführt werden muss.

- Die optimale minimale Konfidenz `min_confidence` beträgt für den betrachteten Datensatz immer 50%.

- Der optimale Wert des Parameters `rhs_size` beträgt immer 1. Die Vorhersage mehrerer nachfolgender Elemente gleichzeitig sorgt somit für eine geringere Vorhersagegenauigkeit.

- Der Parameter c_1, welcher zur Gewichtung des Konfidenz-Werts bei der Regelauswahl verwendet wurde, hat stets einen optimalen Wert von 0,5. Das bedeutet, dass für den vorliegenden Datensatz eine Gleichgewichtung des Konfidenz- und Supportwerts die besten Vorhersageergebnisse erzielt.

[2]Die Vorhersageergebnisse entsprechen der Darstellung in Abbildung 6.3, dort wurden sie auch diskutiert

7 Fazit

Ziel dieser Arbeit war es, Nutzer von Datenportalen gezielter zu den gesuchten Datensätzen zu führen, in dem eine Vorhersage über zukünftige Nutzerinteraktionen getätigt wird. Diese wird eingesetzt, um Vorschläge für Suchfilter zu erstellen. So kann dem Nutzer eine effektivere Suche ermöglicht werden.

In der Arbeit wurde ein Überblick über mögliche Vorhersageverfahren gegeben. Anschließend erfolgte eine Bewertung der Verfahren. Die Sequenzregelanalyse erwies sich am geeignetesten. Dies begründet sich einerseits durch die Nachvollziehbarkeit der Vorhersage, wodurch eine einfache Überwachung und Anpassung des Vorhersageprozesses möglich ist. Anderseits zeigte sich im Vergleich zu anderen Verfahren die problemlose Abbildung des vorhandenen Datensatzes auf das geforderte Eingabeformat.

Es wurden zwei verschiedene Verfahren, welche Sequenzregeln verwenden, eingesetzt. Das erste Verfahren verwendet partiell sortierte Sequenzregeln. Das zweite Verfahren, welches priorisierte partiell sortierte Sequenzregeln nutzt, ist eine in dieser Arbeit vorgestellte Weiterentwicklung des ersten Verfahrens. Es erzeugt in mehreren Durchläufen anhand verschiedener Parameter Sequenzregeln, um Muster verschiedener Länge zuverlässiger zu finden.

Es wurde ein Prozess entwickelt, der die Vorhersageverfahren anwendet. Dazu werden Aufzeichnungen vergangener Sitzungen und der Interaktionsverlauf einer neuen, unbekannten Sitzung als Eingabe genutzt. Daraus wird die folgende Interaktion mit enthaltenen Suchfiltern vorhergesagt. Das Training erfolgt anhand von Aufzeichnungen von früheren Benutzersitzungen, wie sie etwa aus Logdaten gewonnen werden können.

Zwar können durch dieses Vorgehen besonders beliebte, beziehungsweise häufig verwendete Suchfilter vorhergesagt werden. Es ist aber zu beachten, dass ein Training anhand historischer Interaktionen nicht zwangsläufig die besten oder effizientesten Filter vorschlägt.

Um die Vorhersage evaluieren zu können, wurde der Prozess anhand eines vorliegenden

Datensatzes, welcher Aufzeichnungen von Interaktionen von Nutzern des EOWEB GeoPortals enthält, geprüft. Dazu wurde der Datensatz in Test- und Trainingsdaten geteilt und die Vorhersage mit der tatsächlich folgenden Interaktion verglichen. Der Anteil der richtigen Vorhersagen wird durch die Vorhersagegenauigkeit beschrieben.

Die Prüfung der Vorhersagegenauigkeit zeigt, dass die in der Arbeit konzipierte Vorhersage unter Einsatz von Sequenzregeln bessere Ergebnisse als die eingesetzten Vergleichsalgorithmen erzielten. Es erfolgt eine individuell an den Nutzer angepasste Ausgabe, welche von den vorherigen Interaktionen abhängig ist. Somit kann das Ziel der Arbeit als erfüllt betrachtet werden.

In der Arbeit konnte bereits das Verfahren mit der höchsten Vorhersagegenauigkeit ermittelt werden. Ein Bestandteil zukünftiger Forschung dagegen ist der Einsatz dieses Verfahrens in einem Datenportal, um eine Echtzeit-Vorhersage der Suchfilter durchzuführen. Eine Möglichkeit wäre der Einsatz im EOWEB GeoPortal.

Um den Erfolg der in dieser Arbeit entwickelten Vorhersagen zu maximieren, ist zu untersuchen, wie diese in die Benutzerführung einbezogen werden können. Es könnte die Methode der Nutzerbefragung eingesetzt werden, um eine genaue Rückmeldung zu erhalten, wann vorgeschlagene Suchfilter erfolgreich angewendet werden konnten.

Das entwickelte Verfahren, welches priorisierte partiell sortierte Sequenzregeln zur Vorhersage nutzt, wurde in dieser Arbeit nur anhand eines Datensatzes untersucht. In zukünftiger Forschung kann geprüft werden, welche weiteren Anwendungsgebiete sich für dieses Verfahren ergeben.

Auch wenn die Vorhersage von Suchfiltern ein Beitrag für bessere Suchergebnisse in Datenportalen ist, behebt dies nicht alle Ursachen, die dazu führen, dass Nutzer die Suche abbrechen oder nicht das gewünschte Ergebnis antreffen. Die vorgestellte Sequenzregelanalyse kann jedoch auch verwendet werden, um ein besseres Datenverständnis zu erlangen und somit verschiedene Gründe für Abbrüche genauer zu analysieren. Diese Erkenntnisse können genutzt werden um die Suche in Datenportalen weiter optimieren zu können.

Literatur

Ahn, J.-w. u. a. (2007). „Open user profiles for adaptive news systems: Help or harm?“ In: *16th International World Wide Web Conference*

Anam, R., C. K. Ho und T. Y. Lim (2014). „Tree Adapt: Web Content Adaptation for Mobile Devices“. In: *International Journal of Information Technology and Computer Science (IJITCS)*

Baer, J. (2015). „How Apache Drives Music Recommendations At Spotify“. In: *Apache: Big Data 2015*

Basu Roy, S. u. a. (2008). „Minimum-Effort Driven Dynamic Faceted Search in Structured Databases“. In: *Proceedings of the 17th ACM Conference on Information and Knowledge Management*. CIKM '08. Napa Valley, California, USA: Association for Computing Machinery, S. 13–22. ISBN: 9781595939913

Brückner, M. und T. Scheffer (1. Jan. 2013). *Unüberwachtes Lernen: Clustern von Attributen*.
`https://www.cs.uni-potsdam.de/ml/teaching/ws13/ida/Frequent-Itemset-Suche.pdf`

Chantamunee, S., K. Wong und C. Fung (Dez. 2020). „An exploration of user-facet interaction in collaborative-based personalized multiple facet selection“. In: *Knowledge-Based Systems* 209

Chierichetti, F. u. a. (Apr. 2012). „Are web users really Markovian?“ In: *WWW'12 - Proceedings of the 21st Annual Conference on World Wide Web*

Claesen, M. und B. De Moor (Feb. 2015). „Hyperparameter Search in Machine Learning“. In: *MIC 2015: The XI Metaheuristics International Conference*

Deshpande, M. und G. Karypis (Mai 2004). „Selective Markov Models for Predicting Web-Page Accesses“. In: *ACM Trans. Internet Techn.* 4, S. 163–184

Deutsches Zentrum für Luft- und Raumfahrt e.V. (3. Dez. 2020). *DLR - Earth Observation Center*.
`https://www.dlr.de/eoc/desktopdefault.aspx/tabid-8799/`

Dixit, R. u. a. (Nov. 2017). „User needs analysis and usability assessment of DataMed – a biomedical data discovery index“. In: *Journal of the American Medical Informatics Association* 25.3, S. 337–344. ISSN: 1527-974X

DLR EGP (1. Jan. 2020). *METOP GOME-2 - Nitrogen Dioxide (NO2) - Global*. `https://eoweb.dlr.de/guestegp/productDetails/bf8dbf94-ff16-42bf-a957-0e8f80813aff`

Fournier Viger, P., U. Faghihi u. a. (Feb. 2012). „CMRules: Mining sequential rules common to several sequences“. In: *Knowl.-Based Syst.* 25, S. 63–76

Fournier Viger, P., C.-W. Wu, V. Tseng, L. Cao u. a. (2015). „Mining Partially-Ordered Sequential Rules Common to Multiple Sequences“. In: *IEEE Transactions on Knowledge and Data Engineering*

Fournier Viger, P., C.-W. Wu, V. Tseng und R. Nkambou (Mai 2012). „Mining Sequential Rules Common to Several Sequences with the Window Size Constraint“. In: S. 299–304

Fournier-Viger, P., C. Lin u. a. (2016). „The SPMF Open-Source Data Mining Library Version 2.“ In: *Proc. 19th European Conference on Principles of Data Mining and Knowledge Discovery (PKDD 2016) Part III*

Fournier-Viger, P., T. T. Gueniche und V. S. (2012). „Using Partially-Ordered Sequential Rules to Generate More Accurate Sequence Prediction“. In: *Advanced Data Mining and Applications*

Frasincar, F., J. Borsje und L. Levering (2009). „A Semantic Web-Based Approach for Building Personalized News Services“. In: *IJEBR*

Graves, A. u. a. (2009). „A Novel Connectionist System for Unconstrained Handwriting Recognition“. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 31.5, S. 855–868

Gregory, K. u. a. (2020). „Lost or Found? Discovering Data Needed for Research“. In: *Harvard Data Science Review*

Gueniche, T., P. Fournier Viger und V. Tseng (Dez. 2013). „Compact Prediction Tree: A Lossless Model for Accurate Sequence Prediction“. In: Bd. 8347. ISBN: 978-3-642-53916-9

Han, J., M. Kamber und J. Pei (2012). *Data Mining – Concepts and Techniques*

Hansen, T. (2008). *RFID-gestützte Produktempfehlung im stationären Einzelhandel*. Wirtschaftsinformatik - Theorie und Anwendung. Logos-Verlag. ISBN: 9783832519568

Hapke, H. und C. Nelson (2020). *Building Machine Learning Pipelines*. O'Reilly Media. ISBN: 9781492053163

Heyer, G. u. a. (2011). „Interaktive explorative Suche in großen Dokumentbeständen“. In: *Datenbank-Spektrum*, S. 195–206

Hochreiter, S. und J. Schmidhuber (Dez. 1997). „Long Short-term Memory“. In: *Neural computation* 9, S. 1735–80

Hornik, K., B. Grün und M. Hahsler (Okt. 2005). „arules - A Computational Environment for Mining Association Rules and Frequent Item Sets“. In: *Journal of Statistical Software* 14

Ibrahim, R. und M. O. Shafiq (2019). „On Predicting Taxi Movements Modes in Porto City Using Classification and Periodic Pattern Mining“. In: *2019 IEEE 21st International Conference on High Performance Computing and Communications; IEEE 17th International Conference on Smart City; IEEE 5th International Conference on Data Science and Systems (HPCC/SmartCity/DSS)*, S. 1197–1204

Jamshed, A., B. Mallick und P. Kumar (Nov. 2020). „Deep learning-based sequential pattern mining for progressive database“. In: *Soft Computing* 24

Jayalal, S., C. Hawksley und P. Brereton (Jan. 2007). „Website link prediction using a Markov chain model based on multiple time periods“. In: *Int. J. Web Eng. Technol.* 3, S. 271–287

Kennedy, A. (13. Juli 2020). *The Definitive Guide to the Difference Between Filters and Facets*.
`https://www.sajari.com/blog/the-difference-between-filters-and-facets`
Abruf: 02. 03. 2021

Kern, D. und B. Mathiak (2015). „Are There Any Differences in Data Set Retrieval Compared to Well-Known Literature Retrieval?“ In: *Research and Advanced Technology for Digital Libraries*. Hrsg. von S. Kapidakis, C. Mazurek und M. Werla. Cham: Springer International Publishing, S. 197–208. ISBN: 978-3-319-24592-8

Kim, Y. u. a. (2014). „Modeling Dwell Time to Predict Click-Level Satisfaction“. In: *Proceedings of the 7th ACM International Conference on Web Search and Data Mining*. WSDM '14. New York, New York, USA: Association for Computing Machinery, S. 193–202. ISBN: 9781450323512

Konstan, J. A. u. a. (März 1997). „GroupLens: Applying Collaborative Filtering to Usenet News“. In: *Commun. ACM* 40.3, S. 77–87. ISSN: 0001-0782

Marchionini, G. (2006). „Exploratory search: from finding to understanding“. In: *Communications of the ACM*

Martín Abadi u. a. (2015). *TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems*. Software available from tensorflow.org

McAlpine, K., E. Miranda und S. Hoggar (1999). „Making music with algorithms: A case-study system“. In: *Computer Music Journal* 23.2, S. 19–30

Molnar, A. und C. H. Muntean (2019). „Paving the Way for 5G Through the Convergence of Wireless Systems“. In: IGI Global. Kap. User Based Adaptive Multimedia Delivery Over 5G Network

Molnar, A., J. Virseda und V. Frias-Martinez (2015). „Insights from EducaMovil: Involving Teachers in Creating Educational Content for Mobile Learning Games“. In: *Journal of Interactive Learning Research*

Murdoch, W. und A. Szlam (Feb. 2017). „Automatic Rule Extraction from Long Short Term Memory Networks“. In

Niu, X., X. Fan und T. Zhang (Jan. 2019). „Understanding Faceted Search from Data Science and Human Factor Perspectives“. In: *ACM Transactions on Information Systems* 37, S. 1–27

Paine, T. u. a. (Dez. 2013). „GPU Asynchronous Stochastic Gradient Descent to Speed Up Neural Network Training“. In: *ICLR 2014*

Park, S. H. u. a. (2018). „Sequence-to-Sequence Prediction of Vehicle Trajectory via LSTM Encoder-Decoder Architecture“. In: *2018 IEEE Intelligent Vehicles Symposium (IV)*, S. 1672–1678

Pedregosa, F. u. a. (2011). „Scikit-learn: Machine Learning in Python“. In: *Journal of Machine Learning Research* 12, S. 2825–2830

Petrescu, P. (2014). *Google Organic Click-Through Rates in 2014*. `https://moz.com/blog/google-organic-click-through-rates-in-2014`

Pitman, A. und M. Zanker (Aug. 2011). „An Empirical Study of Extracting Multidimensional Sequential Rules for Personalization and Recommendation in Online Commerce“. In: *10. Internationale Tagung Wirtschaftsinformatik*

Poulos, M., N. Korfiatis und S. Papavlassopoulos (2019). „Assessing Stationarity in Web Analytics: A study of Bounce Rates“. In: *Expert Systems*

Rahdari, B., P. Brusilovsky und D. Babichenko (2020). „Personalizing Information Exploration with an Open User Model“. In: *Hypertext 2020*

Resnick, P. und H. R. Varian (März 1997). „Recommender Systems“. In: *Commun. ACM* 40.3, S. 56–58. ISSN: 0001-0782

Rosvall, M. u. a. (Aug. 2014). „Memory in network flows and its effects on spreading dynamics and community detection“. In: *Nature communications* 5, S. 4630

Sak, H., A. Senior und F. Beaufays (Jan. 2014). „Long short-term memory recurrent neural network architectures for large scale acoustic modeling“. In: *Proceedings of the Annual Conference of the International Speech Communication Association, INTERSPEECH*, S. 338–342

Sarukkai, R. (Juni 2000). „Link Prediction and Path Analysis using Markov Chains“. In: *Computer Networks* 33, S. 377–386

Schindler, S., M. Paradies und A. Twele (2019). „Here is my Query, where are my Results? A Search Log Analysis of The EOWEB Geoportal“. In: *Proceedings of 2019 Big Data from Space*

Singh, H., M. Kaur und P. Kaur (März 2017). „Web page recommendation system based on partially ordered sequential rules“. In: *Journal of Intelligent & Fuzzy Systems* 32, S. 3009–3015

Stinner, F. u. a. (Nov. 2019). „Takeshi: Application of unsupervised machine learning techniques for topology detection in building energy systems“. In: *Journal of Physics: Conference Series* 1343, S. 012041

team, T. pandas development (Feb. 2020). *pandas-dev/pandas: Pandas*. Version latest

Terpilowski, M. (3. März 2021). *mchmm*.
`https://github.com/maximtrp/mchmm`

Thierry, N. (2005). *FIDIS Deliverable - D2.3: Models*.
`http://www.fidis.net/resources/fidis-deliverables/identity-of-identity/`

Velez, M. u. a. (Sep. 2013). „Factors associated with HIV Prognosis in Rural Uganda: Sequence-mining the Medical Record“. In: *Journal of Health Informatics in Africa* 1.1

Wongchokprasitti, C. u. a. (2015). „User Model in a Box: Cross-System User Model Transfer for Resolving Cold Start Problems“. In: *UMAP 2015*

Xu, Y. und D. Mease (Juli 2009). „Evaluating Web Search Using Task Completion Time“. In: *SIGIR '09: Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*, S. 676–677

Xu, Z. (Feb. 2016). „Propagation Routes Analysis of HPAI Outbreaks using Sequential Pattern Mining“. In: *The fourth International Conference on Information Science and Cloud Computing*, S. 022

Yang, D.-L., Y.-L. Hsieh und J. Wu (Jan. 2006). „Using Data Mining to Study Upstream and Downstream Causal Relationship in Stock Market“. In: *Proceedings of the 2006 Joint Conference on Information Sciences*

Zaki, M. (Jan. 2000a). „Sequence Mining in Categorical Domains: Algorithms and Applications“. In: Bd. 1828, S. 162–187. ISBN: 978-3-540-41597-8

– (Jan. 2000b). „Sequence Mining in Categorical Domains: Incorporating Constraints“. In: S. 422–429

– (Jan. 2001). „Zaki, M.J.: SPADE: An efficient algorithm for mining frequent sequences. Machine Learning 42(1), 31-60“. In: *Machine Learning* 42, S. 31–60

Abbildungsverzeichnis

2.1 Beispiel für eine Transaktionsdatenbank 7
2.2 Beispiel Markov-Kette 14
2.3 Schematische Darstellung eines Künstlichen Neuronalen Netzes 16
2.4 EOWEB Benutzeroberfläche 19
2.5 Vorschau der Stickstoffdioxid-Messwerte des METOP-Satelliten 20
2.6 Beispiel eines Systems mit offenem Benutzermodell 23

3.1 Vorhersageverfahren 27

4.1 Verteilung der Interaktionstypen 34
4.2 Histogramm Interaktionen pro Sitzung 35
4.3 Verteilung der Anzahl von Interaktionen pro Sitzung abhängig von der Sitzungsdauer 36

5.1 Schema PSSR 42
5.2 Schema PPSSR 46

6.1 Einfluss der Vorfilter-Schritte der Interaktionen auf Konfidenz und Support der Sequenzregeln 50
6.2 Kreuzvalidierungsverfahren 56
6.3 Vorhersagegenauigkeit der einzelnen Algorithmen 58
6.4 Abhängigkeit der Vorhersagegenauigkeit von der Menge der Trainingsdaten 60

Tabellenverzeichnis

2.1 Vergleich Suchfilter mit Facetten 5

3.1 Zuordnung der Begriffe der Transkationsdatenbank zu den Anforderung des Datensatzes 29

6.1 Verwendete Parametergrößen 52
6.2 Optimale Hyperparameter und Vorhersageergebnisse 61

Selbstständigkeitserklärung

Ich erkläre, dass ich die vorliegende Arbeit selbstständig und nur unter Verwendung der angegebenen Quellen und Hilfsmittel angefertigt habe.

Seitens des Verfassers bestehen keine Einwände die vorliegende Masterarbeit für die öffentliche Benutzung im Universitätsarchiv zur Verfügung zu stellen.

Jena, den 17. März 2021
Tim Surber