Hochschule für Angewandte Wissenschaften Hamburg
Hamburg University of Applied Sciences

Masterarbeit

Eduard Weigandt

Auf Data-Mining basierende Personalisierung im E-Commerce mit implizitem Feedback

Fakultät Technik und Informatik
Studiendepartment Informatik

Faculty of Engineering and Computer Science
Department of Computer Science

Eduard Weigandt

Auf Data-Mining basierende Personalisierung im E-Commerce mit implizitem Feedback

Masterarbeit eingereicht im Rahmen der Abschlussprüfung

im Studiengang Master of Science in Informatik
am Department Informatik
der Fakultät Technik und Informatik
der Hochschule für Angewandte Wissenschaften Hamburg

Betreuender Prüfer: Prof. Dr. Kai von Luck
Zweitgutachter: Prof. Dr. Klaus-Peter Schoeneberg

Eingereicht am: 15. Dezember 2016

Eduard Weigandt

Thema der Arbeit

Auf Data-Mining basierende Personalisierung im E-Commerce mit implizitem Feedback

Stichworte

E-Commerce, Data-Mining, RecSys2015, Yoochoose, Implizites Feedback, Random Forest, Gradient Boosting, XDGBoost, Scikit-Learn

Kurzzusammenfassung

Die vorliegende Masterarbeit beschreibt die Herangehensweise zur Analyse eines kommerziellen Datensatzes, der sich aus den Aufzeichnungen von Nutzer- und Kaufverhalten auf einer unbekannten E-Commerce Plattform zusammensetzt. Das hierbei verfolgte Ziel besteht aus der Klassifzierung von Artikeln mit Data-Mining Verfahren, die dann dazu genutzt werden um Vorhersagen für weitere Käufe zu erstellen. Dafür werden zwei Ensemble Verfahren bestehend aus einem Random Forest und Gradient Boosting verglichen. In Hinsicht auf die Größe und die Unausgeglichenheit in der Verteilung von Käufen ist der verwendete Datensatz besonders und muss dementsprechend vor der Verwendung angepasst werden.

Eduard Weigandt

Title of the paper

Data-Mining based Personalization in E-Commerce with implicit Feedback

Keywords

E-Commerce, Data-Mining, RecSys2015, Yoochoose, Implicite Feedback, Random Forest, Gradient Boosting, XDGBoost, Scikit-Learn

Abstract

This master thesis describes the approach to the analysis of a commercial dataset, which consists of the consumer behavior from an unknown e-commerce platform. The objective here is to classify articles with data mining methods, then using the resulting models to make predictions for further purchases. For this purpose, two ensemble methods consisting of a random forest and gradient boosting are compared. The used dataset is challenging, because of the imbalanced distribution of purchases and the volume of data.

Inhaltsverzeichnis

1 Einleitung

Die Bedeutung von E-Commerce Plattformen in der heutigen Zeit ist unbestreitbar groß. Das Internet bietet die Möglichkeit auf einfache und schnelle Weise ein eigenes Onlineunternehmen auf die Beine zu stellen. Durch die geminderte Hürde entsteht eine hohe Vielfalt an Vertriebsplattformen, die ihre Waren an den Kunden bringen wollen. Jedoch reicht es dadurch auch nicht mehr aus ein gutes Angebot zu präsentieren, um sich von der Masse abzusetzen. Traditionelle Geschäfte können nur bedingt gut eine ausführliche Analyse ihrer Kunden machen und sind darauf beschränkt über verschiedene Arten der freiwilligen Auskunft wie z.B. über Befragungen an Daten zu kommen. Unternehmen im Internet sind diesbezüglich freier und haben eine breite Auswahl an Informationsquellen, die sie zum besseren Verstehen ihrer Kunden mit Hilfe von *Data-Mining* verwenden können. In diesem Kontext bieten Empfehlungssysteme viele Vorteile, um z.B. bestehende Kunden zu neuen Käufen anzuregen oder durch das Entdecken von Kauftrends neue Kunden für sich zu gewinnen (Ekstrand u. a., 2011). Dadurch kann man sich einfacher von der Maße absetzen. Im Groben wird dem Kunden so die Erkundung des Sortiments durch Aufbereitung von relevanten Informationen vereinfacht. Das Problem dahinter kennt man unter dem Namen des *Information Overload*. Dieses ist jedoch nur ein Teilaspekt der Anforderungen von Empfehlungssystemen die man betrachten kann.

Anforderungen vom Kunden In Said u. a. (2012) werden drei wichtige Bereiche von denen aus relevante Anforderungen ausgehen vorgestellt. Der eine besteht wie schon erwähnt in der Zufriedenheit eines Kunden bei der Erfüllung seiner Aufgaben, durch die gezielte Assistierung bei der Exploration von Informationen. Dies geschieht durch die Verbesserung der Qualität in den bereitgestellten Informationen oder im Auffinden von für den Kunden interessanten Artikeln. Durch eine bessere Auswahl an Artikeln kann man jedoch nicht immer die Zufriedenheit der Kunden steigern, da auch eine größere Auswahl an potentiell relevanten Daten den Nutzer überfordern kann (Bollen u. a., 2010).

Perspektive vom Unternehmen Eine andere Perspektive sieht man in den Anforderungen eines Unternehmens, welches primär eine Steigerung der Umsätze verfolgt. Je

nachdem wie das zugrundeliegende Geschäftsmodel aussieht, muss man dafür die Algorithmen im Empfehlungssystem passend auswählen. Damit die strategischen Ziele des Unternehmens widergespiegelt werden (Said u. a., 2012).

Technische Bedingungen Der letzte Bereich besteht aus den technischen Anforderungen an ein solches System. Dazu zählen Abhängigkeiten in den Daten oder der Infrastruktur des Systems, die die Möglichkeiten zur Skalierung oder Robustheit[1] einschränken können (Said u. a., 2012). Im Internet gibt es die Möglichkeit die Interaktionen des Nutzers aufzuzeichnen oder seine direkte Meinung über die Bewertungen abzufragen. Diese Möglichkeiten stehen z.B. einem Fernsehsender nicht in diesem Ausmaß zur Verfügung, was sich wiederum einschränkend auf die Erstellung von Empfehlungen auswirkt. Für eine Online-Plattform wiederum können viele verschiedene Datenquellen gewählt werden, die direkt oder indirekt eine Bewertung vom Kunden darstellen. Die indirekte Variante steht dabei schon von Anfang an zur Verfügung.

Verbindet man alle aufgestellten Anforderungen, so entsteht ein komplexes Problem, welches man mit unterschiedlichen Herangehensweisen bearbeiten kann. Für die hier gemachte wissenschaftliche Arbeit sind besonders die Eigenschaften von implizitem Feedback in Verbindung mit *Data-Mining* Verfahren von Interesse.

1.1 Motivation

Die hier gemachte Arbeit untersucht einen großen Datensatz aus dem E-Commerce Bereich anhand bestimmter wirtschaftlicher Zielsetzungen. Die mit Hilfe von *Machine Learning* Verfahren beantwortet werden. Die erbrachte Leistung in dieser Arbeit besteht in der Untersuchung der Herangehensweise an das gegebene Problem unter Verwendung vom impliziten Feedback und dem Vergleich bestimmter Verfahren.

Die Beschreibung der wichtigen Konzepte im Themengebiet der Empfehlungen ebnet mögliche Ansätze für die Vorhersage von Artikeln mit klassischen Data-Mining Verfahren. Des Weiteren ist die Beantwortung der folgenden Fragen: ”Wird der Kunde kaufen?”

[1] Die Fähigkeit mit Fehlern im System umzugehen.

und ”Was wird der Kunde kaufen?” mit großen wirtschaftlichen Folgen für die strategische Ausrichtung eines E-Commerce Unternehmens verbunden. Denn daraus lassen sich passende Empfehlungen extrahieren, die zu weiteren Käufen anrege könnten.

Deswegen ist die gezielte Untersuchung des Prozesses bei der Erstellung eines Modells ein wichtiger Beitrag für einen solchen Datensatz. Dabei wird sich auch die Bedeutung von interpretierbaren Algorithmen für das Finden neuer Merkmale des Datensatzes angeschaut.

1.2 Ziele & Aufgaben

Das Ziel der hier vorliegenden Arbeit besteht in der Datenanalyse zur Ermittlung von Käufen und der Theorie zur Erstellung von Artikel-Empfehlungen. Das Hauptaugenmerk liegt dabei auf dem *Prozess* zur Erstellung eines Modells im E-Commerce Bereich auf Basis von aufgezeichneten Nutzerverhalten. Der verwendete Datensatz kommt aus der *RecSys 2015 Challenge*[2] und beinhaltet Daten über gemachte Klicks und Käufe von einem unbekannten Onlineshop.

Die grundlegende Methodik hinter der Extraktion von neuem Wissen geschieht mit Hilfe der Theorie hinter *Knowledge Discovery in Databases* (KDD) (Fayyad u. a., 1996), genauer gesagt die Techniken aus dem Teilschritt des *Data-Mining* (DM). Es werden auch neuere Einflüsse aus dem Gebiet der *Data Science* berücksichtigt, die neue Wege für die Herangehensweise aufzeigen. Zur Erfüllung der Hauptziele wird die Eignung von implizitem Feedback in Verbindung mit *Data-Mining* im gegebenen Kontext einer E-Commerce Plattform untersucht. Dafür wird kein eigenständiges Empfehlungssystem gebaut jedoch werden die relevanten Teile bei der Berechnung vorgestellt und kurz mit klassischen Ansätzen aus dem *Data-Mining* verglichen.

Dazu gehört das Studium von dem aktuellen Stand der Forschung sowie dem kritischen Vergleich von bestehenden Lösungen. Beim Prozess des *Data-Minings* sind die folgenden Fragen, die jedoch nicht komplett in dieser Arbeit verfolgt werden, von Interesse:

[2] `http://2015.recsyschallenge.com/` (05.09.2016)

- Welche Methoden eignen sich zur Exploration oder Analyse der Daten?
- Wie unterscheiden sich diese Methoden untereinander?
- Wie findet man Nutzerinteraktionen die zu einem Kauf führen?
- Wie geht man mit großen Datenmengen um?
- Sind die gemachten Vorhersagen erklärbar?
- Wie bewertet man die Vorhersage von Artikeln?
- Wie verändern sich die Vorlieben über die Zeit in den aufgezeichneten Daten?

1.3 Gliederung

In Kapitel 2 werden die relevanten Themengebiete für das zu bearbeitende Thema genauer beleuchtet und analysiert. Dazu zählt zum einen der aktuelle Stand der Forschung bei der Erstellung von Empfehlungen, welcher aufbauend auf klassischen *Data-Mining* Verfahren neue Wege eröffnet. Zum anderen auch eine kurze Übersicht verschiedener Konzepte aus dem *Data-Mining* Umfeld, die bisher im Bereich des E-Commerce mit implizitem Feedback eingesetzt werden. Dies beinhaltet auch die derzeitigen Ansätze für den hier eingesetzten Datensatz aus dem *RecSys 2015 Wettbewerb*.

In Kapitel 3 wird der Aufbau und das Vorgehen bei der Exploration des verwendeten Datensatzes vorgestellt und erläutert. Dazu zählen erste statistische Auswertungen und die daraus resultierenden Merkmale, welche für die nachfolgende *Daten Analyse* eine Rolle spielen. Dazu zählt auch das Erstellen erster Hypothesen zu dem angeschauten Datensatz.

Mit Hilfe der Erkenntnisse aus den vorherigen Kapiteln werden dann in Kapitel 4 Modelle für die gegebenen Daten erstellt, um einen kritischen Vergleich zwischen den vorgestellten Verfahren aus Kapitel 2 zuziehen. Die Priorität liegt dabei in der Beantwortung der in Abschnitt 1.2 und Kapitel 3 aufgestellten Fragen mit Hilfe von *Data-Mining* und in der Untersuchung von bestehenden Funden.

2 Literaturauswertung und Problemanalyse

Das in dem nun folgenden Kapitel vorgestellte Wissen bildet die Basis für das weitere Vorgehen und die darauf aufbauenden Entscheidungen. Des Weiteren findet eine Abgrenzung von den nicht relevanten Themengebieten statt, um den Fokus auf die gesetzten Ziele nicht zu verlieren.

2.1 Personalisierung im E-Commerce

Die hier gemachte wissenschaftliche Arbeit beschäftigt sich nicht mit den wirtschaftlichen oder ethischen Aspekten von Empfehlungssystem, jedoch ist es wichtig die Domäne in der man sich befindet zu definieren. Dazu zählt es den Begriff Personalisierung zu erklären. In Riecken (2000) wird dieser wie folgt beschrieben. Personalisierung berücksichtigt die Bedürfnisse jedes einzelnen Individuums und dessen Ziele, um eine engere Kundenbindung aufzubauen. Dies bedeutet in einem bestimmten Kontext effizient und mit bekanntem Wissen den Kunden bei der Erfüllung seiner Ziele zu unterstützen. Ein großer Teil der bekannten Forschung zum Thema Personaliserung findet im Gebiet der *Human-Computer Interaction*[1] (HCI) statt (Blom u. Monk, 2003). HCI steht nicht im Fokus dieser Arbeit, jedoch beinhaltet Kapitel 3 die Exploration der Nutzerinteraktionen mit dem System sowie die zeitlichen Abläufe[2]. Darüber hinaus werden keine weiteren Aussagen getroffen, da keine expliziten Informationen wie z.B. Nutzerbefragungen zur Verfügung stehen.

2.1.1 Herausforderungen

Im Gegensatz zu einem traditionellen Geschäft vermisst man übers Internet eine individuelle Kundenlösung, die üblicherweise über den Verkäufer im Kundengespräch erfolgt (Goy u. a., 2007). Zur Bewältigung dieses Problems werden massenweise Daten über

[1] Die Forschung wie Menschen mit Computern interagieren und wie gut oder schlecht unsere Systeme dazu geeignet sind.

[2] Beispielsweise die Dauer einer Session oder die Verweildauer eines Nutzers.

die Kunden gesammelt. Dies wirft neue Fragen über den ethischen Aspekt dieses Vorgehens auf. In Paraschakis (2016) werden dazu mehrere Kategorien benannt, von denen die folgenden für diese Arbeit von Interesse sind:

- **Datensammlung**: Es kann ein Fehlen von Transparenz beim Vorgang des Sammelns und dem Umgang mit Daten bestehen. Je nach Land gibt es unterschiedliche oder auch gar keine Regelungen zur Aufzeichnung von Informationen[3]. Auf den vorliegenden Datensatz bezogen beinhaltet dieser keine genaue Erklärung wie die Daten gesammelt wurden außer das die Klicks und Käufe von Nutzersessions mitgeschnitten wurden (Ben-Shimon u. a., 2015). Hier muss man sich darauf verlassen, dass die Datensammlung mit dem Einverständnis der Nutzer durchgeführt wurde.

- **Benutzerprofile**: Unter dem Gesichtspunkt der Sicherheit kann man die Daten für *Phishing* und *Social Engineering* Angriffe gegen die Nutzer einsetzen. Das führt dazu, dass neue Techniken zum Schutz dieser Daten entwickelt wurden (Canny, 2002). Die Betreiber einer E-Commerce Plattform werden durch die Verfälschung von Empfehlungen mittels falscher Profile angreifbar. Wie auch bei dem vorherigen Punkt kann man keine wirkliche Aussage treffen inwieweit die hier verwendeten Daten irgendwelche Manipulationen aufweisen bzw. soll dies auch nicht Gegenstand der Arbeit sein. Jedoch werden gefundene Auffälligkeiten in den Daten in Kapitel 3 aufgezeigt.

- **Veröffentlichung von Daten**: Auch anonymisierte Daten beinhalten potentiell persönliche Informationen (Narayanan u. Shmatikov, 2006). Wenn man jedoch eine zu aggressive Verschleierung von persönlichen Daten durchführt können wichtige Erkenntnisse für Empfehlungen verloren gehen. Auch in dem hier untersuchten Datensatz werden anonymisierte Daten verwendet, was diesen Aspekt besonders in der Vordergrund rückt. Zu dem erschweren restriktive Datenschutzgesetze oder wirtschaftliche Aspekte[4] die Veröffentlichung von neuen Datensätzen auf

[3] Die EU Richtlinien zum Schutz natürlicher Personen bei der Verarbeitung personenbezogener Daten und zum freien Datenverkehr: `http://eur-lex.europa.eu/legal-content/DE/TXT/HTML/?uri=CELEX:31995L0046&from=en` (2016.12.13)

[4] Der Wettbewerbsvorteile durch die gesammelten Informationen.

denen man aufbauen könnte. Somit sind solche großen Veröffentlichungen von Aufzeichnungen eine Besonderheit.

2.2 Data Mining in Empfehlungssystemen

2.2.1 Empfehlungssysteme

Im Abschnitt zum Thema Empfehlungen mit Matrix Factorization wird beispielhaft eine mögliche Variante zur Berechnung von Empfehlungen anhand von Bewertungen vorgestellt. Eine wichtige Aussage die sich in den letzten Jahren herauskristallisiert hat ist jedoch das *alles eine Empfehlung sein kann* (Amatriain, 2014). Seien es Vorschläge für Dokumente, Musik, Kaufartikel oder Nachrichten die einen interessieren könnten (Ekstrand u. a., 2011). Wirtschaftlich gesehen war das Erstellen von Empfehlungen zur Anfangszeit aufwendig und kostspielig für Unternehmen, weil z.B. das Wissen darüber fehlte. Das hat sich für den E-Commerce mit besseren Computern und spätestens durch die Bemühungen und Veröffentlichungen von z.B. Amazon[5] gewandelt (Linden u. a., 2003). Diese zeigten das durch Empfehlungen eine höhere Kundenbindung und eine bessere Konversion von Besuchern erreicht werden konnte (Marshall, 2006).

2.2.2 Data-Mining

Eine der einfachsten Definitionen von *Data-Mining* ist *die Entdeckung eines Modells für Daten* (Leskovec u. a., 2014). Das genannte Modell in diesem Fall kann viele unterschiedliche Konzepte darstellen. Die besondere Eigenschaft, die beim *Data-Mining* Prozess häufig hervorgehoben wird, ist die Interpretierbarkeit und nicht die Genauigkeit eines erstellten Modells (Murphy, 2012).

Exploite vs Explore Im Kontext von Empfehlungssystemen muss man sich zwischen der Ausnutzung (*Exploite*) von bekannten Vorlieben oder der Erforschung (*Explore*) neuer entscheiden (Lempel, 2012). In den meisten Fällen wird jedoch auf die Genauigkeit einer Empfehlung gesetzt, da die meisten Auswertungen offline mit einem statischen

[5] `http://www.amazon.de` (2016.12.13)

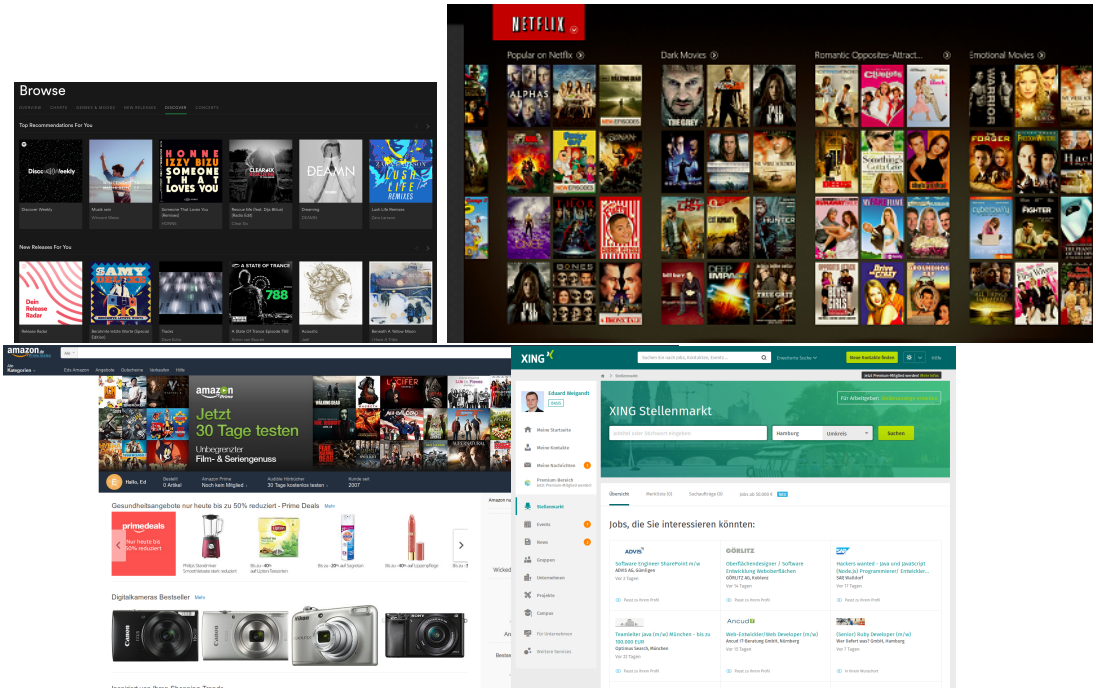

Abbildung 2.1: Systeme mit Empfehlungen: Musik, Filme, Jobs und Artikel

Quelle: `http://spotify.com`, `http://netflix.com`, `http://amazon.com`, `http://xing.com`

Datensatz ausgeführt werden Beel u. a. (2013). Dadurch kann man keine direkte Evaluierung von Ansätzen zur Entdeckung neuer Vorlieben machen.

Modellierung Das *statistische* Modell bietet eine gute Grundlage für die Abbildung von Daten (James u. a., 2013). Es fasst alle wichtigen Informationen der Daten zusammen. Dazu zählt z.B. welche Werte die Daten unter bestimmten Wahrscheinlichkeiten annehmen können. Eine weitere Variante zur Modellierung von Daten besteht im Einsatz von *unüberwachten Machine Learning* Verfahren. Diese eignen sich gut für Daten bei denen man nur wenig Wissen darüber besitzt, welche Bestandteile von den Daten wichtig sind bzw. worauf man den Fokus legen sollte. Im Gegensatz dazu sind überwachte Verfahren gut für das Verstehen von bekannten Daten, die alle Informationen zur Urteilsbildung beinhalten.

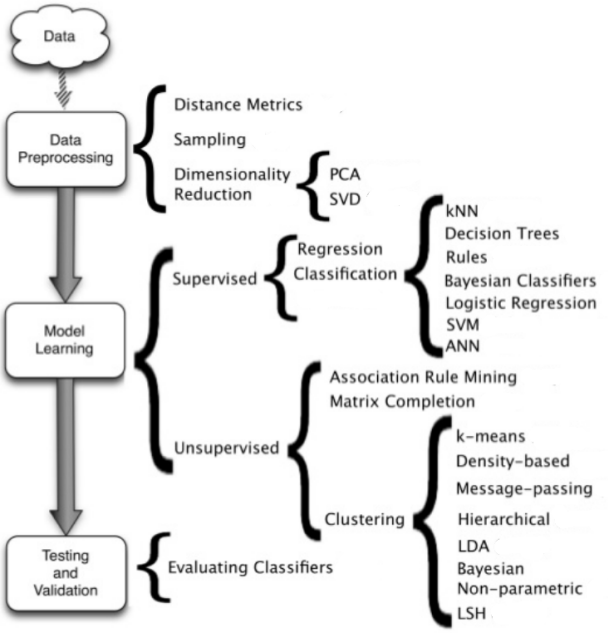

Abbildung 2.2: Klassischer *Data Mining* Prozess und die dazugehörigen Verfahren.

Quelle: Amatriain (2013)

2.2.2.1 Data-Mining Prozess

Der grundlegende Prozess beim *Data-Mining* (DM) besteht aus den nachfolgenden drei Schritten (Amatriain, 2013), die man in Abbildung 2.2 sehen kann. Der erste ist die **Daten Vorverarbeitung** bei der jeweils eine Filterung, Säuberung oder Transformation der aufgezeichneten Daten verfolgt wird (Wickham u. a., 2014). Dies is notwendig, da die Ausgangsform der Daten meistens mit den verwendeten *Machine Learning* Verfahren nicht kompatibel ist. Diese kann unnötige oder fehlerhafte Attribute[6] beinhalten.

Beispiel 1 *In der Sensorik*[a] *kann es vorkommen das Messfehler auftreten, die sich als Ausreißer in den Daten zeigen. Diese sollten aus den Daten gefiltert werden.*

[a] Die Wissenschaft und die Anwendung von Sensoren zur Messung und Kontrolle von Veränderungen von umweltbezogenen, biologischen oder technischen Systemen. (Quelle: `https://de.wikipedia.org/wiki/Sensorik_(Technik)` (2016.12.13))

[6] Aufgezeichnete Daten setzen sich aus Objekten zusammen, denen unterschiedliche Informationen (Attribute) zugeordnet sind.

Transformation der Daten Ein weiterer Problembereich besteht in einer zu großen Menge von gesammelten Daten, die die weitere Verarbeitung behindern können. Aus diesem Grund existieren Techniken, um die Dimension der Daten zu verkleinern (Paterek, 2007) (Bsp. *Sampling*) bzw. zusammenzufassen. Für den hier genutzten Datensatz mit den Nutzerhistorien könnte das die Entfernung von zu kleinen oder nur einmalig auftretenden Interaktionen mit dem System bedeuten, wenn diese keine neuen Informationen für das allgemeine Modell beitragen können. Eines der *erstaunlichsten* Ergebnisse die nach dem *Netflix* Wettbewerb gemacht wurde bestand darin, dass nicht die Menge an Daten sondern die Qualität der Algorithmen hinter dem Modell[7] entscheidend ist. Dadurch wird die Bedeutung des Menschen hervorgehoben, der ein solches Modell oder den passenden Algorithmus mit seinem Fachwissen erstellen muss.

2.2.2.2 Stetige und Kategoriale Merkmale

Die gefundenen Merkmale eines Datensatzes können jeweils auf zwei Arten charakterisiert werden (James u. a., 2013). Es kann zwischen *quantitativen* (stetigen) und *qualitativen* (kategorialen) Merkmalen unterschieden werden. Bei den ersteren redet man von kontinuierlichen Größen, wie z.B. die Anzahl von Artikeln im Warenkorb. Bei den letzteren existiert eine Kategorie denen die Werte angehören, wie z.B. ein männlicher oder weiblicher Kunde. Die *quantitativen* Merkmale werden bevorzugt als Regressions Probleme behandelt und die *qualitativen* als Klassifizierungs Probleme (James u. a., 2013, Seite 28).

Nach Boriah u. a. (2008) ist die Messung der Distanz oder Ähnlichkeit zwischen zwei Datenpunkten eine Kernanforderung von vielen *Data Mining* Methoden. Im Gegensatz zu stetigen Merkmalen ist die Vorstellung von Ähnlichkeit und Distanz bei kategorialen Merkmalen nicht vollständig klar. Die unterschiedlichen Ausprägungen einer Kategorie sind untereinander nicht geordnet und können somit nicht einfach verglichen werden. Aus diesem Grund existieren verschiedene Verfahren, die entweder stetige oder kategoriale Merkmale verwenden. Es gibt allerdings auch solche bei denen diese Unterscheidung nicht gemacht werden muss.

[7] Quelle: `https://www.youtube.com/watch?v=WdzWPuazLA8`

2.2.2.3 Klassifizierung

Der zweite Schritt im Prozess besteht in der **Daten Analyse**. Bei der Erstellung von Empfehlungen verfolgt man immer bestimmte Ziele, welche sich durch die aktuellen Aufgaben des Kunden oder des Unternehmens definieren. Je nachdem welche Arten von Merkmalen der Datensatz besitzt kann man unterschiedliche Verfahren auswählen, um ein Modell aufzustellen.

Machine Learning bietet mehrere Herangehensweisen dafür an, um einen Datensatz besser zu verstehen (Murphy, 2012). Es gibt z.B. *Unüberwachtes und Überwachtes Lernen*. Beim ersteren muss der Algorithmus die Eingabedaten beschreiben und charakterisieren, um dann eine Vorhersage für unbekannte Eingaben zu machen. Für *Überwachtes Lernen* wiederum wird eine Funktion angelernt, die die Beziehungen zwischen Eingabe- und Ausgabedaten schätzt. Der verwendete Datensatz in dieser Arbeit besitzt sowohl Eingaben als Nutzerverhalten wie auch eine Ausgabe in Form der am Ende gekauften Artikel.

Beispiel 2 *Schaut man sich einen Kunden an, der sich kurz vor der Winterzeit längere Zeit im Bekleidungssortiment aufhält, so könnte man diesen in die Gruppe für Winterbekleidung einordnen. Damit wäre das korrespondierende Ziel des Empfehlungssystems passende Kleidungsstücke für den Winter vorzuschlagen.*

Eine mögliche Methode um das genannte Beispiel umzusetzen besteht in der Anwendung einer Klassifizierung welche einen Merkmalsraum[8] auf bestimmte Klassen abbildet (Amatriain u. a., 2011). In Abbildung 2.3 sieht man eine beispielhafte Visualisierung für diesen Fall. Einige der erfolgreichen Verfahren zur Klassifizierung sind z.B. *K-Nearest-Neighborhood* (KNN) Koren (2008), *Bayes Analyse* (Rendle u. a., 2009) und *Support Vector Machine* (SVM) (Xia u. a., 2006). Durch erste Experimente sind für *KNN* und *SVM* jedoch Probleme in der Skalierbarkeit aufgefallen. Die Größe des Datensatzes bei *KNN* verhindert das Laden der benötigten Daten in den Arbeitsspeicher. Für *SVM* wurde die Berechnung ohne ein Ergebnis nach mehreren Stunden abgebrochen.

[8] Ein Merkmal kennt man auch unter dem Begriff eines Features.

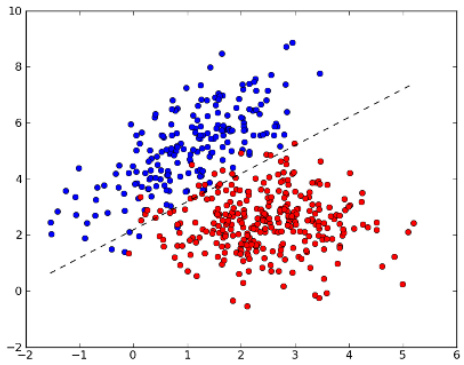

Abbildung 2.3: Binäre Klassifikation.

Ein weiteres Beispiel für eine Klassifizierung sieht man in der Einordnung eines Nutzers anhand seines Verhaltens auf der Seite in die Gruppe der kaufenden Kunden. Dabei kann es von Interesse sein diesem Kunden weitere passende Artikel zu seiner aktuellen Auswahl anzubieten um den Umsatz zu steigern. Im Fall eines Kunden der am Ende nichts kauft würden sich Vorschläge die zu einem Kauf anregen anbieten. Der letzte Punkt zielt wieder auf die Balance zwischen *Explore vs. Exploite*.

Ein weiteres potentielles Merkmal was eine Kaufbereitschaft suggeriert sieht man in der Dauer einer Browser-Session in Verbindung mit einer hohen Anzahl an angeschauten Artikeln (Romov u. Sokolov, 2015). Je nachdem wie das zu untersuchende Problem ausschaut kann man zwischen zwei Typen von Klassifizierungen wählen. Der erste Typ ist die binäre Klassifizierung, welche in den erläuterten Beispielen gezeigt wurde und bei der eine Zuordnung zwischen zwei Klassen stattfindet. Alle weiteren Aufgabestellungen die mehr als zwei Klassen enthalten werden unter dem Typ der *Multiclass* Klassifizierung zusammengefasst. Für den in dieser Arbeit verwendeten Datensatz reicht die binäre Klassifizierung für das Finden eines Kunden der am Ende einen Kauf tätigt aus. Abschließend muss für einen Artikel entschieden werden, ob man diesen vorschlägt oder nicht.

Eine weiterführende Erläuterung wie eine Klassifizierung bei Empfehlungen angewandt

werden kann sieht man im Unterkapitel der Memory-Based Verfahren für *Collaborative Filtering* sowie in Abschnitt 2.6.

2.2.2.4 Regression

> *Die Regressionsanalyse schätzt die funktionalen Abhängigkeiten zwischen Merkmalen, um Zusammenhänge zu verstehen und gezielt zu steuern. Runkler (2015)*

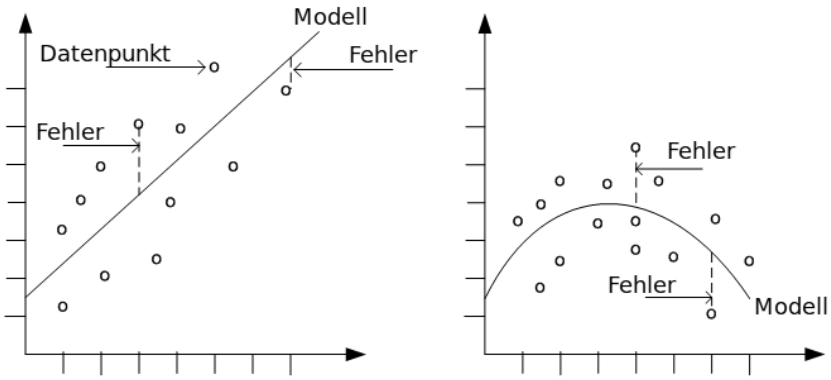

Abbildung 2.4: Bsp. lineare (*links*) und nicht-lineare (*rechts*) Regression

Die Regressionsanalyse ähnelt einer Klassifizierung, außer in dem Punkt das bei der Regression eine kontinuierliche Variable vorhergesagt werden muss Murphy (2012). Auf den E-Commerce angewand bedeutet das:

Beispiel 3 *Eine Vorhersage für die Bewertung eines für den Kunden unbekannten Artikel anhand des aufgezeichneten Kundenverhaltens oder dem Kontext (Aufenthaltsdauer, Alter, Wochentag, etc.) auf der Webseite zu machen.*

Damit man aufgezeichnete Daten mit kategorialen Merkmalen für die Analyse nutzen kann, müssen diese für die meisten Verfahren transformiert werden[9]. Dabei kann es passieren, dass eine Explosion des Merkmalsraum stattfindet und der Aufwand in der Verarbeitung zu hoch wird (Romov u. Sokolov, 2015).

[9] Quelle: `http://scikit-learn.org/stable/modules/preprocessing.html#encoding-categorical-features` (2016.11.19)

In Abbildung 2.4 sieht man zwei Beispiele für jeweils ein Modell, was an lineare Daten angepasst wurde und eins was nicht-lineare Daten beschreibt. Da beim implizitem Feedback nicht immer nur kontinuierliche Variablen vorkommen muss individuell entschieden werden ob sich eine Regressionsanalyse dafür eignet. In James u. a. (2013) wird dafür ein ausführlicher Vergleich zwischen Klassifizierung und Regression gemacht.

2.2.3 Data Science

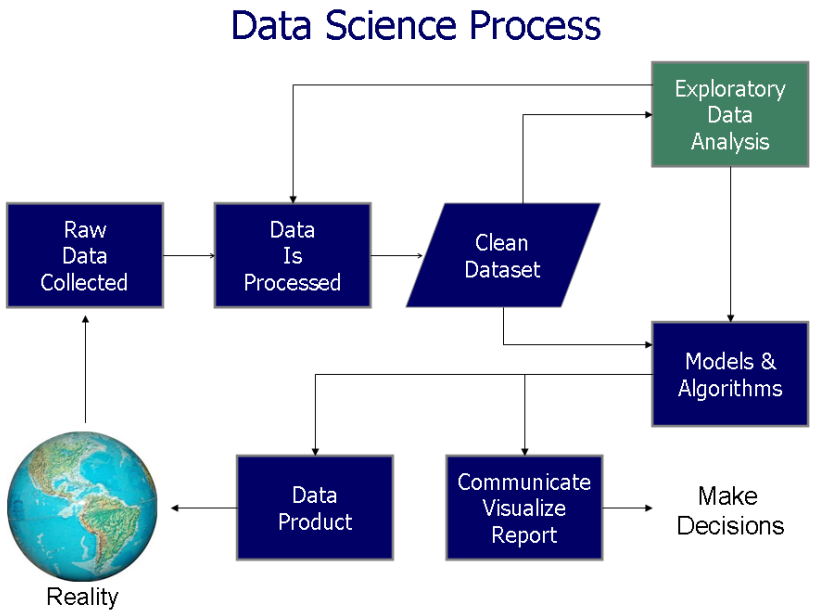

Abbildung 2.5: *Data Science* Prozess

Quelle: `https://en.wikipedia.org/wiki/Data_science` (2016.11.05)

In den letzten Jahren wurden viele bekannte Disziplinen unter den heutigen Anforderungen der Flut an Daten Herr zu werden in ein neues Licht gerückt. Ein weit verbreiteter Begriff der in diesem Zusammenhang genannt wird ist *Data Science*[10]. Dieser kombiniert viele bekannte Techniken und lässt sich am ehesten mit den Prinzipien aus dem *KDD* Prozess vergleichen. Jedoch ist der Schwerpunkt mehr auf den Daten und dem praktischen Umgang mit diesen in Verbindung mit den vielseitigen *Open Source*

[10] `https://en.wikibooks.org/wiki/Data_Science:_An_Introduction/A_History_of_Data_Science` (2016.11.05)

Werkzeugen, die man zur Modellierung und Visualisierung[11] einsetzen kann. Für die Erfüllung der Aufgaben kommen universelle oder auch domänenspezifische Programmiersprachen, wie z.B. *R*[12], Julia[13] oder Python[14] zum Einsatz. In dieser Arbeit wurde Python ausgewählt, da dafür das grundlegende Wissen schon vorhanden war und viele der benötigten Verfahren für die wissenschaftliche Analyse als freie Bibliotheken angeboten werden (McKinney, 2012).

Quelle:`http://jupyter.org` (2016.11.05)

Jupyter In Abbildung 2.5 sieht man, dass eine der zentralen Komponenten vom *Data Science Prozess* die explorative Daten-Analyse ist. Zur Umsetzung dieser braucht man ein Werkzeug, welches die Möglichkeit zur Ausführung von Operation auf den Daten sowie die passende Darstellung der Ergebnisse bietet. Dadurch soll es ermöglicht werden zu einer gemachten Hypothese schnell eine Rückmeldung zu erhalten. Darüber hinaus ist die Dokumentation des ganzen Prozesses entscheidend, damit andere leichter auf der erbrachten Arbeit aufsetzen können. Die eingesetzten Verfahren und Ergebnisse lassen sich zudem so einfacher durch Dritte überprüfen. Das hierfür gewählte Werkzeug ist das *Jupyter Notebook*[15] was die erläuterten Voraussetzungen mitbringt. Weitere Alternativen zu *Jupyter* sind *Apache Zeppelin*[16] oder *Beaker Notebook*[17].

[11] `http://datasciencemasters.org/` (2016.12.13)
[12] `https://www.r-project.org/` (2016.12.13)
[13] `http://julialang.org/` (2016.12.13)
[14] `https://www.python.org/` (2016.12.13)
[15] `https://jupyter.org/` (2016.11.29)
[16] `https://zeppelin.apache.org/` (2016.11.30)
[17] `http://1beakernotebook.com/` (2016.11.30)

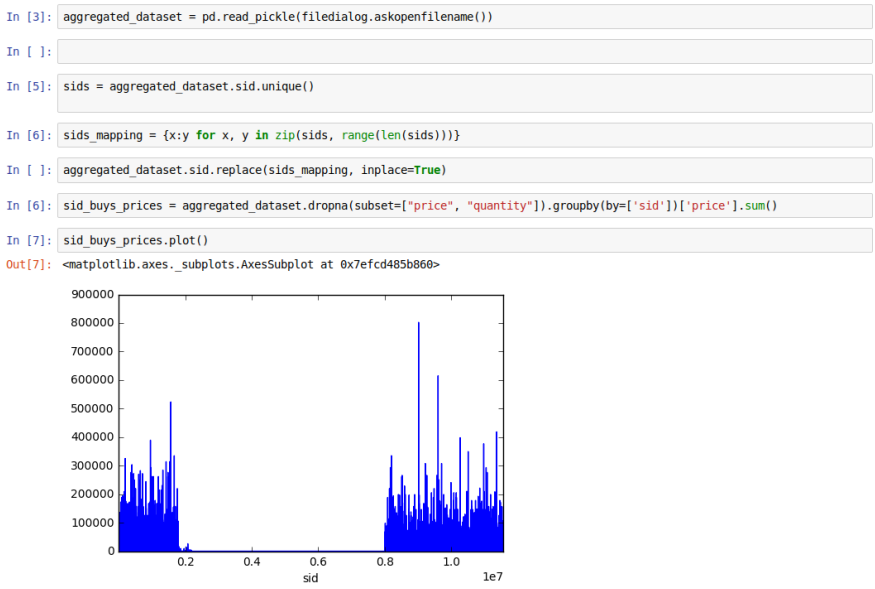

Abbildung 2.6: Jupyter Notebook Darstellung im Browser.

Jupyter erlaubt die Erstellung einer Dokumentation in Form von *Notebooks* zu einer Untersuchung mit ausführbaren Codefragmenten und Visualisierungen (siehe Abbildung 2.6). Für die Modellierung kann aus einer Vielzahl an Programmiersprachen gewählt werden, da die Oberfläche von Jupyter als Webanwendung umgesetzt wurde. Die eigentliche Berechnung findet auf der Serverseite statt und wird dann z.B. an den Python-Interpreter weiter delegiert.

Pandas & Numpy

Abbildung 2.7: Logo von Numpy und Pandas

In der Python-Community[18] gibt es viele etablierte Bibliotheken für die Umsetzung von wissenschaftlichen Anforderungen. Die Eckpfeiler auf denen diese jeweils Aufbau-

[18] https://www.scipy.org/

en sind *Numpy*[19] (Numerical Python) und Pandas[20] (McKinney, 2012). *Numpy* liefert einen effizienten Container für numerische Daten. Dieser ist ein multidimensionales Array, welches unterschiedliche mathematische Operationen zwischen anderen Arrays unterstützt. Darauf setzt wiederum *Pandas* auf um weiterführende Datenstrukturen wie z.B. den *Dataframe* damit umzusetzen. Dieser bietet Methoden um die Daten darin zu transformieren, zu filtern und statistische Auswertungen darauf auszuführen an. *Dataframes* kann man sich als zweidimensionale Tabellen vorstellen wie man in Abbildung 2.8 sehen kann. Auf diesen kann man z.B. speicher-effiziente Projektionen der Daten ausführen, um so neue Merkmale des Datensatzes zu finden.

```
In [5]: clicks_dataset
```

Out[5]:

	sid	timestamp	aid	cid
0	1	2014-04-07 10:51:09.277	214536502	0.0
1	1	2014-04-07 10:54:09.868	214536500	0.0
2	1	2014-04-07 10:54:46.998	214536506	0.0
3	1	2014-04-07 10:57:00.306	214577561	0.0
4	2	2014-04-07 13:56:37.614	214662742	0.0
5	2	2014-04-07 13:57:19.373	214662742	0.0
6	2	2014-04-07 13:58:37.446	214825110	0.0

Abbildung 2.8: Beispiel Dataframe

Scikit-Learn

Abbildung 2.9: Logo von Scikit-Learn

Abgerundet wird das Paket mit dem *Scikit-Learn*[21] Projekt, was auch als ”*wissenschaftlicher Werkzeugkasten*” bezeichnet wird. Dieser beinhaltet alle wichtigen Algo-

[19] http://www.numpy.org/ (2016.12.13)
[20] http://pandas.pydata.org/ (2016.12.13)
[21] http://scikit-learn.org/ (2016.12.13)

rithmen zur Implementierung des *Data-Mining* Prozesses. Der Vergleich zwischen den besprochenen Verfahren findet mit Hilfe des *Scikit-Learn* Projekts statt. Denn das Paket bietet viele Methoden zur Evaluierung und Erstellung eines Modells, die man deswegen nicht eigenständig implementieren muss.

2.3 RecSys Challenge

RecSys

Die alljährliche *ACM* Konferenz zum Thema Empfehlungssystem[22] fördert den Austausch neuer Erkenntnisse sowie die Lösung aktueller Probleme bei der Erstellung von Empfehlungen aller Art. Eines der Höhepunkte für Interessierte besteht in der *RecSys Challenge*[23], welche ein halbes Jahr vorher beginnt und zur Konferenzzeit mit einem Workshop und der Verkündung der Sieger endet. Das Ziel jedes Wettbewerbes liegt in der Lösung eines speziellen Problems im genannten Themengebieten. Eine vergleichbare Plattform die das Bearbeiten solcher Problemstellungen fördert, findet man in Kaggle[24]. Die Internetseite von Kaggle ist ein Ort wo gezielt von Unternehmen und Nutzern eingestellte Datensätze unter einer bestimmten Fragestellung untersucht werden. Eines der hier untersuchten Verfahren liefert in diesen Wettbewerben häufig erstaunliche Ergebnisse, wo andere Verfahren versagen[25]. Ein großer Vorteil dieser Herangehensweise an eine Problemstellung besteht im regen Austausch von Wissen und der Überprüfung der gemachten Ergebnisse durch eine Community mit viel Erfahrung. Die gesammelten Erkenntnisse dabei sind für alle kostenlos einsehbar und nachstellbar.

Yoochoose

Der *RecSys* Wettbewerb wird jedes Jahr von einer anderen Organisation ausgeführt. So übernahm 2015 die Firma *YOOCHOOSE*[26] die Austragung mit einer eigens dafür formulierten Herausforderung. *YOOCHOOSE* stellt im kommerziellen Bereich Lösung zur

[22] https://recsys.acm.org/ (2016.10.15)
[23] http://recsyschallenge.com (2016.10.15)
[24] https://www.kaggle.com/ (2016.12.05)
[25] http://www.kdnuggets.com/2016/03/xgboost-implementing-winningest-kaggle-algorithm-spark-flink.html (2016.10.01)
[26] http://www.yoochoose.com (2016.08.10)

Berechnung von qualitativ hochwertigen Empfehlungen für kleine und mittelständische Unternehmen bereit (Ben-Shimon u. a., 2015).

Der veröffentlichte Datensatz der Organisation enthält jeweils eine Aufzeichnung von Kunden gemachter Klicks auf einer E-Commerce Plattform. Ein Teil dieser Aufzeichnungen besteht aus den getätigten Käufen von Kunden. Eine explizite Analyse des vollständigen Datensatzes findet im Kapitel zur Exploration der Daten statt.

Bewertung der Teilnehmerlösungen

Die Sieger des Wettbewerbs wurden durch die Berechnung der folgenden Punktzahl ermittelt, die man in der Gleichung 2.1 sehen kann.

$$Score(Sl) = \sum_{\forall s \in Sl} \begin{cases} if\ s \in S_b & \rightarrow \frac{|S_b|}{|s|} + \frac{|A_s \cap B_s|}{|A_s \cup B_s|} \\ else & \rightarrow -\frac{|S_b|}{|S|} \end{cases} \tag{2.1}$$

Die einzelnen Komponenten haben die folgende Bedeutung:

Sl sind die Sessions die man in seiner Lösung eingereicht hat.

S sind alle Sessions aus dem Testset.

s ist eine Session aus der eingereichten Lösung.

S_b sind die Sessions in denen wirklich was gekauft wurde.

A_s sind die vorhergesagten Artikel für die Session.

B_s sind die wirklich gekauften Artikel in der Session.

$\frac{|A_s \cap B_s|}{|A_s \cup B_s|}$ ist der Jaccard-Koeffizient[27] zwischen der Vorhersage und der Wirklichkeit.

Wie man anhand der einzelnen Komponenten sehen kann werden jeweils zwei Aufgaben berücksichtigt und bewertet. Die erste besteht darin kaufende Sessions zu finden

[27] Der Jaccard-Koeffizient ist eine Vergleichsmetrik, die die Ähnlichkeit zweier Mengen darstellt. Referenz: `https://de.wikipedia.org/wiki/Jaccard-Koeffizient` (2016.08.20)

und die zweite ist die Vorhersage der gekauften Artikel. Für falsche Sessions gibt es darüber hinaus eine negative Bewertung. In der Endwertung hat das Modell des Siegerteams die Hälfte aller getätigten Käufe vorhergesagt[28]. Da für die hier gemachte Arbeit nicht der Wettbewerb in Vordergrund steht, sondern das Verstehen und der Vergleich der eingesetzten Verfahren, wird diese Gleichung nicht verwendet.

2.4 Implizites Feedback

Ein wichtiger Aspekt bei der Erstellung von Empfehlungen ist die Art der vorliegenden Daten, auf denen man seine Analyse aufbaut. In diesem Fall liegen die Daten als Aufzeichnungen von Nutzeraktionen vor (Ben-Shimon u. a., 2015). Die jeweiligen Aktionen der Kunden wurden beim Besuch auf einer unbekannten E-Commerce Plattform aufgezeichnet. Diese Art der Daten wird in den nachfolgenden Abschnitten genauer beschrieben und diskutiert.

2.4.1 Mangel an Daten

Der größte Teil der bestehenden Forschung basiert auf einigen wenigen veröffentlichten Datensätzen, die meistens nur explizites Feedback in Form von Bewertungen enthalten. Zu den bekanntesten zählen z.B. der *MovieLens* (Harper u. Konstan, 2015) und der *Netflix* Datensatz (Netflix, 2009). Dem Datenmangel und den Vorteilen vom explizitem Feedback geschuldet wurden bisher recht einseitige Untersuchungen von Algorithmen zur Erstellung von Empfehlungen geführt. Aushilfsweise wurden vereinzelt explizite Bewertungen durch eigens definierte Zuordnungen[29] in implizite umgewandelt (Rendle u. a., 2009). Daraus erkennt man, dass der vorliegende Datensatz potentiell neue und wertvolle Erkenntnisse in sich birgt. Es existieren auch Ausnahmen wie dem *Million Song*[30] Datensatz der Metadaten und die Hörgewohnheiten zu vielen Musiklieder enthält.

[28] Pressemitteilung: `https://yoochoose.com/de/Press-Release-Recsys-2015-Challenge` (2016.11.23)
[29] Beispiel: Eine Bewertung von vier bis fünf könnte als ein Besuch der Artikelseite interpretiert werden.
[30] `http://labrosa.ee.columbia.edu/millionsong/` (2016.12.13)

2.4.2 Formen von Feedback

Beide Arten von Feedback haben ihre Vor- und Nachteile. So ist explizites Feedback in Form von Bewertungen eine präzise Methode um eine Präferenz zu einer Sache auszudrücken. Jedoch müssen die Nutzer diese Angaben aktiv machen oder vom System dies bezüglich befragt werden. Die Möglichkeit solche Informationen zu bekommen besteht nicht immer.

Im Gegensatz dazu ist implizites Feedback eine Datenquelle die in vielen unterschiedlichen Formen vorkommen kann (Hu u. a., 2008) ohne eine direkte Kundeninteraktion. Im E-Commerce könnten es die Kaufgewohnheiten oder das Suchmuster der Kunden sein. Auch Mausbewegungen und das Verharren auf bestimmten Abschnitten einer Webseite zählen zum Nutzerverhalten, welches man zum Entdecken von unbekannten Mustern analysieren kann. Die Menge dieser Daten ist um einiges größer und einfacher aufzuzeichnen da z.B. schon alle Anfragen an einen Webserver von vornherein mitgeschnitten werden.

Ein entscheidender Faktor der bei beiden Formen über einen längeren Zeitraum an Bedeutung gewinnt ist die Veränderung von Vorlieben (Siddiqui u. a., 2014). Geschmäcker ändern sich über die Zeit, was von den aktuellen Ansätzen nicht ausreichend berücksichtigt wird. Die Berücksichtigung der zeitlichen Veränderung von Vorlieben ist nicht Gegenstand dieser Arbeit, da nur ein statischer Datensatz verwendet wird.

2.4.3 Datendichte

Auch wenn beide Arten diverse Unterschiede in bestimmten Gebieten aufweisen, gibt es doch Gemeinsamkeiten bei der Tauglichkeit in der Analyse. In Abbildung 2.10 sieht man einen Graphen der Nutzerbewertungen zu bestimmten Filmen aufzeigt. Die Datendichte ist ein wichtiges Kriterium, welches bei der Analyse entscheidend sein kann. In diesem Beispiel haben die älteren Filme in dem *MovieLens* Datensatz mehr Bewertungen als die neueren. Zudem existieren teilweise keine neueren Bewertungen von älteren Nutzern. Ist die Dichte der Daten zu gering kann man schlechter Überschneidungen bzw. Gemeinsamkeiten zwischen den Nutzern oder Artikeln finden. Deswegen wird meis-

tens eine Schätzung über den ganzen Datensatz gemacht, um eine Grundlage für weitere Berechnungen zu liefern (Rendle u. a., 2009).

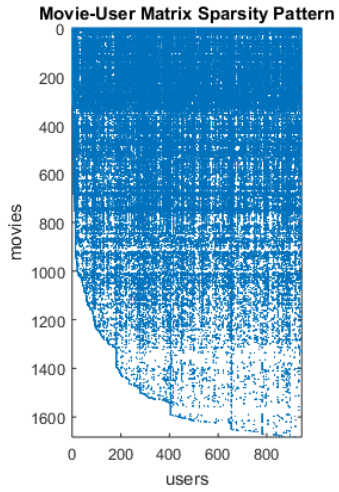

Abbildung 2.10: MovieLens100k: Nutzerbewertungen von Filmen

Quelle:`http://blogs.mathworks.com/loren/2015/04/22/the-netflix-prize-and-production-machine-learning-systems-an-insider-look/` (2016.10.19)

2.4.4 Eigenschaften von explizitem und implizitem Feedback

Die weiteren Besonderheiten vom implizitem Feedback im E-Commerce werden in den nachfolgenden Abschnitten mit Hilfe von Hu u. a. (2008) genauer diskutiert.

Grad des Vertrauens in die Sichtungen

Am Beispiel des Nutzerverhaltens sieht man dass bei einem zeitlich langen Besuch auf einer Artikelseite entweder ein hohes Interesse an dem aktuellen Artikel besteht oder der Nutzer z.Z der Sichtung nicht vor seinem Computer war. Bei dem umgekehrten Fall, wo eine Artikelseite nicht besucht wurde, verhält es sich ähnlich. Der Kunde hat vielleicht diesen Artikel wegen der großen Auswahl noch nicht entdeckt oder kein Interesse an diesem gezeigt. Daraus kann man folgern, dass ein Faktor für die Vorliebe zu einer Sache gebraucht wird. Dieser muss das prozentuale Vertrauen in die beobachteten Daten widerspiegeln.

Qualität der Daten

Wie man anhand des vorherigen Beispiels sehen kann ist die Qualität von implizitem Feedback je nach Datenform durch das Vorhandensein von Rauschen oder Fehlern in den aufgezeichneten Daten gemindert. Der Grund dafür ist der Mangel einer direkten Überprüfung der aktuellen Beobachtung mit Hilfe des Kunden. Alternative kann man auch genügend Datensammeln, um anhand des Mittelwerts die Ausreißer als solche zu identifizieren.

Objektivität der Informationen Bei expliziten Bewertungen wiederum kann es jedoch passieren, dass ein Kunde zwei Filme unterschiedlich bewertet und trotzdem beide gleich gut findet. Eine abgegebene Bewertung ist ein subjektiver Wert den jeder Mensch unterschiedlich für sich selbst definiert. Damit würde das Nutzerverhalten eine objektivere Form darstellen wenn man die negativen Eigenschaften überwinden kann.

Aspekt der Security Eine weitere Herausforderung entsteht wenn das System auf geschäftsschädigende Weise ausgetrickst wird. Ein Empfehlungssystem kann nicht von vornherein ein nicht echtes Kundenprofil entdecken. Dadurch können solche Arten von Angriffen durch vorgetäuschte Nutzerprofile falsche Trends entstehen lassen (Mobasher u. a., 2007). Der Aspekt eines robusten Systems gegen solche Angriffe wird nicht in dieser Arbeit behandelt jedoch das Säubern der Daten vom Rauschen.

Evaluierung

Im Gegensatz zu direkten Bewertungen kann man beim impliziten Feedback erstellte Empfehlungen schlecht evaluieren. Jedoch sind vorgeschlagene Artikel die am Ende auch gekauft wurden ein gutes Indiz für eine zufriedenstellende Empfehlung. Trotzdem kann man nicht mit Sicherheit sagen zu welchem Grad der Artikel am Ende auch zufriedenstellend war. Zum Beispiel kann ein Artikel auch als Geschenk für jemanden gekauft worden sein. Zur Vereinfachung dieses Problems werden die Daten in unterschiedliche Datensätze aufgeteilt. Einen für das Anlernen des Modells und einen weiteren zur Überprüfung anhand der getätigten Käufe. Der endgültige Kauf ist damit das ausschlaggebende Kriterium wie gut die Genauigkeit einer Vorhersage ist. Dafür schaut man sich

die tatsächlich gekauften Artikel und den Anteil der vorher vorgeschlagenen Artikeln darin an (Davis u. Goadrich, 2006).

2.4.5 Fazit

Implizites Feedback lässt sich einfach in vielen Formen wiederfinden. Im Gegensatz dazu ist es beim expliziten Feedback nicht der Fall. Bei beiden Arten muss eine Interaktion mit dem System stattfinden, jedoch ist die Hürde beim impliziten Feedback im Internet um einiges kleiner. Darüber hinaus kann es bei beiden Arten passieren, dass die Datendichte von Nutzern zu Artikeln zu klein ist und dadurch nur schwer Gemeinsamkeiten zwischen den Nutzern oder Artikeln gefunden werden können.

Schaut man sich den aktuellen Stand der Forschung an so verwenden die meisten Studien explizite Bewertungen, da viele veröffentlichte Datensätze nur diese enthalten und die Interpretation dieser einfacher ist. Solche Gründe führen zu einer recht einseitigen Veröffentlichung von wissenschaftlichen Arbeiten. Es ist also wahrscheinlicher, dass man im impliziten Feedback neues Wissen für Empfehlungssysteme entdecken kann.

2.5 Collaborative Filtering

Empfehlungssysteme lassen sich mit unterschiedlichen Verfahren realisieren. Zur Zeit stammen jedoch die am häufigsten eingesetzten Verfahren aus dem Bereich des *Collaborative Filtering* (CF) (Ekstrand u. a., 2011), die schon mit recht wenigen Informationen eine gute Vorhersage machen können. Das Kernkonzept besteht darin anhand von Bewertungen oder Verhalten der Nutzer mit dem System eine Vorhersage zu treffen, was dem Nutzer aus dem Sortiment gefallen könnte. Um den Gesichtspunkt der Personalisierung einzubringen bedeutet dies ähnliche Benutzer- oder Artikelgruppen zu finden. Andere Arbeiten die keinen personalisierten Ansatz verfolgen liefern meistens schlechtere Ergebnisse (Pilászy u. Tikk, 2009). Da der zu untersuchende Datensatz sich gut für *CF* Methoden eignet, werden die alternativen Verfahren aus der Beschreibung herausgelassen. Denn durch die anfängliche Exploration der Daten in Abschnitt 3.3 wurden erste Mängel in den vorhandenen Metadaten aufgezeigt, die für diese Verfahren nachteilig sind.

Die nun folgenden Abschnitte beschreiben die Herangehensweise aus Sicht eines Empfehlungssystems, welches zum theoretischen Vergleich für die *Data-Mining* Umsetzung genommen wird.

2.5.1 Problemdefinition

Ein Problem für CF lässt sich in der folgenden Form definieren. Es gibt eine Menge von Nutzern U und eine Menge an Artikeln I. Die einzelnen Bewertungen eines Nutzers werden dann in einer $U - I$ Matrix dargestellt (siehe Tabelle 2.1), wobei R_ij die Vorliebe eines Nutzers i zu einem Artikel j kennzeichnet (Shi u. a., 2014). Da man kein vollständiges Wissen zu allen Beziehungen besitzt ist es eine schwach besetzte Matrix. Mit den gegebenen Informationen kann jedoch eine Schätzung für die möglichen Bewertungen von unbekannten Artikel anhand ähnlicher Nutzer für den aktuellen Kunden berechnet werden. Daraus filtert man dann Empfehlungen mit absteigender Relevanz (Ranking) für den Nutzer, die z.B. noch nicht gekauft wurden oder anhand der Historie von diesem oder anderer Nutzer die größte Wahrscheinlichkeit zu einem Kauf besitzen.

	$Terminator$	$Matrix$	$\cdots$	$Artikel_n$
$Jens$	2	3	$\ldots$	r_1
$Lisa$	?	4	$\ldots$	r_2
$Hans$	5		$\ldots$	r_2
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$
$User_n$	r_n	r_n	$\ldots$	r_n

Tabelle 2.1: Schwachbesetzte Matrix: Nutzer Bewertungen für Artikel

Beim impliziten Feedback würden nicht aufgezeichnete Beobachtungen als 0 in der Matrix gekennzeichnet, weil dafür erst einmal keine Aussage getroffen werden kann. Alle übrigen Einträge wären dann mit einer 1 gekennzeichnet, weil die entsprechenden Artikel angeschaut oder gekauft wurden.

2.5.2 Memory-Based

In Shi u. a. (2014) wird eine umfassende Übersicht über CF und darüber hinaus aufgezeigt. Die Autoren fassen die einzelnen Verfahren in zwei Kategorien zusammen. Die erste Kategorie umfasst die Methoden von *Memory-Based* Verfahren, welche sich auf die Gemeinsamkeiten in den abgegebenen Bewertungen von Nutzern zu den Artikeln konzentrieren. Diese Kategorie beinhaltet zwei grundlegende Techniken, die jeweils entweder die Nutzer oder die Artikel mit deren Bewertungen untersuchen.

2.5.2.1 User-Based

$$\hat{R}_{ij} = \frac{1}{C} \sum_{k \in Z_i} similarity(i,k) R_{kj} \tag{2.2}$$

Eines der ersten populären Verfahren nutzt die bestehenden Bewertungen von Nutzern aus um neue Vorschläge zu generieren (siehe Gleichung 2.2). Zu diesem Zweck werden die vorhandenen Bewertungen aggregiert und mit Hilfe einer Vergleichsmetrik[31] ($similarity(i,k)$) in Relation zueinander gestellt (Amatriain u. a., 2011). Die Nutzer mit ähnlichen Werten ($k \in Z_i$) werde dann ausgewählt und bilden mit deren Bewertungen die Grundlage für eine Vorhersage ($\hat{R}_{ij}$) über einen Artikel (j) den der aktuelle Kunde (i) noch nicht bewertet hat. Dadurch kann man für den Kunden unbekannte Artikel anhand seiner Bewertungs-Historie und den Bewertungen anderer ähnlicher Nutzer in der Gruppe empfehlen (R_{kj}). Diese Methode zur Berechnung von Empfehlungen anhand von Ähnlichkeiten fällt in den Bereich der *Nearest-Neighborhood* (NN) Verfahren.

Vergleichsmetrik Ein häufig eingesetztes Verfahren zur Berechnung der Gemeinsamkeiten ist der Korrelationskoeffizient. Dieser beschreibt den Zusammenhang zwischen zwei Merkmalen zueinander (Amatriain u. a., 2011). Das bedeutet, dass es keine kausale Verbindung zwischen diesen existieren muss.

[31] z.B. Korrelationskoeffizient oder Kosinus-Ähnlichkeit

$$Pearson(x, y) = \frac{\sum(x, y)}{\sigma_x \times \sigma_x} \tag{2.3}$$

In Gleichung 2.3 sieht man die Berechnung für zwei Datenpunkte x und y sowie die Kovarianz $\sum(x, y)$. Das σ ist die jeweilige Standardabweichung der einzelnen Datenpunkte.

2.5.2.2 Item-Based

Im zweiten Fall schaut man sich die vom Nutzer bewerteten Artikel aus dessen Historie an. Dazu werden die Ähnlichkeiten zwischen diesen berechnet, um neue unbekannte Artikel vorzuschlagen. Diese Variante ist eine Optimierung zur vorherigen, da man nicht alle Paare zwischen den Nutzern berechnen muss. Der Kern dieser Variante besteht darin, dass man die gekauften Artikel von einem Nutzer auf ihre Gemeinsamkeiten untersucht, da Kunden dazu neigen Artikel mit ähnlichen Merkmalen zu kaufen (Deshpande u. Karypis, 2004).

2.5.2.3 Bewertung

Wie man an diesen beiden Ansätzen sieht, sind für die Ermittlung von Bewertungen jeweils alle Paare von Nutzern oder Artikeln zu berechnen. Zudem ist die Genauigkeit eines Modells von einer guten Funktion zur Bestimmung der Ähnlichkeit ($similarity$) abhängig. Eine weitere Variable die man bestimmen muss ist die richtige Größe der Menge von ähnlichen Nutzern (Ekstrand u. a., 2011). Diese Größe ist domänenspezifisch und muss je nach Szenario bestimmt werden was bei dem benutzten Datensatz mangels der Experten nicht direkt möglich ist. Außer man setzt ein weiteres Lernverfahren dafür ein, um den idealen Wert zu finden. Beide Verfahren können für implizites Feedback eingesetzt werden, jedoch könnten beide bei einer zu hohen Datengröße scheitern. Darüber hinaus muss eine Zuordnung von den aufgezeichneten Nutzerverhalten in mathematische Größen gemacht werden damit eine Vergleichsmetrik berechnet werden kann. Trotz der genannten Probleme werden diese Verfahren in Kombination mit anderen, um besserer Ergebnisse zu liefern eingesetzt (Koren, 2008).

2.5.3 Model-Based

In diesem Abschnitt wird der Prozess der Erstellung von Empfehlung näher vorgestellt. Dafür wird sich das *Matrix Factorization* Verfahren genauer angeschaut und erläutert. Die Beschreibung dieses Verfahrens soll repräsentativ für ein domänenspezifisches Verfahren stehen und soll die allgemein relevanten Schritte mit Beispielen aufzeigen. Die man wiederum im Kapitel 4 wiederverwenden oder in einem Vergleich gegenüber stellen kann.

2.5.3.1 Empfehlungen mit Matrix Factorization

Die zweite Kategorie aus dem Bereich des CF besteht in der Erstellung eines Vorhersagemodells, welches mit Hilfe der $U - I$ Matrix antrainiert wird. Die häufigste und vielversprechendste Methode z.Z. ist die *Matrix Factorization* (MF) (Koren u. a., 2009). Die grundlegende Idee hinter MF ist, wie der Name schon sagt, in der Zerlegung einer Matrix in Faktoren. Das Prinzip hinter MF sagt aus, dass man die Beziehung zwischen einem Nutzer und einer Sache durch Eigenschaften oder Vorlieben die beide besitzen beschreiben kann. Eine andere Definition nennt es auch die *Zerlegung der Beziehungen zwischen zwei kategorialen Variablen*, die hier U und I sind (Rendle, 2010).

Beispiel 4 *Ein Kunde kauft gerne teure Schuhe einer bestimmten Marke. Die Schuhe besitzen die Eigenschaften, dass sie unter einer Marke verkauft werden sowie zu einem teureren Segment gehören. Das sind die jeweiligen Vorlieben die der beobachtete Kunde besitzt.*

Geht man weiter und bewertet die vorliegenden Eigenschaften nach ihrer Bedeutung kann man durch den Vergleich zu anderen Nutzern dessen Gemeinsamkeiten finden.

Beispiel 5 *Für einen Kunden der ausschließlich Markenschuhe kauft sind No Name Artikel von kleiner Bedeutung. Andere Kunden mit dieser Ausprägung könnten für diesen Kunden interessante Artikel gekauft haben.*

Die Darstellung dieser Vorlieben kann durch eigenständige Matrizen repräsentiert werden. Die Umwandlung dieser separaten Matrizen in die ursprüngliche Matrix wird durch das Skalarprodukt realisiert. Die Form und Größe der Teilmatrizen kann dabei variieren, da diese die verborgenen Beziehungen zwischen den Nutzern und Artikeln darstellen.

Die Anzahl dieser verborgenen Beziehungen ist dabei jedoch kleiner als die ursprüngliche Matrix mit allen Kunden und Artikeln. Es findet also gleichzeitig auch eine Reduktion des Problemraums statt.

Matrixzerlegung Wie schon im Abschnitt zu den Memory-Based Ansätzen erläutert scheitern *Nearest-Neighborhood* Verfahren an dem Problem der Berechnung bei steigender Anzahl an Daten (Sarwar u. a., 2000). Um diesem Problem Herr zu werden kann man versuchen die Dimension der Daten zu verkleinern. Eine beliebte Methode hierfür ist die *Singular Value Decomposition* (SVD) welche aus dem Gebiet des *Information Retrieval* stammt (Paterek, 2007). *SVD* wird auch im *Data-Mining* Prozess zur Vorverarbeitung der Daten genutzt. In dem Kontext von *MF* stellt es jedoch auch eine Methode dar, um Schätzungen für die Vorlieben zu berechnen.

$$R = U \cdot S \cdot V' \tag{2.4}$$

In Gleichung 2.4 sieht man die Grundform von SVD, dabei is R unsere $U - I$ Matrix, welche man auch Näherungsweise in Form von einzelnen kleineren Matrizen darstellen kann. Bei der einleitenden Problemdefinition von CF wurde eine *Nutzer x Artikel* Matrix gezeigt, die alle im System abgegebenen Bewertungen von Nutzern zu Filmen repräsentiert. Auf einer E-Commerce Plattform mit Millionen von angemeldeten Nutzern und Artikeln würde diese Matrix eine nicht mehr beherrschbare Größe annehmen. In Tabelle 2.2 sieht man das Ergebnis, was durch das Anwenden von *SVD* entsteht.

$$\underset{\text{Nutzer}\times\text{Artikel}}{\underset{R}{\begin{pmatrix} x_{11} & x_{12} & \dots & x_{1n} \\ x_{21} & x_{22} & \dots & \\ \vdots & \vdots & \ddots & \\ x_{m1} & & & x_{mn} \end{pmatrix}}} = \underset{\text{Nutzer}\times\text{ r}}{\underset{U}{\begin{pmatrix} u_{11} & \dots & u_{1r} \\ \vdots & \ddots & \\ u_{m1} & & u_{mr} \end{pmatrix}}} \underset{r \times r}{\underset{S}{\begin{pmatrix} s_{11} & 0 & \dots \\ 0 & \ddots & \\ \vdots & & s_{rr} \end{pmatrix}}} \underset{\text{Nutzer}\times\text{ r}}{\underset{V'}{\begin{pmatrix} v_{11} & \dots & v_{1n} \\ \vdots & \ddots & \\ v_{r1} & & v_{rn} \end{pmatrix}}}$$

Tabelle 2.2: Bewertungsmatrix aufgeteilt in Nutzer- und Artikelkonzepte.

Bewertung von SVD Ein Nachteil den man mit *SVD* beachten muss ist, dass die verwendete Matrix keine Null-Werte beinhalten darf. Das wird im übernächsten Abschnitt

zur Erstellung eines Modells näher erläutert. Ein weiterer Nachteil besteht darin, dass die Berechnungen der einzelnen Gewichtungen für die Stärke einer Vorliebe nicht parallel gemacht werden. Eine Alternative die dieses Problem löst sieht man in *Alternating least squares* (ALS), was trotz einem steigenden r Wert in der Komplexität linear wächst. Während des *Netflix* Wettbewerbs wurden zudem viele weitere Erweiterungen für *SVD* vorgestellt. Dazu zählt zum einen die inkrementelle Berechnung der Schätzungen[32] und zum anderen die Vereinfachung der Berechnung durch das Weglassen der Parametrisierung für die Nutzermatrix U.

Verborgene Beziehungen In Tabelle 2.2 sieht man wie die Beziehungen zwischen Nutzern und Artikeln auf kleiner dimensionierte Matrizen reduziert wurden. Die dafür berechneten Werte stellen die verborgenen Beziehungen zwischen den einzelnen Teilen der ursprünglichen Bewertungsmatrix dar. Die Spaltenanzahl r der U und V' Matrizen legt jeweils fest wie genau die am Ende erstellten Vorhersagen sind. Der beste Wert für r muss je nach Aufgabe dafür gefunden werden. Ein zu großer Wert bedeutet eine bessere Annäherung an die gegebenen Daten, was zu einem sehr speziellen Modell (Overfitting) führt, welches schlechte Vorhersagen für neue Daten liefert. Aus diesem Grund wurden mehrere Methoden zur Regulierung des Modells entwickelt (Hu u. a., 2008).

Diese Art der Darstellung der entscheidenden Merkmale eines Datensatzes hat den Vorteil, dass man nicht gezwungen ist explizit die wichtigen Merkmale und Zusammenhänge zu identifizieren. Das Bedeutet jedoch nicht, dass keine weiteren neuen Merkmale berücksichtigt werden können, sondern das sie in anderer Form mit ins Modell einberechnet werden müssen. Eine gute Übersicht wie das gemacht werden kann sieht man in Koren u. a. (2009).

Anlernen des Modells In Gleichung 2.5 sieht man eine Funktion f die als Eingabeparameter die Informationen über den Nutzer (p_i) und den Artikel (q_j) bekommt. Daraus wird mit dem Skalarprodukt die finale Bewertung für die Beziehung zwischen diesen berechnet. So wie in den vorherigen Verfahren ist hier wiederum das Skalarprodukt die Funktion zur Bestimmung der Ähnlichkeit.

[32] `http://sifter.org/~simon/journal/20061211.html` (2016.11.13)

$$f(p_i, q_j) \rightarrow R_{ij} \tag{2.5}$$

Über die bekannten Bewertungen kann man die unbekannten bestimmen, dazu muss man die bestehende Gleichung 2.5 nur umstellen. Eine mögliche Methode besteht in der Optimierung der Funktion aus Gleichung 2.6. Dabei wird versucht die Differenz zu den bekannten Bewertungen (r_{ij}) und den generierten ($q_j^T p_i$) zu minimieren. Das dabei erstellte Modell ist an die gegebenen Daten angepasst und muss für neuere Daten wieder neu berechnet werden, außer man setzt ein inkrementelles Verfahren ein (Song u. a., 2015).

$$\sum_{(i,j)\in K} (r_{ij} - q_j^T p_i)^2 \tag{2.6}$$

Ranking einer Empfehlung Die im vorherigen Schritt beschriebene Schätzung von neuen Bewertungen bildet die Ausgangsmenge von Artikeln, die dem Kunden vorgeschlagen werden können. Damit endet die Suche nach den richtigen Empfehlungen noch nicht. Das Problem die passenden Vorschläge zu finden kann man auch als ein Ranking verstehen (Rendle u. a., 2009), bei dem die die den Vorlieben des Kunden am ehesten ähneln höher in der Liste einsortiert werden.

Darstellung der Ergebnisse In Abbildung 2.11 sieht man jeweils zwei unterschiedliche Suchergebnisse für den Suchbegriff: "*rote jacke mit kapuze*". Die Präsentation der einzelnen Suchergebnisse stellt hier das berechnete Ranking für den Suchbegriff dar. Im linken Bild sieht man eine Variation aus männlichen und weiblichen Models, die zum größten Teil rote Jacken präsentieren. Es wurde während der Suche kein Kundenkonto genutzt und die Suchseite wurde zum ersten Mal aufgerufen. Damit wusste das System nichts über den Besucher außer dem angegeben Suchbegriff. Das gezeigte Ranking beinhaltet die beliebtesten und neusten Artikel für beide Geschlechter. Die Problematik beim erstmaligen Benutzen eines Systems wird als *Kalt-Start* Problem bezeichnet. Die Lösung dieses in diesem Fall ist das Anzeigen von beliebten Artikeln.

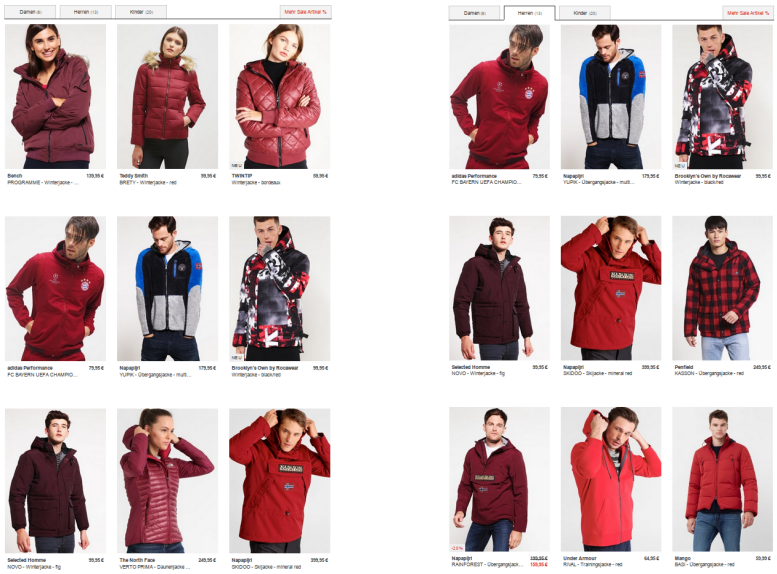

Abbildung 2.11: Zalando.de Suche: ”rote jacke mit kapuze”

Nach dem ersten Aufruf einer Artikeldetailseite für Herren wurde die Suche nochmal wiederholt. Beim zweiten Versuch wurde direkt eine Weiterleitung zum Herren-Sortiment von der Seite gemacht. Das Ranking konnte durch die neue Information eines Geschlechts eine bessere Auswahl an Artikeln vorschlagen. Diese Personalisierung kann jedoch auch negative Formen annehmen, wenn man außerhalb seiner gewohnten Muster neue Dinge entdecken will. Das Problem ist unter dem Begriff *Filter Bubble* bekannt (Örnek, 2016) (Pariser, 2011).

Bewertung Durch die Eigenschaften von Matrizen und den möglichen Operationen auf diesen kann man je nach Verfahren relativ einfach eine Parallelisierung[33] der kostspieligen Berechnung des Modells machen und zudem den Problemraum reduzieren. Trotzdem bleibt eine hohe Komplexität in der Berechnung durch die Anzahl der Daten. Zwei wichtige Aspekte die sowohl für *Memory-Based* als auch für die *Model-Based* Verfahren eine Hürde darstellen sind einmal das *Kalt-Start* und das *Filter-Bubble* Problem. Die Berücksichtigung dieser würde den Rahmen der hier vorliegenden Arbeit zu sehr ausweiten.

[33] Eine Referenz-Implementierung findet man in der LIBMF Bibliothek: `http://www.csie.ntu.edu.tw/~cjlin/libmf` (14.11.2016)

Eine wichtige Eigenschaft von Ansätzen, die eine Zerlegung des Problems in Faktoren verwenden besteht in der hohen Flexibilität neue Aspekte der Daten in die Berechnung einfließen zu lassen und dabei innerhalb des gleichen Lernverfahrens zu bleiben. Trotzdem muss man individuell je nach Domäne entscheiden, ob die lange Berechnung für eine erhöhte Genauigkeit sinnvoll ist.

Ein wichtiger Aspekt bei der *Matrix Factorization* besteht in der direkten Verwendung von z.B. abgegebenen Bewertungen. Im Gegensatz zu klassischen *Data-Mining* Verfahren werden hier die verborgenen Merkmale berechnet. Es muss kein explizites "*Feature Engineering*" betrieben werden, jedoch kann man gefundene Merkmale in die Berechnung einfließen lassen.

2.5.4 Factorization Machines

Das im Unterkapitel für Empfehlungen mit Matrix Factorization vorgestellte Modell bietet einen spezialisierten Ansatz für die Berechnung von Empfehlungen, der zu aller erst dafür gedacht ist mögliche Bewertungen vorherzusagen.

Das Lernverfahren dahinter wird damit oft für eine bestimmte Domäne verwendet und ist nicht immer Übertragbar. Zur Lösung von allgemeinen Vorhersage-Aufgaben kann man auf *Factorization Machines* (FM) zurückgreifen, die erstmals in Rendle (2010) vorgestellt wurden und über empirische Experimente eine hohe Effektivität aufweisen.

Diese können das vorgestellte Matrix-Modell nachahmen und benötigen als Eingabe statt der $U - I$ Matrix mit den Interaktionen zwischen Nutzer und Artikel die extrahierten Merkmale eines Datensatzes. Damit werden nicht mehr die verborgenen Beziehungen berechnet, sondern direkt die Zusammenhänge zwischen den Merkmalen. *FM* lassen sich am besten mit *Supportvektormaschinen*[34] vergleichen, wobei viele der negativen Eigenschaften wegfallen (Rendle, 2010).

Eine wichtige Verbesserung durch FM besteht in der linearen Komplexität, die den Be-

[34] Ist ein Klassifikator den man auch zur Regression einsetzen kann.

rechnungsaufwand verbessert und das Anlernen größerer Datensätze ermöglicht. Zudem besteht nicht mehr das Problem eine Basisschätzung für den Datensatz zu finden wenn dieser in Form einer dünn besetze Matrix vorliegt, wie bei der *Matrix Factorization* benötigt wird.

2.5.4.1 Faktorzerlegung mit Feature-Klassen

In Yan u. a. (2015) wird eine Kombination aus zwei Verfahren zur Klassifizierung verwendet. Das daraus entstandene Modell hat im *RecSys 2015* Wettbewerb den dritten Platz errungen. Die entscheidenden Komponenten der Lösung setzen sich wie folgt zusammen:

1. Feature Extraktion (manuell)
2. Anlernen neuer Features (automatisch)
3. Training der einzelnen Modelle
4. Kombination der Klassifzierer

Feature Extraktion Es wurden drei Kategorien als Quelle für die Merkmale des Datensatzes ausgewählt. Die erste bestand darin die bekannten Informationen[35] über die Artikel zu verwenden. In der zweiten Kategorie wurden die Nutzer durch die Länge des Einkaufs und in ihrem Verhalten beim Klicken auf Artikel beschrieben. Dabei viel auf dass die gekauften Artikel in einer Klicksequenz binär enkodiert wurden[36]. Dieser Ansatz verbindet die Information eines gekauften Artikels mit deren Position in der Klickstrecke. Beide Entscheidungen für die Wahl der Merkmale unterscheiden sich merklich gegenüber den anderen Teilnehmerlösungen (z.B. Romov u. Sokolov (2015) oder Yağci u. a. (2015)). Abschließend wurden die zeitlichen Abläufe der einzelnen Sessions (Wochentag, Monat, etc.) extrahiert und direkt als kategoriale Merkmale verwendet. Es wurde keine genaue Begründung für die gewählten Features gegeben, deswegen is es zu Vermuten das bei der Erstellung die besten über Experimente gefunden wurden.

[35] Kategorie, Preis, ID, etc.

[36] Bsp. Wenn drei Artikel A, B und C nach einander angeklickt wurden und B der gekaufte Artikel ist, ergibt sich die folgenden Darstellung: ”010”.

Training der einzelnen Modelle Für das eigentliche Modell wurde eine duale Strategie ausgewählt die *Gradient Boosting Decision Tree* (GBDT) mit FMs verbindet. Bei *GBDT* werden Entscheidungsbäume antrainiert. Diese haben einzeln betrachtet eine hohe Interpretierbarkeit. Die Schwächen einzelner Entscheidungsbäume liegen in den nicht konstant gleichen Modellen, die nach dem Anlernen entstehen können (Friedman, 2001). Diese Schwäche wird hier durch das *Boosting* Verfahren umgangen. Eine nähere Beschreibung zu diesem wird im nächsten Abschnitt gegeben.

Bewertung FMs gehören zu einer neuen Klasse von Verfahren, die sich erstmals mit vorhandenen Techniken über die Zeit messen müssen. Die Experimente aus dem Wettbewerb liefern jedoch vielversprechende Erkenntnisse, die für weitere Untersuchungen dieser sprechen.

2.6 Ensemble Learning

Da die meisten Lösungsansätze im *RecSys* Wettbewerb unterschiedliche Abwandlungen von Verfahren aus dem Bereich des *Ensemble Learnings* enthalten, werden in dem nun folgenden Abschnitt wichtige Konzepte angeschaut.

Bias und Varianz

Die Fehler eines Vorhersagemodells können jeweils in zwei Fehlerkomponenten unterteilt werden (James u. a., 2013). Die erste resultiert in einer Unteranpassung (engl. *bias*) an die Daten. Vergleicht man dafür das erstellte Modell mit den korrekten Daten kann man eine allgemeine Abweichung zu diesen messen. Die zweite Fehlerkomponente besteht in der Varianz (engl. *variance*) bei der einzelne Vorhersagen in ihrer Genauigkeit angeschaut werden.

Visualisierung der Fehlerkomponenten In Abbildung 2.12 sieht man die Visualisierung der einzelnen Fehlerarten. Je näher ein Punkt zur Zielscheibenmitte positioniert ist, desto besser ist das Vorhersagemodell dahinter. Eine Unteranpassung bei der alle Vorhersagen eine Abweichung zum Optimum haben kann auf ein schlechtes Modell zurückgeführt werden. Im Gegensatz dazu liegen die Vorhersagen bei einer hohen Varianz

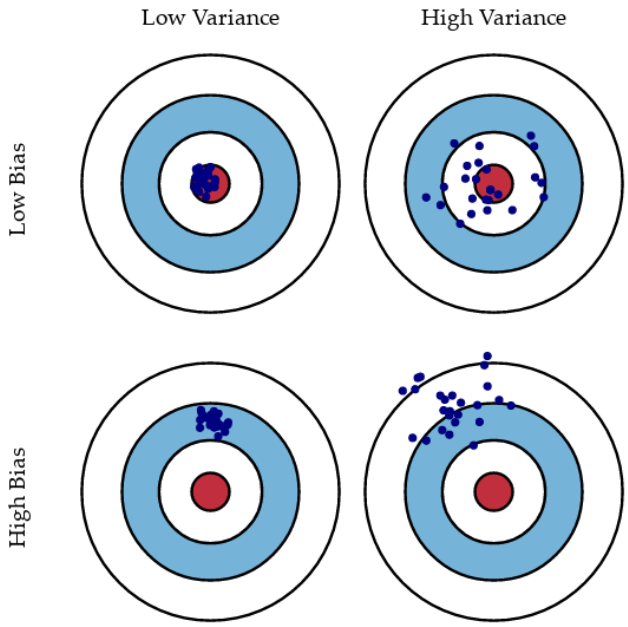

Abbildung 2.12: Fehlerkomponenten eines Modells

Quelle: http://scott.fortmann-roe.com/docs/BiasVariance.html (2016.11.14)

verteilt auf der Scheibe. Dies kann bei hohen Schwankungen im Datensatz passieren. Das grundlegende Problem besteht darin ein allgemeines Modell zu finden, was sowohl die Trainingsdaten als auch neue ungesehene Testdaten vorhersagen kann. Bei einer hohen Varianz hat man ein Modell, was die Trainingsdaten gut vorhersagen kann, jedoch auch das Rauschen darin mit lernt. Im Fall eines hohen Bias werden wichtige Zusammenhänge in den Trainingsdaten nicht komplett erfasst. Zur Behandlung dieser beiden Fehlerkomponenten wird im folgenden Unterkapitel das Konzept von *Ensembles* vorgestellt.

Ensemble

Ensemble Learning stellt einen anpassungsfähigen Ansatz im Gebiet des *Machine Learnings* dar (Murphy, 2012). Für die Lösung eines gemeinsamen Problems werden mehrere *schwächere* Algorithmen zur Klassifizierung verwendet. Es wird dabei versucht die Anfälligkeiten der einzelnen Modelle in ihren Fehlern durch die Kombination dieser zu reduzieren. Die Verbindung bekannter Algorithmen brachte auch schon im *Netflix* Wett-

bewerb eine Verbesserung in der Fehlerquote (Amatriain, 2013), wobei am Anfang die Skalierung auf größere Datenmengen ein Hindernis darstellte. In Gleichung 2.7 sieht man den Aufbau einer solchen Kombination, wobei w_m eine Gewichtung darstellt die dem jeweiligen Modell zugeordnet ist.

$$f(y|x,\pi) = \sum_{m \in M} w_m f_m(y|x) \tag{2.7}$$

Die jeweiligen Gewichtungen müssen jeweils über ein entsprechendes Lernverfahren bestimmt werden. Dafür gibt es unterschiedliche Ansätze. Diese werden im nächsten Abschnitt kurz anhand einer eingereichten Lösung aus dem *RecSys* Wettbewerb vorgestellt.

Boosting und Bagging

Nach Murphy (2012) ist Boosting ein *gieriger* Algorithmus für das Anpassen eines lernfähigen Basismodells in Form der Gleichung 2.7, wo w_m durch einen schwachen Algorithmus oder einen Basisklassifikator generiert wird. Der Kern von *Boosting* nach Freund u. a. (1996) besteht darin iterativ eine Sequenz von Vorhersagemodellen aufzubauen, dafür wird jede Instanz aus dem Datensatz eine Wertung gegeben die Aussagt wie *schwierig* es war diese zu klassifizieren. In den nachfolgenden Iterationen werden diese bewerteten Instanzen berücksichtigt und bekommen jeweils eine neue Gewichtung entsprechend der gemessenen Abweichung zugeordnet. In jedem iterativen Schritt wird so ein genaueres Modell aufbauend auf dem vorherigen erstellt. Die finale Vorhersage wird dann über eine gewichtete ”Mehrheitsentscheidung” aus allen Modellen ausgewählt (Friedman u. a., 2001).

Beim *Bagging* wiederum werden alle Teilmodelle über den Mittelwert miteinander verbunden. Dafür werden mehrere einzeln angelernte Modelle ausgewählt (Breiman, 1996). Eine gute Übersicht der bekannten Methoden sieht man in Dietterich (2000). Für die wissenschaftliche Arbeit hier wurde im Kapitel 4 ein genauer Vergleich zwischen diesen beiden Verfahren mit dem verfügbaren Datensatz gemacht.

Abbildung 2.13: Beziehung von Lernalgorithmen in ihrer Flexibilität und Interpretierbarkeit.

Quelle: James u. a. (2013)

Interpretierbarkeit vs. Flexibilität In Abbildung 2.13 sieht man die Einordnung der verschiedenen Verfahren, die beim *Ensemble Learning* zum Einsatz kommen. Man kann erkennen, dass Bagging und Boosting jeweils als sehr flexibel eingestuft werden. Die Bedeutung dieser Eigenschaft besteht darin, dass diese Verfahren auch komplexere Datensätze abbilden können. Der Nachteil dabei besteht in der nicht klaren Interpretierbarkeit der gemachten Ergebnisse. Es existieren jedoch auch gegenläufige Meinungen über die Bedeutung der Interpretierbarkeit eines Modells Lipton (2016), da jeder etwas anderes darunter versteht und das Konzept dahinter sehr komplex sein kann.

2.6.1 Ensemble mit Entscheidungsbäumen

In Romov u. Sokolov (2015) wird ein Modell vorgestellt, was durch die Kombination verschiedener Modelle ein besseres Ergebnis geliefert hat. Zudem ist diese wissenschaftliche Arbeit der Sieger im *RecSys 2015* Wettbewerb. Das entwickelte Modell ist eine Zweiphasen-Klassifizierung bei der zuerst erkannt wird, ob ein Kunde etwas kau-

fen wird, damit anschließend Empfehlungen für diesen vorgeschlagen werden können. In beiden Phasen wird jeweils eine binäre Klassifizierung ausgeführt.

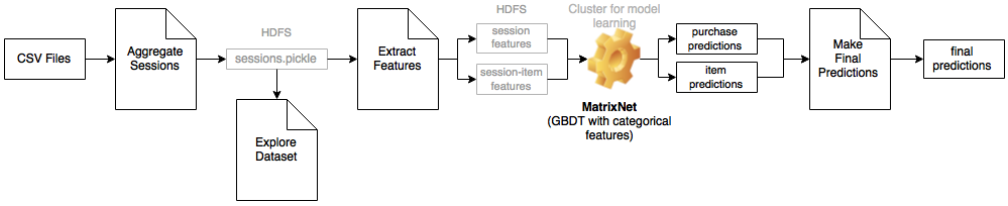

Abbildung 2.14: Pipeline der Teilnehmerlösung mit Gradient Boosting.

Quelle: `https://github.com/romovpa/ydf-recsys2015-challenge` (2016.10.26)

Analyse Pipeline

In Abbildung 2.14 sieht man die einzelnen Teilschritte der genutzten Daten-Analyse von Romov u. Sokolov (2015). Diese bestehen jeweils aus fünf Phasen die hintereinander in einer Pipeline ausgeführt wurden. Die verwendete Implementierung wurde nicht bereitgestellt, was die Nachbildung der Ergebnisse erschwert. Der Grund dafür war die Verwendung eines firmeninternen Frameworks[37].

Die Berechnung des Modells entstand verteilt auf 150 Maschinen innerhalb von 12 Stunden. Dabei wurde ein 60gb großes Modell generiert welches zur endgültigen Vorhersage eingesetzt wurde. Durch diese veröffentlichten Benchmarks erkennt man eine sehr hohe Komplexität in der Berechnung, die sich wiederum in einer hohen Genauigkeit ausgezahlt hat. Hier zeigen sich die sehr großen Anforderungen um das Modell zu berechnen, damit ist das Nachstellen auf einem konventionellen Computer nicht praktikabel.

Extraktion von Features Während dieses Schrittes wurden grundsätzliche Features, die in Korrelation mit dem Nutzerverhalten stehen, extrahiert. Die Exploration der Daten wurde dabei nur oberflächlich ausgeführt. Die dabei gefundenen Merkmale konnten zu zwei Kategorien zugeordnet werden, aus denen die endgültigen Features generiert wurden. Zum einen aus dem zeitlichen Ablauf der Session[38] und zum anderen aus der

[37] `https://yandex.com/company/technologies/matrixnet/`
[38] Bsp. Start und Ende der Session oder die Länge einer Session.

Interaktion mit den Artikeln innerhalb einer Session[39]. Diese Teilung wurde auch in Cohen u. a. (2015) so ausgeführt. Eine wichtige Entscheidung die hier getroffen wurde ist, dass die kategorialen Merkmale übernommen wurden. Für solche Merkmale muss eine Klassifizierung oder eine Mustererkennung gemacht werden, da z.B. eine Ordnung oder Abhängigkeit in den einzelnen Werten besteht (Murphy, 2012). Aus dieser Entscheidung kann man die lange Berechnungszeit schlussfolgern, da kategoriale Variablen viele unterschiedliche Werte annehmen können, die zu berücksichtigen sind.

In Yağci u. a. (2015) wurde sich ebenfalls für das manuelle Definieren von Features entschieden, wobei am Ende die Modellauswahl und Genauigkeit dadurch limitiert wurde. Die Abhängigkeiten zwischen den Merkmalen mussten beispielsweise von Hand kodiert werden.

Blackbox Algorithmus Wie schon erwähnt ist die Implementierung Romov u. Sokolov (2015) nicht einzusehen. Die Autoren beschreiben jedoch die Lösung als ein Verfahren, das mit *Gradient Boosting Decision Trees* arbeitet. Dies ist die gleiche Methode die auch in Yan u. a. (2015) erfolgreich eingesetzt wurde, jedoch nicht zum Sieg reichte. Es kann spekuliert werden, ob die finale Auswahl an Features oder die Einstellung der einzelnen Parameter des Verfahrens zur schlechteren Wertung geführt haben.

Die Begründung zu dieser Wahl besteht darin, dass lineare Verfahren zur Klassifizierung eine zu schwache Leistung in der Genauigkeit liefern und unfähig sind komplexe Interaktionen zwischen den Kunden und Artikeln zu finden. Das endgültige Modell verwendet eine hohe Anzahl an kategorialen Features. In Yağci u. a. (2015) wiederum wird ein *Random Forest*(Breiman, 2001) eingesetzt, das einzelne Entscheidungsbäume zur Aufteilung des Problems nutzt. Die Implementierung ist zu dem einfacher nachzustellen, da *scikit-learn* als Basis genutzt wird und alle verwendeten Parameter beschrieben sind.

[39] Bsp. Anzahl an Klicks eines Artikels in der Session.

2.6.1.1 Bewertung

Die Präsentation der wissenschaftlichen Ausarbeitung von Romov u. Sokolov (2015) besticht darin, dass der ausgeführte Prozess veröffentlicht wurde. Dadurch kann man relativ einfach die ersten Schritte nachstellen, wenn man einen *Apache Spark* Cluster[40] aufbaut oder diese Teile auf eine andere Bibliothek migriert.

Die Implementierung wurde leider nicht veröffentlicht und ist damit eine Blackbox. Es wurde auch kein Vergleich zu anderen Methoden präsentiert. Alternative Verfahren wurden als nicht geeignet eingestuft. Es ist also von Interesse einen genaueren Vergleich zwischen diesen beiden Verfahren zu machen, da dieser im Grunde zwischen *Boosting* und *Bagging* gemacht werden muss. Eine weitere Auffälligkeit sieht man in dem gemachten Aufwand in der Berechnung, welche sehr viele Ressourcen benötigte. Es ist fragwürdig inwieweit man das Modell in einem realen System verwenden kann, wenn jedes mal ein solcher Aufwand betrieben werden muss. Dies war z.B. eine der Hürden, die im *Netflix* Wettbewerb beim Sieger-Modell aufgekommen ist (Amatriain, 2013).

[40] `https://spark.apache.org/docs/latest/spark-standalone.html` (2016.11.17)

3 Exploration der Daten

In diesem Kapitel werden jeweils wichtigen Aspekte des benutzten Datensatzes explorativ untersucht. Dazu gehört eine ausführliche Beschreibung des aufgezeichneten Formates und die Eigenschaften eines solchen kommerziellen Datensatzes. Die Funde werden dann dafür genutzt um besondere Merkmale und Hypothesen zur Beschreibung des Datensatzes aufzustellen.

3.1 Allgemeines Vorgehen beim KDD Prozess

Wie schon in der Einleitung beschrieben werden die unterschiedlichsten Disziplinen für die Exploration und Analyse des Datensatzes eingesetzt. Eine davon besteht im KDD Prozess, welcher spezifische Vorgehensschritte bei der Erfüllung des Ziels neues Wissen zu entdecken definiert (Fayyad u. a., 1996). Dabei wird zuerst das Verständnis der Domäne aufgebaut bzw. definiert. Anschließend findet die Sammlung oder Auswahl eines Datensatzes statt, der für die Analyse genutzt werden soll. Dieser muss wie erwähnt bereinigt und transformiert werden, damit man *Machine Learning* Verfahren darauf anwenden kann. Das richtige Verfahren zu diesem Zweck muss je nach Art der gefundenen Merkmale im Datensatz dann gewählt werden. Die Ergebnisse aus den einzelnen Schritten werden interpretiert und nach jedem Durchlauf für die Verfeinerung des Prozesses genutzt.

Nach dem beschriebenem Vorgehen beinhaltet das aktuelle Kapitel alle Schritte bzw. Teilaspekte[1] vor der eigentlichen Analyse und Interpretation, die wiederum in Kapitel 4 beschrieben werden.

3.2 Herausforderung

Wie in der Einleitung beschrieben bestehen die Ziele darin einen Datensatz mit implizitem Feedback durch *Data-Mining* Verfahren besser zu Verstehen und die Eignung zu

[1] Es werden Vorschläge für die Transformation und Bereinigung des Datensatzes gemacht, die jedoch zu einem späteren Zeitpunkt umgesetzt werden.

erkunden.

Dafür müssen die jeweiligen Merkmale des Datensatzes identifiziert werden, um darauf eine Analyse auszuführen. Weiterhin sollen die gemachten Erkenntnisse bei der Verwendung der Modelle aufgezeigt werden. Davor muss jedoch der Datensatz bereinigt und in das geeignete Format überführt werden.

3.3 Untersuchung vom implizitem Feedback

3.3.1 Datenquelle

Der eingesetzte Datensatz[2] vom *Yoochoose* Unternehmen wird als ausschließliche Datenquelle für alle Evaluierungen eingesetzt. Die enthaltenen Daten wurden 2014 auf einer unbekannten E-Commerce Plattform innerhalb von sechs Monaten aufgezeichnet. Damit beinhalten diese alle Interaktionen von Kunden mit Artikeln zwischen den Monaten April und September.

Informationen zur Session

Zum Download wurde ein Archiv mit drei Dateien bereitgestellt. In der ersten *yoochoose-clicks.dat* sind alle Sessions drin die einen oder mehrere Klicks auf Artikel ausgeführt haben. Zusätzlich zu den angeklickten Artikeln ist der Zeitpunkt und die Kategorie vermerkt. Verbunden werden diese Informationen mit einer Session ID. Diese repräsentiert nicht direkt einen spezifischen Kunden, sondern eher den Besuch eines Kunden und seine Interaktionen mit der Seite. Es könnte somit auch möglich sein, dass der gleiche Kunde unter einer anderen Session ID vorkommen kann. Insgesamt sind es *9 249 729* eindeutige Sessions.

[2] Download: `http://2015.recsyschallenge.com/challenge.html` (2016.06.05)

3.3.2 Klickstrecken

Das Format der *yoochoose-clicks.dat* kann man in der Tabelle 3.1 sehen. Die Größe der Datei ist *1,5 GB* mit insgesamt *33 003 944* Zeilen. In *Pandas* eingelesen verbraucht die komplette Datei im Speicher knapp *2 bis 3 GB*.

Kategorien eines Artikels Von den angeklickten Artikeln sind *10 769 610* speziell beworbene Artikel und werden im Datensatz mit der Kategorie ”S” markiert, die für die weitere Verarbeitung auf einen numerischen Wert abgebildet wurde. Zudem fällt es auf, dass mit *16 337 653* mehr als die Hälfte aller Artikel keiner Kategorie zugeordnet sind. Die meisten weiteren Artikel ordnen sich in die Kategorien zwischen den Zahlen 1 und 12 ein. Durch das Fehlen so vieler Kategorien und den großen Anteil einer einzigen, sieht man ein Ungleichgewicht in den Metadaten zu den Artikeln. Die übrigen Artikel haben eine Zuordnung zu einem bestimmten Markensortiment.

Session ID	Zeitstempel ID	Artikel ID	Kategorie
2	2014-04-07T14:02:36.889Z	214551617	0
3	2014-04-02T13:17:46.940Z	214716935	0
3	2014-04-02T13:26:02.515Z	214774687	0
4	2014-04-07T12:09:10.948Z	214836765	0

Tabelle 3.1: Extrahierte Nutzerhistorie

Es sind während der Sessions *52 739* eindeutige Artikel angeklickt worden, davon sind *35 228* zu mehr als einer der insgesamt *339* Kategorien zugeordnet. Daraus kann man schließen, dass ein Artikel zu mehr als einer Kategorie gehören kann. Hier kann leider kein Unterschied zu redaktionellen Fehlern bestimmt werden. In Tabelle 3.2 kann man die Verteilung zwischen den einzelnen Kategorien sehen. Diese mehrfache Zuordnung zu einer Kategorie bedeutet für die Klassifizierung einen höheren Aufwand, da die hohe Überlappung eine klare Unterscheidung erschwert.

3.3.3 Käufe

In der zweiten Datei *yoochoose-buys.dat* sind *1 150 753* weitere Zeilen, die die jeweiligen Käufe für die besprochenen Sessions definieren. Die Artikel, die final gekauft wurden müssen nicht unbedingt im Trainingsset vorkommen, wobei die meisten das tun.

Kategorie	Keine	1	2	3	4	5	6
Anzahl	16337653	1671754	1292249	789713	480569	471923	414696
Kategorie	7	8	9	10	11	12	Promotion
Anzahl	389910	44840	105282	69820	70264	19357	10769610

Tabelle 3.2: Anzahl der Artikeln pro Kategorie.

Das verwendete Format ähnelt dem aus der ersten Datei, jedoch ist die Spalte mit den Kategorien durch den Preis und die Anzahl ersetzt worden (siehe Tabelle 3.3). Das erste was auffällt ist, dass im Gegensatz zur Anzahl der Sessions und gemachten Klicks eine viel kleinere Zahl an Käufen gemacht wurde. Genauer gesagt in ca. 5 % der Sessions wird auch etwas gekauft. Damit besteht das Problem, dass eine Klasse im Datensatz dominiert. Das kann dazu führen, dass ein so angelerntes Modell ausschließlich diese eine Klasse vorschlägt, da diese am Wahrscheinlichsten ist. Im nachfolgenden Kapitel 4 wird ein Vorschlag gemacht, um dieses Problem zu umgehen.

Session ID	Zeitstempel	Artikel ID	Preis	Anzahl
420374	2014-04-06T18:44:58.314Z	214537888	12462	1
420374	2014-04-06T18:44:58.325Z	214537850	10471	1
281626	2014-04-06T09:40:13.032Z	214535653	1883	1

Tabelle 3.3: Extrahierte Kaufhistorie der Nutzer.

Zeitliche Verteilung der Käufe In Abbildung 3.1 sieht man die prozentuale Verteilung der einzelnen Käufe auf die einzelnen Monate und Wochentage. Daraus kann man sehen, dass die meisten Käufe vor und nach den Sommerferien ausgeführt wurden[3]. Dass lässt darauf schließen, dass während der Urlaubszeit weniger Käufe getätigt werden, weil man vielleicht nicht anwesend ist. Das gleicht sich soweit mit den Erkenntnissen aus den vorherigen Arbeiten zu diesem Datensatz (Cohen u. a., 2015)(Yağci u. a., 2015)(Romov u. Sokolov, 2015). Eine Alternative Erklärung könnte in einem saisonal abhängigen Sortiment von Artikeln liegen. Betrachtet man die Käufe pro Wochentag sieht man, dass die meisten Käufe ums Wochenende herum getätigt werden und einen Tiefpunkt am Dienstag erreichen. Beides sind wertvolle Informationen die die Bedeutung der Zeitpunkte als

[3] Amazon Einnahmen nach Quartal sortiert: `https://www.statista.com/statistics/276418/amazons-quarterly-net-income/` (2016.11.21)

wichtige Merkmale kennzeichnen. Die aufgestellten Hypothesen zur Sommerzeit wurden durch das Befragen von Branchen Experten bestätigt.

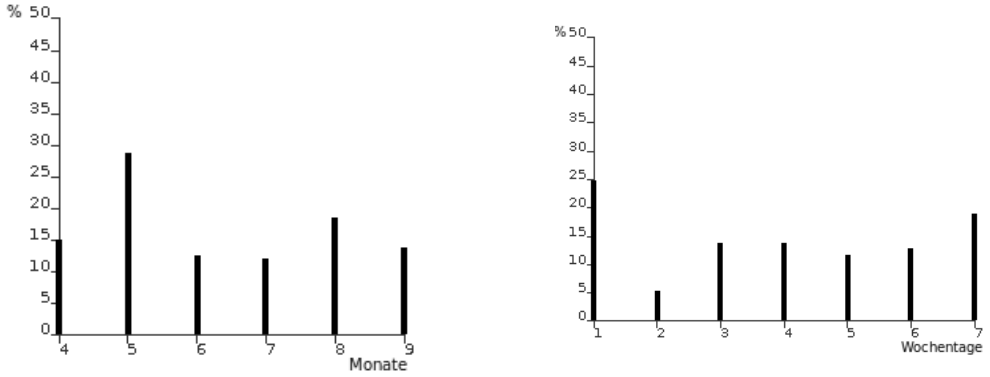

Abbildung 3.1: Der Anteil an Käufen pro Monat (links) und Wochentag (rechts).

Informationen eines Artikels Innerhalb der Sessions sind alle gekauften Artikel voneinander getrennt aufgelistet. Beim Durchschauen der Sessions wird dann klar, dass ein und derselbe Artikel mit einer anderen Anzahl nur wenige Sekunden nach dem ersten Eintrag wieder auftaucht. Für die weitere Verarbeitung können diese Einträge mit dem letzten Zeitpunkt aufsummiert werden oder man ignoriert die gekaufte Menge als Merkmal. Einzelne Artikel werden auch nach einem Kauf noch mal angeklickt. Es kann vielleicht sein, dass der Kunde seinen gemachten Kauf so überprüfen möchte. Zudem besitzen mehr als die Hälfte aller Einträge weder eine Anzahl oder einen Preis, da diese Werte wohl für diesen Zeitpunkt nicht zur Verfügung gestanden haben (siehe Abbildung 3.2). Wie auch in den Kategorien besteht hier ein sehr starkes Ungleichgewicht. Deswegen sind der Preis und die Anzahl fragwürdige Merkmale. Eine Abhilfe könnte durch die Verkleinerung dieser großen Klasse von Einträgen bringen. Die Daten könnten auch bereinigt werden, indem man aus vorherigen Käufen die Preise für die fehlenden Artikel ausließt. Damit verfälscht man jedoch den Datensatz. Denn Artikelpreise können sich über die Zeit verändern und das Kaufverhalten beeinflussen, wie man an den vielen Käufen in der *Sales* Kategorie sehen kann.

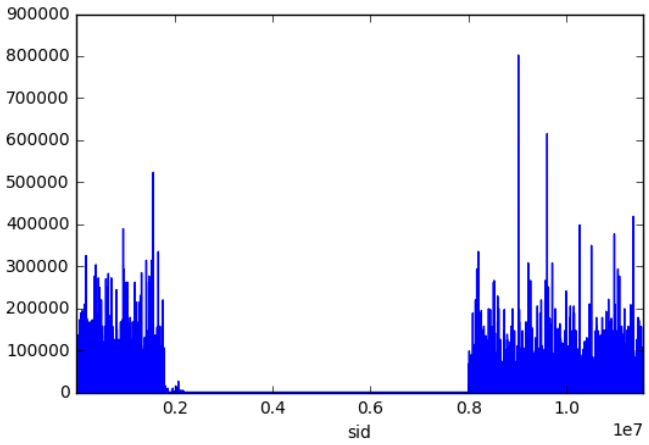

Abbildung 3.2: Nach den Session IDs aufsummierte Preise.

3.3.4 Verbindung von Klickstrecken mit Käufen

Als nächsten Schritt wurde die Verbindung beider Dateien vollzogen, damit man alle Sessions in ihrer vollständigen Form vorliegen hat. In Abbildung 3.3 sieht man jeweils eine kleine statistische Beschreibung über die Dauer aller Sessions. Die kompletten Sessions bilden die Grundlage, aus der wiederum nach einer Extraktion von Features die angewandten Verfahren ihre Daten zum Anlernen (*Train*) und Evaluieren (*Test*) verwenden.

	timestamp
	duration
count	9249729
mean	0 days 00:06:44.859064
std	0 days 00:13:02.157610
min	0 days 00:00:00
25%	0 days 00:00:33.596000
50%	0 days 00:02:14.387000
75%	0 days 00:06:57.300000
max	2 days 20:18:57.996000

Abbildung 3.3: Beschreibung der Sessiondauer.

Da im Datensatz jeweils auch Einträge mit nur einem Klick existieren ist der kleinste vorhanden Wert einer Session mit 0 beziffert. Wie man sieht liegt der Mittelwert bei 6 Minuten bei einer Standardabweichung von 13 Minuten.

Alternative zur zeitlichen Dauer Die Abweichung in der zeitlichen Dauer ist recht groß, jedoch kann man das Kriterium der Dauer auch in Form von gemachten Klicks innerhalb einer Session abbilden. In Abbildung 3.4 sieht man die jeweiligen Werte für die gemachten Klicks in einer Session. Auch hier sieht man eine große Abweichung unter den Werten.

	aid
count	9249729.0000
mean	3.5681
std	3.7875
min	1.0000
25%	2.0000
50%	2.0000
75%	4.0000
max	200.0000

Abbildung 3.4: Beschreibung der Sessiondauer anhand gemachter Klicks.

Warenkorbinhalt Wie schon in Abschnitt 2.5.3.1 erklärt werden einem Kunden oft die beliebtesten N Artikel als Basisschätzung vorgeschlagen. Um dem nachzugehen sieht man in Abbildung 3.5 die Verteilung von eindeutigen Artikeln in allen Warenkörben. Dabei fällt auf dass die meisten Kunden genau wussten was sie kaufen wollten und dementsprechend nur eine Art von Artikel im Warenkorb gelegt haben[4]. Eine weitere Eigenschaft, die man in den Daten sehen kann ist das von den beliebtesten Artikeln um die 90% im Warenkorb gelandet sind. Hinter dieser Verteilung kann man das Konzept

[4]Es wurden nur die eindeutigen Artikel angeschaut und nicht die wirkliche Anzahl eines Artikels. Ein Warenkorb mit zwei roten Jeanshosen wird als Kauf mit einem Artikel gezählt.

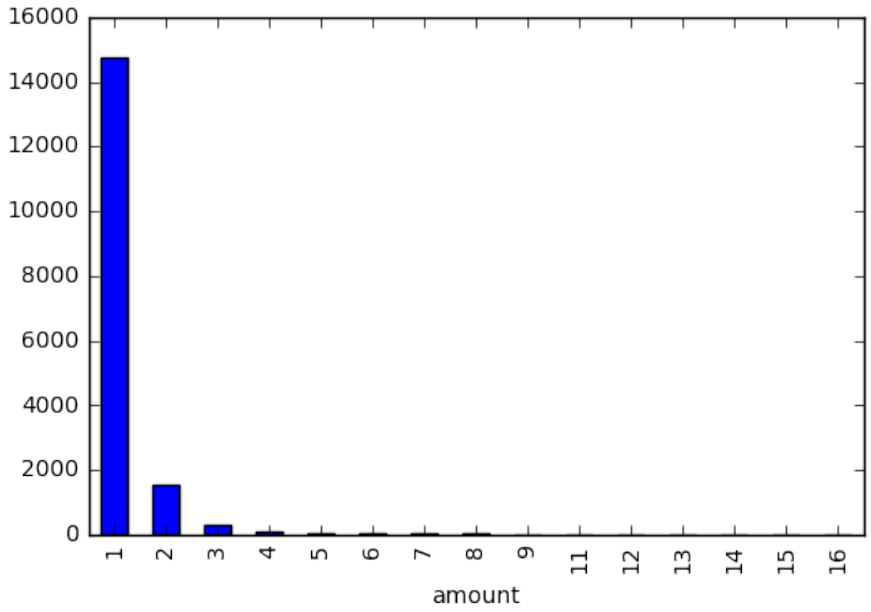

Abbildung 3.5: Verteilung der Anzahl von gekauften Artikeln im Warenkorb.

für die *Long Tail* Theorie wieder finden (Hitt u. Anderson, 2007), die sich genau mit dieser Verteilung im E-Commerce beschäftigt.

Datendichte Als nächstes wurde sich die Dichte der Daten angeschaut. Zu diesem Zweck wurden alle eindeutigen Artikel und Sessions herausgesucht und miteinander in ihrem Aufträten verglichen. Das führt zu einer sehr dünn besetzten Matrix bei der weniger als ein Prozent gefüllt ist. Dies ist keine gute Voraussetzung für *Nearest Neighborhood* oder *Matrix Factorization* Verfahren, jedoch können an dieser Stelle die Factorization Machines eingesetzt werden. Alternativ kann man versuchen die Unausgeglichenheit des Datensatzes, wie auch schon im Unterabschnitt 3.3.3 zu den Käufen erwähnt, zu reduzieren.

3.3.5 Fazit

In diesem Kapitel wurden auffällige Aspekte des Datensatzes aufgezeigt, die ein tieferes Verstehen des gegebenen Problems ermöglichen sollen. Die schnelle Exploration und Visualisierung der Daten wurde durch den Einsatz von *Jupyter Notebooks* in Ver-

bindung mit den in Unterabschnitt 2.2.3 beschriebenen Bibliotheken stark vereinfacht.

Einige der wichtigsten Merkmale konnten so schnell entdeckt werden. So z.B. der Zusammenhang zwischen den Zeiten einer Woche oder eines Tages mit den gemachten Käufen. Man kann klare Kaufmuster je nach Wochentag und Uhrzeit erkennen. Die vollständige Bedeutung dieser Zusammenhänge kann jedoch nur erahnt werden, da keine direkten Befragungen der Kunden möglich waren. Jedoch wird in Kapitel 4 versucht bei der Erstellung des Modells darauf einzugehen.

Zudem konnte gezeigt werden, dass die meisten Kunden einen recht kleinen Warenkorb kaufen, zumeist nur aus einem Artikel bestehend. Die dabei vorgekommenen Artikel besitzen recht wenige Metadaten, die sich aus mehreren Kategorien und Preisen zusammensetzen. Darüber hinaus fehlt die Hälfte aller Metadaten, so dass die Menge und der Preis als Merkmale potentiell keine große Signifikanz besitzen. Die gekauften Artikel wiederum gehören größtenteils zu den N Beliebtesten, wobei in den Restkäufen dann viele Käufe mit individuellen Artikeln getätigt wurden.

4 Erstellung eines Modells

4.1 Methodik beim Erarbeiten des Modells

In Abschnitt 3.1 wurde ein grober Umriss des *KDD* Prozesses gemacht (siehe auch Abbildung 4.1). In dem nun folgenden Abschnitt wird eine genauere Einordnung der einzelnen Schritte, die für diese Arbeit genutzt wurden gemacht. Dafür muss man sich jedoch die dafür wichtigen Schritte des Prozesses genauer anschauen. Alle hier verwendeten Definitionen stammen jeweils aus Fayyad u. a. (1996).

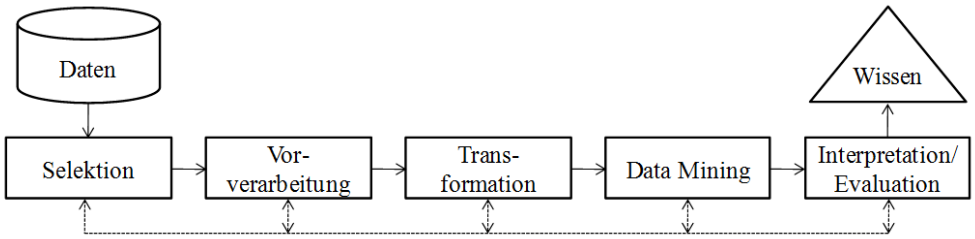

Abbildung 4.1: Aufbau von Knowledge Discovery in Databases.

Domänenspezifisches Wissen aufbauen & Ziele definieren

Der vollständige KDD Prozess ist interaktiv und iterativ in all seinen Schritten. Darüber hinaus müssen unterschiedliche Entscheidungen zwischen diesen getroffen werden, die den ganzen Verlauf beeinflussen können. Eine der ersten Aufgaben besteht im Aufbau von domänenspezifischem Wissen und der Skizzierung von den verfolgten Zielen.

In Kapitel 2 wurden dafür alle beteiligten Perspektiven und Themengebiete vorgestellt, die aus den Herausforderungen vom *E-Commerce* und den Verfahren zur Erstellung von Empfehlungen bestehen. Darüber hinaus wurden wichtige Aspekte vom impliziten Feedback beleuchtet. Die hier verfolgten Ziele sind die Beschreibung des Prozesses bei der Erkenntnisgewinnung und das Klassifizieren vom Nutzerverhalten.

Datenauswahl

Im zweiten Schritt wird die Sammlung eines Datensatzes beschrieben, die die Selektion einer Untermenge beinhalten kann. Damit soll eine Fokussierung auf den wichtigen Teil der Daten stattfinden.

Dieser Schritt gestaltet sich einfach, da der relevante Datensatz über den *RecSys 2015* Wettbewerb bereitgestellt wurde und somit kein weiterer Aufwand notwendig ist. Eine kurze Einführung in den Wettbewerb und die Organisation dahinter wurde in Abschnitt 2.3 gegeben.

Datenbereinigung

Ein wichtiger Schritt für die weitere Verarbeitung besteht in der Säuberung von Fehlern und Ausreißern in den Daten. Durch diese Vorverarbeitung garantiert man, dass nur relevante Daten in die Erstellung eines Modells einfließen. Darüber hinaus muss sich eine Strategie überlegt werden, wie man fehlende Daten behandelt.

In Abschnitt 3.3 wurden für die Behandlung von fehlenden Daten einige mögliche Strategien vorgestellt. Im Fall der fehlenden Preise und Kategorien wurde festgestellt, dass auch eine Auswahl alternativer Merkmale anstatt dieser verwendet werden kann. Die gesamte Anzahl an möglichen Merkmalen in diesem Fall ist sehr groß und ermöglicht diese Entscheidung. Da der Datensatz von einer anderen Partei aufgezeichnet wurde und anonymisiert ist, kann keine Entscheidung getroffen werden ob fehlerhafte Daten vorhanden sind. Dies wird in Kauf genommen und über ein Verfahren abgeschwächt, was mit einem solchen Datenbestand umgehen kann (Breiman, 2001).

Transformationen

Im vierten Schritt geht es um die Reduktion bzw. Projektion des verwendeten Datensatzes in ein Format, was die Daten während des Prozesses repräsentieren soll. Dafür werden beispielsweise zielbezogene Merkmale aus dem Datensatz extrahiert und für den Rest des Prozesses weiter verwendet.

Hierfür wurde in Kapitel 3 eine ausführliche Exploration der Daten beschrieben, welche als Grundlage für die Extraktion von wichtigen Merkmalen in Abschnitt 4.2 genutzt wurde. Zudem wird in Abschnitt 4.3 eine Strategie vorgeschlagen, um den unausgeglichen Datensatz in die passende Form zu bringen und gleichzeitig zu reduzieren. Die Reduzierung des Datensatzes ist bei der ursprünglichen Größe ein sehr wichtiger Vorverarbeitungsschritt.

Auswahl einer Methode

Je nachdem welche Ziele man verfolgt muss dementsprechend eine passende Methode wie z.B. Regression oder Klassifizierung ausgewählt werden um ein Modell was die Daten abbildet zu erstellen. Dafür wurden in Abschnitt 2.2 diese klassischen *Data-Mining* Methoden vorgestellt. Für den Bereich der Empfehlungssysteme ist in Abschnitt 2.5 eine Übersicht gegeben worden. Die finale Auswahl viel auf die Klassifizierung, weil die Bestimmung ob ein Artikel in einer Session gekauft wird darauf hinweist ob die Session mit einem Kauf enden wird. Damit kann man beide Fragen aus dem *RecSys* Wettbewerb mit einem Modell beantworten. Das performanteste Modell zu diesem Zweck wird durch die Experimente in Abschnitt 4.5 ermittelt.

Erste Ansätze finden

Im sechsten Schritt wird eine explorative Untersuchung mit Modellen ausgeführt, um erste Hypothesen zum Datensatz aufzustellen. Mit diesen werden dann markante Muster in den Daten gesucht. Dafür muss jedoch die Entscheidung getroffen werden ob man die Genauigkeit eines Modells im Fokus hat oder die Interpretierbarkeit.

In der hier gemachten Arbeit werden sich jeweils beide Aspekte angeschaut, da schon Lösungen für den bestehenden Datensatz existieren jedoch in ihrer Beschreibung und Umsetzung der Verfahren nicht komplett sind. Deswegen wurde eine ausführliche Untersuchung der Daten und der Verfahren gemacht.

Data-Mining

Die letzten beiden Schritte im Prozess werden zusammengefasst betrachtet, da diese eng miteinander verbunden sind. Zu diesen beiden gehört zum einen die Anwendung der Data-Mining Verfahren und zum anderen die Interpretation der Ergebnisse daraus. Die dadurch gemachten Erkenntnisse können jeder Zeit den kompletten Prozess beeinflussen. Die Erläuterung zur Interpretation wird ausführlich in Abschnitt 4.5 gemacht.

4.2 Identifizierung der Merkmale

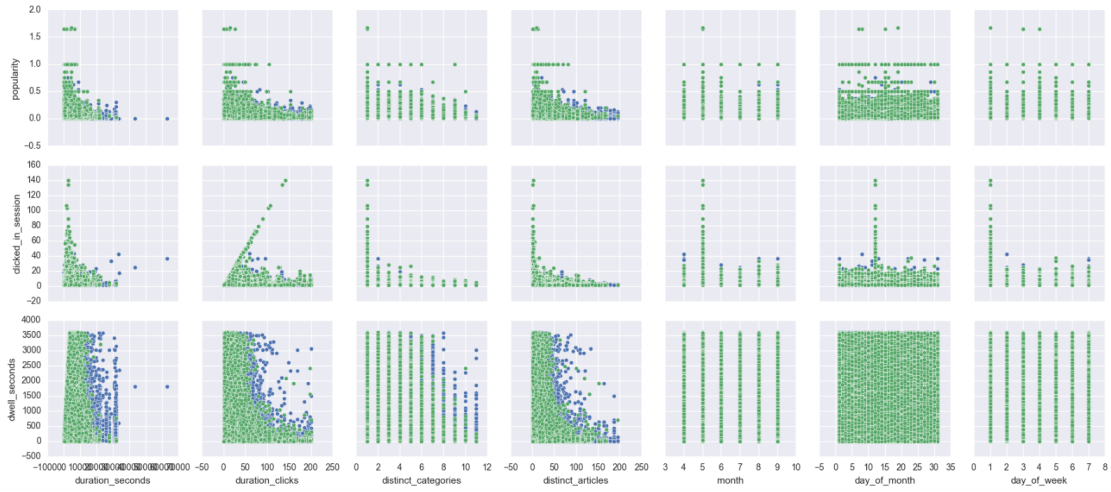

Abbildung 4.2: Paarweiser Vergleich von Merkmalen. (grün=Kauf, blau=kein Kauf)

Durch die Funde im vorherigen Kapitel sind einige bestimmte Merkmale des Datensatzes aufgefallen. Diese wurden genutzt um eine Extraktion von Features durchzuführen, die in Tabelle 4.1 und Tabelle 4.2 zu sehen sind. Dabei sind in den Spalten mit dem Buchstaben **S** die Features, die die Session beschreiben. Alle Spalten mit **A** definieren die Merkmale eines Artikels der angeklickt wurde. Diese Features bilden die Grundlage beim eigentlichen *Data Mining* Prozess und werden über ein eigens dafür geschriebenes Java Programms extrahiert. Die finale Aufstellung der Features wird iterativ bestimmt. Dabei werden die folgenden Schritte jeweils ausgeführt:

1. Feature über explorative Analyse auswählen.
2. Feature aus den Daten extrahieren.
3. Verfahren mit neuen Featureset anlernen und in ihrer Genauigkeit vergleichen.

Die Identifizierung von expliziten Merkmalen in den aufgezeichneten Daten ist einer der großen Unterschiede gegenüber der *Matrix Factorization*, die diese verborgenen Merkmale selbständig findet. Damit ähneln die *Factorization Machines* in den Eingabedaten den klassischen Data-Mining Verfahren.

S1	Dauer der Session in Sekunden.
S2	Dauer der Session in Klicks.
S3	Monat des gemachten Klicks. (4-9)
S4	Tag des Monats. (1-31)
S5	Wochentag des gemachten Klicks. (1-7)
S6	Stunde des Tages. (0-23)
S7	Minute der Stunde. (0-59)
S8	Anzahl an eindeutigen Kategorien in der Session.
S9	Anzahl an eindeutigen Artikeln in der Session.

Tabelle 4.1: Auflistung aller Session Features.

A1	Popularität des Artikels unter allen gekauften Artikeln (Käufe / Klicks).
A2	Ist dieser Artikel als erstes angeklickt worden.
A3	Ist dieser Artikel als letztes angeklickt worden.
A4	Die Zeit zwischen diesem und dem nächsten Klick. (Verweildauer)
A5	Gesamtanzahl an Klicks auf diesen Artikel in der Session.

Tabelle 4.2: Auflistung aller eingesetzter Artikel Features.

4.3 Unausgeglichener Datensatz

In Kapitel 3 wurde die Größe des Datensatzes mit seinen Merkmalen untersucht, dabei sind mehrere Dinge aufgefallen. Es gibt eine ungleichmäßige Verteilung von Käufen gegenüber der Anzahl an gemachten Sessions. Damit besteht die Gefahr, dass bei einer Aufteilung des Datensatzes in Trainingsset und Testset die Verteilung der beiden

Klassen unausgeglichen ist. Das eingesetzte Lernverfahren könnte dann über die dominierende Klasse negativ beeinflusst bzw. verzerrt werden (Nisbet u. a., 2009, Seite 240). Um dieses Problem zu lösen gibt es unterschiedliche Ansätze zu den bekanntesten gehören das *Over-Sampling* und das *Under-Sampling* (Nisbet u. a., 2009, Seite 240). Beim ersteren erhöht man die unterlegene Klasse über Kopien von Einträgen. *Under-Sampling* reduziert wiederum die überrepräsentierten Klasse aus dem Datensatz. Der erste Ansatz kommt für kleinere Datensätze in Frage und wird deswegen hier nicht verwendet.

Über *Under-Sampling* werden alle positiven Sessions mit Käufen in den finalen Datensatz übernommen und eine gleich große Anzahl an negativen Fällen. Die Auswahl der letzteren wird jeweils zufällig ausgeführt und erzeugt am Ende einen Datensatz mit insgesamt mehr als 5 Millionen Klicks, der damit etwa 16 % des ursprünglichen Datensatzes ausmacht.

4.4 Datenbereinigung

Kategorien

Wie schon erwähnt gehören die jeweiligen Artikel mehreren Kategorien an. Drei davon haben bestimmte Bedeutungen. Zum einen existieren für die Hälfte der Artikel keine eindeutigen Kategorien, zum anderen wurde mehr als die Hälfte aller Käufe in der Sales-Promotion Kategorie getätigt. Die letzte besondere Gruppe von Artikeln sind, die die einer Marke[1] zugeordnet sind. Zur Vereinfachung wurde dieses kategoriale Merkmal normalisiert, so dass nur noch die Kategorien in dem Bereich von 1 bis 15 auftreten können.

4.5 Experimente

4.5.1 Random Forest

Im ersten Ansatz wurden sich Entscheidungsbäume angeschaut, die in Form eines Ensembles in ihrem Mittelwert kombiniert werden. Die dafür genommene Implementie-

[1] Tom Tailor, Fila, Nike, etc.

rung ist mit einem *Random Forest* (RF) realisiert (Breiman, 2001). Im Gegensatz zu reinen Entscheidungsbäumen hat ein RF kein Problem mit Überanpassung durch Rauschen. Des Weiteren können große Mengen an *qualitativen* Merkmalen verarbeitet werden. Jedoch berücksichtigen *RF* keine bestehenden Zusammenhänge zwischen den einzelnen Merkmalen untereinander.

Ablauf des Algorithmus Die Komponente des Zufalls und die Kombination als Ensemble liefern für Datensätzen mit vielen Merkmalen und komplexen Zusammenhängen konstante Ergebnisse. *RF* funktionieren im Groben nach dem folgenden Algorithmus aus Friedman u. a. (2001):

1. Für Baum b aus der Menge B führe aus:
   a) Wähle eine n große Stichprobe s aus dem Trainingsset aus. (*Bootstrap*)
   b) Lerne den Baum b mit den Daten s an bis die Baumtiefe d erreicht ist.
      i. Wähle eine zufällige Anzahl m an Merkmalen aus.
      ii. Wähle die beste Teilung von Merkmalen. (*Split-point*)
      iii. Erstelle aus den beiden Teilmengen zwei Kinderknoten.
2. Ausgabe das *Ensemble* aus allen Bäumen.

Ziel der Klassifizierung Das angestrebte Ziel mit diesem Modell besteht darin eine binäre Klassifizierung auf dem Nutzerverhalten auszuführen. Dieses soll gleichzeitig die beiden Fragen, ob eine Session etwas kauft und was genau gekauft wird, aus dem *RecSys* Wettbewerb beantworten. Diese Strategie wurde auch in Cohen u. a. (2015) gewählt. Wird ein Artikel aus der Session als einer, der gekauft wird, klassifiziert, so nimmt man an, dass die Session mit einem Kauf endet.

Bsp. Erstellung eines Entscheidungsbaums In Abbildung 4.3 sieht man die wichtigsten Teilschritte, die beim Anlernen eines Beispielbaums ineinander greifen. Dazu wird zuerst ein Datensatz (a) mit den entsprechenden Labels (b) gebraucht. Aus diesem Datensatz wählt man eine Teilmenge für den zu erstellenden Baum aus. Dadurch garantiert man, dass jeder Baum zum größten Teil an einem anderen Abschnitt des Datensatzes angelernt wird. Dieses Vorgehen nennt man in der Statistik auch *Bootstrap* Methode.

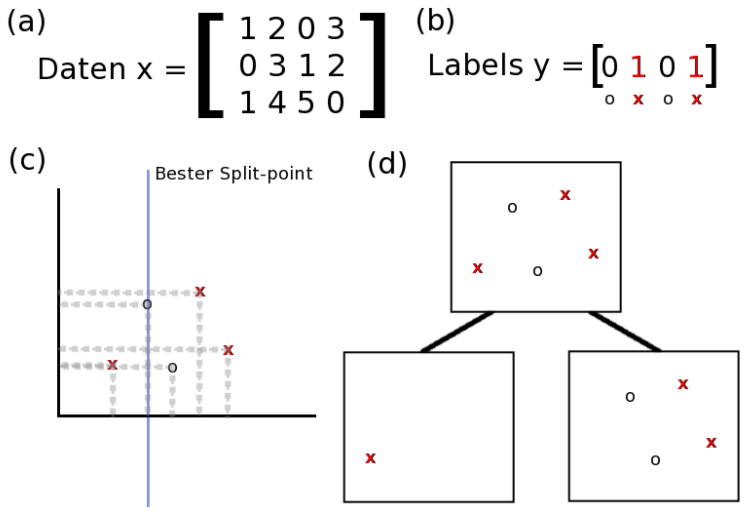

Abbildung 4.3: Beispiel Ablauf beim Erstellen neuer Knoten.

Danach wählt man eine zufällige Anzahl aus allen vorhanden Merkmalen aus (c). Bei dieser Entscheidung wird keine Rücksicht auf Zusammenhänge zwischen den Merkmalen genommen. Der Zufall für sich genommen birgt einen entscheidenden Vorteil für das Verfahren. Dieser mindert die Gefahr, dass die Bedeutung eines bestimmten Merkmals überschätzt wird, wenn dieses Merkmal immer wieder ausgewählt wird Breiman (2001).

Die zufällig gewählten Merkmale in der angeschauten Menge können als Punkte zur Aufteilung in Unterknoten genommen werden (grau gestrichelte Linien), jedoch wird davon nur die beste Aufteilung ausgewählt (blaue Linie). Die Metriken über die man die beste Aufteilung finden kann, berechnen entweder den Gewinn an neuen Informationen (*Information Gain*), die für die Klassifizierung eines Datenpunktes hinzukommen oder die Wahrscheinlichkeit das ein Datenpunkt in dieser Menge richtig klassifiziert wird (*Gini-Koeffizient*) (James u. a., 2013).

Danach werden diese beiden Mengen auf zwei neue Kinderknoten verteilt (d). Je nach dem was für eine Baumtiefe gewählt wurde, werden diese Knoten wieder nach dem selben Verfahren aufgeteilt. Der linke Knoten besitzt in diesem Fall nur noch ein Merkmal,

damit würde bei einer Klassifizierung dieser eine eindeutige Gewichtung zu einem Label benennen können.

Einstellung der Parameter Wie man im beschriebenen Ablauf des Algorithmus in Abschnitt 4.5.1 sieht, gibt es einige Parameter die man bestimmen kann, um die Genauigkeit des Modells zu verbessern. (I) Der erste ist die Anzahl an Bäumen, die man für das jeweilige Problem anlernen will. (II) Der zweite Parameter ist die minimale oder maximale Baumtiefe bis zu der man die Merkmale untereinander aufteilt. Zusätzlich zu der Tiefe kann man auch die (III) Anzahl der zufällig ausgewählten Merkmale einstellen. Diese werden wiederum durch eine Metrik auf neue Knoten aufgeteilt. Dafür wird jeweils die Qualität der möglichen Aufteilungen untereinander verglichen (IV). Jeder dieser einzelnen Parameter muss berücksichtigt und auf den passenden Wert gesetzt werden.

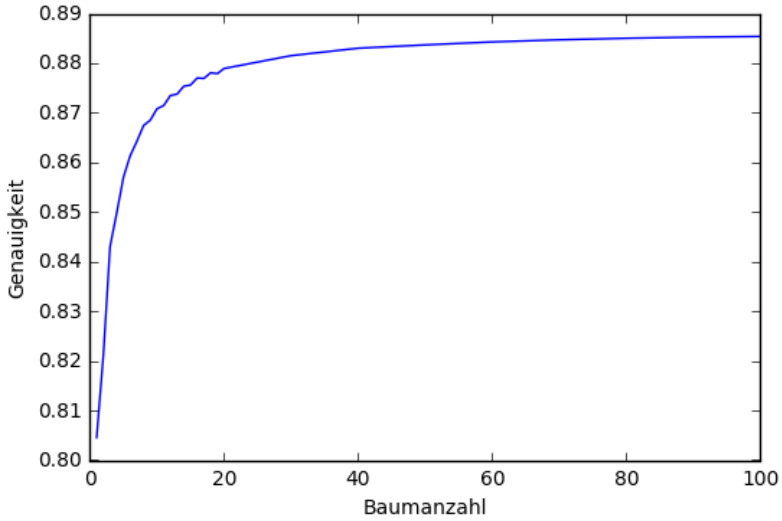

Abbildung 4.4: Genauigkeit des Modells abhängig von der Baumanzahl.

(I) Anzahl der Bäume bestimmen

Zur Bestimmung der passenden Baumanzahl wurden zu Anfang mehrere Testreihen mit unterschiedlichen Werten ausgeführt. In Abbildung 4.4 sieht man die jeweiligen Wertungen für 1 bis 100 Bäume je *RF*. Die höchste Genauigkeit sieht man bei 100 Bäumen,

die bei ca. 88,53% liegt. Die gemessene Zeit[2] für das Anlernen von 40 Bäumen beträgt 1 Minute und 30 Sekunden mit 8 Threads. Durch das parallele Anlernen der einzelnen Bäume kommt man auf eine überschaubare Lernzeit. Die Standardabweichung zwischen 30 und 100 Bäumen liegt bei 0,13% bei einem Mittelwert von 88,41%. Daraus erkennt man dass keine signifikante Steigerung in der Genauigkeit mit mehr als 40 Bäumen erreicht wird, jedoch eine um so längere Lernphase.

Auswertung Die Anzahl an verwendeten Bäumen beeinflusst die Genauigkeit eines Modells, jedoch kann man ab einer bestimmten Größe keine signifikanten Verbesserungen mehr erwarten (Breiman, 2001). Eine genauere Auflistung der Wertung für den *RF* mit 40 Bäumen sieht man in Tabelle 4.3. Die Messung wurde auf dem generierten Testset ausgeführt, damit auch sichergestellt wird, dass man keine Überanpassung ans Trainingsset hat. Der Datensatz wurde dafür in ein Trainings- und Testset im Verhältnis 70/30 aufgeteilt. Je nach Datensatz kann eine andere Aufteilung zwischen den beiden Mengen verwendet werden. Diese kann ein wichtiger Faktor bei der Berechnung der Genauigkeit sein. In diesem Fall wurde eine ähnliche Verteilung wie im *RecSys* Wettbewerb gewählt[3].

	precision	recall	f1-score	count
Kein Kauf	0,88	0,94	0,91	968389
Kauf	0,87	0,78	0,82	542480
avg / total	0,88	0,88	0,88	1510869

Tabelle 4.3: Bewertung des Modells anhand von F1-Score, Preciseion, Recall

Metriken zur Evaluierung Im *RecSys* Wettbewerb ist der *Jaccard-Koeffizient* als Metrik eingesetzt worden, welcher in Abschnitt 2.3 vorgestellt wird. Die bisher verwendete Genauigkeit liefert das gleiche Ergebnis wie der *Jaccard-Koeffizient* bei einer binären Klassifizierung. In Tabelle 4.3 sind zusätzlich die Präzision (engl. *precision*) und Trefferquote (engl. *recall*) für den *F1-Score* berechnet. Die Berechnung der Präzision und Trefferquote wird für jede Klasse über die Gleichung 4.1 umgesetzt.

[2] Die verwendete CPU war Intel(R) Core(TM) i7-3840QM CPU @ 2.80GHz.
[3] Im Wettbewerb wurde ein Trainingsset mit Labels und ein Testset ohne bereitgestellt.

$$\frac{richtige\ Vorhersagen}{richtige\ Vorhersagen\ +\ falsche\ Vorhersagen} \tag{4.1}$$

Die Präzision drückt das Verhältnis zwischen vorhergesagten und den tatsächlichen Labels einer Klasse aus. Bei der Trefferquote wird jedoch für $falsche\ Vorhersagen$ die andere Klasse genommen. Damit wird bei der Trefferquote das Verhältnis zwischen vorhergesagten und den für die angeschaute Klasse nicht relevanten Labels berechnet. Der *F1-Score* ist die Kombination dieser beiden Metriken über den gewichteten Mittelwert (siehe Gleichung 4.2).

$$2 \times \frac{precision \times recall}{precision + recall} \tag{4.2}$$

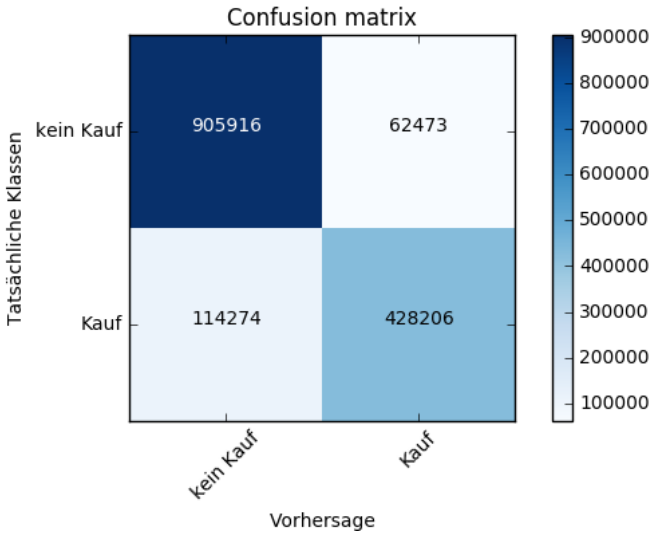

Abbildung 4.5: Wahrheitsmatrix der Vorhersage zu den tatsächlichen Werten.

Wahrheitsmatrix Eine andere Ansicht der berechneten Werte aus der Tabelle 4.3 sieht man in der Wahrheitsmatrix in Abbildung 4.5, wo man die zugrundeliegenden Werte noch mal explizit sehen kann. Die diagonale in der Matrix zeigt die jeweils richtigen Vorhersagen auf. Über eine Wahrheitsmatrix kann man einen schnellen Überblick gewinnen

in welchem Verhältnis jede Klasse in den gemachten Vorhersagen zu den tatsächlichen Werten steht. Beispielsweise ist die Minimierung fehlerhaft klassifizierter Kaufentscheidungen, die am Ende zu keinem Kauf führen im E-Commerce wichtig. Denn so entgeht dem Unternehmen die Chance diesen Kunden passende Artikel vorzuschlagen, die doch zu einem Kauf führen könnten.

(II) Baumtiefe

Die *Scikit-Learn* Bibliothek bietet eine frei-wählbare Begrenzung der Baumtiefe. Die angelernten Entscheidungsbäume erreichen für den Datensatz bis zu eine Tiefe von 50 Knoten, wenn man eine vollständige Aufteilung der zufällig gewählten Merkmale von einer Ebene zur nächsten macht. Die optimale Tiefe wurde durch *Cross-Validierung*[4] bei 40 gefunden. Die Unterschiede in der Qualität der Vorhersagen sind von Wert zu Wert jedoch nur sehr gering. Trotzdem ist es wichtig den optimalen Wert zu finden, so dass man das beste mögliche Modell in vertretbarer Zeit berechnen kann. Es fällt auf, dass auch wenn RF unterschiedliche Parameter anbieten diese jedoch nur marginal eine Verbesserung bringen.

(III) Auswahl der Merkmale

Zur Auswahl stehen einmal die $\sqrt{x}$, der $\log_2(x)$ sowie eine selbst definierte Anzahl y an Merkmalen. Das x in diesem Fall steht für die Gesamtanzahl an Merkmalen. Über *Cross-Validierung* wurde $\sqrt{x}$ als optimale Methode für den Datensatz ermittelt und für den weiteren Verlauf auch so gewählt. Beim ursprünglichen *Bagging* Verfahren würden hier alle Merkmale gewählt werden (Friedman u. a., 2001) (James u. a., 2013).

(IV) Metrik zur Aufteilung

Für diesen Parameter wurde der *Gini-Koeffizient*(Breiman u. a., 1984) als Metrik genommen, da diese am besten mit diesem Datensatz funktioniert.

[4] Das Trainingsset wurde drei mal kopiert und jedes mal wurde aus einem anderen Abschnitt ein Teil der Daten für das Testset genommen.

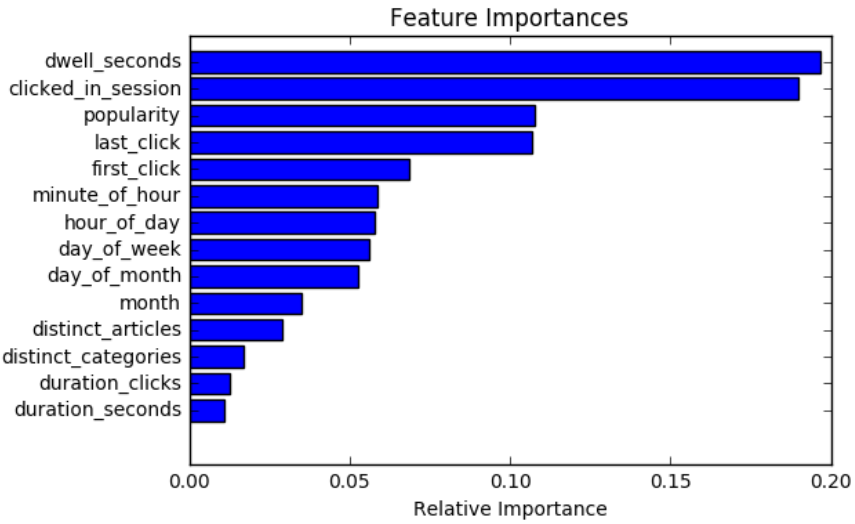

Abbildung 4.6: Bedeutung der einzelnen Merkmale für die Klassifizierung.

4.5.1.1 Bewertung der einzelnen Merkmale

Im Abschnitt zur Exploration der Daten wurden mehrere Hypothesen für den Datensatz aufgestellt. Eine interessante Eigenschaft von Entscheidungsbäumen liegt in der Fähigkeit den Einfluss einzelner Merkmale für das Modell zu berechnen. In Abbildung 4.6 sieht man den entsprechenden Graphen für die erarbeiteten Merkmale mit der Bedeutung[5] für die Klassifizierung.

Interpretation der Sessiondauer Es ist auffällig, dass die totale Sessiondauer in Klicks oder Sekunden keine große Bedeutung für die Klassifizierung spielt. Die Verbindung zwischen der Dauer und dem zu kaufendem Artikel wird damit dementsprechend nicht vollständig abgebildet, da kein direkter Bezug zum Artikel eingefangen wird. Eine weitere Erklärung kann in den vielen Abweichungen in der Sessiondauer gesehen werden, die mit dem Modell nicht erfasst werden. Damit wird auch teilweise gezeigt, dass die Annahmen zur Bedeutung der zeitlichen Abläufe aus der Exploration der Daten so nicht stimmen oder ein Zusammenhang zwischen anderen Merkmalen nicht gefunden wurde, wegen der zufälligen Auswahl dieser.

[5] Die Anzahl in Prozent wie oft ein Merkmal in den Bäumen vorkommt.

Interpretation der Verweildauer Eine wichtige Erkenntnis liefert die Aufenthaltsdauer eines Nutzers, welche zwischen zwei Klicks gemessen wurde und hier entscheidend für die Klassifizierung ist. Gleich danach folgt die Gesamtanzahl an Klicks auf den Artikel in der Session. Die meisten Sessions sind von kurzer Dauer, jedoch ist die Wahrscheinlichkeit eines Kaufs am Ende der längeren Sessions um so höher. Die Verweildauer eines Kunden auf einem Artikel wurde in der Exploration der Daten nicht berücksichtigt, jedoch später bei der Extraktion neuer Merkmale nach dem Vorbild aus Yağci u. a. (2015) hinzugefügt. Darüber hinaus wird hiermit die Frage, die in Abschnitt 2.4.4 zur Qualität von implizitem Feedback aufgestellt wird, beantwortet. Die Popularität eines Artikels in Klicks und Käufen sowie die Verweildauer darauf dominieren damit hier die Vorhersagegenauigkeit.

Die Merkmale, ob ein Artikel zu Anfang oder zum Schluss einer Session angeklickt wurde, in Verbindung mit der Gesamtanzahl an Klicks für diesen bestätigen die Hypothese, dass die meisten Kunden genau wissen was sie kaufen wollen. Zumindest sieht man das anhand dieses Modells.

	Merkmal	Kein Kauf	Kauf
Durchschnitt (Trainingsset)		0.64053021	0.35946979
	duration_seconds	-0.05582399	0.05582399
	duration_clicks	0.00145103	-0.00145103
	distinct_categories	0.0187149	-0.0187149
	distinct_articles	0.00474265	-0.00474265
	month	0.02815872	-0.02815872
	day_of_month	-0.01706299	0.01706299
	day_of_week	-0.02725145	0.02725145
	hour_of_day	-0.06932373	0.06932373
	minute_of_hour	-0.02360196	0.02360196
	first_click	0.02692344	-0.02692344
	last_click	0.03455434	-0.03455434
	popularity	-0.26688538	0.26688538
	clicked_in_session	-0.23083795	0.23083795
	dwell_seconds	-0.06428783	0.06428783
		0	1

Tabelle 4.4: Aufteilung der Merkmale in ihrer Bedeutung zur Klassifizierung eines tatsächlich gekauften Artikels.

4.5.1.2 Vergleich zweier Vorhersagen

In diesem Abschnitt wird ein genauer Blick auf die Bedeutung der Merkmale während einer Vorhersage geworfen. Dafür werden zwei Einträge aus dem Datensatz ausgesucht von denen der eine gekauft wurde und der andere nicht. Die ausgeführte Vorhersage liefert dazu die richtige Klassifizierung. Wie auch in Abbildung 4.6 für das ganze Modell die Bedeutung der einzelnen Merkmale gezeigt wird, kann man auch direkt für einzelne Vorhersagen die Berechnung dahinter ausgeben lassen[6]. In Tabelle 4.4 sieht man die Aufteilung der Vorhersage in die genutzten Merkmale für den tatsächlich gekauften Artikel und in Tabelle 4.5 für den nicht gekauften.

Zusammensetzung der Entscheidung Summiert man alle Werte in einer Spalte kommt man zur Wertung für die jeweilige Klasse. Je höher der Wert am Ende ist, desto wahrscheinlicher ist es, dass der Eintrag zur entsprechenden Klasse gehört.

Entscheidungsbaum als Blackbox Durch diese Art der Untersuchung kann man einen genaueren Blick in die einzelnen Bestandteile einer Vorhersage werfen. Dadurch hat man keine Blackbox, die eine Klassifizierung berechnet, sondern ein transparentes Verfahren was die Gewichtungen der einzelnen Merkmale liefern kann. Anhand dieser kann man genau sehen, welche Merkmale in der Situation den falschen Ausschlag für die Entscheidung gegeben haben.

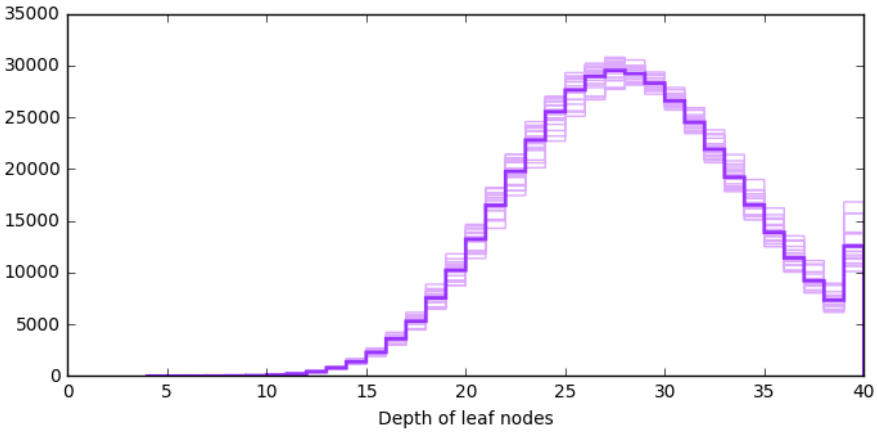

Abbildung 4.7: Blatttiefe über alle Bäume im Random Forest.

[6]Die dafür benötigte Bibliothek: `https://github.com/andosa/treeinterpreter`.

Die vollständige Ausgabe eines Baums eignet sich jedoch nicht für die Überprüfung oder Interpretation eines Modells. Denn bei einem vollständigen Binärbaum mit der Tiefe 20 müssten schon 2^{20} ($\sim$1 Million) Knoten ausgegeben werden. In Abbildung 4.7 sieht man die angetroffenen Blatttiefen aller Bäume des hier eingesetzten Modells[7]. Dabei erkennt man, dass zwischen einer Tiefe von *25* und *30* mehr als *30 000* Knoten vorhanden sind.

Interpretation einer kaufenden Session In Tabelle 4.4 sieht man, dass die Popularität und die Anzahl an Klicks des Artikels für die Vorhersage am meisten beitragen. In der Gesamtübersicht für alle Vorhersagen aus der Abbildung 4.6 ist die Verweildauer am entscheidendsten. Hier sieht man also einen Baum als Beispiel, wo die Verweildauer eine kleinere Bedeutung spielt. Es ist sogar so, dass die Stunde des Tages, an dem auf den Artikel geklickt wurde hier wichtiger als die Verweildauer ist. Diese Varianz in den einzelnen Bäumen wird im Mittelwert über das *Ensemble* entfernt.

	Merkmal	Kein Kauf	Kauf
Durchschnitt (Trainingsset)		0.64053021	0.35946979
	duration_seconds	-0.01529682	0.01529682
	duration_clicks	0.06244913	-0.06244913
	distinct_categories	0.03286368	-0.03286368
	distinct_articles	0.07098494	-0.07098494
	month	-0.00425526	0.00425526
	day_of_month	-0.02220764	0.02220764
	day_of_week	0.02061443	-0.02061443
	hour_of_day	-0.02627606	0.02627606
	minute_of_hour	-0.09233336	0.09233336
	first_click	0.01392769	-0.01392769
	last_click	0.01943669	-0.01943669
	popularity	-0.15997078	0.15997078
	clicked_in_session	0.1240235	-0.1240235
	dwell_seconds	0.13550965	-0.13550965
		0.8	0.2

Tabelle 4.5: Aufteilung der Merkmale in ihrer Bedeutung zur Klassifizierung eines nicht gekauften Artikels.

[7] Die verwendete Bibliothek: `https://github.com/aysent/random-forest-leaf-visualization/` (2016.12.01)

Es wurde öfters auf die Unausgeglichenheit des Datensatzes bei der Datenanalyse hingewiesen. Diese Eigenschaft wird auch in dem Durchschnitt für diesen Teil des Datensatzes wieder entdeckt. In der zweiten Zeile aus Tabelle 4.5 sieht man die jeweiligen Werte aus dem Trainingsset, die darauf deuten, dass mehr Sessions ohne Käufe als mit nur wenigen gekauften Artikeln vorkommen.

Fazit

Nach dem hier erstellten Modell sind die Verweildauer auf einem Artikel sowie die Anzahl an gemachten Klicks von allen verwendeten Merkmalen die, die am häufigsten vorkommen. Ein Teil der gemachten Hypothesen aus der Exploration der Daten konnten hier nicht nachgewiesen werden, was jedoch auf einen Mangel in den verwendeten Merkmalen schließen lässt. Trotzdem liefert das Modell eine Genauigkeit von ca. 88% bei der Vorhersage für das Testset, was mehr als 30% genauer im Vergleich zu einer zufällig gewählten Klassifizierung ist. Diese Wertung konnte zum größten Teil durch die Analyse der Daten und das Extrahieren von passenden Merkmalen erreicht werden, da ein *Random Forest* nur geringfügige Verbesserungen durch das Optimieren von Parametern bietet. Die Extraktion von neuen Merkmalen ist leider durch den Mangel an weiterem Fachwissen stark limitiert.

Ein großer Vorteil eines *Random Forest* ist die Verwendung von *qualitativen* Merkmalen, die im Abschnitt Stetige und Kategoriale Merkmale vorgestellt wurden. Dadurch muss man nicht die einzelnen möglichen Werte eines kategorialen Merkmals enkodieren[8]. Ein weiterer Vorteil liegt in der von Anfang an hohen Genauigkeit von 79%, die bei diesem Datensatz gemessen wurde.

4.5.2 Gradient Boosting

Im vorherigen Abschnitt wurde eine weiterentwickelte *Bagging* Variante mit der *Random Forest* Implementierung angeschaut. In dem nun folgenden Kapitel wird eine *Boosting* Variante aus Friedman (2001) für die Entwicklung eines Ensembles erläutert und mit dem ersten Modell verglichen. Beide *Ensemble* Varianten wurden im *RecSys* Wett-

[8] Bsp. Das Merkmal für das Geschlecht wird dann mit zwei neuen Merkmale enkodiert, die dafür stehen ob eine Person eine Frau oder ein Mann ist.

bewerb erfolgreich eingesetzt, jedoch ohne eine genauere Untersuchung der Gründe für oder gegen ein Verfahren. Um diese beiden trotzdem zu vergleichen wird die gleiche Menge an Merkmalen eingesetzt. Nach Freund u. a. (1996) sollte *Boosting* ein besseres Ergebnis liefern, aber in der Zeit zum Anlernen länger brauchen. *Bagging* und *Boosting* sind nicht auf ein bestimmtes Verfahren beschränkt, jedoch werden zum besseren Vergleich bei beiden Varianten Entscheidungsbäume eingesetzt.

XGBoost und Scikit Learn

Im ersten Versuch wurde die Implementierung von *Scikit-Learn*[9] verwendet, dabei ist zuerst die lange Berechnungszeit aufgefallen. Diese lässt sich dadurch erklären, dass *Boosting* im Gegensatz zu einem *Random Forest* ein sequentielles Gradientenverfahren einsetzt. Aus diesem Grund wird hier die Implementierung aus dem *XGBoost*[10] Projekt (Chen u. Guestrin, 2016) verwendet, die auch eine Schnittstelle zu *Scikit-Learn* anbietet. *XGBoost* wurde erstmals im Wettbewerb zur Berechnung der Eigenschaften vom Higgs Boson [11] eingesetzt. Die Vorteile von *XGBoost* nach den Angaben der Autoren liegen in der hohen Flexibilität bei der Berechnung in den Gebieten der Regression, Klassifikation, Ranking oder einer eigens definierten Zielfunktion[12]. Dazu kommt eine hohe Performanz durch die Implementierung hinzu, die verschiedenste Optimierungen einsetzt, um eine teilweise parallele Berechnung von Bäumen zu ermöglichen[13]. So wie beim vorherigen Verfahren wird auch mit diesem eine binäre Klassifizierung angestrebt, die in diesem Fall mit Regression umgesetzt ist. Der Einfachheit halber werden in diesem Abschnitt nur die wichtigsten Unterschiede zum vorherigen Verfahren aufgezeigt und beschrieben.

Grober Ablauf von Gradient Boosting Nach Friedman u. a. (2001) wird der Algorithmus für *Gradient Boosting* wie folgt zusammengefasst. Der Fokus wird nach jedem Anlernen eines Entscheidungsbaums auf die fehlerhaft klassifizierten Abschnitte gerich-

[9] `http://scikit-learn.org/stable/modules/generated/sklearn.ensemble.GradientBoostingClassifier.html` (2016.12.13)
[10] `https://xgboost.readthedocs.io/en/latest/` (2016.12.13)
[11] Es wurde eine Klassifizierung auf dem Resultat der Zerlegung eines Higgs Boson Partikels ausgeführt. `https://www.kaggle.com/c/higgs-boson` (2016.12.04)
[12] Es ist möglich seine eigenen Funktionen für die Optimierung zu definieren.
[13] Quelle: `https://zhanpengfang.github.io/418home.html` (2016.12.04)

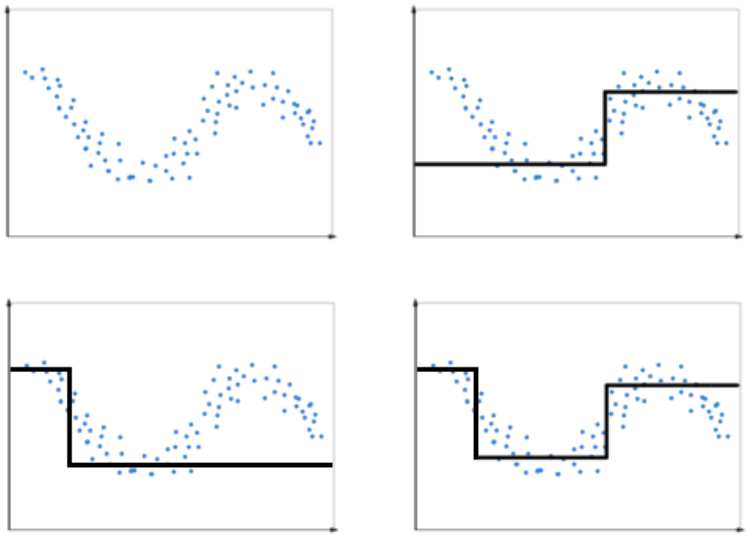

Abbildung 4.8: Beispiel Ablauf beim Erstellen neuer Bäume.

tet. Der nächste Baum in der Sequenz wird darauf trainiert diese Abweichungen zu den tatsächlichen Daten zu minimieren. So verbessert der zweite Baum die Vorhersage des ersten und der dritte, die des zweiten und so weiter. Am Ende findet dann eine Kombination dieser schwachen Klassifizierer, die einen hohen Bias haben, in einem komplexen Modell statt.

In Abbildung 4.8 sieht man dazu ein vereinfachtes Beispiel zur Regression. In dem linken oberen Graphen sieht man den Datensatz für welchen ein Modell gefunden werden soll. Der erste Baum berechnet im rechten oberen Graphen eine Funktion, die die letzteren Extremstellen abbildet und den Werteraum in zwei Partitionen spaltet. Anhand dieses ersten Modells werden die Abweichungen zum eigentlichen Datensatz berechnet. Diese werden dann als Grundlage für ein neues Modell genommen, was wiederum zum unteren linken Graphen führt. Die Kombination dieser beiden Modelle sieht man dann in dem unteren rechten Graphen, welcher dadurch alle Extremstellen vorhersagen kann.

Gradientenverfahren Eine wichtige Komponente in *Gradient Boosting* ist, wie man am Beispiel sehen kann, die schrittweise Verkleinerung der Fehler bzw. der Abweichung zu den tatsächlichen Daten. Nach Runkler (2015) ist das *Gradientenverfahren* eine itera-

tive approximative Optimierungsmethode für differenzierbare Funktionen. Das bedeutet das eine Funktion $y = f(x)$ minimiert wird, indem der Parametervektor $x = x_0$ zufällig initialisiert und dann iterativ für $k = 1, 2, ..., K$ mit der Gleichung 4.3 aktualisiert wird.

$$x^{(k+1)} = x^{(k)} + a^{(k)} d^{(k)} \tag{4.3}$$

Der Term d^k ist der Gradient der Funktion zur Berechnung der Abweichung von f an der Stelle x^k. Jedes Teilmodell schätzt somit den Gradienten der gewählten Verlustfunktion. XGBoost bietet hier die Möglichkeit entweder die implementierten Funktionen zu benutzen oder eine eigene Funktion zu definieren. Darüber hinaus ist es auch möglich eine Abbruchbedingung zu definieren, die aussagt nach wie vielen Berechnungsschritten ohne eine Verbesserung zu erzielen abgebrochen werden soll.

4.5.2.1 Lernrate und Baumanzahl

Im Gegensatz zu einem *Random Forest* bieten sich mit *Gradient Boosting* mehrere Parameter zur Optimierung des Modells an. Davon haben die Lernrate und die Baumanzahl eine besondere Bedeutung in ihrer Kombination Friedman (2001). Die Lernrate bestimmt für jeden Schritt in der Iteration den Anteil zu welchem der angelernte Entscheidungsbaum in das finale Modell einbezogen wird (siehe $a^{(k)}$ aus Gleichung 4.3). Angenommen man erreicht mit 100 Teilmodellen das Optimum für die gegebenen Daten, wo der Bias und die Varianz ausgeglichen sind, also keine Unter- oder Überanpassung an die Daten vorhanden ist. Dann werden bei einer Lernrate von 1 alle 100 Teilmodelle komplett übernommen. Nimmt man jedoch eine Lernrate von 0.1, so müssten ca. 1000 Teilmodelle berechnet werden, damit man wieder zu diesem Optimum kommt. Dadurch kann man auch die Ausführungszeit steuern, indem man die Lernrate anhebt und die Baumanzahl verkleinert, da so weniger berechnet werden muss. Das verschlechtert die Vorhersagekraft des Modells für das Trainingsset, jedoch kann man mit weniger Ressourcen schneller ein Modell anlernen.

In Abbildung 4.9 sieht man zwei Versuchsreihen, bei denen jeweils 100 und 1000 Bäume mit unterschiedlichen Lernraten verwendet wurden. Dabei fällt auf: Je höher die Lernrate, desto höher auch die Genauigkeit in den Trainingsdaten. Das gleiche gilt auch für die Baumanzahl. Der Unterschied in der Bewertung für das Trainingsset und dem

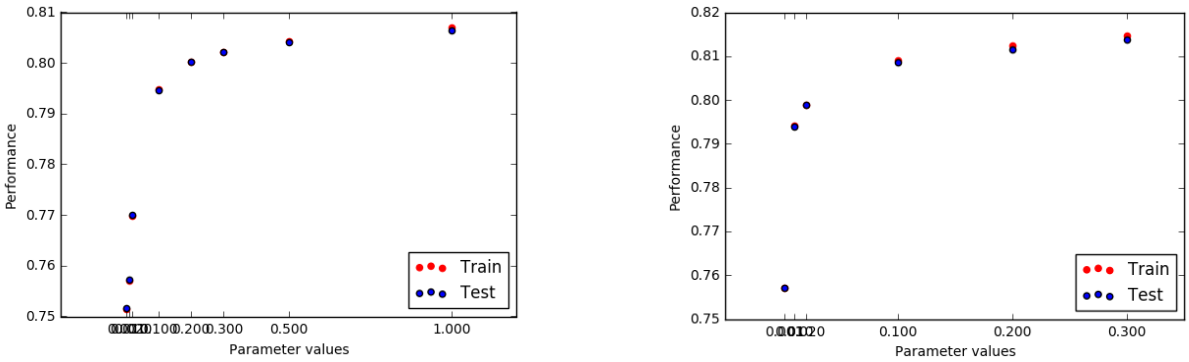

Abbildung 4.9: Genauigkeit des Modells mit 100 (links) und mit 1000 (rechts) Bäumen. Die verwendeten Lernraten: 0.001, 0.01, 0.02, 0.1, 0.2, 0.3, 0.5, 1.0

Testset für den vorliegenden Datensatz ist hierbei sehr gering. Jedes Antrainieren eines Modells mit 1000 Bäumen dauerte im Schnitt 19 Minuten, bei 100 Bäumen sind es ca. 2 Minuten.

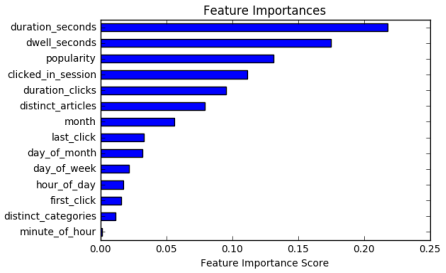

Abbildung 4.10: Die Tiefe 3 resultiert in einer Genauigkeit 0.8064.

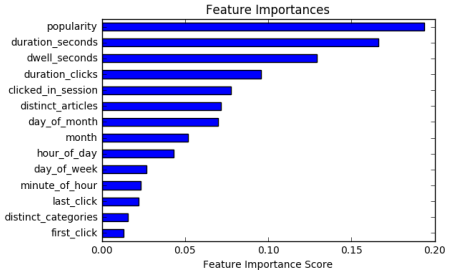

Abbildung 4.11: Die Tiefe 6 resultiert in einer Genauigkeit 0.8155.

4.5.2.2 Baumtiefe

Die Baumtiefe bei *Gradient Boosting* fällt im Vergleich zum *Random Forest* um einiges flacher aus, da *Boosting* den Bias zur Minimierung der Fehlerrate nutzt. Die tiefen Bäume beim *Random Forest* begünstigen dafür eine hohe Varianz, weil viele Eigenschaften der Trainingsdaten einbezogen werden, um diese dazu zu nutzen die Fehlerrate durch den Durchschnitt zu verringern. Wählt man also einen zu kleinen Wert für die Tiefe, werden weniger Zusammenhänge unter den Merkmalen in einem Teilbaum erfasst. In

den einzelnen Abbildungen von 4.10 bis 4.12 sieht man jeweils die aufsummierten Einflüsse von Merkmalen unter verschiedenen Baumtiefen.

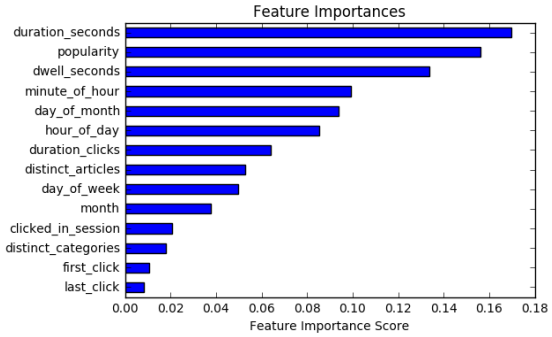

Abbildung 4.12: Die Tiefe 15 resultiert in einer Genauigkeit 0.8603.

Wie auch beim *Random Forest* beeinflussen die zeitlichen Merkmale die Vorhersage am geringsten. Wieder sind die Aufenthaltsdauer und die Popularität eines Artikels ganz weit oben. Zu diesen kommt nun auch die Sessiondauer in Sekunden hinzu, welche bei einer geringen und großen Tiefe an erster Stelle auftaucht. Man erkennt wie die Baumtiefe die Bedeutung eines Merkmals verändern kann.

Visualisierung von Teilbäumen Für die Visualisierung sind die Entscheidungsbäume aus *Gradient Boosting* einfacher darzustellen, dafür bieten diese einzeln angeschaut eine schwache Aussagekraft, da diese den Fehler des vorherigen Baums minimieren und erst in ihrer Summe eine gute Vorhersage liefern. Es ist aber hier auch ersichtlich welche Bedeutung ein Merkmal für das *Ensemble* einnimmt. In Abbildung 4.13 sieht man den letzten Baum aus einem Beispiel *Ensemble*. Je näher man an die Wurzel des Baums geht, desto mehr Gewicht besitzt die gemachte Entscheidung. Außerdem kann man sehr gut alle Interaktionen zwischen den Merkmalen sehen, was in einem Graphen wie Abbildung 4.12 nicht sichtbar wird.

Komponente des Zufalls Beim *Random Forest* existieren zwei Komponenten des Zufalls, die entweder bei der Auswahl eines Teils der Daten (Bootstrap) oder der Merkmale zum Greifen kommen. Die gleiche Optimierung kann man auch zur Minimierung der Berechnungszeit mit XGBoost anwenden (Friedman u. a., 2001). Der passende Wert ist

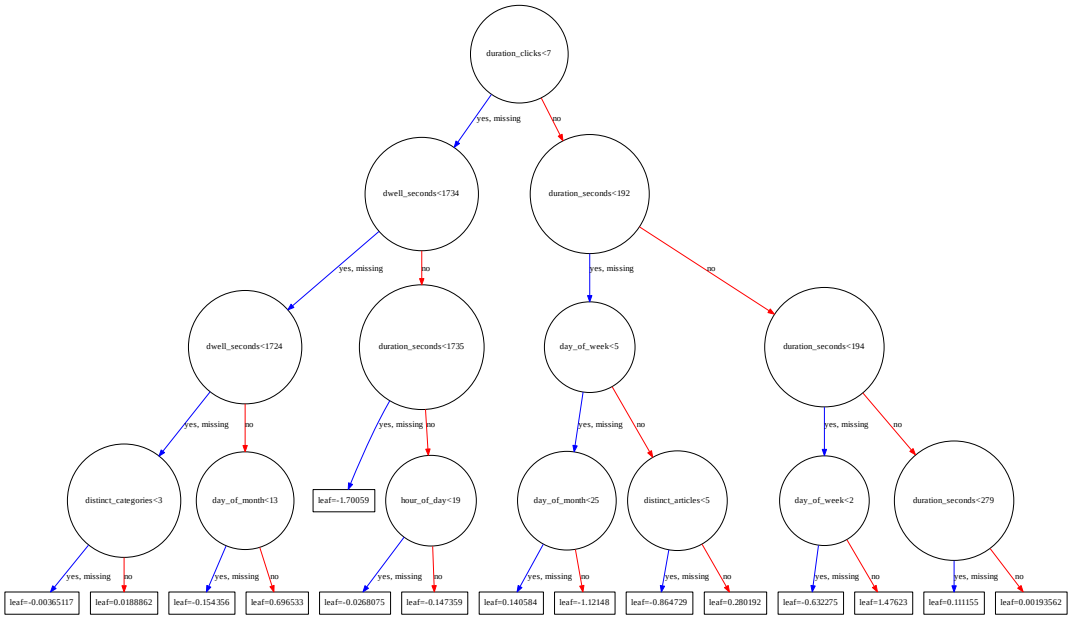

Abbildung 4.13: Der letzte Entscheidungsbaum mit einer Tiefe von 4.

idealerweise über *Cross-Validierung* zu finden, jedoch kann die vollständige Berechnung sehr lange dauern.

Weitere Parameter zur Regulierung des Modells

Gradient Boosting mit XGBoost bietet viele weitere Parameter[14] zur Erstellung eines allgemeineren Modells, wie z.B. zur Reduktion der Varianz an (Chen u. Guestrin, 2016). Damit können alle Aspekte bei der Berechnung der Entscheidungsbäume und des Gradientenverfahrens verändert werden. Diese wurden bei den finalen Experimenten herausgelassen, da die Differenz zwischen den Fehlern vom Trainingsset und dem Testset sehr gering ausgefallen sind. Zudem dauerte die Anlernphase für den Datensatz am Schluss je nach Einstellung zwischen einer bis drei Stunden für ein Modell, was die weitere Optimierung auf einem herkömmlichen Computer erschwerte. Ein potentieller Ansatz um dieses Problem zu umgehen könnte die zeitliche Zerlegung des Datensatzes in die einzelnen Monate nach dem Vorbild aus Yağci u. a. (2015) bringen.

[14] Übersicht: `https://xgboost.readthedocs.io/en/latest/parameter.html#parameters-for-tree-booster`

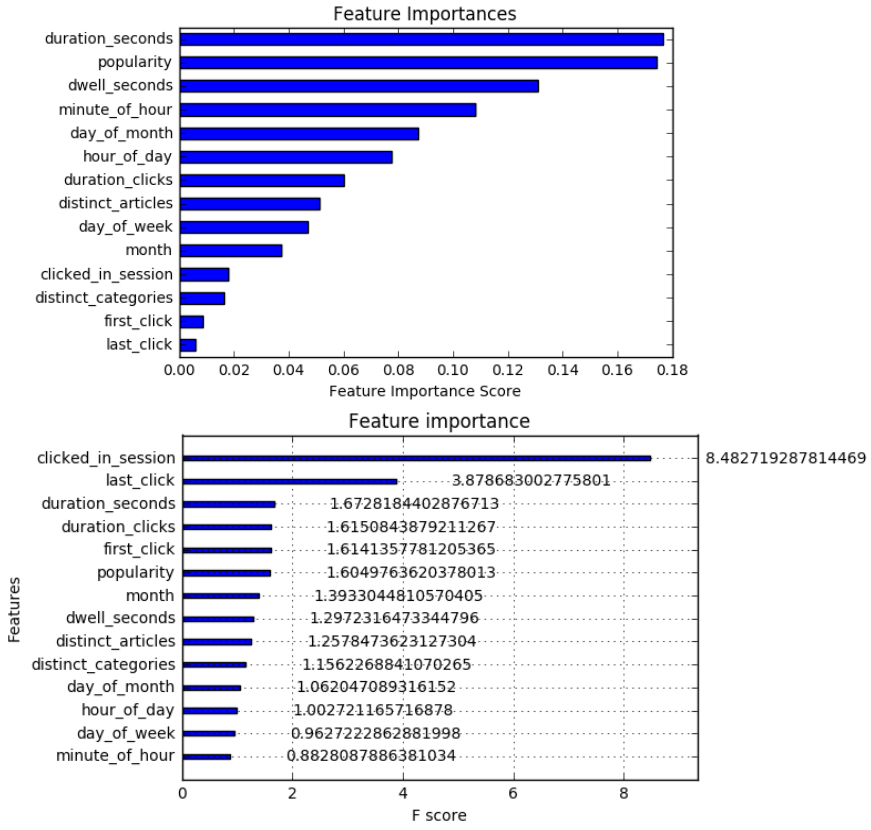

Abbildung 4.14: Vorkommen von Merkmalen in den Bäumen (oben) und der *Information Gain* (unten).

4.5.2.3 Interpretation der Merkmale in Anzahl und Genauigkeit

In Abbildung 4.14 sieht man zwei Graphen aus dem finalen Modell gegenüber gestellt. Der obere Graph zeigt die Wichtigkeit der Merkmale an der Häufigkeit ihres Auftauchens in den Bäumen. Der untere Graph wiederum zeigt welches Merkmal den meisten *Information Gain* (Genauigkeit) in einem Pfad gebracht hat[15]. Dabei fällt auf, dass auch wenn die Dauer einer Session und die Beliebtheit am häufigsten in den Bäumen vorkommen, diese nicht den größten Beitrag zur Genauigkeit leisten. Die Anzahl an Klicks und der zuletzt gemachte Klick für sich gesehen tragen etwas weniger als die Hälfte zur gesamten Genauigkeit bei. In Tabelle 4.6 sieht man die finale Berechnung der Genauigkeit für dieses Modell.

[15] Quelle. `https://xgboost.readthedocs.io/en/latest/R-package/discoverYourData.html` (2016.12.13)

	precision	recall	f1-score	count
Kein Kauf	0,89	0,91	0,90	968389
Kauf	0,84	0,81	0,82	542480
avg / total	0,87	0,87	0,87	1510869

Tabelle 4.6: Bewertung des Modells anhand von F1-Score, Preciseion, Recall

4.6 Fazit

In diesem Kapitel wurde die Vorgehensweise und die Methodik für die vorliegende Problemstellung anhand zweier Modelle erläutert und gezeigt. Eine der wichtigsten Erkenntnisse, die sich in diesem Kapitel schon früh herausstellte ist die Anpassung früherer Schritte anhand von neu gefundener Informationen im *KDD* Prozess. Während der Erstellung eines Modells haben sich z.B. neue Aspekte des Datensatzes offenbart, die zu neuen Merkmalen, die man wiederum im finalen Modell verwenden konnte, führten.

Es waren nicht immer alle Zusammenhänge, die dazu beigetragen haben, dass bestimmte Merkmale in ihrer Bedeutung einen größeren Stellenwert bekommen, klar. Die Interpretierbarkeit des finalen Modells ist durch die Verwendung von Entscheidungsbäumen an bestimmten Punkten hilfreich, jedoch muss hinterfragt werden inwieweit sich diese Ergebnisse auf Maßnahmen, die darauf basierend getroffen werden übertragen lassen. Je nachdem was für Merkmale in die finale Vorhersage geflossen sind wurde die Genauigkeit der Vorhersagen schlechter oder besser. Die Entwicklung der richtigen Merkmale ist damit eine Kernaufgabe, die durch einen Fachexperten übernommen werden sollte.

In Unterabschnitt 4.5.2 wurde die *Gradient Boosting* Implementierung mit *XGBoost* angeschaut. Dafür wurden die gleichen Merkmale wie für den *Random Forest* verwendet. Die ersten Unterschiede gestalteten sich zum einen in der längeren Anlernphase und zum anderen in den komplexeren Einstelllungen der Parameter, bei denen man um einiges mehr optimieren muss und kann. Die Genauigkeit des Modells auf den Testdaten ist stark davon abhängig, welche Kombination von Werten genommen wurde. Diese entscheidet im Rückschluss die Auswahl der Merkmale und somit die erkannten Zusammenhänge. Damit ist die Benutzung von *Cross-Validierung* hier besonders wichtig, was jedoch nicht immer möglich ist, da die Menge der Daten und die sequentielle Natur

des Verfahrens limitierend sind. Es konnte am Schluss eine ähnliche Genauigkeit wie bei *Random Forest* erreicht werden, jedoch mit einem höheren Zeitaufwand.

Die *XGBoost* Implementierung ist ein sehr flexibles Werkzeug, deswegen verwundert es nicht das dieses heutzutage in den meisten Wettbewerben zum Einsatz kommt. Trotz der relativ langen Anlernphase kann man mit der richtigen Einstellung der Parameter gute Ergebnisse erzielen. Im Gegensatz zum *Random Forest* bietet sich hier viel Raum zur Optimierung an. Dafür bietet ein *Random Forest* einen guten Start für ein erstes Modell, um damit erste Merkmale des Datensatzes zu bewerten.

Bei der Bedeutung der Merkmale gibt es keinen zu großen Unterschied zwischen den beiden Verfahren, außer dass die Sessiondauer eines Kunden den höchsten Ausschlag gibt. So sind die Aufenthaltsdauer eines Kunden in Verbindung mit der Popularität eines Artikels die wichtigsten Merkmale, die dazu Beitragen ob ein Kunde etwas kauft oder nicht. Der zuletzt gemachte Klick aus der Session ist über *Gradient Boosting* im Zusammenhang mit der Anzahl an gemachten Klicks für den Artikel ein sehr markante Feature-Kombination, was erst durch die Untersuchung des Anteils zur Genauigkeit sichtbar wurde (siehe Abbildung 4.14).

5 Fazit & Ausblick

Fazit

Implizites Feedback

In dieser Arbeit wurde ein Einblick in die Besonderheiten vom impliziten Feedback und die Bedeutung dieses für die Interpretation und Verwendung in einem *Data-Mining* Prozess gegeben. Es bieten sich viele Möglichkeiten an aus dem Nutzerverhalten einer E-Commerce Plattform wichtige Merkmale zu extrahieren. Dazu benötigt man jedoch Fachwissen, um bekannte Zusammenhänge zu finden und mit einfließen zu lassen. Trotzdem liefern schon kleine Mengen an Merkmalen eine gute Grundlage für das Anlernen eines *Data-Minig* Verfahrens. Die Ergebnisse daraus zeigen erste Zusammenhänge zwischen den einzelnen Merkmalen auf.

Data Mining Wettbewerbe

In den letzten Jahren wird das Thema *Machine Learning* immer häufiger diskutiert, da viele große Unternehmen in die Weiterentwicklung investieren (Parloff, 2016). Durch die Organisation unterschiedlicher Wettbewerbe wie der *RecSys Challenge* oder einer Plattform wie *Kaggle*, eröffnen sich neue Möglichkeiten mit realen Datensätzen bekannte Verfahren zu vergleichen und weiter zu entwickeln. Damit wird auch der öffentliche Austausch von Erkenntnissen gefördert, welcher zu neuen Lösungsansätzen führt wie dem *XGBoost* Projekt. Der *Open Source* Gedanke hat die Entwicklung von z.B. der *Scikit-Learn* Bibliothek begünstigt, was wiederum den Einstieg in die verschiedenen Techniken und Verfahren leichter gestaltet.

Modellierung mit Entscheidungsbäumen

Die eingesetzten Verfahren haben gezeigt, dass man mit schon einer kleinen Anzahl an Merkmalen aus dem Datensatz ein brauchbares Modell anlernen kann. Dafür eigenen sich *Ensembles* in Form von *Random Forest* und *Gradient Boosting*. Ein *Random Forest* liefert mit einer passenden Selektion von Merkmalen in kurzer Zeit gute Ergebnisse.

Damit eignen sie sich gut für den Prozess der Evaluieren neuer Merkmale, da man den Einfluss dieser schnell überblicken kann. Weitere Verbesserungen in der Genauigkeit können wegen des Mangels an Parametern, aber meistens nur über *Feature Engineering* erreicht werden.

Die *Gradient Boosting* Implementierung mit *XGBoost* liefert vergleichbare Ergebnisse, die jedoch erst nach einer aufwendigen Optimierung der einzelnen Parameter sichtbar werden. *XGBoost* bietet viele Verbesserungen gegenüber anderen Implementierungen, wie die teilweise Parallelisierung beim Erstellen eines Modells. Trotzdem ist die Berechnung sehr zeitaufwendig und verbraucht viele Ressourcen. Dafür kann man aber alle Aspekte des Verfahrens nach seinen Wünschen einstellen.

Ausblick

Erstellung von Empfehlungen

Es wurden die wichtigen Aspekte und Konzepte aus den Themengebieten der Empfehlungssysteme und dem Data-Mining angeschaut. Damit wurde dann ein Modell für das zugrundeliegende Nutzerverhalten erstellt. Allerdings wurden dann keine Empfehlungen anhand dieses Modells berechnet. Damit fehlt ein entscheidender Schritt zu einem Empfehlungssystem. Die Auswertung der Daten hat gezeigt, dass die meisten kaufenden Kunden wissen was sie kaufen wollen. Der Rest der Kunden, der nichts kauft und 95% des Datenverkehrs ausmacht, könnte jedoch anhand dieser eine Empfehlung für Artikel bekommen um einen Kauf anzuregen. Es stellt sich die Frage, wie kann man das erstellte Modell mit einem Empfehlungssystem verbinden?

Veränderung der Vorlieben

Der Aspekt der zeitlichen Veränderung von Vorlieben wurde aus der Arbeit komplett herausgelassen. Trotzdem existieren schon anfängliche Arbeiten, die diesen aufgreifen und das statische Modell eines Datensatzes durch ein inkrementelles Verfahren erweitern, jedoch nicht die Erkennung einer Veränderung untersuchen. Einige wichtige Fragen bezüglich dieses Themas könnten lauten: "*Wie viel Zeit muss vergehen bis man eine*

Veränderung an den Vorlieben bemerken kann?" oder "*Wie viele und was für welche Merkmale muss man untersuchen, um eine Veränderung festzustellen?*"

Online Datenströme

Verfahren die *online* arbeiten während das System aktiv genutzt wird müssen eine große Anzahl an Informationen schnell verarbeiten. Das darunter liegende Modell muss dem entsprechend immer wieder angepasst werden. Die Daten kommen in Form eines Datenstroms rein und sind deswegen nicht statisch und enthalten potentiell viele Fehler. Diese Vorgehensweisen beim Verarbeiten sind näher an der Realität dran, als die Berechnung eines statischen Modells im Umgang mit den eintreffenden Informationen (Beel u. a., 2013). Diese Ansätze müssen jedoch auch Probleme in der Verarbeitung großer Datenströme berücksichtigen. Der in dieser Arbeit verwendete Datensatz eignet sich besonders gut für solche Ansätze, da dieser im Grunde eine Sequenz von Interaktionen mit einem Onlineshop darstellt. Die Evaluierung unter solchen Voraussetzungen liefert meistens andere Ergebnisse (Beel u. a., 2013), die damit aber auch näher an der Realität sind.

Interaktive Exploration

Die Visualisierung und Exploration des Datensatzes waren entscheidende erste Schritte bei der Analyse, da man sich mit den Daten vertraut machen konnte. Durch die Erkenntnisse daraus konnten wichtige Entscheidungen in der Entwicklung von Merkmalen zu den Daten getroffen werden. Diese wiederum führten zu einem besseren Modell. Welche Aspekte im Datensatz besonders wichtig sind kann man nur durch die richtige Auswahl von Merkmalen und einer passenden Visualisierung durchführen. Dazu muss man jedoch mehrere Varianten ausprobieren, da nicht alle Darstellungsformen sich für die einzelnen Aspekte eignen. Durch die anfängliche Unwissenheit über die Daten kommt man oft zu keinem guten Ergebnis, weil die Daten oder die Dimension des Merkmalraums zu groß ist. Eine intelligentere Selektion der Darstellung könnte hier Abhilfe schaffen. Hierfür ist die folgende Frage von Interesse: "*Welche Arten der Visualisierung eignen sich für einen Datensatz bestehend aus dem Nutzerverhalten?*"

Literaturverzeichnis

[Amatriain 2013] Amatriain, Xavier: Mining large streams of user data for personalized recommendations. In: *ACM SIGKDD Explorations Newsletter* 14 (2013), Nr. 2, S. 37–48 9, 37, 41

[Amatriain 2014] Amatriain, Xavier: The Recommender Problem Revisited. In: *Proceedings of the 8th ACM Conference on Recommender Systems*. New York, NY, USA : ACM, 2014 (RecSys '14). – ISBN 978–1–4503–2668–1, 397–398 7

[Amatriain u. a. 2011] Amatriain, Xavier ; Jaimes, Alejandro ; Oliver, Nuria ; Pujol, Josep M.: Data mining methods for recommender systems. In: *Recommender Systems Handbook*. Springer, 2011, S. 39–71 11, 26

[Beel u. a. 2013] Beel, Joeran ; Genzmehr, Marcel ; Langer, Stefan ; Nürnberger, Andreas ; Gipp, Bela: A Comparative Analysis of Offline and Online Evaluations and Discussion of Research Paper Recommender System Evaluation. In: *Proceedings of the International Workshop on Reproducibility and Replication in Recommender Systems Evaluation*. New York, NY, USA : ACM, 2013 (RepSys '13). – ISBN 978–1–4503–2465–6, 7–14 8, 79

[Ben-Shimon u. a. 2015] Ben-Shimon, David ; Tsikinovsky, Alexander ; Friedmann, Michael ; Shapira, Bracha ; Rokach, Lior ; Hoerle, Johannes: RecSys Challenge 2015 and the YOOCHOOSE Dataset. In: *Proceedings of the 9th ACM Conference on Recommender Systems*. New York, NY, USA : ACM, 2015 (RecSys '15). – ISBN 978–1–4503–3692–5, 357–358 6, 19, 20

[Blom u. Monk 2003] Blom, Jan O. ; Monk, Andrew F.: Theory of Personalization of Appearance: Why Users Personalize Their Pcs and Mobile Phones. In: *Hum.-Comput. Interact.* 18 (2003), September, Nr. 3 5

[Bollen u. a. 2010] Bollen, Dirk ; Knijnenburg, Bart P. ; Willemsen, Martijn C. ; Graus, Mark: Understanding Choice Overload in Recommender Systems. In: *Proceedings of the Fourth ACM Conference on Recommender Systems*. New York, NY, USA : ACM, 2010 (RecSys '10). – ISBN 978–1–60558–906–0, 63–70 1

[Boriah u. a. 2008] Boriah, Shyam ; Chandola, Varun ; Kumar, Vipin: Similarity measures for categorical data: A comparative evaluation. In: *red* 30 (2008), Nr. 2, S. 3 10

[Breiman 1996] Breiman, Leo: Bagging predictors. In: *Machine learning* 24 (1996), Nr. 2, S. 123–140 37

[Breiman 2001] Breiman, Leo: Random forests. In: *Machine learning* 45 (2001), Nr. 1, S. 5–32 40, 52, 57, 58, 60

[Breiman u. a. 1984] Breiman, Leo ; Friedman, Jerome ; Stone, Charles J. ; Olshen, Richard A.: *Classification and regression trees*. CRC press, 1984 62

[Canny 2002] Canny, John: Collaborative Filtering with Privacy via Factor Analysis. In: *Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. New York, NY, USA : ACM, 2002 (SIGIR '02). – ISBN 1–58113–561–0, 238–245 6

[Chen u. Guestrin 2016] Chen, Tianqi ; Guestrin, Carlos: XGBoost: A Scalable Tree Boosting System. In: *CoRR* abs/1603.02754 (2016). http://arxiv.org/abs/1603.02754 68, 73

[Cohen u. a. 2015] Cohen, Nadav ; Gerzi, Adi ; Ben-Shimon, David ; Shapira, Bracha ; Rokach, Lior ; Friedmann, Michael: In-House Solution for the RecSys Challenge 2015. In: *Proceedings of the 2015 International ACM Recommender Systems Challenge*. New York, NY, USA : ACM, 2015 (RecSys '15 Challenge). – ISBN 978–1–4503–3665–9, 10:1–10:4 40, 45, 57

[Davis u. Goadrich 2006] Davis, Jesse ; Goadrich, Mark: The Relationship Between Precision-Recall and ROC Curves. In: *Proceedings of the 23rd International Conference on Machine Learning*. New York, NY, USA : ACM, 2006 (ICML '06). – ISBN 1–59593–383–2, 233–240 24

[Deshpande u. Karypis 2004] Deshpande, Mukund ; Karypis, George: Item-based top-N Recommendation Algorithms. In: *ACM Trans. Inf. Syst.* 22 (2004), Januar, Nr. 1, 143–177. http://dx.doi.org/10.1145/963770.963776. – DOI 10.1145/963770.963776. – ISSN 1046–8188 27

[Dietterich 2000] Dietterich, Thomas G.: Ensemble methods in machine learning. In: *International workshop on multiple classifier systems* Springer, 2000, S. 1–15 37

[Ekstrand u. a. 2011] Ekstrand, Michael D. ; Riedl, John T. ; Konstan, Joseph A.: Collaborative filtering recommender systems. In: *Foundations and Trends in Human-Computer Interaction* 4 (2011), Nr. 2, S. 81–173 1, 7, 24, 27

[Fayyad u. a. 1996] Fayyad, Usama M. ; Piatetsky-Shapiro, Gregory ; Smyth, Padhraic: Advances in Knowledge Discovery and Data Mining. Version: 1996. http://dl.acm.org/citation.cfm?id=257938.257942. Menlo Park, CA, USA : American Association for Artificial Intelligence, 1996. – ISBN 0–262–56097–6, Kapitel From Data Mining to Knowledge Discovery: An Overview, 1–34 3, 42, 51

[Freund u. a. 1996] Freund, Yoav ; Schapire, Robert E. u. a.: Experiments with a new boosting algorithm. In: *Icml* Bd. 96, 1996, S. 148–156 37, 68

[Friedman u. a. 2001] Friedman, Jerome ; Hastie, Trevor ; Tibshirani, Robert: *The elements of statistical learning*. Bd. 1. Springer series in statistics Springer, Berlin, 2001 37, 57, 62, 68, 72

[Friedman 2001] Friedman, Jerome H.: Greedy function approximation: a gradient boosting machine. In: *Annals of statistics* (2001), S. 1189–1232 35, 67, 70

[Goy u. a. 2007] In: Goy, Anna ; Ardissono, Liliana ; Petrone, Giovanna: *Personalization in E-Commerce Applications*. Berlin, Heidelberg : Springer Berlin Heidelberg, 2007. – ISBN 978–3–540–72079–9, 485–520 5

[Harper u. Konstan 2015] Harper, F. M. ; Konstan, Joseph A.: The MovieLens Datasets: History and Context. In: *ACM Trans. Interact. Intell. Syst.* 5 (2015), Dezember, Nr. 4, 19:1–19:19. http://dx.doi.org/10.1145/2827872. – DOI 10.1145/2827872. – ISSN 2160–6455 20

[Hitt u. Anderson 2007] Hitt, Michael A. ; Anderson, Chris: *The Long Tail: Why the Future of Business Is Selling Less of More*. http://www.longtail.com/about.html. Version: 2007 49

[Hu u. a. 2008] Hu, Yifan ; Koren, Yehuda ; Volinsky, Chris: Collaborative Filtering for Implicit Feedback Datasets. In: *Proceedings of the 2008 Eighth IEEE International Conference on Data Mining*. Washington, DC, USA : IEEE Computer Society, 2008 (ICDM '08). – ISBN 978–0–7695–3502–9, 263–272 21, 22, 30

[James u. a. 2013] James, Gareth ; Witten, Daniela ; Hastie, Trevor ; Tibshirani, Robert: *An introduction to statistical learning*. Bd. 6. Springer, 2013 8, 10, 14, 35, 38, 58, 62

[Koren 2008] Koren, Yehuda: Factorization meets the neighborhood: a multifaceted collaborative filtering model. In: *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining* ACM, 2008, S. 426–434 11, 27

[Koren u. a. 2009] Koren, Yehuda ; Bell, Robert ; Volinsky, Chris: Matrix factorization techniques for recommender systems. In: *Computer* (2009), Nr. 8, S. 30–37 28, 30

[Lempel 2012] Lempel, Ronny: Recommendation Challenges in Web Media Settings. In: *Proceedings of the Sixth ACM Conference on Recommender Systems*. New York, NY, USA : ACM, 2012 (RecSys '12). – ISBN 978–1–4503–1270–7, 205–206 7

[Leskovec u. a. 2014] Leskovec, Jure ; Rajaraman, Anand ; Ullman, Jeffrey D.: *Mining of massive datasets*. Cambridge University Press, 2014 7

[Linden u. a. 2003] Linden, Greg ; Smith, Brent ; York, Jeremy: Amazon.Com Recommendations: Item-to-Item Collaborative Filtering. In: *IEEE Internet Computing* 7 (2003), Januar, Nr. 1, 76–80. http://dx.doi.org/10.1109/MIC.2003.1167344. – DOI 10.1109/MIC.2003.1167344. – ISSN 1089–7801 7

[Lipton 2016] Lipton, Z. C.: The Mythos of Model Interpretability. In: *ArXiv e-prints* (2016), Juni 38

[Marshall 2006] Marshall, Matt: Aggregate Knowledge raises $5M from Kleiner, on a roll. In: *Venturebeat* (2006). http://venturebeat.com/2006/12/10/aggregate-knowledge-raises-5m-from-kleiner-on-a-roll/ 7

[McKinney 2012] McKinney, Wes: *Python for data analysis: Data wrangling with Pandas, NumPy, and IPython*. " O'Reilly Media, Inc.", 2012 15, 17

[Mobasher u. a. 2007] Mobasher, Bamshad ; Burke, Robin ; Bhaumik, Runa ; Williams, Chad: Toward Trustworthy Recommender Systems: An Analysis of Attack Models and Algorithm Robustness. In: *ACM Trans. Internet Technol.* 7 (2007), Oktober, Nr. 4. http://dx.doi.org/10.1145/1278366.1278372. – DOI 10.1145/1278366.1278372. – ISSN 1533–5399 23

[Murphy 2012] Murphy, Kevin P.: *Machine learning: a probabilistic perspective*. MIT press, 2012 7, 11, 13, 36, 37, 40

[Narayanan u. Shmatikov 2006] Narayanan, Arvind ; Shmatikov, Vitaly: How To Break Anonymity of the Netflix Prize Dataset. In: *CoRR* abs/cs/0610105 (2006). http://arxiv.org/abs/cs/0610105 6

[Netflix 2009] Netflix: Netflix Prize Data Set. (2009). http://archive.ics.uci.edu/ml/datasets/Netflix+Prize 20

[Nisbet u. a. 2009] Nisbet, Robert ; Miner, Gary ; Elder IV, John: *Handbook of statistical analysis and data mining applications*. Academic Press, 2009 56

[Paraschakis 2016] Paraschakis, Dimitris: Recommender Systems from an Industrial and Ethical Perspective. In: *Proceedings of the 10th ACM Conference on Recommender Systems*. New York, NY, USA : ACM, 2016 (RecSys '16). – ISBN 978–1–4503–4035–9, 463–466 6

[Pariser 2011] Pariser, Eli: *The filter bubble: What the Internet is hiding from you*. Penguin UK, 2011 32

[Parloff 2016] Parloff, Roger: *AI Partnership Launched by Amazon, Facebook, Google, IBM, and Microsoft*, 2016 (accessed 2016.12.13). http://fortune.com/2016/09/28/ai-partnership-facebook-google-amazon/ 77

[Paterek 2007] Paterek, Arkadiusz: Improving regularized singular value decomposition for collaborative filtering. In: *Proceedings of KDD cup and workshop* Bd. 2007, 2007, S. 5–8 10, 29

[Pilászy u. Tikk 2009] Pilászy, István ; Tikk, Domonkos: Recommending New Movies: Even a Few Ratings Are More Valuable Than Metadata. In: *Proceedings of the Third ACM Conference on Recommender Systems*. New York, NY, USA : ACM, 2009 (RecSys '09). – ISBN 978–1–60558–435–5, 93–100 24

[Rendle 2010] Rendle, Steffen: Factorization machines. In: *2010 IEEE International Conference on Data Mining* IEEE, 2010, S. 995–1000 28, 33

[Rendle u. a. 2009] Rendle, Steffen ; Freudenthaler, Christoph ; Gantner, Zeno ; Schmidt-Thieme, Lars: BPR: Bayesian Personalized Ranking from Implicit Feedback. In: *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence*. Arlington, Virginia, United States : AUAI Press, 2009 (UAI '09). – ISBN 978–0–9749039–5–8, 452–461 11, 20, 22, 31

[Riecken 2000] Riecken, Doug: Introduction: Personalized Views of Personalization. In: *Commun. ACM* 43 (2000), August, Nr. 8, 26–28. http://dx.doi.org/10.1145/345124.345133. – DOI 10.1145/345124.345133. – ISSN 0001–0782 5

[Romov u. Sokolov 2015] Romov, Peter ; Sokolov, Evgeny: RecSys Challenge 2015: Ensemble Learning with Categorical Features. In: *Proceedings of the 2015 International ACM Recommender Systems Challenge*. New York, NY, USA : ACM, 2015 (RecSys '15 Challenge). – ISBN 978–1–4503–3665–9, 1:1–1:4 12, 13, 34, 38, 39, 40, 41, 45

[Runkler 2015] Runkler, Thomas A.: *Data Mining: Modelle und Algorithmen intelligenter Datenanalyse*. Springer-Verlag, 2015 13, 69

[Said u. a. 2012] Said, Alan ; Tikk, Domonkos ; Shi, Yue ; Larson, Martha ; Stumpf, Klara ; Cremonesi, Paolo: Recommender systems evaluation: A 3D benchmark. In: *ACM RecSys 2012 workshop on Recommendation utility evaluation: beyond RMSE, Dublin, Ireland*, 2012, S. 21–23 1, 2

[Sarwar u. a. 2000] Sarwar, Badrul ; Karypis, George ; Konstan, Joseph ; Riedl, John: Application of dimensionality reduction in recommender system-a case study / DTIC Document. 2000. – Forschungsbericht 29

[Shi u. a. 2014] Shi, Yue ; Larson, Martha ; Hanjalic, Alan: Collaborative Filtering Beyond the User-Item Matrix: A Survey of the State of the Art and Future Challenges. In: *ACM Comput. Surv.* 47 (2014), Mai, Nr. 1, 3:1–3:45. http://dx.doi.org/10.1145/2556270. – DOI 10.1145/2556270. – ISSN 0360–0300 25, 26

[Siddiqui u. a. 2014] Siddiqui, Zaigham F. ; Tiakas, Eleftherios ; Symeonidis, Panagiotis ; Spiliopoulou, Myra ; Manolopoulos, Yannis: xStreams: Recommending Items to Users with Time-evolving Preferences. In: *Proceedings of the 4th International Conference on Web Intelligence, Mining and Semantics (WIMS14)*. New York, NY, USA : ACM, 2014 (WIMS '14). – ISBN 978–1–4503–2538–7, 22:1–22:12 21

[Song u. a. 2015] Song, Qiang ; Cheng, Jian ; Lu, Hanqing: Incremental matrix factorization via feature space re-learning for recommender system. In: *Proceedings of the 9th ACM Conference on Recommender Systems* ACM, 2015, S. 277–280 31

[Wickham u. a. 2014] Wickham, Hadley u. a.: Tidy data. In: *Under review* (2014) 9

[Xia u. a. 2006] Xia, Zhonghang ; Dong, Yulin ; Xing, Guangming: Support Vector Machines for Collaborative Filtering. In: *Proceedings of the 44th Annual Southeast Regional Conference*. New York, NY, USA : ACM, 2006 (ACM-SE 44). – ISBN 1–59593–315–8, 169–174 11

[Yan u. a. 2015] Yan, Peng ; Zhou, Xiaocong ; Duan, Yitao: E-Commerce Item Recommendation Based on Field-aware Factorization Machine. In: *Proceedings of the 2015 International ACM Recommender Systems Challenge* ACM, 2015, S. 2 34, 40

[Yağci u. a. 2015] Yağci, A. M. ; Aytekin, Tevfik ; Gürgen, Fikret S.: An Ensemble Approach for Multi-label Classification of Item Click Sequences. In: *Proceedings of the 2015 International ACM Recommender Systems Challenge*. New York, NY, USA : ACM, 2015 (RecSys '15 Challenge). – ISBN 978–1–4503–3665–9, 7:1–7:4 34, 40, 45, 64, 73

[Örnek 2016] Örnek, Deniz: *Die Behandlung der Filter Bubble bei Recommender Systemen*, Hochschule für Angewandte Wissenschaften Hamburg, Master Thesis, 2016. `http://users.informatik.haw-hamburg.de/~ubicomp/arbeiten/master/oernek.pdf` 32

Abbildungsverzeichnis

Tabellenverzeichnis

Versicherung über Selbständigkeit

Hiermit versichere ich, dass ich die vorliegende Arbeit ohne fremde Hilfe selbständig verfasst und nur die angegebenen Hilfsmittel benutzt habe.

Hamburg, 15. Dezember 2016 Eduard Weigandt