Zürcher Hochschule
für Angewandte Wissenschaften

School of
Engineering

InIT Institut für angewandte
Informationstechnologie

Projektarbeit (Informatik)

Hilti Big Data Competition: Recommender Systeme

Autoren	Marek Arnold Gabriel Eyyi
Hauptbetreuung	Thilo Stadelmann Kurt Stockinger
Industriepartner	Hilti Befestigungstechnik AG (9470 Buchs)
Externe Betreuung	Sebastian Nau
Datum	19.12.2014

Zürcher Hochschule
für Angewandte Wissenschaften

School of
Engineering

Erklärung betreffend das selbständige Verfassen einer Projektarbeit an der School of Engineering

Mit der Abgabe dieser Projektarbeit versichert der/die Studierende, dass er/sie die Arbeit selbständig und ohne fremde Hilfe verfasst hat. (Bei Gruppenarbeiten gelten die Leistungen der übrigen Gruppenmitglieder nicht als fremde Hilfe.)

Der/die unterzeichnende Studierende erklärt, dass alle zitierten Quellen (auch Internetseiten) im Text oder Anhang korrekt nachgewiesen sind, d.h. dass die Projektarbeit keine Plagiate enthält, also keine Teile, die teilweise oder vollständig aus einem fremden Text oder einer fremden Arbeit unter Vorgabe der eigenen Urheberschaft bzw. ohne Quellenangabe übernommen worden sind.

Bei Verfehlungen aller Art treten die Paragraphen 39 und 40 (Unredlichkeit und Verfahren bei Unredlichkeit) der ZHAW Prüfungsordnung sowie die Bestimmungen der Disziplinarmassnahmen der Hochschulordnung in Kraft.

Ort, Datum:

……

Unterschriften:

……

……..

……

Das Original dieses Formulars ist bei der ZHAW-Version aller abgegebenen Projektarbeiten zu Beginn der Dokumentation nach dem Titelblatt mit Original-Unterschriften und -Datum (keine Kopie) einzufügen.

Zusammenfassung

Grosse Unternehmen bieten auf ihren Online-Shops eine Vielzahl unterschiedlicher Produkte an. Oft ist es für den Kunden schwierig, bei dieser Menge das passende Produkt zu finden. Die Problematik besteht darin, dass dem Kunden noch nicht genau bekannt ist, was er sucht. Eine Möglichkeit, diesem Problem entgegenzuwirken, ist der Einsatz eines Recommender-System. Dabei werden dem Kunden aufgrund des vorherigen Kaufverhaltens und des Kaufverhaltens ähnlicher Benutzer Produkte empfohlen.

Das vorliegende Projekt besteht darin, im Rahmen der *Hilti Big Data Analytics Competition 2015* ein Prototyp eines Recommender-System zu entwickeln, um das Vertriebsportal Hilti Online (HOL) zu optimieren. Dabei stehen von der Firma Hilti umfangreiche anonymisierte Datensätze über ihre Produkte, deren Nutzung und ihre Kunden zur Verfügung.

Unter Verwendung dieser Daten konnten verschiedene Vorgehensmodelle für ein Recommender-System entwickelt werden, mit dem Ziel einen konfigurierbaren Prototyp eines Recommender-System zu implementieren. Zu diesem Zweck wurden die gegebenen Datensätze analysiert und mögliche Beziehungen zwischen den Datensätzen gesucht. In der Analyse wurden keine Beziehungen zwischen den Datensätzen gefunden. Weiter ergab die Analyse, dass die gekauften Produkte über keine Benutzerbewertungen verfügen. Aus diesem Grund wurden zuerst Konzepte für ein implizites Bewertungssystem erarbeitet, um diese anschliessend experimentell zu untersuchen.

Das Resultat der vorliegenden Arbeit ist ein konfigurierbarer Prototyp eines Recommender-System. Der Prototyp ermöglicht es dem Benutzer, Daten einzulesen und für jeden Kunden Empfehlungen zu erstellen. Ausserdem verfügt der Benutzer über die Möglichkeit, mehrere Konfigurationsparameter für das Recommender-System einzustellen, um das System für den gegeben Datensatz zu optimieren.

Die Optimierung der Konfigurationsparameter des Prototyps, mithilfe empirischer Untersuchgen, ergaben eine Erfolgsgüte von 50% sowie Precision von 54% und Recall von 47%.

Abstract

Large companies often offer a wide variety of products in their web shops. As a consequence, a lot of customers are overwhelmed by the flood of information when trying to find a suitable product. The main problem is that the customers often do not know exactly what they need. One solution to solve this problem is a recommender system, which provides recommendations for the customers based on both their and similar customers' previous buying behaviour.

Hilti wants to improve the user experience of their web shop customers on Hilti online (HOL) by introducing a recommender system to grant their customers the "Amazon experience". For this purpose, Hilti launched the *Hilti Big Data Analytics Competition 2015* and provided comprehensive datasets of their customers and products for the participants of the competition.

The goal of this thesis is therefore to investigate different concepts for a recommender system for HOL. Furthermore, a configurable prototype of a static recommender system is provided. First, specialist literature was consulted to get an overview of the state of the art technologies. Subsequently, the dataset provided by Hilti was analysed and first concepts were elaborated. Afterwards a testing system was implemented and the concepts were evaluated. The best concept was examined further and was finally implemented in the prototype.

The final prototype operates on static data using a collaborative filtering hybrid implementation combined with a k-means clustering algorithm. To achieve the final F1-score of 50% with recall of 54% and precision of 47% by four given recommendations per customer, the customers are distributed over 16 clusters and for each item 64 features are learned.

Inhalt

Kapitel 1 Einleitung

In der Bauwirtschaft nimmt laut Bundesamt für Statistik, der Anteil des E-Commerce an der Beschaffung sowie am Verkauf von Gütern und Dienstleistungen jährlich zu [1]. Die herkömmlichen Vertriebswege, wie Verkaufsläden, Telefon oder der Verkauf von Produkten über einen Vertreter werden vermehrt durch E-Commerce ersetzt. Eine der grössten Herausforderungen dabei ist es, dem Kunden aus der Flut von Artikeln diejenigen zu präsentieren, die ihn interessieren könnten, und dadurch den Umsatz des Unternehmens zu steigern. Grosse Unternehmen wie Amazon und Netflix zeigen, wie man das Medium Internet erfolgreich als Absatzmarkt nutzt, indem sie ihren Kunden auf sie angepasste Artikel/Filme empfehlen. Amazon erzielt 30% und Netflix 70% des Umsatzes über die Empfehlungen die sie ihren Kunden geben [2].

1.1 Problemstellung

Hilti Online (HOL) ist das Online-Verkaufsportal der Firma Hilti. Bisher ist auf HOL durch eine hierarchische Struktur jedes Produkt mit drei Mausklicks erreichbar, allerdings muss der Kunde wissen, was er benötigt. Die Vielzahl an Produkten welche auf HOL angeboten werden, erschweren es dem Kunden das passende Produkt zu finden. Dies führt zu einem hohen Zeitaufwand für den Kunden bei der Bestellung von Produkten, dadurch sinkt die Wahrscheinlichkeit, dass HOL als Bezugsquelle verwendet wird.

1.2 Industriepartner

Der Industriepartner der vorliegenden Arbeit ist der internationale Werkzeughersteller Hilti. Der Hauptsitz der Hilti Gruppe befindet sich in Schaan im Fürstentum Liechtenstein. Ihr Ziel ist es, über HOL einen grösseren Umsatz zu erzielen.

1.3 Bestehende Arbeit

Als Ausgangslange dieses Projekts dient die Implementierung eines Recommender-System für Filme von Dr. Kurt Stockinger. Diese Implementierung basiert auf einer Aufgabe des Online Machine-Learning Kurses von Andrew Ng [3].

1.4 Zielsetzung / Aufgabenstellung

In diesem Abschnitt werden die Aufgabenstellung und das Ziel der vorliegenden Arbeit vorgestellt und diskutiert. Die offizielle Aufgabenstellung ist im Anhang in *Abschnitt 7.2* zu finden.

Die originale Aufgabenstellung von Hilti lautet:

> *„Develop a recommendation engine that provides tailor-made leads for our HOL customers. Translate B2C concepts into the B2B world and offer our customers an "amazon experience". This task is a combination of a cross selling engine and a conceptual marketing work.*
> *[..] answer the question: Given customer xy, which product (tool or consumable) should I recommend to him? "* [4].

Aus der Aufgabenstellung von Hilti geht hervor, dass einem Benutzer das richtige Produkt aus den Klassen *Tool* oder *Consumable* empfohlen werden soll. Folglich ist das Ziel dieser Arbeit:

> *„Entwicklung eines Prototyps eines Recommender-Systems für HOL, basierend auf anonymisierten Verkaufs-, Kunden- und Produktmasterdaten."*

Dieses Ziel wird auf die Problemstellung angewendet und kann wie folgt interpretiert werden:

> Das Ziel der Arbeit ist die Entwicklung von Verfahrensmodellen für die Erstellung eines konfigurierbaren Prototyps und dessen Umsetzung.

1.5 Anforderungen

Aus der Aufgabenstallung von Hilti ergeben sich folgende Anforderungen:

- **Empfehlung von *Tools* und *Consumables***
  Den Kunden sollen nur Produkte aus den Klassen *Tools* und *Consumables* empfohlen werden.
- **Statische Daten**
  Der Prototyp muss nur auf statischen Daten arbeiten und muss daher keine neuen Kunden und Produkte aufnehmen können.

1.6 Vorausgesetztes Wissen

Für die vorliegende Arbeit werden grundlegende Kenntnisse in der Statistik und in der Linearen Algebra benötigt. Des Weiteren sind Kenntnisse in den Technologien SQL und MATLAB von Vorteil.

1.7 Ausblick auf die Kapitel

Nachfolgend werden die Kapitel dieser Arbeit kurz erläutert.

Im Kapitel **Theoretische Grundlagen** werden Themen behandelt auf welche die vorliegende Arbeit aufbaut.

Im Kapitel **Methoden** werden die Daten von Hilti analysiert, Messgrössen definiert und Konzepte für ein implizites Bewertungssystem entwickelt.

Im Kapitel **Experimente** werden die durchgeführten Experimente beschrieben und die Resultate diskutiert.

Im Kapitel **Diskussion und Ausblick** werden die Resultate anhand der Anforderungen evaluiert, weiter werden Wege zur Weiterführung dieser Arbeit vorgestellt

Kapitel 2 Theoretische Grundlagen

In diesem Kapitel werden die theoretischen Grundlagen für die vorliegende Arbeit vermittelt. Zuerst werden im *Abschnitt 2.2* Recommender-Systems definiert sowie deren Ziele, Zwecke und Schwachstellen aufgezeigt. Danach werden in den nächsten Abschnitten drei Recommender-System-Verfahren vorgestellt und anhand von Beispielen beschrieben. Im *Abschnitt 2.9* werden zwei Clustering-Verfahren sowie ein Hilfsmittel für die Analyse von Clustering-Verfahren beschrieben.

2.1 Vorgaben und Abgrenzungen

Die vorliegende Arbeit beschränkt sich auf die Analyse von drei Recommender-System-Verfahren (*vgl. Abbildung 1*), dies wurde als Vorgabe von den Betreuern definiert.

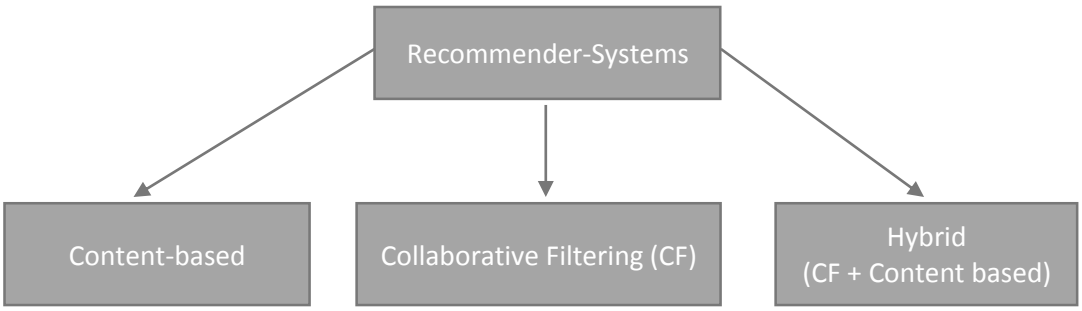

Abbildung 1 Überblick Recommender-Systems [5]

Zudem beschränkt sich die vorliegende Arbeit auf zwei Clustering-Verfahren (*vgl. Abbildung 2*) die in MATLAB schon implementiert sind und die in der Literatur häufig verwendet werden [5]. Als partitionierendes Verfahren wurde k-Mean ausgewählt und als hierarchisches Verfahren wurde ein agglomeratives Clustering-Verfahren verwendet. Beide Clustering-Verfahren werden im *Abschnitt 2.9* genauer beschrieben.

In der *Abbildung 2* sind die zwei Clustering-Verfahren dargestellt.

Abbildung 2 Clustering-Verfahren

2.2 Recommender-Systems

Als Recommender-Systems werden Systeme bezeichnet, deren Aufgabe darin besteht, Prognosen oder Empfehlungen für Benutzer abzugeben [6].

In der *Encyclopedia of Machine Learning* wird das Ziel eines Recommender-System wie folgt definiert:

> *„The goal of a recommender system is to generate meaningful recommendations to a collection of users for items or products that might interest them."* [7]

In der vorliegenden Arbeit wird der Begriff „Item" verwendeten. Item ist die allgemeine Bezeichnung für das Objekt, welches das System dem Benutzer empfehlen soll [8].

2.2.1 Der Zweck von Recommender-Systems

Es gibt viele Gründe, warum Waren- oder Dienstleistungsanbieter Recommender-Systems nutzen sollten. Nachfolgend werden kurz die wichtigsten Gründe für ein Recommender-System beschrieben, um den Zweck von Recommender-Systems besser verstehen zu können [9].

- **Umsatzsteigerung**
  Das Hauptziel von Recommender-Systems im kommerziellen Betrieb ist der Verkauf von zusätzlichen Items, die der Benutzer ohne eine Empfehlung nicht gekauft hätte. Im nicht direkt kommerziellen Betrieb werden ähnliche Ziele angestrebt, beispielsweise die Erhöhung der Anzahl Benutzer, die den zur Verfügung gestellten Service nutzen.

- **Steigerung der Benutzerzufriedenheit**
  Die Kombination von akkuraten Empfehlungen und einem benutzerfreundlichen User Interface erhöht die Wahrscheinlichkeit, dass der Benutzer das System in der Zukunft vermehrt nutzen wird.

- **Kundenbedürfnisse besser verstehen**
  Ein weiterer wichtiger Grund Recommender-Systems einzusetzen, ist die Beschreibung der Benutzer Präferenzen. Diese Präferenzen werden explizit gesammelt oder vom System vorausgesagt. Anhand dieser Präferenzen lassen sich die Items besser für die Kundenbedürfnisse anpassen und so die Verwaltung der Lagerbestände oder die Produktion optimieren.

2.2.2 Begriffsdefinitionen

Allgemeine Begriffsdefinitionen für Recommender-Systems, wie sie im vorliegenden Dokument verwendet werden [10].

Name	*Parameter*	*Beschreibung*
Items	$I_i : I_1..I_n$	Produkte und Dienstleistungen: Es gibt n Items im System, die mit i indiziert werden.
Item-Eigenschaften	$F_k : F_1..F_p$	Eigenschaften der Items: Jedes Item hat p Eigenschaften, die mit k indiziert werden.
Eigenschafts-Vektoren	$x^{(i)} : x^{(1)}..x^{(n)}$	Eigenschafts-Vektoren der Items: Es gibt für jedes Item i einen Eigenschafts-Vektor $x^{(i)}$. $x_k^{(i)}$ bezeichnet die Eigenschaft k des Items i.
Benutzer	$U_j : U_1..U_m$	Benutzer des Systems: Es gibt m Benutzer im System, die mit j indiziert werden.
Benutzer-Präferenzen	$\theta^{(j)} : \theta^{(1)}..\theta^{(m)}$	Jeder Benutzer hat Präferenzen für jede Eigenschaft eines Items, diese werden durch den Vektor $\theta^{(j)}$ repräsentiert. $\theta_k^{(j)}$ bezeichnet die Präferenz des Benutzers j für die Eigenschaft k.
Bewertungen	$y^{(i,j)} : Y$	Y: Alle gegebenen Bewertungen der Items durch die Benutzer $y^{(i,j)}$: Gegebene Bewertung des Items i durch den Benutzer j.
Berechnete Bewertungen	$y'^{(i,j)}=(\theta^{(j)})^T x^{(i)}$	In einem idealen System entspricht die Bewertung eines Items der Summer der Produkte der entsprechenden Präferenzen und Eigenschaften. $y'^{(i,j)}$: Berechnete Bewertung des Items i durch den Benutzer j.

Tabelle 1 Begriffsdefinition

2.2.3 Algorithmische Schwachstellen von Recommender-Systems

In diesem Abschnitt werden die Schwachstellen von Recommender-Systems beschrieben.

- **Sparsity-Problem**
  Bei der Initialisierung eines Recommender-System oder bei einer sehr grossen Anzahl von Items können Probleme auftauchen. Der Grund ist die kleine Anzahl Items, die von Benutzern effektiv bewertet werden. Dies führt zu einer dünnbesetzten Bewertungsmatrix und somit zu einem drastischen Abfall der Empfehlungsgüte. Dies ist vor allem bei Collaborative Filtering Systems ein Problem, da die Wahrscheinlichkeit Benutzer zu finden, die gleiche Items gleich oder ähnlich bewertet haben, sinkt [11].

- **Kaltstart-Problem (The Cold-Start Problem)**
  Neue Items oder neue Benutzer stellen eine grosse Hürde für Recommender-Systems dar, da dem System noch zu wenige Informationen über den Benutzer oder über das Item vorliegen. Dieses Problem wird in der Literatur als Kaltstart-Problem bezeichnet [12].

  André Klahold beschreibt das Problem bei Collaborative Filtering Systems wie folgt:
  „Da Empfehlungen offensichtlich nur auf dem Verhalten anderer Benutzer ausgesprochen werden können, muss ein Collaborative Filtering-Verfahren zunächst eine „kritische Menge" an Benutzeraktionen erfasst haben, bevor es Empfehlungen aussprechen kann. Das gilt nicht nur für die Collaborative Filtering-Matrix an sich (und damit für neue Empfehlungselemente), sondern insbesondere auch für jeden neuen Benutzer." [13].

- **Über Spezifikation (Over Fitting)**
  Eine der grössten Gefahren beim Trainieren von Recommender-Systems ist das Over Fitting. Dabei werden Zusammenhänge erlernt, die nur in den Trainingsdaten bestehen.

  In der Encyclopedia of Machine Learning wird das Problem folgendermassen beschrieben: *„It is trivial to find a rule set that is complete and consistent on the training data. To achieve this, one only needs to convert each positive example into a rule. Each of these rules is consistent (provided the data set is not inconsistent), and collectively they cover the entire example set (completeness). However, this is clearly a bad case of overfitting because the theory will not generalize to new positive examples. "* [14].

2.2.4 Regularisierung

In vielen Verfahren für Recommender-Systems wird ein Regularisierungs-Term verwendet. Dieser dient dazu, die erlernten Parameter möglichst klein zu halten und Extremwerte zu verhindern. Dadurch wird die Gefahr des Over Fitting reduziert und das System gewinnt an numerischer Stabilität. Zur Gewichtung des Regularisierungs-Terms wird häufig ein Parameter λ verwendet [15].

Der Regularisierungs-Term erhöht die Komplexität des Problems durch die Einschränkung des Systems. Es gilt den richtigen Wert für den Regularisierungs-Parameter λ zu finden, daher einen Wert der Over Fitting möglichst verhindert, ohne dabei das System zu blockieren [16].

Nachfolgend werden zwei grundlegende Verfahren von Recommender-Systems vorgestellt, das Content-Based-Filtering und das Collaborative Filtering.

2.3 Content-Based-Verfahren

Dieses Verfahren basiert auf den gegebenen Eigenschaften der zu empfehlenden Items. Die gegebenen Eigenschaften ermöglichen es eine Prognose über einen Benutzer erstellen zu können. Content-Based-Systems fokussieren die Eigenschaften des Items. Ähnliche Items werden erkannt, indem die Gleichheit der Eigenschaften der Items untersucht wird [17].

2.4 Content-Based-Verfahren: Umsetzung

Die Formeln in diesem Abschnitt stammen aus dem Online-Kurs von Adrew Ng [18].

2.4.1 Idee

Um die Vorlieben der Benutzer darzustellen werden ihre Präferenzen als Präferenzen der Eigenschaften der Items definiert. Das System lernt diese Präferenzen anhand der gegebenen Bewertungen und Eigenschaften der Items. Das System kann später anhand der erlernten Präferenzen und der gegebenen Eigenschaften der Items Empfehlungen geben.

2.4.2 Beispiel Filmbewertung

In diesem Beispiel soll das Content-Based-Verfahren anhand von Filmbewertungen veranschaulicht werden.

- **Ausgangslage**

Die Firma MusterMovieSeller verkauft online Filme an Kunden. Nach dem Kauf eines Films verfügt der Kunde über die Möglichkeit Filme zu bewerten. Dem Kunden steht eine Bewertungsskala von 0 bis 5 zur Verfügung, wobei eine 0 für eine sehr schlechte Bewertung und eine 5 für eine sehr gute Bewertung steht. Nicht bewertete Filme werden mit einem Fragezeichen gekennzeichnet. Zusätzlich zu den Bewertungen werden die Filme, anhand von gegebenen Daten in die Genre Action und Drama eingeordnet. Je besser sich ein Film einem Genre zuordnen lässt, desto grösser ist dessen Eigenschaftswert. Der Eigenschaftswert kann Werte zwischen 0 und 1 annehmen.

Die Items sind in diesem Fall die Filme (I_n), welche von MusterMovieSeller verkauft werden. Die Eigenschaften der Filme entsprechen ihren Genres.

In der *Tabelle 2* sind die Bewertungen der Benutzer und die Eigenschaftswerte der Filme eingetragen

	Teil 1: Bewertungen			*Teil 2: Eigenschaftswerte*	
	Alice(U_1)	Bob(U_2)	Carol(U_3)	Action (F_1)	Drama (F_2)
(I_1) Shooter	5	5	0	0.9	0
(I_2) Rambo	5	?	?	1	0.01
(I_3) Heat	?	4	0	1.0	0
(I_4) Gladiator	0	0	5	0.1	1.0
(I_5) Oldboy	0	0	5	0	0.9

Tabelle 2 Beispiel Content-Based-Verfahren

Legende für Tabelle 2

- Eigenschaftswert F_n:
  Beispiel: Der Film Shooter hat die Eigenschaften $F_1 = 0.9$ und $F_2 = 0$
  Der Film Shooter lässt sich gut dem Genre Action zuordnen
- Bewertungsvektor von Benutzer (U_n):
  Beispiel: Für Bob(U_2) = $[5, ?, 4, 0, 0]^T$

Die *Tabelle 2* zeigt die Benutzerbewertungen von Alice(U_1), Bob(U_2) und Carol(U_3), sowie die Eigenschaftswerte der fünf Filme. Die Benutzerbewertungen sind im ersten Teil der Tabelle und die Eigenschaftswerte sind im zweiten Teil der Tabelle ersichtlich. Zur Verdeutlichung wird der Benutzer Bob(U_2) genauer betrachtet. In der *Tabelle 2* ist ersichtlich, dass der Benutzer Bob den Film Shooter mit einer 5, den Film Heat mit einer 4 und die Filme Gladiator und Oldboy mit einer 0 bewertet hat. Bob hat den Film Rambo noch nicht bewertet, deshalb ist ein Fragezeichen eingetragen. Anhand der Eigenschaftswerte wird sichtbar, dass die bevorzugten Filme von Bob dem Genre Action zugeordnet werden können.

- **Ziel**

Das Ziel ist es, die Benutzer-Präferenzen von Bob zu erlernen.

- **Vorgehen**

Anhand der Bewertungen von Bob und den Eigenschaften der Filme können nun die Präferenzen von Bob geschätzt werden. Die Bewertung eines Filmes entspricht im Idealfall der Summe der Produkte der Eigenschaften des Films mit den entsprechenden Benutzer-Präferenzen. Im Allgemeinen kann jedoch nur die Abweichung über alle bewerteten Filme minimiert werden, in dem die Benutzer-Präferenzen angepasst werden.

In dem folgenden System sind die Benutzer-Bewertungen von Bob und die für ihn berechneten Bewertungen gegenüber gestellt, es gilt den Fehler dieses Systems zu minimieren [18].

$$
\begin{aligned}
5 &\approx (\theta^{(2)})^T x^{(1)} = \theta_1^{(2)} * 0.9 + \theta_2^{(2)} * 0.0 \\
4 &\approx (\theta^{(2)})^T x^{(3)} = \theta_1^{(2)} * 1.0 + \theta_2^{(2)} * 0.0 \\
0 &\approx (\theta^{(2)})^T x^{(4)} = \theta_1^{(2)} * 0.1 + \theta_2^{(2)} * 1.0 \\
0 &\approx (\theta^{(2)})^T x^{(5)} = \theta_1^{(2)} * 0.0 + \theta_2^{(2)} * 0.9
\end{aligned}
$$

Aus der Minimierung des Fehlers dieses Systems folgt:

$$\theta^{(2)} \approx \begin{bmatrix} 4.5 \\ 0 \end{bmatrix}$$

- **Prognose**

Anhand der ermittelten Benutzer-Präferenzen von Bob kann nun eine Prognose für die Bewertung des Films Rambo durch Bob wie folgt erstellt werden:

Benutzer-Präferenzen von Bob:

$$\theta^{(2)} \approx \begin{bmatrix} 4.5 \\ 0 \end{bmatrix}$$

Eigenschafts-Vektor des Films Rambo:

$$x^{(2)} \approx \begin{bmatrix} 1.0 \\ 0.001 \end{bmatrix}$$

Prognose für die Bewertung des Films Rambo durch Bob:

$$(\theta^{(2)})^T x^{(2)} = 4.5 * 1.0 + 0.0 * 0.001 = 4.5$$

Das Recommender-System würde Bob den Film Rambo empfehlen.

2.4.3 Verallgemeinerung

- **Gegeben: Y, $\mathbf{x^{(1)}}..\mathbf{x^{(n)}}$ Ziel: $\boldsymbol{\theta^{(j)}}$**

In diesem Schritt werden die Benutzer-Präferenzen eines Benutzers $\theta^{(j)}$ anhand der von ihm bewerteten Items gefunden. Dazu benutzen wir die folgende Funktion.

$$\boldsymbol{min_{\theta^{(j)}}}\left[\frac{\mathbf{1}}{\mathbf{2}}\sum_{\boldsymbol{i:r(i,j)=1}}\left((\boldsymbol{\theta^{(j)}})^{\boldsymbol{T}}\boldsymbol{x^{(i)}}-\boldsymbol{y^{(i,j)}}\right)^{\mathbf{2}}+\frac{\boldsymbol{\lambda}}{\mathbf{2}}\sum_{\boldsymbol{k=1}}^{\boldsymbol{p}}\left(\boldsymbol{\theta_k^{(j)}}\right)^{\mathbf{2}}\right]$$

Formel 1 Finden der Benutzer-Präferenzen für einen Benutzer

In der nachfolgenden Tabelle werden die Terme der *Formel 1* beschrieben.

Term	*Beschreibung*
$\sum_{i:r(i,j)=1}$	Summe: Summe über alle Items i, die der Benutzer j bewertet hat.
$\left((\theta^{(j)})^{T}x^{(i)}-y^{(i,j)}\right)^{2}$	Squared error: Quadrierte Abweichung der berechneten von den tatsächlichen Bewertungen der Items durch die Benutzer.
$\sum_{k=1}^{p}\left(\theta_k^{(j)}\right)^{2}$	Regularisierungs-Term: Wird addiert, um die Werte der Benutzer-Präferenzen möglichst klein zu halten.
$\frac{\lambda}{2}$	Regularisierungs-Parameter: Wird zur Gewichtung der Regularisierung verwendet.

Tabelle 3 Erklärung für Formel 1

- **Gegeben: Y, $\mathbf{x^{(1)}}..\mathbf{x^{(n)}}$ Ziel: $\boldsymbol{\theta^{(1)}}..\boldsymbol{\theta^{(m)}}$**

Für die Auffindung der Benutzer-Präferenzen aller Benutzer, wird die *Formel 1* wie folgt erweitert [1]:

$$\boldsymbol{min_{\theta^{(1)}..\theta^{(m)}}}\left[\frac{\mathbf{1}}{\mathbf{2}}\sum_{\boldsymbol{j=1}}^{\boldsymbol{m}}\sum_{\boldsymbol{i:r(i,j)=1}}\left((\boldsymbol{\theta^{(j)}})^{\boldsymbol{T}}\boldsymbol{x^{(i)}}-\boldsymbol{y^{(i,j)}}\right)^{\mathbf{2}}+\frac{\boldsymbol{\lambda}}{\mathbf{2}}\sum_{\boldsymbol{j=1}}^{\boldsymbol{m}}\sum_{\boldsymbol{k=1}}^{\boldsymbol{p}}\left(\boldsymbol{\theta_k^{(j)}}\right)^{\mathbf{2}}\right]$$

Formel 2 Finden der Benutzer-Präferenzen für alle Benutzer

In der nachfolgenden Tabelle wird der in der *Formel 2* eingeführte Term beschrieben. Der restliche Term ist in der *Tabelle 3* beschrieben.

Term	*Beschreibung*
$\sum_{j=1}^{m}$	Summe: Summe über alle Benutzer j.

Tabelle 4 Erklärung für Formel 2

2.4.4 Vorteile

Im folgenden Abschnitt sind die wichtigsten Vorteile aufgelistet [19].

- **Benutzer Unabhängigkeit**
  Content-Based Verfahren benötigen nur die Bewertungen des aktiven Benutzers um Empfehlungen erstellen zu können. Das Verhalten anderer Benutzer spielt keine Rolle.
- **Transparenz**
  Die Korrektheit der Empfehlungen des Recommender-System lässt sich anhand der Eigenschaften oder Beschreibungen der Items überprüfen.
- **Neue Items**
  Content-Based Verfahren sind in der Lage neue Items aufzunehmen.

2.4.5 Nachteile

Im folgenden Abschnitt sind die wichtigsten Nachteile aufgelistet [20].

- **Limitierte Inhaltsanalyse**
  Content-Based-Verfahren können nur eine bestimmte Anzahl Eigenschaften für die Empfehlung verwenden. Oft wird auch domänenspezifisches Wissen benötigt.
- **Über-Spezifikation**
  Bei einer Über-Spezifikation liefert das System dem Benutzer keinen Mehrwert. Das Verfahren kann dem Benutzer keine neuen Items empfehlen.
  Beispiel: Ein Benutzer bewertet nur Filme von Stanley Kubrick. Aus diesem Grund würde das System dem Benutzer nur Filme von Stanley Kubrick empfehlen und keine weiteren Filme die ihn interessieren könnten.
- **Neue Benutzer (Kaltstart-Problem)**
  Das System benötigt genügend Bewertungen eines Benutzers um für diesen zuverlässige Empfehlung geben zu können. Bei neuen Benutzern führt dies zu Problemen, da noch keine Bewertungen vorhanden sind.
- **Eigenschaften abhängig**
  Content-Based-Verfahren hängen stark von den Eigenschaften der Items ab, auf die sie angewendet werden.

2.5 Collaborative Filtering

Die zugrundliegende Annahme dieses Verfahrens ist, dass Benutzer die einige Items ähnlich bewertet haben, auch andere Items ähnlich bewerten werden. Daher können die Bewertungen eines Benutzers für Items, die er selbst nicht bewertet hat, anhand der Bewertungen dieser Items von ähnlichen Benutzern geschätzt werden [21].

2.6 Collaborative-Filtering: Umsetzung

Die Formeln in diesem Abschnitt stammen aus dem Online-Kurs von Adrew Ng [22].

2.6.1 Idee

Mit Collaborative-Filtering können die Eigenschaften von Items anhand der Bewertungen durch Benutzer und deren gegebenen Präferenzen erlernt werden. Durch die erlernten Eigenschaften der Items und den gegebenen Präferenzen der Benutzer ist es möglich, Bewertungen von Items, welche ein Benutzer nicht bewertet hat, zu schätzen [23].

2.6.2 Beispiel

In der *Tabelle 5* sind fünf Items gegeben, welche von vier Benutzern bewertet wurden. Gesucht sind die Eigenschaften der Items, um die fehlenden Bewertungen schätzen zu können.

	Teil 1: Bewertungen				*Teil 2: Eigenschaftswerte*	
	Alice(U_1)	Bob(U_2)	Carol(U_3)	David(U_4)	F_1	F_2
(I_1) Shooter	5	5	0	?	?	?
(I_2) Rambo	5	?	?	?	?	?
(I_3) Heat	?	4	0	0	?	?
(I_4) Gladiator	0	0	5	5	?	?
(I_5) Oldboy	0	0	5	5	?	?

Tabelle 5 Beispiel Collaborative Filtering Verfahren

Gegebene Benutzer-Präferenzen (*vgl. Abschnitt 2.2.2*) :

$$\theta^1 = \begin{bmatrix}5\\0\end{bmatrix};\ \theta^2 = \begin{bmatrix}5\\0\end{bmatrix};\ \theta^3 = \begin{bmatrix}0\\5\end{bmatrix};\ \theta^4 = \begin{bmatrix}0\\5\end{bmatrix}$$

Anschliessend wird gezeigt, wie die Eigenschaften des Films Shooter gefunden werden können.

- **Finden des Item-Eigenschafts-Vektors $x^{(1)}$**

Anhand der Bewertungen und der Benutzer-Präferenzen können die Eigenschaften des Films Shooter (I_1) ermittelt werden, da die Bewertung eines Items im Idealfall dem Produkt der Benutzer-Präferenzen und der Item-Eigenschaften entspricht. Im Allgemeinen kann jedoch nur die Abweichung der berechneten von der tatsächlichen Bewertung minimiert werden, indem die Eigenschaften der Items angepasst werden.

In dem folgenden System sind die Benutzer-Bewertungen des Films Shooter und die berechneten Bewertungen gegenüber gestellt, es gilt den Fehler dieses Systems zu minimieren:

$$5 \approx (\theta^{(1)})^T x^{(1)} = x_1^{(1)} * 5 + x_2^{(1)} * 0$$
$$5 \approx (\theta^{(2)})^T x^{(1)} = x_1^{(1)} * 5 + x_2^{(1)} * 0$$
$$0 \approx (\theta^{(3)})^T x^{(1)} = x_1^{(1)} * 0 + x_2^{(1)} * 5$$

Aus der Minimierung des Fehlers dieses Systems folgt:

$$x^{(1)} = \begin{bmatrix} 1.0 \\ 0.0 \end{bmatrix}$$

- **Resultat**

Anhand des ermittelten Eigenschafts-Vektors für den Film Shooter kann nun eine Prognose für die Bewertung des Films Shooter durch David wie folgt erstellt werden:

Benutzer-Präferenzen von David:

$$\theta^{(2)} \approx \begin{bmatrix} 0 \\ 5 \end{bmatrix}$$

Eigenschafts-Vektor des Films Shooter:

$$x^{(2)} \approx \begin{bmatrix} 1.0 \\ 0.0 \end{bmatrix}$$

Prognose für die Bewertung des Films Shooter durch David:

$$(\theta^{(4)})^T x^{(1)} = 0.0 * 1.0 + 5.0 * 0.0 = 0.0$$

Das Recommender-System würde David den Film Shooter nicht empfehlen.

2.6.3 Verallgemeinerung

- **Gegeben: Y, $\boldsymbol{\theta^{(1)}}..\boldsymbol{\theta^{(m)}}$ Ziel: $\mathbf{x^{(i)}}$**

In diesem Schritt werden die Eigenschaften eines Items $x^{(i)}$ anhand der gegebenen Bewertungen und der Benutzer-Präferenzen ermittelt. Dazu wird folgende Funktion verwendet.

$$\boldsymbol{min}_{\boldsymbol{x^{(i)}}} \left[\frac{\mathbf{1}}{\mathbf{2}} \sum_{\boldsymbol{j:r(i,j)=1}} \left((\boldsymbol{\theta^{(j)}})^{\boldsymbol{T}} \boldsymbol{x^{(i)}} - \boldsymbol{y^{(i,j)}} \right)^{\mathbf{2}} + \frac{\boldsymbol{\lambda}}{\mathbf{2}} \sum_{\boldsymbol{k=1}}^{\boldsymbol{p}} \left(\boldsymbol{x_k^{(i)}} \right)^{\mathbf{2}} \right]$$

Formel 3 Collaborative Filtering über ein Item

In der nachfolgenden Tabelle werden die Terme der *Formel 3* beschrieben.

Term	*Beschreibung*
$\sum_{\boldsymbol{j:r(i,j)=1}}$	Summe: Summe über alle Benutzer j, die das Item i bewertet haben.
$\left((\boldsymbol{\theta^{(j)}})^{\boldsymbol{T}} \boldsymbol{x^{(i)}} - \boldsymbol{y^{(i,j)}} \right)^{\mathbf{2}}$	Squared error: Quadrierte Abweichung der geschätzten von den tatsächlichen Bewertungen der Items durch die Benutzer.
$\sum_{\boldsymbol{k=1}}^{\boldsymbol{p}} \left(\boldsymbol{\theta_k^{(j)}} \right)^{\mathbf{2}}$	Regularisierungs-Term: Wird addiert um die Werte der Eigenschaften der Items möglichst klein zu halten.
$\frac{\boldsymbol{\lambda}}{\mathbf{2}}$	Regularisierungs-Parameter: Wird zur Gewichtung der Regularisierung verwendet.

Tabelle 6 Erklärung für Formel 3

- **Gegeben: Y, $\theta^{(1)}..\theta^{(m)}$ Ziel: $x^{(1)}..x^{(n)}$**

Nun sollen die Eigenschaften aller Items auf einmal gefunden werden. Mit der folgenden Formel ist dies möglich.

$$\boldsymbol{min}_{x^{(1)}..x^{(n)}}\left[\frac{1}{2}\sum_{i=1}^{n}\sum_{j:r(i,j)=1}\left((\theta^{(j)})^{T}x^{(i)}-y^{(i,j)}\right)^{2}+\frac{\lambda}{2}\sum_{i=1}^{n}\sum_{k=1}^{p}\left(x_{k}^{(i)}\right)^{2}\right]$$

Formel 4 Collaborative Filtering über alle Items

In der nachfolgenden Tabelle wird der in der *Formel 4* eingeführte Term beschrieben. Der restliche Term ist in der *Tabelle 6* beschrieben.

Term	*Beschreibung*
$\sum_{i=1}^{n}$	Summe: Summe über alle Items i.

Tabelle 7 Erklärung für Formel 4

2.6.4 Vorteile

Im folgenden Abschnitt sind die wichtigsten Vorteile aufgelistet [24].

- **Eigenschaften lernen**
  Collaborative Filtering Verfahren ermöglichen es die Eigenschaften eines Items zu erlernen, *vgl. Abschnitt 2.5.1.*
- **Hohe Qualität**
  Collaborative Filtering bezieht die Bewertungen aller Benutzer mit ein. Dadurch ist es möglich die Eigenschaften der Items gut einzuschätzen und dementsprechend gute Vorschläge zu generieren.
- **Cross-Genre Empfehlungen**
  Da die Empfehlungen anhand der Bewertungen aller Benutzer generiert werden, können dem Benutzer Items empfohlen werden, die sich von seinen bisher bewerteten Items stark unterscheiden, aber den Benutzer dennoch interessieren könnten**.**

2.6.5 Nachteile

Im folgenden Abschnitt sind die wichtigsten Nachteile aufgelistet [25].

- **Keine Transparenz**
  Empfehlungen des Systems lassen sich kaum nachvollziehen, da die Empfehlungen auf Ähnlichkeiten der Benutzer und deren Verhalten basieren. Welche Benutzer dafür verglichen werden ist nicht bekannt.
- **Neue Items (Kaltstart-Problem)**
  Collaborative Filtering Verfahren benötigen viele Bewertungen für ein Item um Prognosen über die Bewertungen von Items zu geben.
- **Benutzer-Präferenzen**
  Es kann sehr schwer sein, die Präferenzen der Benutzer zu ermitteln.
- **Sparsity Problem**
  Um akkurate Prognosen erstellen zu können, werden viele Item-Bewertungen durch viele Benutzer benötigt.

2.7 Collaborative Filtering Hybrid

Die Vorhersagen der Benutzer-Bewertung der Items sollen nur anhand der gegebenen Bewertungen erlernt werden.

2.8 Collaborative Filtering Hybrid: Umsetzung

Die Formeln in diesem Abschnitt stammen aus dem Online-Kurs von Adrew Ng [26].

2.8.1 Idee

Die beiden Verfahren Content-Based und Collaborative Filtering kombinieren, um die Benutzer-Präferenzen und Item-Eigenschaften zu erlernen.

2.8.2 Algorithmus

- **Gegeben: Y Ziel: $\boldsymbol{\theta^{(1)}..\theta^{(m)}}$, $\boldsymbol{x^{(1)}..x^{(n)}}$ lernen**

Nachfolgend wird beschrieben, wie Content-Based-Verfahren und Collaborative Filtering Verfahren kombiniert werden können.

Ablauf

1. Die Benutzer-Präferenzen θ auf zufällige Werte initialisieren.
2. Item-Eigenschaften x anhand der gegebenen θ mit Collaborative Filtering Verfahren lernen.
3. Benutzer-Präferenzen θ anhand der gegebenen x mit Content-Based-Verfahren lernen.
4. Vorhersagen für die Bewertungen von Items anhand der gelernten Eigenschaften der Items und Benutzer-Präferenzen berechnen.

Um genauere Vorhersagen für die Bewertungen von Items zu erhalten, werden die *Schritte 2 und 3* mehrmals wiederholt.

2.8.3 Vereinfachung

Um den Ablauf zu vereinfachen, werden die *Schritte 2 und 3* kombiniert. Dazu definieren wir die Funktion J:

$$\boldsymbol{J\left(x^{(1)}..x^{(n)},\theta^{(1)}..\theta^{(m)}\right)} = \frac{1}{2}\sum_{(i,j):r(i,j)=1}\left((\theta^{(j)})^{T}x^{(i)} - y^{(i,j)}\right)^2 + \frac{\lambda}{2}\sum_{i=1}^{n}\sum_{k=1}^{p}\left(x_k^{(i)}\right)^2 + \frac{\lambda}{2}\sum_{j=1}^{m}\sum_{k=1}^{p}\left(\theta_k^{(j)}\right)^2$$

Formel 5 Collaborative Filtering Hybrid über alle Items

Daraus resultiert das folgende Minimierungsproblem:

$$min_{x^{(1)}..x^{(n)},\theta^{(1)}..\theta^{(m)}}\left(J\left(x^{(1)}..x^{(n)},\theta^{(1)}..\theta^{(m)}\right)\right)$$

Formel 6 Collaborative Filtering Hybrid Minimierungsproblem

Die Item-Eigenschaften x und die Benutzer-Präferenzen θ können nun in einem Schritt erlernt werden.

Ablauf

1. $x^{(1)}..x^{(n)}, \theta^{(1)}..\theta^{(m)}$ auf kleine zufällige Werte initialisieren. Die Werte sollten voneinander verschieden sein, um die Symmetrie zu brechen.
2. $J\left(x^{(1)}..x^{(n)}, \theta^{(1)}..\theta^{(m)}\right)$ minimieren z.B. mit Gradient Descent.
3. Nun können Bewertungen mit den gelernten x und θ vorhergesagt werden: $\theta^T x$

2.8.4 Vektorisierung

Zur effizienten Berechnung der vorhergesagten Bewertungen können die Parameter, wie nachfolgend beschrieben, in Matrizen geschrieben werden.

- Bewertungen Y:

$$Y = \begin{bmatrix} 5 & 5 & 0 & 0 \\ 5 & ? & ? & 0 \\ ? & 4 & 0 & ? \\ 0 & 0 & 5 & 4 \end{bmatrix}$$

In den Zeilen dieser Matrix stehen die Produkte und in den Spalten die Benutzer.

- Item-Eigenschaften-Vektoren x:

$$x = \begin{bmatrix} -\left(x^{(1)}\right)^T - \\ \vdots \\ -\left(x^{(n)}\right)^T - \end{bmatrix}$$

- Benutzer-Präferenzen θ:

$$\theta = \begin{bmatrix} -\left(\theta^{(1)}\right)^T - \\ \vdots \\ -\left(\theta^{(m)}\right)^T - \end{bmatrix}$$

- Vorhergesagte Bewertungen:
    - Low rank matrix factorization: $\theta^T x$

$$\begin{bmatrix} (\theta^{(1)})^T x^{(1)} & \dots & (\theta^{(m)})^T x^{(1)} \\ \vdots & \ddots & \vdots \\ (\theta^{(1)})^T x^{(n)} & \dots & (\theta^{(m)})^T x^{(n)} \end{bmatrix}$$

2.8.5 Vorteile

Im folgenden Abschnitt sind die wichtigsten Vorteile aufgelistet [27].

- **Geringer Verwaltungsaufwand**
  Es werden nur die Bewertungen der Items durch die Benutzer benötigt. Es müssen daher keine zusätzlichen Daten erfasst werden (*vgl. Idee in Kapitel 2.6*).
- **Flexibilität**
  Collaborative Filtering Hybride benötigen nur die Bewertungen der Items durch die Benutzer. Dadurch können sie auf beliebige Items angewandt werden.
- **Hohe Qualität**
  Collaborative Filtering Hybride beziehen die Bewertungen aller Benutzer mit ein. Dadurch ist es möglich die Item-Eigenschaften gut einzuschätzen und dementsprechend gute Vorschläge zu generieren.
- **Cross-Genre Empfehlungen**
  Da die Empfehlungen anhand der Bewertungen aller Benutzer generiert werden, können dem Benutzer Items empfohlen werden, die sich von seinen bisher bewerteten Items stark unterscheiden, aber den Benutzer dennoch interessieren könnten.

2.8.6 Nachteile

Im folgenden Abschnitt sind die wichtigsten Nachteile aufgelistet.

- **Keine Transparenz**
  Empfehlungen des Systems lassen sich kaum nachvollziehen, da die Empfehlungen auf Ähnlichkeiten der Benutzer und deren Verhalten basieren. Welche Benutzer dafür verglichen werden, ist nicht bekannt (*vgl. Abschnitt 2.6.5*).
- **Neue Items (Kaltstart-Problem)**
  Collaborative Filtering Verfahren benötigen viele Bewertungen für ein Item um Prognosen über die Bewertungen von Items zu geben (*vgl. Abschnitt 2.4.5*).
- **Sparsity-Problem**
  Um akkurate Prognosen erstellen zu können, werden viele Item-Bewertungen durch viele Benutzer benötigt (*vgl. Abschnitt 2.6.5*).

2.9 Clusteranalyse

Das Ziel der Clusteranalyse ist die Einteilung einer Anzahl von Items in homogene Gruppen. Zu diesem Zweck werden die Items durch ihre Eigenschaften beschrieben. Items innerhalb einer Gruppe sollen über möglichst ähnliche Eigenschaften, Items in unterschiedlichen Gruppen über möglichst unterschiedliche Eigenschaften verfügen [28].

In den nächsten zwei Abschnitten werden zwei grundlegenden Clustering-Verfahren kurz vorgestellt.

2.9.1 k-Mean-Clustering

Das k-Mean-Clustering ist ein Algorithmus der eine Menge von Daten automatisch in kohärente Gruppen (Cluster) einteilt. Dabei wählt der Algorithmus zunächst willkürliche Punkte im Vektorraum als Anfangszentren und ordnet die Vektoren den nächstgelegenen Anfangszentren zu, danach führt der Algorithmus iterativ eine Verbesserung der Anfangszentren durch, solange bis sich die Anfangszentren nicht mehr ändern [29].

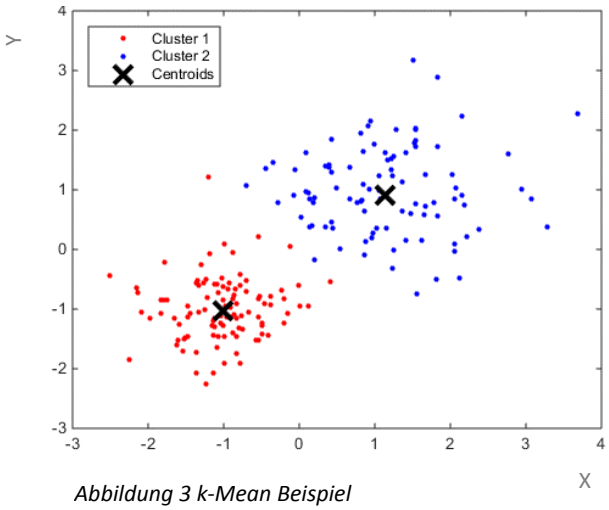

Abbildung 3 k-Mean Beispiel

Die *Abbildung 3* stammt aus einem Bespiel von MathWorks [30] und stellt das Ergebnis eines k-Mean Clusterings von zufällig generierten Punkten mit zwei Clustern in einem Streudiagramm dar. Die Anfangszentren (Centroids) sind mit einem schwarzen Kreuz gekennzeichnet.

2.9.2 Hierarchische Clusteranalyse

Hierarchische Clusterverfahren sind distanzbasierte Verfahren zur Clusteranalyse. Dabei wird die Datenmenge in eine abgestufte Folge (Hierarchie) von Clustern eingeteilt.

Hierarchische Clusterverfahren werden in zwei Typen unterteilt. Die agglomerierenden Verfahren teilen zunächst jedes Item einem Cluster zu und verschmelzen danach sukzessive zwei benachbarte Cluster. Die teilenden Verfahren arbeiten genau umgekehrt, zuerst wird ein grosser Cluster erstellt, welcher anschliessend schrittweise in kleinere Cluster aufgeteilt wird [32].

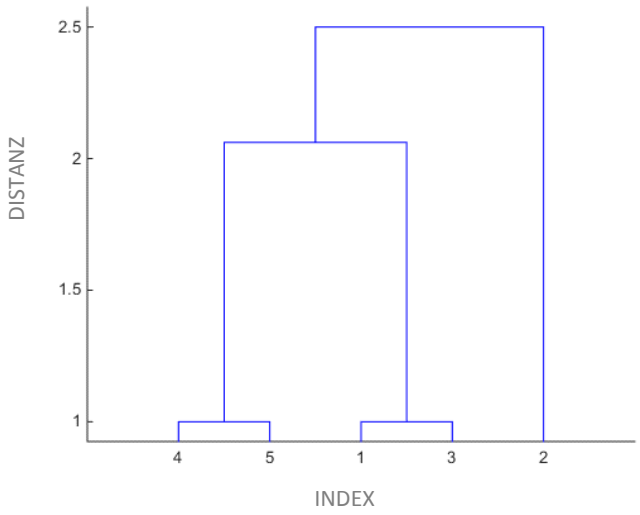

Abbildung 4 Hierarchisches Clustering Beispiel

Die *Abbildung 4* stammt aus einem Beispiel von MathWorks [31] und stellt das Ergebnis eines Hierarchischen Clustering in einem Dendrogramm dar, wobei die horizontale Achse den Index-Wert des Objekts und die vertikale Achse die Distanz zwischen den Objekten darstellt.

2.9.3 Silhouette-Koeffizient

Der Silhouette-Koeffizient ist ein von der Cluster-Anzahl unabhängiges Gütemass für die Struktur des Clustering. Er ist definiert als die durchschnittliche Silhouette aller Objekte eines Clustering [33].

Die Silhouette $s(o)$ eines Objekt o ist definiert als:

$$\boldsymbol{s(o) = \frac{b(o) - a(o)}{\max\{a(o), b(o)\}}}$$

Der Wert $a(o)$ beschreibt den Abstand eines Objekts zum Repräsentanten seines Clusters und der Wert $b(o)$ beschreibt den Abstand zum Repräsentanten des nächstgelegenen Cluster.

Für die Berechnung des Abstands können verschiedene Distanzmasse oder Ähnlichkeitsmasse verwendet werden [34].

Der Wertebereich von s(o) ist: $-1 \leq s(o) \leq 1$

- $s(o) \approx 1 \quad \Rightarrow gute\ Zuordnung\ des\ Objekts\ o\ zu\ seinem\ Cluster$
- $s(o) \approx 0 \quad \Rightarrow indifferente\ Zuordnung\ des\ Objekts\ o\ zu\ seinem\ Cluster$
- $s(o) \approx -1 \Rightarrow schlechte\ Zuordnung\ des\ Objekt\ o\ zu\ seinem\ Cluster$

2.9.3.1 Interpretation des Silhouette-Koeffizienten

Die Interpretation des Silhouette-Koeffizienten ist unabhängig von der Cluster-Anzahl und ist in der *Tabelle 9* dargestellt [35].

Silhouetten Koeffizient	*Interpretation*
0.71 – 1.00	Starke Struktur
0.51 – 0.70s	Brauchbare Struktur
0.26 – 0.50	Unbrauchbare Struktur
≤ 0.25	Keine Struktur

Tabelle 8 Interpretation des Silhouette-Koeffizienten

Kapitel 3 Methoden

In diesem Kapitel werden zuerst die Daten von Hilti analysiert. Auf den Ergebnissen aufbauend, wird in *Abschnitt 3.3* ein passendes Recommender-System-Verfahren ausgewählt. Im *Abschnitt 3.6* werden die entwickelten Konzepte für Bewertungssysteme vorgestellt. Für die Entwicklung des Prototyps werden in *Abschnitt 3.7* Verfahrensmodelle entwickelt. Schliesslich wird in *Abschnitt 3.8* der entwickelte Prototyp vorgestellt.

Der Ablauf ist in *Abbildung 5* dargestellt.

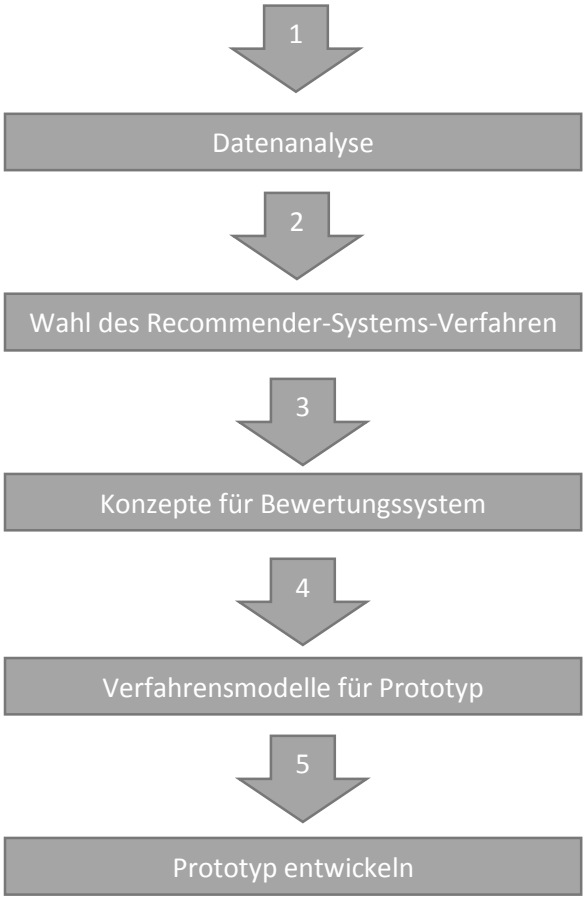

Abbildung 5 Ausblick

3.1 Vorbereitung

Die Wahl eines geeigneten Verfahrens für ein Recommender-System basiert auf der Qualität des gegeben Datensatzes. Aus diesem Grund wurde der gegebene Datensatz genauer analysiert.

Im *Abschnitt 3.1.1* werden die Tabellen von Hilti vorgestellt und genauer beschrieben.
Im *Abschnitt 3.2* werden die wichtigsten Datenanalysenergebnisse vorgestellt.

3.1.1 Beschreibung der Datensätze von Hilti

Die zu untersuchenden Daten der Firma Hilti sind in zwei Tabellen aufgeteilt. Die erste Tabelle heisst *Hilti_dataset_activities_training* (*vgl. Tabelle 9*) und enthält Informationen über die Aktivitäten der Kunden mit dem Kundendienst, die zweite Tabelle heisst *Hilti_dataset_training* (*vgl. Tabelle 10*) und enthält Kaufinformationen zu den Produkten. Zusätzlich stellt die Firma Hilti für beide Tabellen je einen Validierungsdatensatz zur Verfügung, der die gleiche Struktur aufweist.

Die folgenden zwei Tabellen zeigen einen Auszug aus den Datensätzen, Raute-Zeichen stellen leere Einträge dar. Die genauen Beschreibungen der Spalten der Tabellen von Hilti sind im Anhang in *Abschnitt 7.3* zu finden.

CustomerID	**6**	**6**	**9**	**9**	**9**
ActDateFrom	19.10.12	03.01.14	12.01.14	16.01.14	04.04.14
CreatedBy	650	650	650	650	650
Category	MT	TEL	MT	TEL	TEL
Function	8	8	8	8	8
Objective	J	#	#	J	#
Result	#	#	#	#	#
NumAct	1	1	2	1	1

Tabelle 9 Auszug aus Hilti_dataset_activities_training

CustomerID	**6**	**6**	**9**	**9**	**9**
ProductCode	XAATTSBF	WMUPTSBC	WMUPTSBC	JIBTEKBO	JIBTEQBO
IPCClass	XAATT	WMUPT	WMUPT	JIBTE	JIBTE
IPCLine	XAAT	WMUP	WMUP	JIBT	JIBT
IPCClassDistinct	1	5	5	1	1
PurchaseDate	01.01.13	01.01.13	14.01.13	14.01.13	14.01.13
Quantity	1	1	1	0	1
QtyUnit	ST	ST	ST	ST	ST
NetSales	2114	953	953	953	2505.41
SalesChannel	C2	C2	C2	C2	C2
EngagementStatus	L4	L4	L3	L3	L3
PotentialClass	D	D	C	C	C
FleetUser	N	N	N	N	N
HOLUser	No	No	Yes	Yes	Yes
ShipToTrade	PJVO	PJVO	PJVO	PJVO	PJVO
VisitFrequency	F3	F3	F6	F6	F6
NumberEmployees	9	9	13	13	13
ShipToPostalCode	ICFHRE	ICFHRE	BCINBU	BCINBU	BCINBU
DeletionFlag	#	#	#	#	#
Territory	279	279	508	508	508

Tabelle 10 Auszug aus Hitli_dataset_training

3.2 Datenanalyse

Um ein besseres Bild über die Daten zu erhalten, haben wir uns für die Datenanalyse folgende Fragen gestellt:

- **Wie viele verschiedene Kunden, Produkte und Klassen sind in der Tabelle enthalten?**
  In der *Tabelle 11* sind die Häufigkeiten der Kunden, Produkte und Klassen aufgelistet.

Beschreibung	*Absolute Häufigkeit*
Kunden	28587
Produkte	948
C-Klasse (IPCClass)	293
B-Klasse (IPCLines)	132
A-Klasse (IPCClassDistinct)	5

Tabelle 11 Kennzahlen aus der Tabelle Hilti_dataset_training

Aus den Daten wird ersichtlich, dass die Produkte und Klassen hierarchisch organisiert sind. Jedes Produkt ist einer C-Klasse, jede C-Klasse einer B-Klasse und jede B-Klasse ist einer A-Klasse untergeordnet. Diese Hierarchie ist in der *Abbildung 6* visuell dargestellt.

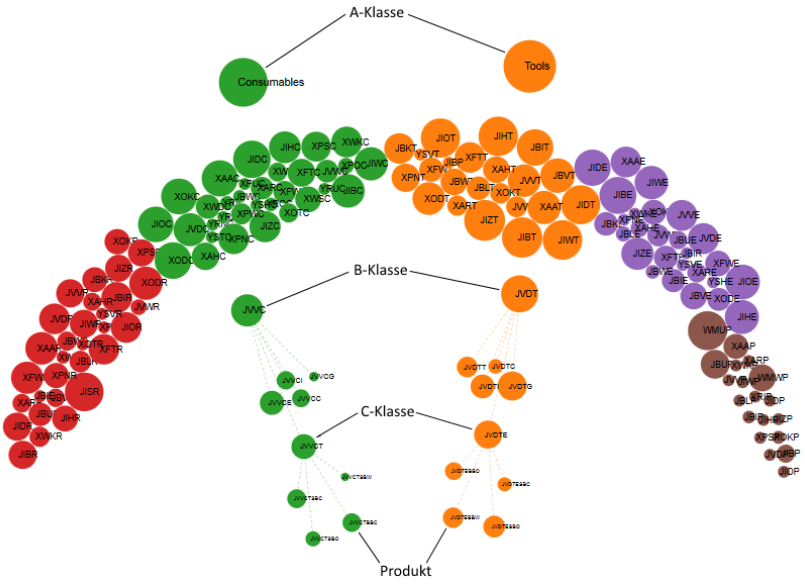

Abbildung 6 Hierarchie

Aus den Anforderungen in *Abschnitt 1.5* geht hervor, dass nur Produkte aus den A-Klassen *Tools* und *Consumables* (Orange und Grün in der *Abbildung 6*) empfohlen werden sollen. Produkte aus den anderen drei Klassen sollen nicht empfohlen werden.

- **Verfügen die Validierungstabellen über neue Benutzer bzw. neue Produkte?**
  Die Validierungstabellen beinhalten keine weiteren Benutzer sowie keine weiteren Produkte.

- **Gibt es Kunden die nur in einer Tabelle vorkommen?**
  Alle Kunden die in der Tabelle *Hilti_dataset_activities_training* eingetragen sind, haben auch einen Eintrag in der Tabelle *Hilti_dataset_training*.

- **Sind in den Tabellen Benutzerbewertungen der gekauften Produkte enthalten?**
  Die Tabellen enthalten keine Bewertungen durch die Benutzer. Somit wurden die gekauften Produkte von den Benutzern nicht explizit bewertet.

- **Wie viele Datensätze gibt es und über welchen Zeitraum sind diese verteilt?**
  Das Trainingsdatenset (*Hilti_dataset_training*) verfügt über zwei Millionen Einträge und deckt einen Zeitraum von über zwei Jahren ab. Die weiteren Informationen sind in der *Tabelle 12* aufgelistet. Eine Zeile in der entsprechenden Tabelle stellt einen Einkauf bzw. eine Aktivität eines Kunden dar. In der Spalte *Zeitraum (d)* steht die Differenz vom letzten Eintrag mit dem ersten Eintrag.

Name	*Anzahl Zeilen*	*Zeitraum (d)*
Hilti_dataset_activities_training	789‘184	2‘254‘556
Hilti_dataset_activities_validation	347‘578	60
Hilti_dataset_training	2‘733‘728	915
Hilti_dataset_validation	59’168	60

Tabelle 12 Metainformationen der Tabellen

  Der grosse Zeitraum in *Hilti_dataset_activities_training* lässt sich durch Einträge, welche weit in der Zukunft liegen, erklären.

- **Sind die Kundendienstaktivitäten mit dem Kauf verbunden?**
  Zwischen den beiden Tabellen *Hilti_dataset_activities_training* und *Hilti_dataset_training* wurden keine Verbindungen gefunden. Bei der Analyse wurde für alle Einträge über die Interaktionen mit dem Kundendienst im Zeitraum von bis zu 30 Tagen nach der Interaktion ermittelt, ob der entsprechende Kunde im untersuchten Zeitraum einen Kauf getätigt hat. Mehrere Kunden haben zwar mehrere Aktivitäten mit dem Kundendienst sowie mehrere Käufe im untersuchten Zeitraum getätigt, allerdings wird aus diesem Versuch nicht ersichtlich, ob sich eine Aktivität des Kunden mit dem Kundendienst positiv auf einen Kauf ausgewirkt hat.

3.2.1 Fazit der Datenanalyse

Die wichtigsten Ergebnisse der Datenanalyse sind die fehlenden expliziten Benutzerbewertungen, die fehlenden Zusammenhänge zwischen den Tabellen *Hilti_dataset_training und Hilti_dataset_activities_training*, und die hierarchische Organisation der Produkte und Klassen.

Aus den Ergebnissen folgen zwei Erkenntnisse, welches die Entwicklung eines ersten Prototyps erleichtern. Zum einen können die Tabellen isoliert betrachtet werden, da keine Zusammenhänge gefunden wurden. Zum anderen ermöglicht die hierarchische Struktur, dass die Empfehlungen auf der Stufe von Klassen anstelle von Produkten gegeben werden.

3.3 Wahl des Verfahrens

Die von Hilti zur Verfügung gestellten Daten enthalten weder Eigenschaften, Präferenzen noch explizite Bewertungen, allerdings können implizite Bewertungen, wie im *Abschnitt 3.6* beschrieben, aus den Daten extrahiert werden.

In der nachfolgenden Tabelle sind die benötigten Daten für die in *Kapitel 2* vorgestellten Verfahren aufgelistet.

	Content-Based	*Collaborative Filtering*	*Collaborative Filtering Hybrid*
Eigenschaften	X		
Präferenzen		X	
Bewertungen	X	X	X

Tabelle 13 Wahl des Verfahrens

Da weder Eigenschaften noch Präferenzen vorhanden sind, wird ein Verfahren basierend auf einem Collaborative Filtering Hybriden entwickelt. Nachfolgend wird in dieser Arbeit Collaborative Filtering als Bezeichnung des hybriden Verfahrens verwendet.

3.4 Methodische Grundüberlegungen

An dieser Stelle wird noch einmal das Ziel der vorliegenden Arbeit in Erinnerung gerufen:

> Das Ziel der Arbeit, ist die Entwicklung von Verfahrensmodelle für die Erstellung eines konfigurierbaren Prototyps und dessen Umsetzung.

Im folgenden Abschnitt sind die methodischen Grundüberlegungen, basierend auf dem Ziel der vorliegenden Arbeit, aufgelistet.

In einem ersten Schritt müssen Konzepte für ein implizites Bewertungssystem entwickelt werden, da die Daten von der Firma Hilti keine expliziten Benutzer-Bewertungen enthalten, auf diesen Konzepten aufbauend, müssen in einem zweiten Schritt Verfahrensmodelle für die Entwicklung eines Prototyps in MATLAB entwickelt werden. In einem dritten Schritt müssen für die Überprüfung der Leistungsfähigkeit dieser Konzepte und Verfahrensmodelle Messgrössen definiert werden. Für die empirische Überprüfung der erarbeiteten Konzepte und Verfahrensmodelle soll im letzten Schritt ein konfigurierbarer Prototyp eines Recommender-System in MATLAB entwickelt werden.

3.5 Messgrössen

3.5.1 Bewertungsmatrix

In Recommender-Systems sind die Benutzer-Bewertungen der Items durch die Benutzer die wichtigste Datenquelle. Die Daten liegen als Bewertungsmatrix vor, in der für jedes Benutzer-Item Paar ein Wert für den Grad der Vorliebe des Benutzers für das entsprechende Item steht.

- **Beispiel**

In diesem Beispiel entsprechen die Filme den Items und Anna, Beat und Claudia sind die Benutzer.

Filme	*Anna*	*Beat*	*Claudia*
Inception	2	5	0
Harry Potter 1	0	3	4
Harry Potter 2	5	0	5
Snatch	1	2	5

Tabelle 14 Bewertungsmatrix mit Annotationen

In der *Tabelle 14* sind die Bewertungen der Filme Inception, Harry Potter 1 und 2 und Snatch durch die Personen Anna, Beat und Claudia dargestellt. Die Bewertungen gehen von eins bis fünf, wobei fünf der besten Bewertung entspricht. Eine Null in der Tabelle bedeutet, dass die entsprechende Person den Film nicht bewertet hat. Aus den Bewertungen in der *Tabelle 14* kann nun eine Bewertungsmatrix generiert werden. Die Dimension der Bewertungsmatrix ist abhängig von der Anzahl der Filme und Benutzer.

In diesem Beispiel ist die Bewertungsmatrix definiert als:

$$Bewertungsmatrix = [Anzahl\ Benutzer\ X\ Anzahl\ Filme] = \begin{bmatrix} 2 & 5 & 0 \\ 0 & 3 & 4 \\ 5 & 0 & 5 \\ 1 & 2 & 5 \end{bmatrix}$$

Die Zeilen der Matrix stellen die verschiedenen Filme und die Spalten die verschiedenen Benutzer dar.

3.5.2 Dichte (density)

Für die Untersuchung der Konzepte haben wir die Dichte der Bewertungsmatrix berechnet. Die Dichte ist die Anzahl der nicht Nullwerte durch die Anzahl aller Werte in der Bewertungsmatrix.

$$\boldsymbol{Dichte} := \frac{\boldsymbol{C}}{\boldsymbol{N}}$$

Formel 7 Dichte

Symbol	*Beschreibung*
C	Die Anzahl der nicht Nullwerte in der Bewertungsmatrix
N	Die Anzahl aller Werte in der Bewertungsmatrix

Tabelle 15 Symbole für Dichte

- **Beispiel Dichteberechnung**

$$\mathbf{Bewertungsmatrix} = \begin{bmatrix} \mathbf{1} & \mathbf{9} & \mathbf{22} \\ \mathbf{3} & \mathbf{0} & \mathbf{1} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} \end{bmatrix}$$

Aus der obigen Bewertungsmatrix folgt, dass C = 5 und N = 9 ist. Demzufolge beträgt die Dichte: $\frac{\mathbf{5}}{\mathbf{9}}$

3.5.3 Erfolgsgütemass

Für die Untersuchung der Leistungsfähigkeit der erarbeiteten Konzepte wurde das Erfolgsgütemasse (F1) verwendet.

In den folgenden Abschnitten wird beschrieben, wie das Erfolgsgütemass berechnet wird. Dabei dient die Berechnung von Precision und Recall als Grundlage.

Die *Tabelle 16* stellt die Wahrheitsmatrix für die vier Werte **richtig positiv**, **falsch negativ**, **falsch positiv** und **richtig negativ** dar. Diese Werte werden für die Berechnung von Recall und Precision benötigt [36].

Wahrheitsmatrix (Konfusionsmatrix)

	Kunde hat Produkt in dieser C-Klasse gekauft ($r_p + f_n$)	*Kunde hat kein Produkt in dieser C-Klasse gekauft ($f_p + r_n$)*
(Klassifikator) Empfehlung positiv (r_p+f_p)	richtig positiv (r_p)	falsch positiv (f_p)
(Klassifikator) Empfehlung negativ ($f_n + r_n$)	falsch negativ (f_n)	richtig negativ (r_n)

Tabelle 16 Wahrheitsmatrix

Legende für Tabelle 16

- Richtig positiv (r_p):
  Der Kunde hat ein Produkt in dieser C-Klasse gekauft, und das System hat ihm diese C-Klasse richtig empfohlen.
- Falsch negativ (f_n):
  Der Kunde hat ein Produkt in dieser C-Klasse gekauft, und das System hat ihm diese C-Klasse fälschlicherweise nicht empfohlen.
- Falsch positiv (f_p):
  Der Kunde hat kein Produkt in dieser C-Klasse gekauft, und das System hat ihm diese C-Klasse fälschlicherweise empfohlen.
- Richtig negativ (r_n):
  Der Kunde hat kein Produkt in dieser C-Klasse gekauft, und das System hat ihm diese C-Klasse nicht empfohlen

1. ***Recall:***
   Recall entspricht dem Anteil der korrekt als positiv klassifizierten Objekte an der Gesamtheit der tatsächlich positiven Objekte.
   $$P(positiv\ erkannt \mid tatsächlich\ positiv) = \frac{r_p}{r_p + f_n}$$
2. ***Precision:***
   Precision entspricht dem Anteil der korrekt als positiv klassifizierten Objekte an der Gesamtheit der als positiv klassifizierten Objekte.
   $$P(richtig\ positiv \mid positiv\ erkannt) = \frac{r_p}{r_p + f_p}$$
3. ***F1-Mass (Erfolgsgütemass):***
   Die Kombination von Precision und von Recall mittels des gewichteten harmonischen Mittels wird als F1-Mass bezeichnet.
   $$\boldsymbol{F1 = 2 * \frac{(Precision * Recall)}{(Precision + Recall)}}$$

3.6 Konzepte für ein Bewertungssystem

Der Einsatz eines Collaborative Filtering Hybrid Verfahrens setzt eine Bewertungsmatrix voraus, die aus den Bewertungen der Benutzer generiert werden kann. Allerdings verfügen die Daten von Hilti über keine expliziten Benutzer-Bewertungen. Aus diesem Grund muss eine Bewertungsmatrix anhand impliziter Bewertungen generiert werden.

Es gibt mehrere Möglichkeiten die Daten zu interpretieren, um implizite Benutzer Bewertungen zu erhalten. In den folgenden Abschnitten sind die Anforderungen an die Bewertungsmatrix und die von uns erarbeiteten Konzepte, für eine implizite Benutzer-Bewertung, aufgeführt.

3.6.1 Anforderungen an die Bewertungsmatrix

Die Bewertungsmatrix soll eine möglichst hohe Dichte aufweisen, denn aus der Literatur ist bekannt, dass bei der Anwendung eines Collaborative Filtering Verfahrens auf Matrizen mit geringer Dichte Probleme auftreten können *(vgl. Abschnitt 2.2.3)*.

Aus den Erkenntnissen der Datenanalyse im *Abschnitt 3.2* ist bekannt, dass die Produkte und Klassen hierarchisch organisiert sind. Aus diesem Grund wird vermutet, dass eine Bewertungsmatrix auf Stufe von C-Klassen eine höhere Dichte aufweist, als eine Bewertungsmatrix auf Stufe von Produkten. Diese Vermutung soll mit dem ersten Experiment verifiziert werden.

3.6.2 Bewertungssystem

Um die Bewertungen, wie in den nachfolgenden Abschnitten beschrieben, aus den Daten zu extrahieren, wurden SQL-Abfragen verwendet, die Abfragen dazu sind im Anhang in *Abschnitt 7.4* zu finden.

Mit diesen Abfragen werden eindimensionale Listen erzeugt, welche die Bewertungsmatrizen Spaltenweise repräsentieren. Die resultierenden Listen werden anschliessend Komma-Separiert gespeichert.

3.6.2.1 Beispiel

Dieses Beispiel dient zur Veranschaulichung der einzelnen Konzepte. Für alle Konzepte wird eine Bewertungsmatrix anhand der Beispieldaten aus der *Tabelle 17* berechnet.

- **Ausgangslange**

Für dieses Beispiel steht die *Tabelle 17* zur Verfügung. In der Tabelle sind die Produktkäufe der Kunden aufgelistet. Der Kunde Meier hat drei Produkte aus der C-Klasse Bohrmaschine und zwei Produkte aus der C-Klasse Schraubenzieher gekauft. Der Kunde Felix hingegen hat zwei Produkte aus der C-Klasse Bohrmaschine und zwei Produkte aus der C-Klasse Messgerät gekauft.

Kunde	**Meier**	**Meier**	**Meier**	**Meier**	**Meier**	**Meier**	**Felix**	**Felix**	**Felix**	**Felix**
C-Klassen	B	B	B	B	S	S	B	B	M	M
Produkt	B_1	B_2	B_5	B_5	S_5	S_2	B_2	B_2	M_2	M_4
Stückzahl	1	4	5	5	2	4	3	1	3	2

Tabelle 17 Beispieldaten

Legende für Tabelle 17

- B: Bohrmaschine
- B_n: Produkt aus der C-Klasse Bohrmaschine
- S: Schraubenzieher
- S_n**:** Produkt aus der C-Klasse Schraubenzieher
- M: Messgerät
- M_n: Produkt aus der C-Klasse Messgeräte

- **Bewertungsmatrix**

Die *Tabelle 17* verfügt über keine Benutzerbewertungen. Daher besteht die Bewertungsmatrix aus lauter Nullen. Die Dimension der Bewertungsmatrix ist 3x2, da in der Tabelle drei verschiedene C-Klassen und zwei verschieden Kunden existieren.

Die Bewertungsmatrix ist somit definiert als:

$$Bewertungsmatrix = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}$$

Die erste Spalte beinhaltet die Bewertungen des Benutzers Meier und die zweite Spalte beinhaltet die Bewertungen des Benutzers Felix. Die Zeilen der Bewertungsmatrix stehen für die einzelnen C-Klassen, wobei die erste Zeile die C-Klasse Bohrmaschine, die zweite Zeile die C-Klasse Schraubenzieher und die dritte Zeile die C-Klasse Messgerät repräsentiert.

3.6.3 Konzept 1: Boolean-Rating

Eine Möglichkeit ist ein Boolean-Rating. Bei einem Boolean-Rating bestehen die Bewertungen aus den Werten 0 und 1. Dabei bedeutet eine 1 in der Matrix den Kauf mindestens eines Produkts aus der entsprechenden C-Klasse durch den Kunden. Eine 0 bedeutet, dass der Kunde kein Produkt aus der entsprechenden C-Klasse gekauft hat. Anhand dieser Bewertungen werden alle fehlenden Einträge in der Bewertungs-Matrix erlernt und anschliessend mit den Validierungs-Daten verglichen.

Auf das obige Beispiel angewendet ergibt dies folgende Bewertungsmatrix:

$$Bewertungsmatrix = \begin{bmatrix} 1 & 1 \\ 1 & 0 \\ 0 & 1 \end{bmatrix}$$

3.6.4 Konzept 2: Count-Rating

Eine zweite Möglichkeit ein implizites Bewertungssystem zu erstellen ist, wenn die Bewertungen eines Kunden den Anzahl Käufen eines Produkts aus einer C-Klasse entsprechen. Falls der Kunde keine Produkte einer C-Klasse erworben hat, beträgt die Bewertung dieser C-Klasse 0.

Auf das obige Beispiel angewendet ergibt dies folgende Bewertungsmatrix:

$$Bewertungsmatrix = \begin{bmatrix} 4 & 2 \\ 2 & 0 \\ 0 & 2 \end{bmatrix}$$

3.6.5 Konzept 3: Count-Quantity-Rating

Eine dritte Möglichkeit ein implizites Bewertungssystem zu erstellen ist, die Stückzahl in die Bewertungen mit einfliessen zu lassen. Die Bewertungen entsprechen der Summe der Stückzahl der gekauften Produkte in einer C-Klasse eines Kunden. Falls der Kunde nie ein Produkt einer C-Klasse gekauft hat, ist die Bewertung dieser Klasse 0.

Auf das obige Beispiel angewendet ergibt dies folgende Bewertungsmatrix:

$$Bewertungsmatrix = \begin{bmatrix} 15 & 4 \\ 6 & 0 \\ 0 & 5 \end{bmatrix}$$

3.6.6 Clustering der Kunden nach Bewertungen

Eine weitere Möglichkeit die Dichte der Bewertungsmatrix zu erhöhen ist der Einsatz von Clustering-Verfahren. Die Cluster können anhand der Bewertungen der Kunden erstellt werden, sodass Kunden, die ähnliche Produkte ähnlich bewertet haben, im gleichen Cluster sind.

3.7 Verfahrensmodelle

Für die Entwicklung eines Prototyps eines Recommender-System wurden zwei Vorgehensmodelle erarbeitet. Die Vorgehensmodelle basieren auf dem Collaborative Filtering Hybrid Verfahren und zeigen auf, wie die Prognosematrix erlernt und die Erfolgsgüte optimiert werden kann.

3.7.1 Allgemeines Verfahren – Collaborative Filtering Hybrid

- **Ziel**

Eine Prognosematrix (P) soll generiert werden und deren Erfolgsgüte (F1) soll berechnet werden.

- **Idee**

Die Prognosematrix soll mit Hilfe eines Collaborative Filtering Hybrid Verfahren generiert werden.

- **Vorgehen**

Zuerst wird die Bewertungsmatrix (Y) von allen Benutzern eingelesen. Anhand dieser Bewertungen generiert das Collaborative Filtering Verfahren eine Prognosematrix (P). Mithilfe der Bewertungsmatrix (Y) und der Prognosematrix (P) wird die Erfolgsgüte berechnet (*vgl. Abbildung 7*).

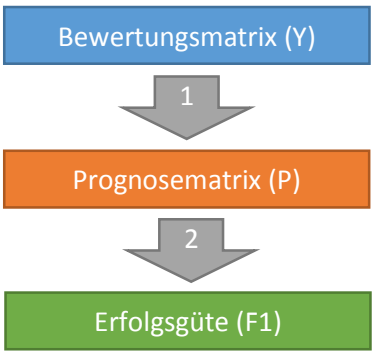

Abbildung 7 Allgemeines Vorgehen

Legende für Abbildung 7

- Bewertungsmatrix (Y): Generiert durch die erarbeiteten Konzepte
- Prognosematrix (P): Resultat des Collaborative Filtering Hybrid Verfahren

3.7.2 Erweitertes Verfahren – Collaborative Filtering Hybrid mit Clustering

- **Ziel**

Das Ziel des erweiterten Verfahrens ist die Verbesserung der Empfehlungen.

- **Idee**

Eine Möglichkeit um die Erfolgsgüte zu verbessern ist die Dichte der Matrix zu vergrössern. Die Dichte der Bewertungsmatrix lässt sich mithilfe von Clustering-Methoden vergrössern.

- **Vorgehen**

Zuerst wird die Bewertungsmatrix (Y) von allen Benutzern eingelesen. Mithilfe eines Clustering-Algorithmus wird die Bewertungsmatrix in Cluster (Gruppen) aufgeteilt. Einzelne Cluster verfügen dann über eine höhere Dichte als die Bewertungsmatrix. Für diese Cluster werden separat alle Bewertungen geschätzt (P_1 bis P_4). Diese Schätzungen werden mithilfe von Collaborative Filtering Hybrid durchgeführt. Die Cluster werden zur Prognosematrix vereint, um die Erfolgsgüte berechnen zu können (*vgl. Abbildung 8*).

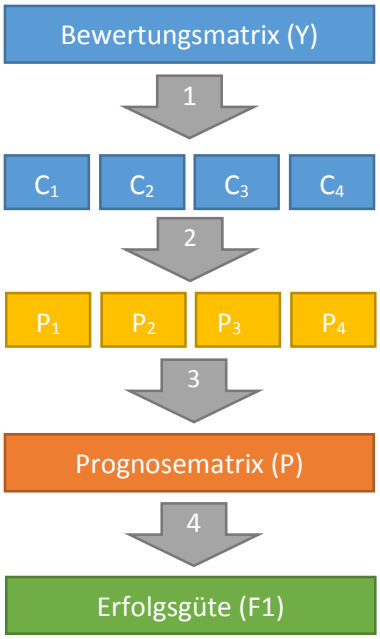

Abbildung 8 Vorgehen mit Clustering

Legende für Abbildung 8

- C_n: Cluster mit der Nummer n
- P_n: Prognose für Cluster n

3.8 Prototyp

Der entwickelte Prototyp besteht aus zwei Komponenten. Zu einem die Grundkomponente, welche auf der Übungsaufgabe von Andrew Ng basiert, und den Collaborative Filtering Hybird Algorithmus sowie die Minimierungsfunktion enthält. Zum anderen die Komponente welche mithilfe der Verfahrensmodelle entwickelt wurde. Diese Komponente stellt Clustering-Funktionen, Import- und Export-Funktionen sowie Automatisierungsfunktionen zur Verfügung. Dies ermöglicht eine empirische Überprüfung der Bewertungssystem-Konzepte.

Im folgenden Abschnitt sind die Anforderungen zusammengefasst aufgelistet und im *Abschnitt 3.8.3* ist der Ablauf des Prototyps beschrieben.

3.8.1 Anforderungen an den Prototyp:

- **Statische Daten**
  Der Prototyp soll die Verarbeitung von statischen Daten ermöglichen.
- **Daten-Import**
  Die Daten von Hilti sollen direkt importieret werden können
- **Datei-Export**
  Der Prototyp soll alle Resultate in ein CSV-File abspeichern.
- **Empfehlungssystem Algorithmus**
  Der Algorithmus des Empfehlungssystems soll bei Bedarf ausgetauscht werden können, ohne dass grosse Änderungen am Prototyp durchgeführt werden müssen.
- **Clustering-Verfahren**
  Das Clustering Verfahren soll bei Bedarf gewechselt werden können.
- **Automatisierung**
  Der Prototyp soll über Parameter-Listen konfiguriert werden können.

3.8.2 Verwendete Software

Für die vorliegende Arbeit wurde die unten aufgeführte Software eingesetzt.

- **Datenbank**
  MySQL v5.6.20
- **MATLAB**
  MATLAB R2014a (8.3.0.532) 64bit
- **Betriebssystem**
  Windows Server 2012 R2

3.8.3 Ablauf Empfehlungsdienst

Anhand des in *Abbildung 9* dargestellten Ablaufs, werden im folgenden Abschnitt die wichtigsten Funktionen, bezogen auf die erarbeiteten Vorgehensmodelle, beschrieben. Die Funktionen wurden mit dem Dateinamen und Dateiendung *.m* beschriftet, wobei diese Dateiendung eine MATLAB-Datei definiert. Im Anhang in *Abschnitt 7.3* ist der genaue Ablauf in einem Sequenzdiagramm dargestellt.

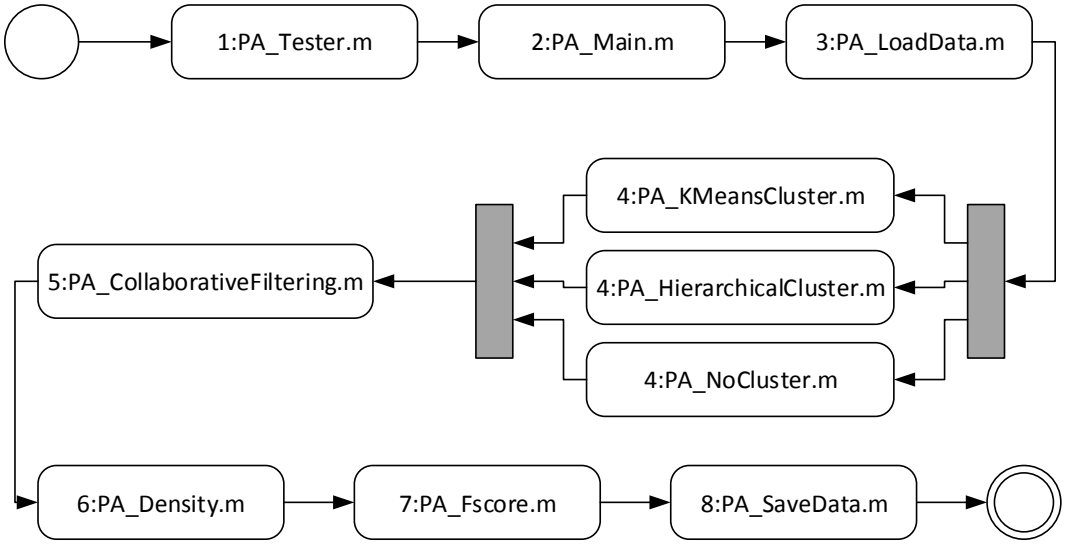

Abbildung 9 Ablauf Empfehlungsdienst

- **PA_Tester.m**
  Ruft das Hauptprogramm (PA_Main.m) mit den vorkonfigurierten Parameter auf.
- **PA_Main.m**
  Ist das Hauptprogramm und startet die weitern Funktionen der Reihen nach auf.
- **PA_LoadData**
  Lädt die **.csv* –Dateien, welche die Benutzerbewertungen und die Validierungen enthalten und speichert diese in Matrizen ab.
- **PA_KMeansCluster.m**
  Führt das Clustering-Verfahren k-Mean durch.
- **PA_HierarchicalCluster.m**
  Führt ein hierarchisches Clustering-Verfahren durch.
- **PA_NoCluster.m**
  Führt kein Clustering-Verfahren durch.
- **PA_CollaborativeFiltering.m**
  Berechnet die **Prognosematrix** mithilfe eines Collaborative Filtering Hybrid Algorithmus.
- **PA_Density.m**
  Berechnet die Dichte der Matrix.
- **PA_Fscore.m**
  Berechnet die Erfolgsgüte.
- **PA_SaveData**
  Speichert die Ergebnisse in eine *.csv-Datei.

Kapitel 4 Experimente

In diesem Kapitel werden die Ausgangslagen und Ziele der durchgeführten Experimente beschrieben. Die Resultate der Experimente werden direkt nach dem jeweiligen Experiment beschrieben und diskutiert. Alle Experimente wurden im MATLAB auf einen virtuellen Windows 2012 R2 Server durchgeführt.

Der Ablauf ist in *Abbildung 10* dargestellt.

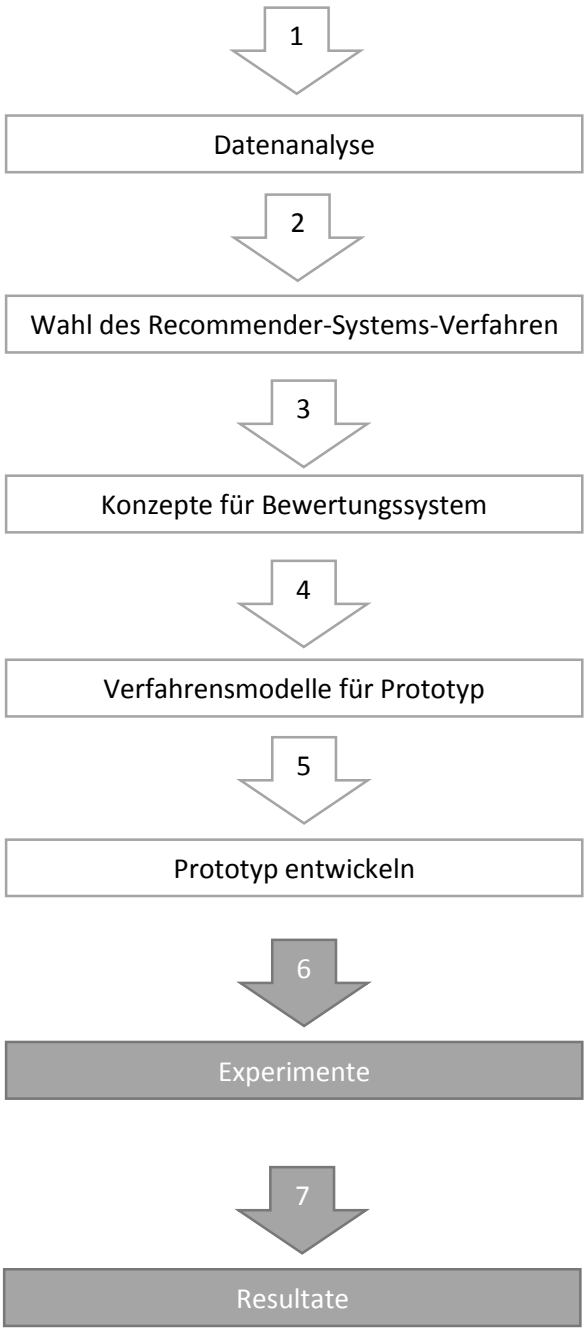

Abbildung 10 Überblick

4.1 Experiment 1

4.1.1 Experiment 1: Dichteberechnung

- **Ausgangslage**

Das erste Experiment beinhaltet zwei verschiedenen Arten der Dichteberechnung der Bewertungsmatrix, denn mithilfe einer dichteren Bewertungsmatrix, kann die Erfolgsgüte erhöht werden (*vgl. Abschnitt 3.6.1*).

- **Ziel**

Das Ziel des ersten Experiments ist die Verifikation der Vermutung von *Abschnitt 3.6.1*.

- **Testparameter**

In der *Tabelle 18* stehen die Eckdaten zur Berechnung der Bewertungsmatrix-Dichte, dabei wurden nur die Produkte und C-Klassen berücksichtigt, die sich in den A-Klassen *Tools* und *Consumables* befinden (*vgl. Abschnitt 1.5*).Die Werte stammen aus der Tabelle Hilti_dataset_training.

Beschreibung	*Wert*
Anzahl Kunden	28‘587
Anzahl Produkte	713
Summe der verschiedenen Produkte	299‘660
Anzahl C-Klassen:	190
Summe der verschiedenen C-Klassen	212‘538

Tabelle 18 Testparameter Experiment 1

Legende für Tabelle 18

- Summe der verschiedenen Produkte:
  Ist die Summe der gekauften Produkte aller Kunden. Falls vom gleichen Kunden ein Produkt mehrmals gekauft wurde, werden die weiteren Käufe nicht berücksichtig.
- Summe der verschiedenen C-Klassen:
  Ist die Summe der C-Klassen der gekauften Produkte aller Kunden. Falls vom gleichen Kunden ein Produkt einer C-Klasse mehrmals gekauft wurde, werden die weiteren Käufe nicht berücksichtig.

4.1.2 Resultate von Experiment 1

In diesem Experiment wurde die Dichte der Bewertungsmatrix auf zwei verschiedenen Arten berechnet. Zuerst wurde die Dichte der Bewertungsmatrix auf Stufe von Produkten und dann auf Stufe von C-Klassen berechnet.

1. Produkte Bewertungsmatrix:

$$Dichte(Produkte) = \frac{299'660}{713 * 28'587} = \mathbf{0.0147}$$

Die Dichte der Bewertungsmatrix für Produkte beträgt ca. 1.47%.

2. C-Klassen Bewertungsmatrix:

$$Dichte(C - Klassen) = \frac{212'538}{190 * 28'587} = \mathbf{0.0391}$$

Die Dichte der Bewertungsmatrix für C-Klassen beträgt ca. 3.91%.

4.1.3 Auswertung

Die Dichte auf Stufe von C-Klassen ist mehr als doppelt so gross wie die Dichte auf Stufe von Produkten, dennoch ist die Dichte in beiden Fällen gering. Die geringe Dichte lässt sich dadurch Begründen, dass viele Produkte nicht gekauft wurden.

Zusammenfassend bestätigt das erste Experiment unsere Vermutung vom *Abschnitt 3.6.1*, dass bei einer Bewertungsmatrix auf Stufe von C-Klassen eine höhere Dichte erzielt werden kann als auf Stufe von Produkten.

4.2 Experiment 2

4.2.1 Experiment 2: Ohne Clustering

- **Ausgangslange**

Das zweite Experiment beinhaltet Tests über die erarbeiteten Konzepte die im *Abschnitt 3.6* beschrieben wurden. In diesem Experiment sollen noch keine Clustering Verfahren angewendet werden.

- **Ziel**

Es soll ermittelt werden, welches der erarbeiteten Konzepte zur Generierung von Bewertungen die besten Resultate erzielt.

- **Testparameter**

Für die Anzahl Eigenschaften gibt es in der Literatur keinen Richtwert. Aus diesem Grund wurden für das zweite Experiment die Eigenschaften von 1 bis 20 gewählt und für die Anzahl Empfehlungen wurden Werte zwischen 2 und 10 gewählt. Für den Lambda-Wert haben wir uns bei diesem Experiment an dem Online-Kurs von Adrew Ng orientiert, dieser verwendet in seinen Beispielen Lambda Werte unter 10.

Beschreibung	*Wert*
Anzahl Eigenschaften	1 bis 20
Anzahl Empfehlungen	2,5,7,10
Lambda	1 bis 10
Bewertungssystem	Konzepte 1 bis 3

Tabelle 19 Testparameter Experiment 2

In der *Tabelle 19* sind die Testparameter aufgelistet. Zuerst soll untersucht werden, wie sich der Algorithmus auf den Daten von Hilti verhält.

4.2.2 Resultate von Experiment 2

Im *Diagramm 1* sind die Resultate des zweiten Experiments zusammengefasst. Die dargestellten Werte F1, Precision und Recall entsprechen jeweils den höchsten beobachteten Werten, wobei die Anzahl Eigenschaften und Lambda nicht dargestellt sind.

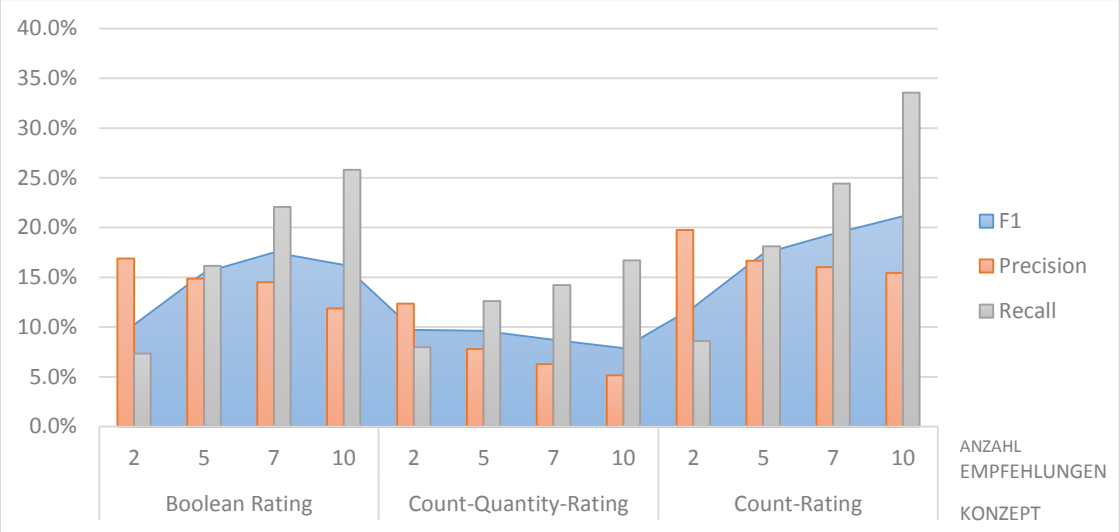

Diagramm 1 Vergleich der Bewertungssystem-Konzepte

- **Beschreibung**

Im obigen Diagramm sind die Precision- und Recall-Werte als Säulen und die F1-Werte als Fläche dargestellt. Das Diagramm verfügt über zwei horizontale Achsen, wobei die obere Achse die Anzahl Empfehlungen und die untere Achse die verschiedenen Konzepte anzeigt. Die vertikale Achse zeigt in Prozent die Werte von F1, Precision und Recall an.

4.2.3 Auswertung

Der beste Testdurchlauf wurde mit dem Konzept 2, dem Count-Rating (*Abschnitt 3.6.4*), bei zehn Empfehlungen und 20 gelernten Eigenschaften, erreicht. Der beste Wert für F1 beträgt dabei 21.4% mit einer Precision 15.6% und einem Recall 34%.

4.3 Experiment 3

4.3.1 Experiment 3: Mit Clustering

- **Ausgangslange**

Dieses Experiment basiert auf den Ergebnissen des zweiten Experiments. Das Konzept mit den besten Ergebnissen im zweiten Experiment soll mit Hilfe von Clustering-Verfahren genauer untersucht werden.

- **Ziel**

Das Ziel des dritten Experiments ist die Ermittlung des Clustering-Verfahrens, welches die besten Resultate generiert.

- **Testparameter**

In diesem Experiment sollen die ersten vier Testparameter gleich wie im Experiment 2 gewählt werden. Die Anzahl der Cluster soll in diesem Experiment nicht zu gross gewählt werden, damit die Einteilung der Benutzer einfacher nachvollziehbar ist.

Beschreibung	*Wert*
Anzahl Eigenschaften	1, 10, 20
Anzahl Empfehlungen	2,5,7,10
Lambda	1 bis 10
Bewertungssystem	Konzepte 1 bis 4
Clustering-Verfahren	k-Mean und Hierarchische Clusteranalyse
Anzahl Cluster	2, 4, 8, 10

Tabelle 20 Testparameter Experiment 3

4.3.2 Resultate von Experiment 3

Im *Diagramm 2* sind die Resultate des dritten Experiments zusammengefasst. In diesem Experiment wurde als Bewertungssystem, dass im Konzept 2 (Count-Rating) vorgestellte System verwendet. Dieses Bewertungssystem erzielte im Experiment 2 die besten Resultate.

Die dargestellten Werte F1, Precision und Recall entsprechen jeweils den höchsten beobachteten Werten, wobei die Anzahl Eigenschaften und Lambda nicht dargestellt sind.

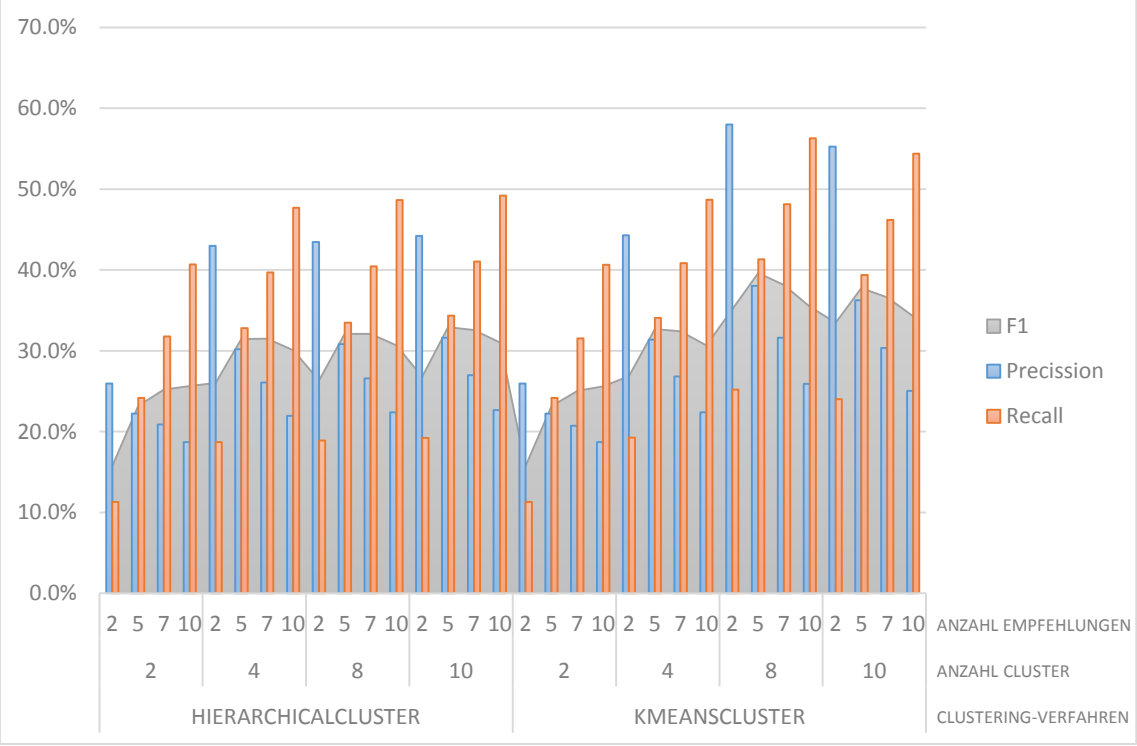

Diagramm 2 Vergleich Clustering-Verfahren für Konzept 2

- **Beschreibung**

Im obigen Diagramm sind die Precision- und Recall-Werte als Säulen und die F1-Werte als Fläche dargestellt. Das Diagramm verfügt über drei horizontale Achsen, wobei die oberste Achse die Anzahl Empfehlungen, die mittlere Achse die Anzahl Cluster und die unterste Achse die Clustering-Verfahren anzeigt. Die vertikale Achse zeigt in Prozent die Werte von F1, Precision und Recall an.

4.3.3 Auswertung

Die besten Ergebnisse wurden mit dem k-Means Clustering-Verfahren mit fünf Clustern, bei sieben Empfehlungen und einer gelernten Eigenschaft erzielt. Der beste Wert für F1 beträgt dabei 39.6% mit Precision 38.0% und Recall 41.3%.

4.4 Experiment 4

4.4.1 Experiment 4: Clusteranalyse

- **Ausgangslange**

Dieses Experiment basiert auf den Ergebnissen des dritten Experiments. Das Clustering-Verfahren mit der besten Erfolgsgüte soll genauer analysiert werden.

- **Ziel**

Das Ziel des vierten Experiments ist die Ermittlung der optimalen Cluster-Anzahl.

- **Testparameter**

In diesem Experiment sollen die ersten vier Testparameter gleich wie im Experiment 2 gewählt werden. Für die experimentelle Ermittlung der optimalen Anzahl der Cluster werden zwischen 2^0 und 2^6 Clustern gewählt.

Beschreibung	*Wert*
Anzahl Eigenschaften	1, 5, 10, 20, 25, 30
Anzahl Empfehlungen	2,5,7,10
Lambda	1, 5, 10
Bewertungssystem	Konzept 2 (Count-Rating)
Clustering-Verfahren	k-Mean Clusteranalyse
Anzahl Cluster	1,2,4,8,10,16,32,33,40,48,64

Tabelle 21 Testparameter Experiment 4

4.4.2 Resultate von Experiment 4

Im *Diagramm 3* sind die Resultate des vierten Experiments zusammengefasst. In diesem Experiment wurde empirisch geprüft welche Anzahl an Cluster die Erfolgsgüte (F1-Mass) erhöht. Als Clustering-Verfahren wurde das k-Mean-Clustering verwendet, da es die besten Resultate im dritten Experiment erzielte.

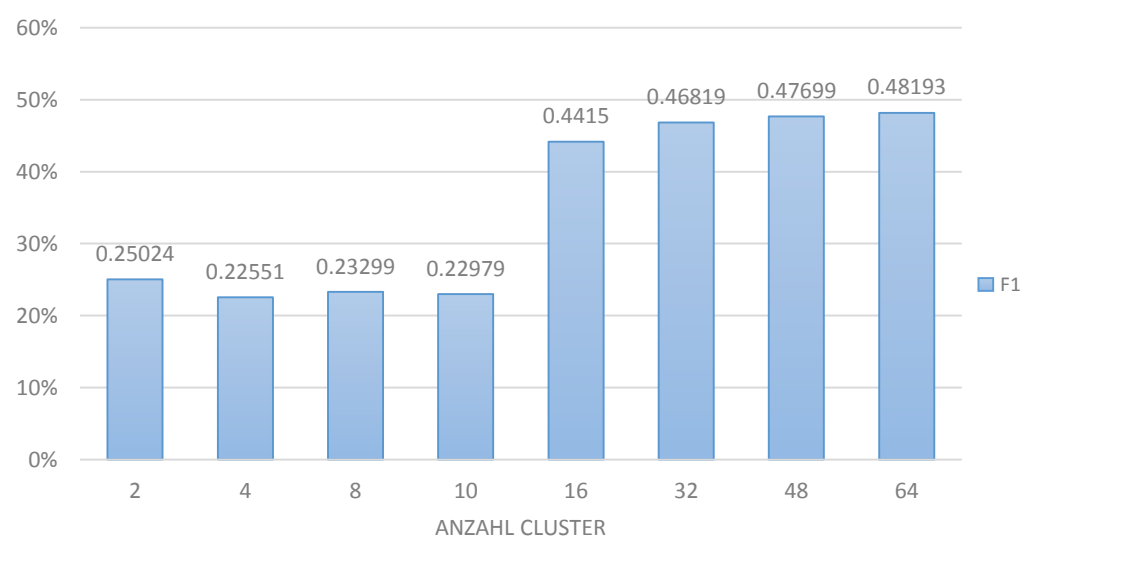

Diagramm 3 F1 für verschiedene Cluster

- **Beschreibung**

Das *Diagramm 3* stellt die besten erzielten F1-Werte für verschiedene Anzahl Cluster in Prozent dar. In der horizontalen Achse ist die Anzahl der Clusters dargestellt.

4.4.3 Auswertung

Im *Diagramm 3* ist ersichtlich, dass sich die Erfolgsgüte ab 16 Clustern signifikant verbessert. Bei 16 Clustern beträgt die Erfolgsgüte 44.15%.

4.5 Zusammenfassung der Experimente

Für die experimentelle Untersuchung des Collaborative Filtering Hybrid Verfahrens wurden in diesem Kapitel vier aufeinander aufbauende Experimente durchgeführt.

In diesem Abschnitt werden die Resultate der vier Experimente zusammengefasst.

Das Resultat des ersten Experiments ist die Erkenntnis, dass die Dichte der Bewertungsmatrix auf Stufe von C-Klassen doppelt so gross ist wie auf Stufe von Produkten. Die Dichte der Bewertungsmatrix auf Stufe von C-Klassen beträgt beinahe 4%. Aus der Erkenntnis des ersten Experiments wurde im zweiten Experiment, für die Untersuchung der Leistungsfähigkeit der erarbeiten Konzepte für ein implizites Bewertungssystem, die Bewertungsmatrix auf Stufe von C-Klassen generiert. Die besten Ergebnisse wurden mit dem Konzept Count-Rating erzielt. Im dritten Experiment wurden zwei Clustering-Verfahren untersucht und festgestellt, dass mit dem k-Mean-Clustering auf den gegeben Daten von Hilti bessere Resultate erzielt werden konnten. Folglich wurde das Clustering-Verfahren im vierten Experiment untersucht, um die optimale Cluster-Anzahl ermitteln zu können. Das Resultat des vierten Experiments ist eine Cluster-Anzahl von 16 Clustern.

4.6 Optimierung

In den vier Experimenten wurde festgestellt, dass die besten Resultate mit einer Bewertungsmatrix auf Stufe von C-Klassen, mit einem implizierten Bewertungssystem nach dem Konzept von Count-Rating, sowie mit dem Clustering-Verfahren k-Mean und mit einer Cluster-Anzahl von 16 erzielt werden konnten.

Für die Optimierung des Collaborative Filtering Hybrid Verfahrens werden in diesem Abschnitt verschiede Konfigurationsparameter (Anzahl Eigenschaften und Anzahl Empfehlungen) untersucht.

4.6.1 Untersuchung der Erfolgsgüte bei 16 Clustern

Im *Diagramm 4* sind die Ergebnisse mit 16 Clustern dargestellt.

Die verschiedenen Anzahl Eigenschaften und Anzahl Empfehlungen wurden durch eine empirische Untersuchung ermittelt. Bei der Untersuchung wurde festgestellt, dass eine kleinere Anzahl Eigenschaften als 64 zu kleineren Werten für das F1-Mass führt. Für die vier verschiedenen Anzahl Eigenschaften wurden je ein bis vier C-Klassen empfohlen.

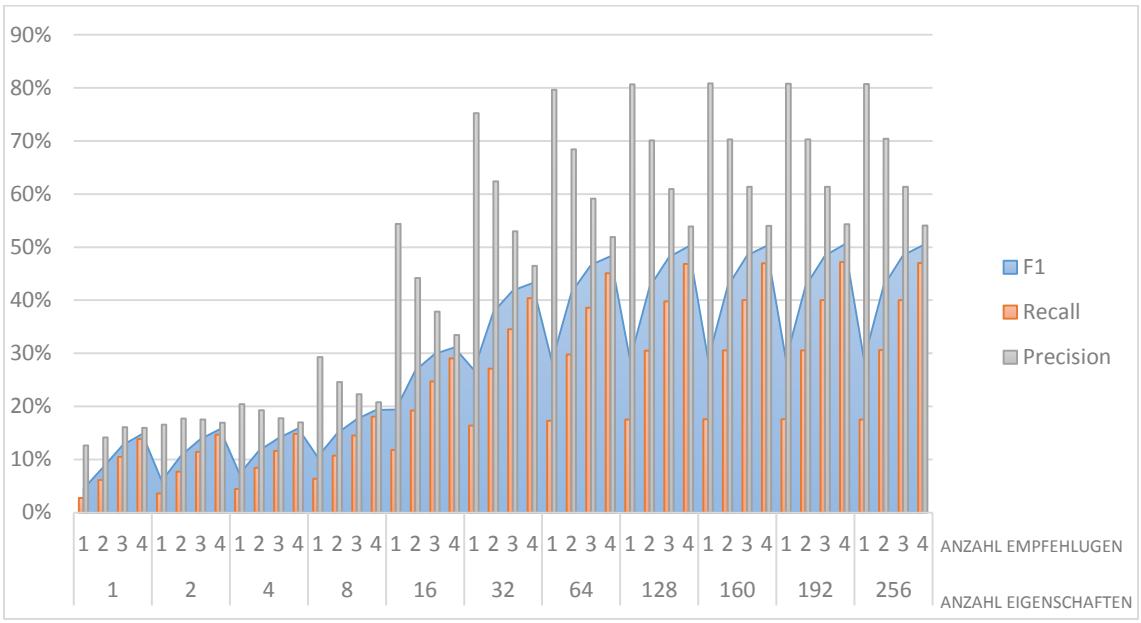

Diagramm 4 Untersuchung der Erfolgsgüte bei 16 Clustern

- **Beschreibung**

Im *Diagramm 4* sind die Precision- und Recall-Werte als Säulen und die F1-Werte als Fläche dargestellt. Das Diagramm verfügt über zwei horizontale Achsen, wobei die obere Achse die Anzahl Empfehlungen und die untere Achse die Anzahl Eigenschaften anzeigt. Die vertikale Achse zeigt in Prozent die Werte von F1, Precision und Recall an.

4.6.2 Auswertung

Im Diagramm 4 ist ersichtlich, dass das F1-Mass mit der Vergrösserung der Anzahl Eigenschaften zunimmt. Ab 128 Eigenschaften verändert sich das F1-Mass nur geringfügig.
Unabhängig von der Anzahl Eigenschaften, nehmen das F1-Mass und die Recall-Werte jeweils pro Empfehlung zu. Bei den Precision-Werten ist erst ab 16 Eigenschaften eine Regelmässigkeit erkennbar. Ab 16 Eigenschaften nehmen die Precision Werte pro Empfehlung ab.

4.6.3 Fazit Optimierung

Das Collaborative Filtering Hybrid Verfahren konnte durch die Optimierung der Konfigurationsparameter signifikant verbessert werden. Bei 128 Eigenschaften und 4 Empfehlungen konnte ein F1-Wert von 50.1% sowie Recall von 46.9% und Precision von 53.9% erreicht werden.

4.7 Clusteranalyse

In diesem Abschnitt wird das Clustering-Verfahren k-Mean analysiert, da die höchste Erfolgsgüte mit diesem Clustering-Verfahren erreicht werden konnte.
Zuerst wird im *Abschnitt 4.7.1* die Benutzerverteilung untersucht. Danach wird im *Abschnitt 4.7.2* die Zuordnungsstärke des Clustering mithilfe des Silhouette-Koeffizienten untersucht und im *Abschnitt 4.7.3* wird die Dichte der einzelnen Cluster verglichen.

4.7.1 Benutzerverteilung bei 16 Clustern

Für eine genauere Analyse wurden die absoluten Häufigkeiten der Benutzer in den verschiedenen Clustern berechnet und visualisiert.

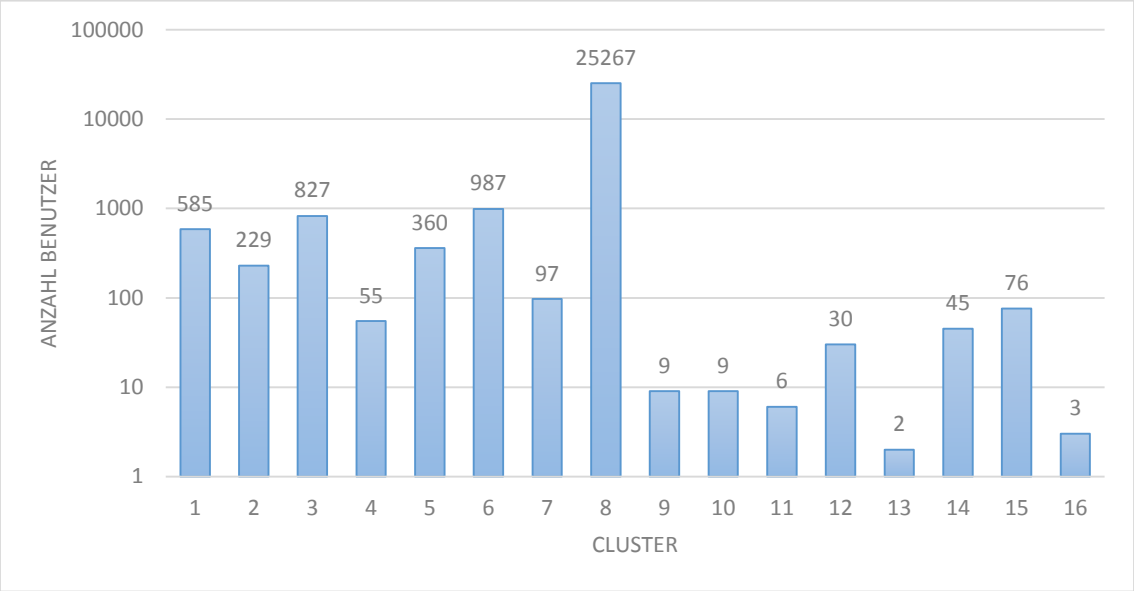

Diagramm 5 Benutzerverteilung bei 16 Clustern

- **Beschreibung**

Im *Diagramm 5* ist die Verteilung der Benutzer in den 16 Clustern dargestellt. In der horizontalen Achse sind die einzelnen Cluster und in der vertikalen Achse die Anzahl Benutzer in logarithmischer Skala dargestellt.

4.7.1.1 Auswertung

Im obigen Diagramm ist ersichtlich, dass sich der Grossteil der Benutzer im achten Cluster befindet. Der Anteil der Benutzer im achten Cluster beträgt ca. 88% aller Benutzer.

4.7.2 Zuordnungsstärke des Clustering

In diesem Abschnitt wird die Zuordnungsstärke des Clustering mithilfe des Silhouette-Plots und des Silhouette-Koeffizient untersucht.

Als Distanzmass bzw. Ähnlichkeitsmass wurden der euklidische Abstand, der Jaccard-Koeffizent und der Cosinus-Koeffizient verwendet. Aufgrund der dünnbesetzten Bewertungsmatrix, konnte mit dem Cosinus-Koeffizienten als Ähnlichkeitsmass in MATLAB kein Ergebnis erzielt werden.

4.7.2.1 Silhouette-Plot 1 – Euklidischer Abstand

In der *Abbildung 11* wurde der Silhouette-Value für jedes Cluster visuell dargestellt. Bei diesem Plot wurde der euklidische Abstand (*vgl. Abschnitt 2.9.3*) als Distanzmass verwendet.

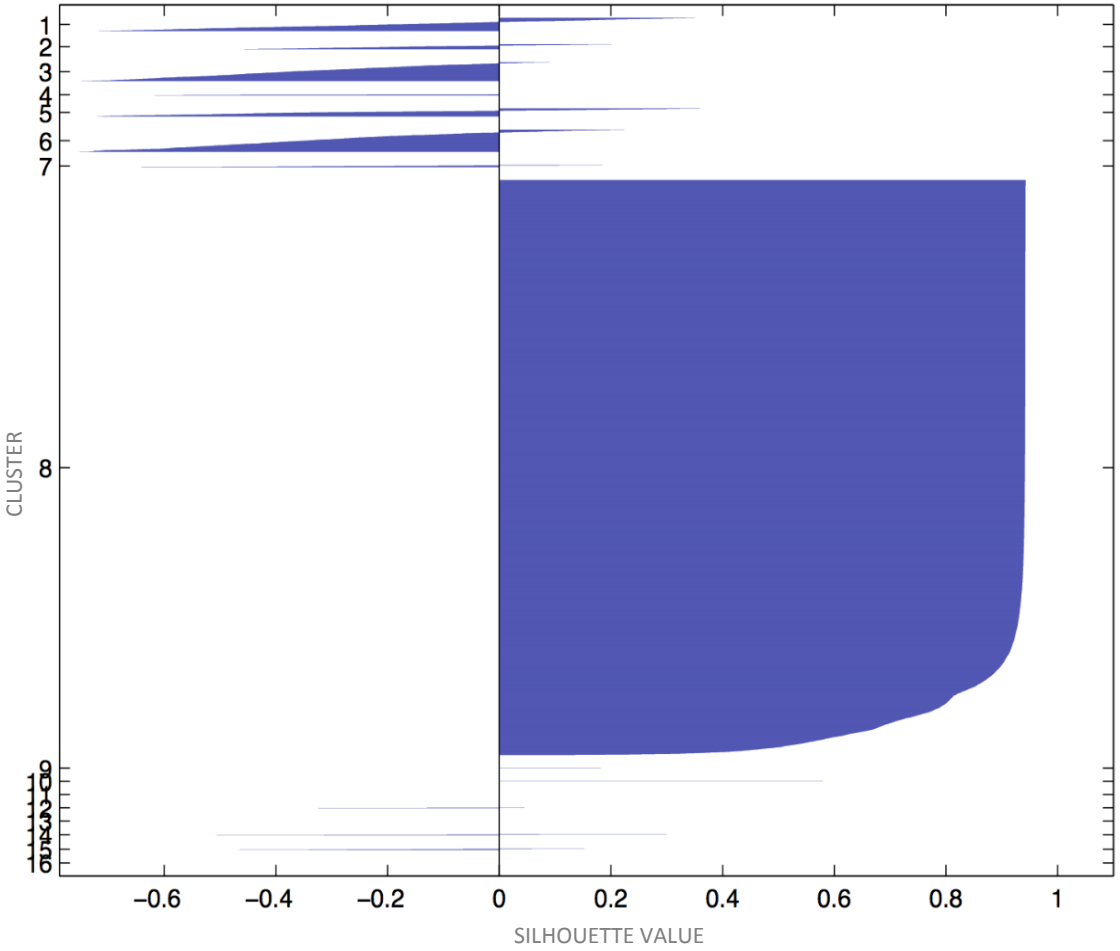

Abbildung 11 Silhouette-Plot

- **Beschreibung**

Die obige Abbildung zeigt den Silhouette-Value jedes Benutzers im Cluster. Die vertikale Achse ist in den 16 Clustern eingeteilt und jedes Cluster ist in der Mitte beschriftet. Die horizontale Achse stellt den Silhouette-Value dar.

4.7.2.2 Auswertung

In der *Abbildung 11* ist ersichtlich, dass der Grossteil der Silhouette-Value über 0.8 beträgt. Folglich verfügt das Clustering über eine gute Zuordnung (*vgl. Abschnitt 2.9.3*). Aufgrund der guten Zuordnung im Cluster 8 beträgt der Silhouette-Koeffizient über 0.769, dementsprechend verfügt das Clustering über eine starke Struktur (*vgl. Abschnitt 2.9.3*).

4.7.2.3 Silhouette-Plot 2 – Jaccard-Koeffizient

In der *Abbildung 12* wurde der Silhouette-Value für jedes Cluster visuell dargestellt. Dabei wurde als Ähnlichkeitsmass der Jaccard-Koeffizient (*vgl. Abschnitt 2.9.3*) verwendet.

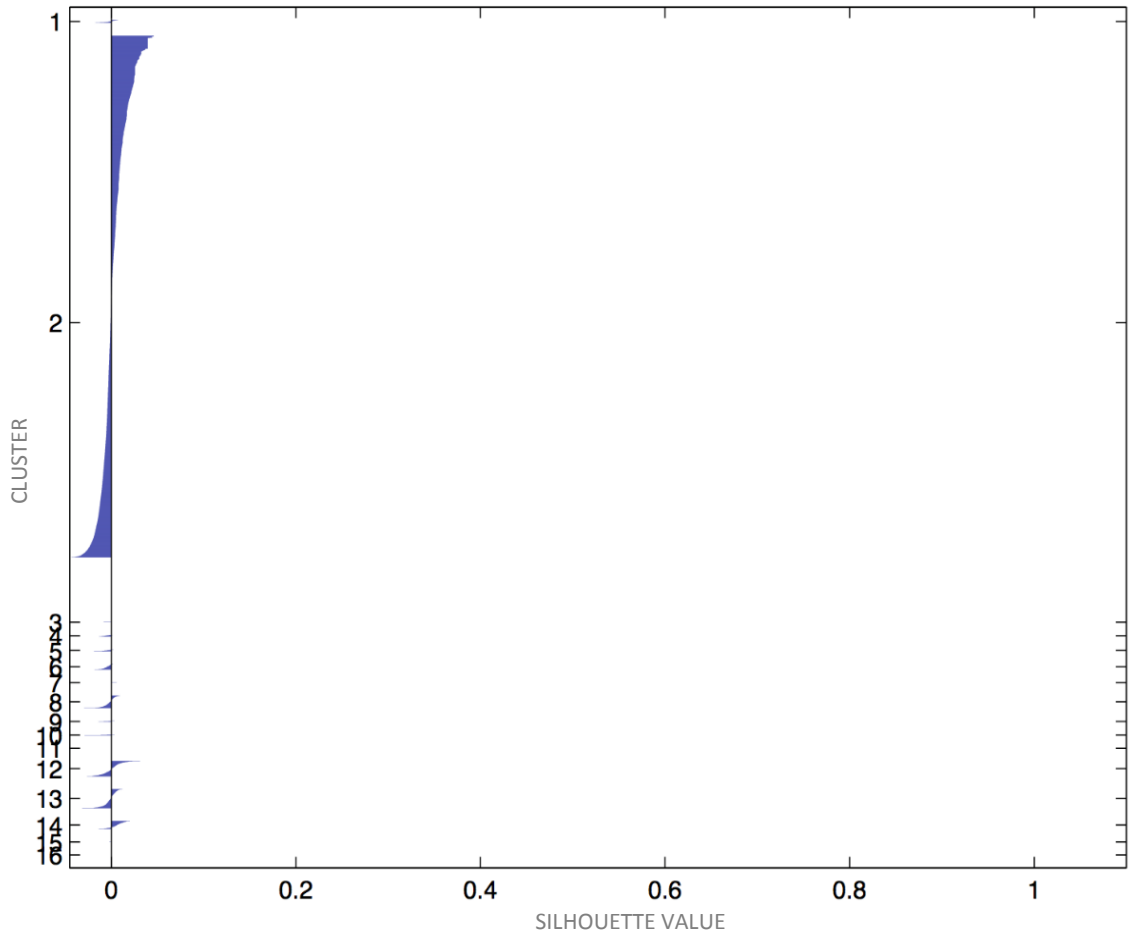

Abbildung 12 Silhouette Plot

- **Beschreibung**

Die obige Abbildung zeigt den Silhouette-Value jedes Benutzers im Cluster. Die vertikale Achse ist in den 16 Clustern eingeteilt und die horizontale Achse zeigt den Silhouette-Value.

4.7.2.4 Auswertung

In der *Abbildung 12* ist ersichtlich, dass kein Silhouette-Value über 0.2 vorhanden ist. Daraus folgt, dass ein Clustering basierend auf dem Ähnlichkeitsmass Jaccard-Koeffizient keine Struktur aufweist.

4.7.3 Dichteanalyse bei 16 Clustern

Im *Diagramm 6* wird die Dichte der einzelnen Cluster mit der Anzahl der Benutzer im Cluster verglichen.

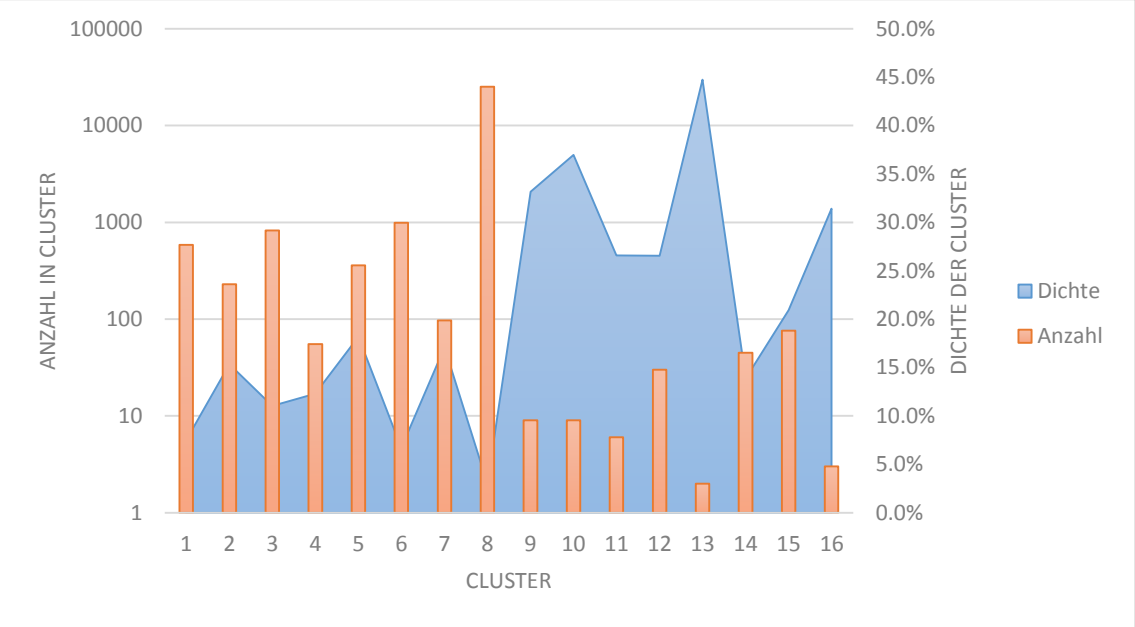

Diagramm 6 Dichte pro Cluster

- **Beschreibung**

Das obige Diagramm verfügt über zwei vertikale Achsen, die linke vertikale Achse stellt die Anzahl der Kunden im Cluster in logarithmischer Skala dar und die rechte vertikale Achse stellt die Dichte der Cluster in Prozent dar.

4.7.3.1 Auswertung

Im Diagramm ist ersichtlich, dass Cluster mit weniger Benutzer eine unverkennbar höhere Dichte aufweisen, als Cluster mit vielen Benutzer.

4.7.4 Fazit Clusteranalyse

Die Analyse der Benutzerverteilung ergab, dass ca. 88% aller Benutzer dem Cluster 8 zugeordnet wurden (*vgl. Diagramm 5*). Die Benutzer in diesem Cluster haben alle einen Silhouette-Value von ca. 0.8 (*vgl. Abbildung 11*), somit verfügt das Cluster 8 über eine gute Zuordnung. Die Dichte in Cluster 8 ist mit 3% kleiner als die Dichte der ursprünglichen Bewertungsmatrix ohne Clustering. Ein Grossteil der Benutzer in den übrigen Clustern haben Silhouette-Values die kleiner als 0 sind. Dies bedeutet, dass die Zuordnung der Benutzer in diesen Clustern schlecht ist. Die Dichte in diesen Clustern liegt zwischen 7% und 45% und somit deutlich höher als die des Cluster 8.

Zusammenfassend konnten mithilfe des Clustering die Ausreisser von den übrigen Benutzern separiert werden.

Kapitel 5 Diskussion und Ausblick

5.1 Zusammenfassung

Aus der Datenanalyse folgten wichtige Erkenntnisse:

- Es existieren keine expliziten Bewertungen der Produkte durch die Benutzer.
- Zwischen den Einkäufen und den Interaktionen mit dem Kundendienst wurden keine Zusammenhänge gefunden.

Basierend auf der Datenanalyse wurde ein Collaborative Filtering Hybrid Verfahren für den Prototyp verwendet, da weder Eigenschaften der Produkte noch Präferenzen der Kunden bekannt waren. Für die Umsetzung des Verfahrens wurden verschiedene Konzepte für ein implizites Bewertungssystem entwickelt und anschliessend mit Hilfe des Prototyps getestet. Aus diesen Experimenten ging hervor, dass das Count-Rating die besten Ergebnisse lieferte.

Zur weiteren Verbesserung der Ergebnisse wurde das für den Prototyp verwendete Verfahren durch ein Clustering-Verfahren für die Benutzer erweitert, dazu wurden, wie in *Abschnitt 4.3* beschrieben, zwei Verfahren getestet. In den Experimenten lieferte k-Mean die besseren Ergebnisse.

Abschliessend wurden experimentell die besten Werte für die Anzahl Cluster und die Anzahl Eigenschaften ermittelt. Die daraus resultierenden Ergebnisse und die dazu verwendeten Konfigurationsparameter werden im nächsten Abschnitt beschrieben.

5.2 Ergebnisse

Gute Ergebnisse wurden in den Experimenten mit den folgenden Parametern erreicht:

Parameter	*Wert*
Bewertungsmatrix	Count-Rating (*vgl. Abschnitt 4.4.3.2*)
Cluster-Algorithmus	k-Mean
Anzahl Cluster	16
Anzahl Eigenschaften	64

Tabelle 22 Parameter für gute Ergebnisse

Mit grösseren Werten für die Anzahl der Cluster und Eigenschaften stagnierte die Erfolgsgüte, damit das System möglichst einfach blieb und um die Gefahr der Überspezifikation des Systems zu verringern, wurden die kleinsten Werte verwendet, die eine hohe Erfolgsgüte aufwiesen, d.h. ein F1-Mass von ca. 50% erreichten.

Mit den Parametern aus der *Tabelle 22* erreichte der Prototyp die folgenden Werte:

Anzahl Empfehlungen	*Precision*	*Recall*	*Interpretation*
1	79,6%	17.3%	Für fast 80% aller Kunden war die vorgeschlagene Klasse von Interesse, allerdings wurden nur 17.3% aller Klassen empfohlen, die für die Benutzer von Interesse gewesen wären.
4	51.9%	45.1%	Mehr als die Hälfte aller empfohlenen Klassen waren für die Benutzer von Interesse und es wurden 45.1% aller Klassen empfohlen, die für die Benutzer von Interesse gewesen wären.

Tabelle 23 Interpretierte Ergebnisse bei 16 Clustern und 64 Eigenschaften

In *Tabelle 23* werden die erreichten Werte interpretiert. Bei einer Integration auf HOL und der damit einhergehenden Empfehlung von Produkten anstelle von C-Klassen ist mit einer verringerten Erfolgsgüte zu rechnen, da die Zuteilung von Produkten zu Empfohlenen C-Klassen wie in *Abschnitt 5.5.2* beschrieben, zusätzliche Ungenauigkeiten generiert.

5.3 Rückblick auf die Aufgabenstellung

„Entwicklung eines Prototyps eines Recommender-System für HOL, basierend auf anonymisierten Verkaufs-, Kunden- und Produktmasterdaten.“

Das Resultat der vorliegenden Arbeit ist ein Prototyp eines Recommender-System in MATLAB.

Im Folgenden werden die Anforderungen von Hilti aus *Abschnitt 1.5* mit der erarbeiteten Lösung verglichen.

- **Empfehlung von Produkten aus den Klassen *Tools* und *Consumables***

Der entwickelte Prototyp erfüllt die Anforderung nicht vollständig, denn der Prototyp wurde so entwickelt, dass er Empfehlungen auf Stufe von C-Klassen liefert und nicht auf Stufe von Produkten.

- **Statische Daten**

Der entwickelte Prototyp erfüllt diese Anforderung. Der Prototyp kann statische Daten verarbeiten.

5.4 Vermutungen

Aus der Analyse der Cluster in *Abschnitt 4.7* insbesondere dem Fazit in *Anschnitt 4.7.4* ergeben sich folgende Vermutungen:

- Die hohen Silhouette-Values der Benutzer in Cluster 8 führen zur Vermutung, dass die Benutzer welche diesem Cluster zugeordnet sind, alle Produkte aus ähnlichen C-Klassen gekauft haben.
- Die Cluster 1-7 und 9-16 haben eine hohe Dichte, bei gleichzeitig niedrigen Silhouette-Values, deswegen liegt die Vermutung nahe, dass die Benutzer in diesen Clustern viele Produkte aus verschiedenen C-Klassen gekauft haben und die Übereinstimmung der gekauften Produkte unter den Benutzern gering ist.

5.5 Weiterführende Arbeiten

5.5.1 Alle Daten aus *Hilti_dataset_training* verwenden

In der vorliegenden Arbeit werden nur die Kaufdaten für die IPCClassdistinct *Tools* und *Consumables* verwendet. Zur Verbesserung der Ergebnisse sollte untersucht werden, wie die bisher ungenutzten Daten aus dieser Tabelle verwendet werden können.

5.5.2 IPC-Klasse-Produkt

Der Prototyp arbeitet auf der Stufe von IPC-Klassen, um dem Kunden Produkte zu empfehlen, muss der Prototyp erweitert werden. Ein Ansatz dazu wäre es die meist verkauften Produkte pro IPC-Klasse zu speichern und anstelle der IPC-Klassen zu empfehlen. Eine Möglichkeit bessere Ergebnisse zu erhalten wäre, das beste Produkt pro Kunden-Cluster zu ermitteln.

5.5.3 Clustering-Verfahren

Die Aufteilung der Kunden in verschiedene Cluster spielt im vorliegenden System eine wichtige Rolle, es wurden jedoch lediglich zwei Clustering-Verfahren getestet und für den verwendeten Algorithmus k-Mean sollten die Vermutungen in *Abschnitt 5.4* überprüft werden.

Tabelle *Hilti_dataset_activities_training* erneut untersuchen

Zur weiteren Verbesserung der Kunden-Cluster sollten die Interaktionen der Kunden mit dem Kundendienst weiter untersucht werden, um Ähnlichkeiten zwischen den Kunden zu finden.

Distanzmasse

In den Experimenten wurden drei Distanzmasse getestet, das euklidische, der Jaccard-Koeffizient und der Cosinus-Koeffizient. Weitere Distanzmasse für die Distanz basierten Clustering-Verfahren sollten getestet werden.

Dichte basierte Cluster

Ein Dichte basiertes Clustering-Verfahren sollte implementiert und getestet werden, dabei wird die Zugehörigkeit eines Kunden zu einem Cluster durch die Maximierung der Dichte der Cluster ermittelt.

Weitere Clustering-Verfahren

Weitere Verfahren, um die Benutzer in Cluster aufzuteilen, sollten untersucht und getestet werden.

5.5.4 Integration auf HOL

Der Prototyp arbeitet mit statischen Daten, zur Integration auf HOL benötigt es noch einige Anpassungen.

Handhabung neuer Kunden

Es braucht Verfahren um neu Kunden in das System integrieren zu können. Dies könnte z.B. durch die Normalisierung der Daten erreicht werden. Dabei werden die Bewertungen der Kunden normalisiert, was zur Folge hat, dass Kunden die noch kein Produkt gekauft haben, vom System die Produkte empfohlen bekommt, die bei den anderen Kunden am beliebtesten sind. Dadurch ist es möglich auch für neue Kunden sinnvolle Empfehlungen zu geben.[1]

Handhabung neuer Produkte

Es müssen Verfahren gesucht werden um neue Produkte in das System zu integrieren.

Erfassen von Verhaltensdaten

Bisher wird für die Generierung der Bewertung nur das Kaufverhalten der Kunden verwendet. Ein Weg das System weiter zu verbessern wäre, mehr Informationen über das Verhalten der Kunden, wie Produkte die sich ein Kunde ansieht, ohne sie zu kaufen, zu sammeln und in das implizite Bewertungssystem zu integrieren.

Kapitel 6 Verzeichnisse

6.1 Literaturverzeichnis

[1] Bundesamt für Statistik BFS. (10.03.2014). *Informationsgesellschaft – Indikatoren. Unternehmen - E-Commerce* [Online].
URL: http://www.bfs.admin.ch/bfs/portal/de/index/themen/16/04/key/approche_globale.indicator.30204.302.html?open=1,2,10,313,327#327 [Stand: 13.12.14]

[2] Hilti. (11.07.2014). *The 2015 Hilti Big Data Analytics Competition [Online].*
URL:
https://www.hilti.com/medias/sys_master/he3/haa/9128239005726/Hilti_Big_Data_Analytics_Competition_2015_English.pdf?mime=application%252Fpdf&realname=Hilti_Big_Data_Analytics_Competition_2015_English.pdf [Stand: 15.12.14]

[3] A. Ng. (16.06.14). *Standford Machine Learning* [Online].
URL: https://class.coursera.org/ml-005/lecture [Stand: 16.12.14]

[4] Hilti. (11.07.2014). *The 2015 Hilti Big Data Analytics Competition [Online].*
URL:
https://www.hilti.com/medias/sys_master/he3/haa/9128239005726/Hilti_Big_Data_Analytics_Competition_2015_English.pdf?mime=application%252Fpdf&realname=Hilti_Big_Data_Analytics_Competition_2015_English.pdf [Stand: 15.12.14]

[5] S. Hendriks. (06.2007). *Visualisierung und Vergleich der Clusterverfahren anhand von QEBS-Daten* [Online].
URL: http://edoc.hu-berlin.de/master/hendriks-sophia-2007-06-14/PDF/hendriks.pdf
[Stand: 16.12.14]

[6] F. Bohnert. (01.2004). Einsatz von Collaborative Filtering zur *Datenprognose*. *Seminararbeit im Rahmen des Einführungsseminar Data Mining im Wintersemester 2003/2004* [Online].
URL:
http://www.mathematik.uni-ulm.de/sai/ws03/dm/arbeit/bohnert.pdf [Stand: 13.12.14]

[7] P.Melville und V.Sindhwani. „Recommender Systems“, in *Encyclopedia of Machine Learning*, C. Sammut und G. Webb, Hrsg. New York: Springer, 2010. S. 829-838.

[8] F. Ricci, Lior Rokach und Bracha Shapira. *Recommender Systems Handbook*. New York, Dordrecht, Heidelberg, London: Springer 2011, S. 1.

[9] F. Ricci, Lior Rokach und Bracha Shapira. *Recommender Systems Handbook*. New York, Dordrecht, Heidelberg, London: Springer 2011, S. 5-7.

[10] A. Ng. (16.06.14). *Standford Machine Learning* [Online].
URL: https://class.coursera.org/ml-005/lecture [Stand: 16.12.14]

[11] A. Klahold. Empfehlungssysteme. *Recommender Systems – Grundlagen, Konzepte und Lösungen*. Wiesbaden: Vieweg und Teubner, 2009, S. 66.

[12] P.Melville und V.Sindhwani. „Recommender Systems“, in *Encyclopedia of Machine Learning*, C. Sammut und G. Webb, Hrsg. New York: Springer, 2010. S. 836.

[13] A. Klahold. Empfehlungssysteme. *Recommender Systems – Grundlagen, Konzepte und Lösungen*. Wiesbaden: Vieweg und Teubner, 2009, S. 66.

[14] P.Melville und V.Sindhwani. „Recommender Systems“, in *Encyclopedia of Machine Learning*, C. Sammut und G. Webb, Hrsg. New York: Springer, 2010. S. 877.

[15] P.Melville und V.Sindhwani. „Recommender Systems“, in *Encyclopedia of Machine Learning*, C. Sammut und G. Webb, Hrsg. New York: Springer, 2010. S. 833.

[16] A. Ng. (16.06.14). *Standford Machine Learning* [Online].
URL: https://class.coursera.org/ml-005/lecture [Stand: 16.12.14]

[17] A. Klahold. Empfehlungssysteme. *Recommender Systems – Grundlagen, Konzepte und Lösungen*. Wiesbaden: Vieweg und Teubner, 2009, S. 37.

[18] A. Ng. (16.06.14). *Standford Machine Learning* [Online].
URL: https://class.coursera.org/ml-005/lecture [Stand: 16.12.14]

[19] F. Ricci, Lior Rokach und Bracha Shapira. *Recommender Systems Handbook*. New York, Dordrecht, Heidelberg, London: Springer 2011, S. 78-79.

[20] F. Ricci, Lior Rokach und Bracha Shapira. *Recommender Systems Handbook*. New York, Dordrecht, Heidelberg, London: Springer 2011, S. 78-79.

[21] F. Bohnert. (01.2004). Einsatz von Collaborative Filtering zur *Datenprognose*. *Seminararbeit im Rahmen des Einführungsseminar Data Mining im Wintersemester 2003/2004* [Online].
URL:
http://www.mathematik.uni-ulm.de/sai/ws03/dm/arbeit/bohnert.pdf [Stand: 13.12.14]

[22] A. Ng. (16.06.14). *Standford Machine Learning* [Online].
URL: https://class.coursera.org/ml-005/lecture [Stand: 16.12.14]

[23] F. Bohnert. (01.2004). Einsatz von Collaborative Filtering zur *Datenprognose*. *Seminararbeit im Rahmen des Einführungsseminar Data Mining im Wintersemester 2003/2004* [Online].
URL:
http://www.mathematik.uni-ulm.de/sai/ws03/dm/arbeit/bohnert.pdf [Stand: 13.12.14]

[24] S.Muno. (8.12.2008). *Seminararbeit zum Thema Recommender-Systeme* [Online].
URL: http://www.is.informatik.uni-duisburg.de/courses/sem_ss08/papers/p06_recommendersystems.pdf [Stand: 16.12.2014]

[25] F. Ricci, Lior Rokach und Bracha Shapira. *Recommender Systems Handbook*. New York, Dordrecht, Heidelberg, London: Springer 2011, S. 78-79.

[26] A. Ng. (16.06.14). *Standford Machine Learning* [Online].
URL: https://class.coursera.org/ml-005/lecture [Stand: 16.12.14]

[27] S.Muno. (8.12.2008). *Seminararbeit zum Thema Recommender-Systeme* [Online].
URL: http://www.is.informatik.uni-duisburg.de/courses/sem_ss08/papers/p06_recommendersystems.pdf [Stand: 16.12.2014]

[28] M.Wiedenbeck und C.Züll, „Klassifikation mit Clusteranalyse: Grundlegende Techniken hierarchischer und k-means-Verfahren“, ZUMA How-to-Reihne, Nr.10, S.1-18, 2001.

[29] A. Klahold. Empfehlungssysteme. *Recommender Systems – Grundlagen, Konzepte und Lösungen*. Wiesbaden: Vieweg und Teubner, 2009, S. 86.

[30] Mathworks. (2014). *k-Means and k-Medois Clustering* [Online].
URL: http://ch.mathworks.com/help/stats/kmeans.html [Stand: 16.12.2014]

[31] Mathworks. (2014). *k-Means and k-Medois Clustering* [Online].
URL: http://ch.mathworks.com/help/stats/hierarchical-clustering.html [Stand: 16.12.2014]

[32] S. Hendriks. (06.2007). *Visualisierung und Vergleich der Clusterverfahren anhand von QEBS-Daten* [Online].
URL: http://edoc.hu-berlin.de/master/hendriks-sophia-2007-06-14/PDF/hendriks.pdf [Stand: 16.12.14]

[33] M.Hubert, P.Rousseeuw und A. Struyf, „Clustering in an Object-Oriented Environment”, *Journal of Statistical Software*, Volume 1, Nr. 4, S.8-25.

[34] A. Klahold. Empfehlungssysteme. *Recommender Systems – Grundlagen, Konzepte und Lösungen*. Wiesbaden: Vieweg und Teubner, 2009, S.71-75.

[35] M.Hubert, P.Rousseeuw und A. Struyf, „Clustering in an Object-Oriented Environment”, *Journal of Statistical Software*, Volume 1, Nr. 4, S.10.

[36] Wikipedia. *Beurteilung eines Klassifikators* [Online].
URL: http://de.wikipedia.org/wiki/Beurteilung_eines_Klassifikators [Stand: 16.12.2014]

6.2 Glossar

In der vorliegenden Arbeit werden folgende Begriffe verwendet:

Begriff	*Erklärung*
Tools	IPCClassDistinct Kategorie Werkzeuge in der Produkthierarchie von Hilti
Consumables	IPCClassDistinct Kategorie Verbrauchsmaterialien in der Produkthierarchie von Hilti
Recommender System	Definition in Abschnitt 2.1
HOL	Hilti Online, das online Vertriebsportal von Hilti.
Collaborative Filtering	Verfahren von Recommender-Systems zum Erlernen von Eigenschaften.
Content-Based Verfahren	Verfahren von Recommender-Systems zum Erlernen von Benutzervorlieben.
Gradient Descent	Verfahren zur schrittweisen Minimierung einer gegebenen Funktion.
Prognosematrix	Matrix mit allen berechneten Bewertungen der Benutzer, der Aufbau dieser Matrix entspricht der Bewertungs-Matrix Y
Item	Produkt oder Dienstleistung
IPC Class	Ist eine Stufe über den Produkten in der Produktklassifizierung von Hilti.

Tabelle 24 Terminologie

6.3 Abbildungsverzeichnis

6.4 Diagrammverzeichnis

6.5 Tabellenverzeichnis

6.6 Formelverzeichnis

Kapitel 7 Anhang

7.1 Inhaltsverzeichnis von CD

01 Projektmanagement
11 Aufgabenstellung
12 Sitzungsprotokolle

02 Prototyp
21 Sourcecode
22 Experimente
23 SQL

03 Hitli Daten
31 Trainingsdatensatz
32 Validierungsdatensatz

04 Dokumentation

7.2 Sequenzdiagramm

Im Diagramm 7 ist der Ablauf des Prototyps in einem Sequenzdiagramm dargestellt.

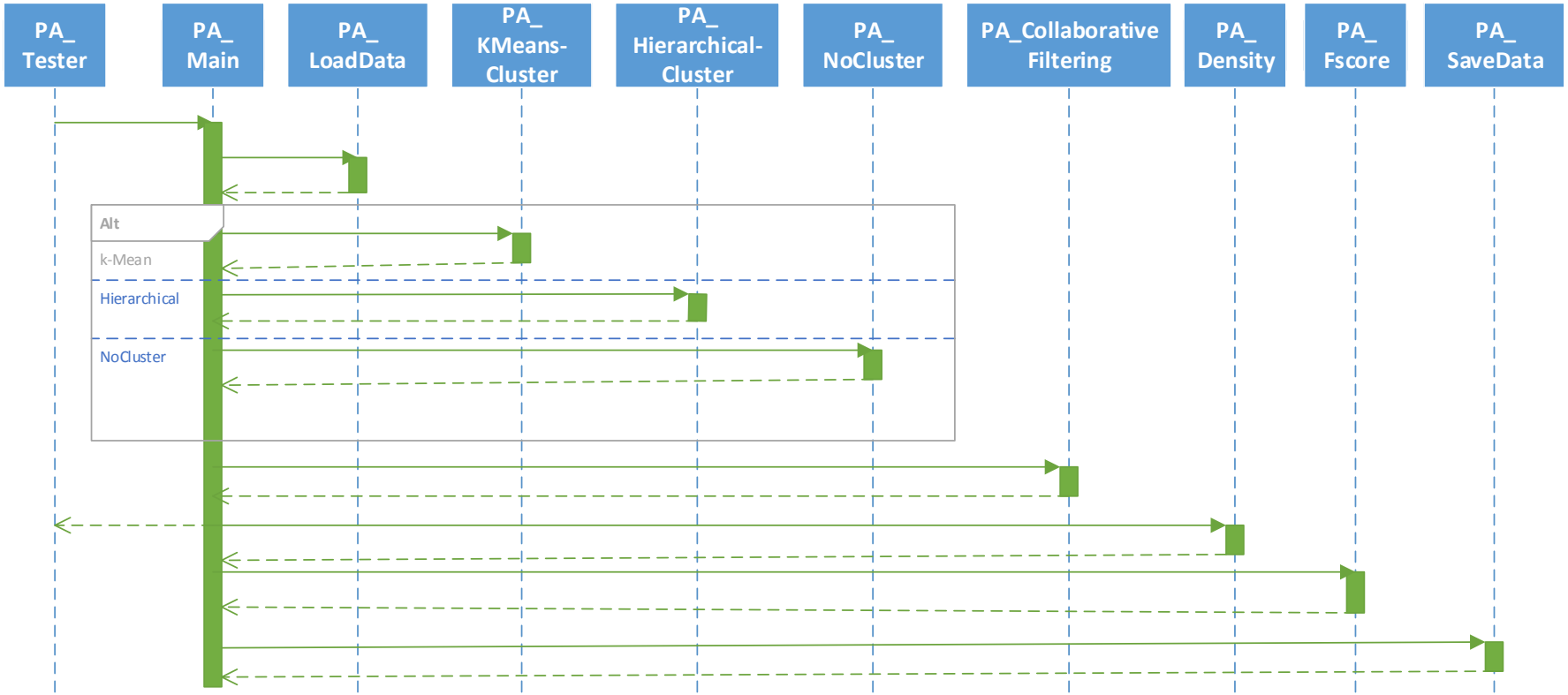

Diagramm 7 Sequenzdiagramm Prototyp

7.3 Daten von Hilti

Hilti_dataset_activities

Title	Description	Type
ActDateFrom	Date when activity took place	Date
Category	Interaction type (e.g. meeting, call, sms, etc.)	Categorical
CreatedBy	Identifier of a Hilti employee who created the activity in the system	Text
CustomerID	Customer identifier	Text
Function	Job position of the contact in his company (e.g. CIO, job site manager, foreman)	Categorical
NumAct	Number of activities of the same category with the same customer on a specific date	Numerical
Objective	Activity objective	Categorical
Result	Activity result	Categorical

Tabelle 25 Hilti_dataset_activities

Hilti_dataset

Title	Description	Type
CustomerID	Customer identifier	Text
ProductCode	Product identifier	Text
IPCClass	Product type (one level below the IPCLine in the product hierarchy)	Categorical
IPCLine	Product category	Categorical
IPCClassDistinct	1 - Tools, 2 - Consumables, 3 - Accessories, 4 - Spares/Repair, 5 - Others	Categorical
PurchaseDate	PurchaseDate	Date
Quantity	Number of bought items	Numerical
QtyUnit	Unit of bought item. H - hours, KG - Kilograms, M - Meters, ST - Pieces, TAG - Days	Categorical
NetSales	Sales net value associated with this purchase	Numerical
SalesChannel	Sales channel used for the purchase. C1 - Customer Service, C2 - Hilti Center, C3 - Territory Sales, C4 - HOL, C5 - Hilti ProShop	Categorical
EngagementStatus	Relationship level between the company and the customer (calculated based on variery of parameters, e.g. number of sales channels used). L1 refers to the most engaged customer and L5 to the least engaged ones.	Categorical
PotentialClass	Growth potential. A -the biggest growth potential to E - the smallest growth potential.	Categorical
FleetUser	If the user has a fleet (leasing) contract. A - Active FM Customer, I - Inactive FM Customer, N - Non FM Customer	Categorical
HOLUser	If the user uses the online sales channel (HOL)	Boolean
Ship To Trade	Area of work of the customer's location, where the goods were delivered	Categorical
VisitFrequency	Category of visit frequency from sales person (F1 - Yearly, F2 - Half-yearly, F3 - Never, F4 - Every quarter,	Categorical

	F5 - Every 2 months, F6 - Monthly, F7 - Biweekly, F8 - Weekly, F9 - Semi-weekly, F10 - Every second day)	
NumberEmployees	Number of employees in customer's company	Numerical
ShipToPostalCode	Postal code of the delivery address	Text
DeletionFlag	If the user is deleted from the database. Soft delete. X - customer deleted	Categorical
Territory	Sales territory. Assigned based on different rules (dependent on the trade (area of work) of the customer and the geographical location)	Text

Tabelle 26 Hilti_dataset

7.3.1 Datenaufbereiten für MySQL

Alle CSV-Dateien haben als Linebreak Symbol `^M`. Um die CSV-Dateien direkt ins MySQL importieren zu können muss das Fileformat angepasst werden.

Vorgehen:

1. Terminal öffen und ins Verzeichnis der CSV-Dateien navigieren
   ```
   $cd path
   ```
2. CSV Linebreak kontrollieren
   ```
   $cat –v Filename
   ```
3. Fileformat kontrollieren
   ```
   $vim Filename
   ```
   ESC-Taste drücken
   ```
   :set ff
   ```
   Falls nicht fileformat=unix steht Schritt 4 ansonsten Schritt 5
4. Fileformat ändern
   ```
   $vim Filename
   ```
   ESC-Taste drücken
   ```
   :set fileformat=unix
   :w
   ```
5. Fileencoding kontrollieren
   $vim Filename
   ESC-Taste drücken
   ```
   :set fileencoding
   :q
   ```
6. Fileencoding ändern
   ```
   $vim Filename
   ```
   ESC-Taste drücken
   ```
   :set fileencoding =utf8
   ```
   ESC-Taste drücken
   ```
   :w
   ```

7.4 MySQL-Abfragen

7.4.1 Haupt-Tabellen erstellen

Purchases:

```
CREATE TABLE IF NOT EXISTS `test`.`purchases` (
    `recID` INT NULL,
    `CustomerID` INT NULL,
    `ProductCode` VARCHAR(45) NULL,
    `IPCClass` VARCHAR(45) NULL,
    `IPCLine` VARCHAR(45) NULL,
    `IPCClassDistinct` INT NULL,
    `PurchaseDate` DATETIME NULL,
    `Quantity` INT NULL,
    `QtyUnit` VARCHAR(45) NULL,
    `NetSales` FLOAT NULL,
    `SalesChannel` VARCHAR(45) NULL,
    `Engagement` VARCHAR(45) NULL,
    `PotentialClass` VARCHAR(45) NULL,
    `FleetUser` VARCHAR(45) NULL,
    `HOLUser` VARCHAR(45) NULL,
    `ShipToTrade` VARCHAR(45) NULL,
    `VisitFrequency` VARCHAR(45) NULL,
    `NumberEmployees` INT NULL,
    `ShipToPostalCode` VARCHAR(45) NULL,
    `DeletionFlag` VARCHAR(45) NULL,
    `Territory` INT NULL
)  ENGINE=InnoDB;
```

Purchases_validation

```
CREATE TABLE IF NOT EXISTS `test`.`purchases_validation` (
      `recID` INT NULL,
    `CustomerID` INT NULL,
    `ProductCode` VARCHAR(45) NULL,
    `IPCClass` VARCHAR(45) NULL,
    `IPCLine` VARCHAR(45) NULL,
    `IPCClassDistinct` INT NULL,
    `PurchaseDate` DATETIME NULL,
    `Quantity` INT NULL,
    `QtyUnit` VARCHAR(45) NULL,
    `NetSales` FLOAT NULL,
    `SalesChannel` VARCHAR(45) NULL,
    `Engagement` VARCHAR(45) NULL,
    `PotentialClass` VARCHAR(45) NULL,
    `FleetUser` VARCHAR(45) NULL,
    `HOLUser` VARCHAR(45) NULL,
    `ShipToTrade` VARCHAR(45) NULL,
    `VisitFrequency` VARCHAR(45) NULL,
    `NumberEmployees` INT NULL,
    `ShipToPostalCode` VARCHAR(45) NULL,
    `DeletionFlag` VARCHAR(45) NULL,
    `Territory` INT NULL
)  ENGINE=InnoDB;
```

Activities

```
CREATE TABLE IF NOT EXISTS `test`.`activities` (
  `recID` INT NULL,
  `ActDateFrom` DATETIME NULL,
```

```
  `CreatedBy` INT NULL,
  `Category` VARCHAR(45) NULL,
  `Function` INT NULL,
  `CustomerID` INT NULL,
  `Objective` VARCHAR(45) NULL,
  `Result` VARCHAR(45) NULL,
  `NumAct` INT NULL
) ENGINE = InnoDB
```

7.4.2 Haupt-Tabellen füllen

Purchases

```
LOAD DATA LOCAL INFILE 'D:/ZHAW/PA/data/raw_data/Hilti_dataset_training.csv'
INTO TABLE test.purchases
FIELDS TERMINATED BY ','
ENCLOSED BY '\"'
LINES TERMINATED BY '\n'
IGNORE 1 LINES;
```

Purchases_validation

```
LOAD DATA LOCAL INFILE 'D:/ZHAW/PA/data/raw_data/Hilti_dataset_validation.csv'
INTO TABLE test.purchases_validation
FIELDS TERMINATED BY ';'
LINES TERMINATED BY '\n'
IGNORE 1 LINES;
```

Activities

```
LOAD DATA LOCAL INFILE 'D:/ZHAW/PA/data/raw_data/Hilti_dataset_activi-
ties_training.csv'
INTO TABLE test.activities
FIELDS TERMINATED BY ','
ENCLOSED BY '\"'
LINES TERMINATED BY '\n'
IGNORE 1 LINES;
```

7.4.3 Hilfs-Tabellen erstellen und füllen

7.4.3.1 ActivitiesJoinPurchases

```
DROP TABLE IF EXISTS `ActivitiesJoinPurchases`;
CREATE TABLE IF NOT EXISTS `ActivitiesJoinPurchases` (
      `RecordID` INT AUTO_INCREMENT PRIMARY KEY,
      `ActivitiesRecordID` INT NULL,
      `PurchasesRecordID` INT NULL,
    `CustomerID` INT NULL,
    `PurchaseDate` DATETIME NULL,
    `ActDateFrom` DATETIME NULL,
    `NumActivities` INT NULL,
    `Objective` VARCHAR(42) NULL,
    `Result` VARCHAR(42) NULL
)  ENGINE=InnoDB;

INSERT INTO ActivitiesJoinPurchases (
      ActivitiesRecordID,
      PurchasesRecordID,
      CustomerID,
      PurchaseDate,
      ActDateFrom,
```

```
      NumActivities,
      Objective,
      Result)
SELECT
      activities.recID,
      purchases.recID,
      purchases.CustomerID,
      purchases.PurchaseDate,
      activities.ActDateFrom,
      activities.NumAct,
      activities.Objective,
      activities.Result
FROM
      purchases
      JOIN
            activities
            ON
                  activities.CustomerID = purchases.CustomerID;
```

7.4.3.2 ActivitiesJoinPurchases in interval 30d

```
DROP TABLE IF EXISTS `ActivitiesJoinPurchases_in_interval_30d_0d`;
CREATE TABLE IF NOT EXISTS `ActivitiesJoinPurchases_in_interval_30d_0d` (
      `RecordID` INT AUTO_INCREMENT PRIMARY KEY,
      `ActivitiesRecordID` INT NULL,
    `CustomerID` INT NULL,
    `NumActivities` INT NULL,
    `Result` VARCHAR(128) NULL
)  ENGINE=InnoDB;

INSERT INTO ActivitiesJoinPurchases_in_interval_30d_0d (
      ActivitiesRecordID,
      CustomerID,
      NumActivities,
      Result)
SELECT
      ActivitiesRecordID,
      CustomerID,
      NumActivities,
      Result
FROM
      activitiesjoinpurchases
WHERE
      PurchaseDate >= DATE_SUB(ActDateFrom, INTERVAL 30 DAY)
      AND PurchaseDate <= ActDateFrom;
```

7.4.3.3 ActivitiesJoinNoPurchases in interval 30d

```
DROP TABLE IF EXISTS `ActivitiesJoinNoPurchases_30d`;
CREATE TABLE IF NOT EXISTS `ActivitiesJoinNoPurchases_30d` (
      `RecordID` INT AUTO_INCREMENT PRIMARY KEY,
      `ActivitiesRecordID` INT NULL,
    `CustomerID` INT NULL,
    `ActDateFrom` DATETIME NULL,
    `NumActivities` INT NULL,
    `Objective` VARCHAR(42) NULL,
    `Result` VARCHAR(42) NULL
)  ENGINE=InnoDB;

INSERT INTO ActivitiesJoinNoPurchases_30d (
      CustomerID,
      ActivitiesRecordID,
      ActDateFrom,
      NumActivities,
```

```
      Objective,
      Result)
SELECT
      activities.CustomerID,
      activities.recID,
      activities.ActDateFrom,
      activities.NumAct,
      activities.Objective,
      activities.Result
FROM
      activities
WHERE
      NOT EXISTS (
            SELECT purchases.CustomerID
            FROM purchases
            WHERE
                  purchases.CustomerID = activities.CustomerID
                  AND PurchaseDate >= DATE_SUB(ActDateFrom, INTERVAL 30 DAY)
                  AND PurchaseDate <= ActDateFrom
      );
```

7.4.3.4 Customers

```
DROP TABLE IF EXISTS `Customer`;
CREATE TABLE IF NOT EXISTS `Customer`(
      CustomerID INT PRIMARY KEY
);

INSERT INTO Customer (
      CustomerID
)
SELECT
      DISTINCT CustomerID
FROM
      purchases
ORDER BY
      CustomerID;
```

7.4.3.5 Producthirarchie

```
DROP TABLE IF EXISTS `producthirarchie`;
CREATE TABLE IF NOT EXISTS `producthirarchie`(
      IPCClass VARCHAR(32) NULL,
      IPCLine VARCHAR(32) NULL,
      IPCClassDistinct INT NULL
);

INSERT INTO producthirarchie (
      IPCClass,
      IPCLine,
      IPCClassDistinct
)
SELECT
      DISTINCT purchases.IPCClass,
      purchases.IPCLine,
      MIN(purchases.IPCClassDistinct)
FROM
      purchases
GROUP BY
      IPCClass,
      IPCLine
ORDER BY
      IPCClass;
```

7.4.3.6 Products

```
DROP TABLE IF EXISTS `products`;
CREATE TABLE IF NOT EXISTS `products`(
	ProductCode VARCHAR(32) PRIMARY KEY NULL,
	IPCClass VARCHAR(32) NULL,
	IPCLine VARCHAR(32) NULL,
	IPCClassDistinct INT NULL
);

INSERT INTO products (
	ProductCode,
	IPCClass,
	IPCLine,
	IPCClassDistinct
)
SELECT
	DISTINCT purchases.ProductCode,
	purchases.IPCClass,
	purchases.IPCLine,
	MIN(purchases.IPCClassDistinct)
FROM
	purchases
GROUP BY
	ProductCode,
	IPCclass,
	IPCLine
ORDER BY
	IPCClassDistinct,
	IPCLine,
	IPCClass,
	ProductCode;
```

7.4.4 Statistics

7.4.4.1 Sparseness

```
SELECT
	COUNT(DISTINCT CustomerID) AS NumberOfCustomers,
	COUNT(DISTINCT ProductCode) AS NumberOfProducts,
	COUNT(DISTINCT IPCClass) AS NumberOfSubClasses2,
	COUNT(DISTINCT CustomerID, IPCClass) AS NumberOfRatingsClasses,
	COUNT(DISTINCT CustomerID, ProductCode) AS NumberOfRatingsProducts
FROM purchases
WHERE
	IPCClassDistinct = 1 OR
    IPCClassDistinct = 2;Ratings
```

7.4.4.2 Boolean

7.4.4.2.1 Training

```
SELECT
	IF( EXISTS(  SELECT *
                        FROM purchases
                        WHERE purchases.CustomerID = customer.CustomerID AND
purchases.IPCClass = producthirarchie.IPCClass) ,1,0) AS Rating
FROM
	customer
	JOIN
		producthirarchie
	ON
		IPCClassDistinct <= 2;
```

7.4.4.2.2 Validation

```
SELECT
```

```
      IF( EXISTS(  SELECT *
                        FROM purchases_validation
                        WHERE   purchases_validation.CustomerID   =   cus-
tomer.CustomerID  AND  purchases_validation.IPCClass  =  producthirarchie.IP-
CClass) ,1,0) AS Rating
FROM
      customer
      JOIN
            producthirarchie
      ON
```

7.4.4.3 Count

7.4.4.3.1 Training

```
SELECT
      ( SELECT COUNT(purchases.recID)
            FROM purchases
            WHERE  purchases.CustomerID  =  customer.CustomerID  AND  pur-
chases.IPCClass = producthirarchie.IPCClass) AS Rating
FROM
      customer
      JOIN
            producthirarchie
      ON
            IPCClassDistinct <= 2;
```

7.4.4.3.2 Validation

```
SELECT
      ( SELECT COUNT(purchases_validation.recID)
            FROM purchases_validation
            WHERE purchases_validation.CustomerID = customer.CustomerID AND
purchases_validation.IPCClass = producthirarchie.IPCClass) AS Rating
FROM
      customer
      JOIN
            producthirarchie
      ON
            IPCClassDistinct <= 2;
```

7.4.4.4 Count-Quantity

7.4.4.4.1 Training

```
SELECT
      IF( EXISTS(  SELECT *
                        FROM purchases
                        WHERE purchases.CustomerID = customer.CustomerID AND
purchases.IPCClass = producthirarchie.IPCClass)
            ,
                  ( SELECT SUM(purchases.Quantity)
                              FROM purchases
                              WHERE   purchases.CustomerID   =   cus-
tomer.CustomerID AND purchases.IPCClass = producthirarchie.IPCClass)
            ,
                  0) AS Rating

FROM
      customer
      JOIN
            producthirarchie
      ON
            IPCClassDistinct <= 2;
```

7.4.4.4.2 Validation

```
SELECT
      IF( EXISTS(  SELECT *
```

```
                        FROM purchases_validation
                        WHERE   purchases_validation.CustomerID   =   cus-
tomer.CustomerID  AND  purchases_validation.IPCClass  =  producthirarchie.IP-
CClass)
            ,
                  ( SELECT SUM(purchases_validation.Quantity)
                                    FROM purchases_validation
                                    WHERE purchases_validation.CustomerID =
customer.CustomerID AND purchases_validation.IPCClass = producthirarchie.IP-
CClass)
            ,
                  0) AS Rating

FROM
      customer
      JOIN
            producthirarchie
      ON
            IPCClassDistinct <= 2;
```