Recommender Systems (RSY) - Mini-Challenge 1

Studiengang Data Science (HS2023)

Daniel Perruchoud, Institut für Data Science

1 RSY Leistungsnachweis

Im Modul "Recommender Systems (RSY)" beruht der Leistungsnachweis auf der Demonstration praktischer und theoretischer Kompetenzen.

Die praktische Kompetenz wird mit Hilfe zweier Mini-Challenges geprüft, die theoretische Kompetenz mittels schriftlicher MSP, welche nach Abgabe der zwei Mini-Challenges absolviert wird.

Die Gesamtbeurteilung setzt sich zusammen aus zwei benoteten Mini-Challenges (Gewicht je 25%) und schriftlicher MSP (Gewicht 50%).

2 Mini-Challenge Grundlagen

2.1 Software

Die Mini-Challenge 1 wird mit R und RStudio bearbeitet. Zum Einsatz kommen insbesondere die R-Pakete `recommenderlab`, `tidyverse`, und optional andere.

2.2 Daten

Grundlage der Mini-Challenge sind Filmratings von 943 Nutzer*innen für 1664 Filme, gesammelt zwischen 19. September 1997 und 22. April 1998 (Quelle: MovieLens). Die Daten sind integrierter Bestandteil von `recommenderlab` und können via `data("MovieLense")` geladen werden.

3 Mini-Challenge Abgabebedingungen

3.1 Lieferobjekte

Analysen sind als R-Studio Notebooks abzugeben, **mindestens** gefordert sind

- ein .HTML-File,
- ein .RMD-File. und
- ein separates .R-Skript (für eigene Funktionen).

Titel der Mini-Challegen und Autorenschaft sind im Namen zu vermerken (z.B. "HS23_RSY_MC1_DanielPerruchoud"). Die Analysen sind termingerecht per e-mail an "daniel.perruchoud@fhnw.ch" zu senden.

3.2 Erarbeitung

Die Mini-Challenge 1 darf allein, in 2-er oder 3-er Gruppen erarbeitet werden. Dabei analysieren Gruppenmitglieder auf unterschiedlich gesampelten Datensätzen und diskutieren individuelle Resultate anhand der Erkenntnisse der anderen Gruppenmitglieder (z.B. führen 3-er Gruppen alle Analysen auf 3 unterschiedlichen Samples der Datensätze aus).

Hinweise:

Eine Zusammenarbeit zwischen Gruppen darf sich nur auf konzeptionelle Aspekte beschränken, insbesondere darf kein Code von anderen Gruppen oder aus dem Internet kopiert werden.

3.3 Hilfsmittel

Die Verwendung von ChatGPT oder vergleichbaren AI-Tools ist erlaubt. Ihre Verwendung ist im Lieferobjekt für entsprechende Code-Stücke zu vermerken und übergeordnet am Ende der Analyse in einem separaten Abschnitt kurz zu beurteilen und diskutieren (Länge 250-500 Wörter). Zu beurteilen sind dabei für welche Task das AI-Tool eingesetzt worden ist, und welche Ansprache-Strategie (Prompting Strategie) verwendet wurde. Zudem soll beschrieben werden, welche Prompting Strategie am erfolgreichsten war, d.h. am meisten a) zum Lösen der Aufgabe und b) zum Kompetenzerwerb beitragen konnte.

3.4 Abgabetermin

Der Abgabetermin für Mini-Challenge 1 ist der 15. Dezember 2023.

4 Mini-Challenge Beurteilungskriterien

Die Notenvergabe beruht auf den unten aufgeführten Beurteilungskriterien.

4.1 Vollständigkeit

Die Analysen sind inhaltlich vollständig gemäss Beschreibung der Mini-Challenge zu lösen.

4.2 Korrektheit

Die abgebenen Analysen werden auf inhaltliche Korrektheit geprüft. Lauffähigkeit der Codes und dokumentierte Tests eigener Funktionen sind notwendige Grundvoraussetzung dafür.

4.3 Nachvollziehbarkeit

Die Analysen sind so zu gestalten, dass sowohl die

- zugrundeliegende Überlegungen,
- deren Implementierung und
- die abgeleiteten Resultate

nachvollziehbar sind. Das setzt voraus, dass die

- Notebooks als Ganzes ideal strukturiert sind, d.h. ein navigierbares Inhaltsverzeichnis und sinnvolle Kapitel und Abschnitte enthalten,
- Codes gemäss Best Practice Standards strukturiert, formatiert und kommentiert sind, d.h. insbesondere sinnvolle Namensgebung und kommentierte Funktionen verwenden,
- Analyseresulate mit Tabellen und Grafiken dargestellt und auch mit Text vollständig diskutiert sind.

4.4 Best Practice Standards

Zudem sollen Wiederholung durch kopieren von Code vermieden und in Funktionen ausgelagert werden, welche vor Verwendung sinnvoll getestet werden.

5 Mini-Challenge Inhalte & Lernziele

Im Rahmen der Mini-Challenge wird ein Recommender System für Spielfilme entwickelt und getestet, d.h. es geht um die

- Erzeugung von Empfehlungen mittels Collaborative Memory- und Model-based Recommenders,
- Off-line Evaluierung von Recommender Systems und Analyse von Top-N Empfehlungen,
- Eigene Implementierung zentraler Komponenten.

Die eigenen Implementierungen zentraler Komponenten sind weiter unten mit dem Label **DIY** (Do it yourself) gekennzeichnet.

Lernziele der Mini-Challenge umfassen folgende Punkte

- Grundkonzepte für Recommender Systems
- Modell-Metriken für Ranked Evaluation Probleme
- Evaluierungs-Schemata zur Beurteilung von Recommender Systems
- Recommender System-Algorithmen und deren Hyperparameter
- Hyperparameter-Optimierung und Kreuzvalidierung von Recommender Systems

6 Collaborative Movie Recommender

Die Mini-Challenge besteht aus zehn Aufgaben, die nachgehend aufgeführt sind. Die Anzahl Punkte pro Aufgabe richtet sich nach Aufwand und Komplexität. Für alle Aufgaben sind **Codes vollständig zu kommentieren** und **Analysen präzise zu dokumentieren**.

Ideen und Fragen können jederzeit in der RSY-Kontaktstunde diskutiert werden.

6.1 Explorative Datenanalyse [10 Punkte]

Aufgabe 1: Untersuche den vollständigen MovieLens Datensatz und beantworte folgende Fragen:

1. Welches sind die am häufigsten geschauten Genres/Filme?
2. Wie verteilen sich die Nutzerratings der Filme gesamthaft bzw. nach Genres?
3. Wie verteilen sich die mittleren Ratings pro Film bzw. pro Nutzer*in?
4. Welchen Einfluss hat die Normierung der Ratings pro Nutzer*in auf die Verteilung der mittleren Nutzerratings?
5. Welche strukturellen Charakteristika und Auffälligkeiten zeigt die User-Item Matrix?

Hinweise:

- Die explorative Datenanalyse ist auf dem gesamten Datensatz (vor Datenreduktion!) durchzuführen.
- Die explorative Datenanalyse soll sinnvoll Visualisierungen und Tabellen verwenden und die oben gestellten Fragen präzise beantworten.

6.2 Datenreduktion [6 Punkte]

Aufgabe 2: Reduziere den MovieLens Datensatz auf rund 400 Nutzer*innen und 700 Filme, indem du Filme und Nutzer*innen mit sehr wenigen Ratings entfernst.

Untersuche und visualisiere den Effekt der Datenreduktion, d.h.:

1. Anzahl Filme und Nutzer*innen sowie Sparsity vor und nach Datenreduktion,
2. Mittlere Nutzerratings pro Film vor und nach Datenreduktion,
3. **Zusatz für Gruppen**: Quantifiziere die “Intersection over Union” aller reduzierten Datensätze paarweise.

Hinweise:

- Diese Aufgabe ist eine Fortsetzung von Aufgabe 1.
- Die Analyse zur Datenreduktion soll sinnvoll Visualisierungen und Tabellen verwenden.
- Ohne explizite Hinweise sind die nachfolgenden Analysen auf den reduzierten Datensätzen durchzuführen.
- 2-er Gruppen verwenden zwei unterschiedliche reduzierte Datensätze, 3-er Gruppen drei unterschiedliche reduzierte Datensätze.

6.3 Analyse Ähnlichkeitsmatrix [12 Punkte]

Aufgabe 3: Erzeuge einen IBCF Recommender und analysiere die Ähnlichkeitsmatrix des trainierten Modelles für den reduzierten Datensatz.

1. Zerlege den Datensatz in Trainings- und Testdaten im verhältnis 4:1,
2. Trainiere ein IBCF Modell mit 30 Nachbarn und Cosine Similarity,
3. Bestimme die Filme, die am häufigsten in der Cosine-Ähnlichkeitsmatrix auftauchen und analysiere deren Vorkommen und Ratings im reduzierten Datensatz.
4. Wiederhole die Analyse, indem du bei der Datenpartitionierung die Anzahl nicht-maskierter Produkte der Test-User veränderst und kommentiere den Einfluss auf die Resultate.

Hinweise:

- Diese Aufgabe ist eine Fortsetzung von Aufgabe 2.
- Die Analyse soll sinnvolle Visualisierungen verwenden.
- Die Aufgabe ist mit Hilfe der Funktion `evaluationScheme()` durchzuführen. Untersuche zunächst Funktionsweise und Parameter der Funktion `evaluationScheme()`.
- Die Anzahl nicht-maskierter Produkte der Test-User kann in der Datenpartitionierung mit `evaluationScheme()` über den Parameter `given` gesteuert werden.

6.4 Implementierung Ähnlichkeitsmatrix [18 Punkte]

Aufgabe 4 (DIY): Implementiere Funktionen zur Berechnung von Ähnlichkeitsmatrizen bei IBCF Recommenders für (a) Cosine Similarity mit ordinale Ratings und (b) Jaccard Similarity mit binären Ratings.

1. Vergleiche die Resultate beider Funktionen hinsichtlich Übereinstimmung und Laufzeit mit dem Resultat der Funktion `Recommender()` und der eines anderen R-Paketes anhand 100 zufällig gewählter Filme,
2. Visualisiere und vergleiche die Verteilung der Ähnlichkeiten von Cosine Similarity für ordinale Ratings und von Jaccard Similarity für binäre Ratings mittels den von dir implementierten Funktionen.

Hinweise:

- Diese Aufgabe dient dazu das Verständnis der in Memory-based Recommender Systems verwendeten Kernkomponenten zu vertiefen.
- Die eigene Implementierung ist gründlich zu prüfen und zu dokumentieren, wobei for-Loops zu vermeiden sind.
- Für den geforderten Vergleich sind bestehende Funktionen aus anderen R-Paketen zu verwenden.
- Überlege insbesondere wie die Parameter `normalize`, `na_as_zero` der Funktion `Recommender()` aus dem Paket `recommenderlab` zu setzen sind, damit eine Übereinstimmung mit der selber implementierten Funktion resultiert.

6.5 Produktabdeckung - Top-N Listen von IBCF und UBCF [12 Punkte]

Aufgabe 5: Vergleiche und diskutiere Top-N Empfehlungen von IBCF und UBCF Modellen mit 30 Nachbarn und Cosine Similarity für den reduzierten Datensatz.

1. Berechne die Top-15 Empfehlungen aller Testnutzer*innen via IBCF und UBCF,
2. Vergleiche die Top-15 Empfehlungen von IBCF vs UBCF für drei Testnutzer*innen mittels Tabelle,
3. Visualisiere und diskutiere für alle Testnutzer*innen summarisch die Verteilung der Top-15 Empfehlungen von IBCF und UBCF.

Hinweise:

- Diese Aufgabe ist eine Fortsetzung von Aufgabe 3.
- Die Analyse soll sinnvoll Visualisierungen und Tabellen verwenden und die Resultate diskutieren.
- Die Aufgabe beleuchtet, wie Empfehlungen unterschiedlicher Modelle den Produktkatalog abdecken. Diskutiere in diesem Zusammenhang die pauschale Behauptung “Recommender Systeme machen für alle Nutzer*innen die gleichen Empfehlungen”.

6.6 Personalisierte Empfehlungen - Top-N Listen von IBCF und UBCF [16 Punkte]

Aufgabe 6: Untersuche den Einfluss von Ratings und Modelltyp auf Top-N Empfehlungen für den reduzierten Datensatz und vergleiche die Empfehlungen über alle Testnutzer*innen in den Top-15 Listen, wenn Modelltyp und Rating verändert werden.

Vergleiche die Verteilung übereinstimmender Empfehlungen aller Testnutzer*innen in den Top-15 Listen für

1. IBCF vs UBCF, beide mit ordinalen Ratings und Cosine Similarity,
2. IBCF vs UBCF, beide mit ordinalen, normalisierten Ratings und Cosine Similarity.

Hinweise:

- Diese Aufgabe ist eine Fortsetzung von Aufgabe 5.
- Gefordert ist eine vergleichende, statistische Analyse inklusive Visualisierung. Der erste Schritt dabei ist die Übereinstimmung der Empfehlungen pro Nutzer*in zu untersuchen.

- Implementiere eine Funktion für die Überprüfung der Übereinstimmung von Empfehlungen.

6.7 Analyse Top-N Listen - IBCF vs SVD [8 Punkte]

Aufgabe 7: Vergleiche Memory-based und Modell-based Recommenders bezüglich Überschneidung der Top-N Empfehlungen für den reduzierten Datensatz, analog zur vorangehenden Aufgabe.

Vergleiche wie sich der Anteil übereinstimmender Empfehlungen der Top-15 Liste verändert für

1. IBCF (Cosine Similarity, 30 Nachbarn, ordinale Ratings) vs Truncated SVD Modelle mit Variation der Anzahl Singulärwerte

Hinweise:

- Diese Aufgabe ist eine Fortsetzung von Aufgabe 6.
- Gefordert ist eine vergleichende, statistische Analyse inklusive Visualisierung.
- Die Anzahl Singulärwert ist zu variieren von 10 auf 20, 30, 40, und 50.

6.8 Implementierung Top-N Metriken [16 Punkte]

Aufgabe 8 (DIY): Implementiere Funktionen, um aus den Top-N Listen aller Nutzer*innen die Item-space Coverage@N und Novelty@N eines Recommenders zu beurteilen.

Visualisiere diese System-Metriken als Scatterplot “Novelty vs Coverage” für Top-N Listen der Länge N = 5, 10, 15, 20, 25, 30 mit

1. IBCF (Cosine Similarity, 30 Nachbarn),
2. Truncated SDV (30 Singulärwerte).

Hinweise:

- Diese Aufgabe dient dazu die zentrale Frage der Evaluierung von Recommender Systemen zu vertiefen und dient als Input für die Entscheidung von Aufgabe 9.
- Die eigene Implementierung ist sinnvoll zu testen und zu dokumentieren.

6.9 Wahl des optimalen Recommenders [20 Punkte]

Aufgabe 9: Bestimme aus 5 unterschiedlichen Modellen das für Top-N Empfehlungen “beste” Modell und verwende zusätzlich einen Top-Movie Recommender.

1. Verwende für die Evaluierung 10-fache Kreuzvalidierung,
2. Begründe deine Wahl von Metriken und Modell,
3. Analysiere das “beste” Modell für Top-N Recommendations mit N = 10, 15, 20, 25 und 30,
4. Optimiere das “beste” Modell hinsichtlich Hyperparametern.

Hinweise:

- Diese Aufgabe ist eine Fortsetzung von Aufgabe 8.
- Untersuche und visualisiere die Variabilität der Vorhersage mittels Kreuzvalidierung und verwende für den Top-Movie Recommender `recommenderlab's POPULAR` Modell.
- Die Wahl des “besten” Modelles ist auf Basis der Resultate aller vorangehenden Analysen zu treffen und zu begründen.

6.10 Implementierung Top-N Monitor [20 Punkte]

Aufgabe 10 (DIY): Untersuche die relative Übereinstimmung zwischen Top-N Empfehlungen und präferierten Filmen für das “beste” Modell und zwei weiteren Modelle bzw. -parametrisierungen.

1. Fixiere 20 zufällig gewählte Testnutzer*innen für alle Modellvergleiche,
2. Bestimme den Anteil der Top-N Empfehlung nach Genres pro Nutzer*in,
3. Bestimme pro Nutzer*in den Anteil nach Genres seiner Top-Filme (=Filme mit besten Bewertungen),
4. Vergleiche pro Nutzer*in Top-Empfehlungen vs Top-Filme nach Genres,
5. Definiere eine Qualitätsmetrik für Top-N Listen und teste sie.

Hinweise:

- Diese Aufgabe ist eine Fortsetzung von Aufgabe 9.
- Der Top-N Monitor erlaubt zu überprüfen, ob die gemachten Empfehlungen den Präferenzen der Nutzer*innen entsprechen und verwendet eine einfach verständliche Visualisierung.
- Die eigene Implementierung einer Qualitätsmetrik ist gründlich zu prüfen und zu dokumentieren.