**Unser Empfehlungssystem**

Wir haben ein hybrides Empfehlungssystem entwickelt, das den **Content-based Algorithm** mit dem **User Collaborative Algorithm** kombiniert. Unser Ziel ist es, dem eingeloggten Nutzer Filme vorzuschlagen, die den Präferenzen anderer ähnlicher Nutzer entsprechen, wobei wir insbesondere auf Übereinstimmungen in den **Genre-Kategorien** achten.

**Technischer Ablauf**

Im Code beginnen wir damit, die Daten der Filme und der gesamten Nutzerbasis zu laden und in entsprechende Variablen zu speichern. Anschließend ermitteln wir für jeden Nutzer dessen bevorzugte Genres und speichern diese als separate Listen. Für den eingeloggten Nutzer extrahieren wir dessen Genres und favorisierten Filme.

Danach durchlaufen wir alle anderen Nutzer, wobei der eingeloggte Nutzer wird dabei ausgeschlossen. Für jeden dieser Nutzer überprüfen wir die Ähnlichkeit zum eingeloggten Nutzer und fügen diesen Wert einer speziellen Ähnlichkeitsliste hinzu.

Diese Liste sortieren wir absteigend nach den Ähnlichkeitswerten, sodass der ähnlichste Nutzer an erster Stelle steht. Von diesem ähnlichsten Nutzer entnehmen wir dann dessen Filmempfehlungen, die anschließend an den Controller zurückgegeben werden.

**Zukünftige Entwicklungen und Datenbankintegration**

Aktuell ist unser System auf die bestehenden Datenmengen zugeschnitten. Für zukünftige, größere Nutzerdatenbanken planen wir den Einsatz von **Machine Learning (ML)**. ML-Technologien könnten uns dabei helfen, Informationen effizienter und schneller zu verarbeiten. Zum jetzigen Zeitpunkt wäre der Aufwand für die Implementierung von ML – einschließlich zusätzlicher Treiber, Software und Bibliotheken – jedoch unverhältnismäßig hoch und würde sich angesichts unserer aktuellen Datenbankgröße nicht lohnen.

Ähnliches gilt für die Integration einer dedizierten **Datenbank wie SQL**. Während SQL-Datenbanken für große Datenmengen ideal sind, hätte ihre Einbindung in unserem derzeitigen Stadium die Entwicklung erschwert. Für Entwickler kann es aufwendiger sein, SQL-Datenbankdateien zu lesen und Fehler direkt in der Datenbank zu beheben. Dies hätte wiederum Konsequenzen für die Entwicklung und Fehlerbehebung des Algorithmus haben können. Eine SQL-Datenbank wird erst dann sinnvoll, wenn unsere Datenmengen signifikant anwachsen.