Inhalt

Prinzipieller Programmablauf

- Die Filmwertungen einzelner Nutzer werden mit Hilfe der Methode „importData()“ eingelesen, wobei der erste String-Parameter der Pfad zu der Datei mit den entsprechenden Wertungen ist und der zweite String-Parameter der Pfad zu der Aufschlüsselungsdatei für Filme und dem jeweiligen Namen ist.
  Die importierten Daten werden in einem ein- bzw. zweidimensionalen Array gespeichert.
  - users_movies[user][movie] ist ein zweidimensionales Array, wobei der Array-Index die jeweilige ID des Benutzers bzw. des Films darstellt.
  - moviename[movie] ist ein eindimensionales Array mit dem jeweiligen Filmnamen, wobei auch hier der Array-Index die Film-ID darstellt.
- Die Funktion generateSimTable() generiert mit Hilfe der Pearson-Geleichung ein weiteres zweidimensionales Array, wobei hier ebenfalls die Array-Indizes für die Benutzer stehen. Daten werden hier doppelt gespeichert, da ein Benutzerpaar zwei Mal überprüft wird. Hier könnte eine Optimierung durchgeführt werden. Das befüllte Array lautet users_users[userU][userN]
- Um die nachfolgenden Berechnungen durchführen zu können, wird mithilfe der Funktion getOrderedNeighbours eine gewünscht große Nachbarschaft zu einem konkreten Benutzer berechnet. Hierzu wird das users_users Array an der Stelle des konkreten Benutzers durchlaufen. Für jeden gefundenen Benutzer wird ein „Neighbour“-Objekt initiiert, welches den Ähnlichkeitswert und die Anzahl der gleichen Filme mit dem Nutzer hält. Die Klasse Neighbour implementiert eine Sortierfunktion, die zunächst nach dem Gleichheitswert sortiert und anschließend nach der Anzahl an Überschneidungen, so dass Benutzer mit vielen Überschneidungen denen mit nur einem gemeinsamen Film bevorzugt werden. Eine Liste von Nachbarn kann somit dann sortiert werden.
- Mit den berechneten Werten kann nun eine Vorhersage für von einem Benutzer noch nicht bewertete Filme gemacht werden. Die Funktion getPrediction() erhält dafür als Parameter
  - Der User, dessen Abstimmung vorhergesagt werden soll
  - Der Film, der vorhergesagt werden soll
  - Die Liste mit der vorher definierten Anzahl an Nachbarn,
  - Einen Grenzwert für den Ähnlichkeitswert.

  Alle Nachbarn, die einen Mindestähnlichkeitswert haben, werden in die Berechnung mit eingenommen. Das hat zur Folge, dass die tatsächlich verwendete Nachbarschaft kleiner ausfallen kann.
- Mit dieser Methode kann nun eine Filmempfehlung erfolgen. Dazu werden für alle Filme, die der Benutzer noch nicht bewertet hat, die jeweiligen Vorhersagen getroffen, wie der Benutzer den Film bewerten würde. Dies geschieht mittels der Methode getRecommendations(), welche dieselben Parameter wie getPrediction() benötigt, da getPrediction() für jeden Film einmal aufgerufen wird. Zusätzlich kann noch die gewünschte Anzahl der Empfehlungen definiert werden.
  Für jeden Film wird ein Movie Objekt erstellt, dass die vorhergesagte Nutzerbewertung hält. Die Liste alle Filme wird dann noch der voraussichtlichen Userbewertung sortiert und die ersten X Filme, entsprechend der gewünschten Anzahl, als Teilliste wieder zurückgegeben. In der Sortierfunktion könnten weitere Parameter mit in Betracht gezogen werden, ähnlich wie in der Nachbarschaft. Für die gekürzte Liste wird schließlich noch die Methode matchMovieNames() aufgerufen, so dass jeder Film neben seiner ID ebenfalls noch den entsprechenden Namen erhält.

Programmstart

Der Einfachheit halber ist die jar-Datei nicht parametrisiert. Beim Ausführen werden 10 Filmvorschläge für den User 1 generiert sowie der MAE-Wert für eine Nachbarschaft von 525 und einem Grenzwert von 0,2 ausgegeben.

Die Dateien „u.item“, „u1.base“ und „u1.test“ müssen im selben Verzeichnis wie die jar-Datei liegen.

Auswertung

Die Messwerte liegen der Abgabe bei. Die nachfolgende Grafik visualisiert die Abhängigkeit des MAE-Werts von der Nachbarschaftsgröße und dem Grenzwert. Aus Visualisierungsgründen wurde die MAE-Achse invertiert, um eine bessere Sicht auf den Graphen zu erhalten. Die Skala geht somit tatsächlich von 1 bis 0.8.

Es ist deutlich zu sehen, dass mit abnehmender Grenzwertgröße und zunehmender Nachbarschaftsgröße bessere Vorhersagen gemacht werden können. Dadurch, dass bei gegebenen Grenzwert die tatsächliche Nachbarschaftsgröße auch bei noch so großem Parameter nicht weiter ansteigt, erfolgt keine Verschlechterung bei zunehmender Nachbarschaftsgröße.