Nicolás Risso

Sistemas Recomendadores
Tarea 1

1.0.- Análisis de los datos entregados

Los datos entregados corresponden a pares "usuario" e "ítem" con ratings enteros entre 1.0 y 10.0. La matriz generada con estos datos posee las siguientes características:

Número de Usuarios	77805
Número de Ítems	185613
Cantidad de Ratings	432171
Densidad de Datos	0.002993 %

Corresponde a una matriz de muy baja densidad, por lo que se recurrió a la representación de matriz sparse para ahorrar memoria y tiempo de cómputo.

Número de Ítems Promedio por Usuario	5.554540 ± 43.925808
Rating Promedio por Usuario	7.597847 ± 1.843557

Los usuarios en promedio evaluaron 5.5 ítems, pero se puede ver que esto varía enormemente entre los usuarios. La desviación corresponde a 43.92 y existen usuarios que evaluaron más de 1000 ítems y otros que sólo han evaluado 1 ítem.

2.0.- Implementación de los algoritmos

La estructura del código está basada en el Framework de Recomendación Crab[1] en donde se separa el motor de recomendación en un modelo, un algoritmo de recomendación y las medidas de similaridad utilizadas.

El modelo corresponde al contenedor de los datos y guarda varios datos pre calculados, entre ellos los promedios por usuarios y por ítem. El modelo implementado corresponde a una matriz en donde se mapean los datos. Debido a la densidad de la matriz, se decidió utilizar la representación sparse por columnas y por filas para los distintos algoritmos.

Los algoritmos implementados corresponden a popularidad, filtrado colaborativo basado en usuarios, filtrado colaborativo basado en ítems y slope one.

Para el algoritmo de popularidad se utilizó el promedio de cada ítem como medida de popularidad.

El filtrado basado en usuarios se construyó utilizando la correlación de Pearson ajustada con pesos calculados a base del número de ítems que ambos usuarios han evaluado. Corresponde a la correlación ajustada descrita por *Herlocker* en *"An algorithmic framework for performing collaborative filtering"* [2]. Se utilizó un parámetro k = 10.

$$Cor_{i,j} = p * Pearson(i,j)$$

$$p = \begin{cases} \frac{N}{k}, N > k \\ 1, e.o.c \end{cases}$$

Para el filtrado basado en ítems se utilizó la similaridad coseno ajustada entre ítems. La implementación se basó en *"Collaborative Filtering Recommender Systems"* [3].

El algoritmo de slope one se basó en el paper *"Slope One Predictors for Online Rating-Based Collaborative Filtering"* [4]. En este caso no se utilizó la aproximación expuesta debido a la baja densidad del set de datos.

Todos los algoritmos se utilizaron para predecir los ítems especificados y sus resultados se encuentran en archivos análogos al de predicción.

---

[1] http://muricoca.github.io/crab/
[2] http://dl.acm.org/citation.cfm?id=312682
[3] http://files.grouplens.org/papers/FnT%20CF%20Recsys%20Survey.pdf
[4] http://lemire.me/fr/documents/publications/lemiremaclachlan_sdm05.pdf

3.0.- Resultados

Para la parte 1, la predicción se realizó utilizando las medidas de slope one y las de popularidad. Se utilizó como primer recomendador slope one y en segunda instancia el recomendador por popularidad, debido a que se generan pocos ratings utilizando un solo un método.

Para la parte 2, se utilizó el mismo enfoque que en la parte 1, utilizando el mismo algoritmo para evaluar posibles recomendaciones. Para reducir los tiempos de cómputo se armaron las listas sobre los 100 ítems más evaluados por los usuarios.

4.0.- Análisis / Comparación entre los métodos

En términos de implementación, el filtrado basado en usuarios fue muy similar al de ítems. Ambos se basaban en generar las correlaciones entre todos los pares de usuarios / ítems, lo cual resultó muy costoso. El cálculo de la correlación de Pearson y la similaridad coseno se basan en encontrar un set de usuarios/ítems co-evaluados, y es en donde se utiliza el mayor tiempo de cómputo.

En cuanto a coverage, se probaron los algoritmos sobre el set a predecir:

	Predicción
Filtrado basado en usuarios	17.2 % (86 / 500)
Filtrado basado en ítems	9.8 % (49 / 500)
Slope one	48.8 % (244 / 500)
Popularidad	71.2 % (356 / 500)

El algoritmo de Slope one logra una mejor cobertura dentro del set a predecir. Además se puede ver que el set a predecir posee mucho ítems que no existen en el de entrenamiento.