Nicolás Risso
21/09/2014

Sistemas Recomendadores
Tarea 1

1.0.- Análisis de los datos entregados

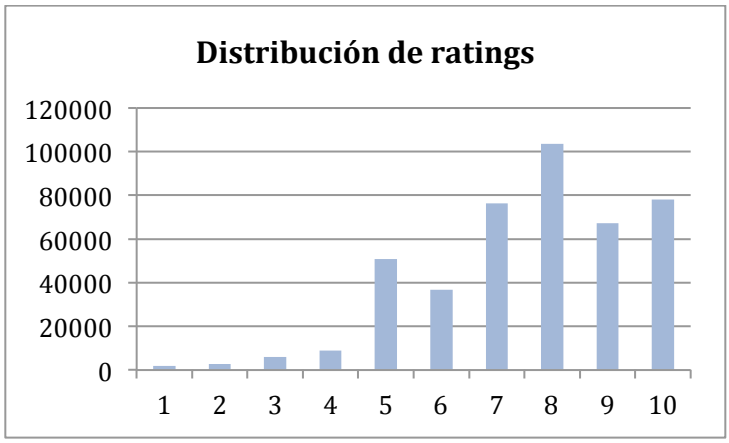
Los datos entregados corresponden a pares "usuario" e "ítem" con ratings enteros entre 1.0 y 10.0. La matriz generada con estos datos posee las siguientes características:

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 datos en diccionarios 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.



Los ratings están claramente marcados hacia los valores altos, siendo 8.0 la moda.

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 serie de diccionarios que almacenan los datos. Se decidió trabajar con dos diccionarios principales, uno de usuarios a ítems y otro de ítems a usuarios, con el fin de facilitar los distintos algoritmos.

Los algoritmos implementados corresponden a popularidad, filtrado colaborativo basado en usuarios, filtrado colaborativo basado en ítems y slope one. Todos ellos fueron implementados utilizando un modelo que es pre calculado, los cuales no están incluidos en el repositorio debido a el tamaño de dichos archivos. Los modelos llegaban a pesar 1.0 GB, y el código que los genera si está incluido en el repositorio.

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.

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

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

El algoritmo de filtrado por usuarios opera buscando vecinos más cercanos para cada usuario. El parámetro asociado a la cantidad de vecinos se estableció como 20. El modelo pre calculado de este algoritmo corresponde a la matriz de correlaciones entre usuarios.

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

---

[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

$$sim(i,j) = \frac{\sum_{u \in U}(R_{u,i} - \bar{R}_i)(R_{u,j} - \bar{R}_j)}{\sqrt{\sum_{u \in U}(R_{u,i} - \bar{R}_i)^2}\sqrt{\sum_{u \in U}(R_{u,j} - \bar{R}_j)^2}}$$

El modelo pre calculado para la recomendación basa en ítems corresponde a la matriz de correlaciones entre ítems.

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.

Como método de ponderación en slope one, se utilizó el ajuste con pesos. Cada peso corresponde a la cantidad de ratings que ha recibido cada ítem.

$$P(u)_j = \frac{\sum_{i \in S(u)-\{j\}}(dev_{i,j} + u_i)c_{j,i}}{\sum_{i \in S(u)-\{j\}} c_{j,i}}$$

Además se incluyó un parámetro para filtrar ítems con pocos ratings. Se estableció que cada ítem dentro de la predicción debe tener al menos 2 ratings. El modelo pre calculado para Slope one corresponde a una matriz de desviaciones entre items.

En caso de no poder predecir el rating del ítem, para todos los algoritmos se utilizó como valor por defecto la semi suma entre el promedio del ítem con el promedio del usuario.

$$default(u)_j = \frac{\bar{R}_u + \bar{R}_j}{2}$$

3.0.- Resultados

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

Para la parte 1, la predicción se realizó con el algoritmo de Slope one debido al análisis y comparación realizados.
Al igual que lo anterior, para la parte 2, la predicción se realizó con el modelo de Slope one debido a sus resultados.

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

[4] http://lemire.me/fr/documents/publications/lemiremaclachlan_sdm05.pdf

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:

	Coverage
Filtrado basado en usuarios	17.2 % (86 / 500)
Filtrado basado en ítems	9.0 % (45 / 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 muchos ítems que no existen en el de entrenamiento, lo que produce los errores con el algoritmo de popularidad.

Para las siguientes medidas se decidió dejar de lado el algoritmo de popularidad, debido a que está incluido como valor por defecto en los otros modelos. Además se utilizó la metodología de K-Folds para cada métrica, con número de *folds* para cada algoritmo igual a 5.

Para medir el *Accuracy* de cada algoritmo de recomendación se utilizó la medida del RMSE (Root-mean-square deviation):

	RMSE
Filtrado basado en usuarios	1.191
Filtrado basado en ítems	1.060
Slope one	0.751

Se puede ver que además de tener mejor coverage, el algoritmo de Slope one genera un error menor al intentar predecir los ratings. Esto se debe a que el algoritmo de Slope one tiene buen rendimiento con pocos ratings, y el set de entrenamiento posee una muy baja densidad de datos.

Para medir la Precisión de cada modelo se utilizó como métrica el MAP (Mean Average Precision), con listas de 10 items por usuario:

	MAP
Filtrado basado en usuarios	0.121
Filtrado basado en ítems	0.121
Slope one	0.128

El MAP el promedio de ítems relevantes encontrados por cada algoritmo. En este caso todos los algoritmos se comportaron de manera similar.

Debido a las métricas de error y coverage, se decidió utilizar el algoritmo de slope one para la parte 1 y la parte 2 de la tarea.