Pontificia Universidad Católica de Chile, Departamento de Ciencias de la Computación

Tarea 1
Sistemas Recomendadores

Santiago Larrain P.

7 de Septiembre de 2014

1. Introducción

En esta tarea se tuvo la oportunidad de experimentar con diferentes algoritmos de recomendación sobre un set de datos. Específicamente se utilizaron los siguientes algoritmos:

1. Popularidad
2. User-based collaborative filtering
3. Item-based collaborative filtering
4. Slope one

Dado los algoritmos mencionados anteriormente se quiere lograr predecir un conjunto de datos desconocidos. Esto es, recomendar un listado de 10 ítems para un conjunto de usuarios y predecir el rating de ciertos ítems desconocidos.

Este trabajo está dividido en las siguientes secciones: en la sección 2 se describirá el set de datos con que se experimentó, la sección 3 describirá los aspectos relevantes de las implementaciones, en la sección 4 se describirán los experiementos realizados, finalmente en la sección 5 se mostrarán las conclusiones y análisis más relevantes del trabajo.

2. Set de datos

El set de datos con que se experimentó está compuesto por una serie de usuarios que tienen asociado una lista de ítemes con un puntaje (*ratings*) que van desde 1 hasta 10. A continuación se muestra una serie de estadísticas del set de datos.

- $\#(Items) = 185{,}613$
- $\#(Usuarios) = 77{,}805$
- $Densidad = 0,00299\,\%$
- $RatingPromedio : \bar{r} = 7{,}5978 \pm 1{,}8435$

Además en la figura 2.1 se puede observar la distribución de las frecuencias de los ratings de los usuarios. En ello se puede ver que la mode estadística de los *ratings* es 8.

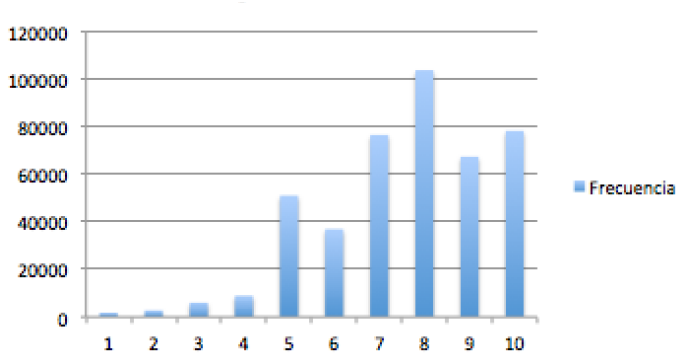

Figura 2.1: Histograma de los rating respecto al puntaje

3. Implementación

Para efecto de todas las implementaciones se utilizó el lenguaje de programación *Python*, junto con su librería de cálculo scientífico *Numpy*. Además dada la gran dimensionalidad de los datos, estos fueron representados de forma *sparse* mediante un diccionario anidado de 2 niveles, donde el primer nivel corresponde a los usuarios y el segundo nivel a los ítems de cada uno de ellos y su respectivo rating.

Como notación general se utilizará $\hat{P}_{u,i}$ el valor predecido para el usuario u y el ítem i. N_u son los vecinos más cercanos a u. El valor del *rating* que el usuario u le asignó al ítem n es $r_{u,n}$. El conjunto $R_{i,j}$ son los valores de co-rating para los ítems i y j. Por último $\bar{r}_n$ es la media de los *ratings* del usuario n.

3.1. Popularidad

Por aspectos de eficiencia y simpleza se optó por implementar la media por cada ítem para el cálculo de la popularidad. Así para cada usuario la predicción del ítem i es como se muestra en la ecuación 3.1

$$\hat{P}_i = \frac{1}{\#(U_i)} \sum_{u \in U_i} r_u, i \tag{3.1}$$

Donde U_i es el conjunto de todos los usuarios que han rankeado el item i, mientras que $r_{u,i}$ es el valor de dicho ranking.

3.2. User-based collaborative filtering [1]

Para la implementación de *User-based collaborative filtering* se utlizó la fórmula de la ecuación 3.2

$$\hat{P}_{u,i} = \bar{r}_u + \frac{\sum_{n \in N_u} \rho(u,n)(r_{n,i} - \bar{r}_n)}{\sum_{n \in N_u} \rho(u,n)} \tag{3.2}$$

Donde N_u son los vecinos cercanos a u. Y $\rho(u,n)$ es la función de similaridad entre el usuario u y n, la que corresponde a la correlación de *pearson* la que se describe en la ecuación 3.3. Por aspecto de eficiencia, las correlaciones se pre-computaron en una matriz de $n \times n$ donde n es la cantidad de usuarios.

$$\rho(u,n) = \frac{\sum_{i \in R_{u,n}} (r_{u,i} - \bar{r}_u)(r_{n,i} - \bar{r}_n)}{\sqrt{\sum_{i \in R_{u,n}} (r_{u,i} - \bar{r}_u)^2} \sqrt{\sum_{i \in R_{u,n}} (r_{n,i} - \bar{r}_n)^2}} \tag{3.3}$$

Donde $R_{u,n} = \{r_i | r_{i,u} \neq 0 \wedge r_{i,n} \neq 0\}$.

3.3. Item-based collaborative filtering [3]

En este modelo la predicción $\hat{P}_{u,i}$ (predicción del rating del usuario u al item i) fue calculada con la ecuación 3.4:

$$\hat{P}_{u,i} = \frac{\sum_{n \in N_u} \rho(i,n) r_u, n}{\sum_{n \in N_u} \rho(i,n)} \tag{3.4}$$

Donde $\rho(i,n)$ representa la similiridad entre el item i y n, la cual se muestra en detalle en la ecuación 3.5. La cual, al igual que en la sección anterior, fue pre-computada y almacenada en el disco en una matriz de $m \times m$, donde m es la cantidad de ítems.

$$\rho(i,j) = \frac{\sum_{u \in R_{i,j}} (r_{i,u} - \bar{r}_u)(r_{j,u} - \bar{r}_j)}{\sqrt{\sum_{u \in R_{i,n}} (r_{i,u} - \bar{r}_u)^2} \sqrt{\sum_{u \in R_{i,j}} (r_{j,u} - \bar{r}_u)^2}} \tag{3.5}$$

3.4. Slope one [2]

Por último, pero no menos importante, en la implementación del algoritmo *Slope One* los datos fueron representados de la misma forma que en las secciones anteriores, a diferencia de haber pre-computado una matriz de deviaciones de dimensionalidad $\#(items) \times \#(items)$.

La fórmula utilizada para la predicción de ratings está desarrollada en la ecuación 3.6

$$\hat{P}_{u,i} = \frac{1}{\#(R_j)} \sum_{i \in R_j} dev_{j,i} + u_i \tag{3.6}$$

Donde R_j el conjunto de items relevantes para el usuario u y el item i, mientras que $dev_{j,i}$ representa la deviación promedio entre los items i y j y que es calculado como se muestra en la ecuación 3.7. Al igual que el resto de los modelos, las desviaciones fueron pre-computadas y almacenadas en disco.

$$dev_{j,i} = \sum_{u \in R_{j,i}} \frac{u_j - u_i}{\#(R_{j,i})} \tag{3.7}$$

4. Experimentos

En esta sección se relatarán los experimentos realizados con los diferentes métodos, con el fin de facilitar la réplica de ellos.

Para los modelos de *User based* e *Item based* se calculó el *RMSE* y la media de *Precision@10* utilizando validación cruzada con *5-folds* para diferentes valores del parámetro K. Específicamente la división entre set de entrenamiento y de prueba para *User based* fue en base a los usuarios, mientras que para *Item based* fue con respecto a los ítems. En ambos casos, cuando el modelo no permite predecir el *rating* de cierto ítem, se optó por utilizar la semi-suma entre el promedio del usuario y el promedio del ítem. También como ítem relevantes se utilizaron aquellos cuyo *rating* sea mayor o igual a la media del usuario más su desviación estándar.

En el caso de *ItemBased*, al separar los set de datos respecto a los ítemes, el listado de *top-N* fue realizado para aquellos usuarios que tienen ítems relevantes en el set de test.

El cuadro 4.1 muestra los resultados del experimento.

	Precision@10		*RMSE*	
K	*UserBased*	*ItemBased*	*UserBased*	*ItemBased*
5	0.0255	0.0014	1.2337	1.0484
10	0.0505	0.0015	2.4743	1.0481
20	0.0752	0.0016	3.7213	1.0491
30	**0.0996**	0.0015	4.9708	1.0495

Cuadro 4.1: Resultados experimentales con los métodos *UserBased* y *ItemBased* para diferentes valores de K

Para el caso de *Slope One* se utilizó validación cruzada con *5-folds*. En este caso no se utilizó ningún parámetro. Al igual que en el caso de *ItemBased* la división para la validación cruzada fue con respecto a los ítems, por lo que al momento de calcular los top-10, éstos fueron calculados para los usuarios que tenían ítems relevantes en el set de prueba.

En el caso de no poder predecir cierto *rating* se utilizó el promedio del usuario. Los resultados sobre las métricas son:

1. **RMSE = 0.9420**
2. $MeanPrecison@10 = 0{,}00096$

5. Análisis y Conclusiones

Dado los resultados experimentales de la sección anterior, queda claro que el algoritmo con mejor desepeño es *Slope One* con un *RMSE* de 0.9420 para la tarea de predicción de *ratings*. Sin embargo *User Based Collaborative Filtering* obtuvo el mejor puntaje en la precisión con el parámetro $K = 30$. Por consiguiente en los entregables las preducciones serán realizadas con *Slop One*, mientras que las recomendaciones serán con *User Based CF*.

Una dificultad sobre la tarea due que se encontró con algunos ítemes en el listado de predicciones que no se encontraron en el set de entrenamiento, para resolver el tema se utilizó el promedio de los *rankings* del usuario. Por otro lado para los ítems que no contienen subconjunto de co-rating en común se realizó la predicción como una semi-suma del promedio del usuario y el promedio del item.

La principal dificultad de la tarea fue calcular las similitud entre usuarios e ítems para los algoritmos de *User-based collaborative filtering* y *Item-based collaborative filtering*, los que fueron de 8 hrs y 13 hrs respectivamente. Para el algoritmo de *Slop-one* también ubieron

dificultades en calcular la matriz de desviaciones para todos los ítems, sin embargo el tiempo fue de 2 hrs aproximadamente.

En el casos de *User-based* sólo se computaron las correlaciones para aquellos usuarios cuyo conjunto de co-rating fuese mayor a 1, dado que de lo contrario la correlación de pearson retorna cero o 1 y deja de ser representativo. La decisión anterior conllevo a que la implementación no requiera tanta memoria y el problema se simplificó bastante. Algo similar a lo anterior se utilizó para el caso de *Item-based*. Utilizar esta simplificación es posible sólo en caso de que la matriz R sea muy *sparse*.

La gran desventaja del algoritmo *Slop-One* fue que es necesario calcular las deviaciones para todos los pares de ítems, resultando que la matriz de desviaciones ocupó 630 MB en memoria del disco, la cual al ser cargada en *RAM* ocupó hasta 5 GB. Y fue el algoritmo que más demoraba al momento de predecir *ratings*.

Otro aspecto ha considerar es el criterio que se utilizó para determinar el conjunto de ítems relevantes para ciertos usuario. El criterio utilizado fue de aquellos ítems cuyo *rating* sean mayores a $mean + std$. Dicho criterio no es universal para todos los usuarios y se encontraron casos donde el valor de la media más la desviación estándar era superior a 10, lo que conllevó a un listado de ítems relevantes vacío.

Como último se concluye que los métodos implementados en esta tarea no son escalables y se requiere profundizar en aquellos métodos de predicción de ratings basado en modelos y no en memoria. Si bien, en [2] se jactan que el modelo es escalable en la práctica mantener una matriz de $\#(items) \times \#(items)$ no es sostenible.

Referencias

[1] Jonathan L. Herlocker, Joseph A. Konstan, Al Borchers, and John Riedl. An algorithmic framework for performing collaborative filtering. *Proceedings of the 22Nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '99:230–237, 1999.

[2] Daniel Lemire and Anna Maclachlan. Slope one predictors for online rating-based collaborative filtering. *Proceedings of SIAM Data Mining (SDM'05)*, 2005.

[3] Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. Item-based collaborative filtering recommendation algorithms. *Proceedings of the 10th International Conference on World Wide Web*, pages 285–295, 2001.