Sistemas de Recomendaciones

Taller 1: Modelos Colaborativos

Valentina Grajales Olarte, Juliana Dávila Rodríguez
Universidad de los Andes, Bogotá, Colombia
{go.laura10, dr.juliana10}@uniandes.edu.co

Reconocimiento sobre dataset de trabajo

MovieTweetings es un conjunto de datos estructurado que consta de calificaciones de películas de IMDB recopiladas de post realizados en Twitter
Este dataset consta de tres archivos que contienen toda la información estructurada:

user.dat: Contiene el id de twitter de los usuarios que votaron películas. Presenta el siguiente formato

user_id::twitter_id

items.dat: Contiene la información referente a las peliculas votadas. Este esta formado por

imdb_movie_id::movie_title (year)::gender|gender|gender⋯.

ratings.dat: Contiene el puntaje que cada usuario de a una determinada película. Tiene el siguiente formato

user_id::movie_id::rating::rating date

Después de realizar la descargar el dataset de trabajo, se procedió a dividir y formatear el archivo de rating de el siguiente modo

loaded-rating.csv: El cual contiene el 92.3% de los datos ratings y sirve para construir el modelo

unloaded-ratings.csv: Contiene el 7.7% de los datos de ratings y es usado para evaluar el modelo de predicción

El formato de estos archivos es el siguiente

user_id,movie_id,rating

Esto se realizó con el propósito de cargar el modelo de ratings en Apache Mahout y poder realizar predicciones mediante este

Construcción de modelos colaborativos usuario-usuario e item-item

a) Tanto para la construcción del modelo colaborativo basado en usuario-usuario e item-item se realizó mediante la carga del archivo **loaded-rating.csv** a Apache Mahout.

b)Para determinar el tipo de modelo que se iba usar en cada caso se hizo uso de las clases contenidas en org.apache.mahout.cf.taste.impl.recommender y las cuales fueron:

- **GenericUserBasedRecommender**: El cual implementa recomendaciones simples basadas en usuarios a partir de un Modelo de Datos(DataModel) y una vecindada(UserNeighborhood) para realizar recomendaciones.

- **GenericItemBasedRecommender**: Esta clase implementa la recomendación basa en ítems y la cual a partir de un Modelo de Datos(DataModel) y la similitud entre ítems(ItemSimilarity) para producir recomendaciones.

La predicción de ratings se realizó mediante la recomendación de item a el usuario que se está consultando, de modo que la información obtenida era representada por un Array de Arrays, los cuales contienen el id de la película y la predicción de dicho elemento.

Para la predicción basado en el índice de Jaccard, distancias coseno y correlación de Pearson se utilizaron las clases provistas por Apache Mahout, contenidos en org.apache.mahout.cf.taste.impl.similarity los cuales son en su respectivo orden

- **TanimotoCoefficientSimilarity**: Es una extensión del índice de Jaccard, el cual esta definido como el tamaño de la intersección entre las muestras dividido por el tamaño de la unión; en el índice de Tanimoto el coeficiente de distancia es una proporción del coeficiente de Jaccard:

$$T_d(X,Y) = -\log_2(T_s(X,Y))$$

donde Ts(X,Y) es el índice o coeficiente de Jaccard:

$$T_s(X,Y) = \frac{\sum_i (X_i \wedge Y_i)}{\sum_i (X_i \vee Y_i)}$$

- **PearsonCorrelationSimilarity:** Es la covarianza entre dos variables (personas o items) dividida por el producto de sus desviaciones estándar.

$$\rho_{X,Y} = \frac{E(XY) - E(X)E(Y)}{\sqrt{E(X^2) - E(X)^2}\,\sqrt{E(Y^2) - E(Y)^2}}.$$

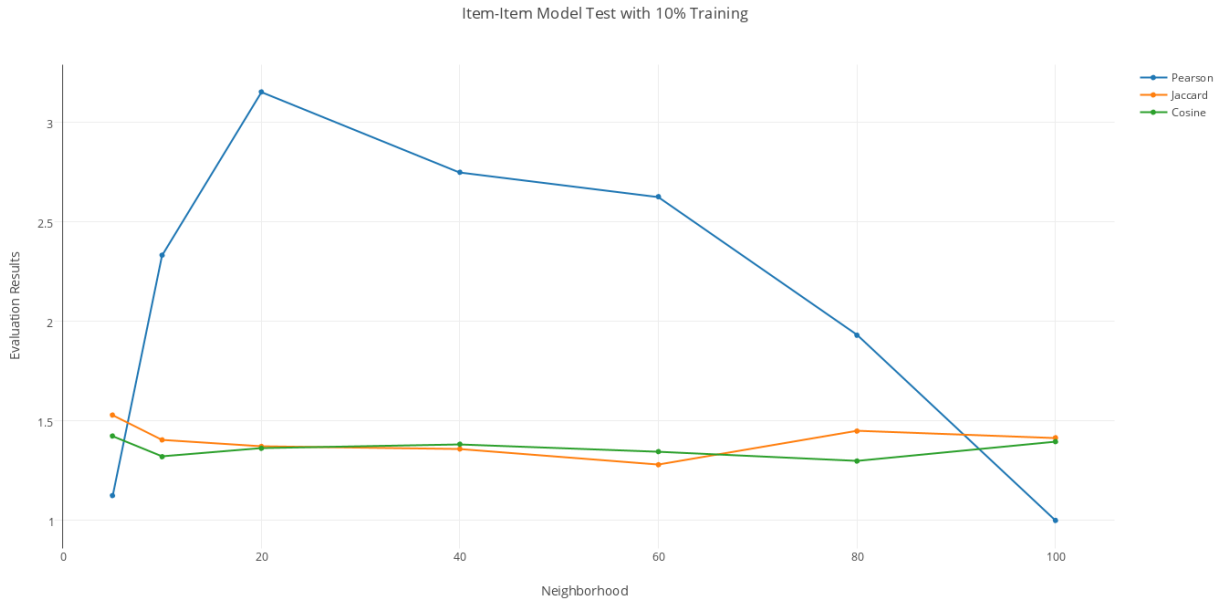
➢ **UncenteredCosineSimilarity:** Ángulo comprendido entre dos vectores que corresponden a las muestras pero ajustamos al cálculo de acuerdo a media de calificación de usuario:

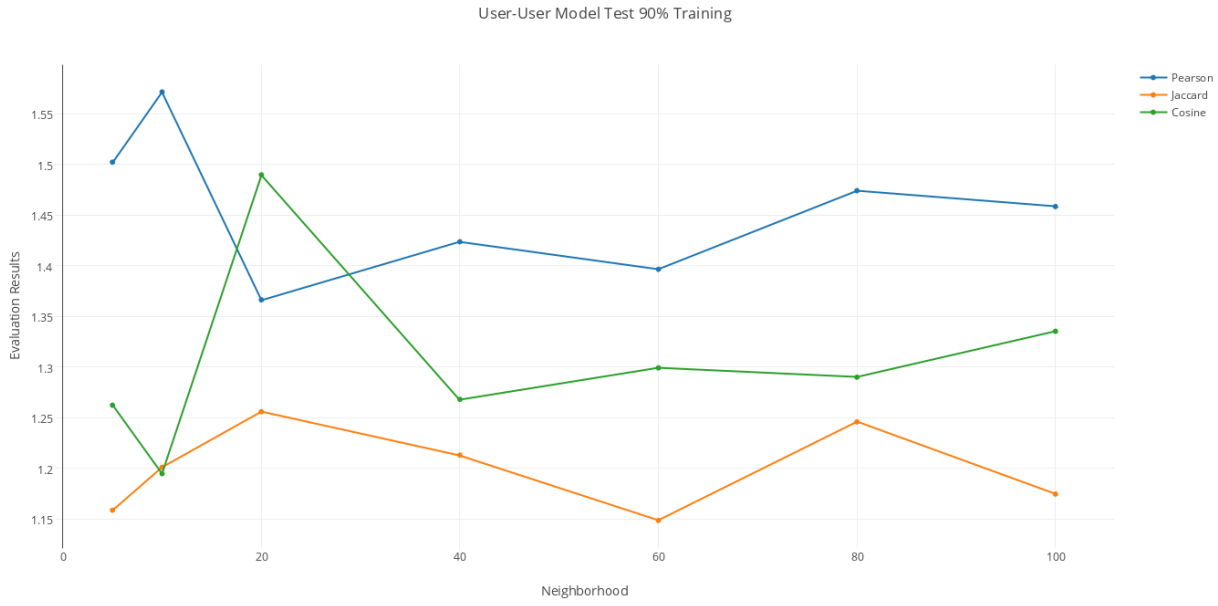
$$sim\,(i,\,j) = \frac{\sum_{u \in U} \left(R_{u,i} - \overline{R}_u\right)\left(R_{u,j} - \overline{R}_u\right)}{\sqrt{\sum_{u \in U} \left(R_{u,i} - \overline{R}_u\right)^2} \sqrt{\sum_{u \in U} \left(R_{u,j} - \overline{R}_u\right)^2}}$$

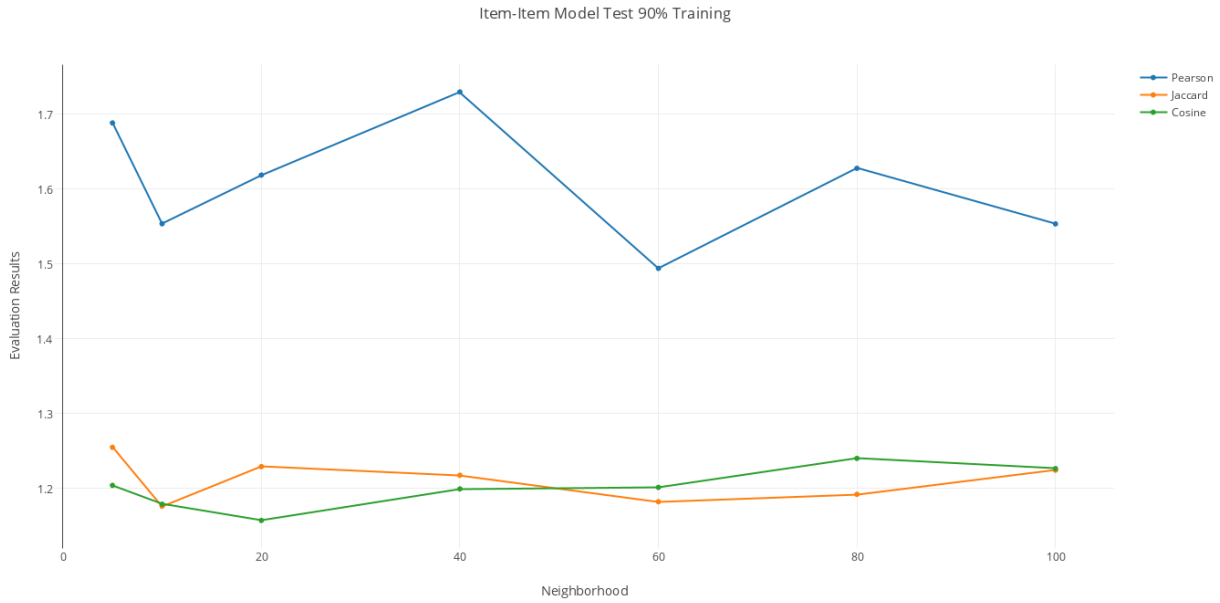
Donde $R_{u,i}$ es la recomendación dada por el usuario i al ítem u.

Para evaluar el desempeño de las recomendaciones se hizo uso de la función *evaluate*, en donde el número más cercano a 0 indica un perfecta predicción. Para los modelos basados en usuarios se estima que los números cercanos a 1 presentan un buen desempeño.

El tamaño inicial del *dataset* para la construcción del modelo de recomendaciones, afecta directamente el desempeño para realizar dichas predicciones, así como la precisión se ve afectada por el número de vecinos y el tipo de modelo que se esta usando, por lo general en los datos resultantes de las pruebas se muestra que para un mismo modelo la precisión cambia y empeora en la similitud de Pearson con mayor número de datos, en las otras dos similitudes tiende a ser constante. En las similitudes de Coseno y Jaccard tiende a tener peores resultados con menor número de vecinos. Todos los datos sobre las pruebas se adjuntan a este documento en archivos txt, así como los gráficos en archivos png.

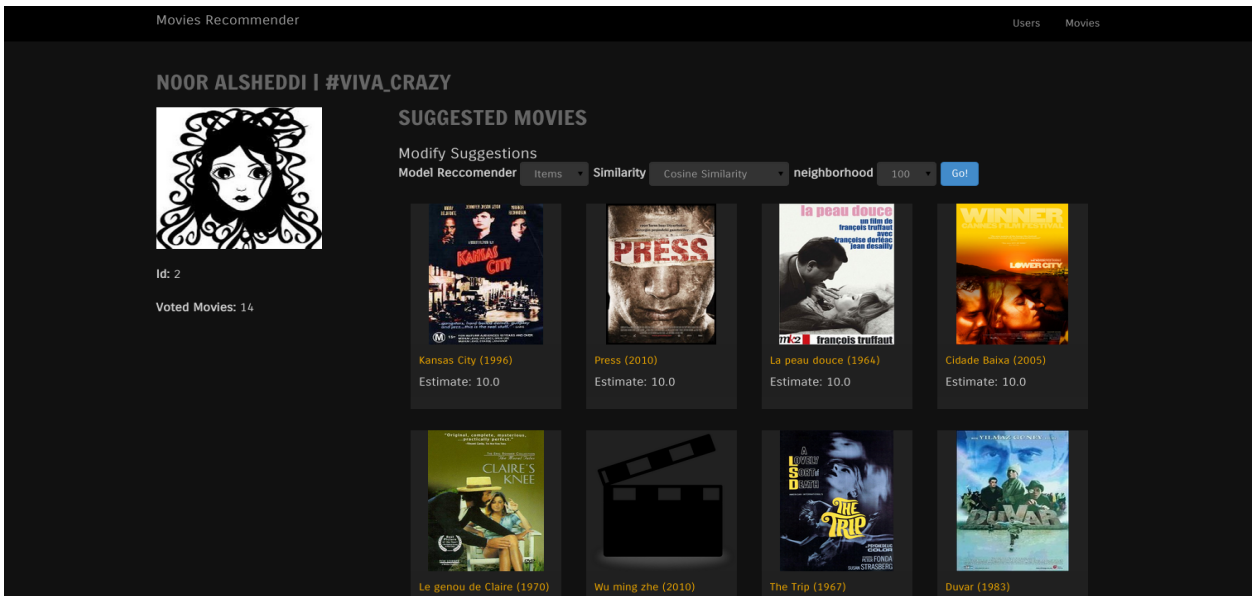

Item-Item Model Test with 10% Training
Pearson
Jaccard
Cosine
Evaluation Results
3
2.5
2
1.5
1
0
20
40
60
80
100
Neighborhood


User-User Model Test 90% Training
Pearson
Jaccard
Cosine
Evaluation Results
1.55
1.5
1.45
1.4
1.35
1.3
1.25
1.2
1.15
0
20
40
60
80
100
Neighborhood



Construya una aplicación Web

La aplicación web fue realizada en JRuby on Rails, se utilizó una base de datos en SQLite3 para consultar información relacionada con los *Usuarios* y *Peliculas,* no para procesamiento de datos. Se hizo uso de la gema jruby_mahout con algunas modificaciones para incluir todas las similitudes requeridas en el enunciado del taller.
El código del proyecto se encuentra en
https://github.com/judaro13/movies_recomendations
A continuación se muestran una toma de pantalla sobre las recomendaciones para un usuario



Para cada usuario se muestran un listado de recomendaciones, permitiendo elegir el modelo, similitud y tamaño de la vecindad para recalcular dichas recomendaciones.
Igualmente se implementó la creaciónn de usuarios, el voto de las películas, y en general los requerimientos planteados en el enunciado del taller.

El taller permitió ver las múltiples diferencias entre los modelos de recomendación basados en usuarios e ítems, así como de coeficientes de similitud. Se observó también el cómo las predicciones y recomendaciones se ven afectadas de acuerdo al número de vecinos y datos de entrenamiento.

El uso de *Apache Mahout* facilitó los cálculos al disponer de algoritmos de similaridad y predicción ingresando los datos disponibles de usuarios y películas con la respectiva recomendación. Al ingresar diversos datos para el cálculo de vecinos y de similitud se pudo observar las ventajas y desventajas de cada uno.

En general concluimos que la elección de los métodos de recomendación basados en filtros colaborativos depende en gran medida de la disponibilidad de datos (calificaciones) de usuarios y productos (películas) y del número de datos. Ya que el modelo esta basado en la relación de los usuarios con los productos es importante disponer de una conexión entre ellos; bien sea al tener usuarios que recomiendan productos comunes o *ítems* que tienen recomendaciones comunes.