**Sistemes Recomanadors**

## Tècniques de recomanació

Aquí teniu més informació sobre les tècniques de recomanació per a fer el projecte.

- *Collaborative filtering*: són estratègies que es basen en recomanar a l'usuari actiu els items que han agradat a usuaris semblants a ell. Hi ha moltes variants i algorismes. En el projecte haureu d'implementar l'estratègia basada en els algorismes "*k*-means" + "Slope One".

  La idea és usar una tècnica de clustering (agrupament) dels diferents usuaris, per tal de tenir grups d'usuaris diferents, que tenen preferències diferents sobre els items. Així doncs, primer s'haurà de classificar l'usuari actiu en un dels grups d'usuaris que es tenen. Un cop sabeu a quin grup pertany, llavors aplicareu l'algorisme "Slope One" per a estimar els ratings dels items que agraden als usuaris d'aquest grup. Finalment, ordenareu el conjunt d'items segons la preferència/utilitat estimada de major a menor.

  Anem a veure en que consisteixen els dos algorismes:

  ***k*-means**: Un dels algorismes de clustering particionals més populars és l'algorisme de clustering *k*-means (MacQueen, 1967). Començant amb una partició inicial a l'atzar de centroides de *k* clústers, explota la idea de canviar la partició actual a una altra que disminueixi la suma de quadrats de distàncies de les observacions als centroides dels clústers. És a dir, algorísmicament:

  1. Assignar aleatòriament *k* observacions com a centroides dels *k* clústers
  2. Mentre hi hagi alguna observació que canviï la seva pertinença als clústers fer:
     a) Assignar cada observació al centroide de clúster més proper
     b) Calcular els nous centroides dels clústers segons les observacions que hi pertanyen

  Evidentment, el nombre *k* és un paràmetre del mètode.

  **Slope One** (Demire and MacLachlan, 2005): aquest algorisme bàsic consisteix en estimar/predir la valoració que donaria l'usuari actiu a un item donat, a partir de les valoracions d'altres usuaris. L'algorisme proposa una funció de predicció/estimació de la forma $f(x) = x + b$ (una recta de pendent u, d'aquí el nom), que pre-computa la diferència mitjana entre les valoracions d'un item i un altre, per als usuaris que els han valorat tots dos.
  Concretament, només es tenen en compte les valoracions dels usuaris que han valorat algun item en comú amb l'usuari actiu i només les valoracions d'items que l'usuari actual també ha valorat.

  Anem a veure en que consisteix l'algorisme **Slope One**, però primer necessitem introduir la terminologia:

  $u$ és un vector amb les valoracions de l'usuari $u$
  $u_j$ és el rating/valoració de l'usuari $u$ a l'item $j$
  $S(u)$ és el subconjunt d'items valorats per l'usuari $u$
  $\chi$ és el conjunt de totes les avaluacions de tots els usuaris. És a dir la matriu de ratings
  $card(S)$ denota els nombre d'elements del conjunt $S$

$\bar{u}$ és l mitjana de les valoracions de l'usuari $u$
$S_i(\chi) = \{u \in \chi \mid i \in S(u)\}$ és el conjunt d'usuaris que han valorat l'item $i$
$P(u)$ és un vector amb les prediccions per a les valoracions de l'usuari $u$
$P(u)_j$ és la predicció per a la valoració de l'usuari $u$ a l'item $j$
Donat un conjunt de ratings $\chi$ i dos items $j$ i $i$, amb ratings $u_j$ i $u_i$ per a algun usuari $u$, tenint en compte que es vol fer la predicció/estimació de la forma $f(x) = x + b$, per a una valoració, $u_j$, es pot veure que per minimitzar l'error de la predicció, el terme $b$ ha de ser la desviació de la valoració de l'item $j$ respecte a l'item $i$, i com que hi poden haver-hi diversos usuaris que els hagin valorat tots dos, es calcula la desviació mitjana:

$$dev_{j,i} = \sum_{u \in S_{j,i}(\chi)} \frac{u_j - u_i}{card(S_{j,i}(\chi))}$$

Per tant, la predicció de la valoració per al rating $u_j$ és:

$$P(u)_j = u_i + dev_{j,i}$$

Tenint en compte no només $u_i$ sinó també les altres valoracions de $u$ es pot fer una mitjana. Per tant, finalment la predicció que proposa el **Slope One** és:

$$P^{S1}(u)_j = \frac{1}{card(R_j)} \sum_{i \in R_j} (u_i + dev_{j,i})$$

$$\text{on } R_j = \{i \mid i \in S(u), i \neq j,\ card(S_{j,i}(\chi)) > 0\}$$

Es pot fer una simplificació del càlcul quan els ratings són densos, és a dir, quan $card(S_{j,i}(\chi)) > 0$ per a quasi tots els $j$ i $i$, el qual vol dir que la major part del temps:

$$R_j = S(u), \quad j \notin S(u)$$
$$R_j = S(u) - \{j\}, \quad j \in S(u)$$

I tenint en compte que:

$$\bar{u} = \sum_{i \in S(u)} \frac{u_i}{card(S(u))} \simeq \sum_{i \in R_j} \frac{u_i}{card(R_j)}$$

Així doncs es pot simplificar el càlcul de la predicció d'aquesta manera:

$$P^{S1}(u)_j = \frac{1}{card(R_j)} \sum_{i \in R_j} (u_i + dev_{j,i}) = \frac{1}{card(R_j)} \sum_{i \in R_j} u_i + \frac{1}{card(R_j)} \sum_{i \in R_j} dev_{j,i}$$
$$\simeq \bar{u} + \frac{1}{card(R_j)} \sum_{i \in R_j} dev_{j,i}$$

Finalment la predicció aproximada seria:

$$P^{S1\simeq}(u)_j = \bar{u} + \frac{1}{card(R_j)} \sum_{i \in R_j} dev_{j,i}$$

Es pot veure un exemple a https://es.wikipedia.org/wiki/Slope_One

- *Content-based filtering*: són estratègies que es basen en recomanar a l'usuari actiu els items semblants als que li han agradat a ell. També hi ha moltes variants i tècniques. En el projecte haureu d'implementar una estratègia basada en l'algorisme "*k*-nearest neighbours (*k*-NN)".

  La idea és recollir els items que li agraden a l'usuari actiu, i llavors calcular els items més semblants als que li agraden. Els items estan descrits amb una sèries de característiques que els determinen. Així, si els items tenen característiques molt semblants llavors els items seran molt semblants. La similitud entre dos items es calcularà com una combinació (existeixen diferents mètodes per fer-la) de la similitud de les seves característiques, i la similitud entre els valors d'una característica es calcularà segons el seu tipus.

  Per a calcular els items més semblants a un donat fareu servir l'algorisme "*k*-nearest neighbours (*k*-NN)". Un cop calculats tots els items més semblants, només cal ordenar decreixentment el conjunt d'items segons el grau de similitud. S'ha de tenir en compte que, en aquest cas, la similitud s'ha de ponderar d'alguna manera amb les valoracions disponibles dels items que es comparen.

  Anem a veure en que consisteix l'algorisme *k*-nearest neighbours (*k*-NN):

  ***k*-nearest neighbours (*k*-NN)**: L'algorisme dels *k* veïns més propers a un item donat està basat en el *k*-nearest neighbours classifier (*k*-NN) (Cover and Hart, 1967; Fix and Hodges, 1951). L'algorisme fa una cerca seqüencial sobre tot el conjunt d'items *C*, guardant els *k* items més semblants a l'item donat d'entre els processats fins al moment. Al final de processar tot el conjunt d'items *C*, tindrà els *k* items més semblants a l'item donat. L'item donat, en el nostre cas serà cadascun dels items que li han agradat a l'usuari. És a dir, algorísmicament:

  1. Per a cada item del dataset d'items *C* fer:
     a. Calcular la similitud/dissimilitud entre cada item i l'item donat
     b. Guardar a cada pas els, com a molt, *k* items més semblants al'item donat
  2. Retornar el conjunt dels, com a molt, *k* items més semblants al'item donat

- *Hybrid approaches*: són estratègies que combinen tècniques basades en les dues aproximacions anteriors. La combinació pot ser de diferent maneres. Una de les més senzilles és fer la fusió/unió dels items obtinguts per l'estratègia *Collaborative-filtering* i per l'estratègia *Content-based*. Una altra consisteix en fer la intersecció dels items obtinguts per l'estratègia *Collaborative-filtering* i per l'estratègia *Content-based*.

## Avaluació de la qualitat de les recomanacions

Per facilitar l'avaluació, un dels paràmetres d'entrada de l'algorisme recomanador ha de ser el conjunt d'items del que es poden escollir les recomanacions (que podria ser el conjunt total de dades disponibles o un subconjunt). A més, l´algorisme recomanador ha de poder oferir els resultats en el mateix format text que l'entrada, es a dir, registres amb l'informació: *IdUsuari IdItem Valoració*

Els *data sets* es dividiran en dos subconjunts: el de *training* i el de *test.* Per avaluar la qualitat de les recomanacions, s'executarà l'algorisme recomanador sobre els usuaris del subconjunt de test, que es el paràmetre mencionat en el paràgraf anterior (utilitzant com entrada les dades del subconjunt de *training*), i els resultats (recomanacions suggerides) es compararan amb les dades del subconjunt de *test* (recomanacions reals) computant la mesura DCG (*Discount Cumulative*

*Gain*). Donades la llista *LR* de *p* items (ordenada per valoració) que retorna l'algorisme de recomanació, i la llista ordenada *LT* de *Q* items del conjunt de test (cadascun amb la seva valoració), la DCG es calcula com:

$$\mathrm{DCG_p} = \sum_{i=1}^{p} \frac{2^{rel_i} - 1}{\log_2(i+1)}$$

On $rel_i$ és la valoració de cada element de *LR* en *LT* (o 0 si no hi és), i *i* és la posició que ocupa cada element en *LR*. Si diversos elements tenen la mateixa valoració a *LR*, es considerarà que ocupen la mateixa posició. Més detalls a l'enllaç:

https://en.wikipedia.org/wiki/Discounted_cumulative_gain

## Referències


Cover, T.M. and Hart, P.E. (1967). Nearest neighbour pattern classification. *IEEE Trans. Inform. Theory*, IT-13(1):21–27, 1967.

Fix, E. and Hodges, J.L. (1951). Discriminatory analysis, nonparametric discrimination: Consistency properties. *Technical Report 4*, USAF School of Aviation Medicine, Randolph Field, Texas, 1951.

Demire, D. and MacLachlan, A. (2005). Slope One Predictors for Online Rating-based Collaborative Filtering. In *SIAM Data Mining (SDM'05).* Newport Beach, California. April, 2005.

MacQueen, J.B. (1967). Some Methods for classification and Analysis of Multi-variate Observations. Proc. of *5th Berkeley Symposium on Mathematical Statistics and Probability*, Berkeley, University of California Press, 1:281-297, 1967.