User Item

Algoritme:

[bron]: Mahout in action blz 43

```
for every item i that u has no preference for yet
   for every other user v that has a preference for i
         compute a similarity s between u and v
         incorporate v"s preference for i, weighted by s, into a running average
return the top items, ranked by weighted average
```

item-item amazon

```
For each item in product catalog, I1
For each customer C who purchased I1
   For each item I2 purchased by
         customer C
          Record that a customer purchased I1 and I2
For each item I2
    Compute the similarity between I1 and I2
```

Stappenplan:

1. **stap 1:Invoeren van waarderingswaarden**
    1. Bepaal welke user en welke aritkelen er mee doen
        - Alle users
        - **Alle verkochte artikelen** (niet alle mogelijke artikel)
        - Afweging 1 : neem wel of geen die maar 1 artikel hebben gekocht
    2. maak een file met : *userid, itemid, rating*
    3. Lees deze file in class UserPreference (privagte int[] itemids;private int userid; private double[] preferences)
       Aanbeveling :
        - Maak gebruik van een TreeMap (tbv snelheid en sorteren) TreeMap<Integer, UserPreference> map
        - Neem per persoon : alleen (**gesorteerd** op itemnr) de door de user aanbevolen items (met aanbeveling) op

       Programmeer tips tav UserPreference:
        1. lees basis programmeren
        2. lees User-Item
2. **stap 2: Kies een gebruiker X** (userid) waarvoor je een aanbeveling voor wilt maken
3. **Stap 3 : op zoek naar een personen met dezelfde smaak.**
   Ga de artikelen (items) van de gekozen persoon vergelijken met die van andere personen
   geef aan een Pearson correlation formule de **ratingsarray** (van alle producten) van de **gekozenpersoon** en de **andere persoon**
   u krijgt een getal terug die een maat is voor de gelijkheid (tussen de twee personen)
   Ga op zoek naar een aantal personen (K) die het beste bij de gekozen persoon passen (bijv.: hoge pearson getal) (bijv: K is 5 of 10 stuks)
   Resultaat : een list gesorteerd op overeenkomstwaarde (bijv pearson) van userids
4. **Stap 4 : op zoek naar onbekende artikelen voor gebruiker X**
   maak method die alle artikelen (itemsids) opzoekt die user X niet heeft gewaardeerd
5. **Stap 5 : gewogen waardering**
   Voor ALLE onbekende artikelen voor gebruiker X wordt voor de K personen Krijgen een NIEUWE waardering namlijke : overeenkomst (pearson) * waardering persoon Y voor het artikel
   de som van de K waarderingen worden gedeeld door de som van de overeenkosmt getallen (pesarson)

   Resultaat: de waardering die persoon X er (waarschijnlijk) aan zou geven
6. **Stap 6 : toon artikelen op volgorde van de gewogen nieuwe waardering**

Gegevens

Artikelen : 101 t/m 106
Gebruikers : 1 t/m 7
Waardering : 1.0 t/m 5.0

	101	102	103	104	105	106
1	2.5	3.5	3.0	3.5	2.5	3.0
2	3.0	3.5	1.5	5.0	3.5	3.0
3	2.5	3.0	-	3.5	-	4.0
4	-	3.5	3.0	4.0	2.5	4.5
5	3.0	4.0	2.0	3.0	2.0	3.0
6	3.0	4.0	-	5.0	3.5	3.0
7	-	4.5	-	4.0	1.0	-

Data set

dataset java vul code
Een stukje van de data verzameling :
userid, itemid, rating

```
1,101,2.5
1,102,3.5
1,103,3.0
1,104,3.5
1,105,2.5
1,106,3.0
2,101,3.0
2,102,3.5
2,103,1.5
2,104,5.0
2,105,3.5
2,106,3.0
3,101,2.5
3,102,3.0
```

Inlezen

```
0:: entrykey=1  userid=1
   (101,2.5)  (102,3.5)  (103,3.0)  (104,3.5)  (105,2.5)  (106,3.0)
1:: entrykey=2  userid=2
   (101,3.0)  (102,3.5)  (103,1.5)  (104,5.0)  (105,3.5)  (106,3.0)
2:: entrykey=3  userid=3
   (101,2.5)  (102,3.0)  (104,3.5)  (106,4.0)
3:: entrykey=4  userid=4
   (102,3.5)  (103,3.0)  (104,4.0)  (105,2.5)  (106,4.5)
4:: entrykey=5  userid=5
   (101,3.0)  (102,4.0)  (103,2.0)  (104,3.0)  (105,2.0)  (106,3.0)
```

```
5:: entrykey=6  userid=6
   (101,3.0)  (102,4.0)  (104,5.0)  (105,3.5)  (106,3.0)
6:: entrykey=7  userid=7
   (102,4.5)  (104,4.0)  (105,1.0)
```

Opzoek naar medegebruikers die bij de gekozen gebruiker passen

Kies een gebruiker : bijvoorbeeld gebruiker nummer 7
Let op : de persoon 7(de gekozenen) heeft geen waardiging gegeven aan artikel 101, 103 en106
We gaan op zoek naar gebruikers die het beste passen bij gebruiker nummer 7
HOE : pearson correlation coefficient of euclidian distance

TO DO:

1. Bepaal de ALLE artikelen die relevant zijn (in voorbeeld 101 t/m 106)
   Dus niet alle artikelen,
   je kan een deel verzameling nemen van alle artikelen die een waarding hebben
2. Bepaald gebruikers die relevant zijn (in voorbeeld 1 t/m 6)
   nummer 7 is de persoon waar je een aanbeveling voor gaat zoeken
   Tip :
   - laat alle personen vervallen die maar voor 1 artikel een waardering hebben gegeven
   - Kies de 20 of 50 of .. aantal personen die bij de gekozen persoon past

KEUZES
Artikelen : 101 t/m 107 Personen : 1 t/m 6
Persoon 7: 0, 4.5, 0, 4.5 , 1.0, 0 rating lijst

Pearson

Persoon nummer 7 vergeleken met de andere personen

user	pearson	euclidean distance similarity
1	0.9912407071619304	0.3483314773547883
2	0.38124642583151175	0.25824569976124334
3	-0.9999999999999998	0.38742588672279304
4	0.8934051474415644	0.3567891723253309
5	0.924473451641905	0,4
6	0.6628489803598702	0.2674788903885893

Pearson berekening in Excel

Verbeteringen

1. OP ZOEK NAAR DE GESCHIKSTE PERSOON:
   - Stel een drempel vast (geschiktheid > drempel)
   - Stel vast dat je opzoek bent naar **K (bijv 3,5,10) personen** die het beste bij je passen

nearest Neighbour pagina

Nearest Neighborg (k=3) & drempelwaarde
Stel K=3 (drie user die het beste passen bij user =7)
Alleen die pearson waarde opnemen die groter is dan .. (bijv: >0,35)

user	pearson K=3 and drempel >0,35
1	0.9912407071619304
4	0.8934051474415644
5	0.924473451641905

Opzoek naar verschillen met gekozen persoon

Opzoek naar artikelen die de gekozen persoon NIET heeft gewaardeerd en de anderen wel
Per persoon : userid, pearson , rating (persoon wel, gekozen NIET)
Let op : neem alleen die personen in de lijst op die minimaal 1 (extra) artikel hebben gewaardeerd

	101	102	103	104	105	106		gelijk	verschillend
1	2.5	3.5	3.0	3.5	2.5	3.0		(102,3.5) (104, 4.0) (105,2.5)	(101,2.5) (103,3.0) (106,3,0)
2	3.0	3.5	1.5	5.0	3.5	3.0		(102,3.5) (104, 5.0) (105,3.5)	(101,3.0) (103,1.5) (106,3,0)
3	2.5	3.0	-	3.5	-	4.0		(102,3.5) (104, 3.5)	(101,2.5) (106,4,0)
4	-	3.5	3.0	4.0	2.5	4.5		(102,3.5) (104, 4.0) (105,2.5)	(103,3.0) (106,4.5)
5	3.0	4.0	2.0	3.0	2.0	3.0		(102,4.0) (104, 3.0) (105,2.0)	(101,3.0) (103,2.0) (106,3,0)
6	3.0	4.0	-	5.0	3.5	3.0		(102,4.0) (104, 5.0) (105,3.5)	(101,3.0) (106,3,0)
7	-	4.5	-	4.0	1.0	-			

```
Totaal verschillende items : 101, 103, 106

userid=1 similarity =0.9912407071619304
   (101,2.5)  (103,3.0)  (106,3.0)
```

```
userid=4 similarity =0.8934051474415644
   (103,3.0)  (106,4.5)
userid=5 similarity =0.924473451641905
   (101,3.0)  (103,2.0)  (106,3.0)
```

Film wegingsfactoren

Loop alle users na
En dan per film
1) bepaalde de som (similarity * rating)
2) som (similarity)

Artikel waardering wordt Totale = som (sim* rating)/ totale som (similarity)
let op voor de artikel : Alleen bij een persoon voorkomt wordt de rating opgenomen

uitwerking voorbeeld

voorbeeld: user 7: itemid =101 sim=0.9912407071619304 rating_item_101=2,5 sim*rating=2.478101767904826

	similarity	**101**		**103**		**106**	
usesrid	**pearson**	**ranking**	**ranking * pearson**	**ranking**	**ranking * pearson**	**ranking**	**ranking * pearson**
1	0.9912407071619304	2,5	2.478101767904826	3.0	2.973722121485791	3	2.973722121485791
4	0.8934051474415644	-	-	3.0	2.6802154423246933	4,5	4.02032316348704
5	0.924473451641905	3	2.773420354925715	2.0	1.84894690328381	3	2.773420354925715
		sumpears 101	sum 101	sumpears 106	sum 103	sumpears 106	sum 106
		1.915714	5.2515211	2.809119	7.502884	2.809119	9.767466
	predicted ranking	5.2515211/1.915714= 2,7412869		7.502884/2.809119= 2,670903		9.767466/2.809119= 3,477056	

Vervolgens de artikelen sorteren op RAKING en tonen !!

dwz **106, 101, 103**

itemid=106 predictedvalue=9.767466/2.8091192=3.4770563
itemid=101 predictedvalue=5.251522/1.9157141=2.741287
itemid=103 predictedvalue=7.5028844/2.8091192=2.6709027

Update

userid =7; itemid=106 rating= 2.8
Aanbeveling voor userid =7;

	101	**102**	**103**	**104**	**105**	**106**		**gelijk**	**verschillend**
1	2.5	3.5	3.0	3.5	2.5	3.0		(102,3.5) (104, 4.0) (105,2.5)(106,3,0)	(101,2.5) (103,3.0)
2	3.0	3.5	1.5	5.0	3.5	3.0		(102,3.5) (104, 5.0) (105,3.5) (106,3,0)	(101,3.0) (103,1.5)
3	2.5	3.0	-	3.5	-	4.0		(102,3.5) (104, 3.5) (106,4,0)	(101,2.5)
4	-	3.5	3.0	4.0	2.5	4.5		(102,3.5) (104, 4.0) (105,2.5) (106,4.5)	(103,3.0)
5	3.0	4.0	2.0	3.0	2.0	3.0		(102,4.0) (104, 3.0) (105,2.0) (106,3,0)	(101,3.0) (103,2.0)
6	3.0	4.0	-	5.0	3.5	3.0		(102,4.0) (104, 5.0) (105,3.5) (106,3,0)	(101,3.0)
7	-	4.5	-	4.0	1.0	2.8			

itemid=101 predictedvalue=5.228653/1.9078536=2.7405944
itemid=103 predictedvalue=6.507851/2.4752963=2.62912

update II

Gegeven de situatie dat je van User 7 (== User X) alle pearson waarden (tov andere users) hebt bewaard.
Gevraagd:

1. een andere user (bijv user 3) dan user X (==7) doet van een item de rating updaten of geeft een nieuw item een rating (insert).
   Wat moet je doen om weer van user X (==7) een passende aanbeveling te maken?
2. User X (==7) doet van een item de rating updaten of geeft een nieuw item een rating (insert).
   Wat moet je doen om weer van user X (==7) een passende aanbeveling te maken?

GROUPLENS

informatie over grouplens zie grouplens.org (kies (allereerst) MovieLens 100K dataset) en gropuplens informatie

Gegevens : aanbeveling voor userid=20 (topK=5 , drempelwaarde=0.3)
bepalen van filter algoritme: pearson, cos of ?? Om een indruk te kunnen krijgen van de data bereken eerst de sparity Sparse data
   Is de data sparse ???
   De erbij passende filter algoritme wordt ?
Wat voor bijzonderheden kom je tegen?
Hoe meet je of data sparse is? + resultaat
filter keuze
bijzonderheden die je tegen komt

oplossingen voor de gevonden bijzonderheden

output voorbeeld : userid=20 topk=5 drempel=0.3