Recommendation Algorithms In Short

Popular Items

1. Choose no.of months to consider = "M"
2. Extract the "Item" and "datetime of transaction" over the past "M" no.of months.
3. calculate no.of sales for each item for each month for the past M months.
4. For each item : Take the worst/min no.of sales amongst the M months.
5. Sort the list according to sales [DESCENDING]
6. This list gives the popular items

Popular Items

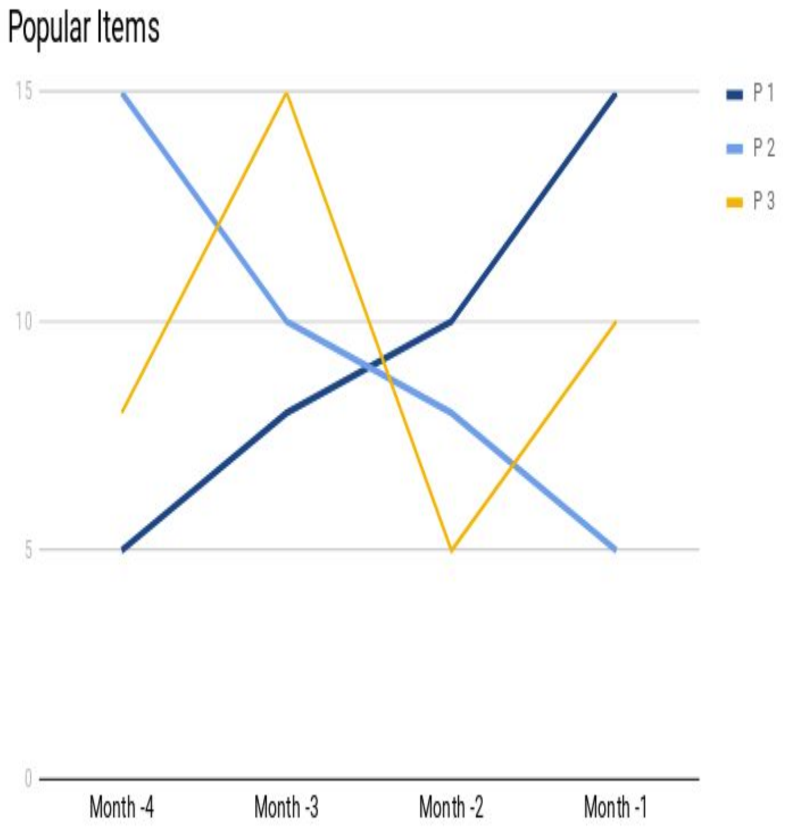
Example for m=3

Item	Sales in march	Sales in Feb	Sales in Jan
P1	10	8	13
P2	5	9	12
P3	24	7	20

Item	Rank	Score
P1	1	8
P3	2	7
P2	3	5

Limitation:

All three trends on the right will show the same popularity. But clearly there is a sense of increasing and/or decreasing popularity which is not captured in the metric used above



Similar Items

1. **Input :** List of Items with it's Attribute Name and Value
2. **Compute Score** for each Attribute:Value pair using idf

> score = (current Attribute Weight) * log(no.of Attribute:Value pair / count of current Attribute:Value pair)

3. Join back this DataFrame with Item Code to get : **[Product : Attribute:Value pair : Score]**

4. Considering **each product as a Vector** where Attribute:Value pair are co-ordinates and Scores are the magnitute , Take modulus of each product vector and divide it to make it a UNIT VECTOR

> i.e. : for each product - take root of sum of squares of scores and divide each of the score by this value

5. Now for each pair of 2 products do a **DOT PRODUCT** of their unit vectors to a final Similarity Score

6. for a product P1 sort the list by the Similarity score of (P1: P2, P1:P3, ..., P1:Pn) to get the Most similar product for P1

Similar Items : Example (find similar items for P1)

Items	Attr:Val
P1	<Color : Red >
P1	< Type : Kurta >
P1	< Size : M >
P2	<Color : Yellow >
P2	< Type : Scarf >
P3	<Color : Red >
P3	< Type : Kurta >
P3	< Size : L >

Attr:Val	Score
<Color : Red >	0.6
< Type : Kurta >	0.6
< Size : M >	0.9
<Color : Yellow >	0.9
< Type : Scarf >	0.9
< Size : L >	0.9

Items	Attr:Val	Score
P1	<Color : Red >	0.6
P1	< Type : Kurta >	0.6
P1	< Size : M >	0.9
P2	<Color : Yellow >	0.9
P2	< Type : Scarf >	0.9
P3	<Color : Red >	0.6
P3	< Type : Kurta >	0.6
P3	< Size : L >	0.9

Similar Items

Items	Attr:Val	Score (norm)
P1	<Color : Red >	0.48
P1	< Type : Kurta >	0.48
P1	< Size : M >	0.73
P2	<Color : Yellow >	0.7
P2	< Type : Scarf >	0.7
P3	<Color : Red >	0.48
P3	< Type : Kurta >	0.48
P3	< Size : L >	0.73

$\Leftrightarrow$

Unit Vectors :

P1 = 0.48 <C:R> + 0.48 <T:K> + 0.73 <S:M>

P2 = 0.7 <C:Y> + 0.7 <T:S>

P3 = 0.48 <C:R> + 0.48 <T:K> + 0.73 <S:L>

$\Downarrow$

items	Dot Product (Similarity Score)
P1 . P3	0.46
P1 . P2	0
P2. P3	0

$\Rightarrow$

Item	Similar Item	rank
P1	P3	1
P1	P2	0

Trending Items

Trends are different for different products since they have different buying frequency. So we perform following steps for all products

1. For last one year, find the number of times the product was bought, t.
2. Calculate the size of window for every product, w days
   $W = 365/t$. Where $1 <= W <= 7$
3. Temporally divide the data into train and test for the product.
   Days for training set = 6 * W, followed by
   Days for test set = 1 * W
4. Find the stats like mean(μ) and standard deviation(σ) over training set
5. Find the number of distinct user buying the product, x and calculate Z-score

   $$\mathbf{Z = (x-\mu)/\sigma}$$

   Higher the z-score, trendier it is.

Trending items

For e.g. the products - toothpaste1, televsionA7, bedsheet

product	t	w
toothpaste1	10000	1
televsionA7	18	7
bedsheet	100	3

product	mean	Std dev	z-score	rank
toothpaste1	13.16666667	5.115336418	0.3583993668	3
televsionA7	1.5	1.643167673	0.9128709292	1
bedsheet	6.333333333	2.338090389	0.7128324356	2

	W days - 1	W days - 2	W days - 3	W days - 4	W days - 5	W days - 6	W days - 1
toothpaste1	10	6	15	20	11	17	15
televsionA7	3	1	0	1	0	4	3
bedsheet	7	8	4	9	7	3	8

Frequently bought together

Rank the products according to the frequency of them being bought in the same transactions.

Transaction id	Product Id
T1	P1
T1	P2
T2	P3
T2	P1
T2	P2
T3	P2
T3	P3
T3	P1

Product 1	Product 2	No. of times bought together
P1	P2	3
P1	P3	2
P2	P3	2

Hence, for P1 the most frequently bought is P2
For P2, the most frequently bought is P1
For P3, the most frequently bought is either P1 or P2

CFII

Item-item similarity is being calculated Using co-occurrence of items in the set of a user past purchased products.

item_code	item_code1	count
P1	P2	0.8
P1	P3	0.2
P2	P4	0.7
P2	P3	0.2
P2	P10	0.1

Formula for Item-Item Similarity

Count = **(tcount/norm1) / ((norm-tcount)/(totalCount - norm1))**

tcount = count user_i bought item j

Norm1 = count of all items user_i has bought

totalCount = Total number of items

Norm = number of times item j bought by other users

Prediction is made by sum "Count" for specifies number of items.

For P2 prediction will be (0.7+0.2+0.1)

BTAB

Count number of times same set of products are purchased by different users. This shows the affinity of products to be purchased along with other product.

Detailed analysis with the help of example is given in following slides.

User’s transaction history

user	**item**
u1	p1
u1	p2
u1	p3
u2	p1
u2	p3
u3	p2
u3	p3
u4	p3
u4	p4

Item set purchased by individual users

user	**item**	**item1**
u1	p1	p2
u1	p1	p3
u1	p2	p1
u1	p2	p3
u1	p3	p1
u1	p3	p2
u2	p1	p3
u2	p3	p1
u3	p2	p3
u3	p3	p2
u4	p3	p4
u4	p4	p3

Number of times these two items are purchased by different users

item	item1	count
p1	p3	1
p1	p3	2
p2	p1	1
p2	p3	2
p3	p1	2
p3	p2	2
p3	p4	1
p4	p3	1

Number of times these items are purchased irrespective of users

item	count
p1	2
p2	2
p3	4
p4	1

Formula to calculate score for items affinity

Training users count = 4

Score =

$$\frac{(count / item1count)}{((lit(0.5)+itemcount-count) / (lit(0.1 + 4) + item1count))}$$

item	**item1**	**count**	**itemcount**	**item1count**	**score**
p1	p3	1	2	4	1.35
p1	p3	2	2	4	8.1
p2	p1	1	2	2	2.034
p2	p3	2	2	4	8.1
p3	p1	2	4	2	2.44
p3	p2	2	4	2	2.44
p3	p4	1	4	1	1.45
p4	p3	1	1	4	4.05