Contents

[Cross Sell 2](#_Toc18572432)

[Cross sell steps 2](#_Toc18572433)

[Cross sell grid for opportunity identification 3](#_Toc18572434)

[Next Best Product to Recommend Model Framework 3](#_Toc18572435)

[Affinity Analysis 4](#_Toc18572436)

[What to use? Apriori or FP-Growth?? 5](#_Toc18572437)

[Conclusion 5](#_Toc18572438)

[Models of Interest 5](#_Toc18572439)

[Matrix Factorization 5](#_Toc18572440)

[Introduction 5](#_Toc18572441)

[Association Mining 7](#_Toc18572442)

[Association Rules Mining 7](#_Toc18572443)

[support 8](#_Toc18572444)

[confidence 8](#_Toc18572445)

[lift 9](#_Toc18572446)

[Problem in our dataset for association mining 10](#_Toc18572447)

[Comparison of association mining to recommendation system 10](#_Toc18572448)

[Some potentially useful libraries 10](#_Toc18572449)

[Models under Consideration 11](#_Toc18572450)

[Simple Matrix Factorixation (Dot product of user and item features) 11](#_Toc18572451)

[Deep Recommendation System 12](#_Toc18572452)

[Recommendation System (Using Metadata) (Needs other metadata) 12](#_Toc18572453)

[Flowchart 13](#_Toc18572454)

# Cross Sell

Cross-sell involves the sale of multiple products offered by a single product/service provider to a new or existing customer. Up-sell is selling higher value products/services to an existing customer.

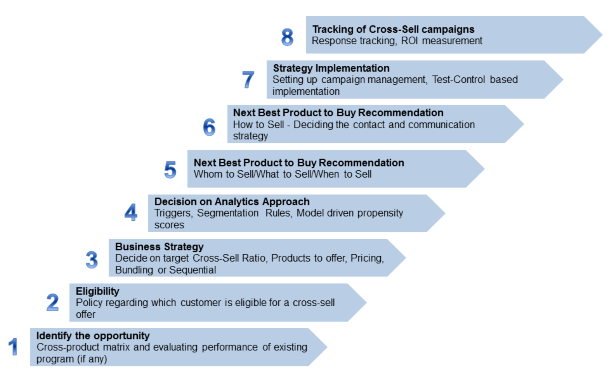
* You plan to purchase a mobile phone within a price range of Rs. 30,000(~$500). However, you eventually end up purchasing a mobile phone of Rs.42,000 ($650) because the salesman presented various other phones with fantastic features and you got swayed away with them. (This is Up-Selling).
* You plan to purchase a mobile phone worth Rs. 30,000(~$500), but the salesman offered you a charming deal of buying mobile phone with exclusive JBL headphones for Rs.40,000 (~$634) only and you again got swayed away. (This is Cross-Selling).

(from <https://www.analyticsvidhya.com/blog/2015/08/learn-cross-selling-upselling/>)

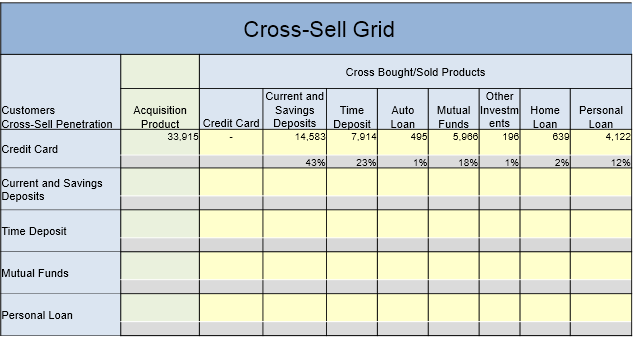
Topics to cover

* Cross-sell grids (https://www.analyticsvidhya.com/blog/2015/08/learn-cross-selling-upselling/)
* Affinity analysis

# Cross sell steps

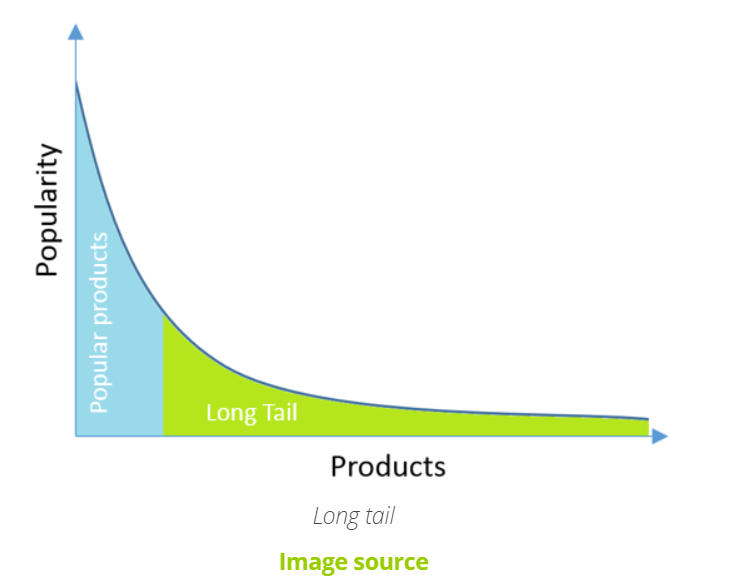


## Cross sell grid for opportunity identification



Here we can see 43% of customer having credit card have current and savings deposit. What we should be interested in is developing a cross selling engine that explores untapped opportunities. For example, home loan, auto loan is not subscribed by the people who use credit card. What we want to do is cross sell these loans to the credit card users. This is clearly an opportunity that should be exploited.

Segment those 2% customers (home loan with credit card), and look for similar customers in the customer pool. As I have already stated in the previous document. We should identify the products in long tail and suggest them to customers who are likely to buy those products.



## Next Best Product to Recommend Model Framework

Next Best Product to Recommend Models are the foundation of cross-sell targeting analytics. These encompass triggers, segmentation, regression models and optimization.

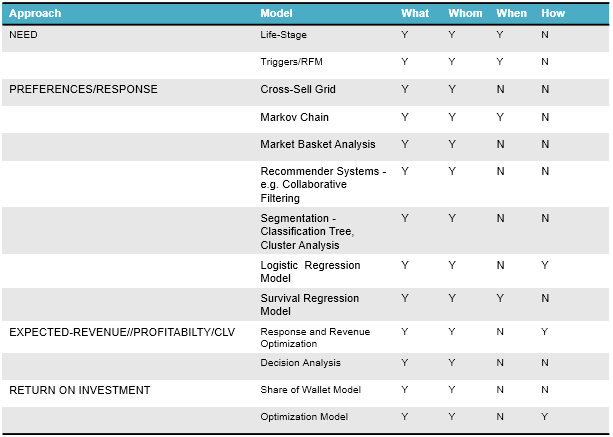
The models provide answers to the following questions:

What – choice of product

Whom – selection of customers

When – timing

How – contact strategy



# Affinity Analysis

Affinity analysis is the heart of Market basket analysis (MBA). It can discover co-occurrence relationships among activities performed by specific users or groups. In retail, affinity analysis can help you understand the purchasing behavior of customers. These insights can drive revenue through smart cross-selling and upselling strategies and can assist you in developing loyalty programs, sales promotions, and discount plans.

Market Basket Analysis is one of the key techniques used by large retailers to uncover associations between items. It works by looking for combinations of items that occur together frequently in transactions. To put it another way, it allows retailers to identify relationships between the items that people buy.

Association Rules are widely used to analyze retail basket or transaction data, and are intended to identify strong rules discovered in transaction data using measures of interestingness, based on the concept of strong rules.

## What to use? Apriori or FP-Growth??

Apriori scans the database each time for comparison. FP-growth needs just two scans to produce output. The input and output are both same for both algorithms. FP-Growth is more efficient and scalable because it does not require to scan database at each iteration. To use FP-Growth in python use FP-Growth (<https://pypi.org/project/pyfpgrowth>). To use apriori in python use mlextend ( <https://rasbt.github.io/mlxtend>).

# Conclusion

The question is cross selling different than recommendation system does not quite make sense. We can conclude that the recommendation system along with association mining are tools for cross selling. We can choose any one of them based on our needs.

# Models of Interest

The models that might interest our cross selling may be:

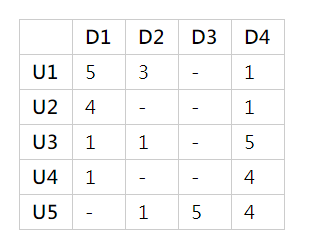
* Association mining (Affinity Analysis: a part of Market Basket Analysis)
* Classification models
* Regression Models
* Markov chain

# Matrix Factorization

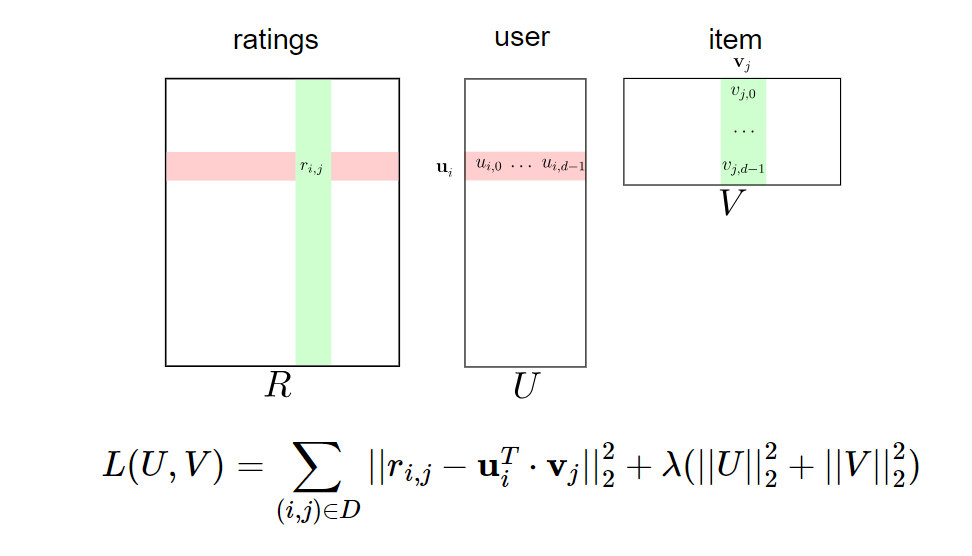
## Introduction

Just as its name suggests, matrix factorization is to, obviously, factorize a matrix, i.e. to find out two (or more) matrices such that when you multiply them you will get back the original matrix. The intuition behind using matrix factorization to solve this problem is that there should be some latent features that determine how a user rates an item. For example, two users would give high ratings to a certain movie if they both like the actors or actresses in the movie, or if the movie is an action movie, which is a genre preferred by both users.

In trying to discover the different features, we also assume that the number of features would be smaller than the number of users and the number of items. It should not be difficult to understand this assumption because clearly it would not be reasonable to assume that each user is associated with a unique feature (although this is not impossible). And anyway, if this is the case there would be no point in making recommendations, because each of these users would not be interested in the items rated by other users. Similarly, the same argument applies to the items.



We can represent users and the items like in the matrix shown above. The users have rated each item from 1 to 5. The ‘- ‘represent that the items have not been rated yet. Our purpose it to find the latent matrix that can approximate the above matrix.



* Train UU and VV on observed ratings data ri,jri,j
* Use UTVUTV to find missing entries in sparse rating data matrix R

# Association Mining

Our data set for esewa is a transactional data. The data contains the items bought by the different users. Usually the association mining is done for frequent itemset that were bought in a single purchase. But in our case, we cannot know if the purchase were done in a single purchase or multiple purchase (in esewa transactional data). The practice for the association mining is to group the orders by their order id. (The purchase with the same order id means that the items were bought in a single purchase). E.g.

|  |  |
| --- | --- |
| Order\_id | Item\_id |
| 5 | 33120 |
| 5 | 28985 |
| 5 | 9327 |
| 3 | 45918 |
| 3 | 30035 |

The order id 5 has 3 items. That means the items 33120, 28985 and 9327 were bought in a single purchase. When we start making association rules, they are grouped by order\_id.

Let us suppose that 5 orders made were as follows:

order 1: apple, egg, milk

order 2: carrot, milk

order 3: apple, egg, carrot

order 4: apple, egg

order 5: apple, carrot

Iteration 1: Count the number of times each item occurs

|  |  |
| --- | --- |
| item | occurrence count |
| {apple} | 4 |
| {egg} | 3 |
| {milk} | 2 |
| {carrot} | 2 |

{milk} and {carrot} are eliminated because they do not meet the minimum occurrence threshold.

Iteration 2: Build item sets of size 2 using the remaining items from Iteration 1 (i.e.: apple, egg)

|  |  |
| --- | --- |
| Item | occurrence count |
| {apple, egg | 3 |

Only {apple, egg} remains and the algorithm stops since there are no more items to add.

# Association Rules Mining

Once the item sets have been generated using apriori, we can start mining association rules. Given that we are only looking at item sets of size 2, the association rules we will generate will be of the form {A} -> {B}. One common application of these rules is in the domain of recommender systems, where customers who purchased item A are recommended item B.

Here are 3 key metrics to consider when evaluating association rules:

## support

This is the percentage of orders that contains the item set. In the example above, there are 5 orders in total and {apple,egg} occurs in 3 of them, so:

support{apple,egg} = 3/5 or 60%

The minimum support threshold required by apriori can be set based on knowledge of your domain. In this grocery dataset for example, since there could be thousands of distinct items and an order can contain only a small fraction of these items, setting the support threshold to 0.01% may be reasonable.

## confidence

Given two items, A and B, confidence measures the percentage of times that item B is purchased, given that item A was purchased. This is expressed as:

confidence{A->B} = support{A, B} / support{A}

Confidence values range from 0 to 1, where 0 indicates that B is never purchased when A is purchased, and 1 indicates that B is always purchased whenever A is purchased. Note that the confidence measure is directional. This means that we can also compute the percentage of times that item A is purchased, given that item B was purchased:

confidence{B->A} = support {A, B} / support{B}

In our example, the percentage of times that egg is purchased, given that apple was purchased is:

confidence{apple->egg} = support {apple, egg} / support{apple}

= (3/5) / (4/5)

= 0.75 or 75%

A confidence value of 0.75 implies that out of all orders that contain apple, 75% of them also contain egg. Now, we look at the confidence measure in the opposite direction (ie: egg->apple):

confidence{egg->apple} = support {apple, egg} / support{egg}

= (3/5) / (3/5)

= 1 or 100%

Here we see that all of the orders that contain egg also contain apple. But, does this mean that there is a relationship between these two items, or are they occurring together in the same orders simply by chance? To answer this question, we look at another measure which considers the popularity of both items.

## lift

Given two items, A and B, lift indicates whether there is a relationship between A and B, or whether the two items are occurring together in the same orders simply by chance (i.e.: at random). Unlike the confidence metric whose value may vary depending on direction (e.g.: confidence{A->B} may be different from confidence{B->A}), lift has no direction. This means that the lift {A, B} is always equal to the lift{B,A}:

lift{A,B} = lift{B,A} = support{A,B} / (support{A} \* support{B})

In our example, we compute lift as follows:

lift {apple, egg} = lift {egg, apple} = support {apple, egg} / (support{apple} \* support{egg})

= (3/5) / (4/5 \* 3/5)

= 1.25

One way to understand lift is to think of the denominator as the likelihood that A and B will appear in the same order if there was no relationship between them. In the example above, if apple occurred in 80% of the orders and egg occurred in 60% of the orders, then if there was no relationship between them, we would expect both of them to show up together in the same order 48% of the time (ie: 80% \* 60%). The numerator, on the other hand, represents how often apple and egg actually appear together in the same order. In this example, that is 60% of the time. Taking the numerator and dividing it by the denominator, we get to how many more times apple and egg actually appear in the same order, compared to if there was no relationship between them (ie: that they are occurring together simply at random).

In summary, lift can take on the following values:

\* lift = 1 implies no relationship between A and B.

(i.e.: A and B occur together only by chance)

\* lift > 1 implies that there is a positive relationship between A and B.

(i.e.: A and B occur together more often than random)

\* lift < 1 implies that there is a negative relationship between A and B.

(i.e.: A and B occur together less often than random)

In our example, apple and egg occur together 1.25 times more than random, so we conclude that there exists a positive relationship between them.

# Problem in our dataset for association mining

As stated earlier, the association mining mainly applies for frequent itemset that were bought in a single purchase. The items bought in a s single purchase has same order id. The association rules are then mined. But in our esewa dataset, we don’t have such type of data. What we have is items bought by the same users (but not in a single transaction).

But there could be a work around. Instead of grouping by order\_id (meaning that the items were bought in a single purchase), we could however group by the user\_id. But another problem can arise because of such grouping. The user may have bought the same item many times during a period of certain time. Association rules don’t take this into account. It is not concerned with how many times the user has interacted with the item.

# Comparison of association mining to recommendation system

The association mining does not seem to be favorable for the dataset that we have for esewa. The esewa does not fit into the frame of market basket analysis. The reason it is called market basket analysis is because the user adds one or more product at once in his basket (cart) and checks out. The association rules are then generated from such transaction.

Moreover, the association rules are more generalized, and do not depend on the past purchases of the users. Recommendation system (using matrix factorization) are more personalized and more effective for such datasets. But they need ratings to recommend.

|  |  |
| --- | --- |
| Association mining | Recommendation system |
| Frequent itemsets | Collaborative approach |
| No cold start problem | Cold start problem |
| No need for rating | Needs rating (explicit or implicit) |

# Some potentially useful libraries

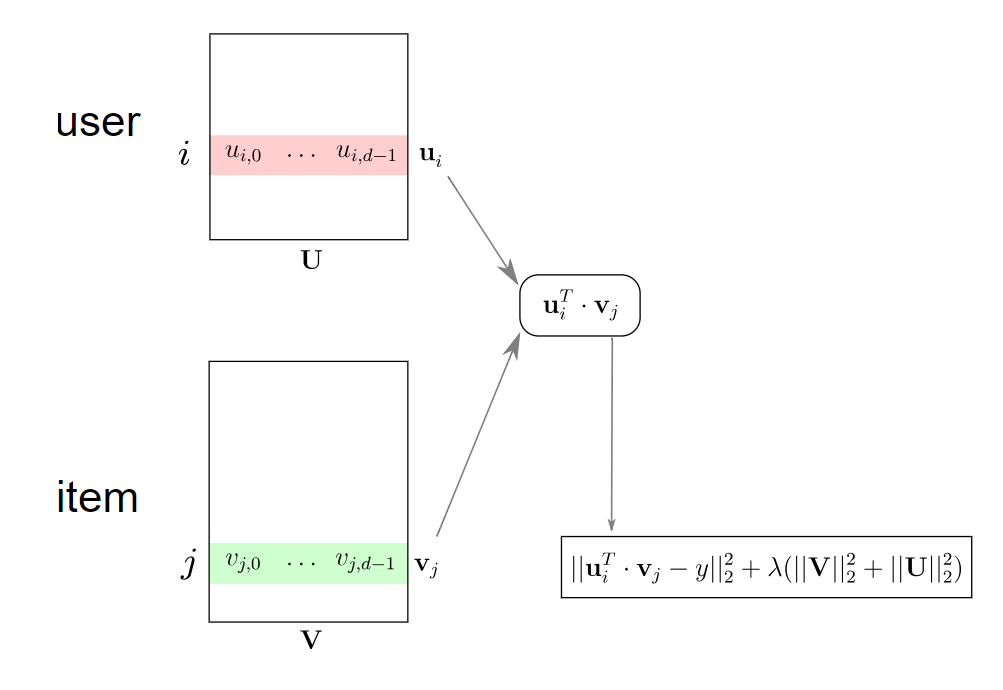
<https://github.com/enaeseth/python-fp-growth>

<https://pypi.org/project/pyfpgrowth/>

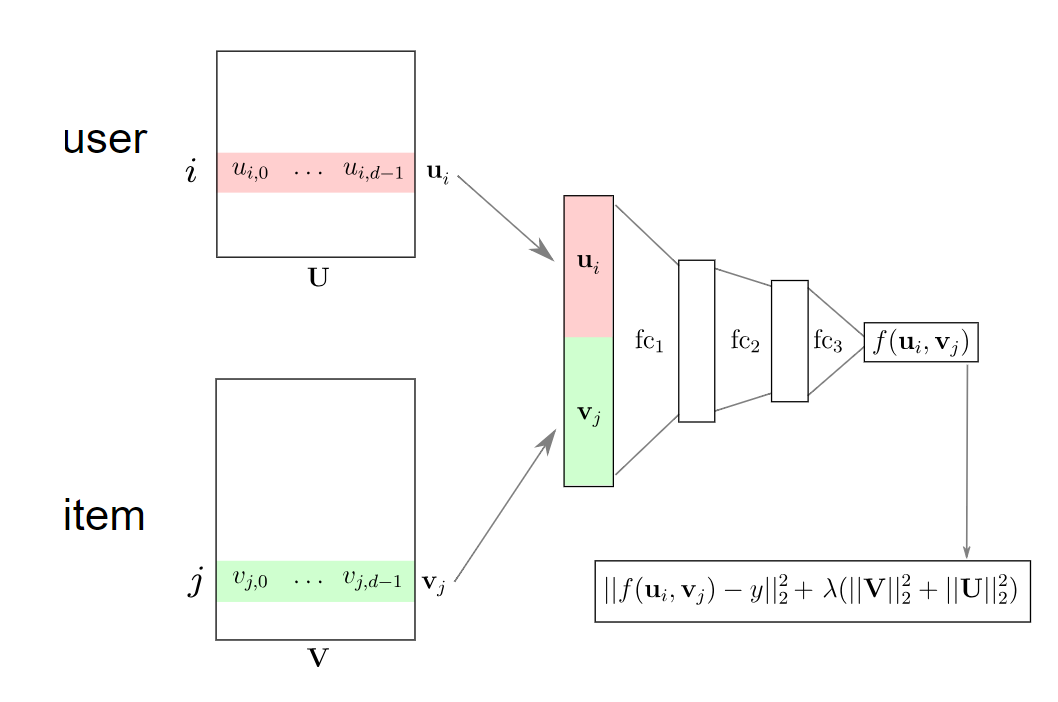
<https://www.kaggle.com/datatheque/association-rules-mining-market-basket-analysis>

# Models under Consideration

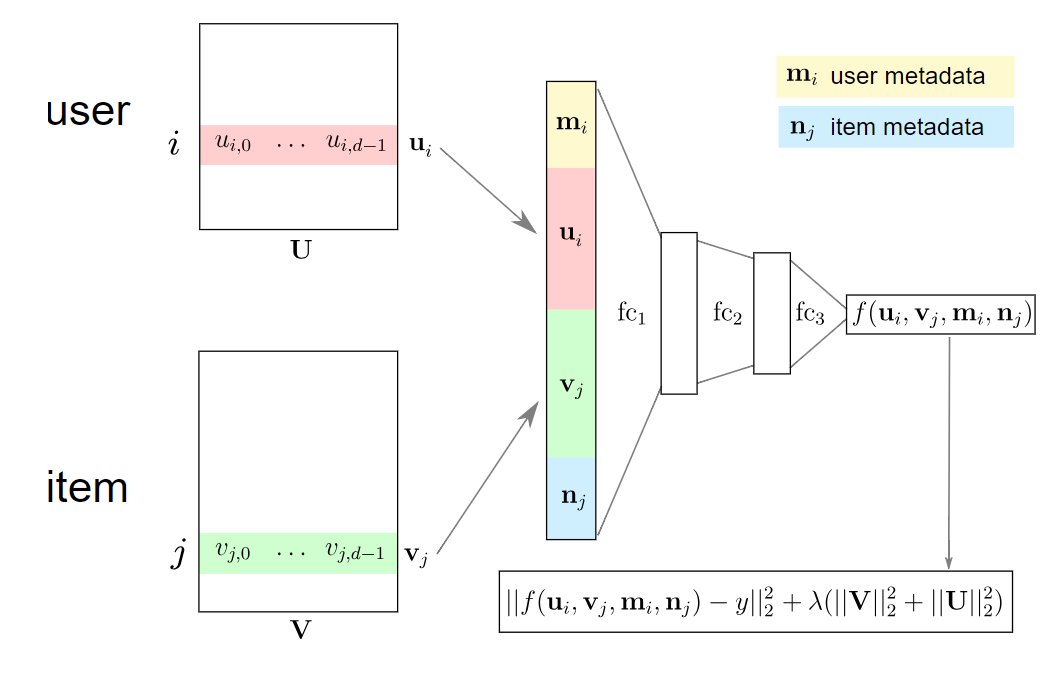
## Simple Matrix Factorixation (Dot product of user and item features)



## Deep Recommendation System



## Recommendation System (Using Metadata) (Needs other metadata)



# Flowchart

Figure 1 Data preperation

