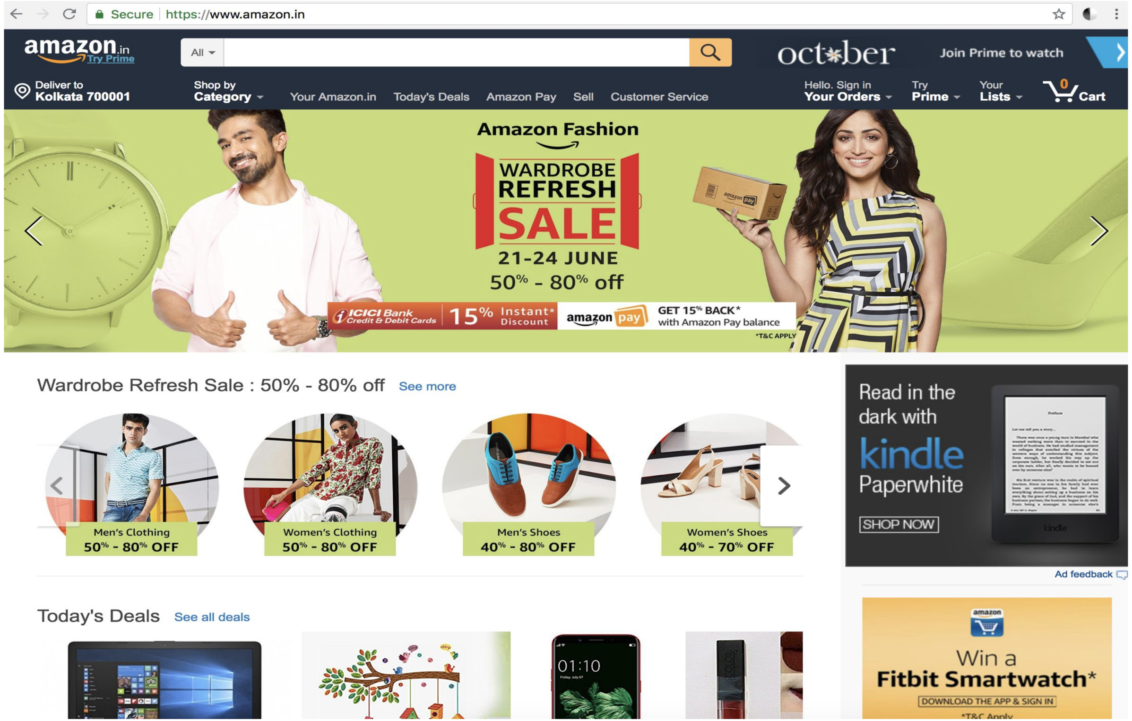
How to build recommendation systems.

An understanding

Me

- Joydeep Bhattacharjee.
- Principal Engineer at Nineleaps
- Teach machine learning at Tecmax Infotech
- Blog at https://medium.com/@joydeepubuntu/latest
- Follow me at @alt227Joydeep
- Send me an email at joydeepubuntu@gmail.com or connect with me on linkedin.

What is a recommendation engine



Definition

A system geared towards providing a holistic experience in line with the business focus of the platform and the needs of the visitor/customer.

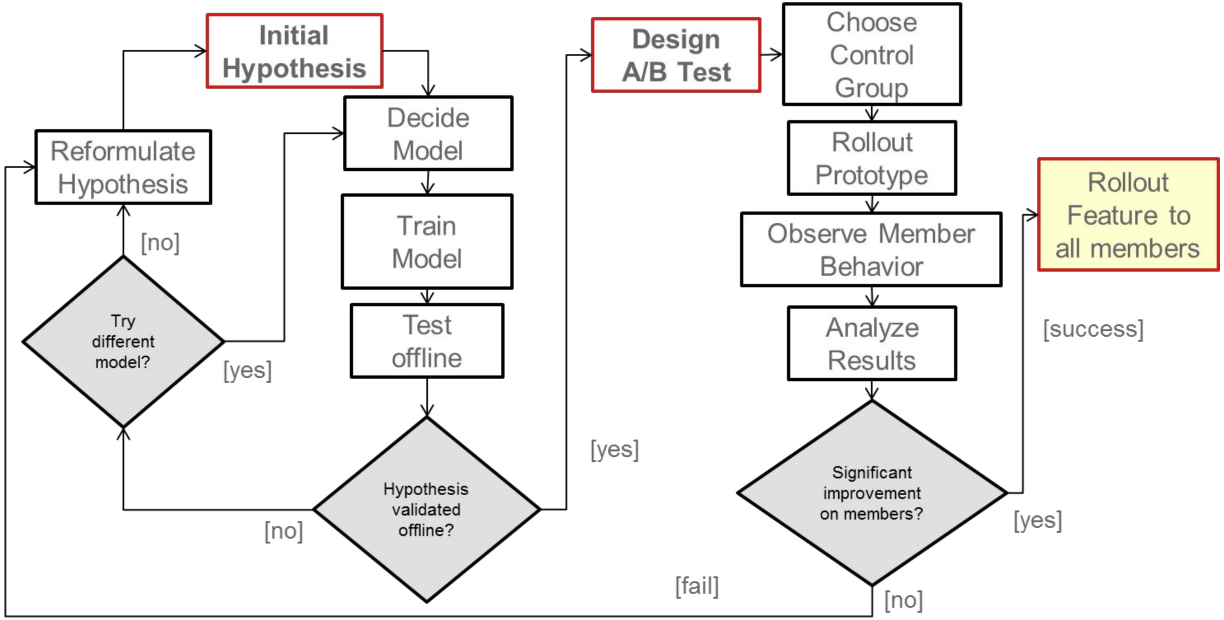
The System

Recommendation systems have two parts:

1. Feature Extraction.
2. The recommendation algorithm.

How feature extraction.

Using A/B tests.



Ways of creating a recommendation engine

Traditionally:

- Popularity models
- Creating a classifiers.

State of the art:

- Content Based filtering
- Collaborative filtering
    - User-User collaborative filtering.
    - Item-Item Collaborative filtering.
- Hybrid methods

Popularity model

```sql
select top 5 Item_code, sum(Quantity)
from customer_invoice
group by Item_code
Order by sum(Quantity) desc
```

Advantages

- Easy to incorporate into the pipeline.
- Easy to understand and maintain.
- Precision and recall can be surprisingly high in this case. (almost as high as 30%).

Simple python recommender

.

```
In [3]: nbrs = NearestNeighbors(n_neighbors=3, algorithm='ball_tree').fit(X)

In [4]: distances, indices = nbrs.kneighbors(X)
```

Adv-Disadv

Advantages

- Personalisation.
- It can even work if the work the previous history is not available.

Disadvantages

- The predictions are normally not that good based on the complexity.
- It's not scalable.

Item to Item Collaborative filtering

First choose a similarity measure. A simple and effective one is the cosine.

$sim(i, j)$ is given by

$$sim(i, j) = \cos(\vec{i}, \vec{j}) = \frac{\vec{i} \cdot \vec{j}}{\|\vec{i}\|_2 * \|\vec{j}\|_2}$$

where "·" denotes the dot-product of the two vectors.

Item to Item Collaborative filtering - II

Then comes the main algorithm

for each item in Product catalog, I1

for each customer C who purchased I1

for each item purchased by Customer C

Record that a customer purchased I1 and I2

for each item I2

Compute the similarity between I1 and I2

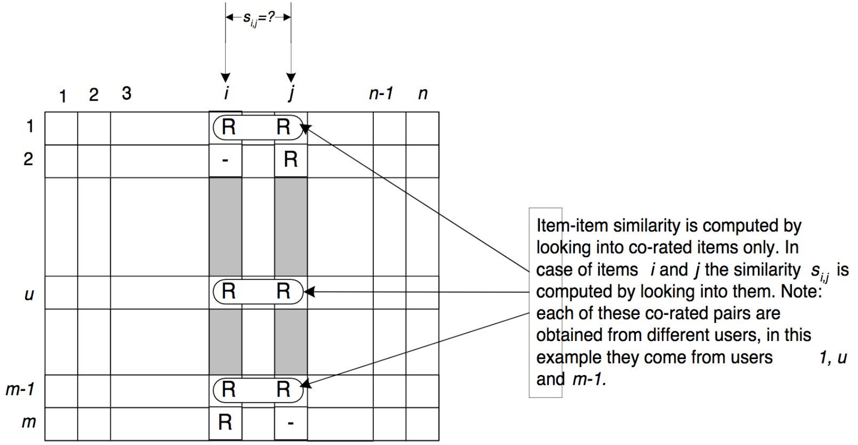


Figure 2: Isolation of the co-rated items and similarity computation

Image source: http://files.grouplens.org/papers/www10_sarwar.pdf

Over to jupyter notebook.

https://github.com/infinite-Joy/kernels/blob/master/recomender-systems/item%20to%20item%20collaborative%20filtering.ipynb

All code in this repo:

https://github.com/infinite-Joy/kernels/tree/master/recomender-systems

References

- https://www.kaggle.com/gspmoreira/recommender-systems-in-python-101
- https://www.cs.umd.edu/~samir/498/Amazon-Recommendations.pdf
- https://medium.com/@tomar.ankur287/item-item-collaborative-filtering-recommender-system-in-python-cf3c945fae1e
- http://www.cs.carleton.edu/cs_comps/0607/recommend/recommender/itembased.html
- http://files.grouplens.org/papers/www10_sarwar.pdf
- https://www.netflixprize.com/assets/ProgressPrize2008_BellKor.pdf
- https://stackoverflow.com/questions/2323768/how-does-the-amazon-recommendation-feature-work
- https://en.wikipedia.org/wiki/Affinity_analysis
- https://www.analyticsvidhya.com/blog/2016/06/quick-guide-build-recommendation-engine-python/