Context Product Recommendation - Project Plan

Prashant Kumar

May 10, 2018

1 Introduction

Recommender Systems (Rec) are "software tools and techniques providing suggestions for items to be of use to a user". Everyone encounters such systems on a daily basis when browsing the web, for example in the form of the Who to Follow suggestions on Twitter, friend recommendations on Facebook, and item suggestions on Amazon. There are various approaches to a recommendation problem which implies algorithms like popularity based, collaborative filtering based and many more. However, the inclusion of some contexts in the data could improve recommender systems that do not use contextual information. Therefore, it would be ideal to break this problem into incremental steps or phases that would be beneficial in planning out the project on a modular scale.

2 Data Collection

To be able to use the algorithms corresponding to this problem domain like Nearest Neighbor method, SVD, Matrix Factorization etc., a user-item interaction matrix or user-item-context interaction tensor is used as input. This user item interaction matrix could provide deeper insight into the analytic data if we have user data and user action added to our raw data. Since, we intend to have a more user centric approach, I believe that including User information is a more accurate representation of the problem at hand rather than just basing our implementation on item-item similarity approach without taking the item attributes into consideration.

So, first thing would be to collect all analytics data from the websites that are owned by the company and stored on a Hadoop or any big data cluster as the data would be huge from my point of view. To sum it up,

- Collect all analytics data from the websites.
- Store it on a big data cluster or cloud storage with reliability, consistency and compatibility into consideration.
- The data needs to include user information like user id, user action (viewed item, purchased item, added to cart)

3 Data Processing - Useful with Data Science Team

***** Considering we have open source focus here.***

The first step in constructing data sets from this analytics data is filtering out the events of the relevant website section. All kinds of implicit feedback such as views, clicks, buys etc that is present in the data can be relevant for the recommendation process. That is why we need to clean and process our data before we process all data so that we don't end up spending wasting resources and time on computation which is not informative. This would include :

- Importing data into a data frame format as it is easy to read for developers or any normal person. It's more human centric. The data could be in any format where we can emphasize of having the data in formats preferring JSON, EXCEL format.
- Cleaning the NULL values and empty rows.
- Doing feature engineering using correlation to figure out useful contributing column.
- Then the interactions that will be used in the interaction matrix need to be filtered out.
- When the analytics data is filtered, the next step is to get the remaining interactions in the right structure

Other important thing to be considered is how the user id is assigned to any user. I mean is the user id assigned to a user everytime he/she signs in or one user id is fixed for a company account. Also what happens when the user clears the cookies and then use it for visiting website. It will be useful to look into this matter as well and make sure we don't end up having multiple user id assigned to one user as it doesn't really help in finding a clearer picture of the underlying data patterns. This challenge is common in web based systems and we consider it to be important to be aware of these limitations in the data sets.

The interaction value in the data set could be simply set to 1 for each purchase event in the analytics data. The contextual information is gathered from the information that is present in the analytics data depending on the type of context that is used. For temporal information, the timestamp, which is the Unix time (seconds elapsed since 01-01-1970) can be used to determine the day of the week and the hour of the day that the interaction occurred.

4 Recommender System

Implementation Approach

One of the important question that needs to be answered is to build a model that is able to accommodate bigger size of datasets. As we intend to include context influence in the dataset, the computational time needed could be a thing to look into early.

If we assume that we intend to use the existing packages like Surprise, CRAB or CARSkit. The existing recommendation packages leaves out items that the user has already interacted

with when generating the recommendations lists for that user. In context-unaware recommender systems this is common practice, as the user is then already familiar with the item. In context-aware recommender systems, however, this is not as straightforward. In some cases, when a user has already interacted with an item in one context, it makes sense to recommend that same item in a different context. An example could be the recommendation of restaurants for a certain occasion. The user might have already visited a restaurant when celebrating a birthday over dinner, but that same restaurant might also be very suitable for a business lunch. It then does make sense to recommend the restaurant in the second context as well.

5 Evaluation

For evaluation, we need a set of use cases covering majority of the trends and scenarios possible.

We could separate the evaluation approach into offline and online mode where offline mode includes metrics like Precision, Recall, RMSE, MAE etc. Offline mode is usually catered to academic experiments. However, we need a method that would be suitable for business world problems and can be fruitful to the business.

While in the offline evaluation we typically use the split-validation, in the online evaluation, the A/B-testing (or multivariate testing) is the todays most prominent approach. We may integrate few different Recommendation Systems, divide the users into groups and put the Recommendation Systems into fight. It is a bit costly approach, because it consumes the development resources, so we can use the estimated difficulty of integration and future customizations/adjustments costs as one of the measures, which might a-priori reduce the pool of candidates. Now lets say we have the integration ready system and are able to divide the online users into A/B-test groups. We may either use our own hashing of their UID cookies. To do the experiment, we should determine one good place on the website/application where to test the recommendations, because we sure dont want to do the full integration of all the candidate Recommendation System early in the pilot stage. In the case where we have expected bigger traffic on the website, we may choose a conservative strategy to, for example, release only 20 percent of the traffic to the testing, keeping ourself and the rest 80 percent users safe in case some of the candidate Recommendation System would be completely broken and recommend odd stuff.

Now suppose the whole thing is up and running. The easiest measures are the Click-Through Rate (CTR) and the Conversion Rate (CR) of the recommendations. Displayed set of N recommendations 20 times, from which 3 times a user clicked on at least one of the recommended items? Then the CTR is 15 percent. Indeed, clicking is nice, but it probably led the user to a detail-page and we might want to know what happened next. But CTR and CR can be bit of a failing case sometimes which is difficult to answer without seeing the functionality of the website of the company.

In that case, we could just start a new session with empty cookies, simulate the behavior of a user and check whether the recommendations are sane. If we have a QA team, it is their expertise to get this thing done. Empiric evaluation is both complicated and easy at once. Its complicated, because it does not produce any numbers you could present on the

product board. But its also easy, because, it is human centric, we will simply recognize which recommendations are good and which are bad. If we choose oddly-working recommender, were putting ourselves into a lot of future trouble even if the CTR/CR are high at the moment.

But of course, besides quality, we should care about the Return of Investment (ROI). Simply put, we might have determined that the A/B-testing fold 1 lead to increase of X percent in conversion rate over baseline fold 0 (our current solution), that the margin was Y SEK for average successfully recommended item, and that it required Z recommendation requests to achieve that. We could do the math, project the expenses/incomes in case we put the given recommendation system on 100 percent of the traffic, integrating also into other sections of the website.

One of the most important factors to consider is Customer Lifetime Value (CLV) where we should search for nice recommendations with high empirical quality and a reasonably positive ROI.

6 Business Objectives and Risk

6.0.1 Objectives

1. BO-1 : Increase the traffic on the website by 50 percent within 6 month following the initial release
2. BO-2 : Achieve an increase on the average accuracy of the matching capability of 0.5 during the first 3 months and 1.0 within 12 months after the initial release.
3. BO-3 : Achieve an overall user experience satisfaction of 50 percent within 6 months after the initial release and increase it to maximum within 12 months after the initial release

6.0.2 Risks

1. RI-1 : Too few users might use the recommendations engine, increasing data scarcity and reducing as well the ROI.
2. RI-2 : The recommender system might recommend bad items if the items pool is not controlled.
3. RI-3 : The recommender system might raise data privacy concerns on the users sides.

Mitigation action for RI-1: In order to mitigate the problem in RI-1, the existence of the recommender engine must be known by all the users. Additionally, users must be prompted explicitly to provide their opinion on the recommendations.

Mitigation action for RI-2: As a mitigation action for RI-2, we suggest that items pools be pre-filtered in order to keep only desirable items.

Mitigation action for RI-3: As far as data privacy is concerned, measures must be taken to ensure that connections of users do not appear in the recommendations stage.