Product Recommendation for Ecommerce Website

What is the problem you are attempting to solve?

According to Statista, over 4.33 billion people access the internet globally. Many service providers and retailers have moved online and the popularity of ecommerce platforms continue to rise . Unlike physical stores, online retailers are not restricted to limited shelf space. Retailers are able to sell a wide variety of products online because the costs of logistics is lower. However, the huge amount of goods available makes it difficult for customers to navigate through their product of interest.

How is your solution valuable?

To influence the user's buying decision and increase the company's revenue, we will create a product recommender system to provide a suggested list of items with the current item being viewed.

What is your data source and how will you access it?

Retailrocket, a service provider that creates product recommendation engine for eCommerce platform, has released data from a real-world ecommerces website on Kaggle. The dataset contains three tables: events, category tree, and item properties. The events table captures visitors' behavioral data that was collected over 4.5 months in 2015. The item properties table consists of four columns (timestamp, itemid, property, and value) which provides a concatenated snapshots of 417,053 unique items for every week during the collection period. Finally, the category tree table consists of 1,669 rows and two columns with a categoryid and its corresponding parentid. Not all categoryid have a parentid.

The files has been downloaded from Kaggle and stored it in Google Drive.

What techniques from the course do you anticipate using?

During the initial stage of data preprocessing, users with no activity and items with only a single view or activity will be removed. User activity such as views, add-to-cart, and transaction will be used as ratings. Then, we'll use Scipy library to create a user-item/item-property sparse matrix. The matrix will be split into a train and test set.

Since we are working with implicit feedback in our dataset, we'll use LightFM, a python recommendation algorithm. It also incorporates item and user properties of collaborative filtering into a matrix factorization to reduce dimensionality.

For evaluation, we will use classification accuracy metrics, like ROC AUC, to assess the recommendation algorithm ability classifying relevant and irrelevant items. Recall will be weighted more than precision since we are most interested in the probability that a relevant item is recommended.

What do you anticipate to be the biggest challenge you'll face?

This will be the first time I'm building a recommender system from scratch so there is a lot of information I must process. First, the dataset needs to be cleaned and transformed into a sparse matrix. Then, I must create a train and test set for modeling. Another challenge to consider is resolving the shortcomings of the recommender system, such as making suggestions for new items or users in the cold-start problem.