AML Group 29 Final Report

Zhining Qiu, Ran Pan, Tao Wang, Smaranjit Ghose, Zhenyu Yuan

1. Background and context

With the rapid growth of the Internet, more and more famous retail brands begin to start their own online stores and E-commerce sites. However, with an extensive and comprehensive selection of products for customers, people often have trouble finding what really interests them. Therefore, designing an efficient recommendation system is imperative for every E-commerce website for their customers. In this project, we would like to design and implement a product recommendation system for H&M Group, a multinational clothing company that provides fast-fashion clothing for people in all age groups, in order to help their customers to locate H&M products of interest faster and more accurately, eventually achieving customer satisfaction.

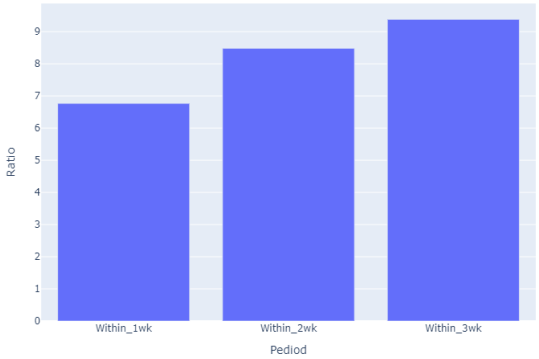
Our goal is to provide a top-12 collection of articles for each customer which they are most likely to buy based on their previous shopping experience. Customers can get what they possibly want and clerks can easily find recommendations for valued individuals.

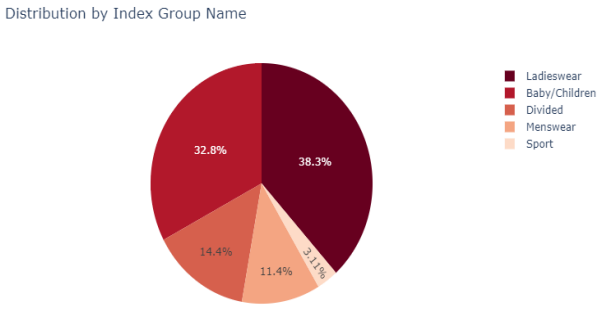
2. Exploratory Data Analysis

We retrieved data from the Kaggle H&M Personalized Fashion Recommendation competition. We have 31.8 millions of transaction histories from 2018 to 2020. Those transactions include 1.4 million unique customers and 100 thousands of articles. Our data preprocessing includes missing data processing, multicollinearity detection, customer purchasing patterns prediction, and dominant variable identification.

Type casting was performed on specific attributes such as age, status,etc for each of the csv files to reduce the total memory used while operating on them. For the missing data in the dataframes, using custom imputations was found to be most effective. Using median or mean for handling the missing values of attributes such as price and age was avoided since our recommendation system would be highly sensitive to these features given that different age groups have different preferences and budgets for apparel shopping. Hence, for such observations of missing values, the corresponding input vectors were dropped.

After removing highly correlated or repeated columns, our customer table only has 4 variables: **customer_id, cms_encode, fnf_encode, age.** We also retrieve customer purchasing patterns as 6.8% of the customers tend to purchase the product with different color/size in the next 1 week. 8.5% of customers would again purchase it in the next two weeks and 9.4% of customers would purchase the same product with different color/size again in the next three weeks. That feature may also be representative in later model training.





Dominant categories for the product group with the maximum number of transactions is Ladieswear followed by Baby/Children on a close second. We could add a weighted percentage for each category within our model and test whether dominant variables are significant.

3. Model Methodology

- *Content based filtering*

  First of all, we have decided to apply a content based filtering algorithm using cosine similarity as the selection metric. A content-based filtering technique filters information (items) based on the correlation between a user's past behaviors & preferences and the content of the items. Compared to a collaborative filtering system, which simply chooses recommendations based on the correlation between people with similar interests, a content-based filtering algorithm considers the content of the items. Specifically in this project, we first created a user-feature matrix by joining the transactions dataset with the user information dataset. Then, we created an item-feature matrix by merging the transaction dataset with the item (article) dataset. Due to the high dimensionality of the dataset, we performed dimensionality reduction using PCA on the user feature dataset as well as the item feature dataset. Then, based on the cosine similarity method, we have calculated the similarity scores of items for every user and we can select the top-12 similar items for recommendations for every customer.

- *Transformer-based model*

  This method treats customers' purchase sequence like a text sequence.

  During preprocessing, each customer's purchase history was converted into a sequence of articles like ["Trouser 51", "Shoe 231",...]. Each article was then mapped into higher dimensional space via several trainable embedding layers through features of the item (colors, type, etc.). The article embedding is then concatenated with an embedding of purchase week. Thus, each purchase history is turned into a tensor of shape (length, embedding), and this is the input to the model. This is analogous to tokenization and embedding in NLP.

  For each customer, the prediction target is simply their purchase history shifted forward by one index. This is a tensor of shape (length, 1), each item being the index of items.

  Then, the input embedding goes through a transformer encoder layer (where attention for future items are not allowed), a dense layer, and a softmax layer. This yields an output of shape (length, num_items), each row a probability distribution over the items. Then, the model is trained based on the cross entropy loss between the output and the target.

  For prediction, each customer's full purchase history is fed to the model, and the model takes the top 12 items for the customer's next purchase.

  This model trains very slowly, because the number of items is about 100,000 (most operations are just softmaxs) and the number of customers is above 1,000,000. About 1.5 training epoch will use up the quota of a Kaggle GPU session. We don't have access to multi-GPU training, so we can only afford to train for one epoch. (Based on experience from the NLP world, such tasks usually take 15 epoches to converge). No doubt this will limit the model performance.

- *K-means clustering*

  For the k-means clustering, the pre-processing steps are similar to that of the transformer based model. THe data is first normalized with Standard Scale and min-max before feeding it to the algorithm. Due to the larger number of dimensions of the combined data frame of transactions and customers, certain attributes like postal code are dropped while pca is used for other columns.

- *AlternatingLeastSquares*

  This method assumes we can get a sparse matrix to present the relation between users and the products they want to buy, and we can factorize it into two factor matrices, one presents the

features of users and another presents features of products. By multiplying these two factor matrices, we can get the original sparse matrix.
To get the optimal factors matrix, we apply the least square method alternately to the two factors matrixes. In each iteration, we make one of the two factors matrixes constant and apply least square to another matrix and update it
When the training is finished, we can calculate the similarity between products according to
The factors matrix represents features of products and we can recommend the 12 articles with the most similarity to the customer.

4. Model evaluation

To evaluate the performance of the algorithms used for the recommendation system of apparels, the metric of Mean Average Precision is used. For our study, we are interested in the relevance of the first 12 suggestions as well as their relative order since beyond 12 suggestions, a customer is unlikely to look for more options for the choice they already made given the time constraint while shopping, especially on e-commerce platforms.

$$MAP@k = \frac{1}{U}\sum_{u=1}^{U}\left(\frac{1}{min(r,k)}\sum_{n=1}^{k}P(n)\cdot rel(n)\right)$$

The above equation can be interpreted as the computation of average of the precisions upto the kth prediction considering only the ones where rel(n) is valid and then taking a mean over the predictions for all the users in the test/train data.

The table given below summarizes the performance of the experiments performed:

Model	MAP@12
Content based filtering	0.0215
K-Means	0.0210
Transformer based	0.0028
AlternatingLeastSquares	0.0042

For the four above algorithms, we could see that content based filtering and K-means clustering provide similar results of 0.021, while the other algorithms have less significant results.
The transformer based model, which is the most complicated, does not perform well. As discussed above, this model is under-trained due to limits in computation power. It might perform better if sufficiently trained, but we simply don't have the resources to investigate further.

5. Conclusion

Recommendation system for these H & M articles is quite hard to generate. In this dataset, we are lacking user behavior or user demographic data. We could possibly generate a more robust model with more features from users. All our models are predicting based on a large portion of product side attributes. Our optimal model is the content based filtering with a score of 0.0215. The K-means model has the performance of 0.021. We could also see that models like alternating least squares do not have a great prediction based on the given data.