Fashion recommendation system

1st Jordano Isaias Garza-Murillo
Computer Science Master's
Tecnológico de Monterrey
Monterrey, Nuevo León, México
A00819506@itesm.mx

2nd Carlos Axel García-Vega
Computer Science Master's
Tecnológico de Monterrey
Monterrey, Nuevo León, México
A01754346@itesm.mx

3rd José Alberto Montán-López
Computer Science Master's
Tecnológico de Monterrey
Monterrey, Nuevo León, México
A01375993@tec.mx

***Abstract*—Fast fashion companies, such as H&M, are trying today to have recommendation systems in which they predict the garments that a customer can buy according to their tastes and preferences. The objective of this study is to carry out this recommendation algorithm using machine learning techniques from which they will be compared to know which is the one that best approximates reality. To this end, the research question is as follows: Can you predict what the customer would like to buy in the future based on their history of past purchases? This question is answered by experiments within the data provided. This can be translated into the methodology used in which it consists of performing an exploratory analysis of the data and deciding which are the most important variables in which they will serve as parameters for the machine learning techniques such as neural networks, K-Nearest Neighbor, Principal Component Analysis, K-means, among others. The results will be obtained by applying the aforementioned and comparing according to the evaluation metrics of each method to decide which is the one that best suits this challenge.**

***Index Terms*—Machine Learning, Benchmark, H&M, fashion recommendation**

I. Introduction

The present research concerns the issue of a system of recommending garments to customers for a large company such as H&M. Recommendation systems are, basically, a software system that filters information of interest to the user to propose as wisely as possible the product most appropriate to their needs. Based on previous behaviors, such as purchases in a certain store, these systems evaluate the degree of interest of this user in certain products and automatically searches for similar products with a high probability of attracting their attention.

Personalized recommendations find application in various areas and are usually found in all those web services (web stores, online audiovisual transmission services publications) that offer many objects, whether books, clothes or movies, of which only a part could be interesting for the user. The recommendations would be a way to help the user find those pieces of their liking among all those that make up the global catalog of articles, since they make a pre-selection based, for example, on previous searches of the user.

These automated proposals mean, without a doubt, a relief for consumers, because, instead of having to go through an endless list of irrelevant offers until they find what they are looking for, they help them, in theory, to break down the most interesting of the least interesting. Website owners, for their part, expect this positive effect to be reflected in an increase in traffic or sales. In eCommerce, good recommendations always lead to full shopping carts that expand profit margins. However, the other side of the coin of mere calculation from mathematical algorithms lies in the lack of the human component. Even the most refined calculations fail in very basic human behaviors, which means that, sometimes, the proposals received are not quite what the consumer expected or make him even despair.

it is important to note that these systems are one of the most present applications of Artificial Intelligence in our day to day. Who does not open their Netflix, Amazon or Youtube account every day and enjoys (or suffers) from the recommendations of new content or products. Of course, these systems work on all online content or sales platforms, being booming and constantly evolving, due to the great benefits they report.

Now speaking of the research specifically, was carried out with an in-depth analysis of the data using EDA (Exploratory Data Analysis) techniques, which will be explained later in section B. This serves to know the nature and behavior of the data and see if it is necessary to modify, add, delete, change, among other actions of the samples. Similarly, for the recommendation system, machine learning techniques will be used to track the movement and calculate the preferences of users towards the garments that are sold in the store, page or mobile application of the H&M company.

The main objective of this project is to predict which garments the customer is more biased to buy according to the purchase history that he has. The output of the algorithm will be a table with two columns in which it describes the customer and the garments that he would tentatively be buying.

II. Method and Data

In this section we describe the data used to solve the problem, as well as the exploratory data analysis (EDA).

A. Data

H&M provides three datasets for the fashion recommendation task including articles, customers, and transactions. In addition, they also provide a set of images for each product including the color variant.

The first dataset ”articles” contains the metadata for all the articles the shop offers. In the dataset we can find several attributes such as the id of the product, the product code, the name, the group it belongs, color group and color, section and group name, and other attributes regarding the description of each product. Attributes in the dataset are duplicated since are represented as a numerical value and with a string for the human understanding. The data is important because it contains the information of each article that can be used to determine what to recommend to the customer.

The second dataset contains information about the customers. This dataset has seven attributes related to the customer, these are the following: customer_id that is the unique identification of each client, age, postal code, club member status, fashion news frequency and Fashion News (FN), and the status of the customer. Most of the attributes are categorical data and some of them like the club member status are binary.

The last dataset contains records of customer purchases such that each instance corresponds to a single article purchase. For instance, for a customer purchase of two articles, two rows would be registered on the dataset. The dataset contains five attributes including the date of purchase, customer id, article id, the price of the article, and the sales channel id that can be online or physical store.

Table I summarizes the number of instances and attributes for each dataset.

TABLE I
DATASETS DIMENSIONS.

	#instances	#attributes
Articles	105542	25
Customers	1371980	7
Transactions	31788324	5

All datasets are related to each others so we can create a large dataset that combines all the information and train the model, but before we need to explore the data in detail.

B. Exploratory Data Analysis

EDA is a critical phase in the data mining process which gives us a more detailed insight of the data we will be working with. As we seen in Section II-A, since data is segmented into three datasets, first we explored them separately.

1) Articles: The unique attribute that presents missing values (exactly 416) is “detail_desc” which is a brief description of the product, however, we did not get rid of these instances since do not really affect the purpose of this work. We perform a deeper visualization on some attributes to see the percentage of articles that contain a specific value (See Appendix A). We may observe that most of the articles belongs to garment upper body group, followed by garment lower body group and garment full body group (See Fig. 11). Most of them have a solid graphical appearance (See Fig. 12) and the most prevalent colours are black, dark blue and white (See Fig. 13).

It is worth mentioning that this dataset present some paired categorical-numerical attributes, for instance, “product_type_name” and “product_type_code”.

After doing correlation analysis the results are provided as a heatmap plot (See Fig. 10) where “article_id” and “product_code” are perfectly correlated, thus we dropped the last to avoiding have information overload.

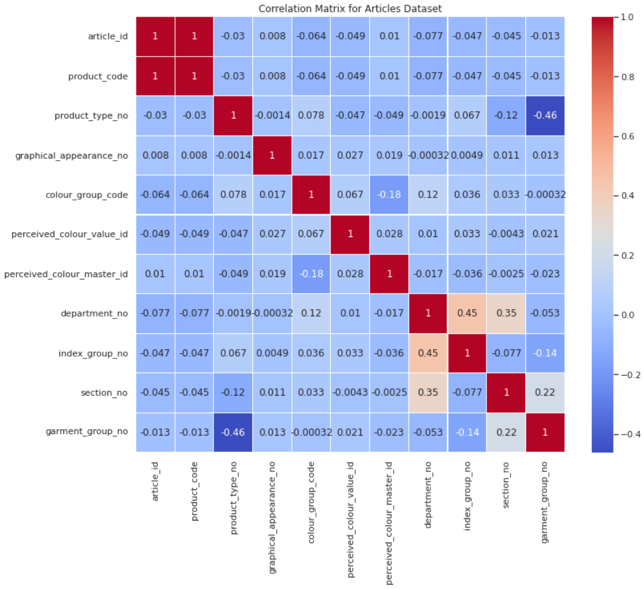


Fig. 1. Articles covariance matrix.

2) Customers: The only information related to the costumer is its age. Figure 2 shows the distribution of ages regarding all customers. Customers in mean are 36 years old.

After find out missing values in “FN” and “Active” attributes these were actually related with non-register customers, therefore we changed this missing values for numerical meaning register customers or not. Once customer dataset is properly modified, covariance matrix (see Fig. 3) shows that there are three attributes sufficiently correlated to get rid of two.

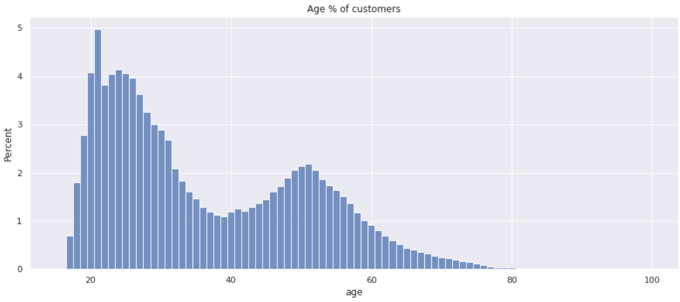


Fig. 2. Costumer ages distribution.

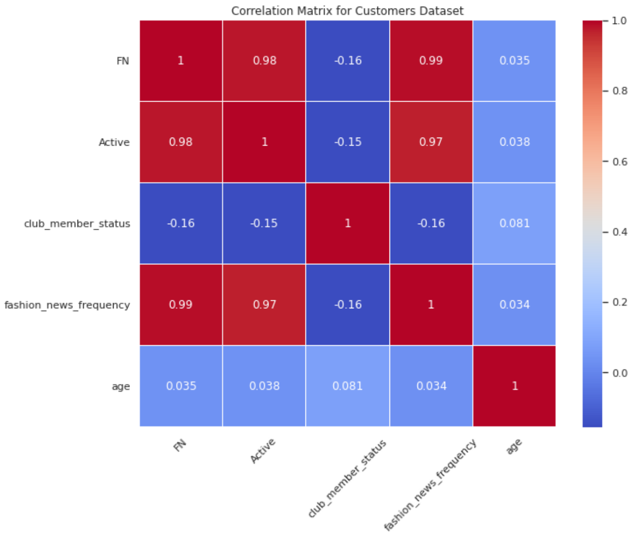


Fig. 3. Customer covariance matrix.

3) Transactions: Transactions dataset does not have missing values, also correlation matrix (see Fig. 4) shows that there is not any strong correlation among their attributes.

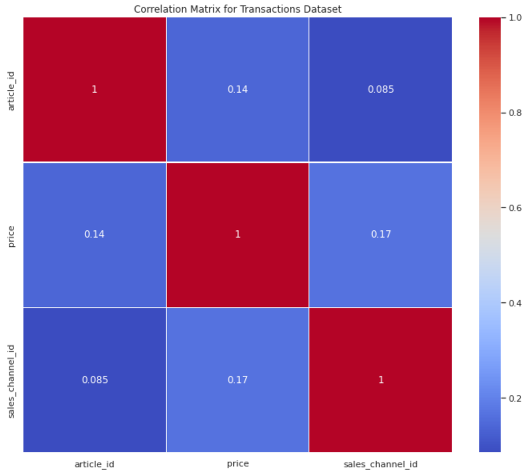


Fig. 4. Transactions covariance matrix.

C. Modeling

Modelling users' preferences on items based on their own historical data might lead us to a personalized recommender system, therefore, it was decided to use two different models to perform the recommendation system, these were Neural Collaborative Filtering (NCF) and Bilateral Variational Autoencoder (BiVAE), both based on collaborative filtering technique. Each of these models will be described in more detail below:

1) NCF: This algorithm, based on deep learning networks to tackle the collaborative filtering key problem in recommendation on the basis of implicit feedback [5]. Unlike some algorithms based on the same technique where they restored to matrix factorization and applied an inner product on the latent features of users and items; NCF replaces this inner product with a neural architecture. This idea arises since the inner product simply combines the multiplication of latent features linearly and this will not be enough to map the complex structure of user interaction data.

The limitation of the MF is easier to appreciate in Figure 5. This drawback lies in the fact that if we changed the values on the vector u_4 to make it closer to u_1, that is p'_4, we are incurring a large ranking loss.

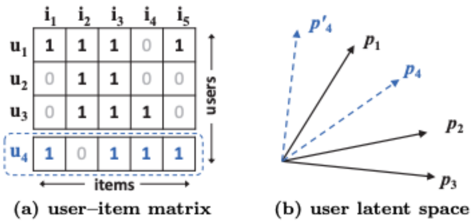


Fig. 5. Matrix Factorization limitation.

Due to this limitation, for the general NCF framework (see Fig. 6) was proposed an instantiation of NCF by using a multi-layer perceptron (MLP) to learn the user-item interaction function, then the neural matrix factorization which ensembles MF and MLP.

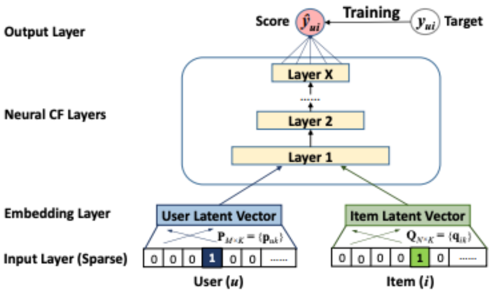


Fig. 6. NCF general framework.

Specifically, before embedding both neural networks, user and item latent vectors are mapped with a function that makes an element-wise product of them, and then the activation non-linear function of the edge weights of this product allows the MF be more expressive than the linear MF model. Therefore, authors chose the sigmoid activation function and it learns the edge weights data with the log loss. They called this MF as Generalized Matrix Factorization (GMF).

2) BiVAE: A collection of users, a set of things, and a set of interactions, such as ratings, clicks, and purchases between some user-item pairings, make up preference data in collaborative filtering (CF). Preference data is often represented as an interaction matrix between the user and the object. It's a type of dyadic data in which measurements are linked to pairs of items derived from two separate groups of objects. Users (row-wise) and items (item-wise) are the two methods to display such data (column-wise) [3].

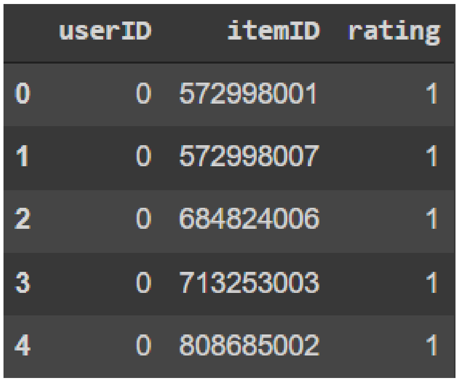
Finally, we want to find presentations for both ends of the dyadic data (users and elements) that, when combined, can explain the affinities between the user and the item. In the area of CF, latent factoring or factorization models are commonly used to achieve this goal. The simplicity, efficiency, efficacy, and flexibility of the latter are key factors in their success. However, because it can only capture linear patterns in both data and latent spaces, this group of models is recognized to have limited modeling potential.

To get over this issue, non-linear neural-based techniques have lately sparked a lot of attention. Notably, the Variatonal Encoder (VAE) [4] model has recently been applied to CF with significant performance increases over various competing techniques. The fact that VAE is probabilistic is one potential explanation for its outstanding outcomes on the CF challenge. Indeed, VAE differs from neural networks in that it does not strive to learn deterministic forms, but instead learns distributions across these representations, permitting it to account for latent space uncertainty. This characteristic is very useful when working with sparse data or data with few observations. [3].

Despite its impresive performance, VAE was built for vector-based data and so does not fully capture the two-way character of dyadic data. Only users are in the User-VAE, whereas items are handled as characteristics in a feature space of users, and the same is true for the Item-VAE. Because of the incompatibility between VAE and the two-way structure of preference data, it's unclear how to expand such a model on the item side in a reasonable way, for example, to include side information like item textual descriptions, photos, and so on.

Bilateral Variational Autoencoder (BiVAE) [3] is presented as a solution to the disadvantages of VAE outlined before. It combines a generative model of user-item interactions (or dyads) with a pair of inference models (user- and item-based, correspondingly) parameterized using multilayer neural networks in a single framework to auto-encode dyadic preference data. The suggested BiVAE is ”bilateral” in that it handles users and things equally, making it more suited to two-way or dyadic data than the vanilla VAE. In comparison to traditional one-sided variational autoencoders, BiVAE can capture uncertainty on both sides of dyadic data, which improves its resilience and performance on sparse preference data.

III. Data Preparation



	userID	itemID	rating
0	0	572998001	1
1	0	572998007	1
2	0	684824006	1
3	0	713253003	1
4	0	808685002	1

Fig. 7. First five elements of the training dataframe. In this case for the user indexed as “0”.

As mentioned in the modeling section, both methods use a set of users and items to generate a recommendation for a new item. This condition requires us to use only the transaction dataset that contains information about each purchase a customer has made. We are interesting on the customer_id and article_id columns to create the input data for the models but before we need to filter the data.

We created both training and test sets based on the dates of the purchase such that recent dates correspond to the test set and the rest of the data correspond to the training set. Transactions are from 2018-09-20 to 2020-09-15, two years of data, however we are limiting the data to use just the last complete month for training models and two weeks for the test ,thus training data uses August 2020 and the last 15 days of the data are used for test set. Afterwards, we filter the training set to take only those customers that have purchased at least 10 different articles such that we have users with more information to be used. The last step was to match users and items on both data sets due to a new item, not used for training, can create troubles on predictions. We start with 254163 customers and end up with 13462 users as well as 19962 items after the data filtering process.

Figure 7 shows a dataframe used for training models, columns are userID that contains the number of the customer, itemId with the unique number of each article, and rating that in this application represent if the customer have purchased that article any time.

A. Experiments setup

For both models it was performed 5 experiments tuning their hyper-parameters. In the case of NCF model, the only tuned values were the number of epochs, the learning rate, and the batch size. On the other hand, besides already mentioned parameters, for BiVAE model, we tuned the latent dimensions and the encoder dimensions. Experiments on both models are shown in Table II and III, for NCF and BiVAE, respectively.

IV. Results

Models generate a data frame with user ID, item ID, the score of the prediction, and the rank of the item to be recommended (see example on Figure 8).

We use Mean Average Precision at k, Precision at K, and Recall at K metrics to compare models and analyze the

TABLE II
NCF PARAMETER TUNING FOR EACH EXPERIMENT.

Epoch	Learning Rate	Batch Size
5	0.001	64
5	0.005	128
10	0.009	128
15	0.0001	256
20	0.01	64

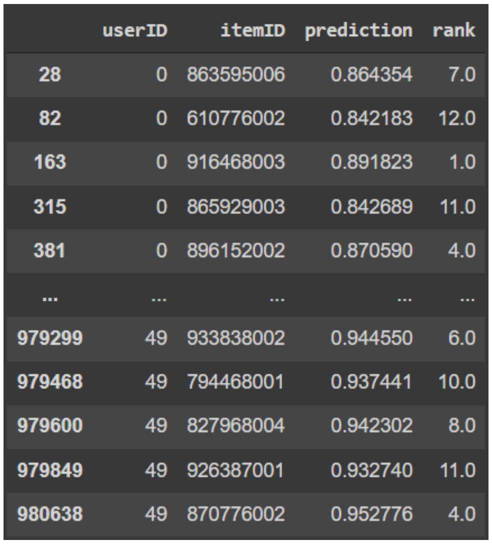
TABLE III
BIVAE PARAMETER TUNING FOR EACH EXPERIMENT.

Latent dim	Encoder dim	Epoch	Learning Rate	Batch size
50	100	5	0.001	128
40	90	10	0.005	64
60	110	15	0.009	256
100	200	20	0.01	512
5	10	5	0.0001	64

performance of each one. We used $k = 12$, i.e., we selected the top 12 items to recommend based on the score, for computing performance of both methods as well as 50 different customers to be tested (see Table IV).

Also, we can see this metrics results in boxplots. The five execution were gropued in one boxplot for every metric there is, in other words we will have three boxplots for each algorithm.

The training time is of equal importance since it gives us a guideline of which method requires more waiting time and which is faster with the aforementioned configurations and the computational power that was available.



```
         userID     itemID  prediction  rank
28            0  863595006    0.864354   7.0
82            0  610776002    0.842183  12.0
163           0  916468003    0.891823   1.0
315           0  865929003    0.842689  11.0
381           0  896152002    0.870590   4.0
...         ...        ...         ...   ...
979299       49  933838002    0.944550   6.0
979468       49  794468001    0.937441  10.0
979600       49  827968004    0.942302   8.0
979849       49  926387001    0.932740  11.0
980638       49  870776002    0.952776   4.0
```

Fig. 8. The output of the model containing the user Id, item ID, value of prediction, and the rank of the item to recommend.

TABLE IV
MODEL EVALUATION. BEST PERFORMANCE IS SHOWN IN RED-BOLD.

Model	mAP@12	Recall@12	Precision@12
NCF	0.0081	0.0357	**0.0117**
	0.0056	0.0278	0.0083
	0.0030	0.0145	0.0100
	0.0024	0.0132	0.0083
	0.0006	0.0035	0.0033
BiVAE	0.0043	0.0305	0.0067
	0.0058	0.0243	0.0050
	0.0133	**0.0412**	0.0067
	0.0008	0.0025	0.0017
	0	0	0

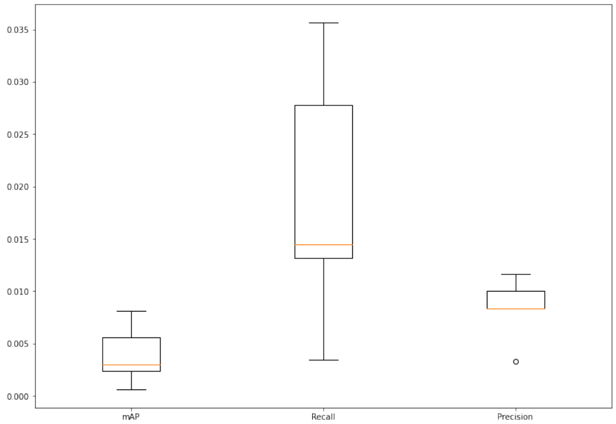


Fig. 9. Metrics boxlpots for NCF

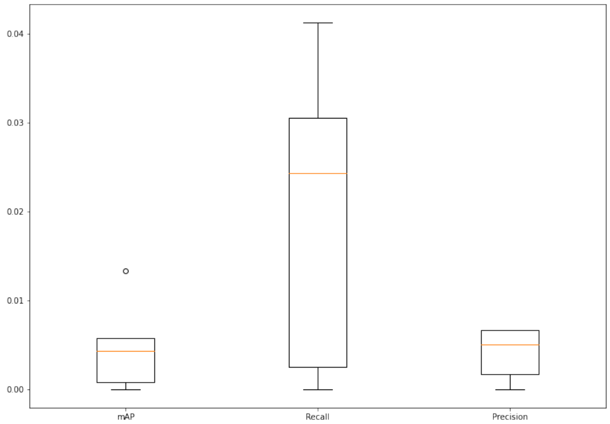


Fig. 10. Metrics boxlpots for BiVAE

V. DISCUSSION

Previous results showed that third experiment with BiVAE performed better on mAP@12 and recall@12 metrics, moreover, NCF first experiment, performed better on precision@12. In average, BiVAE performed better than NCF.

Now speaking of times, the NCF algorithm had a range, in seconds, from 1465 to 5270 taking into account all the settings mentioned above. On the other hand, the BiVAE

algorithm had a considerable decrease in this time range, being these from 237 to 829 seconds.
As can be seen, 2 of the 3 metrics that performed best were from BiVAE, so there is evidence that this method is more efficient and faster.

VI. Conclusion

This work was done to contribute to a problem or real-life situations that a company could have to improve its sales. A recommendation system is essential to automate the way advertising is displayed to customers. The methods described in this research could be said to be relatively new, a development by Microsoft in which anyone can make use of them.

As a team we decided to apply these two algorithms (NCF and BiVAE) to create the recommendation system with information from the H&M company to corroborate if it could be compared with the existing methods used for this type of system. Given the results explained in this work we can conclude that they are indeed a viable option when choosing an algorithm to recommend products to customers according to their purchase history. The evaluation metrics between the traditional methods and these two new ones are similar. The way in which this could be improved, and as future work would be to use more data, but as a requirement to this more computational power would be needed when training the model.

Complete EDA and code used to train the models can be found in the following github repository: https://github.com/Mercurio005/H-M-Recomender.git

A. Future work

Many other methods, tests, experiments of recommendation systems have been left out due to the time that was given to us in addition to some requiring more computational power than we had available. In total, within the repository that has just been published by Microsoft, there are 33 different algorithms that use different types of methodologies to recommend such as content based filtering that uses item features to compute the recommendation. From this total, we chose two that were the ones that best suited our problem of the company HM, NCF and BiVAE.

We would like to implement this problem with other types of algorithms to see if there are improvements in the mAP evaluation metric and make a comparison of these.
In the same way the issue of computational power is something that we would like to solve, since in the market today there are computers, for example students from Tecnologico de Monterrey have access to a computer called NVIDIA DGX-1 System with eight Tesla V100 GPUs which could allow us to have enough computational power, or cloud services (google colab, amazon web services, among others). That will allow us to run the program with more inputs since this is a very important issue when doing in training because it directly affects the evaluation metric, that is, if we have vast information to train, our system will have more chances to predict correctly, resulting in a higher mAP.

References

[1] Provost, F., & Fawcett, T. (2013). Data Science for Business: What you need to know about data mining and data-analytic thinking. " O'Reilly Media, Inc.".

[2] Salah, A., Truong, Q. T., & Lauw, H. W. (2020). Cornac: A Comparative Framework for Multimodal Recommender Systems. J. Mach. Learn. Res., 21, 95-1.

[3] Truong, Q. T., Salah, A., & Lauw, H. W. (2021, March). Bilateral variational autoencoder for collaborative filtering. In Proceedings of the 14th ACM International Conference on Web Search and Data Mining (pp. 292-300).

[4] Liang, D., Krishnan, R. G., Hoffman, M. D., & Jebara, T. (2018, April). Variational autoencoders for collaborative filtering. In Proceedings of the 2018 world wide web conference (pp. 689-698).

[5] He, X., Liao, L., Zhang, H., Nie, L., Hu, X., & Chua, T. S. (2017, April). Neural collaborative filtering. In Proceedings of the 26th international conference on world wide web (pp. 173-182).

Appendix A
Article Visualizations

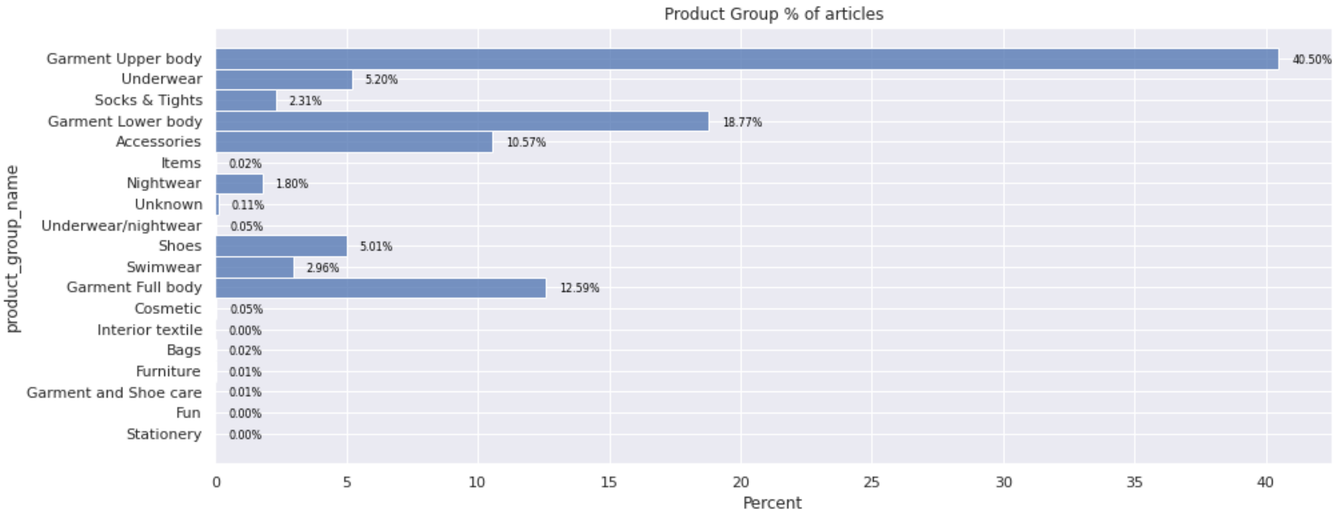


Fig. 11. Percentage of articles belonging to a specific product group.

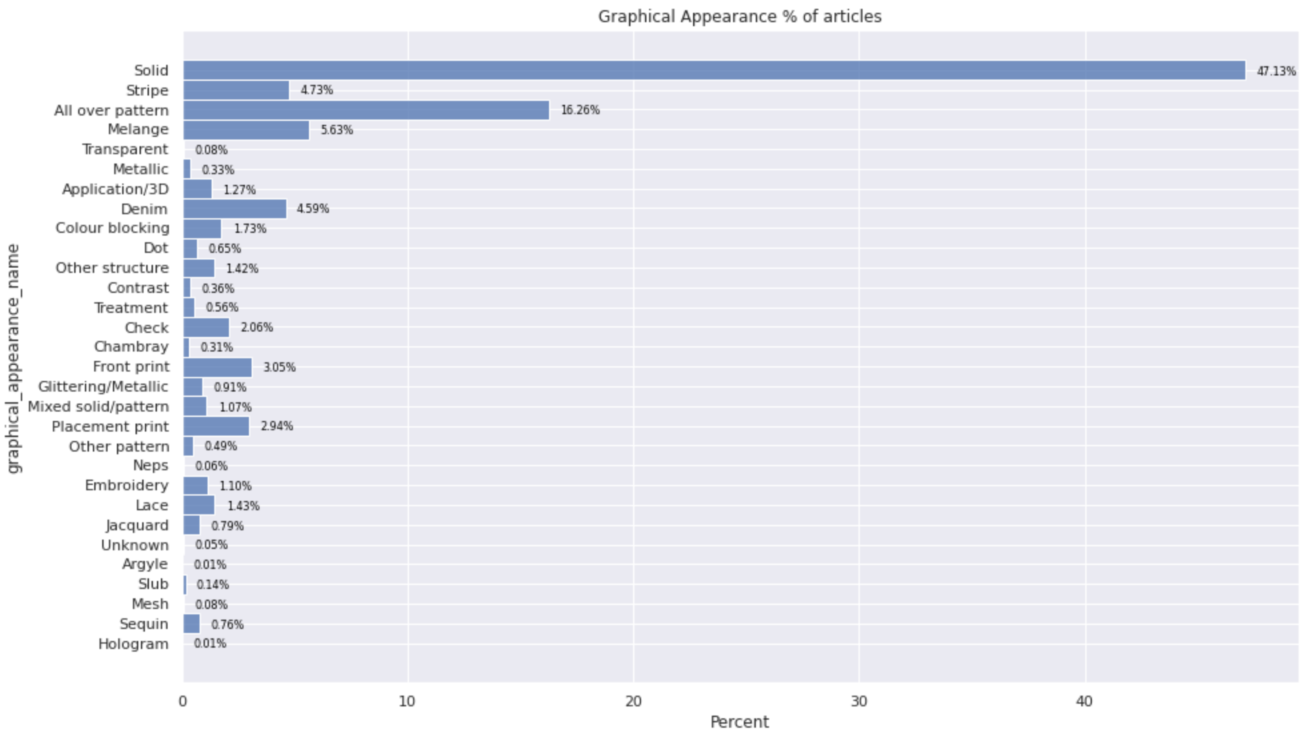


Fig. 12. Percentage of articles belonging to a specific graphical appearance.

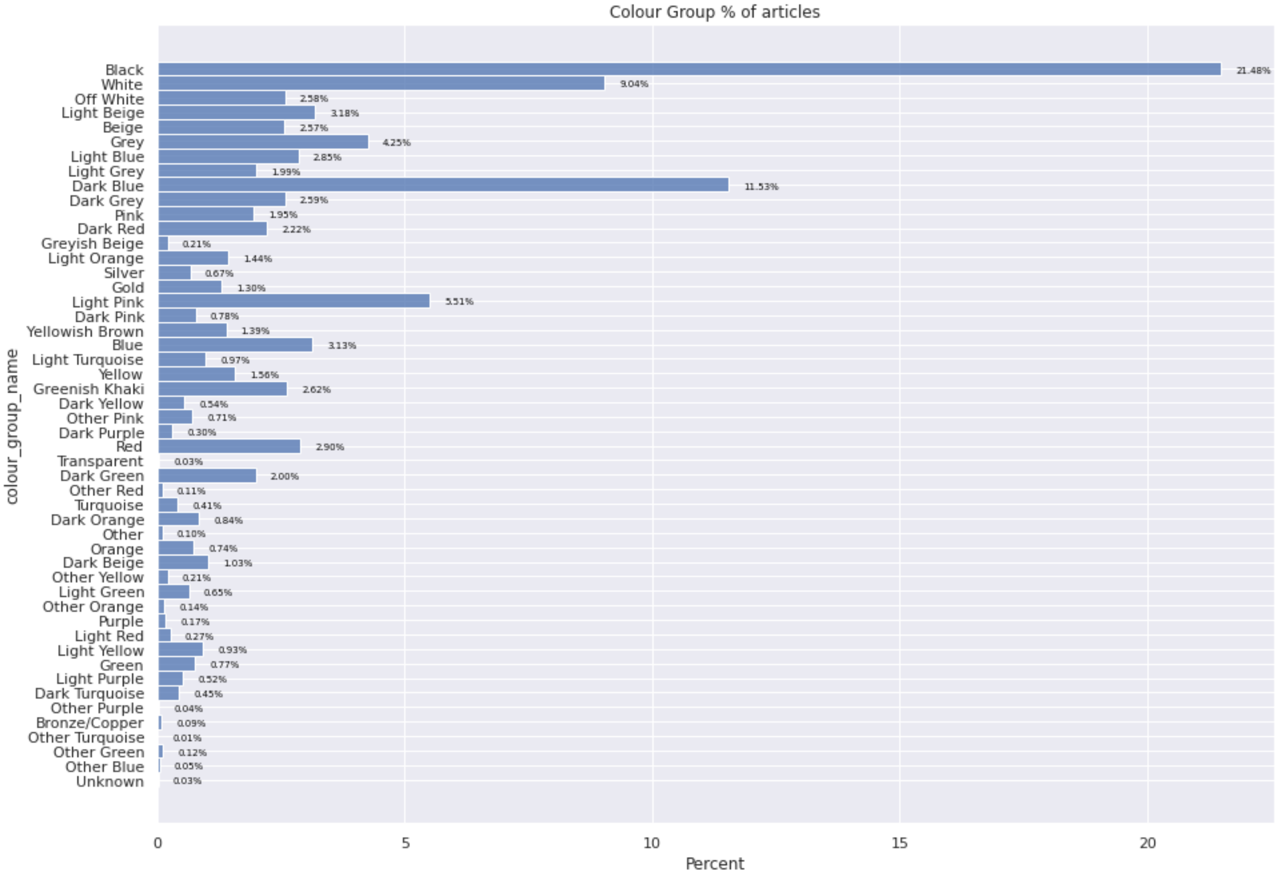


Fig. 13. Percentage of articles belonging to a specific colour group.