Available online at www.ijournalse.org

Emerging Science Journal

(ISSN: 2610-9182)

Vol. 7, No. 1, February, 2023

A Comparative Study of Collaborative Filtering in Product Recommendation

Agori Argyro Patoulia [1], Athanasios Kiourtis [1]*, Argyro Mavrogiorgou [1], Dimosthenis Kyriazis [1]

[1] *Department of Digital Systems, University of Piraeus, 18534 Piraeus, Greece.*

Abstract

Product recommendation is considered a well-known technique for bringing customers and products together. With applications in music, electronic shops, or almost any platform the user daily deals with, the recommendation system's sole scope is to help customers and attract new ones to discover new products. Through product recommendation, transaction costs can also be decreased, improving overall decision-making and quality. To perform recommendations, a recommendation system must utilize customer feedback, such as habits, interests, prior transactions as well as information used in customer profiling, and finally deliver suggestions. Hence, data is the key factor in choosing the appropriate recommendation method and drawing specific suggestions. This research investigates the data challenges of recommendation systems, specifying collaborative-based, content-based, and hybrid-based recommendations. In this context, collaborative filtering is being explored, with the Surprise library and LightFM embeddings being analysed and compared on top of foodservice transactional data. The involved algorithms' metrics are being identified and parameterized, while hyperparameters are being tuned properly on top of this transactional data, concluding that LightFM provides more efficient recommendation results following the evaluation's precision and recall outcomes. Nevertheless, even though the Surprise library outperforms, it should be used when constructing user-friendly models, requiring low code and low technicalities.

Keywords:

Recommendation Systems;

Collaborative Filtering;

Surprise Library;

LightFM;

Hyperparameters Tuning.

Article History:

Received:	16	May	2022
Revised:	14	September	2022
Accepted:	23	September	2022
Available online:	12	October	2022

1- Introduction

Recommendation Systems (RS) had a global market value of USD 2.69 billion in 2021, but it is anticipated to grow to USD 15.10 billion by 2026, with a CAGR of 37.79 % from 2022 to 2026 [1]. The recommendations that are being provided by several companies worldwide are using data analysis to identify items (e.g., products, movies) that best fit someone's preferences and needs. It is not surprising that Amazon knows which book a customer will buy or which top news story they intend to read on Twitter, given the proliferation of data on the internet. With the current developments and progress in Artificial Intelligence (AI) and the overall competition, which is growing among multiple businesses and enterprises, it is considered of crucial importance for a company to be able to search, map, and provide its consumers with the relevant data and information to enhance the consumer experience and increase digitalization. This can be efficiently achieved through using the appropriate RS. RS can reduce transaction costs of identifying and choosing products in an online shop-ping environment [2], while they can also improve decision making process and quality [3]. In the field of e-commerce, RS may enhance revenues since they can be transformed into effective means of offering and selling more products [4]. Another example includes scientific libraries, where RS can support users to move beyond

* **CONTACT**: kiourtis@unipi.gr

DOI: http://dx.doi.org/10.28991/ESJ-2023-07-01-01

simple catalogue searches. Therefore, the importance of using efficient and accurate RS techniques within a system to offer relevant and dependable recommendations for users can be considered of vital importance and significance.

In general, a RS is classified based on the information that the RS requires for the recommendations and the way that the recommendations are being produced. Two (2) basic kinds, namely (i) content-based filtering and (ii) collaborative-based filtering, exist to assist the latter. Customer information and product metadata information are two types of data utilized in content-based filtering. For instance, in this type of filtering based on a movie’s genre, if someone has previously watched a romantic movie, the RS will presume that the customer enjoys romantic comedies, and as a result, the suggestion output will include romantic comedies [5]. Making suggestions in a different approach would entail examining a customer’s transactions (i.e., interactions with offered products) and attempting to foresee the rating that the customer would assign to an unrated item. Regarding collaborative-based filtering, the latter is performed by not looking into the different categories and tags but by finding similar customers that have interacted with the same products that are of interest to the target customer.

To forecast the overall rating that the target consumer would give to every product that they have not yet engaged with, the RS uses these similar customers and their preferences as its source data. Finding products that exhibit some similarities to those that the target client has already purchased, however, represents an alternative strategy that could help in making this prediction (i.e., item-based recommendations). Hence, in comparison with the collaborative-based recommendation, the difference is that now the focus is on the products and not on the customers, to proceed with making recommendations. Hence, collaborative-based filtering is based on transactional data, being captured at the point of sale [6], dealing with information that is captured through sales transactions. In that case, it records the time of the transaction, the location where it occurred, the price points of the purchased items, the payment method, the provided discounts, as well as additional quantitative and qualitative metrics associated with the transaction. As soon as it has been decided on the methodology to be used, the construction of the RS must be initiated. However, it should be kept in mind that each RS suffers from the "Cold Start Problem" [7], a problem that arises when new products or users are being added to the system. Having new customers and products means that there is not enough information for them to perform recommendations and matches. Hence, a new product cannot be initially recommended to customers efficiently when it is introduced to the RS without any ratings or reviews, and as a result, it is hard to predict the choice or interest of users correctly and efficiently, leading to less accurate recommendations.

Having in mind that a RS’s goal is to recommend products to customers, it should be provided an efficient way for the system to understand whether a customer prefers one product or not. However, considering that most of the time there are no ratings in transactional data, such as in movies, this makes it more difficult to recommend a specific product by only using sales logs and transactions. Consequently, it is important to provide an efficient way to help the system understand whether the customer is interested in similar products to the ones that have already been purchased or not. Towards this goal, the current manuscript dives into collaborative-based filtering methods in the field of product recommendation, through analyzing the use of specific collaborative-based RS techniques and performing a comparative analysis among them. The overall goal of this research is to: (i) provide the way that a RS works; (ii) identify the method that best matches with the provided data (i.e., collaborative-based, content-based, or hybrid-based); (iii) provide the importance of a "rating" in order to identify the customer and the interactions with the product; (iv) identify the methods and algorithms that create the optimum outcome for each different case; (v) depict the way that a RS could be evaluated; and (vi) identify whether the recommendations being produced by the RS are aligned with the customer’s needs and requirements.

The manuscript is structured as follows. Section 2 provides the related work for specific methodologies for implementing collaborative-based RS, while Section 3 provides the background and the vision of the proposed comparative approach. Section 4 includes the evaluation results of the comparative analysis among the different collaborative-based RS, including a short discussion of the outcomes and future steps. Section 5 includes an overview of the current research as well as our concluding remarks.

2- Literature Reviews

Although the first thing that comes to mind when referring to RS is mostly related to movies and books, multiple RS implementations for products exist as well, since businesses and companies need to get closer to all the different types of customers. Apart from the automated versions of RSs that are being used in multiple enterprises and that are distributed by big companies [8] (e.g., Microsoft, Amazon, IBM), this manuscript follows a bottom-up approach towards the most computationally efficient RSs’ approaches. Two (2) of these approaches, which are being further evaluated, are studied in this section, namely the (i) Surprise library and (ii) LightFM.

2-1- Surprise Library

The Surprise library [9] offers a selection of estimators for rate prediction that implement traditional techniques, such as basic similarity-based algorithms and matrix factorization-based algorithms, like Singular Value Decomposition (SVD) or Non-negative Matrix Factorization (NMF). Additionally, the Surprise library offers model evaluation tools including built-in metrics and cross-validation iterators.

2-1-1- Surprise Library: Matrix Factorization-Based Algorithms - SVD Models

The SVD variant is one of the most effective RSs. It is a type of matrix factorization that minimizes the discrepancy between the anticipated ratings and the actual ratings from the original utility matrix by using gradient descent to make predictions for a user's ratings. Hence, gradient descent minimizes Root Mean Square Error (RMSE) when predicting these new ratings. Considering Figure 1 [10], it should be noticed how the rating matrix 'R' includes missing values. When predicting current ratings using the matrix factors, the Matrix Factorization algorithm employs a technique like gradient descent to reduce inaccuracy. In order to "fill in the gaps" in the rating matrix and forecast the ratings that each user would give to each item in the dataset, an algorithm like SVD is used.

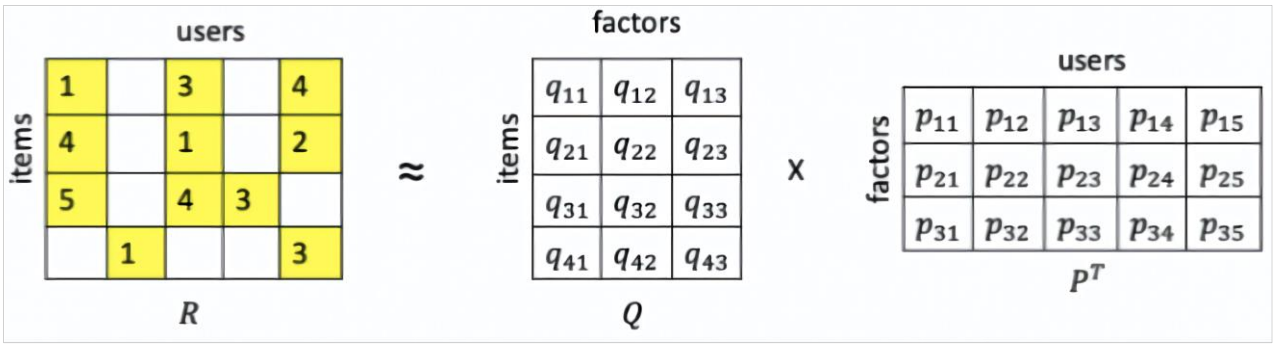

users
items
R
factors
items
Q
users
factors
P^T
≈
x
1 3 4
4 1 2
5 4 3
1 3
q_{11} q_{12} q_{13}
q_{21} q_{22} q_{23}
q_{31} q_{32} q_{33}
q_{41} q_{42} q_{43}
p_{11} p_{12} p_{13} p_{14} p_{15}
p_{21} p_{22} p_{23} p_{24} p_{25}
p_{31} p_{32} p_{33} p_{34} p_{35}

Figure 1. Matrix factorization

The SVD process is like other Machine Learning (ML) models, where the steps include the instantiation of the model and the fitting on the train set, leading to the test set prediction. Grid Search can also be incorporated to the SVD models, for further tuning. Nevertheless, there exist several parameters which should be noted, including: (i) the number of factors, (ii) the number of iterations (i.e., epochs) to run, (iii) the learning rate, and (iv) the regularization term. Considering the SVD family, Surprise library supports the following algorithms:

- *SVD*: It breaks down a matrix into its individual feature vector arrays, one for each row and column;
- *SVDpp*: It is a development of SVD that takes implicit ratings into account;
- *NMF*: It is a non-negative matrix factorization-based collaborative filtering approach.

2-1-2- Surprise Library: Similarity-Based Algorithms - K-Nearest Neighbors (KNN) Models

KNN is another often employed model. These algorithms select which recommendation to predict by looking at the nearby neighbors. Compared to SVD, KNN includes additional hyperparameters, such as the similarity measure (e.g., cosine, Pearson), and the minimum number of neighbors to consider. Considering the KNN models, Surprise library supports the following algorithms:

- *KNN Basic*: It works on top of the basic KNN algorithm;
- *KNN with Means*: It considers the mean ratings of each user;
- *KNN with Z-Score*: It considers the Z-Score normalization of each user;
- *KNN Baseline*: It considers a baseline rating.

Moreover, Surprise library has implementations of clustering models, which are mainly used in combination with different algorithms, to produce the final recommendations. The reason behind this is that even though clustering models can be straightforward, they lack in delivering efficient results. Considering the clustering models, Surprise library supports the following algorithms:

- *Slope One*: It is an implementation of the Slope One algorithm;
- *Co-clustering*: It is based on the co-clustering data mining technique.

2-2- LightFM

Python-based recommendation engine LightFM implements several well-known algorithms for both implicit and explicit feedback. In a hybrid matrix factorization approach, users and objects are represented as linear combinations of the latent factors underlying their content attributes. The method outperforms both collaborative-based and content-based models in "Cold Start" or sparse interaction data scenarios (using both user and item information) when interaction data is copious, doing at least as well as a pure collaborative matrix factorization model [11]. A LightFM model learns embeddings (i.e., latent representations in a high-dimensional space) for users and items to convey user preferences over objects. These representations are multiplied together to produce ratings for each item for a specific user, with items with higher scores generally being those that the user is more interested in Tagliabue et al. [12]. The representations of the user and item are expressed in terms of the representations of their features, with each feature's embedding being estimated separately before the features are combined to provide the representations of the user and thing [13]. The latent vector sum of user "u" determines the user's latent representation (Equation 1):

$$q_u = \sum j \in f_u \tag{1}$$

The same goes for the item "i" (Equation 2):

$$p_i = \sum j \in f_i \tag{2}$$

The dot product of the item and user representations, with item and user feature biases applied, gives the prediction of the model for a user "u" and an item "i" (Equation 3):

$$\hat{r}_{ui} = f(q_u * p_i + b_u + b_i) \tag{3}$$

It should be noted that the structure of the LightFM model is motivated by two considerations: (i) the model must learn user and item representations from interaction data, and (ii) the model must compute recommendations for new items and users.

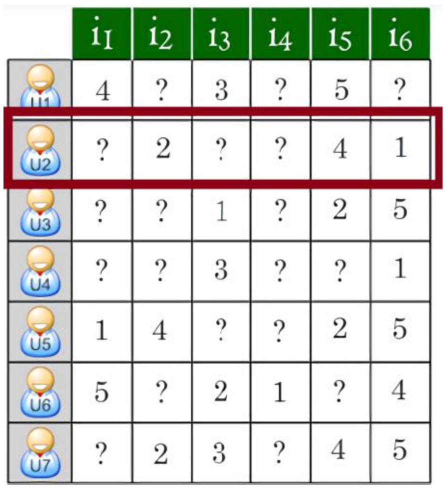
3- Proposed Approach

The approach relies on recommending foodservice products to customers. Through foodservice transactional data (January 2020 - August 2020) and taking advantage of the collaborative-based filtering methods of Section 2, the overall goal is to find the most efficient RS, as to bring customers and products together.

3-1- Overall Process

3-1-1- Recommendation Procedure

Figure 2 represents the problem that the RS must solve. Each RS has users/customers who have already bought some products, where the numbered values in the figure's cells represent the relationships between products and customers. However, a customer cannot have bought every existing product, and as a result there exist cells marked with "?".



	i_1	i_2	i_3	i_4	i_5	i_6
U1	4	?	3	?	5	?
U2	?	2	?	?	4	1
U3	?	?	1	?	2	5
U4	?	?	3	?	?	1
U5	1	4	?	?	2	5
U6	5	?	2	1	?	4
U7	?	2	3	?	4	5

Figure 2. Users and Items matrix

The RS will try to predict and fill in the "empty" cells through the collaborative-based filtering methodology of Figure 3, where five (5) different steps are included and further explored within the document.

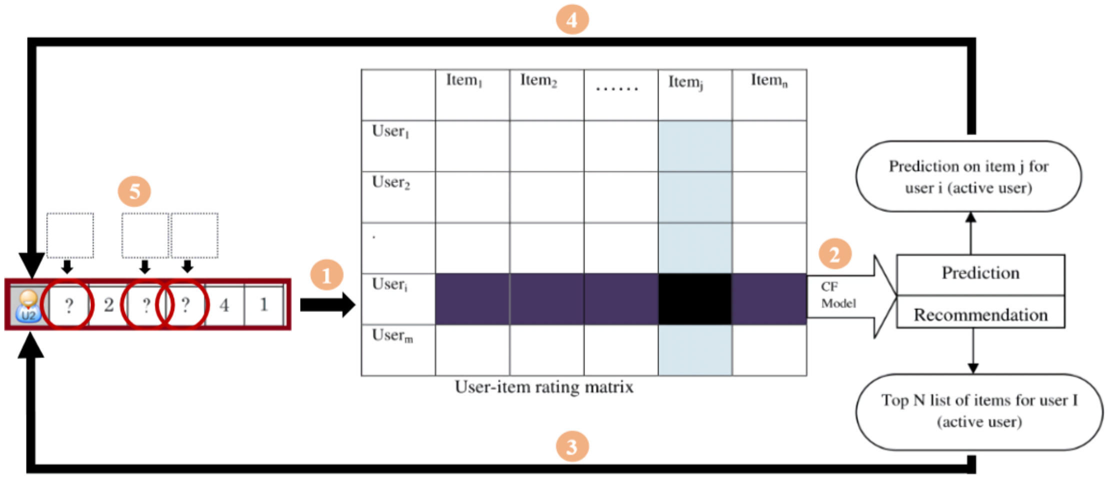


Figure 3. Collaborative filtering methodology

In more detail, every algorithm from Surprise library and LightFM's embeddings, can predict the "rating" that the customer would give to an unrated product (i.e., cells marked with '?'). Then, to get the 'Top N' recommended products for each customer, this list of predictions must be sorted in descending order and only the first 'N' products should be kept. However, apart from that, one should consider the rating metric, prior to concluding to any results.

3-1-2- The Rating Metric

An important issue to be considered deals with the rating metric. A classification or ranking of someone or something based on a comparison of their quality, standard, or performance is the best way to define rating [14]. In this case, the rating metric helps the RS to understand the degree of likeness, between a customer and a product. For example, in Figure 2 the user 'U2' has given a '2' rating to 'i2' product and a '4' rating to 'i5' product. Anyone could tell, that the user 'U2' preferred 'i5' over 'i2'. As a result, improvisations should be performed to determine the degree of likeness among the customers and products.

The graph of Figure 4 depicts the distributions of four (4) different rating metrics used by the RS, namely:

- *Quantity:* It refers to the number of units that a product can be found into an invoice

Example: Ann bought 3 sandwiches.

Quantity = 3

- *Frequency per Invoice (Frequency1) - Per customer and product:*

Example: Ann came yesterday and bought sweets, coffee, donuts. Ann came today and bought only sweets.

Frequency1(Ann, sweets) = 2/2 = 1

Frequency1(Ann, coffee) = 1/2 = 0.5

Frequency1(Ann, donuts) = 1/2 = 1

- *Frequency per Quantity (Frequency2) - Per customer and product*:

Example: Ann came yesterday and bought 3 sweets, 2 coffee, 1 donut. Ann came today and bought only 2 bags of sweets.

Frequency2(Ann, sweets) = (3 + 2)/(3 + 2 + 1 + 2) = 5/8 = 0.6

Frequency2(Ann, coffee) = 2/(3 + 2 + 1 + 2) = 2/8 = 0.25

Frequency2(Ann, donuts) = 1/(3 + 2 + 1 + 2) = 1/8 = 0.1

- *Remove Bias*: Biases are easily incorporated into the data since user interaction data is observational rather than experimental. They frequently originate from various subsets of data and create recommendation models that capture and even scale these prejudices, which results in systemic racism and unsatisfactory decisions [15]. In our situation, the distribution of training data would differ from the distribution of test data due to a number of unfavorable circumstances, such as the RS exposure mechanism or public opinion. Consequently, this metric is attempting to reduce the bias in the data, by subtracting the mean (i.e., average rating per product) from each rating score.

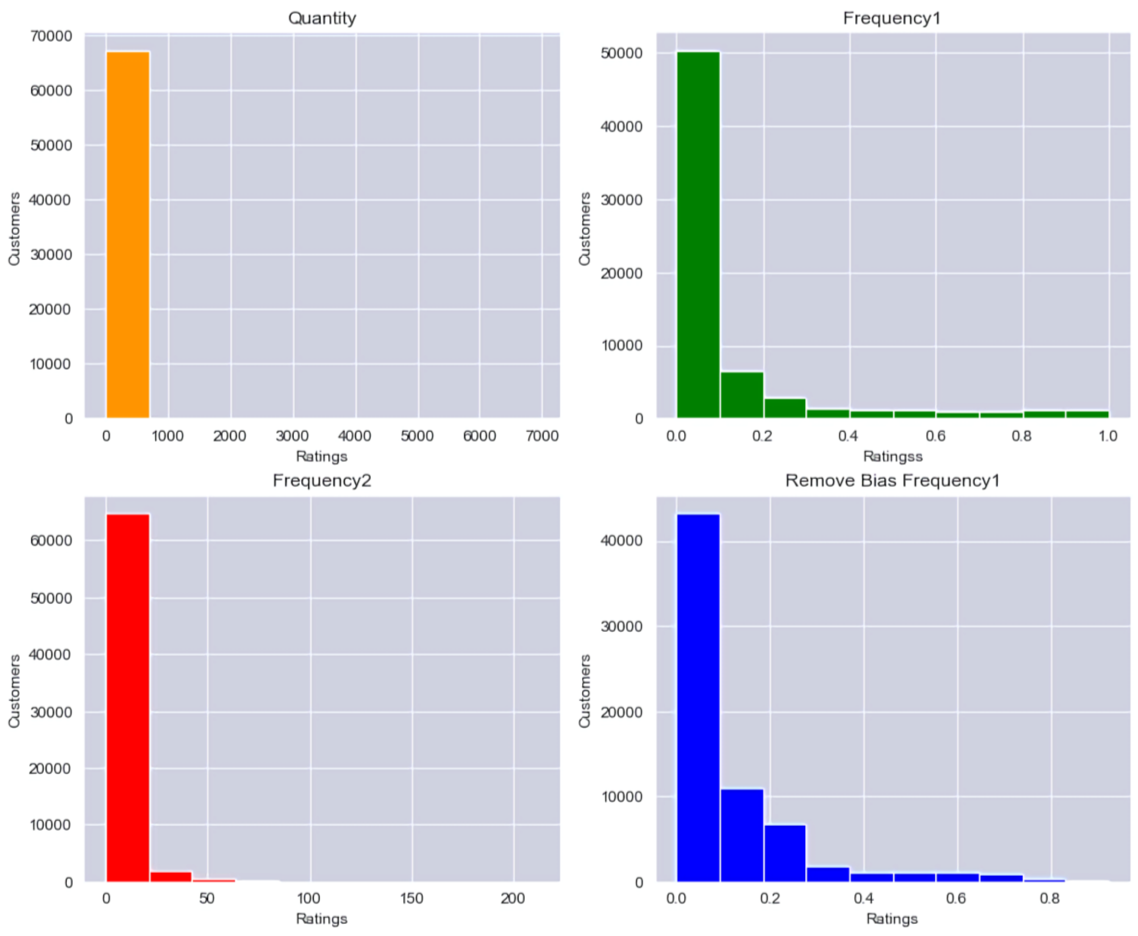


Figure 4. Rating metrics' distributions

3-1-3- Filtering

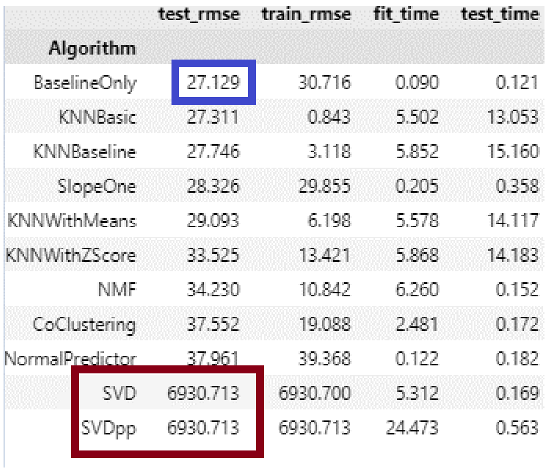
Prior to using some algorithms to make predictions, the "Cold Start Problem" should be also addressed. To avoid unfortunate situations with new customers or new products, meaning that the RS has not enough data to match new customers and new products, there must be filtered out the customers with fewer than 9 invoices, while both the train set and the test set must include the same customers and products. It should be mentioned that the period for the train set is five (5) months (January 2020 - May 2020), while the period for the test-set is two (2) months (June 2020 - July 2020). The total number of customers was 5548, while the products were 421.

3-2- Surprise Library

Several built-in algorithms from the AlgoBase base class are implemented in this library (e.g., predict, fit, test). The documentation for the prediction algorithms package contains a list and descriptions of the available prediction algorithms [9].

3-2-1- Cross Validation

There is still a chance of overfitting on the test set when comparing various settings (i.e., hyperparameters) for estimators, such as the number of latent factors - n factors – a setting that must be manually set for an SVD. This is because the parameters can be adjusted until the estimator performs at its best. In this method, the model may pick up information about the test set, and assessment metrics may no longer reflect generalization performance. Another portion of the dataset can be used as a "validation set" to overcome this problem. Then, training is carried out on the training set, followed by assessment on the validation set, and when it appears that the experiment has been successful, final evaluation can be carried out on the test set. The quantity of samples that can be used for model learning, however, is substantially decreased by splitting the available data into three (3) groups, and the outcomes may vary depending on the random selection of the train and validation sets. Cross validation (CV) is a method for resolving this issue. In that case, the validation set is no longer mandatory for doing CV, but a test set should still be stored for final evaluation. The training set is divided into "k" smaller sets in the basic technique, known as "k-fold" CV. Each of the "k" folds is created following [16]. Another practical use of CV is to have a first look at which algorithm best fits the data, before undergoing any data modifications through scaling or hyperparameter tuning. Looking into the Quantity-rating metric - CV results (Figure 5) it is apparent that the SVD algorithms could not fit the data and as a result they need Hyperparameter Tuning as to provide better results. On the other hand, the BaselineOnly algorithm managed to rank first with a RMSE score around ~28 (27,129).



Algorithm	test_rmse	train_rmse	fit_time	test_time
BaselineOnly	27.129	30.716	0.090	0.121
KNNBasic	27.311	0.843	5.502	13.053
KNNBaseline	27.746	3.118	5.852	15.160
SlopeOne	28.326	29.855	0.205	0.358
KNNWithMeans	29.093	6.198	5.578	14.117
KNNWithZScore	33.525	13.421	5.868	14.183
NMF	34.230	10.842	6.260	0.152
CoClustering	37.552	19.088	2.481	0.172
NormalPredictor	37.961	39.368	0.122	0.182
SVD	6930.713	6930.700	5.312	0.169
SVDpp	6930.713	6930.713	24.473	0.563

Figure 5. CV results for quantity

3-2-2- Prediction Evaluation - RMSE

RMSE (Equation 4) is a standard way to calculate the model's error in forecasting data of quantitative type, where '$\hat{y_i}$' are predicted values, 'y_i' are observed values, and 'n' is the number of observations.

$$RMSE = \sqrt{\sum_{i=1}^{n} \frac{(\hat{y_i} - y_i)^2}{n}} \tag{4}$$

3-2-3- Hyperparameter Tuning

Hyperparameter Tuning depicts the process of evaluating different settings (i.e., hyperparameters) for estimators. These include: (i) Grid Search and (ii) Randomized Search. In the current research, it has been performed Randomized Search for the SVD model, using the Quantity rating metric and the following parameter grid: (i) Number of factors: it deals with the latent factors, which are the characteristics of the items (e.g., music genre). By removing its latent components, the SVD reduces the utility matrix's dimension while mapping each user and each object into an r-dimensional latent space. This mapping makes it easier to represent the connections between users and items [17], (ii) Number of epochs: it includes the number of iterations of the Stochastic Gradient Descent (SGD) procedure, (iii) lr_all: it has to do with the learning rate for all the parameters, and (iv) reg_all: it represents the regularization term for all the parameters. In Figure 6, it can be seen a graphic representation of the Randomized Search, with five (5) iterations for SVD, using the above parameter grid and Quantity rating metric.

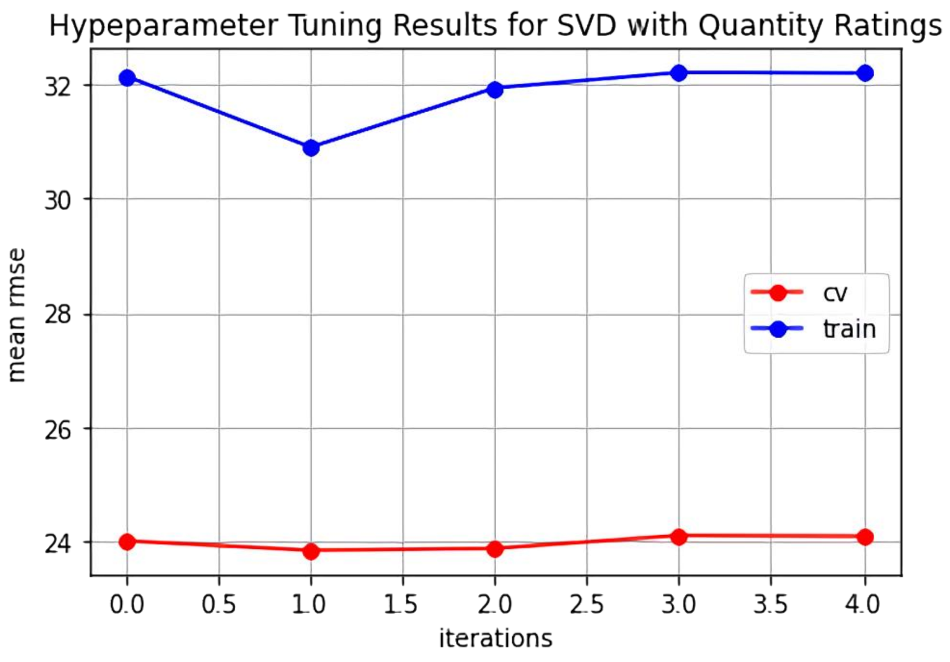


Figure 6. Hyperparameter tuning for SVD with Quantity

The parameters that provided the lowest RMSE score are the ones, given to the SVD model for the data fitting. Figure 7 depicts a visual representation of the distribution of the SVD estimations, and the training data provided to the model.

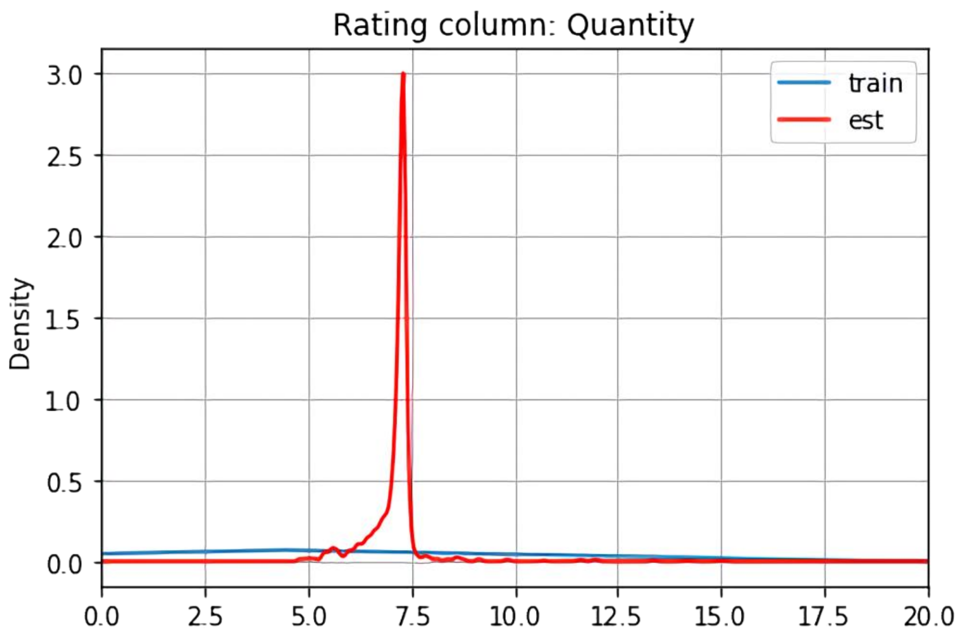


Figure 7. Estimated and train set values

Through Figure 7, most of the estimated values is accumulated around [2.5 - 5.0]. This does not create major concern, because the RS's goal is not to provide accurate predictions but to recommend products among the 'Top N' that the customer prefers and would purchase in the future. Hence, to proclaim each customer's 'Top N', it must be sorted the customer's estimated values in descending order and keep the first 'N', as for those with the highest estimated values.

3-3- LightFM

A hybrid latent representation recommendation model is offered by LightFM. The model learns embeddings (i.e., latent representations in a high-dimensional space) for users and items to convey user preferences over objects. Higher scores indicate that the item is more likely to be of interest to the user; these representations are multiplied together to generate ratings for each item for a single user. Each feature's embedding is approximated before the features are combined to create the representations for users and items. The descriptions of the people and things are expressed in terms of descriptions of their features. For instance, if the features "musical fantasy", "Judy Garland", and "Wizard of Oz" are used to describe the movie "Wizard of Oz" then the embedding of the film will be found by adding the embeddings of the individual features. The same guidelines apply to user features. The embeddings are learned using SGD methods, as indicated in Kulkarni [17].

3-3-1- Generate Numeric Identifier

LightFM only expects numeric identifiers (ids). However, the provided foodservice data contains ids for identifying products and customers. Hence, there must be created unique identifiers for each product and customer.

3-3-2- Hyperparameter Tuning

As in the previous case, before training the model, it must be performed some sort of Hyperparameter Tuning. Below is provided the parameter grid for the LightFM model tuning: (i) Number of components: it deals with the dimensionality of the latent embeddings, (ii) Learning schedule: (a) ADAptive GRADient (Adagrad): it is a class of sub-gradient methods that accomplish more informative gradient-based learning [18], and (b) ADAptive Learning Rate (Adadelta): it is a brand-new per-dimension learning rate gradient descent approach. Beyond the standard stochastic gradient descent, the method dynamically adapts over time using only first order information and has a low computing overhead [19], (iii) Loss function [20]: (a) Logistic: it is useful when both positive and negative interactions are present, (b) Bayesian Personalized Ranking (BPR) pairwise loss: it maximizes the difference in prediction be-tween a chosen negative example and a positive example. When there are solely positive interactions and Receiving Operating Characteristic (ROC) curve/Area Under Curve (AUC) optimization is sought, this can be helpful, (c) Weighted Approximate-Rank Pairwise (WARP) loss: by repeatedly sampling negative examples up until a rank-violating one is discovered, it maximizes the rank of positive examples. When there are only beneficial interactions, it is helpful, and the top of the recommendation list should be optimized, and (d) k-Order Statistic (OS) WARP: it is the 'kth' order statistic loss, dealing with a modification of WARP that uses the 'kth' positive example for any given user as a basis for pairwise updates, and (iv) Learning rate: it is the primary learning rate for the Adagrad learning schedule.

4- Evaluation

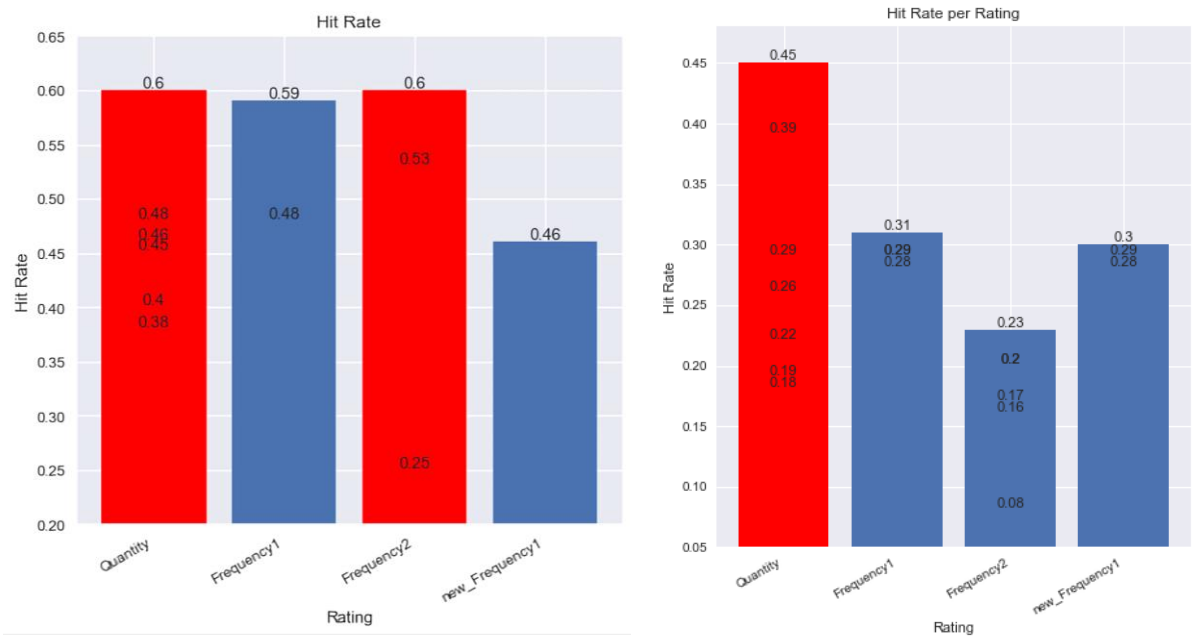
Having foodservice transactional data (January 2020 - August 2020) and taking advantage of collaborative filtering, the goal is to find the most efficient RS, as to bring together customers and new products, or products that customers have not purchased.

4-1- Hit Rate

In settings of the recommender, the Hit Rate (HR) is simply the fraction of users for which the correct answer can be found in the recommendation list of length 'L' [21], where '$|\cup_{hit}^{L}||$' is the number of users for which the correct answer is included in the top L recommendation list, and '$|\cup\, all|$' is the total number of users in the test set (Equation 5).

$$HR = \frac{|\cup_{hit}^{L}\ |}{|\cup\, all|} \tag{5}$$

As it can be seen, the larger the 'L' is, the higher the HR becomes, because the likelihood that the right response will be in the recommended list is higher. There-fore, it is important to choose a reasonable value for 'L'. The HR is the main evaluation metric and depicts how successful a model is at making recommendations. The graph of Figure 8-a depicts the HR results per rating column for LightFM, while Figure 8-b depicts the HR results per rating column for Surprise library. LightFM with the highest HR was executed through using the Quantity rating metric.



(a) (b)

Figure 8. (a) LightFM HR per rating metric, (b) Surprise library HR per rating metric

Figure 9 displays Surprise library's HR results, per rating metric and algorithm. SVD, and Quantity rating metric were the golden combinations for the Surprise library.

4-2- Recommendation Preview

A recommendation preview is provided, for a specific customer (with a unique identifier), where food purchases' recommendations are provided concerning prior food purchases and food purchases based on the recommendations.

Customer Id: B1A3085E-3617-EA11-A81C-000D3A497E15,

Customer Name: Ann Brown Recommendations: ['CAP CALDO REGULAR, 'BOTTLED WATER 0,5L', 'ESPRESSO FREDDO BRAZILIAN', 'ESPRESSO FREDDO ARABICA', 'CAP CALDO REGULAR ARABICA'

Will buy: ['CINAMON ROLL', 'BOTTLED WATER 0,5L', 'ESPRESSO FREDDO ARABICA', 'FREDDO REGULAR ARABICA', 'ESPRESSO FREDDO BRAZILIAN', 'ΚΟΥΛΟΥΡΙ ΘΕΣΣΑΛΟΝΙΚΗΣ', 'FREDDO REGULAR BRAZILIAN'] Hits: 3 ['ESPRESSO FREDDO BRAZILIAN', 'ESPRESSO FREDDO ARABICA', 'BOTTLED WATER 0,5L']

Prior Purchases: ['TOAST', 'SANDWICH', 'PIE', 'BREAD']

Prior Purchases in Recommendations: (0, []) New Products: 6 ['ESPRESSO FREDDO BRAZILIAN', 'ESPRESSO FREDDO ARABICA', 'CINAMON ROLL', 'FREDDO REGULAR BRAZILIAN', 'FREDDO REGULAR ARABICA', 'BOTTLED WATER 0,5L'].

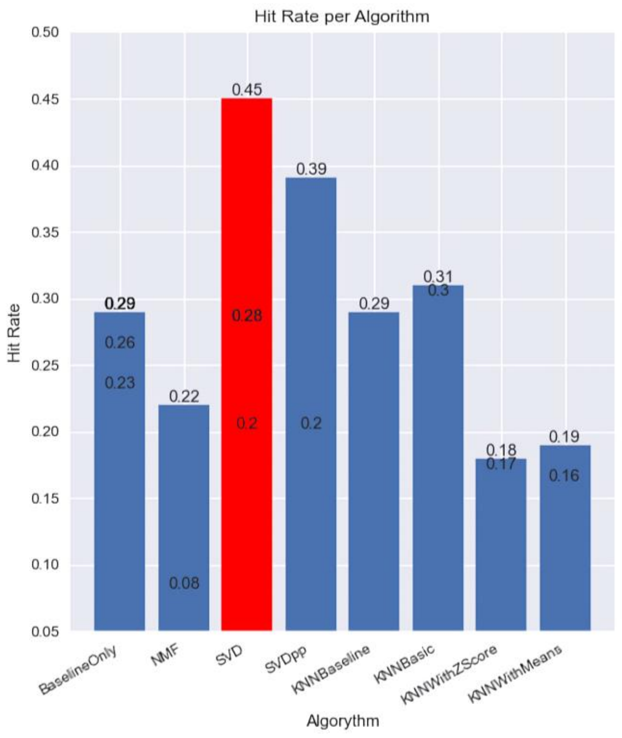


Figure 9. Surprise library HR per algorithm

4-3- Long Tail

The Long Tail (Figure 10) is being utilized to examine popularity trends in click-, rating-, and purchase-related user-item interaction data. Only a small portion of things typically have a high number of interactions; this is known as head. The majority of products are in the Long Tail, although interactions with them are rather rare [22].

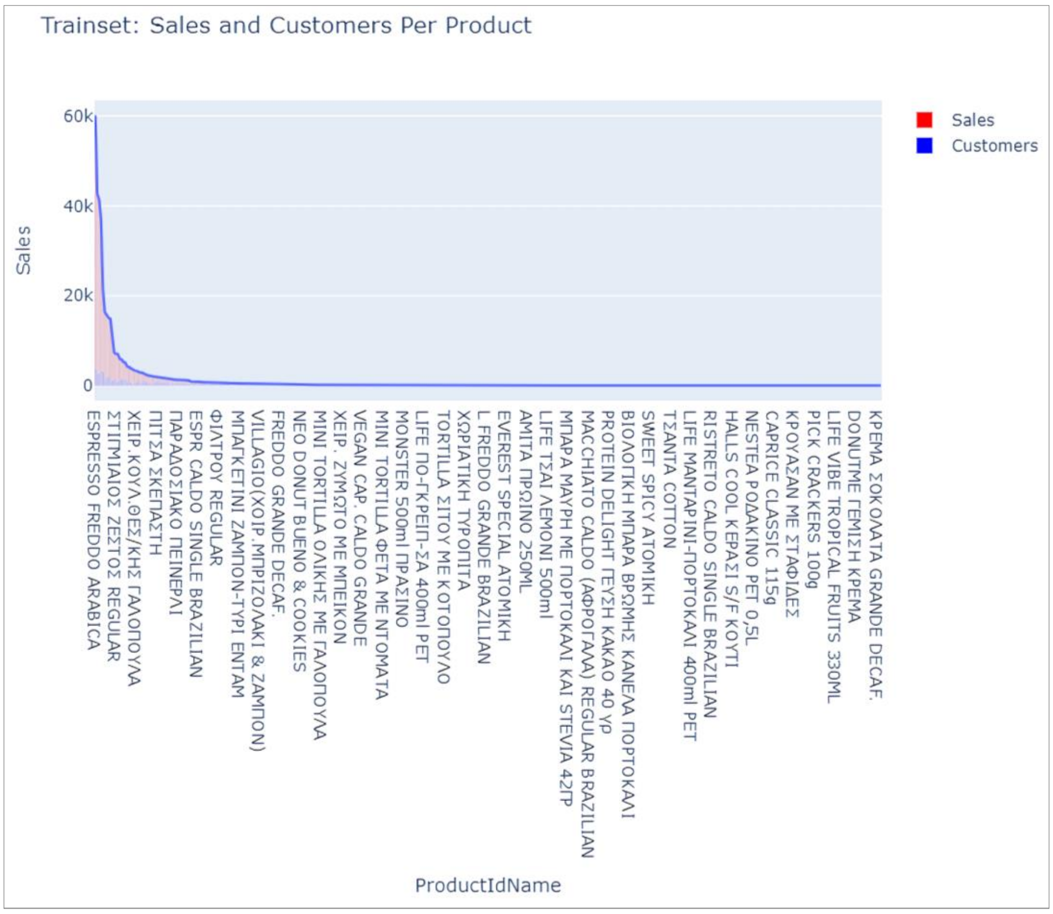


Figure 10. Long Tail (sales per product)

It is not difficult for an RS to learn to correctly predict these items because there are many observations of popular items in the train data. Most users are already familiar with these products, thus recommending them may not provide a personalized experience or aid consumers in finding fresh, pertinent merchandise. Relevant recommendations are those of goods the user has given high marks for in the test data.

4-4- Comparison based on Precision & Recall

Prior to referring to the Precision & Recall metrics of the results, there should be identified the following attributes:

- *True Positive (TP)*: On TP, the algorithm predicts that the user is going to purchase the product (recommendation) and the user purchases the product.
- *False Positive (FP)*: On FP, the algorithm recommends the product, but the user does not purchase it.
- *False Negative (FN)*: On FN, the algorithm does not recommend the product, but the user purchases it.
- *True Negative (TN)*: On TN, the algorithm does not recommend the product, and the user does not purchase the product.

Concerning the Precision metric, this is defined as the percentage of predictions per the recommendations that are provided correctly. On the other hand, the Recall metric refers to the portion of relevant products in the recommendations (i.e., Relevant recommendations are those for products that the user has given a favourable rating in the test data). Through the graphs of Figures 11 and 12, LightFM reached higher Precision and Recall results. This results into a variety of products being recommended and more personalized recommendations for the customers. Surprise library on the other hand, suffer from its model limitations and had relatively low Precision and Recall results. This affects the final recommendations, as to use only a small portion of the available products and to have repetitiveness of the 'Top N' set, resulted into not efficient personalized recommendations for the customers.

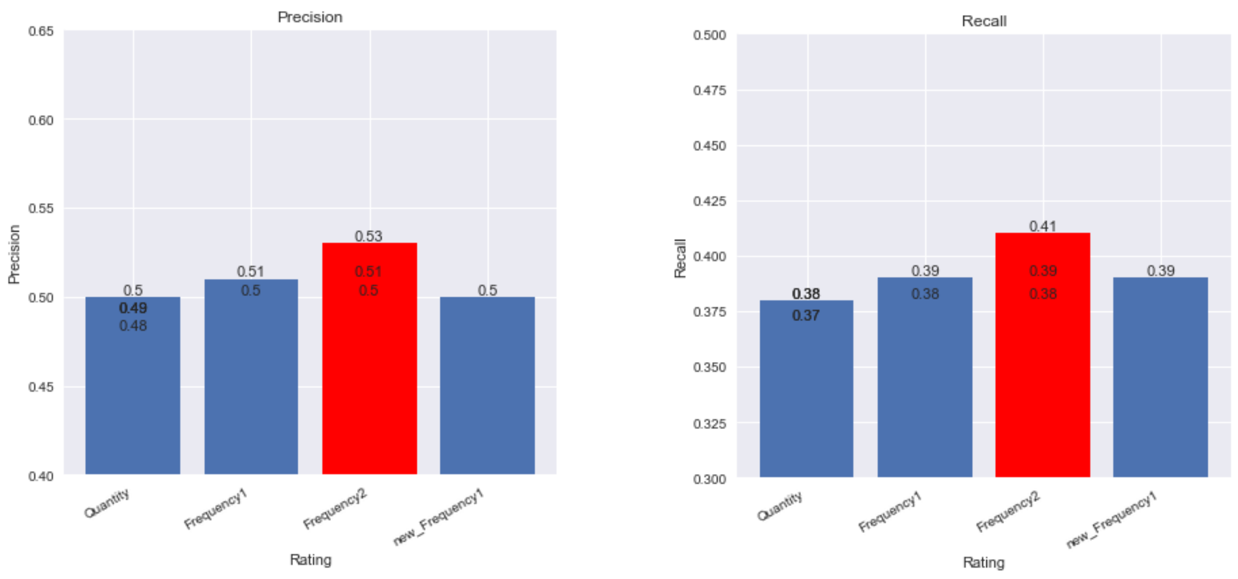


Figure 11. LightFM Precision and Recall

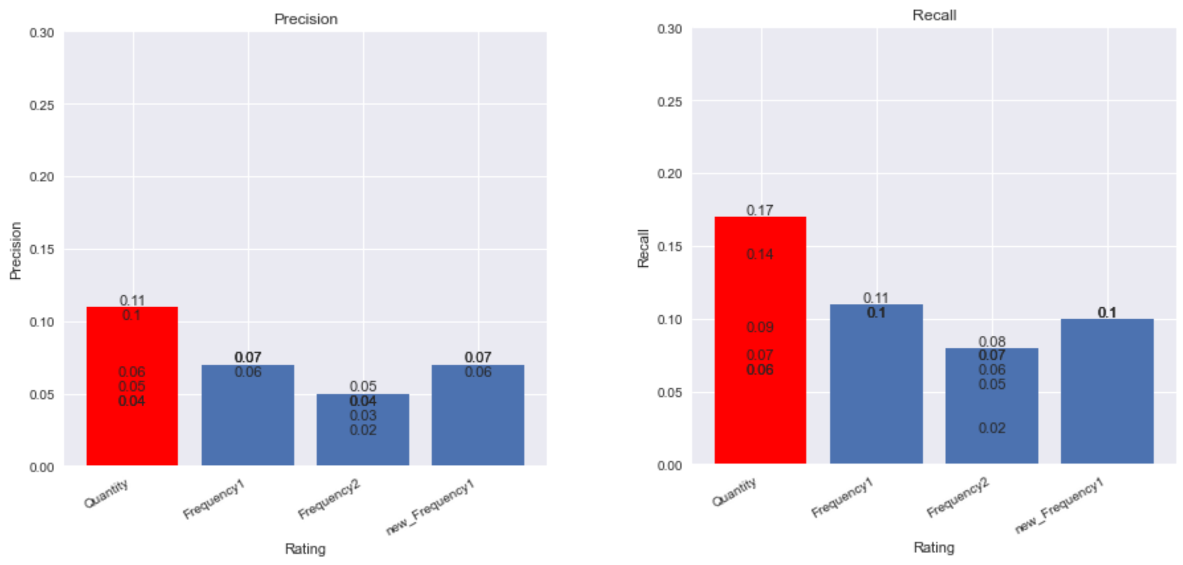
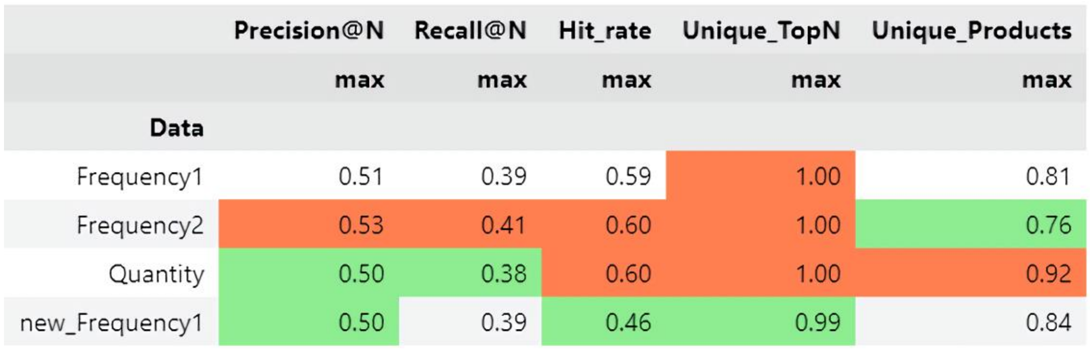


Figure 12. Surprise library Precision and Recall

4-5- Discussion of Results

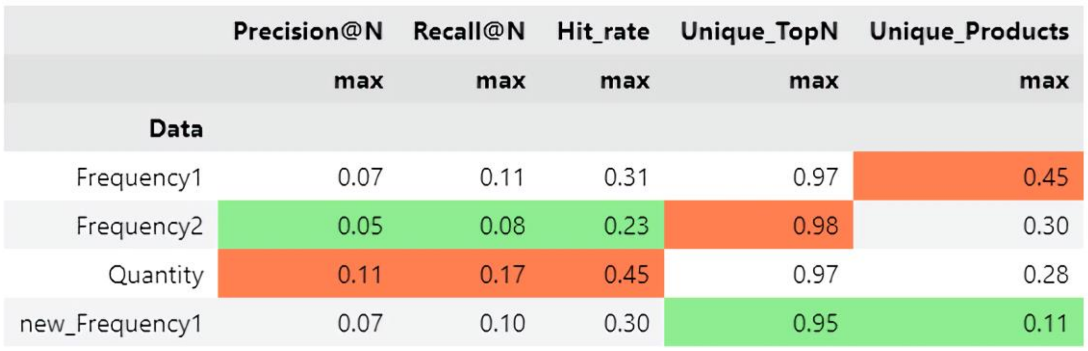
Based on the results, the best model to be used in this case is LightFM. “Less is better” when it comes to LightFM model (Figure 13). Both highest HR scores were made using loss function as warp and 30 epochs. Regardless of data type, the Frequency2, or the Quantity, the best solution is the simplicity for this scenario.



	Precision@N	Recall@N	Hit_rate	Unique_TopN	Unique_Products
	max	max	max	max	max
Data					
Frequency1	0.51	0.39	0.59	1.00	0.81
Frequency2	0.53	0.41	0.60	1.00	0.76
Quantity	0.50	0.38	0.60	1.00	0.92
new_Frequency1	0.50	0.39	0.46	0.99	0.84

Figure 13. **LightFM best evaluation results**

For Surprise library (Figure 14), SVD combined with the Quantity data, outperformed by far the other models, with a HR of 0.45. In any case, Surprise library’s best run did not reach LightFM’s 0.6 HR score. Moreover, Precision (0.11) and Recall (0.17) did not provide promising results for the variety in the provided recommendations.



	Precision@N	Recall@N	Hit_rate	Unique_TopN	Unique_Products
	max	max	max	max	max
Data					
Frequency1	0.07	0.11	0.31	0.97	0.45
Frequency2	0.05	0.08	0.23	0.98	0.30
Quantity	0.11	0.17	0.45	0.97	0.28
new_Frequency1	0.07	0.10	0.30	0.95	0.11

Figure 14. **Surprise library best evaluation results**

A results’ overview of the LightFM and Surprise library best runs is displayed in Figure 15, through a radar plot.

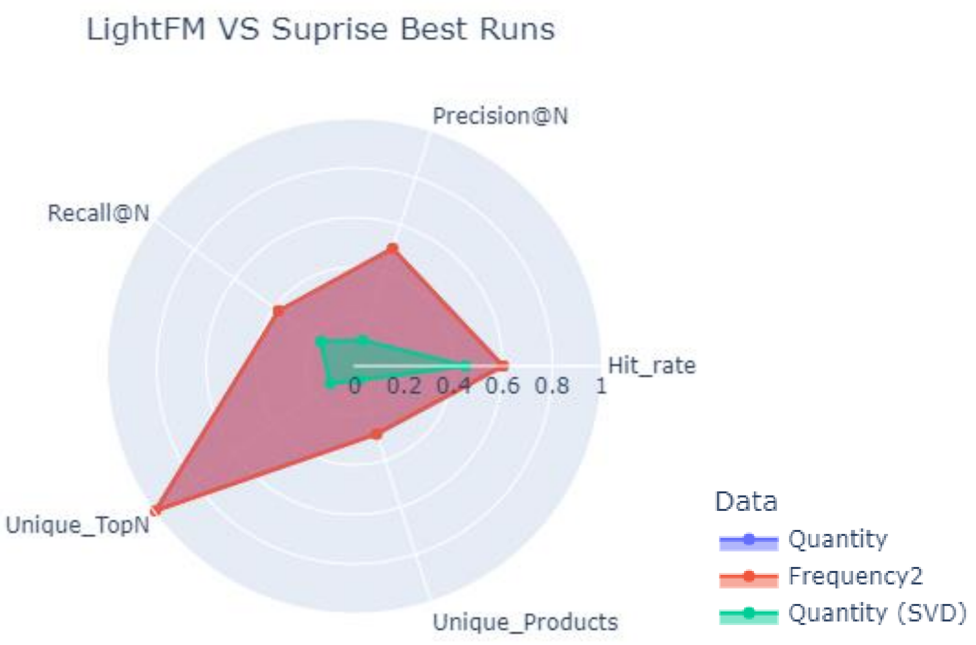


Figure 15. **Radar Plot of LightFM in comparison with Surprise library**

Based on the derived results, our next steps include additional evaluations among different libraries, algorithms, and embeddings, with respect to the Surprise library and LightFM, considering more complex RS, with more complicated recommendations, training, testing, and validation tests. Furthermore, it is within our next goals to perform additional experiments in diverse areas and domains, covering not only the product sales domain but also cross-sector domains combining knowledge and requirements from the entertainment and product sales sectors (e.g., recommendations of products based on movie preferences). To this end, we focus on extending the Hyperparameters Tuning methodology, to be performed more efficiently following our previous research in Lytra et al. [23], into a more domain-specific way, where hyperparameters would be finetuned without any user input, considering past domain-specific tunings performed for other scenarios and use cases, addressing data preprocessing [24] and interoperability [25] challenges. To this end, it is within our next plans to also consider additional evaluation metrics within upcoming RS analyses, such as: (i) Unique Top N Recommendations: It deals with the percent of unique Top N-sets, recommended per customer, (ii) Unique Products in Top N: It has to do with the percent of the available products, that got included in all the Top N – sets, or (iii) Hit Products: It deals with the products that got a 'Hit' (i.e., the product made it to the Top N and the customer bought it.) and the number of 'Hits' each product made. Above all, our further goals include additionally evaluating the derived RS methods deployed in the infrastructure of the Diastema project [26], exploiting high-end hardware specifications (e.g., Solid State Drives (SSD) compared to Hard Disk Drives (HDD), DDR4 memories instead of DDR3 with higher capacities and more efficient CPUs), and integrating the derived RS techniques within the microservices platform of beHEALTHIER [27].

5- Conclusion

It is undeniable that RSs have become an essential feature in the current digital world. Since it is almost impossible to discover all the products or content on a website, a RS can have an important role in improving the overall user experience by exposing additional products and inventories with such characteristics. In this research, it was provided an efficient way to facilitate a RS to understand whether the customer is interested in similar products to the ones that have already been purchased or not, through specific collaborative-based filtering methods in the field of product recommendation. The Surprise library and LightFM were introduced, while a comparative analysis was performed between them. The derived results from executing the Surprise library's algorithms and LightFM's embedding model show that embedding implementations (i.e., LightFM) surpass basic or more mainstream techniques (e.g., KNN-based approaches, matrix factorization) that the Surprise library uses. Indeed, the embedding method is based on matrix factorization calculations, but LightFM is a relatively new library that uses SGD as its core. LightFM has also different learning rates techniques (Adagrad, Adadelta) and loss functions like WARP, terms being used in Neural Networks, which can be best described as more complicated and sophisticated models from SVD, SVDpp, and NMF. The Surprise library includes a variety of models but also suffers from these models' limitations. For that reason, LightFM has established its position as a well-known and widely used RS implementation library, also supporting hybrid-based collaborative filtering techniques through enabling processing metadata for users and items as well as implicit feedback. For the Surprise library, this result in comparison with LightFM was apparent, but it is still a popular library among the ML cycles as it is widely used to facilitate the construction and evaluation of low-technicality and complexity models. As a result, both LightFM and Surprise libraries should be used, targeting different target groups depending on the complexity and the details of the RS that should be specified and implemented.

6- Declarations

6-1- Author Contributions

Conceptualization, A.A.P. and A.K.; methodology, A.M.; software, A.A.P.; validation, A.A.P., A.K. and A.M.; formal analysis, A.A.P.; investigation, A.A.P.; resources, A.K.; data curation, A.M.; writing-original draft preparation, A.A.P.; writing-review and editing, A.K.; visualization, A.M.; supervision, D.K.; project administration, D.K.; funding acquisition, D.K. All authors have read and agreed to the published version of the manuscript.

6-2- Data Availability Statement

Data sharing is not applicable to this article.

6-3- Funding

The research has been co-financed by the European Union's Horizon 2020 research and innovation programme under grant agreement No 826106 (InteropEHRate project) and by the European Union and Greek national funds through the Operational Program Competitiveness, Entrepreneurship and Innovation, under the call RESEARCH – CREATE – INNOVATE (project codes: BeHEALTHIER - T2EDK-04207, DIASTEMA - T2EDK-04612).

6-4- Institutional Review Board Statement

Not applicable.

6-5- Informed Consent Statement

Not applicable.

6-6- Conflicts of Interest

The authors declare that there is no conflict of interest regarding the publication of this manuscript. In addition, the ethical issues, including plagiarism, informed consent, misconduct, data fabrication and/or falsification, double publication and/or submission, and redundancies have been completely observed by the authors.

7- References

[1] Dhawad, A., Sarkar, S., Agarwal, A., & Darlapudi, R. K. (2021). Recommendation Engines: Traditional vs. Deep Learning Approaches. 2021 6th International Conference on Computing, Communication and Security (ICCCS), Las Vegas, NV, USA, IEEE. doi:10.1109/icccs51487.2021.9776342.

[2] Afolabi, I. T., Makinde, O. S., & Oladipupo, O. O. (2019). Semantic Web mining for Content-Based Online Shopping Recommender Systems. International Journal of Intelligent Information Technologies, 15(4), 41–56. doi:10.4018/IJIIT.2019100103.

[3] Pantano, E., Priporas, C. V., Stylos, N., & Dennis, C. (2019). Facilitating tourists' decision making through open data analyses: A novel recommender system. Tourism Management Perspectives, 31, 323–331. doi:10.1016/j.tmp.2019.06.003.

[4] Mehta, Y., Singhania, A., Tyagi, A., Shrivastava, P., Mali, M. (2020). A Comparative Study of Recommender Systems. ICDSMLA 2019, Lecture Notes in Electrical Engineering, 601. Springer, Singapore. doi:10.1007/978-981-15-1420-3_112.

[5] Sharma, J., Sharma, K., Garg, K., & Sharma, A. K. (2021). Product recommendation system a comprehensive review. IOP Conference Series: Materials Science and Engineering, 1022(1), 12021. doi:10.1088/1757-899X/1022/1/012021.

[6] Lebanoff, L., Peterson, C., Dechev, D. (2019). Check-Wait-Pounce: Increasing Transactional Data Structure Throughput by Delaying Transactions. Distributed Applications and Interoperable Systems, DAIS 2019, Lecture Notes in Computer Science, 11534. Springer, Cham, Switzerland. doi:10.1007/978-3-030-22496-7_2.

[7] Natarajan, S., Vairavasundaram, S., Natarajan, S., & Gandomi, A. H. (2020). Resolving data sparsity and cold start problem in collaborative filtering recommender system using Linked Open Data. Expert Systems with Applications, 149, 113248. doi:10.1016/j.eswa.2020.113248.

[8] Sadman, N., Gupta, K. D., Haque, A., Poudyal, S., & Sen, S. (2020). Detect Review Manipulation by Leveraging Reviewer Historical Stylometrics in Amazon, Yelp, Facebook and Google Reviews. Proceedings of the 2020 The 6th International Conference on E-Business and Applications. doi:10.1145/3387263.3387272.

[9] Hug, N. (2020). Surprise: A Python library for recommender systems. Journal of Open Source Software, 5(52), 2174. doi:10.21105/joss.02174.

[10] Cui, Z., Zhao, P., Hu, Z., Cai, X., Zhang, W., & Chen, J. (2021). An improved matrix factorization based model for many-objective optimization recommendation. Information Sciences, 579, 1–14. doi:10.1016/j.ins.2021.07.077.

[11] Shriver, D. (2018). Toward the development of richer properties for recommender systems. Proceedings of the 40th International Conference on Software Engineering: Companion Proceeedings. doi:10.1145/3183440.3195082.

[12] Tagliabue, J., Yu, B., & Bianchi, F. (2020). The Embeddings That Came in From the Cold: Improving Vectors for New and Rare Products with Content-Based Inference. RecSys '20: Proceedings of the 14th ACM Conference on Recommender Systems. doi:10.1145/3383313.3411477.

[13] Wu, L., Quan, C., Li, C., Wang, Q., Zheng, B., & Luo, X. (2019). A context-aware user-item representation learning for item recommendation. ACM Transactions on Information Systems, 37(2), 1–29. doi:10.1145/3298988.

[14] Zhang, F., Mao, J., Liu, Y., Xie, X., Ma, W., Zhang, M., & Ma, S. (2020). Models versus Satisfaction. Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval. doi:10.1145/3397271.3401162.

[15] Chen, J., Dong, H., Wang, X., Feng, F., Wang, M., & He, X. (2020). Bias and debias in recommender system: A survey and future directions. arXiv preprint arXiv:2010.03240. doi:10.48550/arXiv.2010.03240.

[16] Bisong, E. (2019). More Supervised Machine Learning Techniques with Scikit-learn. In: Building Machine Learning and Deep Learning Models on Google Cloud Platform, Apress, Berkeley, United States. doi:10.1007/978-1-4842-4470-8_24.

[17] Kulkarni, N. (2020). Customer Behaviour Prediction. PhD Thesis, National College of Ireland, Dublin, Republic of Ireland.

[18] Zhang, M., Hao, B., Ge, Q., Zhu, J., Zheng, R., & Wu, Q. (2022). Distributed Adaptive Subgradient Algorithms for Online Learning Over Time-Varying Networks. IEEE Transactions on Systems, Man, and Cybernetics: Systems, 52(7), 4518–4529. doi:10.1109/TSMC.2021.3097714.

[19] Kokilambal, S. (2021). Intelligent Content Based Image Retrieval Model Using Adadelta Optimized Residual Network. International Conference on System, Computation, Automation and Networking (ICSCAN), Puducherry, India. doi:10.1109/icscan53069.2021.9526470.

[20] Jiang, G., Wang, H., Chen, J., Wang, H., Lian, D., & Chen, E. (2021). xLightFM: Extremely Memory-Efficient Factorization Machine. Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval. doi:10.1145/3404835.3462941.

[21] Vaz, C. M. P., de Resende, J. M., Franchini, J. C., Debiasi, H., & Nunes, M. R. (2022). Evaluation and recommendations for the use of dynamic penetrometers. Soil and Tillage Research, 220, 105373. doi:10.1016/j.still.2022.105373.

[22] Wei, Y., Wang, X., Li, Q., Nie, L., Li, Y., Li, X., & Chua, T.-S. (2021). Contrastive Learning for Cold-Start Recommendation. Proceedings of the 29th ACM International Conference on Multimedia. doi:10.1145/3474085.3475665.

[23] Lytra, G., Mavrogiorgou, A., Kiourtis, A., & Kyriazis, D. (2021), Hyperparameter Optimization on Classification and Regression Algorithms. IOSR Journal of Computer Engineering (IOSR-JCE), 23(4), 34-50. doi:10.9790/0661-2304013450.

[24] Mavrogiorgou, A., Kiourtis, A., Manias, G., & Kyriazis, D. (2021). Adjustable data cleaning towards extracting statistical information. Public Health and Informatics, Studies in Health Technology and Informatics, 1013–1014, IOS Press, Amsterdam, Netherlands. doi:10.3233/SHTI210332.

[25] Kiourtis, A., Mavrogiorgou, A., Kyriazis, D., Maglogiannis, I., & Themistocleous, M. (2016). Towards Data Interoperability: Turning Domain Specific Knowledge to Agnostic across the Data Lifecycle. 2016 30th International Conference on Advanced Information Networking and Applications Workshops (WAINA). doi:10.1109/waina.2016.69.

[26] Kiourtis, A., Karamolegkos, P., Karabetian, A., Voulgaris, K., Poulakis, Y., Mavrogiorgou, A., & Kyriazis, D. (2022). An Autoscaling Platform Supporting Graph Data Modelling Big Data Analytics. Studies in Health Technology and Informatics, 295, 376–379. doi:10.3233/SHTI220743.

[27] Mavrogiorgou, A., Kleftakis, S., Mavrogiorgos, K., Zafeiropoulos, N., Menychtas, A., Kiourtis, A., Maglogiannis, I., & Kyriazis, D. (2021). beHEALTHIER: A Microservices Platform for Analyzing and Exploiting Healthcare Data. 2021 IEEE 34th International Symposium on Computer-Based Medical Systems (CBMS). doi:10.1109/cbms52027.2021.00078.