ZE GERMANS LIMITED

ManyGiftsUK

Group G - Authors:
Li-lou Dang-Thai, m20200743
Maximilian Maukner, m20200645
Steffen Hillmann, m20200589
Ehsan Meisami Fard, m20201050

Presentation
Github Repository

Table of Contents

1. Introduction

Among many growing business models during the last decade, the growth of e-commerce has been quite astonishing and eye-catching. A benefit of digital products and services in most cases is the elimination of the requirement of storage and space for physical products. As a result, e-commerce companies have been able to increase the variety of product offerings for their customers [1]. However, infinite choice does not necessarily result in a superior customer experience. Iyengar and Lepper [2] starkly challenge the notion that having more choices is necessarily more intrinsically motivating than having fewer. Moreover, participants in their study reported greater satisfaction when facing fewer options to select from. Although search engine technology has been helpful for customers to find their desired products immediately, by being able to accumulate data on a user level, companies have the opportunity to improve customer experience through personalized recommendations by creating customer profiles, monitoring user behavior and preferences [3]. Organizations have been designing surveys and assessment systems to receive explicit feedback from users regarding a specific service or product. For instance, Netflix and other streaming platforms gather star ratings for movies from their users which determines their preferences. However, in an imperfect world, explicit feedback is not always obtainable. As an example, a Netflix user might be reluctant to rate movies after watching them. An alternative solution in regards to a user's unwillingness to rate products and services is to gather intelligence concerning user preferences through implicit feedback [4]. For instance, a user watching drama movies several times could indicate their preference for that specific genre.

The all-occasion gift retailer, ManyGiftsUK, launched its website and turned its business entirely to the digital world. As a consequence, the online retailer has been able to collect massive amounts of data about its customers and expects to build a recommender system that improves its customer experience by offering products based on user's preferences and purchasing habits. The present project conducts an investigation into a few algorithms for processing implicit feedback.

Firstly, according to the structure of a CRISP Methodology, the business objectives of this project are illustrated followed by the translation of these objectives into technical objectives. Secondly, after an overview of the given dataset, the required steps for data preparation as the groundwork for further analyses are explained. Thirdly, an extensive exploratory data analysis is shown which depicts the initial underlying information and insights. Moreover, different algorithms and techniques for building an implicit recommender system have been implemented and discussed. Furthermore, different solutions for the cold start problem have been examined. Lastly, the deployment phase deals with elaborating on the challenges and recommendations in implementing the recommender system.

2. Business Understanding

2.1. Business Objective

Over the last two years, ManyGiftsUK has digitized its business and moved it to the web. Products are now marketed exclusively online, either through their website or via Amazon. This step allowed ManyGiftsUK to accumulate a large amount of data about its customers. By using the collected data, the management wants to explore its business to understand its customers. However, ManyGiftsUK's main objective is to build and implement a recommender system that facilitates user choices by recommending items depending on their preferences and to improve the user experience when making purchases on their website. Furthermore, the company would like to be able to assist new customers with relevant item suggestions.

2.2. Situation Assessment

The provided dataset represents customer transactions occurring between 01/12/2010 and 09/12/2011. It counts 25.900 valid transactions of 4372 customers from 38 different countries. Despite useful information about the customer transactions, such as the quantity, the unit price, and the invoice date to name a few, the data lacks socio-demographic information regarding the customer which could hamper the resolution of the cold start problem.

2.3. Data Mining Goals

The above-mentioned business objectives can be translated into rather more technical data mining objectives. Thus, the main objective is to use the clients' past interactions (such as purchases, refunds, and cancelations) to create an effective recommendation system model which will suggest relevant items to each customer, and which will be efficient with new users as well. Moreover, to evaluate our performance, the AUC score will be used on the different models. The data mining success criteria would be to have a system that yields better results than not implementing a system. Besides, another goal is to deploy an app on Heroku using Flask for our recommender system, which will display the suggested items to the users.

3. Data Mining Process

3.1. Data Understanding

The investigated dataset consists of 8 features - 5 are categorical and 3 are numerical - and 541.909 records, each of them stands for a particular item contained in a transaction. In total 25.900 valid transactions were observed, associated with 4070 unique items and 4372 customers from 38 different countries. Moreover, it is important to note that many of ManyGiftsUK's customers are wholesalers.

3.2. Data Preparation

Data preparation consists of selecting features, cleaning the data, and eventually implementing feature engineering to discover new information.

First, the missing values were examined. Only two columns had missing values, namely the *Description* (1,454) and the *CustomerID* (135,080) where all the missing values detected in the *Description* column were also entailed in the *CustomerID.* That is, the dataset contains in total 135,080 rows with Nan values which accounts for almost 25% of the total. At first, these records were not deleted but given a fictitious value so that the information is not lost and to work with it in the further course of this project. However, an extensive analysis showed that new CustomerIDs are not representative, so these were finally deleted. An optimization potential would be to classify this data so that it is not only split under one CustomerID but on several different CustomerIDs. The reason for the high number is that it represents customers and their purchases that were saved before they migrated to the digitized webshop.

To handle the missing values we first looked at the entries containing a unit price equal to zero. Since these (in total 1,454) had no entry for *Description* we decided to delete them for the time being, as they do not generate any necessary information. Further, the *StockCode* variable indicated that some entries were samples.

Moreover, 8,506 observations which represent 2.3% of the complete data, have a negative *Uniprice* or *Quantity*. That is, two observations were found with a *Unitprice* lower than zero. By looking at the description it showed that the entry corresponded to "adjust bad debt", and was therefore deleted from the data frame.

Following the cleaning of the missing values and the negative entries from *Unitprice* and *Quantity*, duplicates were investigated. A total of 5,263 duplicated entries were found and deleted.

Furthermore, different descriptions have been assigned for certain types of *Stockcodes*. Therefore, a new column namely *SC_clean* was created which stores a unique value for each description. This information is required for the following recommendation system model.

Besides, all *InvoiceNo* starting with the letter C are canceled purchases and have a negative quantity of products. Out of a total of 18,523 unique invoice numbers, 3,654 invoices are canceled. They represent a refund ratio of 19.7%. After analyzing the canceled orders, no unique match was found with the related invoices as the ID is different. For instance, *Customer 17584* has two invoice numbers that have the same quantity and products, but the *InvoiceDate* of the

canceled entry has a difference of over one month. In this case, the invoice with the prefix C can be assumed to be the refunded order. In another example, *InvoiceNo C536391* does not have a matching purchase order. The *InvoiceDate* and *Index* shows that the refund order was made earlier in the records, meaning that it had occurred before the company captured the data.
Regarding the high refund ratio, it was deemed interesting to understand why customers tend to cancel so many orders. For that purpose, the *StockCode* column was decoded. The *StockCode* contains either a five-digit numeric value, a five-digit value, with at least one letter, or just a text without a numeric value. Looking at the number of orders, refund orders, and their corresponding percentages, it showed that the amount of total refund orders is higher than the number of orders. The description also indicates that there are not products, but additional costs, samples, or reduced fees. Therefore, the following StockCodes that did not represent a product, were deleted: 133 C2 (carriage), 1099 POST (postage), 11 BANK CHARGES, 16 CRUK (cruk commission), 3 PADS (pads to match all cushions), 77 D (discount), 454 M (samples), and 253 manual. There were a total of 2,284 Stockcodes. After these entries were deleted, the refund ratio decreased to 2.3%. Nevertheless, some products that have a refund rate of over 20%, should be looked at closely.

For further analysis, multiple features arose from feature engineering the data. That is, the *TotalPrice* was calculated from the *Unitprice* and *Quantity,* whereas the *InvoiceDate* generated new individual columns such as the *year, month, day, week, order time, weekday,* and *quarter*.
Moreover, only 7 outliers were identified and deleted. Four of them were found in the *Quantity* and three in the *TotalPrice*. By performing the data preprocessing, a total of 17,377 entries were deleted.

3.3. Exploratory Data Analysis

Exploratory data analysis is the process of performing initial investigations and analysis to extract as much knowledge as possible from the dataset and look out for unanticipated findings. This process is crucial since these findings could help in the implementation of the recommender system. That is, this report will cover selected visualizations and analyses we found particularly interesting.[1]

3.3.1. Most popular products and sales distributions by country

First, we take a closer look at ManyGiftsUK's portfolio to find the most ordered products. However, these items and their rank differ from country to country. Thus, it is interesting to explore the distribution of sales per country first to get a bigger picture of ManyGiftsUK's customers.

It can be observed that 90% of the customers are UK residents, followed by Germany and France with approximately 2%. Consequently, a similar distribution of sales by country can be seen. That is, UK customers account for over 80% of sales, however, this is an under-proportional performance considering that the majority of the customers are coming from the UK. These findings show that customers from outside the UK are making larger purchases and thus most likely represent foreign wholesalers. Furthermore, it is also worth mentioning that the country rankings differ according to the highest count of invoices and the total gross amount of sales. Thus, the top 3 countries for invoices count are the UK, Germany, France whereas the UK, the Netherlands and, Ireland are the most profitable regions.

Second, the most popular products are investigated. As already mentioned in *Part 3.1*. ManyGiftsUK offers a range of 4070 unique products (before data cleaning). The most popular items, without considering the quantity, are *"white hanging heart t-light holder"* followed by *"regency cakestand 3 tier"* and *"jumbo bag red retrospot"*. However, when taking into account the quantity ordered per product the most "popular" item is the *"world war 2 glider"* as shown in

[1] Note: All the figures in *Part 3.3. EDA* represents the numbers after data cleaning and preprocessing.

Figure 2. This is because a few customers, most likely wholesalers, have bought large quantities of this product. In particular, customer *16333* alone has bought over 10,000 pieces.

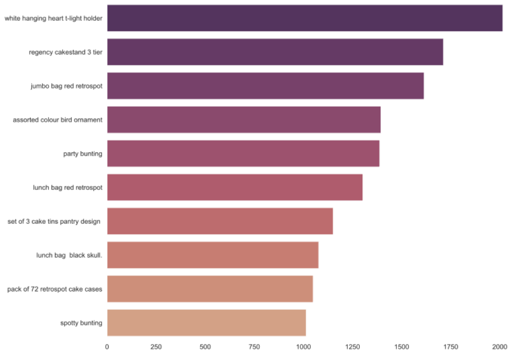

Figure 1: Most ordered items without quantity

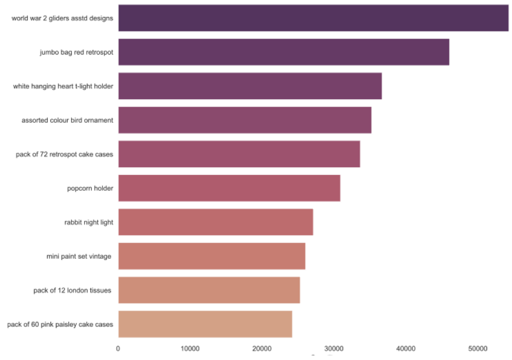

Figure 2: Most ordered items by quantity

3.3.2. Customers ranked according to their relative turnover compared to total revenues

Following the previous subsection, we know that wholesalers account for a significant share of ManyGiftsUK's customers. This finding is reinforced in Figure 3. One can see that only 5% of the customers (217 from a total of 4,333) account for almost 50% of the total revenues, which is marked by the orange line in Figure 3. To go even further, the visualization also shows that 20% of the customers represent even 75% of the total turnover.

A similar trend is visible when looking at the cumulative revenue share by product in section 4. Customers and Orders of the "EDA" Jupyter Notebook where 10% of the products explain around 60% of total revenues.

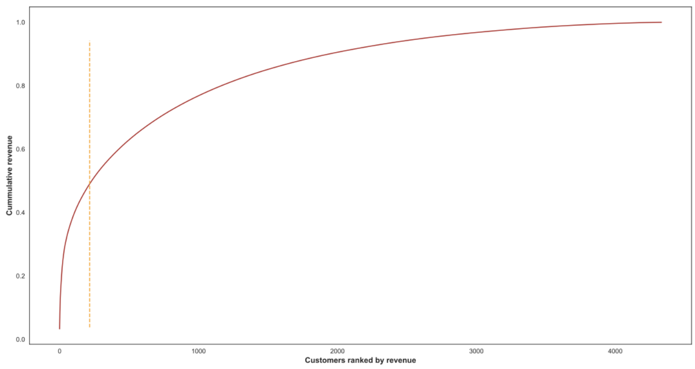

Figure 3: Cumulative revenue share by customer

3.3.3. Density of total price and quantity of orders

Lastly, the following histograms present even deeper insights into the densities of the *total price* and *quantity* per order. In *Figure 4* one can see that the majority of sales per order ranged between 1 and 15 pounds. *Figure 5* shows that the customers bought normally between 1-2 and 10-12 items. One could make the assumption that even if the wholesalers account for the highest revenues the individual customers make the most orders.

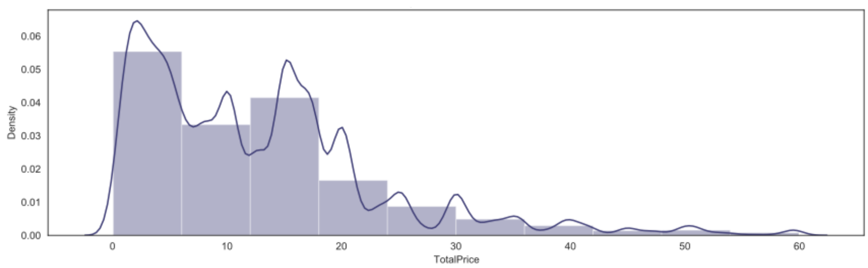

Figure 4: Density of total price of orders

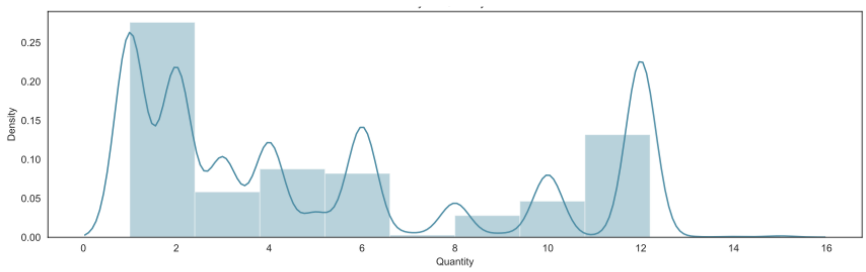

Figure 5: Density of quantity of orders

3.3.4. Further EDA

In the previous subsections, the purchasing behavior of customers was analyzed in detail. Furthermore, we found out that no important patterns can be formed when looking at the time dimension. However, from September onwards, the number of transactions and the purchase amount per order is higher than in the other months. Therefore, we conclude that Q3 had the highest sales in terms of both total amount and quantity, due to the pre-Christmas period and Christmas shopping. Besides, the average sales are constant from Monday to Friday. Sales start at 6 am and end at 8 pm, with hourly sales being normally distributed. Although the company offers a website, there is no operation on Saturday. Moreover, when the website is activated, an increasing number of members can be observed. Additionally, the percentage of repeat customers is 65.5%.

For even further visualizations and analyses, please refer to the Jupyter Notebook _2_BC4_G_EDA.ipynb.

3.4. Implicit Recommendation Systems

In this case, the focus is on implicit recommendation systems, as the dataset does not entail any explicit source of information. There are three essential forms of recommender systems that are mainly utilized in various machine learning platforms.

First of all, *Collaborative Filtering Recommendation System* is on the premise of the collection of information based on user's preferences. The basic notion of using such an algorithm is based on the assumption that the users are more likely to purchase or consume a specific product or service if they already had an experience with the respective product or service. The system recognizes similarities between users based on their consumption behavior and consequently suggests new recommendations based on this inter-user comparison. A common example for this recommender system is product suggestions on Amazon (People who bought *Product A* also bought *Product B*).

Secondly, the *Content-based Recommendation System* is based on the attributes of the content to create an individualized recommendation. Different features and events can be linked to the ratings of the users. As a result, the system can learn from the user profile and suggest appropriate recommendations [5]. A familiar instance for this recommender system would be the movie suggestions on Netflix (Because you watched *Movie X*, you would also like *Move Y)*.

Lastly, there are *Hybrid Recommendation Systems* based on combined collaborative filtering as both techniques mentioned above have both their strengths and weaknesses [5].

However, all of these models and techniques work based on data and input which can not function well when dealing with new users. This phenomenon is also known as the *Cold Start Problem* which is discussed in the next section of this report.

3.4.1. Cold Start Problem

A major challenge of the best-known recommendation system technique, the *Collaborative Filtering (CF)* approach, is to suggest items to new users which is also called the *cold-start user* problem. One method to deal with such a problem is to get to know the user beforehand by integrating a quick interview. The objective of the interview is to gain insights and information about the new user and feed that information to the recommender system. Consequently, the CF system can use this information to compute the similarity between new and other users.

An alternative approach to deal with this problem is the *Popularity Strategy*. In this case, the most popular items are shown to the new users: the items that are bought with the most frequency. Additionally, this approach can also include the recency rate of items and choose sort items by purchase frequency and recency. In this respect, the higher the frequency and the lower the recency, the better. Furthermore, the implementation of this method is uncomplicated and computationally inexpensive. In the case of explicit item ratings, the items with the highest ratings in conjunction with a minimum set of review sample size can be utilized to recommend to the new users.

In this case, the dataset given by ManyGiftUK entails information about the region of each new user who has made a purchase. If ManyGiftUK is able to identify the region of a new user prior to its first purchase, ManyGiftUK can utilize the *Popularity Strategy* and suggest the most popular items to the user given its region. Due to its simplicity, it is advisable for ManyGiftUK to use this strategy to tackle the cold-start problem and dive into more complicated techniques to approach this problem in the future.

For now, the dataset to build the recommender system incorporates users with prior purchasing history. In other words, the dataset is clean of new customers.

3.4.2. Model Building

3.4.2.1. Alternative Least Square

As mentioned before, CF System is a well-studied technique that will be utilized for building a recommendation system for ManyGiftsUK. With a variety of models to build a CF system, *Alternative Least Square (ALS)* is chosen as the main method for this project. The method is also used for building the web application that is discussed in the *Deployment* section. For this specific model, we implement a *Sparsity Reduction* method which eliminates the customers that have a frequency of fewer than five transactions and less than five item frequency. Consequently, 242 users are eliminated from the dataset. Similarly, the number of items is reduced to 3304 from the initial 3888 which results in a reduction of 584. The *Sparsity* after implementing this method is reduced to 2.923% while the initial *Sparsity* was 2.357%. Furthermore, the data is split into a train and a split dataset. The test split begins from the transaction that occurred after the 16th of March 2011.

As the next step, after the creation of the sparse matrices, a popular model is built which functions as the base model to evaluate the ALS model. The ALS model could be used to mainly suggest new products to a user by giving in the customer identification number in the function. Additionally, it can suggest similar items based on the item giving as input.

3.4.2.2. LightFM

The LightFM model is a Python recommendation algorithm for implicit and explicit feedback. One of the good aspects of this model is that it operates with all the information given on customers, products, and transactions history, and incorporates it into the traditional matrix factorization algorithm, in order to suggest the most relevant recommendations [6]. The Light FM uses both content-based methods and collaborative filtering.

Important to note that the dataset does not entail any explicit information about the item such as a score or rating. However, LightFM uses implicit information to suggest new items to users. On the one hand, item purchases are seen as a positive indication: The user does like the item otherwise it would not have purchased it. On the other hand, cancellation of the respective purchase is seen as a negative indication: The user made a mistake or does not have any further interest in the respective product. However, occasionally, the item may have been purchased as a gift for a friend, or the product may still have been not as expected after the purchase. In these cases, a purchase is not an indication of positive user-item interaction. As a one-time event might be caused by different reasons, a recurring event is likely to present the user's preferences. Thereby, the frequency of an event reflects the confidence in a certain observation. Therefore a new feature *EventStrength* was created, in this case, a purchase is worth 3.0 as a positive, and cancellation is worth -2.5 negative implicit feedback.

3.4.3. Model Evaluation

3.4.3.1. Alternative Least Square

The ALS model is evaluated based on the following metrics: Precision, Map, NDCG, AUC. In this case, Precision is a measure of exactness and determines the fraction of relevant items retrieved out of all items retrieved. For instance, the proportion of recommended items that are actually preferred. Furthermore, NDCG is a measure of ranking quality. In this case, the popular model has performed better than the ALS model in respect to all metrics mentioned. The mean AUC of the ALS model is 0.508 where the mean AUC of the popular model is 0.5231.
However, the LightFM model has performed superior which is discussed in the next section.

3.4.3.2. LightFM

As the LightFM library provides the evaluation metric of the AUC score, it was used to assess the performance of the trained model. It measures the ranking between two items and could be interpreted as a probability of a user preference for item A compared to item B. The trained LightFM model got an AUC score of 0.935 for the train data, against 0.864 for our test data. Which shows that the model might be a bit overfitted.

4. Evaluation

4.1. Business Objectives Review

Overall, the recommendations systems built meet the requirements and objectives of this project set by ManyGiftsUK. The EDA and analysis implemented, investigated the customers and items of the company and shed light on a few interesting insights which ManyGiftsUK is able to capitalize on. After assessing models in the previous section. The LightFM model has been superior in terms of mean AUC which could be utilized to recommend new products to customers and find similar products for a given product. As a result, the recommendation system facilitates user choices by recommending products in respect to their preferences and consequently improves the user experience by suggesting products that the user might find due to the existence of a high variety of product range.

4.2. Limitation and Future Work

In this section, some of the limitations of the recommender system will be discussed followed by proposed solutions that could be integrated into future work.
As mentioned before, for the cold-start problem which is a well-studied subject, there are a few options that could eventually propose a better solution than utilizing the popular strategy. As an alternative, ManyGiftsUK could conduct and design an introduction interview to generate insights about the user's preferences and inclinations. Thereby, the user could give out some information as an exchange for personalized item suggestions in return.
Furthermore, it's worth mentioning that the dataset to build the model has not been separated by user category - wholesalers, regular customers. Despite implementing outlier detection which excludes some of the invoices with a high amount of purchases of one specific item, the separation might lead to better results. As a result, two distinct models can be built - one for wholesalers and the other for regular customers. However, official differentiation of customer category should be a prerequisite step prior to proceeding with this suggestion.

5. Deployment

The knowledge gained will need to be organized and presented as such that the user (ManyGiftsUK in this case) could easily use it. In this case, a personalized web application has been developed based on the ALS algorithm. In contrast to an isolated model, this Recommender System offers an intuitive user experience.

The system offers two features: By making use of the *Customerid*, it will recommend the top ten items for the respective user with the corresponding *Stockcode, Description,* and *Score*. This recommendation feature is based on the user-item collaborative filtering algorithm which is based on the ALS model.

Similarly, by making use of the *Stockcode*, the system will suggest the top ten items that are similar to the given item with the corresponding *Stockcode, Description,* and Score. The second feature is based on an item-item collaborative filtering model.

It's important to note that by generating more data the system needs to be updated and adjusted to the new information prior to major changes in the data collection approach and in regular intervals.

5.1. Challenges

One of the challenges with using implicit feedback to build recommendation systems is the interpretation of signal strengths. In many cases, a user action such as purchase could be based on accident or simply be a gift for friends and relatives. On the other hand, a cancellation might not necessarily indicate a disinterest in that product but rather a logistic and distribution complication. For instance, the user might decide to purchase the item in a retail store to avoid and eliminate delivery waiting time. Overall, with the absence of explicit data, the interpretation of implicit data remains.

5.1. Recommendations

The dataset provided by ManyGiftsUK entails implicit data. The implicit information for user's preferences and disinterest are the purchase and cancellation behavior, respectively. It is advisable to start gathering explicit data from the users by asking for feedback regarding the items they have purchased and canceled. Thereby, better and more insightful models can be built.

Additionally, ManyGiftsUK has the opportunity to expand its implicit feedback collection by adding a tracking system of user behavior while it is browsing on the website. Information dimensions such as *Clicks*, *Items in basket* could be utilized to enrich the implicit data for model optimization.

6. Conclusion

Overall, we were able to help ManyGiftsUK and its management to achieve their initial business objectives, which was to build and implement a recommender system that facilitates user choices by recommending items depending on their preferences and to improve the user experience when making purchases on their website. Before model building and selection, we analyzed the entire dataset to identify some type of customer's behaviors. Besides, we performed data cleaning by removing outliers and feature engineering for further information.

Moreover, we built two collaborative filtering recommendation models and evaluated their respective performance by mainly the mean AUC. As a result, the LightFM model has shown superior performance relative to the ALS model.

Lastly, the challenge was not only to come up with a recommender system but also to develop and deploy an environment in which the system can be used very easily. Thereby, a web application has been built to meet this objective.

7. References and Sources

[1] G. Adomavicius and A. Tuzhilin, “Towards the Next Generation of Recommender Systems: A Survey of the State-of-the-Art and Possible Extensions”, IEEE Transactions on Knowledge and Data Engineering 17 (2005), 634–749.

[2] Iyengar, S. S., & Lepper, M. R. When choice is demotivating: Can one desire too much of a good thing?. Journal of personality and social psychology 79(6), (2000), 995.

[3] Hu, Y., Koren, Y., & Volinsky, C. . Collaborative filtering for implicit feedback datasets. 2008 Eighth IEEE International Conference on Data Mining (2008), 263–272.

[4] D.W. Oard and J. Kim, “Implicit Feedback for Recommender Systems”, Proc. 5th DELOS Workshop on Filtering and Collaborative Filtering, pp. 31–36, 1998.

[5] Burke, Robin. “Hybrid Recommender Systems: Survey and Experiments”. In: User Modeling and User-Adapted Interaction 12.4 (2002), pp. 331–370.

[6] Kula, M., 2021. Welcome to LightFM’s documentation! — LightFM 1.15 documentation. [online] Making.lyst.com. Available at: <https://making.lyst.com/lightfm/docs/home.html> .