A Machine Learning Based Fashion Recommendation System

Shailja Tyagi

Abstract

Suggestion algorithms which present content based on previous preferences have become popular in recent years on platforms such as TikTok, Instagram, and YouTube. These algorithms connect individuals with products they want to buy or consume, providing value to both the individual and retailers and creators on the platform.We propose to create a similar suggestion algorithm for apparel and fashion. Our algorithm will utilize supervised machine learning over a large pre-existing dataset of apparel purchases to provide fashion suggestions based on explicitly stated individual preferences. This approach can assist individuals who want to discover more outfits, or struggle to find outfits which truly resonate with their personal style, and can help connect retailers with potential customers.

The code for this work is available here.

I. Introduction

Fashion, a potent form of self-expression, plays a pivotal role in our lives, shaping our confidence and projecting our unique identities. Yet, the modern fashion landscape, brimming with endless choices, often leaves individuals grappling with the challenge of creating a wardrobe that resonates with their personal style. The abundance of options can lead to frustration and indecision, making the search for the perfect outfit an overwhelming endeavor.

In response to this fashion conundrum, we can harness the potential of supervised machine learning algorithms. By leveraging an extensive dataset comprising apparel purchases and user preferences, this algorithm offers a solution by generating personalized recommendations that resonate with individual styles. This transformative approach simplifies the process of discovering outfits that resonate with a user's unique fashion preferences and enhances their confidence in their choices.

Recommender systems, a dynamic part of advancing technology, have a significant role in this context. These systems, rooted in machine learning, are designed to provide tailored predictions to users across various digital platforms. They draw insights from user behavior and preferences to offer recommendations that go beyond generic suggestions, continually evolving to adapt to the expanding supply range and customer base of online services.

This approach not only benefits individuals by boosting confidence in their fashion choices but also holds value for retailers. As it can effectively connect them with potential customers seeking attire which appeals to their preferences.

II. Related Work

While several existing approaches aim to provide outfit ideas to users based on pre-made combinations (Isinkaye 261-273), our approach takes a distinct route by leveraging pre-existing

datasets comprising apparel purchases and user preferences to predict new outfits tailored to individuals' personal styles through a ranking system. An interesting approach concerning item-based filtering and mix, and match features included the Dresoo App (Outfit Builder – Dresoo). This app easily prompts the user to decide what type of article of clothing they want and provides them with a list of similar items compared to the pre-selected item by the user. Another approach has been proposed by Analytics Vidhya, a movie recommender ("Movie Recommendation System"). Their approach included item-based filtering, KNN, and cosine similarity. These three elements utilized in their approach are typically the most common approach to building recommendation algorithms. Their overall goal was to create a movie recommendation algorithm based on previously rated movies and suggest similar movies which the user would most likely enjoy watching.

III. Background

A. Pandas

Python Pandas is a powerful and widely used open-source library designed for data manipulation and analysis. It provides easy-to-use data structures, such as data frames and series, that enable users to efficiently handle and manipulate structured data. Pandas is particularly adept at handling tabular data, akin to spreadsheets or SQL tables, making it a go-to tool for data cleaning, exploration, and transformation. With its intuitive and versatile functions, Pandas allows users to load data from various file formats, filter and select data, perform computations, and generate summary statistics. It serves as an essential component in the data science toolkit, facilitating data preparation and preprocessing tasks that are fundamental to data analysis and machine learning workflows.

B. Recommender Systems

A recommender system's main goal is to estimate ratings for items and provide a personalized list of the recommended items to a given user (Stieg). There are many different approaches to this task, such as the comparison of ratings, item or user attributes and demographic relations.

Most basic recommender systems include fundamental elements, such as a group of users, a collection of items, and the connection between them, typically represented by user-item ratings. These ratings are commonly expressed as integers, often using a scale ranging from one to five.

Recommender systems filtering can be categorized into two main sections:

1. Item-based collaborative filtering: relies on analyzing a set of similar items (bought and not bought in our case) consisting of received user ratings per item, the items which have been rated yet are used to recommend similar items by picking out N items from the similarity list that have been rated by U and calculating the rating based on these N ratings.

2. User-based collaborative filtering: relies on analyzing a similar set of users determined by their ratings for a specific item and then recommending other items

which haven't been rated yet or in this case purchased by picking out a few users who have rated an item and then calculating a new rating based on these N number of ratings given by others.

In the H&M dataset we are using item based filtering for our algorithm.

C. Cosine-based Similarity

Cosine-based similarity is a technique used to measure similarities between items in item-based collaborative filtering. In this approach, each item is represented as a vector within a user-space. To calculate similarity, the method involves computing the angle between two of these vectors and then determining the cosine of that angle, which quantifies the similarity between the items ("Cosine Similarity"). For instance, when comparing two items, denoted as "a" and "b," their similarity is determined through this cosine-based calculation.

$$Cos\theta = \frac{\vec{a} \cdot \vec{b}}{\|\vec{a}\| \, \|\vec{b}\|} = \frac{\sum_1^n a_i b_i}{\sqrt{\sum_1^n a_i^2} \sqrt{\sum_1^n b_i^2}}$$

Cosine based similarities formula

D. Supervised Machine learning

In this work, we utilize supervised machine learning. Supervised machine learning is a subset of machine learning that uses labeled datasets to train algorithms to classify data or predict outcomes accurately. This method is helpful in evaluating the model's accuracy through predictions. During the training phase, the system is fed with labeled data sets, which instruct the system what output variable is related to each specific input value. The trained model is then presented with the test data. The data presented is the labeled data, but the labels haven't been revealed to the algorithm yet. The overall goal of the testing data is to measure how accurately the algorithm performs on unlabeled data. In simple terms, the supervised learning process is improved by constantly measuring the resulting outputs from the testing data and fine-tuning the algorithm to get it as accurate as possible.

E. Evaluation

To evaluate the performance of recommender algorithms, it's common practice to test them against various datasets. In this paper, cross-validation is employed to assess algorithm prediction performance, as elaborated in the following subsection. Evaluating algorithm prediction performance involves the use of different metrics, and the choice of which metric to employ depends on the specific objectives of the algorithm and the goals of the measurement. The subsequent section will outline the metrics relevant to this paper.

F. Cross-validation

Cross-validation, sometimes referred to as rotation estimation, is a statistical data analysis technique used to validate the performance of algorithms when paired with a dataset. Its primary objective is to assess how well an algorithm generalizes to new data. Cross-validation allows for the comparison of different algorithms by measuring their performance (SciKit-Learn).

In cross-validation, a dataset is divided into "z" equally sized partitions, with one of these partitions serving as the test set, while the remaining "z - 1" partitions become the training sets. The algorithm is then trained using the training partitions. After completing the training, the model is evaluated using the test partition, generating test data. This process repeats until each partition has been used as the test set.

G. Dataset

For this algorithm, we will utilize an existing, publicly available dataset, specifically the H&M Personalized Fashion Recommendations dataset (“H&M Personalized Fashion Recommendations”). Within this dataset, we will focus solely on extracting the article IDs and customer IDs. While the dataset originally contains a comprehensive list of transactions, including information about who made the purchase and the specific items they bought, our analysis will narrow down to the essential article and customer IDs for the purpose of building our recommendation system.

IV. Methodology

The fundamental concept for the algorithm is that given, a list of customer purchased rated articles of clothing, the algorithm needs to be able to predict a ranking of how likely to least likely of non-purchased articles of clothing a score of some sort which recommends the customer articles of clothing which may resonate with their style since it would be similar to their taste in fashion based on the rated articles of clothing.

For instance, if a customer really likes wearing sweaters, the algorithm should be able to recommend another sweater which the customer may like depending on its predicted score.

A. Manipulating the data

	t_dat	customer_	article_id	price	sales_channel_id
14	########	000aa7f0c	5.02E+08	0.016932	2
15	########	000aa7f0c	5.02E+08	0.016932	2
16	########	000aa7f0c	6.75E+08	0.008458	2
17	########	000aa7f0c	6.72E+08	0.033881	2
18	########	000aa7f0c	6.72E+08	0.033881	2
19	########	000aa7f0c	6.32E+08	0.033881	2
20	########	000aa7f0c	6.32E+08	0.033881	2
21	########	000aa7f0c	6.32E+08	0.033881	2
22	########	000aa7f0c	6.32E+08	0.033881	2
23	########	000aa7f0c	6.8E+08	0.016932	2
24	########	000aa7f0c	6.77E+08	0.042356	2
25	########	000aa7f0c	6.77E+08	0.042356	2
26	########	000aa7f0c	6.86E+08	0.016932	2
27	########	000aa7f0c	6.86E+08	0.016932	2
28	########	000aa7f0c	6.81E+08	0.016932	2
29	########	000aa7f0c	6.81E+08	0.016932	2
30	########	000aa7f0c	6.92E+08	0.025407	2
31	########	000aa7f0c	6.92E+08	0.025407	2
32	########	000aa7f0c	6.41E+08	0.010153	2
33	########	000aa7f0c	6.41E+08	0.010153	2
34	########	000aa7f0c	6.64E+08	0.016932	2
35	########	000aa7f0c	6.64E+08	0.016932	2
36	########	000aa7f0c	6.81E+08	0.011847	2
37	########	000aa7f0c	6.81E+08	0.011847	2
38	########	000aa7f0c	5.53E+08	0.033881	2
39	########	000aa7f0c	5.53E+08	0.033881	2
40	########	000aa7f0c	3.77E+08	0.008458	2
41	########	000aa7f0c	3.77E+08	0.008458	2
42	########	000aa7f0c	7.01E+08	0.042356	2
43	########	000aa7f0c	7.01E+08	0.042356	2
61	########	00401a36	6.13E+08	0.016932	2
62	########	00401a36	6.34E+08	0.010153	2

The original H&M dataset

	article_id	000aa	00401	01f59
0	501820043	[2]	0	[10]
1	674681001	[35]	0	0
2	671505001	[2]	[26]	[41]
3	631848002	[26]	0	0
4	680187001	[48]	[48]	[21]
5	676827002	[49]	0	0
6	685687002	[32]	[19]	[26]
7	680912006	[38]	0	[36]
8	692454002	[6]	0	0
9	640639001	[42]	[39]	0
10	664421002	[25]	0	[14]
11	680912009	[18]	0	0
12	553139001	[35]	0	[48]
13	377277001	[12]	[18]	[8]
14	700819006	[46]	[29]	[33]
15	613456009	0	[35]	0
16	633675001	[47]	[16]	[42]
17	648719001	0	[24]	[24]
18	427114015	0	[11]	0
19	567475001	[26]	[31]	[26]
20	567594001	0	[33]	[48]
21	681358001	[18]	[42]	0
22	613456001	[13]	[28]	[15]
23	573937001	[45]	[33]	0
24	622745001	[38]	[15]	0
25	617322004	0	[36]	0
26	657497007	0	[12]	[9]
27	625773001	0	0	[10]
28	674336001	0	0	[40]
29	456163026	[25]	0	[42]

The transformed H&M dataset

In order to create a recommendation algorithm, the pre-existing dataset needed to be manipulated into a new dataset called ‘matrix’ through pandas. The new dataset consists of sets of purchased articles of clothing (article_id), sets of customers who have purchased items (customer_id), and sets of ratings per article of clothing. Each row would contain an article id and rating given by each customer (ratings).

Before finding out the perfect layout for the matrix, it was initially laid out so that it consists of one column of customer ids and several columns of ratings given by the customer, where its label is the article id. However, this proved to be ineffective since it was more reasonable to have each column represent the purchases for each customer and iterate through each column than the first item in each column to implement the item-based filtering technique.

	customer_id	501820043	674681001	\
0	000aa7f0dc06cd7174389e76c9e132a67860c5f65f9706...	[2]	[2]	

	671505001	631848002	680187001	676827002	685687002	680912006	692454002	...	\
0	[3]	[2]	[2]	[1]	[2]	[1]	[1]	...	

	567594001	681358001	613456001	573937001	622745001	617322004	657497007	\
0	0	0	0	0	0	0	0	

	625773001	674336001	456163026
0	0	0	0

Initial Dataframe

This dataset consists of a file which contains a large dataset of the purchases for each customer, as well as additional irrelevant information. A visualization of the file that is used:

t_dat	customer_id	article_id	price	sales_channel_id
19Sep18 – 21Sep20	1362281 unique values	109m – 956m	0 – 0.59	1 – 2
2018-09-20	000058a12d5b43e67d225668fa1f8d618c13dc232df0cad8ffe7ad4a1091e318	0663713001	0.050830508474576264	2
2018-09-20	000058a12d5b43e67d225668fa1f8d618c13dc232df0cad8ffe7ad4a1091e318	0541518023	0.03049152542372881	2
2018-09-20	00007d2de826758b65a93dd24ce629ed66842531df6699338c5570910a014cc2	0505221004	0.01523728813559322	2
2018-09-20	00007d2de826758b65a93dd24ce629ed66842531df6699338c5570910a014cc2	0685687003	0.0169322033898305 08	2
2018-09-20	00007d2de826758b65a93dd24ce629ed66842531df6699338c5570910a014cc2	0685687004	0.0169322033898305 08	2
2018-09-20	00007d2de826758b65a93dd24ce629ed66842531df6699338c5570910a014cc2	0685687001	0.0169322033898305 08	2

transactions_train a list of transactions and customer and article ids.

This file contains over 956 million transactions made by about 1.4 million customers which can appear to be too much data and cause the algorithm to run in a slow amount of time. So, the original dataset was shortened and used to predict ratings of the items not purchased by the customers.

Final Dataframe

	article_id	000aa
0	501820043	[2]
1	674681001	[35]
2	671505001	[2]
3	631848002	[26]
4	680187001	[48]
5	676827002	[49]
6	685687002	[32]
7	680912006	[38]
8	692454002	[6]
9	640639001	[42]
10	664421002	[25]
11	680912009	[18]
12	553139001	[35]
13	377277001	[12]
14	700819006	[46]
15	613456009	0
16	633675001	[47]
17	648719001	0
18	427114015	0
19	567475001	[26]
20	567594001	0
21	681358001	[18]
22	613456001	[13]
23	573937001	[45]
24	622745001	[38]
25	617322004	0
26	657497007	0
27	625773001	0
28	674336001	0
29	456163026	[25]

The article_id column shows several different ids, each representing an article's clothing. This included both purchased and non-purchased articles of clothing.

The ratings columns show several customers who rated some of the items on a scale from 1-5. For instance, the first customer rated the first article of clothing, 501820043, a 2. (Randomly chosen)

The articles of clothing which received a rating of 0, means that it hasn't been purchased. And those are the articles of clothing which will receive a rating of how likely the customer would be most likely to purchase an item based on their other previously purchased items, through the algorithm.

B. Training The data

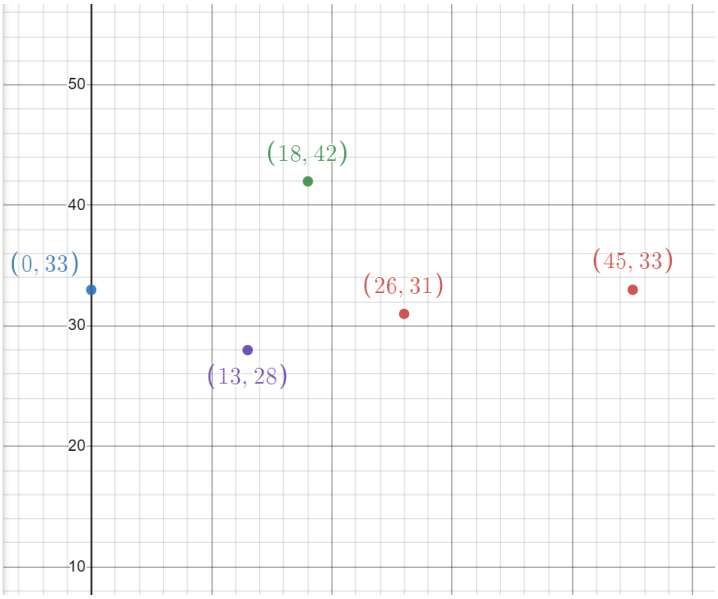
With the H&M dataset, the dataset would be split into training and testing datasets. The training dataset would be a subset of one customer and their purchases. The testing dataset would be a subset of the other customers excluding the customer used for the training dataset and their purchases. After training, the model should then be capable of predicting a rating of how likely the customer would be most likely to purchase an item based on the given, previously purchased items. And to evaluate the model's accuracy, the algorithm would run the testing data (unlabeled data) and predict a new output. And if the output seems to predict a somewhat reasonable rating, then the model would prove to be accurate.

The training of the algorithm was conducted by implementing item-based collaborative filtering, the details of which are elaborated below:

In this example, the rating for article id 567594001, by customer 000aa has no rating, signifying it hasn't been purchased yet...

19	567475001	[26]	[31]
20	567594001	0	[33]
21	681358001	[18]	[42]
22	613456001	[13]	[28]
23	573937001	[45]	[33]

There are several ways to find the nearest article of clothing. Here, I use the cosine similarity. Through using cosine similarity, you need to have the non-purchased item to have a value of 0. Look at the graph below. The x-axis represents ratings by customer 000aa and the y-axis rating ratings by customer 00401. Then, we can find points for each movie in the space. For instance, article id 567475001 corresponds to the point (26,31) in space.



Graphed article ids

The cosine similarity uses cos(θ) to measure the distance between two vectors. As θ increases, cos(θ) decreases (cos(θ) = 1 when θ = 0 and cos(θ) = 0 when θ = 90). Therefore, as the value of θ is smaller, the two vectors are considered closer (the similarity increases).

There are several ways to determine the numerical distance between each article id, however I used the NearestNeighbors method. NearestNeighbors() in the sklearn.neighbors library (IBM) can be used to calculate the distance between articles of clothing using the cosine similarity and find the "nearest neighbors" for each article of clothing.

Using this method, it returns distances and indices.

```
indices:  [[ 0  1  2  3  4  8 26 27 18 11 10 22 15 17 28 25 13 21  6 24  5  7 19  9
  14 23 16 20 12 29]
```

The indices show the indices of the nearest neighbors for each article of clothing. Each row corresponds to the row in the matrix. The first element in a row is the most similar (nearest) article of clothing. Generally, it's the article of clothing itself. The second element is the second nearest, the third is the third nearest, so on so forth. For example, the indices shown indicate the nearest neighbors for the article of clothing 680912009. The first nearest article of clothing is 674681001, then 671505001, etc.

```
distances:  [[0.00000000e+00 0.00000000e+00 1.11022302e-16 1.11022302e-16
  1.11022302e-16 1.11022302e-16 1.11022302e-16 1.11022302e-16
  1.11022302e-16 1.11022302e-16 1.11022302e-16 1.11022302e-16
  1.11022302e-16 1.11022302e-16 1.11022302e-16 1.11022302e-16
  1.11022302e-16 1.11022302e-16 1.11022302e-16 2.22044605e-16
  2.22044605e-16 2.22044605e-16 2.22044605e-16 2.22044605e-16
  3.33066907e-16 3.33066907e-16 3.33066907e-16 3.33066907e-16
  3.33066907e-16 5.55111512e-16]
```

The distances show the distance between each article of clothing. The smaller the number, the more closer (similar) it is by ratings. Additionally, each value in the array corresponds to the number in the indices array.

Calculating the predicted rating using the distances.

The formula to calculate the predicted rating is as follows:

$$R(\mathbf{a_1}, c) = \{\sum_{a2}, S(\mathbf{a_1}, \mathbf{a_2})R(\mathbf{a_2}, c)\}/ \sum a_2\, S(\mathbf{a_1}, a_2)$$

$R(\mathbf{a_1}, c)$: the rating for each article of clothing $\mathbf{a_1}$ by customer **c**

$S(\mathbf{a_1}, \mathbf{a_2})$: the similarity between article of clothing, $\mathbf{a_1}$ and article of clothing $\mathbf{a_2}$

$j \in J$ where J is the set of the similar articles of clothing to article $\mathbf{a_1}$

This formula simply implies that the predicted rating for an article of clothing $\mathbf{a_1}$ by customer (**c**) is the weighted average of ratings for the similar articles of clothing by customer **c**. The weight for each rating, $(S(a_1, k)/\sum_{a2} S(a_1, a_2))$, becomes greater when article $\mathbf{a_1}$ and article **k** are closer. The denominator of this term makes the sum of all the weights equal 1.

C. Validation

To validate the effectiveness of my algorithm, I employed a masking technique. In this validation approach, the primary objective was to assess the system's ability to accurately predict ratings for five articles of clothing without prior knowledge of whether they had been purchased. The rationale behind this masking process is crucial—it ensures that the system does not have access to the answers (i.e., which items were actually bought) before being tasked with the question of how to rank these items. To execute this, I initiated the process by selecting the first five article IDs. These IDs were "masked" by assigning each of them a value of -1.

Consequently, the system was intentionally blinded to whether these five items had received ratings or not. However, for the remaining articles, excluding the selected five, the system had access to their article IDs and associated ratings, which it would utilize to predict ratings for the masked items. This rigorous approach helped in evaluating the system's predictive capabilities without any prior knowledge of the actual purchases.

To evaluate the performance of the algorithm, I utilized the predict() method, which used a new validation matrix consisting of both masked and unmasked items. The evaluation criterion involved examining the algorithm's output. If it correctly ranked all rated items in the order of most likely to least likely to be purchased, the algorithm received a score of 1. In cases where one or two items were inaccurately ordered, it received a score of 0.5. If three or more items were incorrectly ranked, the algorithm received a score of 0. This validation process helped evaluate the algorithm's predictive accuracy in real-world scenarios if someone needed an outfit.

V. Experimental results

	article_id	000aa	00401	01f59
0	501820043	[2]	0	[10]
1	674681001	[35]	0	0
2	671505001	[2]	[26]	[41]
3	631848002	[26]	0	0
4	680187001	[48]	[48]	[21]
5	676827002	[49]	0	0
6	685687002	[32]	[19]	[26]
7	680912006	[38]	0	[36]
8	692454002	[6]	0	0
9	640639001	[42]	[39]	0
10	664421002	[25]	0	[14]
11	680912009	[18]	0	0
12	553139001	[35]	0	[48]
13	377277001	[12]	[18]	[8]
14	700819006	[46]	[29]	[33]
15	613456009	0	[35]	0
16	633675001	[47]	[16]	[42]

```
There are 8 number of recommended articles.
1: 567594001
2: 657497007
3: 427114015
4: 617322004
5: 613456009
6: 648719001
7: 674336001
8: 625773001
```

In our research, we observed qualitative results that highlight the system's ability to make recommendations based on previous user purchases. For instance, if a user expressed a preference for specific items, the system inferred that they might also like other items. For example, if a person bought item 1 and item 2, the system predicted they might have an affinity for item 3. Additionally, we found that the system leveraged collective user behavior, such as if person A bought item 1 and person B bought item 2, it would predict that person C might be interested in item 2 due to its popularity among similar users. An illustrative example of this process can be seen in the input of article id 63367501, which led to the highly recommended output of article id 567594001. This showcases how the system effectively recommends items that are likely to resonate with the user's preferences, based on the analysis of past user purchases and preferences.

Below, the first customer prefers the article id 633675001 the most and the algorithm recommends article id 567594001.

Preferred article of clothing

Recommended article of clothing

In our results analysis, we observed a specific case where the algorithm's recommendations aligned closely with user preferences. For the first customer, who expressed a strong preference for article ID 633675001 based on their rating, the algorithm accurately recommended article ID 567594001. This particular recommendation was for a light blue woven occasion dress designed for ladieswear, featuring a long satin design with a V-neck and wrapover front, concealed fastenings, and side slits in the skirt. The algorithm's ability to suggest a relevant item showcases its proficiency in understanding individual style preferences. Additionally, the algorithm recommended article ID 567594001 to the same customer, which was a cardigan in off-white from the knitwear category. This fine-knit cropped cardigan contained subtle glittery threads and featured a button-down front. These recommendations underscore the algorithm's effectiveness in providing personalized fashion suggestions that resonate with the user's expressed preferences.

Additionally, the validation score recorded an impressive value of 0.9403131991051454, approaching the maximum score of 1. This result underscores the algorithm's exceptional performance and its ability to accurately predict user preferences, making it a robust and reliable fashion recommendation system.

VI. Challenges and limitations

In the development and implementation of our algorithm, we encountered several notable limitations and challenges. One challenge stemmed from the algorithm's ability to operate effectively when provided with articles of clothing lacking any attributes or images. This limitation primarily pertained to the absence of visual and descriptive information about the clothing items, which are often critical factors in fashion recommendation systems. Without access to attributes or images, the algorithm is most likely to face difficulties in understanding and categorizing the items effectively, potentially leading to less accurate recommendations. Furthermore, the number of data in our dataset presented a significant challenge. Given the vast number of customers and items, encompassing thousands of data points, processing the entire dataset became a daunting task, leading us to select subsets for analysis.

Additionally, one key limitation lies in the algorithm's omission of attribute-based recommendations. It does not consider specific item attributes such as the type of clothing, color, and style, potentially hindering its usefulness for users seeking clothing items based on particular attributes. Moreover, the algorithm does not incorporate imagery data, which limits its ability to provide recommendations based on visual aspects.

While addressing these challenges and limitations, it's essential to acknowledge that scalability, computational efficiency, and data sparsity are recurring challenges in the field of recommender systems. Balancing the need for comprehensive data analysis with practical computational constraints remains an ongoing concern.

References

"Cosine Similarity." GeeksforGeeks, 2 Oct. 2020, www.geeksforgeeks.org/cosine-similarity/.

"H&M Personalized Fashion Recommendations." Kaggle.com, www.kaggle.com/competitions/h-and-m-personalized-fashion-recommendations/data?select=transactions_train.csv.

IBM. "What Is the K-Nearest Neighbors Algorithm? | IBM." Www.ibm.com, 2023, www.ibm.com/topics/knn.

Isinkaye, F.O., et al. "Recommendation Systems: Principles, Methods and Evaluation." Egyptian Informatics Journal, vol. 16, no. 3, Nov. 2015, pp. 261–273, www.sciencedirect.com/science/article/pii/S1110866515000341, https://doi.org/10.1016/j.eij.2015.06.005.

Karabiber, Fatih. "Cosine Similarity." Www.learndatasci.com, www.learndatasci.com/glossary/cosine-similarity/.

Dresoo – the Best Online Outfit Creator. dresoo.com/.

Real Python. "Build a Recommendation Engine with Collaborative Filtering." Realpython.com, Real Python, 10 July 2019, realpython.com/build-recommendation-engine-collaborative-filtering/.

"Recommendation System Using K-Nearest Neighbors |Use Case in Python." Analytics Vidhya, 20 Aug. 2020, www.analyticsvidhya.com/blog/2020/08/recommendation-system-k-nearest-neighbors/.

SciKit-Learn. "3.1. Cross-Validation: Evaluating Estimator Performance — Scikit-Learn 0.21.3 Documentation." Scikit-Learn.org, 2009, scikit-learn.org/stable/modules/cross_validation.html.

Stieg, Cory. "What Is a Recommender System in Machine Learning?" Codecademy Blog, 6 Dec. 2022, www.codecademy.com/resources/blog/what-is-recommender-systems-machine-learning/.

Zach. “Leave-One-out Cross-Validation in Python (with Examples).” Statology, 4 Nov. 2020, www.statology.org/leave-one-out-cross-validation-in-python/.