FACULTY
OF INFORMATION
TECHNOLOGY
CTU IN PRAGUE

ASSIGNMENT OF MASTER'S THESIS

Title:	Recommendation based on product images
Student:	Bc. Kristýna Tauchmanová
Supervisor:	doc. Ing. Pavel Kordík, Ph.D.
Study Programme:	Informatics
Study Branch:	Knowledge Engineering
Department:	Department of Applied Mathematics
Validity:	Until the end of winter semester 2020/21

Instructions

Survey recommender systems and focus on recommendation algorithms using product images (not exclusively). Design recommender systems for various scenarios such as homepage, related products, shopping cart or recommnedation based on image provided. Evaluate performance of algorithms using offline data provided by Recombee. Discuss an improvement of recall and catalog coverage thanks to image embedding (if any).

References

Will be provided by the supervisor.

Ing. Karel Klouda, Ph.D.	doc. RNDr. Ing. Marcel Jiřina, Ph.D.
Head of Department	Dean

Prague February 18, 2019

FACULTY
OF INFORMATION
TECHNOLOGY
CTU IN PRAGUE

Master's thesis

Recommendation based on product images

Bc. Kristýna Tauchmanová

Department of Applied Mathematics

Supervisor: doc. Ing. Pavel Kordík, Ph.D.

August 3, 2020

Acknowledgements

I would like to thank my supervisor, Pavel Kordík, for his guidance and valuable advice. In the second place, I would like to thank my parents for the constant support not only during studies and lastly to my fellow students and closest friends who made the whole university experience enjoyable and the hard days bearable.

Declaration

Citation of this thesis

Tauchmanová, Kristýna. *Recommendation based on product images.* Master's thesis. Czech Technical University in Prague, Faculty of Information Technology, 2020.

Abstrakt

Cílem této diplomové práce je analýza doporučovacích sytémů a návrh takových systémů s použitím obrázků produktů. Práce obsahuje analýzu poskytnutých dat a návrh doporučovacích algoritmů pro různé scénáře. Součástí práce je i teoretický úvod do problematiky doporučovacích sytémů a zpracování obrazu. V závěru se práce také zabývá offline vyhodnocením navržených modelů.

Klíčová slova Doporučovací systémy, zpracování obrazu

Abstract

The aim of this master thesis is the analysis of recommendation systems and the design of such systems using product images. The work includes analysis of the provided data and design of recommendation algorithms for various scenarios. A part of the work is also a theoretical introduction to the issues of recommendation systems and image processing. At the end, the thesis also deals with the offline evaluation of the proposed models.

Keywords Recommendation systems, image processing

Contents

List of Figures

List of Tables

Introduction

E-commerce websites usually store a huge amount of information. They offer tens of thousands of products which, for a common customer, is unrealistic to go through. The customer is then lost in the wide range of offer and without proper navigation, it is hard to get oriented. In the optimistic cases, the web site is a basis of a large number of products uncovered with user interactions, in the pessimistic cases, it can lead to a loss of customers.

Recommendation systems often work with user interactions on the web site and recommend products based on user behaviour. But products itself can be a source of information too. Especially product images (which is often crucial in online retail) can help to understand the user behaviour better and by recommending the right products, guide the customer through the site comfortably.

The objective of this thesis is to survey recommendation systems and design such systems with the usage of product images.

This thesis is logically divided into five chapters. The first chapter introduces recommendation systems, describes the data they are using, presents several scenarios in which recommendation systems can be used, introduce three basic recommendation techniques and defines common evaluation metrics.

The second chapter deals with image processing, detecting of image features, dividing image features into two basic categories (low-level and high-level features), explains the difference between them and in the end describes methods for extracting image embeddings.

The third chapter surveys recent related works, studies their usage of the images in recommendation systems and related tasks, and divides the research work into three categories according to the used recommendation technique.

The fourth chapter introduces the architecture of a common recommendation system, analyses the provided dataset, studies user behaviour and examines product images. Based on the observations and previous research, it proposes several recommendation algorithms, which are then evaluated and the results are presented in the last, fifth chapter.

Chapter 1

Recommendation systems

This first chapter introduces recommendation systems, explains their purpose and describes the data they are using. It also presents several scenarios in which recommendation systems can be used, introduces three basic recommendation techniques and defines common evaluation metrics.

1.1 Purpose

Learning from others is our natural behaviour. Sharing knowledge and experience was helping us survive from the very beginning and our instincts remained the same even at present. In ancient times, a human, as a community being, passed information about sources of food, so the group of people he was living with could get fed and outlive winter and so he himself. In the modern world and developed society, we do not have to worry about the basic needs that much, but the urge to make our life easier remained. We still learn from the experience of others and trust people we are close to. We listen to their recommendations and behave according to them. We watch movies our friends like, we buy food our parents used to (and we loved as children), we visit places that many people before us visited and wrote a positive review on some social network...

In the online world, this is what recommendation systems do - they search for users similar to us, study their preferences and recommend us products they liked and we have not discovered yet. They try to recommend us items we might like and help us explore a wider range of products we are looking for or we could be interested in on basis of our past behaviour on a web site and on the behaviour of others.

The trend of using recommendation systems is supported by an increase in online services such as e-shops, social networks, streaming services and other types of e-commerce. Online streaming services such as Netflix or Spotify contain millions of items (videos or songs) which a common visitor cannot process all. Recommendation systems help her/him get oriented and guide through content which can be relevant for her/him. Such platforms collect data about users and their interaction with the web page. Typically it is a song they listened to, a music interpreter they followed, a video they clicked on or rating they gave to some product. According to these historical data, recommenders offer relevant items (songs, series, movies, ...) at present. They can also take context into consideration. Users' desires change during the day, for example, someone wants to wake up listening to pop music, work with acoustic music for better concentration and listen to experimental rock in the evening. Weather or season of the year could be important, too. December is a typical time of Christmas carols and fairy tales, spring could be a good opportunity to buy another umbrella. All this information can be used to predict user preferences.

1.2 Data

E-commerce sites are great sources of data. Visitors of the web pages "walk" through web pages, read or watch the content, click on links and interact in many other ways. To create a recommendation model, it is needed to store and process this information. We can divide this data into three categories:

- items,
- users,
- interactions.

They will be described in the next subsections.

1.2.1 Items

Items can be any product that the online service offers to their customers. In case of streaming services, we talk about videos or music, in e-shops, there are products like electronics, books or furniture, or in case of online newspapers, even news articles.

Every item can be described by various attributes. For example, a dress has its colour, used fabric, length or size, headphones have their driver size, sensibility or impedance, articles have a date of publication and article itself.

Some of the attributes may be computed by the recommendation system. From the unstructured text (description of the item, article, . . .) we can extract tags or category of the item. An image of the item can be processed and its features extracted to lower their dimension from 2D to a vector. These attributes together characterize an item and describe its content which can be compared to another item.

Recommendation systems use items to recommend them to customers (users). We will name the items as $I = (i_1, \dots, i_n)$.

1.2.2 Users

Users are visitors of a web site. As items, they can also be characterized with attributes such as their login name, e-mail, address or age. This information can be given explicitly (directly by user) when purchasing an item (delivery address, IP address, . . .), or implicitly (computed by recommendation system) like a preferred music style or presumed age.

As well as in items, these data can be used when recommending items to a user.

We will name users as $U = (u_1, \dots, u_m)$.

1.2.3 Interactions

One of the most important information a user can provide to any online service is her/his behaviour on a web site. We understand interaction as any user-item activity on a website. This kind of data is easily available for example from a database of purchases or server logs.

Interactions can be of various types, depending on a specific web site. They can be either implicit or explicit. Here are some examples of common interactions:

Detail views This is a case when a user views a web page with detailed information about an item, such as music performer profile, movie full description or a product detail page. This is a type of implicit interaction, where the dealer or service provider watches a user and gets her/his feedback without explicit questions.

Purchases If a customer finishes an order of an e-commerce web site and the deal is done, it is considered as a purchase. Typically it can be trousers bought on a web store, in case of online streaming services, it can be a fully watched movie or fully listened song without interruption. Together with detail views, this is a natural behaviour when using an online service and it also belongs to an implicit interaction.

Cart additions This interaction occurs when an item catches the user's attention enough to put it into a shopping basket, but still does not have to end up with a purchase. It is an implicit interaction.

Bookmarks Bookmarks are used for saving items that are somehow interesting for the user. They can be movies or articles that the user wants to come back to later or a product that is currently out of stock but wants to purchase it in the future. This is also considered as an implicit interaction.

Ratings Rating is a typical representation of an explicit interaction. A user provides a direct feedback to an item. Users can evaluate an item by stars (1 - 5) and thereby express her/his opinion on the item. In this case, 5 stars mean the best rating and 1 star stands for a negative experience with the item. Users' opinion can also be shown by Like and Dislike button (e. g. YouTube) or by a number, where the range from negative to positive values reflect users opinion.

Likes If Like button stands alone without its opposite, a Dislike button, we find them as implicit interactions. These interactions are characteristic of social networks.

Shares As well as Likes this is also an implicit interaction. By sharing (for example an article or a post on a social network), a user is expressing her/his interest in the particular item.

Commentary This specific kind of explicit interaction is non-structured unlike previous examples of interactions. User has the possibility to express her/his experience with the item in an open text form and for needs of a recommendation system it needs to be processed (e. g. sentiment analysis).

1.3 Scenarios

There are many places where recommendation systems can be used. In online web stores, correct placement can activate the visitors in a positive way and make them browse the shop even though they came with a determination to buy one specific product. That leads to more interactions on the web and therefore more information a recommendation system can use to make better, more aware predictions. The customer can explore more products that the store offers and may be interested in and also lead to better user experience with the online store as well as higher sales.

For a better understanding of the whole concept of product recommendation, we can have a look at real-life situations. Let's consider an ordinary brick and mortar clothes shop: a journey of a customer starts with a shop window - a retailer shows the "top sellers", seasonal clothes and new arrivals there. A shop window demonstrates what a store can offer to the customer and what style of clothes they sell. For a customer that has never visited the store, this can be a crucial moment - either the shop window attracts the customer's attention, or she/he passes by without entering the shop. If the offer corresponds with the idea of what the customer is looking for and she/he enters, there are various types of clothes divided by category on racks. Sometimes there are also big signs above each category for better orientation or mannequins dressed in fashionable outfits to display possible combinations of the offered pieces of clothes and accessories. Similar products are stored together, so, for example, one half of the store is dedicated to women's apparel and the other to the menswear. Denim trousers are concentrated in one corner, so the shop visitor does not have to go from rack to rack to look at all of them. There is also a shop assistant ready to help and answer all of the customer's questions. When the visit is successful and the customer chooses a new piece of clothes to extend her/his wardrobe, the journey ends at the cash desk. Usually, this is the last chance to catch a customer's eye. Sellers like to place accessories, such as sunglasses, purses, belts or hair bands, near the cash desk. Maybe they are not the reason the customer came to the store, but they may be cheap enough to add them to their cart. Then the customer pays and leaves the store. If she/he was feeling comfortable during the process and was satisfied with the purchase, it is possible that she/he may come back.

Analogically we can apply these principles to an online retail shop with clothes. A landing page substitutes a shop window, a product catalog with items divided into categories represents signs above the rack with clothes, mannequins can be replaced with articles about new fashion trends and accessories by the cash desk can be carried out as "Haven't you forgotten something?" panel with recommended items that are usually bought together.

As we can see, recommendation systems give the business an opportunity to guide the customers through the website just as in the brick and mortar. A customer can have an analogous personal shopping experience online and "off-line". We will introduce common scenarios and placement of recommendation systems in e-commerce in the following sections.

1.3.1 Landing page

As it was said before, landing page (or home page) is the shop window of a web store. Usually, it is the first contact with the store itself, where a customer can get the idea of products that the website offers. A customer can get easily inspired without needing to click on every category and product page. This is an opportunity for retailers to show the visitors various types of items that they may be interested in and persuade them to spend some time exploring their products. Recommendation systems can help to pick the right items for every user according to their history of interactions.

Users can be divided into two categories - first-time visitors and returning visitors. In case of first-time visitors, it is hard to guess what their tastes are. The recommendation system does not have any footprints/interactions, so it is unable to make any specific predictions of the customer's interest. In real life (offline) store, there are shop assistants that may guess the customer's style by her/his appearance or simply by asking what the customer would like

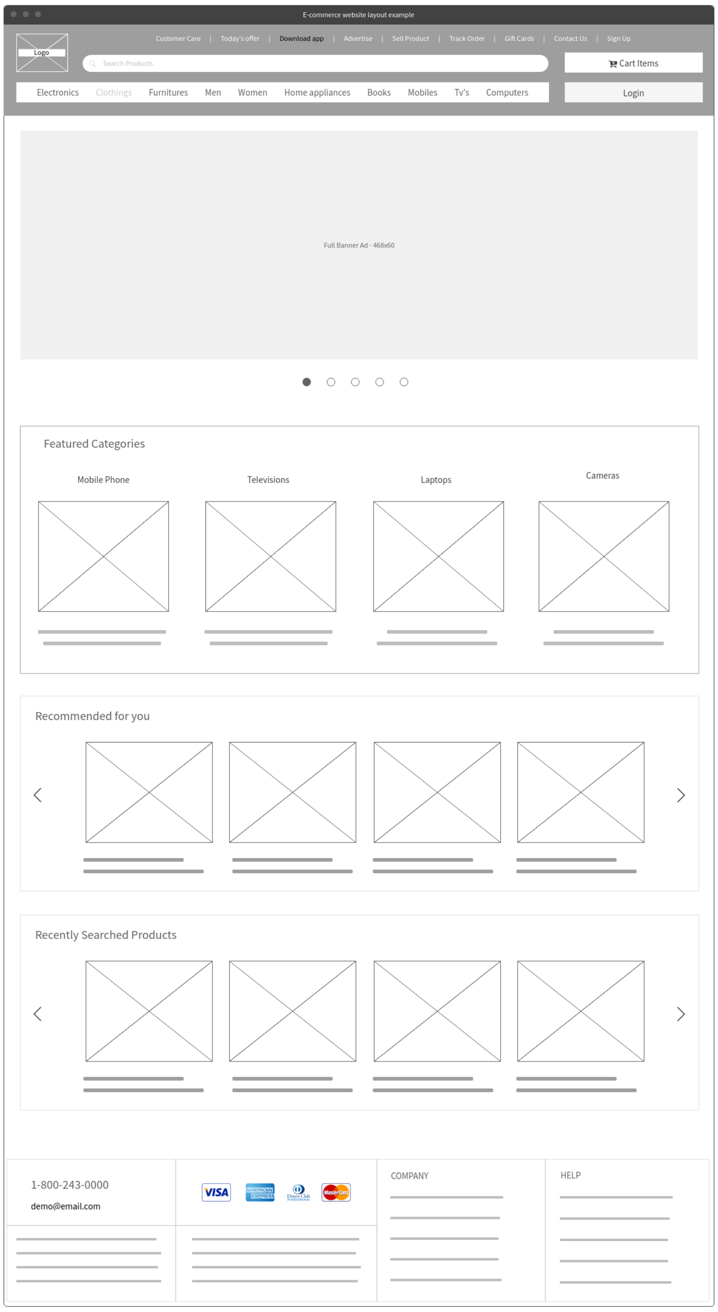

E-commerce website layout example
Logo
Customer Care | Today's offer | Download app | Advertise | Sell Product | Track Order | Gift Cards | Contact Us | Sign Up
Search Products
Cart Items
Electronics Clothings Furnitures Men Women Home appliances Books Mobiles Tv's Computers
Login
Full Banner Ad - 468x60
Featured Categories
Mobile Phone
Televisions
Laptops
Cameras
Recommended for you
Recently Searched Products
1-800-243-0000
demo@email.com
COMPANY
HELP

Figure 1.1: Example of a landing page layout

to buy. In an online store, it is usual to let the new customers browse the site for a while before making personalised recommendations. In the meantime, the recommendation engine can show the customer item-based offers, such as “trending” items, seasonal products or items on sale. This approach is based on the Pareto 80/20 principle, where 80 % of the income from sales comes from 20 % of all products.

When there are enough interactions made by the user, the recommendation system is able to provide aware, more personalized recommendations. Thanks to the known history of a customer, the recommender is able to be a competitive rival to a real shop assistant. Results of the recommendation can still focus on trending items or items on sale, but now they may be more based on categories the customer is interested in. For example, if a user looked at climbing ropes before and now there is a new model of climbing shoes, there is a high possibility this would attract her/his attention as these items come from the same category. On the other hand, if the customer already bought the same type of product, it is good to consider dropping it from the recommendations.

This approach can be applied not only to online stores but also to other e-commerce sites. Streaming services such as Netflix or Spotify can recommend recently most viewed movies or songs to the first-time visitors, but newly released movies from users’ favourite genre or a new album by their favourite music performer to the regular visitor.

Typically we can meet banners with items named “Recommended for you”, “Latest products”, “Bestsellers” or “Inspired by your browsing history” and so on.

Example of a landing page layout is shown in Figure 1.1, a practical example can be seen in Figure 1.3.

1.3.2 Product page

A product page is a place with detailed information about a specific product. Usually, it contains a text description, a price, an image of an item (or even more images) and sometimes reviews from the other customers or the average rating of the product.

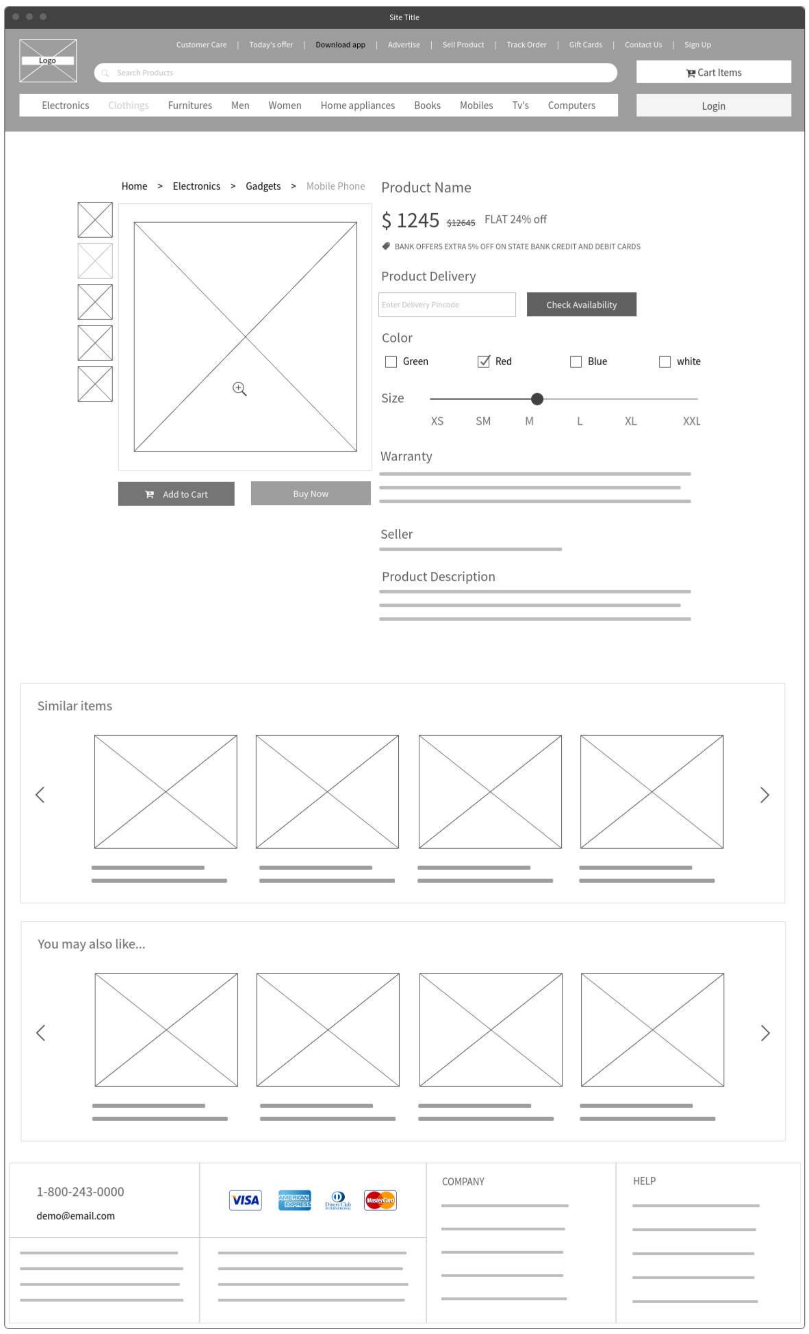

Site Title
Logo
Customer Care | Today's offer | Download app | Advertise | Sell Product | Track Order | Gift Cards | Contact Us | Sign Up
Search Products
Cart Items
Electronics Clothings Furnitures Men Women Home appliances Books Mobiles Tv's Computers
Login
Home > Electronics > Gadgets > Mobile Phone
Product Name
$ 1245 ~~$12645~~ FLAT 24% off
BANK OFFERS EXTRA 5% OFF ON STATE BANK CREDIT AND DEBIT CARDS
Product Delivery
Enter Delivery Pincode
Check Availability
Color
Green
Red
Blue
white
Size
XS SM M L XL XXL
Warranty
Seller
Product Description
Add to Cart
Buy Now
Similar items
You may also like...
1-800-243-0000
demo@email.com
VISA
AMERICAN EXPRESS
Diners Club
MasterCard
COMPANY
HELP

Figure 1.2: Example of a product page layout

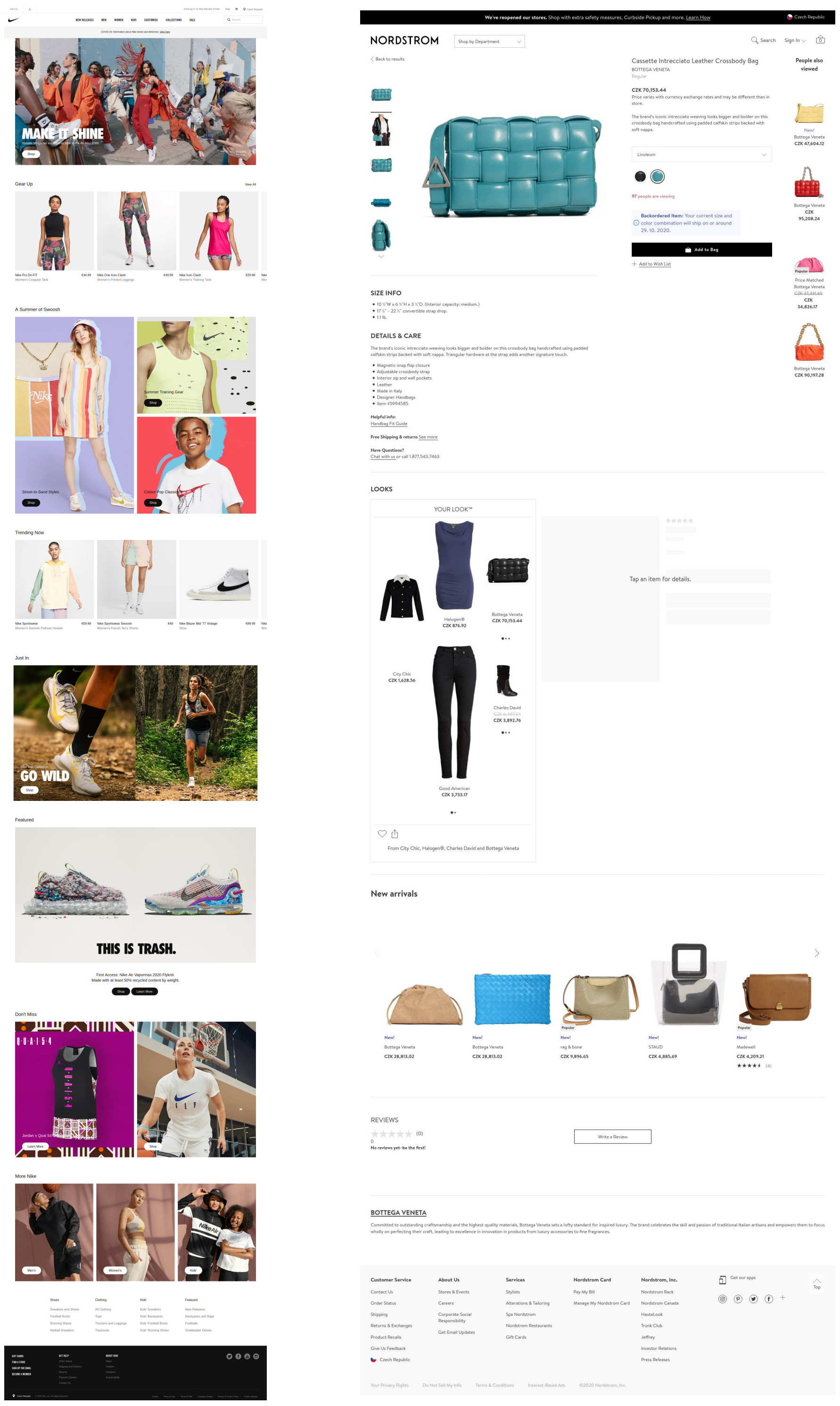


Figure 1.3: Practical examples of a landing page (left) and a product page (right)

This is a page which the customer clicks on when a product catches her/his eye. It is important to mention, that visitors may come to this page from non-direct traffic - online campaigns, advertisements, results of internet search engines, . . . The main task of a recommendation system here is to keep the customer's attention and display the most relevant items.

Customers are still in a search mode and showing them products that are somehow related can help them find the item they want and eventually buy. There can be various strategies used and common approach is to offer similar products or (and) complementary products. This is an analogy to a rack in a clothes shop - dresses are stored together so the customer can choose the style, colour and size that fits her/him. Next to the dresses, there are some sweaters that can complete the look. These strategies are called up-sell and cross-sell. In case of up-selling, the seller tries to maximize the profit by offering similarly priced or more expensive comparable product, while cross-selling is a technique where the seller tries to sell a different product, such as accessories or other additional items.

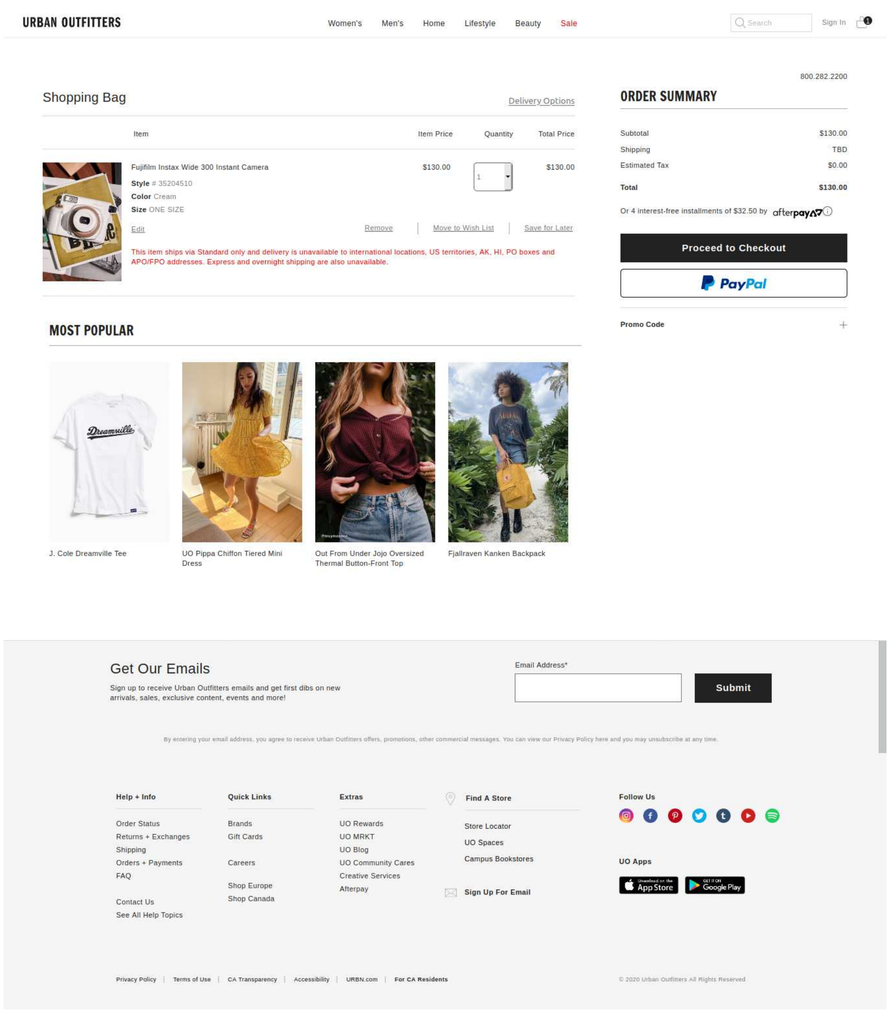
We can meet these recommendations in a group of items named for instance "You might also like. . . ", "Alternative products", "Customers who bought this item also bought. . . " or "Similar products".

Example of a product page layout is shown in Figure 1.2. Practical examples of two product pages shows Figure 1.3.

1.3.3 Cart page

A cart page, or checkout page, is situated at the end of the customer's shopping process. The customer is about to complete the purchase, confirm the order and then leave the shop. Just as in a brick and mortar a similar approach can be used in online retail.

As it was said before, sellers like to place small inexpensive items near the cash desk. They take advantage of the time while the customer is waiting in a queue to pay for the items she/he chose. The customer is in a shopping mood, ready to spend some money and so the sellers rely on this moment to attract customer's attention one more time. In a supermarket, we can find chewing gums or chocolate bars right above the conveyor belt or nail polish


URBAN OUTFITTERS

Women's Men's Home Lifestyle Beauty Sale

Search Sign In 1

Shopping Bag

Delivery Options

	Item	Item Price	Quantity	Total Price
	Fujifilm Instax Wide 300 Instant Camera **Style** # 35204510 **Color** Cream **Size** ONE SIZE	$130.00	1	$130.00

Edit Remove | Move to Wish List | Save for Later

This item ships via Standard only and delivery is unavailable to international locations, US territories, AK, HI, PO boxes and APO/FPO addresses. Express and overnight shipping are also unavailable.

MOST POPULAR

J. Cole Dreamville Tee

UO Pippa Chiffon Tiered Mini Dress

Out From Under Jojo Oversized Thermal Button-Front Top

Fjallraven Kanken Backpack

800.282.2200

ORDER SUMMARY

Subtotal	$130.00
Shipping	TBD
Estimated Tax	$0.00
Total	**$130.00**

Or 4 interest-free installments of $32.50 by afterpay

Proceed to Checkout

PayPal

Promo Code +

Get Our Emails

Sign up to receive Urban Outfitters emails and get first dibs on new arrivals, sales, exclusive content, events and more!

Email Address*

Submit

By entering your email address, you agree to receive Urban Outfitters offers, promotions, other commercial messages. You can view our Privacy Policy here and you may unsubscribe at any time.

Help + Info: Order Status, Returns + Exchanges, Shipping, Orders + Payments, FAQ, Contact Us, See All Help Topics

Quick Links: Brands, Gift Cards, Careers, Shop Europe, Shop Canada

Extras: UO Rewards, UO MRKT, UO Blog, UO Community Cares, Creative Services, Afterpay

Find A Store: Store Locator, UO Spaces, Campus Bookstores

Sign Up For Email

Follow Us

UO Apps

Download on the App Store

GET IT ON Google Play

Privacy Policy | Terms of Use | CA Transparency | Accessibility | URBN.com | For CA Residents

© 2020 Urban Outfitters All Rights Reserved

Figure 1.4: Example of a cart page

bottles and hairbands near the cash desk in a clothes shop. And that is where recommendation systems can also inspire.

The customer is often focused on their main purchase. Recommending related items will help the customer remember if she/he had not forgotten anything. When a visitor of a web store came to the checkout page, she/he already made other interactions on the site. There is at least one item in the cart and it is possible that she/he clicked on other items on her/his path through the web store, too. The recommendation system can take all these interactions into consideration and offer the customer additional items. It can be recommending trendy socks to Oxford shoes to complete a smart casual outfit or waterproof phone holder to a mountain bike. Another method is to offer the customer items she/he already saw, but have not bought yet.

Banners with recommended items on the checkout page can be named “You may also need”, “Haven’t you forgotten something?”, “Items usually bought together” and many different ways.

A practical example of a cart page is shown in Figure 1.4.

1.3.4 Category page

Another possible placement of recommendations is a category page. A category page associates products of the same type and purpose and sometimes divides them into subcategories.

As in case of a landing page, we can divide site visitors into two groups - first-time visitors and returning visitors. Problems with first-time visitors remain the same as well as the solution - the recommendation system can offer them the most favourite items from the category, hot deals, trending products and so on. To the returning visitors, a recommendation system is able to show more aware products and the approach can be inspired by the recommendations on a product page.

One of the dilemmas of recommendations on the category page is a conflict of exploration versus exploitation tactics. The first method enables the recommendation system to show items from different categories that are strongly related to the current category, whereas exploitation method suggests items strictly from the chosen category. A solution to this problem cannot be unambiguous as it strongly depends on the e-commerce sector, items the site is offering and type of customers.

Just like on the landing page, banners with item recommendations on the category page can be entitled as “Recommended for you”, “These items can match your style“ or “Customers who viewed this category viewed these related products too” - there are no conventions and naming the banners is totally up to the seller’s imagination.

Example of a category page layout is shown in Figure 1.5.

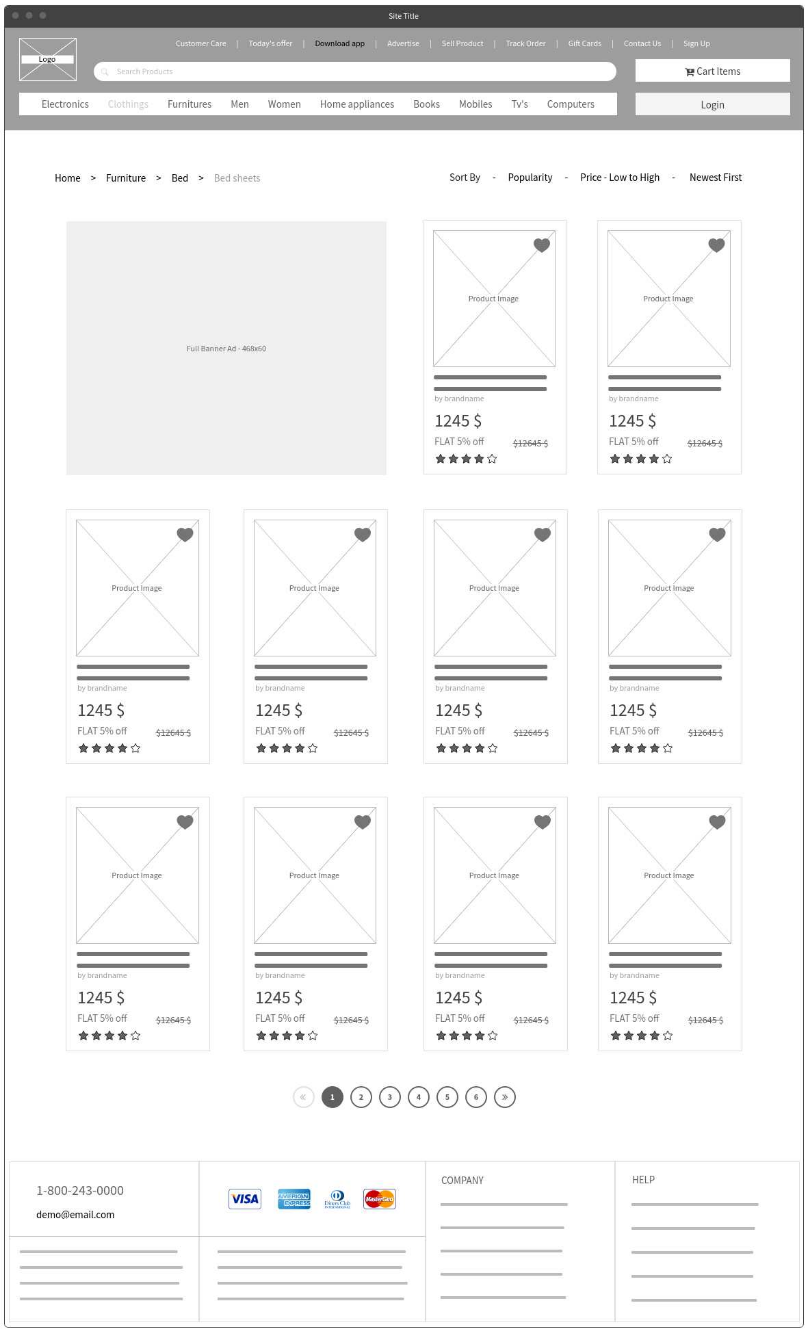


Figure 1.5: Example of a category page layout

1.3.5 Messaging

Messaging is another tactic for recommending products to customers. There are various ways, how to address the customer directly - for example via e-mail or Facebook Messenger. A personalized recommendation can keep the customer informed about the seller's offer and make them come back.

This is a scenario when the user already visited the seller's site, probably had a positive shopping experience and trusted the seller enough to give her/his e-mail address and conclusion to marketing mail. That also means that the recommendation system has already some interactions on the site to make aware recommendations.

There are also several recommendation strategies, which depend on the seller's intentions. The recommendation system can be inspired by the approach on the landing page and focus on the new products or items on sale, or the content of the e-mail can be more personalized and filter just the products which the customer is interested in. According to the previous behaviour and history of purchases, the recommender can again take into consideration seasonal products.

1.4 Recommendation techniques

Despite the fact that every scenario and dataset deserves an individual approach, we can find similarities in designing the recommendation systems. Depending on the data a recommendation system uses, we can divide the systems into three basic categories ([1], [2]):

- content-based recommendation systems,
- recommendation systems based on collaborative filtering,
- hybrid recommendation systems.

We will look at them closely in the next subsections.

1.4.1 Content-based filtering

Recommendation systems based on content-based filtering focus mainly on items and their description. Depending on the implementation they may also process a history of user's interactions and compare items with user interests. They use item attributes to compare likeness between each item and items that the particular user interacted with before. The system analyzes items description and transforms it into a form that allows comparing items and calculating their similarity.

One of the main difference between recommendation systems based on content-based filtering and collaborative filtering is that content-based filtering method does not take other users into account. It concentrates just on the specific user behaviour and does not make predictions based on the interests of other users. This approach allows the recommender to provide informed item offers without much knowledge about the user and recommend items with much less information about user interactions compared to the collaborative filtering method.

Content-based filtering, as the name would suggest, works with the content of items. Similarities of items are calculated based on their attributes and description. These recommendation systems use provided information about an item and process it into a structure, that is comparable with the other items. The content of an item is defined by its metadata, item description, the category of an item or an image. Usually, the item is defined by a vector of features, so all unstructured attributes like text description or multidimensional data such as images have to be transformed into comparable, typically one dimensional, feature space.

For example, an attribute can be discretized into binary vector - true and false values (or 1 and 0 respectively), if the item corresponds to the item or not (e. g. movie has "true"/1 value for comedy, fantasy, mystery and romance, but "false"/0 for horror, thriller and western). These attributes can be manually set or automatically extracted from a text description. In the second case, the system is processing the plain text to get keywords that define the item - firstly it removes all stop words such as determiners ("the", "a", "an", ...), prepositions (e. g. "before", "with", "to") and other commonly used words as "good", "want" or "be". The item is then defined by list or vector of words occurring in the description. Extracting information from an image is, on the other hand, much more complex task and will be mentioned in the next chapter. Vector of features can be additionally processed with

TF-IDF method [3]. This method puts weights to each “word” (feature) and determines its importance not only within the item but also within the whole dataset (all items).

When the item is defined by its vector of features, a metric is set so the similarity between the vectors can be calculated. We can meet with popular cosine similarity or euclidean distance.

Then there are several variants how to work with the processed vectors of item attributes. One of them is not taking the user into consideration at all and compare always two items only. These recommendation systems offer items similar to a specific item and can be used on a product page in “Similar items” banner. They retrieve items with similar attributes and can help the user explore items that are just slightly different from the currently viewed item.

Thanks to focusing only on the items, this technique can be used for first-time visitors just as for returning visitors as it does not depend on user’s history at all.

Another approach is to recommend items to the user according to her/his historical interactions and interests. The user profile is built on items she/he interacted with (viewed, rated, . . .) in the past and the recommendation system then offers items that are the most similar to the user profile. User is defined by the same vector of features like items. Values of each feature in this vector correspond to users’ interests, for instance, it can be an aggregation of vectors of items that the user interacted with. The same metric for similarity calculation like in the first approach can be used there and recommended items are the ones that are the closest to the user. This type of recommendations are personalized and reflect user’s interest. The main advantage is that it can be used already from one interaction of a user, so in case of a product page scenario, it can be deployed immediately (user already clicked on a detail of a product).

Content-based filtering method can be used in various situations. It is suitable for “Similar items” banner on a product page, “You may also like” banner on a landing page or even for e-mail newsletters, where the seller can point out on campaigns and products related to the customer’s preferences.

The biggest advantage of this method is the fact that (depending on the chosen variant and implementation) it needs very little information about the user. It works primarily with the attributions of items and can recommend relevant items even to first-time visitors. But this is also where its weakness lies - it needs metadata to every item. Usually, the item attributes are annotated manually which could be time consuming and expensive. Sometimes it is even difficult to describe an item, for example in case of a work of art. And that is what needs to be taken into consideration when using content-based recommendation systems - not every dataset and scenario is suitable for this method.

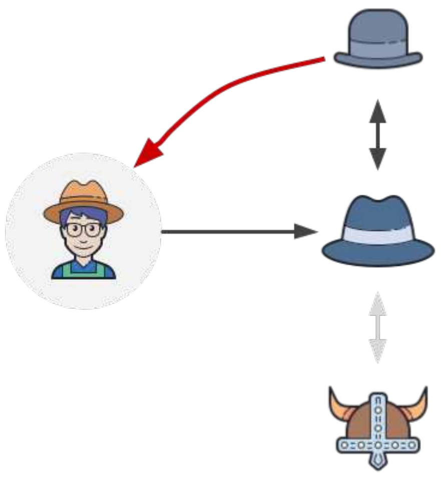


Figure 1.6: A content-based filtering diagram

1.4.2 Collaborative filtering

Recommendation systems that use collaborative filtering method are based on human behaviour. They assume that people trust other people from their social environment and listen to their recommendations. In other words, the system recommends products to customers according to a group of people with similar taste and interest (collaboration). This method analyzes interactions that users made in the past, tries to find patterns in their behaviour and groups them into communities which are defined by their preferences in items. The system then recommends items that the users might like in the future according to historical interactions of the rest of the users from the community.

Collaborative filtering technique, by its definition, is based just on interactions between users and items and does not use any information about the content of the recommended items. Despite that, it is still capable of recommending relevant items without understanding the attributes and features of items.

Collaborative filtering methods work with so-called rating matrix. In this matrix, the rows represent users and columns represent items, where values in the matrix correspond to the rating (or preference) of the specific item by the specific user. The recommendation system then tries to predict missing values in the rating matrix. There are several approaches to how this can be done and based on them, the recommendation systems can be divided into two groups - using memory-based (or neighbourhood) method or model-based method. They will be described in the next subsections.

1.4.2.1 Memory-based methods

Memory-based methods, or also neighbourhood methods, focus on relationships between users or items. They are making predictions about future interactions using past interactions between the examined object and subject. They use all the user-item relations stored in the database and retrieve the closest neighbours (a group of objects with similar interactions) to the current object, calculating the similarity with statistical methods. Based on the group of neighbours and their preferences, the system uses its own algorithm to choose top-N subjects to be recommended. According to the examined object (target of interest) of these methods, we can further divide them into two categories - user-based methods and item-based methods. User-based methods search for users with similar interests to recommend items that they liked and the current user has not discovered yet, whereas item-based methods search for similar items according to all user ratings.

1.4.2.1.1 User-based methods depend on the assumption that users that had similar interests in the past, will have similar interests in the future. As it was already said earlier, this theory comes from our natural behaviour when we prefer the same things or activities as our friends and family.

The goal is to recommend items to a user according to ratings of other users in the dataset. The system tries to find “friends” (neighbours) of the specific user, chooses the best possible items from their history that the user has not rated before, and recommends these items to the original user.

This method works with rows in the rating matrix, where every row represents one user. A user is then described by a vector of ratings, where values in the vector determinate the relationship to the items - they can be either known or unknown. The similarity between two users is calculated between these vectors, the metric depends on the implementation. The nearest neighbours are the users with the highest similarity measure. The recommendation system predicts rating to the unknown values in the user vector using ratings of the user’s neighbours and items with the highest rating are used for recommendation.

1.4.2.1.2 Item-based methods, as the name suggests, focus on items and do not work with a similarity of users, unlike the user-based method. They are based on a presumption that items are related when they are, for instance, bought together.

The objective is to recommend new items to a user that are similar to items she/he positively rated before. The system finds items similar to items the user rated in the past and estimates the item’s rating on evaluation of the nearest similar items. At the end, the system recommends items with the highest proposed rating that the user have not rated before.

A recommendation system with item-based method works with columns in the rating matrix. Just like users and rows in the rating matrix in a user-based method, columns represent items in the item-based method. An item is described by a vector of rating, where values in the vector determinate relationship to the users. The similarity between items can be calculated with the same metrics as in user-based method.

1.4.2.2 Model-based methods

Model-based methods make predictions based on a model created from the users’ interactions, such as rating matrix. The recommendation systems process the data to create a new structure (model) with compressed information

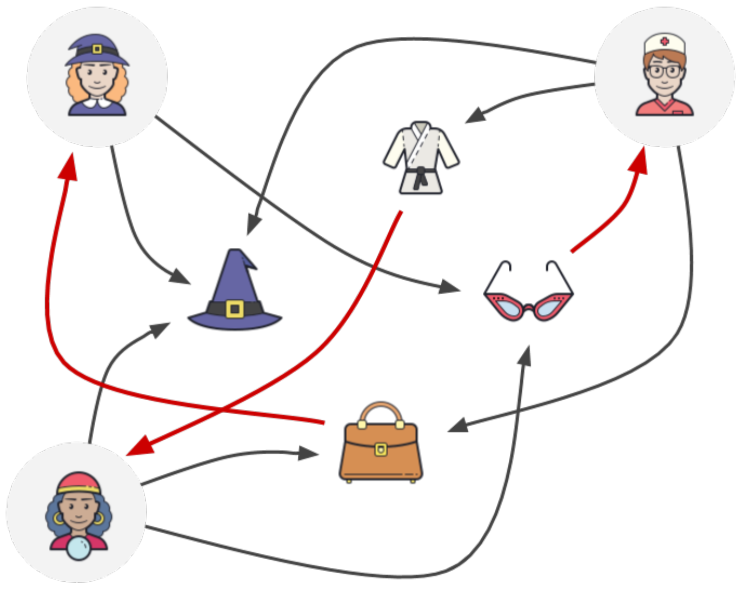


Figure 1.7: A collaborative filtering diagram

about the interactions. This part of a recommendation system is called training. During training, the system uses machine learning algorithms to find patterns in data so they could be converted into a more compact form and still provide aware predictions in collaborative filtering tasks. Algorithms such as Bayes models, clustering, matrix factorization or neural networks can be used for training. Some information is lost during this process.

Model-based methods are usually used on big datasets to speed up real-time recommendations because memory-based recommendation systems can be slow on real-life data with millions of users.

Collaborative filtering approach is fairly used in e-commerce. Thanks to the examination of online social behaviour, it can recommend relevant products to customers/visitors of these services. Every method introduced earlier in this section has its advantages and disadvantages, which should be thought of when designing a recommendation system.

It is not rare that collaborative filtering methods are sensitive to data. Large dataset, typical for online web stores, with millions of users and thousands of items, usually provide sparse user-item matrices. A typical problem caused by this phenomenon is the *cold start problem*. This occurs when a new user visits a site (or "first-time visitor", how was mentioned in section 1.3). Because collaborative filtering methods are based on interactions between users and items, new users need to make enough interactions to show their preferences so the system could provide reliable recommendations. A solution to this problem can be, for example, an alternative approach (mentioned earlier in 1.3) or explicit specification of the searched item by the user. Cold start problem also affects new items - as they have no interactions when added to the dataset, they will not be recommended. This problem does not relate to content-based filtering method, which requires mostly "just" metadata and does not have to depend on user-item interactions.

On the other hand, collaborative filtering methods can recommend much more diverse items than content-based methods, which tend to offer similar items, usually from the same category. This can help the customer (user) to explore a wider, more diverse selection of products (items) and get better oriented in the online store.

When using memory-based collaborative filtering, one of the advantages is the explainability of the results. They are also easy to scale (adding new users and items to the system) and implement. But with large scale also comes data sparsity and higher computational complexity, which negatively influence performance. This affects mainly the user-based method because of the large number of users. When using the item-based method, the similarities between items can be pre-calculated so the real-time recommendations can be faster.

That is also the reason why model-based approaches are often used - the model is computed offline so that the real-time performance improve by comparison with memory-based algorithms.

1.4.3 Hybrid filtering

Many modern recommendation systems do not use just pure collaborative filtering or content-based filtering approach. They combine these two techniques into one which results in a method called hybrid filtering.

Hybrid recommendation systems can be implemented in various ways, for example by chaining multiple models (linking one after another), combining outputs of multiple separated models into one common output by voting or creating a model that processes mixed types of data.

Depending on the implementation, hybrid recommendation systems can overcome weaknesses typical for content-based or collaborative filtering. They can solve little diversity of recommended items in case of content-based filtering, or sparsity and cold-start problem in collaborative filtering. Some of them are also capable of more accurate recommendations.

With this definition, design and implementation of a hybrid recommendation system are almost limitless.

1.5 Evaluation

As it was said before, recommendation systems help users to get oriented and guide them through the content on a web site. The objective is to recommend the users relevant items, keep them active, engaged and satisfied on the service they use. For instance, when somebody is interested in sports news, it is reasonable to suspect that she/he will be interested more in sports news rather than cultural ones, so recommending new articles from sport may convince the user to come back. On the other hand, the situation in retail can be different - recommending items that are similar to the items the user already interacted with can be boring and bring nothing new to the user. Recommending PC monitors to a user that purchased one a week ago will barely lead to any interactions even though these recommendations are relevant, but offering additional items such as gaming keyboard or mouse could be the items that will attract user's attention and click on them.

But how to measure if the recommendation system really works? How to measure the "satisfaction of a user", recommendation relevancy or if the recommended items are diverse enough? Evaluation of recommendation systems can help answer these questions and also confirm or disprove assumptions made when creating the system because human intuition can be misleading.

There are two approaches of evaluation of recommendation systems - offline or online ([4], [5], [6]). Both methods will be discussed in the next subsections.

1.5.1 Offline

Offline evaluation of recommendation system is performed on historical data. These data contain collected interactions made in past and are used for modelling as well as for the evaluation of the recommendation system. It is assumed that users will interact the same way (or close to it) in the future as they interacted on the site in the past. The behaviour of users is then simulated by using these data.

The dataset is usually split into two disjoint sets - train and test set, where the recommendation system is modelled on the train set and evaluated on the test set, or other methods like cross-validation is used.

Offline methods are a cheap way to evaluate a recommendation system. They allow comparing multiple algorithms without interaction with real users, along with choosing a suitable parameter setting of the final recommendation system for deployment and filtering out the improper approaches. On the other hand, offline evaluation is very limited and the interaction with a real user after deployment unknown. Everything is based on the assumption that user's behaviour will not change much and measuring the direct influence on the user is impossible. Therefore, evaluation based on historical data only is a non-trivial task.

Offline evaluation is often used in academic studies and researches. In business, it helps to choose top candidate algorithms for further testing, such as user studies or online experiments.

Here is an example of common metrics:

1.5.1.1 Predictive Accuracy Metrics

Predictive accuracy metrics focus on the ability of a recommendation system to predict missing values in a rating matrix. The system predicts values from a test set on the basis of a train set and compares the predicted value with actual value.

Mean Absolute Error (or MAE) is one of the typical rating prediction measures. It averages absolute difference between predicted rating calculated by the recommendation system and the actual rating of an item by a user. The equation is shown in 1.1, where N is a set of ratings from a test set, $\widehat{r}_{ui}$

is predicted rating of item i by user u and r_{ui} its actual rating.

$$MAE = \frac{1}{|N|} \sum_{\{u,i\} \in N} |\widehat{r}_{ui} - r_{ui}| \tag{1.1}$$

Root Mean Squared Error (RMSE) shown in 1.2 is an alternative to mean absolute error. In contrast with MAE, RMSE penalises higher errors more than MAE, which puts the same weight on each error.

$$RMSE = \sqrt{\frac{1}{|N|} \sum_{\{u,i\} \in N} (\widehat{r}_{ui} - r_{ui})^2} \tag{1.2}$$

1.5.1.2 Classification Accuracy Metrics

In real-life applications, it is common that there is no or very little information about users' explicit ratings, so the recommendation system works with implicit interactions. In that case, one of the typical approaches is to measure if the recommended items are relevant for the user or not. The system classifies items as relevant or irrelevant to the users, thus there are four possible scenarios that may occur (shown in Table 1.1):

1. True Positive - number of items, that are relevant for the user and the system recommended them,

2. False Positive - number of items, that are irrelevant for the user but the system recommended them,

3. False Negative - number of items, that are relevant for the user but the system did not recommend them,

4. True Negative - number of items, that are irrelevant for the user and the system did not recommend them.

The resulting table is called a confusion matrix and metrics that work with these values are often used in machine learning and information retrieval.

Precision is defined as a ratio of recommended relevant items to all the items that were predicted as relevant by the recommendation system.

$$Precision = \frac{TP}{TP + FP} \tag{1.3}$$

	Relevant	Not relevant
Recommended	True positive (TP)	False positive (FP)
Not recommended	False negative (FN)	True negative (TN)

Table 1.1: Confusion matrix

Recall, sometimes also referred to as sensitivity, is then a ratio of recommended relevant items to all truly relevant items.

$$Recall = \frac{TP}{TP + FN} \tag{1.4}$$

F$_1$-score is also known as F-score or F-measure. It considers both the precision and the recall and it is a harmonic mean of the two metrics.

$$F_1\text{-}score = \frac{2 * Precision * Recall}{Precision + Recall} = \frac{2 * TP}{2 * TP + FP + FN} \tag{1.5}$$

When calculating precision and recall, the items are classified into two classes - items, that are relevant according to the recommendation system and items that are irrelevant. Usually, the system decides this task on some fixed threshold (e. g. predicted user rating of a movie is more than 60 %, so the movie is considered relevant for the specific user).

Receiver operating characteristic curve (ROC curve) allows to graphically visualize the performance of a model varying the threshold. It is an alternative to precision and recall metrics and it uses the recall and false positive rate (FPR, 1.6) to plot a curve.

The area under the curve (AUC) is then another metric, which summarizes the ROC curve and may help to decide between multiple recommendation models when the dominant model is not obvious from the plot.

$$FPR = \frac{FP}{FP + TN} \tag{1.6}$$

1.5.1.3 Catalog coverage

Apart from accuracy, there are also other metrics that describe results of recommendation systems. One of them is catalog coverage.

As it was said before, the Pareto 80/20 principle is strong in e-commerce. It is not uncommon that most of the user interactions are related to a small part of items. Some systems then may provide recommendations that focus just on this subset of popular items. Recall of these systems is usually high, but they do not offer items that help the user explore new content and stay just with the “trending” items.

Depending on the business strategy and placing of the recommendation system, it may be considered to “sacrifice” accuracy at the expense of catalog coverage to provide recommendations of as many different items as possible (and exploit “long tail”).

Catalog coverage is calculated as a portion of items, that the algorithm is able to recommend. It evaluates the whole recommendation system. The equation is shown in 1.7, where U_{test} is a set of test users and $TopN(u)$ are the best N items that the recommendation system recommended to the user u.

$$catalog\ coverage = \frac{\bigcup_{u \in U_{test}} TopN(u)}{|I|} \tag{1.7}$$

Similarly, we can define user coverage.

1.5.1.4 Other

Just like catalog coverage in the previous subsection, there are far more metrics beyond accuracy for evaluating recommendation systems. Accuracy may be the primary metric to measure the system quality but in some specific datasets and scenarios, the recommendations could not be useful to the user even though they are relevant and accurate.

For example, let’s have a look at an online streaming service with movies and series - if a user already watched first 7 episodes of a particular series, there is a high probability that she/he will watch the next 13 episodes and even continue with the second season. By recommending all of the unwatched episodes from the series, we will get high accuracy, but from the user perspective, it might not be considered a good recommendation system. It brings the user nothing new, does not help her/him to discover unexplored content and offers the user items that she/he probably would watch anyway, without any recommendation. Also, the diversity of the recommended items is low.

In the next paragraphs, the most common metrics beyond accuracy will be briefly introduced - novelty, serendipity and diversity.

Novelty comes from the idea of recommending novel items that the user has not discovered yet. It is usually connected with accuracy and high novelty can mean low accuracy as the new items may not be relevant and useful to the user. Novelty has various metrics and definitions in the literature.

Serendipity, just like novelty, also does not have an unambiguous definition. Basically, it is a measure of how surprising the recommendations are for the user. The recommended items should be interesting and new to the user, the system should offer items that the user would hardly find and still would be a good and pleasant surprise for her/him. For instance, recommending a just-released album of a favourite music interpreter may be novel to the user, but barely surprising. Recommending of random items, on the other hand, may be very surprising, but not relevant. So serendipity can be defined as a representation of delightful surprise for the user.

Diversity, unlike the previous two metrics, is user-independent. It does not depend on information about a user and requires information just about items within a recommendation list. It can be defined as the opposite of similarity and calculated as a dissimilarity (distance) between the results of a recommendation. Diversity is useful in scenarios, where recommending similar items may not be helpful for the user. A typical example is holiday packages - offering hotels from the same resort may not be as interesting as offering a list of different resorts. By recommending different items, the system helps the user explore the offer quicker.

1.5.2 Online

In online evaluation, real users are engaged. It enables to show how users interact with the recommendation system when deployed and how it stands up real-time. Users are usually divided into two separate groups where each group interacts with a different recommendation system and interactions of users are studied.

This is called A/B testing. Users should be equally distributed between A and B group and then different algorithms are presented to the groups. Observation results of user behaviour are compared between these two proposed recommendation systems.

Online evaluation enables to measure if a new recommendation system is better than the currently deployed, how it really affects users, if the user behaviour had changed or if the assumptions based on offline evaluations are correct. So that results of A/B testing are trustworthy, there are some considerations to be made and what is going to be evaluated - if the influence of algorithm parameters are observed, the user interface should remain the same and vice versa - if, for instance, the number of recommended items is observed, it should be run on the same algorithm. Also, the group separation is crucial and it is good to run A/A test first - dividing users into two groups and observing their behaviour with the same setting for both groups. They should behave the same.

Online evaluation is easily interpretable and provides clear results if the recommendation system really works. On the other hand, it is costly, time-consuming and risky, if the new recommendation system is not tuned enough and provides irrelevant results. Users could lose interest or stop trusting the service completely. That is why it is important to run offline tests first and deploy a recommendation system after a detailed study of past user behaviour, especially in business.

1.5.2.1 Click-Through Rate

One of the most used metrics is the Click-Through Rate (CTR). CTR calculates the ratio between interacted recommended items and a number of recommended items. It helps to find out how many items are consumed out of the recommendations and also how effective the recommendation system is for the users in recommending relevant items (and therefore useful for users). The metric is shown in equation 1.8 where C_u indicates the “clicks”/number of interacted recommended items and R_u is the number of recommended items.

$$CTR = \frac{|C_u|}{|R_u|} \tag{1.8}$$

1.5.2.2 Conversion Rate

Conversion rate (CR) is an important business metric. It shows how successful the service is. This metric does not have to relate to sales only but can be any key performance indicator. Besides purchased items in online retail or fully watched movies in streaming services, it can be subscribing to newsletters, providing any personal information by the user or spending a specific amount of time on the web site.

Chapter 2

Image processing

The image processing chapter deals with detecting of image features, dividing image features into two basic categories (low-level and high-level features), explains the difference between them and introduces problems related with image features. In the end, it describes methods for extracting image embeddings.

2.1 Image features

In the digital world, an image can be stored as a matrix of values with multiple layers where every number determines the intensity of the specific pixel (value at a given position). Although this representation is suitable for displaying the image, the system has very little information about the content. There are no examined relations between the matrix values and without manual annotation, the system does not have any idea what the image depicts.

Image features have no exact definition and strongly depend on the given problem. Detection of them is a complex task that includes image processing whose objective is to find interesting parts of an image. These are further used in computer vision applications such as CBIR (content-based image retrieval), image classification, recognition, tracking or even photogrammetry. As the detection of image features is usually an initial step for these computer vision algorithms, it is greatly connected to the specific tasks and therefore there is a huge collection of image features in image processing.

The main property of a feature detector should be repeatability. The features are being detected on large datasets with different images, often with similar content (the same scene, the same type of products etc.) and it is desired to find these relationships between them. Just like in other cases, the overall algorithm will often be only as good as its feature detector (as the detected features are the basis for further computation), so the detector should be able to find the same feature in different images.

Features can be minor details and structures such as points, edges, curves or gradients and colour or even larger shapes and common objects. They can be divided into two groups according to the complexity of detection - low-level features and high-level semantic features [7]. Both will be introduced in next subsections.

2.1.1 Low-level features

Low-level features are visual descriptors that can be easily detected by automatic systems. They can be divided into three groups according to the resulting dimension of stored structure - numerical features (0D), histograms and descriptors (1D) and feature images (2D).

Numerical features are represented by real numbers. Their biggest advantage is direct usability or usability right after normalization without further processing. Here are some examples of numerical features:

- dimensions of the detected segment,
- dimensions of the bounding box,
- coefficients of Fourier transform,
- mean value (mode, median, standard deviation, . . .)
- entropy,
- Hu moments,
- etc.

Histograms and descriptors are features represented by a vector of values. Typical representatives include:

- histogram - frequency distribution of values in one layer of a matrix (image),
- SIFT (Scale-invariant feature transform) - descriptor of locally significant interest points; invariant to rotation, scaling and illumination changes; described by a vector of 128 B [8],
- SURF (Speeded up robust features) - based on SIFT, but it is faster and more robust against different image transformations; 64 B long vector [9],
- GLOH (Gradient Location and Orientation Histogram) - similar to SIFT descriptor,
- HOG (Histogram of oriented gradients) - focuses on the structure of an object, convenient for image recognition and object detection; it divides an image into portions and extracts gradients and orientation of edges for each; length of the descriptor depends on the size of an input image,
- and many others...

For descriptors like SIFT or SURF, it is typical that they describe local features. Usually, there are many of them found in one image and so they are further processed. One of the methods is called *Bag of Features*, where descriptors of all images from a test set are clustered into the required number of groups (these groups/clusters make a dictionary of the dataset) and then every image is represented by a histogram of detected descriptors divided into clusters according to the pre-computed dictionary.

Feature images can be seen as a function or mapping from image to image. Every pixel of the resulting image holds information about the wanted feature instead of intensity (as it was in the original image). The resulting image can be binary (pixels are boolean variables), where the value at every pixel determines whether the feature is present or not, or values can represent certainty measure of the feature existence. These kinds of features are often computed based on pixel neighbourhood, which makes them local. Typically, they are detected by using convolutional filters that are detecting features like dots, edges or texture. There are many image processing methods on how to detect desired features and it is common that they are combined and chained. Usually, images need to be preprocessed before the feature detecting (for instance with noise reduction, smoothing, . . .) and so it is not uncommon that feature images are used as an initial step for further feature detection.

Every method can be applied to the whole image or just its part. There are many textbooks that deal with this topic in more detail, for instance [10].

2.1.2 High-level semantic features

In the previous subsection, the low-level features were defined as minor details of an image, for example, lines, points, texture, shape, pixel gradient or colour, which can be captured by automatic systems. On the other hand, there are also other features that can describe the content of an image.

From a human perspective, an image can contain features like objects or actions. High-level semantic features are defined as features commonly used by human to describe images. They can be keywords or textual description of the image content, for instance, “Girl with a Pearl Earring” or “The Yellow House”.

High-level semantic features do not necessarily correlate with detected low-level features (e.g. grass does not have to be green in all images - depends on the variety, weather and light conditions and can vary from light yellow to dark green). This fact makes detecting them a non-trivial task. The absence of a direct link or bridge between low-level features and high-level semantics is known as the *semantic gap*. Automatic systems may not understand the relation between the pixels with many shades of green (low-level feature) and grass (human perception of the content).

The most accurate and the easiest way of detecting such features is manually, but it is also time-consuming and expensive, so with a big database of images, it is not an option. That is the reason why machine learning techniques are widely used for this task. There are many convolutional neural networks trained on large (earlier annotated) image datasets, which are able to detect common objects. These neural networks use both low-level and high-level features - in the first layers they learn filters for finding lines, dots, curves etc. and the later layers use these features to learn to recognize objects and shapes.

The list of methods for detecting high-level semantic features is longer than the presented options and depends on the application. Research dealing with this topic is quite widespread and some proposed solutions can be found in [11], [12] or [13].

2.2 Image embedding

Image representation is a problem that any image processing application deals with. As it was mentioned before, images are usually stored in two-dimensional arrays with multiple layers, where values represent the intensity of a pixel. Such representation may not be suitable for purposes of recommendation systems.

In real-life applications, especially e-commerce sector, we can assume, that images which appear in the datasets are either professional product photos or any other natural image taken by a camera. Images like that store a lot of redundant information, because neighbouring pixels are highly correlated with each other. In recommendation systems, the image features are being detected because of comparison - measuring the similarity between every two images. In this case, images are high-dimensional data that store a lot of unnecessary information that is not easy to process.

An embedding is a mapping of high-dimensional data to low-dimensional real vectors. It is a set of techniques used for compression of the input data so the redundant information is filtered out and similar items are close to each other. In practice, it means that a vector of a given number of values (n) should be able to represent every image so that similar items appear in the n-dimensional space close one to another. The similarity can be either visual or contextual (or both), according to the task. For instance, a dress and trousers can be considered similar when they are sewn from the same fabric with floral print, as well as two dresses can be considered similar even the fabric and cut are completely different.

Extraction of image features depends on the application. Every recommendation system requires different features and there is no versatile best practice. For example, choosing just high-level semantic features may affect a recommendation system in a way that it offers items just from the same product category, but will not help the user to discover items with the same style from a different category. This either may or may not be the desired behaviour of a recommendation system. Accuracy of the recommendation system also correlates with the chosen image embedding method.

Chapter 3

Related work

In this chapter a survey of recent related works is conducted. It studies the usage of the images in recommendation systems and related tasks, and divides the research work into three categories according to the used recommendation technique.

3.1 Content-based filtering

As it was mentioned in the section 1.4, a content-based filtering method focuses on items and their description. These recommendation systems are used when there is information about a product (an item is described by some features such as name, type, colour, manufacturer, etc.) and there is no (or very little) information about a customer. Users (customers of a web store) are being tracked and according to their behaviour, the system decides to which items the user pays attention and recommends items similar to user's interests or simply, the system does not consider user interactions at all and recommends items similar to the current item.

One of recommendation system that uses image representation is introduced in [14]. The authors designed an algorithm and tested it on 600 images of clothes and shoes from the Amazon web store and JD web store. First of all, they removed the image background by modified Local Conditional Flooding algorithm and then extracted low-level image features - colour from HSV histogram, texture defined by Tamura features [15] and Gabor filters, and shape from geometrical descriptors. Then the features were normalized and their weight computed. Because the speed of response of the system was in this work important, the authors used an indexing method for quick searching for similar images in the dataset.

As well as in the previous article, researchers in [16] used for image representation low-level features. Histogram of Oriented Gradient (HOG), Shape Context [17] and Hu Moments were integrated together with text information (product text description was compressed by Long short-term memory method) and weighted to emphasize or suppress extracted features according to their importance. Authors collected 5000 products from online stores and evaluated their proposed algorithm. They considered a recommendation to be correct if all from top-N recommended products were in the same category (e.g. glasses, shoes, watches etc.) as a query product. Cosine similarity was used for similarity calculation. They achieved 85 % of accuracy.

In article [18] a neural network classifier is used as a data-driven, visually-aware feature extractor. Batch-normalized Inception architecture was pre-trained on DeepFashion Attribute Prediction dataset ([19]) twice - for category and texture classification. Then it was applied on images from Fashion dataset, the output vectors from the pre-trained convolution neural networks were concatenated and used as image representations. For the top-N recommendations, the authors used a k-NN ranking algorithm.

The semantic gap between low-level and high-level features was tried to be solved by researchers in works [20] and [21]. They proposed image recommendation in vertical search based on ANOVA cosine similarity. They got image features (grey level co-occurrence matrix, Haralick features, Tamura features [15] and Gabor filters) and normalized them. For each term that appeared in the product text description, they computed ANOVA p-value by combining visual features and text-based search of the images and then used it as a weight of the term. Term dictionary of visual synonyms was then constructed according to the term similarity. For user query, they generated expended queries using the dictionary and the text-based search was performed. Images were recommended on cosine similarity score.

Another vertical image search is presented in [22]. The authors combine visual features with user relevance feedback. In the offline part of their proposed method, they extract five visual features for each image in a dataset -

colour moments, colour correlogram, texture, local binary pattern and edge detection. In the online (recommendation) part, they retrieve user feedback from search history with clicked and unclicked images. The relevance of the images from the sets is decided on cosine similarity of the images features vectors and then they are ranked and recommended to the customer. They tested the method on crawled image data from myntra.com, user relevance feedback was simulated with 100 participants. They evaluated their system manually and came to better accuracy of the relevance score than with the CBIR method.

The authors in the article [23] proposed a novel representation of images called Visual Part-based Object Representation. The main principle of the method is to decompose a product image into a set of disjoint parts and let users say which parts of the product they care about. According to their preferences, the products similar to the currently displayed product are recommended with an emphasis on the selected area. In practice, when a customer is looking at a motorbike helmet, which is divided into top, shield and visor and chooses that she/he likes the shield part, the system will favour helmets with similar shields in its recommendations. The authors extracted low-level features from the separated parts (such as HSV colour histogram and Bag-of-Visual-Word histogram) and from the whole product image to represent each item. The proposed method was tested on images of 5 categories from Amazon web store and the authors came to the conclusion that it can achieve better performance than some text-based methods.

Researchers in [24] were focused on designing a user interface to browse fashion products comfortably. They presented a terminal for usage in retail stores where on the touch screen upper and lower parts of apparel (which are sold in the store) are shown. In the middle of the screen, there is a currently selected outfit surrounded by clothes - similar pieces recommended by the store. The further from the selected outfit, the dissimilar products are recommended as an inspiration. Recommending works for upper and lower apparel separately. By clicking on another piece of clothes, the recommendations are recalculated and the screen is changed. For the product representation, they used image preprocessing (removing image background by use of Canny edge detection and dilatation) and then they extracted low-level image features such as correlated colour temperature, colour brightness and colour hue.

3.2 Collaborative filtering

Collaborative filtering approach is based on the assumption that there are groups of people with similar tastes and interests. The recommendation system makes predictions upon analysis of the relationships between a customer and items (the user's interactions) and the similarity of users. This method is widely used because it can achieve good accuracy without a need to understand the content of the items.

Article [25] managed to use collaborative filtering method together with image features. A rating prediction model was created where similar users helped to predict current user ratings. Then for all items, that meet a condition with a rating threshold, image features are extracted. Authors opted for Sparse filtering method to compute eigenvectors of a product picture. They calculated Euclidean distance between these items and already seen items of a user and the most similar products were recommended. They tested the proposed method on Amazon product dataset.

Another approach of combination collaborative filtering and image features can be seen in [26]. The basis of the predictor is matrix factorization which is combined with image features extracted by AlexNet convolutional neural network. They also took into consideration users' behaviour over time and created a time-aware visual predictor. The model is then learned using Bayesian Personalized Ranking method.

The same approach was chosen by researchers in their work [27].

In the article [28] the researchers designed an image recommendation algorithm using feature-based collaborative filtering. Unlike in traditional collaborative filtering method, where a user-item matrix is used, in this work, a user is represented by visual feature space. First of all, every image of a small dataset is segmented into several regions and visual features are computed for every region. For this purpose, they are described by low-level features such as colour, texture and shape. Then the images that are purchased by the same user are clustered according to their features. A user is defined as a set of these clusters. Items that are recommended to a user depend on k-nearest neighbours that are calculated by inter-cluster distance.

3.3 Hybrid filtering

Combining content-based filtering and collaborative filtering methods results in hybrid filtering. This kind of technique can overcome problems of the previous two approaches and lead to better results. Merging can be done in various ways like creating two different models and combining the results at the end or by "spicing" one of the methods by adding some tactics of another.

Researches in [29] proposed a hierarchical user interest mining method, which is based on user-contributed photos in her/his social media sites. They assume that photos shared by users on a web page have the same topic and use their textual description with the photos as an input to their model. They map user's information to hierarchical topic space, ODP (open directory project), which is a manually edited ontology directory. The item representation is mapped the same way. The user interest is calculated by the TF-IDF method and the relevance between a user and an item is measured by cosine similarity. This recommendation based on ODP is similar to collaborative filtering.

In [30] the recommendation system was divided into two stages - in the first stage, the system recommends products based on view-also-view model (classic collaborative filtering) and then the customer manually highlights the regions of suggested items and the system re-ranks recommendations according to image features of the regions. The images are represented by concatenated low-level features such as the colour and texture. Colour is extracted with the usage of HSV colour space and separated into 30 colour bins. The texture is described by LBP (local binary patterns) and SIFT features. Image similarity is calculated with Kullback-Leibler divergence.

As well as the previous article, researchers in [31] focused on fashion. They designed a recommendation system capable of matching clothes and accessories together. The dataset they used is recorded from the Amazon web store. They downloaded images of products and recommendation shown there as relationships between them. For every image, they calculated its feature vector using a convolutional neural network - Caffe deep learning framework which is pre-trained on ImageNet. Then they learn a parameterized distance transform where the distance for objects that are related is smaller than to those that are not. Then the recommendations are made according to the category of a product and type of relation. They created a system that can recommend complementary products and one of the conclusion they came to was that James May was more fashionable than Richard Hammond.

Researchers in [32] also used convolutional neural networks to represent items and users. They designed a dual-net deep network, which takes triples as input - information about a customer and about two products. The CDL (comparative deep learning) architecture has three sub-networks - two are identical convolutional neural networks and extract features from input images, one is a full-connection neural network that captures the user's information. CNNs extracting visual information of items are inspired by AlexNet and produce a 1024-dim vector representing input image. The output of the third sub-network is also a 1024-dim vector, but as an input, it takes user vector - firstly all possible tags from items are converted by word2vector method [33] into vectors, then these vectors are clustered and finally, users are described as bag-of-words applied on these clusters. Relative distances between the inputs are the objective of learning.

Article [34] focused on recommendations based on LSH (Locality Sensitive Hashing) algorithm. SIFT features are extracted from low-resolution images, then the descriptors are converted into eigenvectors and saved to the hashed table. The recommendation system takes an image as an input, returns target pictures (images from the same hash bucket) and with help from collaborative filtering recommends products to the user.

In [35] the authors proposed a hybrid recommendation system that uses aesthetic features of clothing. They trained an aesthetic neural network with AVA dataset [36], where they removed some pathways which have little connection to clothing and added low-level features of the images. The output of the second fully-connected layer was used as aesthetic features. During recommendation, both aesthetic features and CNN features (the second fully-connected layer of Caffe model trained on ImageNet dataset) contributed to the final prediction based on dynamic collaborative filtering. In the end, there was a tensor factorization model capable of better results than a model that used just CNN features.

Another recommendation system that combines CNN image features and collaborative filtering can be found in [37]. The researchers used the CNN model VGG19 pre-trained on ImageNet and performed classification of the products with the help of extracted image feature vectors. Then they used the same features to recommend products with respect to customer's preferences, her/his purchase history, rating of products and diversity of the product categories.

Also, authors of [38] used VGG19 neural network to extract high-level as well as low-level features. Then they found correlations between them with the usage of a convolutional neural network to express style features. All of these features were incorporated into a collaborative learning system based on Bayesian Personalized Ranking, which uses implicit feedback from users and predicts their interest.

Chapter **4**

Analysis and design

Analysis and design chapter introduces the architecture of a common recommendation system, analyses the provided dataset, studies user behaviour and examines product images. Based on the observations and previous research, it proposes several recommendation algorithms.

4.1 Recommender architecture

Recommendation systems are usually a part of a complex online service system. The common usage with basic data flows is shown in Figure 4.1. Users communicate with user interface (web site), which logs information about them and their interactions into a database.

Database store all the information about items, users and their interactions in the approximate form presented in section 1.2. Recommendation system then receives queries from the user interface (such as “recommend items to this user in this scenario”) takes data from the database, computes the recommendations and send them back to the user interface, which displays them to the user.

The recommendation system has to process many tasks before it is able to recommend items to a user. Here are the essential ones:

- extraction of item features (e.g. image processing of the product photos),
- data preprocessing for modelling a recommender,
- calculation of a model for a specific scenario,

- item recommendation for a specific case (user-scenario),
- postprocessing of the recommendation (such as storing the recommendation event to the database and/or evaluating of the model).

The aim of this thesis is to design a recommendation system in an offline environment, that is able to extract item features, create a model with the manually cleansed dataset and evaluate the model on the test offline data.

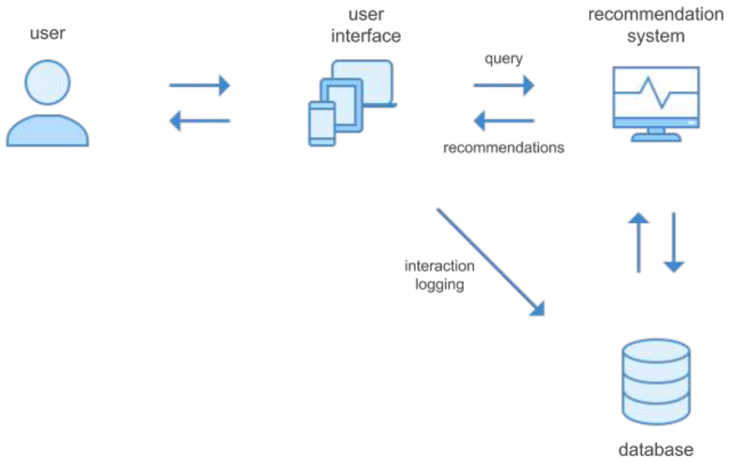


Figure 4.1: Common usage of a recommendation system

4.2 Dataset

For purposes of this master thesis, an anonymous database dump was provided by Recombee. The data comes from an online retailer of furniture and homewares.

4.2.1 Model

The database dump contains user interactions from over a year. The whole dump is over 25 GB in size, the extracted database takes up around 182 GB of disk space. The main structure of the database is composed of 6 tables (the entity-relationship diagram is depicted in Figure 4.2). There are records of 15 000 126 users, 85 444 items and 4 types of interactions:

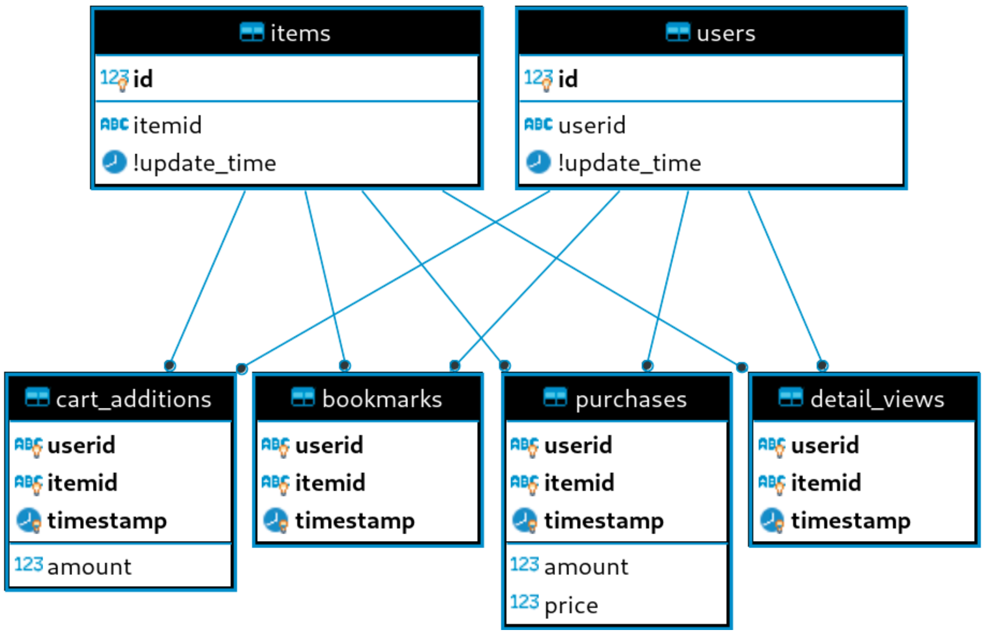


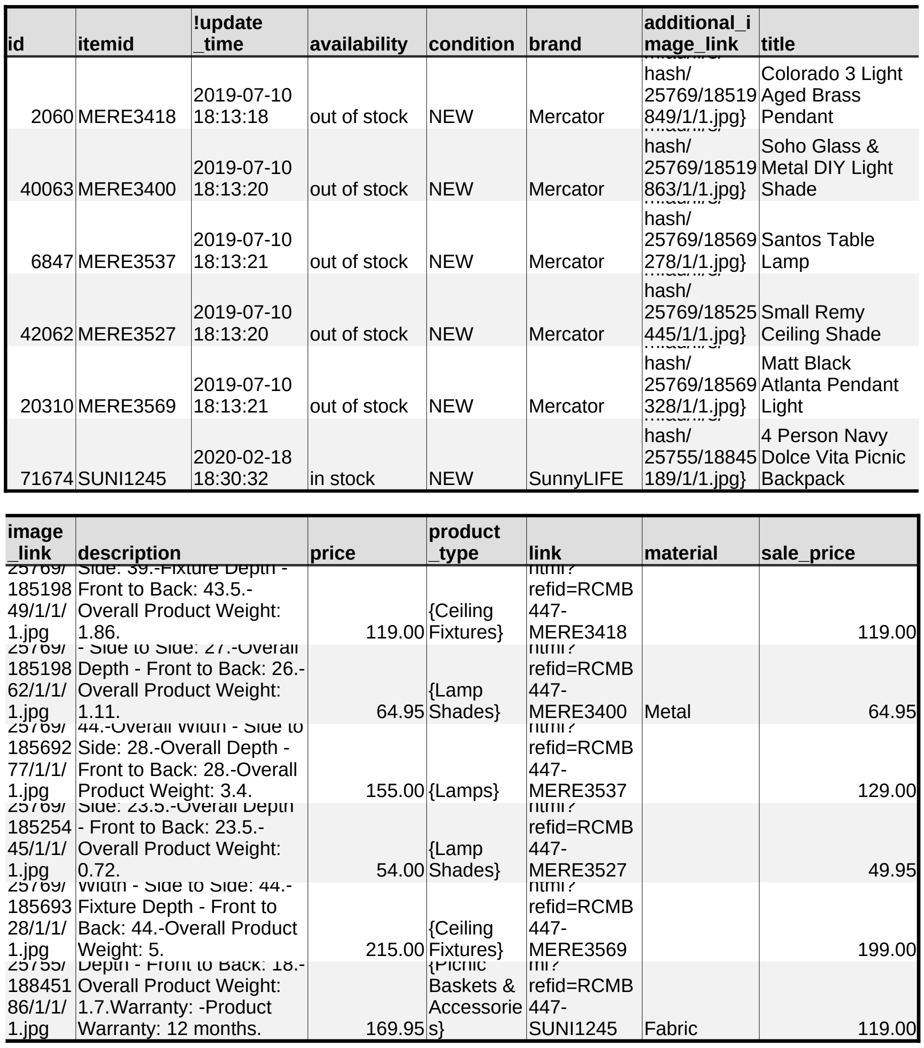
Figure 4.2: ER diagram of used database

1. detail_views (73 609 416 rows)
2. bookmarks (1 136 211 rows)
3. cart_additions (1 368 063 rows)
4. purchases (668 007 rows)

The user table contains just "id" (internal primary key), "userid" and time-stamp with "!update_time". Information about items is wider - there are 42 columns in the table. Most of the columns are either technical or do not contain data at all. The example of the content of selected item parameters is shown in Figure 4.3. The column "description" is an unstructured data - a plain text without a clear and uniform structure.

4.2.2 Images

Out of 85 444 records in table items, there are 80 288 non-NULL values in image_link column. They contain hyperlinks to item images - main pictures that represent the product when there is just one preview image for one item



id	itemid	!update _time	availability	condition	brand	additional_i mage_link	title
2060	MERE3418	2019-07-10 18:13:18	out of stock	NEW	Mercator	hash/ 25769/18519 849/1/1.jpg}	Colorado 3 Light Aged Brass Pendant
40063	MERE3400	2019-07-10 18:13:20	out of stock	NEW	Mercator	hash/ 25769/18519 863/1/1.jpg}	Soho Glass & Metal DIY Light Shade
6847	MERE3537	2019-07-10 18:13:21	out of stock	NEW	Mercator	hash/ 25769/18569 278/1/1.jpg}	Santos Table Lamp
42062	MERE3527	2019-07-10 18:13:20	out of stock	NEW	Mercator	hash/ 25769/18525 445/1/1.jpg}	Small Remy Ceiling Shade
20310	MERE3569	2019-07-10 18:13:21	out of stock	NEW	Mercator	hash/ 25769/18569 328/1/1.jpg}	Matt Black Atlanta Pendant Light
71674	SUNI1245	2020-02-18 18:30:32	in stock	NEW	SunnyLIFE	hash/ 25755/18845 189/1/1.jpg}	4 Person Navy Dolce Vita Picnic Backpack

image _link	description	price	product _type	link	material	sale_price
25769/ 185198 49/1/1/ 1.jpg	Side: 39.-Fixture Depth - Front to Back: 43.5.- Overall Product Weight: 1.86.	119.00	{Ceiling Fixtures}	html? refid=RCMB 447- MERE3418		119.00
25769/ 185198 62/1/1/ 1.jpg	- Side to Side: 27.-Overall Depth - Front to Back: 26.- Overall Product Weight: 1.11.	64.95	{Lamp Shades}	html? refid=RCMB 447- MERE3400	Metal	64.95
25769/ 185692 77/1/1/ 1.jpg	44.-Overall Width - Side to Side: 28.-Overall Depth - Front to Back: 28.-Overall Product Weight: 3.4.	155.00	{Lamps}	html? refid=RCMB 447- MERE3537		129.00
25769/ 185254 45/1/1/ 1.jpg	Side: 23.5.-Overall Depth - Front to Back: 23.5.- Overall Product Weight: 0.72.	54.00	{Lamp Shades}	html? refid=RCMB 447- MERE3527		49.95
25769/ 185693 28/1/1/ 1.jpg	Width - Side to Side: 44.- Fixture Depth - Front to Back: 44.-Overall Product Weight: 5.	215.00	{Ceiling Fixtures}	html? refid=RCMB 447- MERE3569		199.00
25755/ 188451 86/1/1/ 1.jpg	Depth - Front to Back: 18.- Overall Product Weight: 1.7.Warranty: -Product Warranty: 12 months.	169.95	{Picnic Baskets & Accessorie s}	ml? refid=RCMB 447- SUNI1245	Fabric	119.00

Figure 4.3: Example of a database table items

(in case of scenarios like landing or category page). Every image is a professional product photograph of dimensions 500 × 500 pixels. A big part of the photographs depicts products on the white background, in some cases, there are photos of scenes like a bedroom (typical for bed sheets or wallpapers). Examples of these images can be seen in Figure 4.4. The size of all downloaded images takes up 4,1 GB.

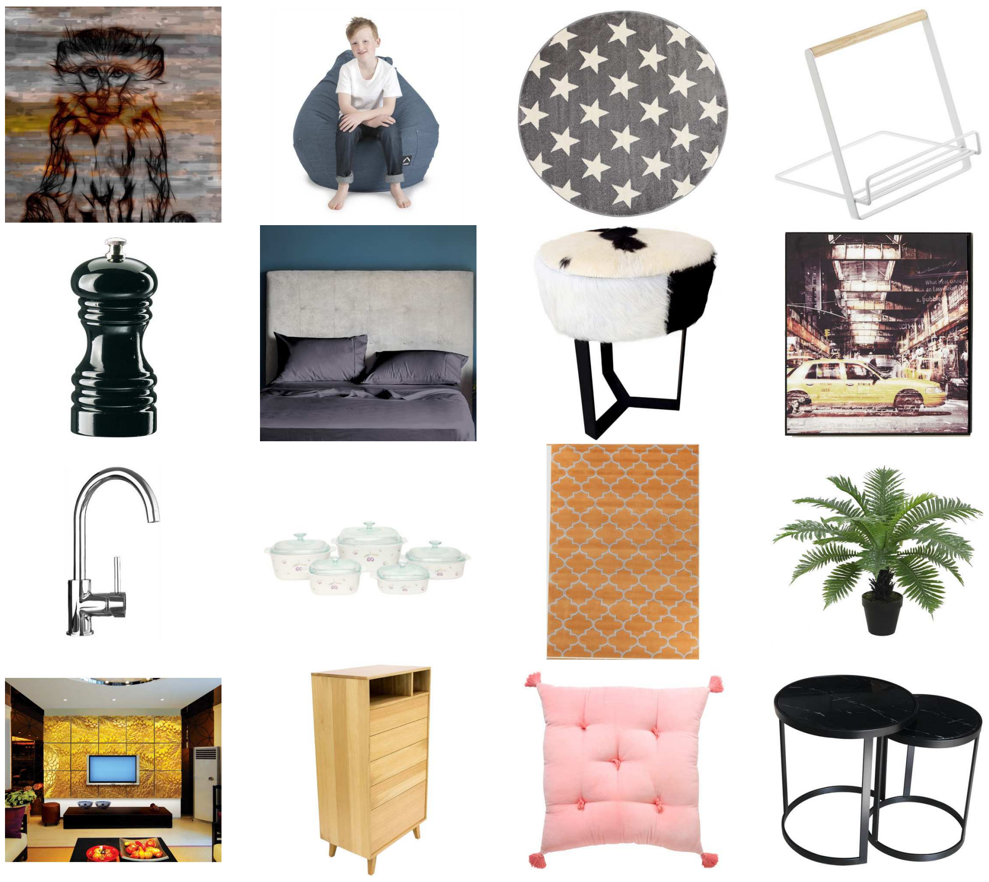


Figure 4.4: Examples of product images

4.2.3 User behaviour

As it was mentioned before, a customer path through a web store is similar to a brick and mortar shop. A user comes to the site, looks at the offer, chooses the products that attract her/his attention, adds it to the cart and buys it.

This behaviour is tracked down by the user interactions on the web site - a detail_views table represents the clicks on the items (a user visited the product page), bookmarks store items that the user found interesting enough to label them (this interaction usually occurs when the user is still searching, before putting the item to the cart), a cart_additions table represent items put to the shopping cart before the purchase itself and the purchases table stores items bought by the user.

Graphs in Figure 4.5 show top 25 favourite categories (out of 214) among

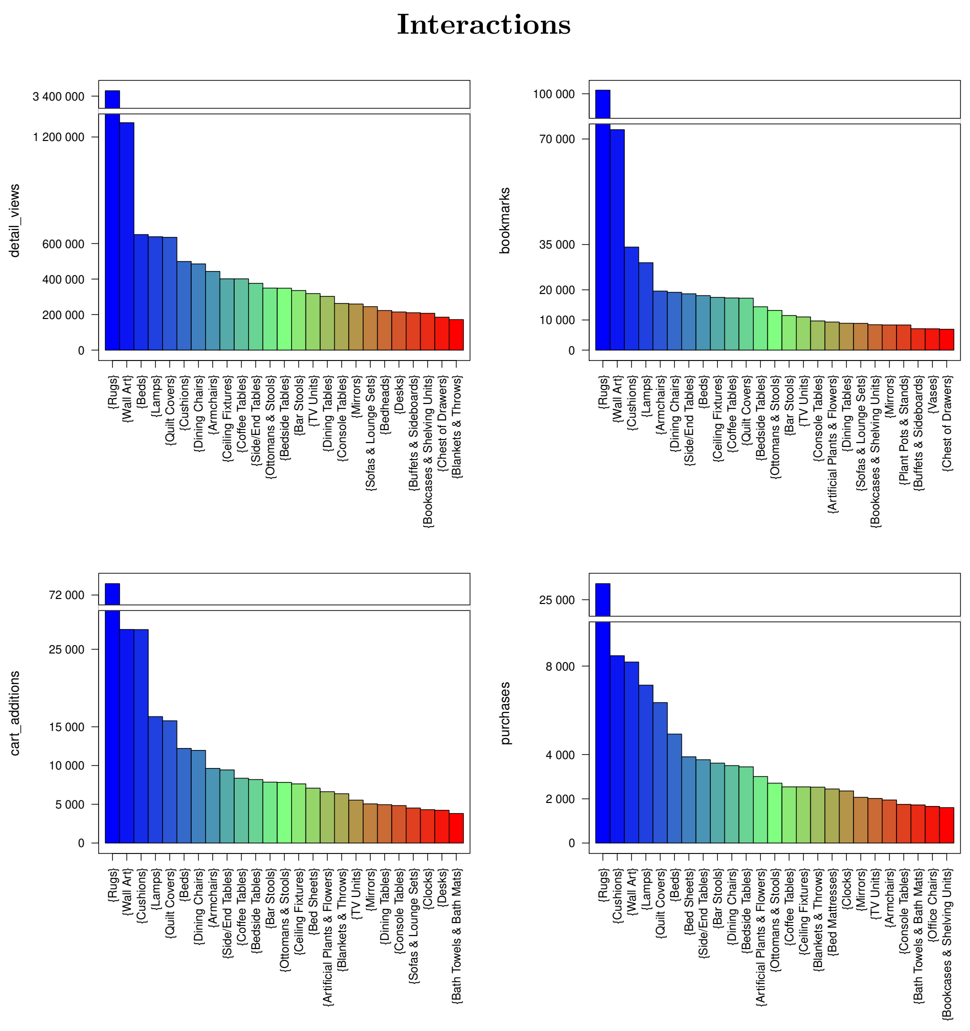


Figure 4.5: Count of interactions according to product category

all four types of interactions. The graph 4.6 visualizes their change of rank during the shopping process. Even though the changes may seem dramatic, it should be considered that there are around 3-4 most favourite categories that do not change their position much. The situation with other categories can be caused by the small differences in the count of interactions (as it is visible in graphs 4.5).

Another point of view is shown in graphs 4.7 and 4.8. 4.7 displays the dependency of the price of an item on user interaction. It is clear that customers of this web store tend to buy cheaper items and look at more expensive prod-

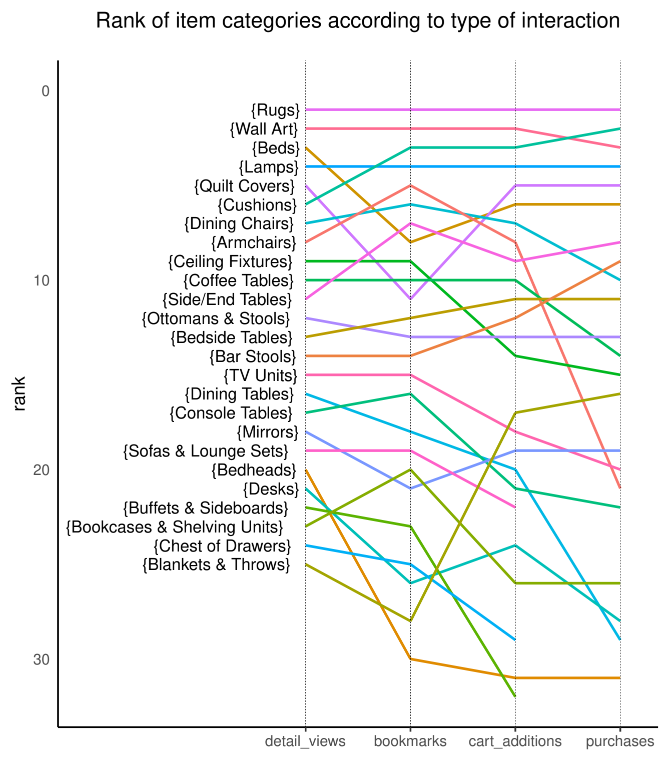


Figure 4.6: Rank change of interactions during shopping process

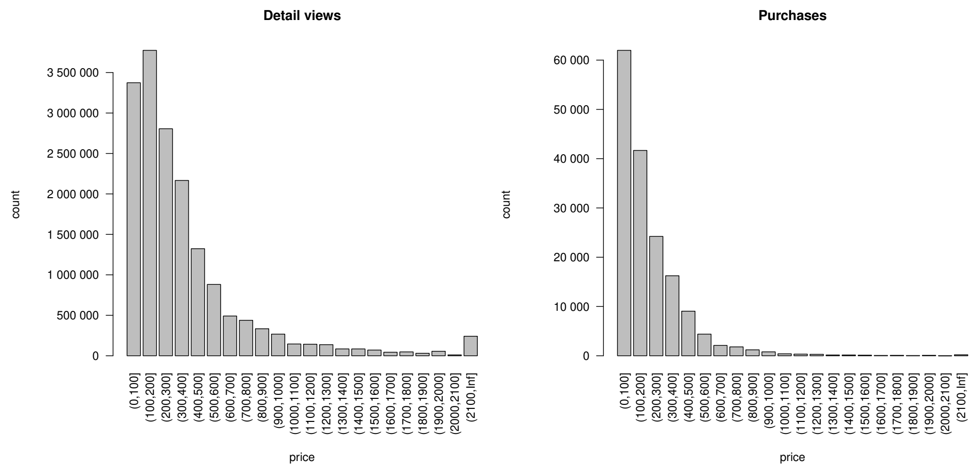


Figure 4.7: Interaction of users according to item price

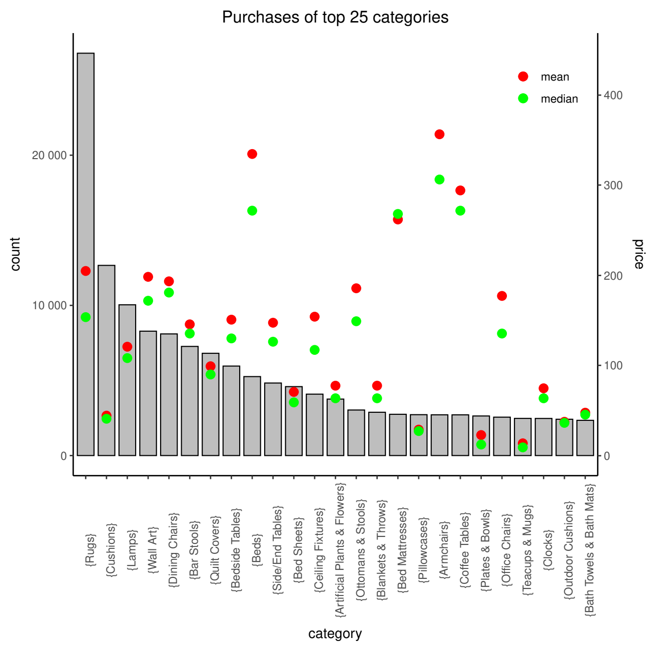


Figure 4.8: Top 25 categories that appeared in individual purchases - comparison between count of occurrences and mean/median price of items within category

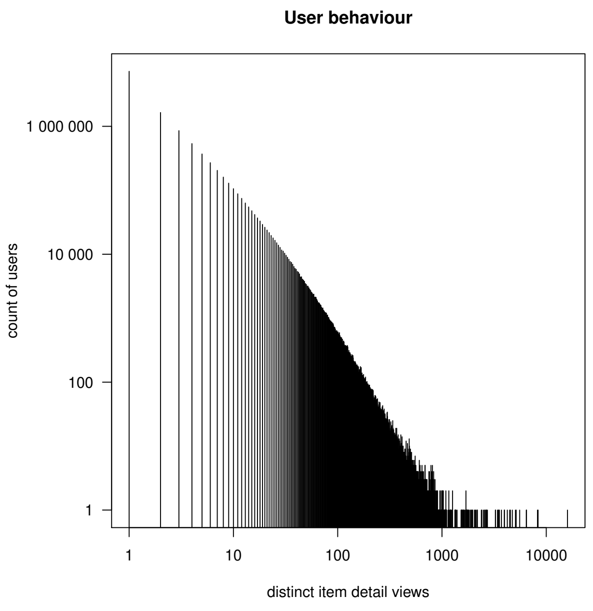


Figure 4.9: Count of users in dependence on count of viewed distinct items

ucts than they actually buy. The graph 4.8 analyses the relationship between most favourite categories and the mean and median price of items belonging to that category. The assumption that the top categories gain their popularity based on the offer and inexpensive products within the category, did not confirm as there is no visible connection between these two measures.

The last graph (4.9) illustrates user engagement with the web site. It shows the number of users in dependence on the number of distinct items they viewed (how many detail_views interactions they made). Both of the graph axes are logarithmic. Most of the users make very little interactions (from 1 to 10) and then leave. On the other side of this range, there are units of users that viewed a big part of the whole store offer.

The influence of user behaviour on the recommendation model will be discussed in the following subsection.

4.2.4 Selecting test data

The original dataset provides over 73 millions of detail_view interactions (visits on product pages) of more than 12 million users. This type of interaction has the most records in the dataset, so it served as the main criterion for user selection. This user behaviour was already presented in the previous subsection in Figure 4.9.

It was found out that most of the users do not interact on the site that much which leads to the sparsity problem. This makes training a recommendation model, which would give satisfactory results, a hard task. To overcome this issue, users that have more than 50 interactions with distinct items were selected.

On the other side from inactive users, there are individuals that have a suspiciously large number of interactions. Such users may be robots crawling on the site. This behaviour could also affect the model results, so users that have 800 and more interactions were filtered out.

Out of more than 12 million users, around 100 thousand users were selected for further processing. The behaviour of these users is shown in the graph 4.10.

From this sample of users, 10 % was selected for modelling recommendation systems (with the usage of random selection with uniform distribution). These

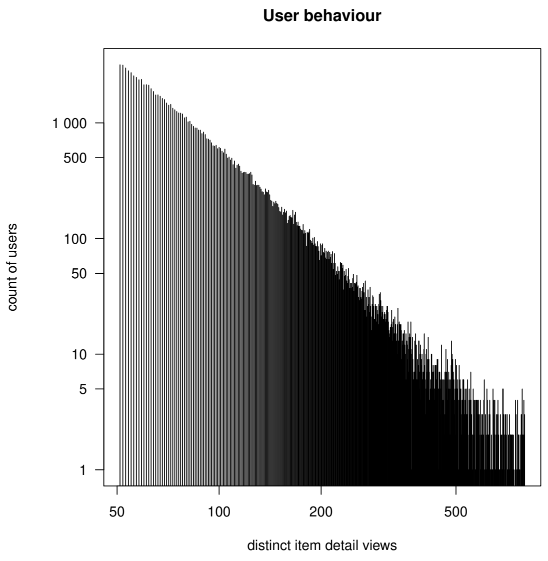


Figure 4.10: Count of users in dependence on count of viewed distinct items

were then divided into train and test sets with ratio 75/25, which resulted in approximately 7500 users in the train set and 2500 users in the test set.

Users in the test set were modified because of evaluation. Every item of a test user was filtered out with a 50 % probability (uniform distribution of random function). This means that approximately half of the original interactions of test users was eliminated.

4.3 Image embeddings

Based on the previous research, two types of image features were extracted from 80 228 item images - low-level and high-level.

As low-level features, colour histograms were chosen to represent the image. The choice is based on the assumption that colour is important to people when buying houseware and home decorations. According to the personal experience of the author of this thesis, a customer usually tends to match furniture and other home details on visual appearance, especially colour. Two types of colour histograms were extracted from the images, RGB and HSV.

RGB histograms store 256 integer values for every colour layer, so the resulting vector is 768 B long. Images in RGB feature space are visualized by

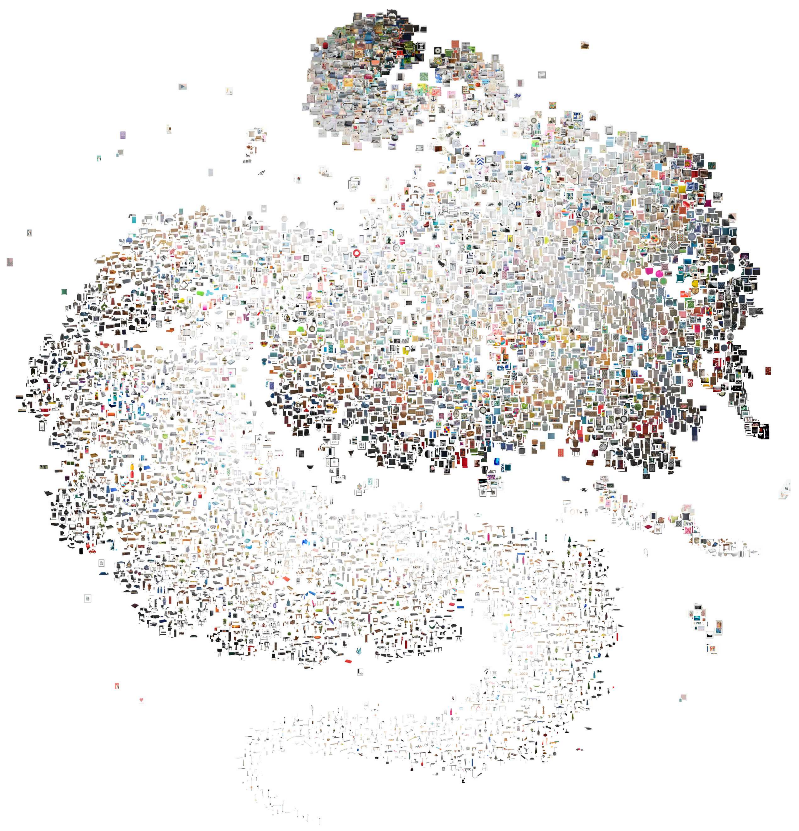


Figure 4.11: t-SNE projection of images represented by RGB histogram

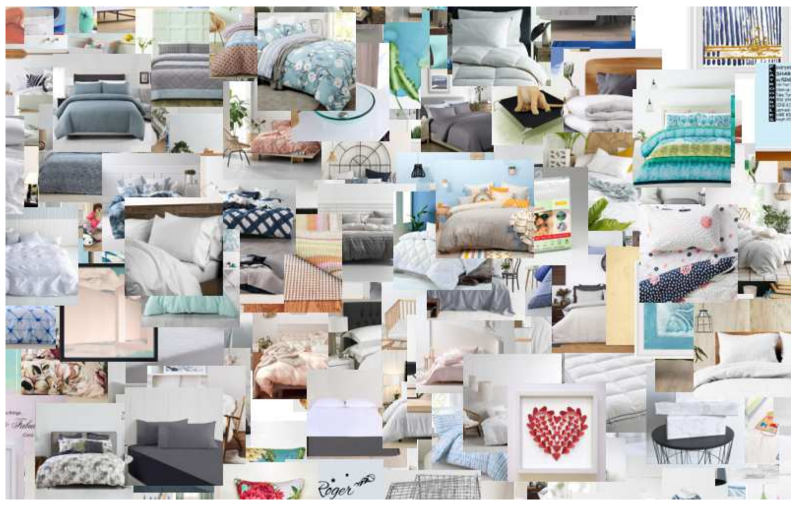


Figure 4.12: Detail view of projection 4.11

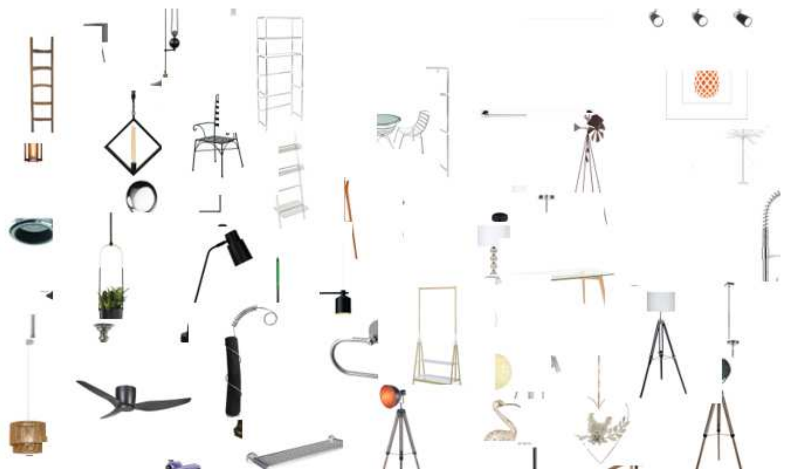


Figure 4.13: Detail view of projection 4.11

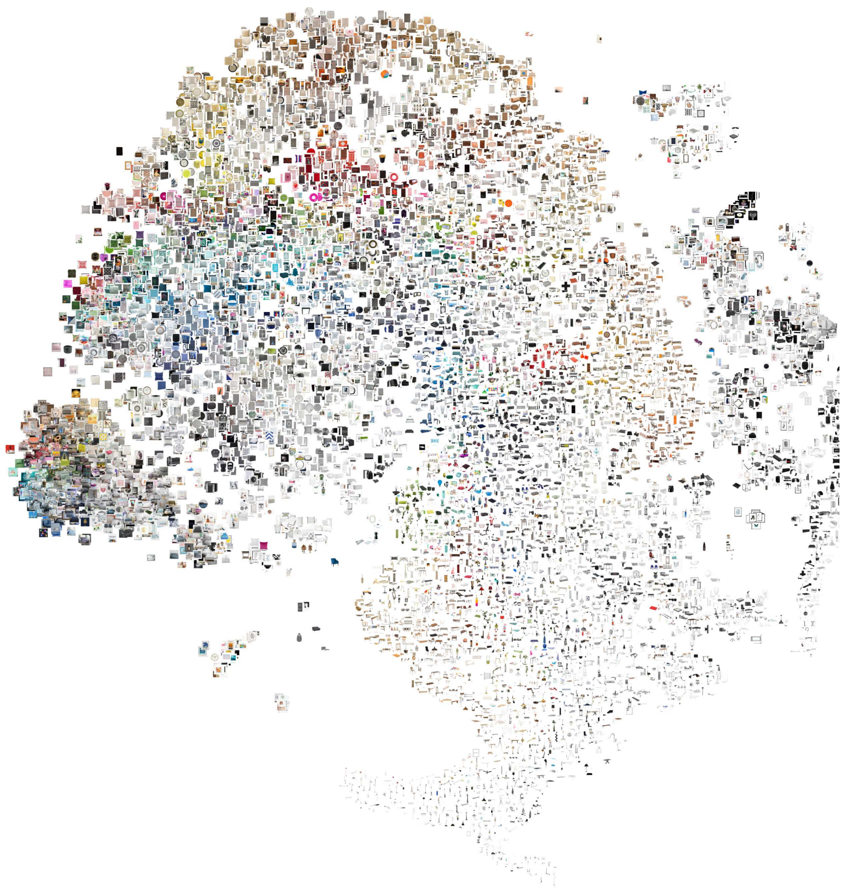


Figure 4.14: t-SNE projection of images represented by HSV histogram

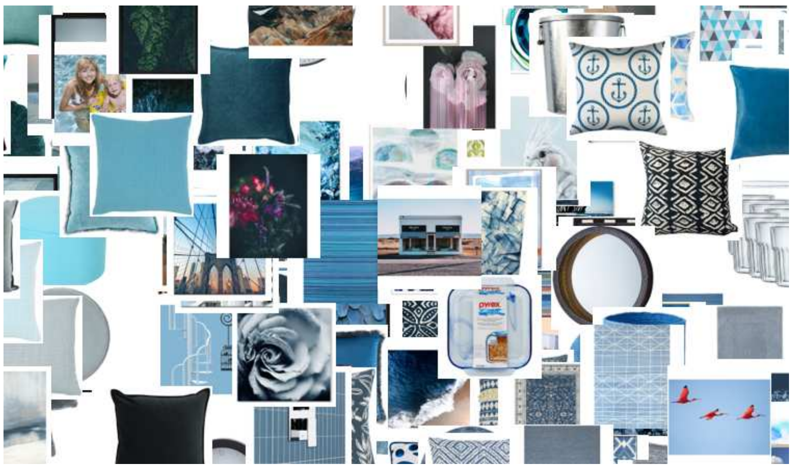


Figure 4.15: Detail view of projection 4.14

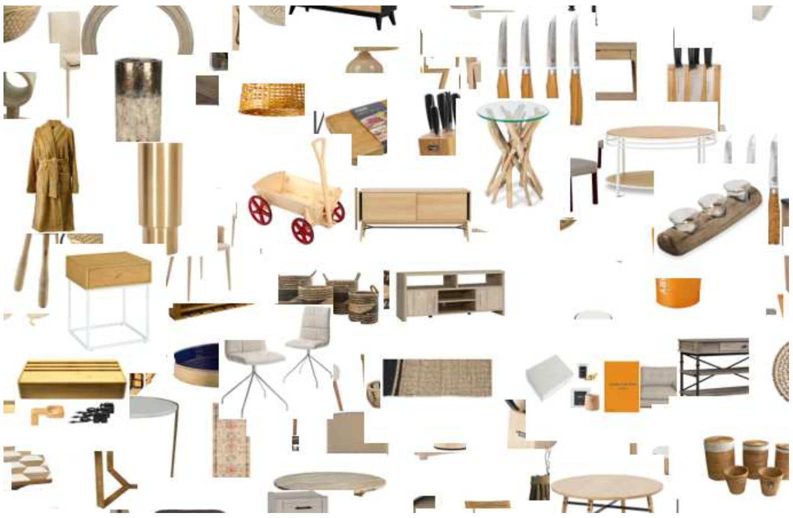


Figure 4.16: Detail view of projection 4.14

t-SNE projection [39] into two-dimensional space and can be seen in Figure 4.11. It can be observed that images with a big proportion of white background gather in the lower part of the visualization and photographs of rooms and other scenes are in the upper part of the Figure. Detail views of this visualisation can be seen in Figures 4.12 and 4.13.

HSV histograms store 180 integer values in the Hue layer, 256 values in the Saturation layer and 256 values in the Value layer. The resulting vector of this image representation is 692 B long. t-SNE projection of product images using HSV image features is shown in Figure 4.14. Compared to previous

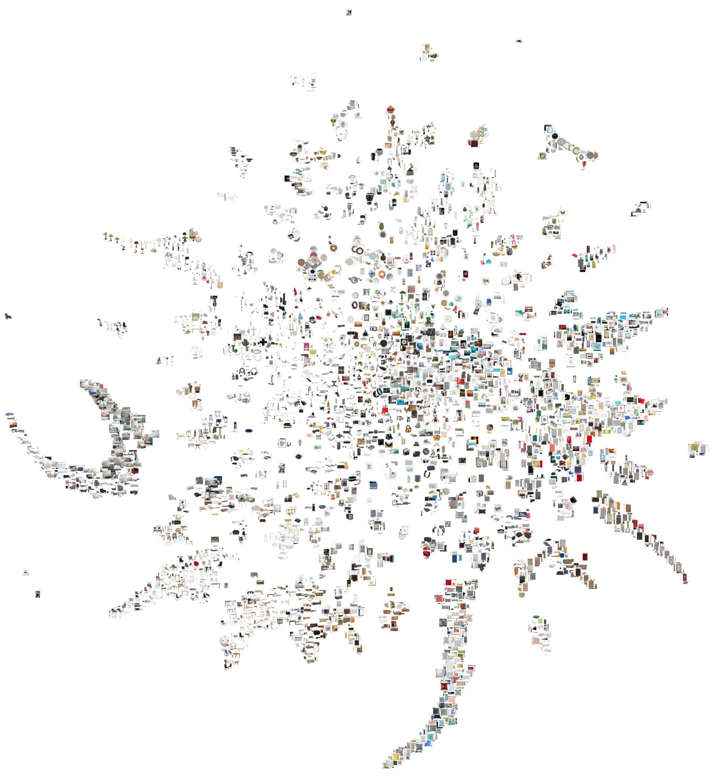


Figure 4.17: t-SNE projection of images represented by the last layer output of VGG16

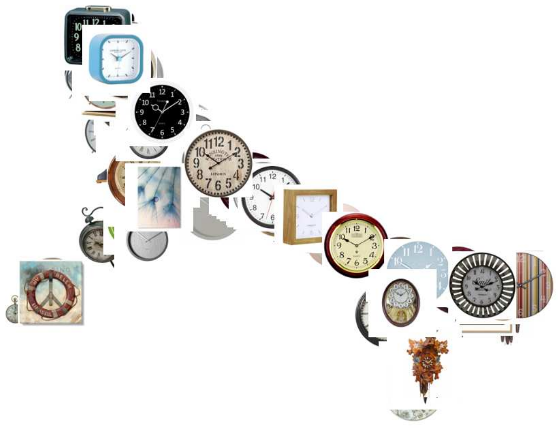


Figure 4.18: Detail view of projection 4.17

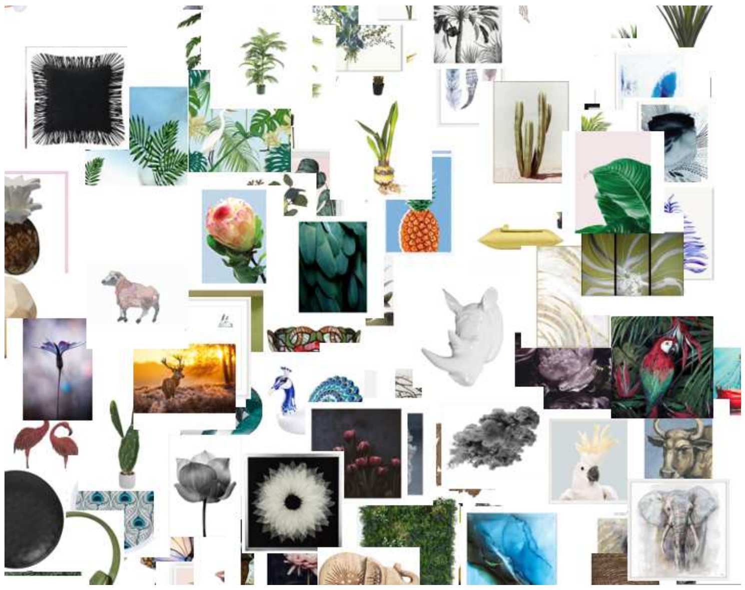


Figure 4.19: Detail view of projection 4.17

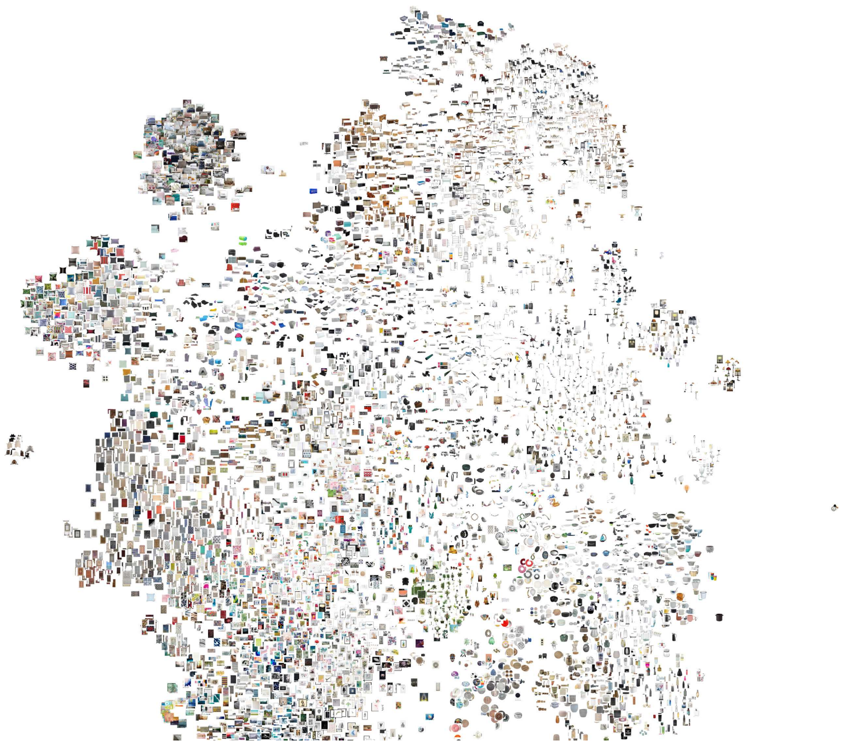


Figure 4.20: t-SNE projection of images represented by the penultimate layer output of VGG16

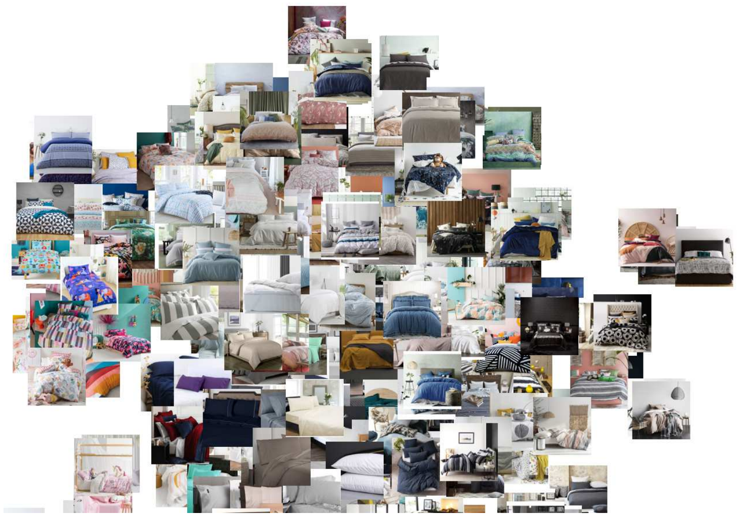


Figure 4.21: Detail view of projection 4.20

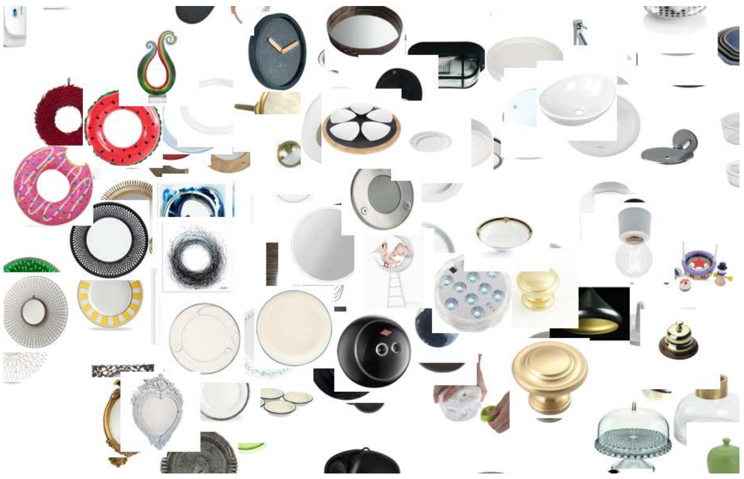


Figure 4.22: Detail view of projection 4.20

RGB histogram, HSV describes images more naturally according to human vision and images with similar hue are closer to each other. Groups of images with similar colour can be seen in detailed cut off in Figures 4.15 and 4.16.

High-level features were extracted by a convolutional neural network. For the needs of this thesis, VGG16 CNN model [40] was used. VGG16 convolutional neural network is 16 layers deep and was trained on ImageNet dataset [41] that has over 14 million images belonging to 1000 classes labelled by humans. Based on the previous research, the outputs of the last layer and the penultimate layer were used to define the images.

The images represented by the last layer are 1000 B long and each byte represents a weight of the class in the image. Their t-SNE projection is shown in 4.17. From the detail views in 4.18 and 4.19, it can be seen that products of the same type/category are close to each other.

Similar Figures are observable in Figures 4.20, 4.21 and 4.22 when the output of the penultimate layer was used. Images were represented by 4096 B long vector which is a mixture of high-level and low-level features.

4.4 Proposed recommendation algorithms using product images

The objective of this thesis is to design recommendation systems for various scenarios with the usage of product images. As it was already presented in section 1.3, there are various types of scenarios and when designing a recommendation system, the placement of the recommended products should be considered. The product page, the category page, the landing page or the cart page may require different approaches according to business goals.

In section 1.4 three basic categories of recommendation systems were presented - content-based RSs, RSs based on collaborative filtering and hybrid RSs. This structure was considered when designing recommendation systems with usage of product images.

4.4.1 Reference recommendation system based on collaborative filtering

At the beginning, a reference recommendation system based on collaborative filtering was built. A user-based k-NN (k-Nearest Neighbor) model was used, which is a memory-based method of collaborative filtering, already presented in 1.4.

The rating matrix R was filled with data from detail_views table, so the resulting matrix of size $|U| \times |I|$ stored information about the number of visits of product pages (r_{ui} = number of visits i by $u, u \in U, i \in I$). User u is then represented by a row in this matrix ($r_{u\bullet}$).

The k-Nearest Neighbor method compares these rows and retrieves k most similar users to the original user u. As a similarity function, a cosine similarity was used (4.1).

$$similarity = \frac{\sum\limits_{i \in I} r_{u_1 i} * r_{u_2 i}}{\sqrt{\sum\limits_{i \in I} r_{u_1 i}^2}\sqrt{\sum\limits_{i \in I} r_{u_2 i}^2}} \tag{4.1}$$

When having all k nearest neighbours, user vectors are summed up and N not-yet-visited items with the highest number of interactions are selected for recommendation to the original user u.

The second reference recommendation system is an analogy to the first one but instead of user-item interactions, user-category matrix was used. It was also based on detail_views interactions and the category of an item was determined by column “product_type” in the table items.

4.4.2 Proposed RSs based on collaborative filtering

Classic user-based collaborative filtering works with rating matrix that stores only interactions between users and items. It does not consider the content of the products (items) at all but in cases like web stores (and especially web stores with furniture, home accessories and decoration where visual appearance is very important), item features can be valuable information.

Another typical issue of a web store is the range of offered products. In the provided dataset, there are over 80 000 of items and for a normal customer, it is impossible to go through them all. Usually, some of the products differ in minor details such as size (carpets, mirrors, . . .), pattern (plates, blankets, . . .) or colour (pillows, bookshelves, . . .). Visiting product pages of several mugs does not necessarily mean, that the user is interested in just these mugs, but it can be presumed that she/he is interested in mugs in general (and did not watch the others just because they were not recommended to her/him). This matter of fact is not taken into consideration when using classic collaborative filtering at all.

The following proposed method has the ambition to overcome these problems and incorporate product content (in the form of image embeddings) into a user-based collaborative filtering method.

The main thought of the proposed method is to group visually similar items into one category and represent users by interactions with these categories except for individual items. For grouping, the k-means clustering method was used where items were represented by one of the earlier introduced image embeddings. Category of an item is then determined by the nearest cluster centroid.

This method works with rating matrix R_c of size $|U| \times |C|$ where C is computed categories and r_{uc} is a number of interactions with the category c

$(c \in C, c \subset I)$:

$$r_{uc} = \sum_{i \in c} r_{ui} \tag{4.2}$$

After creating matrix R_c, there are two approaches to how to use it:

1. using R_c to find similar users with k-NN, but recommend top N items using the original matrix R (TIR), or
2. using R_c to find similar users with k-NN and recommend top N categories using the same matrix R_c (CR).

The first approach can be suitable for landing, cart or product pages and banners like "Recommended for you", "You might also like. . . ", "Inspired by your browsing history" or "Items usually bought together".

The second approach could be used for landing pages, category pages or messaging. On the landing page or in a commercial message it may determine the user's favourite product categories, which can be a basis for recommending new products that do not have any interactions yet.

4.4.3 Proposed content-based RSs

In contrast with classic collaborative filtering, content-based methods focus on item features.

This thesis proposes several content-based recommendation systems that use image embeddings as well as user interactions.

The first recommendation system takes image embeddings presented in the previous section 4.3, then normalization across all dataset of the embeddings can be used and at the end, it recommends top N most similar items based on the cosine similarity (4.1).

This approach does not require any user-item interactions and can be used straight away for scenarios like product page and banners named "Similar items" or "Match your style".

The second recommendation system uses user interactions to represent the user u by the same type of vector as an image. Based on the user interactions $r_{u\bullet}$, the system sums up the image embeddings of items the user visited (img_i) and then compares the user vector with the rest of the items (image embeddings):

$$u = \sum_{i \in I} r_{ui} * img_i \tag{4.3}$$

This method can be used on the landing page as well as product page and banners like “Recommended for you”, “Based on your style” or universal “You may also like. . . ”.

Chapter 5

Results

In the last chapter the proposed recommendation algorithms are evaluated and the results are presented and discussed.

5.1 Methodology

The proposed recommendation systems (4.4) were trained on the provided offline dataset from an actual online retailer. The dataset was presented in section 4.2 and data were cleansed as it was presented in subsection 4.2.4.

Four types of image embeddings we extracted from 80 228 product images (4.3):

1. RGB histogram of length 768 B,
2. HSV histogram (692 B),
3. the output of the last layer of CNN VGG16 (1000 B) and
4. the output of the penultimate layer of CNN VGG16 (4096 B).

User-based k-NN recommendation systems (4.4.2) were evaluated for k = (1, 2, 3, 5, 7, 10, 13, 15, 20, 25, 30, 35, 40, 50, 100, 150, 200, 250, 300, 400, 500), cosine similarity (4.1) was used as the similarity measure and the items of the proposed systems (4.4.2) were clustered into 50, 100, 200, 350 and 500 categories. As the main evaluation metrics for these models, recall (1.4) and coverage (1.7) was chosen - the aim is to balance the recall with catalog coverage so not only popular items were being recommended and the model still performed good. All of the models were evaluated for top 5 items.

Proposed content-based recommendation systems (4.4.3) also used cosine similarity (4.1) as the similarity measure for the top 5 similar items.

The test environment is described in the next section 5.2, results are presented in section 5.3.

5.2 Test environment

The models were tested on a personal laptop Lenovo S430 with 12GB of RAM and Intel®Core$^{\text{TM}}$i5-3210M CPU with 2.50GHz clock speed and 4 cores, running 64bit Arch Linux.

The model was written in Python 3, for the extraction of high-level features, Keras library with pre-trained VGG16 was used.

A PostgreSQL database was provided.

5.3 Evaluation results

5.3.1 Reference recommendation system based on collaborative filtering

In the beginning, a classic user-based k-NN model with user-item rating matrix was trained. There were two approaches - detail_views (works only with an interaction matrix created from detail_views table) and weighted combo that combines all four interaction tables presented in subsection 4.2. It was presumed that approach that uses just one table detail_view would perform better because it does not favour already popular items (products that were added to cart or purchased). The presumptions were correct as it is visible in the graph 5.1 (left).

Results of the detail_views approach served as a benchmark/reference result for the proposed recommendation systems in the next subsection.

The second reference model was trained on the user-category rating matrix according to 4.4.2. The results are shown in the graph 5.1 (right). Category recommendation already performs better than recommending the items because the rating matrix became less sparse due to item compression to 214 categories.

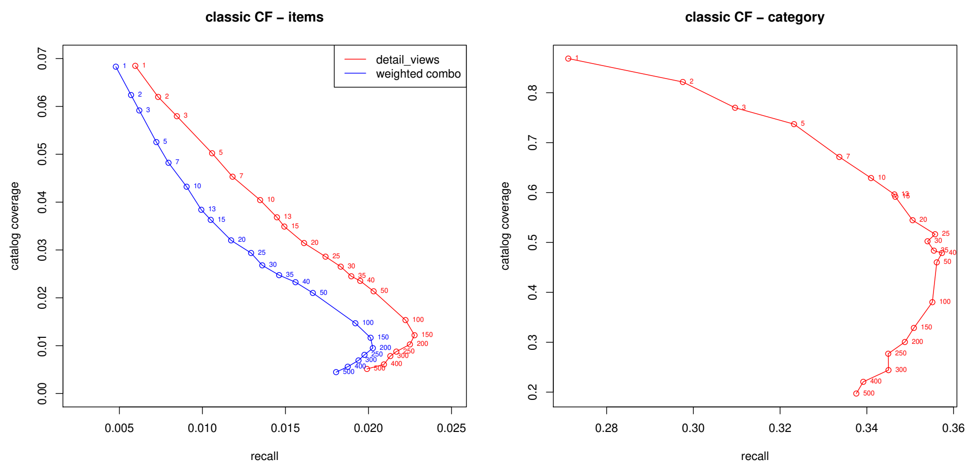


Figure 5.1: classic user-based k-NN method

5.3.2 Proposed RSs based on collaborative filtering

In proposed recommendation systems based on collaborative filtering, four types of image embeddings were observed:

1. HSV histogram (Hsv),
2. HSV histogram normalized by TF-IDF method [3] (HsvTFIDF),
3. the output of the last layer of VGG16 (H1) and
4. the output of the penultimate layer of VGG16 (H2).

All of the embeddings were tested in the method that recommends top 5 items (TIR = top items recommendation) and method that recommends the calculated categories (CR = category recommendation) on different ks as it was introduced in subsection 4.4.2 and 5.1.

There was an assumption that low-level image embeddings may help in the CR method but probably would not perform well in the TIR method just because of the fact that users choose the products primarily according to their type and secondarily on their appearance. On the other hand, high-level

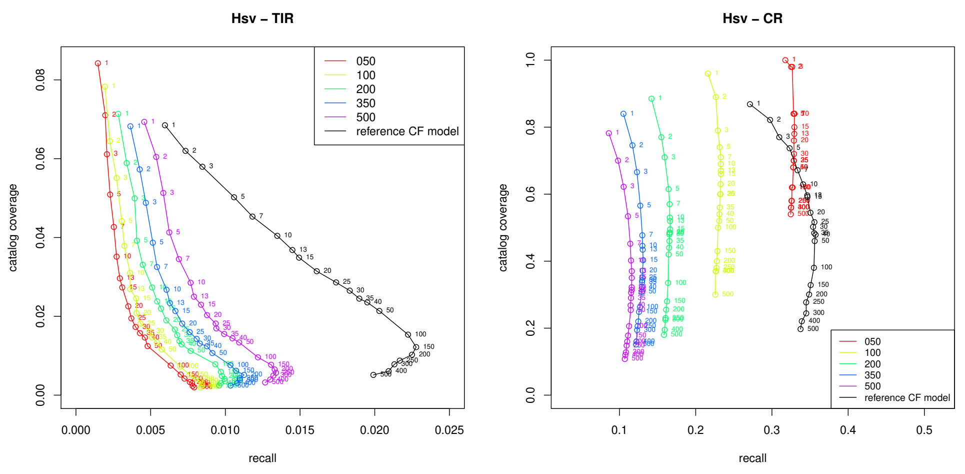


Figure 5.2: Proposed RS based on CF with HSV

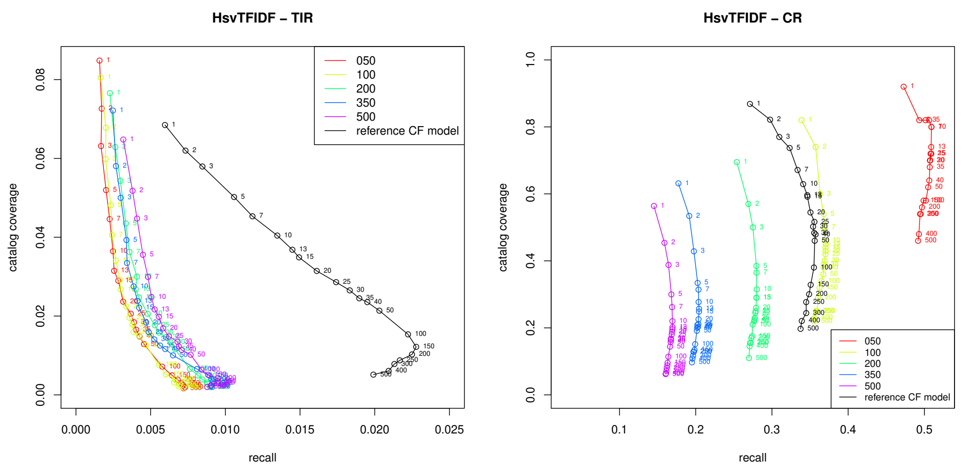


Figure 5.3: Proposed RS based on CF with HSV and TF-IDF

image embeddings may compete well with the reference method in the TIR method as it reflects the content itself (and not just low-level visual features).

Results of the models are shown in Figures 5.2, 5.3, 5.4 and 5.5. In Figure 5.6, the performances of the image embeddings are compared next to each other.

The assumptions turned out correct. TIR model that uses low-level features recommends items that users with similar taste (based on the features) viewed. In practice, it can mean that the model would recommend a user, for

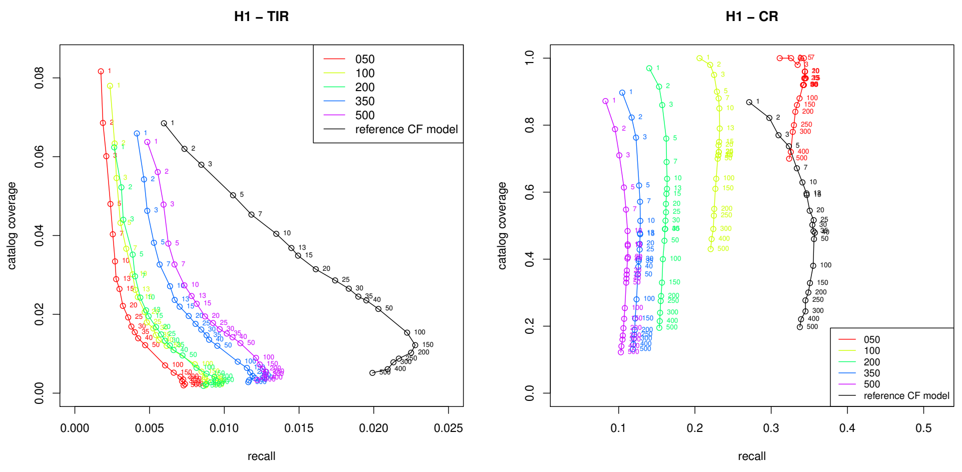


Figure 5.4: Proposed RS based on CF with last layer of VGG16

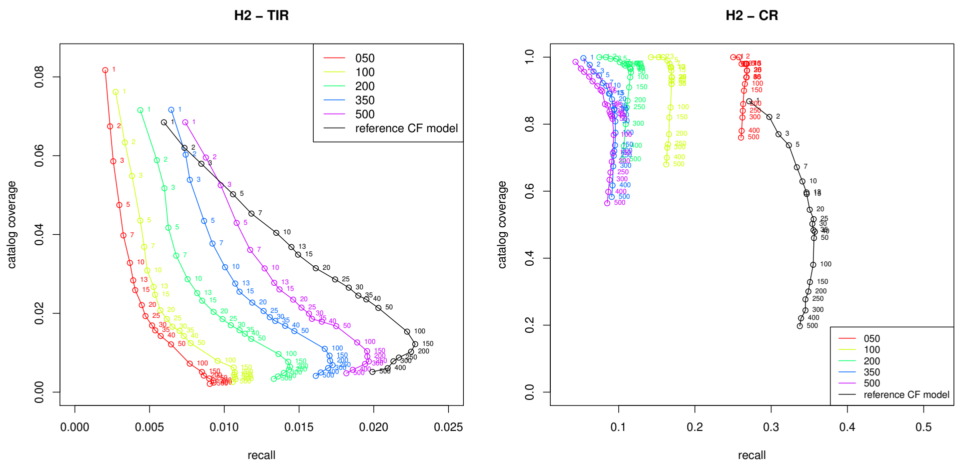


Figure 5.5: Proposed RS based on CF with penultimate layer of VGG16

instance, red items but with dataset like this, it is illogical because customers usually buy things primarily on the type than on the appearance. That is why high-level features performed better while recommending items.

In the case of the CR models, the results were the other way round - low-level features performed better in this case and high-level features turned out not so suitable for this approach. In many cases, the proposed model outperformed the reference model. This model recommends categories of the products and low-level features can determine the style of an item. As it was

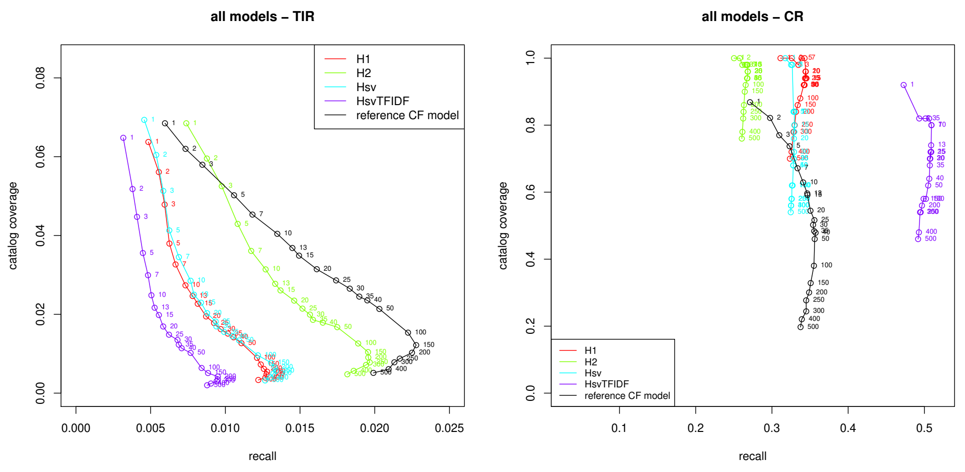


Figure 5.6: Comparison of all proposed RSs based on CF

said before, this could play a big role when buying products so they visually match together. And this is what low-level features meet - they represent items of the same colour near one to another as it is visible in Figures 4.15 and 4.16. A reason why high-level features performed poorly can be the fact, that VGG16 is trained on ImageNet dataset that has 1000 categories and only little of them relate to homeware.

It is evident that TIR models performed better with higher number of clusters and with approximately 150 nearest neighbours, whereas CR models performed better with lower number of clusters and values of recall did not change with k that much. This could make the CR models successful in real-time applications as they were also trained faster.

5.3.3 Proposed content-based RSs

Results of the first proposed content-based recommendation systems are shown in Figures 5.7 - 5.30. The query images are the first bounded images in the row, next five images are top 5 similar images. All four types of image embeddings were used and TF-IDF normalization was applied to the embeddings as well.

Unfortunately, there is no method how to evaluate these proposed models offline because in the provided dataset there was no information about similar items. Results can be deduced from the visual observation.

It was assumed that low-level features would retrieve items similar in colour and high-level features would retrieve items of the same type/category. Low-level features could be then used for recommending items with the same style and help the users to explore a wider range of items, whereas high-level features would retrieve similar items that would differ in minor details and would offer the user alternative items.

It was also assumed that normalization of the image embeddings would disqualify the white background that is present in a big part of images and highlight unusual colours.

It turned out that most of the assumptions were correct, only the normalization of the high-level features is unnecessary because every value of the image embedding is weighted by default.

The second proposed recommendation system that uses the same type of vector for user as for an image was also tested. Recall and coverage was measured on this model but testing results for small subset were bad enough not to continue with evaluating on the full dataset. Coverage of this method was satisfactory, but recall remained at zero for most of the time.

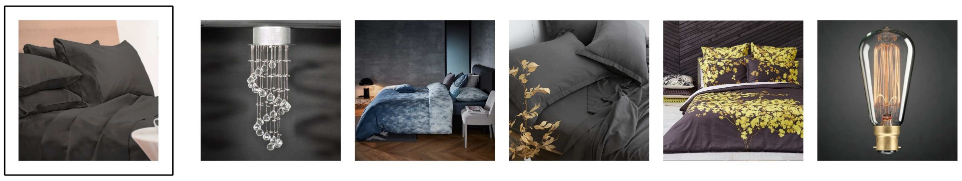


Figure 5.7: RGB

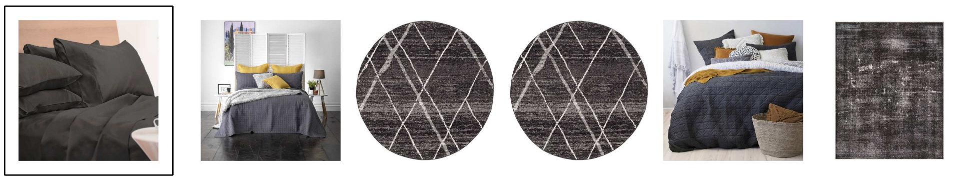


Figure 5.8: RGB + TF-IDF

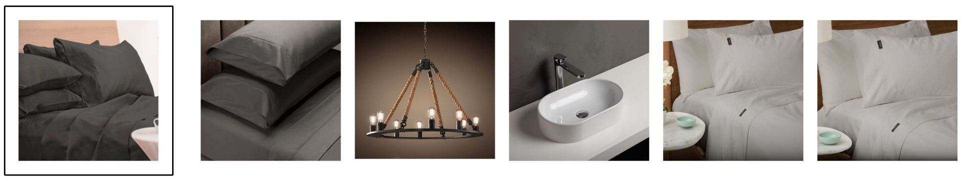


Figure 5.9: HSV

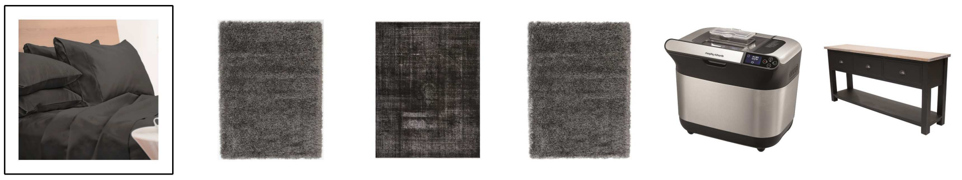


Figure 5.10: HSV + TF-IDF

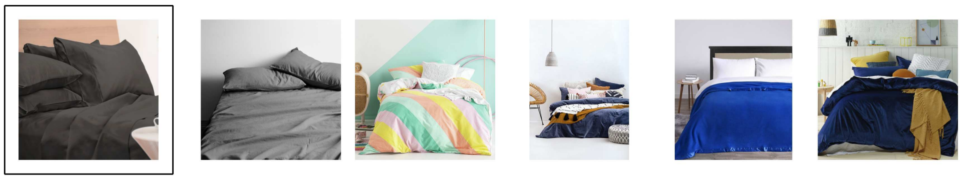


Figure 5.11: VGG16 - last layer

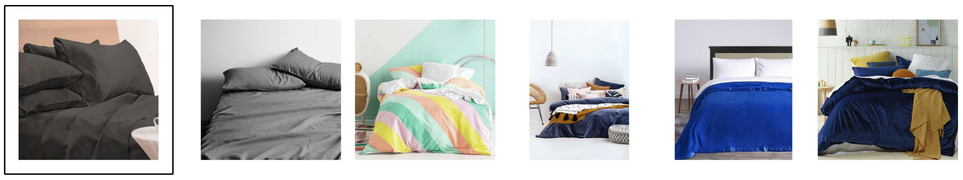


Figure 5.12: VGG16 last layer + TF-IDF

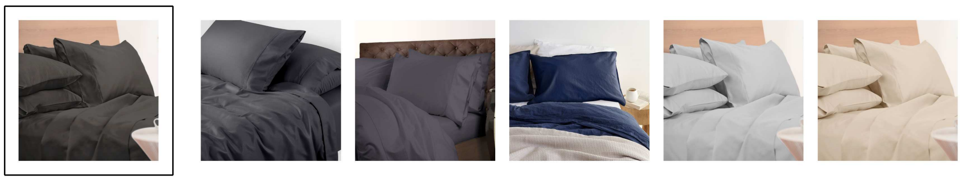


Figure 5.13: VGG16 penultimate layer

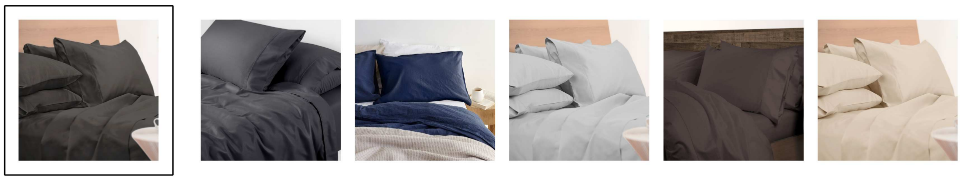


Figure 5.14: VGG16 penultimate layer + TF-IDF

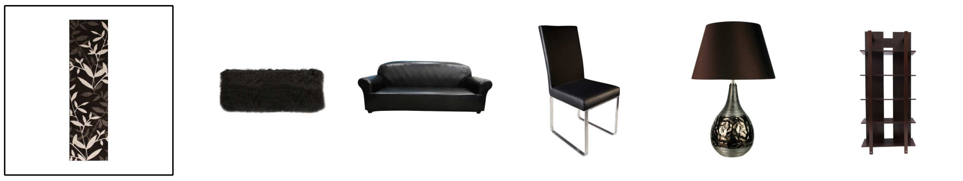


Figure 5.15: RGB

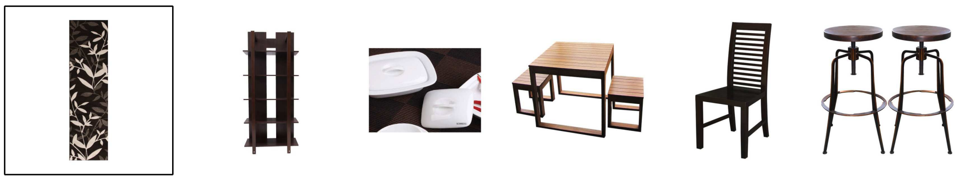


Figure 5.16: RGB + TF-IDF

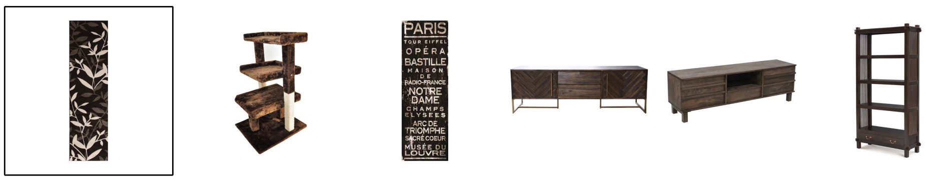


Figure 5.17: HSV

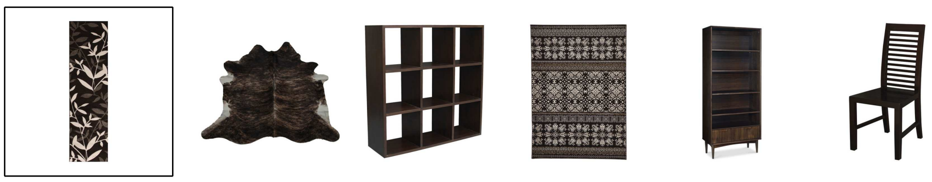


Figure 5.18: HSV + TF-IDF

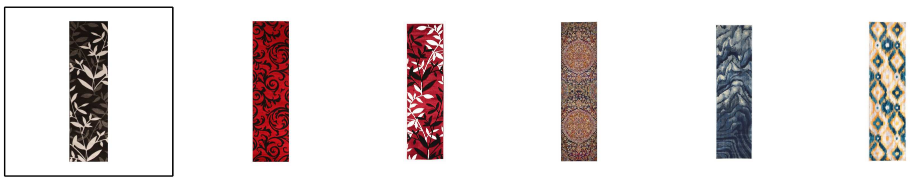


Figure 5.19: VGG16 last layer

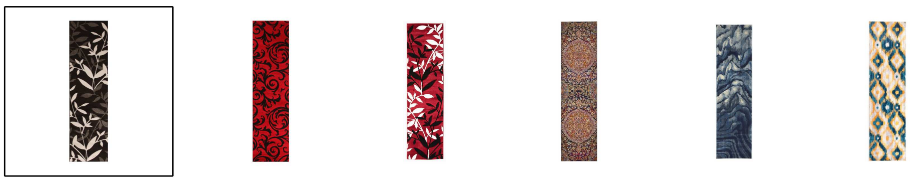


Figure 5.20: VGG16 last layer + TF-IDF

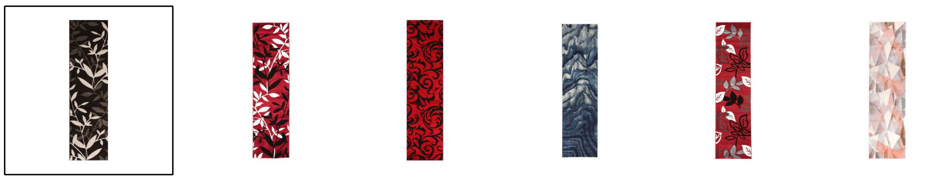


Figure 5.21: VGG16 penultimate layer

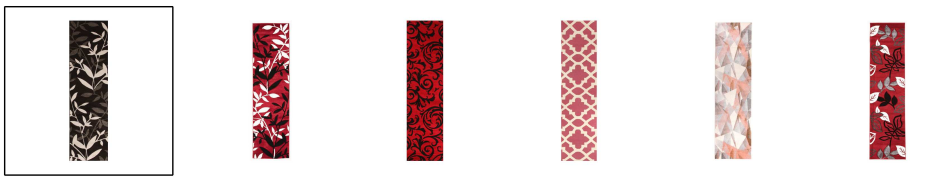


Figure 5.22: VGG16 penultimate layer + TF-IDF

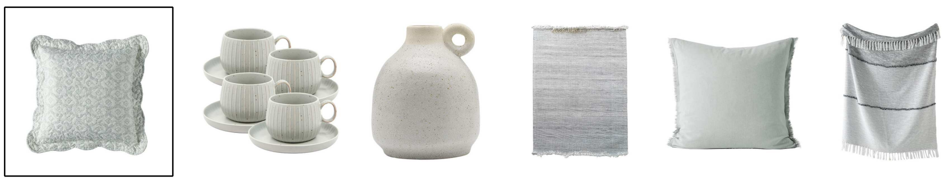


Figure 5.23: RGB

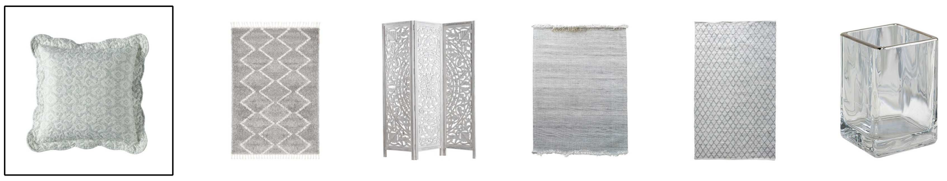


Figure 5.24: RGB + TF-IDF

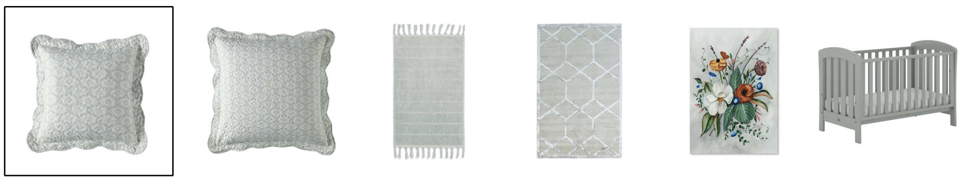


Figure 5.25: HSV

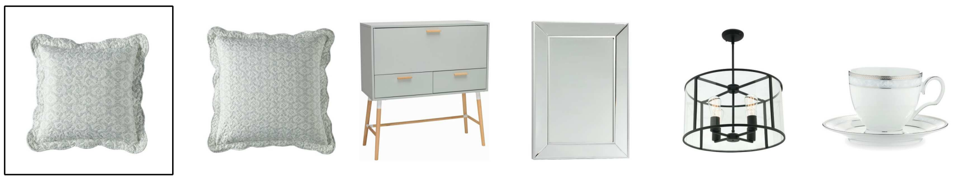


Figure 5.26: HSV + TF-IDF

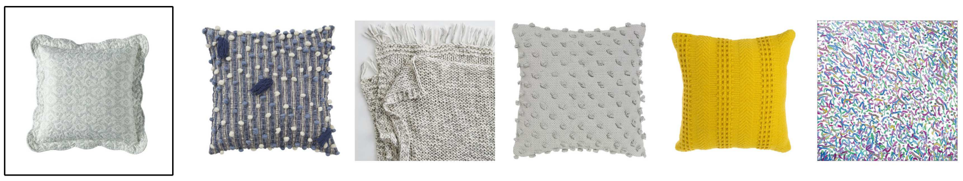


Figure 5.27: VGG16 last layer

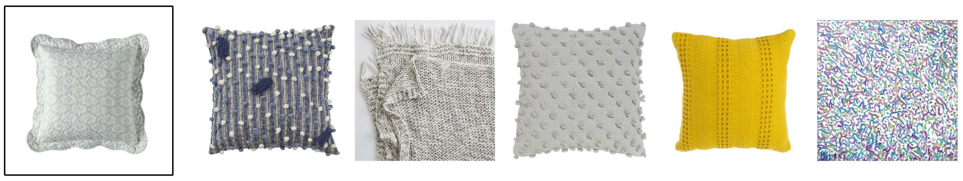


Figure 5.28: VGG16 last layer + TF-IDF

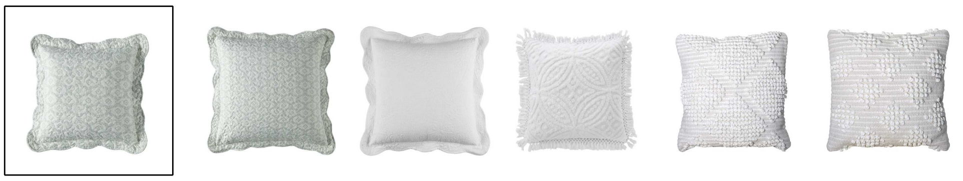


Figure 5.29: VGG16 penultimate layer

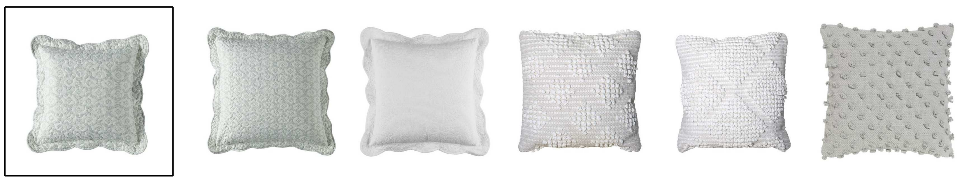


Figure 5.30: VGG16 penultimate layer + TF-IDF

Conclusion

In this work, a survey on recommendation systems was made. The author focused on the usage of product images in recommendation algorithms, studied image processing techniques and extraction of image embeddings. Based on this research, several recommendation algorithms were proposed and their use was suggested for various scenarios, such as the product page, landing page, cart page or messaging. The models were evaluated offline on a provided production dataset and the results were discussed.

In some cases, using image embedding outperformed classic collaborative filtering. Also, a novel method of recommending product categories was presented.

The models were trained on an actual production dataset which is more challenging than common academic datasets. It did not contain explicit ratings and data had to be cleaned before training. Results are more relevant to real world recommendation problems.

It must be mentioned that the model evaluation was performed on offline data, which is not a trivial task. The evaluation of one model took usually several hours and a lot of computational capacity was used for this work.

These matters of facts make in this work a room for further research. The models could be tested online to find out if the assumptions made on the basis of offline evaluation were correct.

Bibliography

[1] Melville, P.; Sindhwani, V. *Recommender Systems*. Boston, MA: Springer US, 2010, ISBN 978-0-387-30164-8, pp. 829–838, doi:10.1007/978-0-387-30164-8_705. Available from: `https://doi.org/10.1007/978-0-387-30164-8_705`

[2] Ricci, F.; Rokach, L.; et al. *Introduction to Recommender Systems Handbook*. Boston, MA: Springer US, 2011, ISBN 978-0-387-85820-3, pp. 1–35, doi:10.1007/978-0-387-85820-3_1. Available from: `https://doi.org/10.1007/978-0-387-85820-3_1`

[3] Sammut, C.; Webb, G. I. (editors). *TF–IDF*. Boston, MA: Springer US, 2010, ISBN 978-0-387-30164-8, pp. 986–987, doi:10.1007/978-0-387-30164-8_832. Available from: `https://doi.org/10.1007/978-0-387-30164-8_832`

[4] Shani, G.; Gunawardana, A. *Evaluating Recommendation Systems*, volume 12. 01 2011, pp. 257–297, doi:10.1007/978-0-387-85820-3_8.

[5] Silveira, T.; Zhang, M.; et al. How good your recommender system is? A survey on evaluations in recommendation. *International Journal of Machine Learning and Cybernetics*, volume 10, 12 2017, doi:10.1007/s13042-017-0762-9.

[6] Ge, M.; Delgado, C.; et al. Beyond accuracy: Evaluating recommender systems by coverage and serendipity. 01 2010, pp. 257–260, doi:10.1145/1864708.1864761.

[7] Upadhyaya, N.; Dixit, M. A Review: Relating Low Level Features to High Level Semantics in CBIR. *International Journal of Signal Processing, Image Processing and Pattern Recognition*, volume 9, 03 2016: pp. 433–444, doi:10.14257/ijsip.2016.9.3.37.

[8] Lowe, D. G. Object recognition from local scale-invariant features. In *Proceedings of the Seventh IEEE International Conference on Computer Vision*, volume 2, 1999, pp. 1150–1157 vol.2.

[9] Bay, H.; Ess, A.; et al. Speeded-Up Robust Features (SURF). *Comput. Vis. Image Underst.*, volume 110, no. 3, June 2008: pp. 346–359, ISSN 1077-3142, doi:10.1016/j.cviu.2007.09.014. Available from: `https://doi.org/10.1016/j.cviu.2007.09.014`

[10] SONKA, V. H. a. R. B., Milan. *Image processing, analysis, and machine vision.* Toronto [u.a.]: Thomson, third edition, 2008, ISBN 0495244384.

[11] Hinton, G. E. Reducing the Dimensionality of Data with Neural Networks. *Science*, volume 313, no. 5786, 2006: pp. 504–507, doi:10.1126/science.1127647.

[12] jia Li, L.; Su, H.; et al. Object Bank: A High-Level Image Representation for Scene Classification & Semantic Feature Sparsification. In *Advances in Neural Information Processing Systems 23*, edited by J. D. Lafferty; C. K. I. Williams; J. Shawe-Taylor; R. S. Zemel; A. Culotta, Curran Associates, Inc., 2010, pp. 1378–1386. Available from: `http://papers.nips.cc/paper/4008-object-bank-a-high-level-image-representation-for-scene-classification-semantic-feature-sparsification.pdf`

[13] Sitaula, C.; Xiang, Y.; et al. Indoor image representation by high-level semantic features. *CoRR*, volume abs/1906.04987, 2019, 1906.04987. Available from: `http://arxiv.org/abs/1906.04987`

[14] Yu, L.; Han, F.; et al. A content-based goods image recommendation system. *Multimedia Tools and Applications*, volume 77, no. 4, Feb 2018: pp. 4155–4169, ISSN 1573-7721, doi:10.1007/s11042-017-4542-z. Available from: `https://doi.org/10.1007/s11042-017-4542-z`

[15] Tamura, H.; Mori, S.; et al. Textural Features Corresponding to Visual Perception. *IEEE Transactions on Systems, Man, and Cybernetics*, volume 8, no. 6, June 1978: pp. 460–473, ISSN 2168-2909, doi:10.1109/TSMC.1978.4309999.

[16] Kawattikul, K. Product Recommendation using Image and Text Processing. In *2018 International Conference on Information Technology (InCIT)*, Oct 2018, pp. 1–4, doi:10.23919/INCIT.2018.8584860.

[17] Belongie; Malik. Matching with shape contexts. In *2000 Proceedings Workshop on Content-based Access of Image and Video Libraries*, June 2000, pp. 20–26, doi:10.1109/IVL.2000.853834.

[18] Tuinhof, H.; Pirker, C.; et al. Image-Based Fashion Product Recommendation with Deep Learning. *Lecture Notes in Computer Science*, 2019: pp. 472–481, ISSN 1611-3349, doi:10.1007/978-3-030-13709-0_40. Available from: `http://dx.doi.org/10.1007/978-3-030-13709-0_40`

[19] Liu, Z.; Luo, P.; et al. DeepFashion: Powering Robust Clothes Recognition and Retrieval with Rich Annotations. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016, ISSN 1063-6919, pp. 1096–1104, doi:10.1109/CVPR.2016.124.

[20] Sejal, D.; Ganeshsingh, T.; et al. Image Recommendation Based on ANOVA Cosine Similarity. *Procedia Computer Science*, volume 89, 2016: pp. 562–567, ISSN 1877-0509, doi:https://doi.org/10.1016/j.procs.2016.06.091. Available from: `http://www.sciencedirect.com/science/article/pii/S1877050916311565`

[21] Sejal, D.; Ganeshsingh, T.; et al. ACSIR: ANOVA Cosine Similarity Image Recommendation in vertical search. *International Journal of Multimedia Information Retrieval*, volume 6, no. 2, Jun 2017: pp. 143–154, ISSN 2192-662X, doi:10.1007/s13735-017-0124-0. Available from: `https://doi.org/10.1007/s13735-017-0124-0`

[22] Sejal, D.; Abhishek, D.; et al. IR_URFS_VF: image recommendation with user relevance feedback session and visual features in vertical image search. *International Journal of Multimedia Information Retrieval*,

volume 5, no. 4, Nov 2016: pp. 255–264, ISSN 2192-662X, doi:10.1007/s13735-016-0111-x. Available from: `https://doi.org/10.1007/s13735-016-0111-x`

[23] Chi, H.-Y.; Chen, C.-C.; et al. UbiShop: Commercial item recommendation using visual part-based object representation. *Multimedia Tools and Applications*, volume 75, no. 23, Dec 2016: pp. 16093–16115, ISSN 1573-7721, doi:10.1007/s11042-015-2916-7. Available from: `https://doi.org/10.1007/s11042-015-2916-7`

[24] Piazza, A.; Zagel, C.; et al. Outfit Browser – An Image-data-driven User Interface for Self-service Systems in Fashion Stores. *Procedia Manufacturing*, volume 3, 2015: pp. 3521–3528, ISSN 2351-9789, doi: https://doi.org/10.1016/j.promfg.2015.07.686. Available from: `http://www.sciencedirect.com/science/article/pii/S2351978915006873`

[25] Wang, Y.; Li, L. An Improved Personalized Recommendation Based on Purchasing Power and Browsed Images. In *2018 IEEE 20th International Conference on High Performance Computing and Communications; IEEE 16th International Conference on Smart City; IEEE 4th International Conference on Data Science and Systems (HPCC/SmartCity/DSS)*, June 2018, pp. 321–328, doi:10.1109/HPCC/SmartCity/DSS.2018.00073.

[26] Wu, Z.; Paul, A.; et al. Improved one-class collaborative filtering for online recommendation. In *2017 International Workshop on Complex Systems and Networks (IWCSN)*, Dec 2017, pp. 205–209, doi:10.1109/IWCSN.2017.8276528.

[27] He, R.; McAuley, J. VBPR: Visual Bayesian Personalized Ranking from Implicit Feedback. 2015, `1510.01784`.

[28] Kim, D.-H. Image Recommendation Algorithm Using Feature-Based Collaborative Filtering. *IEICE Transactions*, volume 92-D, 03 2009: pp. 413–421, doi:10.1587/transinf.E92.D.413.

[29] Feng, H.; Qian, X. Mining user-contributed photos for personalized product recommendation. *Neurocomputing*, volume 129, 2014: pp. 409–

420, ISSN 0925-2312, doi:https://doi.org/10.1016/j.neucom.2013.09.018. Available from: http://www.sciencedirect.com/science/article/pii/S0925231213009363

[30] Hsiao, J.; Li, L. On visual similarity based interactive product recommendation for online shopping. In *2014 IEEE International Conference on Image Processing (ICIP)*, Oct 2014, ISSN 1522-4880, pp. 3038–3041, doi:10.1109/ICIP.2014.7025614.

[31] McAuley, J.; Targett, C.; et al. Image-Based Recommendations on Styles and Substitutes. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '15, New York, NY, USA: ACM, 2015, ISBN 978-1-4503-3621-5, pp. 43–52, doi:10.1145/2766462.2767755. Available from: http://doi.acm.org/10.1145/2766462.2767755

[32] Lei, C.; Liu, D.; et al. Comparative Deep Learning of Hybrid Representations for Image Recommendations. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun 2016, doi: 10.1109/cvpr.2016.279. Available from: http://dx.doi.org/10.1109/CVPR.2016.279

[33] Mikolov, T.; Chen, K.; et al. Efficient Estimation of Word Representations in Vector Space. 2013, 1301.3781.

[34] Liu, D.; Huo, C.; et al. Research of commodity recommendation workflow based on LSH algorithm. *Multimedia Tools and Applications*, volume 78, 02 2018, doi:10.1007/s11042-018-5716-z.

[35] Yu, W.; Zhang, H.; et al. Aesthetic-based Clothing Recommendation. *Proceedings of the 2018 World Wide Web Conference on World Wide Web - WWW '18*, 2018, doi:10.1145/3178876.3186146. Available from: http://dx.doi.org/10.1145/3178876.3186146

[36] Murray, N.; Marchesotti, L.; et al. AVA: A large-scale database for aesthetic visual analysis. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, June 2012, ISSN 1063-6919, pp. 2408–2415, doi:10.1109/CVPR.2012.6247954.

[37] Nelaturi, N.; Devi, G. L. Hybrid Recommender System Leveraging Stacked Convolutional Networks. *Journal of Engineering Science and Technology Review*, volume 11, 2018: pp. 89–96, ISSN 1791-2377, doi: 10.25103/jestr.113.12. Available from: `http://dx.doi.org/10.25103/jestr.113.12`

[38] He, M.; Zhang, S.; et al. Learning to Style-Aware Bayesian Personalized Ranking for Visual Recommendation. *IEEE Access*, volume 7, 2019: pp. 14198–14205, ISSN 2169-3536, doi:10.1109/ACCESS.2019.2892984.

[39] van der Maaten, L.; Hinton, G. Visualizing Data using t-SNE. *Journal of Machine Learning Research*, volume 9, 2008: pp. 2579–2605. Available from: `http://www.jmlr.org/papers/v9/vandermaaten08a.html`

[40] Simonyan, K.; Zisserman, A. Very Deep Convolutional Networks for Large-Scale Image Recognition. 2014, `1409.1556`.

[41] Deng, J.; Dong, W.; et al. ImageNet: A Large-Scale Hierarchical Image Database. In *CVPR09*, 2009.

Appendix A

Acronyms

RS	recommendation system
***k*-NN**	*k*-nearest neighbors
TF-IDF	term frequency-inverse document frequency
TP	true positive
TN	true negative
FP	false positive
FN	false negative
CNN	convolutional neural network
CF	collaborative filtering

Appendix **B**

Contents of enclosed CD

```
readme.txt ...................... the file with CD contents description
src ................................. the directory of source codes
  thesis ............. the directory of LaTeX source codes of the thesis
text ...................................... the thesis text directory
```