Data-driven clustering for new garment forecasting

by

Gianpaolo Luciano Rivera

B.S. Applied Mathematics
B.S. Actuarial Sciences
ITAM, 2019

Submitted to the MIT Sloan School of Management & Operations Research Center
in partial fulfillment of the requirements for the degree of

Master of Business Administration & Master of Science in Operations Research

in conjunction with the Leaders for Global Operations program

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

June 2023

Data-driven clustering for new garment forecasting

by

Gianpaolo Luciano Rivera

Submitted to the MIT Sloan School of Management & Operations Research Center
on May 12, 202, in partial fulfillment of the
requirements for the degree of
Master of Business Administration & Master of Science in Operations Research

Abstract

The ability to detect patterns early in the design process is critical for fashion firms to make decisions, particularly given the speed at which new garments are introduced. Traditionally, most garment defining features were only used by designers and buyers since the data was intractable for a computer: shape, color, fit, etc. By using natural language processing (NLP) techniques that preserve semantics, in combination with traditional data-mining, we unlock the potential to use these garment characteristic and embed them in a numerical space that's tractable. By using this novel approach to fashion data, this thesis develops two custom algorithms to forecasting the size-curve distribution of a new garment. This task is achieved by automatically finding a set of comparables of previous garments and leveraging the know results to make predictions. We develop and implement two main algorithms: *Cluster-While Regress* (CWR) and *k-Nearest Neighbours* (kNN) and show that, with given enough data, the algorithms should can achieve human-level accuracy and automate the comparables-finding process.

Thesis Supervisor: Georgia Perakis
Title: William F. Pounds Professor of Management Science & Co-Director Operations Research Center

Thesis Supervisor: Patrick Jaillet
Title: Dugald C. Jackson Professor of Computer Science & Co-Director Operations Research Center

Acknowledgments

To my parents Irma and Fernando, who have given and continue giving me everything in life. To Paulina who still is the best thing that has happened in my life, and who I am finally spending the rest of with.

To the whole team at Zara: Belén who is one of the best managers I have had. An amazing mother, statistician and supervisor. To David Polo, for helping me unravel the secrets of distributed cloud computing and productivity hardware. And to the rest of the amazing team: Jana, Lucas and Alberte. Thanks for making my internship an terrific experience, for being so supportive and for understanding my constraints. I hope one day we can all meet in person. Similarly, I wanted to acknowledge the long-standing relationship between Zara and the Leaders for Global Operations Program which I hope continues being productive.

To Georgia Perakis and Patrick Jaillet who have rooted for me and trusted my work from day one. I could not have done this without you. I feel truly blessed to have been able to work with such an important scholars. Similarly, I wanted to thank Thomas Roemer, Eileen Jacob and Dawna Levenson who, two years ago deiced I was worthy of the LGO program. Your decision changed my life for the better and is one that I will always cherish. To Patty who has is a cornerstone of the program and who we all probably owe our graduation. To the LGO staff who have made this one of the best experiences in my life.

To the amazing LGO friends that I have made: the group of people who I identify the most with. I wish we always continue this amazing friendship. To Santi, Scott, Daniel & Gus, who were my family far-away from home & the best roommates I could have wished for. Thanks for some many fun days and for supporting me when I needed it the most. To the Baltic Egrets who were the best core-MBA team ever. May we always do one more lap. In particular, to Dragana & Mert Can, so that we continue unraveling deep neural networks. To Manu, who was always there to help me understand further.

Last, to all of my MIT professors who made me grow as a professional and as a person. Special thanks to Alex Jacquillat, Sean Willems, Amr Farhat, Daniel Freund, Rama Ramakrishnan, and Peter Whincop. You all made me see life under different a different lens.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

In order for the fashion industry to launch new garments constantly, firms must make several business-critical decisions in record time. Their ability to make these decisions confidently and accurately, reflecting their unique style, ultimately determine which fashion firms succeed. In this industry, the decision range from the more *creative* endeavours such as design and color of new garments, to more *logistical* touch-points such as quantity ordered, pricing and initial shipment to stores. However, as it is shown in this work, the *creative* and *logistical* decisions are more intertwined than originally thought.

Currently in most firms, designers create new garments based on their experience, fashion trends and brand image. Once the garment designed is finished, most of the logistical decisions are taken, giving us a clear *causal* relationship that end up impacting sales.[1] In figure 1-1 we see an example of how these elements are related.[2] This thesis proposes a novel approach to automatically extract information from these *creative* decisions in order to inform those upstream in the *logistics* stage. In particular, we propose a model to automate one of the decision-points (A in figure 1-1).

1. Under this definition, the causal relationship can be interpreted like a *timeline*: the design of the garment goes first which then, influences the number of garments ordered which then, influences the total sales which then, influence the trends of the season.

2. Under this model, the sales of the garments jointly impact the *fashion trends* which do help inform the designs. However, this is not a clear-cut relationship like it is for the regular operation of the firm.

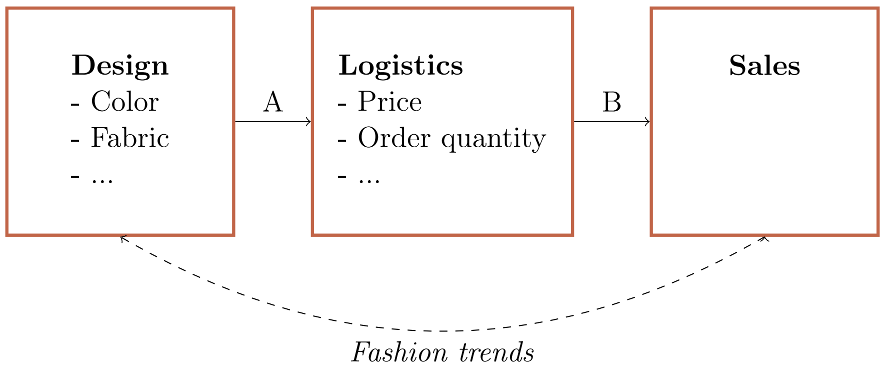


Figure 1-1: The causal decision-making-process for new garments

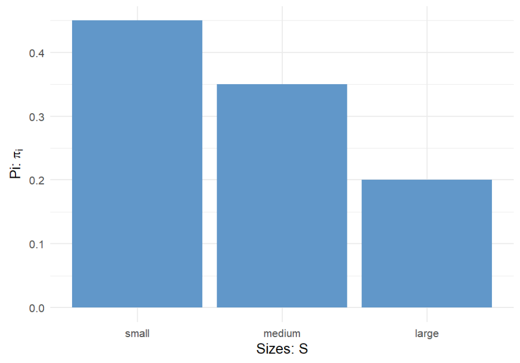
1.1 The size-curve problem

One of the most critical logistical decision can be simply stated as follows: *what is the appropriate size-curve distribution a new garment should have?*. In order to unravel the problem, first, a **size-curve distribution** is defined in simple terms as: the proportion ($\pi_{i,s}$) each size s encompasses of the total order quantity Q_i for any given garment i. For example, 35% smalls, 40% mediums and 25% larges.[3] Graphical examples of the size-curve distribution can be found in figure 1-2. In this example, less extreme sizes (e.g. small & medium) are larger. In reality, since population sizes follows a normal distribution, it follows that the aggregated demand should roughly, should replicate this shape. However, for an individual garment, this size-curve can be completely different and unique, depending on the branding, the design itself or even market trends. Basically, the true size-curve is a garmetn-specific measure. Importantly, note that under this model deciding the size curve $\pi_{i,s}$ represent an independent decision from the total order quantity Q_i, since this number is assumed decided beforehand.[4]

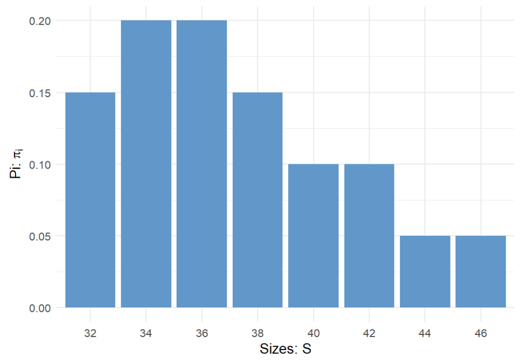
Given the definition, the problem can now be re-stated as simply *finding the right size-curve for any given new garment that closely follows the future demand for the garment.* I.e. Finding the right $\hat{\pi}_{i,s}$ so that the right sizes are available for the consumers. This particular size-curve problem is just a subset of logistical decisions made regarding new garments, however, its a critical one since it involves several independent decisions, as many as available

3. The formal definition can be found in definition 3) in section 2.3.

4. Although this problem assumes a known and fixed Q_i, deciding how much to order for each garment is a critical problem in the Operations Research literature, aptly named the *News Vendor Problem*, (Winston 2022)



(a) USA sizes



(b) European sizes for pants

Figure 1-2: Two size-curve distributions with different potential sizes S

sizes we have. A further exploration of the nuances of the *logistics* pertaining the size-curve are found in section 1.3.1.

Status-quo approach

At this stage, the new garment has been fully designed and the total order quantity is known. However, there is still relatively little information to predict how will the garment perform and therefore, making further decisions is an even harder problem. To get around this issue, designers rely on their superb business know-how and find *comparable garments*, usually only one, from a previous season (Garro 2011; Gallien et al. 2015). Using this comparables where performance data is already available, mostly sales, the remaining decisions can be taken with a higher degree of certainty. This process is based under the assumption that similar items should perform alike. Designers and buyers work in tandem, taking into account the current trends, supply chains, overall economy forecasts, potential markets where the garment will be sold and high level business objectives. Some of the decisions taken using this comparables methodology include the size-curve distribution, retail price-point, and initial week of sales. In practice, this *business heuristic* works excellently and its refinement, amongst others, is what allows firms to succeed.

However, the process still relies on subjective know-how of the buyers which can be unreliable at times. More importantly this process is not:

permanent: since the employees do not work forever at a single firm and do not have perfect memory. Also, firms rarely store which comparables were used to make which decisions. Nor,

replicable: since different designers can view different garments as more similar to each other.

Moreover, there are several implicit biases that will be explored in-depth at section 2.3. In short, these implicit biases skew the data that is available, even for these comparable garments and hence, make all future forecasts skewed as well. This sub-problem is referred to as *censorship in the data* and can be understood as: *in a store with infinite inventory, where clients always are exposed to all sizes of the garment, what quantity of the garment will sell?*. Clearly, a hard problem to solve if we only observe the truncated data. As an example, if a firm buys only 100 units of one garment and they sell all of them, we could consider the accuracy to be perfect. However, the firm could have sold more.

1.2 Methodology and contributions

The objective of this thesis is to forecast the size curve distribution for any new garment, aiding the manual status quo process. Hence, any forecasting algorithms must be able to:

1. Find *good* set of comparable garments in an automated manner. These comparable should be *similar* to those the designers would have chosen.

2. Forecast the size curve distribution that is, ideally, at least at least as accurate as the buyers (in retrospect).

The largest contributions of this thesis is to treat the fashion garment characteristics as useful data structures. Then, using this data, we develop two custom forecasting algorithms to solve the problem above. These contributions are covered extensively in chapters 2 and 3 respectively.[5]

We consider the following methodology. First, we present a brief introduction to Inditex /

5. The literature upon which this work is built on is reviewed directly on its relevant sections. This stylistic decision seeks to ensure the easiness of exposition and understanding.

Zara's operations in section 1.3. We explore their business since they are pioneers of the data-driven approach to fashion decision making. We partnered with them during a required 6-month internship for the MIT Leaders for Global Operations Program (LGO) program. Their collaboration and expertise is invaluable and the results presented rely heavily on their guidance.

Chapter 2 presents how the fashion garments are *embedded* into useful data representations by using Natural Language Processing (NLP). This representation is the biggest contribution of this work, since the performance of the forecasting algorithms is determined by the quality and volume of the data. Section 2.1 makes an introduction to NLP: the modern machine learning techniques that will allow us to embed the *garment characteristics*. Section 2.2 discusses what it truly means to embed fashion garments and how can we achieve it. We innovate from previous approaches by using text descriptors called *tags* in a way that families of garments and semantics are preserved. With the data structure ready that encoded the the design itself, we can now define mathematically what constitutes as a *similar* item in 2.2.2. Section 2.3, explores how can we get a good approximation for the forecasting target, the sales, when the *ground truth* is severely censored. Last, section 2.4 defines an *evaluation criteria* for the model, in this case the Weighted mean absolute percentage error (WMAPE), an error metric widely used in the retail business to asses forecast accuracy.

The core of the thesis can be found in chapter 3 where we present two algorithms developed to solve the size-curve problem. These two models are adapted from their baseline version in order for them to work under a fashion industry setting. In particular section 3.1 presents the kNN algorithm, a simple yet powerful model that seeks to replicate exactly the work of the designers and buyers. Section 3.2 presents a custom implementation of the CWR algorithm, adapted from the original presented in Baardman et al. (2017). This algorithm is more complex since it relies on automatically finding suitable clusters of items in order to create more accurate models. However, the clusters themselves also hold value since they serve as a set of *comparables* which the designers can also use. In particular, section 3.2.4 presents our innovation in using Classification and regression Trees (CART) as the underlying forecasting models due to the properties that lend perfectly well to the size-curve problem.

Chapter 4 presents some real-life results using a subset of Zara's data. Since the idea is to use the models in production, section 4.1 presents some examples that replicate the business as usual. Here we will see how does the model perform for the final user in terms of individual garments. Section 4.2, presents the aggregated results to stress-test the models, taking previous seasons as the training set and test on a more recent season to show the performance of the models at scale. Last, section 4.3 presents the practical details derived from the implementation of CART models in the CWR algorithm.

In the closing chapter 5 we conclude the thesis, evaluating the benefits and areas of opportunities of the algorithms, based on the real world data presented in the previous chapters. Section 5.1 explores an interesting phenomena that arises from the models, a trade-off between the observed similarity and accuracy since, ultimately, those two will be at odds. Section 5.2 opens the door for further improvements of the algorithms and discusses potential changes not explored before. In particular, in 5.2.3 we present a provocative idea: reversing the causality arrow presented in 1-1. I.e. if data is used to inform certain decisions, could data be used to inform the whole design process?

1.3 A brief introduction to Inditex / Zara

Inditex is a global fashion conglomerate headquartered in A Coruña, Spain. Founded in 1975 (Inditex 2023b), Inditex is one of the leading retailers for the industry with €27.7 millions in revenue in 2021. They have presence in over 96 countries, more than 6,477 stores accounting for $\approx$4.7M square meters. Zara, its flagship brand, is one of the most popular fashion brands worldwide accounts for almost 70% of Inditex group's revenues.(Inditex 2023a)

Zara has achieved such operational excellence that it allows it to be fast and nibble, reacting quickly to market trends. They label themselves as *Inconformismo Innovador* (innovative nonconformism), seeking continuous improvement, with the client always at the center. In order to achieve this, they are vertically integrated with direct feedback from both stores and suppliers. Their agile and omnichannel inventory distribution system is world-class, with a robust supply chain all across the world. They call their model *Sistema de gestión INTegrada de inventario* (SINT): integrated inventory management system in English, allowing them to fulfill orders both online and directly from stores (i.e. stores act as distribution hubs as well). This system allows them to improve efficiency, shorten delivery times and reduce their carbon footprint, as stated in Inditex.

In order to support SINT, the Inditex group is a pioneer in using big-data and taking an algorithmic approach to inform, end-to-end their operations, (Caro 2012). With the use of real-time data aiding their decision making process, Zara continues to be at the forefront of what is technologically feasible, Olabiyi (2021). An example of this is their modular, open, digital architecture *Inditex Open Platform*, Inditex. Hence, this is why collaborations between Zara and high level research universities like MIT has always been fostered, see Martinez Puppo (2019), Fridley (2018), and Kong (2015) for recent thesis examples.

1.3.1 Initial shipment decisions for new products

Mostly relevant for this thesis and the size curve problem is the *initial shipment decision* for any new product, based on the paper by Gallien et al. 2015. The core idea is that, for a retailer with products that have a short life cycle, the initial allocation to stores has an out-sized relevance into how will the garment perform (i.e. how the products will sell). This initial shipment, represents ≈50% of the total merchandise volume sent to stores, implying that re-stocks are far rarer and smaller. This decision point is represented by B in figure 1-1.[6] The number of new garments Zara distributed was 8,000 in 2011 (Caro 2012). We can now expect this number to have, at least doubled, therefore, making this problem even more relevant.

The main problem for this initial shipment is that there is little to no information regarding the demand for a new garment. And for Zara in particular, the sales of any given garment are greatest in the first couple of weeks after being introduced, with some clothes having a life-span of ≈6 weeks. Hence, there is an incentive to over-ship to prevent stockouts. However, if too much is sent, then Zara can not assure that inventory will be available in the most critical stores where demand was greatest. Last, if you take into account all the different geographies where Zara stores operate, it adds to the complexities of this decision.

Gallien et al. 2015 presents an initial solution to this problem based on a forecasting and

6. Expanding upon the initial operational diagram, figure 2-7 further, presents a more detailed explanation of the complete supply chain; complete with the re-stocks labeled D in the figure.

optimization procedure, showing that they can increase total average season sales by $\approx$2% and reduce the unsold units by approximately $\approx$4%. Their proposed solution is still based on a set of *human experts chosen comparables* in order to give an informed decision on the expected demand and therefore, to make an *educated forecast* for Q_i.[7] The output for their model is the size de-aggregated quantity sent to each store. Clearly, this creates an initial store-level *size-curve* if normalized.

The most relevant module in their pipeline is the *size optimization model* (section 3.5 on their paper), proposing an optimization formulation that de-aggregates the initial shipped quantity into sizes. Their formulation reflects closely Zara's granular store operations where store clerks can *remove* articles from the store floor if there are not available garments from a *major size* (e.g. small, medium or large) since those are the sizes customers are usually expecting. Last, in order to model this granular level demand they assume that the size-level demand follows a Poisson distribution.

Ultimately, their algorithmic pipeline was studied at depth for the elaboration of this thesis. However, since this thesis works with the *aggregated size-curve*, the noisy effects of the initial shipment have all been smoothed out.

Data Disclaimer

The data used to elaborate this thesis was collected from Zara's internal tooling during a required 6-month internship for theLGO program. Due to the sensitive nature of the data and in order to protect several trade secrets, individual results has been modified, data has been masked and highly aggregated. However, the high level ideas and overall conclusions hold true, which are reflective of the true model performance. In particular, the examples presented in chapter 4 are manufactured to exemplify how does the model can perform but do not represent the actual operation of the firm.

7. Note that this thesis seeks to automate this comparable matching procedure.

Chapter 2

The garment characteristics as data

In God we trust. All others must bring data.

W. Edwards Deming (Walton 1988)

2.1 A brief intro to NLP

In the past few years, the treatment of natural language, i.e. speech and text , has become an extremely active area of research: see Clark, Fox, and Lappin (2012) and Jurafsky and Martin (2009) for a comprehensive treatment. The core idea of NLP is to give computers the ability to *understand* language and to return something useful (Chollet 2017). In order to archive this, computers find statistical regularities in the training data and *learn* to recognize patterns and structures.

The idea is that the text itself has to be mapped into objects, usually vectors, that allow the machines to recognize its inherent meaning: its semantics. A piece of text called a *corpus* is first *standardized*,[1] then, it is split into atomic units called *tokens.* These tokens can be individual characters like punctuation symbols, words or even groups of words that hold inherent meaning like phrases. The collection of tokens for a given corpus is called its *vocabulary.* Usually, the vocabulary also includes tokens that act as catch-all for unknown

1. Making sure that the characters are consistent

words. The tokens are later *indexed*[2] and last, the indexed tokens are *embedded* into high dimensional unique vectors. Formally, an embedding can be thought as a map $e(t) : T \to \mathbb{R}^d$, that takes a token t and converts it to a vector in $\mathbb{R}^d$. A small example is show in figure 2-1. In this vector space, the words *tree* and *leaf* are closer together, whereas separated from *dad* and *mom*. However, since these latter two are similar in meaning, we can find them closer together as well.

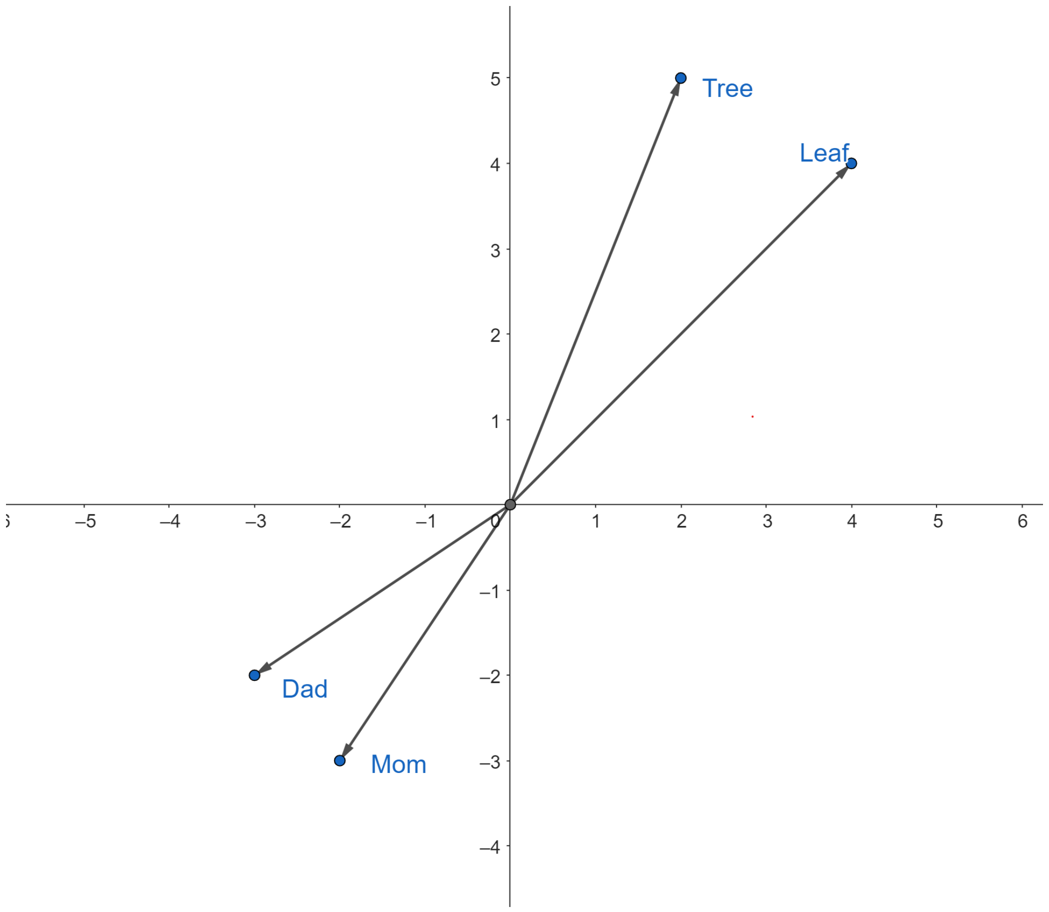


Figure 2-1: A simple embedding of words

This embedding step is crucial since there is a lot of structure, particularly the semantics, that needs to be preserved. Usually, not only the words themselves mean something, they mean something in the context of the text and therefore, their position is equally important. Given the above, word embeddings and processing is a very active area of research. Some of the most popular approaches are the following:

2. Assigned a numerical index

One-hot-encoding: the most simple embedding that simply maps every indexed token to a vector that has a 1 on its corresponding index. Therefore, these vectors are long and sparse, having n distrinct vectors per n unique words in the corpus. This embedding is usually used in *bag-of-words* models that care mostly about the *frequency* of the words. Albeit simple, as the corpus becomes large, the vectors also become increasingly large and lose a lot of the structure of the text.

Word2Vec from Mikolov, Chen, et al. (2013) and Mikolov, Sutskever, et al. (2013). A more modern approach that uses a two-layer neural network architecture to compute the word embeddings. The idea is to preserve the *similarity* between words and therefore some of its semantics. To achieve it, the algorithm proposes the *skip-gram* or *Continuous bag-of-words (CBOW)* models. The objective of this models can be roughly thought as a *fill-in-the-blanks* task; basically considering the word itself, and a sliding window of *context words* surrounding it. The resulting vectors are closer to each other if they share common contexts; words that are farther apart are more dissimilar, like in figure 2-1. Last, since these vector embeddings all have the same dimension, they define a vector space and we can perform regular vector algebra in them. A simple example is as follows: $e(\text{king}) - e(\text{man}) + e(\text{woman}) \approx e(\text{queen})$. I.e. the resulting vector from the operation of *king - man + woman* is very close to the vector that embeds the token *queen* which is what we would expect in the linguistically sense.

Global Vectors (GloVe) from Pennington, Socher, and Manning (2014) A similar approach to Word2Vec in the sense that it seeks to preserve some context. It is an unsupervised learning algorithm that calculates the word-to-word co-occurrence from a text corpus. This means that first, all statistics from words are calculated, i.e. the number of times word j appears in the presence of word i. Using theses statistics, conditional probabilities can be calculated. An example is that $P(\text{solid} \,|\, \text{ice})/P(\text{solid} \,|\, \text{steam}) = 8.9$ whereas $P(\text{gas} \,|\, \text{ice})/P(\text{gas} \,|\, \text{steam}) = 0.085$. I.e. the word *solid* appears more commonly in the presence of *ice* versus that of *steam*; the word *gas* behaves in the same way and therefore, the probability ratios reflects our intuition. By solving a least square problem that uses the aforementioned statistics, the vectors embeddings $e(t)$

are constructed, having the properties that we would expect.[3]

WordPiece and **Bidirectional Encoder Representations from Transformers (BERT)** from Wu et al. (2016) and Devlin et al. (2018) respectively. Although Word2Vec and GloVe take into account some local context based on statistics, the embeddings do not take into account global and positional context. For example, the vector e(running) would be the same in: *she is running a business* versus: *she is running a race*, where the two are relatively different. To distinguish between these two cases, WordPiece segments words into sub-word-level (e.g. *run* and *ing*. By starting at the character level, the most frequent combinations of those symbols are iteratively added into the total vocabulary, making it possible to represent an extremely rich collection of words an phrases, particularly useful for certain Asian languages. This level of granularity allows more information to be preserved and expressed. State-of-the-art language models like BERT use modern transformer architectures (Vaswani et al. 2017) to refine these embeddings even more, which ultimately allows them to achieve what appears to be human-level understanding of language.

The benefits of an embedding are clear, it allows for a representation of the language that computers can not only understand but *operate* with, while preserving the semantics. Seemingly understanding language like we do. Depending on the task the embedding can be chosen. Importantly, most models are already pre-trained and the vectors readily available. This is crucial since the training process can be computationally intensive for large collections of text.

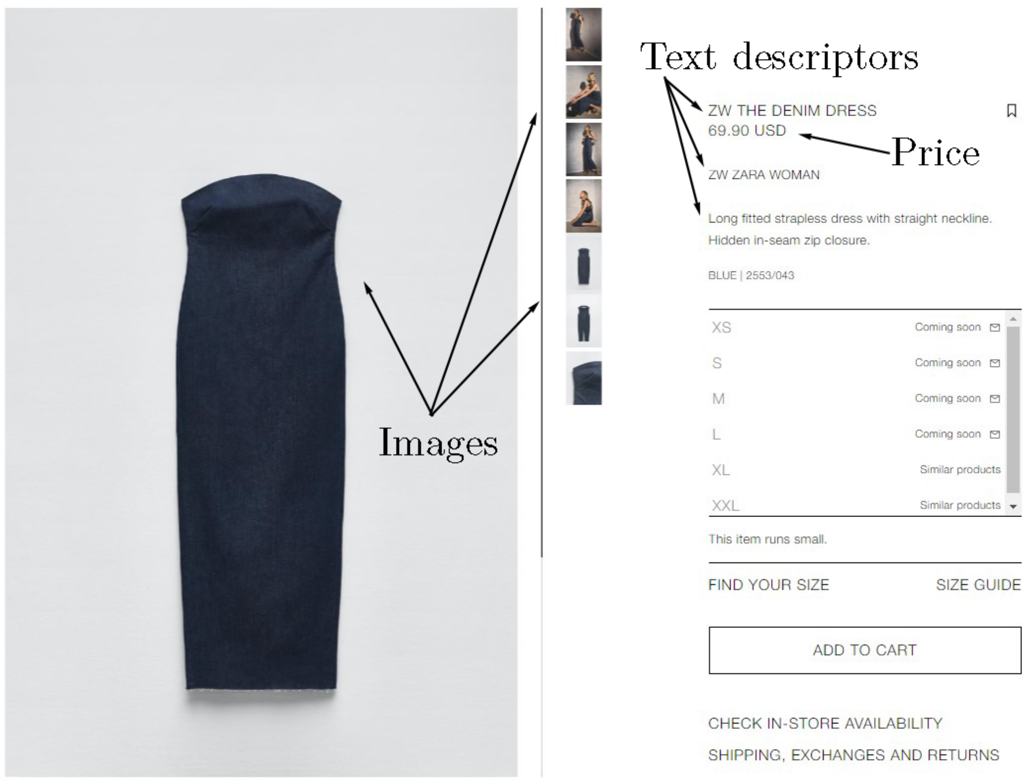
2.2 The design of a garment as data

The design of a garment: its color, the shape of the neck or even what kind of garment it is, like a pair of pants, a t-shirt, etc. seems completely opposite to quantifiable information: to numbers. Yet, for the human mind, we can select items we like based purely on those features. It follows, that the garment design hold *certain patterns* that, at least for our senses, are easily recognizable. We know how to differentiate between a sweater and a jacket

3. Both for GloVe and Word2Vec have the same number of dimensions: d=300.

or discern which items are similar to one-another. Could we quantify these characteristics and patterns and even define numerically what constitute as similar?

In the fashion industry, data from the garments is already stored in computers in the form of images and text descriptions. An example can be found on figure: 2-2. This particular ad of a dress, from Inditex (2023c), contains a lot of data: several images, text strings, fabric type and even some numerical information like price; all to aid the buyer into making a decision. All of that is data, an abstract representation of the garment.[4]



ZW THE DENIM DRESS
69.90 USD

ZW ZARA WOMAN

Long fitted strapless dress with straight neckline.
Hidden in-seam zip closure.

BLUE | 2553/043

This item runs small.

Figure 2-2: A dress ad in Zara's website

Given the advances in machine learning techniques, in both the image processing (Szeliski 2022) and natural language processing (Clark, Fox, and Lappin 2012), we can now unlock this data and discern mathematically those patterns that our eyes can detect. The images and text themselves are *representations* of the garments which then can be mapped to other representations; just like an sku code matches a physical product. These alternative

4. This data, besides being used to populate the online stores, also helps the firm's internal processes since human-friendly representations, for certain purposes, are easier to work with than something more abstract like Stock keeping unit (SKU) codes. If examined carefully, it appears also to be on figure 2-2.

representations are called *embeddings* in the data processing literature and can be made simpler, more abstract, or take different shapes; all depending on its required use and purpose. As an example, the embedding of a color image inside of a computer is a 3D array of numbers, which then can be projected on a screen. Therefore, a first idea, could be to use image processing techniques directly to find comparable garments. However, this would require the photos to be relatively standardized and it would take up a lot more computing power.[5] Since the goal of this thesis is to aid a particular forecasting process, we use a more compact representation of each garment.

Garment characteristic tags

The text representation of the garment *feels* smaller and more simple. However, it can fall short in its descriptive power unless treated correctly. To accurately represent a garment, each item needs to be attached to its *defining characteristics.* This representation is called *garment tags* and consist of a list of **feature-tag** pairs. E.g. A feature could be a *sleeve* and its tags are the descriptors of said feature: *none, short or long.* The combination returns three pairs: *sleeve-none*, *sleeve-short* & *sleeve-long.* A concrete illustration is shown in figure 2-3. Note that this framework is infinitely scalable since more feature can be added easily to be as descriptive as required. Once the garment tags have been set for the garments, we want an embedding of them that preserves all the information, language semantics included, available in the tags.

2.2.1 NLP garment tags embeddings

The goal is to create a compact and unique representation of every garment, ideally a vector. This representation will be an embedding based on its tags since the text on the tags already hold a condensed description of the garment. Since there are several embeddings that map text tokens t to vectors: $e(t) \in \mathbb{R}^d$, we apply the principles of NLP directly, as studied in section 2.1.

Despite having pre-trained models readily available, the garment embedding needs to be done in a more clever way since the tags already hold certain hierarchical structures. Take

5. The most advanced firms are already using the images themselves to find matches and aid the design process. See section 5.2.3 for a discussion.

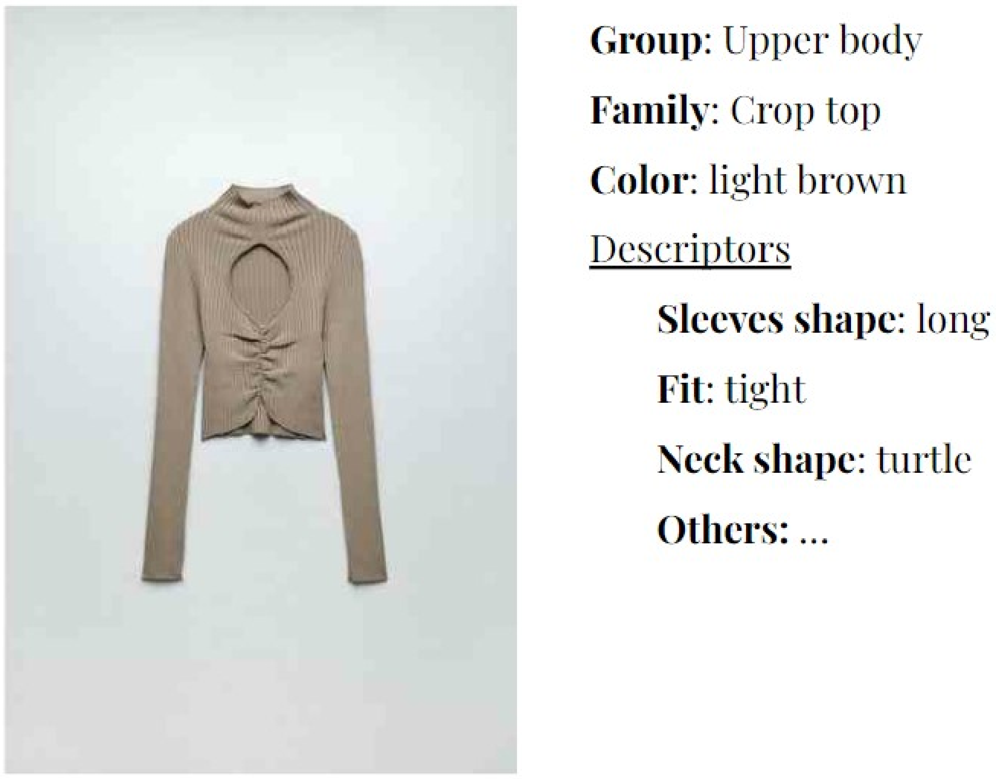


Figure 2-3: An example of garment characteristic tags

figure 2-3, both the *group* and *family* features, are defining clusters already: the family of *crop-tops* versus the family of *trousers*. Therefore, the desired embedding should preserve, in some sense, that bigger cluster information. To achieve this, we can think of *weighting*, those vectors $e(t_{\text{group}})$, $e(t_{\text{family}})$ more than the *descriptive tags* to preserve the group hierarchy. Similarly, order for the tags should be irrelevant since a blue sweater with long sleeves and tight fit, is conceptually very similar as a sweater with tight fit and long sleeves that has blue color.

Formally, assume that pre-trained embeddings are available[6] and therefore, $e(t) \in \mathbb{R}^d$ is defined $\forall\, t \in T$, where T is the set of all possible vocabulary words every possible tags.[7] We also assume that this embedding also preserves the semantics of the words well enough for

6. The choice of embedding is critical in the design of the algorithms, since the quality of the representation greatly impacts the quality of the forecast. Open-source libraries like spaCy or OpenNLP can be good starting point.

7. It is important to validate the assumption on T since some words in the fashion industry might not be as common in the training data-sets of the pipelines.

our purposes.

Definition 1. *For garment i with associated characteristic tags $T_i \subseteq T$, let $\boldsymbol{x}_i = \hat{e}_i(T_i) \in \mathbb{R}^d$ be it is corresponding **Garment-tags-embedding (gte)**, defined as:*

$$\begin{aligned}\boldsymbol{x}_i = \hat{e}_i(T_i) &= w_{group} \cdot \underbrace{e(t_{group}) + w_{family} \cdot e(t_{family}) + w_{color} \cdot e(t_{color})}_{group\ embeddings} \quad (2.1)\\ &+ \underbrace{w_{descriptor\text{-}1} \cdot e(t_{descriptor\text{-}1}) + \ldots + w_{descriptor\text{-}l} \cdot e(t_{descriptor\text{-}l})}_{granular\ descriptors}\\ &= \sum_{t \in T_i} w_t \cdot e(t)\end{aligned}$$

Where $\boldsymbol{w}$ set of non-negative weights that add up to one ($\boldsymbol{w}^t \cdot \mathbb{1} = 1$)

Therefore, the gte is just a weighted linear combination of all the characteristics tags vector embeddings. However, this simple representation defines a new *subspace* in $\mathbb{R}^d$ where every vector $\boldsymbol{x}_i$ is the representation of a garment itself. Albeit its simplicity, this definition has some desirable properties:

- Since the *group embeddings* are weighted more than the individual descriptors it ensures that the clothes *families* are already taken into account

- The representation is invariant under re-ordering since vectors in $\mathbb{R}^d$ commute

- As long as the collection of tags is unique to each garment, its gte should also be unique

- Since the individual components $e(t)$ already preserve semantics, the linear combination does not change that property and should also preserve the *garment semantics.* i.e. similar items are close together

- The computational implementation is extremely quickly since the pre-trained embeddings for the tokens is already computed and the vectors just need to be loaded on to memory

An alternative to using the pre-trained embeddings would be to develop a new algorithmic pipeline to find embedding that more closely match the firms need. However, since this framework is treating with text based tag descriptors and we want to preserve the semantics of the words themselves, a pre-trained model like GloVe works well. A further discussion of this design choice is presented in chapter 4.

Last, since we are not dealing with individual garments but several season of clothes, all with their respective tags. We can start thinking on a complete data-set of garments where $i \in N$. In this manner, each garment has its gte, allowing us to form a matrix $\boldsymbol{X} \in \mathbb{R}^{N \times d}$, where each row is an individual item.

2.2.2 The similarity metric

Having formally defined the gte $\boldsymbol{x}_i$, we can now test if it actually embeds the true characteristics of a garment; those that that humans can easily discern. In particular, we want to quantify *similarity*. Given that $\boldsymbol{x}_i$ is already a vector in $\mathbb{R}^d$ that preserves semantics, we can think of measuring how close are these vectors in space. Therefore, since closenes is similarity we can just measure the distance between the vectors

Definition 2. *The* ***similarity*** $sim(\cdot, \cdot) : \mathbb{R}^d \times \mathbb{R}^d \to [-1, 1]$ *between two vectors* $\boldsymbol{x}_i$ *and* $\boldsymbol{x}_j$ *is defined as:*

$$sim(\boldsymbol{x}_i, \boldsymbol{x}_j) = s_{i,j} = \frac{\boldsymbol{x}_i^t \cdot \boldsymbol{x}_j}{||\boldsymbol{x}_i||_2 \, ||\boldsymbol{x}_j||_2} \tag{2.2}$$

Where $|| \cdot ||_2$ *is the regular* l^2*-norm.*

Definition 2, is non other than the regular *cosine similarity*, widely used in the data analysis literature to measure how close two vectors are (Tan et al. 2019). The domain of s is $[-1, 1]$ where the extremes are only achieved if the vectors are either perfectly opposite or exactly the same. Hence, very similar garments will have similarities around .8 or .9 which can be also interpreted as a a percentage.

In practice, figure 2-4 shows an example of how does the similarity metric behaves given the $\hat{e}$. First the three items are sweaters and therefore, we expect their similarities to be at least positive. Garment A and C are very similar to our eyes, both are tight and have long sleeves, even the color is relatively closely matched, whereas that is not the case versus sweater B. However, due to the color sweater C and B does *feel* marginally more to each other than what A is to B. This examples shows that the gte does encodes what we humans consider as

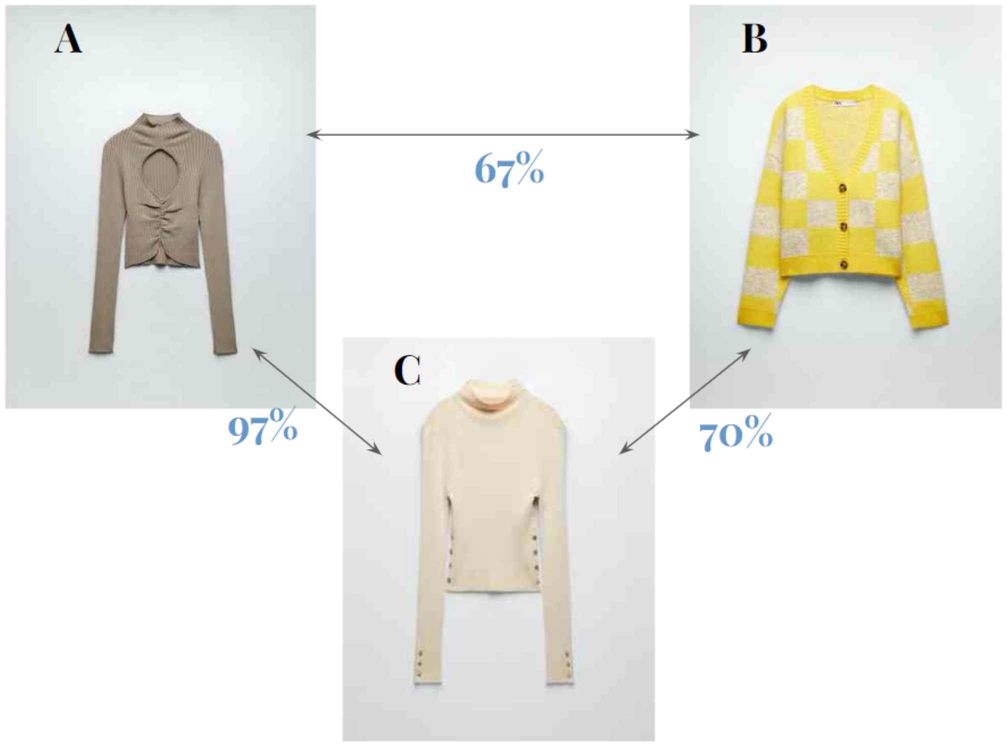


Figure 2-4: A numerical example of similarity between three garments

similar in garments and therefore, it is a valid representation.

Improving the representation to aid in the forecast

If the tags are rich and complete enough the gte should be able to encode all the information of garment itself. However, given the natural operation of a fashion firm itself, there are several other factors that directly influence the final outcome (sales). Some examples of such factors are its price (as seen in figure 2-2), the general economy, or even the week of sales where the garment is introduced. Hence, in order to improve the forecast, all of this data can also be included in the representation. Even a more descriptive emending of the color can be included.[8] Using this information, we should be able to enrich $\boldsymbol{X}$ further. In practice,

8. Red, green, blue color scale (RGB) or hue, saturation value color scale (HSV) are common vector representations that cover all possible colors. Both have three dimensions that encode different properties of colors.

this is achieved by appending more features (dimensions) to the $\boldsymbol{X}$ matrix.

However, despite the new information improving the forecast, there is a trade-off with the garment similarity. The non-aesthetic features (price, logistics, etc.) influence how the garments are sold but we measure how similar the items are simply by its aesthetic characteristics. If we extend this rationale and realized that garments are not sold in a vacuum, we will also observe mixed effects influencing sales. As an example, two very different items exposed in the the same store floor can influence the sales of the other one, making items complementary or supplementary. Hence, some additional information could improve the forecast, however, as viewed only as an extension to the gte it would make the smilarity metric worse. This inherent trade-off is explored at depth in section 5.1.

Another way of improving the gte representation for our purposes is to reduce d. This can be achieved by traditional dimensional reduction techniques such as Principal component analysis (PCA). See Mikolov, Sutskever, et al. (2013) and Tan et al. (2019) for a more comprehensive treatment. For certain experiments, we were able to capture over 95% of the variance, explained only with 50 principal components, see figure 2-5. When taking into that $d = 300$, it represents a substantial improvement in the compactness of the representation.[9] Last, since the individual features of $\boldsymbol{x}_i$ are already meaningless, there is no loss of interpretability by performing a PCA or any other dimensionality reduction.

2.3 Decisions influencing sales: approximating the *ground-truth*

Having encoded the garments themselves in $\boldsymbol{X}$, we now turn our attention to the outcome: its sales. In classical machine learning nomenclature, we have the regressors $\boldsymbol{X}$ but we are missing the target $\boldsymbol{Y}$. To achieve this, first we have to formalize certain concepts introduced in section 1.1.

Definition 3. *For any garment i with potential set of sizes S, a* ***size-curve distribution***

9. Even with less principal components, we were able to get the desired accuracy in the forecast. Similarly, the results presented in 4 are jointly robust to the choice of principal components.

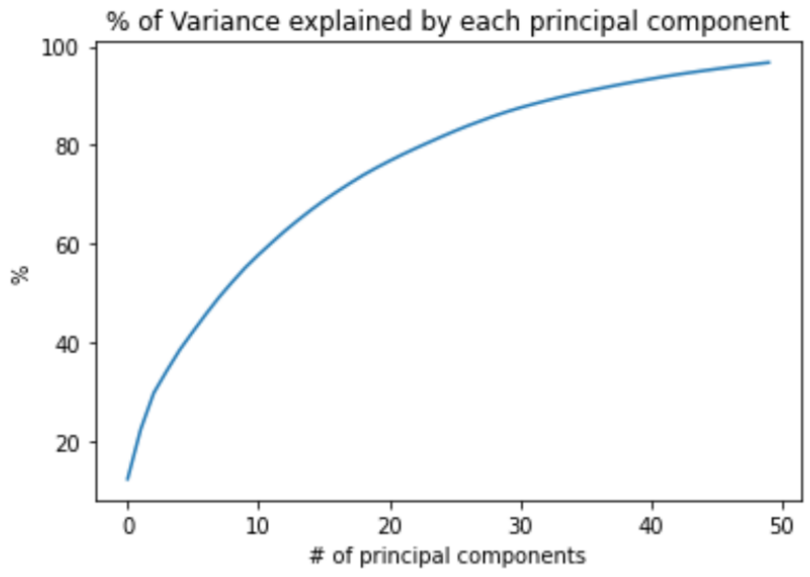


Figure 2-5: PCA analysis on the gtes contained in $\boldsymbol{X}$

is any set of variables $\pi_{i,s}$ *that satisfy the following properties:*

$$\pi_{i,s} \geq 0 \quad \forall\, s \in S \tag{2.3}$$

$$\sum_{s \in S} \pi_{i,s} = \boldsymbol{\pi}_i \cdot \mathbb{1} = 1 \tag{2.4}$$

Essentially, every $\pi_{i,s}$ is the proportion each garment has for that particular size s. For example, if $S = \{\text{small}, \text{medium}, \text{large}\}$ figure 1-2a is retrieved. If we multiply these proportions $\pi_{i,s}$ by the total order quantity Q_i, we get the number of items purchased for each size. The sales of the garments are defined in exactly the same way; utilizing $y_{i,s}$ to denote them. Since it is cumbersome to use the individual sizes s, we will mostly deal with the whole size-curve distribution. To represent this, we use bold lettering to denote the vector composed of all the sizes for any given garment i: where both $\boldsymbol{y}_i$, $\boldsymbol{\pi}_i \in [0, 1]^{|S|}$ are normalized due to identity (2.4). In a similar fashion, if we take all the available garments N we get the desired target matrix $\boldsymbol{Y}$ and $\boldsymbol{\Pi}$ respectively, both with N rows and $|S|$ columns.

The distinction between $\boldsymbol{y}_i$ and $\boldsymbol{\pi}_i$ is crucial. The former defines the *actual observed* sales in the store, whereas the latter, will denote the purchase decision. We assume that we have no control over $\boldsymbol{y}_i$ but the opposite occurs for $\boldsymbol{\pi}_i$. The causality arrow is clear: once the purchase decision $\boldsymbol{\pi}_i$ is made, garments are manufactured and distributed to stores. Then, actual sales are influenced by several other factors. Regardless of the intermediate logistics, *firms expect* for $\boldsymbol{\pi}_i$ to be a good predictor of $\boldsymbol{y}_i$. That is, that whatever they purchased will get sold without any underage (buying less) nor overage (buying more).

An example of the distinction of $\boldsymbol{y}_i$ and $\boldsymbol{\pi}_i$ can be found in figure 2-6. The *gray bars* represent the proportions bought ($\boldsymbol{\pi}_i$) whereas the *green dotted line*, the target, represent the actual sales during the season $\boldsymbol{y}_i$. For this particular example, there was a lot of overage of the small sizes and underage (in proportion) for the larger sizes. However, note also the *blue bars*, these represents an *alternative version* of $\boldsymbol{y}_i$, the new size-curve occurred after the in-season ones, where the clothes were in discount and therefore, potentially leading to more sales. In this alternative version, there was underage in the small sizes and overage in the larger ones. Therefore, *which version is the true target?*

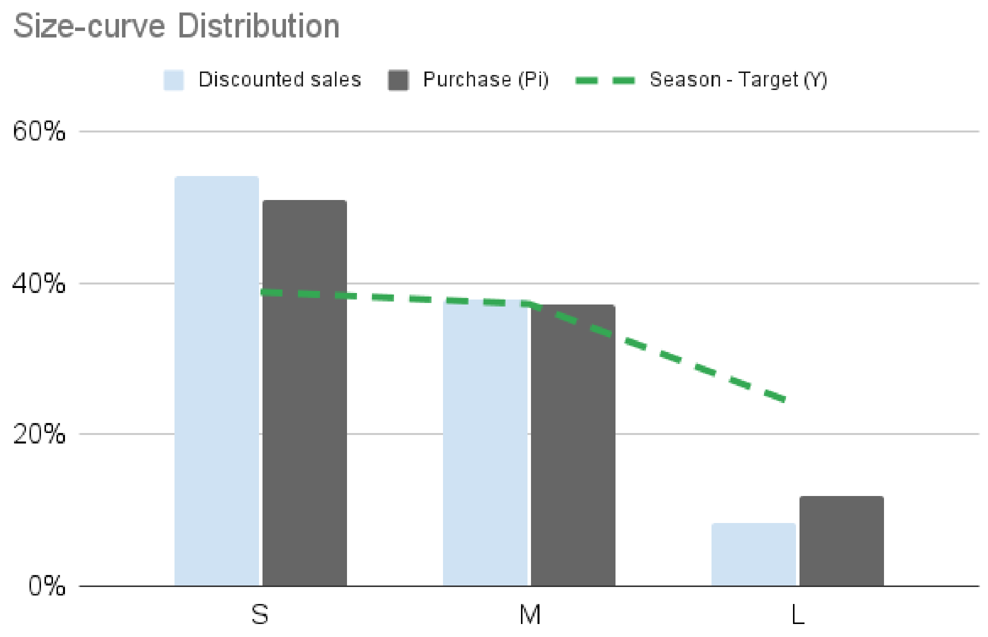


Figure 2-6: Difference between $\boldsymbol{\pi}_i$ and $\boldsymbol{y}_i$

Recalling the task at hand presented in 1.2 we want to train a model that, based on the garment characteristics embedded in $\boldsymbol{X}$, can help us make good predictions on the sales $\boldsymbol{Y}$. This will help the firm make better purchase decision. i.e. for a new garment denoted by o

we want to develop a *decision rule*:

$$\widehat{\boldsymbol{\pi}_o}\left(\boldsymbol{x}_o\right) \approx \boldsymbol{y}_o,$$

without having observed $\boldsymbol{y}_o$. However, in order to achieve this a we have to assume the *independence* between $\boldsymbol{\Pi}$ and $\boldsymbol{Y}$ which is hard to achieve.[10] This is immediately noted because sales are already *censored* by the purchase since we can not sell more items than those available. Additionally, there might be other interactions between the two like distribution to stores and mixed effects on garments. This problem will turn out to be hard to solve precisely and that $\boldsymbol{Y}$ can only be approximated. We call this issue the *ground truth* for the sales curve.[11]

2.3.1 Biases in Y

As mentioned before, finding the *true* $\boldsymbol{Y}$ is not trivial. With, the meaning of true being: *in a universe with infinite inventory, where the garment is perfectly exposed in the store, how much could the firm have sold of any given garment?*. Digging deeper into the issue, we extend figure 1-1 into figure: 2-7, to hone into the relationship between $\boldsymbol{\pi}_i$ and $\boldsymbol{y}_i$. After deciding on the purchase size-curve $\boldsymbol{\pi}_i$, firms have to decide how to send these garments to stores, usually, at a global level, decision point B. This is a nuanced decision that completely detaches the individual stores size curve from its original one $\boldsymbol{\pi}_i$. This happens because of regional variability's in the stores themselves. Albeit the whole world population might follow a given size distribution, at a local level, they do not. People from a given country might be larger than from another one. Therefore, shipping a fraction of Q_i to any given store, following the purchase size curve $\boldsymbol{\pi}_i$ would be a mistake. People in this issue is explored more at depth in Gallien et al. (2015) and briefly summarized in 1.3.1. When the garments arrive at the stores with a different size-curve, this curve also influences the final sale decision on the customer (decision point C). For example, certain firms might have different policies on which garments and sizes to show in the floor. Customers might not find the size they were looking for or similarly, they might not even go to a store because they know the particular brand does not have sizes that they find flattering. Additionally, if an item is selling well, firms might do

10. The sense of independence here is on a causal sense: one does not influencing the other
11. This problem has been greatly researched in the retail literature, aptly named named revenue management.

restocks on one or several stores, decision point D, influencing the final sales. Last, there

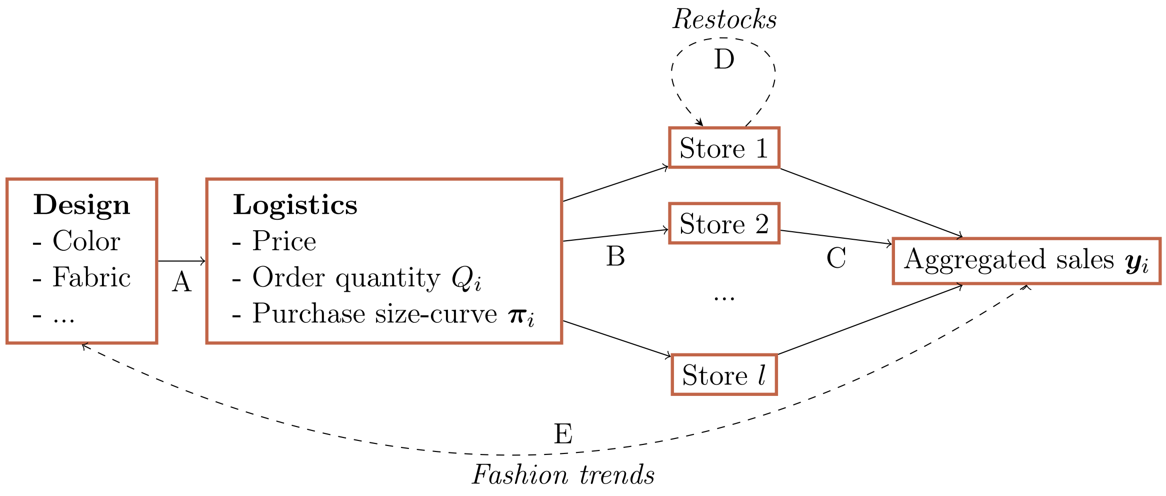


Figure 2-7: End-to-end fashion process

is the issue that seemingly reverses the *clear* causality arrow.[12] Since designers and buyers use the sales from a previous *comparable garment* to make a more *informed* decision on $\boldsymbol{\pi}_i$ for a new garment, the process becomes, in a way, a loop. A *chicken-and-egg* problem or a self-fulfilling prophecy that is not trivial to unravel without robust experimentation. These inherent biases in the relationship $\boldsymbol{\pi}_i \leftrightarrow \boldsymbol{y}_i$ need to be taken into account when creating a forecasting algorithm.

An approximation of Y

Note that even without the human biases, the true $\boldsymbol{y}_i$ can never be observed. However, a good approximation of the *true* $\boldsymbol{y}_i$ can be discerned. The idea is to use the *decoupling* from $\boldsymbol{\pi}_i$, happening when the clothes are distributed to the individual stores at decision point B. Then, if we leverage the fact that newer garments have more stock and are exposed in the store floor more than older skus, we can come up with a *good* approximation of $\boldsymbol{y}_i$ that should be relatively less biased. Last, we take only the full-price sales during the season since, as seen in figure 2-6, the discounted sales can be biased more by the price than by the true distribution. It is also in the interest of the firm to aim to sell their products at full price rather than a discount and therefore, the target is also seasonal full price.

12. This discussion extend on the issue introduced in section 1.1.

Let $\dot{y}^T_{i,s}$ be the sales of garment i and size s during its *first* T-days after the garment arrived to the store. Letting the normalization constant be $\dot{y}^T_i = \sum_{s\in S} \dot{y}^T_{i,s}$, we arrive to:

Definition 4. *The adjusted* ***aggregated sales size-curve distribution*** $\boldsymbol{y}_i$ *is:*

$$\boldsymbol{y}_i = \frac{1}{\sum_{s\in S} \dot{y}^T_{i,s}} \left[\dot{y}^T_{i,s_1}, \ldots, \dot{y}^T_{i,s_{|S|}}\right]^t \tag{2.5}$$

Clearly $0 \leq \dot{y}^T_i \leq Q_i$ since, even in the aggregate firms can not sell more than the total order quantity. But since the individual sales retain sufficient information and are only shown in its normalized version, we can observe both underage and overage. Similarly, it easy to see that this definition of $\boldsymbol{y}_i$ satisfies the properties of equations (2.3) and (2.4) and therefore, it is a valid size-curve.

2.4 The WMAPE metric

In order to define the problem completely, we need a metric to measure the performance of the algorithm itself and that of the buyers and designers. Hence, we want to be coherent with the metrics already used in practice.

Definition 5. *The* ***Weighted mean absolute percentage error (WMAPE))*** *is defined as follows:*

$$\underbrace{\frac{1}{n}\sum_{i=1}^{n} \underbrace{\left[\sum_{s=1}^{S} y_{i,s}\left|\frac{\pi_{i,s}}{y_{i,s}} - 1\right|\right]}_{\textit{individual WMAPE}}}_{\textit{total WMAPE}}, \tag{2.6}$$

In short, the WMAPE is a measure of the *error* in the forecast. The metric is widely used in the retail settings because of its simplicity and ease of understanding by less technical audiences. It works by averaging the errors on the forecast of individual garments. In turn, these individual WMAPE are calculated by adding the weighted percentage errors between the *forecast* $\boldsymbol{\pi}_i$ and the observed rate $\boldsymbol{y}_i$. Note that the weights $\boldsymbol{y}_i$ also sum to one due to (2.4). As an example, the WMAPE for the individual garment in the example shown in figure: 2-6 is $\approx 24\%$.

Advantages and disadvantages of the WMAPE

The importance of the WMAPE metric, is mostly derived from its intuitive business sense. Given that the absolute value is symmetric, we are considering that both underage and overage are equal which, depending on the application it can not always be ideal. Hence, as with all metric is has both advantages and disadvantages that is important to understand before deploying algorithms based in it. Some of the advantages include:

- It can be quickly be converted to revenue if the price of the garments is know
- Ease of understanding by senior leaders ship and widely used in retail settings
- Quick to calculate in computational terms

Similarly, some of its disadvantages are:

- The absolute value is not differentiable so it is harder to adapt to forecasting procedures
- Given the non-linear terms, if demand is close to zero it could lead to extremely large errors which can be misleading (Hyndman and Koehler 2006)
- Since the scale of $\boldsymbol{\pi}_i$ and $\boldsymbol{y}_i$ is already small, the WMAPE will also be small

Given the above, for our particular purposes, the WMAPE will mostly be used as a *benchmark* in comparing the forecast of our algorithms versus that of the buyers and designers.

Chapter 3

Custom algorithms for fashion foresting

Remember that all models are wrong; the practical question is how wrong do they have to be to not be useful

Box (1976)

Having defined data structures that mathematically encode fashion garments, we turn our attention to the forecasting algorithms themselves. A custom way of combining the ingredients defined in chapter 2 in a way to achieve the objective presented in section 1.2. In particular, we present two intuitive algorithms that replicate and automate the status-quo approach of that of the buyers and designers.

During this whole chapter is assumed that the *training data* set $i = 1, \dots, N$ is complete and valid, therefore, $\boldsymbol{X}$ and $\boldsymbol{Y}$ are all known.[1] The *test data*, for purposes of this chapter will only consist of one new garment that has not been sold yet and therefore we assume $\boldsymbol{y_o}$ is still unknown. We denote this new garment by the following sub-index o yielding: $\boldsymbol{x_o}$ and our forecast $\widehat{\boldsymbol{y}}_o$. Last, we denote by $\widehat{C} \subseteq [N]$ the set of indices that denote the set of

1. $\boldsymbol{\Pi}$ should be valid and known as well, however, it is not required for the algorithms since it is only used to benchmark the true value sales vs. the decision the buyers made. It can be thought as *another* solution to that of $\widehat{\boldsymbol{y}}_o$.

comparables used to forecast $\widehat{\boldsymbol{y}}_o$. This set of comparables is as important as the prediction itself.

3.1 The kNN algorithm

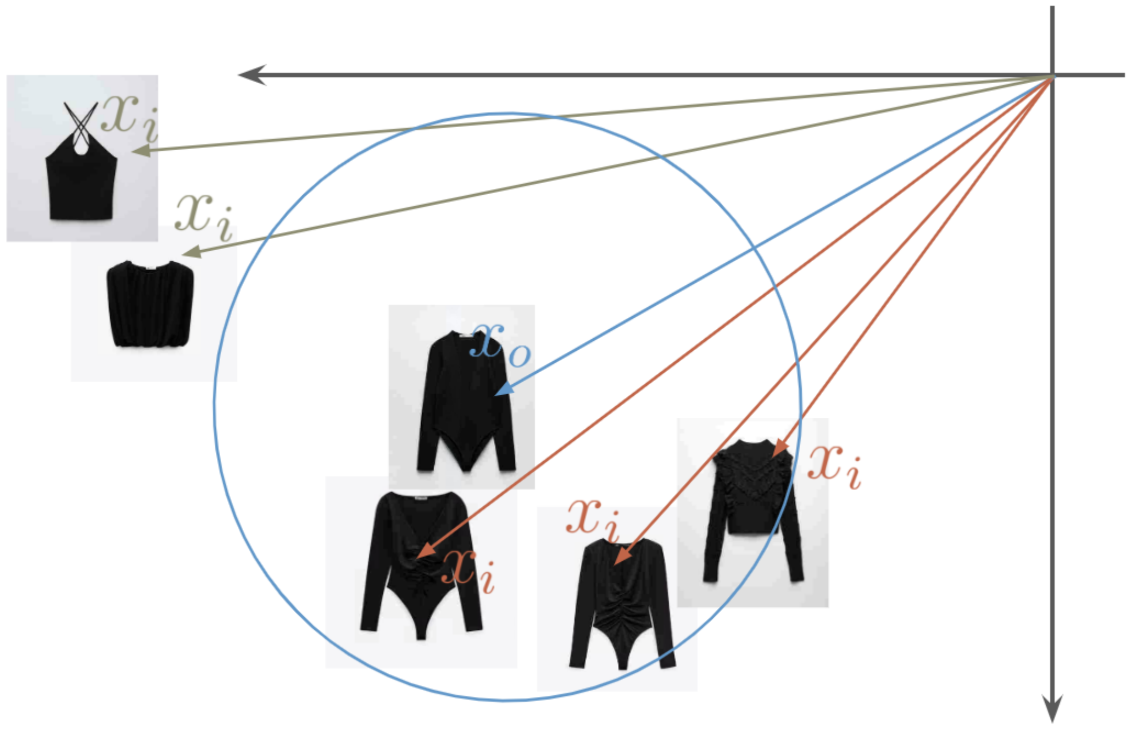


Figure 3-1: The k-nearest neighbors (kNN) algorithm visually

3.1.1 Technical discussion & properties

The nearest-neighbours methods are part of the *model-free* family since they are highly unstructured. Their simplicity is not prime for understanding natural relationships between the features and the outcomes like causal ones. However, they are extremely good prediction engines and at times they are great performers when real data is used, based on chapter 13 from Hastie, Tibshirani, and Friedman (2009).

For regression settings, they perform well for low-dimensional problems so it is important to keep in mind that a PCA might be needed to reduce d in $\boldsymbol{X}$. A critical aspect of these models, is to define what *close* means. Usually, euclidean distance (l^2-norm) is used, requiring the features to be standardized. For this thesis, and taking into account performance in low-dimensional constraints and a custom definition of *closeness*, our algorithm proposes using only the similarity metric defined in equation (2), collapsing everything into a single dimension. For certain applications, defining what constitutes as a *neighbour* can be extremely challenging.

Last, note that this is a *regression* kNN and not a classification one. Under this context, the predicted outcome is usually the average of the k closest responses, whereas in a classification setting, the class would be the *majority vote* amongst the neighbours. For regression problems, one can also think of *weighing* the outcomes of the k neighbours by their respective distance to the new garment. I.e. the closer the neighbour, the more weight its outcome has in the prediction. In this thesis, the simple average works well enough as show in section 4.2.1.

3.1.2 Custom algorithmic formulation

The kNN algorithm seems like a perfect match for needs of the size-curve problem. However, it had not been properly adopted due to the lack of a formal definition for similarity. Given the work described in chapter 2, it is now possible to do so. Therefore, in algorithm 1 on page 40, we present the full pseudo-code for the it. Notice how, by defining the data and therefore the similarity metric, we can easily retrieve both the set of comparables and the prediction. One of the few drawback of this model is that if d and N are large the computation time can be long. Last, note that this algorithm replicates exactly the status quo as long as the sets do contain similar garments and the tags are descriptive enough to describe the garments.

Algorithm 1: Custom implementation of the kNN algorithm for fashion firms with similarity metric

```
1 function similarity ($\boldsymbol{x}_i, \boldsymbol{x}_j$);
  Input  : Two gte vectors of different garments
  Output : $s_{i,j} \in [-1, 1]$
2 return $s_{i,j} \leftarrow \frac{\boldsymbol{x}_i \cdot \boldsymbol{x}_j}{||\boldsymbol{x}_i||_2 \, ||\boldsymbol{x}_j||_2}$
3 function fashion_knn ($\boldsymbol{x}_o, \boldsymbol{X}, \boldsymbol{Y}, k$);
  Input  : The gte of a new garment $\boldsymbol{x}_o$, a training set ($i = 1, \ldots, N$) with gtes ($\boldsymbol{X}$)
           and adjusted aggregated sales size-curve ($\boldsymbol{Y}$) of previous garments and an
           integer $k$ of the number of desired comparables
  Output : A set of indices $\widehat{C}$ of the comparables and the predicted size curve: $\boldsymbol{y}_o$
4 for $i$=1,...,$N$ do
5     calculate $s_{o,i}$ = similarity($\boldsymbol{x}_o, \boldsymbol{x}_i$)
6 end
7 $\widehat{C} \leftarrow \{i_{(1)}, i_{(2)}, \ldots, i_{(k)}\}$: the indices of the $k$ most similar garments measured via $s_{o,i}$
8 $\widehat{\boldsymbol{y}}_o = \frac{1}{k} \sum_{i \in \widehat{C}} \boldsymbol{y}_i$: averaging the $k$ size-curve vectors of which we know the true sales;
9 return ($\widehat{C}, \widehat{\boldsymbol{y}}_o$)
```

3.2 The CWR algorithm

The CWR algorithm is developed in the seminal paper Baardman et al. 2017 and it sits at the core of this thesis. The idea is also simple: *leverage* previous products (called as well comparables) to make better forecasts for any given new products. clearly, this idea also replicates the status-quo at a fashion firm. In order to achieve this, the model uses a novel iterative technique aptly named *cluster-while-regress*, improving upon previous *cluster-then-regress* techniques. Section 3.2.1 first does a literature review on similar techniques that leverage *clustering* to aid in the forecast. Section 3.2.2 then presents the general baseline algorithm that then will be adapted to the size-curve problem in section 3.2.3. In particular, the biggest adaptation is the use of CART as the underlying forecasting models. Last, section 3.2 presents the full algorithmic implementation of the model.

3.2.1 The use of clustering to aid prediction

The benefits of clustering to aid regression have been studied at depth since Späth 1979, starting with linear regression. The core idea is simple and it is still used in more modern approaches. Instead of having a single regression model for the whole data-set, we assign each of the $i = 1, \dots, N$ points to one of the k clusters. Each one of those clusters, has it is own regression model that should be more accurate that a single one that *over smooths* the noise in the data. This idea is particularly usefully in retail contexts where item sales present very noisy patterns. This happens due to seasonality, fast-responding markets and obsolescence trends. However, since there is already intrinsic *categories* that the items can be bundled into, there is implicit information that can be used to aid in the prediction.

At first glance, the idea should be simple to execute, replicating what is already done by firms: take past similar products and use them to predict. More formally, k *initial clusters* are created by clustering algorithms like k-means clustering (k-means), then fit k regressions. This useful approach is also presented in more modern papers like Trivedi, Pardos, and Heffernan 2015. Then, the regressions are combined in a way that aids the forecast or simply use the regression model of the cluster where the new item belongs to. In a way, this technique can be considered a *pre-processing* of the data since it is a two step approach. Hence, it is commonly refereed to as *cluster-then-regress*. However, this technique has several drawbacks since it is not leveraging the whole dataset and the finer sub-structures present once the initial clusters and regressions have been fitted. Since the clustering algorithm and the regressions have different objective function some of the predictive power is lost. Similarly, bias in the clustering algorithms, solely based on $\boldsymbol{X} \in \mathbb{R}^{N \times d}$ that do not necessarily aid the forecast which is based on $\boldsymbol{y} \in \mathbb{R}^{N}$ and hence, it might yield worse accuracy in the aggregate.

Therefore, the goal is to combine both the *clustering* and the *regression* into a single objective. Späth 1979 presents a simple heuristic that reassigns each i to a new cluster in each iteration of the algorithm, as long as it reduces the total prediction error. In more recent times, given the growth of available computing power and proliferation of optimization techniques, approaches likes the ones presented Bertsimas and Shioda (2007) propose solving a compact mixed-integer formulation and Park et al. (2017) propose a *column competition* method. However, this last paper solves it only for the linear regression case. The novelty of Baardman et al. (2017) is that they expand the heuristic of Späth (1979) for any sort of regression. In addition, they present a technique to produce *out-of-sample* estimation. These two ideas complement each other and open the door for a lot more application, particularly in retail settings where the idea is to introduce new items.

3.2.2 General mathematical formulation

Before the custom algorithm is presented in section 3.2, we present the baseline general CWR model from Baardman et al. (2017).[2] Besides the know training data structures: the data matrix $\boldsymbol{X}$ and the response $\boldsymbol{y}$ and a fixed number of desired clusters $K \in \mathbb{Z}^{+}$, denote by $z_{i,k} \in \{0,1\} \quad \forall i = 1, \ldots, N;\ k = 1, \ldots, K$ the binary variable that takes the value 1 if garment i belongs to cluster k and 0 otherwise.[3] In a high level overview, the model can be split into three distinct phases:

Phase 1 - training clusters and regression models. This phase assigns each item i to one of the K clusters and fits a regression model f_k for each one. We then re-assign

2. We deviate slightly from their notation and nomenclature due to exposition purposes and to ensure consistency across this thesis. However, the algorithm is almost the same.

3. Clearly, we can summarize all $z_{i,k}$ in it is matrix form: $\boldsymbol{Z} \in \{0,1\}^{N \times K}$, with the restriction that row-wise, there is only a single entry with a one. This would mean a clear mapping from items to clusters. I.e. each garment is assigned to one and only one cluster.

the points and re-fit the models until convergence is achieved.

Phase 2 - training the cluster classifier. Once the final models $\hat{f}_k(\cdot)$ and clusters assignments $\widehat{\boldsymbol{Z}}$ are set, we train a regular multiple classification model $H(\cdot)$ that seeks to predict which cluster k does every item i belongs to.

Phase 3 - forecasting for a new product. Last, when a new garment $\boldsymbol{x}_o \in \mathbb{R}^d$ is presented, we can use the trained classifier $H(\cdot)$ to get a predicted cluster $\hat{k}$ to where the new garment belongs. By using $\hat{f}_{\hat{k}}$, the assignment yields both $\widehat{\boldsymbol{y}}_o$ and $\widehat{C}$.

The core and stages of the CWR algorithm are now presented in detail.

Phase 1 - training clusters and regression models

The cluster training can be stated as the following optimization problem.

Definition 6. *The* ***CWR problem*** *(P) is:*

$$\min_{z_{i,k},\, f_k} \quad \sum_{i=1}^{N} L\left(y_i, \sum_{k=1}^{K} z_{i,k} \cdot f_k(\boldsymbol{x}_i)\right) + \lambda\, R(f_1, \ldots, f_K) \tag{3.1}$$

$$s.t. \sum_{k=1}^{K} z_{i,k} = 1 \quad \forall i = 1, \ldots, N \tag{3.2}$$

$$z_{i,k} \in \{0, 1\} \quad \forall i = 1, \ldots, N;\; k = 1, \ldots, K. \tag{3.3}$$

Where the R is a regularization function and $\lambda \geq 0$ a penalty term. The core idea of (P) is that every item is assigned only to one cluster, eq. (3.2) and (3.3), creating a *partition* of the data. Every cluster has its own model f_k that seeks to minimize the regularized objective (3.1), given suitable loss function L and regularizer R for the models f.

If the true cluster were know, (P) would be easy to solve since it would become a cluster-then-regress model. However, this formulation of (P) requires the simultaneous assignment of clusters (an integer program) and fitting f_k (potentially non-linear models) to optimality which is computationally hard to do. Particularly when either N scales or more complex forms are selected for f_k like random forests or neural networks. Hence, the time to solve can be intractable.

To circumvent this issue, a simple iterative heuristic is proposed, akin to that of Späth (1979). Given a fixed penalty λ and number of clusters K. First, take an initial assignment of clusters $\hat{z}_{ik}^{(0)}$ for all the points and clusters: $\widehat{\boldsymbol{Z}}^{(0)}$. This initial assignment can be made either randomly or via any traditional clustering method.[4] Then, fit an initial version of the models $\hat{f}_k^{(1)}$. Since the K models can also be used to make *predictions* on the points that do not belong to its training set, we can calculate K loses for each of the points. Therefore, we can re-assign the points to that cluster k where the k-th loss is minimized, i.e. we update $\hat{z}_{ik}^{(1)}$. With the new assignment done, the new models $\hat{f}_k^{(2)}$ are trained, new assignments are now done, and we solve iteratively. The algorithm can terminate due to three reasons:

1. The max number of iterations T is reached.
2. Points do not change clusters anymore: $\widehat{\boldsymbol{Z}}^{(t+1)} = \widehat{\boldsymbol{Z}}^{(t)}$
3. The change in the loss function is small enough (i.e. reaching a suitable *convergence* threshold): $|\hat{L}^{(t+1)} - \hat{L}^{(t)}| \leq \epsilon$, with $\epsilon > 0$.

Diagram 3-2 presents a graphical depiction of the phase 1 iterative algorithm. The time to compute mostly depends on the functional form of f_k since it is fitting K separate models those. However, given modern libraries, it can be solved efficiently. Last, we know that

$$\begin{tikzcd}
\widehat{Z}^{(0)} \arrow[r, "t \leftarrow 1"] & \hat{f}^{(t)} \arrow[rr, bend left, "\text{re-assign points based on losses}"] & & \widehat{Z}^{(t)} \arrow[ll, bend left, "t \leftarrow t+1"] \arrow[r, "\text{exit}"] & \widehat{\boldsymbol{Z}}^{(t^*)}, \hat{f}^{(t^*)}
\end{tikzcd}$$

Figure 3-2: Phase 1 of the CWR algorithm

this algorithm converges to a locally optimal solution in polynomial time. A complete proof is given in Baardman et al. (2017). The core idea is that the loss function in every iteration always decreases: $L\left(\widehat{\boldsymbol{Z}}^{(t+1)}, \hat{f}_k^{(t+1)}\right) \leq L\left(\widehat{\boldsymbol{Z}}^{(t)}, \hat{f}_k^{(t)}\right)$. This happens because of

4. A further discussion of the benefits of each approach are presented in section 4.3.

the re-assignment of points to a new model to which the loss is minimized. Similarly, the proof presents one of the conditions of convergence: when the points do not change clusters anymore.

Phase 2 - training the cluster classifier

Having the final trained models: $\hat{f}^{(t^*)}$ and cluster assignment of points: $\widehat{\boldsymbol{Z}}^{(t^*)}$, we now seek to build a *classification* model, to assign each item i to clusters. We are assuming that the intrinsic characteristics present in $\boldsymbol{X}$ can also be used to predict its final cluster. To achieve so, first take a mapping of the clusters to a vector that indicates which cluster is each point assigned to: $\widehat{\boldsymbol{Z}}^{(t^*)} \mapsto \boldsymbol{\kappa} \in \{1, \ldots, K\}^n$. Basically, each entry κ_i is the column of $\widehat{\boldsymbol{Z}}^{(t^*)}$ where the 1 was located. Armed with both $\boldsymbol{X}$ and $\boldsymbol{\kappa}$, we can now easily train a classification model $H(\cdot)$, like a multinomial logistic regression, support vector machines (SVM), or even a classification kNN. This process creates a mapping from the product space where $\boldsymbol{X}$ lives, directly to a particular cluster $\hat{k}$, which can then be converted into a forecast via the already trained models $\hat{f}_{\hat{k}}^{(t)}$.

Phase 3 - forecasting for a new product

For a new garment, we leverage the mapping described above. Take a new garment $\boldsymbol{x}_o$, make a cluster prediction $\hat{k} = \widehat{H}(\boldsymbol{x}_o)$ and then forecast via the assigned model $\hat{y} = \hat{f}_{\hat{k}}^{(t^*)}(\boldsymbol{x}_o)$. Which completes the process and the algorithm. Besides this forecasting procedure, the paper shows an additional way of making the final forecast: a weighted average of all the forecasts. This weighted average, is based on the assigned probabilities derived by the classification model $\hat{H}$. However, for this custom implementation since the set of comparables $\widehat{C}$ is as equally important, we select the former way of making the assignments. Further discussion of this design choice in the next section.

3.2.3 CWR in the context of the size-curve problem

Given the objectives stated in section 1.2, adapting the CWR algorithm to the size-curve problem requires making some changes to the original formulation presented in the last section 3.2.2. However, the core idea is the same: leveraging previous comparables to forecast the size-curve of a new garment. Conceptually, this can be thought as closing the loop on figure 2-7, iteratively using the observed sales of previous garments to train forecasting models.

The biggest difference comes from the fact that $\boldsymbol{y}_i$ is a vector and not a single number estimate like the original implementation of the CWR algorithm and most machine learning models. This means that for the size-curve problem, we are seeking to do a *multiple* prediction of $|S|$ numbers at once. Additionally, the output vector needs to be a distribution that satisfying equations (2.3) and (2.4). I.e. all quantities in $\widehat{\boldsymbol{y}}_o$ need to be positive and linked together. In order to achieve this, the biggest modification is that we have to loose the flexibility on f and set the regression models to be CART model. Particularly, multi-output CART models. The mathematical considerations are explored at depth in the following section 3.2.4 and its practical implications in 4.3.

Another important consideration for the size-curve problem is the data that constitutes *similarity* is not necessarily the best data to train the classifier. I.e. the data ($\boldsymbol{X}$) required to train the cluster assignments and models on phase 1 ($\widehat{\boldsymbol{Z}}^{(t)}$, $\hat{f}_k^{(t)}$) can be different from the data used to train the classifier $\widehat{H}$ on phase 2. This happens because in phase 1 the objective is to seek *accuracy*, measured via the loss function L. However, for this particular aplication, we make the assumtion that similar items will lead to better accuracy, therefore, the data in $\boldsymbol{X}$ needs to reflect it so. To train the classifier H, we could think of adding data that is less *similar* in a fashion context, but that can help the accuracy of the classifier. In practice, this distinction does not change the algorithm much. However, it just changes how it is implemented algorithmic, without impacting the quality of the comparables nor both the accuracy of the regression and classification. A further discussion of this is presented in section 5.1.

Last as mentioned in the previous section, Baardman et al. (2017) presents two potential forecasting techniques to calculate $\widehat{\boldsymbol{y}}_o$ on phase 3: weighted average of the k clusters versus a single cluster assignment $\hat{k}$. However, since we are interested in both $\widehat{\boldsymbol{y}}_o$ and the *comparables*, we need to be able to concretely return the set $\widehat{C}$. I.e. the indices i's in the training set that are most similar to the new garment o. Therefore, this forces us to use the pure cluster assignment mapping, where the new garment o belongs to one and only one cluster $\hat{k}$, one leaf (due to the use of CART models) and therefore, one set $\widehat{C}$, leaving no room for ambiguity.[5]

3.2.4 The use of CART model for the size-curve problem

As mentioned before, the biggest modification of the CWR model for the size-curve problem is the use of CART models for the regressions on phase 1 $f_k \quad \forall k = 1, \ldots, K$. Albeit seemingly restrictive at first, these models have a lot of useful properties that are leveraged in order to achieve satisfactory results.

In general, the CART models are a kind of tree based models that partition the space $\mathbb{R}^d$ into rectangles, usually in a greedy manner. Once the rectangles are fitted, the algorithm fits a simple model to each one, section 9.2 of Hastie, Tibshirani, and Friedman 2009. For regression trees, the simple model is usually a constant that is calculated by taking the average of all the points that land on each leaf. For classification, the resulting class is the one that most points take. CART models are non-parametric and can be easily interpreted as a set of sequential binary *rules*. Since these rules are based on *linear cuts* on the feature space, tree based models (particularly shallow trees) retain a lot of interpretability. They are simple yet very powerful models.

However, they also come with disadvantages. First, despite its non-parametric nature, CART models have several hyper-parameters that need to be validated. These parameters control the shape of the tree and its complexity. Parameters like *max depth*, *number of minimal points on a leaf* or even the *splitting criteria*, can be modified. As such, these parameters need to be treated with care. Paradoxically, the simplicity of the trees can lead to very complex structures that loose all interpretability and perfectly overfit in the training data which is not desired. Particularly when we seek to do out-of-sample predictions. Last, CART models are not robust and small perturbation in the training data can yield very different structures. Depending on the application this might be a big issue. An in depth discussion fine-tuning the CART models for the size-curve problem is presented in section 4.3.

5. A combined approach could be used: take the weighted average of the clusters to provide $\widehat{\boldsymbol{y}}_o$ and use the single cluster assignment to provide $\widehat{C}$. However, this would come at the explainability of the model which is critical for real-world applications.

Multi-output CART models preserve the structure of $\boldsymbol{y}_i$

Since $\boldsymbol{y}_i$ is a vector of length $|S|$, we have to use models that allow for multi-output predictions. In traditional machine learning literature, when models deal with this issue, the most common approach is to make $|S|$ different models, one for each output.[6]. Based on this, an initial idea is to use the total sales for each size: $\dot{y}_{i,s}$ as presented in section 2.3.1 rather than its normalized version. The algorithm would then normalize the predictions to satisfy the constraints of a size-curve. Alternatively and to save computational time, since the total amount Q_i is know, we can use the reduced degrees of freedom and fit $|S|-1$ models yielding $|S|-1$ predictions. With a know Q_i we can complete the last size and then normalize. However, in practice, these two approaches did not yield good, having low out-of-sample accuracy.[7]

This issues happens because, albeit a simple approach, the elements of $\boldsymbol{y}_i$ are tightly linked together and have a particular structure. Hence, the rationale to use multi-output cart models: a single model is capable of predicting simultaneously all outputs. This does not only help with the lower training time (vs. fitting $|S|$ models) but it also usually leads to better training accuracy. Most computational packages where the multi-output CART is implemented, the splitting criteria takes into account the reduction in all the outputs simultaneously and not only on one, which is extremely desirable in the context of the size-curve problem.

Besides the benefits on the training procedure, the prediction for a multi-output cart model has the following closed form:

$$\widehat{\boldsymbol{y}}_o = \frac{1}{|\widehat{C}_o|} \sum_{i \in \widehat{C}_o} \boldsymbol{y}_i, \tag{3.4}$$

with $\widehat{C}_o$ the set of points that fall on the *predicted leaf* for the new item o. I.e. the predicted size-curve is just a simple average of the comparables size-curves. This means, that once the new garment is assigned to a cluster $\hat{k}$ (with associated CART model $\hat{f}_{\hat{k}}^{(t)}$),

6. See section 3.2.4 in Hastie, Tibshirani, and Friedman 2009

7. Under this approach, several models were tested: multi-output linear regression, least absolute shrinkage and selection operator (LASSO) regression, and even ensemble models but none had the sufficient predictive power to have good accuracy on the out-of-sample normalized WMAPE metric.

the item is then assigned to one an only one leaf, yielding a set of comparables $\widehat{C}_o$ and a corresponding interpretable prediction; precisely what was required from the algorithm.[8] More importantly, since equation (3.4) is the simple average of other size-curves that already satisfy the positively constraint, equation (2.3), is trivially satisfied. However, equation (2.4), the last one remaining to ensure that $\widehat{\boldsymbol{y}}_o$ is a size curve is not as satisfied trivially. Hence we present a simple proof of this property.

Proof: validity of the size-curve prediction. *For any multi-output CART model where the predicted response* $\widehat{\boldsymbol{y}}_o$ *has form as in equation* (3.4)*: it satisfies both the positivity and the normalization constraint* (2.3) *and* (2.4) *respectively. Therefore,* $\widehat{\boldsymbol{y}}_o$ *is a valid size-curve.*

Let $\boldsymbol{Y} \in \mathbb{R}^{N\times|S|}$ be the know aggregated sales size-curves matrix and $f(\cdot)$ any multi-output regression CART model. Without of generality, take any ending leaf of f. By properties or trees, every leaf (ending node), has at least one point of the training set $i_{(1)} \in \{1, \ldots, N\} = [N]$. Let C be the set of all points in this leaf: $C = \left\{i_{(1)}, \ldots, i_{(l)}\right\}$ which is a subset of all the training points $[N]$, with associated predicted response:

$$\boldsymbol{y} = \frac{1}{l}\sum_{i\in C} \boldsymbol{y}_i.$$

Since each $\boldsymbol{y}_i$ is a size curve that is normalized ($\sum_{s\in S} y_{i,s} = \boldsymbol{y}_i \cdot \mathbb{1} = 1$), the following holds:

$$\begin{aligned}
\boldsymbol{y}\cdot\mathbb{1} &= \left(\frac{1}{l}\sum_{i\in C}\boldsymbol{y}_i\right)\cdot\mathbb{1} \\
&= \frac{1}{l}\sum_{i\in C}\left(\boldsymbol{y}_i\cdot\mathbb{1}\right) \\
&= \frac{1}{l}\sum_{i\in C}\left(\cancelto{1}{\sum_{s\in S} y_{i,s}}\right) \\
&= \frac{1}{l}\sum_{i\in C} 1 \\
&= \frac{|C|}{l} = \frac{l}{l} = 1.
\end{aligned}$$

8. Contrast this with a model like a random forest of a neural network that, although perhaps a bit more accurate, there is not a clearly defined set of comparables.

Therefore, regardless of which leaf the new point lands, the average of its points continue to be a distribution, satisfying equation (2.4). If follows that $\widehat{\boldsymbol{y}}_o$ is a valid size-curve.[9] Q.E.D.

One final benefit of the multi-output CART models for the size-curve problem is that, in practice is that it created balanced clusters. This meant that the average number of items per clusters N/K was relatively stable and no one cluster became an *sink*. These balanced clusters can be interpreted as *families* of garments, preserving the traditional super-structures of clothes. It was not rare for all pants were in the same cluster and therefore, in its own tree with internal sub-structures. Contrasting this with other regression models, most tended to have low computational stability, assign most if not all of the points to a given cluster k, and making the CWR model impossible to run. Due to all of the benefits above, the multi-output CART models were selected as the perfect functional form for the custom implementation of the CWR model.

3.2.5 Algorithmic formulation

Taking into account all the particularities of the size-curve problem, a final version of the custom CWR algorithm 2 is presented following the three phases described in section 3.2.2. Note that in this version of the algorithm, the original matrix $\boldsymbol{X}$ can be divided in the data used for regression and classification $\boldsymbol{X}_{\text{reg}}$ and $\boldsymbol{X}_{\text{class}}$ respectively, as mentinoed in section 3.2.3. This to capture the relevant information, of the the gte, similarity features and any other additional information that helps get better accuracy for both kinds of models.

Final considerations and comparison versus the kNN

Although the K parameter in the CWR algorithm sounds analogous to that of the kNN, it is relevant to note that they are very different in both concept and implementation. First, for the CWR on average the number of items in a cluster will be $\dfrac{N}{K}$ which is usually larger than the $\hat{k} \in \{2, 3, 5\}$ that the users want as number of comparables. Alternatively, if we choose a K that makes $\dfrac{N}{K} \approx 5$ (usually implying a large K) the individual models would have extremely low predictive power and would overfit. Similarly, the algorithm could run into computational issues where one of the K clusters does not have any points and therefore,

9. A corollary of this proof is that the average of any subset of rows of $\boldsymbol{Y}$ is also a distribution. A generalization is that the average of distributions is also a distribution itself which make intuitive sense.

no regression can be fitted. To avoid this, it is critical to understand the difference in the K parameter for each one of the models. For the CWR algorithm, K is the number of clusters. These can be conceptually considered as *super-groups.* The comparables for an new item are then given by the training-items present the final leaf of the CART model for that said cluster.[10] For the kNN algorithm, the k does represent the number of comparables desired by the user.

10. This number is usually controlled by the *minimum number of points required on a leaf* on the CART implementation.

Algorithm 2: Custom implementation of the CWR algorithm for fashion firms

```
1  function phase 1 ($\boldsymbol{X}_{\text{reg}}, \boldsymbol{Y}, K, f_k, T$);
   Input  : The training set $\boldsymbol{X}_{\text{reg}}$ and $\boldsymbol{Y}$, containing the gte and additional data for
            known garments, plus the adjusted aggregated sales size-curve. An integer
            $K$ of the total number of clusters, $f_k$, a set of CART models to be trained
            (with previously specified hyper-parameters) and $T$ the max number of
            iterations.
   Output : $\widehat{\boldsymbol{Z}}^{(t^*)}$ the final cluster assignment matrix and $\hat{f}_k^{(t)}$, the trained CART
            models.
2  $\widehat{\boldsymbol{Z}}^{(0)} \leftarrow$ initial assignment of garments to the $K$ clusters;
3  $t \leftarrow 1$;
4  while $t \leq T$ and convergence not attained do
5      for $k \leftarrow 1$ to $K$ do
6          fit $\hat{f}_k^{(t)}$ on the set of points assigned by $\widehat{\boldsymbol{Z}}^{(t-1)}$;
7      end
8      Calculate loss for every point into every model $k$ ;
9      Calculate total loss $L$ and check loss convergence;
10     Update $\widehat{\boldsymbol{Z}}^{(t)}$ re-assign points to clusters with lowest loss ;
11     Check cluster convergence;
12     $t \leftarrow t + 1$;
13 end
14 return $\left(\widehat{\boldsymbol{Z}}^{(t^*)}, \hat{f}_k^{(t^*)}\right)$ function phase 2 ($\boldsymbol{X}_{\text{class}}, Z, H$);
   Input  : The training data for the classifier: $\boldsymbol{X}_{\text{class}}$. A cluster assignment matrix: $Z$
            and the functional form of a classification model $H$
   Output : $\widehat{H}$: a trained classification model that maps garments to clusters
15 $\kappa \leftarrow$ remap of $Z$;
16 Train $\widehat{H}$ based on $\boldsymbol{X}_{\text{class}}$ and $\kappa$ ;
17 return $\widehat{H}$
18 function phase 3 ($\boldsymbol{x}_o$, $\hat{f}_k^{(t^*)}$, $\widehat{H}$);
   Input  : $\boldsymbol{x}_o$ the embedding of a new garment. The phase 1 trained CART models
            $\hat{f}_k^{(t^*)}$ and a phase 2 trained classifier: $\widehat{H}$
   Output : $\widehat{C}$ a set of comparable garments and $\widehat{\boldsymbol{y}}_o$ the size curve prediction
19 $\hat{k} \leftarrow \widehat{H}(\boldsymbol{x}_o^{\text{class}})$ ;
20 $\widehat{\boldsymbol{y}}_o \leftarrow \hat{f}_{\hat{k}}^{(t^*)}(\boldsymbol{x}_o^{\text{reg}})$ ;
21 $\widehat{C} \leftarrow$ the set of indices of the leaf where $\boldsymbol{x}_o^{\text{reg}}$ belongs in $\hat{f}_{\hat{k}}^{(t^*)}$ ;
22 return ($\widehat{C}$, $\widehat{\boldsymbol{y}}_o$) ;
```

Chapter 4

The models in action: out-of-season experiments & results

Having completed both the data structures and algorithms, we can now test the model performance with some real world fashion data. More importantly, we want to benchmark the algorithms vs. the status-quo described in 1.1.

Data preparation

Given how the algorithms are constructed, we utilize two previous fashion seasons (as described in section 1.3.1) to build a complete training set. This is also seeks to take into account the natural yearly seasonality of the garments, i.e. there are different summer and winter styles and clothes. For the test set, we utilize a truncated version of a more recent fashion season, different from the two used in the train set. This ensures that the test garments are more *modern and fashionable* than the training set; replicating a fashion firm operations. This leaves us with approximately $N_{\text{train}} \approx 3500$ and $N_{\text{test}} \approx 1,000$ garments respectively. It is critical to note that this dataset is not the fully available dataset and this is just a small subset of available garments.[1]. In particular, we only select clothes where

1. This fact is due to data completion and privacy constraints. It will be revisited under the conclusions section

the available sizes S are *small, medium* and *large*.[2] Under this dataset, the WMAPE of the buyers, i.e. that one obtained by comparing the purchase decision: $\mathbf{\Pi}$ and the observed sales: $\boldsymbol{Y}$, and therefore the metric to improve is approximately 15%.[3] This is an impressive result that speaks to how good the firm buyers and designers are. The construction of $\boldsymbol{X}$ and hence, its number of dimensions (d), is ad-hoc to depending on the problem at hand. Since adding columns and performing PCA on the gtes is a design choice, it is something that will be explained further in section 4.2.2. Last, to be consistent with modern data-science frameworks, we developed a set of experiments to *cross-validate* the hyper-parameters like K.

4.1 Individual garment predictions

In practice, the models are built to predict at the individual garment level, aiding the process of the designers and buyers when they have to make a decision on the size curve. I.e the prediction $\widehat{\boldsymbol{y}}_o$ and the set of comparables $\widehat{C}$ work as a suggestion for $\boldsymbol{\pi}_o$. However, in order to test the performance of the algorithm, we have to look at how accurate the model is in the aggregate. Therefore, we present and analyze both of those cases.

In several of our experiments, the predictions yielded by the model outperformed the accuracy (measured by the WMAPE) of the buyers. Below, we present a series of examples to illustrate the model capabilities with the real dataset.

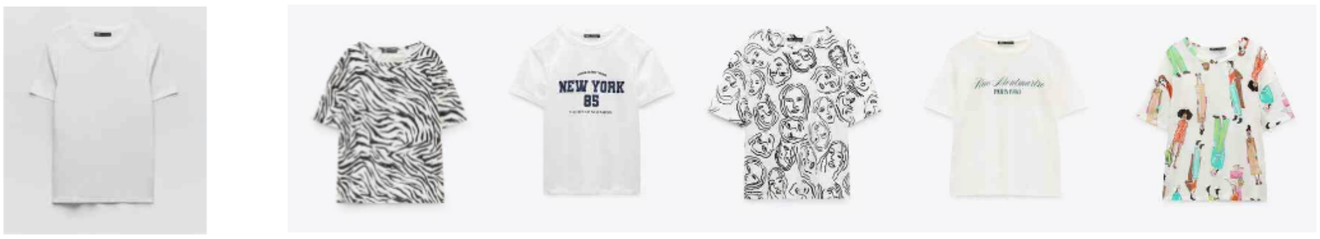
4.1.1 Example 1 - Similarity and accuracy in line

First, observe figure 4-1 and table 4.1. This example shows what happens when the model works exactly like we would expect. The set of comparables in 4-1b appears *similar* enough to the new garment 4-1. This is confirmed on an average similarity metric of 87%. Last, what is even better, is that the predicted size curve $\widehat{\boldsymbol{y}}_o$ only presents an error of **10%** versus the true sales $\boldsymbol{y}_o$, which represents a -57% reduction versus the buyers error of **23%**. This is a clear victory for the model, not only in terms of accuracy but also in terms of automation

2. The model can be generalized to larger sets of S, however, increasing the dimensions of the prediction, can greatly hinder the performance of the model.

3. As mentioned in section 1.3, this number is directional and doesn't represent the actual operations of the firm.

since it is very easy to find even more or less comparables by modifying the k parameter in either algorithm.



(a) o: new garment (b) $\widehat{C}$: set of comparables

Figure 4-1: Example 1

$\boldsymbol{k}$	5			
Avg. Similarity	87%			
S	Small	Medium	Large	**WMAPE**
True $\boldsymbol{y}_o$	48%	40%	12%	-
Purchase $\boldsymbol{\pi}_o$	60%	35%	6%	**23%**
Predicted $\widehat{\boldsymbol{y}}_o$	45%	38%	17%	**10%**

Table 4.1: Example 1 - numerical results

The only concern we can discern from this example regards the similarity of the garments themselves. For the untrained eye the t-shirts might appear similar. However, an expert designer could argue that they are very different: some have stripes, patterns or text, whereas the new one does not. This effect comes from two potential sources:

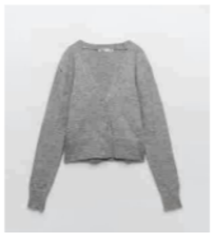
First, it could happen that for this particular dataset, these are the most common t-shirts available in the previous season. Either due to lack of data, or due to newer fashion trends. I.e. this new garment is indeed completely different and unique. This is not unlikely because in theory, that is how fashion should work.

Second, the tags used are not descriptive enough to capture these minor details: fabric patterns, global color scheme, etc. Hence, the gte are good but not good enough for the algorithms to cluster the same garments the designers would be thinking of.

These considerations are crucial and will be discussed further in section 5.1.

4.1.2 Example 2 - Great comparables, worse accuracy

Similarly, it is important to notice that algorithms are not infallible and can incur in worse accuracy than the buyers. For example, take figure 4-2 and table 4.2. Here, we present a new garment whose set of comparables $\widehat{C}$ is very similar: 93% by the avg. similarity metric. However, when we look at the WMAPE, we see that the model prediction **5%** is worse than that of the buyers with only a **2%**. Note that on this example $k = 3$ which might help explain why the errors are larger. Normally, the larger the comparable set, the better the performance.[4]


(a) o: new garment

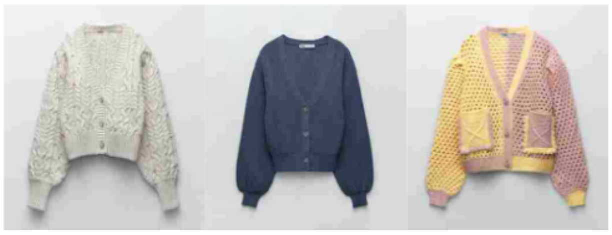

(b) $\widehat{C}$: set of comparables

Figure 4-2: Example 2

$\boldsymbol{k}$	3			
Avg. Similarity	93%			
S	Small	Medium	Large	**WMAPE**
True $\boldsymbol{y}_o$	53%	33%	14%	-
Purchase $\boldsymbol{\pi}_o$	51%	34%	15%	**2%**
Predicted $\widehat{\boldsymbol{y}}_o$	55%	32%	13%	**5%**

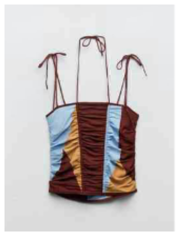
Table 4.2: Example 2 - numerical results

4.1.3 Example 3 - Better accuracy but interesting comparables

Last, we present an example where the accuracy and similarity are reversed: see figure 4-3 and table 4.3. Here, the prediction error $\widehat{\boldsymbol{y}}_o$ is only **7%** versus that of the buyers at **10%** which is clearly an improvement. However, upon closer inspection of the garments they do have big differences, even though at first glance they could appear similar. This is also reflected in a sub-optimal similarly metric of 76%. The rationale behind this phenomena is that, the CWR algorithm is optimizing for forecast accuracy which might not be optimal for comparability. The KNN operates in the opposite direction. Therefore, the rationale

4. This fact is discussed at length in section 5.1.

must be something similar to that of in example 1 4.1.1: these are simply the most similar garments to 4-3a in a previous season and the new one has just never been seen before.


(a) o: new garment

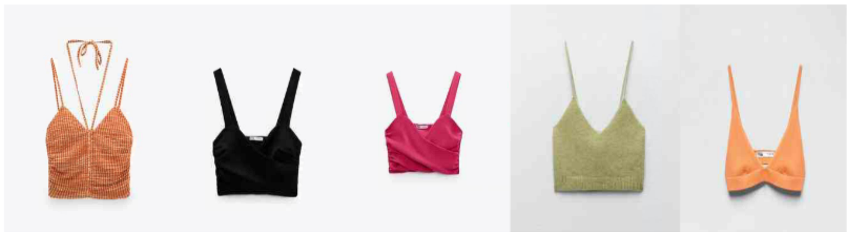

(b) $\widehat{C}$: set of comparables

Figure 4-3: Example 3

$\boldsymbol{k}$	5			
Avg. Similarity	76%			
S	Small	Medium	Large	**WMAPE**
True $\boldsymbol{y}_o$	55%	30%	15%	-
Purchase $\boldsymbol{\pi}_o$	60%	25%	15%	**10%**
Predicted $\widehat{\boldsymbol{y}}_o$	59%	28%	14%	**7%**

Table 4.3: Example 1 - numerical results

4.2 Aggregated prediction results

Regardless of individual results, what we want to validate is the model at scale, utilizing the whole test set. Because, if the algorithms are deployed and the predictions used, we would never observe the *true* value of $\boldsymbol{y}_i$, it would be already biased by $\widehat{\boldsymbol{y}}_o = \boldsymbol{\pi}_o$. Exactly in the same way it is already biased by $\boldsymbol{\pi}_i$. Hence, in order to start *trusting* these predictions, the average error needs to be validated and benchmarked versus the current purchase decision of 15%.

4.2.1 kNN algorithm results

In aggregate, the results for the kNN algorithm are presented in figure 4-4 for several values of k. Since this is the only parameter and the kNN algorithm is simple, the results from the model are straightforward. From the figure we can observe that this particular algorithm does not beat the established purchase decision since it reaches a lowest WMAPE error of 22%. Best results are achieved with a larger number of comparables k which makes sense that usually, the higher the number of data-points we use to forecast, the more precise the accuracy. In this algorithm the model has less error in 31% of N_{test} garments.

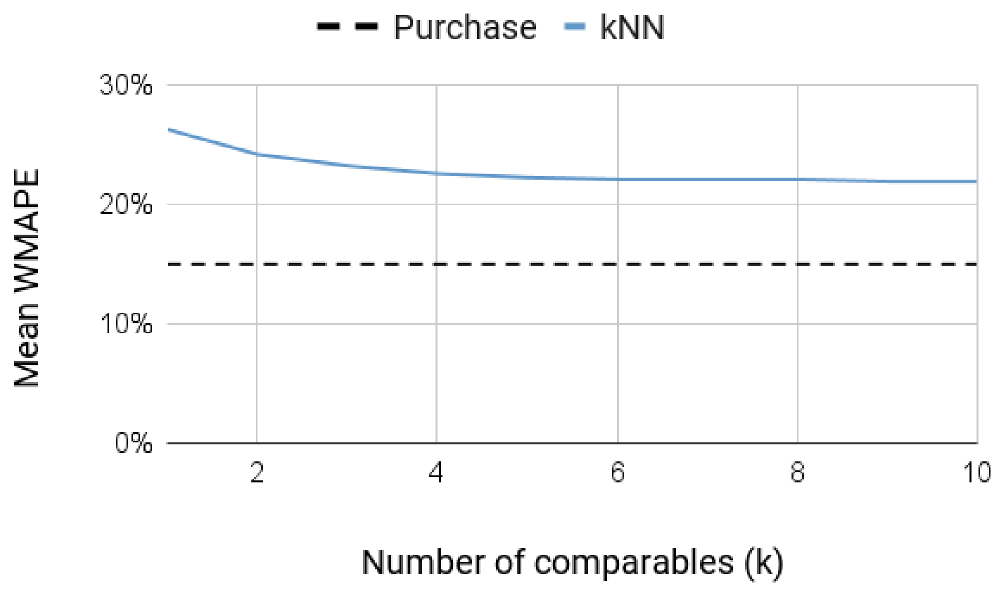


Figure 4-4: Test WMAPE of the kNN algorithm

4.2.2 CWR algorithm results

Since the CWR algorithm has several hyper-parameters including K in number of model-clusters, we can not directly make the comparison with 4-4. However, these hyper-parameters, particularly those inside of the CART models were all cross-validated thoroughly, reducing the over-fitting in the train set.[5]. In figure 4-5, we can observe several realizations of the CWR algorithm, for three modified data-sets, while we control for K and pre-processing on $\boldsymbol{X}$. The individual parameters of each realization are not critical for the interpretation of the results because they can vary depending on K, $\boldsymbol{X}$ and the random initialization algorithm. However, via cross-validation, we select the most relevant parameters for each point on the lines. From the figure, we observe that none of the realizations can match the target despite having very low errors already. In particular, using all the data and $K = 7$ returns the best WMAPE of $\approx 24\%$. Similarly, it is interesting to note that augmenting the number of clusters K does not really change the performance and actually, seems to hinder it. In chapter 5 we explore the implications of this and propose factors to improve the accuracy to match that of the buyers.

5. Despite the benefits of the multi-output CART models described in 3.2.4, these models tend to overfit and it is hard to generalize outside of the train-set like we are trying here.

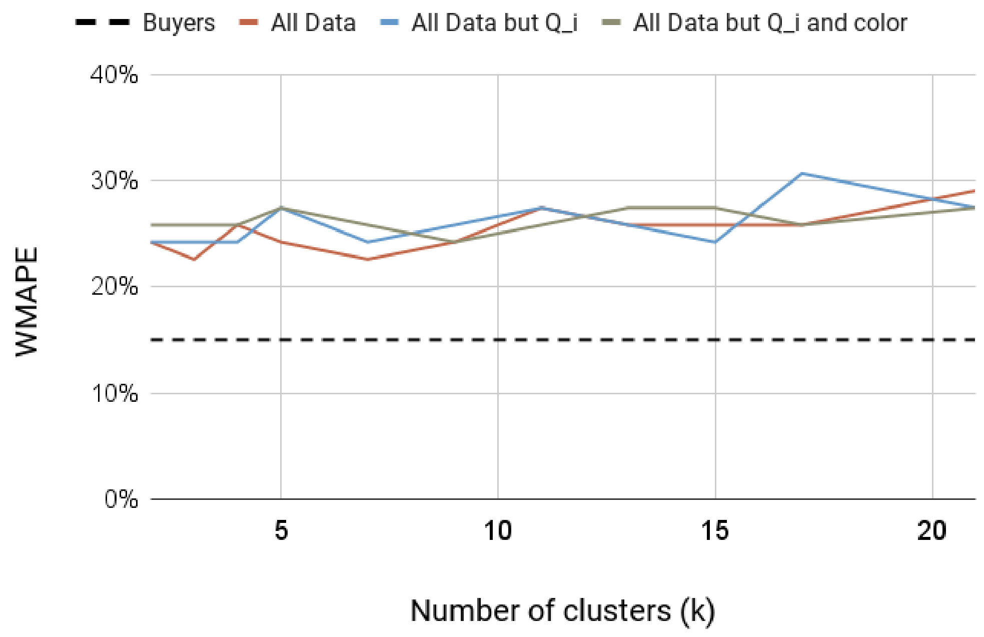


Figure 4-5: Test WMAPE for CWR algorithm

4.3 Fine tuning the CART models - finding the best parameters

Given the complexities of the algorithms there are hyper-parameters to fine-tune. Particularly for the CWR with internal CART model there are over ten hyper-parameters to cross-validate. This made it hard to evaluate the models and interpret some of the results. However, upon many runs a few trends are clear, which allow us to make some recommendations.

Number of PCA components. Whether to use PCA on the gtes is a strong design choice. It mostly depend on the size of N since it will greatly harm the speed at which the algorithms are executed. For the kNN algorithm, there really is not any difference, however, for the CWR there is. The underlying CART models have a lot of trouble working with a large d. In addition, we have to take into account the ratio N/K since it is on average the number of points each cluster will have. If $d \approx N/K$ classical statistical learning theory, Hastie, Tibshirani, and Friedman (2009), tells us that the models do not have a lot of prediction power. Note also that a lot of the dimensions of the gte can be reduced to a few PCA components as seen in figure 2-5. Last, in figure 4-6, we can see how for a fixed K, how does the accuracy of the classifier changes when we add more or less PCA components. There is a clear improvement once we have *enough* components, but more than those, can start behaving like confounding variables.[6] Given all of this evidence, we recommend using PCA and testing at which number of components the CWR algorithm performs best.

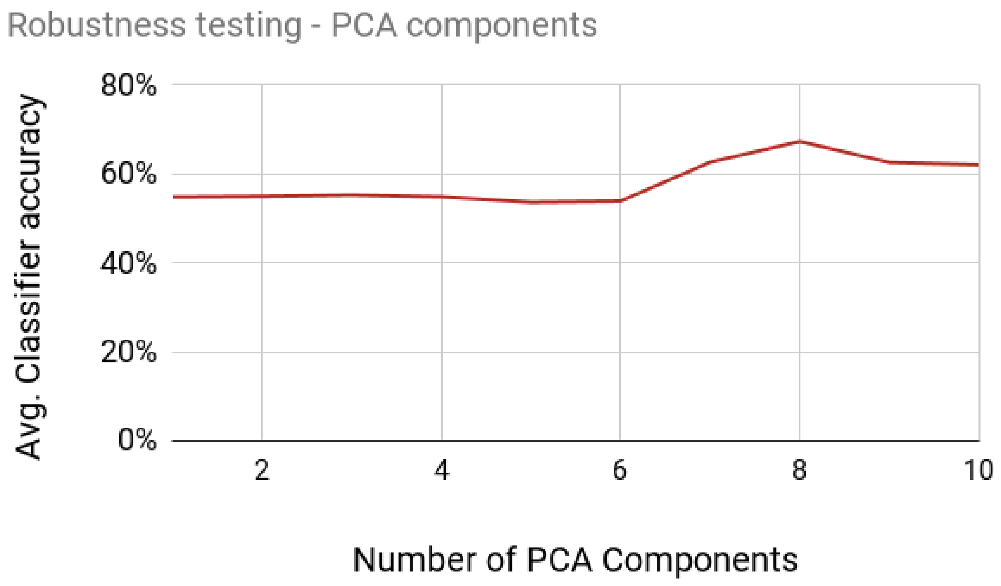


Figure 4-6: PCA robustness test in CWR algorithm

Adding additional data to $\boldsymbol{X}$. At first glance, adding additional data like a different color embedding, price, or even the total quantity ordered Q_i, would appears to be beneficial for the prediction. However, we have to be cognizant of several things: first, adding Q_i completely overrides all other features, including the gtes since all size curves are just a distribution of Q_i. This fact is easy to see in the context of traditional linear regression. Let one of the components of $\boldsymbol{x}_i$ be equal to Q_i, therefore, if $\boldsymbol{y}_i = \boldsymbol{\beta}^t \boldsymbol{x}_i$, then $\hat{\boldsymbol{\beta}} \propto Q_i$, i.e. the estimate is just a re-scaling of Q_i. Albeit the predictions would be accurate, the similarity between garments would be completely lost. This behaviour is not informative nor generalizable in the context of the problem. Something similar happens when adding additional, non-gte data to $\boldsymbol{X}$, there is indeed additional information to improve the forecast, see the different lines on figure 4-5, however, there is also a trade-off with similarity which is understandable. We recommend adding features like price and color since those can help discern between similar items, but treat any other garment level features with care. Else, seek to embed them within the gte. The custom algorithm presented, allows to use different features of $\boldsymbol{X}$ for both the phase 1 and phase 2.

6. This test performs 6 random assignments for the train/test set on the classifier.

CWR - number of clusters k. Different from the kNN algorithm, the CWR algorithm does not appear to be that sensible to changes in k. Given the use of an internal CART models for the regression, we can view the k as an initial *multi-branching* of a tree; a super-level of sorts. Hence, if the correct hyper-parameters for the regression CART is used, the effect of k becomes less relevant. However, if we compare figures 4-5 with figure 4-7, we observe that, although the accuracy on the forecast (measured by WMAPE) is not as sensible to K, there is an opposite effect on the classification, measured by a regular accuracy metric.[7] On the classification step, the performance of the algorithm diminishes inversely proportional to K. The more clusters, the worse the assignment to the correct cluster, irregardless of the dataset used. Despite this, there is indeed variability captured by the classification models which can be seen when we compare the black region with the colored lines. The black region represents a *naive* classification, flipping a coin and assigning to a cluster. These counter-intuitive

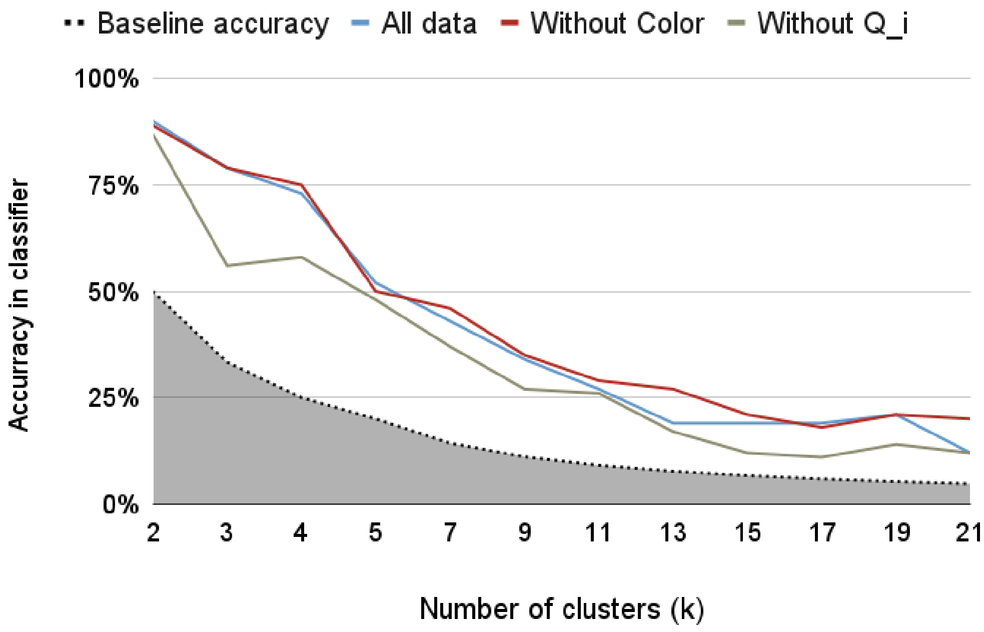


Figure 4-7: PCA robustness test in CWR algorithm

observations should be explored further since they does diminish the effectiveness of the CWR model. Improvements in the effectiveness of k would greatly improve the WMAPE and therefore perform better at the objective. A potential solution would be to transition

7. Number of correctly classified observations divided by N. Alternative, more balanced classification accuracy metrics were used and presented the same patterns.

from CART models to something more robust like a Random Forest (RF) model in the classification part, particularly since the objective of phase 2 is simply to maximize the classification accuracy.

Choice of initial cluster assignments. Although not a hyper-parameter by itself, the choice of cluster initialization is a design decision. Based on our experiments, it does not affect much the final predictions a lot and the algorithm converges all the time. That means that, given this dataset, initializing the clusters via either k-means or *randomly* led to similar forecasting results. However, this choice can led to different final comparables which is not ideal for replicability purposes. In Baardman et al. (2017), the authors suggest to use randomized initial clusters, because it can lead to globally optimal solutions. A random assignment also prevents a situation where the initial clusters are imbalanced and one ends up capturing most of the data points. If that were the case, the model would throw an error and it would terminate which is not ideal. Last, another aspect to consider is that, since the CWR models is not explicitly optimizing for *closeness* most of the times, cluster quality metrics like *silhouette* are hindered and usually does not converge.[8] Therefore, we recommend testing these two (or more approaches) before deployment of the models in production.

CART hyper-parameters for regression and classification. Despite the benefits of the CART models described in 3.2.4, these models have a lot of hyper-parameters that need to be cross-validated. The three most important for this work are: *max tree depth*, *min number of data-points to split* and *min data-points on a node.* With just these three parameters, CART can describe very complex structures in the data. However, note that a very large tree will overfit the data with very few observations in each leaf, while a small tree will not capture all the available variability (Hastie, Tibshirani, and Friedman 2009). Note that these parameters and considerations work in the same way for both the *internal* k regression trees, as well as for the *external* cluster-classification tree. An interesting fact is that, during all of our experiments, the best results were achieved when the parameters on both the internal regression and external classification trees were the same, or at least very similar. Another

8. The silhouette metric balances the cohesion and separation of the clusters. The cohesion measures the intra-cluster distance which it should be ideally small. Similarly, the *separation* measures the dissimilarity of the clusters which ideally, should be large. A silhouette metric close to 1 signals that the clusters are very well separated between themselves.

fact, is that the most accurate models, were the ones with shallow trees (dephts of 3 or 4, signalling that when we bundle a lot of garments together, even those that are not as similar, its predictive power increases drastically. However, we are over-smoothing the data and we are not capturing the true nuisances. In addition, the set of comparables would be irrelevant. Our recommendation is to thoroughly cross-validate these parameters if using CART models. These were the greatest sources of variability on the results. But, note that for the classification model, we subset the N_{train} data-points into a sub-train / test set to validate the parameters for the external CART classification model.

CWR computational performance. Last, we examine the computational performance of the CWR model with this dataset. In general, the algorithm is fairly quick and usually converges in less than 10 seconds, due to data-points not changing clusters anymore. Similarly, in most of the cases, most of the error is captured in the first few iterations of the model and the WMAPE is reduced greatly as seen in figure 4-8. Interestingly, although Baardman et al. 2017 presents a proof guaranteeing convergence, this convergence is guaranteeing based on the loss function dictated by the regression model. Hence, since regression CART models are selected, the loss function is the mean square error which is different than the WMAPE. This is the reason convergence is not completely uniform and there are some *jumps* in later stages of the algorithm. In practice however, the algorithm converged perfectly and a minimal WMAPE is returned. Note that there are certain combinations of K and the CART hyper-parameters where the algorithm can fall into a loop, usually when K is even and a a single data-point jumps between models. For these instances the max iterations can be reached. Similarly, there are instances where the last convergence is not reached but the error kept on decreasing at a very slow rate. It is in the interest of the researcher to define what is *good enough* in terms of rate of convergence in order for the algorithm to be tractable. Last, although the CWR algorithm runs in polynomial time,[9] in practice if either N or d scales rapidly (e.g. using all the 300 dimensions of the gte) the algorithm will suffer a severe penalty in execution time.

9. If the internal regression model runs in polynomial time as well

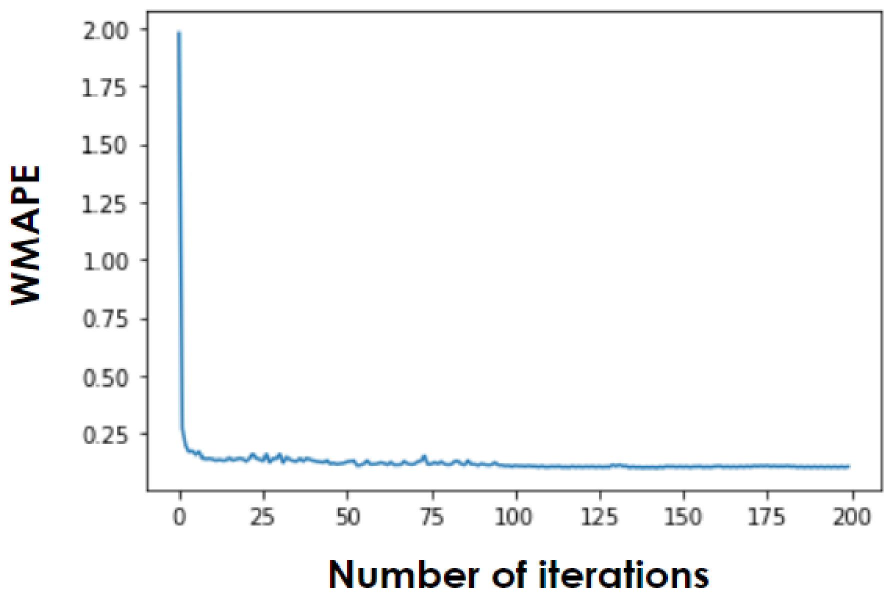


Figure 4-8: Convergence of the CWR algorithm

4.3.1 Summary of results

Based on the data structures developed the two models work as expected, achieving the objectives set out in section 1.2. They make a forecast $\hat{\boldsymbol{y}}_o$ and find a set of relevant comparables $\widehat{C}$. For some cases, the forecast accuracy is better (in retrospective) than that of the buyers with a suitable set of comparables. For others, this does not seem to be the case due to either a wrong set of comparables or a worse forecast accuracy. Hence, at scale, the models can not be used still as into fully replace the status quo processes, noting that there exists an advantage in both terms of data and censorship already present in the benchmark used. However, the performance of the models under this particular dataset, does not invalidate the algorithms nor the techniques developed. A thorough discussion of the numerical results, potential improvements and extensions to the models are presented in the following chapter 5.

Chapter 5

Conclusions and future work

It's the main objective... the growth of the Inditex group. Its daily task is marked by self-improvement and the continual search for new opportunities.

Amancio Ortega, 1999

5.1 Interpreting the results: trade-off between similarity and accuracy

As seen in the previous chapter 4, neither algorithm was able to improve the precision of the buyers and their process. Although discouraging at first, these algorithms lay the foundations to automate a part of the process presented in figure 2-7. Similarly, the downsides of the algorithm also show a way to perfect them. It is worthy to mention that the *business know-how*, derived by the human-element is hard to replicate and even more to improve. The human decision-making process is a lot more nuanced and usually, non-linear, taking into account several un-quantifiable factors. Therefore, it is been show that some of the best results are attained when the human criteria is mixed with that of the computer, see Bansal et al. 2019. Therefore, machine learning models like the ones presented in this thesis, should be treated as complementary to the established human processes.

The value of more and better quality data

Despite the results, one thing is clear all across the experiments: the more data the better the performance of the algorithms. Recall figure 4-4 and contrast it with figure 5-1. For the test set (i.e. the new fashion season) as k increases, the error goes down, but the average similarity also goes down. The correlation between these two metrics is 92%. This is expected because the more data-points used, the better the forecast accuracy and the error goes down. What is not expected, is that the similarity also goes down, i.e. the more garments used to predict the new one, the more likely it is for the set of comparables to be less similar. This downgrade in similarity also happens on the train set, depicted by the blue line, confirming our hypothesis. We call this problem the trade-off between similarity and accuracy.

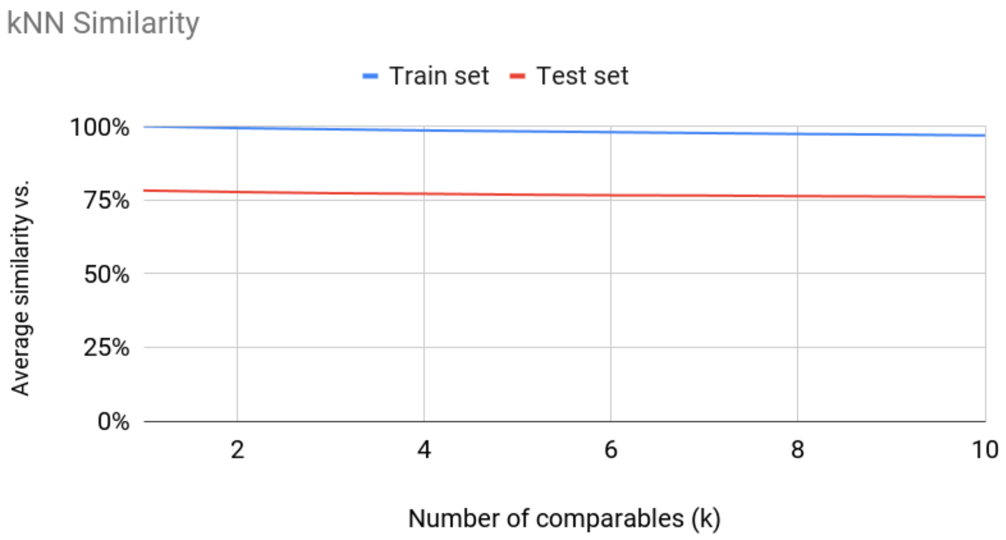


Figure 5-1: Average similarity metric for the kNN algorithm

Figure 5-1 also reveals an interesting fact: there are not enough garments in the dataset to find better comparables. In Gallien et al. (2015), states that Zara released around 8,000 new garments in just one year. This number is from 10 years ago so it would not be surprising that Zara could have doubled that amount. However, the dataset being used only contains 3,500 for the span of one year, due to the constraints outlined across this whole thesis. This mean that for a lot of new garments, there are not any relevant comparables. The delta between the average similarity between the train set and the test set is 20%, confirming this fact. It shows that in-season, there are more items that are similar to one another. When the new season is introduced, we loose a lot of comparable power.

Another issue to be addressed, mentioned in section 1.1, is that prediction of fashion is, by construction, hard and should be treated with care. Fashion trends are ever-changing, it is the point of the whole industry. Fashion mutates and evolves in unpredictable ways. Ultimately, design is a creative endeavour that computers are just starting to scratch the surface of. In particular for our task, the creativity and *recency*, becomes evident as a non-homogeneous train / test split of the dataset. I.e. the test set is a completely new season and therefore, we are aim to do predictions outside of the train set. This usually would go against a traditional machine learning framework, where both the train and test set have the same domain. This fact makes the algorithms harder to generalize since they have to *extrapolate*. A simple example: a comparable set of garments might not even exist for a garment if it is too modern. How will this completely new garment sell? Therefore, expecting new garment to behave like the old ones is not a completely valid assumption. Another simple example, a garment that was unpopular on a previous season, could come back as a *vintage* garment and now sell extremely well.

Irrespective of the challenges, both the *quantity* and *quality* of the data, are bound to aid the firms in their decision making processes. The quantity comes from having more varied garments in the train set, the larger the space the training set covers, the more and better comparables can be found. The train set is bound to increase naturally as time goes on so this should not represent a problem. The quality, comes from improving the representation of the garments themselves. As an example, if firms were to add more tags that capture all the small details (like the print patters seen in example 1 on section 4.1.1) then embeddings would only get more precise. Another idea is to use another *matching* algorithm, perhaps based on the images of the garments themselves as explored in section 2.2. What is clear is that the performance of the algorithms can only improve and, since they are already close to the benchmark, it is very likely that these algorithms can end up aiding in the decision making process.

5.2 Improving the models: future work

5.2.1 Further use-cases for the data

Independent of the results, the usefulness of the data structures built on chapter 2 is substantial. There are multiple additional use-cases that should be explored since $\boldsymbol{X}$ is a compact numerical representation of the garments which is not trivial. This ubiquitous rectangular data-structure can now be used for both clustering, prediction or any other traditional machine learning tasks. Some examples include: recommendation algorithms based on previous purchases, optimization of supply chain based on these covariates or promotion targeting given the performance of an item. However, one of the most useful problems where having covariates greatly aids the prediction is on demand forecasting, i.e. estimating the right Q_i. Although this appears to be a classical news-vendor problem, in more complex settings where the demand distribution is unknown but previous data is available, more advanced techniques like Lin et al. (2022) can be used. This paper presents a novel way of estimating the profit maximizing Q_i, based on covariate information of similar garments and a *risk profile* desired by the firm.

Another example of where can the data structure $\boldsymbol{X}$ can be useful is in the forecast of demand of the the individual sales. I.e. instead of estimating the normalized size curve $\boldsymbol{y}_i$, we estimate the actual sales for each size: $\dot{y}_{i,s}$ as defined in section 2.3. Hence, the total quantity Q_i would just be sum of all of these predicted quantities which solve two problems simultaneously. However, since this increases the degrees of freedom for the prediction, more data would be required. The algorithms presented can be easily adapted to make this forecast instead of the size curve. Some tests were run but, due to the great variability in Q_i for garments that might appear identical, the accuracy was greatly harmed.

Last, note that the*item embedding framework* shown in section 2.2 is not only applicable for fashion garments but can also be applied to any industries where natural language descriptions exist of the items, i.e. tags. A few examples could be other retail industries where items need to be described for their online stores: furniture and decoration, consumer product goods (CPG) and even something more abstract like music. This happens because ultimately, the subspace of $\mathbb{R}^d$ that is built is a numerical representation of an underlying item-space. For example, on music, songs already have several descriptors such as tone, mood and genre, emotions, etc. These can be used to build an embedding of the songs themselves. With these embedding, tools such as recommendation algorithms (based also on similarity) can be built which is also a large use-case for the data structures developed here.[1]

5.2.2 Reduction on the bias: the true size-curves

As explained in section 2.3 and 4.2, the observed $\boldsymbol{y}_i$ is already biased due to the purchase decision $\boldsymbol{\pi}_i$. This happens due to the sequential process described in figure 2-7 and financial constraints. Hence the *approximation* for $\boldsymbol{y}_i$, although adequate for the development of the algorithms, it is still severely lacking in terms of true accuracy. The assumption that $\boldsymbol{\pi}_i$ and $\boldsymbol{y}_i$ are independent does not hold and is reflected in the low 15% error observed for the buyers. Since we never observe the counterfactual, it is important to question if this is actually a fair benchmark or if it is a self-fulfilling prophecy due to the natural operations of the firm. However, it would also be simplistic to think a firm can just overbuy inventory in the sake of numerical accuracy. Hence, we propose two potential ways of eliminating some of these biases and achieving a better estimate for the true size curve $\boldsymbol{y}_i$, without compromising the firm's operations.

Randomized control experiments

In order to quantify the impact of an *unlimited inventory*, firms could select key garments and run controlled experiments on them. The experiment, would consist in taking a pool of stores with similar demographic profiles. Then, randomize which ones get the *treatment* and which ones do not. These would be deemed the *control group* and would function as usual. The ones in the treatment group, would sell the garments for longer, having a larger inventory and expose them in the store floor at a pattern that makes the garments more *available* to consumers. In Gallien et al. (2015), they mention that firms like Zara, most products have a short life cycle, of about 6 weeks. Hence, for certain brands, this would represent a shift from their regular operations which could be hard to implement, particularly from a cultural perspective. A potential effect of this, could be that certain consumers would be dissuaded if they expect to have a quick rotation of products. Alternatively, if the garment catches on by

1. If a client bought item x, he or she might be more willing to buy item y if it is *similar or complementary* to y.

first adopters, followers might want to buy the garment and it could lead to higher sales. It is impossible to predict the outcome without testing. Therefore, if implemented, comparing the two sales curves at the individual size level, could lead to *adjustment factors* that would help make a better approximation for the true $\boldsymbol{y}_i$.

The use of experiments to influence consumer behaviours is not new to the industry and, mostly online retailers have been doing it for years now, see Senecal and Nantel (2004). Similarly, the tech industry runs thousands of these *A/B tests* to understand their consumers. For a brick & mortar retailer, the store front is akin to the landing page on a website, with products being recommended to consumers. Experimenting on these surfaces is a great way for firms to understand and ultimately influence consumer behaviour. The largest drawback from this approach would be that it can be capital and operationally expensive.

Survival analysis

Alternative to experimentation, mathematical techniques called *survival analysis* have been developed to improve the quality of the approximations, Klein et al. (2013) and Klein and Moeschberger (2006). Most of these techniques have been developed for applications where the variable of interest is the *time to event*. However, modern proposals also exist that help finding the true variable in a regression setting like Michael, Kinga, and Márta (2002). The core idea to research is that, in the presence of censored data like the one we observe in the sales (i.e. total sales are rightly censored by the total amount purchased Q_i), can we approximate the true total sales by a function based on $\boldsymbol{x}_i$. These techniques require a sample of data-points which are available in our problem. By modifying the estimates, the paper proves that if N tends to infinity, their proposed estimates for the regression functions, minimized the true error, without ever observing Q_i. Albeit outside of the scope for this thesis, further work in this direction is recommended.

5.2.3 Reversing the causality arrow

To close this work, we explore a provocative idea: could firms use the upstream data in order to optimally design the garments? I.e. can the causality arrow in as see in in figure 1-1 be reversed? In a way, data in some form (fashion trends, previous sales, etc.) are already helping the firms design their garments. However, we are proposing going even further, if we have a numerical representation of the designs of the garments with a corresponding prediction, nothing is impeding us to *select* those features that end up return a fixed desired prediction. This has the potential to upend how firms operate, but, as more fashion firms switch to digital technologies (Balchandani, Berg, et al. 2022), this is naturally a trend of the industry.

Although controversial at first glance, it is true that some firms are already taking these approaches to design: see McCormick (2021) for an example. Similarly, although the fashion design is an extremely creative endeavour, the *unreasonable effectiveness* of data in creative endeavours have allowed new models to make strides in these *traditionally creative* fields, see Ramesh et al. (2022) for examples on images and Brown et al. (2020) for text. Some particular examples for fashion are presented in Dilmegani (2023) where new *generative artificial intelligence* models are already aiding the designers, seemingly creating new garments out of thin air. Ultimately, it is in the firm's best interest to invest in these technologies and to continue research the area because the technological trends are clear and disruptive. In the cooperation between the models and the humans clearly lies the success and differentiation of the firms.

Bibliography

Baardman, Lennart, Igor Levin, Georgia Perakis, and Divya Singhvi. 2017. "Leveraging comparables for new product sales forecasting." *Available at SSRN 3086237.*

Balchandani, Anita, Achim Berg, et al. 2022. *State of Fashion Technology Report 2022 | McKinsey.* https://www.mckinsey.com/industries/retail/our-insights/state-of-fashion-technology-report-2022. (Accessed on 02/26/2023), May.

Bansal, Gagan, Besmira Nushi, Ece Kamar, Walter S Lasecki, Daniel S Weld, and Eric Horvitz. 2019. "Beyond accuracy: The role of mental models in human-AI team performance." In *Proceedings of the AAAI conference on human computation and crowdsourcing,* 7:2–11.

Bertsimas, Dimitris, and Romy Shioda. 2007. "Classification and regression via integer optimization." *Operations research* 55 (2): 252–271.

Box, George E. P. 1976. "Science and Statistics." *Journal of the American Statistical Association* 71 (356): 791–799.

Brown, Tom, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. "Language models are few-shot learners." *Advances in neural information processing systems* 33:1877–1901.

Caro, Felipe. 2012. "Zara: Staying fast and fresh." *The European Case Clearing House, ECCH Case,* 612–006.

Chollet, Francois. 2017. *Deep Learning with Python.* Manning Publications Company. ISBN: 9781617294433. https://books.google.com/books?id=Yo3CAQAACAAJ.

Clark, Alexander, Chris Fox, and Shalom Lappin. 2012. *The handbook of computational linguistics and natural language processing.* Vol. 118. John Wiley & Sons.

Devlin, Jacob, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. "Bert: Pre-training of deep bidirectional transformers for language understanding." *arXiv preprint arXiv:1810.04805.*

Dilmegani, Cem. 2023. *Generative AI Fashion Industry: 5 Use Cases with Case Studies.* https://research.aimultiple.com/generative-ai-fashion/. (Accessed on 02/26/2023), February.

Fridley, Lila. 2018. "Improving Online Demand Forecast using Novel Features in Website Data: A Case Study at Zara." Master's thesis, Massachusetts Institute of Technology.

Gallien, Jérémie, Adam J. Mersereau, Andres Garro, Alberte Dapena Mora, and Martín Nóvoa Vidal. 2015. "Initial shipment decisions for new products at Zara." *Operations Research* 63 (2): 269–286.

Garro, Andres. 2011. "New Product Demand Forecasting and Distribution Optimization: A Case Study at Zara." Master's thesis, Massachusetts Institute of Technology.

Hastie, T., R. Tibshirani, and J. Friedman. 2009. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction, Second Edition.* Springer Series in Statistics. Springer New York. ISBN: 9780387848587.

Hyndman, Rob J., and Anne B. Koehler. 2006. "Another look at measures of forecast accuracy." *International journal of forecasting* 22 (4): 679–688.

Inditex. 2023a. *Zara finance.* https://www.inditex.com/itxcomweb/en/investors/finance. (Accessed on 01/10/2023).

———. 2023b. *Zara history.* https://www.inditex.com/itxcomweb/en/group/history. (Accessed on 01/10/2023).

———. 2023c. *Zara Women US - Summer collection 2023, blue demin dress.* https://www.zara.com/us/en/zw-the-denim-dress-p02553043.html?v1=227077533. (Accessed on 02/08/2023).

———. *Zara's model.* https://www.inditex.com/itxcomweb/es/grupo/nuestro-modelo. (Accessed on 03/01/2023).

Jurafsky, D., and James H. Martin. 2009. *Speech and Language Processing: An Introduction to Natural Language Processing , Computational Linguistics, and Speech Recognition.* Prentice Hall series in artificial intelligence. Pearson Prentice Hall. ISBN: 9780131873216.

Klein, J.P., H.C. van Houwelingen, J.G. Ibrahim, and T.H. Scheike. 2013. *Handbook of Survival Analysis.* Chapman & Hall/CRC Handbooks of Modern Statistical Methods. Taylor & Francis.

Klein, J.P., and M.L. Moeschberger. 2006. *Survival Analysis: Techniques for Censored and Truncated Data.* Statistics for Biology and Health. Springer, New York.

Kong, Evelyne. 2015. "Cannibalization Effects of Products in Zara's Stores and Demand Forecasting." Master's thesis, Massachusetts Institute of Technology.

Lin, Shaochong, Youhua Chen, Yanzhi Li, and Zuo-Jun Max Shen. 2022. "Data-Driven Newsvendor Problems Regularized by a Profit Risk Constraint." *Production and Operations Management* 31 (4): 1630–1644.

Martinez Puppo, Manuel. 2019. "Replenishment in an Integrated Stock World." Master's thesis, Massachusetts Institute of Technology.

McCormick, Packy. 2021. *Shein: The TikTok of Ecommerce.* https://www.notboring.co/p/shein-the-tiktok-of-ecommerce. (Accessed on 01/10/2023), May.

Michael, Kohler, Máthé Kinga, and Pintér Márta. 2002. "Prediction from Randomly Right Censored Data." *Journal of Multivariate Analysis* 80 (1): 73–100. https://www.sciencedirect.com/science/article/pii/S0047259X00919730.

Mikolov, Tomás, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. "Efficient estimation of word representations in vector space." *arXiv preprint arXiv:1301.3781.*

Mikolov, Tomás, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. "Distributed representations of words and phrases and their compositionality." *Advances in neural information processing systems* 26.

Olabiyi, Kiitan. 2021. *Fashion and Big Data.* https://medium.com/data4fashion/fashion-and-big-data-application-cb946dd76844. (Accessed on 01/10/2023), October.

Park, Young Woong, Yan Jiang, Diego Klabjan, and Loren Williams. 2017. "Algorithms for generalized clusterwise linear regression." *INFORMS Journal on Computing* 29 (2): 301–317.

Pennington, Jeffrey, Richard Socher, and Christopher D Manning. 2014. "Glove: Global vectors for word representation." In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP),* 1532–1543.

Ramesh, Aditya, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. 2022. "Hierarchical text-conditional image generation with clip latents." *arXiv preprint arXiv:2204.06125.*

Senecal, Sylvain, and Jacques Nantel. 2004. "The influence of online product recommendations on consumers' online choices." *Journal of retailing* 80 (2): 159–169.

Späth, Helmuth. 1979. "Algorithm 39. Clusterwise linear regression." *Computing* 22:367–373.

Szeliski, Richard. 2022. *Computer vision: algorithms and applications.* Springer Nature.

Tan, P.N., M. Steinbach, A. Karpatne, and V. Kumar. 2019. *Introduction to Data Mining.* Pearson.

Trivedi, Shubhendu, Zachary A. Pardos, and Neil T. Heffernan. 2015. "The utility of clustering in prediction tasks." *arXiv preprint arXiv:1509.06163.*

Vaswani, Ashish, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. "Attention is all you need." *Advances in neural information processing systems* 30.

Walton, Mary. 1988. *The Deming Management Method: The Bestselling Classic for Quality Management!* Penguin.

Winston, Wayne L. 2022. *Operations Research: Applications and Algorithms.* Cengage Learning. ISBN: 9780357907818.

Wu, Yonghui, Mike Schuster, Zhifeng Chen, Quoc V Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, et al. 2016. "Google's neural machine translation system: Bridging the gap between human and machine translation." *arXiv preprint arXiv:1609.08144.*

Acronyms

***k*-means** *k*-means clustering. 41, 62, 77

BERT Bidirectional Encoder Representations from Transformers. 22, 77

CART Classification and regression Trees. 15, 16, 41, 46, 47, 48, 49, 50, 51, 58, 59, 61, 62, 63, 77

CBOW Continuous bag-of-words. 21, 77

CWR Cluster-while-regress. 7, 15, 16, 41, 42, 43, 45, 46, 47, 50, 51, 56, 58, 59, 60, 61, 62, 63, 77

GloVe Global Vectors. 21, 22, 27, 77

gte Garment-tags-embedding. 26, 27, 28, 29, 40, 50, 52, 54, 55, 59, 60, 63, 77

HSV hue, saturation value color scale. 28, 77

kNN *k*-nearest neighbors. 7, 15, 38, 39, 45, 50, 51, 56, 57, 58, 59, 61, 77

LASSO least absolute shrinkage and selection operator. 48, 77

LGO MIT Leaders for Global Operations Program. 15, 18, 77

NLP Natural Language Processing. 15, 19, 24, 77

PCA Principal component analysis. 29, 54, 59, 77

RF Random Forest. 62, 77

RGB Red, green, blue color scale. 28, 77

SINT *Sistema de gestión INTegrada de inventario*. 16, 17, 77

SKU Stock keeping unit. 23, 77

SVM support vector machines. 45, 77

WMAPE Weighted mean absolute percentage error. 15, 34, 35, 48, 54, 56, 57, 58, 61, 63, 77