

DECODING STYLE: LEVERAGING MACHINE LEARNING FOR
FASHION TREND DETECTION

by

KORA DUMPERT

A THESIS

Presented to the Department of Data Science
and the Robert D. Clark Honors College
in partial fulfillment of the requirements for the degree of
Bachelor of Science

May 2024

An Abstract of the Thesis of

Kora Dumpert for the degree of Bachelor of Science
in the Department of Data Science to be taken June 2024

Title: Decoding Style: Leveraging Machine Learning for Fashion Trend Detection

Approved: Dr. Thanh Nguyen
Primary Thesis Advisor

Clothing, once a necessity for human survival, has evolved into a powerful means of self-expression and social identification. Today, the fashion industry stands as a multi-trillion-dollar global force, shaping economies and cultures. However, its volatile nature and environmental footprint necessitate innovative approaches to trend identification and inventory management. This research explores the fusion of machine learning techniques with fashion trend analysis to offer an accessible solution. By leveraging image clustering algorithms, this study identifies key patterns and trends within fashion apparel. The research unveils significant insights into Spring/Summer 2024 color preferences, item types, and material/print trends. An example of trends identified include red colors, dresses, and stripes. Notably, the identified trends are cross-referenced with traditional fashion publications to assess accuracy. While the study acknowledges certain limitations, particularly in item type differentiation, it proposes avenues for future research. Ultimately, this research not only offers a glimpse into the future of fashion trend analysis but also presents a pathway towards more sustainable and efficient inventory planning. By harnessing the power of machine learning, the fashion industry can align production with consumer preferences, minimizing waste and environmental impact while maximizing economic efficiency.

Acknowledgements

I would like to express my deepest gratitude to my advisors, Dr. Thanh Nguyen, and Dr. Trond Jacobsen, for their unwavering support and patience throughout the thesis process. Dr. Nguyen, your instrumental role in guiding me from initial discussions to project implementation has been invaluable. Without your expertise and guidance, this thesis would not have come to fruition. Dr. Jacobsen, your support during my time at the Clark Honors College has been truly indispensable. I would also like to extend my thanks to Dr. Stephen Fickas for his valuable input on my methodology and experimental design, which played a crucial role in turning my ideas into reality.

To my family and friends, thank you for being my rock and unwavering support system. To my parents, thank you for instilling in me the value of hard work and for your constant encouragement throughout my life. To my sister, Elise, your light-hearted perspective, and infectious laughter have provided much-needed relief during the challenges of this process. To my best friend Lauren, your support and guidance in integrating personal elements into this academic endeavor have been invaluable. And to all my friends, thank you for reminding me to strike a balance between academic achievement and savoring the moments of joy during my final year of college.

Table of Contents

Chapter 1: Introduction	9
Chapter 2: Literature Review	13
2.1 Machine Learning for image analysis	13
2.2 Types of Machine Learning	13
2.3 Current Work	14
Chapter 3: Methodology	17
3.1 Dataset	17
3.2 Image Pre-Processing	18
3.3 Clustering	22
3.3.1 DBSCAN	22
3.3.2 OPTICS	23
3.3.3 Agglomerative	24
3.3.4 K-means	25
3.4 Implementation	27
3.4.1 DBSCAN	30
3.4.2 OPTICS	33
3.4.3 Agglomerative	35
3.4.4 K-Means	36
3.5 Choosing a Clustering Method	37
3.6 Identifying Trends	38
3.7 Assessing Accuracy	38
Chapter 4: Experiment	39
4.1 K-means Cluster Image Examples	40
4.2 Cluster Contents and Image Makeup	44
4.3 Results	71
4.4 Accuracy & Evaluation	72
Chapter 5: Conclusion	74
References	76
Supporting Materials	
Python File: make_combined_img_folder.py	
Python File: rename_img.py	
Python File: save_no_background.py	

Python File: DBSCAN_clustering.py

Python File: Kmeans_clustering.py

Python File: agglomerative_clustering.py

Python File: OPTICS_clustering.py

List of Figures

Figure 1: John Sebastian Woodstock 1969	11
Figure 2: Refined Tie Dye in Vogue Italia 1970	11
Figure 3: OperaSport Runway Image	17
Figure 4: OperaSport Runway Image with Background Removed	21
Figure 5: Example of image discarded from dataset.	22
Figure 6: DBSCAN Knee Curve Plot	32
Figure 7: DBSCAN Clustering Visualization	32
Figure 8: Reachability Plot	34
Figure 9: Agglomerative Clustering Dendrogram	35
Figure 10: Elbow Plot	37
Figure 11: Example image for trend identification	38
Figure 12: Cluster 0 Example Images	40
Figure 13: Cluster 1 Example Images	40
Figure 14: Cluster 2 Example Images	40
Figure 15: Cluster 3 Example Images	41
Figure 16: Cluster 4 Example Images	41
Figure 17: Cluster 5 Example Images	41
Figure 18: Cluster 6 Example Images	42
Figure 19: Cluster 7 Example Images	42
Figure 20: Cluster 8 Example Images	42
Figure 21: Cluster 9 Example Images	43
Figure 22: Cluster 10 Example Images	43
Figure 23: Cluster 0 Color Frequency Chart	45
Figure 24: Cluster 0 Item Type Frequency Chart	45
Figure 25: Cluster 0 Material/Print Frequency Chart	46
Figure 26: Cluster 1 Color Frequency Chart	47
Figure 27: Cluster 1 Item Type Frequency Chart	48
Figure 28: Cluster 1 Material/Print Frequency Chart	48
Figure 29: Cluster 2 Color Frequency Chart	50
Figure 30: Cluster 2 Item Type Frequency Chart	50

Figure 31: Cluster 2 Material/Print Frequency Chart	51
Figure 32: Cluster 3 Color Frequency Chart	52
Figure 33: Cluster 3 Item Type Frequency Chart	53
Figure 34: Cluster 3 Material/Print Frequency Chart	53
Figure 35: Cluster 4 Color Frequency Chart	55
Figure 36: Cluster 4 Item Type Frequency Chart	55
Figure 37: Cluster 4 Material/Print Frequency Chart	56
Figure 38: Cluster 5 Color Frequency Chart	57
Figure 39: Cluster 5 Item Frequency Chart	57
Figure 40: Cluster 5 Material/Print Frequency Chart	58
Figure 41: Cluster 6 Color Frequency Chart	59
Figure 42: Cluster 6 Item Type Frequency Chart	60
Figure 43: Cluster 6 Material/Print Frequency Chart	60
Figure 44: Cluster 7 Color Frequency Chart	62
Figure 45: Cluster 7 Item Type Frequency Chart	62
Figure 46: Cluster 7 Material/Print Frequency Chart	63
Figure 47: Cluster 8 Color Frequency Chart	64
Figure 48: Cluster 8 Item Type Frequency Chart	65
Figure 49: Cluster 8 Materials/Print Frequency Chart	65
Figure 50: Cluster 9 Color Frequency Chart	67
Figure 51: Cluster 9 Item Type Frequency Chart	67
Figure 52: Cluster 9 Material/Print Frequency Chart	68
Figure 53: Cluster 10 Color Frequency Chart	69
Figure 54: Cluster 10 Item Type Frequency Chart	70
Figure 55: Cluster 10 Material/Print Frequency Chart	70

List of Tables

Table 1: Cluster Size & Number Breakdown	39
Table 2: Cluster 0 Frequency Breakdown	44
Table 3: Cluster 1 Frequency Breakdown	46
Table 4: Cluster 2 Frequency Breakdown	49
Table 5: Cluster 3 Frequency Breakdown	51
Table 6: Cluster 4 Frequency Breakdown	54
Table 7: Cluster 5 Frequency Breakdown	56
Table 8: Cluster 6 Frequency Breakdown	59
Table 9: Cluster 7 Frequency Breakdown	61
Table 10: Cluster 8 Frequency Breakdown	64
Table 11: Cluster 9 Frequency Breakdown	66
Table 12: Cluster 10 Frequency Breakdown	68
Table 13: Key Trend Table Summary	72
Table 14: Trends Seen in Publications	73

Chapter 1: Introduction

170,000 years ago, clothing emerged as an environmental necessity for humans, protecting them from extreme weather (Viegas 2011). However, since then, choice of dress has evolved into a way that people express their interests, affiliations, and often their socioeconomic status. This choice of dress is often described as fashion, which is “a style that is popular at a particular time, especially in clothes” (Cambridge Dictionary, s.v. “fashion”). Additionally, fashion itself is cyclical and is informed through a combination of art, culture, and social norms.

Today, the fashion industry is a 2.5 trillion-dollar industry across the globe (Maloney 2019). In 2019 Americans spent over 380 million dollars on fashion apparel and footwear, and the industry employed over 1.8 million Americans (Maloney 2019). Fashion has significant economic impacts for the United States and around the globe, especially for ecommerce. In 2024, the fashion industry is projected to grow between 2 and 4 percent over the year (Balchandani et al. 2024). Another characteristic of the fashion industry is fluctuations in demand. Supply chains are often the most impacted, and experience a phenomenon called the “bullwhip effect” where consumer demand varies rapidly, and many supply chains cannot keep up (Balchandani et al. 2024)

Demand tends to be highly volatile in the fashion industry, thus forecasting is incredibly important to suppliers and retailers alike. Forecasting that is either inaccurate or too slow can lead to serious business and environmental problems. First, if forecasted demands do not meet the true demands of consumers, businesses lose possible revenue from sales that they could have completed. Additionally, if demand is predicted to be less than reality, retailers and suppliers overproduce, resulting in excess inventory and further financial losses. Environmentally,

accurate demand forecasting is increasingly important as sustainability practices are a high priority amidst global warming and pollution. According to the Natural Resource Defense Council, clothing contributes to one-fifth of the annual 300 million tons of plastic that pollutes the earth and its waterways (Greenfield 2023). It is imperative that forecasting for fashion apparel is tailored to reduce any negative environmental impacts of this industry.

Fashion trends are often defined as apparel styles that have been adopted by a large group of people as a form of social behavior (James Hillman Fashion Consultancy 2021). These trends will often emerge from a fashion designer's seasonal fashion show and are adopted by consumers, or vice versa. Microtrends are apparel patterns that might shift every few months or so, while macrotrends tend to last multiple years (Gaddamadugu 2023). One example of a trend associated with a time period is the emergence of tie dye clothing in the 1960's and early 70s. Tie dye, which emerged in popularity due to its use as Protest Art and Pop Fashion, became a fashion trend for the general public and emerged in high fashion magazines. Figure 1 below depicts John Sebastian at Woodstock 1969, sporting a tie dye shirt. Figure 2 shows a more refined version of the trend in *Vogue Italia* 1970. Thus, social issues such as protesting the Vietnam war found its way into a fashion trend that is easily associated with this time period.



Figure 1: John Sebastian Woodstock 1969

(Dye Happy, 2024)



Figure 2: Refined Tie Dye in Vogue Italia 1970

(Vogue Italia, 1970)

Traditionally, fashion trend identification falls into the hands of individuals who attended fashion shows, and brought their takeaways to fashion magazines like Vogue, or product developers at apparel brands (MasterClass 2021). However, this approach has the potential to shift in the era of big data and the internet.

Machine learning techniques offer a unique solution to fashion trends identification. Machine learning is a subset of artificial intelligence, that aims to use a machine to solve problems like a human might. This learning is generally a combination of computational algorithms that use data and often feedback to determine patterns, predict values, and classify items. Because fashion trends are cyclical and pattern based, machine learning algorithms are well suited for this kind of application. Additionally, as machine learning becomes more accessible, it is a way for manufacturers, brands, and consumers alike to understand trends and determine what will happen next.

Thesis Outline

This thesis will be focused on developing a process for trend analysis that would be accessible to anyone interested in identification fashion trends. First, a method for finding patterns using machine learning cluster analysis will be identified. Second, a process for obtaining and pre- processing the images for the trend analysis will be developed. Next, the process for assigning images into patterns or trend groups will be determined. Then, the results of the trend analysis will be summarized, and a way to determine accuracy of the process will follow. Finally, there will be a discussion of limitations and possible applications for this trend analysis process.

Research Questions

- Q1: How can machine learning techniques such as cluster analysis be used to identify trends in fashion apparel?
- Q2: What are the preprocessing steps for images used in this trend identification?
- Q3: What are the trends that are found from this analysis?
- Q4: How does this process compare to traditional methods when evaluated?

Chapter 2: Literature Review

2.1 Machine Learning for image analysis

One of the promising applications of machine learning is image analysis. Image analysis allows for the collection of extensive data but has historically relied on significant human labor (Belcher et al. 2023). By utilizing machine learning techniques, human labor and human error can both be reduced. The role of machine learning in image analysis is to “compress high dimensional data into much lower- dimensional summaries relevant to a particular task or study objective” (Belcher et al. 2023). While humans can perform this task naturally, there are significant benefits to using a machine learning algorithm, such as reduction of cost and time necessary for processing (Norouzzadeh et al. 2018)

2.2 Types of Machine Learning

Machine learning is often categorized into two different sections: supervised, and unsupervised learning. Supervised machine learning refers to the use of training data that is labeled. Training data is used to teach the algorithm to recognize when something matches a label or does not match a label. For example, a supervised machine learning algorithm may be learning how to differentiate between pictures of cats, and pictures of dogs. This kind of approach is called classification, where the algorithm assigns already known labels to a set of images. The training data set would be composed of various images of cats and dogs, with the correct labels for each animal. Once the algorithm learns how to differentiate between these training images, a test data set will be used. The test data set would be composed of images of cats and dogs, but do not have labels associated with them. The goal of the supervised machine learning algorithm would then be to determine which picture belongs to what animal.

In contrast, unsupervised machine learning models are trained with data that is unlabeled. This approach focuses on identifying patterns in datasets. One example of this is cluster analysis. Clustering identifies similarities and differences, and subsets data based on these results (Han et al, 2021). For example, if an unsupervised machine learning algorithm is given images of dogs and cats, it will find what qualities of the images separate one another. The result could be any number of clusters made up of images. They might have two clusters, one with dogs and one with cats. Or three clusters, one with big dogs, and one with small dogs and cats. However, the algorithm wouldn't determine that these are dogs or cats, just one cluster and another cluster.

2.3 Current Work

Current literature aimed at identifying fashion trends and product forecasting is available, however the approaches taken vary widely. One 2024 paper out of the University of Verona, Italy, uses a neural-network approach in combination with Google Trends to identify popularity of textual terms and relate this to the predicted demand of a new fashion item (Skenderi et al. 2024). Notably, this paper forecasts the sales of a specific item, rather than identifying overarching trends. The research also is generalized to Nunalie, an Italian fast fashion brand and is therefore potentially not as representative as it could be globally.

Another article, published in 2019, delves into the forecasting of fashion items using historical purchase patterns and generalizing this to abstract designs. This approach, using machine learning methods and python implementation, addresses the issue of predicting future demand for items not yet created. While the authors successfully forecasted demand for items, once again, general trends were not noted, and there is an absence of image analysis (Singh et al. 2019).

Recent literature has also shown how clustering can be applied to fashion research. Cluster analysis aims to maximize the similarity between samples in each cluster, while at the same time maximizing the dissimilarity between clusters (Peng and Li 2023). It is often used in exploratory data analysis before further research is performed. Cluster analysis frequently appears in literature across fields that use multivariate data. Some applications include biological fields such as genomics (Oyelade et al. 2016), marketing (Benslama and Jallouli 2020), materials science (Vincent et al. 2023) to name a few.

In fashion trend research, a 2019 article called “Unsupervised Deep Clustering for Fashion Images” develops a unique clustering model based on real world data from Amazon (Yan et al. 2019). Another article, published in 2015, utilizes cluster analysis to group images of clothing and accessory items that are related to one another. The author’s found that their method of clustering items together might predict what people would purchase and pair together (McAuley, Targett, and van den Hengel 2015).

Several studies pull images from the internet and create a cleaned fashion dataset from which they categorize trends (Huang, Lu, and Hsu 2021) and (Vittayakorn et al. 2015). These provide an opportunity for image analysis as applied to fashion images, with many using clustering or other machine learning techniques. However, because of the cyclical nature of fashion, these datasets leave little room for determining future trends as time goes on and the datasets age.

Perhaps the most promising research on fashion trend analysis is a system called “Neo-Fashion”, created in 2021 (Zhao, Li, and Sun 2024). This system utilizes clustering, along with object detection methods to split runway images into multiple different pieces. This is helpful for fashion images, as fashion usually is made up of a variety of items. From here, the study

prepared a training dataset and used this, along with a Region Convolutional Neural Network to categorize each fashion image component. Finally, Neo-Fashion identifies trends in colors, combinations of clothing, and styles.

This research will be working with clustering image analysis to identify trends in fashion. While much less technical than some machine learning techniques, cluster analysis is easily understood and offers an opportunity for accessibility to the public. It also opens doors for integration into inventory planning that could help small retailers worldwide.

Chapter 3: Methodology

3.1 Dataset

To effectively identify future trends, a dataset of images for a season after Winter 2024 needed to be identified. Brands generally show their upcoming collections a season or two in advance, at large runway shows across the globe. Perhaps some of the most notable shows are in Milan, Paris, and New York. For this research, we contacted firstVIEW, a team of photographers that attend runway events and photograph full length pictures of each look in a show. On their website, they have images available for free download categorized by designer, season, and year. After contacting their team directly for permission to use these photos, we decided to choose 26 different brands and 2024 Spring/Summer their collections. These were chosen with the letters of the alphabet. For example, our “B” designer images were from Balmain. If there was no designer for a letter A-Z, a designer was randomly chosen. Figure 3 below is an example of an image from designer OperaSport, and one of our images used in this study.



Figure 3: OperaSport Runway Image
(firstVIEW, 2024)

After the images were downloaded from firstVIEW, they were compiled into a single folder and consisted of 1053 images.

3.2 Image Pre-Processing

1. Renaming folders and images

To obtain better results, we decided to first start by creating an efficient labeling system for folders of brand images and singular images. This provides an opportunity to understand what brands are in each cluster. The brand folders were renamed to just the brand, resulting in folders with names “Balmain” and “Chanel”. The images were named for their brand, along with an arbitrary number for labeling purposes. For example, an image in the Chanel folder could be called “Chanel8.png”. The code snapshot below depicts the function used to relabel the folders and images.

```
def rename_folders_and_images(folder_path):
    for root, dirs, files in os.walk(folder_path, topdown=False):
        for folder_name in dirs:
            # Extract the name before the "-"
            new_folder_name = folder_name.split('-')[0]

            # Create the new path for the folder
            old_folder_path = os.path.join(root, folder_name)
            new_folder_path = os.path.join(root, new_folder_name)

            # Rename the folder
            os.rename(old_folder_path, new_folder_path)

        for file_name in files:
            if file_name.lower().endswith(".jpg"):
                # Extract the folder name (excluding the index)
                folder_name = root.split(os.path.sep)[-1].split('-')[0]

                # Create the new image name
                new_image_name = f"{folder_name}{files.index(file_name) + 1}.png"
                new_image_name = new_image_name.replace(" ", "")

                # Create the new path for the image
                old_image_path = os.path.join(root, file_name)
                new_image_path = os.path.join(root, new_image_name)

                # Rename the image
                os.rename(old_image_path, new_image_path)
```

2. Combined Image Folder

```
def combine_all_brands_images(input_folder, output_folder):
    try:
        os.makedirs(output_folder, exist_ok=True)

        # Remove existing content in the all_brands folder
        for file_in_output_folder in os.listdir(output_folder):
            file_path = os.path.join(output_folder, file_in_output_folder)
            try:
                if os.path.isfile(file_path):
                    os.unlink(file_path)
                elif os.path.isdir(file_path):
                    os.rmdir(file_path)
            except Exception as e:
                print(f"Error deleting {file_path}: {str(e)}")

        # Copy all PNG images from subfolders to the all_brands folder
        for root, dirs, files in os.walk(input_folder):
            for image_file in files:
                if image_file.endswith(".png"):
                    input_path = os.path.join(root, image_file)
                    output_path = os.path.join(output_folder, image_file)
                    shutil.copy2(input_path, output_path)

        print("Combining images into all_brands folder successful.")
    except Exception as e:
        print(f"An error occurred: {str(e)}")
```

In this step, we combined all images from each brand folder into one folder for ease of access. In addition, to allow for this code to be re-ran multiple times, we also included code to remove existing content in the designated folder before adding new content in.

3. Removing Backgrounds

```
def remove_and_save_with_white_background(input_path, output_folder):
    try:
        input_image = Image.open(input_path)
        output_image = remove(input_image)

        # Convert RGBA to RGB
        output_image = output_image.convert("RGBA")

        # Create a new image with a white background
        white_background = Image.new("RGBA", output_image.size, (255, 255, 255, 255))

        # Composite the foreground onto the white background
        result_image = Image.alpha_composite(white_background, output_image)

        # Convert RGBA to RGB
        result_image = result_image.convert("RGB")

        # Save the image with a white background as PNG
        output_filename = os.path.basename(input_path).replace(".png", "_WBR.png")
        output_path = os.path.join(output_folder, output_filename)
        result_image.save(output_path, format="PNG")
    except Exception as e:
        print(f"Error processing {input_path}: {str(e)}")
```

Next, since each background was different, we decided to isolate just the model in each photo by removing the backgrounds all together. Figure 4 depicts figure 3 without a background. This was achieved using assistance from ChatGPT in developing a script to remove all image backgrounds and save the new images with new names. The python packages used were PIL, rembg, and os. The function 'remove_and_save_with_white_background' opens the image using PIL's Image function, and then the rembg library's remove function to take out the background. Then, the function uses Image to convert the image from RGBA to RGB format because further steps must use the RGB image format. A new image is then created with a white background, ensuring the size matches the previously (background removed) output image. Next, Image.alpha_composite adds the first image to the white background image, which is then converted back to RGB format. Finally, the function saves the new image as

“imagenname_WBR.png” to differentiate it from the original image. An example of an image that has its background removed is shown below. This process took about 10 seconds per photo, but the results were quite good.



Figure 4: OperaSport Runway Image with Background Removed

4. Final check of images

Some images had a harder time removing backgrounds than others. Specifically, some images that had people in the background did fail to fully remove their backgrounds. To ensure these were not included in the final analysis, a manual review of photos was conducted. Any photos that had issues with background removal were discarded from the final image collection. 73 images were manually discarded. This left 980 images for the final analysis. An example of an image that was discarded is below.



Figure 5: Example of image discarded from dataset.

3.3 Clustering

There are various methods of cluster analysis. Some of the most well-known algorithms are K-means, OPTICS, DBSCAN, and Agglomerative clustering. We wanted to explore various methods of clustering for this project.

3.3.1 DBSCAN

DBSCAN stands for Density Based Spatial Clustering of Applications with Noise. Density based clustering groups data points to reflect how close they are to one another. This is an example of clustering that does not require the pre-defined number of clusters. DBSCAN estimates density of a dataset by identifying points that lie within a certain neighborhood or radius, *Epsilon*, and will “consider two points connected if they lie within each other’s neighborhood” (Aggarwal and Reddy 2014). DBSCAN also notes some points as *core points*, if they reach or exceed the predefined *MinPts* number of points within their radius, *Epsilon*. DBSCAN also can identify points as noise if they are not density reachable. According to BOOK, “a point *q* is directly density-reachable from a core point *p* if *q* is within the *Epsilon*-neighborhood of *p*” (Aggarwal and Reddy 2014). This algorithm must have predefined *Epsilon*

an *MinPts* values. DBSCAN is generally able to deal with noise and outliers well. DBSCAN is implemented using `sklearn.cluster.DBSCAN` class.

DBSCAN's algorithm is as follows (adapted from Aggarwal and Reddy, 2014):

1. Beginning with arbitrary point, p , identify all points density-reachable to p , considering both *Epsilon* and *MinPts*.
2. If p is a core point, a cluster has been identified.
3. If p is not a core point, DBSCAN assigns p to noise and moves on to the next point in the dataset.
4. The algorithm ends when all points are categorized as belonging to a cluster or as noise.

3.3.2 OPTICS

OPTICS stands for Ordering Points to Identify the Clustering Structure. This cluster method is highly related to DBSCAN, however does not require an exact *Epsilon* or *MinPts* and instead tries an “infinite number of distance parameters” that are smaller than a distance *Epsilon* (Aggarwal and Reddy 2014). OPTICS can help differentiate between clusters with varying levels of density. OPTICS does not actually produce clustering results of data, but outputs cluster ordering. Cluster ordering is a linear list of all objects, which represents their density-based clustering structure (Han et al, 2021). The algorithm is used to create a reachability plot, which checks how densely points are packed. This plot allows the user to decide clusters based on their level of density. OPTICS is implemented using `sklearn.cluster.OPTICS` class.

The OPTICS algorithm works as follows (adapted from ArcGIS Pro):

1. Beginning with arbitrary point, p , search all neighbor distances less than or equal to $Epsilon$.
2. If p has a neighbor distance less than $Epsilon$, p is assigned that distance as it's reachability distance. If all neighbor distances are greater than $Epsilon$, the smallest is assigned as the reachability distance.
3. When no further points are within $Epsilon$, OPTICS moves to another point and repeats steps 1 and 2.
4. After each iteration, reachability distances are calculated and ordered.
5. Finally, the reachability plot is created from each reachability distance and is used to detect clusters.

3.3.3 Agglomerative

This clustering method is a bottom-up hierarchical method that first labels each data point as its own cluster. From here, a dissimilarity matrix is created. The algorithm merges the two closest clusters (based on chosen computed distances) and updates the dissimilarity matrix accordingly until the final cluster contains all data points. In this paper, the chosen distance computation follows *Ward's Criterion*. The ward method minimizes variance in clusters and defines the distance between clusters as how much the sum of squares increases when merged, like K -means SSE method. Wards method intends to keep this growth as small as possible (Aggarwal and Reddy 2014). For clusters C_a and C_b , and their cardinalities are N_a and N_b , the Ward's method to measure the SSE increase when $C_a \cup C_b$ merge is as follows:

$$W(C_{a \cup b}, C_{a \cup b}) - W(C, c) = \frac{N_a N_b}{N_a + N_b} \sum_{v=1}^M (c_{av} - c_{bv})^2$$

$$= \frac{N_a N_b}{N_a + N_b} d(c_a, c_b)$$

The agglomerative clustering algorithm is as follows (adapted from Aggarwal and Reddy, 2014):

1. Compute the dissimilarity matrix between all points
1. **Repeat**
 - a. Merge clusters as $C_{a \cup b} = C_a \cup C_b$. Set new cluster's cardinality as $N_{a \cup b} = N_a + N_b$.
 - b. Insert a row and column containing the distances between the new cluster $C_{a \cup b}$ and the remaining clusters.
2. **Until** only one maximal cluster remains.

Further, Agglomerative clustering commonly employs a dendrogram that is a visual of the hierarchy of clusters. Each level of the dendrogram corresponds to certain clusters. To implement this method, we used `sklearn.cluster.AgglomerativeClustering` class.

3.3.4 K-means

K-means is one of the most popular and easily understood methods for clustering. It is not a hierarchical method, but rather a partitioning method because the number of clusters *K* must be pre- defined prior to use (Han et al 2021). The algorithm begins by choosing *K* data points randomly, which become centroids. Each data point is then assigned to the nearest centroid using a computed distance. In this study, and in many, the Euclidean distance is used.

$$\text{Euclidean Distance} = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

The Euclidian distance measures the distance from one point to another in Euclidean Space. When all points are assigned to their respective clusters, the centroids, or centers of each cluster, must be updated. This is done by finding the mean of all points that belong to each cluster. From here, the process repeats until either the predetermined number of iterations is reached, or the centroid values begin to stop changing.

The K -means algorithm works as follows (adapted from Aggarwal and Reddy, 2014):

1. Select K points as initial centroids.
2. **Repeat**
 - a. Form K clusters by assigning each point to its closest centroid based on chosen method.
 - b. Update the centroid of each cluster.
3. **Until** convergence criterion is met.

Primarily, the K -means algorithm employs the Sum of Squared Errors (SSE) function and works to minimize the function by finding the ideal clustering. Below is the mathematical formulation for the SSE function. Given dataset $D = \{x_1, x_2, \dots, x_N\}$, and the clustering after using K -means represented as $C = \{C_1, C_2, \dots, C_K\}$, the SSE formula is below. Additionally, note that c_k is the centroid of cluster C_k .

$$SSE(C) = \sum_{k=1}^K \sum_{x_i \in C_k} \|x_i - c_k\|^2$$

$$c_k = \frac{\sum_{x_i \in C_k} x_i}{|C_k|}$$

While K-means must have predefined clusters, there are ways to find the optimal number of clusters. One method is called the Elbow Method. This method produces a graph of within-cluster sum of squares (WCSS) also known as inertia. At a certain point, the graph of WCSS begins to slow its rate of change. This point is called the “elbow” and indicates the optimal number of clusters for the given data.

To see which clustering method we wanted to choose, we decided to implement all four above and compare the clustering results.

3.4 Implementation

Four different scripts were created for each clustering mechanism as follows. All methods required functions `load_images`, `preprocess_images`, `extract_features`, `create_clusters_dict`, and `move_to_clusters_folder`.

```

def load_images(folder_path):
    image_files = [f for f in os.listdir(folder_path) if f.endswith(".png")]
    images = []
    for image_file in image_files:
        image_path = os.path.join(folder_path, image_file)
        image = cv2.imread(image_path)
        images.append(image)
    return images, image_files

def preprocess_images(images):
    # resize images
    resized_images = [cv2.resize(image, (100, 100)) for image in images]
    return resized_images

def extract_features(images, n_components=2):
    # Apply PCA to reduce dimensionality
    pca = PCA(n_components=n_components)
    flattened_images = [image.flatten() for image in images]
    features = pca.fit_transform(flattened_images)
    return features

```

1. The `load_images` method loads in images from a folder path, creating a list of all file names (`image_files`), and stores all images in a list (`images`) using OpenCV's `cv2.imread()`. It returns both the list of names, and the list of loaded images.
2. The `preprocess_images` method resizes all loaded images to a 100 x 100 pixel size with `cv2.resize()` and returns the list of `resized_images`.
3. The `extract_features` method extracts features from each image, flattening each image by converting the 2D array into a 1D array. Then, Principal Component Analysis can be used with size of 2 Principal Components to ensure lower dimensional data. This allows the pixel values to be used as features, which are the data points that will be clustered. The pixel values contain information about shapes, patterns, textures, and colors that can be captured from each image.

```

def create_clusters_dict(labels, image_files):
    clusters_dict = defaultdict(list)
    for label, file_name in zip(labels, image_files):
        clusters_dict[label].append(file_name)
    return clusters_dict

def move_to_clusters_folder(clusters_dict, source_folder, destination_folder):
    # This will clear any existing clusters
    if os.path.exists(destination_folder):
        rmtree(destination_folder)
    os.makedirs(destination_folder, exist_ok=True)
    for cluster, files in clusters_dict.items():
        cluster_folder = os.path.join(destination_folder, f"cluster_{cluster}")
        os.makedirs(cluster_folder, exist_ok=True)
        for file_name in files:
            source_path = os.path.join(source_folder, file_name)
            destination_path = os.path.join(cluster_folder, file_name)

            copyfile(source_path, destination_path)

```

4. The `create_clusters_dict` function takes in the cluster labels and the list of image names in `image_files`. It creates a dictionary where keys are the cluster labels and values are the names of image files.
5. The function `move_to_clusters_folder` aids in moving images into folders based on their cluster. Its parameters include the `cluster_dict` created before, along with the source and destination folders. This allows users to see what images belong in each cluster automatically.

3.4.1 DBSCAN

```
def cluster_images(features, epsilon, min_samples):
    dbscan = DBSCAN(eps=epsilon, min_samples=min_samples, metric='euclidean')
    labels = dbscan.fit_predict(features)
    return labels

def create_clusters_dict(labels, image_files):
    clusters_dict = defaultdict(list)
    for label, file_name in zip(labels, image_files):
        clusters_dict[label].append(file_name)
    return clusters_dict

def move_to_clusters_folder(clusters_dict, source_folder, destination_folder):
    os.makedirs(destination_folder, exist_ok=True)
    for cluster, files in clusters_dict.items():
        cluster_folder = os.path.join(destination_folder, f"cluster_{cluster}")
        os.makedirs(cluster_folder, exist_ok=True)
        for file_name in files:
            source_path = os.path.join(source_folder, file_name)
            destination_path = os.path.join(cluster_folder, file_name)
            copyfile(source_path, destination_path)

def visualize_clusters(features, labels):
    plt.scatter(features[:, 0], features[:, 1], c=labels, cmap='viridis', marker='o', s=10)
    plt.title('DBSCAN Clustering')
    plt.xlabel('Feature 1')
    plt.ylabel('Feature 2')
    plt.show()
```

1. DBSCAN's `cluster_images` method uses `sklearn.cluster.DBSCAN` class to initialize a DBSCAN object, with the necessary parameters, `epsilon`, `min_samples`, and identifies the Euclidean distance for the distance metric. It then fits the DBSCAN algorithm to the features extracted previously and returns the cluster labels.

```

def compute_distances_to_nearest_neighbors(features):
    # Compute distances to nearest neighbors
    neigh = NearestNeighbors(n_neighbors=2)
    neigh.fit(features)
    distances, _ = neigh.kneighbors(features)
    return distances[:, 1]

def plot_knee_curve(distances):
    # Sort the distances in ascending order
    sorted_distances = np.sort(distances)

    # Plot the knee curve
    plt.plot(sorted_distances)

    # Add labels and title
    plt.xlabel('Index')
    plt.ylabel('Distance to Nearest Neighbor')
    plt.title('Knee Curve for Determining Epsilon')

    # Show the plot
    plt.show()

def find_epsilon(distances):
    # Calculate the second derivative of distances
    second_derivative = np.diff(np.diff(distances))

    # Find the index corresponding to the maximum second derivative
    knee_index = np.argmax(second_derivative) + 1

    # Get the epsilon value corresponding to the knee point
    epsilon = distances[knee_index]
    print(epsilon)

    return epsilon

```

2. To find the Epsilon value, we use a method called the Knee Curve Plot (Yadav 2020). This method requires the functions above, `compute_distances_to_nearest_neighbors` and `plot_knee_curve`. The function `compute_distances_to_nearest_neighbors` uses `sklearn.neighbors.NearestNeighbors` class to compute the distances between features and their nearest neighbor point. Then, `plot_knee_curve` plots these sorted distances to visualize the knee curve as seen in Figure 6 below. The Epsilon value based on this method, is where the slope of the curve is at its “knee” point – or, where the second derivative is the highest. The next function, called `find_epsilon`, finds this second derivative and returns the value associated with epsilon. In this example, epsilon was about 95.25.

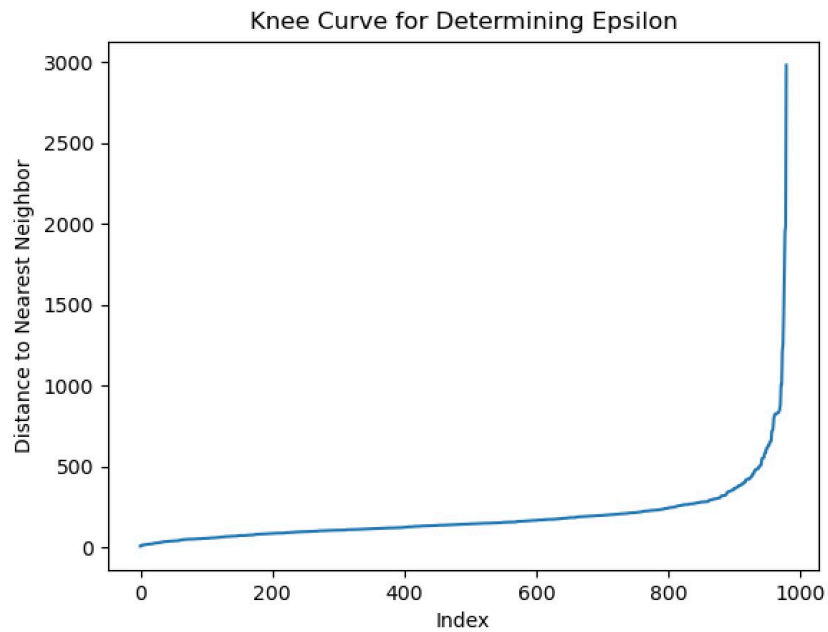


Figure 6: DBSCAN Knee Curve Plot

3. Additionally, the `min_points` is determined by using $2 * \text{Dimension of data}$ (Mullin, 2020). Since PCA made our data 2D, we chose `min_points = 4`.
4. The `visualize_clusters` function produces a plot (Figure 7) of feature data points with various colors corresponding to their assigned clusters.

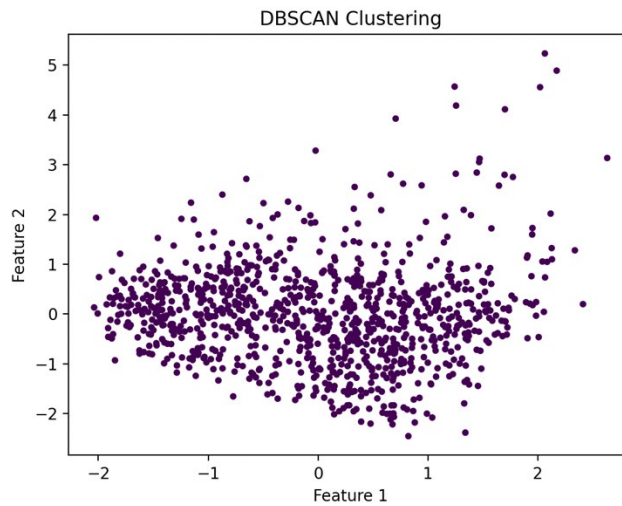


Figure 7: DBSCAN Clustering Visualization

3.4.2 OPTICS

```
def cluster_images(features):
    # Use StandardScaler to normalize features
    scaler = StandardScaler()
    features_normalized = scaler.fit_transform(features)

    # Use OPTICS for clustering on the normalized features
    optics = OPTICS(min_samples=5, xi=0.01, min_cluster_size=5)
    optics.fit(features_normalized)

    # Access reachability distances
    reachability_distances = optics.reachability_[optics.ordering_]

    # Plot reachability plot
    plt.figure(figsize=(10, 6))
    plt.plot(range(len(reachability_distances)), reachability_distances, marker='o', linestyle='-')
    plt.xlabel('Sorted Data Points')
    plt.ylabel('Reachability Distance')
    plt.title('Reachability Plot')
    plt.grid(True)
    plt.show()

    labels = optics.labels_
    return labels

def create_clusters_dict(labels, image_files):
    clusters_dict = defaultdict(list)
    for label, file_name in zip(labels, image_files):
        clusters_dict[label].append(file_name)
    return clusters_dict

def move_to_clusters_folder(clusters_dict, source_folder, destination_folder):
    # This will clear any existing clusters
    if os.path.exists(destination_folder):
        rmtree(destination_folder)
    os.makedirs(destination_folder, exist_ok=True)
    for cluster, files in clusters_dict.items():
        cluster_folder = os.path.join(destination_folder, f"cluster_{cluster}")
        os.makedirs(cluster_folder, exist_ok=True)
        for file_name in files:
            source_path = os.path.join(source_folder, file_name)
            destination_path = os.path.join(cluster_folder, file_name)
            copyfile(source_path, destination_path)
```

1. OPTICS's cluster_images method begins by first scaling features to normalize them with sklearn.preprocessing.StandardScaler. Then, the function employs sklearn.cluster.OPTICS class to initialize an OPTICS object, with the necessary parameters, xi, min_samples, and min_cluster_size. The xi parameter defines the minimum decrease in reachability distance for a cluster. The min_samples parameter identifies the minimum number of images per cluster. It then fits the OPTICS algorithm to the normalized features and returns the cluster labels. The function also produces a reachability plot seen below by first accessing reachability distances, then plotting these distances against the sorted data points.

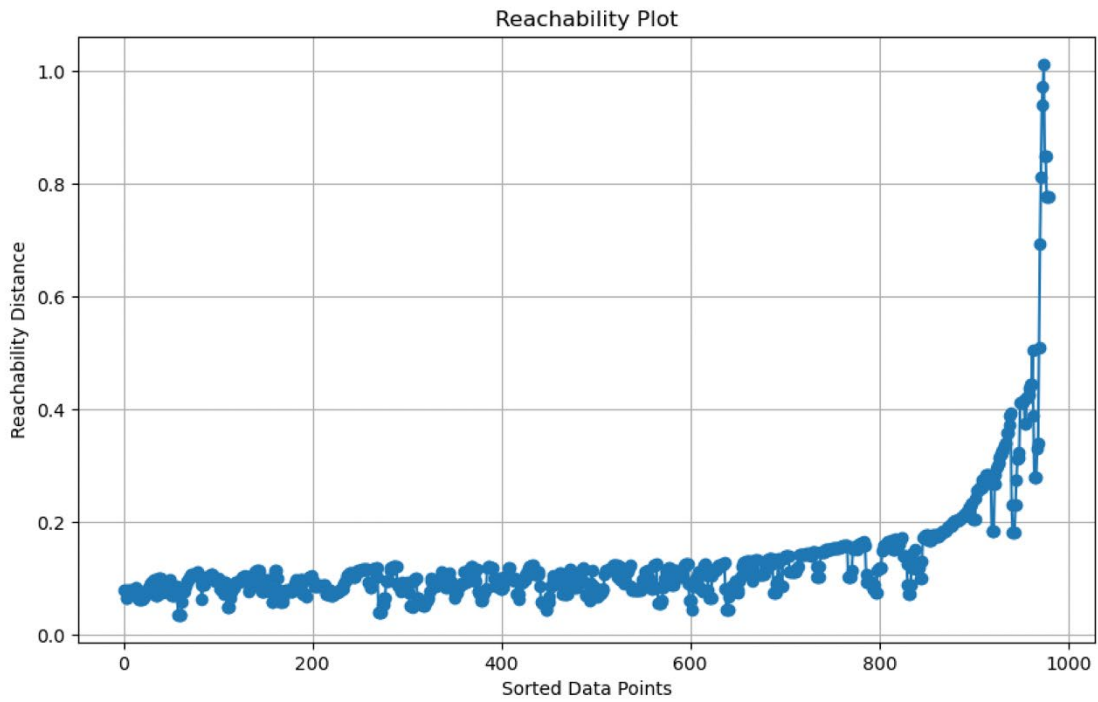


Figure 8: Reachability Plot

Points with similar reachability distances are more likely to be contained in one cluster. Spikes in reachability distances are characterized as separations between clusters, while valleys often represent clusters. The above plot does not appear to differentiate very well between significant clusters, as there are not clearly seen spikes and valleys.

3.4.3 Agglomerative

```
def hierarchical_clustering(features):  
    linkage_matrix = linkage(features, method='ward')  
    dendrogram(linkage_matrix)  
    plt.title('Hierarchical Clustering Dendrogram')  
    plt.xlabel('Sample Index')  
    plt.ylabel('Distance')  
    plt.show()  
  
def cluster_images(features, num_clusters):  
    agglomerative = AgglomerativeClustering(n_clusters=num_clusters, linkage='ward', affinity='euclidean')  
    labels = agglomerative.fit_predict(features)  
    return labels
```

1. The `hierarchical_clustering` method within the agglomerative clustering script computes the hierarchical clustering using the *Ward* approach using `scipy.cluster.hierarchy`'s linkage method.
2. This function then plots the linkage matrix in the form of a dendrogram. The dendrogram depicts how clusters are arranged and allows the user to choose cluster amounts based on this graph. Clusters numbers are chosen based on the visual number of splits in the dendrogram. Based on Figure 9, we chose 17.

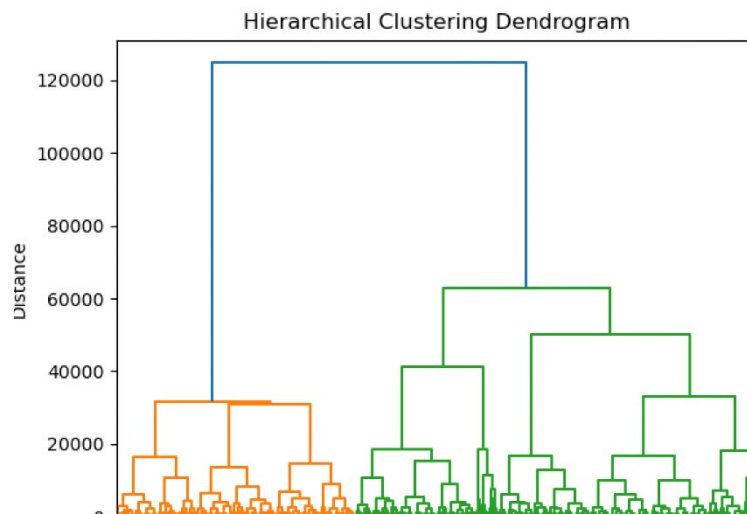


Figure 9: Agglomerative Clustering Dendrogram

Each level of the dendrogram represents a cluster. 17 clusters was chosen as the cutoff point because this is where the clusters began to merge closer to one another.

3. The `cluster_images` function creates an `AgglomerativeClustering` function from `sklearn.cluster`, with the given parameters of `num_clusters`, `ward` linkage, and `Euclidean` as the distance metric. It then fits the model and returns cluster labels.

3.4.4 K-Means

```
def find_optimal_clusters(features, max_clusters=10):
    k_values = range(1, max_clusters + 1)
    inertia = []

    for k in k_values:
        kmeans = KMeans(n_clusters=k)
        kmeans.fit(features)
        inertia.append(kmeans.inertia_)

    # Plot the Elbow curve
    plt.plot(k_values, inertia, marker='o')
    plt.title('Elbow Method for Optimal k')
    plt.xlabel('Number of Clusters (k)')
    plt.ylabel('Inertia')
    plt.show()

    # Choose the optimal number of clusters based on user input
    while True:
        try:
            optimal_k = int(input("Enter the optimal number of clusters based on the Elbow Method: "))
            break # Break out of the loop if the input is successfully converted to an integer
        except ValueError:
            print("Invalid input. Please enter a valid integer.")
    return optimal_k

KMeans_default = KMeans(n_init='auto')

def cluster_images(features, num_clusters):
    kmeans = KMeans(n_clusters=num_clusters)
    kmeans.fit(features)
    return kmeans.labels_
```

1. As discussed previously, *K*-means clustering must have a pre-defined number of clusters. However, to find the optimal number of clusters, the Elbow Method was performed within the `find_optimal_clusters` method. This method creates an elbow plot, where we must decide the ideal number of clusters from this. Figure 10 depicts the plot produced by our data. We identified the optimal number of clusters as 11, due to the slowing of the rate of inertia.
2. The `cluster_images` function creates a *K*-means object from `sklearn.cluster.KMeans` and fits the model to the extracted features, returning cluster labels.

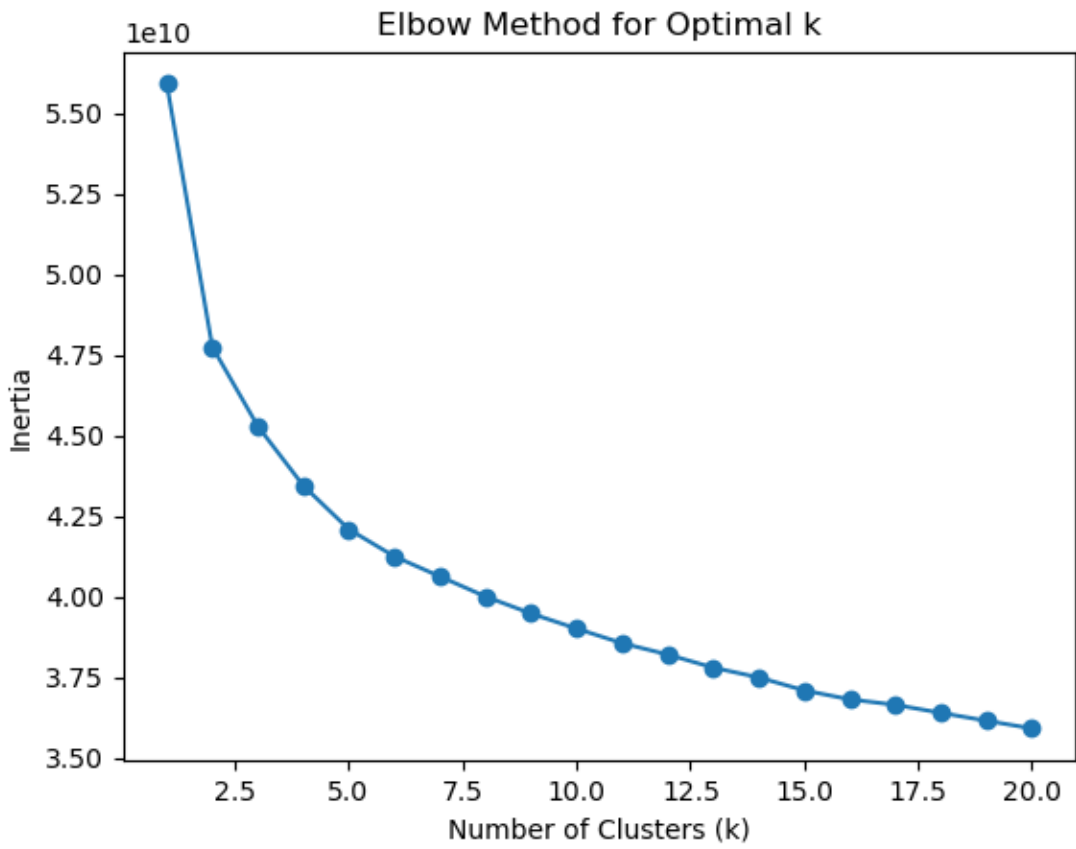


Figure 10: Elbow Plot

3.5 Choosing a Clustering Method

To choose a clustering method, we will examine all clusters created by each clustering algorithm and identify which best represents the dataset.

3.6 Identifying Trends

To identify trends in each cluster, the following procedure is used:

1. Look at each image in a cluster. Identify types of clothing, major colors, material and pattern. Tally up the totals of each color, type, and pattern/material and identify the top 3 of each category for each cluster.
2. For example, the image below is tallied as shown:



Figure 11: Example image for trend identification

Colors: Red, Black

Material/print: Polka dot, Leather, Floral

Type: Skirt, Blouse

3.7 Assessing Accuracy

To assess the accuracy of this trend identification mechanism, we will consult popular fashion magazines such as Vogue, Glamour and The Cut to see if our identifications match or stray away from their Spring/Summer 2024 trend identifications.

Chapter 4: Experiment

Each clustering method was performed, and subsequent clusters were analyzed. Below is a breakdown of cluster size and amount for each method.

Clustering Method	Total Number of Clusters	Cluster Sizes
DBSCAN	1	980
OPTICS	3	967, 8, 5
Agglomerative	17	90, 77, 85, 8, 70, 97, 21, 32, 56, 35, 88, 52, 162, 22, 43, 22, 20
<i>K</i> -means	11	124, 132, 94, 54, 110, 122, 18, 48, 101, 100, 77

Table 1: Cluster Size & Number Breakdown

Based on that table above, we see that OPTICS and DBSCAN are not the best options for this application of clustering methods, due to their low cluster sizes. One, or even three, clusters would not be enough to break down trends from the overall dataset. That leaves Agglomerative and *K*-means. Given the distribution of cluster sizes, and total number of clusters, we chose to stick with *K*-means as our primary method of clustering and use these clusters for further analysis.

4.1 K-means Cluster Image Examples

Cluster 0



Figure 12: Cluster 0 Example Images

Cluster 1



Figure 13: Cluster 1 Example Images

Cluster 2



Figure 14: Cluster 2 Example Images

Cluster 3



Figure 15: Cluster 3 Example Images

Cluster 4



Figure 16: Cluster 4 Example Images

Cluster 5



Figure 17: Cluster 5 Example Images

Cluster 6



Figure 18: Cluster 6 Example Images

Cluster 7



Figure 19: Cluster 7 Example Images

Cluster 8



Figure 20: Cluster 8 Example Images

Cluster 9



Figure 21: Cluster 9 Example Images

Cluster 10



Figure 22: Cluster 10 Example Images

4.2 Cluster Contents and Image Makeup

Cluster 0

Total images: 124

Top Colors	Frequency	Top Item types	Frequency	Top Material/Print	Frequency
White	28	Dress	31	Mesh/Sheer	20
Proportion	22.58%	Proportion	25%	Proportion	16.12%
Yellow	19	Jacket	30	Floral	16
Proportion	15.32%	Proportion	24.19%	Proportion	12.9%
Grey	16	Matching Set*	23	Abstract Designs	7
Proportion	12.9%	Proportion	18.55%	Proportion	5.64%

Table 2: Cluster 0 Frequency Breakdown

*A matching set is a top and bottom of an outfit that match in color and material; made to be worn together

Brands Represented (21/26): 7 Days Active, Balmain, Chanel, Erdem, Feben, Fendi, Ganni, Hermes, Isabel Marant, Jill Sander, KNWL, MiuMiu, Numero21, OperaSport, Rotate, Salvatore Ferragamo, Tory Burch, Versace, Wood Wood, Zimmerman

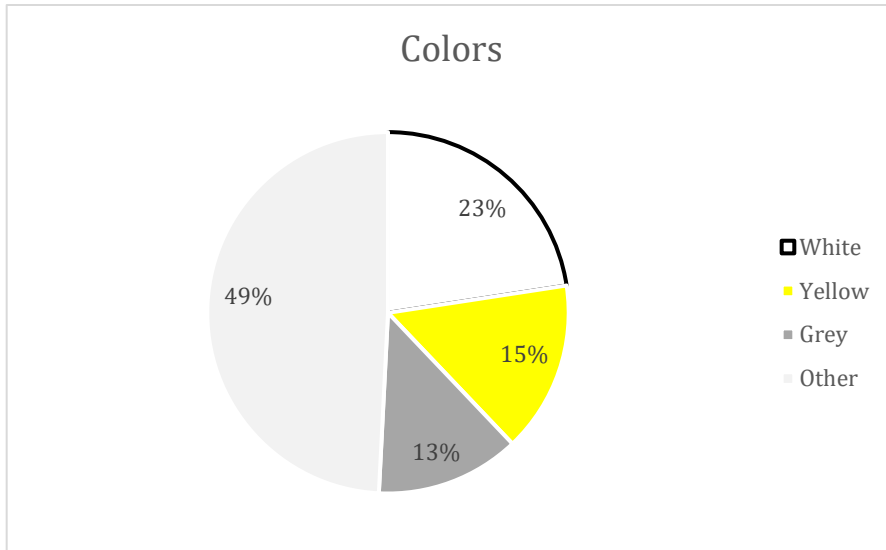


Figure 23: Cluster 0 Color Frequency Chart

This pie chart depicts the top three colors in terms of their frequency among the entire cluster. Colors not in the top 3 are collected under “Other”.

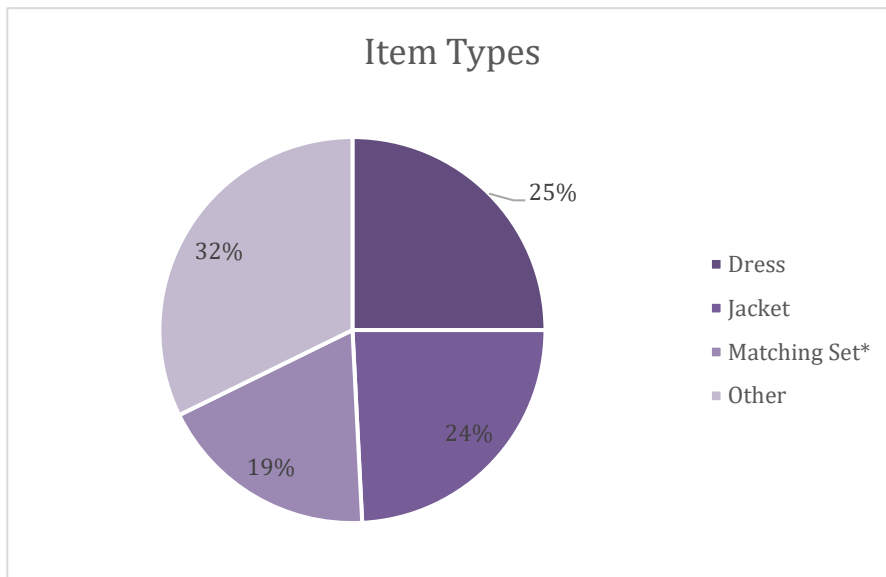


Figure 24: Cluster 0 Item Type Frequency Chart

This figure depicts the top three item styles in terms of proportion to the entire cluster. Items not in the top 3 are classified under “Other”.

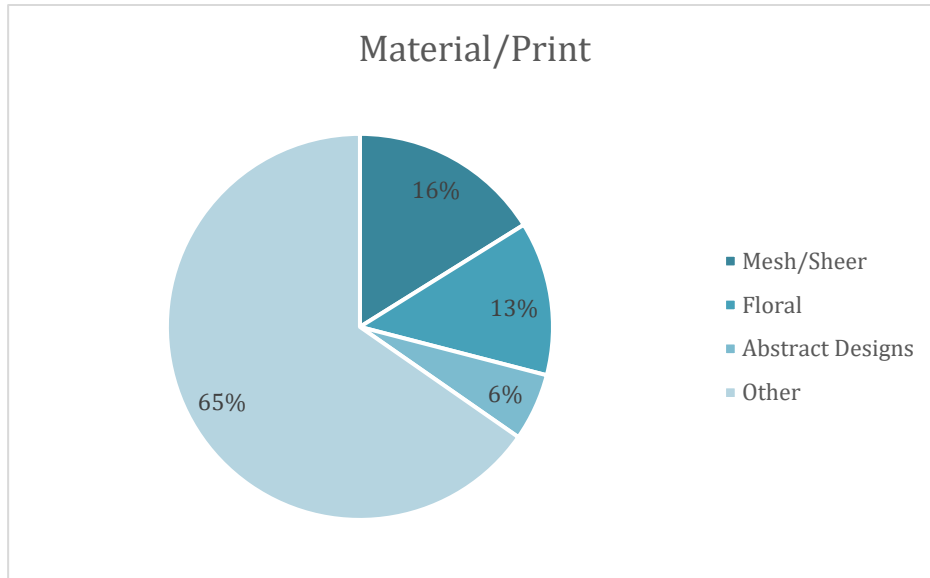


Figure 25: Cluster 0 Material/Print Frequency Chart

This figure is a pie chart that displays the proportion of images that include the top three material/prints within the cluster. Those materials/prints that did not make up the top three materials/prints are included under “Other”.

Cluster 1

Total images: 132

Top Colors	Frequency	Top Item types	Frequency	Top Material/Print	Frequency
Grey	22	Dress	16	Leather	27
Proportion	16.67%	Proportion	12.12%	Proportion	20.45%
Black	28	Jacket	21	Stripes	13
Proportion	21.21%	Proportion	15.9%	Proportion	9.85%
Red	18	Matching Set*	13	Mesh/ Sheer	13
Proportion	13.63%	Proportion	9.85%	Proportion	9.85%

Table 3: Cluster 1 Frequency Breakdown

*A matching set is a top and bottom of an outfit that match in color and material; made to be worn together

Brands Represented (22/26): 7 Days Active, Acne, Chanel, Dolce & Gabbana, Erdem, Feben, Ganni, Hermes, Isabel Marant, Jill Sander, KNWL, MiuMiu, Numero21, OperaSport, Prada, Rotate, Salvatore Ferragamo, Tory Burch, Undercover, Versace, Wood Wood, Yves Saint Laurent

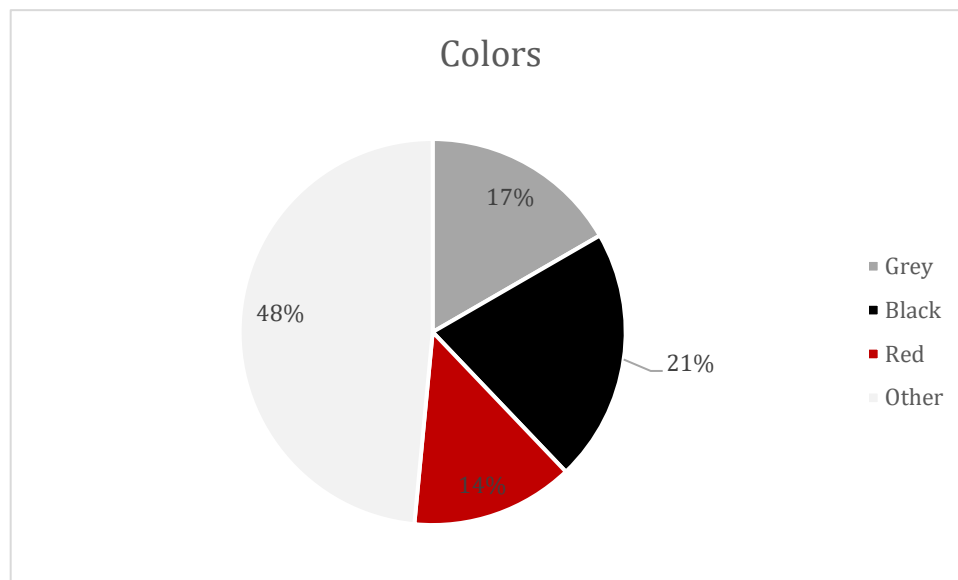


Figure 26: Cluster 1 Color Frequency Chart

This pie chart depicts the top three colors in terms of their frequency among the entire cluster. Colors not in the top 3 are collected under “Other”.

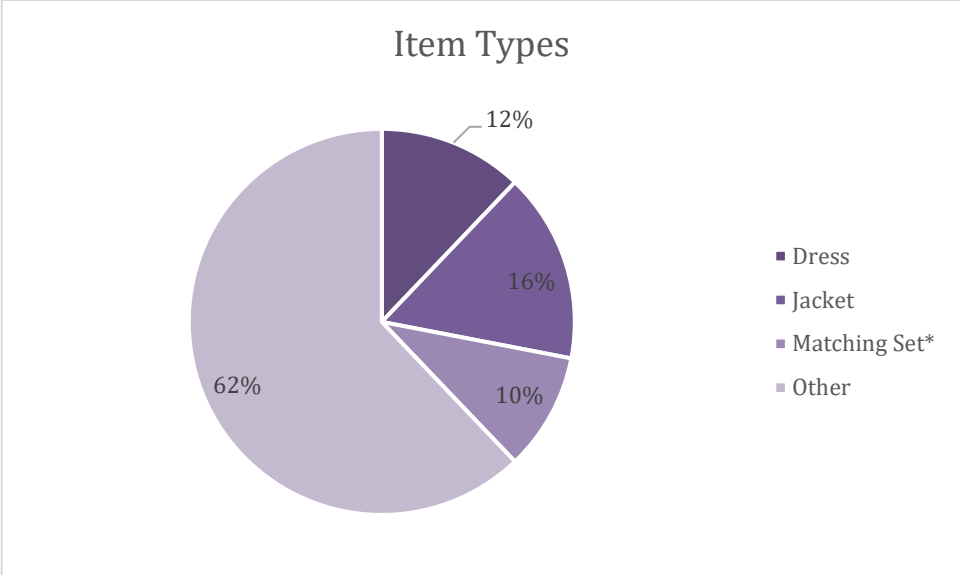


Figure 27: Cluster 1 Item Type Frequency Chart

This figure depicts the top three item styles un terms of proportion to the entire cluster. Items not in the top 3 are classified under “Other”.

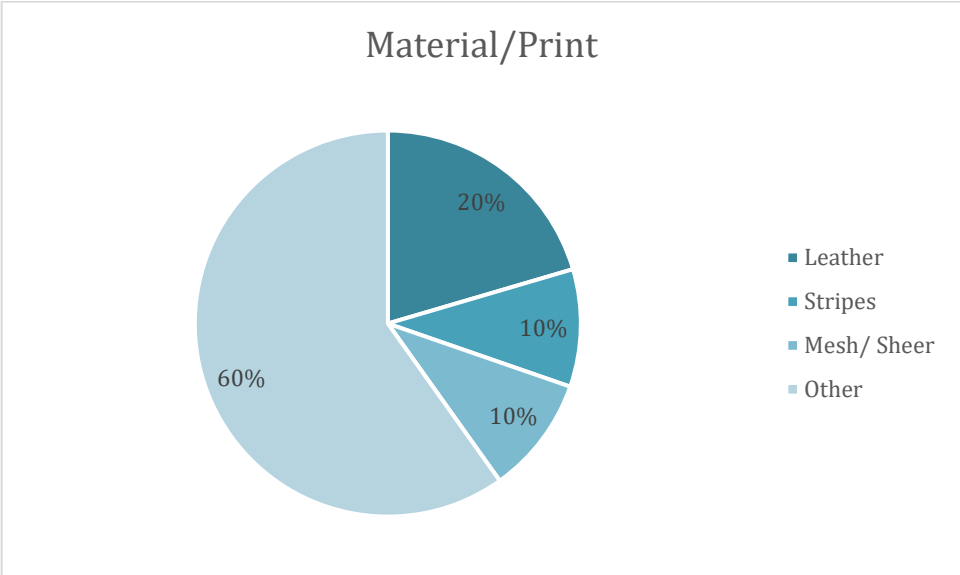


Figure 28: Cluster 1 Material/Print Frequency Chart

This figure is a pie chart that displays the proportion of images that include the top three material/prints within the cluster. Those materials/prints that did not make up the top three materials/prints are included under “Other”.

Cluster 2

Total images: 94

Top Colors	Frequency	Top Item types	Frequency	Top Material/Print	Frequency
Black	51	Dress	24	Mesh/Sheer	15
Proportion	54.26%	Proportion	25.53%	Proportion	15.96%
Purple	12	Pants	14	Floral	6
Proportion	12.77%	Proportion	14.89%	Proportion	6.38%
Blue	11	Jacket	14	Leather	4
Proportion	11.7%	Proportion	14.89%	Proportion	4.26%

Table 4: Cluster 2 Frequency Breakdown

Brands Represented (20/26): 7 Days Active, Acne, Balmain, Chanel, Dolce & Gabbana, Feben, Ganni, Hermes, Isabel Marant, Loewe, MiuMiu, Numero21, OperaSport, Prada, Rotate, Salvatore Ferragamo, Tory Burch, Undercover, Versace, Yves Saint Laurent

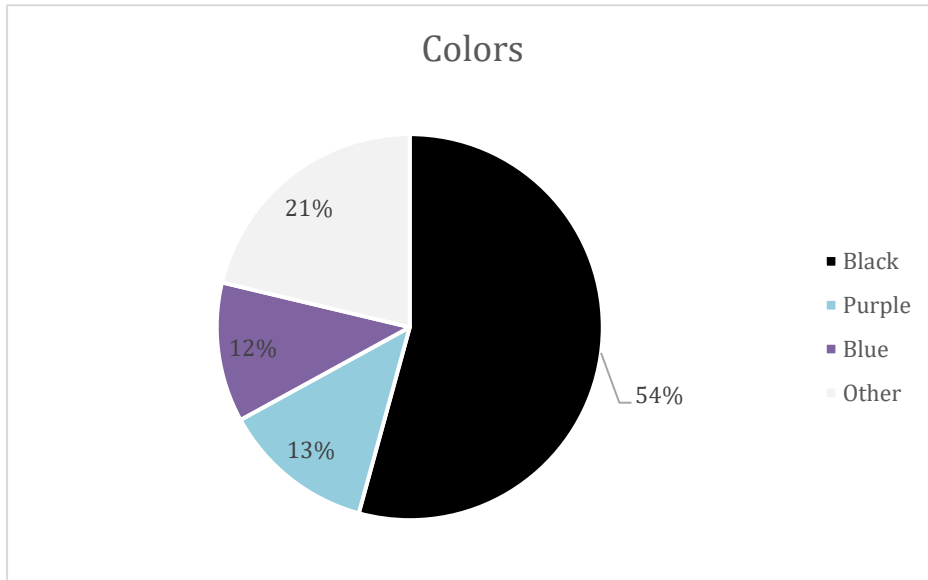


Figure 29: Cluster 2 Color Frequency Chart

This pie chart depicts the top three colors in terms of their frequency among the entire cluster. Colors not in the top 3 are collected under “Other”.

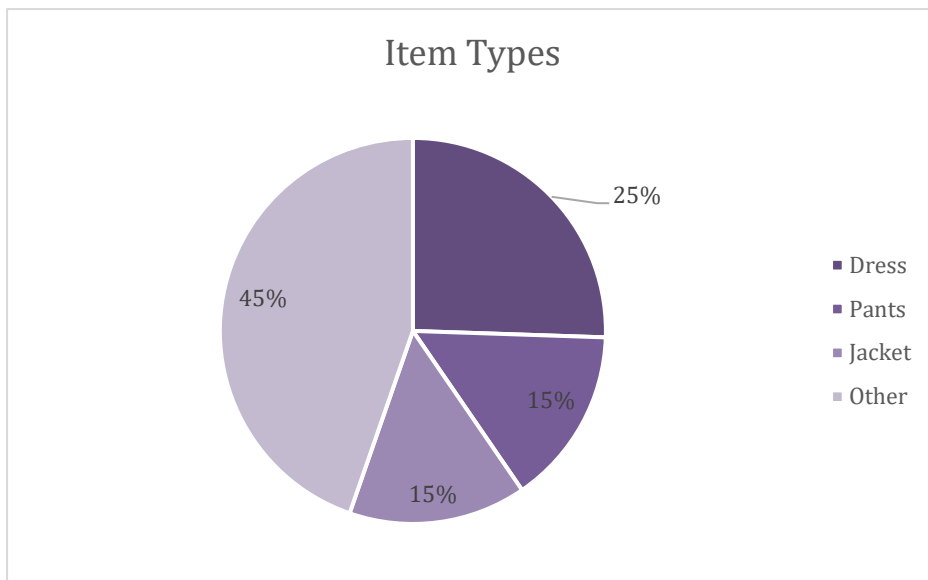


Figure 30: Cluster 2 Item Type Frequency Chart

This figure depicts the top three item styles in terms of proportion to the entire cluster. Items not in the top 3 are classified under “Other”.

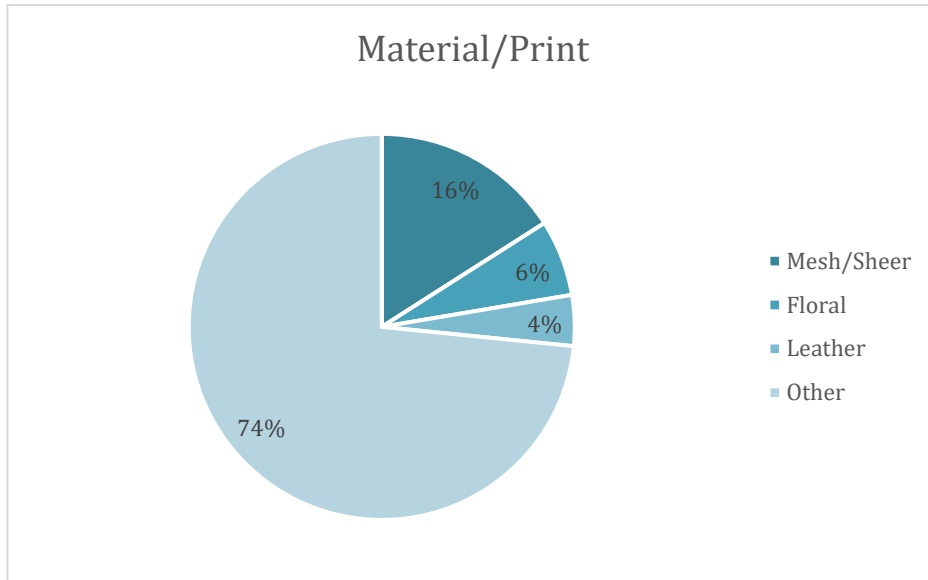


Figure 31: Cluster 2 Material/Print Frequency Chart

This figure is a pie chart that displays the proportion of images that include the top three material/prints within the cluster. Those materials/prints that did not make up the top three materials/prints are included under “Other”.

Cluster 3

Total images: 54

Top Colors	Frequency	Top Item types	Frequency	Top Material/Print	Frequency
Black	17	Dress	16	Leather	5
Proportion	31.48%	Proportion	29.63%	Proportion	9.25%
Red	9	Pants	9	Animal Print	4
Proportion	16.67%	Proportion	16.67%	Proportion	7.4%
Blue	6	Jacket	13	Mesh/Sheer	4
Proportion	11.11%	Proportion	24.07%	Proportion	7.4%

Table 5: Cluster 3 Frequency Breakdown

Brands Represented (15/26): 7 Days Active, Balmain, Erdem, Feben, Ganni, Hermes, Isabel Marant, Jill Sander, KNWL, Loewe, MiuMiu, Numero21, Tory Burch, Undercover, Yves Saint Laurent

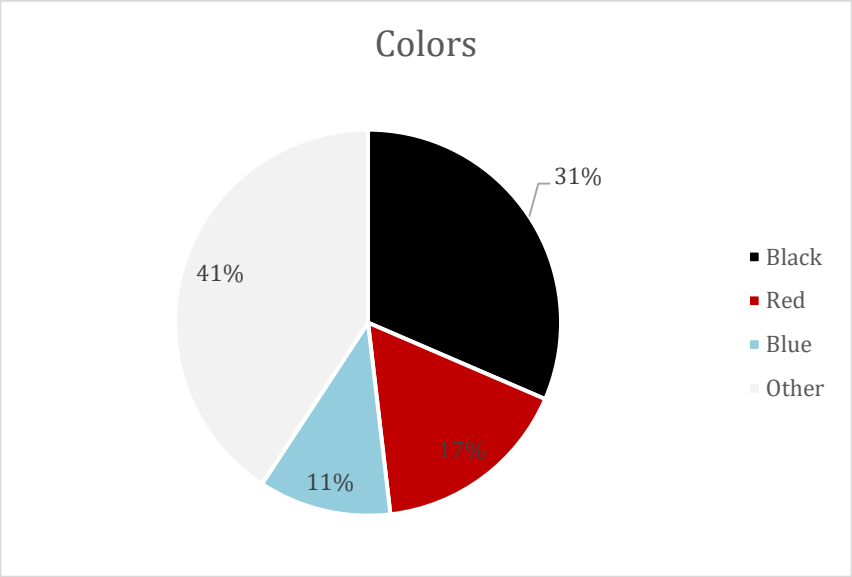


Figure 32: Cluster 3 Color Frequency Chart

This pie chart depicts the top three colors in terms of their frequency among the entire cluster. Colors not in the top 3 are collected under "Other".

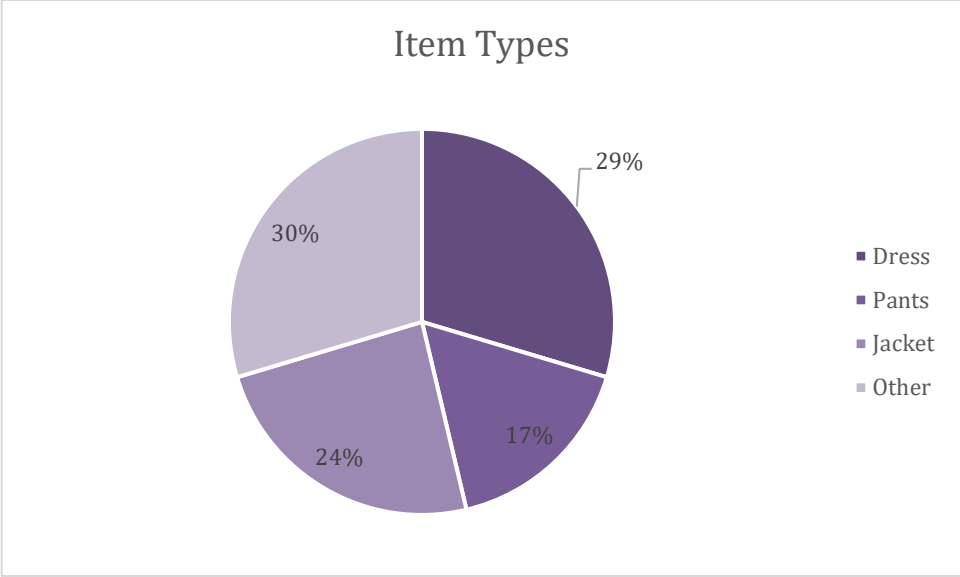


Figure 33: Cluster 3 Item Type Frequency Chart

This figure depicts the top three item styles in terms of proportion to the entire cluster. Items not in the top 3 are classified under “Other”.

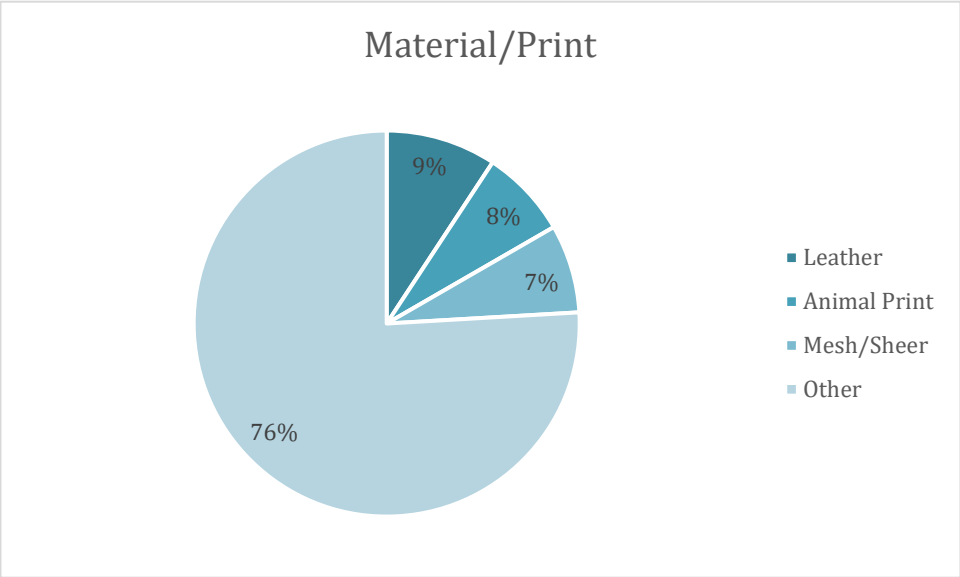


Figure 34: Cluster 3 Material/Print Frequency Chart

This figure is a pie chart that displays the proportion of images that include the top three material/prints within the cluster. Those materials/prints that did not make up the top three materials/prints are included under “Other”.

Cluster 4

Total images: 100

Top Colors	Frequency	Top Item types	Frequency	Top Material/Print	Frequency
Black	42	Dress	30	Leather	7
Proportion	42%	Proportion	30%	Proportion	7%
Grey	12	Pants	12	Sparkles	6
Proportion	12%	Proportion	12%	Proportion	6%
Blue	10	Jacket	11	Mesh/Sheer	15
Proportion	10%	Proportion	11%	Proportion	15%

Table 6: Cluster 4 Frequency Breakdown

Brands Represented (22/26): 7 Days Active, Acne, Chanel, Dolce & Gabbana, Erdem, Ganni, Hermes, Isabel Marant, Jill Sander, KNWL, Loewe, MiuMiu, Numero21, OperaSport, Prada, Rotate, Salvatore Ferragamo, Tory Burch, Undercover, Versace, Wood Wood, Yves Saint Laurent

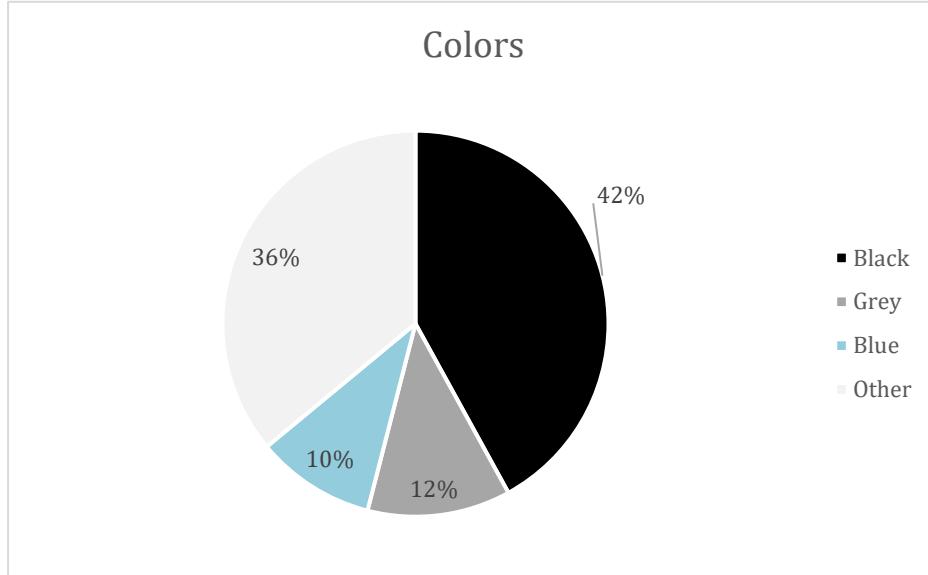


Figure 35: Cluster 4 Color Frequency Chart

This pie chart depicts the top three colors in terms of their frequency among the entire cluster. Colors not in the top 3 are collected under “Other”.

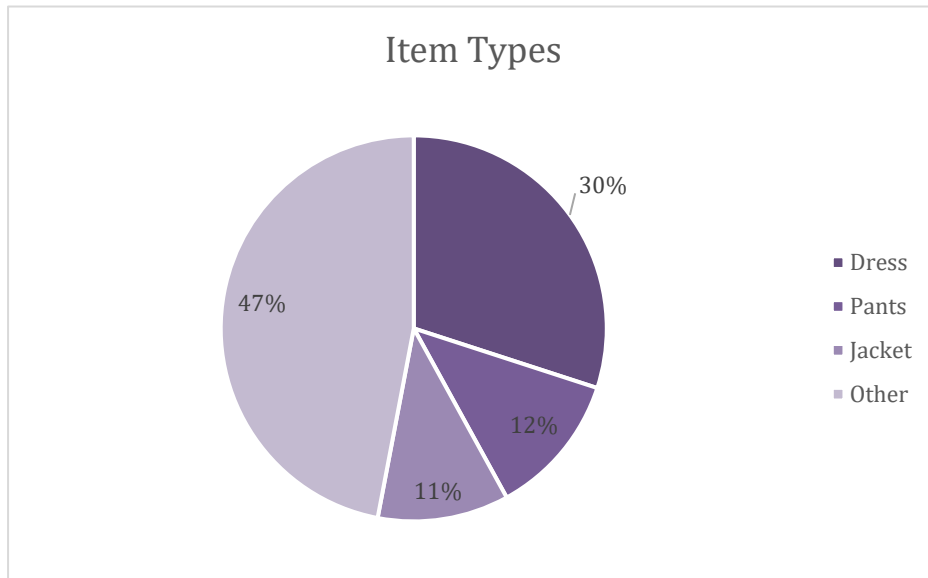


Figure 36: Cluster 4 Item Type Frequency Chart

This figure depicts the top three item styles in terms of proportion to the entire cluster. Items not in the top 3 are classified under “Other”.

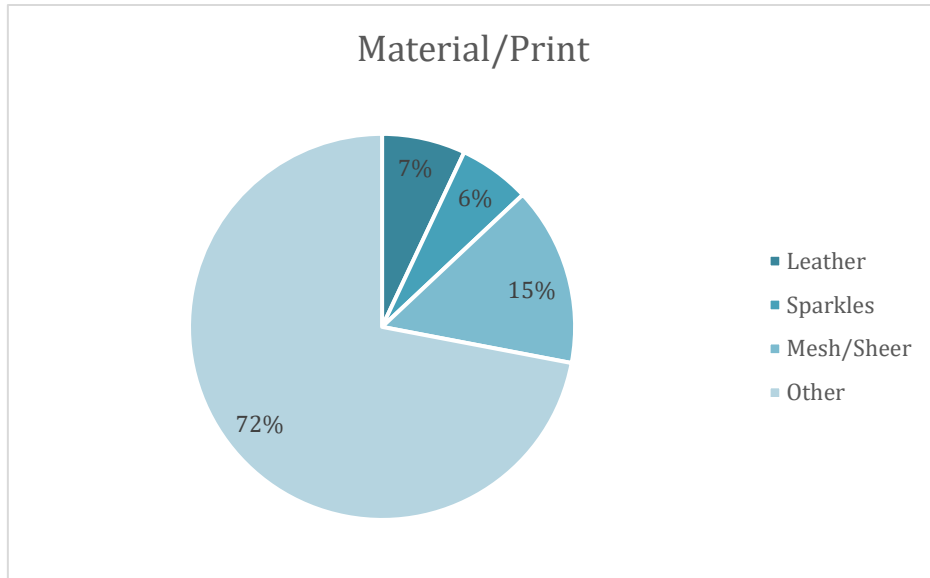


Figure 37: Cluster 4 Material/Print Frequency Chart

This figure is a pie chart that displays the proportion of images that include the top three material/prints within the cluster. Those materials/prints that did not make up the top three materials/prints are included under “Other”.

Cluster 5

Total images: 110

Top Colors	Frequency	Top Item types	Frequency	Top Material/Print	Frequency
Black	37	Dress	29	Leather	10
Proportion	33.64%	Proportion	26.36%	Proportion	9.1%
Grey	19	Shorts	12	Floral	10
Proportion	17.27%	Proportion	10.91%	Proportion	9.1%
Green	14	Jacket	12	Mesh/Sheer	10
Proportion	12.72%	Proportion	10.91%	Proportion	9.1%

Table 7: Cluster 5 Frequency Breakdown

Brands Represented (21/26): 7 Days Active, Acne, Balmain, Chanel, Dolce & Gabbana, Feben, Hermes, Isabel Marant, KNWL, Loewe, MiuMiu, OperaSport, Prada, Rotate, Salvatore Ferragamo, Tory Burch, Undercover, Versace, Wood Wood, Yves Saint Laurent

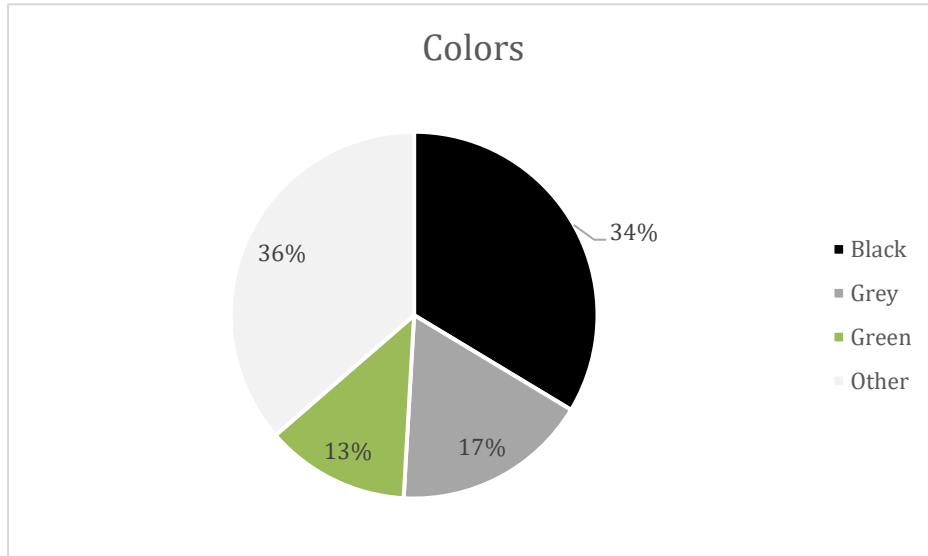


Figure 38: Cluster 5 Color Frequency Chart

This pie chart depicts the top three colors in terms of their frequency among the entire cluster. Colors not in the top 3 are collected under “Other”.

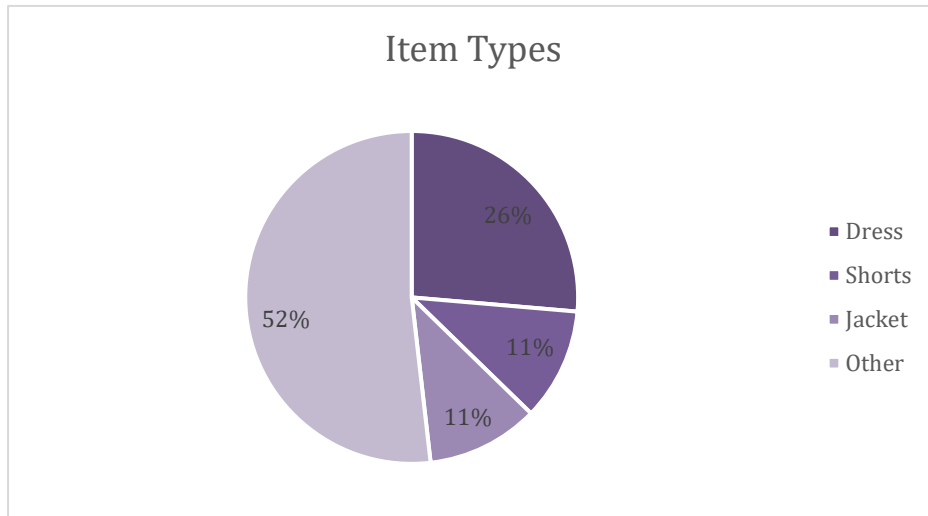


Figure 39: Cluster 5 Item Frequency Chart

This figure depicts the top three item styles in terms of proportions to the entire cluster. Items not in the top 3 are classified under “Other”

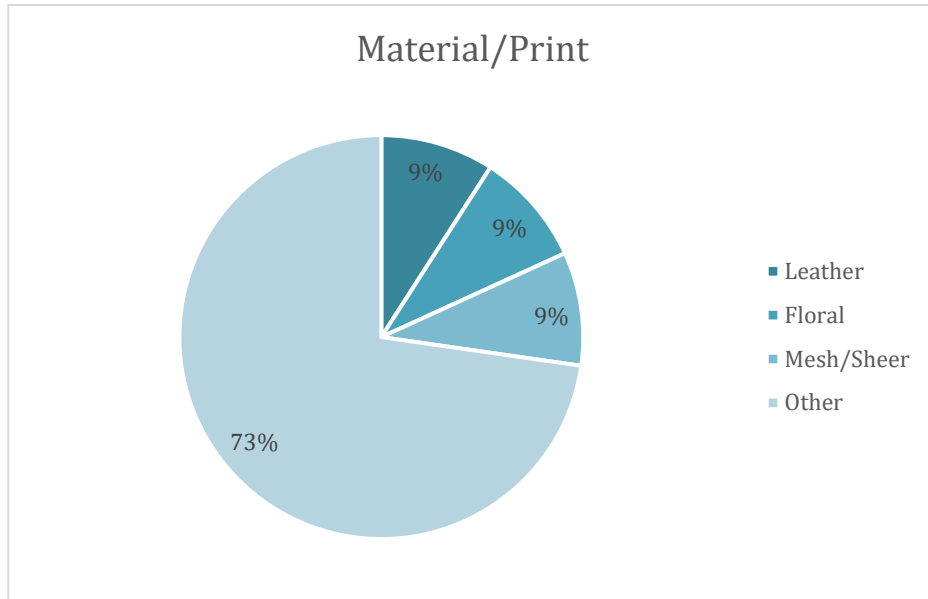


Figure 40: Cluster 5 Material/Print Frequency Chart

This figure is a pie chart that displays the proportion of images that include the top three material/prints within the cluster. Those materials/prints that did not make up the top three materials/prints are included under “Other”.

Cluster 6

Total images: 122

Top Colors	Frequency	Top Item types	Frequency	Top Material/Print	Frequency
White	60	Dress	60	Floral	25
Proportion	49.18%	Proportion	49.18%	Proportion	20.49%
Yellow	28	Pants	12	Stripes	4
Proportion	22.95%	Proportion	9.84%	Proportion	3.28%
Orange	17	Matching Set*	11	Mesh/Sheer	18
Proportion	13.93%	Proportion	9.02%	Proportion	14.75%

Table 8: Cluster 6 Frequency Breakdown

*A matching set is a top and bottom of an outfit that match in color and material; made to be worn together

Brands Represented (21/26): 7 Days Active, Acne, Balmain, Dolce & Gabbana, Feben, Fendi, Ganni, Hermes, Loewe, Miu Miu, Numero21, OperaSport, Prada, Rotate, Salvatore Ferragamo, Tory Burch, Undercover, Versace, Wood Wood, Yves Saint Laurent , Zimmerman

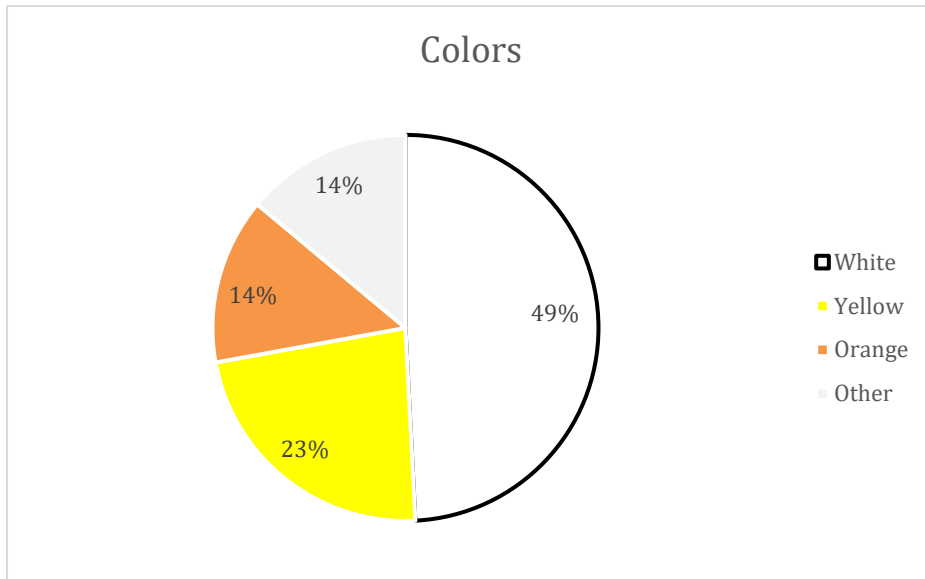


Figure 41: Cluster 6 Color Frequency Chart

This pie chart depicts the top three colors in terms of their frequency among the entire cluster. Colors not in the top 3 are collected under “Other”.

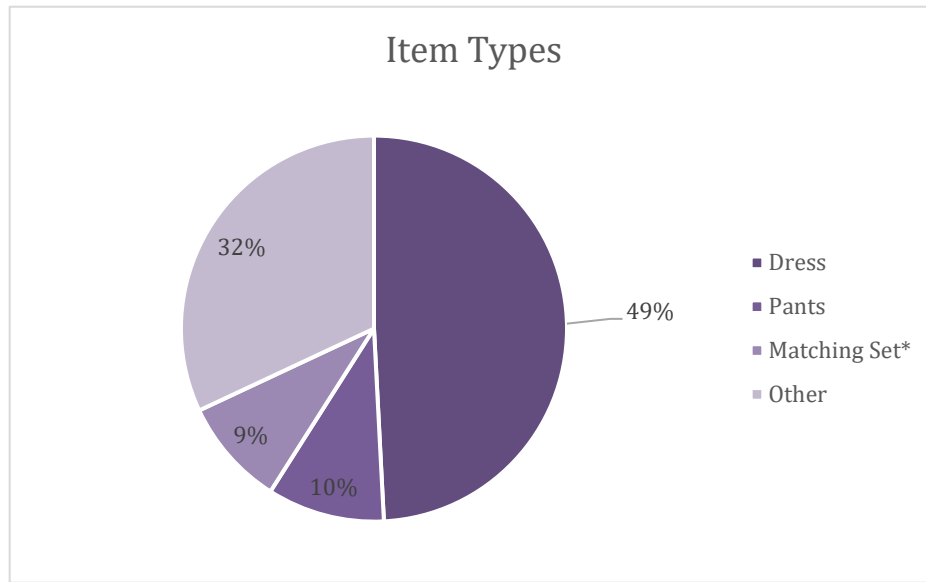


Figure 42: Cluster 6 Item Type Frequency Chart

This figure depicts the top three item styles in terms of proportion to the entire cluster. Items not in the top 3 are classified under “Other”.

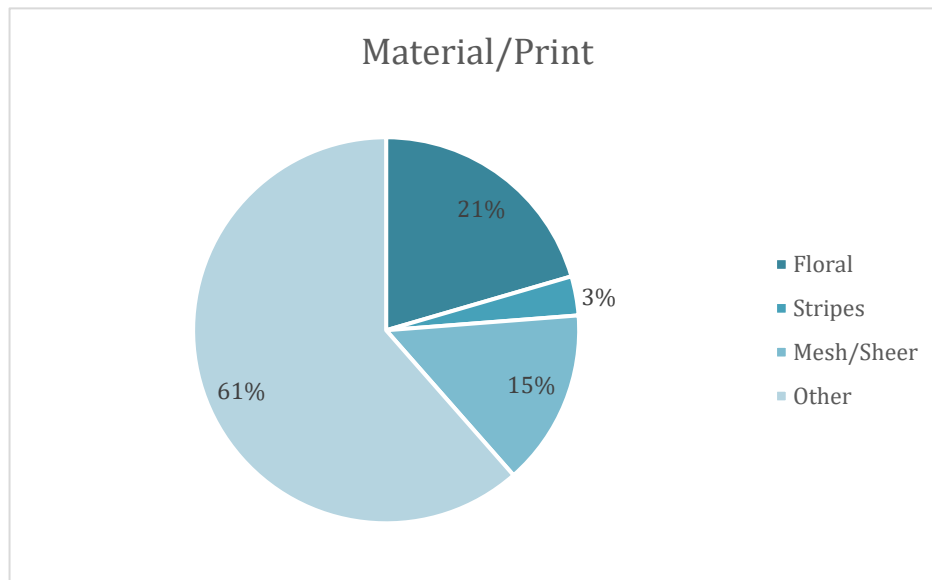


Figure 43: Cluster 6 Material/Print Frequency Chart

This figure is a pie chart that displays the proportion of images that include the top three material/prints within the cluster. Those materials/prints that did not make up the top three materials/prints are included under “Other”.

Cluster 7

Total images: 18

Top Colors	Frequency	Top Item types	Frequency	Top Material/Print	Frequency
Red	7	Dress	7	Floral	2
Proportion	38.89%	Proportion	38.89%	Proportion	11.11%
Black	3	Pants	9	Leather	6
Proportion	16.67%	Proportion	50%	Proportion	33.33%
Brown	3	Shorts	2	Mesh/Sheer	2
Proportion	16.67%	Proportion	11.11%	Proportion	11.11%

Table 9: Cluster 7 Frequency Breakdown

*A matching set is a top and bottom of an outfit that match in color and material; made to be worn together

Brands Represented (10/26): 7 Days Active, Hermes, Jill Sander, KNWL, Miu Miu, Rotate, Tory Burch, Wood Wood, Yves Saint Laurent , Zimmerman

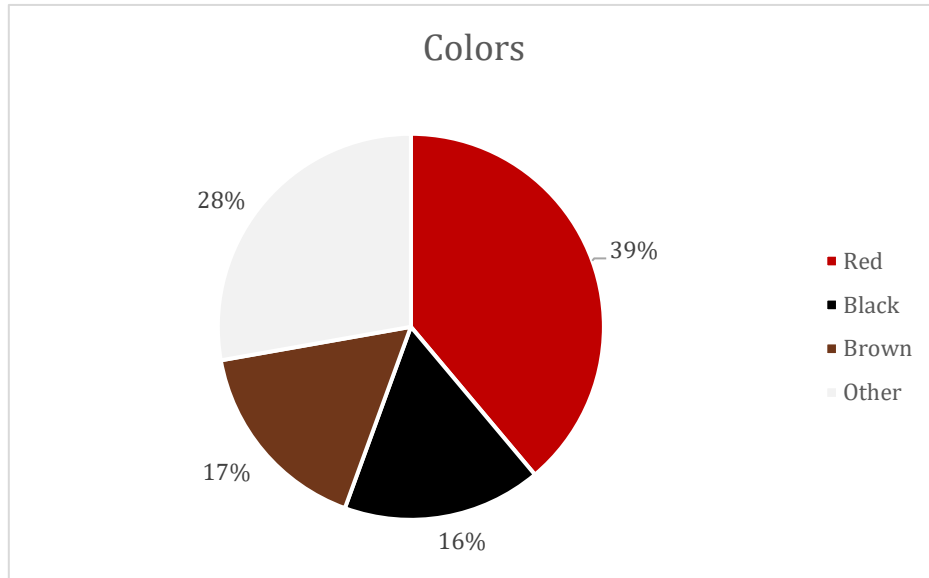


Figure 44: Cluster 7 Color Frequency Chart

This pie chart depicts the top three colors in terms of their frequency among the entire cluster. Colors not in the top 3 are collected under “Other”.

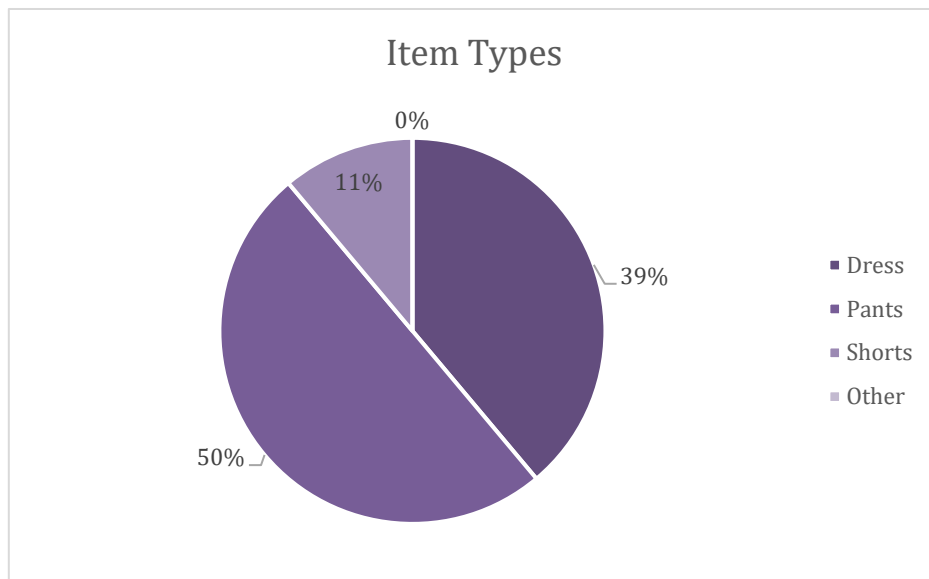


Figure 45: Cluster 7 Item Type Frequency Chart

This figure depicts the top three item styles in terms of proportion to the entire cluster. Items not in the top 3 are classified under “Other”.

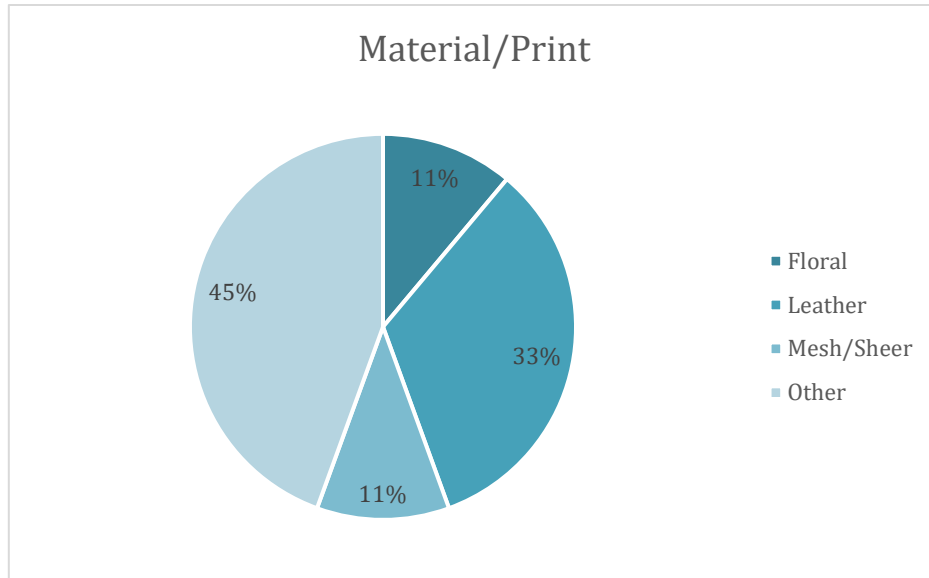


Figure 46: Cluster 7 Material/Print Frequency Chart

This figure is a pie chart that displays the proportion of images that include the top three material/prints within the cluster. Those materials/prints that did not make up the top three materials/prints are included under “Other”.

Cluster 8

Total images: 101

Top Colors	Frequency	Top Item types	Frequency	Top Material/Print	Frequency
White	22	Dress	20	Floral	10
Proportion	21.78%	Proportion	19.8%	Proportion	9.9%
Brown	18	Jacket	17	Stripes	16
Proportion	17.82%	Proportion	16.83%	Proportion	15.84%
Blue	18	Matching Set*	22	Leather	14
Proportion	17.82%	Proportion	21.78%	Proportion	13.86%

Table 10: Cluster 8 Frequency Breakdown

*A matching set is a top and bottom of an outfit that match in color and material; made to be worn together

Brands Represented (24/26): 7 Days Active, Acne, Balmain, Chanel, Dolce & Gabbana, Erdem, Feben, Ganni, Hermes, Isabel Marant, Jill Sander, KNWL, Loewe, Miu Miu, Numero21, OperaSport, Rotate, Salvatore Ferragamo, Tory Burch, Undercover, Wood Wood, Yves Saint Laurent , Zimmerman

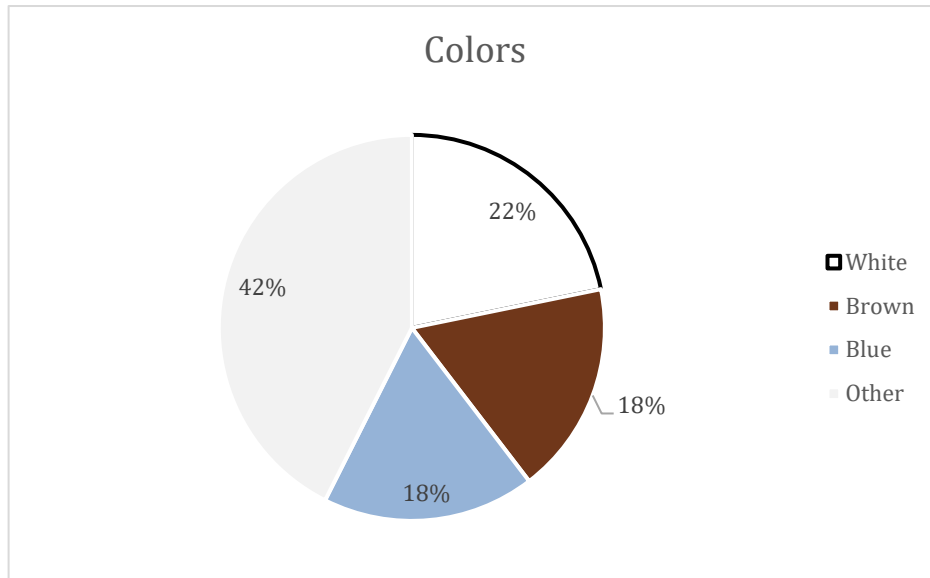


Figure 47: Cluster 8 Color Frequency Chart

This pie chart depicts the top three colors in terms of their frequency among the entire cluster. Colors not in the top 3 are collected under “Other”

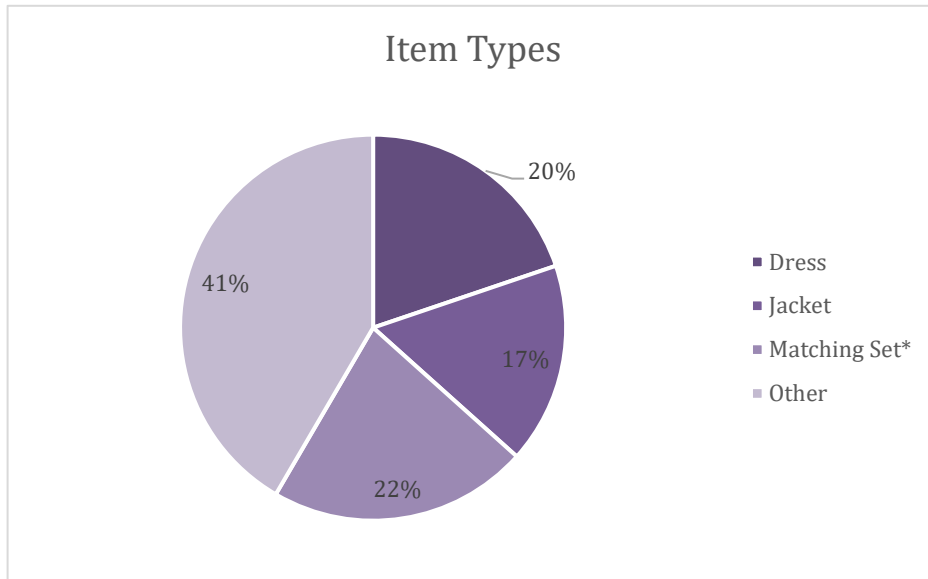


Figure 48: Cluster 8 Item Type Frequency Chart

This figure depicts the top three item styles in terms of proportion to the entire cluster. Items not in the top 3 are classified under “Other”.

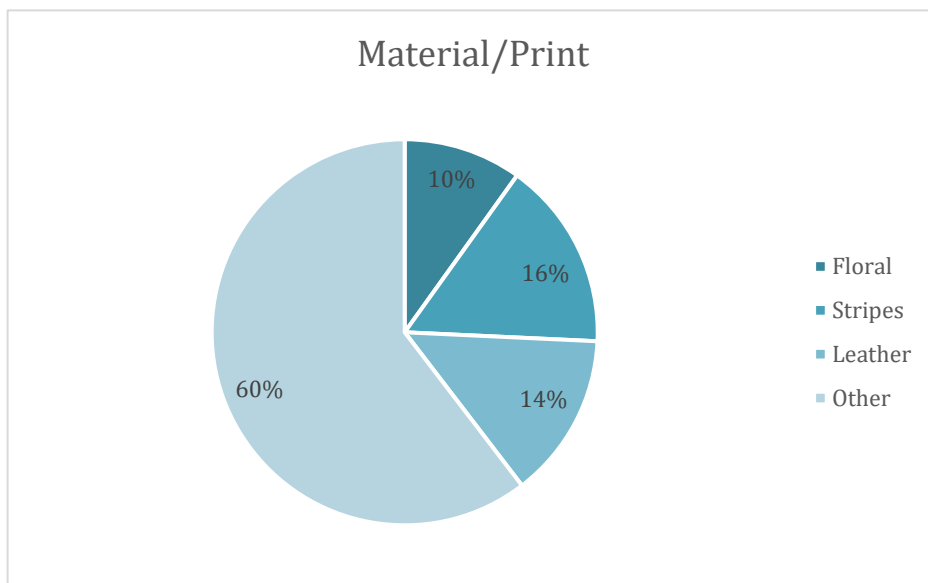


Figure 49: Cluster 8 Materials/Print Frequency Chart

This figure is a pie chart that displays the proportion of images that include the top three material/prints within the cluster. Those materials/prints that did not make up the top three materials/prints are included under “Other”.

Cluster 9

Total images: 77

Top Colors	Frequency	Top Item types	Frequency	Top Material/Print	Frequency
White	25	Dress	32	Floral	14
Proportion	32.47%	Proportion	41.56%	Proportion	18.18%
Green	19	Blouse	9	Stripes	9
Proportion	24.68%	Proportion	11.69%	Proportion	11.69%
Grey	15	Matching Set*	8	Mesh/Sheer	11
Proportion	19.48%	Proportion	10.39%	Proportion	14.29%

Table 11: Cluster 9 Frequency Breakdown

*A matching set is a top and bottom of an outfit that match in color and material; made to be worn together

Brands Represented (15/26): Acne, Balmain, Chanel, Dolce & Gabbana, Fendi, Hermes, Isabel Marant, Loewe, Prada, Rotate, Salvatore Ferragamo, Tory Burch, Yves Saint Laurent, Zimmerman

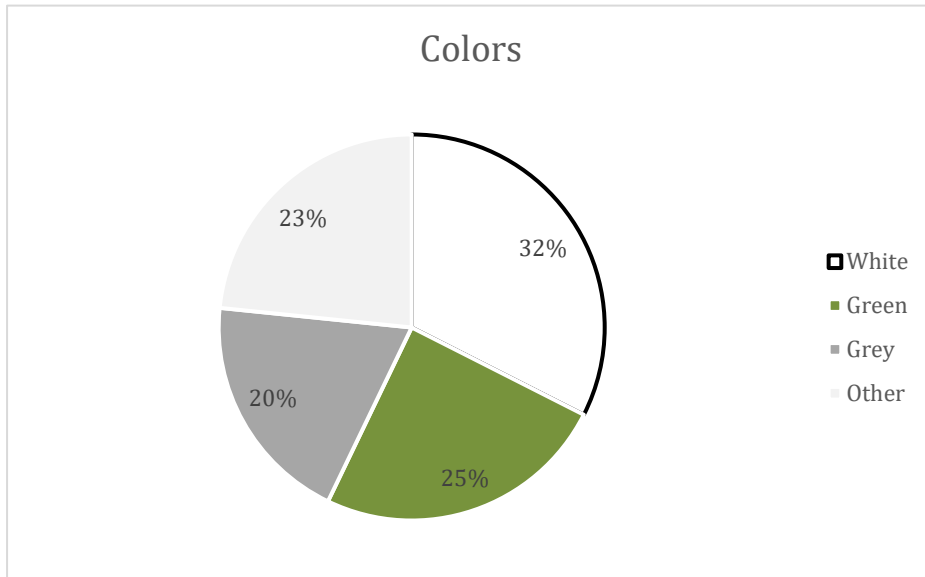


Figure 50: Cluster 9 Color Frequency Chart

This pie chart depicts the top three colors in terms of their frequency among the entire cluster. Colors not in the top 3 are collected under “Other”.

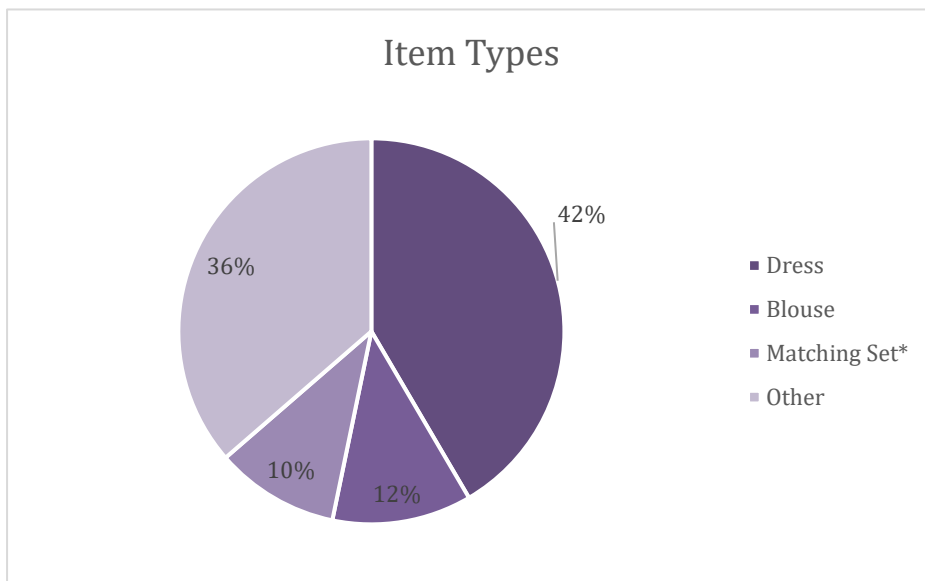


Figure 51: Cluster 9 Item Type Frequency Chart

This figure depicts the top three item styles in terms of proportion to the entire cluster. Items not in the top 3 are classified under “Other”.

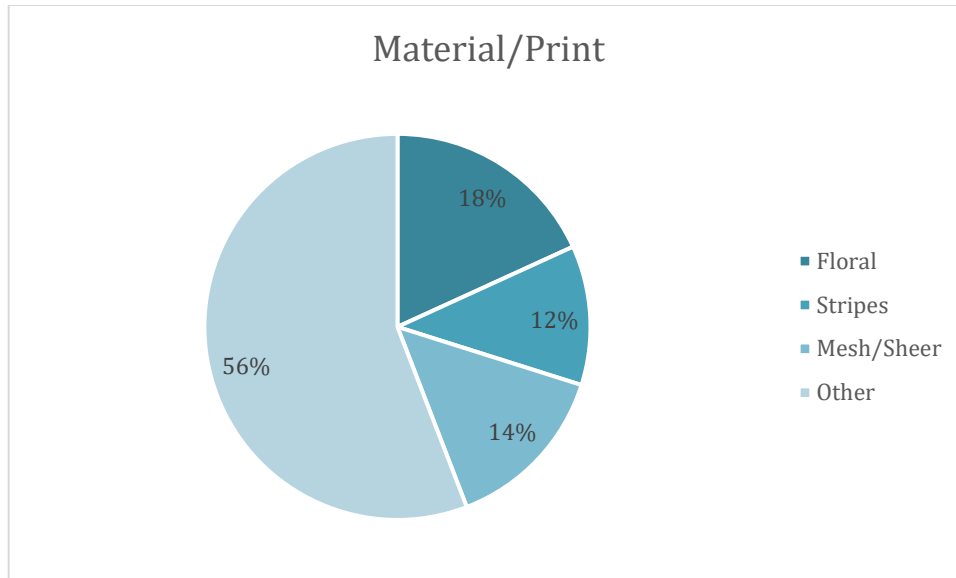


Figure 52: Cluster 9 Material/Print Frequency Chart

This figure is a pie chart that displays the proportion of images that include the top three material/prints within the cluster. Those materials/prints that did not make up the top three materials/prints are included under “Other”.

Cluster 10

Total images: 48

Top Colors	Frequency	Top Item types	Frequency	Top Material/Print	Frequency
Grey	15	Dress	25	Floral	8
Proportion	31.25%	Proportion	52.08%	Proportion	16.67%
Black	9	Jacket	10	Stripes	4
Proportion	18.75%	Proportion	20.83%	Proportion	8.33%
Pink	8	Blouse	7	Animal print	4
Proportion	16.67%	Proportion	14.58%	Proportion	8.33%

Table 12: Cluster 10 Frequency Breakdown

*A matching set is a top and bottom of an outfit that match in color and material; made to be worn together

Brands Represented (13/26): Chanel, Erdem, Feben, Fendi, Ganni, Hermes, Jil Sander, Numero21, OperaSport, Salvatore Ferragamo, Tory Burch, Undercover, Zimmerman

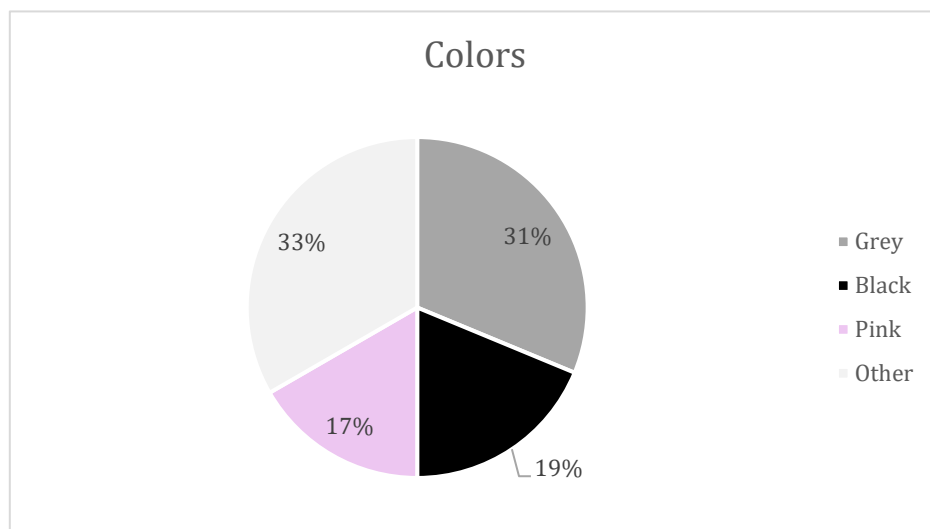


Figure 53: Cluster 10 Color Frequency Chart

This pie chart depicts the top three colors in terms of their frequency among the entire cluster. Colors not in the top 3 are collected under “Other”.

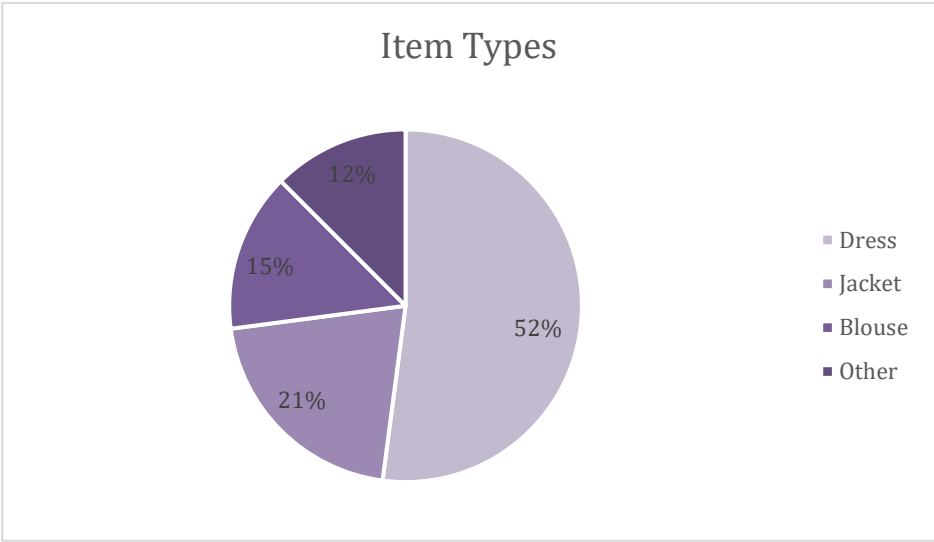


Figure 54: Cluster 10 Item Type Frequency Chart

This figure depicts the top three item styles in terms of proportion to the entire cluster. Items not in the top 3 are classified under “Other”.

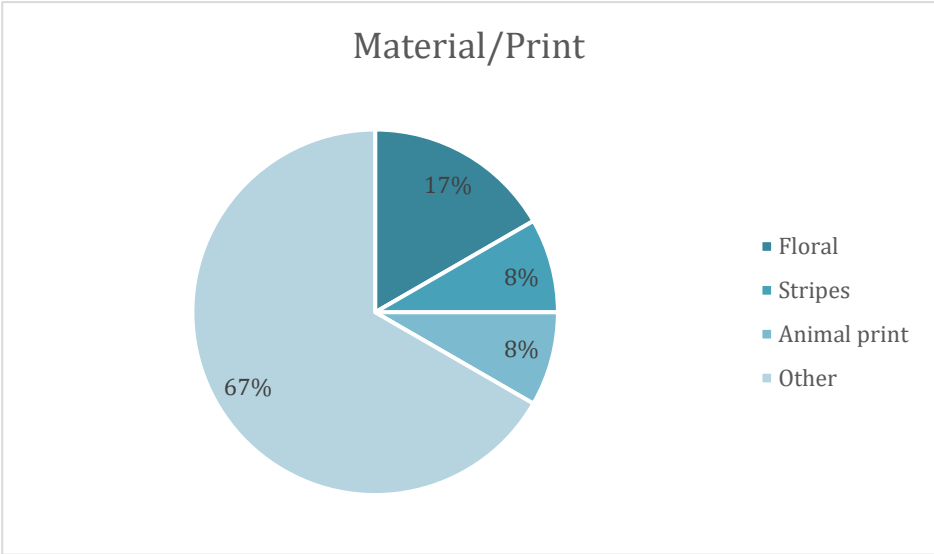


Figure 55: Cluster 10 Material/Print Frequency Chart

This figure is a pie chart that displays the proportion of images that include the top three material/prints within the cluster. Those materials/prints that did not make up the top three materials/prints are included under “Other”.

4.3 Results

Cluster 0 is represented mostly with the color white, dresses, and mesh or sheer materials. Cluster 1 consists of heavily the color black, jackets, and leather materials. Cluster 2 was made up of items with the color black, dresses, and mesh or sheer materials. Cluster 3 is represented by the color black, dresses, and leather materials. Cluster 4 consisted of the color black, dresses, and sheer/mesh items. Cluster 5 contained mostly black colors, dresses, and tied for materials that were leather, sheer/mesh, or floral patterns. Cluster 6 was made up of mostly white colors, dresses, and floral patterns. Cluster 7 is represented by the color red, pants, and leather materials. Cluster 8 is identified with having white colors, matching sets, and stripes. Cluster 9 consisted of white colors, dresses, and floral patterns. Cluster 10 contained grey colors, jackets, and floral patterns.

The most common top three colors identified (outside of black, white, or grey) were blue, which appeared in the top three colors of 4 clusters, and red, which appeared in the top three colors of 3 clusters. Green, brown, and yellow appeared in the top colors of two clusters each. Pink, Orange, and Purple appeared in one cluster's top three colors each.

Dresses appeared in every cluster's top three item types. Jackets appeared in nine of the clusters top three item types. Pants and Matching Sets were identified in five cluster's top three item types. Blouse and Shorts were identified in two of the clusters top three item types.

The most common materials/prints identified were floral and mesh/sheer, which both appeared in the top three materials/prints of seven clusters. Leather followed, appearing in the top three of six clusters. Stripes appeared in 5 clusters' top three materials/prints. Animal Print was present in 2 clusters top three materials/prints. Abstract designs appeared in one cluster's top three materials/prints.

Key Color Trends Identified	Key Item Type Trends Identified	Key Material/Print Trends Identified
Black	Dress	Mesh/Sheer
White	Jacket	Floral
Grey	Matching Set	Leather
Red	Pants	Stripes
Blue	Blouse	Animal Print

Table 13: Key Trend Table Summary


This table identifies the top five trends from all the clusters combined.


4.4 Accuracy & Evaluation

A key aspect of any technical application to a traditional method is assessing the accuracy of the new approach. Evaluating accuracy in this scenario is best done by comparing a traditional method of a magazine article identifying spring/summer trends to our results. We first selected *Glamour Magazine*'s February 2024 article, "Meet the Spring 2024 Fashion Trends You Should Know (and Shop) Now". Author Jake Henry Smith highlights 14 trends in the article, notably with some of the same identifications as we found. *Glamour*'s article mentioned burgundy, animal print, sky blue, stripes, and sheer layering as six of the 14 trends (Smith 2024).

We then evaluated *British Vogue*'s March 2024 article "The Key Spring/Summer 2024 Trends To Know Now" by Ellie Pithers. Pithers notes that this season's "palette was muted, with black and white blotting out" the typically brighter colors associated with summer fashion but noted red as one of the "few tones" to make it in the collections (Pithers 2024). The article highlights shorts, white dresses, roses/florals and sheer skirts among the ten mentioned trends.

Finally, *The Cut* published an article in February 2024 called “Five Spring 2024 Trends that Depict the Season” by fashion market editor Cortne Bonilla. Bonilla identifies white dresses, baby blue, the color black, and transparent (or sheer) materials in her writing (Bonilla 2024).

Trends also identified in above publications: 

Trends NOT identified in above publications: 

Key Color Trends Identified	Key Item Type Trends Identified	Key Material/Print Trends Identified
Black	Dress	Mesh/Sheer
White	Jacket	Floral
Grey	Matching Set	Leather
Red	Pants	Stripes
Blue	Blouse	Animal Print

Table 14: Trends Seen in Publications

This table identifies which of our identified trends also appeared in publications.

Chapter 5: Conclusion

Of the fifteen trends that our method identified, nine of them were also highlighted in one of the three fashion magazine publications above. This left six trends that we found that were not corroborated by traditional methods. Of those trends not identified, four of them were under the Item Type trend category. One color trend and one material/print trend were also not identified. This leads one to conclude that our cluster analysis struggled mainly to cluster and identify Item Types. Given *K*-means clustering does not identify specific items, and instead relies upon features such as colors and patterns, this result is understandable. However, it can be noted that the *K*-Means clustering method was highly successful in clustering images that helped to identify trends in colors and materials/prints.

Limitations of this method lie in differentiation between articles of clothing. In further research, we would suggest adopting a method that segments images into different clothing pieces prior to extracting features and clustering. This would ensure that each individual item of clothing is compared rather than each image. An example of methodology to consider would be “Neo-Fashion”, created in 2021 by Li Zhao, Muzhen Li, and Peng Sun as referred to in the previously mentioned literature review (Zhao, Li, and Sun 2024). Additionally, we might consider more detailed clothing Item Type labels, such as high waisted pants, or capris, instead of only pants to allow for more variety of identification.

Another suggestion for further research would be to include images from social media sites. This is because as social media usage grows, and platforms such as Instagram and TikTok are implementing shopping options for users, clothing trends are sure to emerge or at least be influenced by these sites. To approach this, we would recommend creating a data set of images

from runway fashion, and social media images to acquire a more comprehensive trend identification method.

We hope that this research can be applied to inventory management and purchasing for clothing retailers of any size. By understanding what trends are incoming for the future seasons, inventory can be purchased ahead of time to reflect what consumers more accurately will be interested in. Additionally, by ordering inventory that aligns with trends, ideally this will minimize clothing that will either be discounted or end up in landfills as textile waste. This approach has the potential to save retailers money and eliminate pollution around the world. Further, this trend identification method can be adapted to other consumer purchasing trends. For example, if similar trend identification methods are used for home products, makeup, or technology, production and inventory can be planned accordingly.

Overall, clustering methods such as *K*-means are a highly accessible and simplistic way to introduce machine learning tactics into industry and inventory management. They are easily understood and implemented, even for those with limited experience in machine learning. Our hope is that this research will promote better planning to reduce consumer product waste and encourage others to apply traditionally academic methods to real life issues or interests such as fashion trends.

References

- Aggarwal, Chaur C., and Chandan K. Reddy. 2014. *Data Clustering: Algorithms and Applications*.
- ArcGIS Pro. n.d. 'How Density-Based Clustering Works'. ArcGIS Pro. Accessed 4 May 2024. <https://pro.arcgis.com/en/pro-app/latest/tool-reference/spatial-statistics/how-density-based-clustering-works.htm>.
- Balchandani, Anita, Ewa Starzynska, David Barrelet, Gemma D'Auria, Felix Rolkens, and Imran Amed. 2024. 'The State of Fashion 2024'. McKinsey & Company. 2024. <https://www.mckinsey.com/industries/retail/our-insights/state-of-fashion>.
- Belcher, Byron T., Eliana H. Bower, Benjamin Burford, Maria Rosa Celis, Ashkaan K. Fahimipour, Isabela L. Guevara, Kakani Katija, et al. 2023. 'Demystifying Image-Based Machine Learning: A Practical Guide to Automated Analysis of Field Imagery Using Modern Machine Learning Tools'. *Frontiers in Marine Science* 10 (June): 1157370. <https://doi.org/10.3389/FMARS.2023.1157370/BIBTEX>.
- Benslama, Teissir, and Rim Jallouli. 2020. 'Clustering of Social Media Data and Marketing Decisions'. *Lecture Notes in Business Information Processing* 395: 53–65. https://doi.org/10.1007/978-3-030-64642-4_5/TABLES/5.
- Bonilla, Cortne. 2024. 'Five Spring 2024 Trends You Can Shop Now'. 2024. <https://www.thecut.com/article/spring-summer-2024-trends-buy-now.html>.
- Gaddamadugu, Indu. 2023. 'Applications of Technology in Fashion Trend Forecasting'. *Illumin Magazine*, 2023. <https://illumin.usc.edu/applications-of-technology-in-fashion-trend-forecasting/>.
- Greenfield, Nicole. 2023. 'New York Is Exposing the Fashion Industry for What It Is: A Climate Nightmare'. National Resource Defense Council. 2023.
- Huang, Fu Hsien, Hsin Min Lu, and Yao Wen Hsu. 2021. 'From Street Photos to Fashion Trends: Leveraging User-Provided Noisy Labels for Fashion Understanding'. *IEEE Access* 9: 49189–205. <https://doi.org/10.1109/ACCESS.2021.3069245>.
- James Hillman Fashion Consultancy. 2021. 'What Is a Fashion Trend?' James Hillman Fashion Consultancy. 2021. <https://www.jameshillman.co.uk/blog/2021/3/4/what-is-a-fashion-trend>.
- Maloney, Carolyn B. 2019. 'The Economic Impact of the Fashion Industry'. United States Joint Economic Committee. 2019. <https://www.jec.senate.gov/public/index.cfm/democrats/2019/2/the-economic-impact-of-the-fashion-industry>.
- MasterClass. 2021. 'Fashion Trend Forecasting: How Brands Predict New Styles'. 2021. <https://www.masterclass.com/articles/fashion-trend-forecasting-guide>.

- McAuley, Julian, Christopher Targett, and Anton van den Hengel. 2015. 'Image-Based Recommendations on Styles and Substitutes'. <https://doi.org/10.1145/2766462.2767755>.
- Norouzzadeh, Mohammad Sadegh, Anh Nguyen, Margaret Kosmala, Alexandra Swanson, Meredith S. Palmer, Craig Packer, and Jeff Clune. 2018. 'Automatically Identifying, Counting, and Describing Wild Animals in Camera-Trap Images with Deep Learning'. *Proceedings of the National Academy of Sciences of the United States of America* 115 (25): E5716–25. https://doi.org/10.1073/PNAS.1719367115/SUPPL_FILE/PNAS.1719367115.SAPP.PDF.
- Oyelade, Jelili, Iitunuoluwa Isewon, Funke Oladipupo, Olufemi Aromolaran, Efosa Uwoghiren, Faridah Aameh, Mmoses Aachas, and Ezekiel Aadebiyi. 2016. 'Clustering Algorithms: Their Application to Gene Expression Data'. *Bioinformatics and Biology Insights* 10 (November): 237–53. https://doi.org/10.4137/BBI.S38316/ASSET/IMAGES/LARGE/10.4137_BBI.S38316-FIG1.JPEG.
- Peng, Feng, and Kai Li. 2023. 'Deep Image Clustering Based on Label Similarity and Maximizing Mutual Information across Views'. *Applied Sciences* 2023, Vol. 13, Page 674 13 (1): 674. <https://doi.org/10.3390/APP13010674>.
- Pithers, Ellie. 2024. 'The 10 Key Spring/Summer 2024 Fashion Trends To Know Now | British Vogue'. March 2024. <https://www.vogue.co.uk/article/spring-summer-2024-fashion-trends>.
- Singh, Pawan Kumar, Yadunath Gupta, Nilpa Jha, and Aruna Rajan. 2019. 'Fashion Retail: Forecasting Demand for New Items'. <https://doi.org/10.1145/1122445>.
- Skenderi, Geri, Christian Joppi, Matteo Denitto, and Marco Cristani. 2024. 'Well Googled Is Half Done: Multimodal Forecasting of New Fashion Product Sales with Image-Based Google Trends'. <https://github.com/HumaticsLAB/GTM-Transformer>.
- Smith, Jake Henry. 2024. '12 Spring 2024 Fashion Trends to Start Shopping Now | Glamour'. April 2024. <https://www.glamour.com/story/2024-fashion-trends>.
- Viegas, Jennifer. 2011. 'Humans First Wore Clothing 170,000 Years Ago'. NBC News. 2011. <https://www.nbcnews.com/id/wbna40965564>.
- Vincent, Tom, Kenji Kawahara, Vladimir Antonov, Hiroki Ago, and Olga Kazakova. 2023. 'Data Cluster Analysis and Machine Learning for Classification of Twisted Bilayer Graphene'. *Carbon* 201 (January): 141–49. <https://doi.org/10.1016/J.CARBON.2022.09.021>.
- Vittayakorn, Sirion, Kota Yamaguchi, Alexander C. Berg, and Tamara L. Berg. 2015. 'Runway to Realway: Visual Analysis of Fashion'. *Proceedings - 2015 IEEE Winter*

Conference on Applications of Computer Vision, WACV 2015, February, 951–58.
<https://doi.org/10.1109/WACV.2015.131>.

Yadav, Mrinal. 2020. 'DBSCAN ALGORITHM'. Medium. 2020.
<https://mrinalyadav7.medium.com/dbscan-algorithm-c894701306d5>.

Yan, Cairong, Umar Subhan Malhi, Yongfeng Huang, and Ran Tao. 2019. 'Unsupervised Deep Clustering for Fashion Images'. *Communications in Computer and Information Science* 1027: 85–96. https://doi.org/10.1007/978-3-030-21451-7_8/TABLES/2.

Zhao, Li, Muzhen Li, and Peng Sun. 2024. 'Neo-Fashion: A Data-Driven Fashion Trend Forecasting System Using Catwalk Analysis'. *Clothing and Textiles Research Journal*. <https://doi.org/10.1177/0887302X211004299>.