Graphical user interface, application

Description automatically generated

Text mining – Fashion Trends Analysis

Authors: Eliza Pilarska

Subject: Data Mining

Lecturess: dr Nina Rizun

Gdańsk, 02.02.2023

Abstract

Project analysis was carried out to predict mass market fashion trends in next seasons. Usually, mass market adapts trends 1-2 years later after they are released at fashion shows. It is not random what brands propose, it can be predicted what will appear in shops, or what should appear to bring potential customers. By text mining methods high-fashion collection were analysed and main trends were presented.

Contents page:

1. Introduction

2. Theoretical basis

3. Research methodology

a. Main definitions used in the project

b. Research questions

c. Research Plan

III. Data preparation

IV. Data analysis

4.1. Matrixes creation

4.2. Keywords analysis

4.5. Cluster analysis

4.6. Cosine similarity

5. Results interpretation

6. Knowledge implementation

7. References

8. Code

1. Introduction

Fashion is a sector, that consists of art and craft, macroeconomics, business, sociology and psychology, and even history. Fashion industry [encompasses](https://www.merriam-webster.com/dictionary/encompasses) the design, manufacturing, distribution, [marketing](https://www.britannica.com/topic/marketing), retailing, [advertising](https://www.britannica.com/topic/advertising), and promotion of all types of apparel (men’s, women’s, and children’s) from the most rarefied and expensive haute couture (literally, “high sewing” – high fashion) and designer clothing to ordinary everyday apparel.

Every season new collection are released by hundreds of brands. There are 2 ways of trend adapting – from streets to high fashion and vice versa. The high fashion brands and their shows are usually lead for apparel retailers and producers, especially ones who target in mass market. But to follow trends – they must be recognized at first. Text mining can be a useful tool to recognize main trends among lots of new ideas and old practises - things that are repeated the most in brands newest concepts.

Fashion companies can use it to forecast what can be most desired in next seasons on fashion market, so they can produce similar product that would meet customers’ needs. Usually, trends taken from fashion weeks are released to mass market retail shops 1 or 2 years later. Early awareness and implementation of trends can help the company to become a mass market leader, instead copying and repeating other retailers.

2. Theoretical basis

Data mining is the process of extracting and discovering patterns in large [data sets](https://en.wikipedia.org/wiki/Data_set). Data mining is an [interdisciplinary](https://en.wikipedia.org/wiki/Interdisciplinary) subfield of [computer science](https://en.wikipedia.org/wiki/Computer_science) and [statistics](https://en.wikipedia.org/wiki/Statistics) with an overall goal of extracting information (with intelligent methods) from a data set and transforming the information into a comprehensible structure for further use.  Text mining is the process of deriving from [text](https://en.wikipedia.org/wiki/Plain_text) high-quality [information](https://en.wikipedia.org/wiki/Information) that is not directly and immediately available or is hard to reach. Written resources may include [websites](https://en.wikipedia.org/wiki/Website), [books](https://en.wikipedia.org/wiki/Book), [emails](https://en.wikipedia.org/wiki/Email), [reviews](https://en.wikipedia.org/wiki/Review), articles etc.

The main goal of the project is to extract specific information about the most probable fashion trends in 2023 and 2024 seasons.

Results would of course require a smart arrangement of trends into brand character and target customer, with recall of authors’ rights. The main limits and risks involve final customer, there is always a risk that the mass market will not adopt the trend, and from the other side, that another trend, invented beyond fashion weeks, will become stronger. Full forecast analysis should also contain street fashion and consider trends growing only among people. Data from people’s side are harder to collect, good source for data mining analysis would be social media.

3. Research methodology

The Corpus contains 42 text documents – reviews of the most popular fashion designers 2023/24 Women collections. Data were collected from WGSN platform. WGSN is a trend forecasting company, that consists of 6 parts – Insight, Fashion, Lifestyle & Interiors, Barometer, Instock, Styletrial. Besides their own forecast and analysis reports - a raw data can also be found there, which were used in the project. Texts are based on collection description released on WGSN Fashion, in Women Wear 2023 and 2024 section. The Corpus was created in December 2022. Similar data could be also probably collected from other sources, as WGSN access can be problematic due high fee. Other sources could be for examples fashion platforms (e.g. Vogue), social media, brands sites, brand shops. Designers were chosed subjectively, but the choice is in big part reinforced by various rankings about the most influential and most popular fashion brands. Considered brands: Acne Studio, Balenciaga, Balmain, Hugo Boss, Bottega Veneta, Burberry, Chanel, Chloe, Christian Dior, Coperni, Diesel, Dolce Gabbana, Ester Manas, Fendi, Giorgio Armani, Givenchy, Gucci, Hermes, Isabel arrant, Jil Sander, Louis Vuitton, Maison Margiela, Max Mara, Michael Kors, Miu Miu, Moschino, Nensi Dojaka, Off White, Ottolinger, Patou, Prada, Rokh, Saint Laurent, Schiaparelli, Simone Rocha, Stella McCartney, Tom Ford, Uma Wang, Valentino, Versace, Vetements, Victoria Beckham.

a. Main definitions used in the project

**Corpus** - text collection that has been brought together according to a certain set of predetermined criteria, divided into documents.

**Stopwords** – words that are irrelevant to principal information, usually common words like “the”, “have”, “go” etc.

Stemming -  process of reducing inflected (or sometimes derived) words to their [word stem](https://en.wikipedia.org/wiki/Word_stem), base or [root](https://en.wikipedia.org/wiki/Root_(linguistics)) form—generally a written word form.

**Zipf’s law** - in [probability](https://www.britannica.com/science/probability), assertion that the frequencies f of certain events are inversely proportional to their rank r.  It is mostly obtained to languages - the most frequent word will occur approximately twice as often as the second most frequent word, three times as often as the third most frequent word, etc. It also means that there is relatively not much words that occurs in text very often, and a lots of word that occurs very rarely.

**Bigram** - a sequence of two adjacent elements from a [string](https://en.wikipedia.org/wiki/String_(computer_science)) of [tokens](https://en.wikipedia.org/wiki/Token_(parser)), which are typically letters, syllables, or words. A bigram is an [n-gram](https://en.wikipedia.org/wiki/N-gram) for n=2. In project bigram is analysis of two word as a one term.

**Term frequency matrix, TF(w,d)** – amount of w words occurring in d document

**Binary system (Boolean model)** - method of mathematical expression which uses only two symbols: 1 – subject occurs, 0 – subjects does not occur. In the Boolean model, documents and queries are represented as sets of index terms, which are either present or absent in a document.

**Document frequency, df** – in how many documents the term occurs

**Inverse document frequency – idf** –   numerical statistic that is intended to reflect how important a word is to a [document](https://en.wikipedia.org/wiki/Document) in a collection or [corpus](https://en.wikipedia.org/wiki/Text_corpus). The more frequent term occurs in all documents the lower IDF value is. The more frequent term occurs within a small number of documents the highest IDF value is.

**Adjacency matrix** - a [square matrix](https://en.wikipedia.org/wiki/Square_matrix) used to represent a finite [graph](https://en.wikipedia.org/wiki/Graph_(discrete_mathematics)). The elements of the [matrix](https://en.wikipedia.org/wiki/Matrix_(mathematics)) indicate whether pairs of [vertices](https://en.wikipedia.org/wiki/Vertex_(graph_theory)) are [adjacent](https://en.wikipedia.org/wiki/Neighbourhood_(graph_theory)) or not in the graph. We can represent an unweighted graph with an [adjacency matrix](https://www.baeldung.com/cs/adjacency-matrix-list-complexity).

**Unweighted graph** - It’s an n x n matrix consisting of zeros and ones, where n is the number of nodes. The unweighted graphs tell us only if two nodes are linked.

**Adjacency List** -  collection of unordered lists used to represent a finite [graph](https://en.wikipedia.org/wiki/Graph_(discrete_mathematics)). Each unordered list within an adjacency list describes the set of neighbours of a particular [vertex](https://en.wikipedia.org/wiki/Vertex_(graph_theory)) in the graph.

**Weights** (regarding weighted graph) – association of each edge between terms with a real value w(e)

**Weighted matrix** - matrix whose element wij represents the weight of the edge between the i-th and j-th nodes

**Weighted graph** – a graph with weighted edges

**TF-IDF weighting, w(t,d)** – product of t frequency in d doc and inverse IDF frequency. It is calculated as a product of: W1 terms frequency (in the document – document frequency - tf), logarithm euler number and total numbers of documents divided by number of documents, n hich term W1 occurs in (corpus frequency - df). The more frequent term occurs within a small number of documents the highest TF-IDF value is. The fewer times the term occurs in a document or occurs in many documents – the lower is TF-IDF value. The lowest value is when term occurs in virtually all documents.

**Clustering** - grouping a set of objects in such a way that objects in the same group (called a cluster) are more similar (in some sense) to each other than to those in other groups (clusters).

**Hierarchical clustering** (connectivity model) is a method of [cluster analysis](https://en.wikipedia.org/wiki/Cluster_analysis) that seeks to build a [hierarchy](https://en.wikipedia.org/wiki/Hierarchy) of clusters. Strategies for hierarchical clustering generally fall into two categories:  
-Agglomerative: This is a "[bottom-up](https://en.wikipedia.org/wiki/Top-down_and_bottom-up_design)" approach: Each observation starts in its own cluster, and pairs of clusters are merged as one moves up the hierarchy. Merge clusters that are closest to each other’s until entire corpus forms a single cluster, each merge happens at a different distance.   
-Divisive: This is a "[top-down](https://en.wikipedia.org/wiki/Top-down_and_bottom-up_design)" approach: All observations start in one cluster, and splits are performed recursively as one moves down the hierarchy. The corpus is a single cluster, and the algorithm splits it until every doc is single cluster (inverse of above strategy).

**K-means clustering** (centroid model) - [partition](https://en.wikipedia.org/wiki/Partition_of_a_set) n observations into k clusters in which each observation belongs to the [cluster](https://en.wikipedia.org/wiki/Cluster_(statistics)) with the nearest [mean](https://en.wikipedia.org/wiki/Mean), serving as a prototype of the cluster. This results in a partitioning of the data space into [Voronoi cells](https://en.wikipedia.org/wiki/Voronoi_cell).

**Euclidean distance** – takes the shortest distance among two objects, it is in vector format, considers also word amount. Computes the root of squared differences between the coordinates between two objects.

**Cosine similarity** -  [measure of similarity](https://en.wikipedia.org/wiki/Measure_of_similarity) between two sequences of numbers. For defining it, the sequences are viewed as vectors in an [inner product space](https://en.wikipedia.org/wiki/Inner_product_space), and the cosine similarity is defined as the [cosine](https://en.wikipedia.org/wiki/Cosine) of the angle between them, that is, the [dot product](https://en.wikipedia.org/wiki/Dot_product) of the vectors divided by the product of their lengths. Little angle cosine is near 1, big angle cosine is near -1.

**Elbow Method** - In [cluster analysis](https://en.wikipedia.org/wiki/Cluster_analysis), method used in [determining the number of clusters in a data set](https://en.wikipedia.org/wiki/Determining_the_number_of_clusters_in_a_data_set). The method consists of plotting the [explained variation](https://en.wikipedia.org/wiki/Explained_variation) as a function of the number of clusters and picking the [elbow of the curve](https://en.wikipedia.org/wiki/Elbow_of_the_curve) as the number of clusters to use.

**Gap Statistics Method** - measures how far is the pooled within-cluster sum of squares around the cluster centers from the sum of squares expected under the null reference distribution of data. The expected value is estimated by simulating null reference data of characteristics of the original data but lacking any clusters in it. The optimal number of clusters is then estimated as the value of k for which the observed sum of squares falls farthest below the null reference.

b. Research questions

1. What are the most often used products, materials, colors, prints or styles – what can be predicted as the most trending in future seasons?

2. What could have caused such trend appearance? Can it be considered as recurrent or provoked by something?

3. What kind of products could have been predicted to appear, which are obvious?

4. Does corpus grouping help to recognize trends?

c. Research Plan

I. Finding data source

II. Collecting data

III. Data preparation before actual analysis – cleaning, integration, transformation, and reduction

IV. Data analysis, data modelling and simultaneous results evaluation in order to correct eventual issues.

4.1. Matrixes creation

4.2. Keywords analysis by correlation, histogram, Zipf’s Law, Wordclouds, Bigrams analysis

4.5. Cluster analysis

4.6. Cosine similarity

V. Results interpretation

VI. Knowledge implementation

III. Data preparation

From the text was removed extra whitepsaces, all numbers and punctation marks with additionaly „?” and „#” symbols, especially the hashtag removing was necessary, as they occured often due to fashion data character. The document was converted to lowercase. 337 stopowords were removed, among them designers’ names and surnames which already gave 65 words. Typical words for this kind of text, that were also removed, are seasons names (SS, AW, SS23, SS24 etc.) and also e.g. fashion, catwalk, collection, inspiration, label, buyer, paris, colour, style, fabric, tone, aesthetic, palette. Firstly, stemming was not planned to be used, due to its harmful impact on words, but afterall it has been also implemented, as benefits occurs to be more efficient.

Document length depends on number of apparel collections released by the designer. Clearly the Stella Mccartney doc is much longer text than others, also Vetements and Giorgio Armani are much shorter. Minimum document length was 20 words, maximum 795 words, the average 196,571 words. The same analysis without Stella Mccartney, Vetements and Giorgio Armani documents gives results of minimum – 41, maximum 462, average – 187.1795.

IV. Data analysis

4.1. Matrixes creation

* Document Term Matrix creation (DTM)

Obraz zawierający tekst

Opis wygenerowany automatycznie

The Corpus creates 42x3062 – dimension matrix, in which 95% of rows are zero, so we can say the very majoraty of it is sparse. The inspection of first and last 5 documents and columns from 140 to 145:

Obraz zawierający stół

Opis wygenerowany automatycznieObraz zawierający stół

Opis wygenerowany automatycznie

* Transposed Document Term Matrix (TDM, tf) creation. Statistics remain of course the same. Inspection regarding term and documents relation is transposed, first and last 5 documents, terms 170-175:

Obraz zawierający stół

Opis wygenerowany automatycznie

Obraz zawierający stół

Opis wygenerowany automatycznie

* Creation of a normalized DTM (dtm\_Norm, dtmr and dtmr1 from dtm), which eliminates the difference in documents length.

Limiting the matrix terms to only consider words that occurs at least 2 documents and their length is between 3 and 20 characters, with this method dtmr was created. Another method to create a normalized DTM, is to remove a lot of the uninteresting or infrequent words. Using this method, the dtmr1 was created, with 0.7 as a maximum allowed sparsity value.

Two versions of the new Document-Term Matrix, dtmr1 sparsity decreased to 63%:

Obraz zawierający tekst

Opis wygenerowany automatycznie

Below an investigation of dtmr1 with setting changed to show more terms, sparsity level now is 78%, settings on 0.85. This version dtmr1 was considered as more accurate, as 27 terms is relatively not much.

A picture containing text

Description automatically generated

New documents length: dtmr1 on the left, dtmr in the middle, old dtm length on the right.  
Obraz zawierający tekst

Opis wygenerowany automatycznieObraz zawierający tekst, paragon, zrzut ekranu

Opis wygenerowany automatycznieObraz zawierający tekst, paragon

Opis wygenerowany automatycznie

Dtmr: The minimum document length decreased to 16 (4 less), the maximum is now 567 (210 less), the average 147.1667 (46 less).

Dtmr1: The minimum document length decreased to 11, the maximum is now 319, the average 82.42857.

Stella Mccartney in both dtmr matrixes is still much longer document than other documents, very short documents can be now considered as Vetement, Giorgio Armani, and also Schiarapelli and Burberry.

* TF-IDF creation

Frequency matrix TF-IDF was created to evaluate significance of the terms in document context. However, most of uninteresting or infrequent words have been removed already.

Comparing TF-IDF with TF:

A screenshot of a computer

Description automatically generated with medium confidence

4.2. Keywords Analysis

List of words that occur not less than 15 times in the dtmr Corpus:

Obraz zawierający tekst

Opis wygenerowany automatycznie

List of words that occur not less than 15 times in the dtmr Corpus:

Obraz zawierający tekst

Opis wygenerowany automatycznie

* Relationships between words – correlation

Checking top 6 words strongest correlation to other words, with output correlation limit 0.6, which is considered as high correlation level, but sometimes it was adjusted to the word, receive more or less results. Used on dtm and dtmr1.



Vocabulary:

Silhouette – this word can help to define top trends from the other side – as the most common word appearing with it.

Dopaminebrights – dopamine brights are bold, eye-catching colours, that designers associate with joy and optimism

Matchingset – trend to have all garment parts from the same material and colour, matching with each others.

Lowimpact – low impact regarding environment

Some words, due to root lack, can not be defined, e.g. weve. Some word are rather common words, that didn’t removed from corpus, e.g. apply.

What can be observed from top words correlation? Key words are the most common words appearing in corpus. Their correlation shows which words are their most frequent acompanists. Correlation let to analyse what are further connections to top trends.

It can be observed that leather trend goes toward environment friendly accross most popular brands. It also connects with biker and moto – so motorcycle style leather is currently showed at fashionshows. Resurgence word suggest leather trend comeback. Performance blue as a trending leather colour.

The same analysis can be done with each top word.

* Histogram and Zipf’s Law

Words histogram with minimal corpus frequency 17 for dtmr1:

Chart, histogram

Description automatically generated

Zipf’s Law does not apply to this prject’s corpus – the most common word frequency is almost 50, the next most common is 40, which is not twice less.

* Wordclouds

Wordclouds comparison:

Text, timeline

Description automatically generatedText

Description automatically generated

A. TF, min frequency set 50 B. TF-IDF, min frequency set 50

Text

Description automatically generatedText

Description automatically generated

C. TF, max 100 words D. TF-IDF, set min frequency 1, max 200 words

A: The biggest number of little words, only one green – big word – details. Details were considered as the most important in this method. It refers to party (letsparty, partywear) twice, when other word clouds don’t refer at all or refers only once. We can also find work and business references – also generated only in this wordcloud. Leather and pieces are the biggest words after details.

B: The lowest number of words. The biggest words are denim and leather. No little words appear in it. The littlest words are pink, seasonal, detail, ted, weve, cutout, letsparty, hypertecture.

C: 4 biggest words: classic, leather, silhouette, denim, 5 medium big (pink) words, a lot of little words.

D: Words sizes are the most diversified in this wordcloud, there are 5 colours. The biggest words are denim and leather, same as in C and B wordclouds. Relatively not much little words are in this wordcloud – only 8.

* Bigram analysis

All initial operations must be repeated to prepare corpus for a bigram analysis.

Bigram DTM creation:

The document term matrix for bigram contains 8296 terms and is sparse in 97%.

Text

Description automatically generated

First 6 most occured double words:

Table

Description automatically generated with low confidence

By the same metod as previous dtmrb and dtmr1b were created, with sparsity level in dtmrb 95%, and in dtmrb1 91%.

Bigrams’ Histogram:

Chart, bar chart

Description automatically generated

Some trends appeared as a new ones, not found in one-gram analysis. These can be very specific, e.g. bomber jacket, cargo pants, jersey dress, jeweller pearl, luscious red.

Stemming impact can be clearly seen now. Match set should have probably meaned matching set, leather altern – leather alternatives, feature heel – featured heels etc.

Then data was transformed into Frame Matrix and column X was removed.

4.3. Cluster analysis

Cluster analysis was made based on made document term matrixes. Clustering was carried out with complete and ward.D methods. Ward's minimum variance method aims at finding compact, spherical clusters. The complete linkage method finds similar clusters.

Hierarchical clustering

* **Dtm** based hierarchical clustering, **complete** method

Below highlighted are every 7 cluster splits. At green, blue and orange level where a cluster group was created simultaneously with a different cluster couple, when usually the rest is divided into single clusters.

Chart, bar chart

Description automatically generated

* **Dtm** based hierarchical clustering, **ward.D** method

Diagram, schematic

Description automatically generated

* **Tf-idf** based hierarchical clustering, **complete** method

Chart, diagram, histogram

Description automatically generated

* **Tf-idf** based hierarchical clustering, **ward.D** method

Diagram

Description automatically generated

* **Tf-idf** **with decreased sparsity** based hierarchical clustering, **complete** method

Diagram

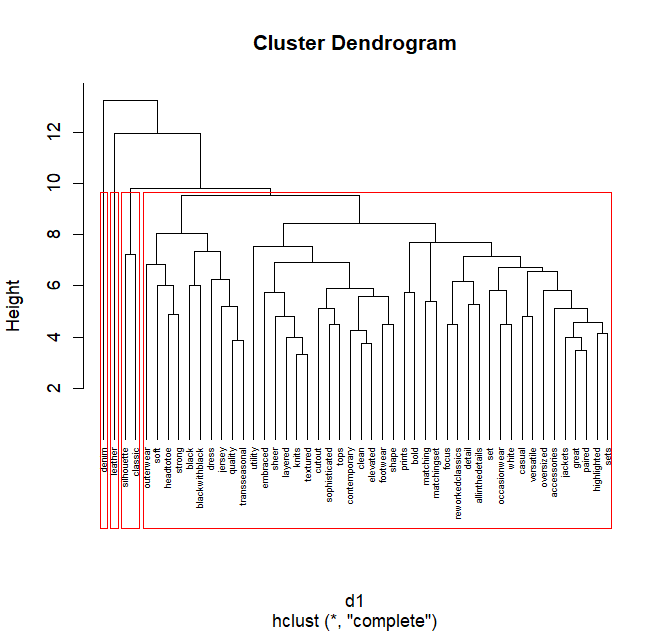
Description automatically generated with low confidence

* **Tf-idf with decreased sparsity** based hierarchical clustering, **ward.D** method

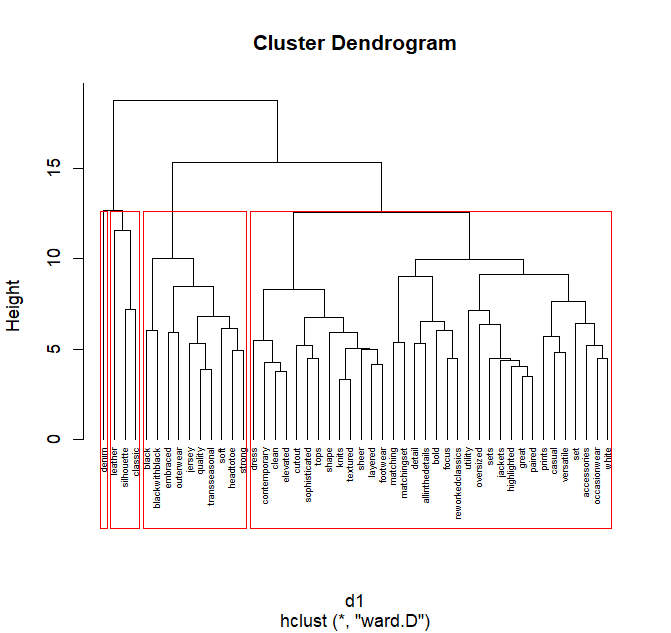
Diagram

Description automatically generated

* **Terms** hierarchical clustering based on **transposed tf-idf** matrix with **decreased sparsity**, as terms amount is much bigger than docs amount. **Complete** method



* **Terms** hierarchical clustering based on **transposed tf-idf** matrix **with decreased sparsity, ward.D** method



K-means clustering

* Dtm based

Elbow method turned out to not be very clear and helpful. It is hard to say which number could be treated as the “elbow”. Gap Statistics Method points 6 or 7 number of cluster to be optimal.

Chart, line chart

Description automatically generatedChart

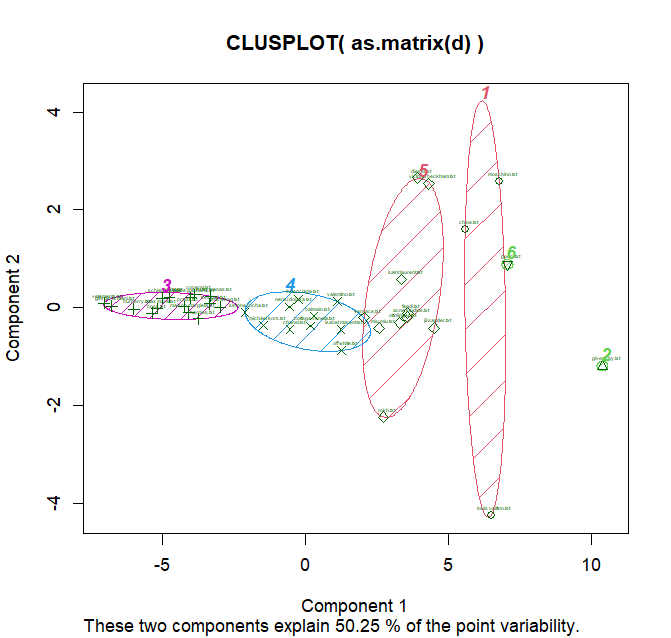
Description automatically generated

6 cluster split below:

Diagram

Description automatically generated with medium confidence

Stella McCartney is a single cluster number 1, far away from the rest, what could cause length of its document. Christian Dior was also described as single cluster number 4.   
Below Clustering performed again without these 2 documents. New optimal number was also computed as 6.



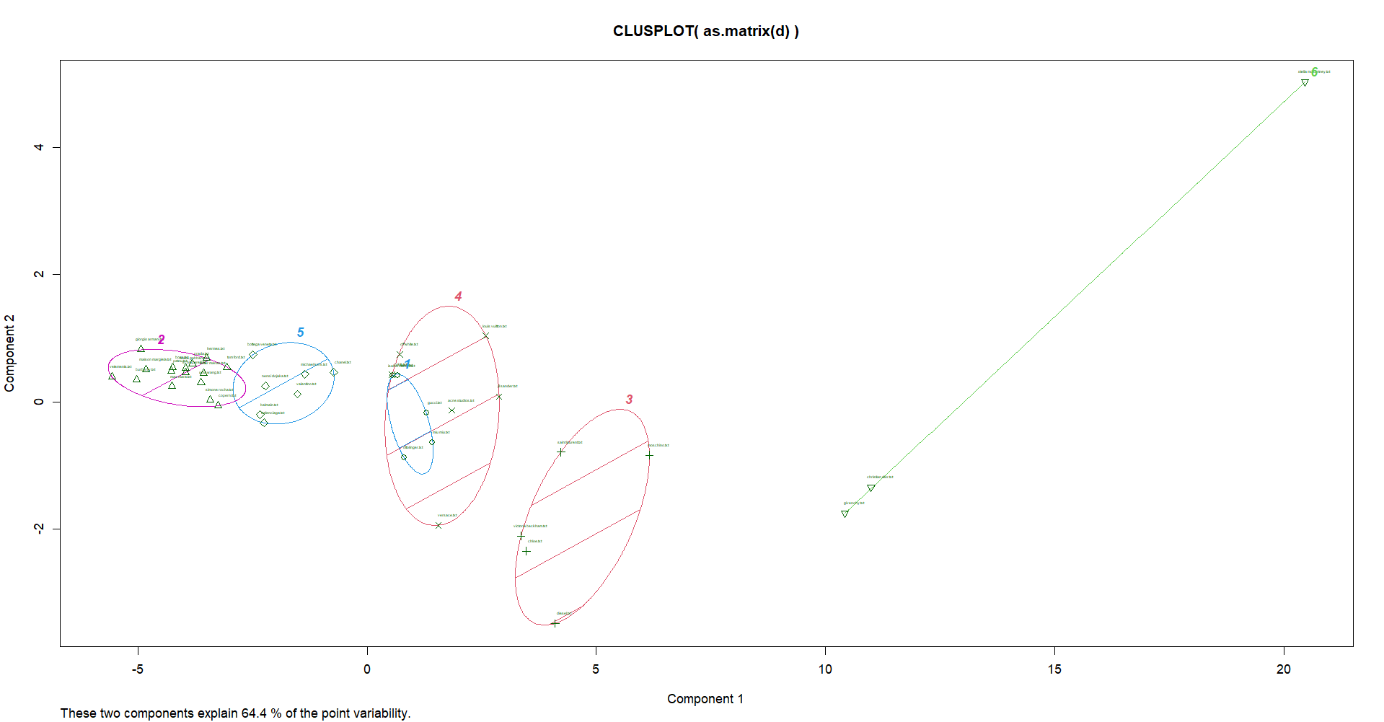
* Dtmr (settings on 0.85 – sparsity 78%) based

Elbow method again doesn’t provide any useful information. However, Gap Statistic’s Method was interpretated as pointing 6 clusters to be optimal cluster number.

Chart, histogram

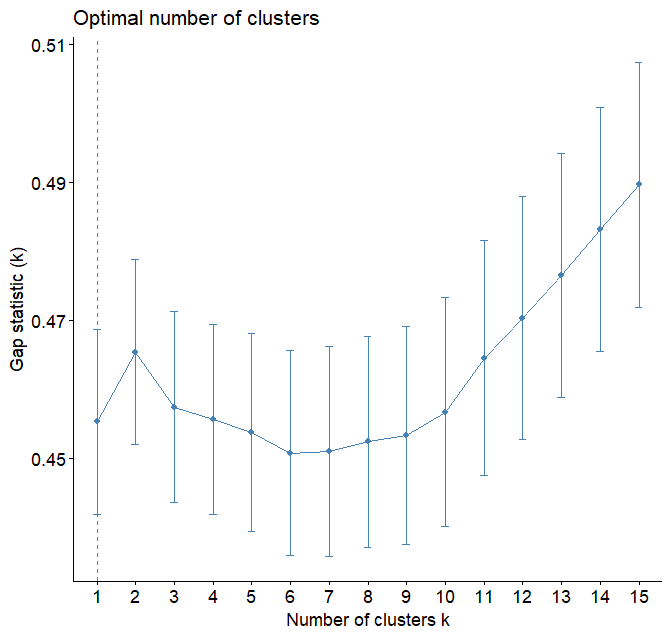
Description automatically generated

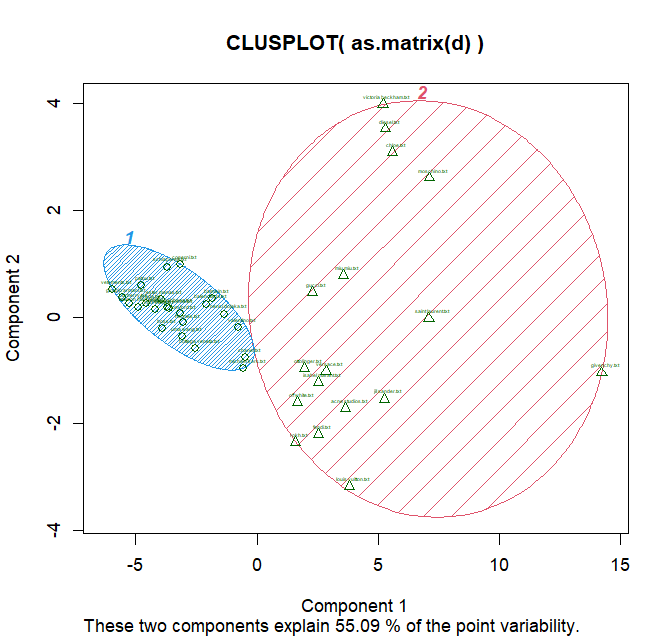
6 cluster split:



The most faraway docs are again Stella McCartney, Givenchy and Christian Dior.

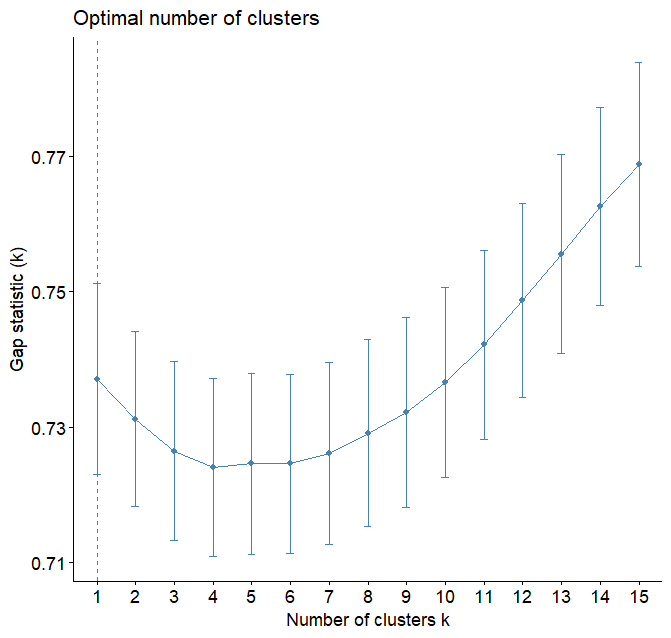
After removing them from the corpus the optimal cluster number pointed by Gap Statistic Method is clearly 2.

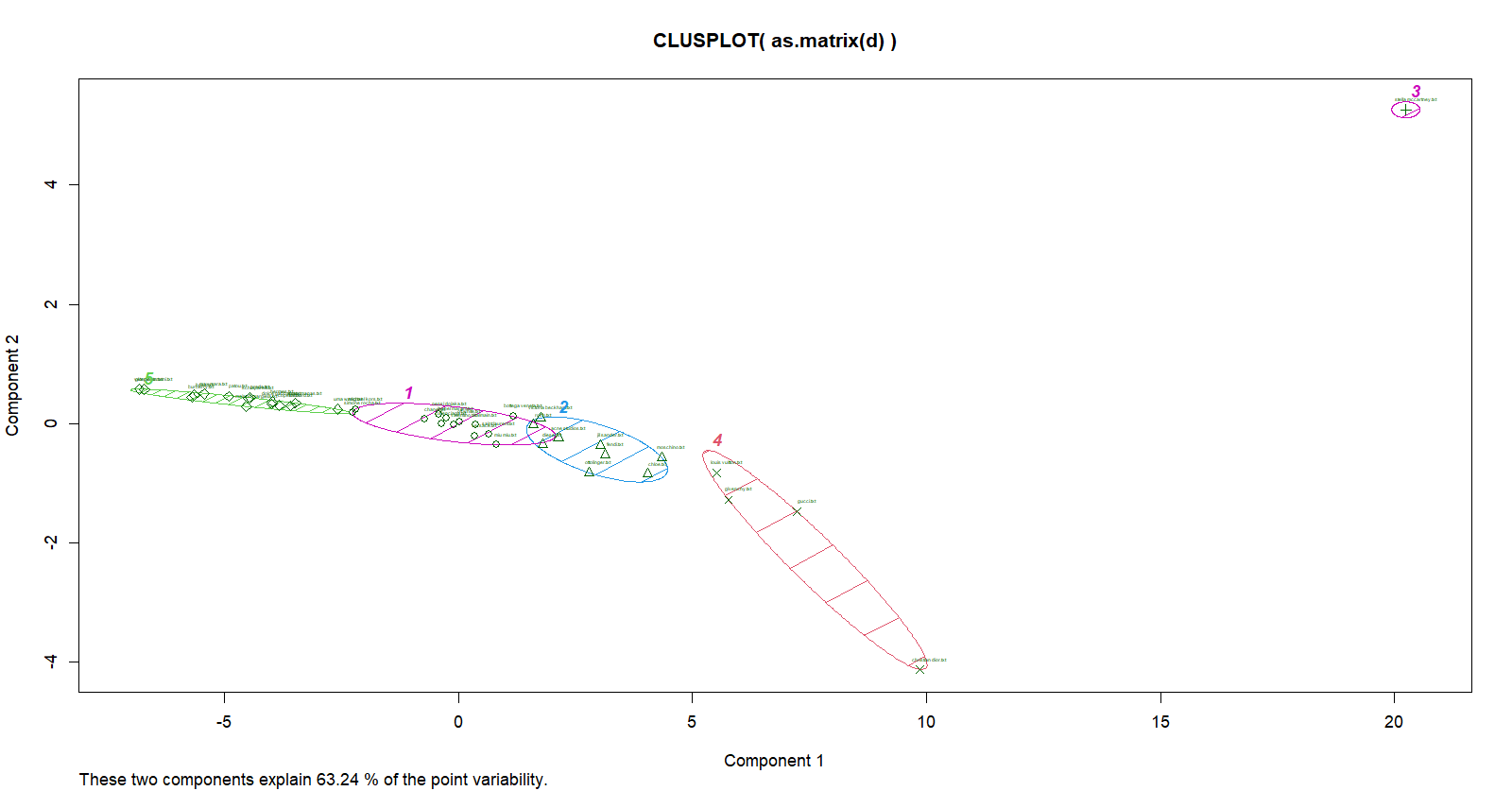




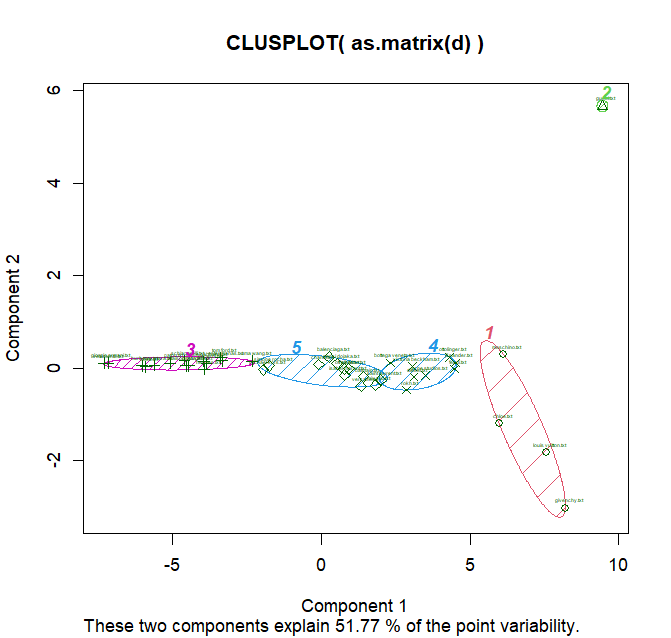
* TF-IDF based clustering

Elbow’s Method again was rejected for further analysis. Gap Statistic’s Method also didn’t provide very clear results. Regarding slight statistic increase near 5 number – docs were grouped into 5 clusters. General results are very similar to dtm results.



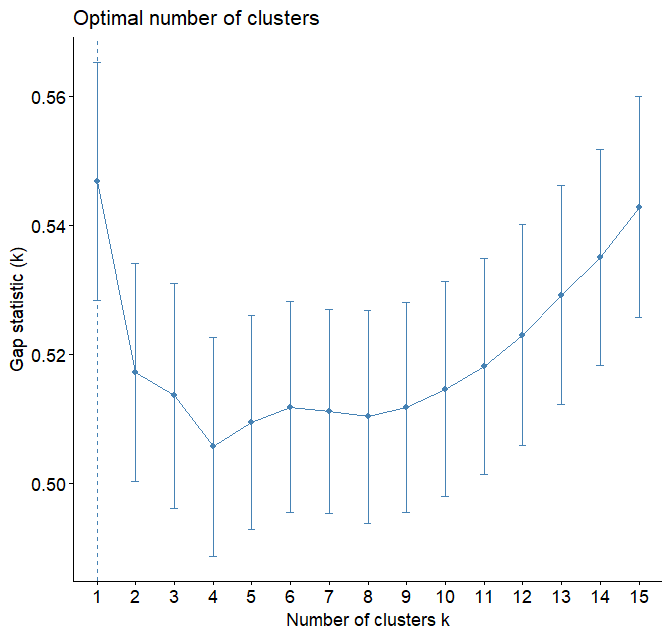


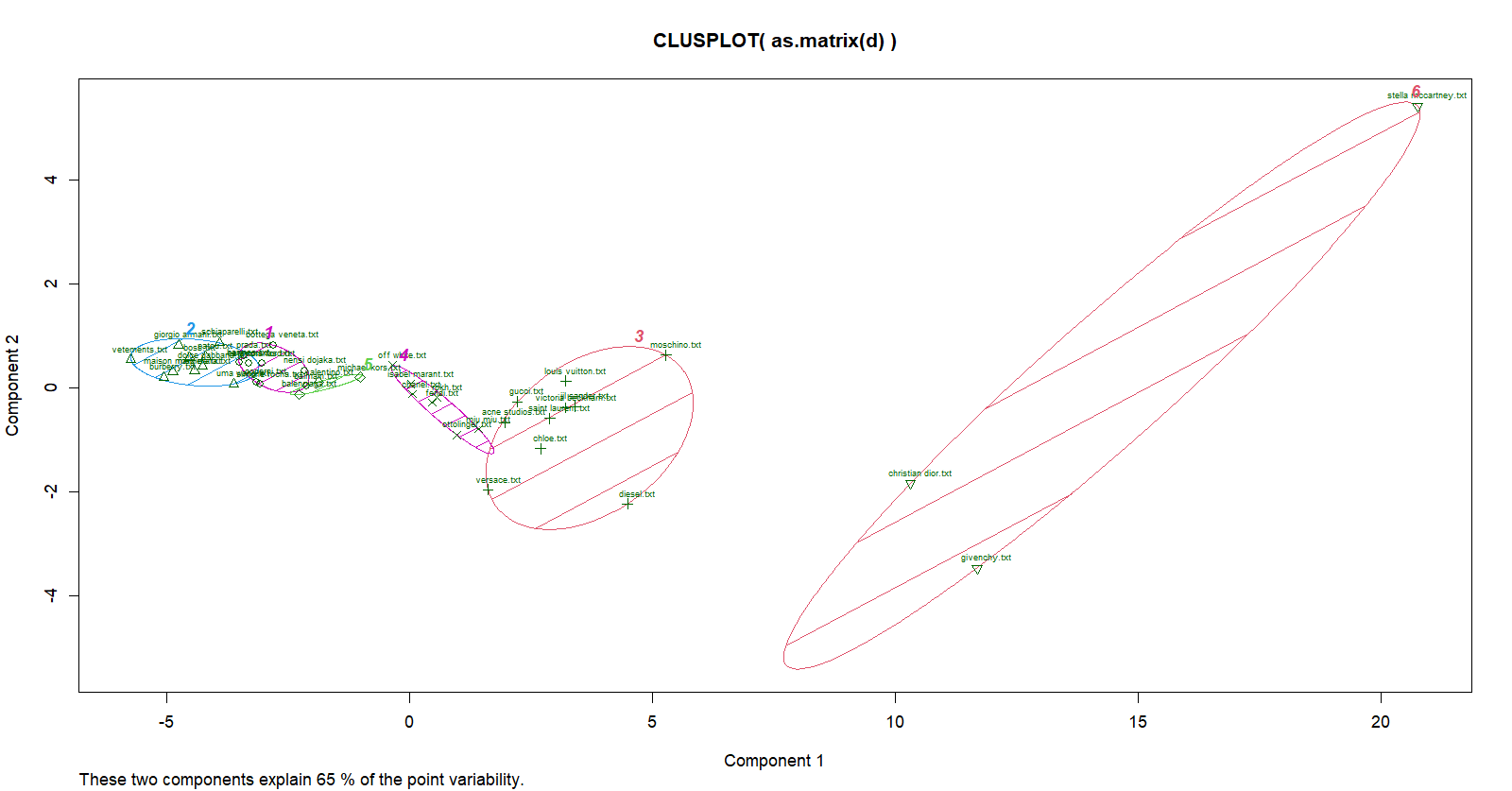
After removing Stella McCartney and Christian Dior not much has changed:



* TF-IDF with decreased sparsity clustering (set on 0.85, sparsity 78%)

Elbow Method was again rejected. Gap Statistic’s Method point again 6 clusters to be the optimal number.



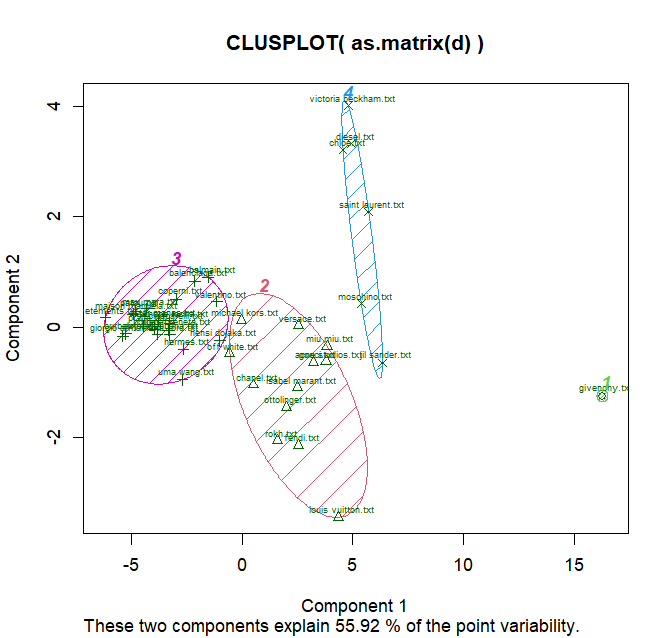


Zooming in:

Diagram

Description automatically generated

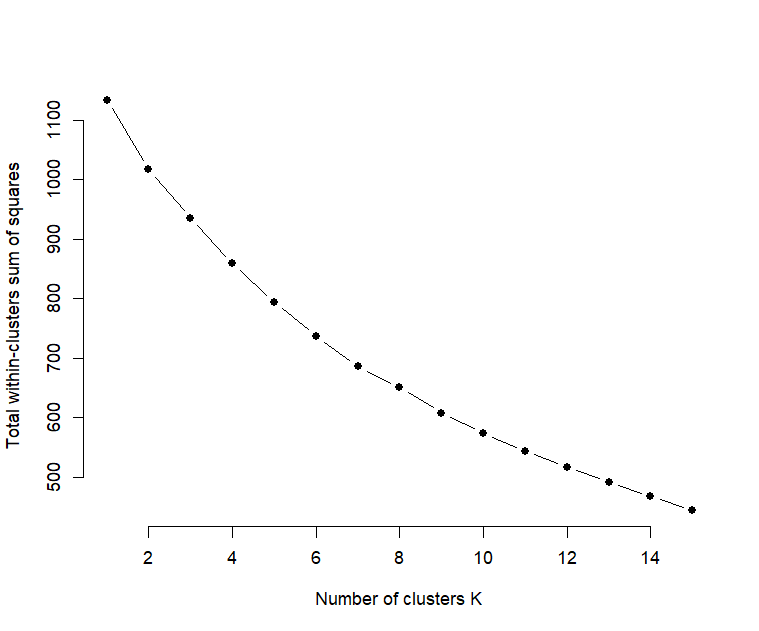
After Stella McCartey and Christian Dior Documents removing, corpus was divided into 4 clusters:



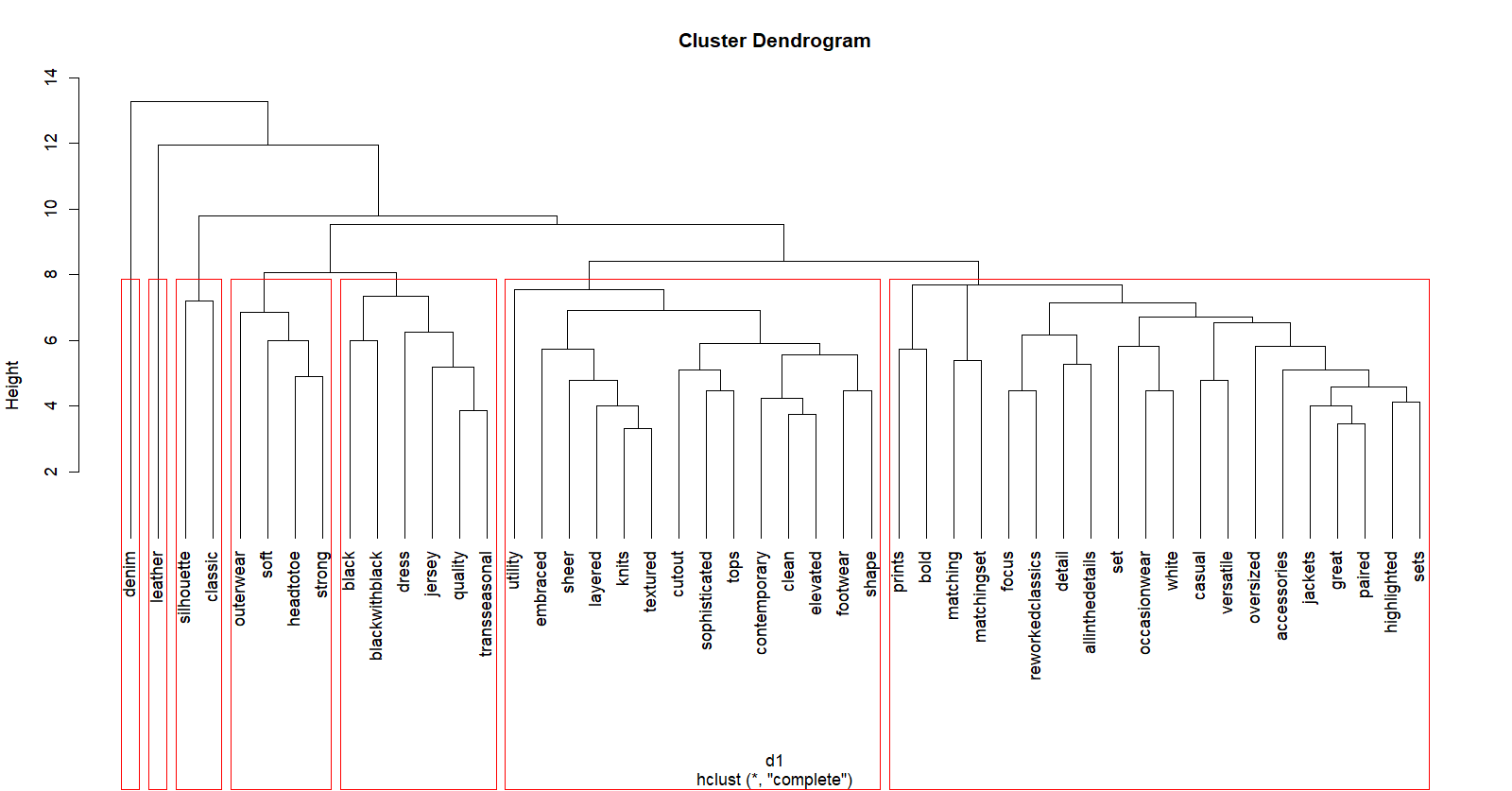
Terms clustering

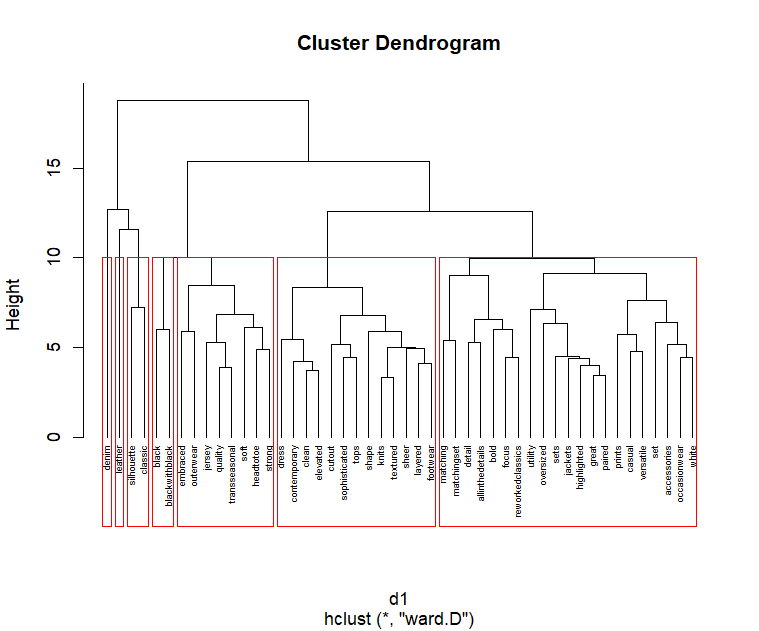
* TDM with removed sparsity based hierarchical clustering (set on 0.7, sparsity 67%, 48 terms, tdm\_s)

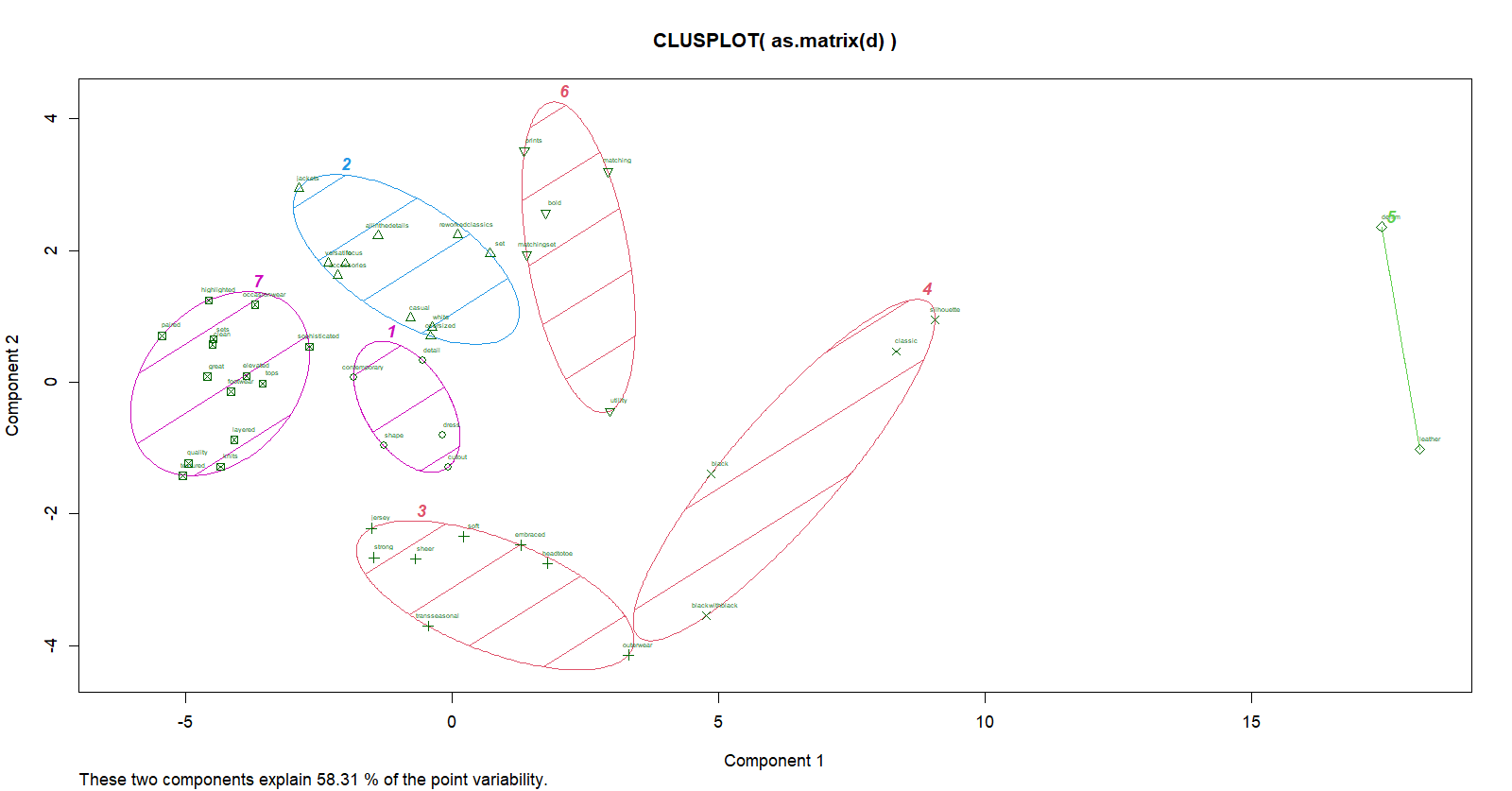
Gap Statistic method is not very clear but in comparison to Elbow Method, 7 was treated as optimal cluster number.

Chart

Description automatically generated





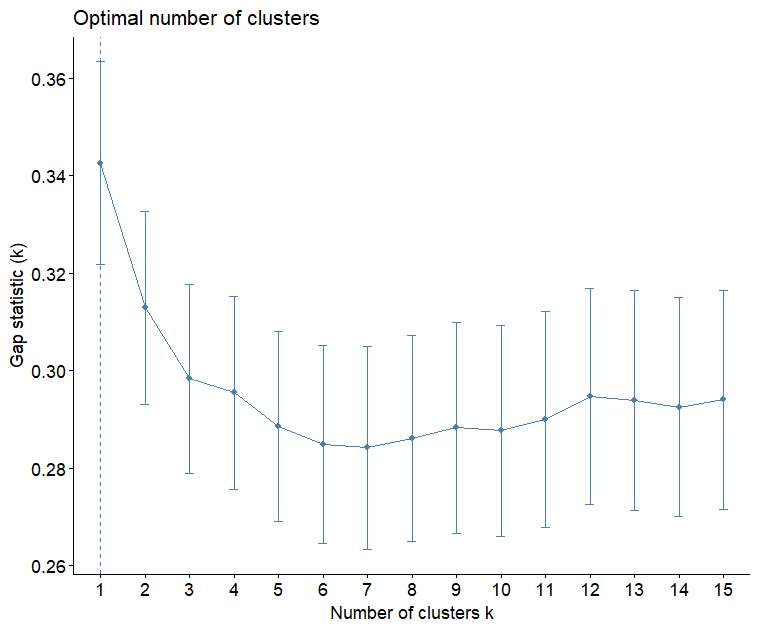


Zoom:

Diagram

Description automatically generated

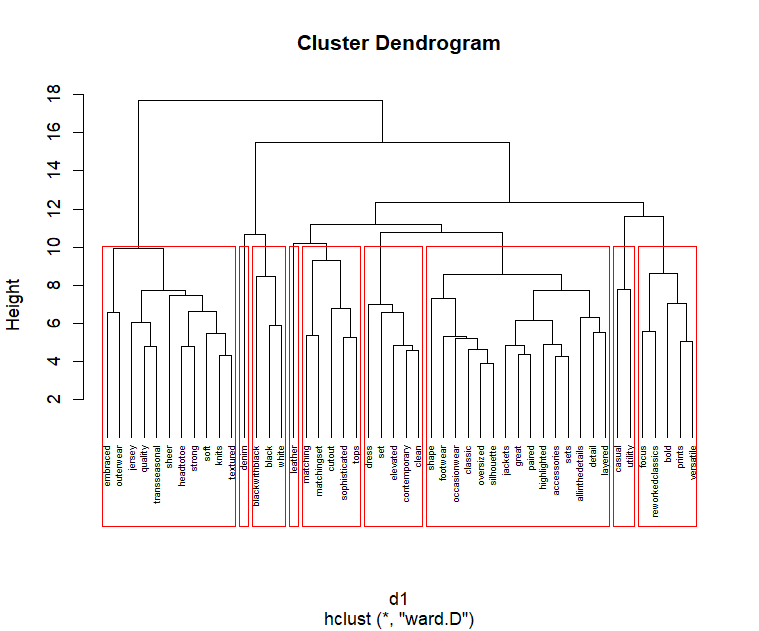
* Tf-idf with removed sparsity based hierarchical clustering (set on 0.7, sparsity 67%, 48 terms, tf\_idf\_t\_s)



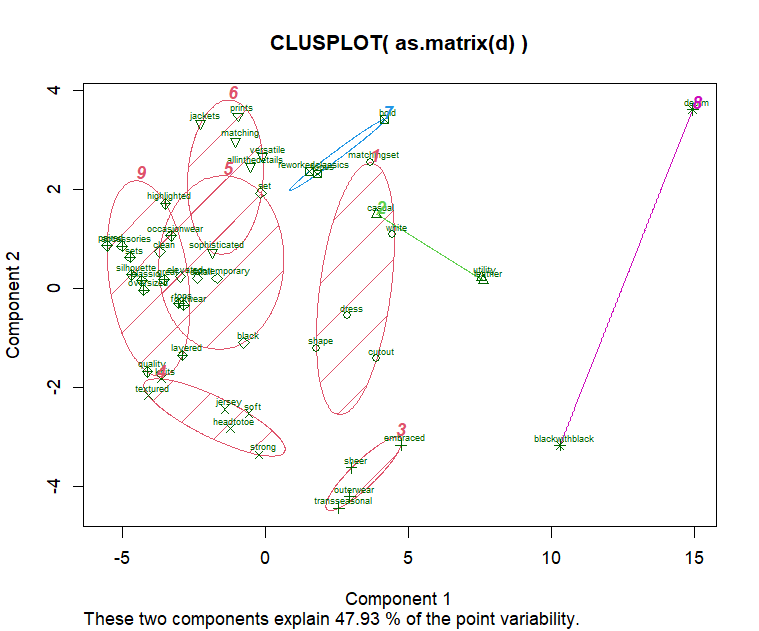
Gap Statistic was interpretated as 9 is the optimal cluster number.

Complete and ward.D methods below.

Diagram, schematic

Description automatically generated

K-means:



e. Cosine similarity

Based on DTM transformed transformed into the frame matrix and with column X removed, cosine distances matrix was built. View of fist 10 rows and columns:

Obraz zawierający tekst

Opis wygenerowany automatycznie

Creating adjacency matrix:

Chart, scatter chart

Description automatically generated

Checking the undirected weighted graph attributes:

Scatter chart

Description automatically generated

Building Graph Plots using different layouts:

* Layout in circle

Diagram, bubble chart

Description automatically generated

* Layout randomly

Diagram, schematic

Description automatically generated

* Layout on sphere

Diagram, schematic

Description automatically generated

5. Results interpretation

Comparative tables regarding terms clustering

\*set description – cluster description based on the content. What terms have in common? What their combination can mean besides their direct single meaning? Cluster content interpretation and categorization.

Exceptions: 1 or 2 terms clusters usually can not be interpretated beyond their direct meaning. A lot of words in one cluster can cause chaos and difficulty in categorization and clear interpretation, because so many words cluster is a description itself.   
Based on most cluster combinations there could be projected a graphic presentation as designs pointing specific trends.

* hierarchical clustering based on TDM:

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| source | method | cluster: | cluster number | | | | | | |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| TDM with removed sparsity (set on 0.7, 67% sparsity, 48 terms) | complete | content | denim | leather | silhouette, classic | outerwear, soft, headtotoe, strong | black, blackwithblack, dress, jersey, quality, transseasonal | utility, embraced, sheer, layered, knits, textured, cutout, sophisticated, tops, contemporary, clean, elevated, footwear, shape | prints, bold, matching, matchingset, focus, reworkedclassics, detail, allinthedetails, set, occasionwear, white, casual, versatile, oversized, accessories, jackets, great, paired, highlighted, sets |
| set description\* | denim, jeans | leather (from previous analysis it is known that leather alternatives are trending, that are supposed to be eco) | classic clothes, timeless | outerwear should be strong and tough, should cover the whole body to protect from cold etc, but it also needs to be comfortable (soft) | black colour and elegancy as timeless, good quality combination | most words concern material structure and look | elegant, detail-oriented, refreshed, permament |
| ward.D | content | denim | leather | silhouette, classic | black, blackwithblack | embraced, outwear, jersey, quality, transseasonal, soft, headtotoe, strong | dress, contemporary, clean, elevated, cutout, sophisticated, tops, shape, knits, textured, sheer, layered, footwear | matching, matchingset, detail, allinthedetails, bold, focus, reworkedclassics, utility, oversized, sets, jackets, highlighted, great, paired, prints, casual, versatile, set, accessories, occasionwear, white |
| set description\* | denim, jeans | leather (from previous analysis it is known that leather alternatives are trending, that are supposed to be eco) | classic clothes, timeless | black colour | tough, resistant | shape and figure highlight | matching sets of clothes or accessories, reuse and permanency |

Red marked are differences between complete method and ward.D.

* hierarchical clustering based on TF-IDF with removed sparsity (set on 0.7, sparsity 67%, 48 terms).

Complete method:

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| cluster: | cluster number | | | | | | | | |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| content | denim | casual, utility | cutout, shape | embraced, outwear, sheer, layered, knits, textured, quality, transseasonal, headtottoe, strong, jersey, soft | blackwithblack | leather | dress, matching, matchingset | black, white, footwear, occasionwear, classic, set, elevated, contemporary, clean | bold, prints, versatile, sophisticated, tops, jackets, great, paired, accessories, sets, highlighted, oversized, silhouette, focus, reworkedclassics, detail, allinthedetails |
| Set description\* | denim, jeans | everyday wear good for every occasion | cutouts make uncommon clothing shapes, can also highlight body shape | outerwear that is made from layers and can be used in more than 1 season, permanent and comfortable | black clothes weared with another black clothes | leather (from previous analysis it is known that leather alternatives are trending, that are supposed to be eco) | all are the whole outfit made from the same metrial | b&w elegance wear for special occasions with heighlighted shoes, clean, minimalistic look, classic refreshed with contemporaneity | as above |

Ward.D method:

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| cluster: | cluster number | | | | | | | | |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| content | embraced, outwear, jersey, quality, transseasonal, sheer, headtotoe, strong, soft, knits, textured | denim | blackwithblack, black, white | leather | matching, matchingset, cutout, sophisticated, tops | dress, set, elevated, contemporary, clean | shape, footwear, occaionwear, classic, oversized, silhouette, jackets, great, paired, highlighted, accessories, sets, allinthedetails, detail, layered | casual, utility | focus, reworkedclassics, bold, prints, versatile |
| Set description\* | tough, resistant, comfortable | denim, jeans | specific no-taint colours b&w | leather (from previous analysis it is known that leather alternatives are trending, that are supposed to be eco) | sets containing tops with cutouts | as above | as above | everyday wear good for every occasion | reworked classics focused on bold but versatile prints |

* k-means clustering based on TDM with removed sparsity, white are cluster contents, grey are set desriptions, as above.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| source | cluster number | | | | | | |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| TDM with removed sparsity (set on 0.7, 67% sparsity, 48 terms) | contemporary, detail, shape, cutout, dress | jackets, allinthedetails, versatile, focus, accessories, casual, white, oversized, set, reworkedclassics | jersey, strong, sheer, soft, embraced, transseasonal, headtotoe, outerwear | silhouette, classic, black, blackwithblack | leather, denim | prints, matching, matchingset, bold, utility | highlighted, occasionwear, sets, clean, paired, sophisticated, great, elevated, tops, footwear, layered, quality, textured, knits |
| New ideas of details and shape creation, dress with cutouts, minimalist focused on details | Connection of casual and oversize, with reworked classics, in white, focused on details such as accessories | Outerwear that can be used in more than 1 season, whole body covering | Classic styled black with silhouette highlight | as above | Printed in bold patterns outfits containing from several parts, all in the same printed material | Highlighted minimalism and elegancy of special occasion clothes |

* k-means clustering based on TF-IDF with removed sparsity (set on 0.7, sparsity 67%, 48 terms)

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| cluster number | | | | | | | | |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| matchingset, white, dress, shape, cutout | casual, utility, leather | sheer, embraced, outerwear, transseasonal | knits, textured, jersey, soft, headtotoe, strong | clean, black, contemporary, elevated, set | prints, jackets, matching, versatile, allinthedetails, sophisticated | bold, reworkedclassics, focus | denim, blackwithblack | highlighted, occasionwear, accessories, paired, sets, silhouette, classic, oversized, great, tops, footwear, layered, quality |
| shape modification by cut outs, white matching sets or dress | everyday and practical leather wear | outerwear that can be used more than 1 season | material kinds | minimalist and nowadays black clothes and sets | matching prints, jacket prints, universal wear with specific details | bold redesigned classics | black jeans | special occasion elegant, high-wuality classics as sets |

Research Questions:

1. What are the most often used products, materials, colors, prints or styles – what can be predicted as the most trending in future seasons?

Keywords, words’ connections and words’ grouping can help in recognition of future season trends. From carried out experiments we can consider future season trends by apparel product, material, construction, colour, style, destination, focused on specific feature, ideology and also their connections..

Product: bomber jacket, cargo pants, featured heels, jeweller pearls, blazer, dress, jacket, footwear, top, bag,

Material: denim, leather, jersey, knit, sheer, soft, textured, satin, fringing

Construction: cut outs, layered, embraced, head to toe, lowrise, highlighted silhouette, strong

Colour: black, black with black, white, luscious red, astro green, florals, prints, dopamine brights, black and white, pink, monochrome, bold, jewel tones, performance blue

Style: classic, oversize, reworked classic, clean, minimalistic, prints, sophisticated, contemporary, casual, modern gothic, darkness, modern academy, core, biker and motorcycle,

Destination: outwear, occasional, everyday, trans seasonal, casual, attention drawing, comfortable, business

Focused on specific feature: matching set, paired, jewellerification, detail oriented, accessories, leather alternatives.

Ideology: bold, sustainable, permanent, good quality, trans seasonal, organic

Connections: bold colours, biker and motorcycle leather, classic bag, classic florals, white denim, black and white, retro denim, matching jeans, business casual.

2. What could have caused such trend appearance? Can it be considered as recurrent or provoked by something?

Trends are dependent on various factors, from global economy to Netflix series. Gothic and dark style trending on fashion shows are clearly connected to recent Netflix series “Wednesday”, similar thing has happened when “The Bridgetons” series was released.

Usually a cycle can be observed – for example during pandemic a pastel colours were trending, and now In opposite to this dopamine bright and bold colours are trending. Additionaly a lot of old trends come back, for example lowrise jeans. Some trends are clearly seasonal, like florals.

In recent years an ecological discussions were handled regarding fashion. Now it calmed down, but instead there are trends about clothes utility, practicalness, good quality and permanency.

3. What kind of products could have been predicted to appear, which are obvious?

Regarding “Wednesday” series it was quite obvious that gothic trend will appear. It also concerns all seasonal trends or long termed that form for a long time.

4. Does corpus grouping help to recognize trends?

Yes, it guides to categorization and wider interpretation of trends. Sometimes terms’ combinations can lead to new discoveries, that could be even illustrated. Grouping helps to notice what specific terms have in common.

6. Knowledge implementation

This analysis could be implemented in fashion retail companies in order to predict future trends and meet consumer needs. Usually fashion brands targeting into mass market doesn’t implement their own new fashion ideas and solutions, they follow other brands – from “high” fashion sphere, or other retail brands. Predicting trends early can help to overtake other brands at the market. It can transfer to sales growth and revenue growth. Constant adaptation and production of the newest trend can help to build brand recognition as the leader.

A private person can also use this data. Person working as self-employed designer can also derive profits, as not only mass-market customer requires trend following. Specific customer groups also adapt tends.

An individual person can also consider this analysis in terms of hobby and trend following “for the sport”.

The analysis can also introduce into further trend analysis. Other analysis can be for example: world situation analysis to predict not showed up yet trends, street wear trends, fashion trend cycle – which old trends can come back, which not, and why, connections between surprise trends appearing.

7. References

Dr Nina Rizun materials - lecture presentations, laboratory materials

WGSN collections reviews

[Elbow method (clustering) - Wikipedia](https://en.wikipedia.org/wiki/Elbow_method_(clustering))

[Determining the number of clusters in a data set - Wikipedia](https://en.wikipedia.org/wiki/Determining_the_number_of_clusters_in_a_data_set#The_gap_statistics)

[K-Means Clustering and the Gap-Statistics | by Tim Löhr | Towards Data Science](https://towardsdatascience.com/k-means-clustering-and-the-gap-statistics-4c5d414acd29)

[The Gap Statistic | VanessaSaurus (vsoch.github.io)](https://vsoch.github.io/2013/the-gap-statistic/#:~:text=The%20gap%20statistic%20is%20a%20method%20for%20approximating,that%20error%20decreases%20steadily%20as%20our%20K%20increases%3A)

[Color Palettes in R | R-bloggers](https://www.r-bloggers.com/2010/06/color-palettes-in-r/)

[r - Adjusting the node size in igraph using a matrix - Stack Overflow](https://stackoverflow.com/questions/12058556/adjusting-the-node-size-in-igraph-using-a-matrix)

[r - Match vertex size to label size in igraph - Stack Overflow](https://stackoverflow.com/questions/14472079/match-vertex-size-to-label-size-in-igraph)

[Weighted vs. Unweighted Graphs | Baeldung on Computer Science](https://www.baeldung.com/cs/weighted-vs-unweighted-graphs)

[How to adjust vertex size on plot based on the number of connections a vertex has - Usage - igraph support forum](https://igraph.discourse.group/t/how-to-adjust-vertex-size-on-plot-based-on-the-number-of-connections-a-vertex-has/780)

[Adjacency list - Wikipedia](https://en.wikipedia.org/wiki/Adjacency_list)

[Weighted vs. Unweighted Graphs | Baeldung on Computer Science](https://www.baeldung.com/cs/weighted-vs-unweighted-graphs)

[Graph (discrete mathematics) - Wikipedia](https://en.wikipedia.org/wiki/Graph_(discrete_mathematics))

[Discover the Main Design Trends of 2022 | Design Trends | Canva - YouTube](https://www.youtube.com/watch?v=IElpMufAXj0)

[tf–idf - Wikipedia](https://en.wikipedia.org/wiki/Tf%E2%80%93idf)

[Zipf's law - Wikipedia](https://en.wikipedia.org/wiki/Zipf%27s_law)

[Dot product - Wikipedia](https://en.wikipedia.org/wiki/Dot_product)

[Voronoi diagram - Wikipedia](https://en.wikipedia.org/wiki/Voronoi_diagram)

[Stemming - Wikipedia](https://en.wikipedia.org/wiki/Stemming)

[Voronoi diagram - Wikipedia](https://en.wikipedia.org/wiki/Voronoi_diagram)

[k-means clustering - Wikipedia](https://en.wikipedia.org/wiki/K-means_clustering)

[Cluster analysis - Wikipedia](https://en.wikipedia.org/wiki/Cluster_analysis)

[Hierarchical clustering - Wikipedia](https://en.wikipedia.org/wiki/Hierarchical_clustering)

[Iloczyn skalarny – Wikipedia, wolna encyklopedia](https://pl.wikipedia.org/wiki/Iloczyn_skalarny)

[Cosine similarity - Wikipedia](https://en.wikipedia.org/wiki/Cosine_similarity)

[Measures of Distance in Data Mining - GeeksforGeeks](https://www.geeksforgeeks.org/measures-of-distance-in-data-mining/)

[Przestrzeń euklidesowa – Wikipedia, wolna encyklopedia](https://pl.wikipedia.org/wiki/Przestrze%C5%84_euklidesowa)

[TFIDF – Wikipedia, wolna encyklopedia](https://pl.wikipedia.org/wiki/TFIDF)

[Binary number - Wikipedia](https://en.wikipedia.org/wiki/Binary_number)

[Dwójkowy system liczbowy – Wikipedia, wolna encyklopedia](https://pl.wikipedia.org/wiki/Dw%C3%B3jkowy_system_liczbowy)

[Algebra Boole’a – Wikipedia, wolna encyklopedia](https://pl.wikipedia.org/wiki/Algebra_Boole%E2%80%99a)

[Zipf’s law | probability | Britannica](https://www.britannica.com/topic/Zipfs-law)

[Text mining - Wikipedia](https://en.wikipedia.org/wiki/Text_mining)

[Data mining - Wikipedia](https://en.wikipedia.org/wiki/Data_mining)

8. Code:

#install.packages("tm")

#install.packages("SnowballC")

#install.packages("ggplot2")

#install.packages("wordcloud")

getwd()

setwd("C:\\Users\\Uzytkownik\\OneDrive\\Pulpit\\politechnika\\TEXT MINING PROJEKT")

wd<-"C:\\Users\\Uzytkownik\\OneDrive\\Pulpit\\politechnika\\TEXT MINING PROJEKT\\data mining texts"

dir(wd)

library(tm)

docs <- Corpus(DirSource(wd))

docs

writeLines(as.character(docs[[1]]))

getTransformations()

docs <- tm\_map(docs,removePunctuation)

docs <- tm\_map(docs, removeNumbers)

#docs <- tm\_map(docs, PlainTextDocument)

writeLines(as.character(docs[[1]]))

for (j in seq(docs)) {

docs[[j]] <- gsub("/", "", docs[[j]])

docs[[j]] <- gsub("@", "", docs[[j]])

docs[[j]] <- gsub("–", "", docs[[j]])

docs[[j]] <- gsub("’", "", docs[[j]])

docs[[j]] <- gsub("“", "", docs[[j]])

docs[[j]] <- gsub("…", "", docs[[j]])

docs[[j]] <- gsub("‘", "", docs[[j]])

docs[[j]] <- gsub(")", "", docs[[j]])

docs[[j]] <- gsub("”", "", docs[[j]])

docs[[j]] <- gsub("?", "", docs[[j]])

docs[[j]] <- gsub("#", "", docs[[j]])

}

#docs <- tm\_map(docs, PlainTextDocument)

writeLines(as.character(docs[[1]]))

docs <- tm\_map(docs, tolower)

#docs <- tm\_map(docs, PlainTextDocument)

writeLines(as.character(docs[[1]]))

length(stopwords("english"))

stopwords("english")

docs <- tm\_map(docs, removeWords, stopwords("English"))

#docs <- tm\_map(docs, PlainTextDocument)

writeLines(as.character(docs[[1]]))

StW<-read.table("C:\\Users\\Uzytkownik\\OneDrive\\Pulpit\\politechnika\\TEXT MINING PROJEKT\\stopwords.txt")

StW

StWW<-as.character(StW$V1)

StWW

docs <- tm\_map(docs, removeWords, StWW)

#docs <- tm\_map(docs, PlainTextDocument)

writeLines(as.character(docs[[1]]))

docs <- tm\_map(docs, stripWhitespace)

#docs <- tm\_map(docs, PlainTextDocument)

writeLines(as.character(docs[[1]]))

stemDocument("modelling", language = "english")

stemDocument("modeller", language = "english")

stemDocument("models", language = "english")

writeLines(as.character(docs[[1]]))

dtm <- DocumentTermMatrix(docs)

dtm

inspect(dtm[1:42, 140:145])

tdm <- t(dtm)

tdm <- TermDocumentMatrix(docs)

tdm

inspect(tdm[170:175,37:42])

filenames <- list.files(setwd("C:\\Users\\Uzytkownik\\OneDrive\\Pulpit\\politechnika\\TEXT MINING PROJEKT\\data mining texts"),pattern="\*.txt")

filenames <-c(filenames)

rownames(dtmr)<-filenames

#ograniczenia dla macierzy

dtmr <-DocumentTermMatrix(docs, control=list(wordLengths=c(3, 20),bounds = list(global = c(2,Inf))))

dtmr1 = removeSparseTerms(dtmr, 0.85)

doc\_length <- as.data.frame(rowSums(as.matrix(dtmr1)))

doc\_length

max\_length<-max(doc\_length)

max\_length

min\_length<-min(doc\_length)

min\_length

aver\_length<-mean(rowSums(as.matrix(dtmr1)))

aver\_length

#create a normalized dtm which eliminates the difference in document length

nn<-rowSums(as.matrix(dtm))

nn

dtm\_Norm<-dtm/nn

dtmr

dtmr1

m0 <- as.matrix(dtm)

write.csv(m0, file="C:\\Users\\Uzytkownik\\OneDrive\\Pulpit\\politechnika\\TEXT MINING PROJEKT\\data mining matrix\\DocumentTermMatrix.csv")

m1<-as.data.frame(as.matrix(dtm\_Norm))

write.csv(m1, file="C:\\Users\\Uzytkownik\\OneDrive\\Pulpit\\politechnika\\TEXT MINING PROJEKT\\data mining matrix\\DocumentTermMatrixNorm.csv")

m2 <- as.matrix(dtmr)

write.csv(m2, file="C:\\Users\\Uzytkownik\\OneDrive\\Pulpit\\politechnika\\TEXT MINING PROJEKT\\data mining matrix\\DocumentTermMatrix\_1.csv")

m3 <- as.matrix(dtmr1)

write.csv(m3, file="C:\\Users\\Uzytkownik\\OneDrive\\Pulpit\\politechnika\\TEXT MINING PROJEKT\\data mining matrix\\SparseDocumentTermMatrix.csv")

freqr <- colSums(as.matrix(dtmr1))

length(freqr)

freq <- sort(freqr, decreasing=TRUE)

head(freq, 6)

tail(freq, 14)

#relationships between terms - correlations

findFreqTerms(dtmr1,lowfreq=15)

findAssocs(dtmr1,"matching",0.6)

#histogram

freqr <- colSums(as.matrix(dtmr1))

length(freqr)

freq <- sort(freqr, decreasing=TRUE)

mk<-min(head(freq, 30))

mk

wf=data.frame(word=names(freq),freq=freq)

library(ggplot2)

# Full Zipf's law

dev.new(width = 150, height = 100, unit = "px") #could be useful

p <- ggplot(subset(wf, freq>1), aes(x = reorder(word, -freq), y = freq))

p <- p + geom\_bar(stat="identity")

p <- p + theme(axis.text.x=element\_text(angle=45, hjust=1))

p

#Zipfs law with minimal frequency = MK

dev.new(width = 200, height = 200, unit = "px") #could be useful

p <- ggplot(subset(wf, freq>17), aes(x = reorder(word, -freq), y = freq))

p <- p + geom\_bar(stat="identity")

p <- p + theme(axis.text.x=element\_text(angle=45, hjust=1))

p

#wordcloud

library(wordcloud)

dev.new(width = 250, height = 250, unit = "px") #could be useful

set.seed(142)

dark2 <- brewer.pal(6, "Dark2")

wordcloud(names(freq),freq, min.freq=70)

set.seed(142)

dev.new(width = 250, height = 250, unit = "px") #could be useful

wordcloud(names(freq), freq, max.words=100)

wordcloud(names(freq), freq, min.freq=70,colors=brewer.pal(6, "Dark2"))

set.seed(142)

dev.new(width = 200, height = 200, unit = "px") #could be useful

dark2 <- brewer.pal(6, "Dark2")

wordcloud(names(freq), freq, max.words=100, rot.per=0.2, colors=dark2)

#bigram

docs\_1 <- VCorpus(DirSource(wd))

docs\_1

docs\_1<- tm\_map(docs\_1,removePunctuation)

docs\_1<- tm\_map(docs\_1, removeNumbers)

for (j in seq(docs\_1)) {

docs\_1 [[j]] <- gsub("/", "", docs\_1[[j]])

docs\_1 [[j]] <- gsub("@", "", docs\_1[[j]])

docs\_1 [[j]] <- gsub("–", "", docs\_1[[j]])

docs\_1 [[j]] <- gsub("’", "", docs\_1[[j]])

docs\_1 [[j]] <- gsub("“", " ", docs\_1[[j]])

docs\_1 [[j]] <- gsub("…", "", docs\_1[[j]])

docs\_1 [[j]] <- gsub("‘", "", docs\_1[[j]])

docs\_1 [[j]] <- gsub(")", "", docs\_1[[j]])

docs\_1 [[j]] <- gsub("”", "", docs\_1[[j]])

docs\_1 [[j]] <- gsub("?", "", docs\_1[[j]])

docs\_1 [[j]] <- gsub("#", "", docs\_1[[j]])

}

docs\_1<- tm\_map(docs\_1, tolower)

docs\_1<- tm\_map(docs\_1, removeWords, stopwords("English"))

StW<-read.table("C:\\Users\\Uzytkownik\\OneDrive\\Pulpit\\politechnika\\TEXT MINING PROJEKT\\stopWordsbigram.txt")

StWW<-as.character(StW$V1)

StWW

docs\_1<- tm\_map(docs\_1, removeWords, StWW)

docs\_1<- tm\_map(docs\_1, stripWhitespace)

for (j in seq(docs\_1)) {

docs\_1[[j]]<-stemDocument(docs\_1[[j]], language = "english")}

docs\_1<- tm\_map(docs\_1, PlainTextDocument)

#building bigrams based DTM MAtrix

NgramTokenizer = function(x) {

unlist(lapply(ngrams(words(x), 2), paste, collapse = " "),

use.names = FALSE)

}

dtm\_n <- DocumentTermMatrix(docs\_1, control = list(tokenize = NgramTokenizer))

dtm\_n

filenames <- list.files(getwd(),pattern="\*.txt")

filenames <-c(filenames)

rownames(dtm\_n)<-filenames

#create a normalized dtm which eliminates the difference in document length

dtmrb <-DocumentTermMatrix(docs\_1, control=list(tokenize = NgramTokenizer, wordLengths=c(3, 20),bounds = list(global = c(2,Inf))))

dtmr1b = removeSparseTerms(dtmrb, 0.95)

nnb<-rowSums(as.matrix(dtm\_n))

nnb

dtm\_Normb<-dtm\_n/nnb

dtmrb

dtmr1b

m0b <- as.matrix(dtm\_n)

write.csv(m0b, file="C:\\Users\\Uzytkownik\\OneDrive\\Pulpit\\politechnika\\TEXT MINING PROJEKT\\data mining matrix\\DocumentTermMatrixBI.csv")

m1b <-as.data.frame(as.matrix(dtm\_Normb))

write.csv(m1b, file="C:\\Users\\Uzytkownik\\OneDrive\\Pulpit\\politechnika\\TEXT MINING PROJEKT\\data mining matrix\\DocumentTermMatrixNormBI.csv")

m2b <- as.matrix(dtmrb)

write.csv(m2b, file="C:\\Users\\Uzytkownik\\OneDrive\\Pulpit\\politechnika\\TEXT MINING PROJEKT\\data mining matrix\\DocumentTermMatrix\_1BI.csv")

m3b <- as.matrix(dtmr1b)

write.csv(m3b, file="C:\\Users\\Uzytkownik\\OneDrive\\Pulpit\\politechnika\\TEXT MINING PROJEKT\\data mining matrix\\SparseDocumentTermMatrixBI.csv")

#histogram bigram

freq\_n <- sort(colSums(as.matrix(dtmr1b)), decreasing=TRUE)

head(freq\_n, 6)

mk<-min(head(freq\_n, 10))

tail(freq\_n, 10)

m<-as.matrix(dtmr1b)

write.csv(m, file="C:\\Users\\Uzytkownik\\OneDrive\\Pulpit\\politechnika\\TEXT MINING PROJEKT\\data mining matrix bigram\\\_DocumentTermMatrixBI.csv")

#\_\_\_\_\_\_\_\_\_\_\_Building the Histogtram (zipf’s law)\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

wf=data.frame(word=names(freq\_n),freq=freq\_n)

wf

p <- ggplot(subset(wf, freq>=mk), aes(x = reorder(word, -freq), y = freq))

p <- p + geom\_bar(stat="identity")+ ggtitle("Histogram of Bigrams for Opinions") +labs(x="Bi

-grams",y="Frequency")

p <- p + theme(axis.text.x=element\_text(angle=90, hjust=1, size=16))

p

set.seed(142)

#dev.new(width = 200, height = 200, unit = "px") #could be useful

dark2 <- brewer.pal(6, "Dark2")

wordcloud(names(freq), freq, max.words=100, rot.per=0.2, colors=dark2)

#install.packages ("cluster")

#install.packages ("fpc")

library(tm)

library(SnowballC)

library(ggplot2)

library(wordcloud)

library(cluster)

library(fpc)

setwd("C:\\Users\\Uzytkownik\\OneDrive\\Pulpit\\politechnika\\TEXT MINING PROJEKT")

MyData <-read.csv("C:\\Users\\Uzytkownik\\OneDrive\\Pulpit\\politechnika\\TEXT MINING PROJEKT\\data mining matrix\\DocumentTermMatrix.csv",

header = TRUE, #are there column names in 1st row?

sep = ",", #what separates rows?

strip.white = TRUE, #strip out extra white space in strings.

fill = TRUE, #fill in rows that have unequal numbers of columns

comment.char = "#", #character used for comments that should not be read in

stringsAsFactors = FALSE #Another control for deciding whether characters should be converted to factor

)

dtm1 = as.data.frame.matrix(MyData)

dtm1 [1:42,1:42]

dtm<-dtm1[,-1]

filenames <- list.files(setwd("C:\\Users\\Uzytkownik\\OneDrive\\Pulpit\\politechnika\\TEXT MINING PROJEKT\\data mining texts"),pattern="\*.txt")

filenames <-c(filenames)

filenames

rownames(dtm)

rownames(dtm) <-filenames

dtm [1:22,1:20]

freq <- sort(colSums(dtm), decreasing=TRUE)

freq

freq1 <- sort(rowSums(dtm), decreasing=TRUE)

freq1

wf=data.frame(word=names(freq),freq=freq)

wf

p <- ggplot(subset(wf, freq>17), aes(x = reorder(word, -freq), y = freq))

p <- p + geom\_bar(stat="identity")

p <- p + theme(axis.text.x=element\_text(angle=45, hjust=1))

p

tdm<- t(dtm) # t(dtm) – transpose matrix DTM into TDM

tf <- as.matrix(tdm)

idf <- log(ncol(tf) / (rowSums(tf != 0)))

tf[170:175,1:5]

idf[280:288]

idf\_sort <- sort(idf, decreasing=FALSE)

head(idf\_sort, 15)

tail(idf\_sort, 15)

# building tf-idf

idf1 <- diag(idf)

tf\_idf <- crossprod(tf, idf1)

colnames(tf\_idf) <- rownames(tf)

write.csv(as.matrix(tf\_idf),file="C:\\Users\\Uzytkownik\\OneDrive\\Pulpit\\politechnika\\TEXT MINING PROJEKT\\TFIDF.csv")

tf\_idf\_t<-t(tf\_idf) #transposed matrix tf\_idf

tf\_idf\_t [355:365,1:3]

freq <- colSums(as.matrix(tf\_idf), na.rm = FALSE)

dev.new(width = 100, height = 100, unit = "px") # if you need

set.seed(42)

wordcloud(names(freq),freq, max.words=50)

# Second view of wordcloud

dev.new(width = 130, height = 130, unit = "px") # if you need

set.seed(142)

wordcloud(words = names(freq), freq = freq, min.freq = 1,

max.words=200, random.order=FALSE, rot.per=0.35,

colors=brewer.pal(8, "Dark2"))

dtm[11:12,11:13]

# if you do not have the Filenames, do the following:

filenames <- list.files(setwd("C:\\Users\\Uzytkownik\\OneDrive\\Pulpit\\politechnika\\TEXT MINING PROJEKT\\data mining texts"),pattern="\*.txt")

filenames <-c(filenames)

filenames

rownames(dtm)

rownames(dtm)<-filenames

d1 <- dist(dtm, method="euclidian")

# make the clustering

fit <- hclust(d=d1, method="complete")min

fit

plot.new()

plot(fit, hang=-1, cex=0.5)

# for a receiving the different dendrograms view ry substituting: method="ward.D" and any other form list above:

groups <- cutree(fit, k=7) # "k" defines the number of clusters you are using

rect.hclust(fit, k=7, border="red") # draw dendogram with red borders around the 4 clusters

###clustering tf-idf

filenames <- list.files(setwd("C:\\Users\\Uzytkownik\\OneDrive\\Pulpit\\politechnika\\TEXT MINING PROJEKT\\data mining texts"),pattern="\*.txt")

filenames <-c(filenames)

filenames

rownames(tf\_idf)

rownames(tf\_idf)<-filenames

tf\_idf<-as.DocumentTermMatrix(tf\_idf,weighting = weightTf)

tf\_idf

d1 <- dist(tf\_idf, method="euclidian")

fit1 <- hclust(d=d1, method="ward.D")

fit1

plot.new()

plot(fit1, hang=-1, cex=0.5)

groups <- cutree(fit1, k=14)

rect.hclust(fit1, k=14, border="red")

# remove the sparsity of the matrix tf\_idf clustering

filenames <- list.files(setwd("C:\\Users\\Uzytkownik\\OneDrive\\Pulpit\\politechnika\\TEXT MINING PROJEKT\\data mining texts"),pattern="\*.txt")

filenames <-c(filenames)

filenames

rownames(tf\_idf\_s)

rownames(tf\_idf\_s)<-filenames

tf\_idf\_s<-removeSparseTerms(tf\_idf, 0.7)

tf\_idf\_s

tf\_idf\_t\_s<-t(tf\_idf\_s) #transposed matrix tf\_idf

tf\_idf\_t\_s

d1 <- dist(tdm\_s, method="euclidian")

fit1 <- hclust(d=d1, method="ward.D")

fit1

plot.new()

plot(fit1, hang=-1, cex=0.5)

groups <- cutree(fit1, k=7)

rect.hclust(fit1, k=7, border="red")

dplyr::mutate(tibble)

help(rownames)

# Perform the TERMS clustering based on Perform the TERMS clustering based on TDM, tfidf, tfidfs

filenames <- list.files(setwd("C:\\Users\\Uzytkownik\\OneDrive\\Pulpit\\politechnika\\TEXT MINING PROJEKT\\data mining texts"),pattern="\*.txt")

filenames <-c(filenames)

filenames

rownames(tdm)

rownames(tdm)<-filenames

tdm <-as.TermDocumentMatrix(tdm,weighting = weightTf)

tdm

tdm\_s <-removeSparseTerms(tdm, 0.75)

tdm\_s

###

tdm\_t= t(tf\_idf)

tdm\_s\_t= t(tf\_idf\_s)

d1 <- dist(tf\_idf\_t, method="euclidian")

fit1 <- hclust(d=d1, method="complete")

fit1

plot.new()

plot(fit1, hang=-1, cex=1)

groups <- cutree(fit1, k=7)

rect.hclust(fit1, k=7, border="red")

dim(tdm)

#clusplot

#K\_MEANS

library(fpc)

#transform the format of dtm for possibility to do the RemoveSparseTerms

dtm <-as.DocumentTermMatrix(dtm,weighting = weightTf)

dtmr<-removeSparseTerms(dtm, 0.85)

dtmr

d <- dist(dtmr, method="euclidian")

kfit <- kmeans(d, 6)

kfit

clusplot(as.matrix(d), kfit$cluster, color=T, shade=T, labels=2, lines=0, cex.txt = 0.4)

#Perform the DOCUMENTS clustering based on DTM matrix

dtm <-as.DocumentTermMatrix(dtm,weighting = weightTf)

dtm

d <- dist(dtm, method="euclidian")

kfit <- kmeans(d, 8)

kfit

clusplot(as.matrix(d), kfit$cluster, color=T, shade=T, labels=2, lines=0, cex.txt = 0.4)

#Perform the DOCUMENTS clustering based on TF-IDF Matrix

filenames <- list.files(setwd("C:\\Users\\Uzytkownik\\OneDrive\\Pulpit\\politechnika\\TEXT MINING PROJEKT\\data mining texts"),pattern="\*.txt")

filenames <-c(filenames)

filenames

rownames(tf\_idf)

rownames(tf\_idf)<-filenames

tf\_idf <-as.DocumentTermMatrix (tf\_idf,weighting = weightTf)

tf\_idf

d <- dist(tf\_idf, method="euclidian")

kfit <- kmeans(d, 7)

kfit

clusplot(as.matrix(d), kfit$cluster, color=T, shade=T, labels=2, lines=0, cex.txt = 0.4)

#Perform the DOCUMENTS clustering based on TF-IDF Matrix with decreased sparsity

tf\_idf\_s= removeSparseTerms(tf\_idf, 0.75)

tf\_idf\_s <-as.DocumentTermMatrix (tf\_idf\_s,weighting = weightTf)

tf\_idf\_s

d <- dist(tf\_idf\_s, method="euclidian")

kfit <- kmeans(d, 4)

kfit

clusplot(as.matrix(d), kfit$cluster, color=T, shade=T, labels=2, lines=0, cex.txt = 0.5)

#term clustering with k-means based on tdm\_s

filenames <- list.files(setwd("C:\\Users\\Uzytkownik\\OneDrive\\Pulpit\\politechnika\\TEXT MINING PROJEKT\\data mining texts"),pattern="\*.txt")

filenames <-c(filenames)

filenames

rownames(tdm)

rownames(tdm)<-filenames

tdm <-as.TermDocumentMatrix(tdm,weighting = weightTf)

tdm\_s= removeSparseTerms(tdm, 0.70)

dm\_t= t(tf\_idf)

tdm\_s\_t= t(tf\_idf\_s)

tdm\_s\_t

inspect(tdm\_s\_t[1:5,1:5])

d <- dist(tf\_idf\_t\_s , method="euclidian")

kfit <- kmeans(d, 9)

kfit

clusplot(as.matrix(d), kfit$cluster, color=T, shade=T, labels=2, lines=0, cex.txt = 0.5)

#based on tf-idf

#\_\_\_\_\_\_\_\_\_\_Elbow Method\_\_\_\_\_\_\_\_check the optimal number of clusters\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

library(tidyverse) # data manipulation

library(cluster)# clustering algorithms

#install.packages("factoextra")

library(factoextra) # clustering algorithms & visualization

library(purrr)

set.seed(123)

# function to compute total within-cluster sum of square

wss <- function(k) {

kmeans(tf\_idf\_t\_s, k, nstart = 10 )$tot.withinss

}

# Compute and plot wss for k = 1 to k = 15

k.values <- 1:15

# extract wss for 2-15 clusters

wss\_values <- map\_dbl(k.values, wss)

wss\_values

# optimal number of clusters should appear to be the bend in the knee (or elbow)

plot(k.values, wss\_values,

type="b", pch = 19, frame = FALSE,

xlab="Number of clusters K",

ylab="Total within-clusters sum of squares")

# \_\_\_\_\_\_\_\_\_Gap Statistic Method\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

set.seed(123)

gap\_stat <- clusGap(tf\_idf\_t\_s, FUN = kmeans, nstart = 25,

K.max = 15, B = 50)

# Print the result

print(gap\_stat, method = "firstmax")

fviz\_gap\_stat(gap\_stat)

####Perform the clustering####

set.seed(123)

# for reproducibility

km.res <- kmeans(tf\_idf\_s, 4, nstart = 25)

km.res

# Visualize

fviz\_cluster(km.res, data = dtm, palette = "jco", ggtheme = theme\_minimal(), labelsize=5)

#install.packages("igraph")

#install.packages("topicmodels")

library("topicmodels")

library("igraph")

setwd("C:\\Users\\Uzytkownik\\OneDrive\\Pulpit\\politechnika\\TEXT MINING PROJEKT")

MyData <-read.csv("C:\\Users\\Uzytkownik\\OneDrive\\Pulpit\\politechnika\\TEXT MINING PROJEKT\\data mining matrix\\DocumentTermMatrix.csv",

header = TRUE, #are there column names in 1st row?

sep = ",", #what separates rows?

strip.white = TRUE, #strip out extra white space in strings.

fill = TRUE, #fill in rows that have unequal numbers of columns

comment.char = "#", #character used for comments that should not be read in

stringsAsFactors = FALSE #Another control for deciding whether characters should be converted to factor

)

dtm1 = as.data.frame.matrix(MyData)

dtm1 [1:45,1:45]

dtm<-dtm1[,-1]

filenames <- list.files(setwd("C:\\Users\\Uzytkownik\\OneDrive\\Pulpit\\politechnika\\TEXT MINING PROJEKT\\data mining texts"),pattern="\*.txt")

filenames <-c(filenames)

filenames

rownames(dtm)

rownames(dtm) <-filenames

dtm [1:22,1:20]

mm\_s = as.matrix(dtm)

#mm<-as.matrix(mm\_s[1:10,])

mm<-as.matrix(mm\_s) # for using all documents

#function cosineSim compute cosine similarity between document vectors

#converting to distance matrix sets diagonal elements to 0

cosineSim <- function(x){

as.dist(x%\*%t(x)/(sqrt(rowSums(x^2) %\*% t(rowSums(x^2)))))

}

#compute cosine similarity between document vectors

cs <- cosineSim(mm)

cs

#filenames2

dtm1 = as.data.frame.matrix(MyData)

dtm1 [1:45,1:45]

dtm<-dtm1[,-1]

filenames <- list.files(setwd("C:\\Users\\Uzytkownik\\OneDrive\\Pulpit\\politechnika\\TEXT MINING PROJEKT\\data mining texts"),pattern="\*.txt")

filenames <-c(filenames)

filenames

rownames(dtm)

rownames(dtm) <-filenames

dtm [1:22,1:20]

write.csv(as.matrix(cs),file="C:\\Users\\Uzytkownik\\OneDrive\\Pulpit\\politechnika\\TEXT MINING PROJEKT\\data mining matrix\\DocumentCosine.csv")

#create the adjacency matrix

min\_cos<-0.2

cs[cs < min\_cos] <- 0

cs <- round(cs,3)

#save adjacency matrix to \*.csv file

write.csv(as.matrix(cs),file="C:\\Users\\Uzytkownik\\OneDrive\\Pulpit\\politechnika\\TEXT MINING PROJEKT\\data mining matrix\\DocumentAdjacencyMatrix.csv")

cs

#adjacency matrix

dat<-read.csv("C:\\Users\\Uzytkownik\\OneDrive\\Pulpit\\politechnika\\TEXT MINING PROJEKT\\data mining matrix\\DocumentAdjacencyMatrix.csv",

header = TRUE,

sep = ",",

colClasses = NA,

na.string = "NA",

skip = 0,

strip.white = TRUE,

fill = TRUE,

comment.char = "#",

stringsAsFactors = FALSE

)

dat

mm1 = as.data.frame.matrix(dat)

mm1=mm1[,-1]

mm1

filenames <- list.files(setwd("C:\\Users\\Uzytkownik\\OneDrive\\Pulpit\\politechnika\\TEXT MINING PROJEKT\\data mining texts"),pattern="\*.txt")

#filenames <-c(filenames[1:5])

filenames <-c(filenames) # for using all documents

filenames

#converting mm1 into matrix format

rownames(mm1)<-filenames

cs<-as.matrix(mm1)

cs

#initializing Igraph package

library(igraph)

#Creating undirected weighted graph

g=graph.adjacency(cs,mode="undirected",weighted=TRUE)

g

#Checking the undirected weighted graph attributes

list.vertex.attributes(g)

list.edge.attributes(g)

V(g)$name

E(g)$weight

#Calculate the degree of each Vertices and assign it’s to the Vertices size feature

deg <- graph.strength(g, mode="all")

deg

V(g)$size <- deg\*30

#Build the Vertices color

hc5 <- terrain.colors(5) # colors – https://www.r-bloggers.com/color-palettes-in-r/

g.max <- max(deg)

vcolors <- 5 - round(4 \*(deg / g.max))

vcolors <- hc5[vcolors]

vcolors

# Build the Graph Plot using different Layouts

lay\_1 <- layout.fruchterman.reingold(g)

plot.igraph(g,layout= lay\_1, edge.arrow.size=0.1,edge.width=E(g)$weight\*30, vertex.label=V(g)$

name,vertex.color=vcolors,vertex.size=V(g)$size, vertex.label.cex=1)

lay\_2 <- layout\_in\_circle

plot.igraph(g,layout= lay\_2, edge.arrow.size=0.1,edge.width=E(g)$weight\*30, vertex.label=V(g)$

name,vertex.color=vcolors,vertex.size=V(g)$size,vertex.label.cex=1)

lay\_3 <- layout\_with\_kk(g)

plot.igraph(g,layout= lay\_3, edge.arrow.size=0.1,edge.width=E(g)$weight\*30, vertex.label=V(g)$

name,vertex.color=vcolors,vertex.size=V(g)$size,vertex.label.cex=1)

lay\_4 <- layout\_randomly(g)

plot.igraph(g,layout= lay\_4, edge.arrow.size=0.1,edge.width=E(g)$weight\*30, vertex.label=V(g)$

name,vertex.color=vcolors,vertex.size=V(g)$size,vertex.label.cex=1)

lay\_5 <- layout\_on\_sphere(g)

plot.igraph(g,layout= lay\_5, edge.arrow.size=0.1,edge.width=E(g)$weight\*30, vertex.label=V(g)$

name,vertex.color=vcolors,vertex.size=V(g)$size,vertex.label.cex=1)

# http://www.kateto.net/wp-content/uploads/2016/01/NetSciX\_2016\_Workshop.pdf

# Algorithm 1: edge betweenness (Newman-Girvan)

ceb <- cluster\_edge\_betweenness(g)

plot(ceb, g)

membership(ceb)

# Algorithm 2: based on propagating labels

clp <- cluster\_label\_prop(g)

plot(clp , g)

membership(clp)

# Algorithm 3: based on greedy optimization of modularity

cfg <- cluster\_fast\_greedy(as.undirected(g))

plot(cfg, as.undirected(g))

membership(cfg)

#terms

filenames <- list.files(setwd("C:\\Users\\Uzytkownik\\OneDrive\\Pulpit\\politechnika\\TEXT MINING PROJEKT\\data mining texts"),pattern="\*.txt")

filenames <-c(filenames)

filenames

rownames(tdm)

rownames(tdm)<-filenames

tdm <-as.TermDocumentMatrix(tdm,weighting = weightTf)

tdm

tdm\_s <-removeSparseTerms(tdm, 0.90)

tdm\_s

tdm\_t= t(tf\_idf)

tdm\_s\_t= t(tf\_idf\_s)

# building Term Document Matrix as a transformed Document Term Matrix

tdm <-as.TermDocumentMatrix(t(dtm),weighting = weightTf)

tdm

# transform Term Document Matrix into the matrix with sparsity is not higher then (for example) 0.2

tdm = removeSparseTerms(tdm, 0.70)

tdm

# completing the matrix column names

filenames <- list.files(getwd(),pattern="\*.txt")

filenames <-c(filenames)

filenames

colnames(tdm)<-filenames

tdm

tdm<- t(dtm) # t(dtm) – transpose matrix DTM into TDM

tf <- as.matrix(tdm)

idf <- log(ncol(tf) / (rowSums(tf != 0)))

tf[170:175,1:5]

idf[280:288]

idf\_sort <- sort(idf, decreasing=FALSE)

head(idf\_sort, 15)

tail(idf\_sort, 15)

# building tf-idf

idf1 <- diag(idf)

tf\_idf <- crossprod(tf, idf1)

colnames(tf\_idf) <- rownames(tf)

write.csv(as.matrix(tf\_idf),file="C:\\Users\\Uzytkownik\\OneDrive\\Pulpit\\politechnika\\TEXT MINING PROJEKT\\TFIDF.csv")

tf\_idf\_t<-t(tf\_idf) #transposed matrix tf\_idf

tf\_idf\_t [355:365,1:3]

mm\_s = as.matrix(tf\_idf\_t\_s)

#mm<-as.matrix(mm\_s[1:10,])

mm<-as.matrix(mm\_s) # for using all documents

#function cosineSim compute cosine similarity between document vectors

#converting to distance matrix sets diagonal elements to 0

cosineSim <- function(x){

as.dist(x%\*%t(x)/(sqrt(rowSums(x^2) %\*% t(rowSums(x^2)))))

}

#compute cosine similarity between document vectors

cs <- cosineSim(mm)

cs

write.csv(as.matrix(cs),file="C:\\Users\\Uzytkownik\\OneDrive\\Pulpit\\politechnika\\TEXT MINING PROJEKT\\data mining matrix\\DocumentCosine.csv")

#create the adjacency matrix

min\_cos<-0.2

cs[cs < min\_cos] <- 0

cs <- round(cs,3)

#save adjacency matrix to \*.csv file

write.csv(as.matrix(cs),file="C:\\Users\\Uzytkownik\\OneDrive\\Pulpit\\politechnika\\TEXT MINING PROJEKT\\data mining matrix\\DocumentAdjacencyMatrix.csv")

cs

#adjacency matrix

dat<-read.csv("C:\\Users\\Uzytkownik\\OneDrive\\Pulpit\\politechnika\\TEXT MINING PROJEKT\\data mining matrix\\DocumentAdjacencyMatrix.csv",

header = TRUE,

sep = ",",

colClasses = NA,

na.string = "NA",

skip = 0,

strip.white = TRUE,

fill = TRUE,

comment.char = "#",

stringsAsFactors = FALSE

)

dat

mm1 = as.data.frame.matrix(dat)

mm1=mm1[,-1]

mm1

filenames <- list.files(setwd("C:\\Users\\Uzytkownik\\OneDrive\\Pulpit\\politechnika\\TEXT MINING PROJEKT\\data mining texts"),pattern="\*.txt")

#filenames <-c(filenames[1:5])

filenames <-c(filenames) # for using all documents

filenames

#converting mm1 into matrix format

rownames(mm1)<-filenames

cs<-as.matrix(mm1)

cs

#initializing Igraph package

library(igraph)

#Creating undirected weighted graph

g=graph.adjacency(cs,mode="undirected",weighted=TRUE)

g

#Checking the undirected weighted graph attributes

list.vertex.attributes(g)

list.edge.attributes(g)

V(g)$name

E(g)$weight

#Calculate the degree of each Vertices and assign it’s to the Vertices size feature

deg <- graph.strength(g, mode="all")

deg

V(g)$size <- deg\*1

#Build the Vertices color

hc5 <- terrain.colors(5) # colors – https://www.r-bloggers.com/color-palettes-in-r/

g.max <- max(deg)

vcolors <- 5 - round(4 \*(deg / g.max))

vcolors <- hc5[vcolors]

vcolors

# Build the Graph Plot using different Layouts

lay\_1 <- layout.fruchterman.reingold(g)

plot.igraph(g,layout= lay\_1, edge.arrow.size=0.1,edge.width=E(g)$weight\*2, vertex.label=V(g)$

name,vertex.color=vcolors,vertex.size=V(g)$size, vertex.label.cex=1, node.size=0.1)

lay\_2 <- layout\_in\_circle

plot.igraph(g,layout= lay\_2, edge.arrow.size=0.1,edge.width=E(g)$weight\*2, vertex.label=V(g)$

name,vertex.color=vcolors,vertex.size=V(g)$size,vertex.label.cex=1)

lay\_3 <- layout\_with\_kk(g)

plot.igraph(g,layout= lay\_3, edge.arrow.size=0.1,edge.width=E(g)$weight\*30, vertex.label=V(g)$

name,vertex.color=vcolors,vertex.size=V(g)$size,vertex.label.cex=1)

lay\_4 <- layout\_randomly(g)

plot.igraph(g,layout= lay\_4, edge.arrow.size=0.1,edge.width=E(g)$weight\*30, vertex.label=V(g)$

name,vertex.color=vcolors,vertex.size=V(g)$size,vertex.label.cex=1)

lay\_5 <- layout\_on\_sphere(g)

plot.igraph(g,layout= lay\_5, edge.arrow.size=0.1,edge.width=E(g)$weight\*30, vertex.label=V(g)$

name,vertex.color=vcolors,vertex.size=V(g)$size,vertex.label.cex=1)

#Algorithm 1: edge betweenness (Newman-Girvan)

ceb <- cluster\_edge\_betweenness(g)

plot(ceb, g)

membership(ceb)

# Algorithm 2: based on propagating labels

clp <- cluster\_label\_prop(g)

plot(clp , g)

membership(clp)

# Algorithm 3: based on greedy optimization of modularity

cfg <- cluster\_fast\_greedy(as.undirected(g))

plot(cfg, as.undirected(g))

membership(cfg)

#Check the most significant 6 terms in each topic, transform them into the matrix format and save to to file

ldaOut.terms <- as.matrix(terms(ldaOut,42))

ldaOut.terms

write.csv(ldaOut.terms,file=paste("LDAGibbs",k,"TopicsToTerms.csv"))

#Transform the probabilities associated with each topic assignment into the matrix format and save to the file

topicProbabilities <- as.data.frame(ldaOut@gamma)

topicProbabilities

write.csv(topicProbabilities,file=paste("LDAGibbs",k,"TopicProbabilities.csv"))

#Transform the Topics into the matrix format and save to the File

ldaOut.topics <- as.matrix(topics(ldaOut))

ldaOut.topics

write.csv(ldaOut.topics,file=paste("LDAGibbs",k, "DocsToTopics.csv"))

# Cosine Distance transformed from Cosine is similarity based on dtm

mm = as.matrix(dtm)

cosineSim <- function(x){

as.dist(x%\*%t(x)/(sqrt(rowSums(x^2) %\*% t(rowSums(x^2)))))

}

cs <- cosineSim(mm)

d1 <- 1-cs

# Replace line:

#d1 <- dist(dtm, method="euclidian")