



Emilie JALRAS

**Mise en place de
systèmes de recommandations**

Rapport de Stage

du 16 juin au 5 septembre
tuteur entreprise : Benoît Baccot
tuteur école : Vincent Charvillat

Remerciements

Avant de commencer ce rapport, j'aimerais remercier les personnes grâce à qui mon stage chez Devatics a été rendu possible.

Je remercie donc particulièrement Benoit Baccot, mon tuteur au sein de Devatics, qui s'est patiemment occupé de moi durant toute la durée de mon stage, ainsi que Vincent Charvillat et Sandrine Mouysset, professeurs et chercheurs à l'ENSEEIH, qui m'ont permis de trouver ce stage, et m'ont suivie chaque semaine avec une motivation toujours présente.

Je remercie également Monsieur Romulus Grigoras, pour m'avoir accueillie au sein de son entreprise ainsi que les autres membres de l'équipe Devatics pour leur accueil chaleureux.

Sommaire

Introduction.....	3
1. L'entreprise.....	3
2. Les systèmes de recommandations.....	3
3. Objectif.....	4
Différentes méthodes.....	7
I. Les méthodes simples.....	10
1. Traitement des descriptions.....	10
a. Tokenisation.....	10
b. Stemming.....	10
c. Suppression de mots parasites.....	11
2. Vecteurs.....	12
a. Binaire.....	12
b. Fréquence.....	12
c. Similarité cosinus.....	14
3. Calcul des similarités.....	15
4. Visualisation : l'interface.....	15
II. Les méthodes augmentées.....	17
III. Les méthodes par clusters.....	18
1. Visualisation : les graphes.....	19
2. Les clusters.....	20
a. ACP.....	20
b. Meanshift.....	21
c. Similarités par clusters.....	23
Conclusion.....	25

Introduction

1. L'entreprise

Devatics est une entreprise basée à Toulouse, à côté de l'Enseeiht. Cette société propose une solution de commerce électronique qui se concentre sur la publicité et les technologies intelligentes d'analyse comportementale en temps réel. Fondée en 2010, elle est dotée d'une vingtaine de salariés, et est en pleine expansion. Elle travaille pour des clients tels que Kaporal Jeans, La Dépêche du Midi, TomPress, Teddy Smith et de nombreux autres.



2. Les systèmes de recommandations

La première chose que j'ai eu l'occasion de faire durant mon stage, a été de me renseigner sur les systèmes de recommandations existants.

Les systèmes de recommandations sont en plein essor, facilitant les ventes pour les sites d'e-commerce, ils ont été beaucoup développés par ces mêmes sites, par exemple Amazon :



The screenshot shows the Amazon.fr product page for the CD 'Kendji Girac'. The page features a large album cover image of the artist, a price of EUR 13,99, and a 'Go' button for the search bar. The navigation bar includes links for 'CD & Vinyles', 'Recherche détaillée', and 'Téléchargement de musique'. The product details section includes the artist's name, format, and a 'Continue with delivery in 1 day' option. The page also displays a 'Panier' (cart) icon and a 'Vendez sur Amazon' button.

Quels sont les autres articles que les clients achètent après avoir regardé cet article?

 Les Feux d'Artifice ~ Calogero CD
★★★★☆ (67)
EUR 15,99

 Fréro Delavega ~ Fréro Delavega CD
★★★★☆ (61)
EUR 12,99

 Merci pour ce moment de Valérie Trierweiler Broché
★★★★☆ (624)
EUR 20,00

 Oreflam - Edition Limitée ~ Luc Arbogast CD
★★★★☆ (50)
EUR 16,99

[Découvrez des articles similaires](#)

Produits fréquemment achetés ensemble

 +  +  **Prix pour les trois: EUR 40,97**
[Ajouter ces trois articles au panier](#)
Afficher la disponibilité du produit et le mode de livraison

Cet article : Kendji ~ Kendji Girac CD EUR 13,99

Fréro Delavega ~ Fréro Delavega CD EUR 12,99

Saltimbanque ~ Keen V CD EUR 13,99

Les clients ayant acheté cet article ont également acheté

Page 1 sur 13

 Fréro Delavega Fréro Delavega ★★★★☆ (61) CD EUR 12,99	 Tout Recommencer David Carreira ★★★★☆ (4) CD EUR 12,99	 Les Feux d'Artifice Calogero ★★★★☆ (67) CD EUR 15,99	 Oreflam - Edition Limitée Luc Arbogast ★★★★☆ (50) CD EUR 16,99	 Saltimbanque Keen V ★★★★☆ (28) CD EUR 13,99	 Mini World India ★★★★☆ (134) CD EUR 9,99	 Vieilles avec Toi Florent Pagny ★★★★☆ (238) CD EUR 11,83	 La Selection Patrick Fiori Patrick Fiori ★★★★☆ (1) CD EUR 11,48
--	---	---	---	--	---	---	--

Dans cet exemple en consultant la fiche produit d'un CD sur Amazon, on trouve en dessous de nombreuses recommandations, basées sur ce que les autres utilisateurs du site ont acheté. Ce sont donc des recommandations user-based.

On trouve en fait trois types de recommandations :

- user-based : ces recommandations sont basées sur les comportements des utilisateurs. (comme dans les exemples ci-dessus)
- content-based : ces recommandations sont basées uniquement sur les produits, par exemple en recommandant des produits similaires au produit étudié.
- mixed : c'est un mélange des deux méthodes précédentes : ces recommandations sont faites en utilisant à la fois des caractéristiques du produit, et une étude du comportement des utilisateurs.

Lors de mon stage je me suis concentrée uniquement sur les systèmes de recommandation basés sur le contenu.

3. Objectif

L'objectif de mon stage était de réaliser des recommandations à partir de données réelles fournies par un des sites clients de Devatics : Phase Eight.



Phase Eight est une chaîne de magasins de prêt à porter née en 1979 au Royaume-Uni et qui s'est étendue depuis à l'international (notamment en Europe, au Moyen-Orient ou encore en Asie).

En 2008 ils ont mis en place leur site internet, d'où sont effectuées des ventes vers une trentaine de pays différents. Ce site est visité par près de 800000 visiteurs chaque mois.

Actuellement voici la vue qu'a un acheteur lorsqu'il désire consulter des informations sur un produit.

The screenshot shows the Phase Eight website interface. At the top, the logo 'Phase Eight' is on the left, and navigation links for 'Shipping To' (France), 'My Account', 'Wish List', and 'My Bag' (0 items) are on the right. A search bar is also present. Below the navigation, a banner reads 'WE NOW SHIP TO 28 EUROPEAN COUNTRIES'. The breadcrumb trail is 'Home > Dresses > Jasmin Dress'. The main product image is a white sleeveless dress with a floral print and a full skirt. To the left of the main image are five smaller thumbnail images showing different views of the dress. To the right, the product details are listed: 'Jasmin Dress', price '€185.00', and reference 'ref: 202633000'. A description states: 'A lightweight cotton fit and flare dress featuring a V-neck and soft floral print. Styled with a waist detail and tulle underlay for a full skirt to accentuate the feminine silhouette. Finished with a keyhole button back and side zip.' Below the description are expandable sections for 'FABRIC, CARE & LENGTH', 'DELIVERY & RETURNS', and 'COLOUR'. The 'SELECT SIZE' section shows options 06, 08, 10, 12, 14, 16, 18, and 20, with checkboxes for each. A 'Size Guide' link is provided. Below the size selection, there is a 'Quantity' selector set to 1, an 'ADD TO SHOPPING BAG' button, and an 'Add to wish list' link. At the bottom, there are social media sharing buttons for Facebook (Like), Twitter (Twee), Pinterest (Pin it), Google+ (g+1), and Email (SHARE). Below the main product image, there are two sections: 'GOES WITH...' and 'YOU MAY ALSO LIKE...'. The 'YOU MAY ALSO LIKE...' section displays four smaller images of different dresses: a dark floral dress, a bright pink dress, a dark floral dress with a different pattern, and a white floral dress.

On voit en bas une catégorie « You may also like... », dans laquelle sont effectuées des recommandations, qui propose une liste de produits similaires.

Mais actuellement ces produits ne sont pas choisis de manière automatique, Phase Eight fournit une liste fixe de produits à recommander pour chaque produit.

L'objectif de mon stage était de mettre en place une méthode de calcul automatique des produits similaires.

L'idée était de se baser sur les caractéristiques accessibles du produit telles que le nom, la description, le prix ou encore la couleur. Pour cela j'avais à ma disposition un serveur de tests sur lequel se trouvait notamment une copie d'une base de données Infobright contenant les caractéristiques des produits Phase Eight, qu'ont mise en place mes collègues. Elle contenait 139816 produits.

J'ai développé mes programmes en Python, un langage que je n'avais jamais utilisé auparavant, et ai donc découvert au cours de ce stage.

Descriptions Binaire



Clemence Wedding Dress



Similarite : 0.36313651960128146
Eva Grosgrain Shoes

Similarite description : undefined
Similarite nom : undefined
id produit : undefined
(id produit d origine : undefined)

toe sho
peep satin look
feat



Similarite : 0.350438220252312
Contessa Fit And Flare Dress

Similarite description : undefined
Similarite nom : undefined
id produit : undefined
(id produit d origine : undefined)

sho peep toe dress fit
feat long



Similarite : 0.3499271061188266
Ally Satin Block Heel Shoes

Similarite description : undefined
Similarite nom : undefined
id produit : undefined
(id produit d origine : undefined)

sho peep
toe satin look
feat

Descriptions Frequence



Clemence Wedding Dress



Similarite : 0.368444471143871
Fit Pleat Peep Toe Shoes

Similarite description : undefined
Similarite nom : undefined
id produit : undefined
(id produit d origine : undefined)

sho peep
toe satin



Similarite : 0.3215542239959113
Eva Grosgrain Shoes

Similarite description : undefined
Similarite nom : undefined
id produit : undefined
(id produit d origine : undefined)

toe sho
peep satin look
feat



Similarite : 0.3040934733743832
Myia Rose Trim Shoes

Similarite description : undefined
Similarite nom : undefined
id produit : undefined
(id produit d origine : undefined)

sho
peep toe
satin

Ces erreurs sont dues à la présence d'éléments non purement descriptifs dans la description. En effet, dans l'exemple ci-dessus il était recommandé dans la description de porter la robe avec des chaussures à talon en satin, et le résultat est que cela parasite le calcul de similarités. Je ne recommande plus des robes mais uniquement des chaussures à talon en satin.

Pour éviter ce genre de problèmes, la solution était d'utiliser la catégorie du produit. Si le produit est une robe de mariée, on ne proposera alors plus des chaussures, mais bien des robes de mariée.

La catégorie du produit ne faisant pas partie des données fournies par Phase Eight, il a donc fallu chercher des moyens détournés pour pouvoir la prendre en compte. D'où les deux méthodes suivantes.

- **méthodes augmentées :**

L'idée dans ces méthodes était de prendre en compte les autres informations fournies sur le produit en base de données : le nom, le prix et enfin la couleur. Pour cela je calculais la similarité sur chacune des données, et je faisais une somme pondérée du tout. Pour le prix je me limitais à regarder si les deux produits appartenaient à la même tranche de prix.

Finalement il s'est avéré que les informations sur la couleur étaient peu exploitables, car décrites avec trop de nuances (par exemple pour du vert on trouvait : « vert », « forêt », « lierre », « olive », « émeraude », « jade », « feuille »...). Il devenait presque impossible que deux produits soient considérés comme ayant la même couleur.

Pour le prix il s'est avéré que l'utilisation de tranches parasitait la similarité. J'ai tenté quelques méthodes utilisant à la place une distance entre prix, mais cela s'est avéré peu satisfaisant.

Au final j'ai donc gardé juste le nom et la description.

L'ajout du nom dans le calcul de similarité donne des informations sur la catégorie, et règle les problèmes évoqués précédemment.

Binaire (description + nom)

Clemence Wedding Dress

Camellia Wedding Dress
 Similarite : 0.5090557082545596
 Similarite description : 0.27269270636988
 Similarite nom : 0.6666666666666667
 id produit : 122
 (id produit d'origine : 47)
 wed flo cre dress hem, fit, feat
 img

Stephanie Wedding Dress
 Similarite : 0.502868899974728
 Similarite description : 0.2571722499368199
 Similarite nom : 0.6666666666666667
 id produit : 269
 (id produit d'origine : 47)
 flo wed satin look
 img

Mariette Wedding Dress
 Similarite : 0.49304842103984714
 Similarite description : 0.2226210525996177
 Similarite nom : 0.6666666666666667
 id produit : 16
 (id produit d'origine : 47)
 wed flo dress fit img
 img

Frequence (description + nom avec IDF)

Clemence Wedding Dress

Clemence Floral Tunic
 Similarite : 0.4048399103072752
 Similarite description : 0.01280181691222496
 Similarite nom : 0.645531972570643
 id produit : 485
 (id produit d'origine : 47)
 hem feat
 img

Stephanie Wedding Dress
 Similarite : 0.20510325881871644
 Similarite description : 0.1551799477453006
 Similarite nom : 0.23838546620099366
 id produit : 269
 (id produit d'origine : 47)
 flo wed satin look
 img

Mariette Wedding Dress
 Similarite : 0.1991731750041636
 Similarite description : 0.1736960651492809
 Similarite nom : 0.2165791490741873
 id produit : 16
 (id produit d'origine : 47)
 wed flo dress fit img
 img

Frequence (description + nom sans IDF)

Clemence Wedding Dress

Mariette Wedding Dress
 Similarite : 0.46947842605971235
 Similarite description : 0.1736960651492809
 Similarite nom : 0.6666666666666667
 id produit : 16
 (id produit d'origine : 47)
 wed flo dress fit img
 img

Stephanie Wedding Dress
 Similarite : 0.4620719790981202
 Similarite description : 0.1551799477453006
 Similarite nom : 0.6666666666666667
 id produit : 269
 (id produit d'origine : 47)
 flo wed satin look
 img

Elodie Wedding Dress
 Similarite : 0.4527183180084917
 Similarite description : 0.13179579502122937
 Similarite nom : 0.6666666666666667
 id produit : 230
 (id produit d'origine : 47)
 wed dress scallop
 satin hem
 img

Ces différentes versions seront expliquées plus loin dans le rapport.

- **méthodes basées sur les clusters :**

Dans cette méthode on n'utilise plus les noms directement comme précédemment. Cette fois on calcule préalablement des clusters à partir des noms des produits, et ensuite on sélectionne lors du calcul des similarités uniquement les produits appartenant aux mêmes clusters préalablement calculés.

Cluster Binaire

Clemence Wedding Dress

Pollyanna Ribbon Tapework Wedding Dress
 Similarite : 0.28365431446558775
 Similarite description : 0.28365431446558775
 Similarite nom : 0.516397794943222
 id produit : 766
 (id produit d'origine : 47)
 stun wed cre dress fit, feat
 feat
 img

Camellia Wedding Dress
 Similarite : 0.2726392706363988
 Similarite description : 0.2726392706363988
 Similarite nom : 0.6666666666666667
 id produit : 122
 (id produit d'origine : 47)
 wed flo cre dress hem, fit, feat
 img

Flora Embroidered Wedding Dress
 Similarite : 0.2700208624336608
 Similarite description : 0.2700208624336608
 Similarite nom : 0.5773502691896258
 id produit : 100
 (id produit d'origine : 47)
 stun wed satin look dress
 feat img
 img

Cluster

Clemence Wedding Dress

Mariette Wedding Dress
 Similarite : 0.1736960651492809
 Similarite description : 0.1736960651492809
 Similarite nom : 0.6666666666666667
 id produit : 16
 (id produit d'origine : 47)
 wed flo dress fit img
 img

Stephanie Wedding Dress
 Similarite : 0.1551799477453006
 Similarite description : 0.1551799477453006
 Similarite nom : 0.6666666666666667
 id produit : 269
 (id produit d'origine : 47)
 flo wed satin look
 img

Belle Tulle Wedding Dress
 Similarite : 0.14426746774181207
 Similarite description : 0.14426746774181207
 Similarite nom : 0.5773502691896257
 id produit : 46
 (id produit d'origine : 47)
 embroidery wed look
 dress, feat
 img

Cette méthode présente l'avantage d'éviter que certaines caractéristiques des noms, comme leur longueur, vienne parasiter la similarité descriptive pour plusieurs produits de la même catégorie. Malheureusement les clusters que nous avons pu calculés n'étaient pas parfaits, et engendraient donc d'autres erreurs au niveau de la similarité, que nous montrerons plus loin dans le rapport.

I. Les méthodes simples

1. Traitement des descriptions

La première étape pour calculer les similarités consiste à découper les descriptions en listes de mots (c'est la tokenisation) puis à prendre les racines des mots, les stems (il s'agit donc du stemming). Pour cela j'ai utilisé une bibliothèque en Python qui permet de traiter des langages : NLTK.

Cette bibliothèque met à notre disposition plusieurs algorithmes qui réalisent la tokenisation et le stemming.

a. Tokenisation

Prenons une description :

```
This full length evening dress in a floral textured fabric fits to the figure beautifully with a feminine fishtail hem. In a striking scarlet shade, this dress features a soft sweetheart neckline and a V-back. Fastens with a concealed centre back zip.
```

Dans la liste des tokenizers disponibles j'en ai retenus deux qui semblaient convenir pour découper une phrase en mots : SpaceTokenizer, qui découpe la phrase selon les espaces, et TreeBank, censé séparer aussi la ponctuation. J'ai fait plusieurs tests dans le but de comparer les durées d'exécution et les résultats obtenus.

```
Tokenization SpaceTokenizer
0.00400018692017
['This', 'full', 'length', 'evening', 'dress', 'in', 'a', 'floral', 'textured', 'fabric', 'fits', 'to', 'the', 'figure', 'beautifully', 'with', 'a', 'feminine', 'fishtail', 'hem.', 'In', 'a', 'striking', 'scarlet', 'shade', ',', 'this', 'dress', 'features', 'a', 'soft', 'sweetheart', 'neckline', 'and', 'a', 'V-back.', 'Fastens', 'with', 'a', 'concealed', 'centre', 'back', 'zip.']

Tokenization Treebank
0.00699996948242
['This', 'full', 'length', 'evening', 'dress', 'in', 'a', 'floral', 'textured', 'fabric', 'fits', 'to', 'the', 'figure', 'beautifully', 'with', 'a', 'feminine', 'fishtail', 'hem.', 'In', 'a', 'striking', 'scarlet', 'shade', ',', 'this', 'dress', 'features', 'a', 'soft', 'sweetheart', 'neckline', 'and', 'a', 'V-back.', 'Fastens', 'with', 'a', 'concealed', 'centre', 'back', 'zip', '.']
```

Au final il s'est avéré que Treebank était un peu plus lent en général, et ne parvenait pas à retirer la ponctuation partout. Par exemple dans l'exemple ci-dessus il reste un point sur « hem. ».

J'ai donc préféré garder SpaceTokenizer et faire un « strip » pour retirer en post-traitement la ponctuation sur les mots de la liste.

b. Stemming

Pour le stemming j'ai testé différents algorithmes fournis par la bibliothèque NLTK. J'en ai finalement retenus deux pertinents : Lancaster et Porter.

Ces deux algorithmes sont adaptés pour le langage anglais, toutefois NLTK fournit d'autres algorithmes si l'on désire utiliser une langue différente, comme le français.

Reprenons la description précédente sous sa nouvelle forme tokenisée :

```
['This', 'full', 'length', 'evening', 'dress', 'in', 'a', 'floral', 'textured',  
'fabric', 'fits', 'to', 'the', 'figure', 'beautifully', 'with', 'a', 'feminine',  
'fishtail', 'hem.', 'In', 'a', 'striking', 'scarlet', 'shade,', 'this', 'dress'  
, 'features', 'a', 'soft', 'sweetheart', 'neckline', 'and', 'a', 'V-back.', 'Fas  
tens', 'with', 'a', 'concealed', 'centre', 'back', 'zip.']
```

Les résultats des tests, effectués avant suppression de la ponctuation, sont les suivants :

```
Stemming Porter  
0.00300002098083  
['Thi', 'full', 'length', 'even', 'dress', 'in', 'a', 'floral', 'textur', 'fabri  
c', 'fit', 'to', 'the', 'figur', 'beauti', 'with', 'a', 'feminin', 'fishtail', '  
hem.', 'In', 'a', 'strike', 'scarlet', 'shade,', 'thi', 'dress', 'featur', 'a',  
'soft', 'sweetheart', 'necklin', 'and', 'a', 'V-back.', 'Fasten', 'with', 'a', '  
conceal', 'centr', 'back', 'zip.']
```

```
Stemming Lancaster  
0.00899982452393  
['thi', 'ful', 'leng', 'ev', 'dress', 'in', 'a', 'flor', 'text', 'fabr', 'fit',  
'to', 'the', 'fig', 'beauty', 'with', 'a', 'feminin', 'fishtail', 'hem.', 'in',  
'a', 'striking', 'scarlet', 'shade,', 'thi', 'dress', 'feat', 'a', 'soft', 'swee  
theart', 'necklin', 'and', 'a', 'v-back.', 'fast', 'with', 'a', 'cont', 'cent',  
'back', 'zip.']
```

L'algorithme de Porter est plus rapide, mais semble moins précis, par exemple pour le terme « beautifully », Porter trouve « beauti », tandis que Lancaster identifie que la racine est « beauty ».

J'ai donc finalement utilisé Lancaster. Voici le résultat dans le cas où on a bien retiré la ponctuation avant d'effectuer le stemming :

```
['thi', 'ful', 'leng', 'ev', 'dress', 'in', 'a', 'flor', 'text', 'fabr', 'fit',  
'to', 'the', 'fig', 'beauty', 'with', 'a', 'feminin', 'fishtail', 'hem', 'in', '  
a', 'striking', 'scarlet', 'shad', 'thi', 'dress', 'feat', 'a', 'soft', 'swee  
art', 'necklin', 'and', 'a', 'v-back', 'fast', 'with', 'a', 'cont', 'cent', 'bac  
k', 'zip']
```

c. Suppression mots parasites

Enfin, la dernière étape consiste à retirer les mots peu importants qui reviennent souvent dans les phrases. Pour cela j'ai utilisé « stopwords », fourni par nltk, qui contient une liste de mots peu significatifs :

Stopwords

```
['i', 'me', 'my', 'myself', 'we', 'our', 'ours', 'ourselves', 'you', 'your', 'yours', 'yourself', 'yourselves', 'he', 'him', 'his', 'himself', 'she', 'her', 'hers', 'herself', 'it', 'its', 'itself', 'they', 'them', 'their', 'theirs', 'themselves', 'what', 'which', 'who', 'whom', 'this', 'that', 'these', 'those', 'am', 'is', 'are', 'was', 'were', 'be', 'been', 'being', 'have', 'has', 'had', 'having', 'do', 'does', 'did', 'doing', 'a', 'an', 'the', 'and', 'but', 'if', 'or', 'because', 'as', 'until', 'while', 'of', 'at', 'by', 'for', 'with', 'about', 'against', 'between', 'into', 'through', 'during', 'before', 'after', 'above', 'below', 'to', 'from', 'up', 'down', 'in', 'out', 'on', 'off', 'over', 'under', 'again', 'further', 'then', 'once', 'here', 'there', 'when', 'where', 'why', 'how', 'all', 'any', 'both', 'each', 'few', 'more', 'most', 'other', 'some', 'such', 'no', 'nor', 'not', 'only', 'own', 'same', 'so', 'than', 'too', 'very', 's', 't', 'can', 'will', 'just', 'don', 'should', 'now']
```

J'ai donc stemmée cette liste de mots, puis j'ai retiré de mes descriptions tous les stems appartenant à cette nouvelle liste.

2. Vecteurs

L'étape suivante consiste à stocker chacune des descriptions traitées sous forme de vecteurs en base de données.

Pour cela j'ai créé des bases de données MySQL de taille nombreDeProduits * nombreDeStems.

a. vecteurs binaires :

idProduct	_gown	_gingham	_rol	_edg	_chain	_funct	_tapework	_cobalt	_sleev	_ellips	_skim	_dainty	_go	_cuffs	_more	_ros	_encrust	_rom	_graph
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
8	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Chaque ligne est un vecteur qui correspond à une description, chaque colonne correspond à un stem.

Dans le cas des vecteurs binaires on mettra un 1 si le stem est présent et zéro sinon.

b. vecteurs fréquence :

Dans le cas des vecteurs fréquence, si le stem est présent, alors la valeur enregistrée sera TF * IDF.

idProduct	_gown	_gingham	_rol	_edg	_chain	_funct	_tapework	_cobalt	_sleev	_ellips	_skim	_dainty	_go	_cuffs_more	_ros	_encrust	_rom	_graph
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0.0517179	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0	0.126603	0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0	0	0.0646474	0	0	0	0	0	0	0	0	0
8	0	0	0	0.57789	0	0	0	0	0	0	0	0	0	0	0	0	0	0

➤ $TF = \text{nombreOccurrencesStem} / \text{nombreStems}$

Par exemple voici les valeurs de TF pour les stems de la description suivante :

```
id :
55
```

```
Stems retenus :
['print', 'jersey', 'top', 'cent', 'wrap', 'front', '3/4', 'leng', 'sleev', 'top',
', 'fit', 'flat', 'ruch', 'sid']
```

liste des stems et leur fréquence TF :

```
print                wrap                sleev                ruch
0.0714285714286     0.0714285714286     0.0714285714286     0.0714285714286

jersey              front
0.0714285714286     0.0714285714286

top                 3/4                fit
0.142857142857      0.0714285714286     0.0714285714286

cent                leng                flat
0.0714285714286     0.0714285714286     0.0714285714286
```

on voit que le mot « top » est en double, il a donc un TF deux fois plus élevé que les autres.

➤ $IDF = \log(\text{nombreDescriptions} / \text{nombreDescriptionsContenantStem})$

L'IDF indique l'importance d'un stem par rapport aux autres stems. Par exemple :

```
_sleev                _dress                _edg
nbeDescriptions :      nbeDescriptions :      nbeDescriptions :
10000.0                10000.0                10000.0
NbeDescriptionsAvecStemPresent : NbeDescriptionsAvecStemPresent : NbeDescriptionsAvecStemPresent :
5962                   3814                   312
IDF :                  IDF :                   IDF :
0.517179097738        0.963906585816        3.46733718417
```

<code>_gown</code>	<code>_go</code>
<code>nbeDescriptions :</code> <code>10000.0</code>	<code>nbeDescriptions :</code> <code>10000.0</code>
<code>NbeDescriptionsAvecStemPresent :</code> <code>169</code>	<code>NbeDescriptionsAvecStemPresent :</code> <code>1</code>
<code>IDF :</code> <code>4.08044165705</code>	<code>IDF :</code> <code>9.21034037198</code>

Les termes « sleeve » (manche), et « dress » (robe) sont présents dans énormément de descriptions (plus de la moitié pour « sleeve »), ce sont donc des mots peu significatifs, qui ne nous aideront pas vraiment à déterminer si deux produits sont similaires. En revanche des termes rares, comme « gown » qui désigne des robes plus formelles ou « edge » (bord) se verront accorder plus d'importance, en effet s'ils sont présents dans les deux descriptions c'est beaucoup plus significatif que pour les termes précédents.

c. Similarité cosinus

Une fois les vecteurs, binaires ou fréquence selon le cas, stockés en base de données, on calcul la similarité en faisant le cosinus de ces deux vecteurs :

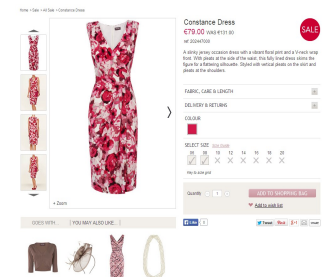
soient v_1 et v_2 les vecteurs considérés : $\text{similarité} = \frac{v_1 \cdot v_2}{(\|v_1\| \cdot \|v_2\|)}$

Voici un exemple de comparaison entre deux produits :

`Id1 :`
`155`
`Constance Dress`

A slinky jersey occasion dress with a vibrant floral print and a V-neck wrap front. With pleats at the side of the waist, this fully lined dress skims the figure for a flattering silhouette. Styled with vertical pleats on the skirt and pleats at the shoulders.

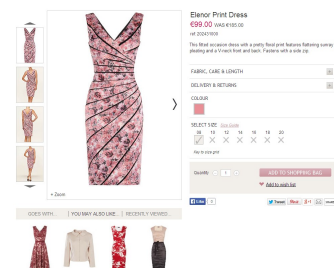
`['slinky', 'jersey', 'occas', 'dress', 'vibr', 'flor', 'print', 'v_neck', 'wrap', 'front', 'ple', 'sid', 'waist', 'ful', 'lin', 'dress', 'skim', 'fig', 'flat', 'silhouet', 'styl', 'vert', 'ple', 'skirt', 'ple']`



`Id2 :`
`560`
`Elenor Print Dress`

This fitted occasion dress with a pretty floral print features flattering sunray pleating and a V-neck front and back. Fastens with a side zip.

`['fit', 'occas', 'dress', 'pretty', 'flor', 'print', 'feat', 'flat', 'sunray', 'ple', 'v_neck', 'front', 'back', 'fast', 'sid', 'zip']`



Voici les stems communs trouvés :

`Stems communs :`
`['occas', 'dress', 'flor', 'print', 'flat', 'ple', 'v_neck', 'front', 'sid']`

Voici les TF – IDF sur les deux descriptions :

slinky :	0.04,	3.04913305028			
jersey :	0.04,	0.967584026262			
occas :	0.04,	2.57046453815			
dress :	0.08,	0.963906585816			
vibr :	0.04,	4.08637639257	fit :	0.0625,	0.807436326962
flor :	0.04,	2.06751297081	occas :	0.0625,	2.57046453815
print :	0.04,	1.92689214322	dress :	0.0625,	0.963906585816
v_neck :	0.04,	2.28572797593	pretty :	0.0625,	1.89512198222
wrap :	0.04,	2.74887219562	flor :	0.0625,	2.06751297081
front :	0.04,	1.35751255969	print :	0.0625,	1.92689214322
ple :	0.12,	2.79852210427	feat :	0.0625,	0.75950068111
sid :	0.04,	1.76142426943	flat :	0.0625,	1.17506146083
waist :	0.04,	1.86175284105	sunray :	0.0625,	5.57275421225
ful :	0.04,	1.06798609513	ple :	0.0625,	2.79852210427
lin :	0.04,	1.55589714551	v_neck :	0.0625,	2.28572797593
dress :	0.08,	0.963906585816	front :	0.0625,	1.35751255969
skim :	0.04,	4.05128507276	back :	0.0625,	1.37594806907
fig :	0.04,	2.97006452681	fast :	0.0625,	1.62353683681
flat :	0.04,	1.17506146083	sid :	0.0625,	1.76142426943
silhouet :	0.04,	1.70374859191	zip :	0.0625,	1.59899265457
styl :	0.04,	1.79155948923			
vert :	0.04,	5.8781358618			
ple :	0.12,	2.79852210427			
skirt :	0.04,	2.59829933714			
ple :	0.12,	2.79852210427			

On observe une forte similarité entre les deux produits, voici le résultat dans le cas binaire (à gauche) et fréquence(à droite) :

Norme1 :	4.69041575982	Norme1 :	0.585851324303
Norme2 :	4.0	Norme2 :	0.55151857852
Similarite :	0.4797016118	Similarite :	0.398466898142

3. Calcul des similarités

La dernière étape du calcul des similarités, consiste à appliquer la similarité cosinus 2 à 2 pour notre produit étudié et chaque autre produit de la base de données (je me limite ici à 1000 produits). Je garde les N produits les plus similaires (pour l'interface j'ai fixé N à 5), en supprimant les doublons (très nombreux dans la base de données qui m'a été fournie).

4. Visualisation : l'interface

Une démarche importante chez Devatics est de pouvoir présenter concrètement son travail aux clients, on m'a donc demandé de réaliser une interface pour permette de consulter les similarités des produits de manière interactive.

J'ai donc mis en place un serveur Python ainsi qu'une page html avec Javascript et CSS associés.

La première version de l'interface tirait un produit au hasard et affichait

simplement les résultats :



Suite

On m'a demandé ensuite de permettre au client de choisir la référence du produit étudié, ce que j'ai fait à l'aide d'un formulaire.

Frequence



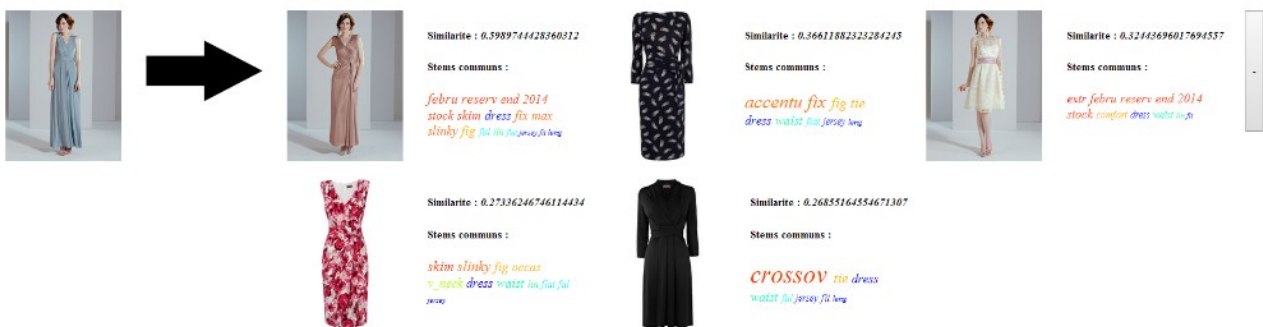
Binaire



Reference :

J'ai aussi du ajouter à l'affichage la similarité du produit et cacher les produits qui dépassaient de la ligne en les faisant apparaître lorsqu'on clique sur un bouton.

Frequence



Binaire



Enfin, on m'a demandé d'ajouter un nuage de mots à partir des stems communs, qui permette d'avoir une compréhension visuelle du TF-IDF de ces différents stems.

Pour cela j'ai choisi d'appliquer une échelle de couleur sur les stems en fonction de leur IDF : de rouge pour un fort IDF à bleu pour un IDF très faible.

J'ai ensuite appliquée une échelle de tailles sur les stems en fonction de leur TF*IDF.

Frequence



Sur l'exemple ci-dessus le stem « tul » en orange est en très gros et a donc un fort TF*IDF, et un fort TF puisqu'il est plus gros que « model », en rouge donc avec un très fort IDF (très rare)

J'ai aussi mise en place une version affichant les diverses similarités de description, nom, prix et couleur pour la version augmentée.

Descriptions Binaire



Descriptions Frequence



I. Les méthodes augmentées

Comme expliqué dans l'introduction, dans cette méthode, je me suis limitée à rajouter la dimension du nom à celle de la description, les résultats obtenus avec les couleurs et le prix n'étant pas pertinents.

Je calcule donc comme précédemment la similarité sur les descriptions, puis j'utilise la même méthode sur les noms.



Ensuite je fais simplement une somme pondérée de ces deux similarités, avec

un coefficient de 0,6 pour le nom et 0,4 pour la description.


J'ai du retirer l'IDF pour les noms en Frequence, car dans cette méthode on cherche en fait via le nom à trouver une catégorie. Les mots désignant des catégories seront ceux à faible IDF, tandis que les mots rares, comme les noms propres, avec un fort IDF, sont peu importants.


Ce problème avec l'IDF est illustré ci-dessous : une robe de mariée appelée « Clémence Wedding Dress » se trouvait associée à une tunique « Clémence Floral Tunic ». Ce problème a bien disparu dans le troisième cas, avec les IDF non pris en compte pour les noms.

Binaire (description + nom)



 →  **Clémence Wedding Dress**

Similarité : 0.5990557082545596
Clémence Wedding Dress
Similarité description : 0.2726392706363988
Similarité nom : 0.6666666666666667
id produit : 122
(id produit d'origine : 47)
wed flo **crush** **hem** **fit** **long**


 **Stephanie Wedding Dress**
Similarité : 0.502868899974728
Similarité description : 0.2571722499368199
Similarité nom : 0.6666666666666667
id produit : 269
(id produit d'origine : 47)
flo **wed** **satin** **look** **long**


 **Mariette Wedding Dress**
Similarité : 0.49304842102984714
Similarité description : 0.3226210523994177
Similarité nom : 0.6666666666666667
id produit : 16
(id produit d'origine : 47)
wed **flo** **dress** **fit** **long**

Frequence (description + nom avec IDF)



 →  **Clémence Wedding Dress**

Similarité : 0.4048399103072752
Clémence Floral Tunic
Similarité description : 0.01280181692223496
Similarité nom : 0.66551972570643
id produit : 455
(id produit d'origine : 47)
hem **fit**


 **Stephanie Wedding Dress**
Similarité : 0.20510225881871644
Similarité description : 0.2551799477453006
Similarité nom : 0.23835146620099366
id produit : 269
(id produit d'origine : 47)
flo **wed** **satin** **look** **long**


 **Mariette Wedding Dress**
Similarité : 0.1991731750041636
Similarité description : 0.1736960651492809
Similarité nom : 0.3261578498741873
id produit : 16
(id produit d'origine : 47)
wed **flo** **dress** **fit** **long**

Frequence (description + nom sans IDF)

 →  **Clémence Wedding Dress**

Similarité : 0.46947842605971235
Mariette Wedding Dress
Similarité description : 0.1736960651492809
Similarité nom : 0.6666666666666666
id produit : 16
(id produit d'origine : 47)
wed **flo** **dress** **fit** **long**

 **Stephanie Wedding Dress**
Similarité : 0.4620719790981202
Similarité description : 0.1551799477453006
Similarité nom : 0.6666666666666666
id produit : 269
(id produit d'origine : 47)
flo **wed** **satin** **look** **long**

 **Elodie Wedding Dress**
Similarité : 0.4527182180084917
Similarité description : 0.1317957950212297
Similarité nom : 0.6666666666666666
id produit : 259
(id produit d'origine : 47)
wed **dress** **scallop** **satin** **hem**

Toutefois l'influence des noms propres reste quand même présent, et si en plus la description est similaire alors même sans IDF on peut obtenir ceci :

Frequence (description + nom sans IDF)

 →  **Melinda Jacket**

Similarité : 0.5265701284863964
Alice Jacket
Similarité description : 0.566425212159811
Similarité nom : 0.4999999999999999
id produit : 274
(id produit d'origine : 203)
edg **jacket**
ey **match** **complet**
hook **pair** **occas** **look**
fast **flat** **dress** **fit** **long** **slim**

 **Melinda Dress**
Similarité : 0.522496838886248
Similarité description : 0.5562420972165621
Similarité nom : 0.4999999999999999
id produit : 172
(id produit d'origine : 203)
crush **melind**
match **complet** **pair**
jacket **occas** **look** **dress**
styl **flat** **fit** **long**

 **Helena Jacket**
Similarité : 0.4121131279201809
Similarité description : 0.2002828480064237
Similarité nom : 0.4999999999999999
id produit : 165
(id produit d'origine : 203)
jacket **tail** **match**
complet **pair** **occas**
look **styl** **flat** **dress** **fit** **long** **slim**

C'est une des raisons pour lesquelles on a désiré mettre en place un vrai calcul des catégories.

L'autre raison est liée à la visualisation, comme nous allons le voir dans le prochain paragraphe.

II. Les méthodes avec clusters

1. Visualisation : les graphes

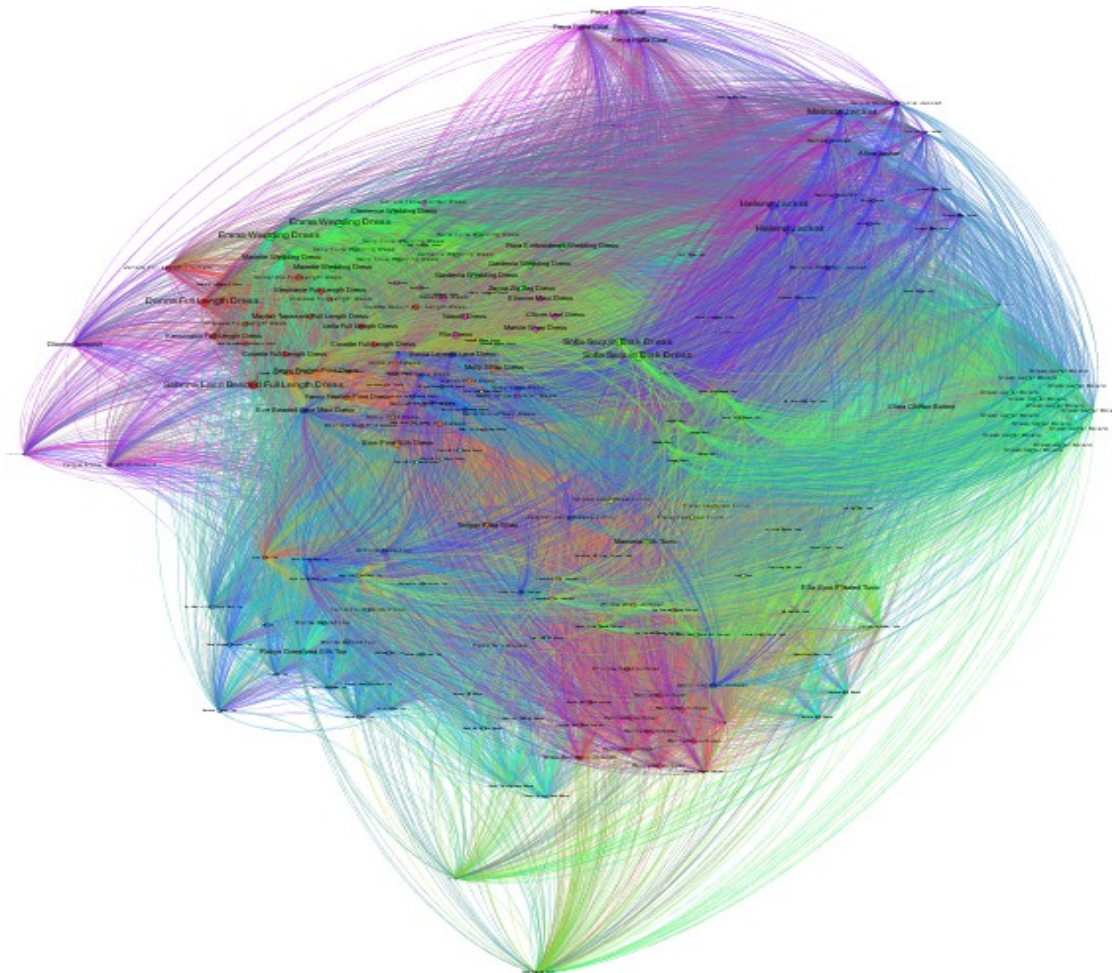
Une autre problématique qui m'a été posée pendant mon stage était de permettre au client de visualiser les résultats des calculs de similarités par des graphes.

Pour cela j'ai utilisé le logiciel Gephi. Ce logiciel permet à partir d'une liste des nœuds et arêtes ainsi que leurs caractéristiques de dessiner le graphe correspondant.

Je créais donc une table en base de données contenant les nœuds de mon graphe (ici les produits), ainsi qu'une autre table avec les arêtes entre deux nœuds et leur poids (correspondant à la similarité des deux produits/nœuds).

A partir de là il fallait régler les paramètres avec Gephi pour tracer le graphe, notamment choisir une spatialisation. J'en ai testé plusieurs et ai fini par en choisir une appelée Force Atlas.

Malheureusement utilisées telles quelles mes données me donnent un résultat in-interprétable :



Les couleurs dans le graphe correspondent aux clusters que j'ai calculés avec l'ACP Meanshift comme je l'expliquerai plus loin.

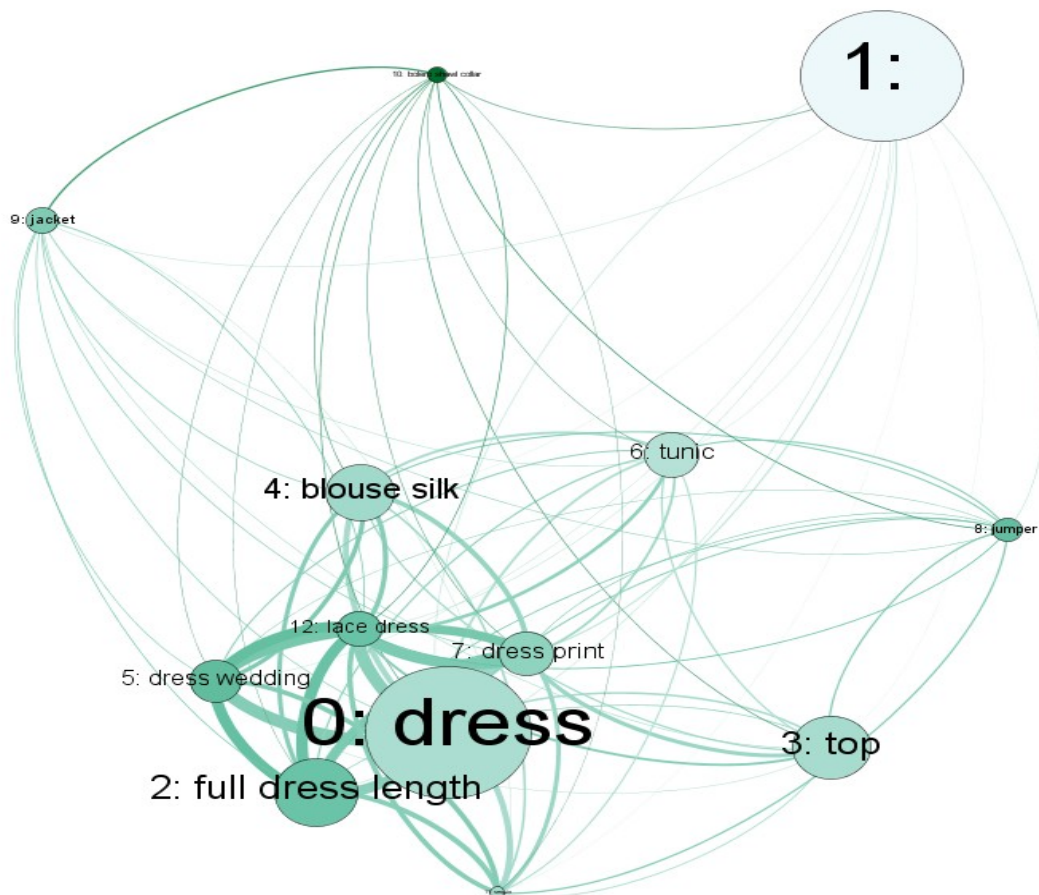
J'ai donc du regrouper les produits par clusters pour permettre d'obtenir un affichage plus lisible.

Chaque nœud est une catégorie, sa taille dépendant du nombre de produits dans la catégorie, et sa couleur correspondant à la similarité intra catégorie. Plus les produits d'une même catégorie sont similaires, plus le nœud sera foncé.

Le label du nœud est calculé de la façon suivante :

Je sélectionne les mots des noms de produits de la catégorie dont l'IDF est supérieur à un certain seuil. (ici 1)

Chaque lien a pour poids la similarité entre les deux catégories qu'il relie.



Pour calculer les similarités intra-catégories et inter-catégories je me suis limitée à 40 produits des par catégorie, en raison des temps de calculs assez élevés.

2. Les clusters

Je vais maintenant expliquer la méthode que j'ai mise en place pour le calcul de catégories.

Ce calcul est effectué à partir des vecteurs fréquence des noms stockés préalablement en base de données.

a. ACP (Analyse en Composantes Principales)

Cette méthode est utile pour réduire les dimensions que l'on utilise.

J'ai au départ une matrice de taille nombreDeProduits * nombreDeStems, et je vais obtenir une matrice de taille nombreDeProduits * q, avec q le nombre de composantes principales désirées.

Voici l'algorithme, que j'ai codé d'abord en Matlab, puis en Python :

*Soit X la matrice de taille nbProduits * nbMots qui contient les tf*idf de chaque mot pour chaque nom de produit.*

Soit Xc la matrice centrée des données (on retire pour chaque produit (ie chaque ligne) la valeur du produit moyen).

*Soit Sigma la matrice de variance covariance $Xc * Xc / nbProduits$*

D le vecteur des valeurs propres de Sigma en Python (en Matlab matrice diagonale)

V la matrice des vecteurs propres de Sigma

On trie D pour avoir les q premiers vecteurs propres.

*Soit W la matrice des q premiers vecteurs propres, normalisés, de taille nbStems*q*

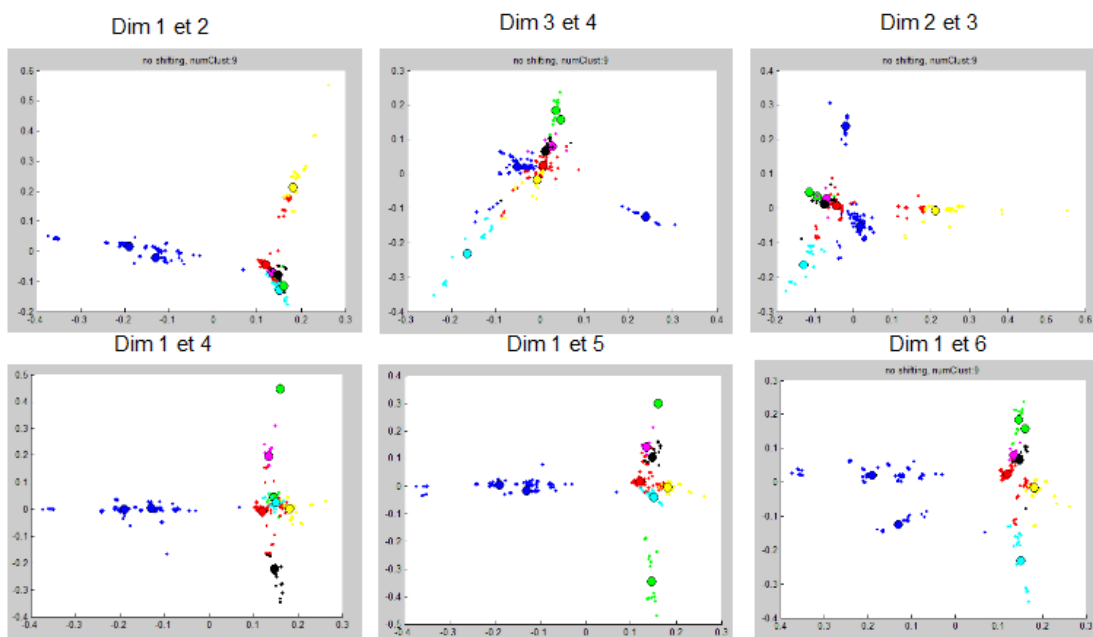
*Soit CP la matrice des q composantes principales, $CP = Xc * W$*

b. Meanshift

J'ai d'abord utilisé une implémentation en Matlab de Meanshift, avant de finalement utiliser une version Python intégrable au reste de mon programme, tirée de la bibliothèque sklearn.

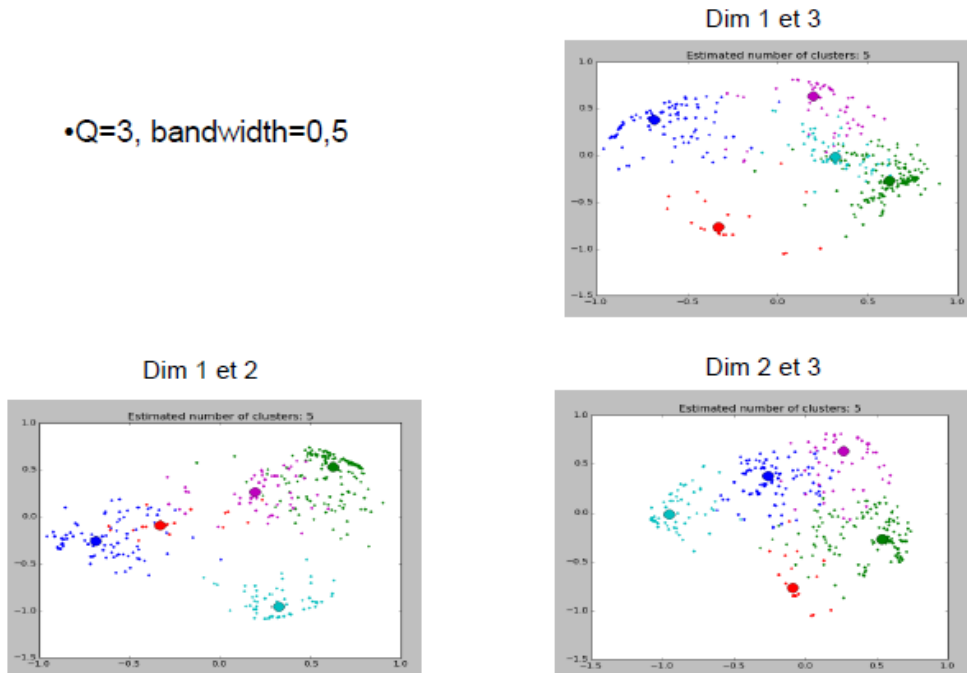
Dans le programme Meanshift je peux faire varier un paramètre, le « bandwidth », ce qui me donne plus ou moins de catégories. Je fais aussi varier le nombre de composantes principales utilisées, « q ».

ACP - Meanshift Matlab (q=6, bandwidth=0,2)



ACP-Meanshift version normalisée Python

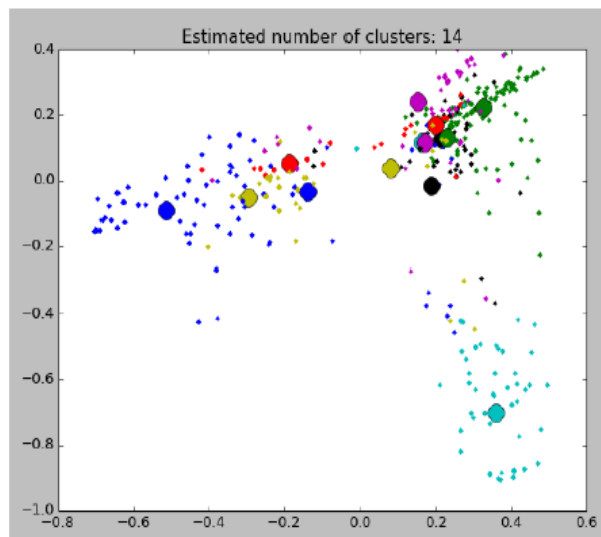
•Q=3, bandwidth=0,5



voici un exemple de catégories trouvées :

ACP-Meanshift (q=10, bandwidth=0,6)

- Catégories calculées.
- 0 : Dress
 - 1 : erreur, Necklace, Coat...
 - 2 : Full Length
 - 3 : Top
 - 4 : Blouse, silk
 - 5 : Wedding Dress
 - 6 : Tunic
 - 7 : Print, Wrap
 - 8 : Jumper
 - 9 : Jacket
 - 10 : Bolero
 - 11 : Sequin
 - 12 : Lace
 - 13 : ? (4 éléments)



c. Similarités par clusters

Les exemples suivants comparent l'algorithme avec nom + description et l'algorithme utilisant directement les clusters.

On observe des améliorations, comme ici où une robe portant le même nom que la veste a bien disparu des propositions.

Frequence (description + nom sans IDF)

Melinda Jacket

Similarité : 0.5265701284863964
Alice Jacket

Similarité description : 0.5664253212159911
Similarité nom : 0.4999999999999999
id produit : 374
(id produit d'origine : 203)

edg jacket
ey match complet
hook pair occas look
fast flat dress fit long slim

Similarité : 0.5224968388866248
Melinda Dress

Similarité description : 0.5562420972165621
Similarité nom : 0.4999999999999999
id produit : 172
(id produit d'origine : 203)

crush melind
match complet pair
jacket occas look dress
styl flat fit long

Similarité : 0.4121131379201689
Helena Jacket

Similarité description : 0.2803838468042237
Similarité nom : 0.4999999999999999
id produit : 165
(id produit d'origine : 203)

jacket tail match
complet pair occas
look styl flat dress fit long slim

Cluster

Melinda Jacket

Similarité : 0.5664253212159911
Alice Jacket

Similarité description : 0.5664253212159911
Similarité nom : 0.4999999999999999
id produit : 374
(id produit d'origine : 203)

edg jacket
ey match complet
hook pair occas look
fast flat dress fit long slim

Similarité : 0.2802828448004237
Helena Jacket

Similarité description : 0.2802828448004237
Similarité nom : 0.4999999999999999
id produit : 165
(id produit d'origine : 203)

jacket tail match
complet pair occas
look styl flat dress fit long slim

Similarité : 0.24519530075419899
Harriet Jacket

Similarité description : 0.24519530075419899
Similarité nom : 0.4999999999999999
id produit : 219
(id produit d'origine : 203)

jacket tail
complet pair look
fast fit long slim

En revanche, de nouveaux problèmes apparaissent en raison du manque de fiabilité des catégories calculées :

Frequence

Similarité : 0.6391605051854885

Stems communs :

trous pocket
straight leg cont back front

Similarité : 0.3913296706475304

Stems communs :

trous tab
pocket smart fast front back

Similarité : 0.34849992368520843

Stems communs :

trous pocket
straight leg cont back front

Cluster

Similarité : 0.4729012629637214

Stems communs :

trous pocket
straight leg cont back front

Similarité : 0.07881786718512776

Stems communs :

straight

Similarité : 0.07874445924912152

Stems communs :

straight fast waist

Dans l'exemple ci-dessus, des pantalons longs (« Full length Trousers ») ont été classés dans la même catégorie que les robes longues (« Full length Dress »). Les pantalons qui étaient proposés avant n'apparaissent plus car ils ont été classés dans une autre catégorie, et comme dans l'échantillon qu'on considère il y a peu de pantalons longs, on sera ensuite obligés de proposer des robes longues.

Frequence (description + nom sans IDF)

Triggys Alba Wrap

Similarité : 0.4107821319996045
Cecils Texture Wrap Top

Similarité description : 0.5939426281067818
Similarité nom : 0.28867513459481287
id produit : 571
(id produit d'origine : 55)

wrap top front
3_4 flat jersey fit long sleeve

Similarité : 0.36180282273022957
Effite Textured Wrap Top

Similarité description : 0.47149435493335473
Similarité nom : 0.28867513459481287
id produit : 613
(id produit d'origine : 55)

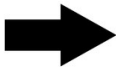
top wrap sid front 3_4
flat jersey fit long sleeve

Similarité : 0.3599416343222698
Polly Plain Wrap Top

Similarité description : 0.46684138391345514
Similarité nom : 0.28867513459481287
id produit : 238
(id produit d'origine : 55)

wrap top 3_4 flat
jersey fit long sleeve

Cluster



Similarité : 0.38582562621748885
Leaf Print Wrap Dress

Similarité description : 0.39582562621748885
Similarité nom : 0.28867519459481207
id produit : 753
(id produit d'origine : 55)

wrap ruch print sid
fit front 3_4 flat jersey long
sleeve



Similarité : 0.3630201709404584
Molly Wrap Dress

Similarité description : 0.3630201709404584
Similarité nom : 0.3333333333333333333
id produit : 52
(id produit d'origine : 55)

wrap print sid front 3_4
flat jersey fit long sleeve



Similarité : 0.33896487046741053
Nanette Print Lace Blouse

Similarité description : 0.33896487046741053
Similarité nom : 0
id produit : 621
(id produit d'origine : 55)

cent print top front flat
fit sleeve



Twiggly Alba Wrap

De même ici, le produit « Twiggly Alba Wrap » a été classé dans une catégorie à part par rapport aux produits similaires, car les autres contiennent le terme « top » et ont donc tous été classés dans la catégorie des hauts.

Au final, cette méthode utilisant directement les clusters serait meilleure que celle de l'augmentation car elle supprime des parasites, mais uniquement à la condition d'avoir de bonnes catégories. Les catégories que j'ai calculées durant ce stage ne sont pas assez bonnes pour faire bien fonctionner notre programme, mais la méthode serait utilisable par exemple si le site nous fournit les catégories.

Conclusion

Avant de réaliser ce stage j'avais des appréhensions car je n'avais jamais été confrontée au monde de l'entreprise auparavant. Cette première expérience fut très enrichissante, aussi bien sur le plan technique qu'humainement. J'ai eu l'occasion de travailler avec des technologies très variées, dont une partie furent une découverte. J'ai appris à utiliser Python et javascript, j'ai eu l'occasion de mettre en pratique mes connaissances du langage SQL, j'ai découvert la visualisation de graphes avec Gephi. J'ai aussi eu l'occasion de profiter de la partie relationnelle de l'entreprise, notamment lors d'une ballade en péniche organisée pour rencontrer les clients.

Pour conclure, ce stage me laissera de bons souvenirs, et m'a donné de quoi envisager plus clairement mon avenir professionnel.