Classifiez automatiquement des biens de consommation

Projet 6 : Parcours Data Scientist

Matthieu Gschwend

Présentation de la problématique

Mise en contexte : Vous êtes Data Scientist au sein de l'entreprise "Place de marché", qui souhaite lancer une marketplace e-commerce.

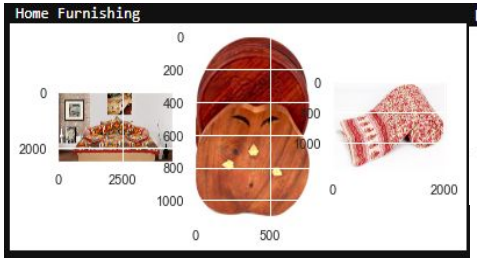
Pour l'instant, l'attribution de la catégorie d'un article est effectuée manuellement par les vendeurs, et est donc peu fiable. De plus, le volume des articles est pour l'instant très petit.
L'ajout de nouveaux produits par les vendeurs se doit d'être simplifié.

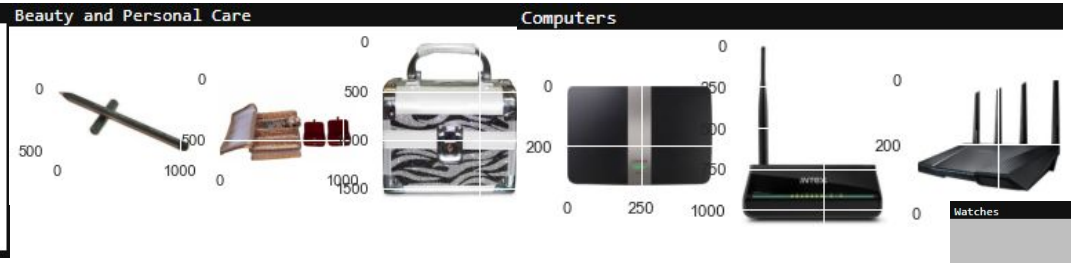
Objectif de la mission : Réaliser une première étude de faisabilité d'un moteur de classification d'articles, basé sur une image et une description, pour l'automatisation de l'attribution de la catégorie de l'article.

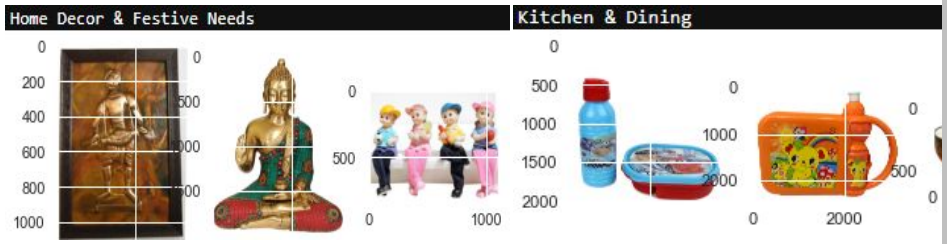
Sommaire

Description du jeu de données

- Le Jeu de données comporte 1050 produits.
- A chaque produit est associée une arborescence de catégories.
- Il existe 7 catégories principales, avec chacune 150 produits associés.
- Les images et descriptions associés à chaque produit.







Clustering selon la description
Traitement des données textuelles

Avant d'utiliser nos techniques de NLP il faut rendre les éléments de notre corpus exploitables.

Ceci passe par les étapes classiques suivantes :

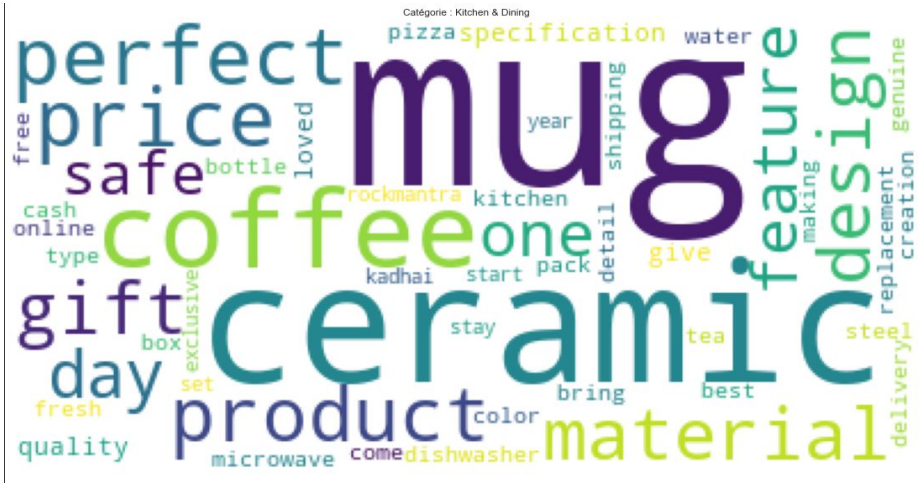
- Passer tous les mots en minuscule
- Supprimer les stops words (petits mot de liaisons ex “the”, “is” ex “and”)
- Supprimer les caractères numériques (parfois présents dans notre cas)
- Lemmatisation (les mots sont ramenés à leur base sémantique ex : suppression des pluriels, verbes ramenés à l'infinitif : “am,” “are,” “is”, “was”, “were” en "be")

Clustering selon la description

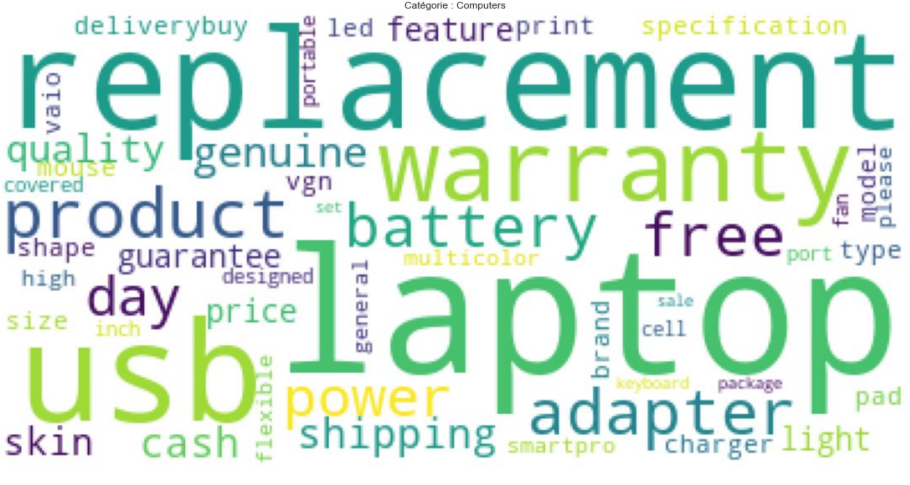
Traitement des données textuelles

Suite à ces traitement nous avons pouvons effectuer un cloud words selon les catégories

Kitchen and Dining



Computers



Clustering selon la description

Approche BoW : Extraire les features

Nous avons utilisé deux approches :

- CountVectorizer : Nombre d'occurrences de chaque mot du corpus dans une description.
- Tf*idf : Fréquence d'un mot dans la description coefficientée par l'inverse de la fréquence de ce mot dans le corpus des descriptions.

Démarche (générale) adoptée pour réaliser les clusterings :

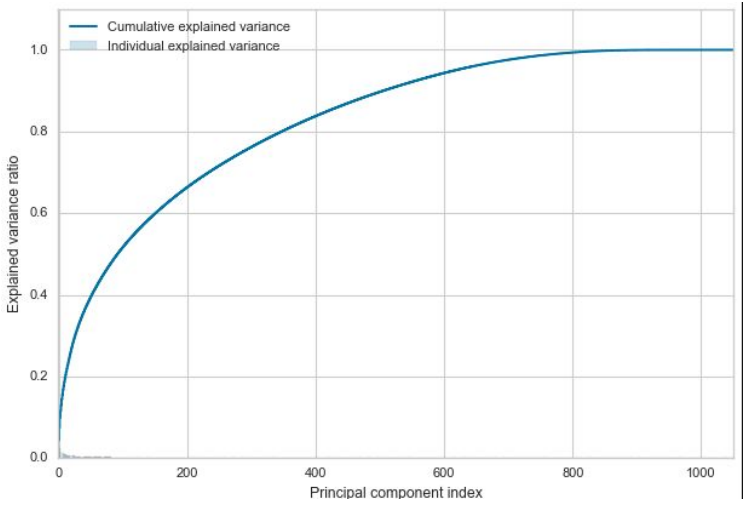
- Réduction dimensionnelle globale (PCA)
- Réduction dimensionnelle locale (TSNE)
- Algorithme de clustering Kmeans avec = 7
- Evaluation de la qualité des clusterings avec l'ARI

Clustering selon la description

Bag of Words : Détails des tests et résultats

Premières remarques : Il ne faut pas scaler les données, le passage par une TSNE est indispensable pour obtenir de bons résultats. La méthode tf*idf donne toujours de meilleurs résultats (même si faibles).

ACP sur les features tf*idf

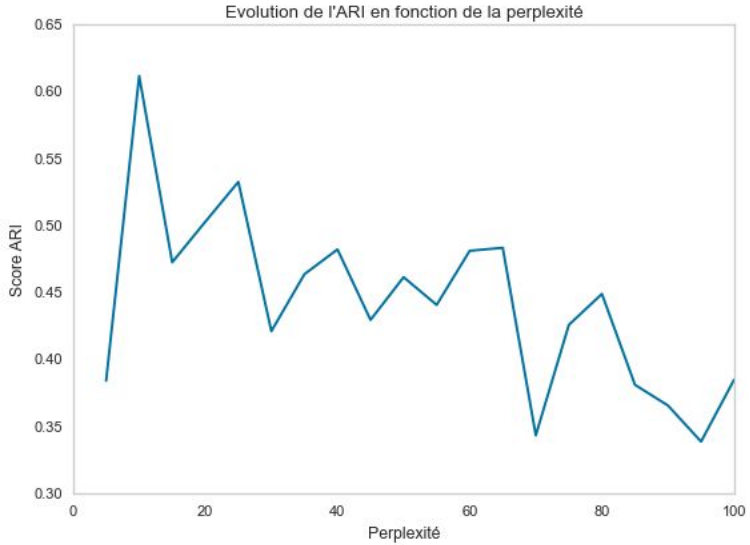


Nous plaçons un seuil de variance expliquée à 99% : le nombre de dimension passe alors de 1050 à 773

Clustering selon la description

Bag of Words : Détails des tests et résultats

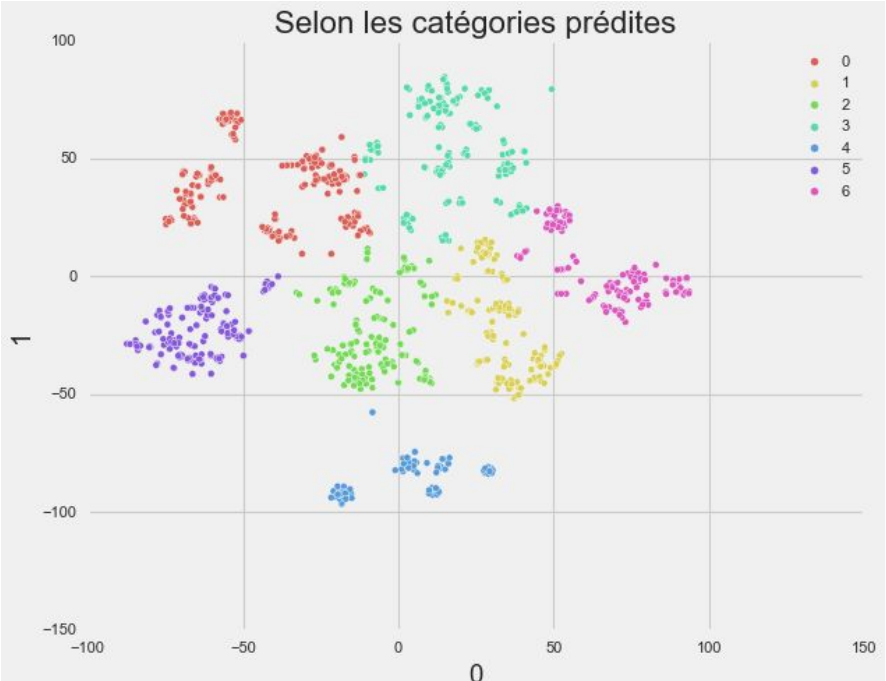
Avec une perplexité de 25 : Les score ARI sont plus élevés après l'utilisation de la PCA, les features provenant de tf*idf donnent les meilleurs résultats (0.45 avant PCA, 0.53 après)

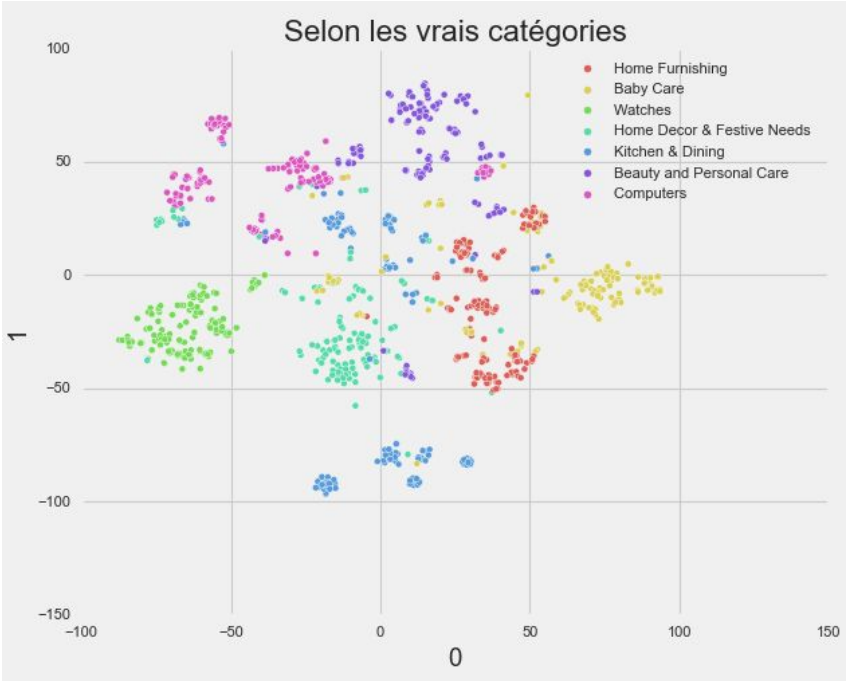


Nous avons fait évoluer la perplexité pour trouver celle qui donnait le meilleur résultat, ici 10 semble convenir parfaitement avec un ARI à 0.61

Clustering selon la description

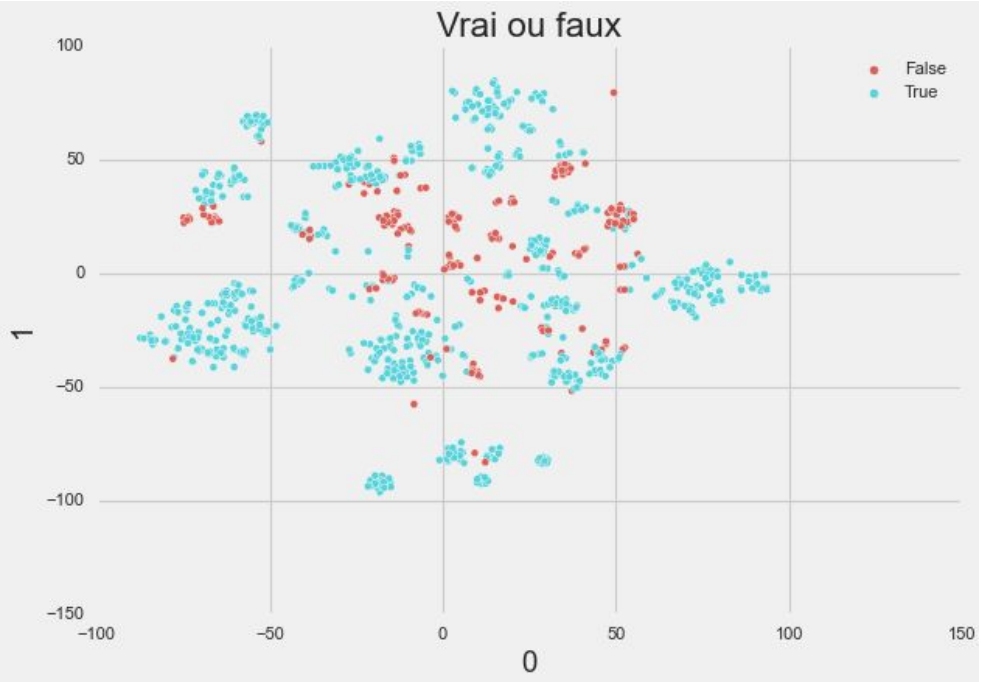
Bag of Words : Détails des tests et résultats





Clustering selon la description

Bag of Words : Détails des tests et résultats



	precision	recall	f1-score	support
0	0.78	0.69	0.73	150
1	0.73	0.87	0.80	150
2	0.71	0.91	0.80	150
3	0.72	0.81	0.76	150
4	0.84	0.83	0.83	150
5	0.95	0.49	0.65	150
6	0.99	1.00	1.00	150
accuracy			0.80	1050
macro avg	0.82	0.80	0.80	1050
weighted avg	0.82	0.80	0.80	1050

Clustering selon la description
Word embedding

Nous avons testé les performances de 3 méthodes de sentence embedding à l'aide des modèles présélectionnés par Linda.

Ces méthodes permettent de représenter une phrase ou un mot par un vecteur de réel en conservant les relations sémantiques entre les mots/phrases

Score ARI suite aux mêmes étapes que la partie précédente :

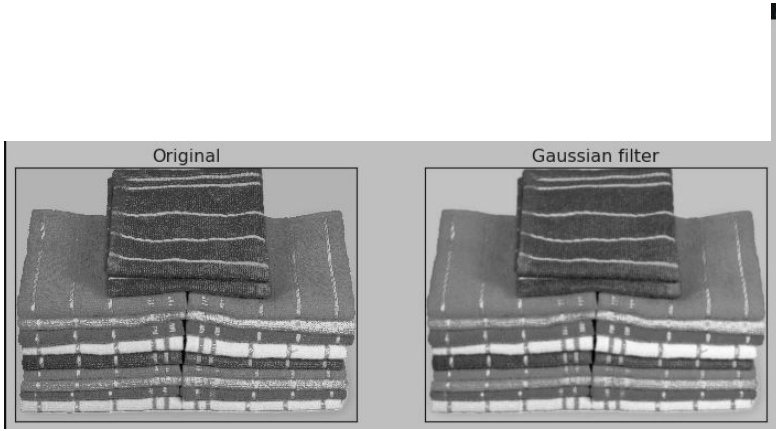
- Word2Vect : ARI 0.31
- Universal Sentence Encoder : ARI 0.46
- BERT : ARI 0.20

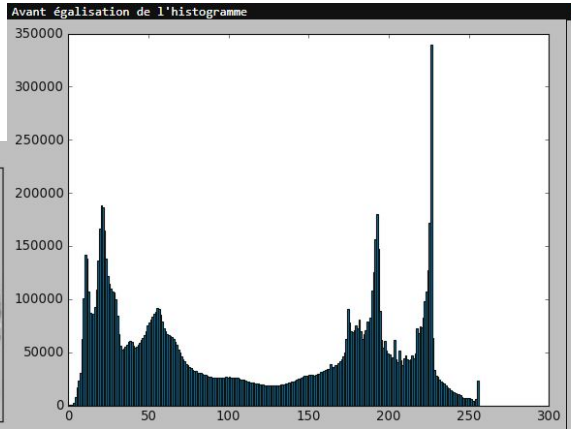
Nos tentatives n'ont malheureusement pas obtenues de meilleurs ARI qu'avec la méthode tfidf.

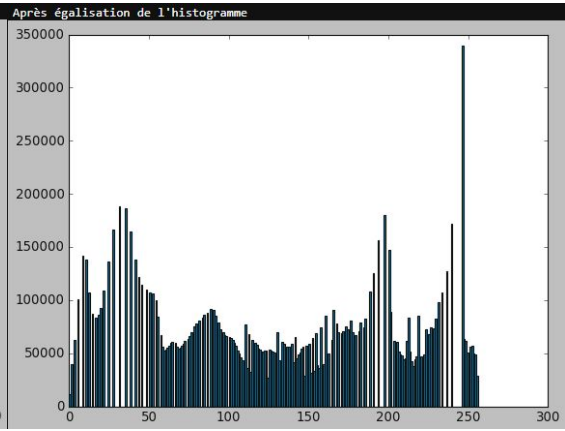
Clustering des images
Traiter les images

Pour améliorer les résultats des clustering nous avons traité les images de la manière suivante :

- Passage en noir et blanc
- Filtrage de l'image
- Égalisation de l'histogramme





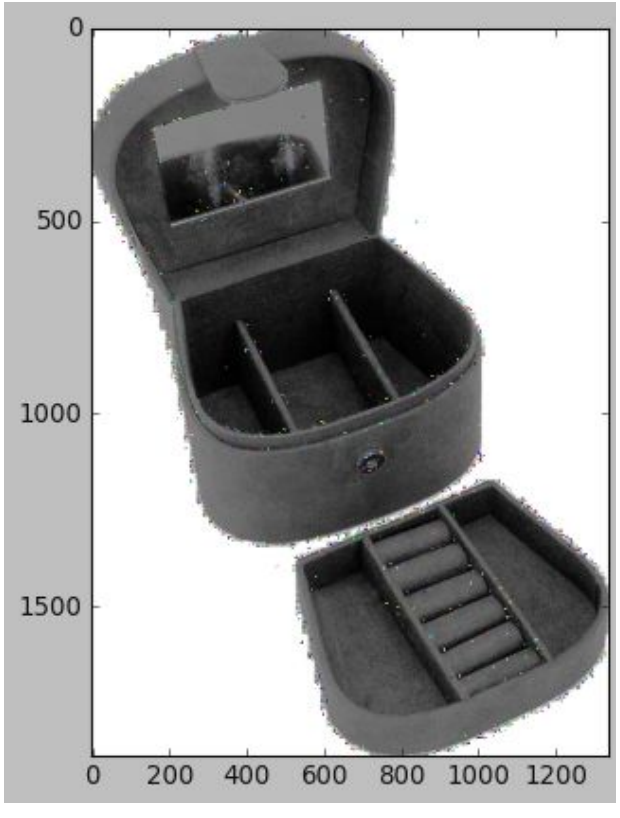


Clustering des images
Utilisation de SIFT

Cette méthode permet de détecter les points d'intérêt de l'image (coins, bords).

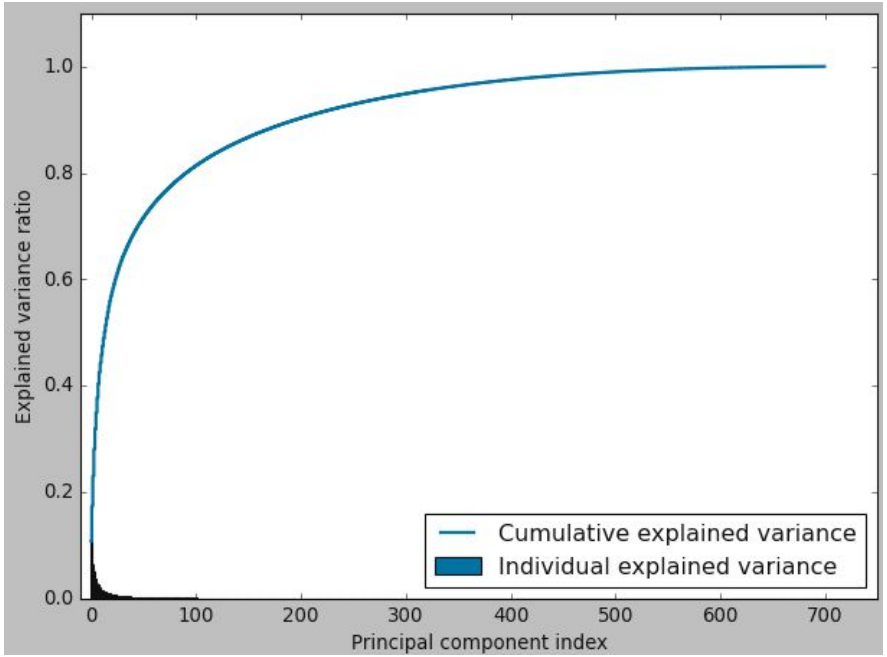
Étapes :

- Pour chaque image trouver les points d'intérêt (limités à 500)
- Faire un clustering de ces points d'intérêt (association de 483069 descripteurs à 700 clusters)
- Création d'un histogramme



Clustering des images

Utilisation de SIFT : Résultats



Une fois l'histogramme créé, le processus est similaire que précédemment :

- Utiliser une ACP
- Effectuer une TSNE
- Utiliser un algorithme KMeans
- Calculer l'ARI

Résultats : Avec un score ARI à seulement 0.023 nous n'avons pas tenté de l'améliorer (changer le nombre de descripteur par image, nombre de cluster pour l'histogramme)

Clustering des images
Transfert learning

Afin d'extraire les features des images nous nous tournons alors vers le modèle VGG16.
C'est un réseau de neurones à convolution créé en 2014 par Google permettant de classifier automatiquement des images selon 1000 catégories.

Input

VGG16

Output

```
coffee_mug (90.45%)
```

Clustering des images

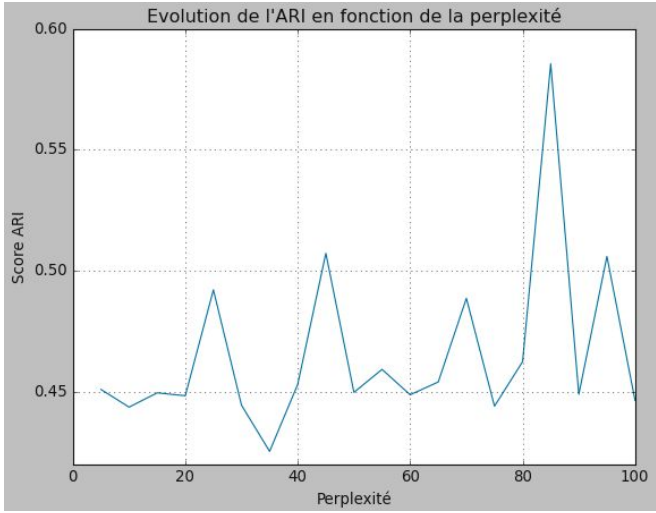
Transfert learning : Mise en pratique

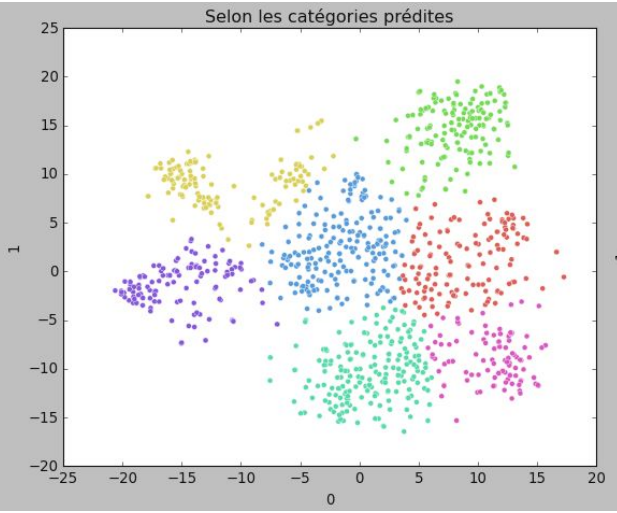
Ce modèle pourrait nous servir de 3 façon différentes :

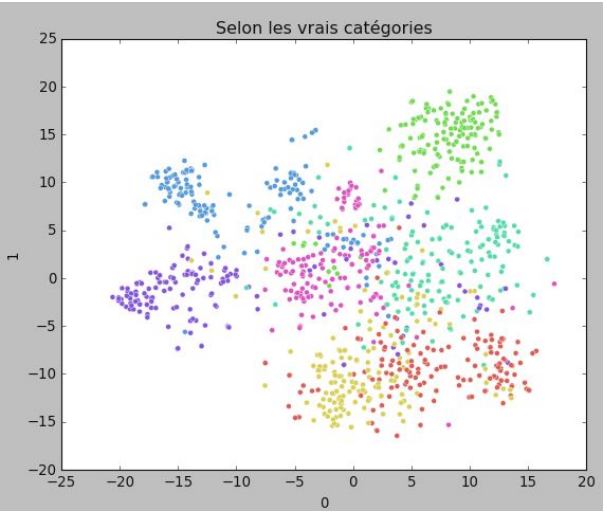
1. Utiliser la prédiction de l'image, puis faire un word embedding de la prédiction, comparer ensuite la distance entre ce mot et les catégories.
2. Utiliser les poids associés au modèle pour effectuer un nouvel entraînement spécifique à nos images / catégories.
3. Utiliser la sortie de l'avant dernière couche de neurone pour nous créer des features.

Clustering des images

Transfert learning : Résultats








```
ARI :  0.5855
```

Clustering des images et des descriptions

- L'idée qui vient naturellement est d'utiliser à la fois les features de tf*idf et de VGG16 pour faire les prédictions. Mais suite à la concaténation de ces features nous obtenons seulement un score ARI de 0.53.
- Une autre solution serait d'utiliser indépendamment les deux clusterings pour faire des prédictions, en cas de désaccord choisir le modèle ayant la plus haute précision (le moins de faux positif) dans les classes prédites.

Features tf*idf

	precision	recall	f1-score	support
0	0.78	0.69	0.73	150
1	0.73	0.87	0.80	150
2	0.71	0.91	0.80	150
3	0.72	0.81	0.76	150
4	0.84	0.83	0.83	150
5	0.95	0.49	0.65	150
6	0.99	1.00	1.00	150
accuracy			0.80	1050
macro avg	0.82	0.80	0.80	1050
weighted avg	0.82	0.80	0.80	1050

Features VGG16

	precision	recall	f1-score	support
0	0.58	0.74	0.65	150
1	0.94	0.79	0.86	150
2	0.64	0.91	0.75	150
3	0.82	0.75	0.78	150
4	0.82	0.55	0.66	150
5	0.94	0.81	0.87	150
6	0.93	0.94	0.94	150
accuracy			0.78	1050
macro avg	0.81	0.78	0.79	1050
weighted avg	0.81	0.78	0.79	1050

Conclusion et ouverture

Notre travail consistait à faire une étude de faisabilité d'un moteur de classification à l'aide de la description et des images des produits.
L'implémentation de plusieurs méthodes à mis en évidence que les features provenant de tfidf (pour le texte) et du transfert learning VGG16 (pour les images) étaient les meilleurs candidats, donnant des résultats pertinents.

Ouvertures :

- Tester d'autres méthodes de word/sentence embedding
- Tester la méthode avec VGG16 de prédiction directe de la catégorie de l'image, puis distance entre ce mot et le nom des catégories après embedding
- Utiliser des méthodes d'apprentissages supervisées classiques