

# “Classifiez automatiquement des biens de consommation”

Projet n°6

Léa ZADIKIAN  
Parcours Data Scientist  
21 Mars 2023

# Classifiez automatiquement des biens de consommation

- I. Présentation de la problématique et du jeu de données**
- II. Présentation des différentes approches de modélisation et de leurs résultats**
- III. Conclusion sur la faisabilité du moteur de classification**



# I. Présentation de la problématique et du jeu de données



# Environnement technique

- Notebook Jupyter 6.4.8
- Python 3.9.12
- Librairies utilisées :
  - Pandas, Numpy
  - Matplotlib, Seaborn, Plotly
  - OpenCV
  - Scikit-Learn
  - NLTK
  - TensorFlow

# Classifier les produits de “Place de marché”

- L'entreprise “**Place de marché**” souhaite lancer une marketplace e-commerce.
- **Sa problématique** : actuellement, l'attribution de la catégorie du produit est effectuée manuellement par le vendeur, et est donc peu fiable.
- **Son objectif** : pour fiabiliser cette catégorisation des produits et améliorer l'expérience utilisateur vendeur et acheteur, “Place de marché” souhaite automatiser cette tâche d'attribution de la catégorie à un article.



## → Notre mission :

Réaliser une **étude de faisabilité d'un moteur de classification pour l'automatisation de l'attribution de la catégorie** de l'article, en se basant sur la description et l'image de l'article.

# Le jeu de données “Place de marché”

**1 fichier .csv :**

- **1.050 lignes**, chaque ligne est un article.
- **15 colonnes**, avec informations sur le produit :
  - Identifiant unique du produit
  - **Nom du produit**
  - Marque du produit
  - URL du produit
  - **Arborescence du produit ( ⇒7 catégories)**
  - Prix / prix remisé
  - **Description du produit**
  - Nom de l'image associée au produit
  - ...
- Pas de produits en doublon. Taux de remplissage des données 98%, pas de valeurs manquantes sur les colonnes utiles à notre mission.

**1.050 articles vendus sur “Place de marché”, répartis équitablement en 7 catégories (150 produits par catégorie) :**

- Home Furnishing
- Baby Care
- Watches
- Home Decor & Festive Needs
- Kitchen & Dining
- Beauty and Personal Care
- Computers

**1 répertoire de 1050 images, une image de chaque produit :**



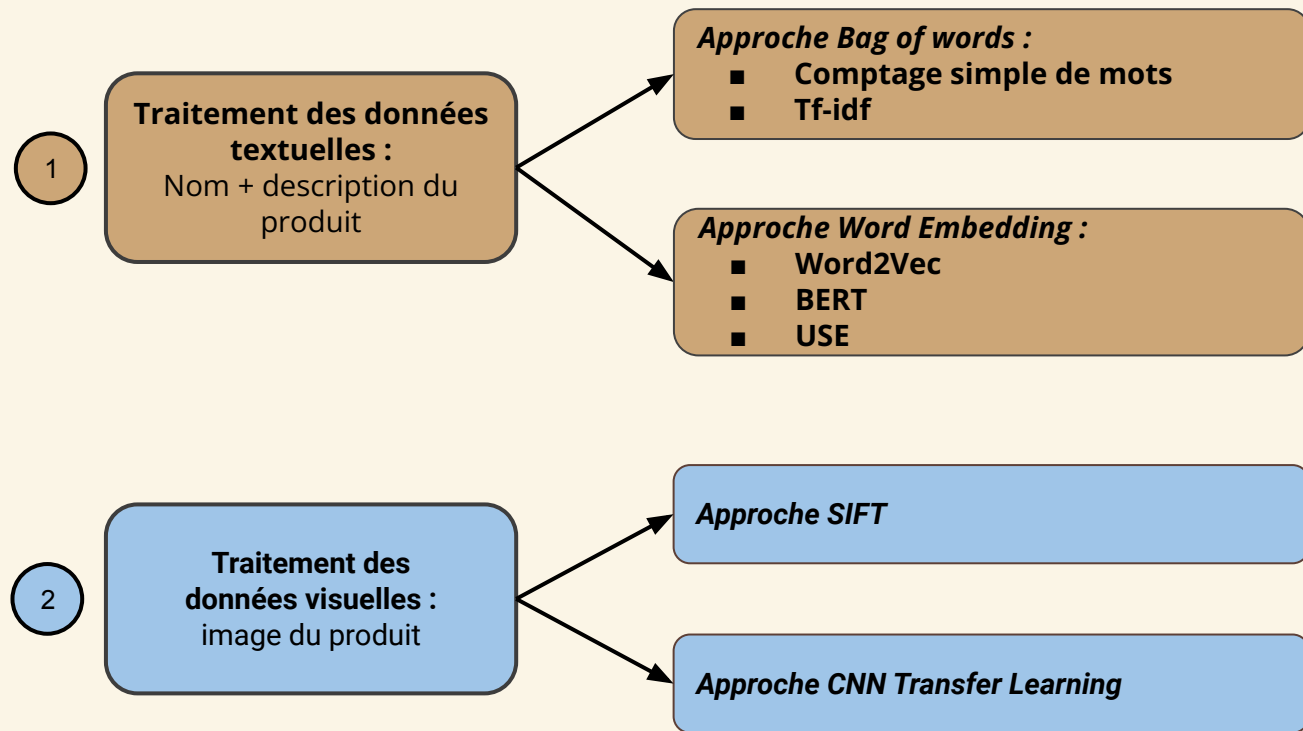
**Nuage de mots sur le nom et la description des produits.**



## II. Présentation des différentes approches de modélisation et de leurs résultats



# Les différentes approches de modélisation





# Etude de faisabilité : démarche générale

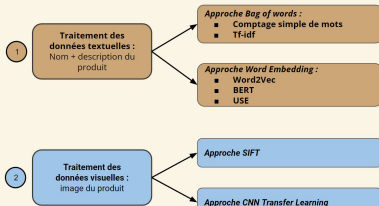
## Pré-traitement

**Données texte** : tokenization, stopwords, lemmatisation,...

**Image** : passage en niveau de gris, égalisation d'histogramme,...

## Feature engineering

Différentes approches d'extraction des features texte ou image :



## Réduction de dimension

ACP, puis t-SNE

### ACP

- Conservation de 99% de la variance expliquée.
- Réduction de dimensions et création de features décorréliées entre elles :
  - Meilleure séparation des données et réduction du temps de traitement pour le t-SNE.

**t-SNE** : Réduction de dimensions à 2 composantes, permettant l'affichage en 2D.

## Clustering par K-Means

- Création de clusters théoriques par K-Means.
- Nombre de clusters = nombre de catégories (k=7)

## Analyse visuelle

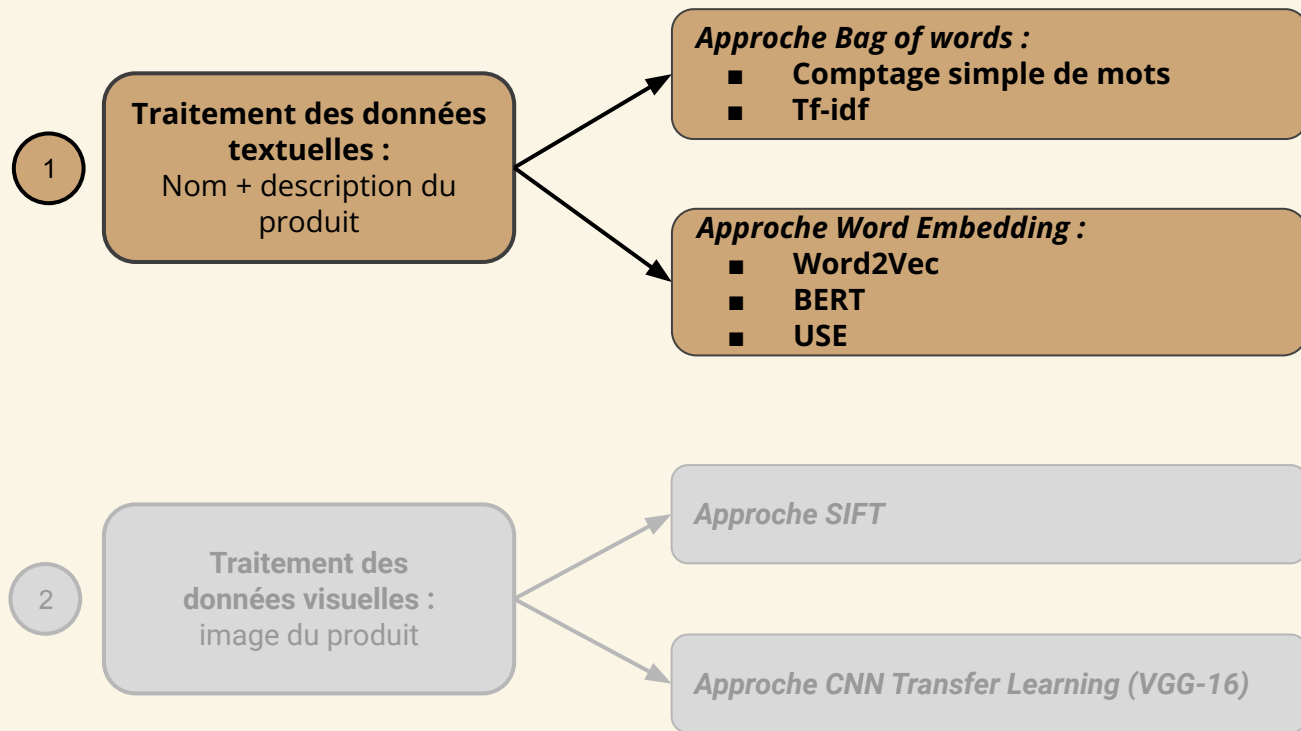
- Affichage des données du t-SNE selon les vraies catégories et selon les clusters.
- L'analyse visuelle permet de conclure sur la possibilité de séparation des produits par catégories et donc sur la faisabilité d'une classification auto.

## Analyse mesure

- Calcul du score ARI, mesure de similarité entre catégories réelles et clusters, pour conforter l'analyse visuelle.
- Analyse par classe.



# Les différentes approches de modélisation



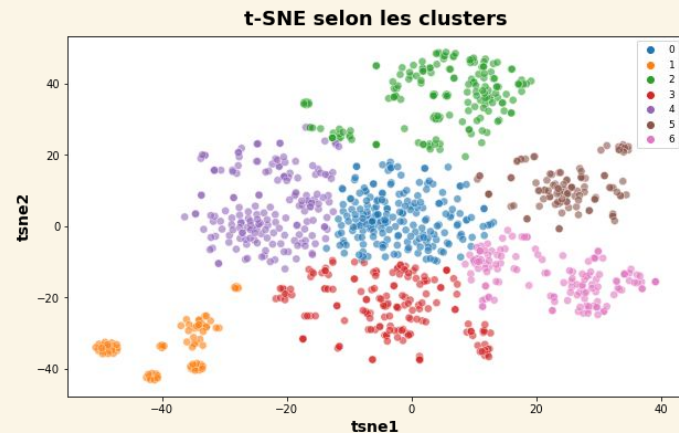
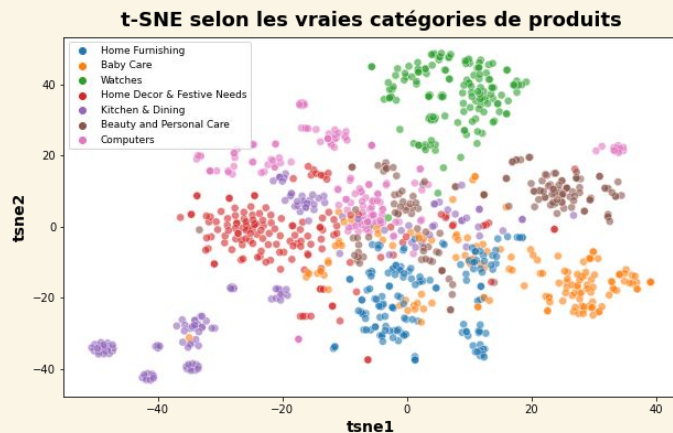
# Pré-traitements des données textuelles

Librairie NLTK

Opérations de pré-traitement	Approche Bag of words		Approche word embedding		
	Comptage simple	Tf-idf	Word2Vec	BERT	USE
1. <b>Passage en minuscule</b>	✓	✓	✓	✓	✓
2. <b>Tokenisation</b> : découpage de la chaîne de caractère en mots.	✓	✓	✓	✓	✓
3. <b>Suppression des stopwords</b> : suppression des "petits mots" peu informatifs et de la ponctuation.	✓	✓	✓	✗	✗
4. <b>Lemmatisation</b> : forme canonique (infinitif, masculin singulier...).	✓	✓	✓	✗	✗

# Approche Bag of words : comptage simple

- Feature texte : **matrice documents - termes** avec le nombre d'occurrences de chaque terme dans chaque document (*countVectorizer*).
- **Dimensions avant réduction ACP** : (1050, 5420)
- **Dimensions après réduction ACP** : (1050, 758)

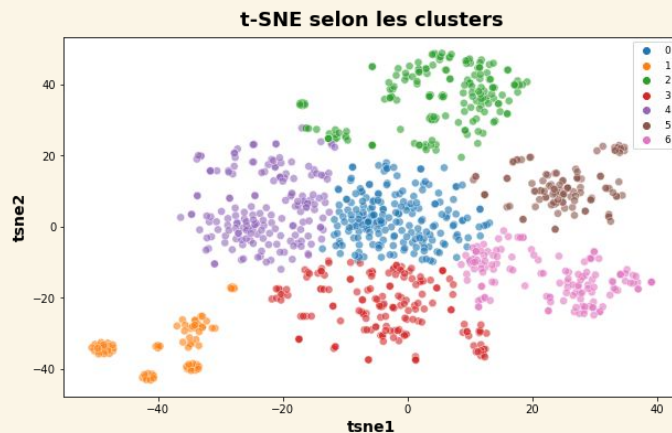
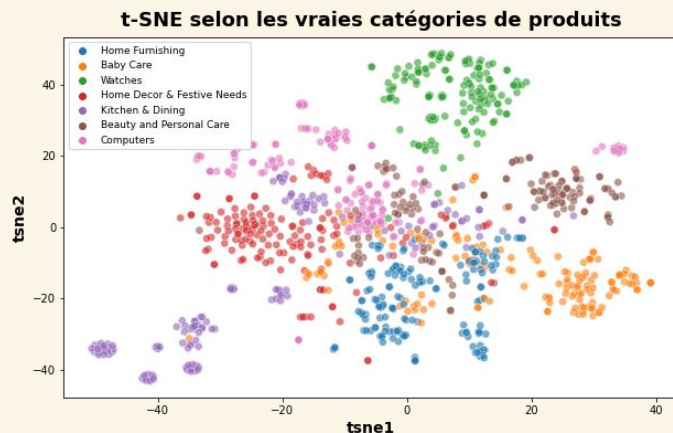


Score ARI = 0.4287

⇒ Le Bag of words comptage simple permet une assez bonne séparation des catégories.

# Approche Bag of words : Tf-idf (Term-Frequency Inverse Document Frequency)

- Feature texte : **matrice documents - termes** avec le score tf-idf de chaque terme dans chaque document (*TfidfVectorizer*).
- **Dimensions avant réduction ACP** : (1050, 5420)
- **Dimensions après réduction ACP** : (1050, 899)



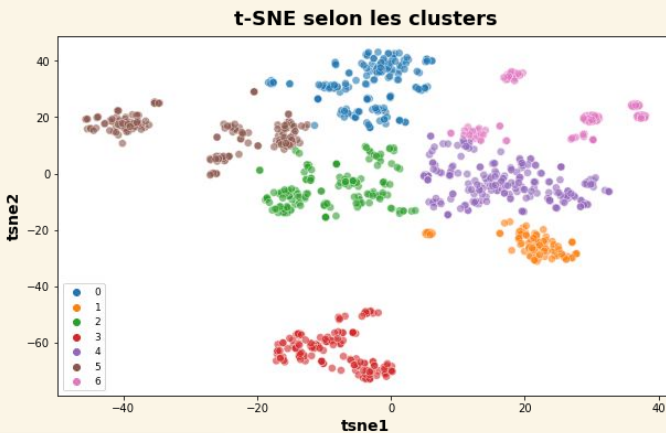
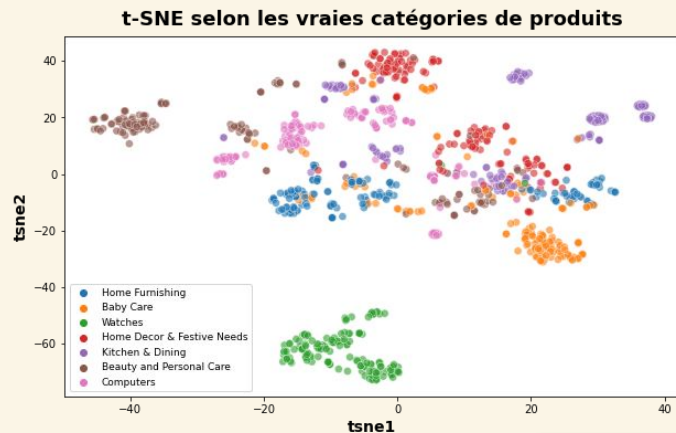
Score ARI = 0.5466

⇒ Le Bag of words Tf-idf permet une assez bonne séparation des catégories.

# Approche word embedding : Word2Vec

TensorFlow / Keras

- **Word2Vec :**
  - Représentation des mots à l'aide de **vecteurs de réels denses**.
  - Des mots proches en termes de sens auront une représentation vectorielle proche.
  - Basé sur un **réseau de neurones peu profond** : 1 couche intermédiaire (souvent 300 neurones), dont les poids forment la représentation vectorielle du mot.
- **Dimensions des features texte** : (1050, 300)



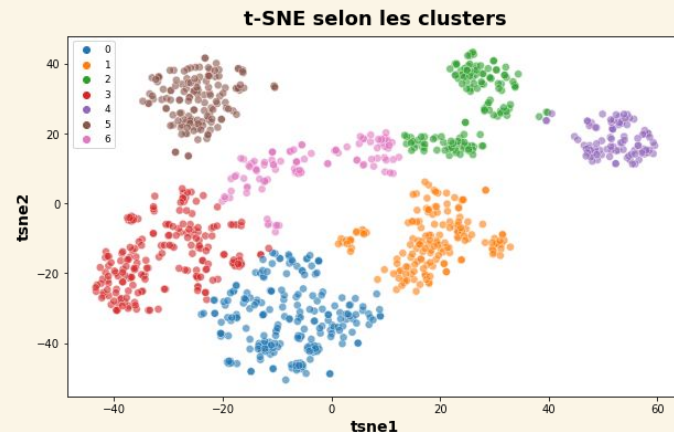
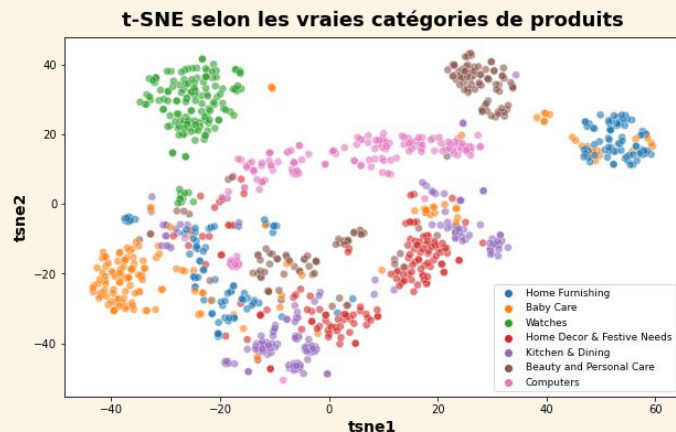
Score ARI = 0,3759

⇒ Word2Vec permet une séparation partielle des catégories.

# Approche word embedding : BERT (Bidirectional Encoder Representations from Transformers)

- **BERT :**
  - Fournit une **représentation vectorielle dense du langage**, en utilisant un réseau de neurones profond pré-entraîné sur l'architecture Transformers
  - Est **bidirectionnel** (prend en compte les mots précédents et suivants dans une phrase ), ce qui permet de mieux capturer le contexte et la sémantique des mots dans une phrase.
- **Dimensions des features texte :** (1050,765)

TensorFlow / Keras



Score ARI = 0,3588

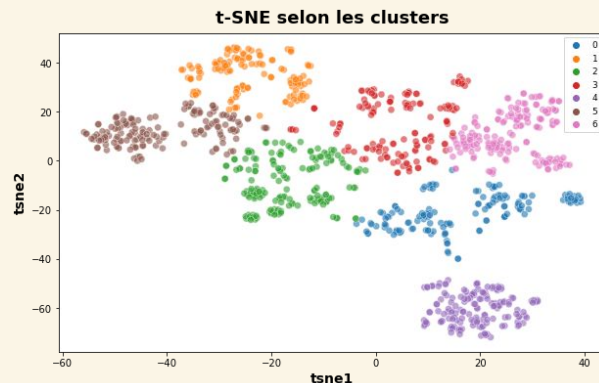
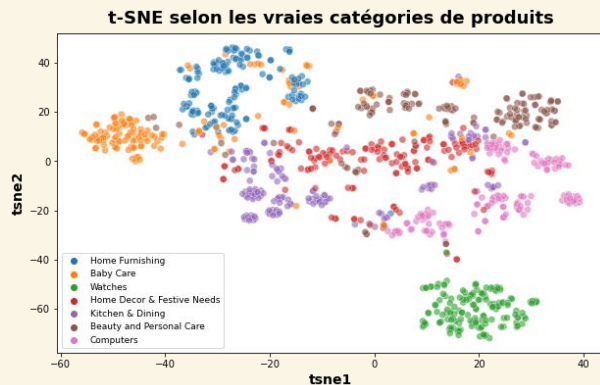
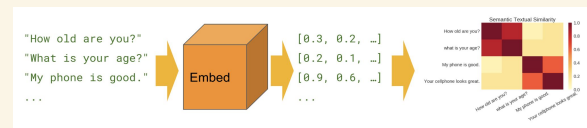
⇒ BERT permet une séparation partielle des catégories.

# Approche word embedding : USE (Universal sentence encoder)

## ■ USE :

- Fournit un encodage de mots ou phrases en vecteurs de haute dimension, utilisant un réseau de neurones profonds pré-entraîné sur une très grande variété de données.
- Dans cet espace, les phrases sémantiquement similaires sont proches, permettant ainsi de mesurer la similarité sémantique entre des phrases.

## ■ Dimensions des features texte : (1050, 512)

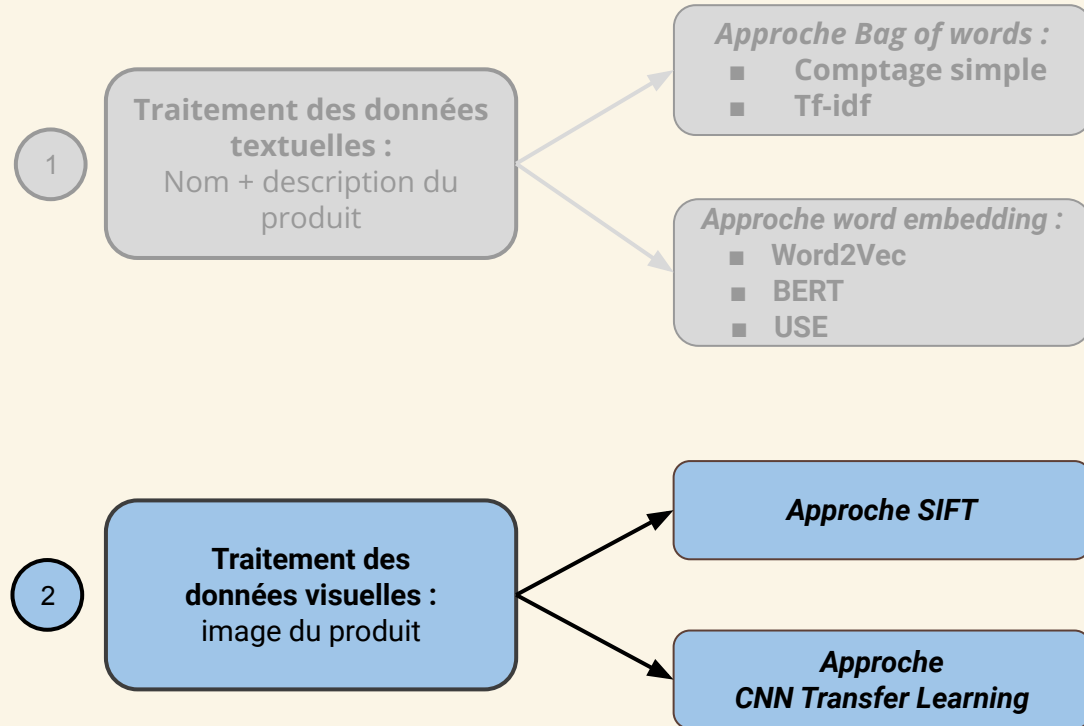


Score ARI = 0,4613

⇒ USE permet une assez bonne séparation des catégories.

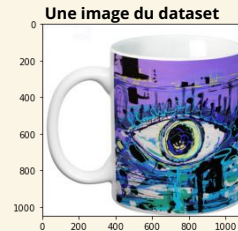


# Les différentes approches de modélisation



# Approche SIFT (*Scale-Invariant Feature Transform*)

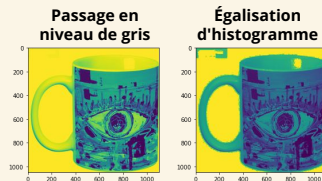
- **SIFT** : algorithme du domaine de la vision par ordinateur de reconnaissance de caractéristiques (feature detection). Il permet de **détecter et d'extraire des descripteurs de points clés dans une image** (bords, contours et points d'intérêt), qui sont invariant aux variations l'échelle et à la rotation, et robuste à l'illumination.



OpenCV

## 1. Pré-traitement des images

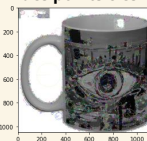
- Passage en niveaux de gris.
- Égalisation d'histogramme.



## 2. Calcul des descripteurs

- Taille d'un descripteur = 128
- Nombre de descripteurs = 517.351
- Temps = 360s

## Affichage des points clés



## 3. Clustering des descripteurs

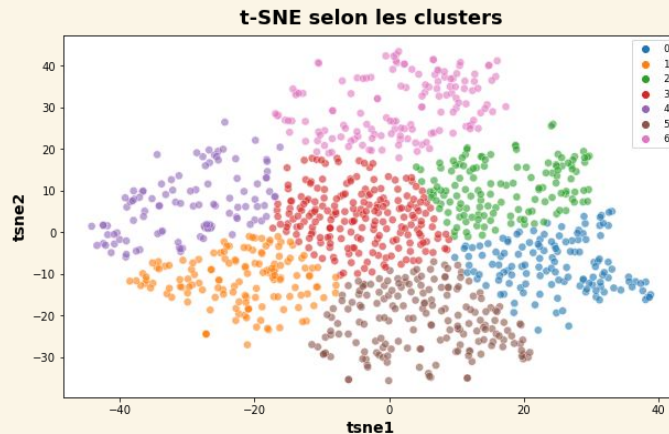
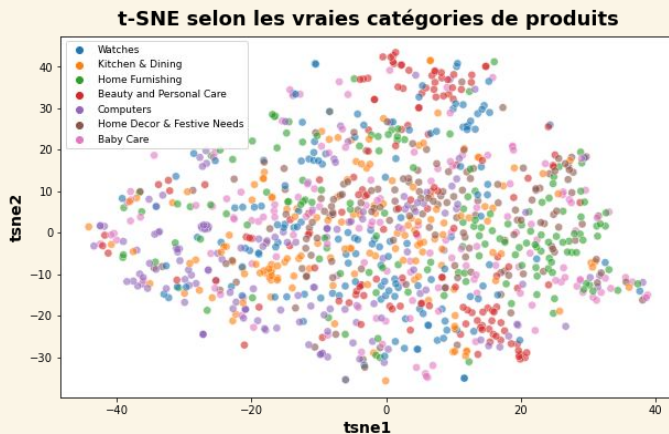
- Algo Mini-Batch K-Means (plus rapide que K-Means pour données volumineuses)
- $k = \sqrt{\text{nb de descripteurs}} = 719$  clusters
- Temps = 9s

## 4. Calcul des features image

- Chaque descripteur est associé à son cluster.
- Comptage par image du nombre de descripteurs associés à un cluster : histogramme par image.
- Dimensions des features : 1050\*719
- Temps = 67s

# Approche SIFT (*Scale-Invariant Feature Transform*)

- Dimension des features images avant réduction ACP : (1050, 719)
- Dimension des features images après réduction ACP : (1050, 498)

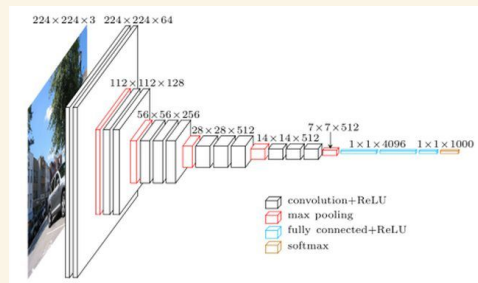


Score ARI = 0,0557

⇒ L'algorithme SIFT ne permet pas la séparation des catégories.

# Approche CNN Transfer Learning avec VGG-16

- **CNN (Convolutional Neural Network)**, réseau de neurones spécialement conçu pour traiter les images en entrée :
  - **Détection et description automatique des features images.**
  - Entraînement d'un classifieur : l'erreur de classification est minimisée afin d'optimiser les paramètres du classifieur et les features.
  - Calcul, à partir de l'entrée, une probabilité pour chaque classe. La classe attribuée à l'objet en entrée correspond à celle de score le plus élevé.
- **VGG-16** : modèle composé de 16 couches et entraîné sur l'ensemble des données ImageNet (plus de 1 million d'images réparties en 1000 classes).
- **Transfer learning** : au lieu de créer un modèle de zéro, utilisation d'un modèle pré-entraîné sur une grande quantité de données pour initialiser un nouveau modèle. Ainsi, on peut réentraîner le modèle sur une tâche spécifique et souvent avec moins de données, ce qui peut accélérer le processus d'apprentissage et améliorer les performances du modèle.

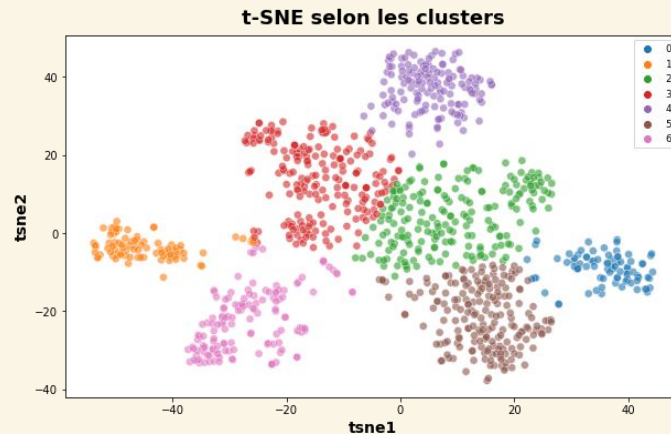
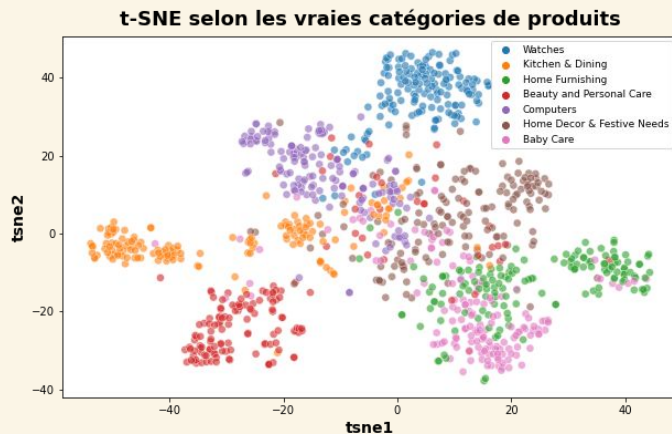


Représentation 3D de l'architecture de VGG-16

# CNN Transfer Learning avec VGG-16 : Résultats

- Création du modèle, chargement et prétraitement des images, création des features images.
- Taille des images en entrée de VGG-16 : (1 x 224 x 224 x 3 )
- Dimension des features images : (1050, 4096)
- Dimension après réduction ACP : (1050, 803)

Keras

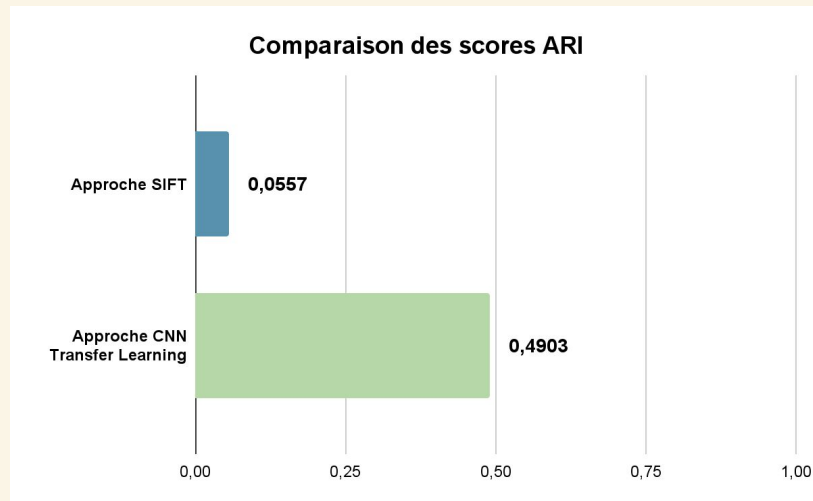


Score ARI = 0,4903

⇒ VGG-16 permet une assez bonne séparation des catégories

# Traitement des données visuelles : Comparaison

- SIFT utilise un processus de traitement d'images pour extraire les caractéristiques de l'image. Les CNN apprennent automatiquement à extraire les caractéristiques image.
- Les CNN ont tendance à donner des **résultats plus précis** que SIFT.
- Les CNN sont en général **plus rapides** que le SIFT.



Temps de calcul des features images	
SIFT	VGG-16
436 secondes	67 secondes



### III. Conclusion sur la faisabilité du moteur de classification

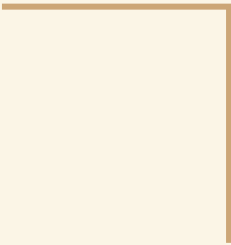


# Faisabilité du moteur de classification

- L'analyse graphique montre visuellement qu'il est **réalisable de séparer automatiquement les produits selon leurs vraies catégories**, à l'aide de leur nom et description, et des images.
- **Le calcul du score ARI** a permis de confirmer l'analyse visuelle.
- Cela démontre donc la **faisabilité de réaliser ultérieurement une classification** pour déterminer automatiquement les images.
- Cette étude de faisabilité représente **la première étape d'une démarche agile** d'un projet data.
- L'étape suivante serait de réaliser une **classification supervisée** avec un nombre plus important de produits, avec une recherche du modèle le plus performant et de **déployer une API** pour rendre disponible le modèle de classification.







Merci pour votre attention !  
Des questions ?

