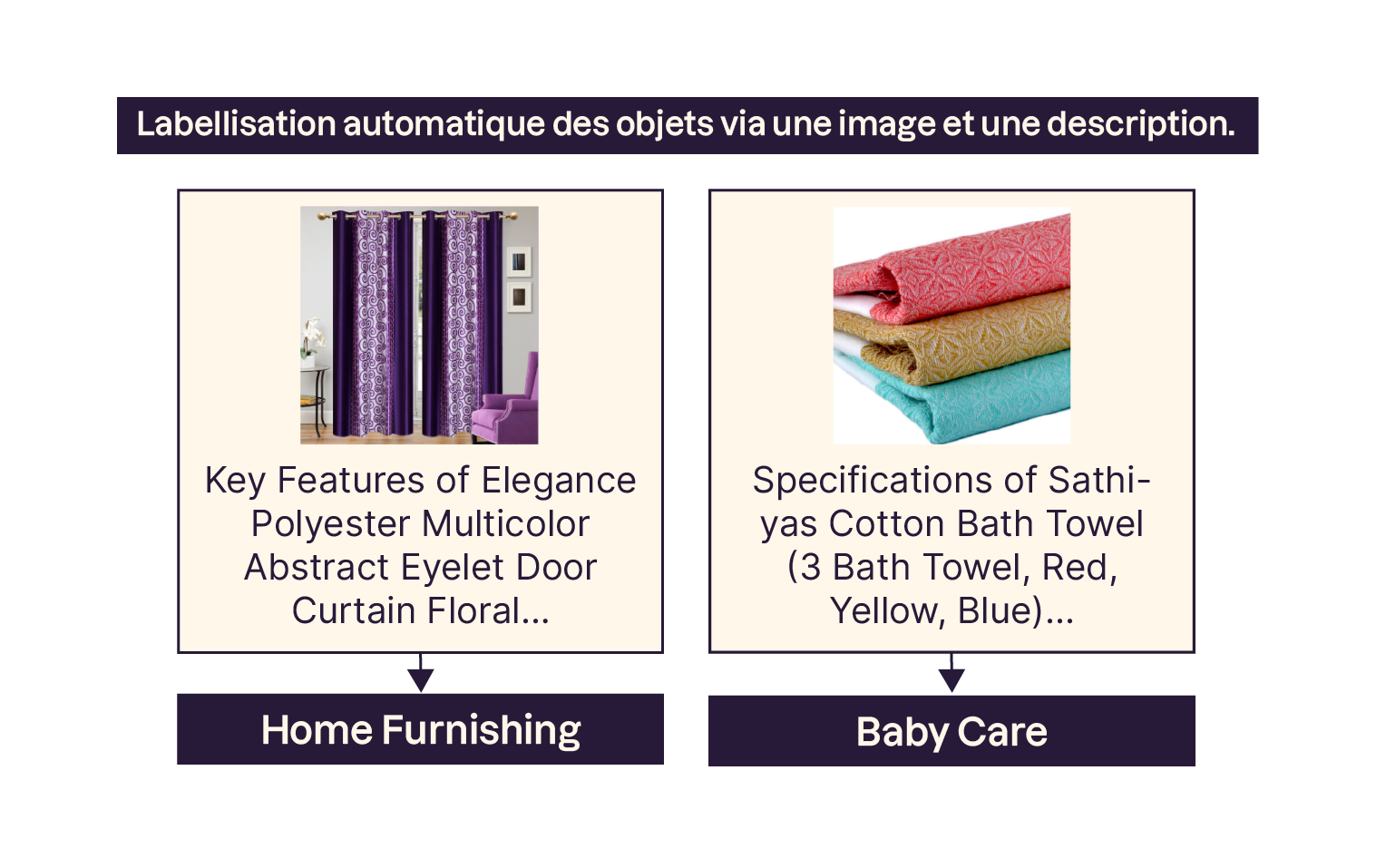
Note méthodologique : preuve de concept

## **I) Dataset retenu**

Sur la marketplace e-commerce de Place de marché, les vendeurs proposent des articles en postant une photo et une description. L'attribution de la catégorie d'un article est effectuée manuellement par les vendeurs. Pour faciliter la mise en ligne de nouveaux articles et la recherche de produits, il est nécessaire d'automatiser cette tâche d'attribution de catégorie.

Le dataset contient 1050 lignes et 15 colonnes, le même nombre d’images et des noms correspondants. On distingue 7 catégories de produits, soit 150 produits par catégorie.



Sur ce projet, j’ai réalisé une étude de faisabilité d’un moteur de classification automatique d’articles à partir de leur image et de leur description. Pour cela j’ai procédé à une extraction de features avec :

* BERT et USE sur les descriptions
* Une méthode de CNN/Transfer learning sur les données image en m’appuyant sur le modèle VGG16

J’ai ensuite effectué des prédictions à partir de ces features en entraînant un t-SNE (t-Distributed Stochastic Neighbor Embedding) pour réduire la dimensionnalité, puis un Kmeans, et en calculant l’ARI (Adjusted Rand Index) pour mesurer la similarité entre les clusters réels et les clusters prédits.

## **II) Les concepts de l’algorithme récent**

BERT et USE ont été publiés en 2018, le VGG16 en 2014. De nouveaux modèles ont émergés depuis, certains plus précis mais plus lourds et plus lents, et d’autres plus légers et rapides au prix d’une légère baisse de précision.

Pour ce projet, j'ai décidé de me focaliser sur une variante de BERT : RoBERTa (2019). BERT et RoBERTa existent en version de base et version large, alors j'ai testé les 2.

BERT (Bidirectional Encoder Representations from Transformers) :

BERT, développé par Google en 2018, est un modèle de langage basé sur l'architecture Transformer. Il se distingue par son apprentissage bidirectionnel (ou profondément masqué), qui permet de capturer le contexte des mots en tenant compte des mots qui les précèdent et les suivent. BERT est pré-entraîné avec deux tâches principales : Masked Language Modeling (MLM), où certains mots d’une phrase sont masqués et le modèle doit les prédire, et Next Sentence Prediction (NSP), qui consiste à déterminer si deux phrases apparaissent dans une séquence logique. BERT a révolutionné le traitement du langage naturel (NLP) en améliorant significativement les performances sur de nombreuses tâches comme la classification de texte, la réponse aux questions et la reconnaissance d'entités nommées.

RoBERTa (Robustly Optimized BERT Approach) :

RoBERTa, développé par Facebook AI en 2019, est une amélioration de BERT qui optimise son entraînement. Contrairement à BERT, RoBERTa supprime la tâche Next Sentence Prediction (NSP), qui s’est révélée non essentielle, et se concentre sur un plus grand volume de données et un apprentissage plus long. Il utilise également une dynamisation du masquage dans la tâche MLM, ce qui signifie que les mots masqués changent au fil des époques d'entraînement, rendant le modèle plus robuste. Ces améliorations permettent à RoBERTa d'atteindre de meilleures performances que BERT sur plusieurs benchmarks tout en conservant la même architecture de base.

Différences entre base et large :

- BERT-base et RoBERTa-base ont respectivement 110 et 125 millions de paramètres.

- BERT-large et RoBERTa-large en comptent respectivement 340 et 355 millions.

Les versions large tendent à surpasser les versions base sur les benchmarks de NLP, en raison de leur capacité à modéliser des dépendances plus complexes entre les mots et les phrases. Cependant, elles nécessitent plus de ressources et n'offrent pas systématiquement de meilleurs résultats sur certaines tâches où les versions base sont déjà performantes.

Sources :

- BERT : Devlin et al., "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding" (2018)

<https://arxiv.org/abs/1810.04805>

- RoBERTa : Liu et al., "RoBERTa: A Robustly Optimized BERT Pretraining Approach" (2019)

<https://arxiv.org/abs/1907.11692>



**III) La modélisation**

Prétraitement des données :

Cette étape a été effectuée lors du précédent projet, les descriptions ont été nettoyées en amont de l'entraînement des modèles. Cela inclut le retrait des stopwords, de la ponctuation, ainsi que des majuscules. La partie tokenisation quant à elle est inclue dans le modèle.

Extraction des features :

C'est ici que les modèles BERT et RoBERTa entrent en jeu, pour extraire des représentations vectorielles des descriptions. Ces modèles, qui fonctionnent sur des architectures Transformer, génèrent des embeddings pour chaque token. Toutefois, plutôt que d'utiliser les embeddings individuels, j'ai choisi de les agrégés pour obtenir une représentation unique du document.

Pour cela, j'ai opté pour une méthode appelée "mean pooling", qui consiste à calculer la moyenne des embeddings de tous les mots présents dans une description. Cette approche permet de condenser l'information contenue dans l'ensemble du texte tout en conservant des caractéristiques essentielles pour la classification.

| Model | Shape | Duration |
| --- | --- | --- |
| BERT\_base | (1050, 768) | 0:00:19 |
| BERT\_large | (1050, 1024) | 0:00:37 |
| RoBERTa\_base | (1050, 768) | 0:00:18 |
| RoBERTa\_large | (1050, 1024) | 0:00:50 |

Réduction de dimensionnalité et démarche d'optimisation :

Une fois les features extraites, nous avons des jeux de données à 768 ou 1024 colonnes, impossible à visualiser sans réduire la dimensionnalité. En vue d'une projection en 2 dimensions, je souhaite utiliser le t-SNE. Seulement le t-SNE prend plusieurs hyperparamètres, aussi a-t-il fallu tester des valeurs différentes afin de retenir la meilleure combinaison possible. Pour ce faire, je me suis appuyé sur l'index de Davies-Bouldin et le Silhouette Score, qui évaluent la qualité des clusters.

Métrique d'évaluation :

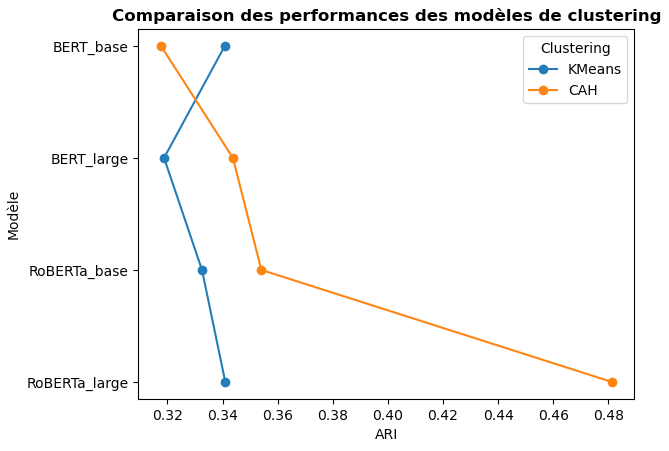
Afin de mesurer la qualité des prédictions de nos modèles, j'ai décidé de m'appuyer sur l'ARI. Il s'agit d'une métrique de similarité utilisée pour évaluer la qualité de clusters en comparant les résultats d'une prédiction avec les catégories réelles. Il varie entre -1 et 1 : Un score de 1 signifie une identité parfaite entre les clusters prédits et clusters réels, un score de 0 indique qu'il n'y a pas de similarité entre eux, un score négatif suggère une structure de clustering qui est pire que le hasard.

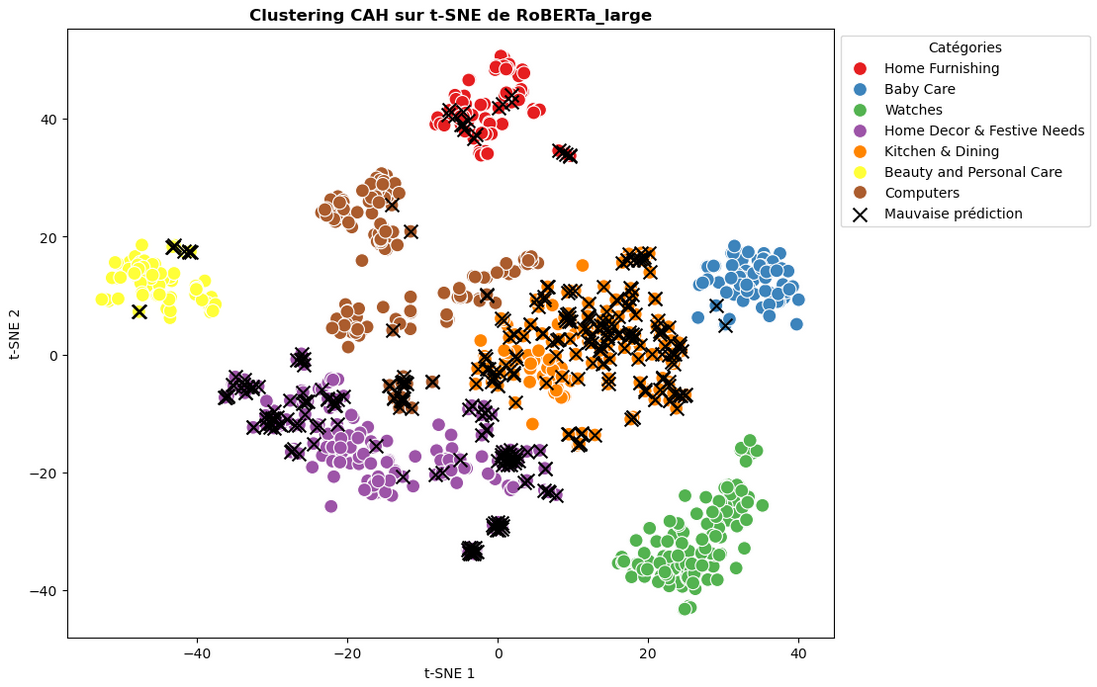
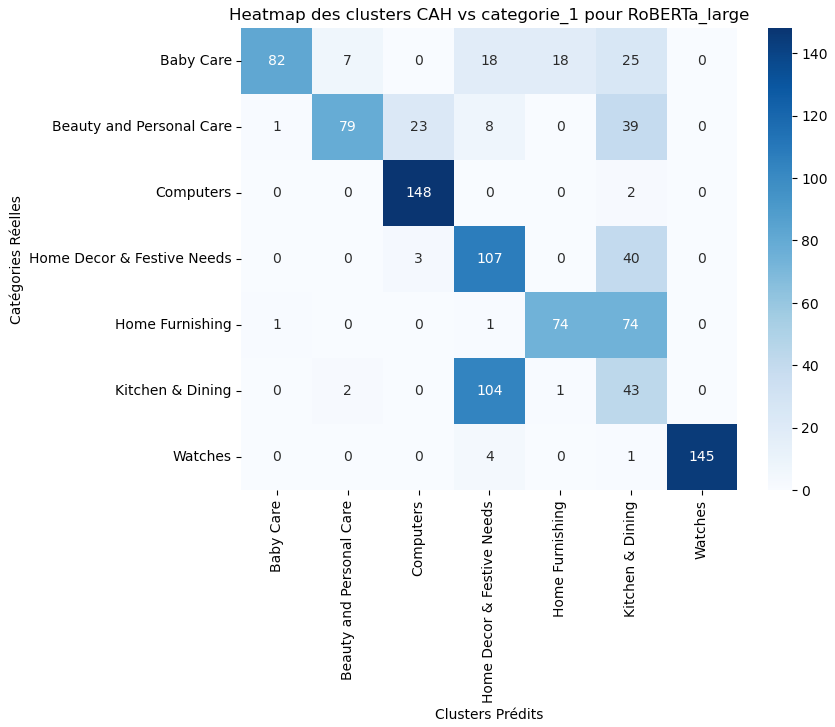
Entraînement des modèles de clustering :

Deux modèles de clustering ont ensuite été entraînés à partir des résultats du t-SNE plutôt que des features extraites directement. En effet, l’application du t-SNE avant l’entraînement de modèles de clustering offre une représentation plus compacte et plus exploitable des features, ce qui réduit le risque de sur-apprentissage ou de mauvaise segmentation lié à une trop grande complexité.

Le premier modèle est un Kmeans, mais face à ses résultats décevants j'ai décidé d'en tester un deuxième, à savoir un CAH (Clustering Hiérarchique Ascendant).

**IV) Une synthèse des résultats**

Comme on peut le constater ici, le score ARI de RoBERTa\_large avec CAH écrase la compétition. Par souci de place, le reste de cette section se focalisera sur ce modèle (et classificateur).

Malgré cela, il reste énormément d'erreurs, et certaines classes plus que d'autres sont extrêmement mal prédites.

Une image contenant texte, capture d’écran, Police, menu

Description générée automatiquement

**V) L’analyse de la feature importance globale et locale du nouveau modèle**

L’analyse de la feature importance vise à comprendre l'influence de chaque variable sur les décisions prises par le modèle. Elle peut être réalisée à deux niveaux :

* Importance globale – Permet d'identifier quelles features ont le plus d'impact sur l'ensemble des prédictions du modèle.
* Importance locale – Permet d'expliquer pourquoi une prédiction spécifique a été faite en analysant l'influence des features pour un individu donné.

Des librairies comme SHAP ou LIME sont souvent utilisées pour cette analyse.

Dans ce projet, nous utilisons un pipeline où les embeddings de modèles de langage sont directement réduits avec t-SNE avant d'être utilisés pour un clustering. Cette structure rend difficle, voire non pertinente l'interprétation classique de feature importance, pour plusieurs raisons :

* Les embeddings sont des représentations denses et non interprétables Contrairement à des variables tabulaires classiques, chaque dimension d’un embedding agrège plusieurs aspects du texte de manière non triviale. Il est donc difficile d’attribuer une importance claire à une dimension spécifique.
* Le t-SNE ne conserve pas l’espace original des features : Il projette les données dans un nouvel espace à basse dimension en préservant les relations de proximité, mais de manière non linéaire. Par conséquent, toute analyse d’importance des features effectuée après cette transformation ne refléterait pas directement l’influence des embeddings initiaux.
* Le clustering est basé sur ces projections : Les classificateurs travaillent sur les données réduites, et non sur les features brutes. Analyser l’importance des features sur ces nouvelles dimensions n’aurait pas de sens, car elles ne correspondent pas directement aux dimensions des embeddings d’origine.

En résumé, bien que l’analyse d’importance des features soit essentielle dans de nombreux modèles supervisés, elle ne s’applique pas directement ici.

**VI) Les limites et les améliorations possibles**

Dans le cadre de ce projet, plusieurs axes d’amélioration pourraient être envisagés :

* Taille et qualité des données - Le dataset utilisé est relativement petit pour une tâche de classification. En général, plus de données permettent une meilleure généralisation des modèles, ce qui est crucial pour la robustesse des résultats. De plus, la qualité des données peut constituer une contrainte supplémentaire. En dépit du nettoyage des données effectué, des erreurs de typographie, des incohérences grammaticales ou des termes ambigus peuvent encore affecter la qualité des informations, et donc des modèles entraînés. Ce problème pourrait être atténué en appliquant des techniques de nettoyage plus avancées, telles que la correction automatique des fautes d’orthographe ou l’harmonisation de la grammaire.
* Enrichissement des données - Un levier potentiel pour améliorer la performance des modèles serait d’ajouter des informations supplémentaires, telles que la marque des produits, des avis utilisateurs, ou des descriptions de produits similaires provenant d’autres plateformes. Cela permettrait aux modèles d’appréhender une plus grande diversité de facteurs influençant les caractéristiques des produits.
* Approche multimodale - Un autre levier serait l’intégration d’informations provenant d’images ou de vidéos associées aux produits. Un modèle qui combine des données textuelles et visuelles pourrait améliorer les résultats de clustering.
* Fine-tuning avec RoBERTa – Cela consisterait (par exemple) à ajuster les couches finales du modèle pré-entraîné pour qu’il puisse mieux s’adapter aux spécificités de notre dataset. Les premières couches qui capturent des caractéristiques de bas niveau (comme la syntaxe du langage), seraient réutilisées, tandis que les couches finales pourraient être entraînées spécifiquement pour la tâche à accomplir.
* Exploration de modèles supervisés pour le clustering - Des algorithmes comme Random Forest, Support Vector Machines (SVM) ou encore les réseaux de neurones multicouches (MLP) pourraient offrir des résultats plus fins. Ces méthodes supervisées pourraient notamment mieux capturer des relations complexes et aider à une meilleure segmentation des données.